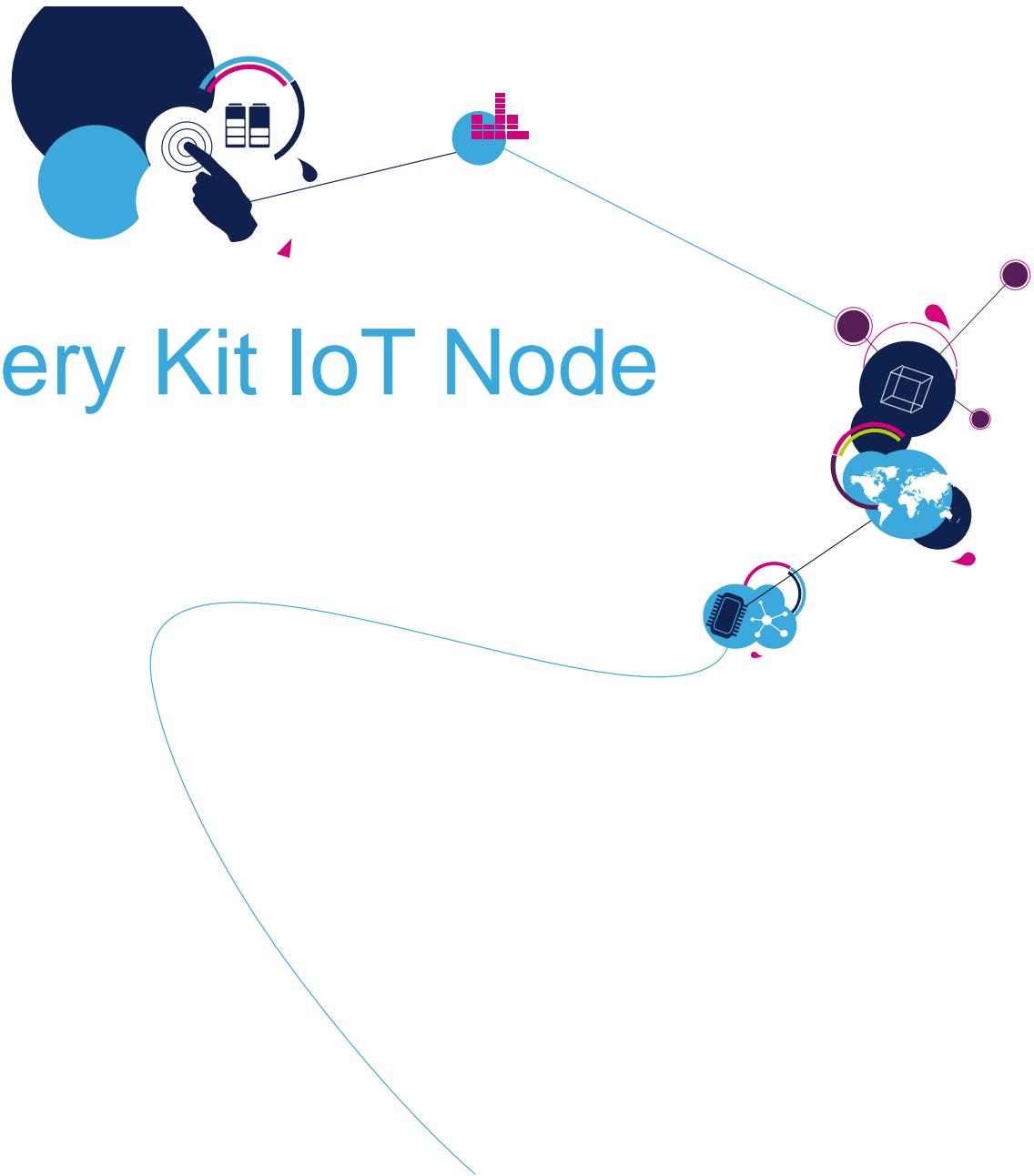


STM32L4 Discovery Kit IoT Node Seminar

Bruno Montanari
Cicero Pompeu

v1.6



Agenda

2

Presentation

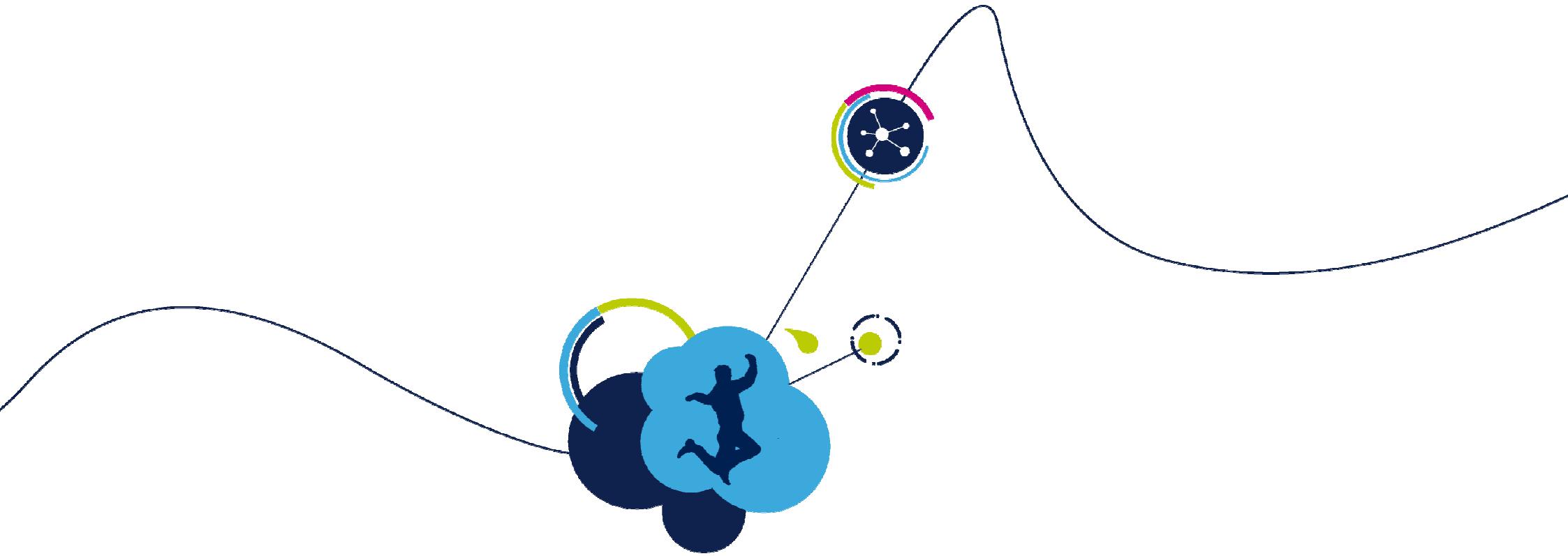
- Training Material Check/Installation Help
- Overview of the STM32 Portfolio
- IAR Installation
- Overview of the STM32L475
- Overview of the STM32L475 Discovery Kit IoT Node
- STM32Cube™ Introduction
- IAR License Installation
- ST-Link Utility Installation
- Lab 1 : Getting Started with STM32CubeMX - Blinky LED
- USB Overview
- NFC Overview
- Sensors Overview
- Lab 2 : Real Time Sensor Logging

Agenda

3

Presentation

- Bluetooth® Low Energy Overview
- Lab 3 : Bluetooth® Low Energy Pairing
- Lab 4 : Bluetooth® Low Energy P2P Communication
- SPIRIT Sub-GHz Overview
- Wi-Fi Module Overview
- Lab 5 : Wi-Fi Client Server
- Lora Overview
- Amazon AWS IoT Overview
- Lab 6 : Creating your Device (“Thing”) on AWS
- Lab 7 : Connect to AWS IoT & Send Sensor Data
- Alexa Voice Demo
- Optional Lab : USB Device Using a Virtual COM Port



Training Material Check / Installation Help

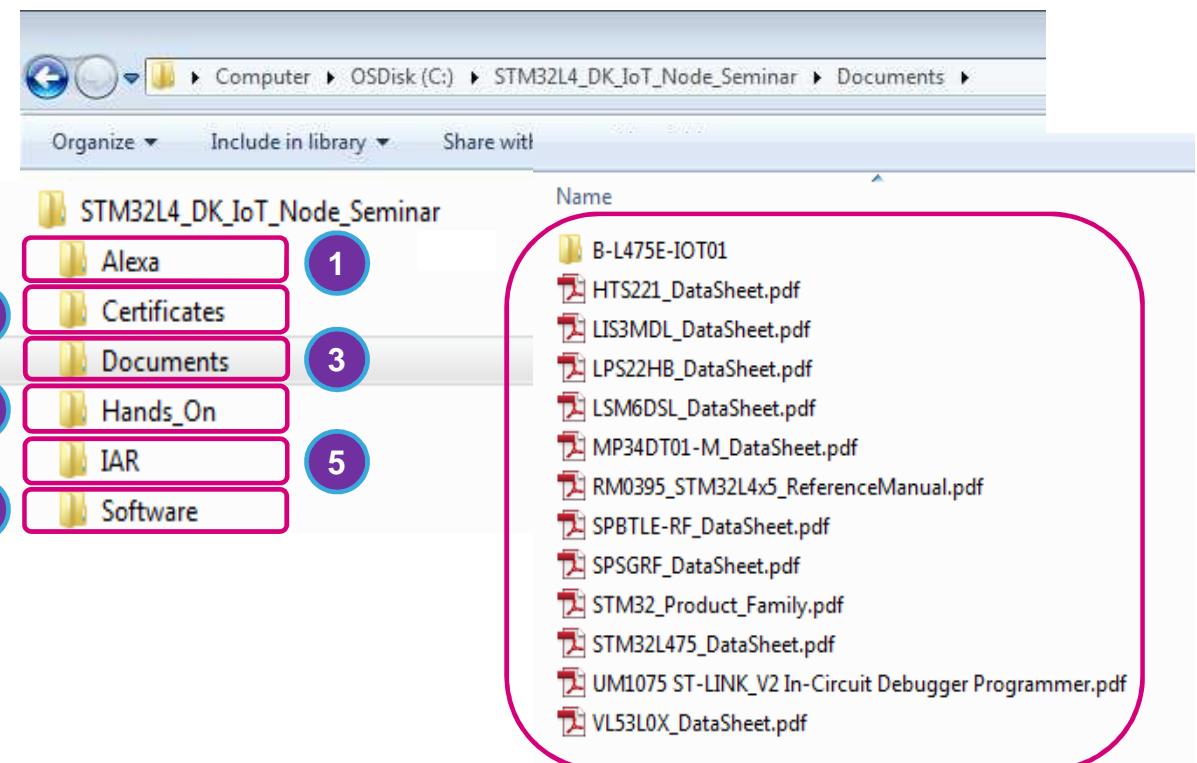
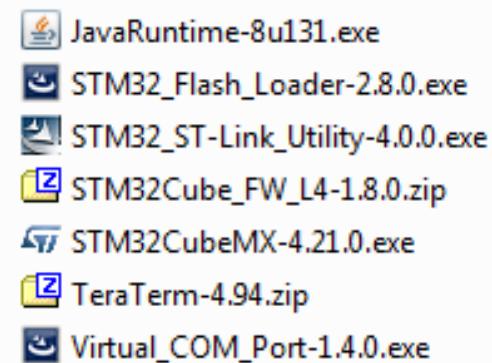
Training Material Installation

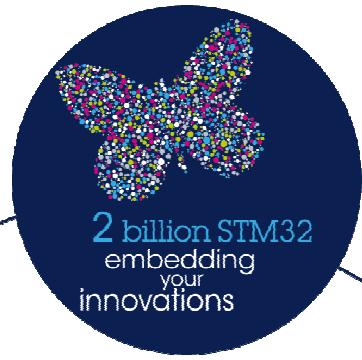
- Each participant should have install STM32CubeMX, the STM32CubeL4 HAL, JAVA, Tera Term, Notepad++. All the material and installers needed and associated lab collateral are available in:

[LINK here](#)

Seminar Directory Content

1. Alexa Skill Code
2. Thing Certificates
3. Documents
4. Hands on Labs
5. IAR
6. Software

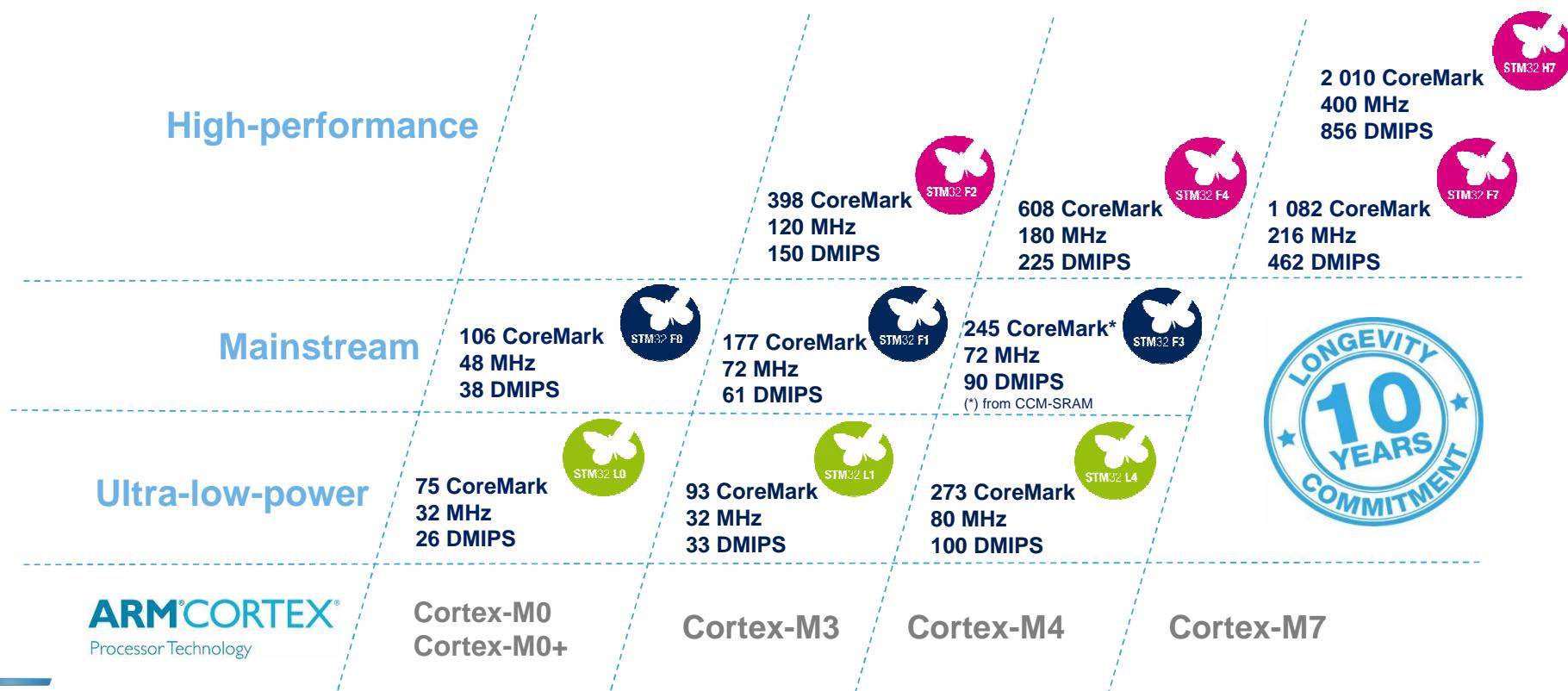




Overview of the STM32 Portfolio

STM32 Portfolio

10 product series / More than 40 product lines



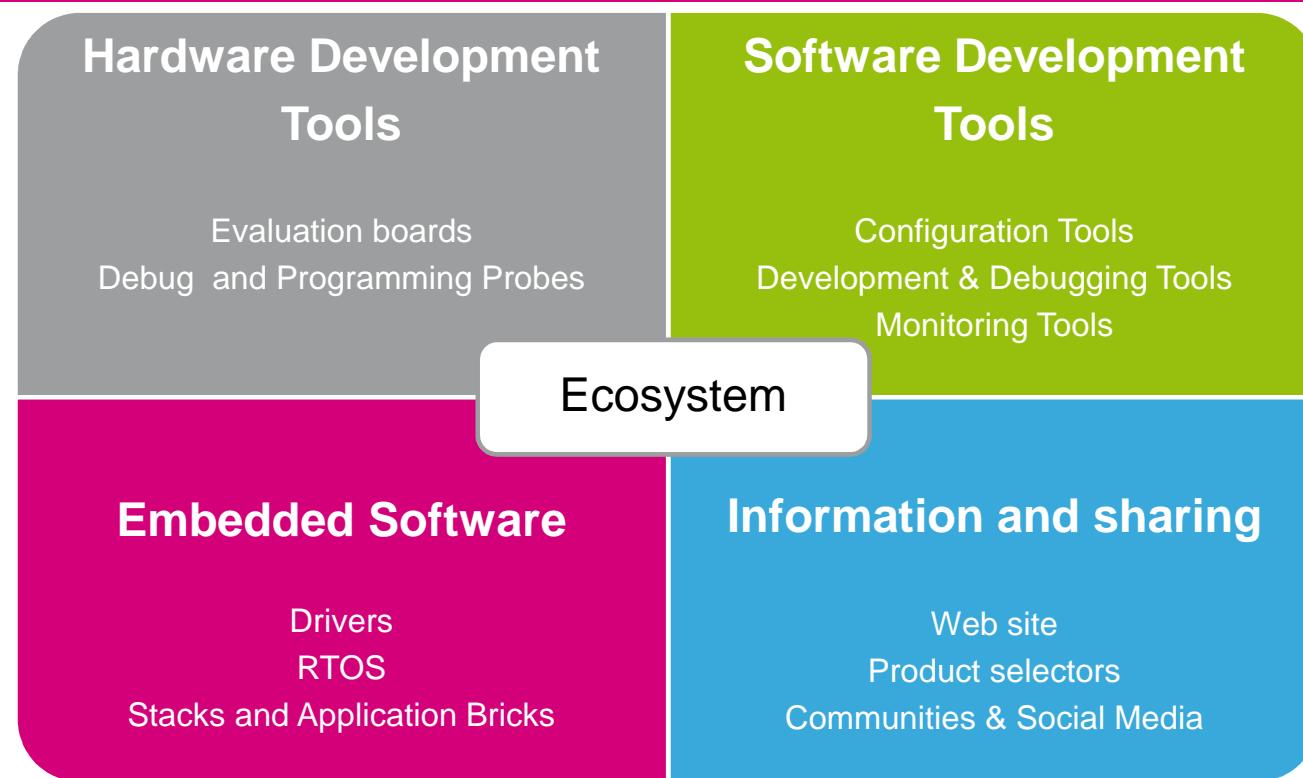
ARM®CORTEX®
Processor Technology



What is the STM32 MCU Ecosystem?

9

All collaterals required to develop with an MCU

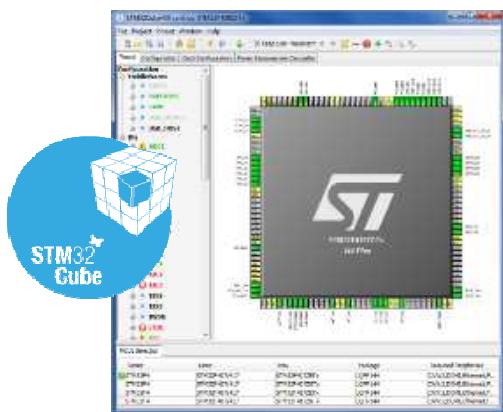


Hardware Development Tools

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

Software Development Tools

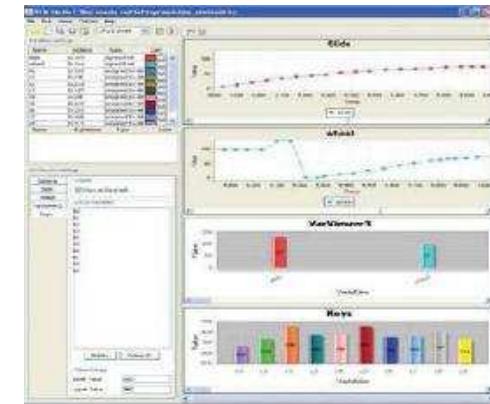
A complete flow, from configuration up to monitoring



STM32CubeMX
Configure & Generate Code



Partners IDEs
Compile and Debug



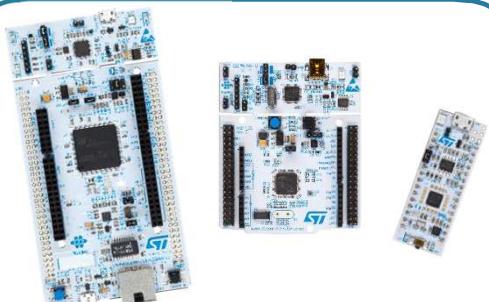
STMStudio
Monitor

STM32 ODE Platform

Hardware



STM32 Nucleo
development boards



Software



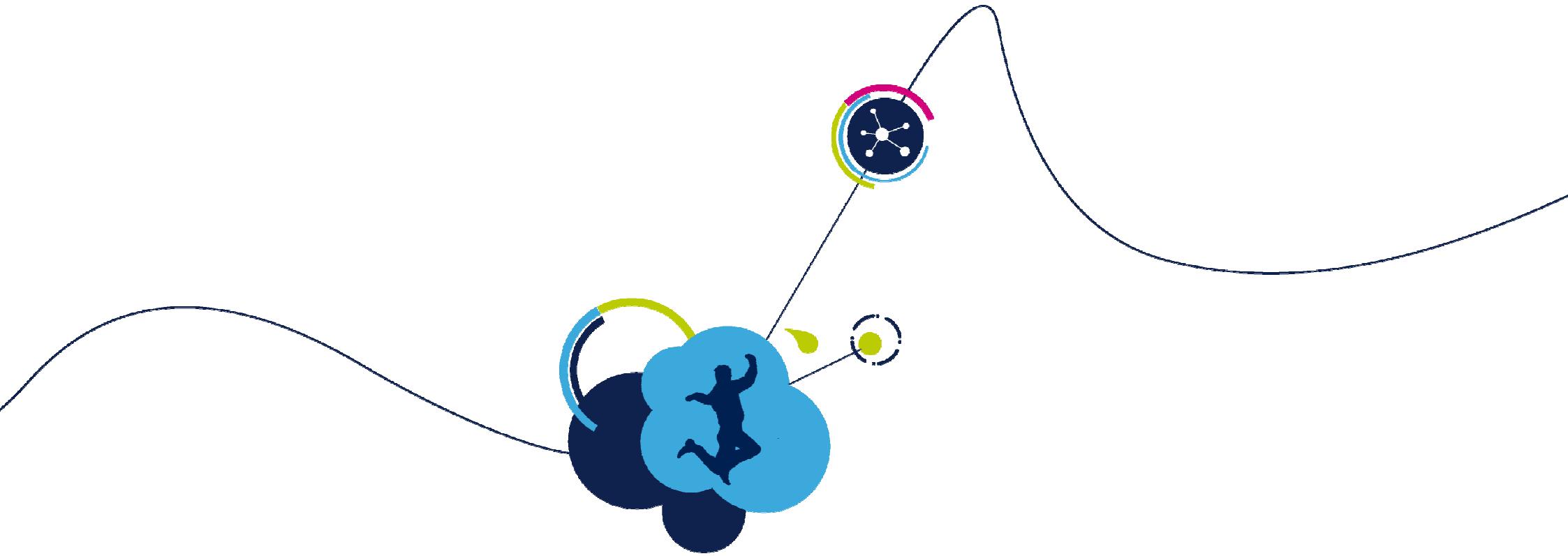
STM32Cube
software library

Expansion

STM32 Nucleo expansion boards
from ST and third parties



STM32Cube expansion SW



IAR Installation

IAR Installation

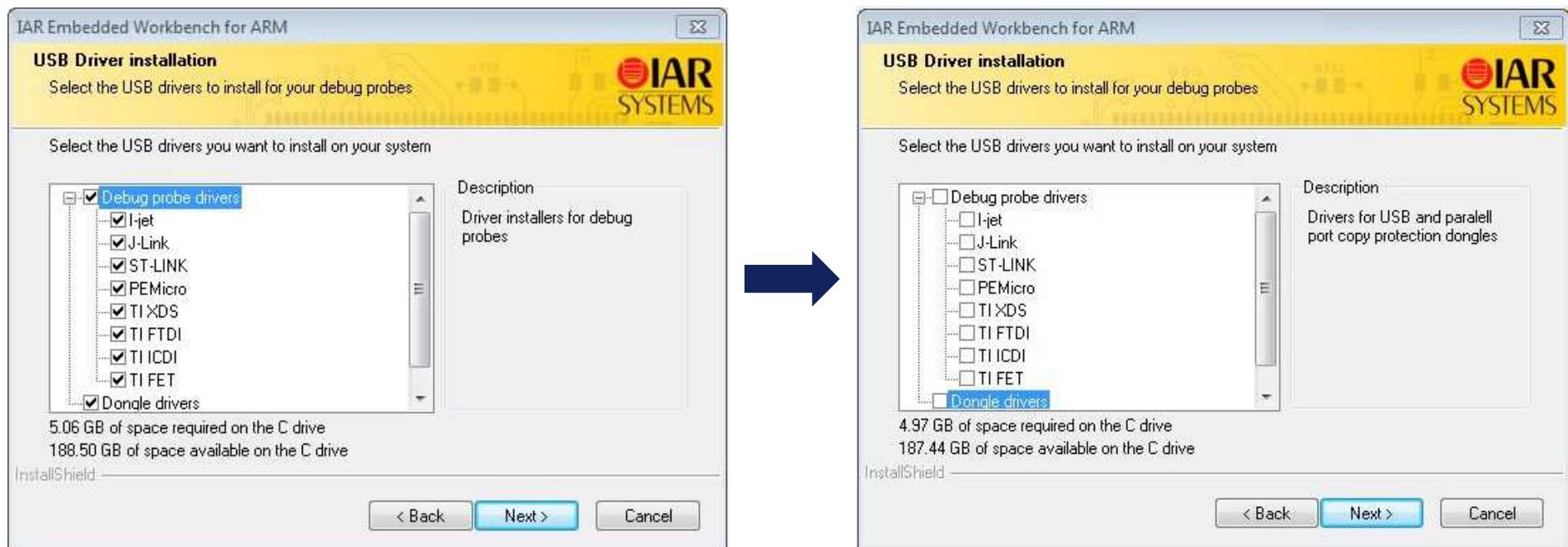
14

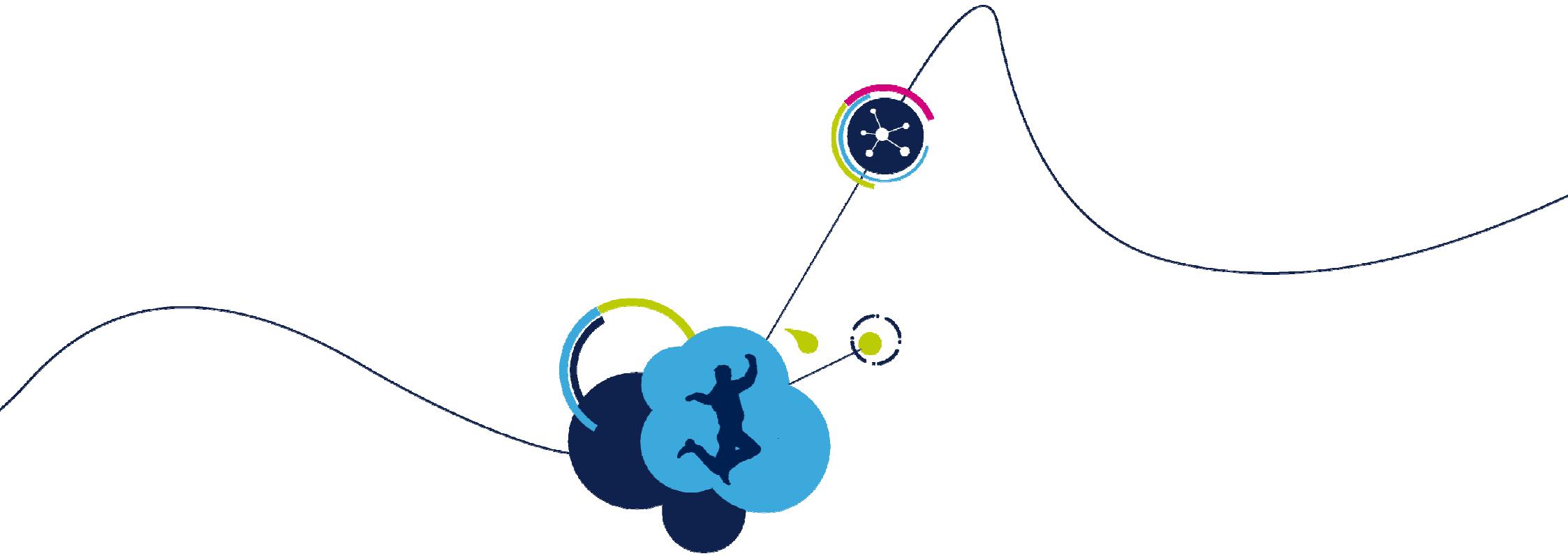
- Run the IAR professional tool suite installer:
C:\STM32L4_DK_IoT_Node_Seminar\IAR
- From the installer menu select Install IAR Embedded Workbench.



IAR USB Driver Installation

- De-select **ALL** the USB drivers when IAR prompts you:





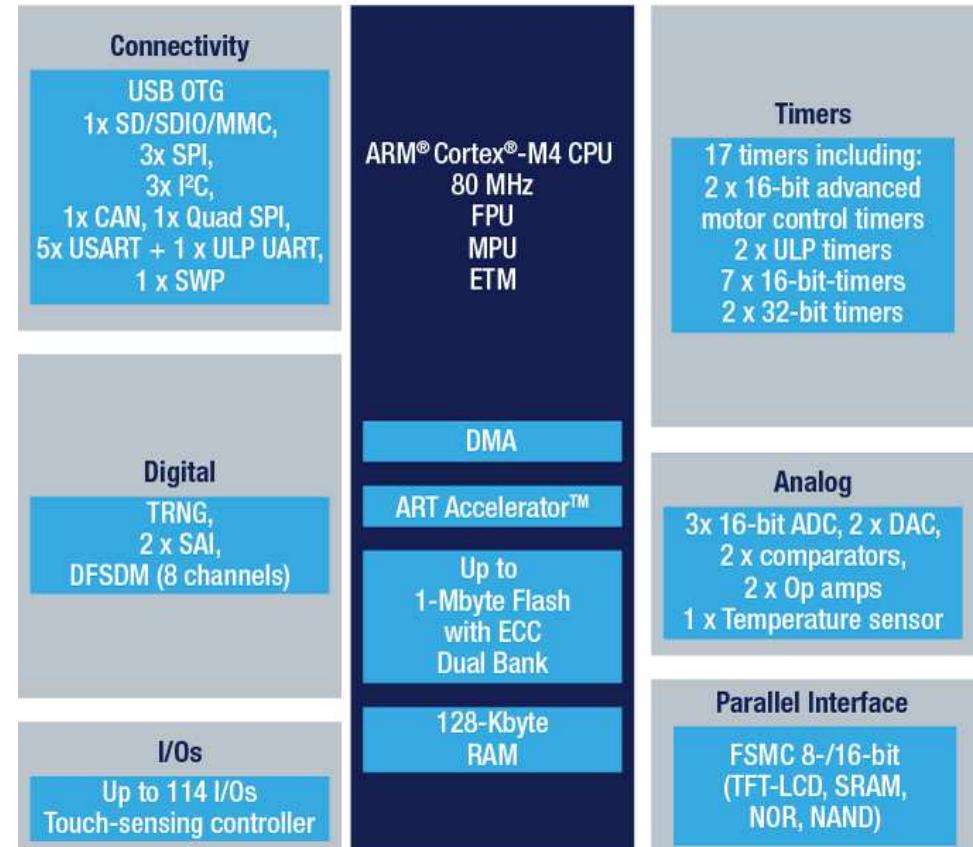
Overview of the STM32L475

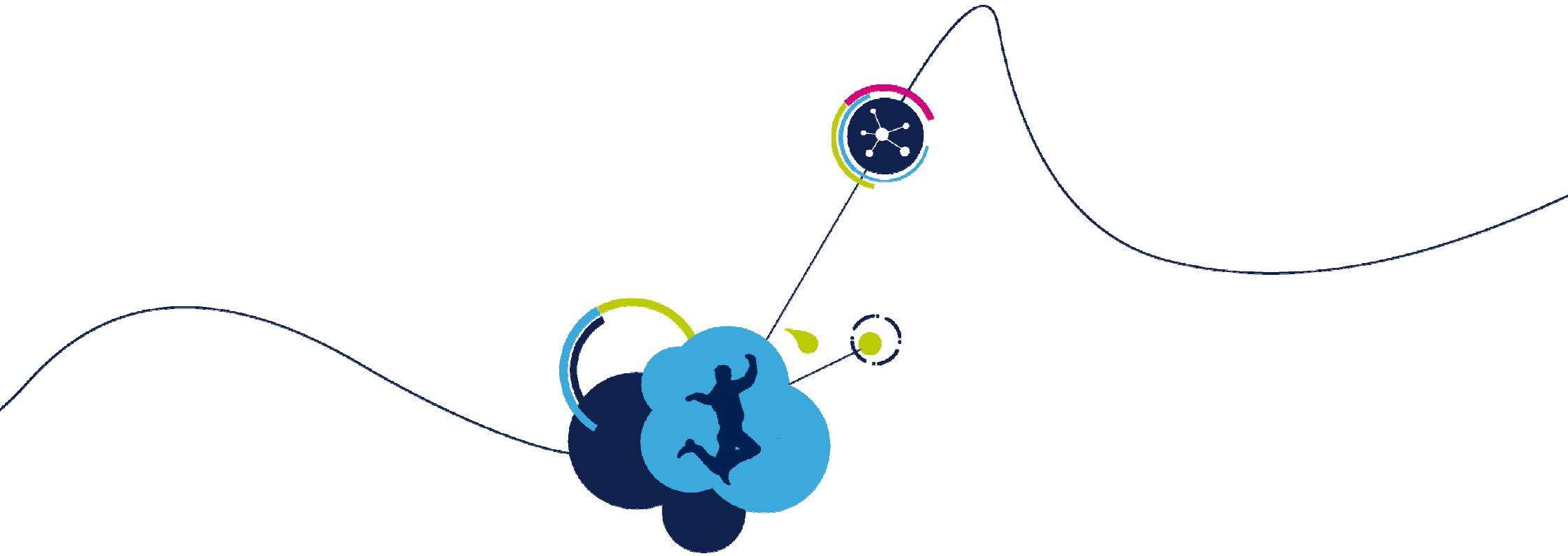
Overview of the STM32L475

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, SWPMI, SDMMC, 2x SAI
- Digital filter for Sigma delta modulator
- 17 x timers
- Analog: Op-Amps, comparators, DAC, VREF, Temp
- RNG

STM32L475



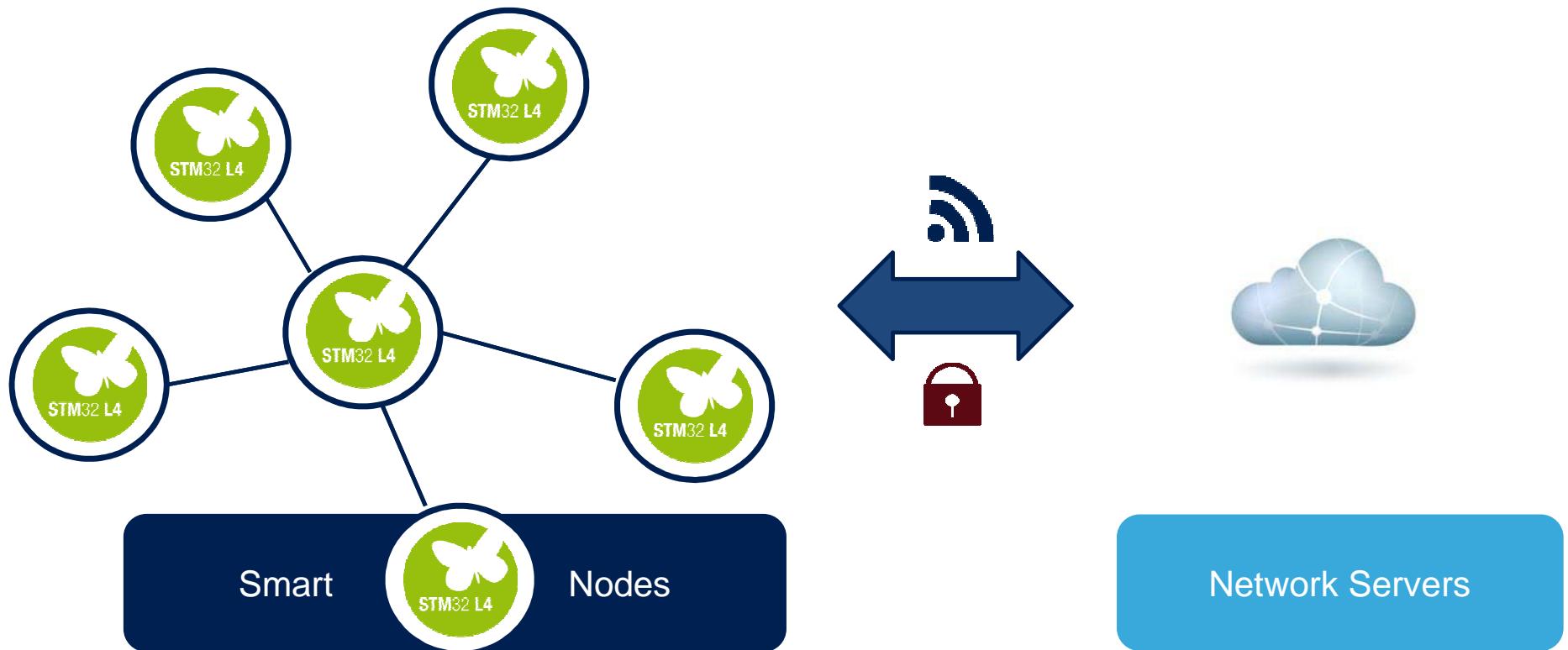


Overview of the STM32L4 Discovery Kit IoT Node

STM32L4 Discovery Kit IoT Node

19

Get connected seamlessly!



STM32L4 Discovery Kit IoT Node

20

Open the door to remote services

Direct connection to cloud servers

Low-power long-range communication

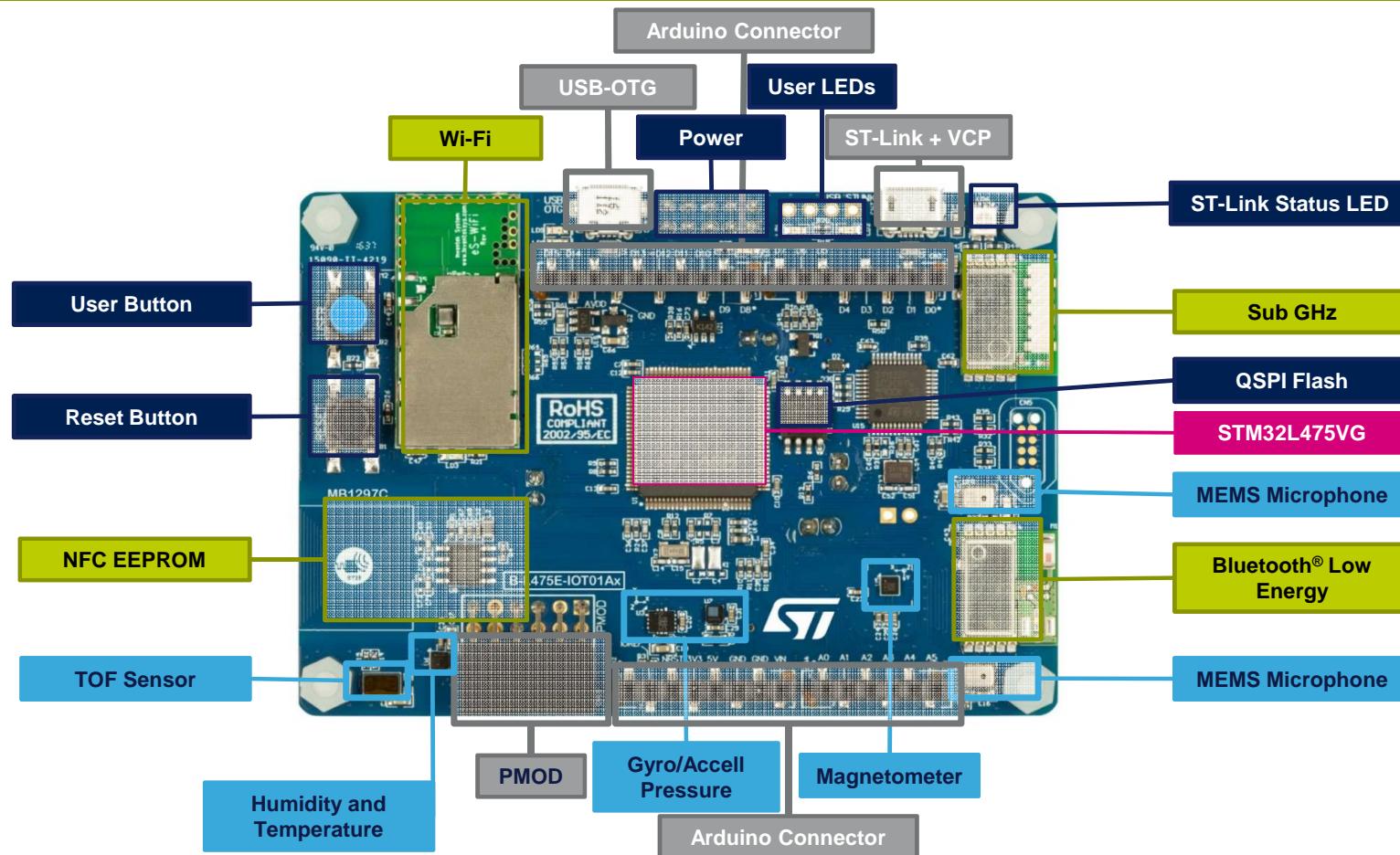
Environmental awareness: humidity, pressure, temp

Detection hub: motion, proximity, audio



STM32L4 Discovery Kit IoT Node

Multi-link communication, multiway sensing



Comprehensive Software Libraries

22

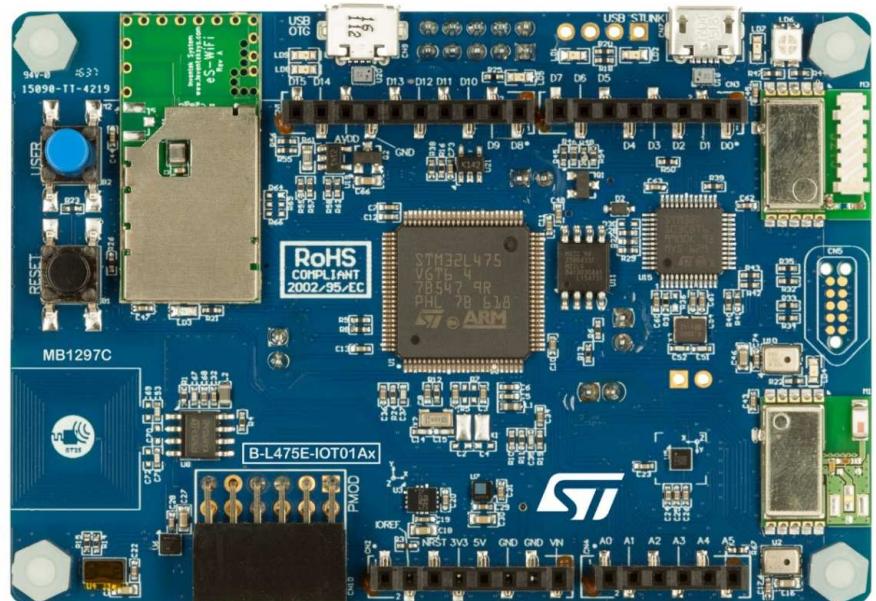
Instant showcase

SW Libraries for STM32L4 MCU & sensors

Connectivity SW protocol stacks

Cloud service connectors (AWS)

Demo examples (X-CUBE-AWS)



Wireless Connectivity – Wi-Fi

- 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®

- 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

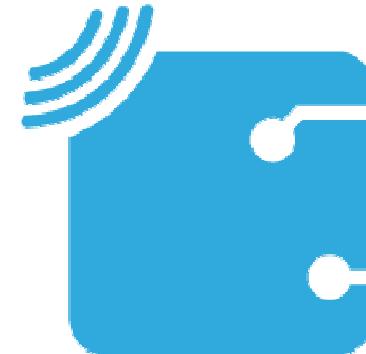
- ST SPSGRF-915 Sub-GHz Module (915 MHz - US)
 - FCC and IC certified module with integrated balun & antenna
 - Supports 2-FSK, GFSK, MSK, GMSK, OOK and ASK modulation schemes
 - Long range (100s of meters) with an air data rate from 1 to 500 kbps



Wireless Connectivity - NFC

- 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

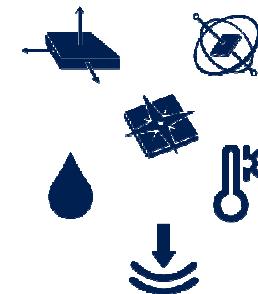


Wired Connectivity Features

- 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

29

- 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



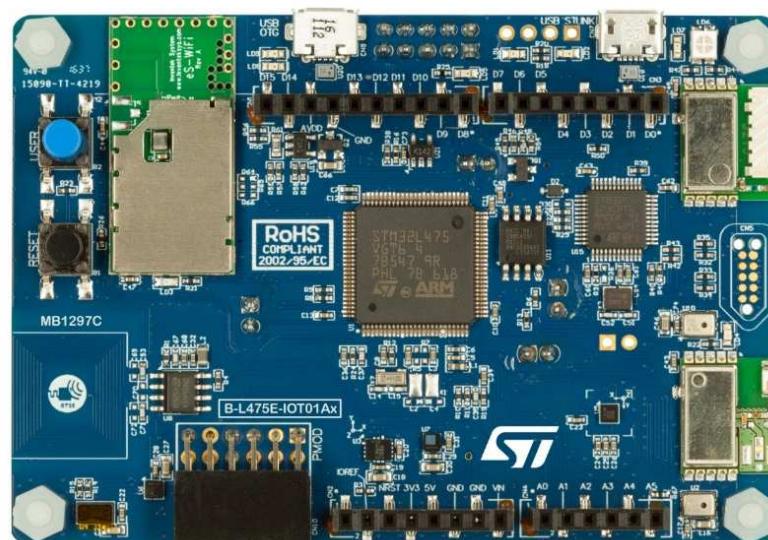
Advantages of a Single Board

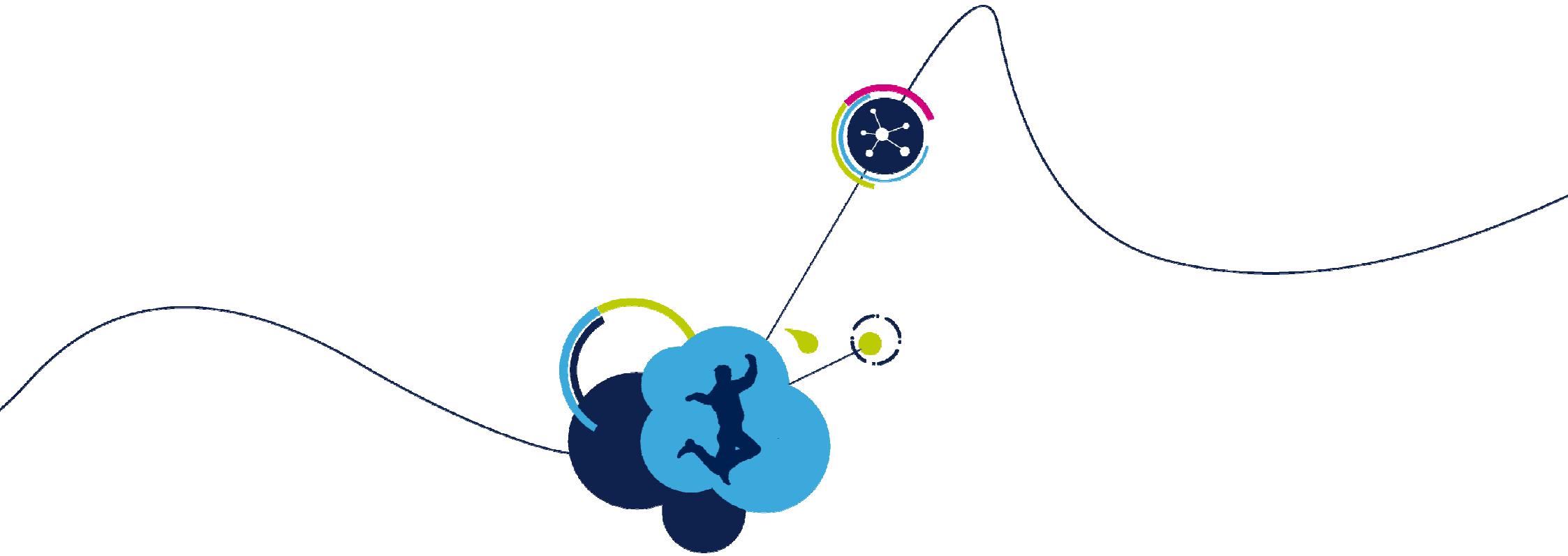
- Easily Debug Hardware Issues.
- Included Collateral is Tightly Coupled
 - BSP Included for all Board Components
 - Cloud Connectivity Reference Solutions Included
- Cost Effective Development Solution (~\$60)
- No Need to Manage & Order Multiple Board SKUs.

Availability

31

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

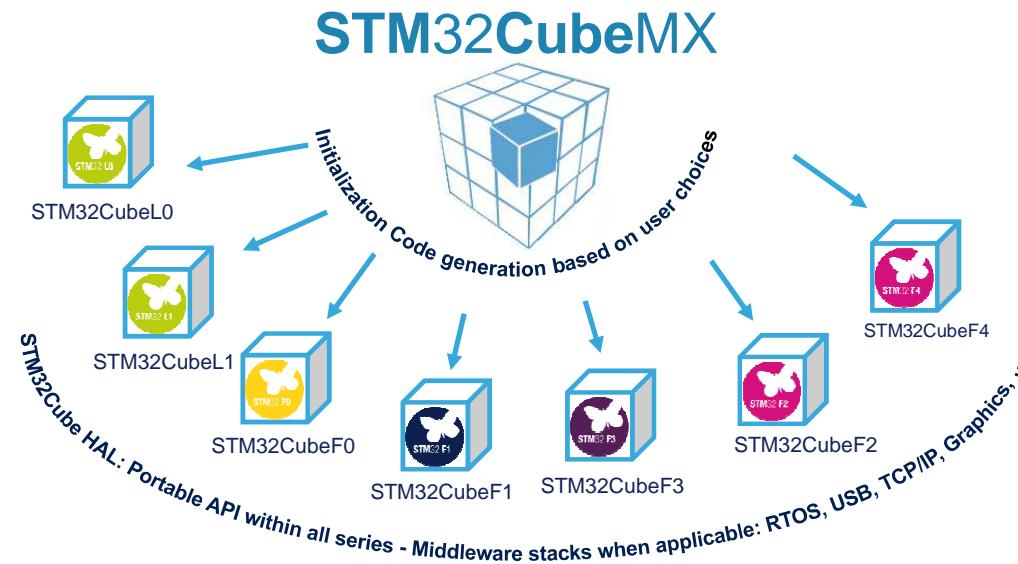




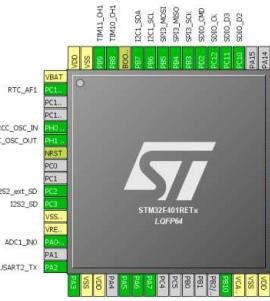
STM32Cube™ Introduction

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



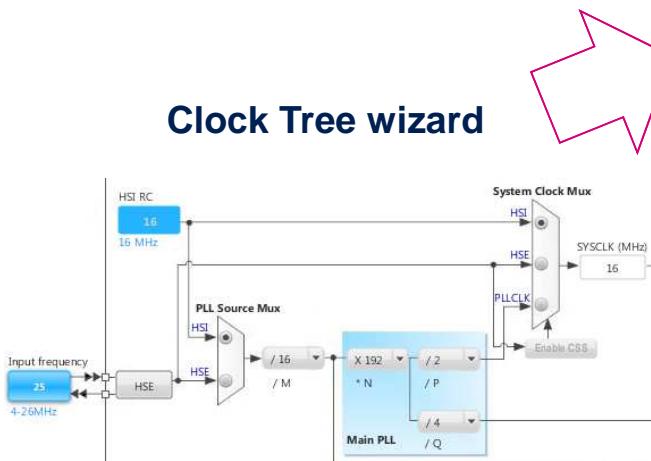
Pinout Wizard



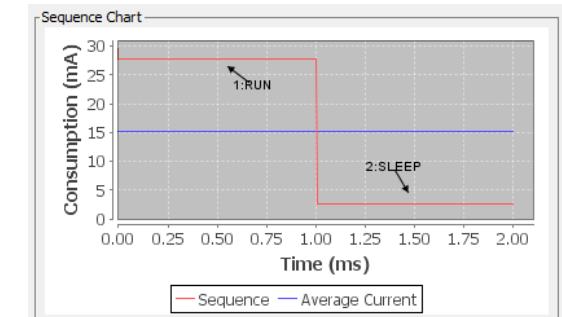
STM32CubeMX

<input type="checkbox"/> Basic Parameters	
Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1
<input type="checkbox"/> 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



Clock Tree wizard



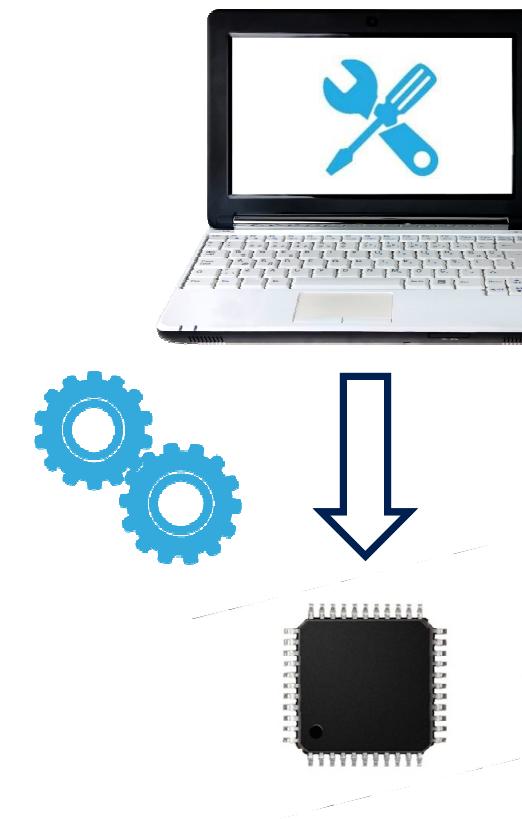
Power Consumption Wizard



lite augmented

STM32CubeMX : User Code Configuration

35

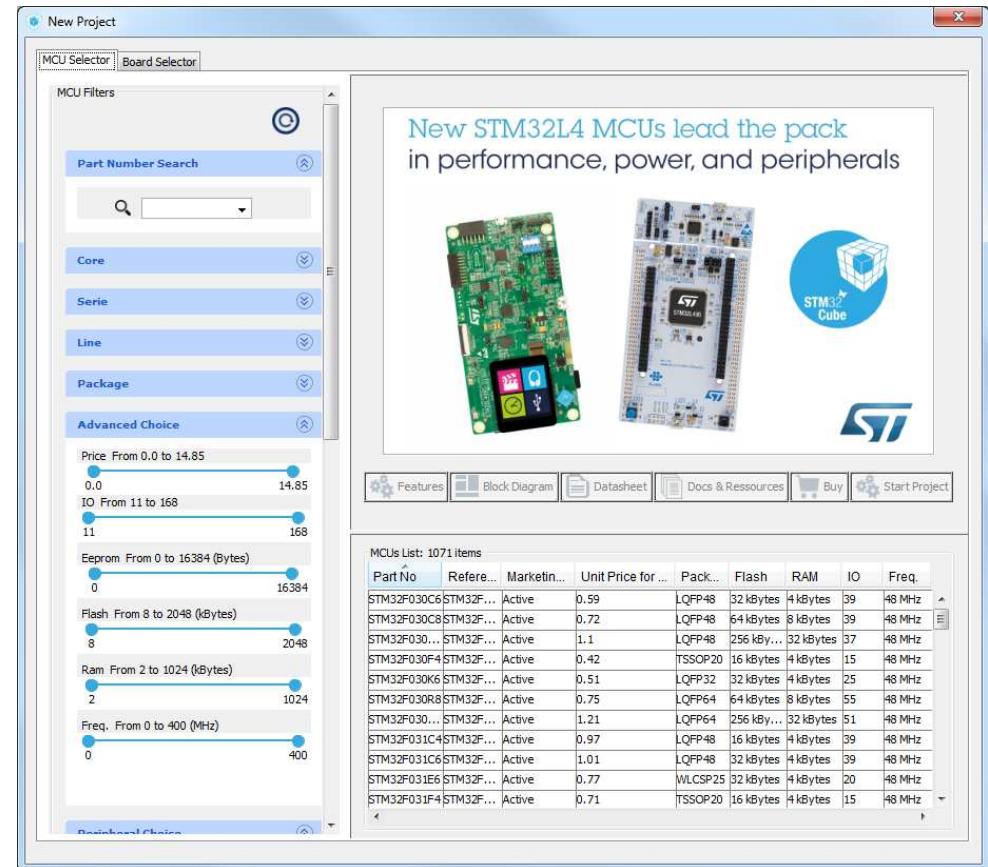


**Generates Initialization C Code
based on user driven
configuration!**

STM32CubeMX : MCU Selector

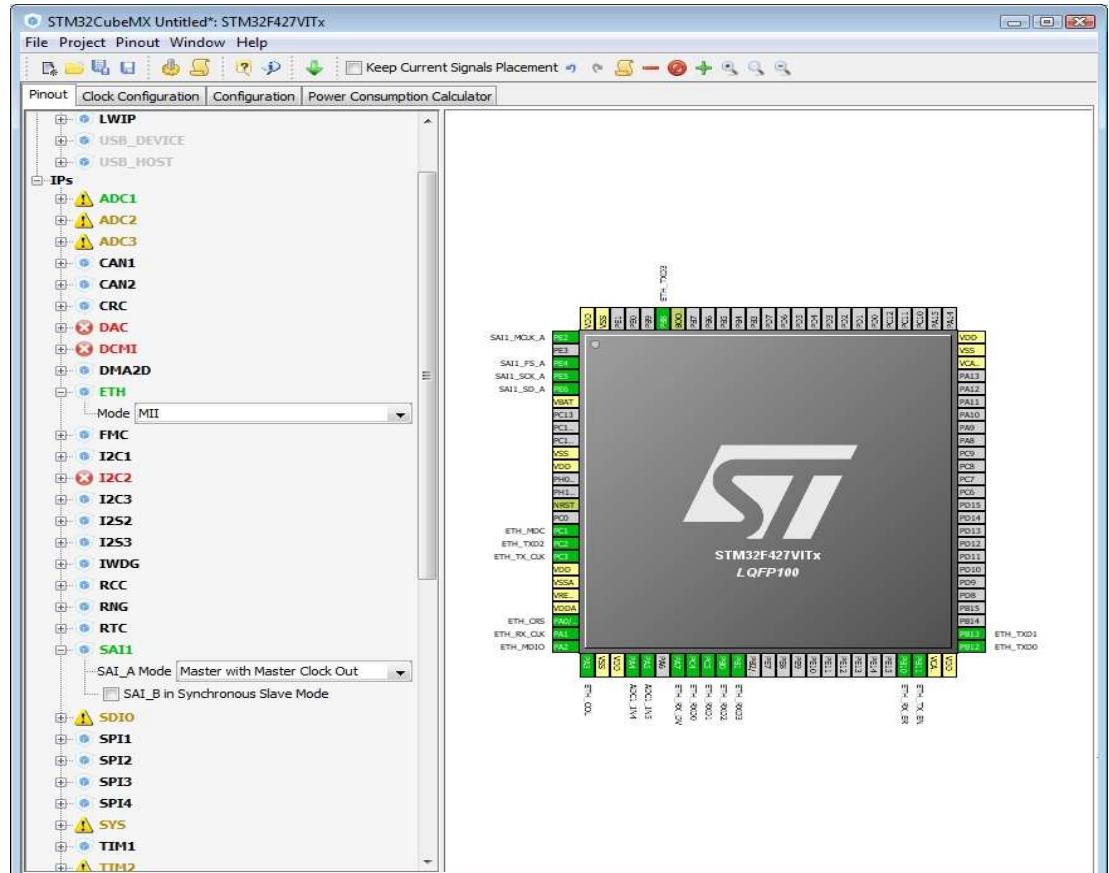
36

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



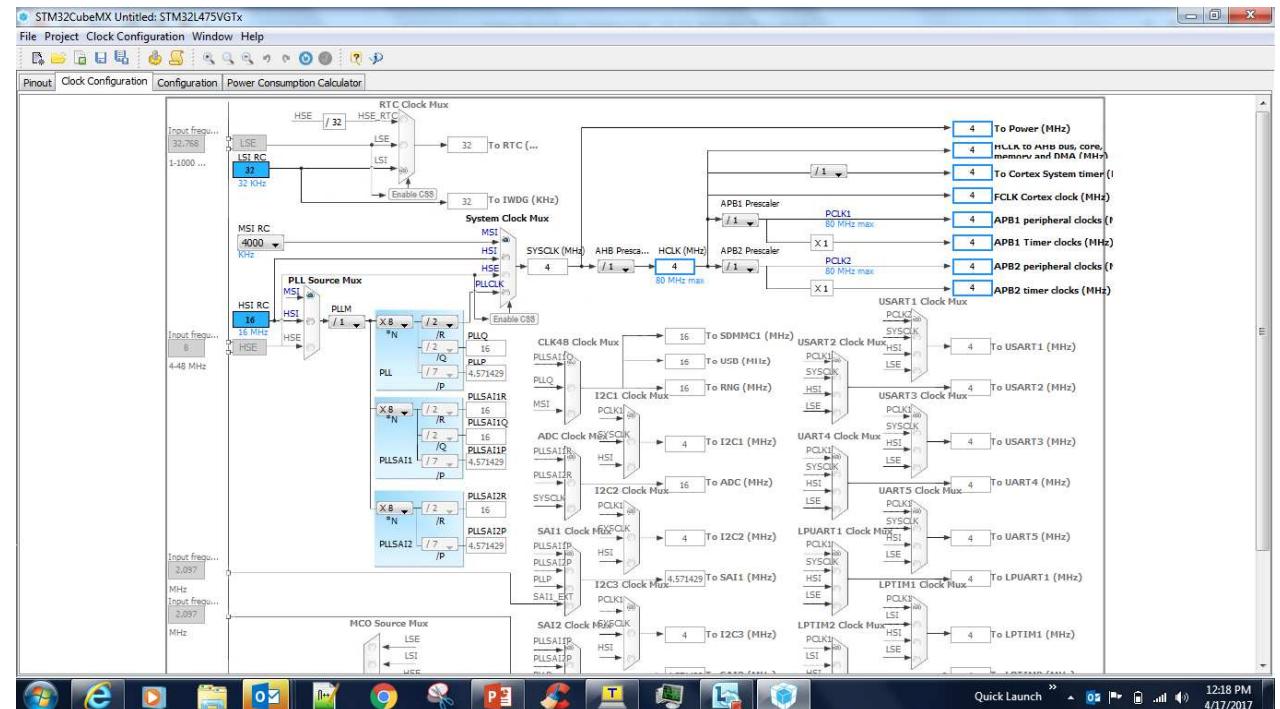
STM32CubeMX : Pin-out Configuration

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



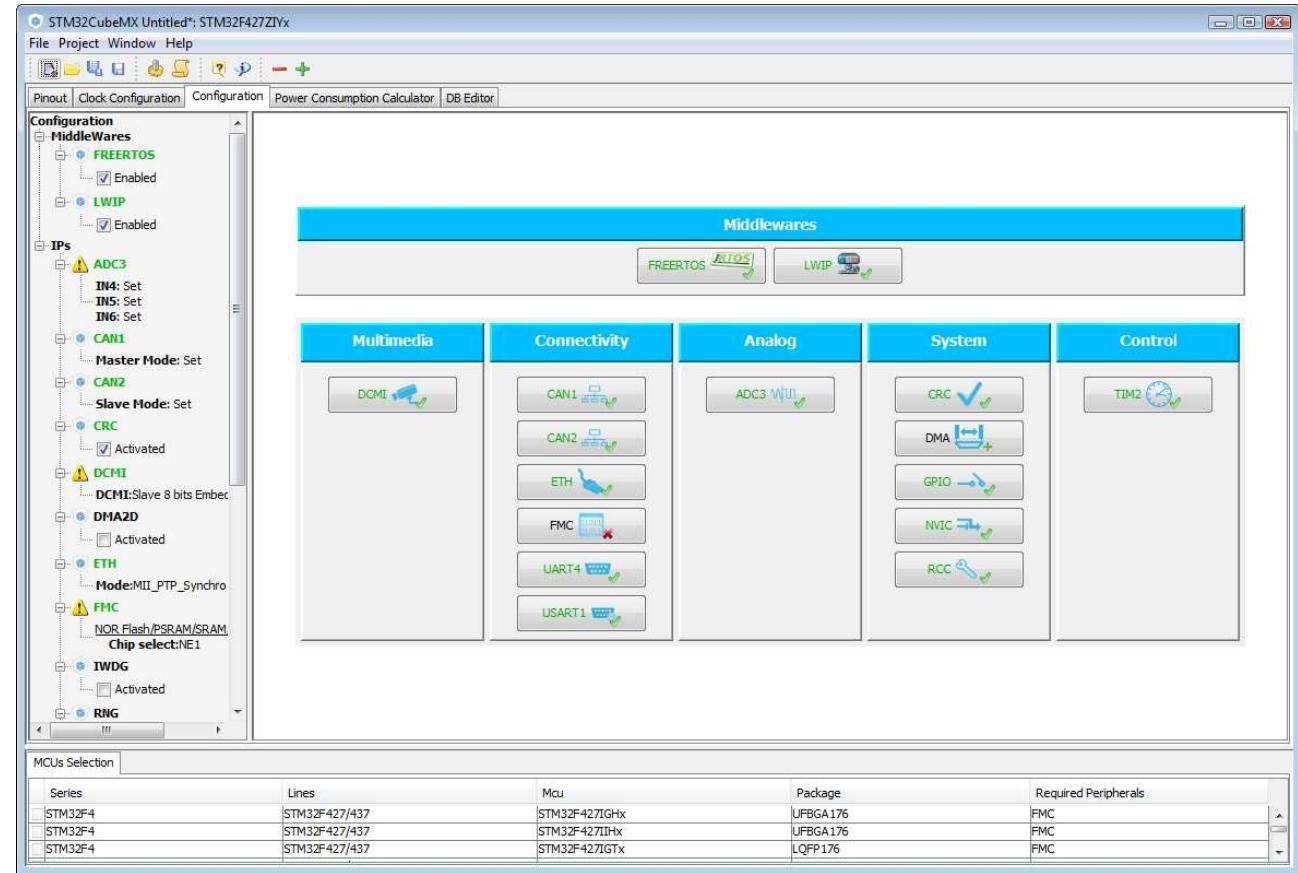
STM32CubeMX : Clock Tree

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



STM32CubeMX : Peripheral Configuration

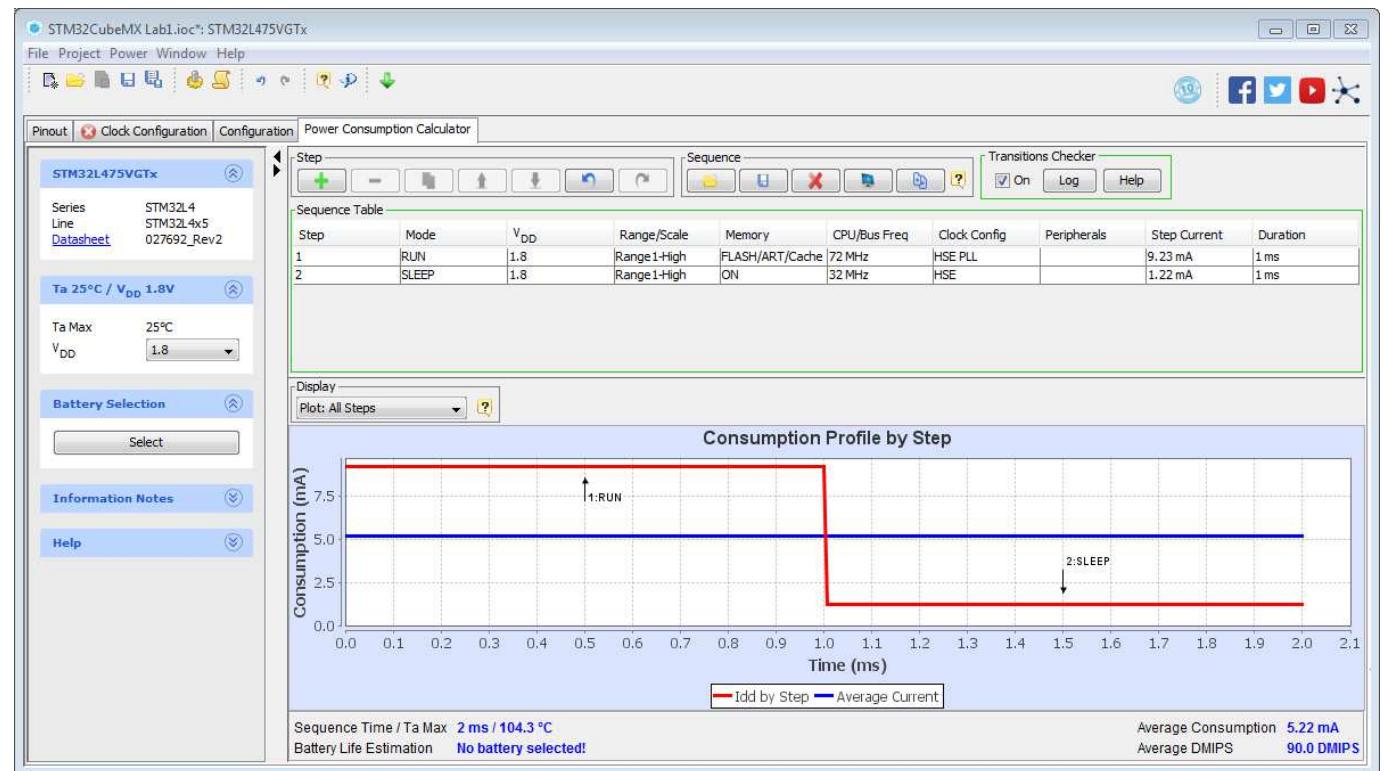
- Global view of used peripherals and middleware
- Highlight of configuration errors
- Manage:
 - GPIO
 - Interrupts
 - DMA



