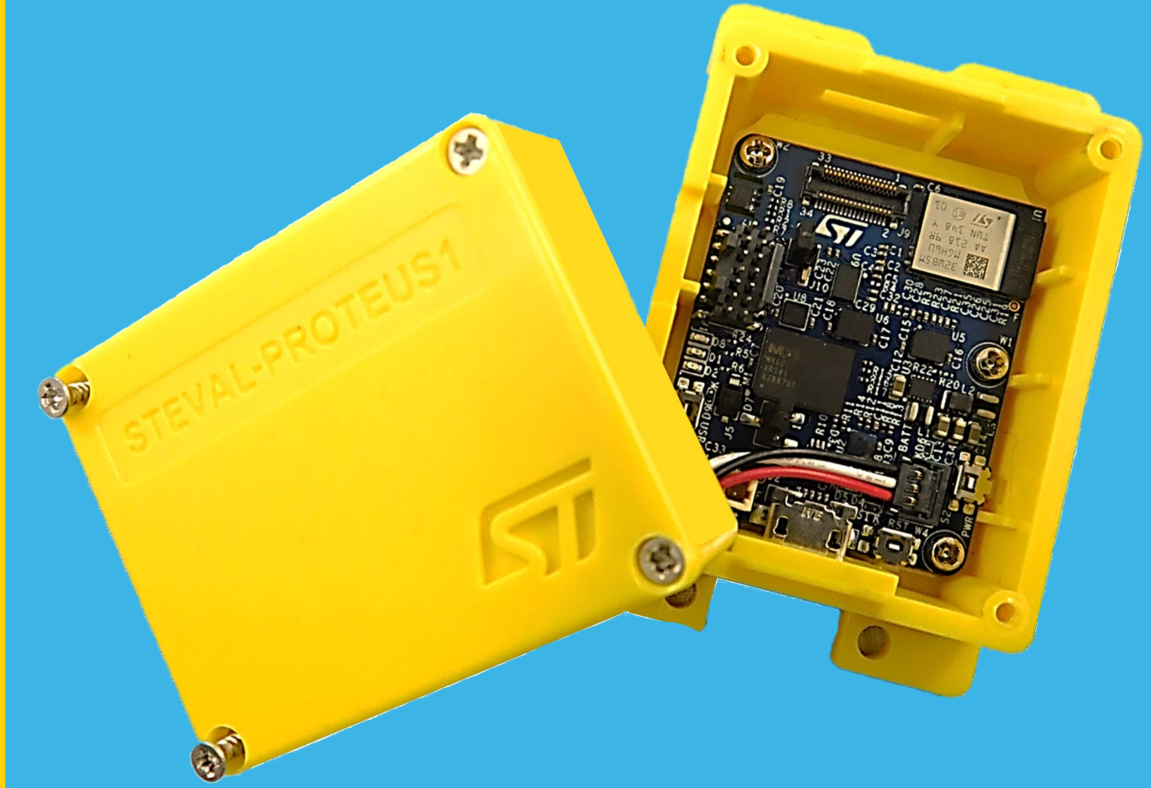
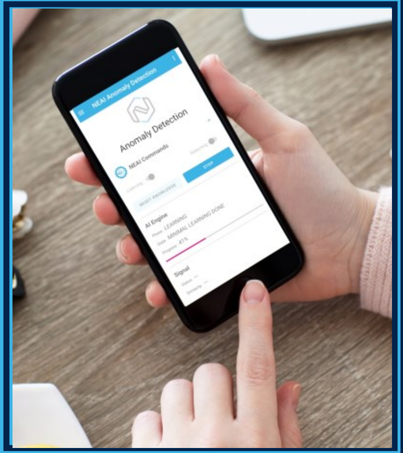




Quick Start Guide

STM32Cube function pack for STEVAL-PROTEUS1 evaluation kit for predictive maintenance application based on artificial intelligence (AI)

(FP-AI-PDMWBSOC)

Version 2.1 (15, Jun 2023)

Agenda

1 Hardware and Software overview

2 Setup & Demo Examples

3 Documents & Related Resources

4 STM32 Open Development Environment: Overview

1- Hardware and Software overview

life.augmented

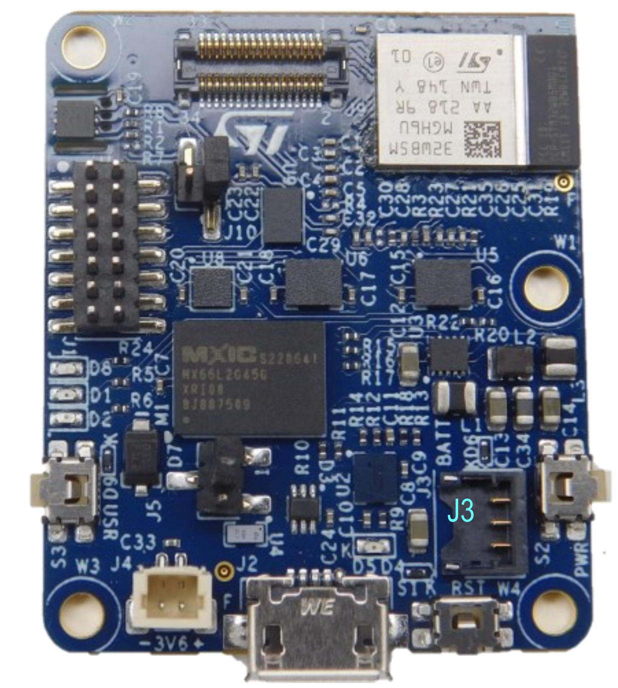
STEVAL – PROTEUS1
Hardware Overview

Industrial sensor evaluation kit for condition monitoring based on 2.4 GHz STM32WB5MMG module

The STEVAL-PROTEUS1 is an evaluation tool designed for temperature and vibration monitoring, based on a 2.4 GHz multiprotocol wireless SoC to address machine or facility condition monitoring for industrial applications. All components are mounted exclusively on the top side of the PCB to ensure an easy mounting on other equipment.

Key Features

- Kit content: the **STEVAL-PROTEUS** main board, LiPo battery 3.7 V, 480 mAh, plastic case and screws
- **STEVAL-PROTEUS**: STM32WB5MMG - ultra-low-power module, dual core 32-bit Arm Cortex-M4 MCU 64 MHz, Cortex-M0+ 32 MHz for real-time radio layer, with 1 Mbyte of flash memory, 256kbyte SRAM, and 2.4GHz RF supporting Bluetooth® Low Energy 5, 802.15.4, Zigbee 3.0, and Thread
- **IIS3DWB** - ultra-wide bandwidth up to 6 kHz, low noise, 3-axis digital accelerometer
- **ISM330DHCX** - iNEMO inertial module with machine learning core and finite state machine with digital output
- **IIS2DLPC** - high-performance ultra-low-power 3-axis digital accelerometer
- **STTS22H** - low-voltage, ultra-low-power, 0.5°C accuracy I²C/SMBus 3.0 temperature sensor
- Memory & Secure: 2Gb QSPI NOR flash memory for data storage, STSAFE-A110 - secure element
- Power: **STBC02** - Li-Ion linear battery charger with LDO, **ST1PS02** - step-down converter with digital voltage selection
- HMI: 3 push-buttons (Reset, User, Power-on with battery), 4 LEDs (three user LEDs, one STBC02 LED status)
- Flexible power supply options - LiPo battery, USB power, and primary battery
- Connectors: SWD connector for debugging and programming capability, 34-pin expansion connector compliant with STMOD+



Latest info available at:
https://www.st.com/en/evaluation-tools/steval-proteus1.html

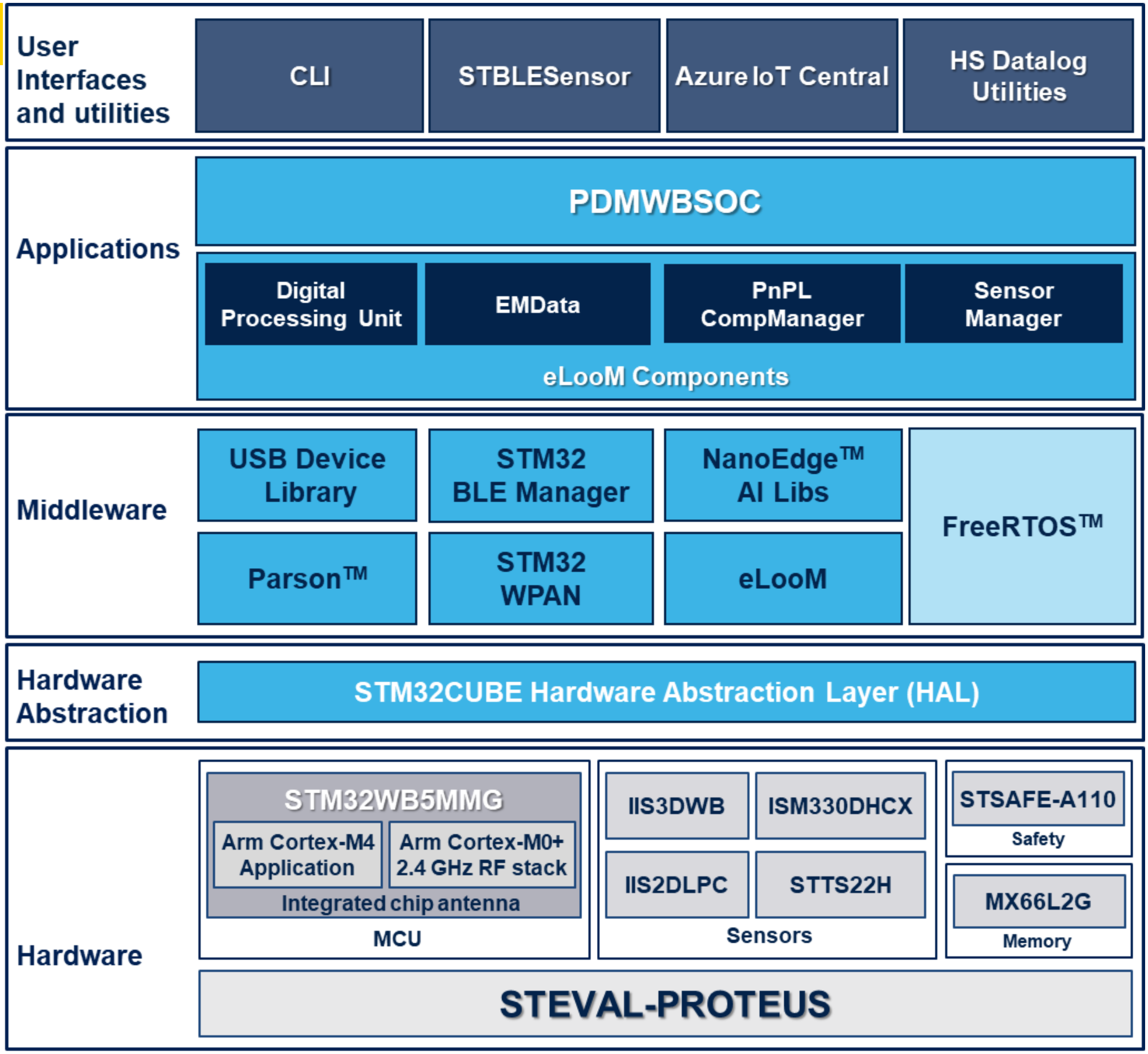
Contains:
FCC ID: YCP-STM32WB5M001
IC: 8976A-STM32WB5M01

life.augmented

FP-AI-PDMWBSOC
FW Architecture

Key Features

- Complete firmware to acquire motion sensor data, process them for Anomaly Detection or N-Class classification, in order to send the results to the **STBLESensClassic** mobile app and/or a PC terminal console.
- Embedded software, middleware and drivers
  - FreeRTOS
  - eLooM to enable modularity and code re-usability at application level
  - STM32 WPAN
  - NanoEdgeAI (NEAI) Anomaly Detection and N-Classification compiled libraries
- Compatible with **NanoEdge™ AI Studio** to enable AI-based solution
- BLE application compatible with **STBLESensClassic** (Android, iOS) for the following tasks:
  - Control and monitor the NEAI libraries execution
  - Change the application libraries properties using a new dedicated setting page
  - Bridging data to Microsoft Azure IoT Central Cloud
  - Firmware upgrade via FUOTA.
- NanoEdgeAI Anomaly Detection model storage (on embedded flash memory)
- Utilities folder including:
  - Bootloader to allow the BLE FUOTA
  - HS_Datalog with FUOTA capability in binary format
  - Batch files to launch python scripts to run DATALOG application and to convert datasets
  - BLE stack (Binary) for STM32WB coprocessor.



User Interfaces and utilities: CLI | STBLESensor | Azure IoT Central | HS Datalog Utilities

Applications: PDMWBSOC

Digital Processing Unit | EMData | PnPL CompManager | Sensor Manager

eLooM Components

Middleware: USB Device Library | STM32 BLE Manager | NanoEdge™ AI Libs | FreeRTOS™ | Parson™ | STM32 WPAN | eLooM

Hardware Abstraction: STM32CUBE Hardware Abstraction Layer (HAL)

Hardware:

MCU: STM32WB5MMG — Arm Cortex-M4 Application | Arm Cortex-M0+ 2.4 GHz RF stack — Integrated chip antenna

Sensors: IIS3DWB | ISM330DHCX | IIS2DLPC | STTS22H

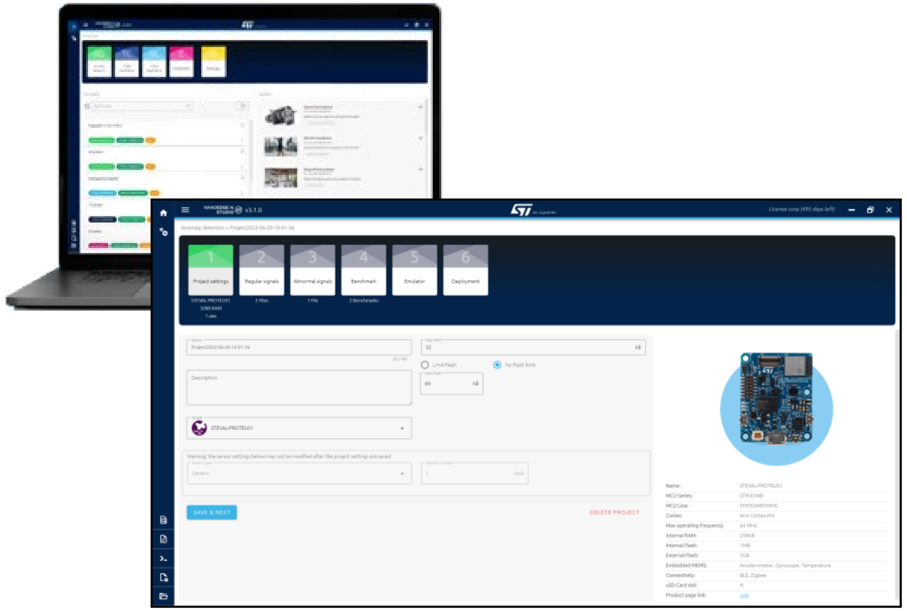
Safety: STSAFE-A110

Memory: MX66L2G

STEVAL-PROTEUS

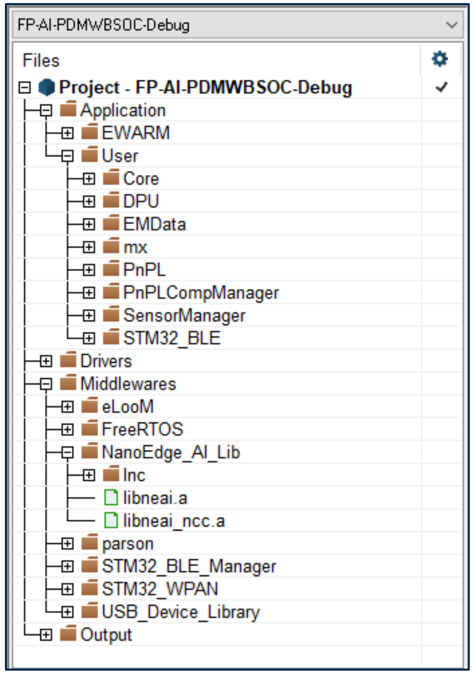
FP-AI-PDMWBSOC
Ecosystem

Complete turnkey solution from Datalog to Anomaly Detection and Classification status on Mobile App and up to Cloud



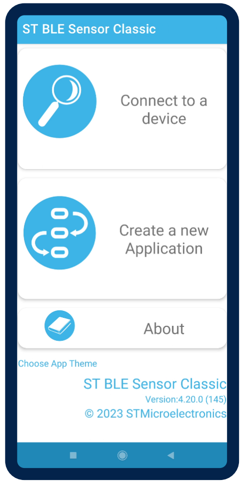
STEVAL-PROTEUS1 supported
in NanoEdgeAIStudio **[v3.3.2]**

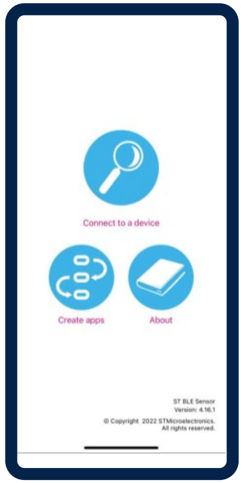
NANOEDGE AI STUDIO



```
FP-AI-PDMWBSOC-Debug
Files
⊟ Project - FP-AI-PDMWBSOC-Debug
  ⊟ Application
    ⊞ EWARM
    ⊟ User
      ⊞ Core
      ⊞ DPU
      ⊞ EMData
      ⊞ mx
      ⊞ PnPL
      ⊞ PnPLCompManager
      ⊞ SensorManager
      ⊞ STM32_BLE
  ⊞ Drivers
  ⊟ Middlewares
    ⊞ eLooM
    ⊞ FreeRTOS
    ⊟ NanoEdge_AI_Lib
      ⊞ Inc
        libneai.a
        libneai_ncc.a
    ⊞ parson
    ⊞ STM32_BLE_Manager
    ⊞ STM32_WPAN
    ⊞ USB_Device_Library
  ⊞ Output
```

KEIL Tools by ARM IAR SYSTEMS STM32 CubeIDE





Azure
IoT Central

2- Setup & Demo Examples

Setup & Demo Examples

Software prerequisites

- **STM32CubeProgrammer Software**
  - Download and install STM32CubeProgrammer
- **FP-AI-PDMWBSOC**
  - Copy the .zip file content into a folder on your PC. The package will contain source code example (Keil, IAR, STM32CubeIDE) based on **STEVAL-PROTEUS**
- **ST BLE Sensor Classic**
  - Application for Android (from v4.20) / iOS (from v4.20) to download from Play Store / App Store

STEVAL-PROTEUS1 kit is not preprogrammed with **FP-AI-PDMWBSOC**
To update the firmware, please follow the instructions available in slide 14-17

Setup & Demo Examples
Hardware prerequisites

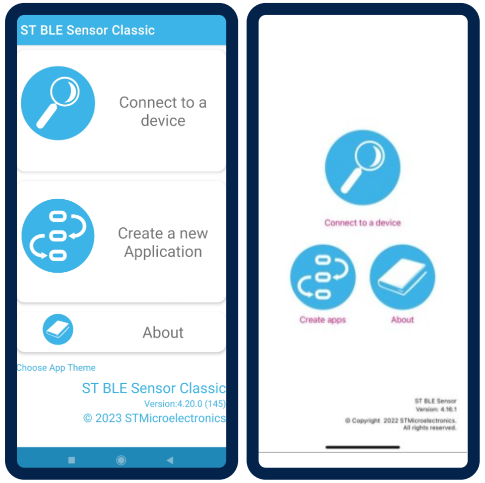
Recommended

- 1 STEVAL-PROTEUS1 evaluation kit
- 1 Laptop/PC with Windows 7, 8 or 10
- 1 USB micro-B cable to supply the sensor-board by PC
- 1 smartphone with ST BLE Sensor Classic App (Android or IOS)

Optional (just for debugging and programming)

- 1 STLINK-V3MINI (or V3MINIE)
- 1 USB micro-B (or Type-C) cable to connect the STLINK-V3MINI (or V3MINIE)

STEVAL-PROTEUS



ST BLE Sensor Classic

Connect to a device

Create a new Application

About

Choose App Theme

ST BLE Sensor Classic
Version:4.20.0 (145)
© 2023 STMicroelectronics

Connect to a device

Create apps

About

ST BLE Sensor
Version: 4.16.1
© Copyright 2022 STMicroelectronics.
All rights reserved.

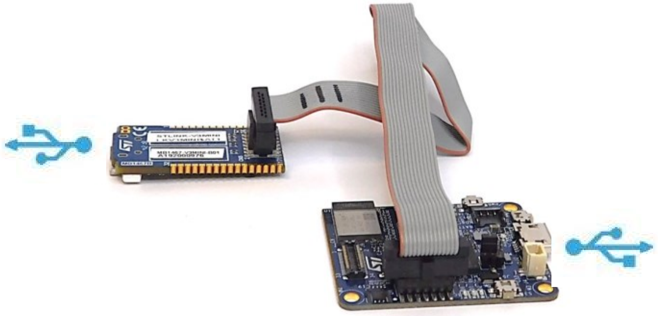
ST BLE Sensor Classic

USB micro-B

USB Type-C

STLINK-V3MINI

STLINK-V3MINIE



2.1- Setup Overview

STEVAL-PROTEUS1

Unboxing

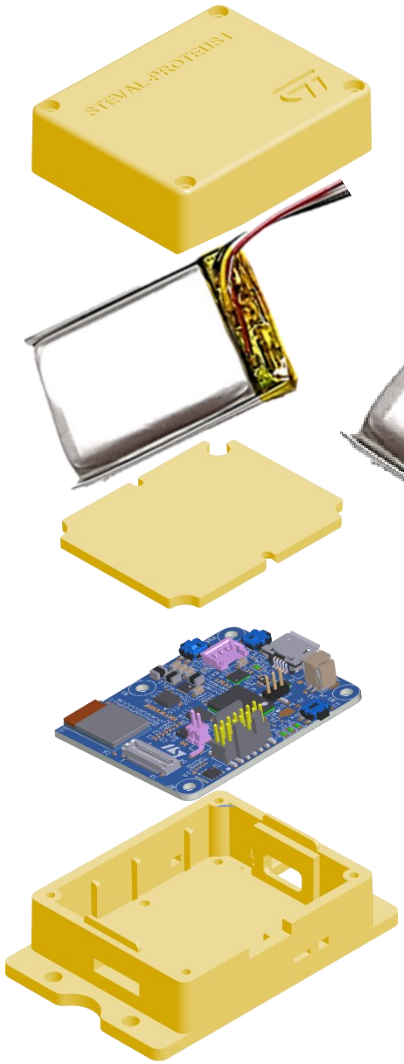
5
Lock the top case to the bottom one with the last four screws included in the kit.

