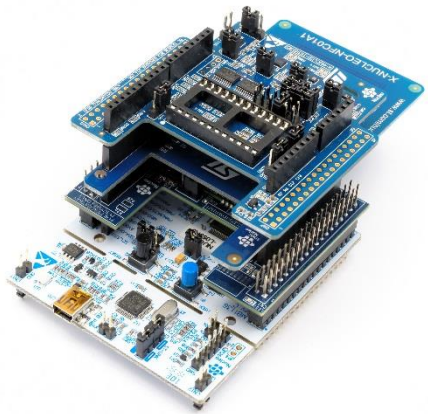
Quick Start Guide

STM32 ODE function pack for IoT node with Wi-Fi or Ethernet, NFC, sensors and motor control, connected to Microsoft Azure cloud

(FP-CLD-AZURE1)



life.augmented

Version 3.3.0 (May 14, 2018)

Quick Start Guide Contents

Motion MEMS and Environmental Sensors Expansion Board

Hardware Overview (2/6)

X-NUCLEO-IKS01A2 Hardware Description

The X-NUCLEO-IKS01A2 is a motion MEMS and environmental sensor expansion board for the STM32 Nucleo. It is compatible with the Arduino UNO R3 connector layout, and is designed around the LSM6DSL 3D accelerometer and 3D gyroscope, the LSM303AGR 3D accelerometer and 3D magnetometer, the HTS221 humidity and temperature sensor and the LPS22HB pressure sensor. The X-NUCLEO-IKS01A2 interfaces with the STM32 microcontroller via the I²C pin, and it is possible to change the default I²C port.

Key Product on board

LSM6DSL

MEMS 3D accelerometer (±2/±4/±8 /±16 g) + 3D gyroscope (±125/±245/±500/±1000 /±2000 dps)

LSM303AGR

MEMS 3D magnetometer (±50 gauss) + MEMS 3D accelerometer (±2/±4/±8/±16 g)

LPS22HB

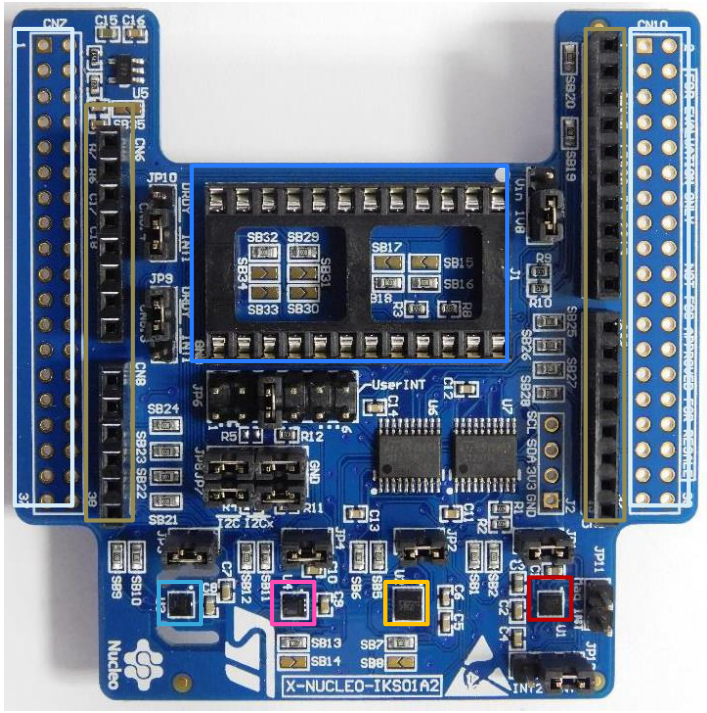
MEMS pressure sensor, 260-1260 hPa absolute digital output barometer

HTS221

Capacitive digital relative humidity and temperature

DIL 24-pin

Socket available for additional MEMS adapters and other sensors



HTS221 LSM6DSL ST morpho connector**

LPS25HB LIS303AGR Arduino UNO R3 connector

DIL 24-pin

** Connector for the STM32 Nucleo Board

Latest info available at www.st.com
X-NUCLEO-IKS01A2

life.augmented

Dynamic NFC tag expansion board
Hardware Overview (3/6)

X-NUCLEO-NFC04A1 Hardware Description

- The X-NUCLEO-NFC04A1 dynamic NFC/RFID tag IC expansion board is based on the ST25DV04K NFC Type V/RFID tag IC with a dual interface 4 Kbits EEPROM that also features an I²C interface. It can be powered by the pin of Arduino connector or directly by the received carrier electromagnetic field.
- The X-NUCLEO-NFC04A1 expansion board is compatible with the Arduino™ UNO R3 connector pin assignment and can easily be plugged onto any STM32 Nucleo board. Various expansion boards can also be stacked to evaluate different devices operating together with the dynamic NFC tag. The board also features an antenna with a 54 mm ISO 24.2 diameter, single layer, copper etched on PCB.

Key products on board

ST25DV04KV
Dynamic NFC/RFID tag IC with 4-Kbit, 16-Kbit or 64-Kbit EEPROM, and Fast Transfer Mode capability

Latest info available at www.st.com
X-NUCLEO-NFC04A1

Two axis stepper motor driver expansion board
Hardware Overview (4/6)

X-NUCLEO-IHM02A1 Hardware description

The X-NUCLEO-IHM02A1 is a two axis stepper motor driver expansion board based on L6470. It provides an affordable and easy-to-use solution for driving low voltage motor control for Stepper Motor in your STM32 Nucleo project. The expansion board includes two L6470s, a fully-integrated micro stepping motor driver used to control stepper motors by means of high-end motion control commands received through SPI. It is capable of driving one or two stepper motors when plugged into an STM32 Nucleo board.

Main features:

- Nominal operating voltage range: 8 V - 45 V DC
- Maximum output peak current: 7.0 A (3.0 A rms) for each motor driver
- Digital voltage supply is selectable (3.3 V or 5.0 V)
- USART communication
- SPI interface (may be connected in a daisy chain configuration)
- Equipped with Arduino UNO R3 connectors
- Layout compatible with ST morpho connectors

Key Products on board

L6470
Fully integrated microstepping motor driver with motion engine and SPI

ST1S14
Up to 3 A step down switching regulator

SMAJ48A
Transil

STPS1L60
Low Drop Power Schottky Rectifier

Latest info available at www.st.com
X-NUCLEO-IHM02A1

(*) only Arduino is mounted by default

life.augmented

STM32L4 Discovery Board for IoT node (B-L475E-IOT01A)

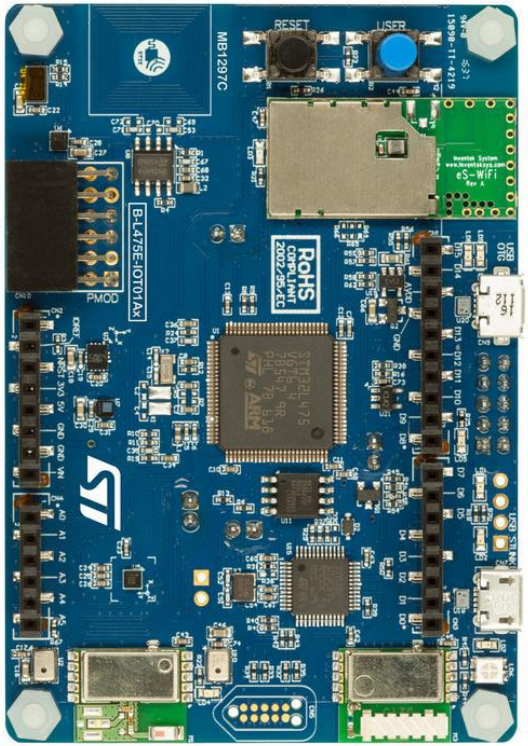
Hardware Overview (6/6)

STM32L4 Discovery Board for IoT node (B-L475E-IOT01A) Hardware Description

The STM32L4 Discovery kit for the IoT node (B-L475E-IOT01A) allows users to develop applications with direct connection to cloud servers. The STM32L4 Discovery kit enables a wide diversity of applications by exploiting low-power multilink communication (BLE, Sub- GHz), multiway sensing (detection, environmental awareness) and ARM® Cortex®-M4 core-based STM32L4 Series features. Arduino™ Uno V3 and PMOD connectivity provide unlimited expansion capabilities with a large choice of specialized add-on boards.

Key Product on board

- Ultra-low-power STM32L4 Series MCUs based on ARM® Cortex® -M4 core with 1 Mbyte of Flash memory and 128 Kbytes of SRAM, in LQFP100 package
- Bluetooth® V4.1 module (SPBTLE-RF)
- Sub-GHz (868 or 915 MHz) low-power-programmable RF module (SPSGRF-868 or SPSGRF-915)
- Wi-Fi® module Inventek ISM43362-M3G-L44 (802.11 b/g/n compliant)
- Dynamic NFC tag based on M24SR with its printed NFC antenna
- 2 digital omnidirectional microphones (MP34DT01)
- Capacitive digital sensor for relative humidity and temperature (HTS221)
- High-performance 3-axis magnetometer (LIS3MDL), 3D accelerometer and 3D gyroscope (LSM6DSL), 260-1260 hPa absolute digital output barometer (LPS22HB), Time-of-Flight and gesture-detection sensor (VL53L0X)
- USB OTG FS with Micro-AB connector
- Expansion connectors: Arduino™ Uno V3, PMOD
- Flexible power-supply options: ST LINK USB VBUS or external sources
- On-board ST-LINK/V2-1 debugger/programmer with USB re-enumeration capability: mass storage, virtual COM port and debug port



Latest info available at www.st.com
B-L475E-IOT01A

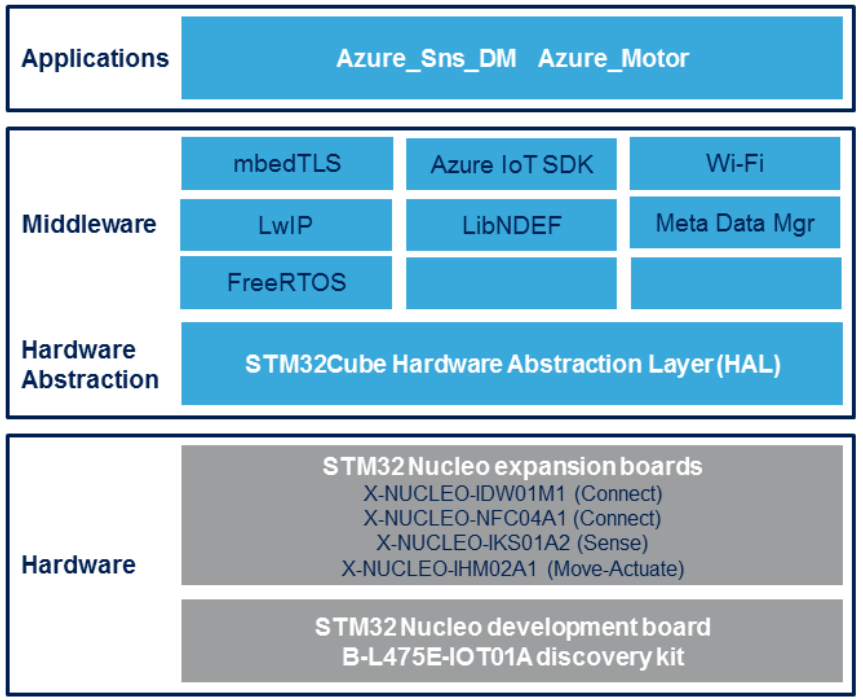
FP-CLD-AZURE1 Software Description

FP-CLD-AZURE1 is an STM32 ODE Function Pack. Thanks to this package you can directly connect your IoT sensor node to the Microsoft Azure IoT, transmit sensors data and receive command from Cloud applications.

Key features

- Complete firmware to safely connect an IoT node with sensors and actuators to Microsoft Azure IoT, using Wi-Fi or Ethernet communication technology. Two sample applications for data telmetry/device management (**Azure_Sns_DM**) and for motor control (**Azure_Motor**)
- Middleware libraries featuring the Microsoft Azure IoT software development kit, Wi-Fi and NFC connectivity, Motor Control, transport-level security (mbedTLS), Real-time Operating System (FreeRTOS), and meta-data management
- Ready-to-use binaries to connect the IoT node to a web dashboard running on Microsoft Azure, for sensor data visualization, actuators control, and device management (FOTA)
- Sample implementations available for STM32L4 Discovery Kit for IoT node (B-L475E-IOT01A) with and without X-NUCLEO-IHM02A1, or for X-NUCLEOIKS01A2, X-NUCLEO-IDW01M1, X-NUCLEO-IHM02A1 and X-NUCLEONFC04A1, when connected in different combinations to a NUCLEO-F401RE, a NUCLEO-L476RG or a NUCLEO-F429ZI development board
- Easy portability across different MCU families, thanks to STM32Cube
- Free, user-friendly license terms
- STM32 Nucleo is Microsoft Azure certified for IoT (for more information on Microsoft Azure Certification please visit http://azure.com/certifiedforiot)

Overall Software Architecture



Latest info available at www.st.com

FP-CLD-AZURE1

life.augmented

Quick Start Guide Contents

FP-CLD-AZURE: STM32 ODE function pack for IoT node with Wi-Fi or Ethernet, NFC and sensors, connected to Microsoft Azure cloud

Hardware and Software overview

Setup & Application Examples

Documents & Related Resources

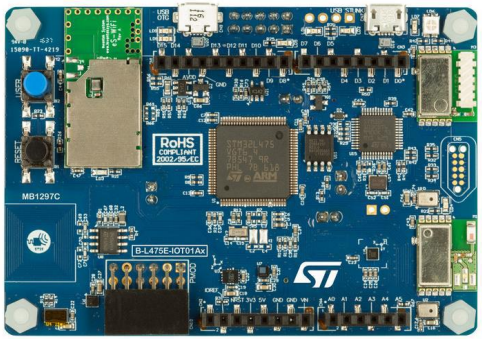
STM32 Open Development Environment: Overview

Setup & Application Examples (Azure_Sns_DM)

HW prerequisites for B-L475E-IOT01A

- 1x B-L475E-IOT01A development board
- NFC-enabled Android™ device (optional)
- Laptop/PC with Windows 7, 8 or 10
- 1 x microUSB cable
- Wi-Fi Router or access to a Wi-Fi network

MicroUSB Cable

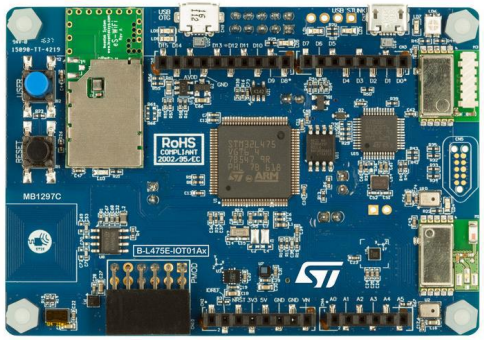


B-L475E-IOT01A

Setup & Application Examples (Azure_Motor)

HW prerequisites for B-L475E-IOT01A (1/2)

- 1x B-L475E-IOT01A development board
- 1x two axis stepper motor driver expansion board (**X-NUCLEO-IHM02A1**): **HW modification required**
- 1x Dynamic NFC tag expansion board expansion board for STM32 Nucleo (**X-NUCLEO-NFC04A1,** optional)
- NFC-enabled Android™ device (optional)
- Laptop/PC with Windows 7, 8 or 10
- 1 x microUSB cable
- Wi-Fi Router or access to a Wi-Fi network



B-L475E-IOT01A

MicroUSB Cable

X-NUCLEO-IHM02A1

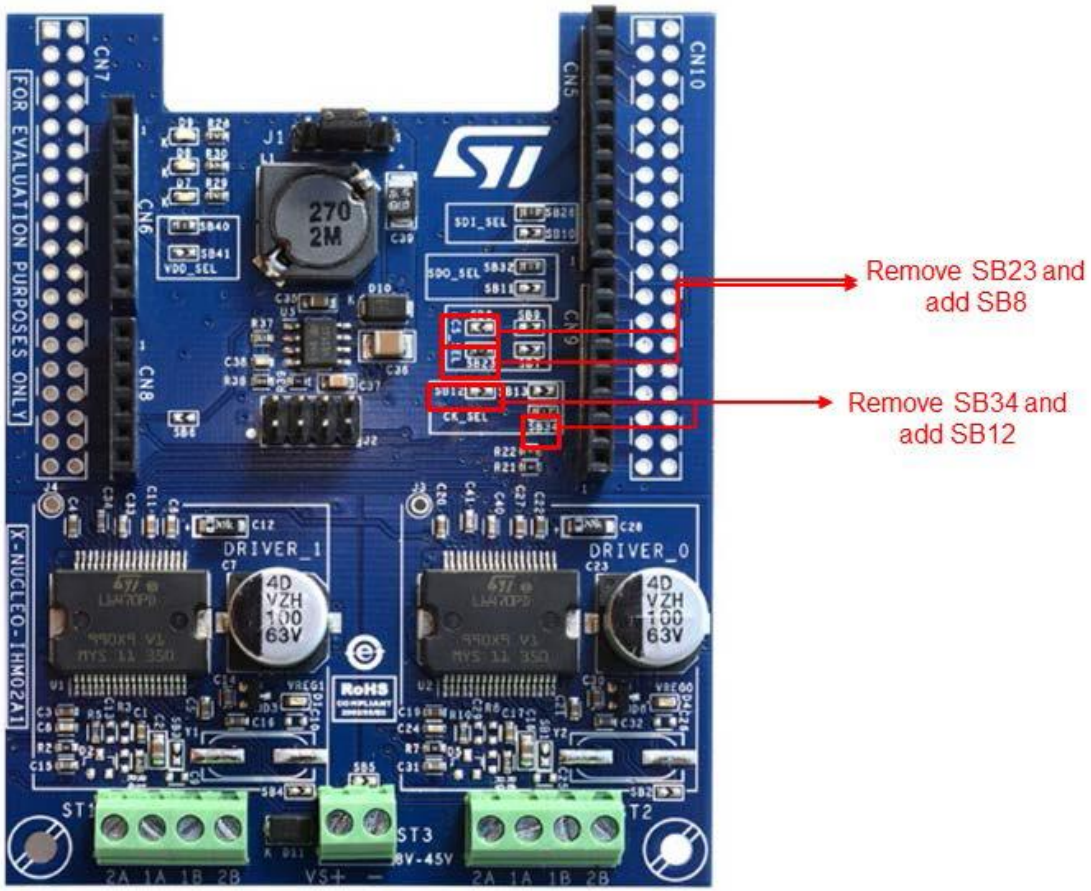
X-NUCLEO-NFC04A1

life.augmented

Setup & Application Examples (Azure_Motor)

HW prerequisites for B-L475E-IOT01A (2/2)

- X-NUCLEO-IHM01A1 can be used with STM32L4 Discovery Kit for IoT node (B-L475-IOT01A), with the following modifications in the hardware:
  - Remove Solder Bridge SB34 and add Solder Bridge SB12
  - Remove Solder Bridge SB23 and add Solder Bridge SB8



Software and Other prerequisites

- **STM32 ST-Link Utility**
  - Download and install STSW-LINK004 from www.st.com

- **FP-CLD-AZURE1**
  - Download FP-CLD-AZURE1 package from www.st.com, copy the .zip file content into a folder on your PC. The package contains binaries and source code with project files (Keil, IAR, System Workbench) based on NUCLEO-F401RE, NUCLEO-L476RG, NUCLEO-F429ZI and B-L475E-IOT01A.