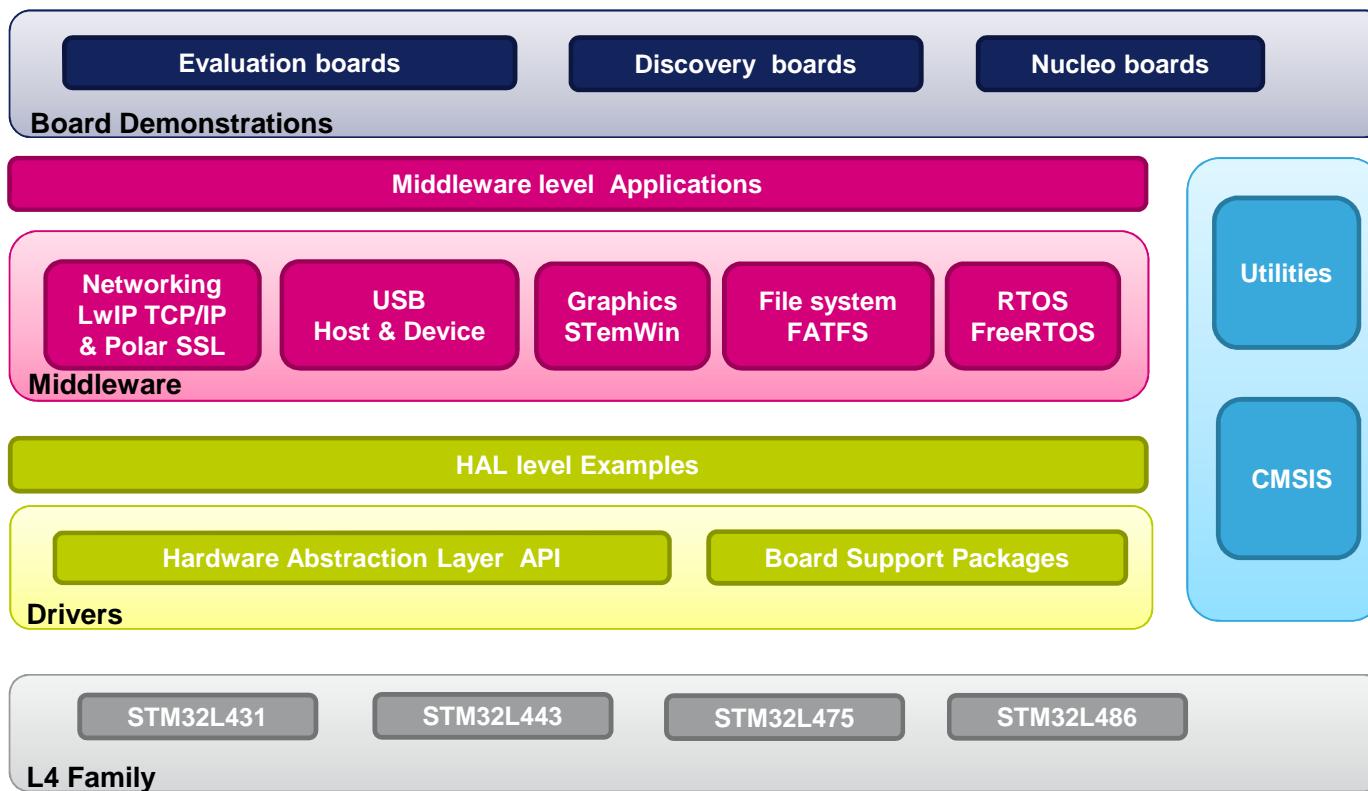
STM32CubeMX : Power Consumption Calculator

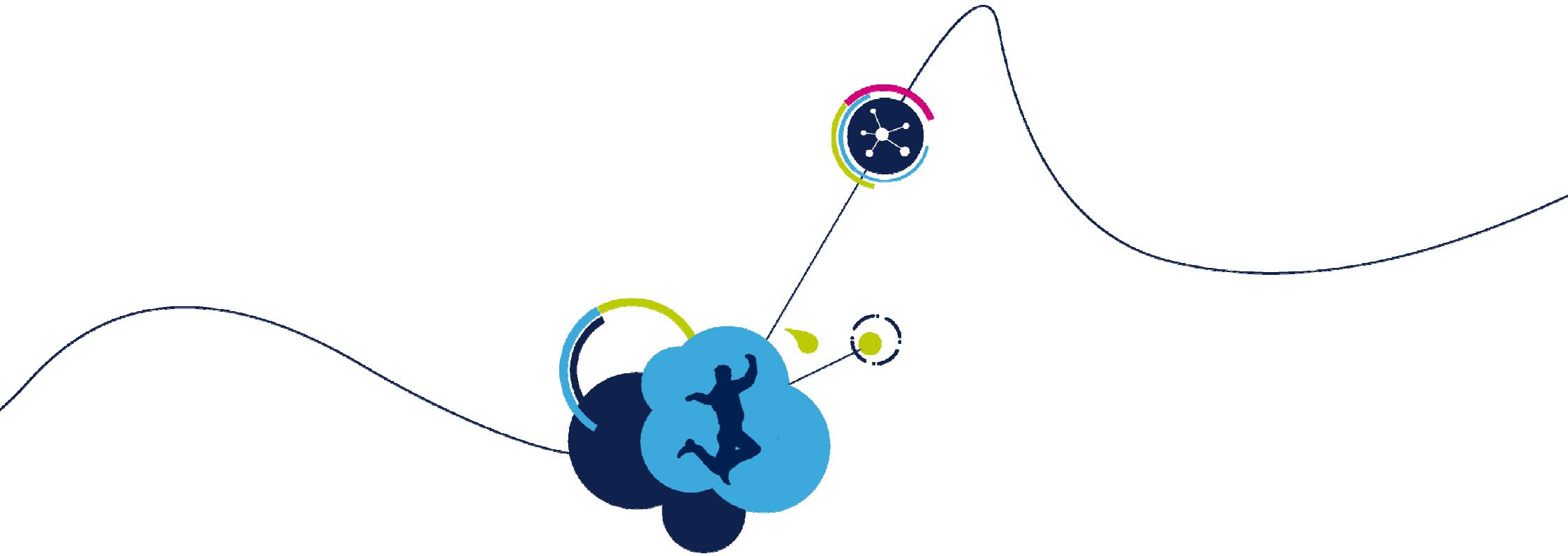
40

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



STM32Cube : Firmware Components



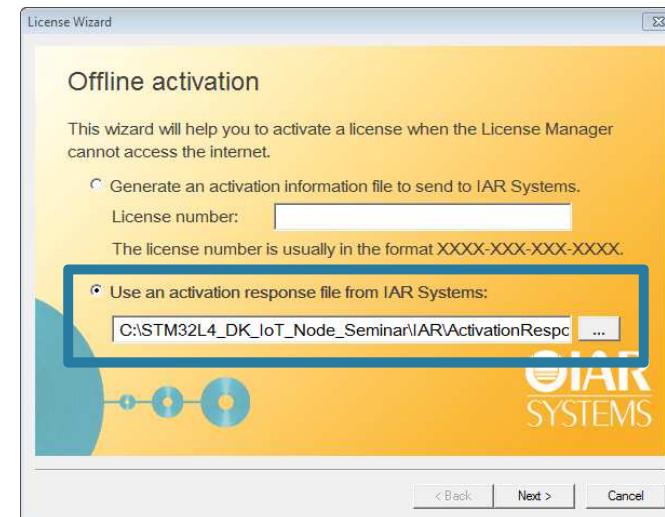
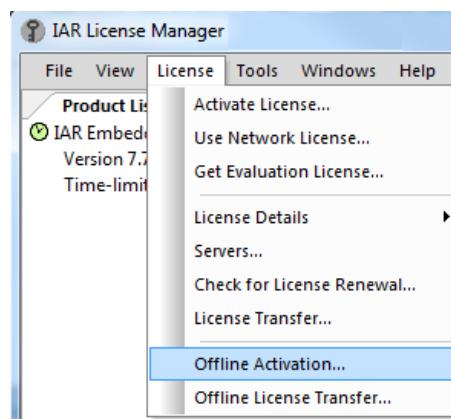


IAR License Installation

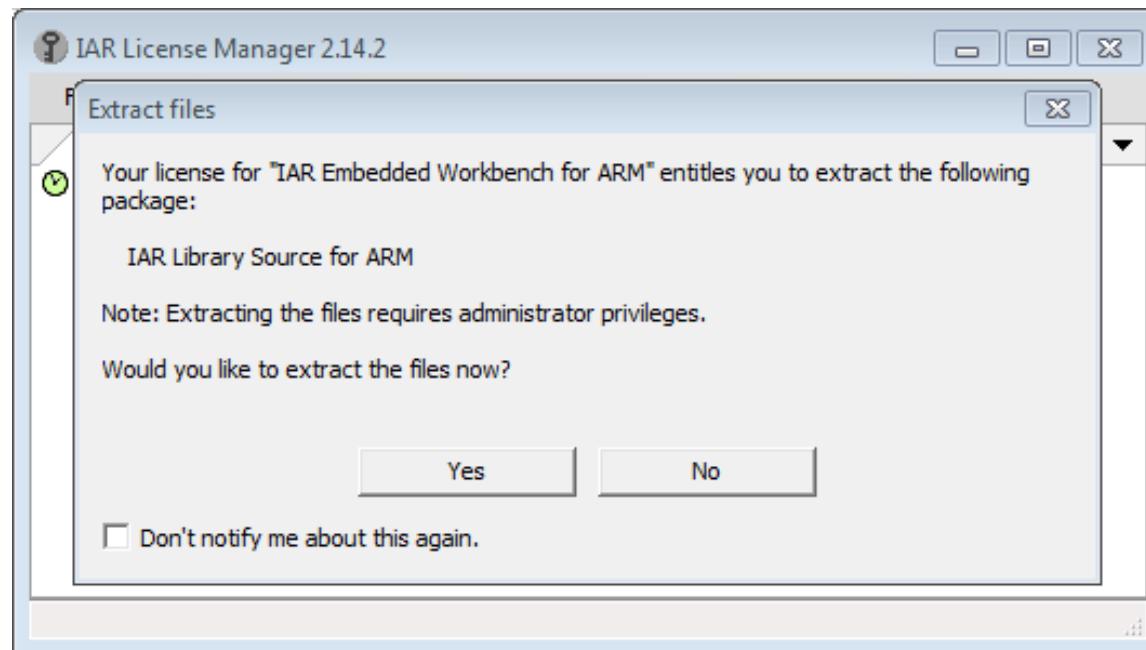
IAR License Installation

- Open IAR
- Go to Help->License Manager
- Go to License->Offline Activation...
- The activation response file is located here:

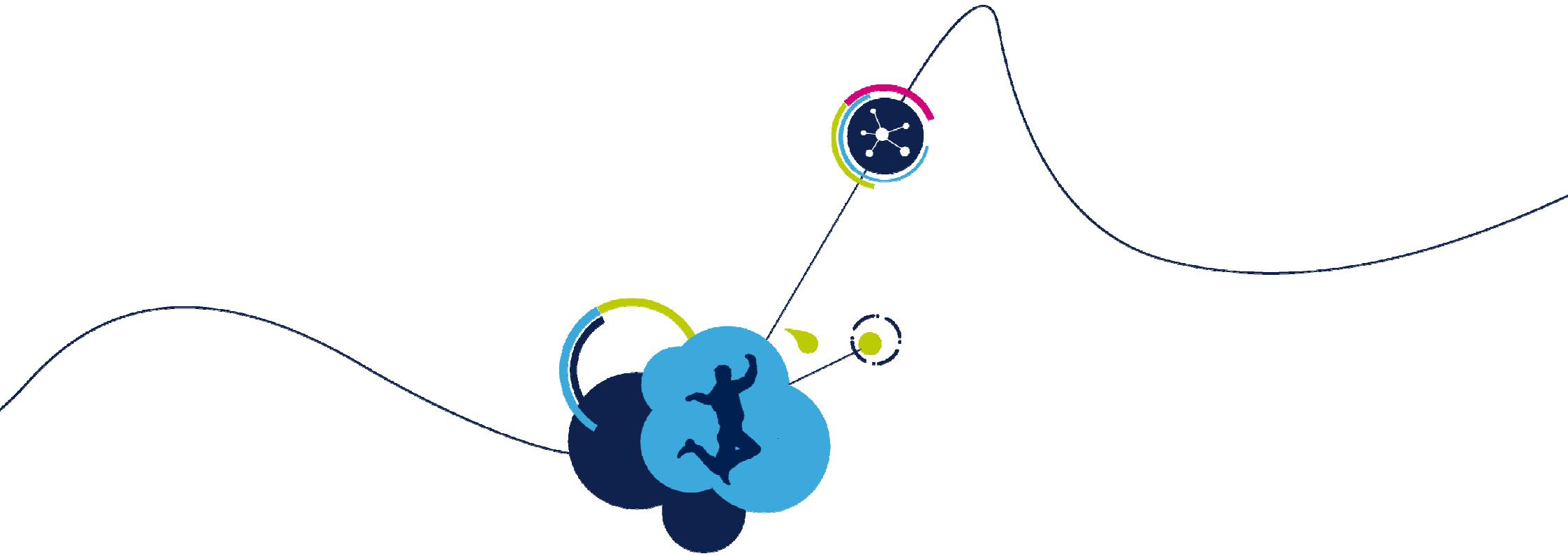
`C:\STM32L4_DK_IoT_Node_Seminar\IAR\ActivationResponse.txt`



IAR License Installation



If you see this pop-up, select "Don't notify me about this again." and click on "No".



ST-Link Utility Installation

ST-Link Utility Installation

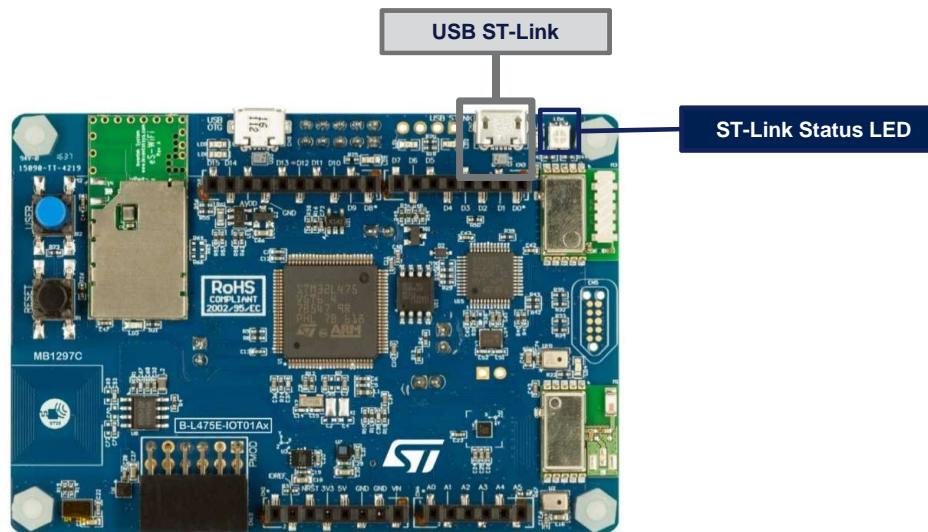
- The ST-Link Utility provides typical flash program / erase / upload / download functions via the ST-LINK/V2 debugger on-board the STM32L475 Discovery Kit IoT Node Board. It also installs the Windows device drivers necessary for the ST-LINK/V2 debugger.
- The ST-Link installer is located here:

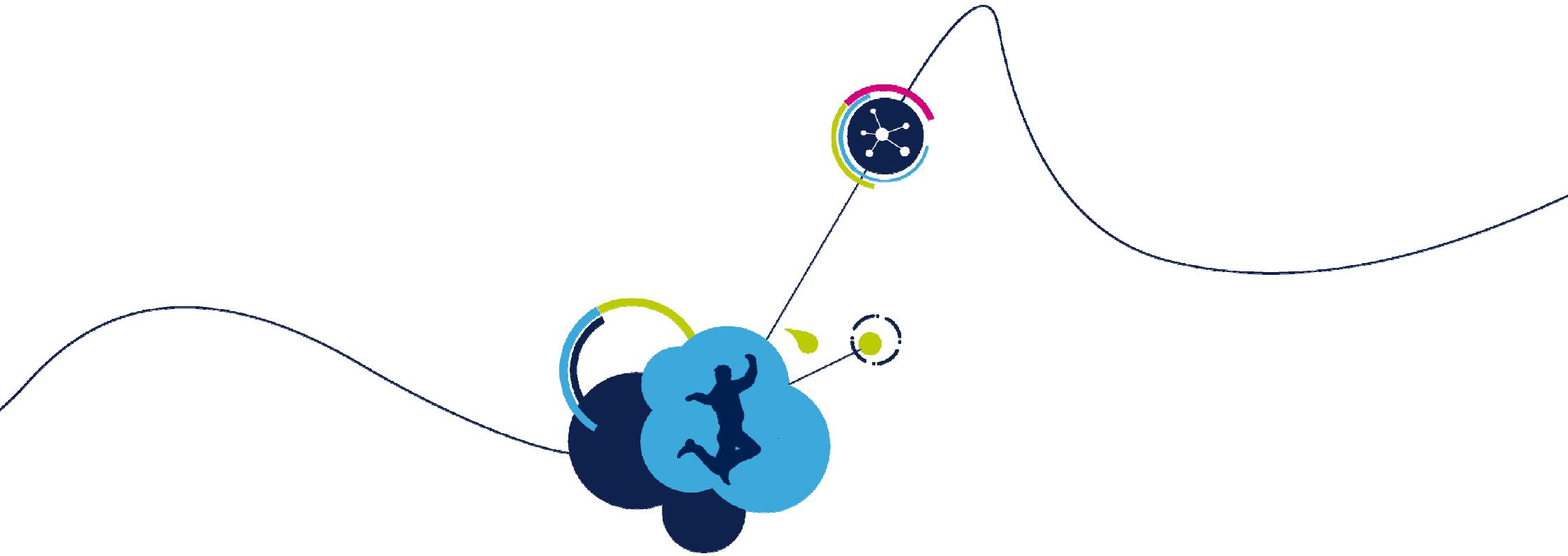
C:\STM32L4_DK_IoT_Node_Seminar\Software

STM32_ST-Link.Utility-4.0.0.exe

ST-Link Driver Installation

- 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 – Blinky LED

Create New STM32CubeMX Project

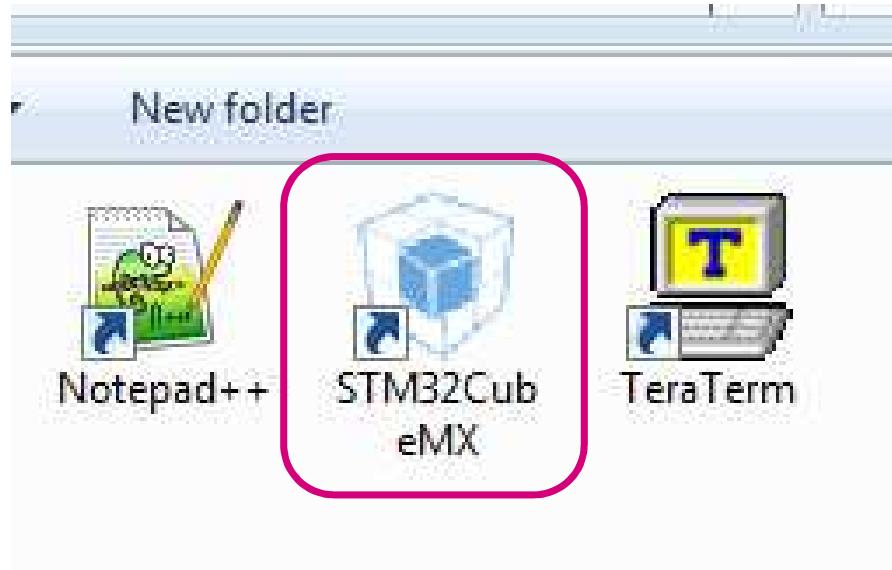
49



- Open the **STM32L4 DK IoT Node Seminar** folder on your Desktop.

Create New STM32CubeMX Project

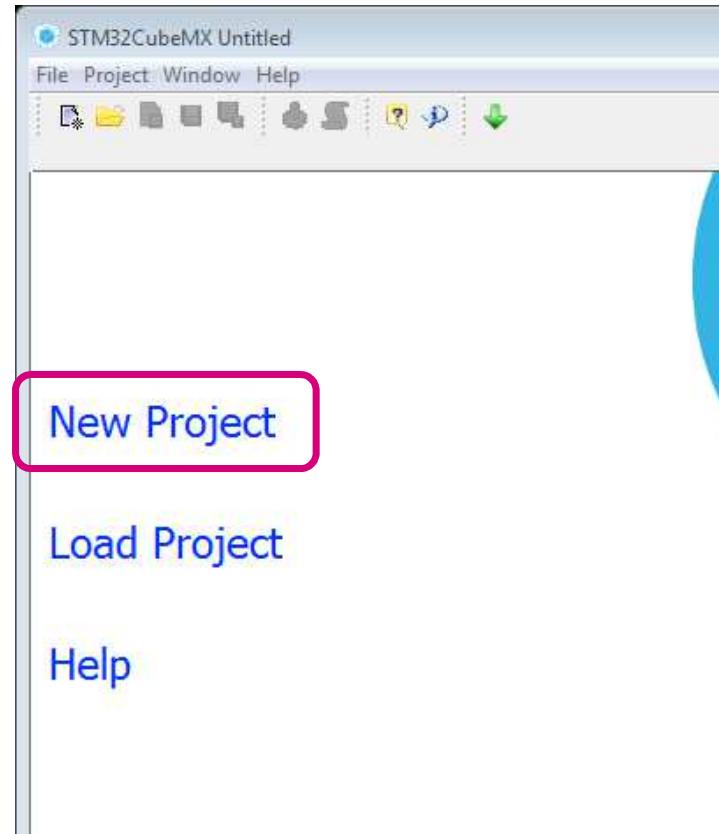
50



- Double-click on the **STM32CubeMX** shortcut

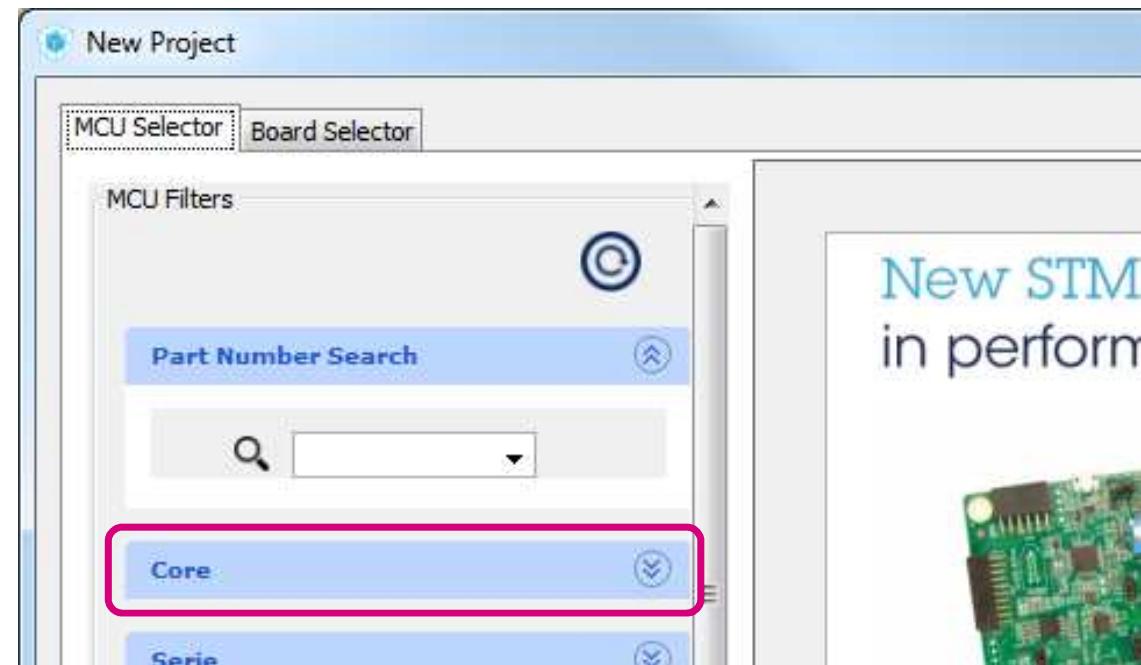
Create New STM32CubeMX Project

- Click on **New Project**



Select the Microcontroller

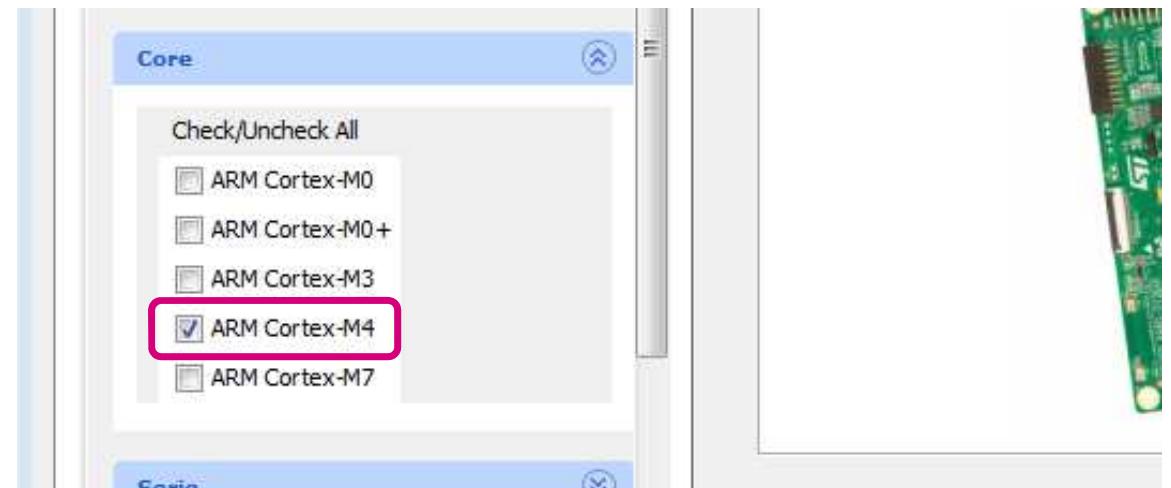
- Expand Core



Select the Microcontroller

53

- Select **ARM Cortex-M4**



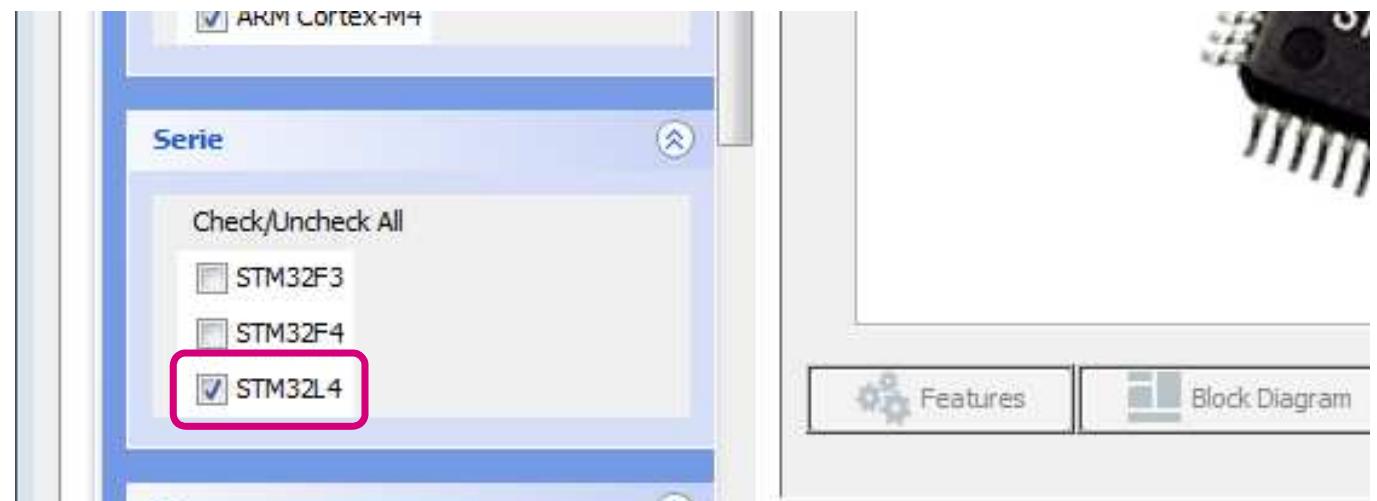
Select the Microcontroller

- Expand Series



Select the Microcontroller

- Select **STM32L4**



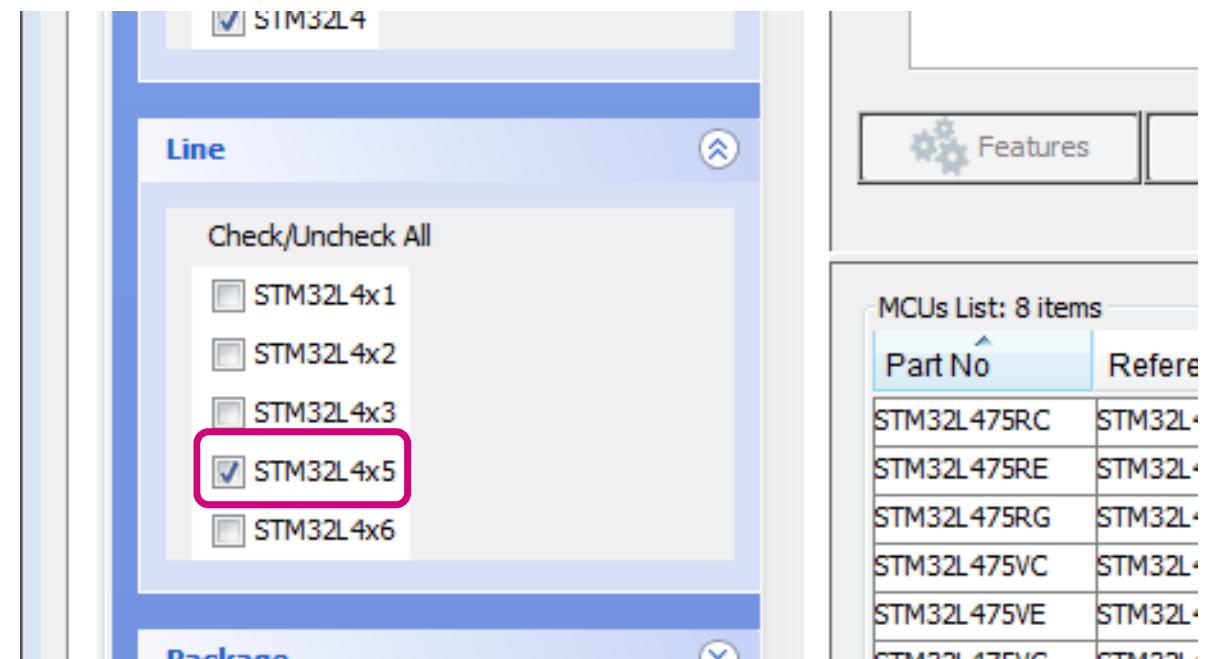
Select the Microcontroller

- Expand Line



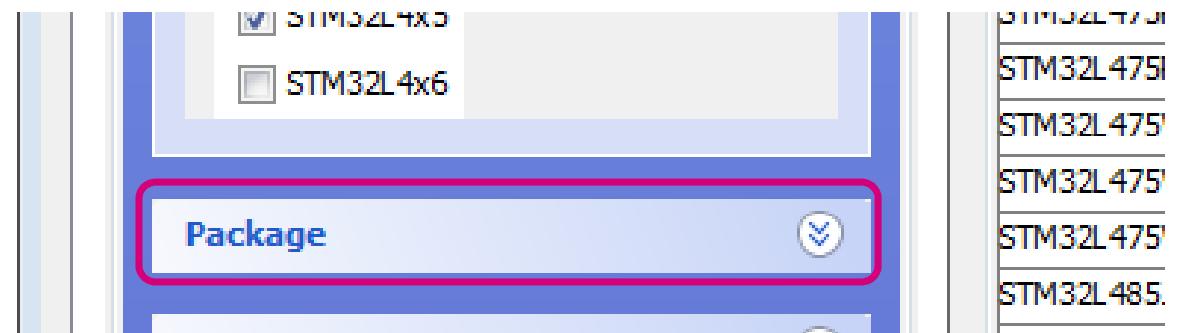
Select the Microcontroller

- Select **STM32L4x5**

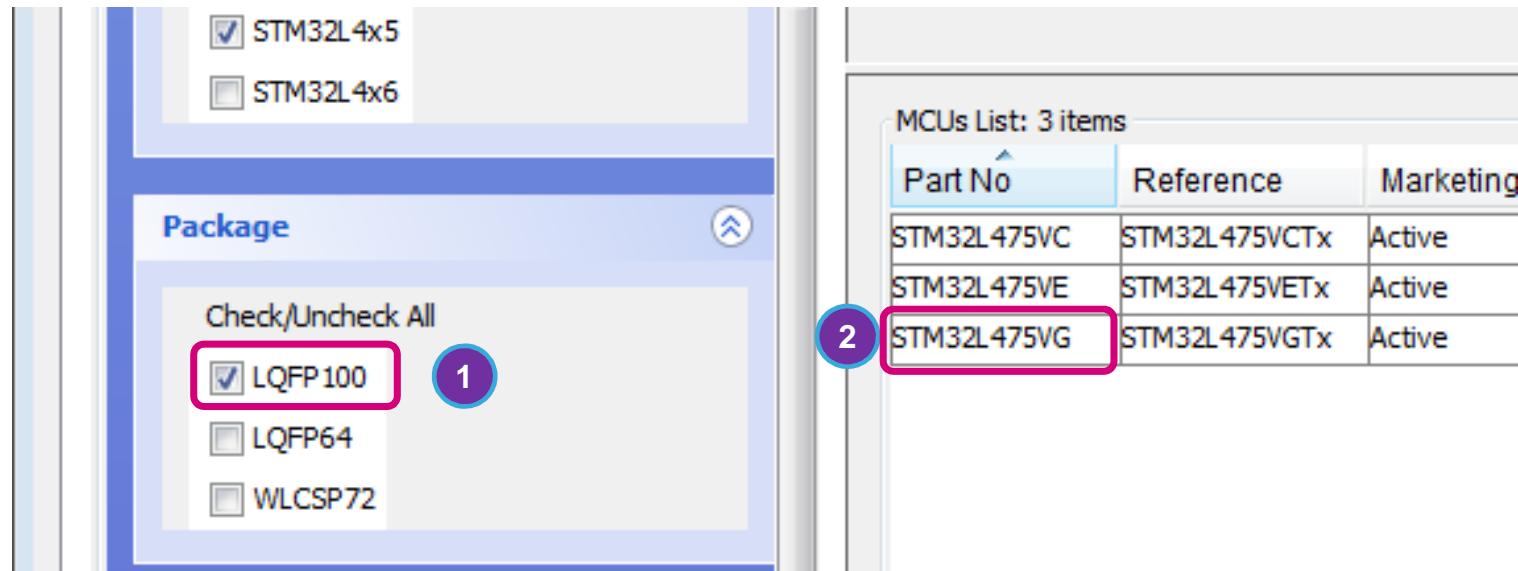


Select the Microcontroller

- Expand Package



Select the Microcontroller



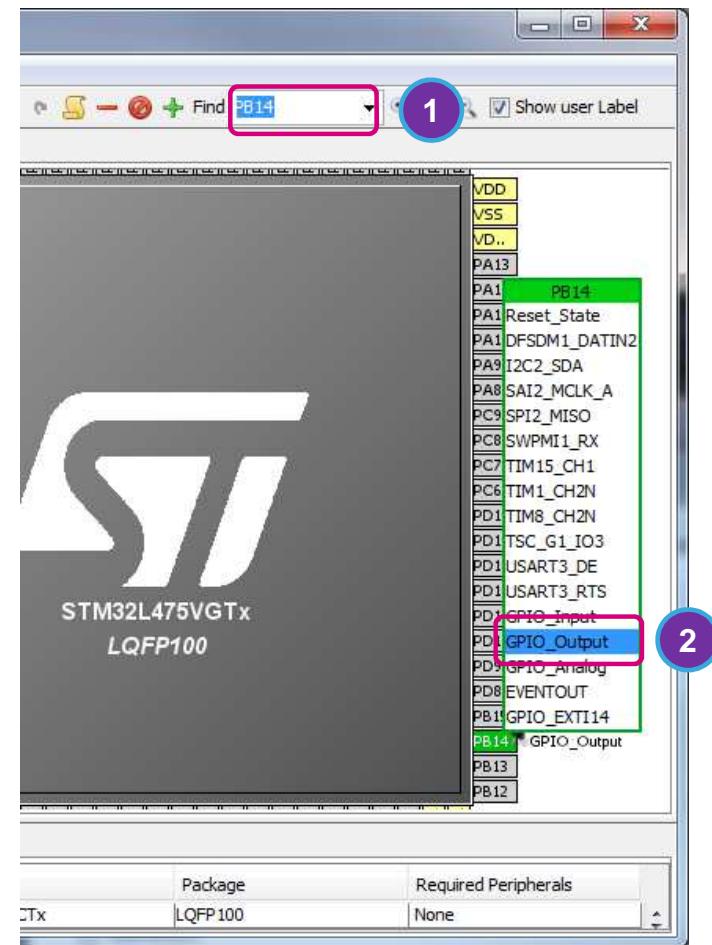
1. Select **LQFP100**
2. Double-click on **STM32L475VG**

GPIO Selection

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

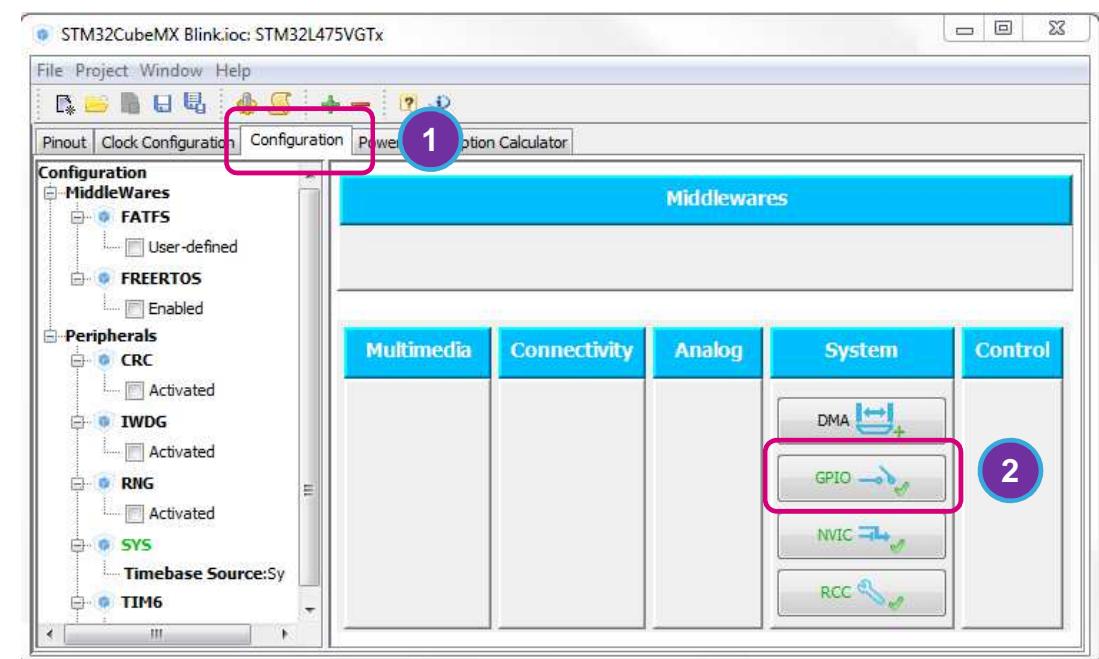


- Use the find toolbar and type **PB14**
- Select **PB14** and set it to **GPIO_Output** mode.



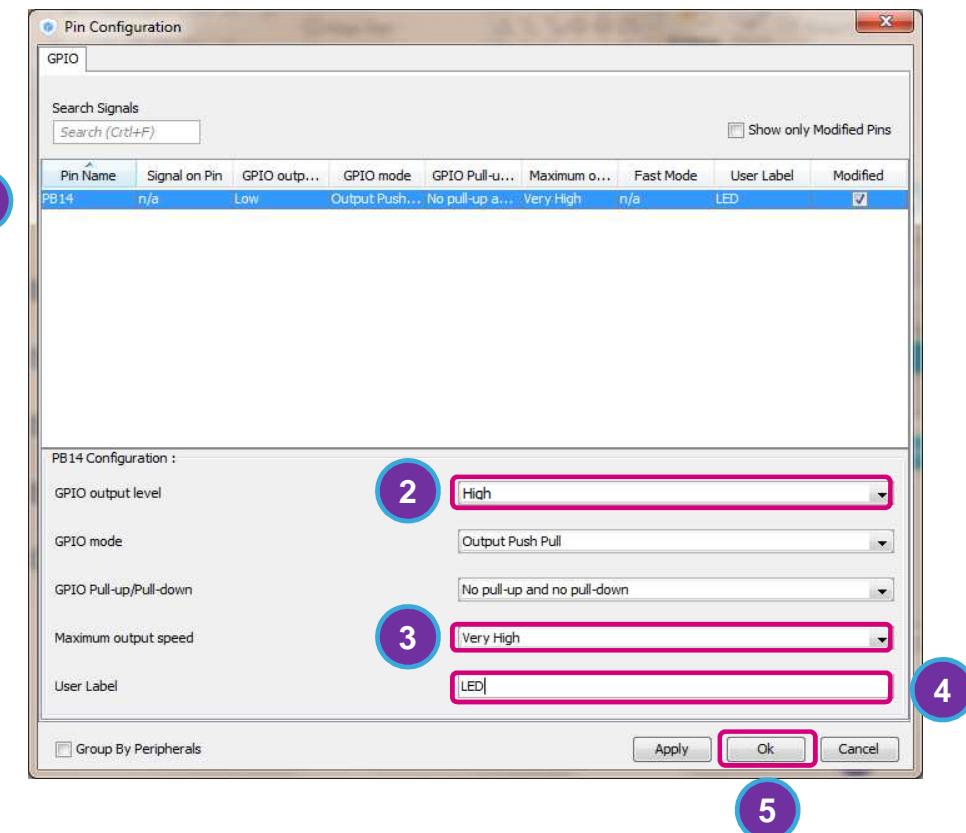
GPIO Configuration

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



GPIO Configuration

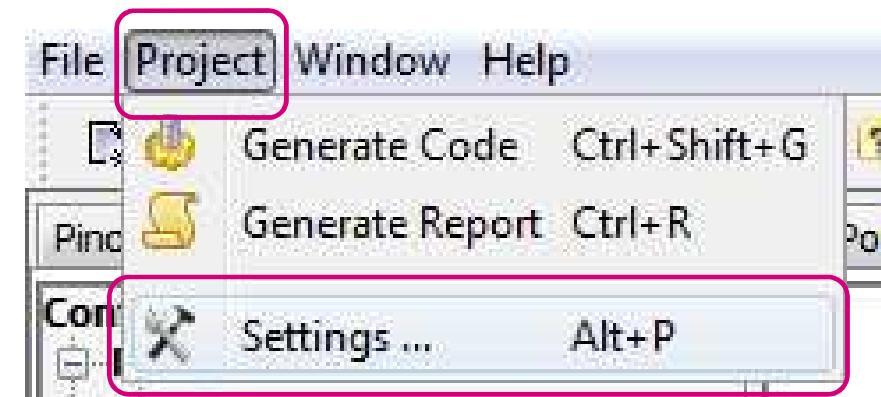
1. Select **PB14**.
2. Set the GPIO output level to **High**.
3. Set the Maximum output speed to **Very High**.
4. Set the User Label to **LED**.
5. Click OK



Project Settings

63

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



Project Settings

64

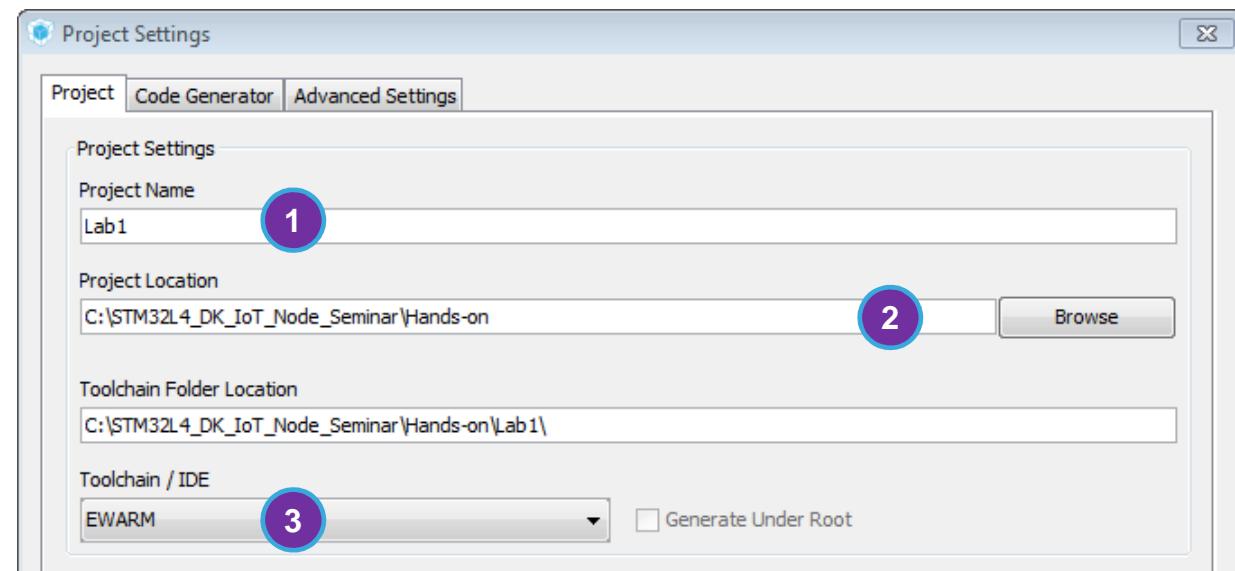
1. Set the project name to **Lab1**.

2. Set the project location:

**C:\STM32L4_DK_IoT_Node_Seminar
\Hands-on**

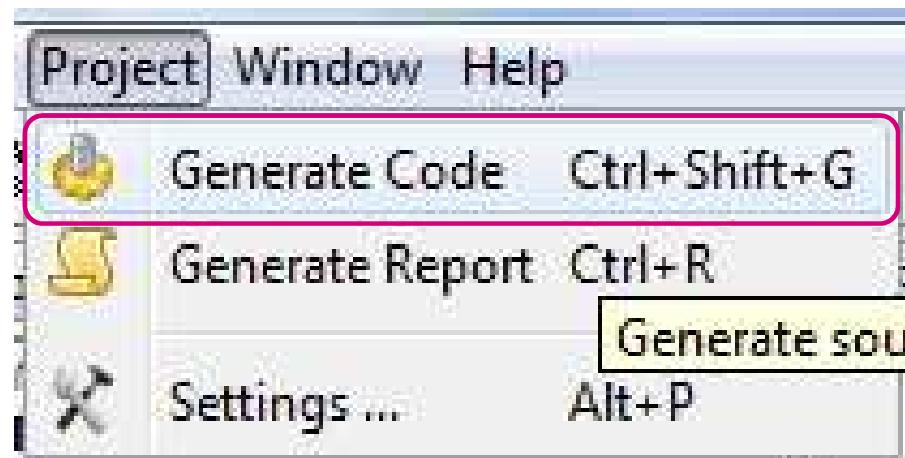
3. Set the IDE Toolchain to **EWARM**.

4. Click **OK**.



Generate the Project

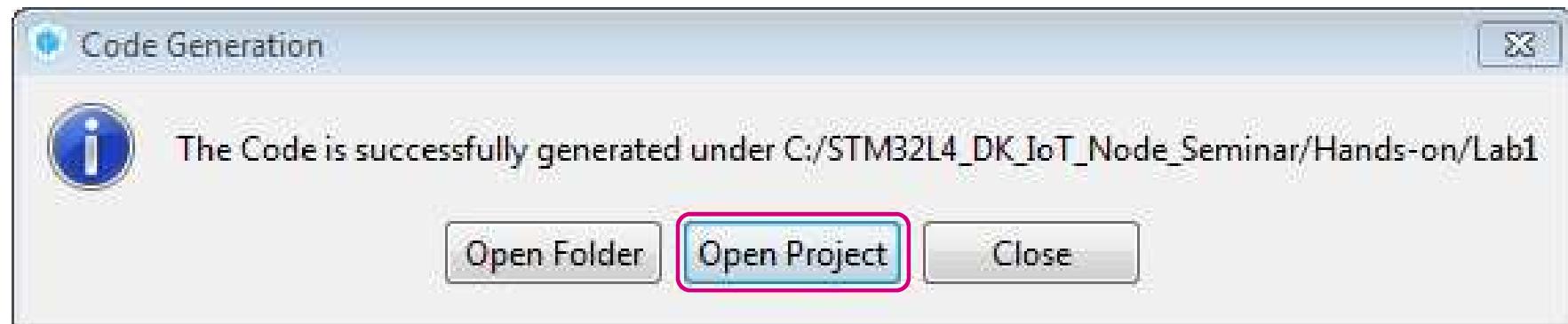
65



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

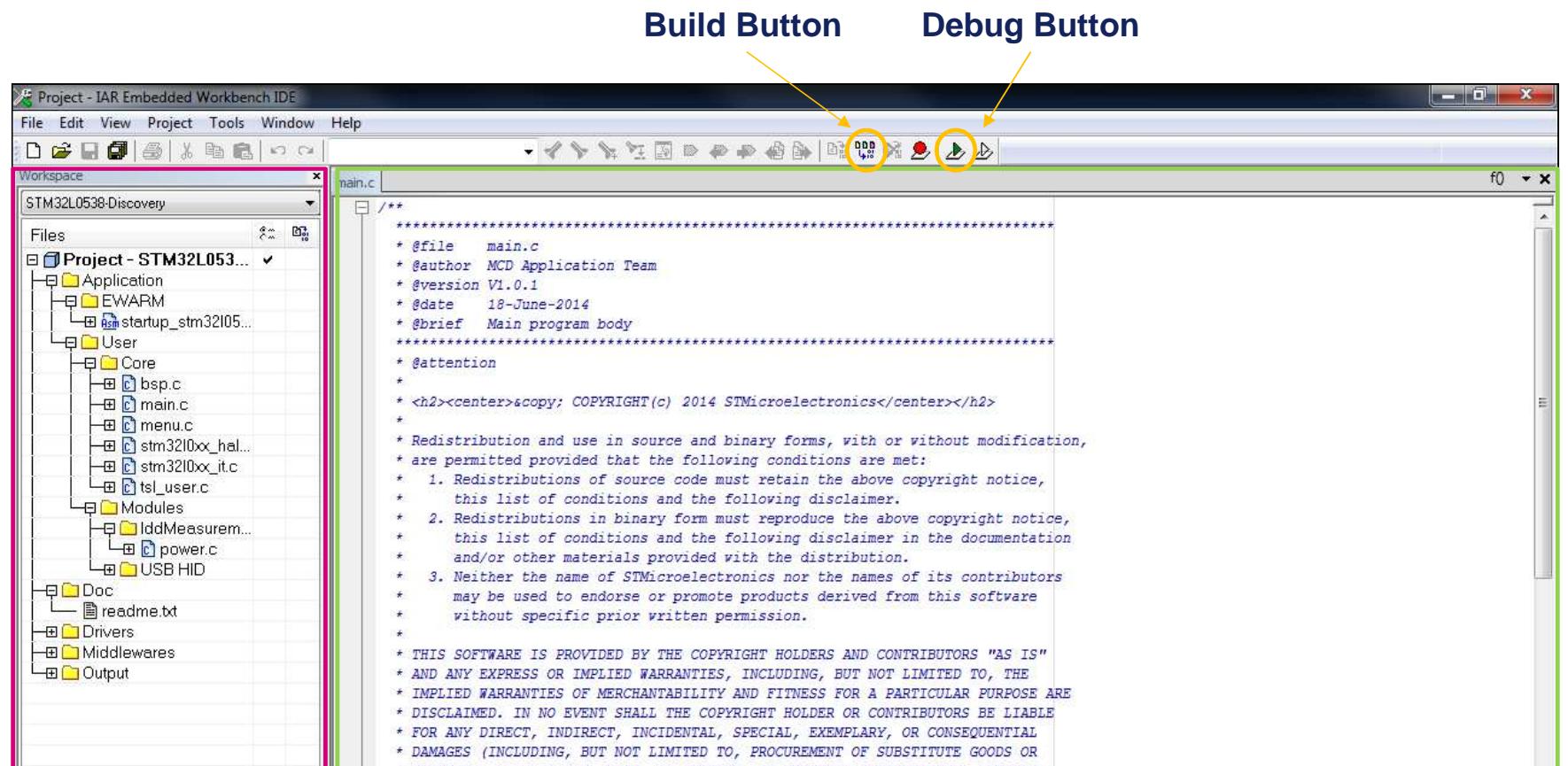
Open the Project

66



- Click **Open Project**

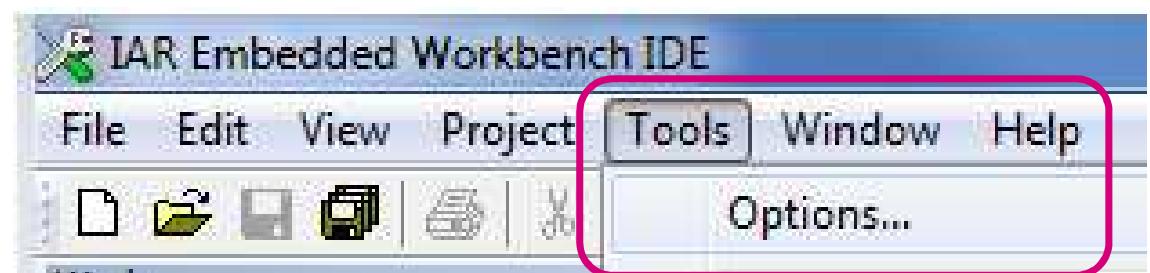
Inside IAR EWARM



Configure IAR to Show Line Numbers

68

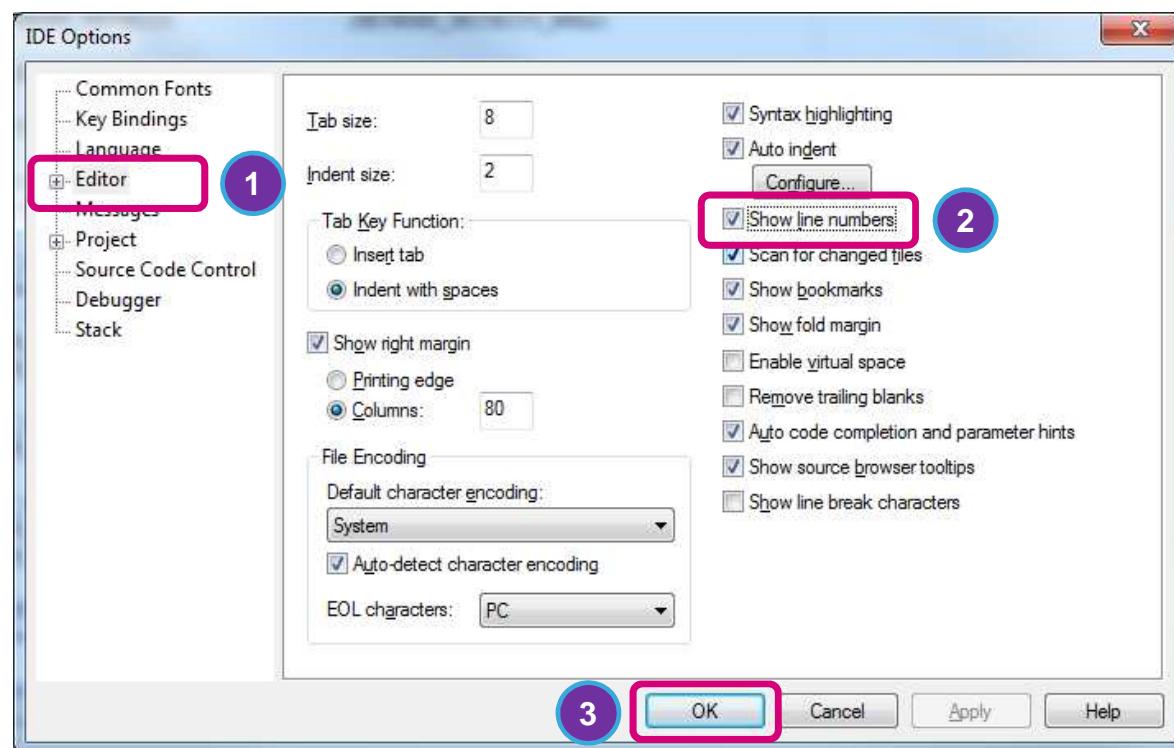
1. Go to Tools → Options



Configure IAR to Show Line Numbers

69

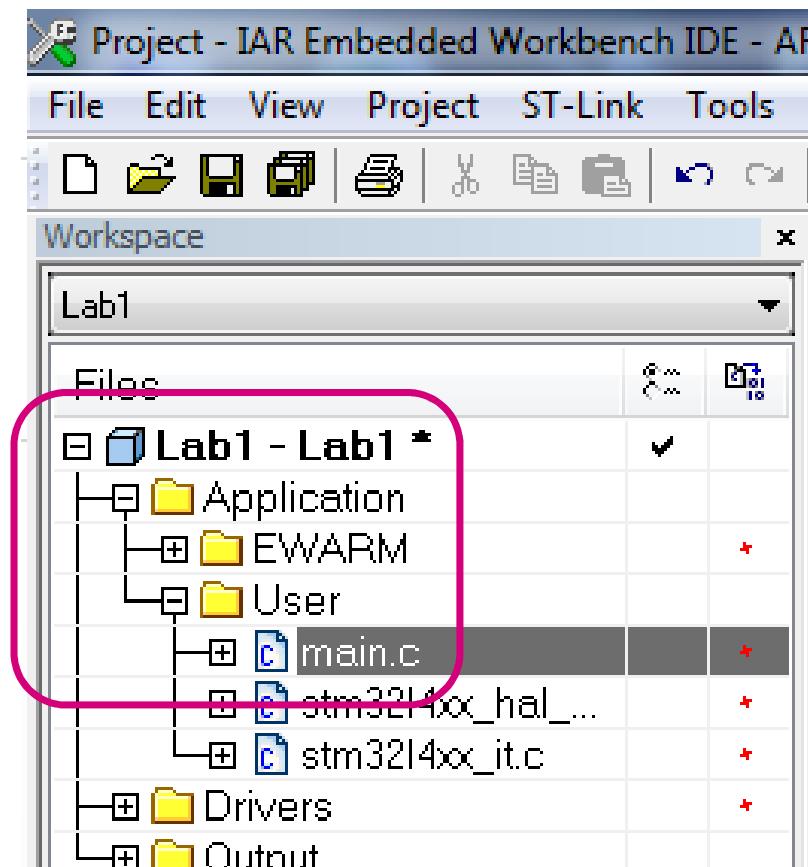
1. Select Editor
2. Check 'Show Line Numbers'
3. Click OK



Edit main.c

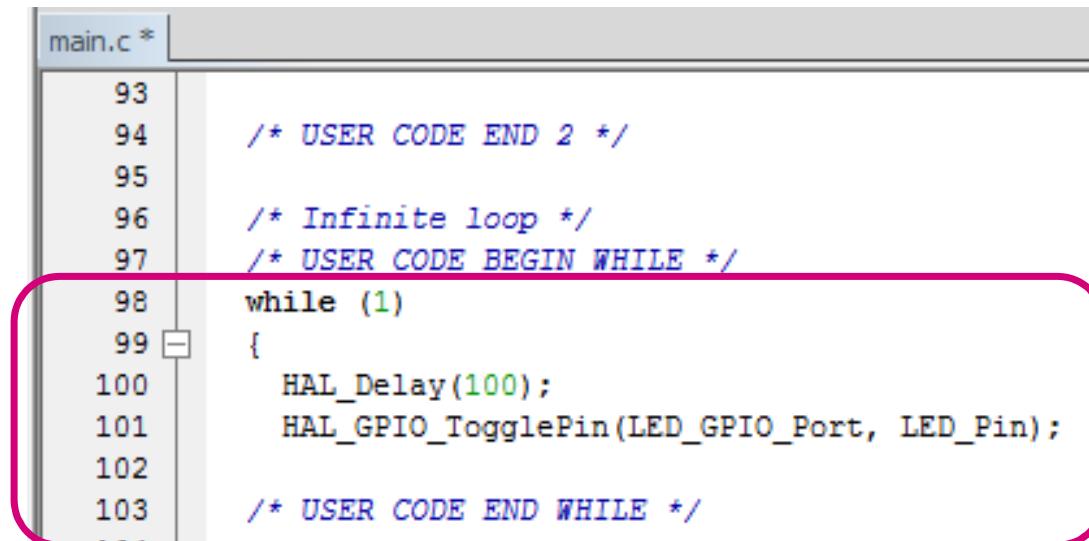
70

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



Edit main.c

71



```
main.c *
93
94     /* USER CODE END 2 */
95
96     /* Infinite loop */
97     /* USER CODE BEGIN WHILE */
98     while (1)
99     {
100         HAL_Delay(100);
101         HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
102
103     /* USER CODE END WHILE */
```

Add the following code inside the **while(1)** loop (line 98):

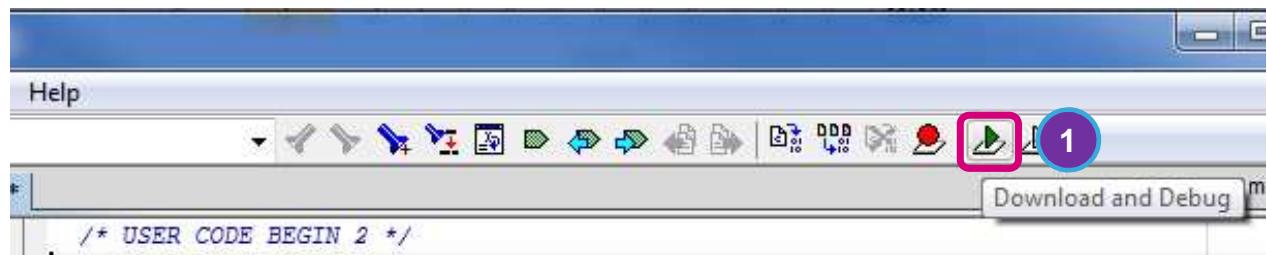
HAL_Delay(100);

HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);

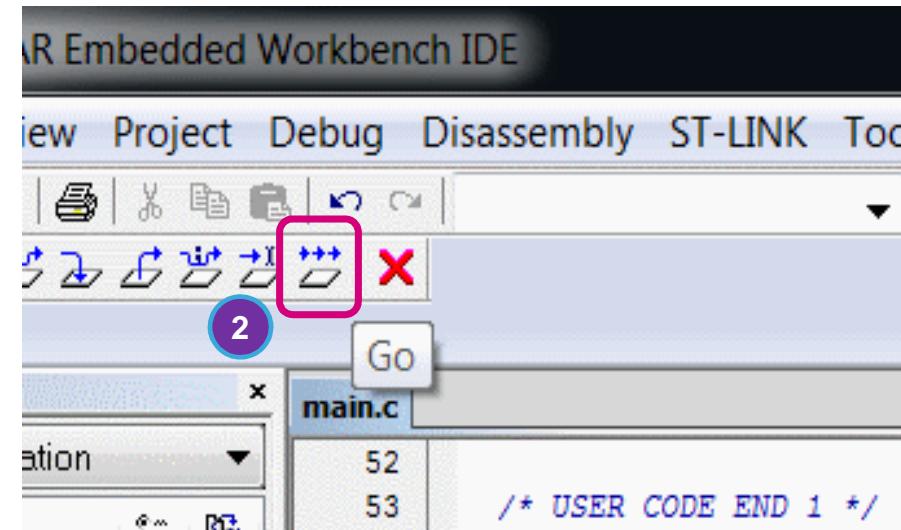
Load and Run

72

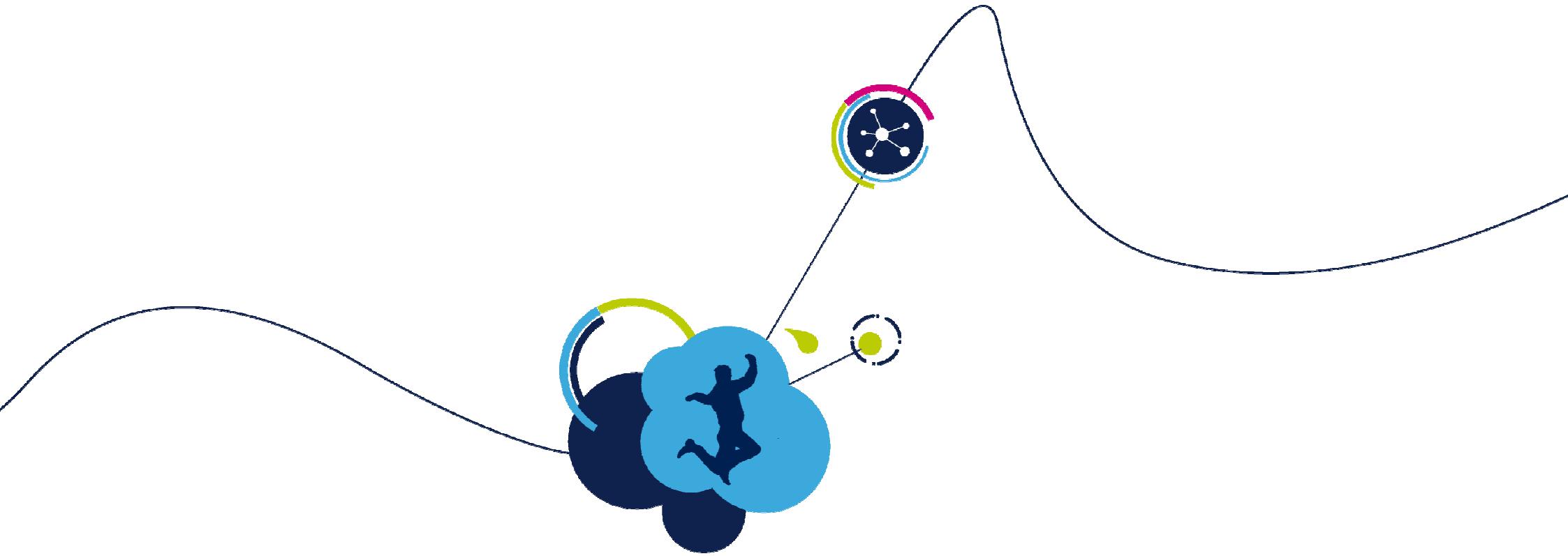
1. Click the GREEN ARROW to Build the Project, **Download** and start the **debugger**. (Ctrl + D)



2. Click the triple-arrow GO button! (F5)



The board LED is now blinking!