4
Plug the battery connector on J3

3
Put the cover on the battery and close it using two screws.

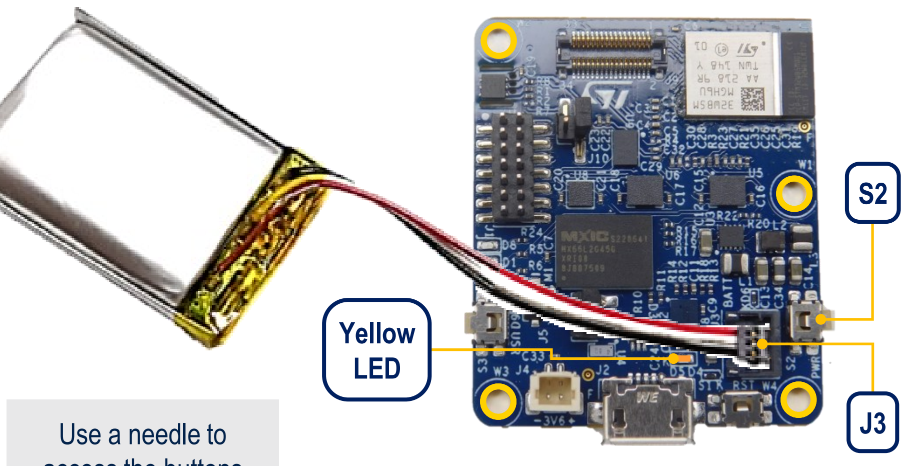
2
Put the Li-Po battery in the top case, insert the battery cable into the dedicated hole.

1
Fix the main board to the case bottom with the four screws included in the kit.



Use a needle to access the buttons behind the holes on the sides of the case.

Power ON
Push and hold S2 until the yellow LED turns on (HW Featrure)



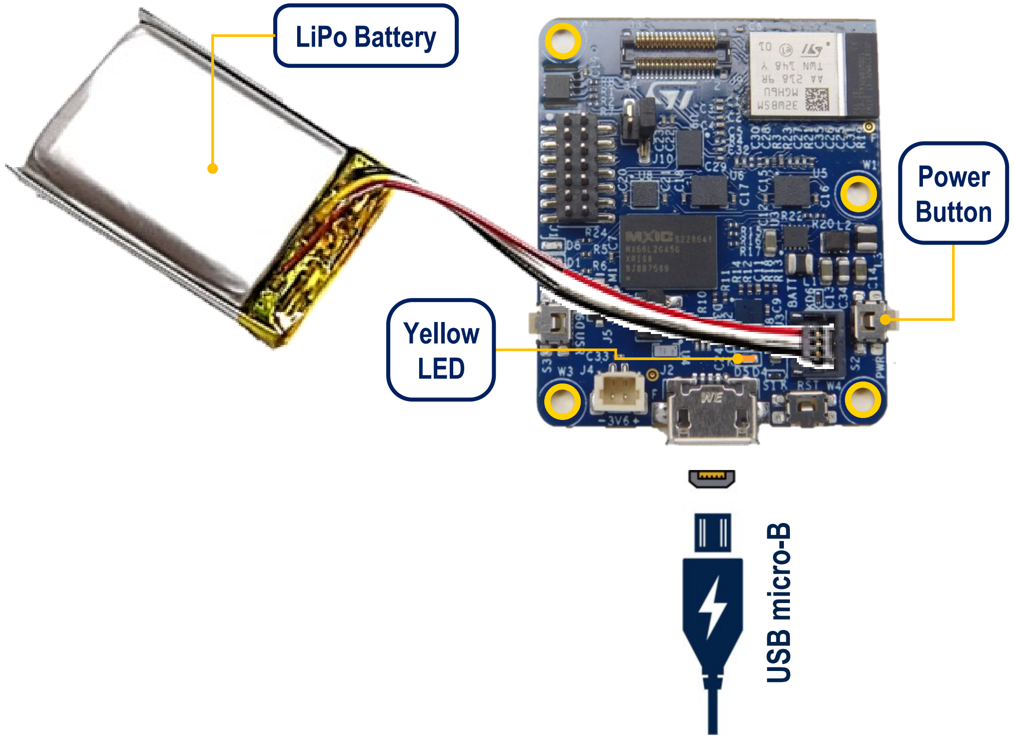
Power OFF
Push and hold S2 until the Yellow LED turns off (SW feature)

STEVAL-PROTEUS1

Power ON/OFF

- **Battery operated only** (no USB cable):
  - **Power ON**: push and hold the power button until the yellow LED turns on (~3 sec).
  - **Power OFF**: push and hold the power button until the yellow LED turns off (~3 sec).

- **Plugged mode** (USB cable)
  - **Power ON**: when USB is plugged-in, the STEVAL-PROTEUS is always on. It doesn’t matter if the battery is present or not.
  - **Power OFF:** unplug the USB cable and, if the battery is connected, act as described above.



life.augmented

STEVAL-PROTEUS Setup

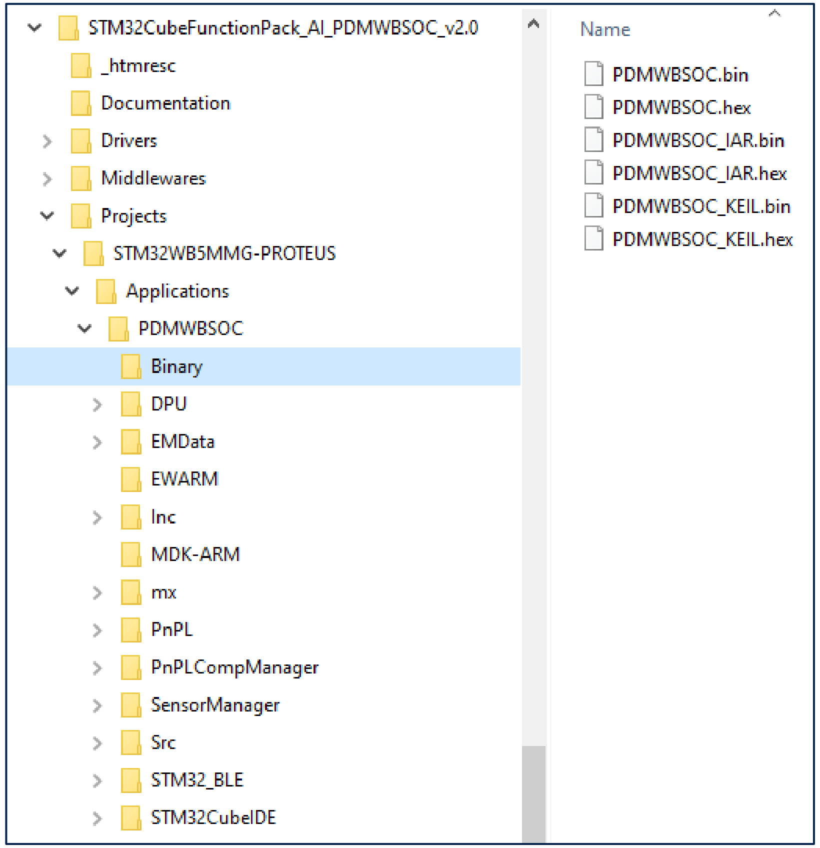
Firmware update

STEVAL-PROTEUS evaluation board is pre-programmed with another default application so, it must be update downloading the FP-AI-PDMWBSOC application.

The easiest way is to use the **pre-compiled binary** provided in the package in the following folder:
Projects\STM32WB5MMG-PROTEUS\Applications\PDMWBSOC\Binary

To update the firmware the user can choose one of the following procedure:

- Save the same binary file (in *.bin format) in your mobile device and upgrade firmware by FUOTA using ST BLE Sensor Classic Mobile App.
- Connect the STEVAL-PROTEUS board to the STLINK-V3MINI (or V3MINIE) programmer, and then use the STM32CubeProgrammer tool.



- STM32CubeFunctionPack_AI_PDMWBSOC_v2.0
  - _htmresc
  - Documentation
  - Drivers
  - Middlewares
  - Projects
    - STM32WB5MMG-PROTEUS
      - Applications
        - PDMWBSOC
          - Binary
          - DPU
          - EMData
          - EWARM
          - Inc
          - MDK-ARM
          - mx
          - PnPL
          - PnPLCompManager
          - SensorManager
          - Src
          - STM32_BLE
          - STM32CubeIDE

Name

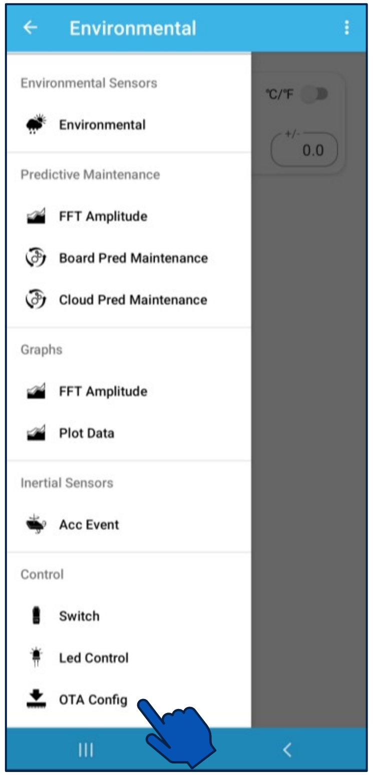
PDMWBSOC.bin
PDMWBSOC.hex
PDMWBSOC_IAR.bin
PDMWBSOC_IAR.hex
PDMWBSOC_KEIL.bin
PDMWBSOC_KEIL.hex

STEVAL-PROTEUS Setup

Firmware update by FUOTA 1/2

How to re-program the STEVAL-PROTEUS by FUOTA:

- Install and launch ST BLE Sensor Classic Mobile App, connect board and follow below steps



Environmental

Environmental Sensors

- Environmental

Predictive Maintenance

- FFT Amplitude
- Board Pred Maintenance
- Cloud Pred Maintenance

Graphs

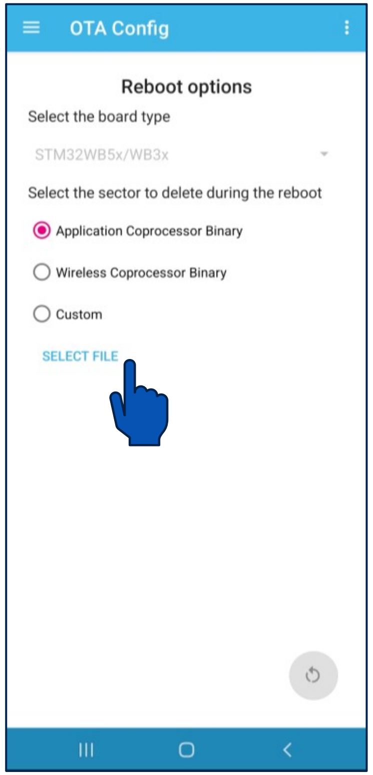
- FFT Amplitude
- Plot Data

Inertial Sensors

- Acc Event

Control

- Switch
- Led Control
- OTA Config



OTA Config

Reboot options

Select the board type

STM32WB5x/WB3x

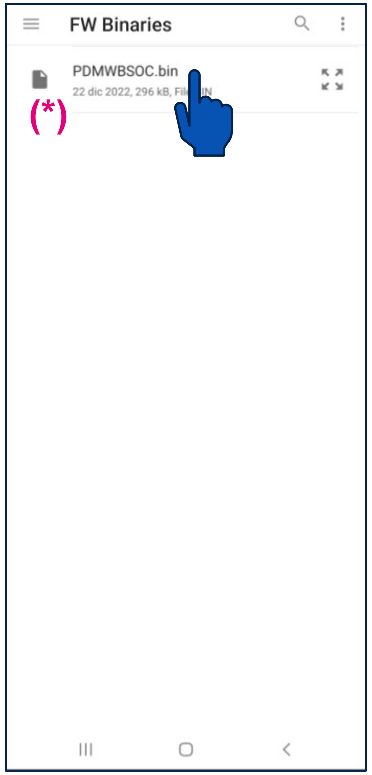
Select the sector to delete during the reboot

● Application Coprocessor Binary

○ Wireless Coprocessor Binary

○ Custom

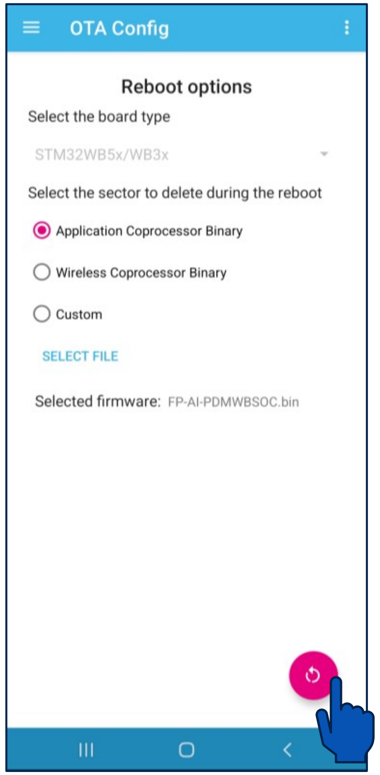
SELECT FILE



FW Binaries

PDMWBSOC.bin
22 dic 2022, 296 kB, File BIN

(*)



OTA Config

Reboot options

Select the board type

STM32WB5x/WB3x

Select the sector to delete during the reboot

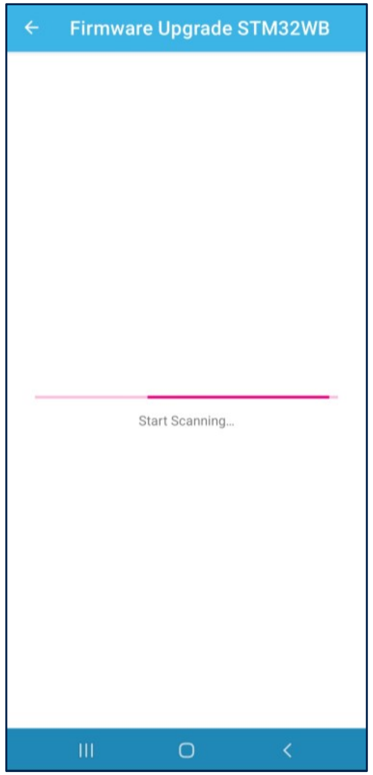
● Application Coprocessor Binary

○ Wireless Coprocessor Binary

○ Custom

SELECT FILE

Selected firmware: FP-AI-PDMWBSOC.bin



Firmware Upgrade STM32WB

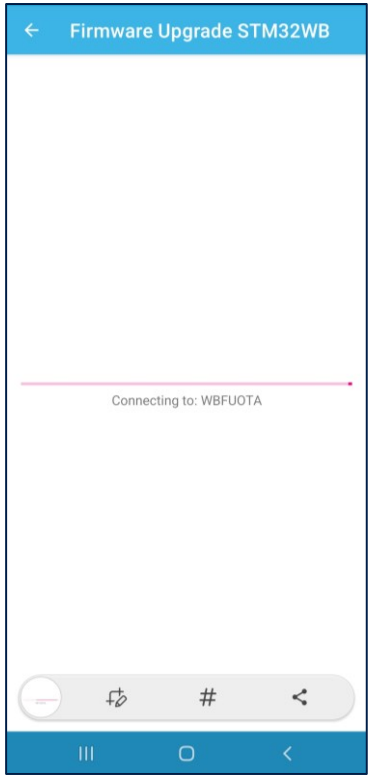
Start Scanning...

(*) Download in your smartphone the binary file from FP package

STEVAL-PROTEUS Setup

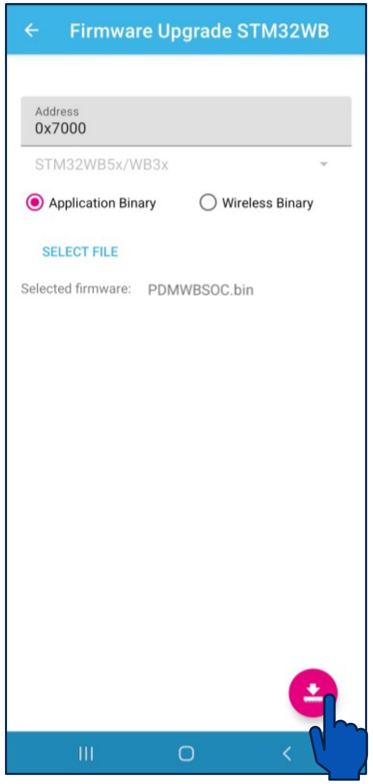
Firmware update by FUOTA 2/2

- How to re-program the STEVAL-PROTEUS by FUOTA



Firmware Upgrade STM32WB

Connecting to: WBFUOTA



Firmware Upgrade STM32WB

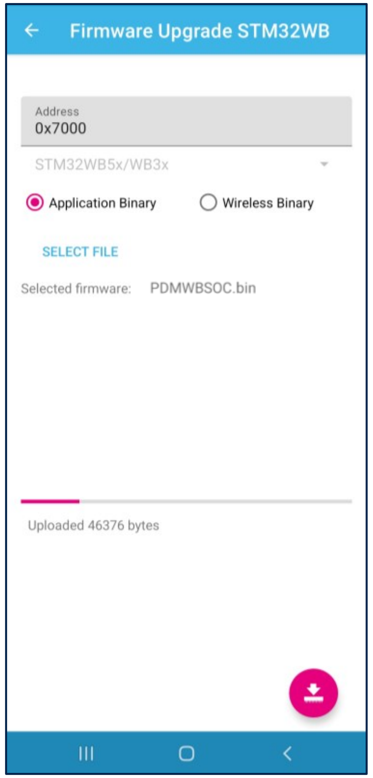
Address
0x7000

STM32WB5x/WB3x

● Application Binary ○ Wireless Binary

SELECT FILE

Selected firmware: PDMWBSOC.bin



Firmware Upgrade STM32WB

Address
0x7000

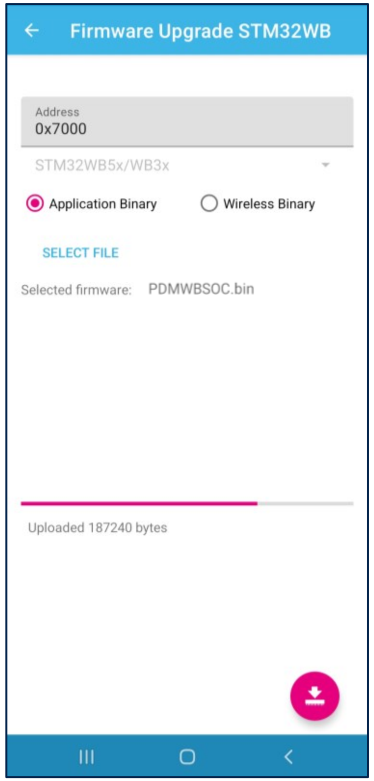
STM32WB5x/WB3x

● Application Binary ○ Wireless Binary

SELECT FILE

Selected firmware: PDMWBSOC.bin

Uploaded 46376 bytes



Firmware Upgrade STM32WB

Address
0x7000

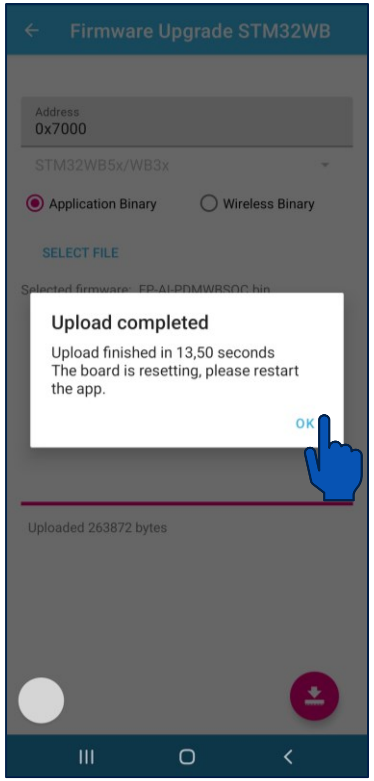
STM32WB5x/WB3x

● Application Binary ○ Wireless Binary

SELECT FILE

Selected firmware: PDMWBSOC.bin

Uploaded 187240 bytes



Firmware Upgrade STM32WB

Address
0x7000

STM32WB5x/WB3x

● Application Binary ○ Wireless Binary

SELECT FILE

Selected firmware: FP-AI-PDMWBSOC.bin

Upload completed

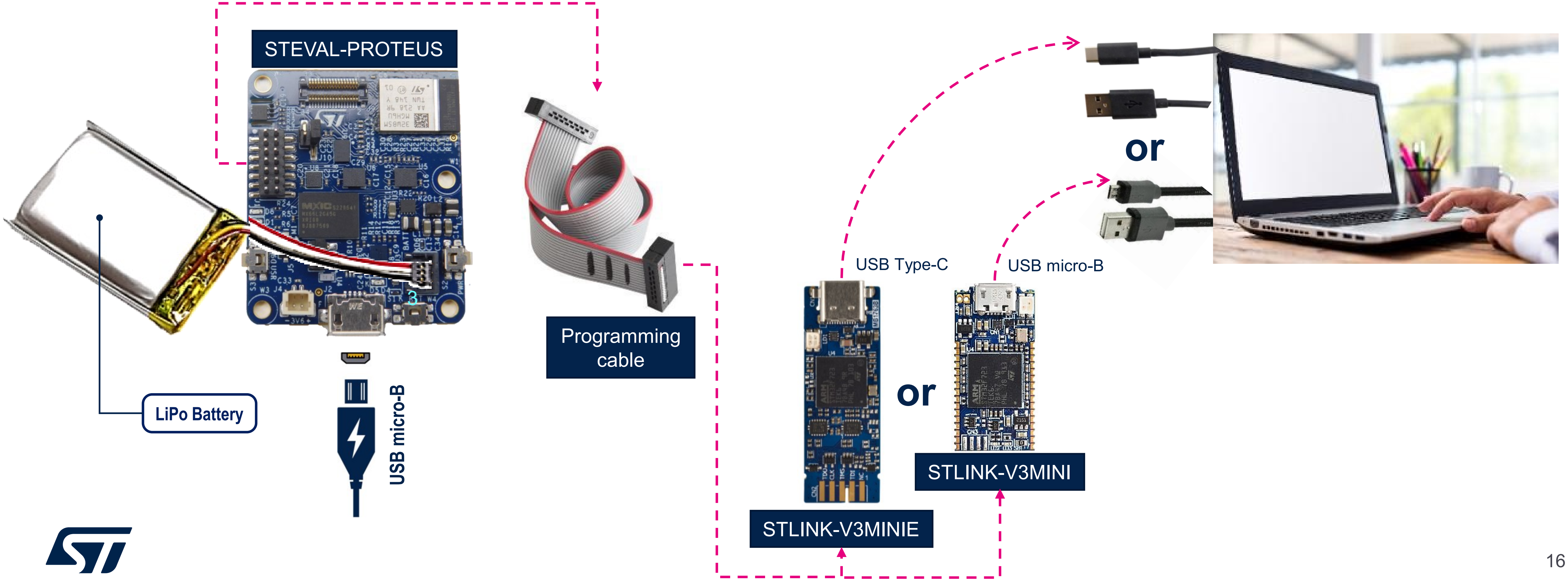
Upload finished in 13,50 seconds
The board is resetting, please restart the app.

