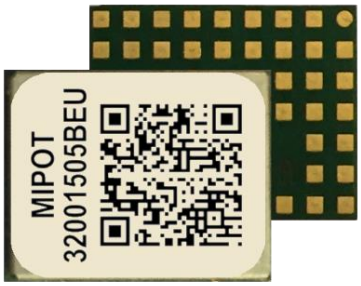
HOW TO SET UP THE 32001505BEU/BUS TO WORK WITH THE THINGS NETWORK

HOWTO



Description

This HOWTO will guide you through an example setup of the hardware and the Things Network console.

Contents

1. Overview

The LoRaWAN module will be controlled by a PC software connecting via the serial port.

The device will join the network with the OTAA (Over The Air Activation) method.

The required material is:

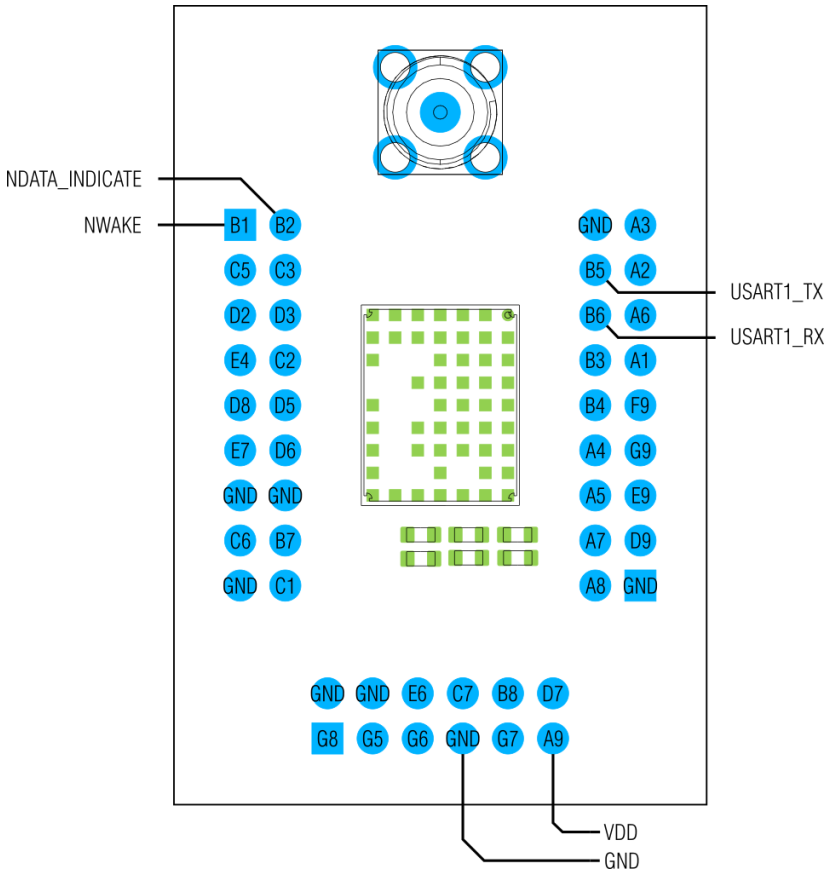
- 30001505BEU DevKit mounting a 32001505BEU (or 30001505BUS with a 32001505BUS)
- USB to UART 3v3 adapter (e.g.: FTDI TTL-232R-3V3)
- Power supply
- Mipot LoRaWAN GUI

To use the Thing Network, it's necessary to register and having a LoRaWAN gateway within range. If unsure about the presence of a LoRaWAN gateway, a new one has to be registered.

In this HOWTO, the screenshots will reference the EU version, for the US version, select the values appropriate for your locale.