- **Serial line monitor,** e.g. TeraTerm (https://ttssh2.osdn.jp/)

- To write/read NFC tag
  - Any **Android application capable** to read/write NFC tag (i.e. ST25 NFC https://play.google.com/store/apps/details?id=com.st.st25nfc)

- To register a new device in Azure IoT Hub and test FP-CLD-AZURE1 with custom Azure account:
  - An active account on Microsoft Azure (https://azure.microsoft.com/en-us/pricing/free-trial/)
  - **Microsoft Device Explorer utility** (https://github.com/Azure/azure-iot-sdk-csharp/tree/master/tools/DeviceExplorer) or **iothub-explorer** (https://github.com/Azure/iothub-explorer). DeviceExplorer utility is used as reference in this Quickstart Guide.

- To test pre-compiled binaries with STM32ODE web dashboard**: Chrome** web browser (https://www.google.com/chrome/); tested with Chrome version v56.0.2924.76

FP-CLD-AZURE1. Sample applications

Start coding in just a few minutes

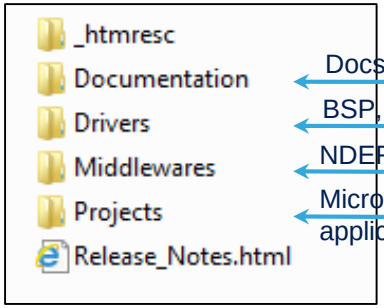
life.augmented
www.st.com

1 www.st.com/stm32ode

2 Select Function Pack:
FP-CLD-AZURE1

3 Download & unpack

FP-CLD-AZURE1 package structure

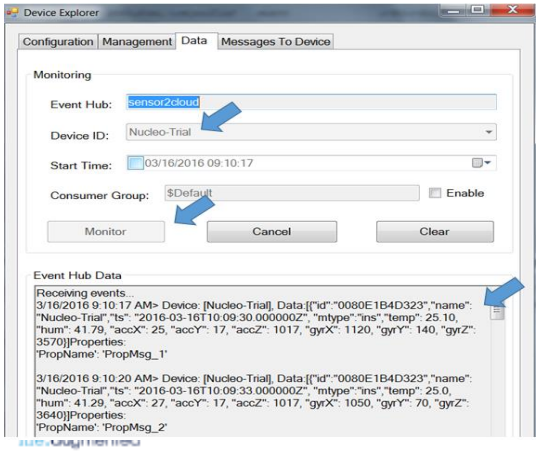


4 Register your device in Azure IoT, provision credentials in source code and recompile the project according to the selected IDE. Alternatively use available pre-compiled binaries

KEIL Tools by ARM · IAR SYSTEMS · Open STM32 Tools

5

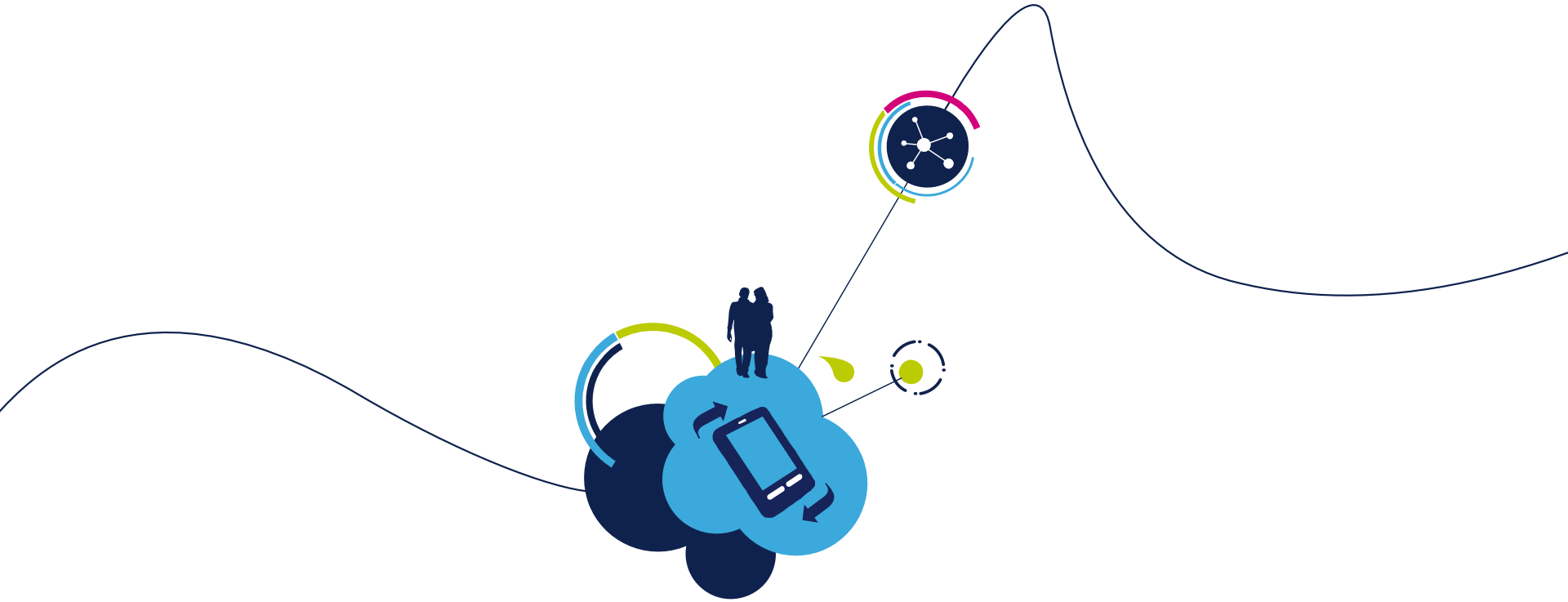
6 Visualize log of sensors data and control the device



FP-CLD-AZURE1. Sample applications for different platforms

Azure_Sns_DM / Azure_Motor

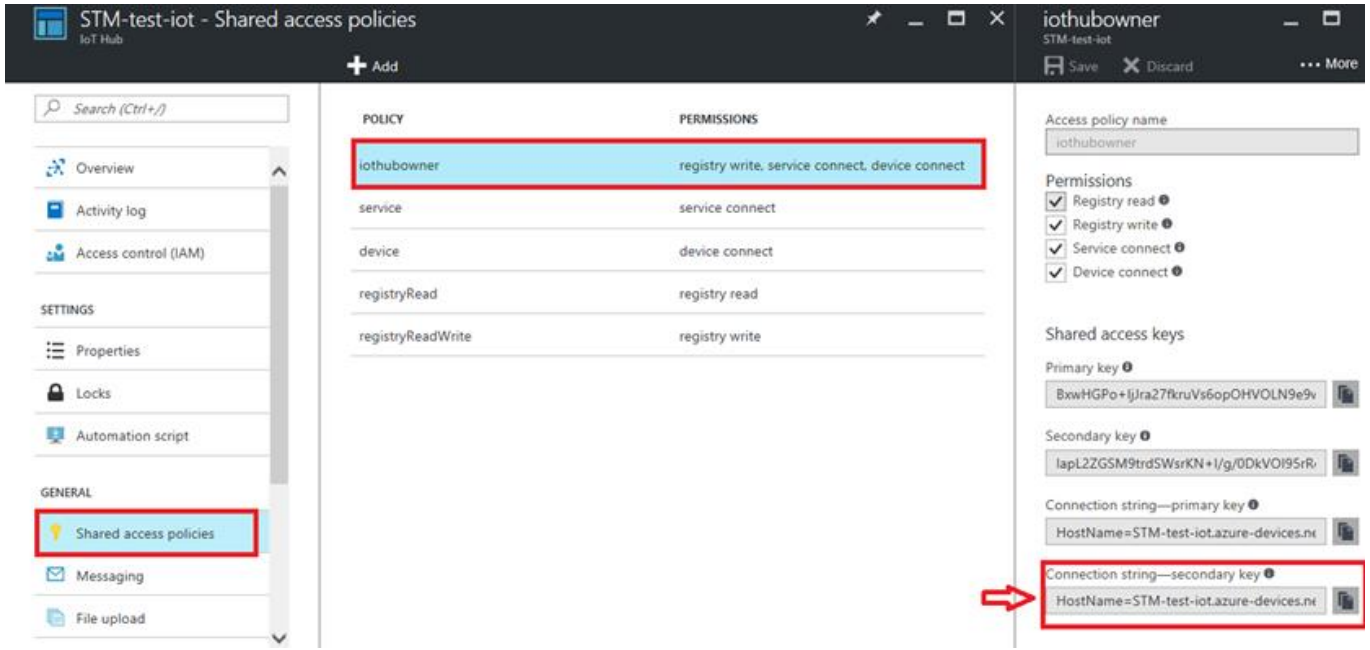
Application name and platform supported	Main Features	Precompiled binary name for usage with personal Azure account	Precompiled binary name for usage with personal IoT Central account	Precompiled binary name for usage with ST Web dashboard
Azure_Sns_DM (NUCLEO-F401RE)	Sensors data telemetry / device control	Azure_Sns_DM.bin	X	Azure_Sns_DM_Web.bin
Azure_Sns_DM (NUCLEO-L476RG)	Sensors data telemetry / device control / FOTA	Azure_Sns_DM_BL.bin	Azure_Sns_DM_BL_IoTCentral.bin	Azure_Sns_DM_BL_Web.bin
Azure_Sns_DM (NUCLEO-F429ZI)	Sensors data telemetry / device control / FOTA	Azure_Sns_DM_BL.bin	X	Azure_Sns_DM_BL_Web.bin
Azure_Sns_DM (B-L475E-IOT01A)	Sensors data telemetry / device control / FOTA	Azure_Sns_DM_BL.bin	Azure_Sns_DM_BL_IoTCentral.bin	Azure_Sns_DM_BL_Web.bin
Azure_Motor (NUCLEO-L476RG)	Device control / Motor Control	Azure_Motor.bin	X	Azure_Motor_Web.bin
Azure_Motor (B-L475E-IOT01A)	Device control / Motor Control	Azure_Motor.bin	X	Azure_Motor_Web.bin



Test FP-CLD-AZURE1 with personal Azure account

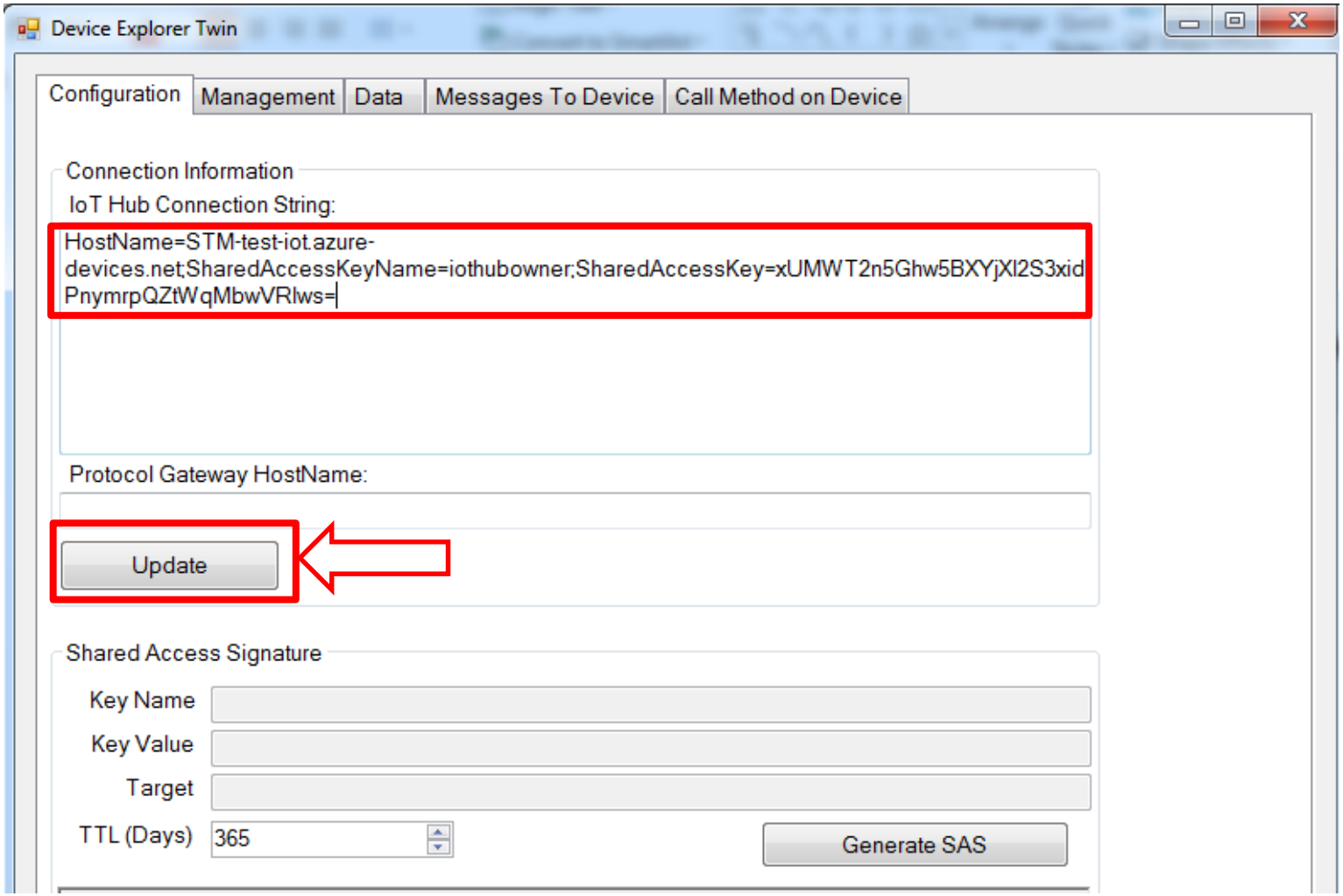
Create Azure IoT Hub and generate device connection string (1/4)

- Enter Azure Portal with your credentials (https://azure.portal.com)
- Create an instance of the Azure IoT Hub following instruction provided here: https://github.com/Azure/azure-iot-device-ecosystem/blob/master/setup_iothub.md
- Note down in a text editor your *IoT Hub connection string*



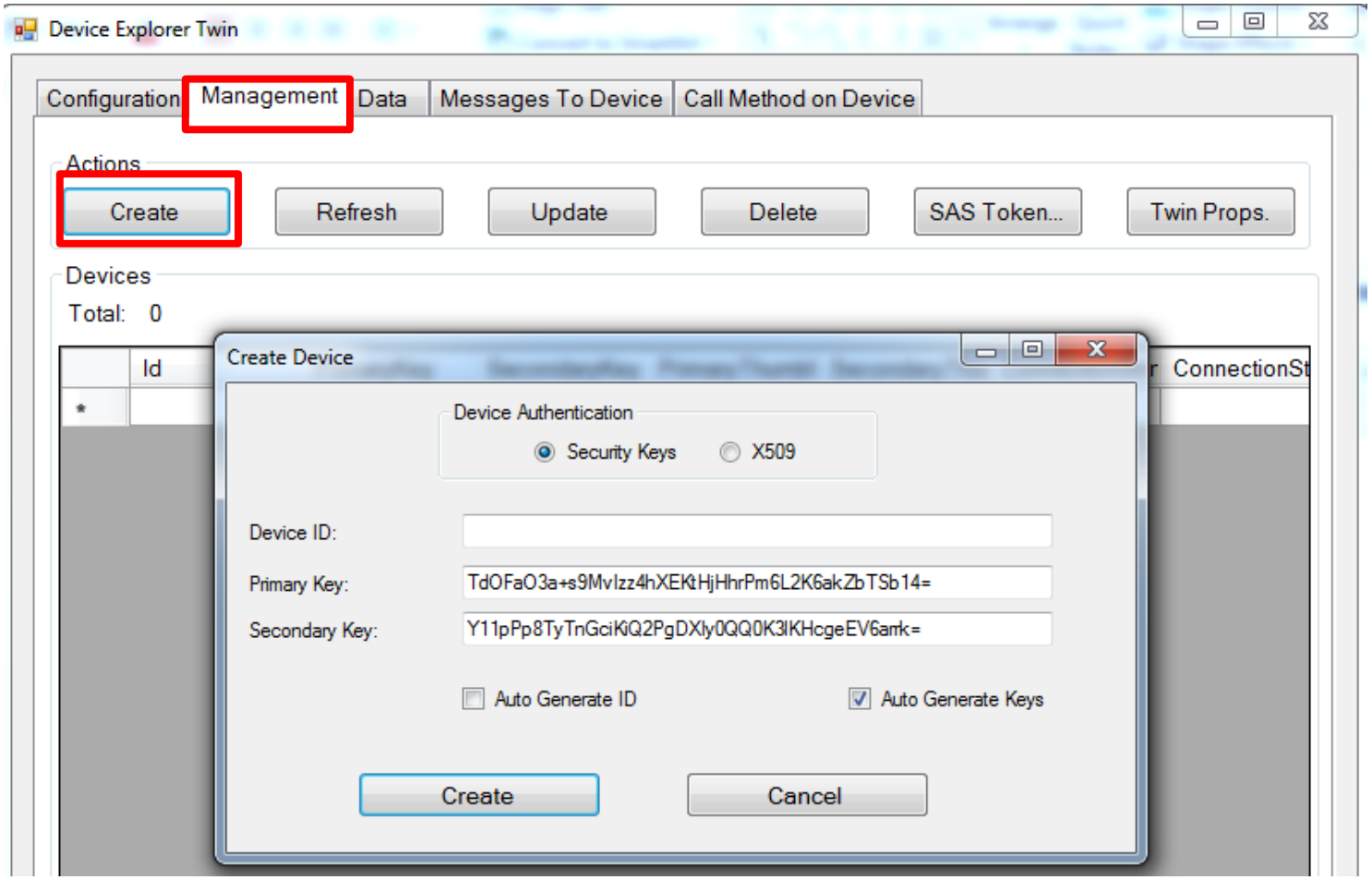
Create Azure IoT Hub and generate device connection string (2/4)