OK

Uploaded 263872 bytes

STEVAL-PROTEUS Setup
Firmware update by STLINK

- ❑ Power the board using Battery or USB plug
- ❑ Follow the connection by cables, as shown in the picture below

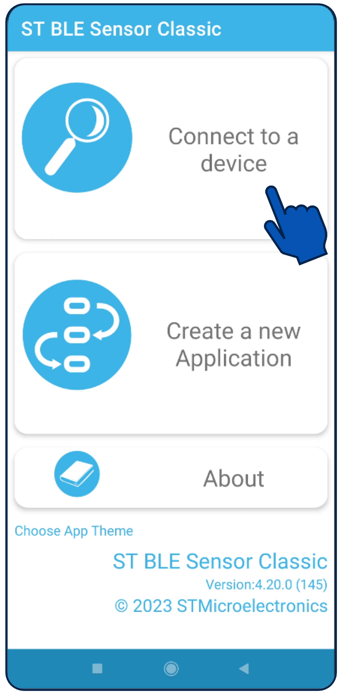


2.2- PDMWBSOC Application:

How to use the ST BLE Sensor Classic App

life.augmented

Discovery View of ST BLE Sensor Classic App



ST BLE Sensor Classic

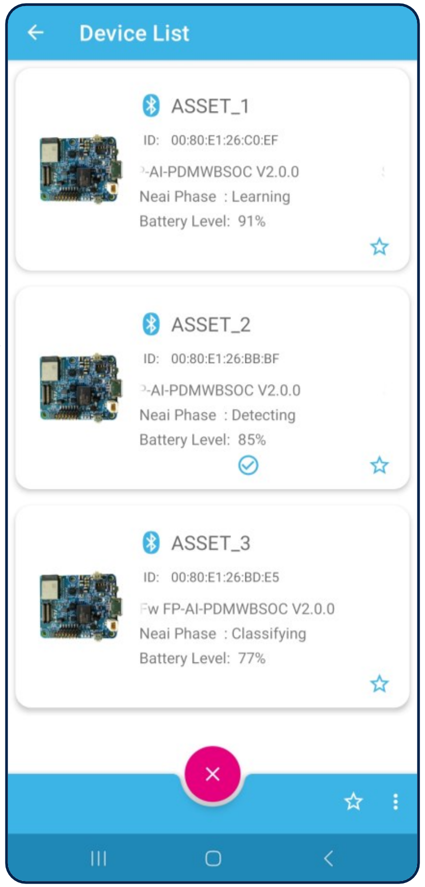
Connect to a device

Create a new Application

About

Choose App Theme

ST BLE Sensor Classic
Version:4.20.0 (145)
© 2023 STMicroelectronics



Device List

ASSET_1
ID: 00:80:E1:26:C0:EF
-AI-PDMWBSOC V2.0.0
Neai Phase : Learning
Battery Level: 91%

ASSET_2
ID: 00:80:E1:26:BB:BF
-AI-PDMWBSOC V2.0.0
Neai Phase : Detecting
Battery Level: 85%

ASSET_3
ID: 00:80:E1:26:BD:E5
w FP-AI-PDMWBSOC V2.0.0
Neai Phase : Classifying
Battery Level: 77%

After tapping on *Connect one Device*, you'll see the list of available boards to which can connect.

For each board are available:

- FW running name (FP-AI-PDMWBSOC)
- NEAI phase (idle, idle trained, learning, detecting, classifying)
- Battery level (0-100%)
- Status Icon, ✓ normal or ⚠ anomaly in case of anomaly detection phase

Until you aren't connected to the board, blue LED blinks slowly

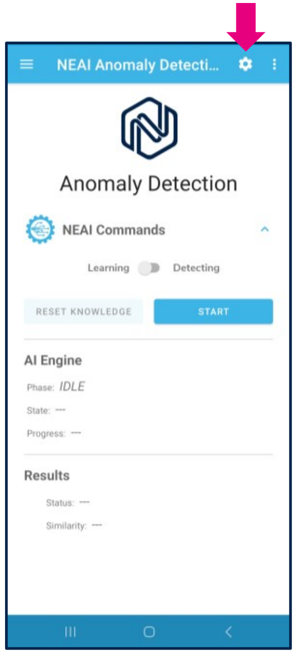
When you are connected to the board, blue LED blinks quickly

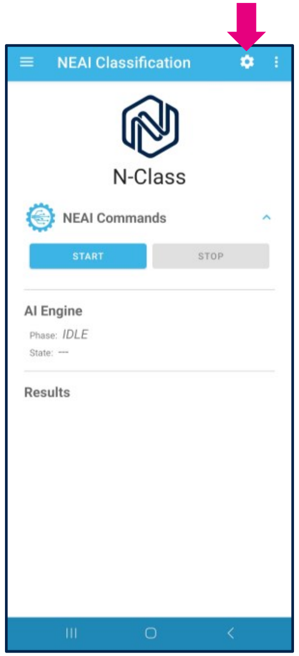
life.augmented

The proper workflow

Open ST BLE Sensor Classic App & Connect Board

Tap on the gear icon to customize AD library parameters and sensor setup (*)

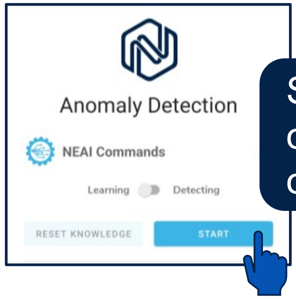




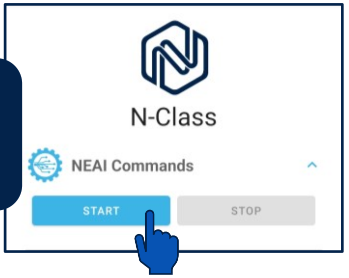
* This step is optional but remember that by default:

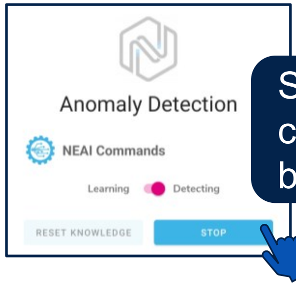
- ISM330DHCX is active with ODR = 3332 Hz and FS = 2 G
- Learning/Detecting phases will end when you'll push stop button (*Time/Signals* parameter is initialized to zero)

It's strongly recommended to setup your sensor according to dataset used to generate NEAI library

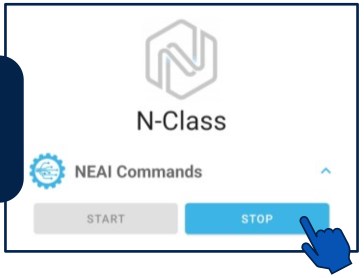


Start learning, detecting or classifying by buttons on NEAI demos

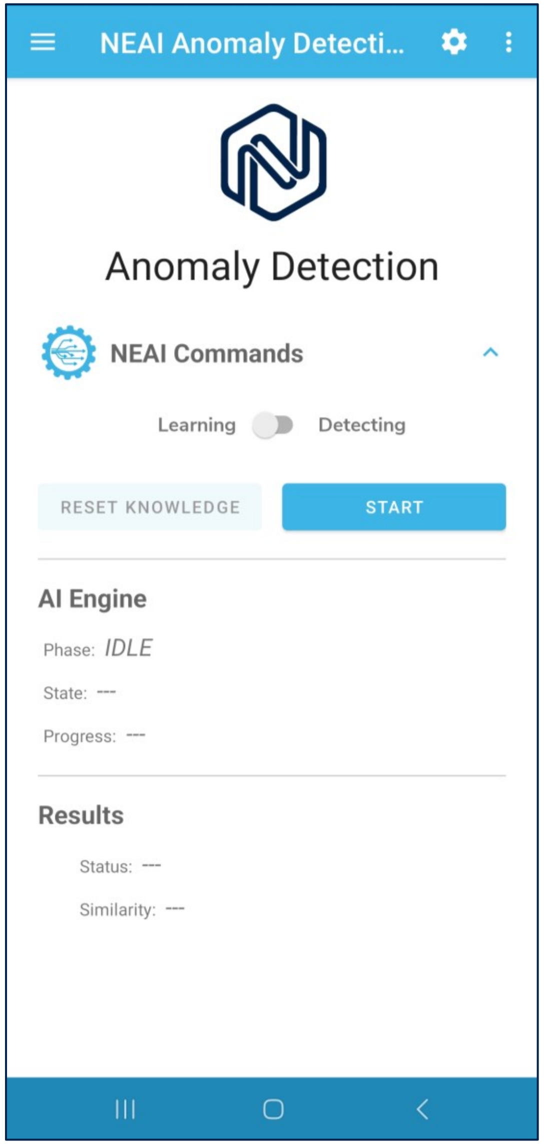




Stop learning, detecting or classifying automatically or by button on NEAI demos



Start learning phase
Three simple steps


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

RESET KNOWLEDGE | START

AI Engine

Phase: IDLE

State: ---

Progress: ---

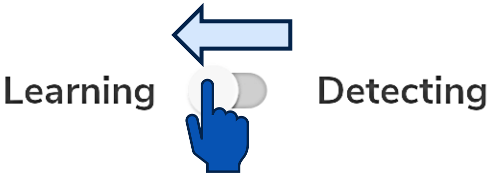
Results

Status: ---

Similarity: ---

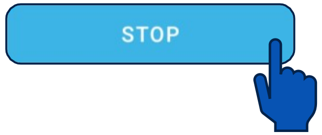
Through this demo you can monitor NanoEdgeAI AD library status and also start/stop learning and detecting phases. To start your first learning follow the steps below:

1. Move the commands switch on the left to enable **Learning**

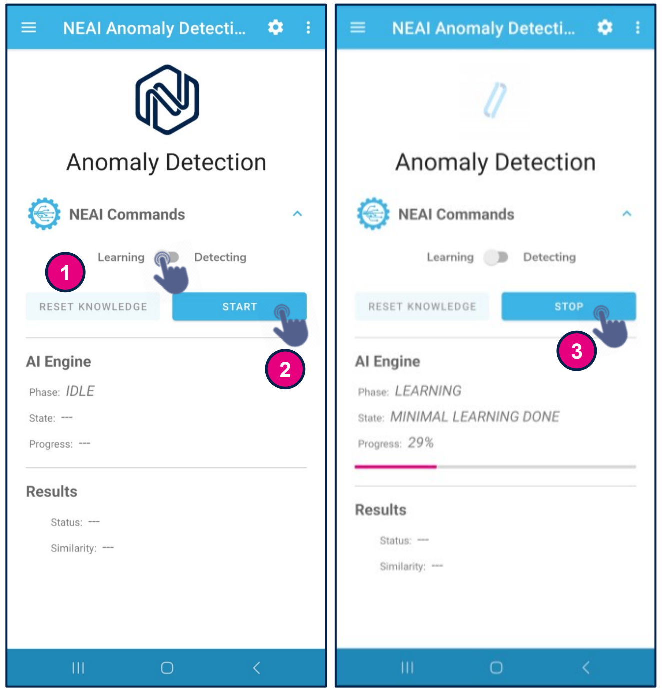


2. Push start button

3. Push stop button when you want.



*(When the processed signals are more than required from the NEAI-AD library itself the state changes in "**MINIMAL LEARNING DONE**")*


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

RESET KNOWLEDGE | START

AI Engine

Phase: IDLE

State: ---

Progress: ---

Results

Status: ---

Similarity: ---

NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

RESET KNOWLEDGE | STOP

AI Engine

Phase: LEARNING

State: MINIMAL LEARNING DONE

Progress: 29%

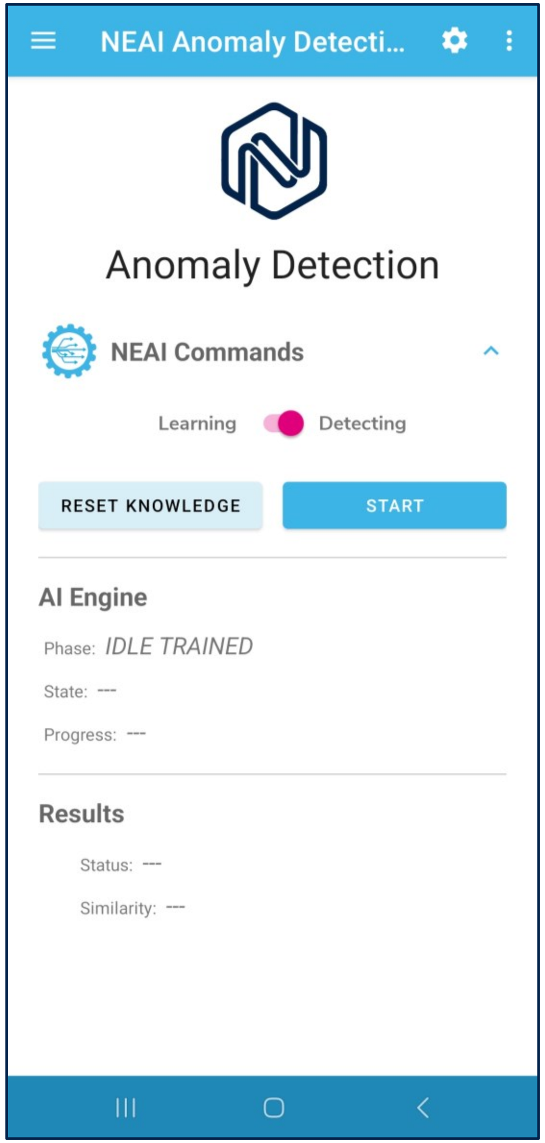
Results

Status: ---

Similarity: ---

Learning started green LED blinks

Start Detecting phase
Three simple steps


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning Detecting

RESET KNOWLEDGE START

AI Engine

Phase: *IDLE TRAINED*

State: ---

Progress: ---

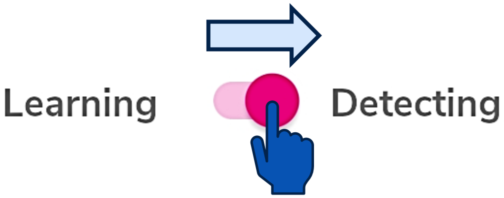
Results

Status: ---

Similarity: ---

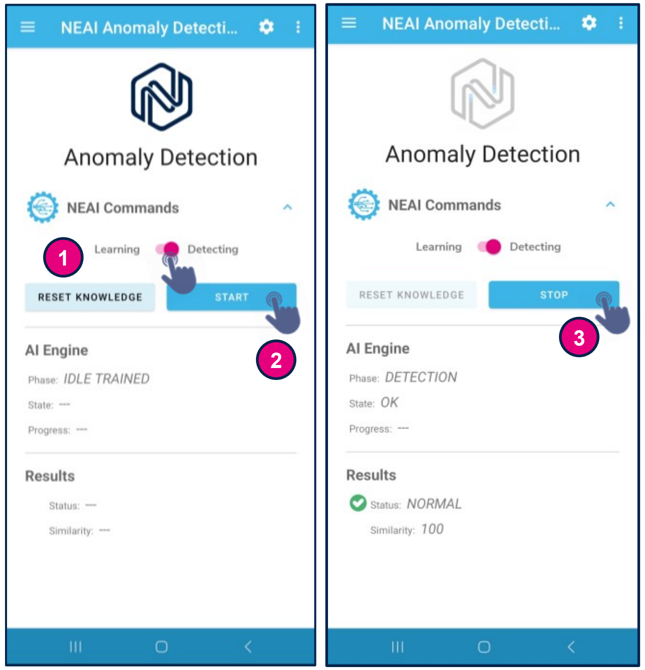
Through this demo you can monitor NanoEdgeAI AD library status and also start/stop learning and detecting phases. To start your first detection, follow the steps below:

1 Move the commands switch on the right to enable **Detection**



2 Push start button

3 Push stop button when you want


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

1 Learning Detecting

RESET KNOWLEDGE START 2

AI Engine

Phase: *IDLE TRAINED*

State: ---

Progress: ---

Results

Status: ---

Similarity: ---

NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning Detecting

RESET KNOWLEDGE STOP 3

AI Engine

Phase: *DETECTION*

State: *OK*

Progress: ---

Results

✓ Status: *NORMAL*

Similarity: *100*

Detection started
Status Normal
green LED ON

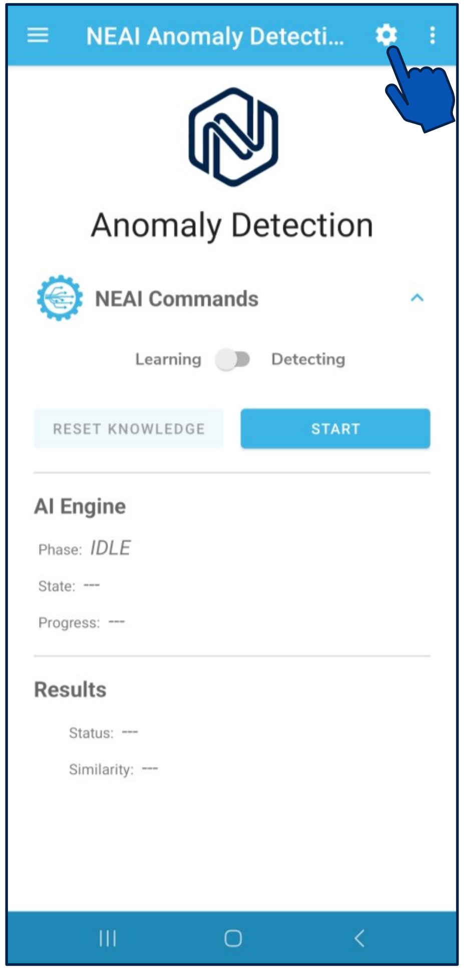
Detection started
Status Abnormal
red LED ON