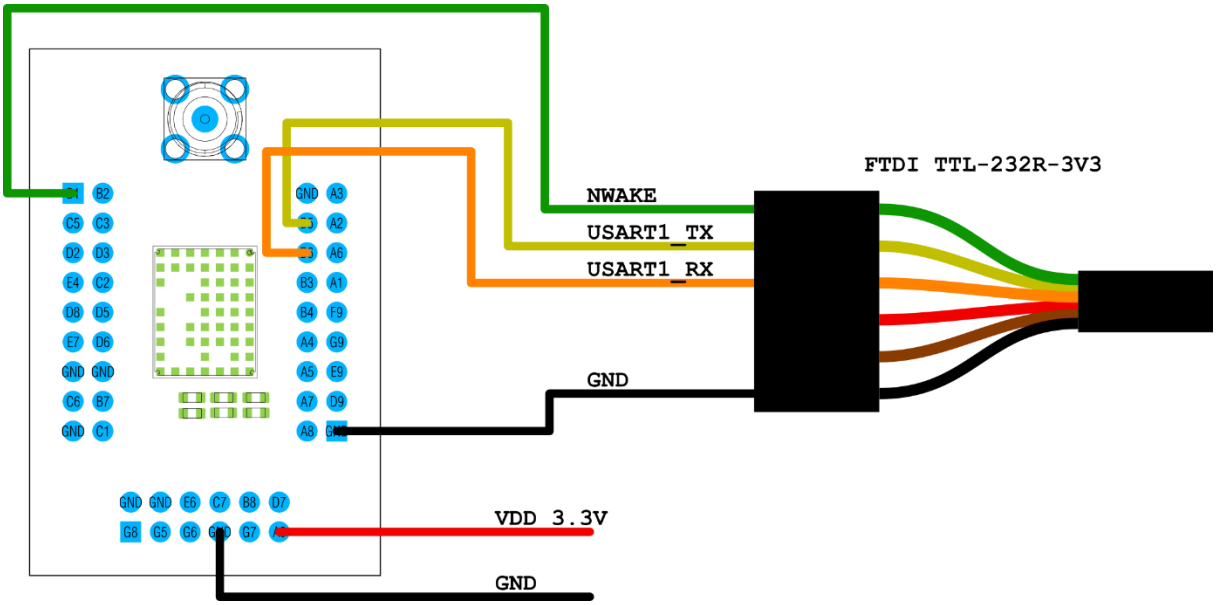
2. Hardware setup

The minimal connection with a host device uses the lines indicated in the following image and are comprised of the following pin:



PIN NAME	DIR	DESC
LPUART_TX	Out	UART TX pin, connect to RX pin of the adapter
LPUART_RX	In	UART RX pin, connect to TX pin of the adapter
NDATA_INDICATE	Out	Goes low when the module has data to send on the serial
NWAKE	In	Pull down to wake up the module from sleep.
VDD	Pwr	2.1 V to 3.6 V
GND	Pwr	Ground pin

2.1. Example of connection with USB to serial cable

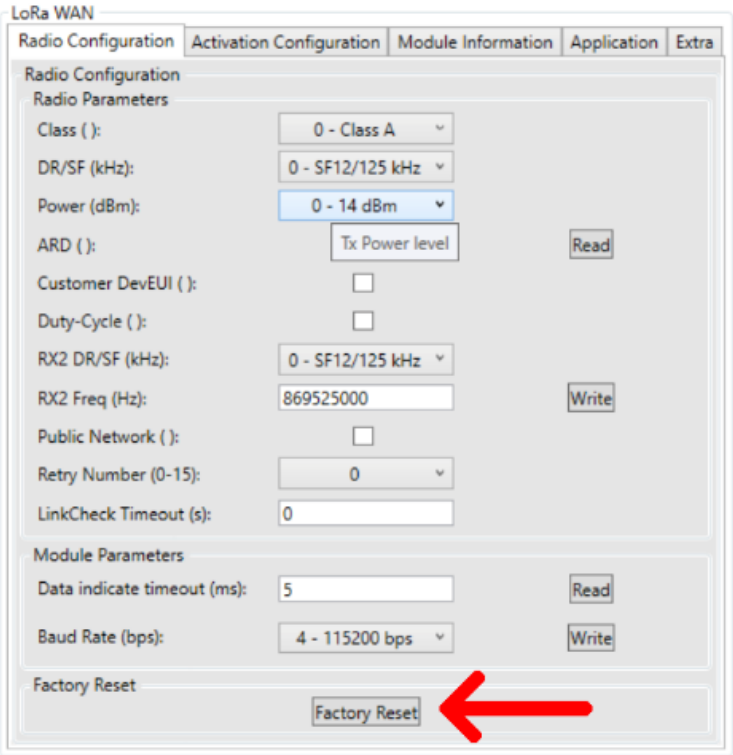


The NWAKE pin is connected to the RTS signal. The Mipot LoRaWAN GUI pull down the signal to wake up the module before sending data.

3. Software setup

3.1. Module reset

To start with a known configuration reset the module using the "*Factory Reset*" button in the "*Radio Configuration*" tab of the GUI.



This will configure the module with the default parameters as shown in the next table.

