
Adding LoRa® RN2483 Click to AVR-IoT WG Board

Introduction

Author: Marius Nicolae, Ioan Pop, Microchip Technology Inc.

This document describes an example application that uses basic LoRaWAN™ operations to transmit data from the sensors of the AVR-IoT WG Board. It was developed for the Microchip LoRa® Technology RN2483 modules on MikroElektronika LoRa click boards.

Every minute, the application sends light intensity and temperature data to the The Things Network (TTN) server. When the LoRaWAN node is not transmitting, the MCU core and the RN module are both put into a Sleep state. The source code for the AVR-IoT WG Board and the LoRa click was configured and generated using Atmel START.

Table of Contents

Introduction.....	1
1. LoRaWAN Ecosystem Overview.....	3
2. Hardware Description.....	4
2.1. Hardware Requirements.....	4
2.2. Hardware Description.....	5
3. Application Overview.....	8
3.1. Software Requirements.....	8
3.2. Project Architecture.....	8
3.3. Registering a Device to TTN.....	9
3.4. Peripheral Configuration.....	14
3.5. Getting the Keys from TTN.....	22
3.6. Data Rate and Spread Factor Modulation (SFM).....	23
4. Functional Description of the Application.....	24
5. Serial Communication with the RN2483 Device.....	26
6. Responses from RN2483 Device through USART.....	28
7. Node.js Installation and Application.....	30
8. Conclusion.....	33
9. References.....	34
The Microchip Website.....	35
Product Change Notification Service.....	35
Customer Support.....	35
Microchip Devices Code Protection Feature.....	35
Legal Notice.....	36
Trademarks.....	36
Quality Management System.....	37
Worldwide Sales and Service.....	38

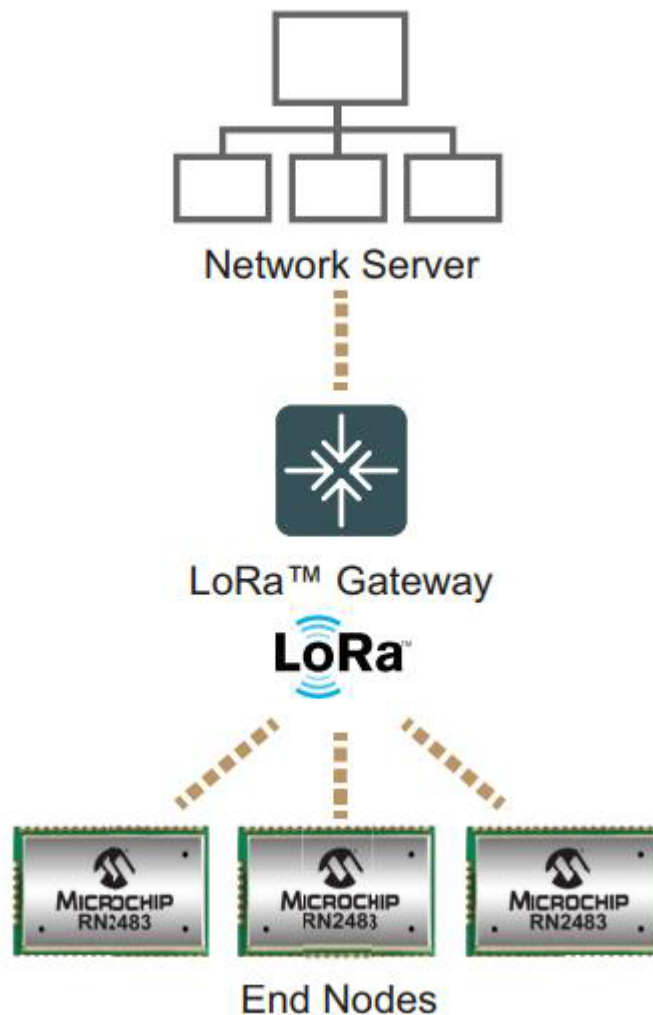
1. LoRaWAN Ecosystem Overview

LoRa is the physical layer or the wireless modulation used for the long range communication link. It uses chirp spread spectrum modulation to transmit over longer distances than older modulation techniques.

LoRaWAN represents the communication protocol and the system architecture for the network. It is based on a mesh architecture where devices send data and all the gateways in range transmit it to the network server. The network server makes the link to the application server.