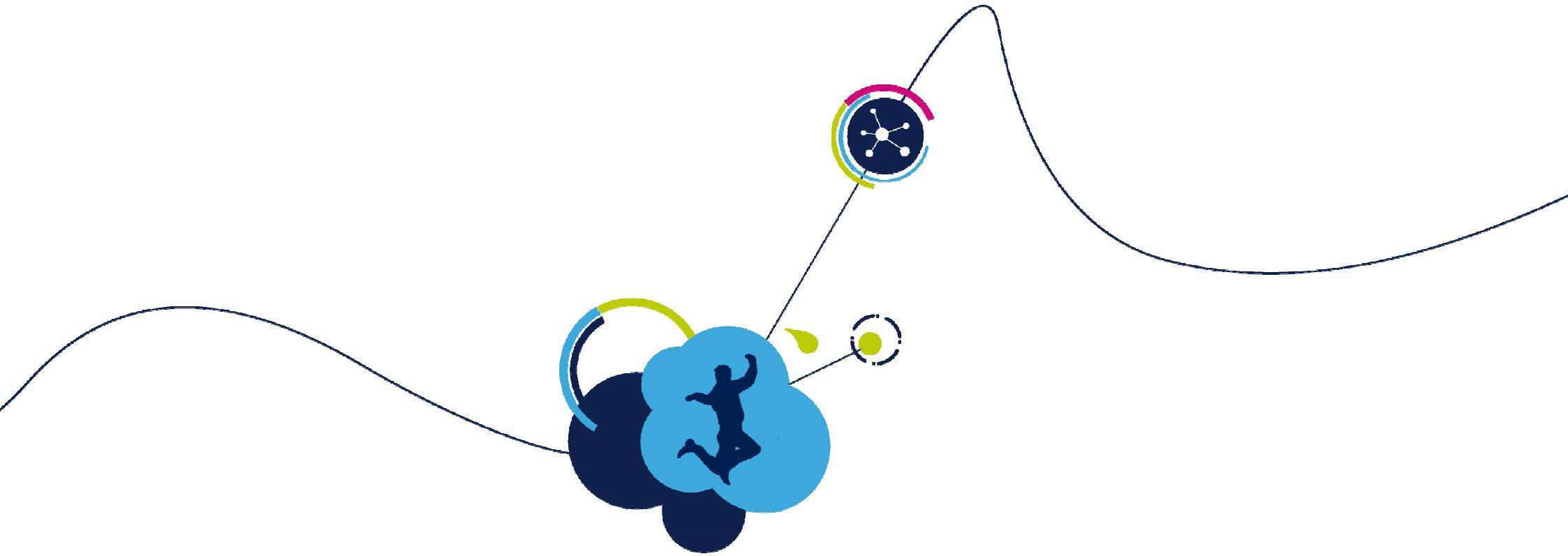
USB Overview

USB Overview

74

- The Universal Serial Bus is a plug and play serial interface designed to easily connect peripherals to a host.
- The physical USB network is implemented as a star network with one host (master) and up to 127 devices (slaves).
- USB uses two wires for power and two wires for data. It supports multiple data transfer speeds:
 - Low-Speed (1.5Mbits/s)
 - Full-Speed (12Mbits/s)
 - Hi-Speed (480Mbits/s)
 - SuperSpeed(up to 10Gbits/s).
- The functionality of USB devices is defined by class codes, communicated by the slave to the USB host to load a suitable driver.

- The **Communication Device Class** supports a wide range of devices that can perform telecommunication and networking functions.
- It is mostly used as a Virtual COM Port and for USB to Wi-Fi or USB to Ethernet dongles.
- The USB Virtual COM Port is an upgrade from the old RS232 serial port. It is a software interface that enables applications to access a USB device as if it were a built-in serial port.



Optional Lab : USB Device Using a Virtual COM Port

- In this example we are going to demonstrate how to:
 - Use STM32CubeMX to configure USB
 - Setup the STM32L475 USB_OTG_FS block as a CDC Virtual COM Port device.
 - Use serial terminal software to exchange data between STM32 and a PC over USB.
- Close the previous IAR and STM32CubeMX project.

Create New STM32CubeMX Project

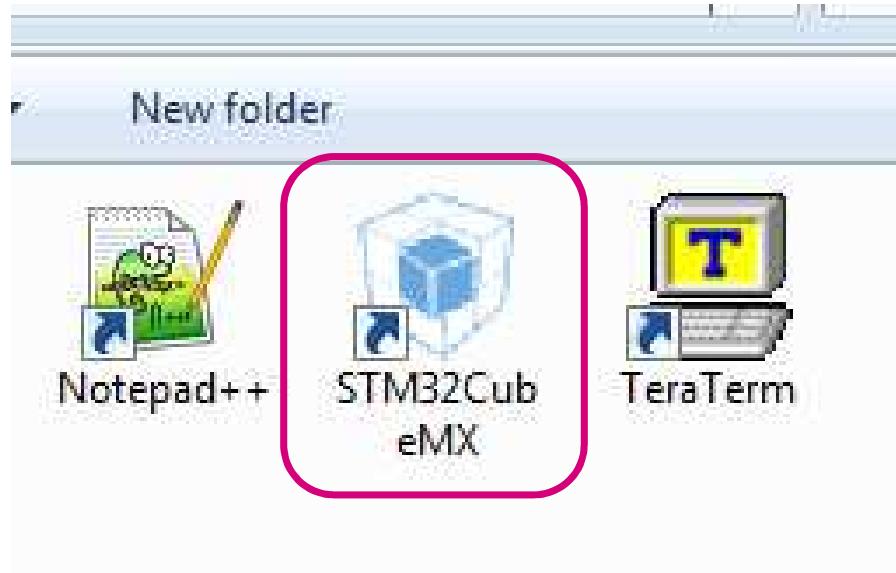
78



- Open the **STM32L4 DK IoT Node Seminar** folder on your Desktop.

Create New STM32CubeMX Project

79

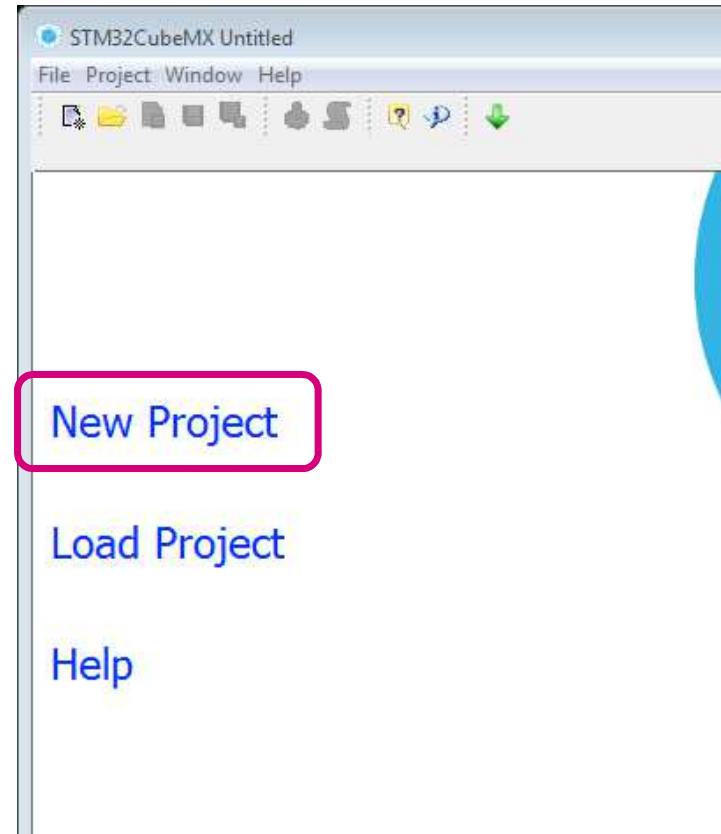


- Double-click on the **STM32CubeMX** shortcut

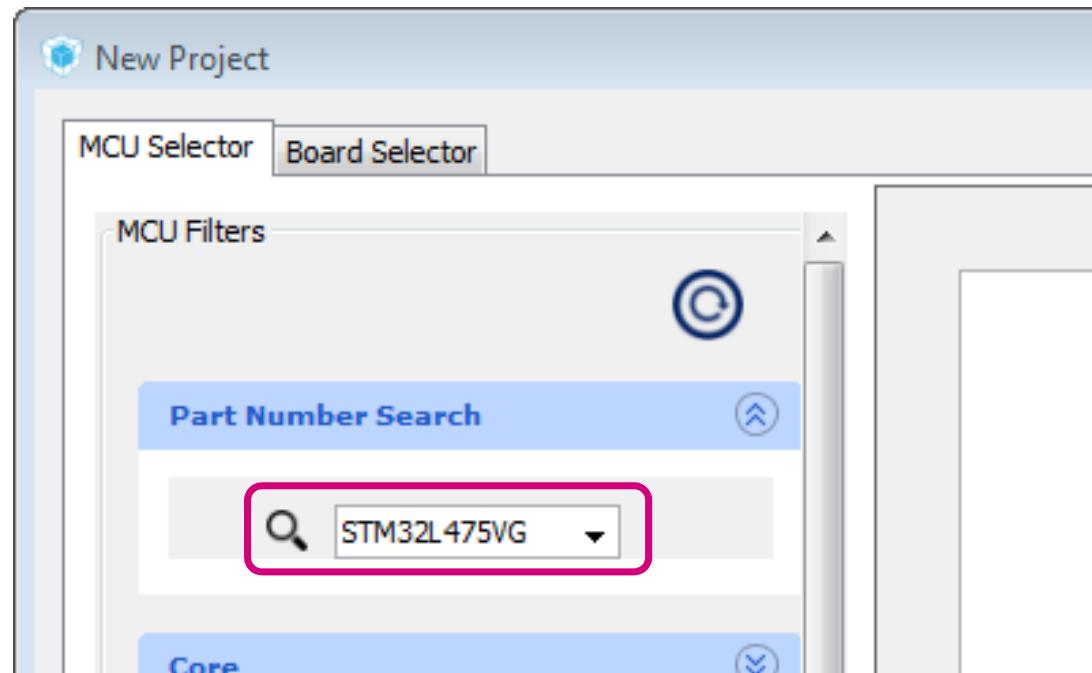
Create New STM32CubeMX Project

80

- Click on **New Project**



Select the Microcontroller



- Click in the **Part Number Search** box and type **STM32L475VG**

Select the Microcontroller

82

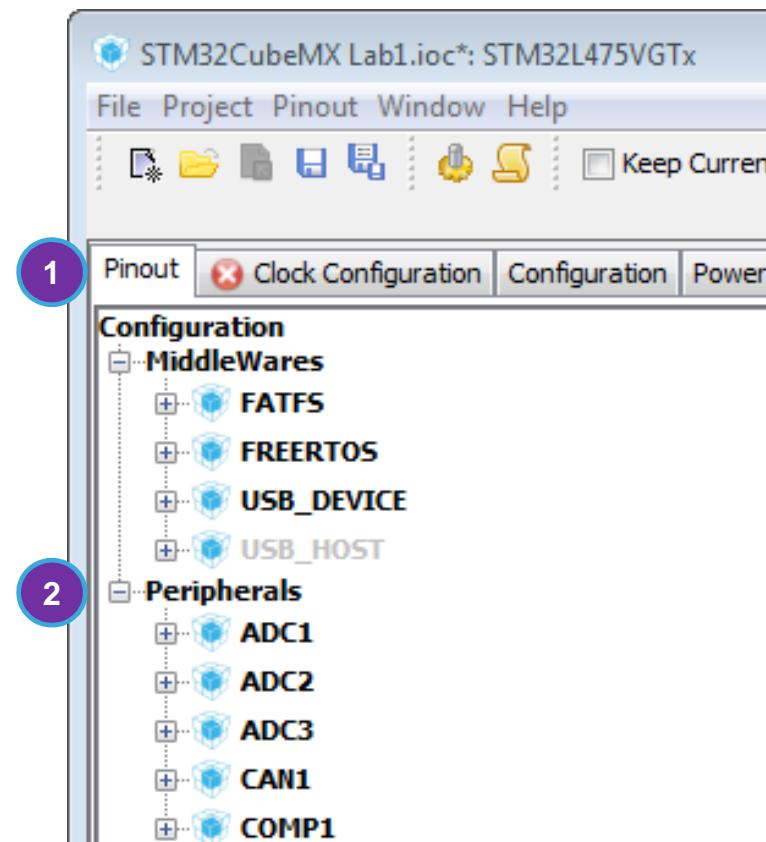
MCUs List: 1 item						
Part No	Reference	Marketing Status	Unit Price for 10kU from U...	Package	Flas	
STM32L475VG	STM32L475VGTx	Active	5.39	LQFP 100	1024	

- Double-click on **STM32L475VG**

Enable USB Device Mode

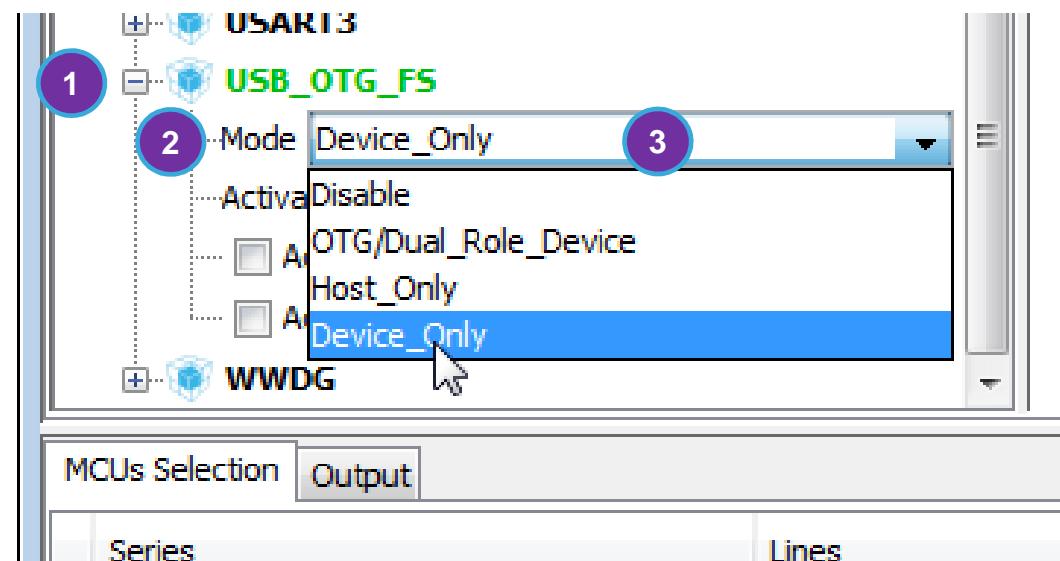
83

1. Select the **Pinout** tab
2. Expand **Peripherals**



Enable USB Device Mode

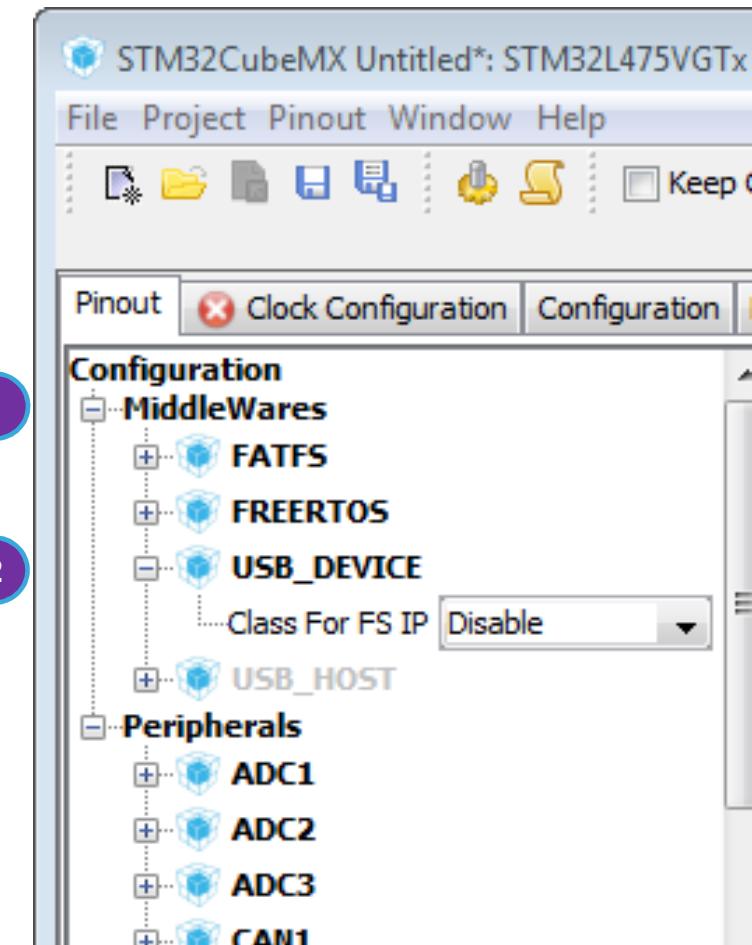
1. Scroll down and select **USB_OTG_FS**
2. Select **Mode**
3. Select **Device_Only**



Enable USB CDC

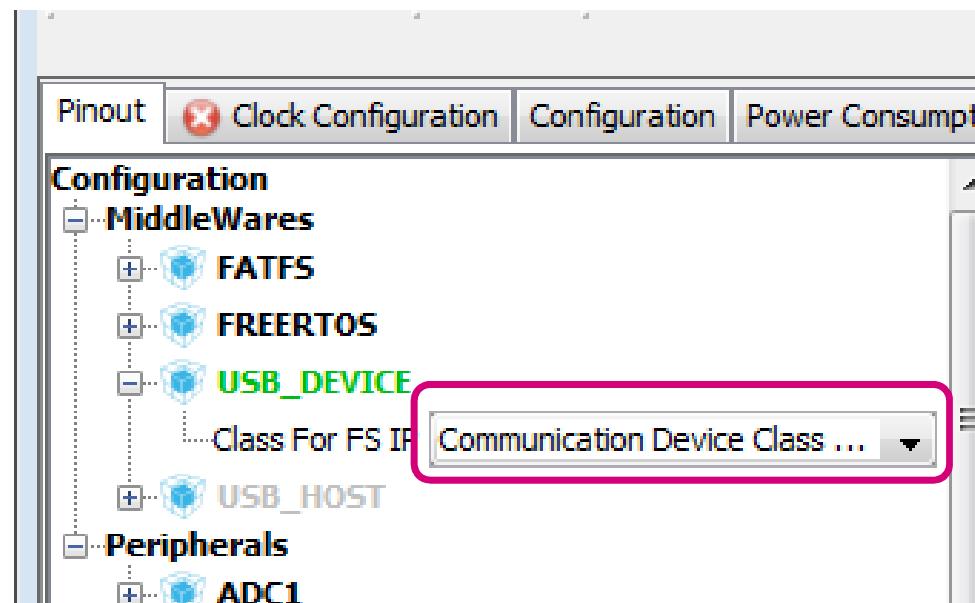
85

1. Scroll up and expand **MiddleWares**
2. Expand **USB_DEVICE**



Enable USB CDC

86

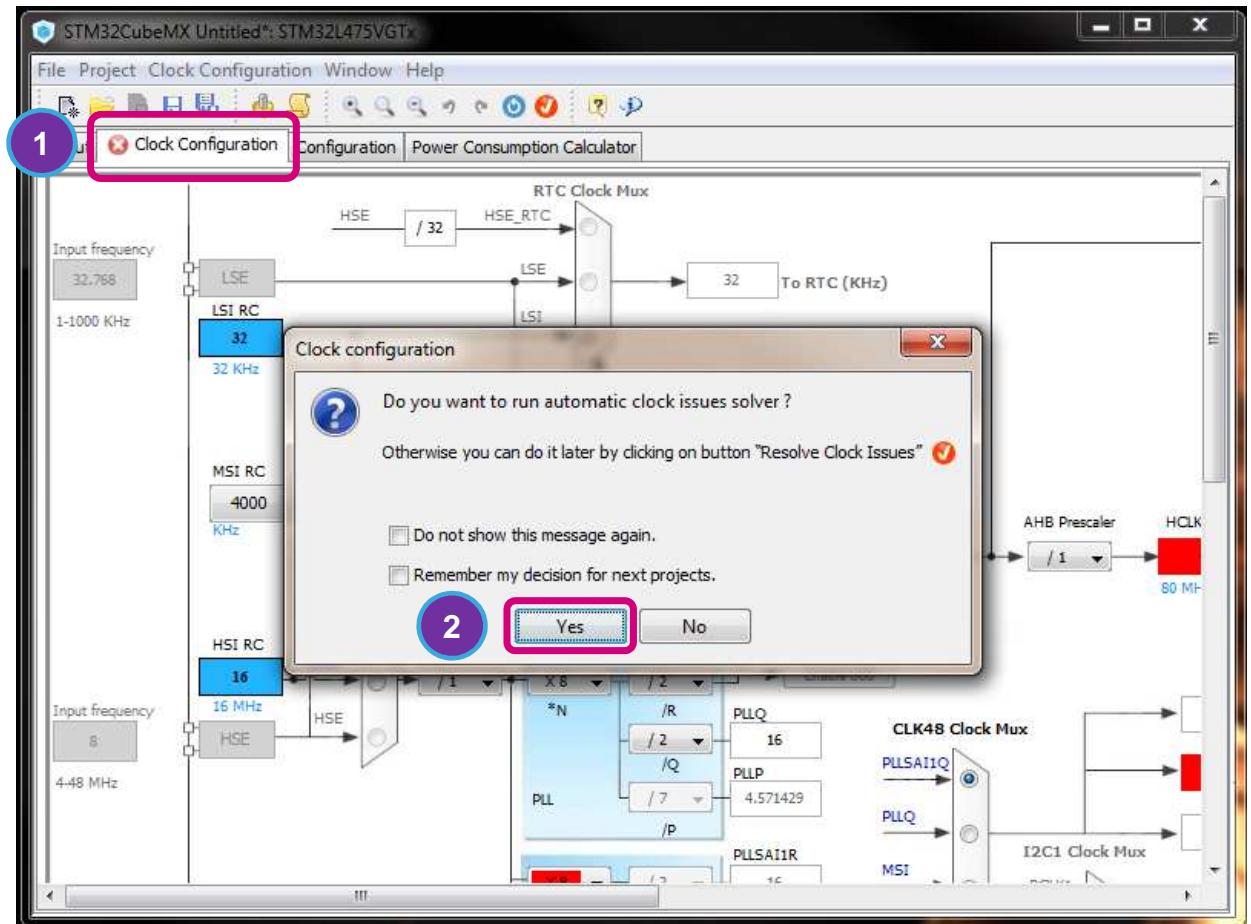


- Select **Communication Device Class**

Clock Configuration

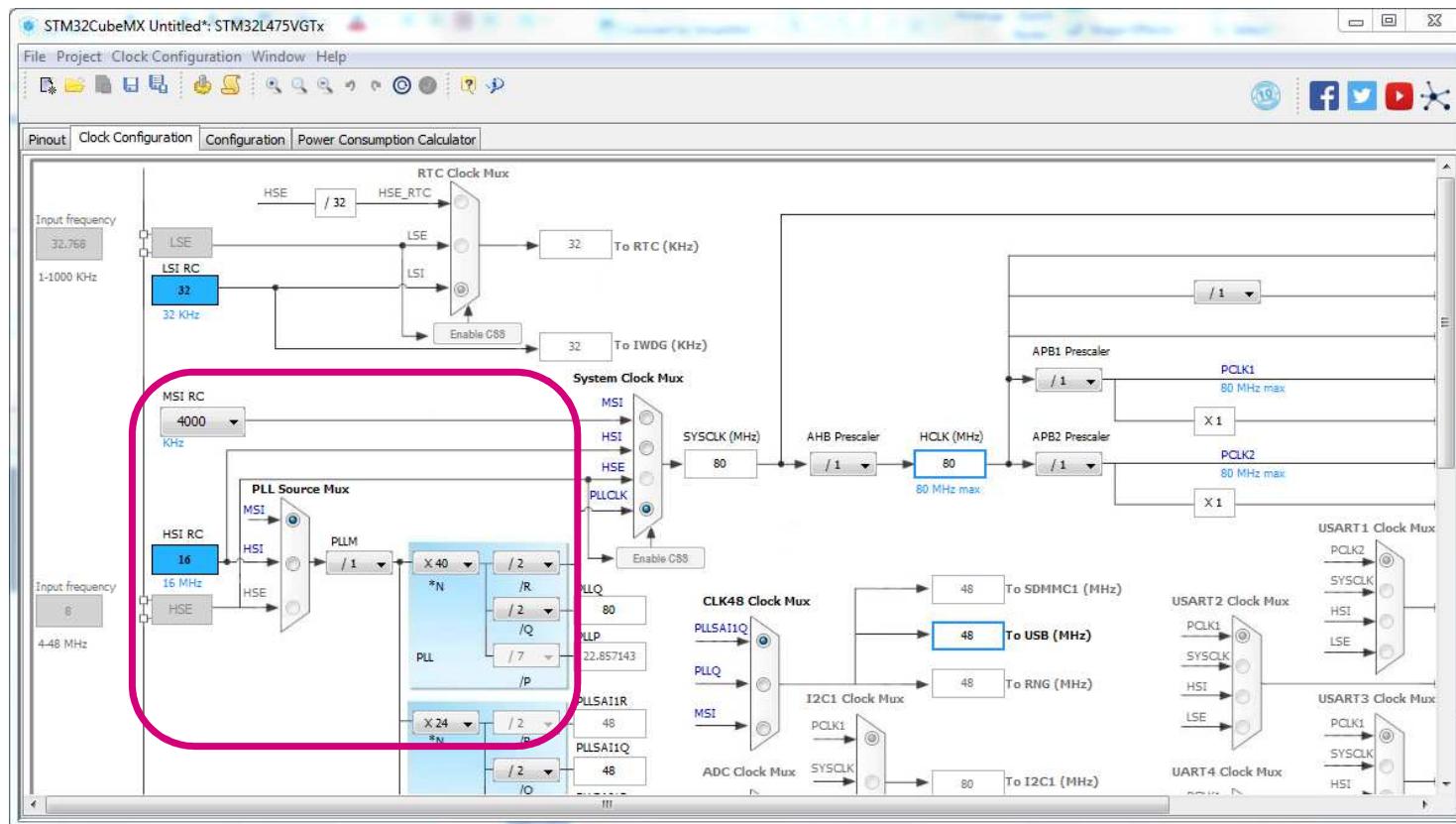
87

1. Select the **Clock Configuration** tab
2. Click **Yes** on the Clock Configuration Message.



Clock Configuration

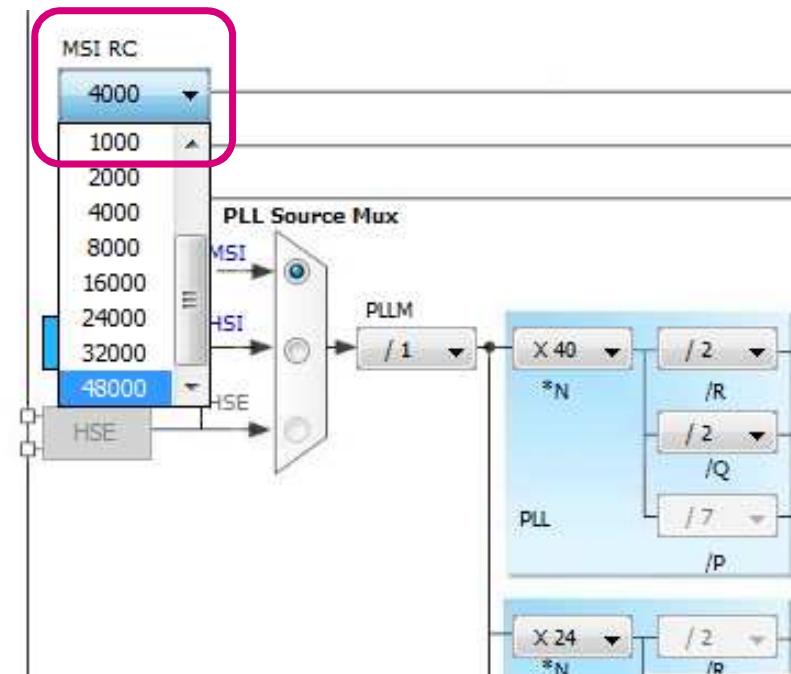
88



Clock Configuration

89

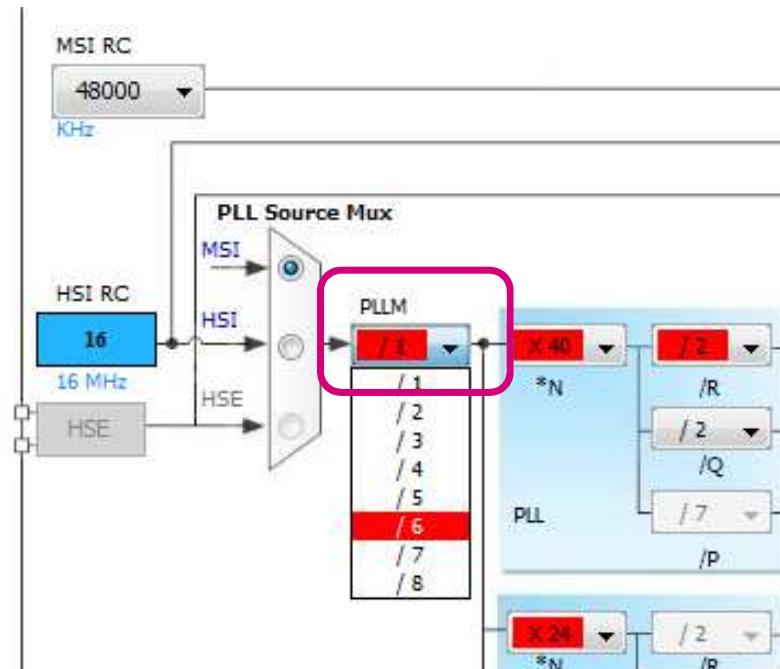
- Select **48000** in the **MSI RC** dropdown menu
- Several other boxes will turn **RED** to indicate conflicts.



Clock Configuration

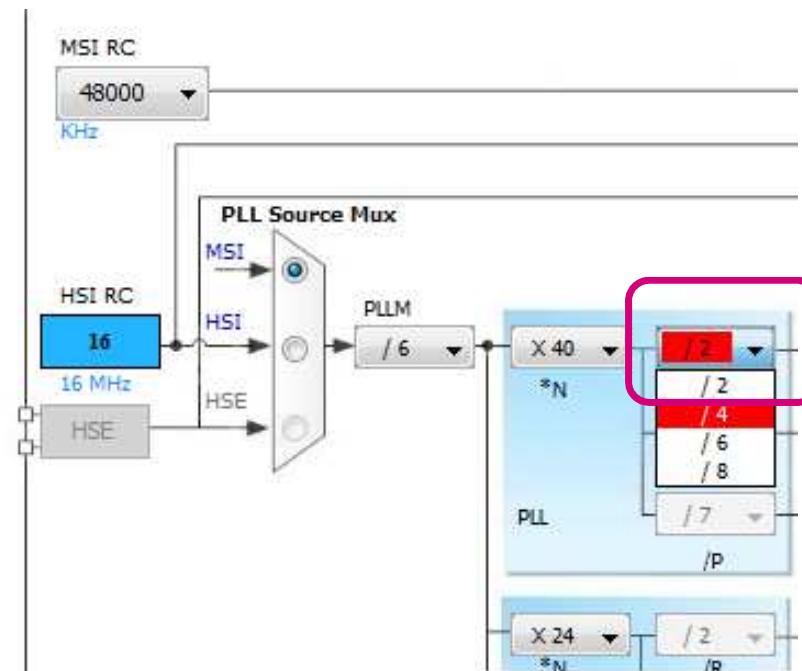
90

- Select **/6** in the **PLL M** dropdown menu



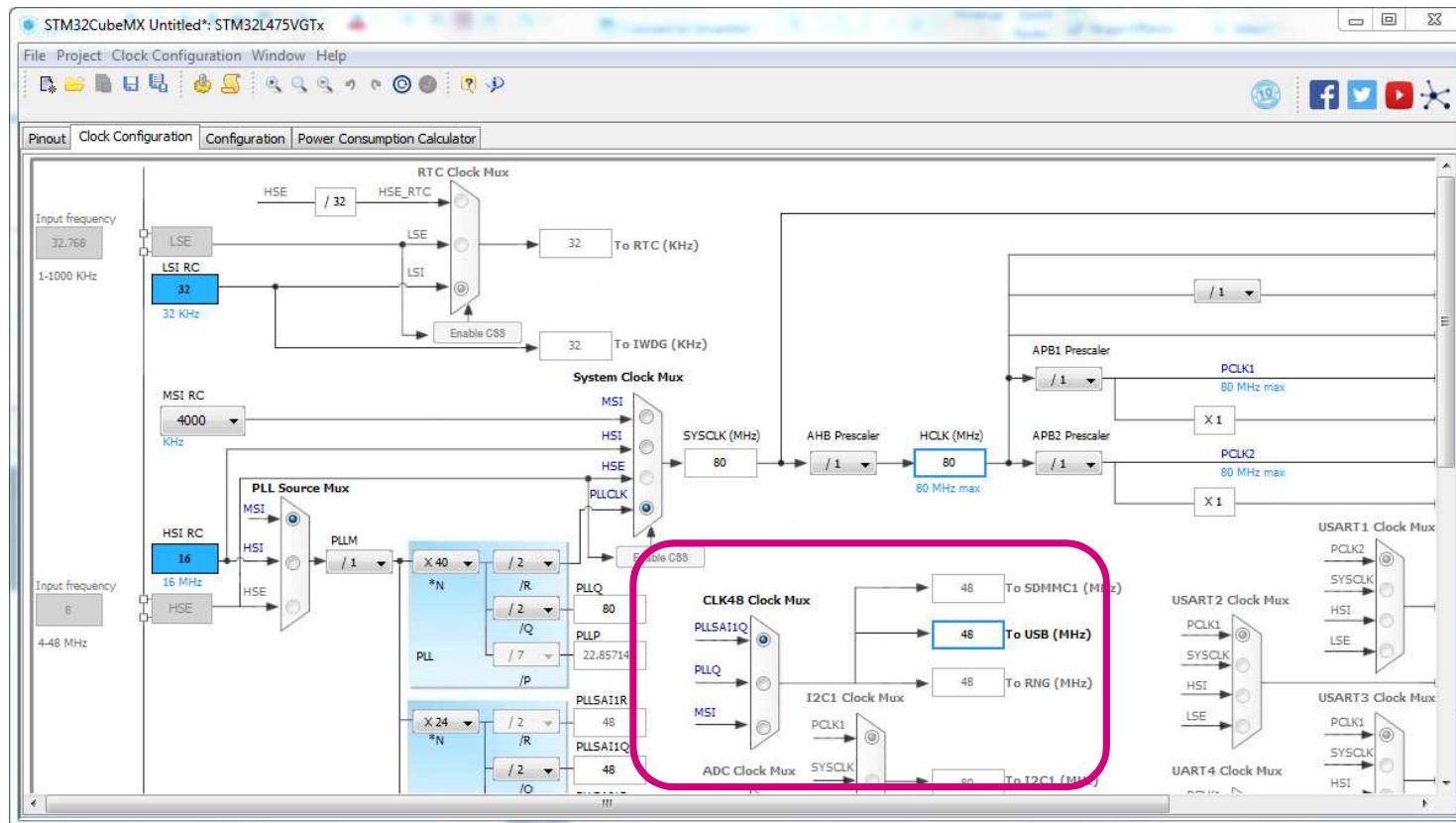
Clock Configuration

- Select **/4** in the **/R** dropdown menu



Clock Configuration

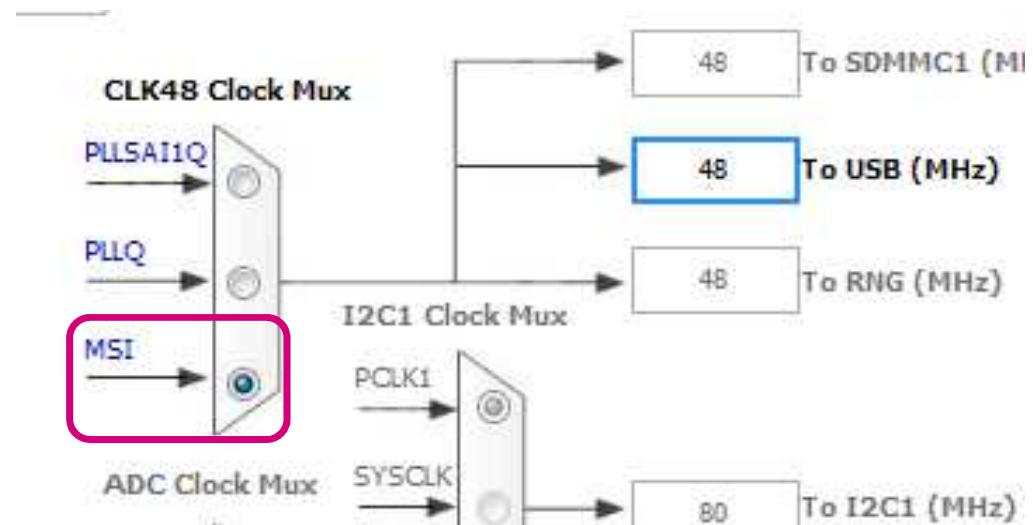
92



Clock Configuration

93

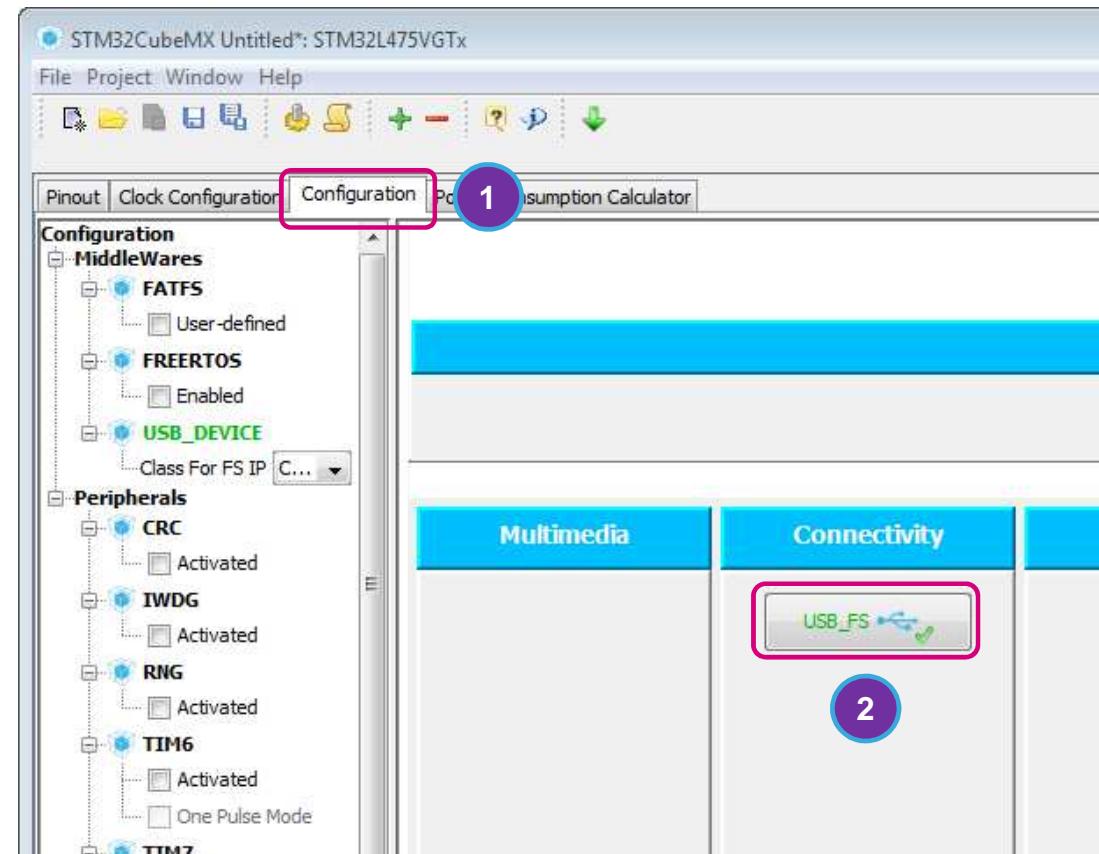
- Select **MSI** as the input for the **CLK48 Clock Mux**



USB Configuration

94

1. Click on the **Configuration** tab
2. Then click on **USB_FS** button

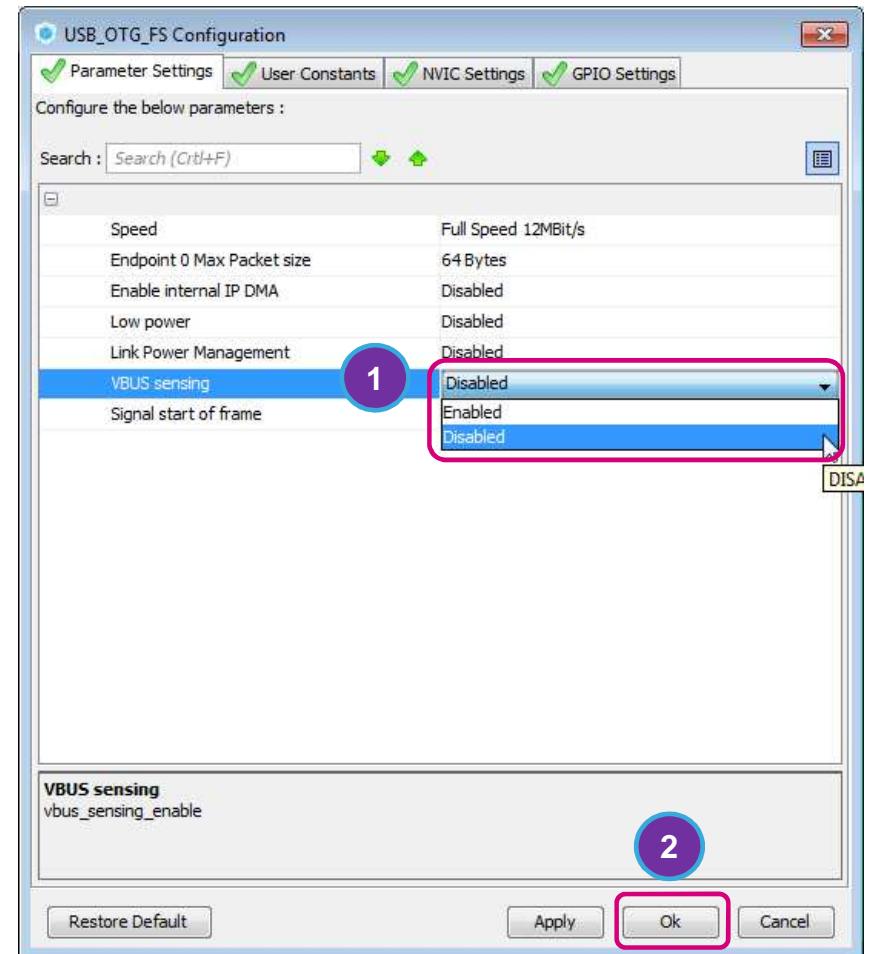


USB Configuration

95

1. Set the VBUS sensing to **Disabled**

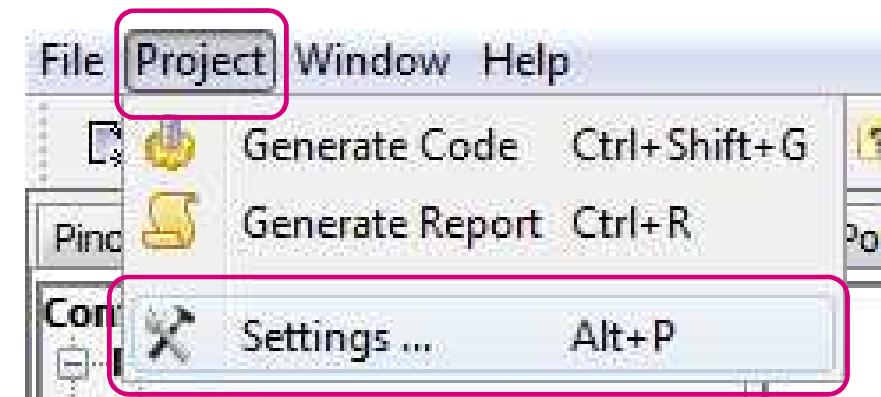
2. Click OK.



Project Settings

96

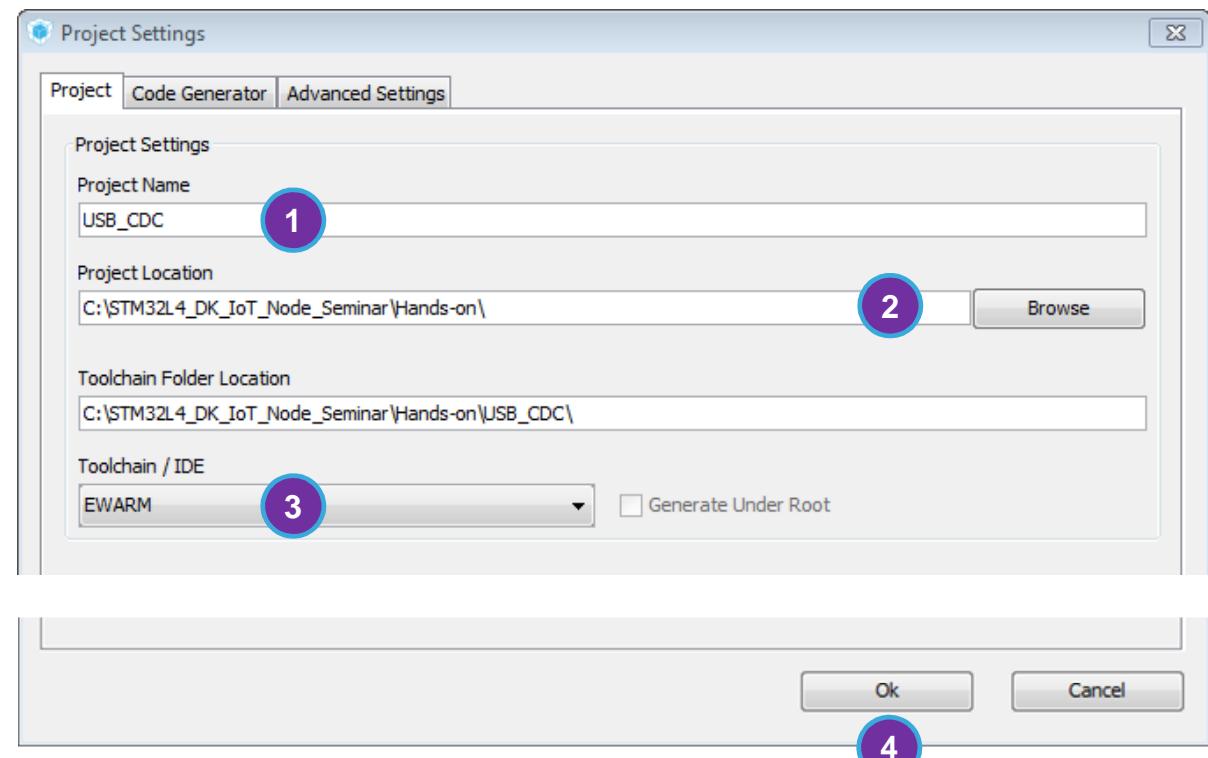
1. Select **Project -> Settings**
(Alt + P)



Project Settings

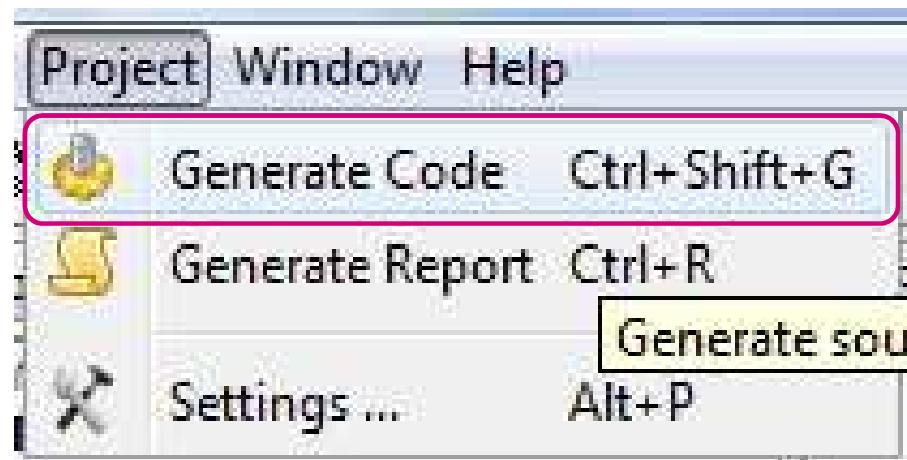
97

1. Set the project name to **USB_CDC**
2. Set the project location:
C:\STM32L4_DK_IoT_Node_Seminar\Hands-on
3. Set the IDE Toolchain to **EWARM**.
4. Click **OK**



Generate the Project

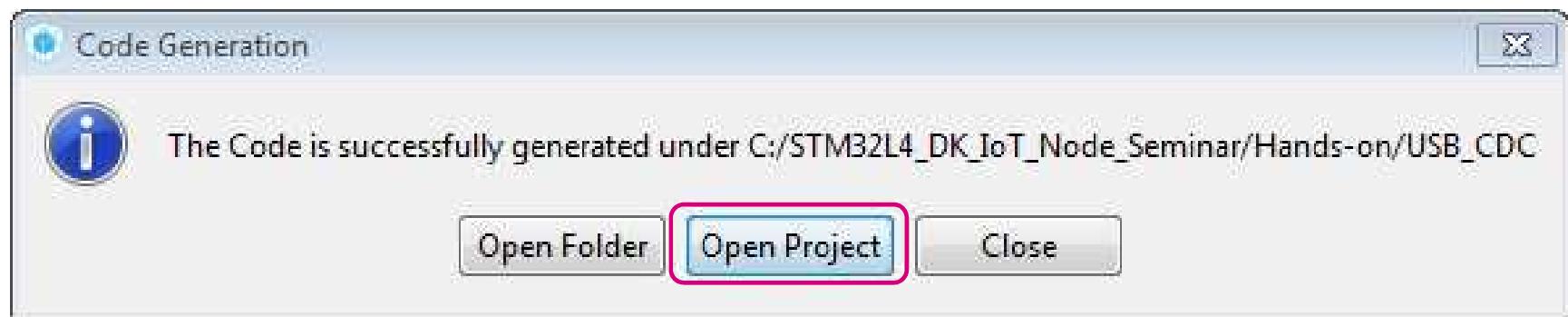
98



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

Open the Project

99

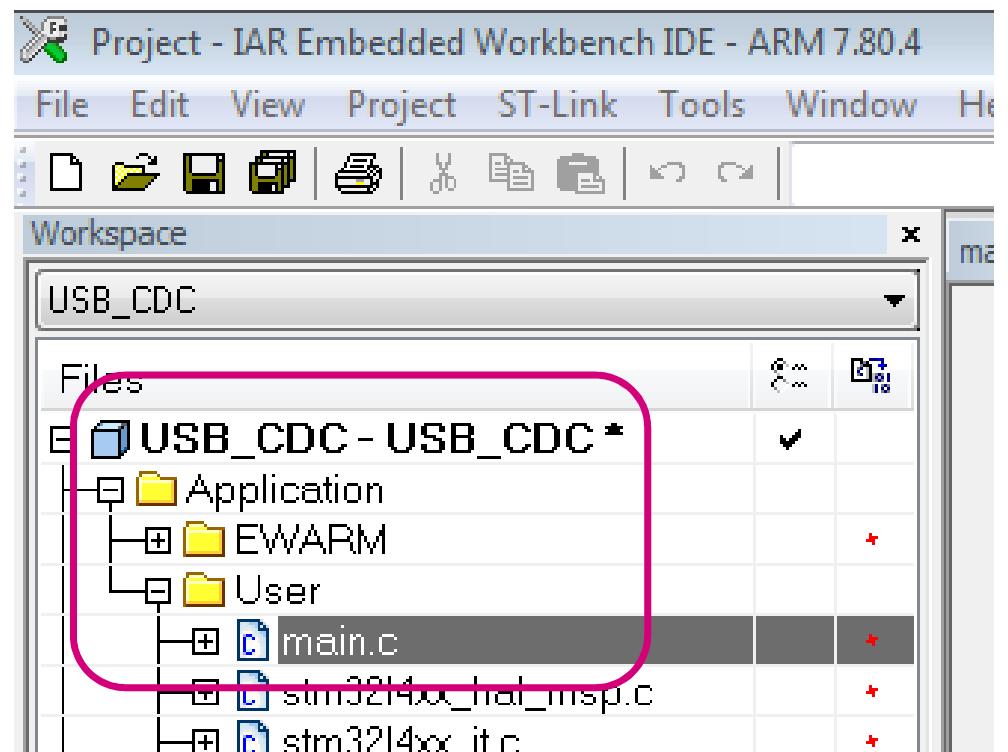


- Click **Open Project**

Edit main.c

100

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



Edit main.c

101

1. Add the USB device CDC header file on line 54:

```
#include "usbd_cdc_if.h"
```

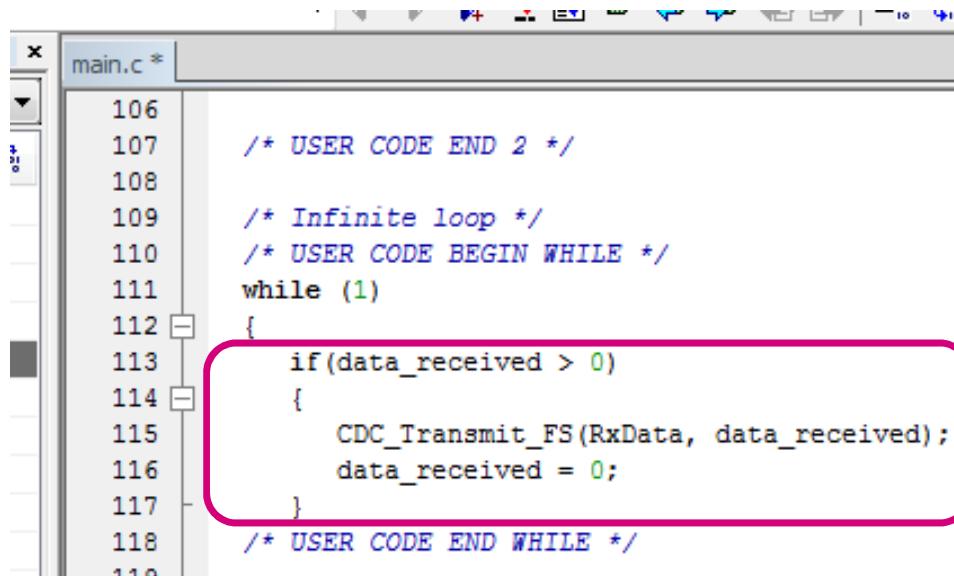
2. Add the following variables on lines 61 & 62:

```
uint8_t RxData[256];  
uint32_t data_received = 0;
```

```
main.c *  
*****  
47 */  
48 /* Includes -----  
49 #include "main.h"  
50 #include "stm32l4xx_hal.h"  
51 #include "usb_device.h"  
52  
53 /* USER CODE BEGIN Includes */  
54 #include "usbd_cdc_if.h" 1  
55 /* USER CODE END Includes */  
56  
57 /* Private variables -----  
58  
59 /* USER CODE BEGIN PV */  
60 /* Private variables -----  
61 uint8_t RxData[256]; 2  
62 uint32_t data_received = 0;  
63 /* USER CODE END PV */  
64  
65 /* Private function prototypes -----  
66 void SystemClock_Config(void);  
67 static void MX_GPIO_Init(void);  
68  
69 /* USER CODE DEACTIVATED */
```

Edit main.c

102



```
main.c *
106
107     /* USER CODE END 2 */
108
109     /* Infinite loop */
110     /* USER CODE BEGIN WHILE */
111     while (1)
112     {
113         if(data_received > 0)
114         {
115             CDC_Transmit_FS(RxData, data_received);
116             data_received = 0;
117         }
118     /* USER CODE END WHILE */
119 }
```

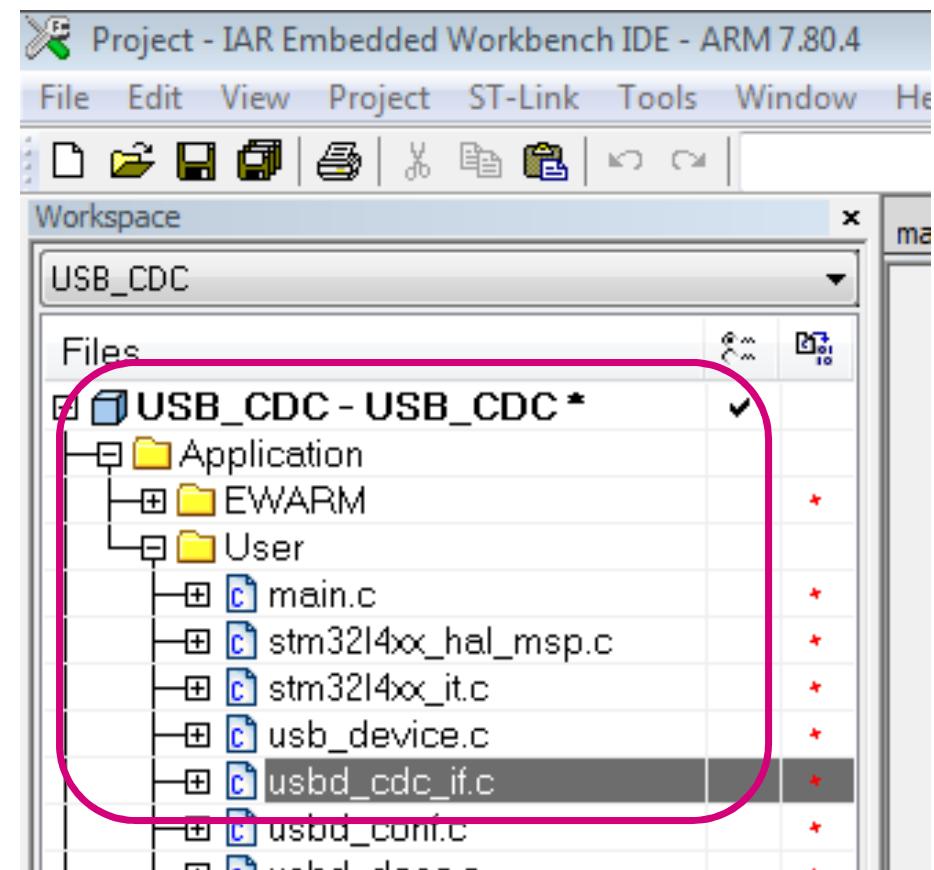
- Add the following code inside the while(1) loop, lines 113-117.

```
if(data_received > 0)
{
    CDC_Transmit_FS(RxData, data_received);
    data_received = 0;
}
```

Edit usbd_cdc_if.c

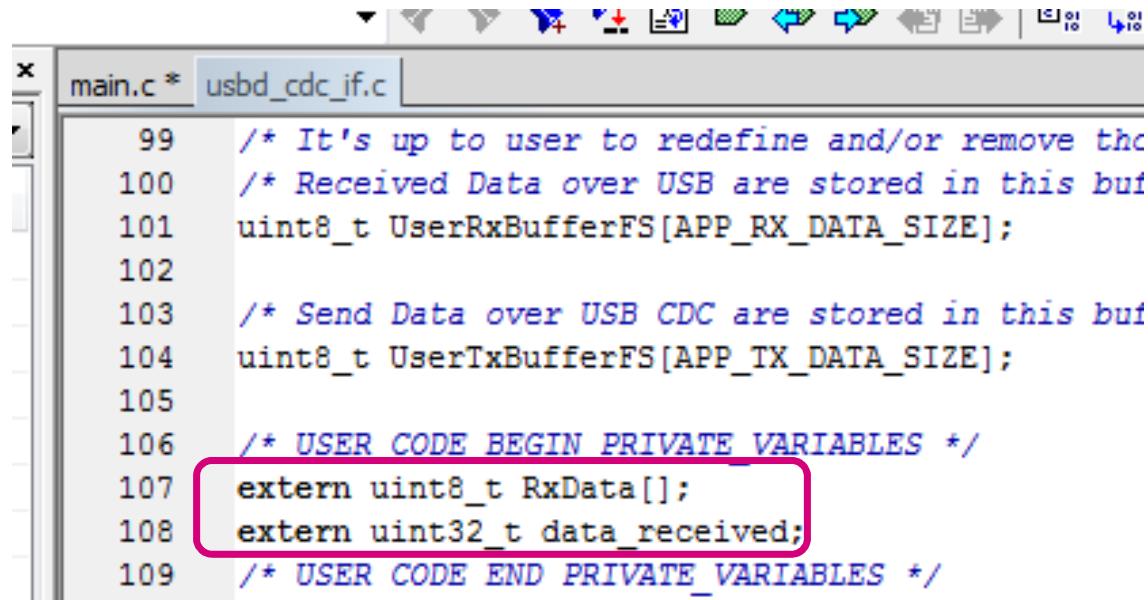
103

- Expand the file tree:
 1. Expand **USB_CDC**
 2. Expand **Application**
 3. Expand **User**
 4. Double-click **usbd_cdc_if.c**



Edit usbd_cdc_if.c

104



```
main.c * usbd_cdc_if.c
99  /* It's up to user to redefine and/or remove the
100 /* Received Data over USB are stored in this bui
101 uint8_t UserRxBufferFS[APP_RX_DATA_SIZE];
102
103 /* Send Data over USB CDC are stored in this bui
104 uint8_t UserTxBufferFS[APP_TX_DATA_SIZE];
105
106 /* USER CODE BEGIN PRIVATE VARIABLES */
107 extern uint8_t RxData[];
108 extern uint32_t data_received;
109 /* USER CODE END PRIVATE VARIABLES */
```

- Add the following variables to lines **107** and **108**:

```
extern uint8_t RxData[ ];
extern uint32_t data_received;
```

Edit usbd_cdc_if.c

105

```
268     static int8_t CDC_Receive_FS (uint8_t* Buf, uint32_t *Len)
269     {
270         /* USER CODE BEGIN 6 */
271         USBD_CDC_SetRxBuffer(&hUsbDeviceFS, &Buf[0]);
272         USBD_CDC_ReceivePacket(&hUsbDeviceFS);
273         data_received = *Len;
274         memcpy(RxData, Buf, data_received);
275
276         return (USBD_OK);
277         /* USER CODE END 6 */
```

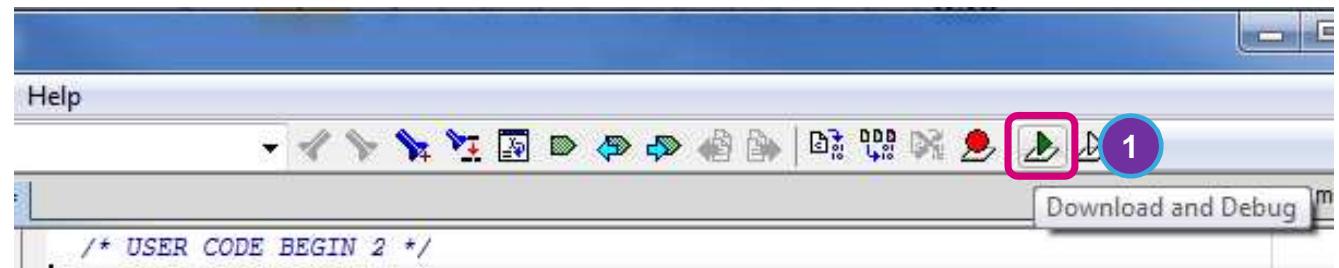
- Add the following code to lines **273** & **274**:

```
data_received = *Len;  
memcpy(RxData, Buf, data_received);
```

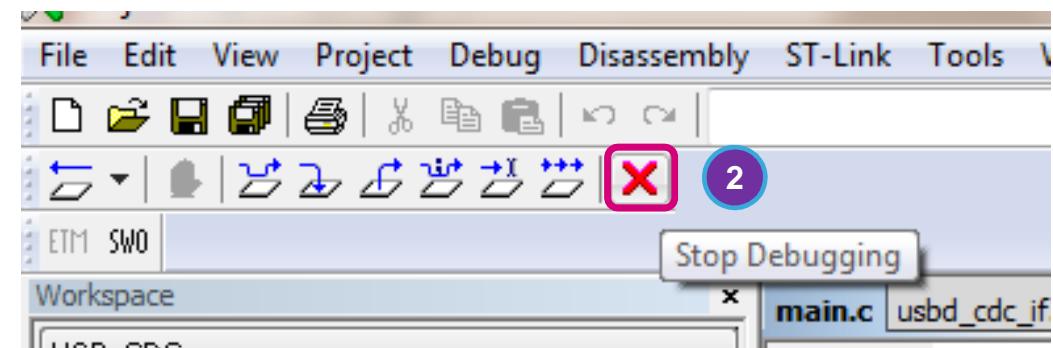
Build and Load

106

1. Click the GREEN ARROW to Build the Project, **Download** and start the **Debugger**. (Ctrl + D)



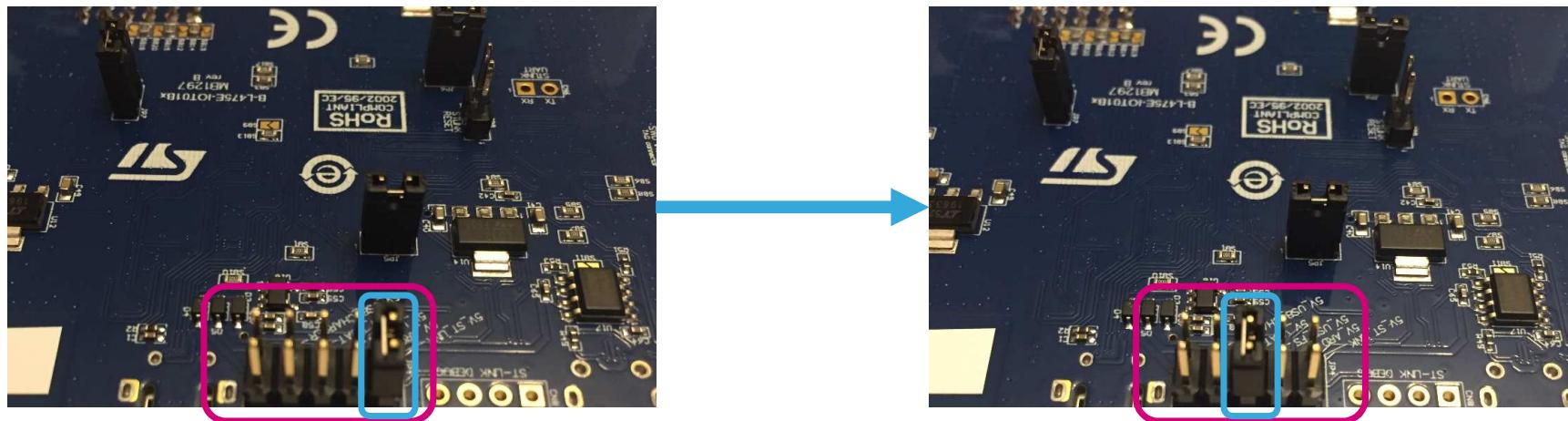
2. Click the X button (Ctrl + Shift + D) to **Stop Debugging**.



Move the DK IoT Board Power Jumper

107

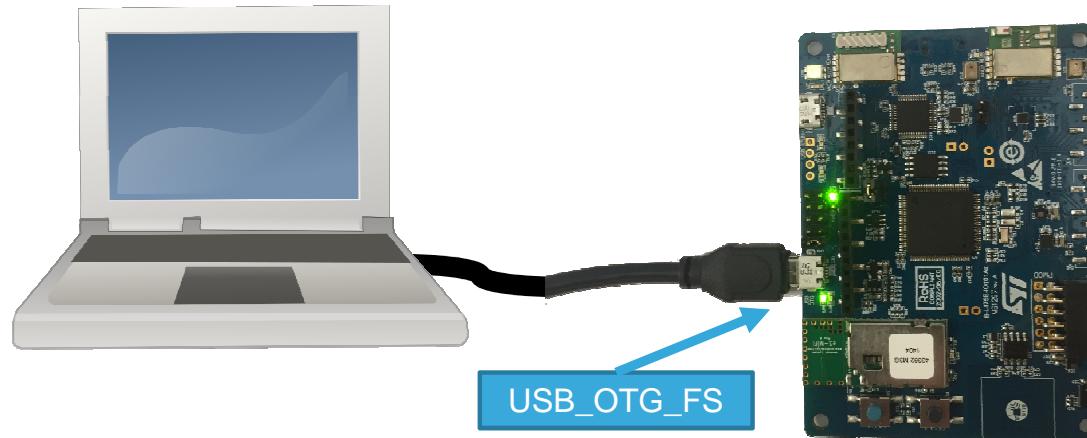
1. Disconnect the USB cable from the on-board ST-Link.
2. Flip your DK IoT board over and move the jumper to the middle position.