Parameter	Value
Class	0 (Class A)
DR/SF	0 (SF12, 125 kHz)
Power	0 (14 dBm)
ADR	1 (Enabled)
Duty Cycle Control	1 (Enabled)

Parameter	Value
Unconfirmed Tx repetitions	0
Enable customer EUI	0 (Disabled)
RX2 Data Rate	0 (SF12, 125 kHz)
RX2 Frequency	869525000 Hz[1]
Public Network Enable	0 (Private network)[2]

[1] 923300000 Hz for the BUS version

[2] This parameter will be changed to work with The Things Network

Command example

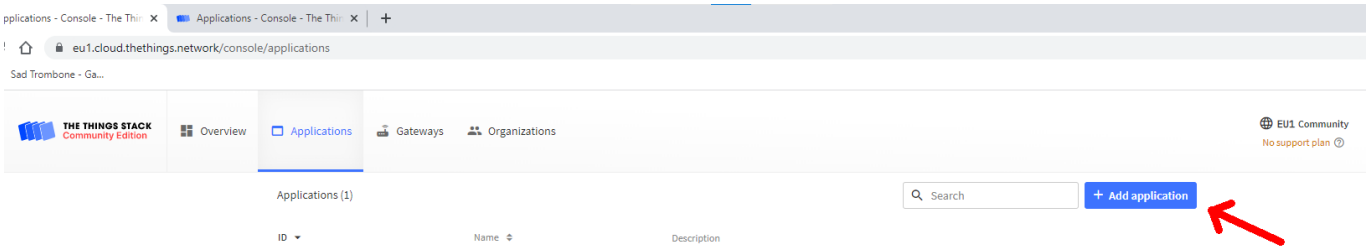
Reset the module to start with a known configuration of the module with the command FACTORY_RESET_CMD (0x31)

Host: 0xAA, 0x31, 0x00, 0x25

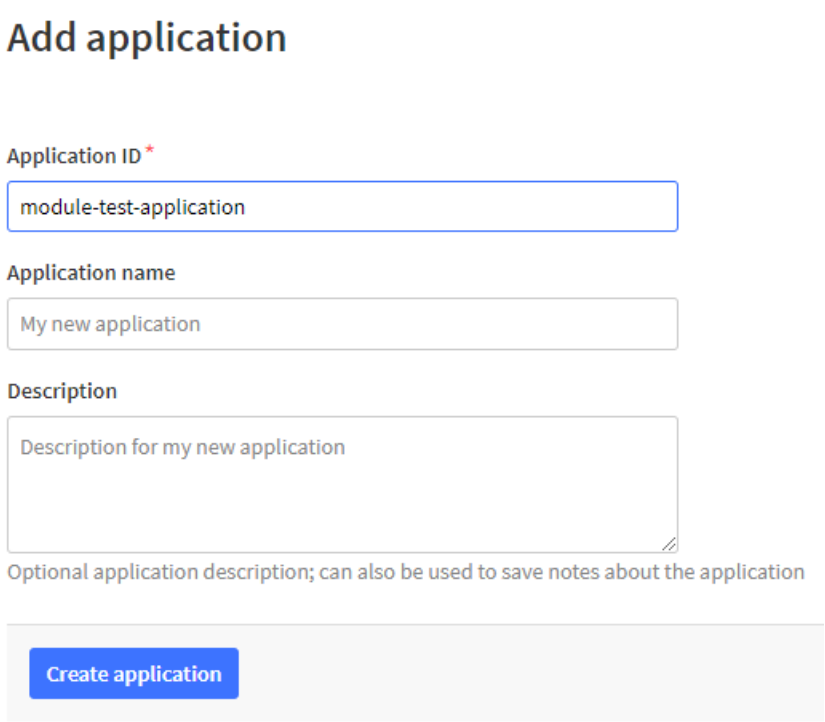
Device:0xAA, 0xB1, 0x01, 0x00, 0xA4

3.2. Creating the application

Log in in the Things Network Console, open the "*Applications*" section and click on "*Add Application*".



Write an application ID and click "*Create application*".



The only mandatory information is the "*Application ID*" and must contain only lowercase letters, numbers and dashes (-)