More information about LoRa and LoRaWAN can be found [here](#).

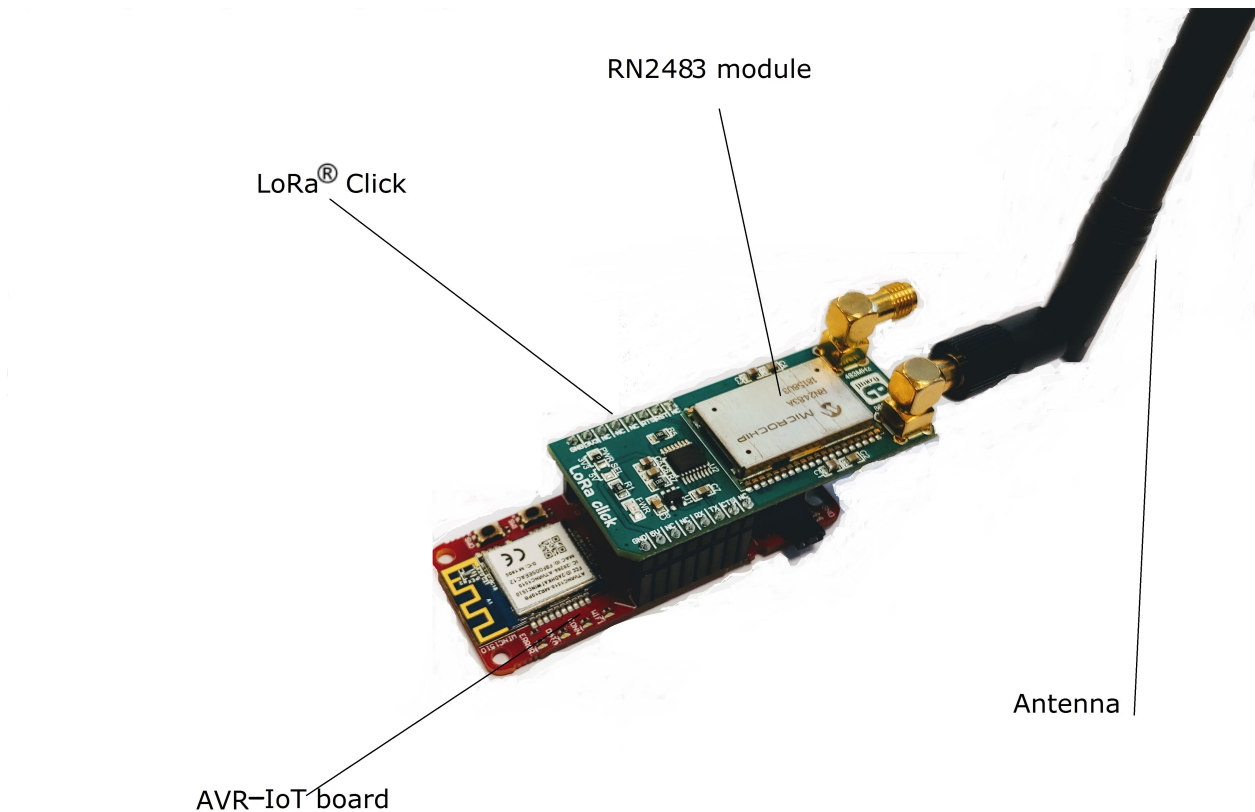
Figure 1-1. LoRaWAN™ Ecosystem



2. Hardware Description

2.1 Hardware Requirements

Figure 2-1. AVR-IoT and LoRa® Click



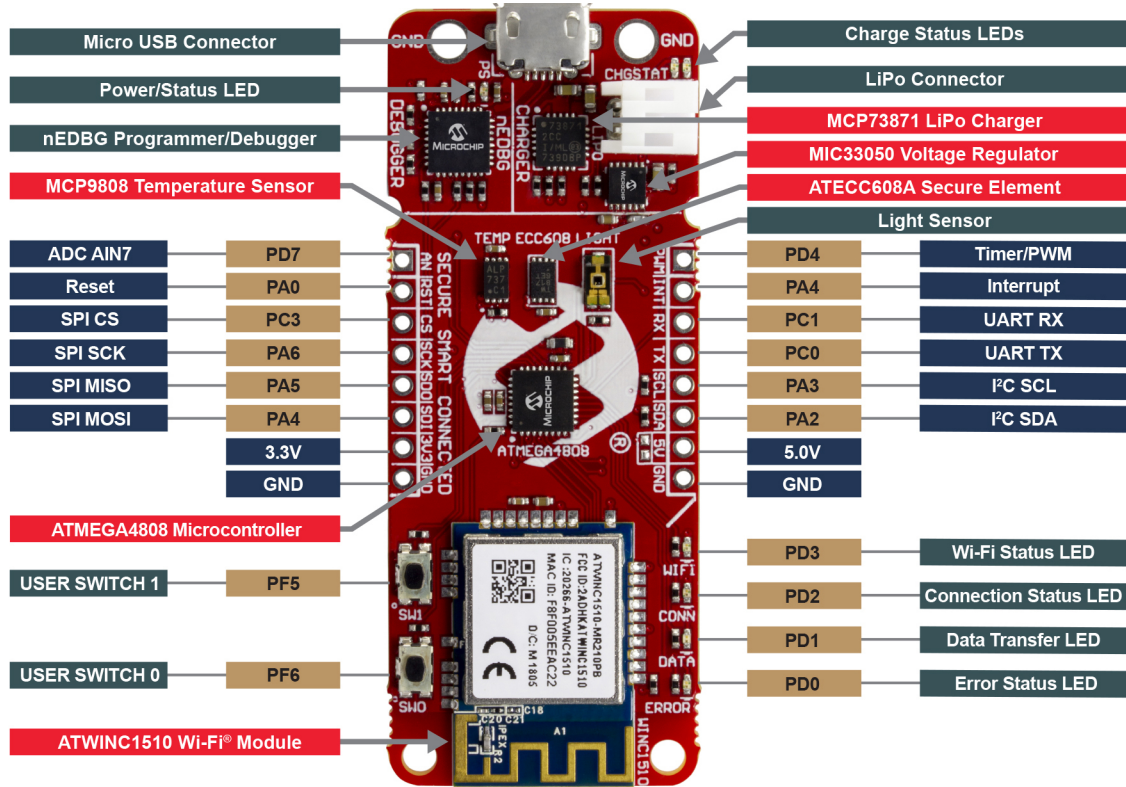
The following hardware is required to develop the LoRaWAN application:

1. AVR-IoT WG Board.
2. MikroElektronika LoRa click (MIKROE-1997), which features an RN2483 module. A MikroElektronika LoRa 2 (MIKROE-2225) click which features an RN2903 module, designed for North America, could also be used instead, but the user must note that the appropriate infrastructure (North American 915 MHz gateway) must also be part of the setup. For the purpose of this document, the LoRa click (RN2483) will be used. An antenna is recommended, but not required.
3. A LoRa™ Technology Gateway used for transmitting the information to the server. Any TTN-compatible gateway works. A list of compatible devices can be found on [their website](#). For the purpose of this document, The Things Gateway was chosen.

2.2 Hardware Description

2.2.1 AVR-IoT WG Board

Figure 2-2. AVR-IoT WG Board Description



The AVR-IoT WG Board is a development board for Internet of Things applications. The board features the following components:

- MCP73871 Li-Ion/LiPo Charger
- 5V to 3.3V MIC33050 Buck Converter
- nEDBG Debugger/Programmer
- WINC1510 Wi-Fi Module
- ATmega4808-MFR Microcontroller
- TEMT6000 Light Sensor
- MCP9808 Temperature Sensor
- ATECC608A CryptoAuthentication™ Device
- micro-USB Connection
- Four User LEDs
- Two User Push Buttons
- mikroBUS™ Header Footprint.

For this application, the ATWINC1510 module and the ATECC608A device will not be used. Additionally, ATWINC1510 will be put into a deep Sleep state, in order to reduce power consumption.

The board can be powered via USB or by a LiPo battery. The MIC33050 regulator regulates the USB 5V signal to 3.3V. The default configuration of the AVR-IoT WG Board delivers 3.3V to the mikroBUS.