- Launch DeviceExplorer; open **Configuration** tab, paste the *IoT Hub connection string* and then click on **Update**



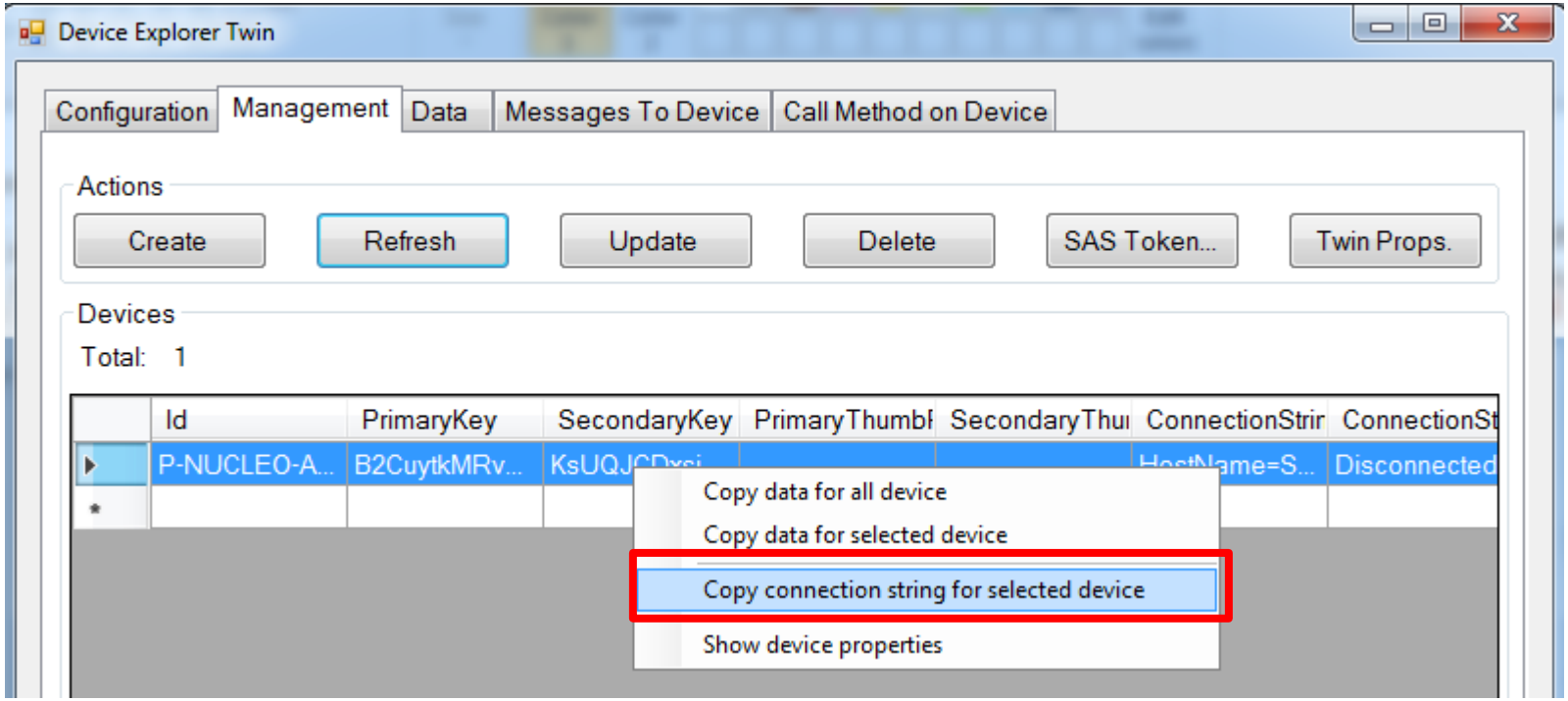
Create Azure IoT Hub and generate device connection string (3/4)

- In **Management** tab click on Create and then insert a unique device ID for your STM32 Nucleo board. An entry for your device is created and listed.



Create Azure IoT Hub and generate device connection string (4/4)

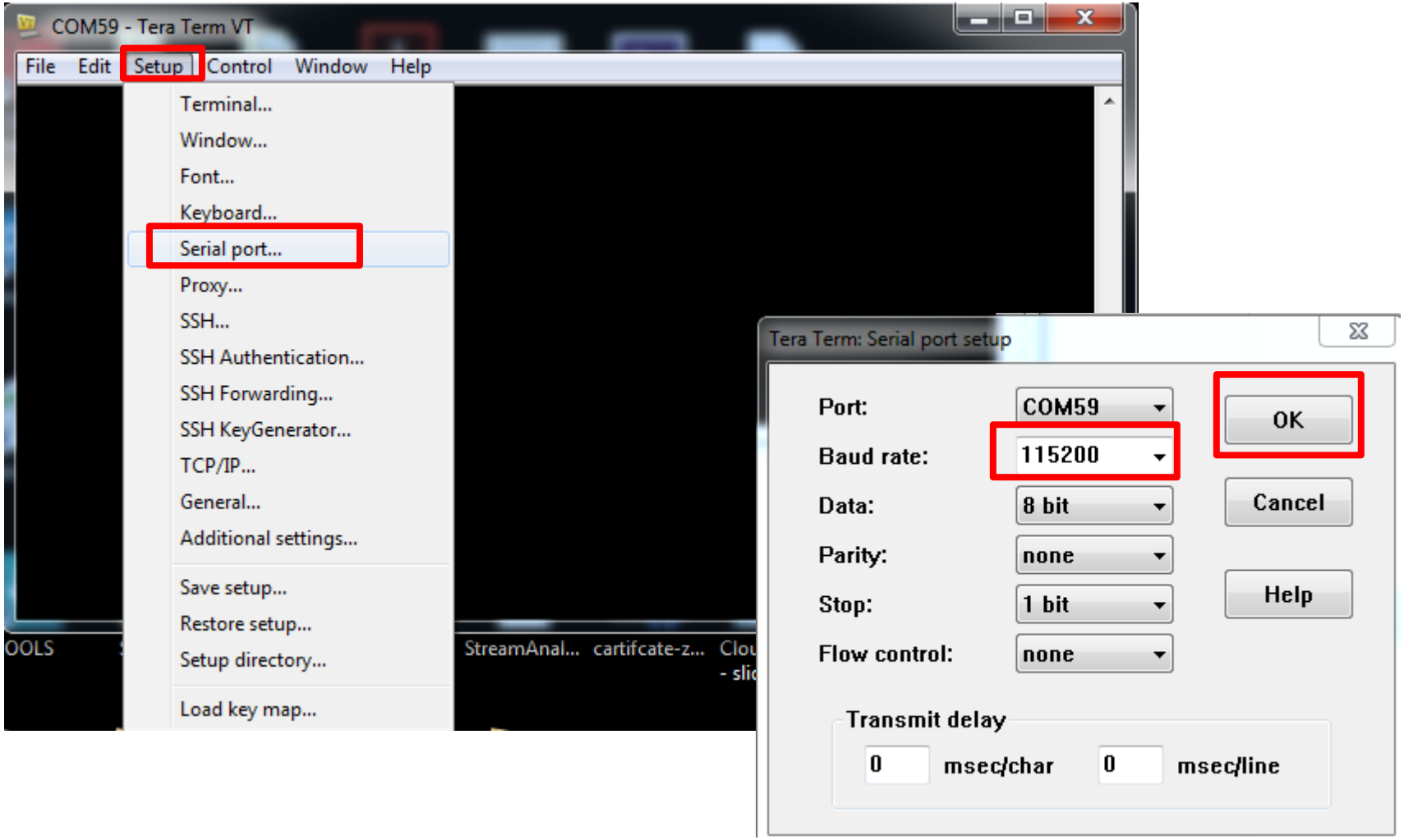
- In **Management** tab select the device just created, use the mouse right click for context menu then select "**Copy connection string for selected device**". Note down the connection string in a text editor for later usage.



FP-CLD-AZURE1. Step by step setup in details

Launch sample application. Configure Serial Terminal

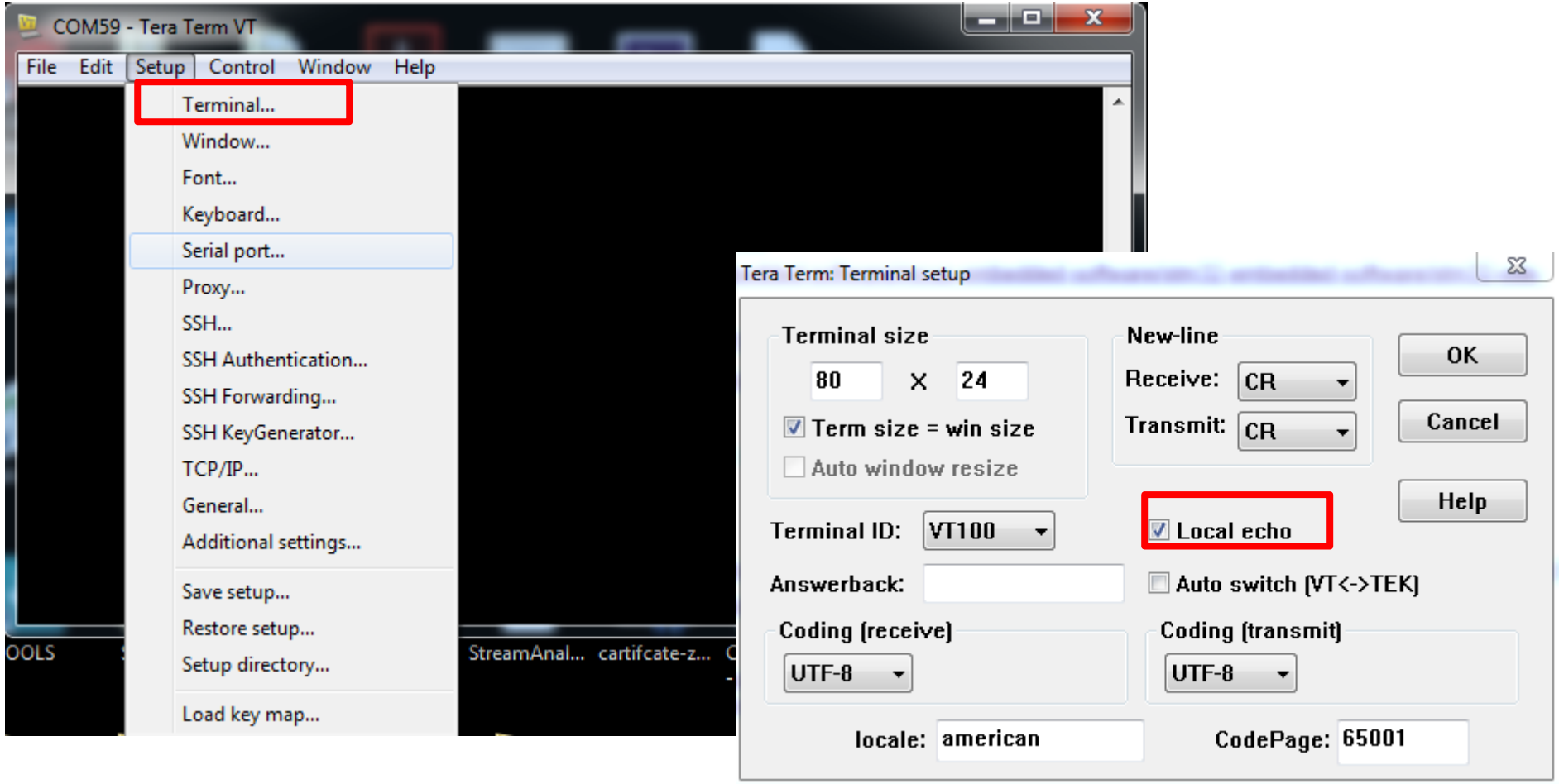
- Open serial terminal then configure baud rate speed to 115200 (**Setup → Serial port** in TeraTerm).



FP-CLD-AZURE1. Step by step setup in details

Launch sample application. Configure Serial Terminal

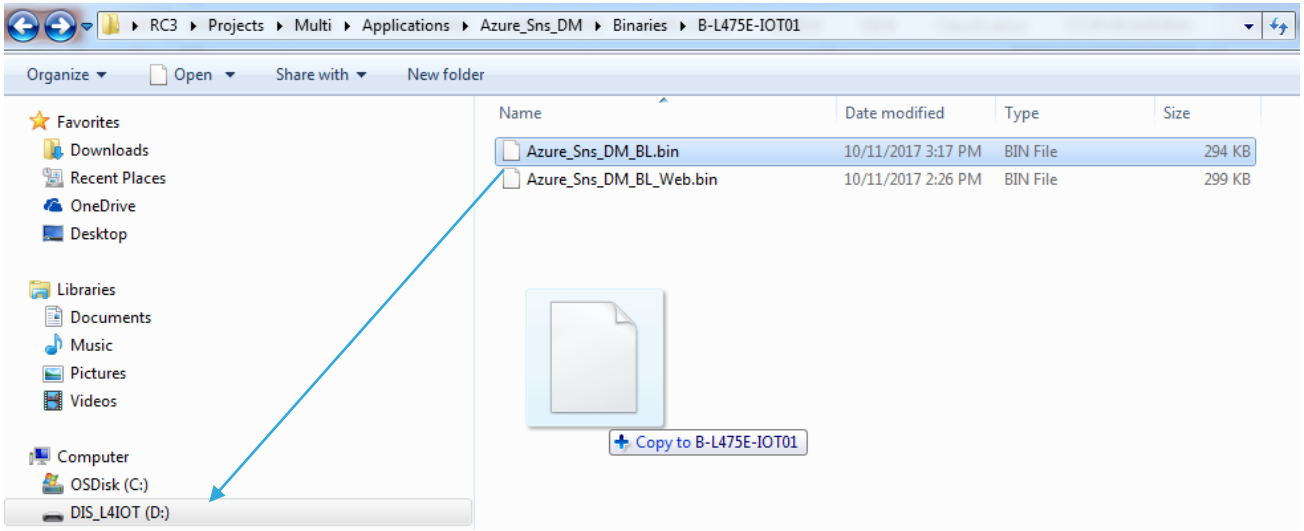
- Enable local echo in Terminal configuration (**Setup → Terminal** in TeraTerm).



FP-CLD-AZURE1. Step by step setup in details

Launch sample application. Use pre-compiled binaries

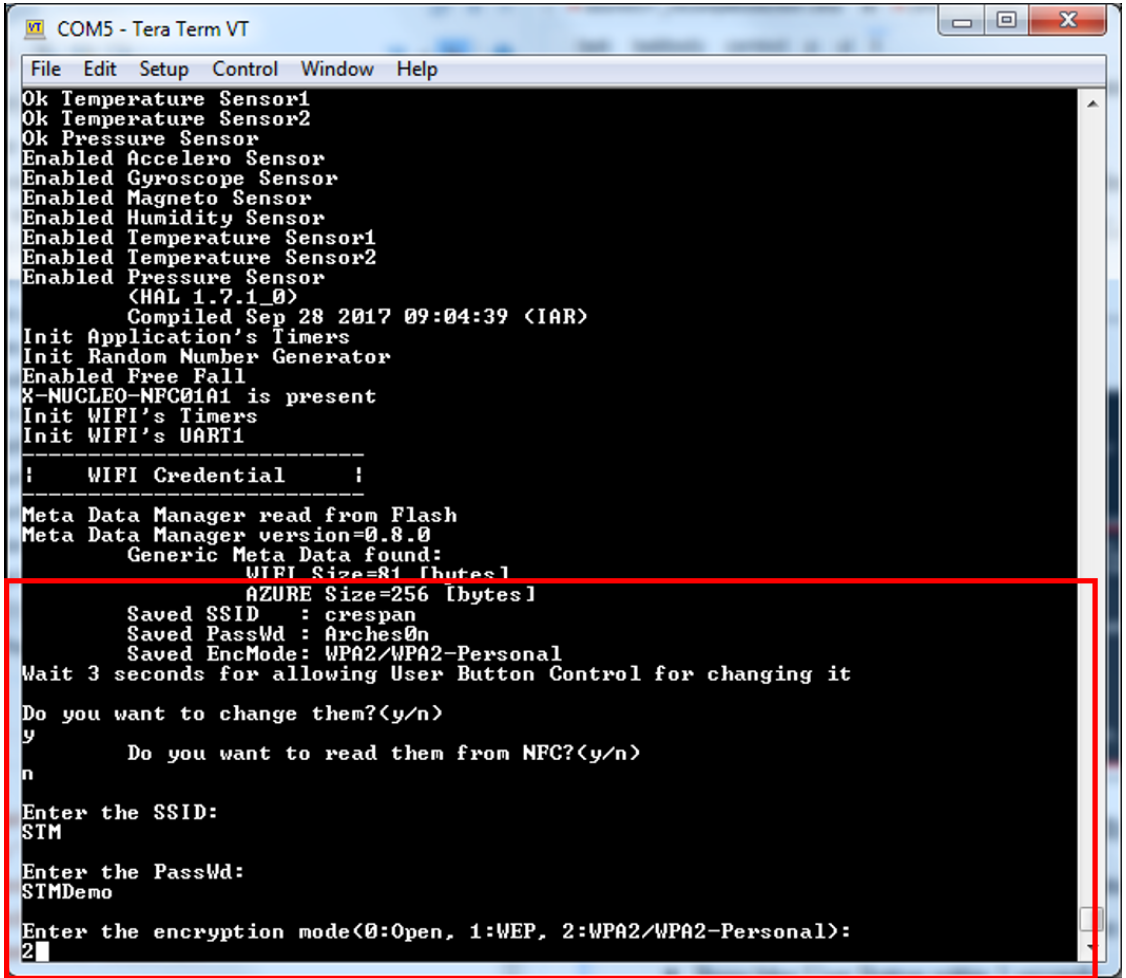
- Depending on application and hardware combination used, configurable pre-compiled binaries can be found inside folders:
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F401RE-Nucleo/Azure_Sns_DM.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F476RG-Nucleo/Azure_Sns_DM_BL.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F429ZI-Nucleo/Azure_Sns_DM_BL.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/B-L475E-IOT01/Azure_Sns_DM_BL.bin`
  - `Projects/Multi/Applications/Azure_Motor/Binaries/STM32F476RG-Nucleo/Azure_Motor.bin`
  - `Projects/Multi/Applications/Azure_Motor/Binaries/B-L475E-IOT01/Azure_Motor.bin`
- To start the application, simply connect to the laptop your board and drag the binary