Connect the board to the PC

108

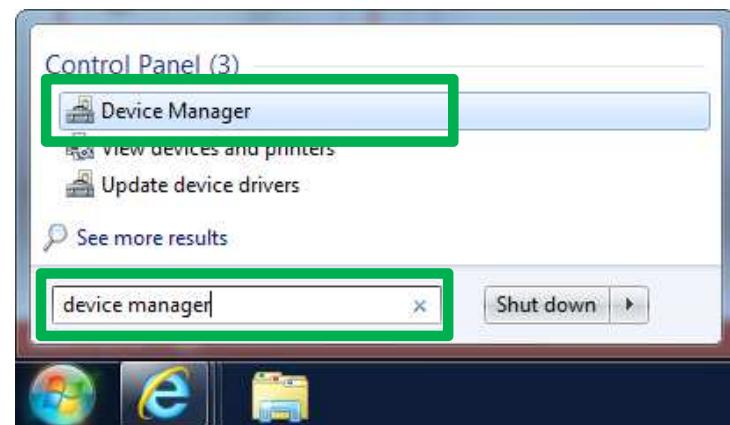
1. Connect the USB cable between the STM32L475 USB_OTG_FS and the PC.
2. Windows should detect the device and automatically install the driver.
3. If the device is NOT detected, you will need to install the driver manually.



Manual Driver Installation

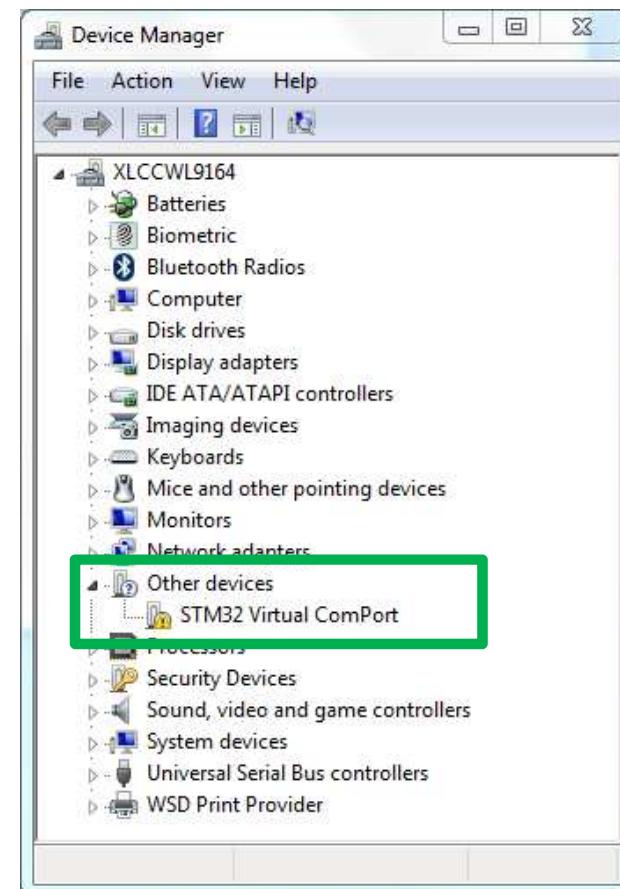
109

- Open the **START Menu** and type ‘device manager’
- Select “**Device Manager**” in the search results that appear



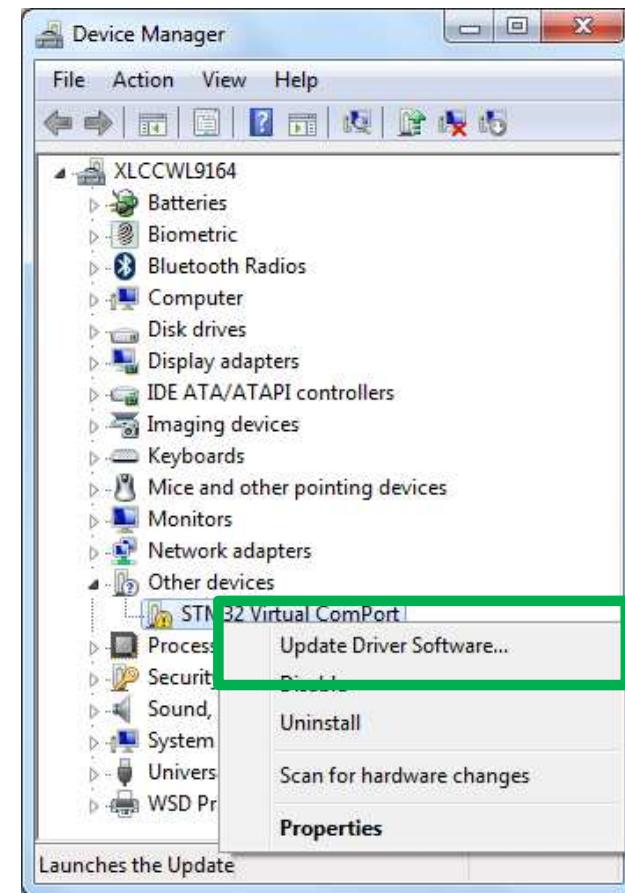
Manual Driver Installation

- Right-click on the **STM32 Virtual ComPort** device under **Other devices**

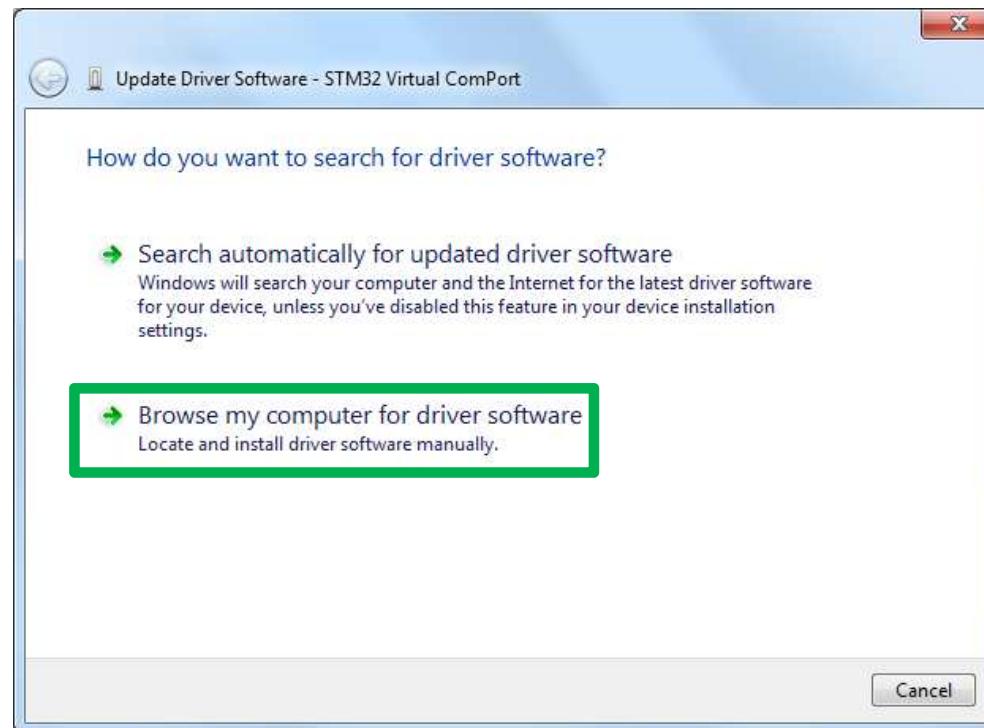


Manual Driver Installation

- Click on **Update Driver Software**

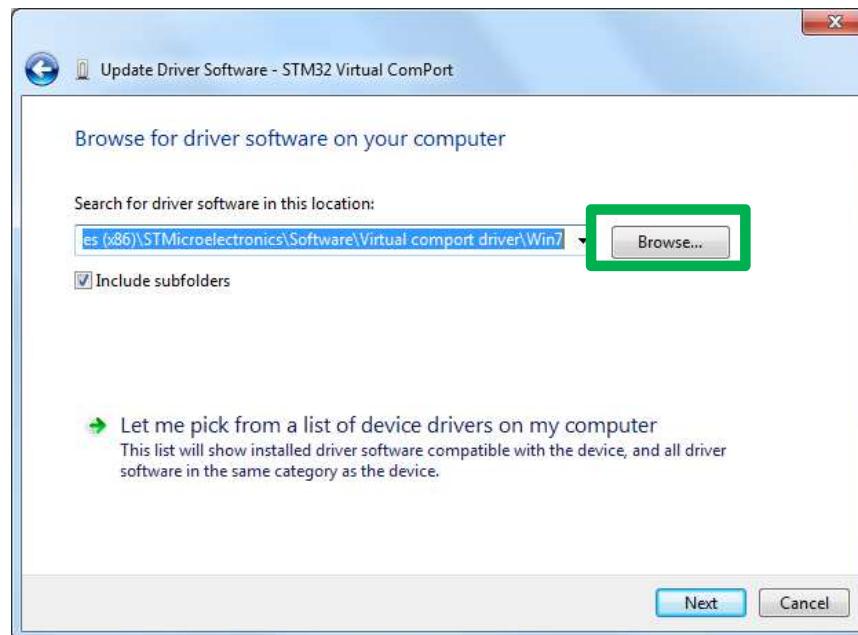


Manual Driver Installation



- Click on **Browse my computer for driver software**

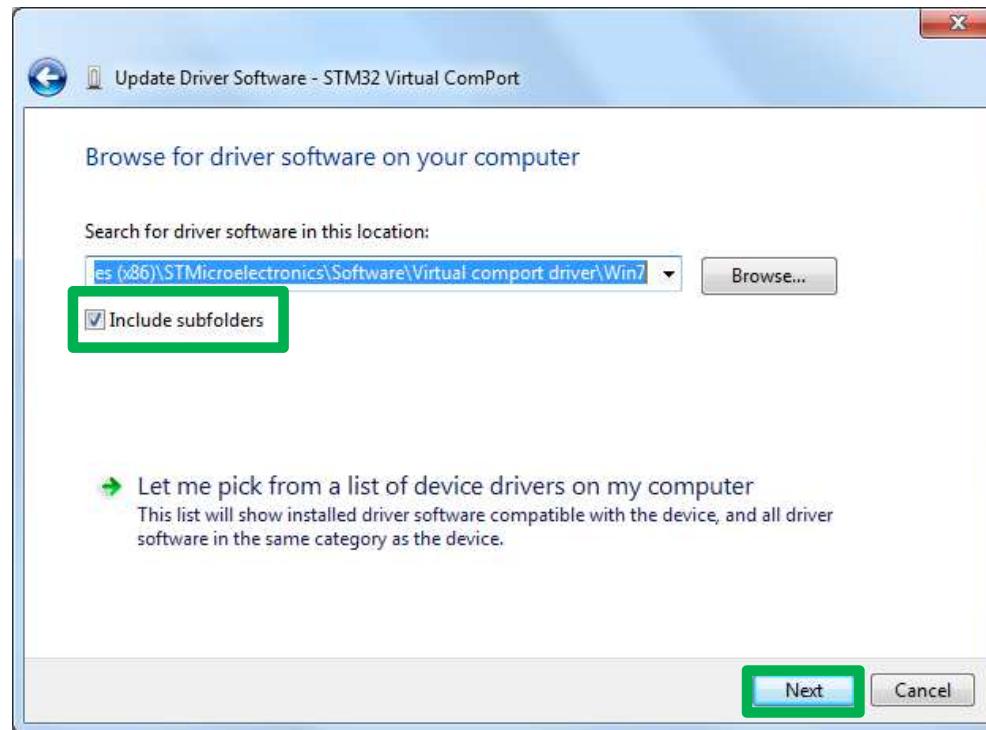
Manual Driver Installation



- Click **Browse**, then navigate to:

C:\Program Files (x86)\STMicroelectronics\Software\Virtual
comport driver

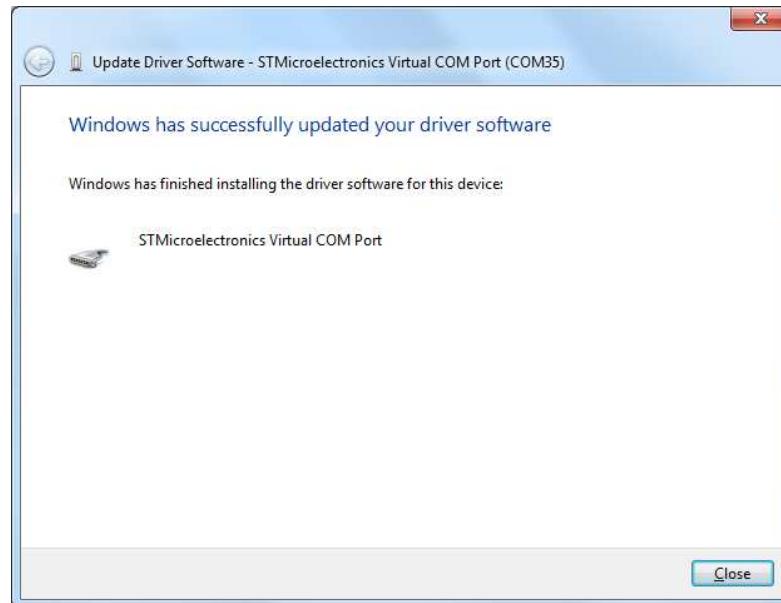
Manual Driver Installation



- Select **Include subfolders**, then click **Next**

Manual Driver Installation

115



- Windows will install the driver
- Click 'Close'

Open TeraTerm

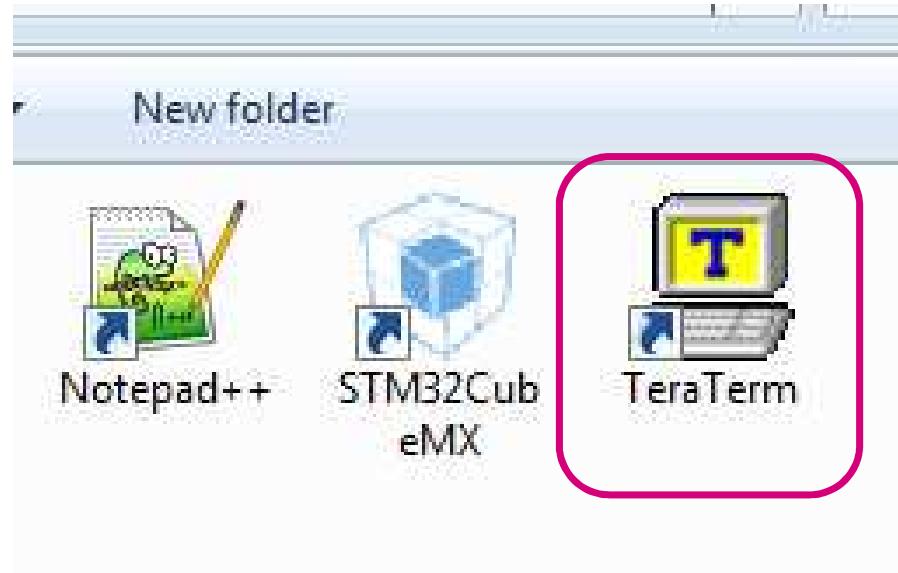
116



- Open the **STM32L4 DK IoT Node Seminar** folder on your Desktop.

Open TeraTerm

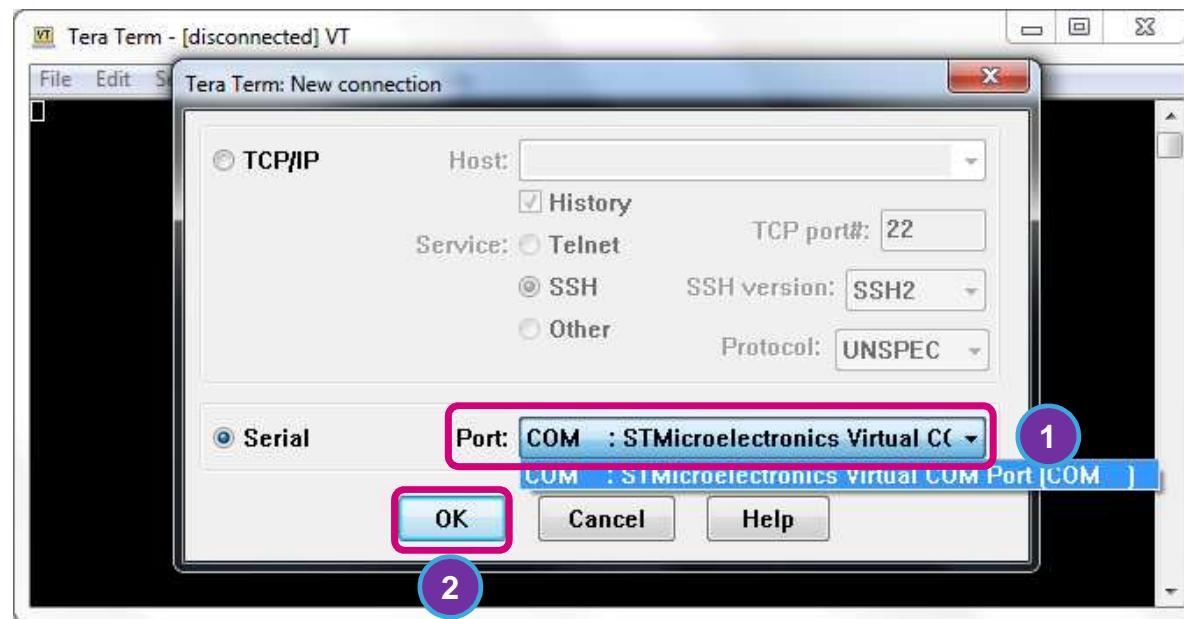
117



- Double-click on the **TeraTerm** shortcut

Configure TeraTerm

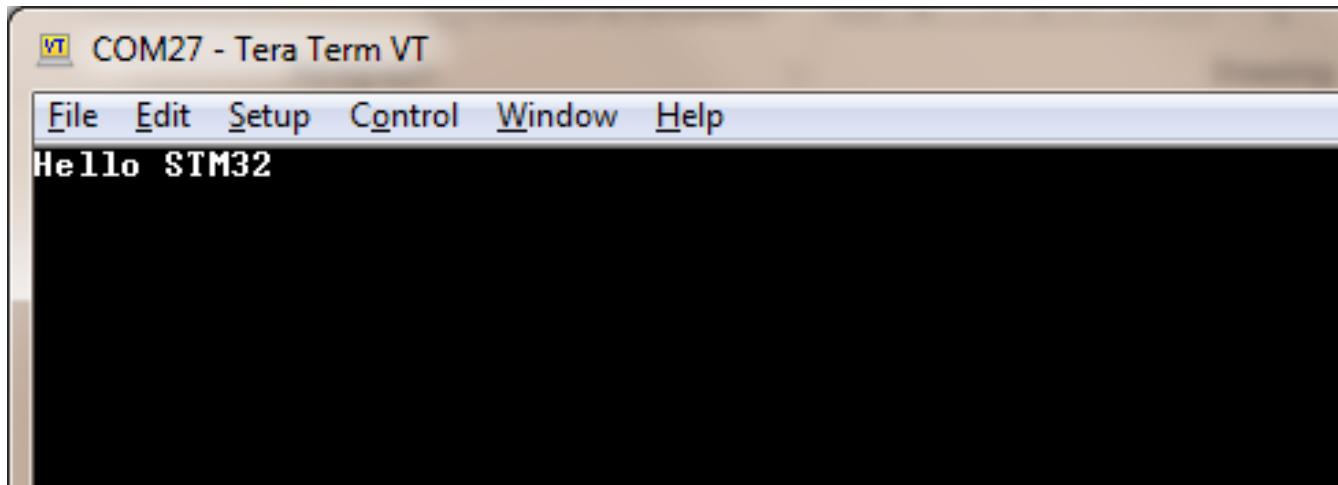
118



1. Select the **STMicroelectronics Virtual COM Port**
2. Click **OK**

TeraTerm Echo Output

119



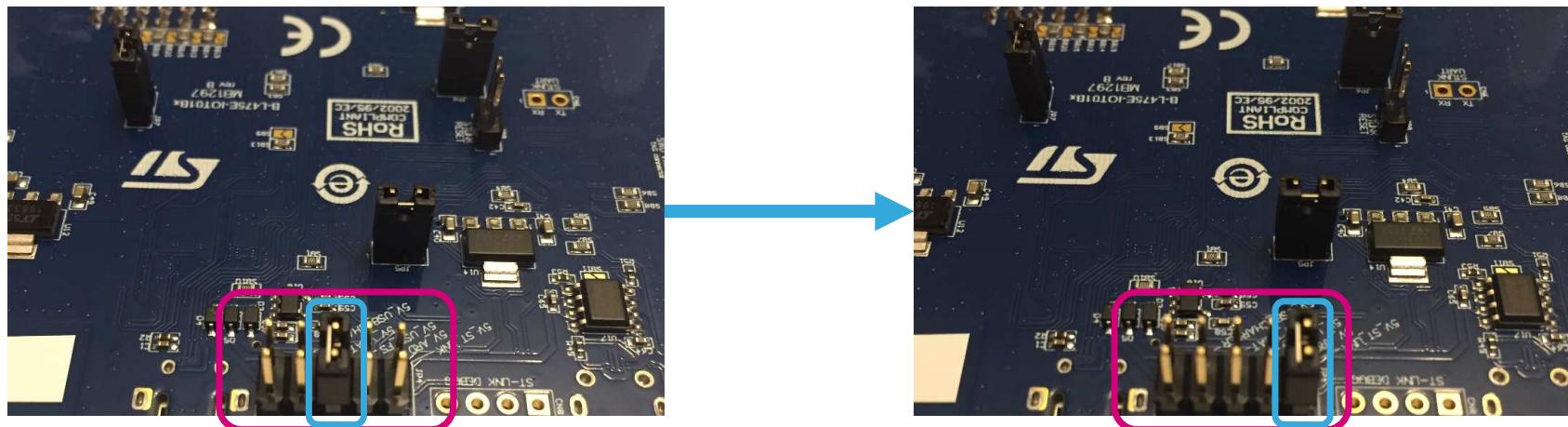
- Type “Hello STM32”
- The Tera Term window will display the characters that are echoed back

How this example can be used?

- You can use the example to:
 - Configure your application
 - Debug
 - Firmware Update
 - Bluetooth Low Energy / SubGHz to USB bridge

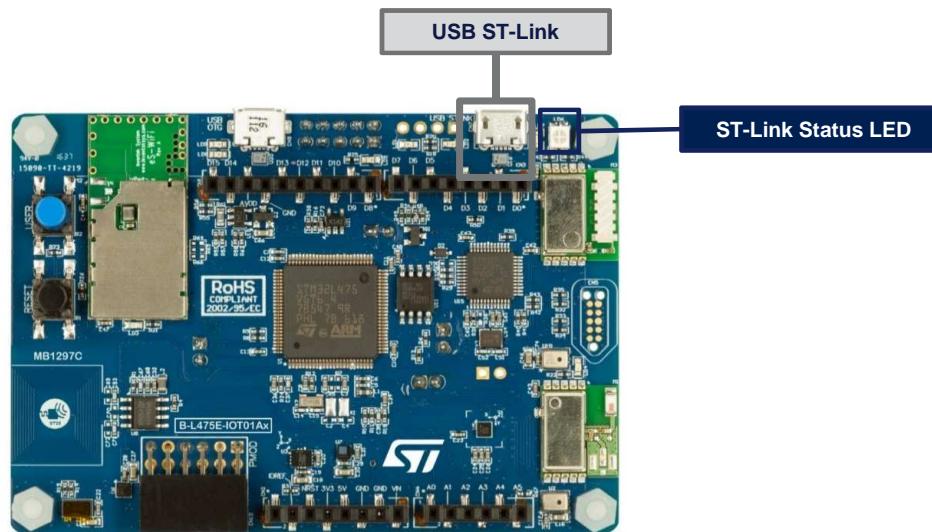
Move the DK IoT Board Power Jumper

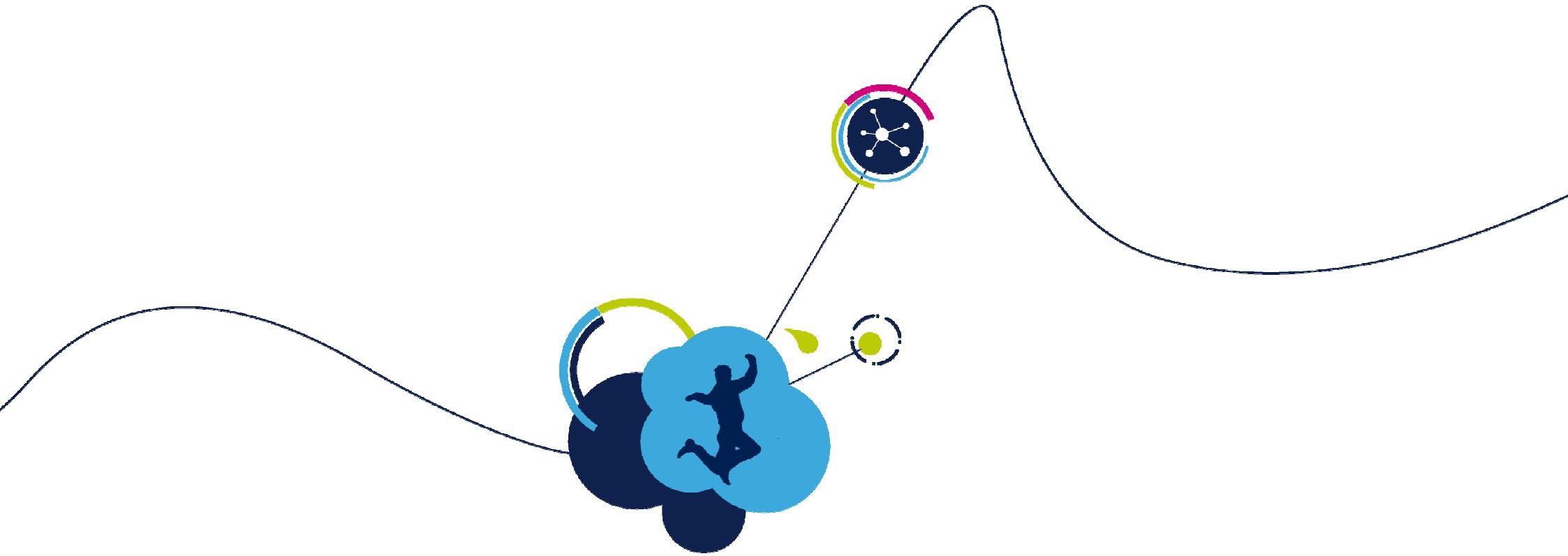
1. Disconnect the USB cable between the STM32L475 USB_OTG_FS and the PC.
2. Flip your DK IoT board over and move the jumper to the 5V_ST_LINK position.



Connect the board to the PC

- 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.



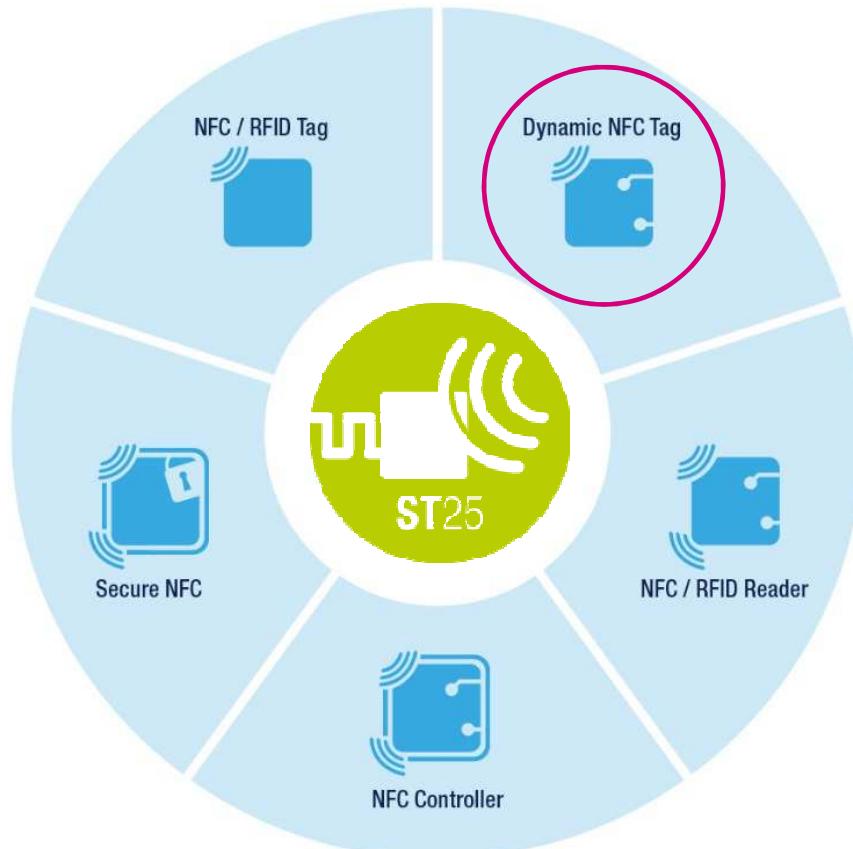


NFC Overview



ST NFC Positioning

124



Complete Portfolio

→ From Tags to High Performance Readers

Market Coverage

→ Covering all NFC application needs and leveraging a rich ecosystem

Acquisition

ST acquired ams AG
NFC / RFID Readers Assets

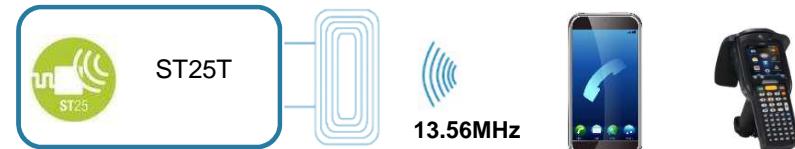


ST25 Product Lines

125

Ticketing, Gaming, Medical, Brand protection, Access control, ...

NFC Tags



Industrial, Consumer, Metering, Appliance, IoT...

NFC Dynamic Tags



POS & mPOS Terminals

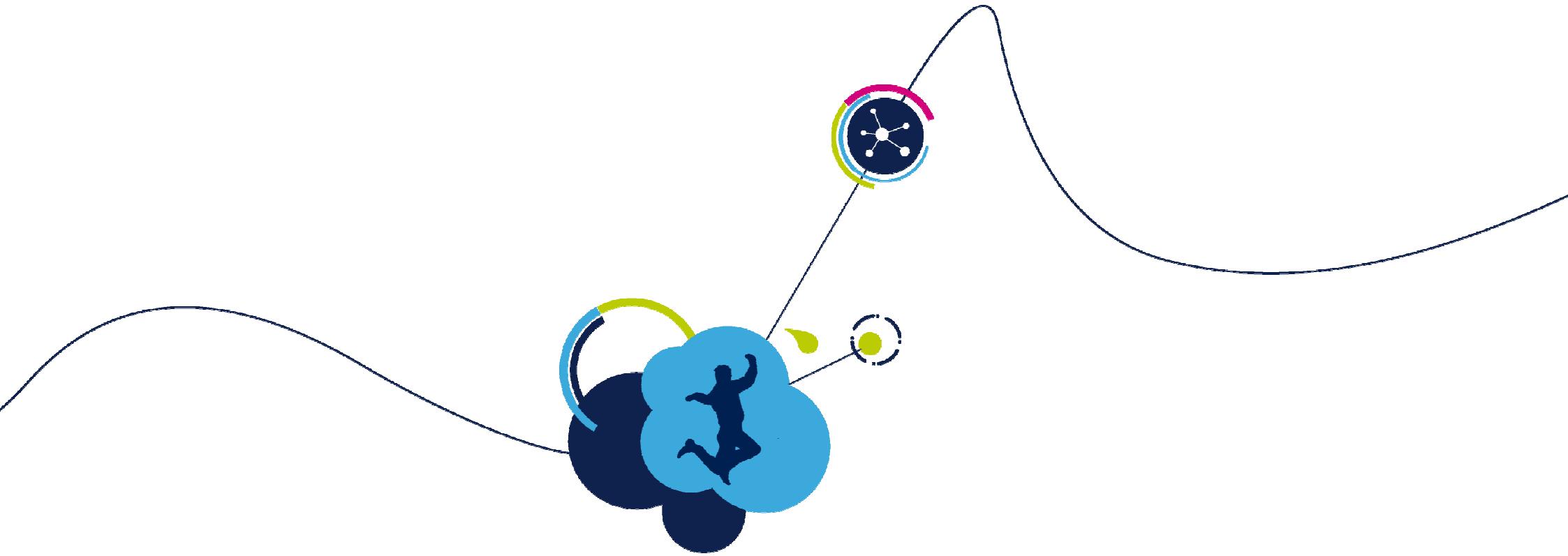
EMVCO Readers



Automotive, Accessory ID, Gaming, Medical, Brand protection, Access control, ...

Readers





Sensors Overview

Motion MEMS : 6-axis IMU and Magnetic Sensors



Key advantages:

- Accuracy
- Power consumption
- Smart Functions

Main features

- Stability and ultra low noise to address demanding applications
- Best in class Low power consumption for extend battery life. $I_{COMBO} = 0.65\text{mA}$ in H. Perf. mode
- Advanced digital features for Sensor HUB, pedometer and significant motion detection functionalities.
- 4kB Embedded FIFO for smart data storage and power saving.
- LSM6DSM: Dedicated auxiliary SPI for OIS applications and Android M compliant



Key advantages:

- High Resolution
- Low Power

Main features

- Ultra-compact high-performance 3D magnetometer
- Wide magnetic sensor dynamic range
- Low magnetic offset
- Embedded Self Test and Temperature compensated

Environmental MEMS : Pressure & Humidity Sensor

128



Pressure
LPS22HB

2x2x0.76 mm

Key advantages:

- High sensitive
- Ultra Low Power
- Embedded temp comp
- Small and thinner

Main features

- Highest Accuracy (pressure): up to 8cm in vertical resolution
- Unprecedented ascent and descent speed estimation
- Ultra Low Power consumption 3uA LP – 15uA HR
- Patented Package Technology enabling Water and Dust resistance
- Embedded FIFO for Pressure and Temperature data



Humidity
HTS221

2x2x0.9 mm

Key advantages:

- Best tradeoff between performance and current consumption

Main features

- Humidity accuracy $\pm 3.5\%$ rH, 20% to 80%rH
- -40°C to 120°C operating temperature range
- Temperature accuracy $\pm 0.5^\circ\text{C}$, 15 to 40°C
- Flexible ODR from 1Hz to 12.5Hz
- Low power consumption 2uA @ 1Hz ODR
- Extended supply voltage (1.7 to 3.6 V)



Microphone
MP34DT01
MP34DT05

3x4x1 mm

Key advantages:

- High SNR
- High AoP
- Flat Frequency Response
- Sensitivity matching

Main features

- Omni-directional PDM output, industry standard footprint (3x4mm)
- High SNR (64dB) allowing improved audio fidelity experience
- High AoP (122.5dBSPL) among TOP port MEMS microphones
- Ground ring around the port hole to improve ESD robustness

Software

100110
0110110
000110

Acoustic Open.Audio Free Libraries, to:

- Estimate the arrival of audio signal
- Create a virtual directional microphone/Beam Forming
- Acoustic Echo Cancelation

Open.Audio

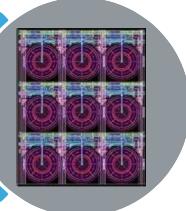
ST is WWide
#1 ToF
supplier

Over 3 years of proven track record in manufacturing
>100M products shipped
300% AAGR

Time of Flight Sensor : VL53L0X

130

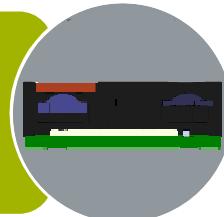
Single Photon Avalanche Diode
Ultra fast time resolution enabling Direct ToF processed in ST CMOS SPAD process



ST Proprietary Time-of-Flight IP
Best compromise of cost, complexity & power vs performance



Compact integrated system
Sensor, filters, optics, VCSEL and driver integrated
Fully calibrated system



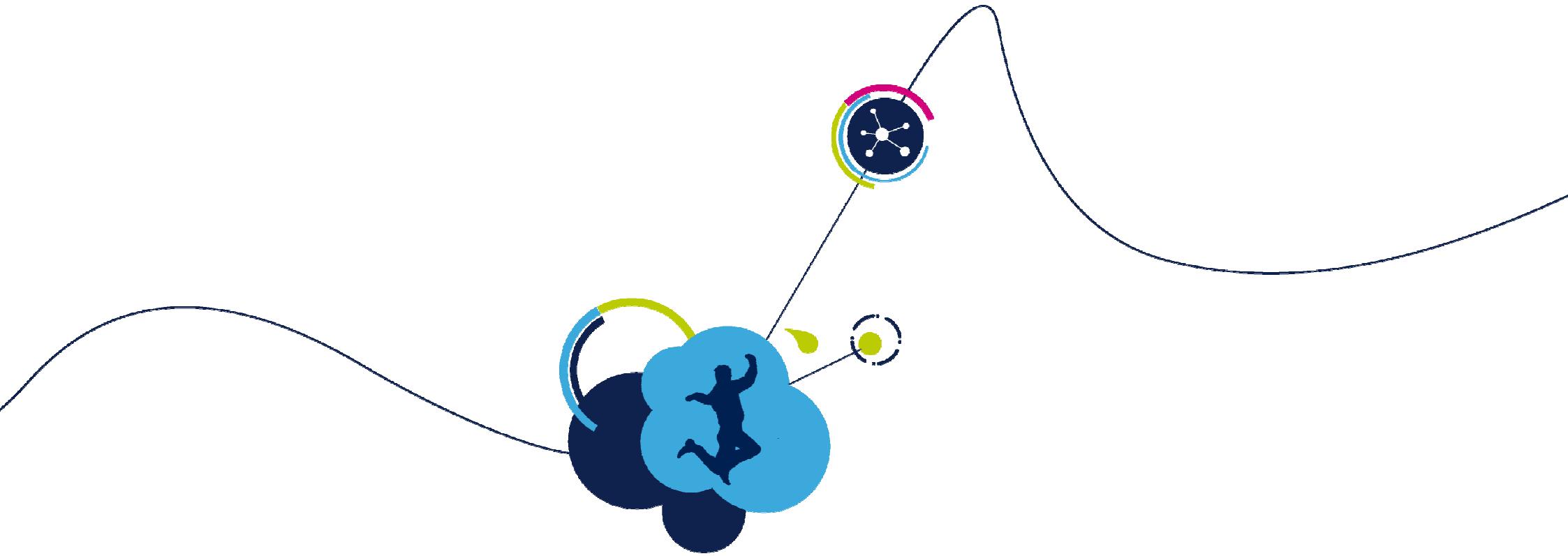
Optimised and reliable supply chain
High volume & low cost



OLGA 4.4 x 2.4 x 1mm

PRODUCT HIGHLIGHTS

- **Fully integrated** (IR 940nm Vcsel emission, filters, SPAD receiving array, advanced µC)
- Range up to **200cm in less than 30ms**
- **High accuracy (typ. 5%)**
- **Low power** (stdby 5µA, active <20mW at 10Hz)
- FoV : 25°
- Laser Class1 device (eye safe)
- Excellent ambient light robustness (**940nm**)
- Works with cover glass
- Complete API package and **Android driver**

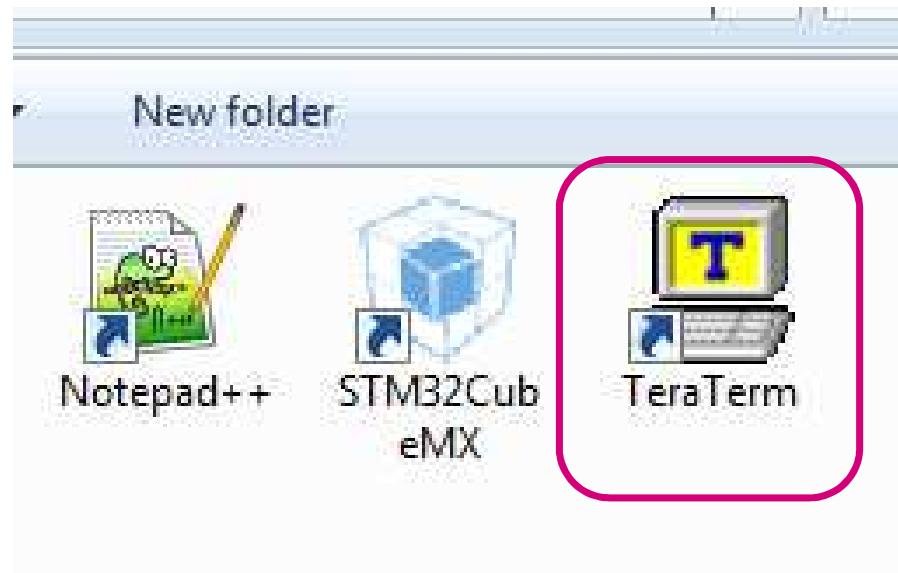


Lab 2 : Real Time Sensor Logging

- In this example we are going to demonstrate how to:
 - Read the on-board MEMS sensor data using the sensor library
 - Use IAR
 - Breakpoints
 - Local Watch to view local variables
 - Live Watch to view global variables
 - Use Tera Term to exchange data between STM32 and a PC over USB
- Close the previous STM32CubeMX project and Tera Term

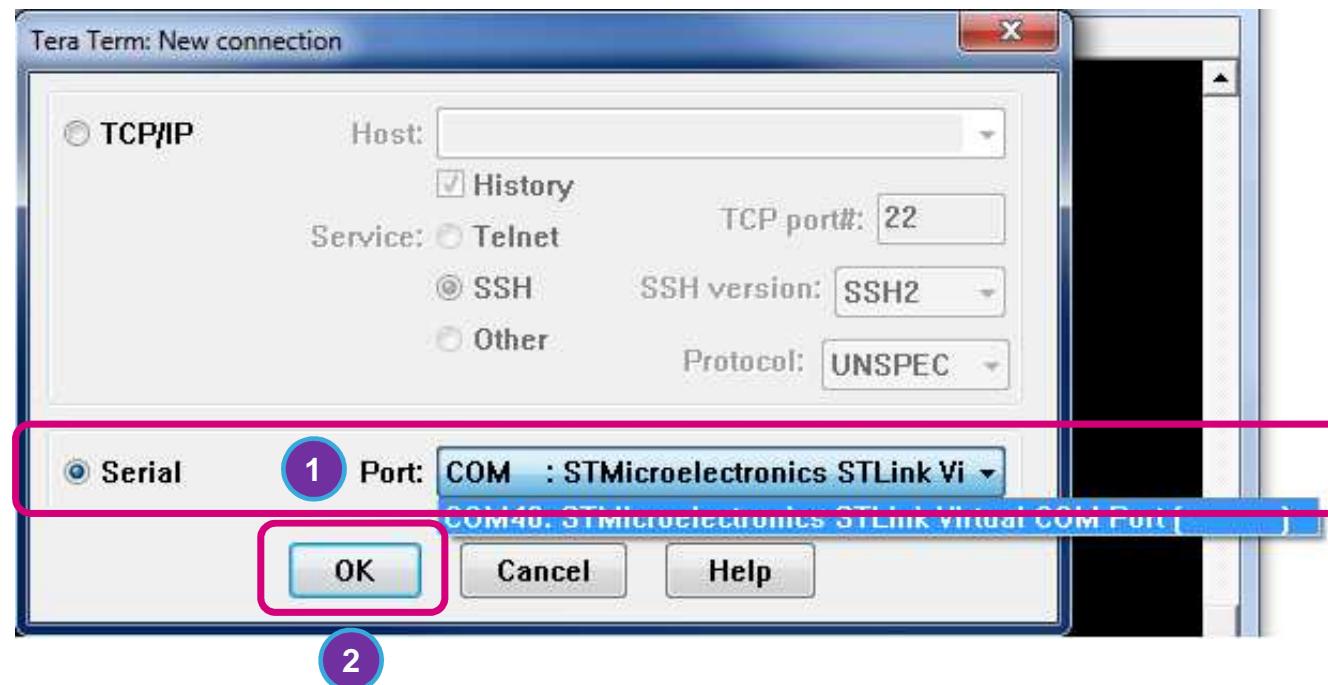
Open TeraTerm

133



- Double-click on the **TeraTerm** shortcut

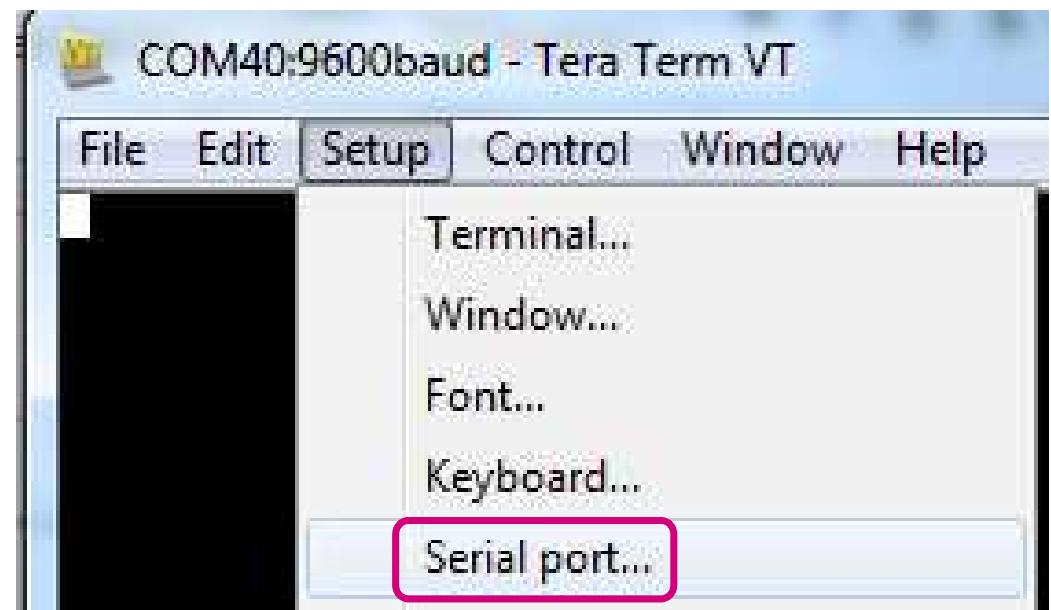
Tera Term Configuration 1/5



1. Select the **STMicroelectronics STLink Virtual COM Port**
2. Click **OK**

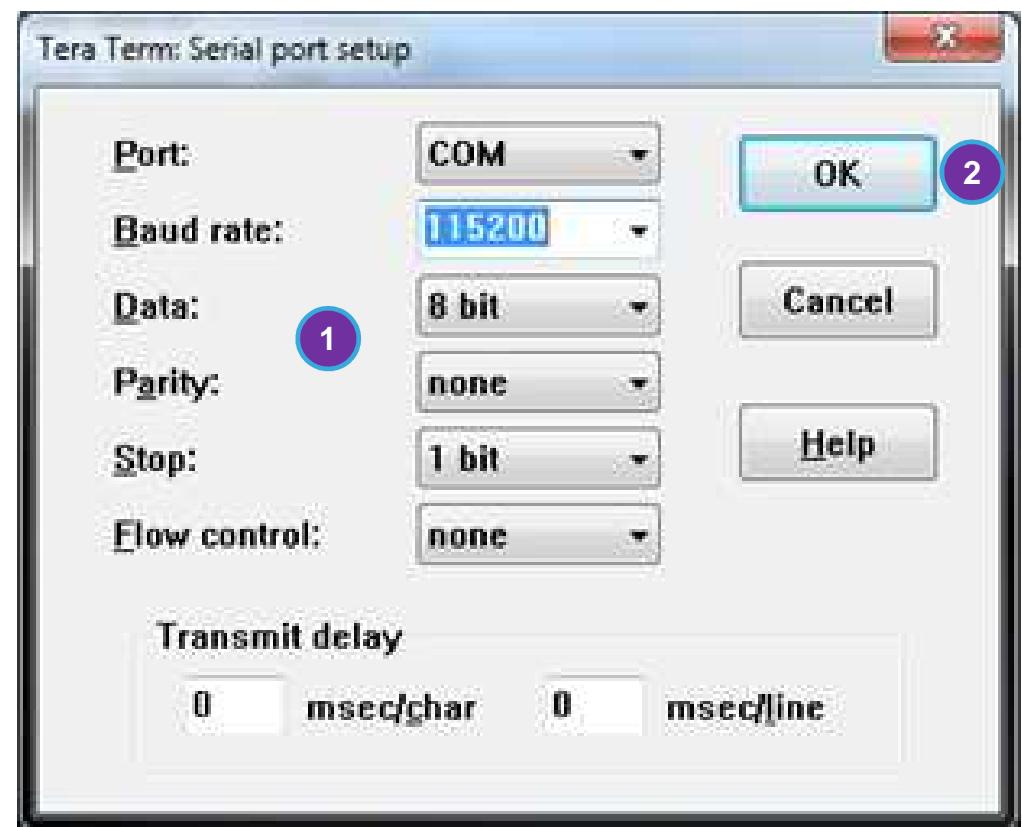
Tera Term Configuration 2/5

1. Click **Setup** -> **Serial port...**



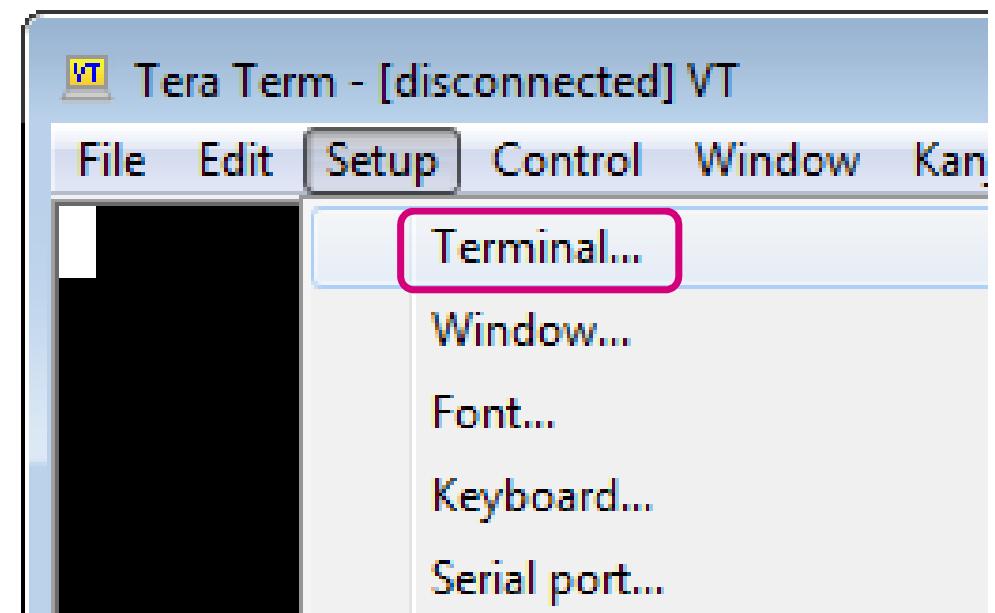
Tera Term Configuration 3/5

1. Set the following:
 - Baud rate : **115200**
 - Data : **8 bit**
 - Parity : **none**
 - Stop : **1 bit**
 - Flow control : **none**
2. Click **OK**



Tera Term Configuration 4/5

1. Click **Setup** -> **Terminal...**



Tera Term Configuration 5/5

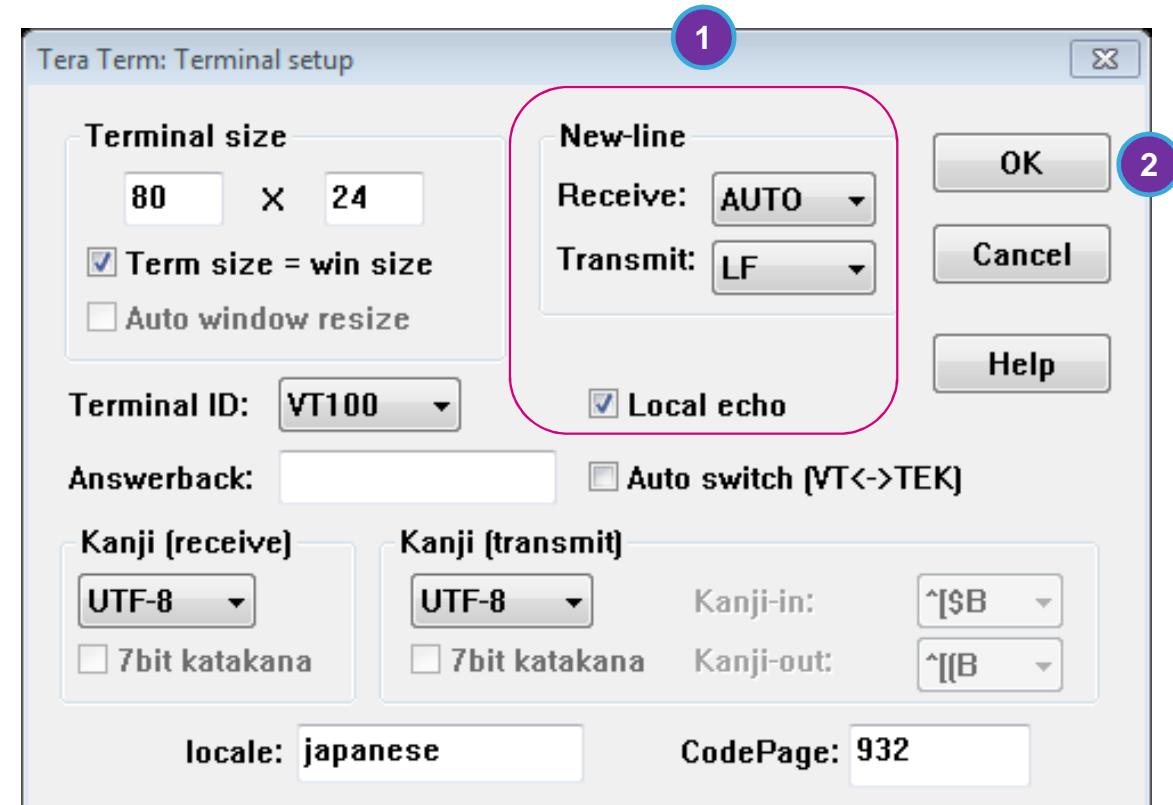
- Set the following:

New-Line Receive : **AUTO**

New-Line Transmit : **LF**

Enable **Local echo**

- Click **OK**



Open the MEMS Project

139

- We are going to configure the **MEMS** project to view data coming from the on-board accelerometer in both raw and normalized formats.
1. Close the previous IAR project.
 2. Double click on the **Project.eww** file located under:

C:\STM32L4_DK_IoT_Node_Seminar\Hands-on\Labs\Projects\BL475E-IOT01\Examples\BSP\EWARM

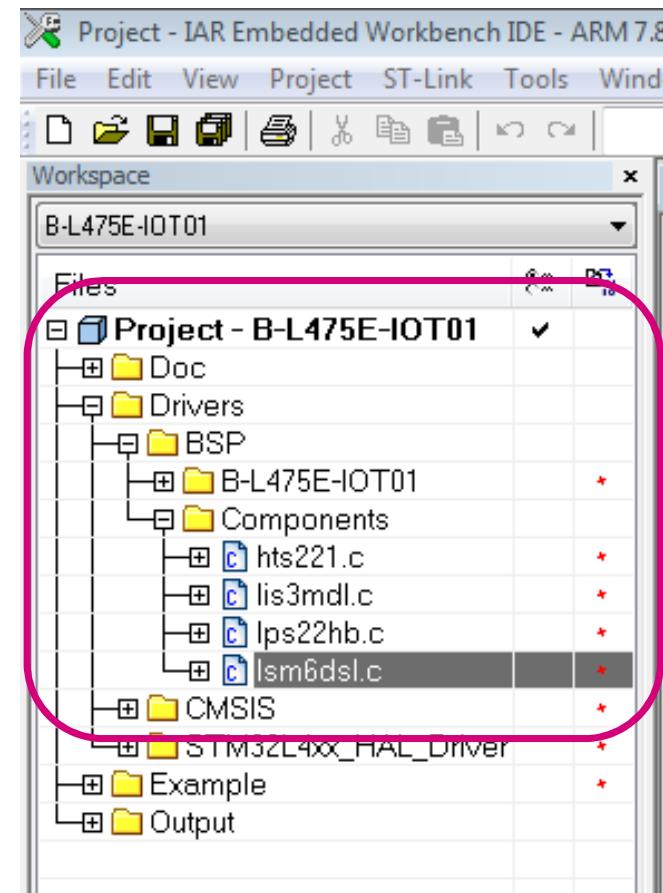


Setting a Breakpoint

140

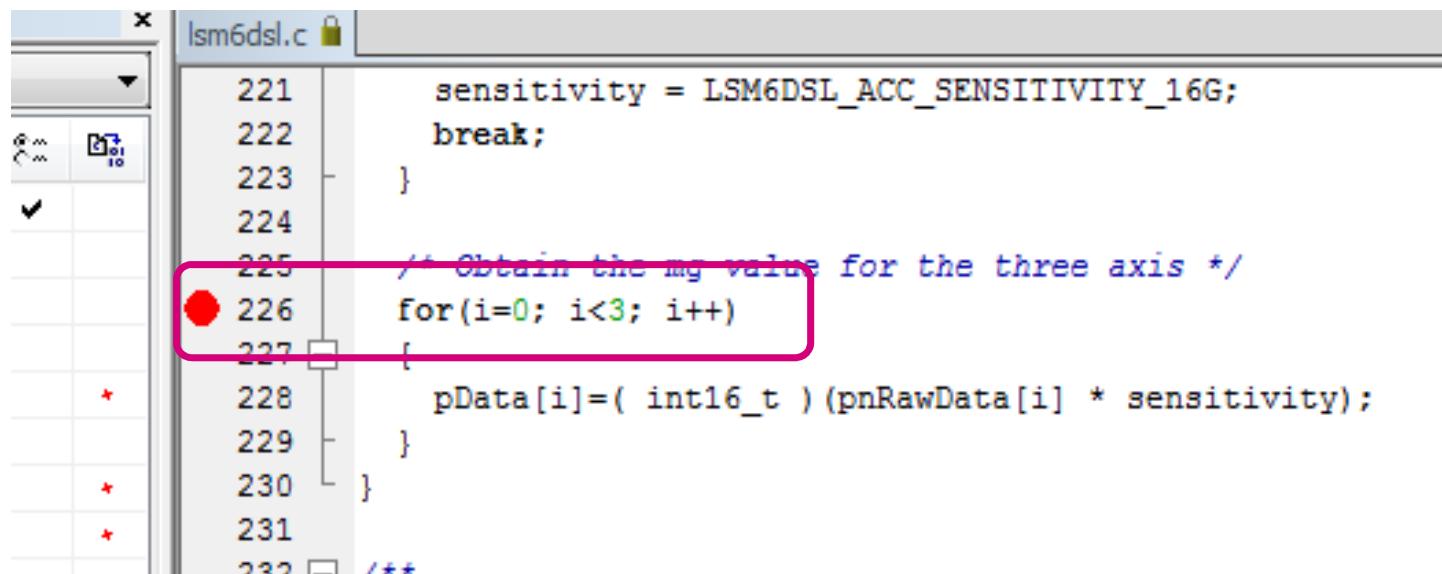
Expand the file tree:

1. Expand **Project – B-L475E-IOT1**
2. Expand **Drivers**
3. Expand **BSP**
4. Expand **Components**
5. Double-click **lsm6dsl.c**



Setting a Breakpoint

141



The screenshot shows a code editor window titled "lsm6dsl.c". The code is written in C and includes a for-loop to calculate sensitivity values for three axes. A red circle marks a breakpoint at line 226, which is highlighted with a red rectangle. The code is as follows:

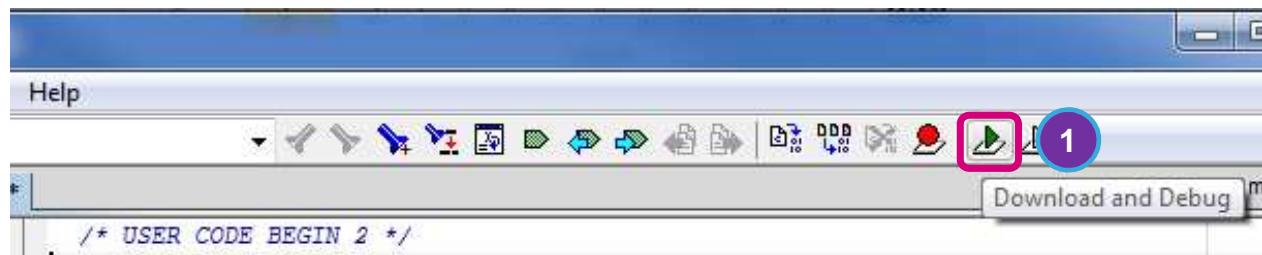
```
221     sensitivity = LSM6DSL_ACC_SENSITIVITY_16G;
222     break;
223 }
224
225 /* Obtain the mg value for the three axis */
226 for(i=0; i<3; i++)
227 {
228     pData[i]=( int16_t )(pnRawData[i] * sensitivity);
229 }
230
231
232 /**
```

- Set a break point at line **226** by left-clicking on the left side of the line number.

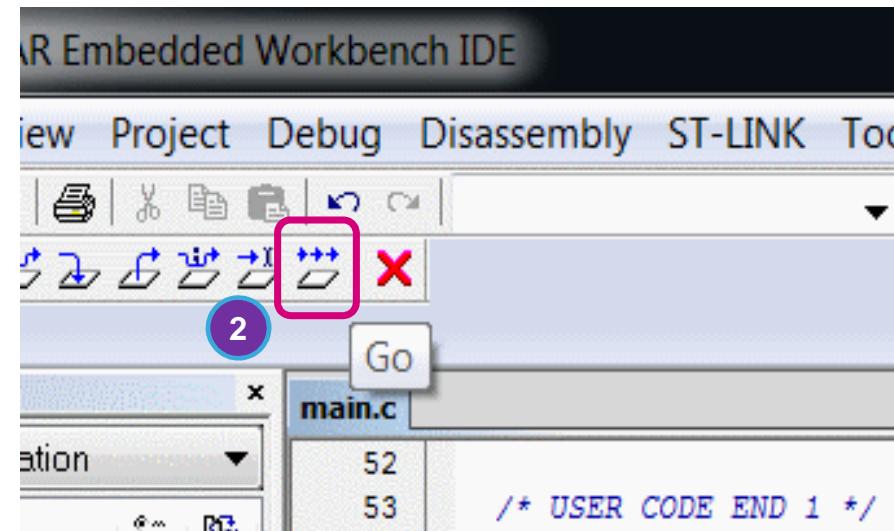
Load and Run

142

1. Click the GREEN ARROW to Build the Project, **Download** and start the **debugger**. (Ctrl + D)



2. Click the triple-arrow GO button! (F5)



Debugging the Firmware : Hitting the Breakpoint

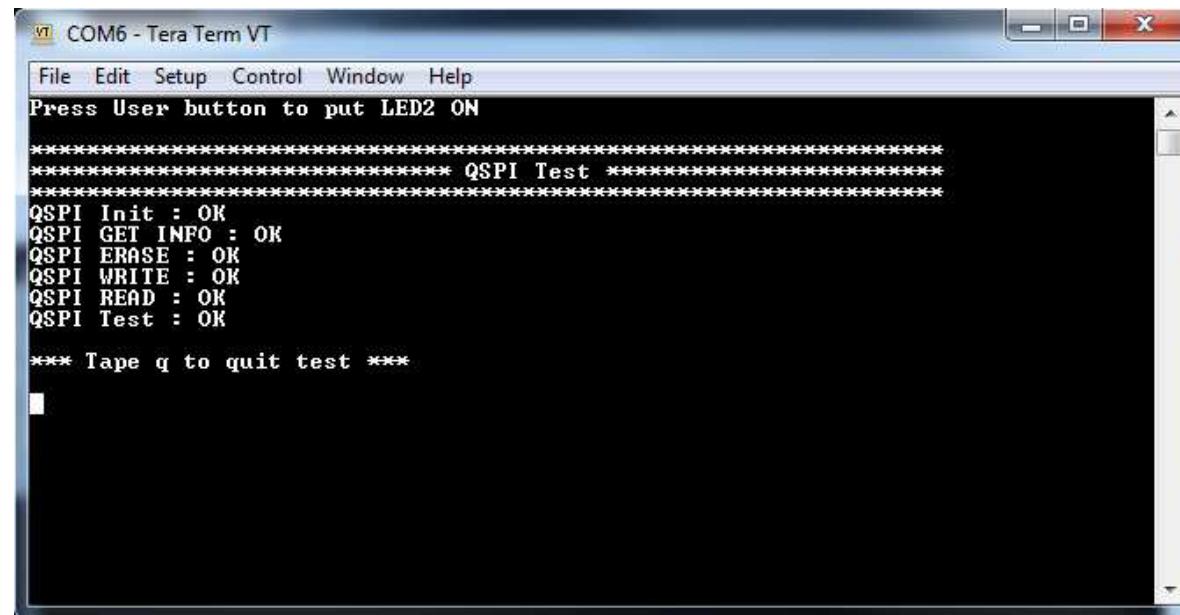
143



- In Tera Term, you will be asked to press the **BLUE** User button.
 - This will turn on **LED2**

Debugging the Firmware : Hitting the Breakpoint

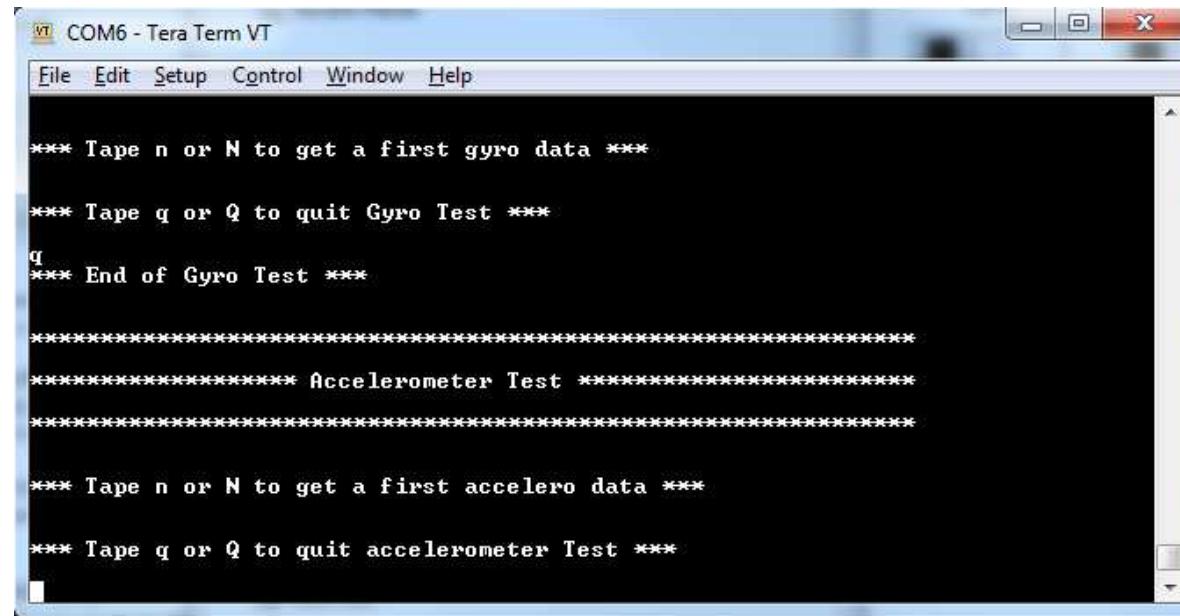
144



- Hit **Q** to quit each test option until you get to the **Accelerometer Test**

Debugging the Firmware : Hitting the Breakpoint

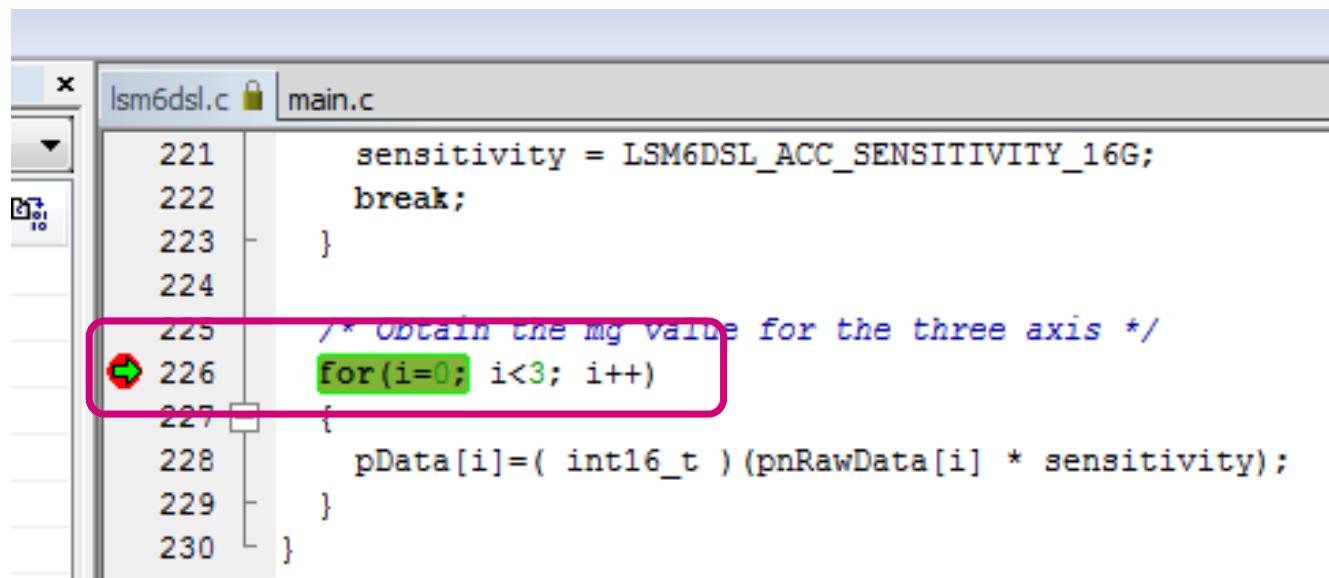
145



- At the **Accelerometer Test**, hit **N** to retrieve real time accelerometer data

Debugging the Firmware : Hitting the Breakpoint

146



The screenshot shows a code editor window for the file 'lsm6dsl.c'. The current tab is 'main.c'. The code is as follows:

```
221     sensitivity = LSM6DSL_ACC_SENSITIVITY_16G;
222     break;
223 }
224
225 /* obtain the mg values for the three axis */
226 for(i=0; i<3; i++)
227 {
228     pData[i]=( int16_t )(pnRawData[i] * sensitivity);
229 }
230 ...
```

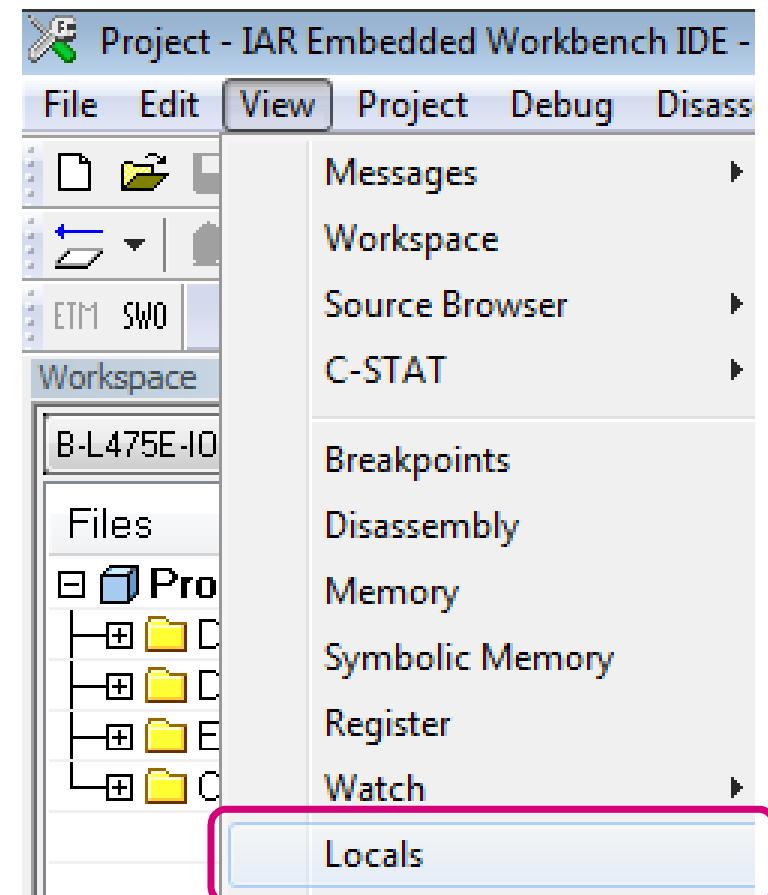
A red circle with a green arrow points to the start of the line 226 'for' loop. A red rounded rectangle highlights the entire line 226. The line numbers are on the left, and the code is on the right.