The ATmega4808 microcontroller has many different peripherals that can be used for different applications. In this application, the following will be used:

- One USART instance for communication with the PC
- One USART instance for communication with the RN2483 module
- One I²C instance for temperature data acquisition
- One ADC channel for light sensor data acquisition
- One timer instance that counts while the microcontroller is in Sleep mode

2.2.2 MikroElektronika LoRa Click

Figure 2-3. LoRa® Click



The LoRa click board [MIKROE-1997](#) is an expansion board that comes with an RN2483 module. It is configured for 3.3V input voltage out of the box.

This Click board™ is LoRaWAN Class A compliant and provides a long-range spread spectrum communication with high interference immunity. The module used on this Click board is fully certified for LoRa Sub-GHz, 433/868 MHz band.

Fulfilling the LoRaWAN Class A specifications, the RN2483 module implements a transmission duty cycle of 1%. For example, if the module transmits a message which requires an on-air time of one second, it must not initiate any other transmission for the next 99 seconds.

Depending on the distance between the node and the gateway, different data rates can be used to increase or decrease the on-air time of a message. If the device is relatively close to the gateway, the highest data rate of five can be used. In order to send the message to a gateway that is further away, an antenna might be required.

The RN2483 module communicates with the AVR-IoT WG Board through a USART interface. The standard baud rate is 57600 symbols/s, but it can be changed through a command. The module receives commands that can change different parameters or initiate the transmission. It will also reply with different messages depending on what command has been sent to it. The list of supported commands for the RN2483 module can be found [here](#).

2.2.3 LoRa Gateway

Figure 2-4. The Things Gateway



Any gateway compatible with the TTN server can be used, but for this application The Things Gateway was used. Setting up the gateway is out of this document's scope. A guide for registering and activating this gateway on the TTN site may be found [here](#).

3. Application Overview

3.1 Software Requirements

Atmel Studio 7 Integrated Development Platform is required for programming, and an Internet connection is needed in order to use Atmel START.

Atmel START is a web-based software configuration tool for starting a new embedded development on Microchip SAM and AVR microcontrollers. Starting from either a new project or an example project, Atmel START enables the selection and configuration of a set of software components from the SAM Advanced Software framework for an embedded application in a usable and optimized manner. Atmel START supports code project generation for Atmel Studio 7, IAR Embedded Workbench, Keil μ Vision®, or generic makefile generation.

An account on the TTN site is needed. Details on how an account can be created can be found [here](#).

3.2 Project Architecture

The project source code is available on GitHub, under the following link:



The application is structured on several levels. The project was generated first in Atmel START and a detailed explanation of how to do this is provided further in the document. The code not generated by Atmel START can be found in the main directory of the project. The files are:

- `atmel_start.c`
- `atmel_start.h`
- `led.c`
- `led.h`
- `lora_handling.c`
- `lora_handling.h`
- `lora_keys.h`
- `low_power.c`
- `low_power.h`
- `main.c`
- `sensors_handling.c`
- `sensors_handling.h`

If the Atmel START generation steps are completed, these files can be copied to the project folder in order to have the full functionality of the application.

The `lora_keys.h` file needs to be changed by every user with their respective keys that are taken from the TTN site.

On the server side, all data storage and display are done by TTN in the user console in the application screen.

This application also offers the possibility of visualizing the data in a web browser through the use of a node.js® application. The node.js application runs a server on the user's machine which takes data from TTN and displays it in the browser.

3.3 Registering a Device to TTN

After the TTN account and application have been created, follow these steps to reach the location of the keys for authentication.

On the TTN site, click on the user profile and a menu will open. Click on **Console**.

