RN2483_Silica Documentation

Release 0

Silica

Mar 24, 2017

Contents

Version 1.40

Date 14 Mar 2017

Reference name BAEVTSS002

CHAPTER 1

Microchip’s Long Range Low Power End Node solution

Sensor Node Lora

CHAPTER 2

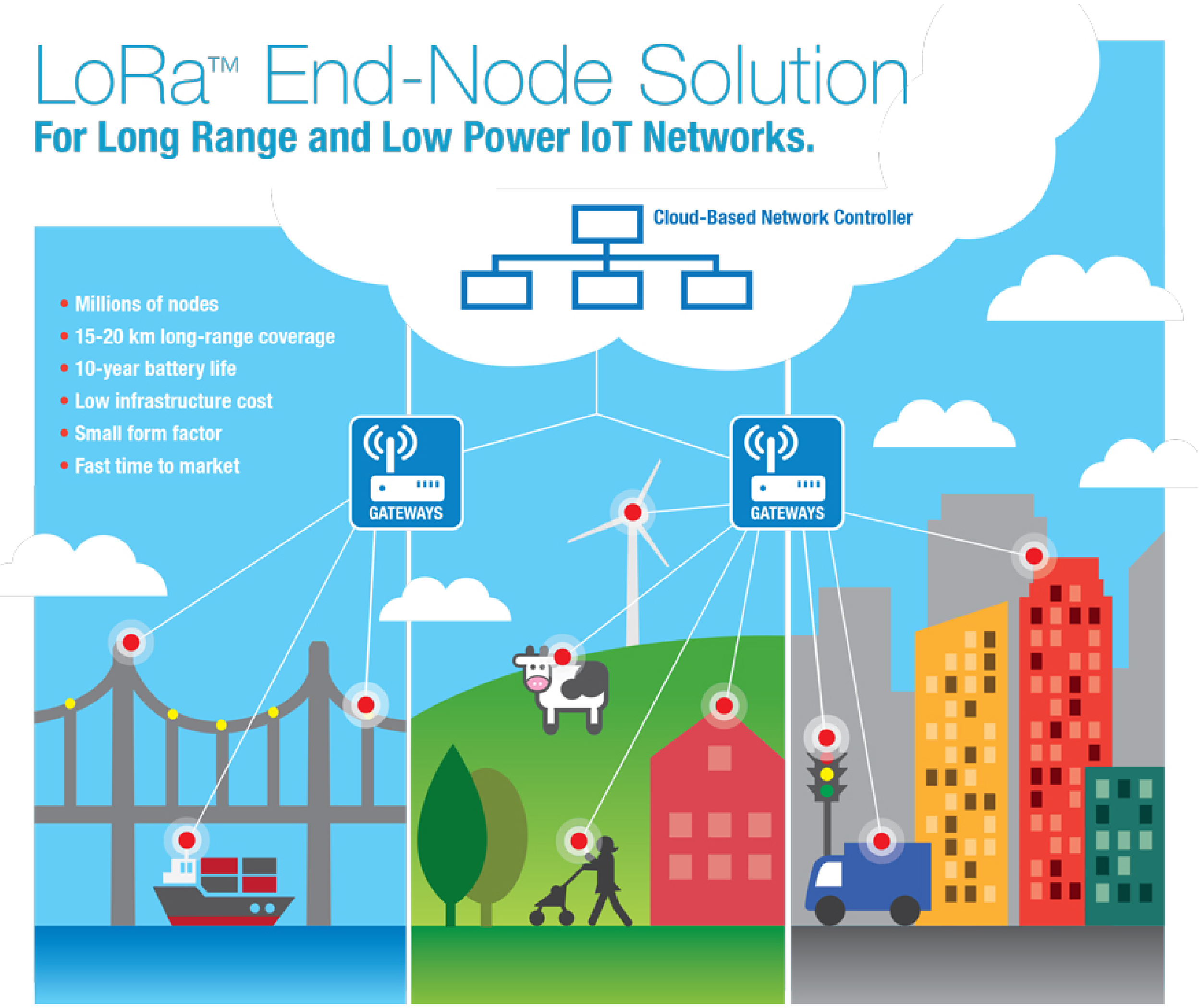
INTRODUCTION

Microchip's Long Range Low Power End Node solution

With the growing Internet of Things, Microchip has a LoRa® Technology wireless solution to address increasing demands on end-devices for long range connectivity, low power for battery operation, and low infrastructure cost for volume deployment.

Microchip's LoRa Technology solution is ready to run out-of-the box and with the complete LoRaWAN™ Protocol and certifications in place, it reduces time to market and saves development costs.

The RN2483 is a LoRa™-integrated modem with a range of more than 15 km (suburban), low power enabling a battery life greater than 10 years and the ability to connect millions of wireless sensor nodes to LoRa gateways and IoT-connected Cloud Servers. This robust system is due to LoRa's unique spread-spectrum based modulation that is capable of demodulation 20 dB below the noise level. This enables high sensitivity for ultra-long range, improved network efficiency and eliminates interference. The RN2483 modem operates over the 433 and 868 MHz license-free Industry Scientific and Medical (ISM) frequency bands and serves as the end-device in the LoRa network infrastructure.


LoRa™ End-Node Solution
For Long Range and Low Power IoT Networks.
Cloud-Based Network Controller
• Millions of nodes
• 15-20 km long-range coverage
• 10-year battery life
• Low infrastructure cost
• Small form factor
• Fast time to market
GATEWAYS
GATEWAYS

The RN2483 is a fully-certified 433/868 MHz module based on wireless LoRa® technology.

The module's embedded LoRaWAN™ Class A protocol enables seamless connectivity to any LoRaWAN compliant network infrastructure, whether public or privately deployed. The module is specifically designed for ease of use, which shortens development time and speeds time to market. LoRa technology is ideal for battery-operated sensors and low power applications such as IoT, M2M, Smart City, Sensor networks, Industrial automation, and more.

Features:

- On-board LoRaWAN™ Class A protocol stack
- ASCII command interface over UART
- Compact form factor 17.8 x 26.7 x 3 mm
- Castellated SMT pads for easy and reliable PCB mounting
- Device Firmware Upgrade (DFU) over UART
- 14 GPIO for control, status, and ADC
- Highly integrated module with MCU, crystal, EUI-64 Node Identity Serial EEPROM, Radio transceiver with analog front end, and matching circuitry

- Environmentally friendly, RoHS compliant
- European R&TTE Directive Assessed Radio Module

Developement tools

Firmware developed using: NXP Kinetis Design Studio. For installation and configuration of the project, follow instruction inside *Quick start guide for firmware 1.6 or major*

Document references

The board reference documentation is available on the Avnet Silica website.

Contents:

Quick start guide for firmware 1.6 or major

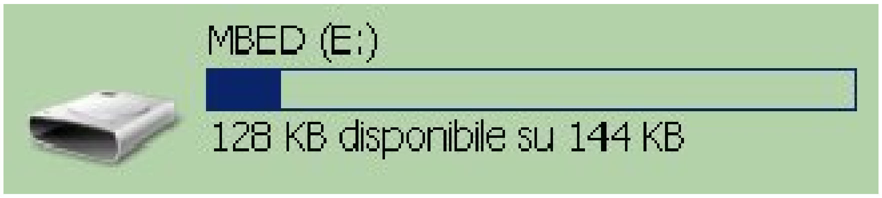
In this guide you will be able to upload your board with the last firmware and join the Lora Network in **ABP** or **OTAA** mode.

Update firmware

If you are unsure about which firmware version you have then update your board with the lastest firmware avaiable.

1. Download the file Lora_v16.bin.zip and unzip it.
2. Connect the board to the PC via Mini Usb cable.

3. Go to the mbed site.
4. Download the lastest mbed serial port driver, it is named as **mbedWinSerial**.
5. Install the driver.
6. When the driver is installed, a MBED mass storage device and a Serial line and a HID device are finded out. Open the MBED mass storage.



MBED (E:)

128 KB disponibile su 144 KB

> **Warning:** If the window opened is named **BOOTLOADER** repeat the procedure, the window **must be named MBED**.

7. Copy the file **Lora_v16b.bin** into the MBED mass storage.
8. The Green led on the board will blink while the file is programmed into the device flash memory.

> **Warning:** If accidentally has been copied the firmware in the **BOOTLOADER** window you have corrupted the MBED firmware and you have to fix it. Please read the fix procedure going to the page *How to fix the MBED firmware*

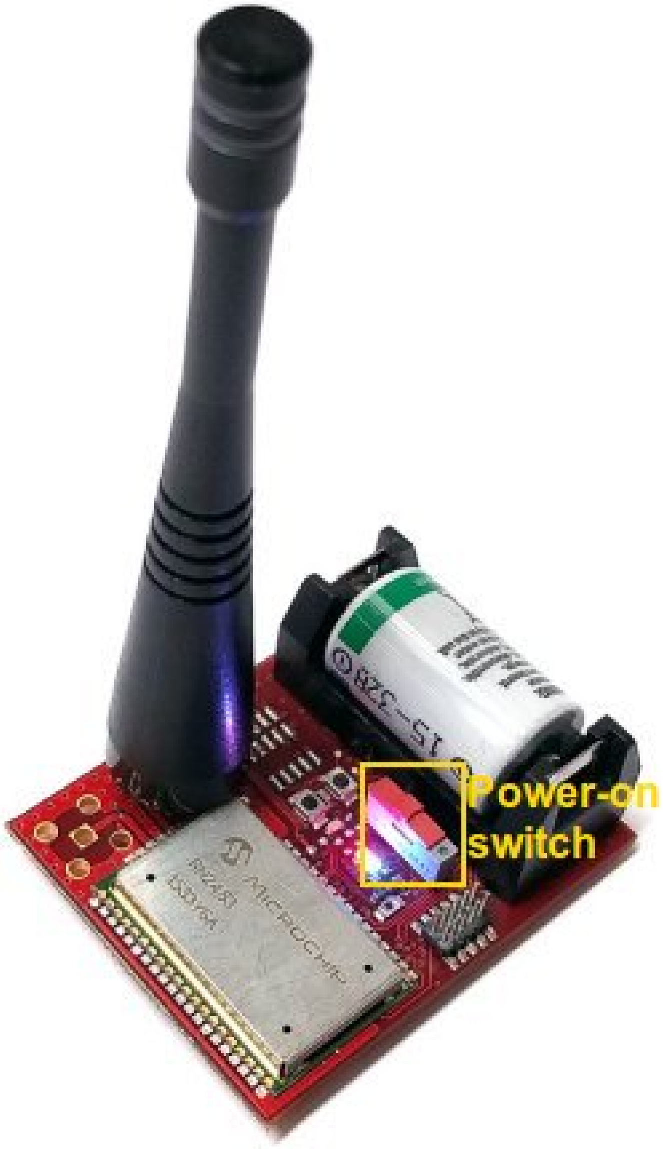
Setup Module

Now you have to choose which kind of connection you want **OTAA** or **ABP**.

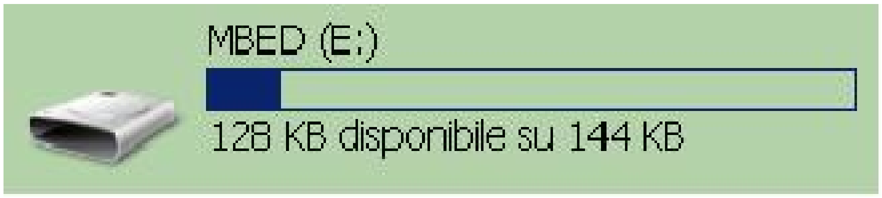
- Over-the-Air Activation (OTAA) is the most secure way to connect the device. The module will perform a join-procedure with the network, during which a **dynamic DevAddr** is assigned and security keys are negotiated with the device.
- The Activation-by-Personalization permit you to hardcode the **DevAddr** as well as the security keys in the module. This way is simpler than OTAA, because you skip the join procedure, but it is less safer than the OTAA.

You have to setup the board in order to send correctly messages to your operator. Follow next steps correctly:

1. Insert antenna and Micro-USB from the windows PC to the board and put the board on the table.
2. If you did not installed the mbed serial driver then go to mbed website otherwise go to the step **5**
3. Download the lastest mbed serial port driver, it is named as **mbedWinSerial**.
4. Install the driver.
5. Now switch on the Lora Sensor Node.



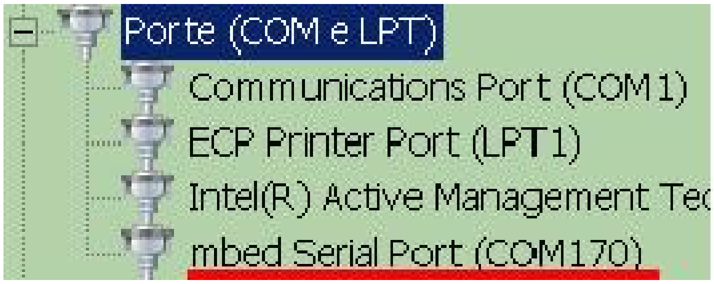
6. If the MBED Windows serial port driver is installed correctly Windows will recognize the board as a memory storage called MBED and as a Serial Com Port.



MBED (E:)

128 KB disponibile su 144 KB

7. Now we have to connect a terminal to the Serial Com. First step, find which number is, going to “Device Manager” and finding the node called “Ports (COM and LPT)”. In this example the number port is 170.



Porte (COM e LPT)

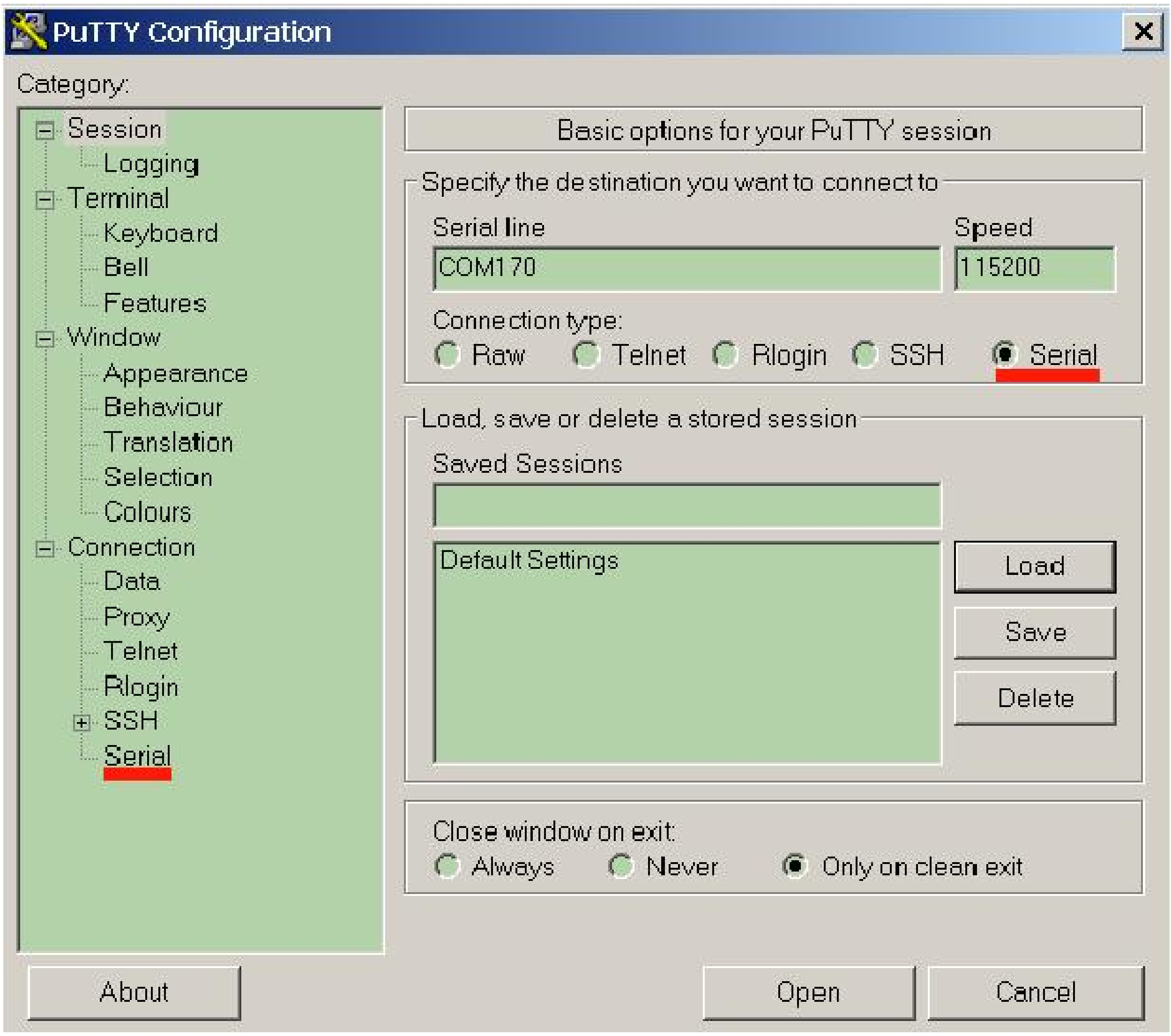
Communications Port (COM1)

ECP Printer Port (LPT1)

Intel(R) Active Management Tec

mbed Serial Port (COM170)

8. Open a terminal like **putty**, you can find it here. Launch it and set the properties signed in red in the figures:



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM170

Speed: 115200

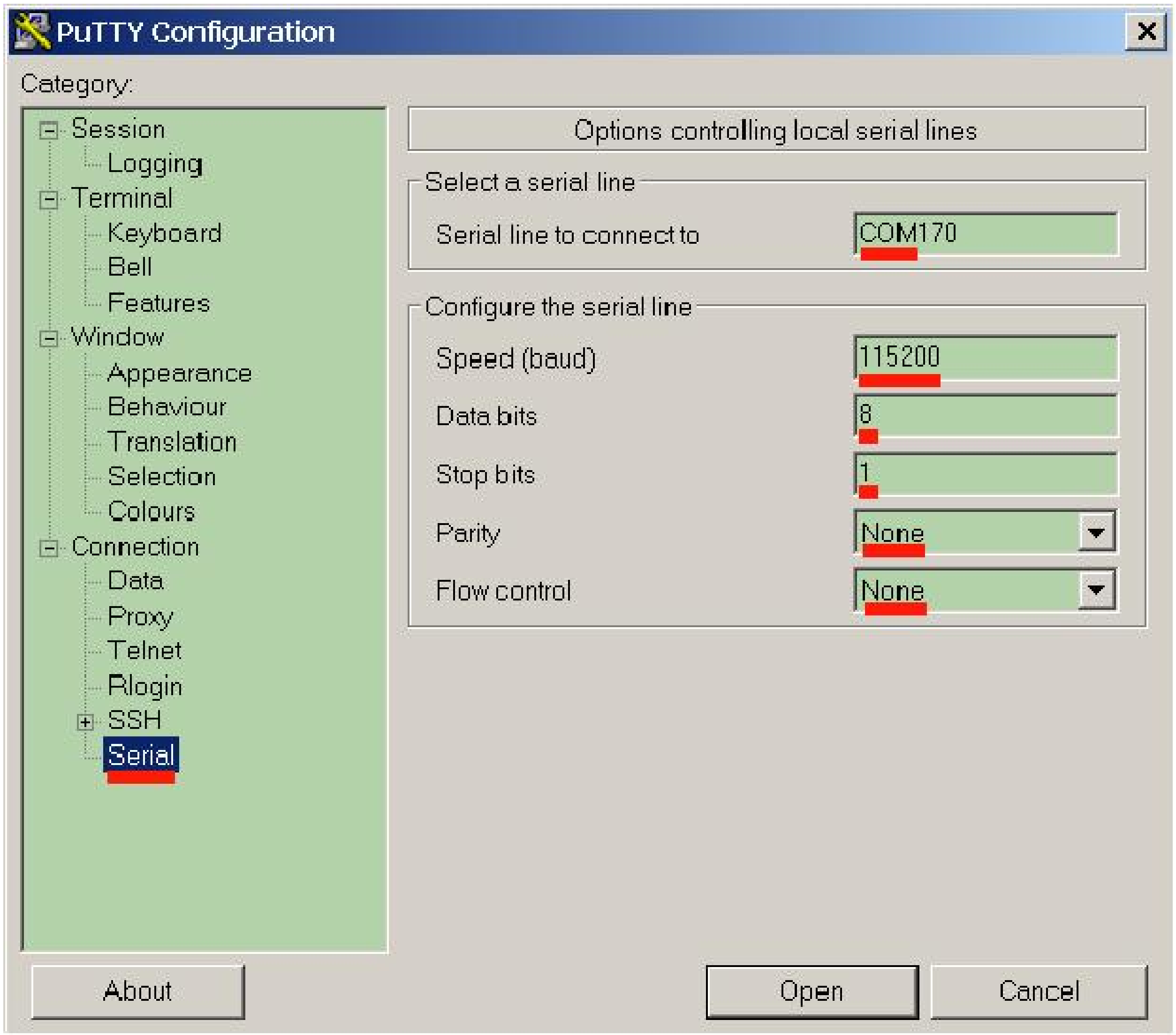
Connection type: ○ Raw ○ Telnet ○ Rlogin ○ SSH ● Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit



Serial line to connect to: COM170

Speed (baud): 115200

Data bits: 8

Stop bits: 1

Parity: None

Flow control: None

9. Press the **Open** button. Now reset the board pressing **S3** button
10. On the putty console you will see the booting text:

```
Module Version: BAEVTSS002 firmware version 1.6
RN2483 1.0.1 Dec 15 2015 09:38:06
```

11. After that you have 5 seconds in order to press **S2** button. Doing so it is possible enter in a menu where you can run special commands.
12. From this menu you will be able to setup your board:

```
Send commands to RN2483. Write 'quit' to exit.
    Enter 1 > Write/Read REGS
    Enter 2 > RESET Module
    Enter 3 > Send Payload
    Enter 4 > Advanced Settings
```

Now if you want to use **OTAA** go to the paragraph *OTAA Configuration* otherwise if you want to use **ABP** go to *ABP Configuration*

OTAA Configuration

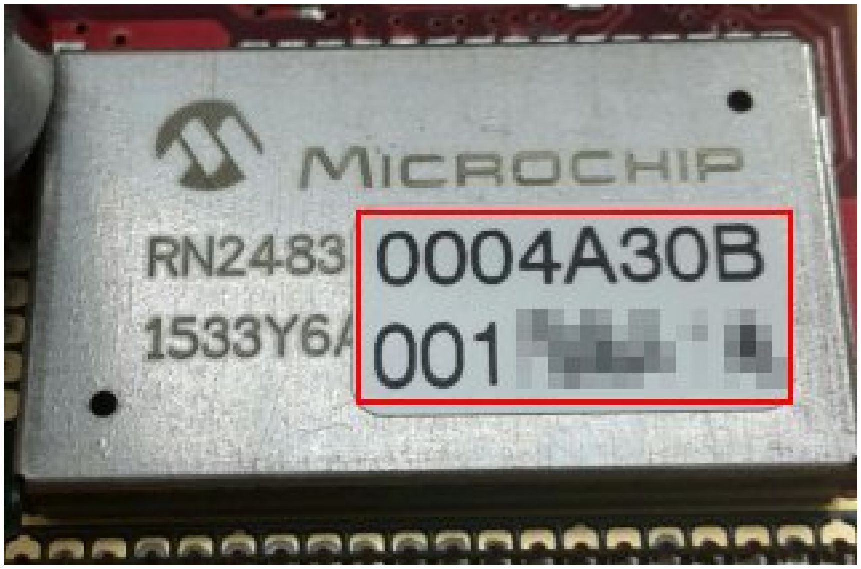
The OTAA configuration requires the following parameters:

- **DevEUI**: 64 bit end-device identifier. It is unique for every device.
- **AppEUI**: 64 bit application identifier. It is unique for every device.
- **AppKey**: 64 bit application identifier. It is unique for every device.

