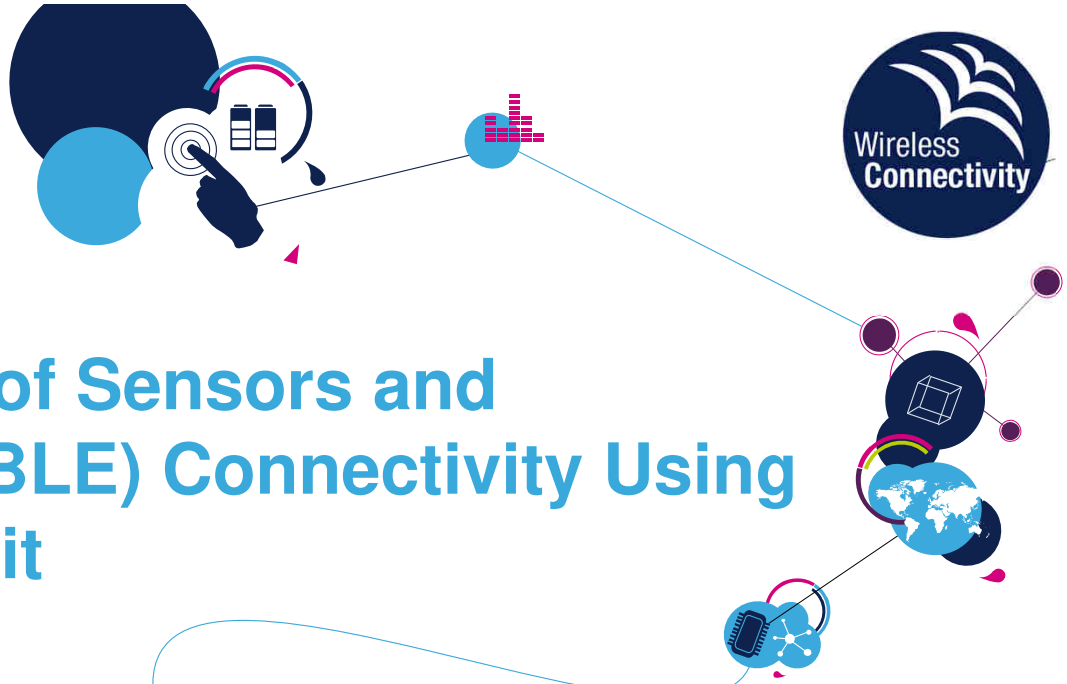


Simplify the Integration of Sensors and Bluetooth Low Energy (BLE) Connectivity Using the BlueNRG-Tile Eval Kit

Hands-on Workshop

IoT Systems Development - Connectivity

WiFi Network: **STMicro**
Password: **STMTT2019**

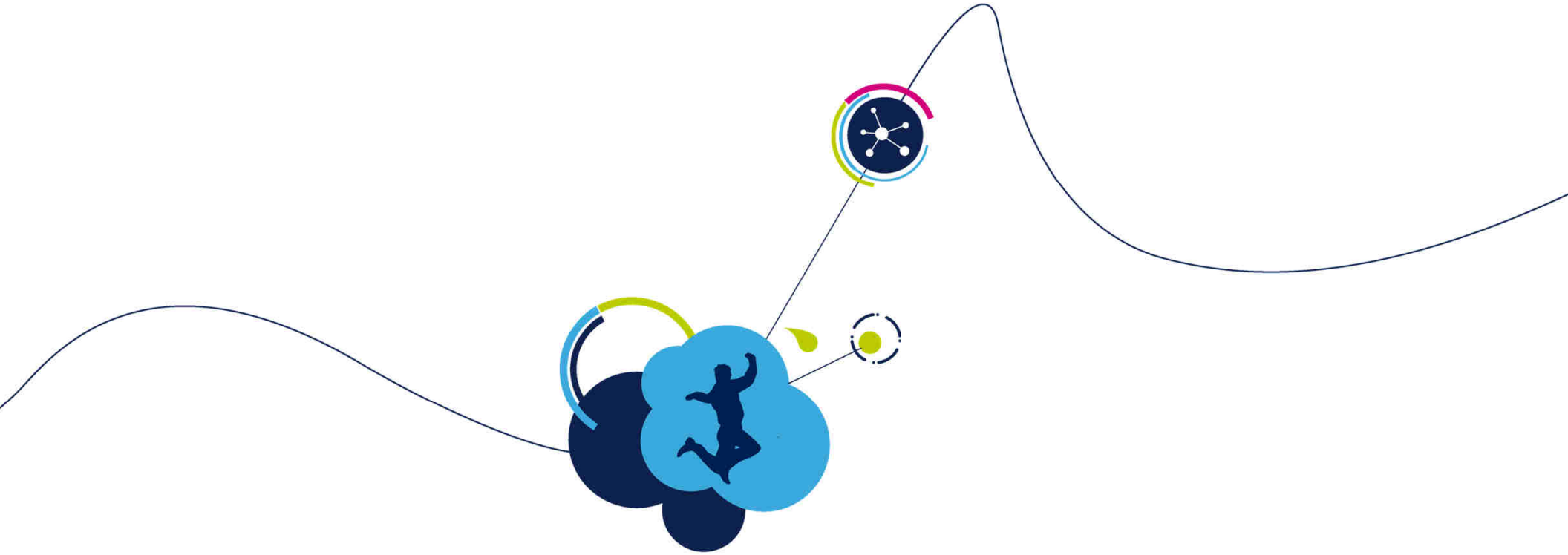
life.augmented

Technology Tour 2019
Anaheim, CA | March 26

Agenda

BlueNRG-Tile
(STEVAL-BCN002V1)

- Training Material Check/Installation Help
- ARM® Keil MDK Installation
- Introduction to Bluetooth® Low Energy
- BlueNRG-Tile Development Kit
- ARM® Keil MDK License Installation
- Lab 1: Getting Started with BlueNRG-Tile “Serial Terminal Test”
- Lab 2: Connecting to the ST BLE Sensor app
- Lab 3: LED characteristic
- Lab 4: Accelerometer embedded events detection
- Lab 5: 9-axis Acc+Gyro+Mag Sensor Data Fusion
- Lab 6: Cloud data logging on IBM Watson
- Lab 7: Bonus – Voice over BLE



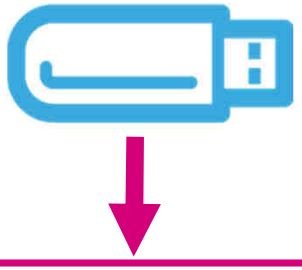
Training Material Check / Installation Help

life.augmented

BlueNRG-Tile Hands-on - LABs Preparation

ON YOUR LAPTOP

- **USB Flash drive** with relevant hands-on material
- **Unzip** the content to **C:**
  - **Mandatory: installation folder** has to be ***C:\BlueNRG_Tile_HandsOn***



Please return it at the end of the workshop

ON YOUR PHONE

- **Install "ST BLE Sensor" app**

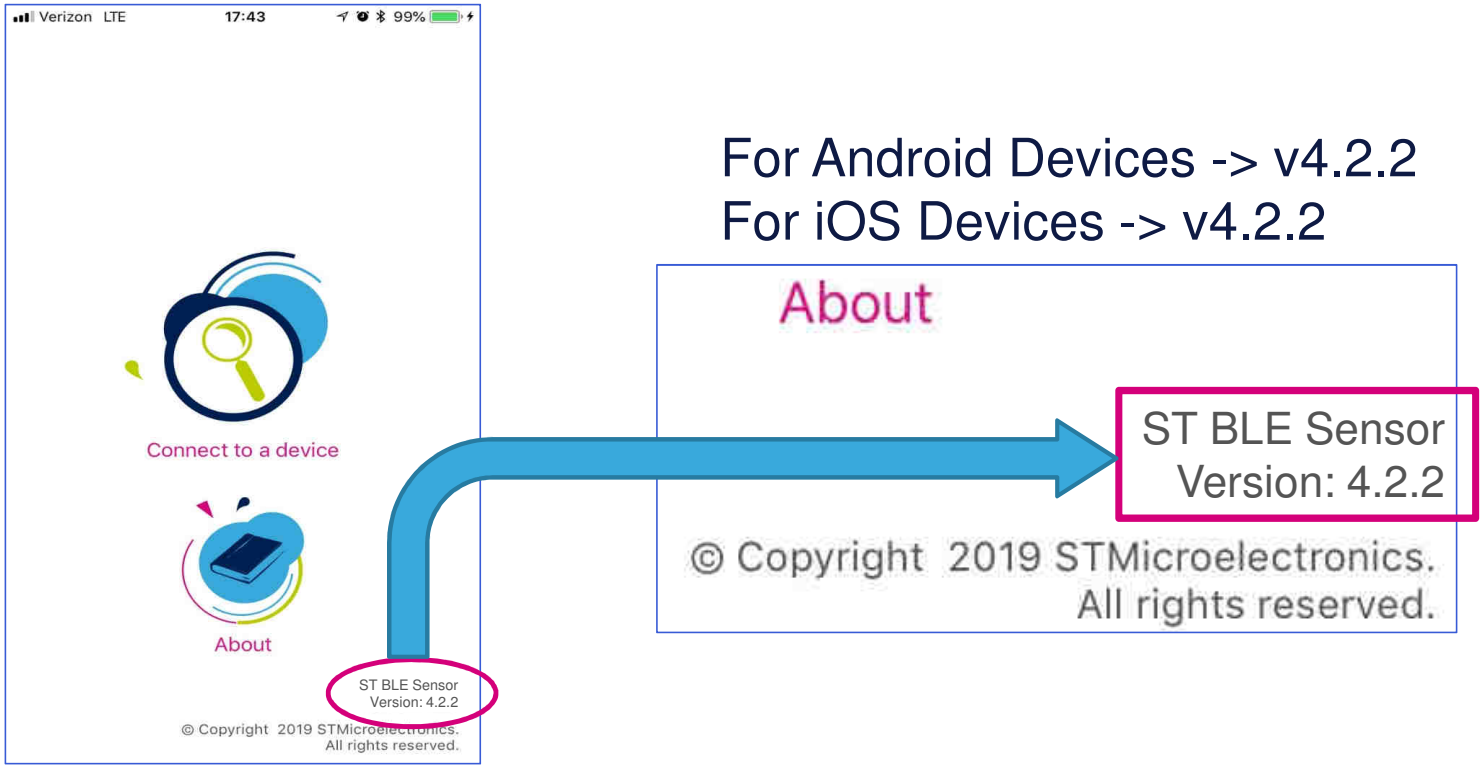
Download on the App Store

GET IT ON Google Play

life.augmented

if you already have the ST BLE Sensor app installed…

- You need to upgrade the "**ST BLE Sensor**" app to the **latest version available on Google Play and Apple iOS app stores**

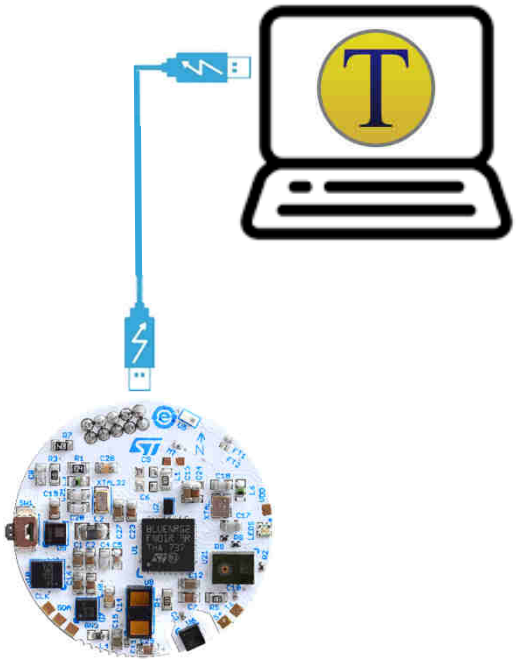


life.augmented

Today's workout!

LAB1

- **Connect** BlueNRG-Tile (STEVAL-BCN002V1) to PC using USB
- **Run** **Tera Term** virtual serial terminal



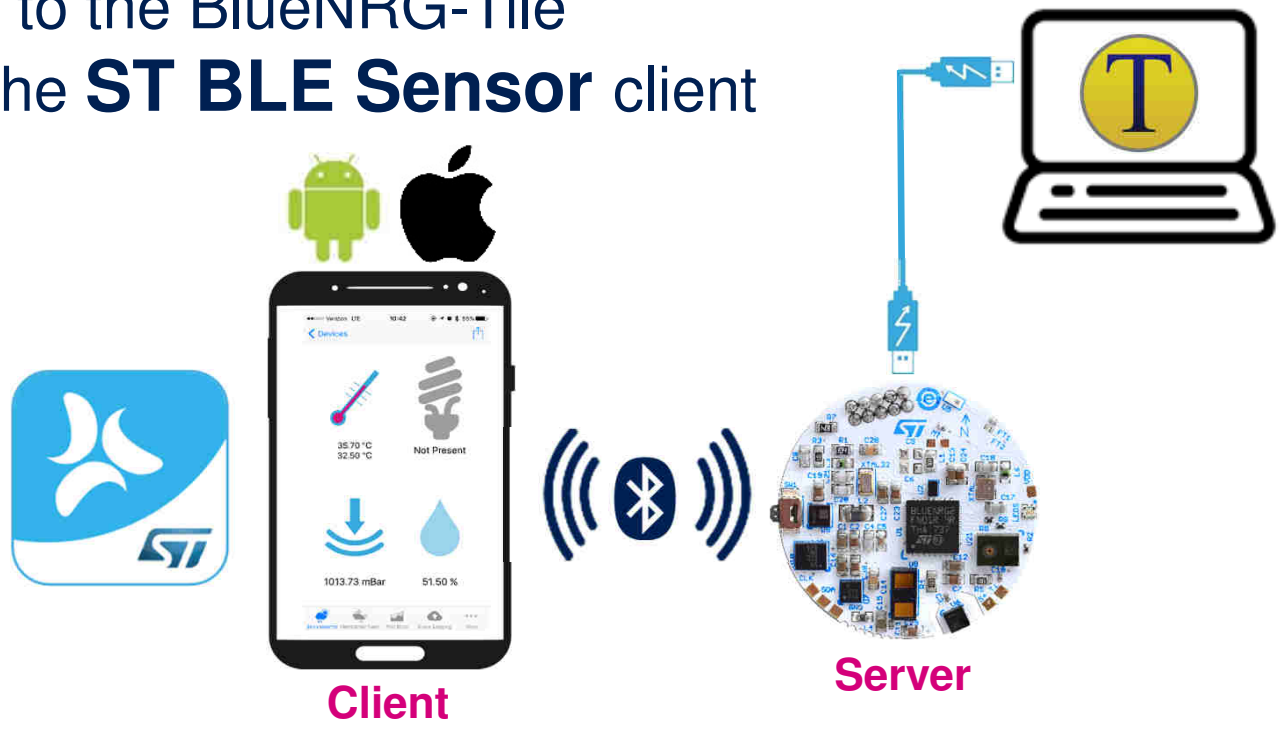
life.augmented

Today's workout!

LAB2

Start a BLE Connection

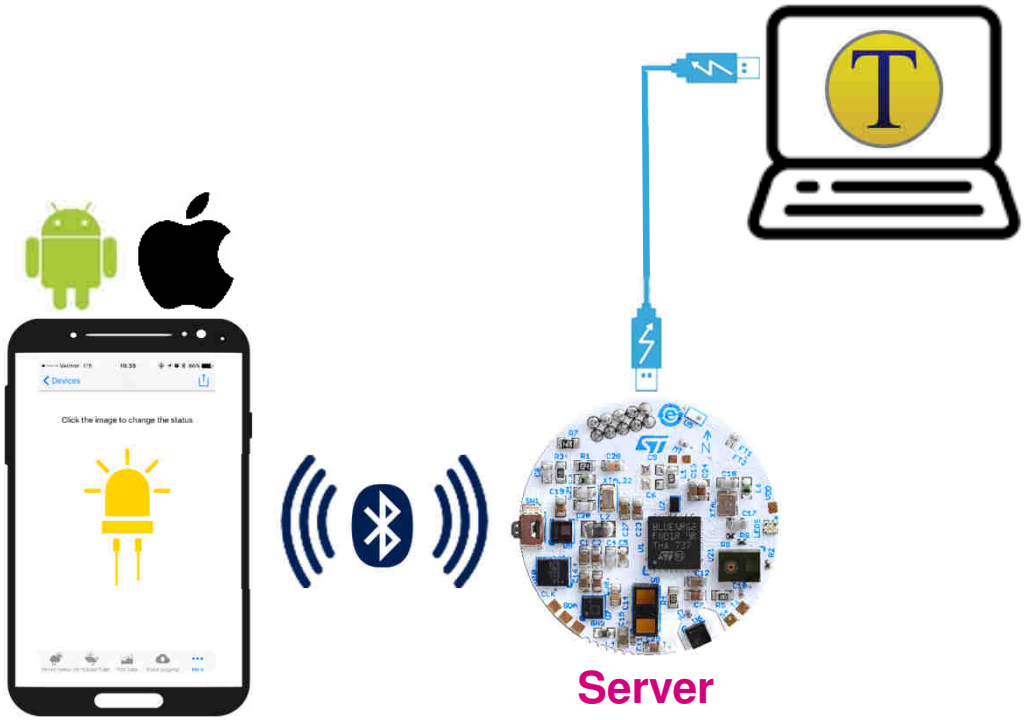
- **Run ST BLE Sensor** and discover ***YOUR OWN*** BlueNRG-Tile
- **Connect** your phone to the BlueNRG-Tile
- **Get** Sensor data on the **ST BLE Sensor** client



Today's workout!

LAB3

Add LED characteristic and remotely toggle the LED

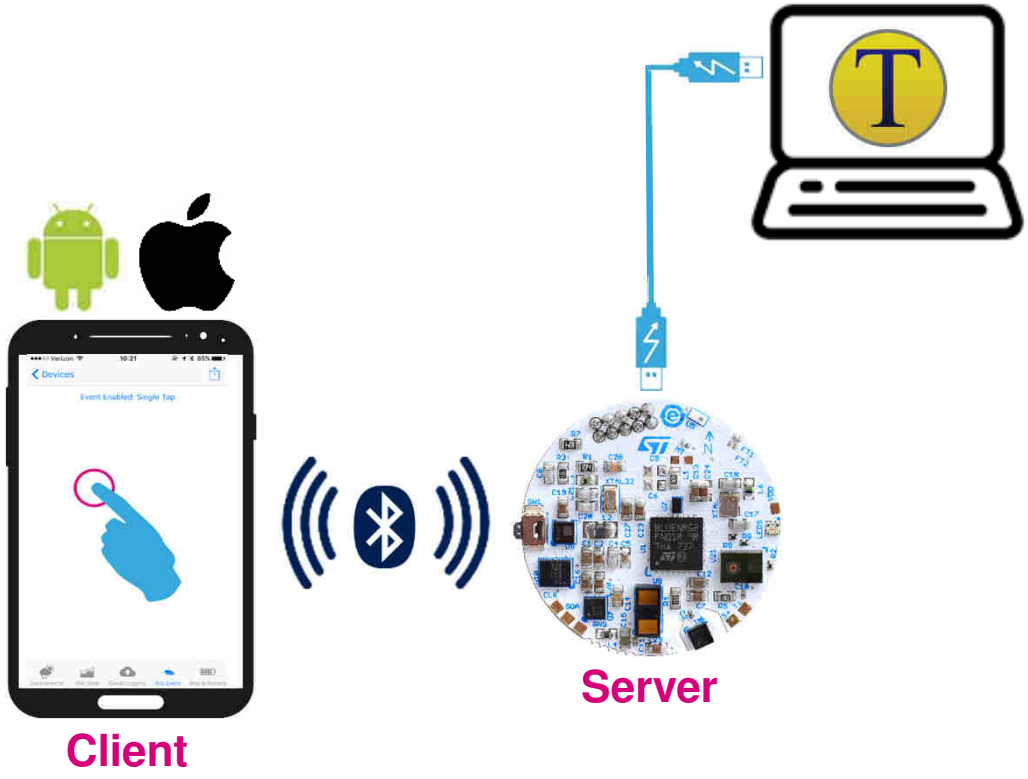


life.augmented

Today's workout!

LAB4

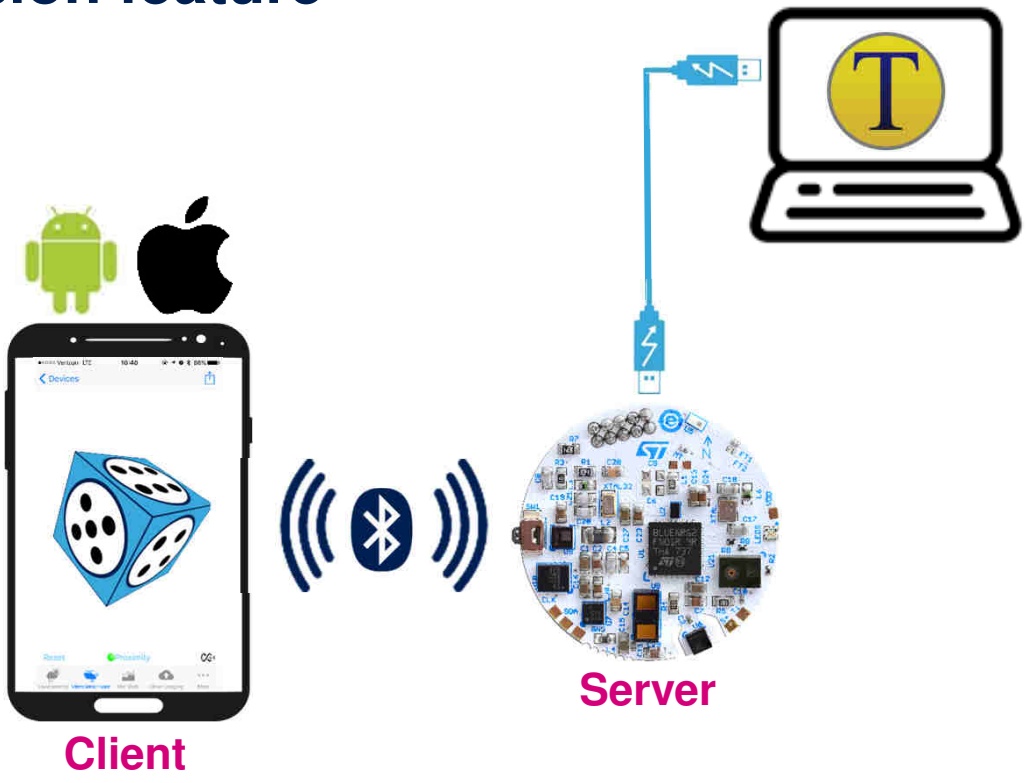
Add Accelerometer embedded **events detection feature**



Today's workout!

LAB5

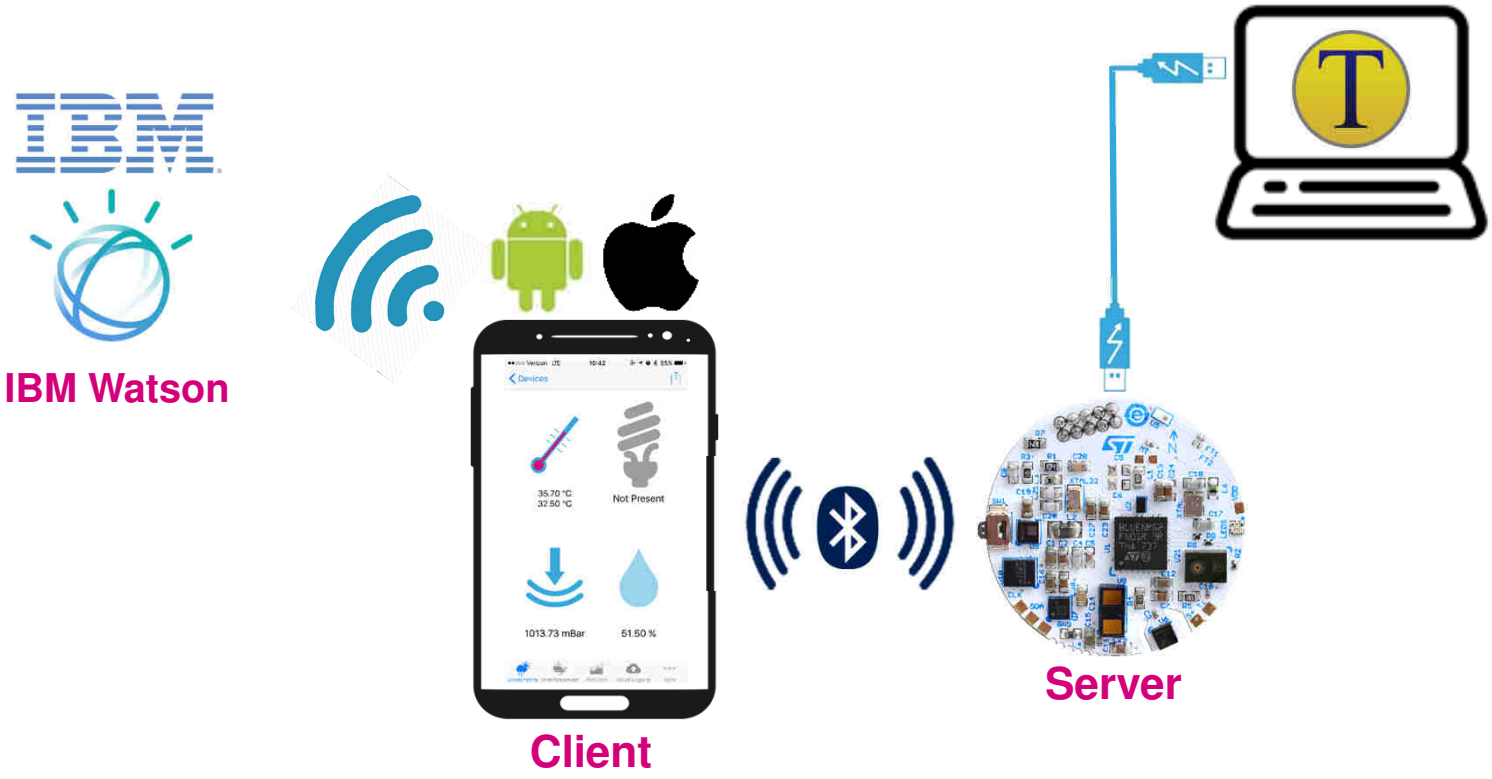
Add 9-axis **Sensor Data Fusion feature**



Today's workout!

LAB6

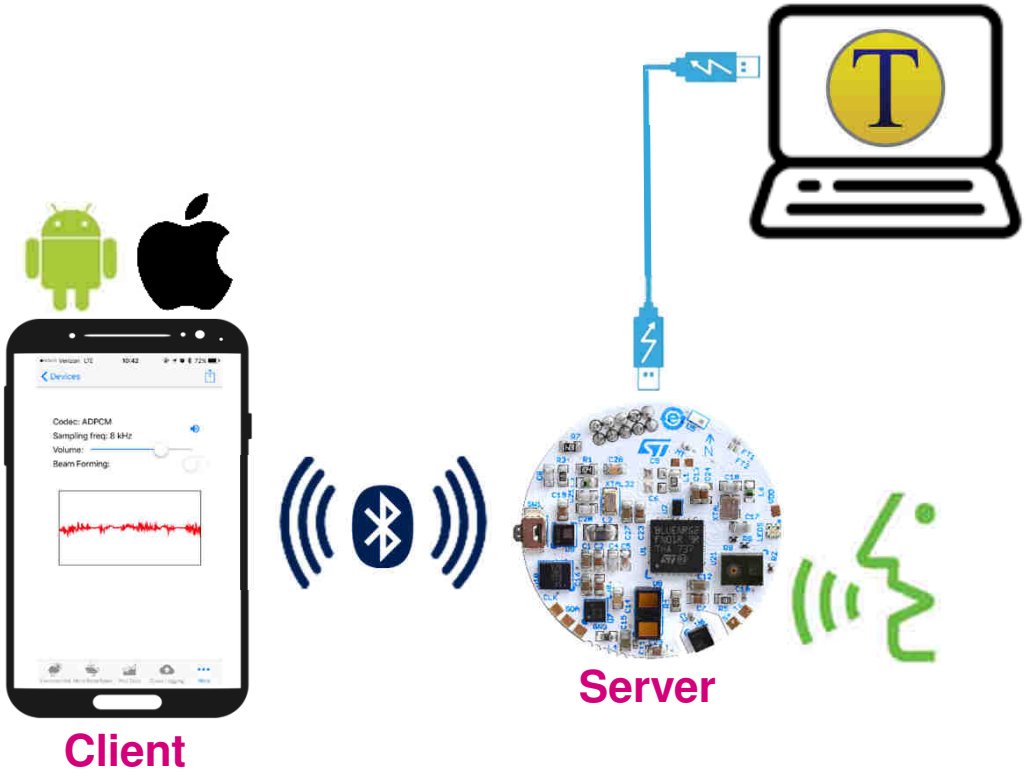
Enable Cloud sensor logging on IBM Watson



Today's workout!

LAB7

Bonus lab: **add Voice over BLE feature**



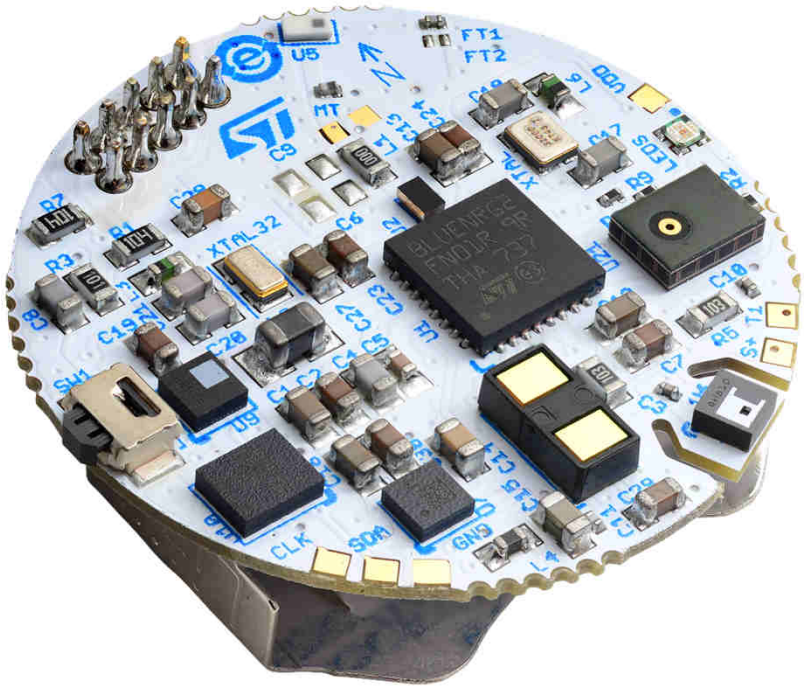
It's a class: remember that…

- **OUR GOAL** is for **ALL OF YOU to successfully complete all the Labs**
- Each lab has a **step-by-step procedure**. We need to go through each step!
- It's not a race. If you're ahead, please help your neighbor… ☺

And there is a prize…

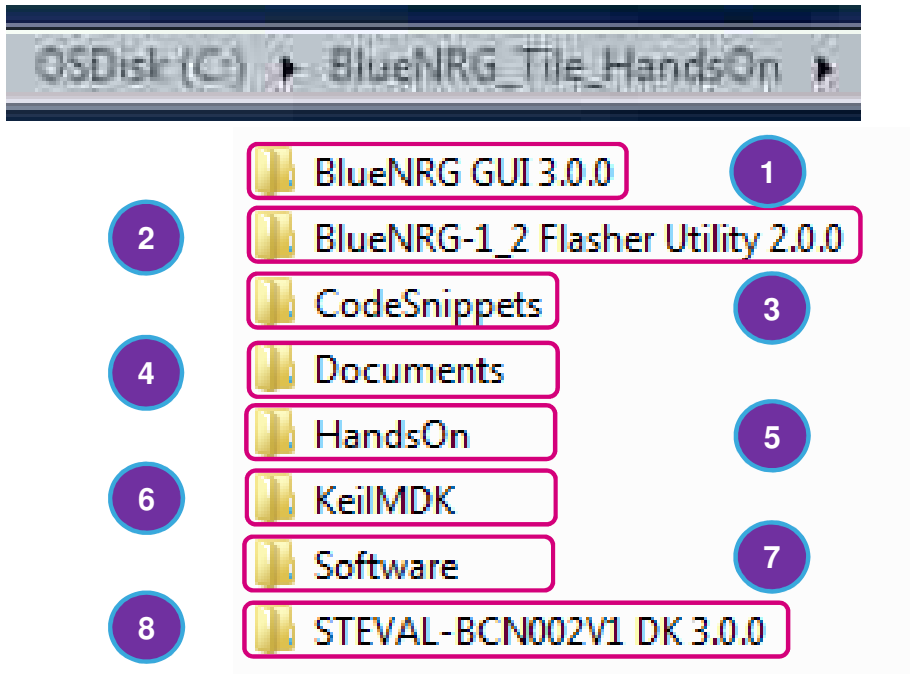
At the end of the workshop each of you will receive a

free BlueNRG-Tile eval kit (STEVAL-BCN002V1B)



life.augmented

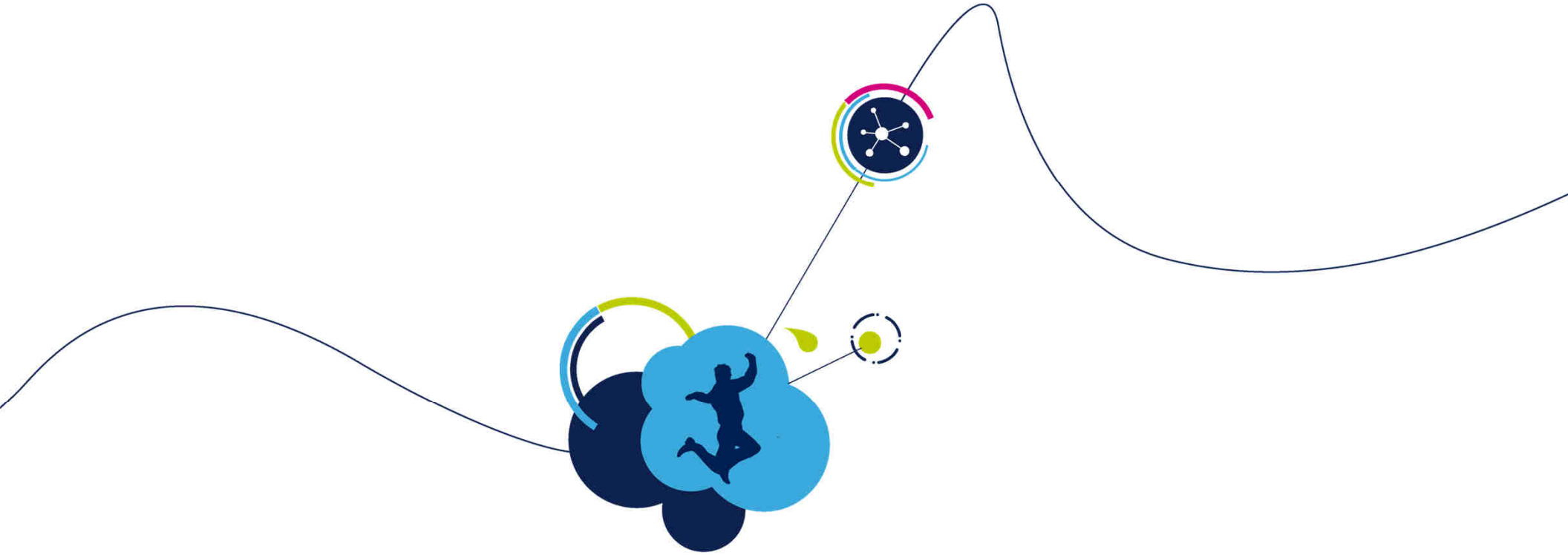
Hands On Directory Content



1. BlueNRG Graphical User Interface
2. UART Flasher Application
3. Code Snippets text file
4. Documents: DS, User Manuals and App Notes
5. Hands on Labs presentation and binary
6. Keil setup files
7. Software Utilities
8. BlueNRG-Tile DK V 3.0.0

Open "**BlueNRG_Tile_HandsOn.pdf**" presentation

life.augmented



ARM Keil MDK Installation

life.augmented

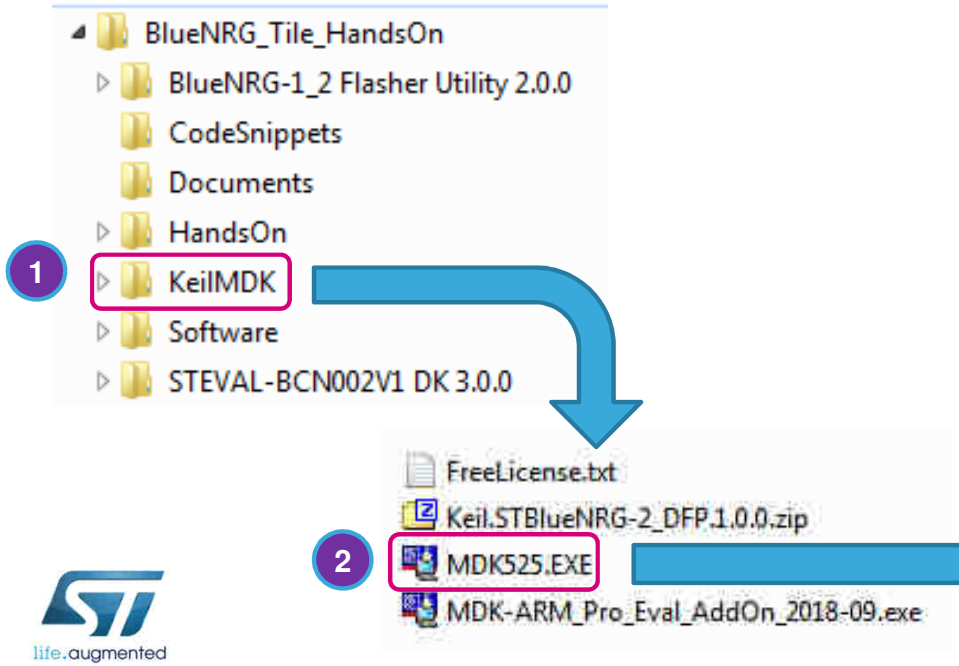
What is Arm Keil MDK?

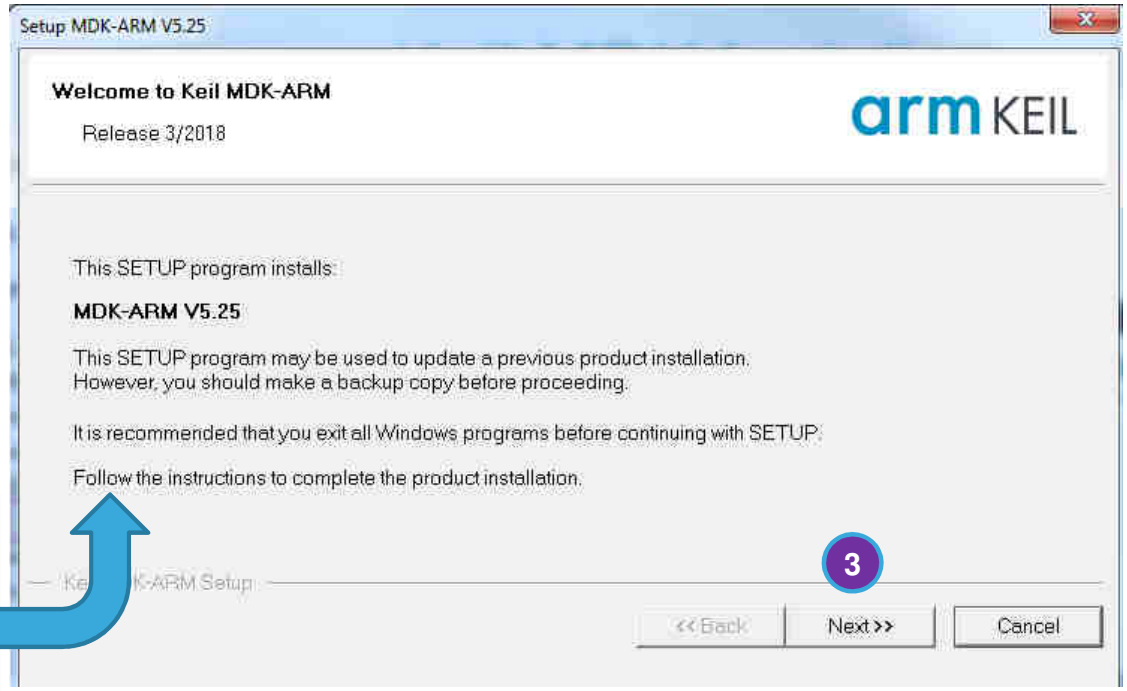
- **MDK**: **M**icrocontroller **D**evelopment **K**it
  - Complete software development environment from Arm for a wide range of Arm Cortex-M based microcontroller devices.
  - MDK includes the **µVision IDE**, **debugger**, and **Arm C/C++ compiler**

- A **FREE** license is available for ST BlueNRG-2 device

Keil MDK Installation 1/6

1. **Go to the folder** **C:\BlueNRG_Tile_HandsOn\KeilMDK**
2. **Double click** on **MDK525.EXE**
3. **Click on Next** in the Setup window





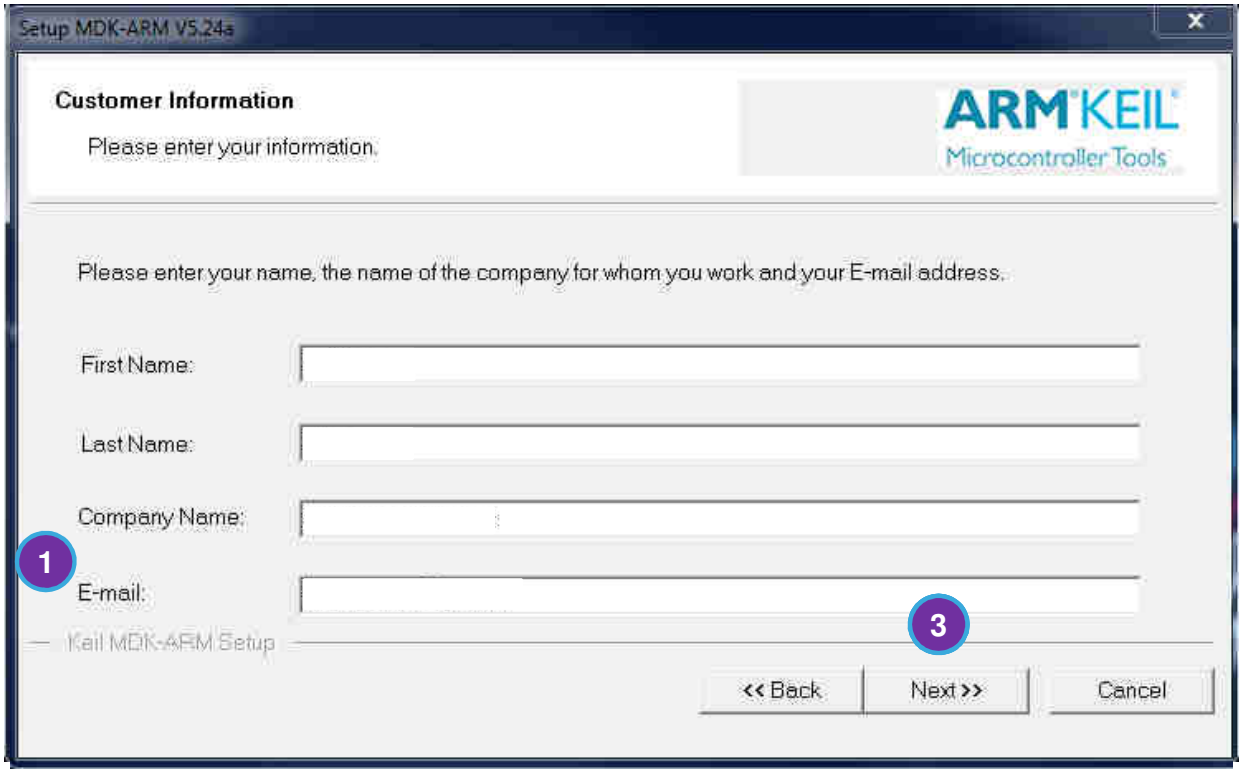
life.augmented

Keil MDK Installation 2/6

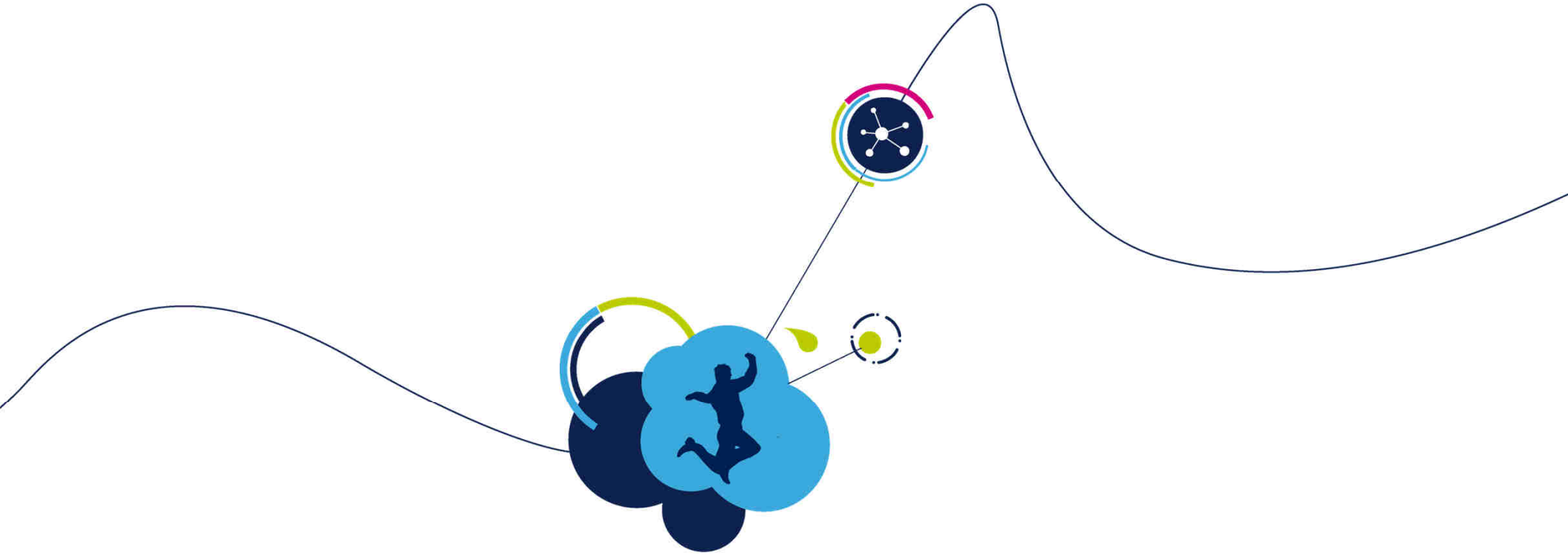
1. **Click** on **Agree** and then **Next**
2. **Click** on **Next**
3. **Fill in your info** and **click** on **Next**

The installation starts.

TO BE CONTINUED IN A FEW MINUTES…



life.augmented

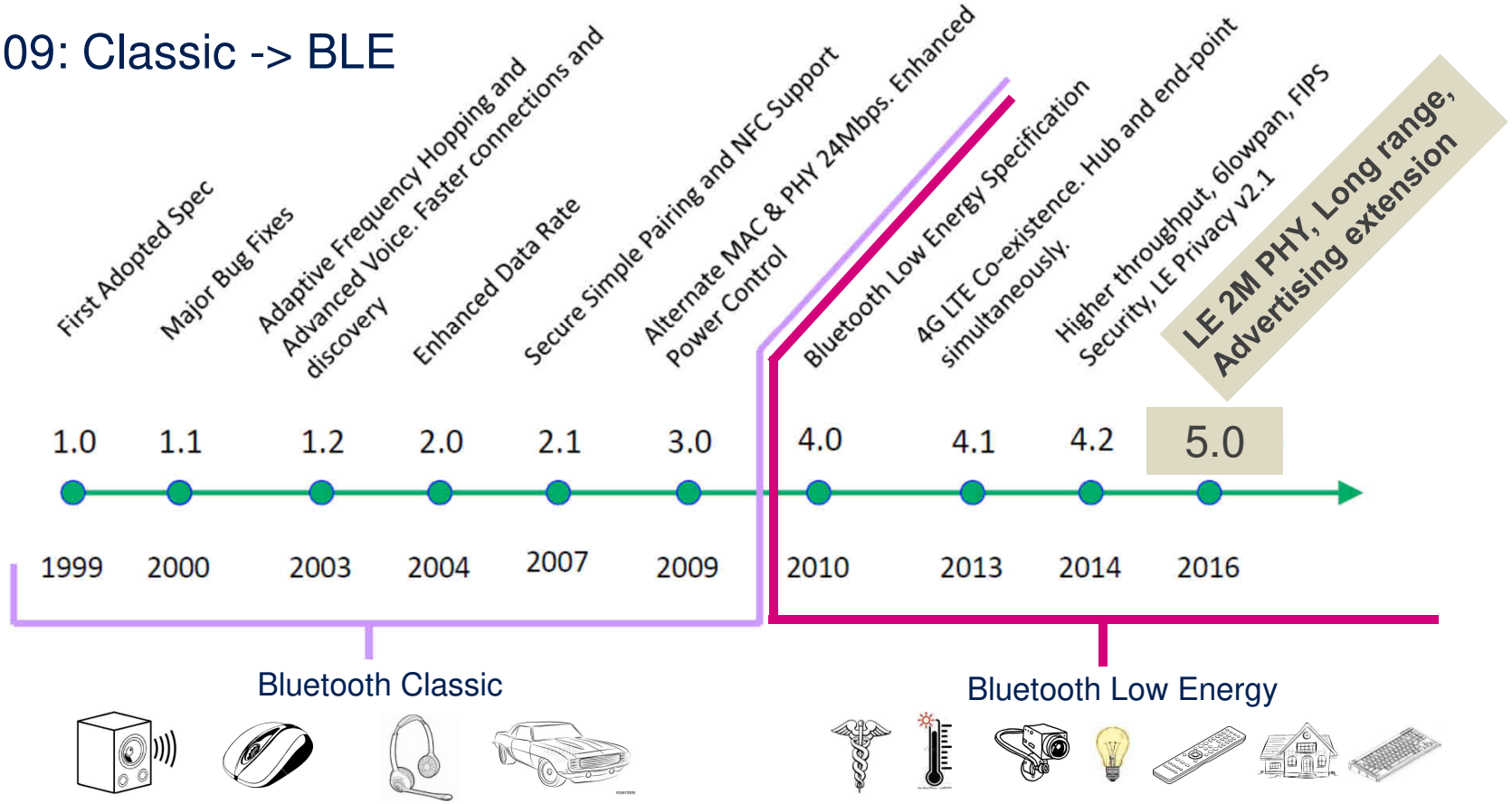


Introduction to Bluetooth Low Energy

life.augmented

Bluetooth® Evolution

- 20 Years in the market
- 2009: Classic -> BLE

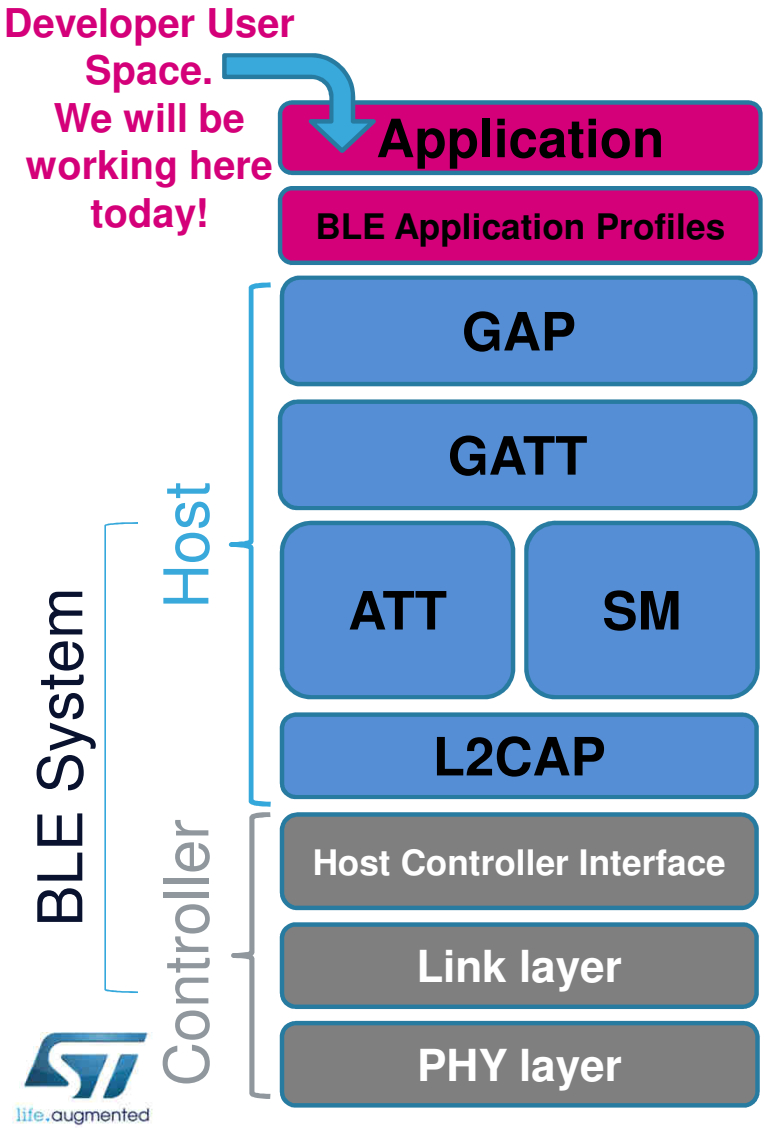


life.augmented

Bluetooth low energy (LE): Designed for Success

- **Lower power** than Bluetooth “Classic” (or Bluetooth 3.0)
  - **Lower duty cycle**
  - **Faster connection**
  - **Lower peak power** (relaxed RF parameters)
- Compatible with all major platforms (iOS, Android, Windows, Linux)
- **Multiple network topologies**:
  - **Point to point**: single master connects to single slave (and each slave can connect to 2 master)
  - **Star**: Multiple slaves connected to a single master
  - **Mesh**: introduced in the 2017
- State of the Art **encryption**, **security** including privacy/authentication

Protocol Stack

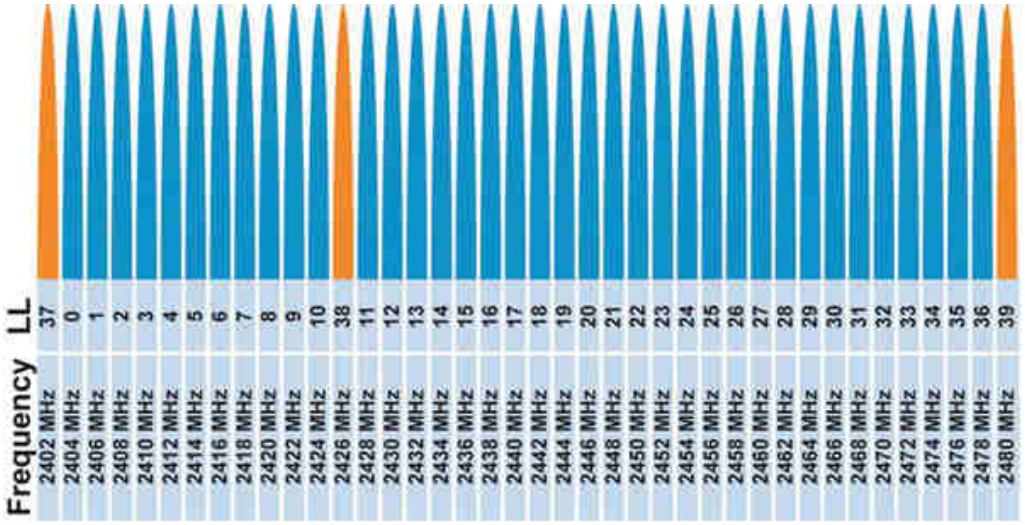


- **Generic Acces Profile**
  - How a device can discover and connect with one another
- **Generic Attribute Profile**
  - How services, characteristics can be discovered and then used
- **Attribute Protocol**
  - Protocol for discovering, reading, and writing attributes on a peer
- **Security Manager**
  - Handles the secure communication
- **L2CAP**
  - Protocol multiplexer. Segmentation and reassembly of packets
- **HCI**
  - Interface between Host and Controller
- **Link Layer**
  - Handles packets, channels, advertising, scanning & connections
- **Physical Layer**
  - Transmits/receives bits

Protocol Stack: PHY

- A **BLE Radio** is a **2.4 GHz ISM Band Transceiver**

- **40 RF Channels**, **2 MHz Channel Spacing**. Two types of channels:
  - **Advertising channels** (**3 - orange**) for Advertising Packets. Used for Discoverability and for Broadcasting/Observing
  - **Data Channels** (**37 - blue**) for Data Packets. Used to send application data in Connection

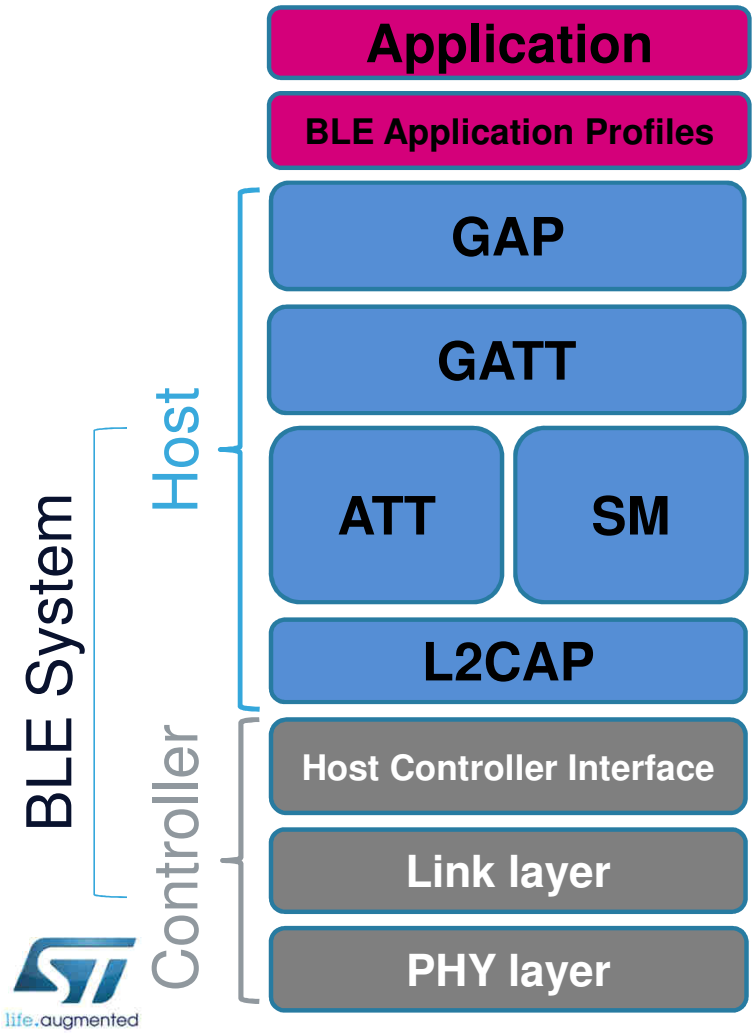


Source : Bluetooth® SIG

- **GFSK Modulation**
  - BT = 0.5
  - Modulation Index = 0.5
  - “pulse shaping” Gaussian filter “smooths” transitions from zero to one reduces spectral width

life.augmented

Protocol Stack - GAP



- Who controls the radio network?
  - **GAP (Generic Access Profile)**
  - Defines 4 roles
    - **Broadcaster**
    - **Observer**
    - **Peripheral**
    - **Central**

Peripheral

Connection

I'm looking for John

Central

Peripheral is in Advertising mode.
Sends Advertising packets.

My name is John
My capabilities are X, Y, Z, etc…

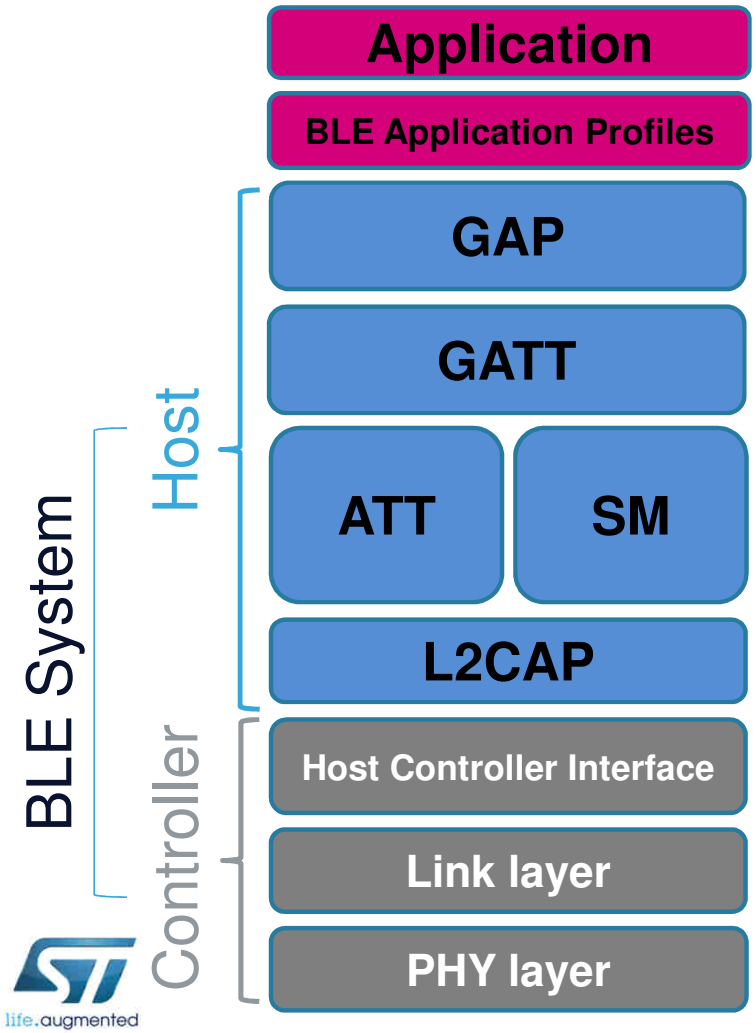
Advertising **packets consists of multiple information units known as AD types**.
E.g.: Local Name, TX Output Power, Manufacturer Specific info, etc…

Central is **Scanning**.
Look for known peripherals by *listening* to advertising packets.

As the John peripheral is found, it initiates a BLE connection.

life.augmented

Protocol Stack - GATT

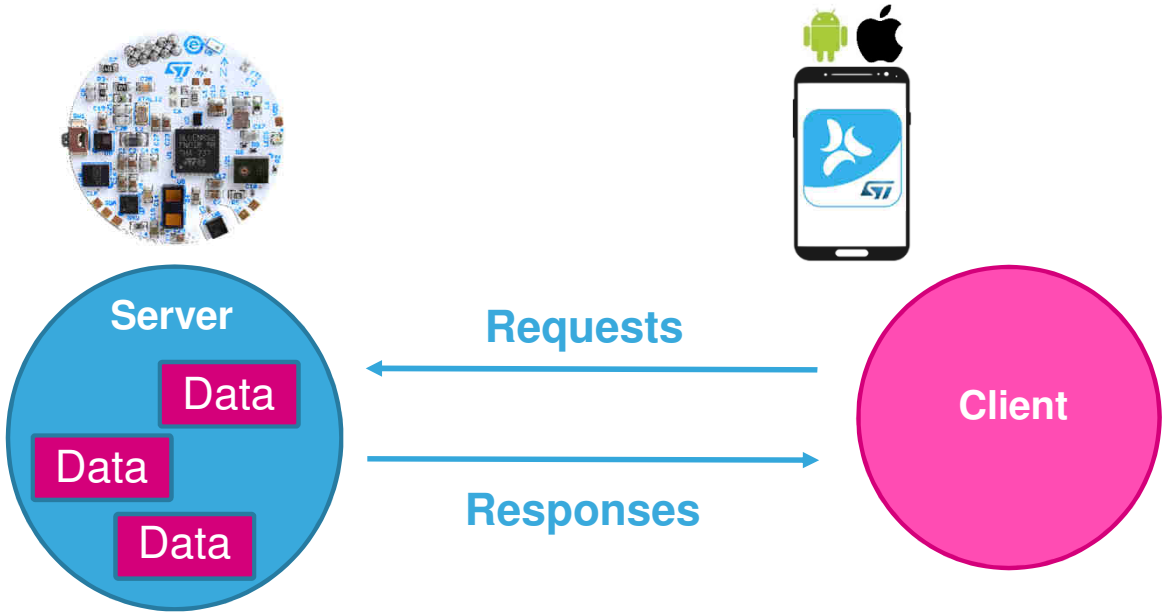


- Who controls the data flow?
  - **GATT (Generic Attribute Profile)**
  - Defines 2 roles
    - **Server**
    - **Client**

life.augmented

GATT

Server contains the data.
Receives requests, executes, responds.
Can indicate value



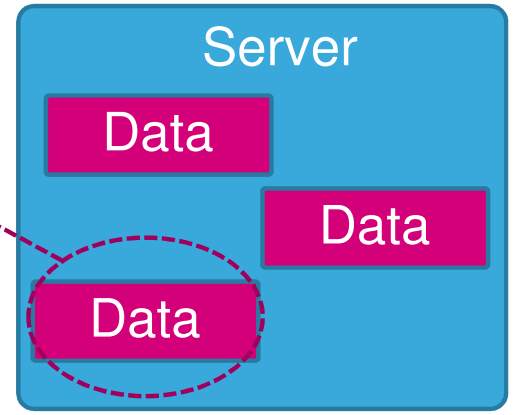
Client talks to the server.
Sends requests, wait for response.
Can confirm indications

Protocol PDU Type	Sent by	Description
Request	Client	Client requests something from server (it always causes a response)
Response	Server	Server sends response to a request from a client
Command	Client	Client commands something to server (no response)
Notification	Server	Server notifies client of new value (no confirmation)
Indication	Server	Server indicates to client new value (it always causes a confirmation)
Confirmation	Client	Confirmation to an indication

life.augmented

Attributes

- **Data are organized** and exposed as **attributes**



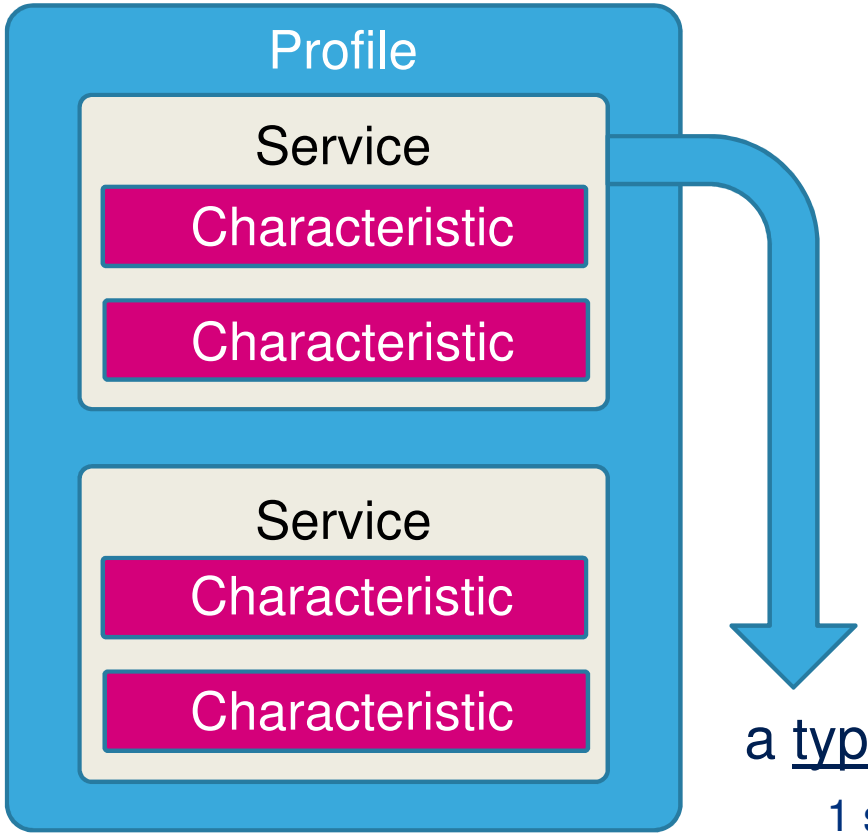
Each **Data** element in the Server is called **Attribute**

- Each attribute has:
  - A **handle**: it identifies the attribute on the server
  - A **type** (defined by Universal Unique ID - **UUID**): what the value means
    - 16-bit UUID: pre-defined by Bluetooth SIG
    - 128-bit UUID: Vendor Specific identifiers
  - A **value** (0 to 512 octets)
- Example

Handle	Type	Value
0x0009	«Device Name»	“Temperature Sensor”
0x0022	«Battery State»	0x04
0x0098	«Temperature»	0x0802

Source : Bluetooth® SIG

GATT Profile



- A GATT Profile defines how attributes are organized and how the application can access them.
- Attributes are organized in Services and Characteristics

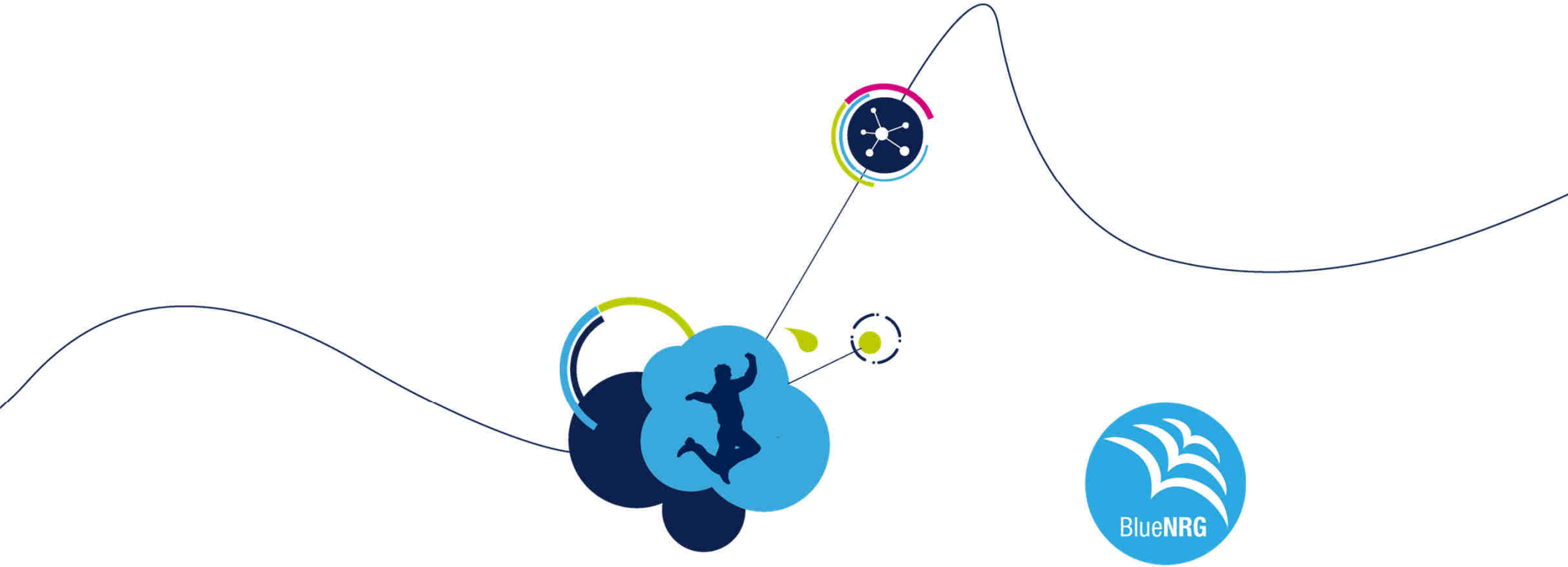
a typical example:

1 service: “ARG” *(Angular Rate and Gravity)*

2 characteristics: “Gyro”, “Acc”

Values: [0,-1,+2], [-10,+15,+950]

life.augmented

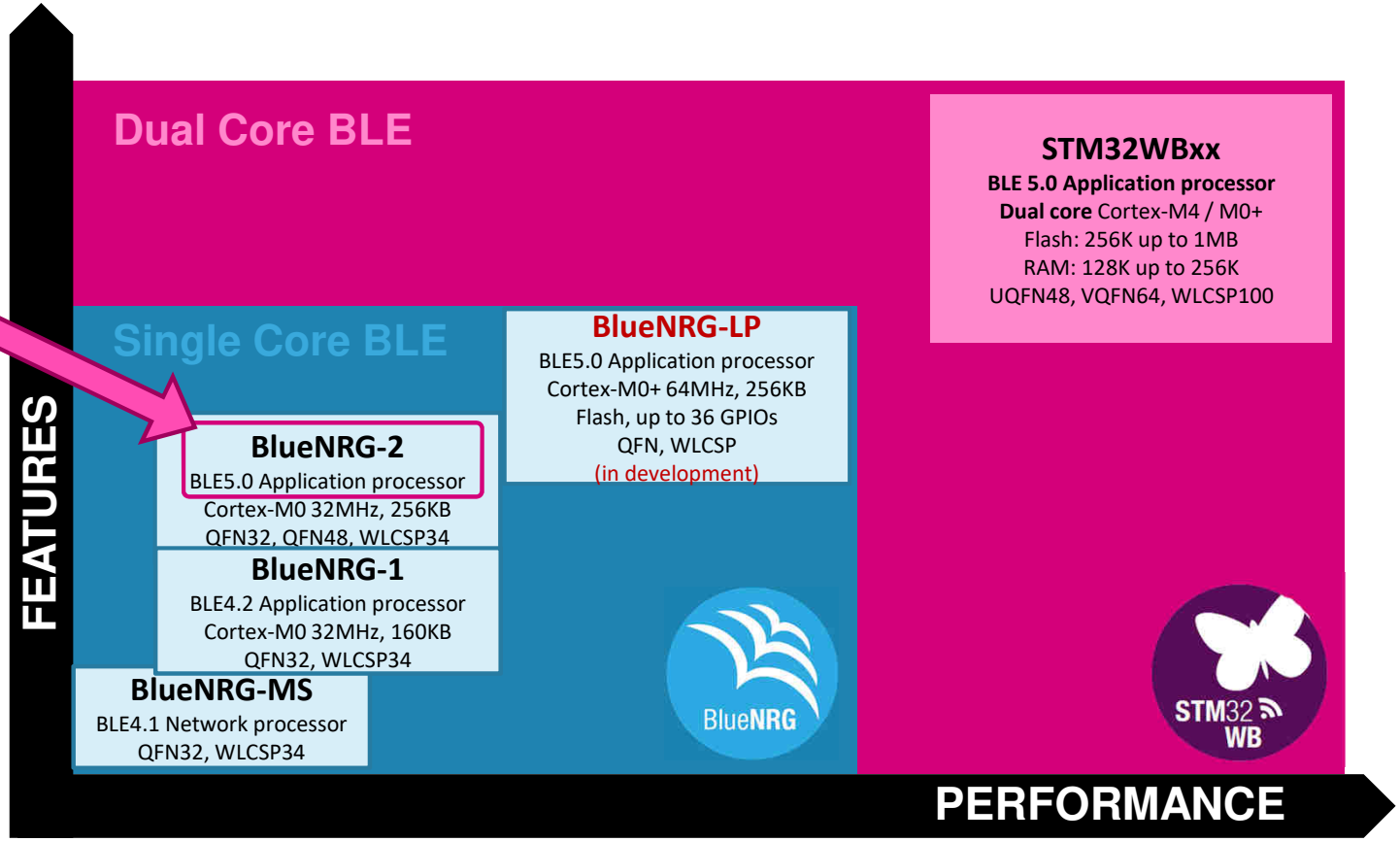


ST BLE devices

life.augmented

ST BLE Roadmap

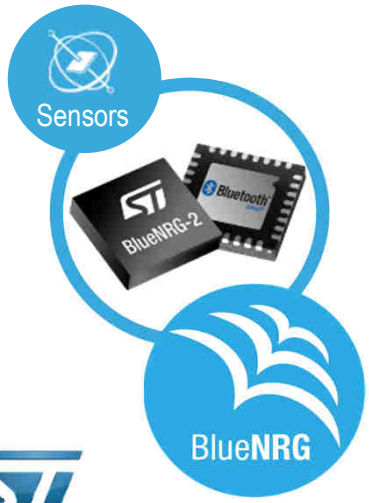
TODAY'S
Hands on



life.augmented

BlueNRG-2 SoC at a glance

The lowest power consumption	Processing power on demand	Flexible memory architecture	Optimized BLE Radio stack	Maximum security
>3 years lifetime on CR2032(*) 59 µA/MHz 0.9 µA sleep	Low-power architecture, Cortex-M0 @ 32 MHz	256 KB eFLASH 24 KB ULL SRAM (with full SRAM data retention)	70 kB FLASH 8 kB RAM 0.9uA with full RAM retention	ECC-256 AES-128 Factory UID Secure KEY



Seamless connection with SENSORS

Robust and Reliable BLE Link

Small form factor

5

Bluetooth 5.0 certified

Privacy 1.2 and secure connection 4.2

data

Faster and more reliable data transfer

(*) Based on the average current consumption in connection mode (7.059 µA, connection interval 1000 ms)

life.augmented

Bluetooth LE programmable processor

Bluetooth LE
5.0
ULP radio

RF
PHY
+8 dBm

BLE 5.0

ARM
Cortex M0
32 MHz

Flash
256 KB

SRAM
24 KB

PROGRAMMABLE
MCU
with eFLASH
to host application code

Application and Profiles

BLE Protocol stack

Advanced High-Performance Bus (AHB)

DMA

PROGRAMMABLE
PERIPHERALS

PDM Interface

Decimation filter

I2C

RTC (timers)

ADC 10 bit
4 channels

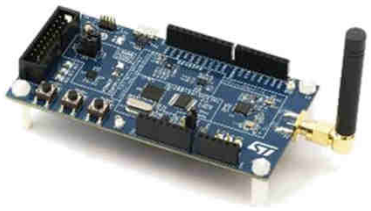
UART

SPI

DC/DC
LDO

GPIO x 26

PROGRAMMABLE
Digital I/O

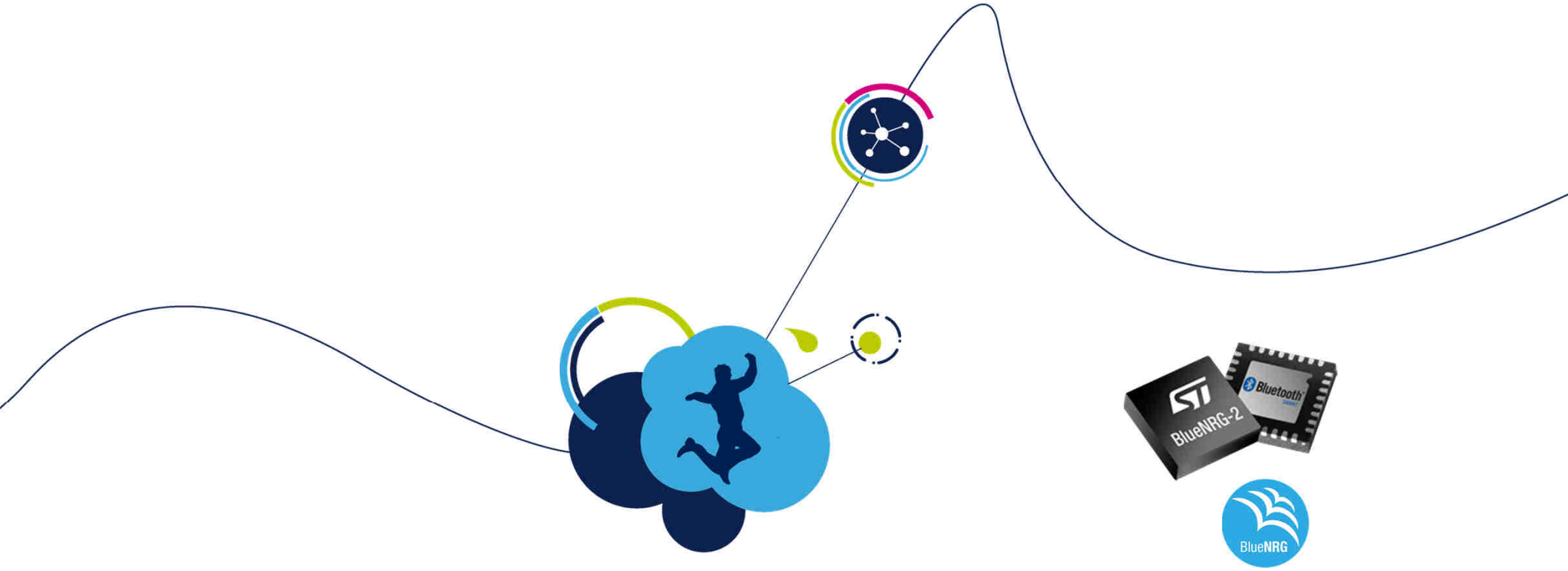


life.augmented

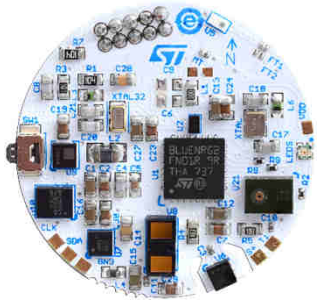
IAR SYSTEMS

KEIL™ Tools by ARM

atollic

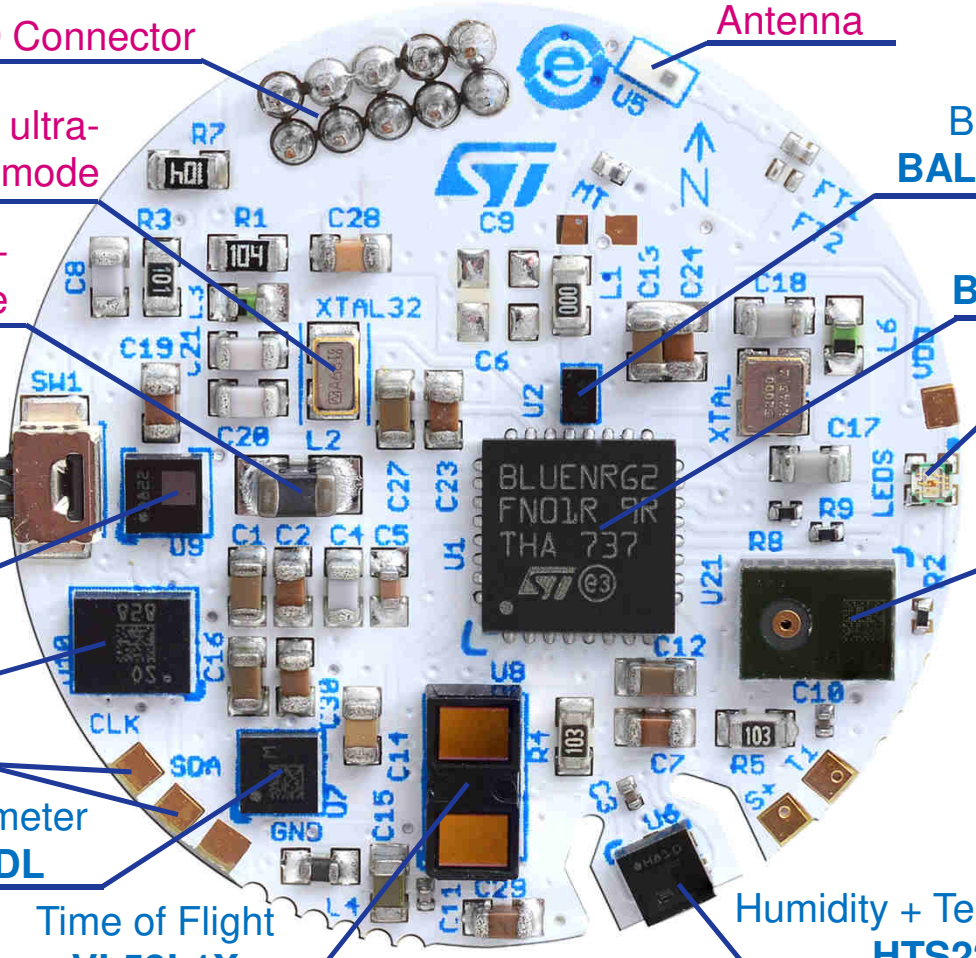


BlueNRG-Tile Development Kit



life.augmented

STEVAL-BCN002V1



SWD Connector

32KHz XTAL to enable ultra-low-power BLE sleep mode

L2 bead to enable low-power BLE active mode

User button

Pressure Sensor
LPS22HH

Accelerometer + Gyro
LSM6DSO

I2C for expansion

Magnetometer
LIS2MDL

Time of Flight
VL53L1X

Antenna

Balun
BALF-NRG

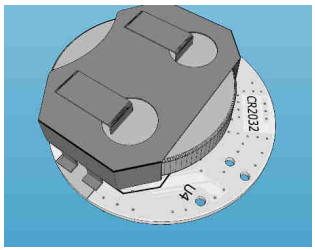
BLE SoC
BlueNRG-232

RGB LED

Microphone
MP34DT05-A

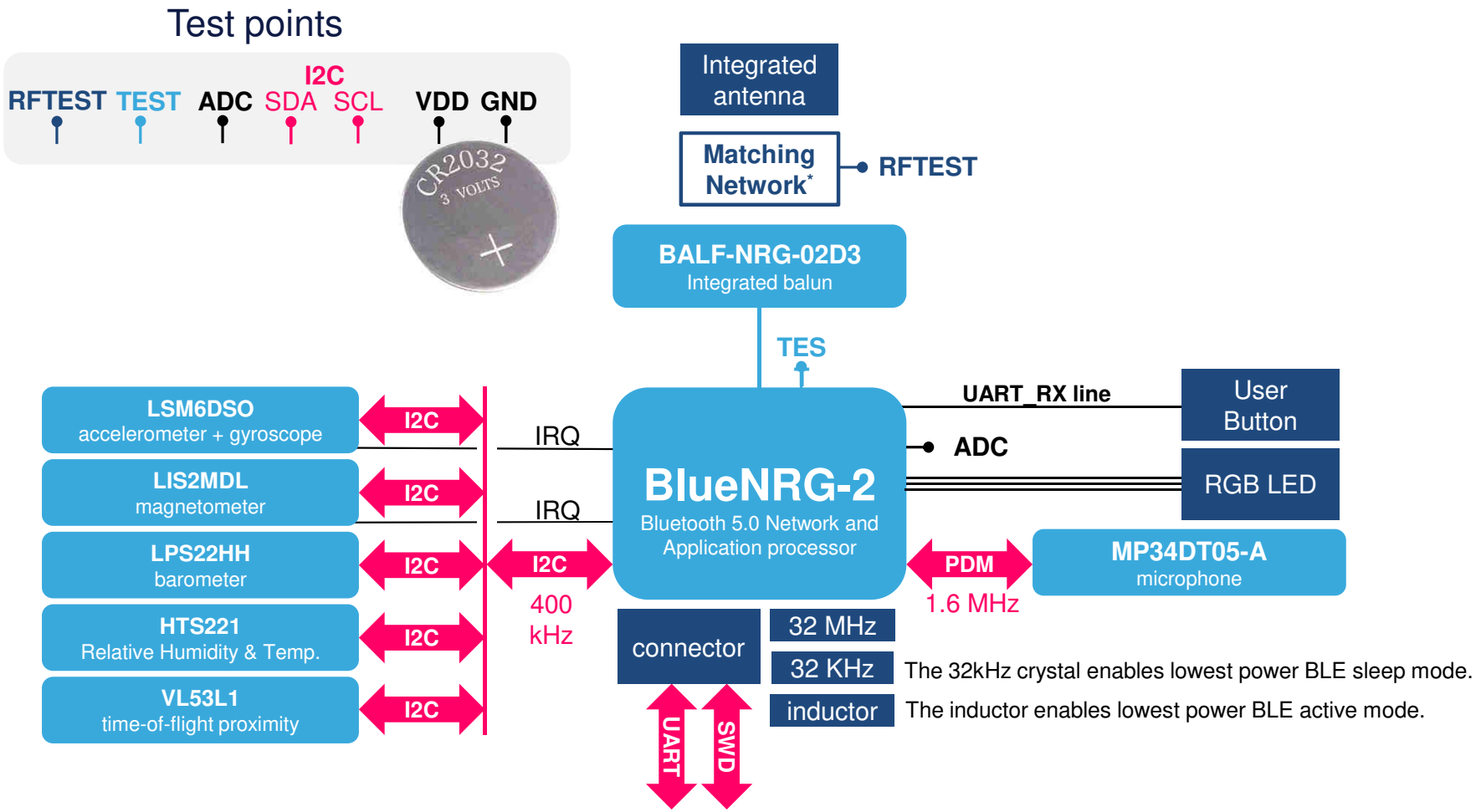
Humidity + Temperature
HTS221

Bottom view



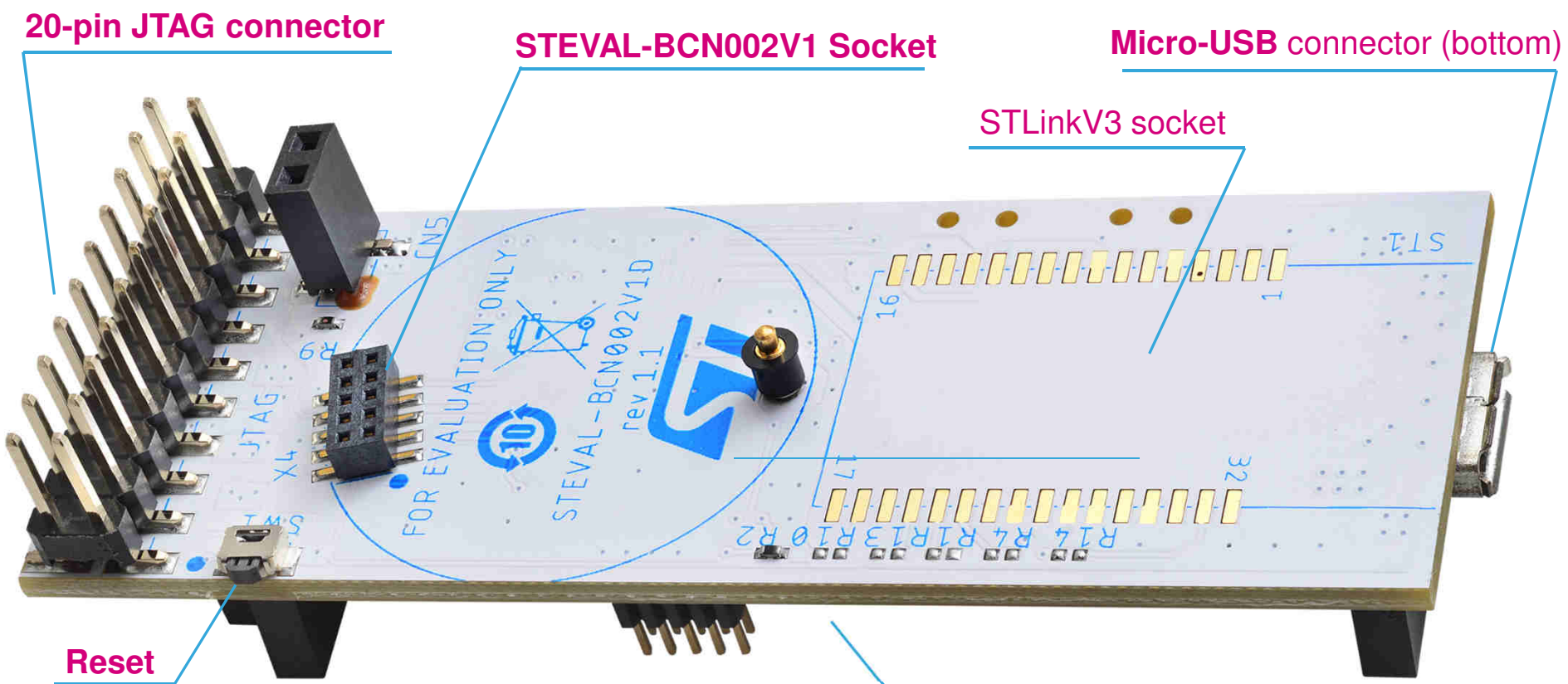
life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