Figure 3-1. TTN Main Site

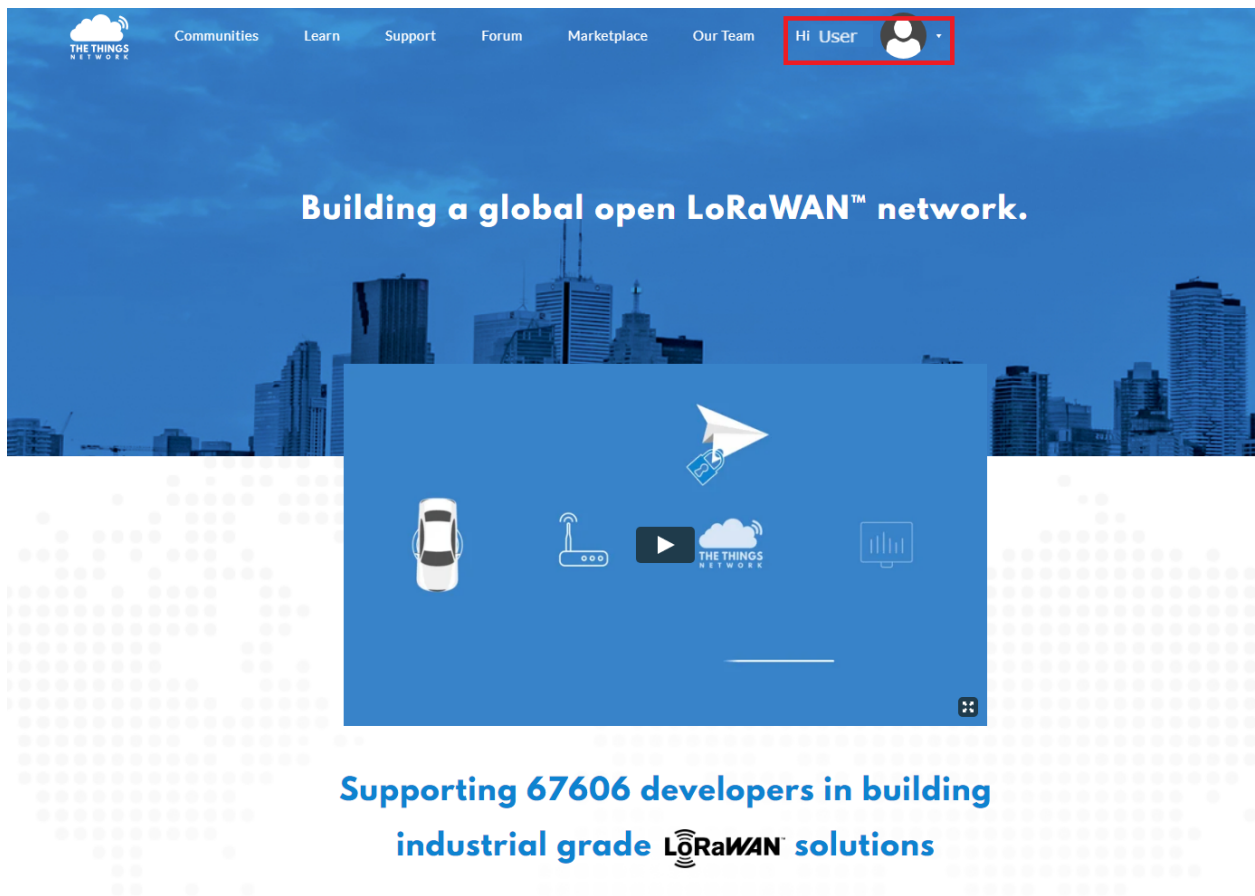
