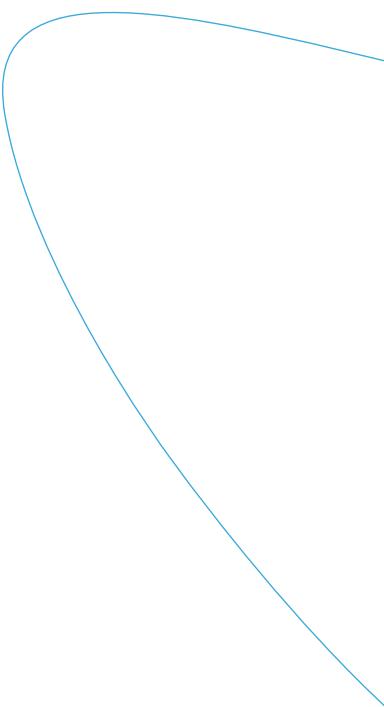




Simplifying Microsoft Azure Connectivity with the STM32L4 Discovery Kit IoT Node



v1.5



Agenda

2

Presentation

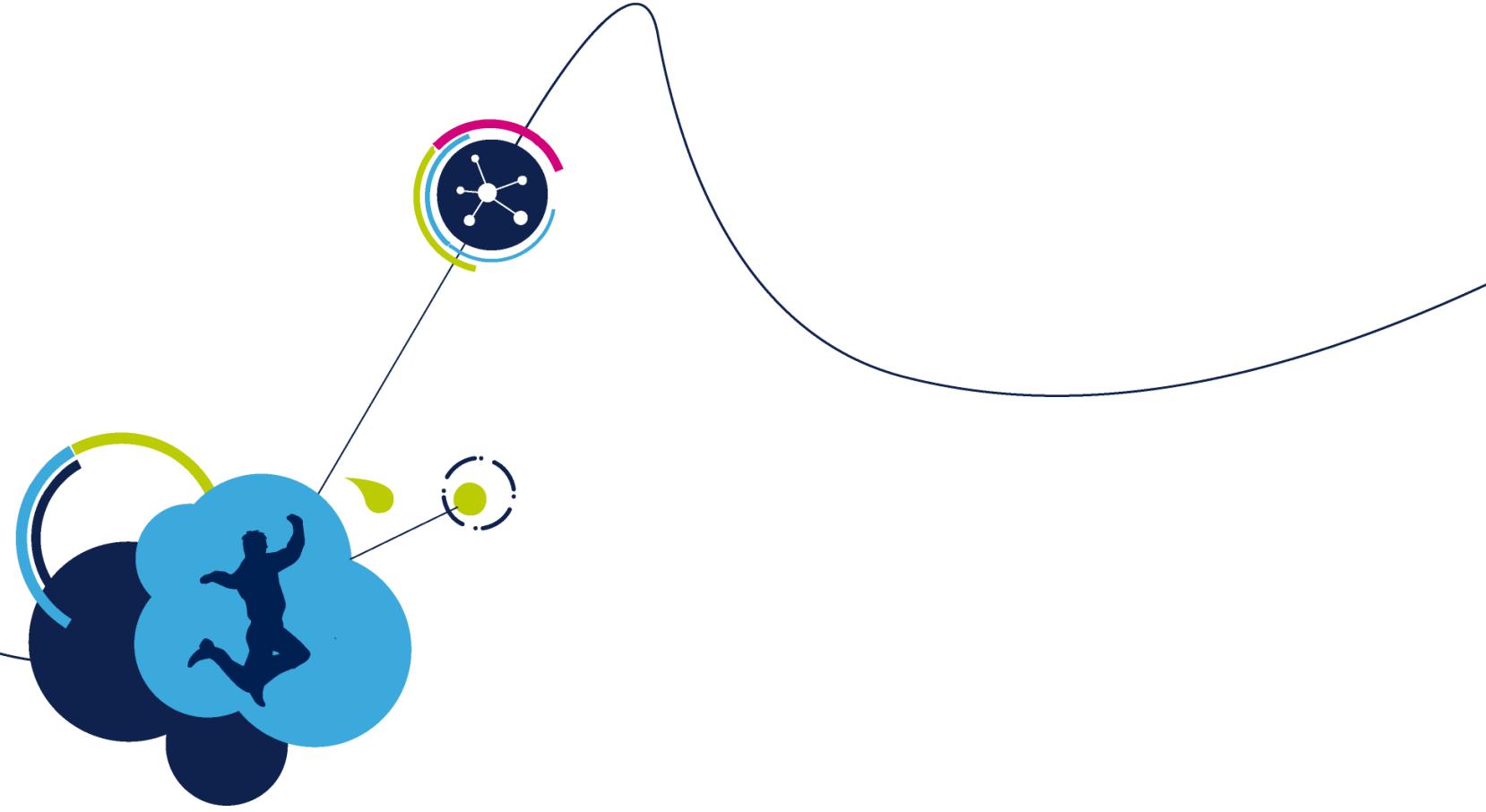
- Training Material Check/Installation Help
- Overview of the STM32 Portfolio
- Keil MDK-ARM Installation
- Overview of the STM32L475
- Overview of the STM32L475 Discovery Kit IoT Node
- STM32Cube™ Introduction
- Keil MDK-ARM License Installation
- Lab 1: Getting Started with STM32CubeMX - Blinking LED
- Microsoft Azure IoT Overview
- Lab 2: Getting Started with Microsoft Azure IoT in 5 minutes!
- FP-CLD-AZURE1 Function Pack Overview
- Lab 3: Connect STM32 IoT Node to Azure IoT Hub
- LUNCH

Agenda

3

Presentation

- Lab 4: Monitor IoT Node Sensor Data sent to Azure Cloud
- Lab 5: Manage IoT Node from the Azure Portal
- Lab 6: Update IoT Node Firmware Over The Air (FOTA)
- Bluetooth® Low Energy Overview
- Demo: Simple BLE phone pairing



Training Material Check / Installation Help

Seminar Day Check List

5

- You would have received an email about a week ago with the Dropbox download link and instructions to install the necessary software.
- Please use the printed checklist handed to you today to make sure that you have completed the necessary steps.

Training Material Installation

- In case you have missed something, we have a few USB Flash drives with the Seminar Installer that we will pass around. This will install Tera Term, Device Explorer, NotePad++, and extract the seminar file to

C:\STM32L4AzureSeminar

- Please insert the USB Drive to your machine. Copy all the files to your desktop and execute the installer (Run as Admin)

STM32L4AzureSeminar_Installer.exe

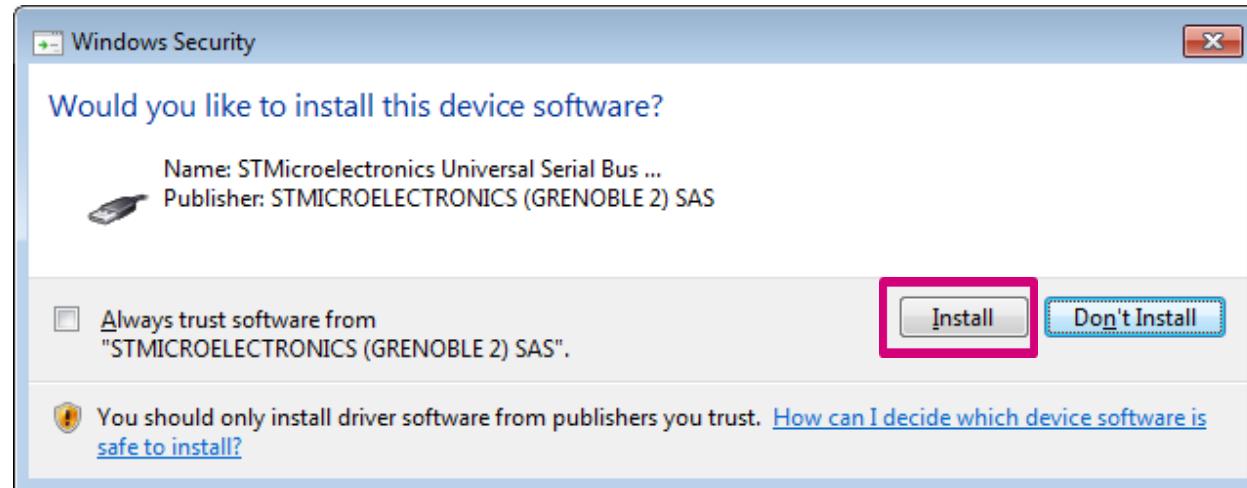
- The USB flash drive also contains this presentation in pdf format

STM32L4_DK_IoT_Azure_Seminar.pdf

Training Material Installation

(Cont'd)

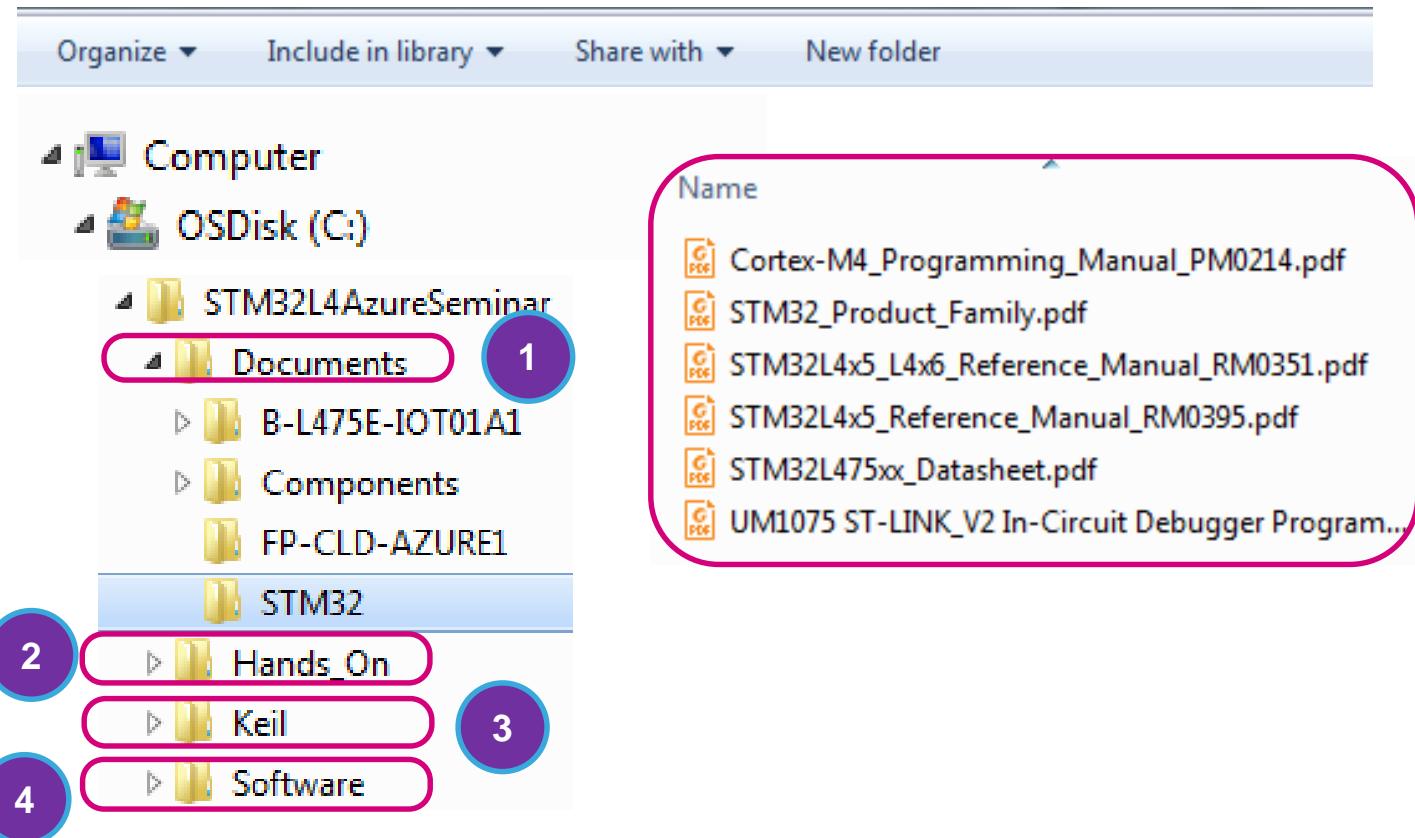
- If you see prompts such as the one below, please click “Install” to allow installing the ST-Link Device Drivers (there would be typically two such prompts).

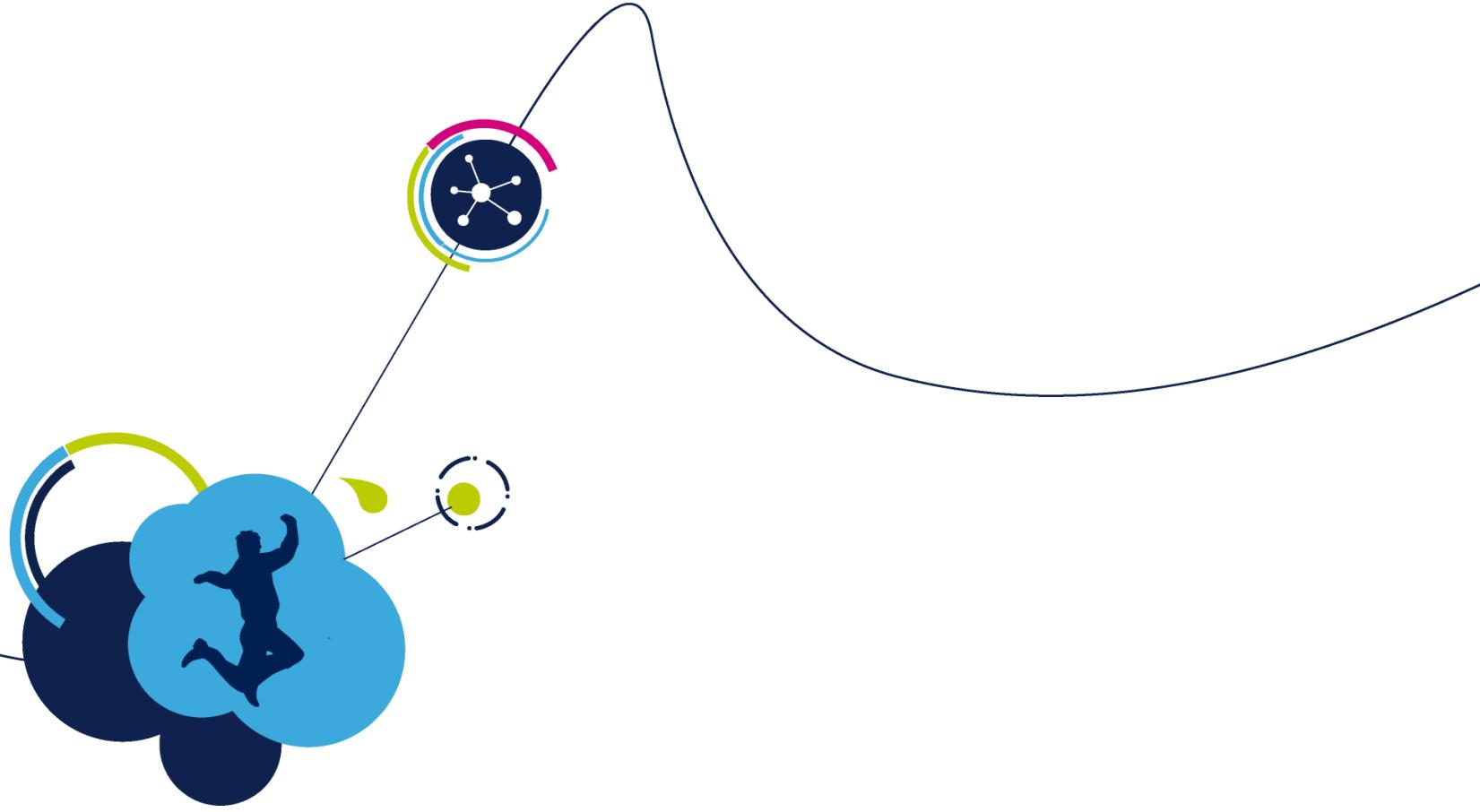


Workshop Directory Content

1. Documents
2. Hands on
3. Keil
4. Software

Name
Azure_Device_Explorer.msi
Notepad++-7.5.1.zip
STM32CubeL4-1.9.0.zip
STM32CubeMX-4.22.1.zip
STSW-LINK004_STM32_ST-LINK.Utility-4....
STSW-LINK009_USB_Drivers-2.0.0.zip
TeraTerm-4.96.exe

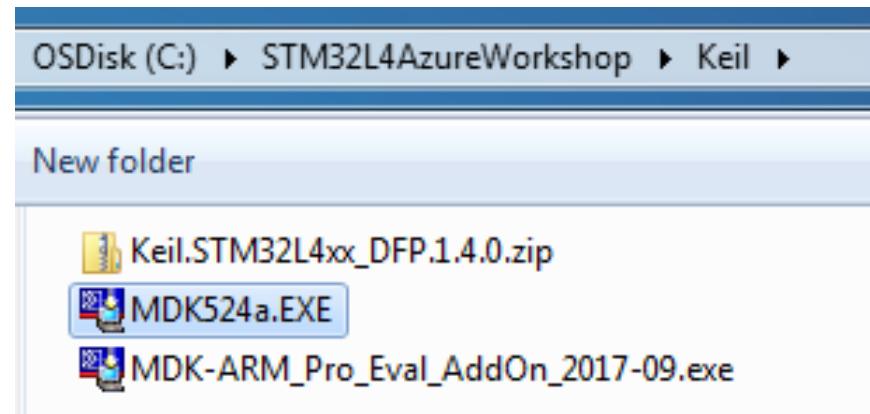




Keil MDK-ARM Tool Installation

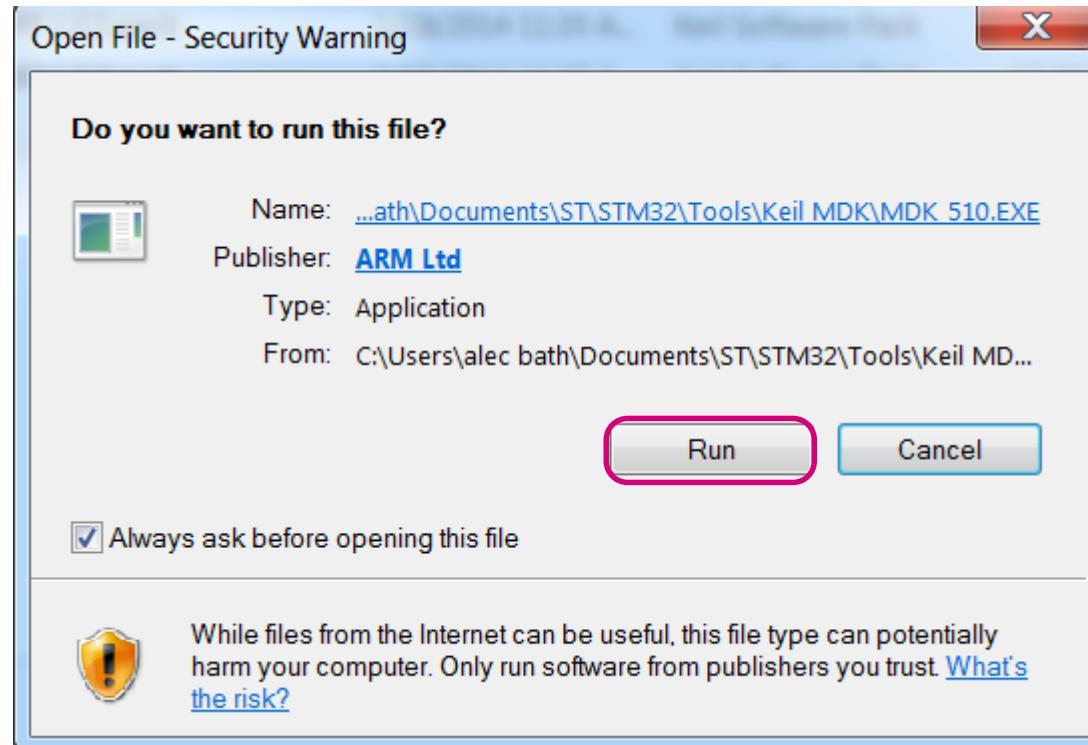
Step 1. Keil Installation

- Browse to C:\STM32L4AzureSeminar\Keil\ and Double-click on the file **MDK524a.exe** to begin the Keil toolchain installation.
 - Click through the default options and accept the 'license agreement' when prompted.
 - If you have an existing installation of Keil you should install in a new directory



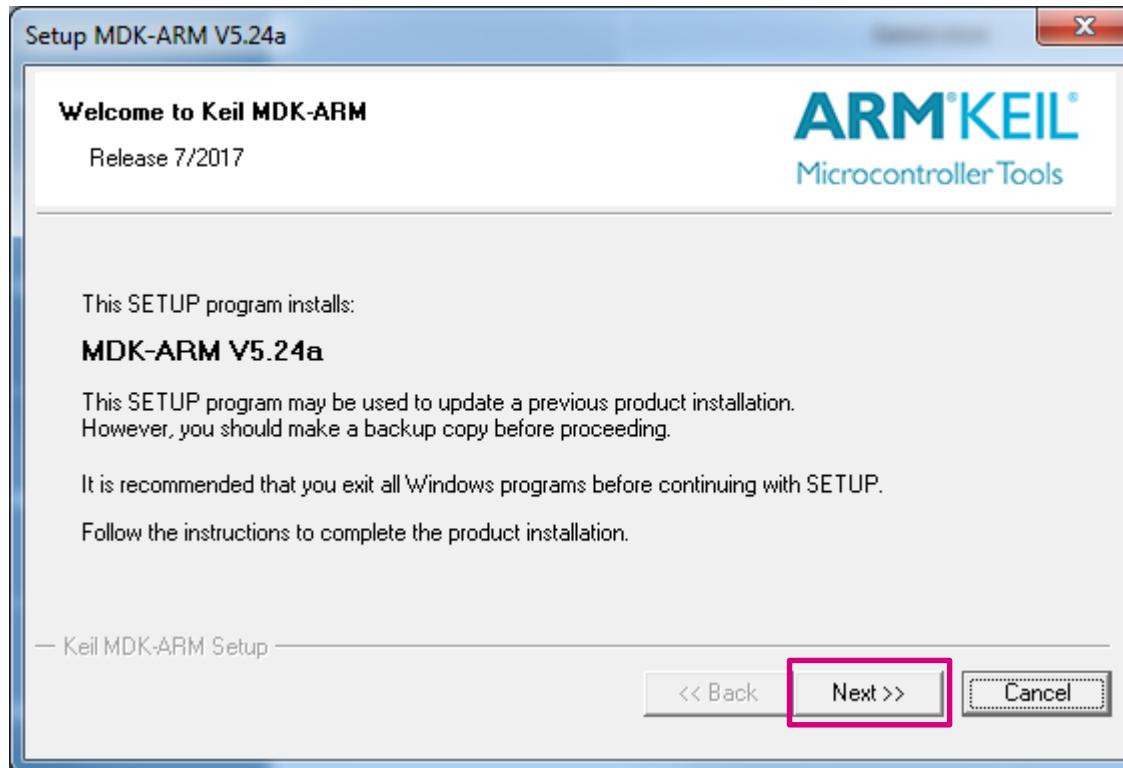
Step 1. Keil Installation

- Click on Run to begin the installation



Step 1. Keil Installation

- Click on Next



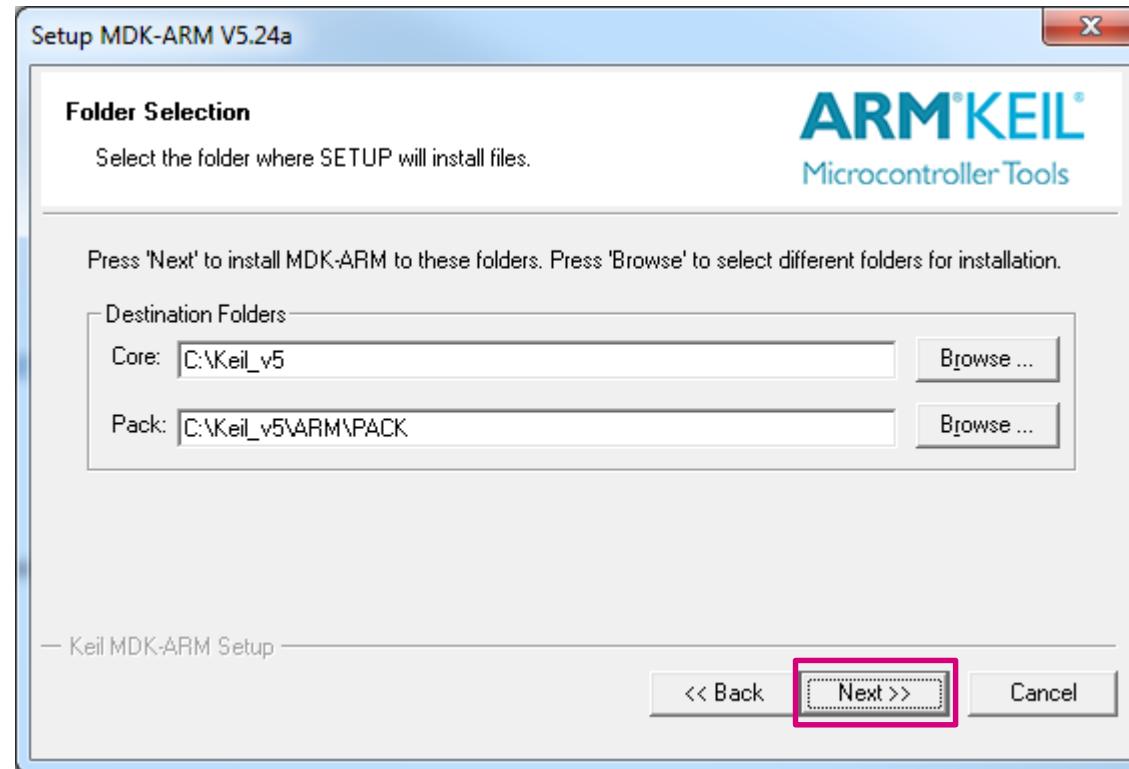
Step 1. Keil Installation

- Accept the license agreement and click on **Next**



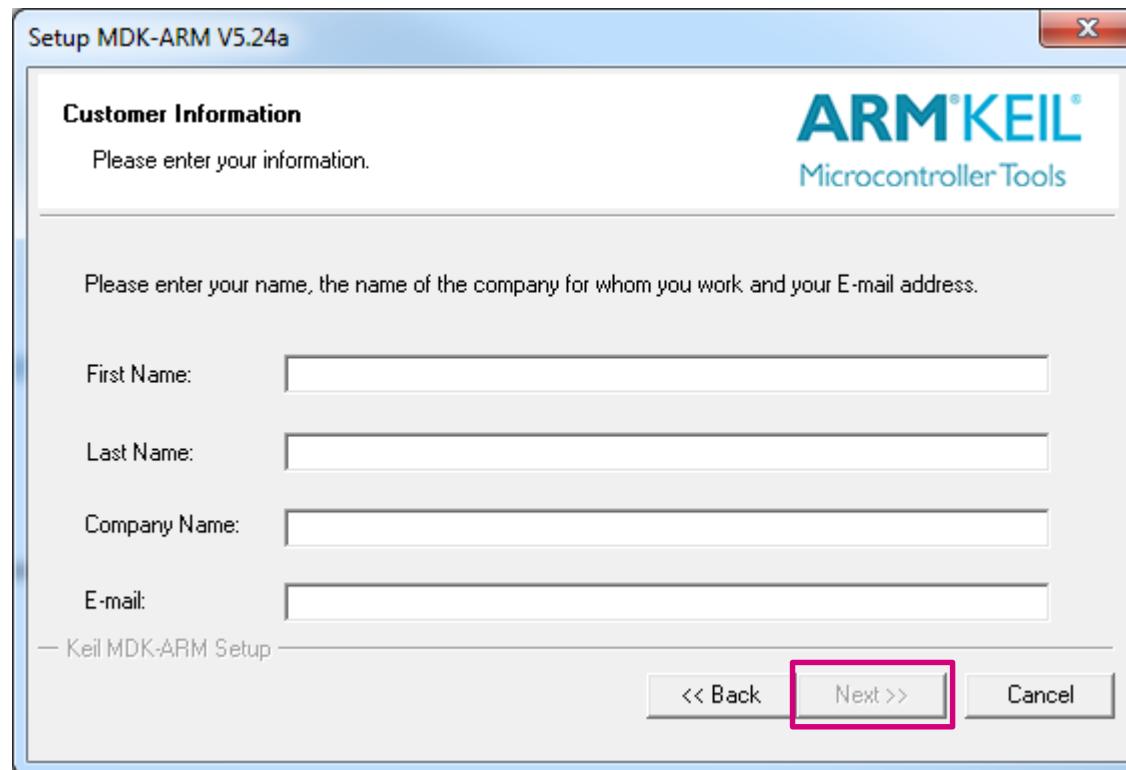
Step 1. Keil Installation

- Please use the default directory path: C:\Keil_v5 and click on Next



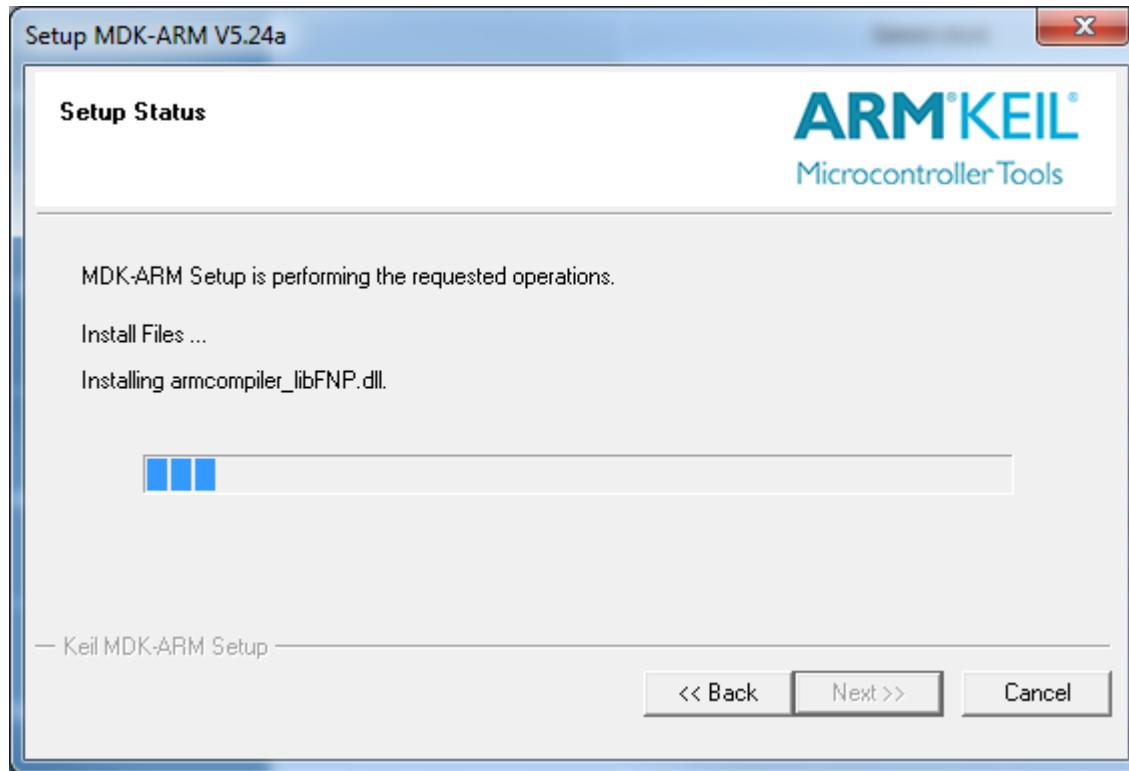
Step 1. Keil Installation

- Enter your contact information and click on **Next**



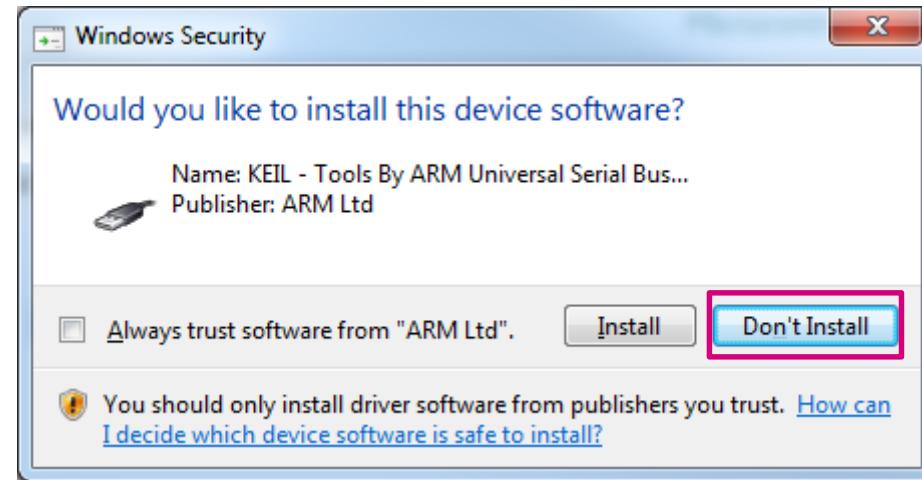
Step 1. Keil Installation

- Installation begins...



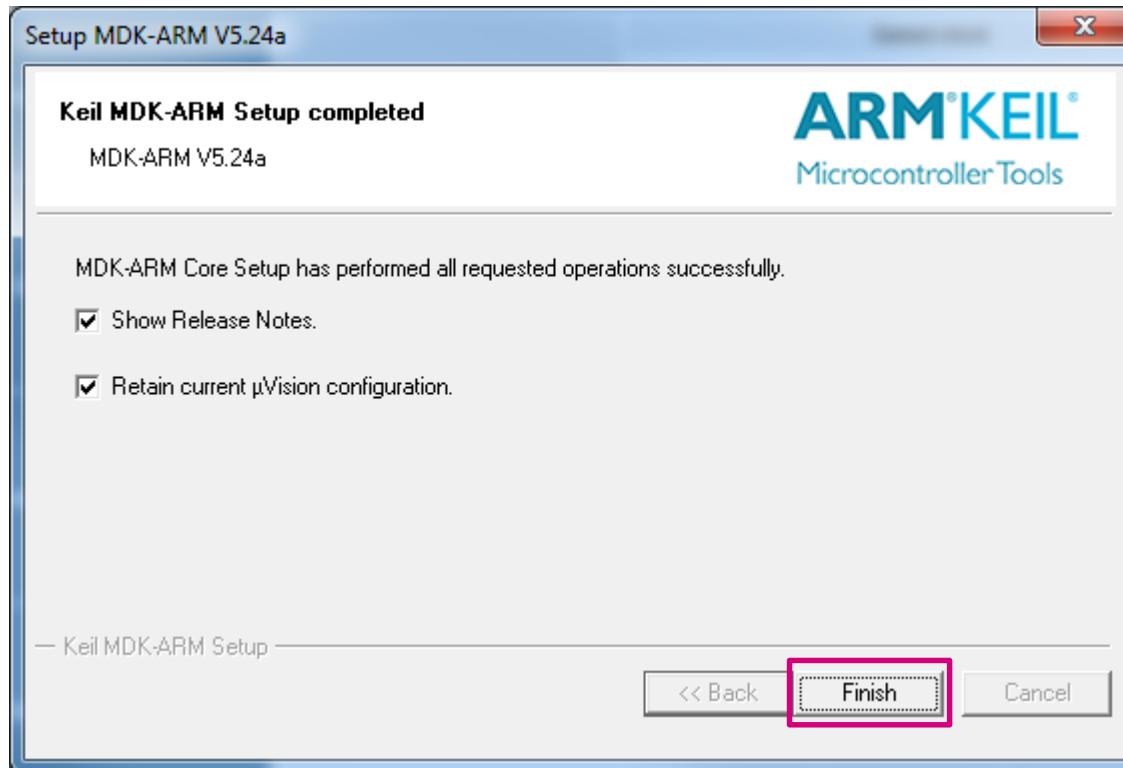
Step 1. Keil Installation

- Before installing **ULINK** device drivers, Windows will block it and asks for confirmation. We will not use it so just click **Don't Install**



Step 1. Keil Installation

- Setup complete.... Click **Finish**
- We will manually install “Software Packs” a bit later



If you get any errors for the “Pack Installer”, don’t worry.
Just click OK, and we will get to that in a bit.

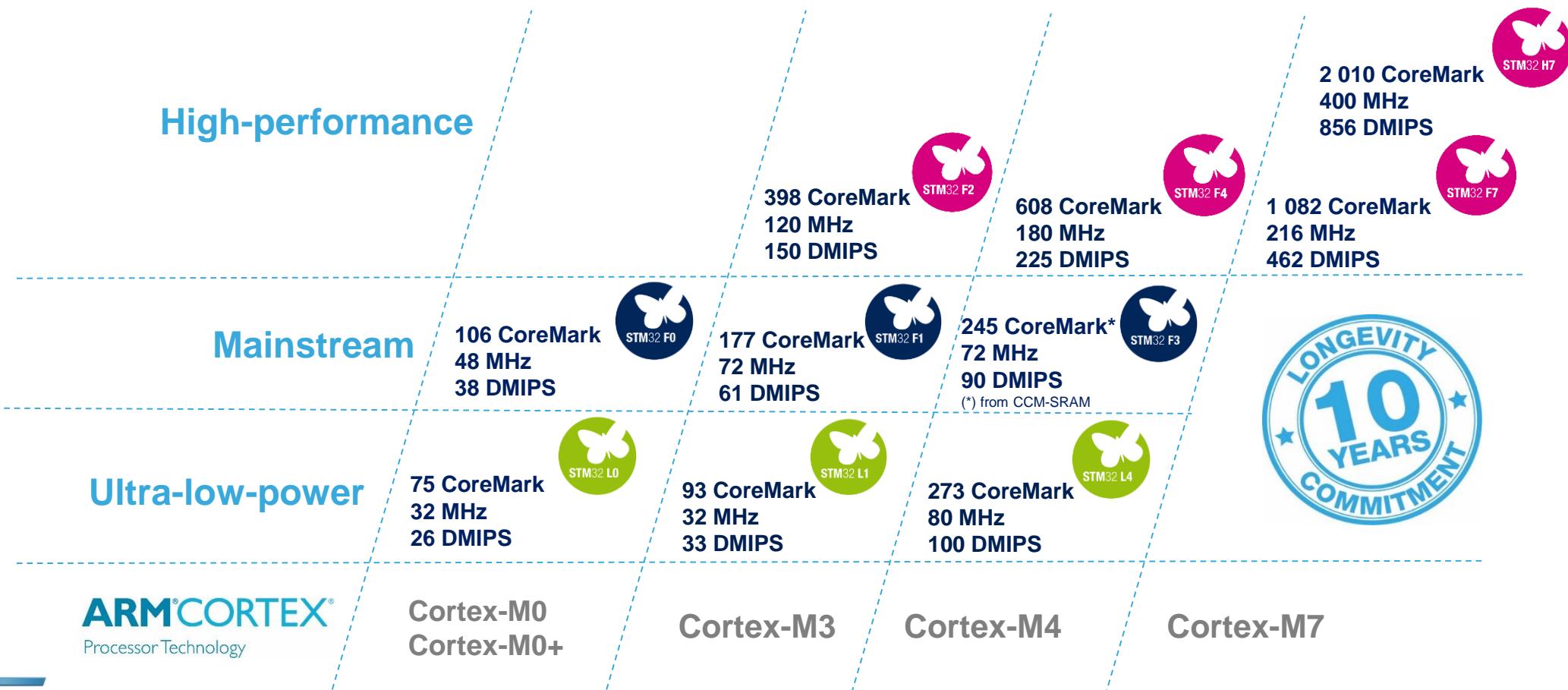


Overview of the STM32 Portfolio

Today - STM32 Portfolio

20

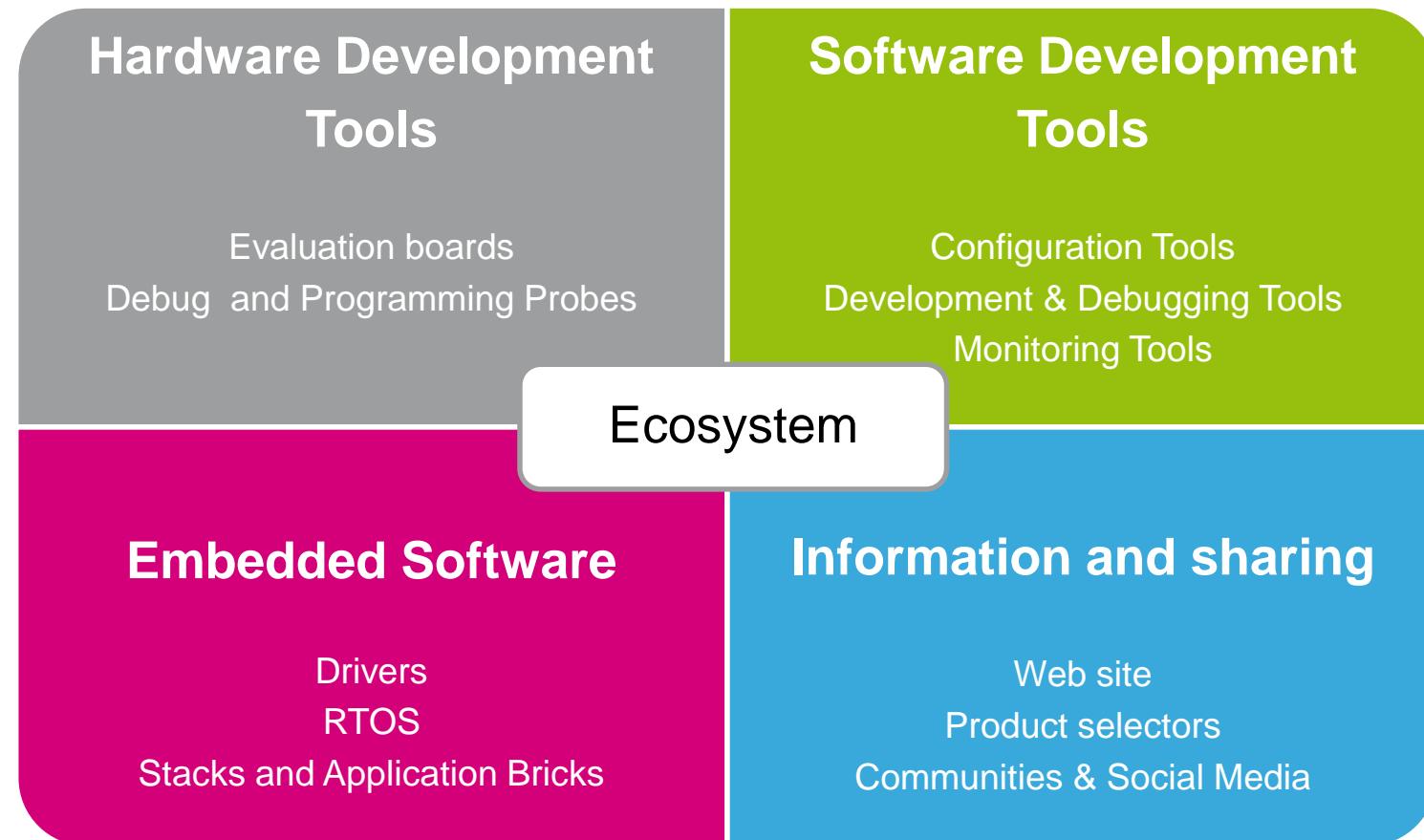
10 product series / More than 40 product lines



What is the MCU Ecosystem?

21

All collaterals required to develop with an MCU



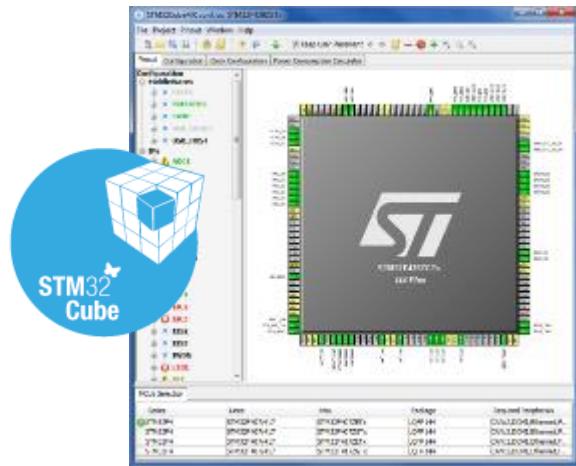
Hardware Development Tools

	 STM32 Nucleo			
Typical use case	Flexible prototyping, Community	Prototyping, Creative demos	Full feature evaluation	3 rd parties
Extension possibilities	+++	++	+	From full evaluation to open hardware
Connectivity	Arduino™ ST Morpho	ST	ST	

STM32 Ecosystem SW Development Tools

23

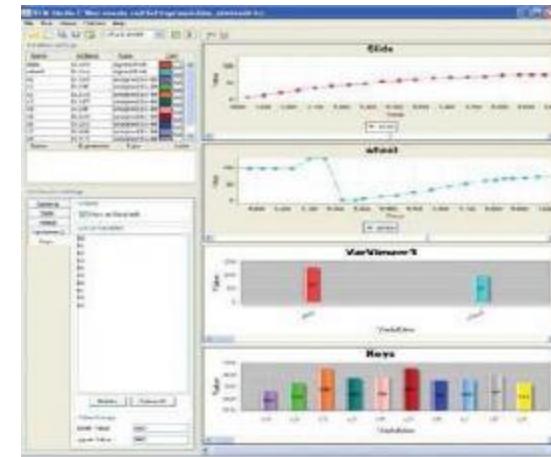
A complete flow, from configuration up to monitoring



STM32CubeMX
Configure & Generate Code

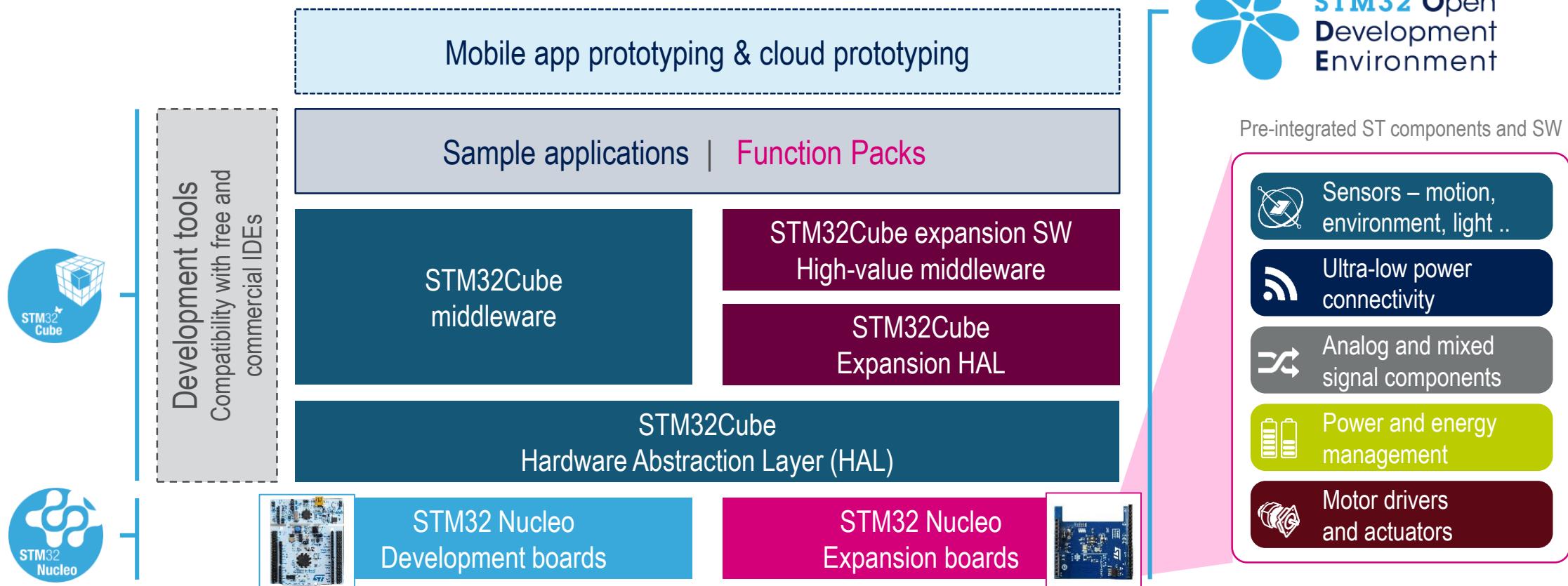


Partners IDEs
Compile and Debug



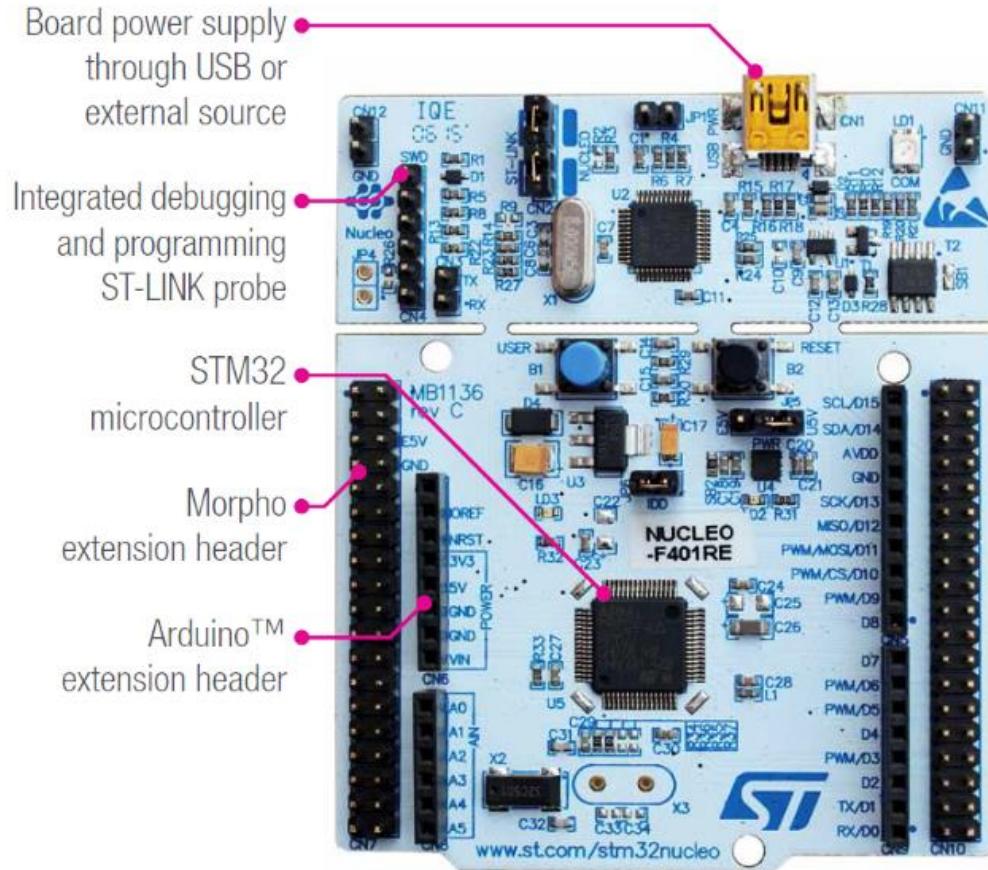
STMStudio
Monitor

Development Software Architecture



STM32 Nucleo Development Boards

A comprehensive range of affordable development boards for all STM32 microcontroller series, with unlimited unified expansion capability, and with integrated debugger/programmer



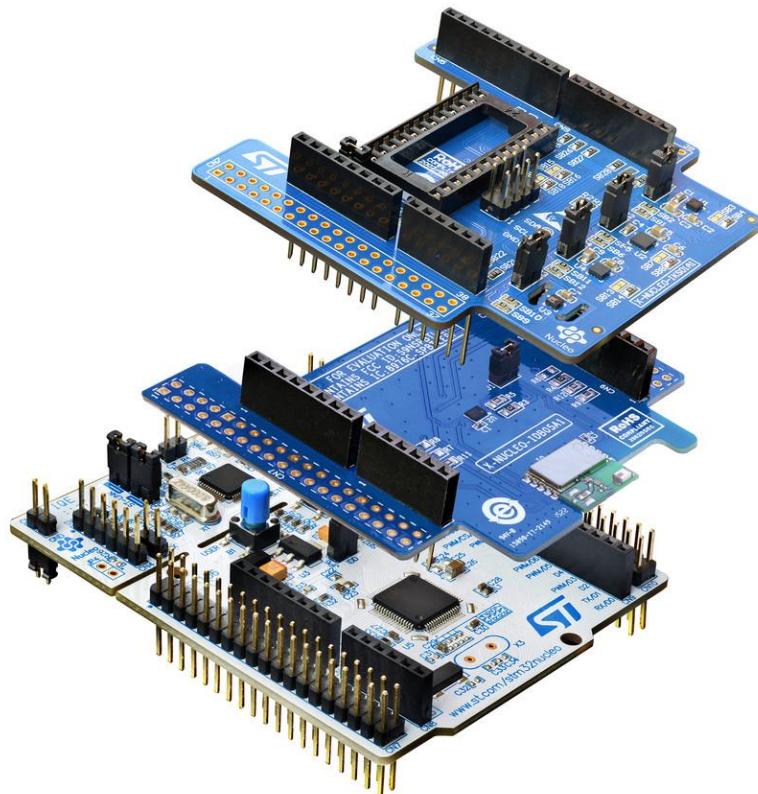
Complete product range
from ultra-low power to high-performance



www.st.com/stm32nucleo

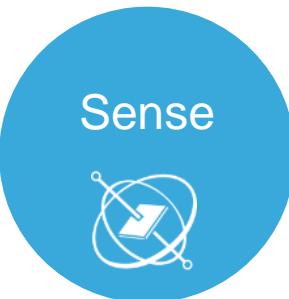
STM32 Nucleo Expansion Boards

26



<http://www.st.com/x-nucleo>

33 expansion boards covering all the key functions



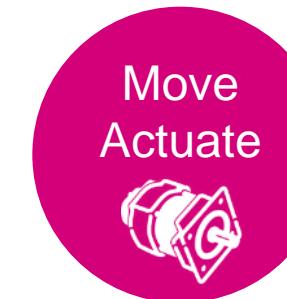
Sense



Connect



Power
Drive



Move
Actuate



Translate

Motion &
environmental sensors

Proximity sensor

Microphone

BLE
Wi-Fi

Sub-GHz

NFC

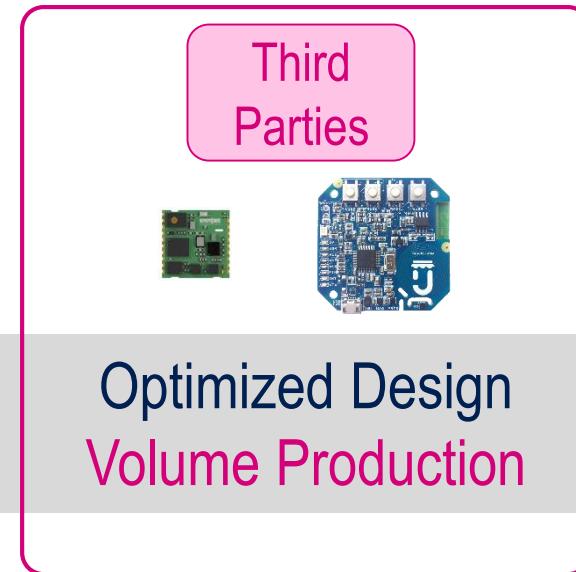
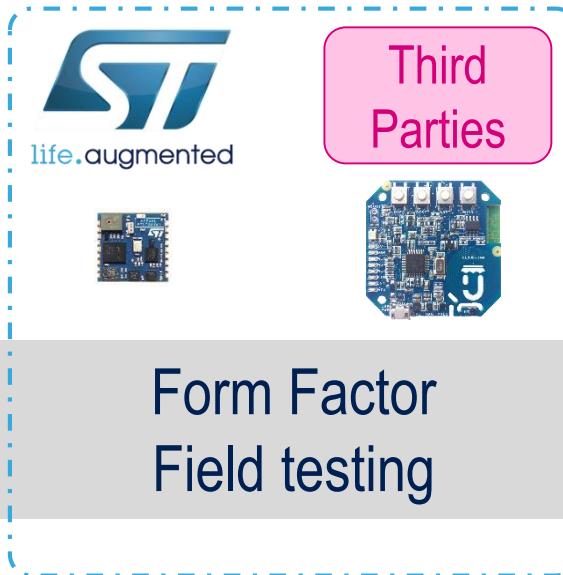
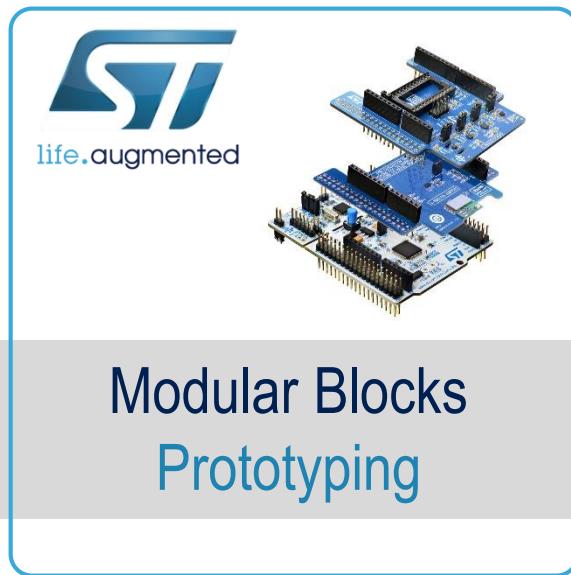
Power management
LED Boost

Motor driver
Actuator

Audio
Op-amp

From Prototype to End Device

Partnership with third parties - Form factor devices and customization



Final
Device



Easy Porting of developed software to final Product

HW/SW optimization and support for production

Simple Vs Advanced Use Cases

28

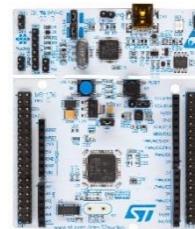
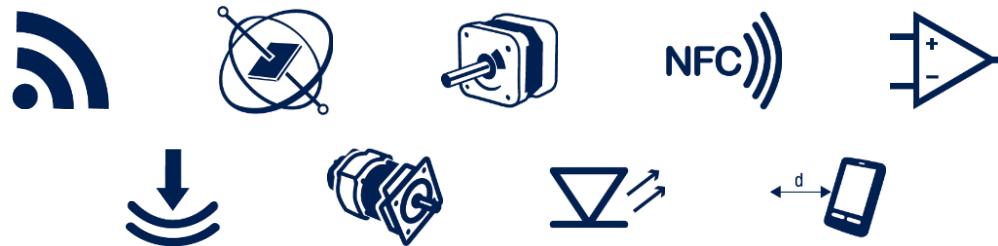
Expansion SW (X-Cube)

vs

Function Pack

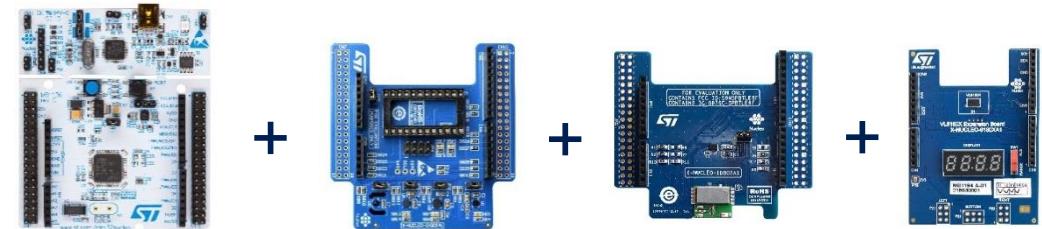
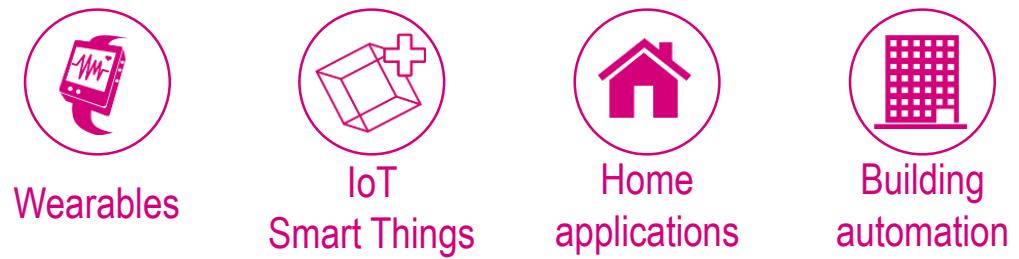
- Prototype with a single expansion board

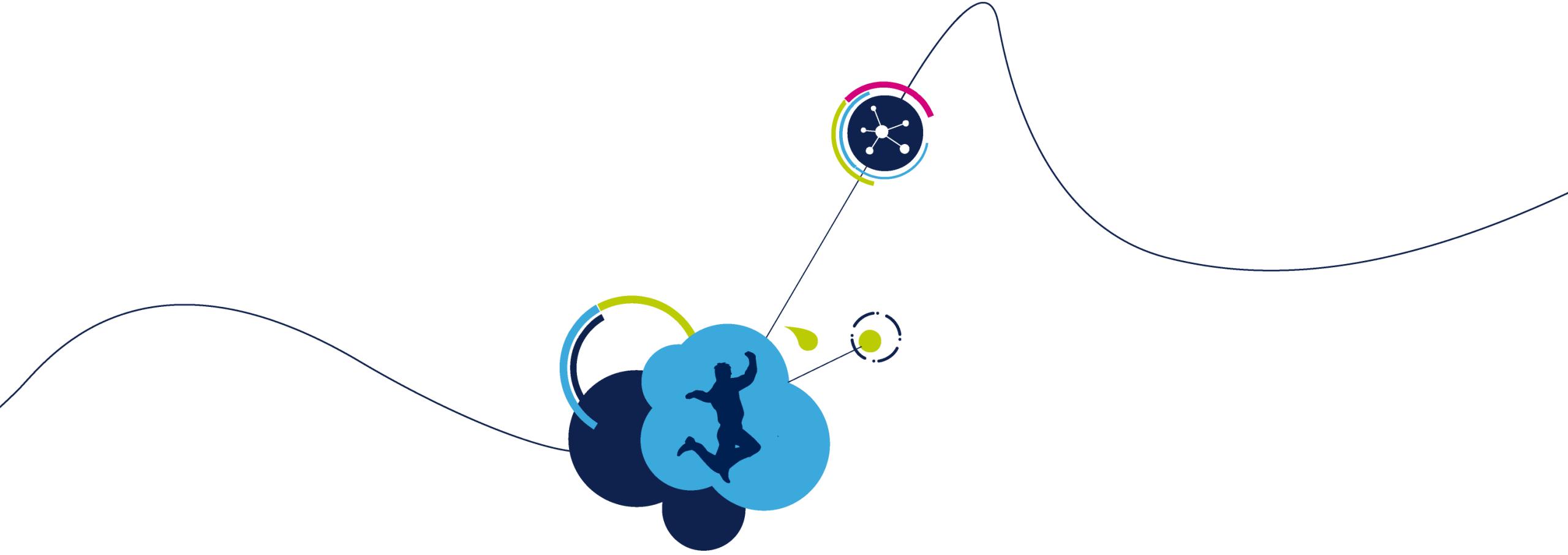
Sample applications



- Create advanced use cases based on multiple expansion boards

Pre-integrated application example





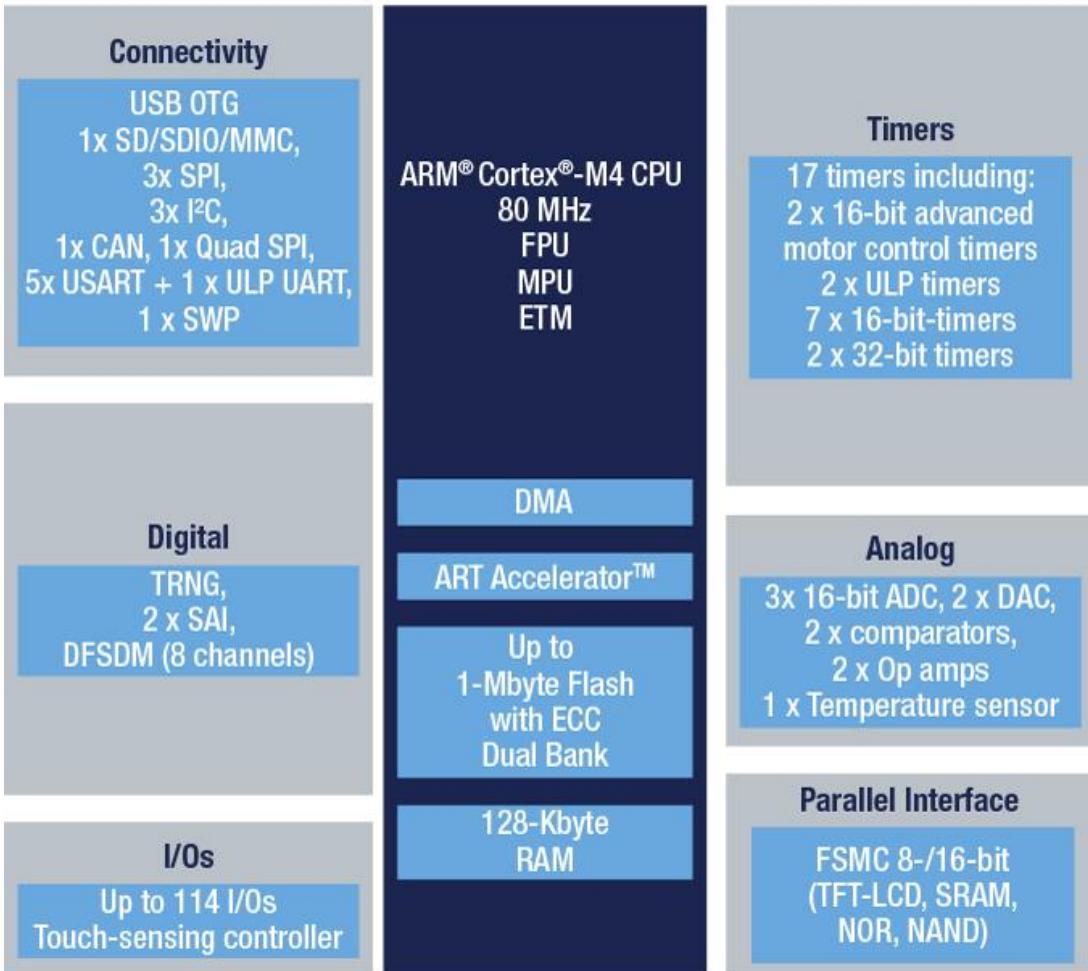
Overview of the STM32L475

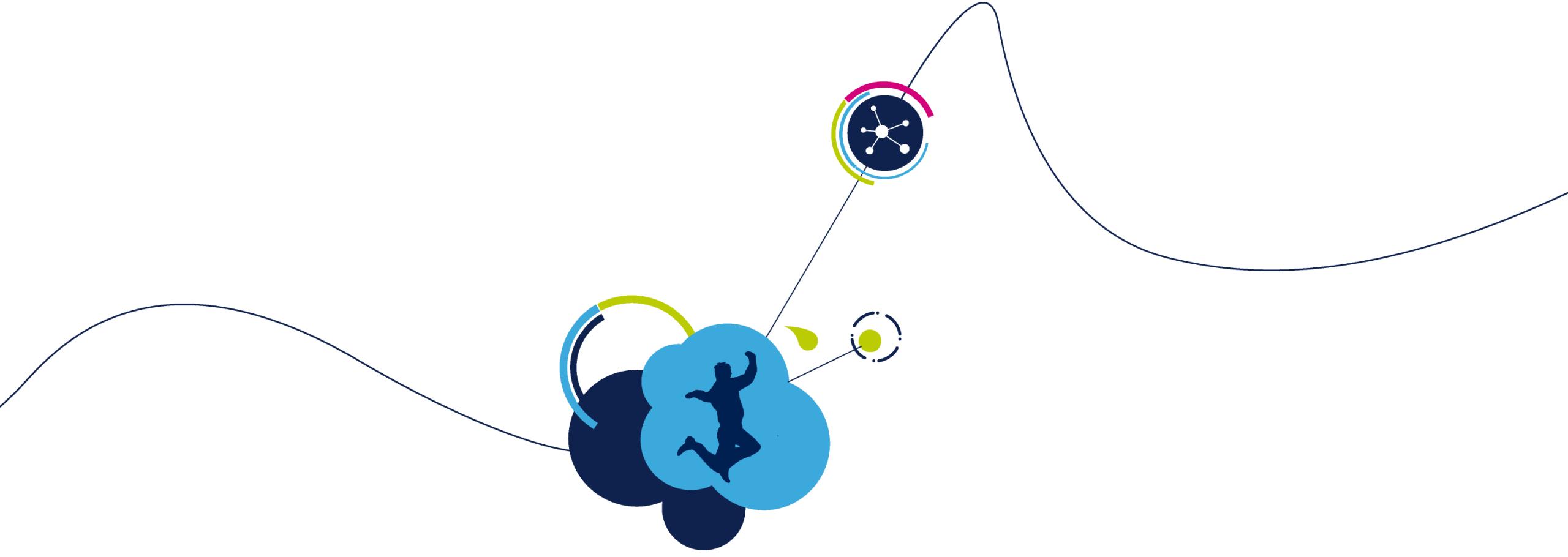
Overview of the STM32L475

Key features

- Cortex M4 @ 80 MHz with DSP, FPU and ART
- 1.71V – 3.6V supply 80 MHz Full functional
- 1MB Flash dual bank / 128KB RAM
- USB OTG FS –LPM Battery Charging Detection
- 3 x Ultra-low-power 12-bit ADC @ 5 MSPS
- Touch-Sensing 24 channels
- Ultra-low power
 - VBAT
 - Down to 160 μ A/MHz dynamic
- I²C FM+
- SPI: variable data length
- USART
- LP UART & 16-bit Timer
- FSMC, Quad SPI
- CAN, SWPPI, SDMMC, 2x SAI
- Digital filter for Sigma delta modulator
- 17 x timers
- Analog: Op-Amps, comparators, DAC, VREF, Temp
- RNG

STM32L475



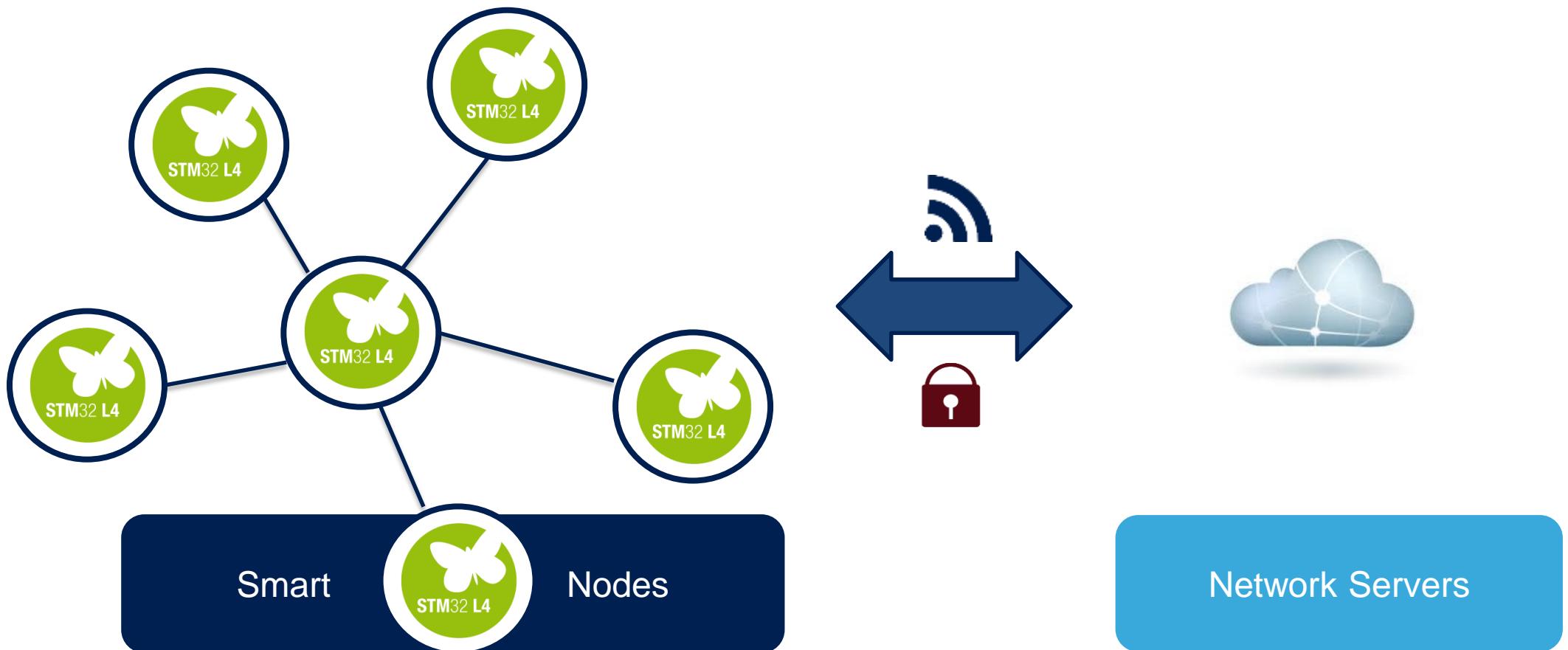


Overview of the STM32L4 Discovery Kit IoT Node

STM32L4 Discovery Kit IoT Node

32

Get connected seamlessly!