FP-CLD-AZURE1. Step by step setup in details

Configure Wi-Fi parameters (NUCLEO-L476RG/F401RE, B-L475E-IOT01A)

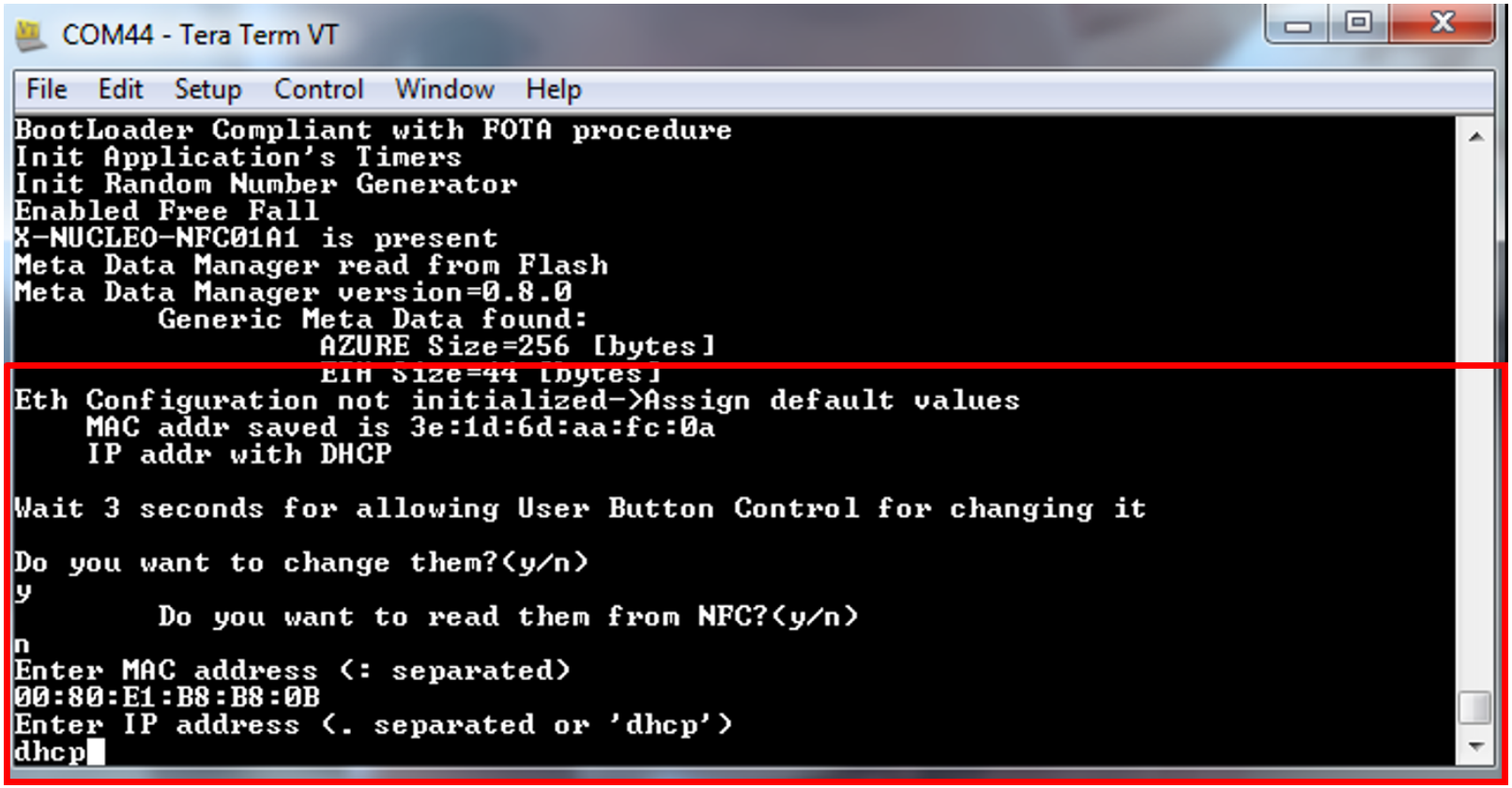
- Open serial terminal to visualize the log of messages
- Default values for Wi-Fi SSID and PWD can be modified after pressing once USER button within 3 seconds. Press n when asked to read from NFC, then enter SSID, PWD and Encryption mode (WPA2/WPA2-Personal) when requested



FP-CLD-AZURE1. Step by step setup in details

Configure Ethernet parameters (NUCLEO-F429ZI)

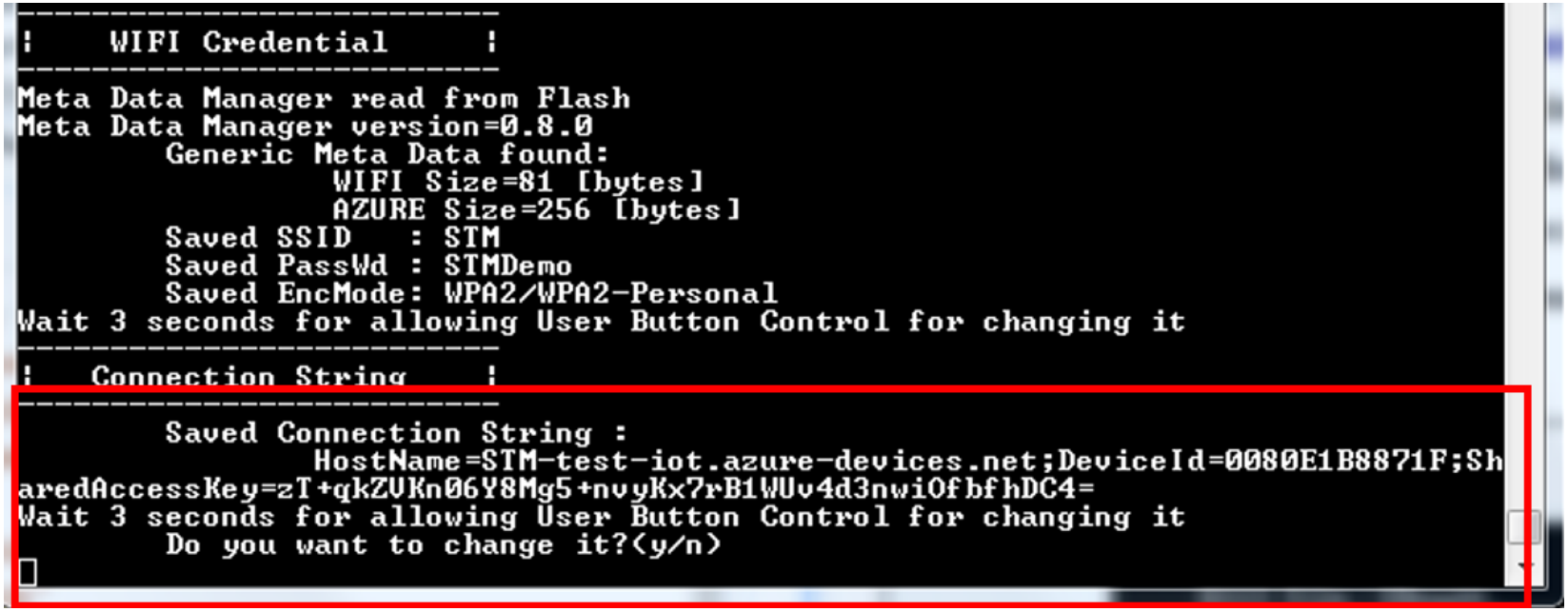
- Open serial terminal to visualize the log of messages
- Default Ethernet configuration can be modified after pressing once USER button within 3 seconds. Press n when asked to read from NFC; enter the MAC address and then dhcp for automatic IP configuration or static IP and gateway addresses



FP-CLD-AZURE1. Step by step setup in details

Configure device connection string (all platforms)

- Once completed Wi-Fi or Ethernet configuration, you need to enter the Azure IoT Hub device connection string (default is NULL)
- Once inserted, the connection string is save into FLASH memory. Press the blue **User Button** within 3 seconds to change it.



FP-CLD-AZURE1. Step by step setup in details

Optional: use NFC for device configuration. Wi-Fi parameters

- Launch the ST25 NFC mobile app and click the Compose NDEF button, then select in menu the Wi-Fi option
- Insert SSID and Password, then approach the mobile phone to the NFC tag and click on *Write to tag*

- Reset the board. Press the blue user button within 3 seconds, and press y when requested to read from NFC.

FP-CLD-AZURE1. Step by step setup in details

Optional: use NFC for device configuration. Ethernet parameters

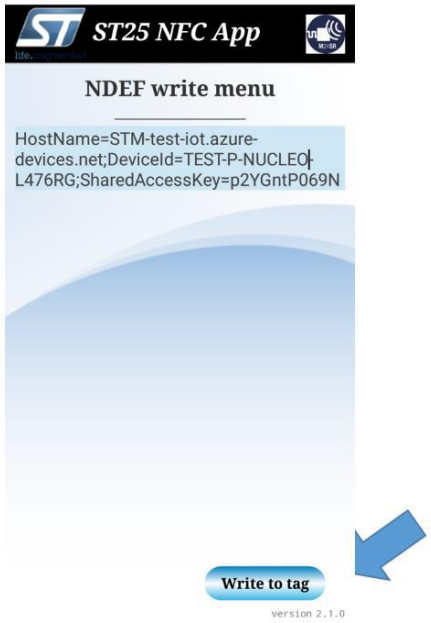
- Launch the ST25 NFC mobile app and click the Compose NDEF button.
- Click on the TEXT button and enter the MAC address as "MAC" followed sequence of six colon-separated hexadecimal numbers and the IP address as "IP" followed by either 4 dot-separated decimal numbers (in case of static pre-assigned address) or 'dynamic' or 'dhcp' (in case of DHCP-assigned address). In case of static IP address you have to add a further line containing **Gateway** followed by 4 dot-separated decimal numbers. Then click on the "Write to tag" button with your mobile phone near the NFC expansion board on your system.

- Reset the board. Press the blue user button within 3 seconds, and press y when requested to read from NFC.

FP-CLD-AZURE1. Step by step setup in details

Optional: use NFC for device configuration. Device connection string

- After Wi-Fi/Ethernet configuration, press again the blue user button within 3 seconds to update the IoT Hub device connection string.
- Go back to your mobile application, click on the **Text** button and paste the IoTHub connection string



- Reset the board. In the serial terminal, press **y** when asked to read connection string from NFC

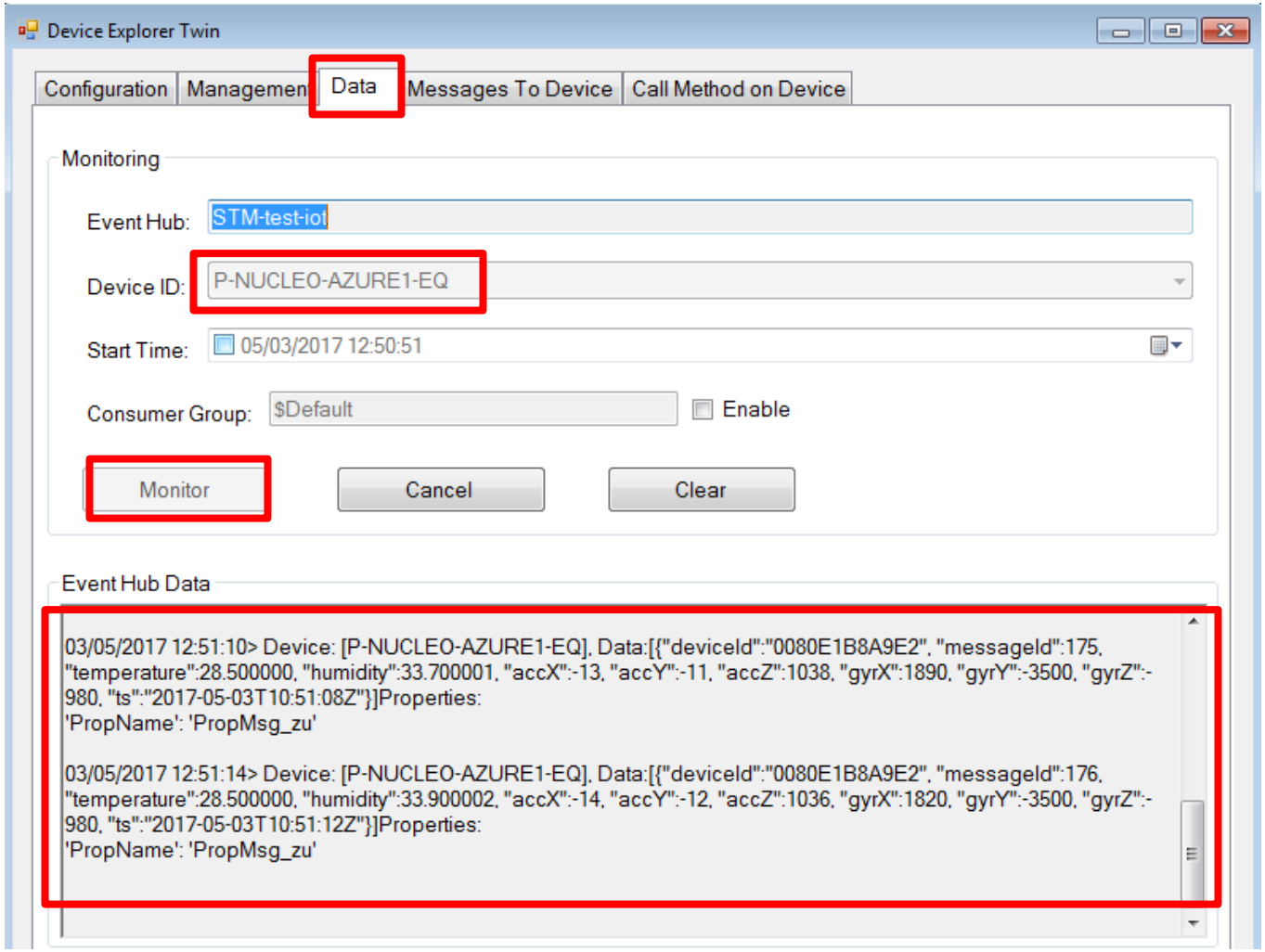
FP-CLD-AZURE1. Step by step setup in details
Visualize messages in DeviceExplorer (1/2)

- After device configuration, the sample application contact the IoT Hub
- Each 2 seconds, a message is created containing sensors data and transmitted to the IoTHub using MQTT protocol. For each message transmitted, a confirmation acknowledge is received back by the device

```
IoTHubClient_LL_SendEventAsync accepted message [1] for transmission to IoT Hub.
IoTHubClient_LL_SendEventAsync accepted message [2] for transmission to IoT Hub.
IoTHubClient_LL_SendEventAsync accepted message [3] for transmission to IoT Hub.
-->DeviceTwin CallBack [1]: Status code = 204
Confirmation received for message [1] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
Confirmation received for message [2] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [4] for transmission to IoT Hub.
Confirmation received for message [3] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [5] for transmission to IoT Hub.
Confirmation received for message [4] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [6] for transmission to IoT Hub.
Confirmation received for message [5] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [7] for transmission to IoT Hub.
Confirmation received for message [6] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [8] for transmission to IoT Hub.
Confirmation received for message [7] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
IoTHubClient_LL_SendEventAsync accepted message [9] for transmission to IoT Hub.
Confirmation received for message [8] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
```

FP-CLD-AZURE1. Step by step setup in details
Visualize messages in DeviceExplorer (2/2)

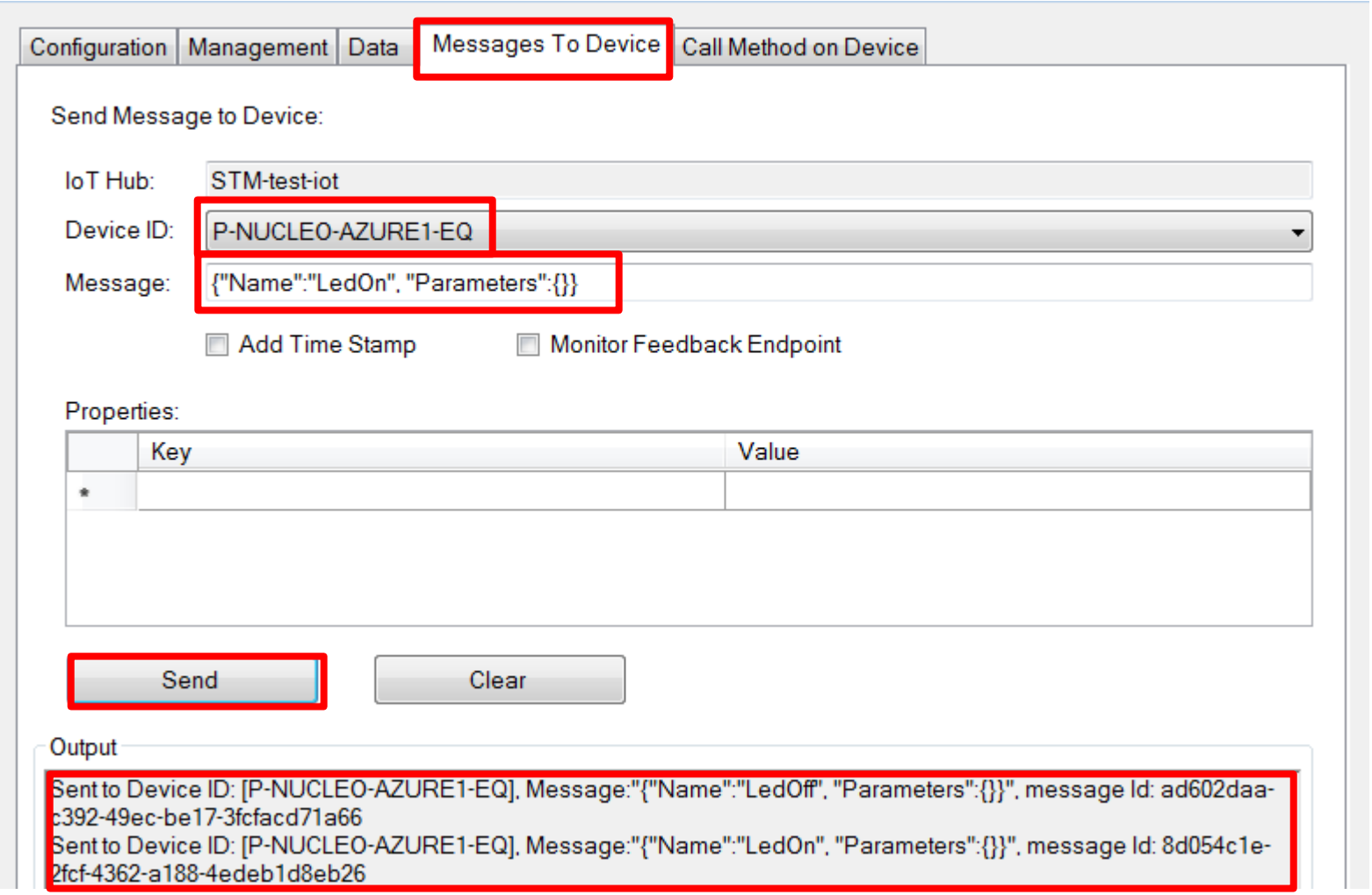
- Open DeviceExplorer; select **Data** tab, then in the drop down menu for **Device Id:** select the id of your registered device. Then click on **monitor** to start visualizing the log of messages received by IoTHub



FP-CLD-AZURE1. Step by step setup in details

Send Cloud to Device messages (1/3)

- Open Device Explorer; select **Message To Device** tab, then in the drop down menu for Device ID select the id of your registered device. Write a message, then click on **Send**. Message sent to device will be displayed in **Output** box



Send Cloud to Device messages (2/3)

- Each message received by the device from IoT Hub is printed over serial terminal.