Remember that detecting phase will end only when you push stop button

life.augmented

How to set Learning phase time 1/2

1



NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning ○ Detecting

RESET KNOWLEDGE | START

AI Engine

Phase: *IDLE*

State: ---

Progress: ---

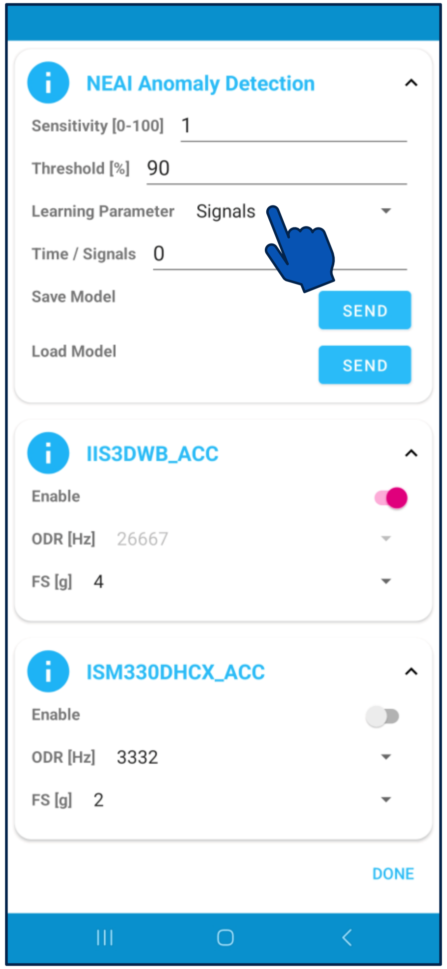
Results

Status: ---

Similarity: ---

First of all tap on gear icon to open the setting page

2



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Signals

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable ●

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

Enable ○

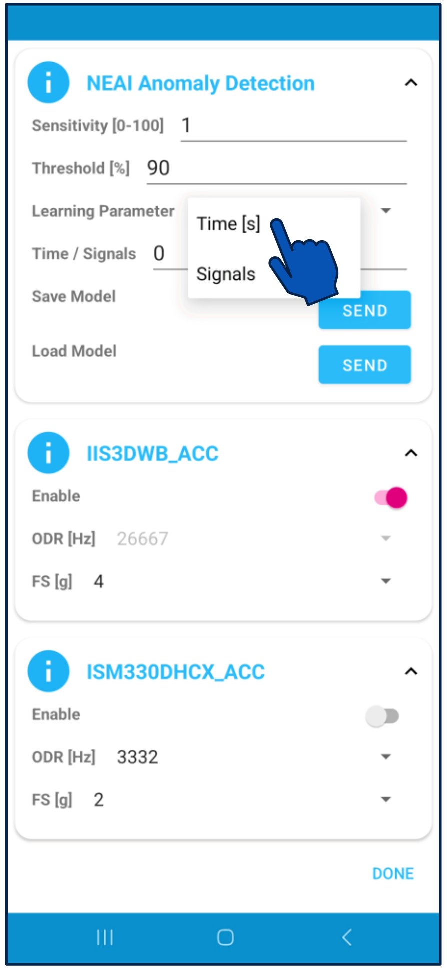
ODR [Hz] 3332

FS [g] 2

DONE

Tap on learning parameter and select *Time [s]* option

3



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter

- Time [s]
- Signals

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable ●

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

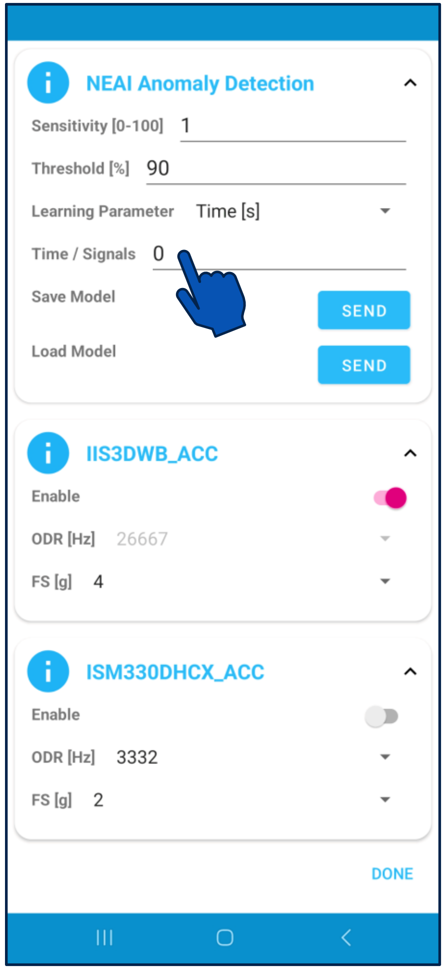
Enable ○

ODR [Hz] 3332

FS [g] 2

DONE

How to set learning phase time 2/2



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

Enable

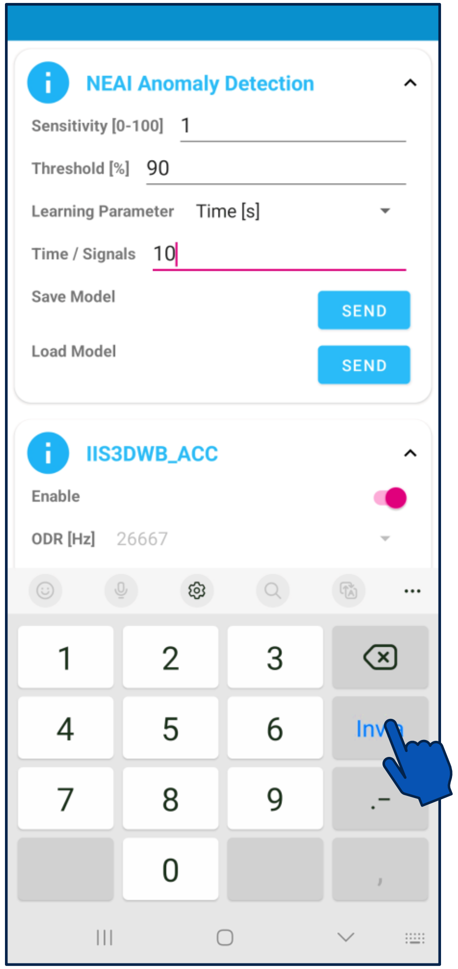
ODR [Hz] 3332

FS [g] 2

DONE

4

Tap on *Time/Signals* parameter



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 10

Save Model SEND

Load Model SEND

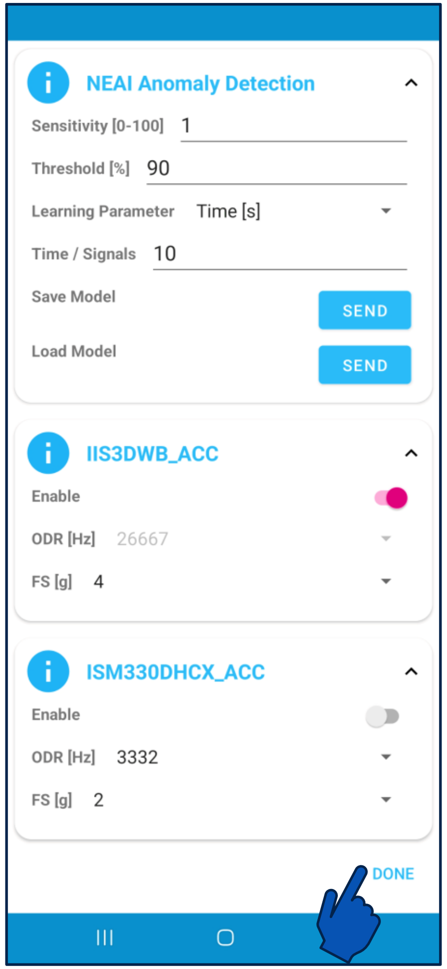
IIS3DWB_ACC

Enable

ODR [Hz] 26667

5

Enter your desired learning duration



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 10

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

Enable

ODR [Hz] 3332

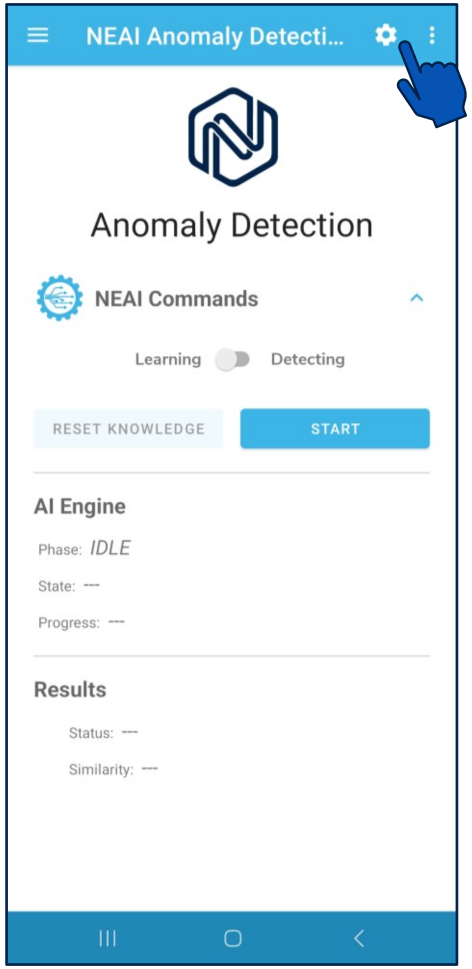
FS [g] 2

DONE

6

END

How to set signals to learn 1/2



NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

AI Engine

Phase: *IDLE*

State: ---

Progress: ---

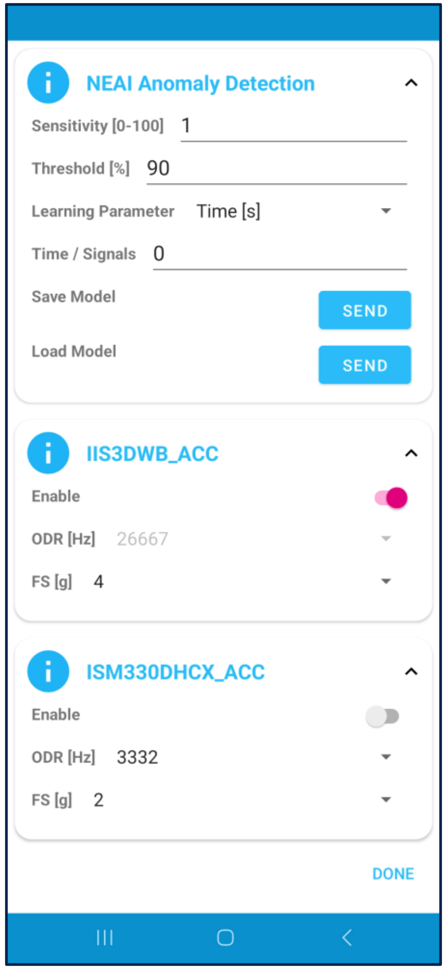
Results

Status: ---

Similarity: ---

First of all tap on gear icon to open the setting page

1



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

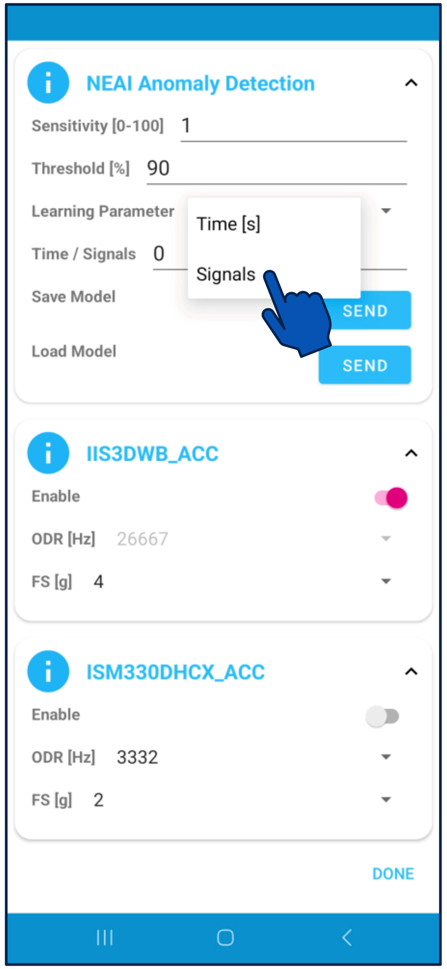
Enable

ODR [Hz] 3332

FS [g] 2

2

Tap on learning parameter and select *Signals* option



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

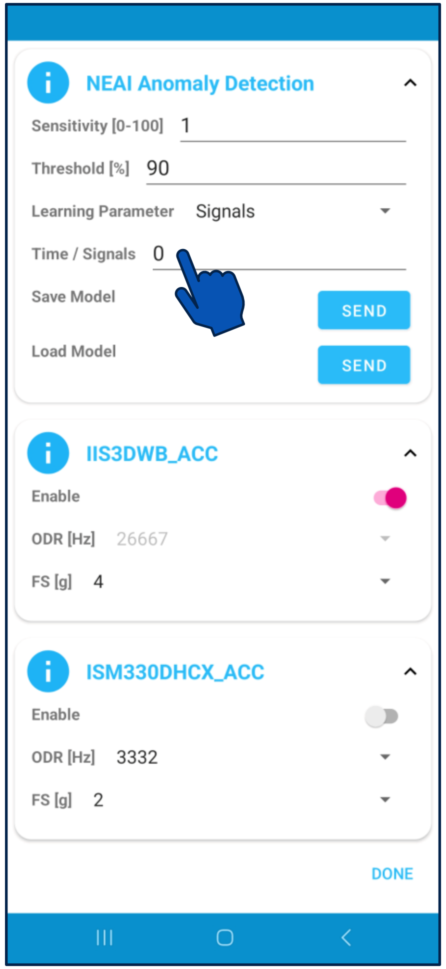
Enable

ODR [Hz] 3332

FS [g] 2

3

How to set signals to learn 2/2



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Signals

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

Enable

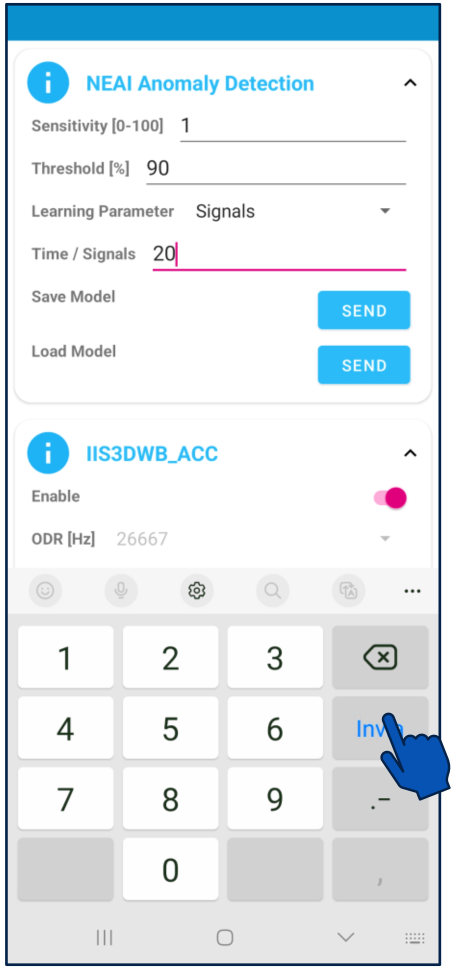
ODR [Hz] 3332

FS [g] 2

DONE

4

Tap on *Time/Signals* parameter



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Signals

Time / Signals 20

Save Model SEND

Load Model SEND

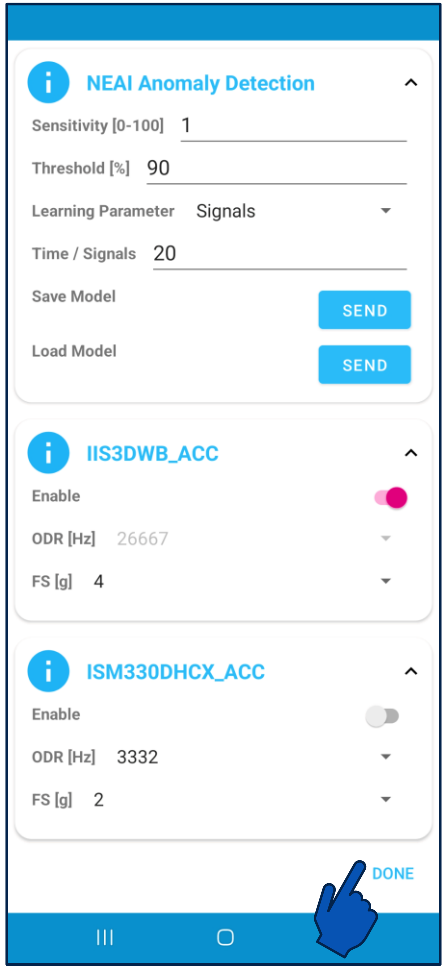
IIS3DWB_ACC

Enable

ODR [Hz] 26667

5

Enter the desired number of signals to learn



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter Signals

Time / Signals 20

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable

ODR [Hz] 26667

FS [g] 4

ISM330DHCX_ACC

Enable

ODR [Hz] 3332

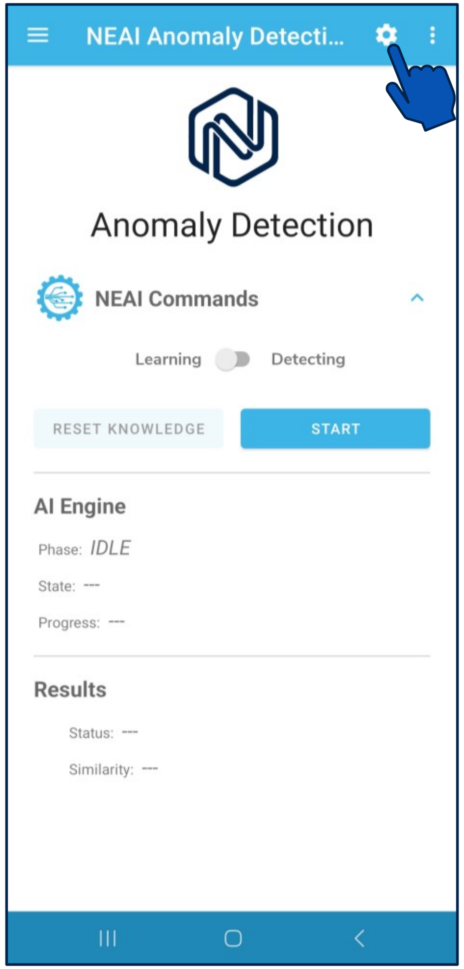
FS [g] 2

DONE

6

END

How to set AD library parameters



NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning ○ Detecting

AI Engine

Phase: *IDLE*

State: ---

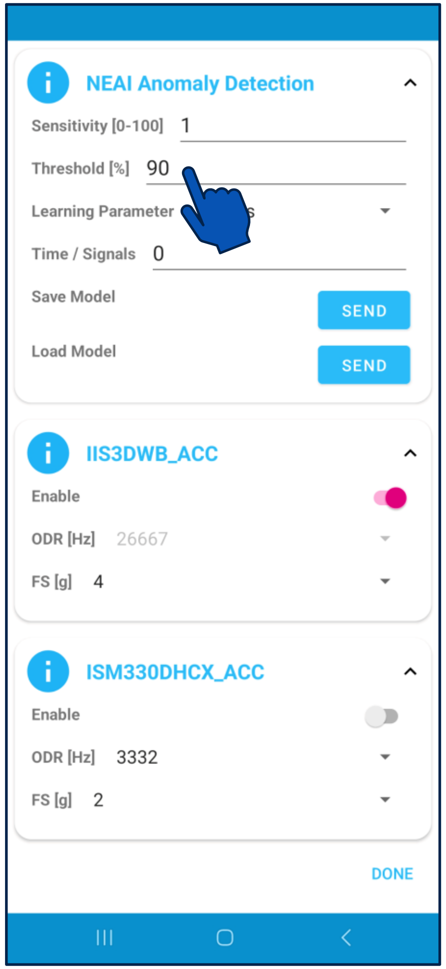
Progress: ---

Results

Status: ---

Similarity: ---

First of all tap on gear icon to open the setting page



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 90

Learning Parameter s

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable ●

ODR [Hz] 26667

FS [g] 4

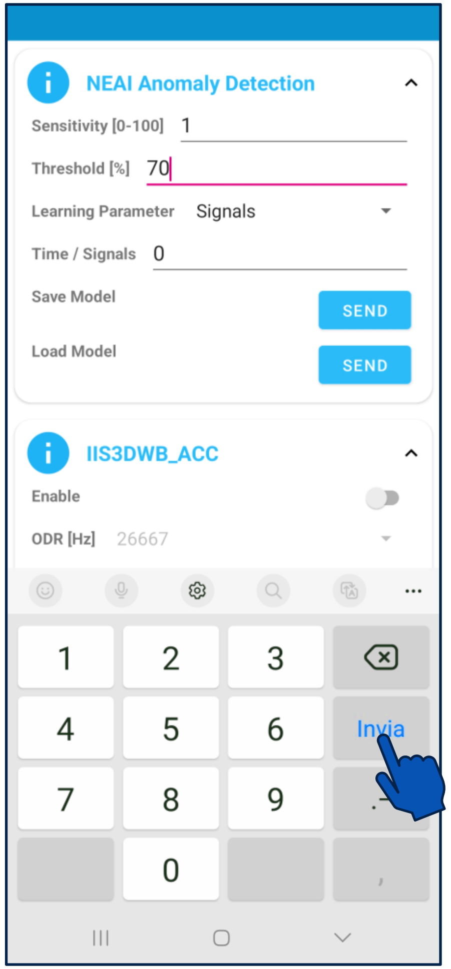
ISM330DHCX_ACC

Enable ○

ODR [Hz] 3332

FS [g] 2

Tap on *Sensitivity* or *Threshold* parameter to set its



NEAI Anomaly Detection

Sensitivity [0-100] 1

Threshold [%] 70

Learning Parameter Signals

Time / Signals 0

Save Model SEND

Load Model SEND

IIS3DWB_ACC

Enable ○

ODR [Hz] 26667

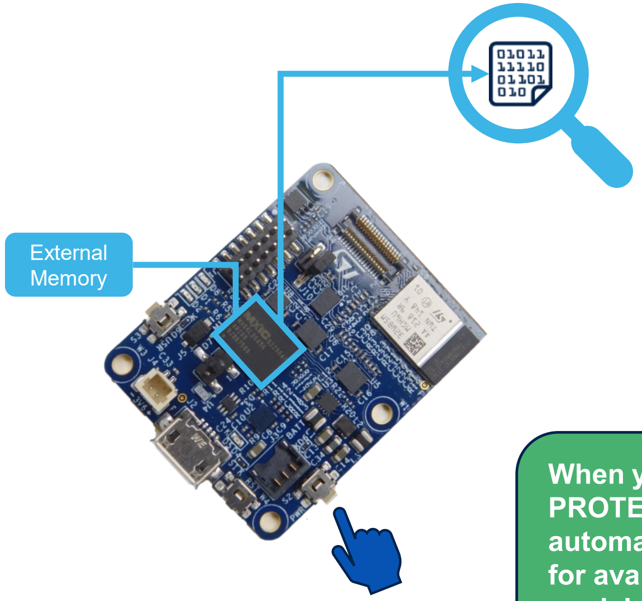
1

2

3

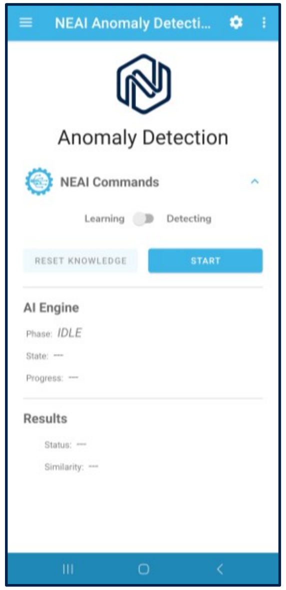
END

How to use save/load NEAI AD model 1/2



1

When you turn-on your PROTEUS, firmware automatically search for available NEAI AD model stored in the external flash memory



NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning Detecting

RESET KNOWLEDGE START

AI Engine

Phase: *IDLE*

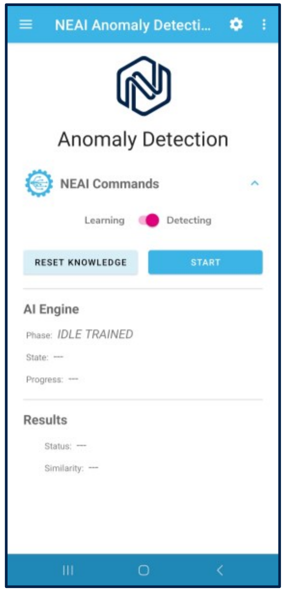
State: ---

Progress: ---

Results

Status: ---

Similarity: ---



NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning Detecting

RESET KNOWLEDGE START

AI Engine

Phase: *IDLE TRAINED*

State: ---

Progress: ---

Results

Status: ---

Similarity: ---

2

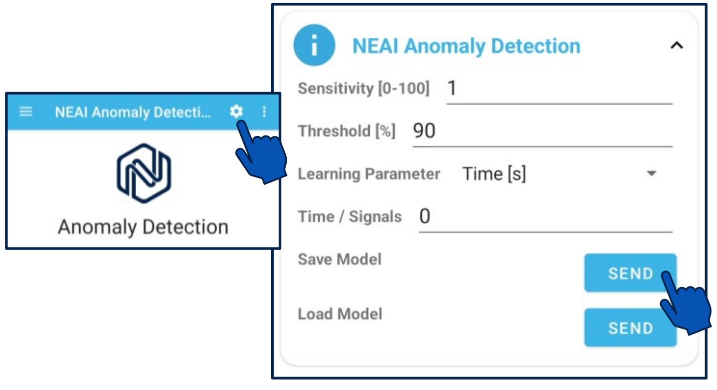
If no model was found, phase will be *IDLE*

Before detect anomalies, you need to start a learning phase

If a model was found, phase will be *IDLE TRAINED*

Now you are ready to start a detecting phase

How to use save/load NEAI AD model 2/2


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Anomaly Detection

Sensitivity [0-100] 1

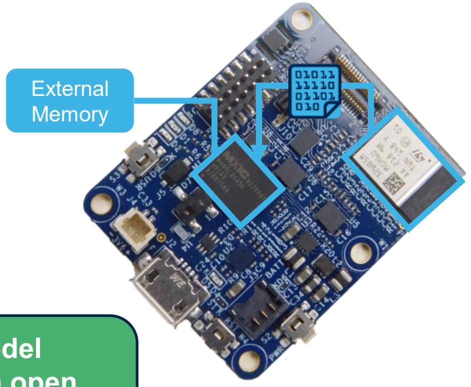
Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 0

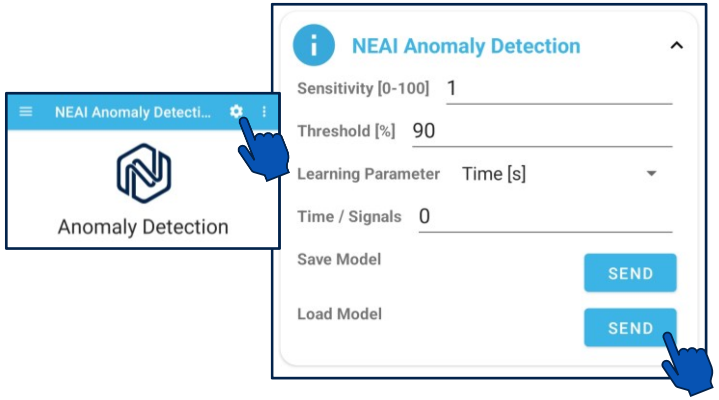
Save Model SEND

Load Model SEND



To avoid to lose your AD model after shutting down, you can open setting page and send ***Save Model*** command

3


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Anomaly Detection

Sensitivity [0-100] 1

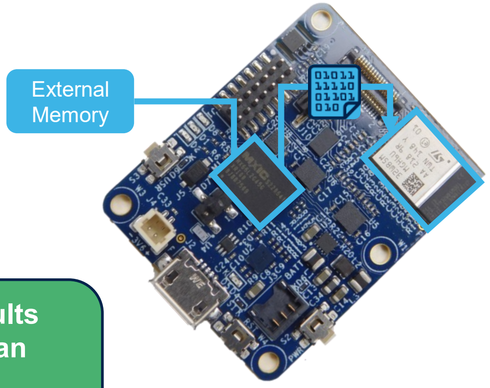
Threshold [%] 90

Learning Parameter Time [s]