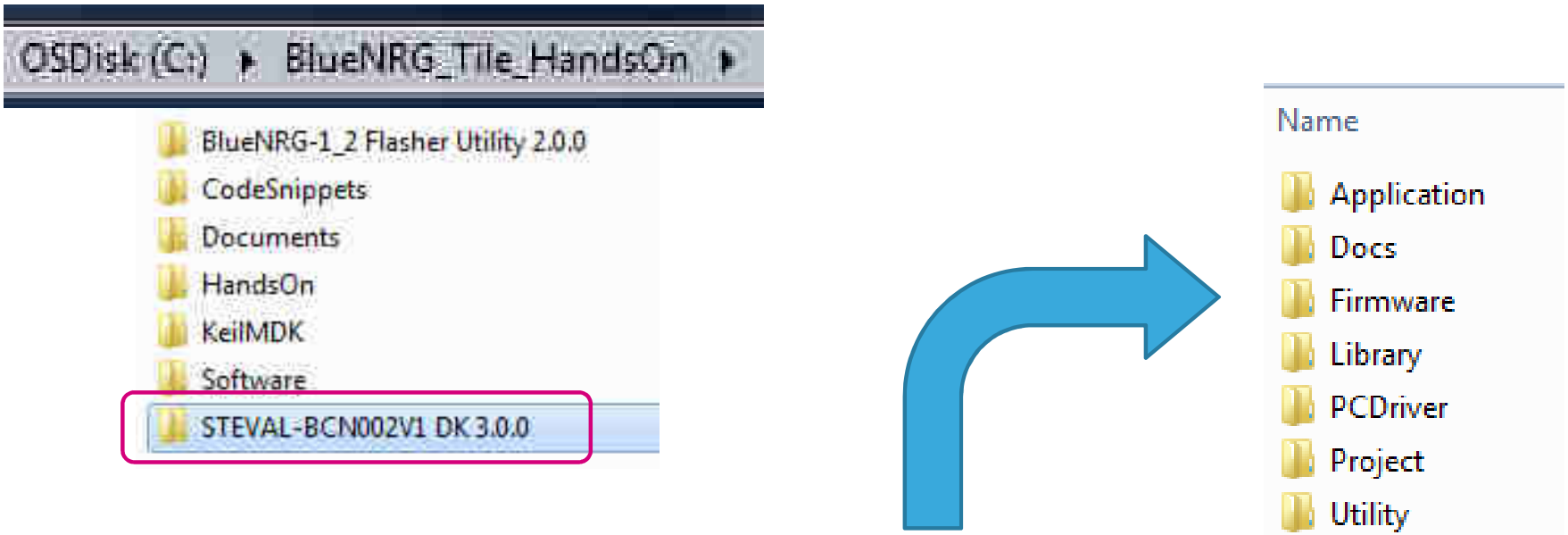
STEVAL-BCN002V1D



life.augmented

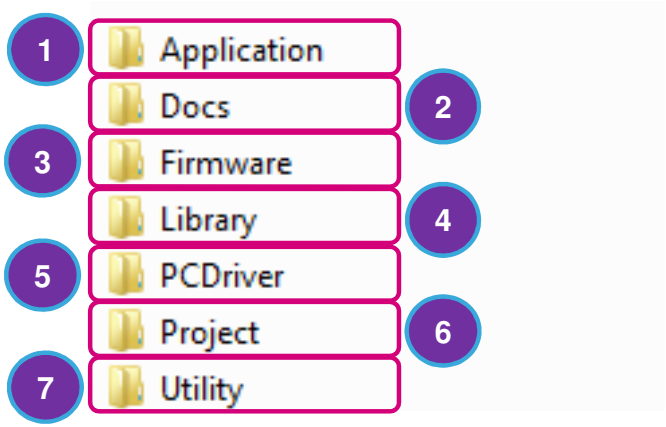
BlueNRG-Tile DK

- Open the **BlueNRG_Tile_HandsOn** folder in C:\



- Open the **STEVAL-BCN002V1 DK 3.0.0** folder

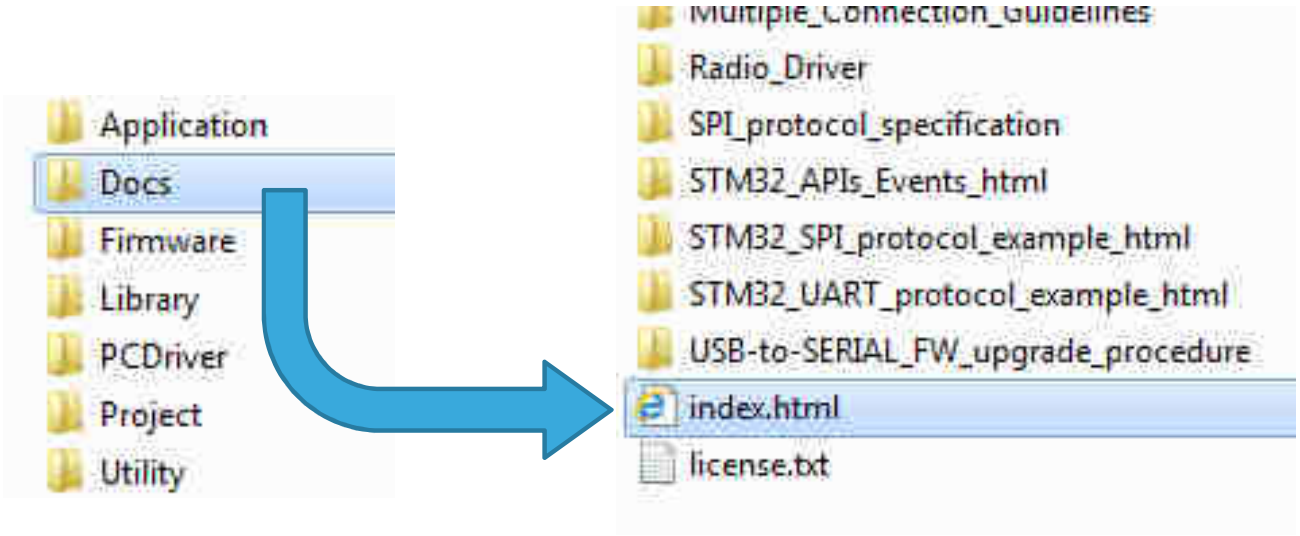
BlueNRG-Tile DK overview



1. PC Applications
2. Documentation
3. Pre-built FW images
4. Low level drivers and BLE stack library
5. Virtual COM port drivers
6. Reference examples in source code
7. Utility section: IAR BlueNRG-2 Flasher

Documents

- Open the **Docs** folder from the **STEVAL-BCN002V1 DK 3.0.0** folder



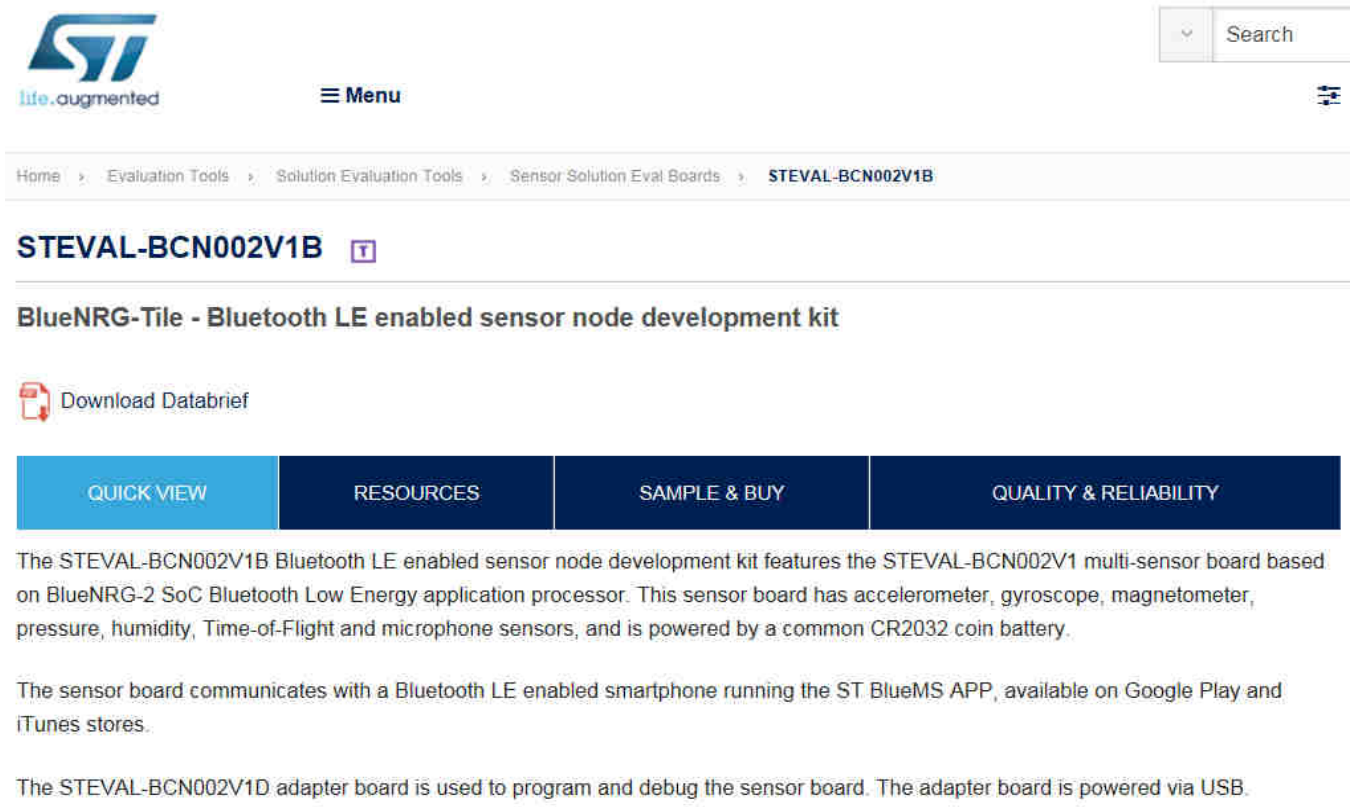
- **Double click** on **index.html**

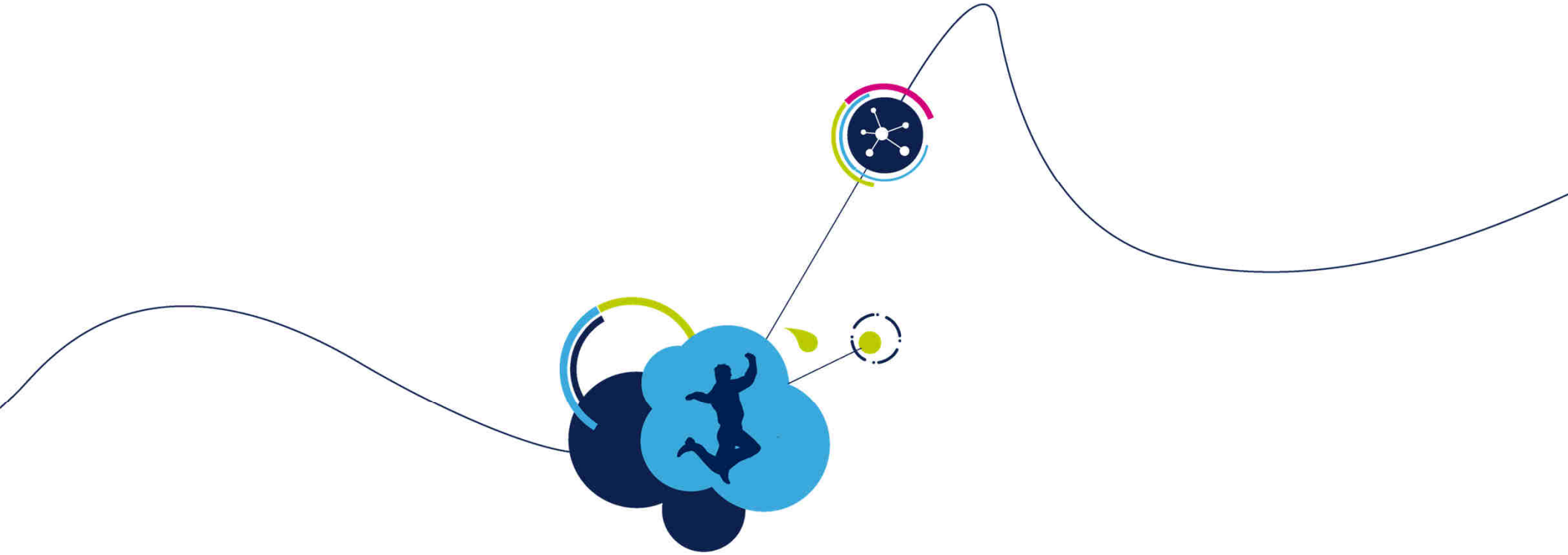
- Device Datasheet
- BLE stack documentation
  - Release notes
  - APIs and events
  - Programming manual
- Device (and kits) documentation
  - PCB design guidelines
  - Bring up guide
  - Getting started

life.augmented

On st.com

http://www.st.com/bluetile



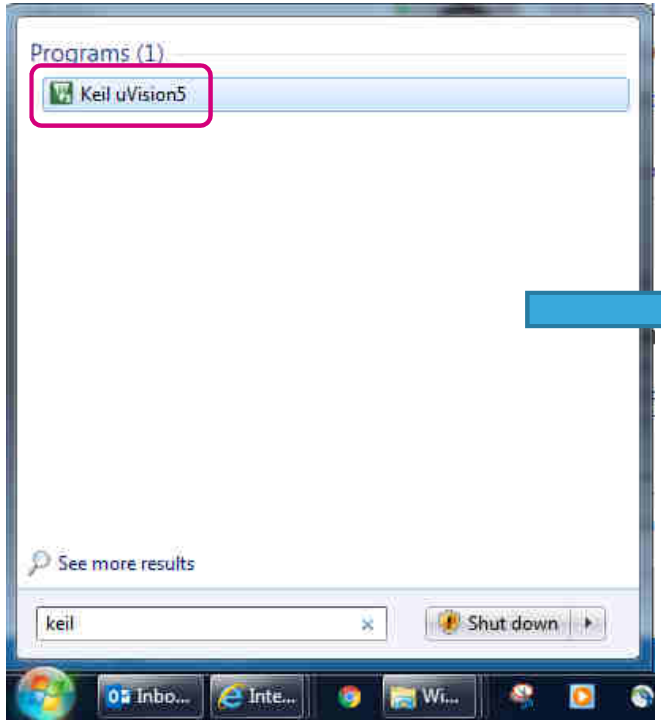


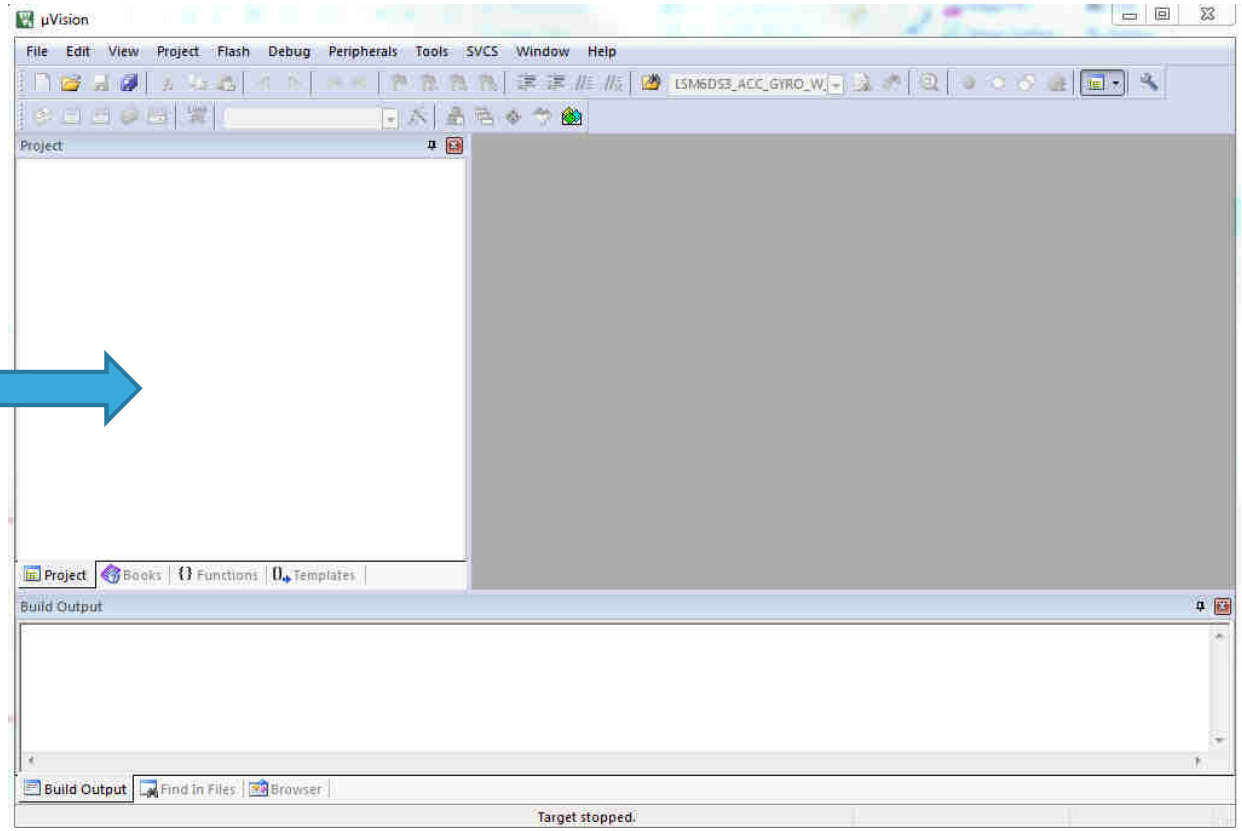
Arm Keil MDK Configuration and License installation

life.augmented

Check on successful tool installation

- **Open Keil uVision5 IDE**
- This is the main GUI

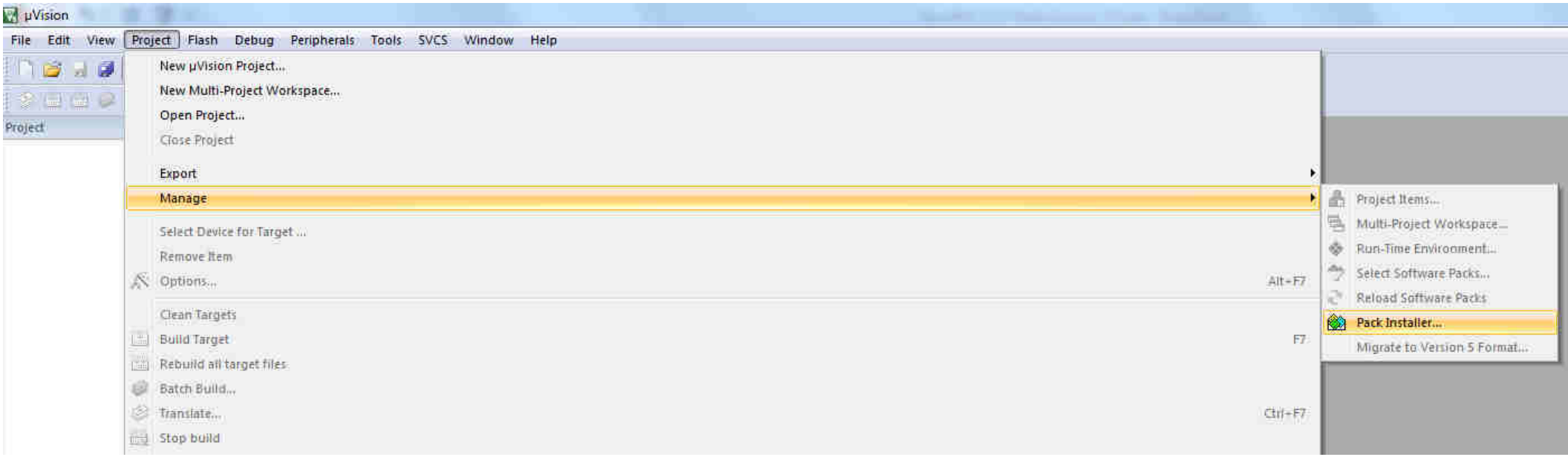




life.augmented

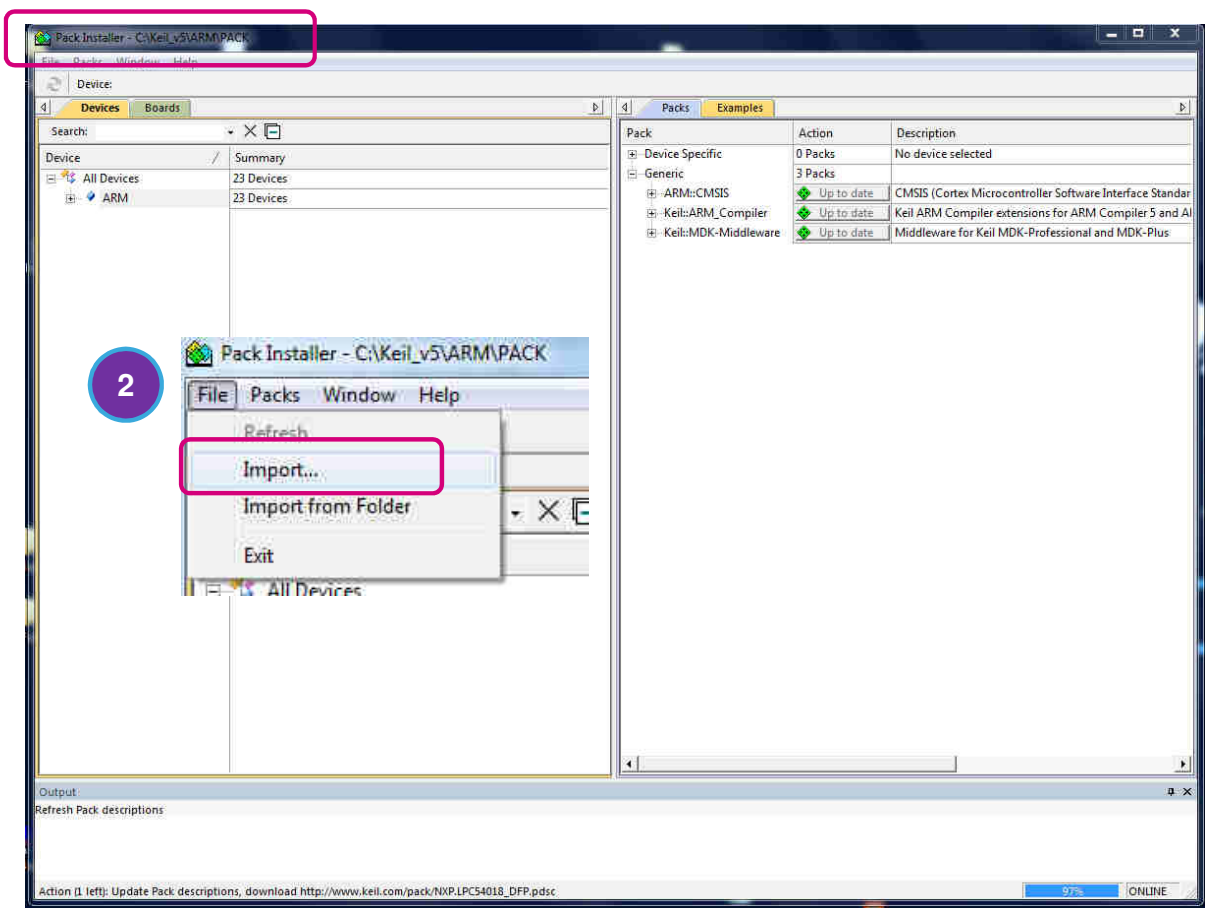
Keil MDK Installation 3/6

- **Open** the **Pack Installer**
  - **Go to Project->Manage->Pack Installer**



Keil MDK Installation 4/6

1. In the **Pack Installer…**
2. **Go to File->Import**

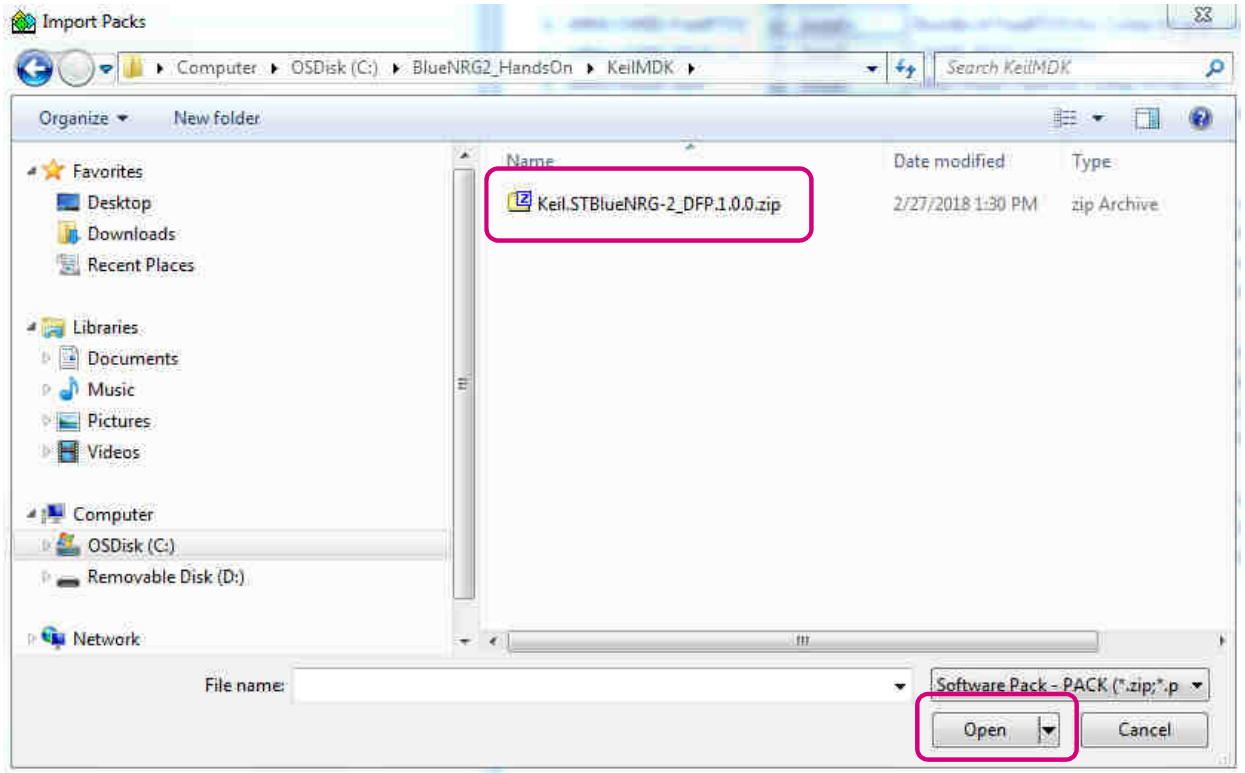


life.augmented

Keil MDK Installation 5/6

1. **Select** the .zip file "**Keil.STBlueNRG-2_DFP.1.0.0.zip**" from the folder "**C:\BlueNRG_Tile_HandsOn\KeilMDK**"

2. **Click** on **Open**



life.augmented

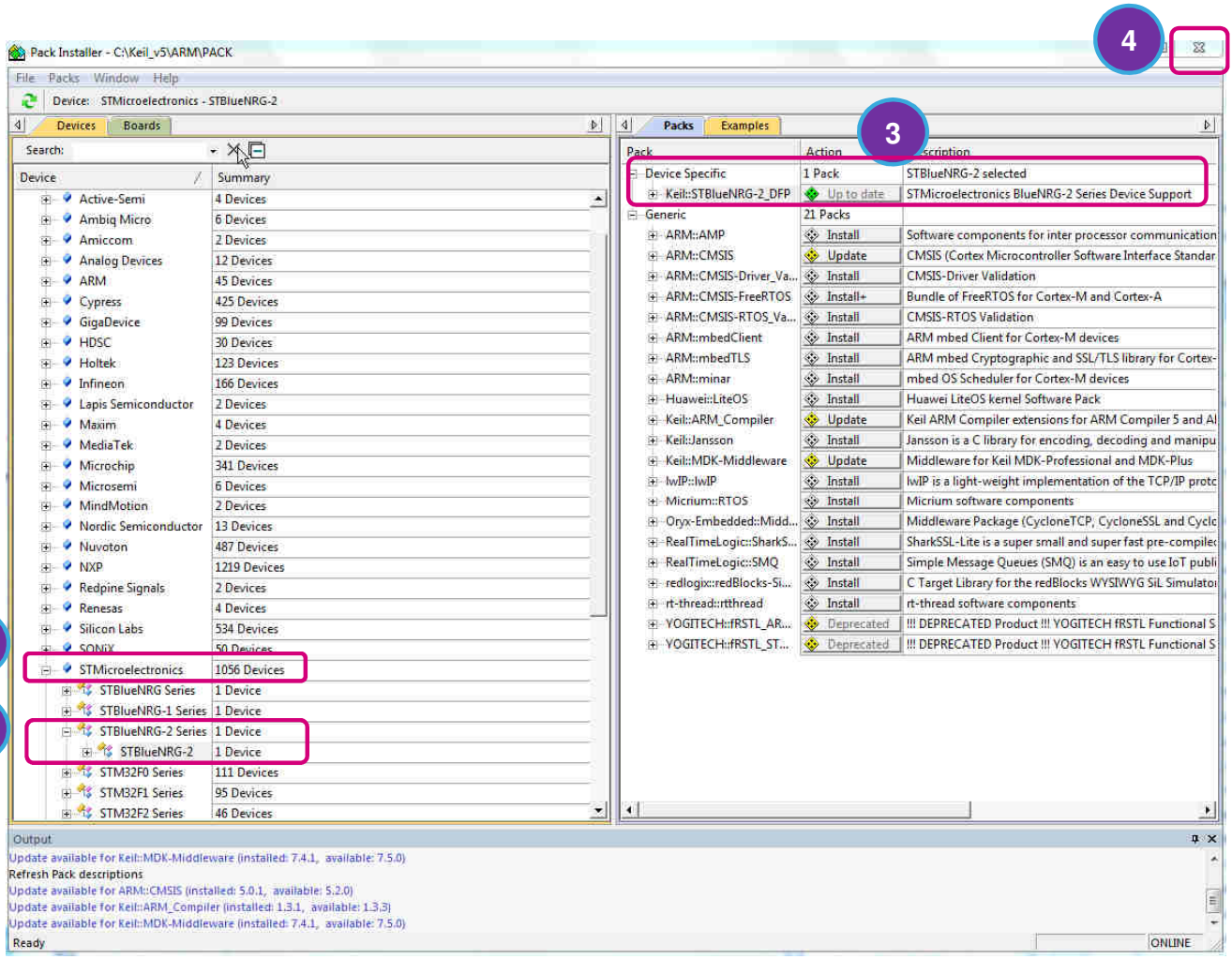
Keil MDK Installation 6/6

1. Once the Device Support pack has been installed, on the **Devices** list in the **left column** select the vendor **STMicroelectronics**

2. **Go to STBlueNRG-2 Series** and **select STBlueNRG-2**

3. Now it will show in green **“Up to date”**

4. **Close** the **Pack Installer**



life.augmented

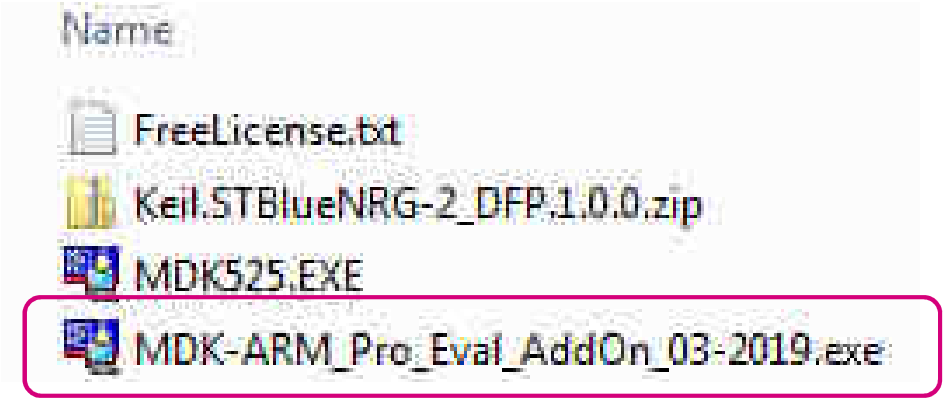
Arm Keil MDK License

- A **FREE** license is available for ST BlueNRG-2 device
  - This procedure requires on-line registration.
  - To avoid potential problems with the Wi-Fi network, we're skipping this for the hands-on and **we will install one temporary license** (expiring end of the month)

- But, **AT THE END OF THE PRESENTATION** you can find the step-by-step procedure for installing the **FREE** and **unlimited license**!

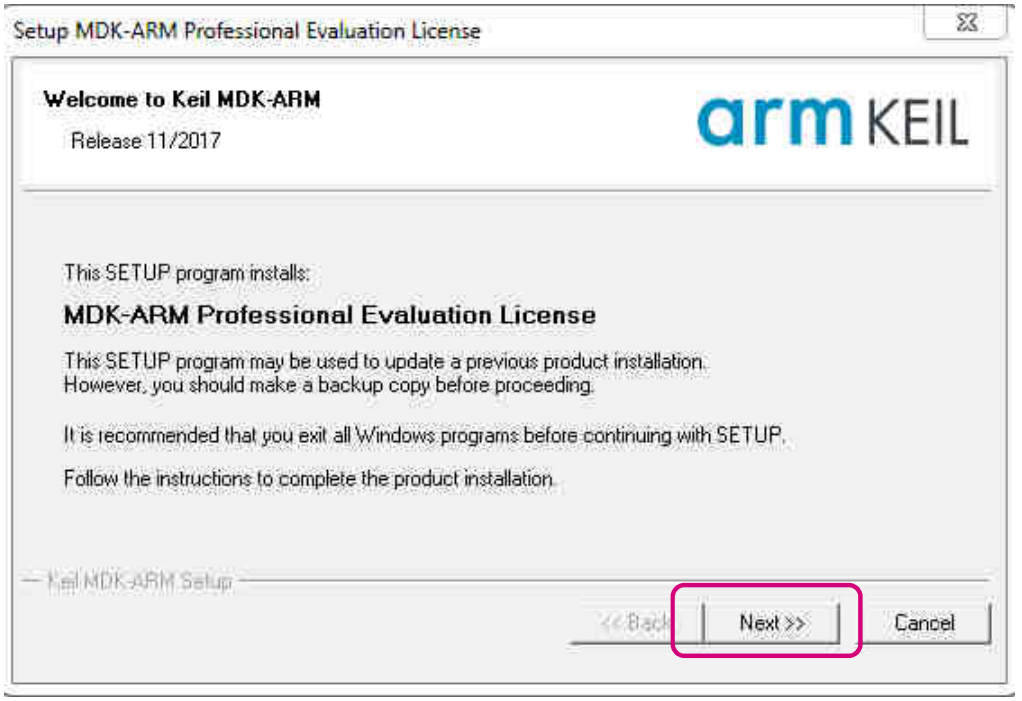
Arm Keil MDK License Installation 1/6

- **Go to** the folder "**C:\BlueNRG_Tile_HandsOn\KeilMDK**"
- **Double click** on the file **MDK-ARM_Pro_Eval_AddOn_03-2019**



Arm Keil MDK License Installation 2/6

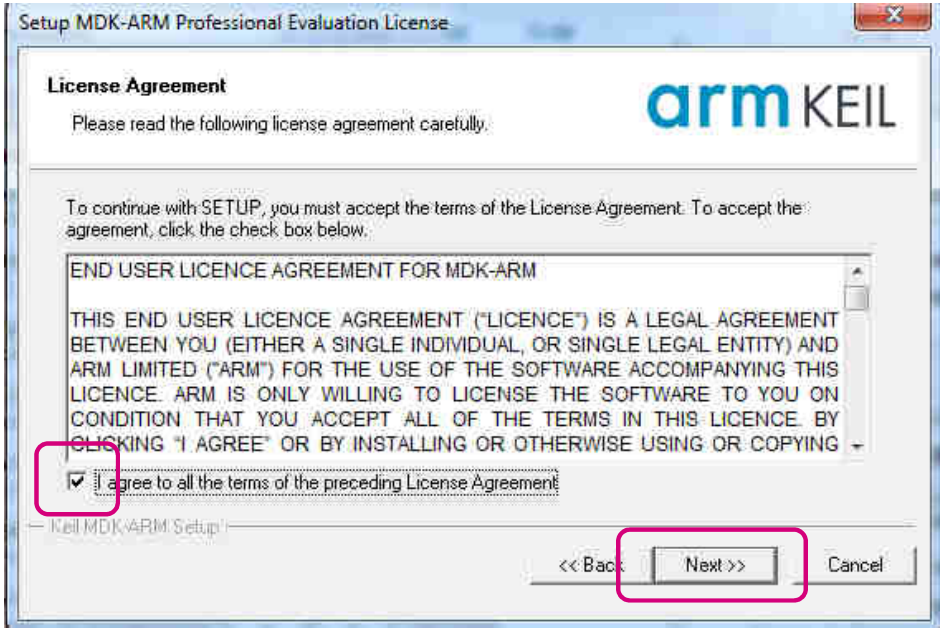
- This pop up window will appear
- **Click** on **Next**



life.augmented

Arm Keil MDK License Installation 3/6

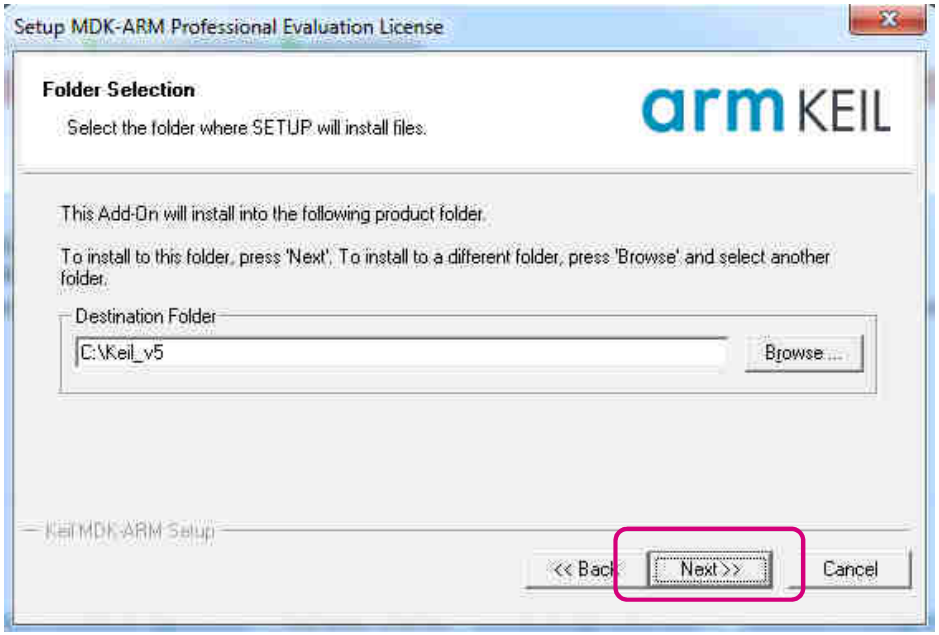
- **Select** on **Agree** **License terms**
- **Click** on **Next**



life.augmented

Arm Keil MDK License Installation 4/6

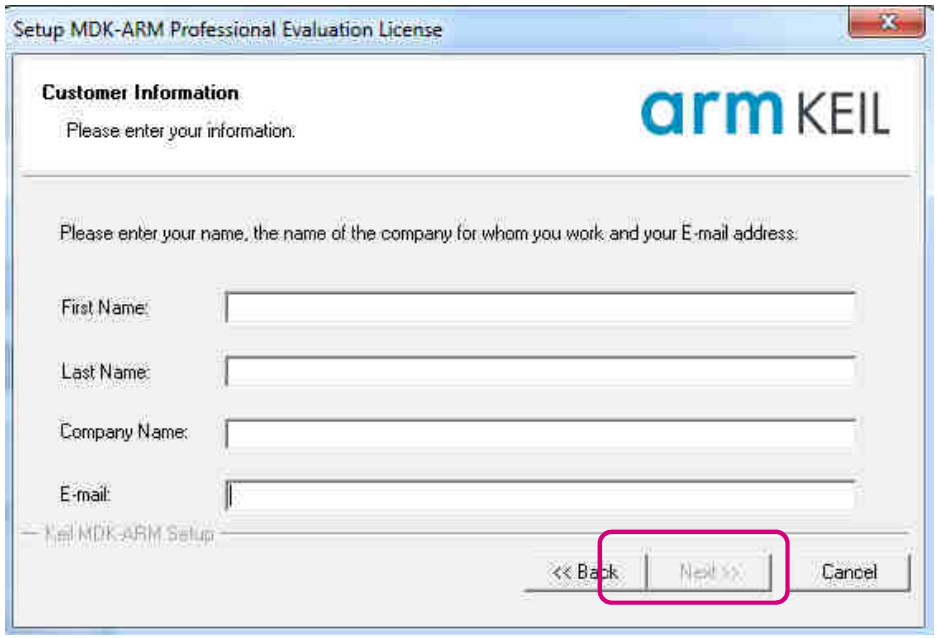
- **Click** on **Next**



life.augmented

Arm Keil MDK License Installation 5/6

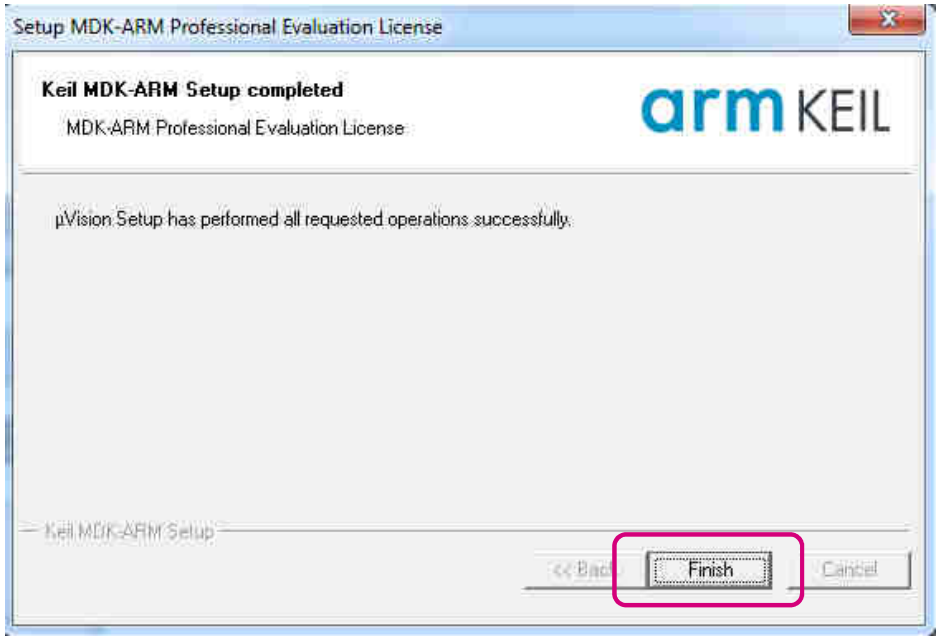
- **Fill in your data** and **click** on **Next**



life.augmented

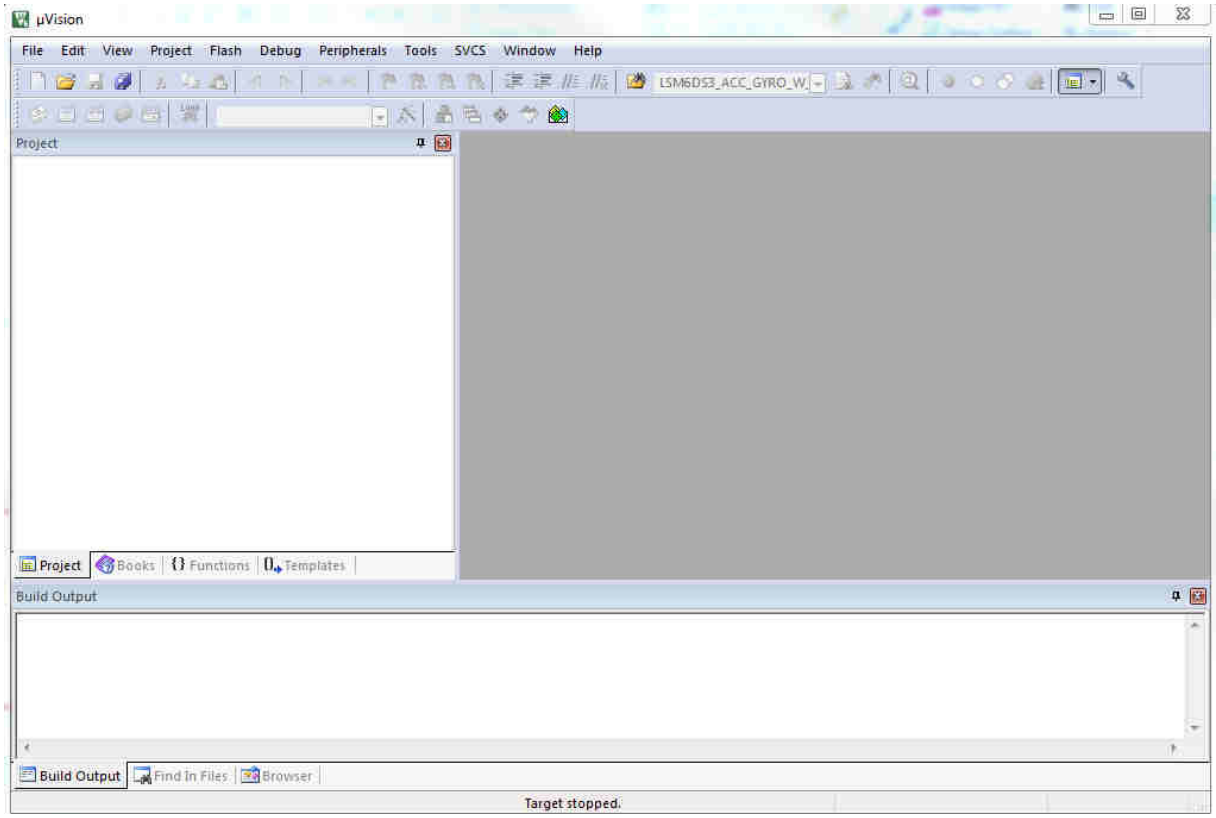
Arm Keil MDK License Installation 6/6

- **Click** on **Finish**

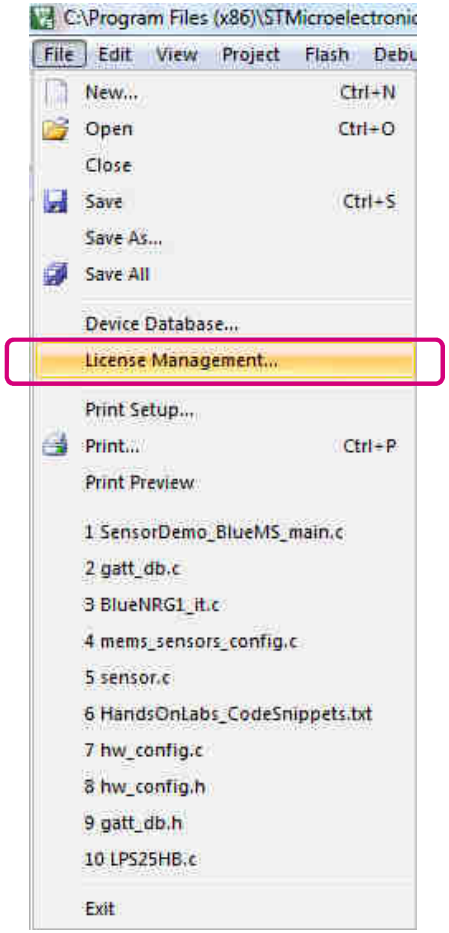


Arm Keil MDK License Installation – **FINAL CHECK**

- **Go back** to **Keil uVision5 IDE**



Arm Keil MDK License Installation – FINAL CHECK



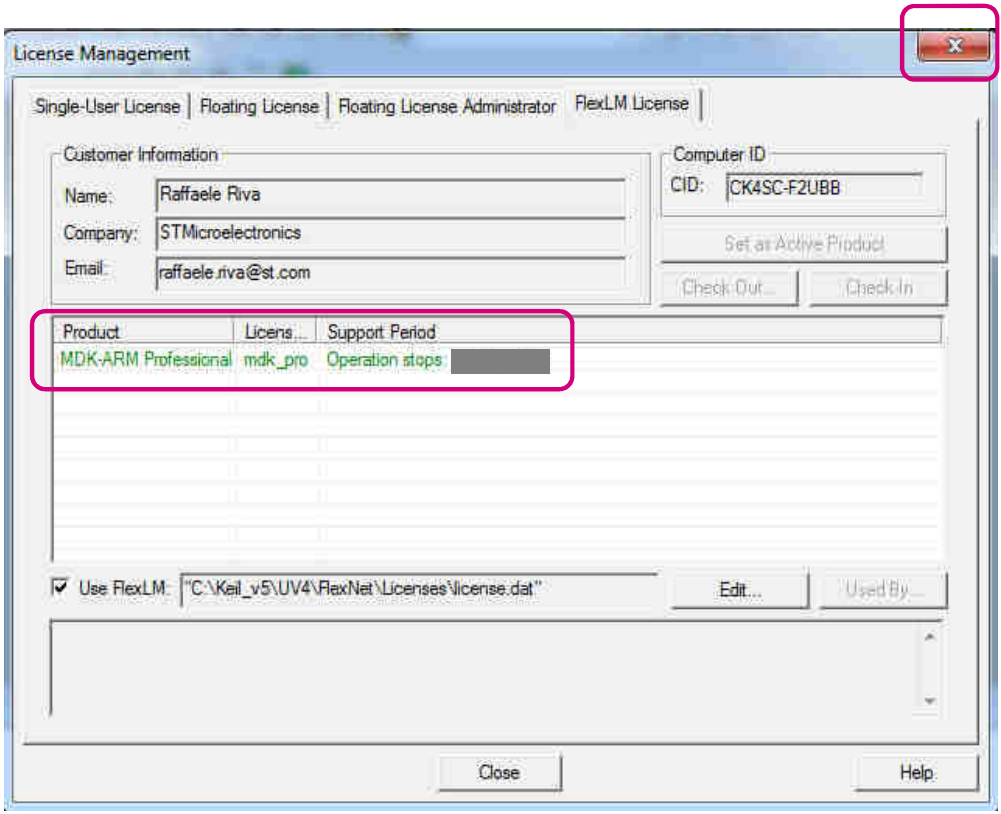
- In Keil uVision tool **open the dialog**

File -> License Management...

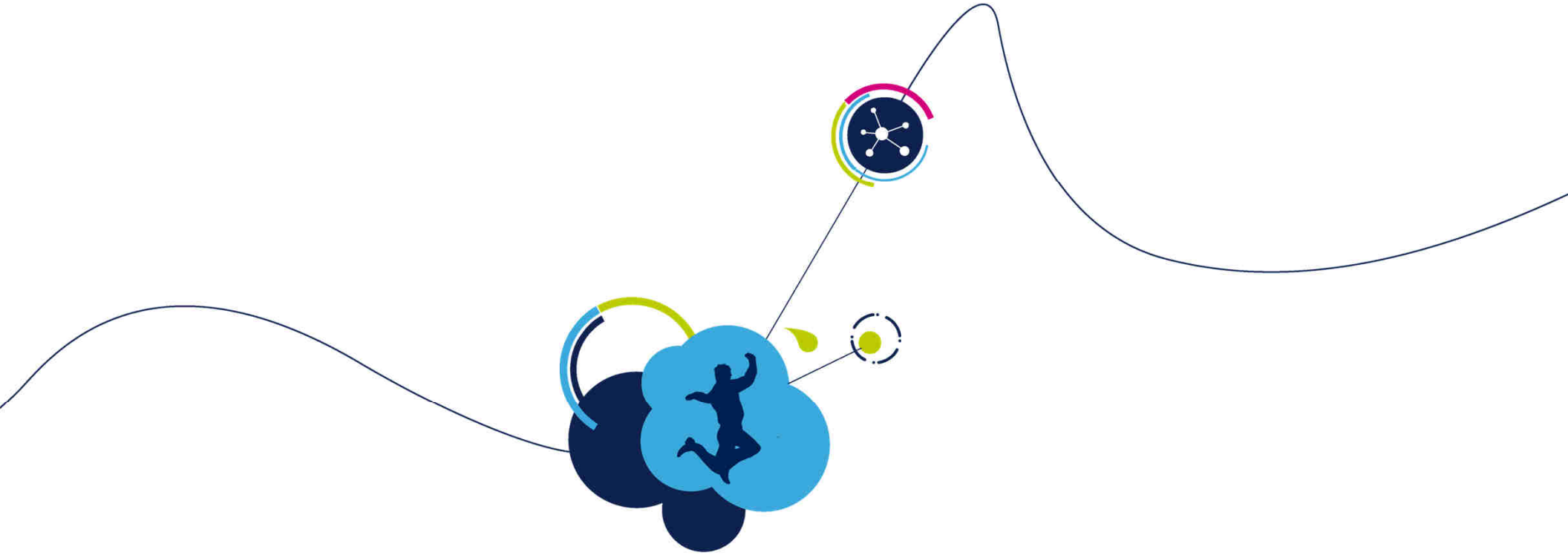
life.augmented

Arm Keil MDK License Installation – **FINAL CHECK**

- **Select** the tab **FlexLM License** and you should see the product **“MDK-ARM Professional”**
- **Close** the **License Management** popup window



life.augmented



Hands on overview

life.augmented

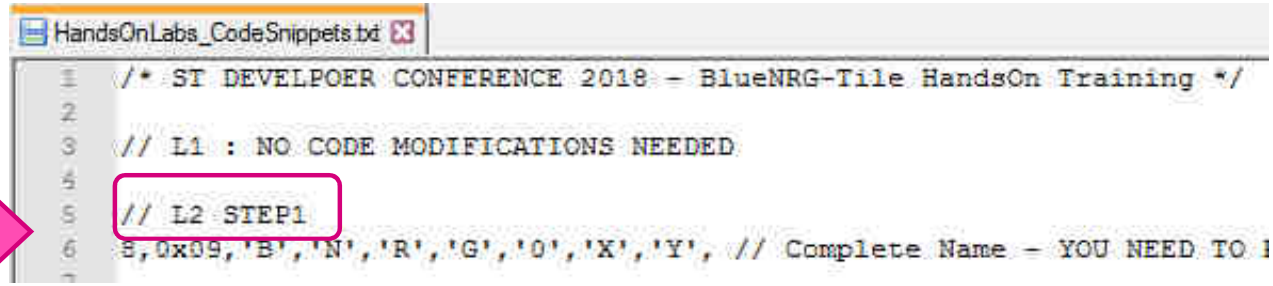
Labs overview

- **Lab 1**: Getting started with STEVAL-BCN002V1 “Serial Terminal Test”
- **Lab 2**: Connecting to the ST BLE Sensor app
- **Lab 3**: LED characteristic
- **Lab 4**: Accelerometer embedded events detection
- **Lab 5**: 9-axis Acc+Gyro+Mag Sensor Data Fusion
- **Lab 6**: Cloud data logging on IBM Watson
- **Lab 7**: Bonus Lab – Voice over BLE

Code Snippets

1. **Go to** the path
2. Open with a text editor the file

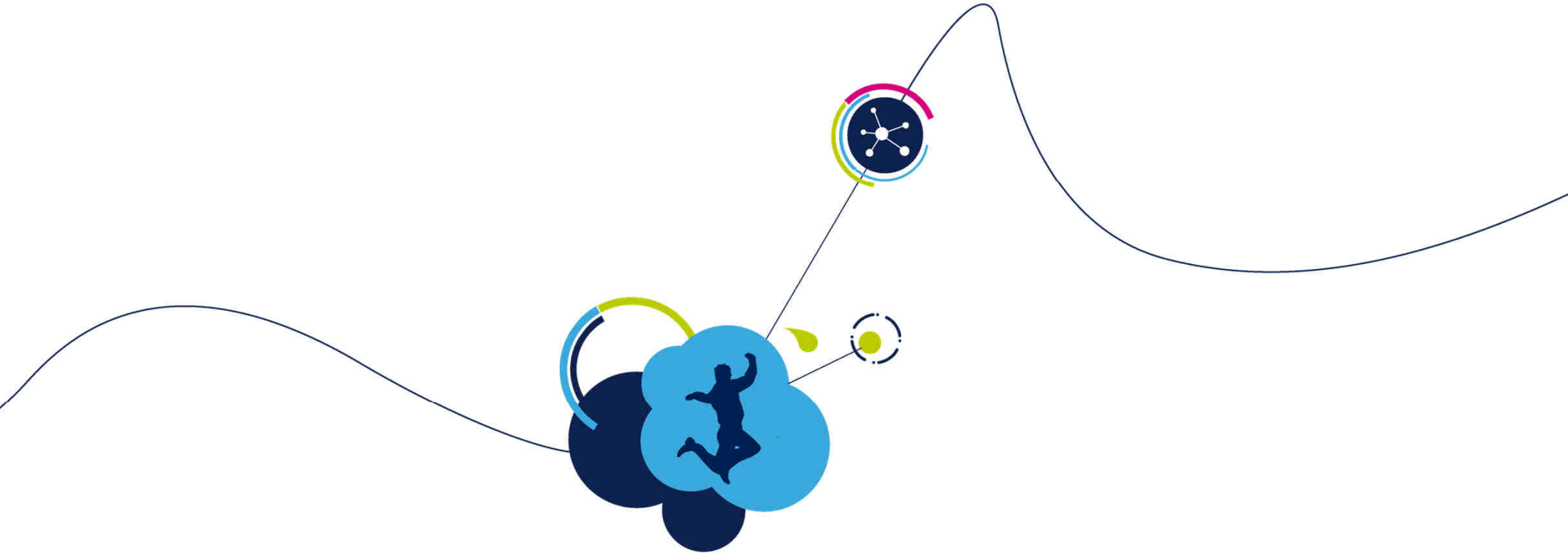
1:1 correspondence with slides titles



Do NOT close this file!

L2 STEP1 : Customize your BlueNRG

life.augmented



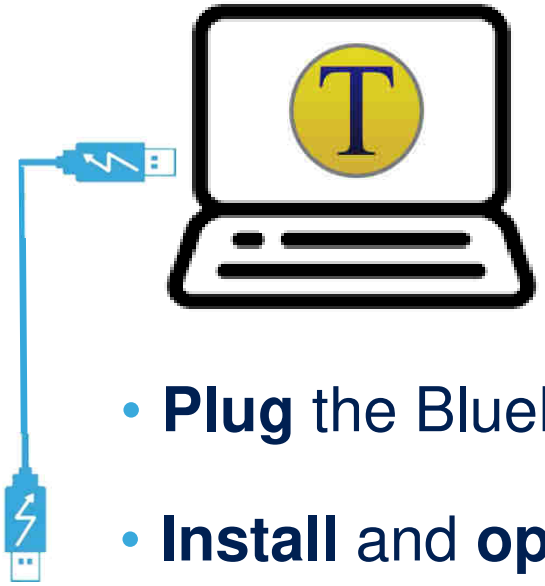
Lab 1

Getting Started with STEVAL-BCN002V1

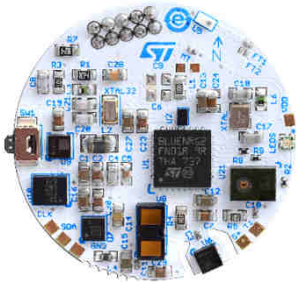
“Serial Terminal Test”

life.augmented

“Serial Terminal Test”



- **Plug** the BlueNRG-Tile to the PC using the USB cable
- **Install** and **open** **Tera Term** and **configure** serial terminal



life.augmented

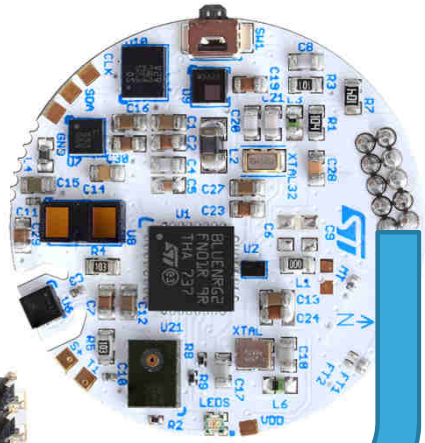
Code modifications

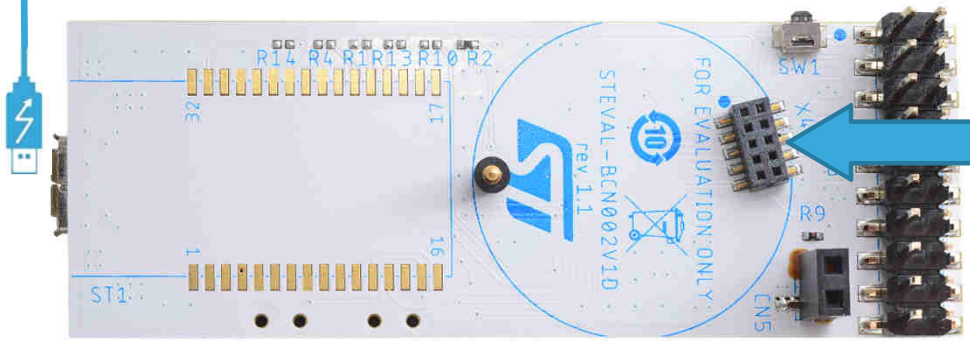
NO MODIFICATIONS NEEDED!

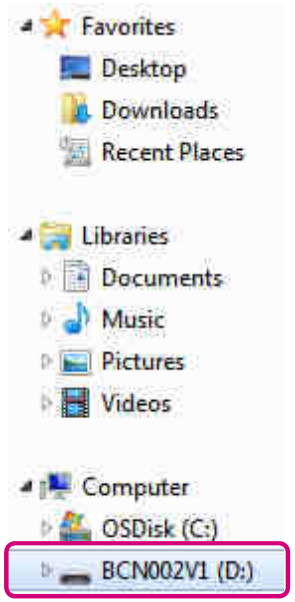
Connect your STEVAL-BCN002V1 to the PC using USB

Connect the USB cable

The BlueNRG-Tile comes already connected to the daughterboard





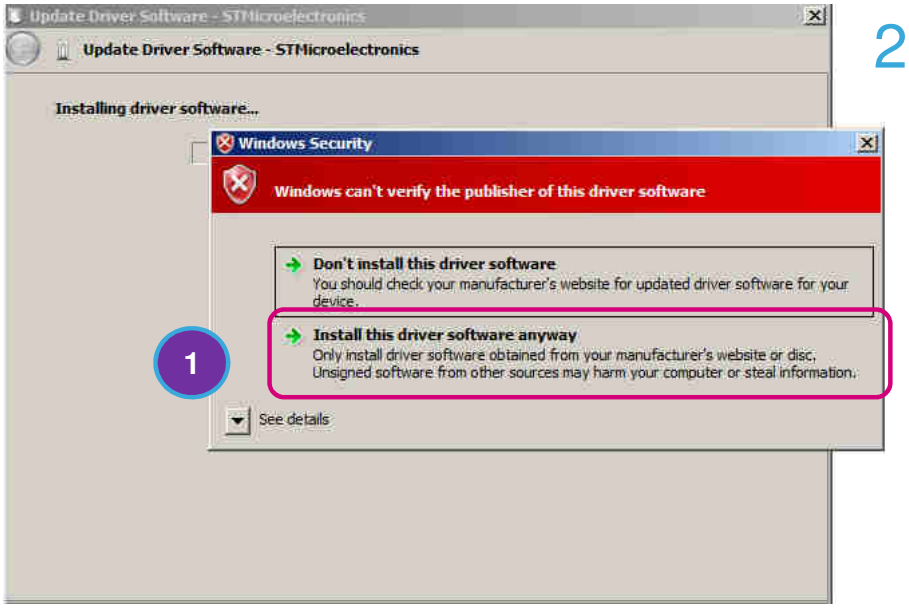


On the PC file system a BCN002V1 drive will appear

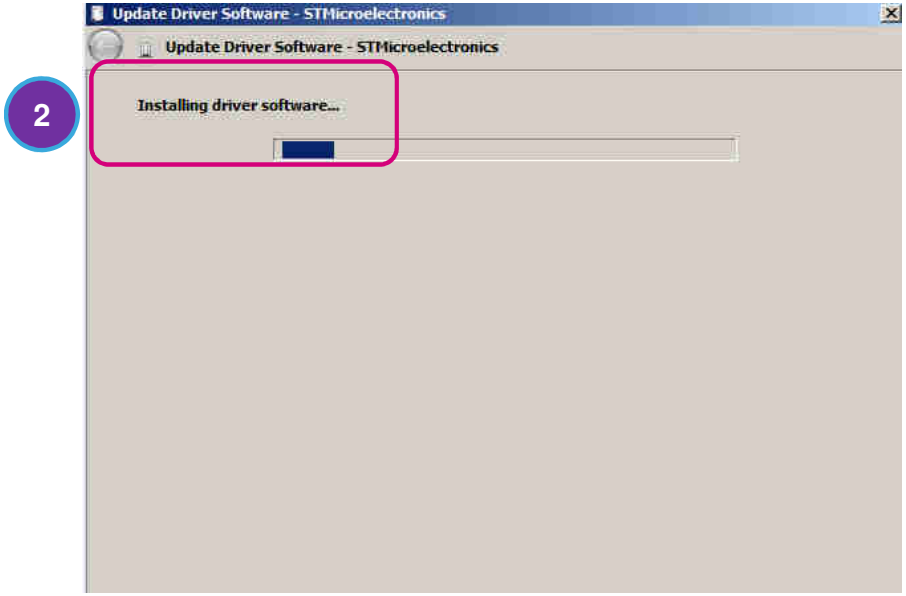
life.augmented

Windows7: Allow the driver installation

1. Click on **Install driver software anyway**



2. Installation starts



life.augmented

Virtual COM port driver

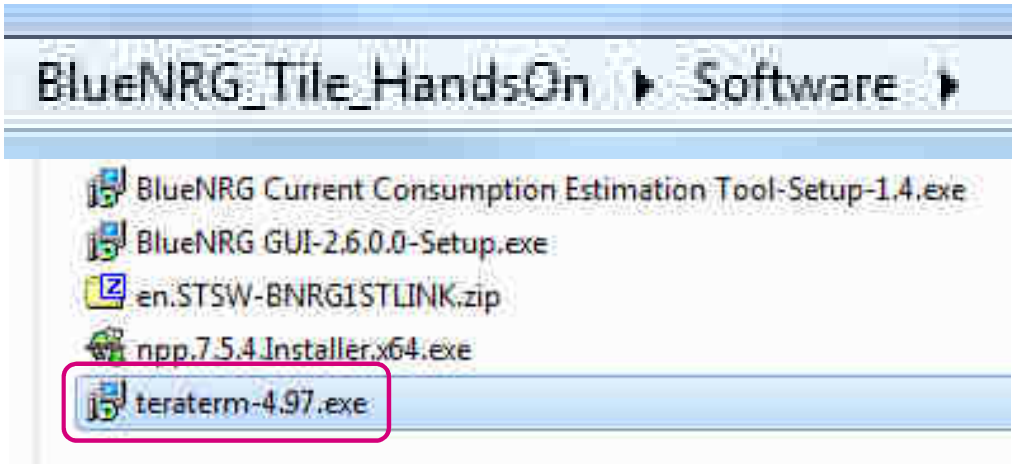
If you have issue with the **STMicroelectronics Virtual COM Port** device driver installation, **raise your hand**.

Here the **instructions** for installing the Virtual COM port driver:

Win7 **Win10**

Tera Term installation 1/10

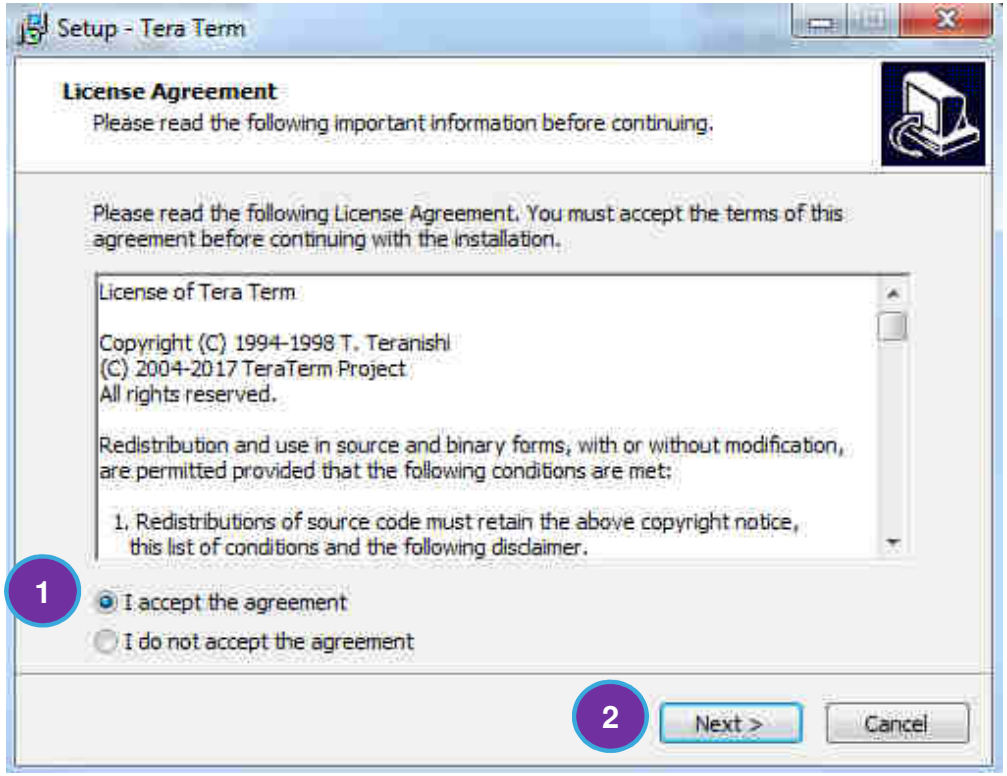
1. **Go to the folder C:\BlueNRG_Tile_HandsOn\Software**



2. **Double Click** on **teraterm-4.97.exe**

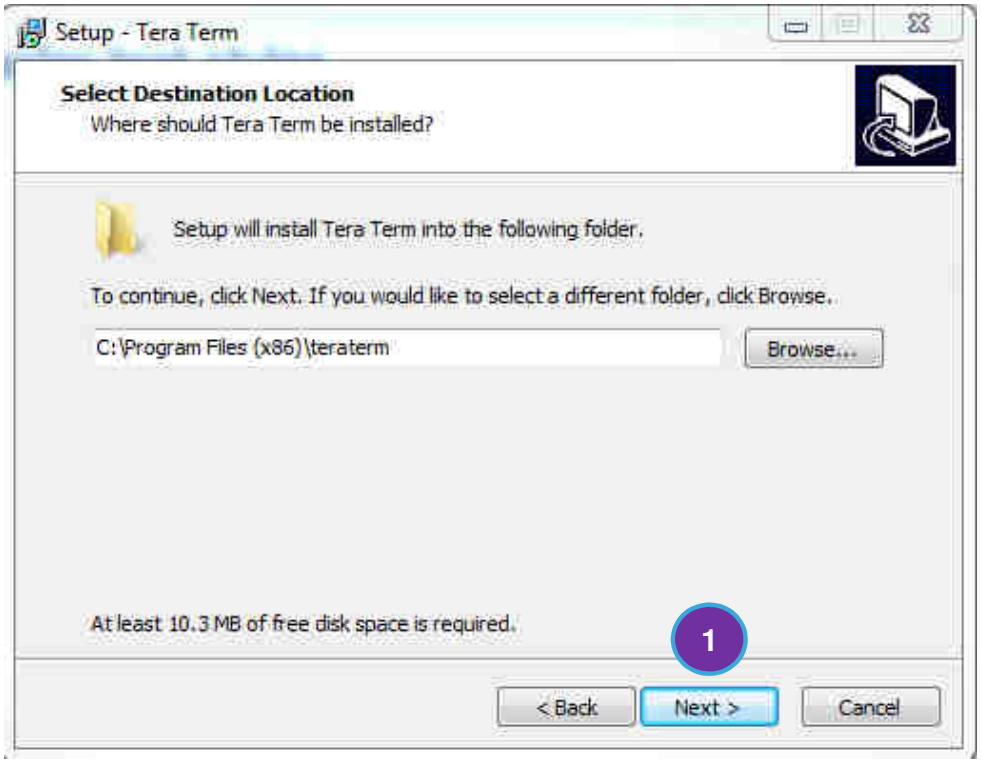
Tera Term installation 2/10

1. **Select Accept the agreement**
2. **Click** on **Next**



Tera Term installation 3/10

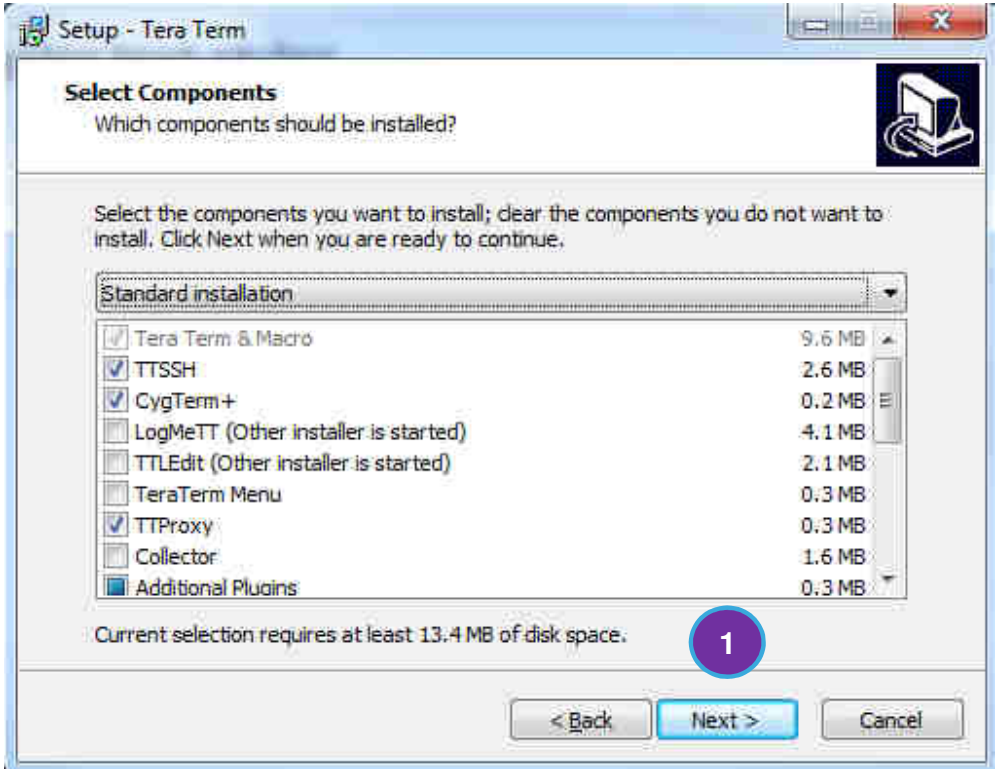
1. **Click** on **Next**



life.augmented

Tera Term installation 4/10

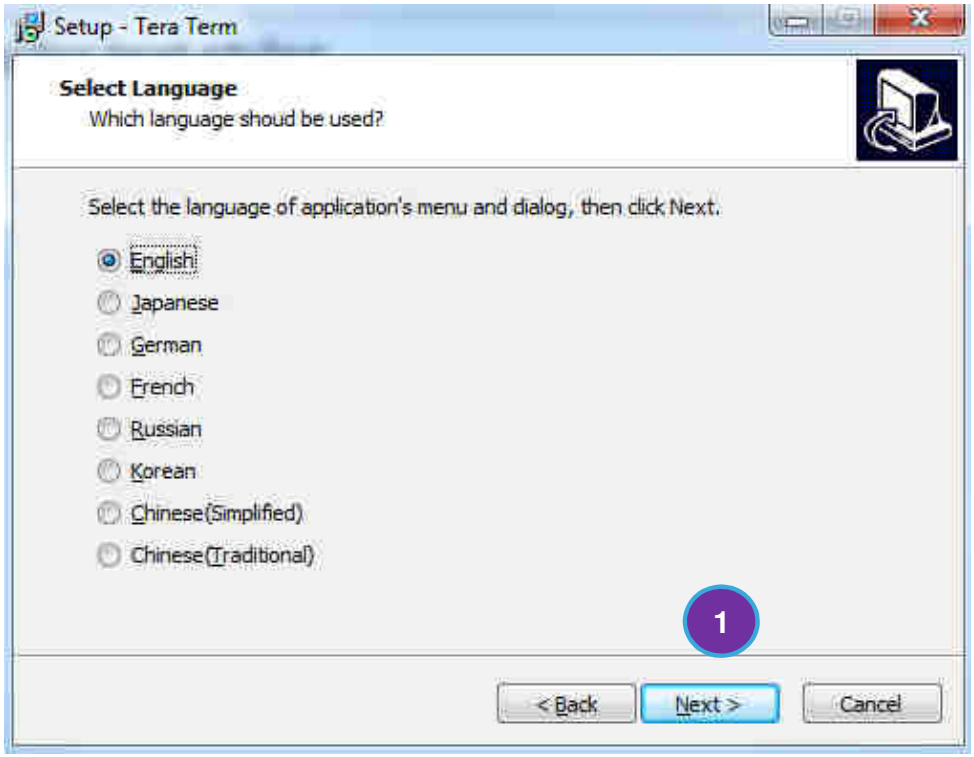
1. **Click** on **Next**



life.augmented

Tera Term installation 5/10

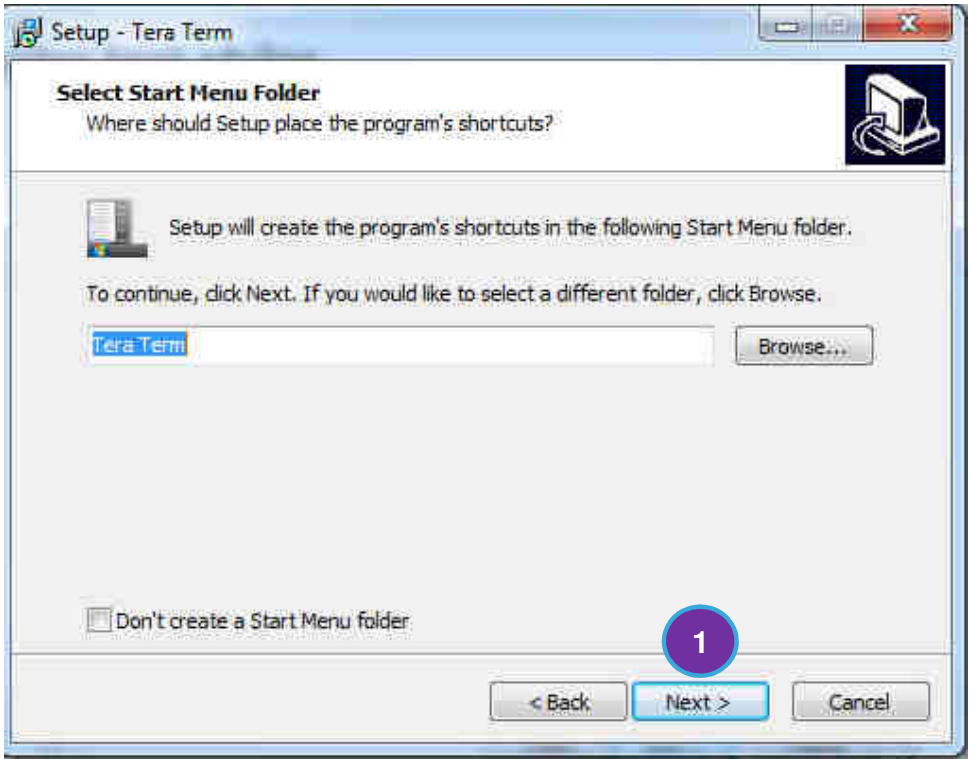
1. **Click** on **Next**



life.augmented

Tera Term installation 6/10

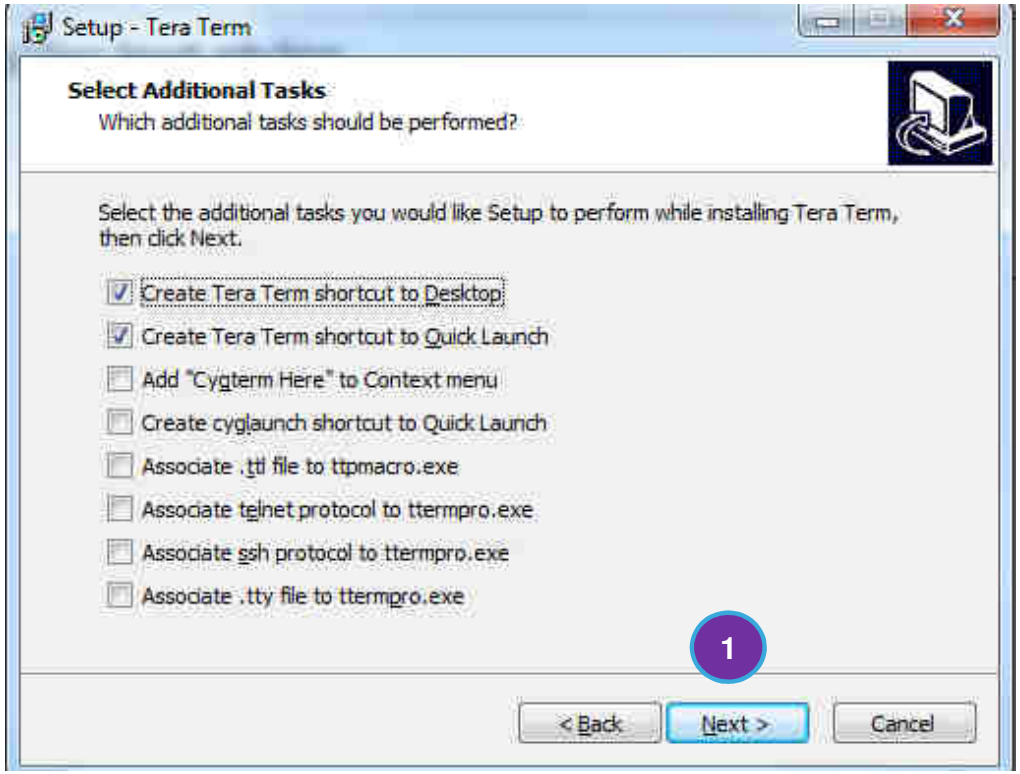
1. **Click** on **Next**



life.augmented

Tera Term installation 7/10

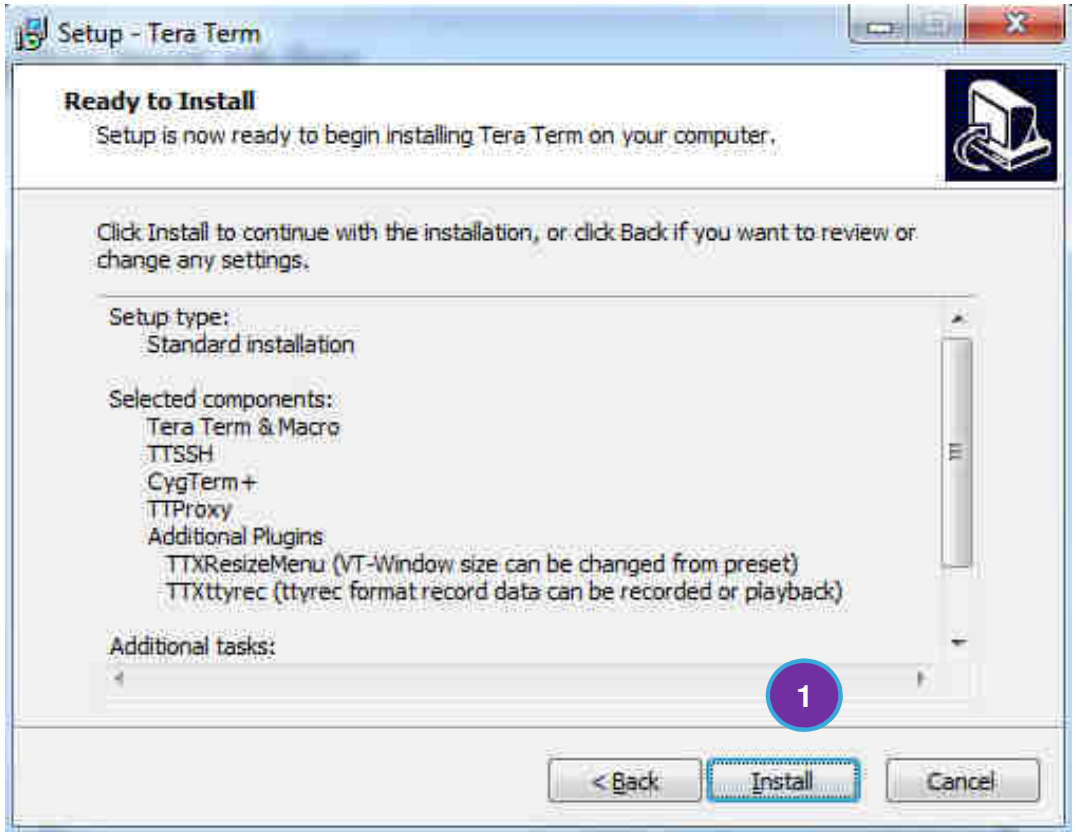
1. **Click** on **Next**



life.augmented

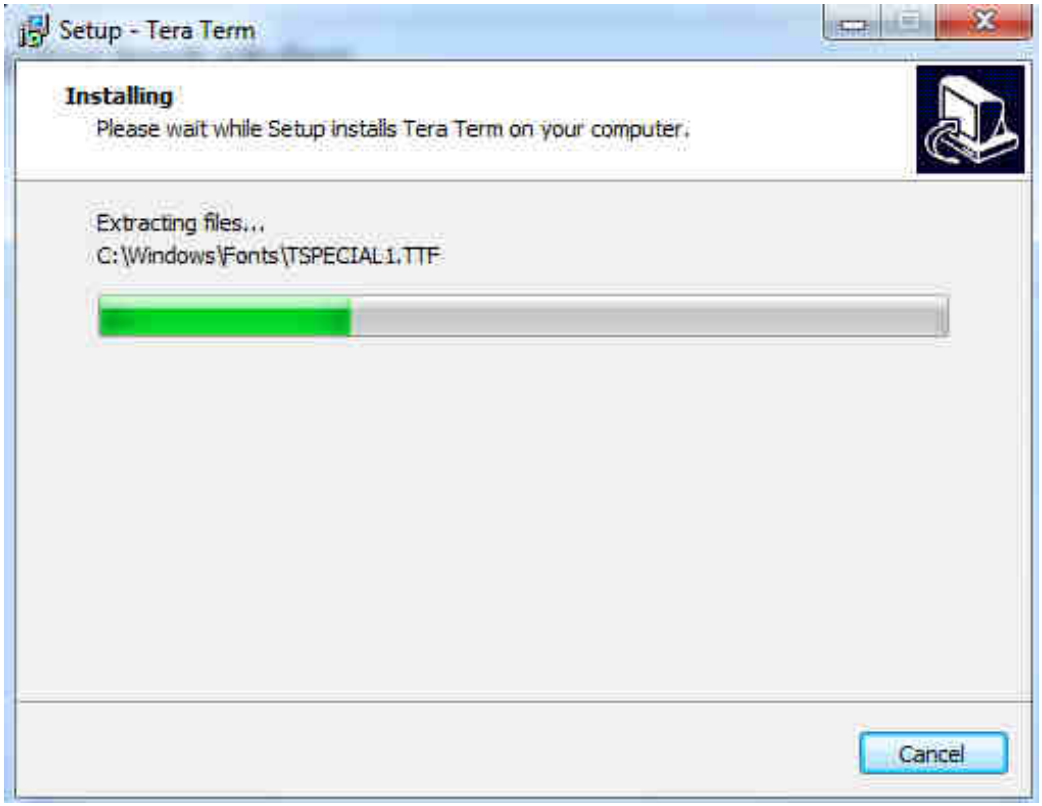
Tera Term installation 8/10

1. **Click** on **Next**



Tera Term installation 9/10

Installation starts...