STM32L4 Discovery Kit IoT Node

33

Open the door to remote services

Direct connection to cloud servers

Low-power long-range communication

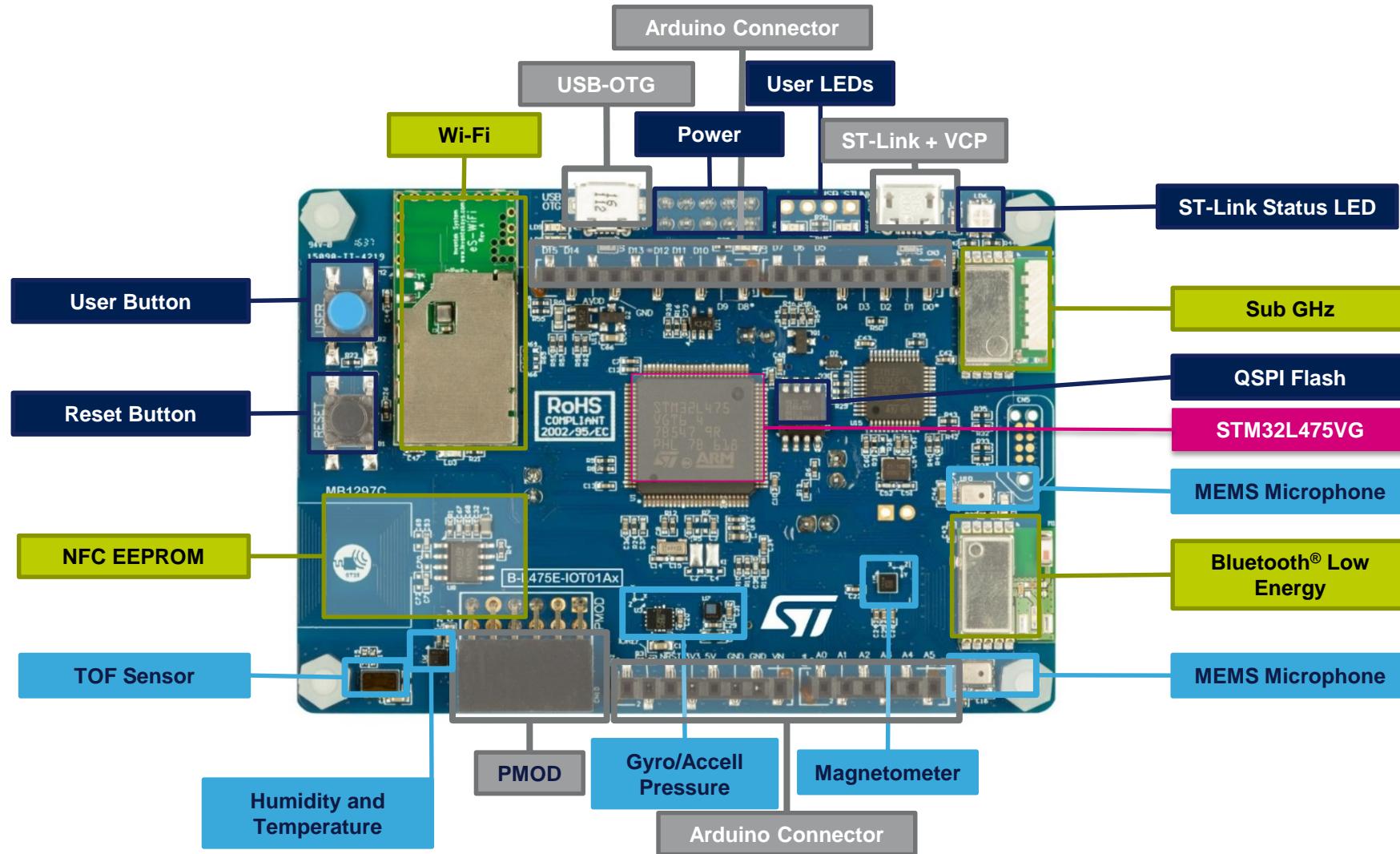
Environmental awareness: humidity, pressure, temp

Detection hub: motion, proximity, audio



STM32L4 Discovery Kit IoT Node

Multi-link communication, multiway sensing



Comprehensive Software Libraries

35

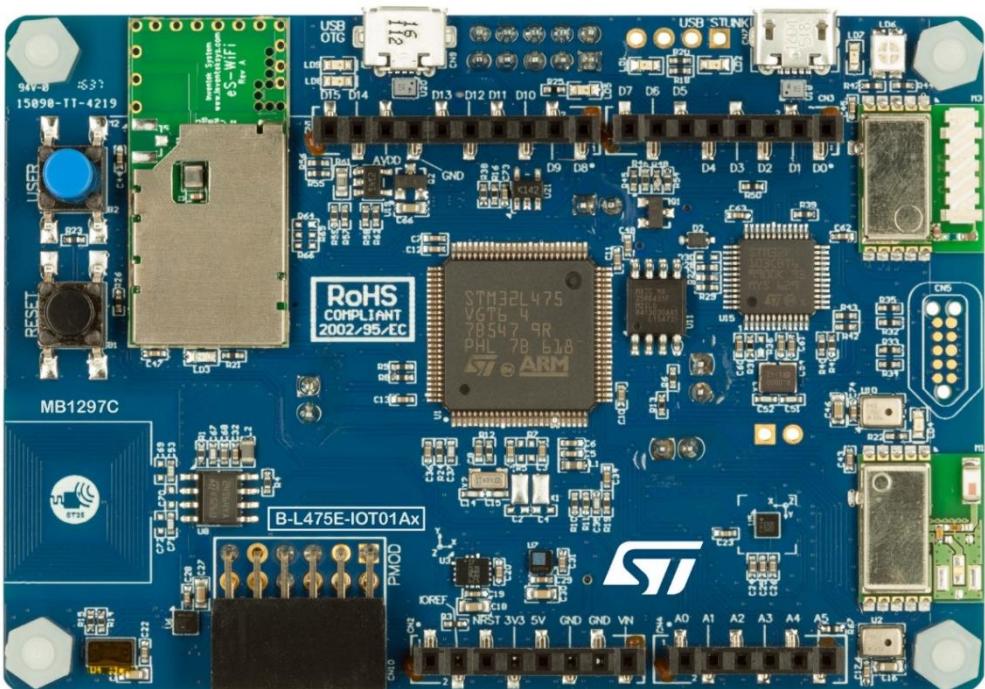
Instant showcase

SW Libraries for STM32L4 MCU & sensors

Connectivity SW protocol stacks

Cloud service connectors (AWS, Azure)

Demo examples
(X-CUBE-AWS, X-CUBE-AZURE, FP-CLD-AZURE1)



Wireless Connectivity – Wi-Fi

36

- Inventek ISM43362 Wi-Fi Module

- 802.11 b/g/n compliant Broadcom MAC/Baseband/Radio module
- Fully contained TCP/IP stack to minimize host CPU requirements
- FCC and CE certified
- Secure Wi-Fi authentication supporting WEP-128, WPA-PSK (TKIP), WPA2-PSK



Wireless Connectivity - Bluetooth®

37

- ST SPBTLE-RF Bluetooth® Low Energy Module
 - Based on our ST BlueNRG-MS Wireless Network Processor
 - Bluetooth® Low Energy 4.1 compliant
 - FCC and BQ certified module with integrated balun & antenna



Wireless Connectivity - SubGHz

38

- ST SPSGRF-915 Sub-GHz Module (915 MHz - US)
 - FCC and IC certified module with integrated balun & antenna
 - Supports 2-FSK, GFSK, MSK, GMSK, OOK and ASK modulation schemes
 - Long range (100s of meters) with an air data rate from 1 to 500 kbps

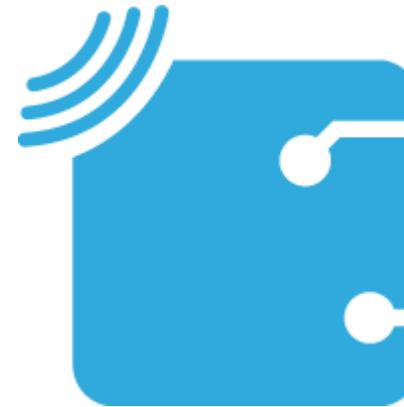


Wireless Connectivity - NFC

39

- ST M24SR64-Y Dynamic NFC/RFID Tag

- ISO14443-A NFC Forum Type 4 RF interface
- 106 Kbps Data Rate
- I²C 1MHz interface
- Up to 64kbit EEPROM memory
- 128-bit password for data protection
- 200 years data retention & 1Mcycles erase/write
- Configurable General Purpose Output signal for MCU wake-up
- RF disable feature



Wired Connectivity Features

- ST-Link V2
 - STM32 Programming and Debug Interface
- USB OTG FS
 - Full Speed USB On-The-Go Communication Interface
- PMOD
 - Peripheral Module Interface Supporting GSM, GPS, etc...
- Arduino Connectors
 - Arduino Compatible Connectors to interface with additional ST X-NUCLEO or 3rd Party Expansion Board (eg: LoRa)

- Full Range of Motion & Environmental MEMS Sensors

- LSM6DSL Accelerometer + Gyroscope Sensor
- LIS3MDL Magnetometer Sensor
- HTS221 Humidity + Temperature Sensor
- LPS22HB Pressure Sensor



- Integrated High Accuracy Proximity/Range Sensor

- VL53L0X Time-of-Flight Range Sensor



- Digital Microphones

- MP34DT01 MEMS Digital Microphones
 - Voice & Audio Recognition Functions
 - Acoustic Beam Forming using the ST Open Software Acoustic Beam Forming Library



User Resource Features

42

- Reset and User Buttons
 - Board Reset and Programmable Application Buttons
- User LEDs
 - Programmable Application LEDs
- QSPI Flash
 - 64Mbit for Data Storage and Program Execution
- Selectable Power Supply
 - ST-Link, USB-OTG, Arduino or External Power

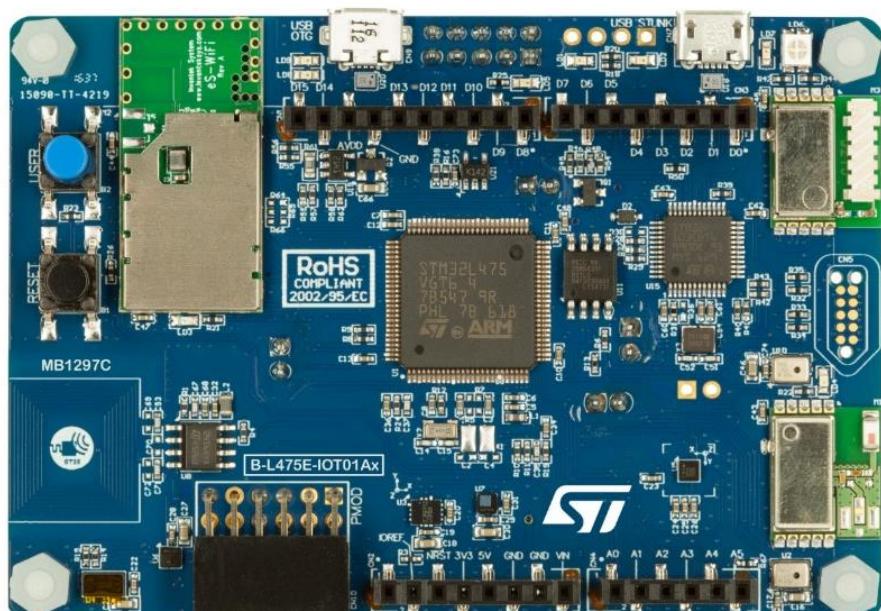
Advantages of a Single Board

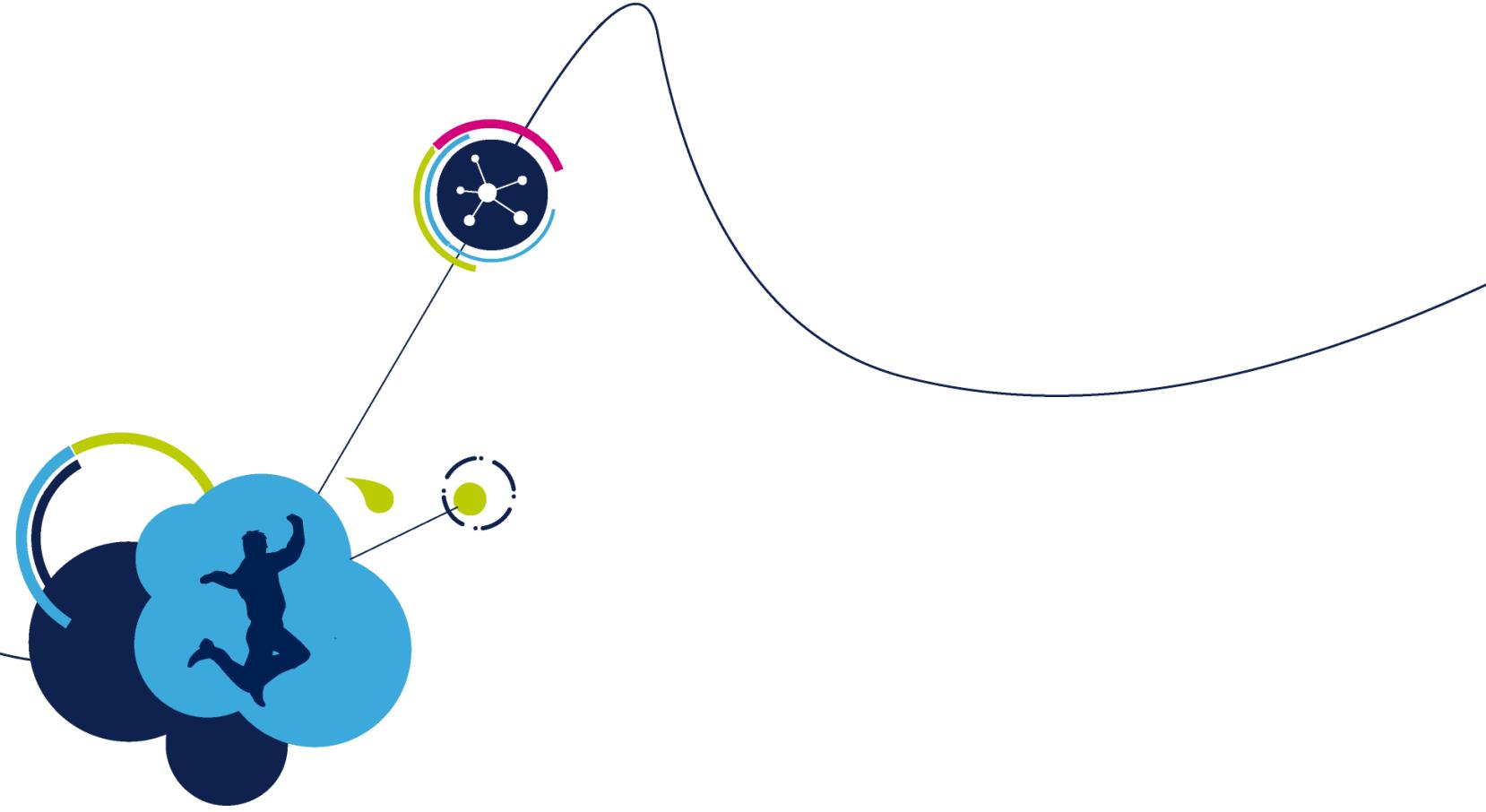
- Easily Debug Hardware Issues.
- Included Collateral is Tightly Coupled
 - BSP Included for all Board Components
 - Cloud Connectivity Reference Solutions Included
- Cost Effective Development Solution (~\$60)
- No Need to Manage & Order Multiple Board SKUs.

Availability

44

Part number	Samples	Mass Market Availability	SubGHz frequency band	Regions with authorized use
B-L475E-IOT01A1	NOW	NOW	915 MHz	US
B-L475E-IOT01A2	NOW	NOW	868 MHz	Rest of the World

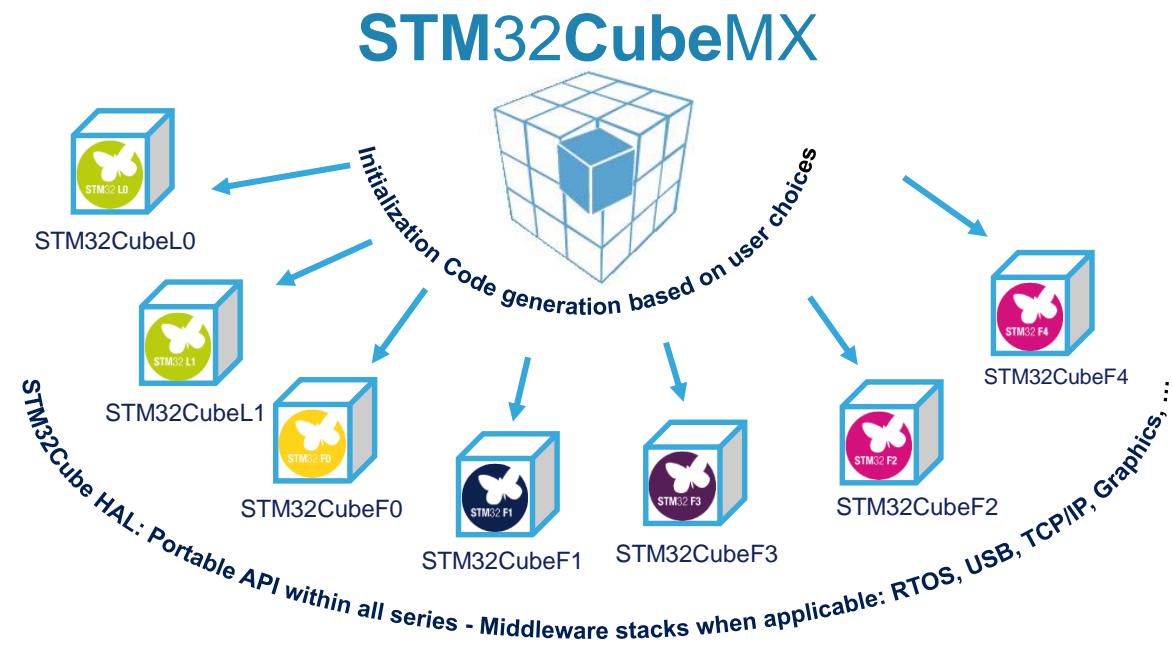




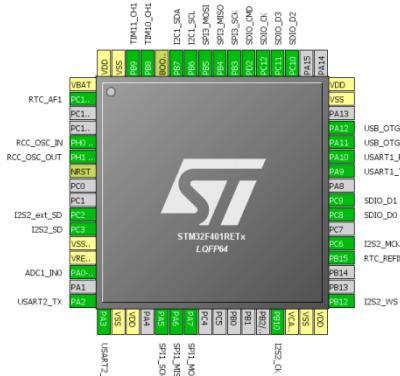
STM32Cube™ Introduction

STM32Cube™ Introduction

- STM32Cube™ includes:
 - STM32CubeMX, a configuration tool for generating user driven initialization
 - Firmware collateral, delivered for every STM32 device series (i.e. STM32CubeF4) with:
 - An STM32 Abstraction Layer embedded software: STM32Cube HAL
 - A consistent set of Middleware: RTOS, USB, TCP/IP, Graphics, ...



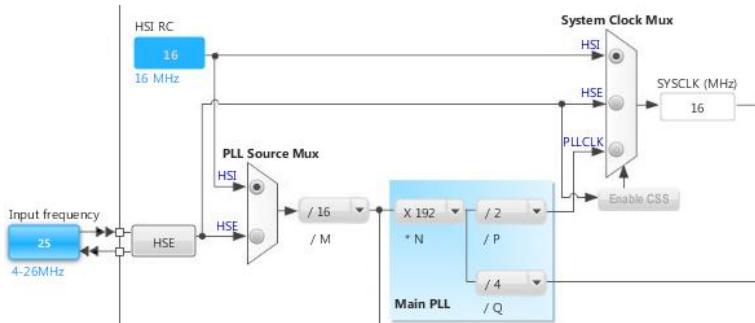
STM32CubeMX : Overview



STM32CubeMX



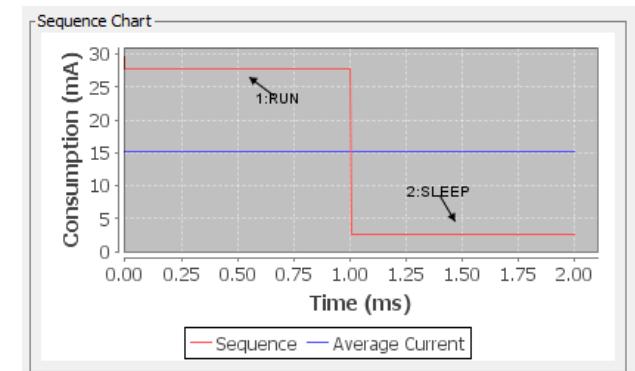
Clock Tree wizard



Basic Parameters	
Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1
Advanced Parameters	
Data Direction	Receive and Transmit
Over Sampling	16 Samples
Baud Rate	
BaudRate must be between 110 Bits/s and 10.5 MBits/s .	

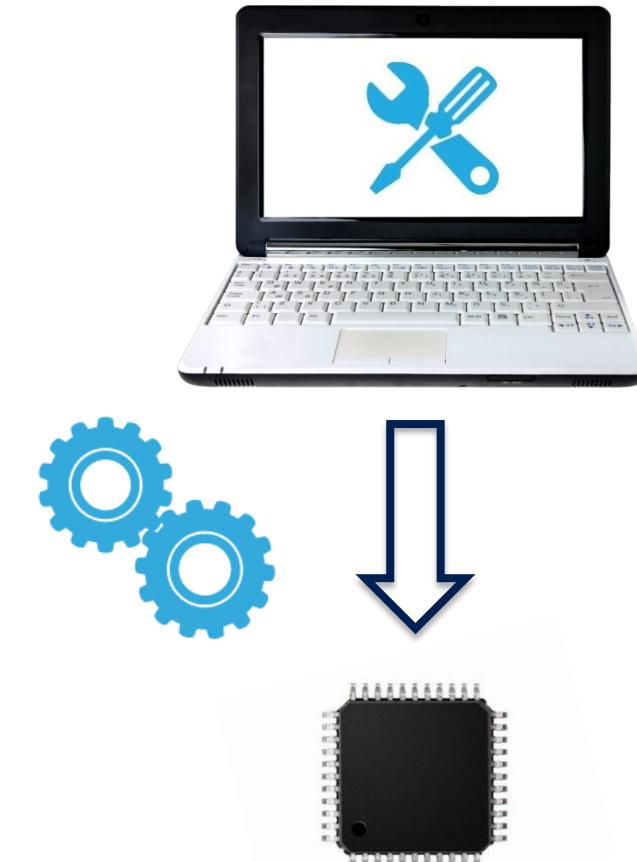
Peripherals & Middleware Wizard

Power Consumption Wizard



STM32CubeMX : User Code Configuration

48

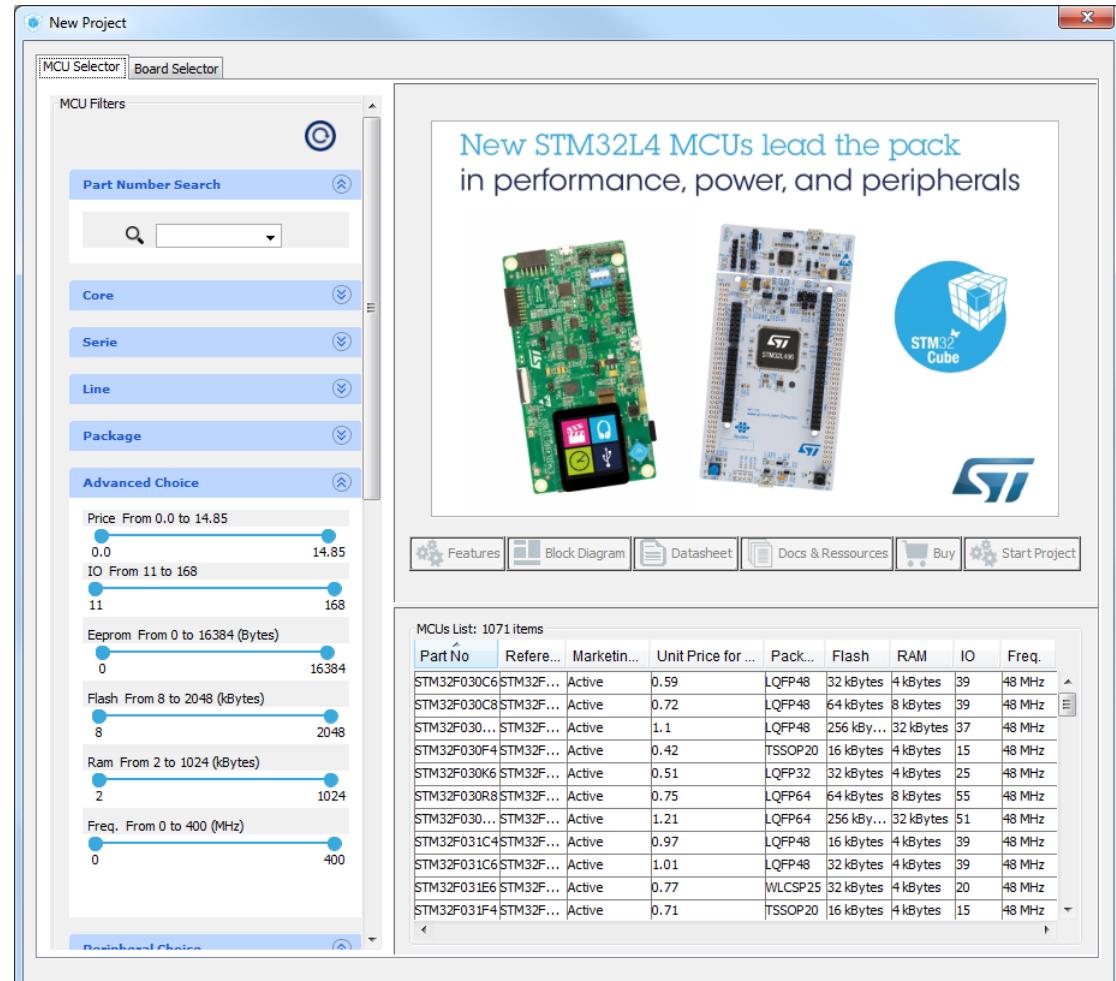


**Generates Initialization C Code
based on user driven
configuration!**

STM32CubeMX : MCU Selector

49

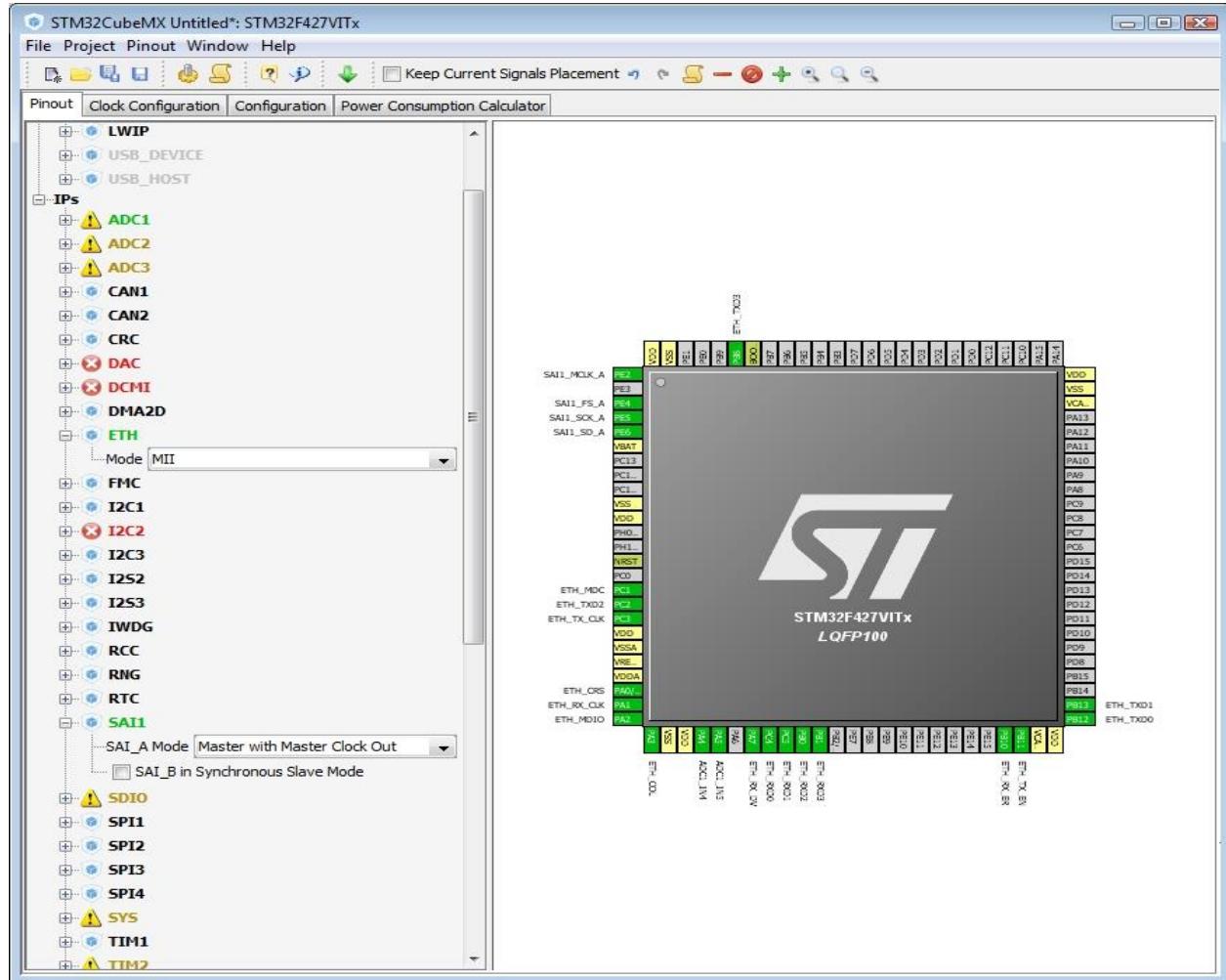
- Filter by:
 - Series
 - Line
 - Package
 - Peripherals



STM32CubeMX : Pin-out Configuration

50

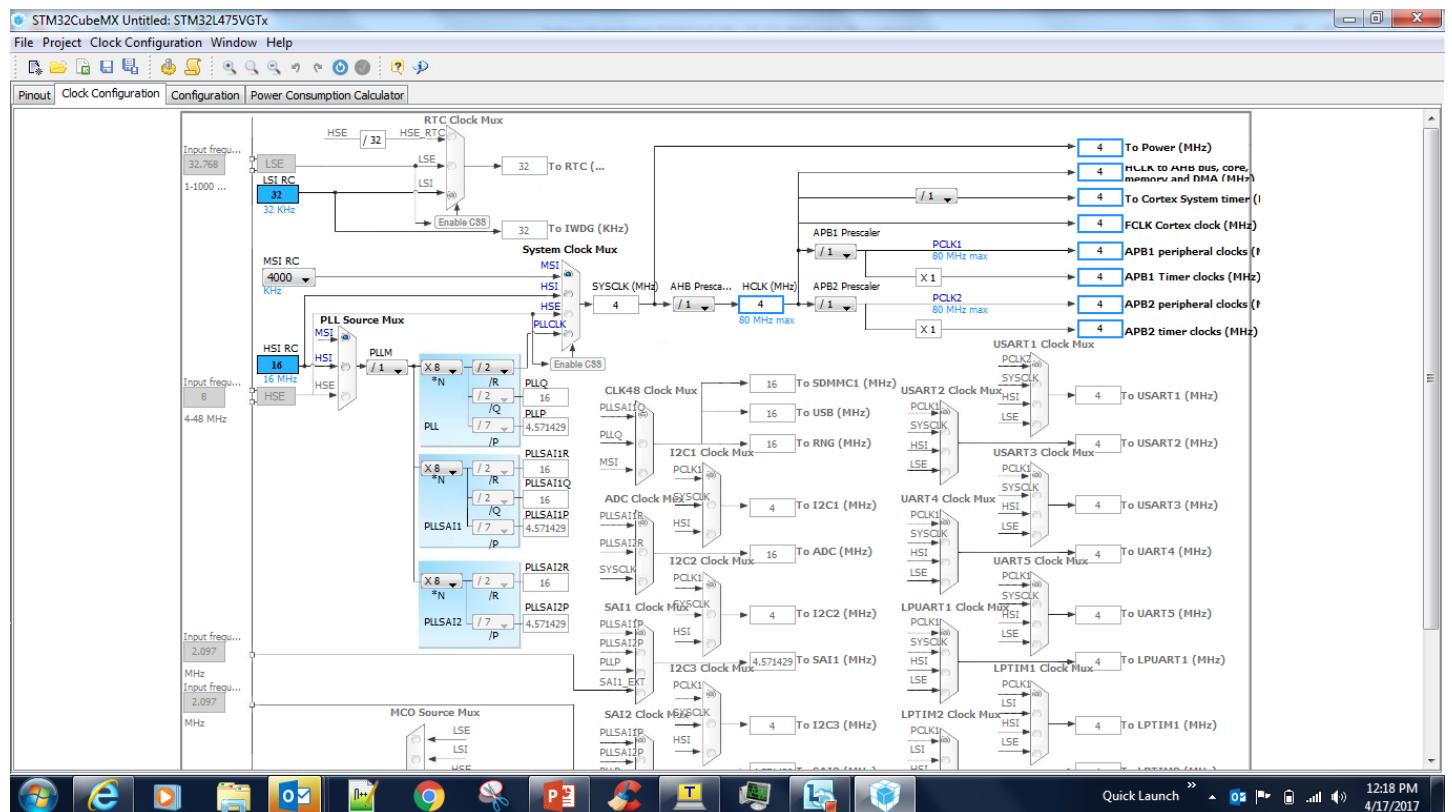
- Pinout from:
 - Peripheral tree
 - Manually
- Automatic signal remapping
- Management of dependencies between peripherals



STM32CubeMX : Clock Tree

51

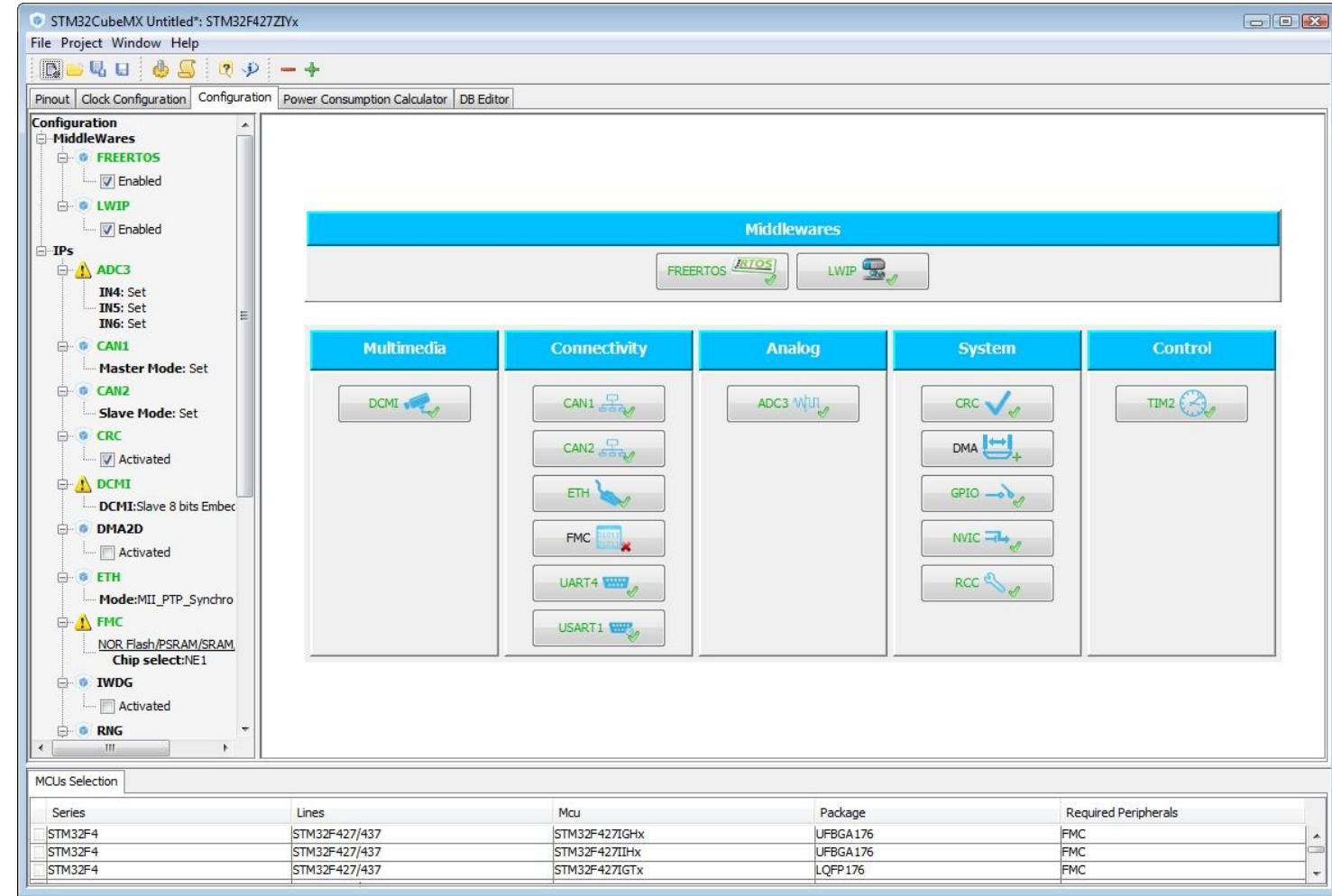
- Immediate display of all clock values
- Management of all clock constraints
- Highlight of errors



STM32CubeMX : Peripheral Configuration

52

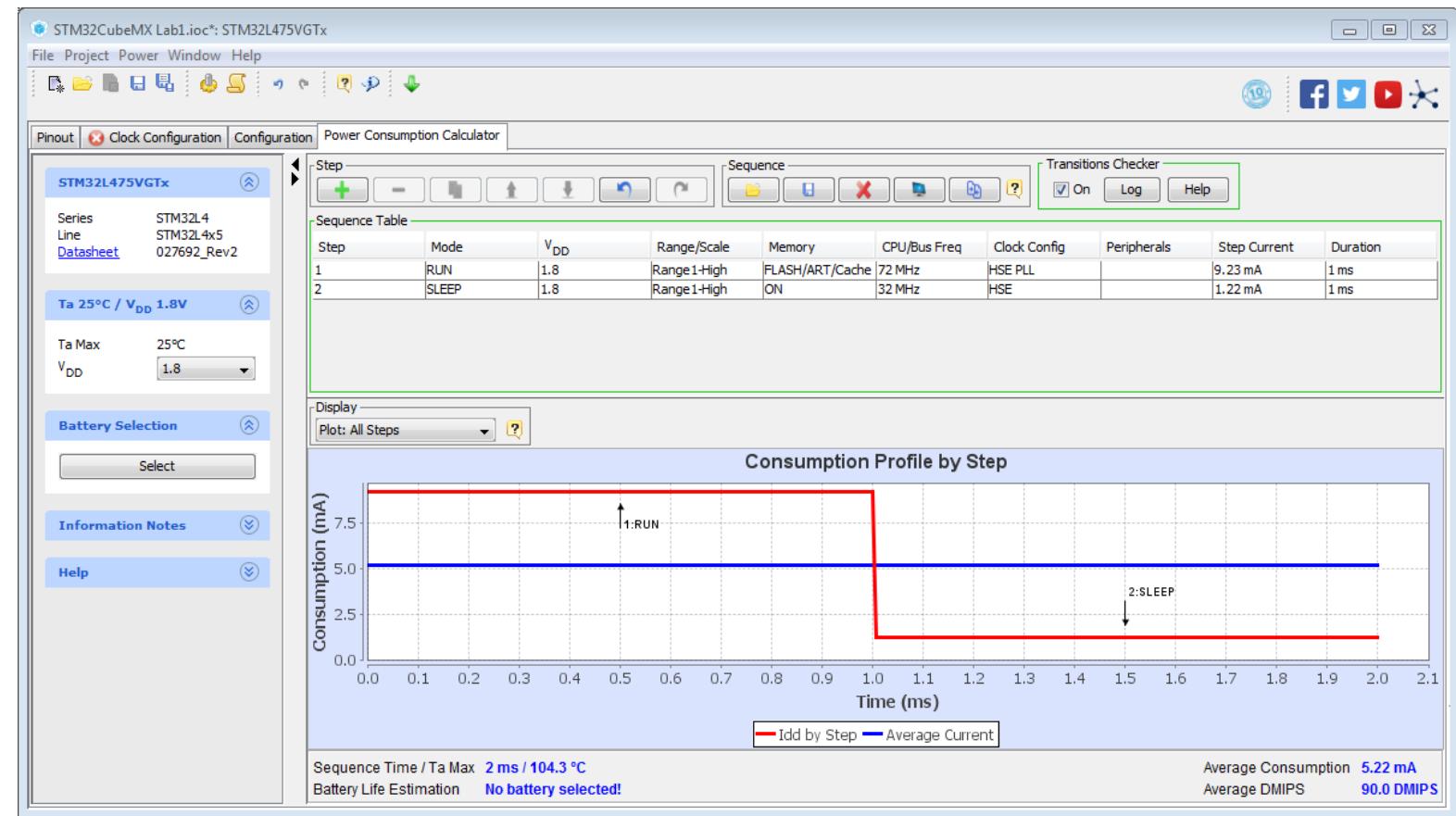
- Global view of used peripherals and middleware
- Highlight of configuration errors
- Manage:
 - GPIO
 - Interrupts
 - DMA



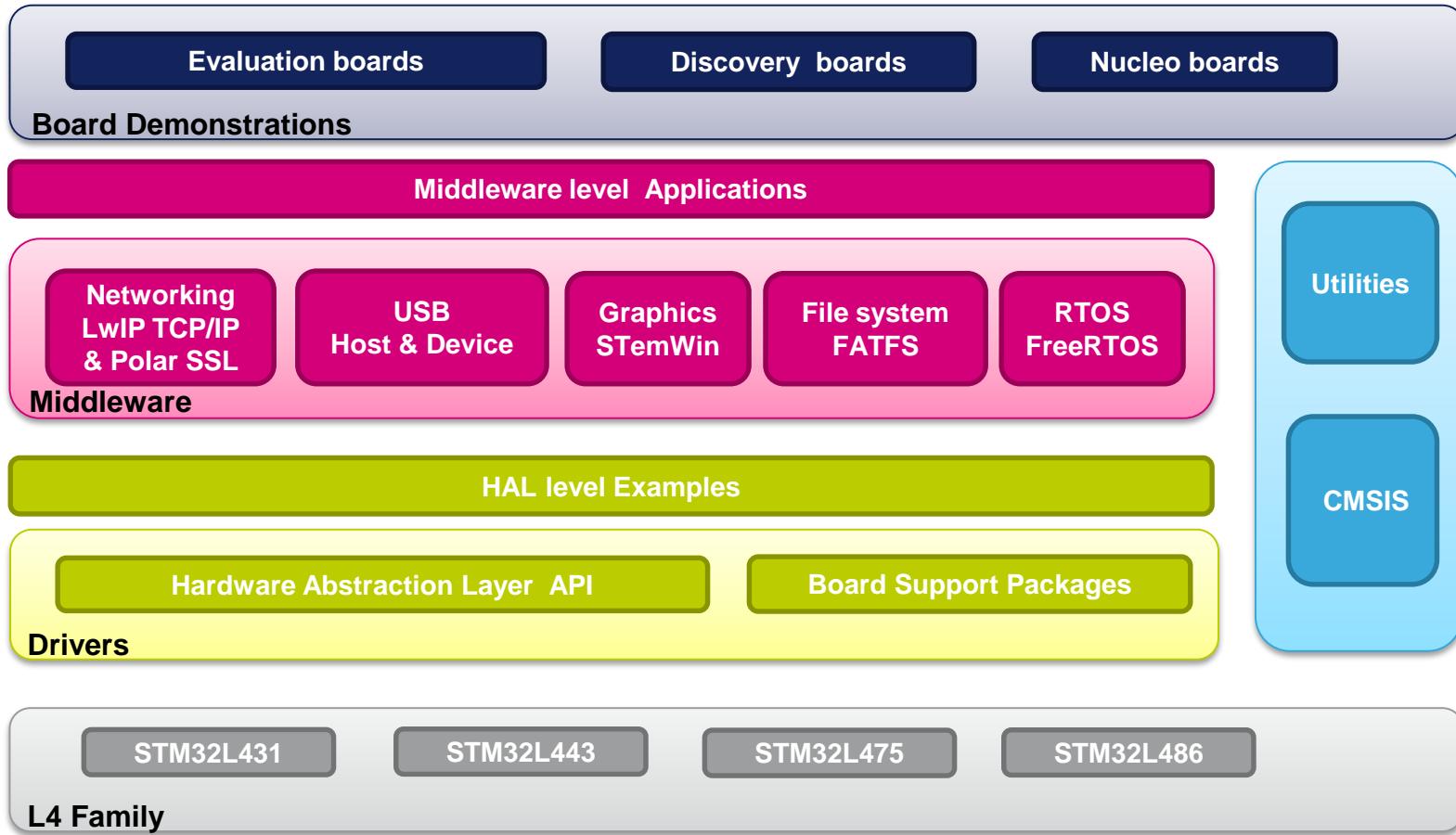
STM32CubeMX : Power Consumption Calculator

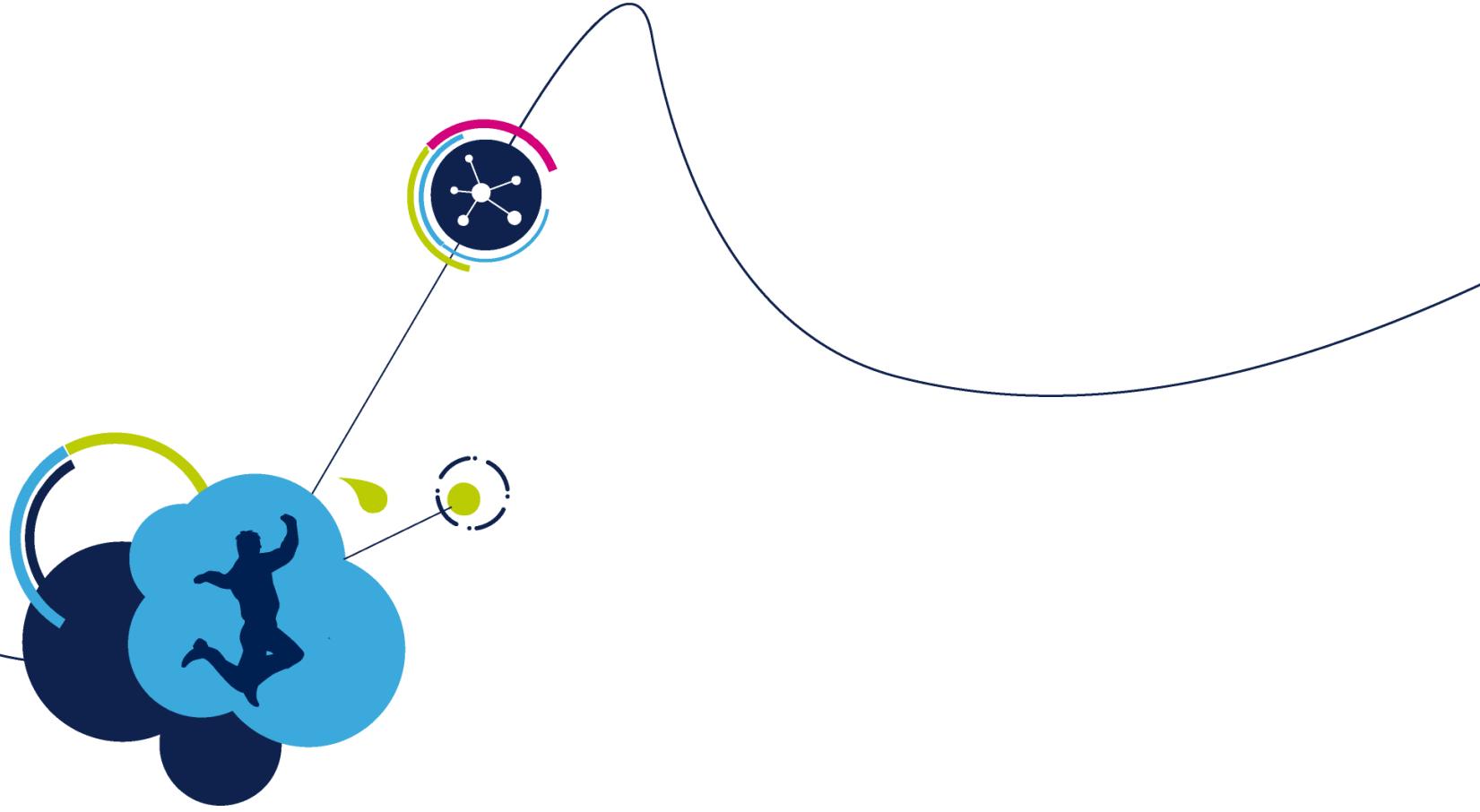
53

- Power step definitions
- Battery selection
- Creation of consumption graph
- Display of
 - Average consumption
 - Average DMIPS
 - Battery lifetime



STM32Cube : Firmware Components



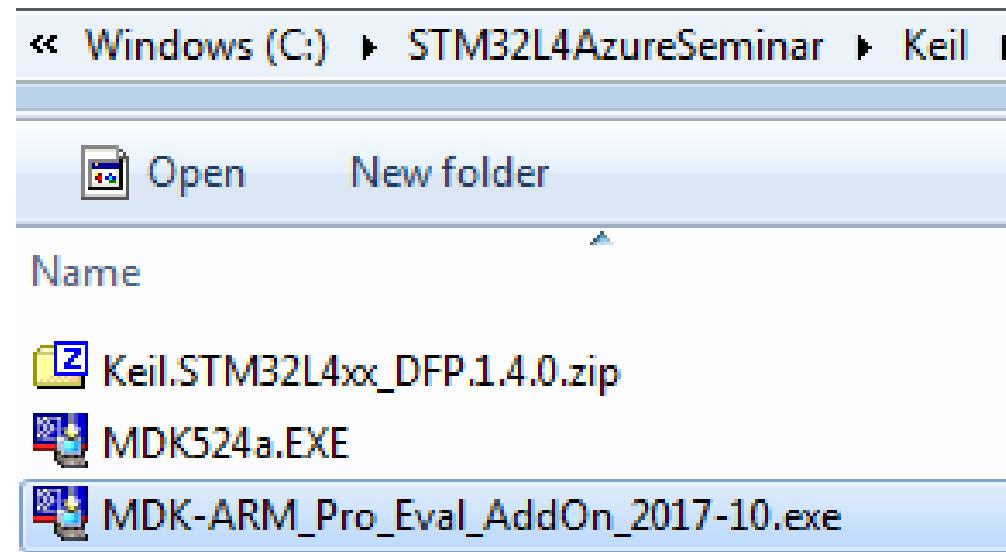


Keil License Installation

Step 2. Keil License Installation

- Browse to C:\STM32L4AzureSeminar\Keil\ and double-click on the file

MDK-ARM Pro Eval AddOn 2017-10.exe



Step 2. Keil License Installation

- Click on Next



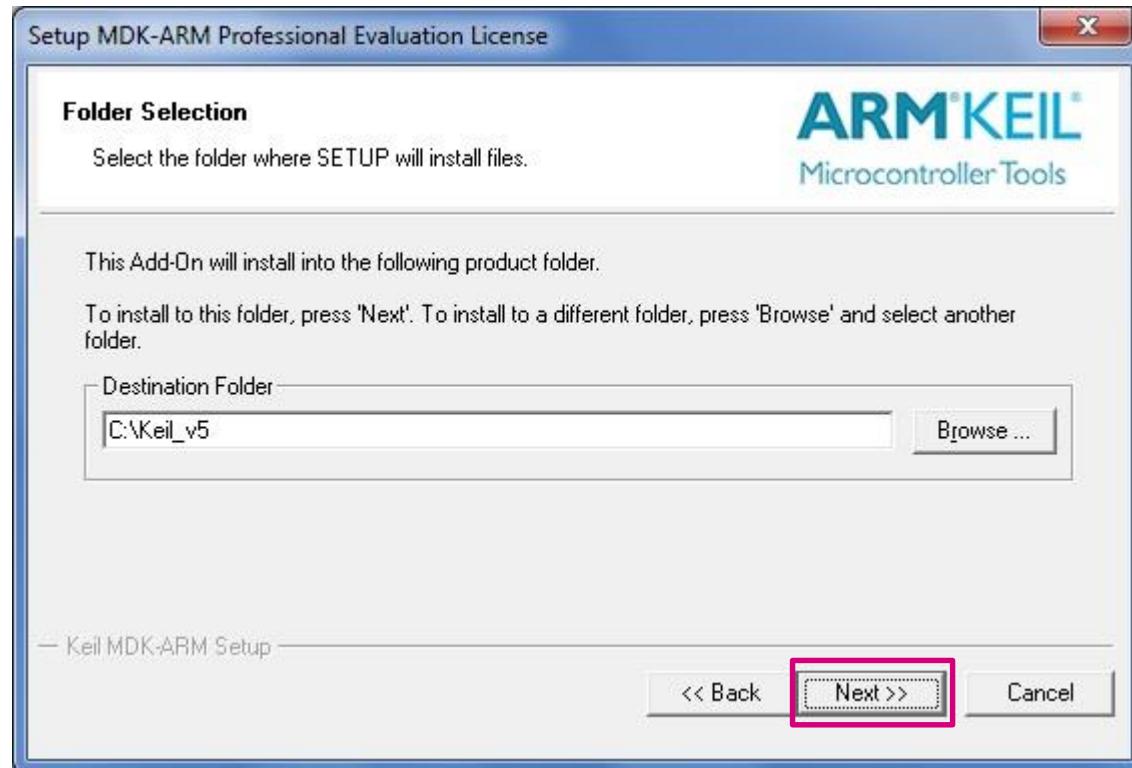
Step 2. Keil License Installation

- Accept the license agreement and click on **Next**



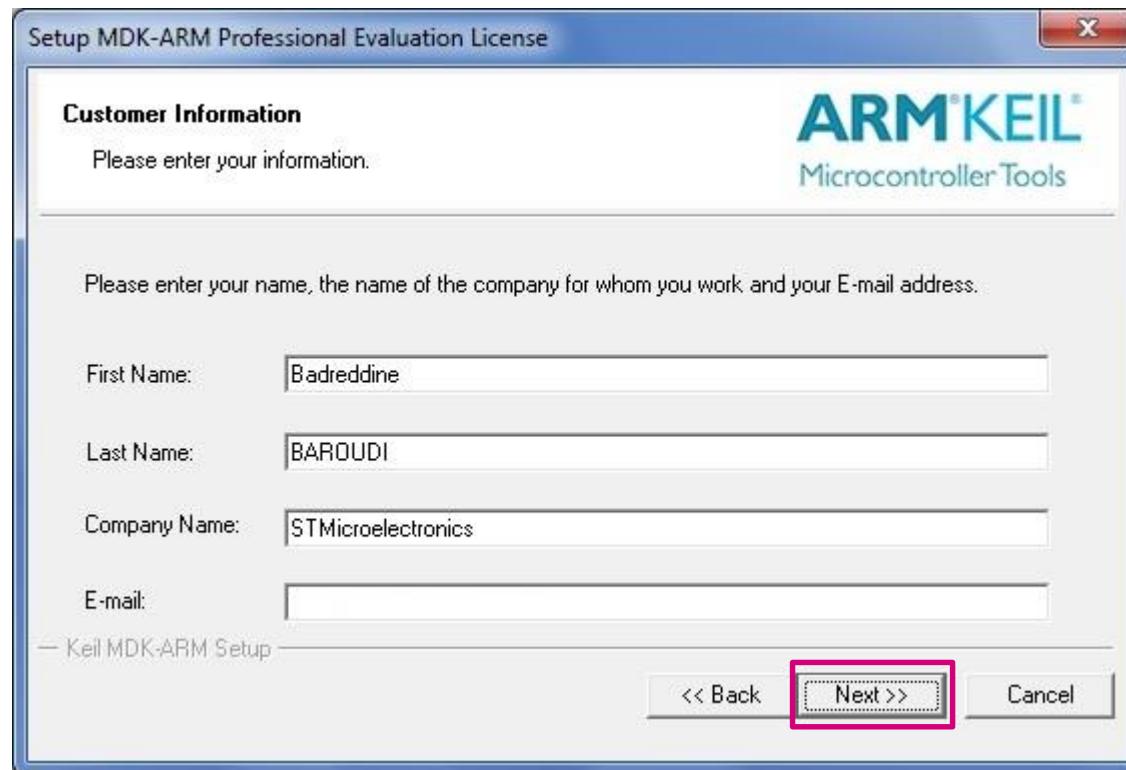
Step 2. Keil License Installation

- Click on Next



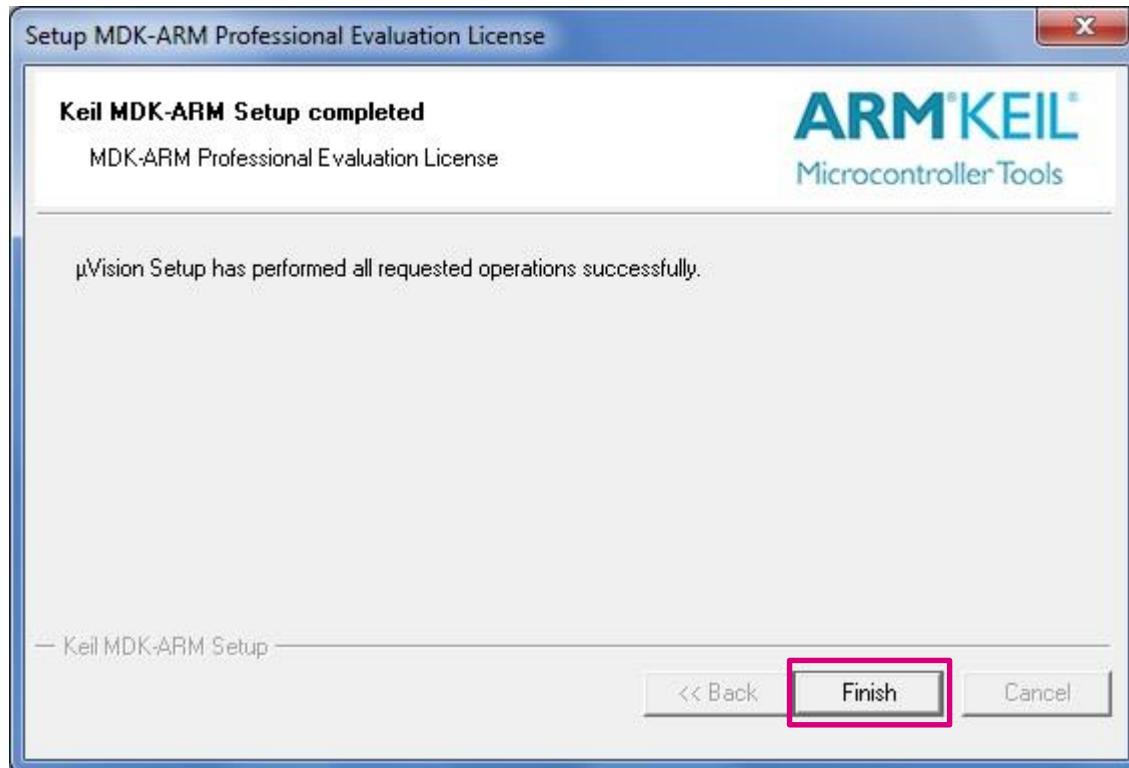
Step 2. Keil License Installation

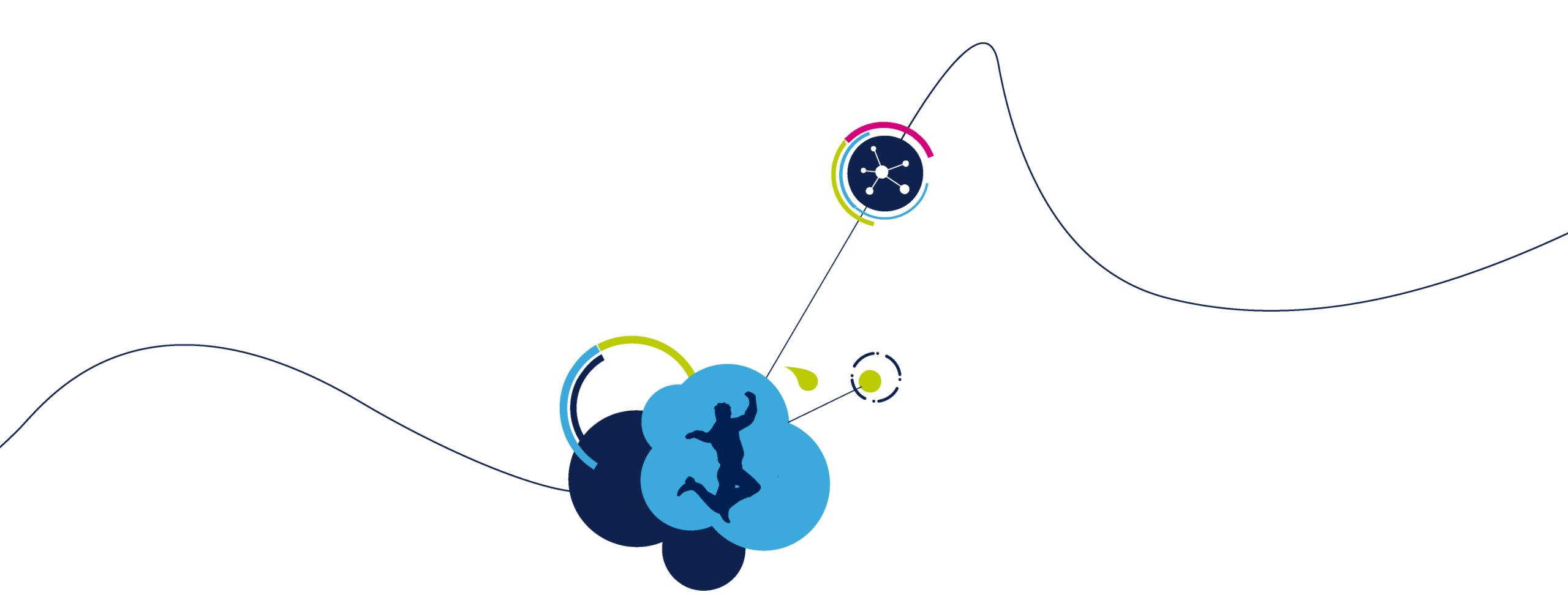
- Enter your contact information and click on **Next**