Time / Signals 0

Save Model SEND

Load Model SEND



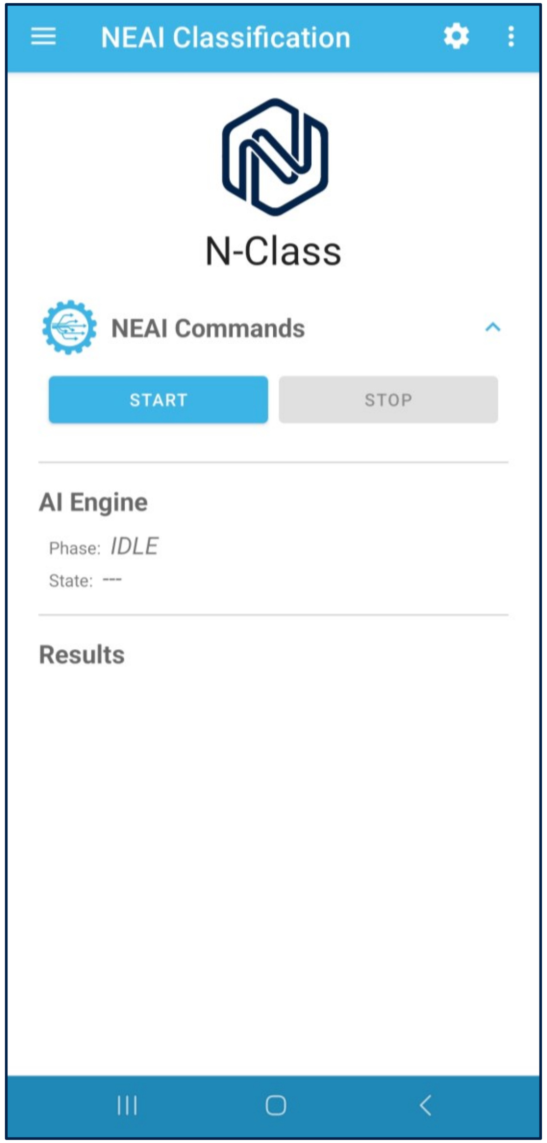
If you are dissatisfied with results come from last learning, you can restore the model saved in the external flash memory by ***Load Model*** command

4

END

Start Classifying phase

Two simple steps



NEAI Classification

N-Class

NEAI Commands

START STOP

AI Engine

Phase: *IDLE*

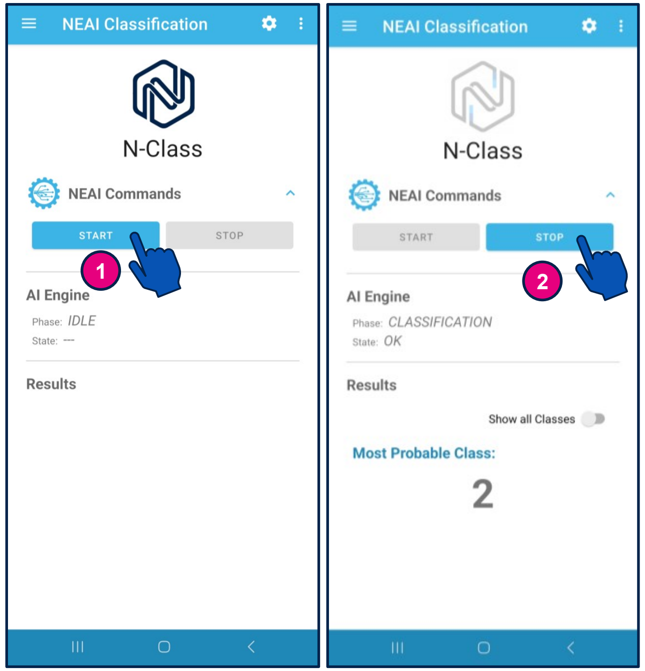
State: ---

Results

Through this demo you can monitor NanoEdgeAI NCC library status and also start/stop classifying phase. To start your first classification, follow the steps below:

1. Push start button START

2. Push stop button when you want STOP



NEAI Classification

N-Class

NEAI Commands

START STOP

1

AI Engine

Phase: *IDLE*

State: ---

Results

NEAI Classification

N-Class

NEAI Commands

START STOP

2

AI Engine

Phase: *CLASSIFICATION*

State: *OK*

Results

Show all Classes

Most Probable Class:

2

Remember that classifying phase will end only when you push stop button

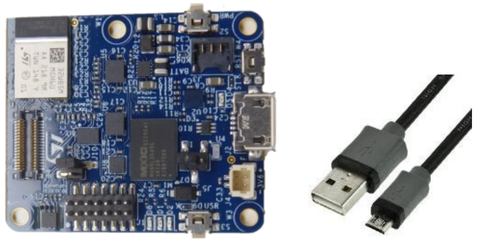
life.augmented

2.3- PDMWBSOC Application:

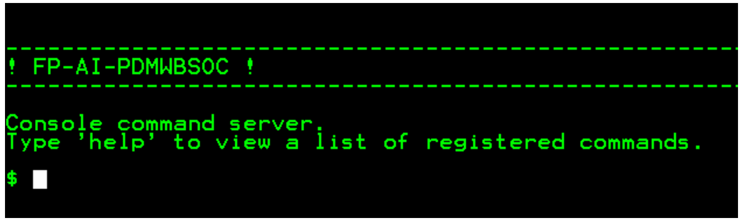
How to use CLI Terminal Console

life.augmented

The proper workflow



Plug USB connector and open Tera Term (*)



```
--------------------------------------------------------------
! FP-AI-PDMWBSOC !
--------------------------------------------------------------

Console command server.
Type 'help' to view a list of registered commands.
$ █
```

Enter *set_neai timer [t]* or *set_neai signals [n]* to customize your learning phase (**)

Start learning, detecting or classifying by specific commands

Stop learning, detectingor classifying automatically or pushing escape button

(**) This step is optional but remember that by default:

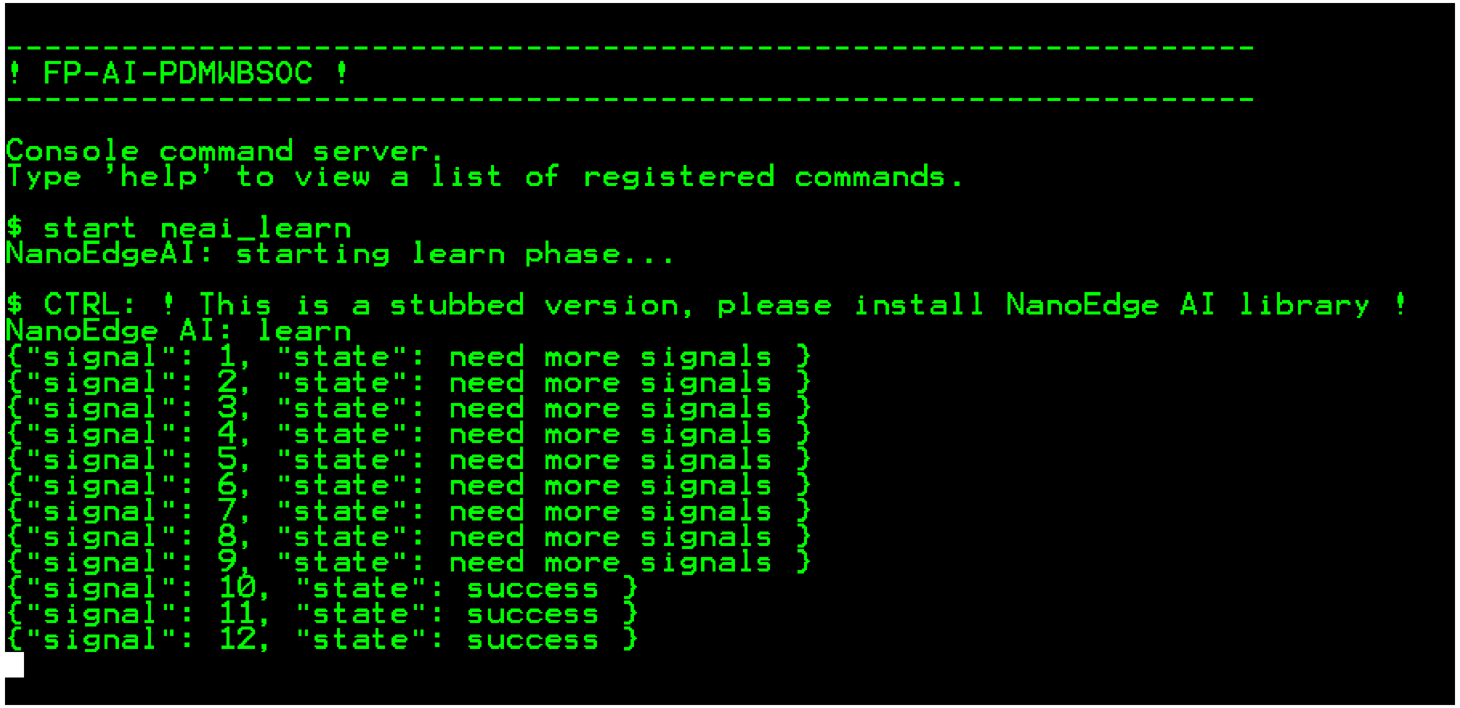
- ISM330DHCX is active with ODR = 3332 Hz and FS = 2 G
- Learning/Detecting phases will end when you'll push stop button (*Time/Signals* parameter is initialized to zero)

It's strongly recommended to setup your sensor according to dataset used to generate NEAI library

(*) It's strongly recommended to plug/unplug USB connector only when the library is not running: NOT plug/unplug USB during learning/detecting/classifying phase.

Anomaly Detection:

Start learning phase



```
-----------------------------------------------------------------
! FP-AI-PDMWBSOC !
-----------------------------------------------------------------

Console command server.
Type 'help' to view a list of registered commands.

$ start neai_learn
NanoEdgeAI: starting learn phase...

$ CTRL: ! This is a stubbed version, please install NanoEdge AI library !
NanoEdge AI: learn
{"signal": 1, "state": need more signals }
{"signal": 2, "state": need more signals }
{"signal": 3, "state": need more signals }
{"signal": 4, "state": need more signals }
{"signal": 5, "state": need more signals }
{"signal": 6, "state": need more signals }
{"signal": 7, "state": need more signals }
{"signal": 8, "state": need more signals }
{"signal": 9, "state": need more signals }
{"signal": 10, "state": success }
{"signal": 11, "state": success }
{"signal": 12, "state": success }
```

The user has to enter the proper command to start the learning phase

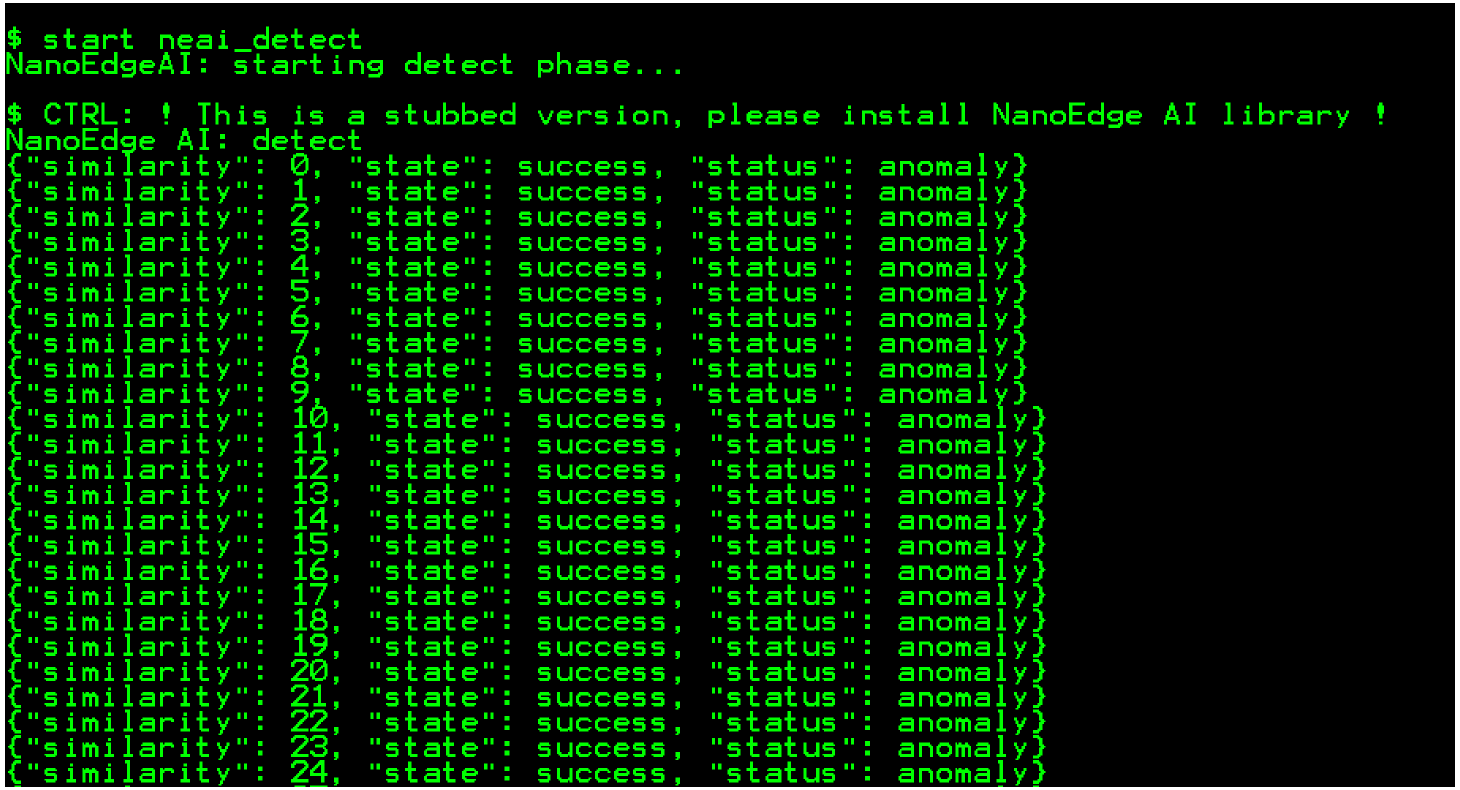
start neai_learn

Learning phase started, green LED blinks

life.augmented

Anomaly Detection:

Start detection phase



```
$ start neai_detect
NanoEdgeAI: starting detect phase...

$ CTRL: ! This is a stubbed version, please install NanoEdge AI library !
NanoEdge AI: detect
{"similarity": 0, "state": success, "status": anomaly}
{"similarity": 1, "state": success, "status": anomaly}
{"similarity": 2, "state": success, "status": anomaly}
{"similarity": 3, "state": success, "status": anomaly}
{"similarity": 4, "state": success, "status": anomaly}
{"similarity": 5, "state": success, "status": anomaly}
{"similarity": 6, "state": success, "status": anomaly}
{"similarity": 7, "state": success, "status": anomaly}
{"similarity": 8, "state": success, "status": anomaly}
{"similarity": 9, "state": success, "status": anomaly}
{"similarity": 10, "state": success, "status": anomaly}
{"similarity": 11, "state": success, "status": anomaly}
{"similarity": 12, "state": success, "status": anomaly}
{"similarity": 13, "state": success, "status": anomaly}
{"similarity": 14, "state": success, "status": anomaly}
{"similarity": 15, "state": success, "status": anomaly}
{"similarity": 16, "state": success, "status": anomaly}
{"similarity": 17, "state": success, "status": anomaly}
{"similarity": 18, "state": success, "status": anomaly}
{"similarity": 19, "state": success, "status": anomaly}
{"similarity": 20, "state": success, "status": anomaly}
{"similarity": 21, "state": success, "status": anomaly}
{"similarity": 22, "state": success, "status": anomaly}
{"similarity": 23, "state": success, "status": anomaly}
{"similarity": 24, "state": success, "status": anomaly}
```

The user has to enter the proper command to start detection phase

start neai_detect

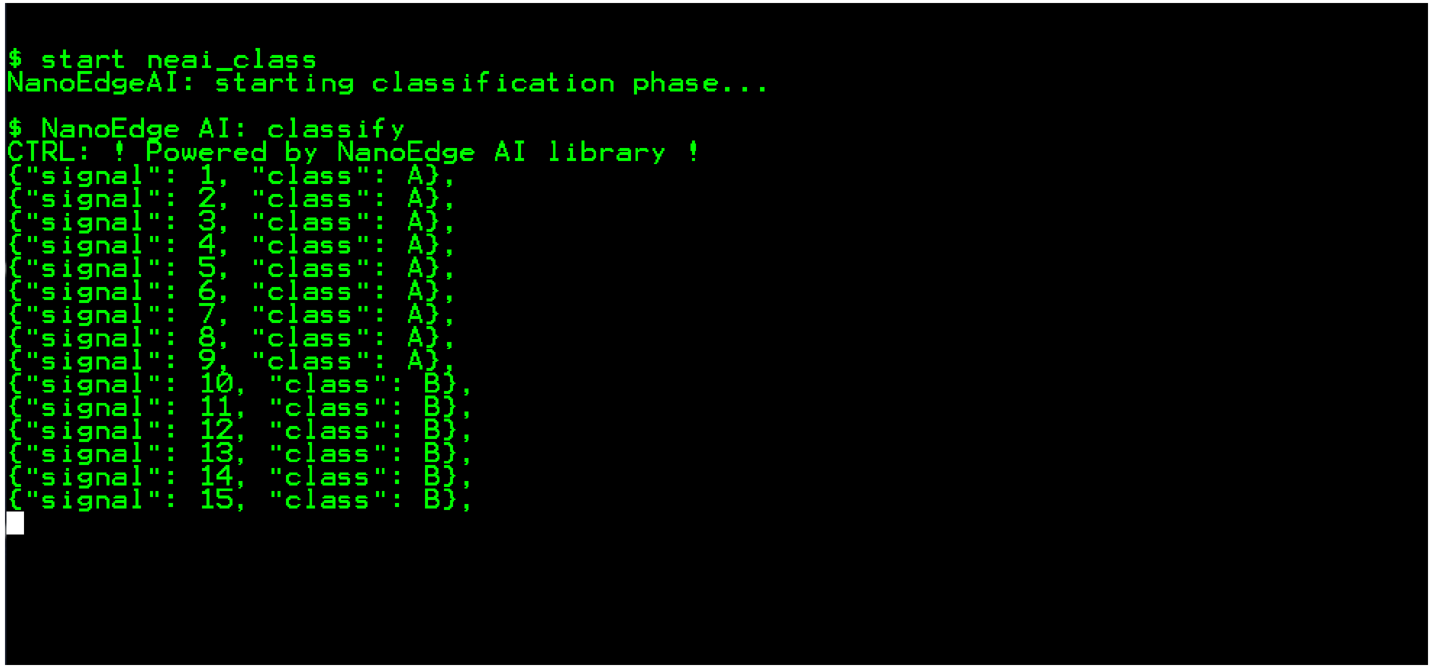
Detection started
Status Normal
green LED ON

Detection started
Status Abnormal
red LED ON

Remember that detecting phase will end only when you push escape button

N-Class Classification:

Start classification phase



```
$ start neai_class
NanoEdgeAI: starting classification phase...

$ NanoEdge AI: classify
CTRL: ! Powered by NanoEdge AI library !
{"signal": 1, "class": A},
{"signal": 2, "class": A},
{"signal": 3, "class": A},
{"signal": 4, "class": A},
{"signal": 5, "class": A},
{"signal": 6, "class": A},
{"signal": 7, "class": A},
{"signal": 8, "class": A},
{"signal": 9, "class": A},
{"signal": 10, "class": B},
{"signal": 11, "class": B},
{"signal": 12, "class": B},
{"signal": 13, "class": B},
{"signal": 14, "class": B},
{"signal": 15, "class": B},
```

The user has to enter the proper command to start classification phase

start neai_class

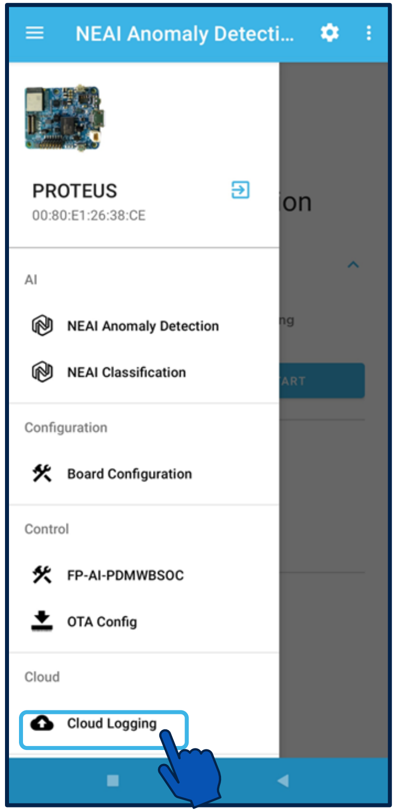
Remember that classifying phase will end only when you push escape button

2.4- PDMWBSOC Application:
Azure IoT Central Cloud Service

life.augmented

STBLESensor App

Azure IoT Central Cloud connection 1/2



NEAI Anomaly Detecti...

PROTEUS
00:80:E1:26:38:CE

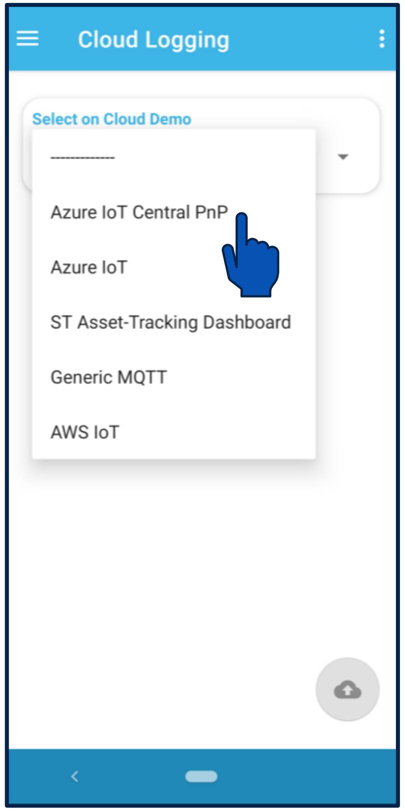
AI
- NEAI Anomaly Detection
- NEAI Classification

Configuration
- Board Configuration

Control
- FP-AI-PDMWBSOC
- OTA Config

Cloud
- Cloud Logging

Open **Cloud Logging**

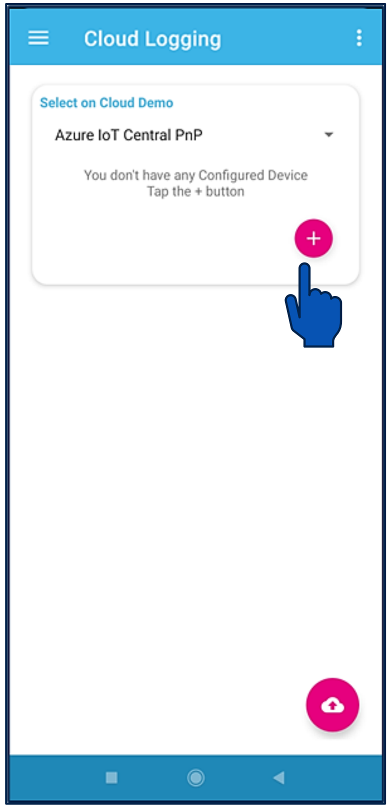


Cloud Logging

Select on Cloud Demo

- ------------
- Azure IoT Central PnP
- Azure IoT
- ST Asset-Tracking Dashboard
- Generic MQTT
- AWS IoT

Select **Azure IoT Central PnP**



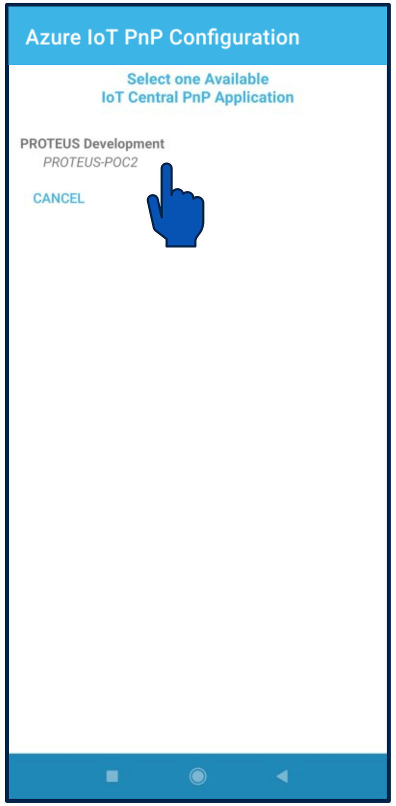
Cloud Logging

Select on Cloud Demo

Azure IoT Central PnP

You don't have any Configured Device
Tap the + button

Add the application **Proteus-poc2**



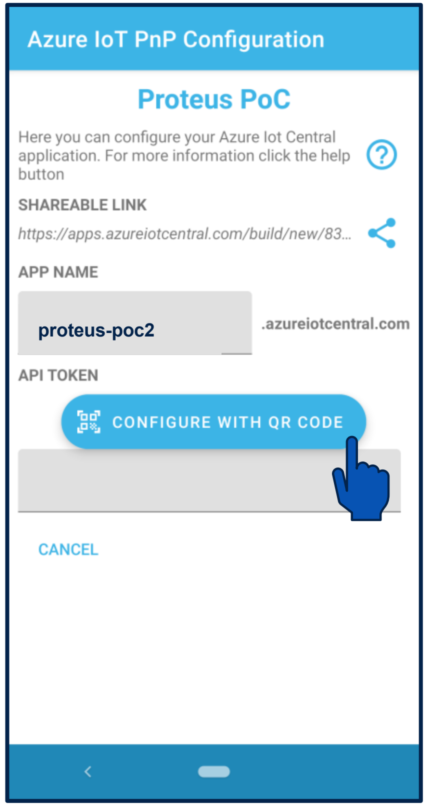
Azure IoT PnP Configuration

Select one Available
IoT Central PnP Application

PROTEUS Development
PROTEUS-POC2

CANCEL

Select the IoT Central App **proteus-poc2**



Azure IoT PnP Configuration

Proteus PoC

Here you can configure your Azure Iot Central application. For more information click the help button

SHAREABLE LINK
https://apps.azureiotcentral.com/build/new/83...