1. From the main menu press '**1**' you will select "**Write/Read REGS**". Will be showed this menu:

```
Enter 1 > set DEVADDR (ABP mandatory)
Enter 2 > set NWKSKEY (ABP mandatory)
Enter 3 > set APPSKEY (ABP optional)
Enter 4 > set DEVEUI (OTAA and ABP mandatory)
Enter 5 > set APPEUI (OTAA mandatory)
Enter 6 > set APPKEY (OTAA mandatory)
Enter 7 > Save Changes
Enter 8 < back
```

2. Enter **4** to insert **DevEUI** address.
3. Insert **16** hexadecimal numbers. You can read the last numbers on the label of the board as in the image.



4. If it succeed will apper *value changed*
5. Now you have to program the **AppEUI** pressing the key **5**
6. Insert **16** hexadecimal numbers. For example **0123456789ABCDEF**.
7. If it succeed will apper *value changed*
8. The last value to insert is the APPKEY pressing the key **6**
9. Insert **32** hexadecimal numbers. For example **0123456789ABCDEF0123456789ABCDEF**.
10. If it succeed will apper *value changed*
11. Now you have to save these changes pressing the key **7**

Once the board is configurated correctly you have to register the device to your provider. In this guide we used Thingpark. To follow the next steps go to *Register Module*.

ABP Configuration

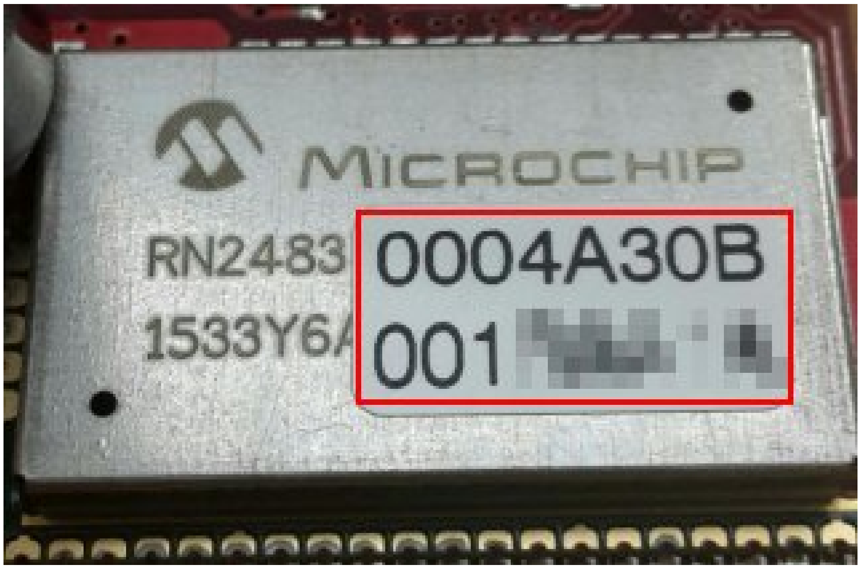
The ABP configuration requires the following parameters:

- **DevAddr**: 32 bit device address (non-unique)
- **NwkSKey**: 64 bit register, it is used for network layer security
- **DevEUI**: 64 bit end-device identifier. It is unique for every device

1. From the main menu press ‘**1**‘ you will select “**Write/Read REGS**”. Will be showed this menu:

```
Enter 1 > set DEVADDR (ABP mandatory)
Enter 2 > set NWKSKEY (ABP mandatory)
Enter 3 > set APPSKEY (ABP optional)
Enter 4 > set DEVEUI (OTAA and ABP mandatory)
Enter 5 > set APPEUI (OTAA mandatory)
Enter 6 > set APPKEY (OTAA mandatory)
Enter 7 > Save Changes
Enter 8 < back
```

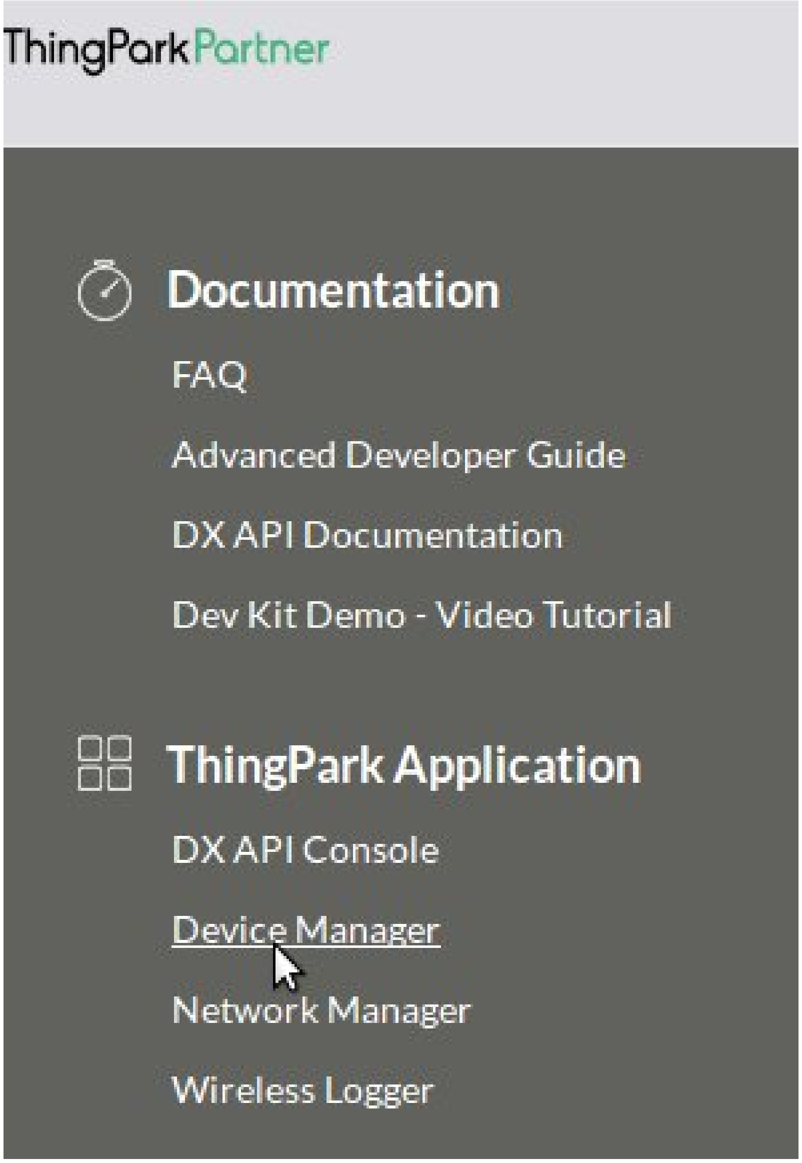
2. Enter **1** to insert **DevAddr** address.
3. Insert **8** hexadecimal numbers. You can read the last numbers on the label of the board as in the image.



4. If it succeed will apper *value changed*
5. Now you have to program the **NwkSKey** pressing the key **2**
6. Insert **32** hexadecimal numbers. For example **0123456789ABCDEF0123456789ABCDEF**.
7. If it succeed will apper *value changed*
8. The last value to insert is the **DevEUI** pressing the key **4**
9. Insert **32** hexadecimal numbers. For example **12345678901234567890123456789012**.
10. If it succeed will apper *value changed*
11. Now you have to save these changes pressing the key **7**
12. Once the board is configurated correctly you have to register the device to your provider. In this guide we used Thingpark. To follow the next steps go to *Register Module*.

Register Module

1. In order to see working the Sensor Node Lora you have to register the device on the activity website clicking on **Try**.
2. After the registration login and you will access the main page.
3. Register your Microchip RN2483 device, to do this, click on **Device Manager** to open a new window.



ThingPark Partner

Documentation

FAQ

Advanced Developer Guide

DX API Documentation

Dev Kit Demo - Video Tutorial

ThingPark Application

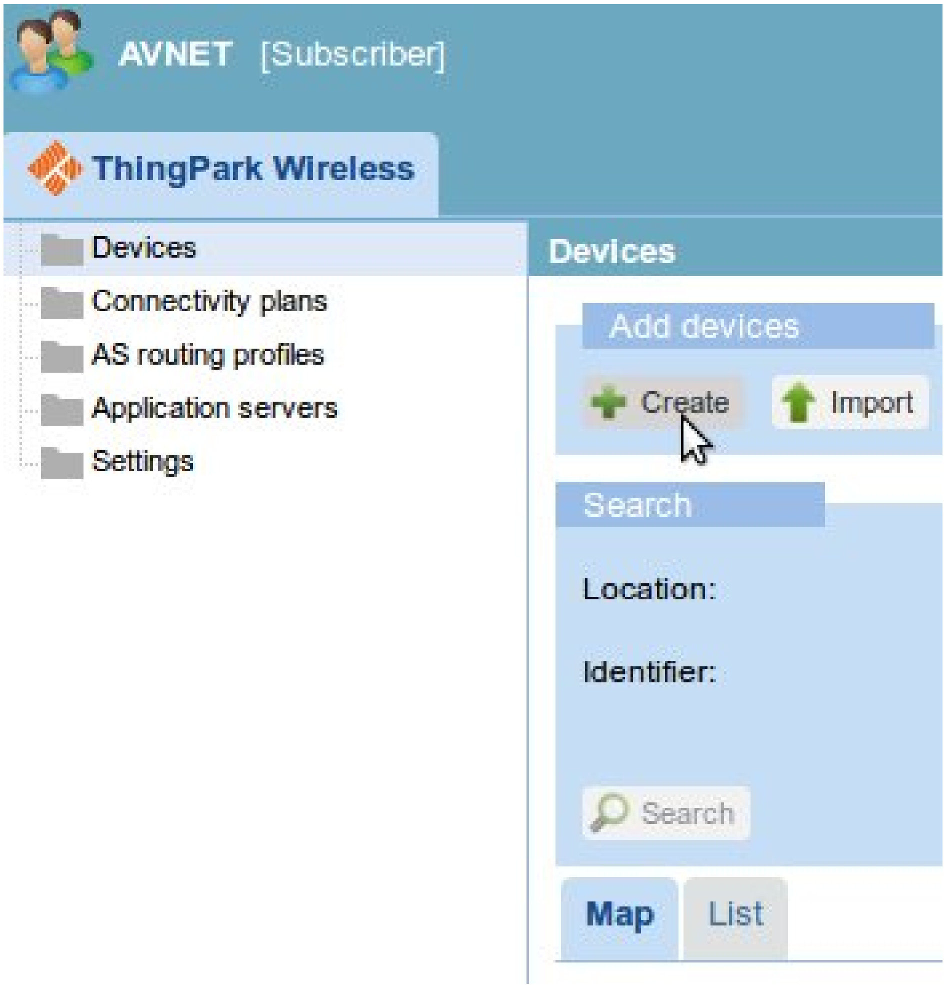
DX API Console

Device Manager

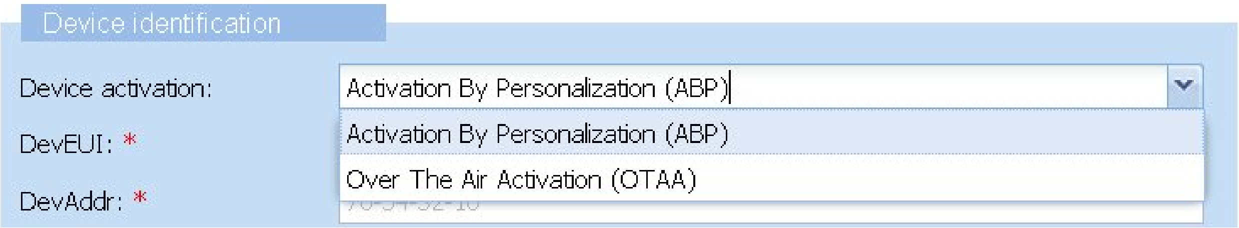
Network Manager

Wireless Logger

4. Click on the **+ Create** button.



5. In this form you have to insert the data we have stored in the RN2483. You have to choose if to use ABP or OTAA mode in the first field named **Device activation** of the **Device identification** group.


Device identification
Activation By Personalization (ABP)
Over The Air Activation (OTAA)
DevEUI: *
DevAddr: *

Device activation: Activation By Personalization (ABP)

6. Insert in every field the values wrote before in the device (*OTAA Configuration* or *ABP Configuration* paragraph).

7. Make sure to define as **Device profile**: **LoRaWAN 1.0 - class A - ETSI**

8. Select a valid **Connection plan**

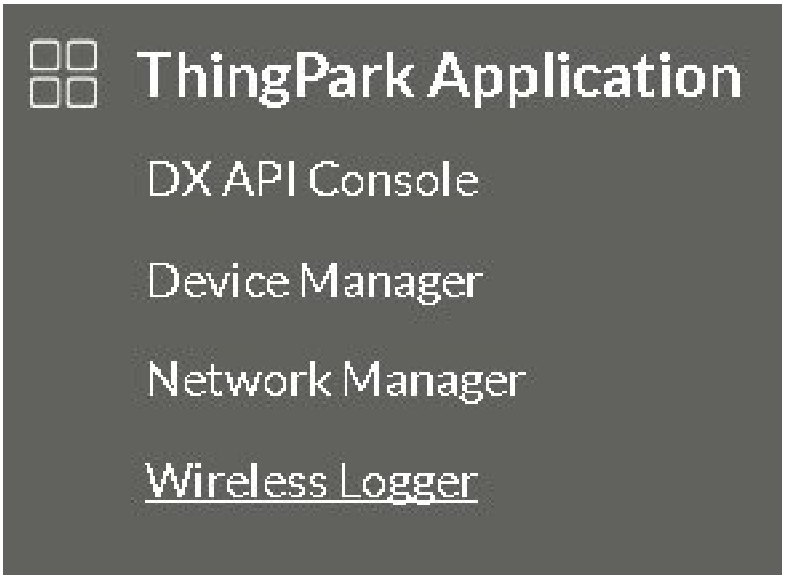
Optional field but useful:

- **Name**: insert a name just to recognize your device.

6. Click on the top right **+ create**.

Warning: If you don't enter the same keys wrote in the device you will be unable to send data correctly in the Lora network.

7. After the registration, you can close the window device manager and on the main page go to **Logger**.



8. In this page you will see all the messages sent by your device.

Use your board

17. From the **main menu** of the device you have to select which mode will be used in the demo. You can have these options:

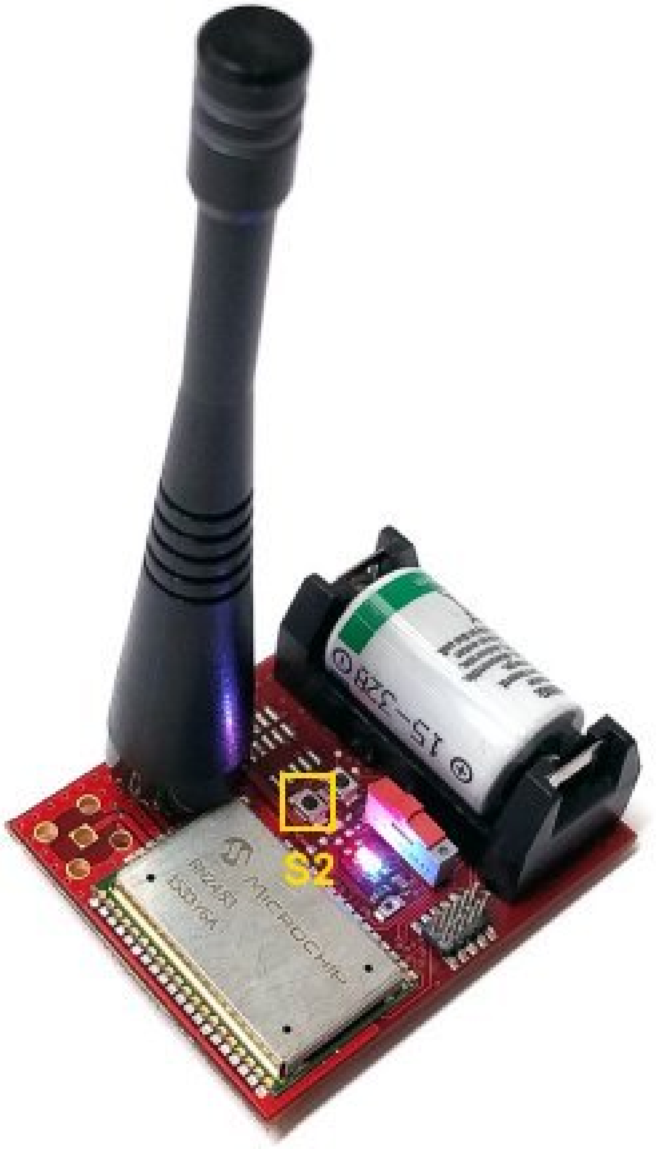
```
Enter 5 > Change mode Demo to OTAA join
Enter 6 < Start Demo in ABP mode
```

or

```
Enter 5 > Change mode Demo to ABP join
Enter 6 < Start Demo in OTAA mode
```

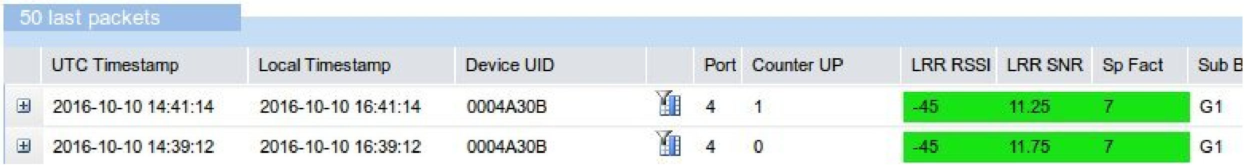
Select the correct one, your choose will be stored in flash memory and used every time you will turn-on the board. Then start the demo, the device will enter automatically in **sleep mode**.

18. Every 2 minutes it will wake up for a short time in order to send a message. It is possible skipping the sleeping phase pressing the **S2** button. When the board is sending a message a blue led will be turned on.



19. After pressing **S2** button you will see the first message sent by your device. Now tilt your board to 90 degrees and wait 30 seconds.

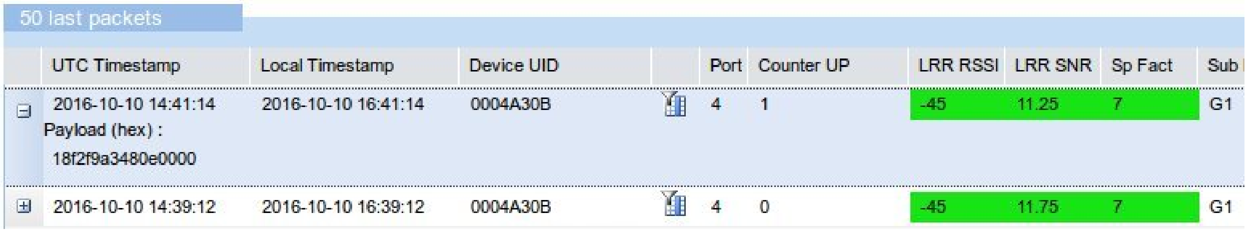
20. The board will send another message.
21. Now it's time to see the data sent. Power off the board.
22. In the logger window, you will have 2 rows, every row is a message received from the server.