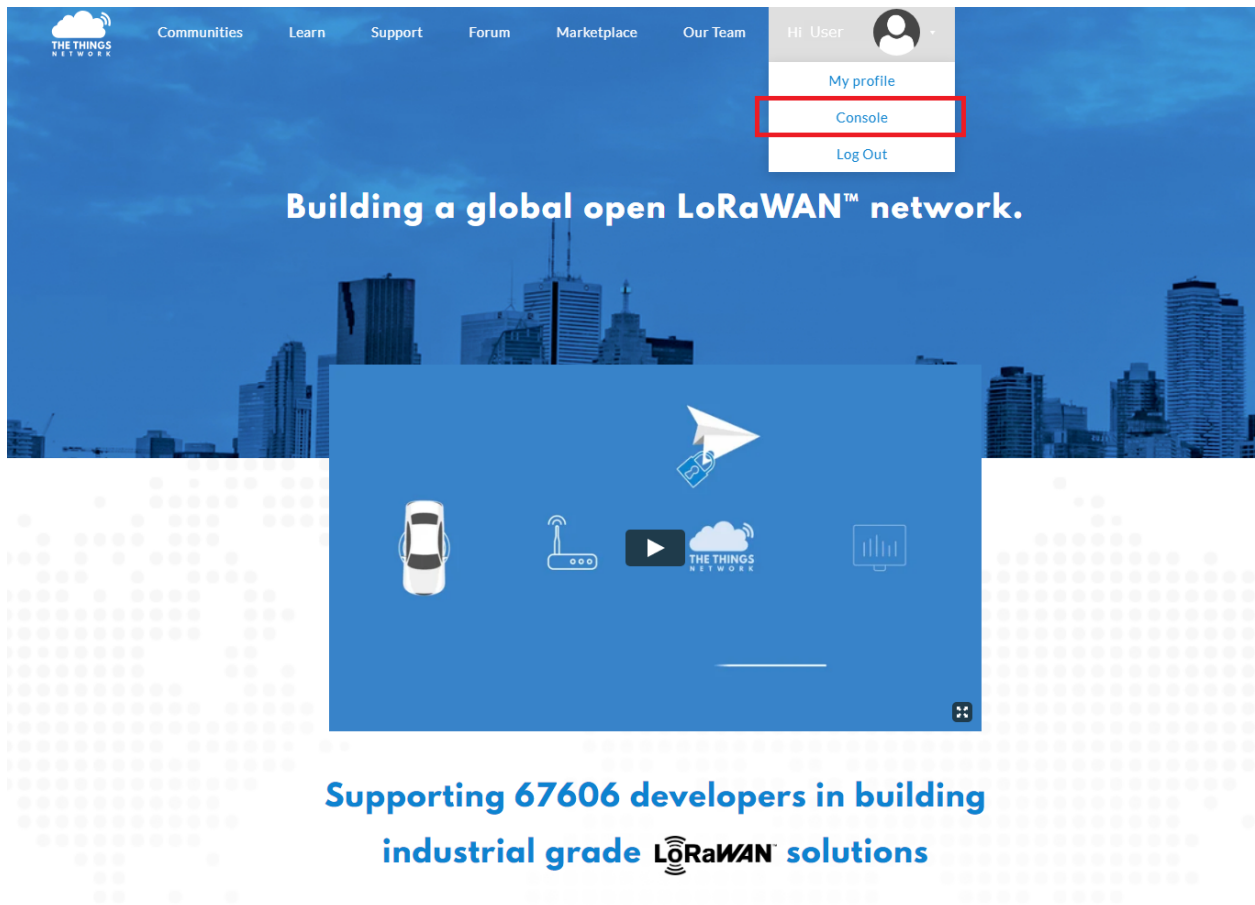
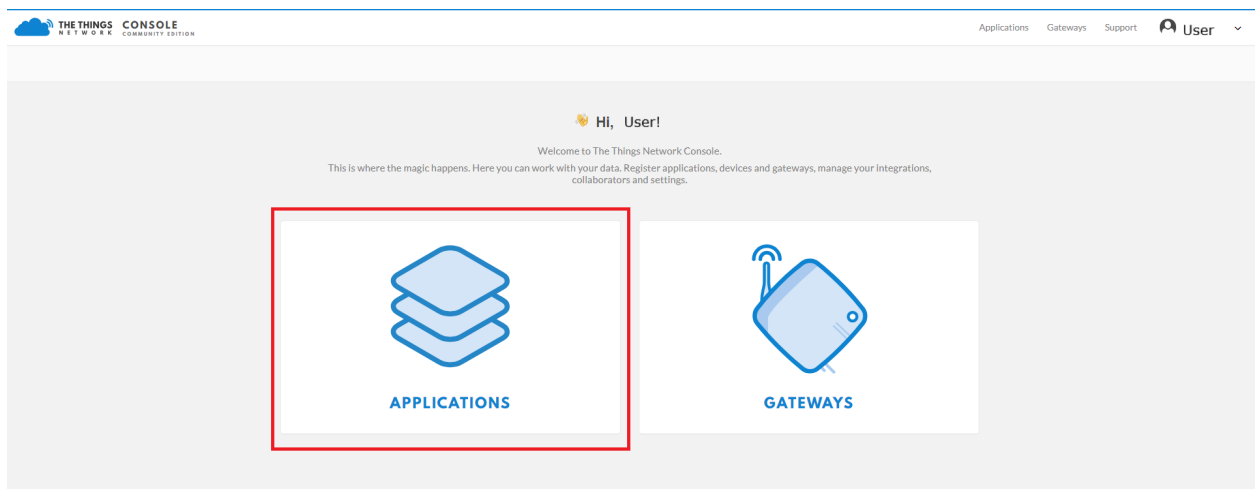