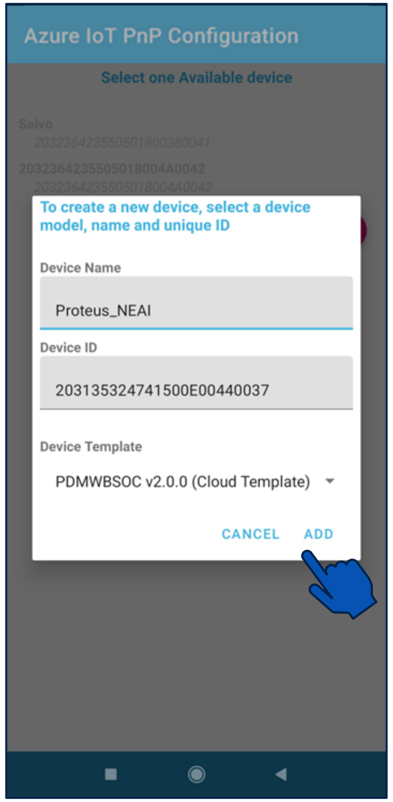
APP NAME
proteus-poc2 .azureiotcentral.com

API TOKEN

CONFIGURE WITH QR CODE

CANCEL

Insert the **QR-Code** retrieved during subscription *(See User Manual for more details)*



Azure IoT PnP Configuration

Select one Available device

Salvo
203236423550501800380041

2032364235505018004A0042
2032364235505018004A0042

To create a new device, select a device model, name and unique ID

Device Name
Proteus_NEAI

Device ID
203135324741500E00440037

Device Template
PDMWBSOC v2.0.0 (Cloud Template)

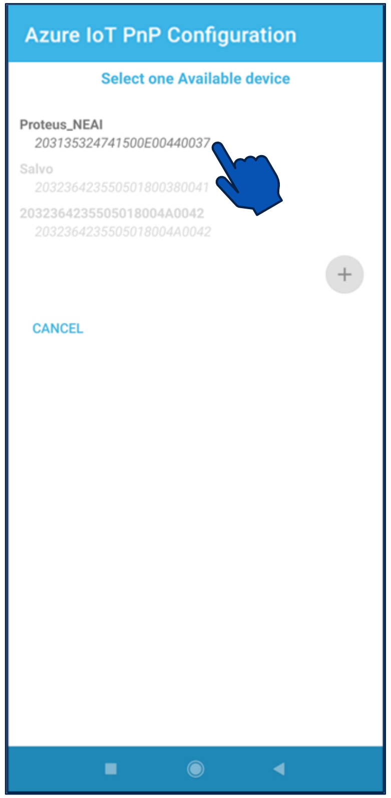
CANCEL ADD

Add your device using the template already published

life.augmented

STBLESensor App

Azure IoT Central Cloud connection 2/2



Azure IoT PnP Configuration

Select one Available device

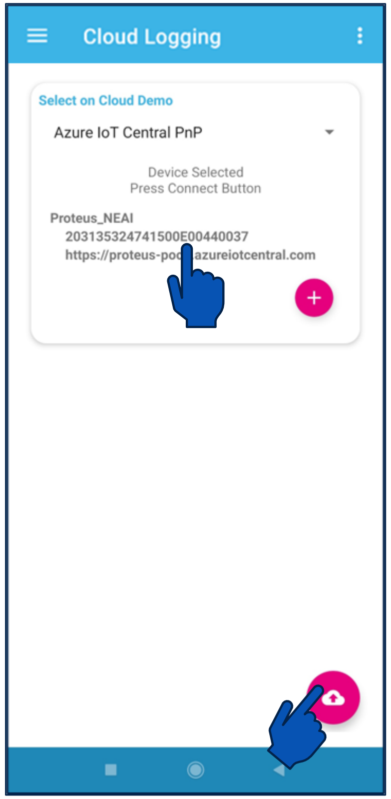
Proteus_NEAI
203135324741500E00440037

Salvo
2032364235505018003B0041

2032364235505018004A0042
2032364235505018004A0042

CANCEL

Device inserted in **proteus-poc2** app



Cloud Logging

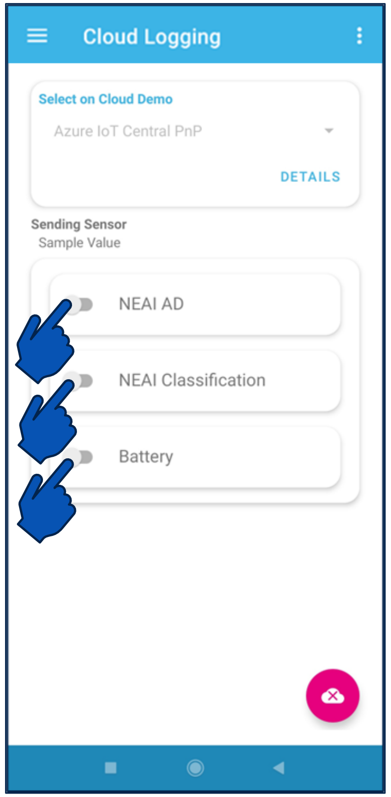
Select on Cloud Demo

Azure IoT Central PnP

Device Selected
Press Connect Button

Proteus_NEAI
203135324741500E00440037
https://proteus-poc2.azureiotcentral.com

Start the connectivity with Azure IoT Central



Cloud Logging

Select on Cloud Demo

Azure IoT Central PnP

DETAILS

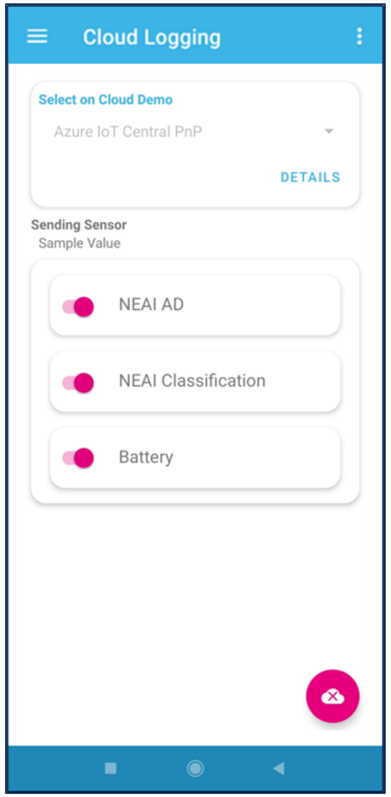
Sending Sensor
Sample Value

NEAI AD

NEAI Classification

Battery

List of all the measurement to send



Cloud Logging

Select on Cloud Demo

Azure IoT Central PnP

DETAILS

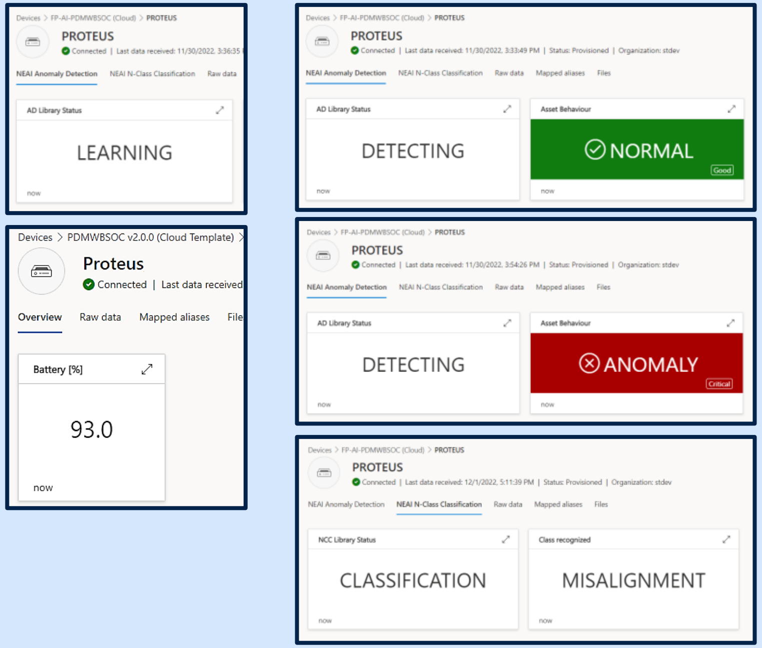
Sending Sensor
Sample Value

NEAI AD

NEAI Classification

Battery

NEAI AD + NEAI Classification + Battery



Devices > FP-AI-PDMWBSOC (Cloud) > PROTEUS
PROTEUS
Connected | Last data received: 11/30/2022, 3:36:35
NEAI Anomaly Detection | NEAI N-Class Classification | Raw data
AD Library Status: LEARNING (now)

Devices > FP-AI-PDMWBSOC (Cloud) > PROTEUS
PROTEUS
Connected | Last data received: 11/30/2022, 3:33:49 PM | Status: Provisioned | Organization: stdev
NEAI Anomaly Detection | NEAI N-Class Classification | Raw data | Mapped aliases | Files
AD Library Status: DETECTING (now)
Asset Behaviour: NORMAL, Good (now)

Devices > PDMWBSOC v2.0.0 (Cloud Template) >
Proteus
Connected | Last data received
Overview | Raw data | Mapped aliases | File
Battery [%]: 93.0 (now)

Devices > FP-AI-PDMWBSOC (Cloud) > PROTEUS
PROTEUS
Connected | Last data received: 11/30/2022, 3:54:26 PM | Status: Provisioned | Organization: stdev
NEAI Anomaly Detection | NEAI N-Class Classification | Raw data | Mapped aliases | Files
AD Library Status: DETECTING (now)
Asset Behaviour: ANOMALY, Critical (now)

Devices > FP-AI-PDMWBSOC (Cloud) > PROTEUS
PROTEUS
Connected | Last data received: 12/1/2022, 5:11:39 PM | Status: Provisioned | Organization: stdev
NEAI Anomaly Detection | NEAI N-Class Classification | Raw data | Mapped aliases | Files
NCC Library Status: CLASSIFICATION (now)
Class recognized: MISALIGNMENT (now)

https://proteus-poc2.azureiotcentral.com/

Measurements provisioned in **Azure IoT Central Dashboard**

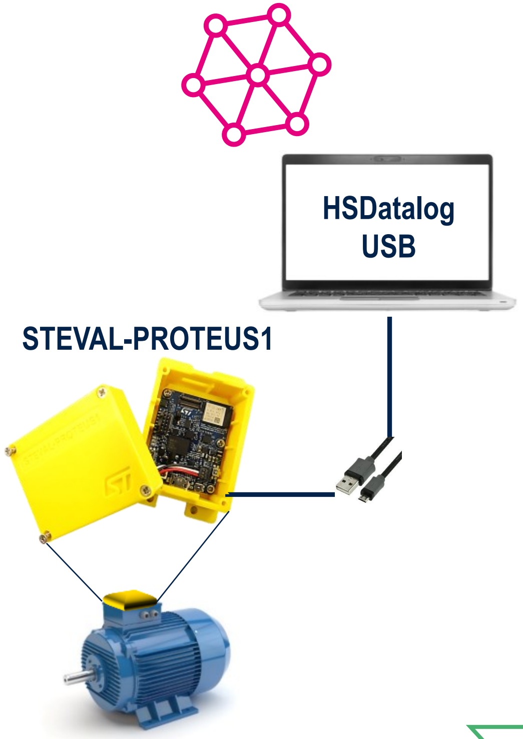
life.augmented

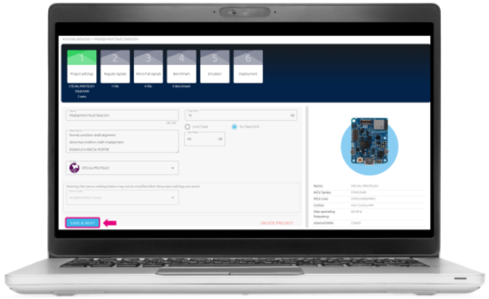
2.5- Setup NanoEdge™ AI library

life.augmented

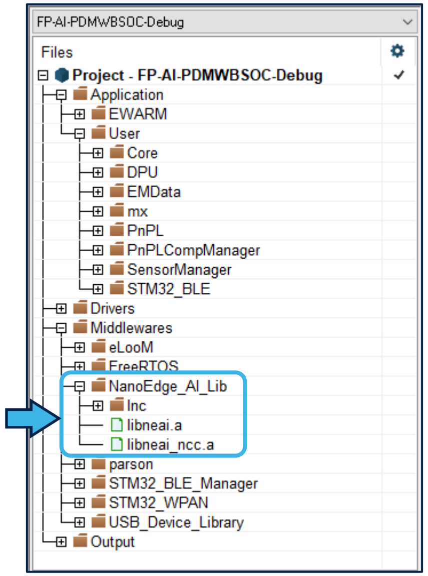
FP-AI-PDMWBSOC
NEAI Machine Learning in STEVAL-PROTEUS1

Capture data → Label Dataset and Classifying → Benchmark to Build ML Libs → ML Libs integration in FW package → Run applications with NEAI AD or NEAI- NClass





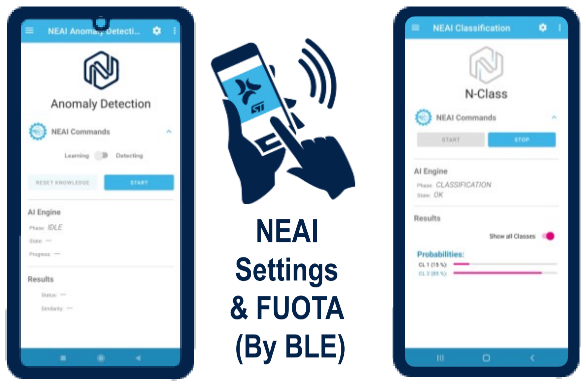
NANOEDGE AI STUDIO



```
FP-AI-PDMWBSOC-Debug
Files
⊟ Project - FP-AI-PDMWBSOC-Debug ✓
  ⊟ Application
    ⊞ EWARM
    ⊟ User
      ⊞ Core
      ⊞ DPU
      ⊞ EMData
      ⊞ mx
      ⊞ PnPL
      ⊞ PnPLCompManager
      ⊞ SensorManager
      ⊞ STM32_BLE
  ⊞ Drivers
  ⊟ Middlewares
    ⊞ eLooM
    ⊞ FreeRTOS
    ⊟ NanoEdge_AI_Lib
      ⊞ Inc
        libneai.a
        libneai_ncc.a
    ⊞ parson
    ⊞ STM32_BLE_Manager
    ⊞ STM32_WPAN
    ⊞ USB_Device_Library
  ⊞ Output
```

NEAI AD

NEAI CLASSIFICATION



NEAI Settings & FUOTA (By BLE)

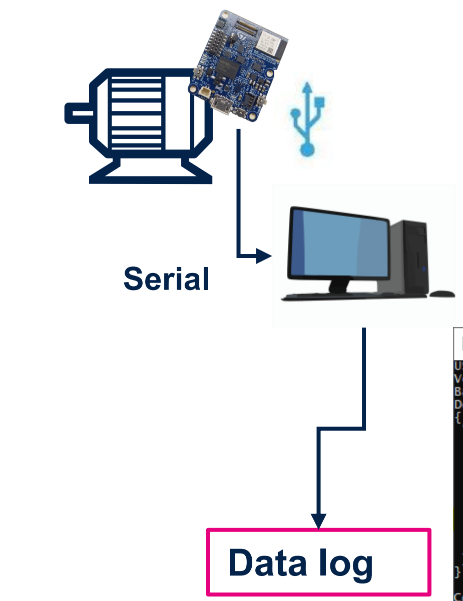
Sensor Datalogging → Normal & Anomaly Dataset / Every Classes Dataset → Modeling / Results → NanoEdgeAI integration → Control by BLE App or CLI Terminal

life.augmented

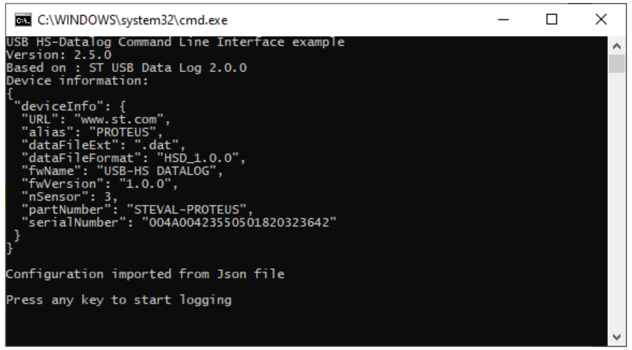
Capture data and create specific datasets

Generating contextual data using HSDATALOG in Utilities folder

1 Generate data set



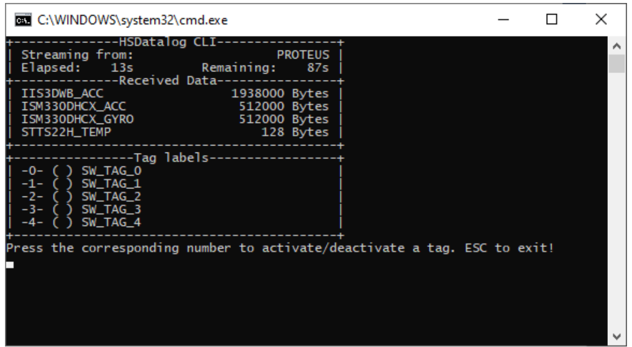
- ❑ The equipment behaviors can be analyzed, creating a lot of different dataset based on vibrometer or accelerometer data, focused on “Normal” & “Anomaly” working condition to monitor and detect. The same approach can be used to classify different “Working Conditions” creating specific dataset to built a N-Class Classification machine learning model.
- ❑ To prepare these datasets, you can use HSDatalog firmware, that can be loaded directly via BLE Sensor Classic App, by FUOTA, available in ***Utilities/HS_Datalog/Proteus_HS_Datalog_FUOTA_reference.bin***
- ❑ Before start take care that on STEVAL-PROTEUS the blue LED is blinking. If it doesn’t, please, press the “RESET” button.
- ❑ **On PC, launch “*Utilities/HS_Datalog/cli_example/USB_DataLog_Run.bat*”** or any other batch file inside “.\PROTEUS_batch_file_examples”
- ❑ An automatic folder will be created, collecting all the ***.DAT** files configured to use for the library generation.



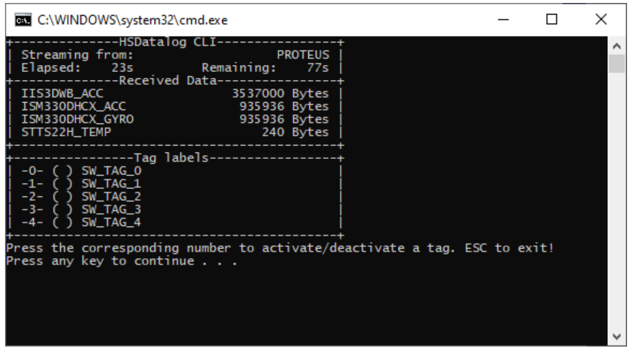
```
USB HS-Datalog Command Line Interface example
Version: 2.5.0
Based on : ST USB Data Log 2.0.0
Device information:
{
 "deviceInfo": {
  "URL": "www.st.com",
  "alias": "PROTEUS",
  "dataFileExt": ".dat",
  "dataFileFormat": "HSD_1.0.0",
  "fwName": "USB-HS DATALOG",
  "fwVersion": "1.0.0",
  "nSensor": 3,
  "partNumber": "STEVAL-PROTEUS",
  "serialNumber": "004A00423550501820323642"
 }
}

Configuration imported from Json file

Press any key to start logging
```



```
+--------------HSDatalog CLI----------------+
| Streaming from:                   PROTEUS |
| Elapsed:    13s         Remaining:    87s |
+-------------Received Data-----------------+
| IIS3DWB_ACC                 1938000 Bytes |
| ISM330DHCX_ACC               512000 Bytes |
| ISM330DHCX_GYRO              512000 Bytes |
| STTS22H_TEMP                    128 Bytes |
+-------------------------------------------+
+----------------Tag labels-----------------+
| -0- ( ) SW_TAG_0                          |
| -1- ( ) SW_TAG_1                          |
| -2- ( ) SW_TAG_2                          |
| -3- ( ) SW_TAG_3                          |
| -4- ( ) SW_TAG_4                          |
+-------------------------------------------+
Press the corresponding number to activate/deactivate a tag. ESC to exit!
```



```
+--------------HSDatalog CLI----------------+
| Streaming from:                   PROTEUS |
| Elapsed:    23s         Remaining:    77s |
+-------------Received Data-----------------+
| IIS3DWB_ACC                 3537000 Bytes |
| ISM330DHCX_ACC               935936 Bytes |
| ISM330DHCX_GYRO              935936 Bytes |
| STTS22H_TEMP                    240 Bytes |
+-------------------------------------------+
+----------------Tag labels-----------------+
| -0- ( ) SW_TAG_0                          |
| -1- ( ) SW_TAG_1                          |
| -2- ( ) SW_TAG_2                          |
| -3- ( ) SW_TAG_3                          |
| -4- ( ) SW_TAG_4                          |
+-------------------------------------------+
Press the corresponding number to activate/deactivate a tag. ESC to exit!
Press any key to continue . . .
```

Press «RETURN» button
The blue LED turns off and the green LED starts blinking

The acquisition will continue for all the configured time or up to «ESC» button pressed

Use NanoEdge AI Studio tool

Label Dataset and Classifying to use with NanoEdgeAI Studio

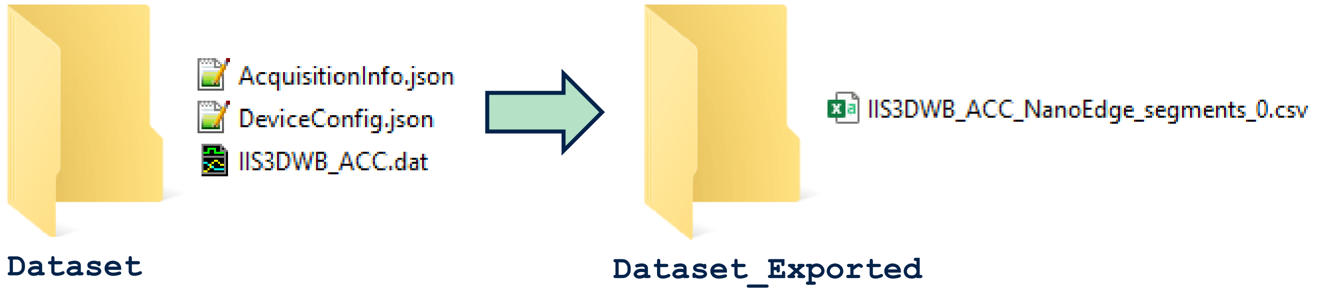
2.1 Convert each datasets in the right format (*.csv) accepted by NanoEdgeAI Studio tool (v3.2.1)

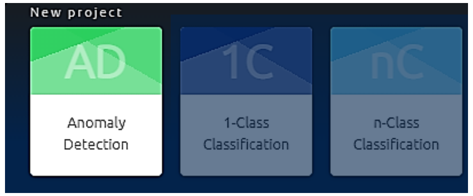
❑ The dataset previously generated using the datalogging firmware, classified as “Normal”, “Anomaly”, or as “Specific Cases to classify”, must be used as input dataset to create the new libraries for Anomaly Detection or N-Class Classification, to include inside the application FW.