Step 2. Keil License Installation

- Click on **Finish**

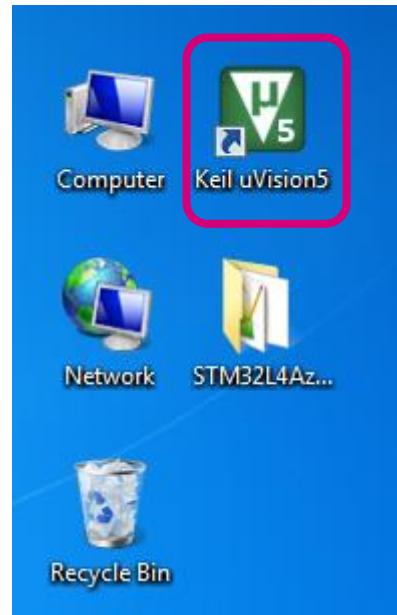




Keil Software Pack Installation

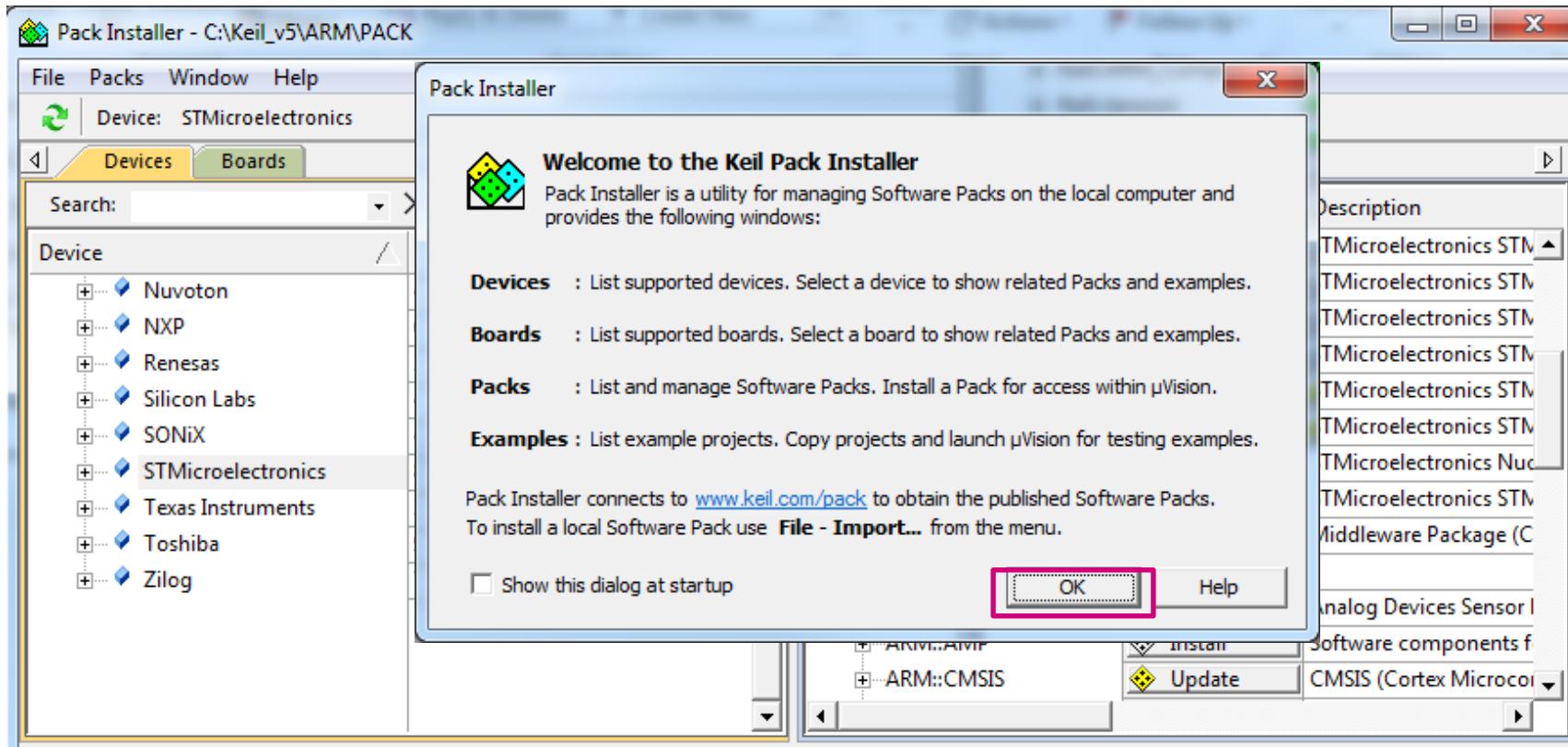
Step 3. Keil Software Packs

- Browse to your Desktop and double-click on Keil uVision5



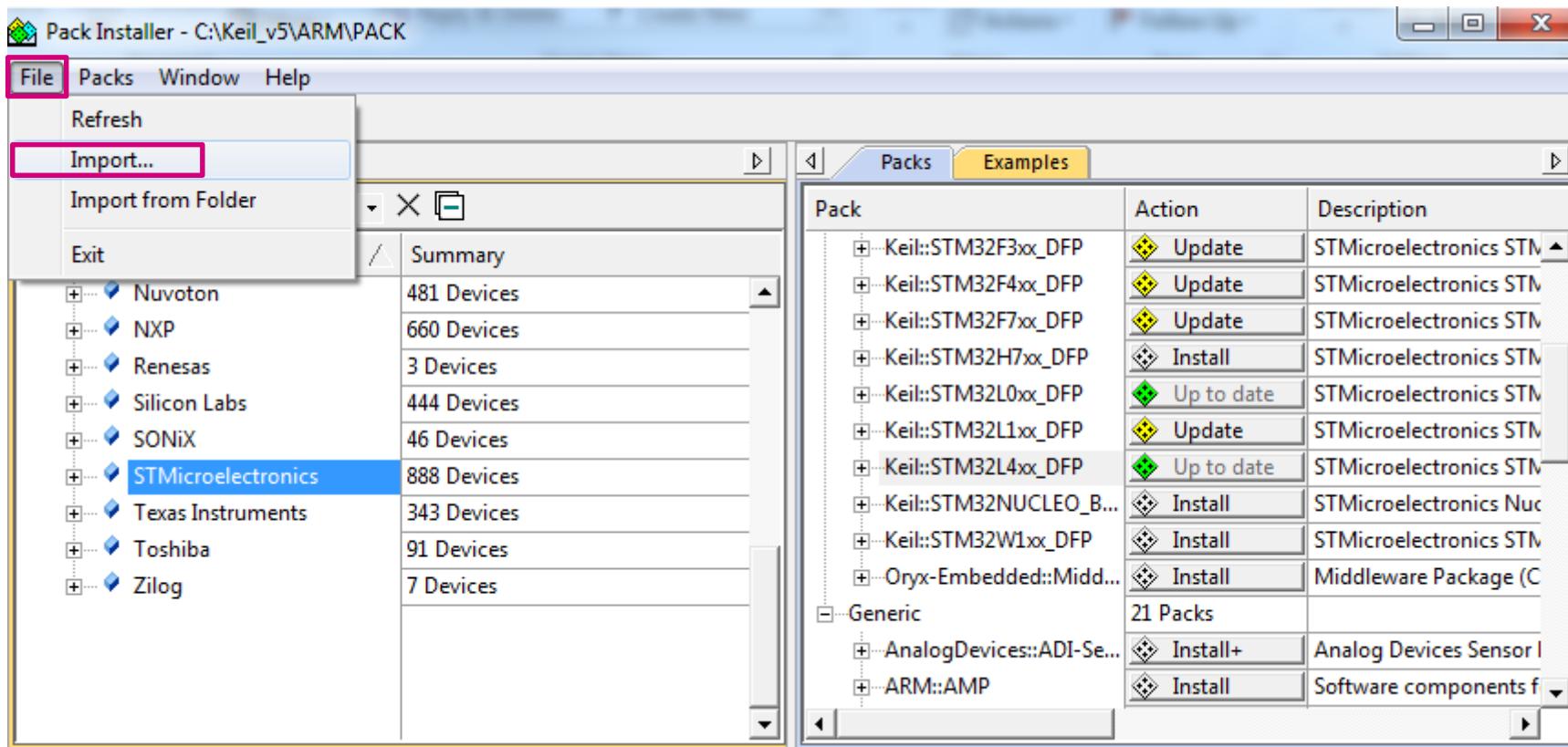
Step 3. Keil Software Packs

- Keil will automatically prompt you to install the Device Family Packs.
- Click on OK
- If the Pack Installer doesn't show up, click on Pack Installer icon



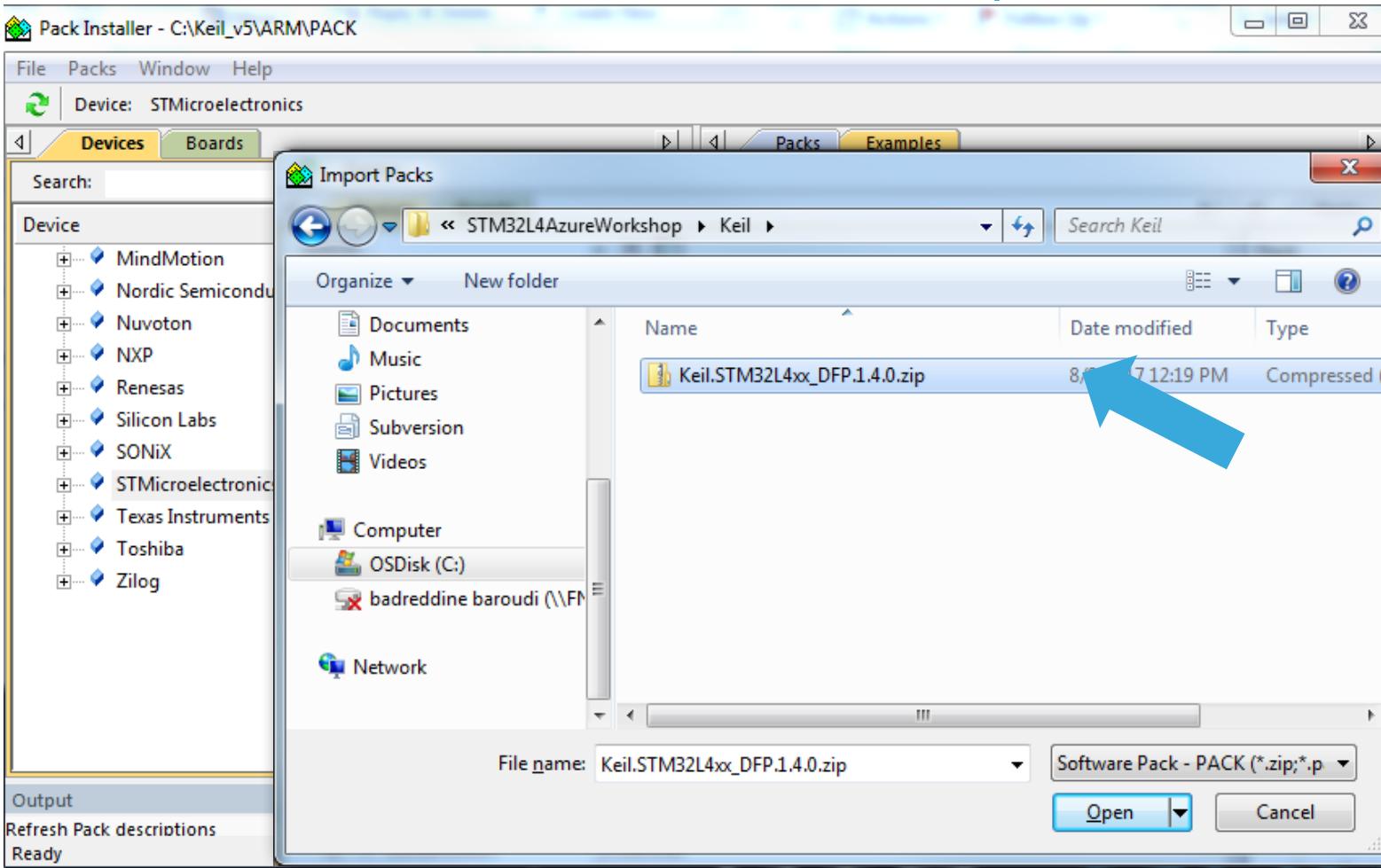
Step 3. Keil Software Packs

- We will manually import the STM32L4xx_DFP software pack...
- In Pack Installer, Select **File** and then **Import**



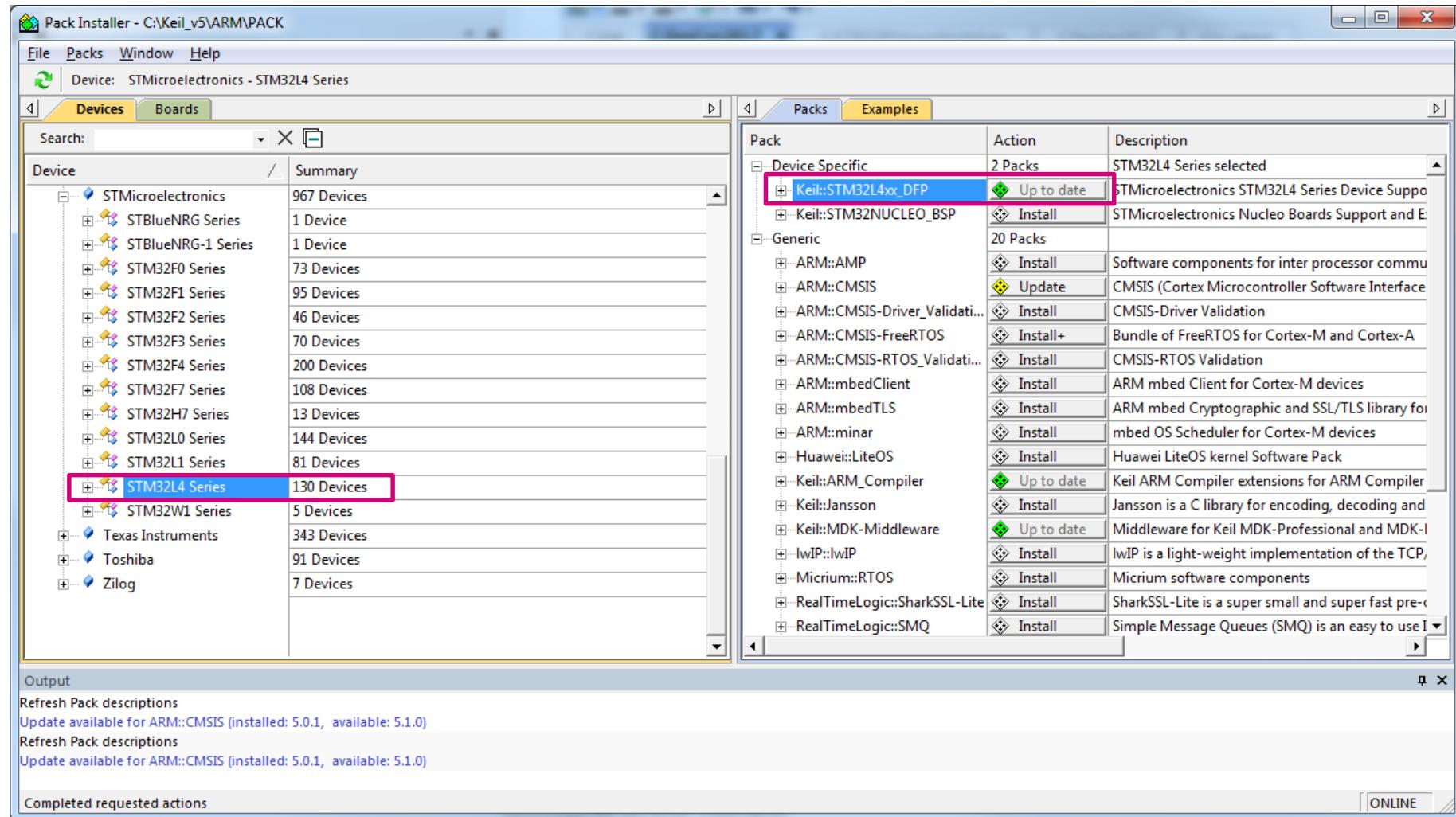
Step 3. Keil Software Packs

Browse to <C:\STM32L4AzureSeminar\Keil>
And then choose [Keil.STM32L4xx_DFP.1.4.0.zip](#)



Step 3. Keil Software Packs

After the STM32L4 Pack is installed, the Pack installer can be closed



Board Distribution



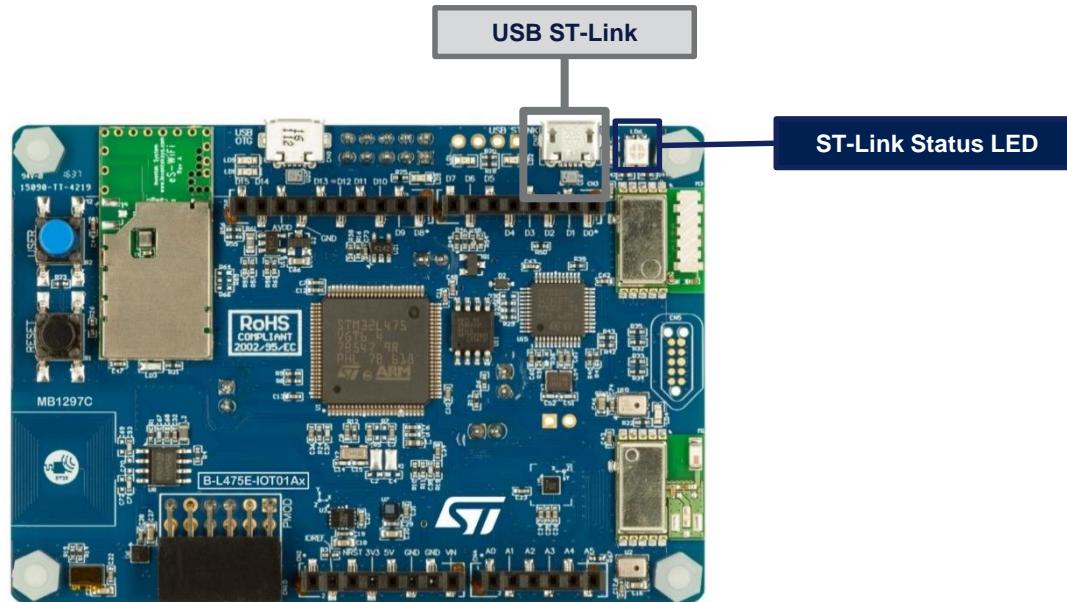
Participant Number
(2 characters)

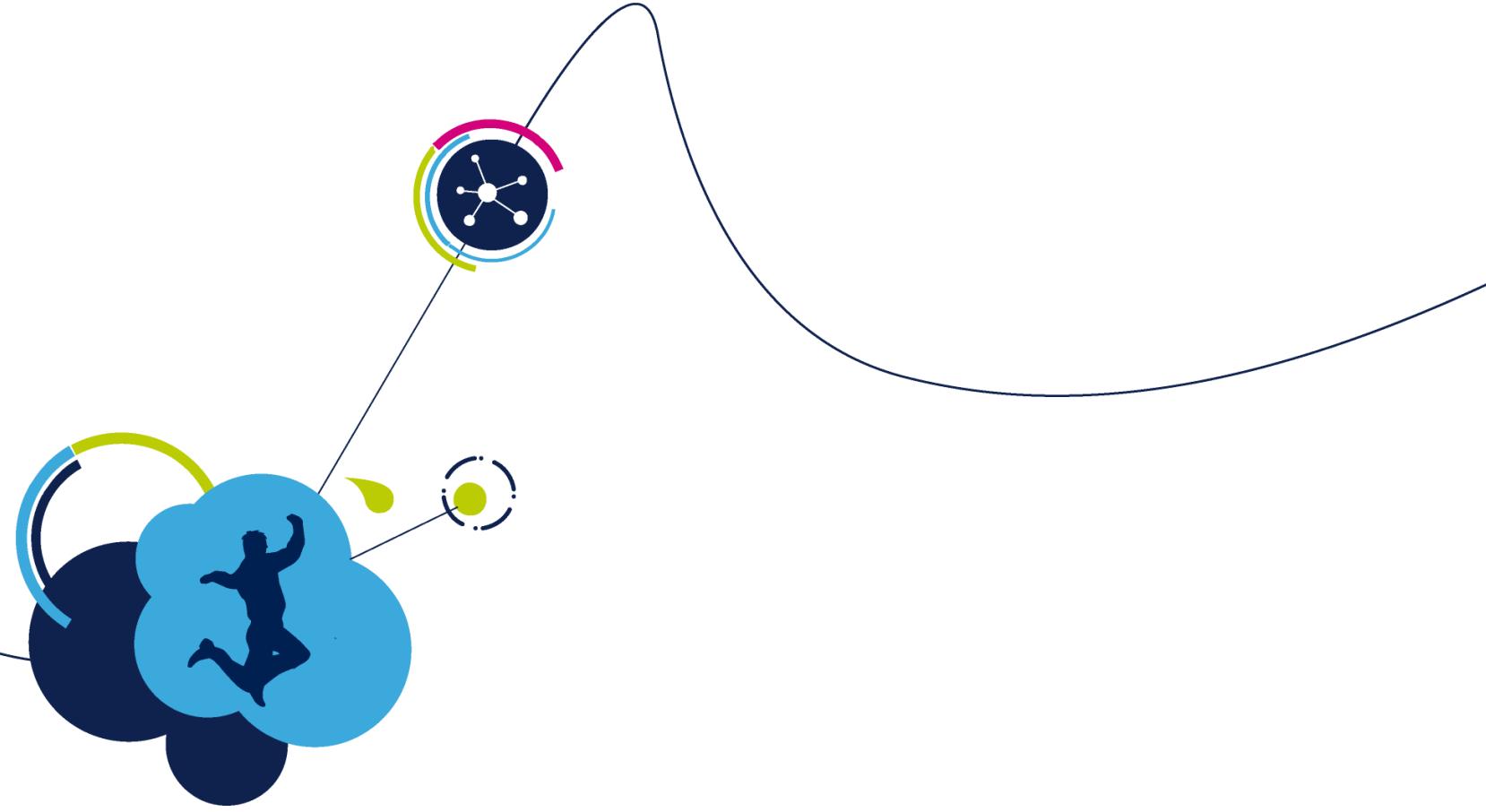
Each board package will have a label with a unique number.
During the lab sessions, this number will be referred to as
your Participant Number.

ST-Link Driver Installation

69

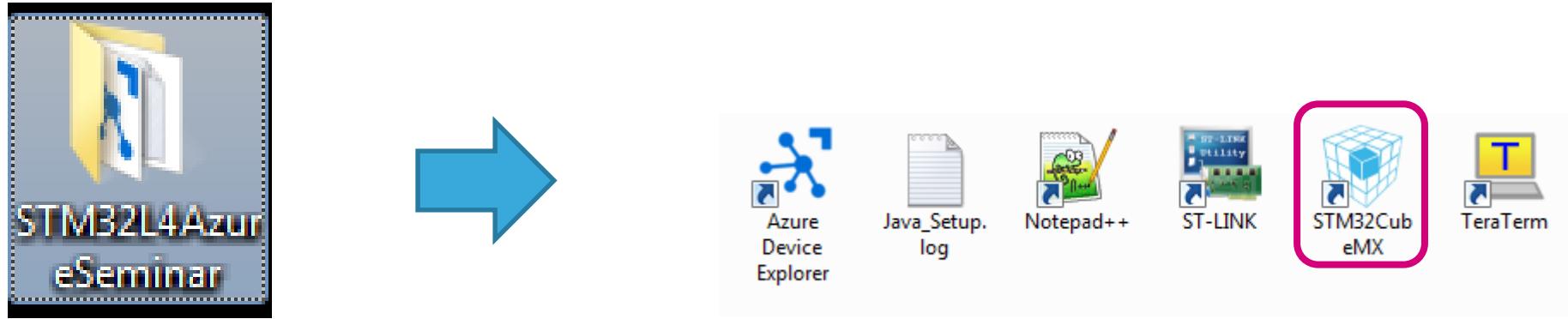
- Connect your PC to the USB **ST-Link**.
- The board will be powered through the **ST-Link** connection.
- The **ST-Link Status LED** will be steady when **ST-Link** is recognized.





Lab 1: Getting Started with STM32CubeMX – Blinking LED

Launch STM32CubeMX

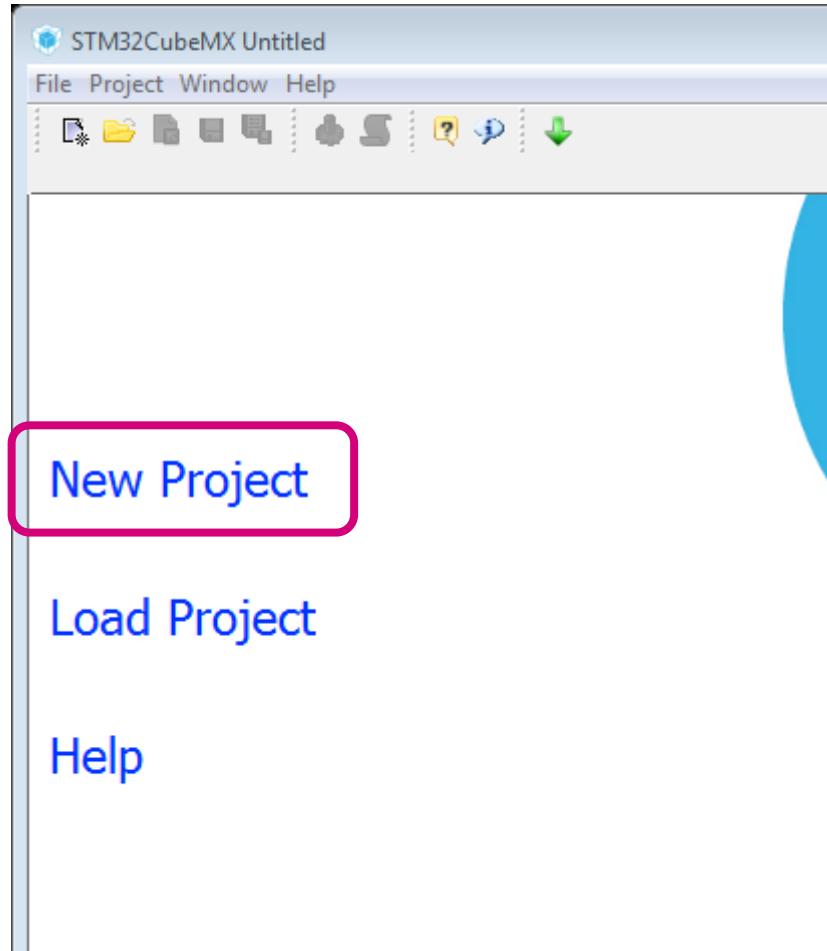


- Open the STM32L4AzureSeminar folder on your Desktop.
- Double click on the **STM32CubeMX** icon.

Create New STM32CubeMX Project

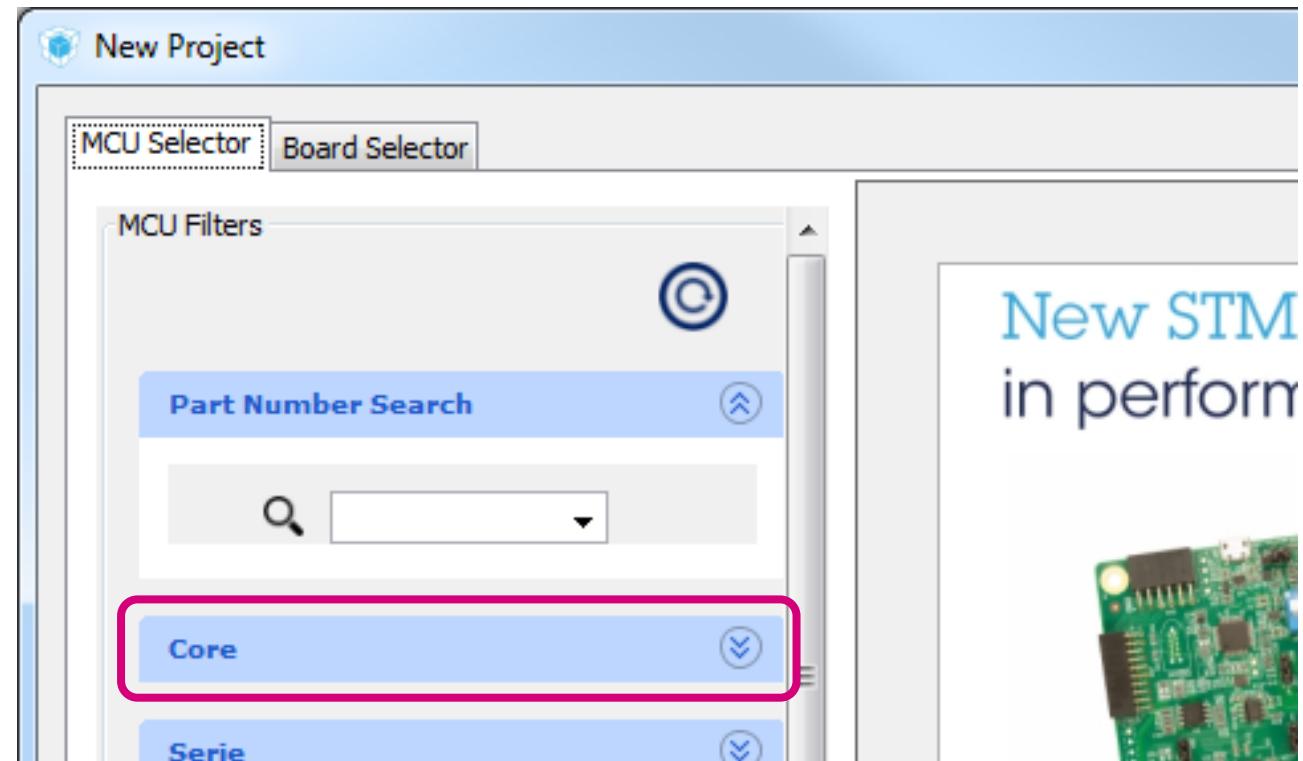
72

- Click on **New Project**



Select the Microcontroller

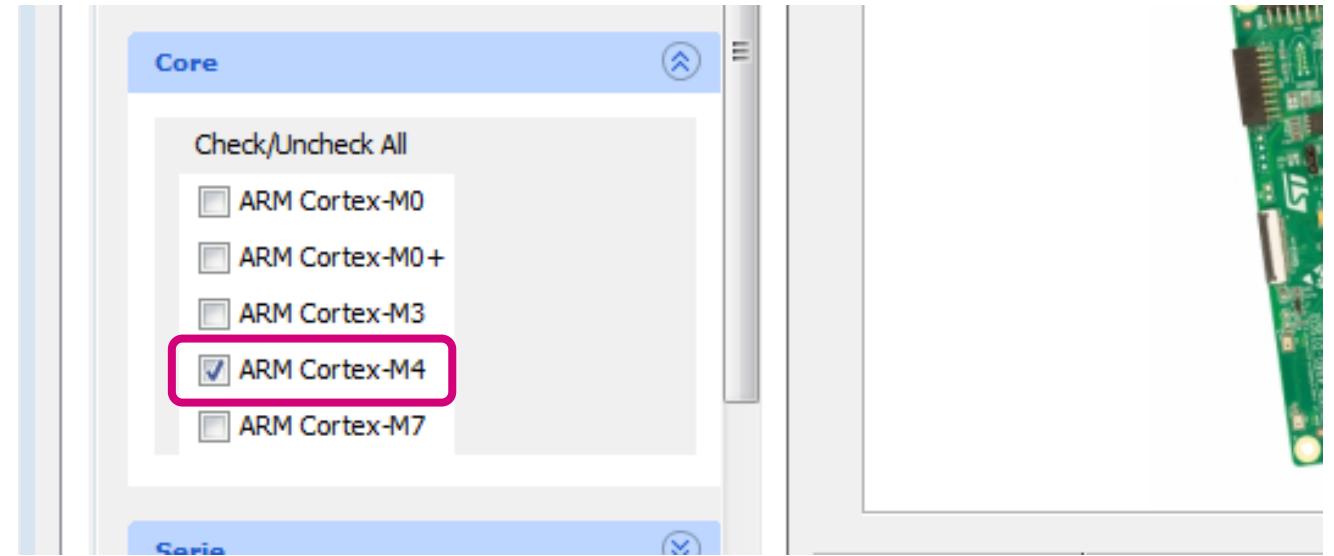
- Expand **Core**



Select the Microcontroller

74

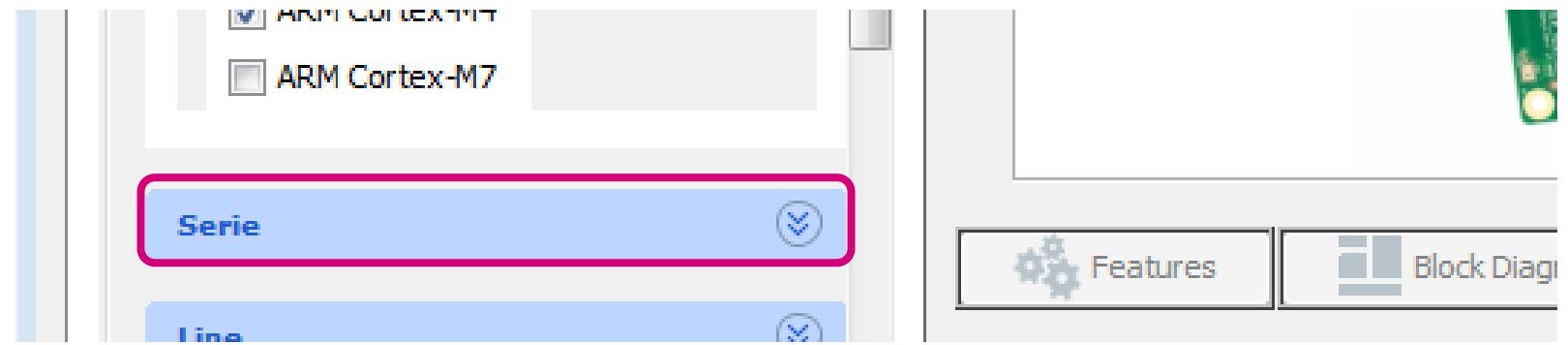
- Select **ARM Cortex-M4**



Select the Microcontroller

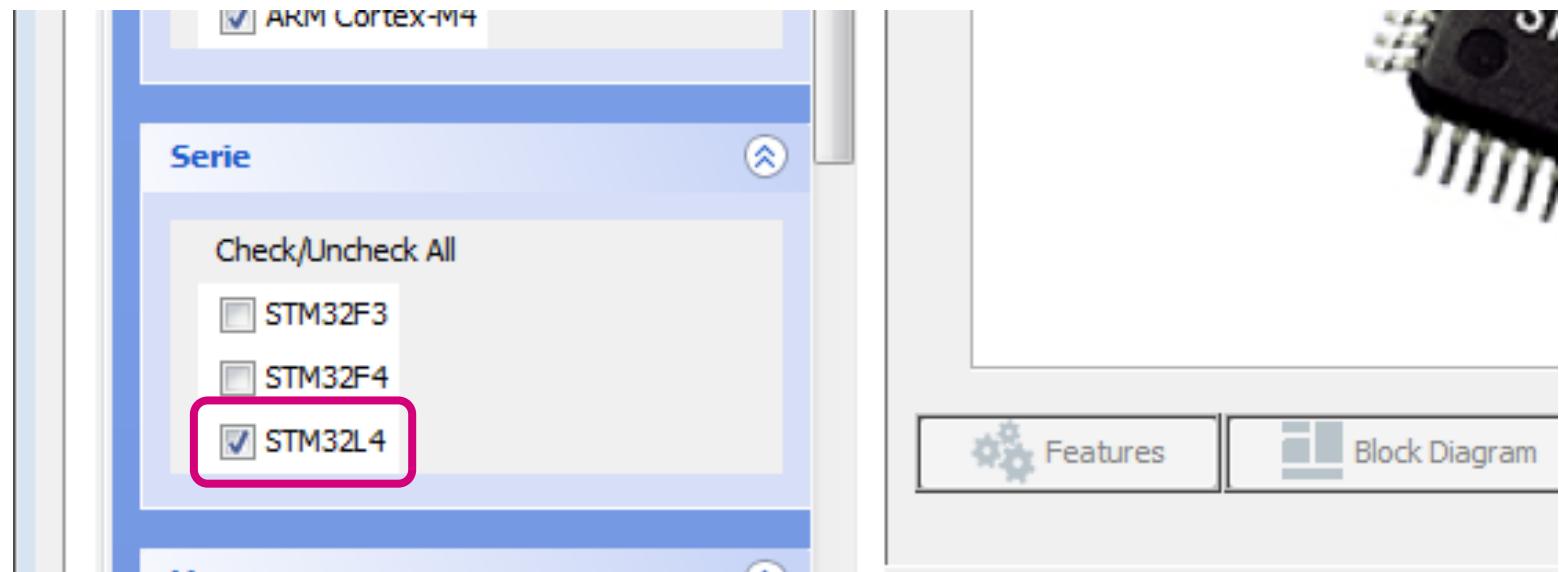
75

- Expand Series



Select the Microcontroller

- Select **STM32L4**



Select the Microcontroller

77

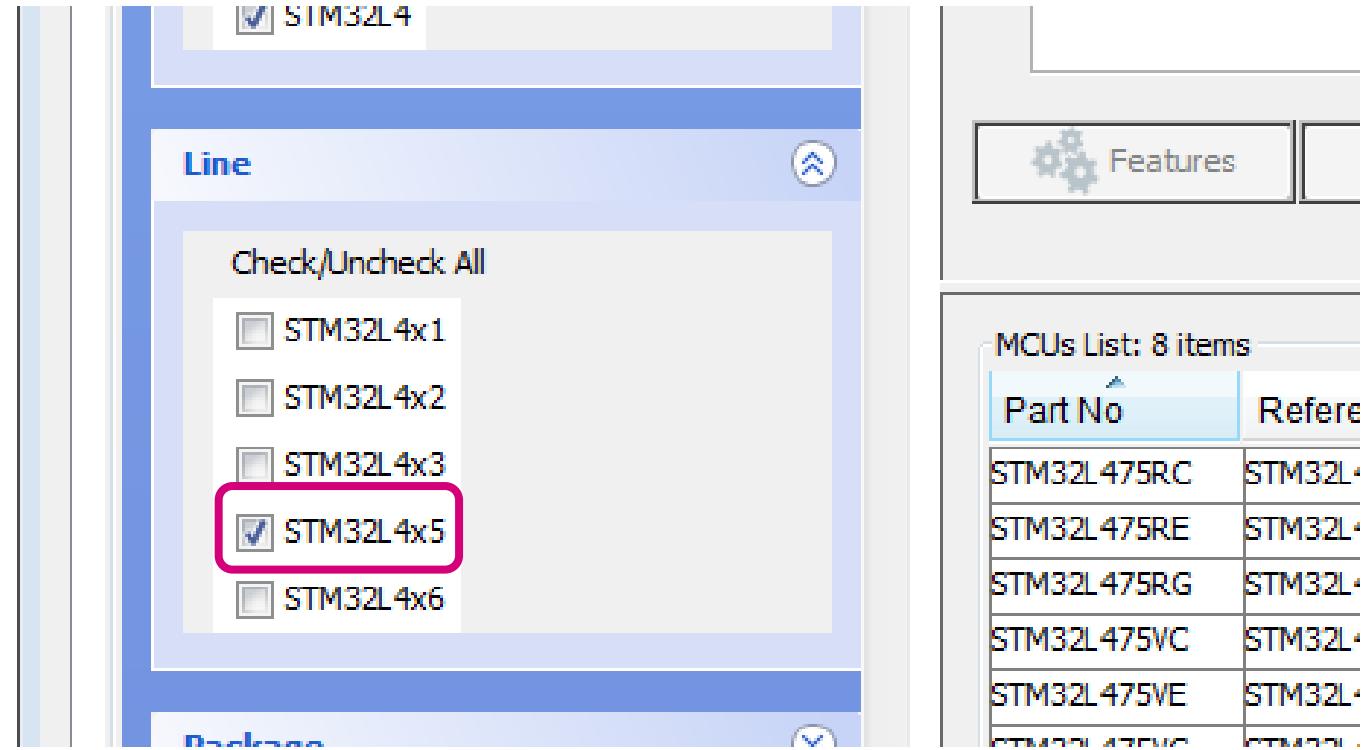
- Expand Line



Select the Microcontroller

78

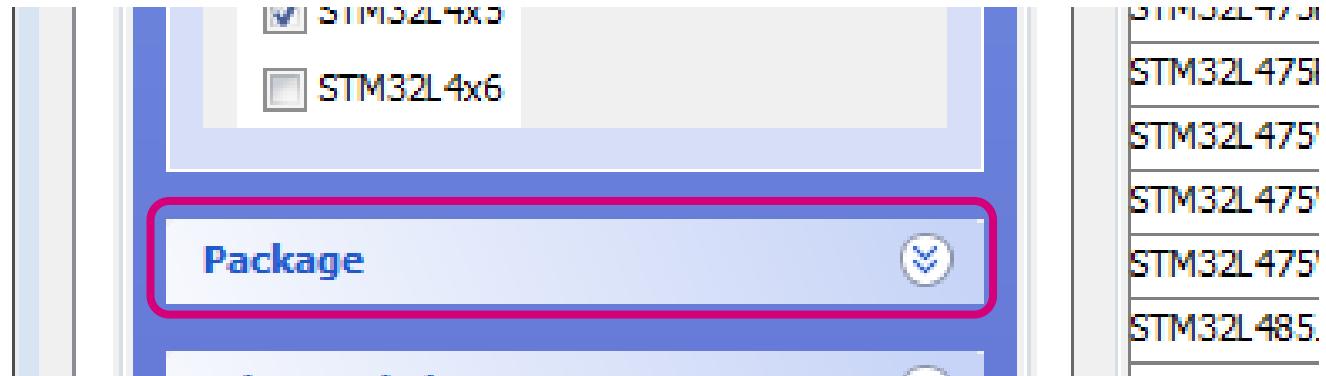
- Select **STM32L4x5**



Select the Microcontroller

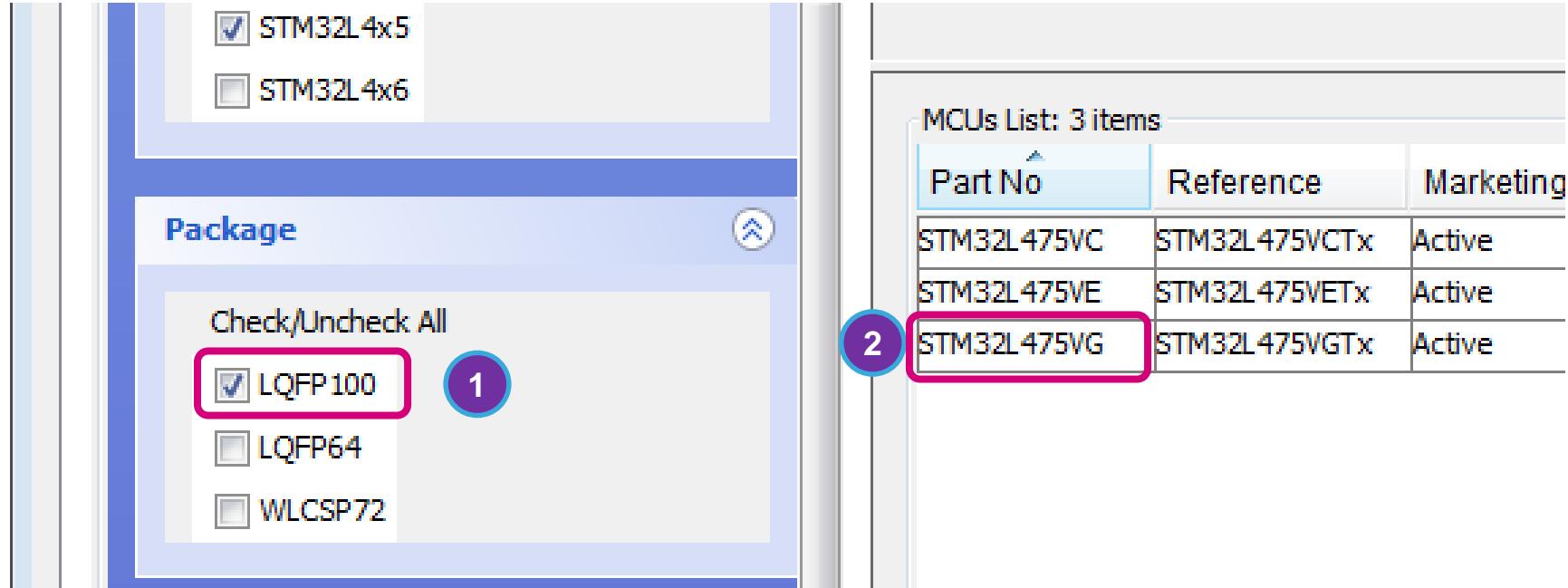
79

- Expand Package



Select the Microcontroller

80



1. Select **LQFP100**
2. Double-click on **STM32L475VG**

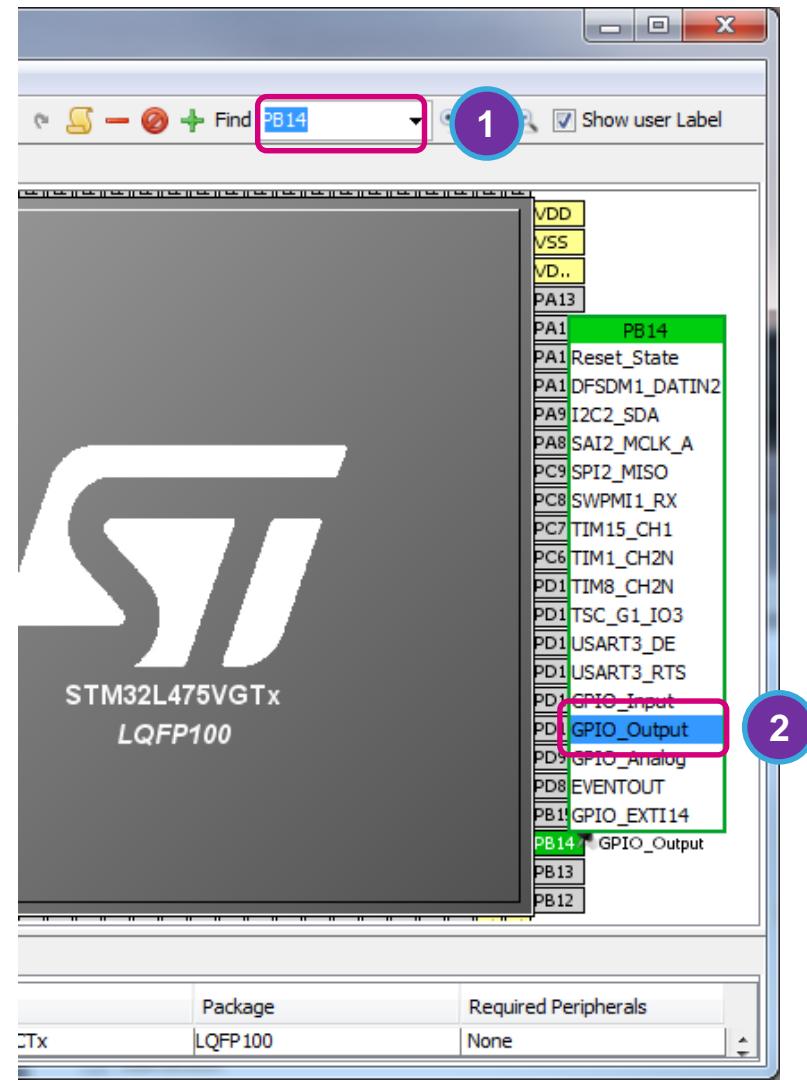
GPIO Selection

- This example will use **LED2** on the DK IoT board.



- Use the find toolbar and type **PB14**
- Select **PB14** and set it to **GPIO_Output** mode.

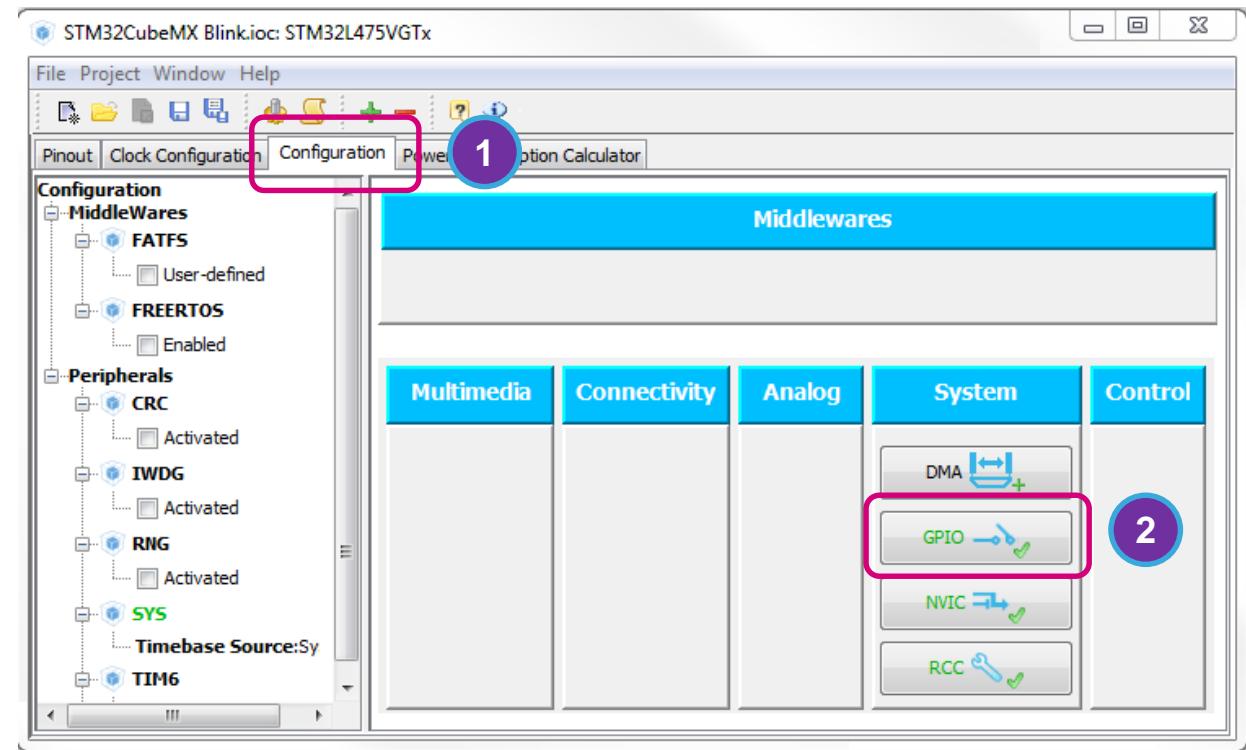
Note: Board schematics can be found in the [IoT board User Manual \(UM2153\)](#)



GPIO Configuration

82

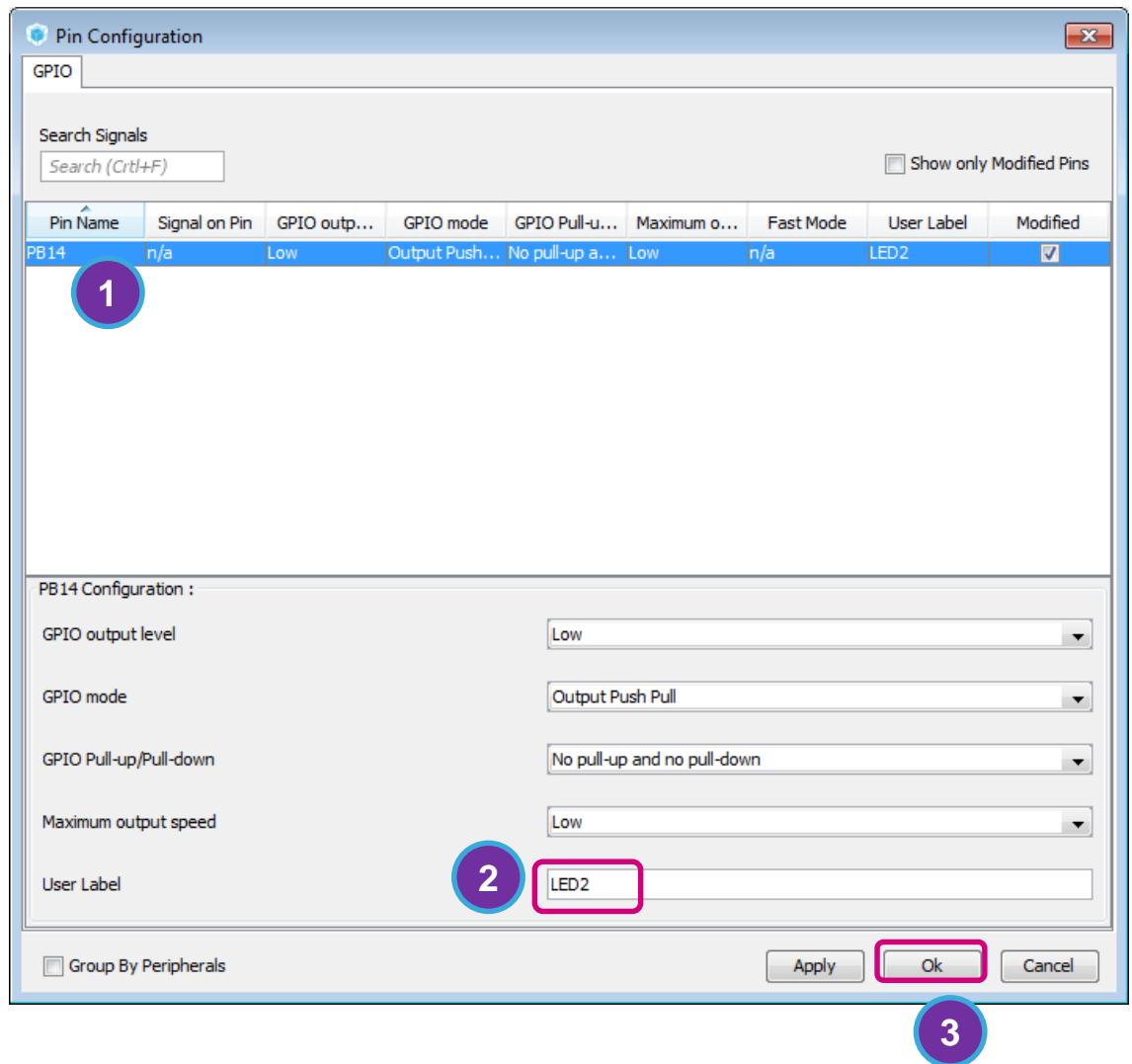
1. Select the **Configuration** tab
2. Select **GPIO** under System.



GPIO Configuration

83

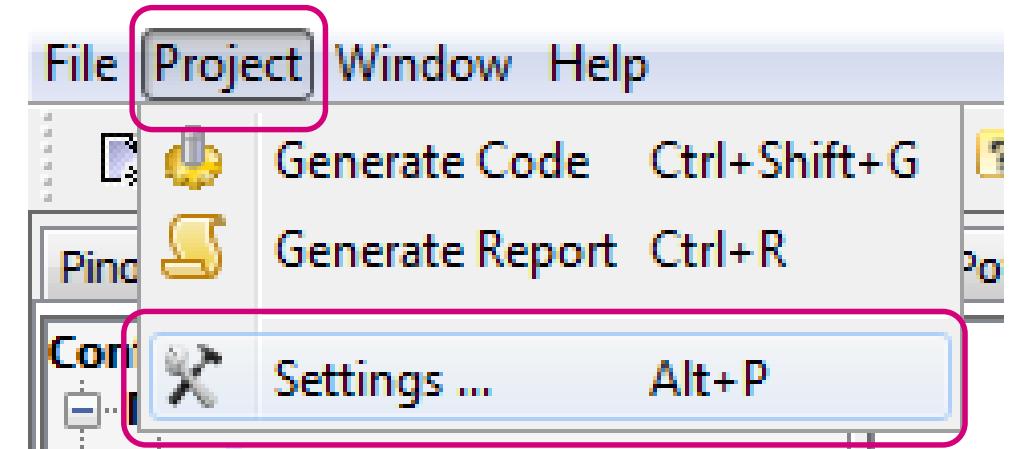
1. Select **PB14**.
2. Set the User Label to **LED2**.
3. Click **OK**



Project Settings

84

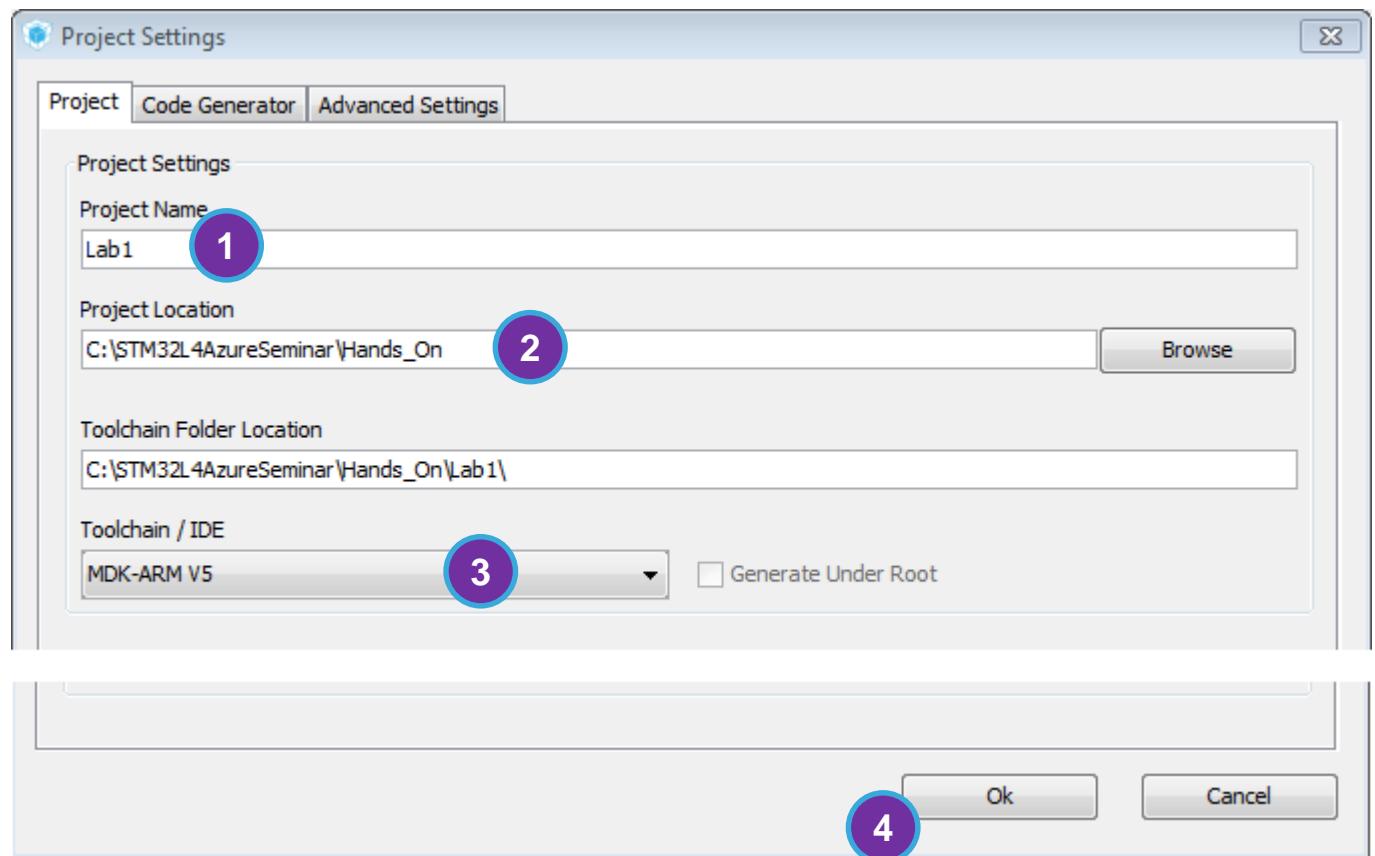
1. Select **Project -> Settings**
(Alt + P)



Project Settings

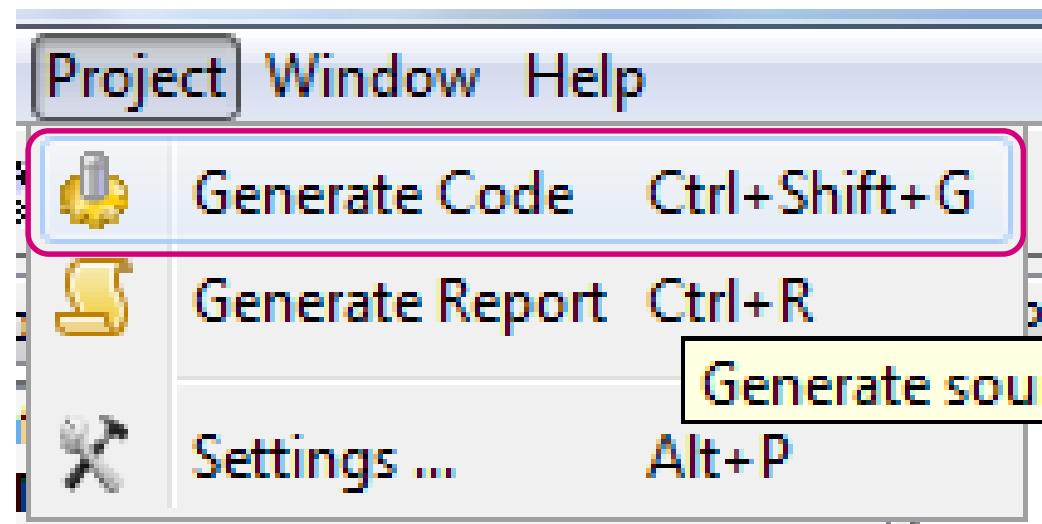
85

1. Set the project name to **Lab1**.
2. Set the project location:
C:\STM32L4AzureSeminar\Hands_On
3. Set the IDE Toolchain to **MDK-ARM V5**.
4. Click **OK**.