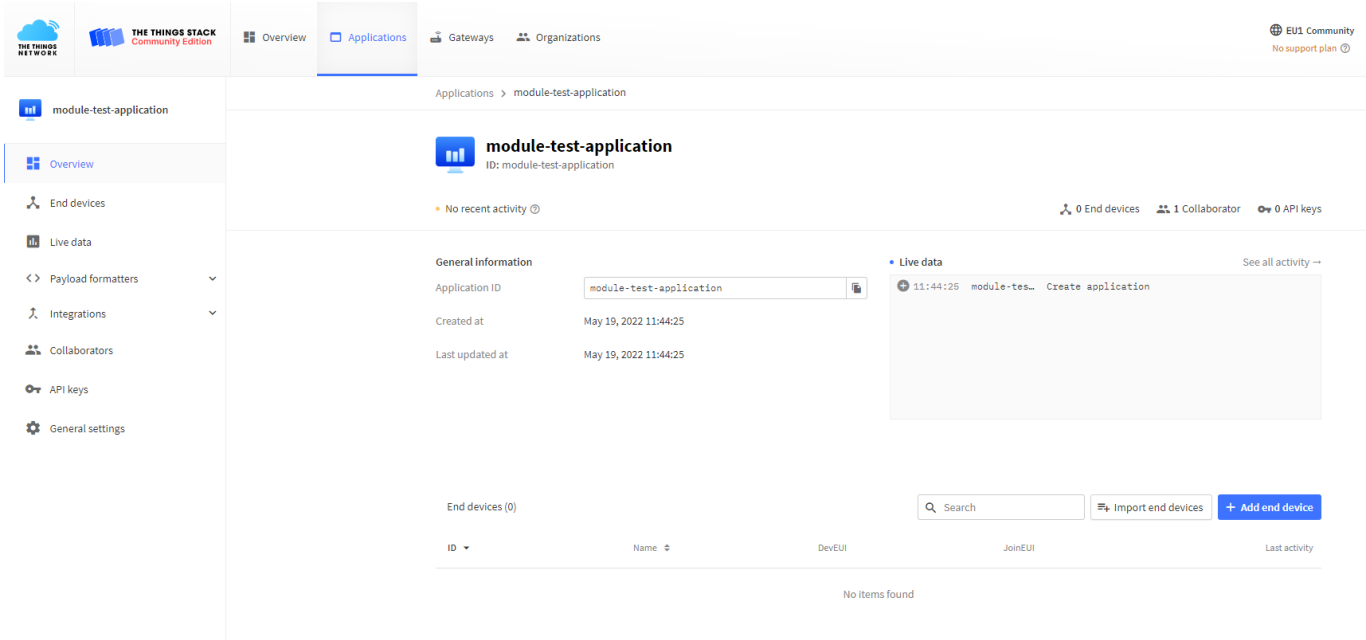
It's not possible to reuse an application id for more than one app, even if the old app is deleted.

The other fields can contain any text and are shown in the application list.

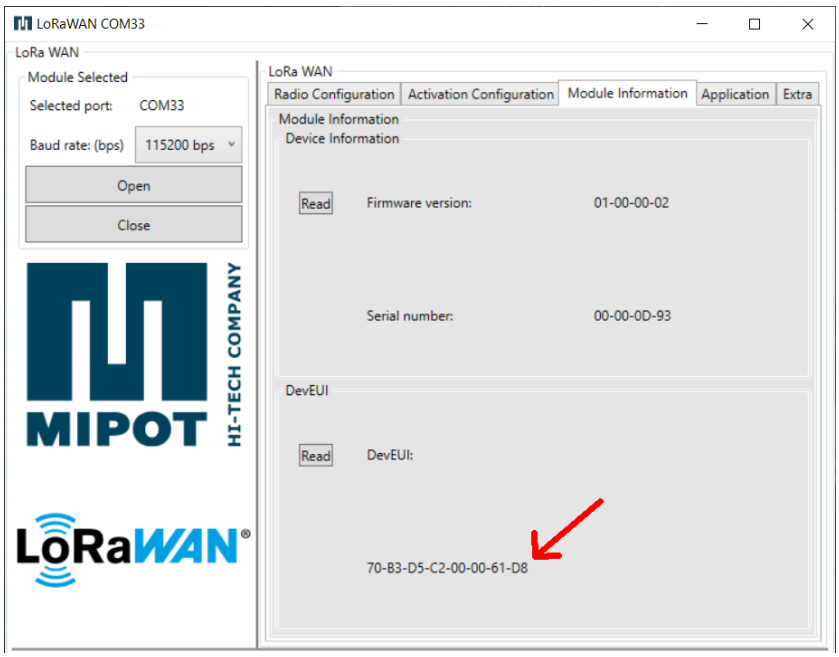
The "*Application name*" and "*Description*" can be changed later in the application "*General settings*".

3.3. Register an end device

Once created, the application page is loaded and from there it's possible to add a new device by clicking to "*Add end device*".



For the next step, it's necessary to know the DevEUI of the end node, which can be read from the Mipot LoRaWAN Demo GUI in the "*Module Information*" tab.



This is the devEUI of the module itself, but if needed the module can be configure to use a custom one.