- By default the application can interpret the following Cloud to Device messages:
  - **Pause**: pause the application
  - **Play**: restart the application
  - **LedOn/LedOff**: turn on/off LED2 onboard STM32-Nucleo
  - **LedBlink**: LED2 onboard STM32-Nucleo will blink for each message received.

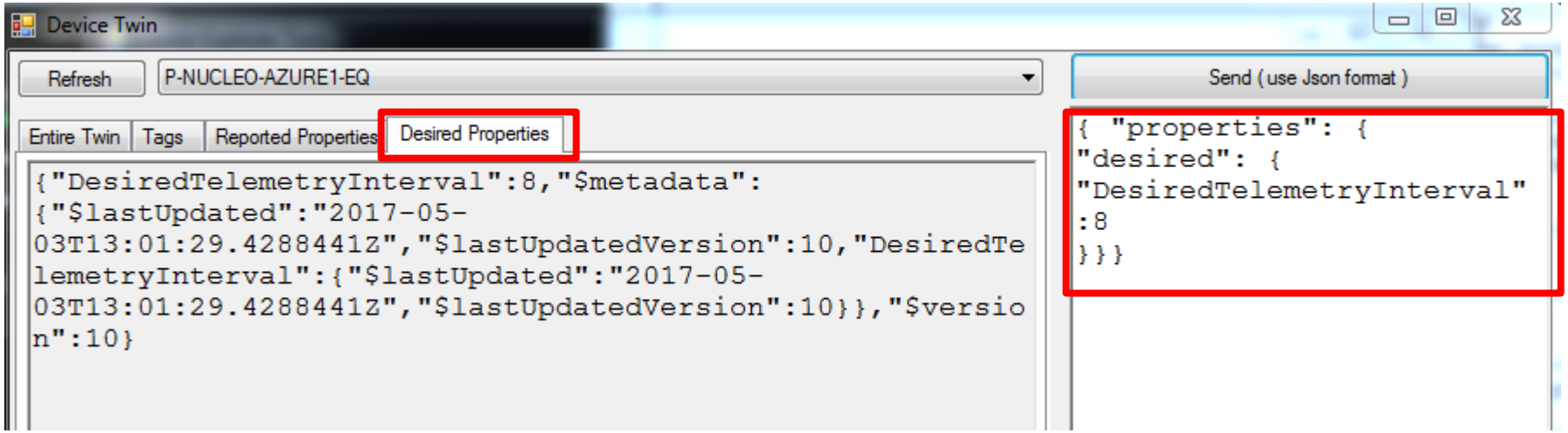
- Messages need to be typed in Device Explorer in the following format:
  - {"Name":“*Message*", "Parameters":{}}), i.e.
    - {"Name":"Pause", "Parameters":{}}
    - {"Name":“Play", "Parameters":{}}
    - …

Send Cloud to Device messages (3/3)

- For Azure_Motor application, some more messages are interpreted and used by X-NUCLEO-IHM02A1
  - **MoveMotor [MotorNum][Angle]**: moves the motor number [0 or 1] of a specific angle [0°-360°];
  - **RunMotor [MotorNum][Speed]**: starts the motor number [0 or 1] with a specific speed [1 - 10];
  - **ResetMotor [MotorNum]**: resets the position for motor number [0 or 1];
  - **GoHomeMotor [MotorNum]**: moves the motor number [0 or 1] to home position;
  - **ComplexMove [ComplexProgramString]**: sends a string containing combination of consecutives
- To learn more on the syntax for motor control commands, see the UM1963 at X-CUBE-SPN2
- Messages need to be typed in Device Explorer in the following format:
  - {"Name":“*Message*", "Parameters":{}}), i.e.
    - {"Name" : "MoveMotor", "Parameters" : {"MotorNum" : 1, "Angle" : 45}}
    - ...

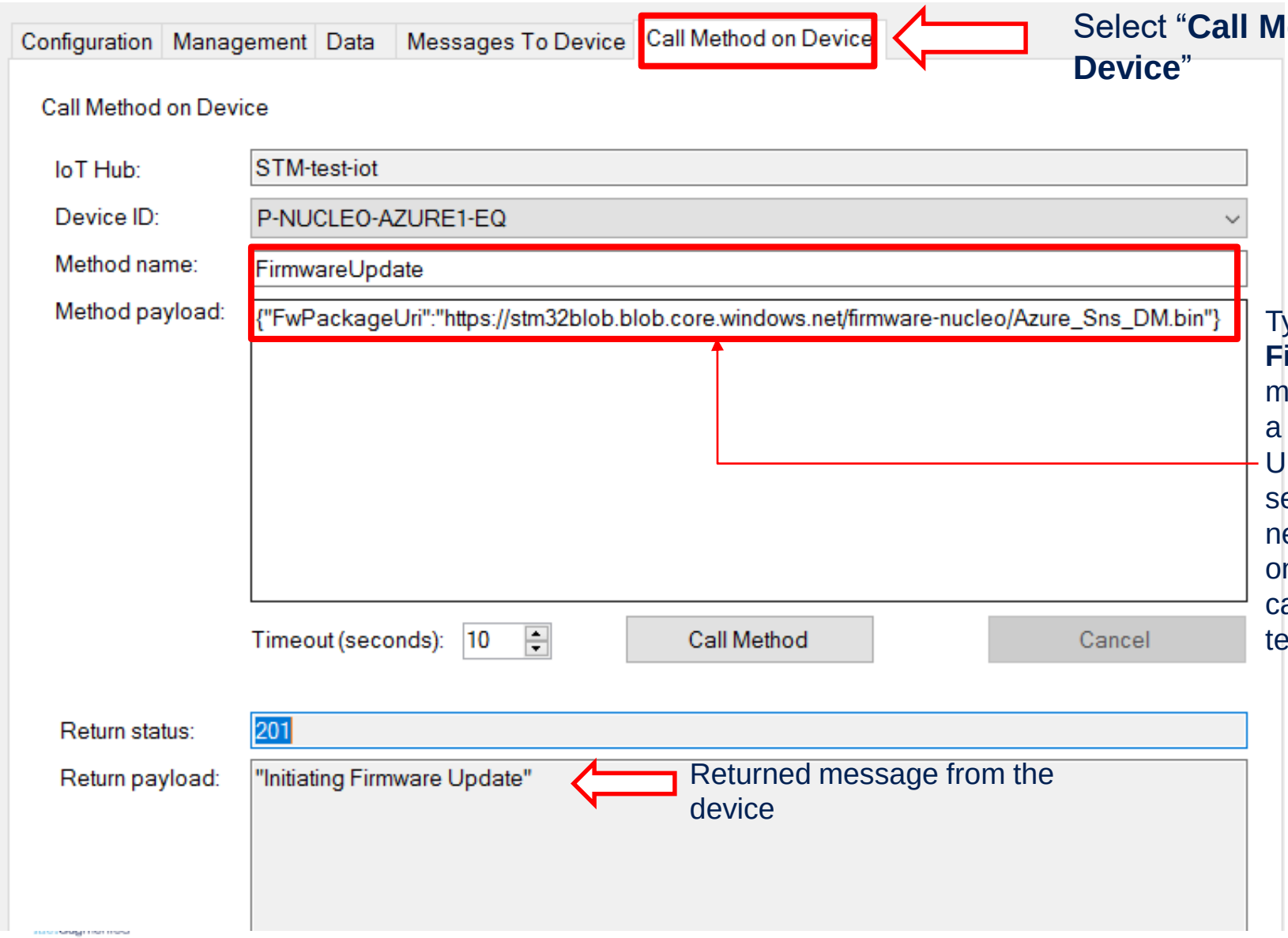
FP-CLD-AZURE1. Step by step setup in details
Change desired properties in DeviceExplorer

- In Device Explorer select **Management** tab, then **Twin Props** and the Id of your device
- In **Twin Props** select **Desired Properties** tab. Then type in the right window the new desired property
- Only `DesiredTelemetryInterval` is supported as desired property by the application



FP-CLD-AZURE1. Step by step setup in details

Call direct method to force Firmware Update over-the-air (NUCLEO-L476RG, NUCLEO-F429ZI, B-L475E-IOT01A) (1/2)



Select "**Call Method on Device**"

Type in **FirmwareUpdate** method adding as a parameter the URL of web server hosting the new binary (the one reported here can be used for testing)

Returned message from the device

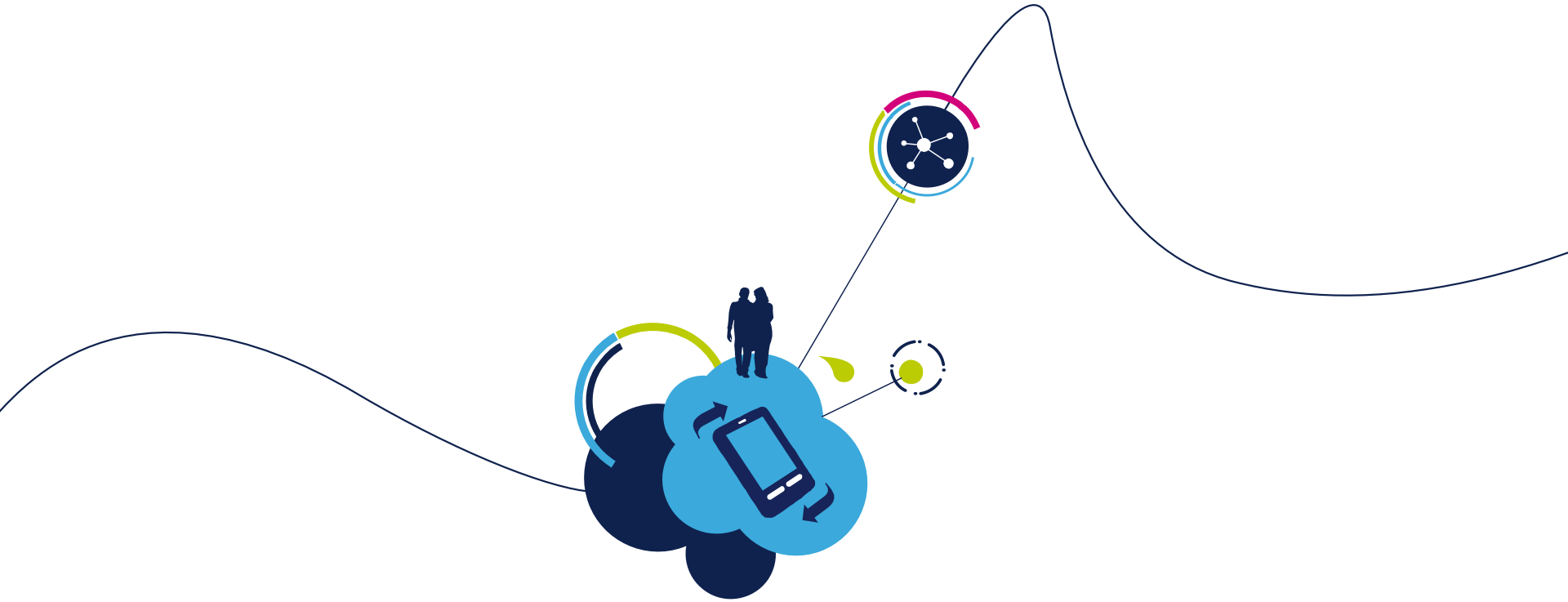
FP-CLD-AZURE1. Step by step setup in details

Call direct method to force Firmware Update over-the-air (NUCLEO-L476RG, NUCLEO-F429ZI, B-L475E-IOT01A) (2/2)


```
Device Method called
Device Method name:    FirmwareUpdate
Device Method payload: {"FwPackageUri":"https://stm32blob.blob.core.windows.net/firmware
ucleo/Azure_Sns_DM.bin"}
Received firmware update request. Use package at: [https://stm32blob.blob.core.windows.n
/firmware-nucleo/Azure_Sns_DM.bin]
Channel 1 for Timer 1 stopped
Download FOTA from: HostName=[stm32blob.blob.core.windows.net] Type=[Secure] port=[443]
le=[/firmware-nucleo/Azure_Sns_DM.bin]
Ok reported State [2]: Downloading
Confirmation received for message [16] with result = IOTHUB_CLIENT_CONFIRMATION_SUCCESS
-->DeviceTwin CallBack [2]: Status_code = 204
Ok io_interface_description
Ok xio_create
Ok xio_setoption
Ok xio_open
Ok xio_send HEAD Request
(Content-Length:) Full OTA size=310211
        PaddingBytes=5
        OTA Round=310216
Start FLASH Erase
End FLASH Erase 152 Pages of 2KB
Ok xio_send GET (000/1211) Request
Ok xio_send GET (001/1211) Request
Ok xio_send GET (002/1211) Request
Ok xio_send GET (003/1211) Request
```

Erase FLASH memory and download new firmware
When download is finished, restart the board and load the new firmware

life.augmented

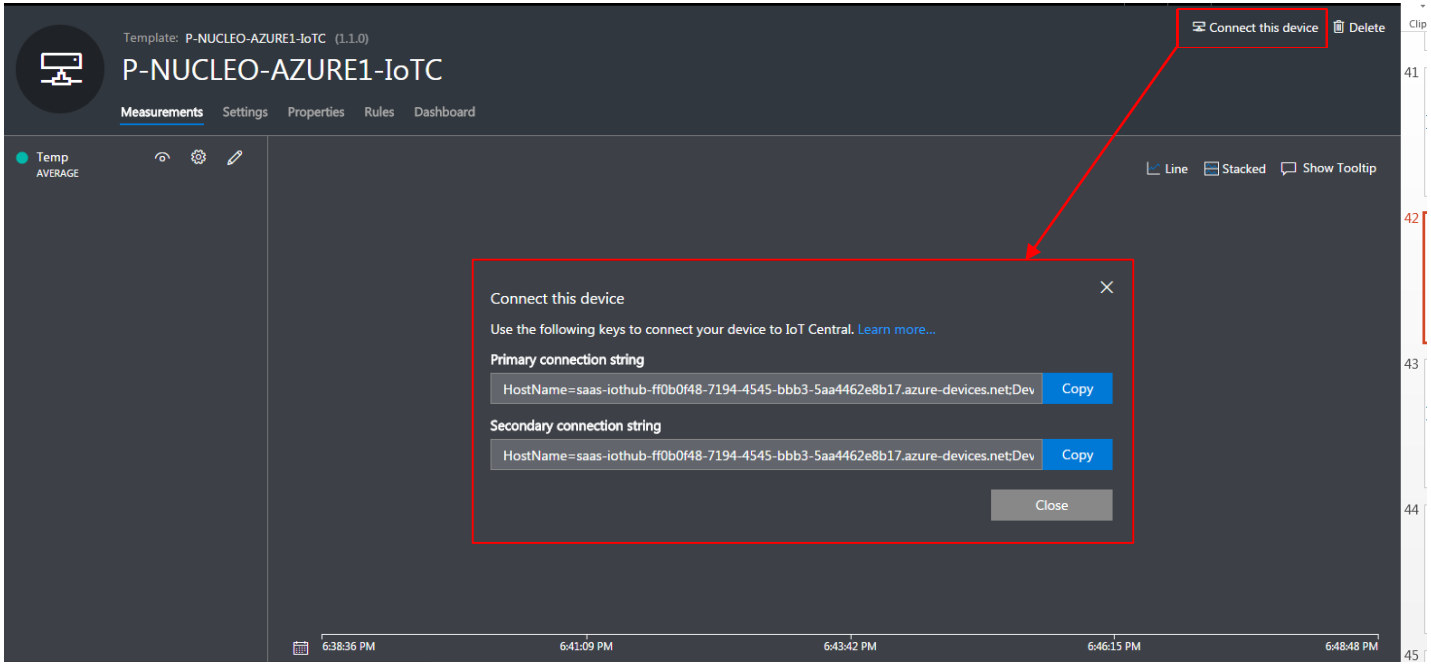


Test FP-CLD-AZURE1 with personal Microsoft IoT Central account

life.augmented

Use pre-compiled binaries for Microsoft IoT Central (NUCLEO-L476RG, B-L475E-IOT01A) (1/2)

- Microsoft IoT Central is a fully managed SaaS (software-as-a-service) based on Azure that simplify the development of IoT Solutions. Learn more on IoT Central at https://www.microsoft.com/en-us/iot-central
- Hands-on material to create a simple application in IoT Central is available at https://docs.microsoft.com/en-gb/microsoft-iot-central/tutorial-add-device.
- Once created the application, you can connect a device by simply clicking on **Connect this device** and then copying the resulting connection string