Generate the Project

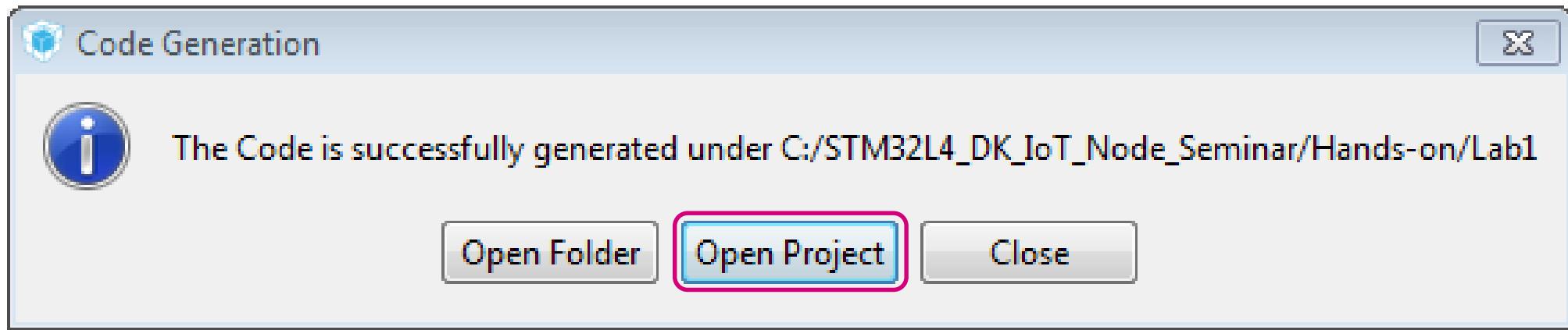
86



- Click **Generate Code** (Ctrl + Shift + G)

Open the Project

87



- Click Open Project

Inside Keil MDK-ARM

88

The screenshot shows the Keil MDK-ARM IDE interface. The Project Window on the left displays the project structure for 'Lab1'. The File Editor Window in the center shows the content of 'main.c'. The Build Output Window at the bottom shows the compilation process and build time.

Project Window: Shows the project structure for 'Lab1' with various source files and CMSIS components.

File Editor Window: Displays the code for 'main.c'. The code includes a detailed copyright notice, redistribution conditions, and disclaimer. It also includes sections for includes, user code begin/end, and private variables.

```
1  ****
2  **** File Name      : main.c
3  * Description     : Main program body
4  ****
5  ** This notice applies to any and all portions of this file
6  * that are not between comment pairs USER CODE BEGIN and
7  * USER CODE END. Other portions of this file, whether
8  * inserted by the user or by software development tools
9  * are owned by their respective copyright owners.
10 *
11 * COPYRIGHT(c) 2017 STMicroelectronics
12 *
13 * Redistribution and use in source and binary forms, with or without modification,
14 * are permitted provided that the following conditions are met:
15 *   1. Redistributions of source code must retain the above copyright notice,
16 *      this list of conditions and the following disclaimer.
17 *   2. Redistributions in binary form must reproduce the above copyright notice,
18 *      this list of conditions and the following disclaimer in the documentation
19 *      and/or other materials provided with the distribution.
20 *   3. Neither the name of STMicroelectronics nor the names of its contributors
21 *      may be used to endorse or promote products derived from this software
22 *      without specific prior written permission.
23 *
24 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
25 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
26 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
27 * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
28 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
29 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
30 * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
31 * CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
32 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
33 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
34 *
35 ****
36 */
37 /* Includes -----
38 #include "main.h"
39 #include "stm324xx_hal.h"
40 /**
41  * @brief Main program entry point
42  */
43 int main(void)
44 {
45     /* Initialize HAL library
46     */
47     /* USER CODE BEGIN PVE */
48     /* Private variables
49     */
50     /* USER CODE END PVE */
51 }
```

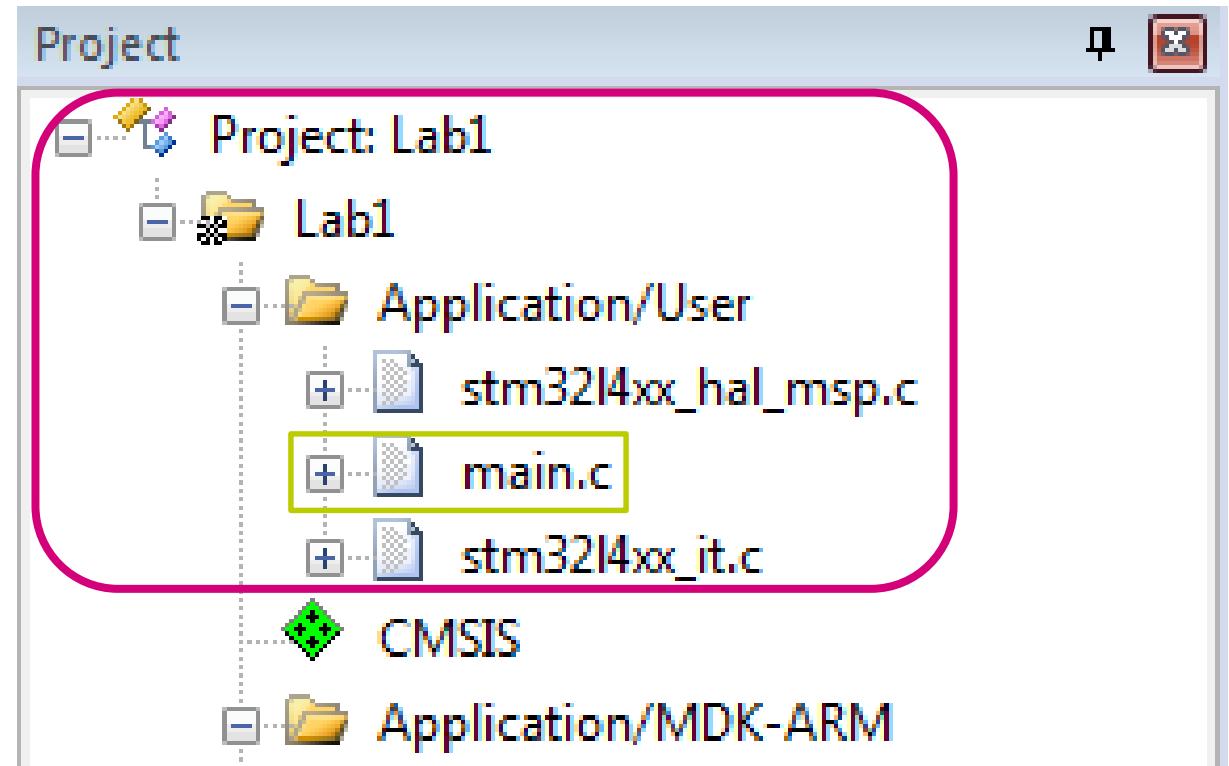
Build Output Window: Shows the compilation process and build time.

```
compiling stm324xx_hal.c...
compiling stm324xx_hal_pwr_ex.c...
compiling system_stm324xx.c...
linking...
Program Size: Code=3244 RO-data=488 RW-data=8 ZI-data=1024
"Lab1\Lab1.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:01:15
```

Edit main.c

89

- Expand the file tree:
 1. Expand **Lab1**
 2. Expand **Application/User**
 3. Double-click **main.c**



Edit main.c

90

```
main.c*
96     /* Infinite loop */
97     /* USER CODE BEGIN WHILE */
98     while (1)
99     {
100         HAL_Delay(100);
101         HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
102
103     /* USER CODE END WHILE */
104
105     /* USER CODE BEGIN 3 */
106
107 }
108 /* USER CODE END 3 */
109
110 }
```

Add the following code inside the **while(1)** loop (line 100):

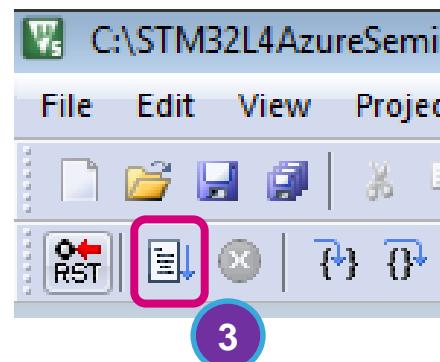
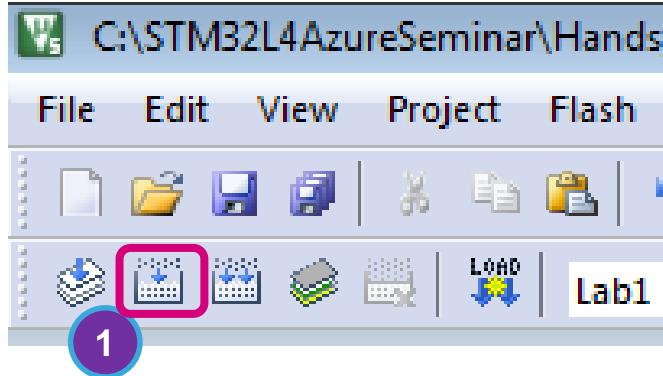
HAL_Delay(100);

HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);

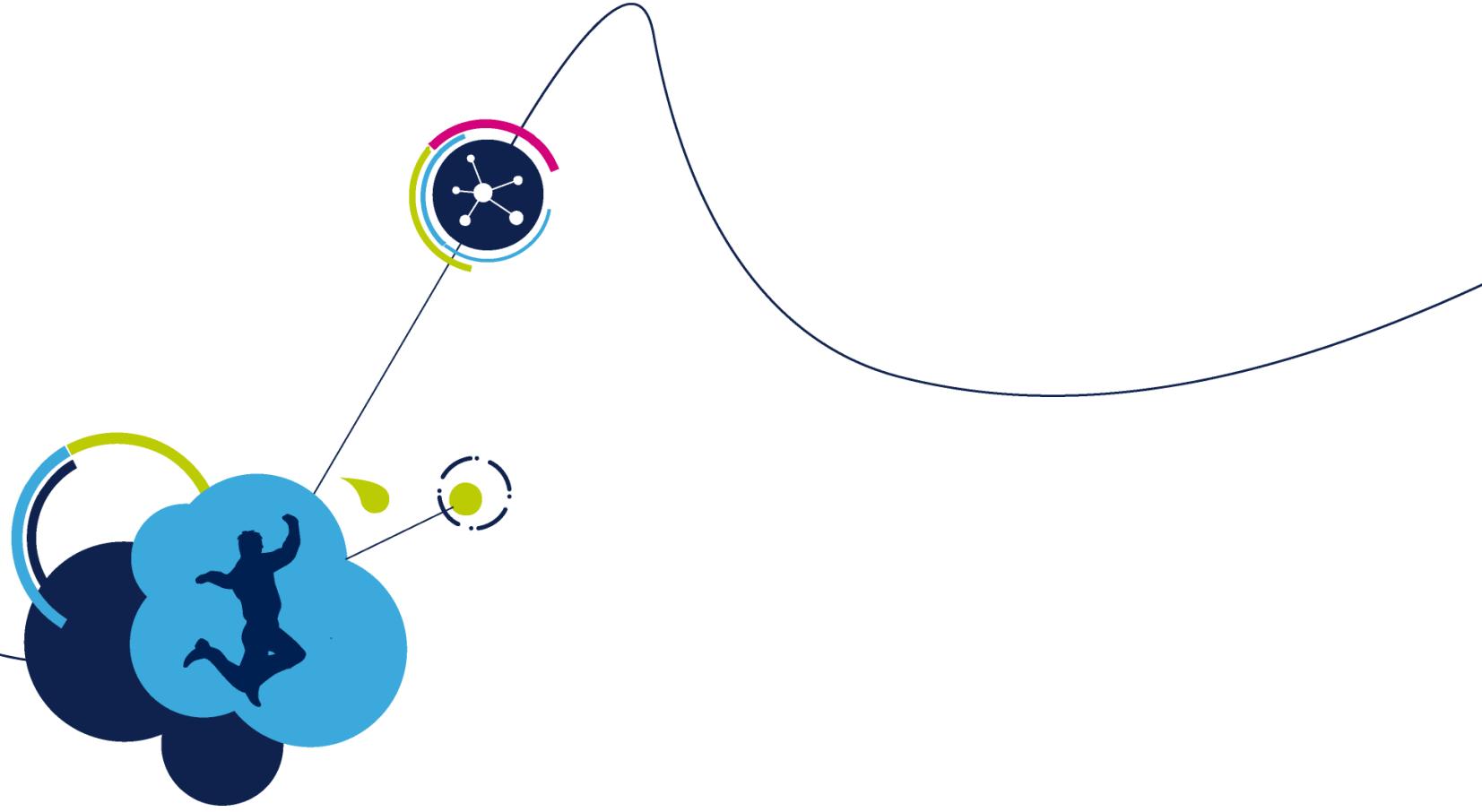
Load and Run

91

1. Click on the Build button or choose Project > Build Target
2. Click the Start/Stop Debug Session button
3. Click the Run button



The board LED should now start blinking!



Microsoft Azure IoT Overview

What is Microsoft Azure IoT?

93

- Microsoft Azure Internet of Things (IoT) services enable secure, bidirectional communication between IoT devices and the cloud.

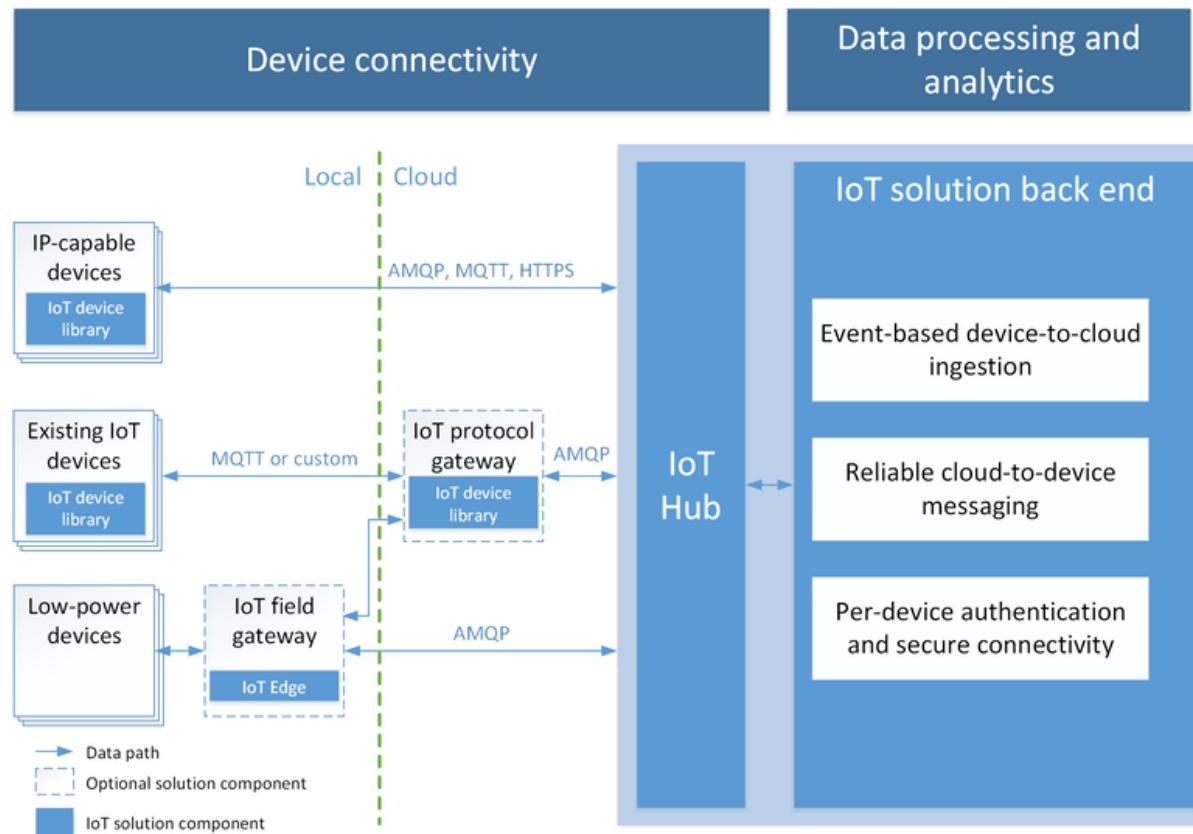


Image Source: Microsoft

What Is JSON

94

- JSON (JavaScript Object Notation) is an open standard lightweight data-interchange format. As a text document, it is easy for users to read and write, and for machines to parse and generate.
- JSON is completely language independent, but uses conventions that are familiar to C-family programmers including C, C++, C#, Java, JavaScript, Perl, Python and many others.

```
{  
    "state": {  
        "desired": {  
            "LED_value": "On"  
        }  
    }  
}
```

What is MQTT

95

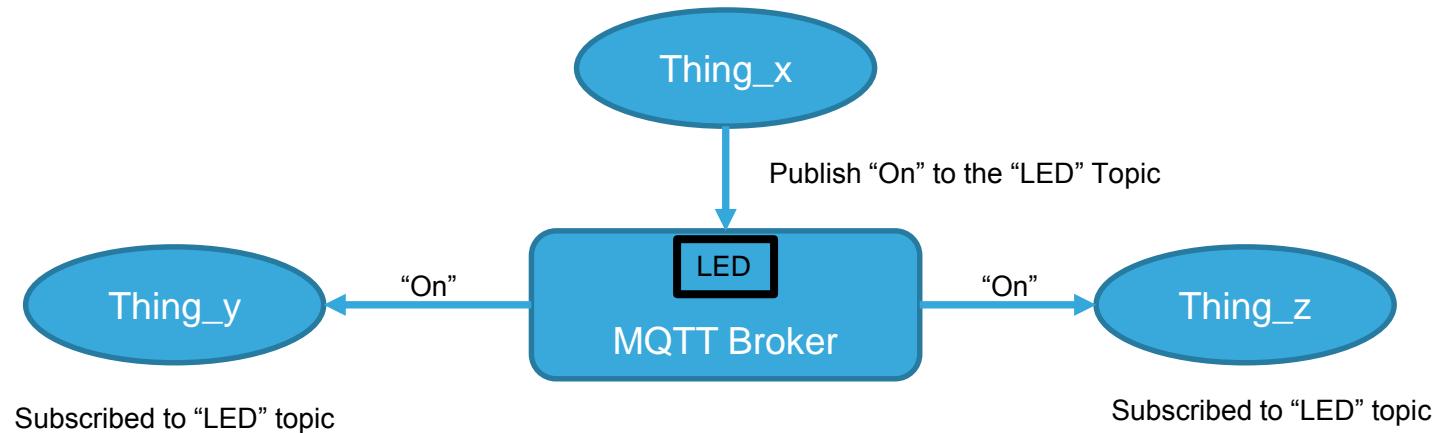
- **MQTT** stands for **MQ Telemetry Transport**. It is an extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks.
- Designers use **MQTT** to minimize network bandwidth and device resource requirements, while ensuring reliability and some degree of delivery assurance. These principles also turn out to make the protocol ideal in the emerging “machine-to-machine” (M2M) or “Internet of Things” world of connected devices, and for mobile applications where bandwidth and battery power are at a premium.

Source: <http://mqtt.org/>

How MQTT Works

96

- MQTT uses a client/server model where every IoT device is a client and is connected to a server, called an MQTT broker (example: Azure IoT Hub).
- The clients send messages to an address, called a topic. The MQTT broker will forward that message to all the clients subscribed to that topic.



Azure IoT Device Twin

97

- A Device Twin is a JSON document that is used to store and retrieve current state information for a IoT node.
- For each new device, the IoT Hub creates a Device Twin and uses it to maintain a persistent state of the device regardless of whether the device is connected to the Internet or not.
- The Device Twin JSON document includes:
 - Tags
 - Desired Properties
 - Reported Properties

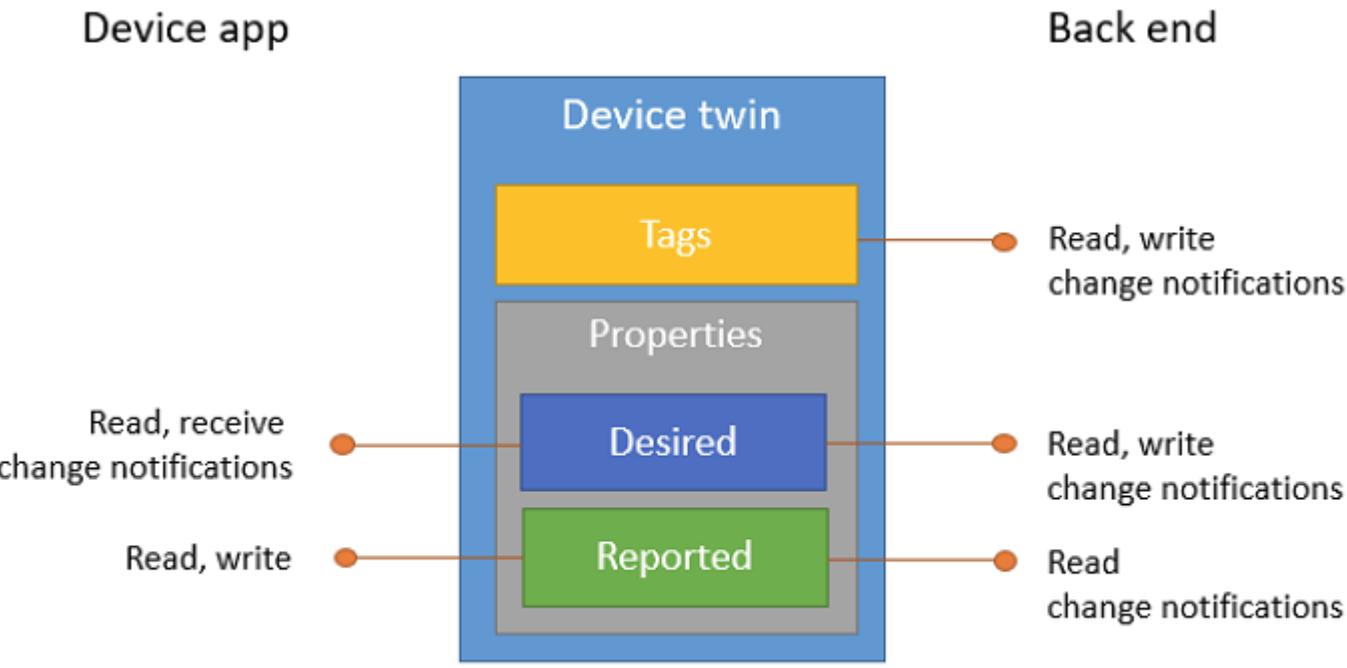


Image Source: Microsoft

Device Management

98

- IoT Hub enabled a set of device management patterns that can be adapted or extended to fit your needs
 - Reboot
 - Factory reset
 - Configuration
 - Firmware update
 - Report progress and status

Configuration

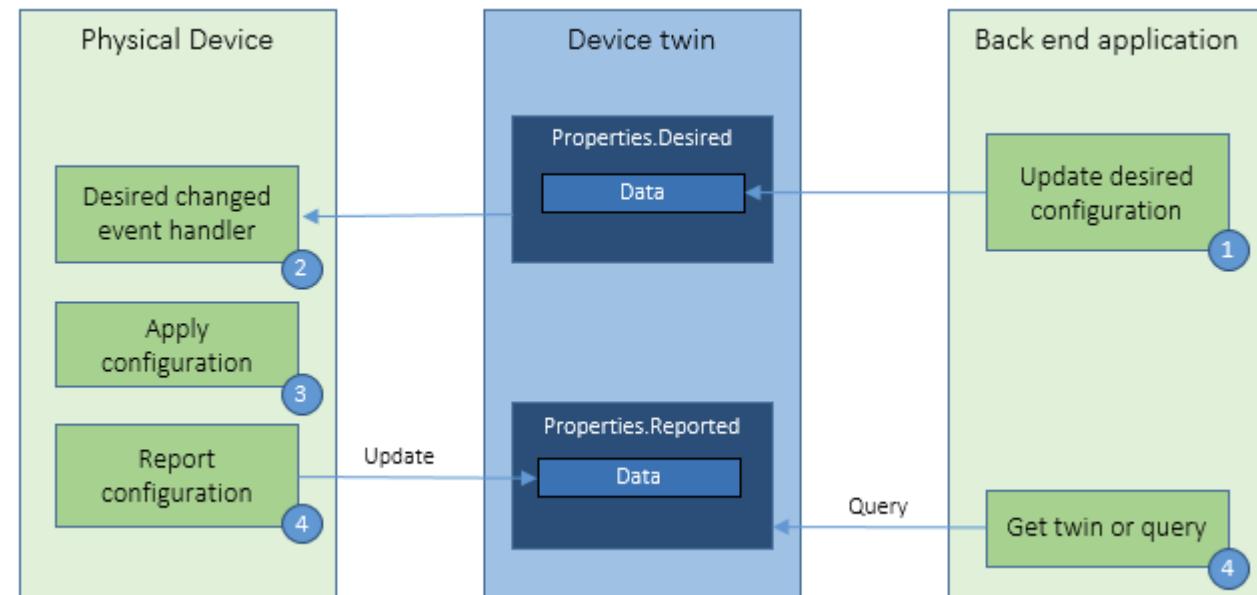


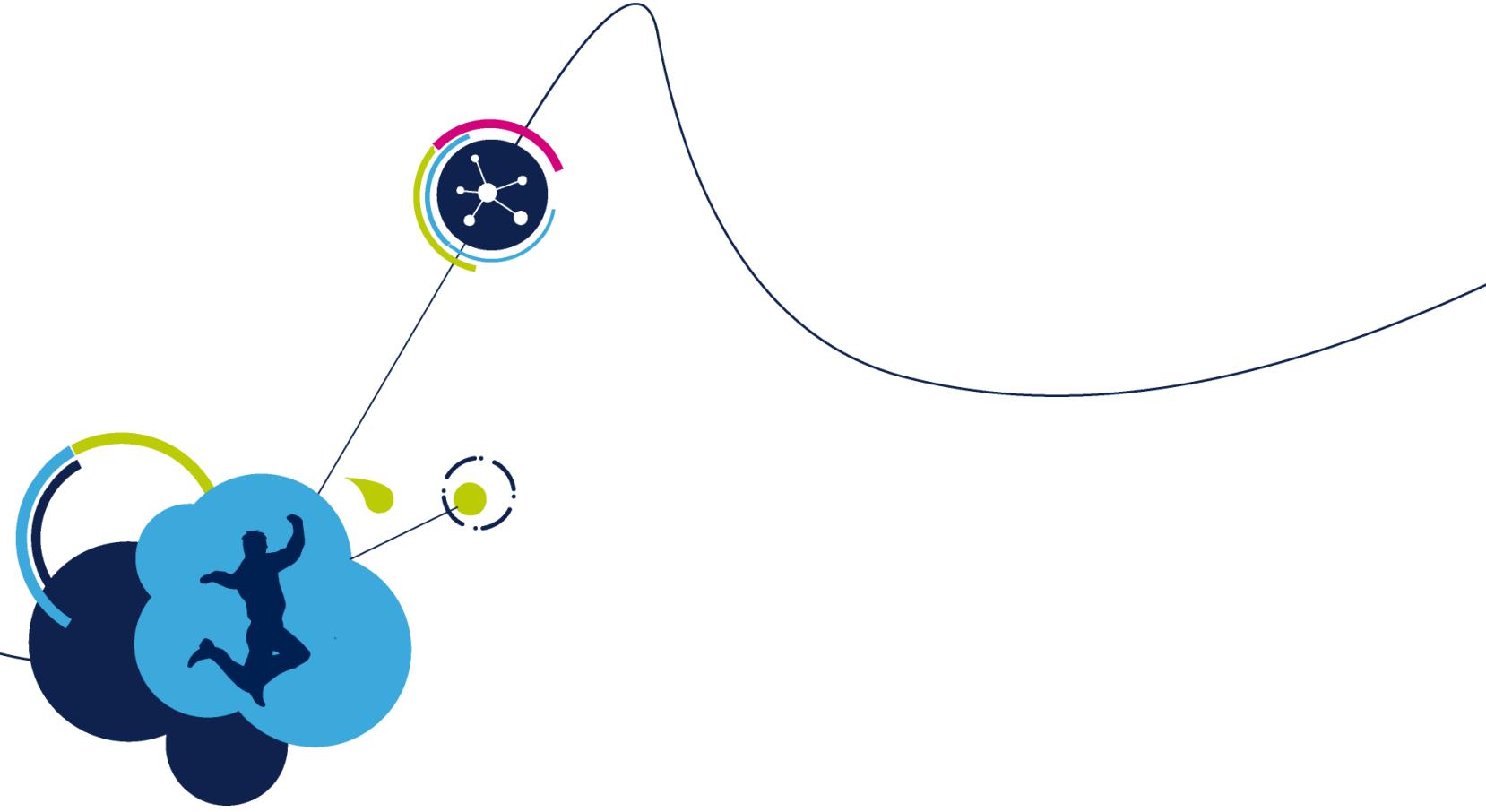
Image Source: Microsoft

The pattern shown here is for Configuration: the back-end app uses the desired properties to configure software running on the device. The device uses reported properties to update the configuration status of the device.

Azure IoT Security

99

- IoT Hub uses security tokens to authenticate devices and services
- Devices and services use a shared access or symmetric key to sign and send security tokens to the IoT Hub to get access
- IoT Hub also allows using X.509 certificates to authenticate a device



Lab 2: Getting Started with Microsoft Azure IoT in 5 minutes!

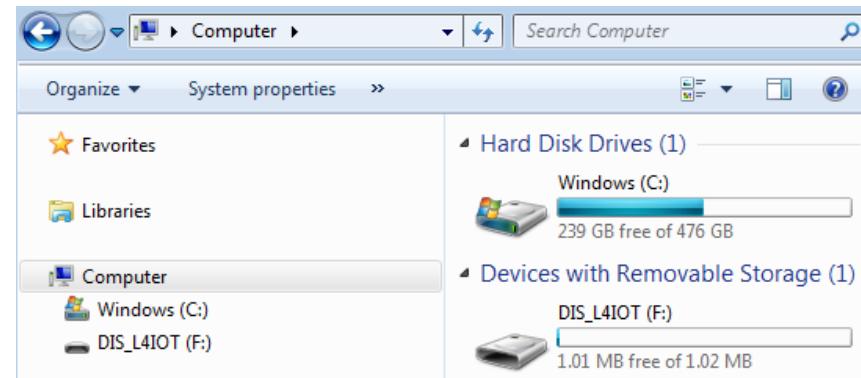
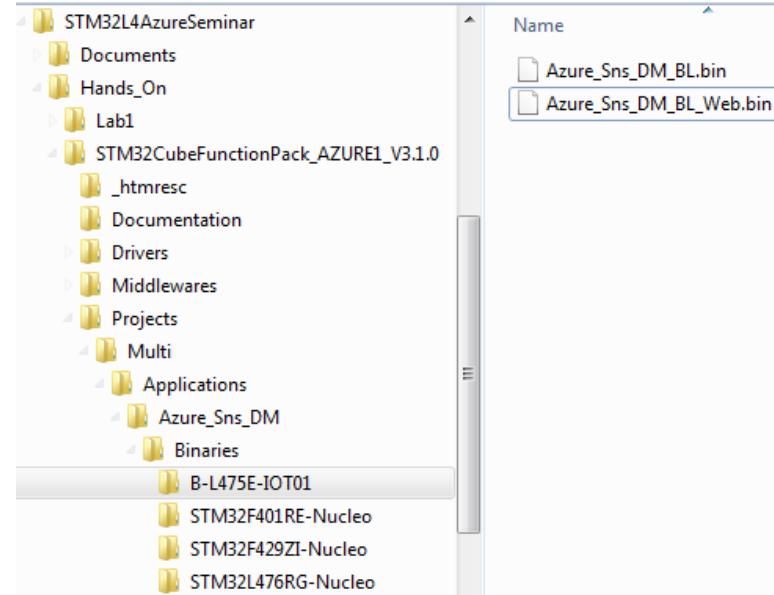
- In this lab, we will:
 - Program the STM32L4 Discovery Kit IoT node with precompiled binaries
 - Input WiFi credentials through a terminal program
 - Establish connection to Azure IoT
 - View sensor data from your node on a web application
 - Explore various Azure features available through the web application

Program your board

102

- In Windows Explorer, navigate to the following folder:

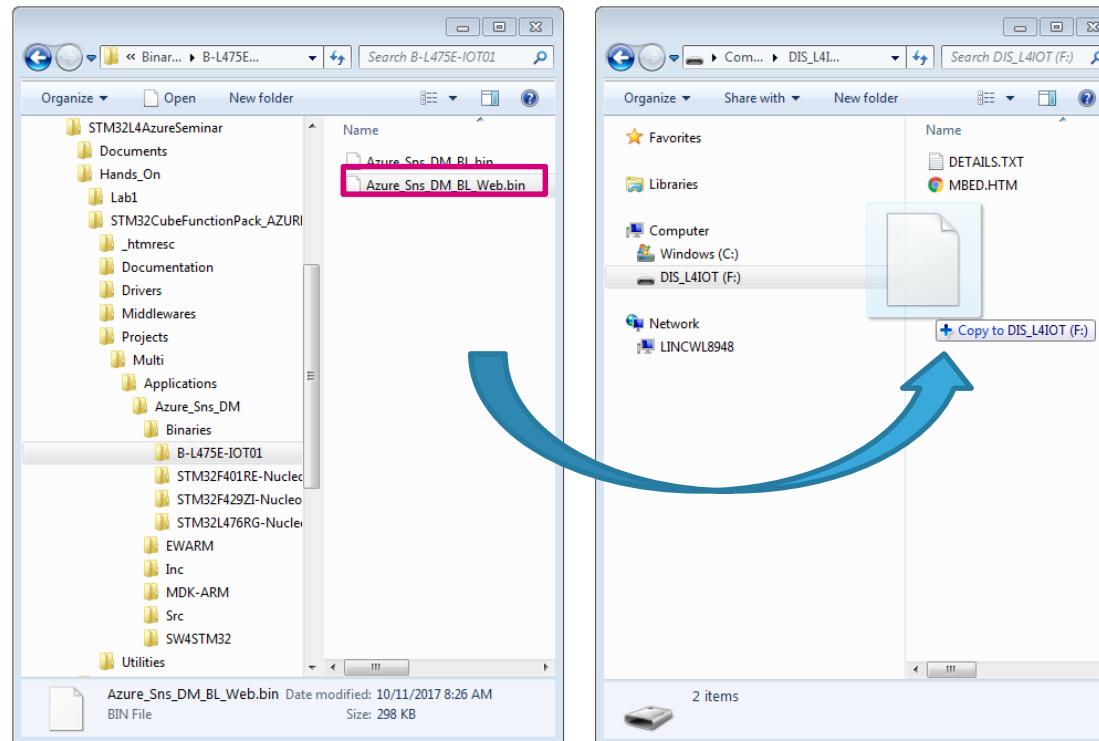
C:\STM32L4AzureSeminar\Hands_On\STM32CubeFunctionPack_AZURE1_V3.1.0\Projects\Multi\Applications\Azure_Sns_DM\Binaries\B-L475E-IOT01\



Program your board

103

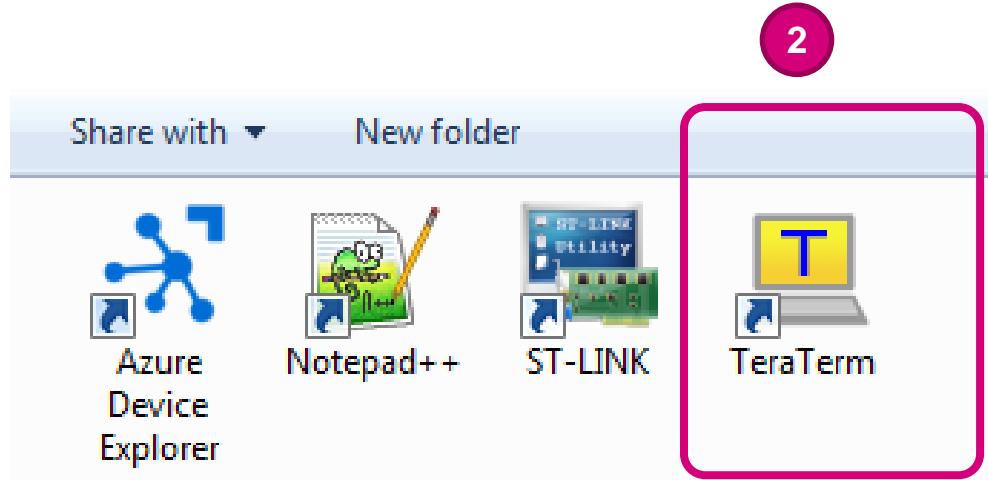
- We will use simple drag-and-drop programming that is enabled by the enhanced ST-LINK/V2-1 debugger built into the STM32L4 Discovery Kit
- Drag and drop the “**Azure_Sns_DM_BL_Web.bin**” file to the Discovery kit “DIS_L4IOT” drive
- Programming is complete when the LD6 LED stops flashing and the “DIS_L4IOT” drive disappears and reappears.



Note: If you prefer, you can also copy and paste the file to the target drive

Open and Configure Serial Terminal

104

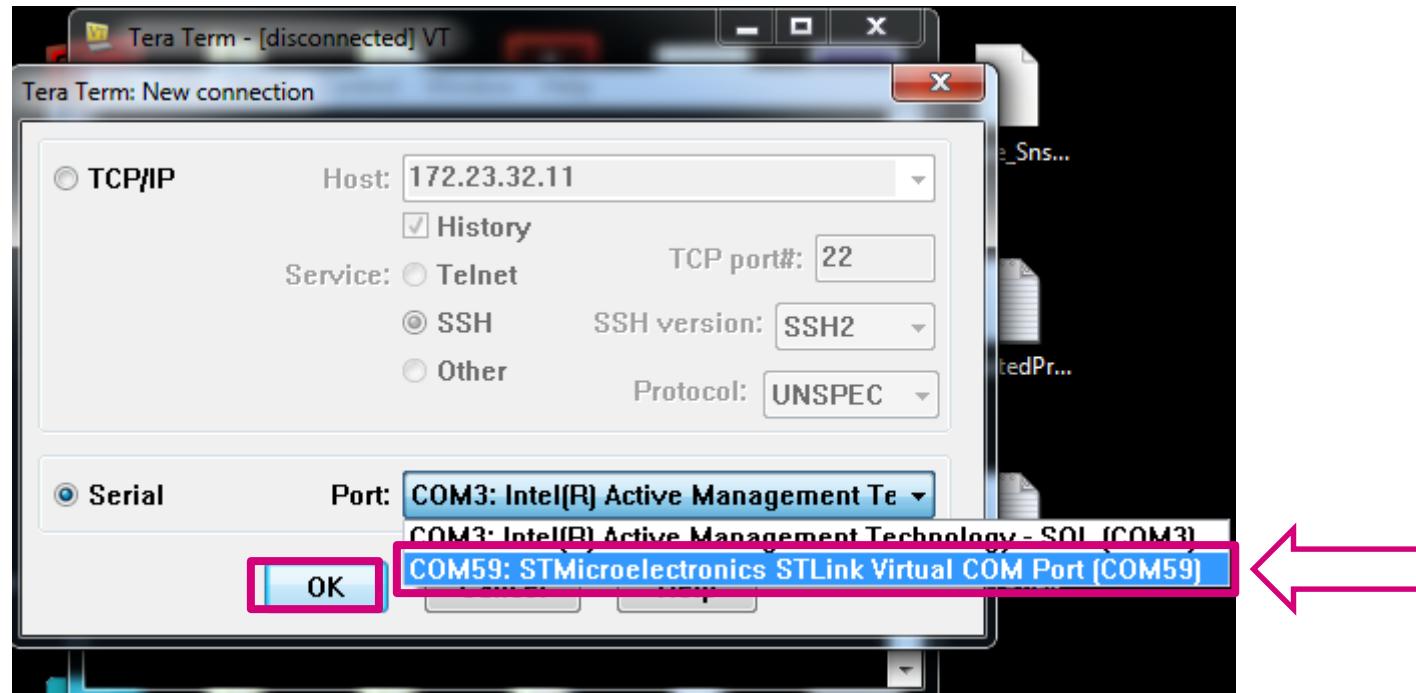


- 1 Open the STM32L4AzureSeminar folder on your Desktop.
- 2 Double-click on the **TeraTerm** shortcut

Open and Configure Serial Terminal (Cont'd)

105

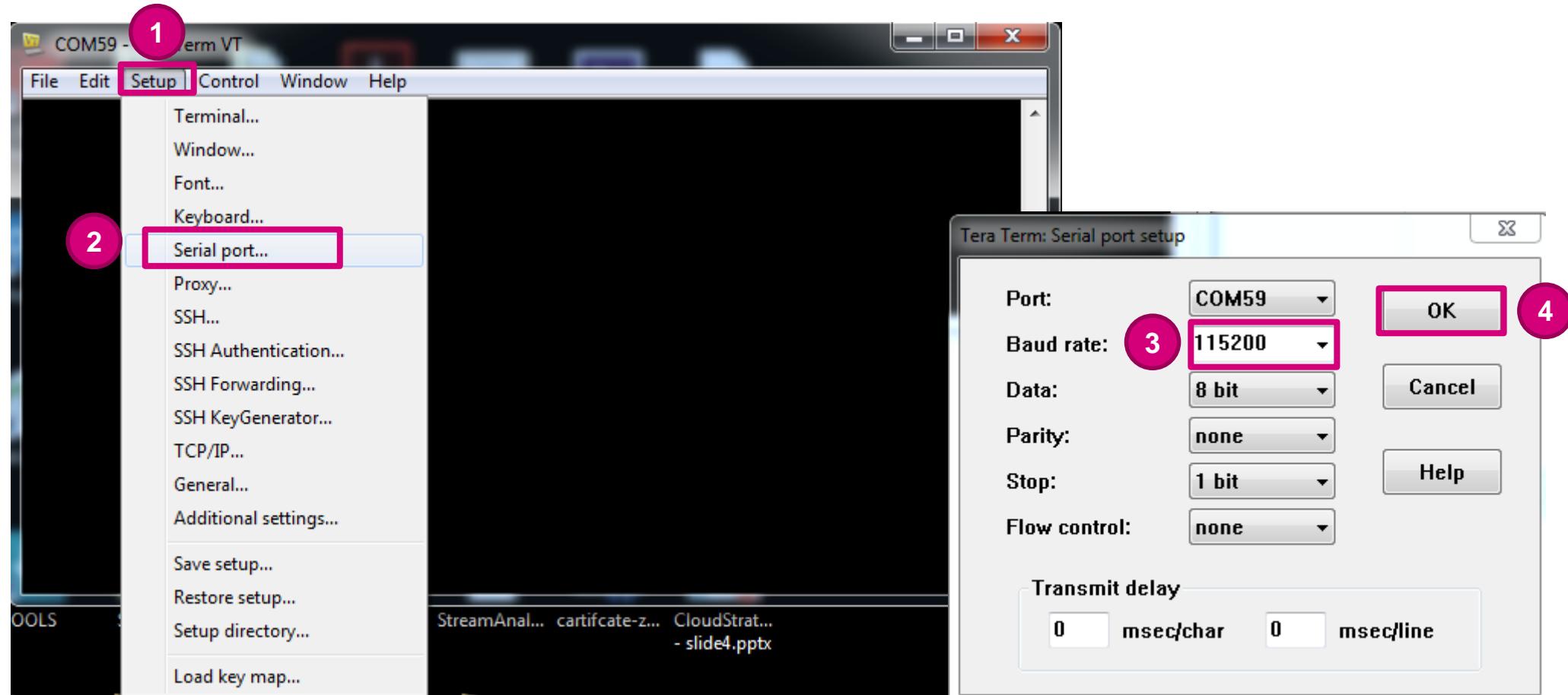
- 1 Select the STMicroelectronics STLink Virtual COM Port
- 2 Click OK



Open and Configure Serial Terminal

106

- Set baud rate speed in serial terminal to 115200 (**Setup → Serial port**)

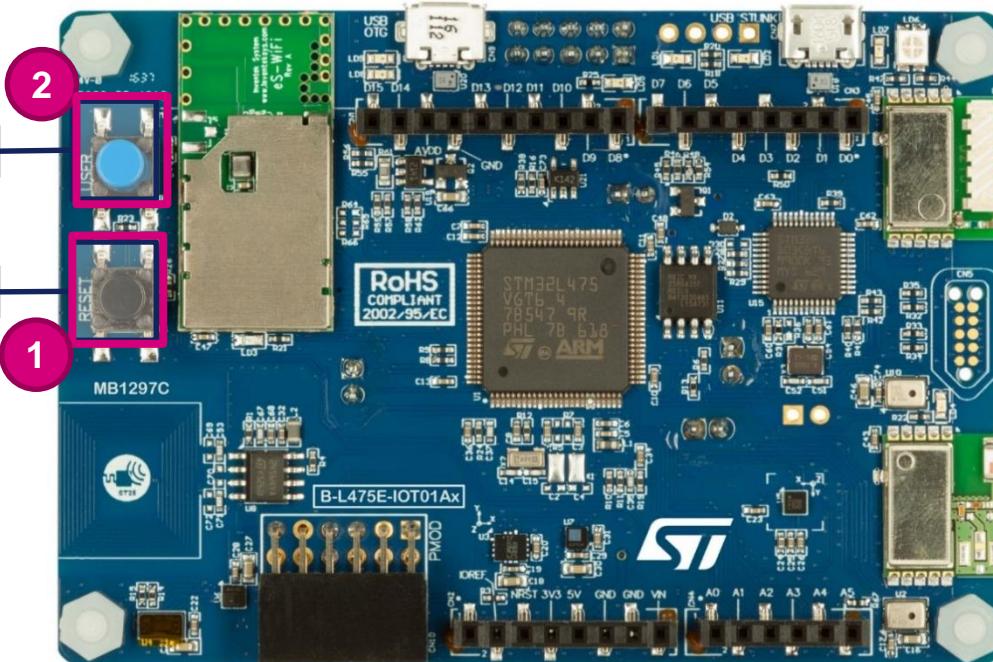


Note: If you see any Japanese characters, go to Setup > General and change language to “UTF-8”

Reset the Board

107

- Press the **black** Reset button
- Then, within 3 seconds, press the **blue** User button



```
COM3 - Tera Term VT
File Edit Setup Control Window Help
UART Initialized
STMicroelectronics Azure_Sns_DM:
Version 3.1.0
B-L475E-IOT01 IoT node Discovery board
Azure SDK Version 1.1.16
Ok Accelero Sensor
Ok Gyroscope Sensor
Ok Magneto Sensor
Ok Humidity Sensor
Ok Temperature Sensor
Ok Pressure Sensor
(HAL 1.7.1_0)
Compiled Oct 11 2017 14:19:07 (IAR)
OTA with one HTTP HEAD + Multiple HTTP one GET of 1024Bytes
Testing BootLoaderCompliance:
Version 1.2.0
BL Version Ok
MagicNum Ok
MaxSize 0x7c000
OTAStrtAdd Ok
BootLoader Compliant with FOTA procedure
Init Application's Timers
Init Random Number Generator
! WIFI Credential !
Meta Data Manager read from Flash
Meta Data Manager version=0.8.0
Generic Meta Data found:
    WIFI Size=81 [bytes]
Default SSID : STM
Default PassWd : STMdemoPWD
Default EncMode: WPA2/WPA2-Personal
Wait 3 seconds for allowing User Button Control for changing it
Do you want to change them?(y/n)
■
```

Change the default WiFi Credentials

108

- Type your **responses** to **questions** as follows:
 - Press Enter after each response
1. Do you want to change them?(y/n)?

y

2. Do you want to read them from NFC?(y/n)

n

3. Enter the SSID:

stm32iot

4. Enter the PassWd:

stm32iot

5. Enter the encryption mode (0:open, 1:WEP, 2:WPA/WPA2-Personal):

2

The screenshot shows a terminal window titled 'COM3 - Tera Term VT'. The window displays a series of commands and responses related to changing WiFi credentials. A red box highlights the user input area where responses are typed.

```
v COM3 - Tera Term VT
File Edit Setup Control Window Help
BootLoader Compliant with FOTA procedure
Init Application's Timers
Init Random Number Generator
! WIFI Credential !
Meta Data Manager read from Flash
Meta Data Manager version=0.8.0
Generic Meta Data found:
    WIFI Size=81 [bytes]
    Default SSID : STM
    Default PassWd : STMdemoPWD
    Default EncMode: WPA2/WPA2-Personal
Wait 3 seconds for allowing User Button Control for changing it
Do you want to change them?(y/n)
y      Do you want to read them from NFC?(y/n)
n
Enter the SSID:
stm32iot
Enter the PassWd:
stm32iot
Enter the encryption mode<0:Open, 1:WEP, 2:WPA2/WPA2-Personal>:
2
New SSID : stm32iot
New PassWd : stm32iot
New EncMode: WPA2/WPA2-Personal
Updating the Generic Meta Data type=WIFI

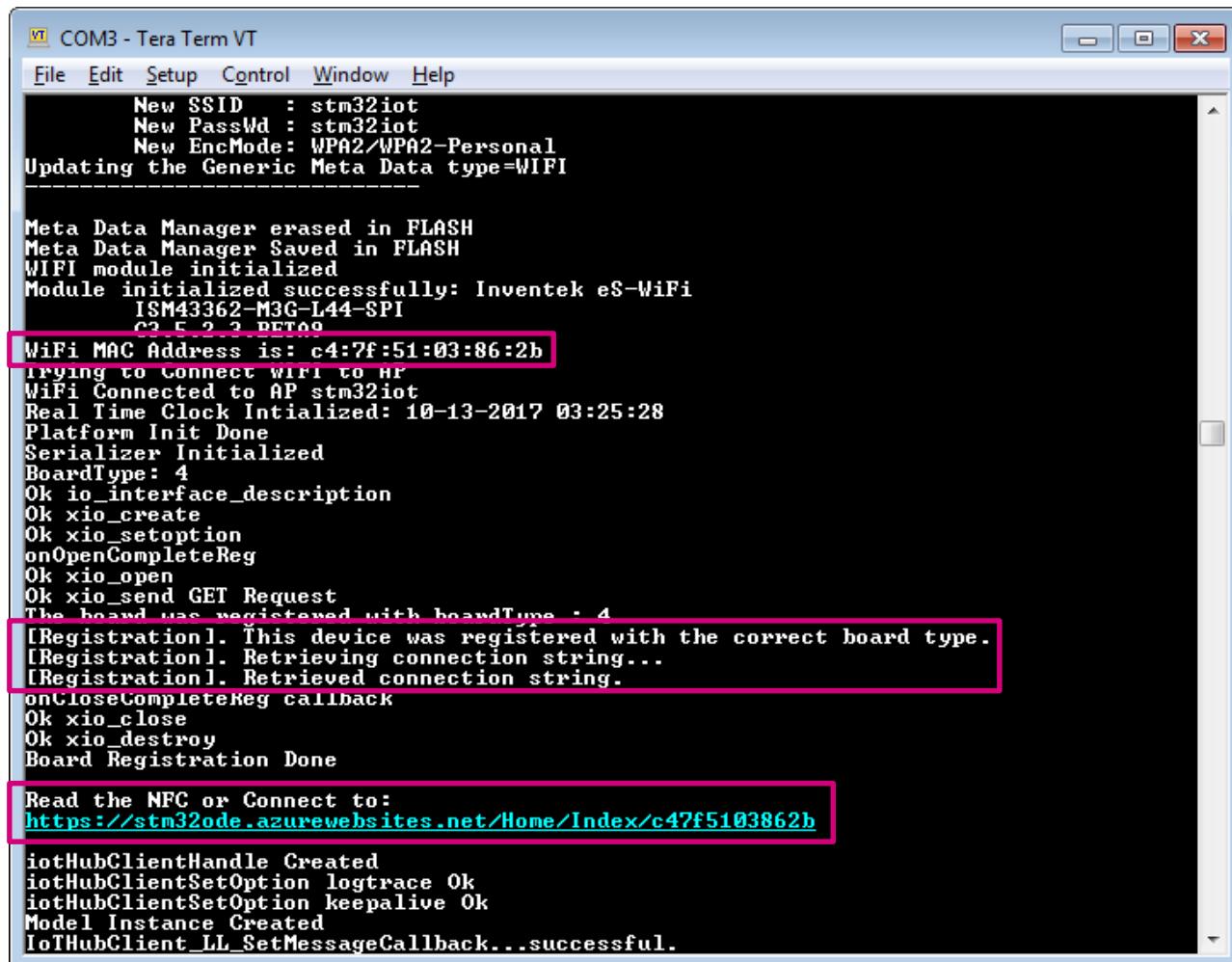
Meta Data Manager erased in FLASH
Meta Data Manager Saved in FLASH
WIFI module initialized
Module initialized successfully: Inventek eS-WiFi
ISM43362-M3G-L44-SPI
C3.5.2.3.BETA9
WiFi MAC Address is: c4:7f:51:03:86:2b
Trying to Connect WIFI to AP
WiFi Connected to AP stm32iot
Real Time Clock Initialized: 10-13-2017 03:25:28
Platform Init Done
Serializer Initialized
BoardType: 4
Ok io_interface_description
Ok xio_create
Ok xio_setopt
```

Note: You cannot use backspace or delete if you make a typo;
in that case, press reset, then blue button and start over.

Auto Registration to Azure IoT Hub

109

- The WiFi credentials are stored in MCU Flash memory for future use
- Board is automatically registered to ST's demonstration Azure IoT Hub
 - The WiFi module's MAC address is used as the unique identifier
 - The unique URL displayed also uses the same unique identifier



The screenshot shows a terminal window titled "COM3 - Tera Term VT". The terminal displays the following log output:

```
New SSID : stm32iot
New PassWd : stm32iot
New EncMode: WPA2/WPA2-Personal
Updating the Generic Meta Data type=WIFI

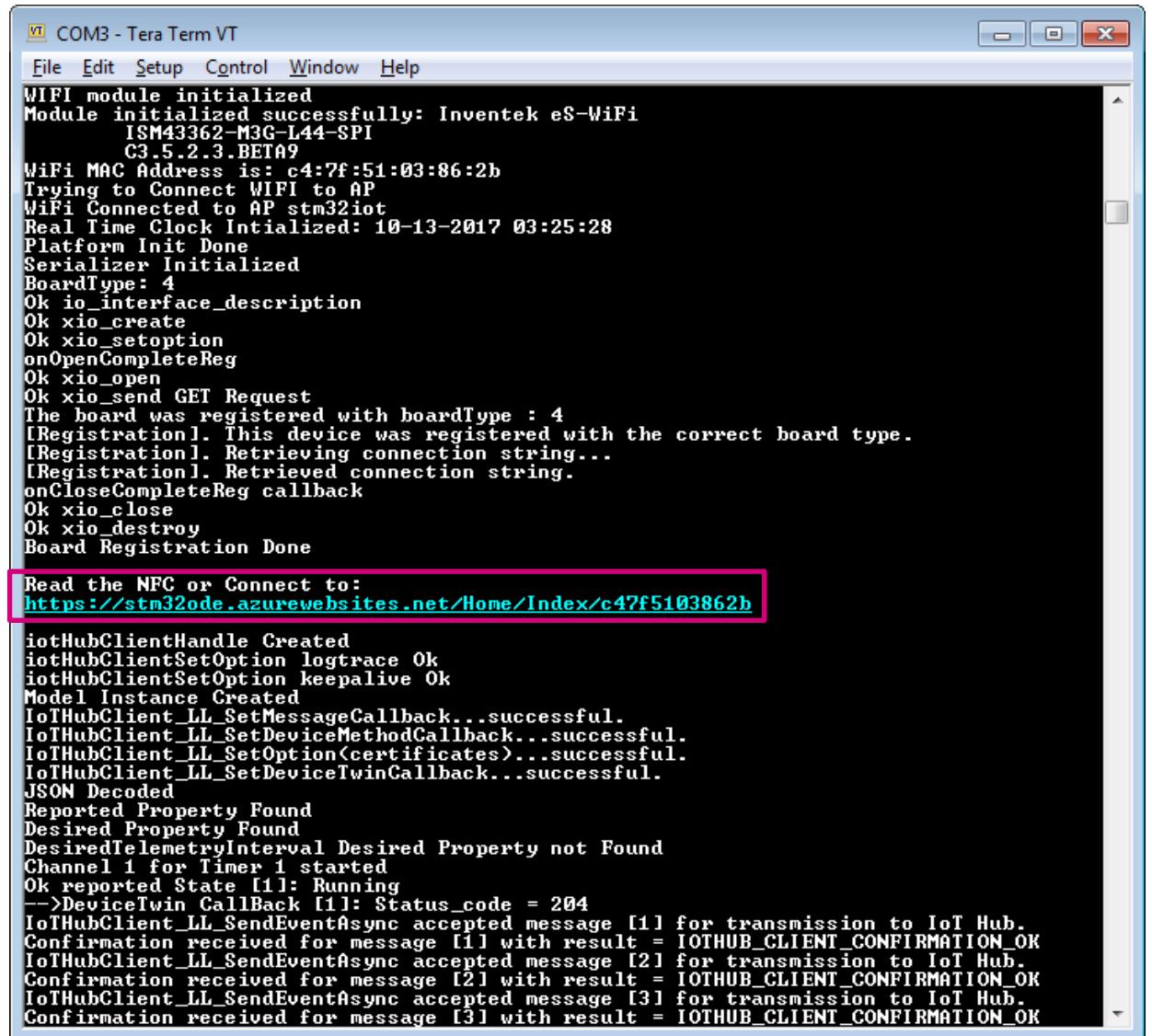
Meta Data Manager erased in FLASH
Meta Data Manager Saved in FLASH
WIFI module initialized
Module initialized successfully: Inventek eS-WiFi
ISM43362-M3G-L44-SPI
C2 F 2 3 PFT09
WiFi MAC Address is: c4:7f:51:03:86:2b
Trying to Connect WiFi to HP
WiFi Connected to AP stm32iot
Real Time Clock Initialized: 10-13-2017 03:25:28
Platform Init Done
Serializer Initialized
BoardType: 4
Ok xio_interface_description
Ok xio_create
Ok xio_setoption
onOpenCompleteReg
Ok xio_open
Ok xio_send GET Request
The board was registered with boardType = 4
[Registration]. This device was registered with the correct board type.
[Registration]. Retrieving connection string...
[Registration]. Retrieved connection string.
onCloseCompleteReg callback
Ok xio_close
Ok xio_destroy
Board Registration Done

Read the NFC or Connect to:
https://stm32ode.azurewebsites.net/Home/Index/c47f5103862b

iotHubClientHandle Created
iotHubClientSetOption logtrace Ok
iotHubClientSetOption keepalive Ok
Model Instance Created
IoTHubClient_LL_SetMessageCallback...successful.
```

Open the Web Application

- Multiple ways to open the web app:
 - Double click on the URL in the TeraTerm terminal window to lauch your browser
 - Use your Android phone to touch the NFC on the board to launch the URL in your browser



```
COM3 - Tera Term VT
File Edit Setup Control Window Help
WIFI module initialized
Module initialized successfully: Inventek eS-WiFi
  ISM43362-M3G-L44-SPI
  C3.5.2.3.BETA9
WiFi MAC Address is: c4:7f:51:03:86:2b
Trying to Connect WIFI to AP
WiFi Connected to AP stm32iot
Real Time Clock Initialized: 10-13-2017 03:25:28
Platform Init Done
Serializer Initialized
BoardType: 4
Ok xio_interface_description
Ok xio_create
Ok xio_setopton
onOpenCompleteReg
Ok xio_open
Ok xio_send GET Request
The board was registered with boardType : 4
[Registration]. This device was registered with the correct board type.
[Registration]. Retrieving connection string...
[Registration]. Retrieved connection string.
onCloseCompleteReg callback
Ok xio_close
Ok xio_destroy
Board Registration Done

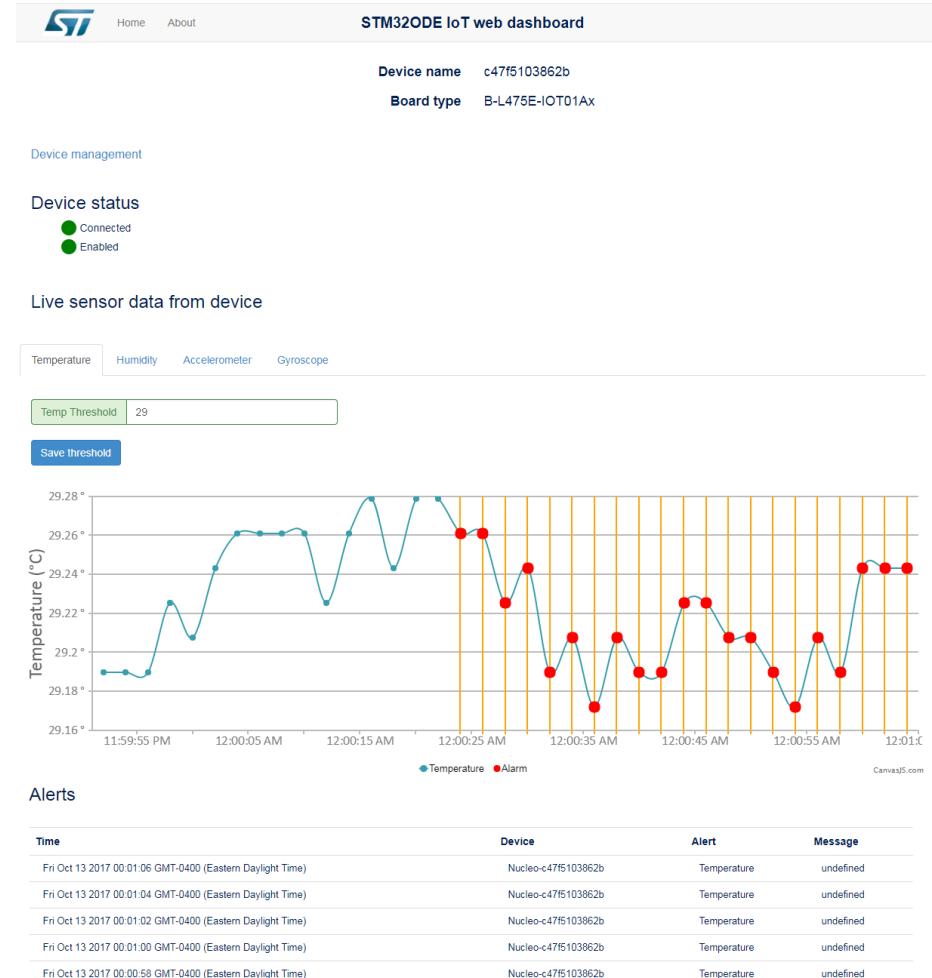
Read the NFC or Connect to:
https://stm32ode.azurewebsites.net/Home/Index/c47f5103862b

iotHubClientHandle Created
iotHubClientSetOption logtrace Ok
iotHubClientSetOption keepalive Ok
Model Instance Created
IoTHubClient_LL_SetMessageCallback...successful.
IoTHubClient_LL_SetDeviceMethodCallback...successful.
IoTHubClient_LL_SetOption(certificates)...successful.
IoTHubClient_LL_SetDeviceTwinCallback...successful.
JSON Decoded
Reported Property Found
Desired Property Found
DesiredTelemetryInterval Desired Property not Found
Channel 1 for Timer 1 started
Ok reported State [1]: Running
-->DeviceTwin CallBack [1]: Status_code = 204
IoTHubClient_LL_SendEventAsynchronous accepted message [1] for transmission to IoT Hub.
Confirmation received for message [1] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsynchronous accepted message [2] for transmission to IoT Hub.
Confirmation received for message [2] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsynchronous accepted message [3] for transmission to IoT Hub.
Confirmation received for message [3] with result = IOTHUB_CLIENT_CONFIRMATION_OK
```

STM32ODE IoT Web Dashboard

111

- The STM32ODE Web Dashboard is a web application hosted on Azure cloud
 - Connects to IoT hub and retrieves sensor data sent from device to cloud
 - Plots data on easy to view graphs
 - Separate tabs for each sensor
 - Allows setting alert thresholds and highlights values exceeding the threshold
 - Unique URL for each device that is registered to the IoT Hub
 - Displays device status
 - Allows device management



Visualize Sensor Data

112

Connected: Device is connected to IoT Hub now

Disconnected:
Device is not connected to IoT Hub now

Enabled:
Device can send data if connected

Warning:
Device close to reaching maximum allowed messages per day

Disabled:
Device has reached maximum allocated data per day; it will be re-enabled next day

STM32ODE IoT web dashboard

Device name: c47f5103862b
Board type: B-L475E-IOT01Ax

Device management

Switch to device management view

Device status

Connected
Enabled

Live sensor data from device

Select dataset to be visualized

Temperature Humidity Accelerometer Gyroscope

Temp Threshold: 29
Save threshold

Set alert threshold and view alerts in graph

Temperature (°C)

Alerts

Time Device Alert Message

Fri Oct 13 2017 00:01:06 GMT-0400 (Eastern Daylight Time)	Nucleo-c47f5103862b	Temperature	undefined
Fri Oct 13 2017 00:01:04 GMT-0400 (Eastern Daylight Time)	Nucleo-c47f5103862b	Temperature	undefined
Fri Oct 13 2017 00:01:02 GMT-0400 (Eastern Daylight Time)	Nucleo-c47f5103862b	Temperature	undefined
Fri Oct 13 2017 00:01:00 GMT-0400 (Eastern Daylight Time)	Nucleo-c47f5103862b	Temperature	undefined
Fri Oct 13 2017 00:00:58 GMT-0400 (Eastern Daylight Time)	Nucleo-c47f5103862b	Temperature	undefined

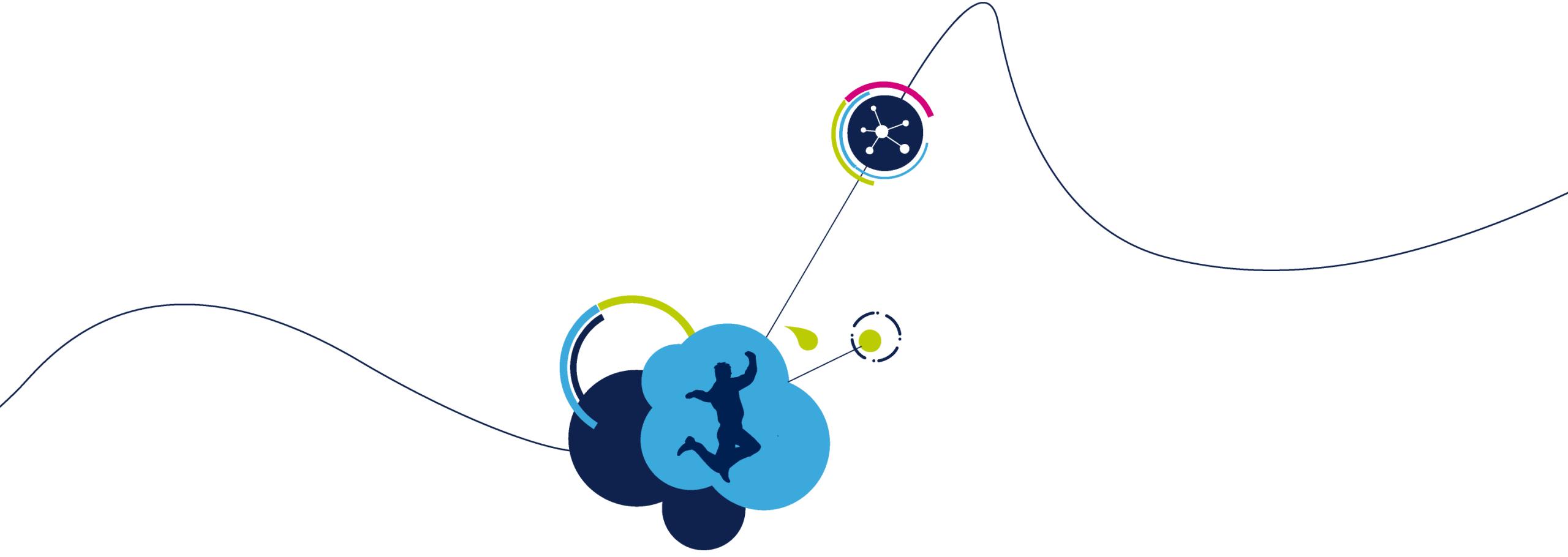
The dashboard interface includes a header with the ST logo and navigation links for Home and About. Below the header, device details are shown: Device name (c47f5103862b) and Board type (B-L475E-IOT01Ax). A 'Device management' button is highlighted with a pink border. The 'Device status' section shows 'Connected' and 'Enabled' status indicators. The main content area displays 'Live sensor data from device' for Temperature, Humidity, Accelerometer, and Gyroscope. A 'Select dataset to be visualized' dropdown menu is open, showing Temperature, Humidity, Accelerometer, and Gyroscope options. A 'Temp Threshold' input field is set to 29, with a 'Save threshold' button below it. A graph for Temperature (°C) shows a blue line with data points and red dots representing alerts. A threshold line is drawn at 29.28°C. The x-axis shows time from 11:59:55 PM to 12:01:00 AM. The y-axis shows temperature from 29.16° to 29.28°. Below the graph, an 'Alerts' section lists five entries with columns for Time, Device, Alert, and Message. The ST logo and 'life.augmented' tagline are visible in the bottom left corner.