Command example

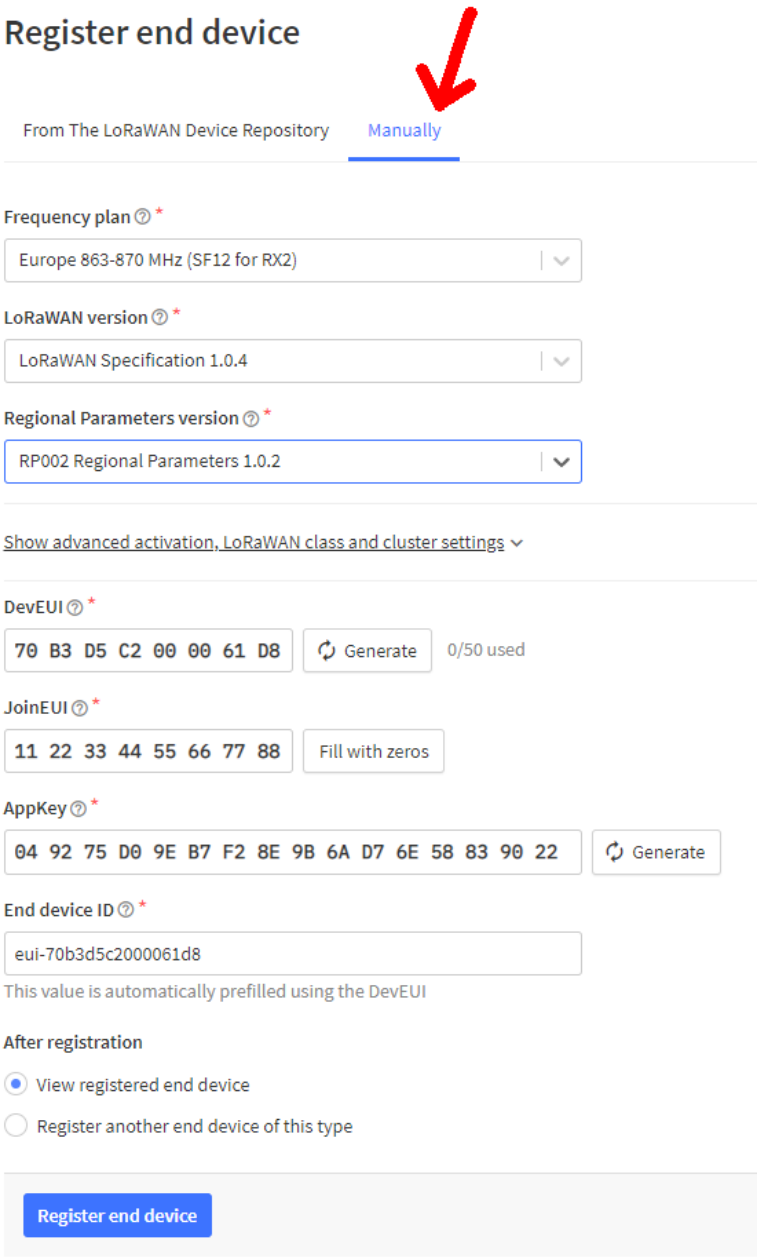
The DevEUI can be read with the GET_DEVEUI_CMD (0x36)

Host: 0xAA, 0x36, 0x00, 0x20

Device:0xAA, 0xB6, 0x08, **0xD8, 0x61, 0x00, 0x00, 0xC2, 0xD5, 0xB3, 0x70**, 0xA5

In this case, the DevEUI is **70 B3 D5 C2 00 00 61 D8**; if needed the module can be configure to use a custom one.

On the TTN's Console, configure the parameters of the new device using the manual configuration.



Select the manual configuration to open the view with all the parameters.

Configure the "*Frequency plan*", "*LoRaWAN version*", and "*Regional Parameters version*" as in the image.

For the EU version SF12 for RX2 is the default configuration of the module, and indicates the spreading factor to use by the gateway when sending a message in the RX2 window.

For the US version, select the frequency plan appropriate for your locale.

The Dev EUI is the value taken from the Module's GUI.

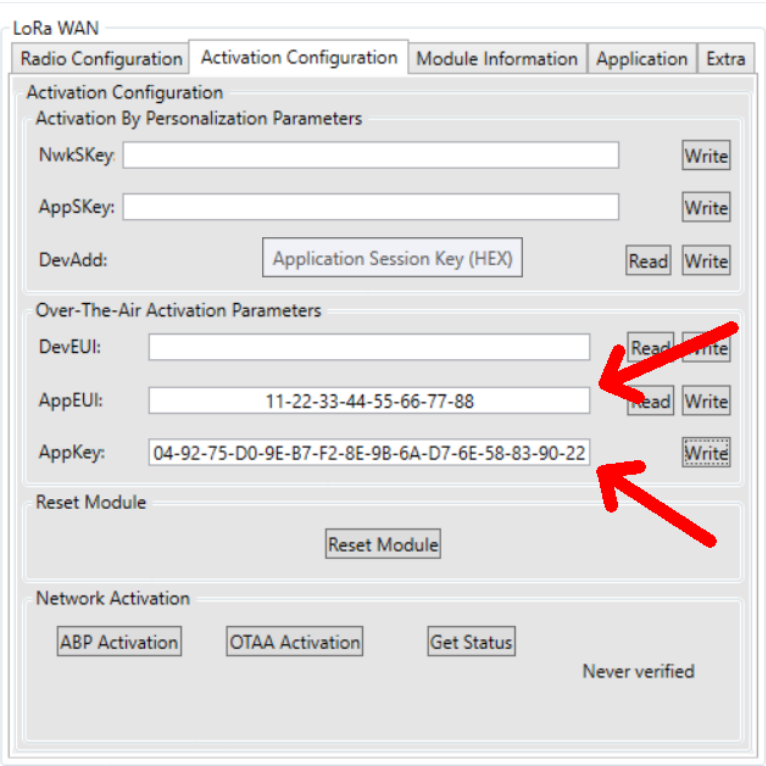
The JoinEUI is a value used to identify the join server. Since the module is programmable, it's possible to write any number except all zeros. Take note of this number as it will be needed to configure the module.