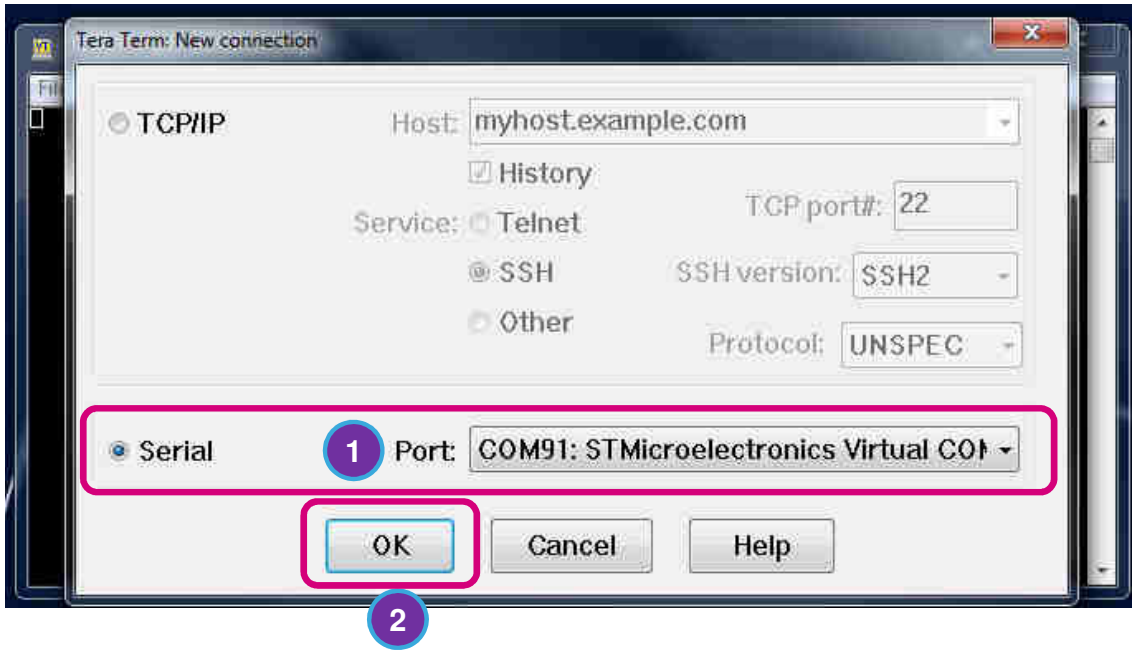


Tera Term installation 10/10

1. **Select Launch Tera Term**
2. **Click** on **Finish**

life.augmented

Tera Term Configuration 1/6

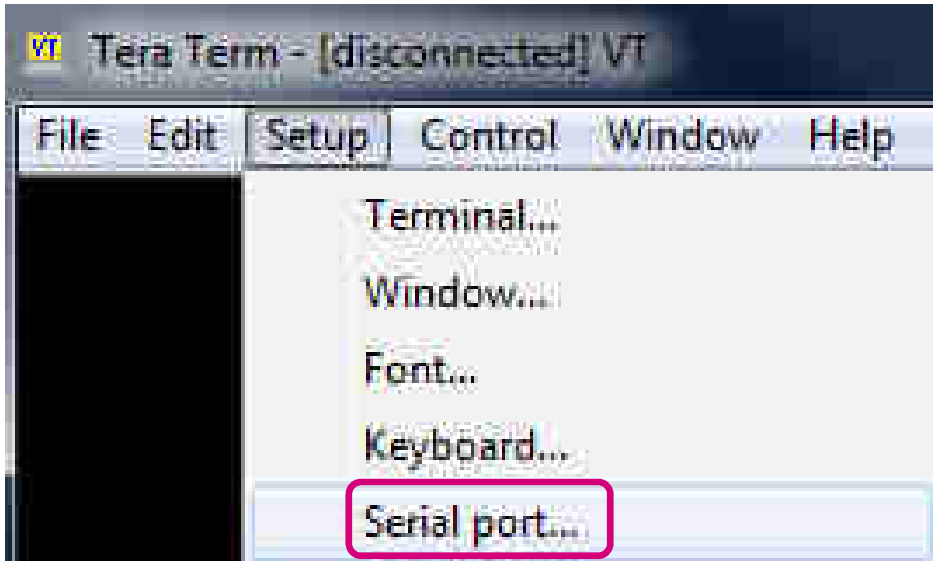


1. **Select** the **STMicroelectronics Virtual COM Port**
   - **NOTE:** on **Win10 PC** the serial port is labeled just as **"COMxx"**
2. Click **OK**

life.augmented

Tera Term Configuration 2/6

1. **Click Setup -> Serial port…**



life.augmented

Tera Term Configuration 3/6

1. **Set** the following:

   Baud rate : 115200

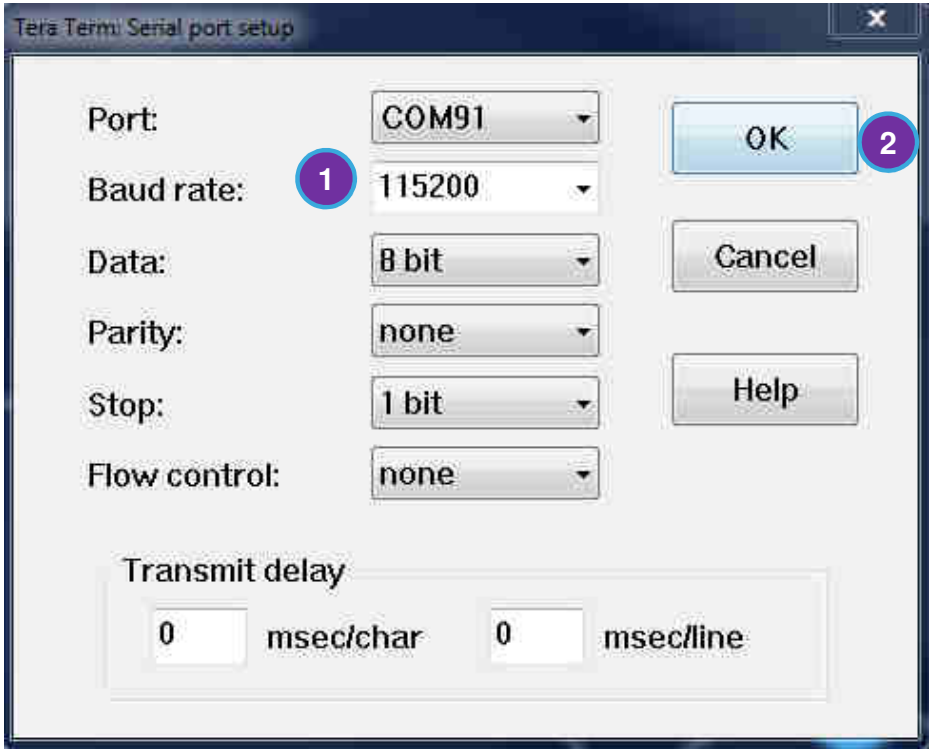
   Data : 8 bit

   Parity : none

   Stop : 1 bit

   Flow control : none

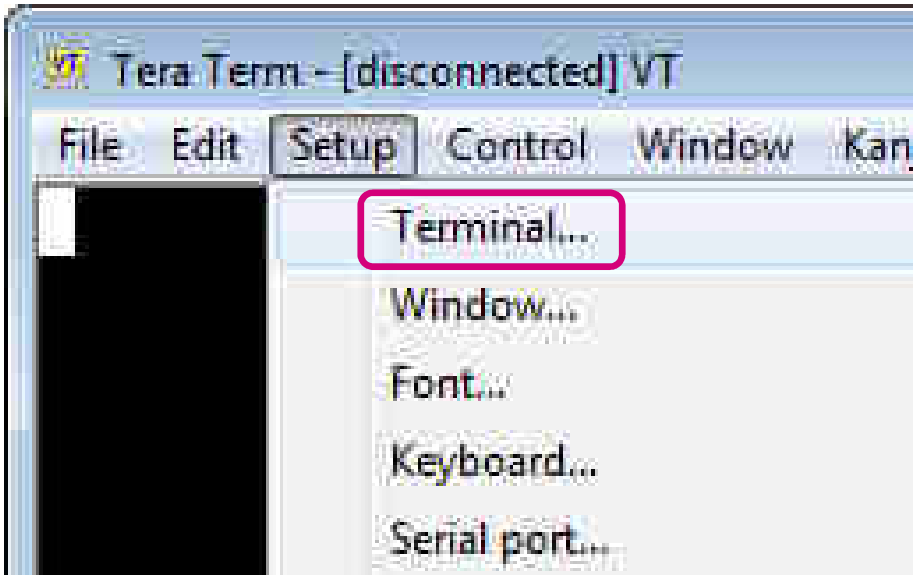
2. **Click OK**



life.augmented

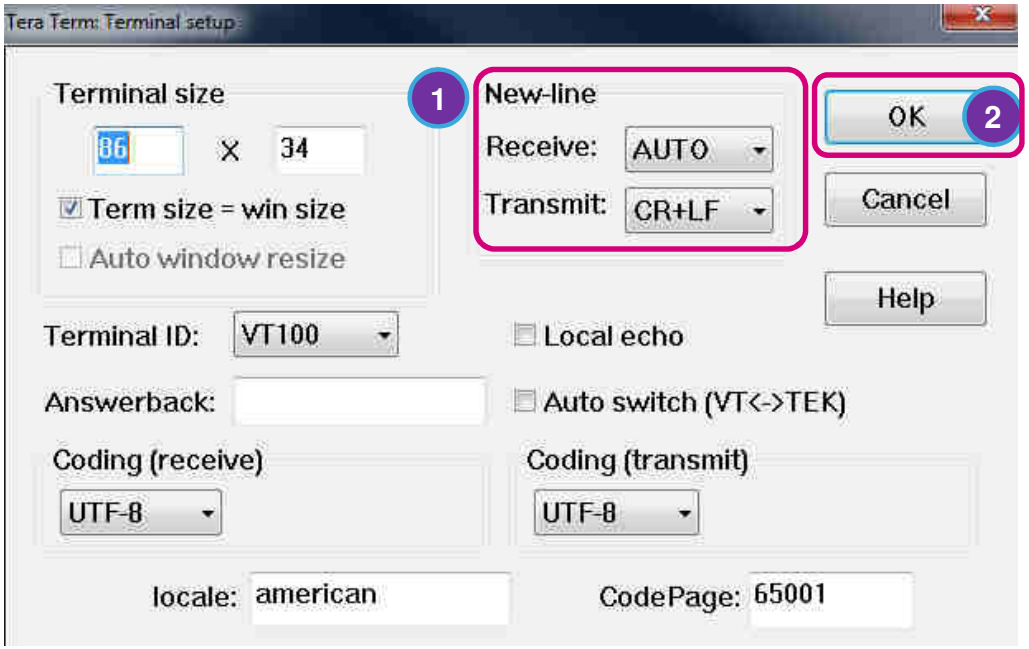
Tera Term Configuration 4/6

1. Click **Setup** -> **Terminal…**



Tera Term Configuration 5/6

1. In the **New-line set**:
   Receive : AUTO
2. **Click OK**

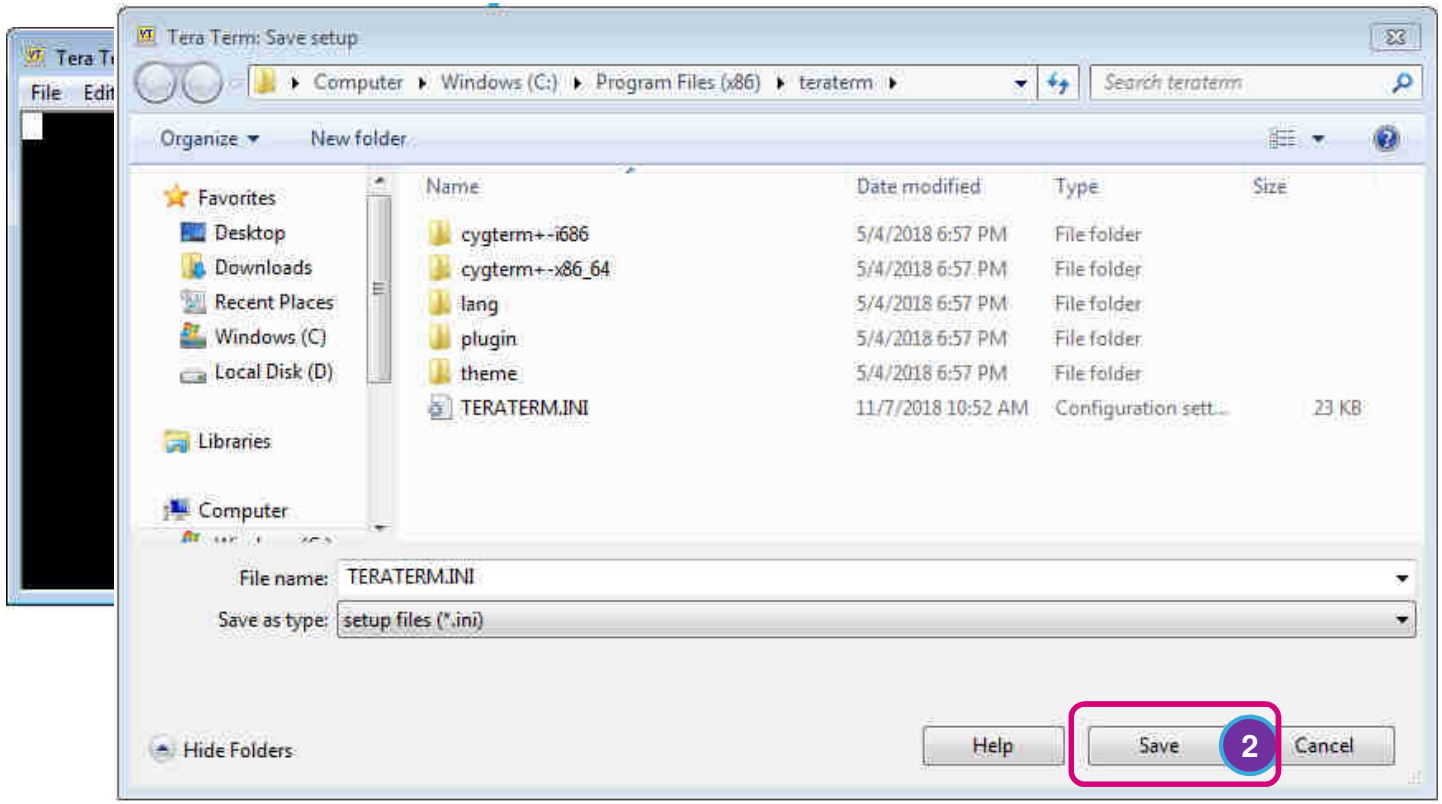


life.augmented

Tera Term Configuration 6/6

1. Click **Setup->Save setup…**

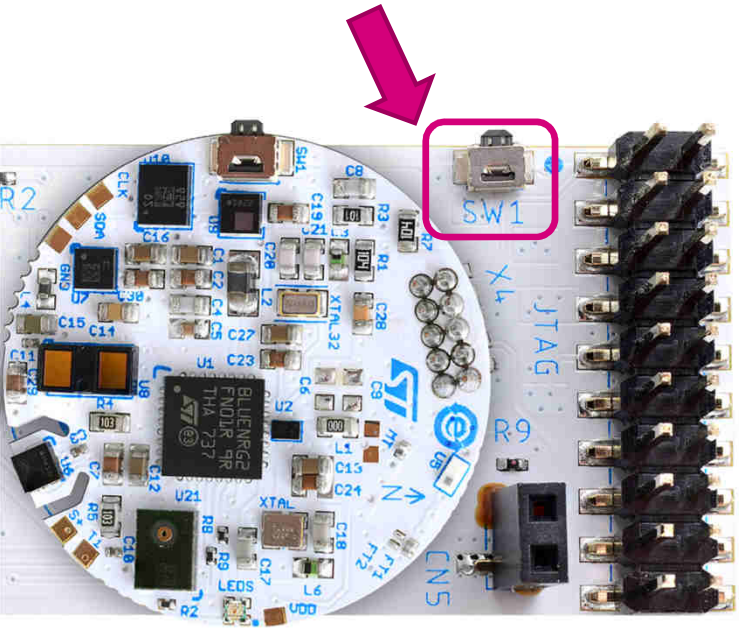
2. Click **Save**

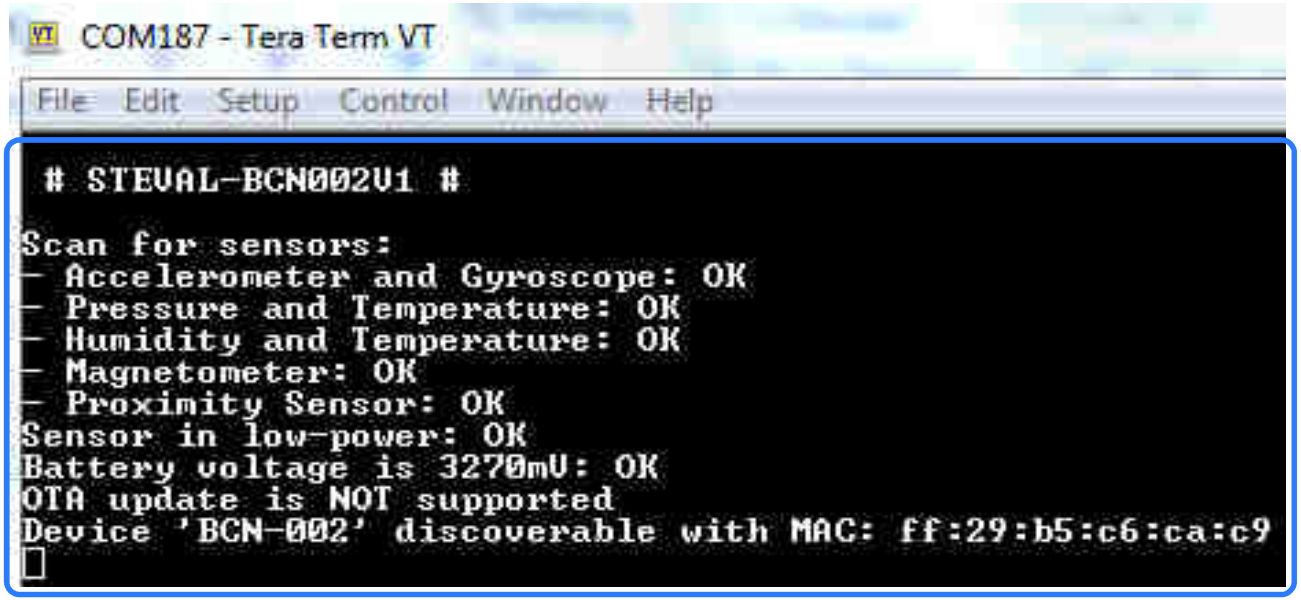


life.augmented

…done!

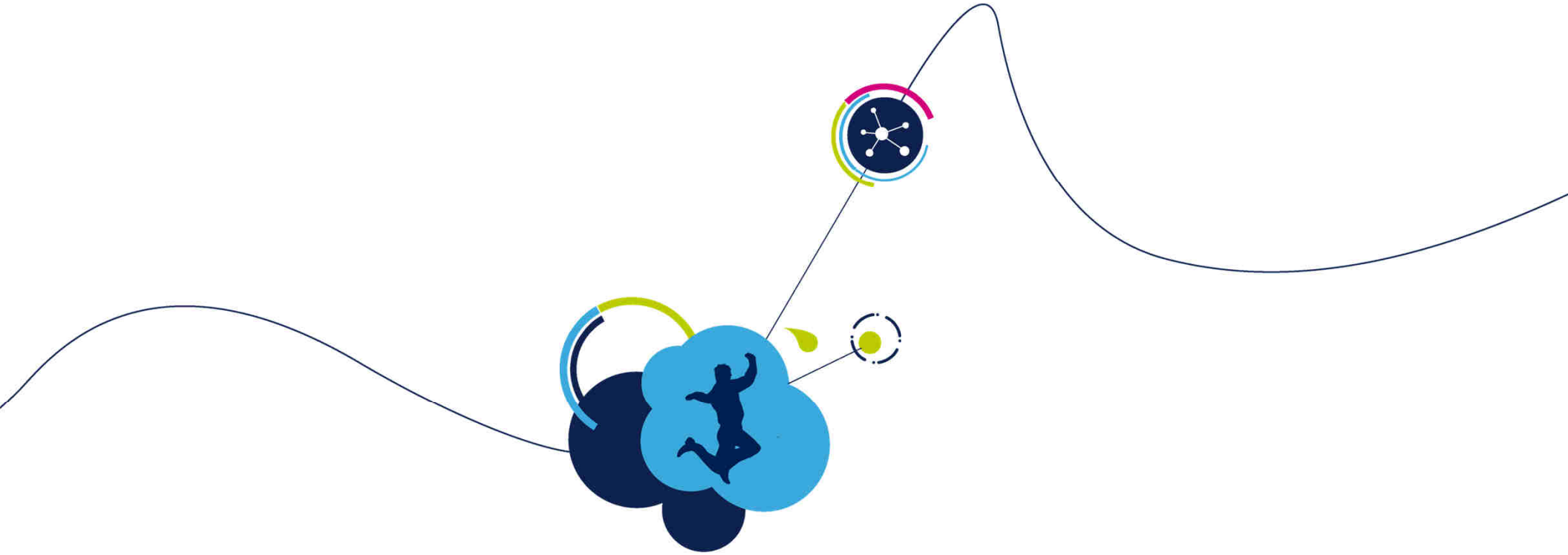
Push SW1 button on the daughterboard





- If you see the output above, **Tera Term** is now properly configured

life.augmented



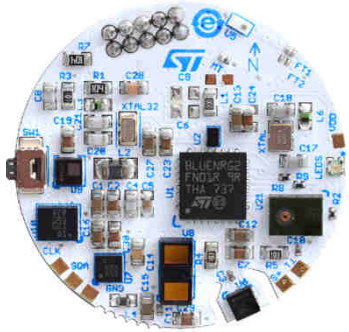
Lab 2

Connect to the ST BLE Sensor app

life.augmented

Customize *YOUR* STEVAL-BCN002V1

+ STEVAL-BCN002V1 DK
Sensor Demo ref FW



Client

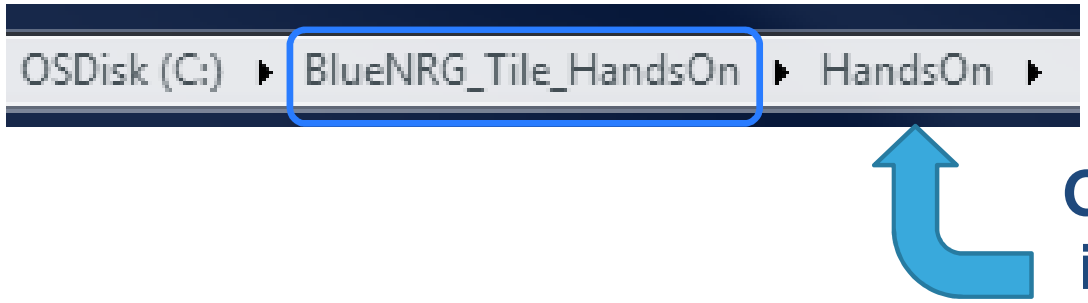
life.augmented

Code modifications

1. **Modify local name** in advertising packet

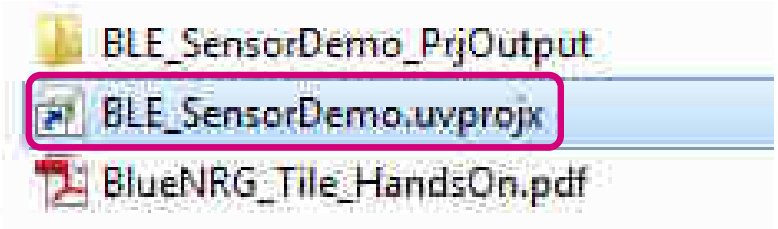
BLE_SensorDemo application

1. In Windows explorer **browse to the path:**



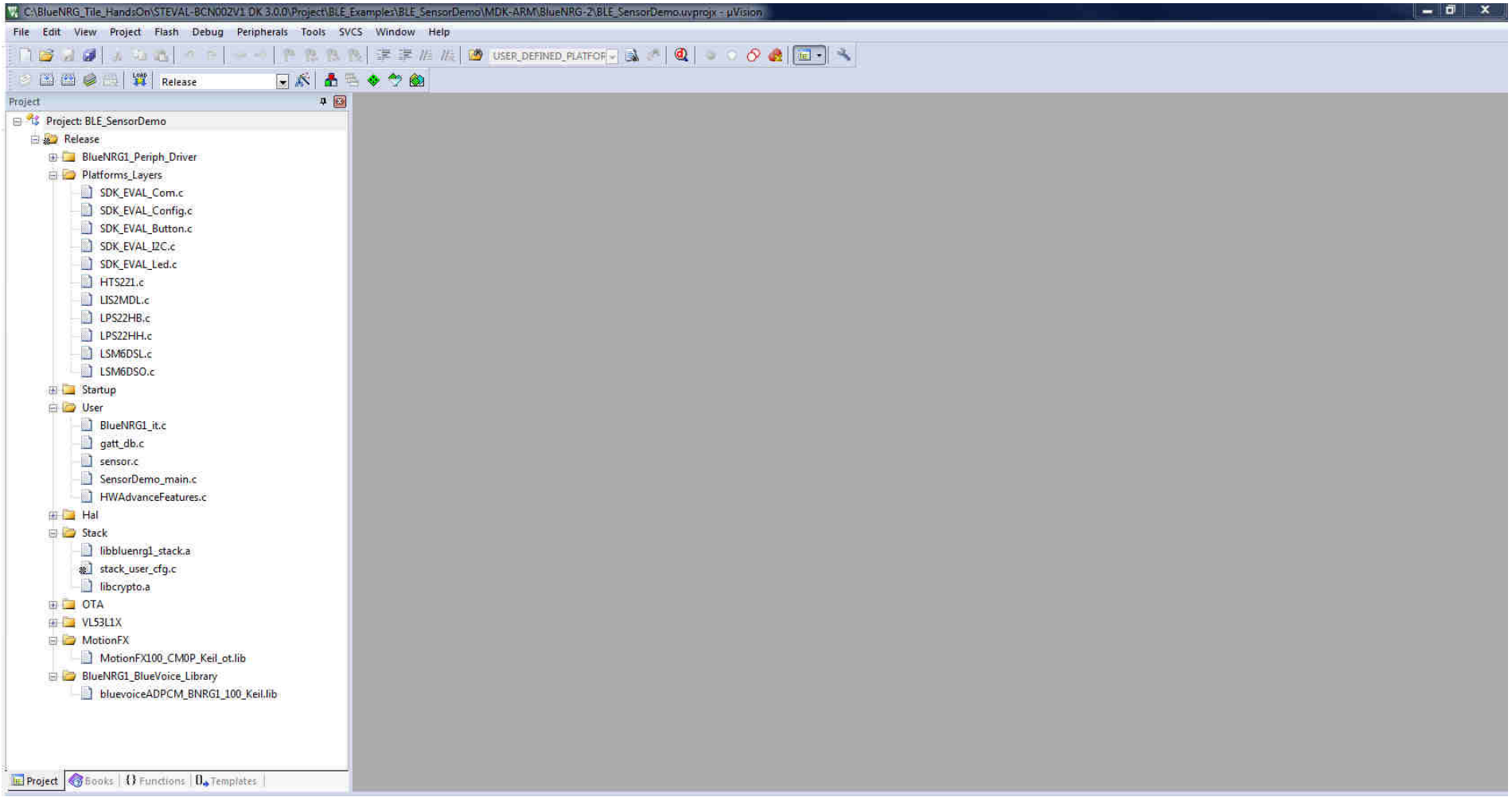
Go down till the root of installation folder and then in HandsOn folder

2. **Double click** on the **shortcut** "**BLE_SensorDemo.uvprojx**"



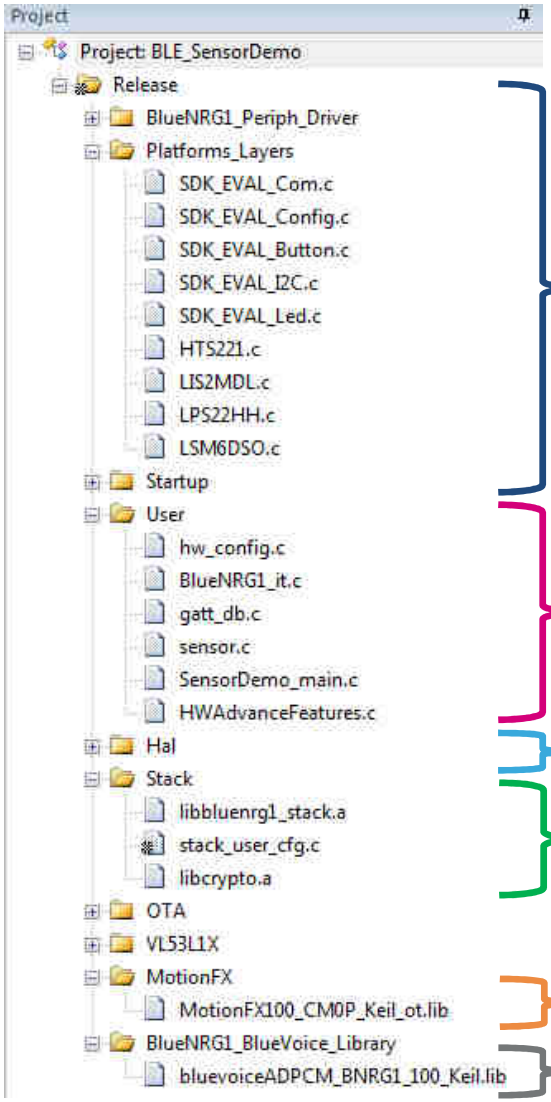
life.augmented

BLE_SensorDemo application



life.augmented

Application structure



HW peripherals drivers and platform layer files

Application source code
- **Main**
- **ATTRIBUTE Database**
- **Application**

Where we work today

Hardware Abstraction Layer and sleep management

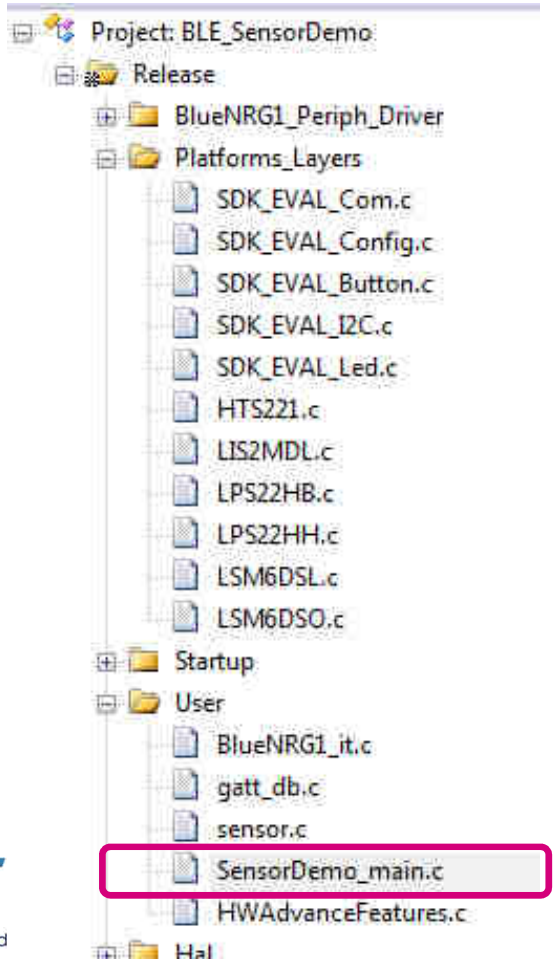
BLE Stack library – provided in binary format

Sensor Data Fusion library

Voice over BLE library

life.augmented

Open the BLE_SensorDemo main



1. **Open** the file **SensorDemo_main.c**
2. **Scroll down** to **line 35**

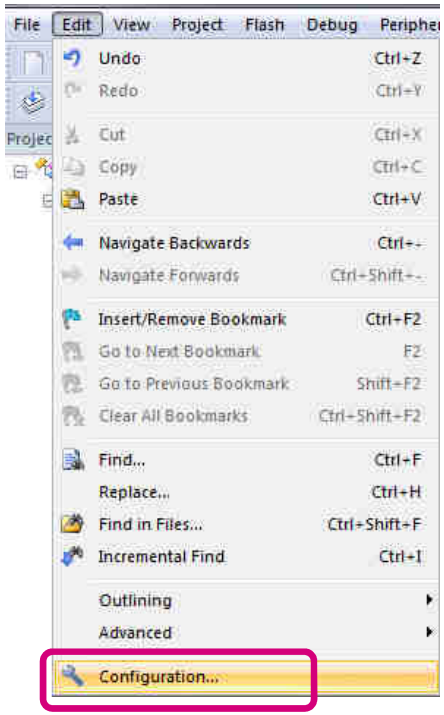
SensorDemo_main.c

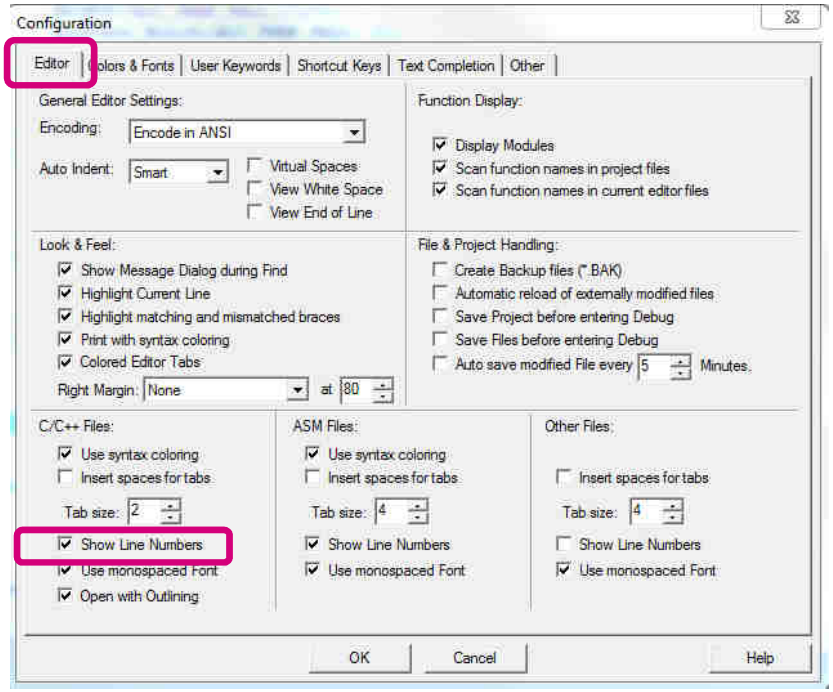
```
35 int main(void) {
36
37   /* System Init */
38   SystemInit();
39
40   /* Identify BlueNRG-2 platform */
41   SdkEvalIdentification();
42
43   /* Init the Hardware platform */
44   PlatformInit();
45
46   /* BlueNRG-2 stack init */
47   BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
48
49   /* Sensor Device Init */
50   Sensor_DeviceInit();
```

life.augmented

Show Line Numbers in KEIL IDE

1. **Go to “Edit->Configuration…”.**
2. In the **Editor** tab **click** on **“Show Line Numbers”**





life.augmented

A look at the main application

line 35 `int main(void) {`

`SystemInit();` Remap the vector table and configure all the interrupts priority

`SdkEvalIdentification();` Identifies STEVAL or custom PCB

`PlatformInit();` HW peripherals initialization

`BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);` BLE stack initialization

`Sensor_DeviceInit();` Sensors initialization

`Set_DeviceConnectable();` Set device in advertising

`while(1){` Start of while loop

`BTLE_StackTick();` Advances the stack FSM

`User_AppTick();` Advances the application FSM. **THIS IS DEVELOPERS USER SPACE!**

```
  } // end while(1)
}
```

life.augmented

BLE flow on the LAB

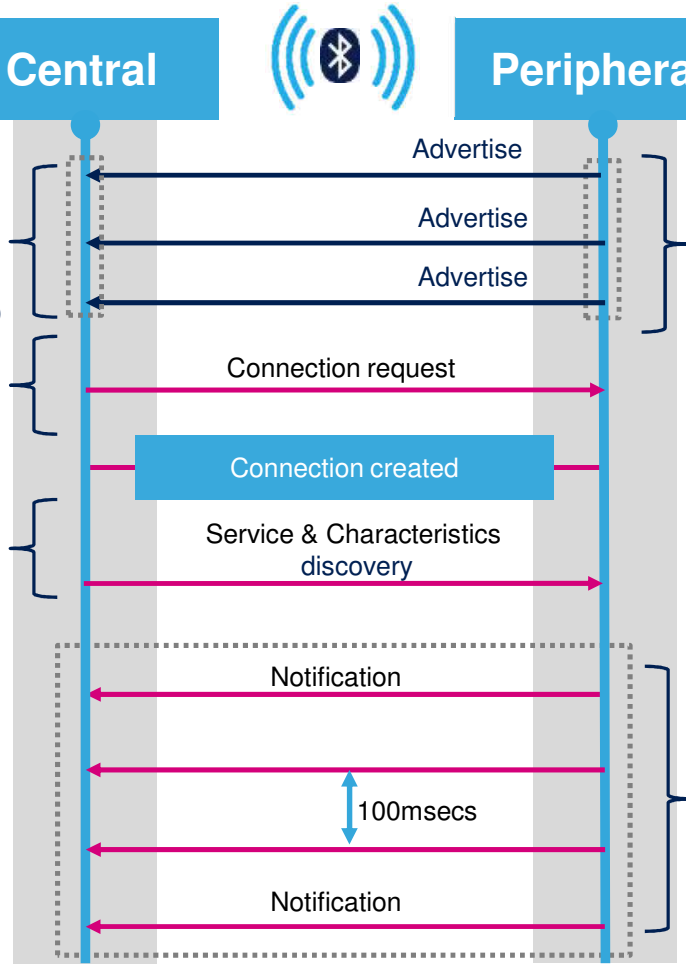
Step 2: ***Scan***

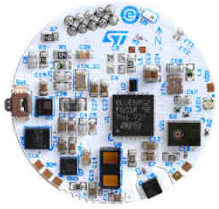
Master is in discovery mode looking for a ***specific*** slave to connect to

Step 3: ***Connection request***

Step 4: ***Services & Chars discovery***

Master (Client) starts the BLE Service and Characteristics discovery procedure to understand the Server ATT DB





Step 1: ***Advertising***

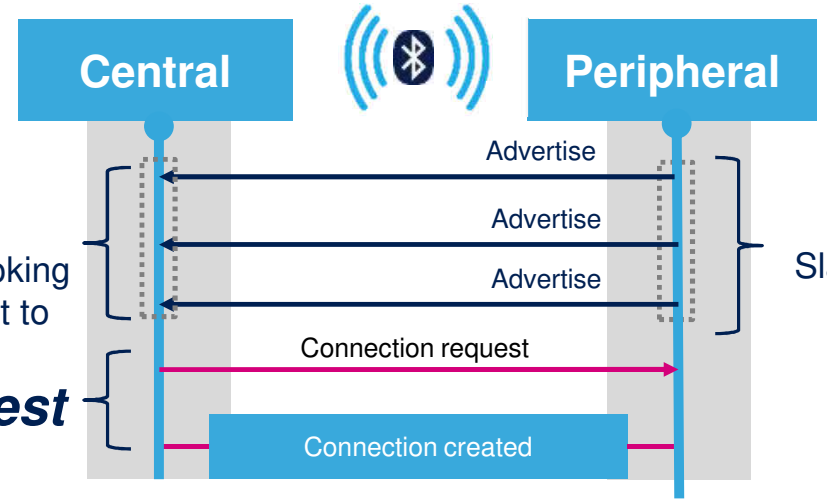
Slave is in Peripheral mode and sends ADV_IND PDU

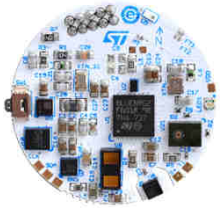
Step 5: ***Data flow***

Slave (Server) starts sending periodically (100 ms) to the master, **notifications packets** of sensors values (acc&gyro and pressure)

life.augmented

Advertising and scanning





Step 2: ***Scan***
Master is in discovery mode looking for a ***specific*** slave to connect to

Step 3: ***Connection request***

Step 1: ***Advertising***
Slave is in Peripheral mode and sends ADV_IND PDU

Master:
needs an app for discovering the slave device in advertising

Off-the-shelf app: e.g.

LightBlue® Explorer

https://itunes.apple.com/us/app/lightblue-explorer/id557428110?mt=8

Custom app: e.g.

life.augmented

Reminder: install ST BLE Sensor app

ON YOUR PHONE

- Look for “**ST BLE Sensor**” on the App Store or Google Play

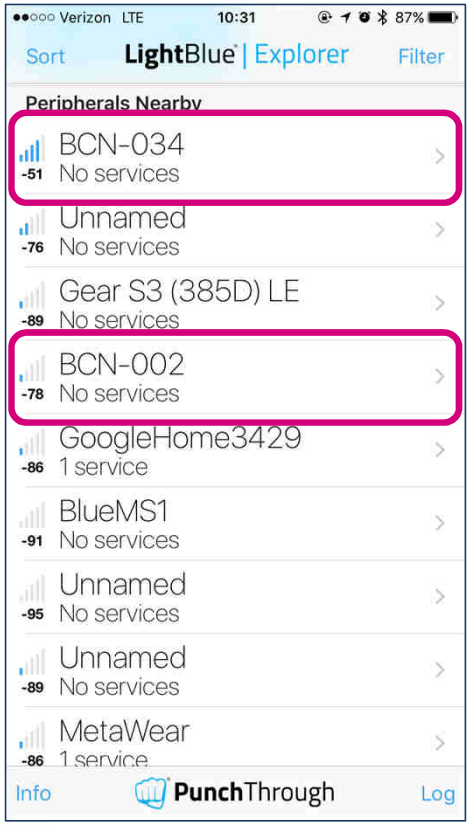
Download on the App Store

GET IT ON Google Play

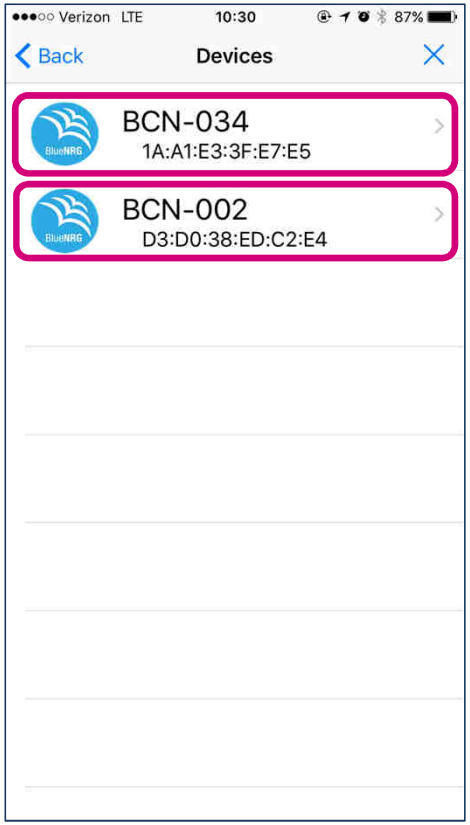
life.augmented

Scan results

LightBlue scan results



ST BLE Sensor scan results



Why some devices are not present in the ST BLE Sensor app?

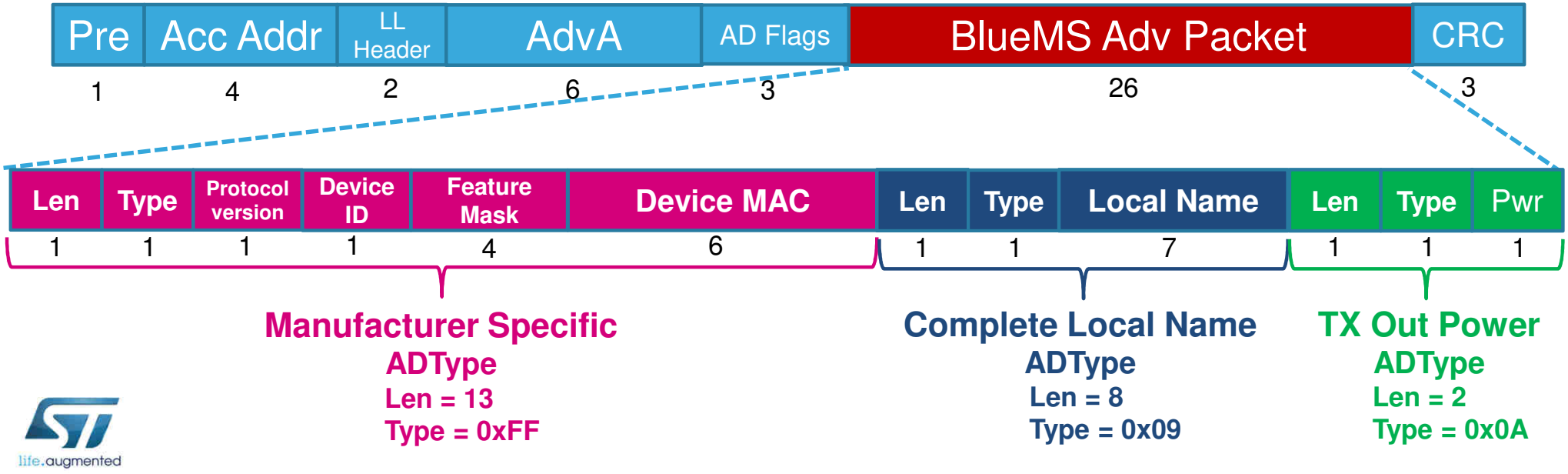
How devices will pop up in the ST BLE Sensor scan list?

life.augmented

ST BlueMS Protocol

- In order to be connected to the **ST BLE Sensor** app, a BLE peripheral SHALL **comply** with a specific advertising packet format

- **ST BlueMS protocol** specifies a **26-byte packet format** composed of Advertisement Types – **ADTypes** - compliant with BT SIG definitions



life.augmented

BlueST SDK for Android and iOS

Android

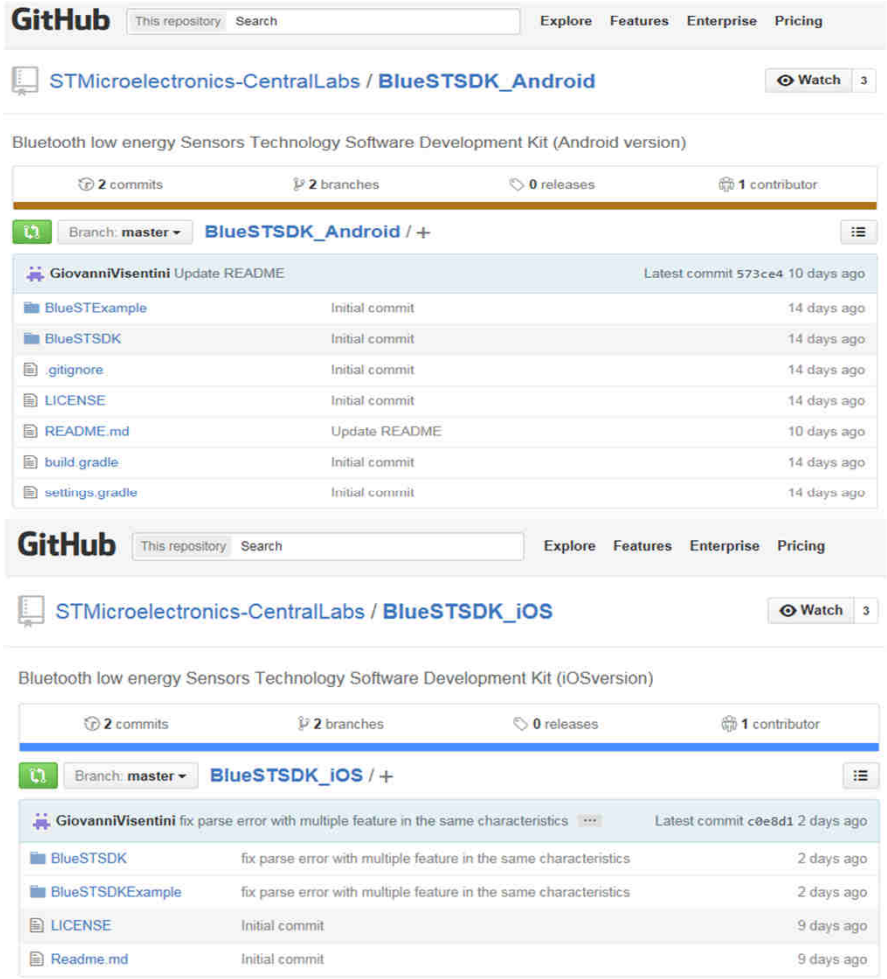
https://github.com/stmicroelectronics-centrallabs/bluestsdk_android

Source code for iOS and Android

available online and maintained on GitHub repos

iOS

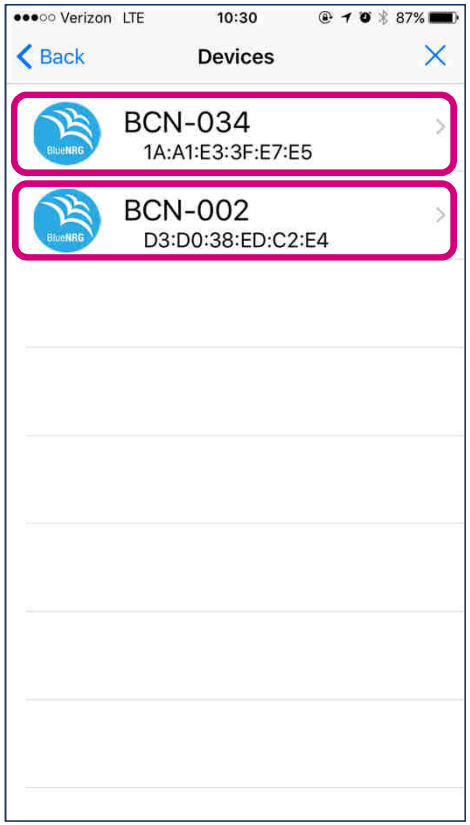
https://github.com/stmicroelectronics-centrallabs/bluestsdk_ios



life.augmented

ST BLE Sensor Scan results

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name



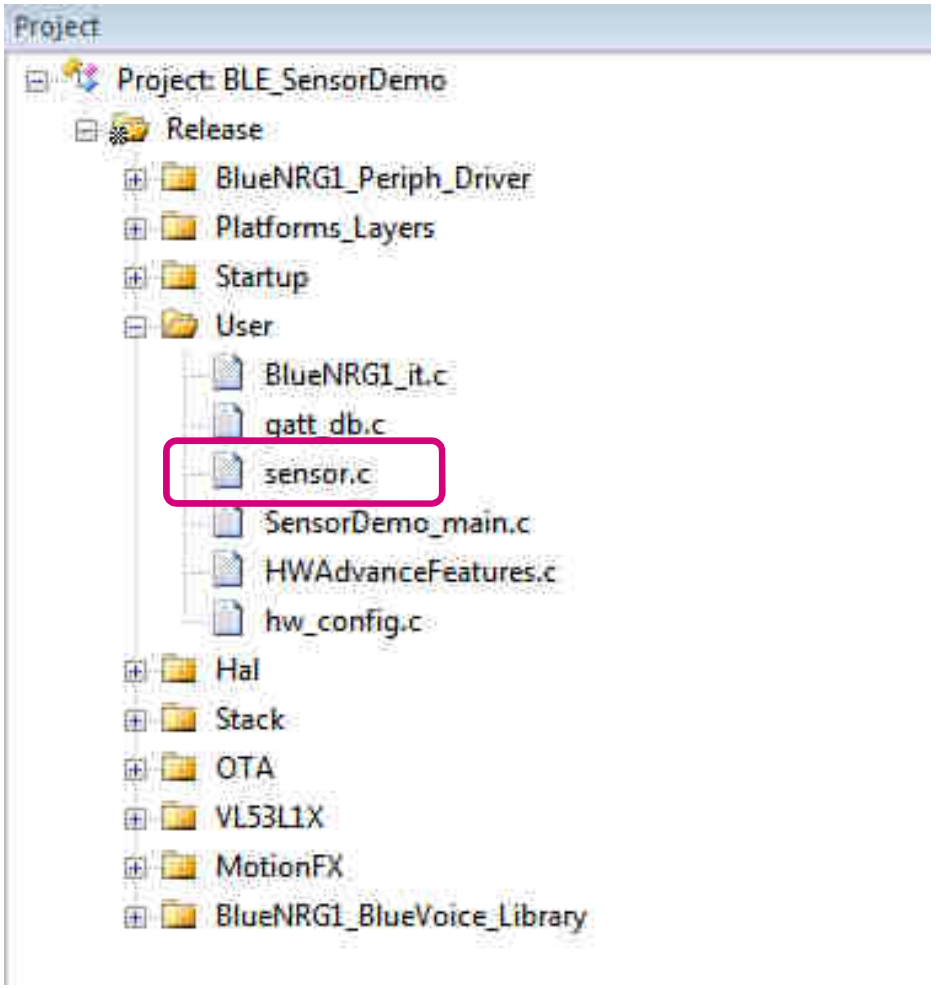
Each ST Platform is recognized by means of a Device ID (for STEVAL-BCN002V1 it is equal to 0x05)

Users can distinguish their node from the Local Name

BlueNRG

life.augmented

L2 STEP1: Customize your BlueNRG-Tile



L2 STEP1: Customize your BlueNRG-Tile

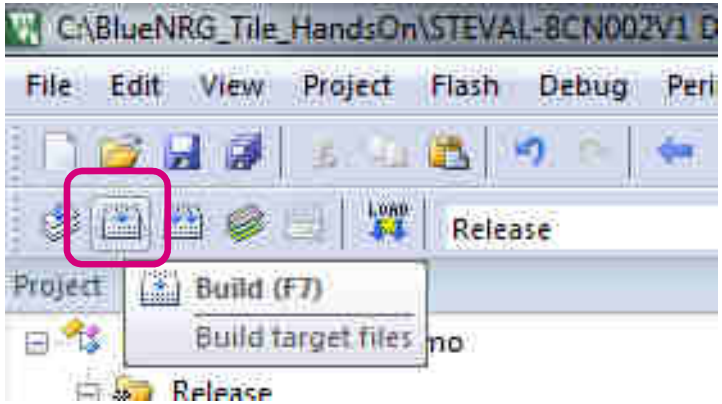
Modify the **local name** in the advertisement payload

1. In the file **sensor.c go to line 51**
2. **Modify X, Y** values in the string
   - **NOTE: COPY THE X,Y VALUES FROM YOUR STEVAL BLISTER**
   - *E.g. if on the box is written **X=3** and **Y=4**, then 'B','C','N','-','0','**3**','**4**'*

```
50  /* Define the BlueNRG-2 Name MUST be 7 char long */
51  #define NAME_ALLMEMS 'B','C','N','-','0','3','4'
```

Build the new code

1. Click on the **Build button** (top left corner) or hit **F7** on your keyboard
2. In the **Build Output** window (bottom) wait for the build to be completed.
   - **BLE_SensorDemo.bin** created
   - "**0 Error(s), 0 Warning(s)**" message appear

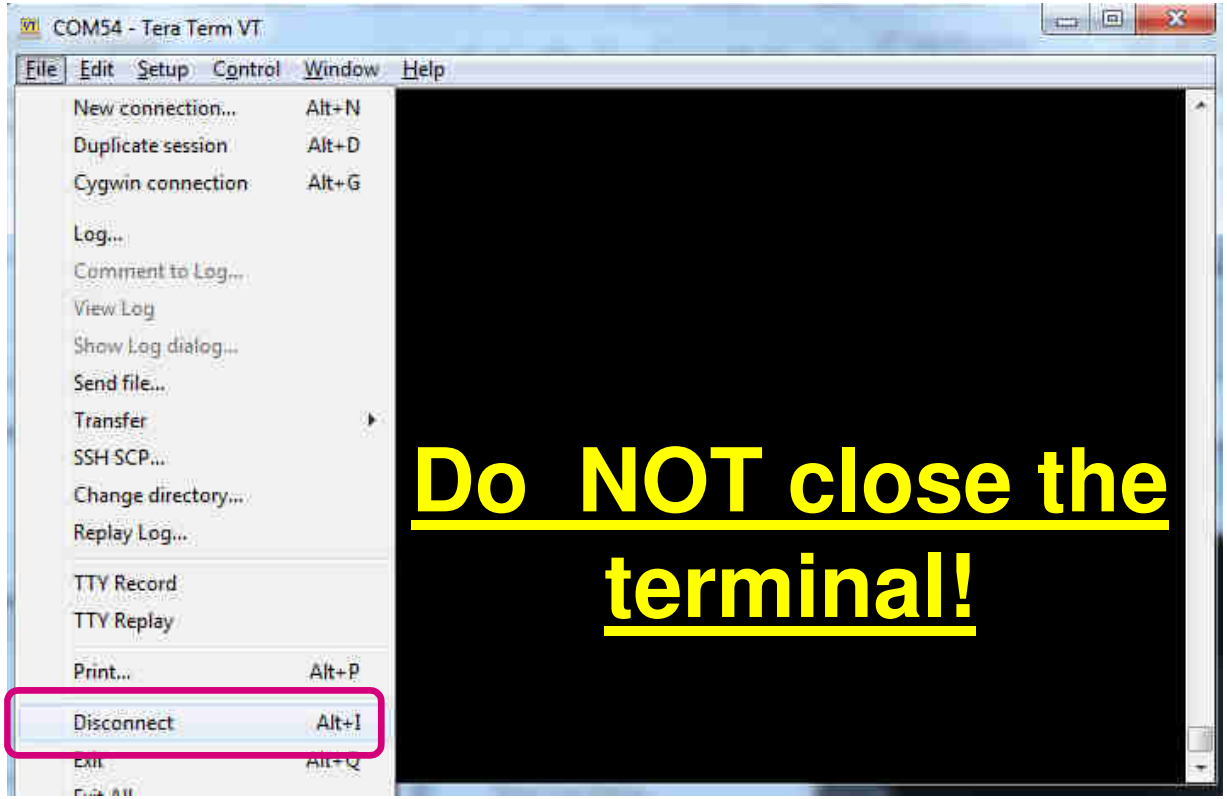



```
Build Output
compiling vl53l1_wait.c...
compiling vl53l1_register_funcs.c...
compiling vl53l1_platform.c...
linking...
Program Size: Code=121908 RO-data=1428 RW-data=1136 ZI-data=21252
FromELF: creating hex file...
After Build - User command #1: fromelf.exe --bin ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf --output ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.bin
"..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:12
```

life.augmented

Disconnect the serial terminal

1. Go back to **Tera Term**
2. Click on the **File->Disconnect**

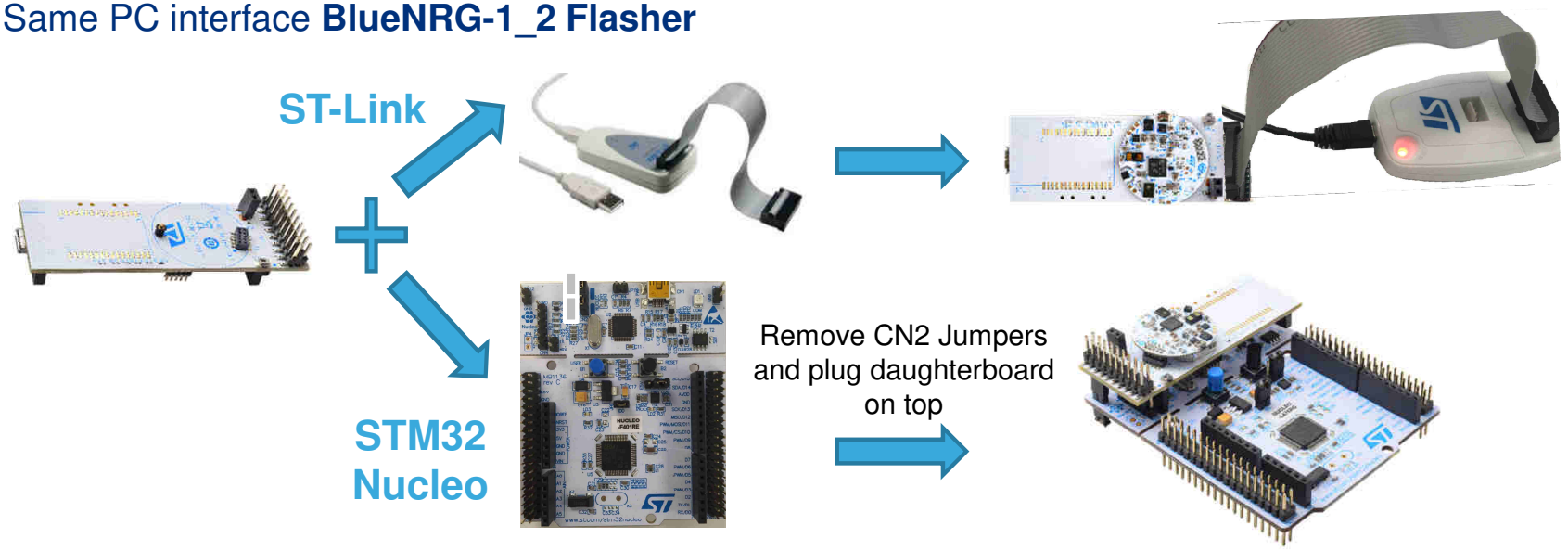


life.augmented

Programming embedded Flash

1. **UART** Bootloader
   - **ROM bootloader**. HW **activation through dedicate pin** (**DIO7**) configured for Boot
   - **PC interface** named **"BlueNRG-1_2 Flasher Utility"** available in the SW package

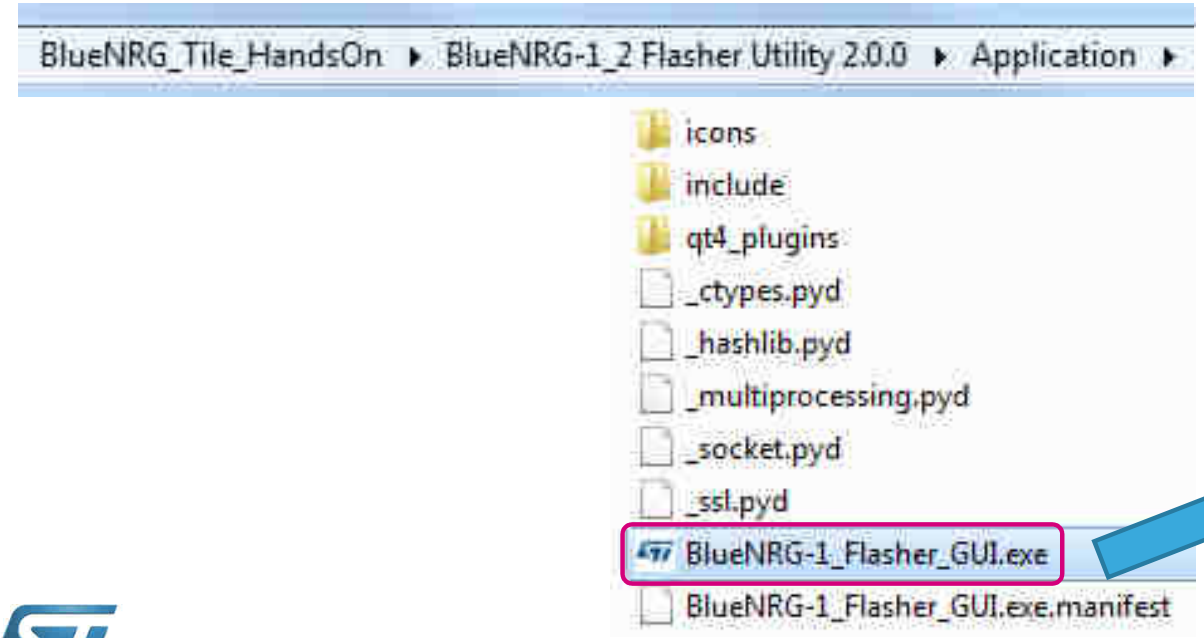
2. **SWD** interface + ST-LINK
   - Interface with the STEVAL-BCN002V1 through the **20-pin JTAG connector**
   - Same PC interface **BlueNRG-1_2 Flasher**

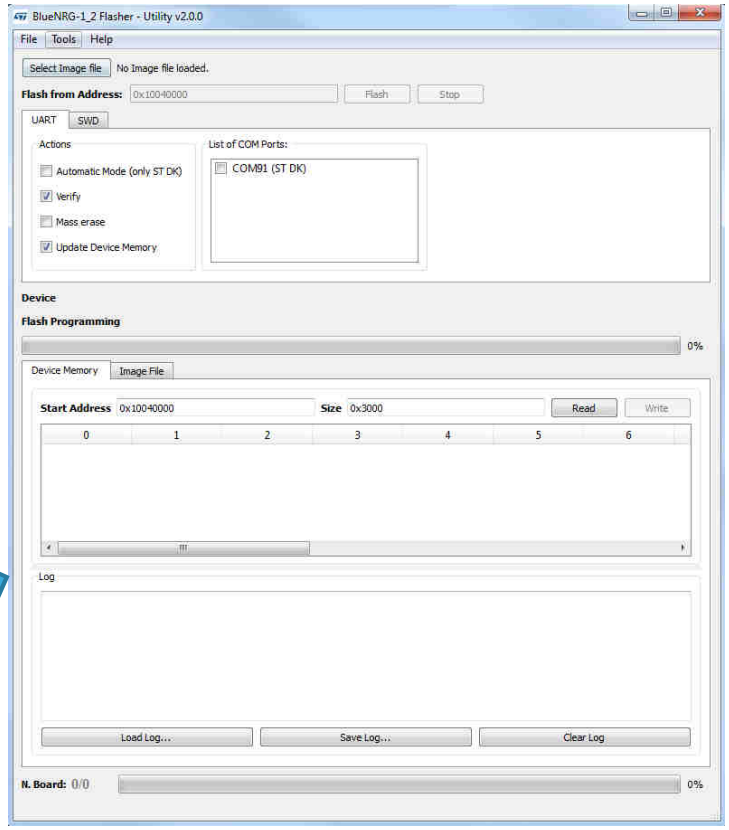


Open the Flasher Utility

1. **Go to**
   BlueNRG_Tile_HandsOn\BlueNRG-1_2 Flasher Utility 2.0.0\Application

2. **Double click** on **BlueNRG-1_Flasher_GUI.exe**

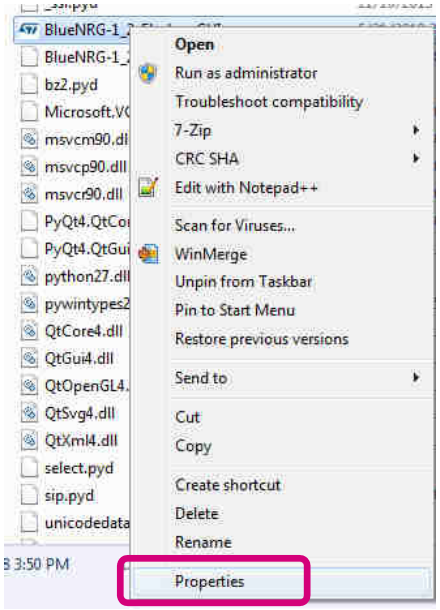


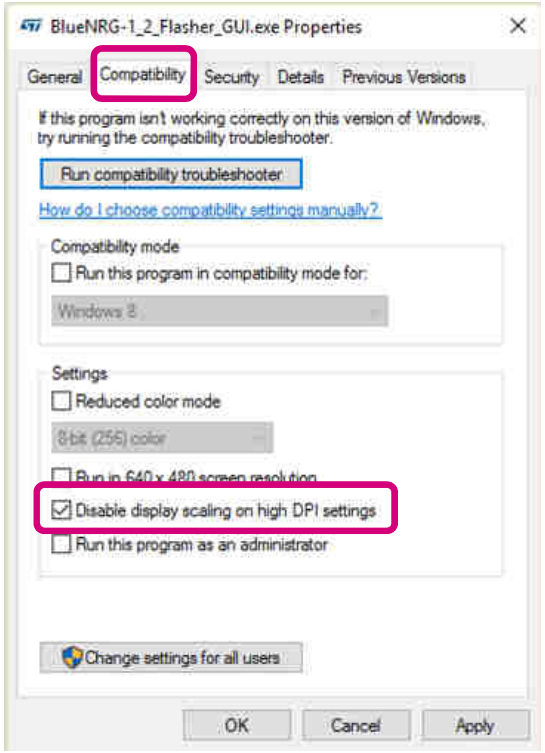


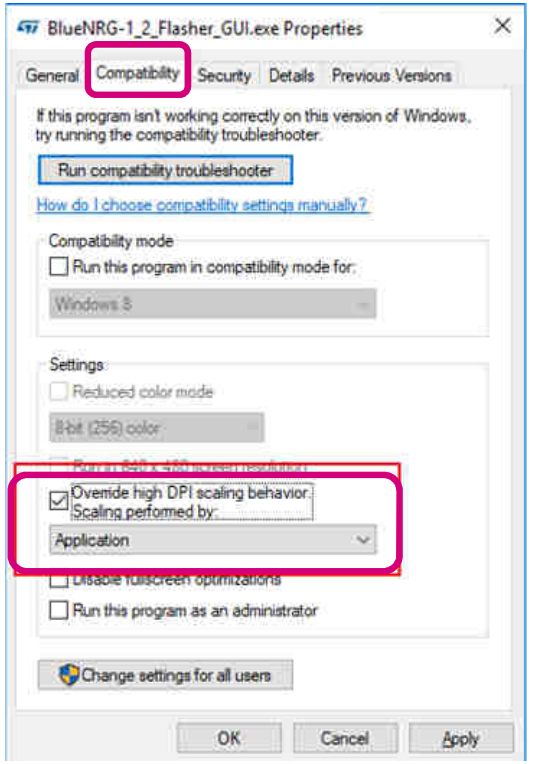
life.augmented

For HD screens…If you've problem with the resolution

- **Right Click** on the .exe file and **select** **Properties**
- In the **Compatibility** tab **select** either **Disable display scaling…** or **Override high DPI scaling… –> Application**

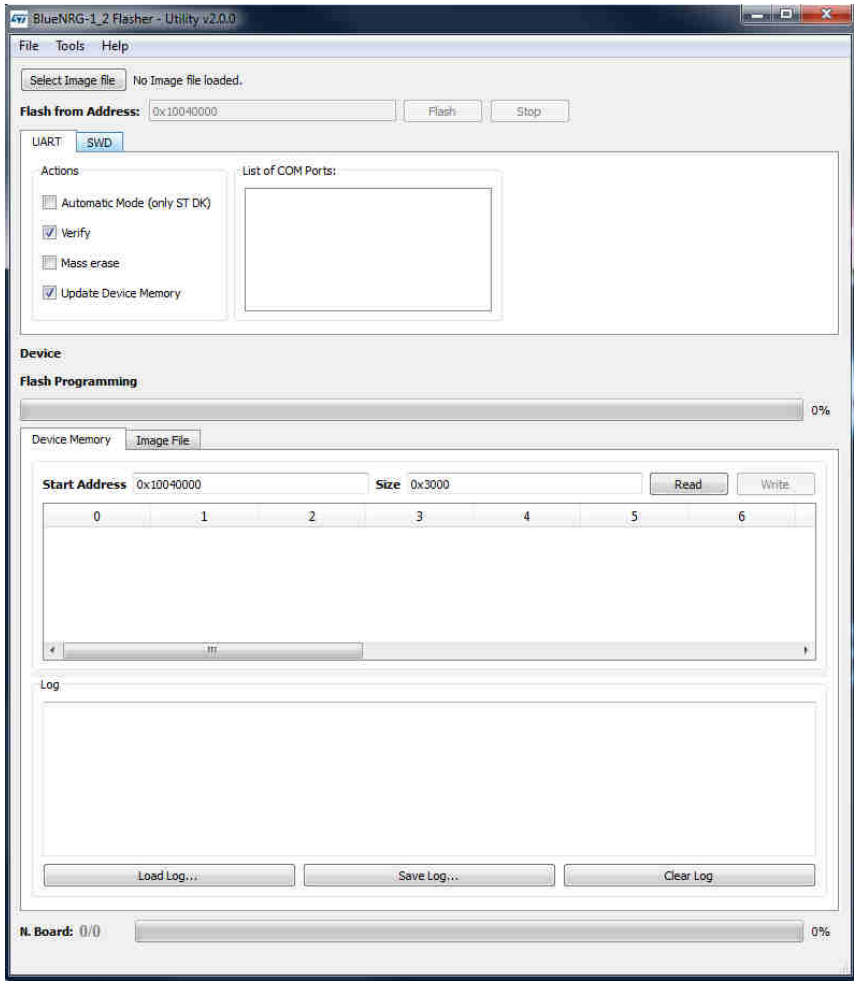




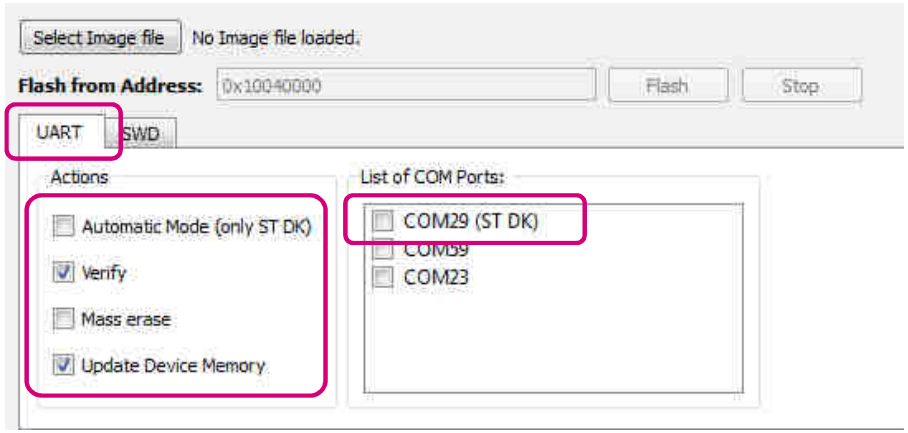


life.augmented

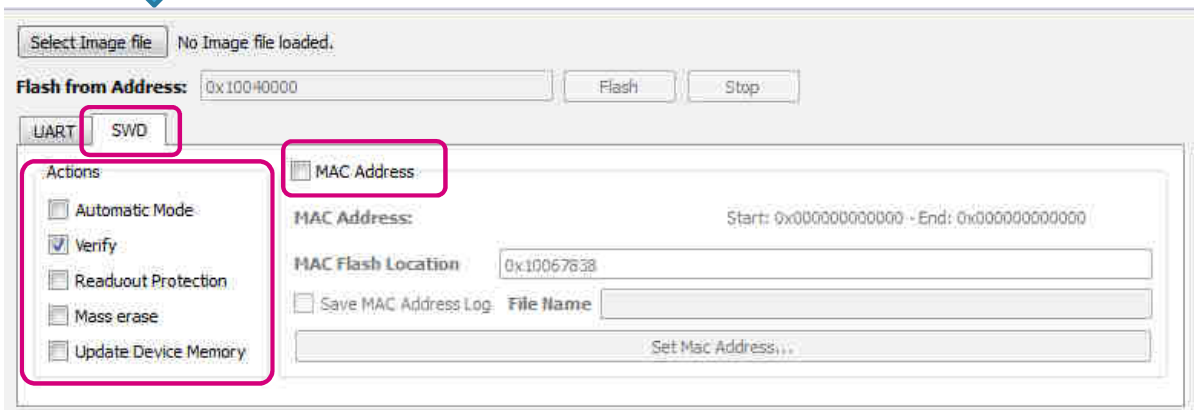
BlueNRG Flasher Utility



UART



SWD

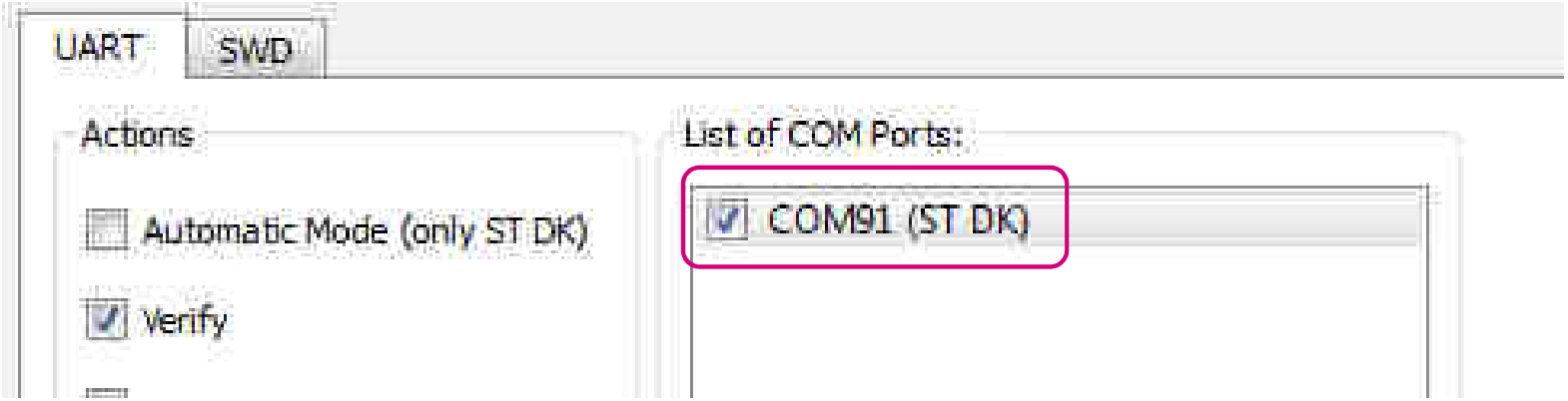


MAC Addresses

- BLE addresses (6 bytes) can be either **public** or **random.**
  - **Public MAC addresses** (6 bytes- 48-bit address) uniquely identifies a BLE device, and are defined by IEEE.
    - The first 3 bytes identify the company trough Organizationally Unique Identifier (OUI).
    - Up to 2^24 addresses from the remaining 3 bytes of the public address
    - Public address has to be stored on a specific device Flash location and to set within the BLE stack through the *aci_hal_write_config_data()*.
  - Random can be either **static** or **private**
    - **Static random addresses**: randomly generated and need to follow some rules defined in the BT specifications (refer to the BT Core specs v4.2, Vol 6, Part B, Section 1.3.2 for the details).
      - NOTE: BLE radio stack generates it autonomously and stores in flash: developer doesn't need to care about programming the address.
    - **Private random addresses** are used when privacy is enabled and according to the BT Core specs.

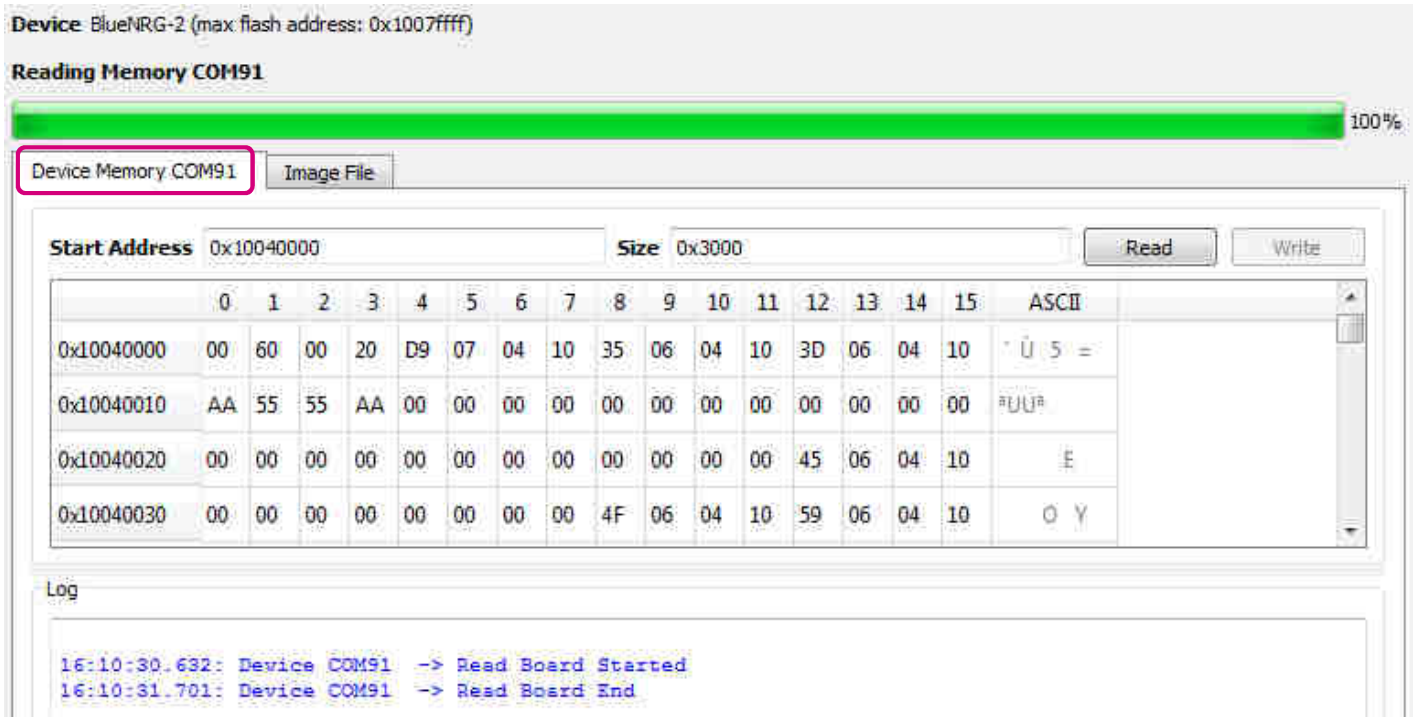
Flash the BlueNRG-2 1/6

1. **Select the COM port** labeled **(ST DK)**



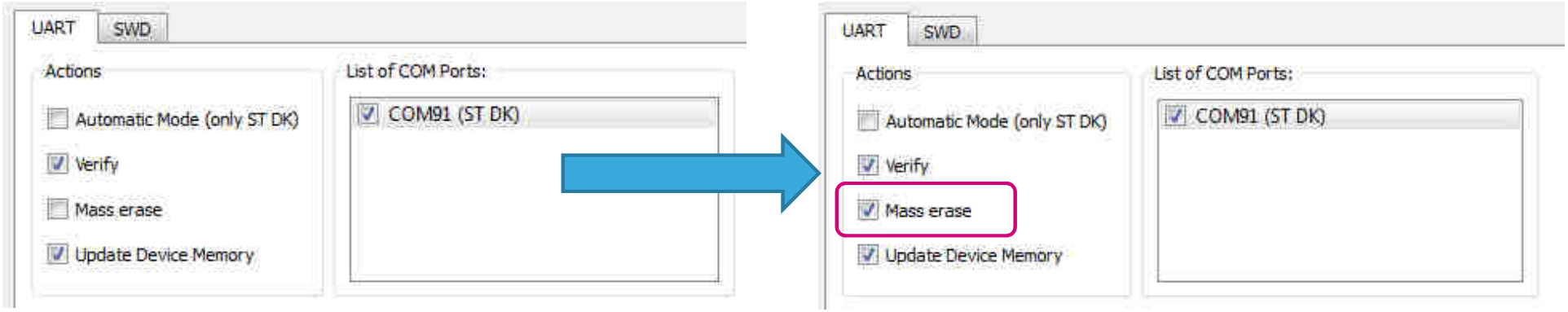
Flash the BlueNRG-2 2/6

1. **Device Memory** will populate with data



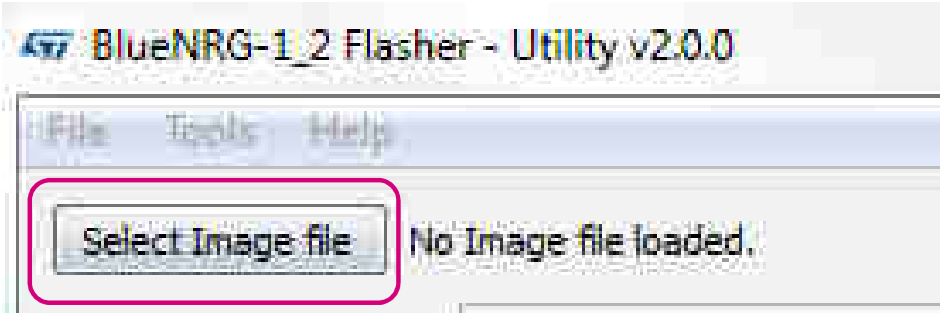
Flash the BlueNRG-2 3/6

1. Select **Mass Erase**



Flash the BlueNRG-2 4/6

1. **Click** on the **Select Image file** button

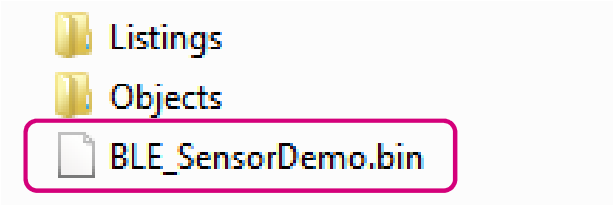


2. And **browse** the following **path**

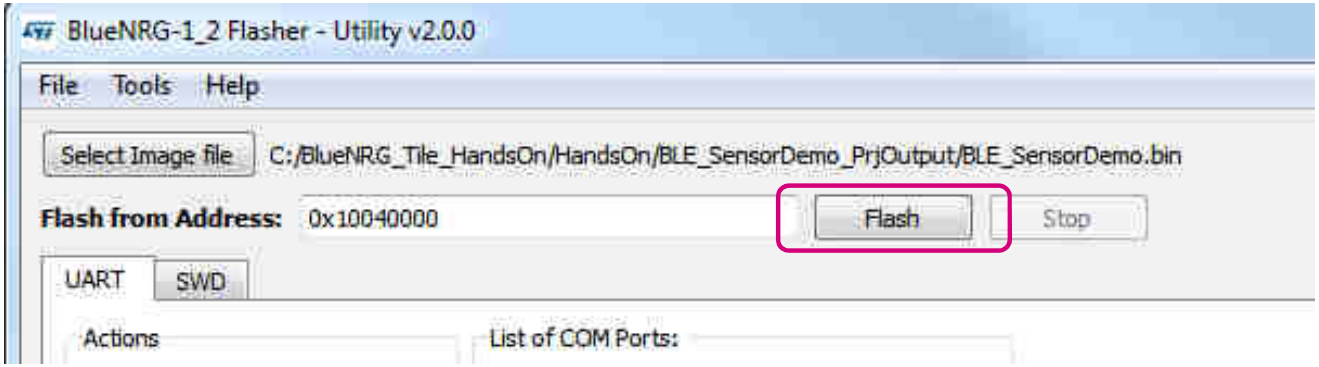
life.augmented

Flash the BlueNRG-2 5/6

1. **Select BLE_SensorDemo.bin and click Open**

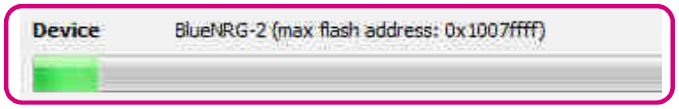


2. **Click on the Flash button**

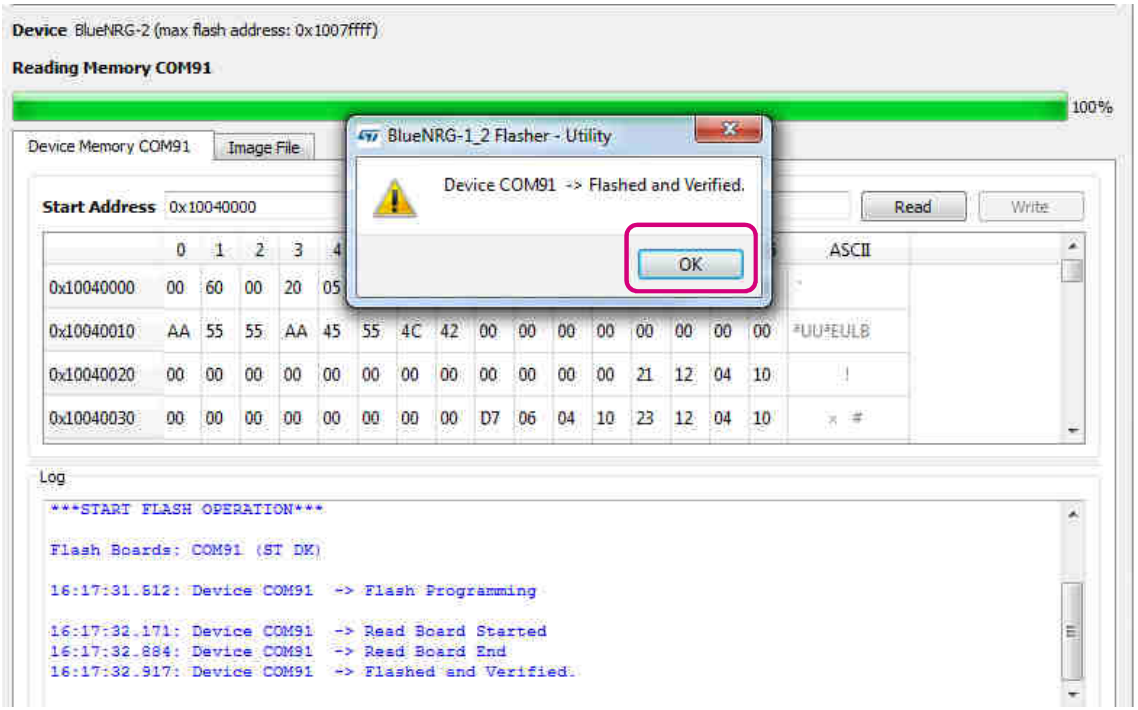


Flash the BlueNRG-2 6/6

1. Flashing starts: **green bar** proceeding



2. **Wait** for the **pop-up** window and **click** on **OK**

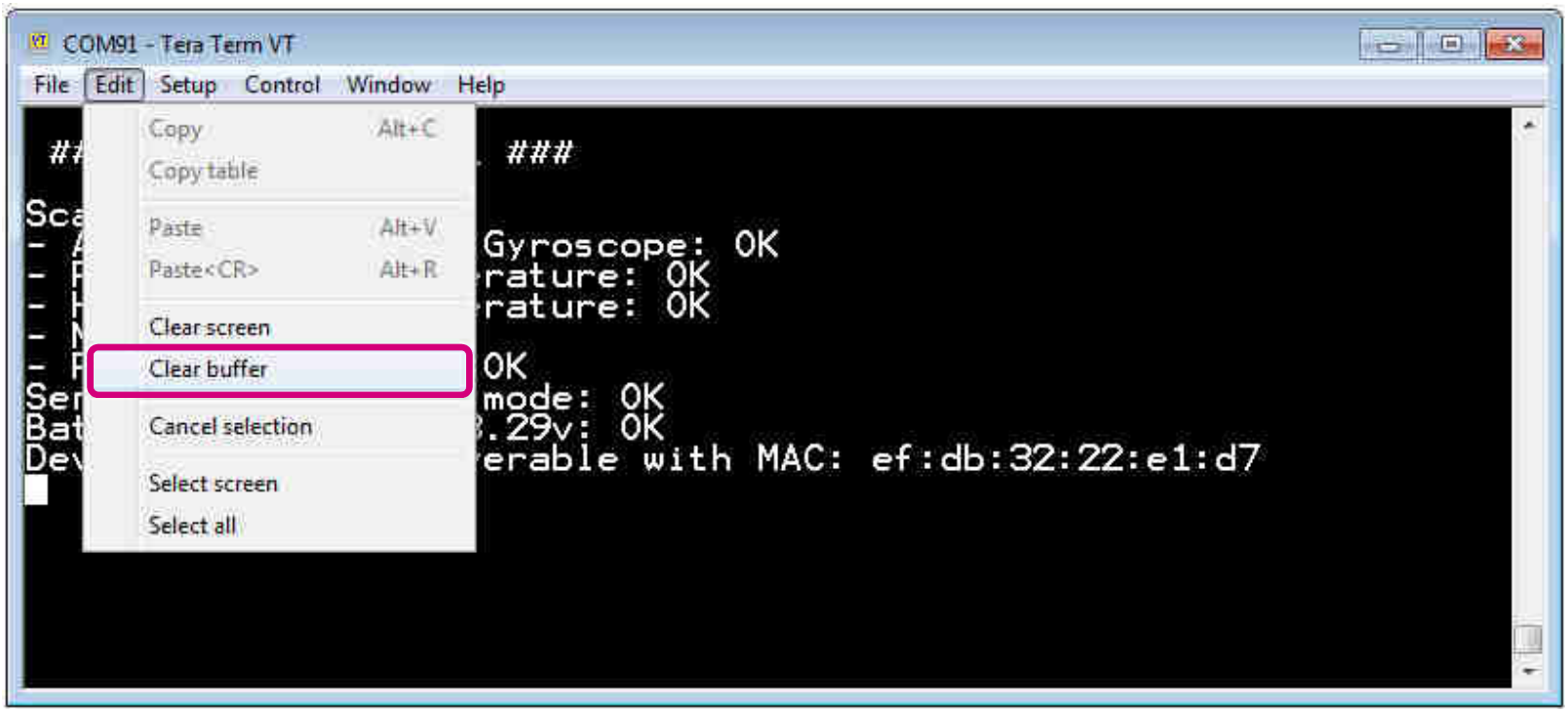


Do NOT close the Utility!