Explore Device Management Features

113

The screenshot shows the Azure Device Management interface for a device named 0080E1B8B80B, which is a Nucleo-L476RG board. The interface is divided into several sections:

- Device name:** 0080E1B8B80B
- Board type:** Nucleo-L476RG
- Telemetry:** A button labeled "Go Back to telemetry view" points to this section.
- Twin:** This section contains fields for "Device Id" (0080E1B8B80B) and "MAC address of this device". It also includes a "Tags" section where "Board type" is set to "2" and there is a checkbox for "Is demo device".
- Properties:** This section includes "Desired" and "Reported" tabs. Under "Desired", "Status" is set to "enabled". Under "Reported", "Azure Fw Version" is listed as "Azure_Sns_DM V3.1.0 SDK=1.1.16" and "Azure Status" is "Running".
- Control device:** A "Pause" dropdown menu and a "Send message to device" button.
- Call Method on Device:** A "Reboot" dropdown menu and a "Call Method" button.
- Firmware update:** A dropdown menu showing "Azure_L476RG.bin" and a "Force firmware update" button.
- Cloud to device messages:** A box labeled "Direct Methods called on device (Quit and Reboot)" is connected to the "Call Method" button.
- Visualize DeviceTwin:** A box containing a list:
 - Device status
 - Reported properties
 - Desired propertiesA red arrow points from this box to the "Twin" section.



FP-CLD-AZURE1 Function Pack Overview

FP-CLD-AZURE1: Features

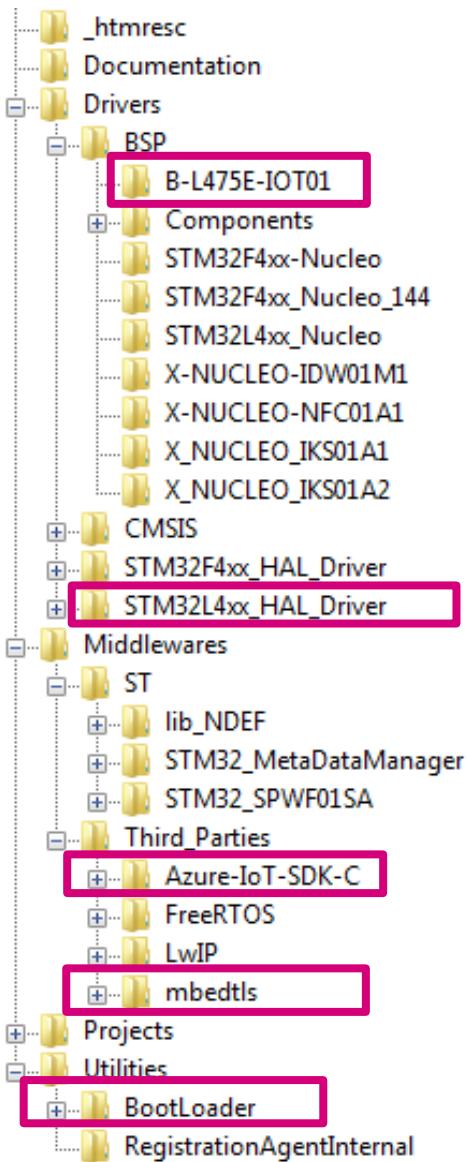
115

- Integrated Microsoft Azure IoT device Software Development Kit. Includes sample application for:
 - Connection to IoT Hub and transmission of Device-to-Cloud messages based on MQTT protocol
 - Implementation of a Device Management solution based on
 - Cloud-to-Device messages
 - Direct Methods
 - Device Properties
 - Firmware Update
- Secure network communication with the cloud application based on mbedTLS library
- Include middleware and drivers for Wi-Fi connectivity, NFC, inertial and environmental sensors
 - Provide software interface to access temperature and humidity sensor (HTS221), pressure sensor (LPS22HB), inertial sensors (LIS3MDL, LSM6DSL), ToF and gesture detection (VL53L0X) and to write and read the M24SR64-Y RFID/NFC tag

Latest info available at www.st.com
FP-CLD-AZURE1

FP-CLD-AZURE1: Folders Structure

116



Drivers and BSP for Discovery Kit IoT node

WiFi and NFC middleware

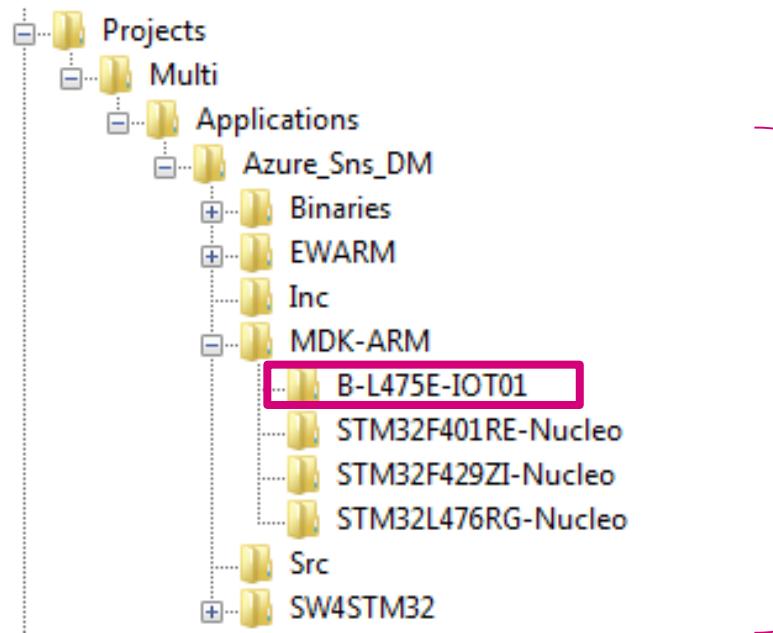
Azure IoT SDK and mbedTLS

Bootloader to handle FW update

FP-CLD-AZURE1: Folders Structure (Cont'd)

117

Sample application based on Azure IoT SDK for data telemetry and Device Management



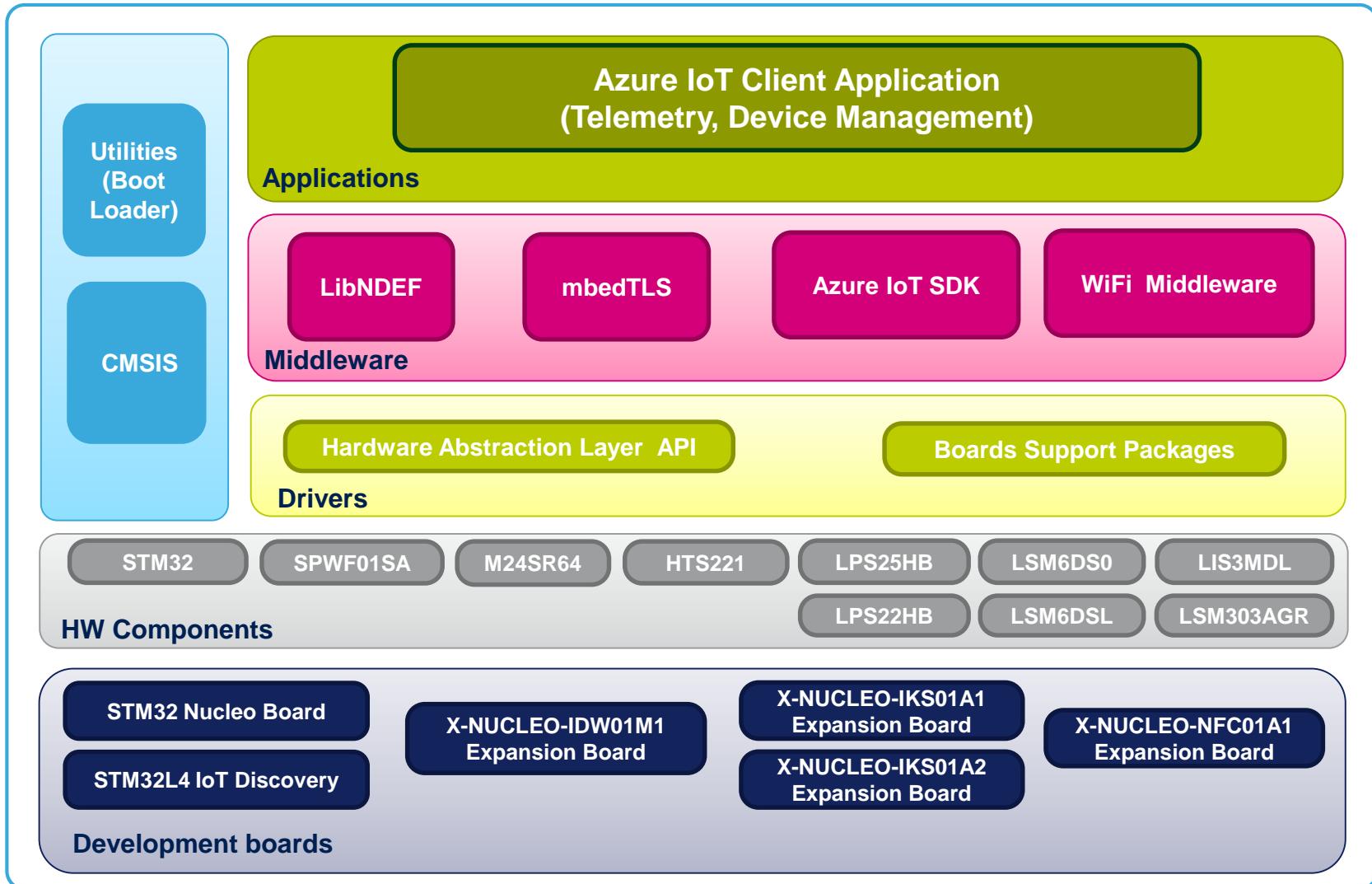
Solution files for 3 IDE:

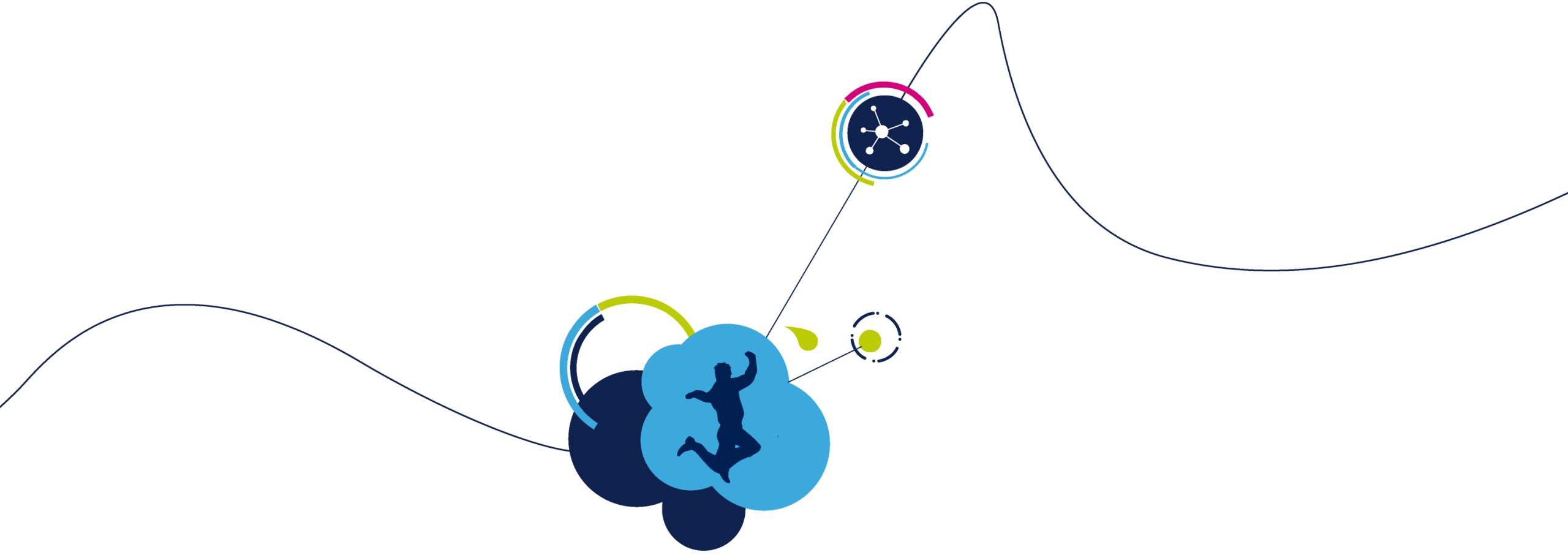
- IAR (EWARM)
- System Workbench (SW4STM32)
- ARM Keil (MDK-ARM)

C:\STM32L4AzureSeminar\Hands-on\STM32CubeFunctionPack_AZURE1_V3.1.0\Projects\Multi\Applications\Azure_Sns_DM\MDK-ARM\B-L475E-IOT01\

FP-CLD-AZURE1: Software Architecture

118





Lab 3: Connect your STM32 IoT Node to Azure IoT Hub

- In this lab, we will:
 - Create and login to a Free **Azure account**
 - Create an **IoT Hub**
 - Create a new **Device** node and retrieve it's **Connection String**
 - Setup your IoT node using the **sample application** provided with the Function Pack

Azure Login: Seminar vs Individual Steps

121

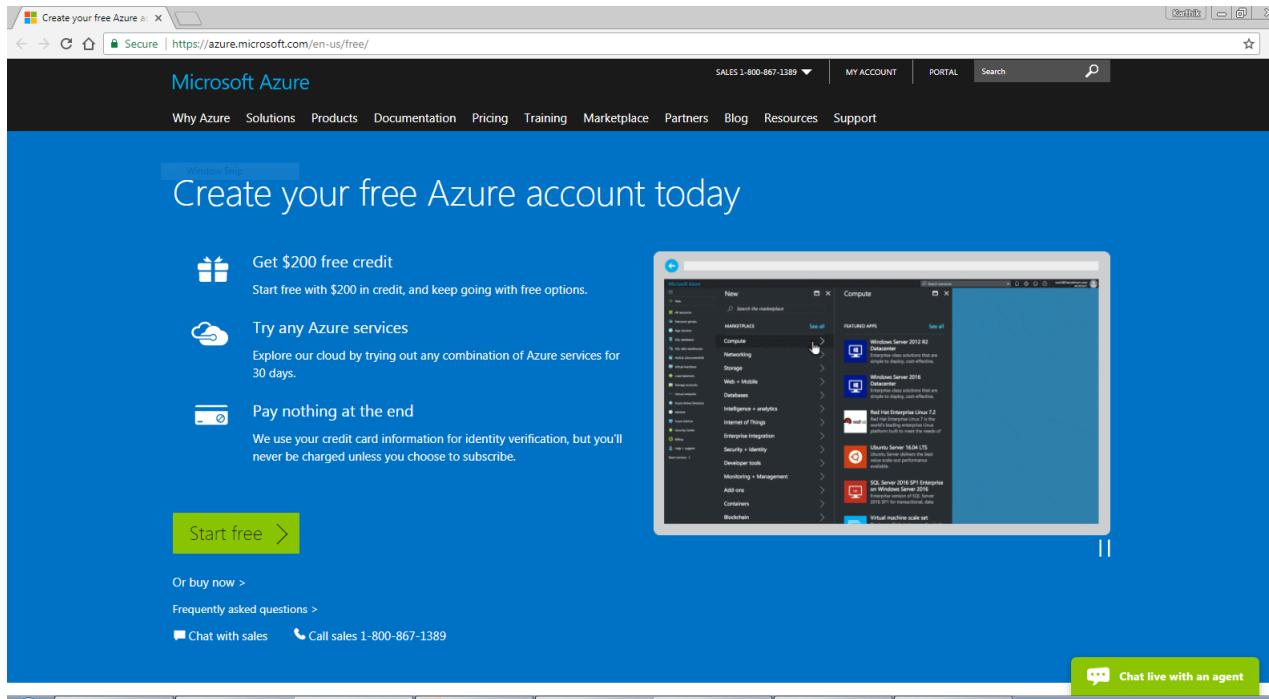
- The following slides show the steps to create a new IoT Hub that you can follow later on your own
- For this seminar, we will use Azure login credentials and IoT Hub already created for this workshop by ST
 - For this workshop session, you should skip the steps to create a new Azure account and to create the new IoT Hub
 - The steps to be skipped during this workshop are marked with “We did these steps for you”.

Step 1: Create a Free Azure account

122

- <https://azure.microsoft.com/en-us/free/>

- Need to create a Microsoft account if you don't already have one
- Need mobile phone and credit card for identity verification
- Free \$200 credit; will not charge credit card; will not automatically renew at the end of trial period.

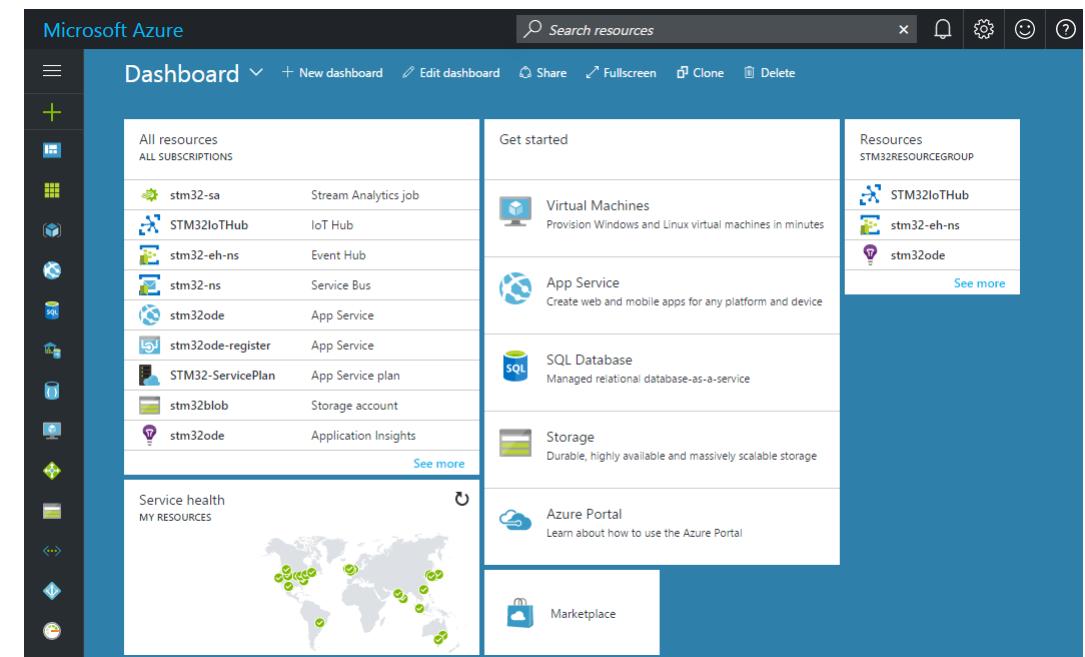
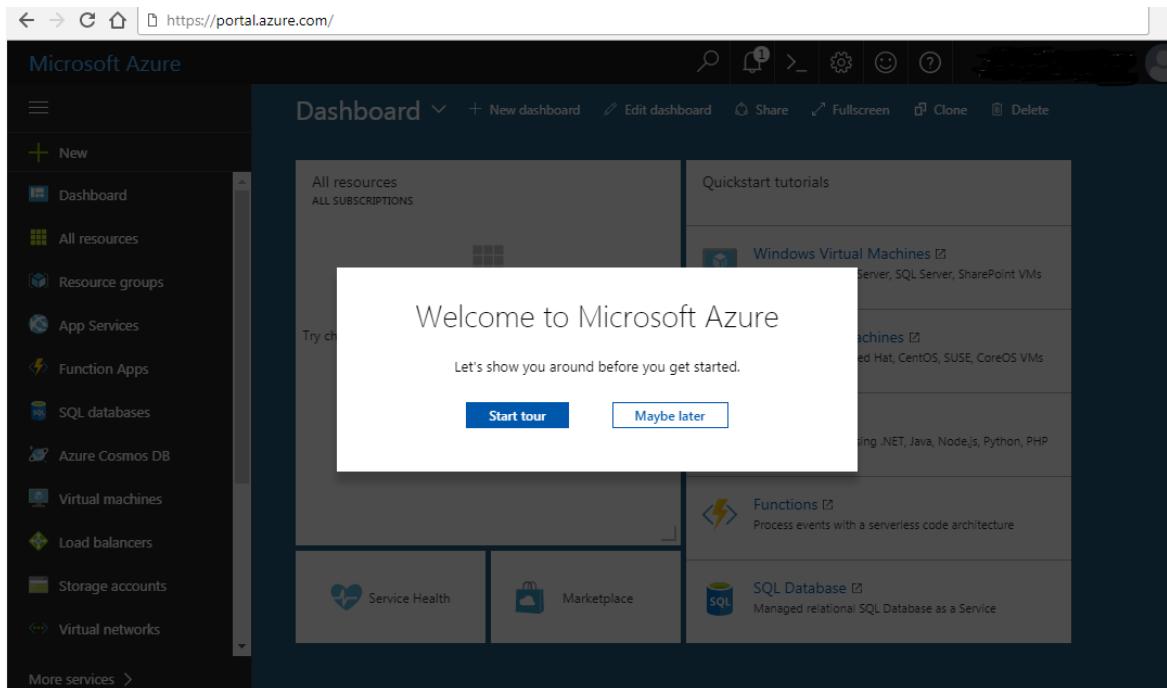


We did these
steps for you

Step 2: Log into your Azure account

123

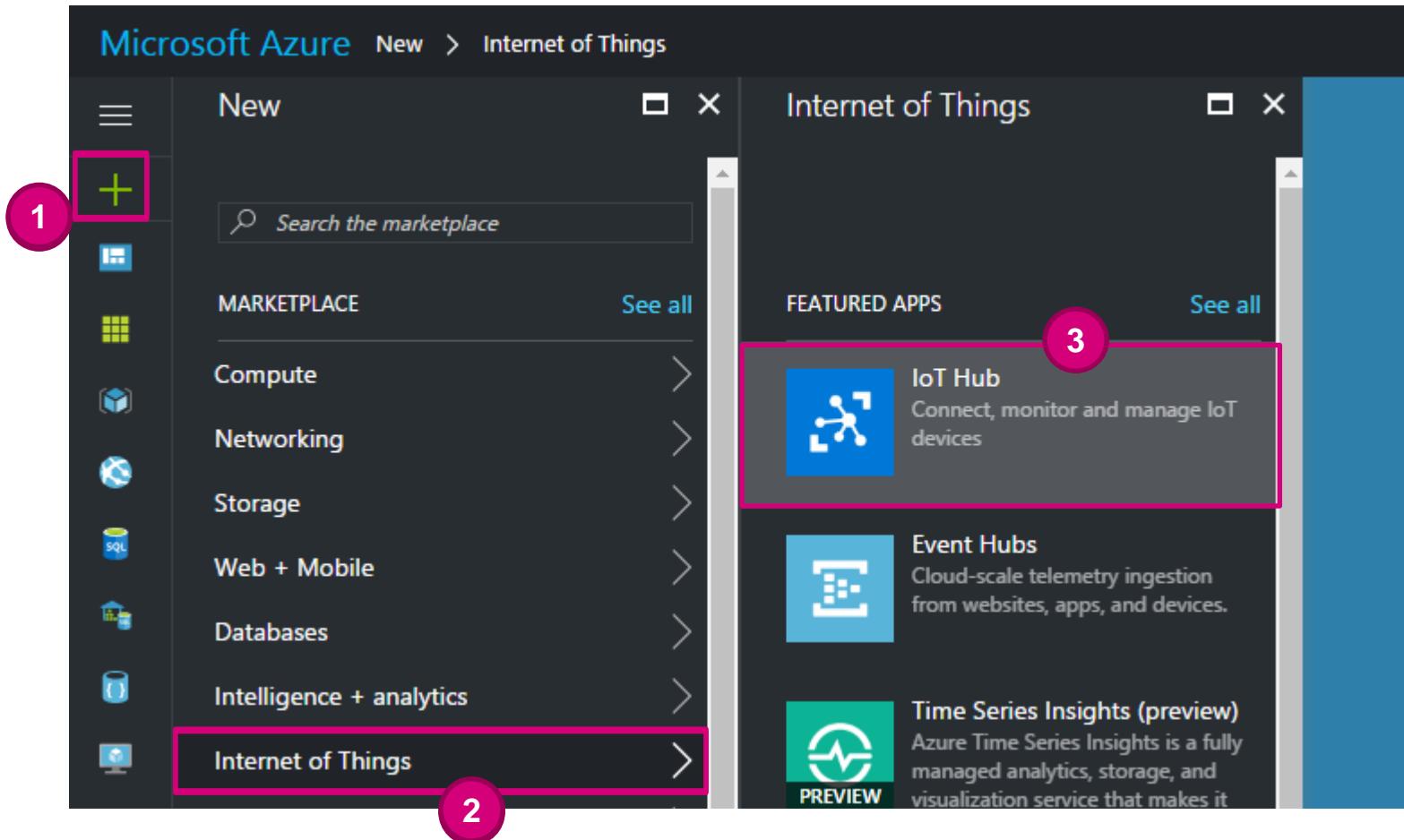
- Make sure your laptop is connected to the internet
- Go to <https://portal.azure.com> and sign in with the credentials below:
 - login: SeminarParticipant@iotcloudservicesst.onmicrosoft.com
 - password: STM32iot



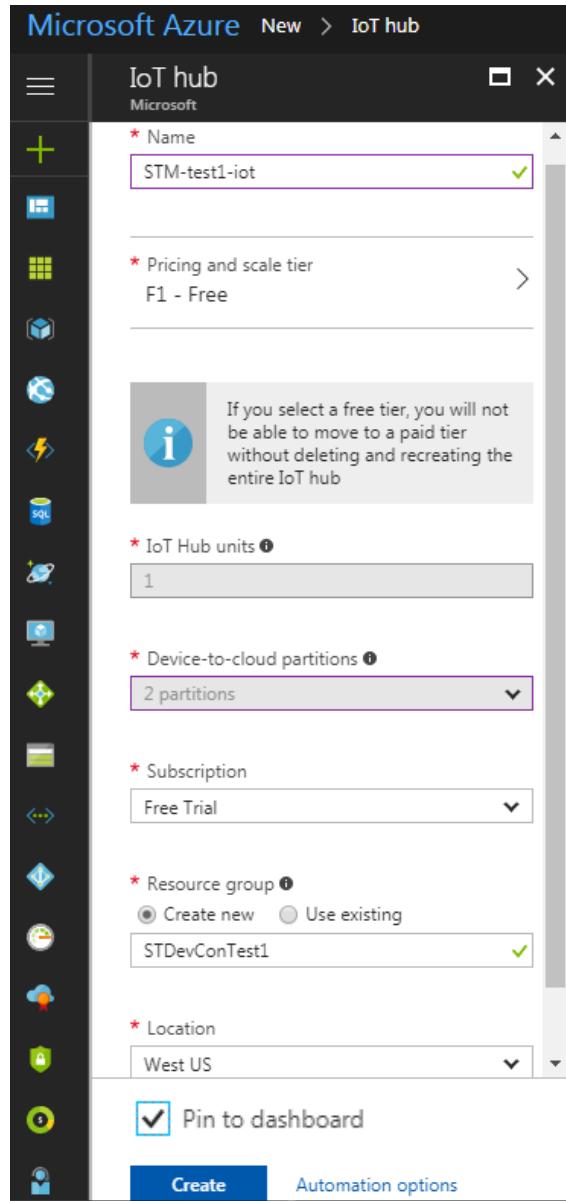
Step 3: Create an IoT Hub

124

- In the left jump-bar, click on + → Internet of Things → IoT Hub



Step 3: Create an IoT Hub (Cont'd)



Insert a unique id for your IoTHub

Sample application doesn't require a specific tier; the free tier can be used.

We did these steps for you

The Subscription would be "Free Trial"

You can create a container folder for your Azure services

Geographical region where the IoT Hub will be hosted

Create the IoT Hub

IoT Hub Created

126

- The IoT Hub is now included in the list of resources and resource groups

All resources
iotcloudservicesst (Default Directory)

+ Add Columns Refresh

Subscriptions: Pay-As-You-Go

Filter by name... All resource gro... All types All locations No grouping

2 items

NAME	TYPE	RESOURCE G...	LOCATION	SUBSCRIPTION
stm32iot	IoT Hub	stm32iot	West US	Pay-As-You-Go
stm32iot	Storage account	stm32iot	West US	Pay-As-You-Go

Step 4: IoT Hub Overview

127

Select IoT Hub

- 1 Click on the All resources icon to view all your resources
- 2 Click on the IoT Hub just created from the list of resources in the left jump bar

All resources
iotcloudservicesst (Default Directory)

+ Add Columns Refresh

Subscriptions: Pay-As-You-Go

Filter by name... All resource gro... All types All locations No grouping

2 items

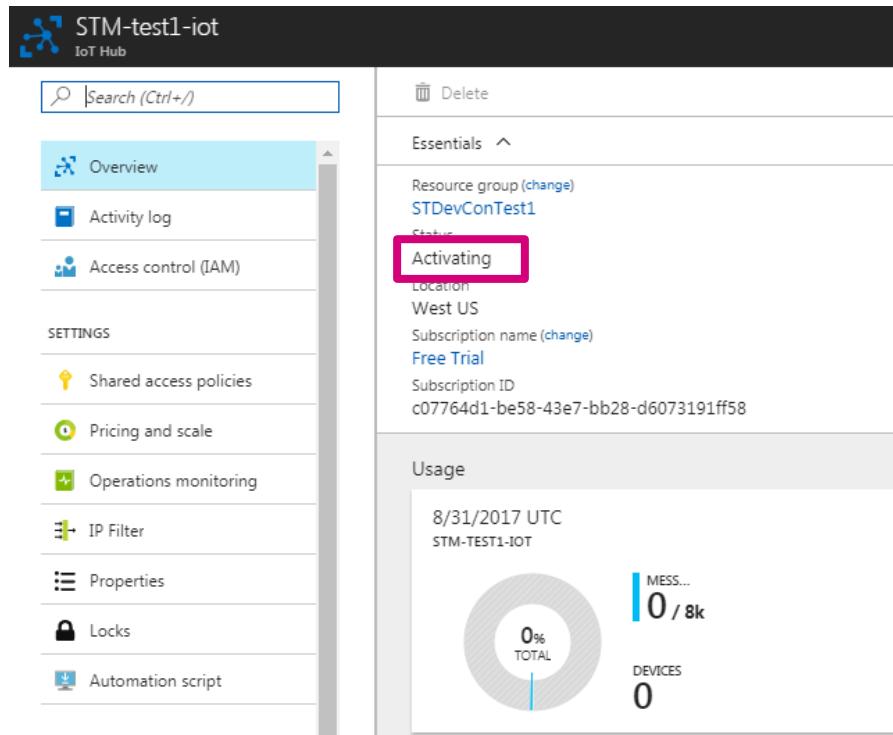
NAME	TYPE	RESOURCE G...	LOCATION	SUBSCRIPTION
stm32iot	IoT Hub	stm32iot	West US	Pay-As-You-Go
stm32iot	Storage account	stm32iot	West US	Pay-As-You-Go

Step 4: IoT Hub Overview

128

wait for “**Active**” status (Cont’d)

- If you see the status of your newly created IoT Hub as “**Activating**”, you may need to wait for a few minutes for activation.



Note: You may also have to log out and log back in for status to change from “**Activating**” to “**Active**” for a newly created IoT Hub.

Step 4: IoT Hub Overview

129

- The overview page will be opened. It shows the **essential data of your IoT Hub** and the **current usage** in terms of registered devices and messages received.

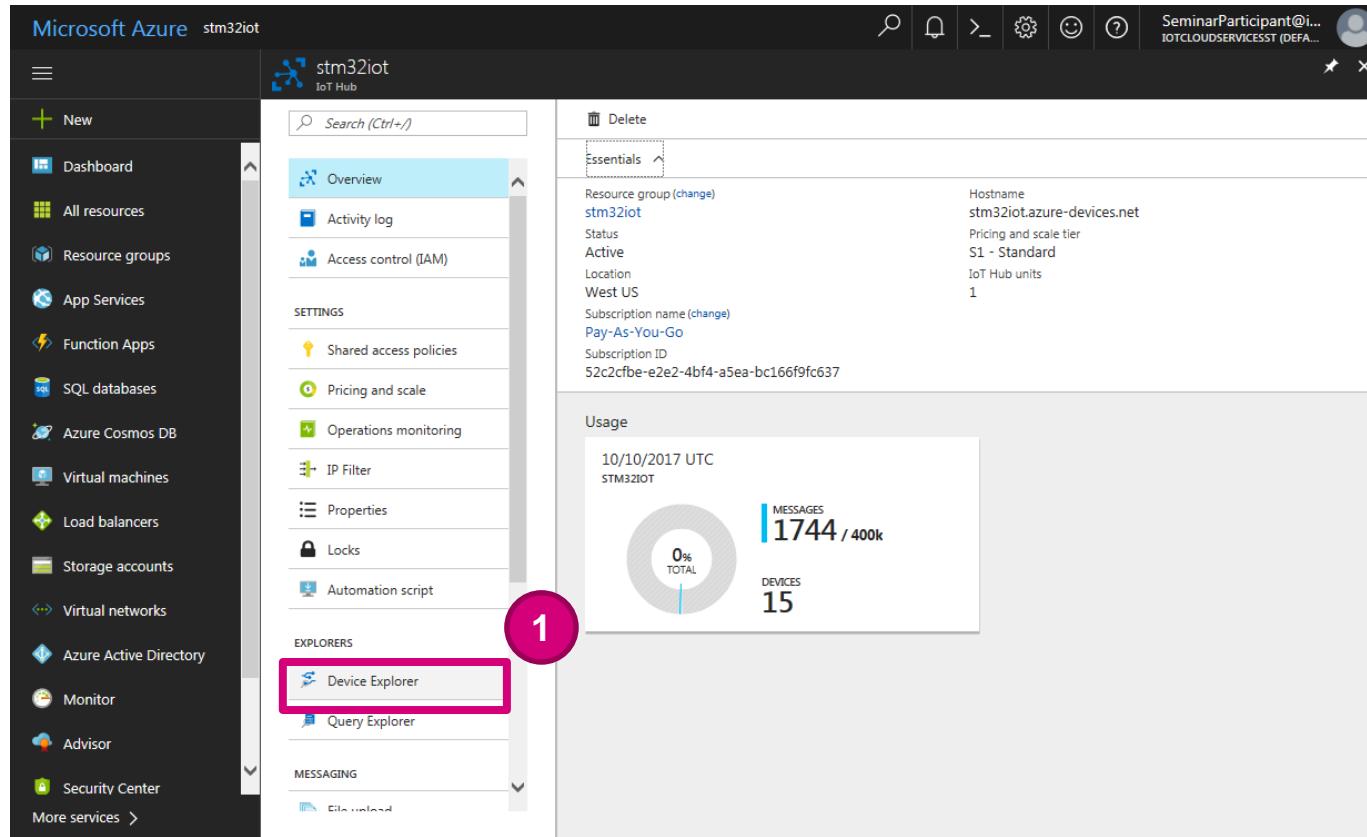
The screenshot shows the Azure IoT Hub Overview page for the hub 'STM-test1-iot'. The left sidebar contains navigation links: Overview (selected), Activity log, Access control (IAM), Shared access policies, Pricing and scale, Operations monitoring, IP Filter, Properties, Locks, and Automation script. The main area is divided into two sections: 'Essentials' and 'Usage'. The 'Essentials' section displays resource group (STDevConTest1), status (Active), location (West US), subscription name (Free Trial), and subscription ID (c07764d1-be58-43e7-bb28-d6073191ff58). The 'Usage' section shows the date (8/31/2017 UTC) and device ID (STM-TEST1-IOT). It includes a donut chart showing 0% total devices and a bar chart showing 0 messages received out of 8k.

Essentials	
Resource group (change)	Hostname
STDevConTest1	STM-test1-iot.azure-devices.net
Status	Pricing and scale tier
Active	F1 - Free
Location	IoT Hub units
West US	1
Subscription name (change)	
Free Trial	
Subscription ID	
c07764d1-be58-43e7-bb28-d6073191ff58	

Usage	
8/31/2017 UTC	MESS...
STM-TEST1-IOT	0 / 8k
0% TOTAL	DEVICES 0

Step 5: Create a new device in the IoT Hub registry

1 Click Device Explorer



Step 5: Create a new device in the IoT Hub registry (Cont'd)

2 Click Add

The screenshot shows the Microsoft Azure Device Explorer interface for an IoT Hub named 'stm32iot'. A pink circle with the number '2' highlights the '+ Add' button in the top navigation bar. The main area displays a query editor with the following content:

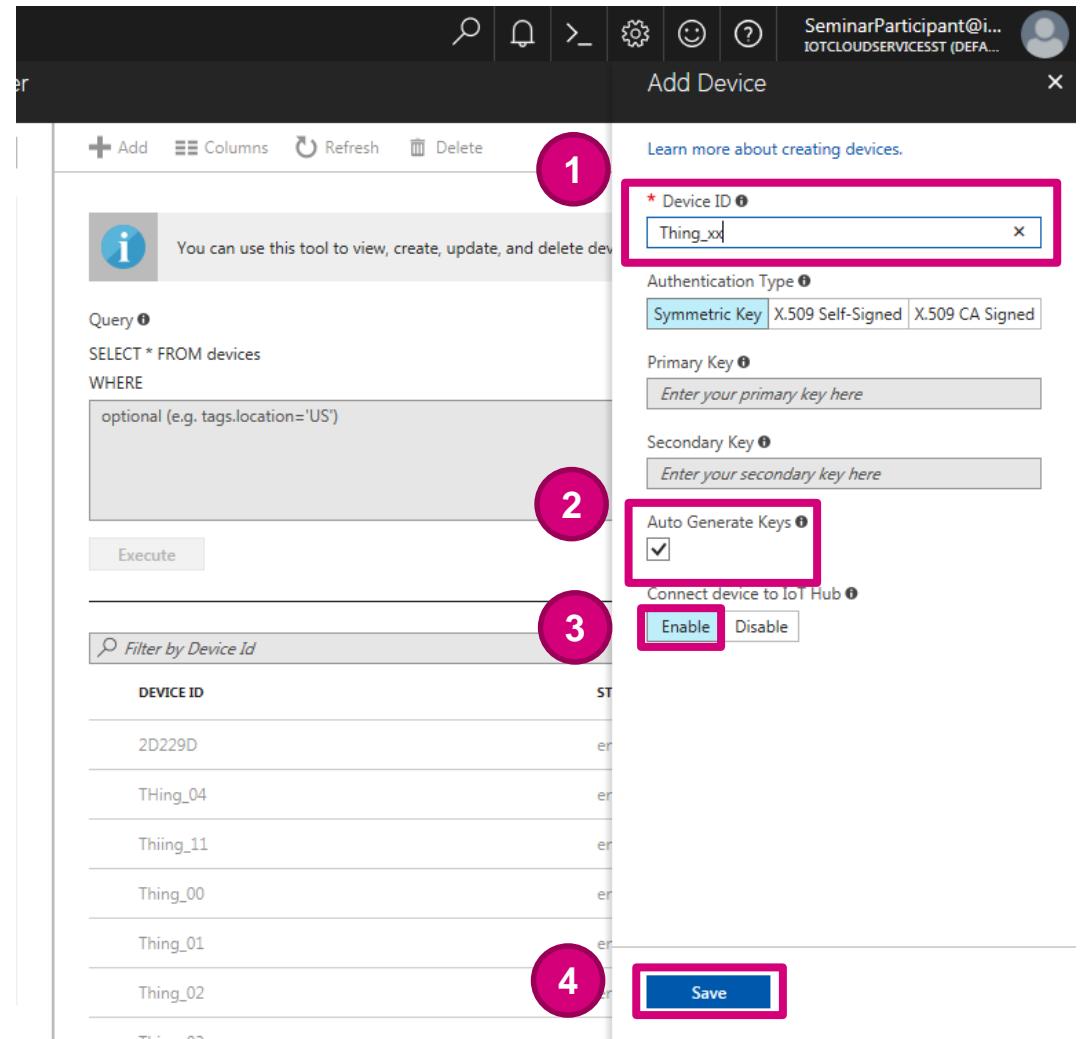
```
SELECT * FROM devices  
WHERE  
optional (e.g. tags.location='US')
```

Below the query editor is a table titled 'DEVICE ID' and 'STATUS' showing the following data:

DEVICE ID	STATUS
2D229D	enabled
THing_04	enabled
Thiing_11	enabled
Thing_00	enabled
Thing_01	enabled
Thing_02	enabled
Thing_03	enabled

Step 5: Create a new device in the IoT Hub registry

- 1 Insert your Device ID “Thing_XX”: “XX” is your participant number (2 characters) printed on the box
- 2 Check Auto Generate Keys
- 3 Enable Connect device to IoT Hub
- 4 Click Save



Step 6: Retrieve Connection string

- 1 After the device is created, open the device in the **Device Explorer** pane.

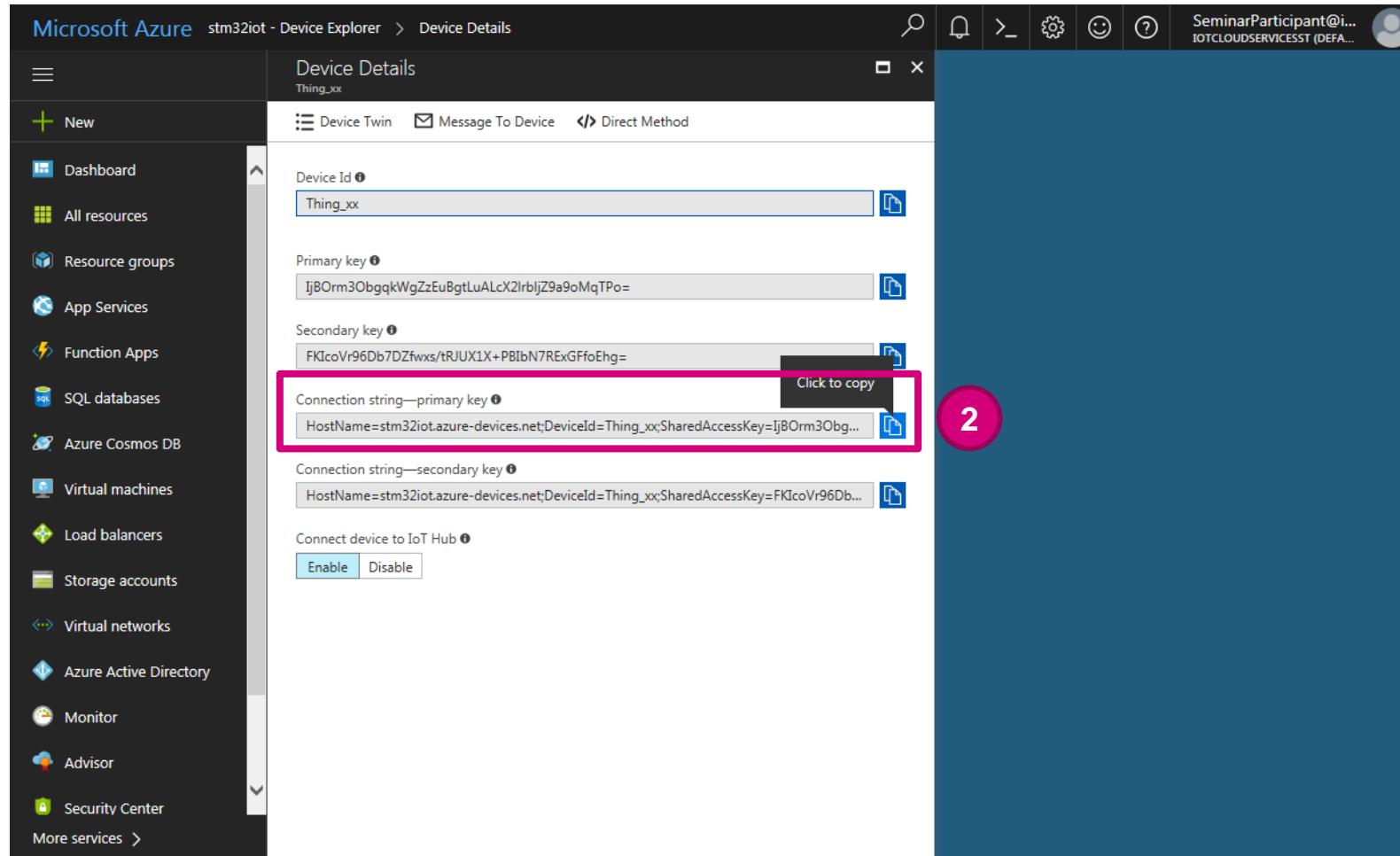
The screenshot shows the Microsoft Azure Device Explorer pane for an IoT Hub named 'stm32iot'. The left sidebar lists various Azure services, and the main pane displays a table of devices. A circled number '1' points to the 'Device Explorer' entry in the sidebar. The table shows the following data:

DEVICE ID	STATUS
Thing_08	enabled
Thing_10	enabled
Thing_13	enabled
Thing_22	enabled
Thing_33	enabled
Thing_xx	enabled

Step 6: Retrieve Connection string (Cont'd)

134

- 2 Copy the Device Connection string into Notepad.



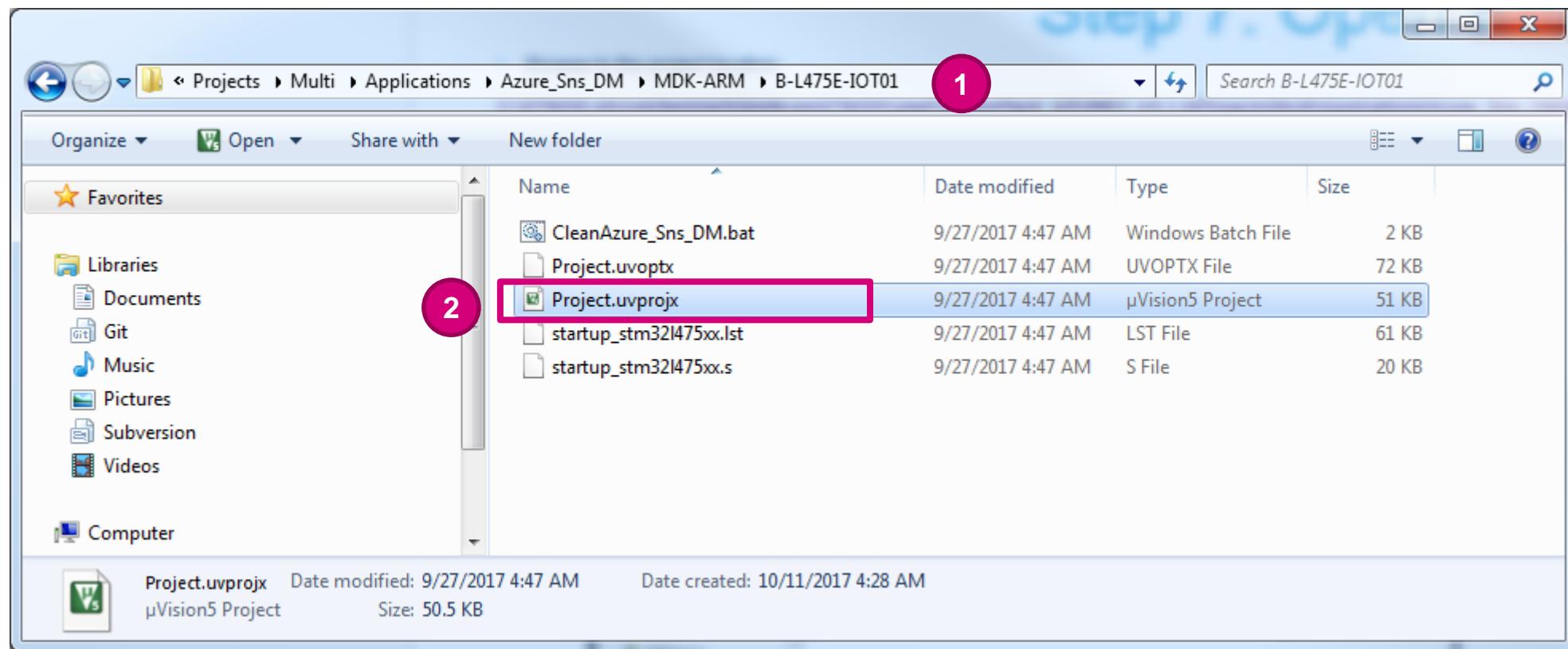
Step 7: Open Keil Project

135

- 1 Browse to the project location:

C:\STM32L4AzureSeminar\Hands-on\STM32CubeFunctionPack_AZURE1_V3.1.0\Projects\Multi\Applications\Azure_Sns_DM\MDK-ARM\B-L475E-IOT01

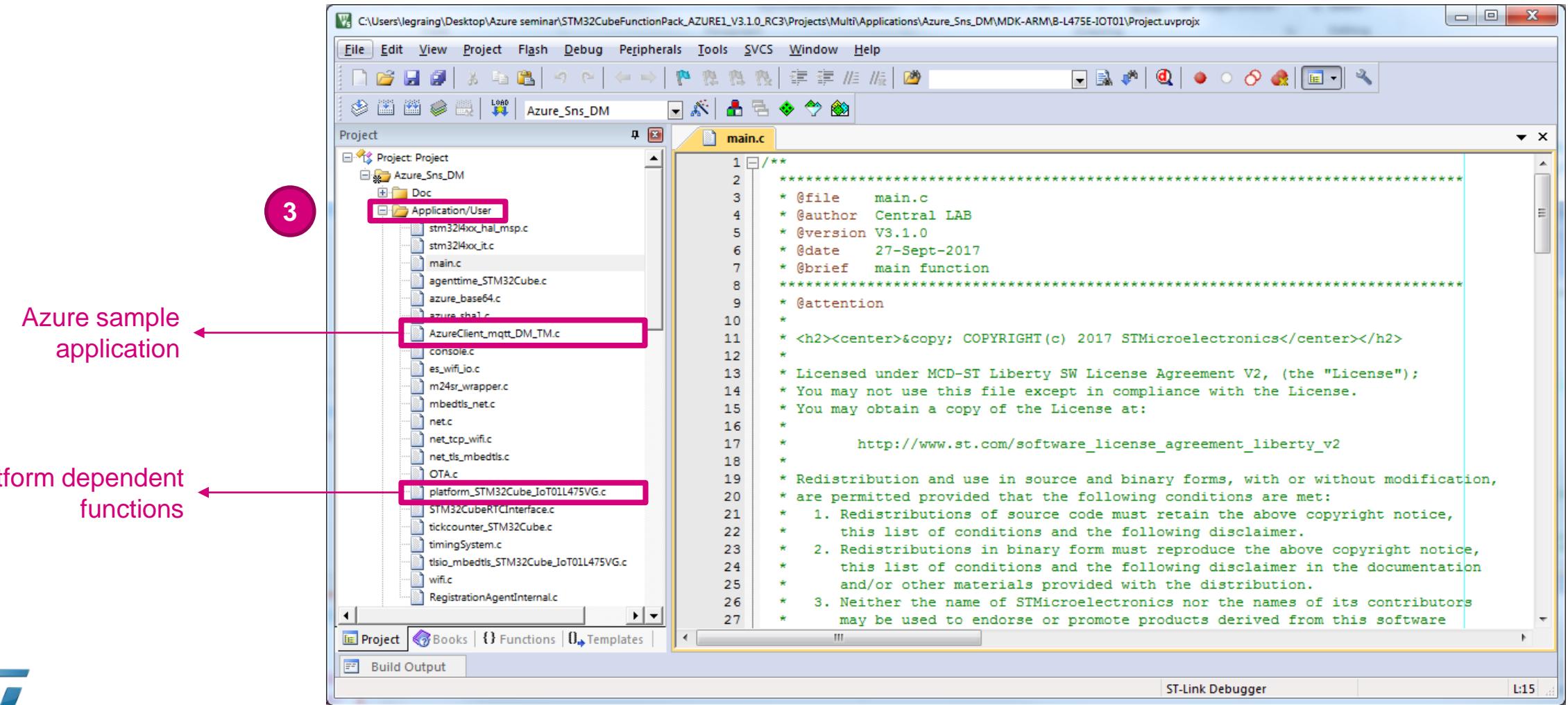
- 2 Double click on **Project.uvprojx** project file to bring up Keil



Step 7: Open Keil Project (Cont'd)

136

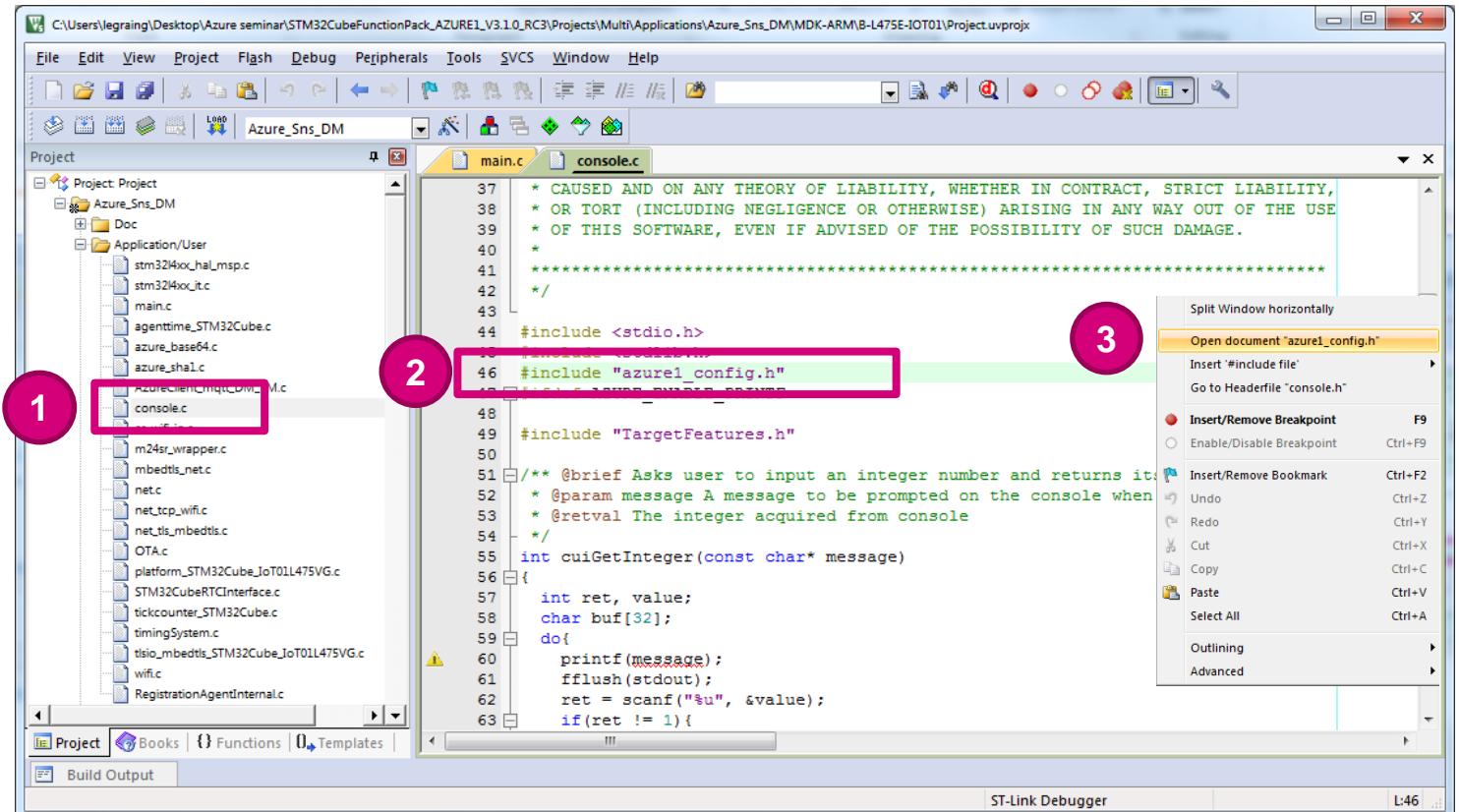
3 Expand Application/User Group in Project Explorer window



Step 8: Disable Registration and set Connection String

137

- 1 In Project Window double-click on `console.c`
- 2 Right-click on line 46
- 3 Click Open document "azure1_config.h"

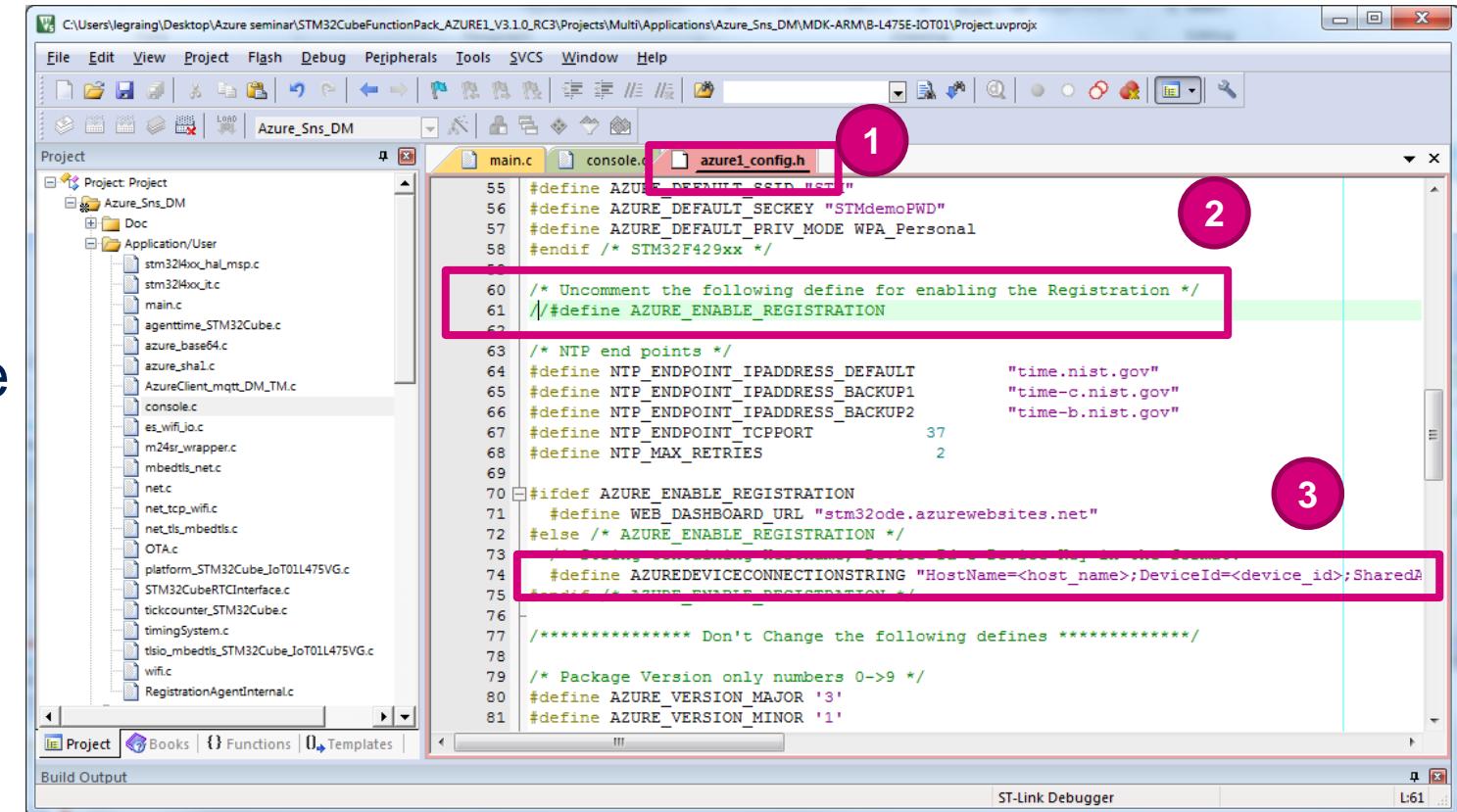


Step 8: Disable Registration and set Connection String (Cont'd)

1 In file `azure1_config.h`

2 Comment Line 61

3 Uncomment line 74 and paste the Connection String retrieved in previous step (slide 134) replacing
`"HostName=<host_name>;DeviceId=<device...>"`



```

55 #define AZURE_DEFAULT_SSID "ST"
56 #define AZURE_DEFAULT_SECKEY "STMdemopWD"
57 #define AZURE_DEFAULT_PRIV_MODE WPA_Personal
58 #endif /* STM32F429xx */
59
60 /* Uncomment the following define for enabling the Registration */
61 // #define AZURE_ENABLE_REGISTRATION
62
63 /* NTP end points */
64 #define NTP_ENDPOINT_IPADDRESS_DEFAULT "time.nist.gov"
65 #define NTP_ENDPOINT_IPADDRESS_BACKUP1 "time-c.nist.gov"
66 #define NTP_ENDPOINT_IPADDRESS_BACKUP2 "time-b.nist.gov"
67 #define NTP_ENDPOINT_TCPPORT 37
68 #define NTP_MAX_RETRIES 2
69
70 #ifdef AZURE_ENABLE_REGISTRATION
71     #define WEB_DASHBOARD_URL "stm32ode.azurewebsites.net"
72 #else /* AZURE_ENABLE_REGISTRATION */
73
74     #define AZUREDEVICECONNECTIONSTRING "HostName=<host_name>;DeviceId=<device...>" //AZURE_ENABLE_REGISTRATION
75
76
77 //***** Don't Change the following defines *****/
78
79 /* Package Version only numbers 0->9 */
80 #define AZURE_VERSION_MAJOR '3'
81 #define AZURE_VERSION_MINOR '1'

```

Make sure to have the Connection String in between double-quotes aka “....”