- In IAR, you will hit the previously set breakpoint at line **226**.

Debugging the Firmware : Viewing Local Variables

147

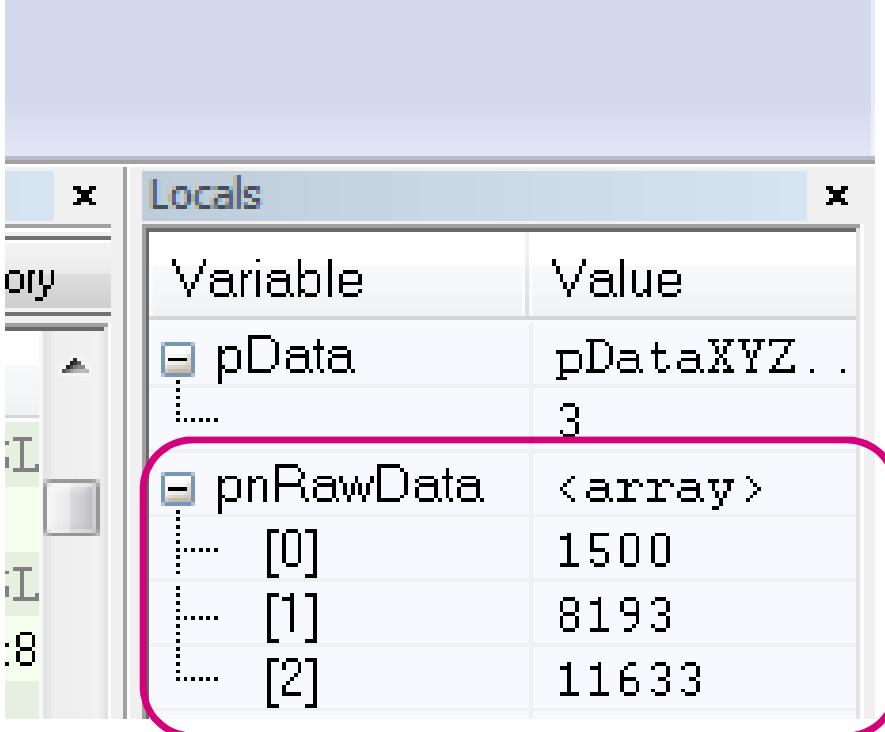
- In IAR, click on **View -> Locals**



Debugging the Firmware : Viewing Local Variables

148

- A sub window will open on the right side of the IDE, with the title “Locals”.
- Expand the `pnRawData` variable array to view the raw accelerometer reading.

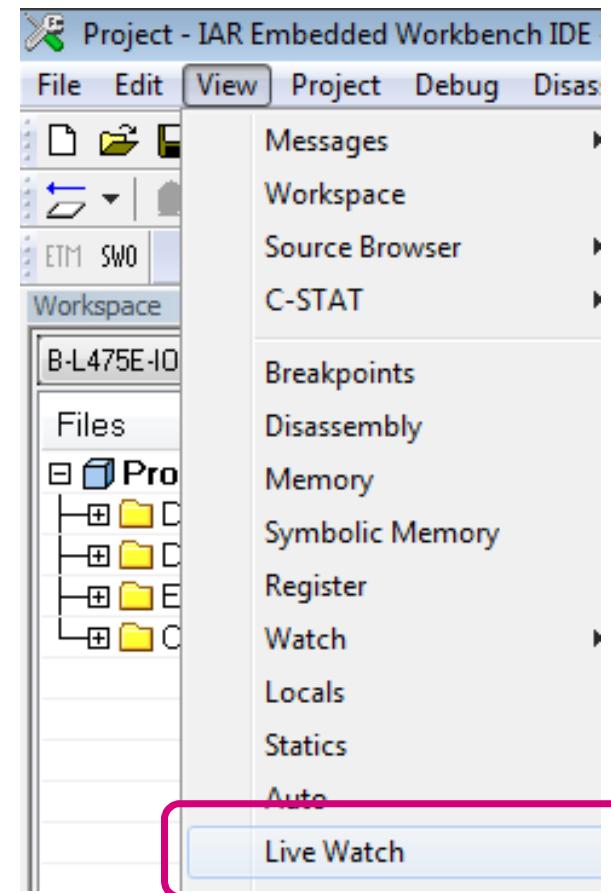


Variable	Value
pData	pDataXYZ...
...	3
pnRawData	<array>
[0]	1500
[1]	8193
[2]	11633

Debugging the Firmware : Viewing Global Variables

149

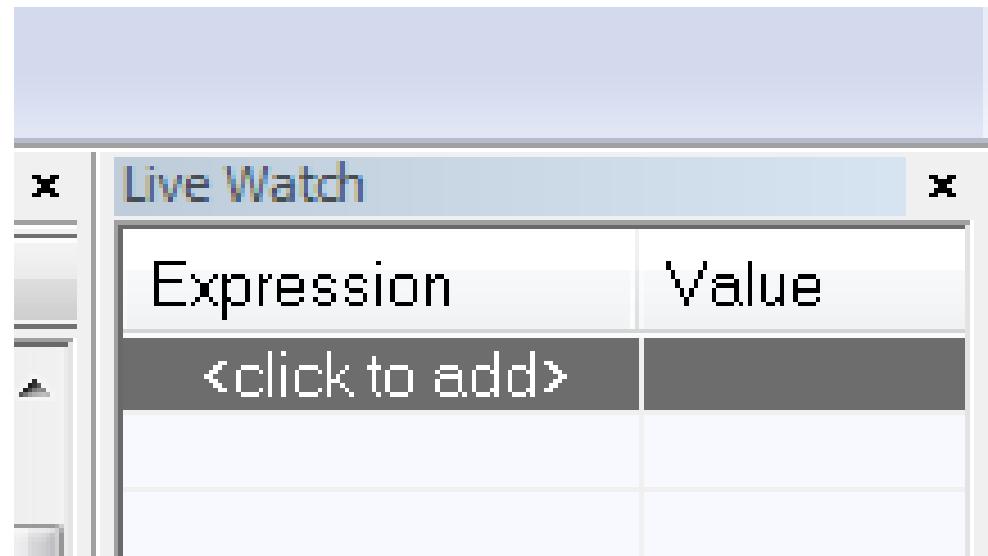
- In IAR, click on **View -> Live Watch**



Debugging the Firmware : Viewing Global Variables

150

- A sub window will open on the right side of the IDE, with the title “Live Watch”
- Click on **<click to add>**



Debugging the Firmware : Viewing Global Variables

151

- Type `pDataXYZ` (case-sensitive)
- Expand `pDataXYZ` variable array to view the accelerometer reading.

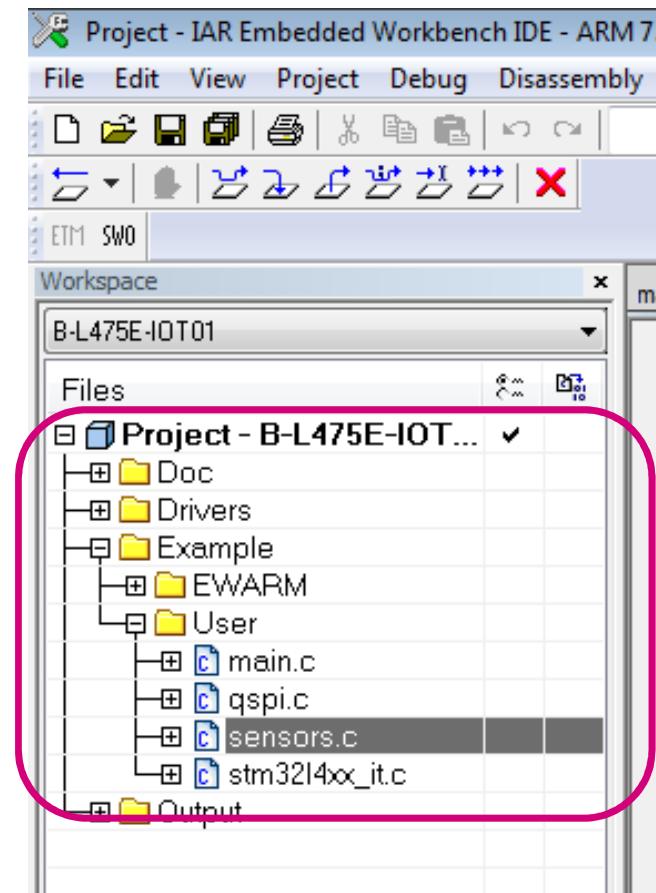
Expression	Value
<code>pDataXYZ</code>	<array>
[0]	1
[1]	-5
[2]	1032
<click to add>	

Debugging the Firmware : Viewing Global Variables

152

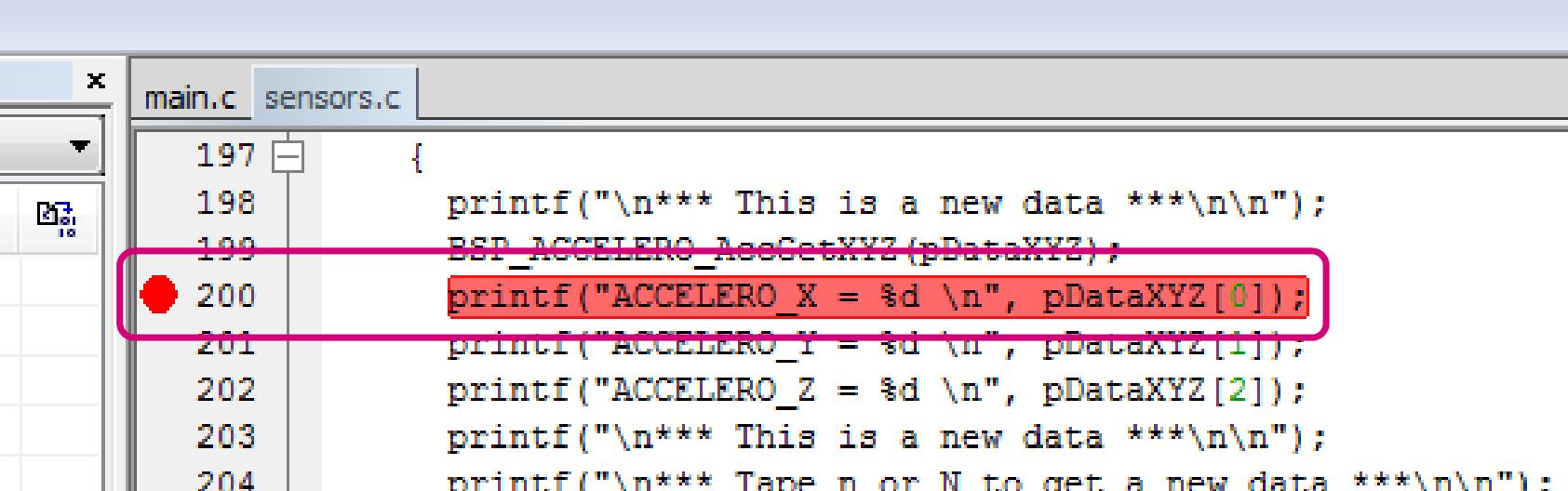
Expand the file tree:

1. Expand **Project – B-L475E-IOT1**
2. Expand **Example**
3. Expand **User**
4. Double-click **sensors.c**



Debugging the Firmware : Viewing Global Variables

153



The screenshot shows a code editor window with two tabs: "main.c" and "sensors.c". The "main.c" tab is active. The code is as follows:

```
197     {
198         printf("\n*** This is a new data ***\n\n");
199         BSP_ACCELERO_AccGetXYZ(pDataXYZ);
200         printf("ACCELERO_X = %d \n", pDataXYZ[0]);
201         printf("ACCELERO_Y = %d \n", pDataXYZ[1]);
202         printf("ACCELERO_Z = %d \n", pDataXYZ[2]);
203         printf("\n*** This is a new data ***\n\n");
204         printf("\n*** Tape n or N to get a new data ***\n\n");
```

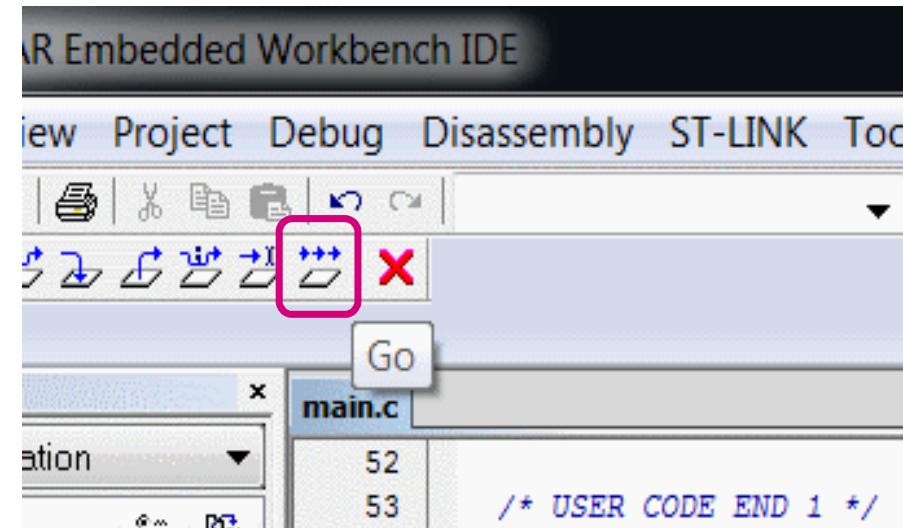
A red circle is placed on the line number 200. A red rectangular box highlights the entire line 200, specifically the printf statement. The code editor has a light gray background with dark gray horizontal and vertical scroll bars.

- Set a break point at line 200 by left-clicking on the left side of the line number.

Debugging the Firmware : Viewing Global Variables

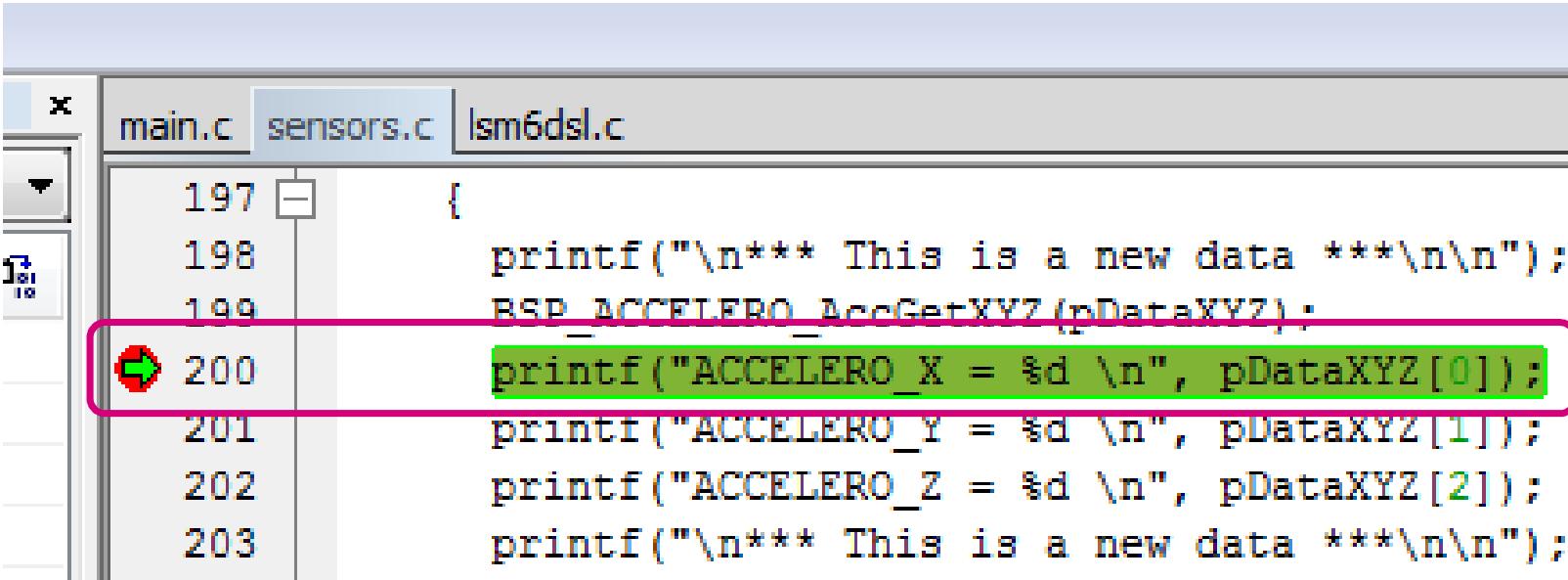
154

- Click the triple-arrow **GO** button! (F5)



Debugging the Firmware : Viewing Global Variables

155



The screenshot shows the IAR Embedded Workbench IDE interface. The tabs at the top are 'main.c', 'sensors.c' (which is currently selected), and 'lsm6dsl.c'. The code editor displays the 'sensors.c' file. A red box highlights line 200, which contains a printf statement. A green arrow points to the first byte of the string in this line, indicating where the debugger will stop. The code is as follows:

```
197     {
198         printf("\n*** This is a new data ***\n\n");
199         BSP_ACCELEROM_AccGetXYZ(pDataXYZ);
200         printf("ACCELEROM_X = %d \n", pDataXYZ[0]);
201         printf("ACCELEROM_Y = %d \n", pDataXYZ[1]);
202         printf("ACCELEROM_Z = %d \n", pDataXYZ[2]);
203         printf("\n*** This is a new data ***\n\n");
```

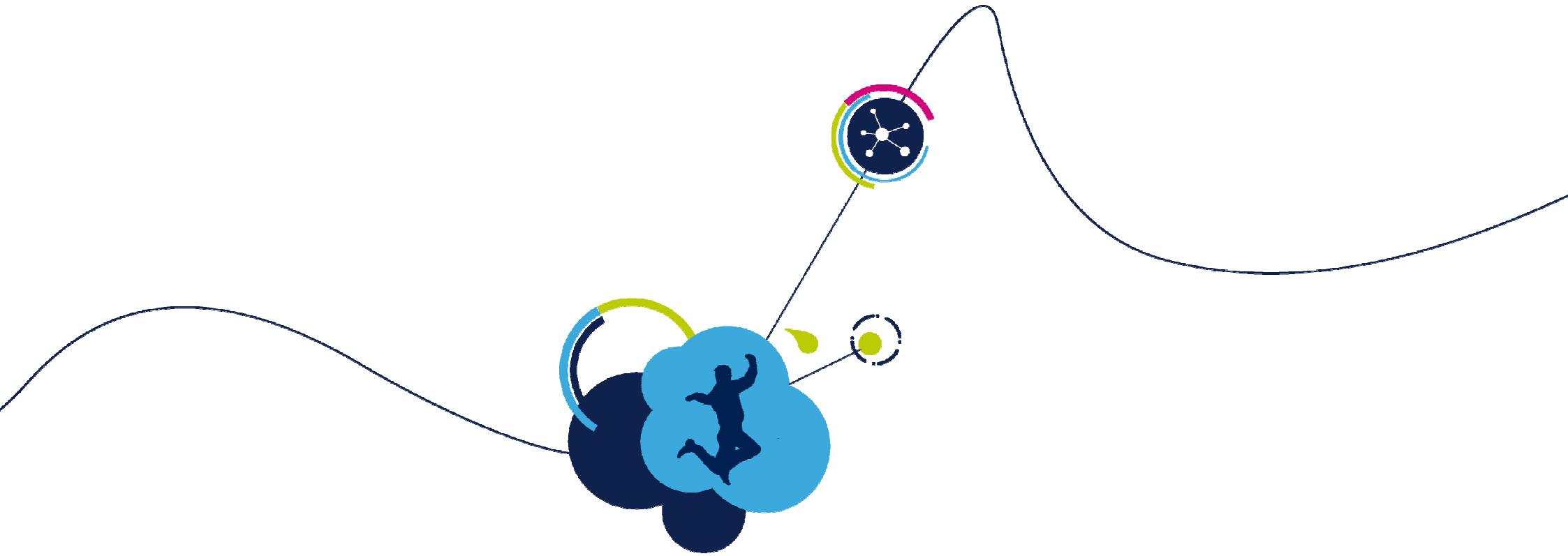
- In IAR, you will hit the previously set breakpoint at line **200**.

Debugging the Firmware : Viewing Global Variables

156

- In the Live Watch window, **pDataXYZ** will now have the normalized accelerometer data (in units of 'g').

Expression	Value
pDataXYZ	<array>
[0]	5
[1]	-7
[2]	1031
<click to add>	



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

159

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

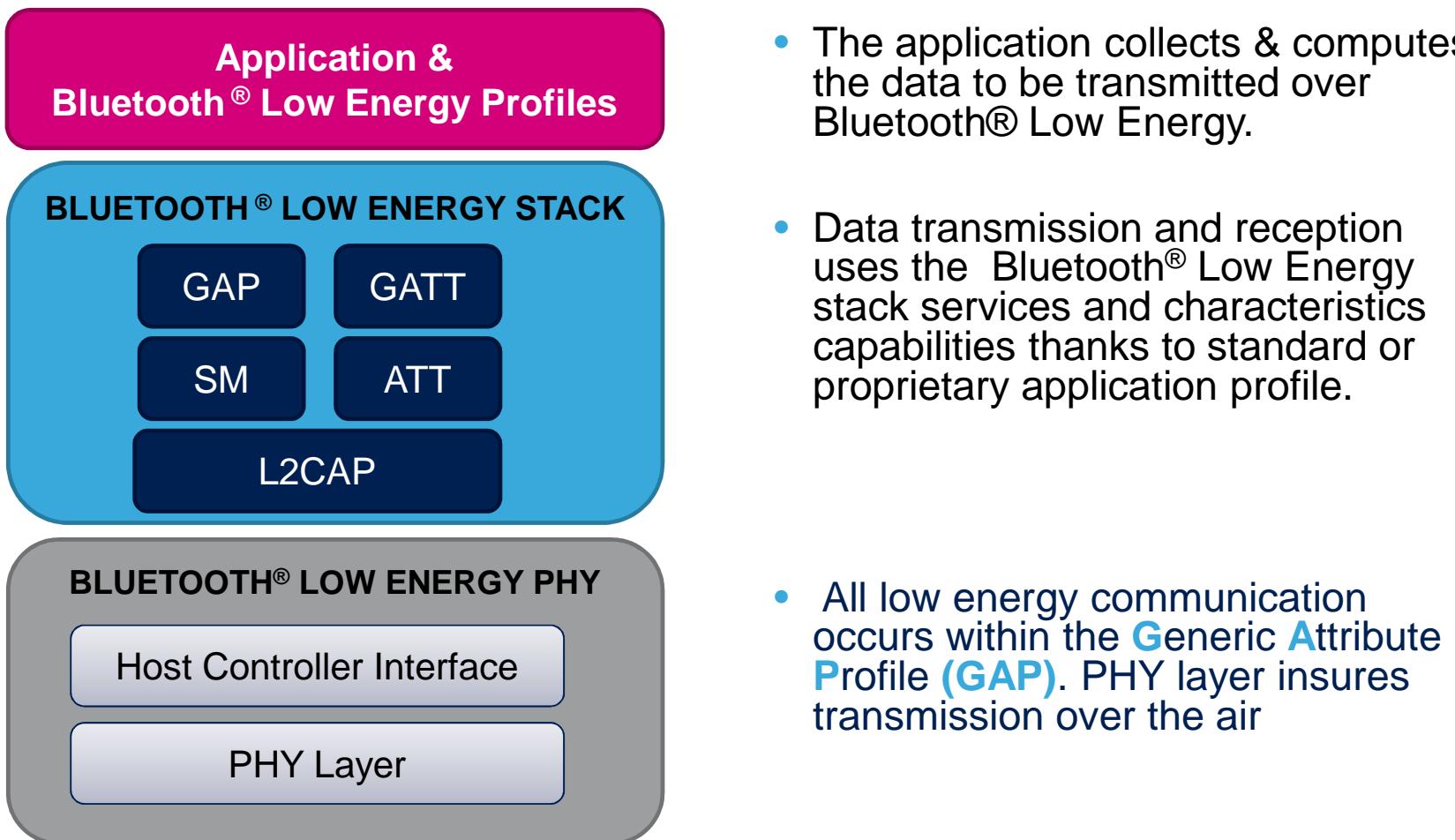


- An implementation of the Bluetooth® core system has only one Primary Controller which may be one of the following configurations:
 - BR/EDR Controller (3.0 and earlier)
 - LE (low energy) Controller (4.0 and newer)
 - Combined BR/EDR Controller portion and LE controller portion into a single Controller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

Source: Bluetooth® SIG

Bluetooth® Low Energy Stack Partitioning

160



ST BlueNRG-MS Solution

Integration



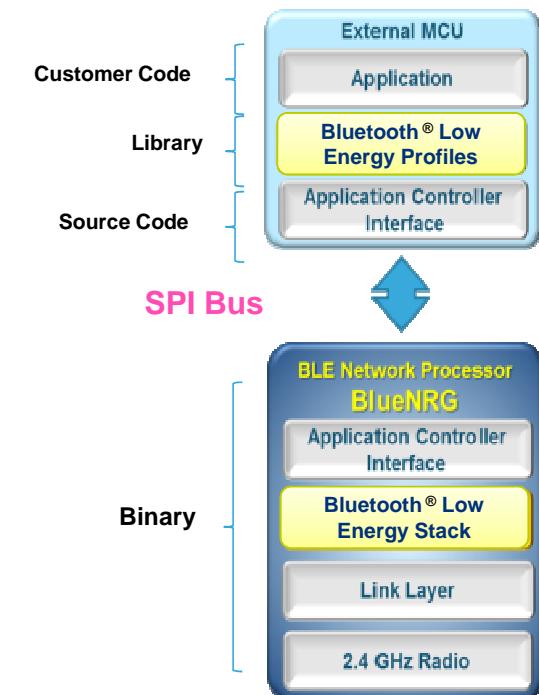
- Single mode Bluetooth® Low Energy wireless network processor
 - 2.4GHz RF transceiver
 - Cortex-M0 microcontroller (running the BT MS stack)
 - AES 128-bit co-processor
 - Master and Slave Mode Bluetooth® Low Energy (4.1) Network Processor.
 - On chip non-volatile Flash memory allows OTA stack upgrade.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

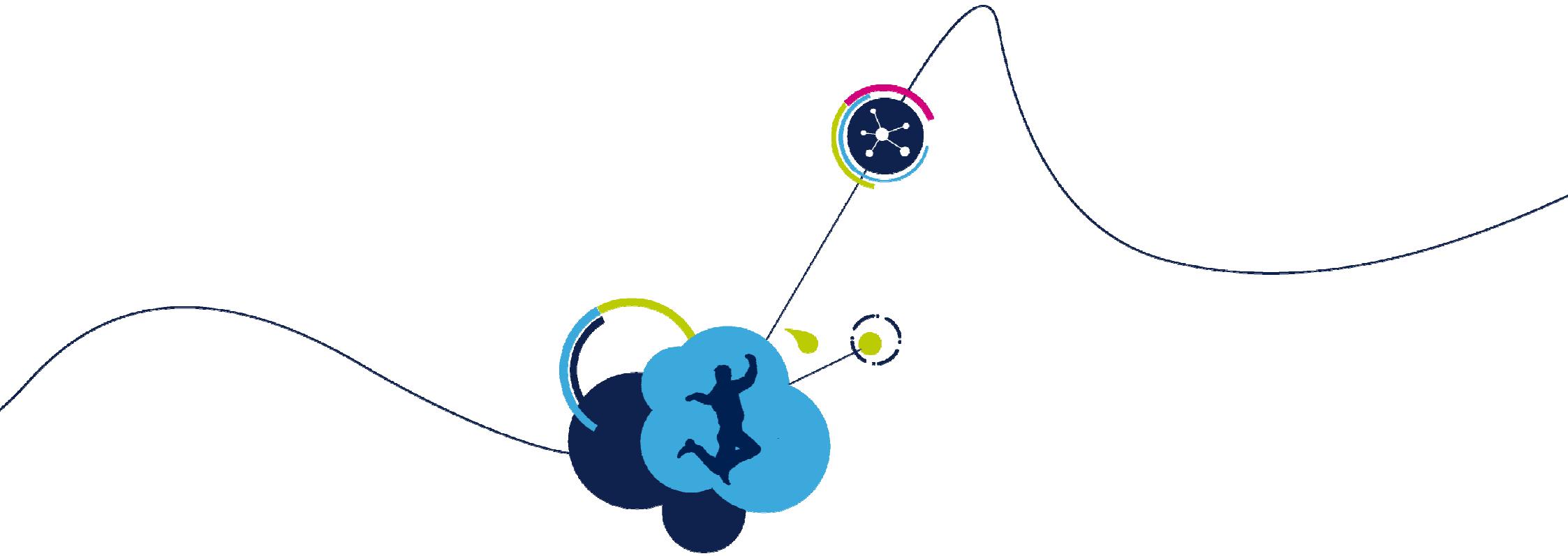
Flexibility

Low power

Best in Class

Small size





Lab 3 : Bluetooth® Low Energy Pairing

Lab Goal

163

- 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.

Lab Goal

164

- 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 Bluetooth Low Energy (BLE) devices implementing peripheral profiles. It supports X-NUCLEO-IDB04A1 / X-NUCLEO-IDB05A1 BlueNRG expansion boards running

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

What's New in Version 2.0

- New app name STM32 BLE Profiles

Open the BlueNRG_HandsOn Project

- 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 IAR project.
 2. Double click on the **HR.eww** file located here:

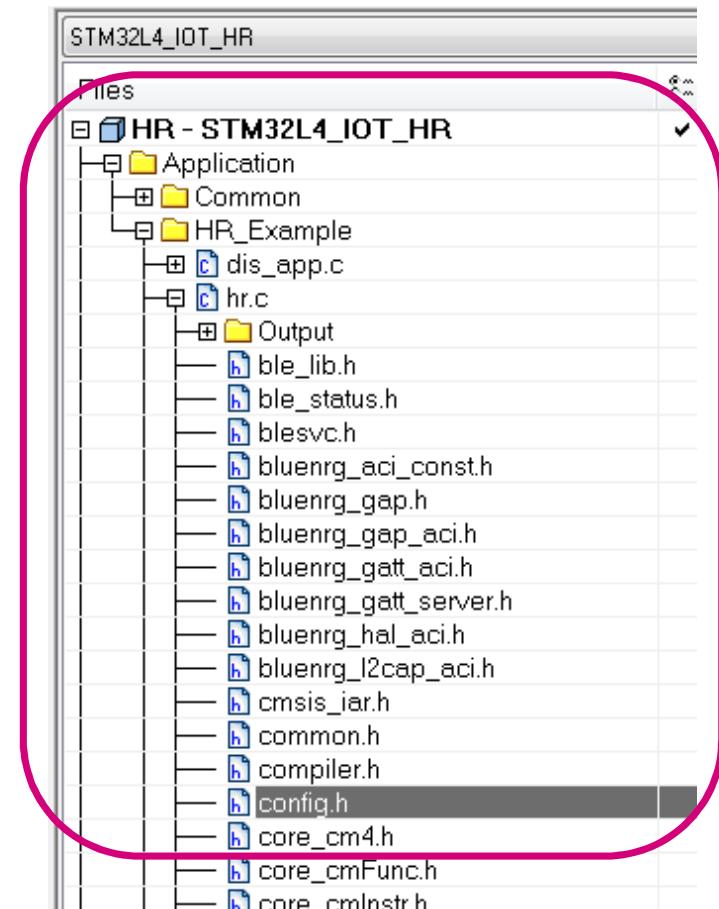
C:\STM32L4_DK_IoT_Node_Seminar\Hands-on\Labs\Projects\B-L475E-IOT01\Applications\BLE\HeartRate\EWARM

BlueNRG Module configuration (1/4)

166

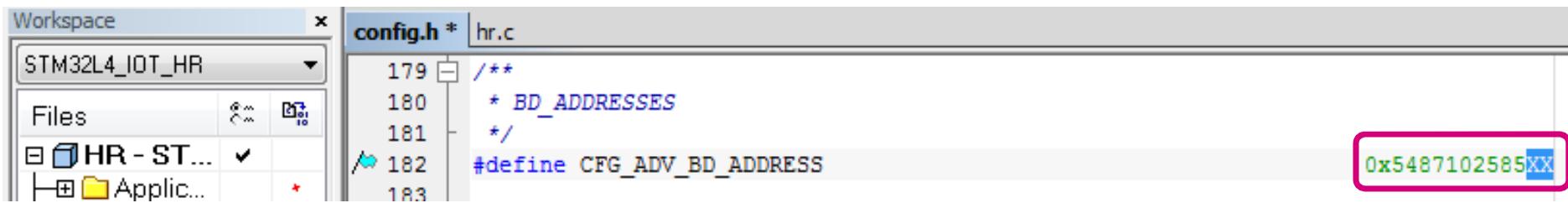
Expand the file tree:

1. Expand **HR – STM32L4_IOT_HR**
2. Expand **Application**
3. Expand **HR_Example**
4. Expand **hr.c**
5. Double-click **config.h**



BlueNRG Module configuration (2/4)

167



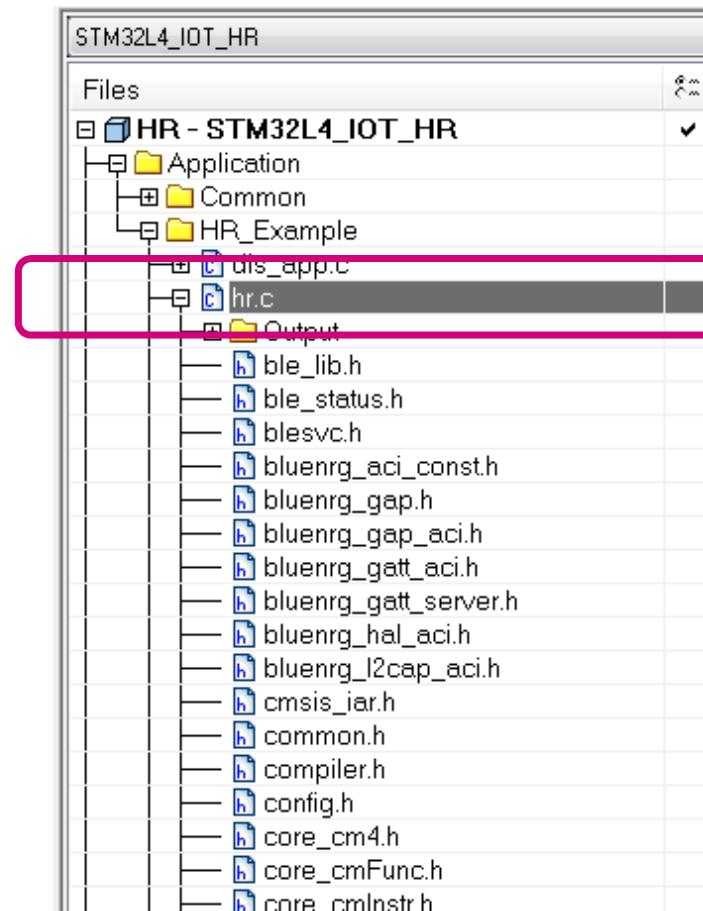
```
Workspace * x
STM32L4_IOT_HR
Files hr.c
179  /**
180  * @BD_ADDRESSES
181  */
182 #define CFG_ADV_BD_ADDRESS 0x5487102585XX
183
```

- Replace the 'xx' in the **CFG_ADV_BD_ADDRESS** (line 182) with any desired number (2 characters) for customization, thus avoiding issues with 2 or more boards.

BlueNRG Module configuration (3/4)

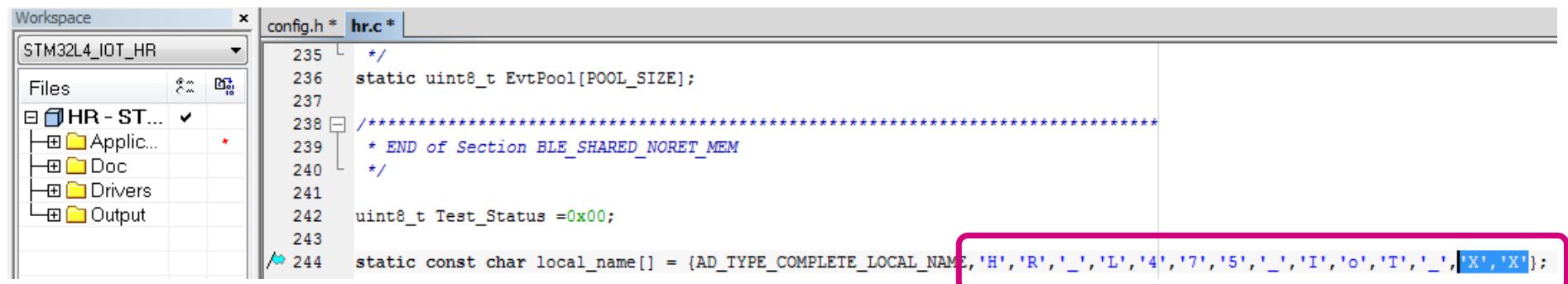
168

1. Double-click **hr.c**



BlueNRG Module configuration (4/4)

169



```
Workspace STM32L4_IOT_HR
Files HR - ST... ✓
  Applic...
  Doc
  Drivers
  Output

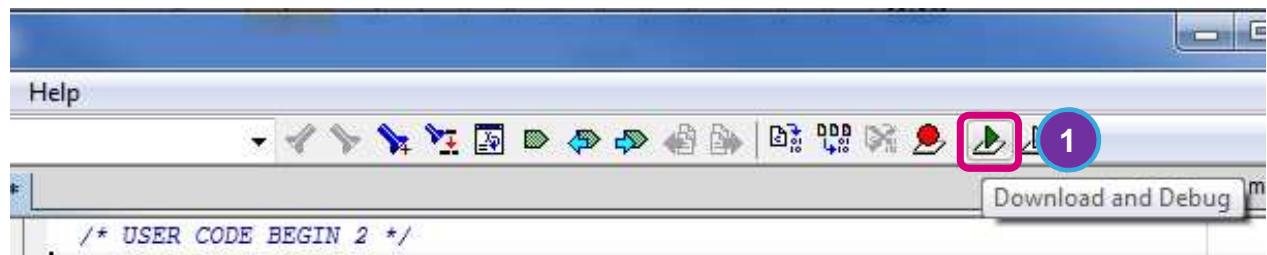
config.h * hr.c *
235  */
236  static uint8_t EvtPool[POOL_SIZE];
237
238  ****
239  * END of Section BLE_SHARED_NORET_MEM
240  */
241
242  uint8_t Test_Status =0x00;
243
244  static const char local_name[] = {AD_TYPE_COMPLETE_LOCAL_NAME, 'H', 'R', '_', 'I', '4', '7', '5', '_', 'I', 'o', 'T', '_', 'X',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

- Replace the 'x', 'x' in the `local_name` (line 244) table with any number (2 characters).

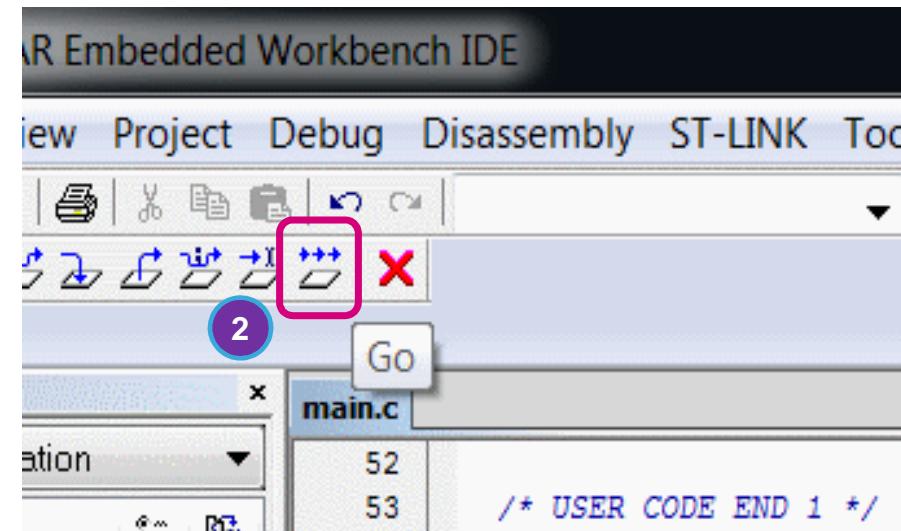
Load and Run

170

1. Click the GREEN ARROW to Build the Project, Download and start the debugger. (Ctrl + D)



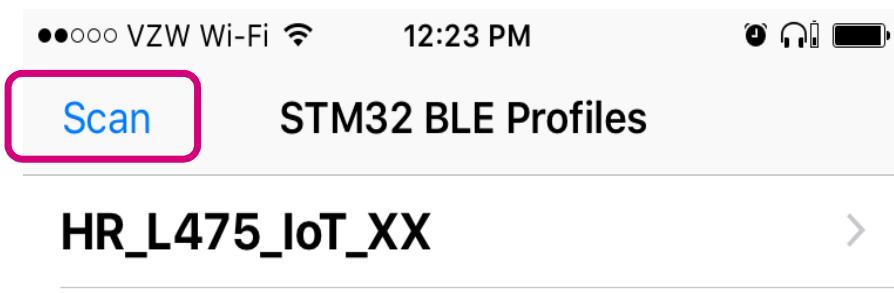
2. Click the triple-arrow GO button! (F5)



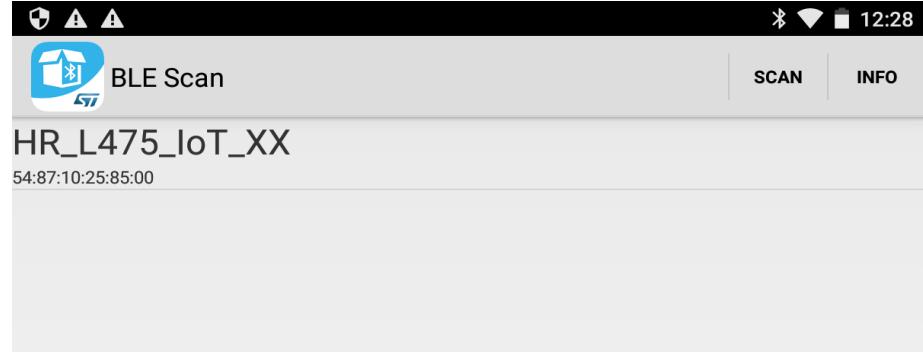
Pair with STM32 BLE Profiles App

171

iOS



Android

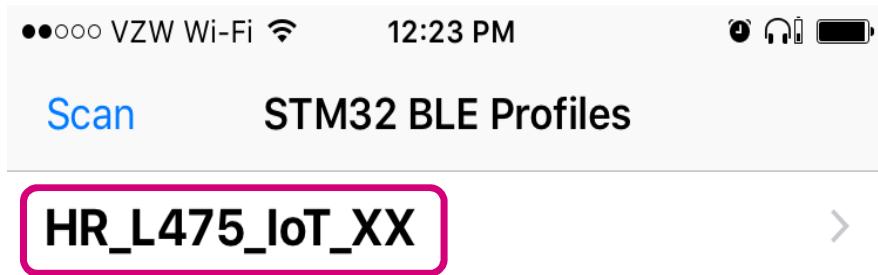


- Make sure Bluetooth® is active on your phone
- Using your phone, open the **STM32 BLE Profiles** app.
- For iOS users click on **Scan**.

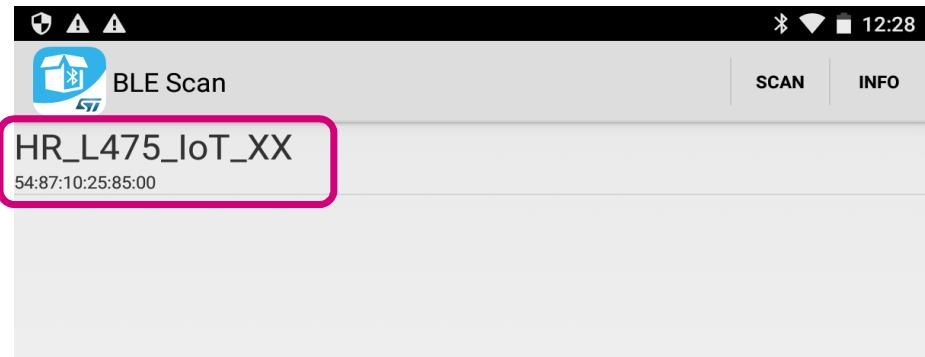
Pair with STM32 BLE Profiles App

172

iOS



Android



- 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

173

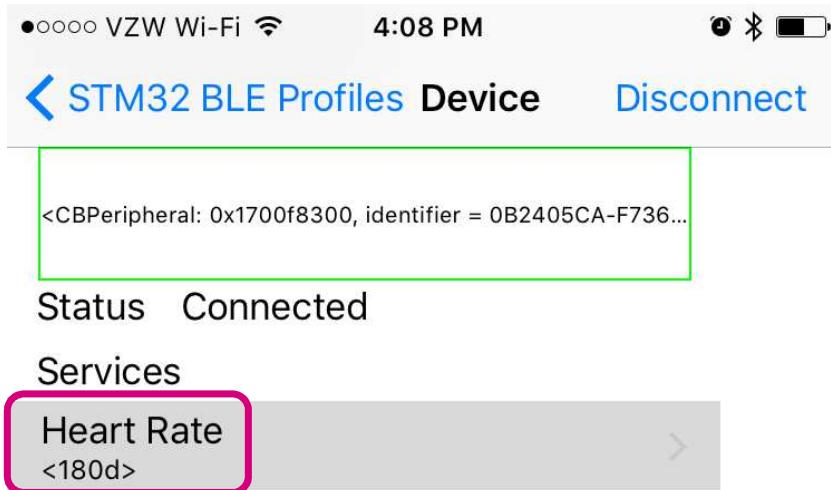


- iOS users, please click **Connect** on the next screen (iOS)

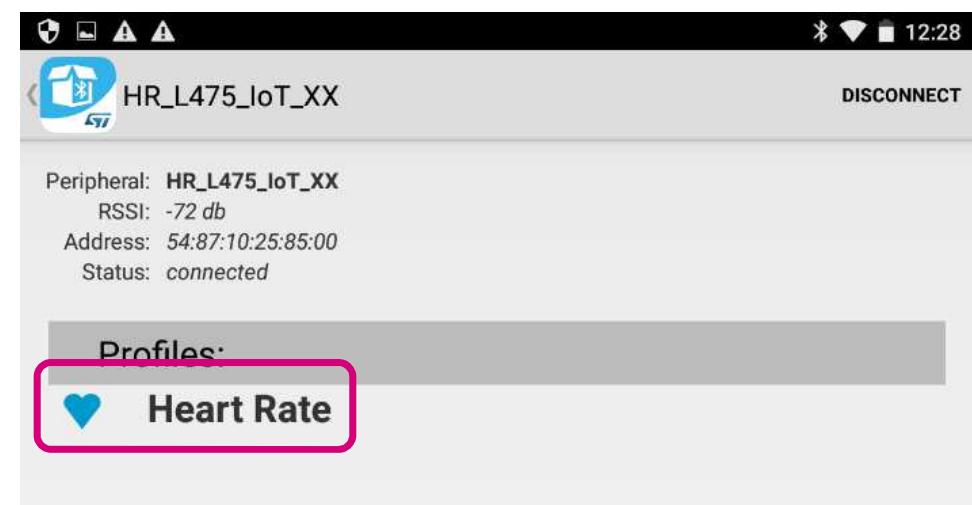
Select the Heart Rate Profile

174

iOS



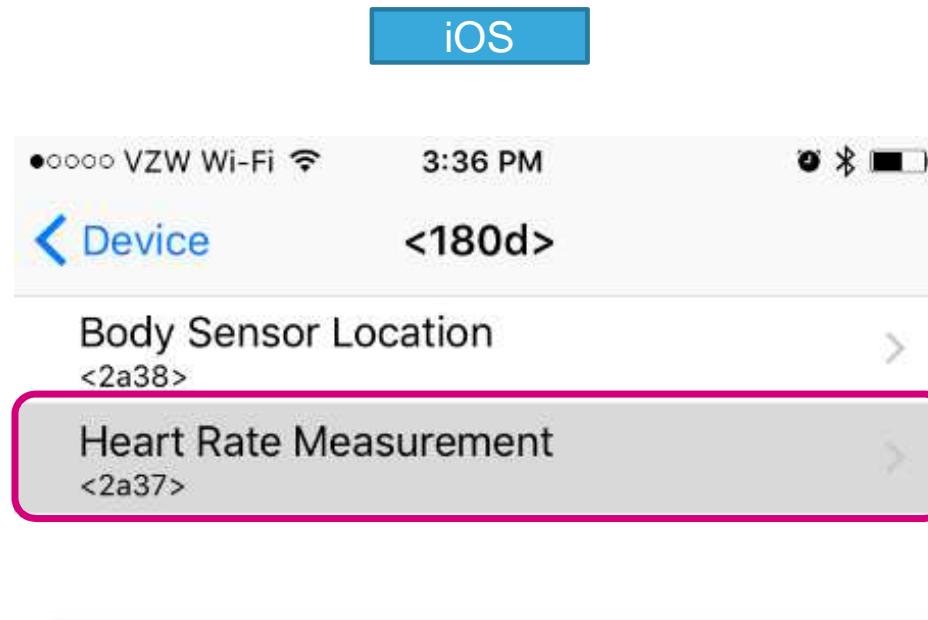
Android



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

Select the Heart Rate Profile

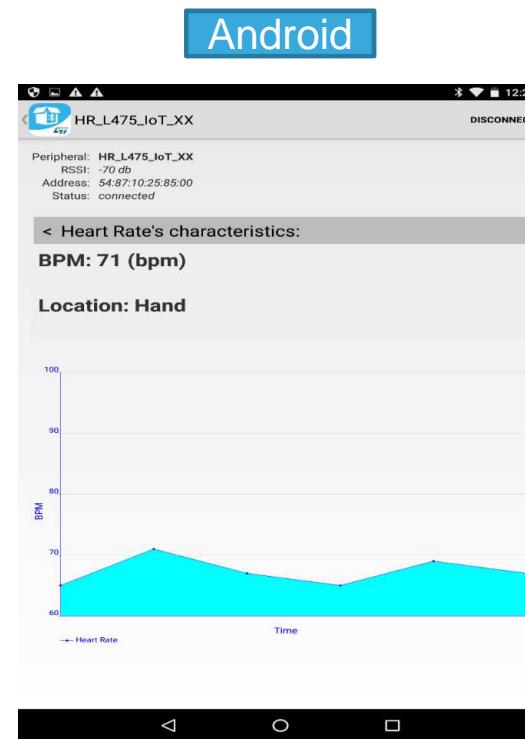
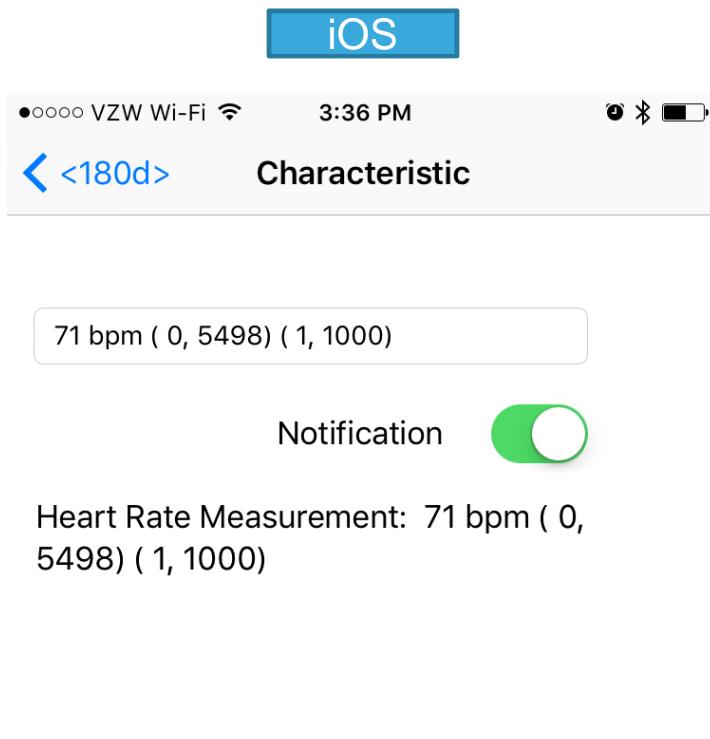
175



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

Display HR Data

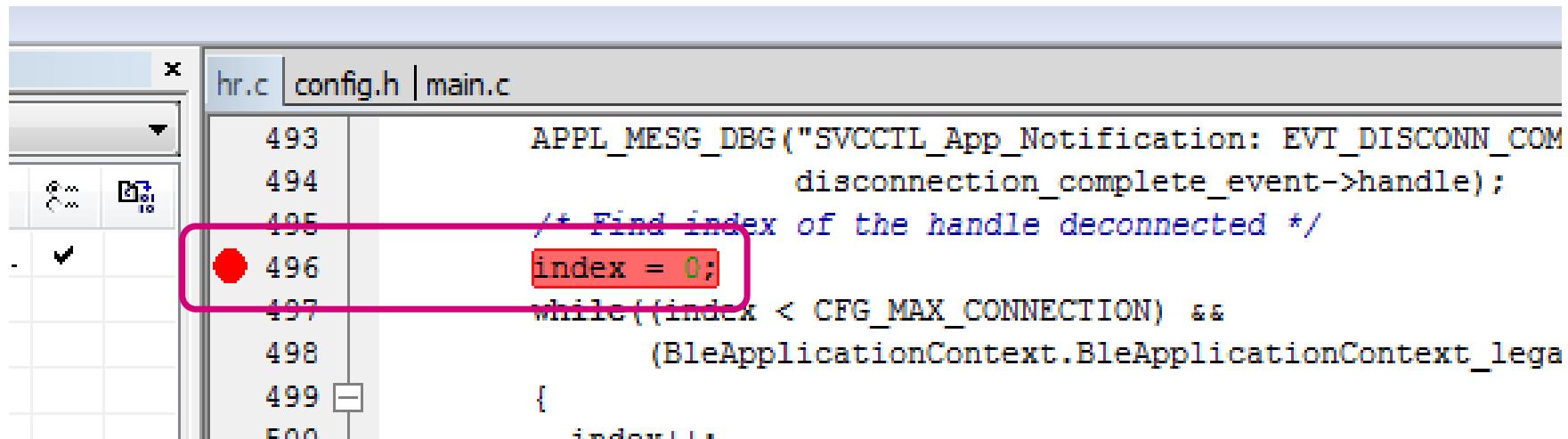
176



- You should see the simulated heart rate.

Debug the Firmware

177



The screenshot shows a code editor window in IAR Embedded Workbench. The tabs at the top are 'hr.c' (which is active), 'config.h', and 'main.c'. The code in 'hr.c' is as follows:

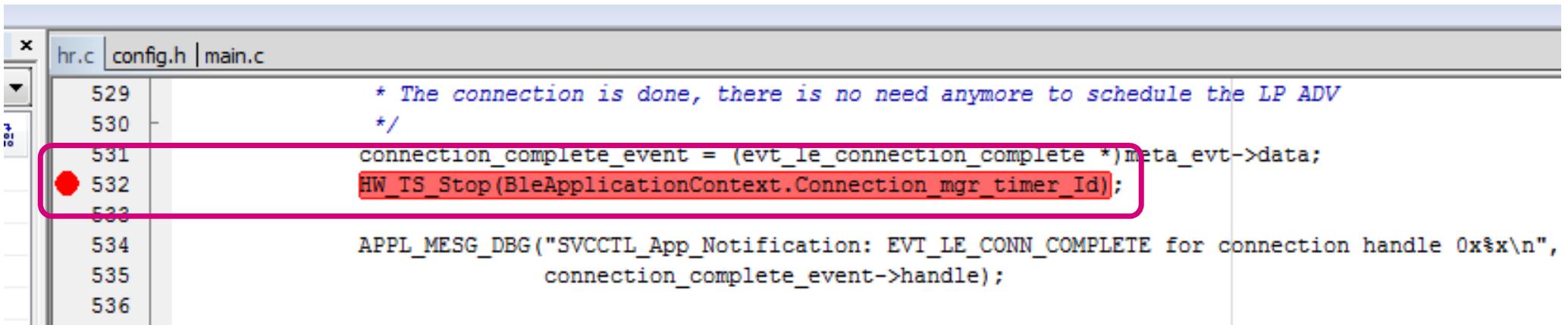
```
493     APPL_MESG_DBG ("SVCCTL_App_Notification: EVT_DISCONN_COM
494                     disconnection_complete_event->handle);
495     /* Find index of the handle disconnected */
496     index = 0;
497     while ((index < CFG_MAX_CONNECTION) &&
498           (BleApplicationContext.BleApplicationContext_lega
499     {
500         index++;
```

A red circle marks a break point at line 496. A red rectangle highlights the assignment statement 'index = 0;'. The code uses color coding for syntax: blue for keywords like APPL_MESG_DBG, disconnection, while, CFG_MAX_CONNECTION, BleApplicationContext, and index; and green for comments and strings.

- In IAR, double-click on the **hr.c** file
- Set a break point at line **496**.

Debug the Firmware

178



The screenshot shows a code editor window with three tabs: hr.c, config.h, and main.c. The main.c tab is active. The code is as follows:

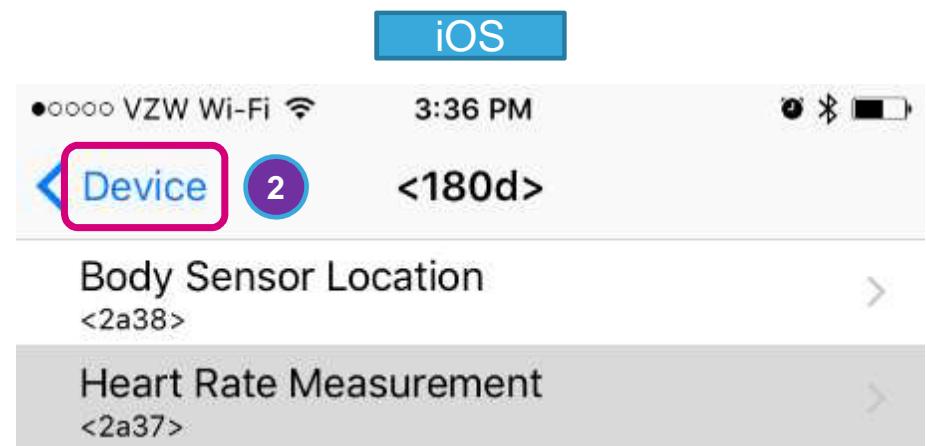
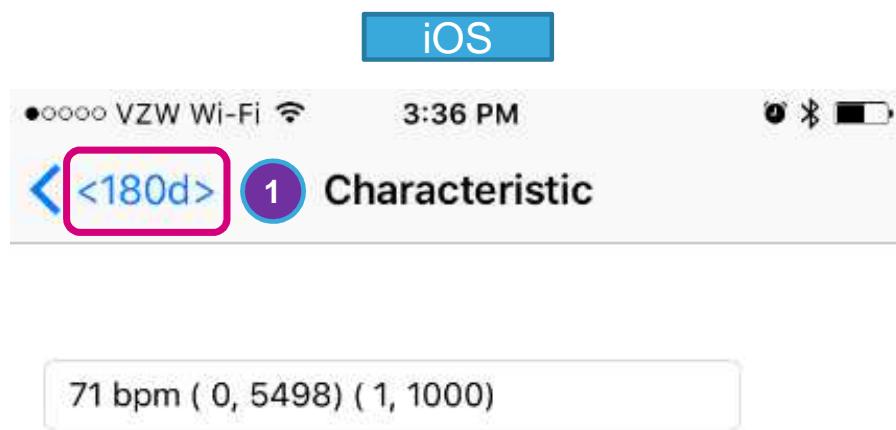
```
529         * The connection is done, there is no need anymore to schedule the LP ADV
530         */
531     connection_complete_event = (evt_le_connection_complete *)meta_evt->data;
532     HW_TS_Stop(BleApplicationContext.Connection_mgr_timer_Id);
533
534     APPL_MESG_DBG("SVCCTL_App_Notification: EVT_LE_CONN_COMPLETE for connection handle 0x%x\n",
535                   connection_complete_event->handle);
536
```

A red circle marks the breakpoint at line 532. A red rectangle highlights the entire line 532, which contains the call to `HW_TS_Stop`. The code editor has a light gray background with dark gray syntax highlighting.

- Set another break point at line 532.
- These break points will halt program execution when a client is connected & disconnected to the device.

Debug the Firmware (Disconnect)

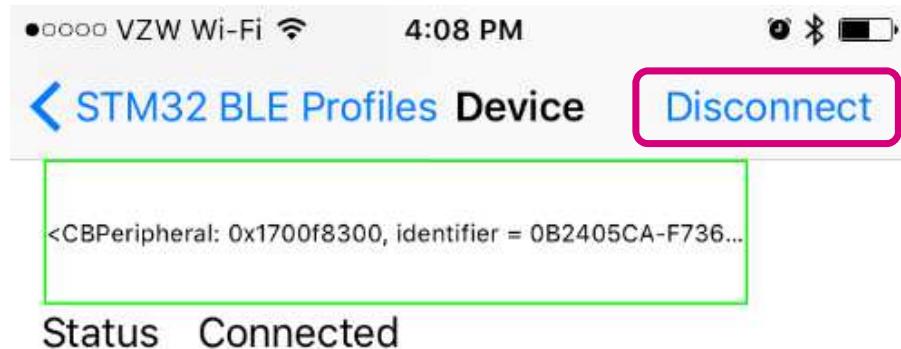
179



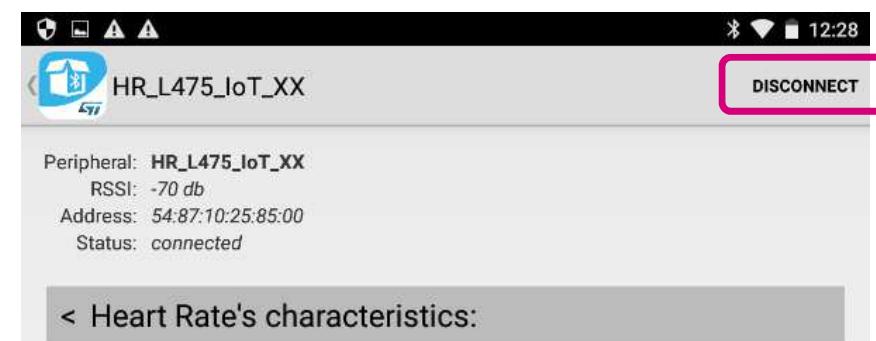
- iOS users, disconnect from your device by:
 1. Clicking on <180d>
 2. Clicking on < Device

Debug the Firmware (Disconnect)

iOS

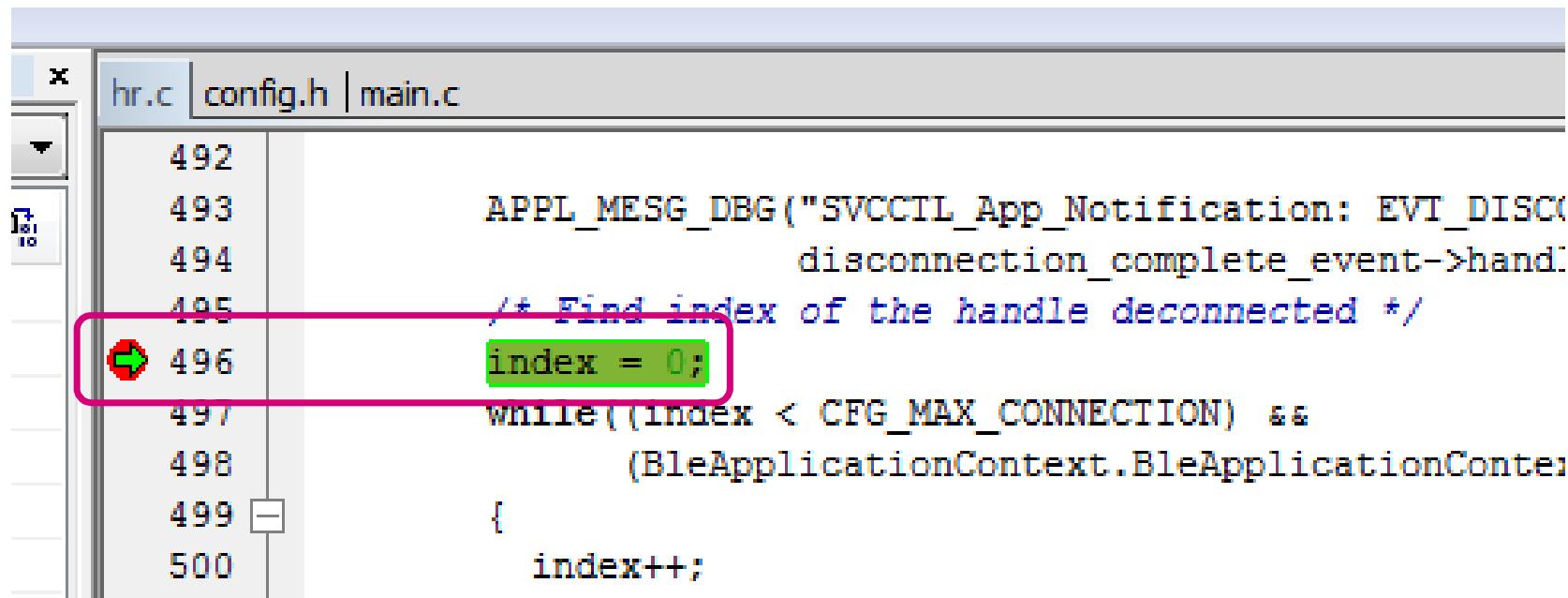


Android



- Click **Disconnect** on your device

Debug the Firmware (Disconnect)



The screenshot shows a code editor window for the file 'hr.c'. The tab bar at the top has 'hr.c' selected, followed by 'config.h' and 'main.c'. The code itself is as follows:

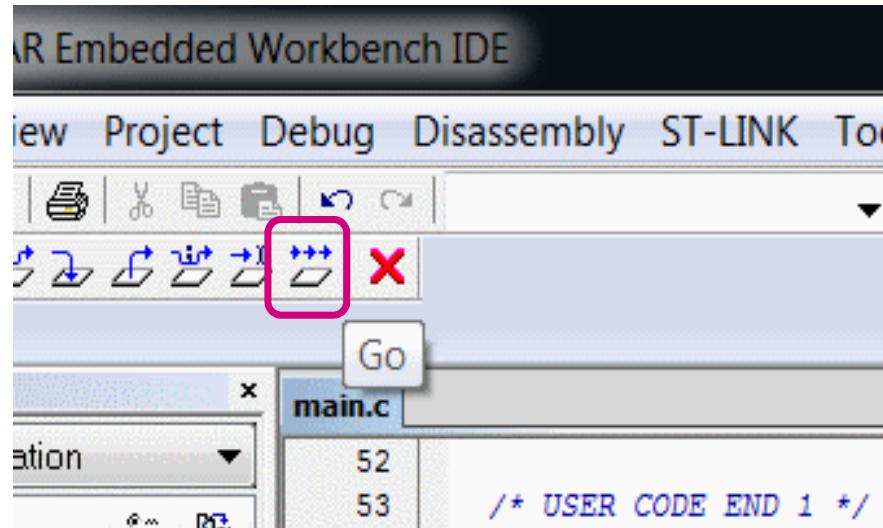
```
492
493     APPL_MESG_DBG("SVCCIL_App_Notification: EVT_DISC(
494                     disconnection_complete_event->handle);
495     /* Find index of the handle disconnected */
496     index = 0;
497     while((index < CFG_MAX_CONNECTION) &&
498           (BleApplicationContext.BleApplicationConte;
499     {
500         index++;
```

A red circle with a green arrow points to the line number 496, which contains the assignment 'index = 0;'. This indicates that a breakpoint has been set at this specific line of code.

- In IAR, you will hit the previously set breakpoint at line 496.

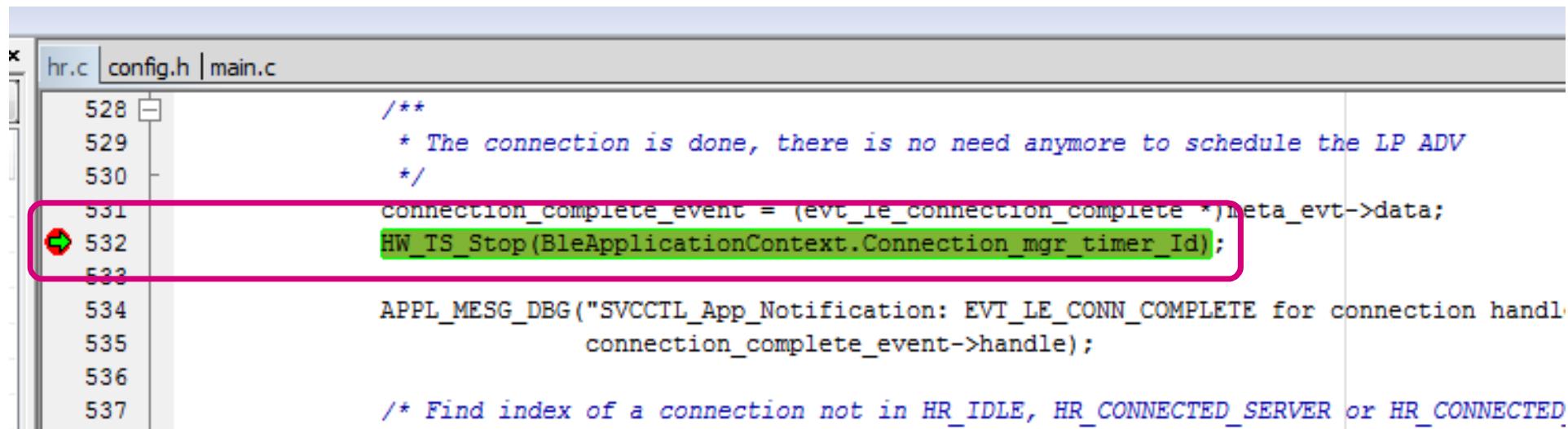
Debug the Firmware (Connect)

182



- Resume code execution by pressing the **Go** button (**F5**) on IAR.
- Re-connect to the device from your phone.

Debug the Firmware (Connect)

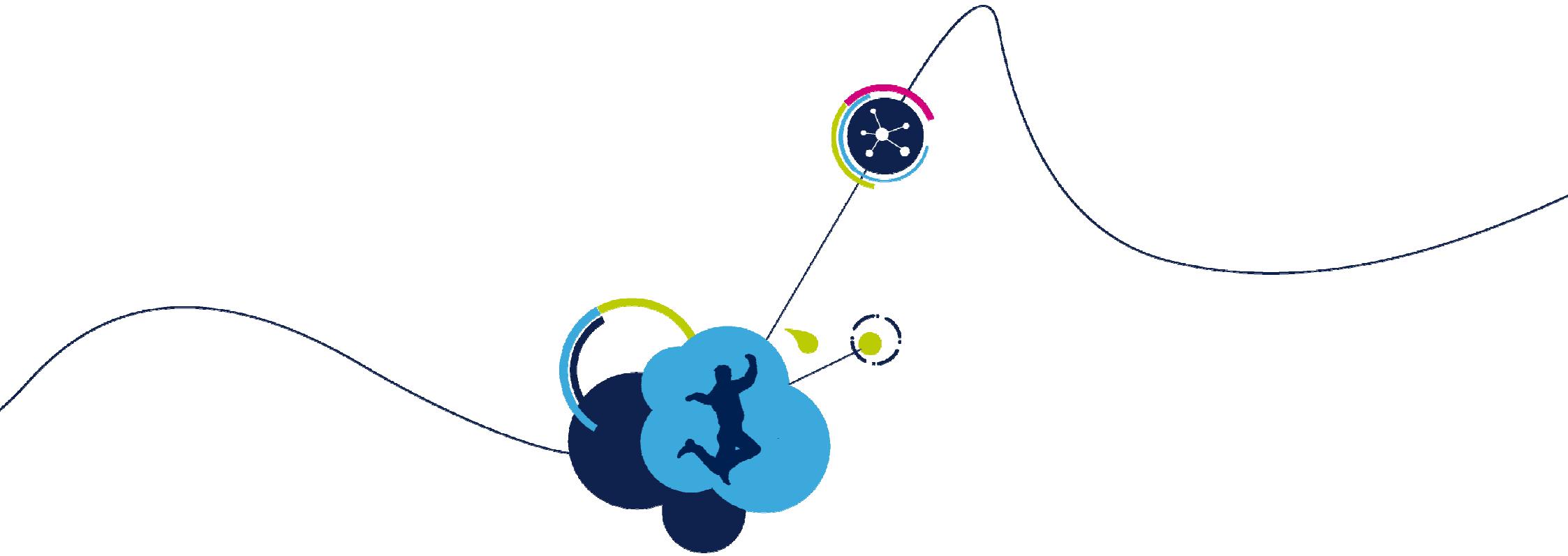


The screenshot shows a code editor window with three tabs: hr.c, config.h, and main.c. The hr.c tab is active. The code is as follows:

```
528     /**
529      * The connection is done, there is no need anymore to schedule the LP ADV
530      */
531     connection_complete_event = (evt_ie_connection_complete *)meta_evt->data;
532     HW_TS_Stop(BleApplicationContext.Connection_mgr_timer_Id);
533
534     APPL_MESG_DBG("SVCCTL_App_Notification: EVT_LE_CONN_COMPLETE for connection handle
535                   connection_complete_event->handle);
536
537     /* Find index of a connection not in HR_IDLE, HR_CONNECTED_SERVER or HR_CONNECTED.
```

A red rectangle highlights line 532, and a green rectangle highlights the entire line 532. A red circle with a green cross is placed on the left margin next to line 532, indicating it is a breakpoint.

- In IAR, you will hit the previously set breakpoint at line **532**.



Lab 4 : Bluetooth® Low Energy P2P Communication

Lab Goal

185

- This lab will demonstrate Point To Point communication using Bluetooth® Low Energy.
- In this lab, I will work with a pair of kits. One board will run as a server, the second as a client.

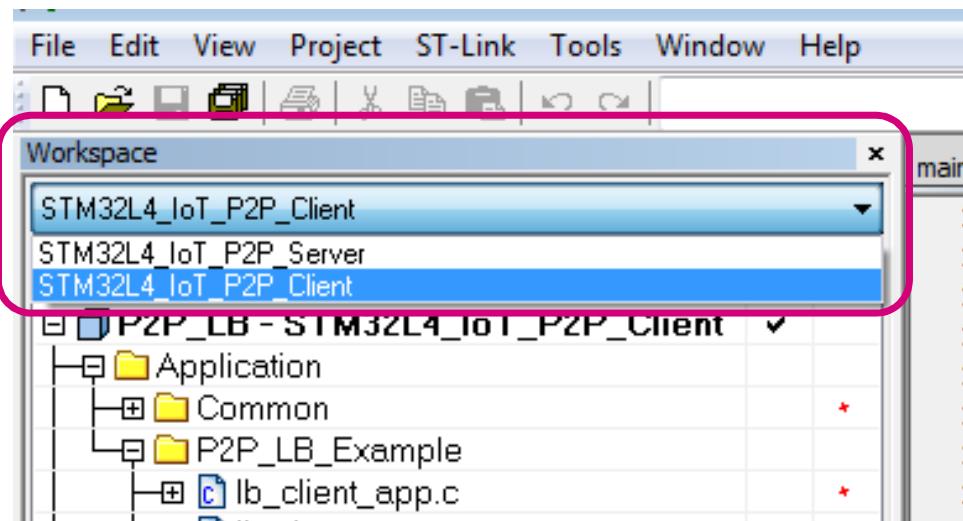


Open the P2P_LedButton Project

- We are going to configure the **P2P_LedButton** project with a unique BlueNRG MAC address for the client and server boards.
 - We will also specify the remote MAC address on the client board.
1. Close the previous IAR project.
 2. Double click on the **P2P_LB.eww** file located here:

C:\STM32L4_DK_IoT_Node_Seminar\Hands-on\Labs\Projects\B-L475E-IOT01\Applications\BLE\P2P_LedButton\EWARM

Select Your Configuration



- From the Workspace project configuration drop list select your configuration
 - `STM32L4_IoT_P2P_Client` if you are running the client firmware
 - `STM32L4_IoT_P2P_Server` if you are running the server firmware

BlueNRG Module configuration (1/4)

Expand the file tree:

1. Expand

P2P_LB - STM32L4_IoT_P2P_Client

or

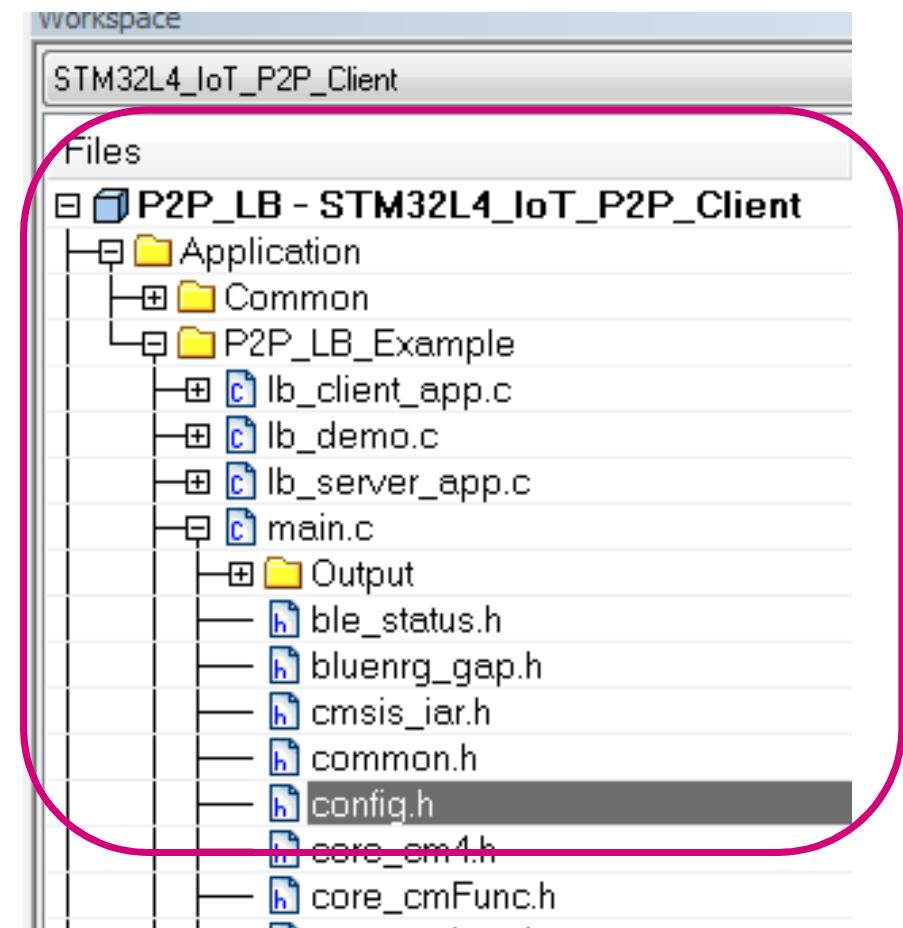
P2P_LB - STM32L4_IoT_P2P_Server

2. Expand **Application**

3. Expand **P2P_LB_Example**

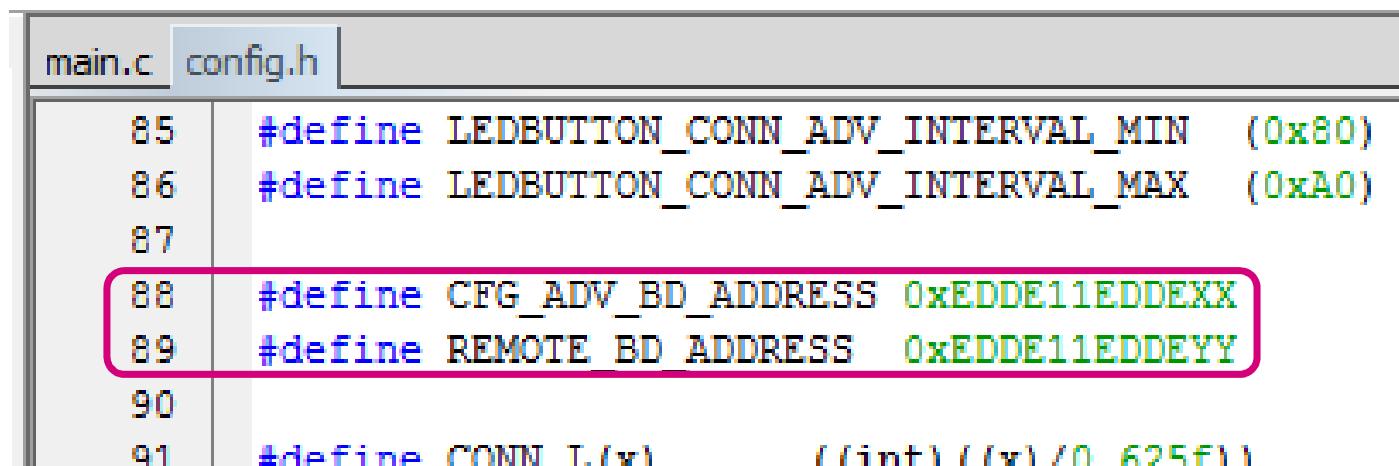
4. Expand **main.c**

5. Double-click **config.h**



MAC Address Configuration (Client)

189



The screenshot shows a code editor with two tabs: "main.c" and "config.h". The "config.h" tab is active, displaying the following code:

```
85 #define LEDBUTTON_CONN_ADV_INTERVAL_MIN (0x80)
86 #define LEDBUTTON_CONN_ADV_INTERVAL_MAX (0xA0)
87
88 #define CFG_ADV_BD_ADDRESS 0xEDDE11EDDEXX
89 #define REMOTE_BD_ADDRESS 0xEDDE11EDDEYY
90
91 #define CONN_T(x) ((int)(x / 0.625f))
```

Lines 88 and 89 are highlighted with a red rectangle.

1. Replace the '**XX**' in the **CFG_ADV_BD_ADDRESS** (line 88) with any number (2 characters).
2. Replace the '**YY**' in the **REMOTE_BD_ADDRESS** (line 89) with any other different number (Server) (2 characters).

MAC Address Configuration (Server)

190

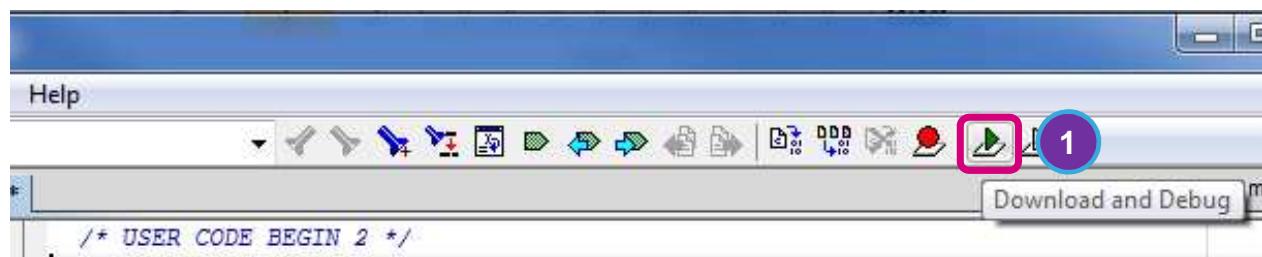
```
119 #if (LB_SERVER == 1)
120
121 #define LEDBUTTON_CONN_ADV_INTERVAL_MIN (0x1F4) /*< 312.5ms */
122 #define LEDBUTTON_CONN_ADV_INTERVAL_MAX (0x640) /*< 1000ms */
123
124
125 #define CFG_ADV_BD_ADDRESS 0xEDDE11EDDEXX
126
```

1. Replace the 'XX' in the **CFG_ADV_BD_ADDRESS** (line 125) with the same previous XX number (2 characters).

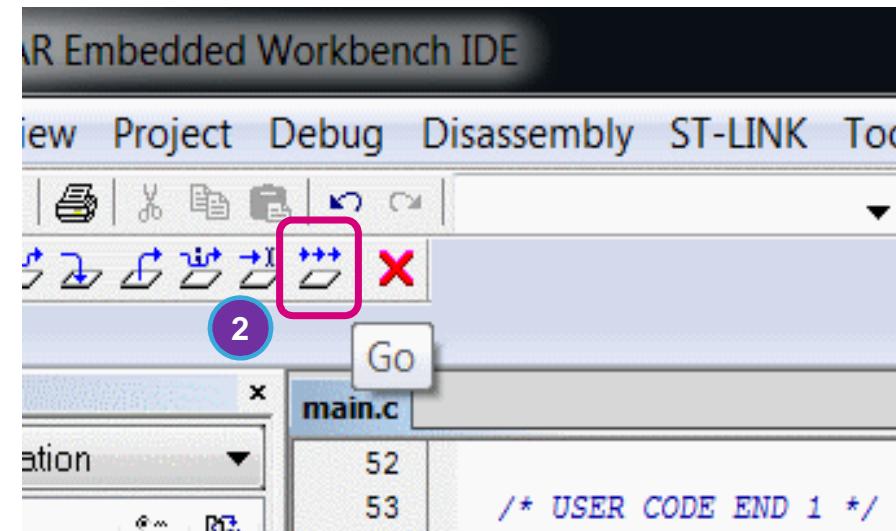
Load and Run

191

1. Click the GREEN ARROW to Build the Project, Download and start the debugger. (Ctrl + D)



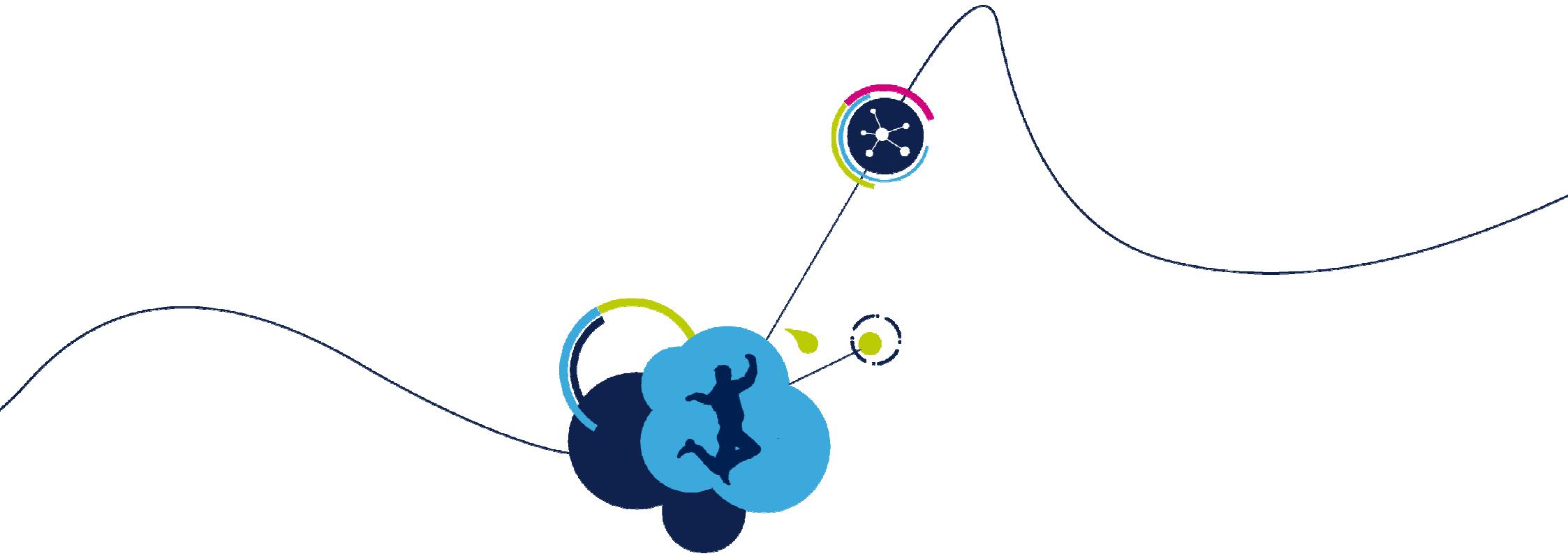
2. Click the triple-arrow GO button! (F5)



Run The Example

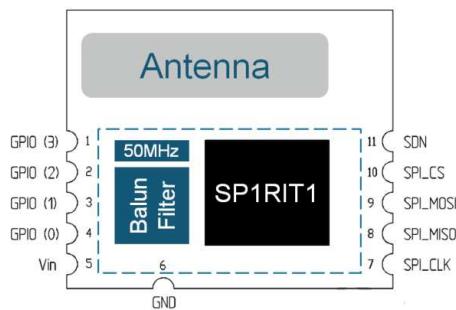
192

- After about 10 seconds the two boards will establish connection.
- Every time you press the blue button on your board, a message will get sent to the second board and toggle the LED.



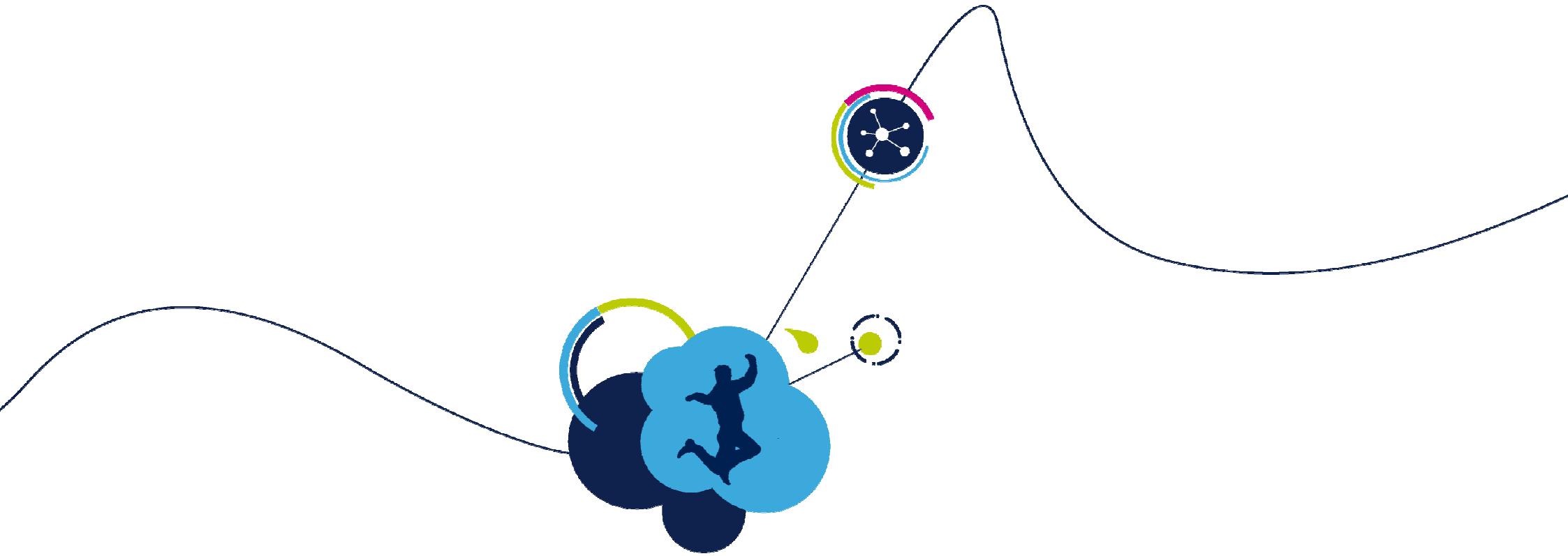
SPIRIT Sub-GHz Overview

SPSGRF-915 and 868 modules



SPIRIT1 Modules for 868MHz and 915MHz

- Module based on:
 - SPIRIT1 low power subGHz transceiver
 - BALF-SPI-01D3 balun and filter
 - Surface Mount antenna
- Tiny size: **13.5 x 11.5mm**
- 500Kbps data rate
- Temp. range from -40 °C to 85 °C
- Receiver sensitivity: **-118 dBm**
- Output power up to **+11.6 dBm**
- **RX: 9mA, Tx: 21mA @ +11dBm**
- **Shut Down: 2.5nA**
- **SPI host interface**
- **CE compliant**
- SPSGRF-915 is an FCC certified module (FCC ID: **S9NSPSGRF**)

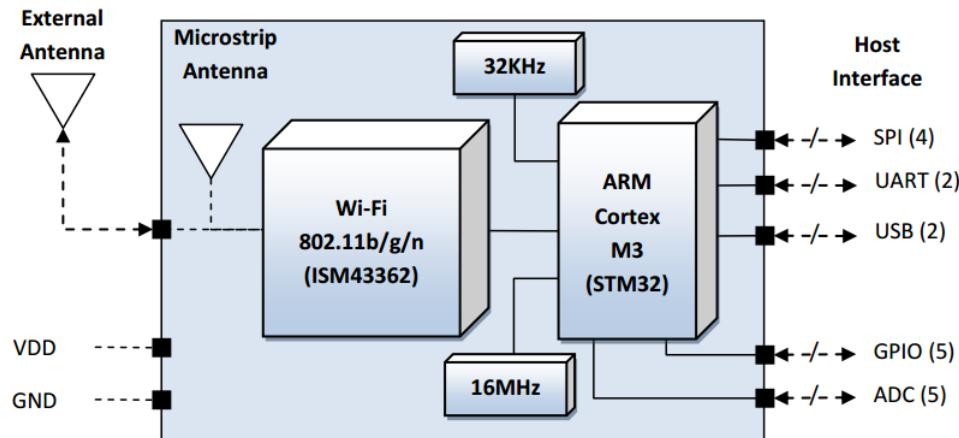


Wi-Fi Module Overview

ISM43362-M3G-L44-E/U

196

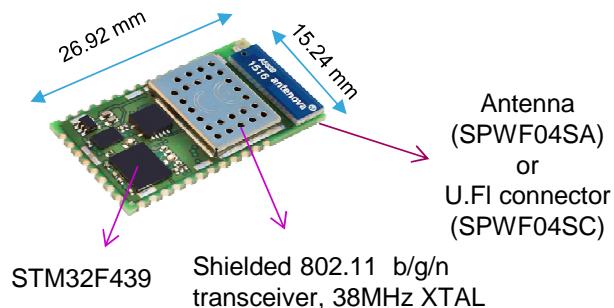
- The ISM43362-M3G-L44-E/U is an embedded 2.4 GHz Wi-Fi module from Inventek. The Wi-Fi module consists of a Broadcom BCM43362, an integrated antenna and an STM32F205 host processor with standard USB, SPI and UART interface capabilities.
- The Wi-Fi module has an integrated TCP/IP stack and requires only simple AT commands to establish connectivity for your wireless product.



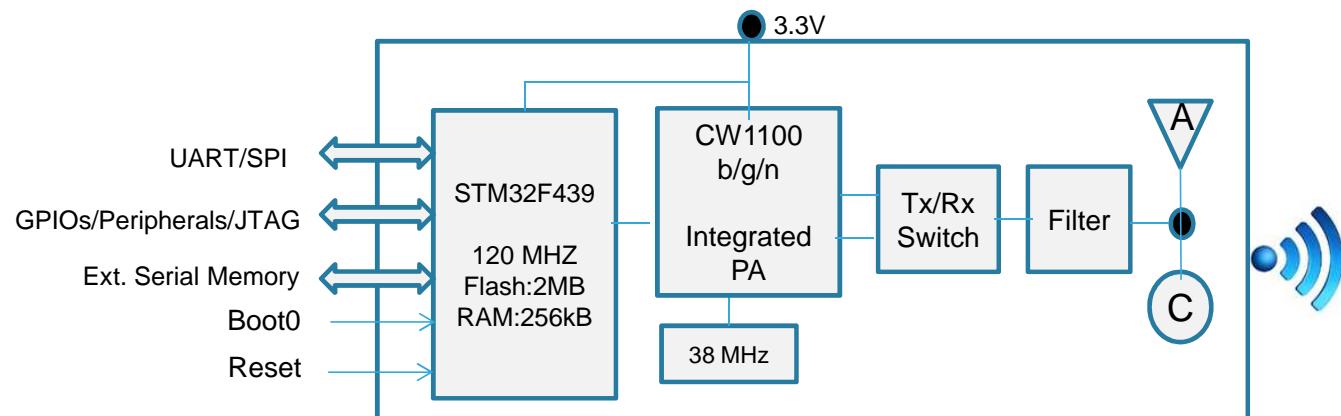
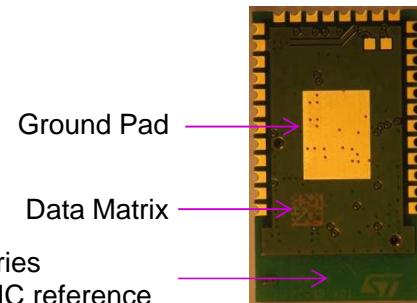
SPWF04S & X-NUCLEO-IDW04A1

197

Front

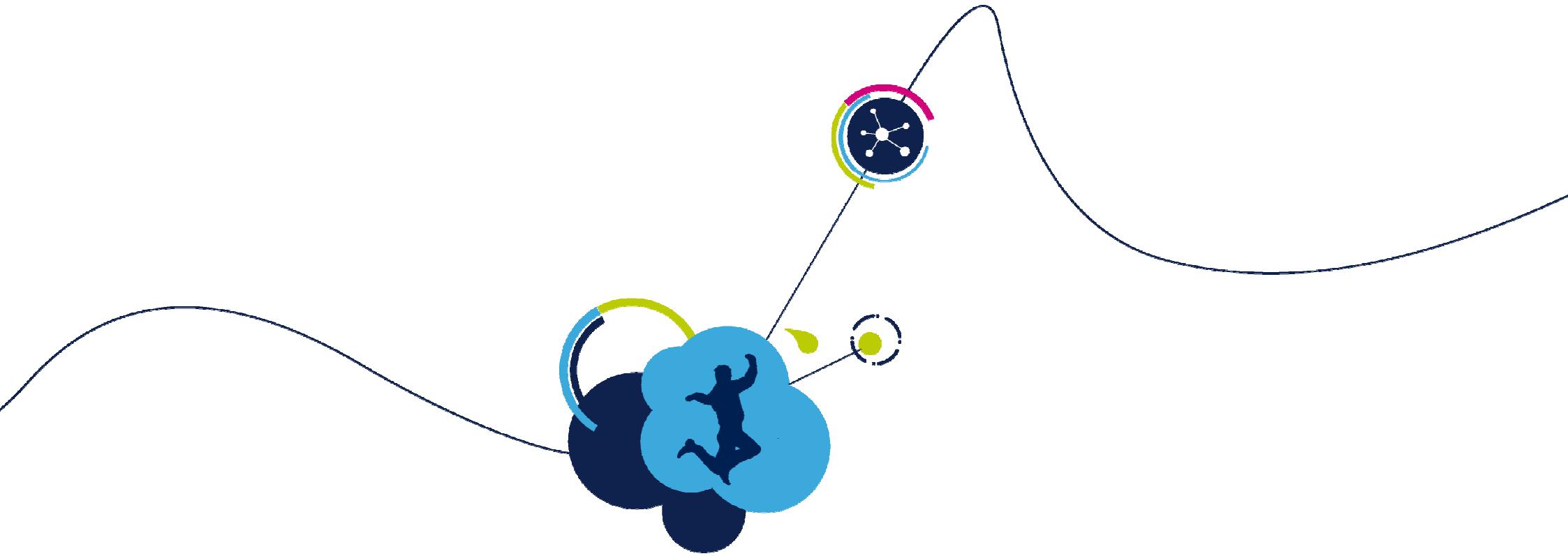


Bottom



- ✓ FW / FS upgrade possible through UART, JTAG, FOTA or SFOTA
- ✓ 3.3V IO interface : no external level shifter needed
- ✓ Dedicated IOs function : UART / SPI / µPython selection, Wi-Fi indication
- ✓ SPWF01 and SPWF04 are HW compatible





Lab 5 : Wi-Fi Client Server

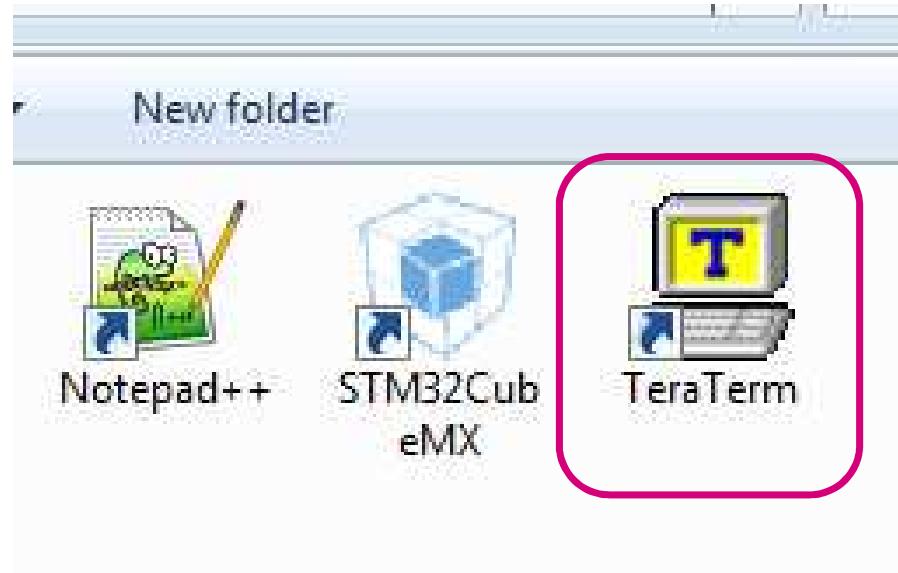
Lab Goal

199

- In this example we are going to demonstrate how to:
 - Connect to a Wi-Fi access point
 - Become the server and receive messages from a client (Cellphone)
- Enjoy sending messages

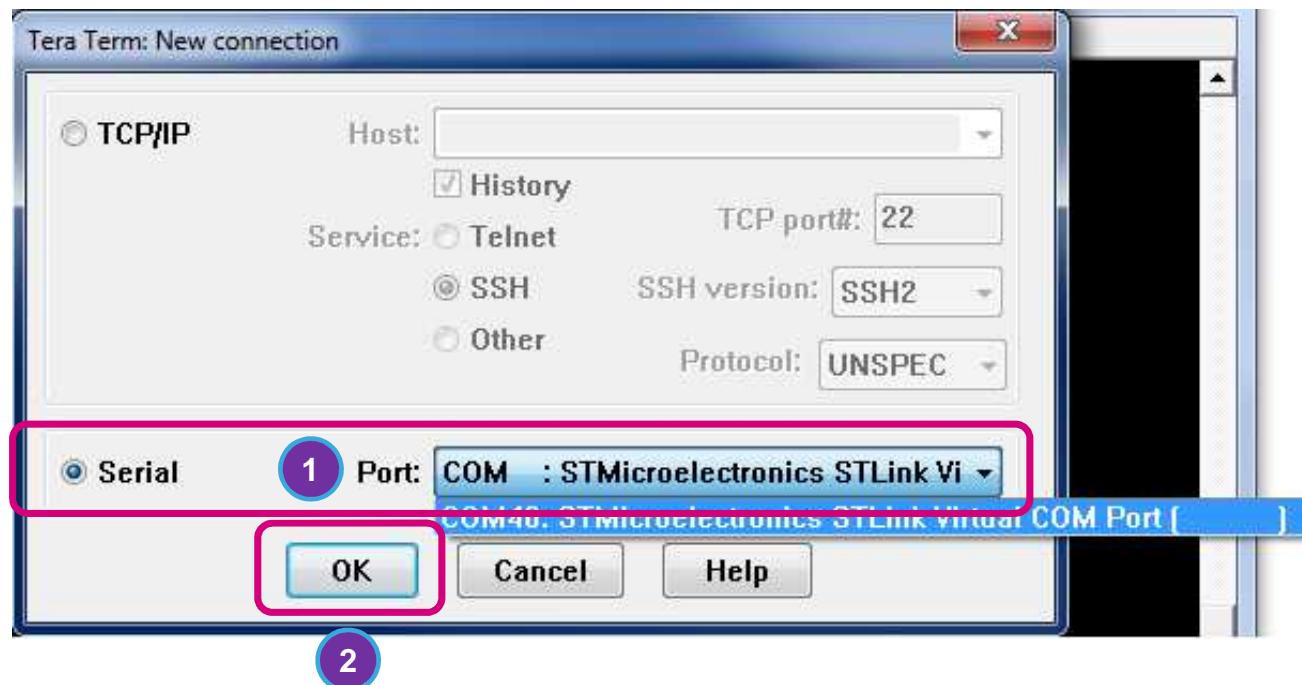
Open TeraTerm

200



- Double-click on the **TeraTerm** shortcut

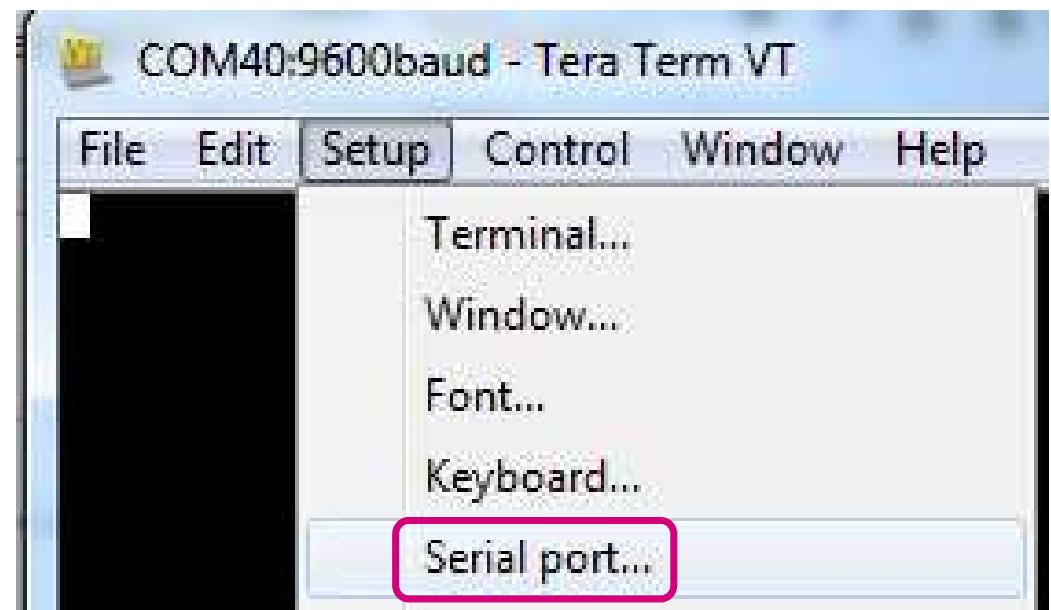
Tera Term Configuration 1/5



1. Select the **STMicroelectronics STLink Virtual COM Port**
2. Click **OK**

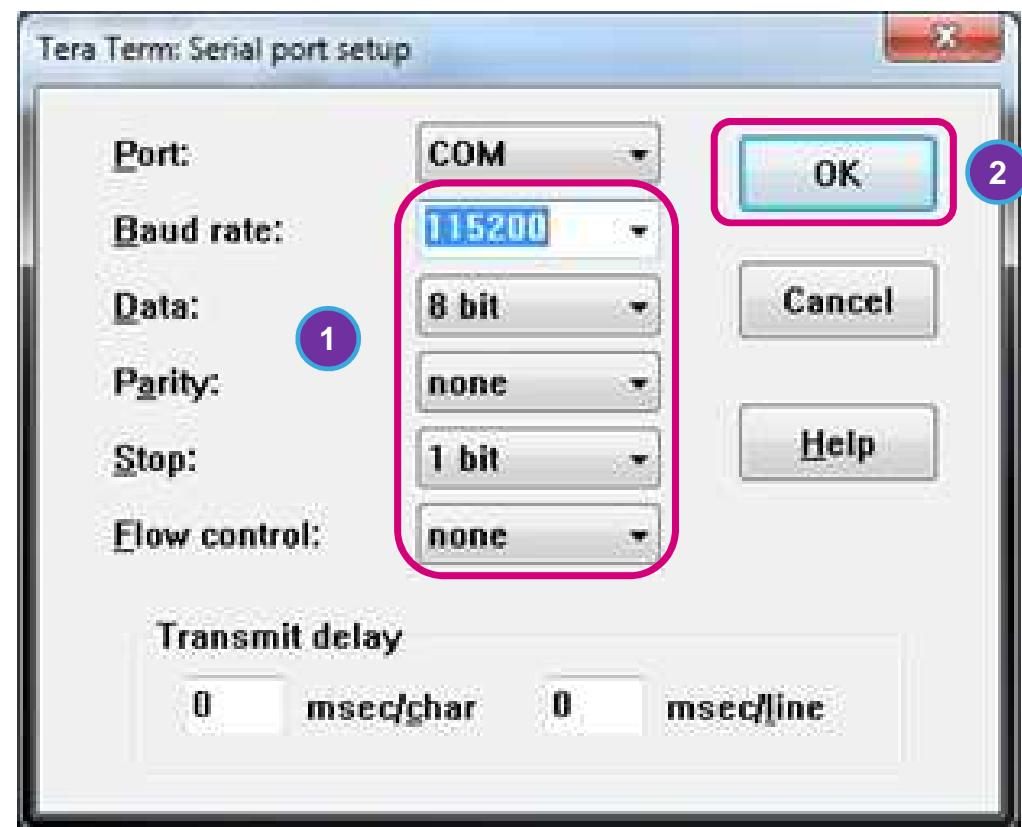
Tera Term Configuration 2/5

1. Click **Setup** -> **Serial port...**



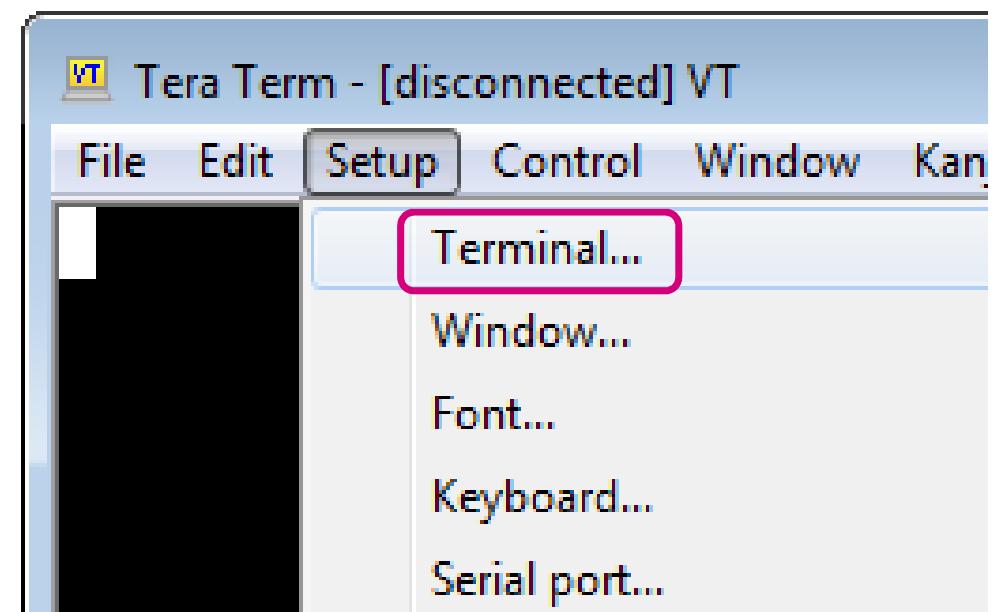
Tera Term Configuration 3/5

1. Set the following:
 - Baud rate : **115200**
 - Data : **8 bit**
 - Parity : **none**
 - Stop : **1 bit**
 - Flow control : **none**
2. Click **OK**



Tera Term Configuration 4/5

1. Click **Setup** -> **Terminal...**



Tera Term Configuration 5/5

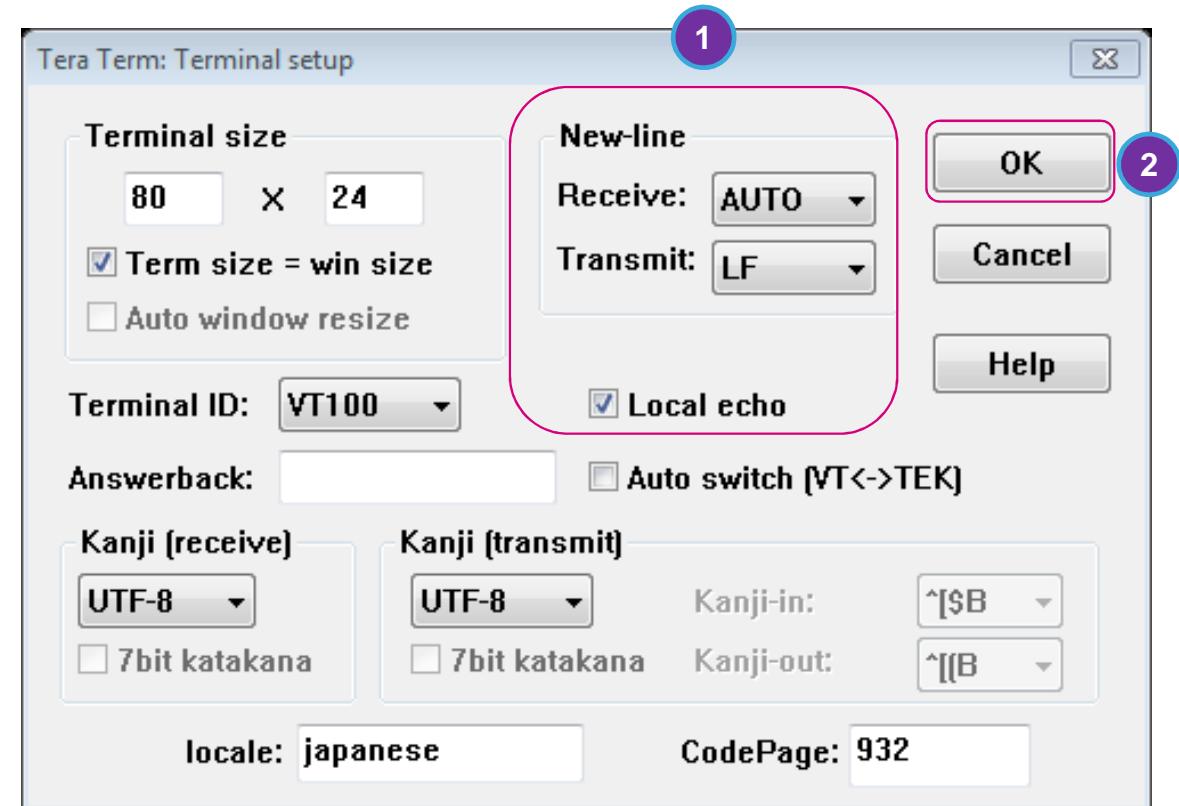
- Set the following:

New-Line Receive : **AUTO**

New-Line Transmit : **LF**

Enable **Local echo**

- Click **OK**



Open the Wi-Fi Project

- We are going to configure the `wifi` program to view debug information and to view data coming from a server.
1. Close the previous IAR project.
 2. Double click on the `Project.eww` file located here:

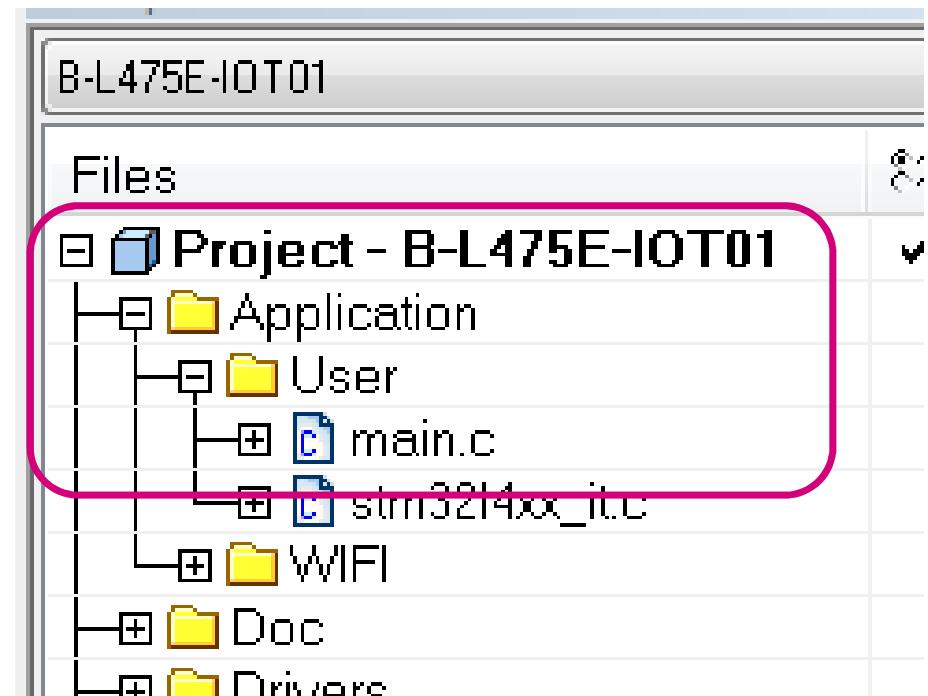
C:\STM32L4_DK_IoT_Node_Seminar\STM32CubeExpansion_WIFI1_v3.0.2\Projects\Multi\Applications\Server_Socket\EWARM\B
-L475E-IOT01

Setting Wi-Fi Credentials

207

Expand the file tree:

1. Expand **Project – B-L475E-IOT1**
2. Expand **Application**
3. Expand **User**
4. Double-click **main.c**



Setting Wi-Fi Credentials

The screenshot shows a code editor window titled "main.c *". The code is as follows:

```
46  */
47  /* Includes -----
48  #include "main.h"
49  /* Private defines -----
50  /* Update SSID and PASSWORD to match own Access point settings */
51  #define SSID      "stm32iot"
52  #define PASSWORD  "stm32iot"
53
```

Annotations are present in the code:

- Annotation 1 is a purple circle with the number 1, positioned over the line `#define SSID "stm32iot"`.
- Annotation 2 is a purple circle with the number 2, positioned over the line `#define PASSWORD "stm32iot"`.

1. Set the Wi-Fi SSID credential on line 51 to `stm32iot`
2. Set the Wi-Fi password credential on line 52 to `stm32iot`

Setting the Server Address

209

```
main.c *  
61  #if defined (TERMINAL_USE)  
62  extern UART_HandleTypeDef hDiscoUart;  
63  #endif /* TERMINAL USE */  
64  uint8_t RemoteIP[] = {aaa,bbb,ccc,ddd};  
65  uint8_t RxData [500];  
66  char* modulename;  
67  uint8_t TxData[] = "STM32 : Hello!\n";  
68  uint16_t RxLen;
```

- On line 64, set the server IP address to the presenters' host IP address.

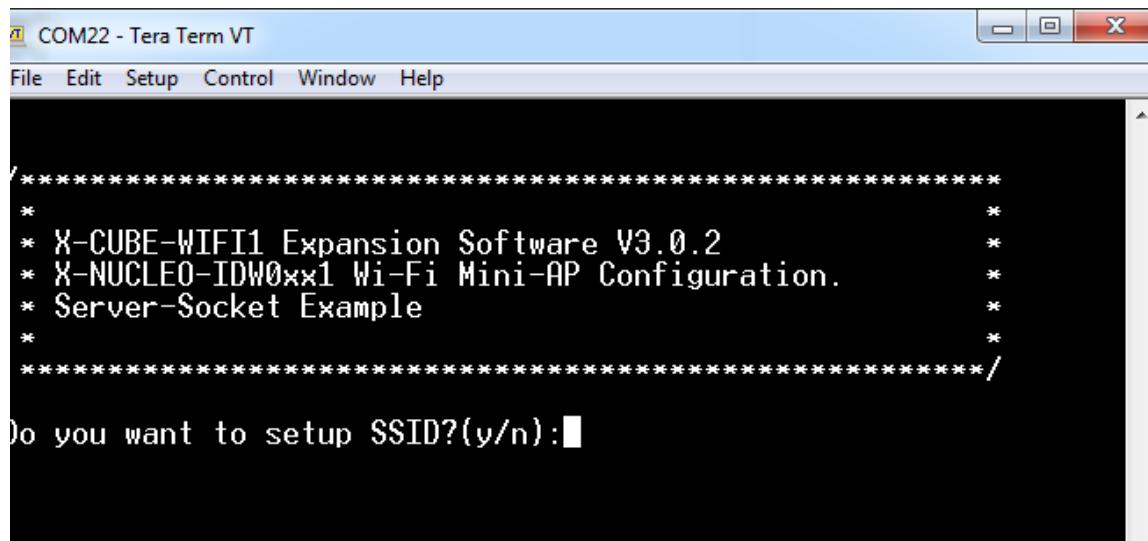
Setting the Server Address

210

The IP address is:

192.168.0.50

- Don't need to configure the SSID, as it's already set in our code



The screenshot shows a terminal window titled "COM22 - Tera Term VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", and "Help". The main text area displays the following output:

```
*****
* X-CUBE-WIFI1 Expansion Software V3.0.2
* X-NUCLEO-IDW0xx1 Wi-Fi Mini-AP Configuration.
* Server-Socket Example
*****
Do you want to setup SSID?(y/n):■
```

Output

212

- It'll start the configuration of the MiniAP and Open the socket at the given IP

COM22 - Tera Term VT

```
* X-CUBE-WIFI1 Expansion Software V3.0.2
* X-NUCLEO-IDW0xx1 Wi-Fi Mini-AP Configuration.
* Server-Socket Example
 ****
Do you want to setup SSID?(y/n):n
Module will connect with default settings.
 ****
* Configuration Complete
* Please make sure a server is listening at given hostname
 ****
Initializing the wifi module...
Build Configuration:
Nucleo: NUCLEO-L476RG
X-Nucleo: X-NUCLEO-IDW04A1
Interface: UART
```

COM22 - Tera Term VT

```
*****
Initializing the wifi module...
Build Configuration:
Nucleo: NUCLEO-L476RG
X-Nucleo: X-NUCLEO-IDW04A1
Interface: UART
End of Initialization..
Initializing complete.
>>setting up miniAP mode...
>>connected...
>>IP address is 192.168.0.50
>>mac addr is 00:80:E1:BD:D2:1C
>>WiFi server socket opening..
>>Server Socket Open OK
```

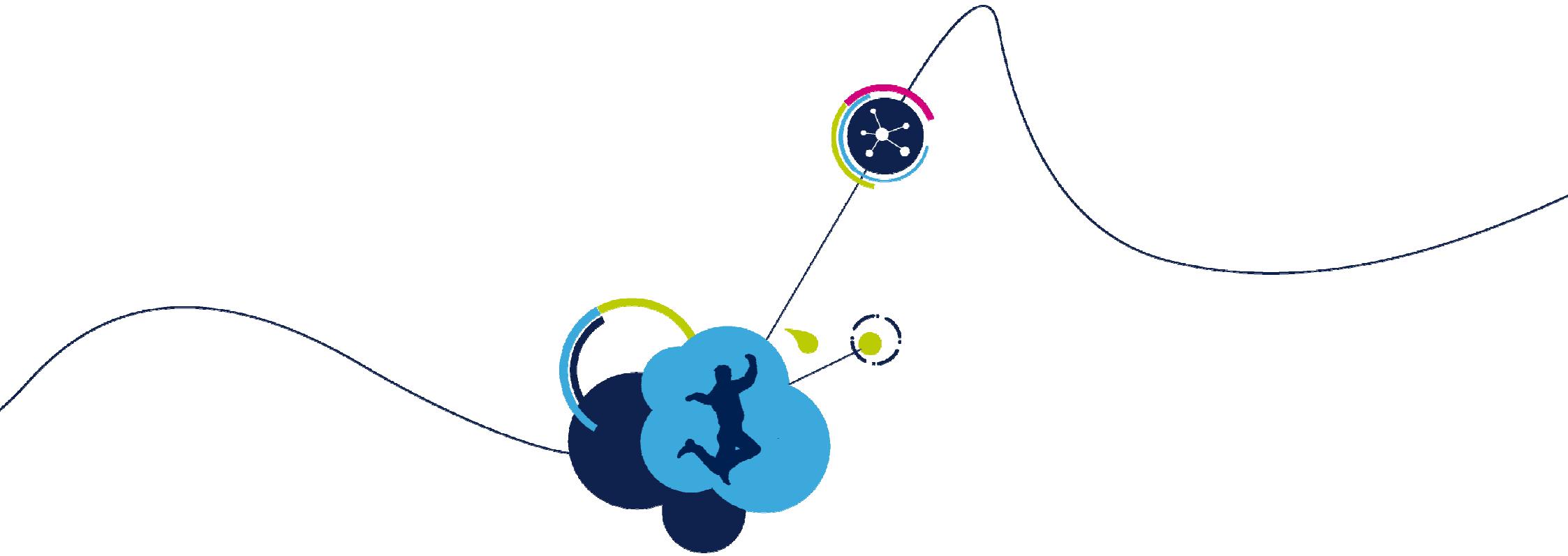
Output

213

- Using a simple App in my phone, I can enter the IP and connect as a client and send messages:

```
>>connected...
>>IP address is 192.168.0.50
>>mac addr is 00:80:E1:BD:D2:1C
>>WiFi server socket opening...
>>Server Socket Open OK
.
.
.
>>client joined callback...
.
.
.
User callback: Client joined...
.
.
.
Data Receive Callback...
Hola Argentina :)
socket ID: 0
msg size: 17
chunk size: 17
```





LoRa Overview

What is LoRa™ ?

215

1. A Sub-GHz wireless technology enabling low data rate communication over long distances

2. Targeting M2M and Internet of Things, IoT applications

3. LoRa™ technology is a solution providing a WAN capability, using a MAC protocol named LoRaWAN



Long range

- Greater than cellular
- Deep indoor coverage
- Star topology



Max lifetime

- Low power optimized
- **10 to 20-year** lifetime
- >10x vs cellular M2M



Multi-usage

- High capacity
- Multi-tenant
- Public network



Low cost

- Minimal infrastructure
- Low-cost end-node
- Open software



True location

- Indoor and outdoor
- Accurate



Bidirectional

- Bidirectional
- Scalable capacity
- Broadcast



Global mobility

- True mobility
- Seamless
- Roaming



Security

- Unique ID
- Application
- Network

LoRaWAN™ Devices Classes

216

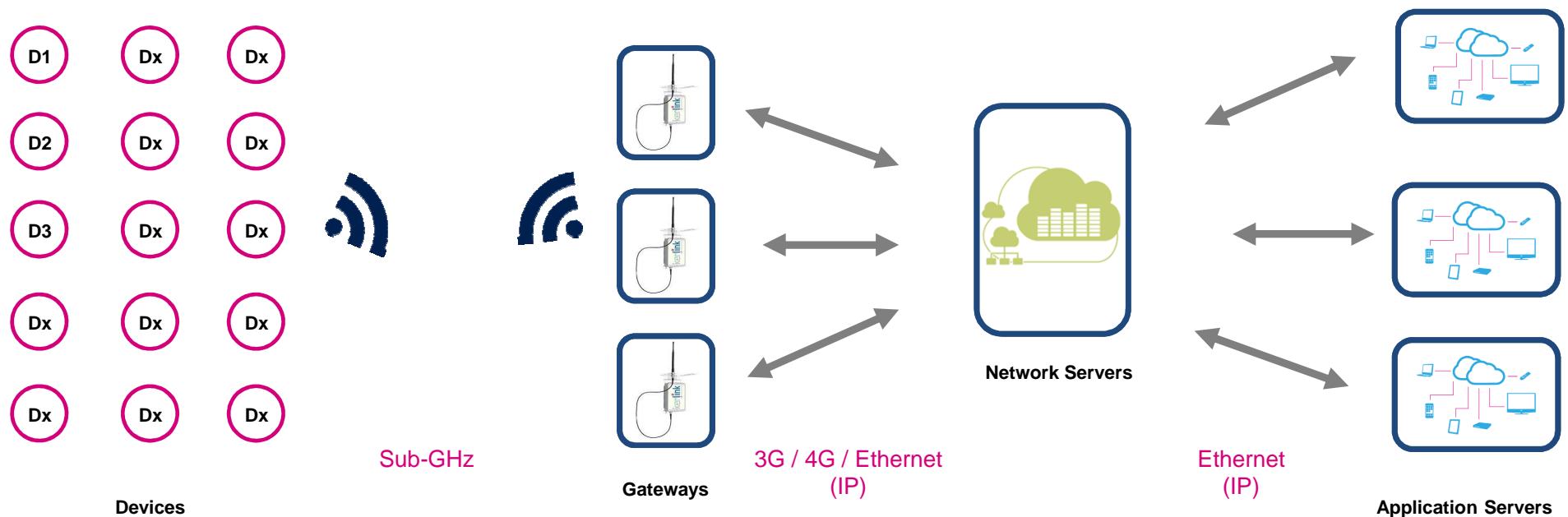
3 classes to cover all use cases

Class name	Intended usage	
A (“all”)	Battery powered sensors (or actuators with no latency constraint) Most energy efficient communication class. Must be supported by all devices.	Mainly uplink with two potential downlink slots after each uplink
B (“beacon”)	Battery powered actuators Energy efficient communication class for latency controlled downlink. Based on slotted communication synchronized with a network beacon.	Programmed downlink slots to allow control within certain latency limits
C (“continuous”)	Main powered actuators Devices which can afford to listen continuously. No latency for downlink communication.	Lowest latency command and control for less power critical devices

LoRa™ Network Protocol

217

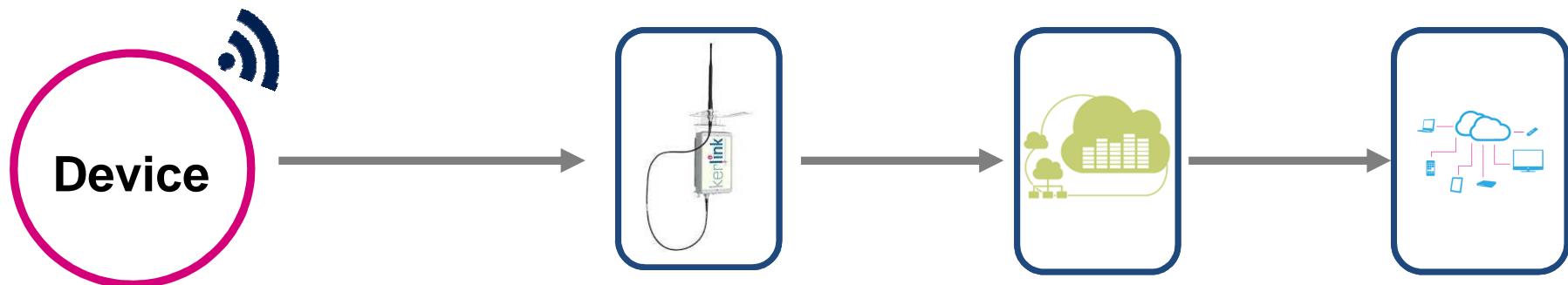
Network Topology Overview



LoRa™ Network Protocol

218

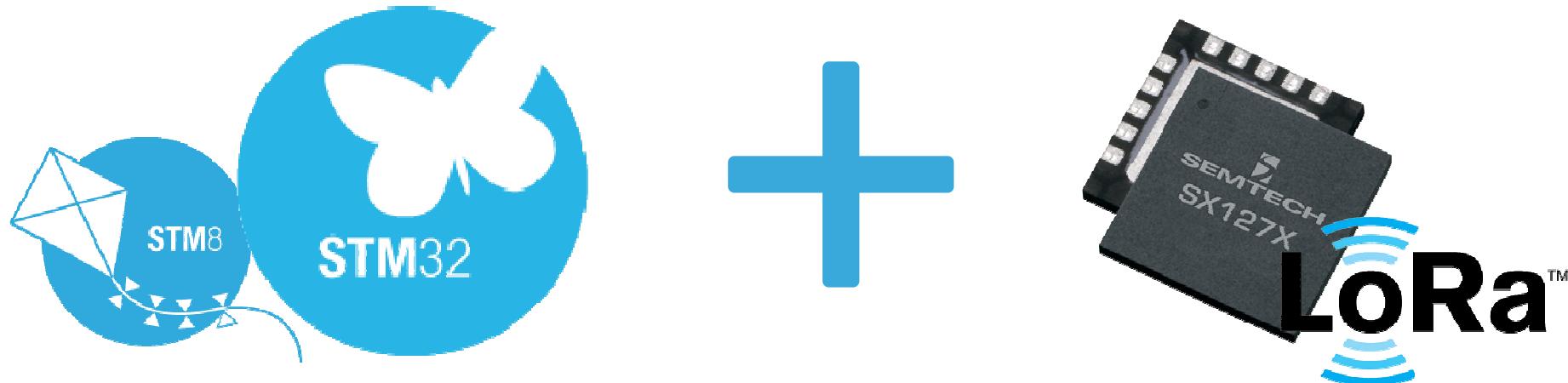
Solutions providers



The Perfect Match: STM32/STM8 + LoRaTM

219

More than 4000 possible combinations



STM32/STM8 are available in more than **1000 references**
LoRa SX127x offers **4 different lines**

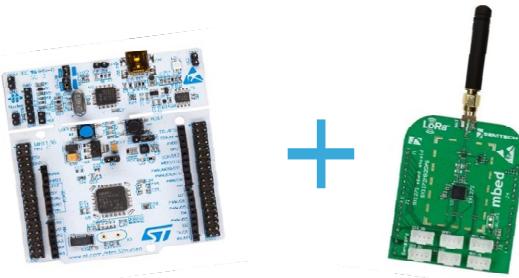
Let's Get Started

220

With a wide and existing ecosystem

(Click on the icon or link)

HW tools



P-NUCLEO-LRWAN1
B-L072-LRWAN1



Dev tools

[STM32CubeMX](#)

[ST-Link Utility](#)

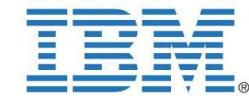
[Partners IDE](#)



[System Workbench for STM32](#)



LoRaWAN™ stack



ARM® mbed™
+ [demo code examples](#)

LoRaWan – Joining Process

2 activation methods exist to join an existing network

1. OTTA: Over the Air Activation = Over the air handshaking

- End-device send a Join Request (J.S) message to A.S including DevEUI, AppEUI, AppKey
 - DevEUI (Global unique end-device identifier) will allow the network to identify the device
 - AppEUI to identify which A.S the end-device is talking to
 - AppKey to identify which application on the A.S the end-device is referring to
- End device receives the Join Accept from the A.S and proceed with
 - Join Accept (authenticates and decrypt the Join Accept)
 - Extract and store DevAddr (the 32-bit device address)
 - Derives the 2 security: Network session Key (NwkSKey) and Application Session Key (AppSKey)

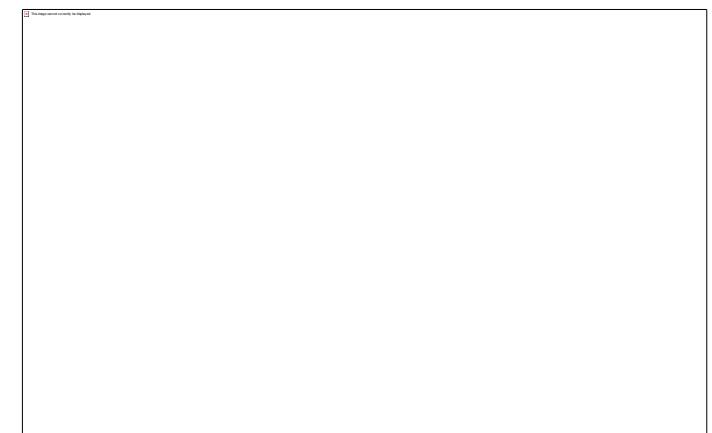
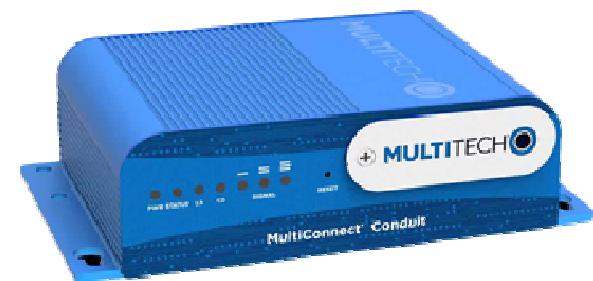
2. ABP: Activation By Personalization (no Over the air handshaking)

- The Device ready out-of-the-box to talk on the network
- The DevAddr, NwkSKey and AppSKey are programmed at factory level (no Over the air handshaking)

Setting up the Demo - Gateway

- Multitech - Gateway

- It is a LoRa Gateway
- Connects to LoRa Network Server through 3G-LTE, 3G, 2G and Ethernet
- Optimized for use in the 868 MHz **and** 915 MHz
- Ideal to demonstrate LoRa where no network available around
- <http://www.multitech.net/developer/products/multi-connect-conduit-platform>



Setting up the Demo – Network Server

223

The LORIOT login page displays three geographic regions for server selection:

- EUROPE & AFRICA:** EU1 (Frankfurt, Germany), AF1 (Johannesburg, South Africa)
- ASIA / PACIFIC:** AP1 (Singapore), AU1 (Sydney, Australia)
- AMERICAS:** US1 (California, USA), SA1 (Sao Paulo, Brazil), CN1 (Shenzhen, China)

Registration of a free community account is available on all servers.

The LORIOT Dashboard includes the following sections:

- account information:** Email: bruno.montanari@st.com, Name: Bruno Montanari, Logout button
- tier COMMUNITY NETWORK:** Welcome to LORIOT.io Community Network. You are now part of a world-wide ecosystem of LoRaWAN developers. Your devices can use any community gateway to reach our network. As a reward for sharing your gateway, we provide you one Free Network Application.
- COMMUNITY NETWORK features:**
 - No account expiration
 - Roaming among all community gateways
 - OpenLoRa Forum support
 - One Free Network Application
- News:** On Monday, 12th of June 2017, the gateway binaries and installers will be updated over the course of the day to version 2.4. Following the update, on any restart of your gateways, the LORIOT software on the gateways will be self-updated with the following changes:
 - Fixed potential overflow issues in scan and uplink code
 - Cisco V2 gateway integration
 - Haxiot gateway integration
 - Added periodic GPS information reporting
 - Log to syslog and /var/log when allowed
 - Static linking of libmpsse for Raspberry Pi
 - Improved installers with auto-restart scripts
- Gateways:** only last 10 shown (Location, Model, MAC, Version, Last data)
- Applications:** only last 10 shown (Name, AppID, Devices)

The Senet Portal dashboard shows:

- 1 Gateways, 1 Devices
- Avg TX Count: 0, Avg PSR: 0
- Devices & Gateways table with two entries:
 - 3431373257368300: Joined: 5/21/17 3:24 PM, Last Heard: 8 seconds ago, Notifier: TAGO - ✓
 - 00250C0001000268: Status: Registered - Active, Updated: 1 minutes ago, Last Heard: 18 seconds ago

The Tago dashboard includes:

- Zenelio:** Humidity (41.3%), Pressure (1015.7 hPa), Light Intensity Measurement (lux vs time)
- Raw data (payload from LoRa module):** Data table showing entries for humidity, pressure, and lux measurements.

Setting up the Demo – FW and HW

224

- Firmware Side

- Download the STM32CubeExpansion_LRWAN package
- Edit the LoraTxData function to send the information you want, in this case, we'll transmitt:

```
sprintf((char *)AppData->Buff, "STM32");
```

- Select the joining mode and the device information.
- This info can be seen in the serial port once compiled and downloaded:



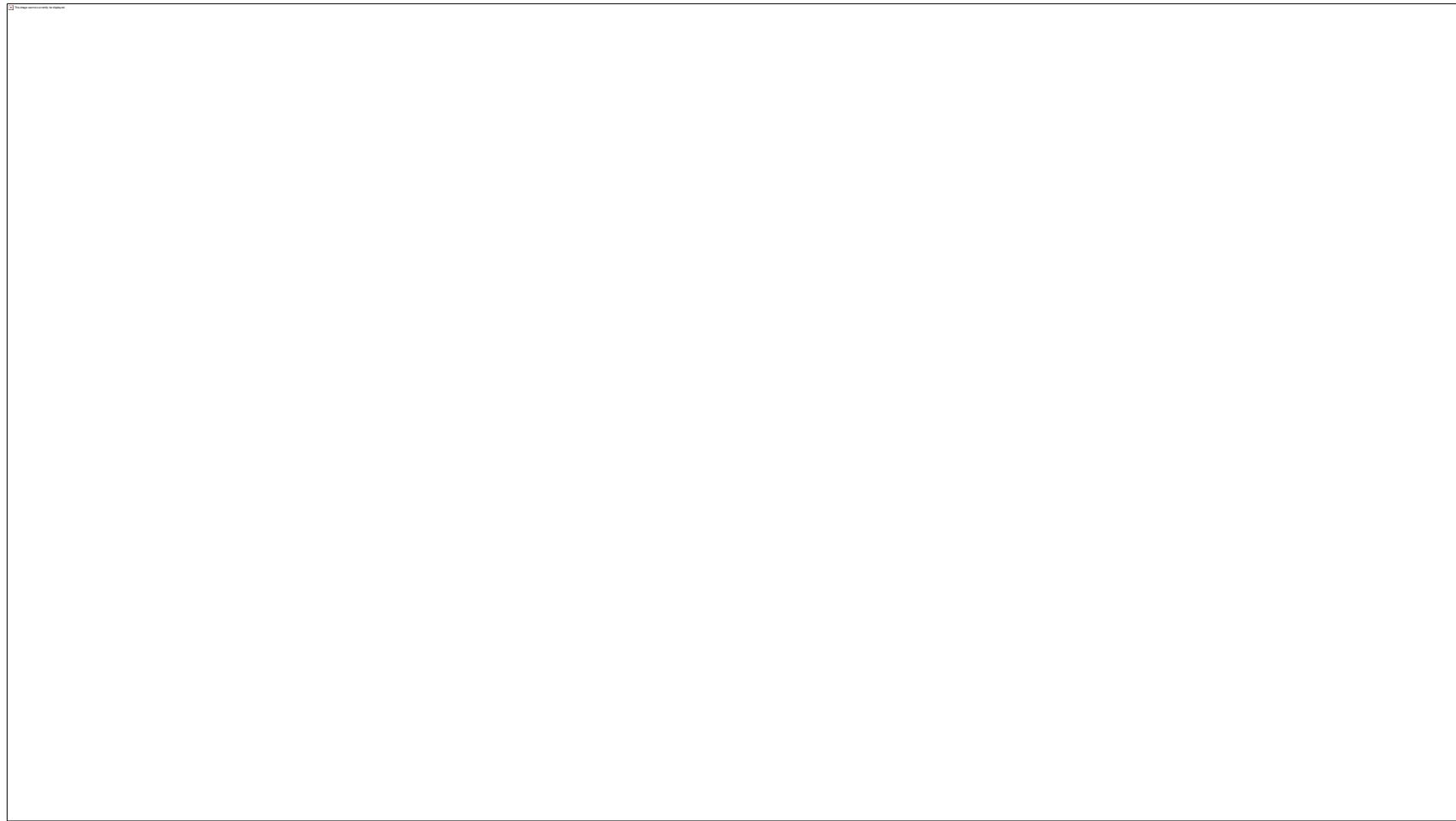
- Hardware Side

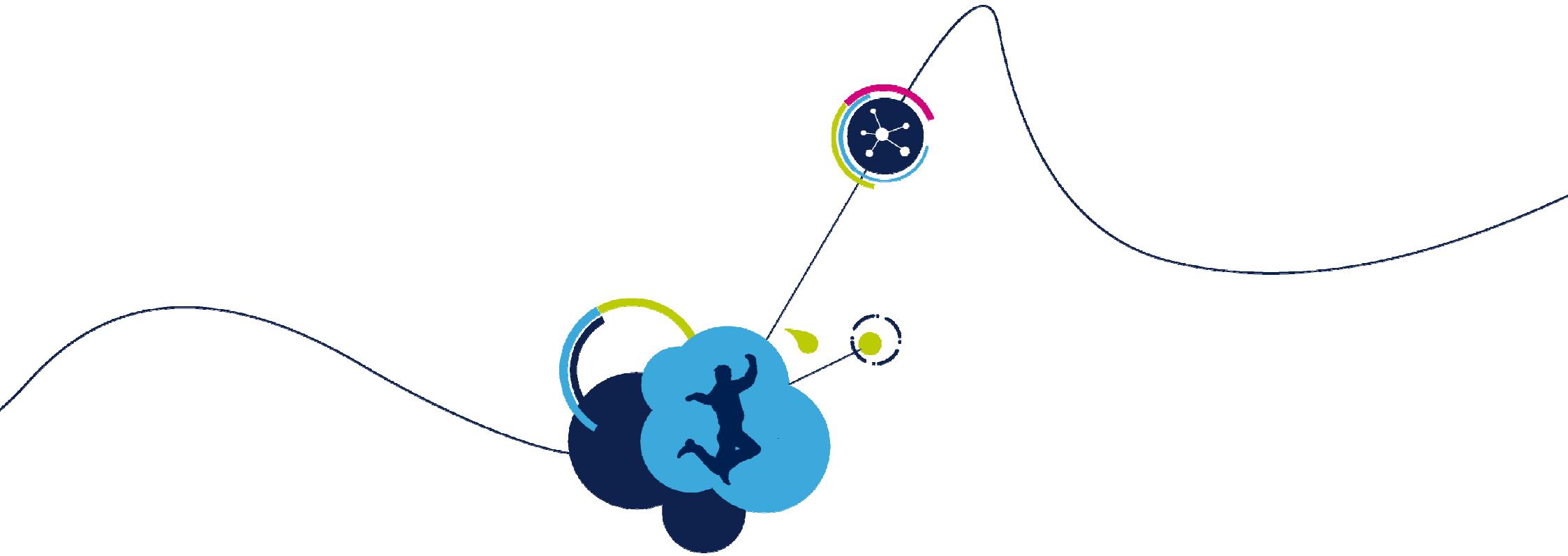


B-L072Z-LRWAN1

The Dash Board+JavaScript – Loriot

225

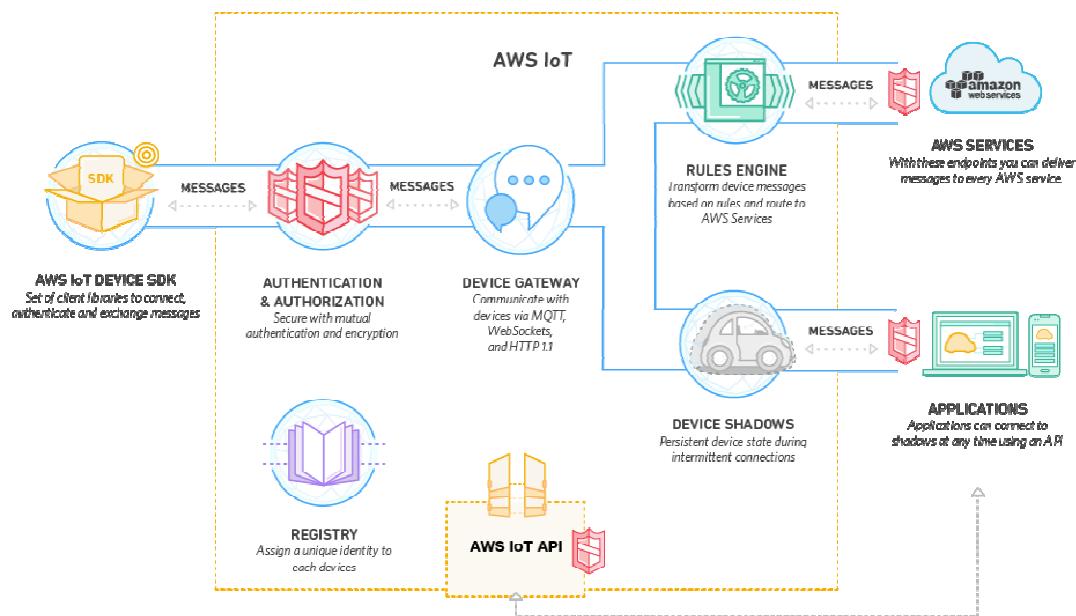




Amazon AWS IoT Overview

What is Amazon AWS IoT?

- The Amazon AWS IoT service enables secure, bidirectional communication between IoT devices and the cloud over MQTT, HTTP and WebSockets.
- AWS IoT devices are authenticated using TLS mutual authentication with X.509 certificates. Once a certificate is provisioned and activated it can be installed on a device. The device will then use that certificate for all requests to AWS MQTT.



What is TLS Mutual Authentication?

228

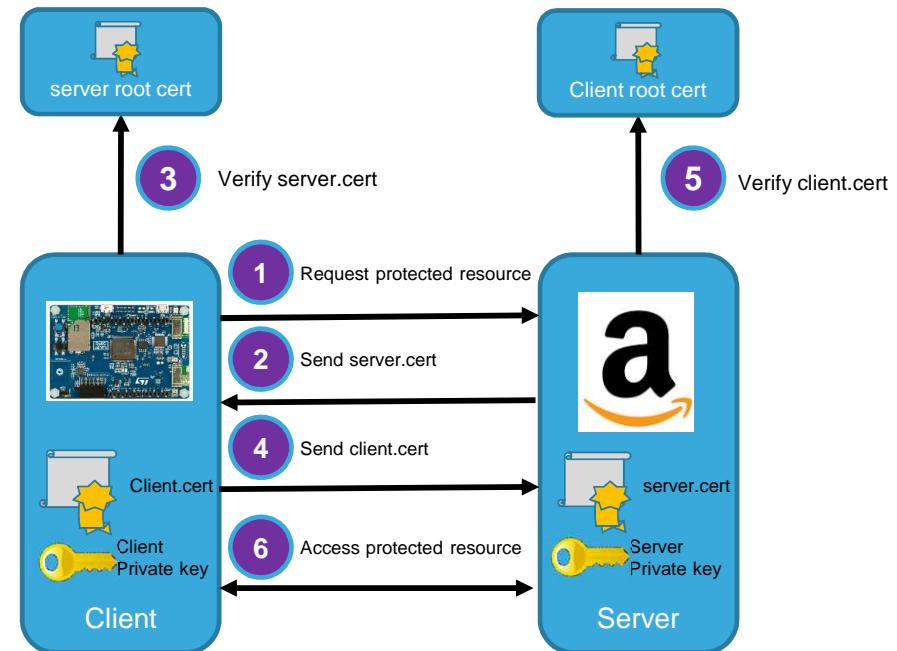
- TLS mutual authentication is used to establish trust between two parties.
- Each party verifies the certificate provided by the other.
- Certificate Authorities (CA) like Verisign are an important part of the mutual authentication.

TLS Mutual Authentication Flow

229

- TLS mutual authentication follows these steps:

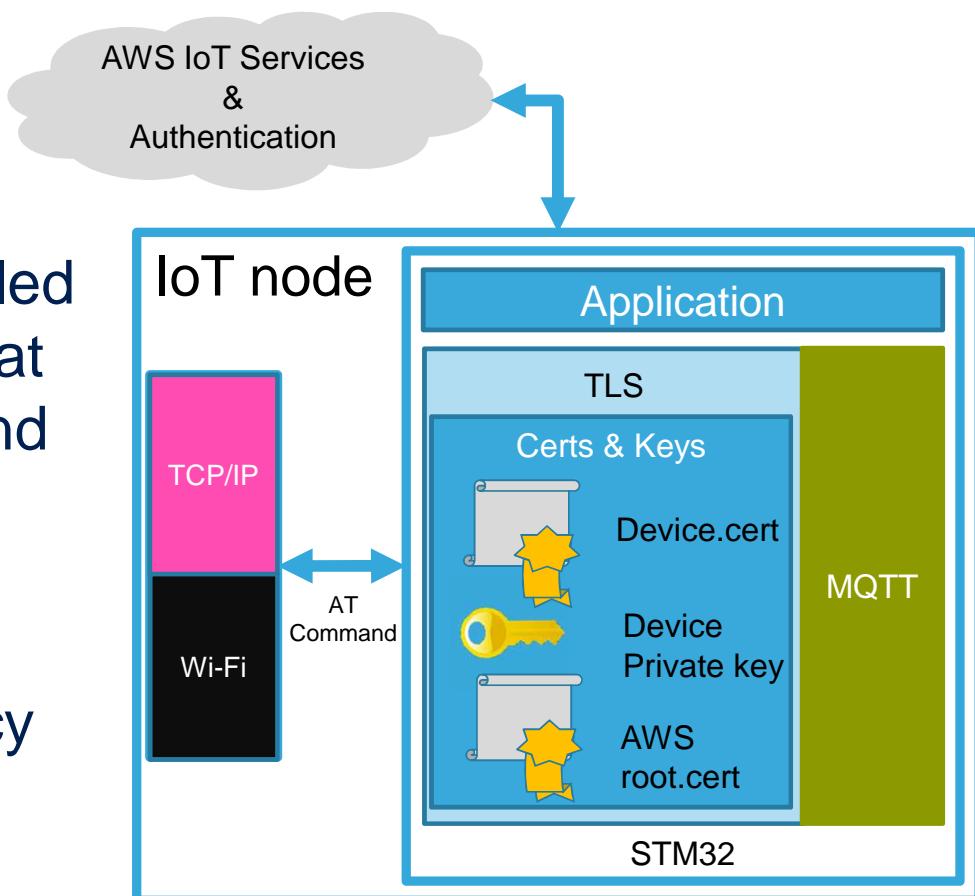
1. A client request access to a protected resource
2. The server presents its certificate to the client
3. The client verifies the server's certificate
4. The client sends it's certificate to the server
5. The server verifies the client's certificate
6. The server gives access to the protected resource requested by the client



AWS Security Overview

230

- AWS provides the device certificate and keys (AWS is the CA for the IoT device).
- The certificates and keys should be installed on the device. The device will then use that certificate and key to authenticate itself and send all requests to AWS IoT.
- To perform AWS IoT operations with your device, you must create an AWS IoT policy and attach it to your device certificate.



AWS IoT Policies

231

- AWS IoT policies are JSON* documents that authorize your device for performing AWS IoT operations.
- AWS IoT defines a set of policy actions describing the operations and resources for which you can grant or deny access. For example:
 - `iot:Connect` represents permission to connect to the AWS IoT message broker.
 - `iot:Subscribe` represents permission to subscribe to an MQTT topic or topic filter.
 - `iot:GetThingShadow` represents permission to get a thing shadow.

* JavaScript Object Notation

AWS IoT Thing Shadow

232

- A Thing Shadow (sometimes referred to as a Device Shadow) is a JSON document that is used to store and retrieve current state information for a Thing (device, app and so on).
- The Thing Shadow service maintains a persistent state of the device regardless of whether the device is connected to the Internet or not.

What Is JSON

233

- 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

234

- **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.

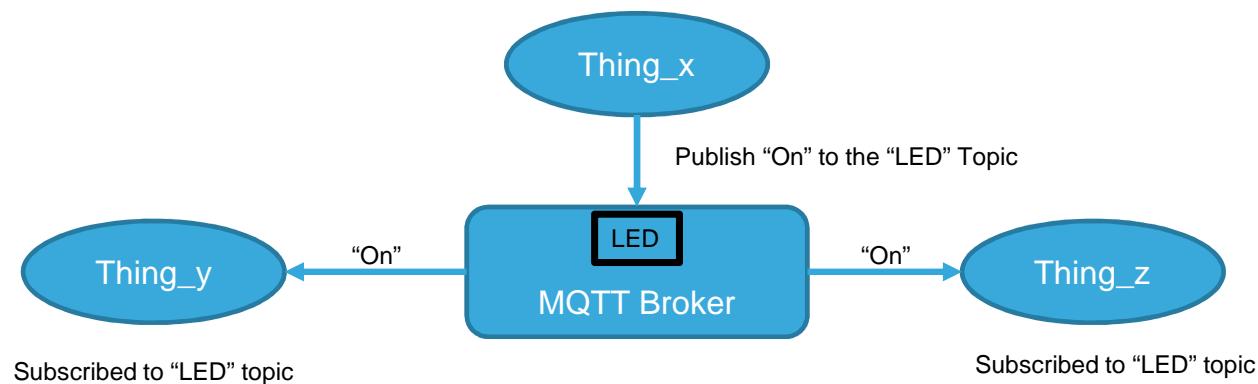
Source: <http://mqtt.org/>

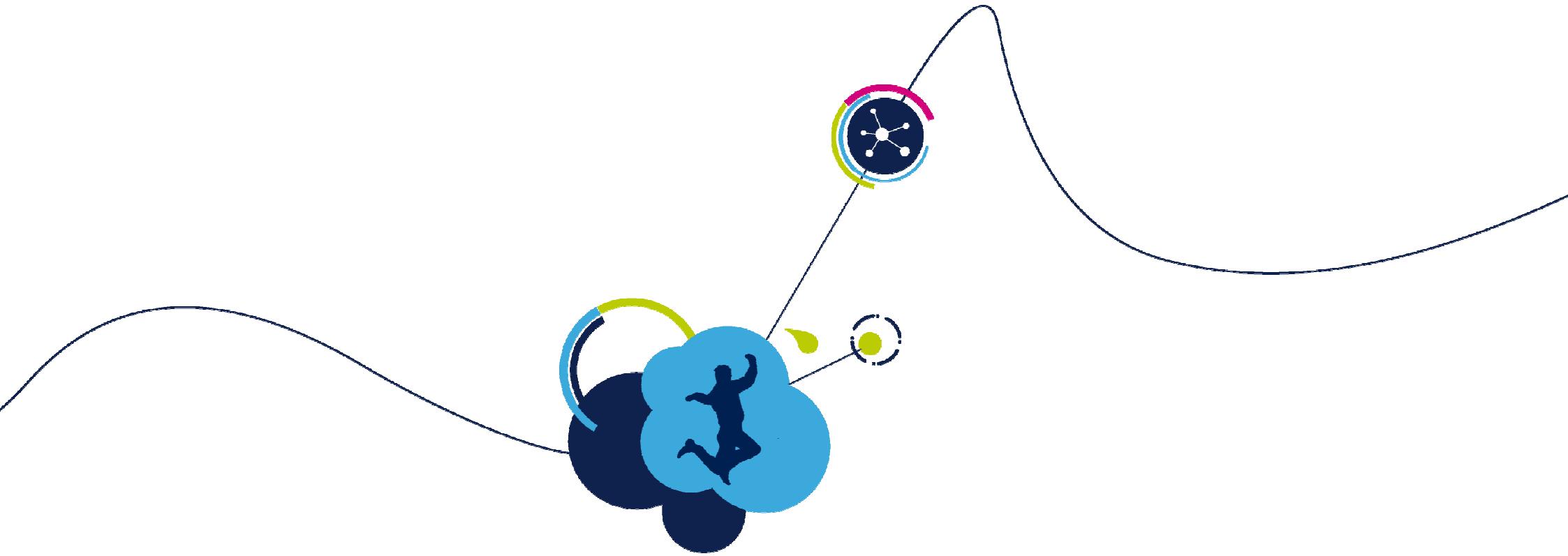


How MQTT Works

235

- MQTT uses a client/server model where every IoT device is a client and is connected to a server, called an MQTT broker (example AWS IoT).
- 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.





Lab 6 : Creating your Device ("Thing") on AWS

- In this lab you are going to:
 1. Create a device on AWS IoT
 2. Generate device certificate and keys
 3. Create a policy for your device
 4. Connect the certificate to the policy

Sign into AWS

238

- We have created an AWS account for you to use today.
- Open the following URL:
<https://stm32iot.signin.aws.amazon.com/console>
- User Name: seminarUser
- Password: stm32iot



Account:

User Name:

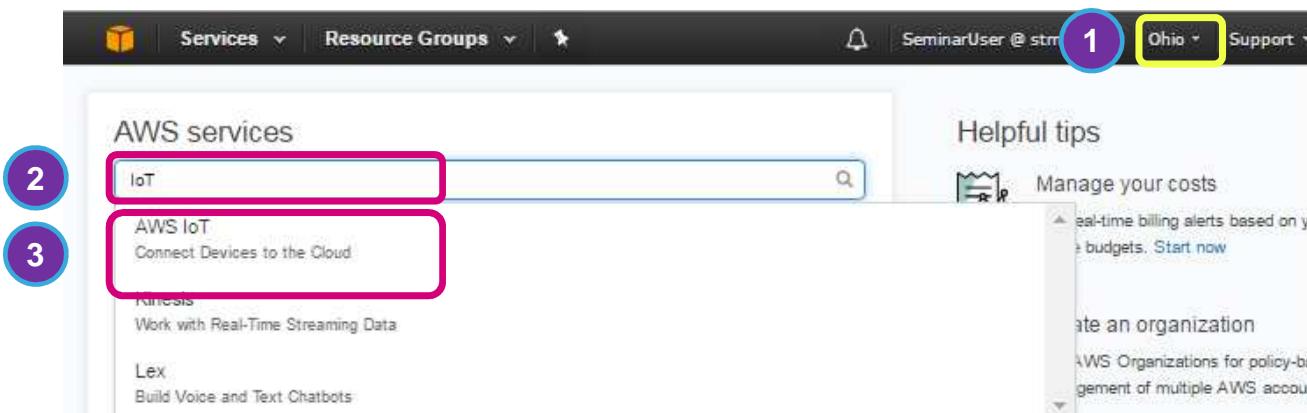
Password:

MFA users, enter your code on the next screen.

- Although we have created an AWS account for you to use today, you will need to sign up for your own account to continue to test AWS functionality.
- Sign up instructions can be found at: <https://aws.amazon.com>

Open AWS IoT Service

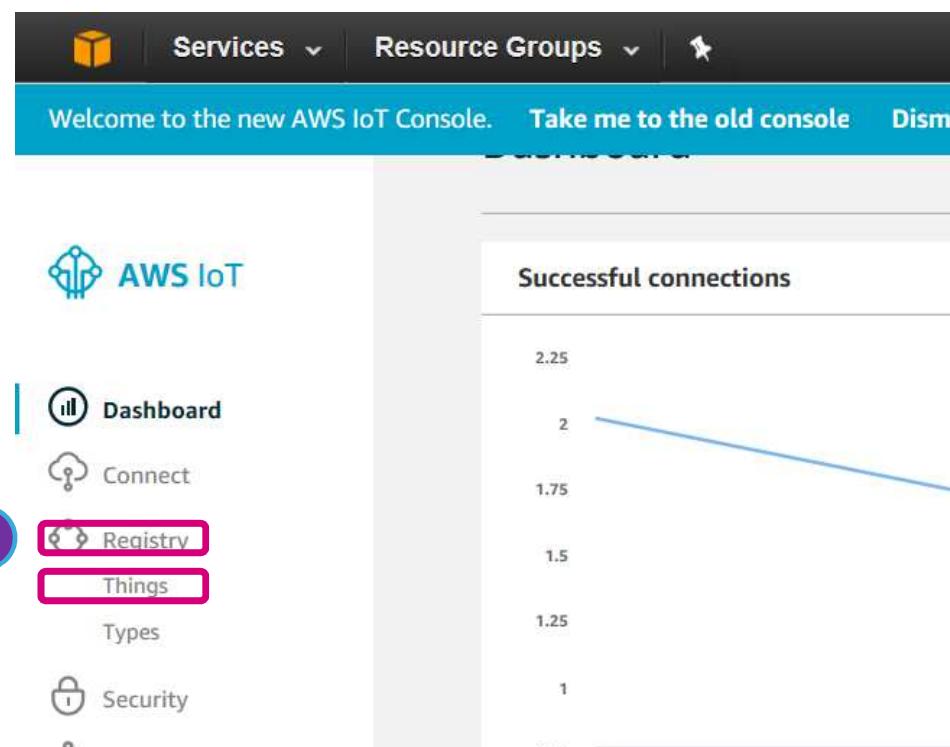
1. Make sure that you are connected to the Ohio server
2. Connect to the AWS IoT service by typing AWS IoT in the AWS services search bar.
3. Click on AWS IoT.



Create a Thing (1/3)

241

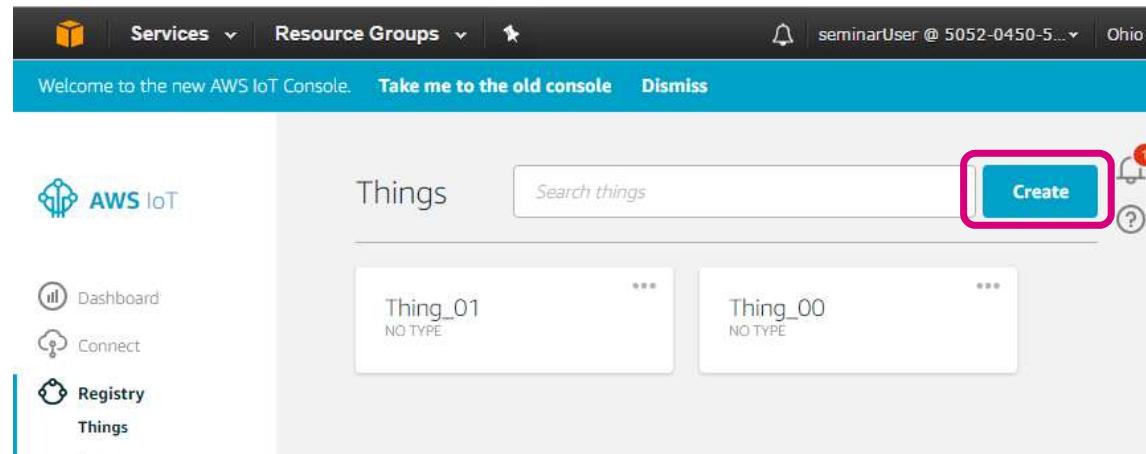
1. Start creating your device by selecting “Registry”.
2. Then select “Things”.



Create a Thing (2/3)

242

- Click the “Create” button on the top right.

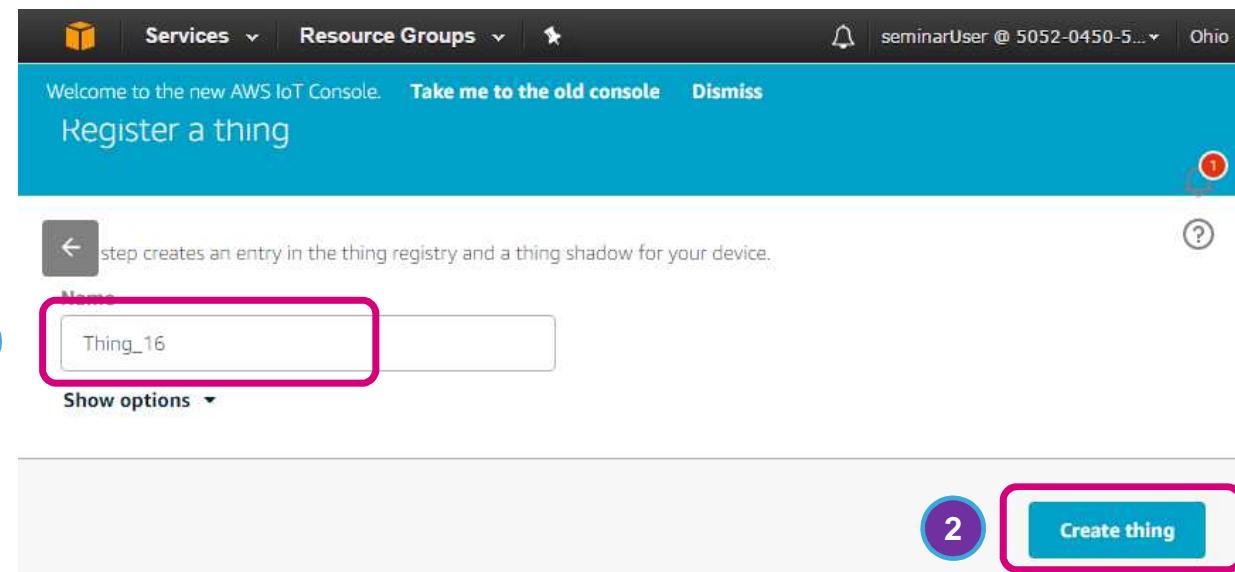


Create a Thing (3/3)

1. Name your “Thing_xx” with xx being your participant number (2 characters).

Example: Thing_08, Thing_16

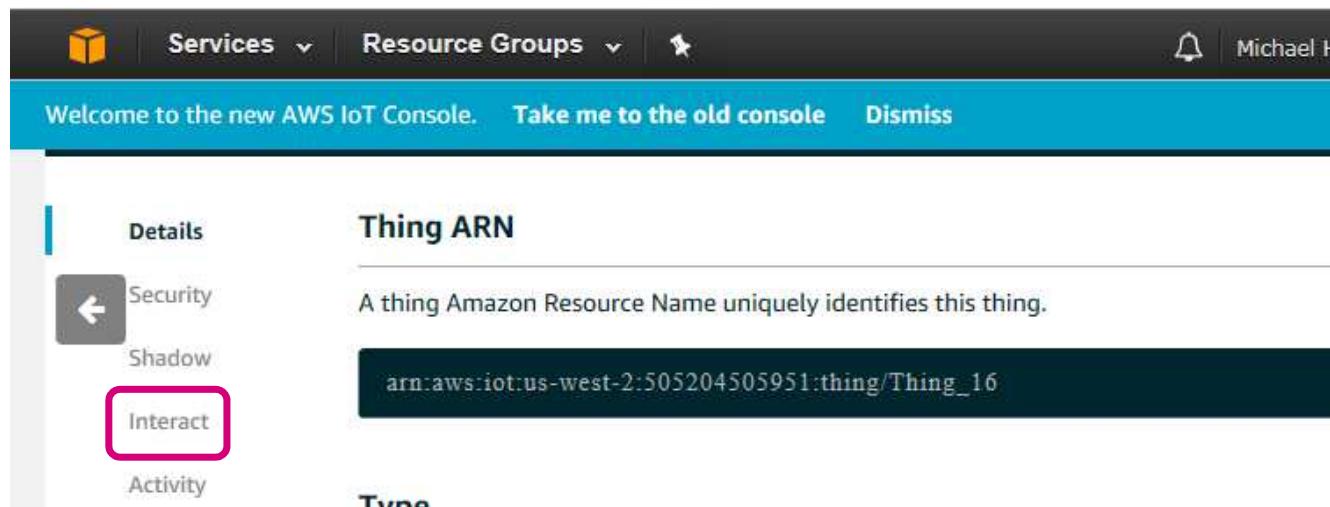
2. Click “Create thing”.



Create a Certificate (1/6)

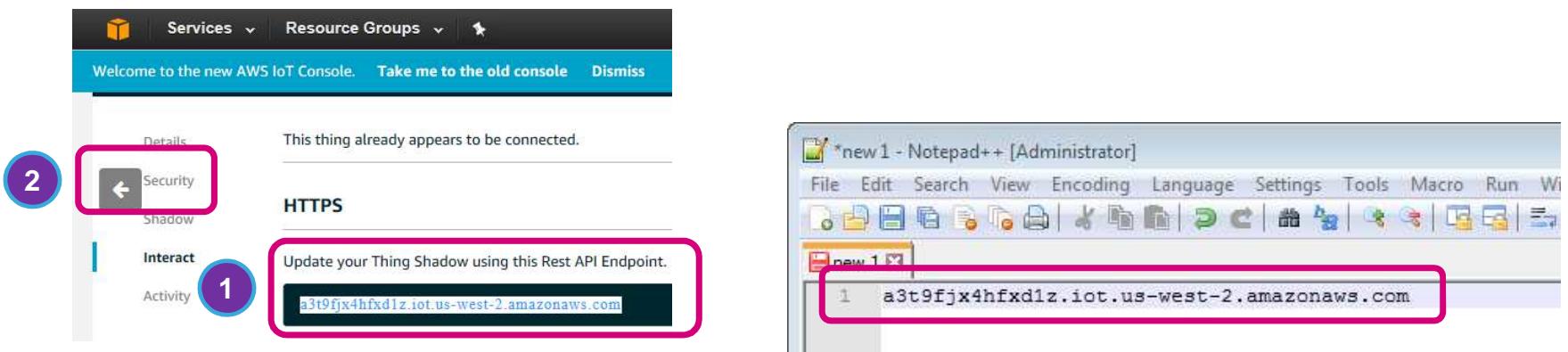
244

- Your thing has now been created. Select “Interact”.



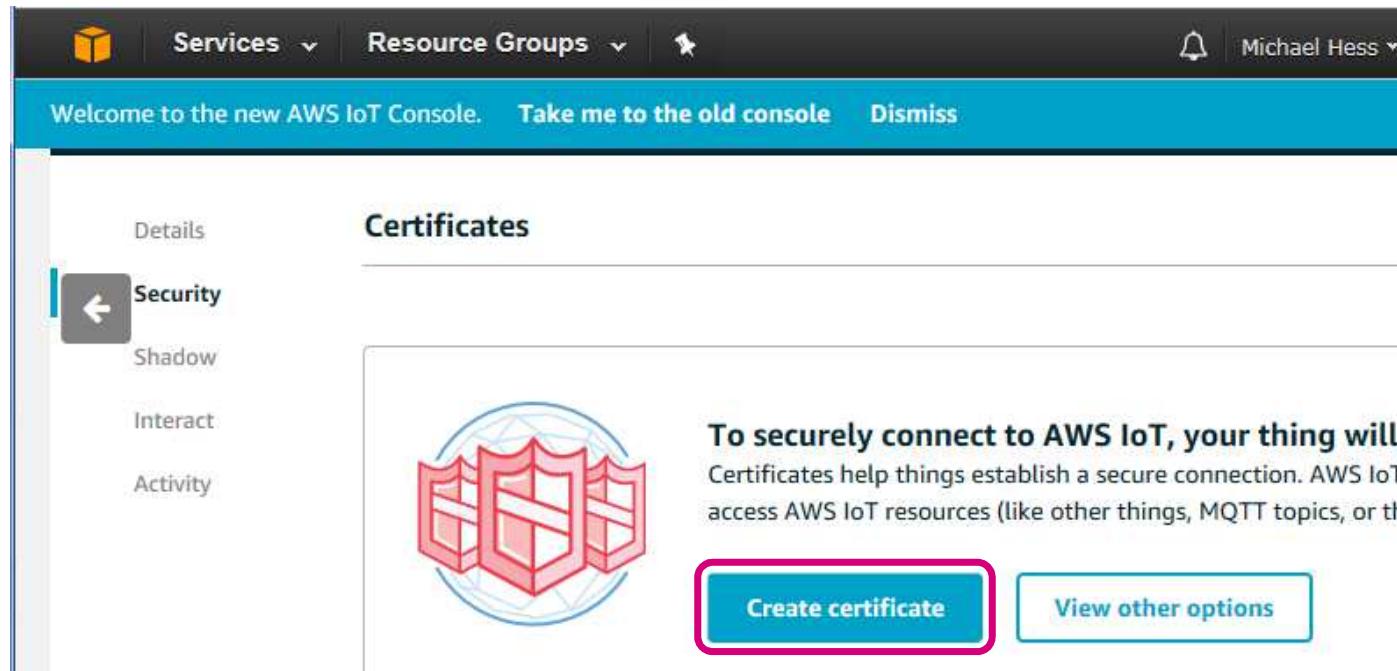
Create a Certificate (2/6)

1. Copy the Rest API Endpoint information to Notepad++.
2. Then, click “Security”



Create a Certificate (3/6)

- Click “Create certificate”.



Create a Certificate (4/6)

247

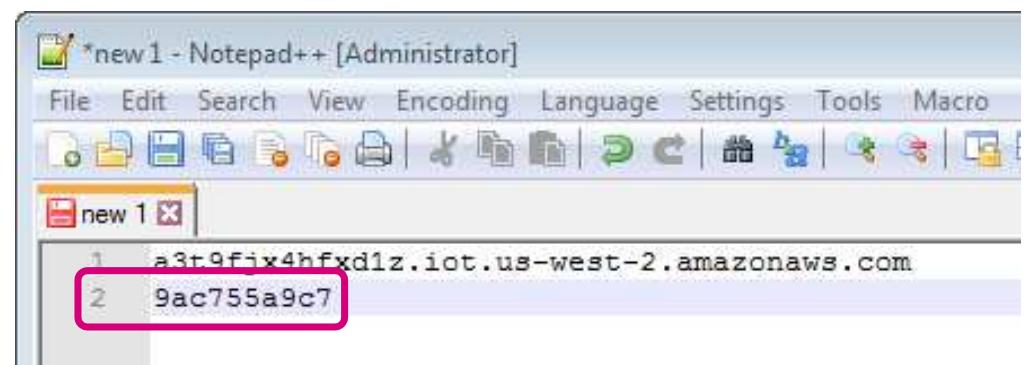
Welcome to the new AWS IoT Console. [Take me to the old console](#) Dismiss

Certificate created!

Download these files and save them in a safe place. Certificates can be retrieved at any time. Keys cannot be retrieved after you close this page.

In order to connect a device, you need to download the following:

A certificate for this thing	9ac755a9c7.cert.pem	Download
------------------------------	-------------------------------------	--------------------------



- Copy the certificate number to Notepad++ (just the number)

Create a Certificate (5/6)



Download these files and save them in a safe place. Certificates can be retrieved at any time. Keys cannot be retrieved after you close this page.

In order to connect a device, you need to download the following:

A certificate for this thing	9ac755a9c7.cert.pem	Download 1
A public key	9ac755a9c7.public.key	Download
A private key	9ac755a9c7.private.key	Download 2

You also need to download a root CA for AWS IoT from Symantec:

A root CA for AWS IoT [Download](#) 3

[Activate](#)

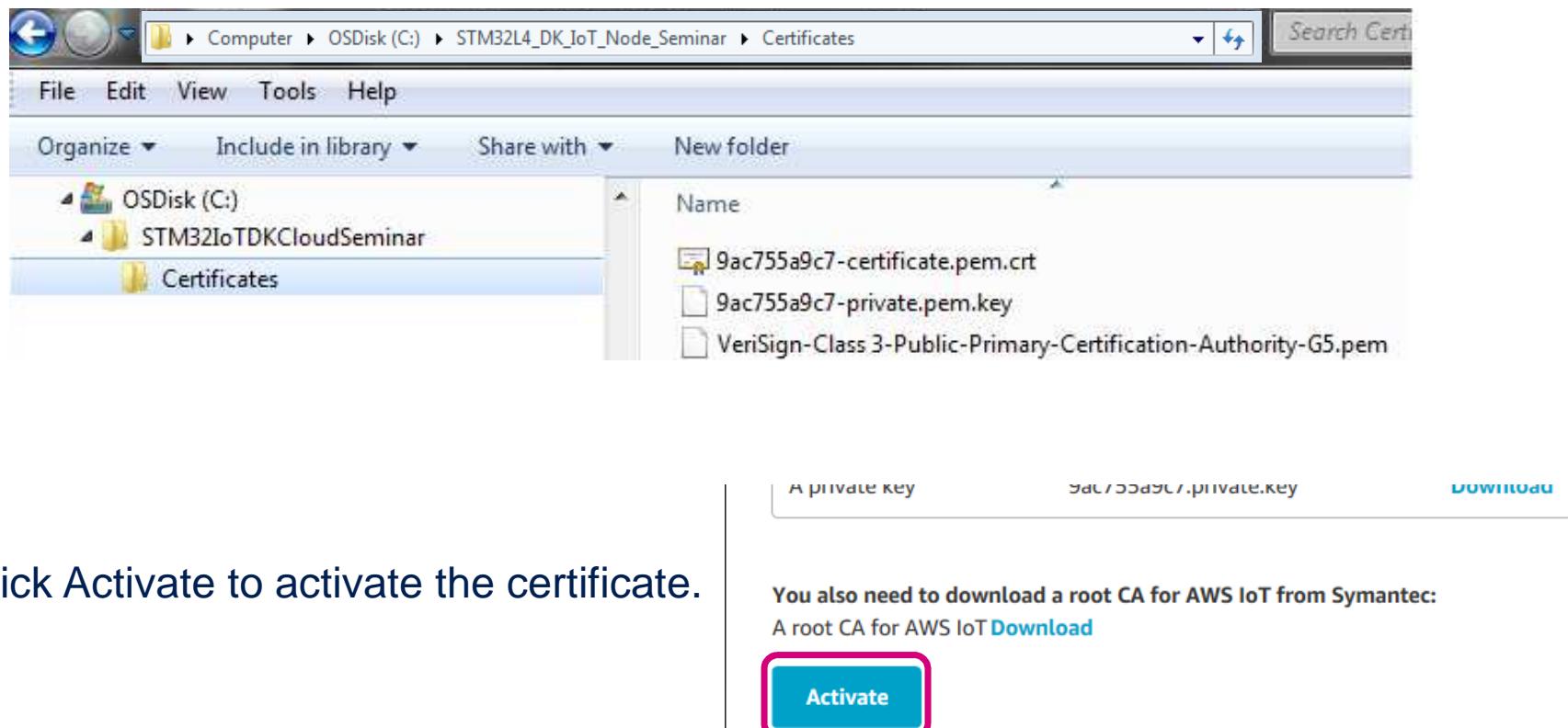
- Save the following files to:

C:\STM32L4_DK_IoT_Node_Seminar\Certificates

1. Your **Thing certificate** (xxxxx.cert.pem),
2. Your **Thing Private key** (xxxxx.private.key)
3. AWS IoT **Root CA**
(Right click on Download then save link as)

Create a Certificate (6/6)

- Your Certificate folder should look similar to this:



Create a Policy (1/4)

250

- Attach a policy to your device by clicking “Attach a policy”

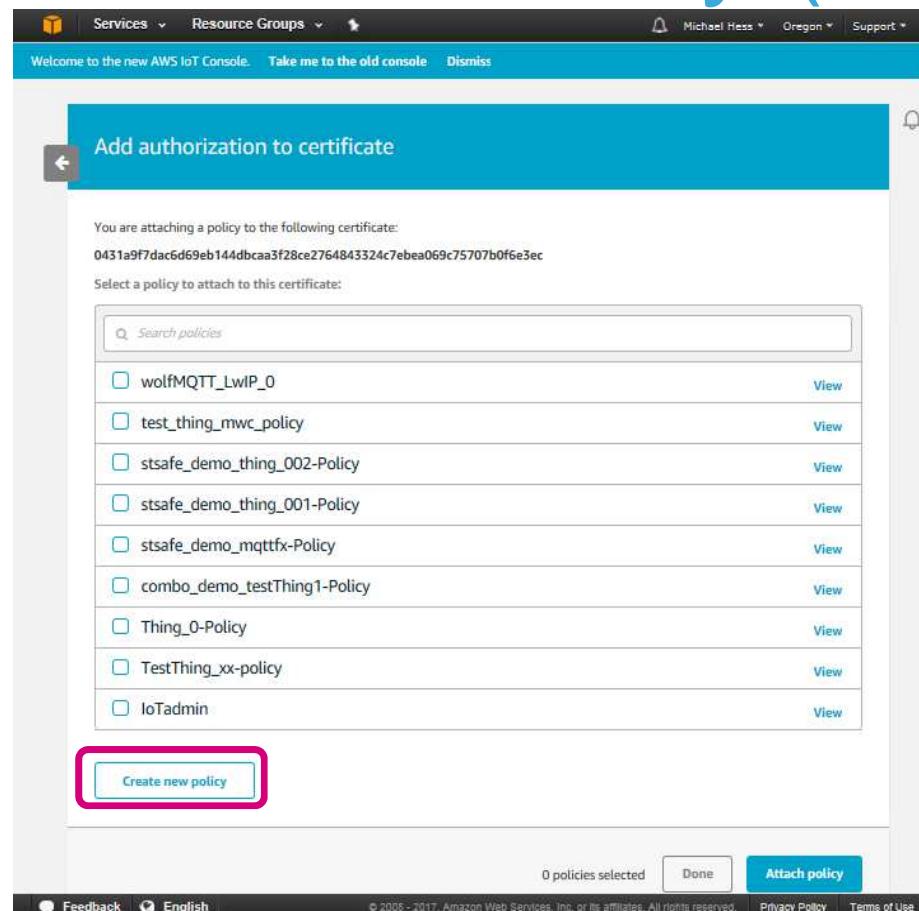
The screenshot shows the AWS IoT Console interface. At the top, there's a navigation bar with 'Services' and 'Resource Groups'. Below it, a banner says 'Welcome to the new AWS IoT Console.' with links to 'Take me to the old console' and 'Dismiss'. A green success message box displays 'Certificate created!' with a back arrow icon. Below the message, a note says: 'Download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys cannot be retrieved after you close this page.' Underneath, a section titled 'In order to connect a device, you need to download the following:' lists three items with download links:

A certificate for this thing	9ac755a9c7.cert.pem	Download
A public key	9ac755a9c7.public.key	Download
A private key	9ac755a9c7.private.key	Download

Further down, a note says: 'You also need to download a root CA for AWS IoT from Symantec: A root CA for AWS IoT [Download](#)'. At the bottom right, there are 'Done' and 'Attach a policy' buttons, with 'Attach a policy' being highlighted by a red rectangle.

Create a Policy (2/4)

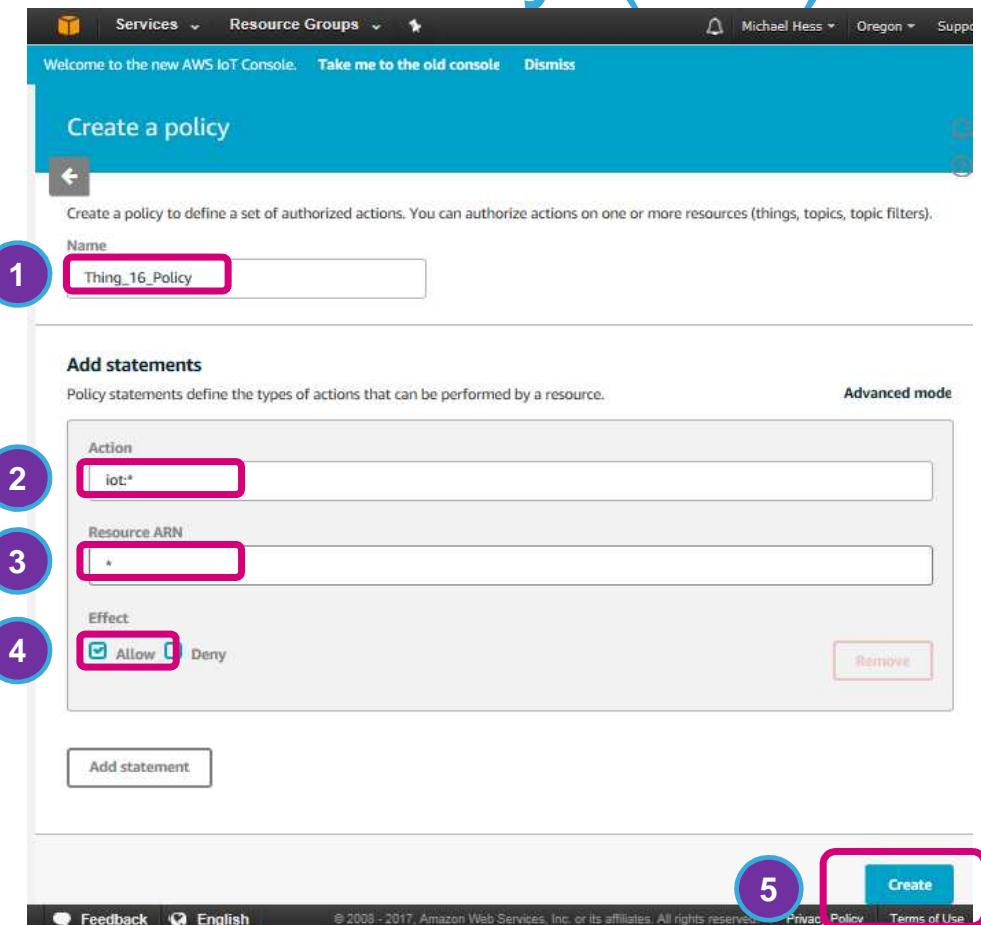
- Click “Create new policy”



Create a Policy (3/4)

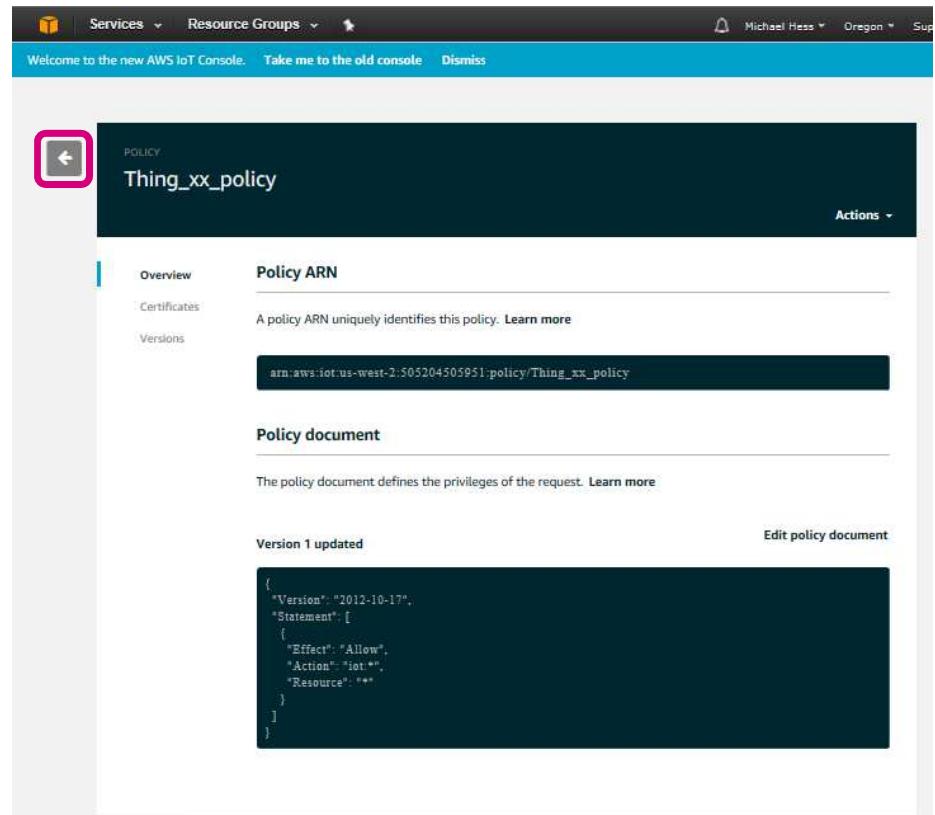
252

1. Name your policy “Thing_xx_policy”
with xx being your participant number
2. Under Action, type “iot:*
3. Under Resource ARN, type “*”
4. Click “Allow”
5. Click “Create”



Create a Policy (4/4)

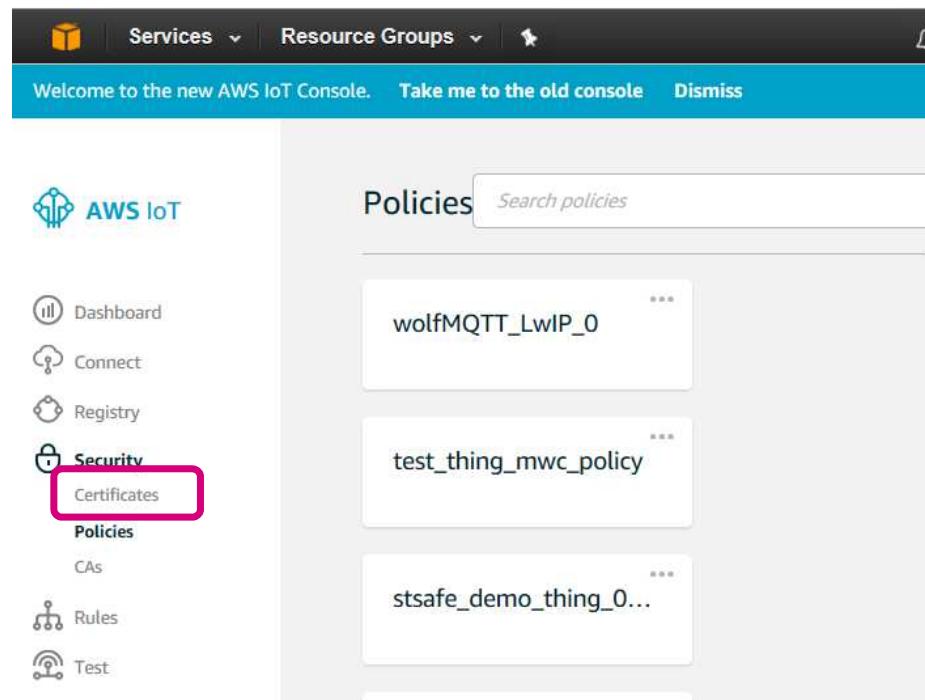
- Your policy has now been created.
- Click the gray arrow.



Attach a Policy (1/5)

254

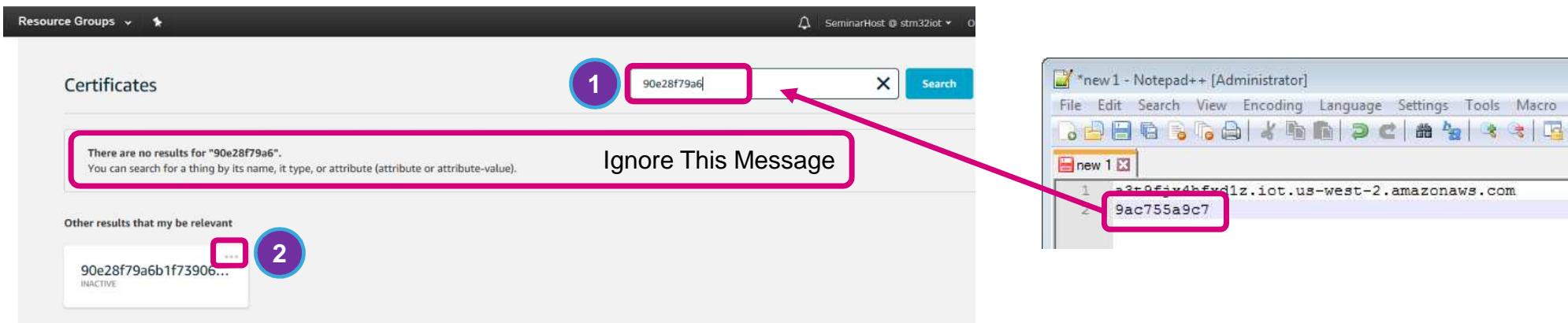
- Select “Certificates” under the “Security” menu.



Attach a Policy (2/5)

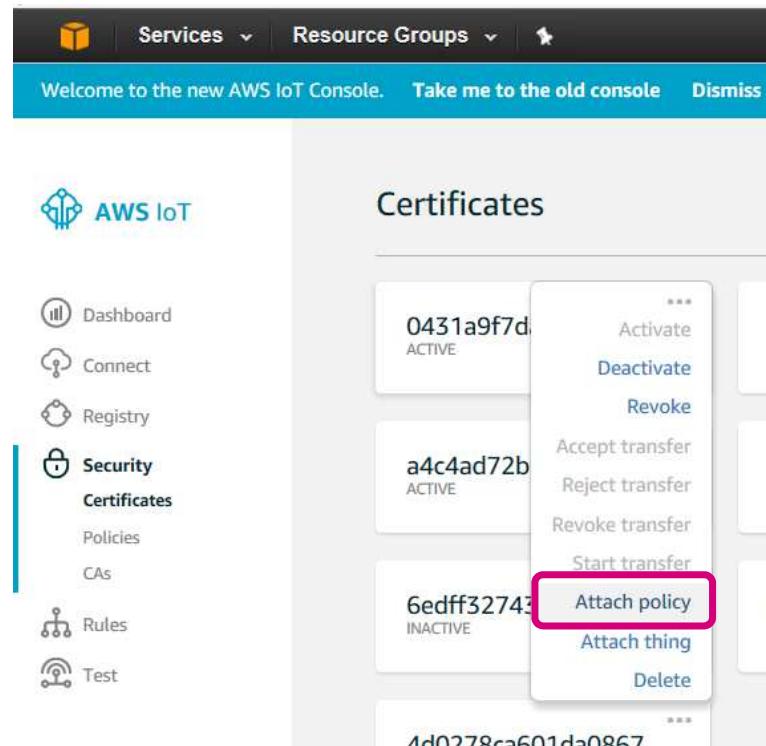
255

1. Use the search bar to locate your certificate using the certificate number.
2. Click the “...” in the top right corner of your certificate.



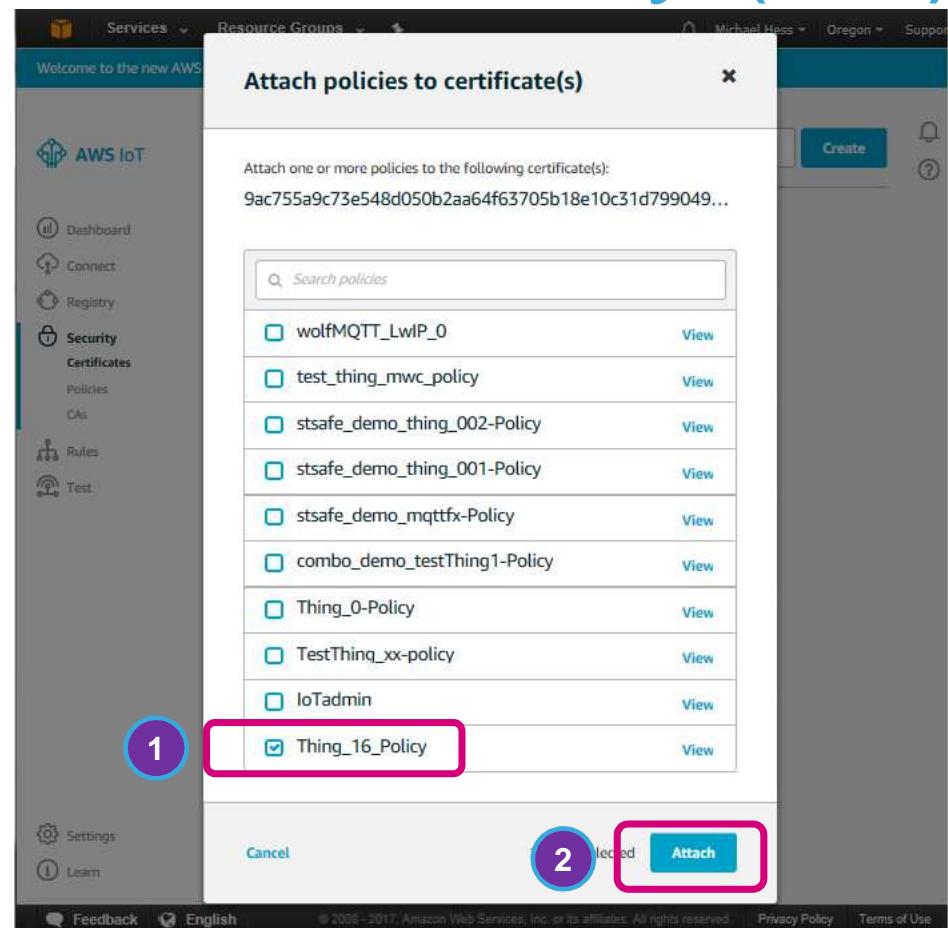
Attach a Policy (3/5)

- Click “Attach policy”.



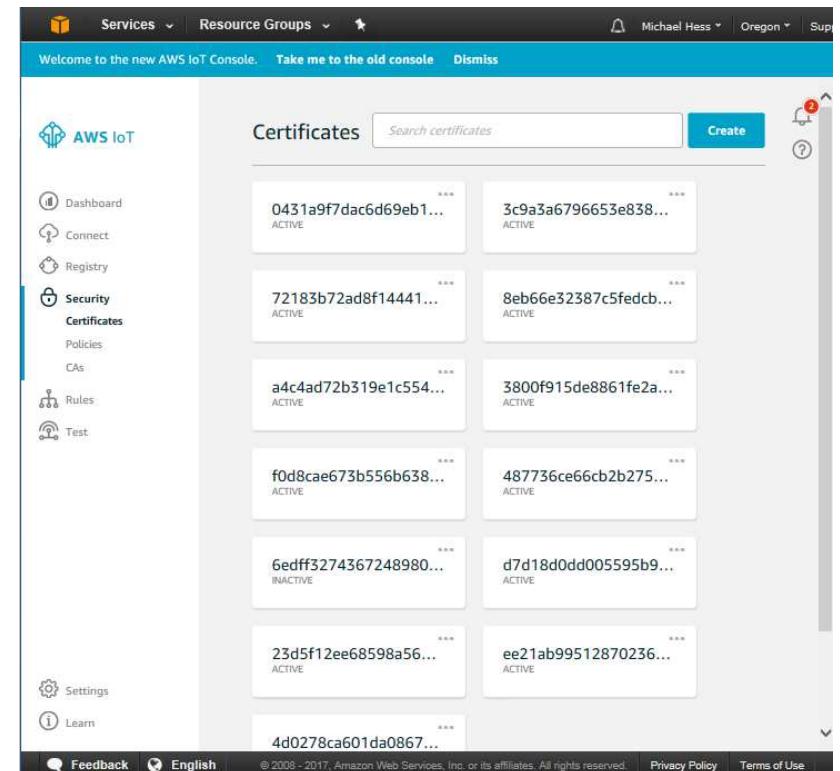
Attach a Policy (4/5)

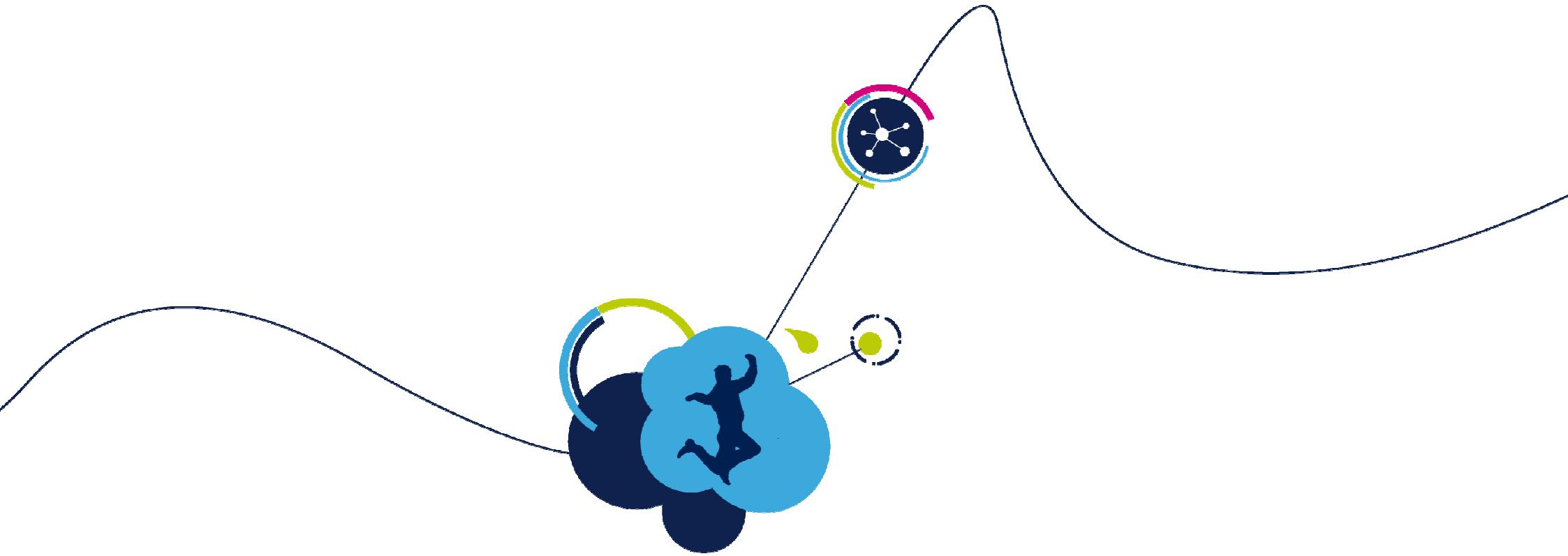
1. Select your policy
2. Click “Attach”



Attach a policy (5/5)

- Your policy has now been attached to your device certificate.
- We are done with the Thing creation on AWS.
- Next is firmware configuration.

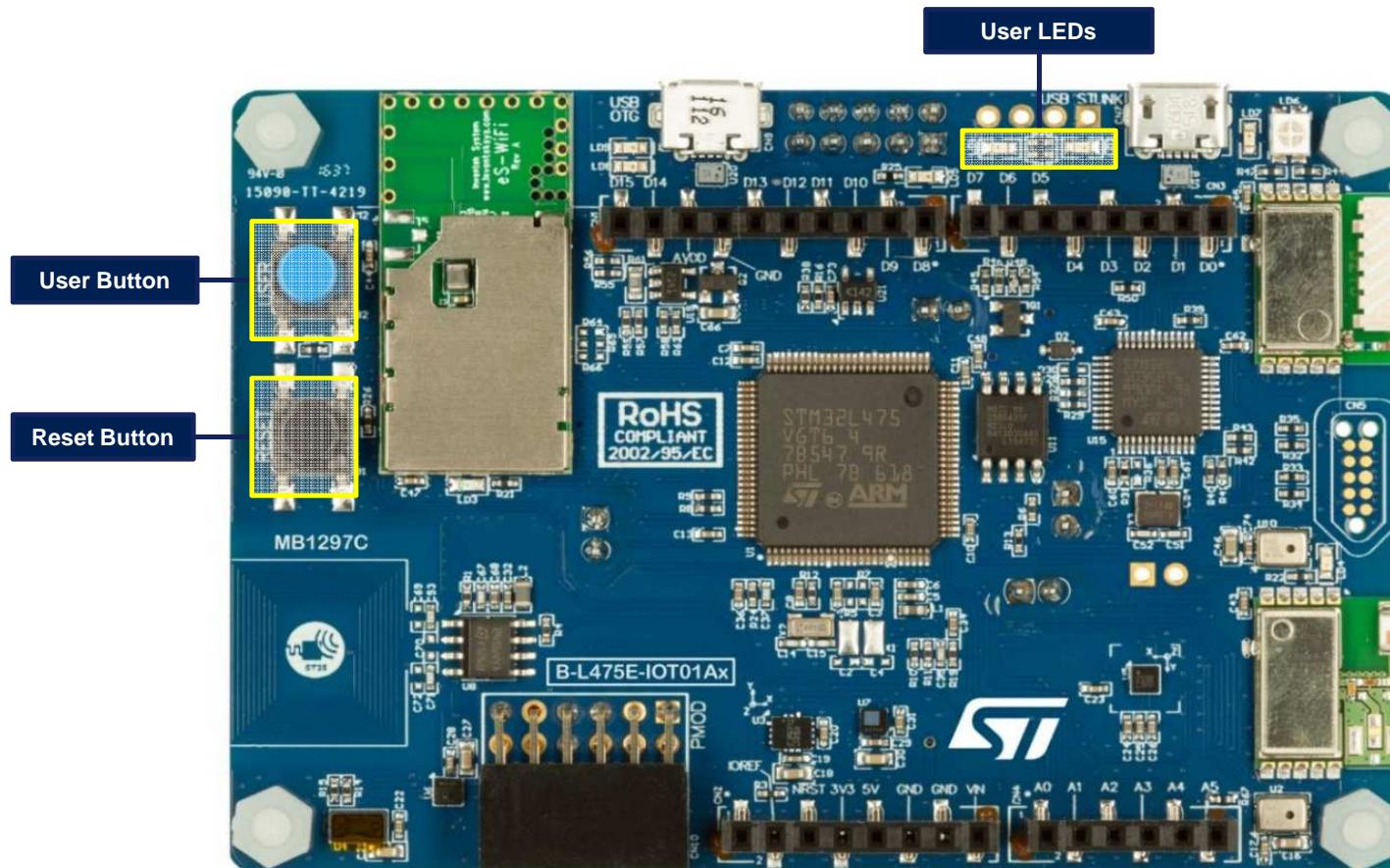




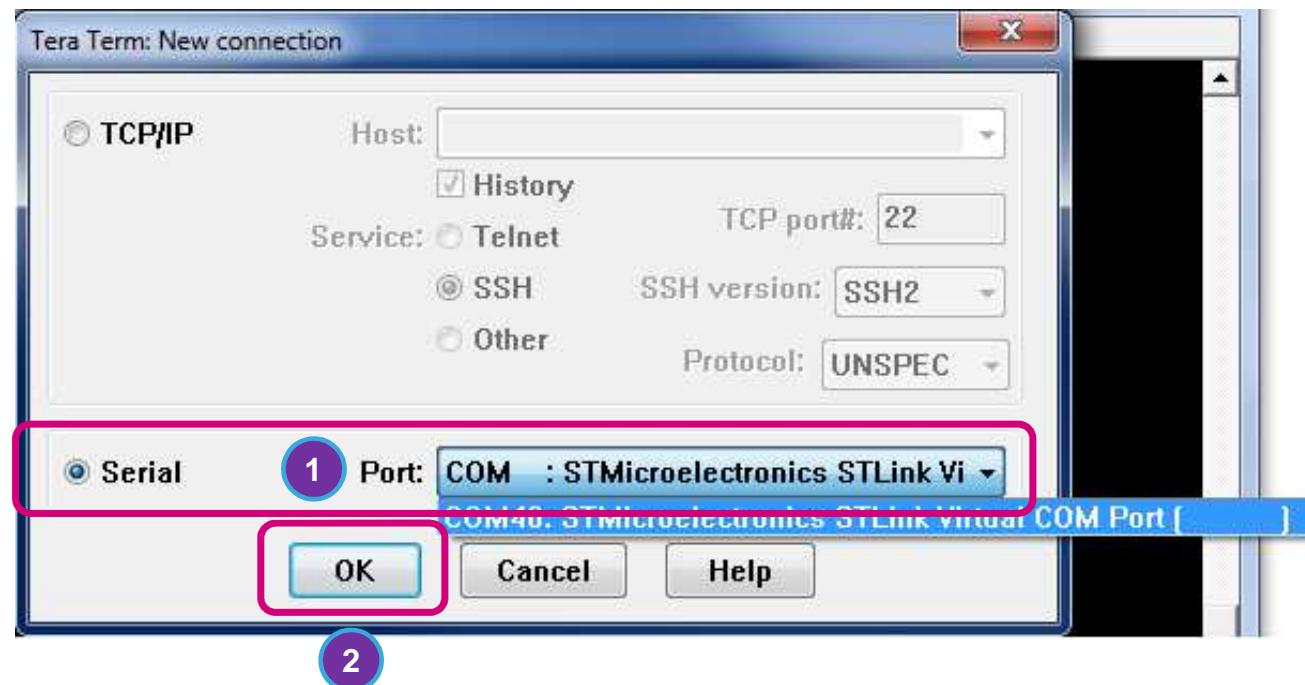
Lab 7 : Connect to AWS IoT & Send Sensor Data

- In this lab you are going to connect your board to AWS IoT:
 1. Load firmware on the STM32L4 Discovery Kit IoT Node
 2. Set the Wi-Fi credentials
 3. Set the AWS Root Certificate, the device Certificate and the device Private Key
 4. Set the MQTT server address and your device name
 5. Interact with your Thing

STM32L4 Discovery Kit IoT Node



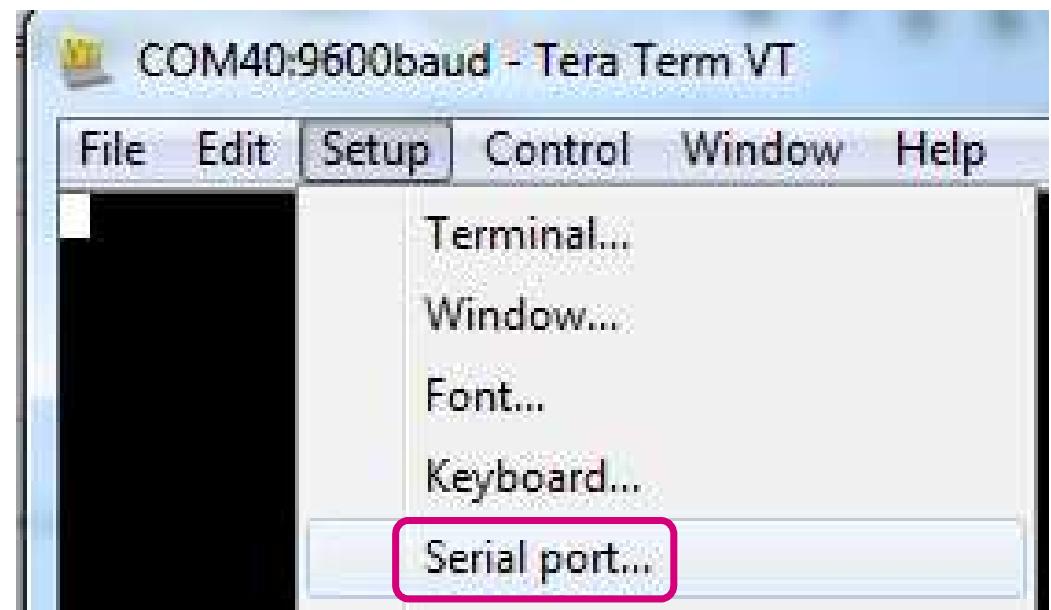
Tera Term Configuration 1/5



1. Select the **STMicroelectronics STLink Virtual COM Port**
2. Click **OK**

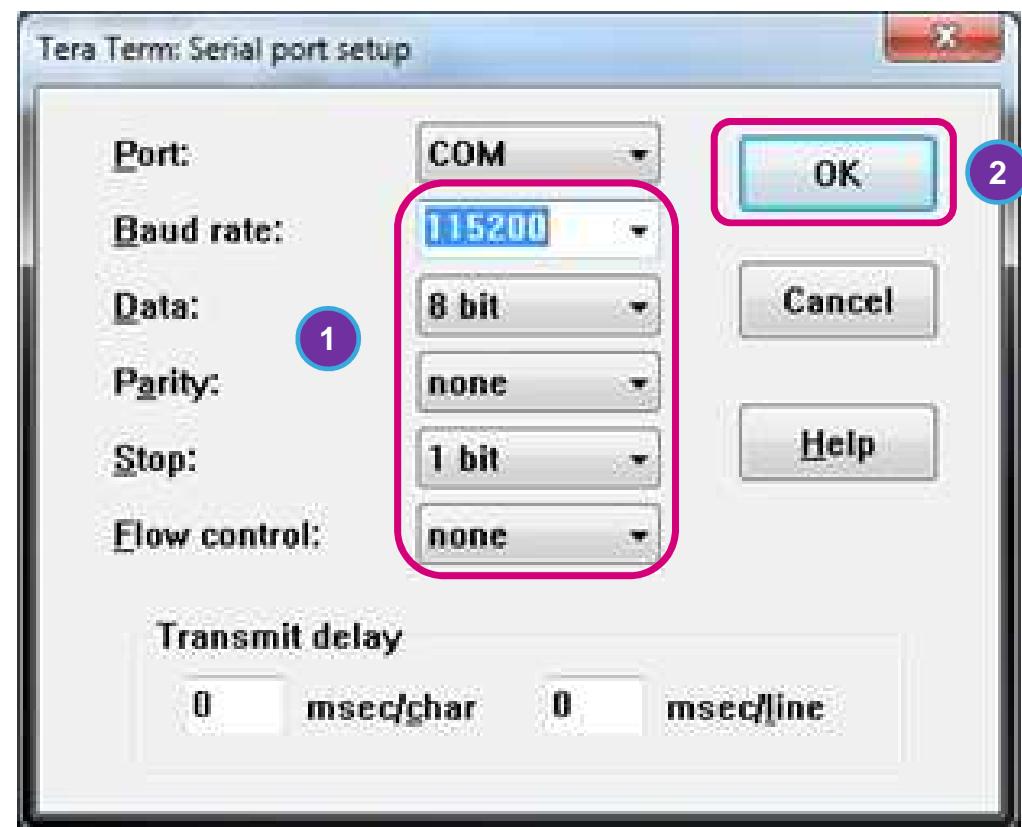
Tera Term Configuration 2/5

1. Click **Setup** -> **Serial port...**



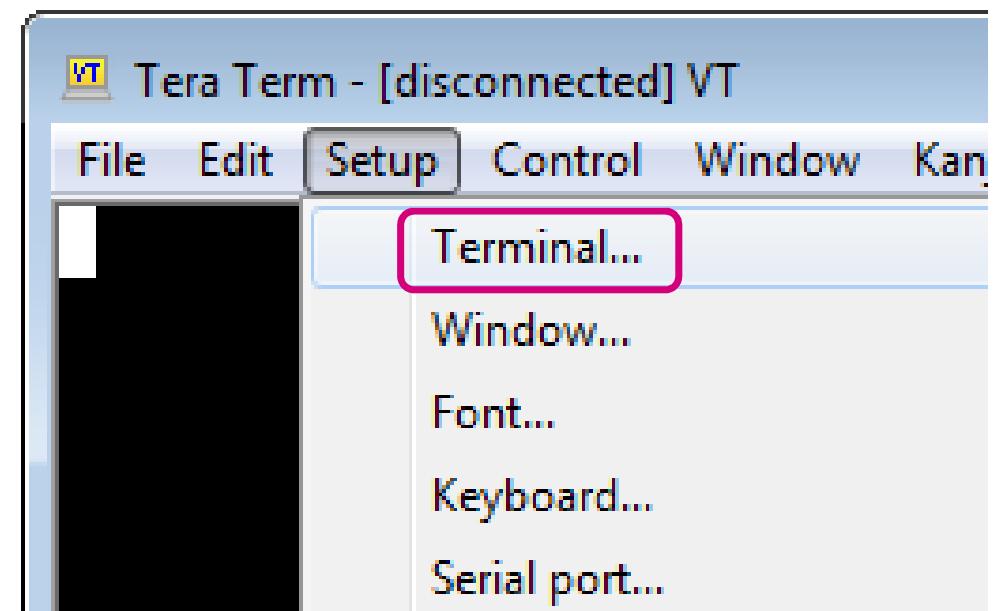
Tera Term Configuration 3/5

1. Set the following:
 - Baud rate : **115200**
 - Data : **8 bit**
 - Parity : **none**
 - Stop : **1 bit**
 - Flow control : **none**
2. Click **OK**



Tera Term Configuration 4/5

1. Click **Setup** -> **Terminal...**



Tera Term Configuration 5/5

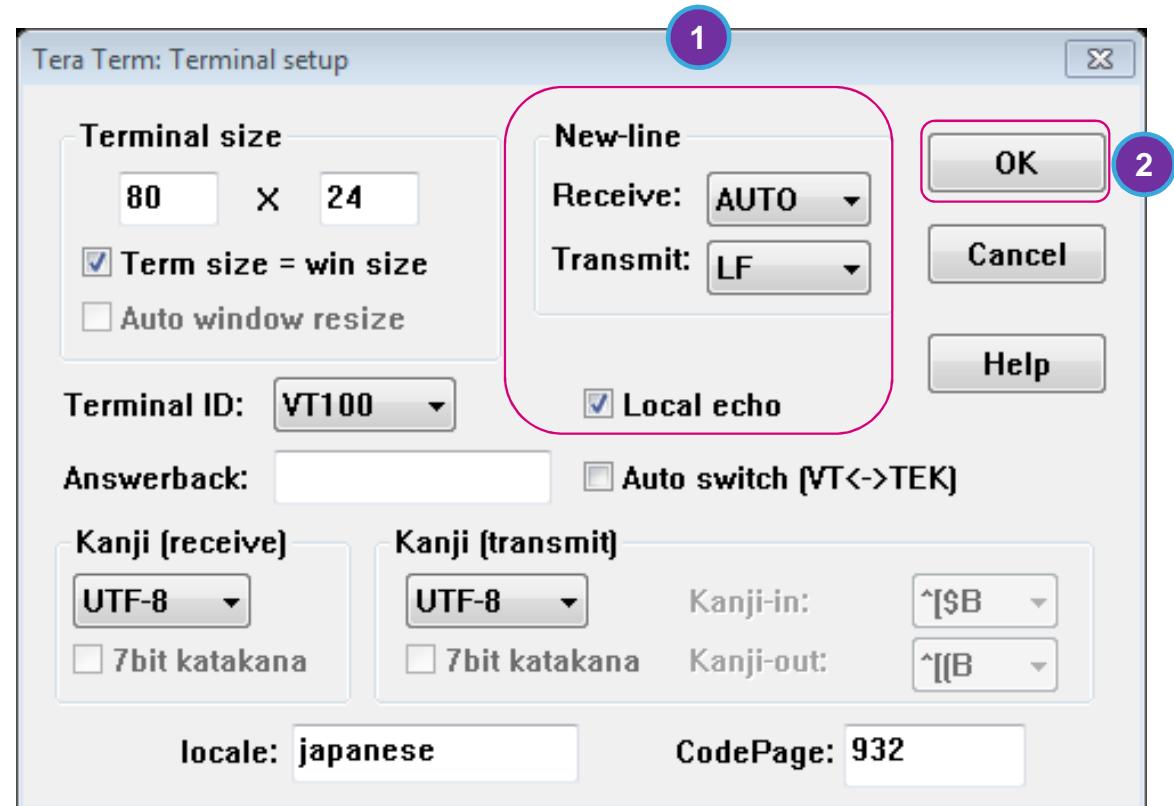
1. Set the following:

New-Line Receive : **AUTO**

New-Line Transmit : **LF**

Enable **Local echo**

2. Click **OK**



Open AWS_HandsOn Project

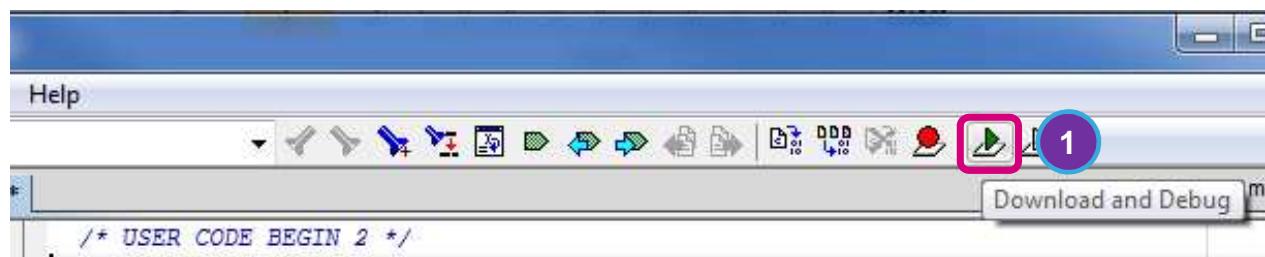
- Now we are going to configure the AWS firmware program to set the network SSID and password, set the AWS Certificates and establish an AWS IoT MQTT connection.
- Close the previous IAR project.
 - Double click on the `project.eww` file located here:

`C:\STM32L4_DK_IoT_Node_Seminar\Hands-on\Labs\Projects\B-L475E-IOT01\Applications\Cloud\AWS\EWARM`

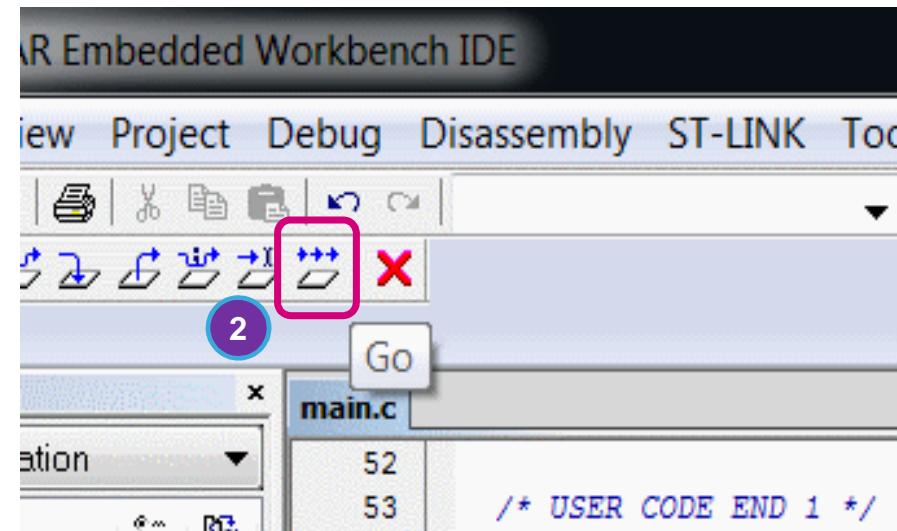
Load and Run

268

1. Click the GREEN ARROW to Build the Project, Download and start the debugger. (Ctrl + D)



2. Click the triple-arrow GO button! (F5)



Wi-Fi Configuration

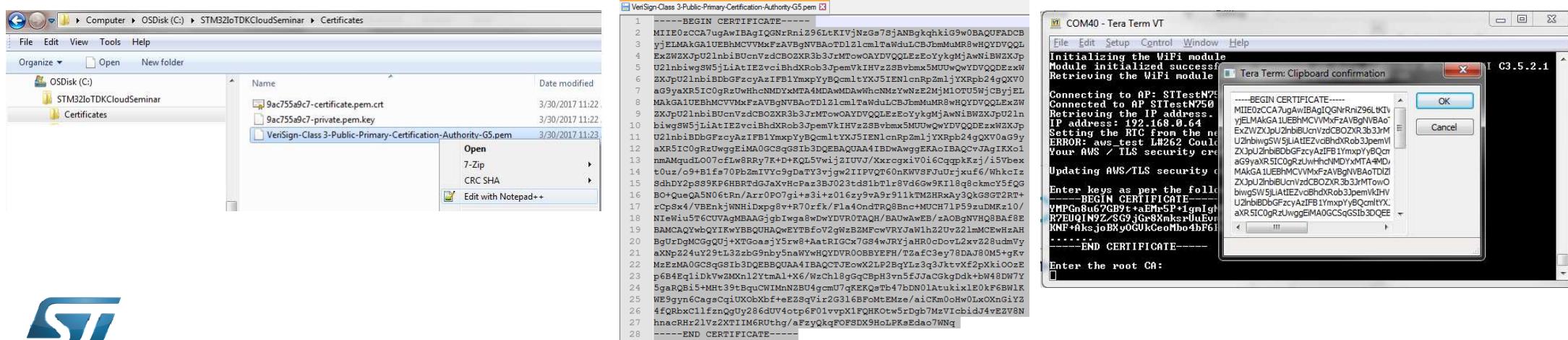
269

1. Set the SSID to **stm32iot**
2. Type **3** for the WPA2 Security mode
3. Use **stm32iot** for network password