50 last packets

	UTC Timestamp	Local Timestamp	Device UID		Port	Counter UP	LRR RSSI	LRR SNR	Sp Fact	Sub B
⊞	2016-10-10 14:41:14	2016-10-10 16:41:14	0004A30B		4	1	-45	11.25	7	G1
⊞	2016-10-10 14:39:12	2016-10-10 16:39:12	0004A30B		4	0	-45	11.75	7	G1

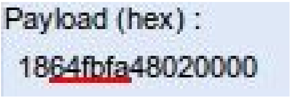
23. If you click on the **+** node you can see the unencrypted data received **Payload (hex):**.



50 last packets

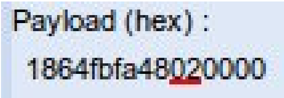
	UTC Timestamp	Local Timestamp	Device UID		Port	Counter UP	LRR RSSI	LRR SNR	Sp Fact	Sub
⊟	2016-10-10 14:41:14 Payload (hex) : 18f2f9a3480e0000	2016-10-10 16:41:14	0004A30B		4	1	-45	11.25	7	G1
⊞	2016-10-10 14:39:12	2016-10-10 16:39:12	0004A30B		4	0	-45	11.75	7	G1

24. All messages start with the number 18, the other three number couples are the data read from the accelerometer mounted on the board. You will see the data changed when you have tilted the board and sent the second message. The accelerometer data are marked in red.



Payload (hex) :
1864fbfa48020000

25. From the firmware version **1.6** sends also the light ambient value, you can read it from the payload after the number 48. In the following figure it is marked with a red line.

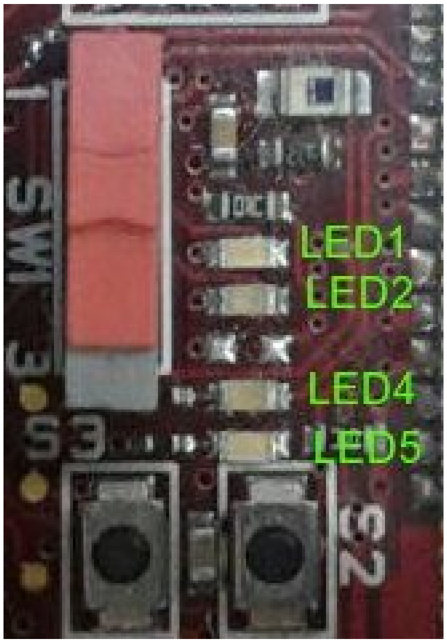


```
Payload (hex) :
  1864fbfa48020000
```

The dark will be rappresented with a low number and a strong light with a high number.

LEDs functionality

The board has four LEDs:



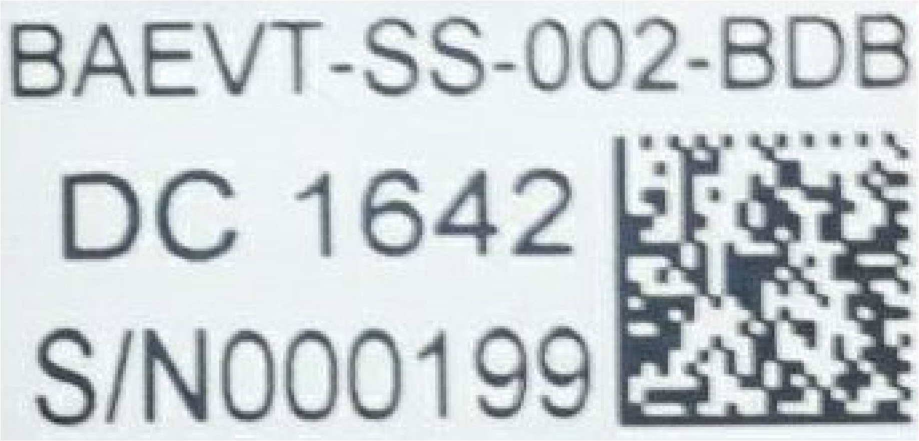
LED	Color	Brief Note
1	Blue	RN2483 GPIO_12
2	Red	RN2483 GPIO_13
3	—	not fitted
4	Green	OpenSDA & MBED
5	Blue	Firmware status

- **LED1** & **LED2** are connected directly via RN2483 GPIOs. By default, they are turned ON during Power-On and on Reset status of the RN2483. After that the firmware sends commands to the RN2483 in order to turn OFF the LEDs. The purpose is to save battery life.
- **LED4** when the Mini-USB cable is connected the led will be turned ON.
- **LED5** is turned ON when the KL26Z is running sending a message over Lora network. It is turned OFF during the deepsleep phase.

Quick start guide for firmware 1.4 or major

This guide permits to send messages over the Lora network. This page is written for Windows users.

Check on the backside of the board a label like this one:



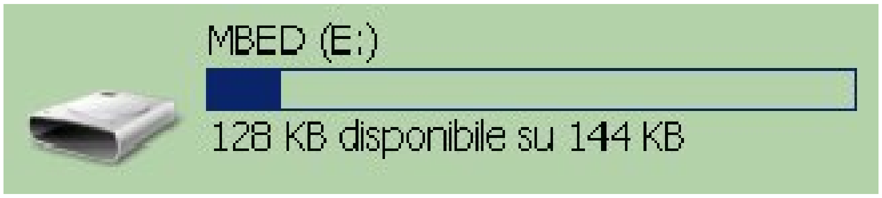
If the serial number *S/N* is lesser than **199** then you have to update the firwmare otherwise skip the next paragraph.

Update firmware

Update your board with the lastest firmware avaiable:

1. Download the file Lora_v15b.bin.zip and unzip it.
2. Connect the board to the PC via Mini Usb cable.

3. Go to the mbed site.
4. Download the lastest mbed serial port driver, it is named as **mbedWinSerial**.
5. Install the driver.
6. When the driver is installed, a MBED mass storage device and a Serial line and a HID device are finded out. Open the MBED mass storage.



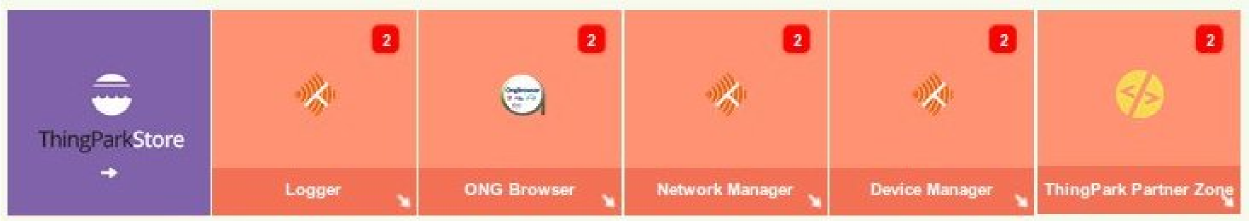
> **Warning:** If the window opened is named **BOOTLOADER** repeat the procedure, the window **must be named MBED**.

7. Copy the file **Lora_v15b.bin** into the MBED mass storage.
8. The Green led on the board will blink while the file is programmed into the device flash memory.

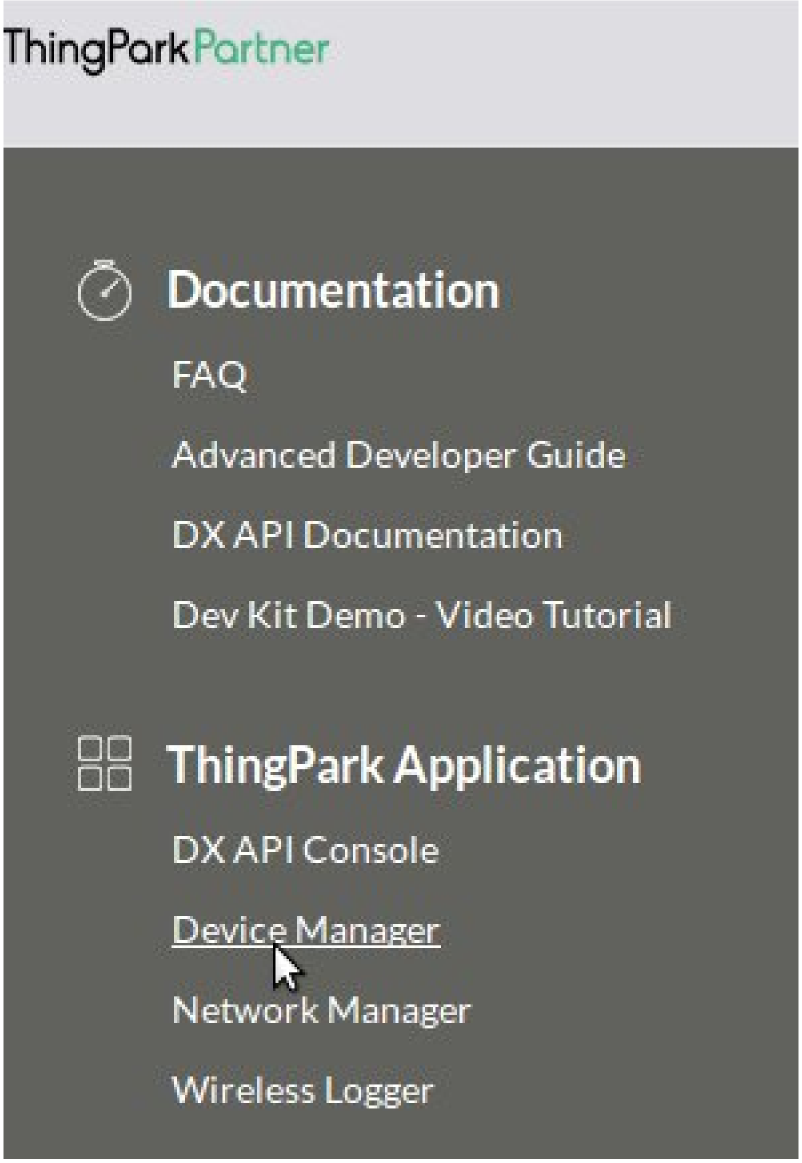
> **Warning:** If accidentally has been copied the firmware in the **BOOTLOADER** window you have corrupted the MBED firmware and you have to fix it. Please read the fix procedure going to the page *How to fix the MBED firmware*

Register Module

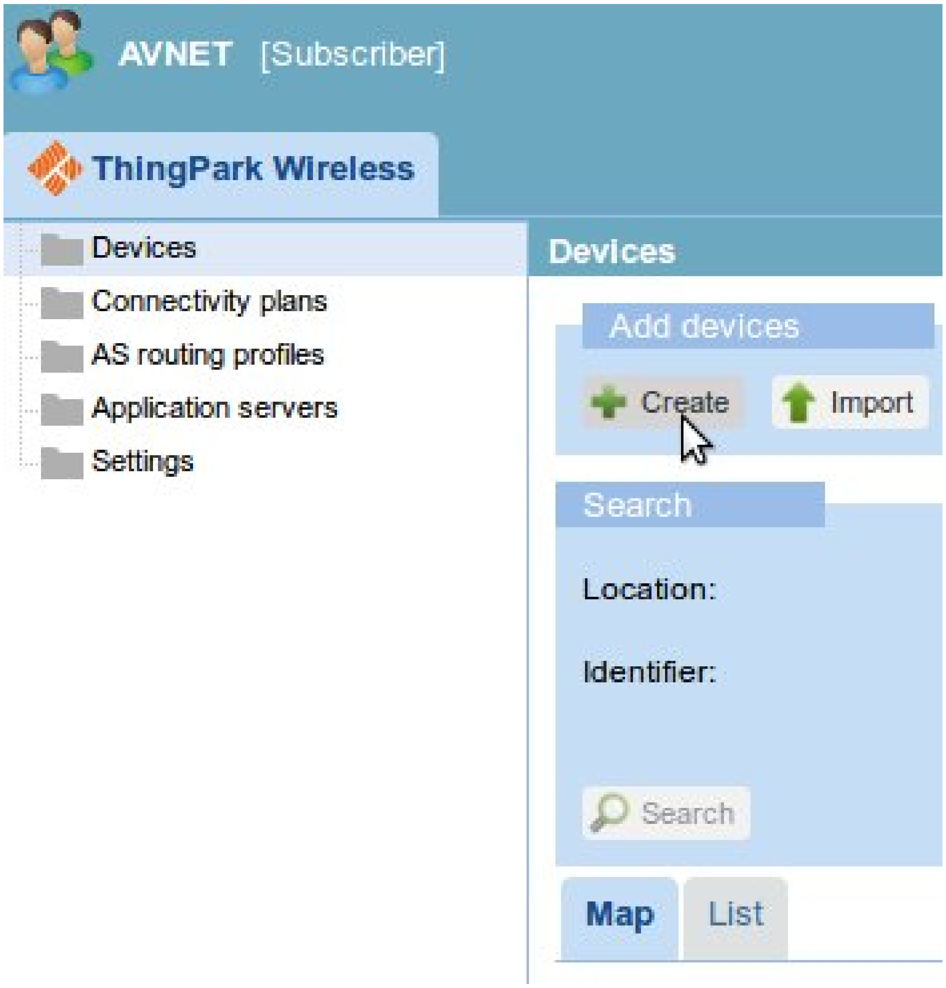
1. In order to see working the Sensor Node Lora you have to register the device on the activity website clicking on **Create an account**.
2. After the registration login and you will access the main page.



3. Register your Microchip RN2483 device, to do this, click on **Device Manager** arrow to open a new window.

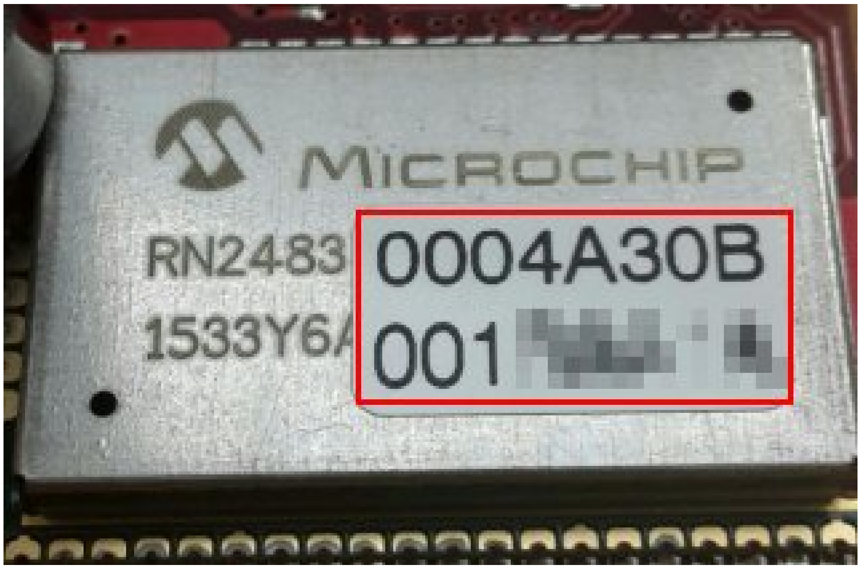


4. Click the right mouse button on the **Devices** folder and select **+ Create**.



5. In this form you have to insert the data we have stored in the RN2483. The following fields are mandatory:

- **Device EUI**: the 16-hex identification key of the device. It is unique for every device, you can read it on the label of the board as in the image. For example the code can be **0004A30B001B9954**



- **Network Address**: This is the **device address** provided by the operator, contains the NetworkID where you want to connect the device.
- **Device profile**: **LoRaWAN 1.0 class A**
- **Network key (hexa)**: we have already saved it in the RN2834 device, it is the **Device EUI** key copied *twice*, for example if the device EUI is **0004A30B001B9954** then the **network session key** you have to insert will be **0004A30B001B99540004A30B001B9954**.
- **Application keys**: Insert **AFBECD56473829100192837465FAEBDC**, port **4**. We have saved it in the RN2834 device. This is the **application sessio key**.
- **Connectivity plan**: choose yours

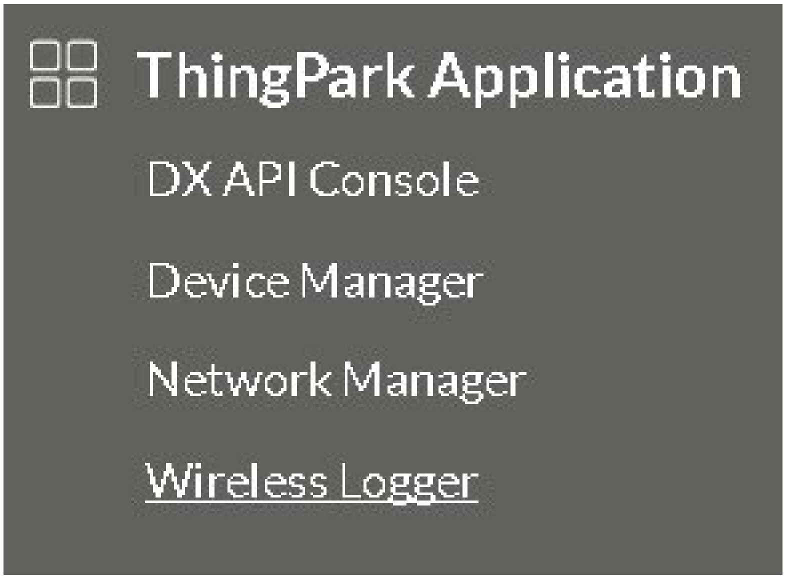
Optional field but useful:

- **Name**: insert a name just to recognize your device.

6. Click on the top right **+ create**.

Warning: If some key is wrong then the device will be unable to send data correctly in the Lora network.

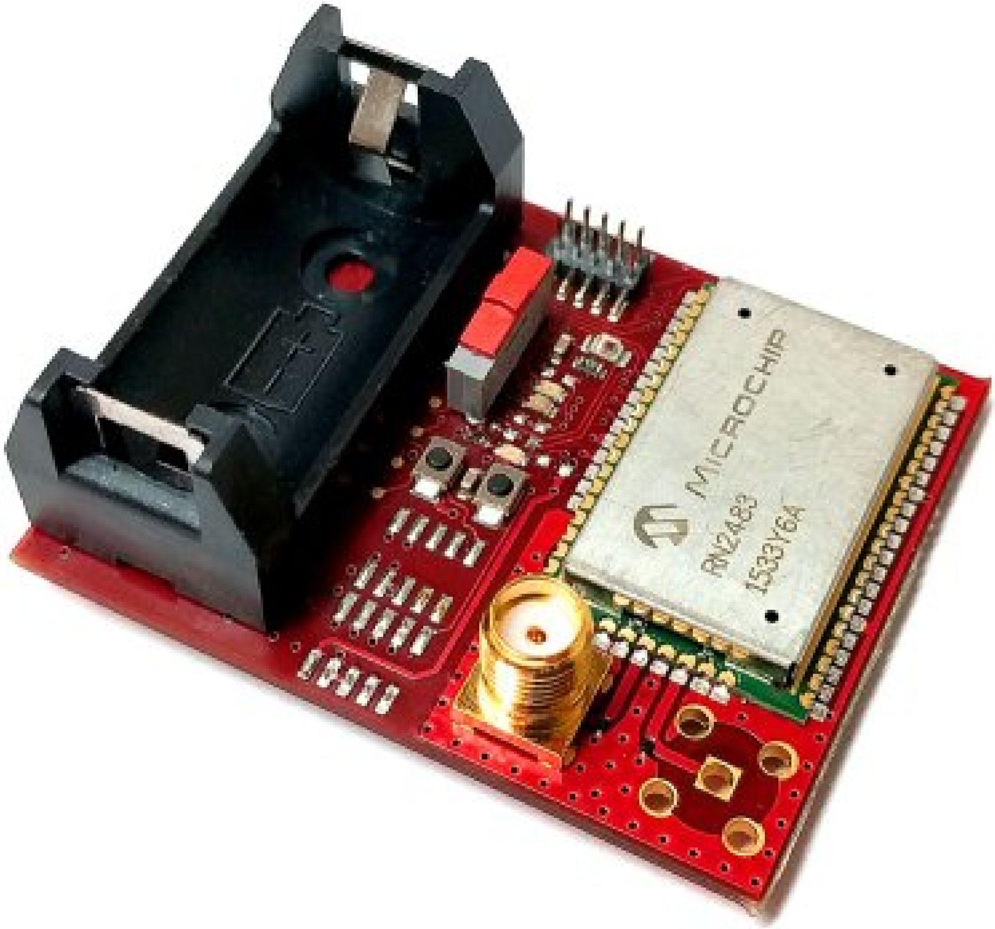
7. After the registration, you can close the window device manager and on the main page go to **Logger**.