life.augmented

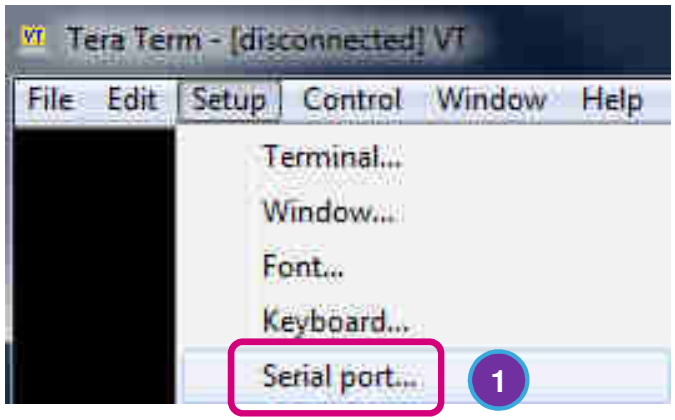
Clean Buffer in the serial terminal

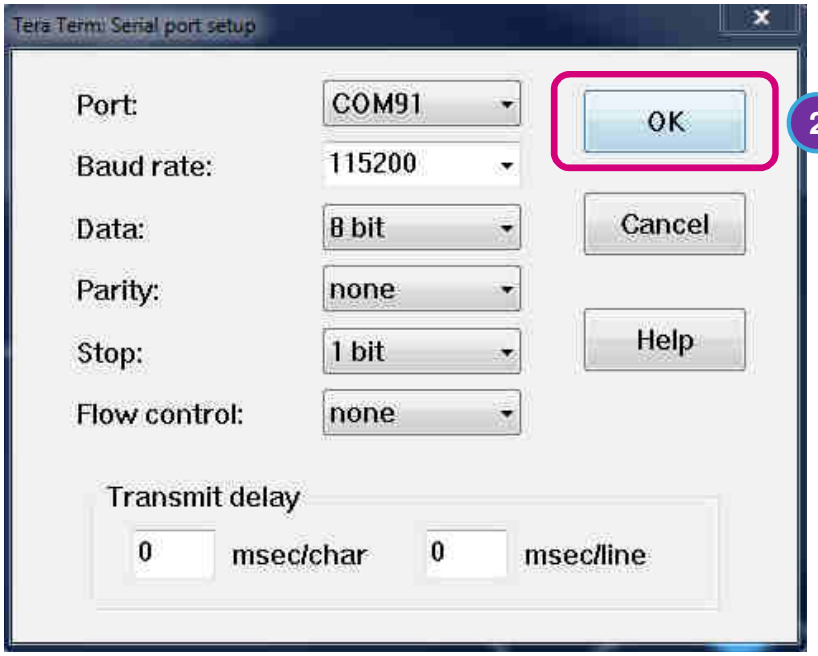
1. In **Tera Term** in order to have the terminal clean, **go to Edit -> Clear buffer**



Reconnect the serial terminal

1. **Click Setup -> Serial port…**
2. Serial port should be already configured.
   Just need to **click** on **OK**

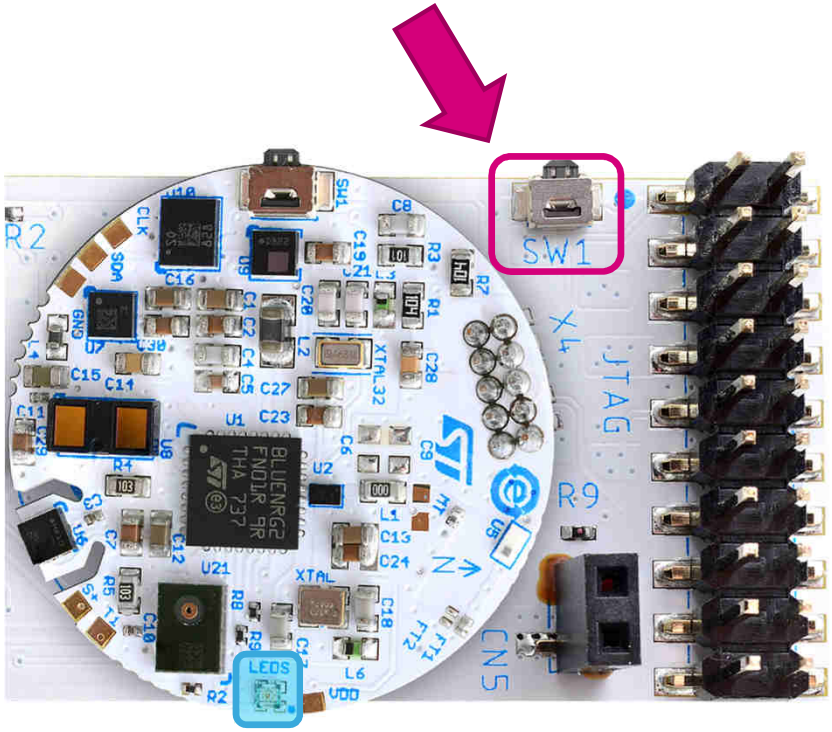


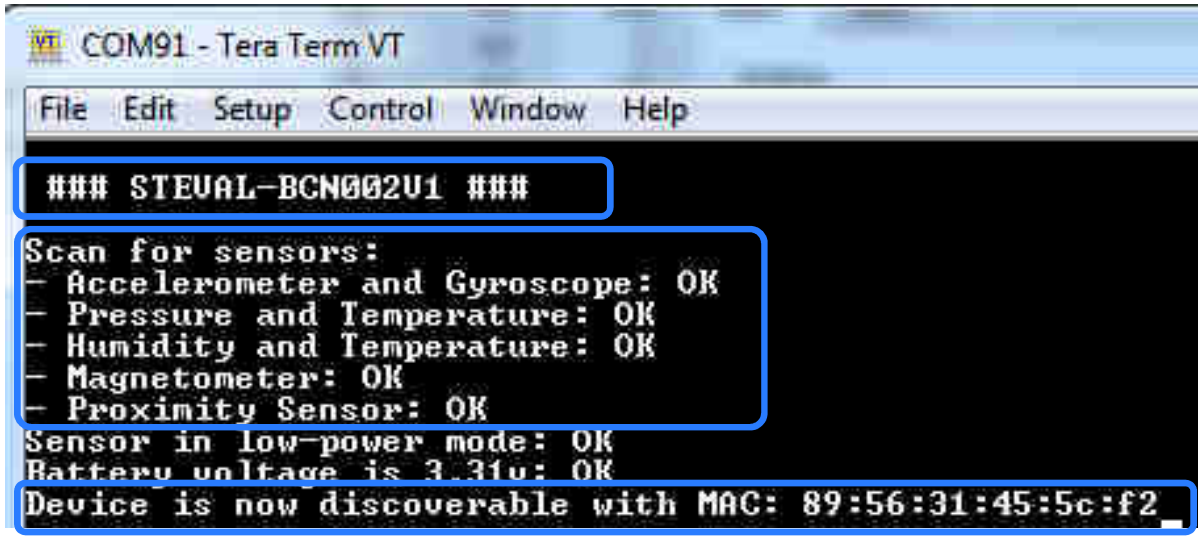


life.augmented

Sanity Check on serial port

Push **SW1 button** on the daughterboard -> **LED blinking Blue**





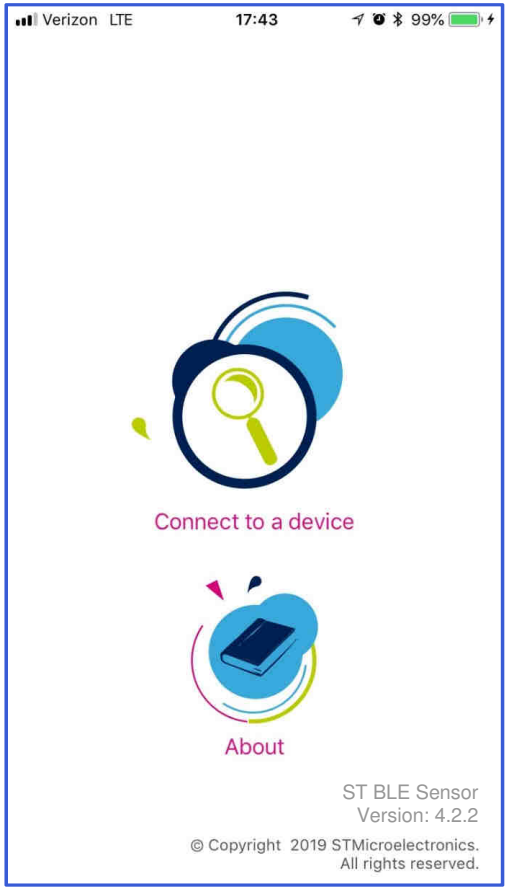
life.augmented

Open the ST BLE Sensor App

Download on the App Store

GET IT ON Google Play

Launch the **ST BLE Sensor** app previously installed



Note: in the following slides all the pictures are referred to the iOS version of the ST BLE Sensor app. The Android version is slightly different

life.augmented

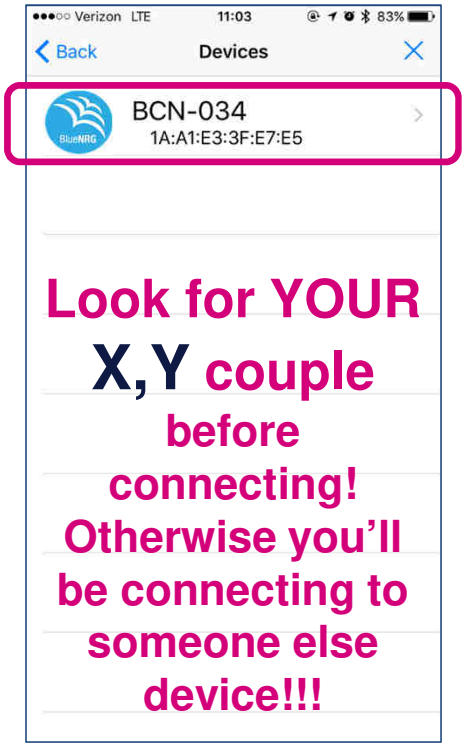
Connect using the ST BLE Sensor App

1

Tap “Connect to a device”

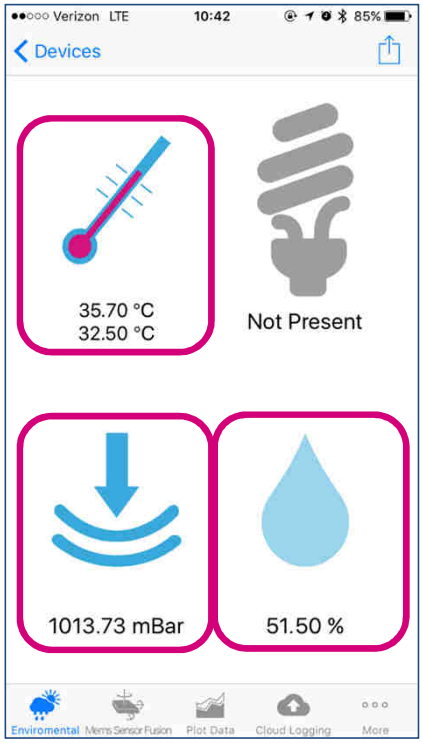
2

Select your STEVAL-BCN002V1



3

You are connected

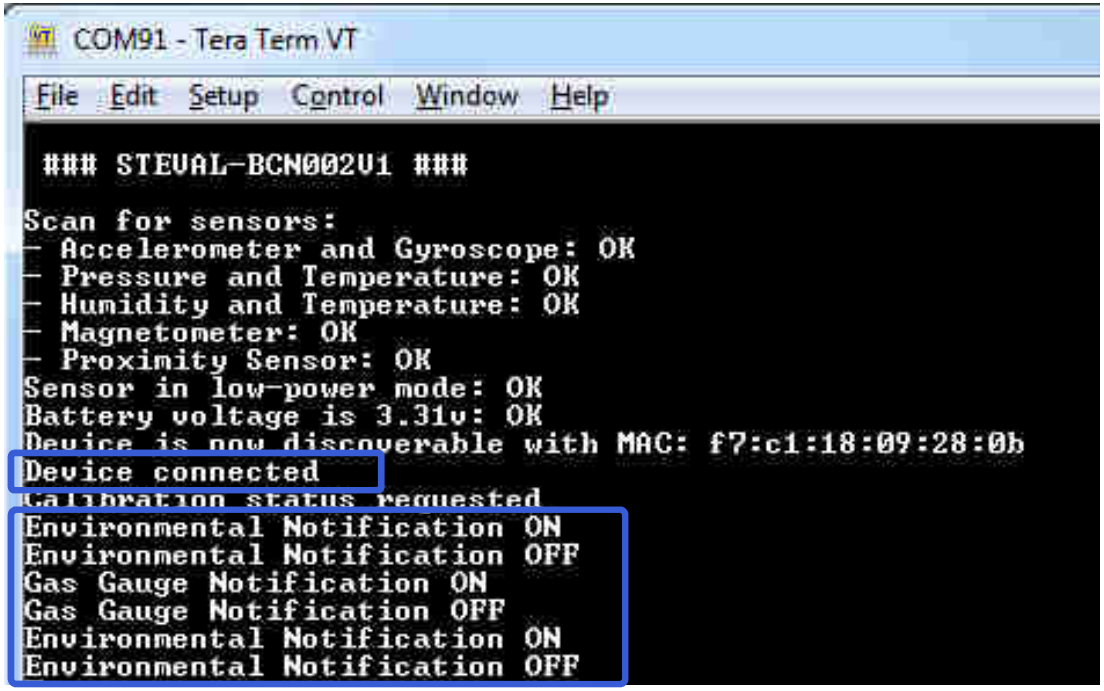


Blow air on the Temperature sensor to change the temperature and humidity values

life.augmented

Tera Term output

- ***"Device connected"*** will appear as connection is created
- Each time user moves on different tabs/screens in the app:
  - **BLE notifications** on different BLE characteristics **are enabled**.
  - **Environmental and Gas Gauge/Battery characteristics** are involved



life.augmented

ST BlueMS Protocol

In file **sensor.c** at line **413**

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name	Len	Type	Pwr

SensorDemo_main.c | sensor.c

```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                    /* Length of AD type Transmission Power */
415   0x0A, 0x00,           /* Transmission Power = 0 dBm */
416   8,                    /* Length of AD type Complete Local Name */
417   0x09,                 /* AD type Complete Local Name */
418   NAME_ALLMEMS,         /* Local Name */
419   13,                   /* Length of AD type Manufacturer info */
420   0xFF,                 /* AD type Manufacturer info */
421   0x01,                 /* Protocol version */
422   0x04,                 /* Device ID: 0x04 = STEVAL-BCN002V1 Board */
423   0x00,                 /* Feature Mask byte#1: LAB3 0x20 (Led) / LAB5 0x22 (Led+Prox) / LAB7 0x6A (Led+Prox+ADPCM Sync+ADPCM Audio) */
424   0x1F,                 /* Feature Mask byte#2: LAB4 0x9F (Acc+Press+Hum+Temp+Batt+Temp2) / LAB5 0xFF (Acc+Gyro+Mag+Press+Hum+Temp+Batt+Temp2)*/
425   0x00,                 /* Feature Mask byte#3: LAB4 0x04 (AccEvents) / LAB5 0x05 (AccEvents+iNemo Compact) */
426   0x00,                 /* Feature Mask byte#4: LAB5 0x40 (eCompass) */
427   0x00,                 /* BLE MAC start */
428   0x00,
429   0x00,
430   0x00,
431   0x00,
432   0x00                  /* BLE MAC stop */
433   };
```

3 bytes TX output power

9 bytes Local Name

13 bytes Manufacturer Specific

life.augmented

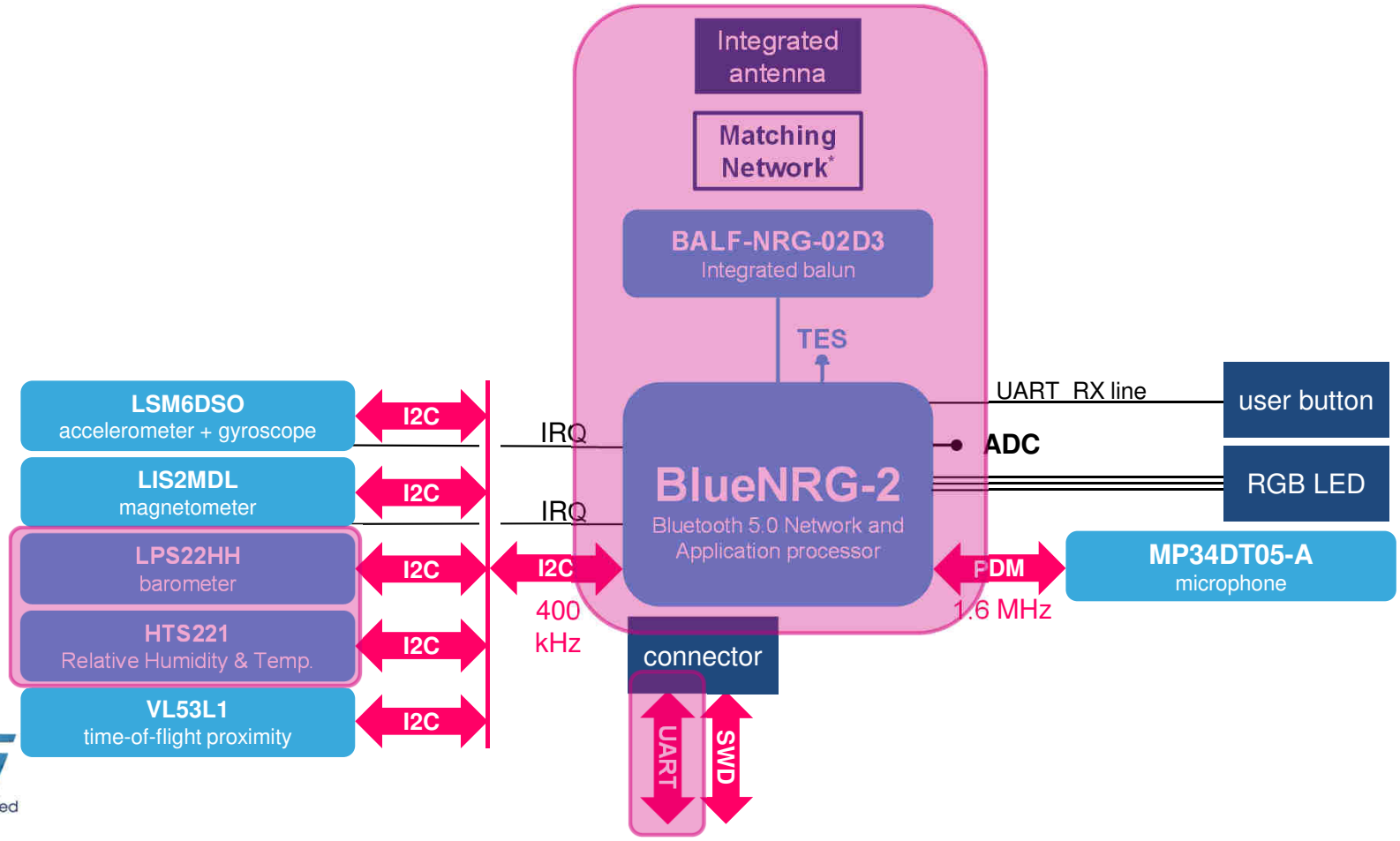
Feature Mask

4 Bytes

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name	Len	Type	Pwr

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	0x00
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0x1E
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	0x00
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x00

STEVAL-BCN002V1 Block Diagram



life.augmented

BlueMS Protocol – complete Feature Mask

4 Bytes

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name	Len	Type	Pwr

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	PoA	ADPC	MicLevel	Proximity	Lux	0x6A
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0xFE
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamforming	AccEvent	FreeFall	SensFusC	0x05
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x40

After all labs

life.augmented

Advertising Data

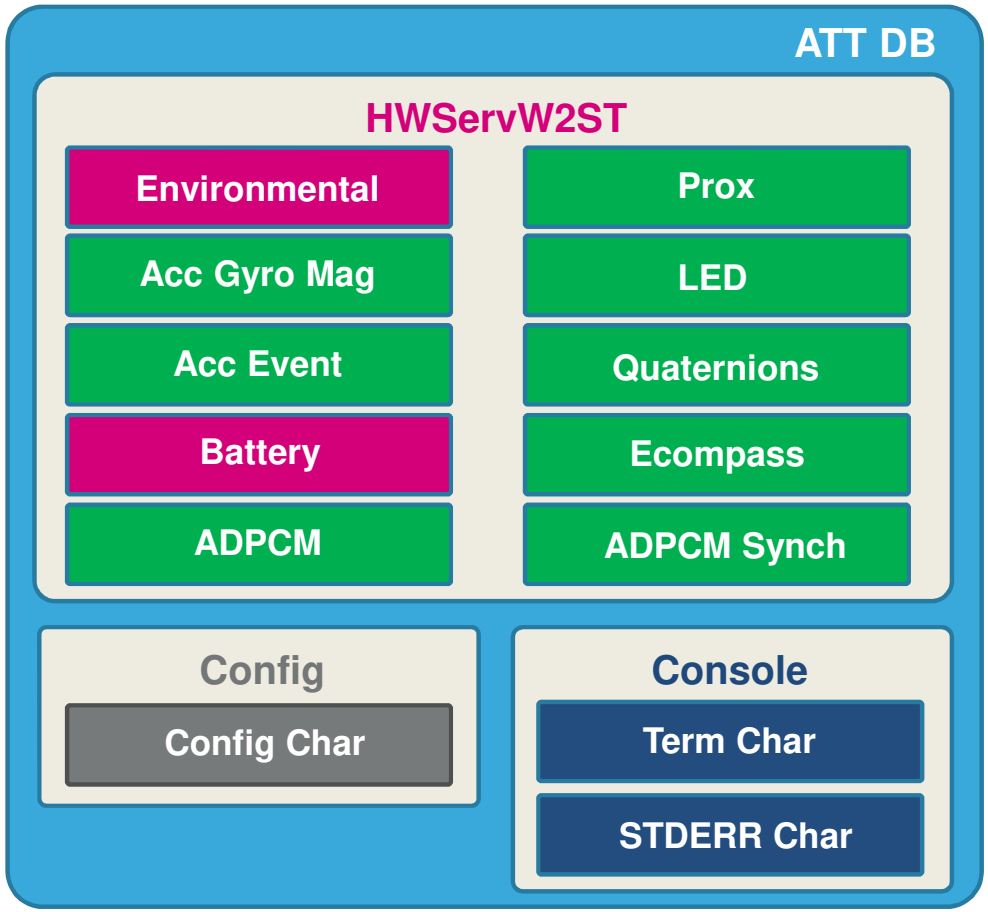
- In file **sensor.c** at line **413**, this is how the **Feature Mask** looks like

```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                      /* Length of AD type Transmission Power */
415   0x0A, 0x00,             /* Transmission Power = 0 dBm */
416   8,                      /* Length of AD type Complete Local Name */
417   0x09,                   /* AD type Complete Local Name */
418   NAME_ALLMEMS,           /* Local Name */
419   13,                     /* Length of AD type Manufacturer info */
420   0xFF,                   /* AD type Manufacturer info */
421   0x01,                   /* Protocol version */
422   0x05,                   /* Device ID: 0x05 = STEVAL-BCN002V1 Board */
423   0x6A,                   /* Feature Mask byte#1: LAB3 0x20 (LED) / LAB5 0x22 (Led+Pr
424   0xFE,                   /* Feature Mask byte#2: LAB4 0x9E (Acc+Press+Hum+Temp+Batt)
425   0x05,                   /* Feature Mask byte#3: LAB4 0x04 (AccEvents) / LAB5 0x05 (
426   0x40,                   /* Feature Mask byte#4: LAB5 0x40 (eCompass) */
427   0x00,                   /* BLE MAC start */
428   0x00,
429   0x00,
430   0x00,
431   0x00,
432   0x00                    /* BLE MAC stop */
433   };
```

life.augmented

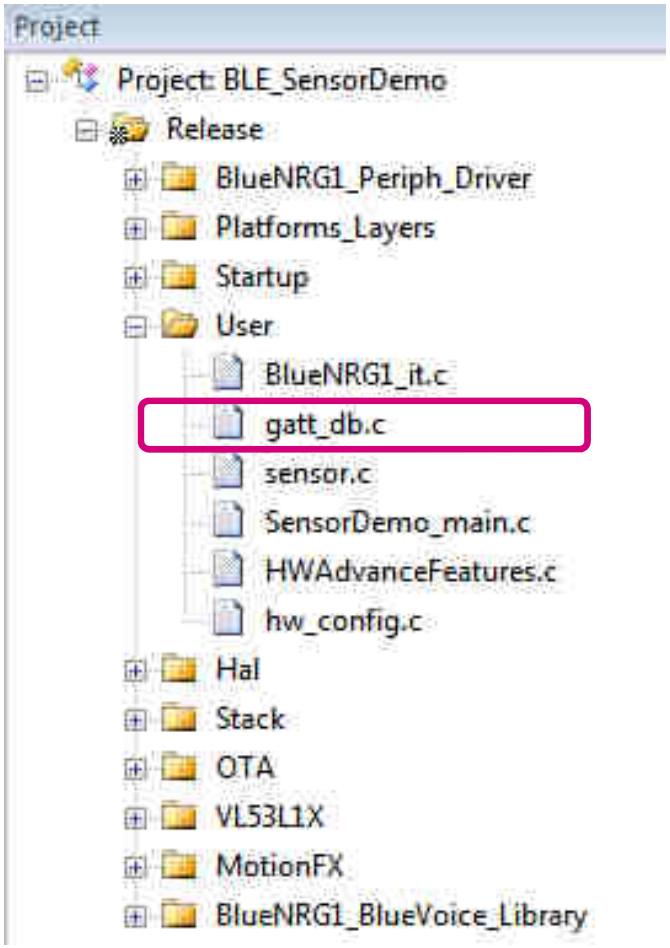
How Feature Mask is mapped onto ATT DB?

- **3 services:** HWServW2ST, Config, and Console
- **Each bit** of the 4-byte Feature Mask in the Advertising packet **corresponds to a HW/SW feature**
- In the Server ATT DB **a BLE Characteristic has to be added for each bit of the Advertising Feature Mask**



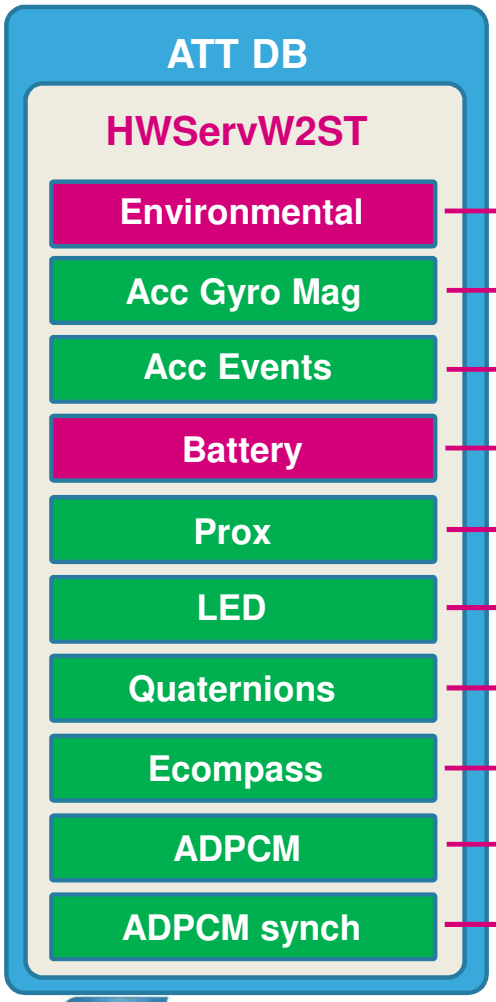
life.augmented

BLE_SensorDemo ATT DB



The **ATT DB** is defined and created in the file **gatt_db.c**

BLE_SensorDemo ATT DB



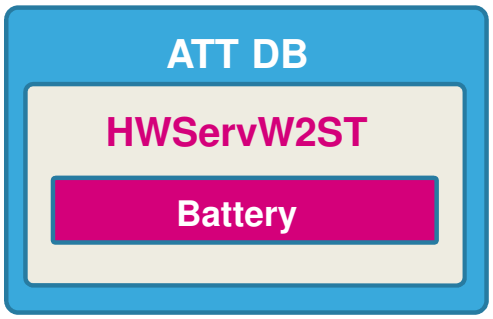
Len	PROPERTIES	UUID TYPE	UUID VALUE
12	N,R	128-bit	001D0000000111e1ac360002a5d5c51b
20	N	128-bit	00E0000000111e1ac360002a5d5c51b
5	N,R	128-bit	0000040000111e1ac360002a5d5c51b
9	N,R	128-bit	0002000000111e1ac360002a5d5c51b
4	N,R	128-bit	0200000000111e1ac360002a5d5c51b
3	N,R	128-bit	2000000000111e1ac360002a5d5c51b
8	N	128-bit	0000010000111e1ac360002a5d5c51b
4	N	128-bit	0000004000111e1ac360002a5d5c51b
20	N	128-bit	0800000000111e1ac360002a5d5c51b
6	N	128-bit	4000000000111e1ac360002a5d5c51b

NOTE: two additional bytes for a timestamp for each char

life.augmented

Example: Battery Characteristic

31	30	29	28	27	26	25	24	0x00
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	
23	22	21	20	19	18	17	16	0x02
Acc	Gyro	Mag	Pressure	Humidity	Temperat	**Battery**	2nd Temp	
15	14	13	12	11	10	9	8	0x00
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	
7	6	5	4	3	2	1	0	0x00
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	

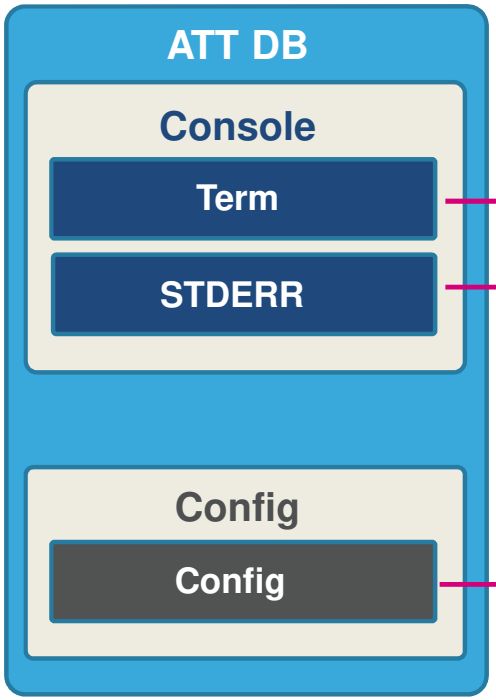


UUID VALUE
00020000000011e1ac360002a5d5c51b

The **UUID values** are linked to the Feature Mask in advertising

life.augmented

BLE_SensorDemo ATT DB



Each characteristic is defined by the following key parameters

Len	PROPERTIES	UUID TYPE	UUID VALUE
20	N,R	128-bit	00000001000E11e1ac360002a5d5c51b
20	N,R	128-bit	0000000200**0E**11e1ac360002a5d5c51b

This byte differentiates the **Service**

Len	PROPERTIES	UUID TYPE	UUID VALUE
20	N, Ww/oR, R,W	128-bit	0000000200**0F**11e1ac360002a5d5c51b

life.augmented

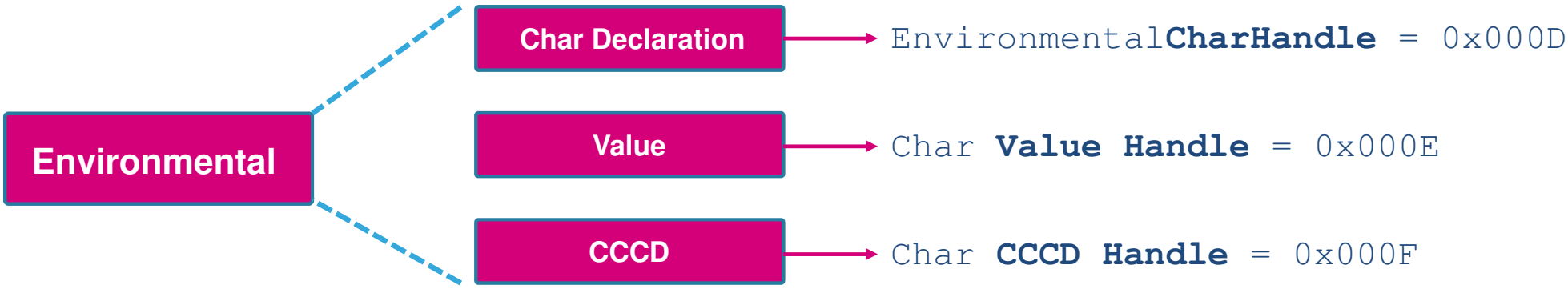
BLE_SensorDemo ATT DB


```
tBleStatus Add_HWServW2ST_Service(void)
aci_gatt_add_service(UUID_TYPE,UUID, …,MAX_NB_ATTRIBUTES,&ServHandle)
aci_gatt_add_char(ServHandle,UUID_TYPE,UUID,Length,PROPERTIES,PERMISSIONS,…,&CharHandle)
```


```
HWServW2STHandle = 0x000C
EnvironmentalCharHandle = 0x000D
AccGyroMagCharHandle = 0x0010
ConfigW2STHandle = 0x002B
ConfigCharHandle = 0x002C
```

life.augmented

BLE_SensorDemo ATT DB



- **Declaration Handle**: used by the application to access the Characteristic
- **Characteristic Value Handle**: used by the Client for Read/Write operations
- **Client Characteristic Configurator Descriptor** (**CCCD**): a GATT descriptor is added by default by the stack, if **char** has **Notify/Indicate property**. Used by Client to enable notifications/indications on char value.

life.augmented

Central: Services and Characteristics discovery procedure

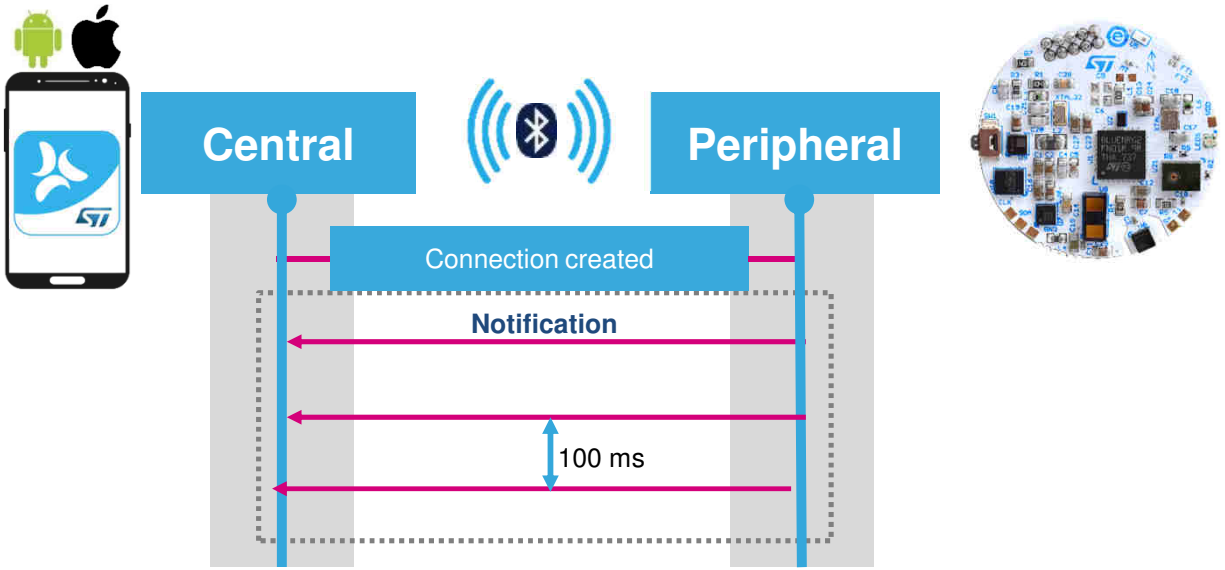
UUIDs **Handles** **Values**

Name	Handle	Property	Value	Value Length
Service User Defined (0x0000000000111E19AB40002A5D5C51B)	0x000C			
Characteristic User Defined (0x001D0000000111E1AC360002A5D5C51B)	0x000D	Read, Notify	0x011F0111023C00018B9CA605	0x0C
Client Characteristic Configuration (0x2902)	0x000F		0x0000	0x02
Characteristic User Defined (0x00E00000000111E1AC360002A5D5C51B)	0x0010	Notify		
Client Characteristic Configuration (0x2902)	0x0012		0x0000	0x02
Characteristic User Defined (0x0000400000111E1AC360002A5D5C51B)	0x0013	Read, Notify	0x0000002405	0x05
Client Characteristic Configuration (0x2902)	0x0015		0x0000	0x02
Characteristic User Defined (0x02000000000111E1AC360002A5D5C51B)	0x0016	Read, Notify	0x88A1232E	0x04
Client Characteristic Configuration (0x2902)	0x0018		0x0000	0x02
Characteristic User Defined (0x20000000000111E1AC360002A5D5C51B)	0x0019	Read, Notify	0x0024E6	0x03
Client Characteristic Configuration (0x2902)	0x001B		0x0000	0x02
Characteristic User Defined (0x00020000000111E1AC360002A5D5C51B)	0x001C	Read, Notify	0x0180000CBA03E8AB63	0x09
Client Characteristic Configuration (0x2902)	0x001E		0x0000	0x02
Characteristic User Defined (0x00000100000111E1AC360002A5D5C51B)	0x001F	Notify		
Client Characteristic Configuration (0x2902)	0x0021		0x0000	0x02
Characteristic User Defined (0x00000040000111E1AC360002A5D5C51B)	0x0022	Notify		
Client Characteristic Configuration (0x2902)	0x0024		0x0000	0x02
Characteristic User Defined (0x08000000000111E1AC360002A5D5C51B)	0x0025	Notify		
Client Characteristic Configuration (0x2902)	0x0027		0x0000	0x02
Characteristic User Defined (0x40000000000111E1AC360002A5D5C51B)	0x0028	Notify		
Client Characteristic Configuration (0x2902)	0x002A		0x0000	0x02
Service User Defined (0x00000000000F11E19AB40002A5D5C51B)	0x002B			
Characteristic User Defined (0x00000002000F11E1AC360002A5D5C51B)	0x002C	Write w/o resp, Notify		
Client Characteristic Configuration (0x2902)	0x002E		0x0000	0x02
Service User Defined (0x00000000000E11E19AB40002A5D5C51B)	0x002F			
Characteristic User Defined (0x00000001000E11E1AC360002A5D5C51B)	0x0030	Read, Write w/o resp, Write, Notify	0x0	0x00
Client Characteristic Configuration (0x2902)	0x0032		0x0000	0x02
Characteristic User Defined (0x00000002000E11E1AC360002A5D5C51B)	0x0033	Read, Notify	0x0	0x00
Client Characteristic Configuration (0x2902)	0x0035		0x0000	0x02

This is how ATT DB looks like from the Central

life.augmented

BLE Sensor – Characteristic update

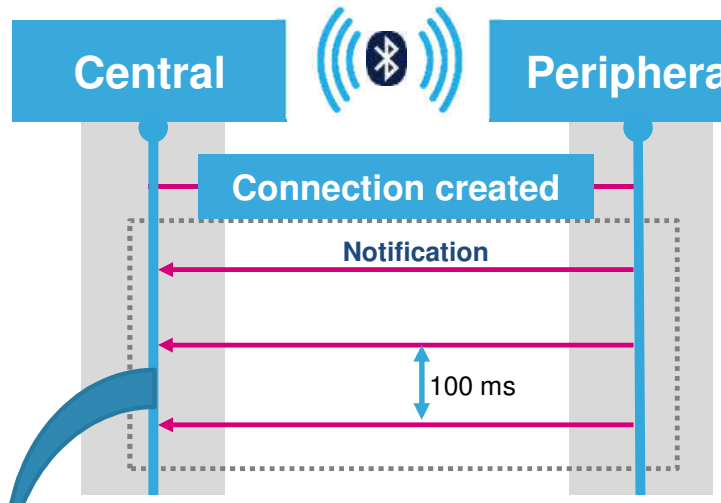


aci_gatt_update_char_value(ServiceHandle, 0x000C, //HWserv handle
CharHandle, 0x000D, //EnvChar handle
Offset, 0x00,
Length, 0x08,
Value) Data) //EnvChar Value

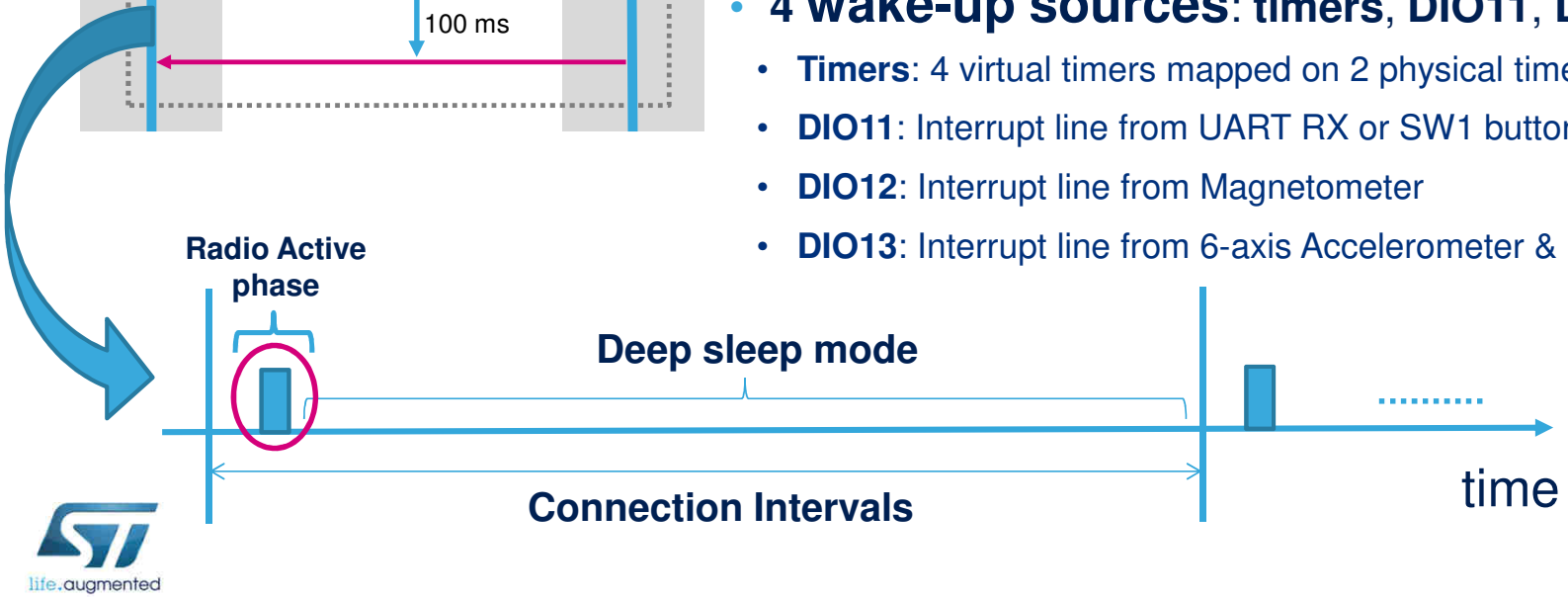
Description	Where
If notifications (or indications) are enabled on the characteristic, this API sends a notification (or indication) to the client.	GATT server

life.augmented

Cortex-M0 Sleep management

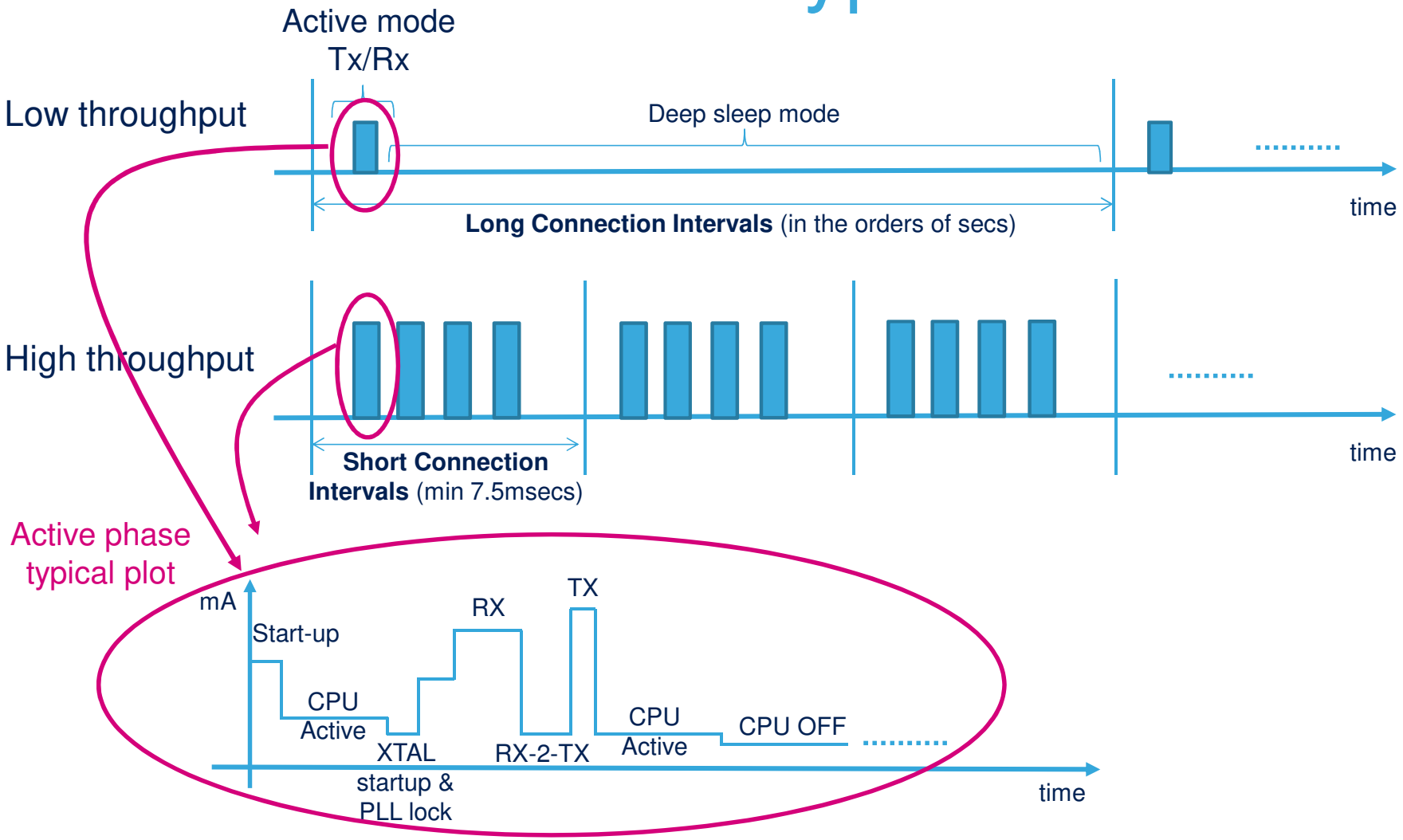


- **Sleep mode** managed through specific a **library** and **API**
  - *BlueNRG_Sleep(sleepMode, gpioWakeBitMask, gpioWakeLevelMask)*
- **4 wake-up sources**: **timers**, **DIO11**, **DIO12** and **DIO13**
  - **Timers**: 4 virtual timers mapped on 2 physical timers managed by the radio IP
  - **DIO11**: Interrupt line from UART RX or SW1 button
  - **DIO12**: Interrupt line from Magnetometer
  - **DIO13**: Interrupt line from 6-axis Accelerometer & Gyroscope



life.augmented

Typical scenarios



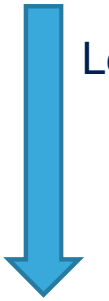
Low Power Modes

- **Deep sleep mode** can represent most of the application time.
- Efficient management of sleep mode significantly lowers the avg power consumption.

BlueNRG-2 flexible low power architecture

Sleeping Mode	Consumption	Notes
RUNNING	1.9mA	Core running
CPU HALT	1.5mA	WFI instruction
WAKETIMER	900nA	GPIOs and Timer Wakeup
NOTIMER	**500nA**	GPIOs only Wakeup

Highest pwr



Low Power library combines requests coming from the application with the radio operating mode

Lowest pwr

life.augmented

Context Save/Restore

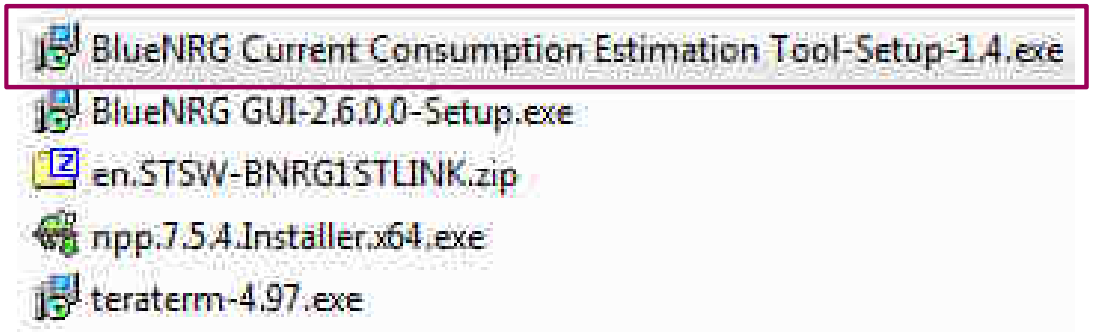
- BlueNRG-2 Low Power Library **handles autonomously entering and exiting to/from the sleep mode**.

- **The library saves peripherals configurations and application context** before entering deep sleep, and **restores the context** on exiting from low power state:
  - **CPU returns to execute the next instruction after the low power function call**
  - **No need to re-initialize peripheral and radio stack**
  - **RAM retention is guaranteed**

NO need for the application to worry about sleep management and RAM retention!

BlueNRG Current Consumption Estimation Tool

- Check on the installation folder in the path "**BlueNRG_Tile_HandsOn/Software**"



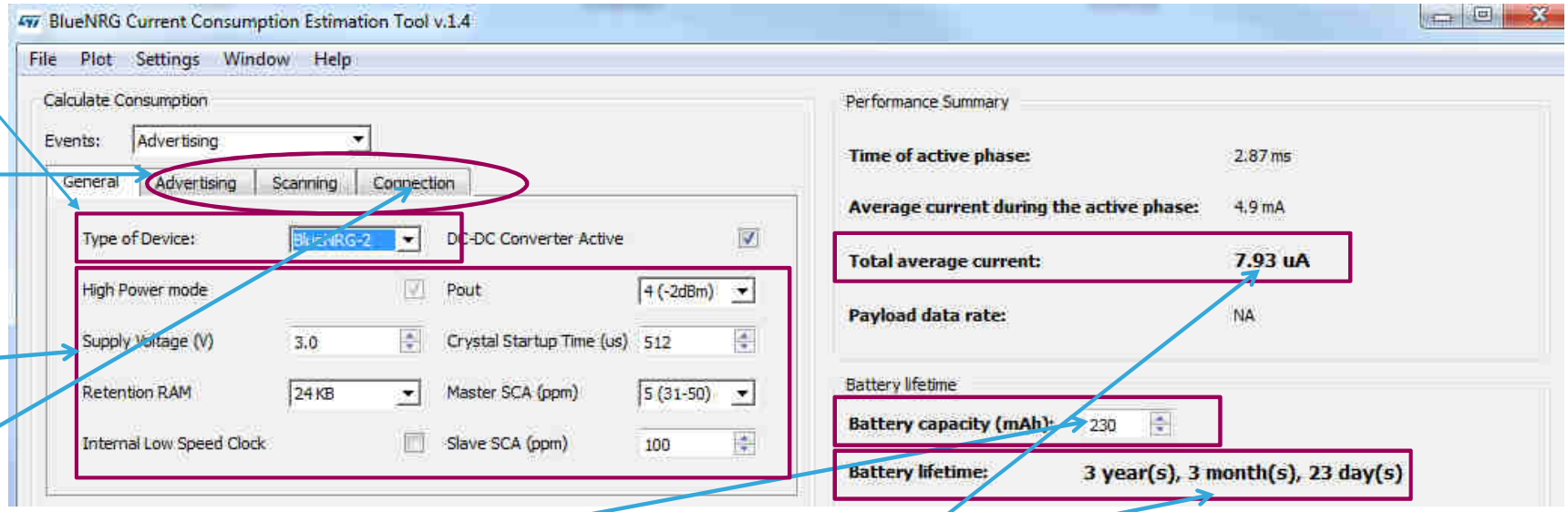
- Or on st.com at this link:

http://www.st.com/content/st_com/en/products/embedded-software/wireless-connectivity-software/stsw-bnrg001.html

BlueNRG Current Consumption Estimation Tool

- **Accurate estimate** of average **current consumption** and **battery lifetime**

1. **Select** the device
2. **Select** the type of event
3. **Configure** the HW operating conditions
4. **Configure** the application use case scenario



5. **Specify** the Battery capacity

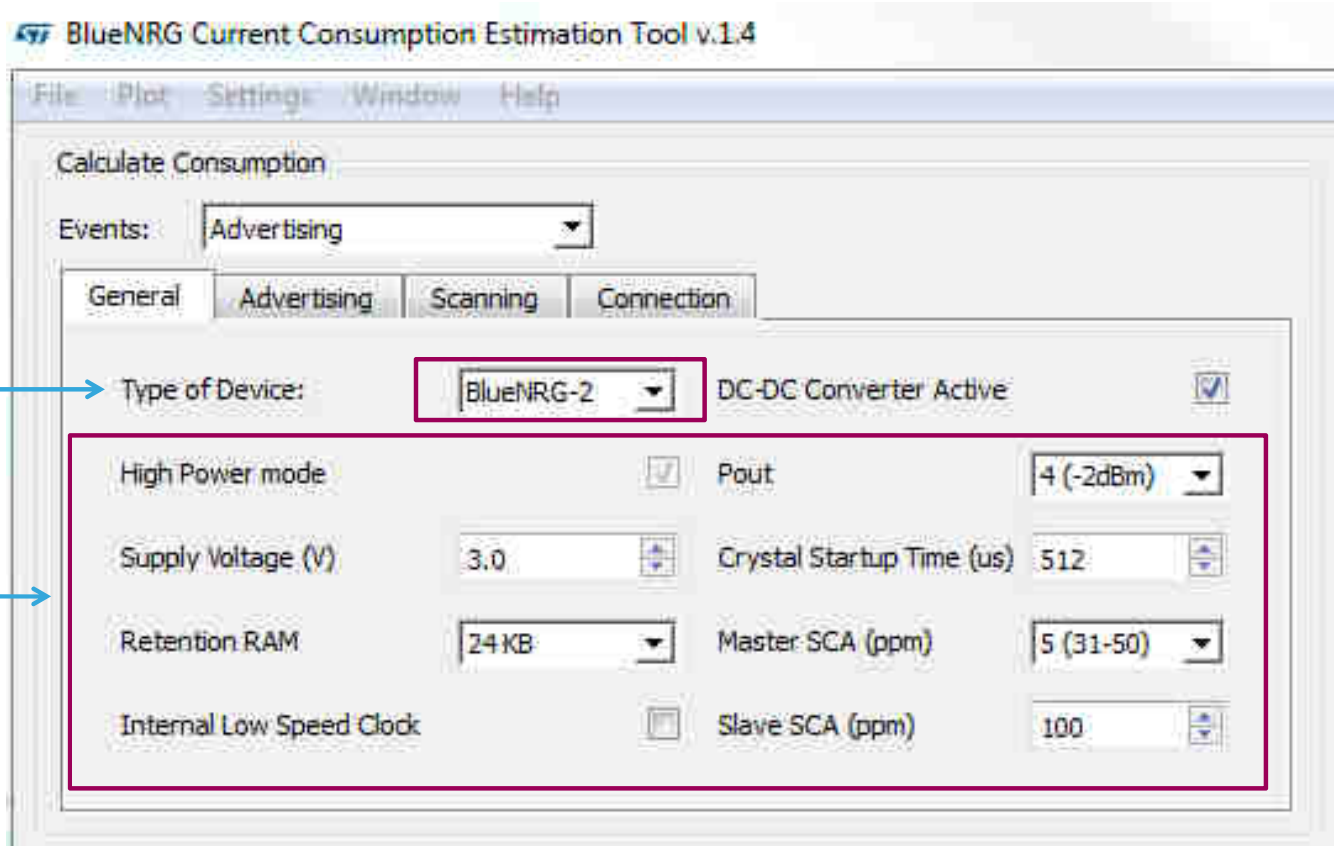
6. **Read** the **average current** and the estimated **battery lifetime**

life.augmented

Advertising example 1/3

1. **Select** the device

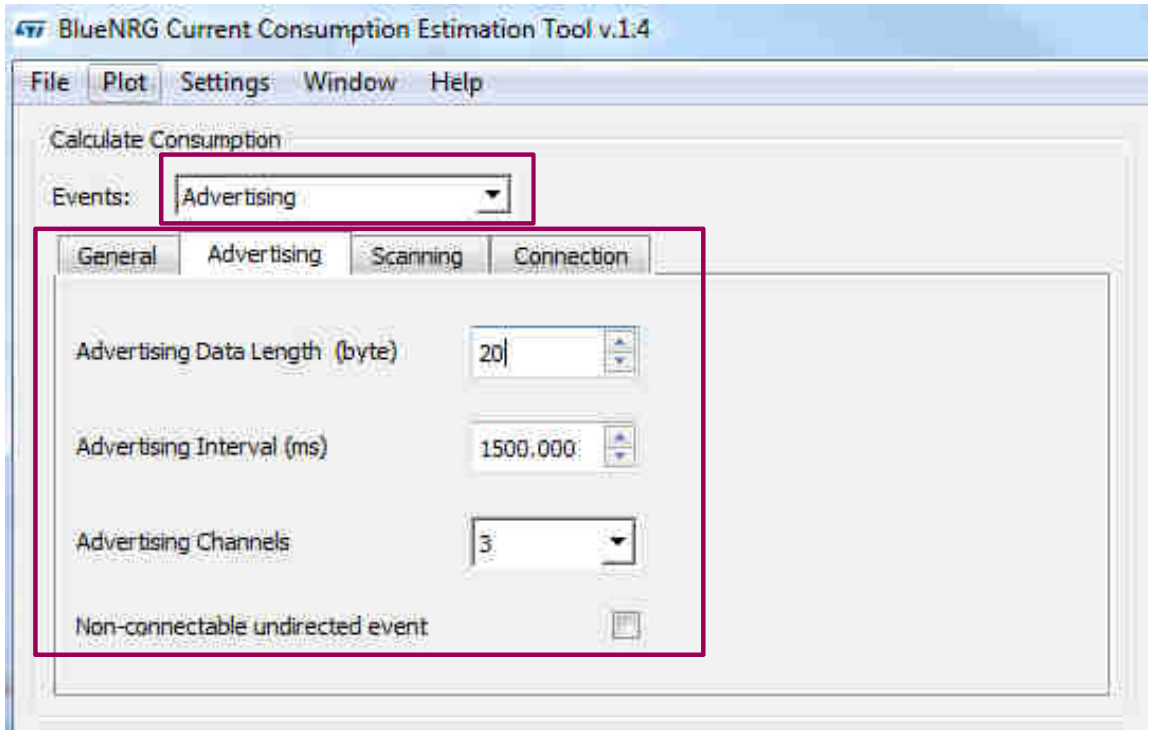
2. **Configure** the HW operating conditions



Advertising example 2/3

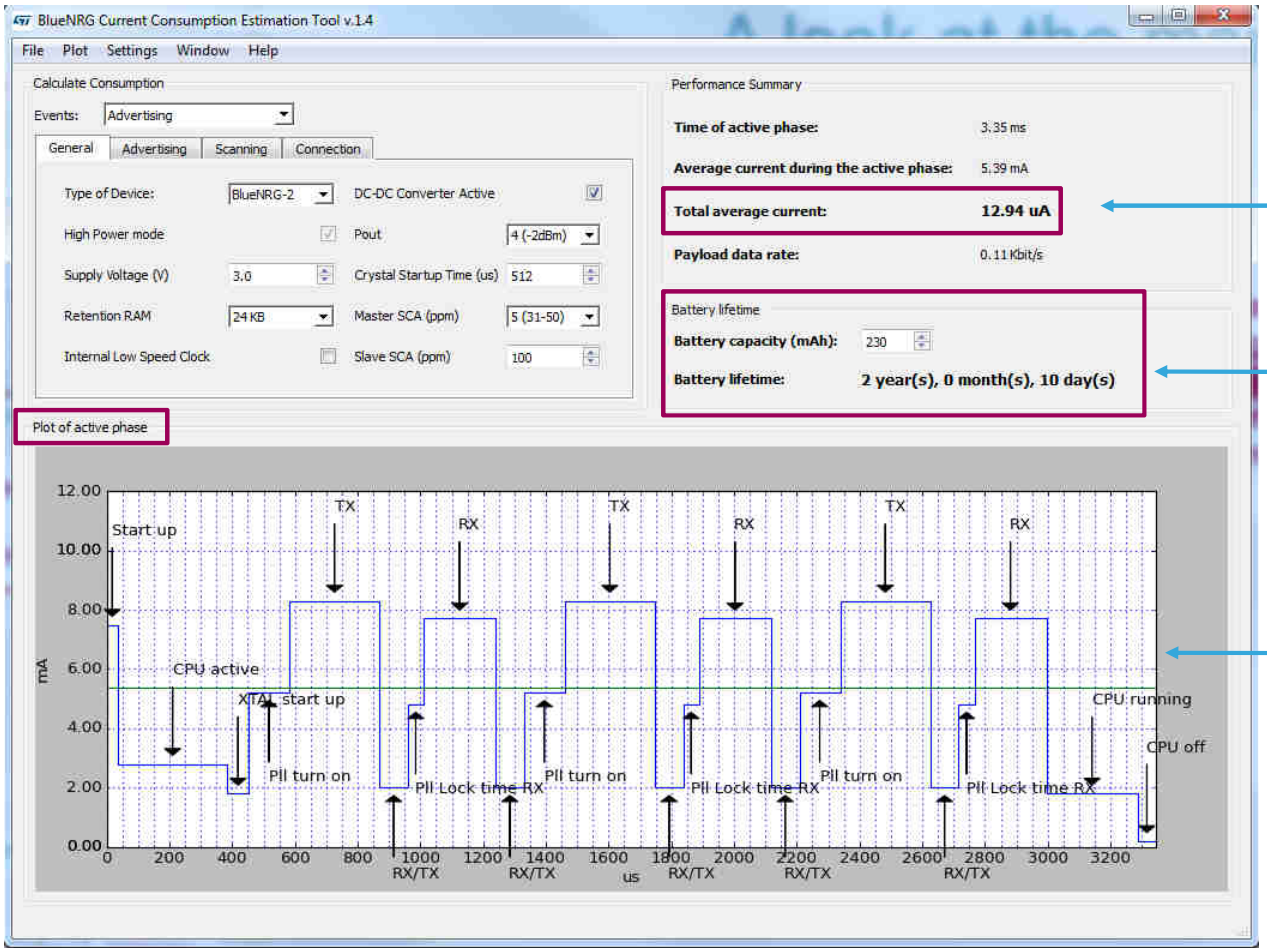
1. **Select** the event type

2. **Configure** the event operating conditions



life.augmented

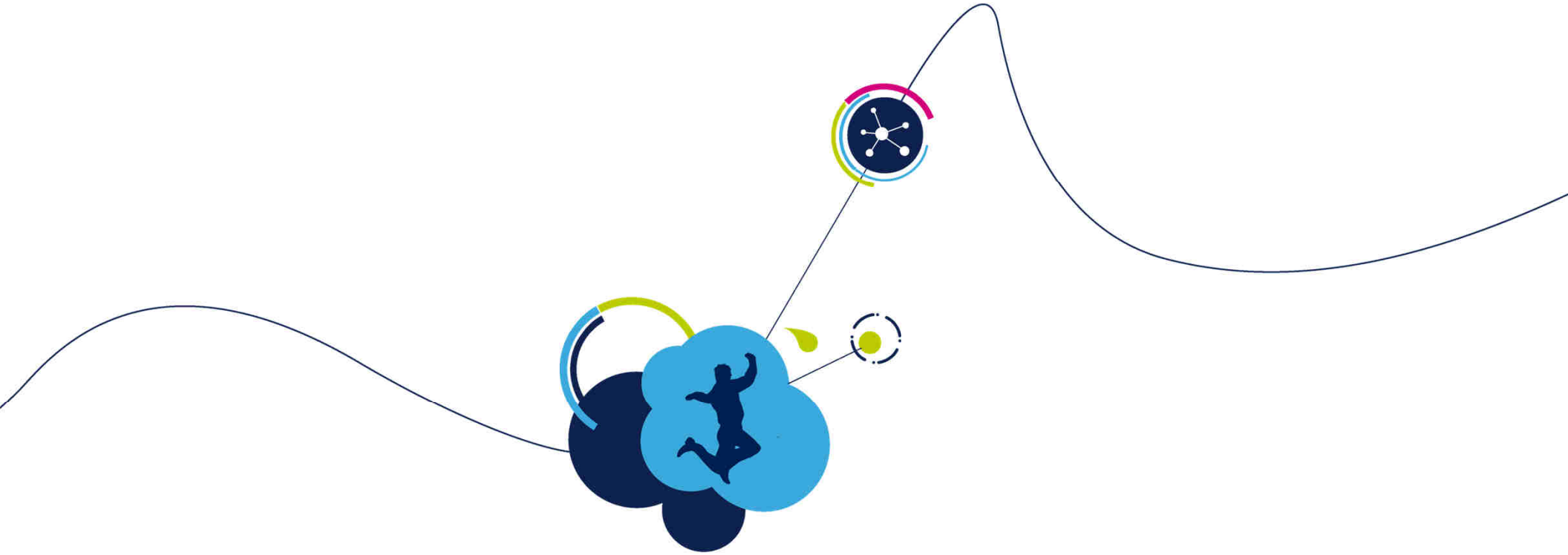
Advertising example 3/3



1. **Get** average current consumption

2. **Get** expected battery lifetime for a **CR2032**

3. **Plot** of the active phase

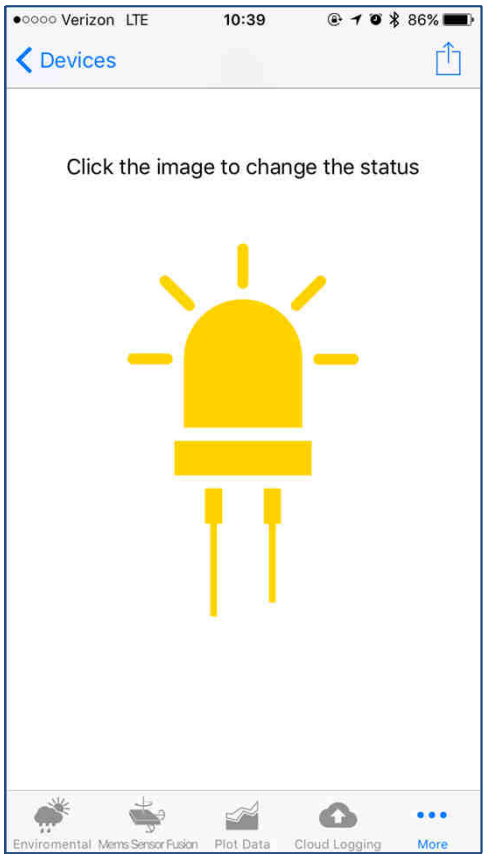


Lab 3

LED characteristic

life.augmented

Enable LED toggling

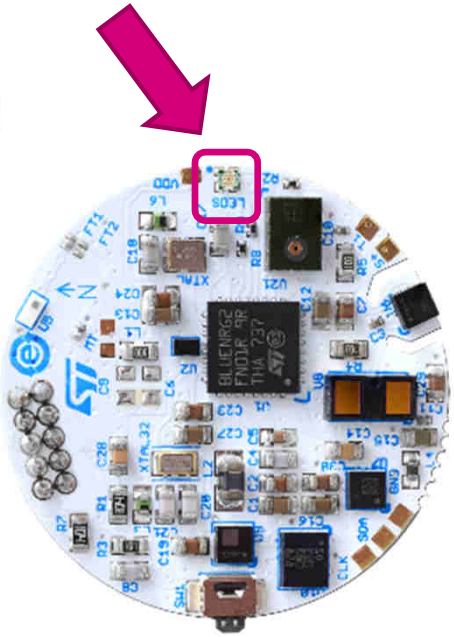


Write Packet

'1' LED ON
'0' LED OFF

Notification Packet

'1' LED ON
'0' LED OFF



1. **Tap** the image on the screen
2. Send **BLE notification packet**
3. Image changes color and the **RED LED toggles**

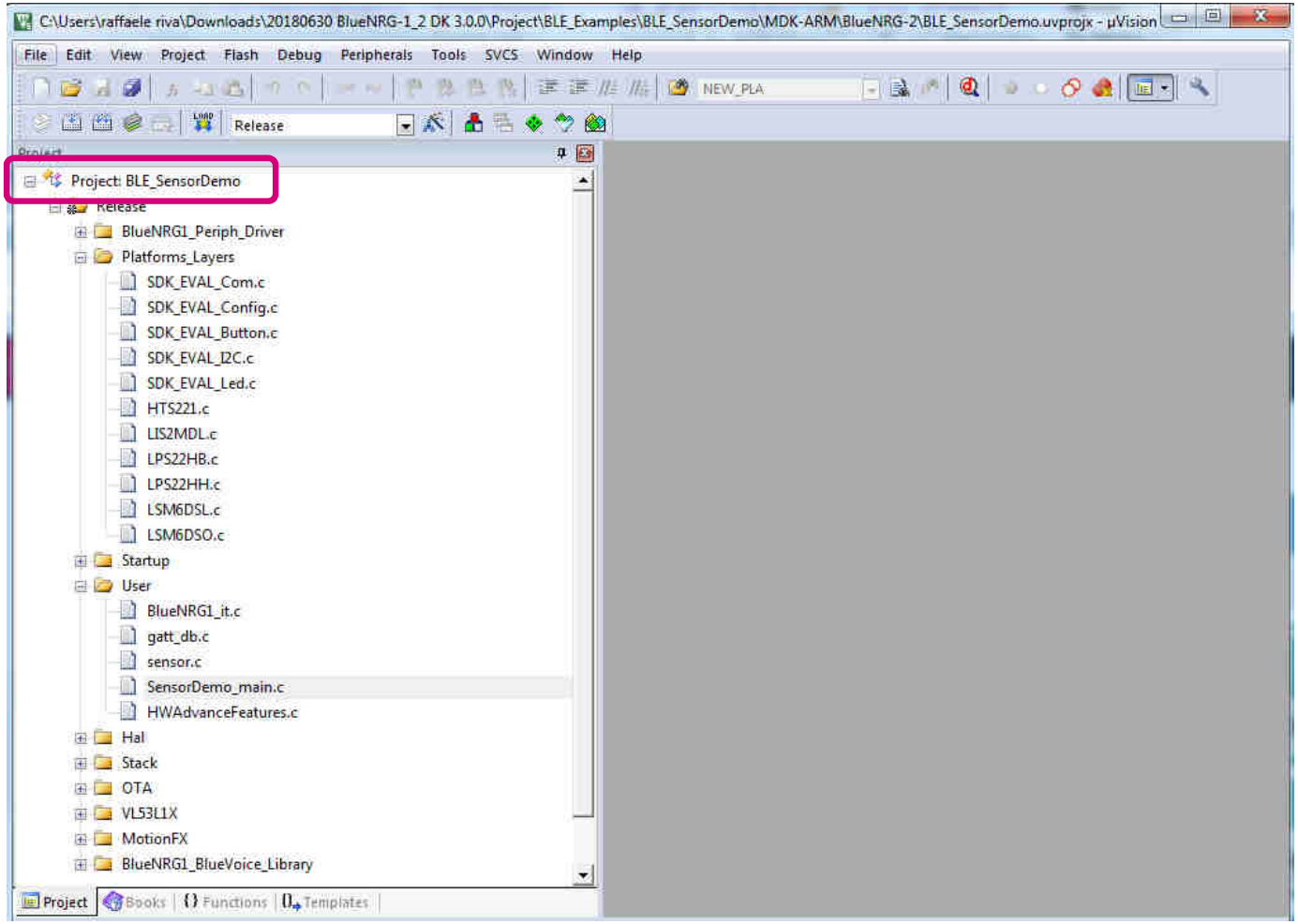
life.augmented

Code flow

1. **Modify advertising packet**
2. Client **enables LED characteristic notifications**
3. Client **Writes in Config Char**
4. Parse **write command** from the client and **send BLE notification** on LED status from the server

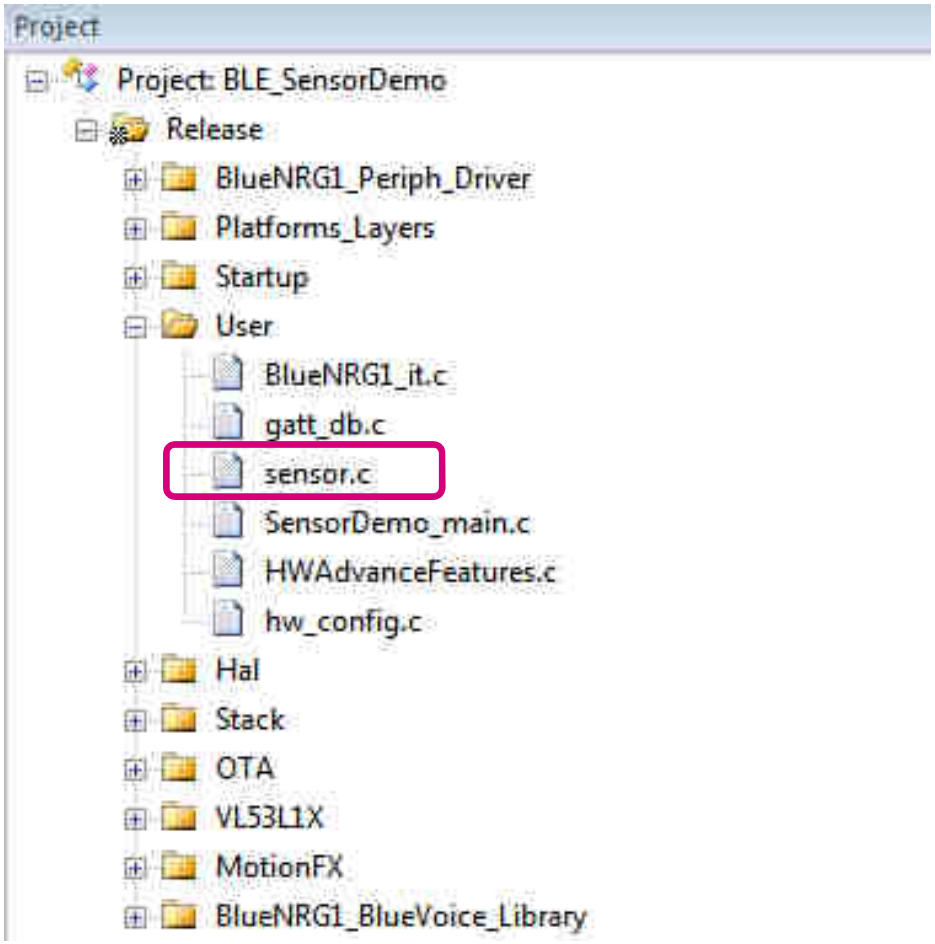
Sensor Demo reference application

Now we **go back** again to **Keil uVision**



life.augmented

L3 STEP1: Modify advertising packet



ST BlueMS Protocol

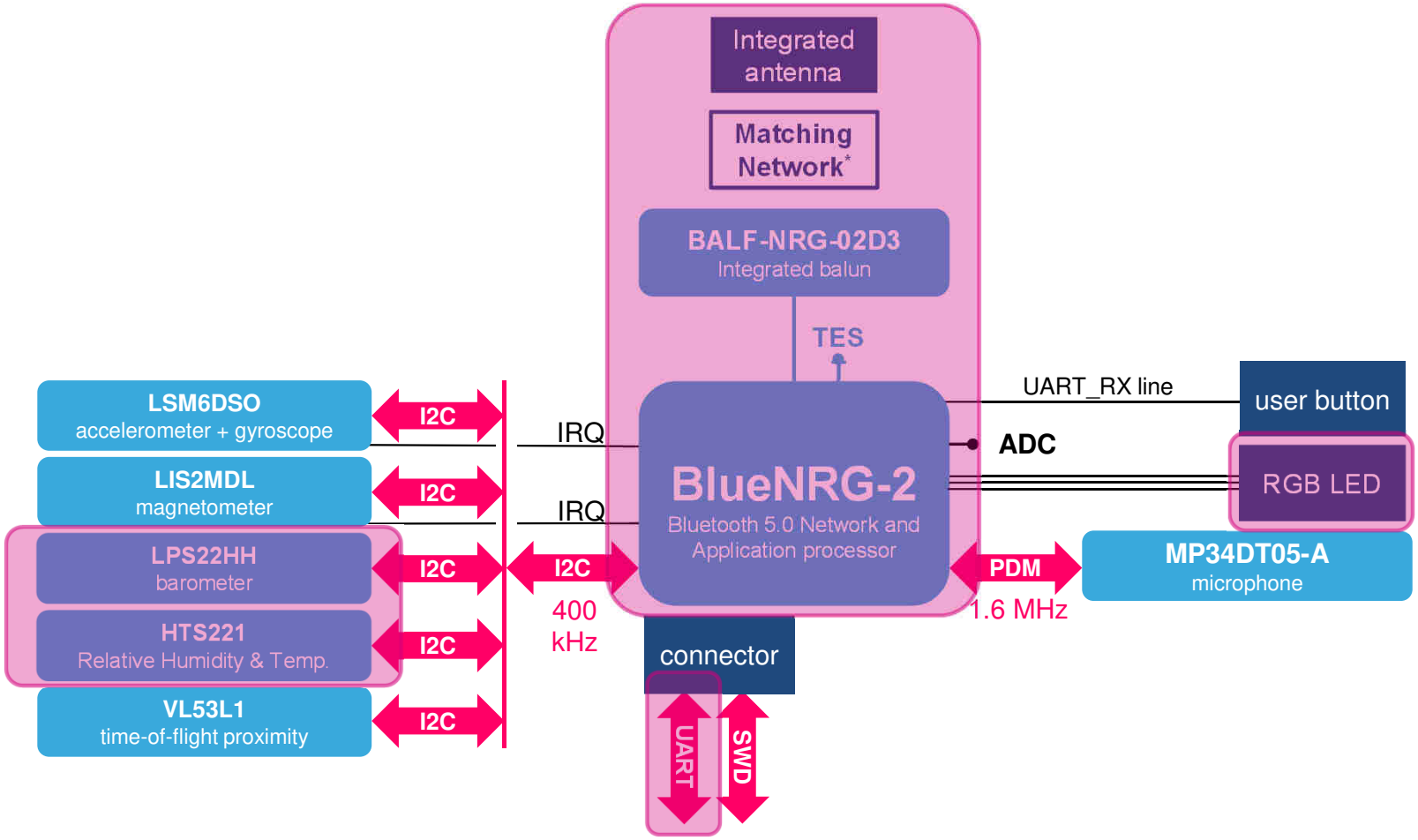
BlueMS

4 Bytes

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	0x20
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0x1E
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	0x00
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x00

life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

L3 STEP1: Modify advertising packet

Modify the Feature Mask in the advertisement payload

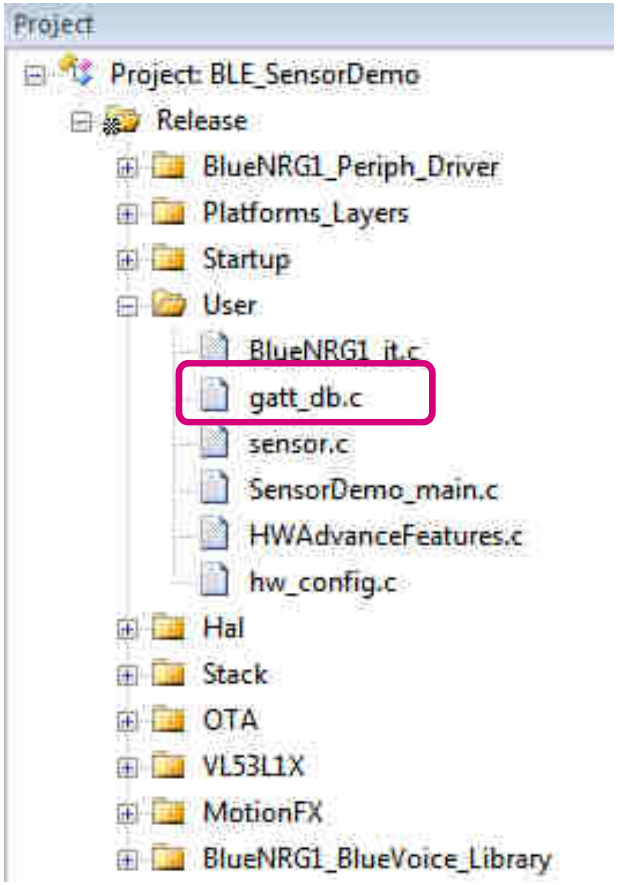
1. In the file **sensor.c**
2. **Go to line 423**
3. **Modify** Feature Mask **byte#1** from **0x00** to **0x20**

sensor.c

```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                      /* Length of AD type Transmission Power */
415   0x0A, 0x00,             /* Transmission Power = 0 dBm */
416   8,                      /* Length of AD type Complete Local Name */
417   0x09,                   /* AD type Complete Local Name */
418   NAME_ALLMEMS,           /* Local Name */
419   13,                     /* Length of AD type Manufacturer info */
420   0xFF,                   /* AD type Manufacturer info */
421   0x01,                   /* Protocol version */
422   0x05,                   /* Device ID: 0x05 = STEVAL-BCN002V1 Board */
423   0x20,                   /* Feature Mask byte#1: LAB3 0x20 (LED)
424   0x1E,                   /* Feature Mask byte#2: LAB4 0x9E (Acc+Press+Hum
425   0x00,                   /* Feature Mask byte#3: LAB4 0x04 (AccEvents) /
426   0x00,                   /* Feature Mask byte#4: LAB5 0x40 (eCompass) */
427   0x00,                   /* BLE MAC start */
```

life.augmented

L3 STEP2: Client enables LED characteristic notifications



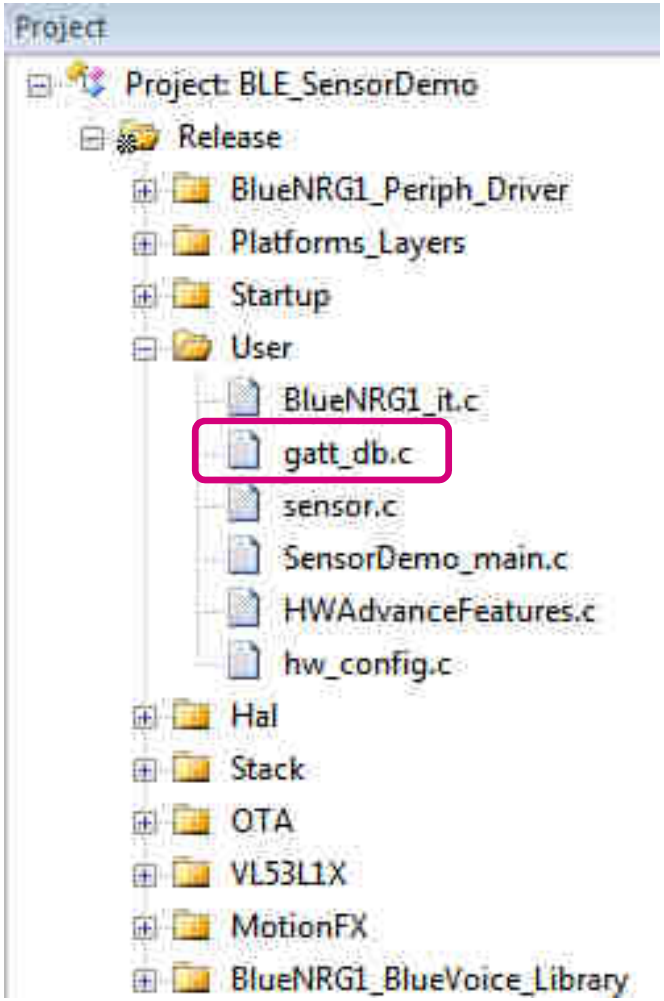
L3 STEP2: Client enables LED characteristic notifications

1. In the file **gatt_db.c** **go to** the **line 686**
2. **Client writes** in the **LED Characteristics Client Configurator Descriptor (CCCD)** and **Server enables notifications** through the ***xFeatureNotification*** **structure** in **line 689** by ***xFeatureNotification.LedNotification = true;***

```
686   } else if (attr_handle == (LedCharHandle + 2)) {
687
688     if (att_data[0] == 01) {
689       xFeatureNotification.LedNotification = true;
690       PRINTF("Led Notification ON\n\r");
691     } else if (att_data[0] == 0) {
692       xFeatureNotification.LedNotification = false;
693       PRINTF("Led Notification OFF\n\r");
694     }
```

life.augmented

L3 STEP3: Client Writes in Config Char

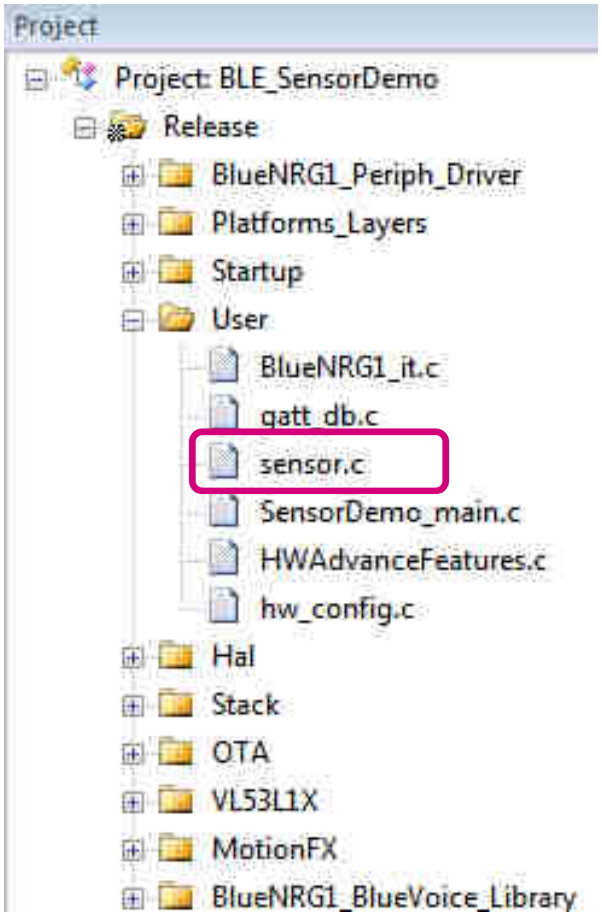


L3 STEP3: Client Writes in Config Char

1. In the file **gatt_db.c** **go to** the **line 816**
2. Client **writes** in the **Config Char** and **Server parses** the command through the function ***ConfigCommandParsing(att_data, data_length)***

```
816   } else if (attr_handle == ConfigCharHandle + 1) {
817
818     /* Received one write command from Client on Configuration characteristic */
819     ConfigCommandParsing(att_data, data_length);
820
821   }
```

L3 STEP4: Parse write command and send BLE notification on LED status



L3 STEP4: Parse write command and send BLE notification on LED status

1. Line **941**: **ConfigCommandParsing**
2. **Check** on the **FeatureMask** (line **947**)
3. If it's the LED feature mask (line **1125**)
4. **Check** on the command value. If **0x01** (case1)
5. **Turn ON LED3**
6. **Send Notification** on the Config char on the command received and parsed
7. **Send Notification on the LED char** with the **LED status**

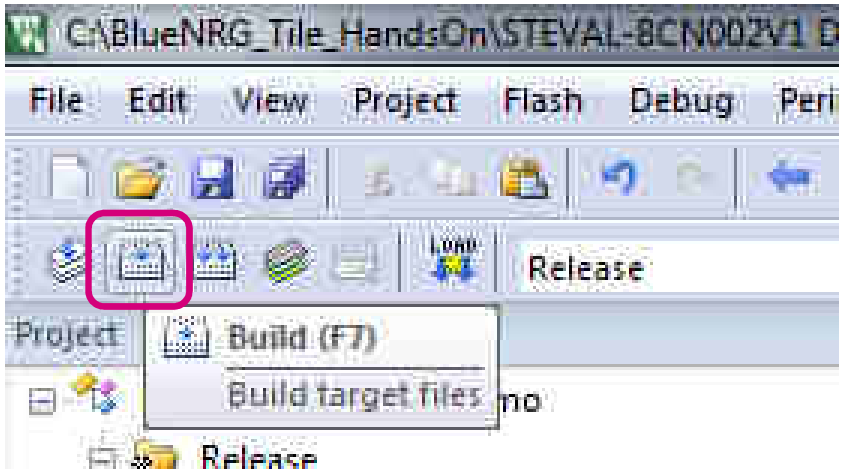
```
941  uint32_t ConfigCommandParsing(uint8_t * att_da
...  FeatureMask = att_data[1] | (att_data[2] <
943    uint8_t Command = att_data[4];
944    uint8_t Data = att_data[5];
945    uint32_t SendItBack = 1;
946
947  switch (FeatureMask) {
```

```
1125  case FEATURE_MASK_LED:
1126    switch (Command) {
1127    case 1:
1128      SdkEvalLedOn(LED1);
1129      PRINTF("Enabled: RGB led\n\r");
1130      Config_Notify(FEATURE_MASK_LED, Command, Data);
1131      break;
1132    case 0:
1133      SdkEvalLedOff(LED1);
1134      PRINTF("Disabled: RGB led\n\r");
1135      Config_Notify(FEATURE_MASK_LED, Command, Data);
1136      break;
1137    }
1138
1139  if (xFeatureNotification.LedNotification)
1140    if (SdkEvalLedGetState(LED1))
1141      Led_Update(ENABLE);
1142    else
1143      Led_Update(DISABLE);
1144  }
1145  break;
```

life.augmented

Build the new code

1. **Click** on the **Build button** (top left corner) or hit **F7** on your keyboard
2. In the **Build Output** window (bottom) **wait** for the build to be completed.
   - **BLE_SensorDemo.bin** created
   - "**0 Error(s), 0 Warning(s)**" message appear