```
Press the User button (Blue) within the next 5 seconds if you want configuration  
(Network and AWS security credentials)  
Do you want to configure the Wifi credentials ? (y/n) y  
1 Enter SSID: stm32iot  
You have entered stm32iot as the ssid.  
Enter Security Mode (0 - Open, 1 - WEP, 2 - WPA, 3 - WPA2):3 2  
You have entered 3 as the security mode.  
3 Enter password: stm32iot
```

Set the AWS Root-CA Certificate

1. Open the **VeriSign-Class 3-Public-Primary-Certification-Authority-G5.pem** with Notepad++
2. Copy the certificate content (Ctrl + A, Ctrl + C)
3. Paste the certificate in Tera Term using “Edit -> Paste” (**Do not use Ctrl + V**) and click OK
4. Click Enter



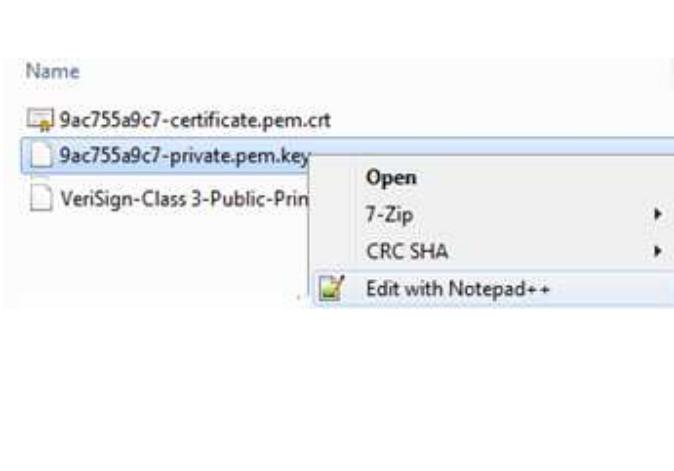
Set your Thing Certificate

1. Open your **xxxx-certificate.pem.crt** with Notepad++
2. Copy the certificate content (Ctrl +A, Ctrl + C)
3. Paste the certificate in Tera Term using “Edit -> Paste” (**Do not use Ctrl + V**) and click OK

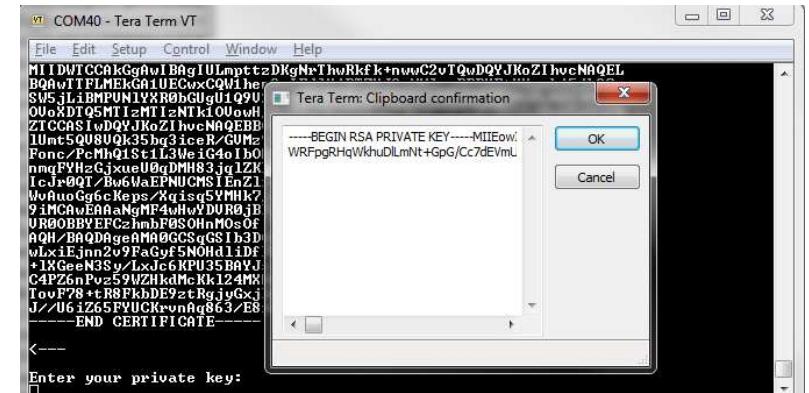


Set your Thing Private Key

1. Open your **xxxx-private.pem.key** with Notepad++
2. Copy the certificate content (Ctrl +A, Ctrl + C)
3. Paste the certificate in Tera Term using “Edit -> Paste” (Do not use Ctrl + V) and click OK



```
9ac755a9c7-private.pem.key [1]
1 -----BEGIN RSA PRIVATE KEY-----
2 MIIEowIBAAKCAQEAK24x08Cf2Ui8Y/4nSeGVsa3LBXvCTflureJx5H8ZUzP1HCM
3 SUettagGzvLCPgcrpgiBYEttrfNeYLMqwhcTUWidz89wyFDVkr3Uvd61high4xROW
4 Rcp4uiiRyKDHoAly2nBR5/EUfta2loutmg6eacVgfMaPG55TS8cmwfzeQvkcs18Q
5 lR611vQxN+1426mcxxF+g/1X3RnE8L3t89yLhwmwRBPHD5pZcQ81IxIgSdmWtxTu
6 DifqmjEpddz+GswGkUj2QRXAUejymBD91Uta8C6gaDwpw6mz9eqKyrlgwetTu06TG
7 2xDCAFULbGm2aiXlmm26Er56pbkseYbp#21WIIDAQABaoIBAC1QgZNQjw5r1bb
8 Q/8iealejxRgJkRNazH7FNN/PwxMCLPyBN61tRo43Jv6B0qExfwNAuCCIAjOaj+
9 d2pGTbxZEYd3dcDahc/CawXfffM9QnsGa829KGjqpY177K+0FWMBdc6TSPNr10tW
10 J1dmnnff6mxFqUlmqnZ5P7vgODra/vYHcJySth4f86enru9nt1NLsxyplqvphy
11 gZop5a3kNjsocgDQJ6kesburjwMxWbScjAlxxlrgD2+9inNxhuerkcing+18FWN
12 C7tr1Kehik3608Fy/PnEEd5PrbkaHSTLQ18t/h3iqHn4PkocNv8sC1Ht7qMKYA
13 xYY3RLRECgYEA/rMDgxFSxv5zNt1xPCIGerRwdImfwv4bmPfJUVXnHr+5doHmeiXv
14 OAqrzFHagWg0rfjp3eGVBL1VpgAJCpuHnnImcuHNHcgDr1GsXeoJy3ghn5+3mb
15 RN9vQjSHZrjodSqrrehx85+4SBNCphcUGGJ76X8s2kLi5k1j8/eDormkCygEA1c7w
16 q1GNE4sktSQUNASHz06y7yxNxhuz777C521Bspkw7gv86InUFL14HbzstH2M2C0dl
17 OdgeAqHVA7Pq8rxrbWRFpgHqWhkul1LnNt+Gp/G/c7dEvnUklDbpMjm/X5WMUji
18 0fmida3ehXlkok7PK3zfudrofxvNv0dy203BqsCgYEAgOlgrQWa98Y9yKdNm9A
19 pMPFY/7gLF71rlmCo/F57vpKeDk74XpWLOpeCxuoVouMn8Qs58Dp+oGhassfBI
20 W1qnu2We/AE02ingTyb3tyjtFFbrLqXsg7tgzjj2dQ+0w1SWrxht2Zf7at3xnX
21 hWxf8tpIw8BWcGuv9hCxkCgYBWAjTQy2tafjWQ2zMsLfFBjNfJc2bcYWC2pLI+G
22 F52dhCBOfOnb2XwOky78R1FnGmbsf26tN2blcDESG6re2/DK7s9VCigDRR0UFUv
23 ON693zCKRdaol10k3621pLwmFylCKC0hgQqpFTVkr353o20t6xro+DIUCGghr6LW
24 izU2RwK8gHQYPhkxujCQHRFxWF5NC1QiZa+FPuCfYwqsf0YRC8EHXLif18DV69yL
25 Y30+twLsRPrz/dFIQ+wiwbp81gU9af1wdgHdER4gwQ92AWyuQED+iHnfIKRMzYrD+
26 1shA4b/Naxi2dfCI50WQwptBRSW3zV8rsMy0H8gbGpk/l130jtvu
27 -----END RSA PRIVATE KEY-----
```



Set the Server and Thing Names

1. Copy the server address from the text file and paste it to Tera Term, then press Enter
2. Set the device name to “**Thing_xx**” with **xx** being your participant number (2 characters). Example: Thing_08, Thing_16

The image shows two side-by-side Tera Term windows. The left window displays a text file with a server address highlighted by a red box. A yellow arrow points from this box to the right window, where the same server address is pasted into the terminal. The right window also has a yellow box highlighting the command to enter the device name, which is currently set to "Thing_16".

```

new1 - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro
a3t9fjx4hfxd1z.iot.us-west-2.amazonaws.com

COM40 - Tera Term VT
File Edit Setup Control Window Help
JidMu7fE6MKPfU1mqN2SP?vgODrA?0yHcJyStshf8S6enru9Nt1LwvxplQuphy
ZOp5s3YkNjsqDQu6KeShurjuMxJhs5jaixx1rqgD+9inWxHuerkC1RG+18PUW
xZb7vIkkeh360BEy7/PnEEd5PxbkaHSTLQ18t/h3igHh4PkoCnV8sC1ht?gMKVA
xu3RLECaVEA/xMdqXFSXvSZNt1xPcIGeRwdImfwU4gbnFIJUxNhix+sdoHMeixU
0ArxZFHaglg0p+jn3eGUB3JUproAJCpuBHnnI CuahHNcg9Dw1GsXsoJy3gdh5+3nb
nN9vQjSHZEjodSxqreJx85+dBMCphcUGGJ16x8S2oK1i5k1jB/eDomrkogVE01C7w
1GNc4skTSQUNMSH2oG7yNxK7T7CS21DsKpw?gu861nUFfI4hBzs2M2ZC0D1
0dgeHqU97PqdFxxBWRpgMhqWhhD1Lm+7Gp/C7dEUuUL1dbpMjhX5WMUji
3fmida3EM1kok7PK3zFuadpoFxobNu0dy285BqsCgYB9gElggKWA98v9KKDsf9A
p/PPY/?rGLF1?rLnCo/P57vpKeDK74XPeW1OPECxuoouhwsq5SDp+6hasfbl
M1qnuZMe/REu2ihgTyb3tydJtFFbrLqX8Sg7tGzHj32dQU+0w18WrxhtZ2F7at3yN
hWxfb1tpisW8BWGuv9CxhCgYBmAjtQy21afjWQzMsLFBjNPJ02bcVWC2pLI+G
K52dhCxFONb2XwOky78K1FnGmbSF26t2K1oDESG6re2/DK7s9UcigDRR0UFUv
oN693zCktao110k3621pLuFy1CKC0hgQqp@TUK353ok20t8xr0+D1UCGhr6LW
izU2RuKBgVPhkbxJgQHFRxFf5Mc1Q1z8+FPuCFYwqsf0YRC8EHXLfI8DU69y0
Y30+tuLSRPz/Df1Q+wivBp81g9afLwdGHdER4gwQ92AVyuQED+iHnf1QKMzYrD+
ishA4b/Naxi2dcC1500qWePR5W3r08rsMyv0H8ghGpk/1130jtvv
----END RSA PRIVATE KEY----

<---
Your server name and thing name need to be entered to proceed
Enter server address: <example: xxx.iot.region.amazonaws.com>
a3t9fjx4hfxd1z.iot.us-west-2.amazonaws.com
end: --->
a3t9fjx4hfxd1z.iot.us-west-2.amazonaws.com
--->
Enter device name: <example: mything1>
Thing_16

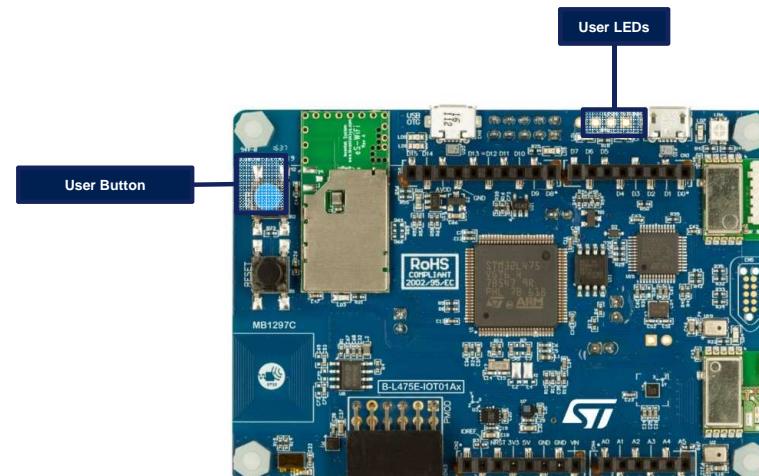
```

Send Data to AWS

274

- The board will start sending sensor data once it is connected to AWS (every 10s for 10mn).
- Press the user button (the blue button) and the board send/receive LED messages to/from AWS IoT.
- The LED on your board will toggle every time the LED message is received.

```
Thing_16
<---  
*** Firmware version management ***  
Press the BLUE user button within the next 5 seconds  
to change the firmware version from 1.0.0 Mar 20 2017 10:26:42, running from bank #1.  
*** AWS connectivity demonstration ***  
AWS IoT SDK Version 2.1.1-  
MQTT connection in progress: Attempt 1/3 ...  
Connected to a3t9fjx4hfxd1z.iot.us-west-2.amazonaws.com:8883  
Subscribed to topic $aws/things/Thing_16/shadow/update  
Press the User button <Blue> to publish LED desired value on the $aws/things/Thing_16/shadow/update topic  
Published to topic $aws/things/Thing_16/shadow/update:  
Temperature: 28.67  
Humidity : 35.30  
Pressure : 1041.19  
Proximity : 20
```



Monitor your Thing Activity

1. Select the AWS Console on your browser
2. Select Registry
3. Select Things, then select your Thing
4. Click on Activity

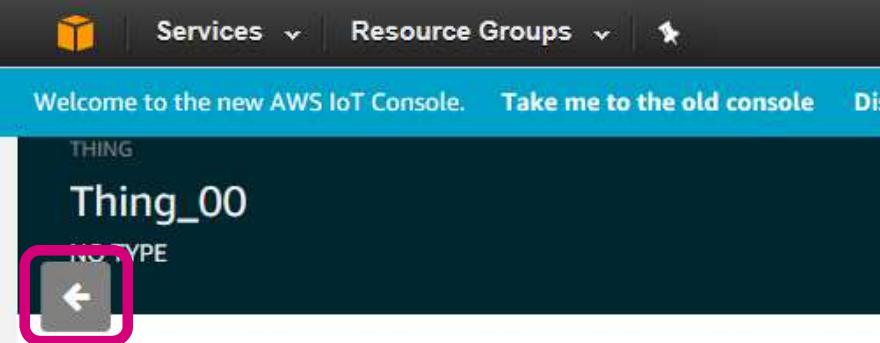
The image consists of two side-by-side screenshots of the AWS IoT Console.

Screenshot 1 (Left): Shows the main AWS IoT dashboard. The sidebar on the left has buttons for 'Dashboard', 'Connect', 'Registry' (which is highlighted with a red box and circled with a blue number 2), 'Things' (circled with a blue number 3), and 'Security'. The main area shows a 'Things' list with items 'Thing_02' and 'Thing_04'.

Screenshot 2 (Right): Shows the details page for a specific thing named 'Thing_00'. The top navigation bar includes 'Services', 'Resource Groups', and a bell icon. Below the title 'Thing_00 NO TYPE', there are tabs for 'Details', 'Security', 'Shadow', 'Interact', and 'Activity' (which is highlighted with a red box and circled with a blue number 4). The 'Activity' section displays the message 'Listening for 0 minutes' and a green status indicator 'Shadow update accepted'.

Interact with your Thing (1/9)

Click the gray back arrow.



Welcome to the new AWS IoT Console. [Take me to the old console](#) Dis

THING

Thing_00

TYPE

◀

Details

Security

Shadow

Interact

Activity

Activity

Listening for 0 minutes

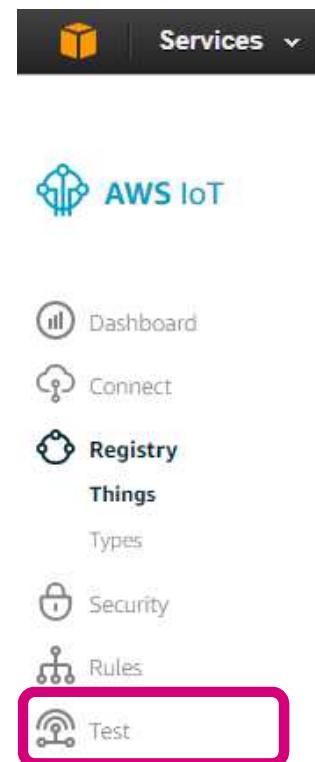
● Shadow update accepted

● Shadow update accepted

Interact with your Thing (2/9)

Select “Test” on the left panel.

This will open a web based MQTT Client.

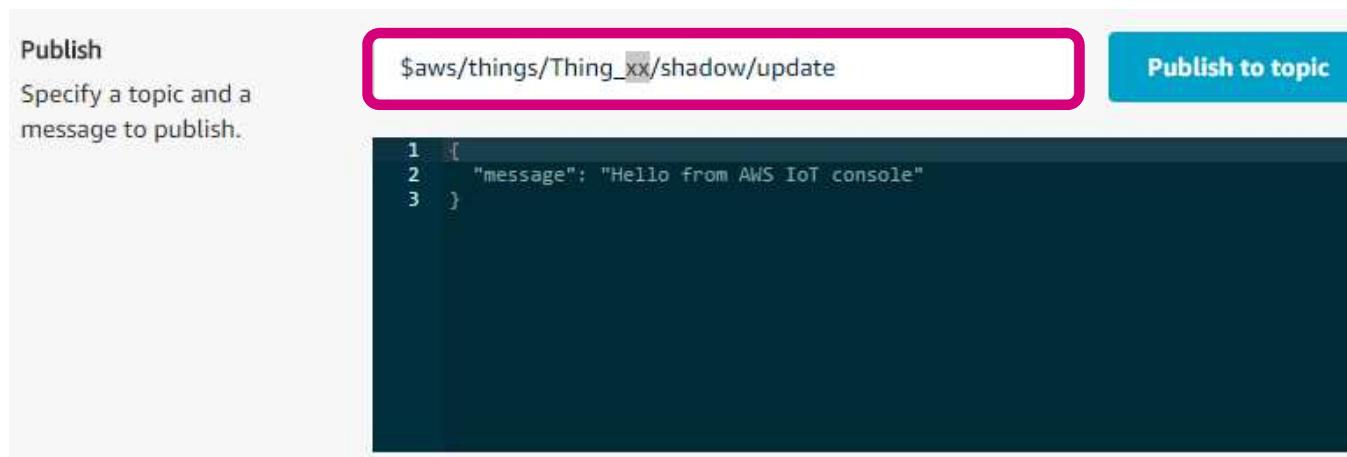


Interact with your Thing (3/9)

Under Publish, type

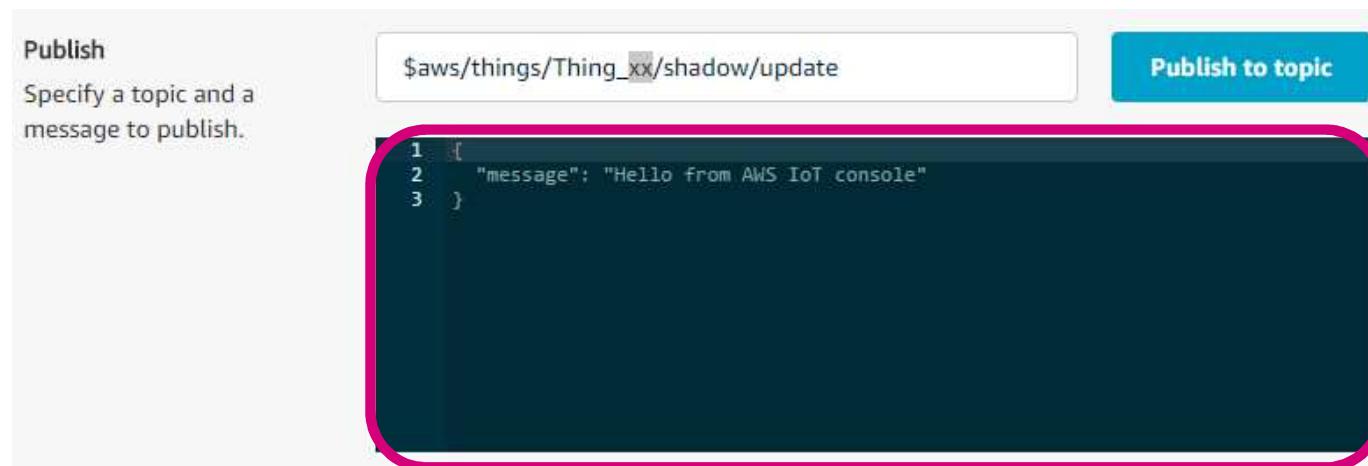
\$aws/things/Thing_xx/shadow/update

with **xx** being your participant number



Interact with your Thing (4/9)

Remove the default message in the dark area.

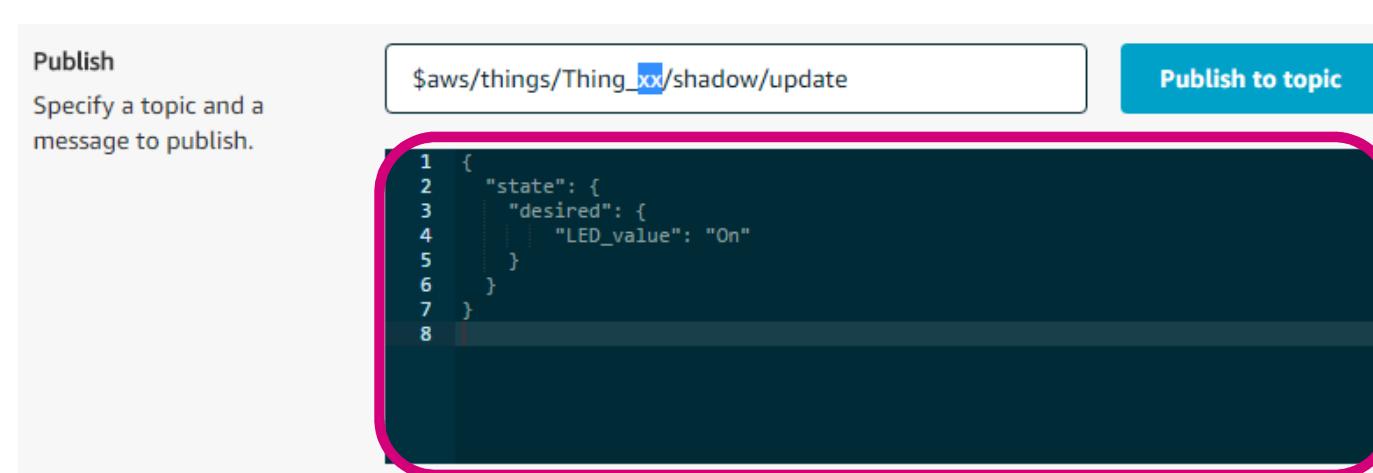


Interact with your Thing (5/9)

280

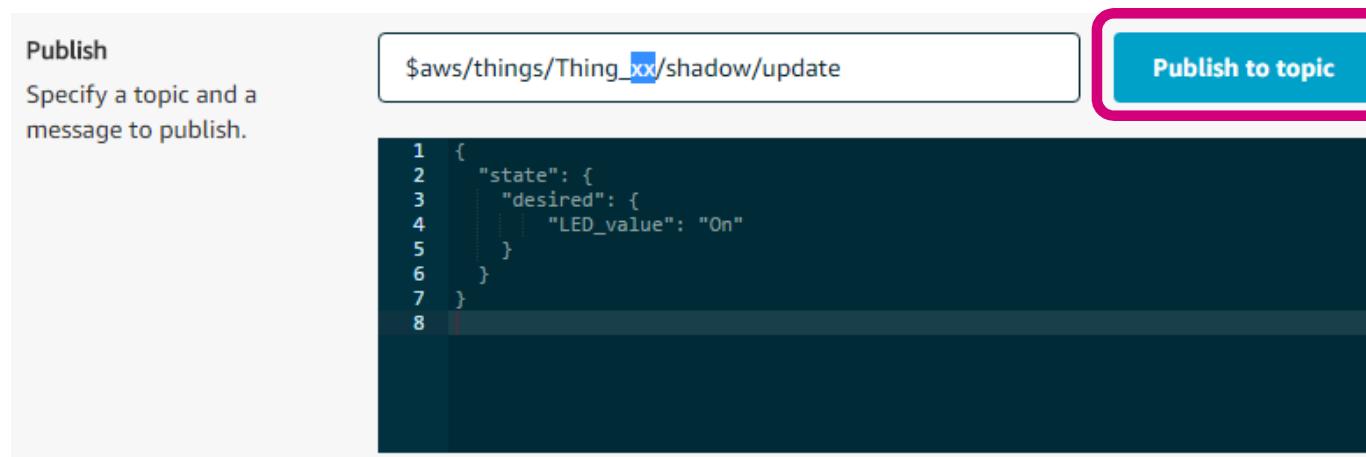
Type the following message:

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



Interact with your Thing (6/9)

Click “Publish to topic” and watch your board’s LED turn ON

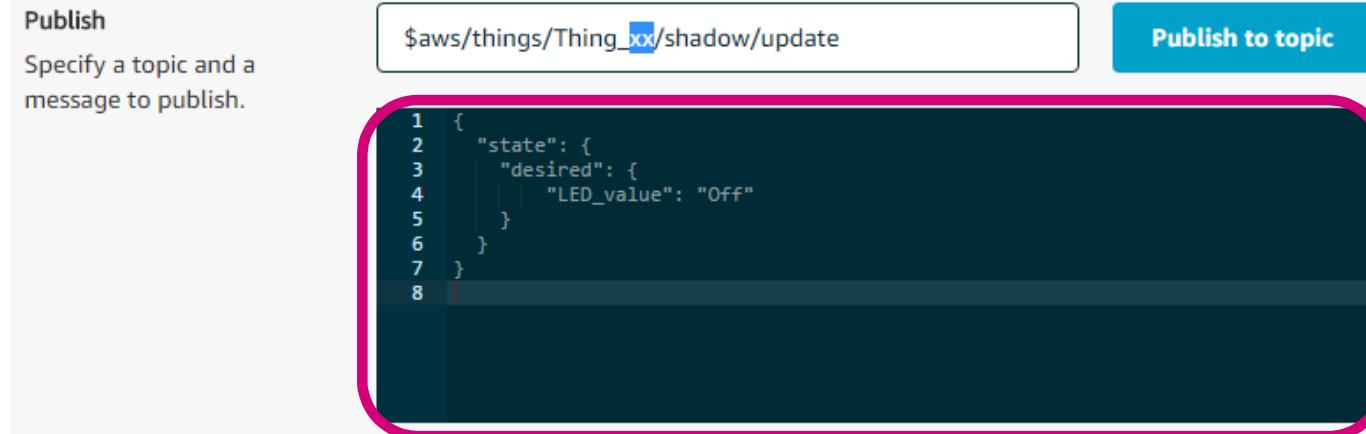


Interact with your Thing (7/9)

282

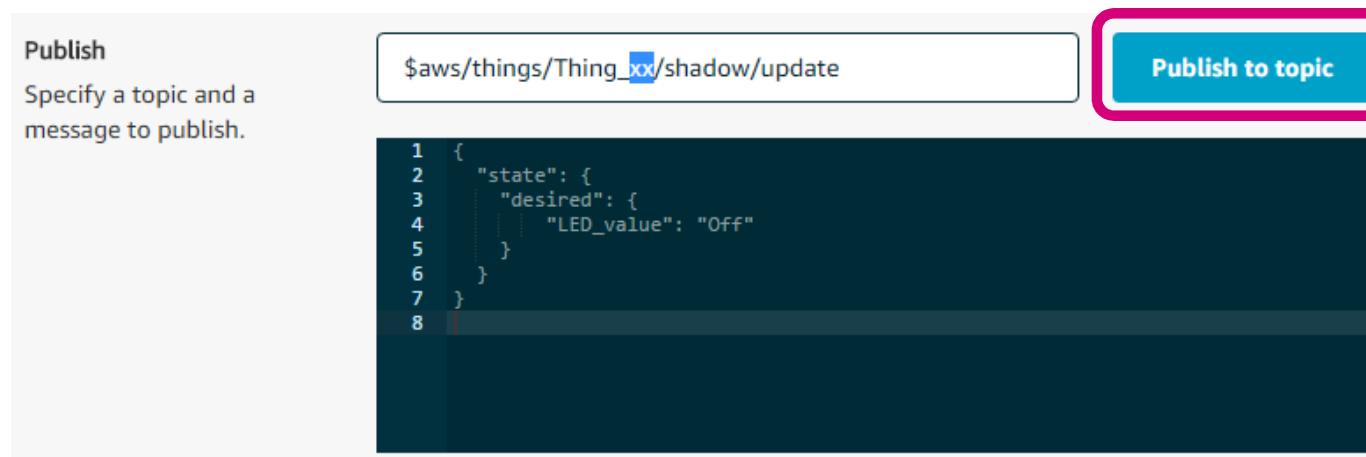
Modify the message as follows

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



Interact with your Thing (8/9)

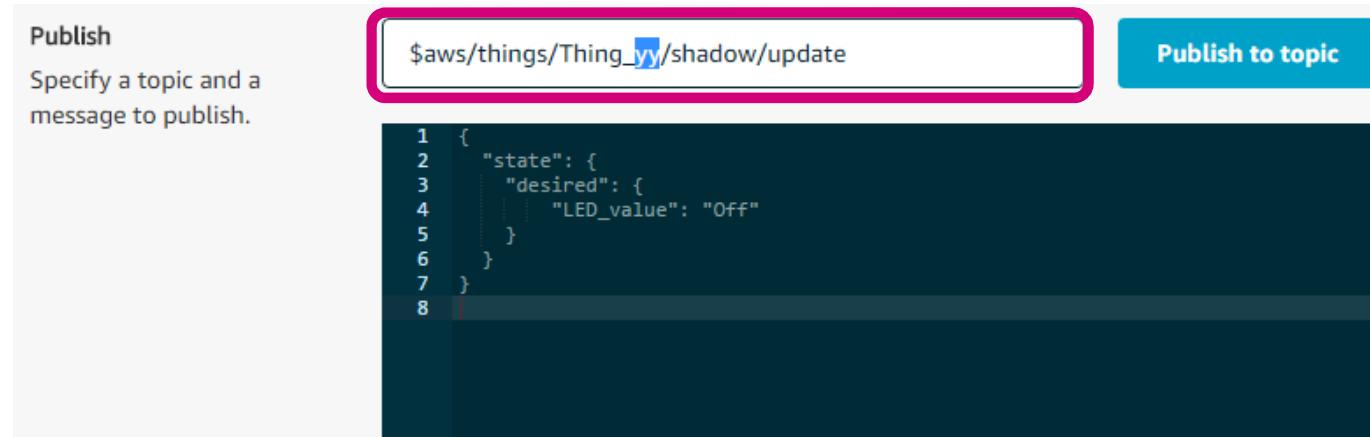
Click “Publish to topic” and watch your board’s LED turn OFF

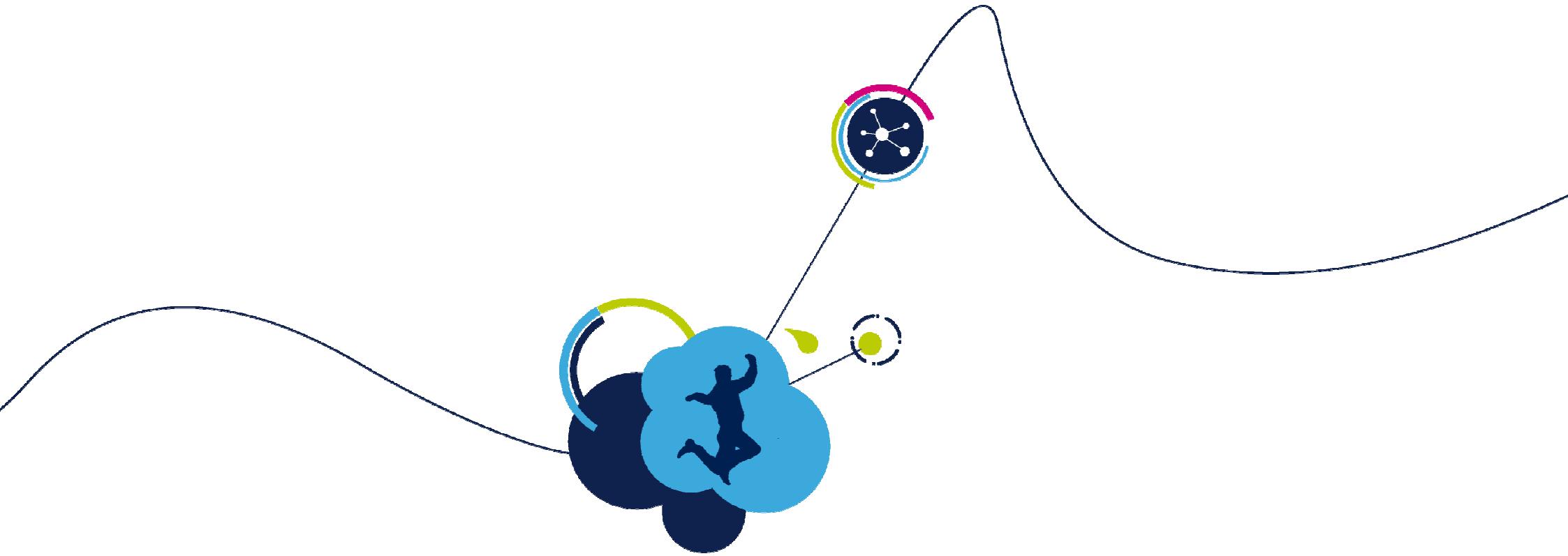


Interact with your Thing (9/9)

Change the publish topic to interact with another participant's board

\$aws/things/Thing_yy/shadow/update



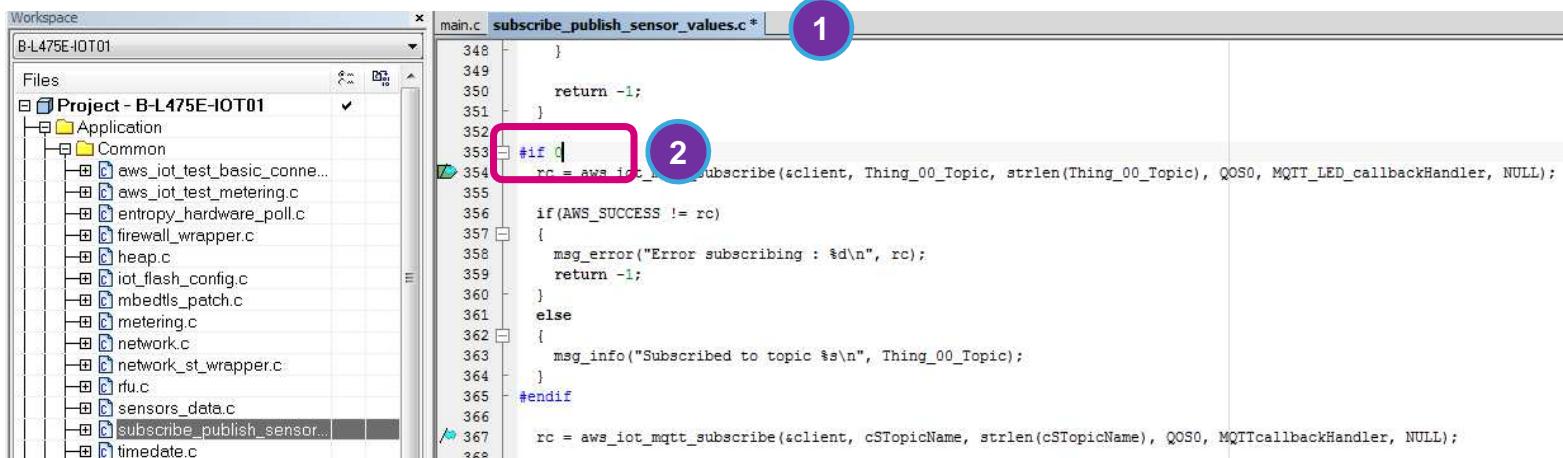


Lab 8 : Connect to a Different MQTT Topic

Change the Subscription Topic

- Now we are going to subscribe to the presenters' publishing topic. The LED on your board will toggle every time the presenter presses the button on his board.

- Open the `subscribe_publish_sensor_values.c` file.
- Change the `#if 0` to `#if 1` (Line 353).



```

Workspace: B-L475E-IOT01
Files: Project - B-L475E-IOT01
      Application
          Common
              aws_iot_test_basic_conn...
              aws_iot_test_metering.c
              entropy_hardware_poll.c
              firewall_wrapper.c
              heap.c
              iot_flash_config.c
              mbedTLS_patch.c
              metering.c
              network.c
              network_st_wrapper.c
              rfu.c
              sensors_data.c
              subscribe_publish_sensor...
              timedate.c

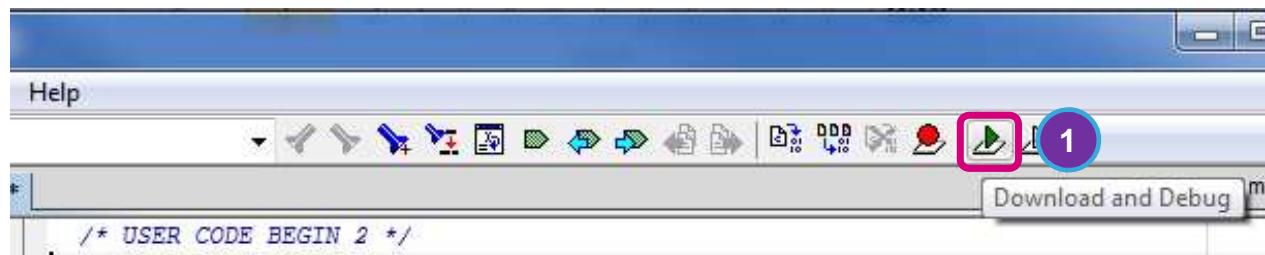
main.c subscribe_publish_sensor_values.c * 1
348     }
349
350     return -1;
351 }
352
353 #if 0
354     rc = aws_iot_mqtt_subscribe(sclient, Thing_00_Topic, strlen(Thing_00_Topic), QOS0, MQTT_LED_callbackHandler, NULL);
355
356     if(AWS_SUCCESS != rc)
357     {
358         msg_error("Error subscribing : %d\n", rc);
359         return -1;
360     }
361     else
362     {
363         msg_info("Subscribed to topic %s\n", Thing_00_Topic);
364     }
365 #endif
366
367     rc = aws_iot_mqtt_subscribe(sclient, cSTopicName, strlen(cSTopicName), QOS0, MQTTcallbackHandler, NULL);
368

```

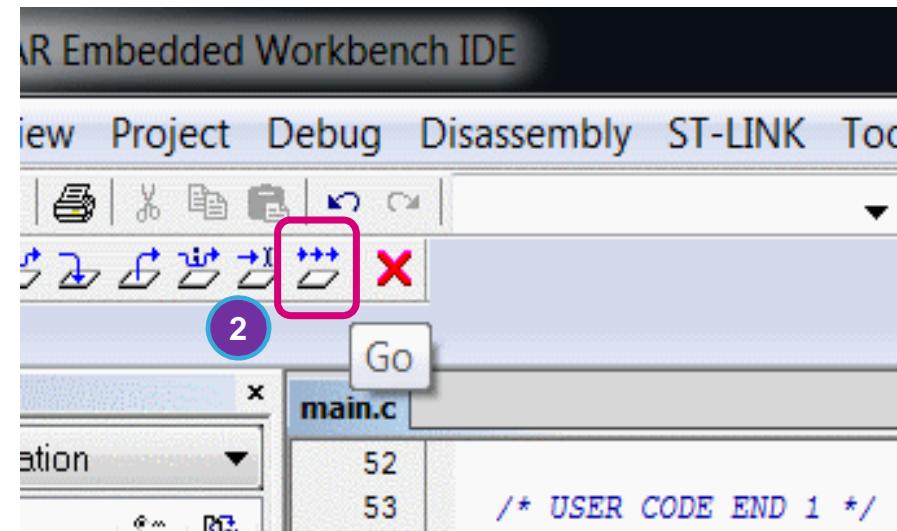
Load and Run

287

1. Click the GREEN ARROW to Build the Project, Download and start the debugger. (Ctrl + D)

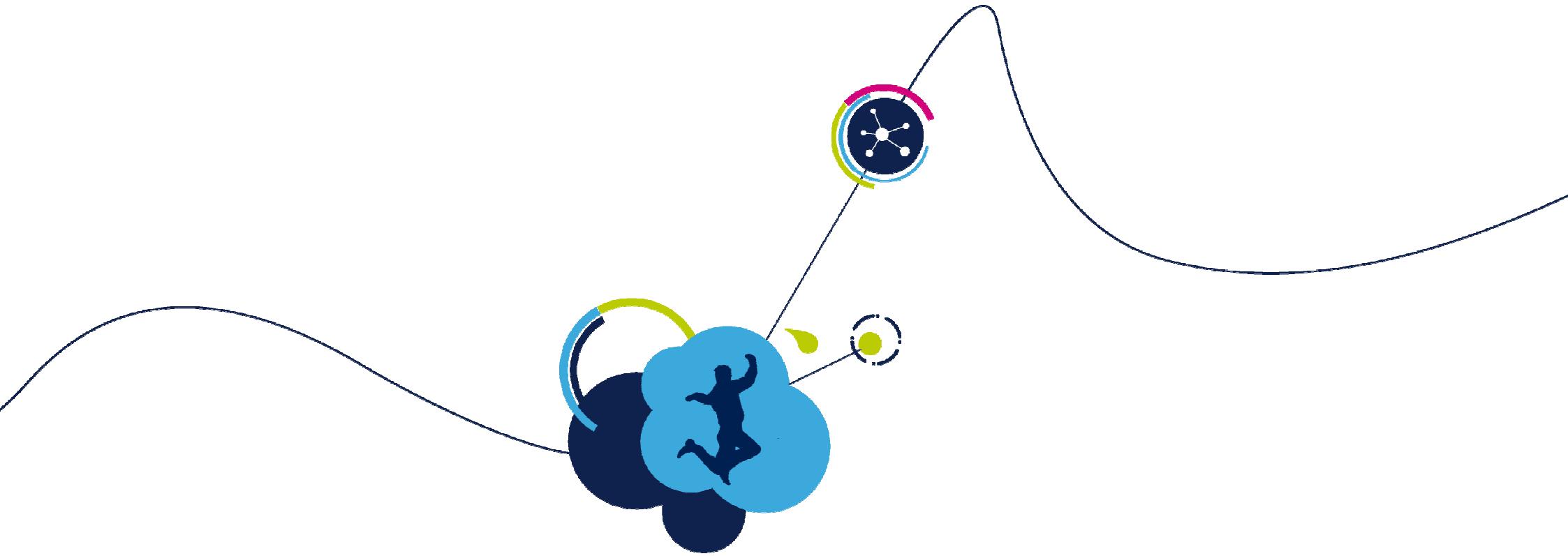


2. Click the triple-arrow GO button! (F5)



Verify your AWS connection

1. Open the TeraTerm console.
2. Wait until the DK IoT board **connects to AWS**.
3. The LED on your board will toggle every time the presenter presses the button on his board.

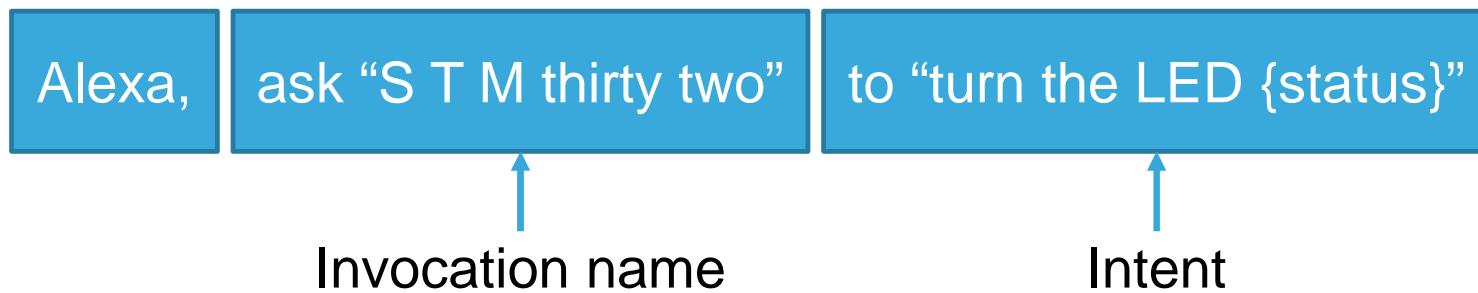


Alexa Voice Demo

What is an Alexa Skill?

290

- The Alexa skill is based on a speech recognition service that allows you to invoke a pre-specified lambda function.
- The Alexa skill uses an invocation name and intent to call the Lambda function.



What is an AWS Lambda Function

- AWS Lambda is a compute service that lets you run code (function) on AWS servers.
- AWS Lambda executes your code in response to events such as:
 - AWS IoT update to MQTT Topic
 - Alexa skill invocation
 - Other AWS services

Using Amazon Alexa (1/2)

292

- We will now use Amazon Alexa to voice control the LED on the DK IoT board.
- The presenter will ask Alexa to turn ON/OFF the LED and you will see your board responding to the command:
 - “Alexa, ask STM32 to turn the LED **ON**”
 - “Alexa, ask STM32 to turn the LED **OFF**”
 - This will trigger a Lambda function



Using Amazon Alexa (2/2)

- The Lambda function will send the following JSON message to Thing_00's topic:

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



- Since all the boards are connected to Thing_00's topic (from the previous lab), the message will be sent to your board and toggle the LED.

More Info On Alexa Skills

- Please refer to the following links to get more info on how to use Alexa to interact with IoT devices:
 - <https://developer.amazon.com/blogs/post/Tx3828JHC7O9GZ9/Using-Alexa-Skills-Kit-and-AWS-IoT-to-Voice-Control-Connected-Devices>
 - <https://developer.amazon.com/alexa-skills-kit/alexa-skill-quick-start-tutorial>
 - <https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/developing-an-alexa-skill-as-a-lambda-function>
- Also we have included the Alexa skill code, the Lambda code and a document in the following directory:

C:\STM32L4_DK_IoT_Node_Seminar\Alexa

Releasing Your Creativity

295



/STM32



@ST_World



st.com/e2e



www.st.com/stm32