8. In this page you will see all the messages sent by your device.

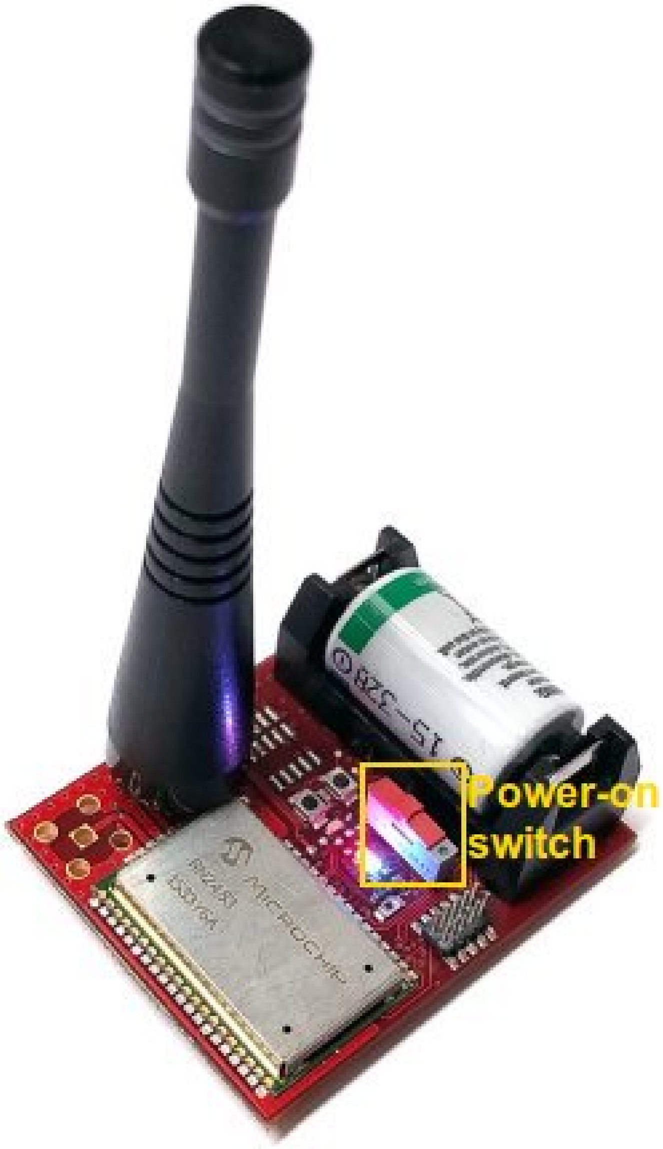
Use your board

Now take your board:

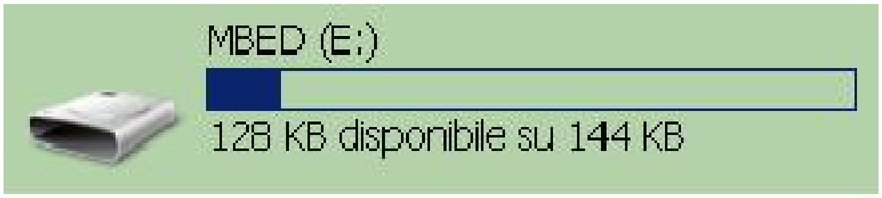


The board doesn't have the correct **device address** in order to send correctly messages to your operator, so you have to set it. Follow next steps correctly:

1. Insert antenna and Micro-USB from the windows PC to the board and put the board on the table.
2. If you did not installed the mbed serial driver then go to mbed website otherwise go to the step **5**
3. Download the lastest mbed serial port driver, it is named as **mbedWinSerial**.
4. Install the driver.
5. Now switch on the Lora Sensor Node.



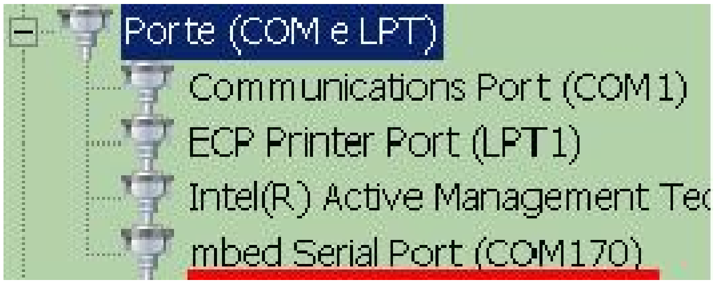
6. If the MBED Windows serial port driver is installed correctly Windows will recognize the board as a memory storage called MBED and as a Serial Com Port.



MBED (E:)

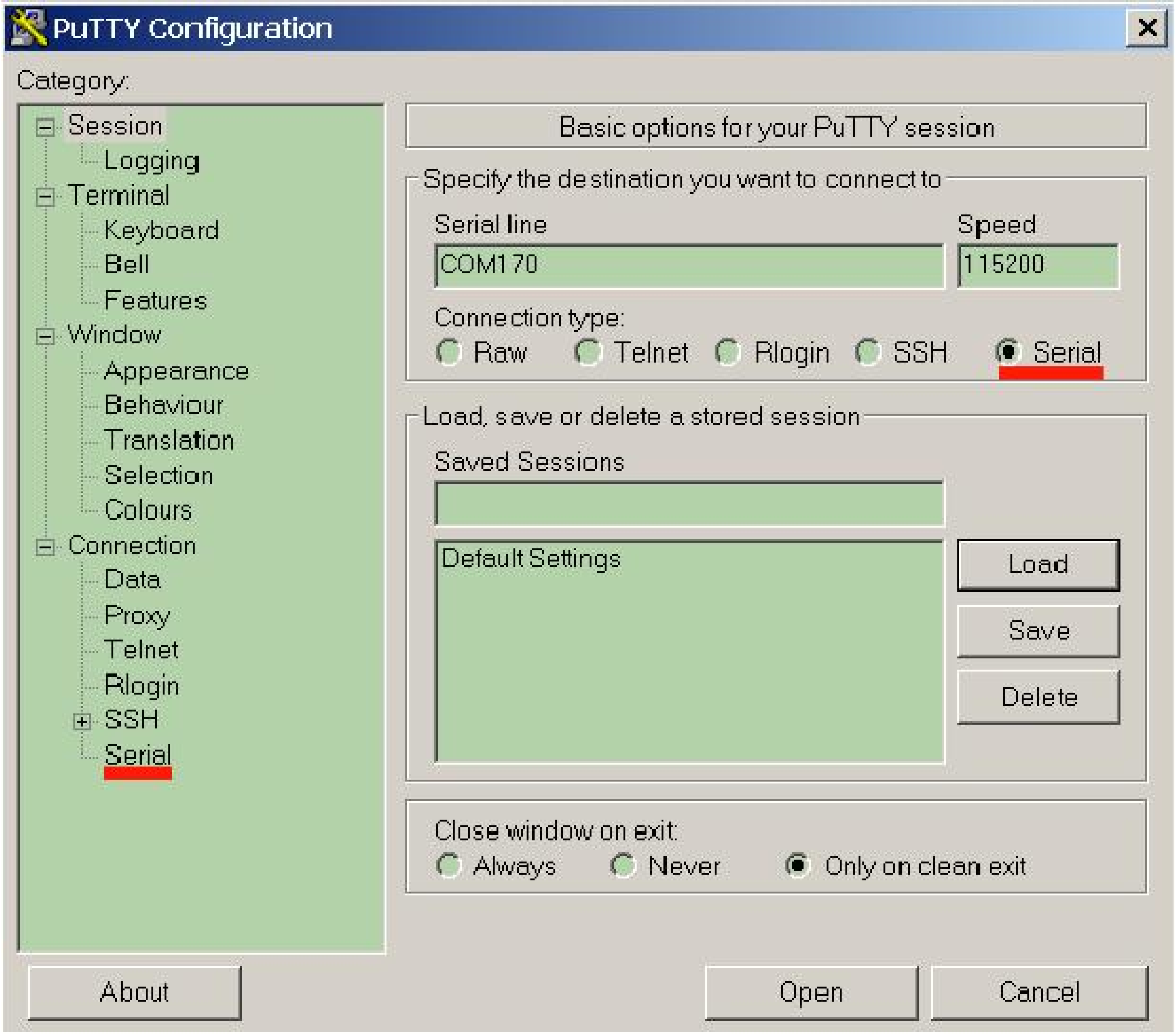
128 KB disponibile su 144 KB

7. Now we have to connect a terminal to the Serial Com. First step, find which number is, going to “Device Manager” and finding the node called “Ports (COM and LPT)”. In this example the number port is 170.



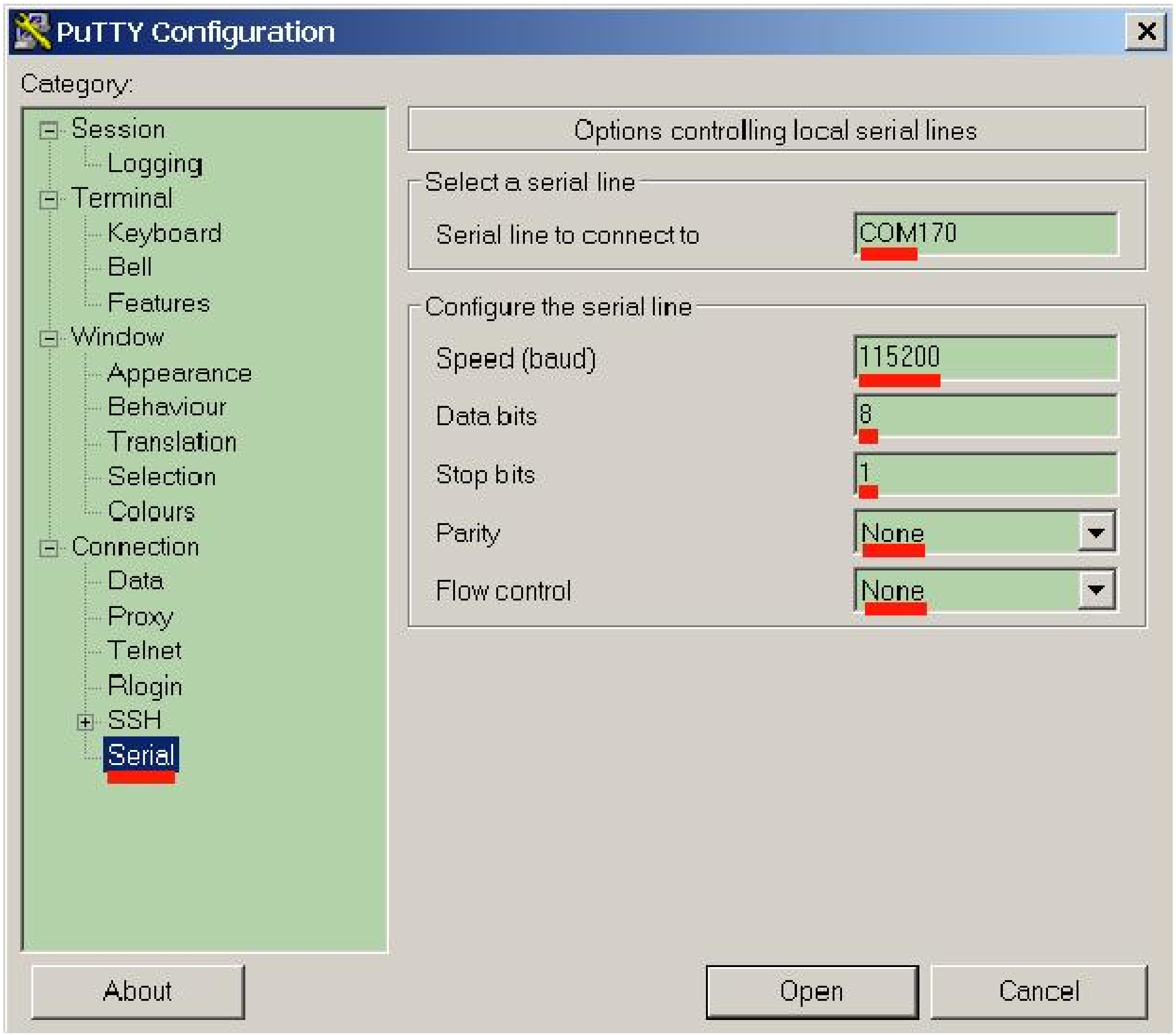
Porte (COM e LPT)
Communications Port (COM1)
ECP Printer Port (LPT1)
Intel(R) Active Management Tec
mbed Serial Port (COM170)

8. Open a terminal like **putty**, you can find it here. Launch it and set the properties signed in red in the figures:



Specify the destination you want to connect to
Serial line: COM170
Speed: 115200
Connection type:
○ Raw ○ Telnet ○ Rlogin ○ SSH ● Serial

Close window on exit:
○ Always ○ Never ● Only on clean exit



Serial line to connect to: COM170

Speed (baud): 115200

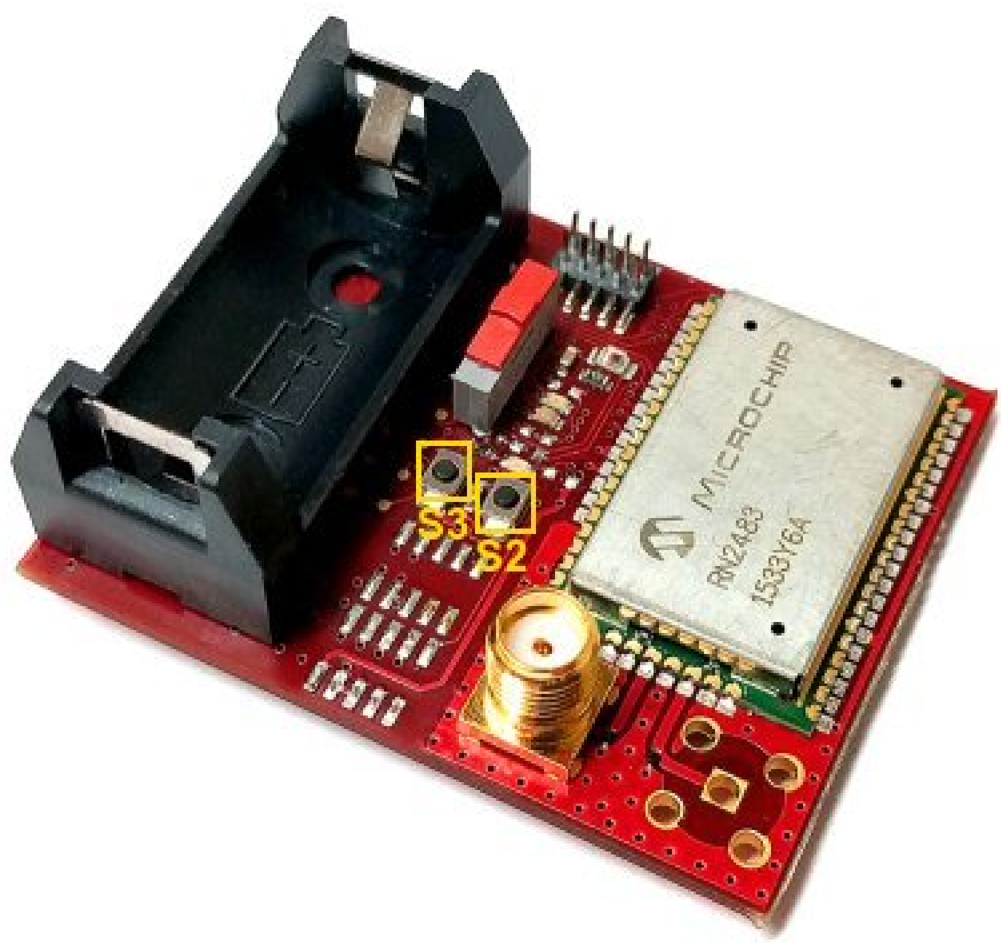
Data bits: 8

Stop bits: 1

Parity: None

Flow control: None

9. Press the **Open** button. Now reset the board pressing **S3** button and after keep the **S2** button pressed for 5 seconds.



10. On the putty console you will see the starting boot text:

```
BAEVTSS002 firmware version 1.5b
RN2483 1.0.1 Dec 15 2015 09:38:06
```

11. Keeping the **S2** button pressed you will see after some seconds the text **Please release the button S2**. Will be displayed a menu like this one:

```
*** MENU CONFIG RN2483 ***
Enter 0 to print this menu
Enter 1 to see values in DEVEUI and DEVADDR registers
Enter 2 to modify DEVEUI
Enter 3 to modify DEVADDR
Enter 4 to modify APPSKEY
Enter 5 to modify NWKSKEY
Enter 6 to save changes
Enter 7 to Microchip default reset
Enter 8 to Avnet default reset
Enter 9 to exit
```

From this menu you will be able to change the registers **DEVEUI**, **DEVADDR**, **APPSKEY** and **NWKSKEY**. furthermore it is possible reset the RN2483 with factory reset by Microchip or by Avnet.

Warning: The BAEVTSS002 is setted by default with **Avnet reset**.

12. In this case, we want change the networkID so you have to press the key **2** to be able to change the **DEVEUI** register, it will be showed:

```
Changing deveui register, current value: 001B9954
Enter new deveui value, 8 hex numbers lenght (press ENTER to abort):
>>
```

As you can see in the first line it is displayed the current value in the register and how many numbers you have to insert in order to change the register.

Note: The current value of the register it is displayed only for **DEVEUI** and **DEVADDR** registers. **APPSKEY** and **NWKSKEY** register are write-only so their current value won't be showed.

13. Insert **8** hexs numbers, for example **12345678**. After the last key will be pressed automatically the value of register will change.

```
Changing deveui register, current value: 001B9954
Enter new deveui value, 8 hex numbers lenght (press ENTER to abort):
>> 12345678
```

Note: If you press Enter before to insert all the keys the procedure will abort. **12345678** is just for example, please contact your Operator in order to know which networkID you have to use. Without the correct networkID you won't be able to send messages.

14. If the procedure will have success will appear the message **value changed**:

```
Changing deveui register, current value: **001B9954**
Enter new deveui value, 8 hex numbers lenght (press ENTER to abort):
>> 12345678
value changed
```

15. Now the **DEVEUI** register is changed but if the **RN2843** will be turned off the modify will be lost. In order to keep the modify you have to save it. To do it, from the menu press the key **6** referred to *Enter 6 to save changes*. During the saving will be showed:

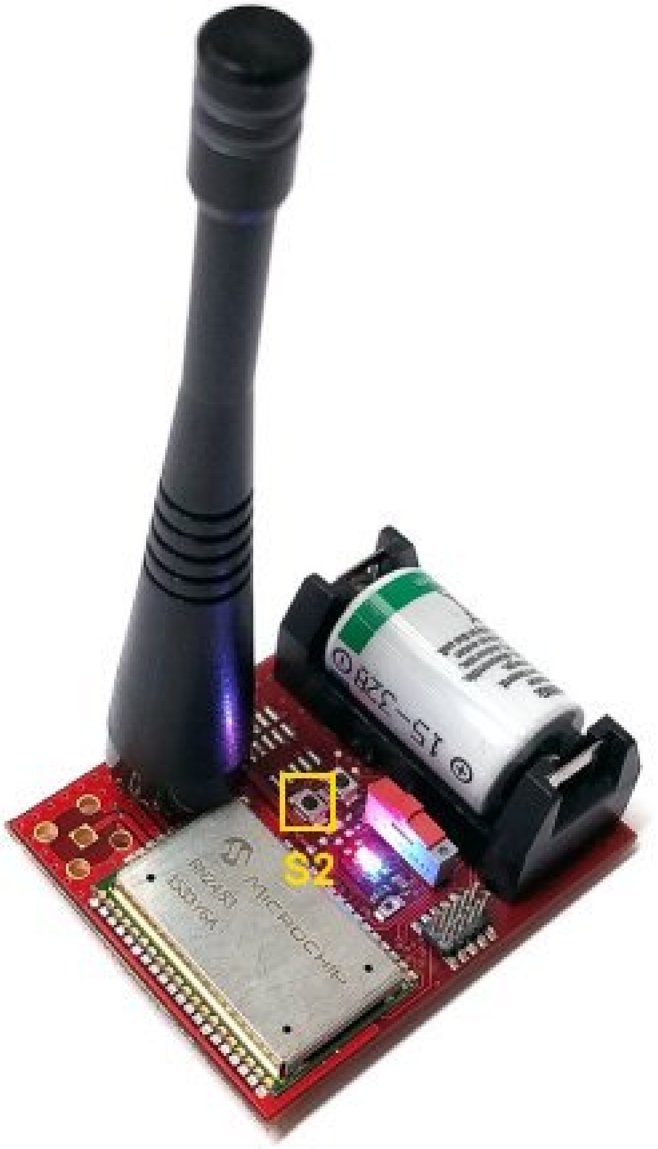
```
Saving modifies, please don't turn off the device...
please wait...
```

16. The saving will terminate when will appear the message **success**:

```
Saving modifies, please don't turn off the device...
please wait...success
```

Note: If you have to change other values use the appropiate keys. If you want only see the value of the readable register press the key **1**

17. Now to start the demo exit from the menu pressing the key **9**. The device will enter automatically in **sleep mode**.
18. Every 2 minutes it will wake up for a short time in order to send a message. It is possible skipping the sleeping phase pressing the **S2** button. When the board is sending a message a blue led will be turned on.

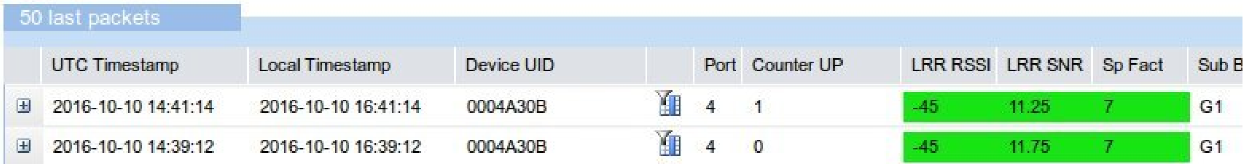


19. After pressing **S2** button you will see the first message sent by your device. Now tilt your board to 90 degrees and wait 30 seconds.