Step 9: Rebuild the Project

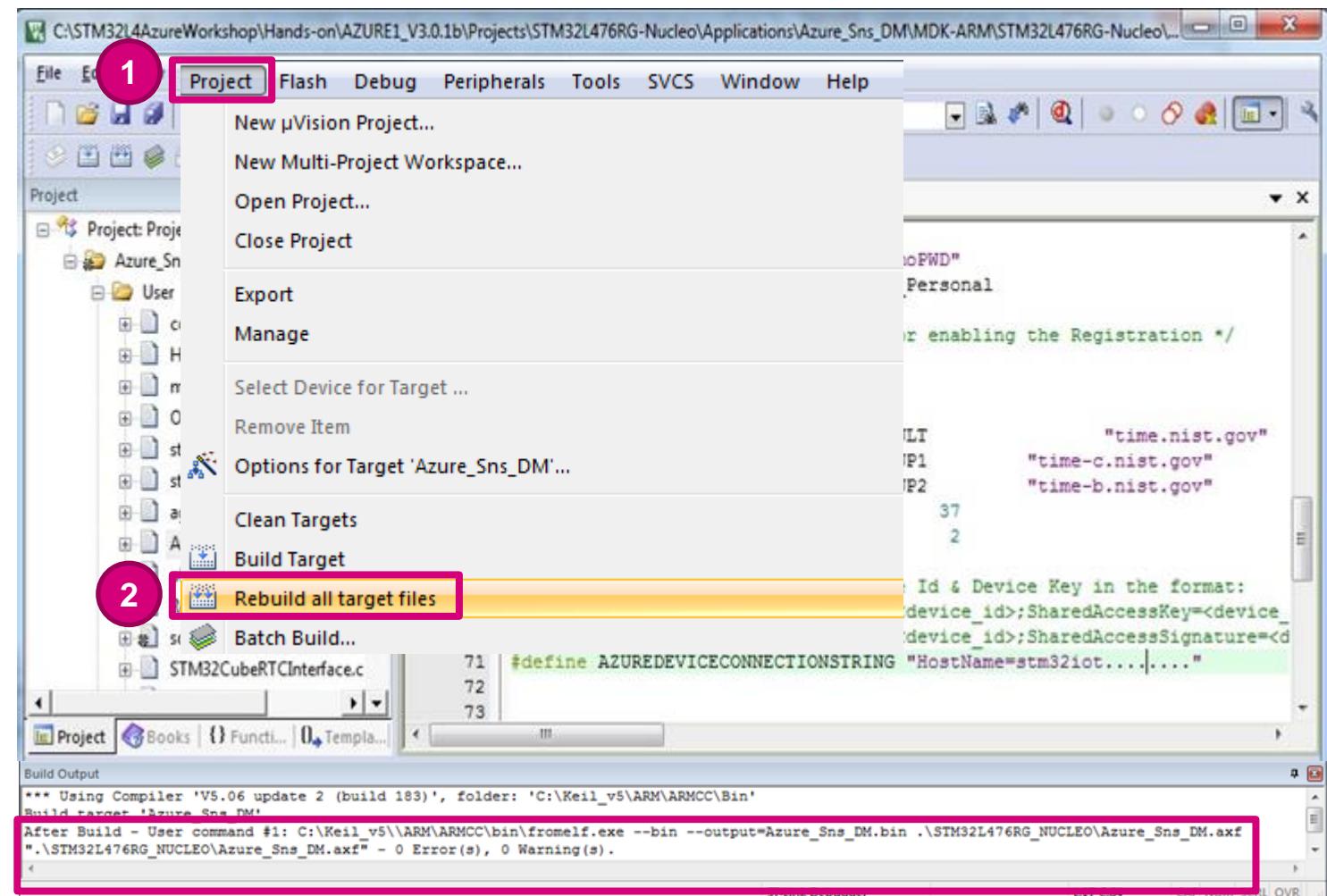
139

1 Click on Project menu

2 Click on Rebuild all target files

In the Build Output window, it should report '0 Errors(s), 0 Warnings'

You are now ready to Program your STM32



Step 10: Program IoT node

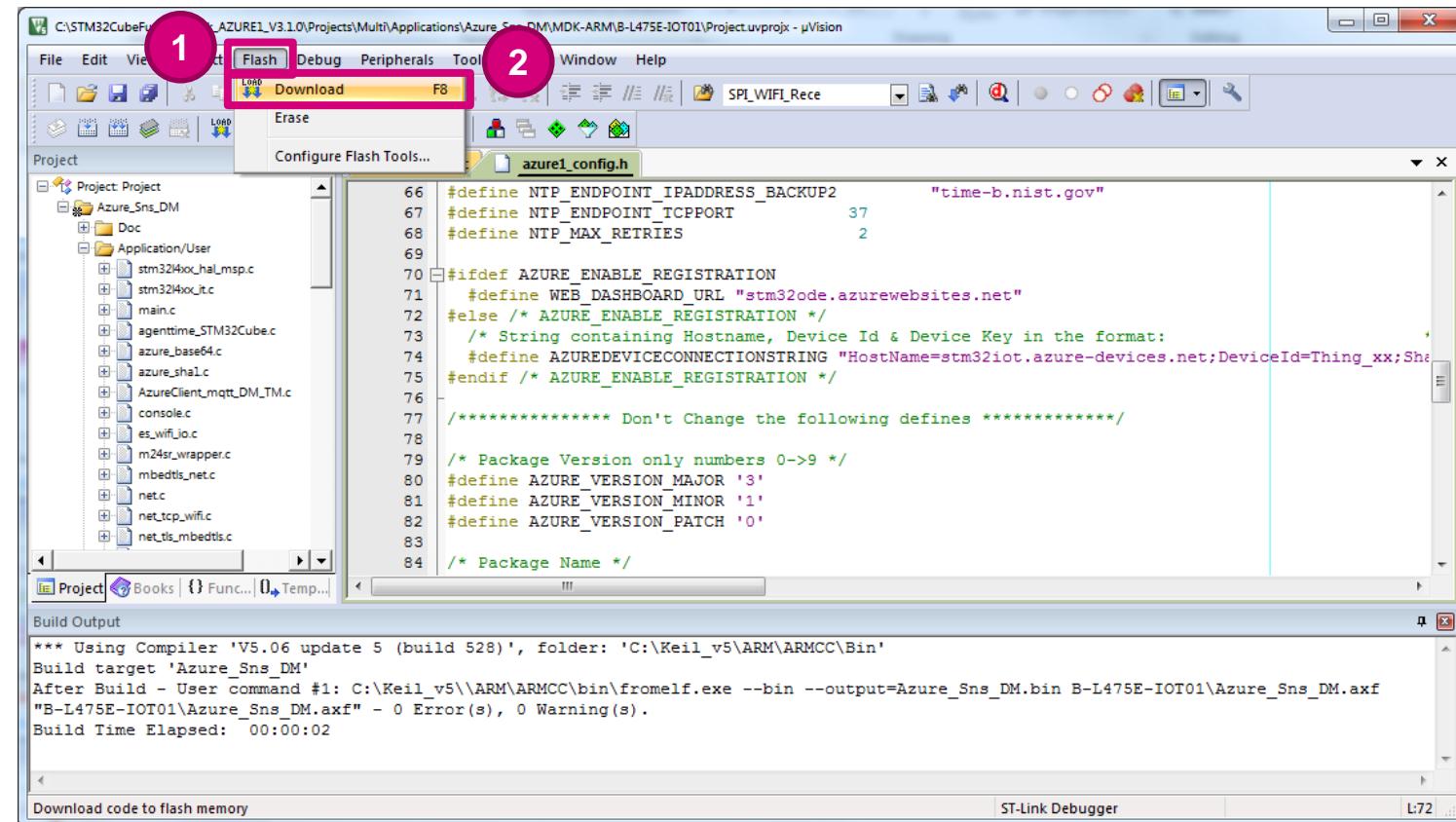
140

1 Click on Flash menu

2 Click on Download

Progress can be monitored in the bottom left **status bar**.

Your IoT node should now be programmed



Step 11: Check Connection

- Restart the application by pressing the Reset Button on Nucleo Development Board (Black button). Wait until the board is connected to Wi-Fi access point and to the IoT Hub
- Each 2 seconds, a message is created and transmitted to the IoT Hub using MQTT protocol. For each message transmitted, a **confirmation acknowledge** is received back by the device

```
Ok reported State [1]: Running
-->DeviceTwinCallBack [1]: Status_code = 204
IoTHubClient_LL_SendEventAsync accepted message [1] for transmission to IoT Hub.
Confirmation received for message [1] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [2] for transmission to IoT Hub.
Confirmation received for message [2] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [3] for transmission to IoT Hub.
Confirmation received for message [3] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [4] for transmission to IoT Hub.
Confirmation received for message [4] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [5] for transmission to IoT Hub.
Confirmation received for message [5] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [6] for transmission to IoT Hub.
Confirmation received for message [6] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [7] for transmission to IoT Hub.
Confirmation received for message [7] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [8] for transmission to IoT Hub.
Confirmation received for message [8] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [9] for transmission to IoT Hub.
Confirmation received for message [9] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [10] for transmission to IoT Hub.
Confirmation received for message [10] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [11] for transmission to IoT Hub.
Confirmation received for message [11] with result = IOTHUB_CLIENT_CONFIRMATION_OK
```

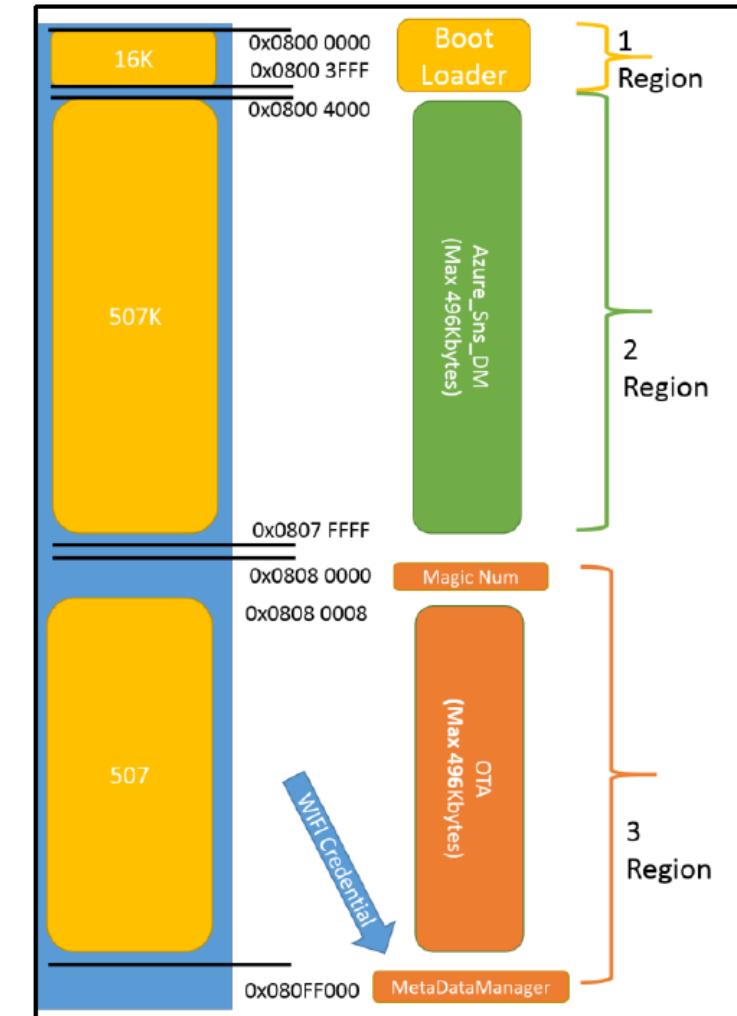
Flash the IoT node

Using Installed Script (Cont'd)

142

The STM32 flash memory is divided in different regions:

1. The first region contains a custom boot loader (required for firmware update)
2. The second region contains the application firmware
3. The third region is used in a firmware update procedure to store the new downloaded firmware before updating it, and to save the Wi-Fi credentials inside the Meta Data Manager. The Meta Data Manager is placed at the end of the Flash



Flash the IoT node

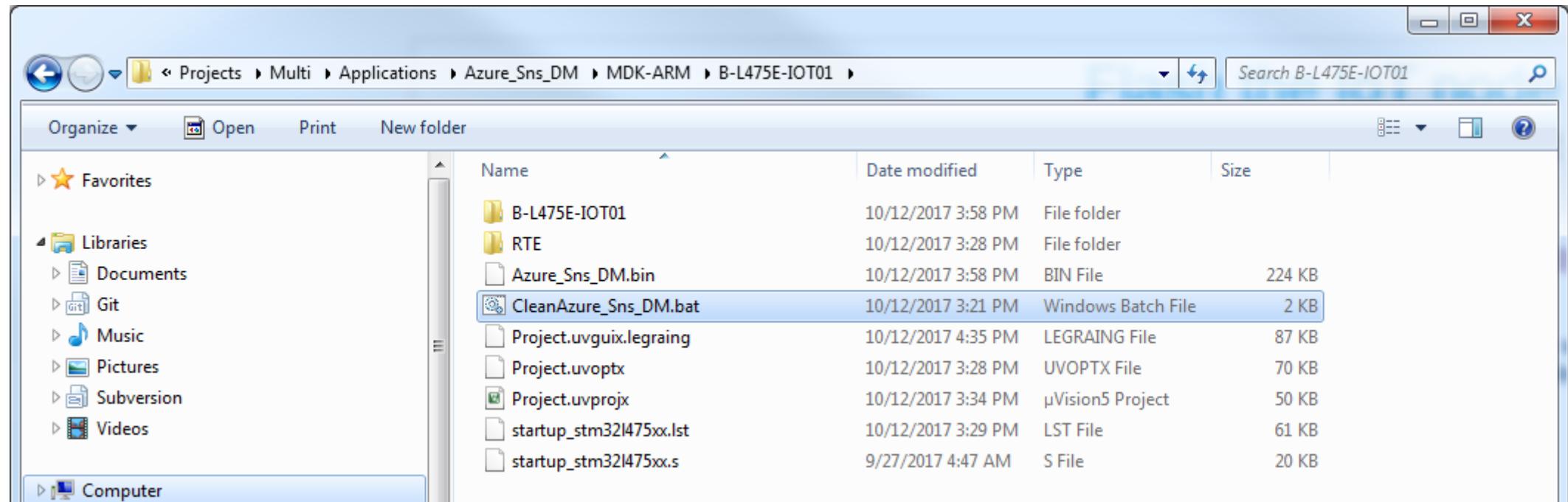
Using Installed Script

143

- Once built the project, it is necessary to launch the “.bat” script located in folder

C:\STM32L4AzureSeminar\Hands-on\STM32CubeFunctionPack_AZURE1_V3.1.0\Projects\Multi\Applications\Azure_Sns_DM\MDK-ARM\B-L475E-IOT01

In order to flash the MCU, the usage of the script is required to install in the MCU a bootloader together with the project binary. Bootloader is used in Lab 6 for Firmware update.



Flash the IoT node

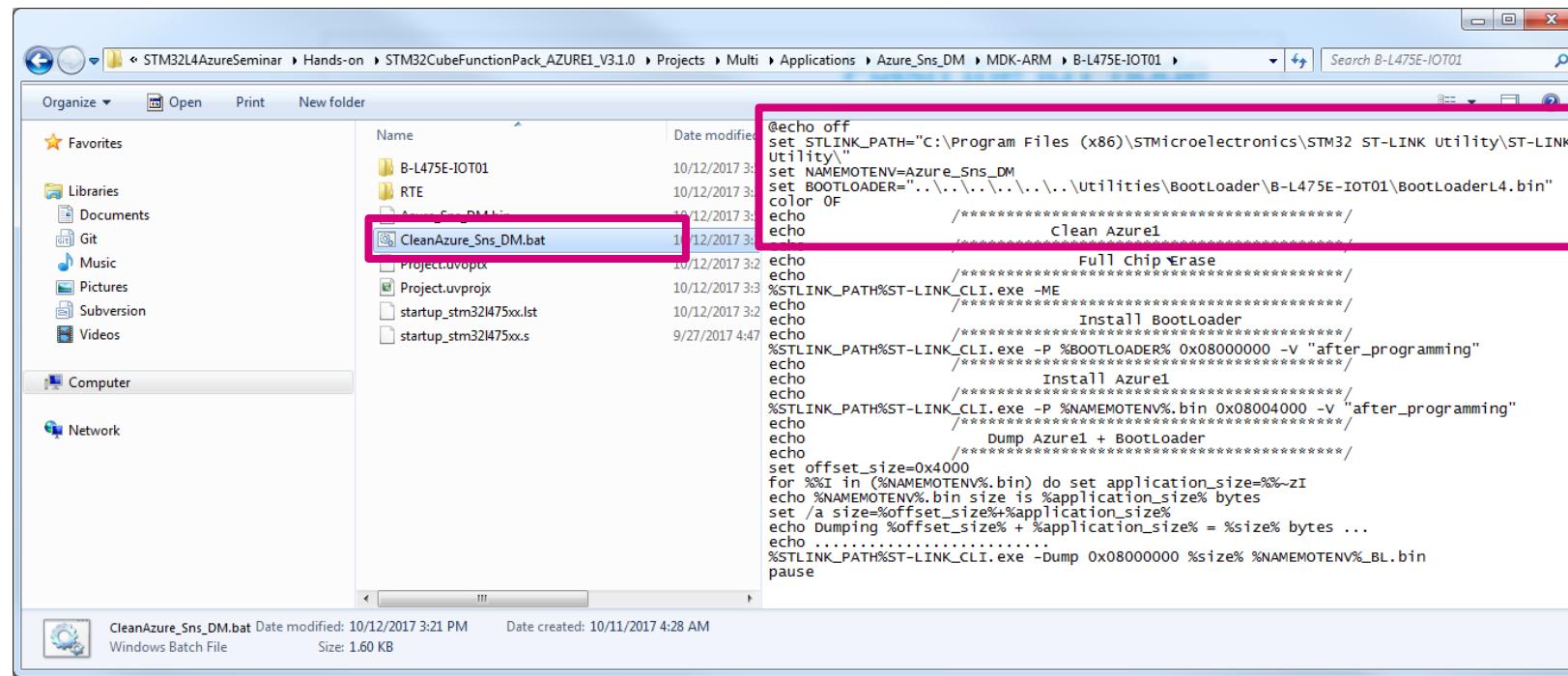
Using Installed Script (Cont'd)

144

- Windows script uses a tool available with ST-Link utility installed by the installer
- Installation path in Windows7 for ST-Link utility is

C:\Program Files (x86)\STMicroelectronicsTraining\ST-LINK\
which is the one used by default in the script

- You can verify that the path in the script correspond to your installation path for ST-Link utility



Note: If you see an error “The system cannot find the path specified”, please run the ST-Link Utility installer from the path below to install the tool at the default path:

C:\STM32L4AzureSeminar\Software\STS-W-LINK004_STM32_ST-LINK_Utility-4.0.0.exe

Flash the IoT node

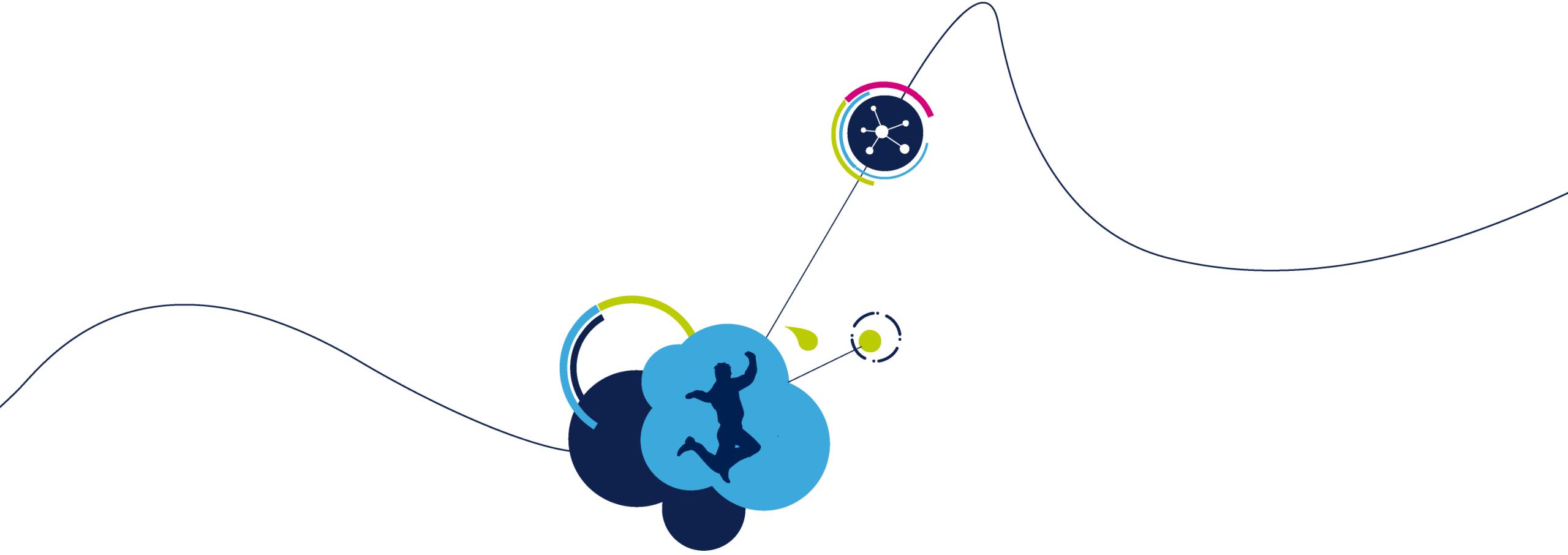
Using Installed Script (Cont'd)

145

- Double click on `CleanAzure_Sns_DM.bat` to launch the script

```
C:\Windows\system32\cmd.exe
=====
Azure_Sns_DM.bin size is 288124 bytes
Dumping 0x4000 + 288124 = 304508 bytes ...
STM32 ST-LINK CLI v3.0.0.0
STM32 ST-LINK Command Line Interface
ST-LINK SN : 066BFF574857847167072640
ST-LINK Firmware version : V2J27M15
Connected via SWD.
SWD Frequency = 4000K.
Target voltage = 3.3 V.
Connection mode : Normal.
Device ID:0x415
Device flash Size : 1024 Kbytes
Device family :STM32L4x1/L4x5/L4x6
Dumping memory ...
Address = 0x08000000
Memory Size = 0x0004A57C
=====
Saving file [Azure_Sns_DM_BL.bin] 100%
Dumping memory to Azure_Sns_DM_BL.bin succeeded
Press any key to continue . . .
```

- Once done, the console should show that the board was programmed successfully
- Press any key to close the console

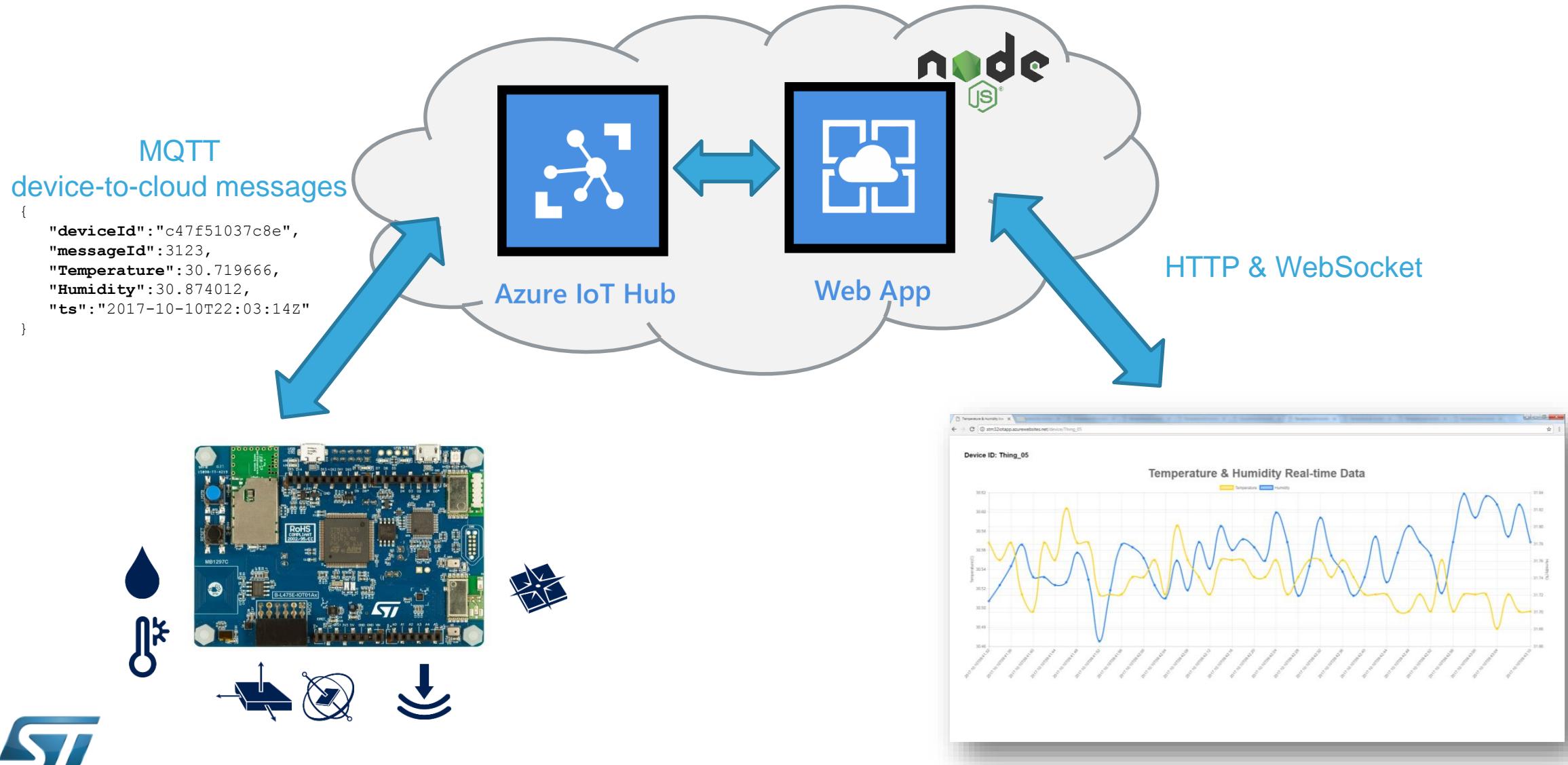


Lab 4: Monitor IoT Node Sensor Data sent to Azure Cloud

- The sample application reads raw data from sensors board, compose a message and through Azure IoT SDK iteratively transmits messages to your IoT Hub
- In this lab you will learn how to visualize the log of messages received by your IoT Hub using a Web App
- You will also learn how to customize and extend data read from the sensor board

Introduction

148

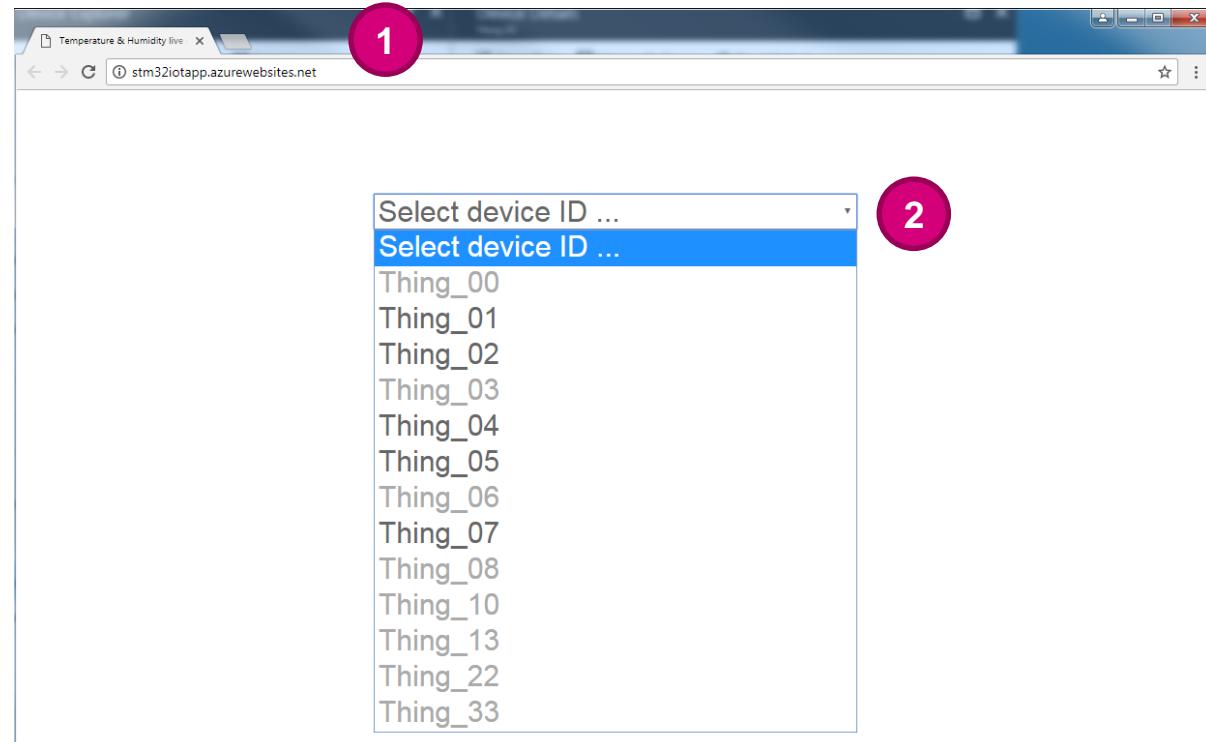


Step 1: Open Web App

1 Open

- <http://stm32iotapp.azurewebsites.net>

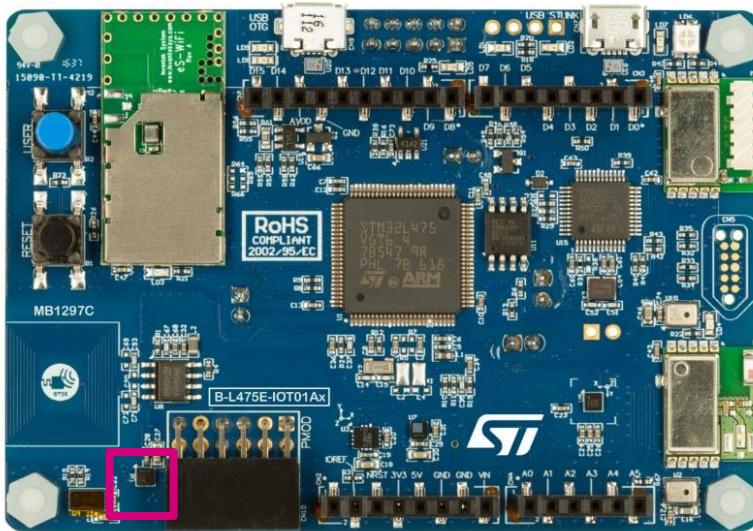
2 Select Device ID



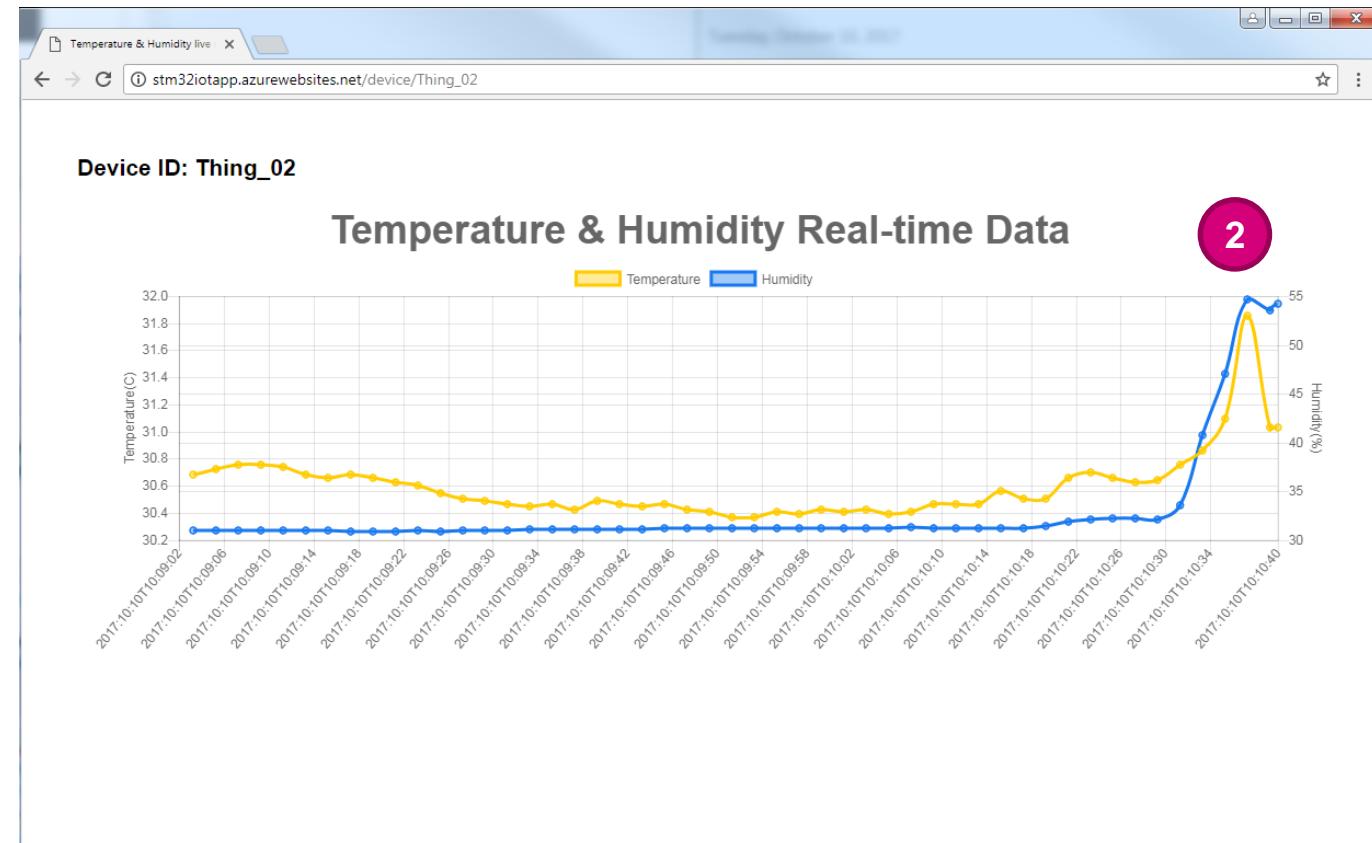
Step 2: Visualize real-time sensor data

150

- 1 Optionally, check data by blowing onto the temperature sensor
- 2 Monitor the spike in the real-time visualization



1



Create your own IoT Web App

151

- To learn how to create your own “Web App”, go to:

<https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-live-data-visualization-in-web-apps>

- You will learn to
 - Create a web app in the Azure portal.
 - Get your IoT hub ready for data access by adding a consumer group.
 - Configure the web app to read sensor data from your IoT hub.
 - Upload a web application to be hosted by the web app.

Customize Usage of the Sensor Board

152

- The message transmitted to your IoT Hub is composed inside function `SendSNSData()` in file `AzureClient_mqtt_DM_TM.c` (`Azure_Sns_DM/User`)

```
/* Read the Data from the Sensors */  
FillTheModelInstance();
```

Read raw data from sensors to Azure1 data structure, using driver APIs
`(BSP_TEMPERATURE_Get_Temp,
BSP_TEMPERATURE_Get_Hum, etc.)`

```
messages = (EVENT_INSTANCE *) calloc(1,sizeof(EVENT_INSTANCE));  
if(messages==NULL) {  
    AZURE_PRINTF("Err: Allocating Memory for messages to IoT Hub\r\n");  
    while(1) {  
        ;  
    }  
} else {  
    messages->this = (void *)messages;  
}
```

```
if (SERIALIZE(&destination, &destinationSize,  
            Azure1->id , Azure1->name, Azure1->ts,  
            Azure1->mtype, Azure1->temp, Azure1->hum,  
            Azure1->accX , Azure1->accY, Azure1->accZ,  
            Azure1->gyrX , Azure1->gyrY, Azure1->gyrZ) != CODEFIRST_OK){  
    AZURE_PRINTF("Err: failed to serialize\r\n");  
    while(1) {  
        ;  
    }  
}
```

Convert sensors data structure into a JSON string

```
if ((messages->messageHandle = IoTHubMessage_CreateFromByteArray(destination, destinationSize)) == NULL) {  
    AZURE_PRINTF("Err: iotHubMessageHandle is NULL!\r\n");  
} else {
```

```
if (IoTHttpClient_LL_SendEventAsync(iotHttpClientHandle, messages->messageHandle, SendConfirmationCallback, messages) != IOTHUB_CLIENT_OK) {  
    AZURE_PRINTF("Err: IoTHttpClient_LL_SendEventAsync.....FAILED!\r\n");  
} else {  
    AZURE_PRINTF("IoTHttpClient_LL_SendEventAsync accepted message [%d] for transmission to IoT Hub.\r\n", SentMessagesCount);  
}
```

Create message handler

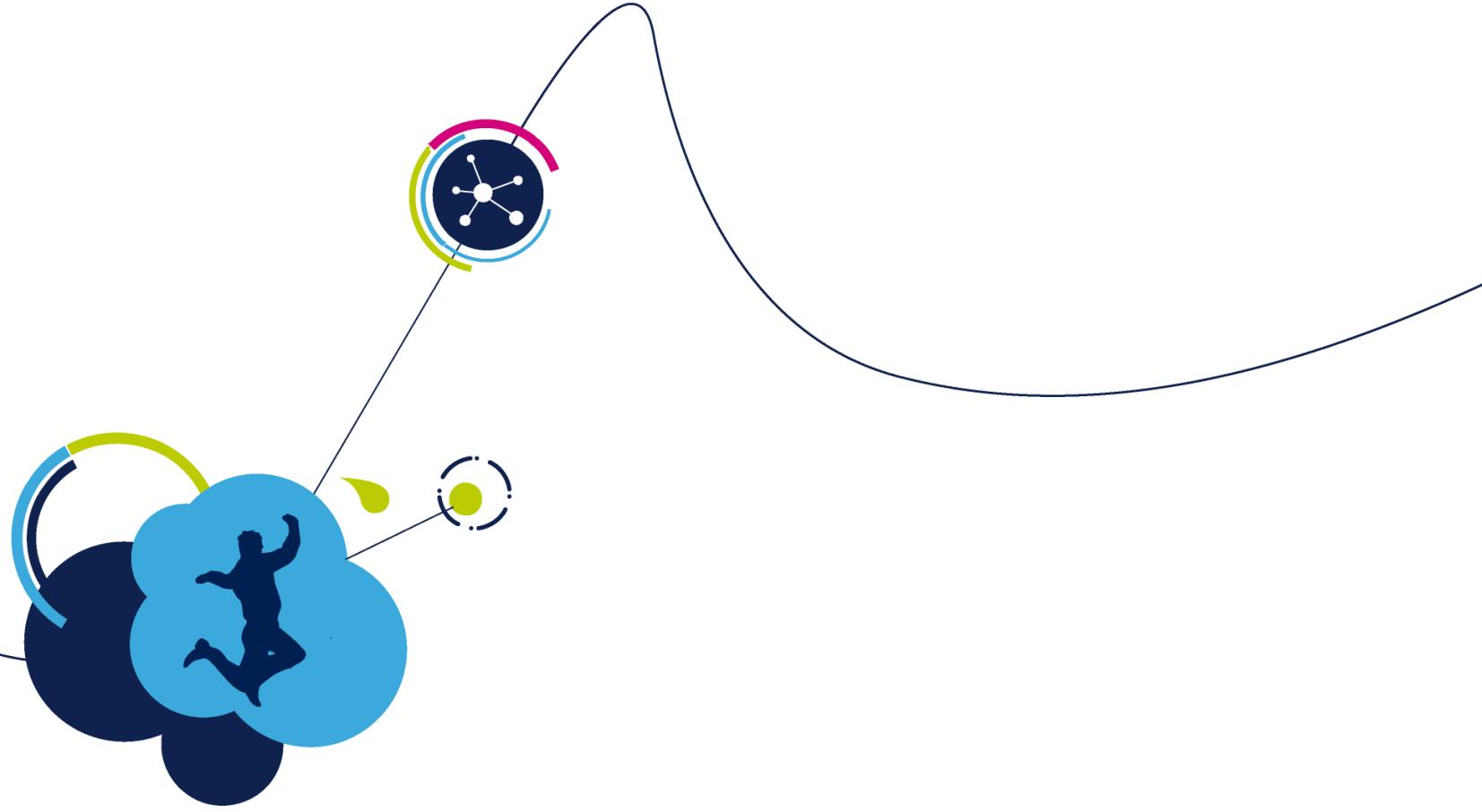
Send the message

of the Sensor Board (Cont'd)

- Function **SendSNSData()** is called by a timer interrupt every two seconds. Transmission rate for messages can be changed using Desired Properties or directly in the code in `AzureClient_mqtt_DM_TM.c` by using **ChangeTelemetryInterval API**

```
void ChangeTelemetryInterval (void* argument)
{
    Azure1_t *Azure = argument;
    if((Azure1->DesiredTelemetryInterval>0) & (Azure1->DesiredTelemetryInterval<=30)){
        Azure1->TelemetryInterval = TargetBoardFeatures.TIM_CC1_Pulse = Azure1->DesiredTelemetryInterval*2000 /* 2Hz Timer Frequency */;
    } else {
        AZURE_PRINTF("Err: Received a Not Allowed desired Telemetry Interval=%d (1<=Allowed<=30)\r\n",Azure->DesiredTelemetryInterval);
    }
}
```

- By default the message transmitted to IoT Hub contains the following raw sensors data
 - Temperature, Humidity, Accelerometers (X,Y,Z), Gyroscope (X,Y,Z)
- The sensors board also generates raw sensor data related to:
 - Pressure (inside function `FillTheModeInstance()` use API: **BSP_PRESSURE_Get_Press**)
 - Magnetometer (inside function `FillTheModeInstance()` use API: **BSP_MAGNETO_Get_Axes**)



Lab 5: Manage IoT Node from Azure portal

- IoT Hub provides three options for device apps to expose functionality to a back-end app:
 - Direct methods for communications that require immediate confirmation of the result.
Direct methods are often used for interactive control of devices such as turning on a fan.
 - Twin's desired properties for long-running commands intended to put the device into a certain desired state. For example, set the telemetry send interval to 30 minutes.
 - Cloud-to-device messages for one-way notifications to the device app.
- In this lab you will learn how to send **cloud-to-device messages** from your **IoT Hub** to your **device** kit using the **Azure Portal**, and how to implement custom functions activated by specific messages (for example, to turn on/off an LED on the IoT Dk board)

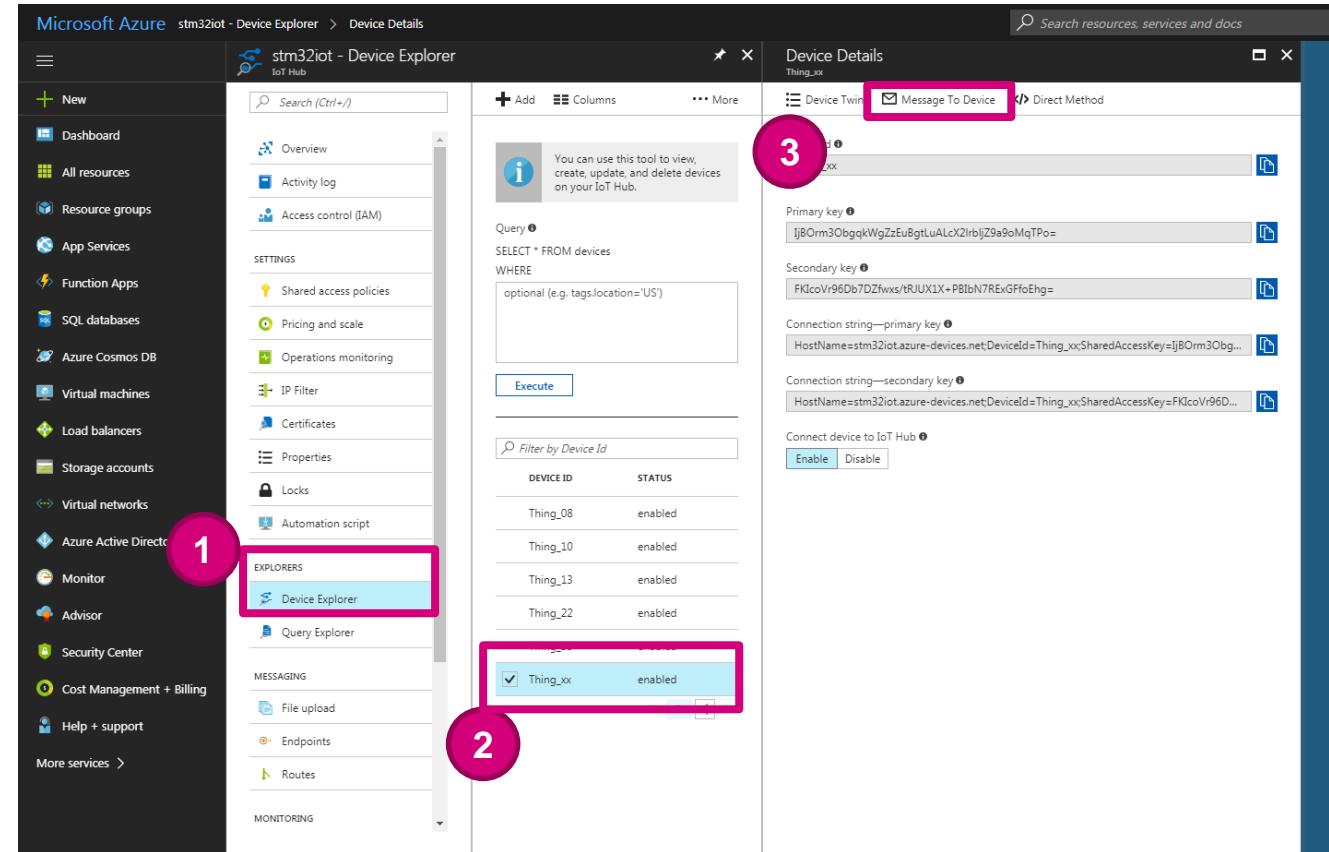
Step 1: Send cloud-to-device message

156

1 In the Azure portal, go to your IoT Hub **Device Explorer**

2 Select you Device

3 Click on Message to Device



Step 1: Send cloud-to-device message

157

Turn LED On

- 1 Copy-paste the message below to the Message Body field to turn the green LED on:

- {"Name": "LedOn", "Parameters": {}}

- 2 Click on **Send**



Device Details Thing_xx

Device Id: Thing_xx

Primary key: Ij8Orm3ObgqkWgZzEuBgtLuAlcX2lrbj...

Secondary key: FKlcovr96Db7DZfwxs/tRUX1X+PB...

Connection string—primary key: HostName=stm32iot.azure-devices.net...

Connection string—secondary key: HostName=stm32iot.azure-devices.net...

Connect device to IoT Hub: Enable

Message To Device Thing_xx

Send Message

Message Body: {"Name": "LedOn", "Parameters": {}}

Properties

Key: Value: Add Property

KEY VALUE

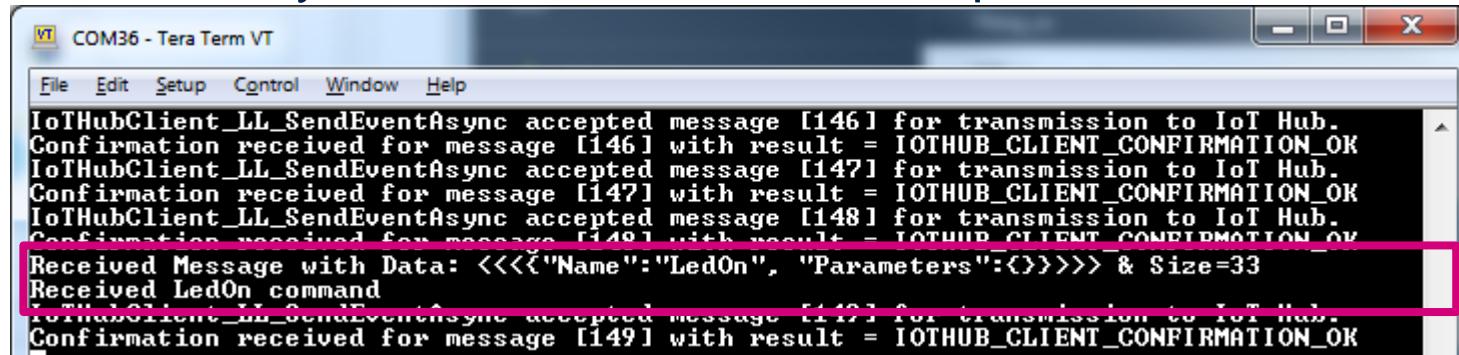
No results

Message Sent
Message to "Thing_xx" has been successfully sent.

Step 2: Visualize and Interpret Message received by device

158

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



The screenshot shows a window titled "COM36 - Tera Term VT". The menu bar includes File, Edit, Setup, Control, Window, and Help. The main window displays a series of log entries from an IoT Hub Client. The entries show messages being accepted for transmission to the IoT Hub and confirmations received for those messages. A specific message is highlighted with a pink rectangle:

```
IoTHubClient_LL_SendEventAsync accepted message [146] for transmission to IoT Hub.  
Confirmation received for message [146] with result = IOTHUB_CLIENT_CONFIRMATION_OK  
IoTHubClient_LL_SendEventAsync accepted message [147] for transmission to IoT Hub.  
Confirmation received for message [147] with result = IOTHUB_CLIENT_CONFIRMATION_OK  
IoTHubClient_LL_SendEventAsync accepted message [148] for transmission to IoT Hub.  
Confirmation received for message [148] with result = IOTHUB_CLIENT_CONFIRMATION_OK  
Received Message with Data: <<<("Name": "LedOn", "Parameters": {})>>> & Size=33  
Received LedOn command  
IoTHubClient_LL_SendEventAsync accepted message [149] for transmission to IoT Hub.  
Confirmation received for message [149] with result = IOTHUB_CLIENT_CONFIRMATION_OK
```

- Microsoft Azure IoT SDK provides means to define user specific callbacks (actions) associated to certain commands
- By default the application can interpret the following messages:
 - Pause → pause the application
 - Play → resume the application
 - LedOn/LedOff → turn on/off the green LED on IoT Dk board
 - LedBlink → the green LED on IoT Dk board will blink for each message received

Step 3: Send cloud-to-device message

159

Turn LED Off

- 1 Copy-paste the message below to the Message Body field to turn the green LED off:

- `{"Name": "LedOff", "Parameters": {}}`

- 2 Click on **Send**



Device Details Thing_xx

Device Id: Thing_xx

Primary key: IjB0rm3ObgqkWgZzEuBgLuALcX2lrbj...

Secondary key: FKlcovr96Db7DZfwxs/rJUX1X+...

Connection string—primary key: HostName=stm32iot.azure-devices.net;...

Connection string—secondary key: HostName=stm32iot.azure-devices.net;...

Connect device to IoT Hub: Enable

Message To Device Thing_xx

Send Message

You can use this tool to send messages to a device in your IoT Hub. Messages have both a body and optional properties organized as a collection of key/value string pairs.

Device Id: Thing_xx

Message Body: {"Name": "LedOff", "Parameters": {}}

Properties:

Add Property

KEY VALUE

No results

Message Sent
Message to "Thing_xx" has been successfully sent.

Step 4: Send cloud-to-device message

160

Blink LED

- 1 Copy-paste the message below to the Message Body field to blink the green LED:

- `{"Name": "LedBlink", "Parameters":{}}`

- 2 Click on **Send**



Device Details
Thing_xx

Device Id: Thing_xx

Primary key: `IjBOrm3ObgqkWgZzEu8gtLuALcX2rbJ...`

Secondary key: `FKlcoV96Db7DZfws/tRJUX1X+P...`

Connection string—primary key: `HostName=stm32iot.azure-devices.net...`

Connection string—secondary key: `HostName=stm32iot.azure-devices.net...`

Connect device to IoT Hub: `Enable`

Message To Device
Thing_xx

Send Message

Message Body: `{"Name": "LedBlink", "Parameters":{}}`

Properties

Key:

Value:

Add Property

KEY	VALUE
No results	

Message Sent
Message to "Thing_xx" has been successfully sent.

Customize Code to Execute Commands

- You can define a new action associated to a message, see example in `AzureClient_mqtt_DM_TM.c`

`WITH_ACTION(Pause),` ← Pause is the message and the name of the callback (action) that will be executed

- Define user specific implementation for the message. See example for Pause in `AzureClient_mqtt_DM_TM.c`

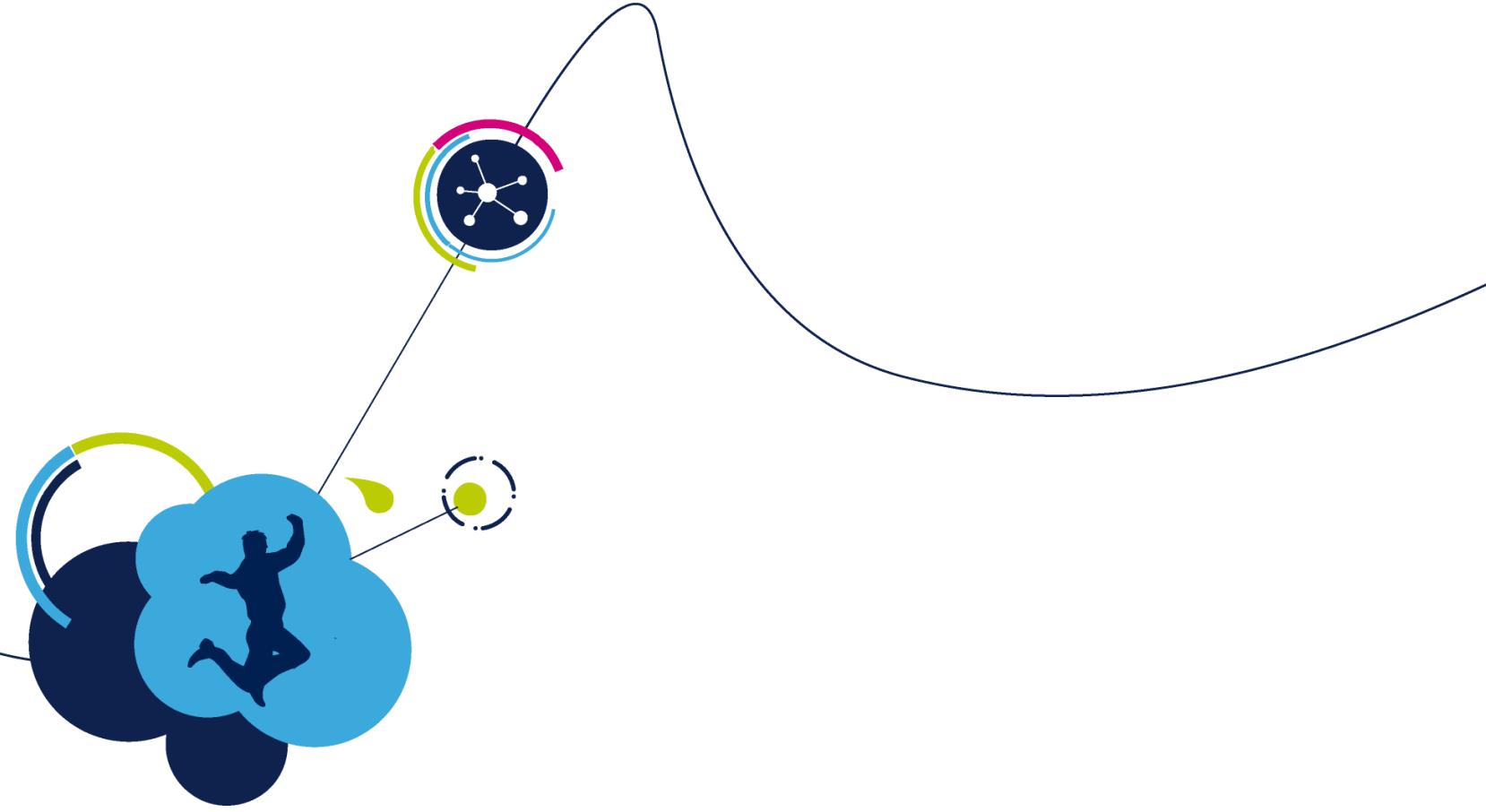
```
EXECUTE_COMMAND_RESULT Pause(Azure1_t *Azure1)
{
    (void)(Azure1);
    AZURE_PRINTF("Received Pause command\r\n");
    StopTimer1();
    return EXECUTE_COMMAND_SUCCESS;
}
```

- Function `ReceiveMessageCallback()` will be executed each time a message is received by the device. This function will take care of triggering the user specific action

```
AZURE_PRINTF("Received Message with Data: <<%.*s>> & Size=%d\r\n", (int)size, buffer, (int)size);

(void)memcpy(temp, buffer, size);
temp[size] = '\0';
executeCommandResult = EXECUTE_COMMAND(userContextCallback, temp);
```

← Trigger callback associated to message received

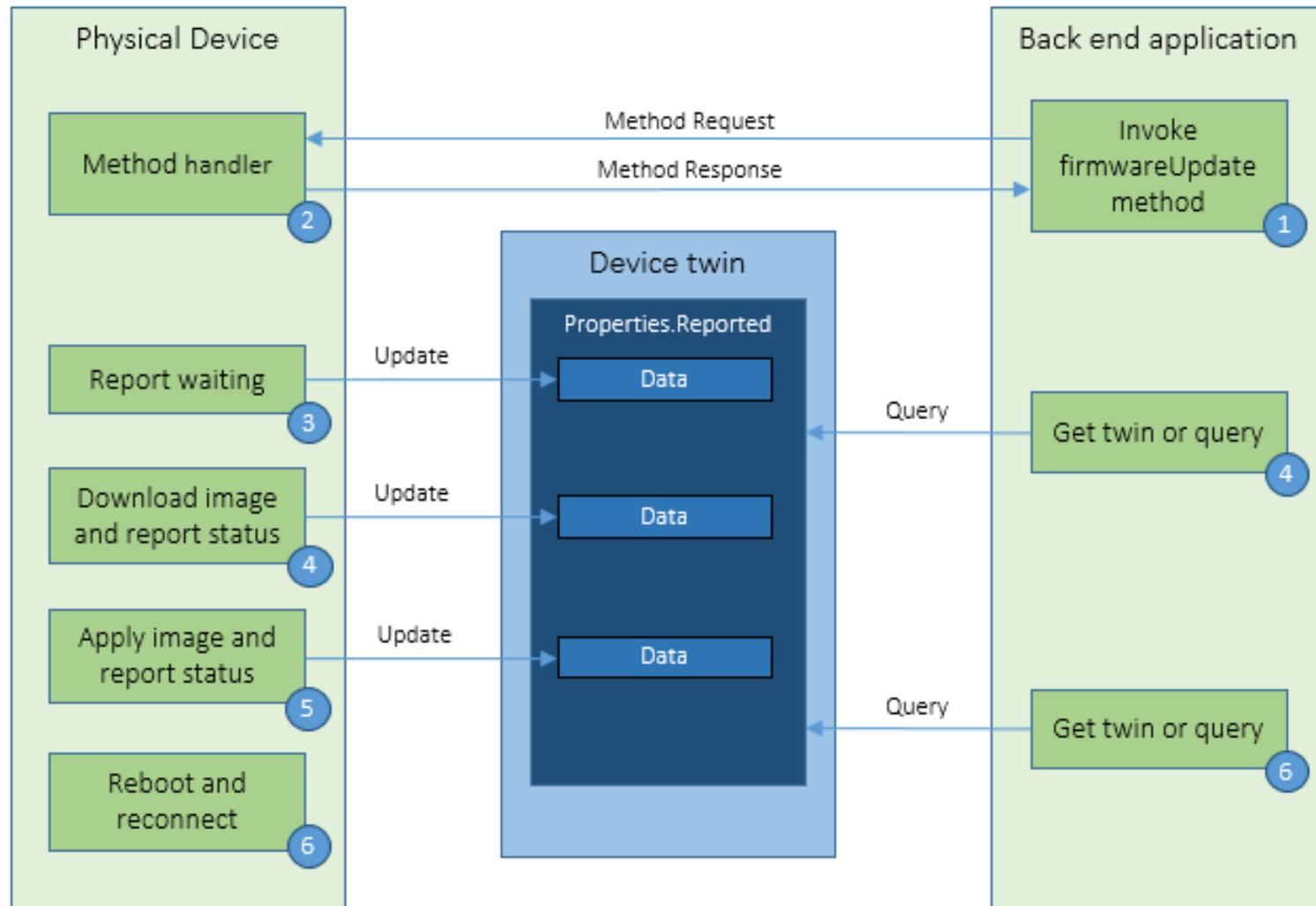


Lab 6: Firmware-over-the-air (FOTA) Update

- In this lab, we will:
 - Trigger a Firmware-Over-The-Air (FOTA) Update using a simple binary stored on the cloud.