❑ Before to use these datasets, use the batch file → ***“Utilities/HS_Datalog/ PROTEUS_batch_file_examples/HS-DL_NanoEdge_Conversion.bat*** starting from the related folder containing at least one ***.dat** file plus two json configuration files.

❑ The python script will convert automatically the ***.dat** datalog stored in a *Dataset* folder, into a ***.csv** format in a folder named as *Dataset_Exported,* requiring just the following instruction:

```
1.  Input Acquisition
    folder name
2.  Data signal length
3.  Output Acquisition
    folder name
```

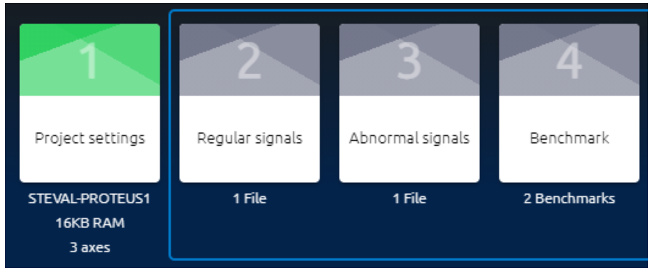




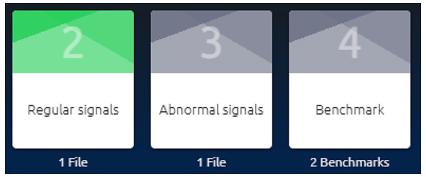
Use NanoEdge AI Studio tool

Label Dataset and Classifying to use with NanoEdgeAI Studio

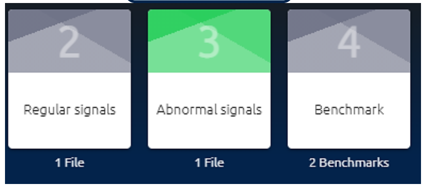
2.2 Create a new project for **Anomaly Detection** library starting from the datasets and build the library

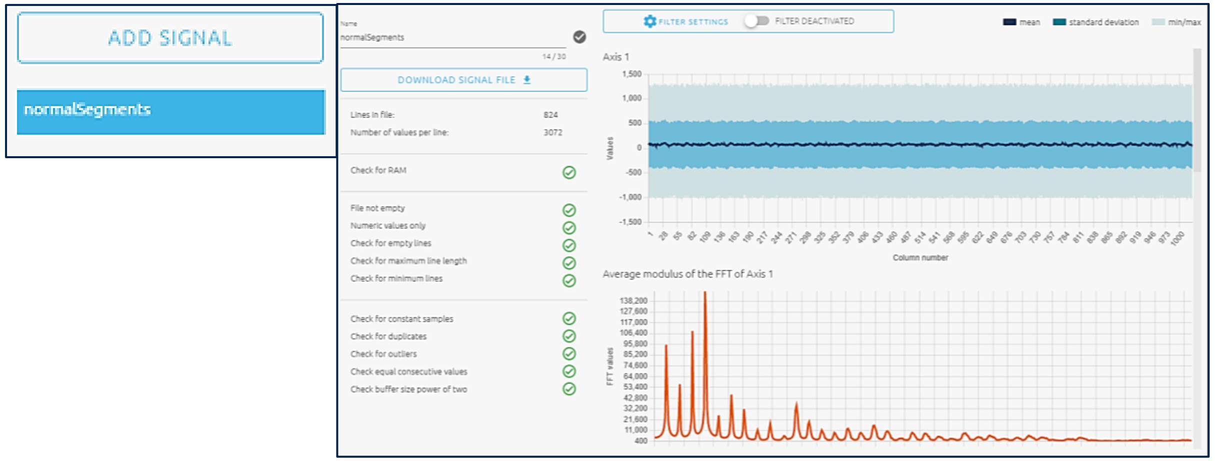


Normal



Anomaly





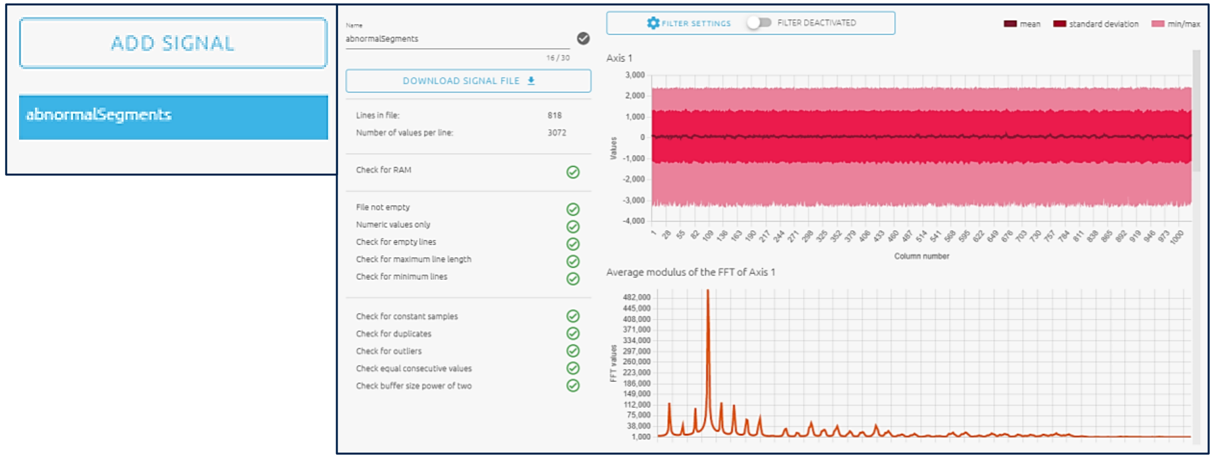
Name
normalSegments
14 / 30

Lines in file: 824
Number of values per line: 3072

Check for RAM ✓

File not empty ✓
Numeric values only ✓
Check for empty lines ✓
Check for maximum line length ✓
Check for minimum lines ✓

Check for constant samples ✓
Check for duplicates ✓
Check for outliers ✓
Check equal consecutive values ✓
Check buffer size power of two ✓



Name
abnormalSegments
16 / 30

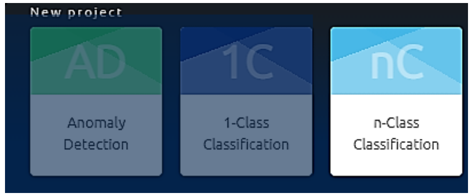
Lines in file: 818
Number of values per line: 3072

Check for RAM ✓

File not empty ✓
Numeric values only ✓
Check for empty lines ✓
Check for maximum line length ✓
Check for minimum lines ✓

Check for constant samples ✓
Check for duplicates ✓
Check for outliers ✓
Check equal consecutive values ✓
Check buffer size power of two ✓

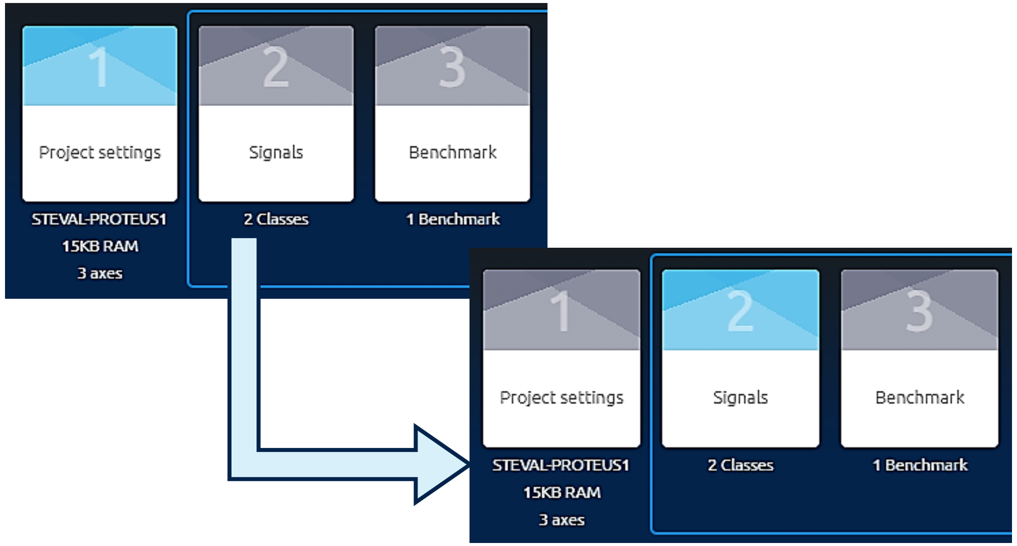
life.augmented

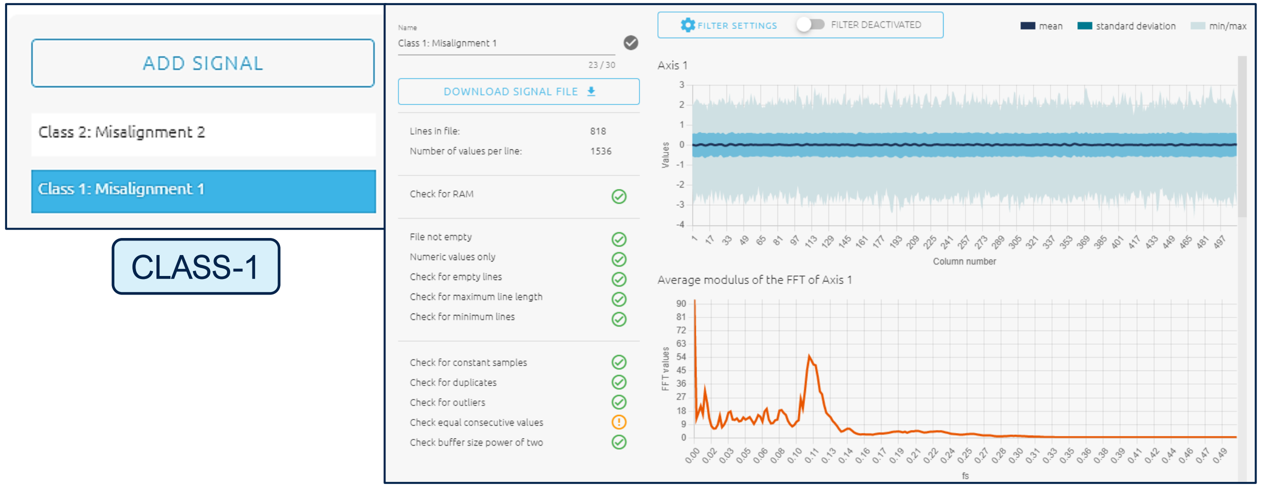


Use NanoEdge AI Studio tool

Label Dataset and Classifying to use with NanoEdgeAI Studio

2.3 Create a new project for **N-Class Classification** library starting from the datasets and build the library





Name
Class 1: Misalignment 1
23 / 30

DOWNLOAD SIGNAL FILE

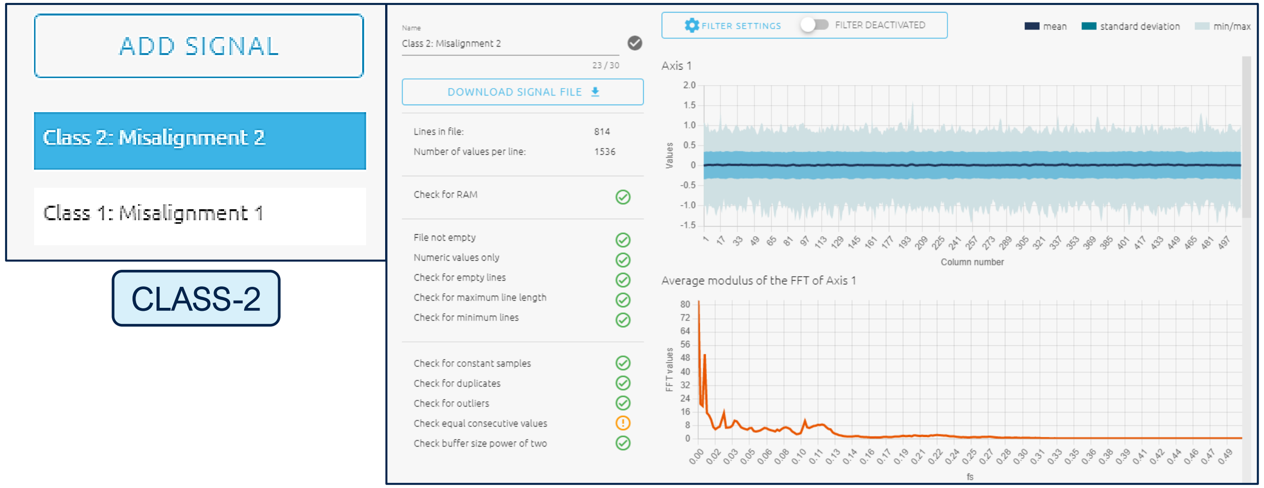
Lines in file: 818
Number of values per line: 1536

Check for RAM ✓

File not empty ✓
Numeric values only ✓
Check for empty lines ✓
Check for maximum line length ✓
Check for minimum lines ✓

Check for constant samples ✓
Check for duplicates ✓
Check for outliers ✓
Check equal consecutive values !
Check buffer size power of two ✓

CLASS-1



Name
Class 2: Misalignment 2
23 / 30

DOWNLOAD SIGNAL FILE

Lines in file: 814
Number of values per line: 1536

Check for RAM ✓

File not empty ✓
Numeric values only ✓
Check for empty lines ✓
Check for maximum line length ✓
Check for minimum lines ✓

Check for constant samples ✓
Check for duplicates ✓
Check for outliers ✓
Check equal consecutive values !
Check buffer size power of two ✓

CLASS-2

life.augmented

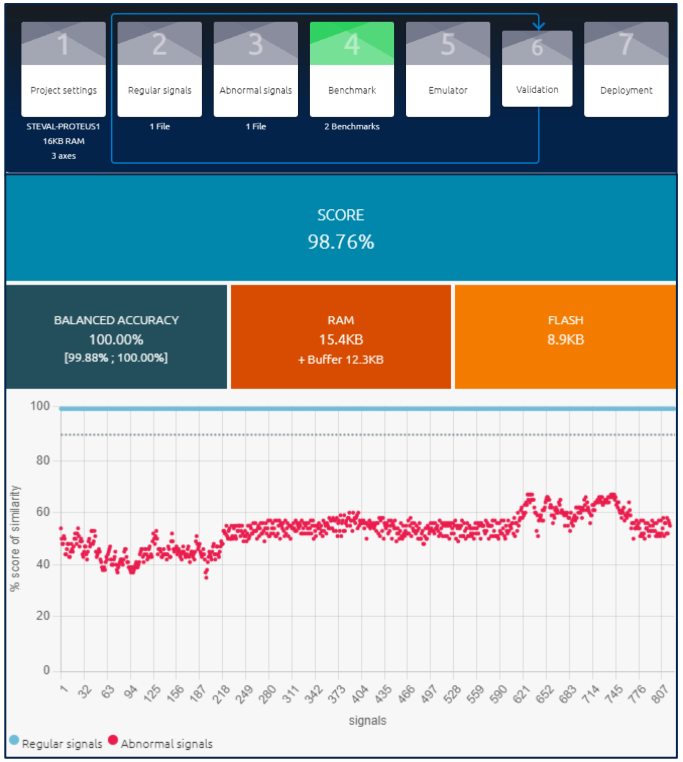
Use NanoEdge AI Studio tool

NANOEDGE AI STUDIO

Run the Benchmark to create the different ML Library

3.1 Machine Library building running a benchmark to find the best library in terms of Accuracy, Performance (Score), RAM and FLASH Size

Anomaly Detection



1 Project settings — STEVAL-PROTEUS1, 16KB RAM, 3 axes

2 Regular signals — 1 File

3 Abnormal signals — 1 File

4 Benchmark — 2 Benchmarks

5 Emulator

6 Validation

7 Deployment

SCORE 98.76%

BALANCED ACCURACY 100.00% [99.88% ; 100.00%]

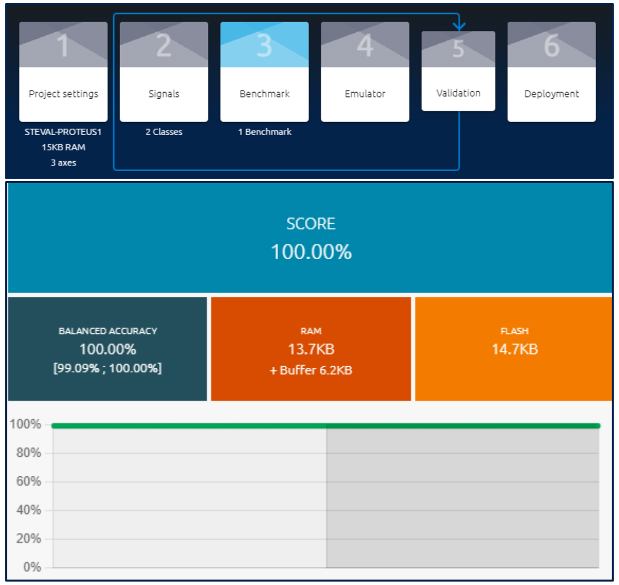
RAM 15.4KB + Buffer 12.3KB

FLASH 8.9KB

Normal

Anomaly

N-CLASS Classification



1 Project settings — STEVAL-PROTEUS1, 15KB RAM, 3 axes

2 Signals — 2 Classes

3 Benchmark — 1 Benchmark

4 Emulator

5 Validation

6 Deployment

SCORE 100.00%

BALANCED ACCURACY 100.00% [99.09% ; 100.00%]

RAM 13.7KB + Buffer 6.2KB

FLASH 14.7KB

CLASS-1 CLASS-2

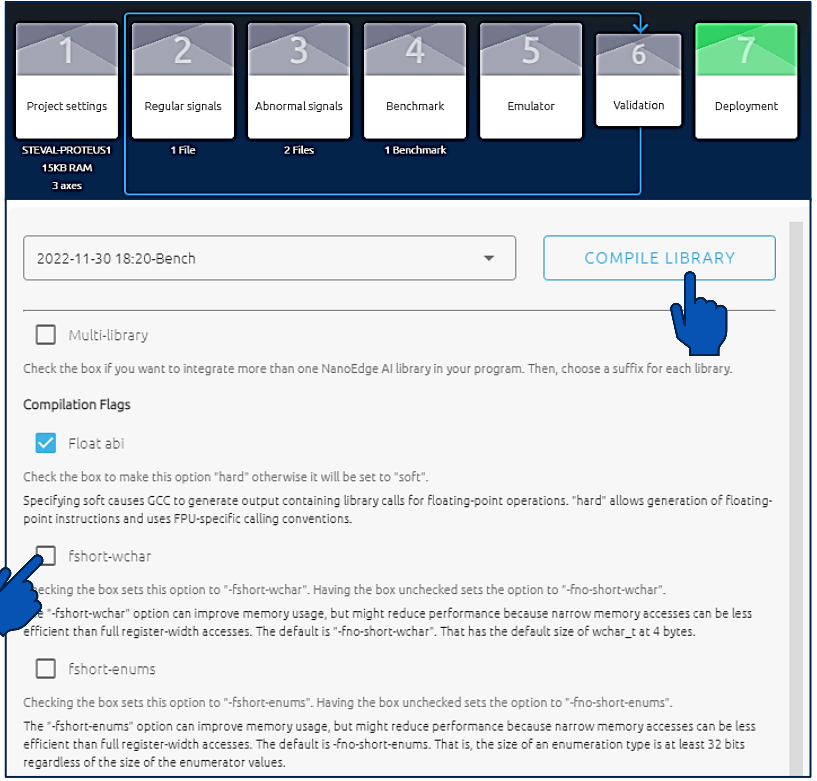
life.augmented

Use NanoEdge AI Studio tool

Choose the best libraries and Deploy in binary format

3.2 Press the “Compile Library” button to deploy a ZIP file including all the library files to include in your application

Anomaly Detection



STEVAL-PROTEUS1
15KB RAM
3 axes

1 File

2 Files

1 Benchmark

2022-11-30 18:20-Bench ▾

COMPILE LIBRARY

☐ Multi-library

Check the box if you want to integrate more than one NanoEdge AI library in your program. Then, choose a suffix for each library.

Compilation Flags

☑ Float abi

Check the box to make this option "hard" otherwise it will be set to "soft".

Specifying soft causes GCC to generate output containing library calls for floating-point operations. "hard" allows generation of floating-point instructions and uses FPU-specific calling conventions.

☐ fshort-wchar

ecking the box sets this option to "-fshort-wchar". Having the box unchecked sets the option to "-fno-short-wchar".

e "-fshort-wchar" option can improve memory usage, but might reduce performance because narrow memory accesses can be less efficient than full register-width accesses. The default is "-fno-short-wchar". That has the default size of wchar_t at 4 bytes.

☐ fshort-enums

Checking the box sets this option to "-fshort-enums". Having the box unchecked sets the option to "-fno-short-enums".