20. The board will send another message.

21. Now it’s time to see the data sent. Power off the board.

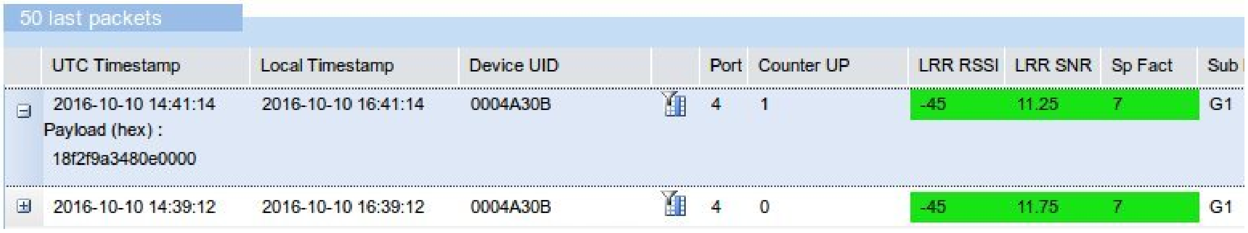
22. In the logger window, you will have 2 rows, every row is a message received from the server.



50 last packets

	UTC Timestamp	Local Timestamp	Device UID		Port	Counter UP	LRR RSSI	LRR SNR	Sp Fact	Sub B
⊞	2016-10-10 14:41:14	2016-10-10 16:41:14	0004A30B		4	1	-45	11.25	7	G1
⊞	2016-10-10 14:39:12	2016-10-10 16:39:12	0004A30B		4	0	-45	11.75	7	G1

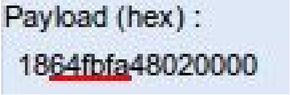
23. If you click on the **+** node you can see the unencrypted data received **Payload (hex):**.



50 last packets

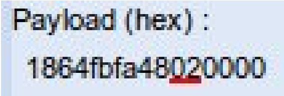
	UTC Timestamp	Local Timestamp	Device UID		Port	Counter UP	LRR RSSI	LRR SNR	Sp Fact	Sub
⊟	2016-10-10 14:41:14 Payload (hex) : 18f2f9a3480e0000	2016-10-10 16:41:14	0004A30B		4	1	-45	11.25	7	G1
⊞	2016-10-10 14:39:12	2016-10-10 16:39:12	0004A30B		4	0	-45	11.75	7	G1

24. All messages start with the number 18, the other three number couples are the data read from the accelerometer mounted on the board. You will see the data changed when you have tilted the board and sent the second message. The accelerometer data are marked in red.



Payload (hex) :
1864fbfa48020000

25. From the firmware version **1.5** sends also the light ambient value, you can read it from the payload after the number 48. In the following figure it is marked with a red line.

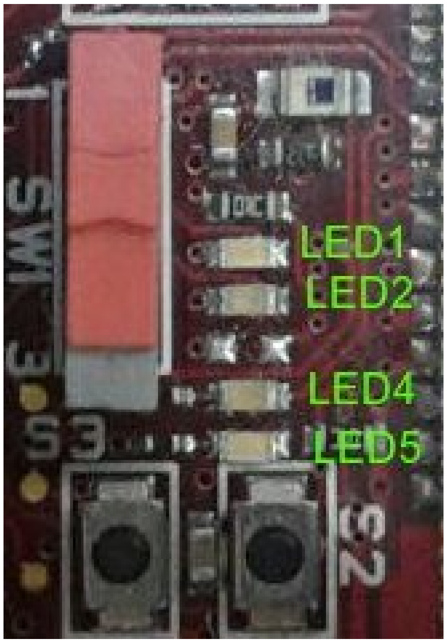


Payload (hex) :
1864fbfa48020000

The dark will be rappresented with a low number and a strong light with a high number.

LEDs functionality

The board has four LEDs:



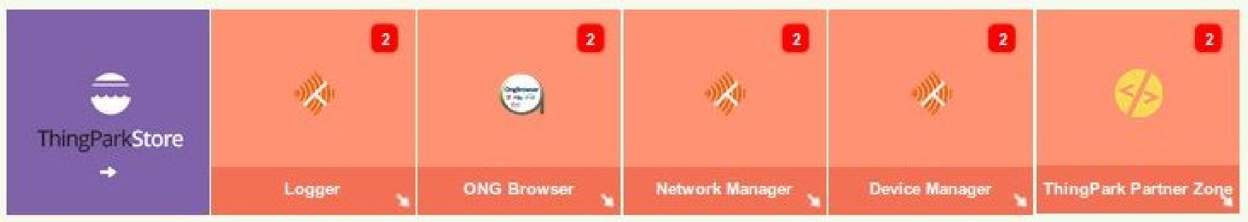
LED	Color	Brief Note
1	Blue	RN2483 GPIO_12
2	Red	RN2483 GPIO_13
3	—	not fitted
4	Green	OpenSDA & MBED
5	Blue	Firmware status

- **LED1** & **LED2** are connected directly via RN2483 GPIOs. By default, they are turned ON during Power-On and on Reset status of the RN2483. After that the firmware sends commands to the RN2483 in order to turn OFF the LEDs. The purpose is to save battery life.
- **LED4** when the Mini-USB cable is connected the led will be turned ON.
- **LED5** is turned ON when the KL26Z is running sending a message over Lora network. It is turned OFF during the deepsleep phase.

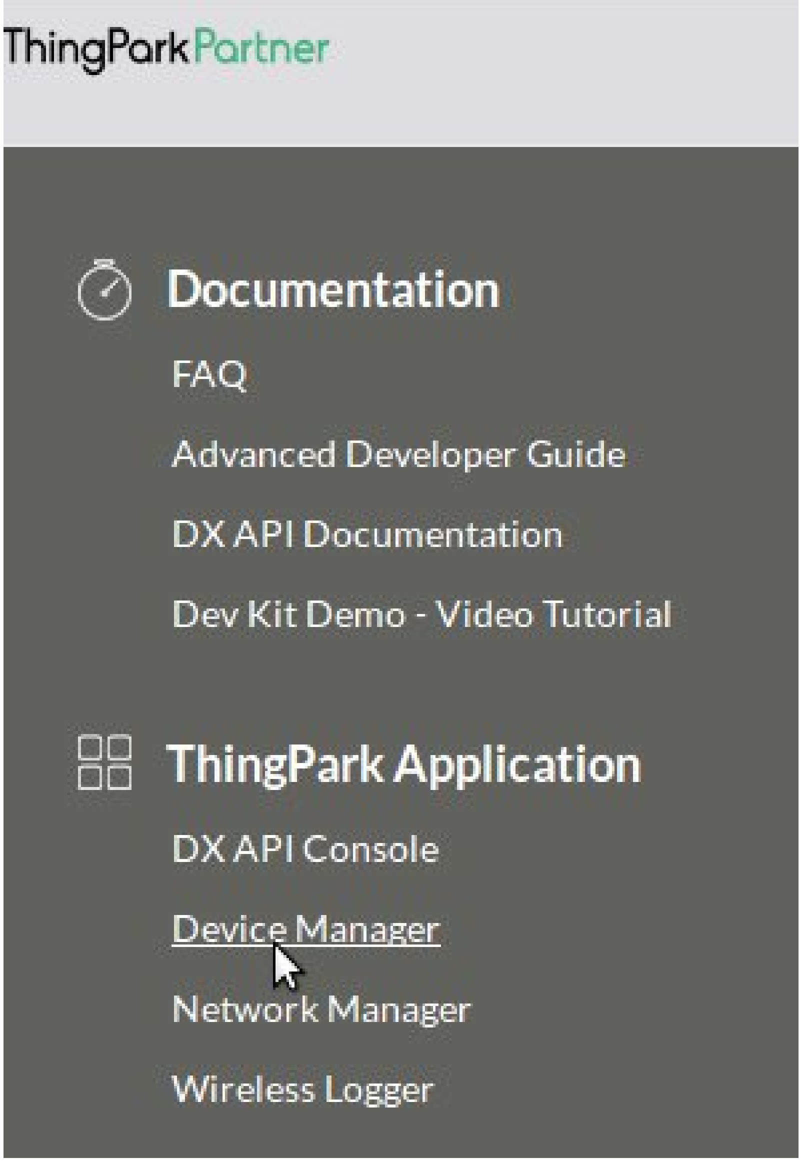
Quick start guide for firmware 1.0

This guide is valid if you can choose by yourself the devaddr on the registration device of the Actility webpage. If you want upgrade the firmware to the newest version then go to *Quick start guide for firmware 1.4 or major*.

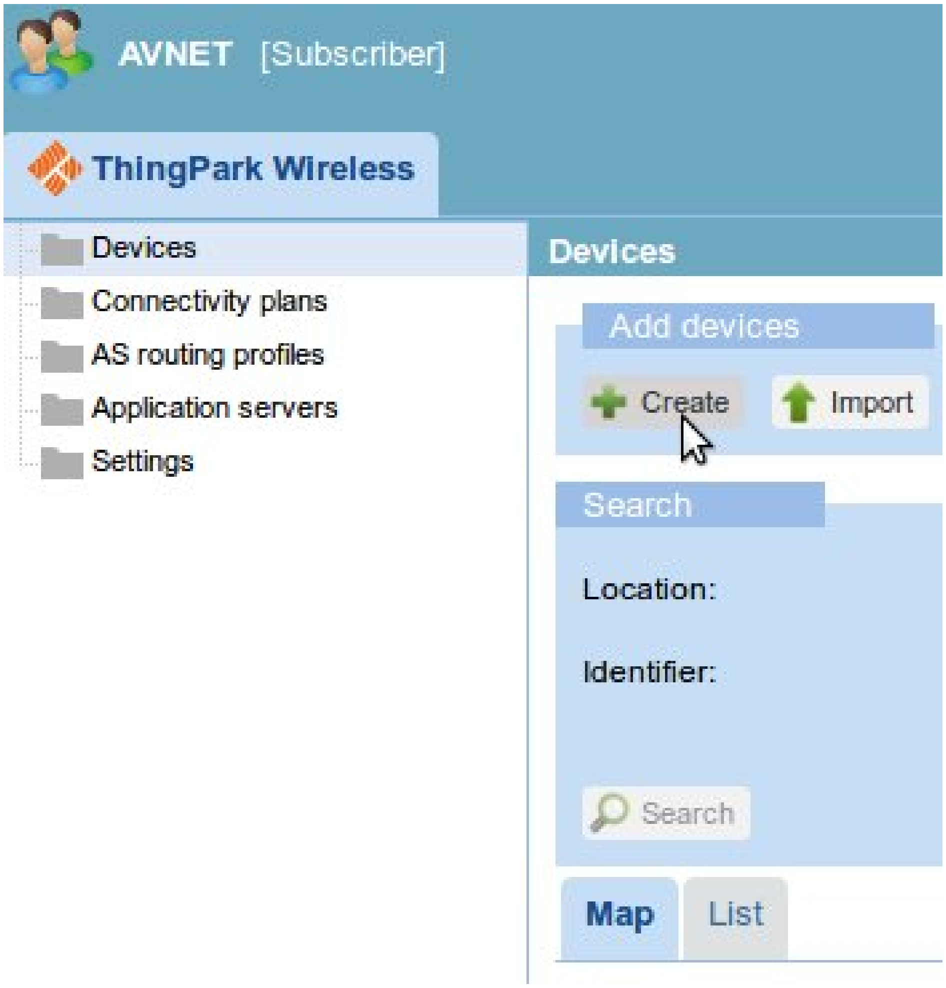
In order to see working the Sensor Node Lora you have to register the device on the activity website clicking on **Create an account**. After the registration login and you will access the main page.



First up we have to register your Microchip RN2483 device, to do this, click on **Device Manager** arrow to open a new window.

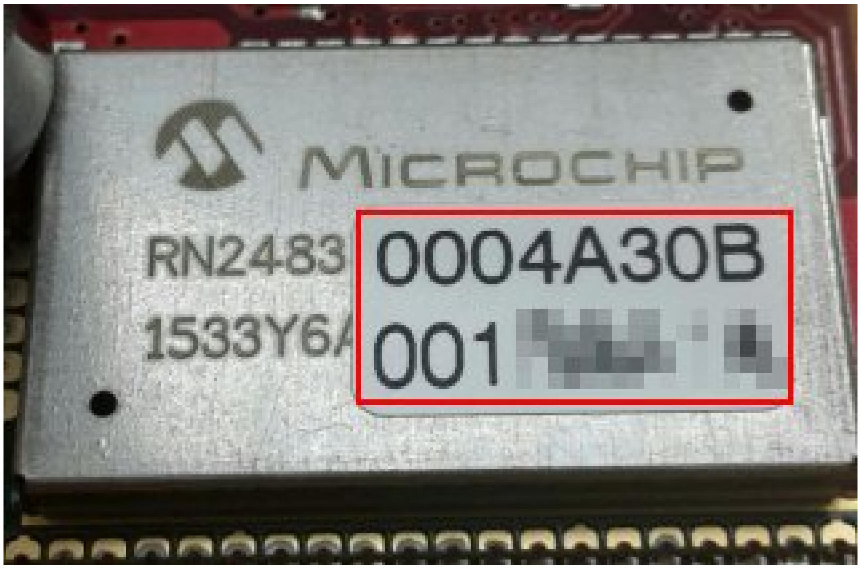


Click the right mouse button on the **Devices** folder and select **+ Create**.



In this form you have to insert the data we have stored in the RN2483. The following fields are mandatory:

- **Device EUI**: the 16-hex identification key of the device. It is unique for every device, you can read it on the label of the board as in the image. For example the code can be **0004A30B001B9954**



- **Network address**: last 4 less significant bytes from the Device EUI key, for example if the code is **0004A30B001B9954** then the network address will be **001B9954**

Note: if your board has the Mini-USB connector then the key will be the last 4 less significant bytes from the Device EUI key but the second character will be **8** and not **0**. So in this example it will be **081B9954**

- **Device profile**: **LoRaWAN 1.0 class A**
- **Network key (hexa)**: we have already saved it in the RN2834 device, it is the **Device EUI** key copied *twice*, for example if the device EUI is **0004A30B001B9954** then the **network key** you have to insert will be **0004A30B001B99540004A30B001B9954**.
- **Application keys**: Insert **AFBECD56473829100192837465FAEBDC**, port **4**. We have saved it in the RN2834 device.
- **Connectivity plan**: choose yours

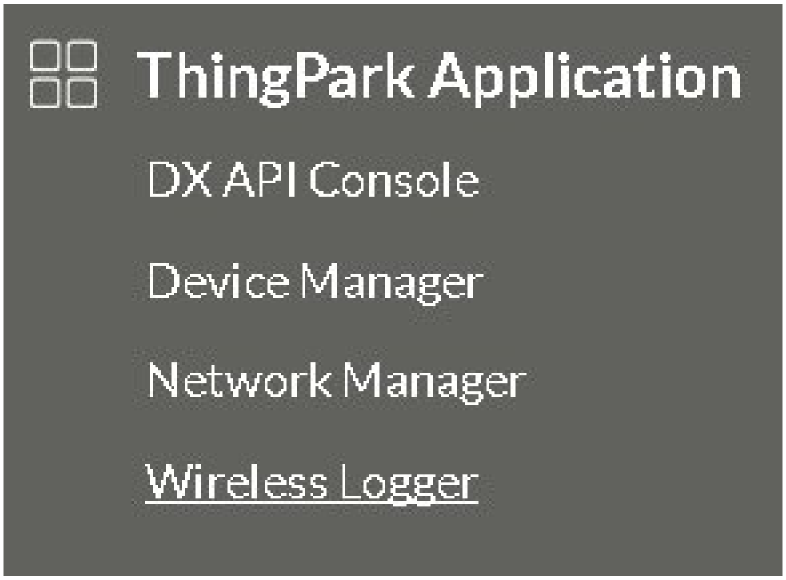
Optional field but useful:

- **Name**: insert a name just to recognize your device.

Then click on the top right **+ create**.

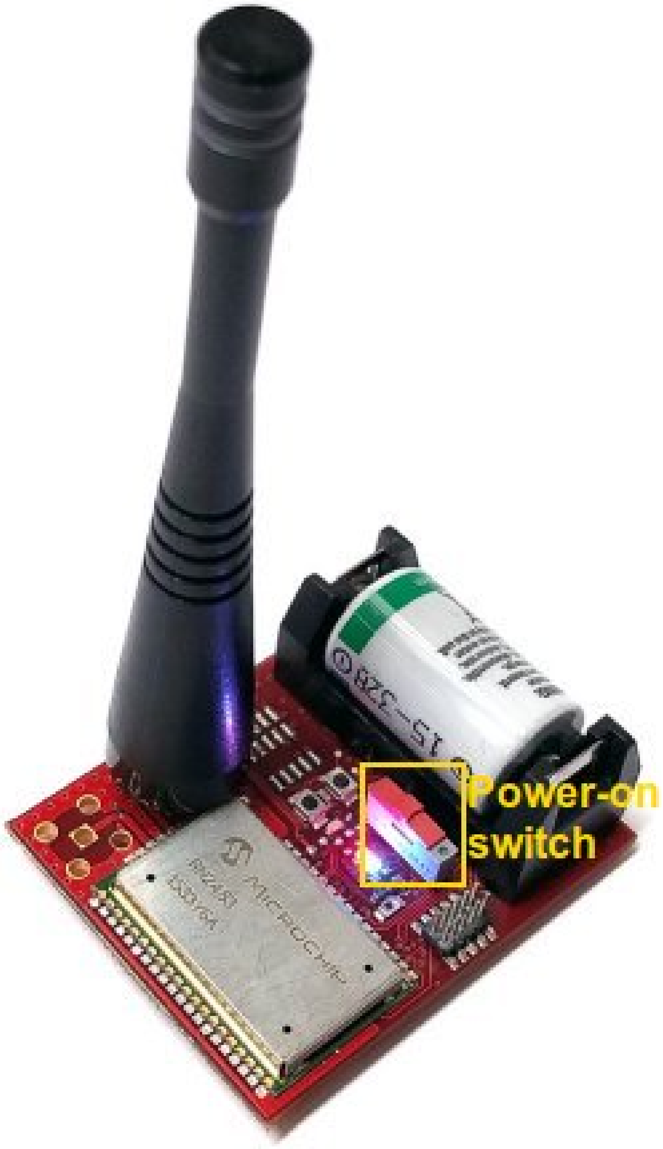
Warning: If some key is wrong then the device will be unable to send data correctly in the Lora network.

After the registration, you can close the window device manager and on the main page go to **Logger**.

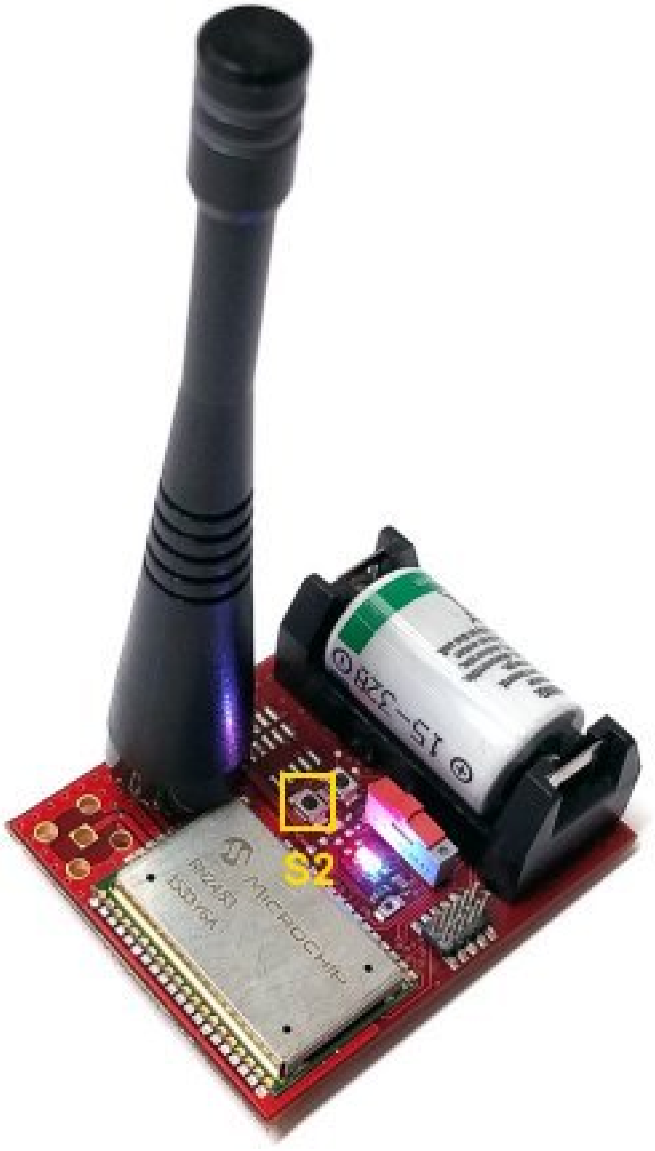


Here you will see all the messages sent by your device. Now take your board:

insert antenna and battery and keeping the board on the table switch on the Lora Sensor Node.