Over-The-Air Firmware Update

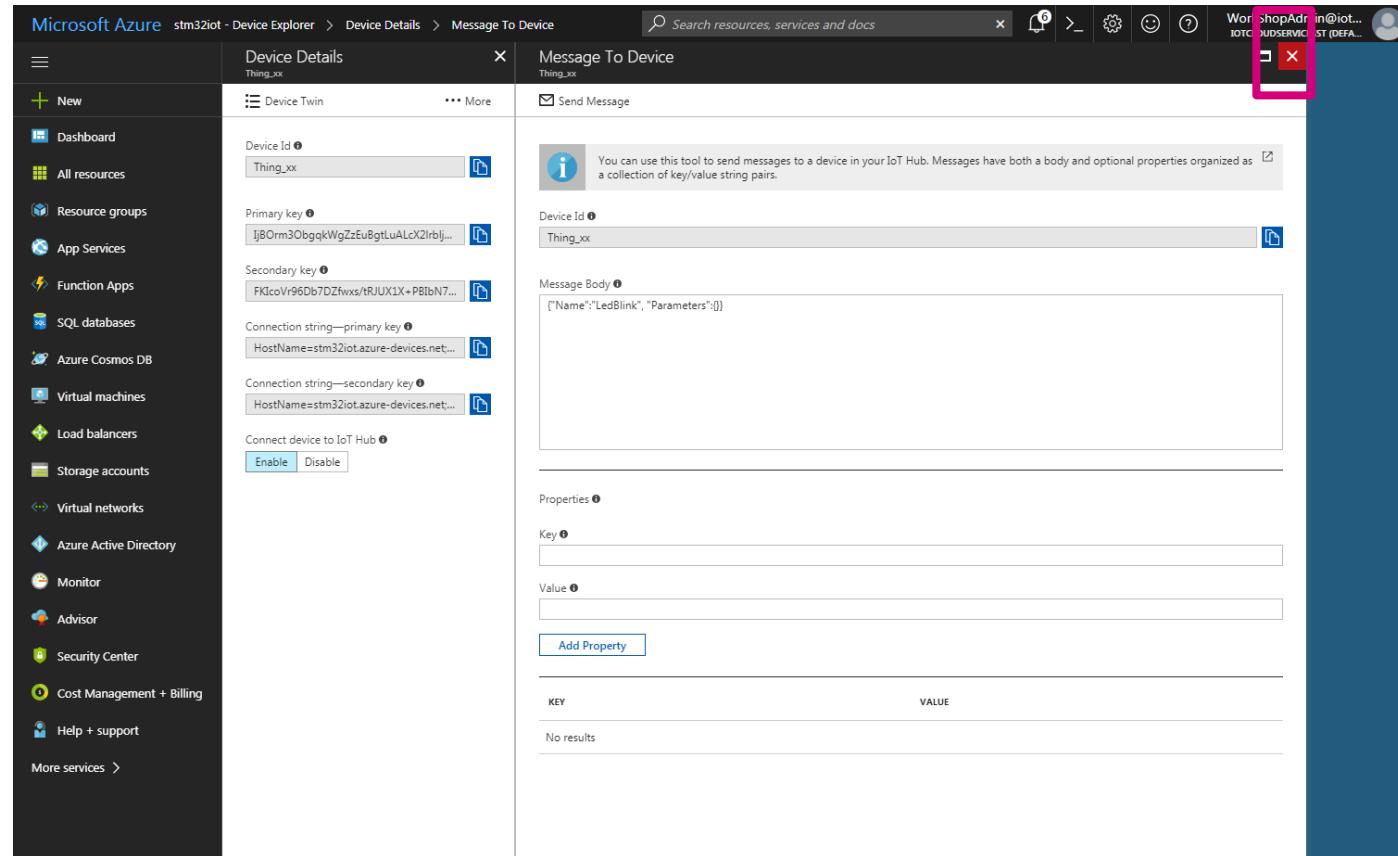
164



Step 2: Initiate Firmware Update method

165

1 Close the Message to Device window



Step 2: Initiate Firmware Update method

166

2 Click on Direct Method

The screenshot shows the Microsoft Azure Device Explorer interface for an IoT Hub named 'stm32iot'. The left sidebar lists various Azure services, and the main area shows a table of devices. A pink box highlights the 'Direct Method' button in the top right corner of the main panel, which is also circled with a pink circle containing the number '2'. The table lists several devices, with 'Thing_xx' selected.

DEVICE ID	STATUS
Thing_08	enabled
Thing_10	enabled
Thing_13	enabled
Thing_22	enabled
Thing_33	enabled
<input checked="" type="checkbox"/> Thing_xx	enabled

Step 2: Initiate Firmware Update method

167

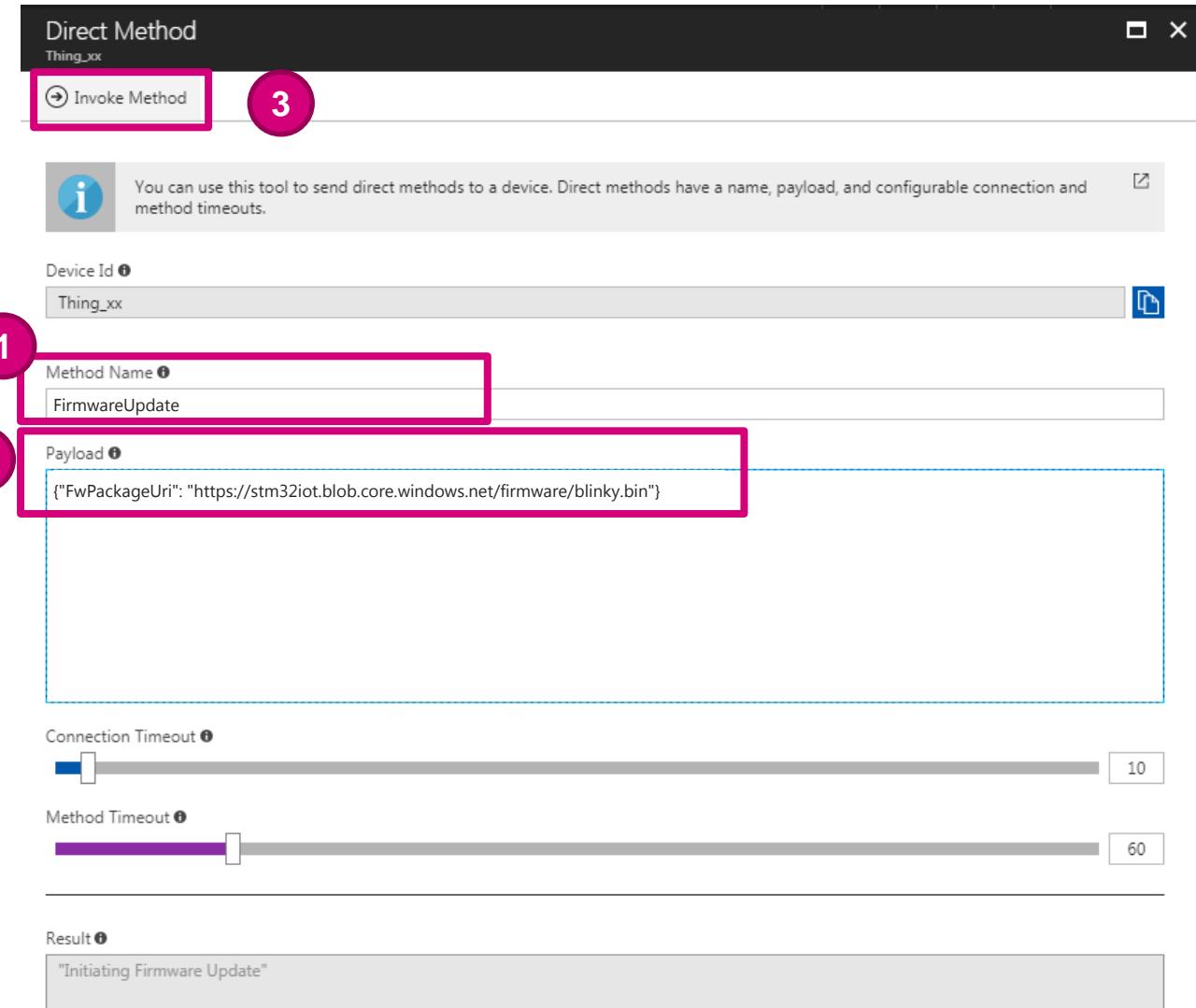
- 1 Copy-paste the message below to Method name

FirmwareUpdate

- 2 Copy-paste the message below to Method payload

```
{"FwPackageUri":  
"https://stm32iot.blob.core.windows.net/  
firmware/blinky.bin"}
```

- 3 Click on Invoke Method



Step 2: Monitor Firmware Update Execution on Device Via Serial Terminal

168

The screenshot shows a Windows application window titled "COM36 - Tera Term VT". The menu bar includes File, Edit, Setup, Control, Window, and Help. The main window displays a black text area with white text representing serial terminal output. The output shows a sequence of messages indicating the execution of a firmware update:

```
Confirmation received for message [16] with result = IOTHUB_CLIENT_CONFIRMATION_OK
Confirmation received for message [17] with result = IOTHUB_CLIENT_CONFIRMATION_OK
IoTHubClient_LL_SendEventAsync accepted message [19] for transmission to IoT Hub.
Received firmware update request. Use package at: [https://stm32iot.blob.core.windows.net/firmware/blinky.bin]
Channel 1 for Timer 1 stopped
download from: hostname=stm32iot.blob.core.windows.net type=secure port=443 file=/firmware/blinky.bin
Ok reported State [3]: Downloading
Confirmation received for message [18] with result = IOTHUB_CLIENT_CONFIRMATION_OK
Confirmation received for message [19] with result = IOTHUB_CLIENT_CONFIRMATION_OK
-->DeviceTwin CallBack [3]: Status_code = 204
Ok io_interface_description
Ok xio_create
Ok xio_setopt
onOpenCompleteFOTA
Ok xio_open
Ok xio_send HEAD Request
Content Length: Full OTA size 7272
Start FLASH Erase
End FLASH Erase 4 Pages of 2KB
Ok reported State [4]: Downloading
-->DeviceTwin CallBack [4]: Status_code = 204
Ok xio_send GET <000/007> Request
Ok xio_send GET <001/007> Request
Ok xio_send GET <002/007> Request
Ok xio_send GET <003/007> Request
Ok xio_send GET <004/007> Request
Ok xio_send GET <005/007> Request
Ok xio_send GET <006/007> Request
Ok xio_send GET <007/007> Request
OTA Update saved
OTA will be installed at next board reset
onCloseCompleteOTA callback
Ok xio_close
Ok xio_destroy
OTA Downloaded
Ok reported State [5]: DownloadComplete
-->DeviceTwin CallBack [5]: Status_code = 204
The Board will restart in for Applying the OTA
Ok reported State [6]: Applying
-->DeviceTwin CallBack [6]: Status_code = 204
Call to HAL_NVIC_SystemReset
```

A red rectangular box highlights the first two lines of the log, which indicate the start of a message transmission to the IoT Hub.

A second red rectangular box highlights the "Start FLASH Erase" and "End FLASH Erase" lines, which are part of the firmware update process.

A red arrow points from the text "When download is finished, the board will restart and load the new firmware" to the "The Board will restart in for Applying the OTA" line in the log.

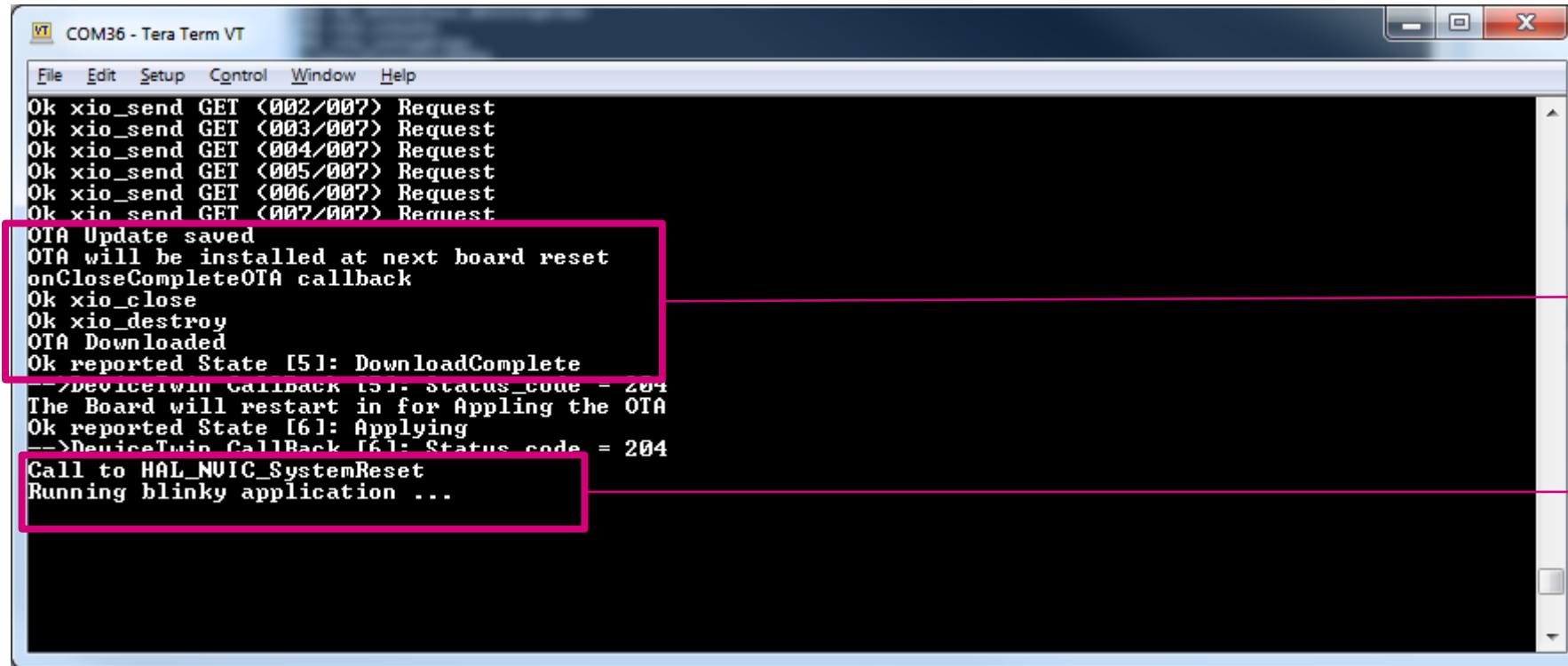
Erase FLASH
memory and
download new
firmware

When download is
finished, the board
will restart and load
the new firmware

Step 2: Monitor Firmware Update Execution on Device Via Serial Terminal (Cont'd)

169

When the download is completed, the board is reset and the new firmware is installed.



The screenshot shows a window titled "COM36 - Tera Term VT". The terminal window displays the following text:

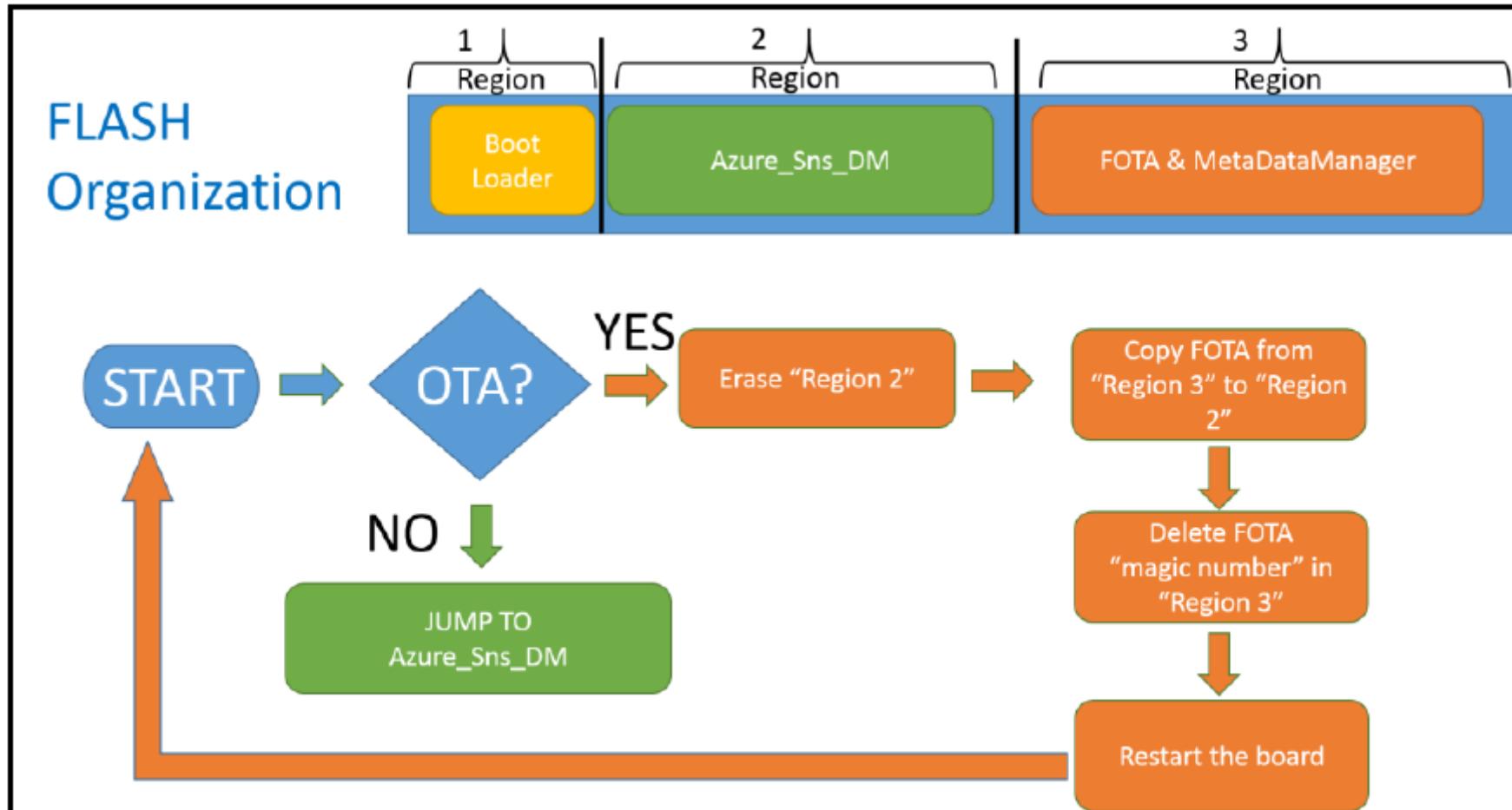
```
Ok xio_send GET <002/007> Request
Ok xio_send GET <003/007> Request
Ok xio_send GET <004/007> Request
Ok xio_send GET <005/007> Request
Ok xio_send GET <006/007> Request
Ok xio_send GET <007/007> Request
OTA Update saved
OTA will be installed at next board reset
onCloseCompleteOTA callback
Ok xio_close
Ok xio_destroy
OTA Downloaded
Ok reported State [5]: DownloadComplete
-->DeviceTwin_Callback [5]: status_code = 204
The Board will restart in for Applying the OTA
Ok reported State [6]: Applying
-->DeviceTwin_Callback [6]: status_code = 204
Call to HAL_NVIC_SystemReset
Running blinky application ...
```

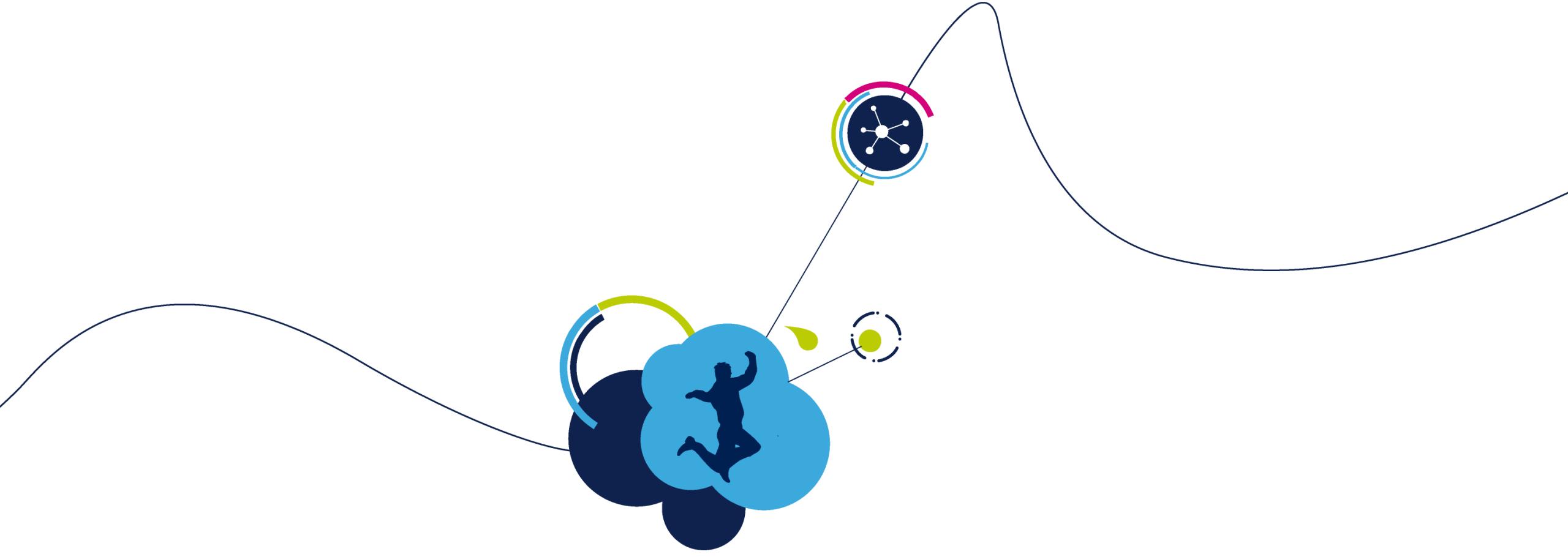
Two sections of the text are highlighted with pink boxes and connected by arrows to annotations:

- The first highlighted section contains the lines:
OTA Update saved
OTA will be installed at next board reset
onCloseCompleteOTA callback
Ok xio_close
Ok xio_destroy
OTA Downloaded
Ok reported State [5]: DownloadComplete
-->DeviceTwin_Callback [5]: status_code = 204
The Board will restart in for Applying the OTA
Ok reported State [6]: Applying
-->DeviceTwin_Callback [6]: status_code = 204
Call to HAL_NVIC_SystemReset
Running blinky application ...
- An arrow points from the right side of the first highlighted section to the text "Download is finished".
- An arrow points from the right side of the second highlighted section to the text "The board restarts and loads the new firmware".

STM32 boot sequence after a Firmware Update

170





Bluetooth® Low Energy Overview

What is Bluetooth® Low Energy?

- Bluetooth® Low Energy technology

- Short range wireless ISM 2.4 GHz
- Optimized for ultra low power
 - <15 mA peak current
 - <50 uA average current
- Fast connection procedure
- Client server architecture
- Low data throughput application
- Security including privacy/authentication/authorization
 - Based on encryption AES128
- Master Role : Central Device (Scanning, Initiating Connection)
- Slave Role : Peripheral Device (Advertising)



Bluetooth® Low Energy Branding

173

2011 Two flavors



- Ultra low power consumption being a pure low energy implementation
- Months to years of lifetime on a standard coin cell battery



- Classic Bluetooth® + Bluetooth® low energy on a single chip
- These are the hub devices of the Bluetooth® ecosystem

2017 Back to one flavor

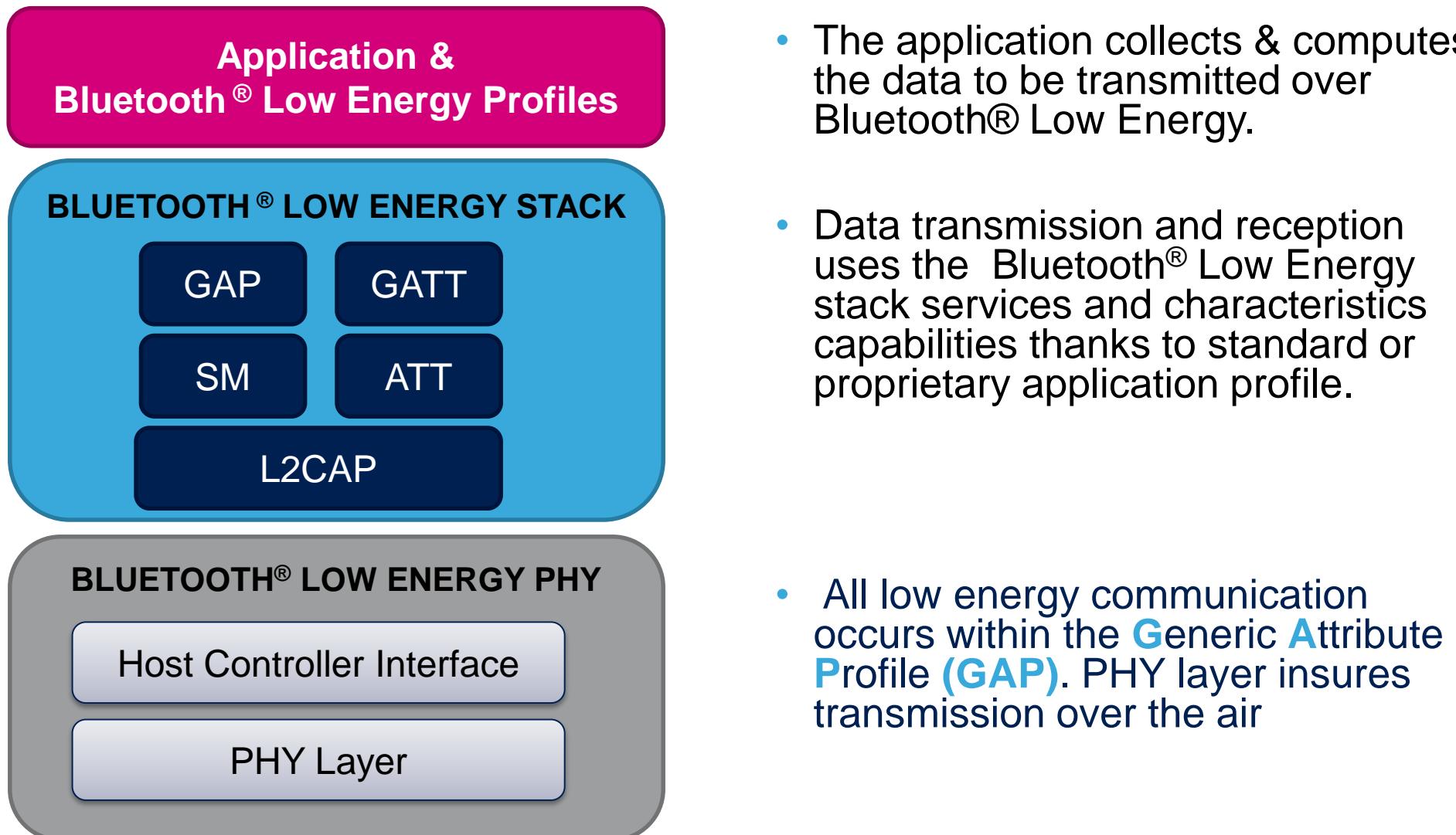


- An implementation of the Bluetooth® core system has only one Primary Controller which may be one of the following configurations:
 - BR/EDR Controller (3.0 and earlier)
 - LE (low energy) Controller (4.0 and newer)
 - Combined BR/EDR Controller portion and LE controller portion into a single Controller (4.0 and newer)

Source: Bluetooth® SIG

Bluetooth® Low Energy Stack Partitioning

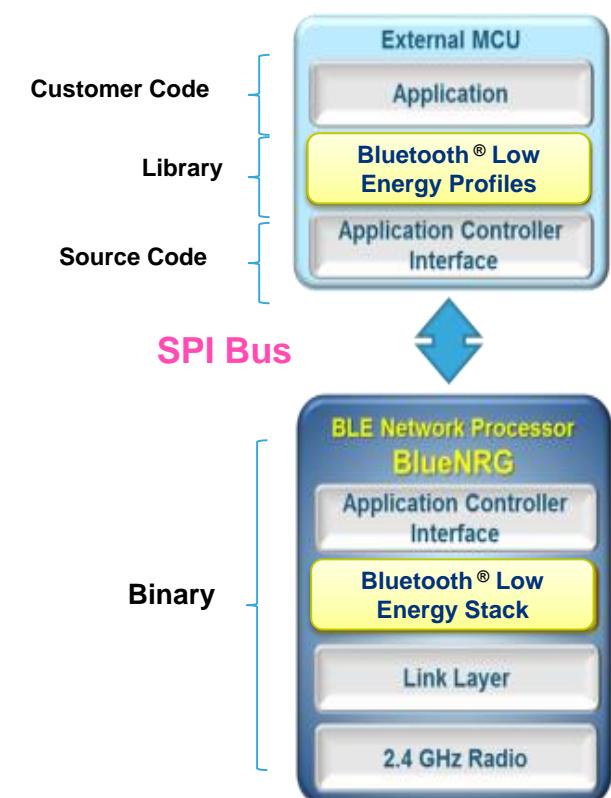
174

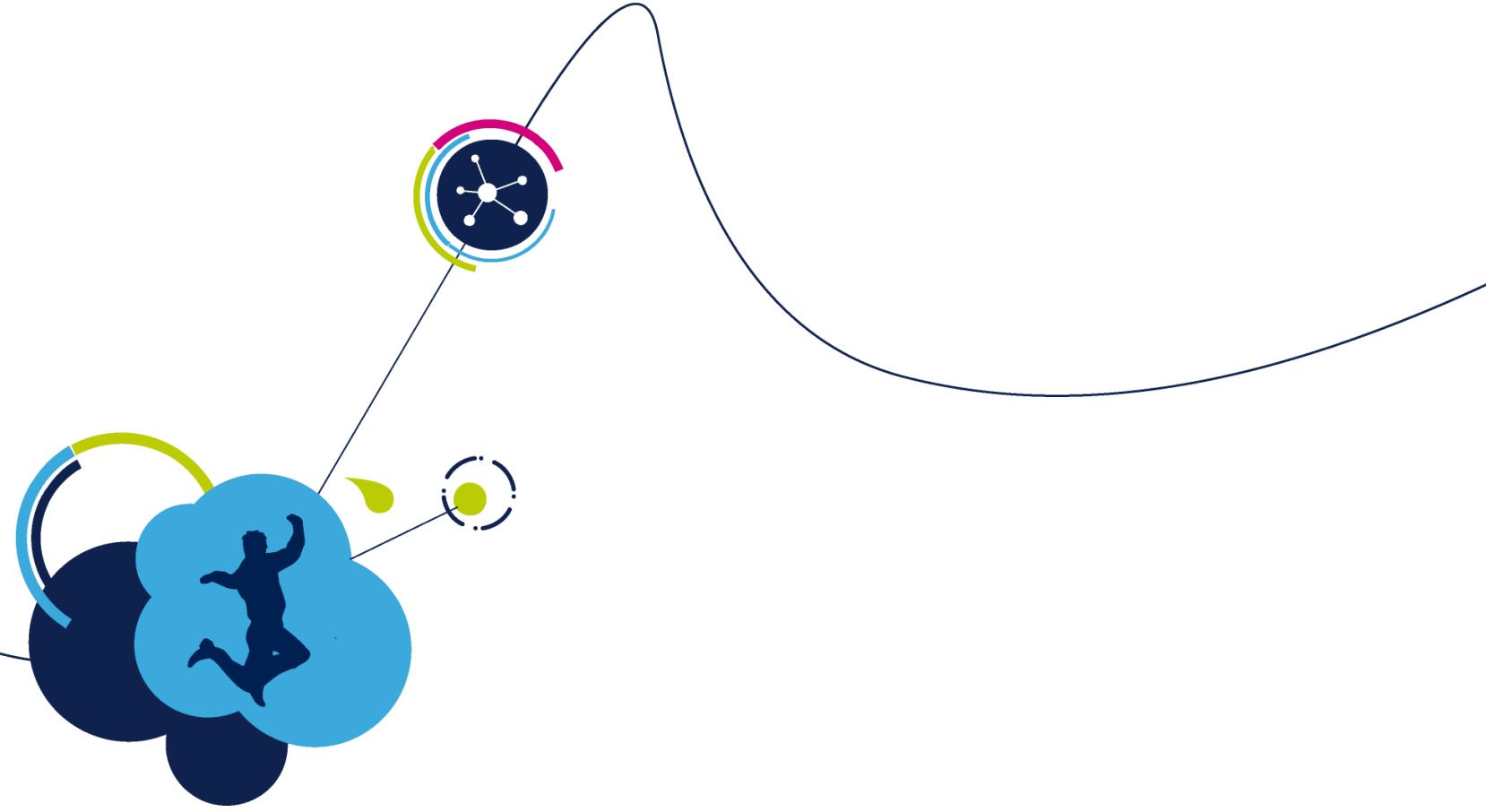


ST BlueNRG-MS Solution



- Single mode Bluetooth® Low Energy wireless network processor
 - 2.4GHz RF transceiver
 - Cortex-M0 microcontroller (running the BT MS stack)
 - AES 128-bit co-processor
 - Master and Slave Mode Bluetooth® Low Energy (4.1) Network Processor.
 - On chip non-volatile Flash memory allows OTA stack upgrade.
 - $I_{CC,RX}$ 7.3mA
 - $I_{CC,TX}$ 8.2mA @ 0 dBm
 - $I_{CC,Sleep}$ 1.7 μ A
 - $I_{CC,Shutdown}$ 2.5nA
- + STM32 Consumption & Size





Demo: Bluetooth® Low Energy Pairing

- This lab will ensure that your BlueNRG device has a unique name and MAC address.
- This lab will demonstrate how to configure a BlueNRG device, communicate with a smartphone and display heart rate data.

- The DK IoT Node will be used as server while the applet is a client.
- You will need to download the **STM32 BLE Profiles** application available on the Apple App store or the Android Google Play store.



STM32 BLE Profiles

STMICROELECTRONICS Libraries & Demo

Everyone

This app is compatible with all of your devices.



STM32 BLE Profiles

By STMICROELECTRONICS INC

Open iTunes to buy and download apps.



[View in iTunes](#)

Description

The STM32 BLE Profiles App is a companion tool to show in human readable form the Bluetooth Low Energy (BLE) devices implementing peripheral profiles. It supports X-NUCLEO-IDB04A1 /X-NUCLEO-IDB05A1 BlueNRG expansion boards running on STM32 Nucleo boards.

[STMICROELECTRONICS INC Web Site](#) [STM32 BLE Profiles Support](#)

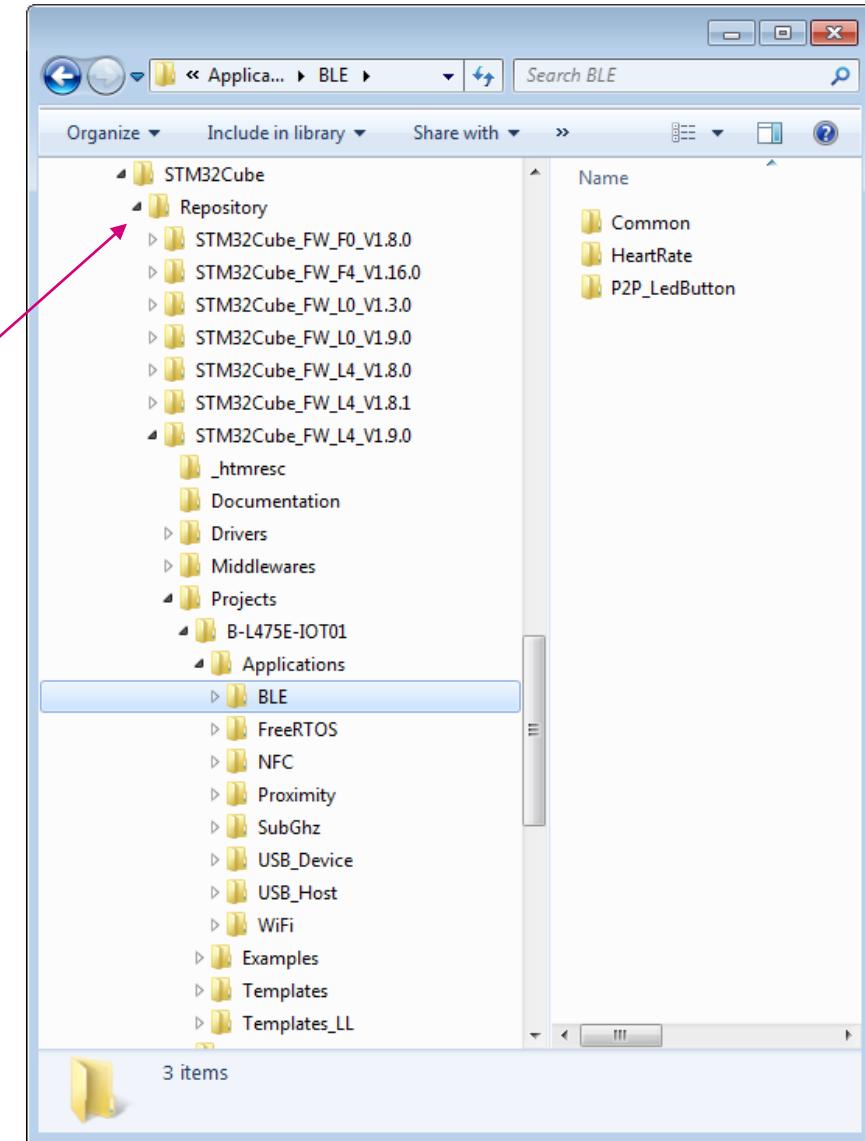
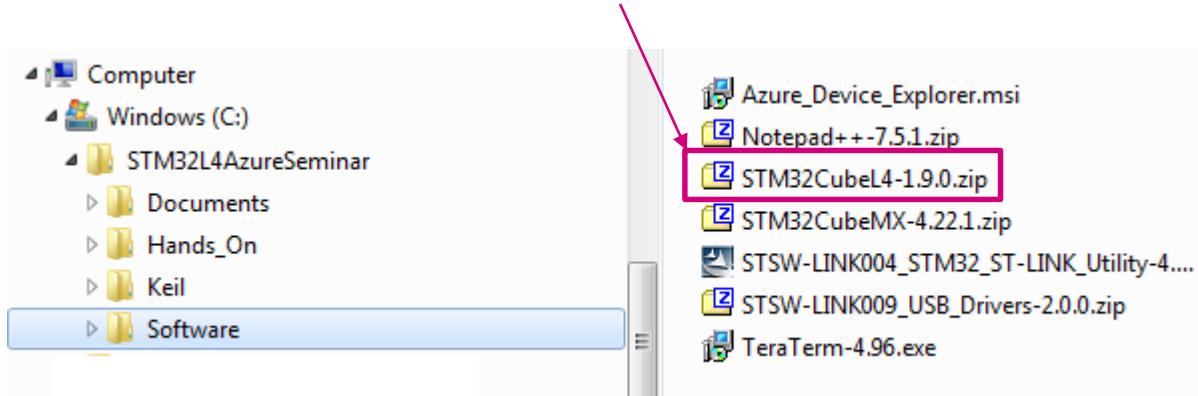
What's New in Version 2.0

- New app name STM32 BLE Profiles

STM32CubeL4 Examples Package

179

- One of the packages installed on your PC is the STM32CubeL4
- This package includes several applications made for the STM32L4 DK IoT node board
- The zip file is extracted within your STM32Cube repository folder
- The zip file is installed by the Seminar Installer at C:\STM32L4AzureSeminar\Software\



Open the BlueNRG_HandsOn Project

180

- We are going to configure the **BlueNRG_HandsOn** program to give each BlueNRG module a unique MAC address and Unique device name. The device name will be used later to identify your board within the **ST BLE Profiles** app.
1. Close the previous Keil MDK-ARM project.
 2. Double click on the **Project.uvprojx** file located here:

`...\\STM32Cube\\Repository\\STM32Cube_FW_L4_V1.9.0\\Projects\\B-L475E-IOT01\\Applications\\BLE\\HeartRate\\MDK-ARM\\`

Note: Your STM32Cube Repository folder path can be found by launching STM32CubeMX, and checking Help > Updater Settings



BlueNRG Module configuration (1/4)

181

Expand the file tree:

1. Expand **Application/Common/ble_services**
2. Double-click **hrs.c**
3. Right-click **config.h** on line 48
4. Select **Open document “config.h”**

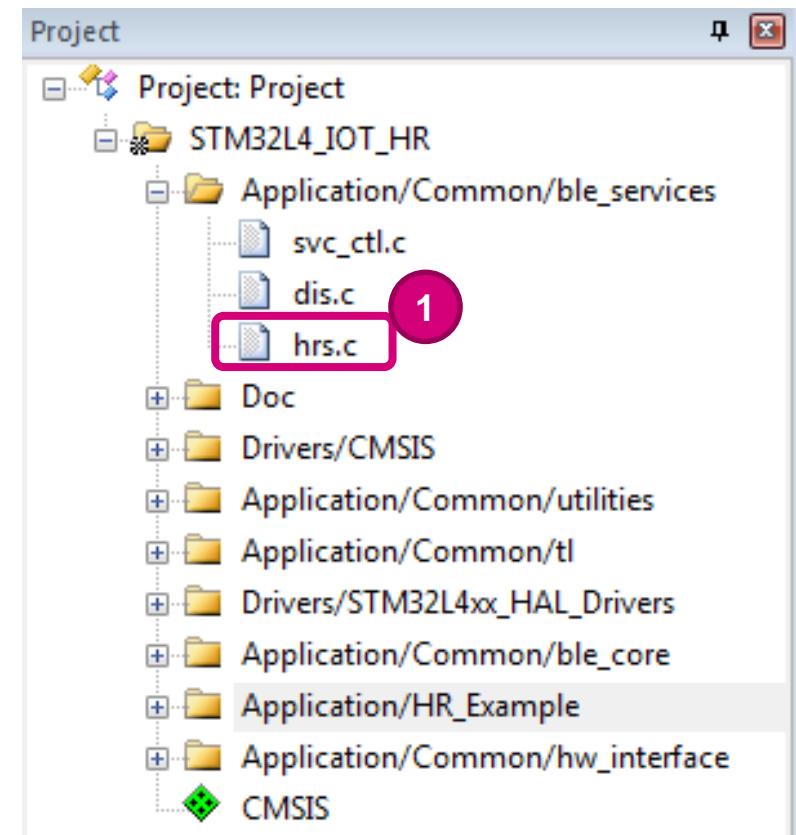
The screenshot shows a code editor with the following code snippet:

```
47
48 #include "config.h" 2
49 #include "debug.h"
50
51 #include "svc_privat
52 #include "ble_lib.h"
53 #include "svc_ctl.h"
54 #include "hrs.h"
```

A context menu is open at line 48, specifically over the `#include "config.h"` directive. The menu items are:

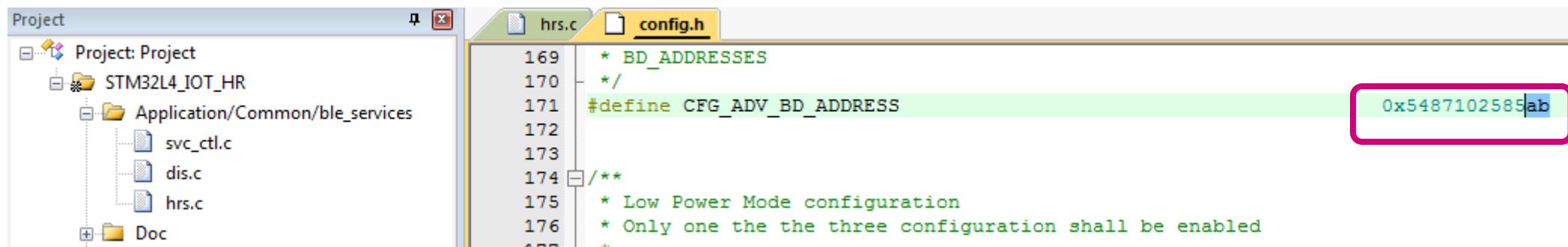
- Split Window horizontally
- Open document "config.h" 3
- Insert '#include file'
- Go to Headerfile "hrs.h"

Numbered circles indicate the steps: 2 is on the `#include "config.h"` line, and 3 is on the **Open document "config.h"** menu item.



BlueNRG Module configuration (2/4)

182



The screenshot shows a code editor interface with a project tree on the left and a code editor window on the right. The project tree shows a project named 'Project' with a folder 'STM32L4_IOT_HR' containing 'Application/Common/ble_services' which includes files 'svc_ctl.c', 'dis.c', and 'hrs.c'. The code editor window has tabs for 'hrs.c' and 'config.h', with 'config.h' being the active tab. The code in 'config.h' is as follows:

```
169 * BD_ADDRESSES
170 */
171 #define CFG_ADV_BD_ADDRESS 0x5487102585ab
172
173
174 /**
175 * Low Power Mode configuration
176 * Only one the three configuration shall be enabled
177 ...
```

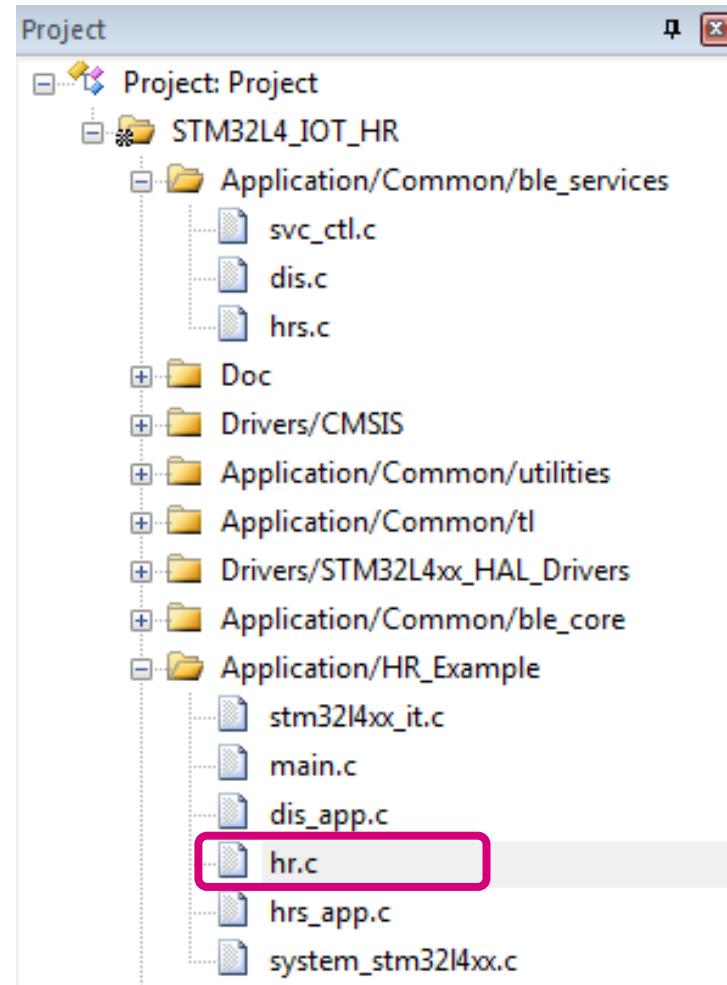
A pink rectangular box highlights the line '#define CFG_ADV_BD_ADDRESS 0x5487102585ab'. The value '0x5487102585ab' is also highlighted with a pink border.

- Replace the 'ab' in the **CFG_ADV_BD_ADDRESS** (line 171) with your participant number found on your box (2 characters).

BlueNRG Module configuration (3/4)

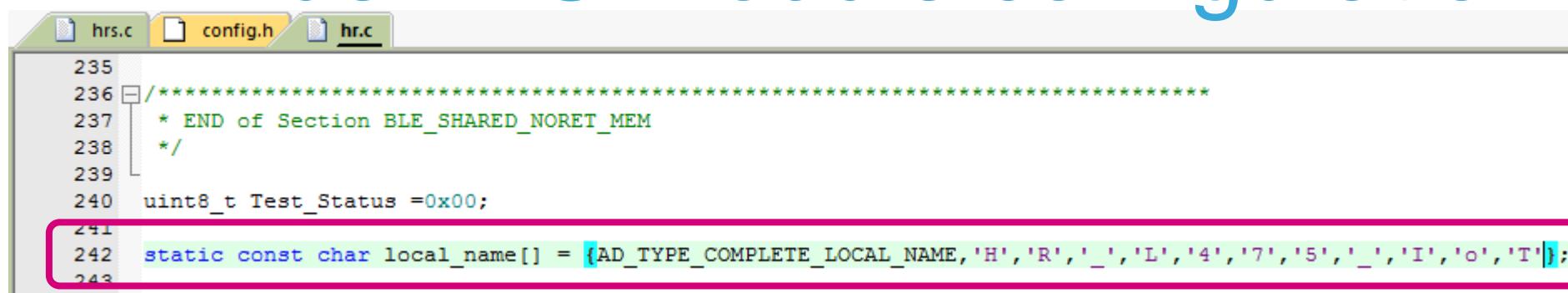
183

1. Expand **Application/HR_Example**
2. Double-click **hr.c**

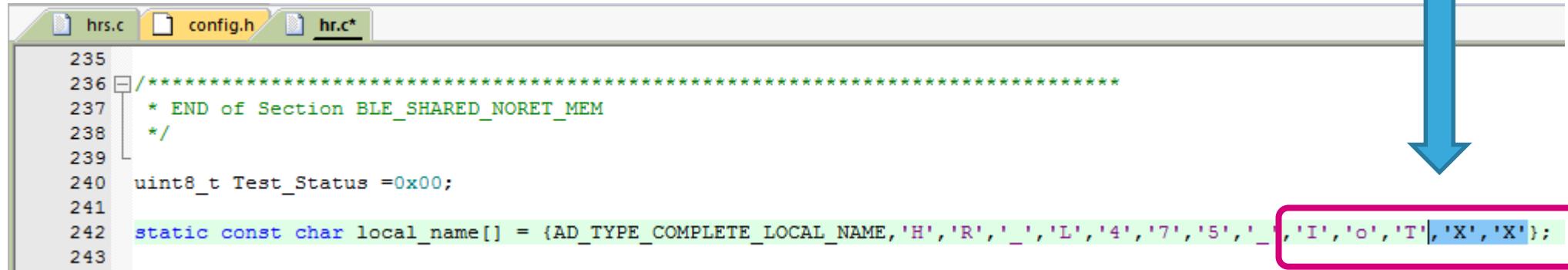


BlueNRG Module configuration (4/4)

184



```
hrs.c config.h hr.c
235
236 /* END of Section BLE_SHARED_NORET_MEM
237 */
238
239
240 uint8_t Test_Status =0x00;
241
242 static const char local_name[] = {AD_TYPE_COMPLETE_LOCAL_NAME,'H','R','_','L','4','7','5','_','I','o','T'};
243
```



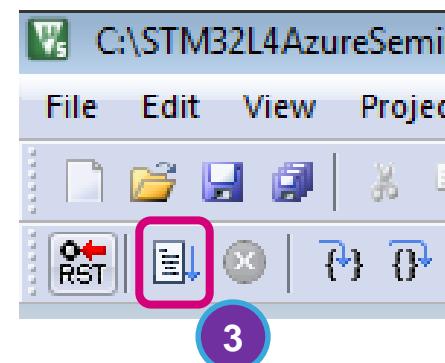
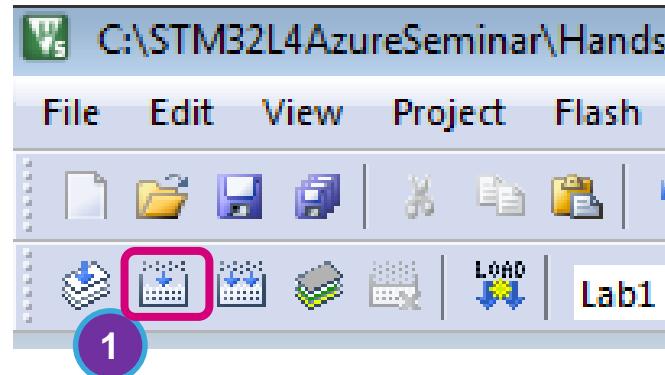
```
hrs.c config.h hr.c*
235
236 /* END of Section BLE_SHARED_NORET_MEM
237 */
238
239
240 uint8_t Test_Status =0x00;
241
242 static const char local_name[] = {AD_TYPE_COMPLETE_LOCAL_NAME,'H','R','_','L','4','7','5','_','I','o','T','X','X'};
243
```

- Add your participant number within single quotes in place of the 'x', 'x' in the `local_name` (line 242) table as shown above.

Load and Run

185

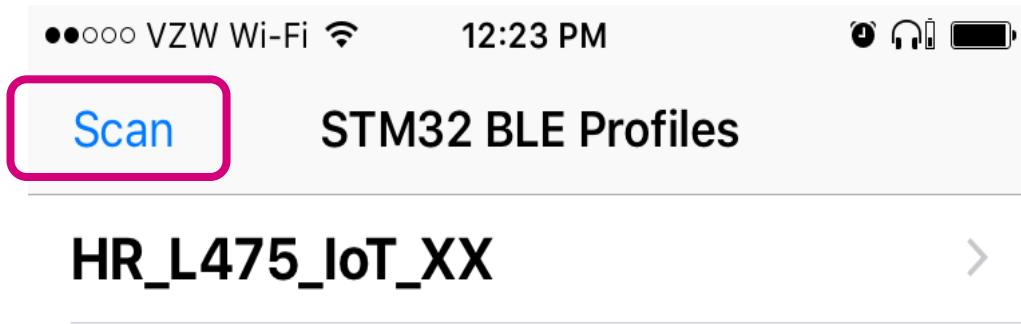
1. Click on the Build button or choose Project > Build Target
2. Click the Start/Stop Debug Session button
3. Click the Run button



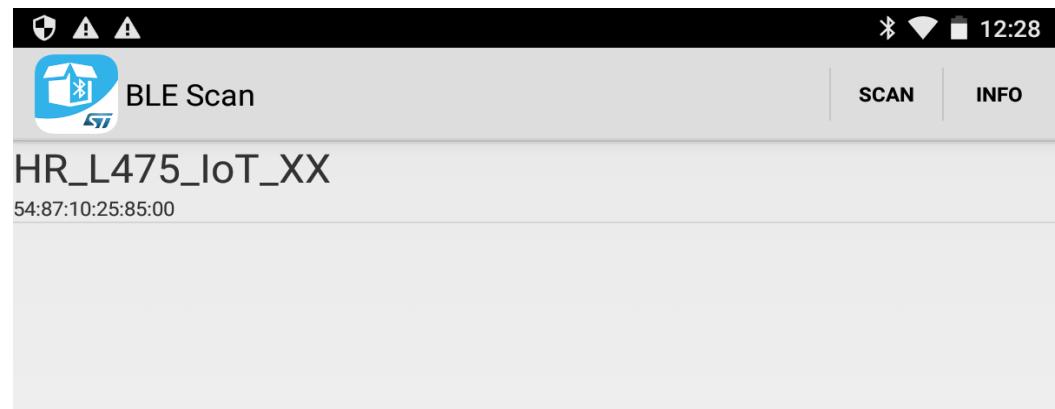
Pair with STM32 BLE Profiles App

186

iOS



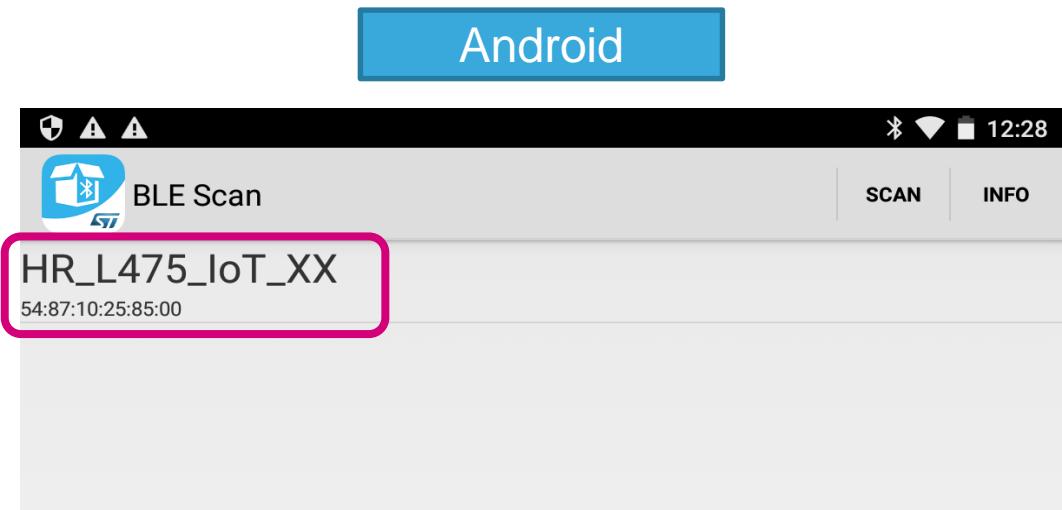
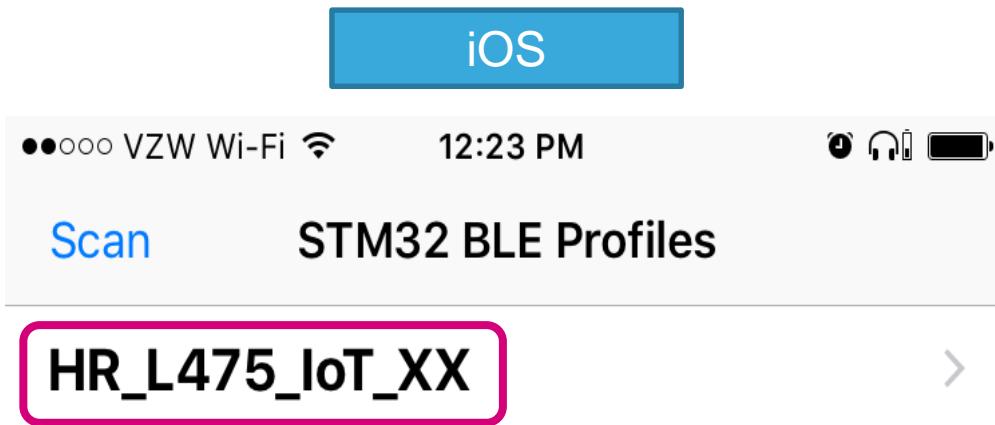
Android



- Make sure Bluetooth® is active on your phone
- Using your phone, open the **STM32 BLE Profiles** app.
- For iOS users click on **Scan**.

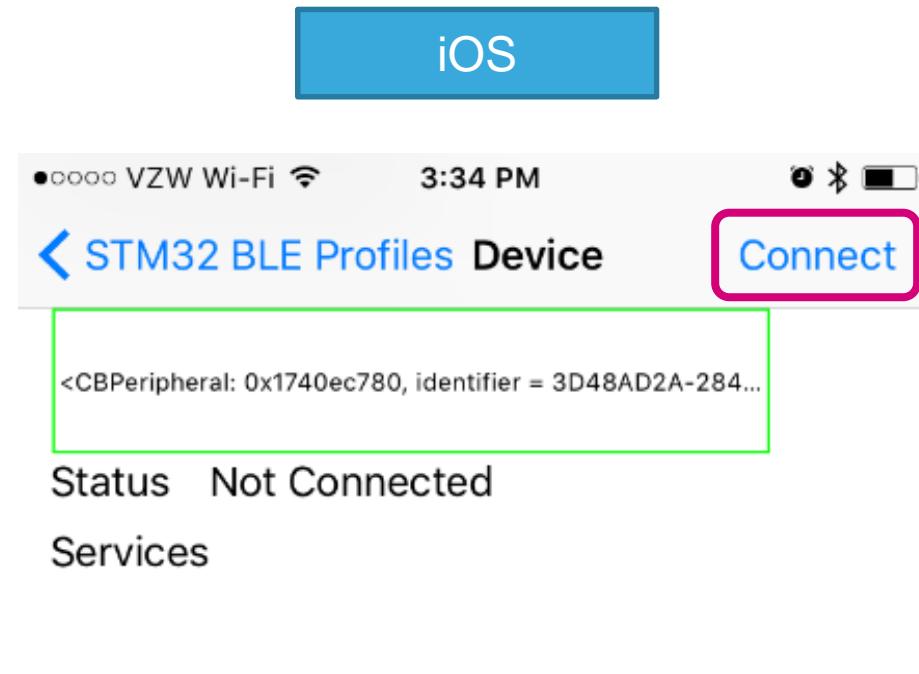
Pair with STM32 BLE Profiles App

187



- Identify your device with device name **HR_L475_IoT_XX**, (**XX** is the number you entered during the board configuration). Click on your device name.

Connect to your Device

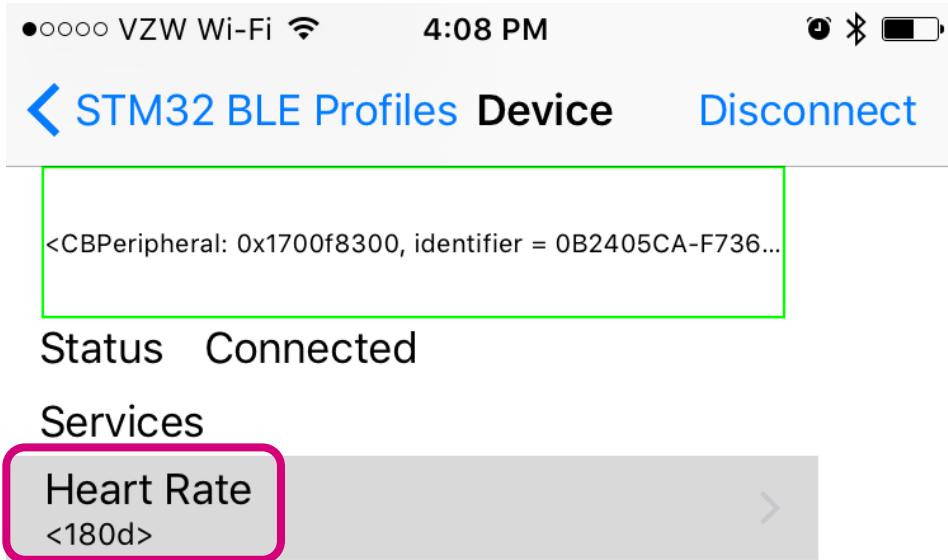


- iOS users, please click **Connect** on the next screen (iOS)

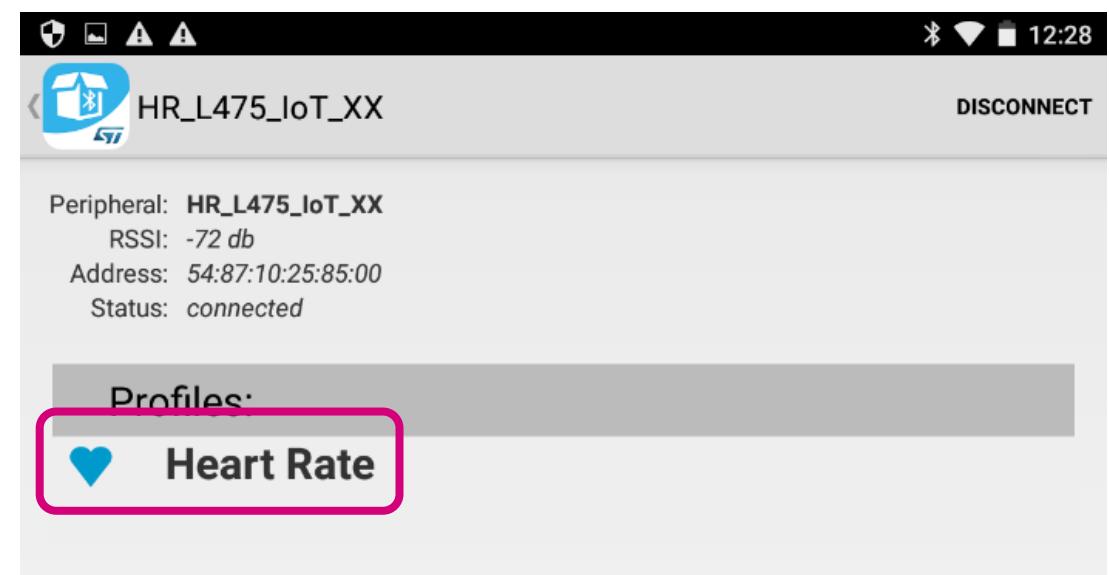
Select the Heart Rate Profile

189

iOS



Android

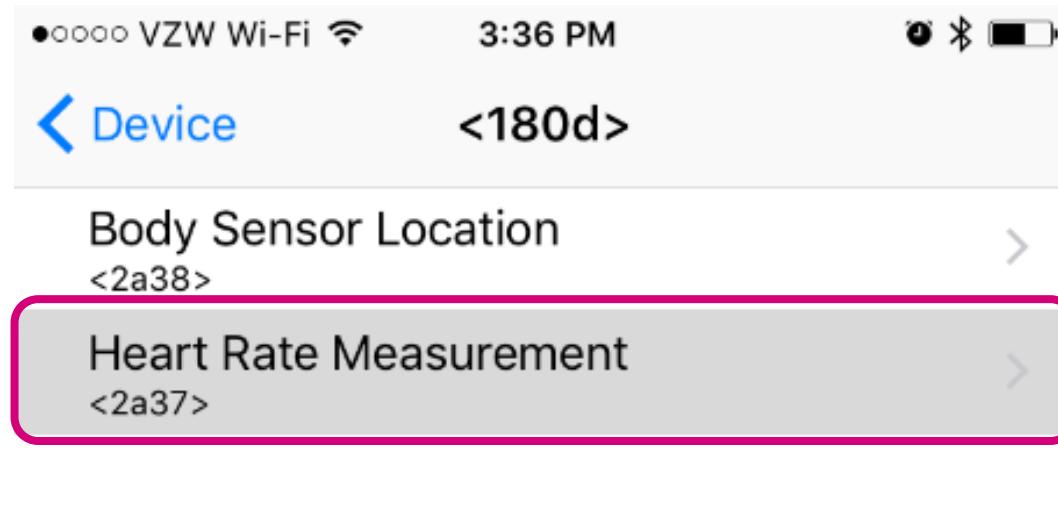


- Click on **Heart Rate** under **Services** (iOS) or **Profiles** (Android)

Select the Heart Rate Profile

190

iOS

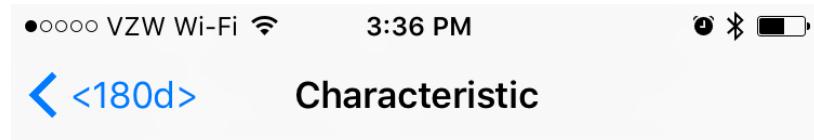


- iOS users, please click on **Heart Rate Measurement**

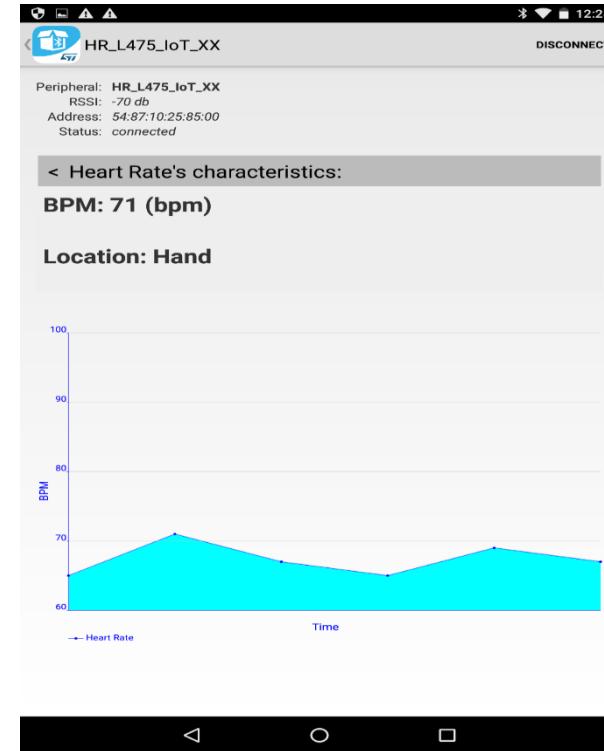
Display HR Data

191

iOS



Android



- You should see the simulated heart rate.