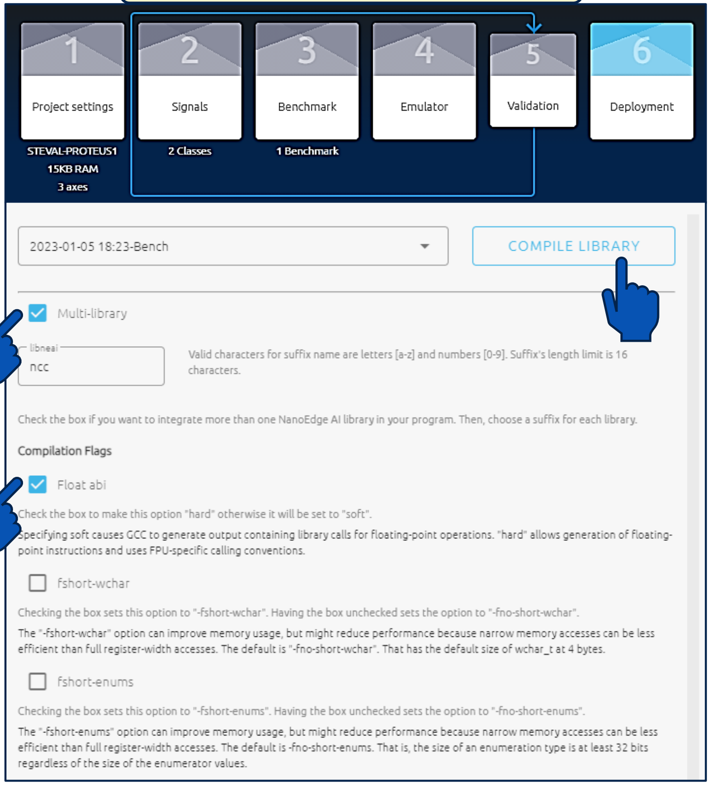
The "-fshort-enums" option can improve memory usage, but might reduce performance because narrow memory accesses can be less efficient than full register-width accesses. The default is -fno-short-enums. That is, the size of an enumeration type is at least 32 bits regardless of the size of the enumerator values.

Set "**fshort-wchar**" flag just to deploy the library for the **Keil**® toolchain

N-CLASS Classification

Set “**Multi-library**“ flag and insert “**ncc**” just to generate specific classification files to insert in the application

Set "**fshort-wchar**" flag just to deploy the library for the **Keil**® toolchain



STEVAL-PROTEUS1
15KB RAM
3 axes

2 Classes

1 Benchmark

2023-01-05 18:23-Bench ▾

COMPILE LIBRARY

☑ Multi-library

libneai: ncc

Valid characters for suffix name are letters [a-z] and numbers [0-9]. Suffix's length limit is 16 characters.

Check the box if you want to integrate more than one NanoEdge AI library in your program. Then, choose a suffix for each library.

Compilation Flags

☑ Float abi

Check the box to make this option "hard" otherwise it will be set to "soft".

Specifying soft causes GCC to generate output containing library calls for floating-point operations. "hard" allows generation of floating-point instructions and uses FPU-specific calling conventions.

☐ fshort-wchar

Checking the box sets this option to "-fshort-wchar". Having the box unchecked sets the option to "-fno-short-wchar".

The "-fshort-wchar" option can improve memory usage, but might reduce performance because narrow memory accesses can be less efficient than full register-width accesses. The default is "-fno-short-wchar". That has the default size of wchar_t at 4 bytes.

☐ fshort-enums

Checking the box sets this option to "-fshort-enums". Having the box unchecked sets the option to "-fno-short-enums".

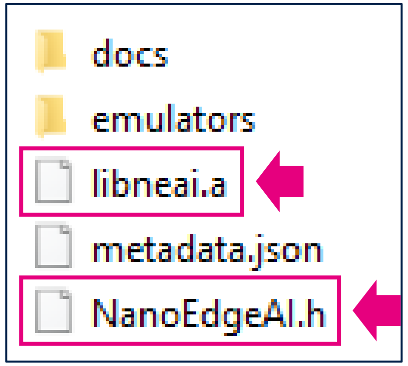
The "-fshort-enums" option can improve memory usage, but might reduce performance because narrow memory accesses can be less efficient than full register-width accesses. The default is -fno-short-enums. That is, the size of an enumeration type is at least 32 bits regardless of the size of the enumerator values.

NEAI Library Integration in the FP

Embed the Machine Learning Library into FP-AI-PDMWBSOC

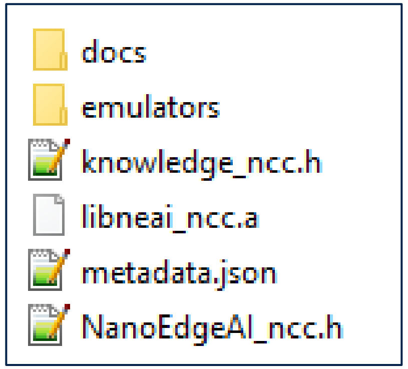
4.1 Replace just the following NEAI files deployed into the project folder

Anomaly Detection


docs
emulators
libneai.a
metadata.json
NanoEdgeAI.h

1. **libneai.a**
   (for IAR&STM32CubeIDE projects)
2. **libneai.lib**
   (generated as explained before and renamed from libneai.a to libneai.lib, for KEIL projects)
3. **NanoEdgeAI.h**

N-CLASS Classification


docs
emulators
knowledge_ncc.h
libneai_ncc.a
metadata.json
NanoEdgeAI_ncc.h

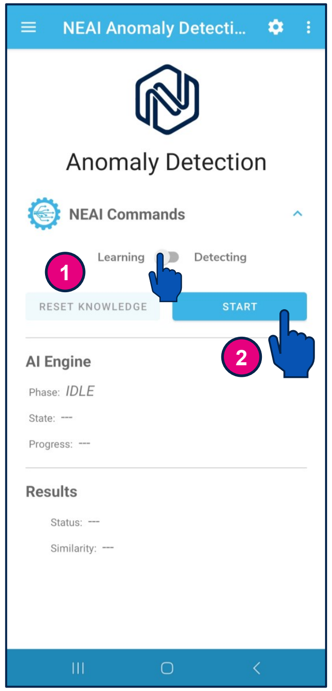
1. **Libneai_ncc.a**
   (for IAR&STM32CubeIDE projects)
2. **Libneai_ncc.lib**
   (generated as explained before and renamed from libneai.a to libneai.lib, for KEIL projects)
3. **NanoEdgeAI_ncc.h**
4. **knowledge_ncc.h**

4.2 Compile again the FW application in order to update the NEAI libraries.
Once updated the application binaries, the BLE FUOTA functionalities allow to update directly the STEVAL-PROTEUS without using programmer tools.

Run Learning and Detection by BLE or CLI

Run applications with NEAI Anomaly Detection

Run NEAI-AD Learning


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

1

RESET KNOWLEDGE | START

2

AI Engine

Phase: IDLE

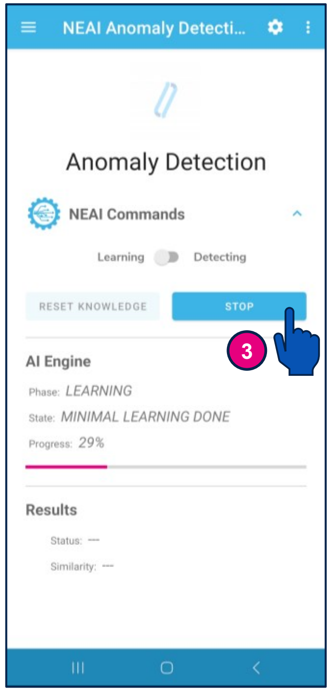
State: ---

Progress: ---

Results

Status: ---

Similarity: ---


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

RESET KNOWLEDGE | STOP

3

AI Engine

Phase: LEARNING

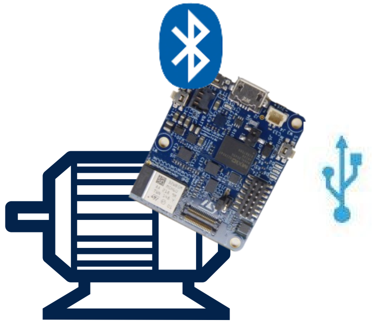
State: MINIMAL LEARNING DONE

Progress: 29%

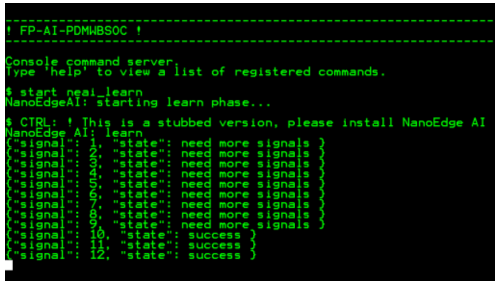
Results

Status: ---

Similarity: ---



start neai_learn

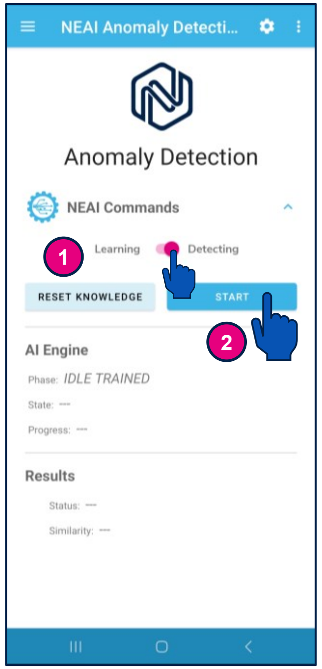

```
--------------------------------------------------------------------------------
! FP-AI-PDMWBSOC !
--------------------------------------------------------------------------------

Console command server.
Type 'help' to view a list of registered commands.

$ start neai_learn
NanoEdgeAI: starting learn phase...

$ CTRL: ! This is a stubbed version, please install NanoEdge AI
NanoEdge AI: learn
{"signal": 1, "state": need more signals }
{"signal": 2, "state": need more signals }
{"signal": 3, "state": need more signals }
{"signal": 4, "state": need more signals }
{"signal": 5, "state": need more signals }
{"signal": 6, "state": need more signals }
{"signal": 7, "state": need more signals }
{"signal": 8, "state": need more signals }
{"signal": 9, "state": need more signals }
{"signal": 10, "state": success }
{"signal": 11, "state": success }
{"signal": 12, "state": success }
```

Run NEAI-AD Detection


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

1

RESET KNOWLEDGE | START

2

AI Engine

Phase: IDLE TRAINED

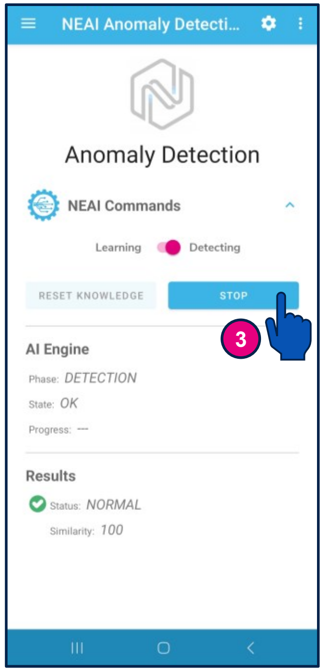
State: ---

Progress: ---

Results

Status: ---

Similarity: ---


NEAI Anomaly Detecti...

Anomaly Detection

NEAI Commands

Learning / Detecting

RESET KNOWLEDGE | STOP

3

AI Engine

Phase: DETECTION

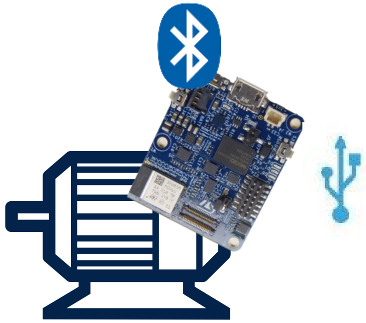
State: OK

Progress: ---

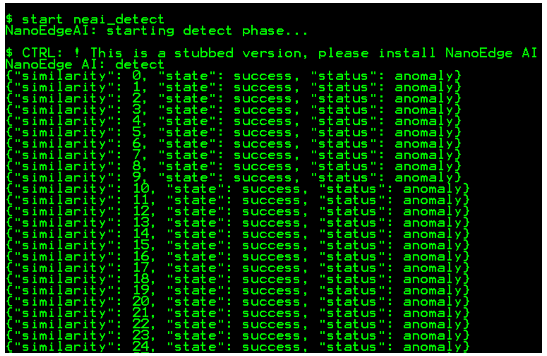
Results

✓ Status: NORMAL

Similarity: 100



start neai_detect


```
$ start neai_detect
NanoEdgeAI: starting detect phase...

$ CTRL: ! This is a stubbed version, please install NanoEdge AI
NanoEdge AI: detect
{"similarity": 0, "state": success, "status": anomaly}
{"similarity": 1, "state": success, "status": anomaly}
{"similarity": 2, "state": success, "status": anomaly}
{"similarity": 3, "state": success, "status": anomaly}
{"similarity": 4, "state": success, "status": anomaly}
{"similarity": 5, "state": success, "status": anomaly}
{"similarity": 6, "state": success, "status": anomaly}
{"similarity": 7, "state": success, "status": anomaly}
{"similarity": 8, "state": success, "status": anomaly}
{"similarity": 9, "state": success, "status": anomaly}
{"similarity": 10, "state": success, "status": anomaly}
{"similarity": 11, "state": success, "status": anomaly}
{"similarity": 12, "state": success, "status": anomaly}
{"similarity": 13, "state": success, "status": anomaly}
{"similarity": 14, "state": success, "status": anomaly}
{"similarity": 15, "state": success, "status": anomaly}
{"similarity": 16, "state": success, "status": anomaly}
{"similarity": 17, "state": success, "status": anomaly}
{"similarity": 18, "state": success, "status": anomaly}
{"similarity": 19, "state": success, "status": anomaly}
{"similarity": 20, "state": success, "status": anomaly}
{"similarity": 21, "state": success, "status": anomaly}
{"similarity": 22, "state": success, "status": anomaly}
{"similarity": 23, "state": success, "status": anomaly}
{"similarity": 24, "state": success, "status": anomaly}
```

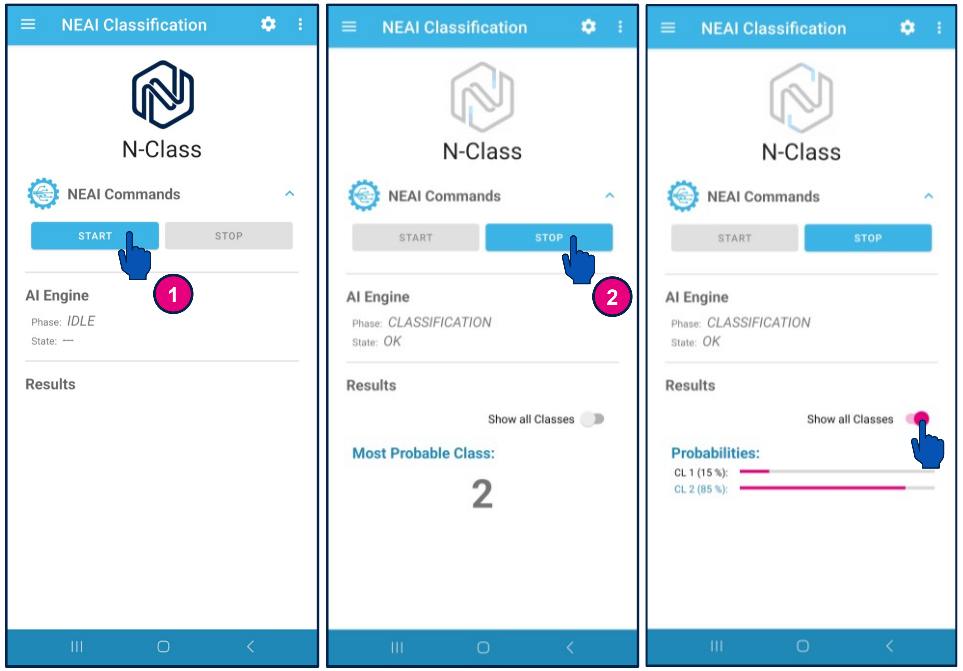
When the phase is IDLE, is mandatory to perform a new **LEARNING** phase before **DETECTING**, using the CLI or directly the BLE Sensor Classic App.

- ❑ **Learn** the normal modes on the edge
- ❑ **Detect** anomaly on your asset

Run **N-Class** classification by BLE or CLI

Run applications with NEAI- Nclass classification

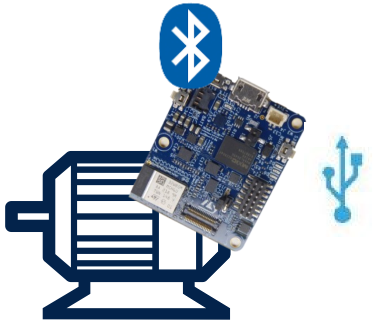
Run NanoEdgeAI N-Class classification



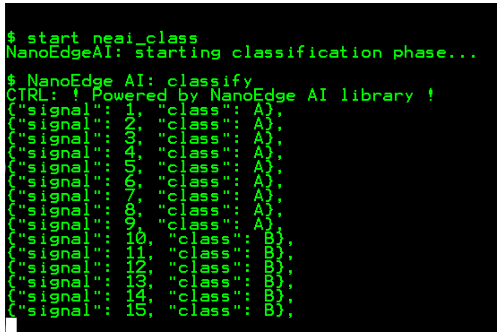
AI Engine
Phase: IDLE
State: ---
Results

AI Engine
Phase: CLASSIFICATION
State: OK
Results
Most Probable Class:
2

AI Engine
Phase: CLASSIFICATION
State: OK
Results
Probabilities:
CL 1 (15 %):
CL 2 (85 %):



start neai_learn



```
$ start neai_class
NanoEdgeAI: starting classification phase...

$ NanoEdge AI: classify
CTRL: ! Powered by NanoEdge AI library !
{"signal": 1, "class": A},
{"signal": 2, "class": A},
{"signal": 3, "class": A},
{"signal": 4, "class": A},
{"signal": 5, "class": A},
{"signal": 6, "class": A},
{"signal": 7, "class": A},
{"signal": 8, "class": A},
{"signal": 9, "class": A},
{"signal": 10, "class": B},
{"signal": 11, "class": B},
{"signal": 12, "class": B},
{"signal": 13, "class": B},
{"signal": 14, "class": B},
{"signal": 15, "class": B},
```

life.augmented

3- Documents & Related Resources

life.augmented

Documents & Related Resources

All documents are available in the DESIGN tab of the related products webpage

❑ STEVAL-PROTEUS1

- ❖ **DB4641**: Industrial sensor evaluation kit for condition monitoring based on the 2.4 GHz STM32WB5MMG module **–** HW Data brief Hardware
- ❖ **UM3000**: Getting started with the STEVAL-PROTEUS1 evaluation kit for condition monitoring based on the 2.4 GHz STM32WB5MMG module **–** HW User Manual
- ❖ **Schematics**, **BOM**, **Gerber files**, Certifications

❑ FP-AI-PDMWBSOC

- ❖ **DB4776:** STM32Cube function pack for STEVAL-PROTEUS1 evaluation kit for anomaly detection based on artificial intelligence (AI) – SW Data brief
- ❖ **UM3069**: Getting started with the STM32Cube function pack for STEVAL-PROTEUS1 evaluation kit for predictive maintenance application based on artificial intelligence (AI) – SW User Manual
- ❖ **QUICK START GUIDE**: STM32Cube function pack for STEVAL-PROTEUS1 evaluation kit for predictive maintenance application based on artificial intelligence (AI)

❑ SW TOOLS

- ❖ **STBLESensClassic :** ST BLE sensor Classic application for Android and iOS
- ❖ **NanoEdgeAIStudio**: Automated Machine Learning (ML) tool for STM32 developers

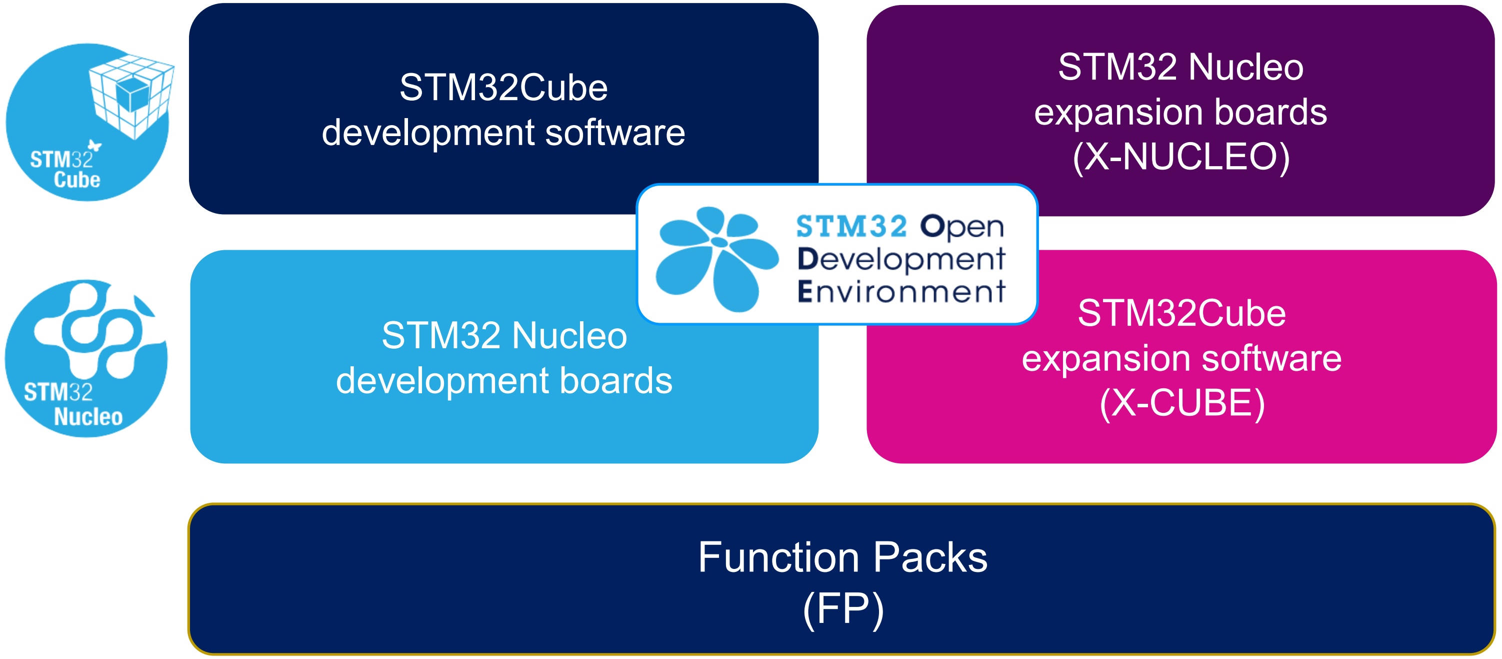
4- STM32 Open Development Environment: Overview

life.augmented

STM32 Open Development Environment

Fast, affordable Prototyping and Development

- The STM32 Open Development Environment (STM32 ODE) is an open, flexible, easy, and affordable way to develop innovative devices and applications based on the STM32 32-bit microcontroller family combined with other state-of-the-art ST components connected via expansion boards. It enables fast prototyping with leading-edge components that can quickly be transformed into final designs



STM32Cube development software

STM32 Nucleo expansion boards (X-NUCLEO)

STM32 Open Development Environment

STM32 Nucleo development boards

STM32Cube expansion software (X-CUBE)

Function Packs (FP)

For further information, please visit www.st.com/stm32ode

Our technology starts with You

life.augmented