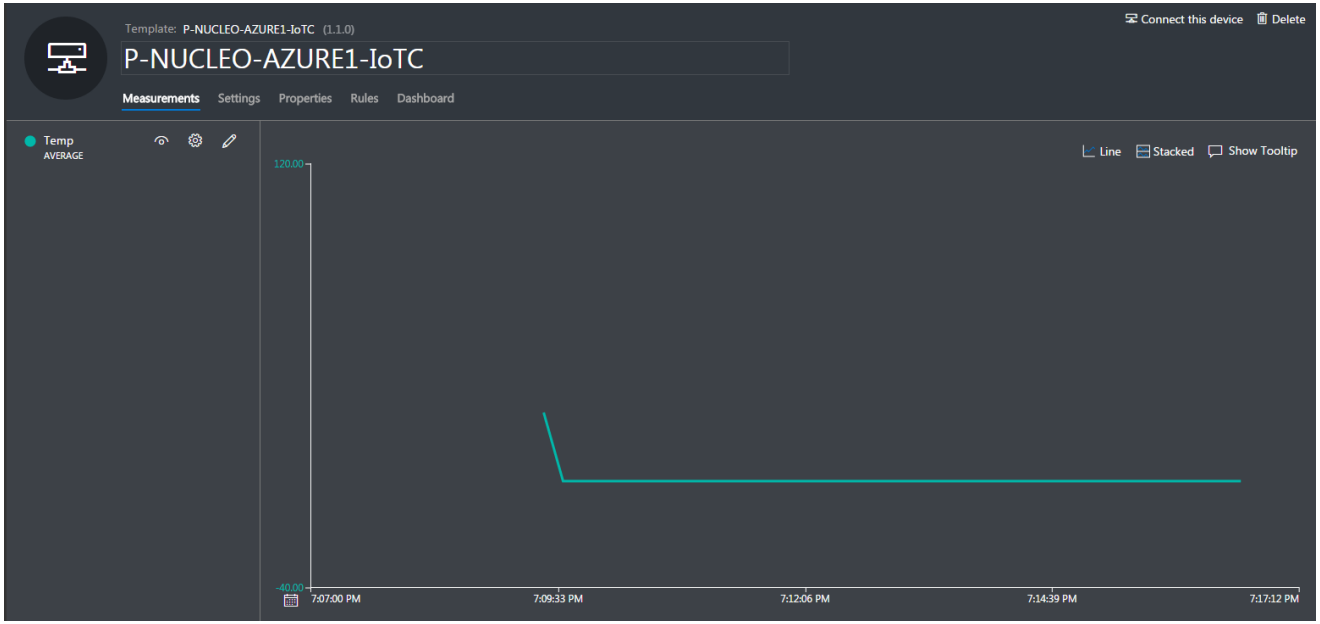


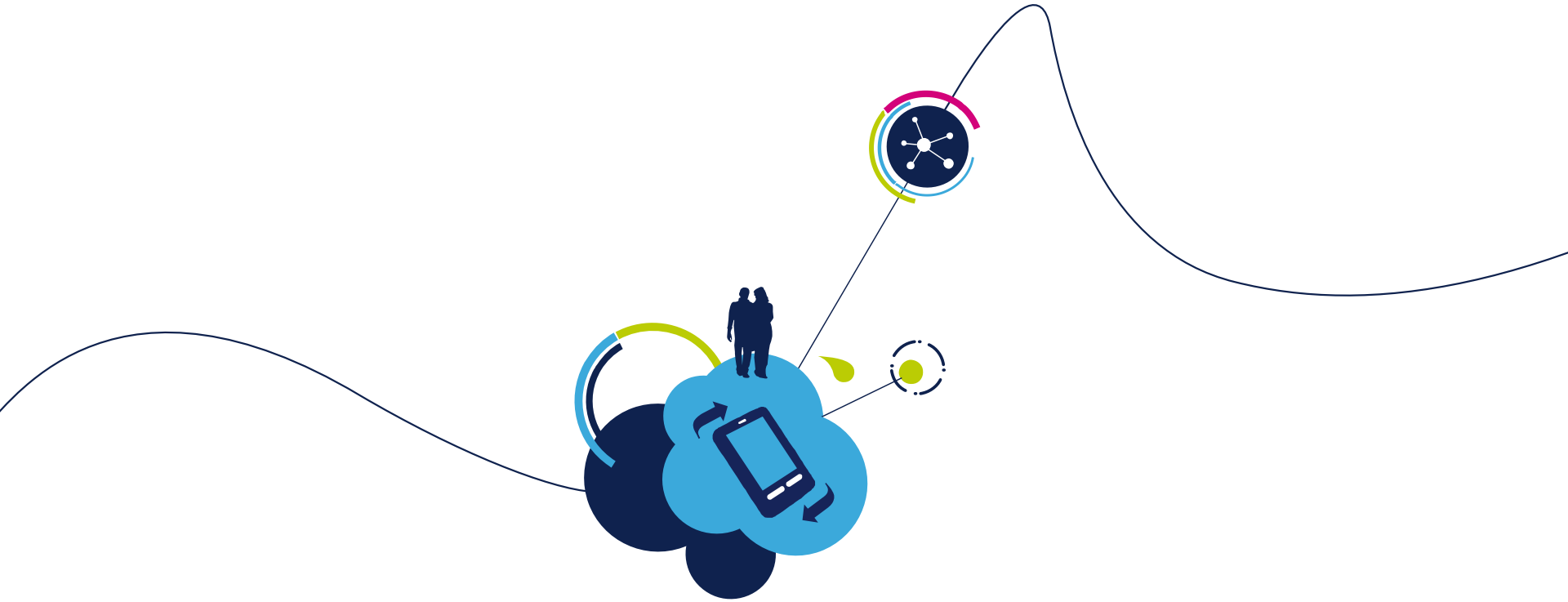
Use pre-compiled binaries for Microsoft IoT Central (NUCLEO-L476RG, B-L475E-IOT01A) (2/2)

- Pre-compiled binaries for NUCLEO-L476RG/B-L475E-IOT01A are provided in folders:
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F476RG-Nucleo/Azure_Sns_DM_BL_IoTCentral.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/B-L475E-IOT01/Azure_Sns_DM_BL_IoTCentral.bin`

- Binaries for IoT Central can be used and configured by dragging the file to the connected board, setting Wi-FI SSID and Password, and then entering IoT Central connection string in serial terminal or by using NFC

- After connecting with IoT Central, the application will start to transmit sensor data and device properties which can be visualize in IoT Central dashboard





Test FP-CLD-AZURE1 with ST Web dashboard

Use pre-compiled binaries for Azure Web dashboard (1/5)

- A web dashboard based on Microsoft Azure has been created to offer developers a quickstart evaluation of features available in FP-CLD-AZURE1.

- Specific ready to use pre-compiled binaries are provided in the FP-CLD-AZURE1 package for Azure_Sns_DM and Azure_Motor applications to connect your boards with the web dashboard:
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F401RE-Nucleo/Azure_Sns_Web.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F476RG-Nucleo/Azure_Sns_DM_BL_Web.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/STM32F429ZI-Nucleo/Azure_Sns_DM_BL_Web.bin`
  - `Projects/Multi/Applications/Azure_Sns_DM/Binaries/B-L475E-IOT01/Azure_Sns_DM_BL_Web.bin`
  - `Projects/Multi/Applications/Azure_Motor/Binaries/STM32F429ZI-Nucleo/Azure_Motor_Web.bin`
  - `Projects/Multi/Applications/Azure_Motor/Binaries/B-L475E-IOT01/Azure_Sns_Motor_Web.bin`

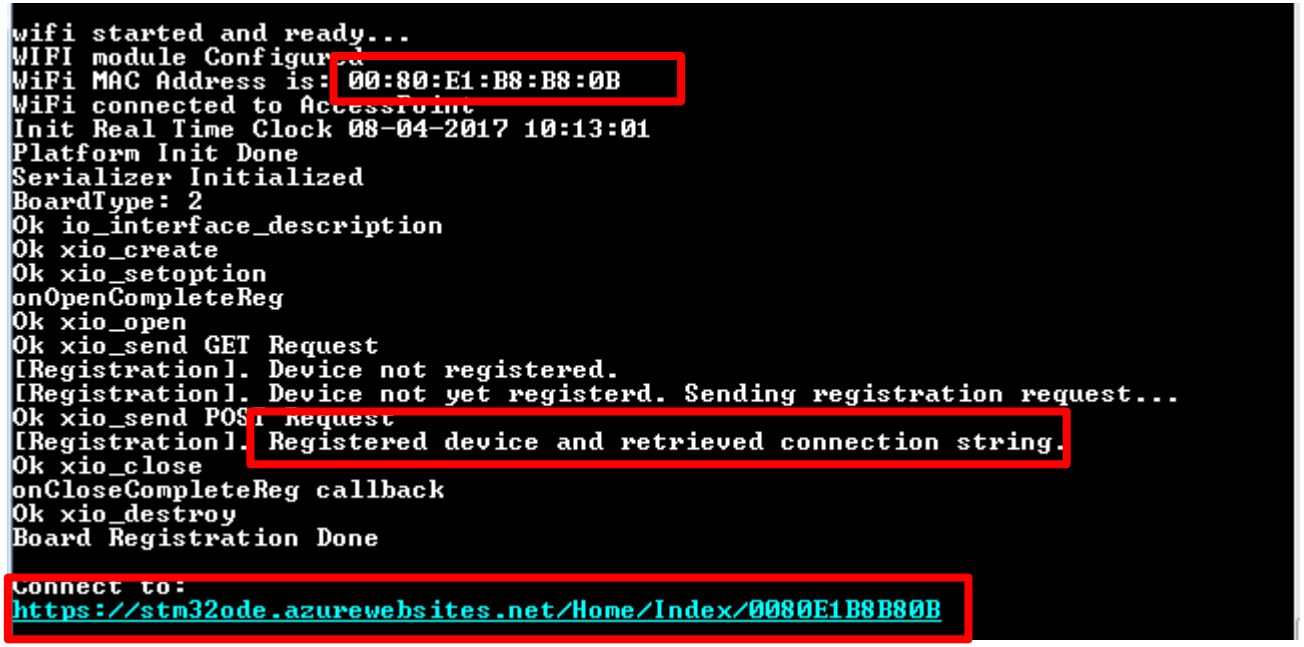
- To start the application, simply connect to the laptop your Nucleo board and drag one of the binaries according to your platform

- The usage of these binaries doesn’t require the creation of an account in Azure since devices can automatically register and retrieve connection string from an ST account in Azure. To learn more on developed Azure solution, check the following link in github:
  - https://github.com/MicrosoftBizSparkItaly/IoTCamp/blob/master/Docs/cloud_architecture_configuration.md

FP-CLD-AZURE1. Step by step setup in details

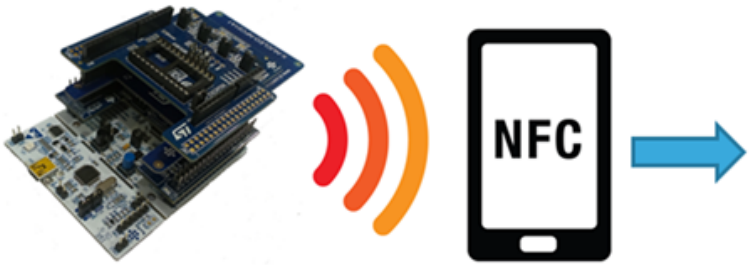
Use pre-compiled binaries for Azure Web dashboard (2/5)

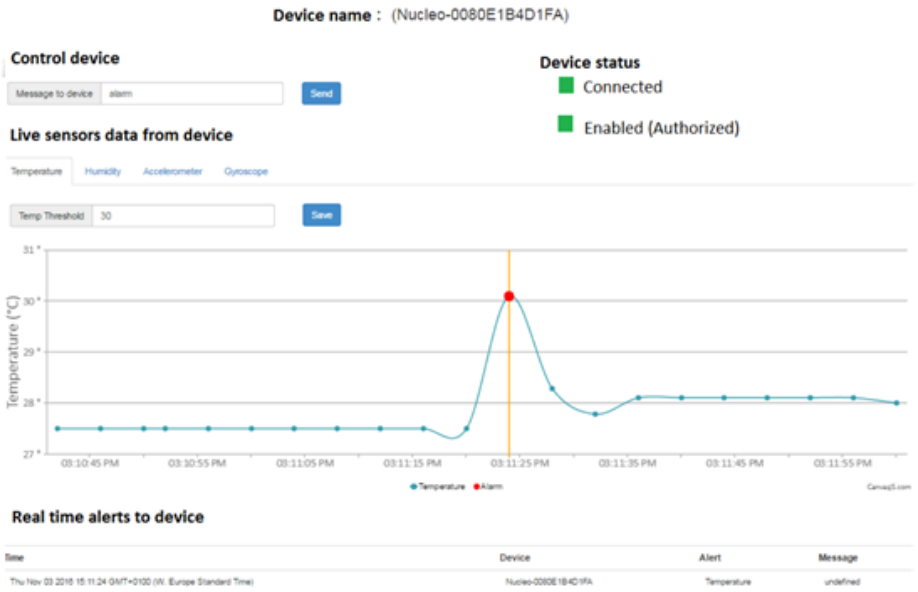
- Open serial terminal to visualize the log of messages. Configure Wi-Fi credentials as described in previous slides (using NFC or in the serial terminal).
- The board register itself to a custom IoTHub using MAC address as device id and automatically retrieves the connection string.
- Web URL is printed over the serial terminal (otherwise go to https://stm32ode.azurewebsites.net and manually insert the MAC address of the board)



Use pre-compiled binaries for Azure Web dashboard (3/5)

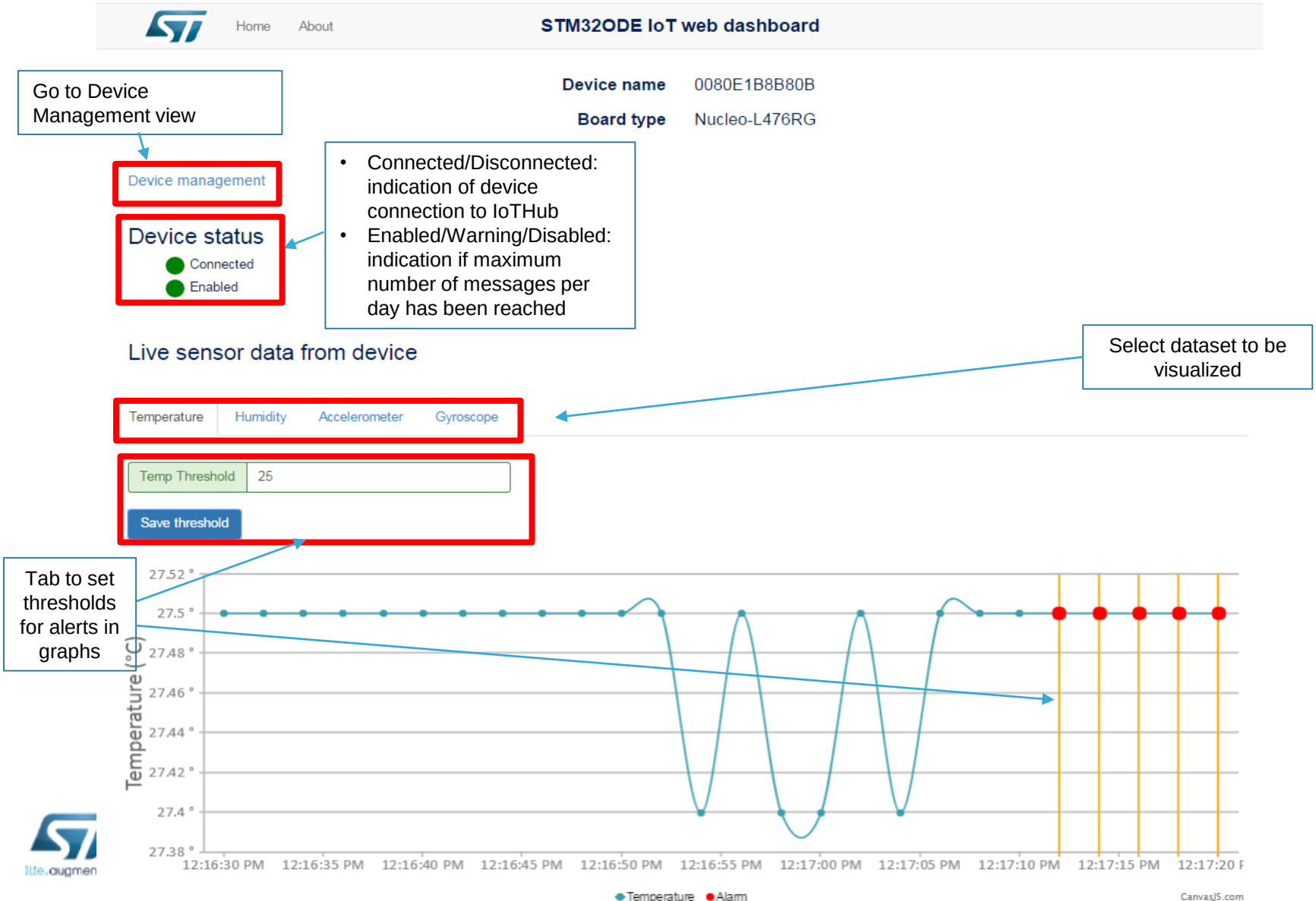
- If you are using an Android mobile, the web page can be automatically opened in your mobile browser by placing the device near to the NFC tag.