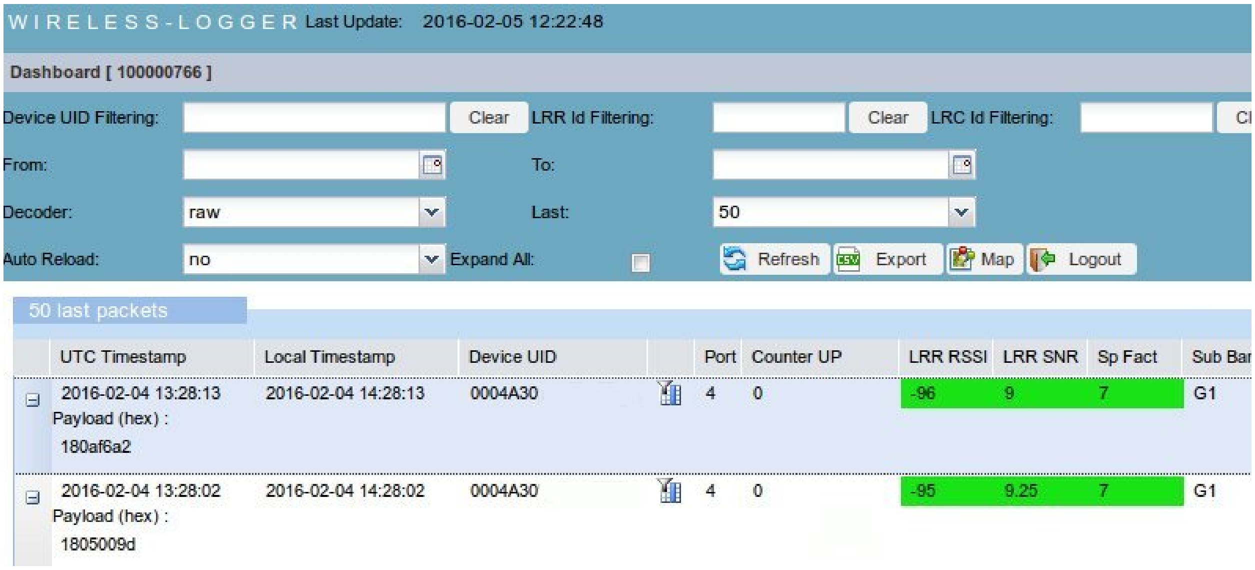


Now the device is in sleeping mode, in 30 seconds it will wake up and it is going to sent a message. For skipping the sleeping phase, press **S2** button.



After pressing **S2** button you will see the first message sent by your device. Now tilt your board to 90 degrees and wait 30 seconds.

The board will send another message. Now it's time to see the data sent. Power off the board. And in the logger window, you will have 2 rows, every row is a message received from the server. If you click on the **+** node you can see the unencrypted data received **Payload (hex):**. All messages start with the number 18, the other three number couples are the data read from the accelerometer mounted on the board. You will see the data changed when you have tilted the board and sent the second message.



WIRELESS-LOGGER Last Update: 2016-02-05 12:22:48

Dashboard [100000766]

Device UID Filtering: Clear LRR Id Filtering: Clear LRC Id Filtering: Cl

From: To:

Decoder: raw Last: 50

Auto Reload: no Expand All: □

50 last packets

	UTC Timestamp	Local Timestamp	Device UID		Port	Counter UP	LRR RSSI	LRR SNR	Sp Fact	Sub Bar
⊟	2016-02-04 13:28:13 Payload (hex) : 180af6a2	2016-02-04 14:28:13	0004A30		4	0	-96	9	7	G1
⊟	2016-02-04 13:28:02 Payload (hex) : 1805009d	2016-02-04 14:28:02	0004A30		4	0	-95	9.25	7	G1

Developing guide

This guide will provide instructions to install the development environment needed to compile and debug the demo firmware of the Sensor Node Lora. The development system is multiplatform, it supports Windows and Linux. This guide is written using Windows. The main steps are:

- Install **Kinetis Design Studio 3.2.0**, upgrade it and install **KSDK 1.3.0**

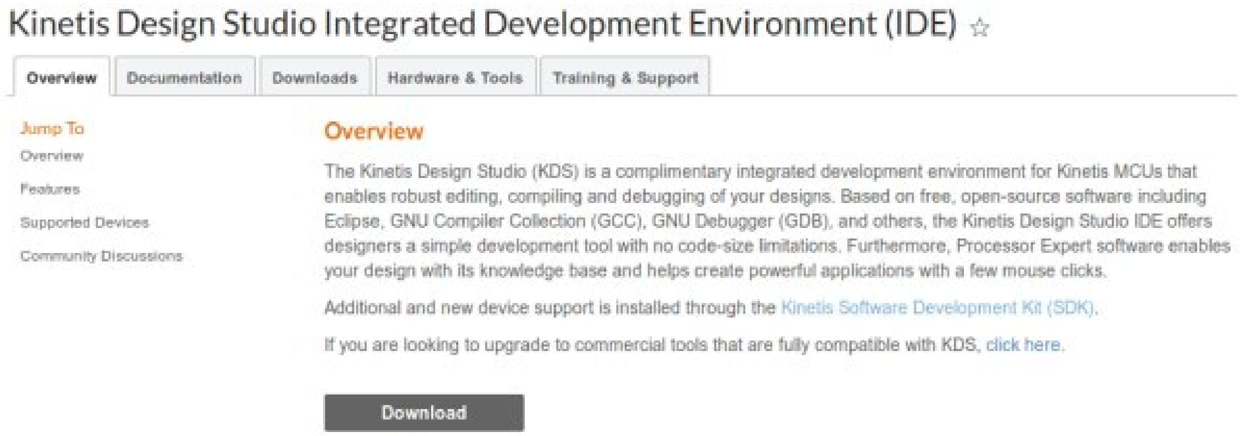
- Import build & debug the source project

Hardware required:

- Sensor Node Lora with battery or Mini-USB cable
- if you don't use Mini-USB cable Segger it is required a J-Link (Segger website) with SWD debug interface.
- PC with Windows or Linux

Install & Update Kinetis Design Studio

1. First up, register at the NXP website registration form
2. Download the IDE from this page. Clicking on **Download** button.



Kinetis Design Studio Integrated Development Environment (IDE)

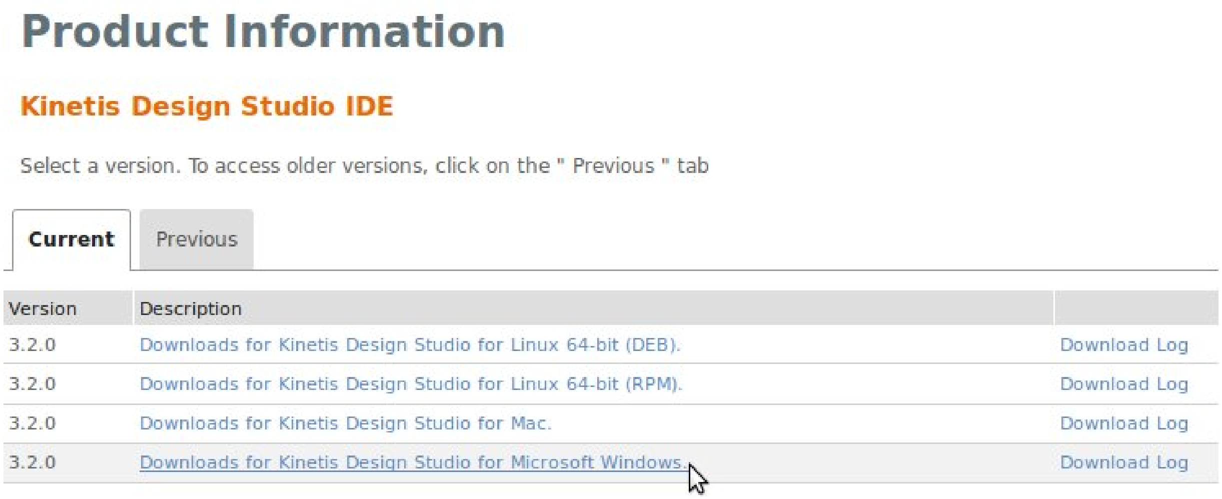
Overview

The Kinetis Design Studio (KDS) is a complimentary integrated development environment for Kinetis MCUs that enables robust editing, compiling and debugging of your designs. Based on free, open-source software including Eclipse, GNU Compiler Collection (GCC), GNU Debugger (GDB), and others, the Kinetis Design Studio IDE offers designers a simple development tool with no code-size limitations. Furthermore, Processor Expert software enables your design with its knowledge base and helps create powerful applications with a few mouse clicks.

Additional and new device support is installed through the Kinetis Software Development Kit (SDK).

If you are looking to upgrade to commercial tools that are fully compatible with KDS, click here.

3. Click on **Downloads for Kinetis Design Studio for Microsoft Windows**.



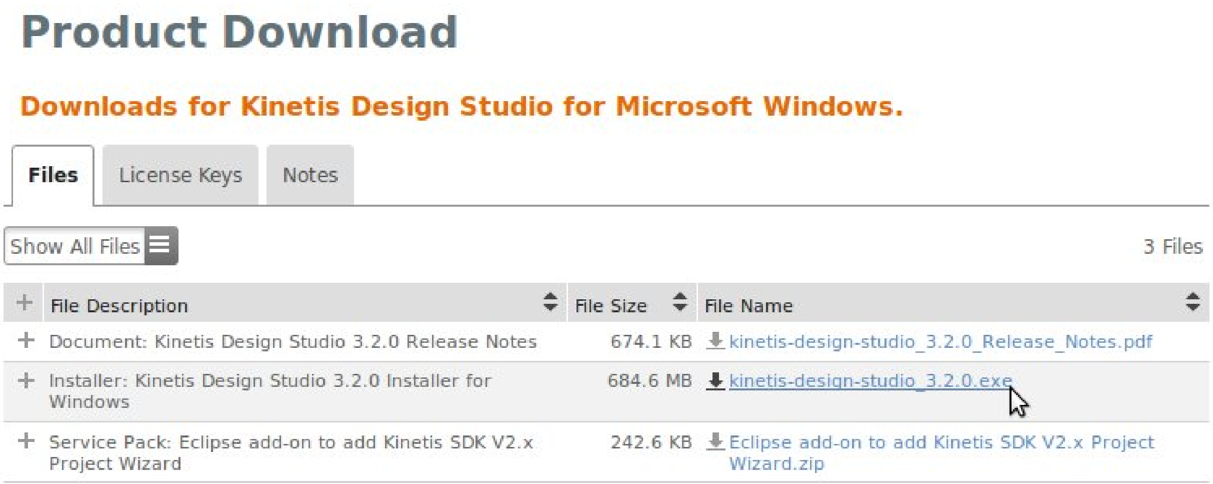
Product Information

Kinetis Design Studio IDE

Select a version. To access older versions, click on the " Previous " tab

Version	Description	
3.2.0	Downloads for Kinetis Design Studio for Linux 64-bit (DEB).	Download Log
3.2.0	Downloads for Kinetis Design Studio for Linux 64-bit (RPM).	Download Log
3.2.0	Downloads for Kinetis Design Studio for Mac.	Download Log
3.2.0	Downloads for Kinetis Design Studio for Microsoft Windows.	Download Log

4. Agree the terms and download the file Installer: **Kinetis Design Studio 3.2.0 Installer for Windows**.

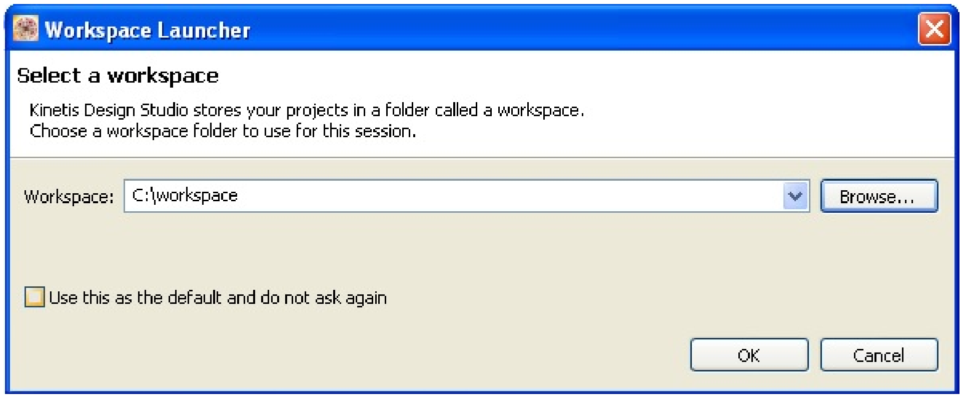


Product Download

Downloads for Kinetis Design Studio for Microsoft Windows.

+	File Description	File Size	File Name
+	Document: Kinetis Design Studio 3.2.0 Release Notes	674.1 KB	kinetis-design-studio_3.2.0_Release_Notes.pdf
+	Installer: Kinetis Design Studio 3.2.0 Installer for Windows	684.6 MB	kinetis-design-studio_3.2.0.exe
+	Service Pack: Eclipse add-on to add Kinetis SDK V2.x Project Wizard	242.6 KB	Eclipse add-on to add Kinetis SDK V2.x Project Wizard.zip

5. Next, launch the downloaded file **kinetis-design-studio_3.2.0.exe** following all the default options.
6. Launch the KDS and select a directory for the workspace. Our project will be imported in this folder. In this guide we used this path:



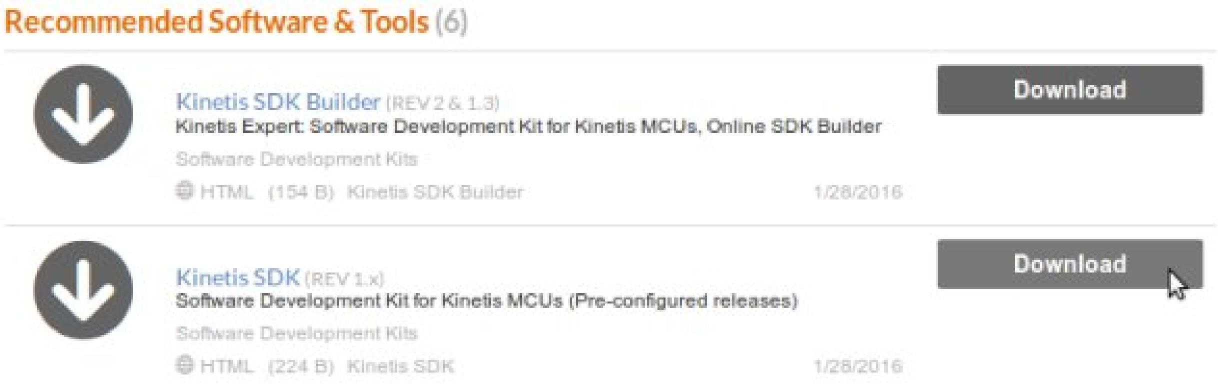
Select a workspace

Kinetis Design Studio stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\workspace

☐ Use this as the default and do not ask again

7. At this point, if you want to modify the project using **processor expert** it is required to install also the package **KSDK 1.3.0** in order to be compatible with the project. You can find this package on KINETIS-SDK page. Select Download button from **Kinetis SDK**.

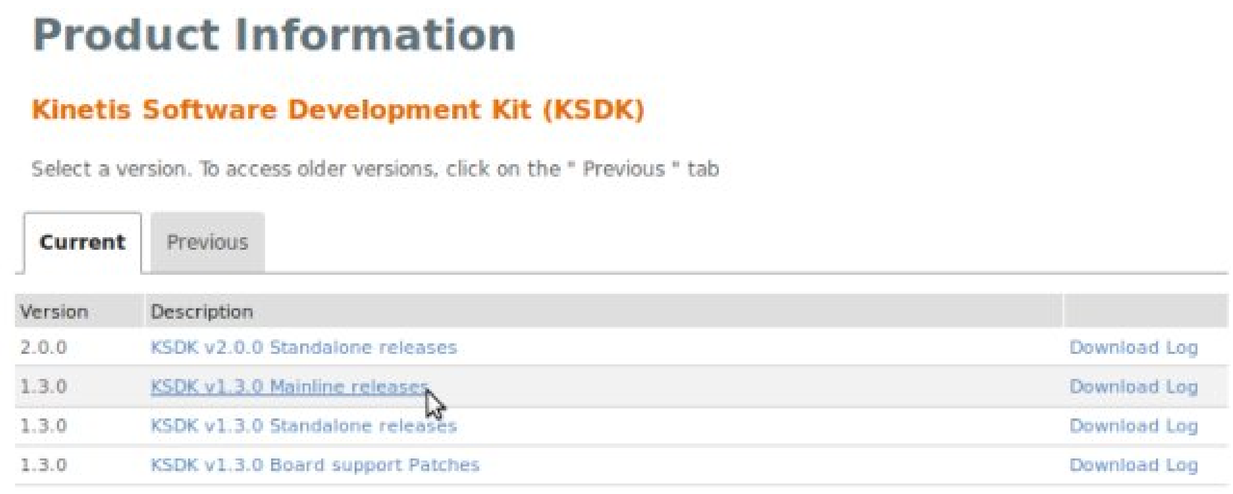


Recommended Software & Tools (6)

Kinetis SDK Builder (REV 2 & 1.3)
Kinetis Expert: Software Development Kit for Kinetis MCUs, Online SDK Builder
Software Development Kits
HTML (154 B) Kinetis SDK Builder 1/28/2016

Kinetis SDK (REV 1.x)
Software Development Kit for Kinetis MCUs (Pre-configured releases)
Software Development Kits
HTML (224 B) Kinetis SDK 1/28/2016

8. Select **KSDK v1.3.0 Mainline releases**



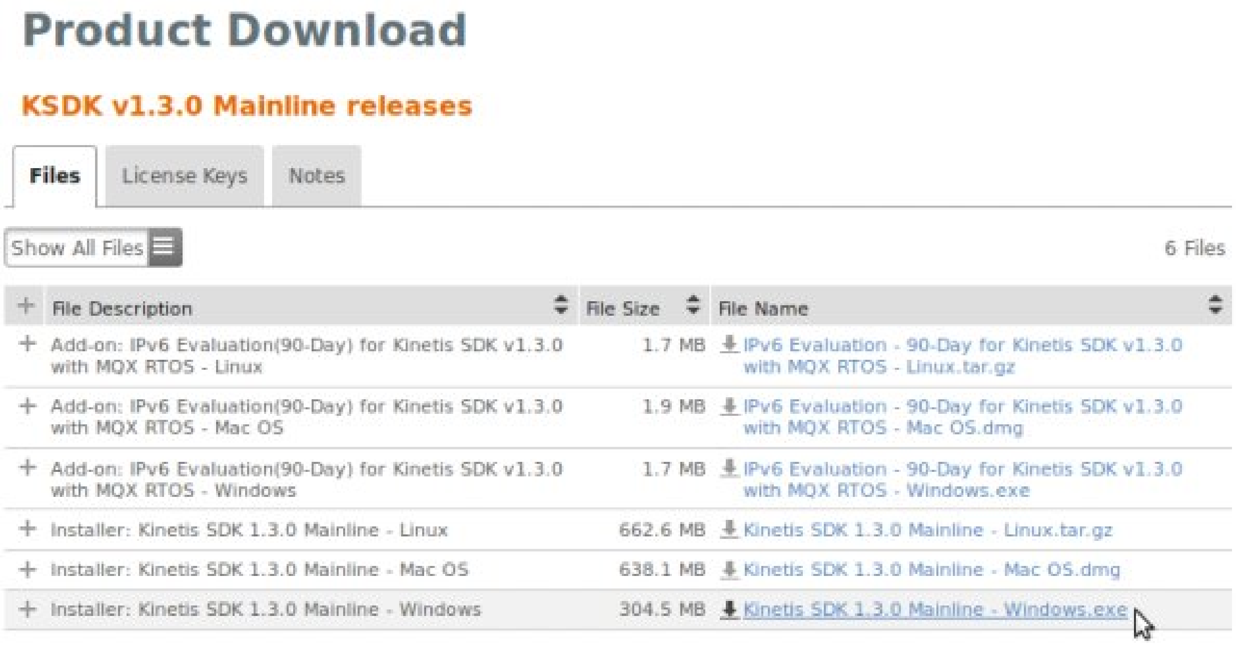
Product Information

Kinetis Software Development Kit (KSDK)

Select a version. To access older versions, click on the " Previous " tab

Version	Description	
2.0.0	KSDK v2.0.0 Standalone releases	Download Log
1.3.0	KSDK v1.3.0 Mainline releases	Download Log
1.3.0	KSDK v1.3.0 Standalone releases	Download Log
1.3.0	KSDK v1.3.0 Board support Patches	Download Log

9. Agree the terms and then download **Kinetis SDK 1.3.0 Mainline - Windows.exe**



Product Download

KSDK v1.3.0 Mainline releases

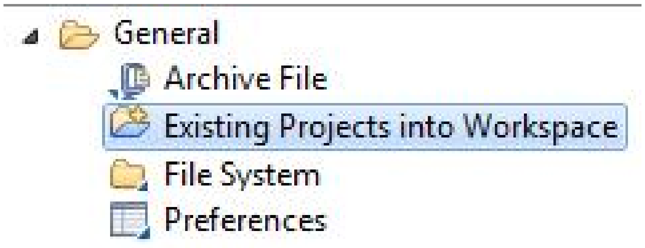
File Description	File Size	File Name
Add-on: IPv6 Evaluation(90-Day) for Kinetis SDK v1.3.0 with MQX RTOS - Linux	1.7 MB	IPv6 Evaluation - 90-Day for Kinetis SDK v1.3.0 with MQX RTOS - Linux.tar.gz
Add-on: IPv6 Evaluation(90-Day) for Kinetis SDK v1.3.0 with MQX RTOS - Mac OS	1.9 MB	IPv6 Evaluation - 90-Day for Kinetis SDK v1.3.0 with MQX RTOS - Mac OS.dmg
Add-on: IPv6 Evaluation(90-Day) for Kinetis SDK v1.3.0 with MQX RTOS - Windows	1.7 MB	IPv6 Evaluation - 90-Day for Kinetis SDK v1.3.0 with MQX RTOS - Windows.exe
Installer: Kinetis SDK 1.3.0 Mainline - Linux	662.6 MB	Kinetis SDK 1.3.0 Mainline - Linux.tar.gz
Installer: Kinetis SDK 1.3.0 Mainline - Mac OS	638.1 MB	Kinetis SDK 1.3.0 Mainline - Mac OS.dmg
Installer: Kinetis SDK 1.3.0 Mainline - Windows	304.5 MB	Kinetis SDK 1.3.0 Mainline - Windows.exe