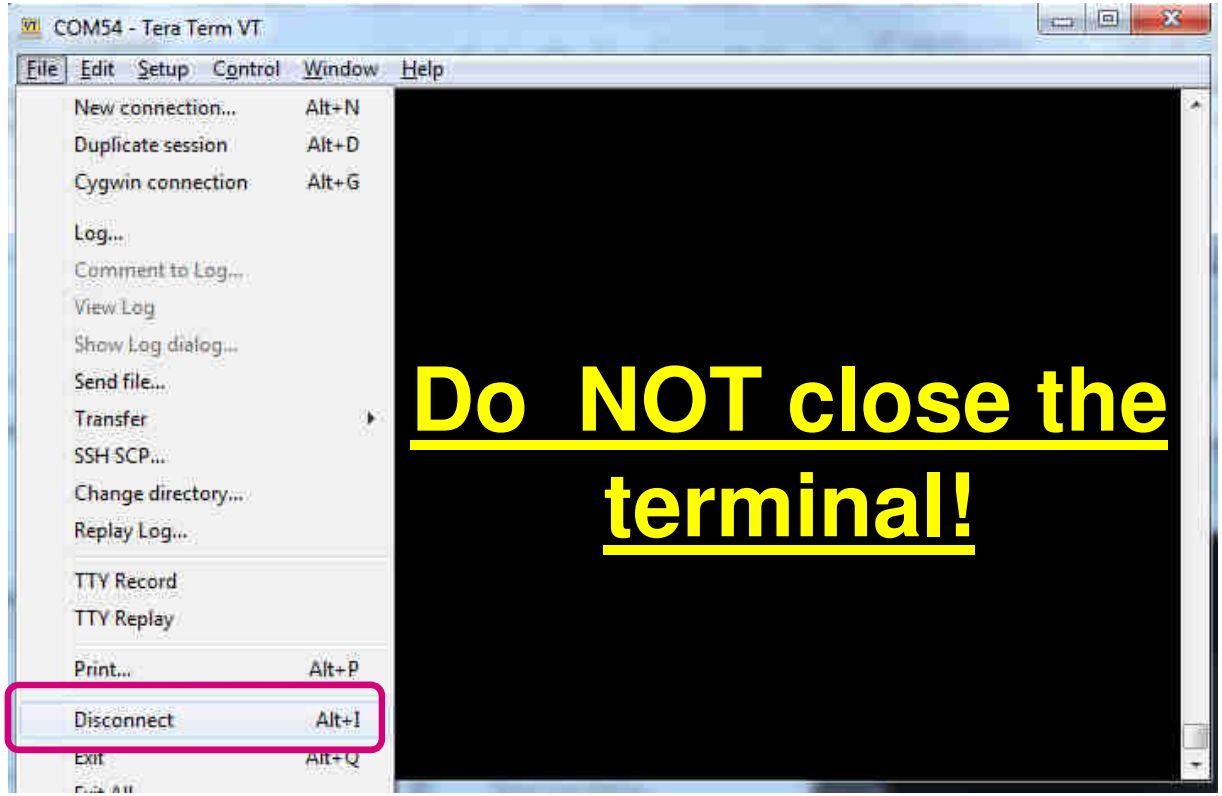



```
Build Output
compiling vl53l1_wait.c...
compiling vl53l1_register_funcs.c...
compiling vl53l1_platform.c...
linking...
Program Size: Code=121908 RO-data=1428 RW-data=1136 ZI-data=21252
FromELF: creating hex file...
After Build - User command #1: fromelf.exe --bin ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf --output ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.bin
"..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:12
```

life.augmented

Disconnect the serial terminal

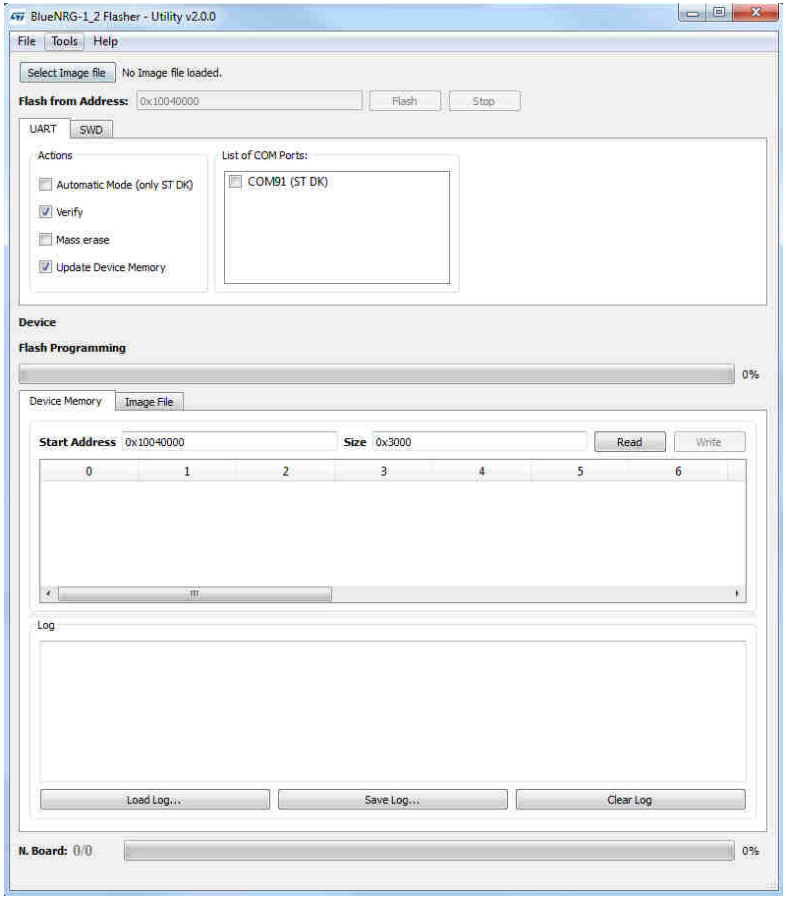
1. **Go back** to **Tera Term**
2. **Click** on the **File->Disconnect**



life.augmented

Re-flash the BlueNRG-2

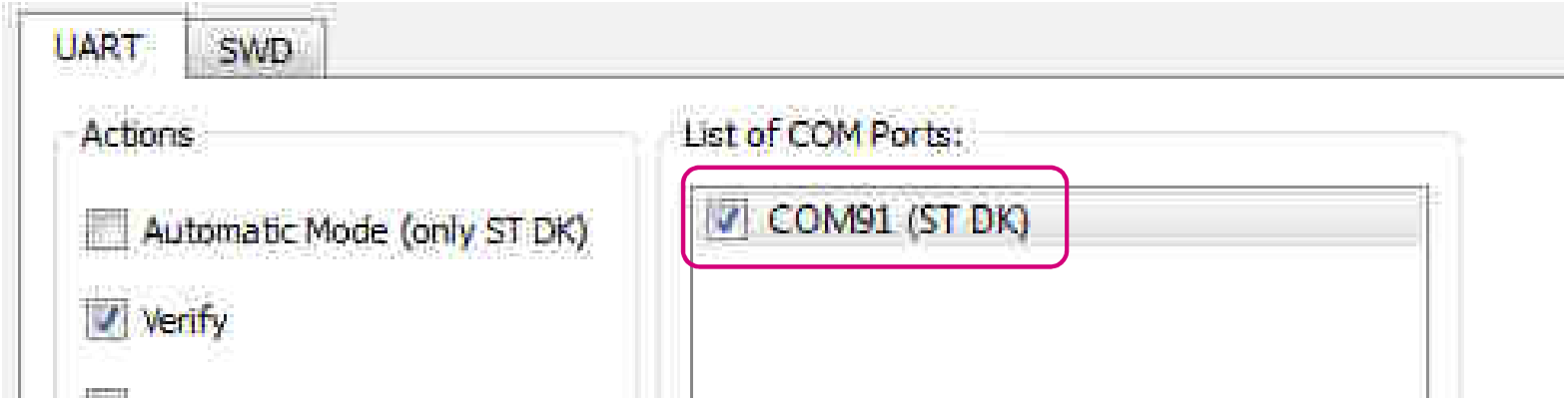
1. **Go back** to the **BlueNRG-2 Flasher Utility**



life.augmented

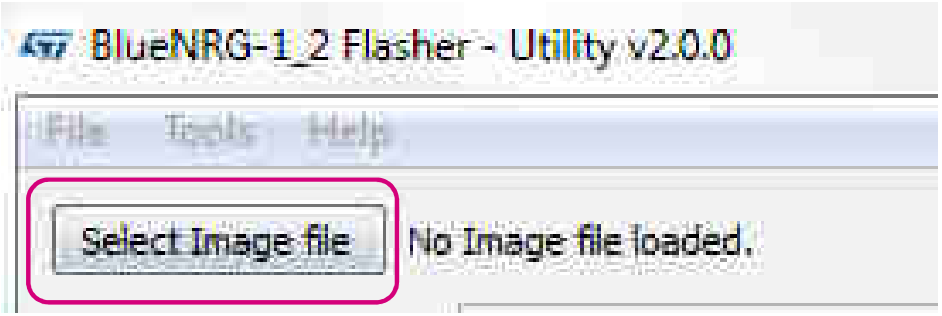
Flash the BlueNRG-2 1/4

1. **Select** the COM port labeled **(ST DK)**



Flash the BlueNRG-2 2/4

1. **Click** on the **Select Image file** button

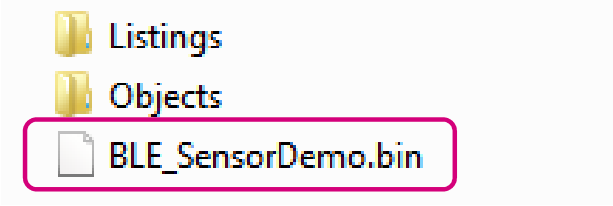


2. And **browse** the following **path**

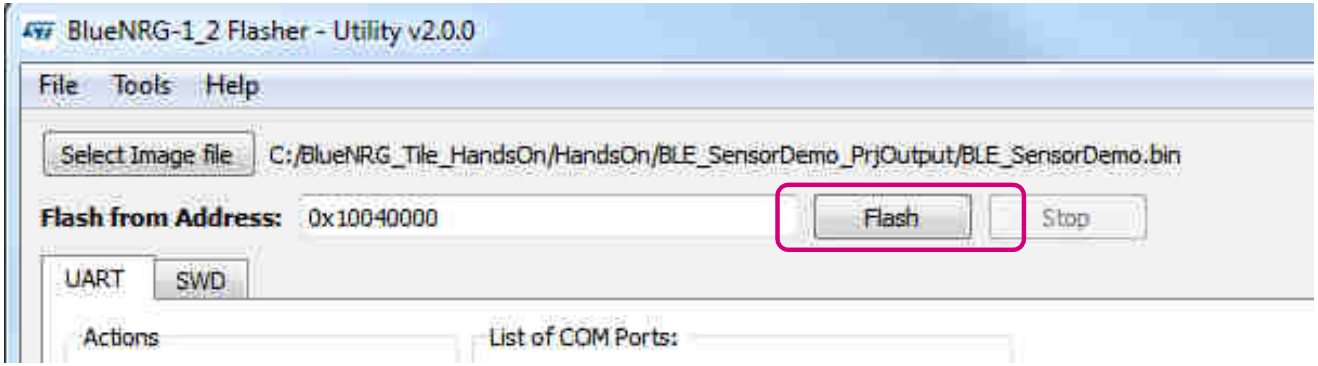
life.augmented

Flash the BlueNRG-2 3/4

1. **Select BLE_SensorDemo.bin** and **click Open**



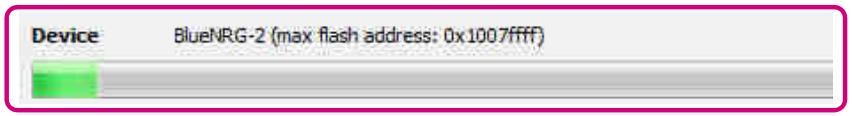
2. **Click** on the **Flash** button



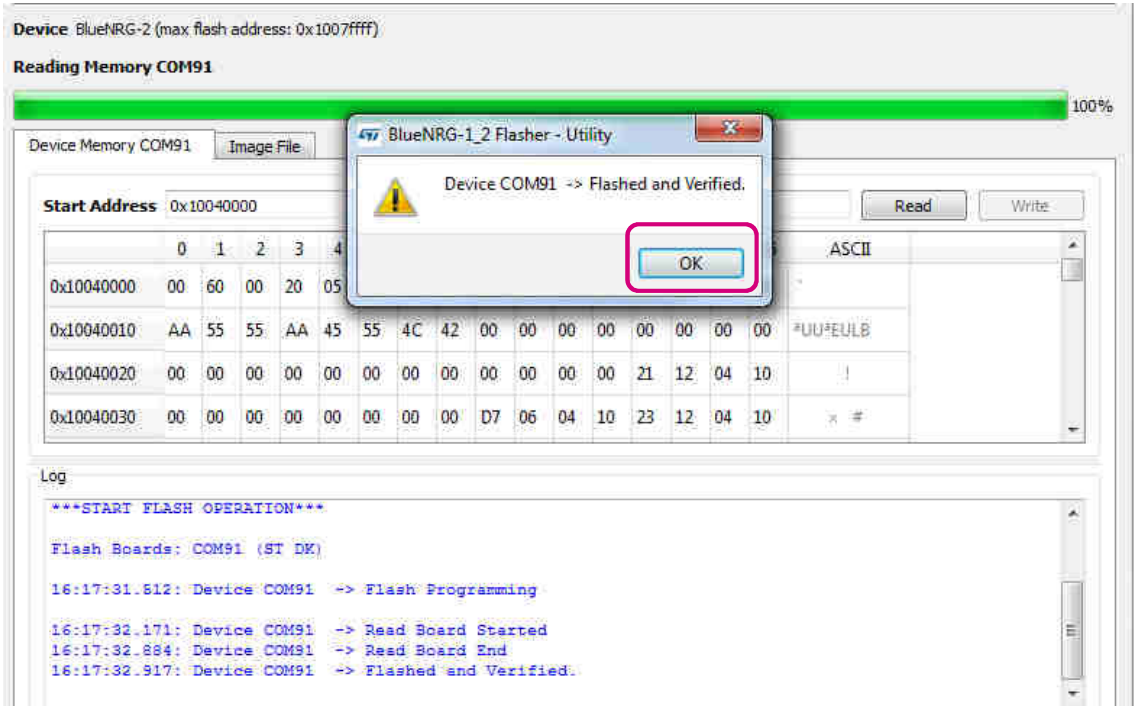
life.augmented

Flash the BlueNRG-2 4/4

1. Flashing starts: **green bar** proceeding



2. **Wait** for the **pop-up** window and **click** on **OK**

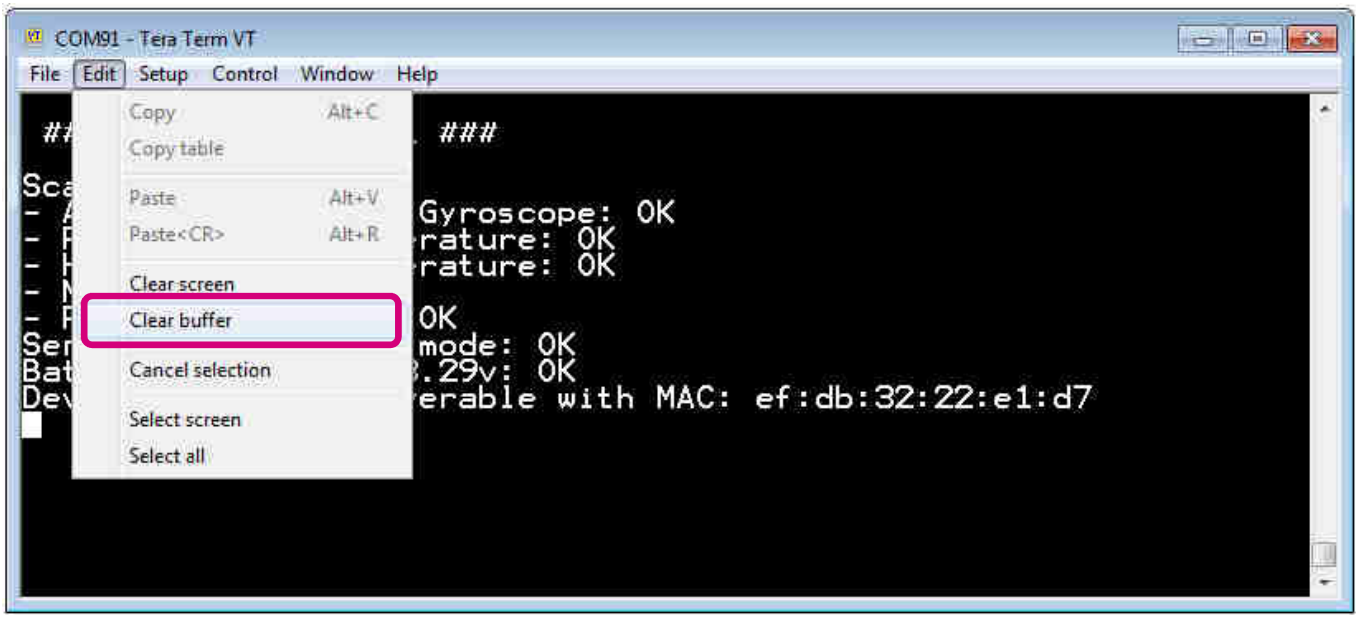


Do NOT close the Utility!

life.augmented

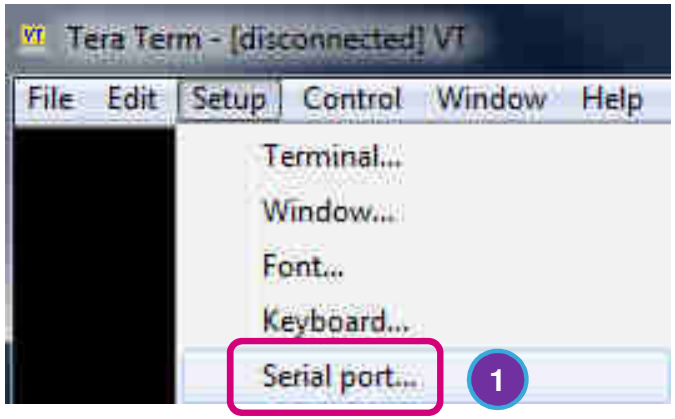
Clean Buffer in the serial terminal

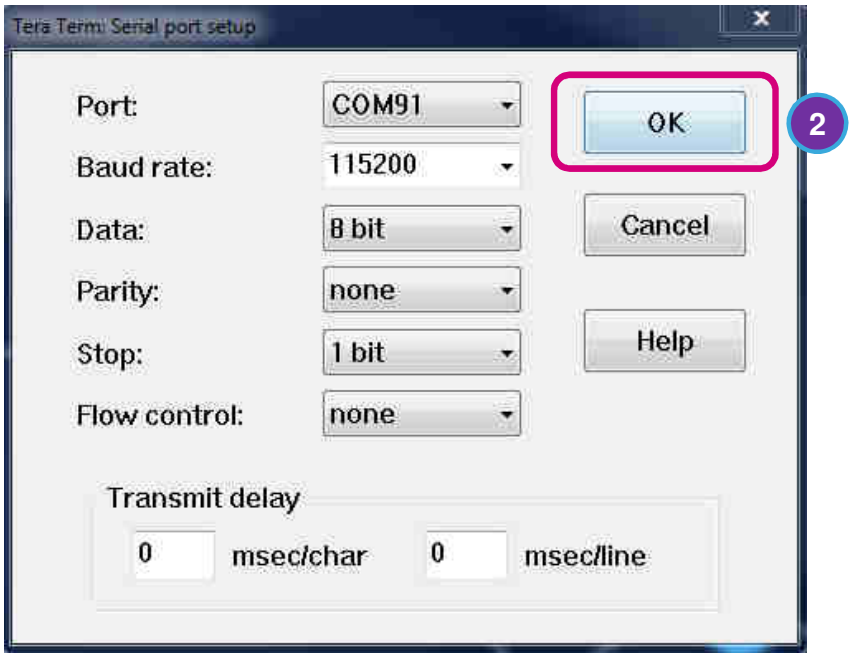
1. In **Tera Term** in order to have the terminal clean, **go to Edit -> Clear buffer**



Reconnect the serial terminal

1. **Select Setup -> Serial port…**
2. Serial port should be already configured.
   Just need to **click** on **OK**

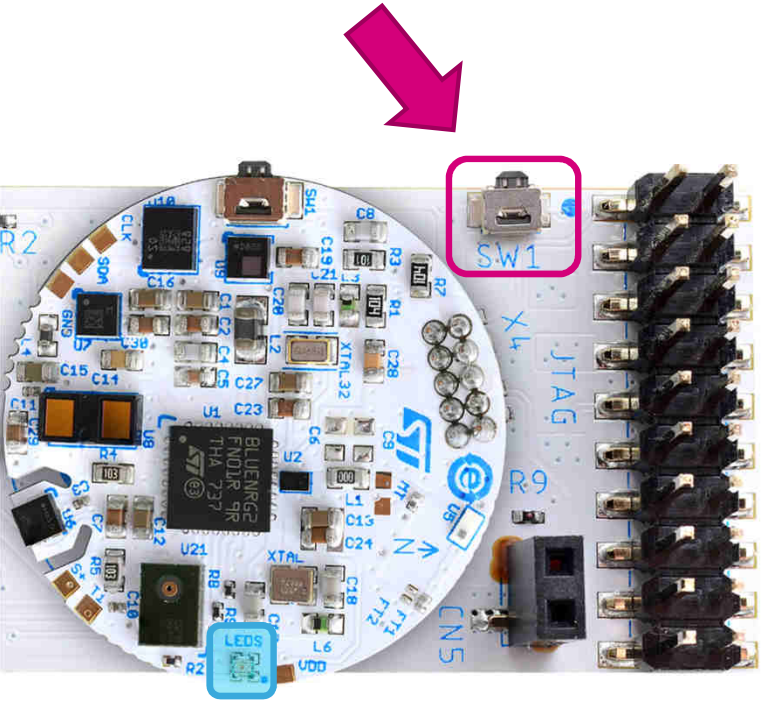


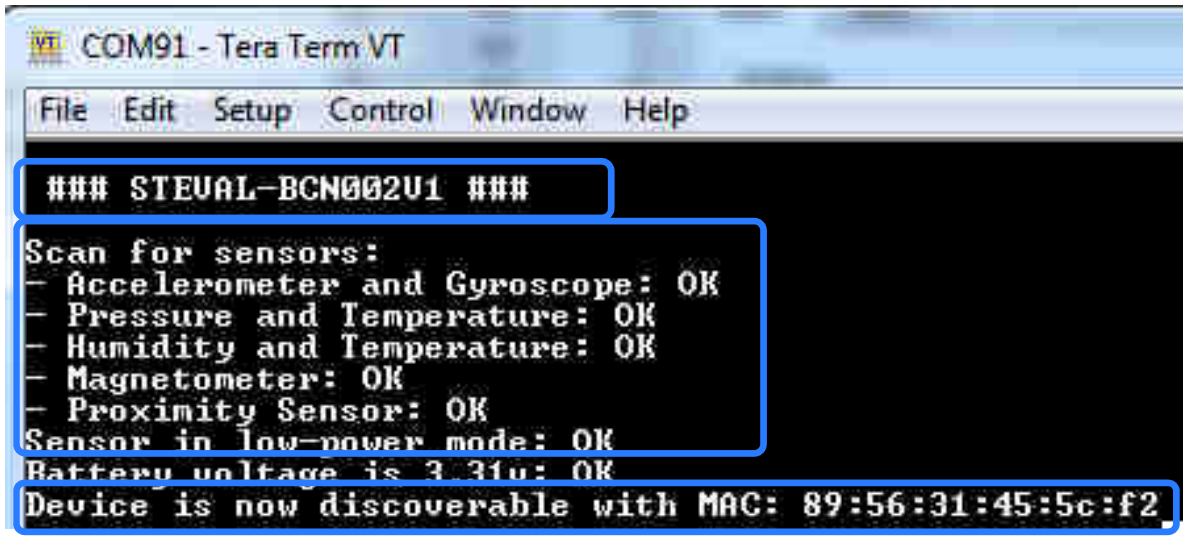


life.augmented

Sanity Check on serial port

Push SW1 button on the daughterboard -> **LED blinking Blue**

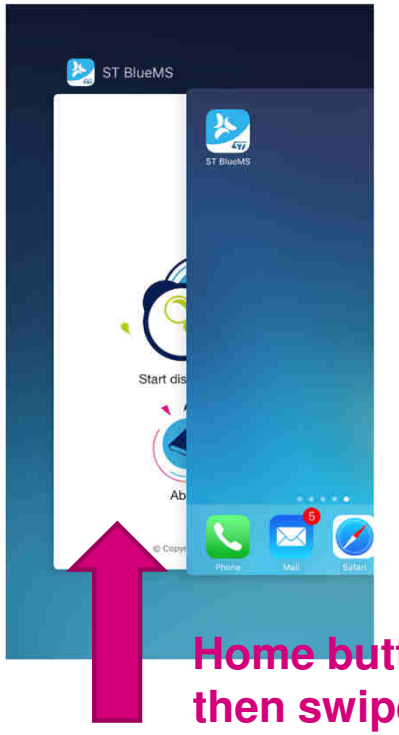




life.augmented

Connect using the ST BLE Sensor App

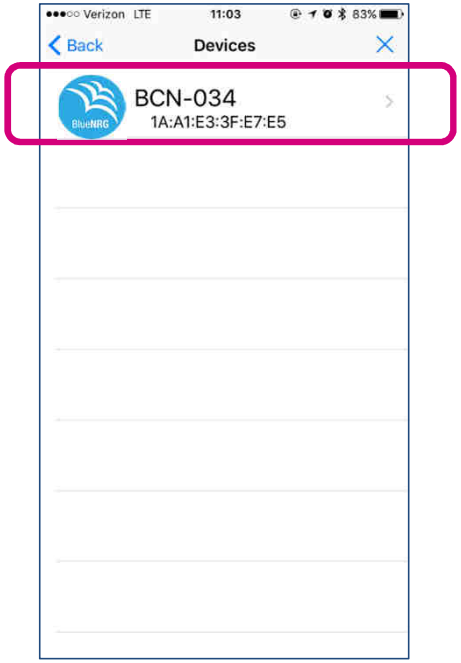
1 **"Kill" the BLE Sensor app**



Home button and then swipe to top

2 **Tap "Connect to a device"**

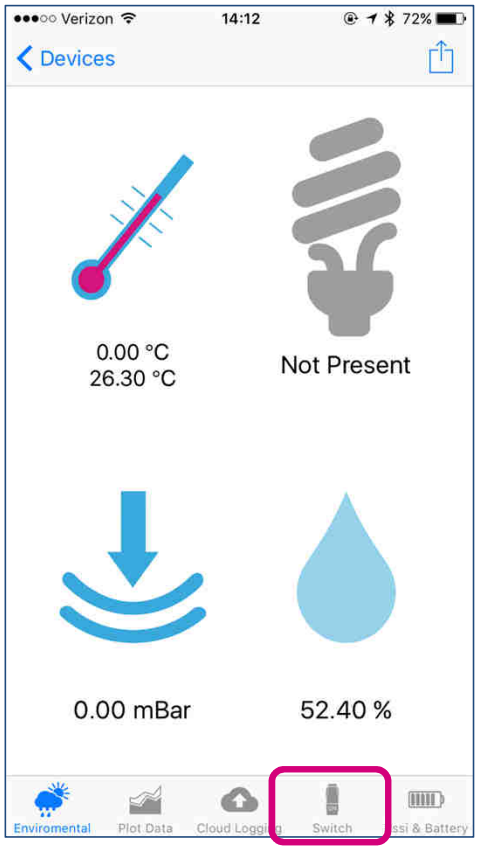
3 **Select your STEVAL-BCN002V1**



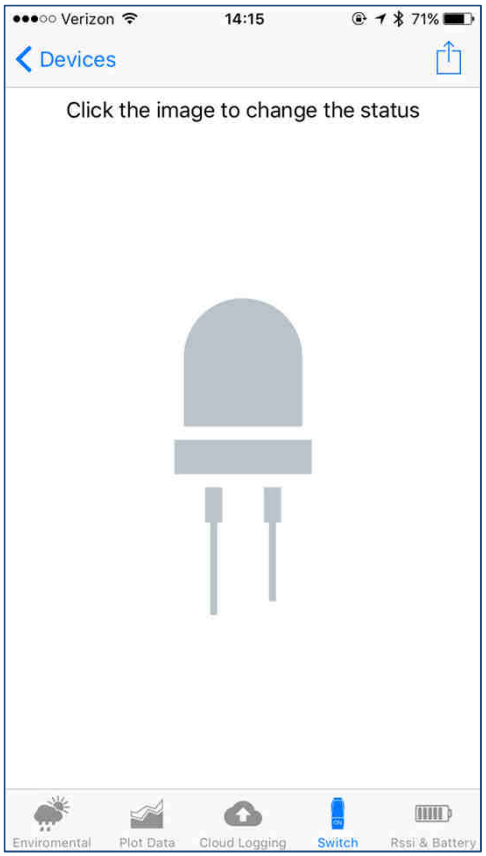
life.augmented

Enabling LED toggling

Click on “**Switch**”

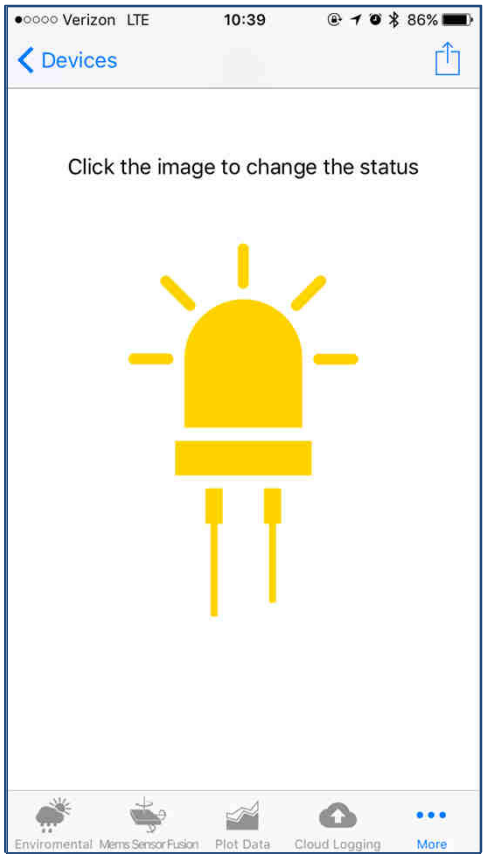


A new page will appear



life.augmented

Enable LED toggling

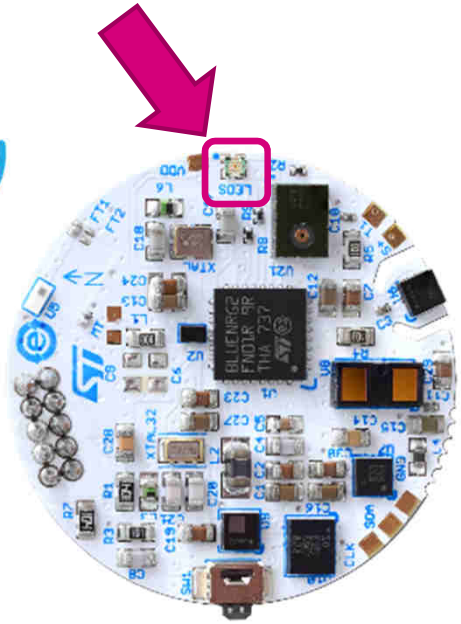


Write Packet

'1' LED ON
'0' LED OFF

Notification Packet

'1' LED ON
'0' LED OFF

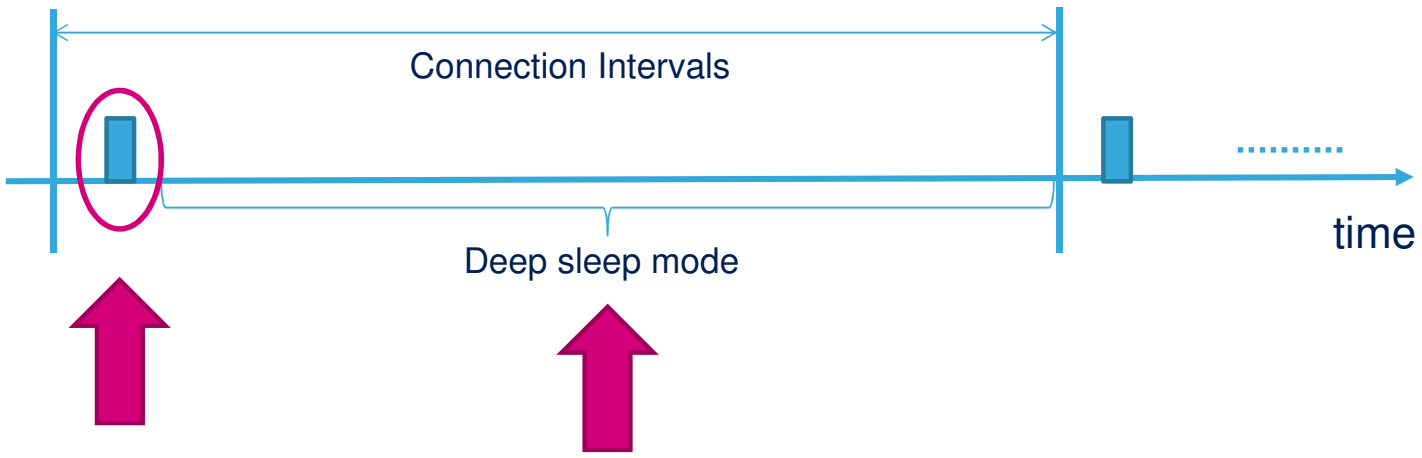


1. **Tap the image** on the screen
2. **Send BLE notification packet**
3. **Image changes color** and **RED LED toggles**

life.augmented

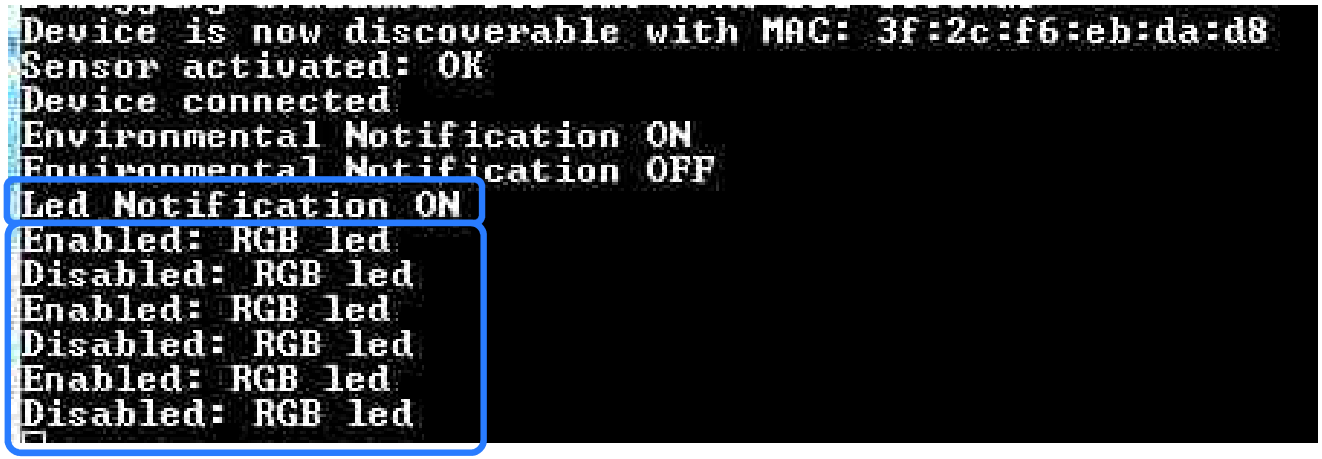
LED fast blinking due to Sleep Mode

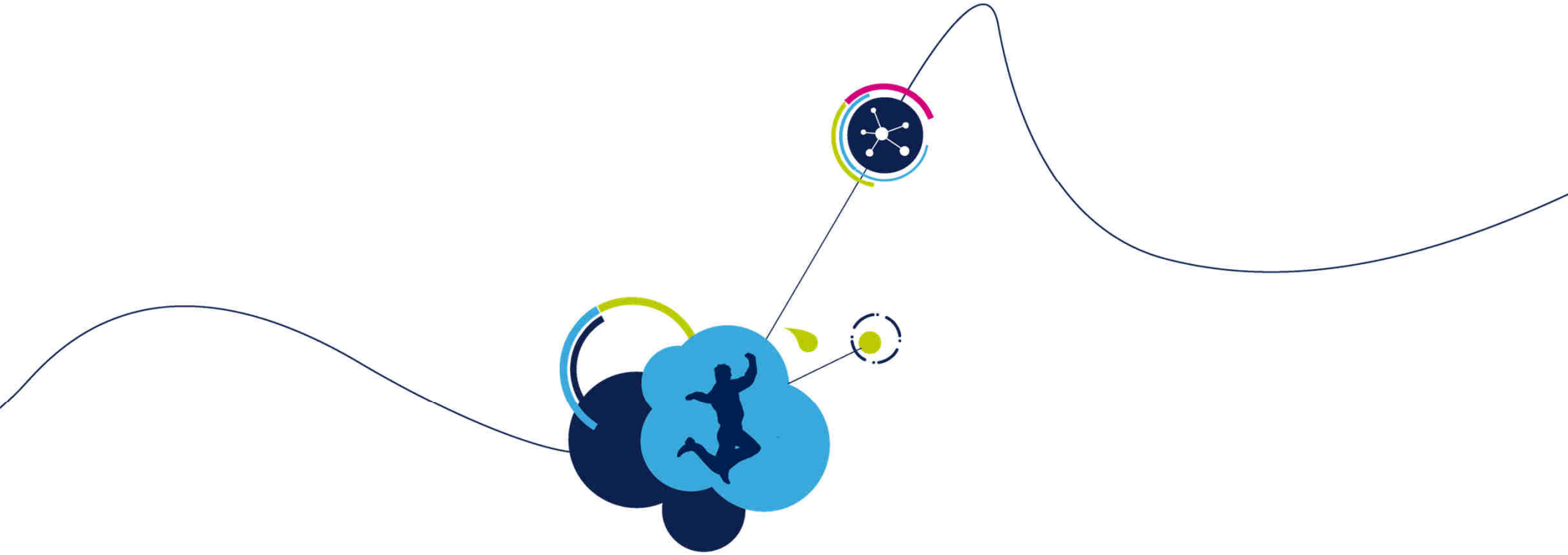
- **Red LED** fast toggling shows the entering/exiting to/from Sleep mode



Tera Term output

- String ***"LED Notification ON"*** will appear as the app tab is enabled
- Each time user **click on the LED icon** in the **BLE Sensor app**, the **Write command is sent** to the board and the **LED toggles** accordingly ("Enabled: RGB led" = ON, "Disabled: RGB led" = OFF)





Lab 4

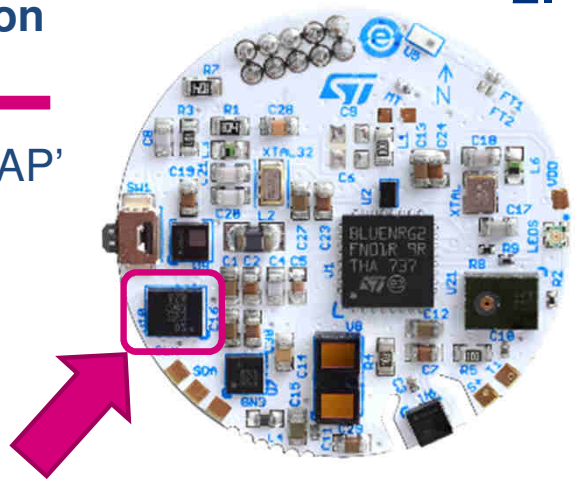
Accelerometer embedded events detection

life.augmented

Example - Single Tap

Notification Packet

‘SINGLE TAP’



1. On the board **perform a Single Tap**
2. **Send BLE notification packet**

NOTE: this is just an example for SingleTap. Other events will be displayed later in the lab!

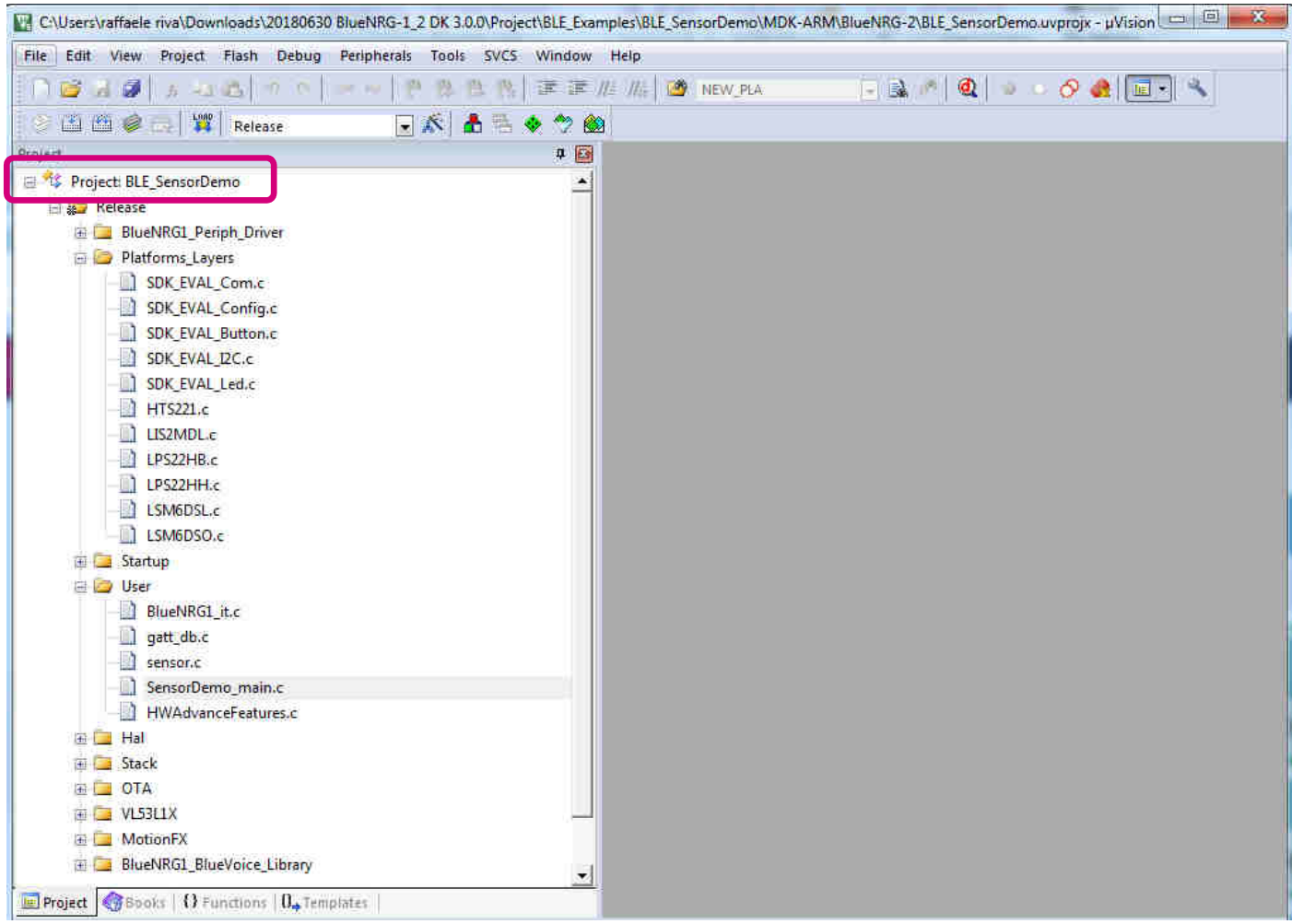
life.augmented

Code modifications

1. **Modify** advertising packet
2. **Enable** Accelerometer **events detection feature**
3. **Read Event Status** and **send BLE notification** through an update on the Accelerometer Event characteristic value

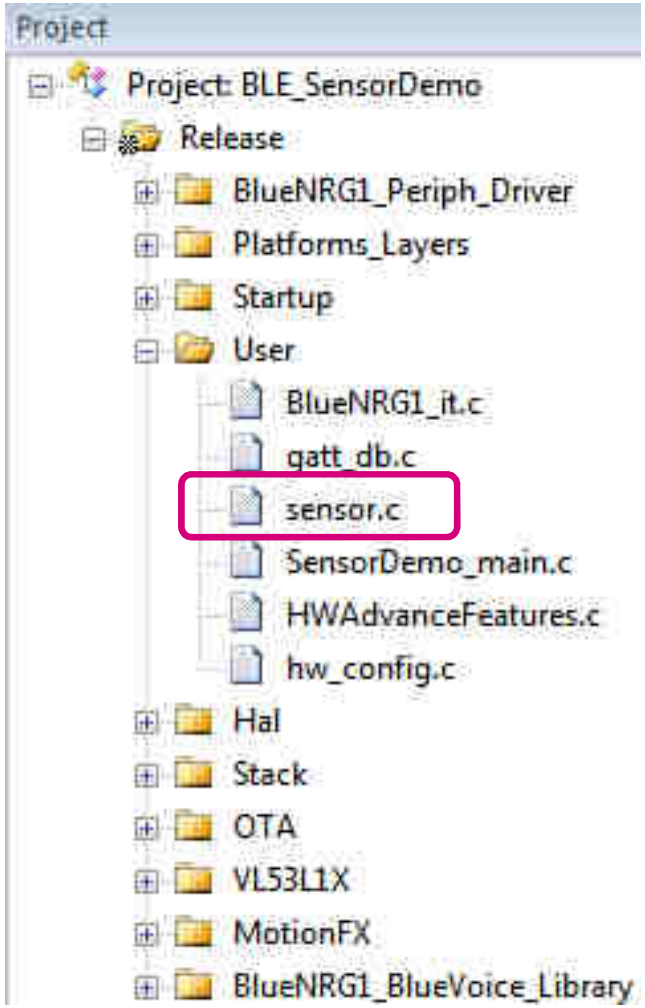
Sensor Demo reference application

Now we **go back** again to **Keil uVision**



life.augmented

L4 STEP1: Modify advertising packet



ST BlueMS Protocol

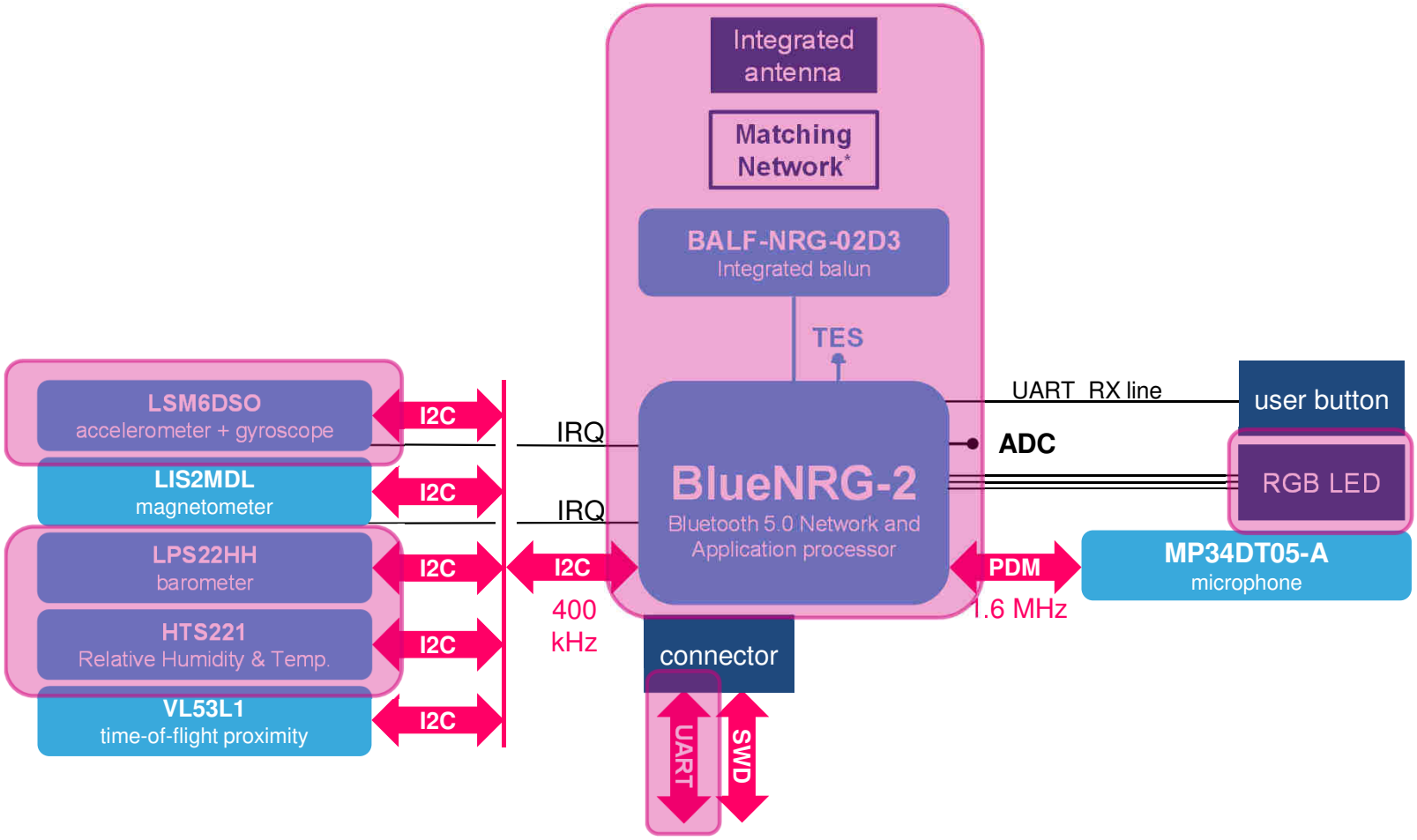
BlueMS

4 Bytes

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	0x20
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0x9E
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	0x04
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x00

life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

L4 STEP1: Modify advertising packet

Modify the Feature Mask in the advertisement payload

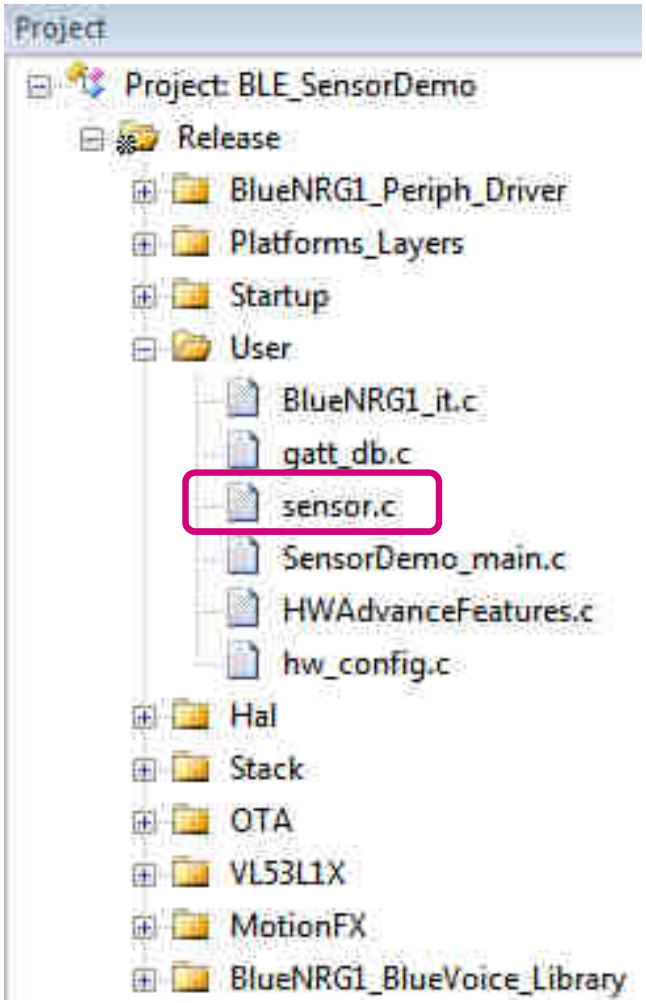
1. In the file **sensor.c**
2. **Go to line 424 and 425**
3. **Modify** the **Feature Mask byte#2** from **0x1E** to **0x9E**
4. **Modify** the **Feature Mask byte#3** from **0x00** to **0x04**

sensor.c

```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                          /* Length of AD type Transmission Power */
415   0x0A, 0x00,                 /* Transmission Power = 0 dBm */
416   8,                          /* Length of AD type Complete Local Name */
417   0x09,                       /* AD type Complete Local Name */
418   NAME_ALLMEMS,               /* Local Name */
419   13,                         /* Length of AD type Manufacturer info */
420   0xFF,                       /* AD type Manufacturer info */
421   0x01,                       /* Protocol version */
422   0x05,                       /* Device ID: 0x05 = STEVAL-BCN002V1 Board */
423   0x20,                       /* Feature Mask byte#1: LAB3 0x20 (LED) / LAB5
424   0x9E,                       /* Feature Mask byte#2: LAB4 0x9E (Acc+Press+Hu
425   0x04,                       /* Feature Mask byte#3: LAB4 0x04 (AccEvents) /
426   0x00,                       /* Feature Mask byte#4: LAB5 0x40 (eCompass) */
```

life.augmented

L4 STEP2: Enable Accel events feature



L4 STEP2: Enable Accel events feature

Enable multiple accelerometer embedded events detection

1. In the same file **sensor.c**
2. **Go to line 54** to **line 58**
3. **Set** the defines **from line 54 to line 58** from 0 to **1**

sensor.c*

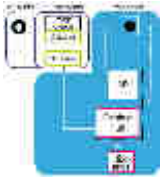
```
53  /* ACCELEROMETER EVENTS */
54  #define ENABLE_FREE_FALL      1
55  #define ENABLE_SINGLE_TAP     1
56  #define ENABLE_HW_WAKEUP      1
57  #define ENABLE_TILT           1
58  #define ENABLE_PEDOMETER      1
```

LSM6DSO at a glance

1. PERFORMANCE AND FEATURE

- **Accuracy:** Tango and Daydream compliance
- **0.55mA** current consumption is **HP combo**
  -15% vs. LSM6DSL/M @ same performance

- New **ultra low power mode: 14uA (@100Hz ODR) Axl only**

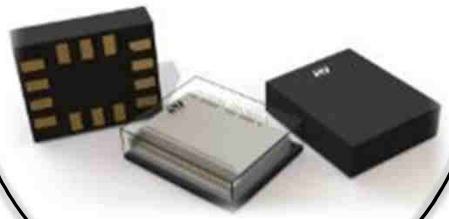


2. FLEXIBILITY: algorithm @ silicon level

- **Pedometer 2.x**
  WeChat Precision

- **FSM** build custom sensors for XL and Gyro

LSM6DSO



2.5x3x0.86 mm

3. NEW STANDARD

- **I3C**

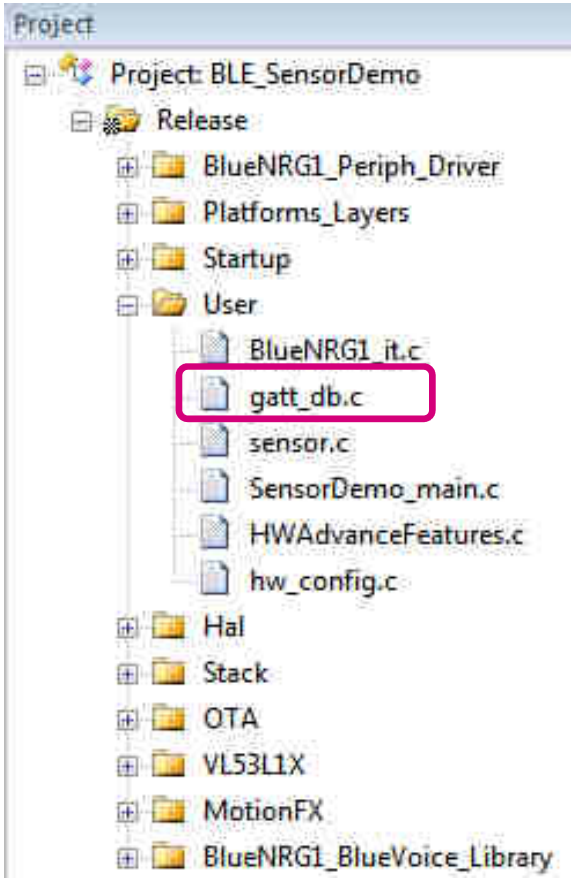
- **FIFO tag**

4. INNOVATION

- MEMS: **Axl T-structure** for mechanical shock robustness
- **Compressed FIFO (3x)**
- **10 patents protecting LSM6DSO innovation**

life.augmented

L4 STEP3: Client enables Acc Event characteristic notifications



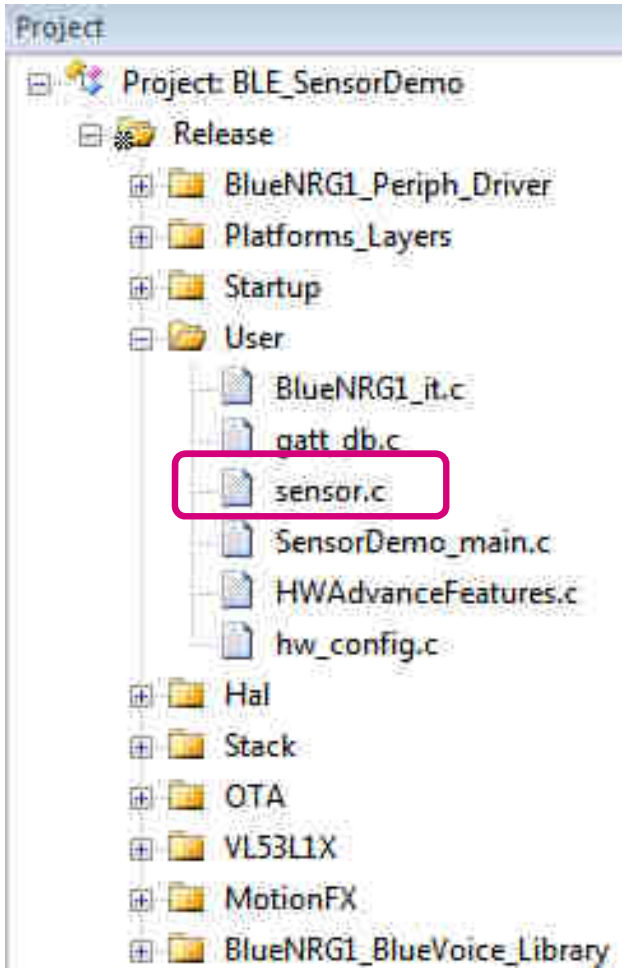
L4 STEP3: Client enables Acc Event characteristic notifications

1. In the file **gatt_db.c** **go to** the **line 717**
2. **Client writes** in the **Acc Event Characteristic Client Configurator Descriptor (CCCD)** and **Server enables notifications** through the ***xFeatureNotification*** **structure** in **line 728** by ***xFeatureNotification.AccEventNotification = true***

```
717   } else if (attr_handle == (AccEventCharHandle + 2)) {
718
719     if (xFeatureNotification.MotionNotification == false && xFeatureNotification.iNemoEngineNotification == false) {
720       if (att_data[0] == 01) {
721         lsm6dso_xl_data_rate_set(0, LSM6DSO_XL_ODR_52Hz);
722         GPIO_WriteBit(GPIO_Pin_7, Bit_RESET); // Proximity OFF
723         EnableHWMultipleEvents();
724         ResetHWPedometer();
725         GPIO_EXTICmd(GPIO_Pin_13, ENABLE);
726         Config_Notify(FEATURE_MASK_ACC_EVENTS, 'm', 1);
727         AccEvent_Notify(0, 3);
728         xFeatureNotification.AccEventNotification = true;
729         PRINTF("AccEvent Notification ON\n\r");
730
731       } else if (att_data[0] == 0) {
732         lsm6dso_xl_data_rate_set(0, LSM6DSO_XL_ODR_OFF);
733         xFeatureNotification.AccEventNotification = false;
734         GPIO_EXTICmd(GPIO_Pin_13, DISABLE);
```

life.augmented

L4 STEP4: Read Event Status and send BLE notification



L4 STEP4: Read Event Status and send BLE notification


```
1179 void MEMSCallback(void) {
1180
1181   lsm6dso_all_sources_get(0, &all_source);
1182
1183   /* Check if the interrupt is due to Single Tap */
1184   if (xHardwareFeaturePresence.HwSingleTAP || xHardwareFeaturePresence.MultipleEvent) {
1185     if (all_source.reg.all_int_src.single_tap) {
1186       SdkEvalLedOn(LED1);
1187       SdkEvalLedOn(LED3);
1188       PRINTF("Event: Single Tap\n\r");
1189 #if ENABLE_SINGLE_TAP
1190       AccEvent_Notify(ACC_SINGLE_TAP, 2);
1191 #endif
1192     }
1193   }
```

In the file **sensor.c**

1. line 1179 ***void MEMSCallback(void)***

   Callback triggered by IO13

2. line 1181 ***lsm6dso_all_sources_get***

   Read accelerometer status registers

3. line 1185 ***if (all_source.reg.all_int_src.single_tap)***

   Check vs. single tap event

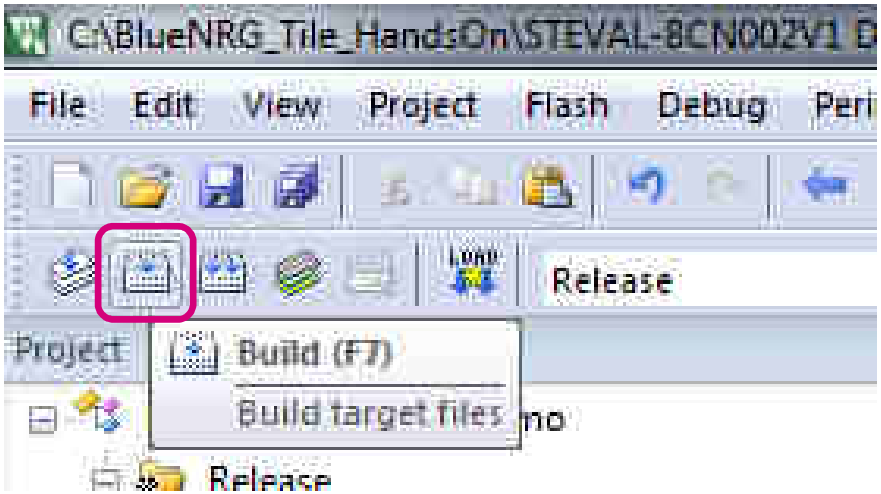
4. line 1190 ***AccEvent_Notify***

   Send BLE notification

life.augmented

Build the new code

1. **Click** on the **Build button** (top left corner) or **hit F7** on your keyboard
2. In the **Build Output** window (bottom) **wait** for the build to be completed.
   - **BLE_SensorDemo.bin** created
   - "**0 Error(s), 0 Warning(s)**" message appear

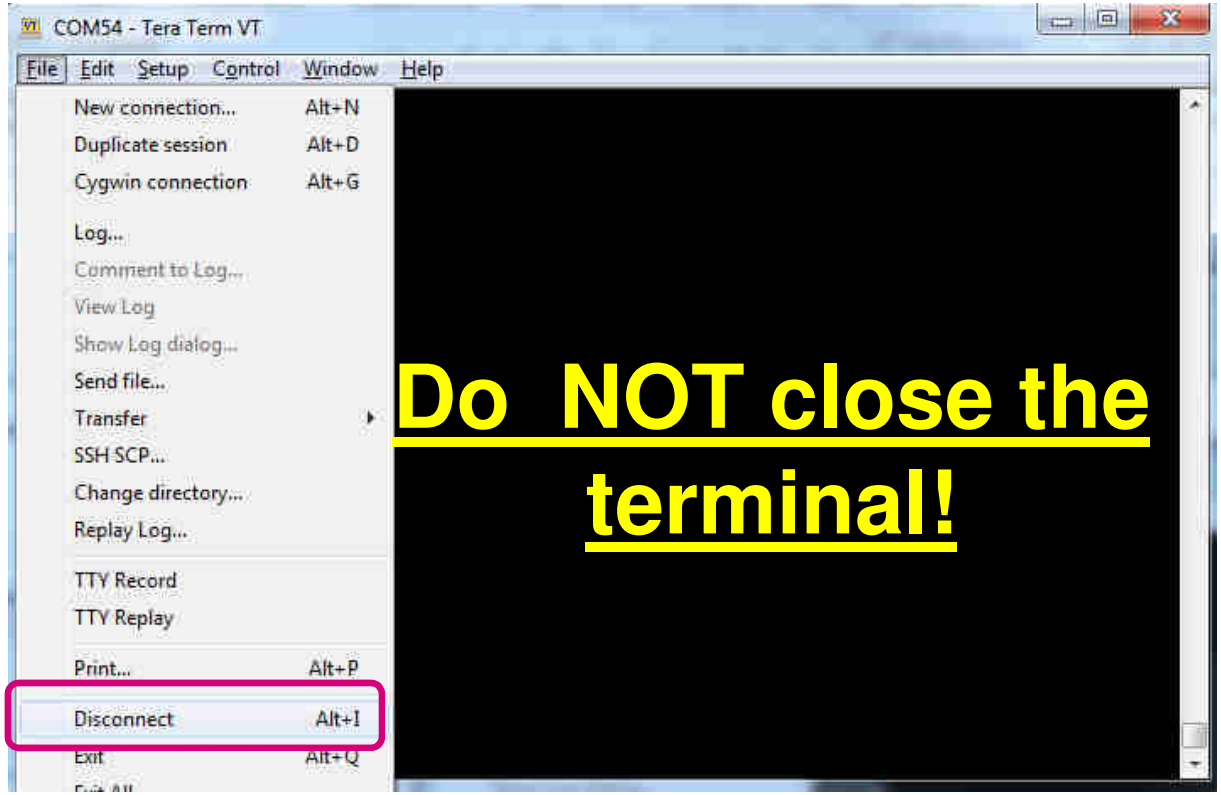



```
Build Output
compiling vl53l1_wait.c...
compiling vl53l1_register_funcs.c...
compiling vl53l1_platform.c...
linking...
Program Size: Code=121908 RO-data=1428 RW-data=1136 ZI-data=21252
FromELF: creating hex file...
After Build - User command #1: fromelf.exe --bin ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf --output ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.bin
"..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:12
```

life.augmented

Disconnect the serial terminal

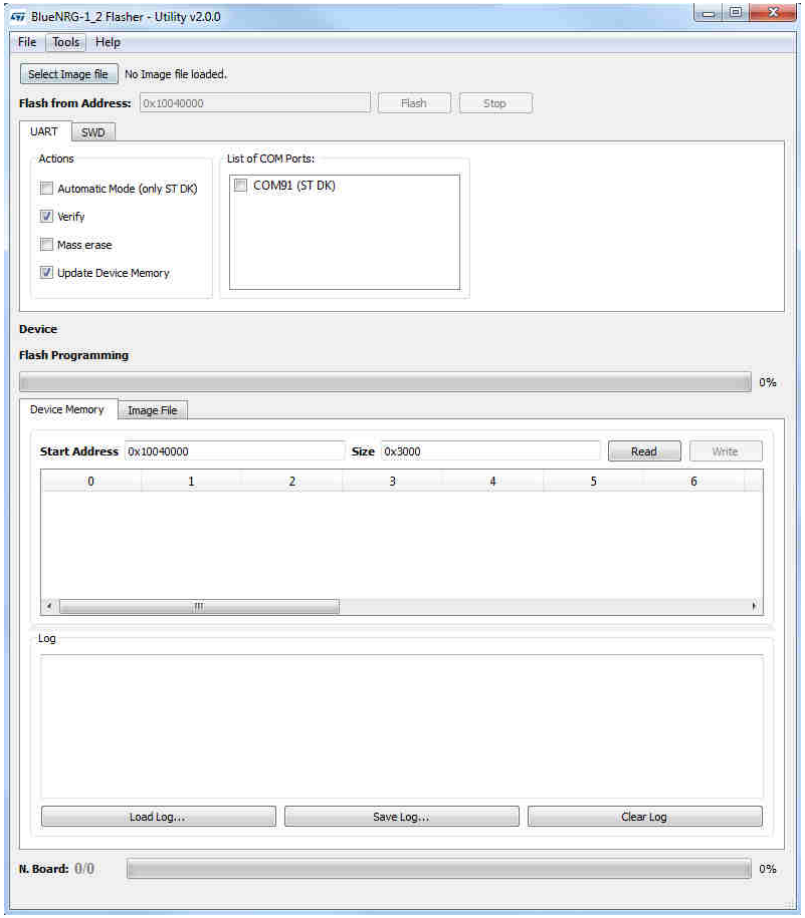
1. **Go back** to **Tera Term**
2. **Click** on the **File->Disconnect**



life.augmented

Re-flash the BlueNRG-2

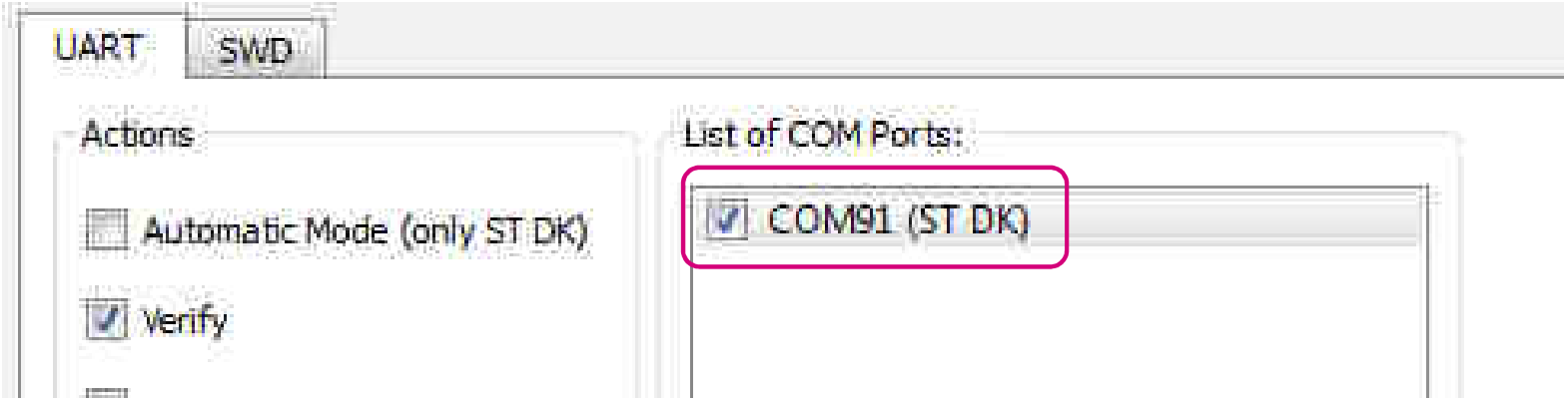
1. **Go back** to the **BlueNRG-2 Flasher Utility**



life.augmented

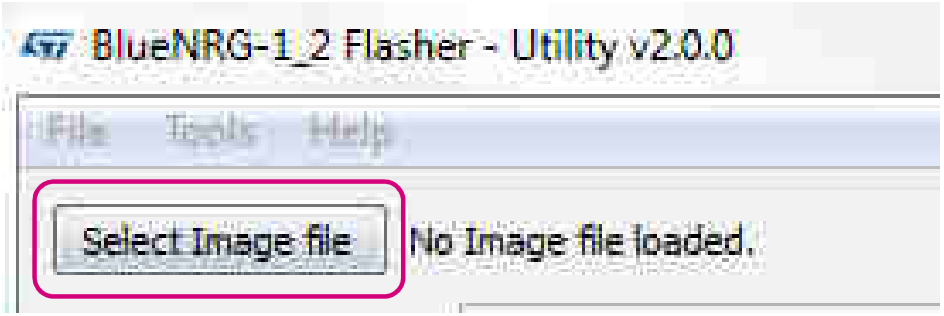
Flash the BlueNRG-2 1/4

1. **Select** the COM port labeled **(ST DK)**



Flash the BlueNRG-2 2/4

1. **Click** on the **Select Image file** button

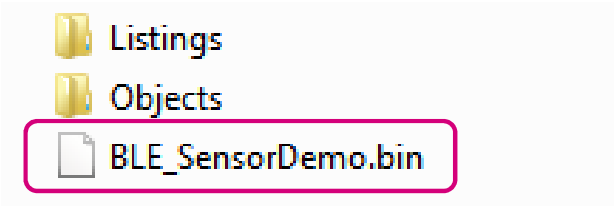


2. And **browse** the following **path**

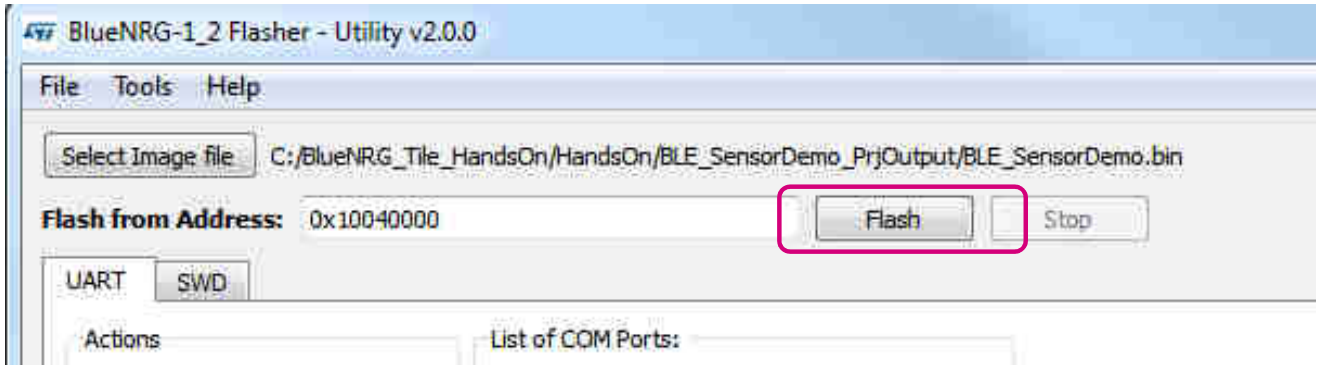
life.augmented

Flash the BlueNRG-2 3/4

1. **Select BLE_SensorDemo.bin** and **Click OK**



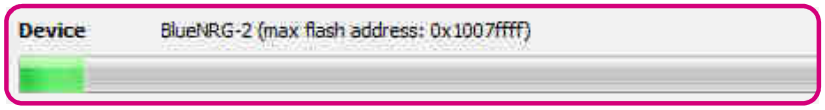
2. **Click** on the **Flash** button



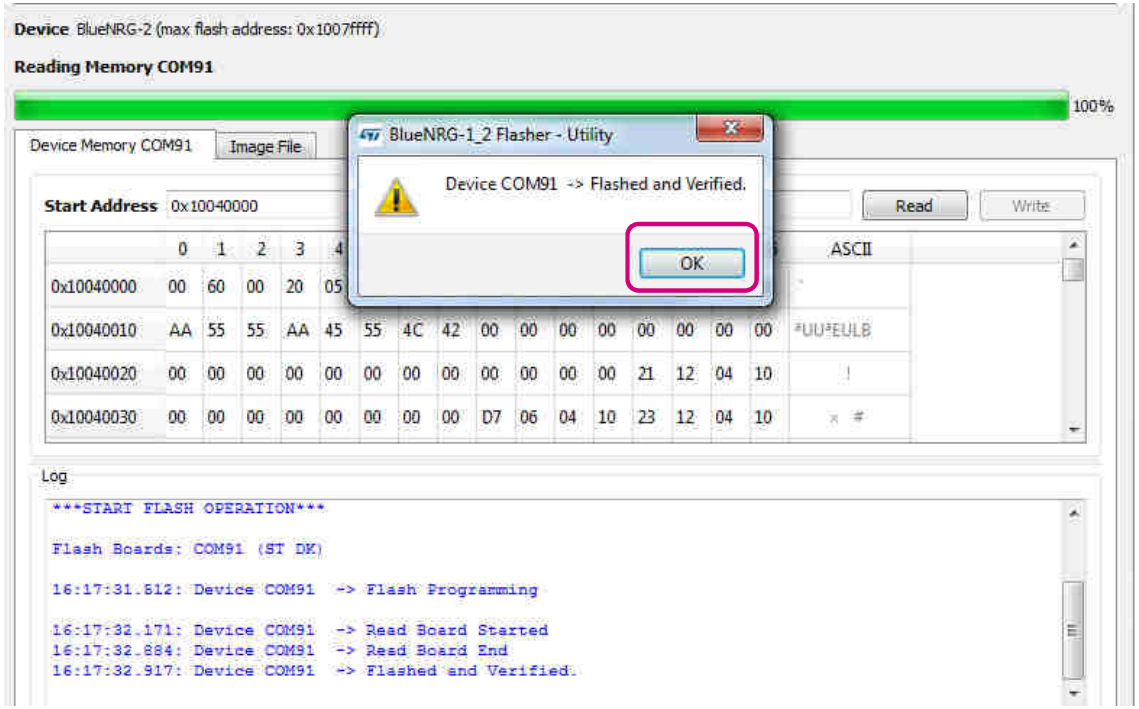
life.augmented

Flash the BlueNRG-2 4/4

1. Flashing starts: **green bar** proceeding



2. **Wait** for the **pop-up** window and **click** on **OK**

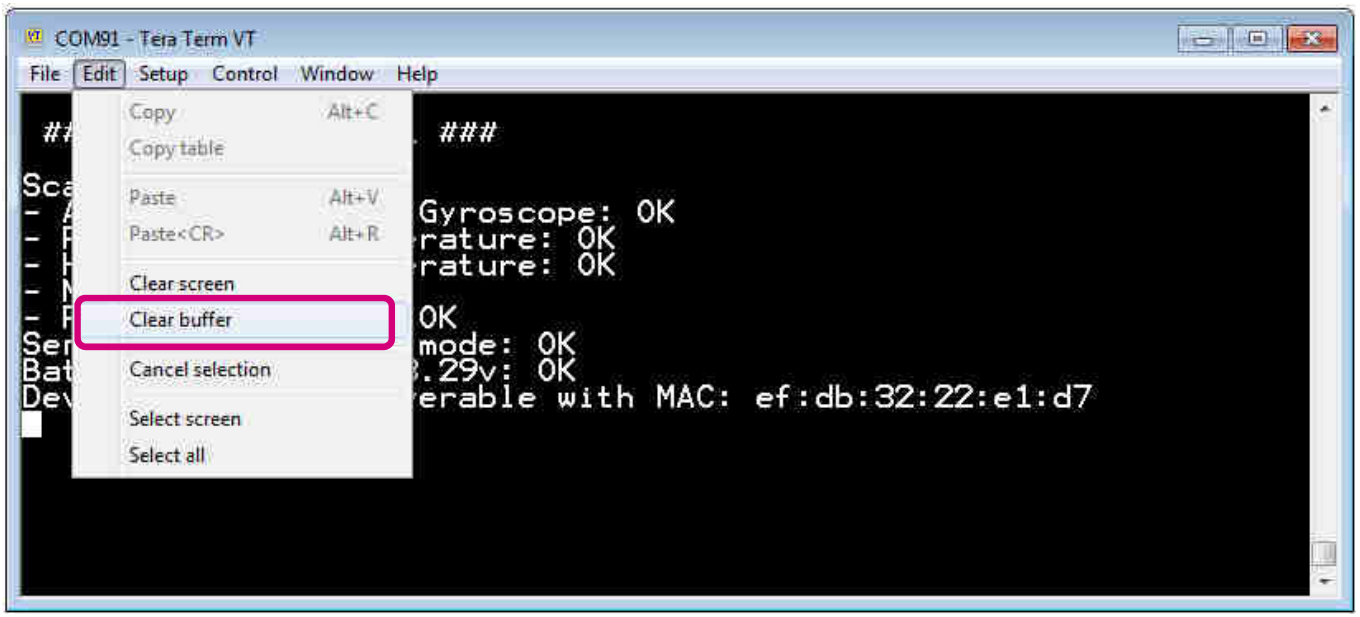


Do NOT close the Utility!

life.augmented

Clean Buffer in the serial terminal

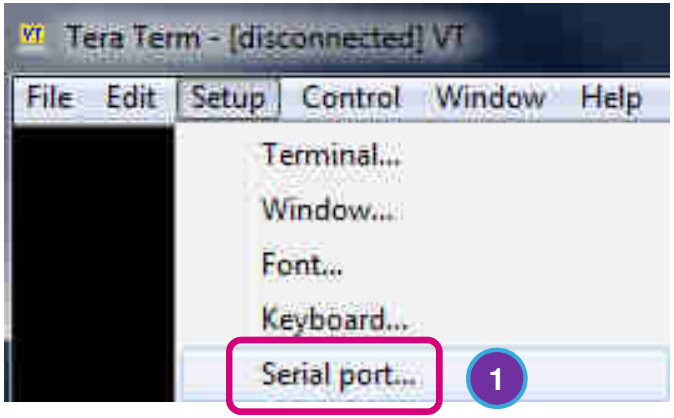
1. In **Tera Term** in order to have the terminal clean, **go to Edit -> Clear buffer**

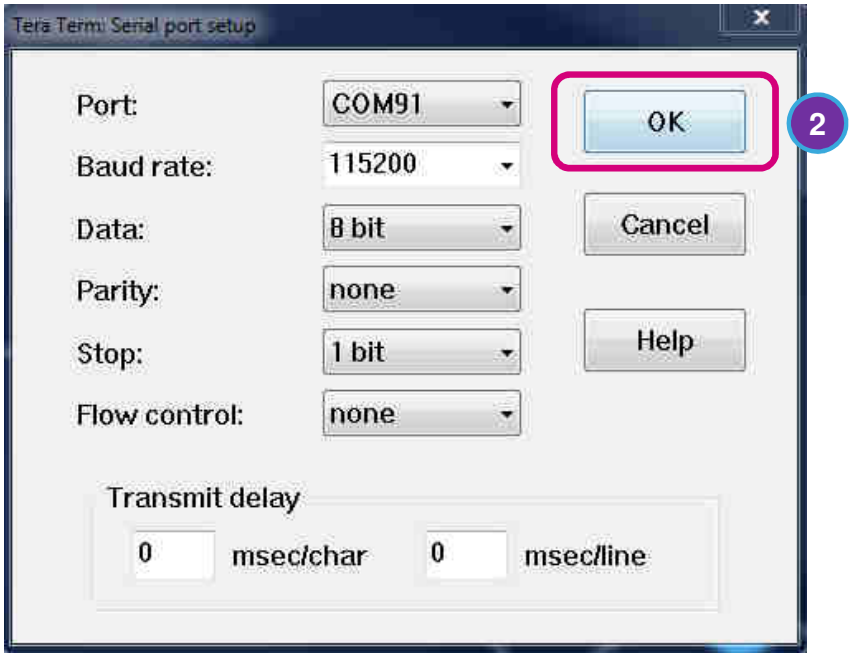


life.augmented

Reconnect the serial terminal

1. **Click Setup -> Serial port…**
2. Serial port should be already configured.
   Just need to **click** on **OK**

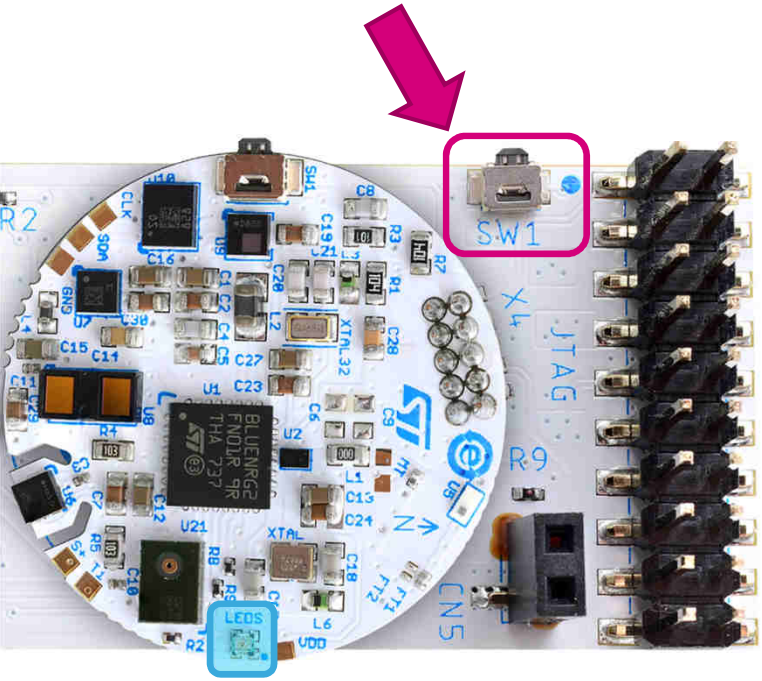


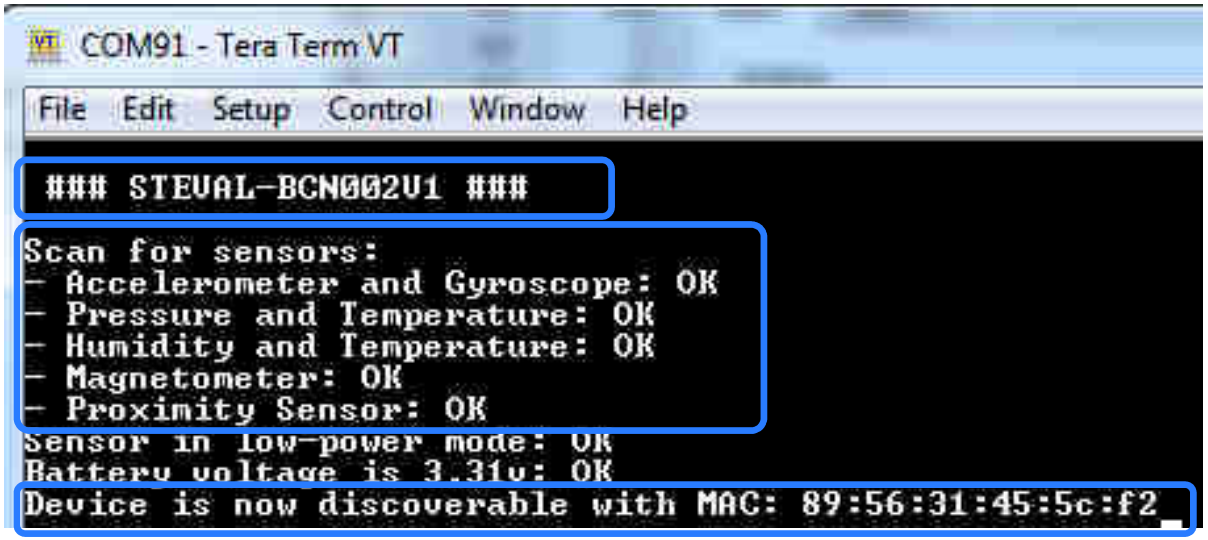


life.augmented

Sanity Check on serial port

Push SW1 button on the daughterboard -> **LED blinking Blue**

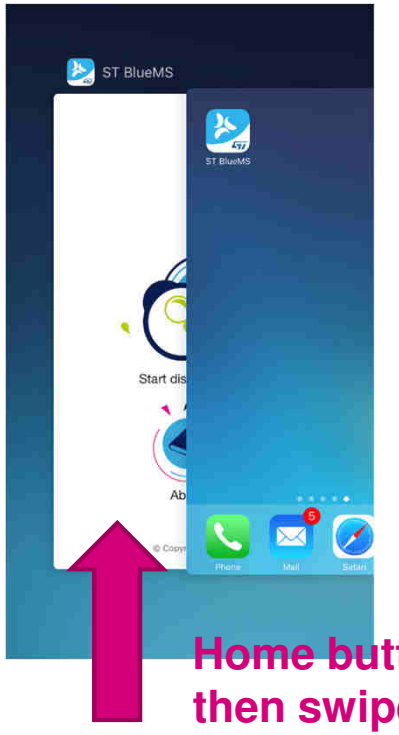




life.augmented

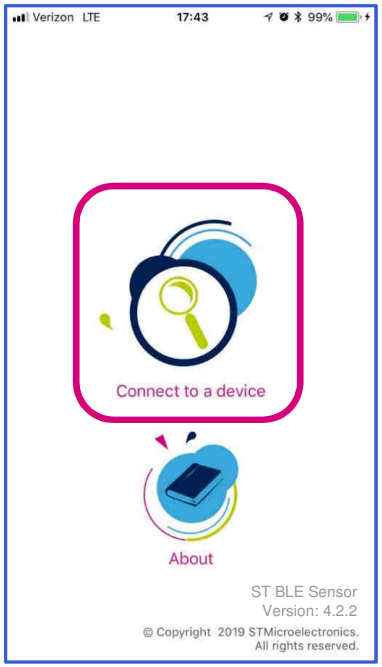
Connect using the ST BLE Sensor App

1 **"Kill" the BLE Sensor app**

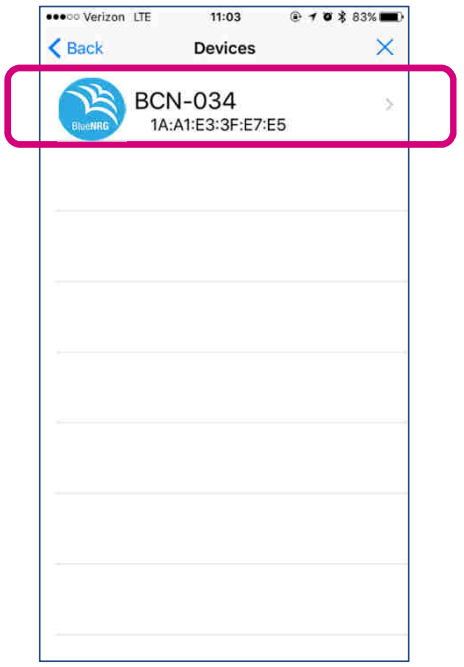


Home button and then swipe to top

2 **Tap "Connect to a device"**

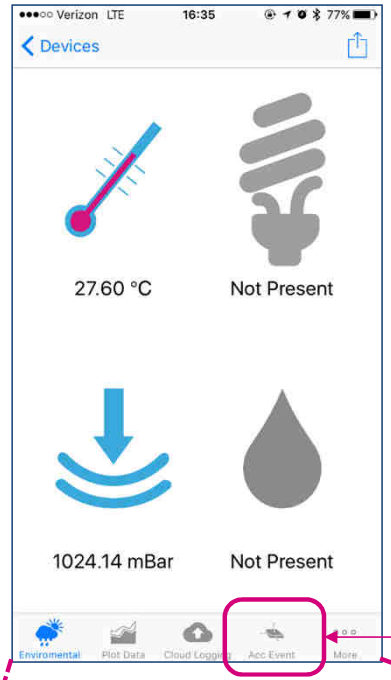


3 **Select your STEVAL-BCN002V1**



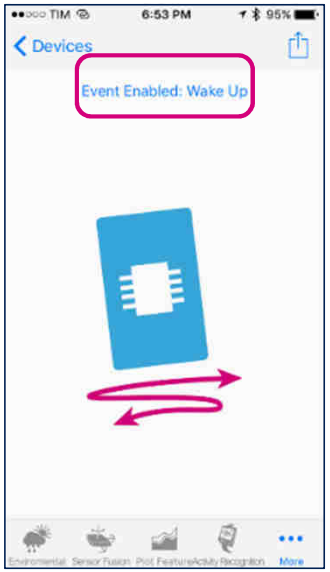
life.augmented

Visualize Single Tap Event in ST BLE Sensor



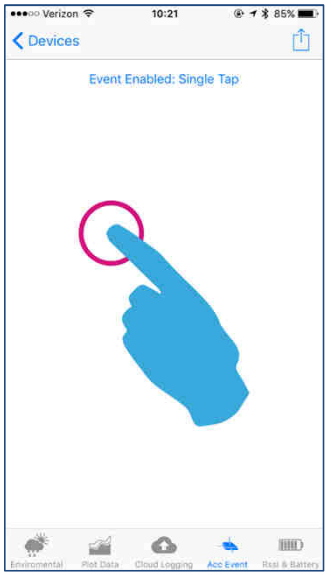
Select
“Acc Event”

Tap
“Event Enabled”



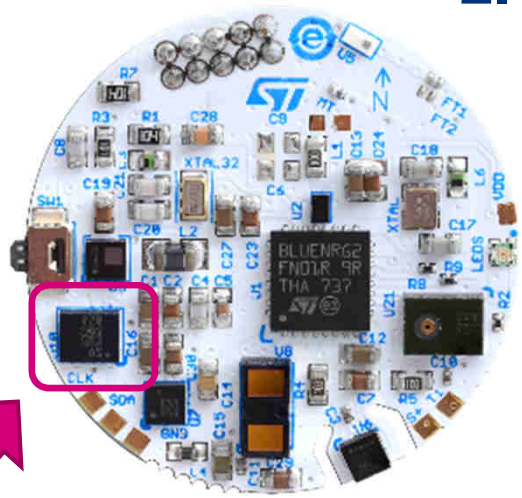
Select
“Single Tap”
and hit **“Select”**

Enabled
“Single Tap”



Event Detection

1. On the board **perform** a **Single Tap**
2. **Send BLE notification packet**



Tap **gently** on the board

Notification Packet

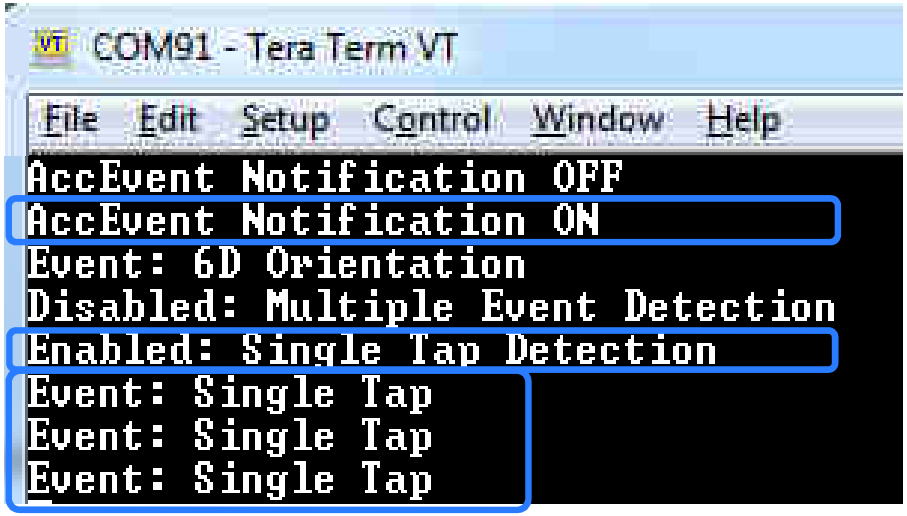
‘SINGLE TAP’

Single Tap

life.augmented

Tera Term output

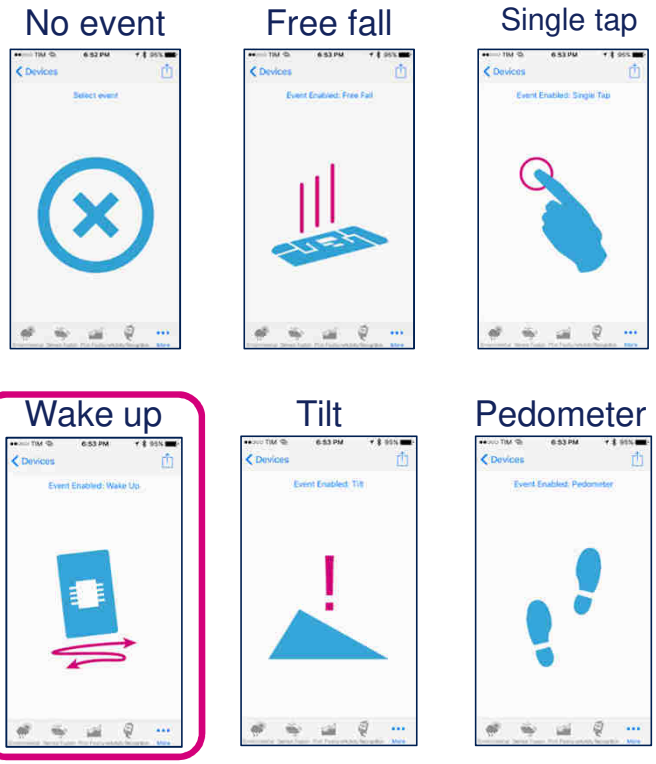
- ***"AccEvent Notification ON":*** **enable notifications** on the Accelerometer Event characteristic
- ***"Enabled Single Tap Detection":*** user **enables Single Tap event detection**
- ***"Event: Single Tap":*** the actual **BLE notification packet** is sent upon detection of the Single Tap event.