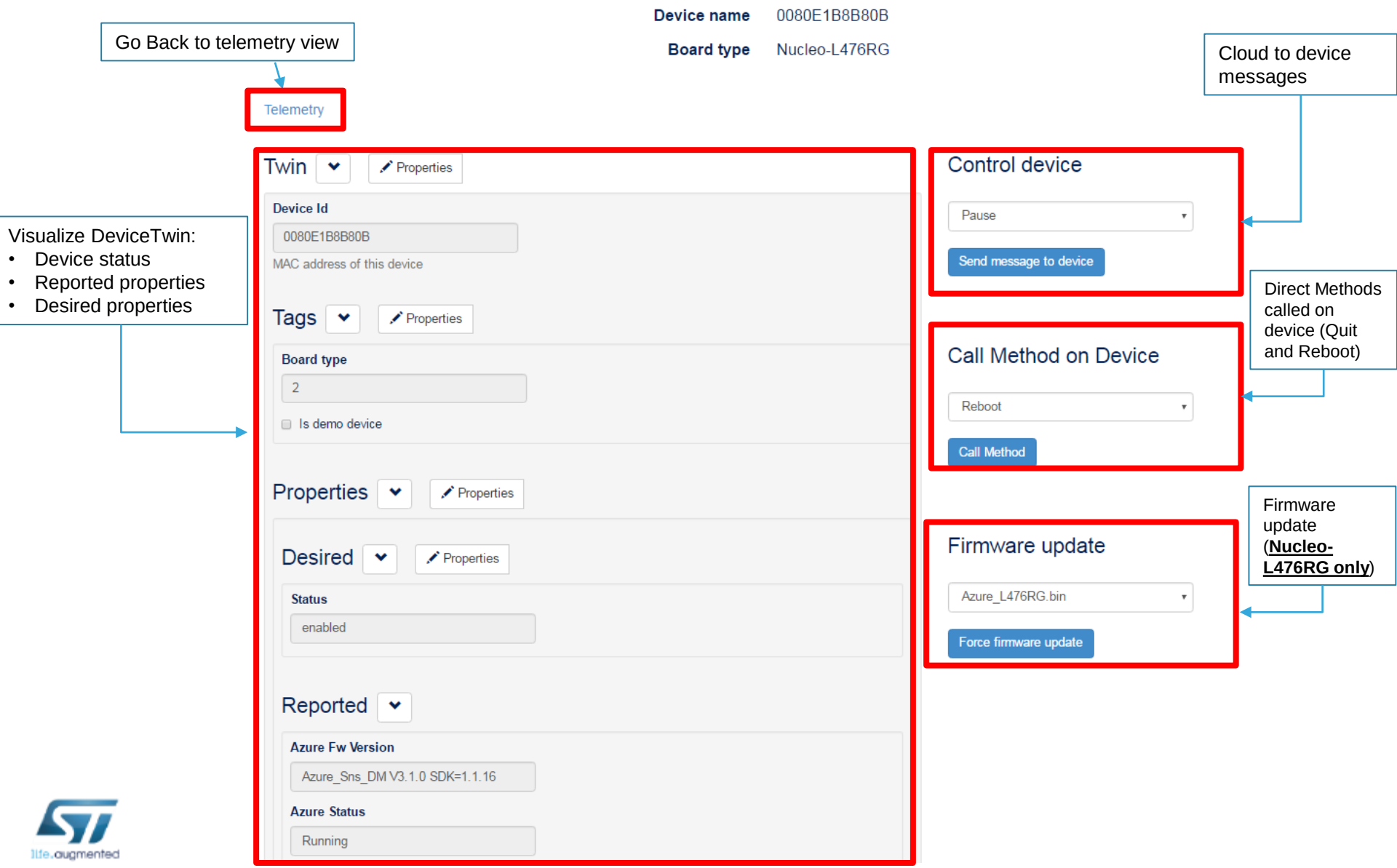
FP-CLD-AZURE1. Step by step setup in details

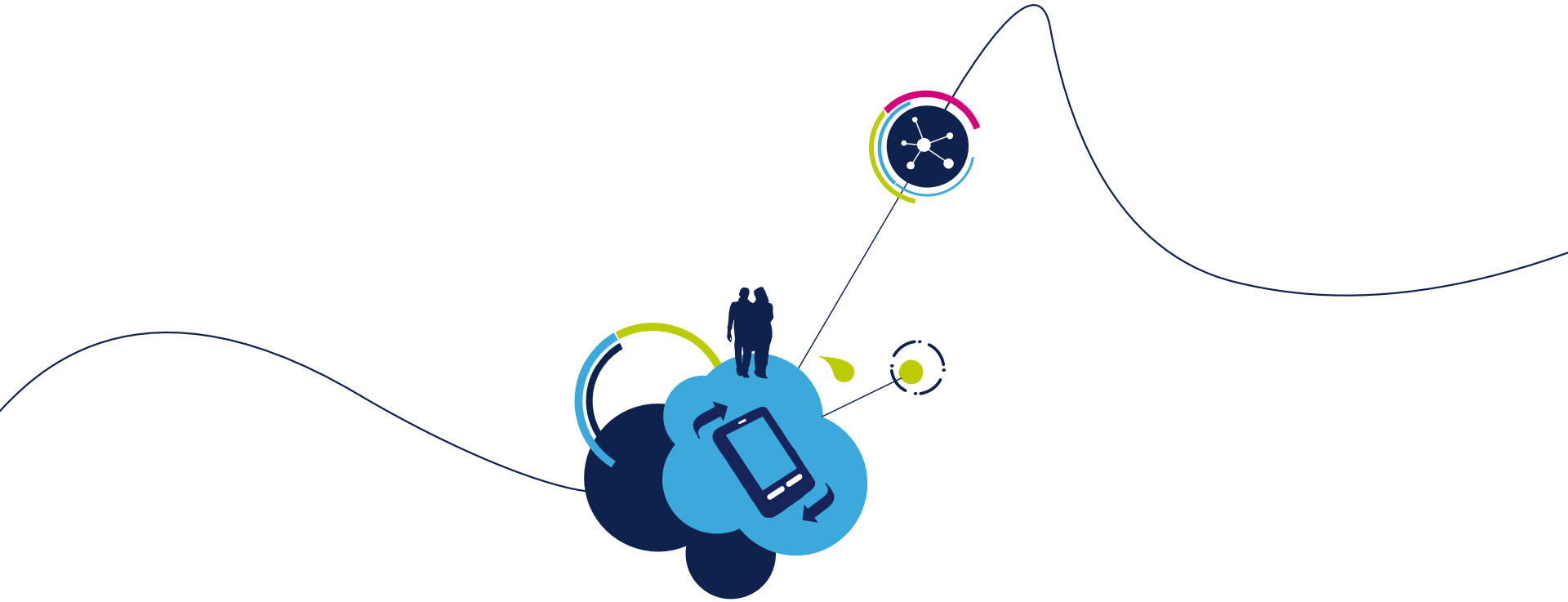
Use pre-compiled binaries for Azure Web dashboard (4/5)



FP-CLD-AZURE1. Step by step setup in details

Use pre-compiled binaries for Azure Web dashboard (5/5)





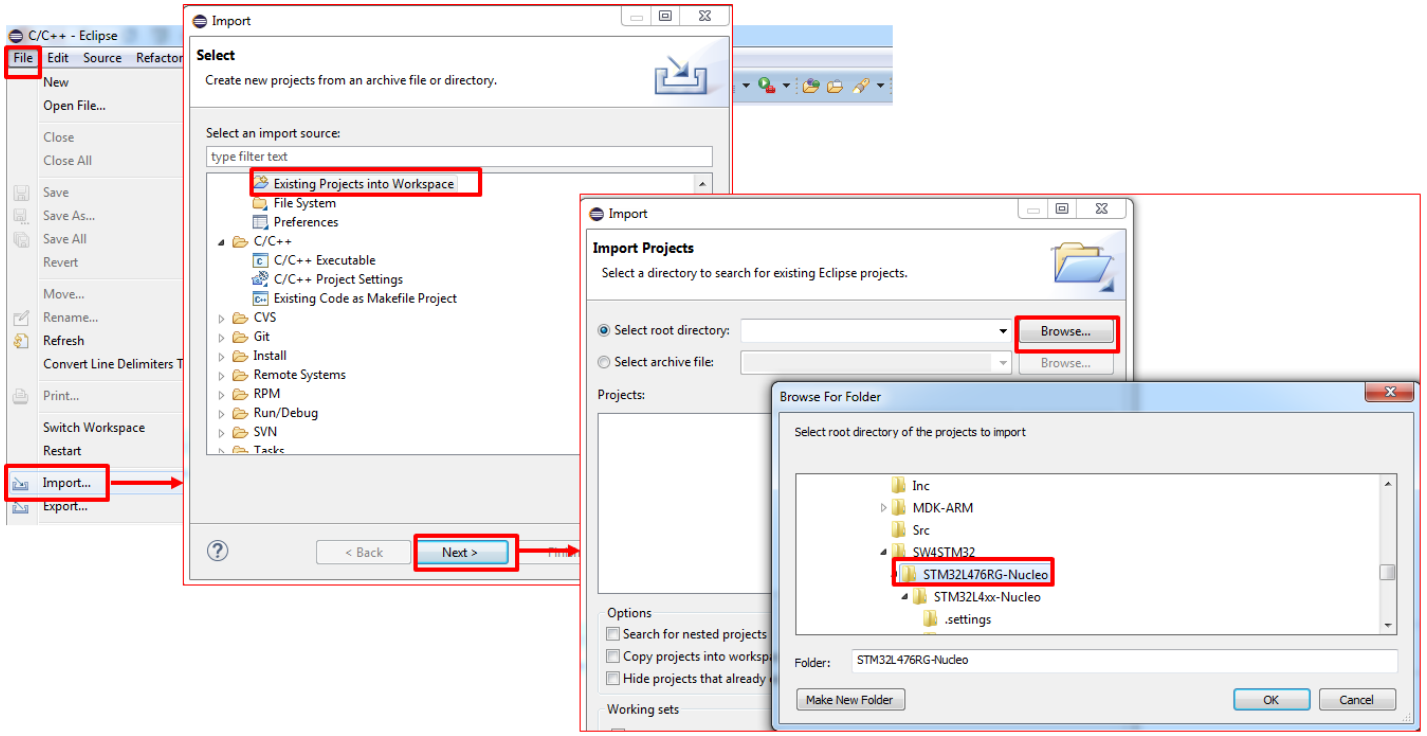
Rebuild solution files in FP-CLD-AZURE1

life.augmented

FP-CLD-AZURE1. Step by step setup in details
Rebuild solution files (1/3)

- Open the available pre-configured projects according to the selected IDE and boards combination. Solution files for each IDE (IAR,ARM-MDK,SystemWorkbench4STM32) can be found in the folder *Projects\Multi\Application\Azure_Sns_DM* or *Projects\Multi\Application\Azure_Motor*

- In SystemWorkbench, click on **File → Import → Existing Project into Workspace**. Then Browse project folder (`SW4STM32/STM32XXXXX-Nucleo`) and click on OK to finish



FP-CLD-AZURE1. Step by step setup in details

Rebuild solution files (2/3)

- To enter in source code **Wi-Fi parameters** (NUCLEO-F01RE/NUCLEO-L476RG + X-NUCLEO-IDW01A1, or B-L475E-IOT01Ax), open the file *azure1_config.h* and add a custom value for *AZURE_DEFAULT_SSID, AZURE_DEFAULT_SECKEY, AZURE_DEFAULT_PRIV_MODE.*

- To enter in source code **Ethernet parameters** (NUCLEO-F429ZI), open the file *platform_STM32Cube_NucleoF429ZI.c* at line 307, and set a custom IP and Gateway configuration (*IP_ADDR0,IP_ADDR1,IP_ADDR2,IP_ADDR3* for IP address, *GW_ADDR0, GW_ADDR1, GW_ADDR2, GW_ADDR3* for Gateway address), or set *EthConfiguration.use_dhcp* flag to 1 to use DHCP.

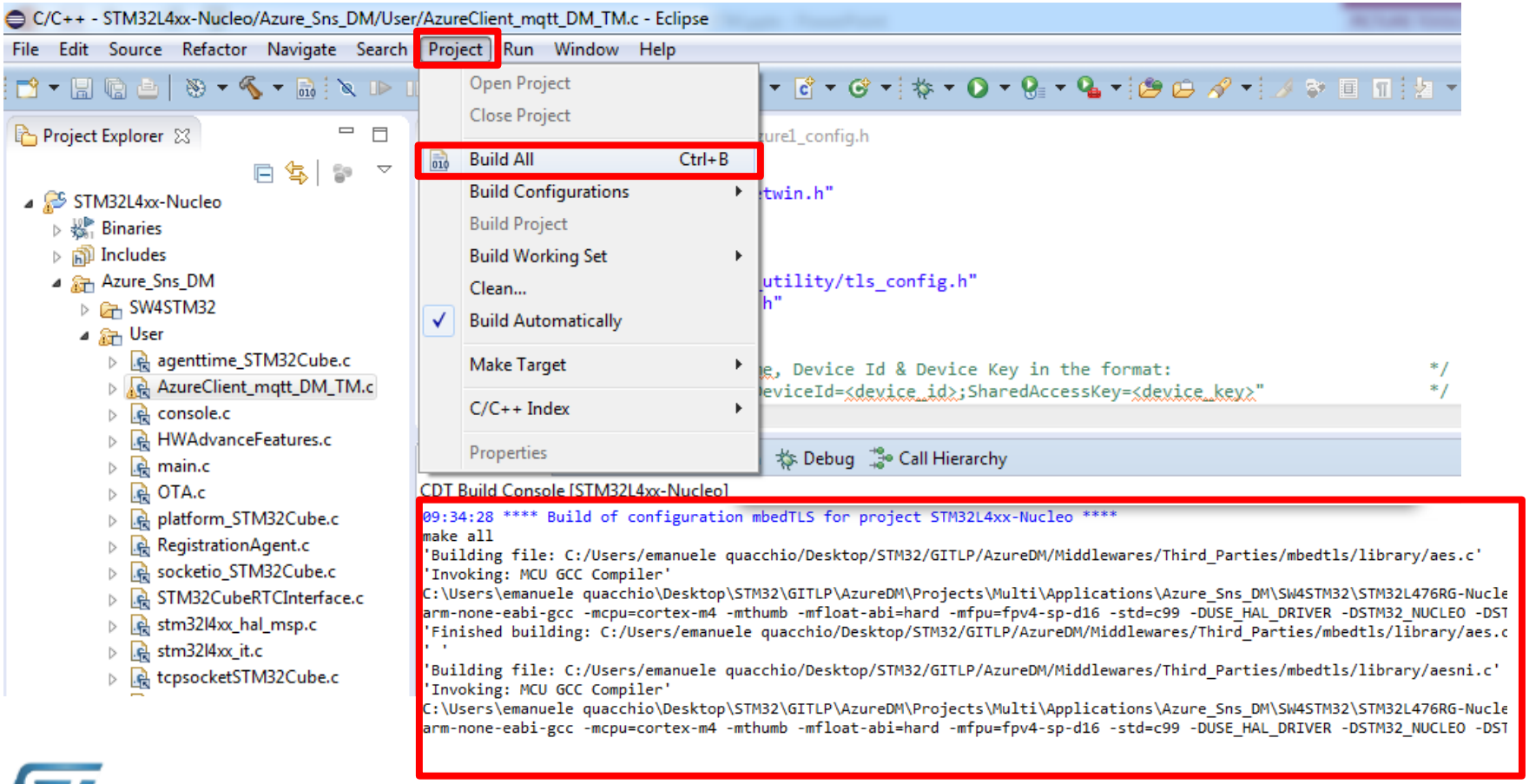
- To enter **IoTHub device connection string** open *azure1_config.h,* uncomment and add a valid connection string for *AZUREDEVICECONNECTIONSTRING, i.e.*

```
#define AZUREDEVICECONNECTIONSTRING "HostName=trial.azurewebsites.net;DeviceId=Nucleo-Trial;SharedAccessKey=XXXXXXX"
```

- To enable communication with IoT Central (for NUCLEO-L476RG and B-L475E-IOT01), open *azure1_config.h* then uncomment define AZURE_IOT_CENTRAL

FP-CLD-AZURE1. Step by step setup in details
Rebuild solution files (3/3)

- Rebuild the solution according to the selected IDE.
- In SystemWorkbench click on **Project → Build All**