10. Install it following all the default options, it will be installed into **C:\Freescale\KSDK_1.3.0**
11. Now launch KDS 3.2.0
12. Select **Help -> Install New Software**
13. Click on **Add...** button
14. Then click on **Archive** button
15. Now select from **C:\Freescale\KSDK_1.3.0\tools\eclipse_update** the file **KSDK_1.3.0_Eclipse_Update**
16. Select the package **KSDK 1.3.0 Eclipse Update**
17. Continue with the wizard. Accept the license agreement during the installation process.
18. Restart KDS

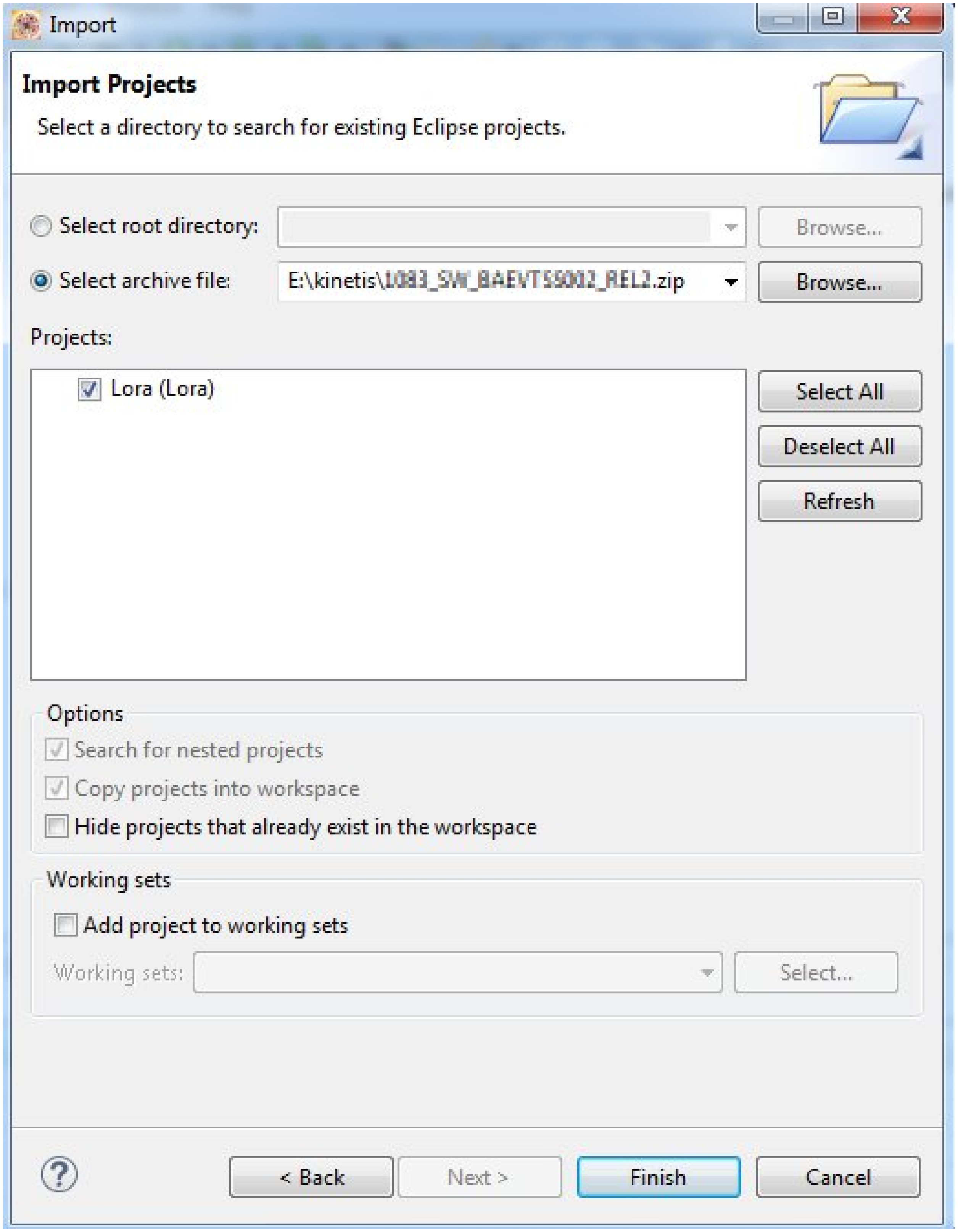
Now you are ready to import the project in your KDS.

Import Project

1. Download the project 1083_SW_BAEVTSS002_REL5b.zip.
2. Then go to **File->Import** and select **Existing Projects into Workspace**.



3. Browse to the zip file containing the project and select the project.



Import Projects

Select a directory to search for existing Eclipse projects.

○ Select root directory:

● Select archive file: E:\kinetis\1083_SW_BAEVTSS002_REL2.zip

Projects:

☑ Lora (Lora)

Options

☑ Search for nested projects

☑ Copy projects into workspace

□ Hide projects that already exist in the workspace

Working sets

□ Add project to working sets

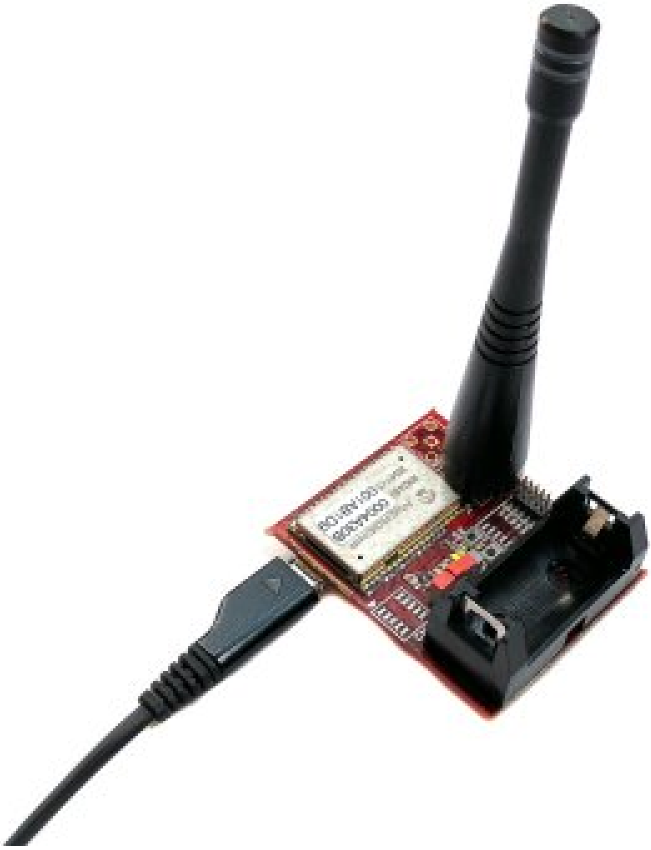
Working sets:

4. Press on **Finish**. Now you are ready to build and debug it. There are two ways to debug the board, the easiest is debug it via USB.

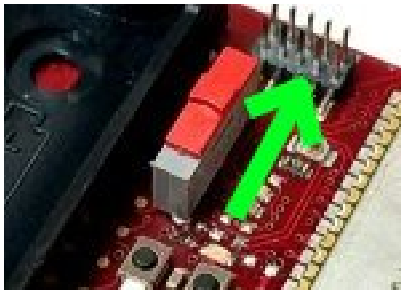
Build & Debug via USB

With the Mini-USB connector you are able to use **OpenOCD** inteface. It is necessary to install the mbed serial port driver.

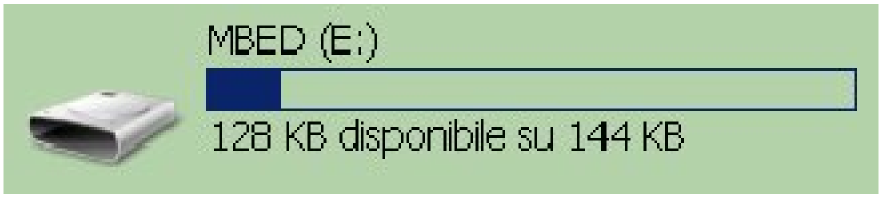
1. Go to the mbed website
2. Download the lastest mbed serial port driver, it is named as **mbedWinSerial**.
3. Install the driver.
4. After the installation, connect the lora-node board to the PC via the **CN2** connector.



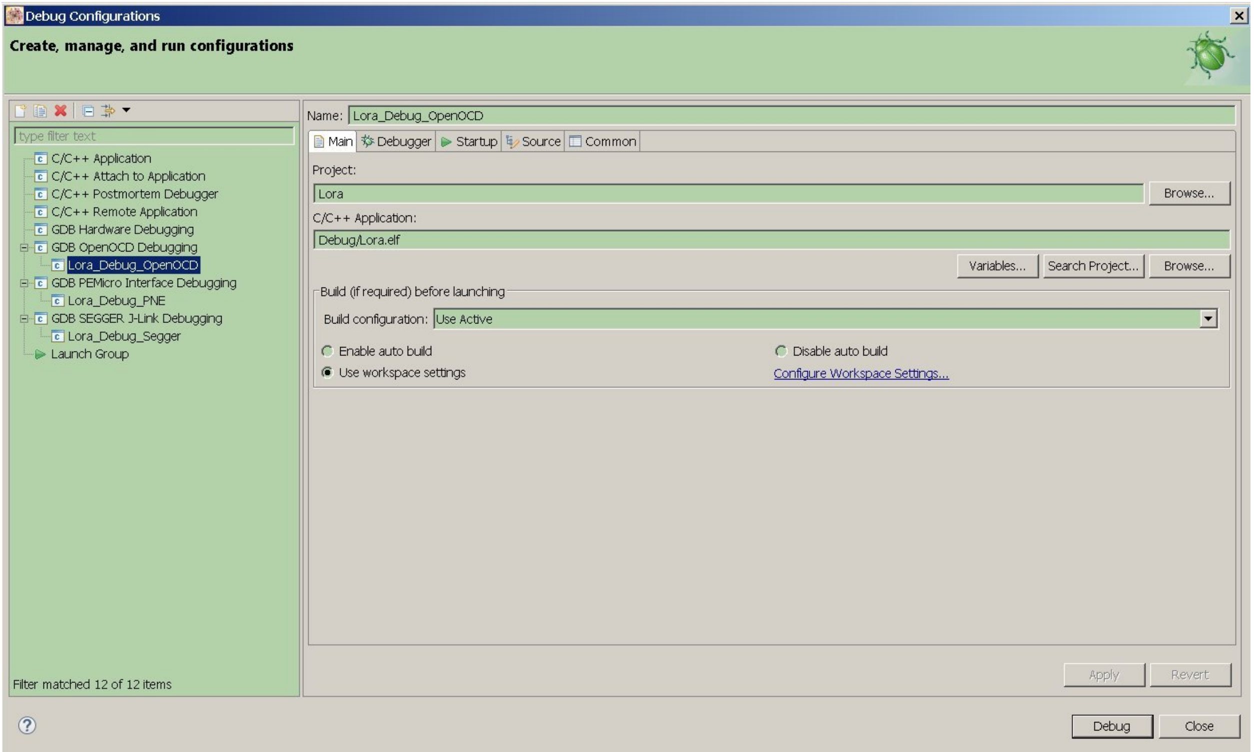
5. Turn On the board switching the **SW1** as in figure.



6. Windows will recognize it as a MBED mass storage device.



7. On the KDS from the menu click on **Project->Build All**, to compile the entire project.
8. Then click on **Run->Debug Confiuration->Lora_Debug_OpenOCD**.



Debug Configurations

Create, manage, and run configurations

Name: Lora_Debug_OpenOCD

Project: Lora

C/C++ Application: Debug/Lora.elf

Build (if required) before launching

Build configuration: Use Active

○ Enable auto build
○ Disable auto build
● Use workspace settings

Filter matched 12 of 12 items

9. Clicking on **Debug** button the debug will start entering on the first line code of the **main()** function.

Warning:

- Due some problem with KDS, sometimes you have to launch the debug twice before the KDS will run the debug correctly.
- During the debug session the sleeping mode doesn't work.

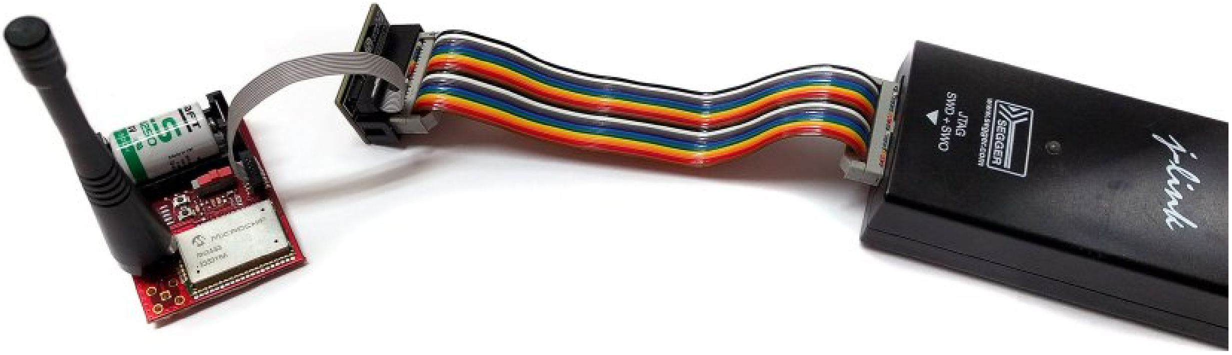
Build & Debug via SWD

1. Go to **Project->Build All**, to compile the entire project. In order to debug it connect the J-Link to the connector **CN4**.

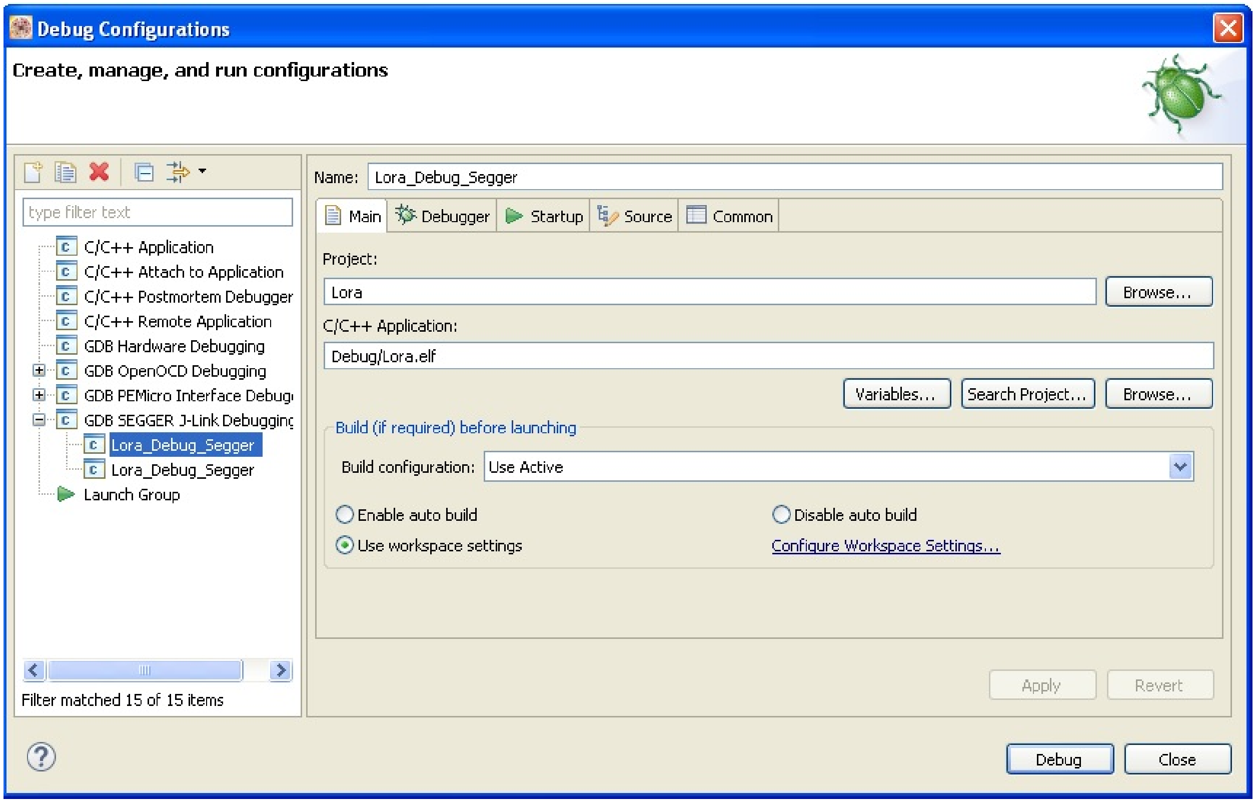
Note: Connector **CN4** is not fitted on the lora-node board. It is a connector **SMD MALE STRIP 2x5 P1.27mm**.

You have to fit it in order to debug the board via SWD.

2. The used debug interface is **SWD**. Then turn on the board switching the **SW1**.



3. Always on the KDS click on **Run->Debug Confiuration->GDB Segger J-Link Debug**.


Debug Configurations

Create, manage, and run configurations

Name: Lora_Debug_Segger

Project: Lora

C/C++ Application: Debug/Lora.elf

Build (if required) before launching

Build configuration: Use Active

○ Enable auto build ○ Disable auto build

● Use workspace settings Configure Workspace Settings...

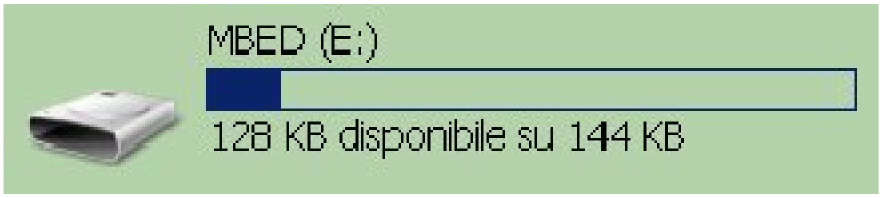
Filter matched 15 of 15 items

4. Clicking on **Debug** button the debug will start entering on the first line code of the **main()** function. During the debug session the sleeping mode doesn't work.

Reset Factory

From the release 1.4 you can reset the RN module throught the menu. Follow the next steps:

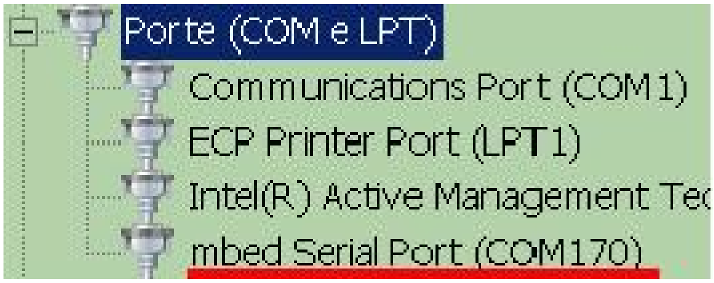
1. Download the file Lora_v15b.bin.zip and unzip it
2. Connect the board from your PC to the board via micro-USB cable. Open the MBED mass storage.