The AppKey is used to create the session key and must be a random number. Take note of this number as it will be needed to configure the module.

Once everything is configured, click on "*Register end device*" to complete the procedure.

3.4. Configure the radio

The module needs to be configured with the correct values of JoinEUI and AppKey to join the network server.



Using the GUI, write the AppEUI and the AppKey in the appropriate fields in the "*Activation Configuration*".

Command example

The AppEUI value is stored in the EEPROM memory starting at the address 0x08 and can be written using the EEPROM_WRITE_CMD (0x32).

In the previous page, on the TTN console the JoinEUI/AppEUI has been set to **11 22 33 44 55 66 77 88**.

Host: 0xAA, 0x32, 0x09, 0x08, **0x88, 0x77, 0x66, 0x55, 0x44, 0x33, 0x22, 0x11**, 0xAF

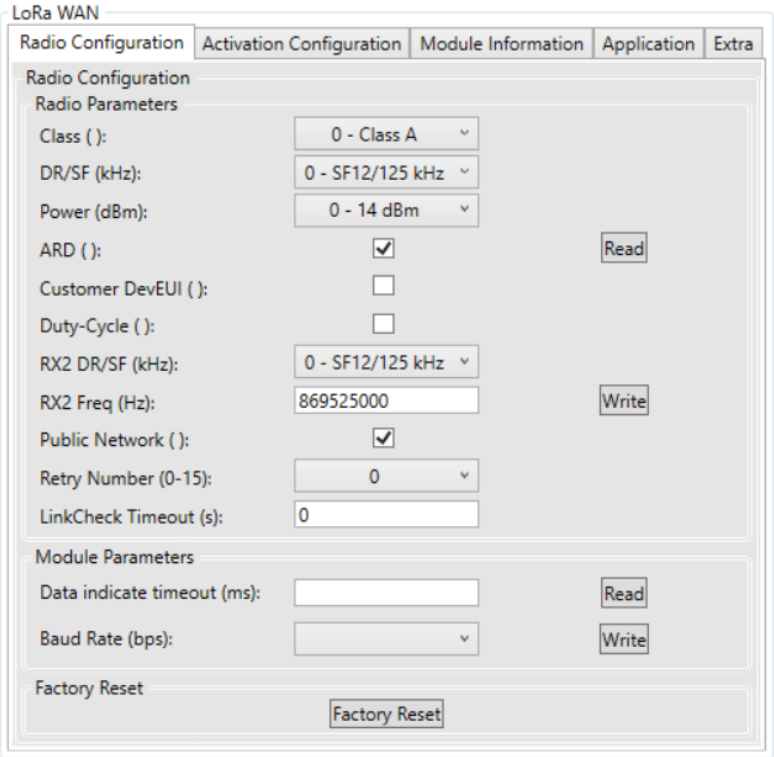
Device:0xAA, 0xB2, 0x01, 0x00, 0xA3

The AppKey is written with the SET_APP_KEY_CMD (0x43).

Host: 0xAA, 0x43, 0x10, **0x22, 0x90, 0x83, 0x58, 0x6E, 0xD7, 0x6A, 0x9B, 0x8E, 0xF2, 0xB7, 0x9E, 0xD0, 0x75, 0x92, 0x04**, 0x7C

Device:0xAA, 0xC3, 0x01, 0x00, 0x92

3.5. Enable the public network



To set the radio configuration, open the “Radio Config” tab and read the configuration.

Configure the parameters as shown in the image on the side and write them in the module.