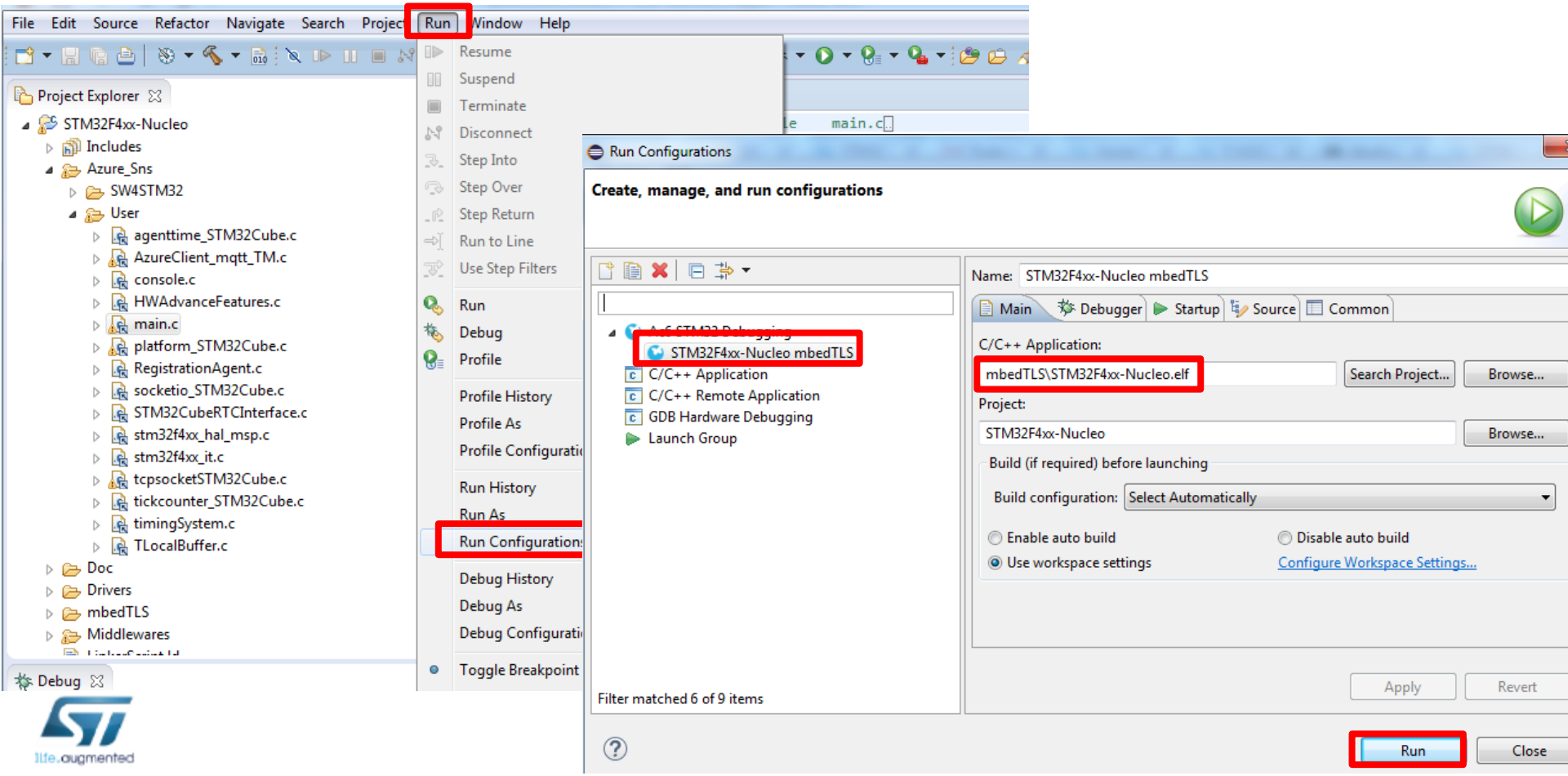


FP-CLD-AZURE1. Step by step setup in details

Flash binary to NUCLEO-F401RE board

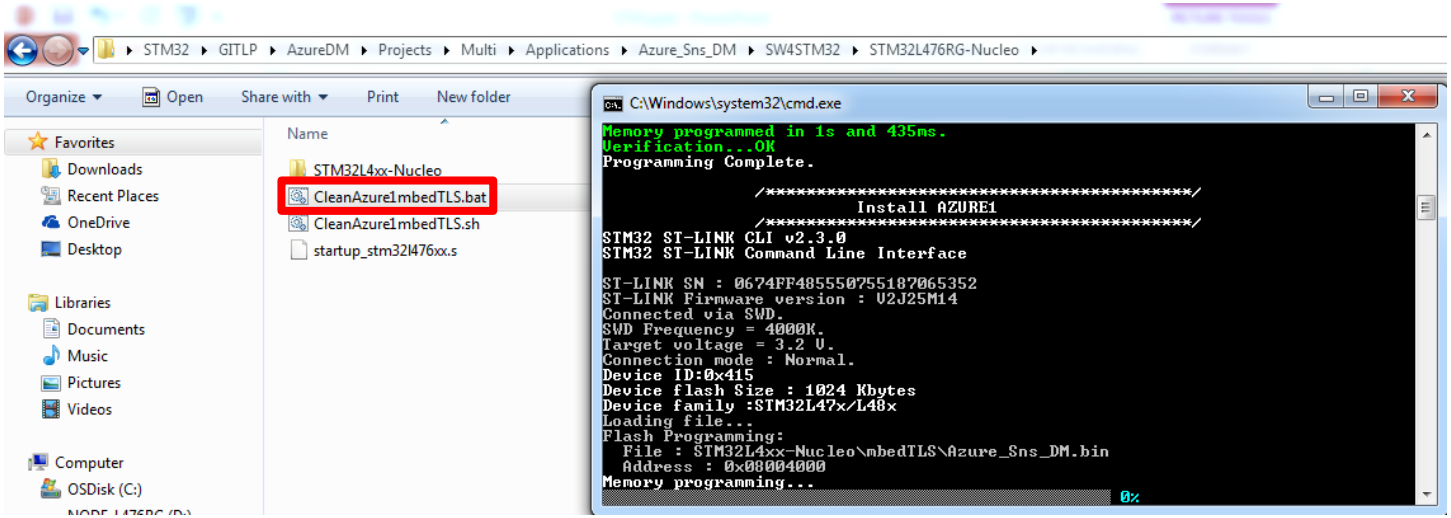
- For the NUCLEO-F401RE platform, which does not support the firmware update-over-the-air procedure and does not require to install a BootLoader, the firmware can be written to the microcontroller from the IDE

- In SystemWorkbench click on **Run→ Run Configuration.** Then select **STM32F4xx-Nucleo mbedTLS** and type in mbedTLS\STM32F4xx-Nucleo.elf; then click on **Run.**



FP-CLD-AZURE1. Step by step setup in details
Flash binary and Bootloader (1/2)

- For NUCLEO-F476RG, NUCLEO-F429ZI, B-L475E-IOT0Ax platforms, once the code has been recompiled, it is necessary to use external scripts to flash the binary together with the bootloader
- Bootloader is necessary to implement Azure IoT device management primitives together with Firmware update application example
- Windows scripts based on ST-LINK command line are available according to the board used and IDE selected (i.e. for SW4STM32):
  - `Projects/Multi/Applications/Azure_Sns_DM/SW4STM32/B-L475E-IOT01/CleanAzure_Sns_DM.bat`
  - `Projects/Multi/Applications/Azure_Sns_DM/SW4STM32/STM32F429ZI-Nucleo/CleanAzure_Sns_DM.bat`
  - `Projects/Multi/Applications/Azure_Sns_DM/SW4STM32/STM32L476RG-Nucleo/CleanAzure_Sns_DM.bat`
- Double click on .bat script to flash bootloader and binary to the device



FP-CLD-AZURE1. Step by step setup in details
Flash binary and Bootloader (2/2)

- For the Linux/OSx operating system there is a similar script, CleanAzure1mbedTLS.sh, that uses OpenOCD command line
- The script is included only in the System Workbench folder and requires to set:
  - the installation path for OpenOCD according to the local configuration;
  - the installation path for STM32 OpenOCD scripts according to the local configuration;
  - the library path for OpenOCD according to the variable used for Linux/OSx environments.

```
#1) Set the Installation path for OpenOCD # example:
#OpenOCD_DIR="C:/Ac6/SystemWorkbench/plugins/fr.ac6.mcu.externaltools.openocd.win32_
1.10.0.201607261143/tools/openocd/" OpenOCD_DIR=""

# 2) Set the installation path for stm32 OpenOCD scripts # example:
#OpenOCD_CFC="C:/Ac6/SystemWorkbench/plugins/fr.ac6.mcu.debug_1.10.0.201607251855/re
sources/openocd/scripts" OpenOCD_CFC=""

# 3) Only for Linux/iOS add openocd library path to _LIBRARY_PATH: # For iOS example:
#export DYLD_LIBRARY_PATH=${DYLD_LIBRARY_PATH}:${OpenOCD_DIR}"lib/" # For Linux example:
#export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${OpenOCD_DIR}"lib/"
```

Quick Start Guide Contents

Documents & Related Resources

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

FP-CLD-AZURE1:

- **DB2891:** STM32 ODE function pack for IoT node with Wi-Fi, NFC and sensors connected to Microsoft Azure cloud– **databrief**
- **UM2043:** Getting started with the FP-CLD-AZURE1 software for IoT node with Wi-Fi, NFC and sensors, connected to Microsoft Azure cloud – **user manual**
- Software setup file

X-NUCLEO-NFC01A1:

- Gerber files, BOM, Schematic
- **DB2353**: Dynamic NFC tag expansion board based on M24SR for STM32 Nucleo – **databrief**
- **AN4624:** Getting started with the STM32 Nucleo and the M24SR expansion board X-NUCLEO-NFC01A1 – **application note**
- **UM1793**: Dynamic NFC tag expansion board based on M24SR for STM32 Nucleo – **user manual**

X-NUCLEO-IDW01M1:

- Gerber files, BOM, Schematic
- **DB2726:** Wi-Fi expansion board based on SPWF01SA module for STM32 Nucleo – **databrief**
- **UM1973**: Getting started with the X-CUBE-WIFI1 Wi-Fi functions and applications software expansion for STM32Cube – **user manual**
- **UM1765**: Getting started with X-NUCLEO-IDW01M1 Wi-Fi expansion board based on SPWF01SA module for STM32 Nucleo – **user manual**

X-NUCLEO-IKS01A1:

- Gerber files, BOM, Schematic
- **DB10619**:Motion MEMS and environmental sensor expansion board for STM32 Nucleo – **product specification**
- **UM1820**: Getting started with motion MEMS and environmental sensor expansion board for STM32 Nucleo – **user manual**

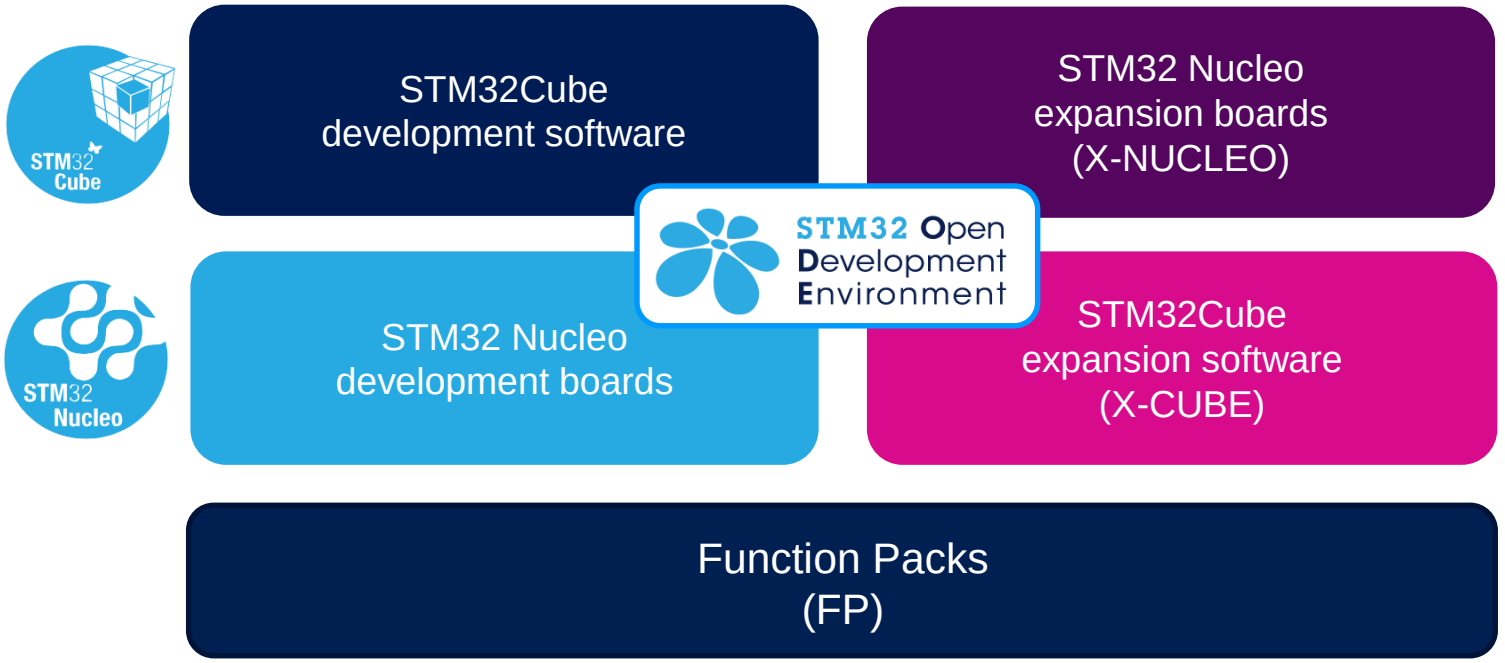
X-NUCLEO-IKS01A2:

- Gerber files, BOM, Schematic
- **DB3009**:Motion MEMS and environmental sensor expansion board for STM32 Nucleo – **product specification**
- **UM2121**: Getting started with motion MEMS and environmental sensor expansion board for STM32 Nucleo – **user manual**

Consult www.st.com for the complete list

STM32 Open Development Environment
Fast, affordable Prototyping and Development

- The STM32 Open Development Environment (ODE) consists of a set of stackable boards and a modular open SW environment designed around the STM32 microcontroller family.

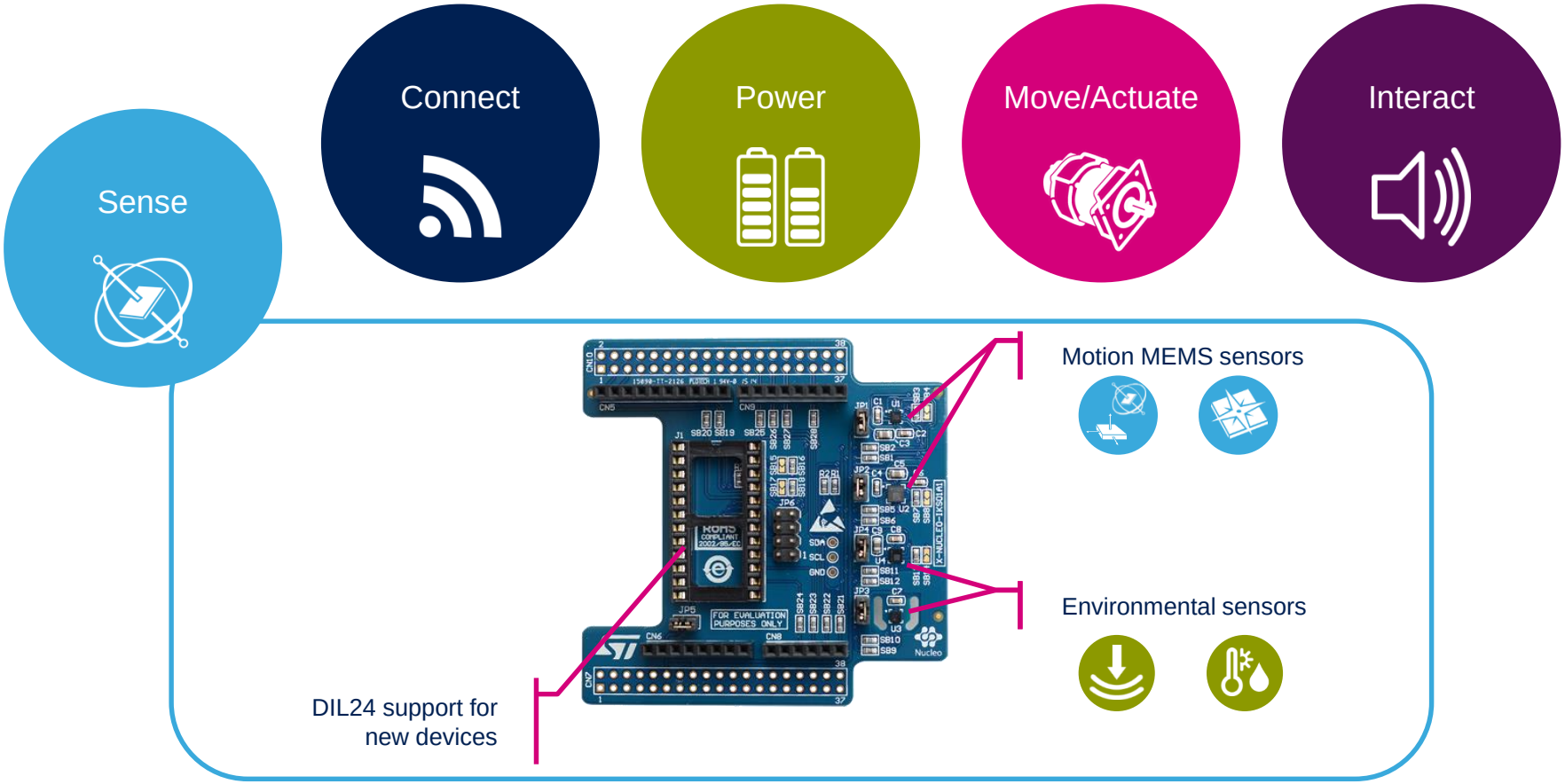


www.st.com/stm32ode

life.augmented

Expansion Boards (X-NUCLEO)

- Boards with additional functionality that can be plugged directly on top of the STM32 Nucleo development board directly or stacked on another expansion board.



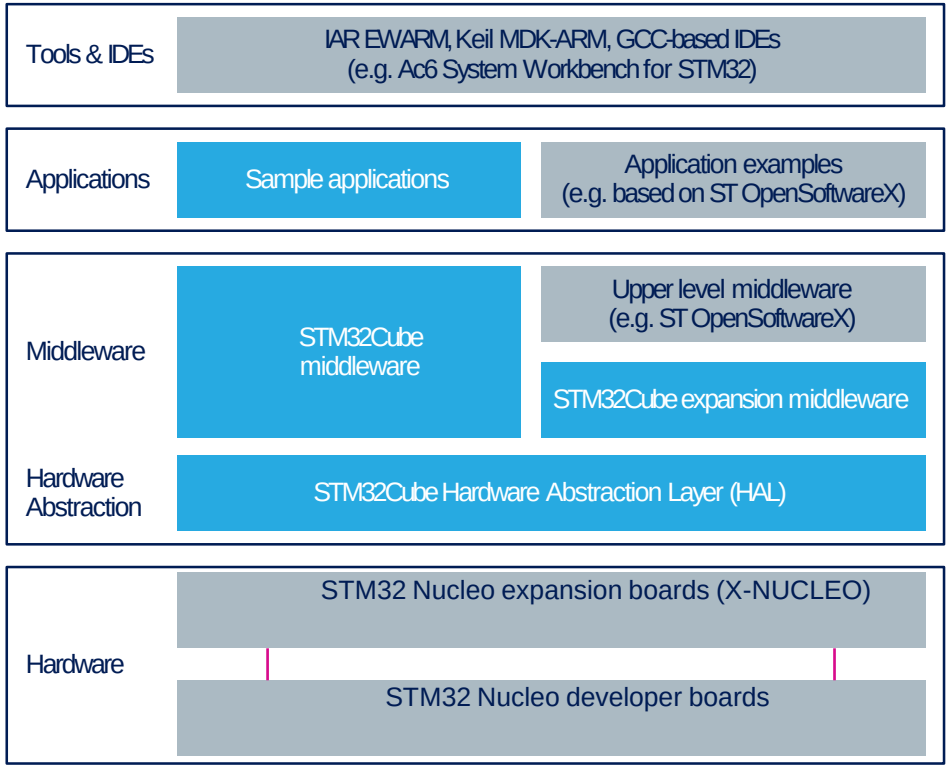
Example of STM32 expansion board (X-NUCLEO-IKS01A1)

STM32 Open Development Environment

Software components

- **STM32Cube software (CUBE)** - A set of free tools and embedded software bricks to enable fast and easy development on the STM32, including a Hardware Abstraction Layer and middleware bricks.

- **STM32Cube expansion software (X-CUBE)** - Expansion software provided free for use with the STM32 Nucleo expansion board and fully compatible with the STM32Cube software framework. It provides abstracted access to expansion board functionality through high-level APIs and sample applications.



- **Compatibility with multiple Development Environments -** The STM32 Open Development Environment is compatible with a number of IDEs including IAR EWARM, Keil MDK, and GCC-based environments. Users can choose from three IDEs from leading vendors, which are free of charge and deployed in close cooperation with ST. These include Eclipse-based IDEs such as Ac6 System Workbench for STM32 and the MDK-ARM environment.

life.augmented

OPEN LICENSE MODELS: STM32Cube software and sample applications are covered by a mix of fully open source BSD license and ST licenses with very permissive terms.

www.st.com/stm32cube

www.st.com/x-cube