MBED (E:)

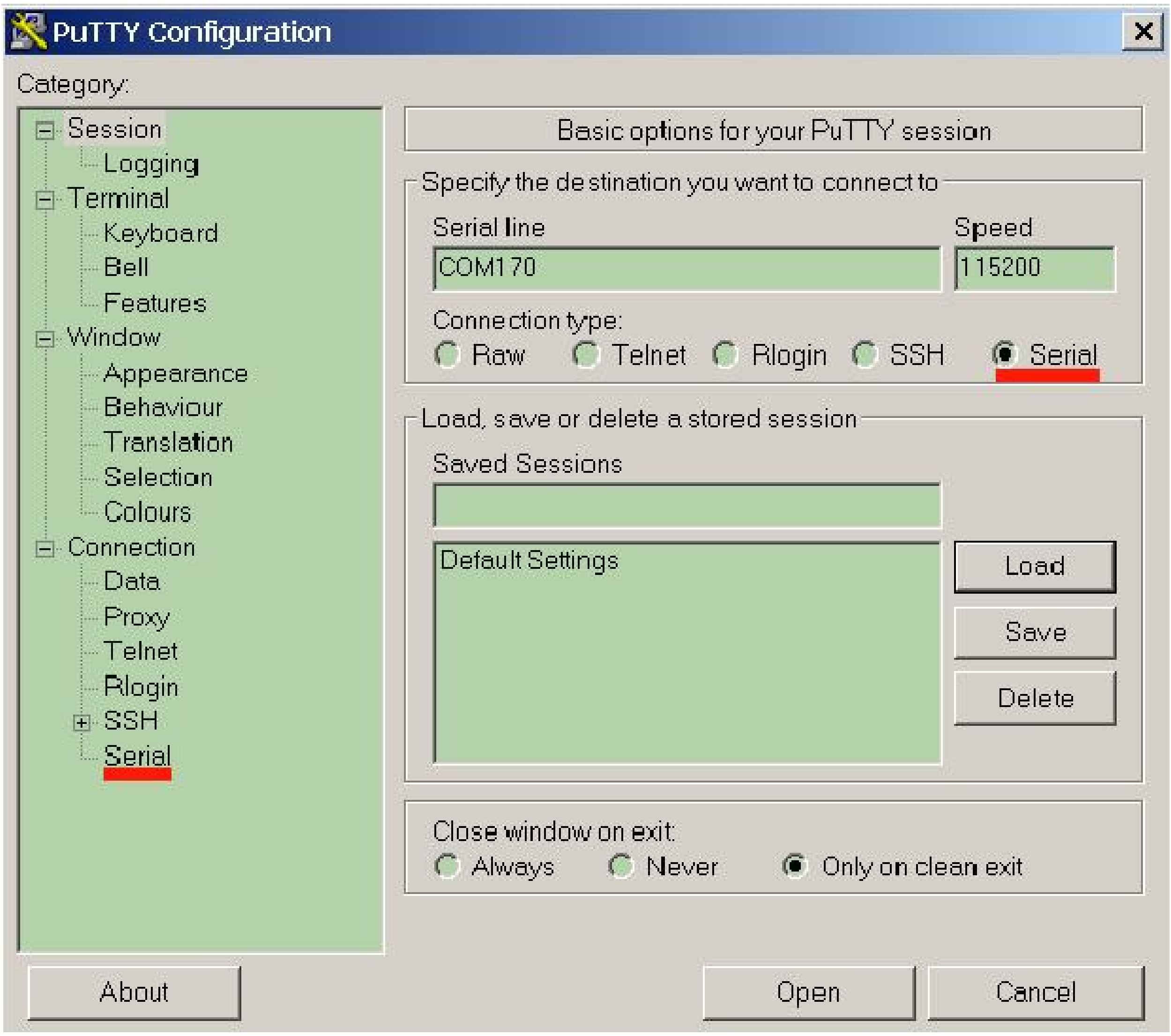
128 KB disponibile su 144 KB

3. Copy the **Lora_v15b.bin** file into the MBED mass storage
4. The Green led on the board will flash while the file is programmed into the device flash memory.
5. Now we have to connect a terminal to the Serial Com. First step, find which number is, going to "Device Manager" and finding the node called "Ports (COM and LPT)". In this example the number port is 170.



Porte (COM e LPT)
Communications Port (COM1)
ECP Printer Port (LPT1)
Intel(R) Active Management Tec
mbed Serial Port (COM170)

6. Open a terminal like **putty**, you can find it here. Launch it and set the properties signed in red in the figures:



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM170

Speed: 115200

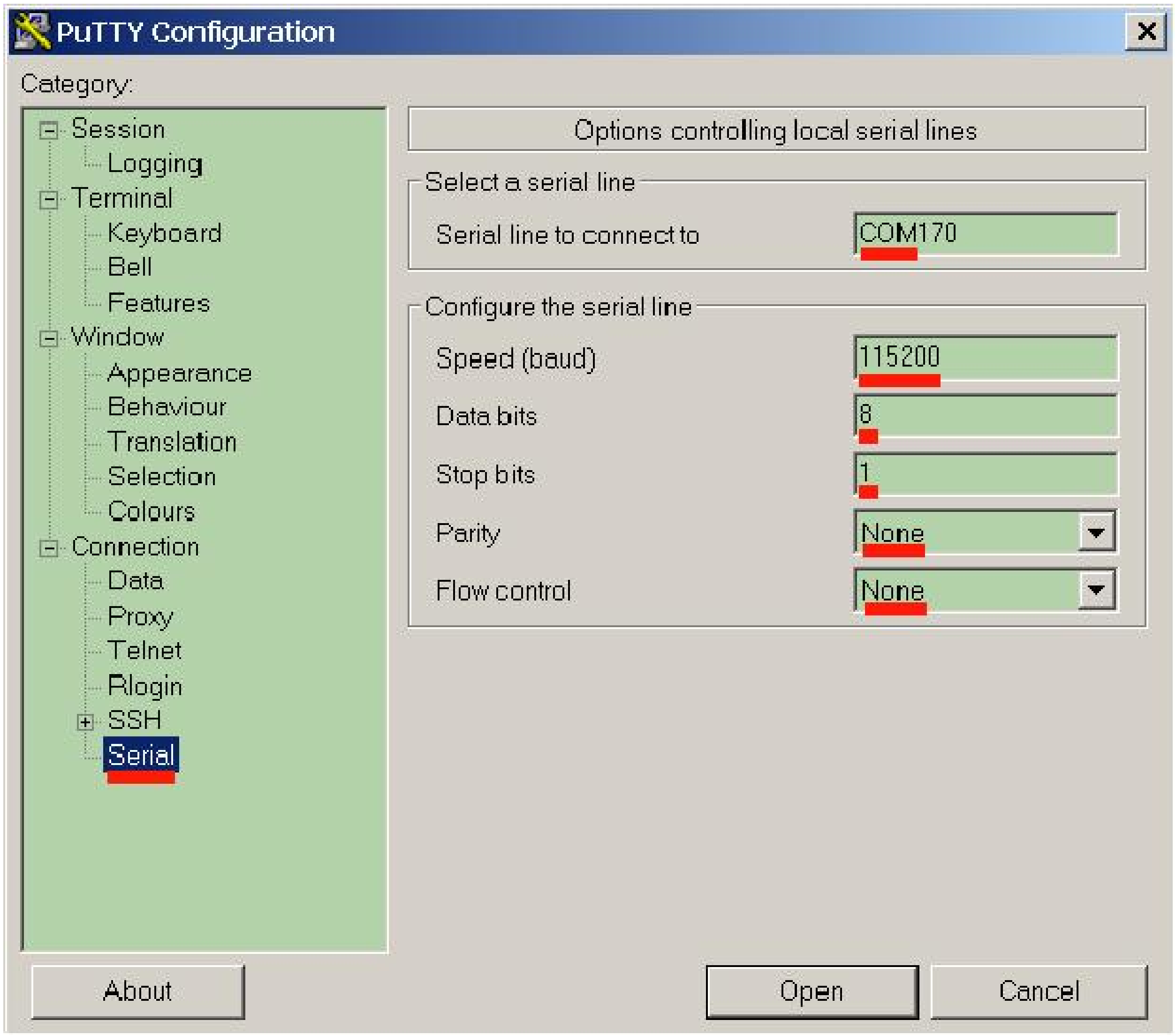
Connection type: ○ Raw ○ Telnet ○ Rlogin ○ SSH ● Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit



Options controlling local serial lines

Select a serial line

Serial line to connect to: COM170

Configure the serial line

Speed (baud): 115200

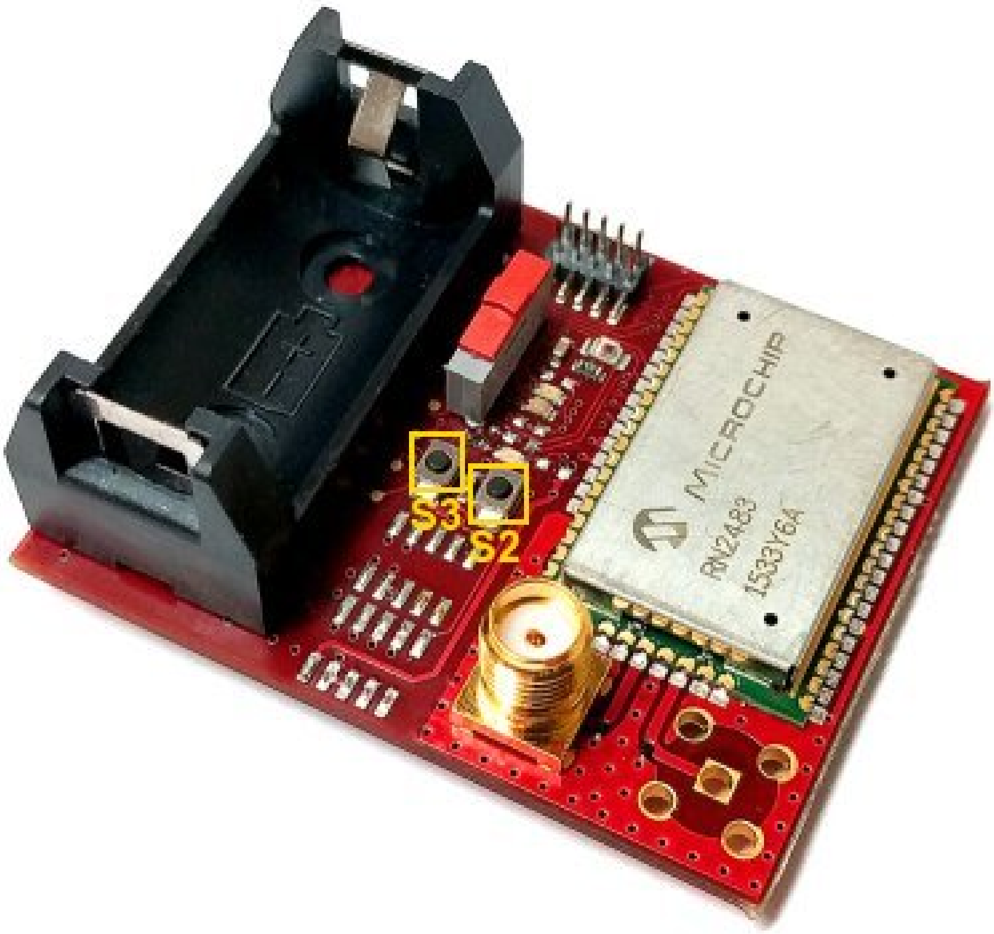
Data bits: 8

Stop bits: 1

Parity: None

Flow control: None

7. Press the **Open** button. Now reset the board pressing **S3**.



8. On the putty console you will see the starting boot text:

```
BAEVTSS002 firmware version 1.5b
RN2483 1.0.1 Dec 15 2015 09:38:06
```

9. Keeping the **S2** button pressed you will see after some seconds the text **Please release the button S2**. Will be displayed a menu like this one:

```
*** MENU CONFIG RN2483 ***
Enter 0 to print this menu
Enter 1 to see values in DEVEUI and DEVADDR registers
Enter 2 to modify DEVEUI
Enter 3 to modify DEVADDR
Enter 4 to modify APPSKEY
Enter 5 to modify NWKSKEY
Enter 6 to save changes
Enter 7 to Microchip default reset
Enter 8 to Avnet default reset
Enter 9 to exit
```

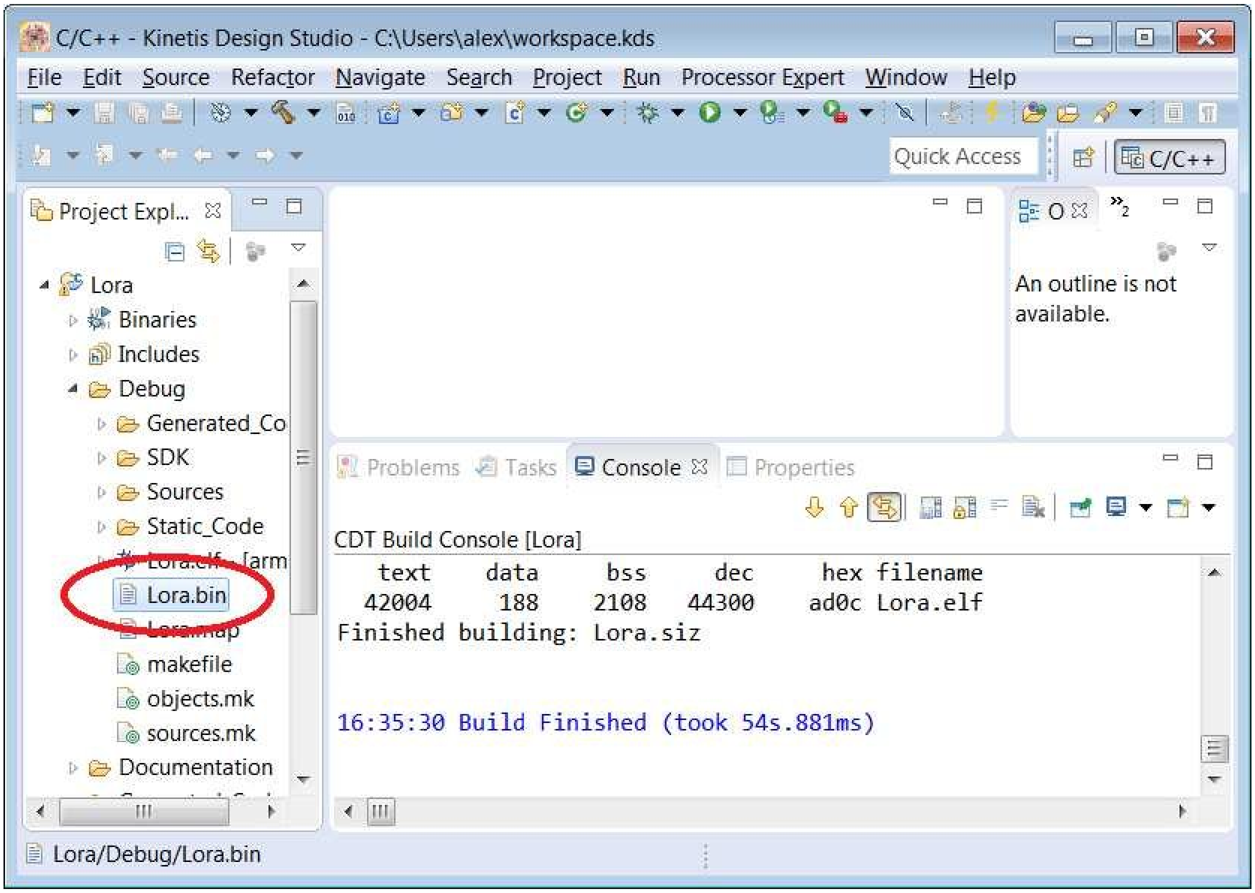
10. From the menu press the key **8** referred to *Enter 8 to Avnet default reset*. During the reset will be showed:

```
Starting Avnet default reset, please don't turnoff the board
```

11. Wait until will be showed the message **success**.

12. Now if you have to set correctly the DEVADDR register, see the paragraph **Update firmware** in *Quick start guide for firmware 1.4 or major*

Note: with the sources you can build the firmware 1.5b. You can generate the bin file to copy in the mass storage simpy using the Lora.bin



```
CDT Build Console [Lora]
   text    data     bss     dec     hex filename
  42004     188    2108   44300    ad0c Lora.elf
Finished building: Lora.siz


16:35:30 Build Finished (took 54s.881ms)
```

Processor Expert

The **Kinetis software development kit (KSDK)** is an extensive suite of robust hardware interface and hardware abstraction layers, peripheral drivers, RTOS abstractions, stacks, and middleware designed to simplify and accelerate application development on Freescale Kinetis MCUs. The addition of **Processor Expert** technology for software and board configuration provides unmatched ease of use and flexibility. Included in the Kinetis SDK is full source code under a permissive open-source license for all hardware abstraction and peripheral driver software. Mainline releases include support for a collection of Kinetis MCUs, whereas standalone releases offer support for one or a few additional Kinetis MCUs only. For details read KSDK page.

Hardware Guide

The board is provided with:

- NXP MKL26Z microprocessor
- NXP MK20DX128VFM5 microprocessor

- Microchip RN2483 Module
- Light Sensor
- Reset button
- Wake-up button
- NXP FXOS8700CQR1 Accelerometer and Magnetometer sensor
- LEDs which one is used directly by KL26Z

The Microchip RN2483 module provides LoRaWAN™ protocol connectivity using a simple UART interface. The NXP **MKLS26Z** is connected to the Microchip modules using the configuration 57600 8N1 without using RTS, CTS lines. Last version used is **1.0.1**.

The Light Sensor is read from the ADC converter peripheral of the MKL26Z.

The Accelerometer sensor is read from I2C interface.

The microcontroller uses the deep sleep mode **VLPS**, it is waken up by LPTimer every 30 seconds or by pin interrupt connected to the **S2** button.

Configuration RN2483

The connection used by **RN2483** is **ABP** (Activation by Personalization). To use this connection it is required to setup the RN device only one time. Every Lora Sensor Node bought is already configured. The commands used were:

> sys factoryRESET
> sys get hweui
> mac set deveui [hweui key read]
> mac set devaddr [last less significant hexs of hweui key]
> mac set appskey AFBECD56473829100192837465FAEBDC
> mac set nwkskey [hweui key repeated two times]
> mac save

After saving this setup is not required repeat the opeation of setup. In order to send data in the Lora network the two used commands are:

- **mac join abp**: used to join the Lora network
- **mac tx cnf 4 18AABBCC48DD0000**: used to send the frame *"18AABBCC48DD0000"* on the port 4

Datasheet and more

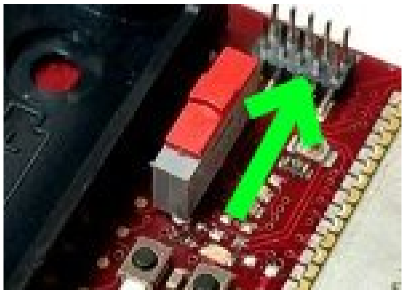
Please refer to Avnet Silica website.

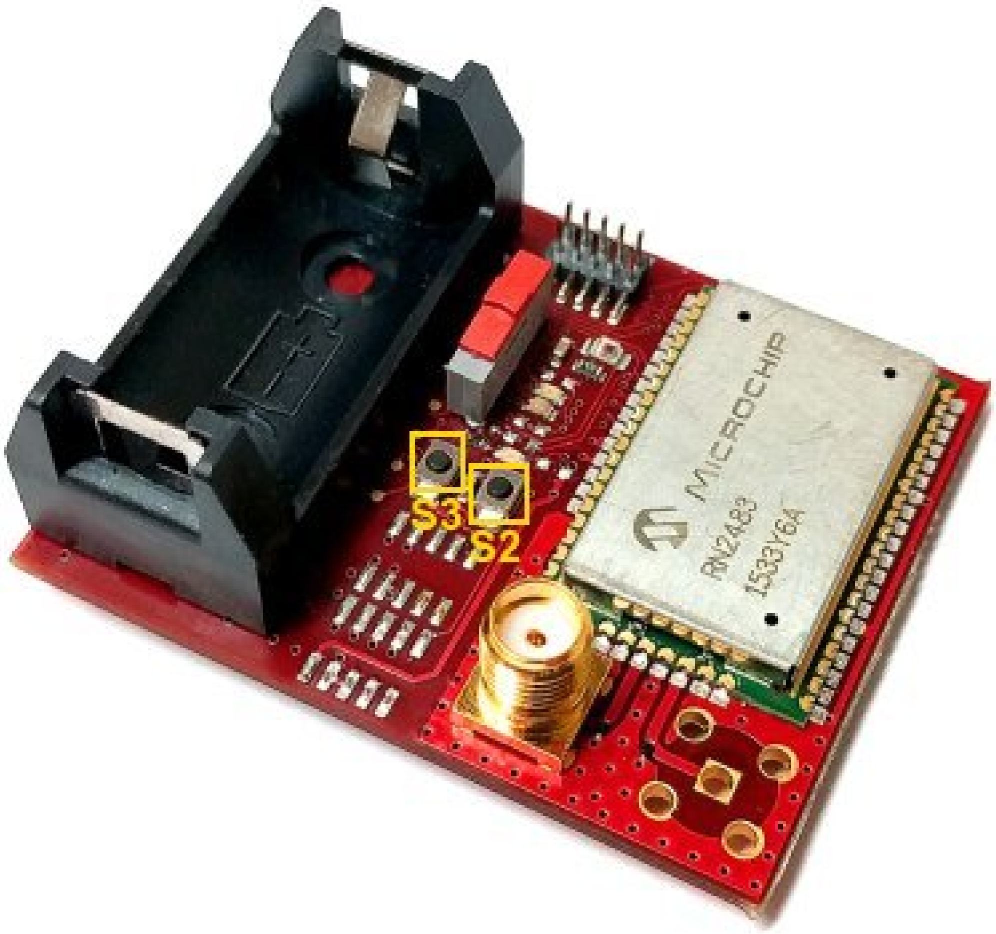
How to fix the MBED firmware

This procedure has to be followed only if the MBED mass storage it isn't recognized when you connect the board to the PC and the MBED drivers are installed in windows. The board uses the **MK20** microprocessor in order to implementing the MBED interface. In this guide we will upload the firmware for the **MK20**. In this way the MBED interface will work again.

1. Connect the board to the PC via USB cable.

2. Switch on the Lora Sensor Node keeping the **S3** button pressed.





3. If the previously steps are done correctly will appear a window named **BOOTLOADER**.
4. Download the file k20dx128_kl26z_if_mbed.bin.
5. And copy the file **k20dx128_kl26z_if_mbed.bin** into the **BOOTLOADER** mass storage.
6. The Green led on the board will blink while the file is programmed into the device flash memory.
7. Unplug and replug the board from the PC. If the procedure has been successfull will popup the **MBED** window.
8. Now with the **MBED** window it is possible update the firmware for the **KL26Z** microprocessor. See the *Quick start guide for firmware 1.4 or major* paragraph.

Index