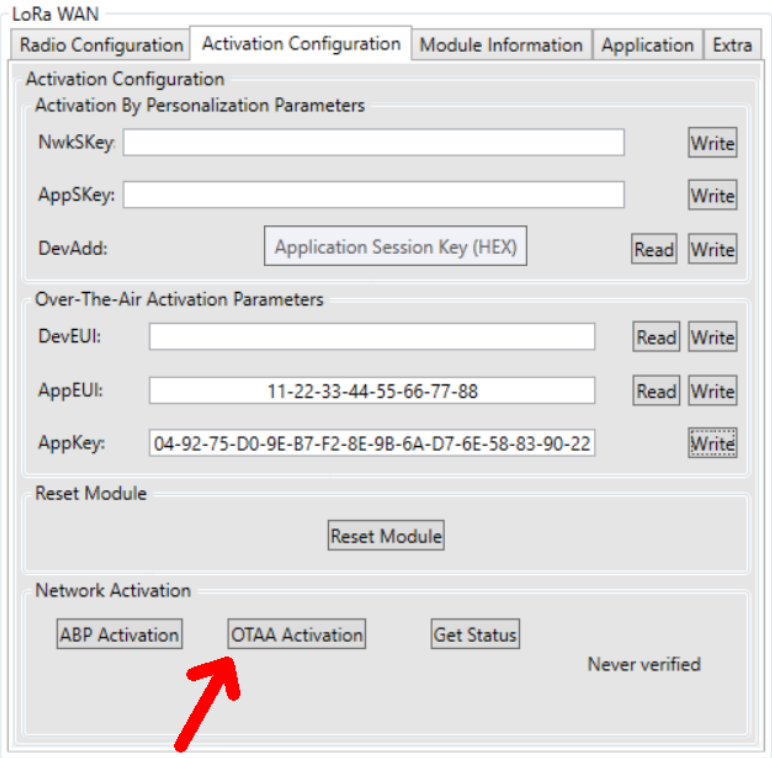
Command example

To enable the public network sync word writing the parameter in the EEPROM:

Host: 0xAA, 0x32, 0x02, 0x2E, 0x01, 0xF3

Device:0xAA, 0xB2, 0x01, 0x00, 0xA3

3.6. Join the network



To join the network server, click on "*OTAA Activation*" button in the "*Activation Configuration*" tab.

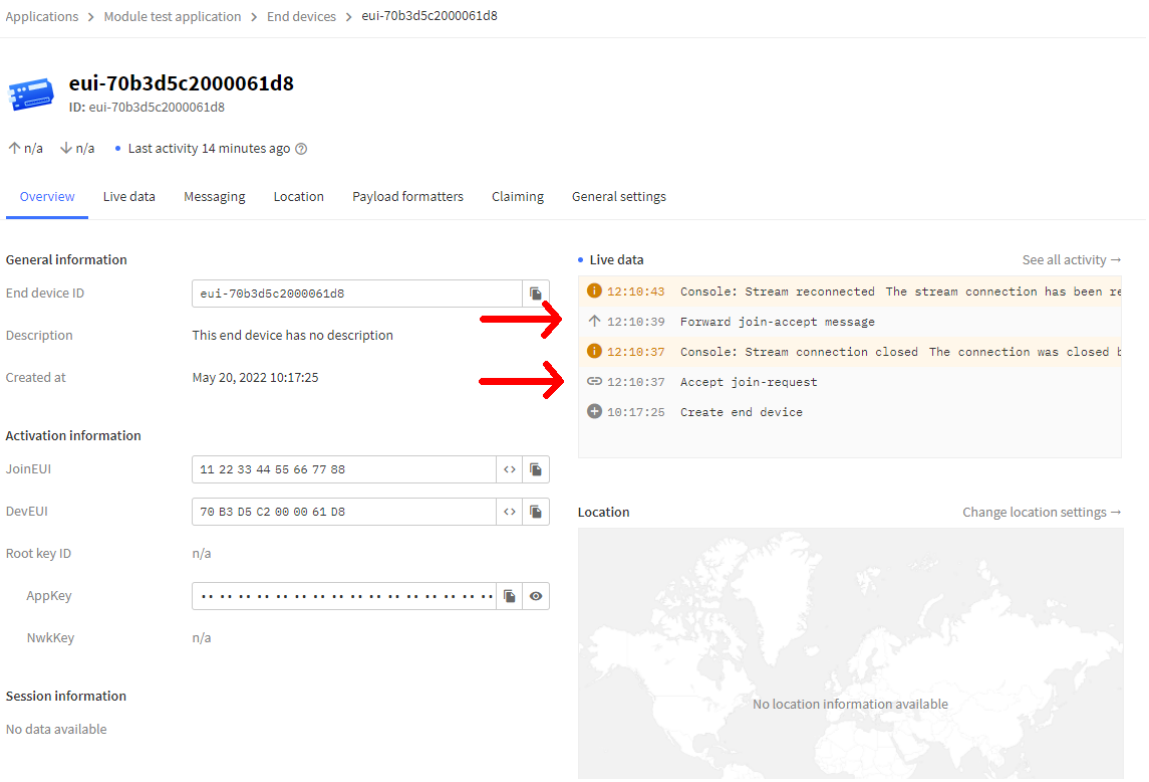
Command example

To join the network server, use the JOIN_CMD (0x40)