NOTE: if the phone display enters sleep mode, the notifications are not sent

LSM6DSO Embedded Events

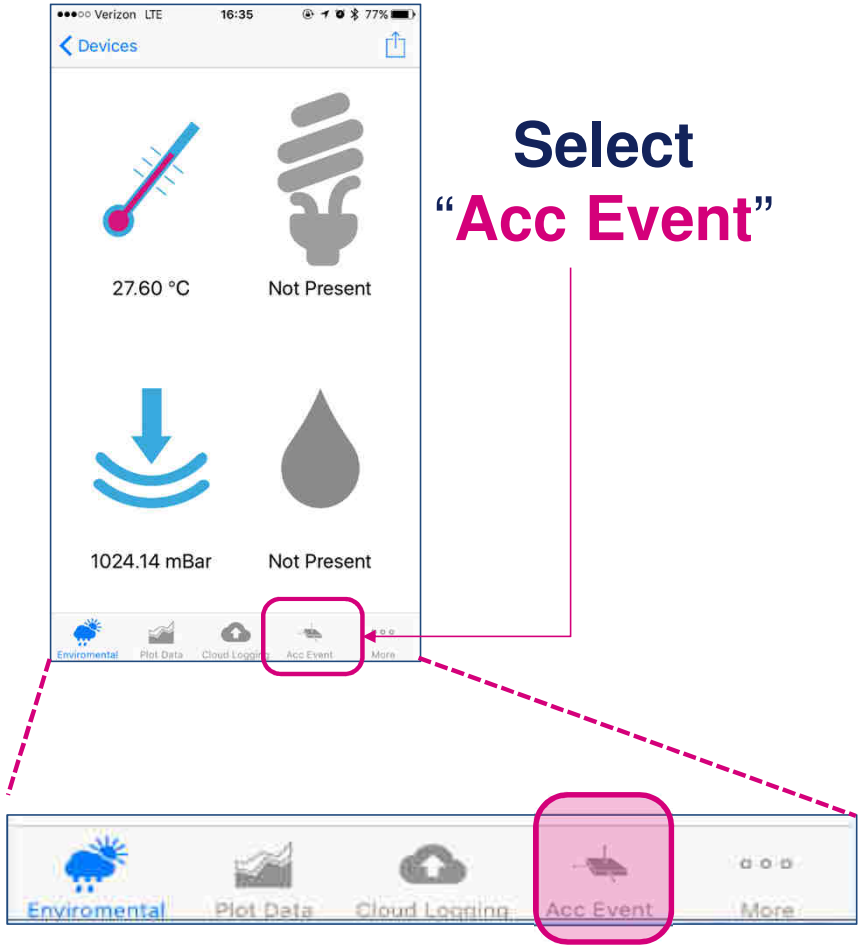
Other possible embedded events

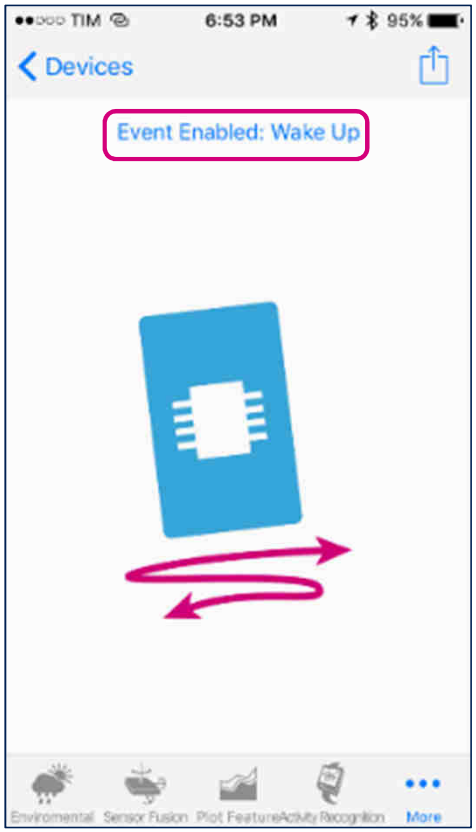


WakeUp example

life.augmented

Visualize Hardware Wakeup Event in BLE Sensor

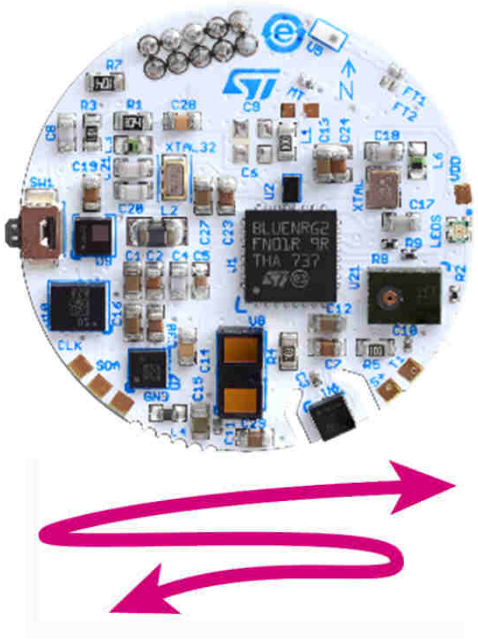




life.augmented

Event Detection

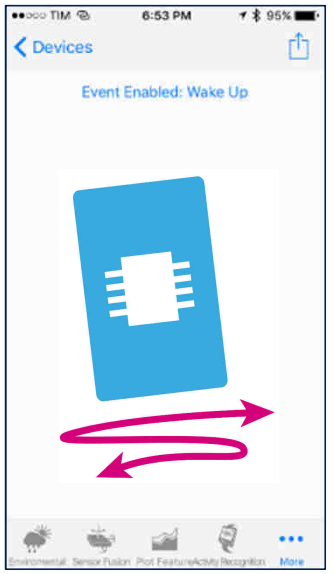
1. **Leave** the board **still** for a few seconds
2. **Shake** the board
3. **Send BLE notification packet**



Notification Packet

‘WAKE UP’

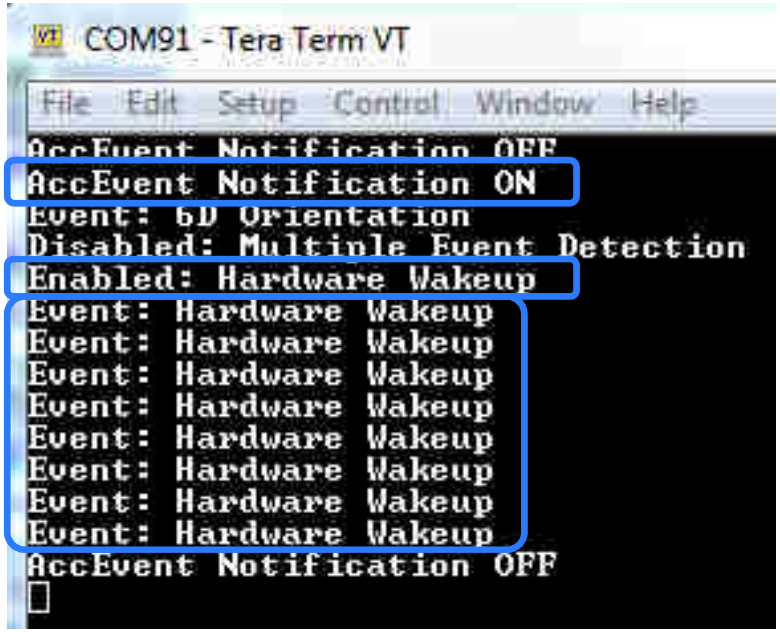
Wake Up



life.augmented

Tera Term output

- ***"AccEvent Notification ON":*** **enable notifications** on the Accelerometer Event characteristic
- ***"Enabled Hardware Wakeup":*** user enable hardware **wakeup event detection**
- ***"Event: Hardware Wakeup":*** the actual **BLE notification packet sent** upon detection of the Hardware Wakeup event.



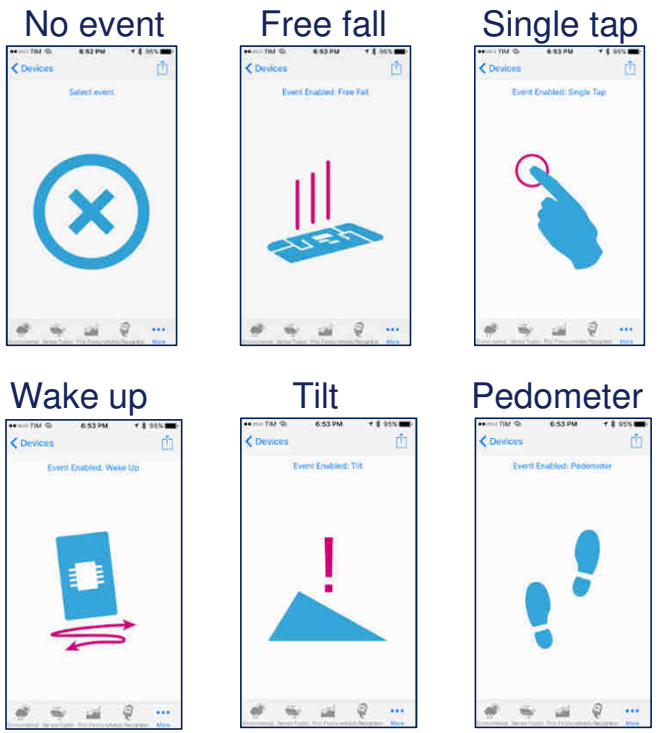
NOTE: if the phone display enters sleep mode, notifications are not sent

life.augmented

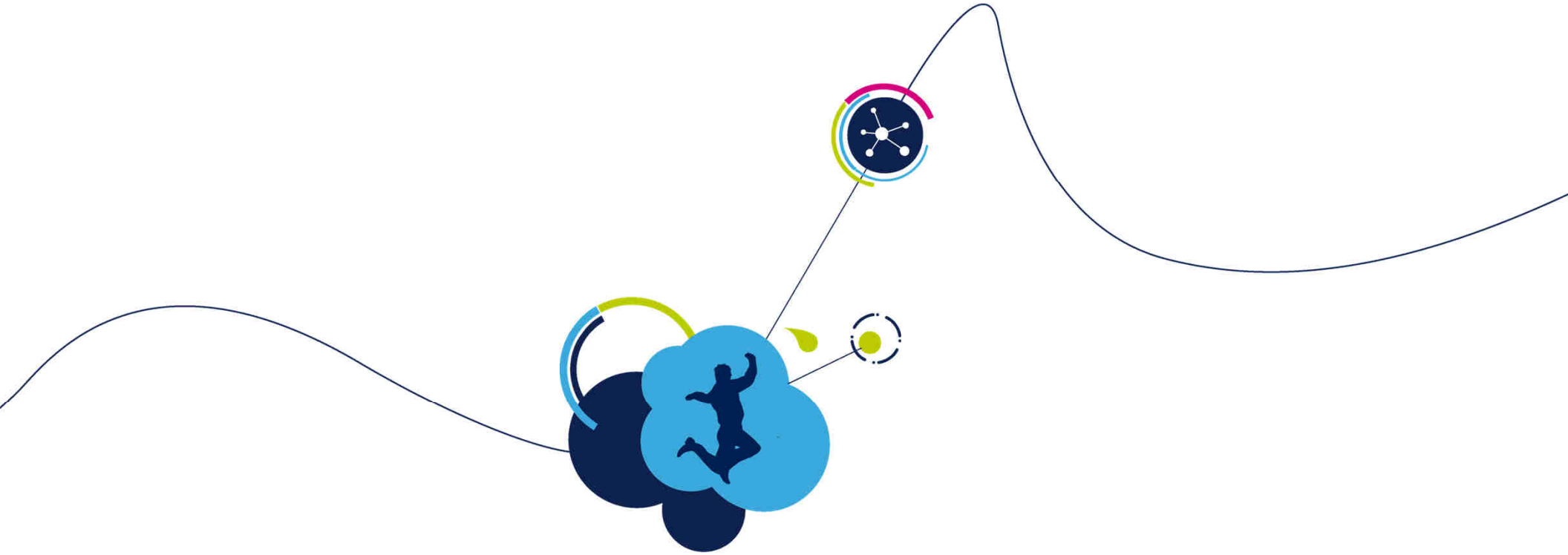
LSM6DSO Embedded Events

You can go ahead later and test other events…

Other possible embedded events



life.augmented



Lab 5

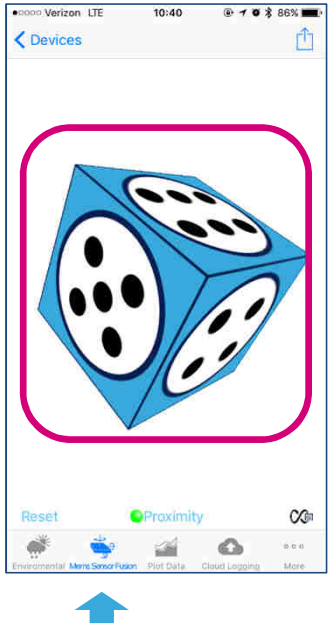
9-axis Acc+Gyro+Mag Sensor Data Fusion

life.augmented

Goal

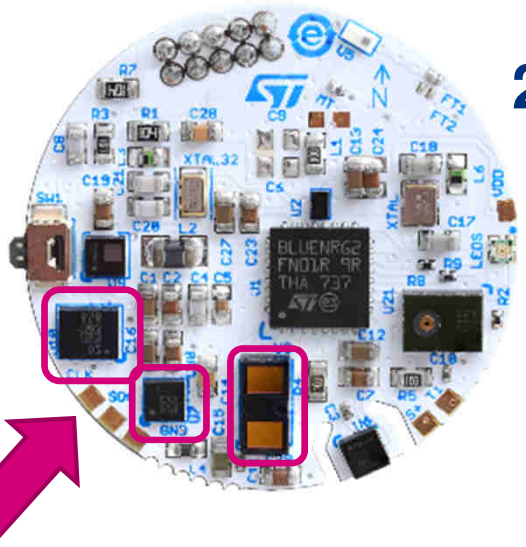
- In this example we are going to demonstrate how to:
  - **Enable** the embedded sensor data fusion library
    - Input: raw Acc+Gyro+Mag sensor data
    - Output: quaternions
  - **Scale the quaternions value** by a scaling factor proportional to proximity detection
  - **Send** scaled data fusion information through **BLE notifications packets** to the ST BLE Sensor app client

Sensor Data Fusion



Notification Packet

QUATERNIONS



1. **Enable** Sensor Fusion library
2. **Send** quaternions information through **BLE notification packet**

Sensors used by data fusion:
Acc+Gyro+Mag and Proximity

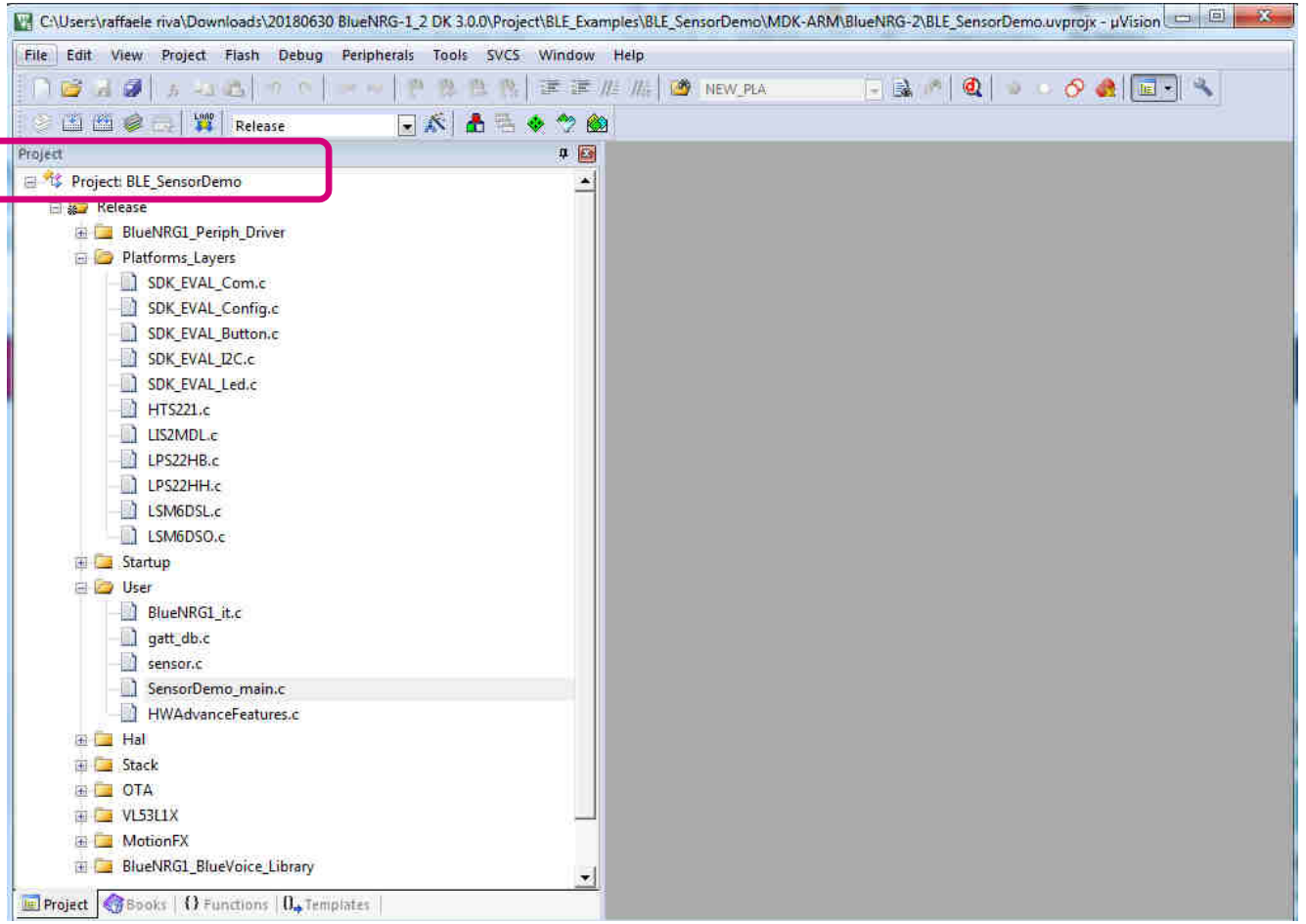
life.augmented

Code modifications

1. **Modify advertising packet**
2. **Enable Sensor Data fusion library**
3. **Enable proximity sensor**
4. **Send quaternions data** – scaled by proximity sensor data - through **BLE notifications packets**

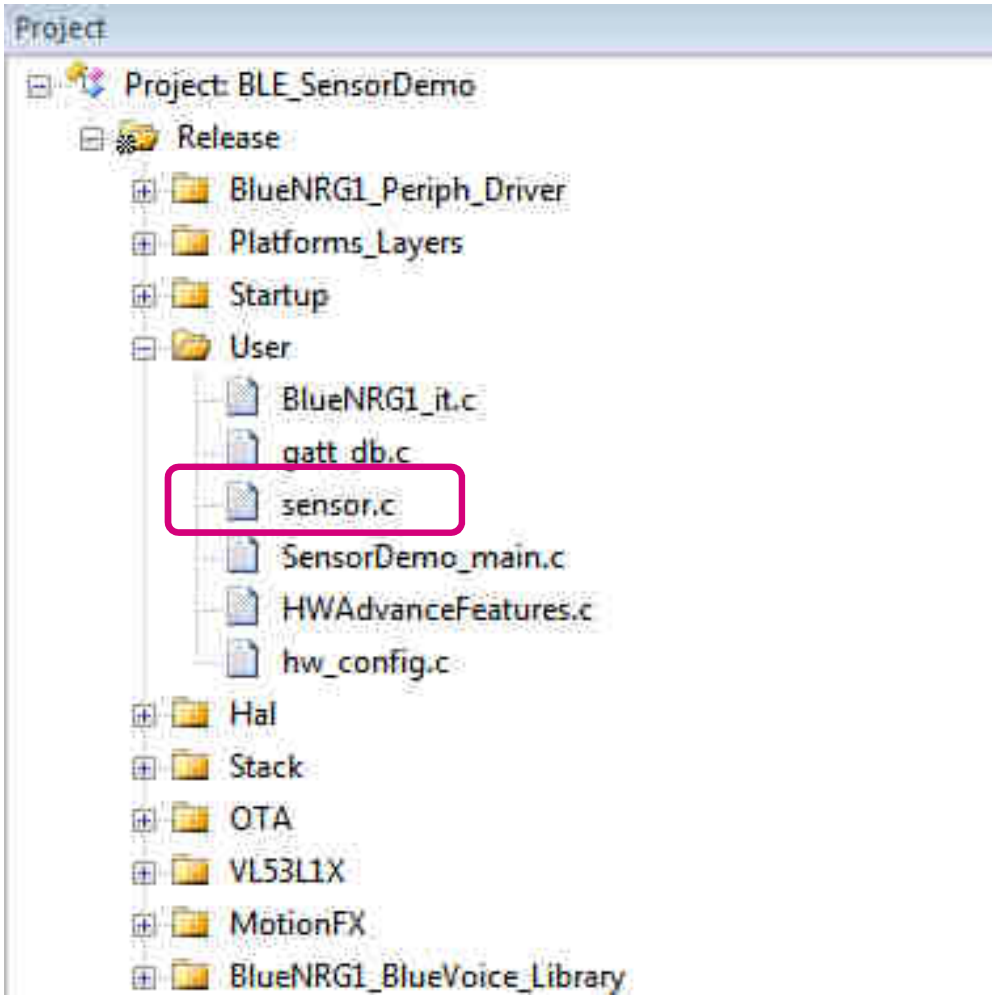
Sensor Demo reference application

Now we **go** **back** again to **Keil uVision**



life.augmented

L5 STEP1: Modify advertising packet



ST BlueMS Protocol

BlueMS

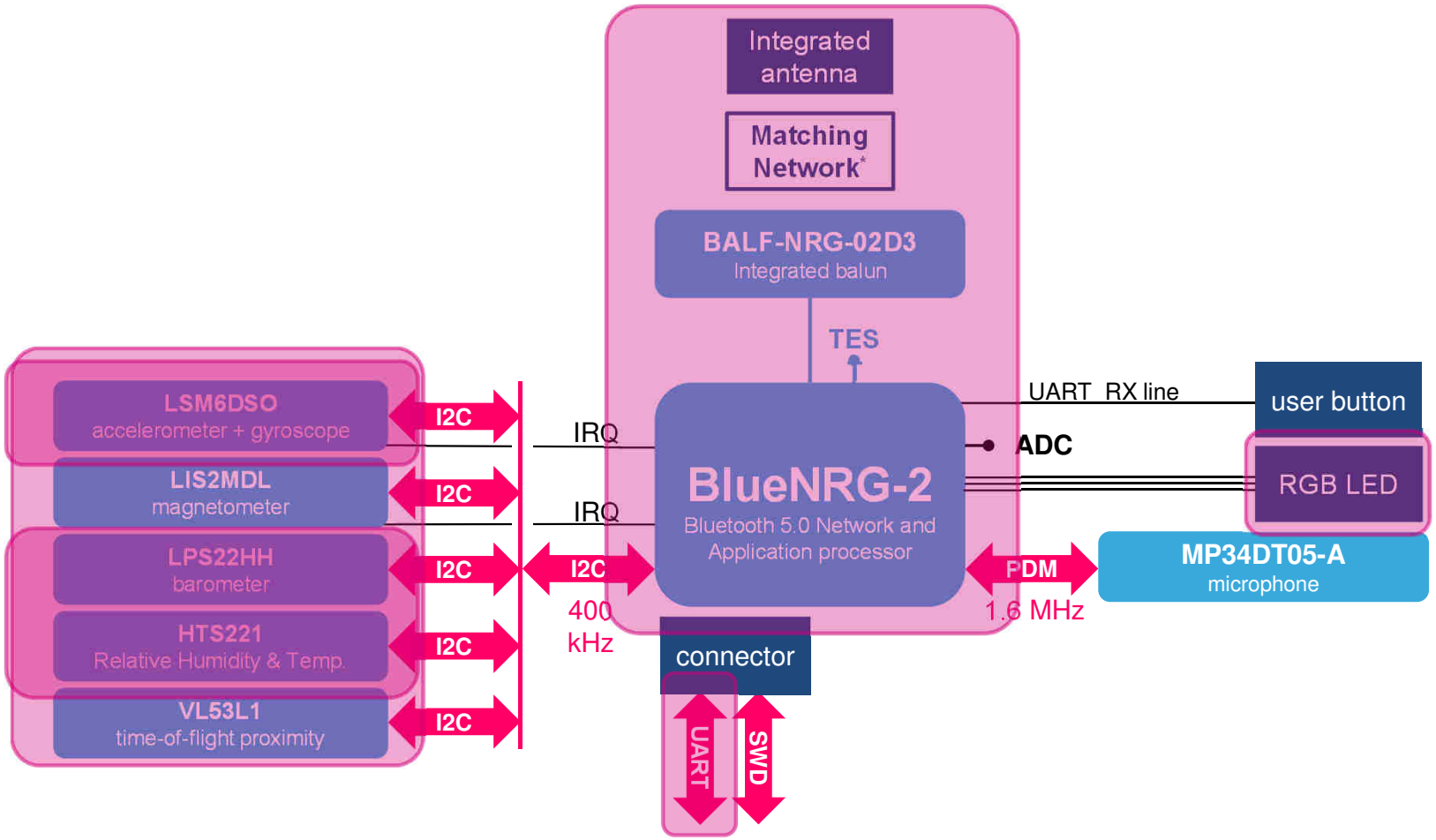
4 Bytes

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name	Len	Type	Pwr

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	0x22
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0xFE
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	0x05
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x40

life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

L5 STEP1: Modify advertising packet

Modify the Feature Mask in the advertisement payload

1. In the file **sensor.c** and **go to line 423, 424, 425** and **426**
2. **Modify line 423 Feature Mask byte#1** from **0x20** to **0x22**
3. **Modify line 424 Feature Mask byte#2** from **0x9E** to **0xFE**
4. **Modify line 425 Feature Mask byte#3** from **0x04** to **0x05**
5. **Modify line 426 Feature Mask byte#4** from **0x00** to **0x40**

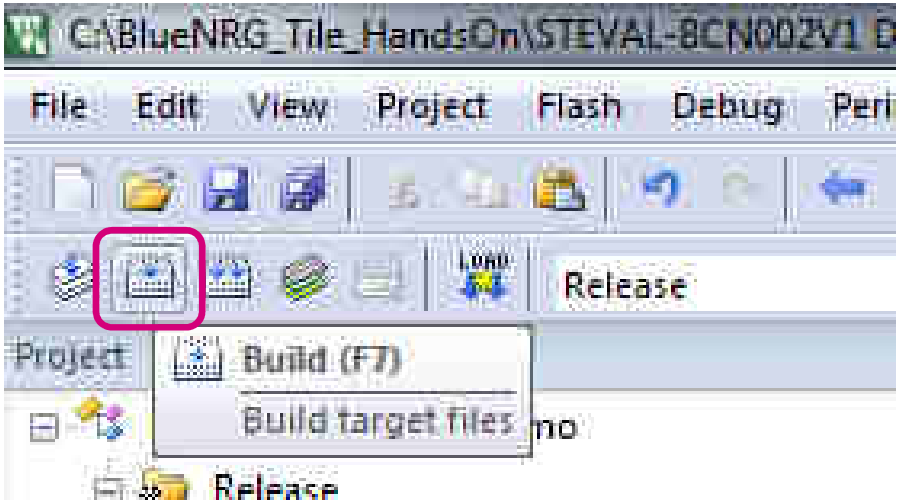
sensor.c

```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                          /* Length of AD type Transmission Power */
415   0x0A, 0x00,                 /* Transmission Power = 0 dBm */
416   8,                          /* Length of AD type Complete Local Name */
417   0x09,                       /* AD type Complete Local Name */
418   NAME_ALLMEMS,               /* Local Name */
419   13,                         /* Length of AD type Manufacturer info */
420   0xFF,                       /* AD type Manufacturer info */
421   0x01,                       /* Protocol version */
422   0x05,                       /* Device ID: 0x05 = STEVAL-BCN002V1 Board */
423   0x22,                       /* Feature Mask byte#1  LAB3 0x20 (LED) / LAB5
424   0xFE,                       /* Feature Mask byte#2  LAB4 0x9E (Acc+Press+Hu
425   0x05,                       /* Feature Mask byte#3  LAB4 0x04 (AccEvents) /
426   0x40,                       /* Feature Mask byte#4  LAB5 0x40 (eCompass) */
427   0x00,                       /* BLE MAC start */
```

life.augmented

Build the new code

1. **Click** on the **Build button** (top left corner) or hit **F7** on your keyboard
2. In the **Build Output** window (bottom) **wait** for the build to be completed.
   - **BLE_SensorDemo.bin** created
   - "**0 Error(s), 0 Warning(s)**" message appear

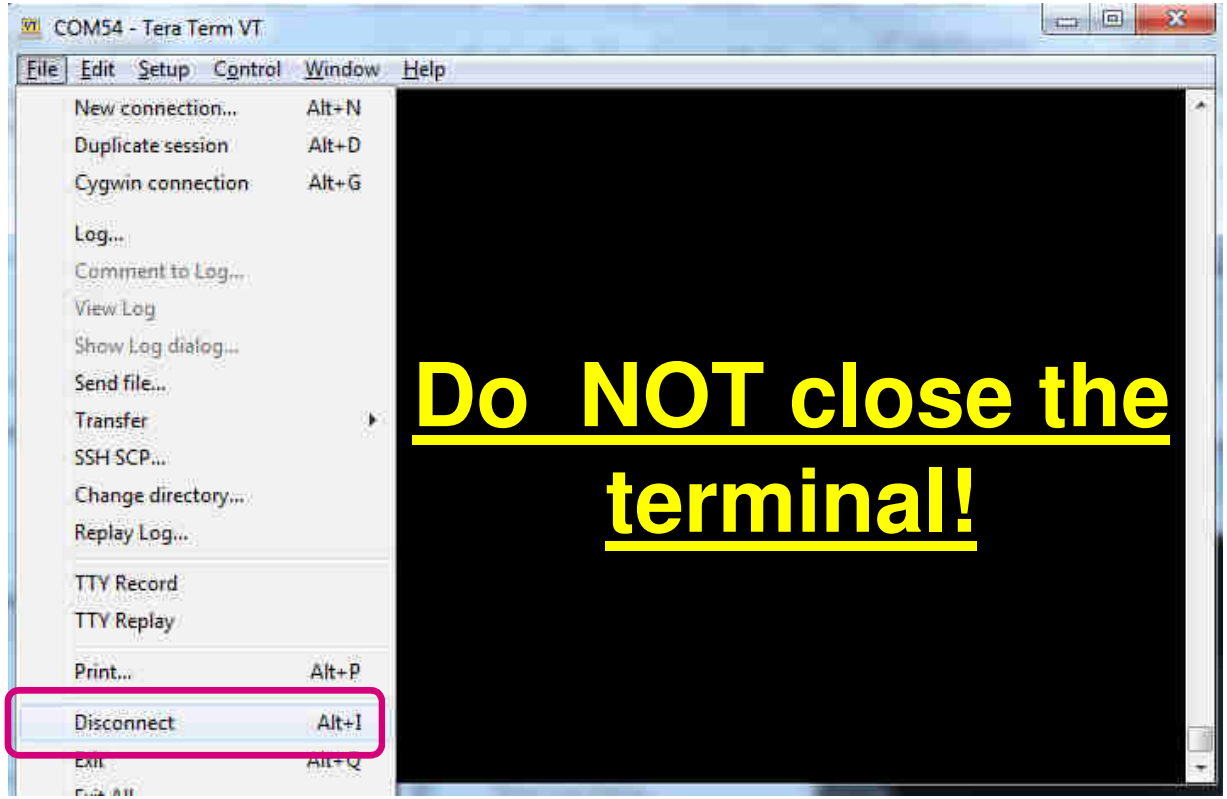



```
Build Output
compiling vl5311_wait.c...
compiling vl5311_register_funcs.c...
compiling vl5311_platform.c...
linking...
Program Size: Code=121908 RO-data=1428 RW-data=1136 ZI-data=21252
FromELF: creating hex file...
After Build - User command #1: fromelf.exe --bin ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf --output ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.bin
"..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:12
```

life.augmented

Disconnect the serial terminal

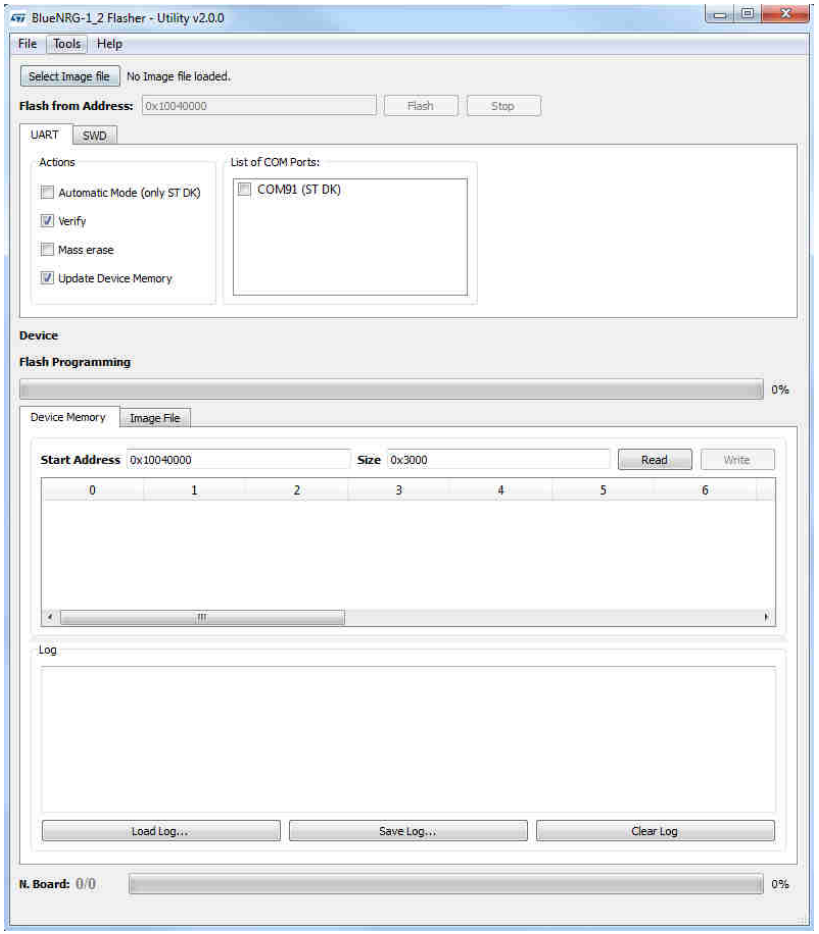
1. **Go back** to **Tera Term**
2. **Click** on the **File->Disconnect**



life.augmented

Re-flash the BlueNRG-2

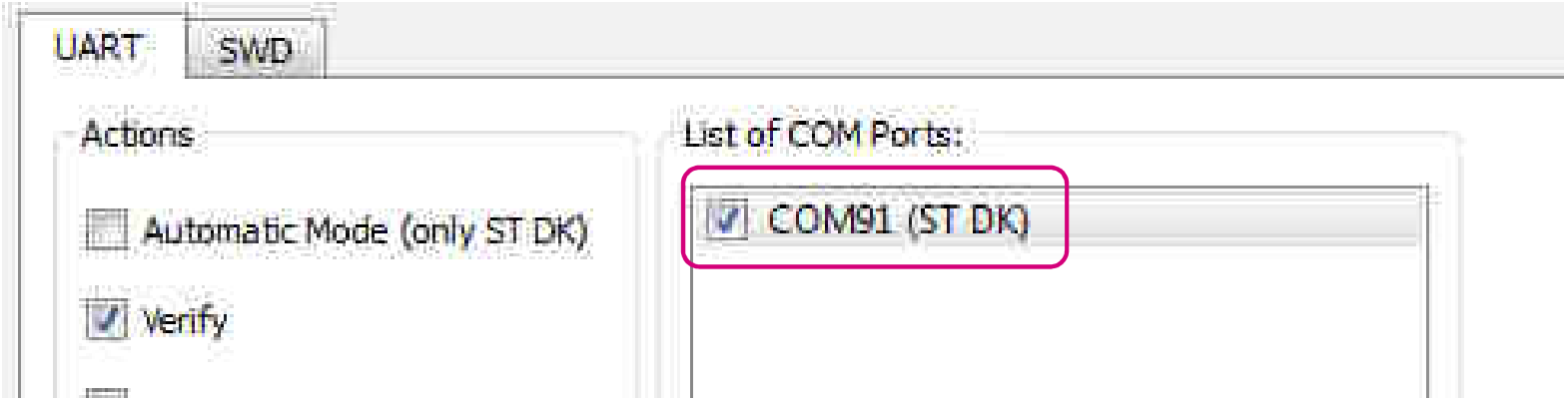
1. **Go back** to the **BlueNRG-2 Flasher Utility**



life.augmented

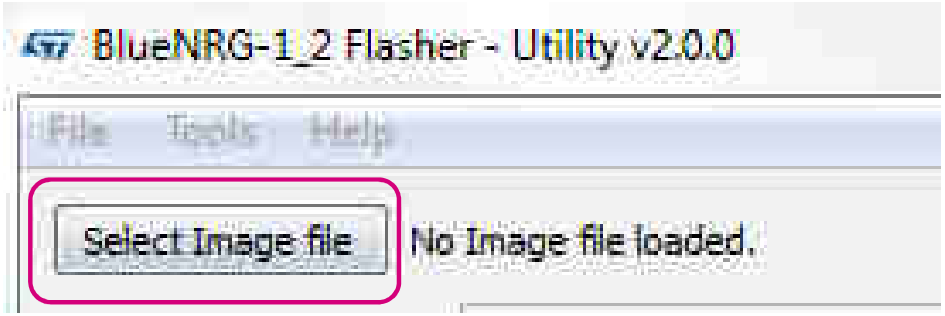
Flash the BlueNRG-2 1/4

1. **Select** the COM port labeled **(ST DK)**



Flash the BlueNRG-2 2/4

1. **Click** on the **Select Image file** button

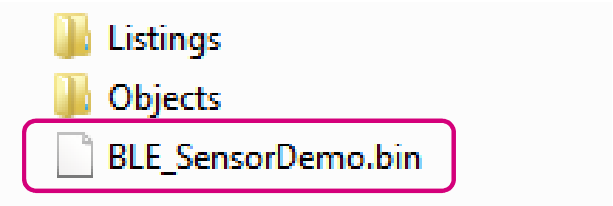


2. And **browse** the following **path**

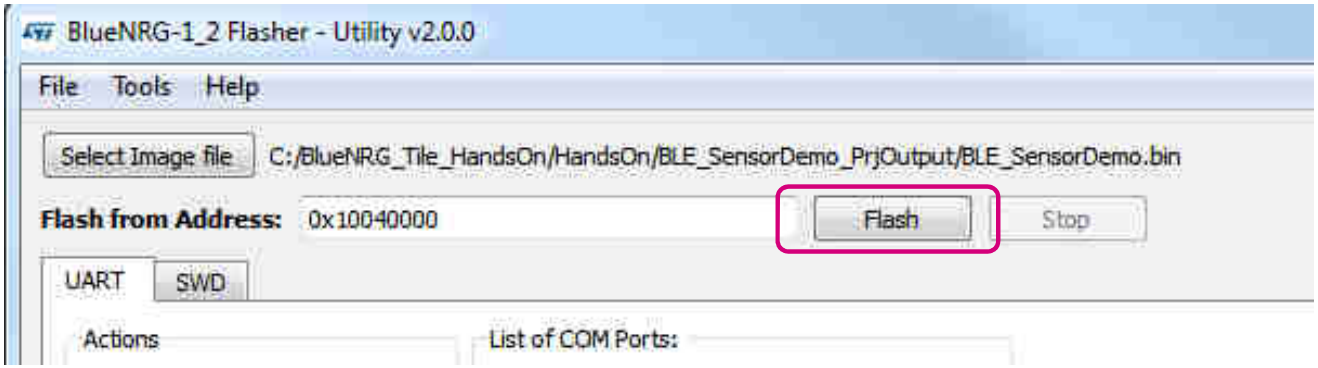
life.augmented

Flash the BlueNRG-2 3/4

1. **Select BLE_SensorDemo.bin and click on Open**



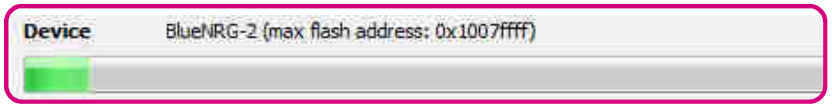
2. **Click on the Flash button**



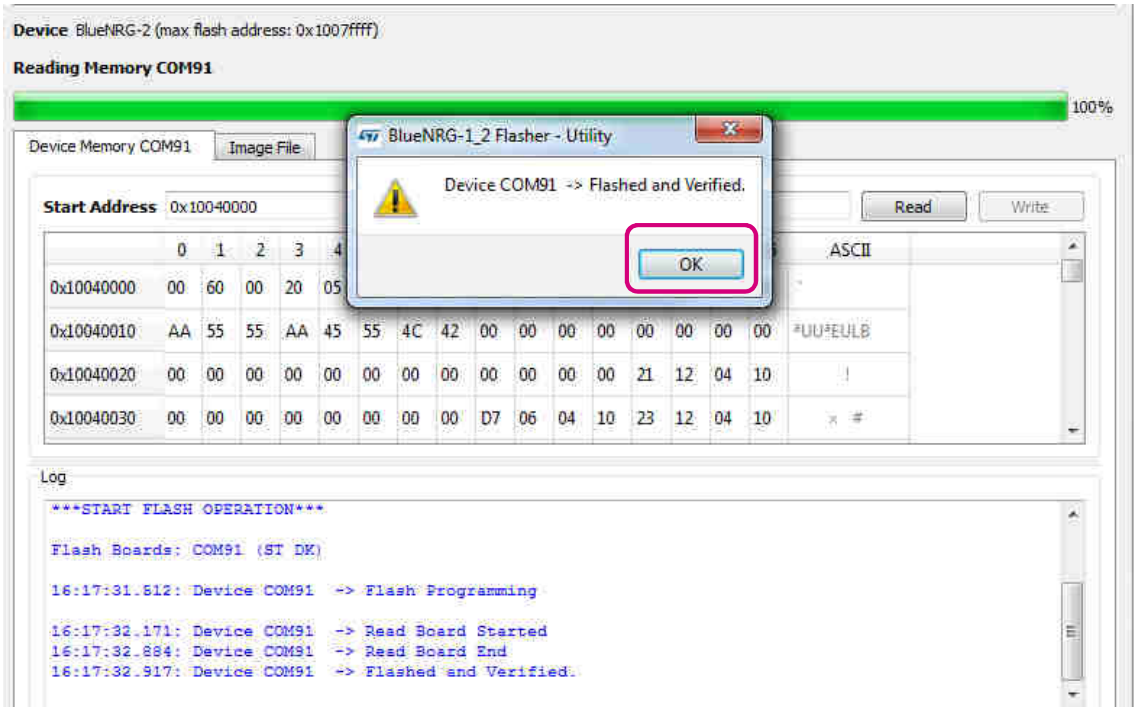
life.augmented

Flash the BlueNRG-2 4/4

1. Flashing starts: **green bar** proceeding



2. **Wait** for the **pop-up** window and **click** on **OK**

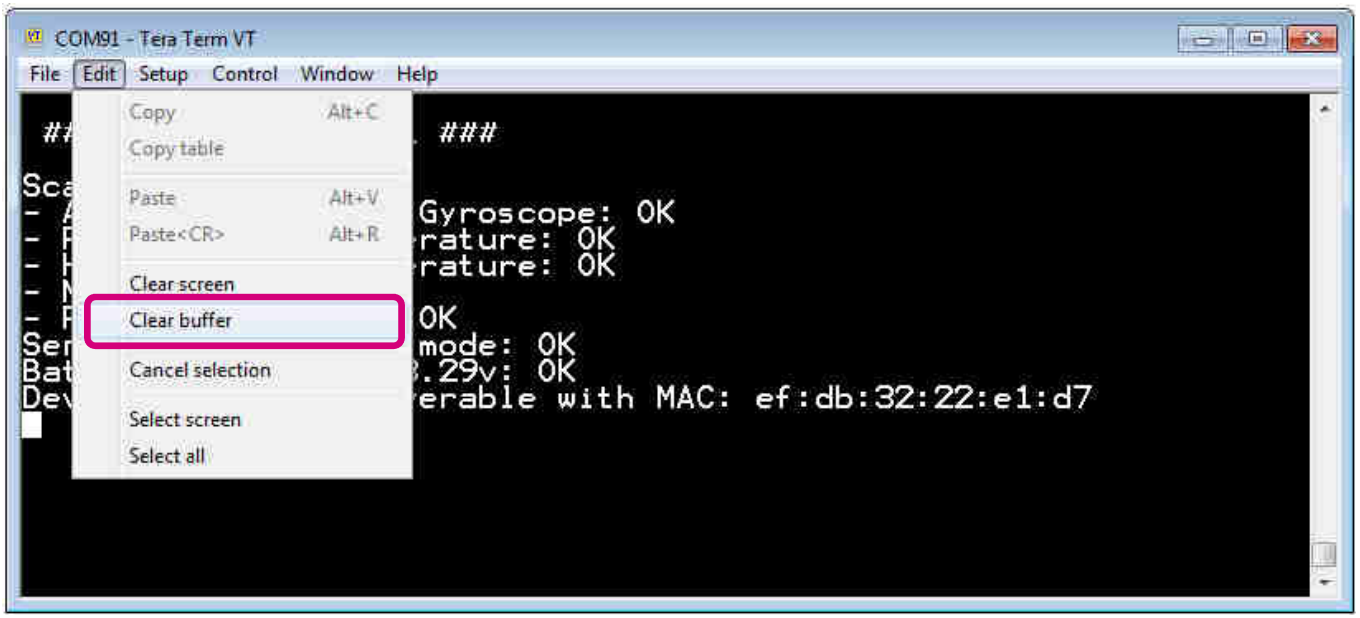


Do NOT close the Utility!

life.augmented

Clean Buffer in the serial terminal

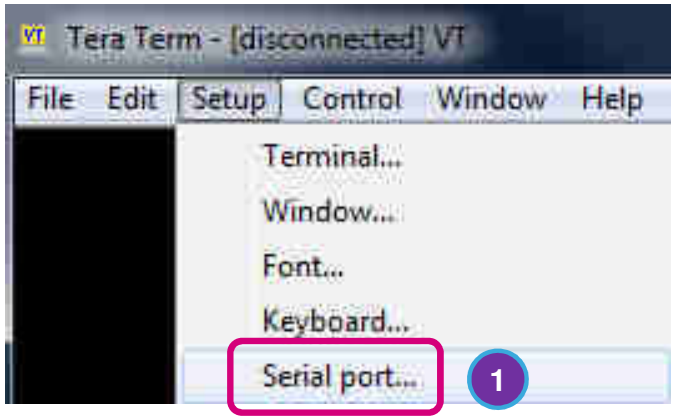
1. In **Tera Term** in order to have the terminal clean, **go to Edit -> Clear buffer**



life.augmented

Reconnect the serial terminal

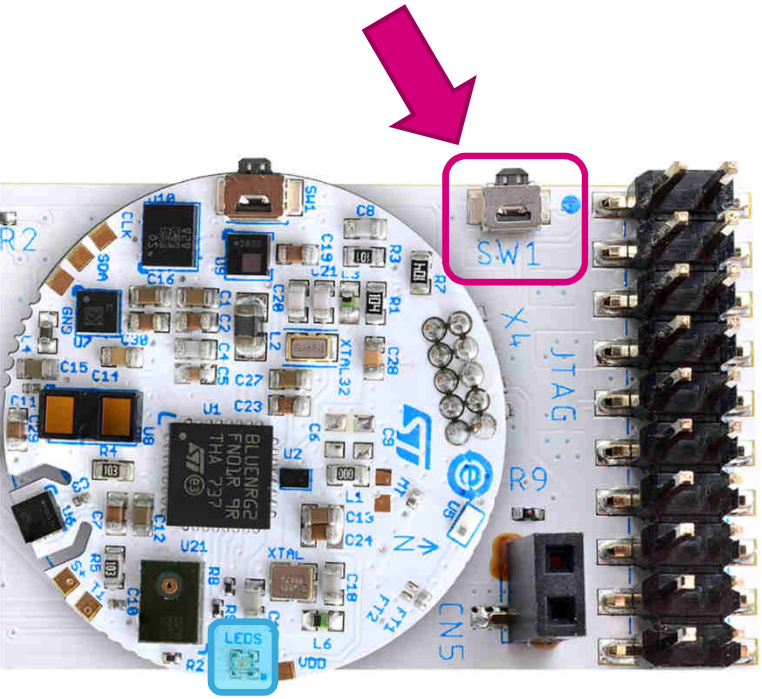
1. **Click Setup -> Serial port...**
2. Serial port should be already configured.
   Just need to **click** on **OK**

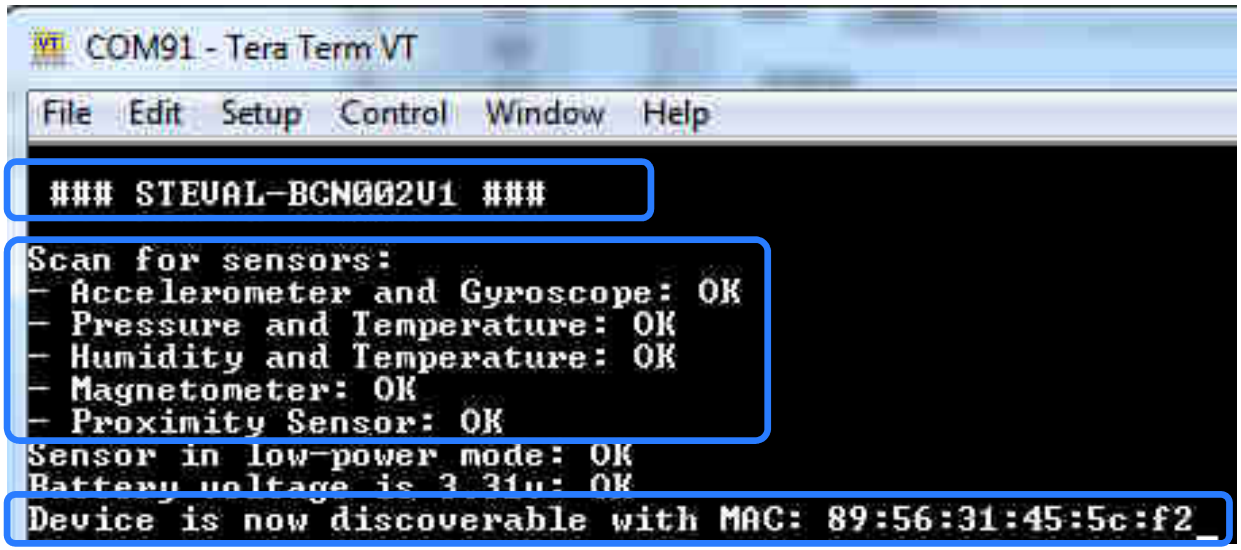


life.augmented

Sanity Check on serial port

Push SW1 button on the daughterboard -> **LED blinking Blue**

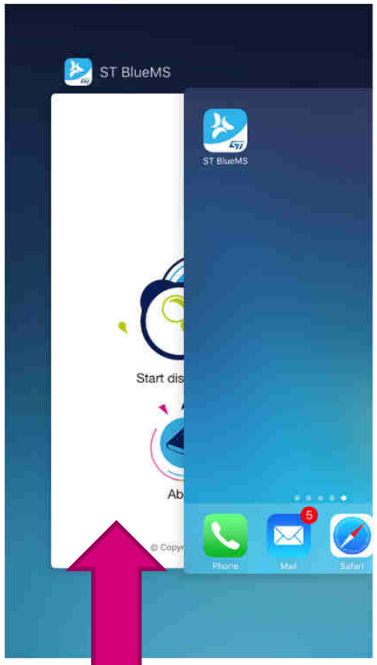




life.augmented

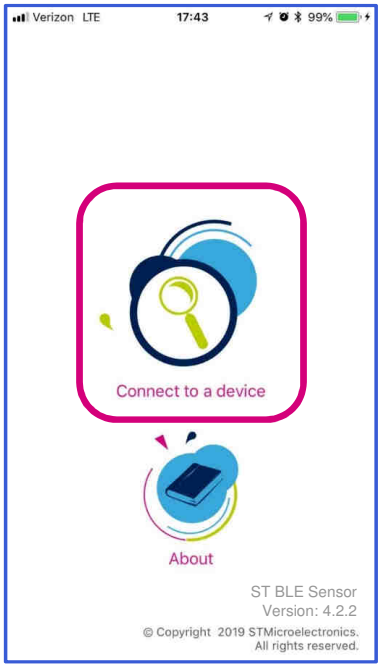
Connect using the ST BLE Sensor App

1 **"Kill" the BLE Sensor app**

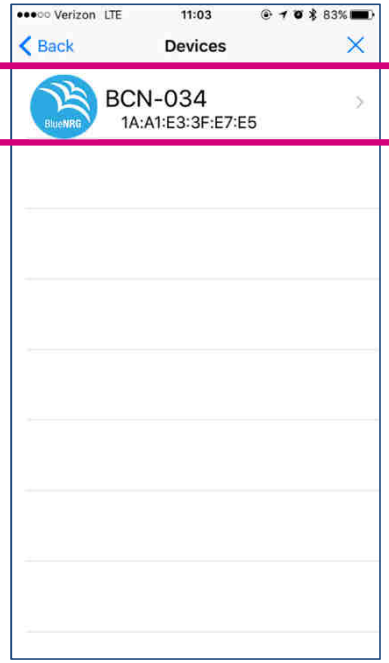


Home button and then swipe to top

2 **Tap "Connect to a device"**



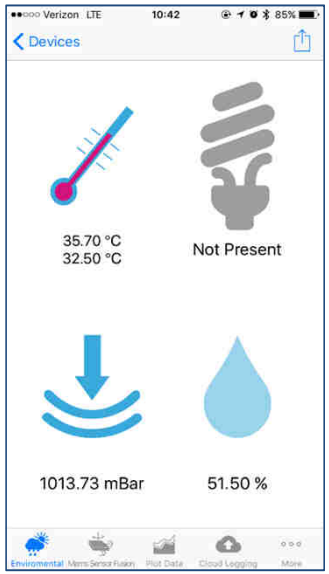
3 **Select YOUR STEVAL-BCN002V1**



life.augmented

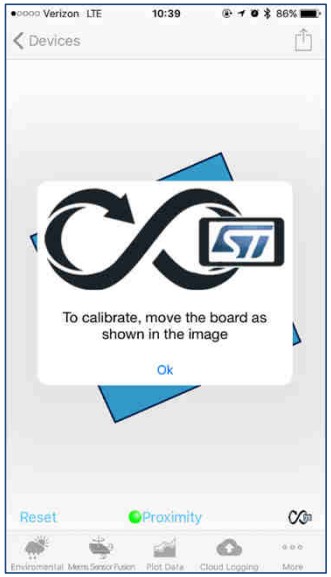
Sensor Data Fusion

Swipe left to **view** the **Mems Sensor Fusion**



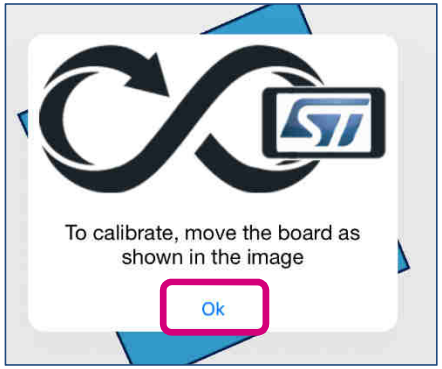
1

"Mems Sensor Fusion" tab



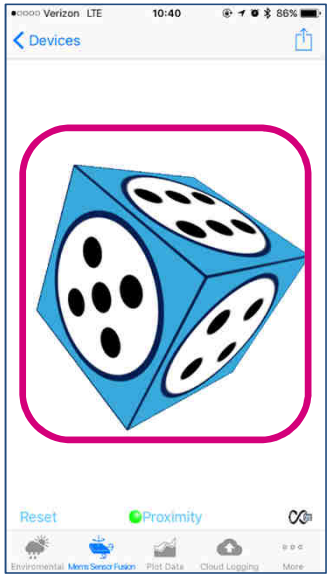
2

Click on **OK**



3

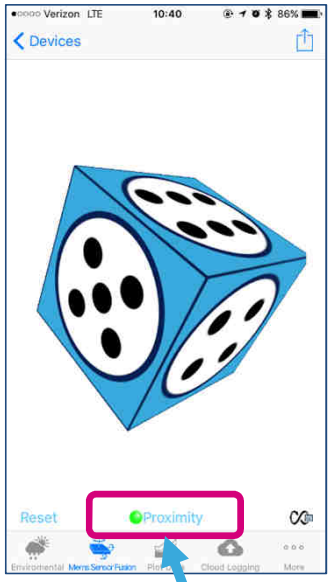
Sensor Fusion **enabled**



life.augmented

Proximity Sensor

Enable Proximity Sensor



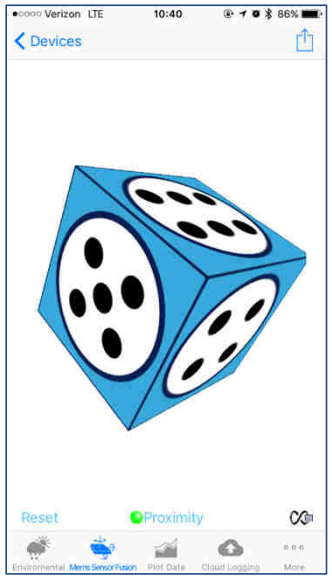
ON Proximity

OFF Proximity

Move your hand far and close to the BlueNRG-Tile

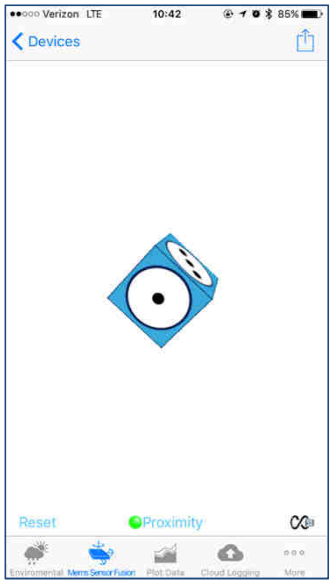
1

Far from BlueNRG-Tile



2

Near to BlueNRG-Tile



Note: ONLY for rendering purposes, the cube is scaled proportional to a clipped 30 cm distance

life.augmented

Proximity Sensor - VL53L1X

3rd gen **ToF sensor** with lens for **long distance** ranging & ROI selection

Product highlights

OLGA: **4.9 x 2.5 x 1.56 mm**
FoV : 27°
Compatible footprint with VL53L0X

Enhanced performances:

- Full FoV ranging : **400cm+** (white target, no IR)
- **Programmable FOV:** SPAD zone selection – full-screen to 4x4

Cutting-edge module and silicon :

- **Fastest miniature ToF product in the market** up to 50Hz
- **Integrated lens** for longer range and better ambient light immunity
- **Programmable settings** to best fit customer's application:
  - Low power with interrupts for user / object detection
  - Long distance ranging
  - High accuracy for small movement detection

Applications

Presence user detection

- Autonomous mode with interrupts
- Low-power
- Long distance 400cm+
- PC, tablets, IoT, portable handsets, security

Obstacle detection:

- Robots: Obstacle avoidance
- Vacuum cleaners: Wall following, cliff detection
- Drones: Take-off and landing, Ceiling detection

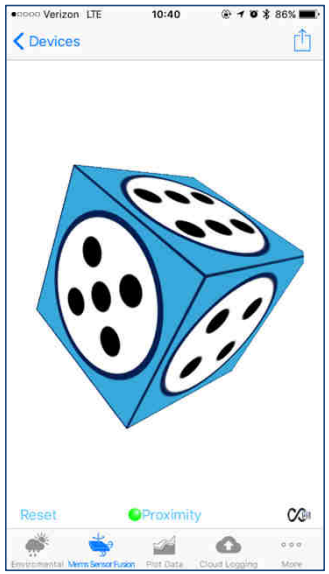
Accurate objects distance scanning

- Vending machines: control of objects in racks
- Coins dispensers: coins counting
- Smart shelves: Consumer scanning

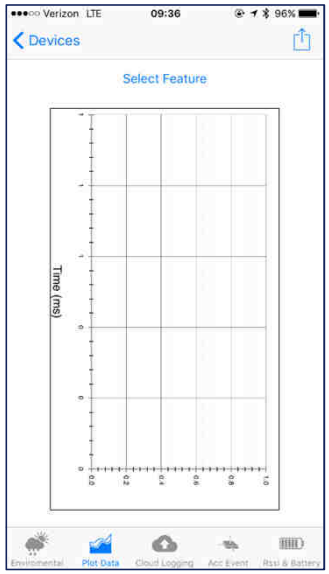
life.augmented

Full-range Proximity Real-time Data Plot

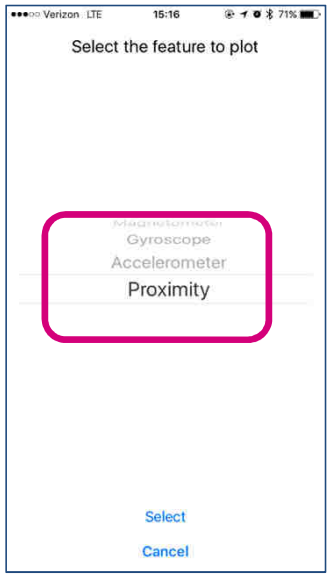
1 **Swipe left** to **view** the real-time data plot



2 **"Plot Feature"** tab

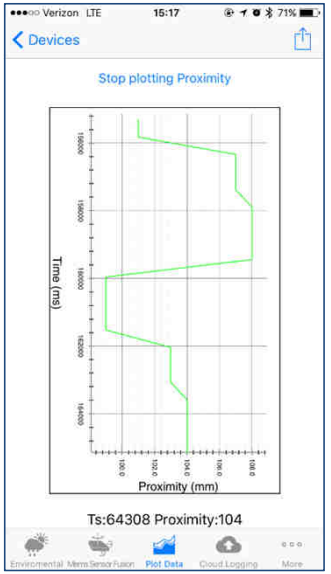


3 **Select Proximity**



4 **Select** the **time frame**

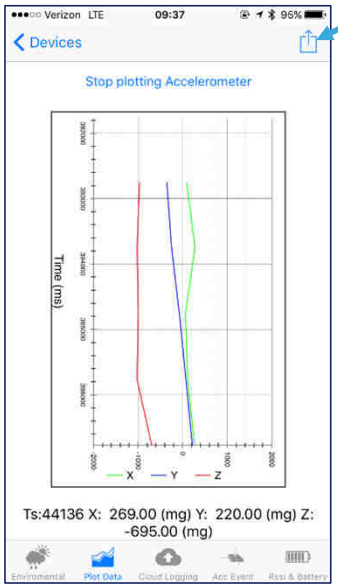
5 **View** the **real-time data plot**



life.augmented

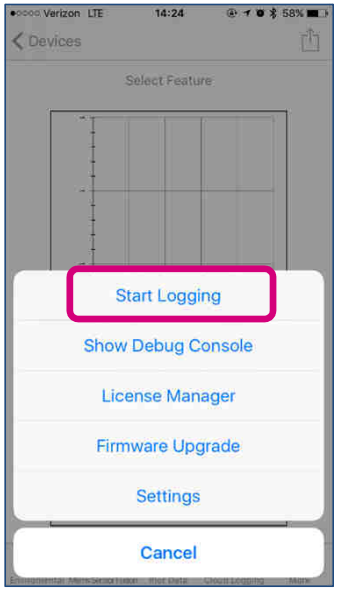
Logging Real-time Data Plot

Click on top right corner



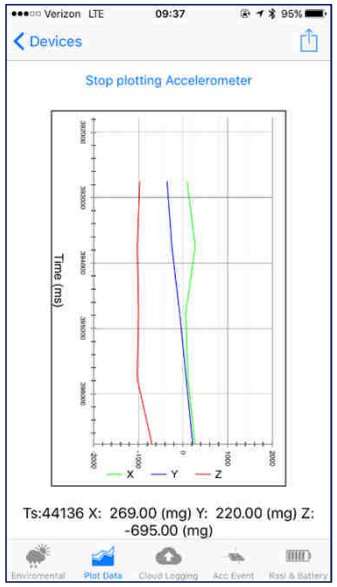
1

Select "**Start logging**"



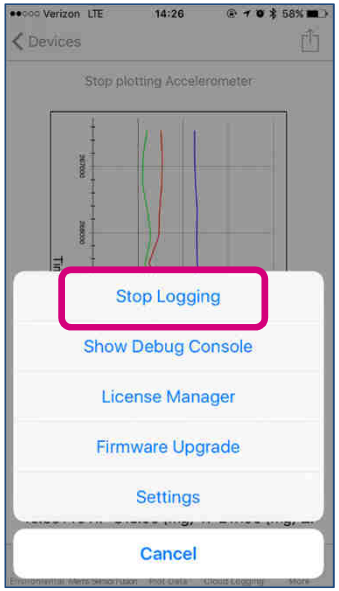
2

After some time **re-click** on the right-top corner



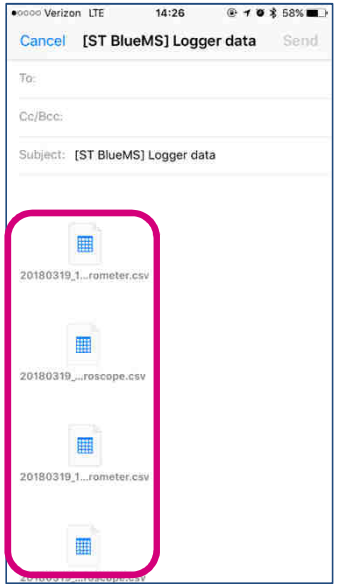
3

Select "**Stop logging**"



4

App opens your email client

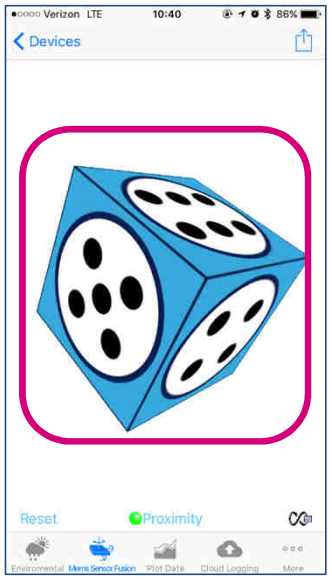


.csv files attached to the email

life.augmented

Sensor Data Fusion

Sensor Fusion **enabled**



Try **static** position: no drift, the cube is perfectly still

- In steady static position the **acc** and **mag** enable an accurate orientation estimate
- acc vibrations and mag bias and interferences are **rejected or compensated by the motionFX library**

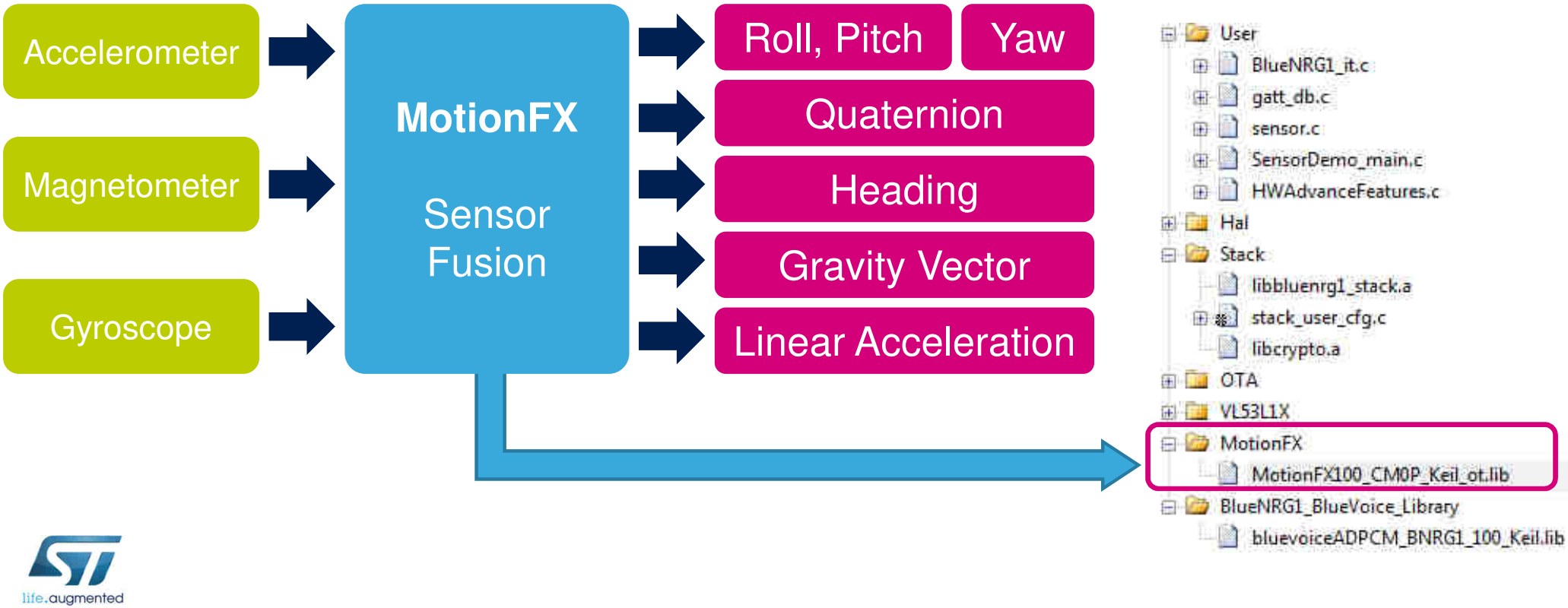
Try highly **dynamic** motion: perfect tracking

- In high dynamic motion the acc cannot be used, therefore the **gyro** is used to update the orientation
- gyro bias is **estimated and compensated at runtime** by the **motionFX** library

life.augmented

Sensor Data Fusion

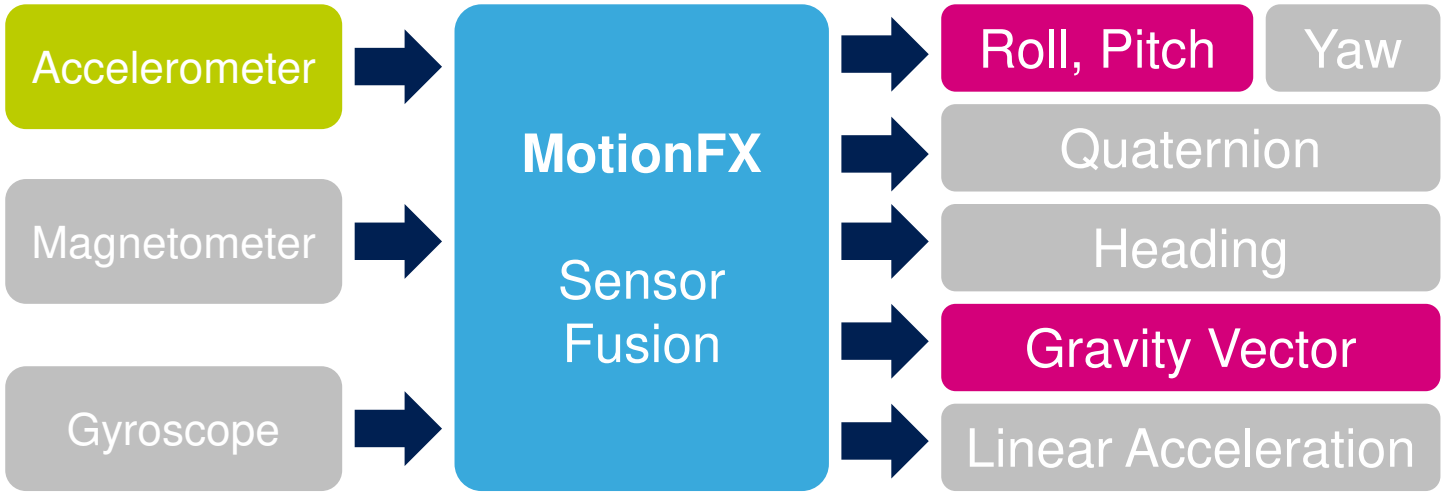
MotionFX library



life.augmented

Sensor Data Fusion

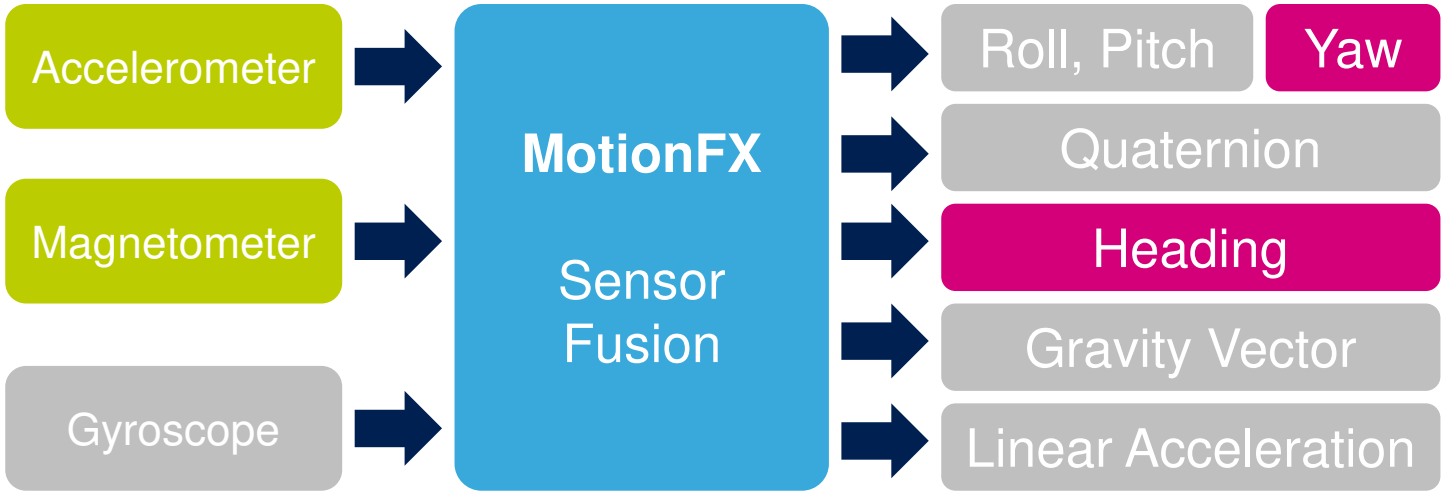
MotionFX library



- **Accelerometer gives roll** and **pitch angles** and the **gravity vector**…but **only in static conditions**! (or low dynamics)

Sensor Data Fusion

MotionFX library



- **Magnetometer gives yaw angle** and **heading**

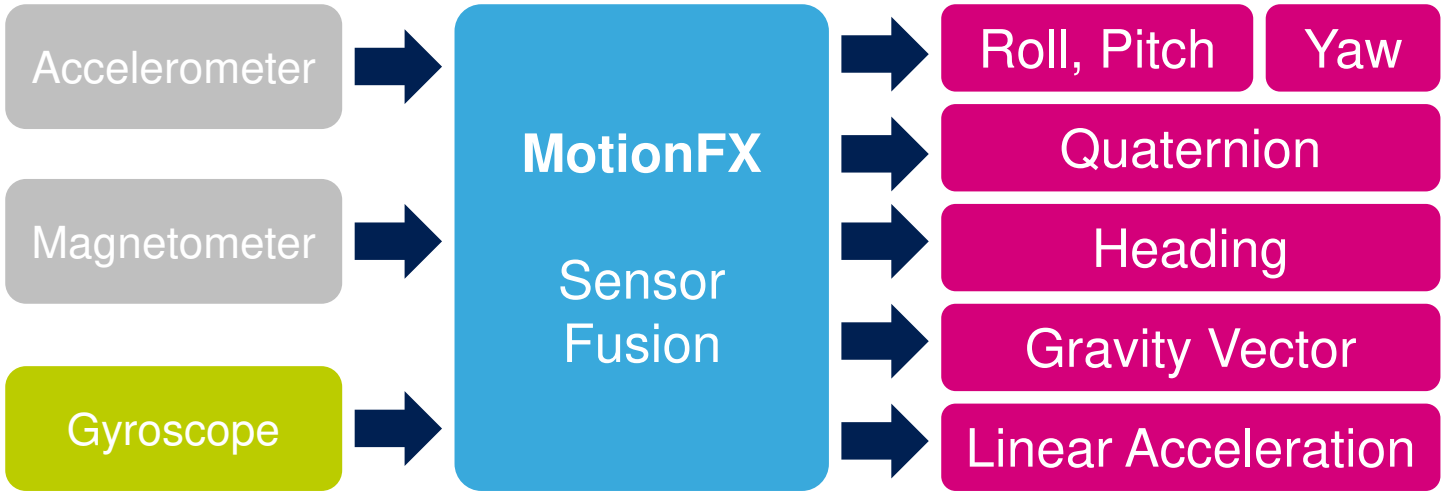
…but **only if tilt is compensated**: the accelerometer is needed!

…and **mag bias** (hard-iron) **must be compensated!**

life.augmented

Sensor Data Fusion

MotionFX library

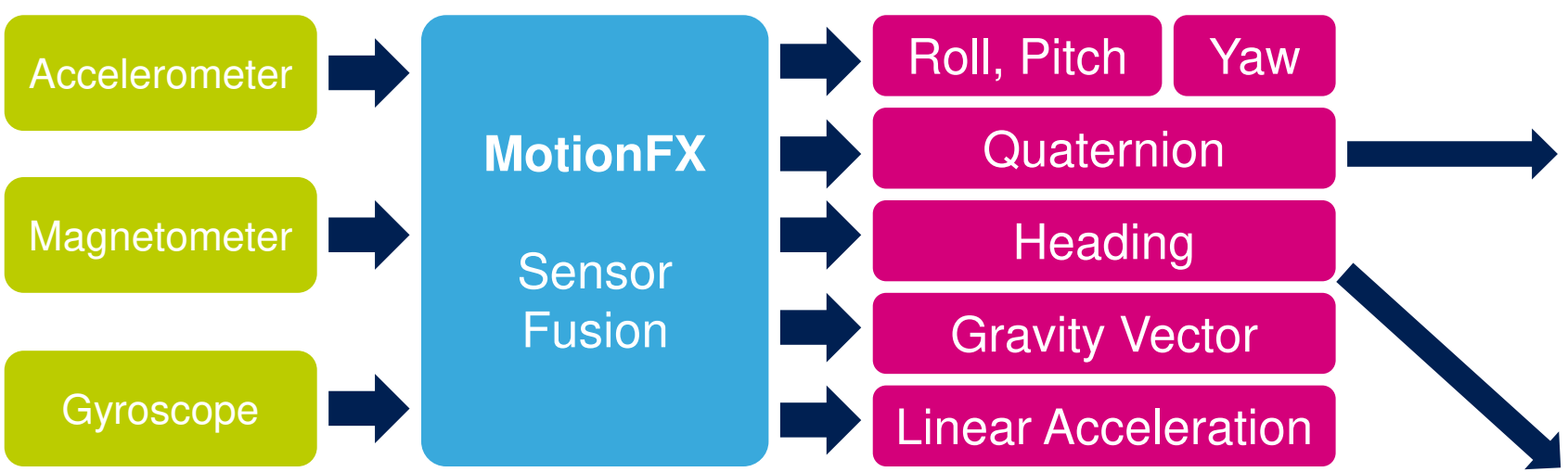


- **Gyroscope gives the new orientation** based on previous orientation

…but **gyro bias offset needs to be compensated** for accuracy!

Sensor Data Fusion

MotionFX library



Motion FX library provides orientation estimation plus:

- **Magnetometer hard-iron offset** compensation.
- **Accelerometer vibrations** rejection.
- **Gyroscope bias** offset compensation.

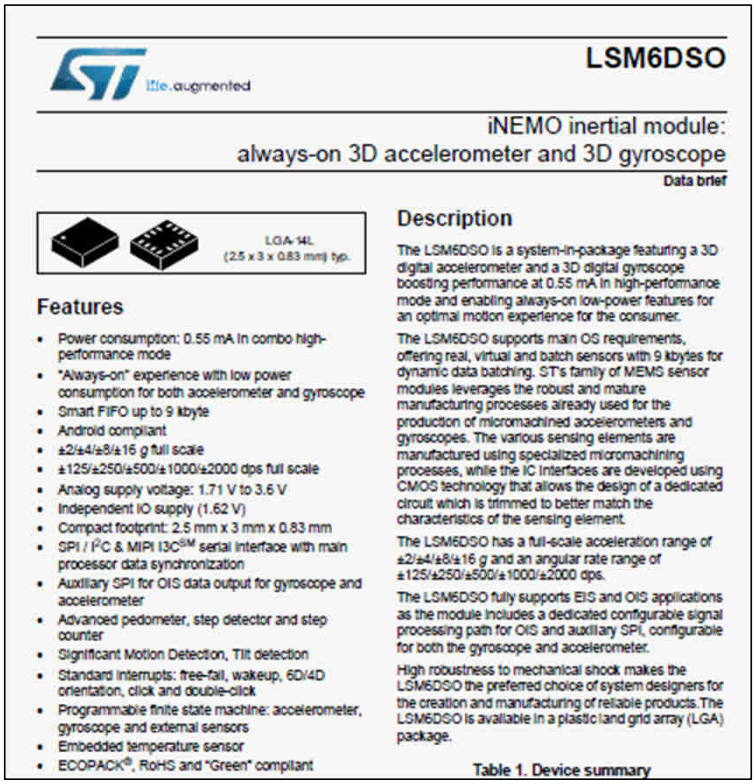
life.augmented

Gyroscope

The gyroscope is the most important sensor in the system
…works in highly dynamic conditions when the Acc cannot be used
…works with magnetic anomalies when the Mag cannot be used

LSM6DSO is an inertial module with **Acc** and **Gyro**

- **Exceptional noise performance** for maximum accuracy in orientation estimation
  - 90 ug/sqrtHz acc noise density
  - 3.8 mdps/sqrtHz gyro noise density
- **Extremely low-power**
  - **650 uA peak power** at ODR 6.66kS/sec with Acc+Gyro in high performance mode!



life.augmented

LSM6DSO

iNEMO inertial module:
always-on 3D accelerometer and 3D gyroscope

Data brief

LGA-14L
(2.5 x 3 x 0.83 mm) typ.

Features

- Power consumption: 0.55 mA in combo high-performance mode
- "Always-on" experience with low power consumption for both accelerometer and gyroscope
- Smart FIFO up to 9 kbyte
- Android compliant
- ±2/±4/±8/±16 *g* full scale
- ±125/±250/±500/±1000/±2000 dps full scale
- Analog supply voltage: 1.71 V to 3.6 V
- Independent IO supply (1.62 V)
- Compact footprint: 2.5 mm x 3 mm x 0.83 mm
- SPI / I²C & MIPI I3C℠ serial interface with main processor data synchronization
- Auxiliary SPI for OIS data output for gyroscope and accelerometer
- Advanced pedometer, step detector and step counter
- Significant Motion Detection, Tilt detection
- Standard interrupts: free-fall, wakeup, 6D/4D orientation, click and double-click
- Programmable finite state machine: accelerometer, gyroscope and external sensors
- Embedded temperature sensor
- ECOPACK®, RoHS and "Green" compliant

Description

The LSM6DSO is a system-in-package featuring a 3D digital accelerometer and a 3D digital gyroscope boosting performance at 0.55 mA in high-performance mode and enabling always-on low-power features for an optimal motion experience for the consumer.

The LSM6DSO supports main OS requirements, offering real, virtual and batch sensors with 9 kbytes for dynamic data batching. ST's family of MEMS sensor modules leverages the robust and mature manufacturing processes already used for the production of micromachined accelerometers and gyroscopes. The various sensing elements are manufactured using specialized micromachining processes, while the IC interfaces are developed using CMOS technology that allows the design of a dedicated circuit which is trimmed to better match the characteristics of the sensing element.

The LSM6DSO has a full-scale acceleration range of ±2/±4/±8/±16 *g* and an angular rate range of ±125/±250/±500/±1000/±2000 dps.

The LSM6DSO fully supports EIS and OIS applications as the module includes a dedicated configurable signal processing path for OIS and auxiliary SPI, configurable for both the gyroscope and accelerometer.

High robustness to mechanical shock makes the LSM6DSO the preferred choice of system designers for the creation and manufacturing of reliable products. The LSM6DSO is available in a plastic land grid array (LGA) package.