STM32L4 Discovery Kit IoT Node

Azure IoT Connectivity Summary

192

- STM32L4 MCU enables your IoT projects with the combination of ultra-low power and high performance
- STM32L4 Discovery Kit IoT is a compact, yet powerful board to explore various connectivity options including Microsoft Azure IoT
- Next Steps
 - You can add BLE and NFC features to the Azure project to create an application that can connect to the cloud and also has short range connectivity with your smartphone
 - Make your own custom web application by following the Azure guide linked earlier
 - Use the rich set of sensors on the board to get creative!
 - Post your projects or ideas on the ST Community website to gather feedback and get support: <https://community.st.com/community/share-your-activities/pages/overview>

Releasing Your Creativity

193



/STM32

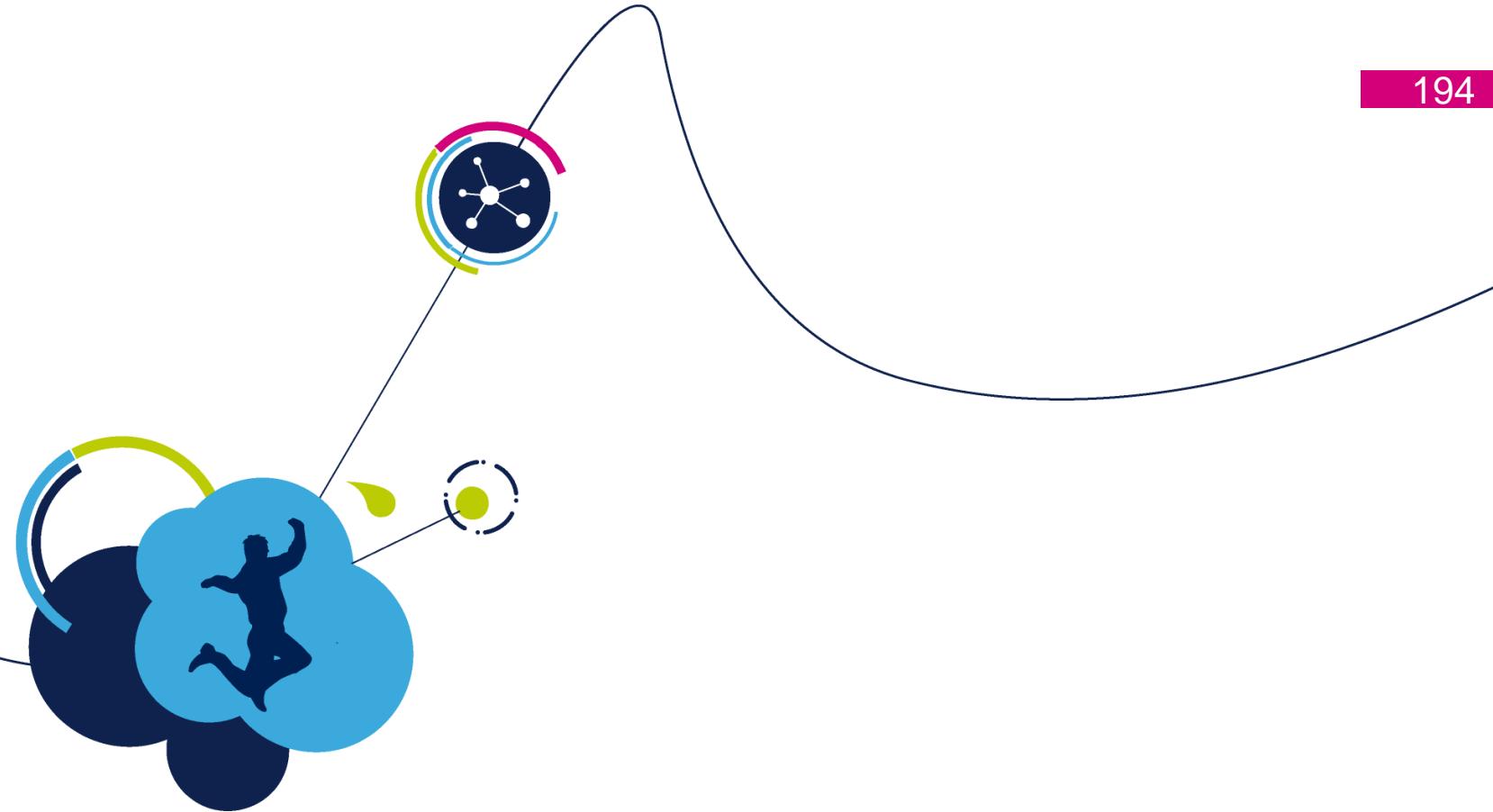


@ST_World



st.com/e2e

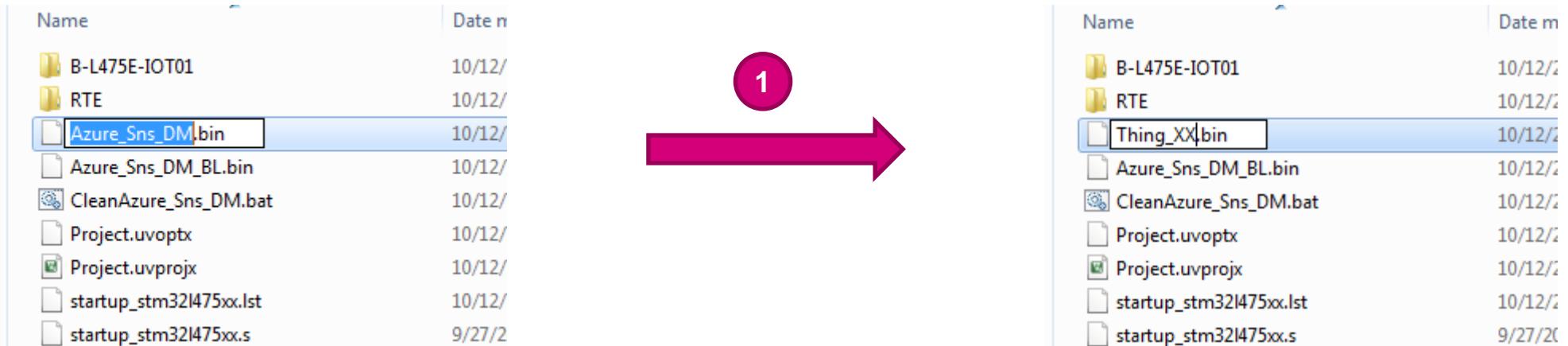




Appendix 1: Upload firmware to Azure storage

Step 1: Prepare Firmware binary for Upload to Azure Cloud Storage

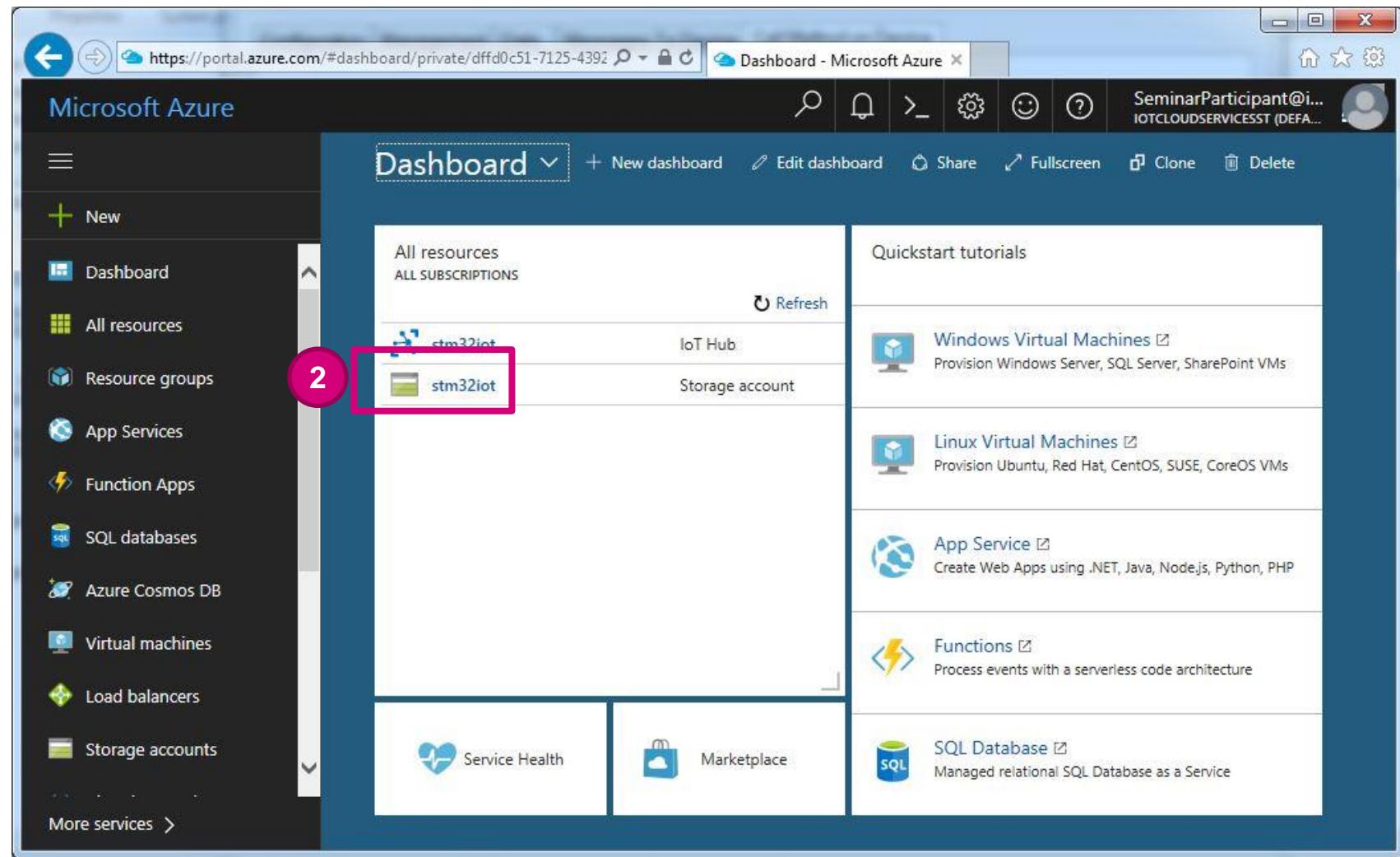
- 1 In your project directory, rename your **Azure_Sns_DM.bin** to **Thing_XX.bin** where **XX** is your participant number



Step 2: Upload firmware to Azure storage

196

- 2 Open the **stm32iot** Storage account from the Azure Portal Dashboard



Step 2: Upload firmware to Azure storage

197

3

Click on the firmware container

The screenshot shows the Microsoft Azure portal interface for a storage account named 'stm32iot'. The left sidebar lists various management options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. A large red circle with the number '3' highlights the 'Overview' link, which is currently selected and highlighted in blue. The main content area displays the storage account's details under the 'Essentials' section, including its resource group ('stm32iot'), status ('Primary: Available'), location ('West US'), and subscription information ('Pay-As-You-Go'). Below this, a search bar allows searching for containers by prefix. A table lists the contents of the 'firmware' container, with one item selected and highlighted in red. The table columns are NAME, LAST MODIFIED, PUBLIC ACCESS..., and LEASE STATE. The selected item has a checked checkbox next to it.

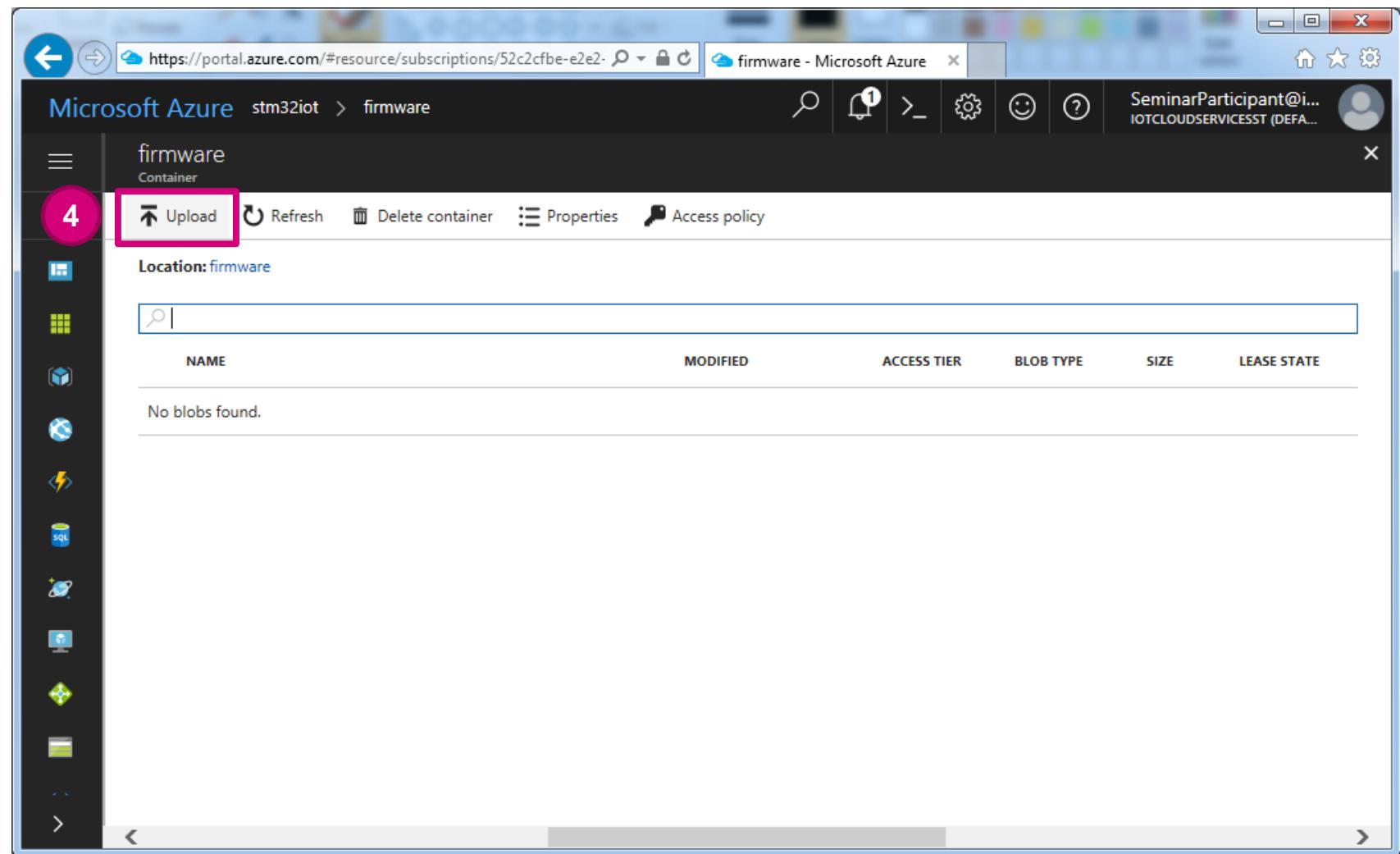
NAME	LAST MODIFIED	PUBLIC ACCESS...	LEASE STATE
firmware	8/29/2017 4:40:37 PM	Blob	Available

Step 2: Upload firmware to Azure storage

198

4

Click on Upload

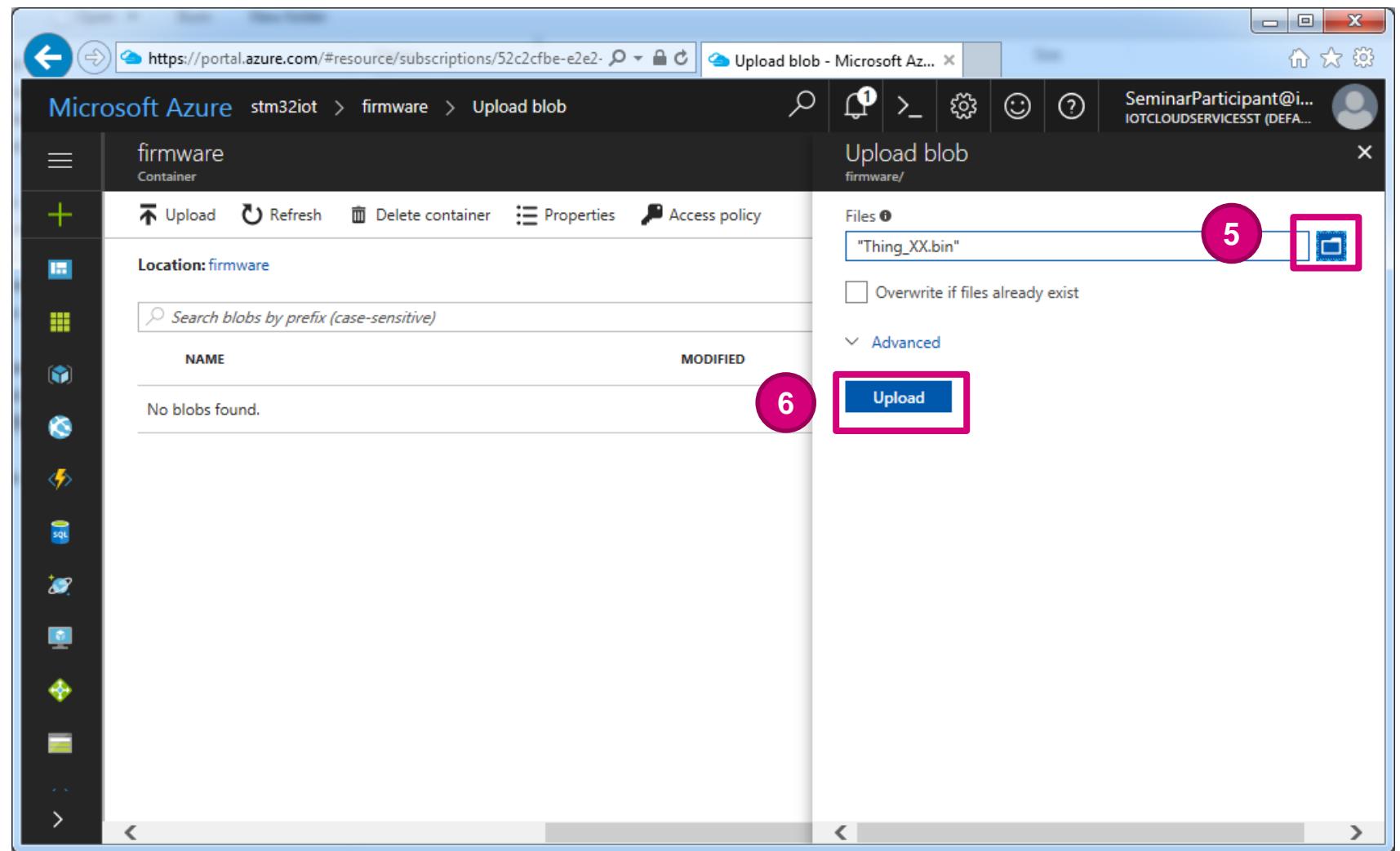


Step 2: Upload firmware to Azure storage

199

- 5 Browse to the location of your binary (located in your project folder)

- 6 Click on Upload

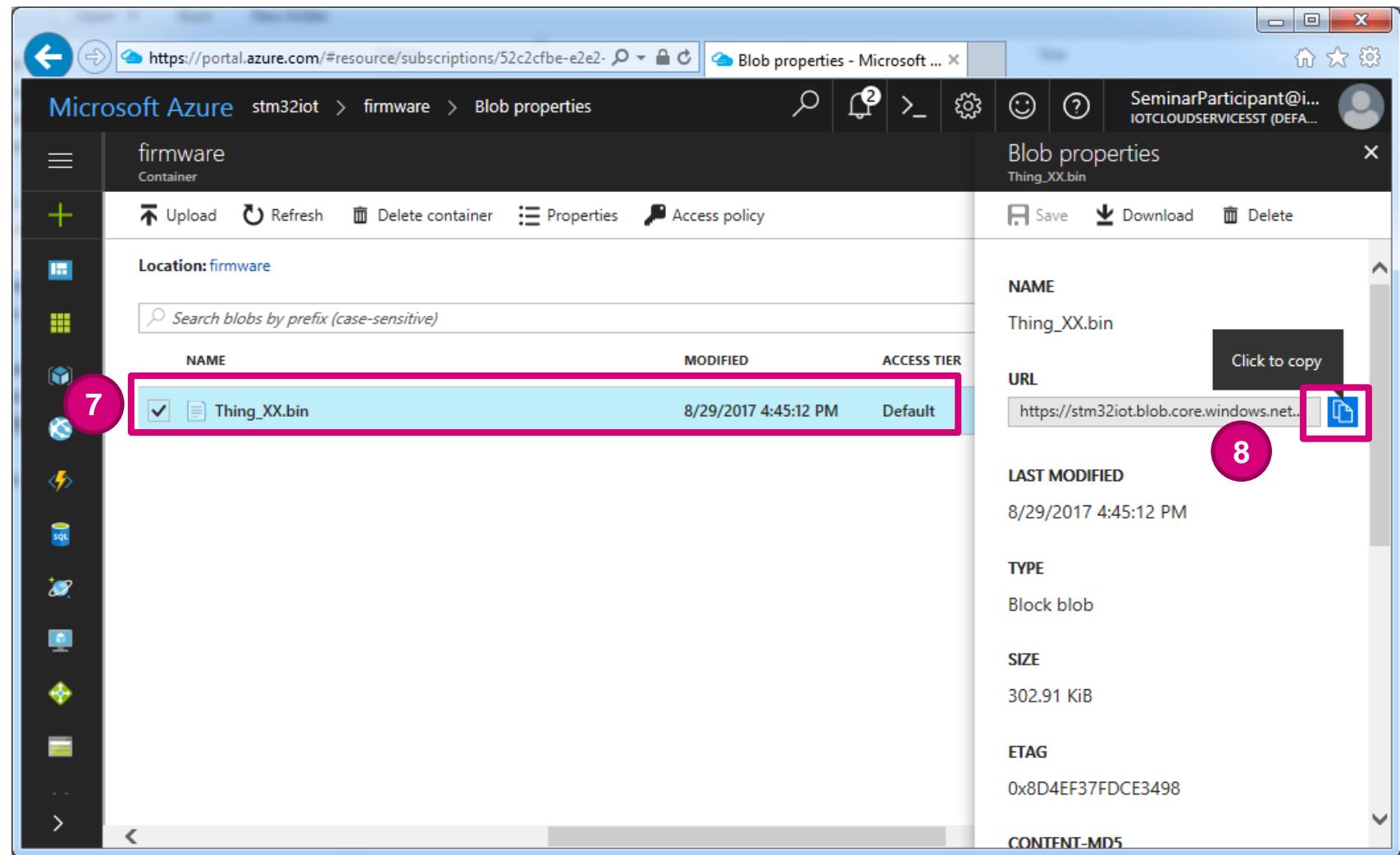


Step 3: Retrieve firmware URL

200

Once the binary has been uploaded:

- 7 Double-click on your Thing_XX.bin blob
- 8 Click to copy the blob URL



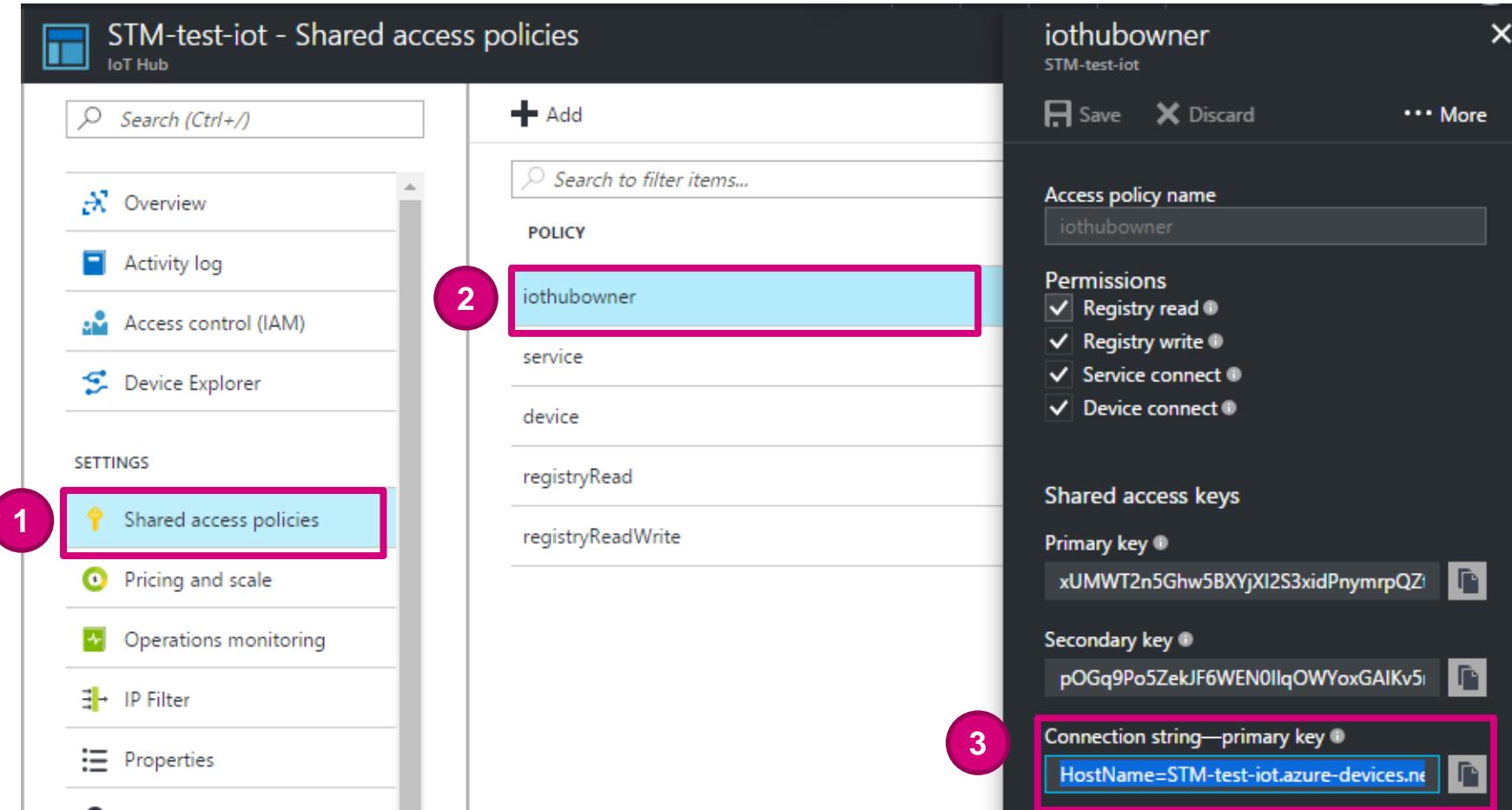


Appendix 2: Using Device Explorer tool to manage devices connecting to your IoT hub

Step 1: Visualize IoT Hub Credentials and Retrieve Connection String

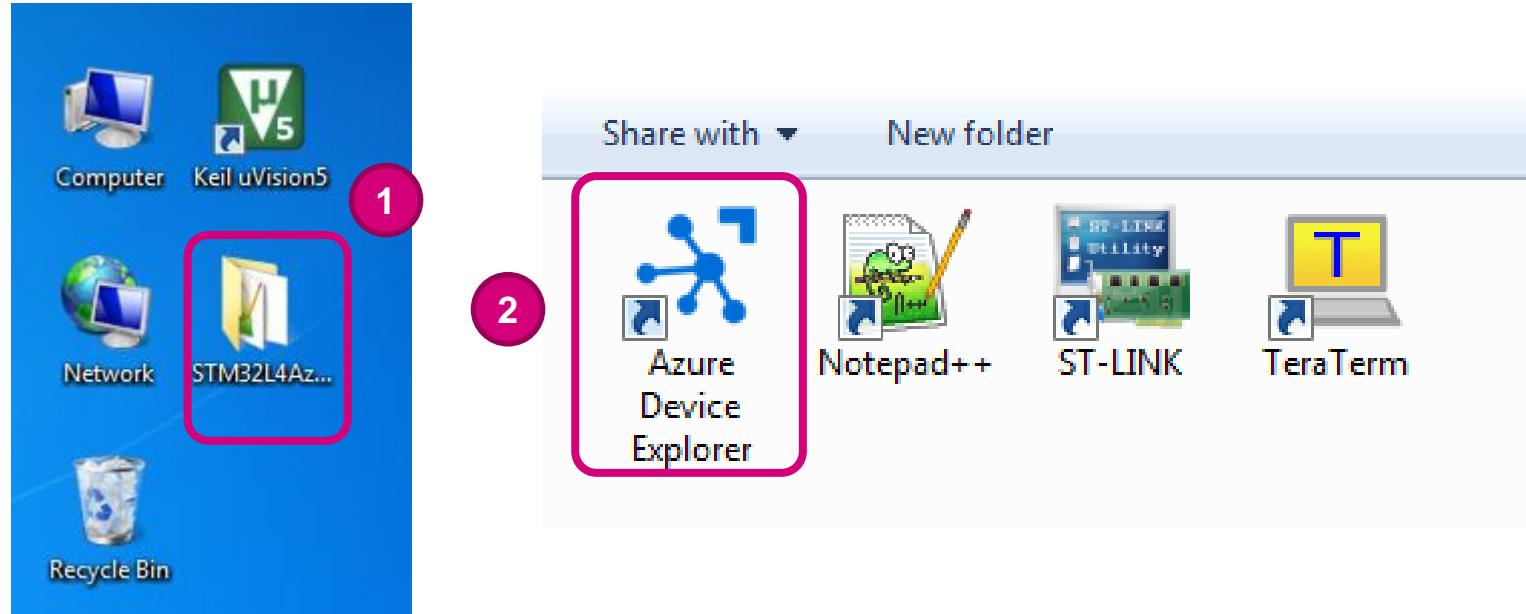
202

- 1 Within settings pane for the IoT Hub, click on Shared access policies pane
- 2 Click on iothubowner in the left side bar
- 3 Copy the Connection string to Notepad++



Step 2: Launch Device Explorer Desktop Application

203

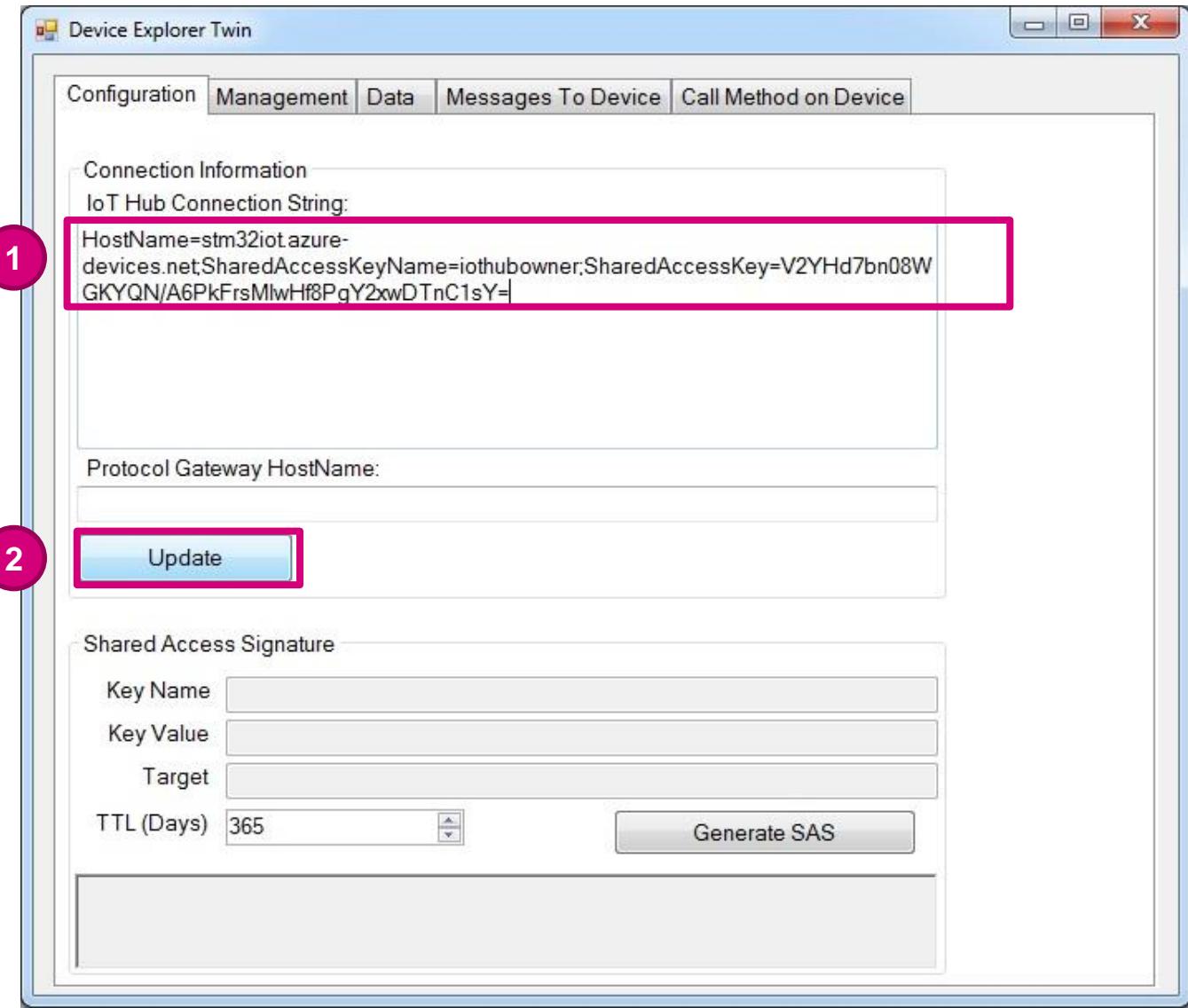


- 1 Open **STM32L4AzureSeminar** folder in your Desktop
- 2 Double-click on the **Device Explorer** shortcut

Step 3: Update IoT Hub Connection String in Device Explorer

204

- 1 Paste the IoT Hub connection string (from Page 64) in the IoT Hub Connection String
- 2 Click on Update

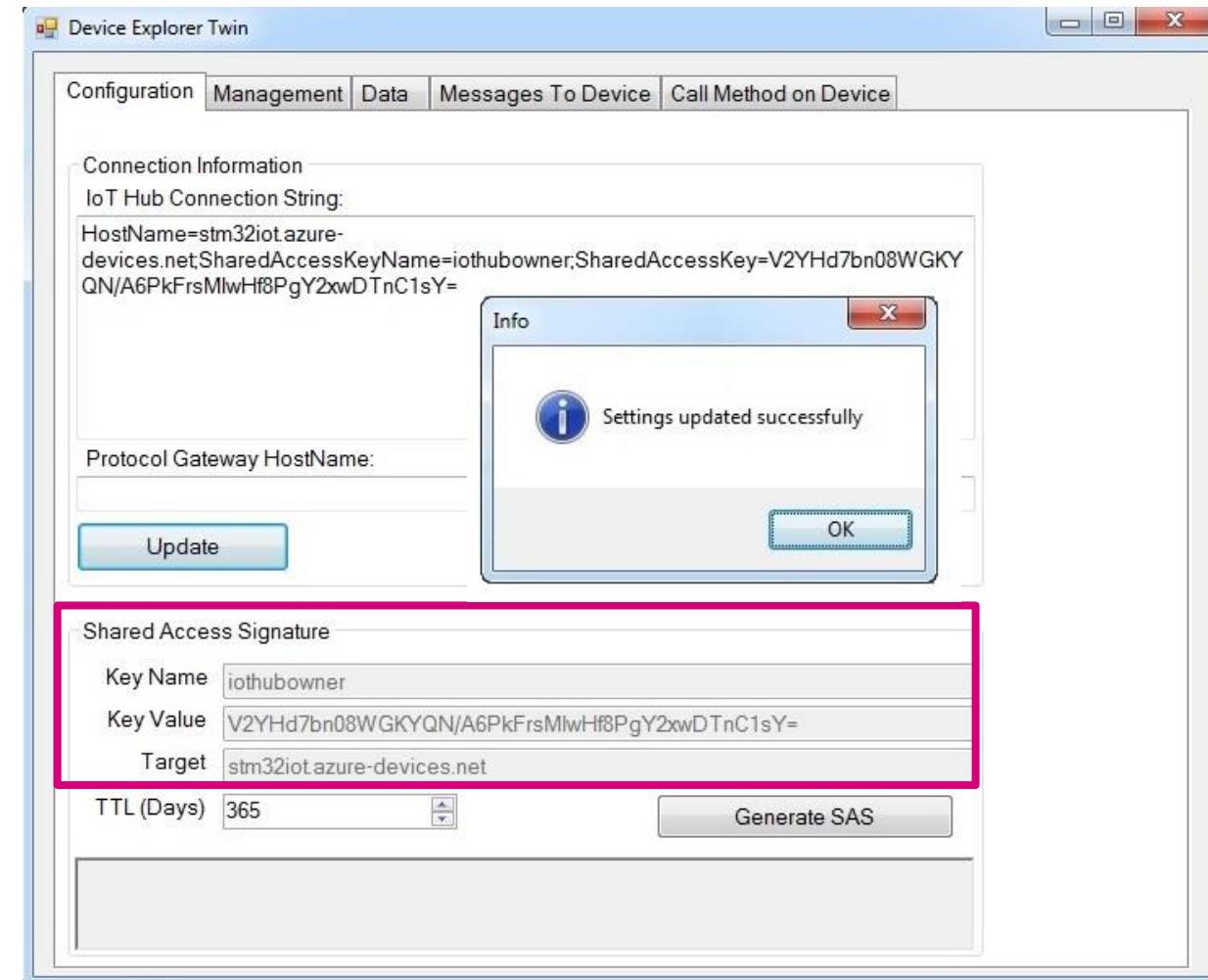


Step 3: Update IoT Hub Connection String

205

in Device Explorer (Cont'd)

- On successful update the section **Shared Access Signature** is populated with the three components of the IoT Hub Connection String:
 - Key Name
 - Key Value
 - Target



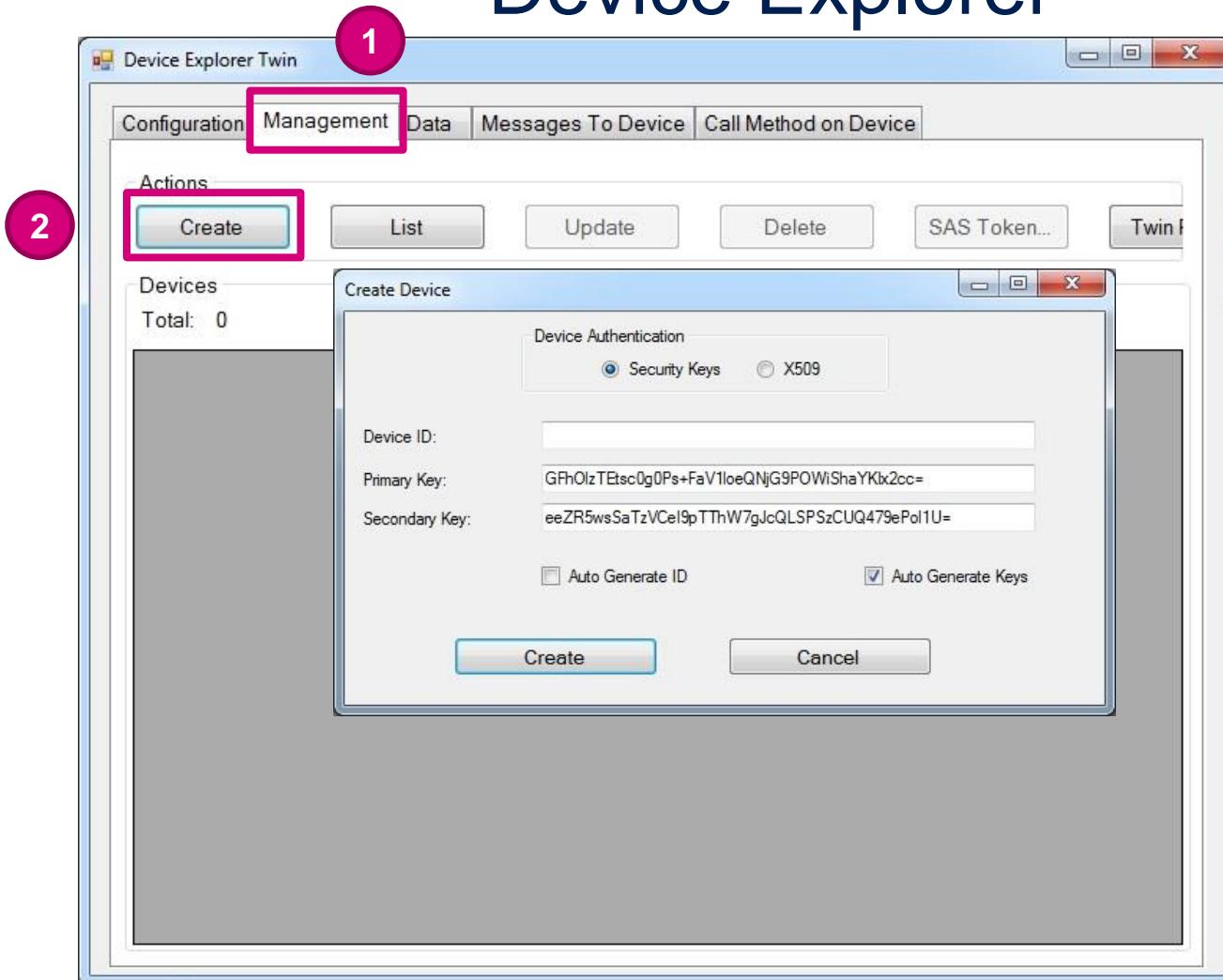
Step 4: Create a New Device in Device Explorer

206

- You are now ready to create your IoT device in the IoT Hub:

- 1 Go to tab Management
- 2 Click on Create

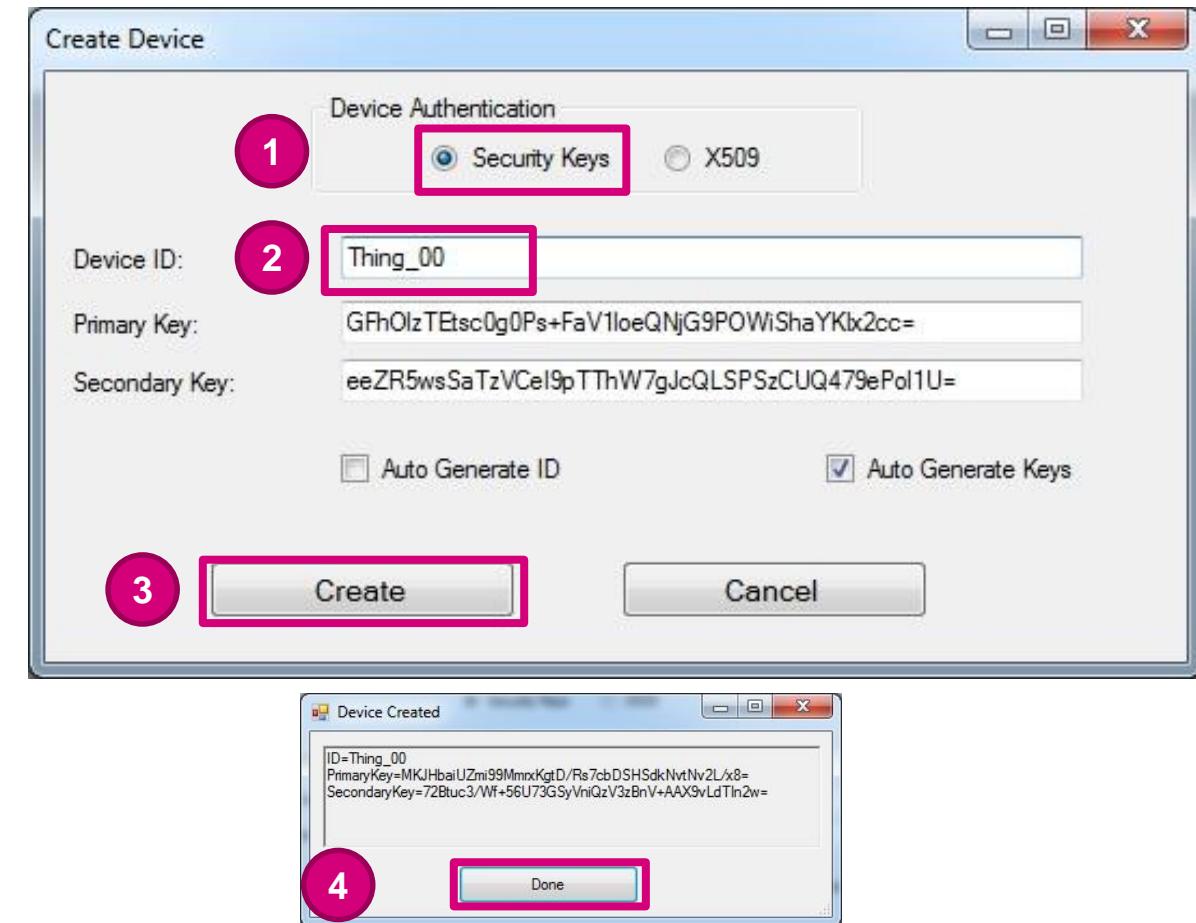
- A **Create Device** window will appear showing Primary and Secondary keys which are automatically generated



Step 4: Create a New Device in Device Explorer (Cont'd)

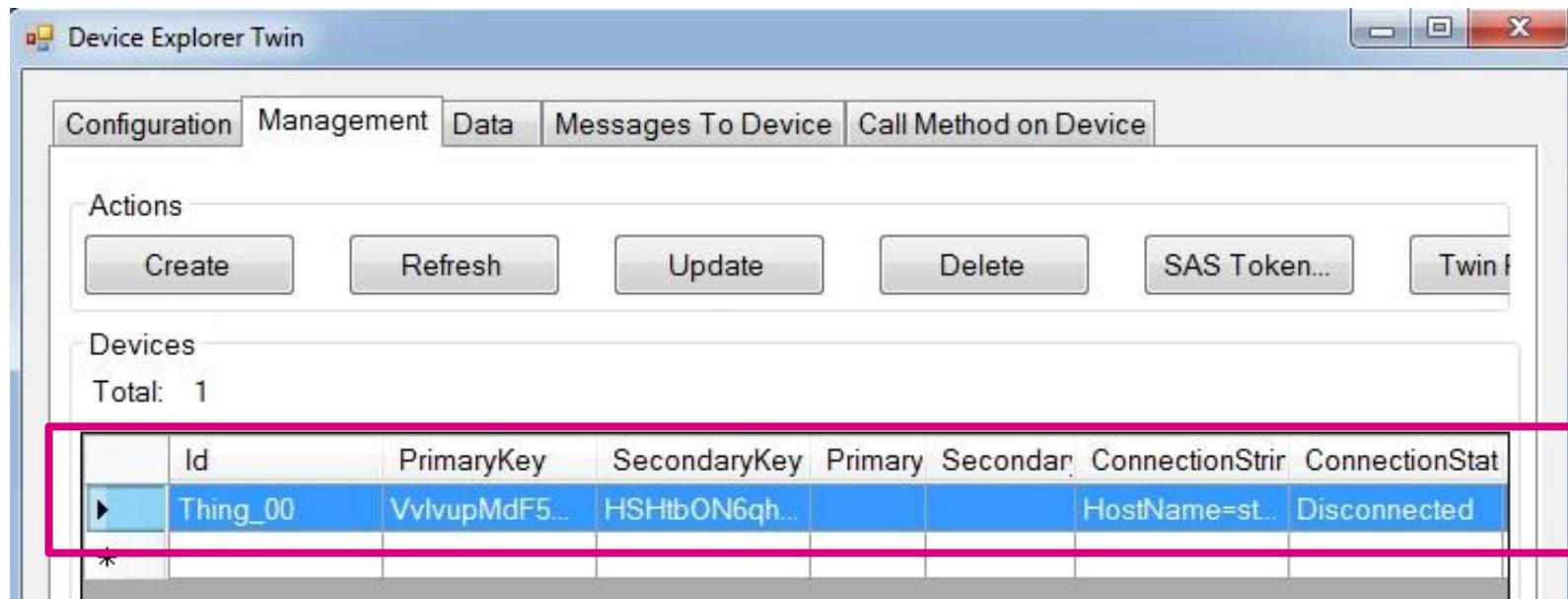
207

- 1 For this tutorial we rely on device authentication with keys. However, also device identification with X509 certificated is supported by NucleoL4
- 2 Insert your device ID “Thing_XX”: “XX” is your participant number (2 characters) printed on the box
- 3 Click on **Create** to create the device in your IoT Hub
- 4 A Device Created window appears, indicating that your device has been successfully registered with this IoT Hub. Click “Done”



Step 4: Create a New Device in Device Explorer (Cont'd)

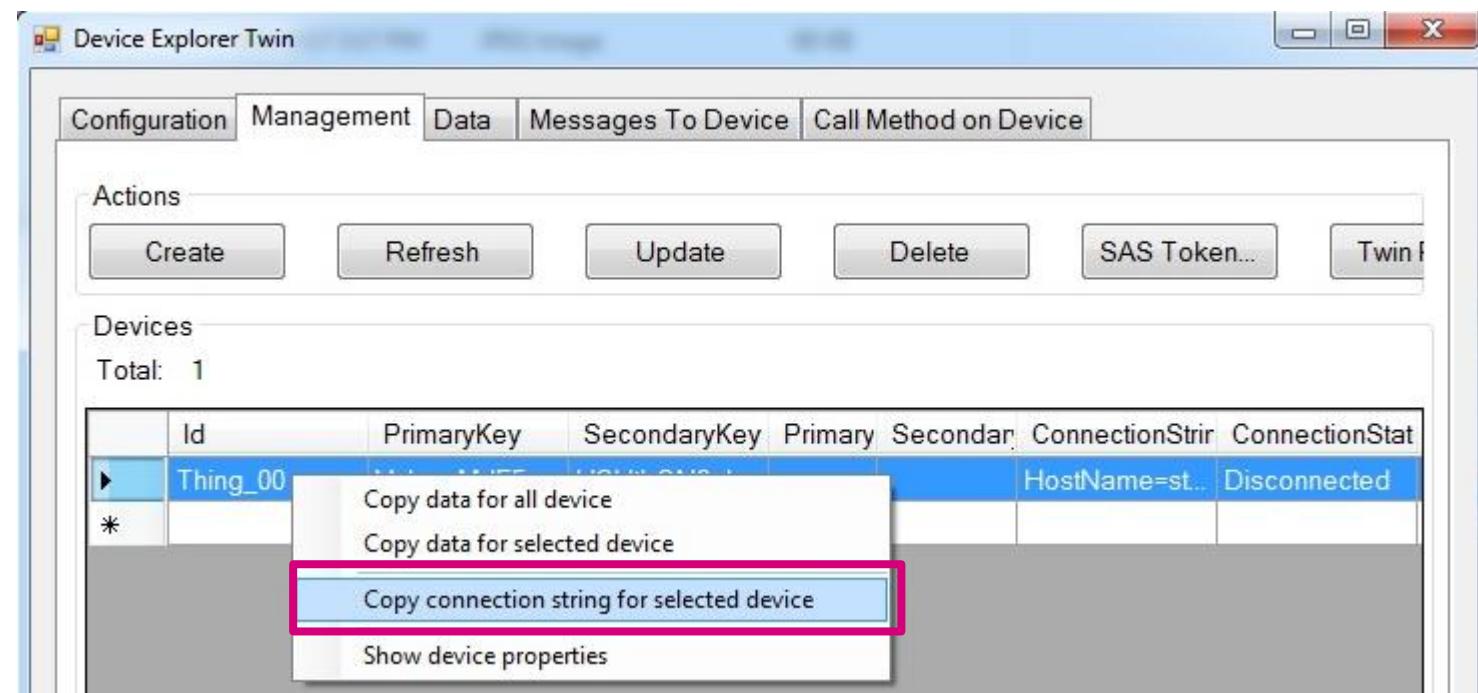
- An entry for your device has been created and listed in Management tab of your device explorer. This entry contains the connection string to be stored in the device to authorize access to your IoT Hub



Step 5: Retrieve Connection string for added device

209

- Select your device ID “Thing_XX”: “XX” is your participant number (2 characters) printed on the box
- Use the mouse right click for context menu then select Copy connection string for selected device
- Paste the connection string in Notepad++. It will be used later



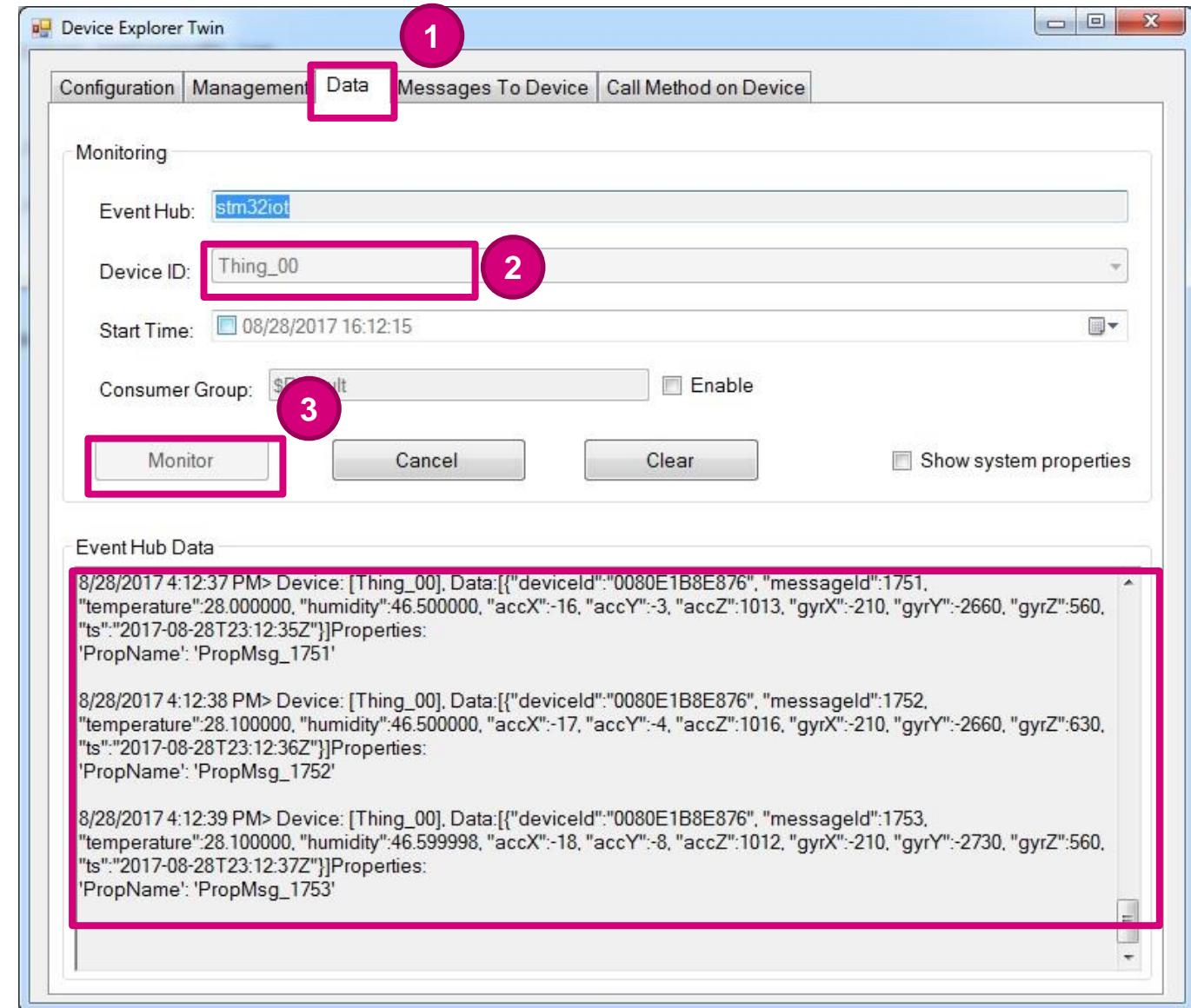
Step 6: Visualize Messages in Device Explorer

210

1 Select Data tab

2 In Device ID, Select your device "Thing_XX": "XX" is your participant number (2 characters)

3 Click on Monitor to start visualizing the log of messages received by IoT Hub



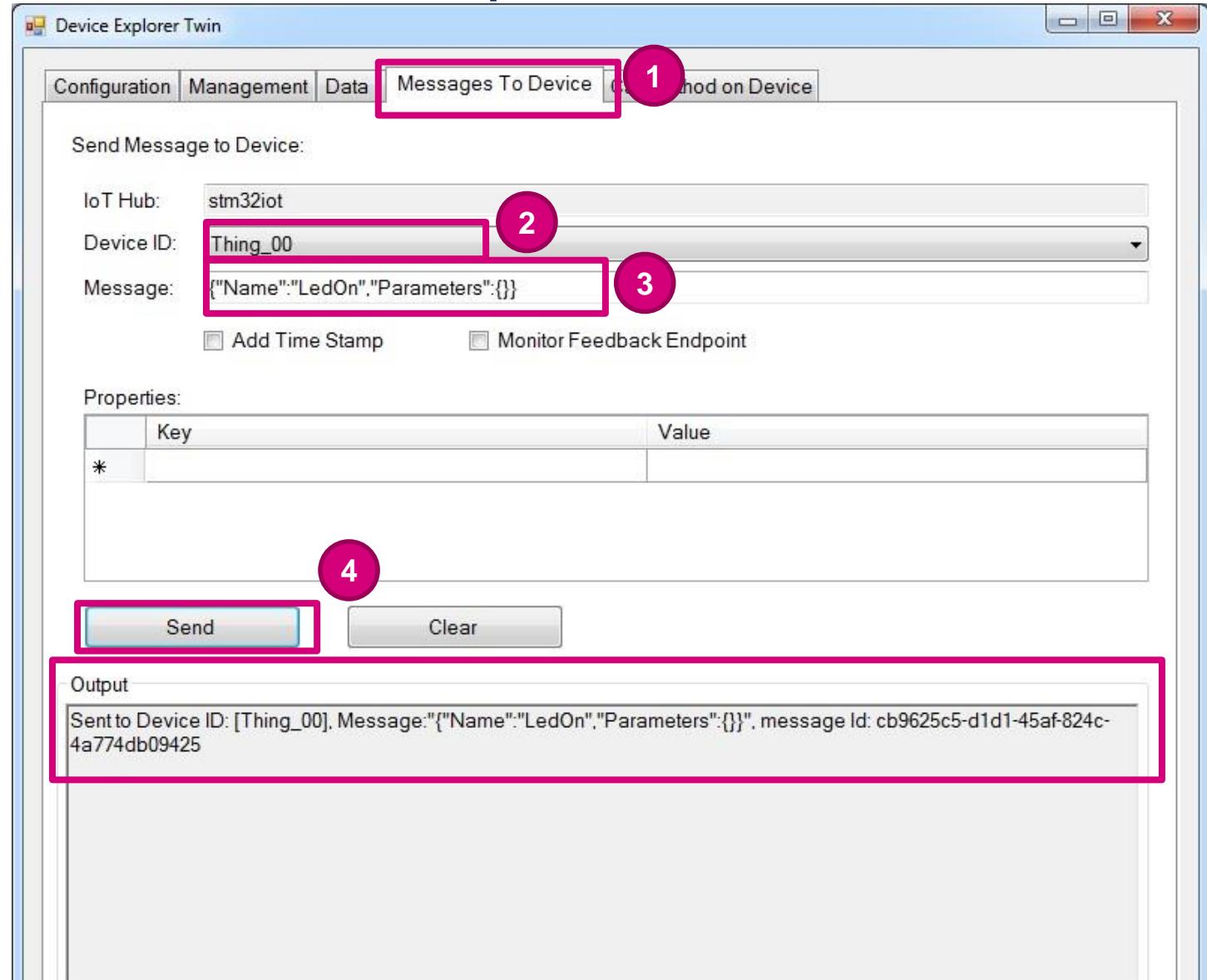
Step 7: Send Message to Device with Device Explorer: Turn LED On

- 1 Select **Message To Device** tab

- 2 In **Device ID**, Select your device "Thing_XX": "XX" is your participant number (2 characters)

- 3 Copy-paste the message below to the **Message** field to turn the green LED on:
 - {"Name":"LedOn", "Parameters":{}}

- 4 Click on **Send**



Step 8: Call FW-update Method on Device

212

with Device Explorer

- 1 Select Call Method on Device tab

- 2 Select your device "Thing_XX": "XX" is your participant number (2 characters)

- 3 Copy-paste the message below to Method name

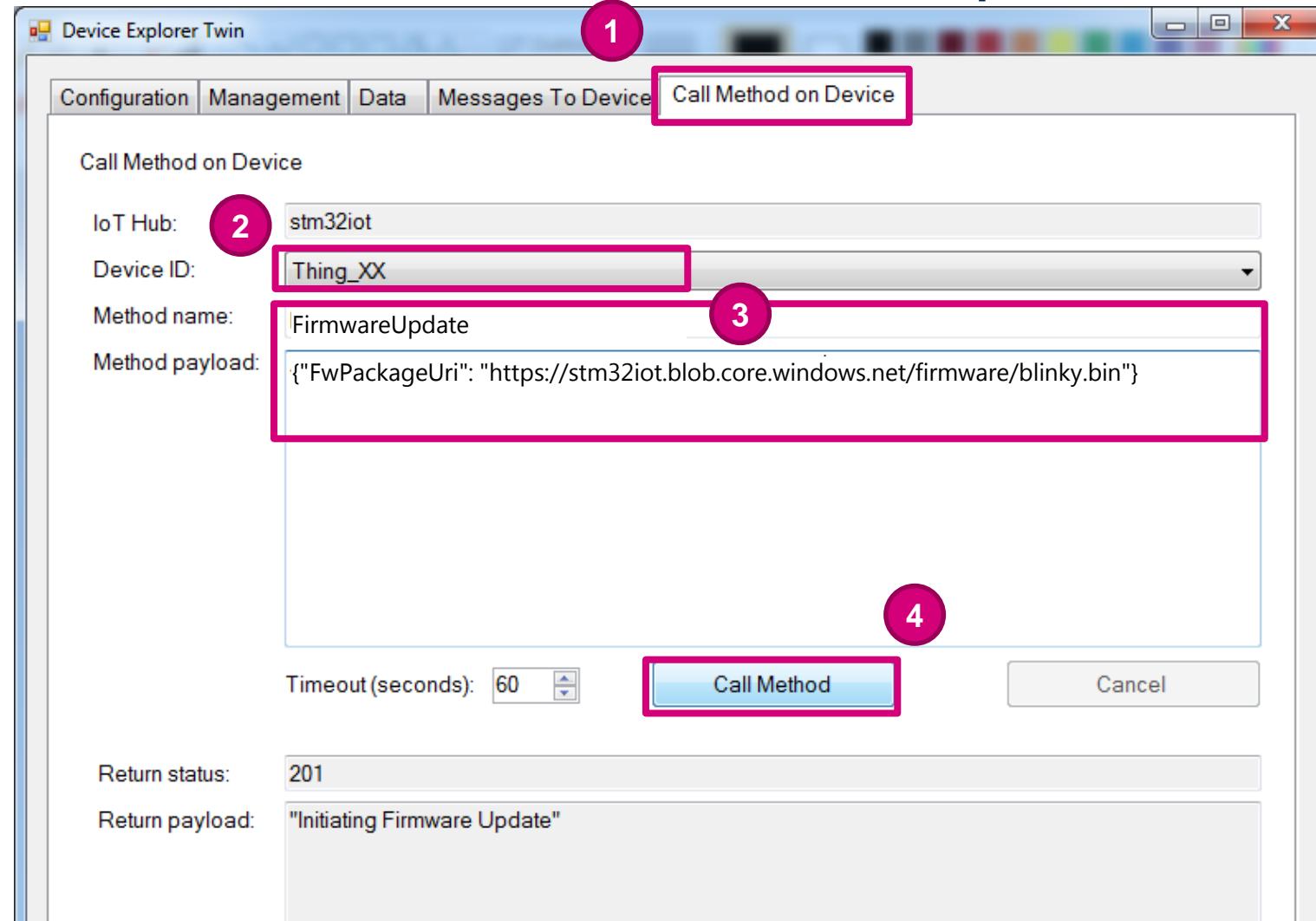
FirmwareUpdate

- Copy-paste the message below to Method payload

```
{"FwPackageUri":  
"https://stm32iot.blob.core.windows.net/  
firmware/blinky.bin"}
```

- Click on Call Method

4



Releasing Your Creativity

213



/STM32



@ST_World



st.com/e2e