Host: 0xAA, 0x40, 0x01, 0x01, 0x14
Device:0xAA, 0xC0, 0x01, 0x00, 0x95

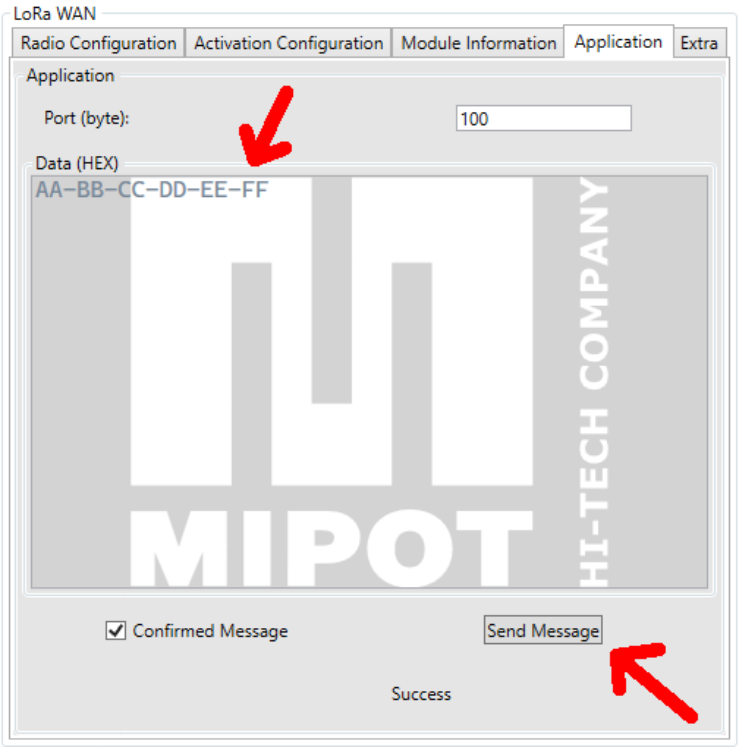
On confirmation the module sends the JOIN_IND(0x41)
Device:0xAA, 0x41, 0x01, 0x00, 0x14

On the end node page from the thing network console, it's possible to see the join event.



Now that the device has joined the network, it's possible to send and receive messages.

4. Sending a confirmed message



Use the "*Application*" tab to send a message.

Here it's possible to write the payload, select the port number and whether the message should be confirmed or not.

Command example

Send a message with the TX_MSG_CMD (0x46) with the “reliable data transmission” option.
For example to send the payload “**0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF**”:

Host: 0xAA, 0x46, 0x08, 0x01, 0x64, **0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF**, 0xA8
Device:0xAA, 0xC6, 0x01, 0x00, 0x8F

Once the procedure is complete, the module sends TX_MSG_CONFIRMED_IND (0x47) followed by a RX_MSG_IND (0x49):

Device:0xAA, 0x47, 0x05, 0x00, 0x00, 0x00, 0x01, 0x01, 0x08
Device:0xAA, 0x49, 0x0C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0xBC, 0xFF, 0x05, 0x00, 0x40

5. Sending an unconfirmed message

To send an unconfirmed message from the GUI, remove the relative tick.

Command example

Send a message with the TX_MSG_CMD (0x46) with the “unreliable data transmission” option.
For example to send the payload “**0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF**”:

Host: 0xAA, 0x46, 0x08, 0x00, 0x64, **0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF**, 0xA9
Device:0xAA, 0xC6, 0x01, 0x00, 0x8F

Once the procedure is complete, the module sends TX_MSG_UNCONFIRMED_IND (0x48) followed by a RX_MSG_IND (0x49):

Device:0xAA, 0x48, 0x03, 0x00, 0x05, 0x00, 0x06
Device:0xAA, 0x49, 0x0C, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x00, 0xBA, 0xFF, 0x0E, 0x00, 0x35

6. Receiving a message from the server

In this example the module is configured as a Class A device, so the module will open its receiving windows only after sending a message. If a downlink message has been scheduled, it will be returned in the RX_MSG_IND (0x49)

Command example

For example, the payload "**0x11, 0x 22, 0x33**" has been scheduled for downlink.

Host: 0xAA, 0x46, 0x08, 0x00, 0x01, 0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF, 0x0C

Device:0xAA, 0xC6, 0x01, 0x00, 0x8F

Device:0xAA, 0x48, 0x03, 0x00, 0x05, 0x00, 0x06

Device:0xAA, 0x49, 0x0F, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x01, 0xA5, 0xFF, 0x0B, 0x01, **0x11, 0x22, 0x33**, 0xE2

7. Revision History

Revision	Date	Description
0.1	6.6.2022	First version
0.2	7.10.2022	-Add command examples -Update screenshots to new GUI