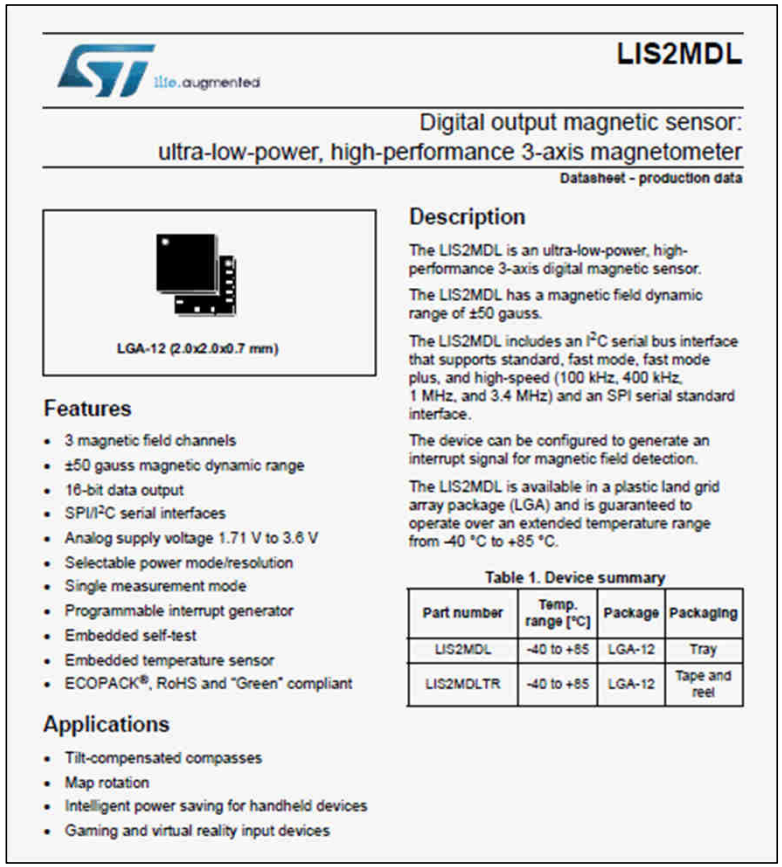
Table 1. Device summary

life.augmented

Magnetometer

LIS2MDL is **digital Mag**

- **Exceptional noise performance and dynamic range** for maximum accuracy in orientation estimation
  - 50 Gauss dynamic range
  - 3 mgauss RMS noise
- **Embedded offset compensation**
  - **Intrinsic offset is estimated and compensated automatically**
  - Extrinsic offset (hard-iron) must be estimated by host and can be compensated internally



life.augmented

LIS2MDL

Digital output magnetic sensor:
ultra-low-power, high-performance 3-axis magnetometer

Datasheet - production data

LGA-12 (2.0x2.0x0.7 mm)

Features

- 3 magnetic field channels
- ±50 gauss magnetic dynamic range
- 16-bit data output
- SPI/I²C serial interfaces
- Analog supply voltage 1.71 V to 3.6 V
- Selectable power mode/resolution
- Single measurement mode
- Programmable interrupt generator
- Embedded self-test
- Embedded temperature sensor
- ECOPACK®, RoHS and "Green" compliant

Applications

- Tilt-compensated compasses
- Map rotation
- Intelligent power saving for handheld devices
- Gaming and virtual reality input devices

Description

The LIS2MDL is an ultra-low-power, high-performance 3-axis digital magnetic sensor.

The LIS2MDL has a magnetic field dynamic range of ±50 gauss.

The LIS2MDL includes an I²C serial bus interface that supports standard, fast mode, fast mode plus, and high-speed (100 kHz, 400 kHz, 1 MHz, and 3.4 MHz) and an SPI serial standard interface.

The device can be configured to generate an interrupt signal for magnetic field detection.

The LIS2MDL is available in a plastic land grid array package (LGA) and is guaranteed to operate over an extended temperature range from -40 °C to +85 °C.

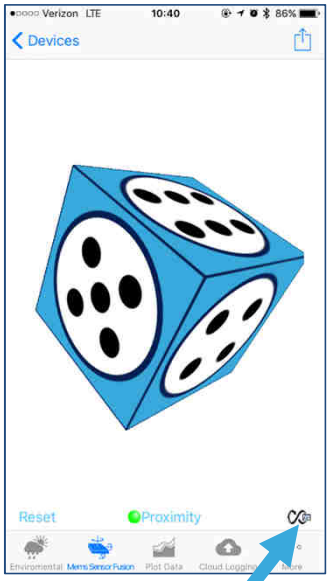
Table 1. Device summary

Part number	Temp. range [°C]	Package	Packaging
LIS2MDL	-40 to +85	LGA-12	Tray
LIS2MDLTR	-40 to +85	LGA-12	Tape and reel

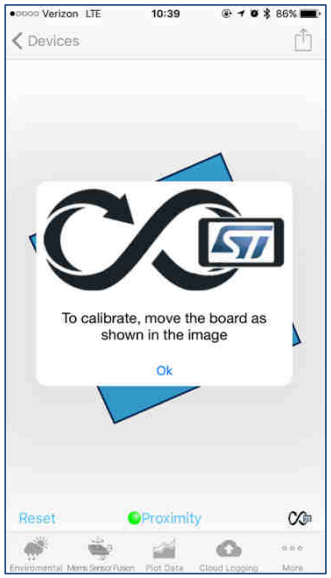
life.augmented

Magnetometer

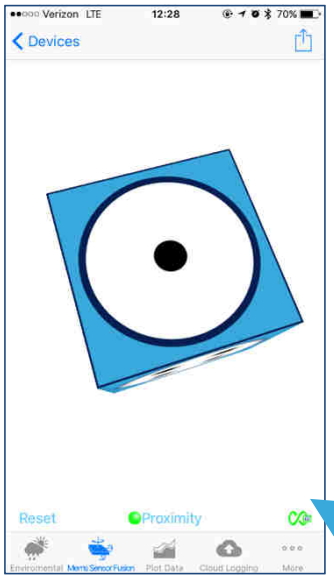
Calibration



Tap



Move the BlueNRG-Tile with the "**8 pattern**" shown in the figure to **calibrate the magnetometer**

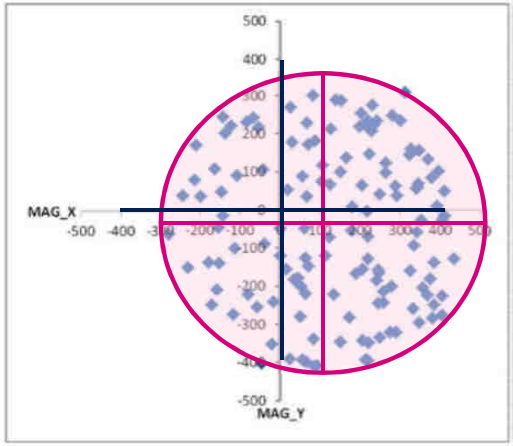


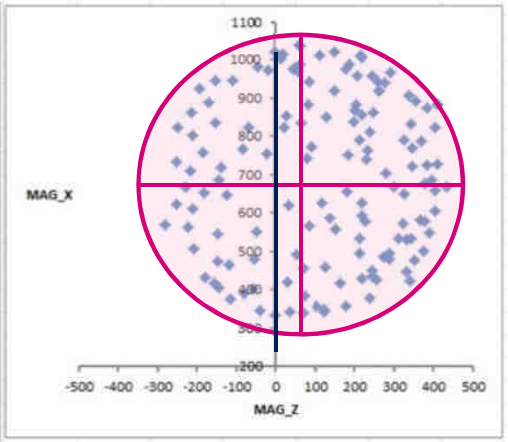
Calibration is **Completed** when the icon becomes **green**.

life.augmented

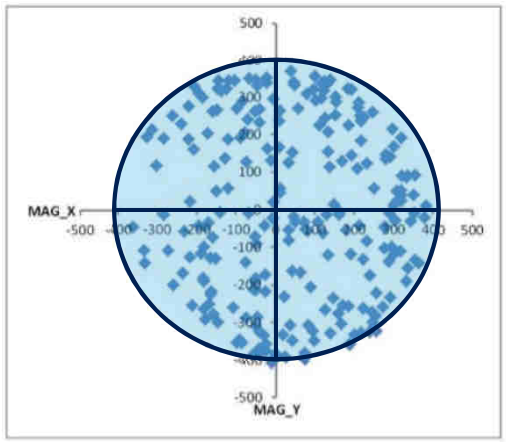
Magnetometer

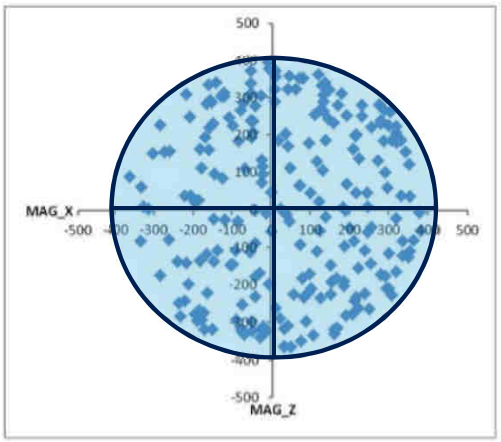
Calibration





Before calibration,
data not centered





After calibration,
data centered:
hard-iron offset
subtracted

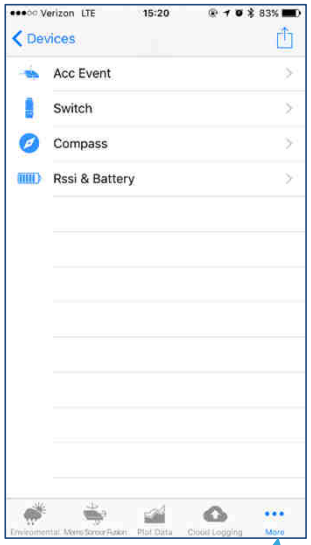
Each magnetometer has **intrinsic offset:**
this is compensated **internally**

Higher precision in the compensation!

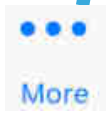
What if magnetometer is NOT calibrated?
Not accurate eCompass

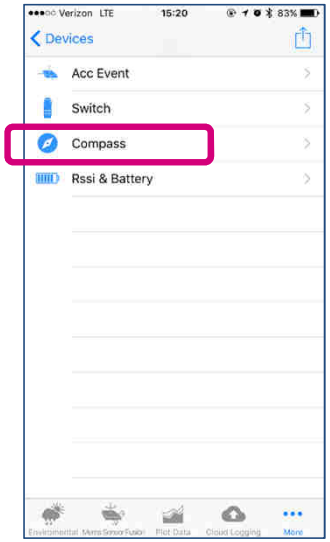
life.augmented

Magnetometer eCompass

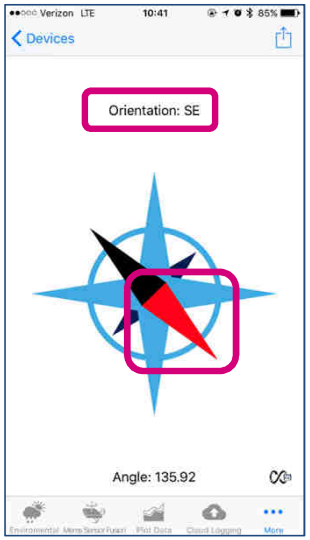


Tap



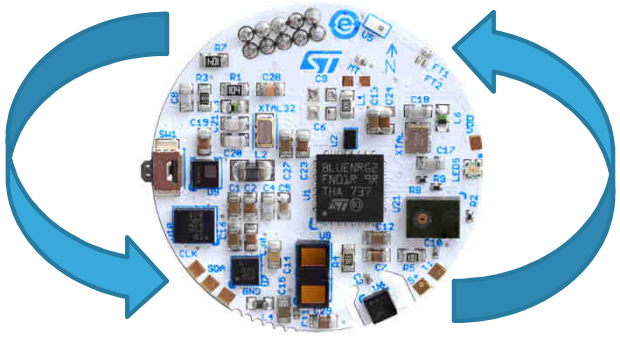


Select “Compass”



Red arrow is related to the current orientation of the BlueNRG-Tile

Rotate the BlueNRG-Tile

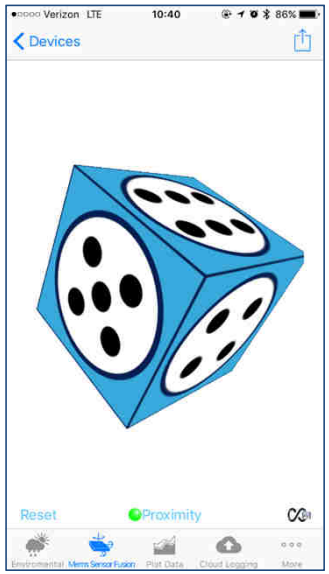


(you can check against the phone eCompass)

life.augmented

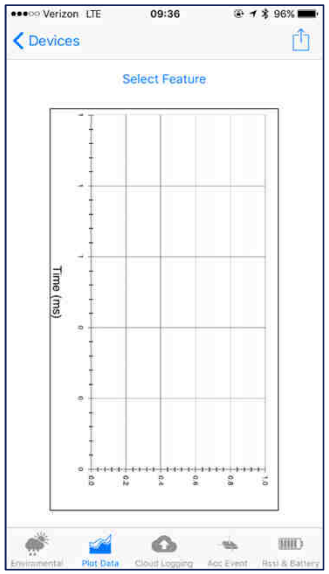
Real-time Data Plot

Swipe left to **view** the **real-time data plot**



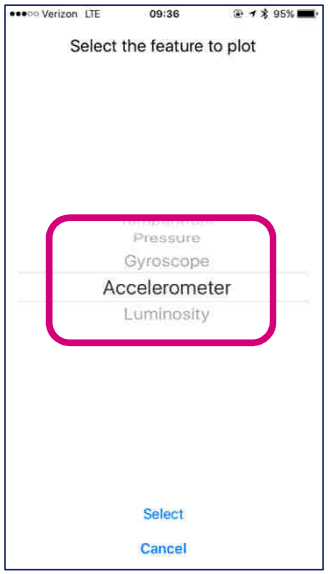
1

"Plot Data" Feature tab



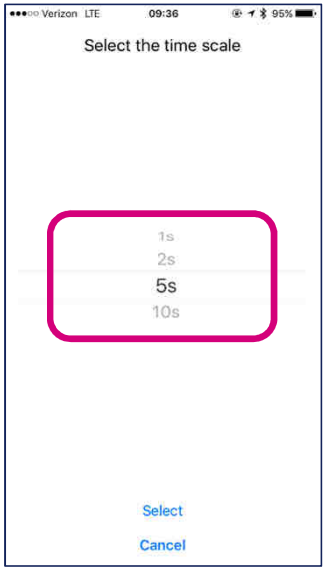
2

Select the **sensor data** to plot



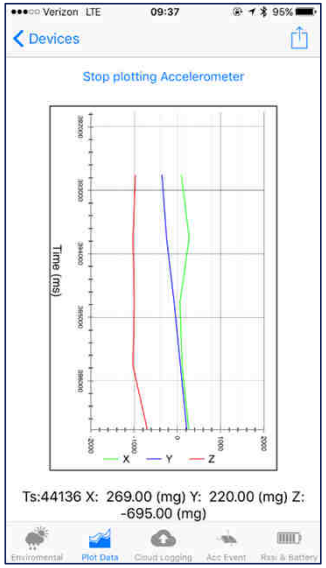
3

Select the **time frame**



4

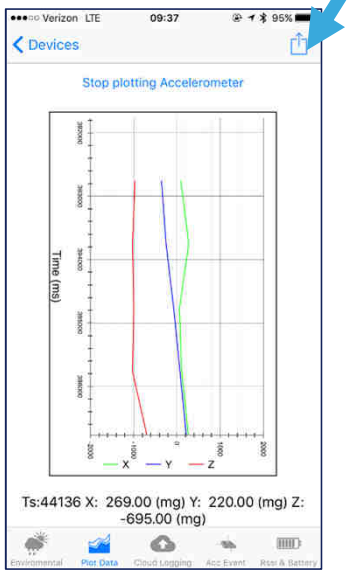
View the **real-time data plot**



life.augmented

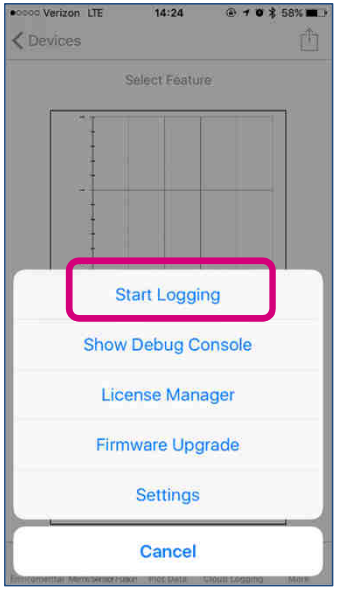
Logging Real-time Data Plot

Click on
top right corner



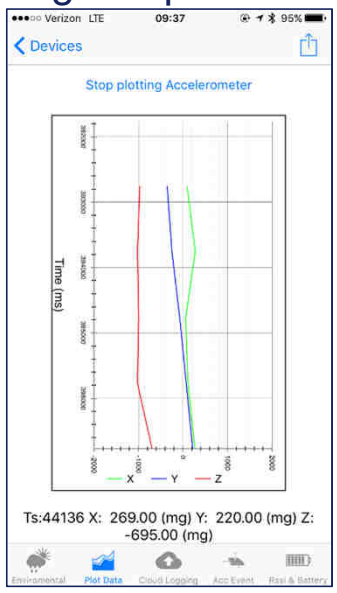
1

Select
"**Start logging**"



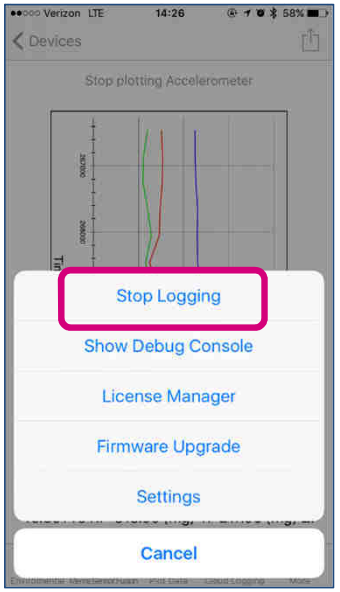
2

After some time
re-click on the
right-top corner



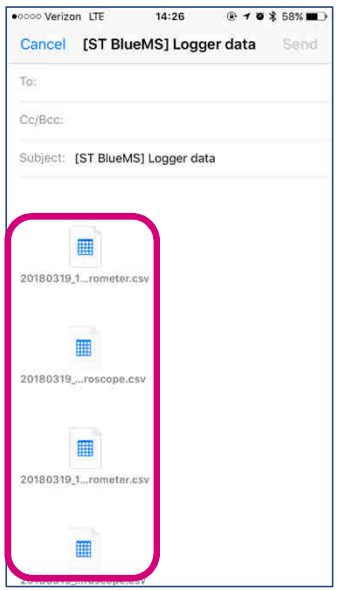
3

Select
"**stop logging**"



4

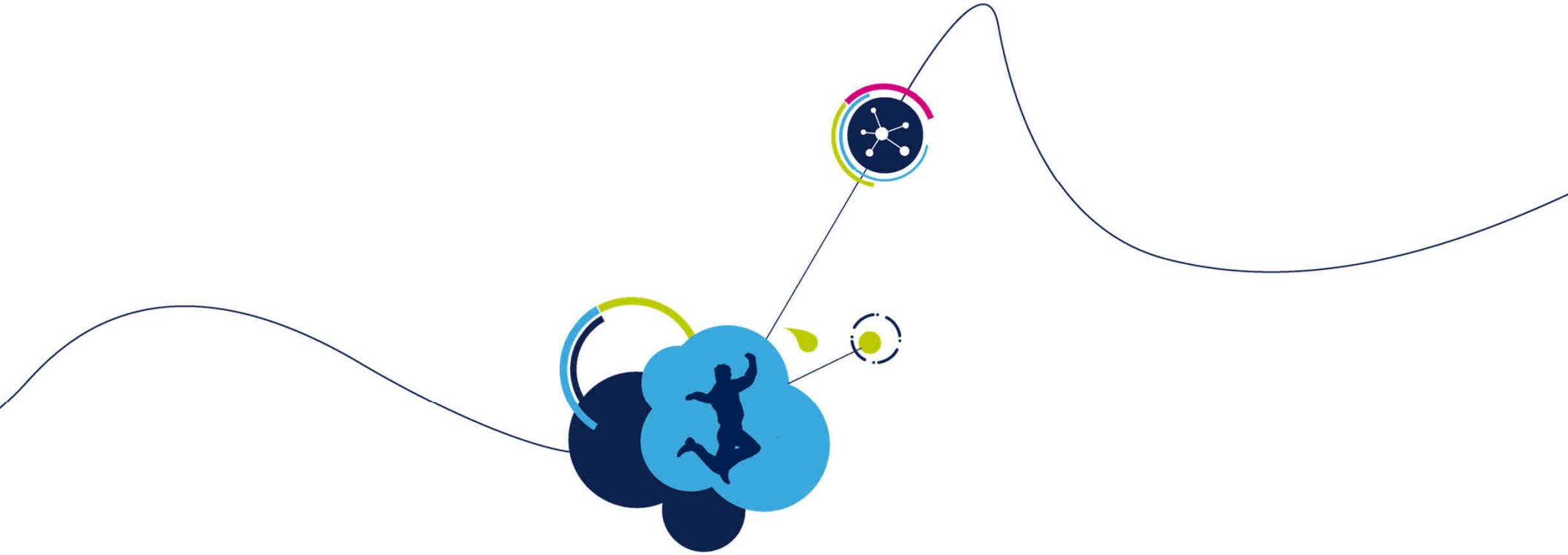
App opens your
email client



.csv files attached
to the email

life.augmented

Do NOT disconnect!
Just stay connected for the next lab…



Lab 6

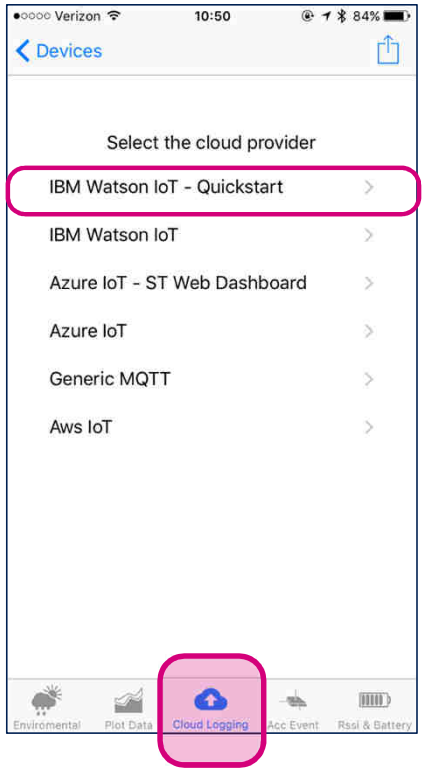
Cloud data logging on IBM Watson

life.augmented

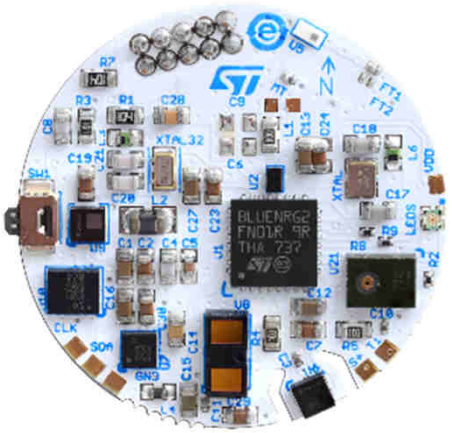
Cloud Logging

1. **Enable** IBM Watson IoT – Cloud Logging
2. **Send** BLE notification packets on Sensor status
3. **Visualize** the data

IBM Watson



Notification Packets



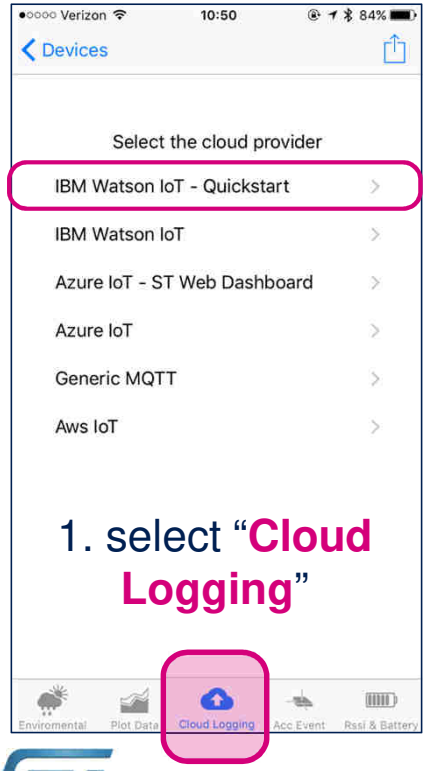
life.augmented

Code modifications

NO MODIFICATIONS NEEDED!

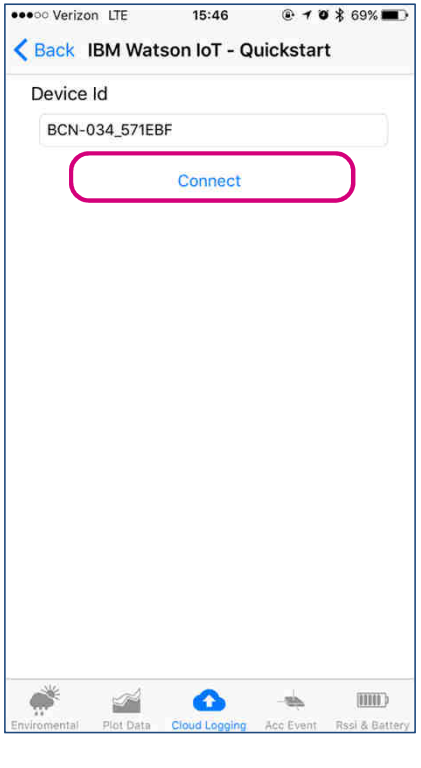
IBM Watson IoT Quickstart

2. **Select "IBM Watson IoT - Quickstart"**

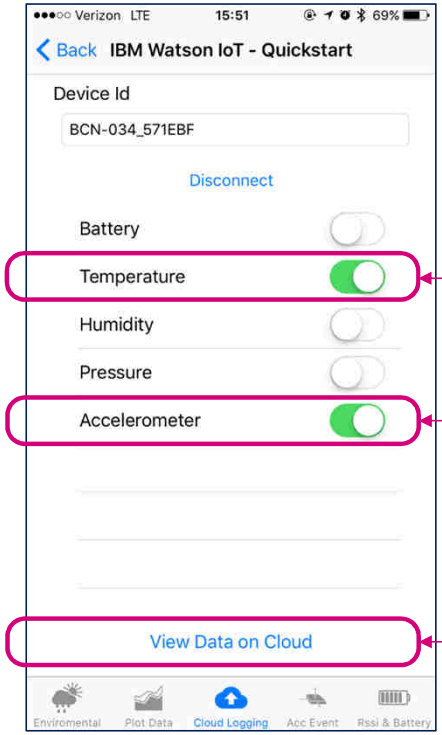


1. select "**Cloud Logging**"

Click "Connect"

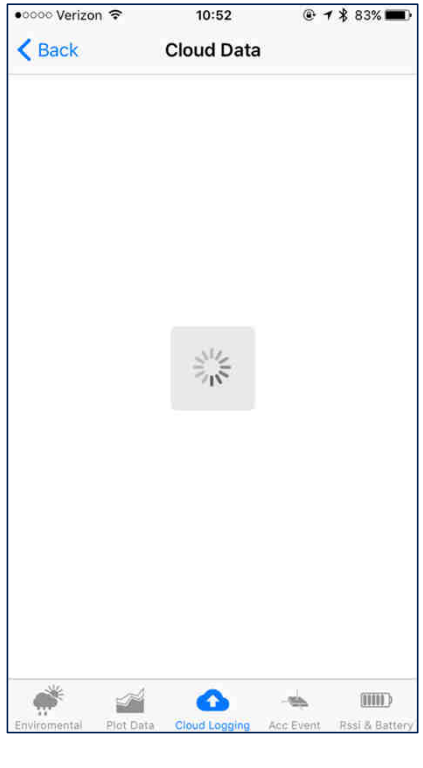


1. **Select** one or multiple features



2. **Click "View Data on Cloud"**

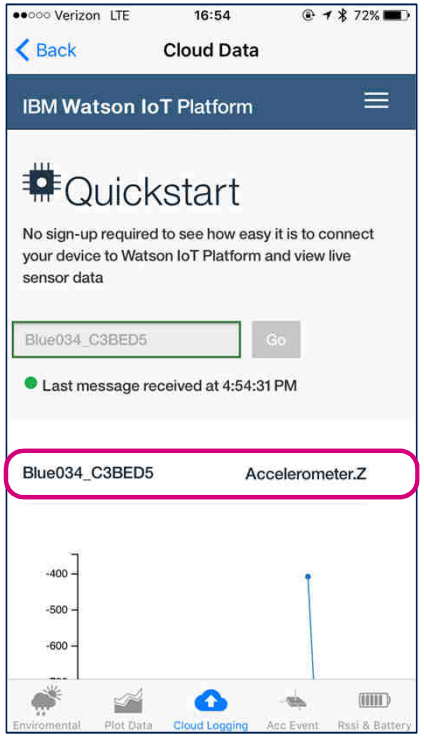
Wait a few seconds



life.augmented

IBM Watson IoT Quickstart

Quickstart will appear



Scroll down to see your selected sensor plot.

You will see the **Plot** of selected feature

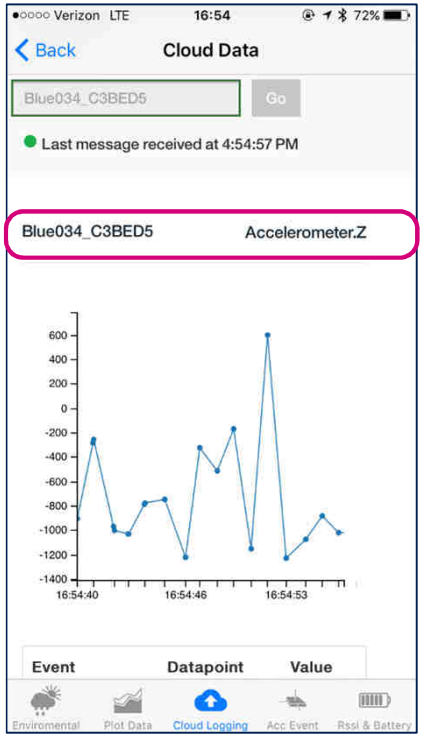
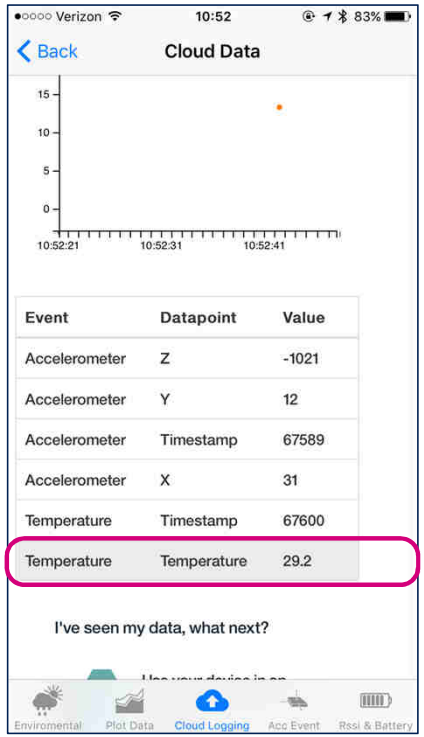
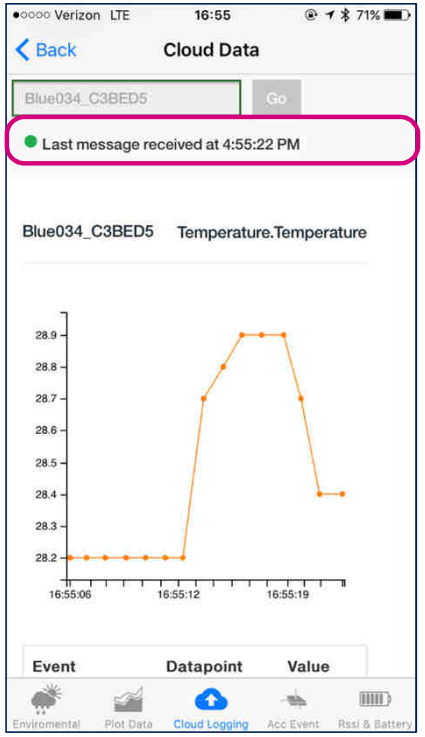


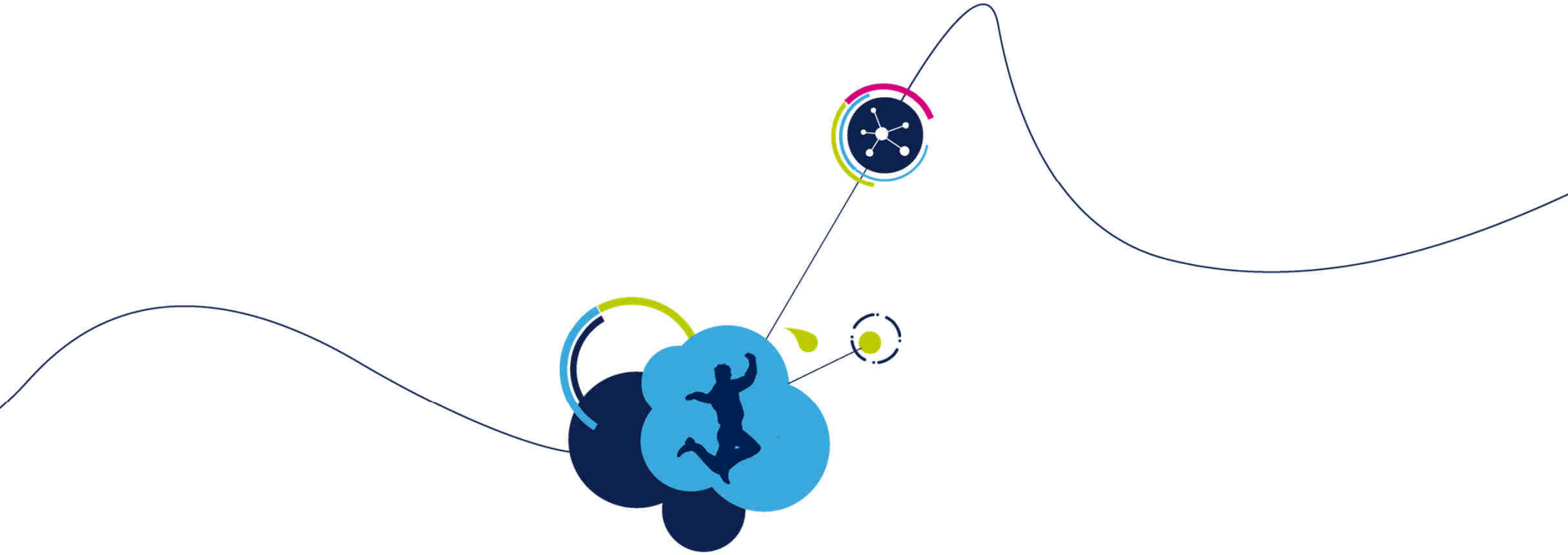
Table of **available features**





Scroll down again to change sensor data or axes

life.augmented



Lab 7

Bonus Lab – Voice over BLE

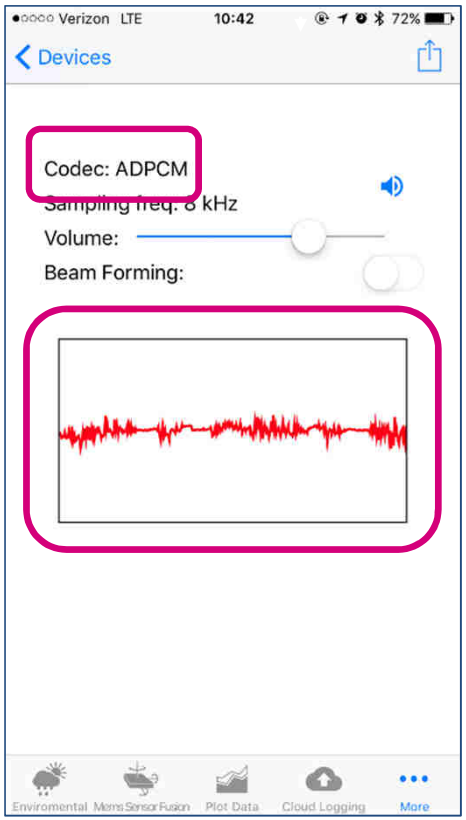
life.augmented

Next Section

Goal

- In this example we are going to demonstrate how to:
  - **Enable the BlueVoice library** for Voice Over BLE streaming
    - Input: raw data from the MEMS microphone
    - Output: ADPCM data streaming @8kHz
  - **Send ADPCM data** through **BLE notifications packets** to the ST BLE Sensor app

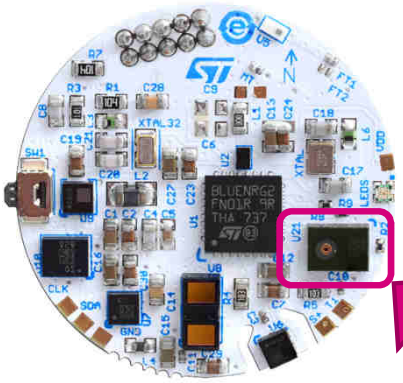
BlueVoice library

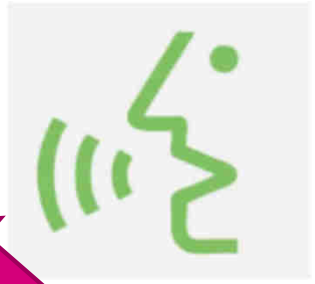


1. **Enable BlueVoice library**
2. **Send voice** to the ST BLE Sensor app client through **BLE notification packet**

Notification Packet

ADPCM compressed audio





MEMS Microphone

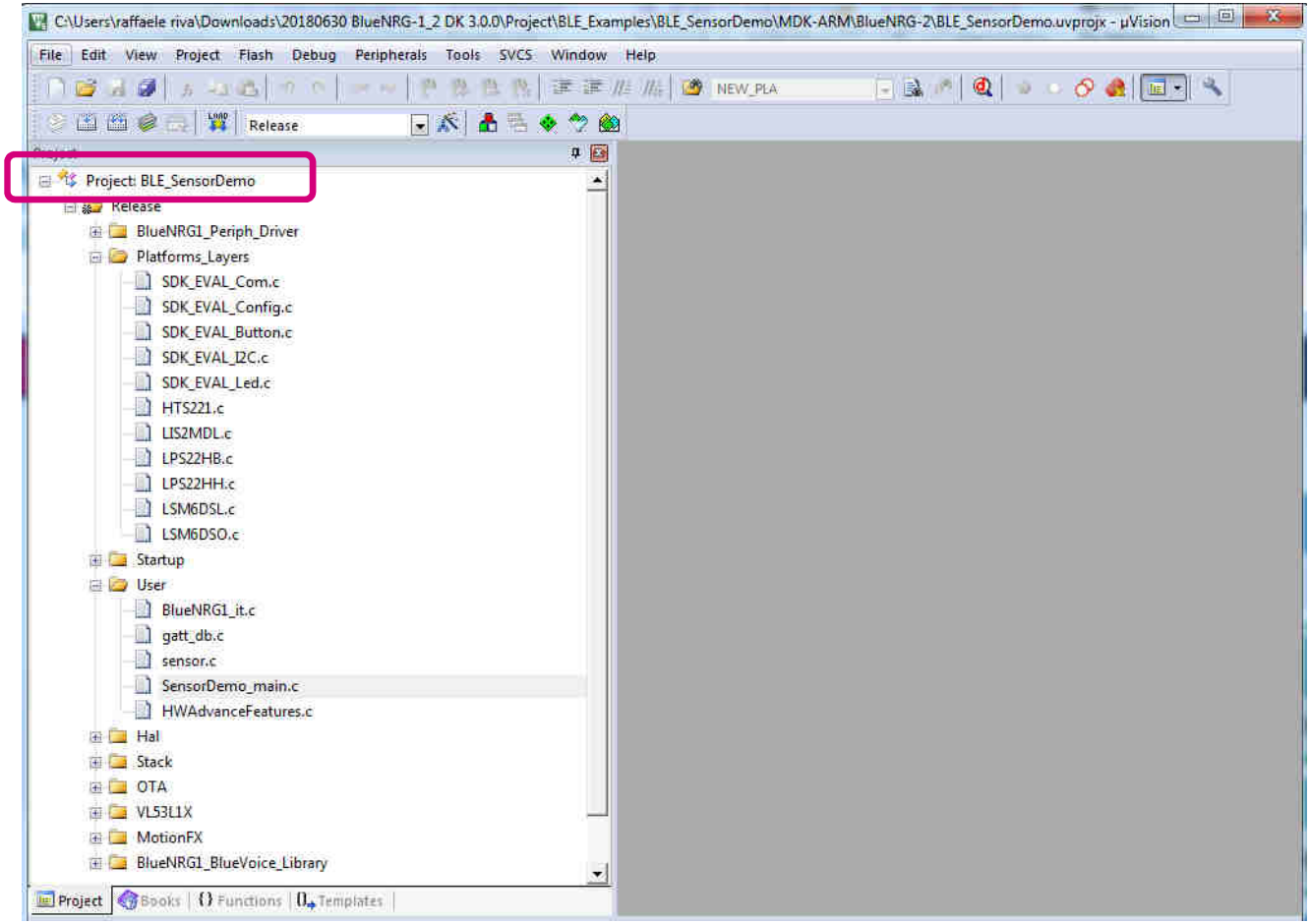
life.augmented

Code modifications

1. **Modify** advertising packet
2. **Enable BlueVoice** (voice over BLE) **embedded library** through preprocessor symbol

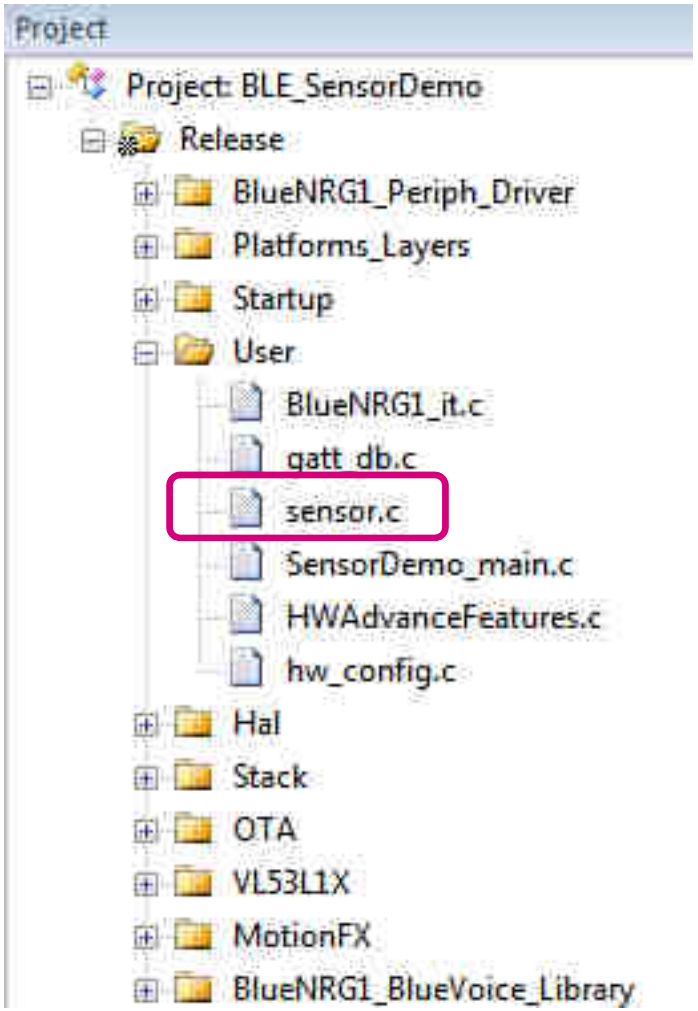
Sensor Demo reference application

Now we **go** **back** again to **Keil uVision**



life.augmented

L7 STEP1: Modify advertising packet



ST BlueMS Protocol

BlueMS

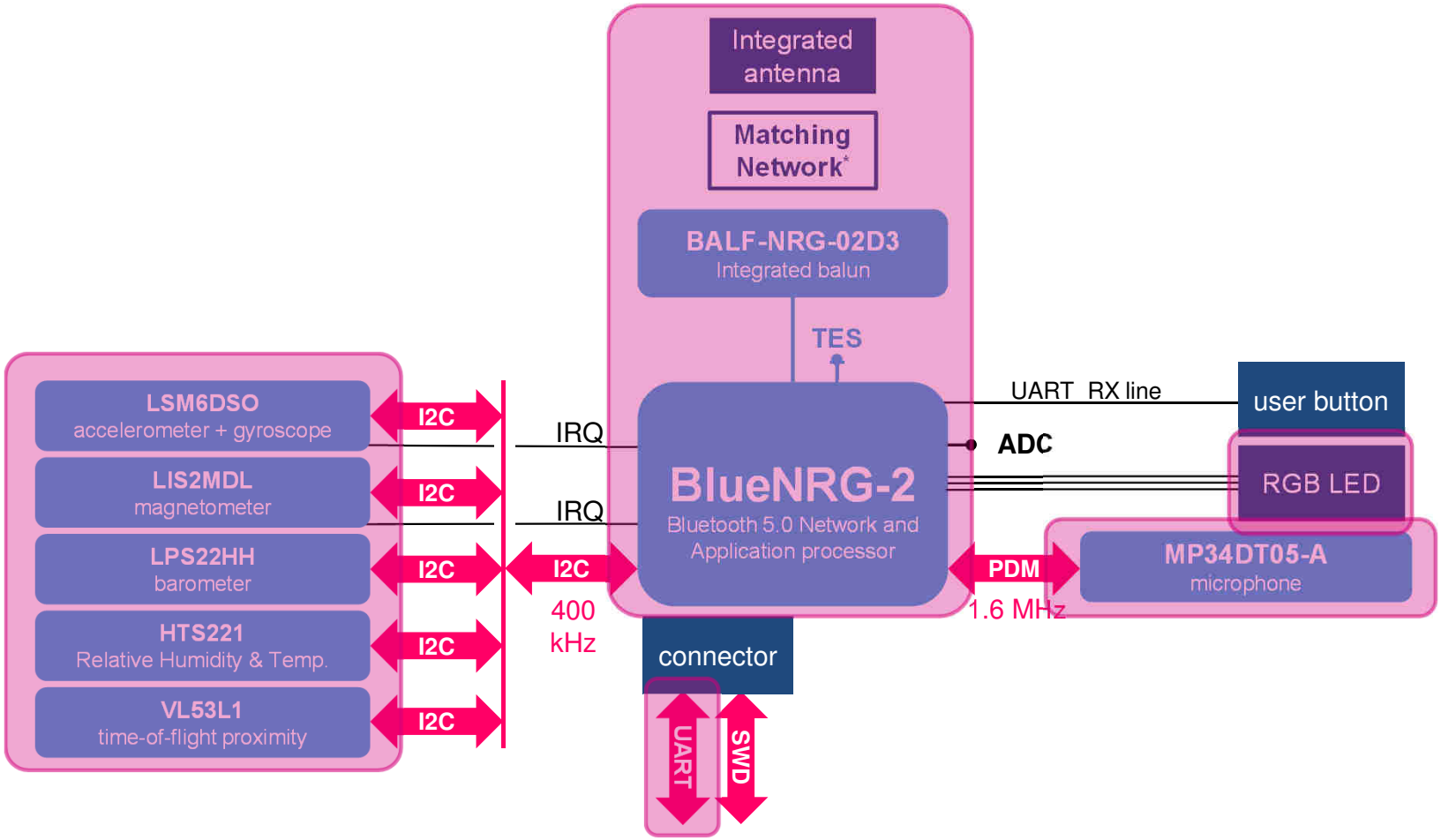
4 Bytes

Len	Type	Protocol version	Device ID	Feature Mask	Device MAC	Len	Type	Local Name	Len	Type	Pwr

31	30	29	28	27	26	25	24	
RFU	ADPCM	Switch	DoA	ADPC	MicLevel	Proximity	Lux	0x6A
23	22	21	20	19	18	17	16	
Acc	Gyro	Mag	Pressure	Humidity	Temperat	Battery	2nd Temp	0xFE
15	14	13	12	11	10	9	8	
RFU	RFU	RFU	RFU	Beamform	AccEvent	FreeFall	SensFusC	0x05
7	6	5	4	3	2	1	0	
SensFus	Compass	MotionInt	Activity	Carry Pos	MemsGes	ProxGes	Pedo	0x40

life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

L7 STEP1: Modify advertising packet

Modify the Feature Mask in the advertisement payload

1. In the file **sensor.c** and **go to line 423**
2. **Modify line 423** Feature Mask byte#1 from **0x22** to **0x6A**

sensor.c

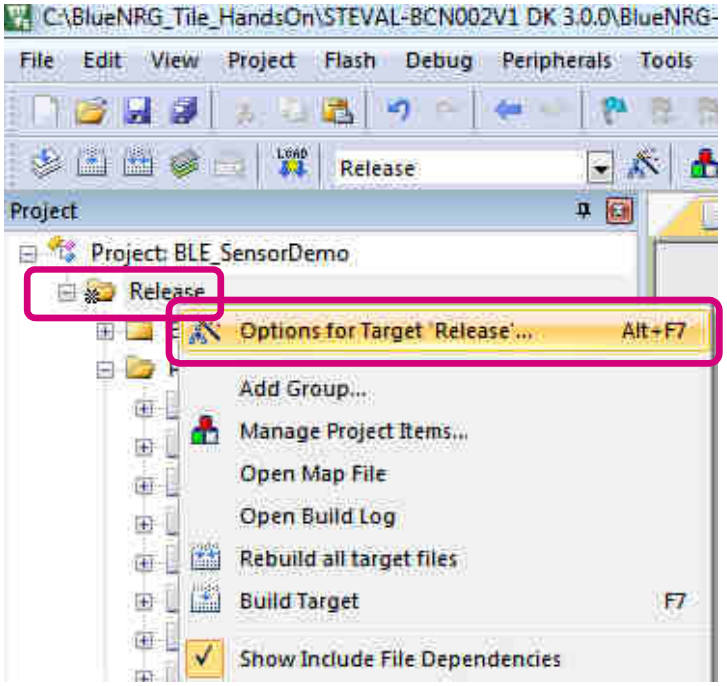
```
411 void Set_DeviceConnectable(void) {
412
413   uint8_t manuf_data[26] = {
414   2,                          /* Length of AD type Transmission Power */
415   0x0A, 0x00,                 /* Transmission Power = 0 dBm */
416   8,                          /* Length of AD type Complete Local Name */
417   0x09,                       /* AD type Complete Local Name */
418   NAME_ALLMEMS,               /* Local Name */
419   13,                         /* Length of AD type Manufacturer info */
420   0xFF,                       /* AD type Manufacturer info */
421   0x01,                       /* Protocol version */
422   0x05,                       /* Device ID: 0x05 = STEVAL-BCN002V1 Board */
423   0x6A,                       /* Feature Mask byte#1: LAB3 0x20 (LED) / LAB5 (
424   0xFE,                       /* Feature Mask byte#2: LAB4 0x9E (Acc+Press+Hur
425   0x05,                       /* Feature Mask byte#3: LAB4 0x04 (AccEvents) /
426   0x40,                       /* Feature Mask byte#4: LAB5 0x40 (eCompass) */
427   0x00,                       /* BLE MAC start */
```

life.augmented

L7 STEP2: Enable BlueVoice library

Add the preprocessor symbol **ENABLE_AUDIO** to the project

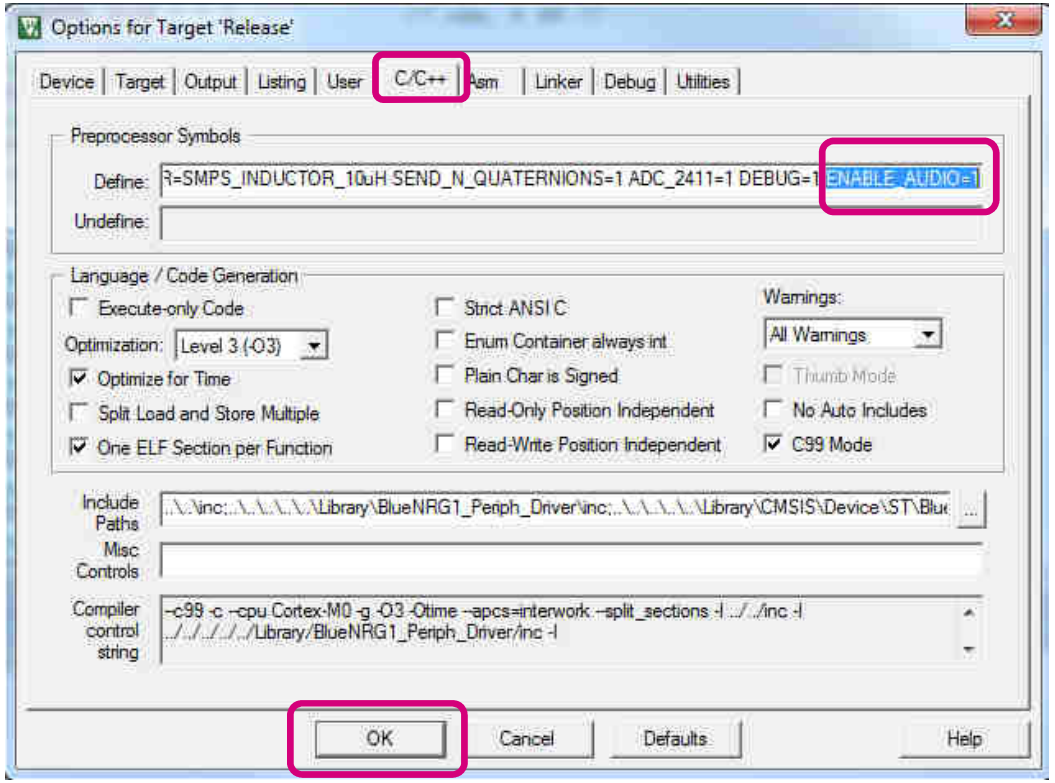
1. **Right-click** on **Release**
2. **Select "Options for Target 'Release'…"**



L7 STEP2: Enable BlueVoice library

In the project options:

1. **Go to** the tab **C/C++** and then in the **Preprocessor Symbols**
2. **Set the symbol ENABLE_AUDIO=1** and then **click** on **OK**

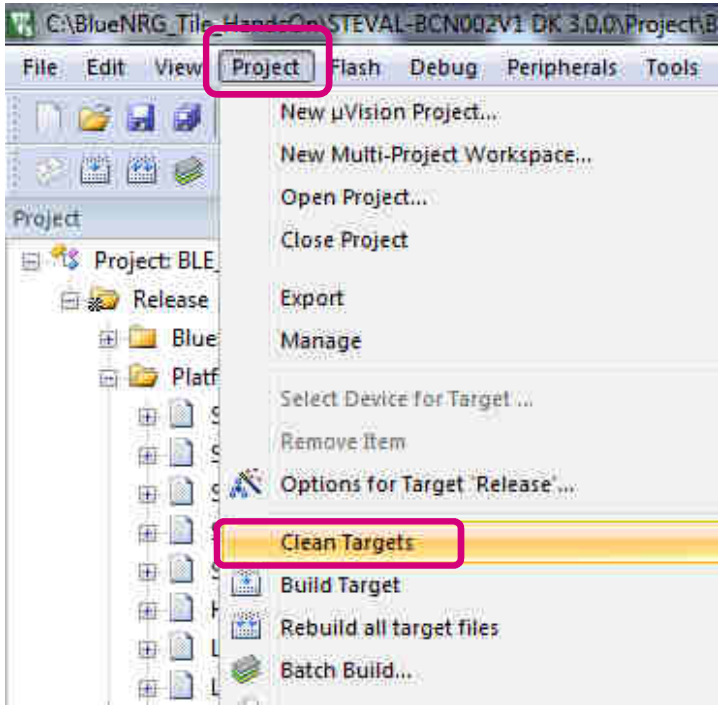


life.augmented

Build the new code

As **we have modified the preprocessor options** it's recommended to clean all the previously compiled object files

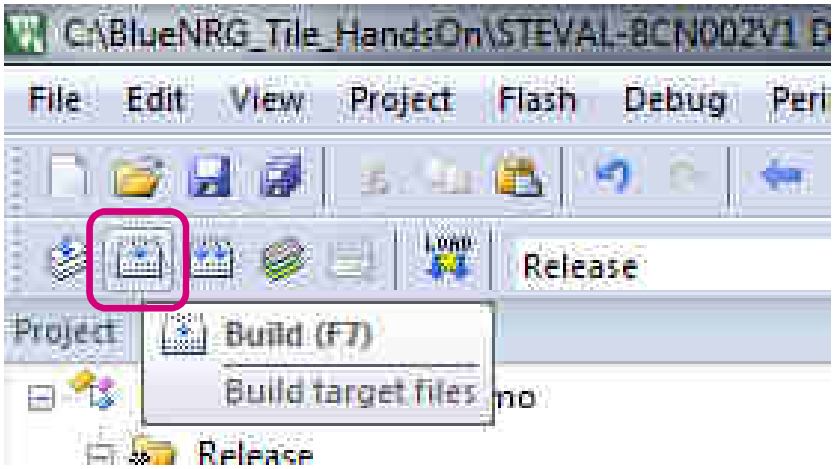
1. **Click** on **Project**
2. **Select** **Clean Targets**



life.augmented

Build the new code

1. **Click** on the **Build button** (top left corner) or hit **F7** on your keyboard
2. In the **Build Output** window (bottom) **wait** for the build to be completed.
   - **BLE_SensorDemo.bin** created
   - "**0 Error(s), 0 Warning(s)**" message appear

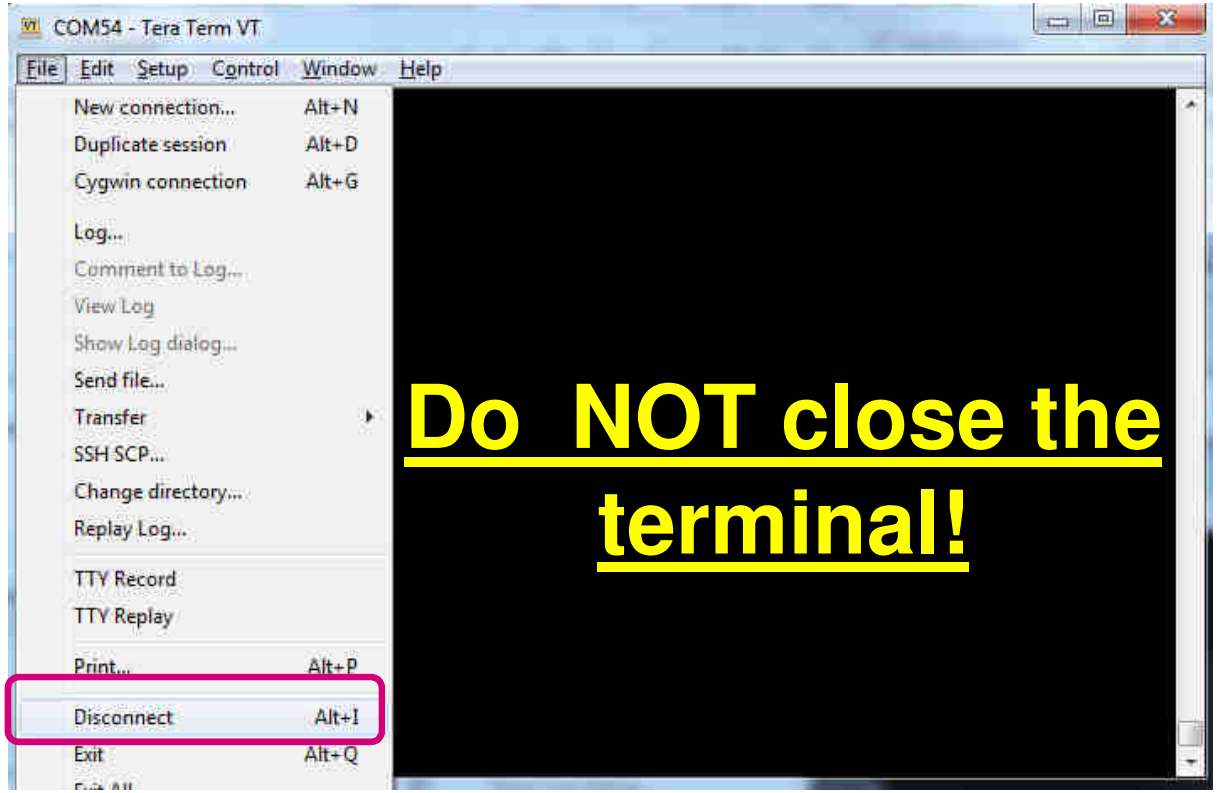



```
Build Output
compiling vl53l1_wait.c...
compiling vl53l1_register_funcs.c...
compiling vl53l1_platform.c...
linking...
Program Size: Code=121908 RO-data=1428 RW-data=1136 ZI-data=21252
FromELF: creating hex file...
After Build - User command #1: fromelf.exe --bin ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf --output ..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.bin
"..\..\..\..\..\..\HandsOn\BLE_SensorDemo_PrjOutput\BLE_SensorDemo.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:12
```

life.augmented

Disconnect the serial terminal

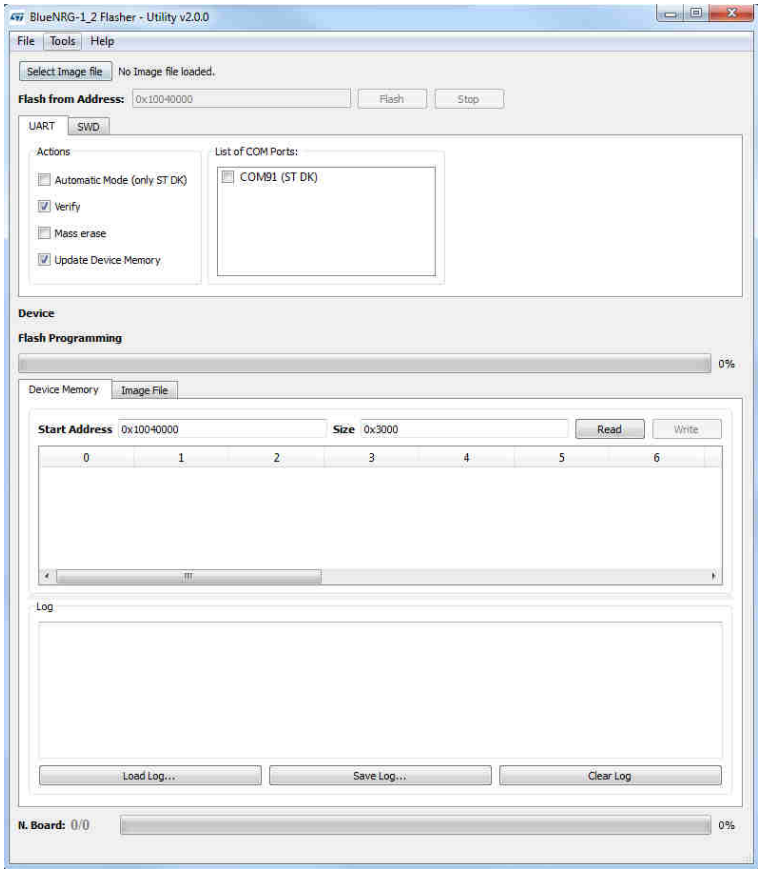
1. **Go back** to **Tera Term**
2. **Click** on the **File->Disconnect**



life.augmented

Re-flash the BlueNRG-2

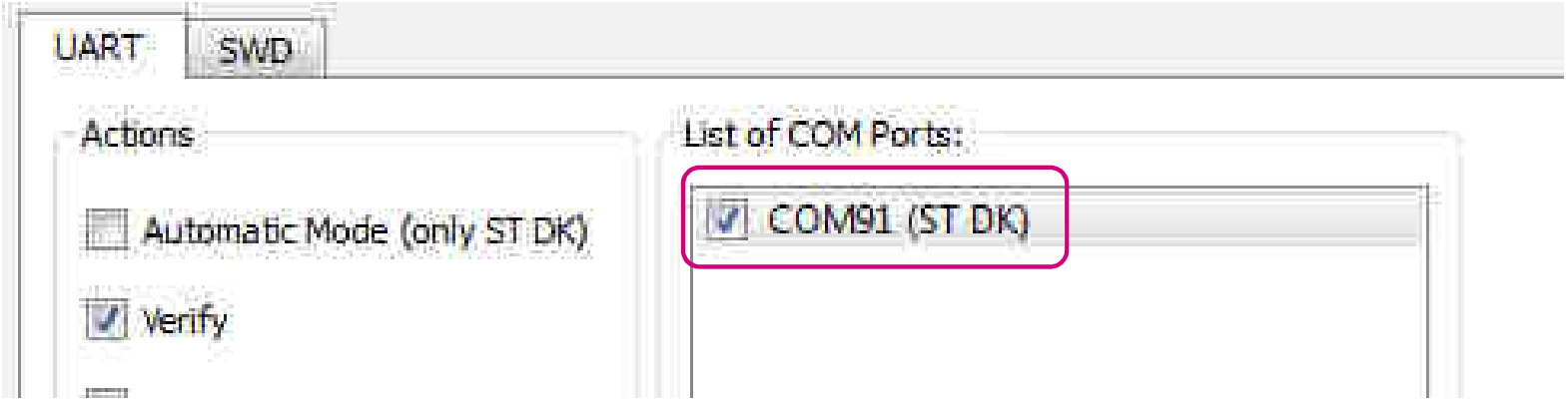
1. **Go back** to the **BlueNRG-2 Flasher Utility**



life.augmented

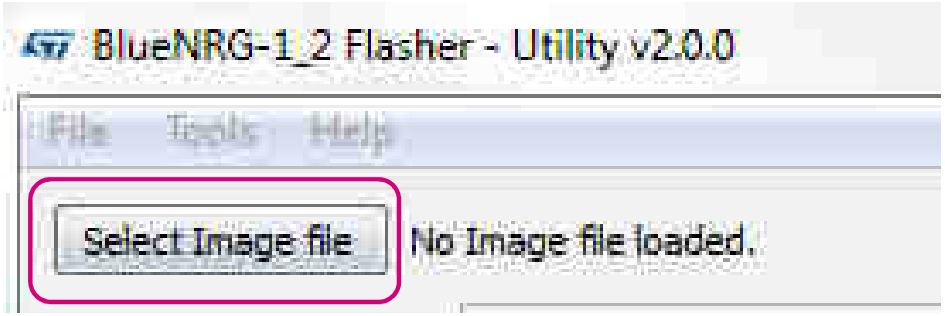
Flash the BlueNRG-2 1/4

1. **Select** the COM port labeled **(ST DK)**



Flash the BlueNRG-2 2/4

1. **Click** on the **Select Image file** button

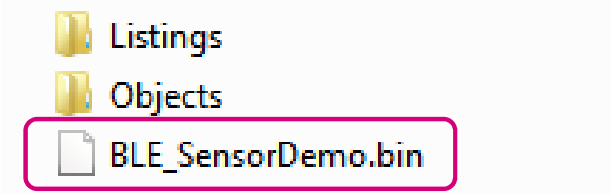


2. And **browse** the following **path**

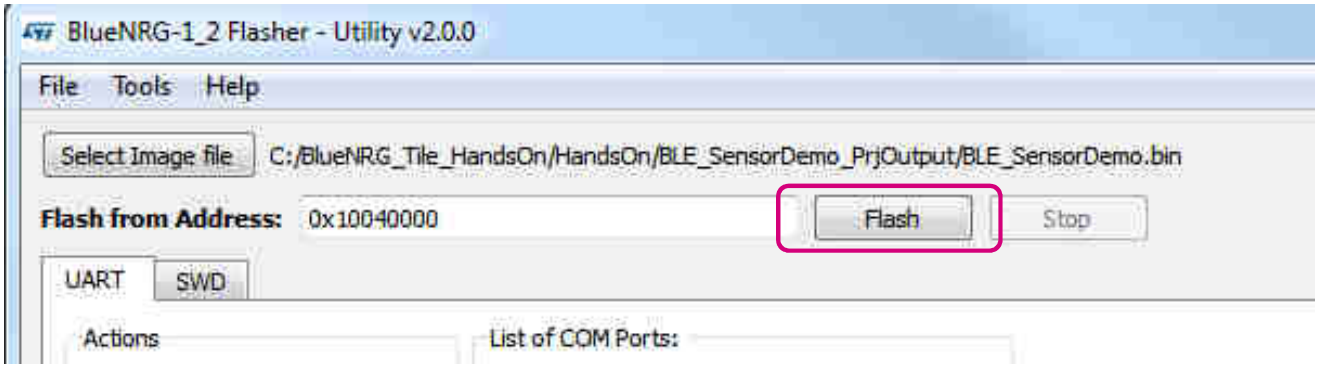
life.augmented

Flash the BlueNRG-2 3/4

1. **Select** BLE_SensorDemo.bin



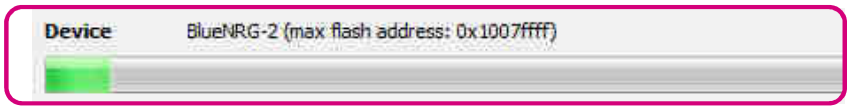
2. **Click** on the **Flash** button



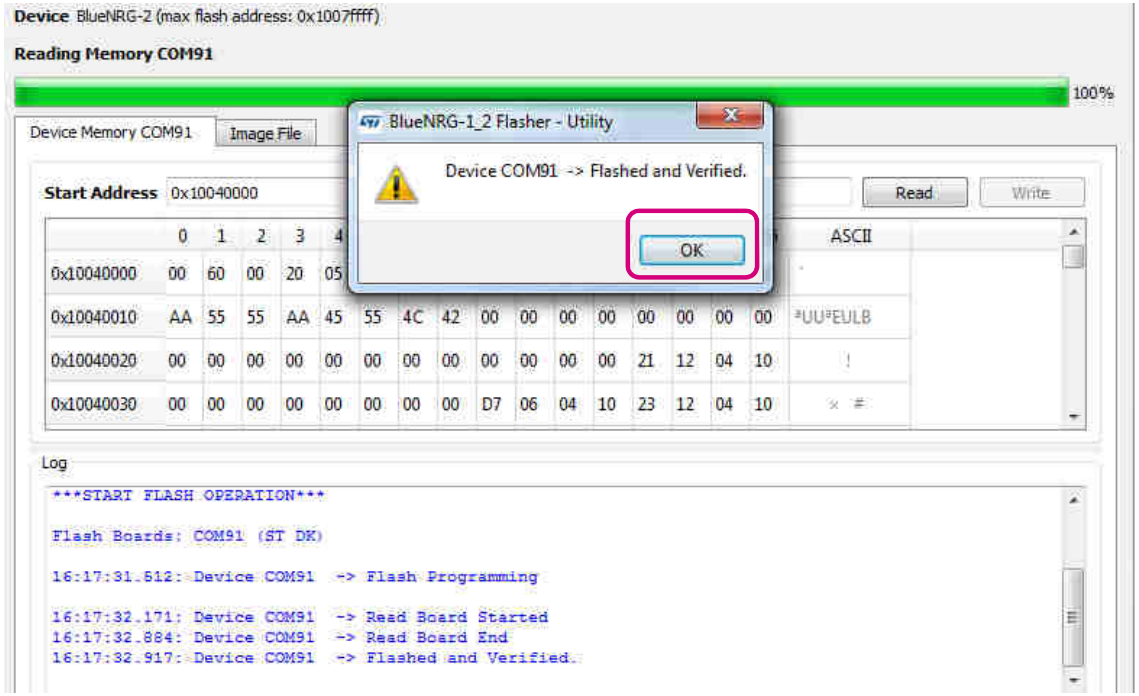
life.augmented

Flash the BlueNRG-2 4/4

1. Flashing starts: **green bar** proceeding



2. **Wait** for the **pop-up** window and **click** on **OK**

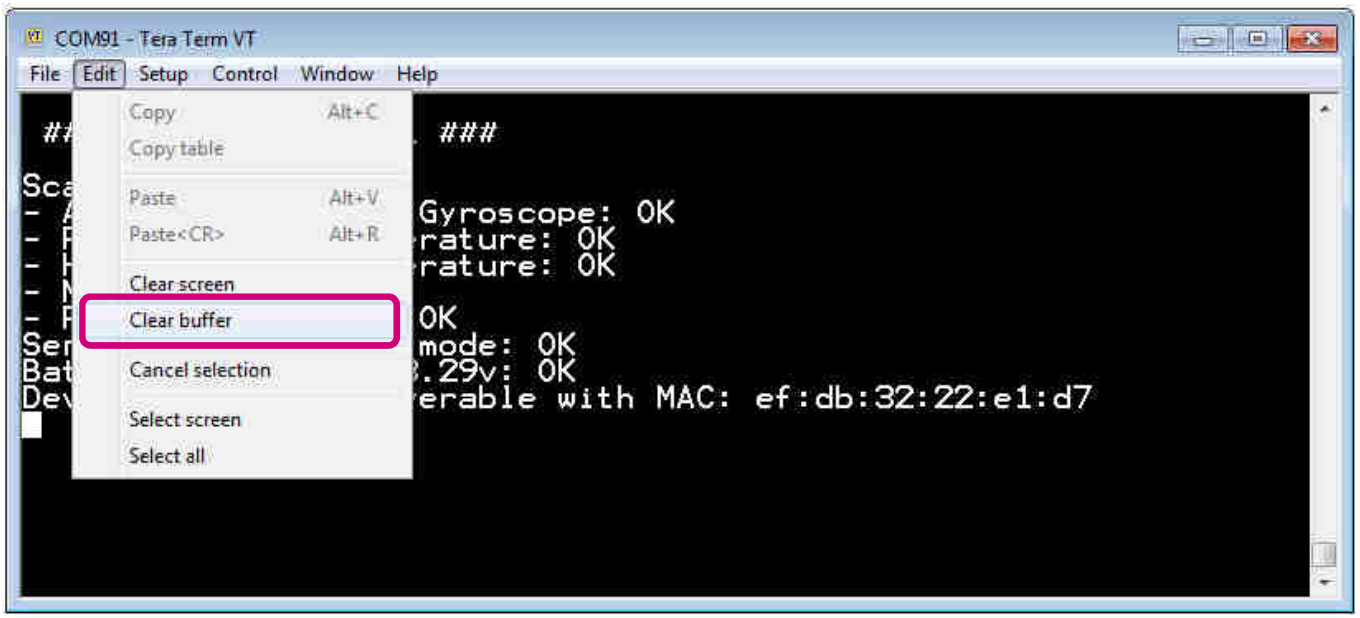


Do NOT close the Utility!

life.augmented

Clean Buffer in the serial terminal

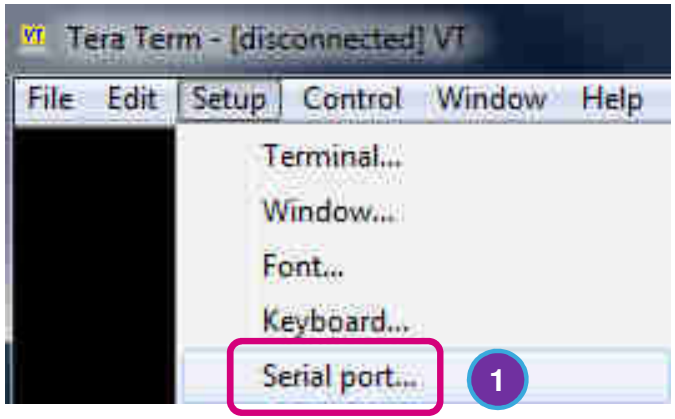
1. In **Tera Term** in order to have the terminal clean, **go to Edit -> Clear buffer**

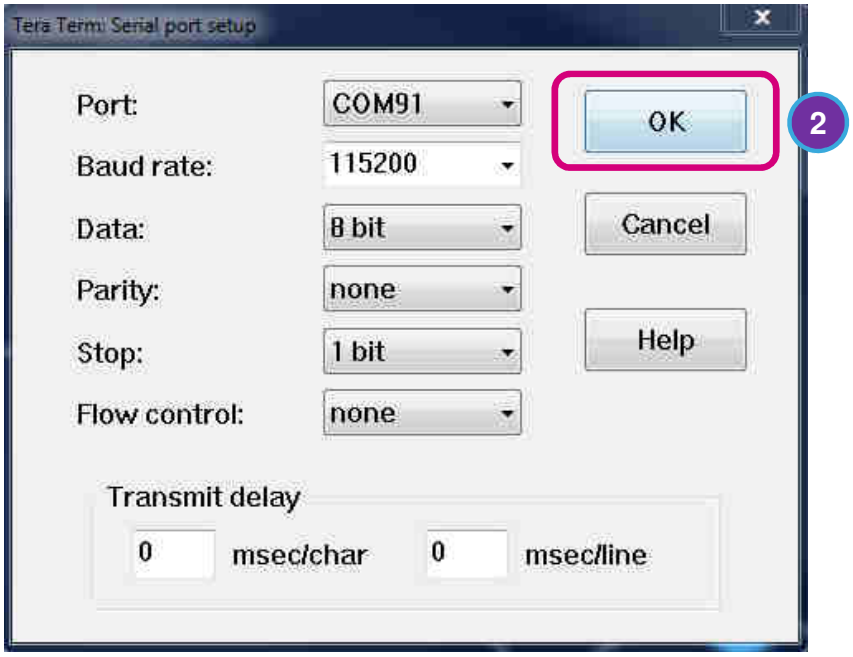


life.augmented

Reconnect the serial terminal

1. **Click Setup -> Serial port…**
2. Serial port should be already configured.
   Just need to **click** on **OK**

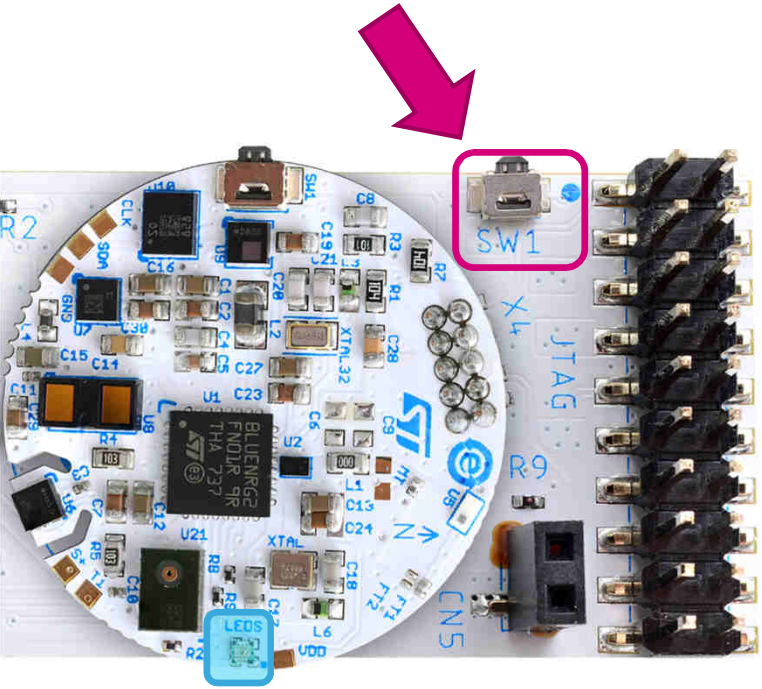


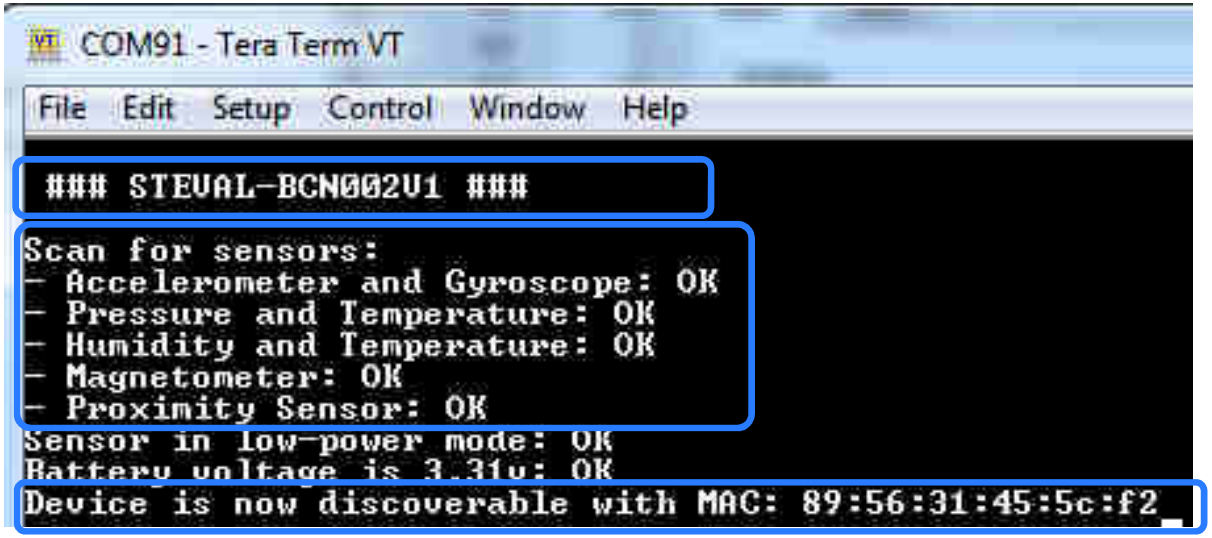


life.augmented

Sanity Check on serial port

Push SW1 button on the daughterboard -> **LED blinking Blue**

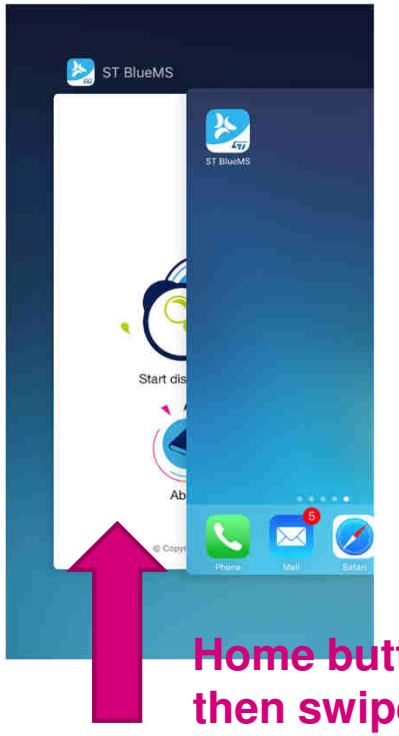




life.augmented

Connect using the ST BLE Sensor App

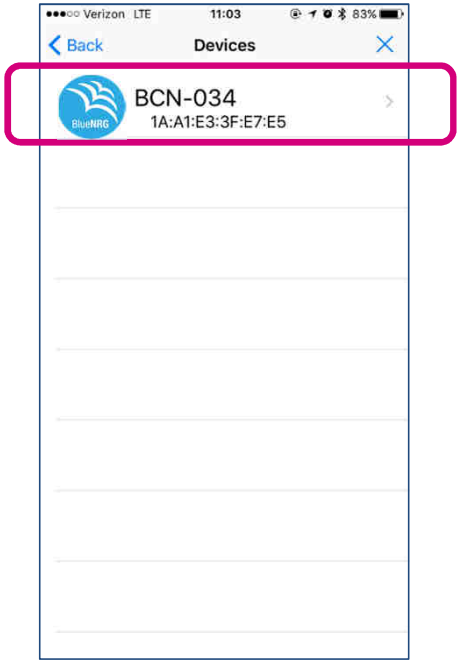
1 **"Kill" the BLE Sensor app**



Home button and then swipe to top

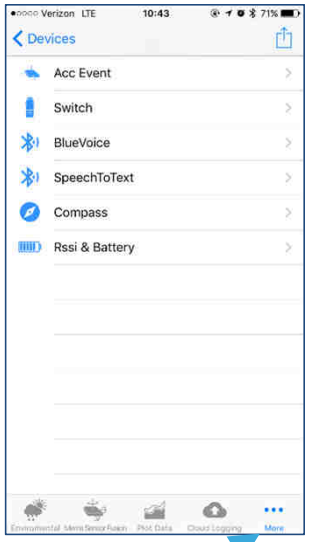
2 **Tap "Connect to a device"**

3 **Select your STEVAL-BCN002V1**



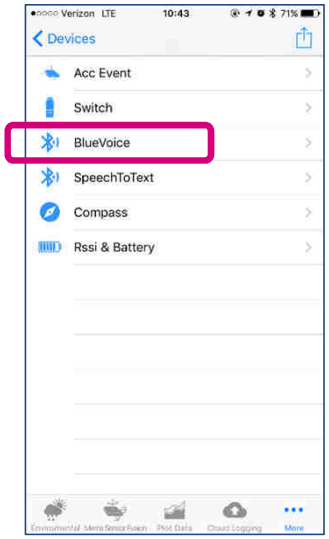
life.augmented

BlueVoice: voice over Bluetooth LE

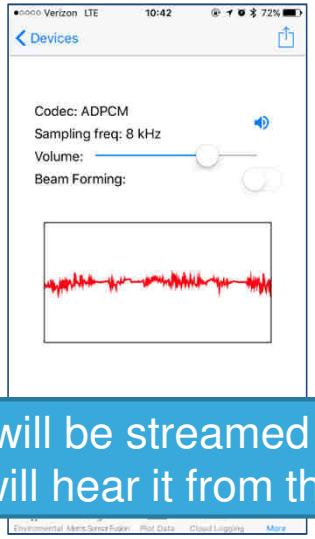


Tap

Select
"BlueVoice"

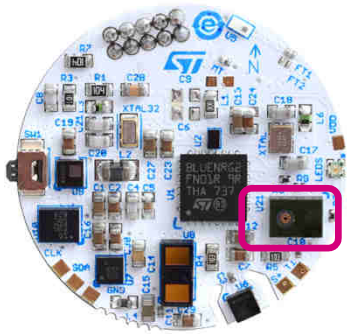


Speak close to
the BlueNRG-Tile



Voice will be streamed over BLE
You will hear it from the phone

Do not silence your phone,
must **NOT** be vibration only!

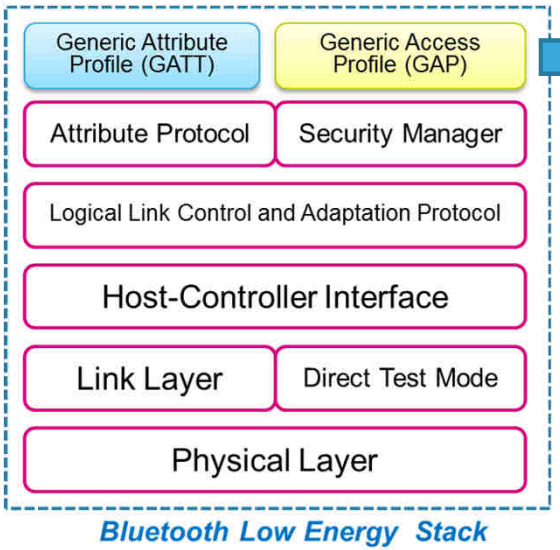


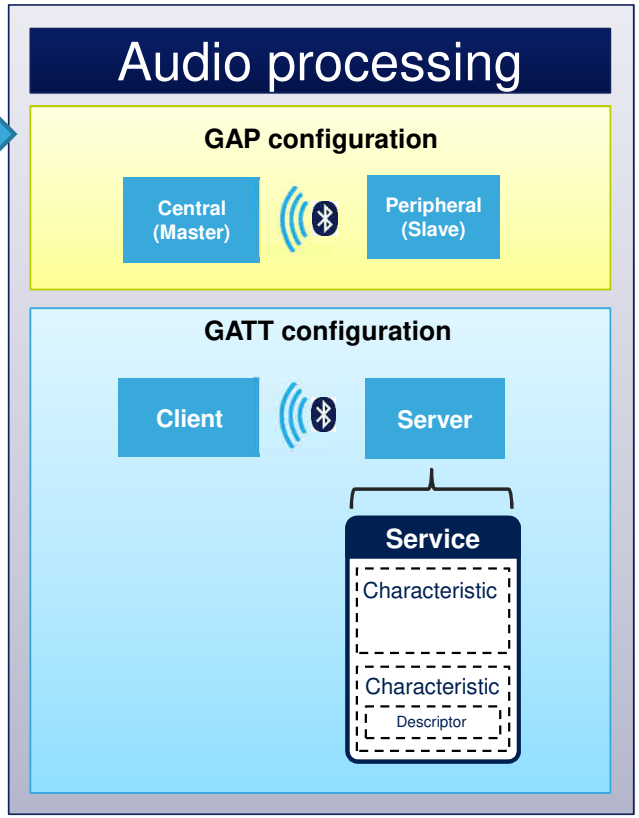
MEMS Microphone

(if the mic captures the audio from the phone speaker, a very high pitch sound can happen!)

life.augmented

Voice over Bluetooth LE





It's a Vendor Specific Service!

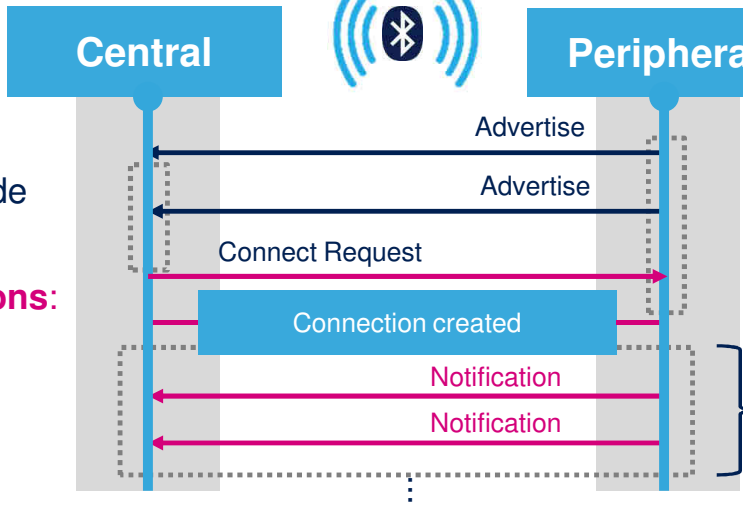
Audio: exported by the Server through **2 dedicated BLE characteristics**

Voice streaming over BLE:
Audio @8kHz
Codec: ADPCM
Bitrate: 32kbps

life.augmented

BlueVoice ADPCM - Audio 8kHz

Discovery Mode



Advertisement Mode

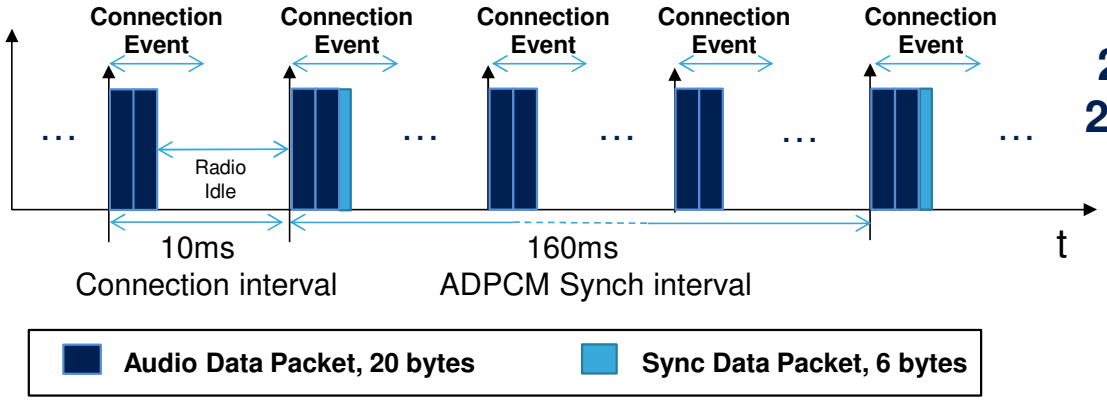
No acknowledge required for **Notifications**:

- OK for Audio streaming
- **Minimal Latency**
- **Optimal Bandwidth**

Audio can be half-duplex

Voice Streaming

- Audio Format: **PCM 16 bit @ 8 kHz**
- **ADPCM Compression** @ 32 Kbps: Low latency and low complexity
- Side Information at low data rate enhances error resilience



2 packets/event
20 bytes/packets
100 events/sec

=

32kbps

life.augmented

BlueVoice architecture

"Packetization"

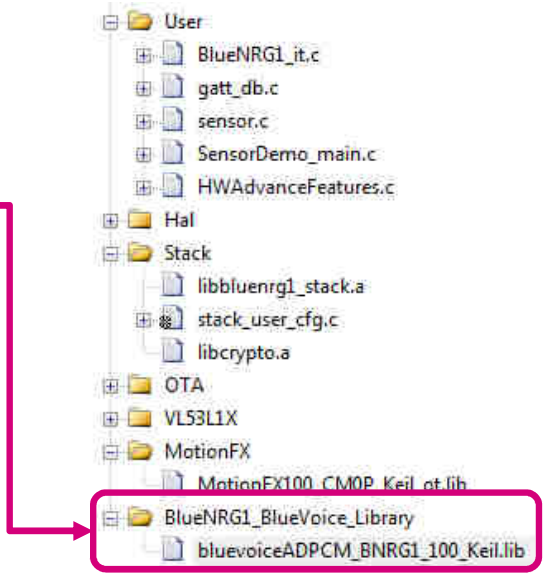
ST digital MEMS microphone

BlueNRG-2

Audio Acquisition → PDM → PDM to PCM conversion → PCM 8kHz → ADPCM Compression → Raw Data → Server - TX Bluetooth SMART

HW implementation

Bitrate 32 kbps

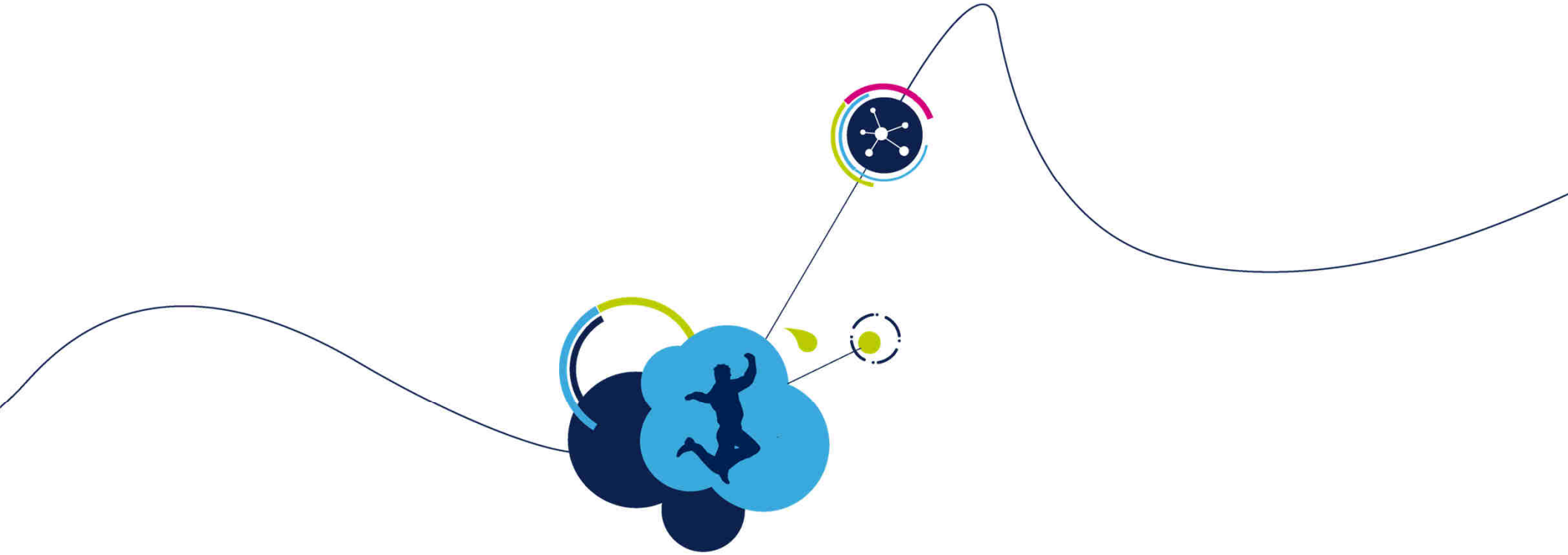


"De-Packetization"

Smartphone app (ST BLE Sensor app)

Client - RX Bluetooth SMART → Raw Data → Audio Decompression → PCM → Serial Audio Out USB, I2S, …

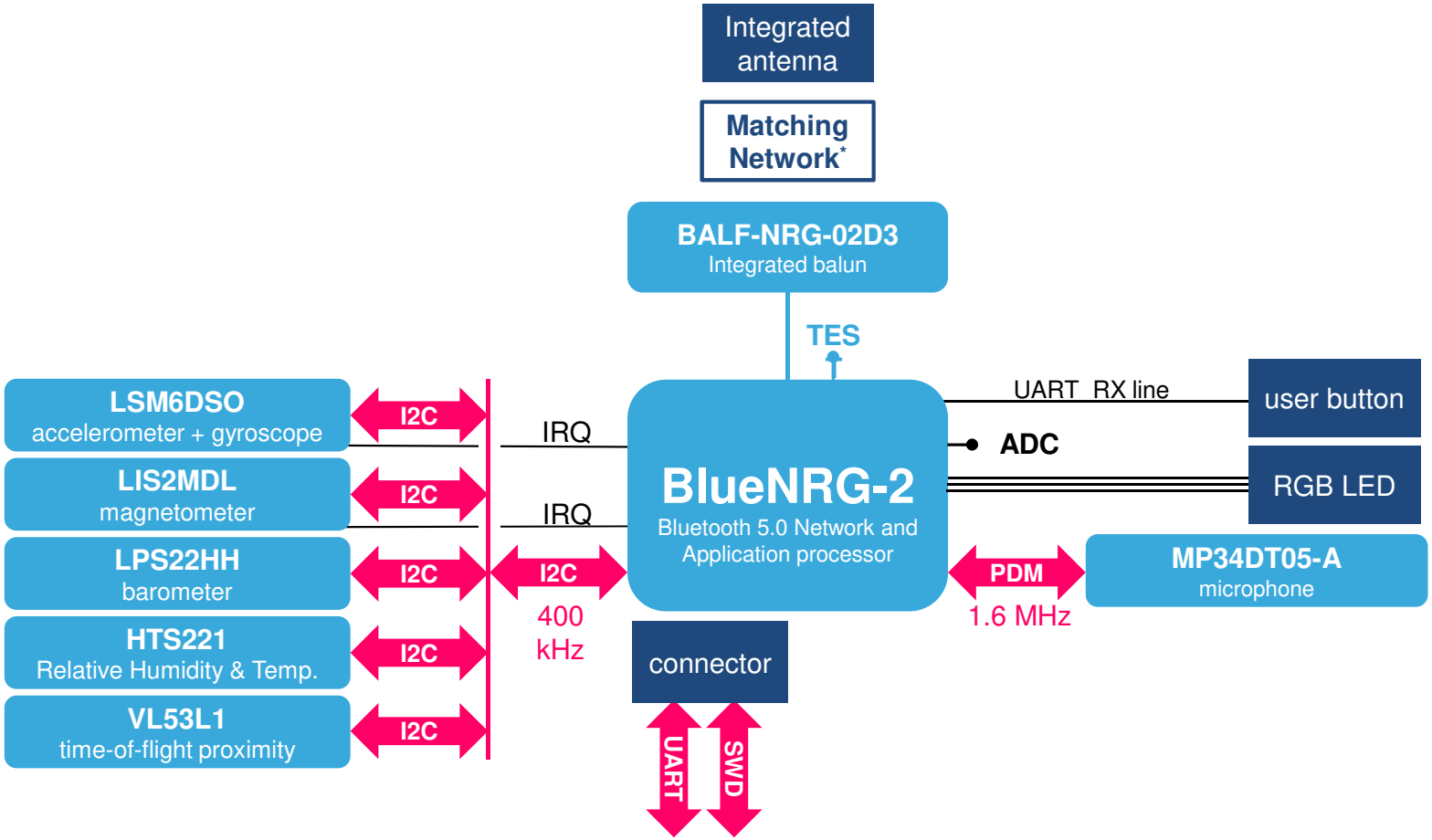
life.augmented



Customizing your design

life.augmented

STEVAL-BCN002V1 Block Diagram



life.augmented

Enable/Disable Sensors & Libraries

- **Dedicated structure** “**FeaturePresence**” for enabling/disabling sensors & libraries individually. File **sensor.h** at line **64**

```
64 typedef struct {
65   bool AccelerometerGyroscopePresence;
66   bool MagnetometerPresence;
67   bool HumidityTemperaturePresence;
68   bool PressurePresence;
69   bool ProximityLightPresence;
70   bool iNemoEngine;
71   bool Pedometer;
72 } FeaturePresence;
```

- **SensorScan() function** set each field of the structure to **false** or **true**. File **sensor.c** at line **319**

Disable the unneeded sensor for optimizing the power consumption!

```
318 // Check sensor list
319 SensorsScan();
320
321 // Configure discovered sensors
322 if (xFeaturePresence.PressurePresence)
323   Init_Pressure_Temperature_Sensor();
324 if (xFeaturePresence.HumidityTemperaturePresence)
325   Init_Humidity_Sensor();
326 if (xFeaturePresence.MagnetometerPresence)
327   Init_Magnetometer();
328 if (xFeaturePresence.AccelerometerGyroscopePresence)
329   Init_Accelerometer_Gyroscope();
330 if (xFeaturePresence.ProximityLightPresence)
331   Init_Proximity_Sensor();
332
333 // Configure sensors in low power mode
334 SensorsLowPower();
```

BLE timings

- **Advertising intervals**:
  - Dedicated API *aci_gap_set_discoverable(Advertising_Type, Advertising_Interval_Min, Advertising_Interval_Max, …)*
  - In file **sensor.c** at line **451**
- **Connection intervals**:
  - Dictated by the Central device. Peripheral has no full control on this.
- **Notifications frequency**
  - Dedicated Virtual Timers (mapped on HW physical timers) for different functionalities
  - Timeouts defined in **sensor.h** at line **172-174**

```
172 #define BATTERY_UPDATE_RATE     1000    // Fixed ODR @  1 Hz
173 #define ENV_SENSOR_UPDATE_RATE   100     // Fixed ODR @  10 Hz
174 #define MOTION_SENSOR_UPDATE_RATE   40    // Fixed ODR @ 25 Hz
```

HW modifications

- In this case redesign is of course necessary
- **Schematics** and **Gerbers** files:
  - online at www.st.com/bluetile
- BlueNRG-2 **pin mapping**
  - Check **BlueNRG-2 DS** at **Table 129**

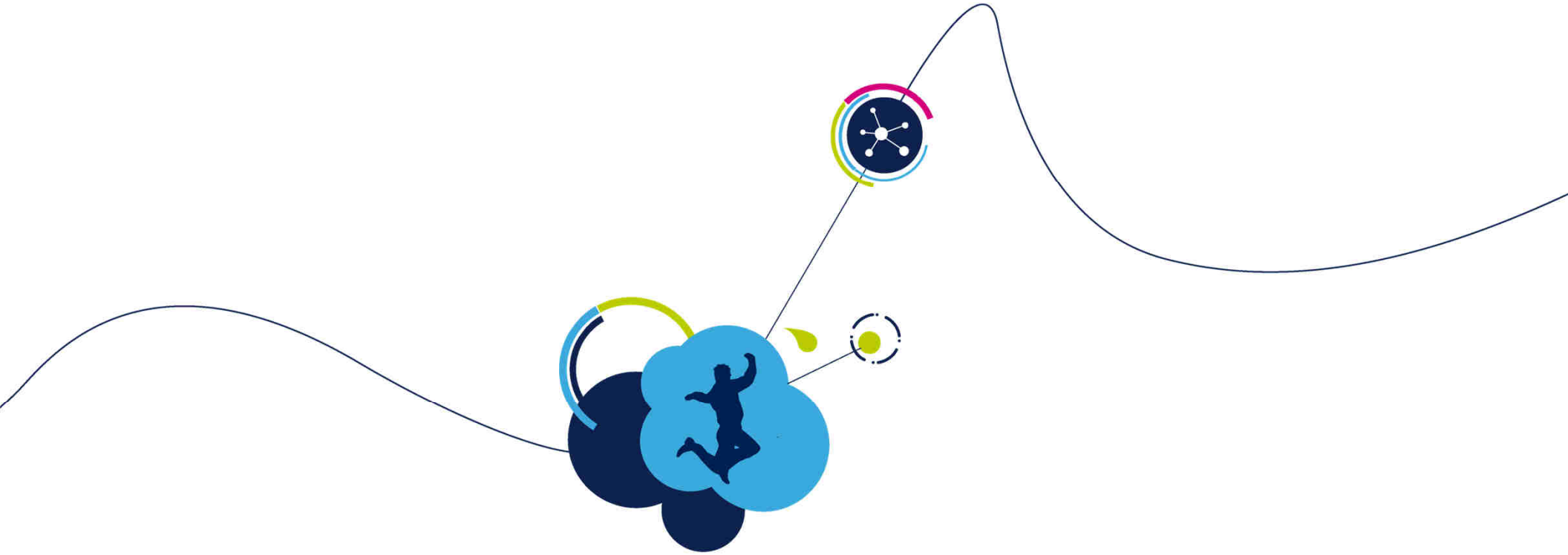
3.12.2 **Functional description**

The table below shows the GPIO configuration table where each IO pin is associated with related functions.

Table 129. **IO functional map**

Pin name(1)	GPIO mode "000"		Serial1 mode '001"		Serial0 mode '100'		Serial2 mode '101'	
	Type	Signal	Type	Signal	Type	Signal	Type	Signal
IO0	I/O	GPIO 0	I	UART_CTS	I/O	SPI_CLK	O	CPUCLK
IO1	I/O	GPIO 1	O	UART_RTS	I/O	SPI_CS1	I	PDM_DATA
IO2	I/O	GPIO 2	O	PWM0	O	SPI_OUT	O	PDM_CLK
IO3	I/O	GPIO 3	O	PWM1	I	SPI_IN	-	-
IO4	I/O	GPIO 4	I	UART_RXD	I/O	I2C2_CLK	O	PWM0
IO5	I/O	GPIO 5	O	UART_TXD	I/O	I2C2_DAT	O	PWM1
IO6	I/O	GPIO 6	O	UART_RTS	I/O	I2C2_CLK	I	PDM_DATA
IO7	I/O	GPIO 7	I	UART_CTS	I/O	I2C2_DAT	O	PDM_CLK
IO8	I/O	GPIO 8	O	UART_TXD	I/O	SPI_CLK	I	PDM_DATA
IO9	I/O	GPIO 9	I	SWCLK	I	SPI_IN	O	XO16/32M
IO10	I/O	GPIO 10	I	SWDIO	O	SPI_OUT	O	CLK_32K
IO11	I/O	GPIO 11	I	UART_RXD	I/O	SPI_CS1	O	CLK_32K
IO12	OD	GPI 12 (2)		-	I/O	I2C1_CLK	-	-
IO13	OD	GPI 13(2)	I	UART_CTS	I/O	I2C1_DAT	-	-
IO14	I/O	GPIO 14	I/O	I2C1_CLK	I/O	SPI_CLK	-	-

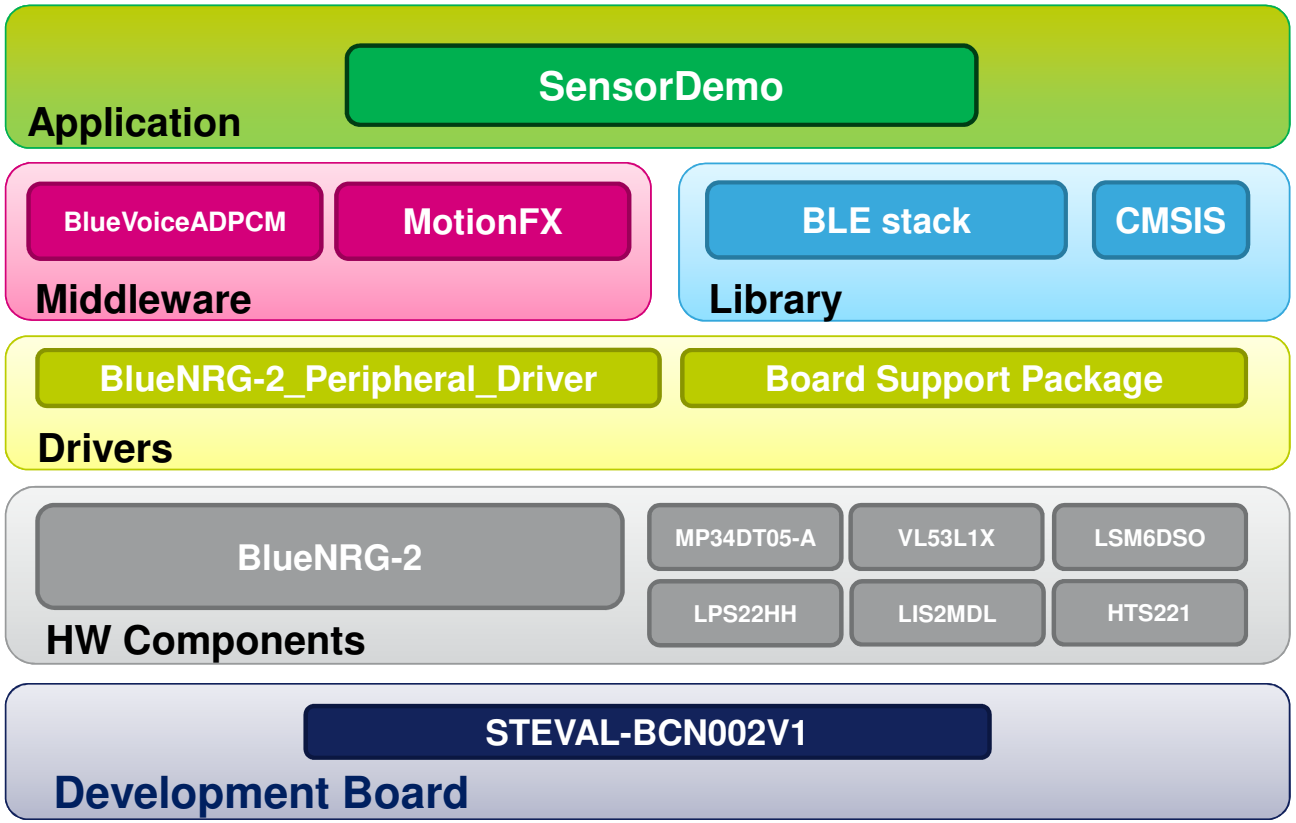
life.augmented



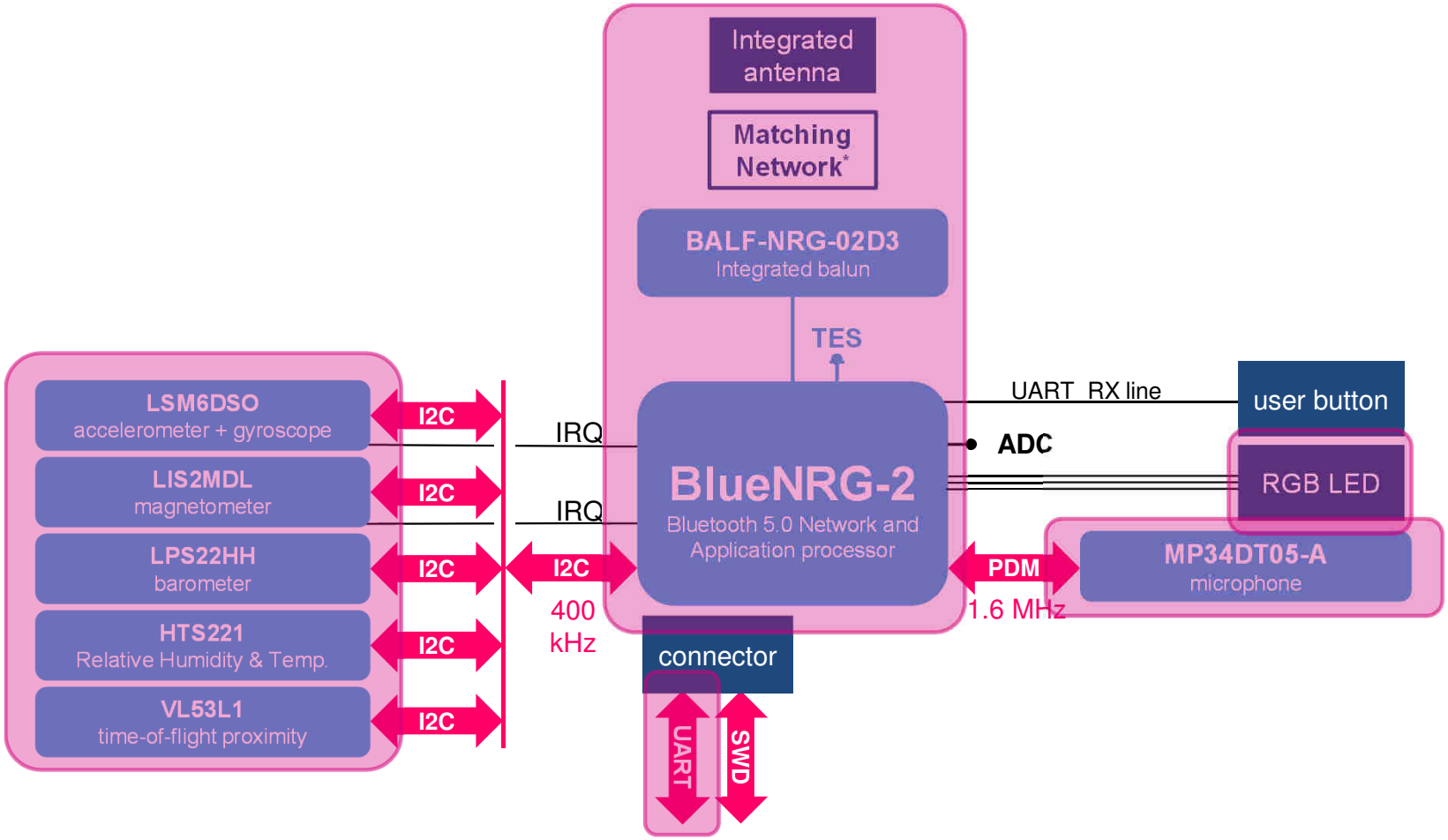
Quick recap

life.augmented

DK 3.0.0 SW architecture



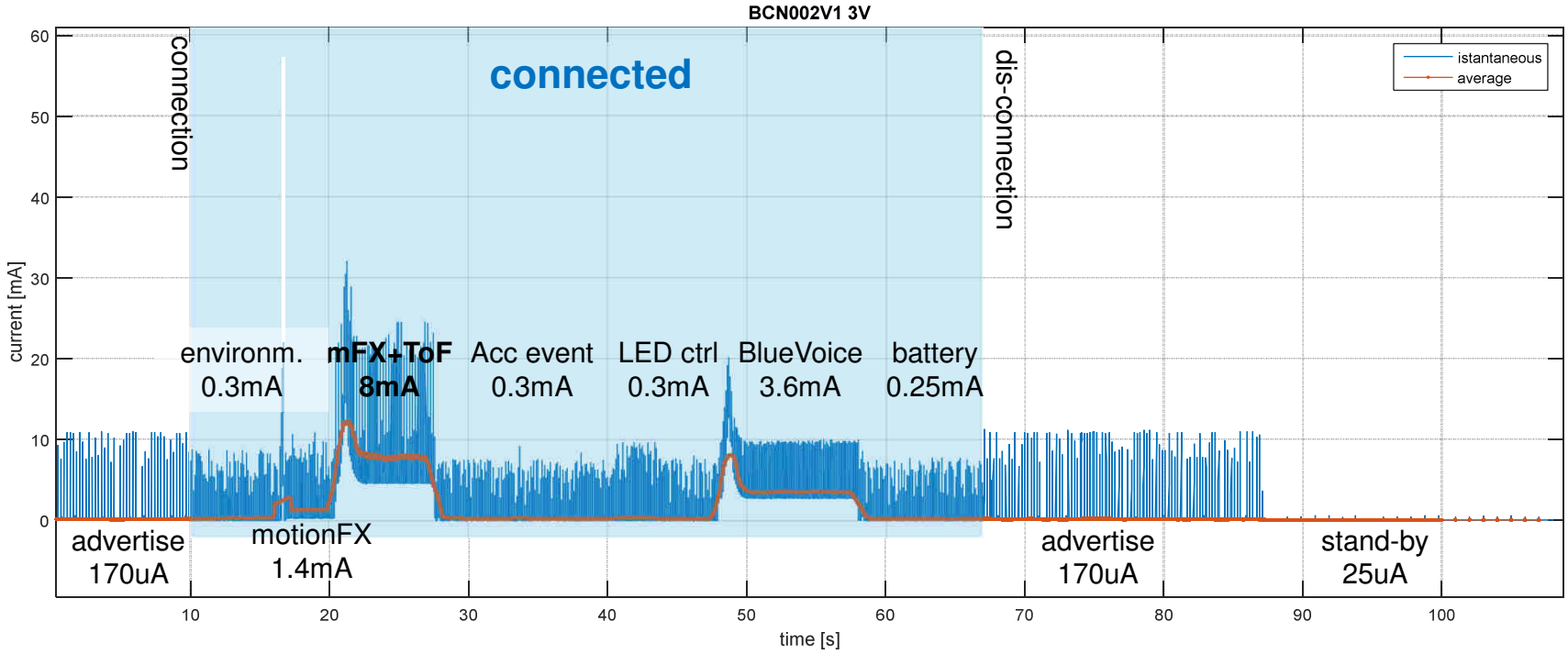
STEVAL-BCN002V1 Block Diagram



life.augmented

Power consumption

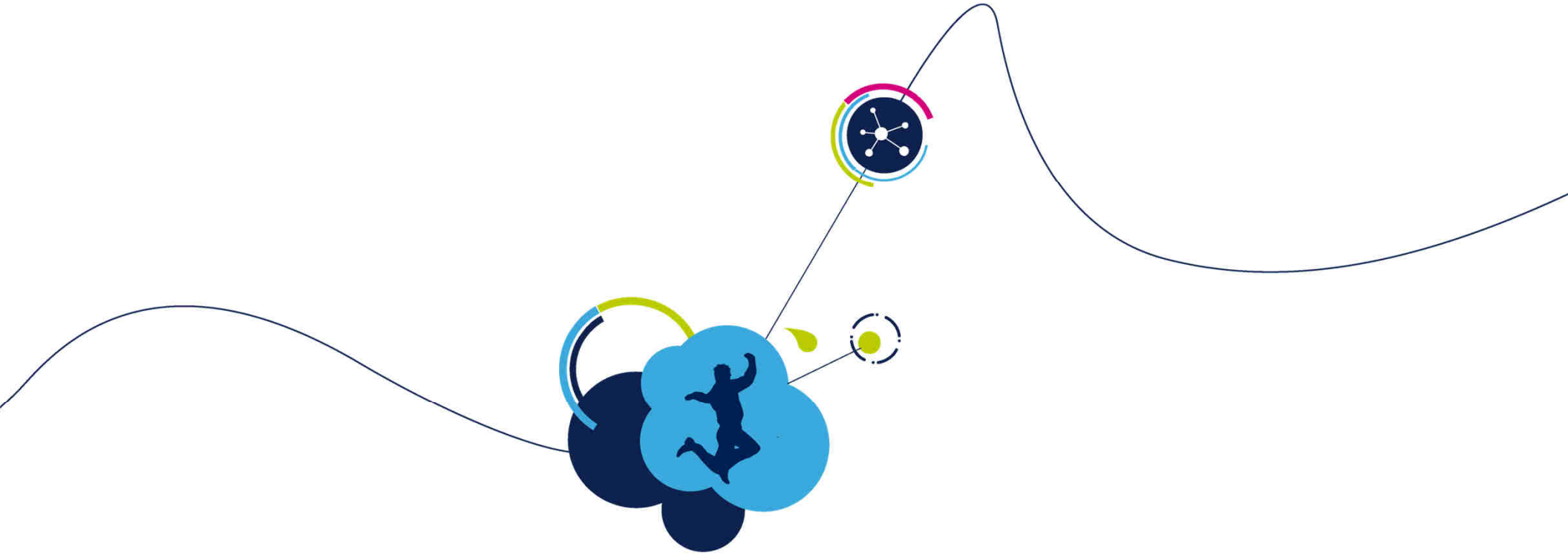
State	Avg power cons
stand-by	25uA
advertise 250ms	170uA
Battery notification	0.25mA
Environmental / AccEvents / LED control	0.3mA
Motion FX (Inertial)	1.4mA
BlueVoice	3.6mA
Motion FX plus Time Of Flight	8mA



life.augmented

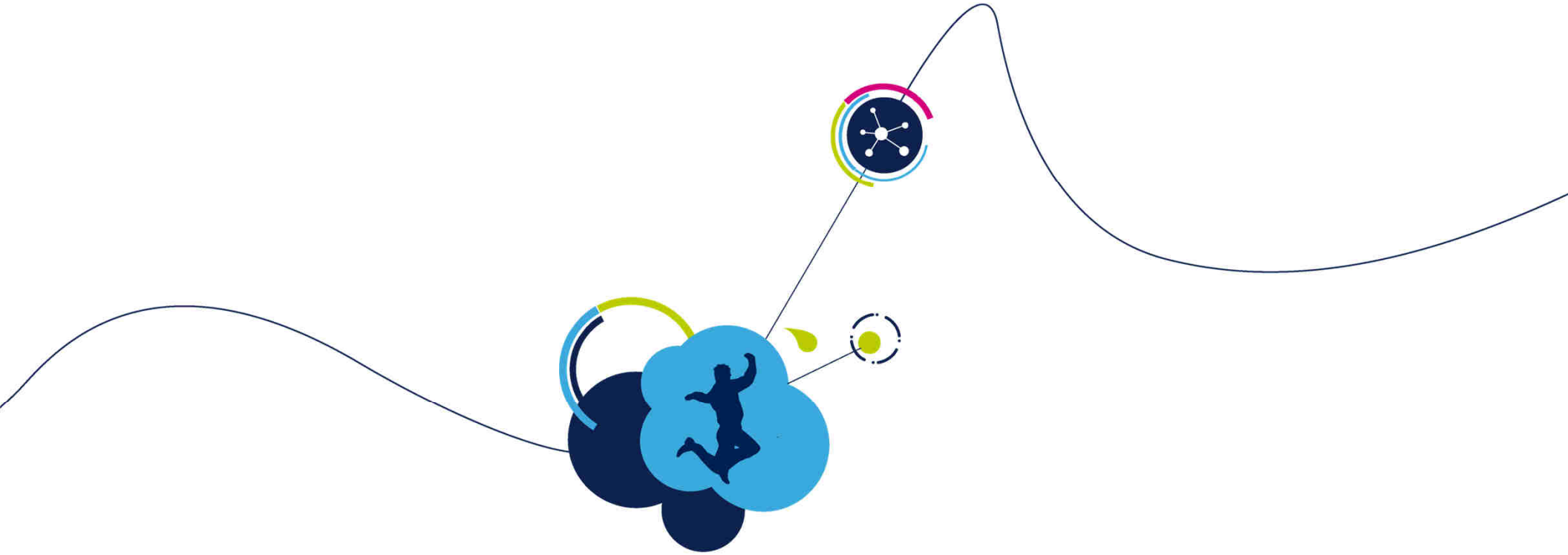
Battery lifetime: CR2032

State	Avg Power Cons	Battery lifetime
stand-by	25uA	8800 hrs / 367 days
advertise 250ms	170uA	1294 hrs / 54 days
Battery notification	0.25mA	880 hrs / 37 days
Environmental / AccEvents / LED control	0.3mA	733 hrs / 30 days
Motion FX (Inertial)	1.4mA	157 hrs / 6.5 days
BlueVoice	3.6mA	61.1 hrs / 2.5 days
Motion FX plus Time Of Flight	8mA	27.5 hrs



The end: Q&A

life.augmented

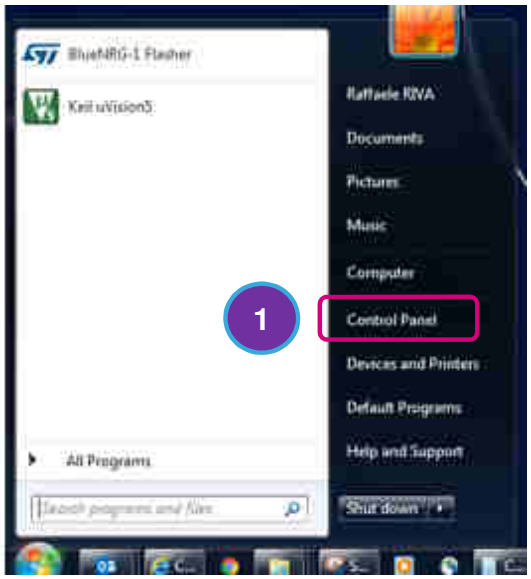


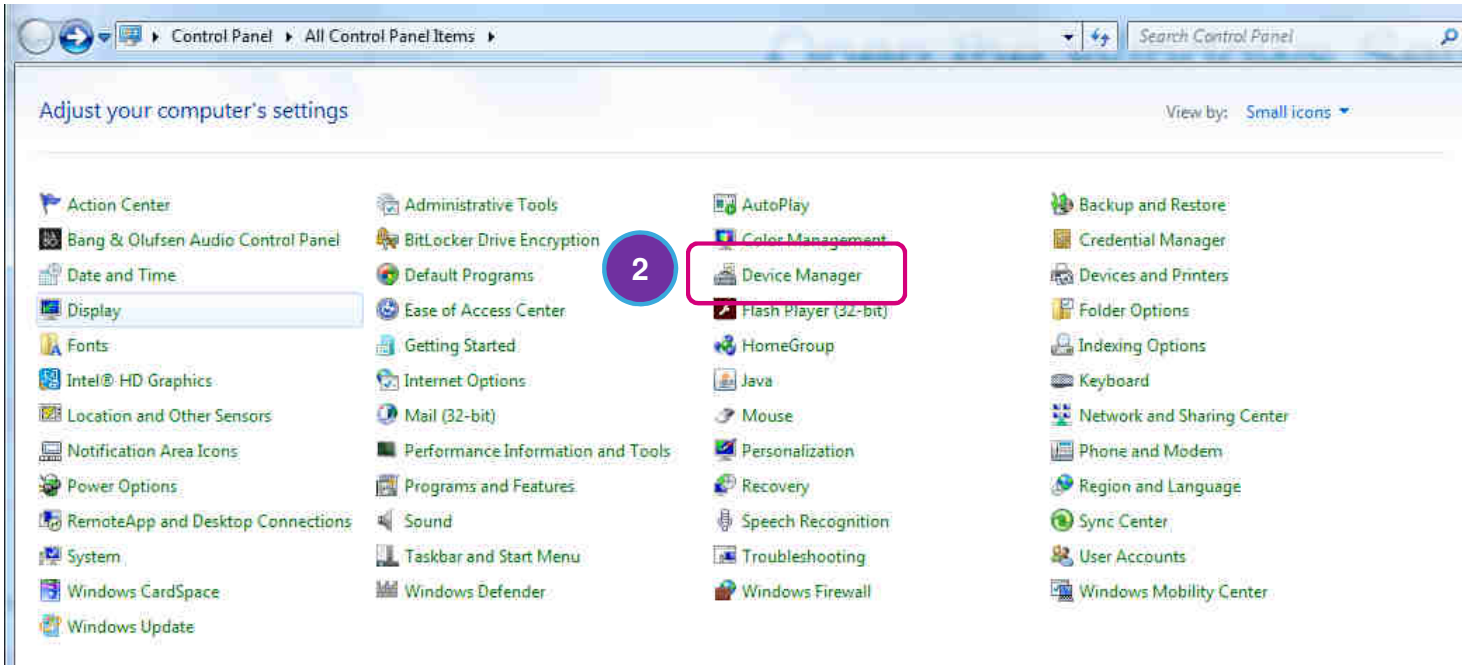
Virtual COM driver installation Win7

life.augmented

Open the Windows Control Panel

1. From **Start Menu** select **Control Panel**
2. Select **Device Manager**

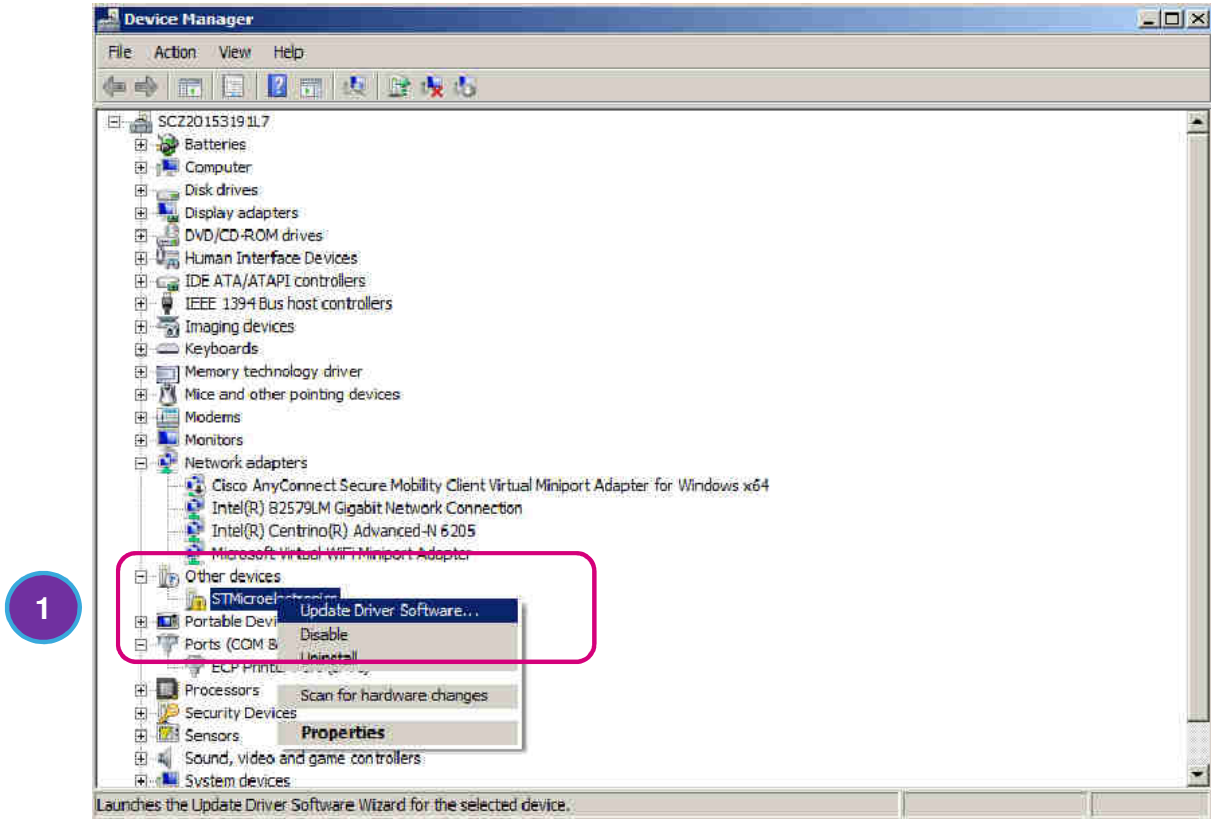




life.augmented

Open the Device Manager

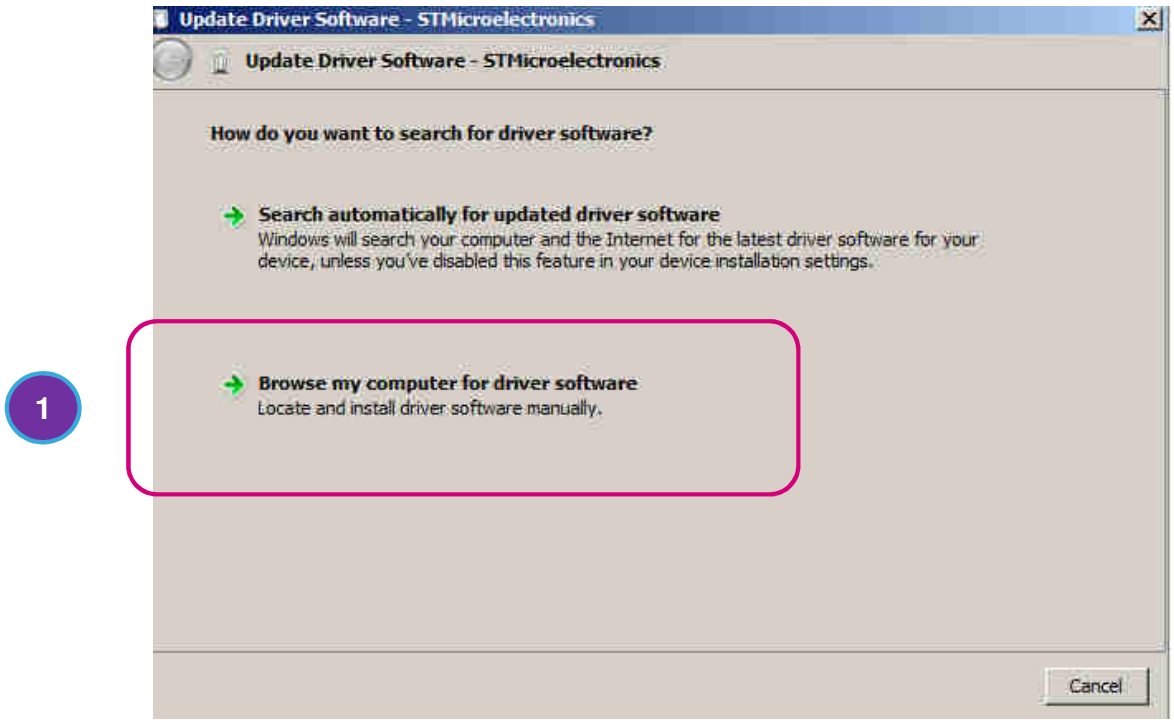
1. Look for **Other devices** and right click and then select **Update Driver Software…**



life.augmented

Look for the VCOM Driver

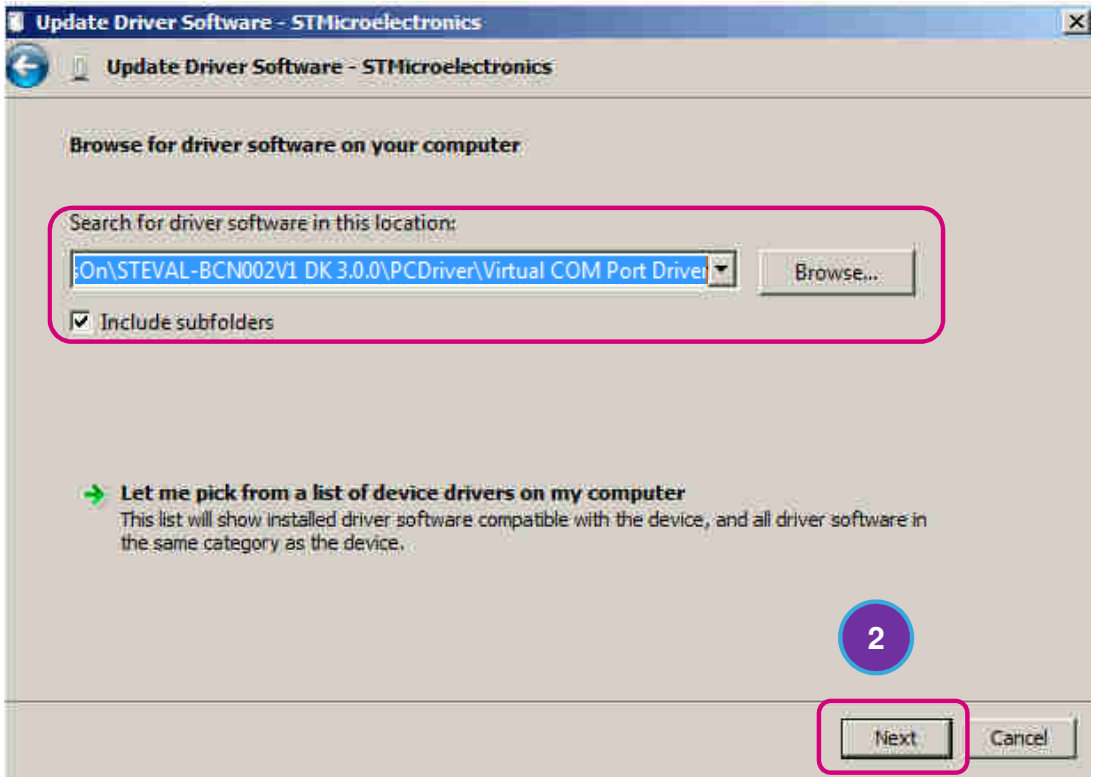
1. Select **Browse my computer for driver software**



Look for the VCOM Driver

1. Click on **Browse** button and **go to** the folder **"C:\BlueNRG_Tile_HandsOn\STEVAL-BCN002V1 DK 3.0.0\PCDriver\Virtual COM Port Driver"**

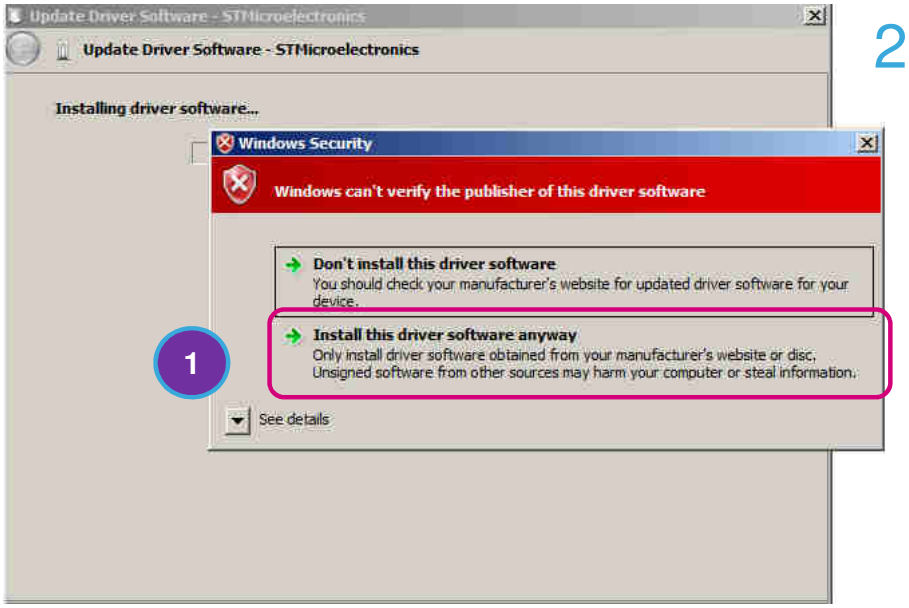
2. Click on **Include subfolder** and then on **Next**



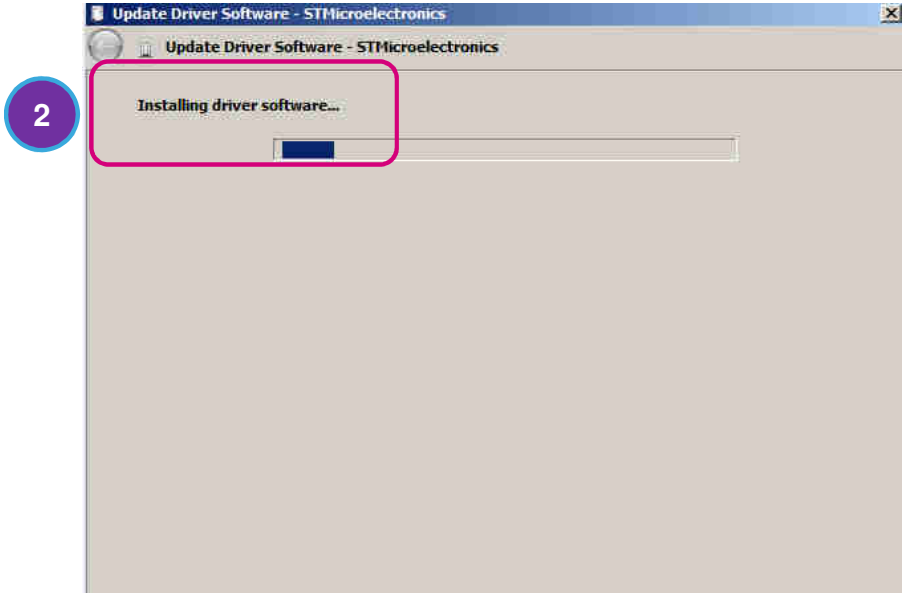
life.augmented

Allow the driver installation

1. Click on **Install driver software anyway**



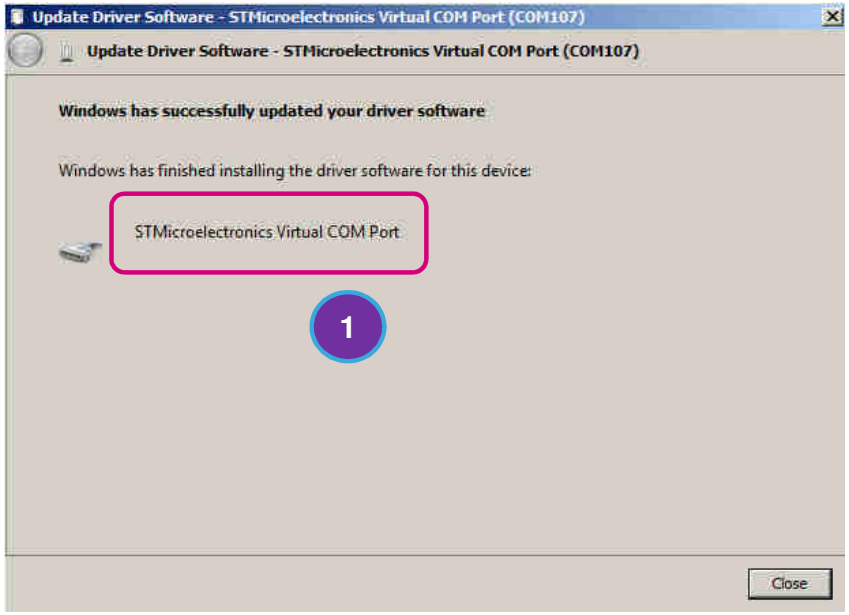
2. Installation starts



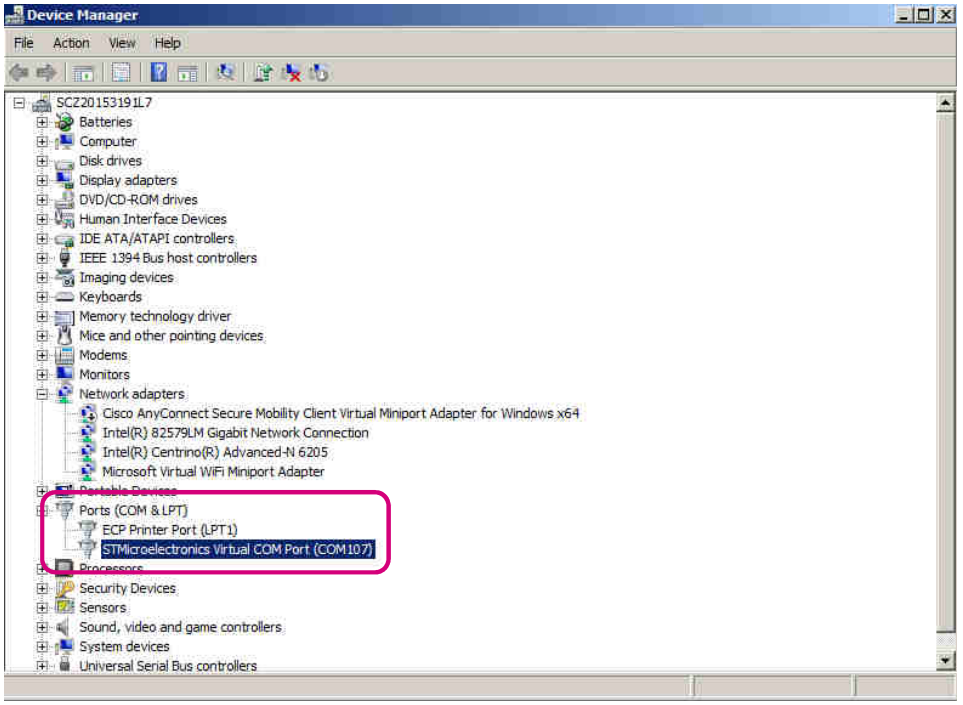
life.augmented

1. Installation completed

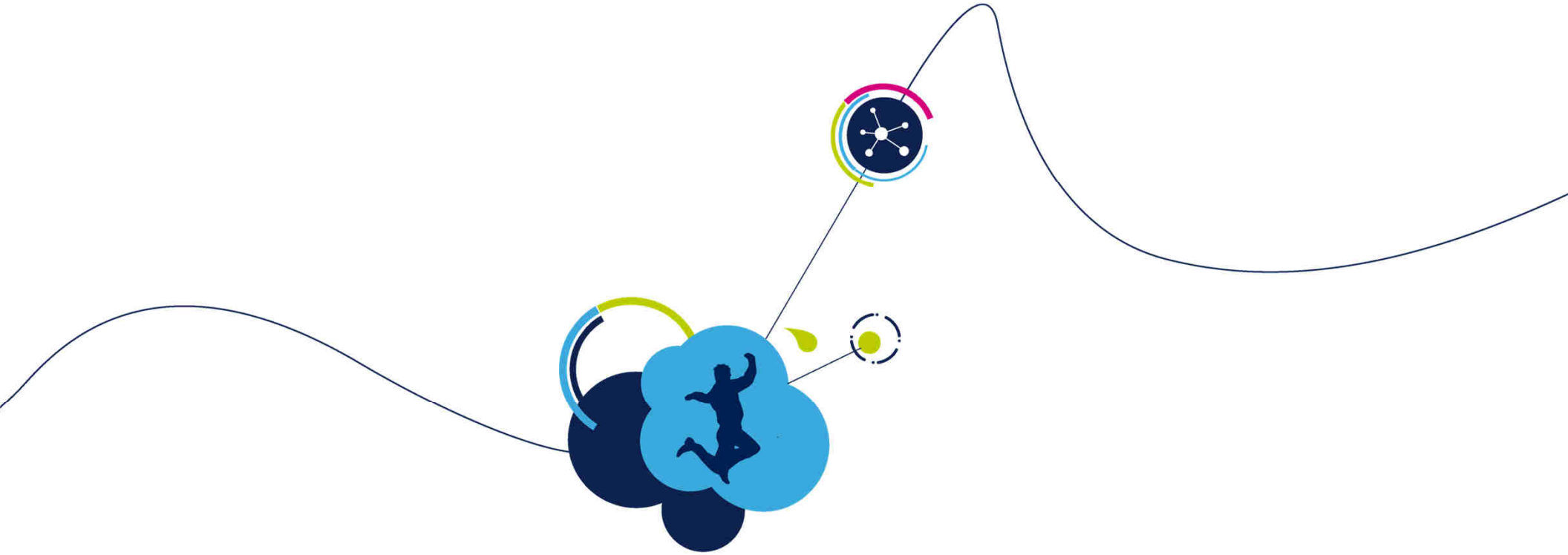
2. Device is in the **COM Ports** list



Done! Now please go back from here!



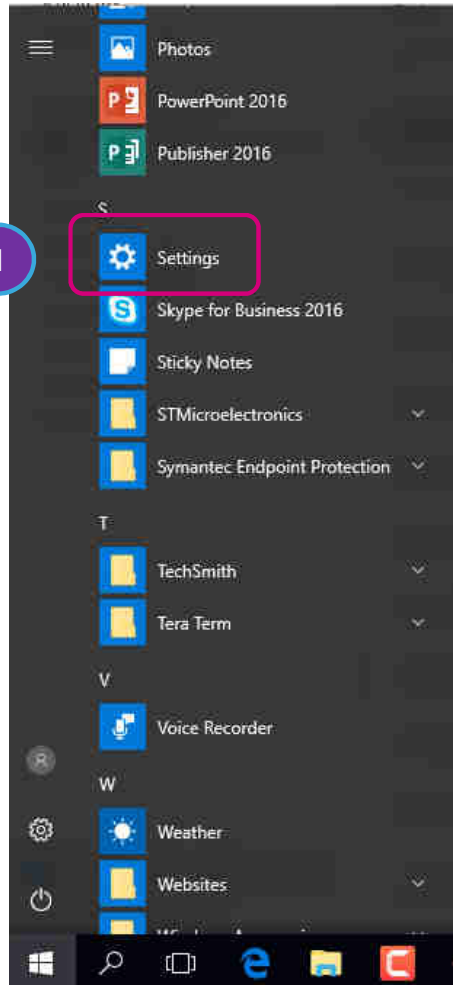
life.augmented



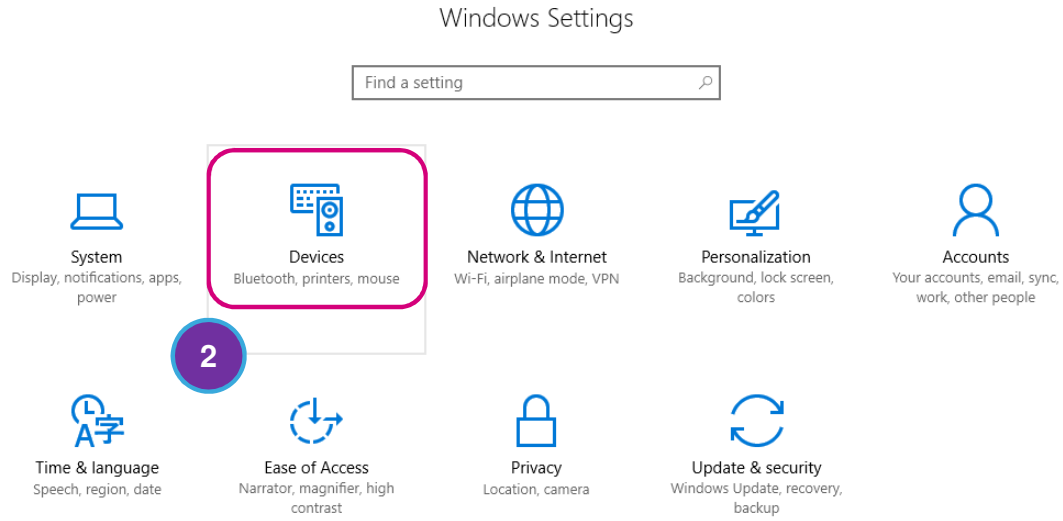
Virtual COM driver installation Win10

life.augmented

Open the Windows Settings



1. From **Start Menu** select **Settings**
2. Select **Devices**

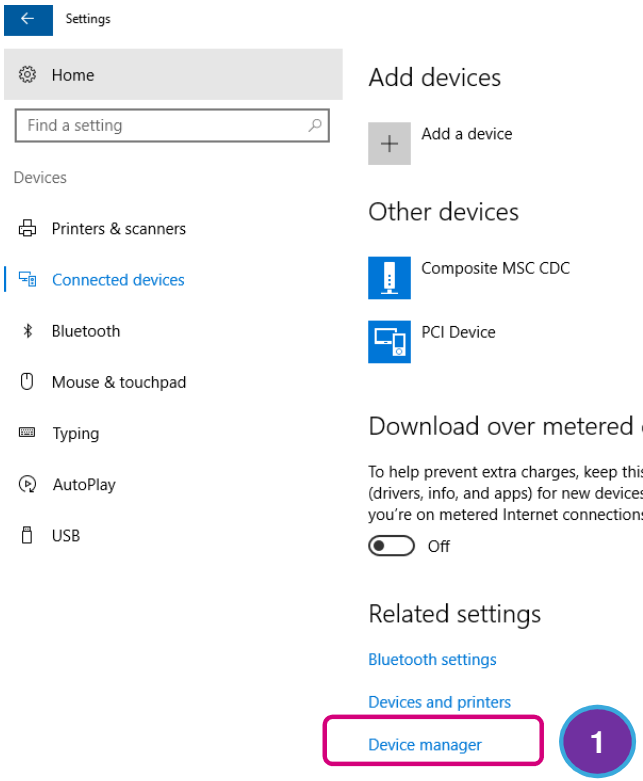


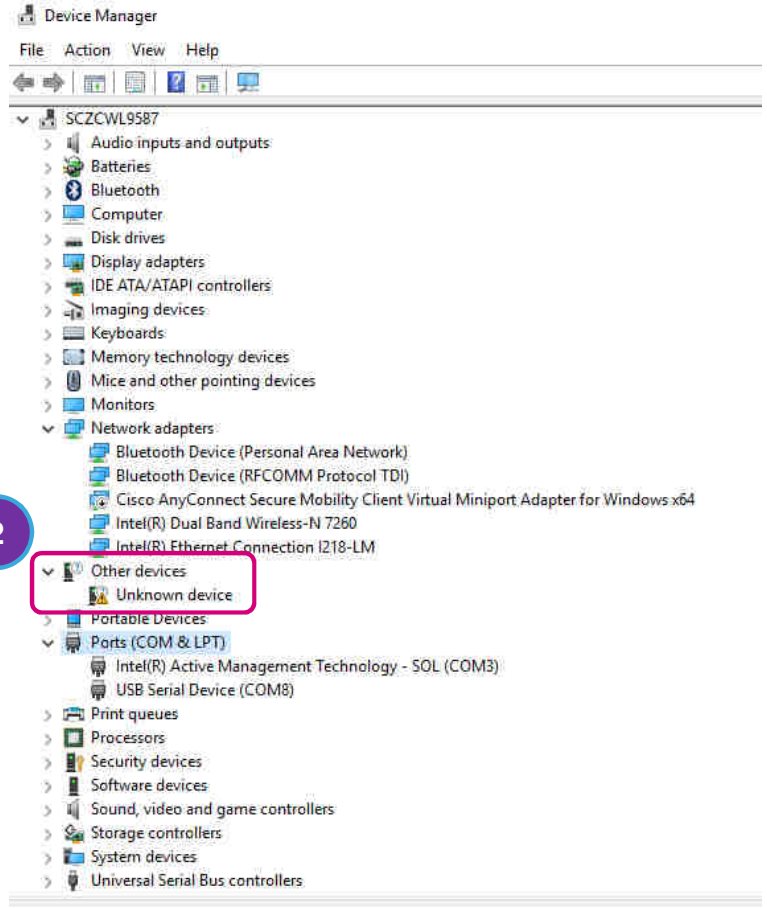
life.augmented

Open the Device Manager

1. Select **Device manager**

2. Select **Other Devices** and **Unknown device**



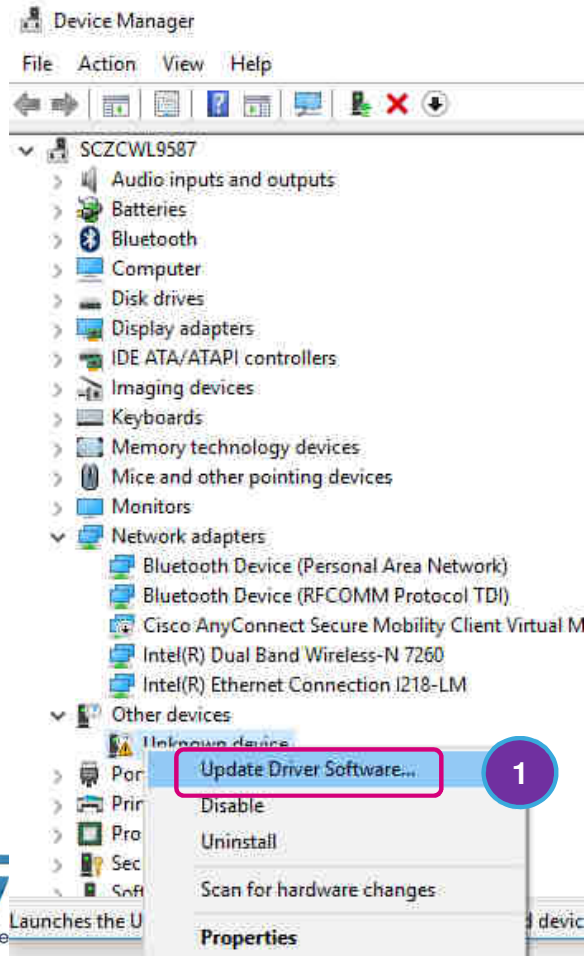


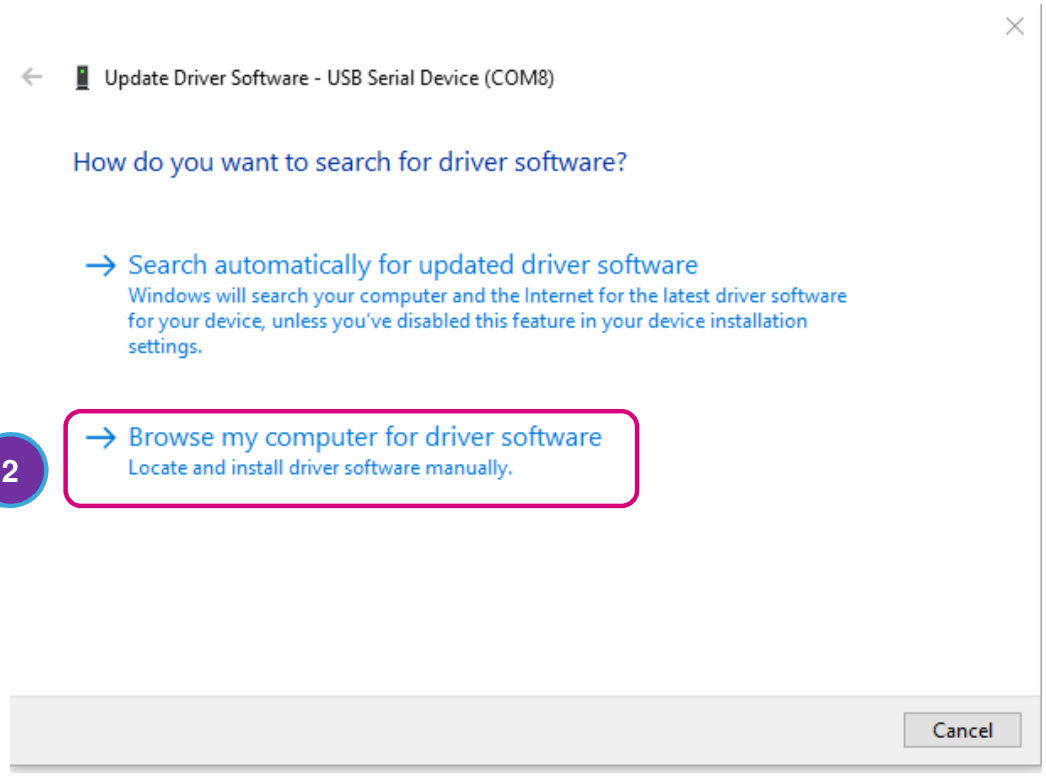
life.augmented

Update Driver Software

1. Right click and then select **Update Driver Software…**

2. Select **Browse my computer for driver software**



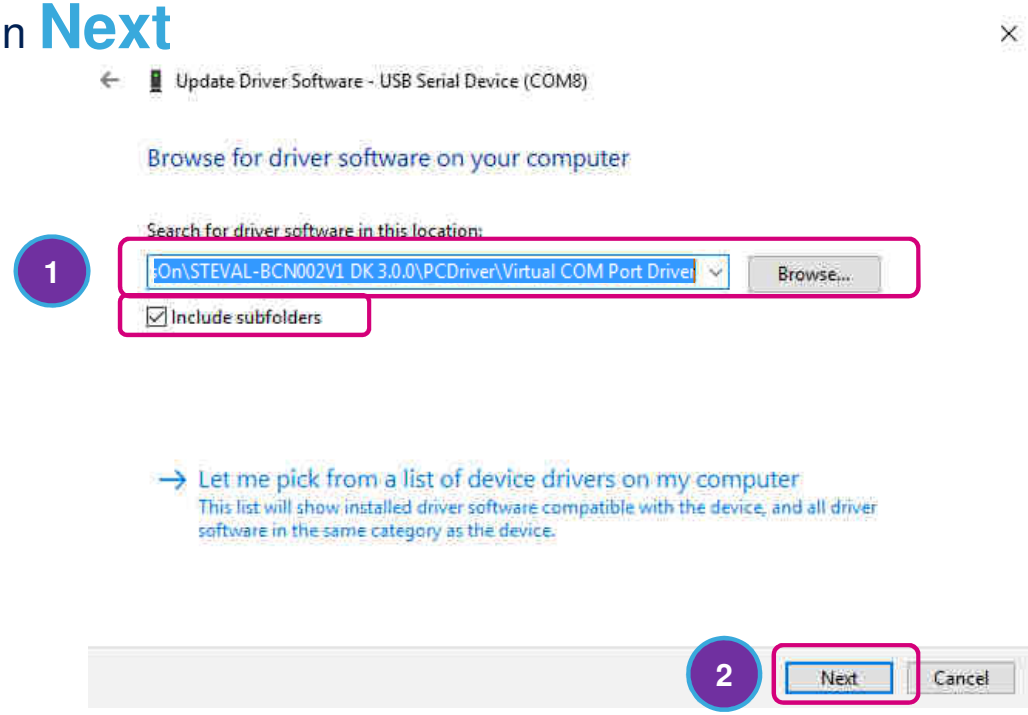


Look for the VCOM Driver

1. Click on **Browse** button and **go to** the folder

   “C:\BlueNRG_Tile_HandsOn\STEVAL-BCN002V1 DK 3.0.0\PCDriver\Virtual COM Port Driver”

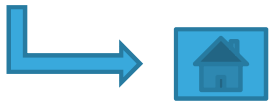
2. Click on **Include subfolder** and then on **Next**

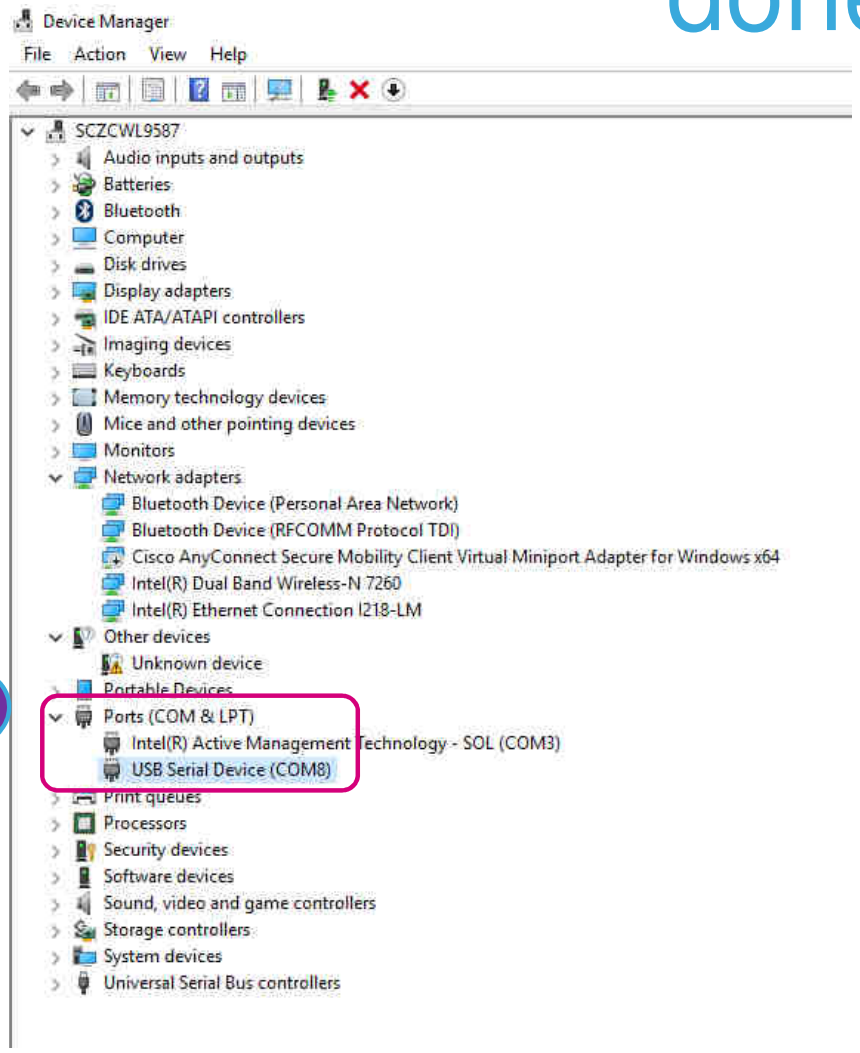


life.augmented

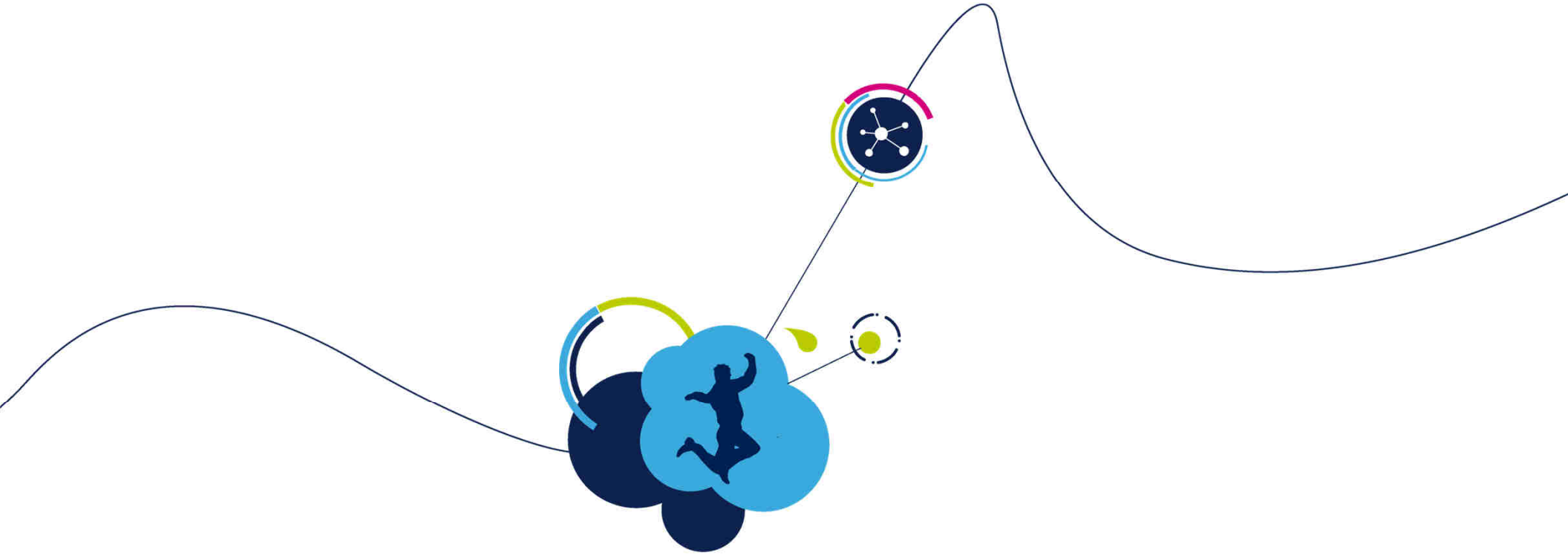
1. Installation completed. Device is in the **COM Ports** list

Done! Now please go back from here!





life.augmented



Keil MDK Free license installation steps

life.augmented

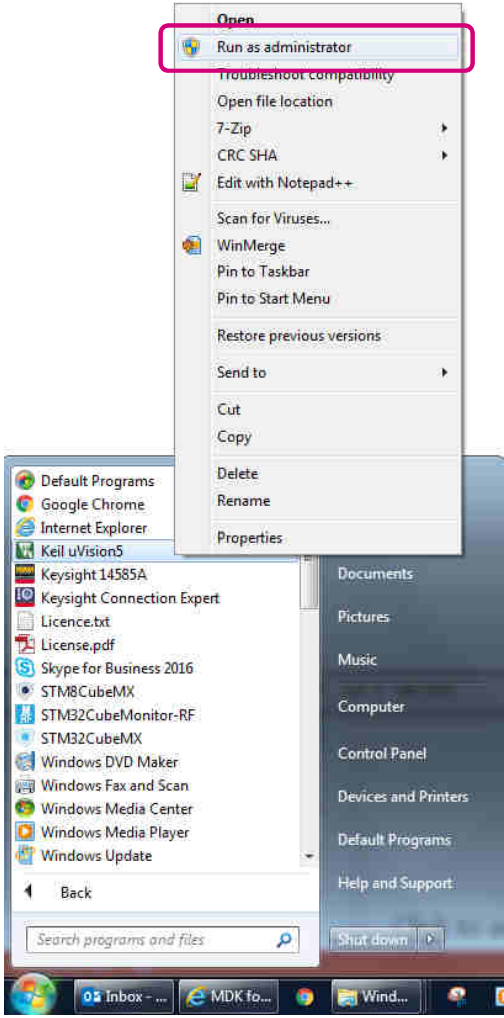
MDK for BlueNRG-2 and STM32F0 Installation & Activation

- You can refer to the following webpage:
  - http://www2.keil.com/stmicroelectronics-stm32/mdk

- Note: it is mentioned STM32L0 and STM32F0, but the same procedure applies to BlueNRG-2 device

Arm Keil MDK License Installation 1/6

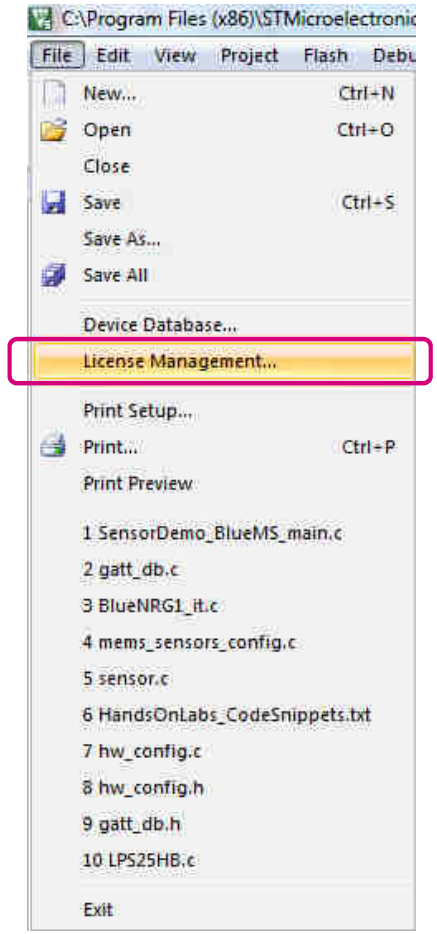
- Login with an account that has **administration rights**.
- Right-click the µVision icon and select **Run as Administrator...** from the context menu.



life.augmented

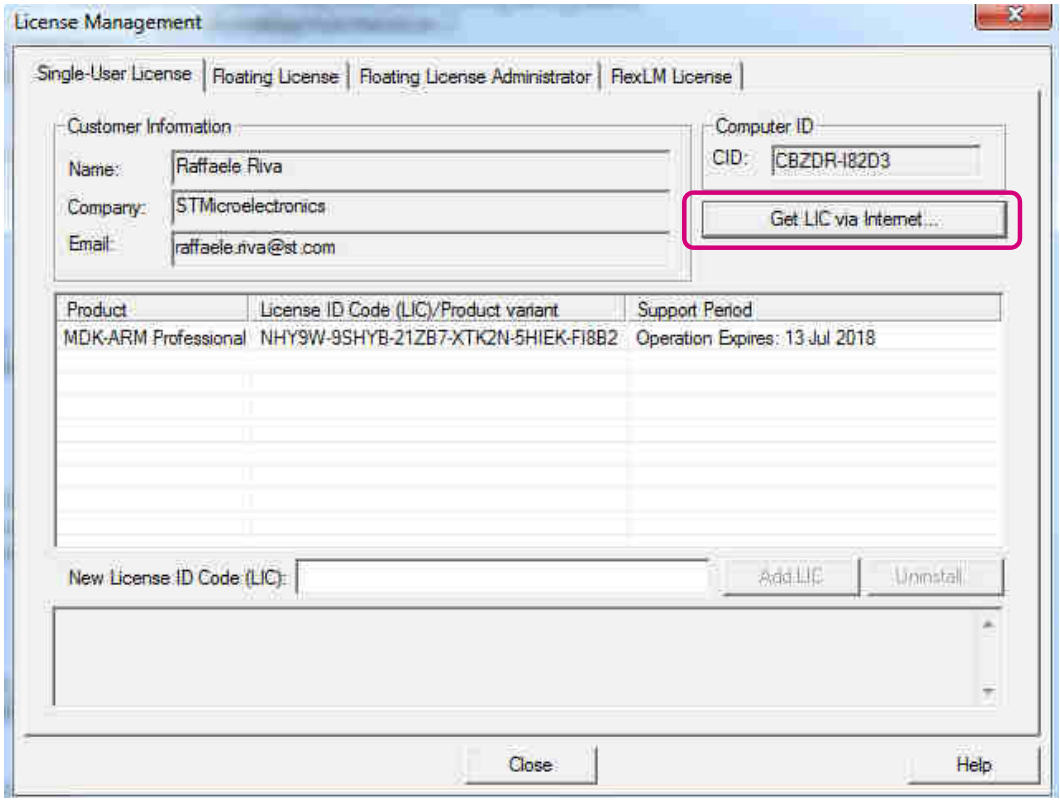
Arm Keil MDK License Installation 2/6

1. Open the dialog **File — License Management...**



Arm Keil MDK License Installation 3/6

Select the **Single-User License** tab and click the button **Get LIC via Internet...**,

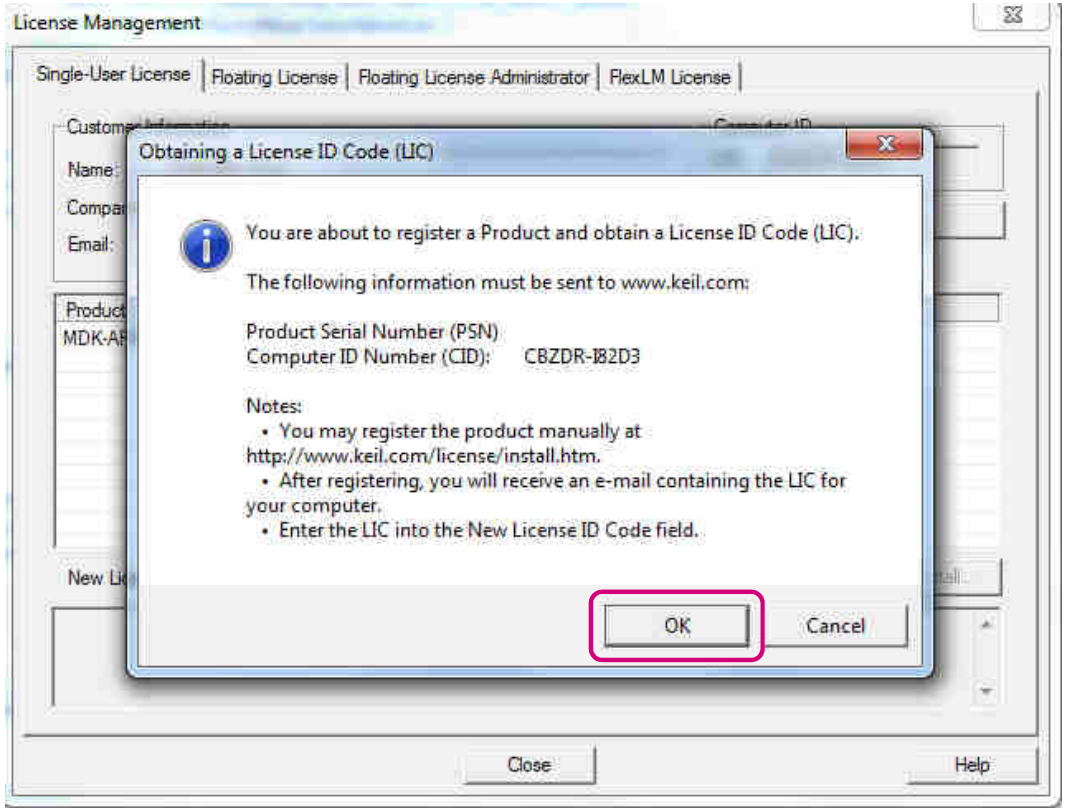


life.augmented

Arm Keil MDK License Installation 4/6

Click the button **OK** to register the product.

This action opens the License Management page on the Keil web site.

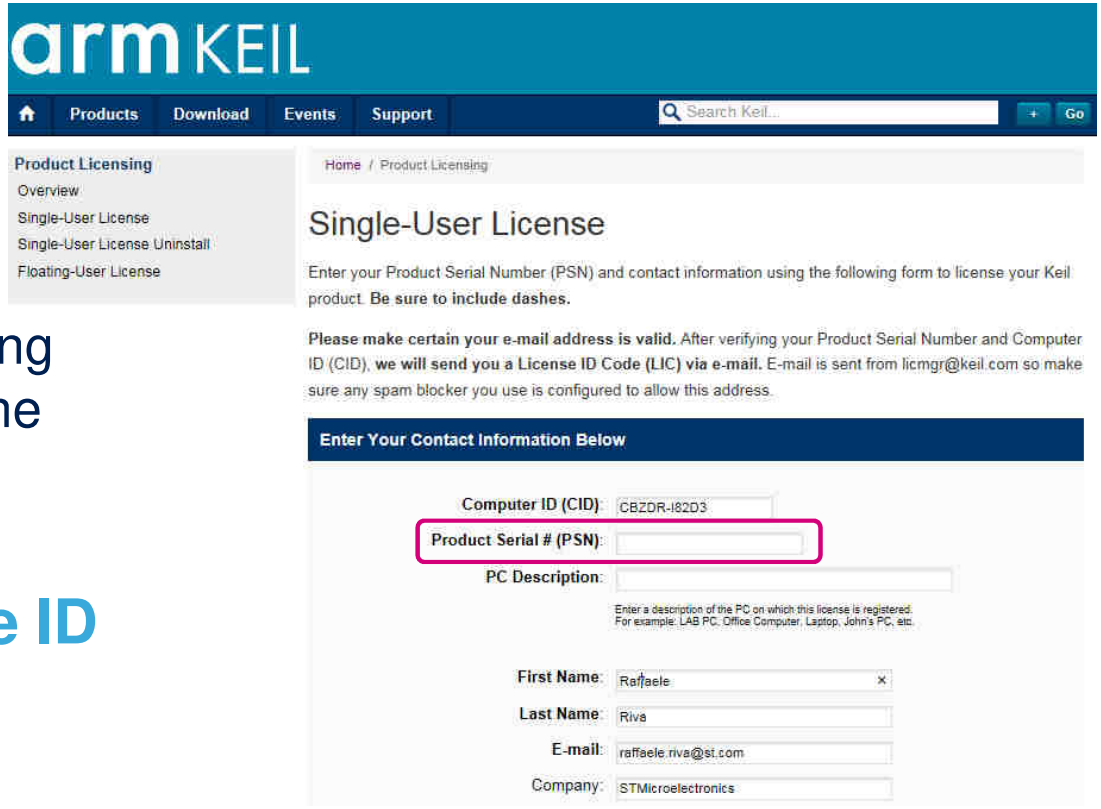


life.augmented

Arm Keil MDK License Installation 5/6

For the latest **Product Serial Number** please check on the Keil website at this URL:

http://www2.keil.com/stmicroelectronics-stm32/mdk



Then enter the PSN in this webpage along with your contact information and click the button **Submit**.

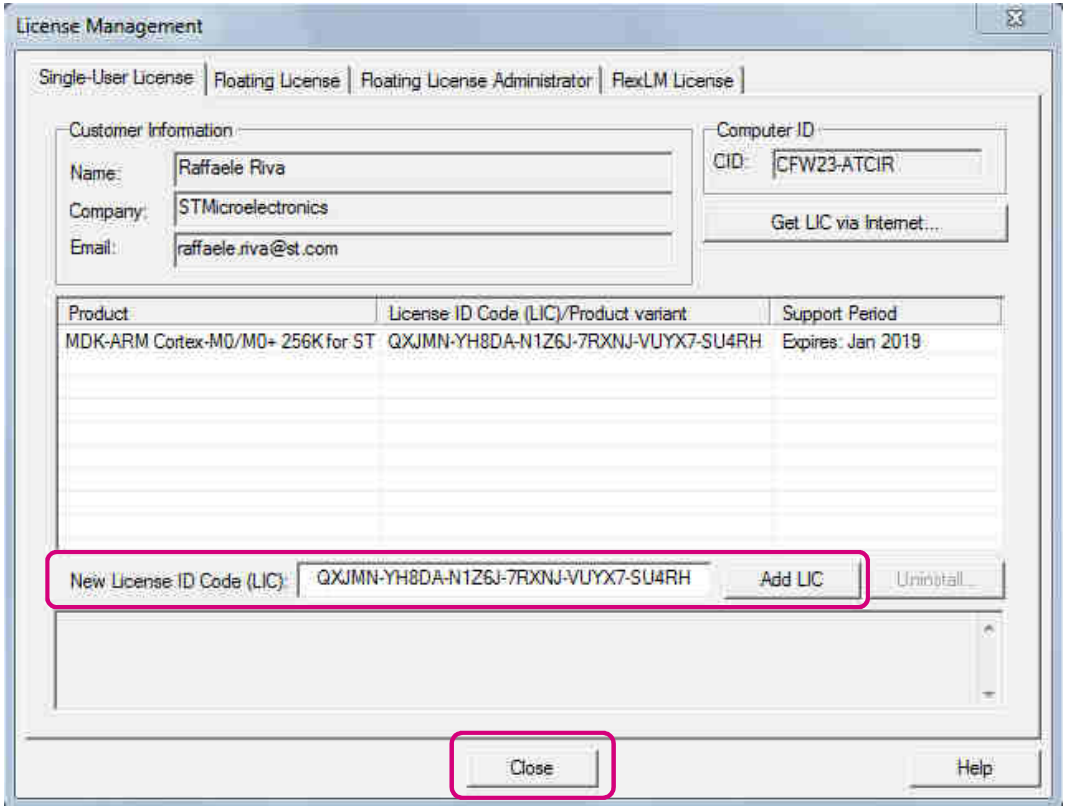
An e-mail is sent back with the **License ID Code (LIC)** within a few minutes.

life.augmented

Arm Keil MDK License Installation 6/6

To activate the Software Product, enter the **LIC** you received by email in the field **New License ID Code (LIC)** and click **Add LIC**.

Then click on **Close**



life.augmented