Figure 3-2. TTN User Menu



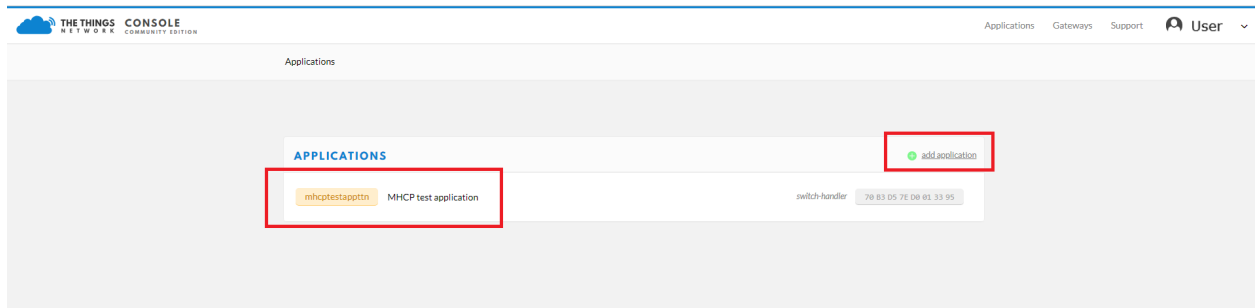
In the following screen, click on **Applications**.

Figure 3-3. TTN Console



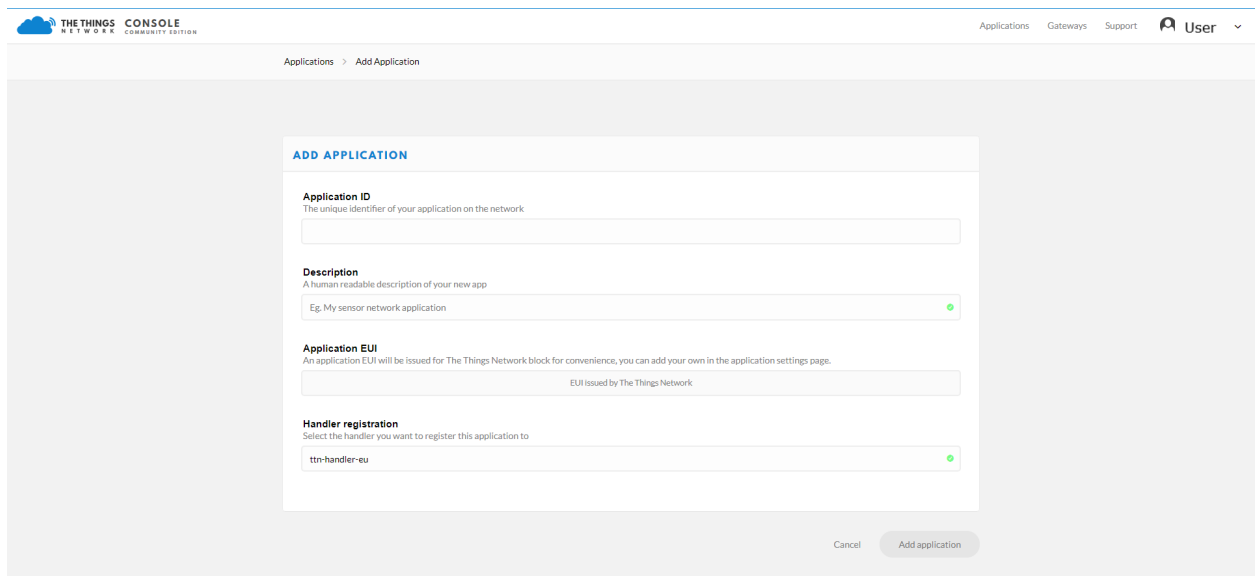
In the next screen, either add a new application or choose one that has already been configured.

Figure 3-4. TTN Applications



To create the application, input the desired name and ID and choose the handler. Switch handler is recommended, but other handlers might work in the user's region.

Figure 3-5. New TTN Application



After the application was created, go to the **Devices** tab in the application menu and click on **register device**.

Figure 3-6. Application Interface

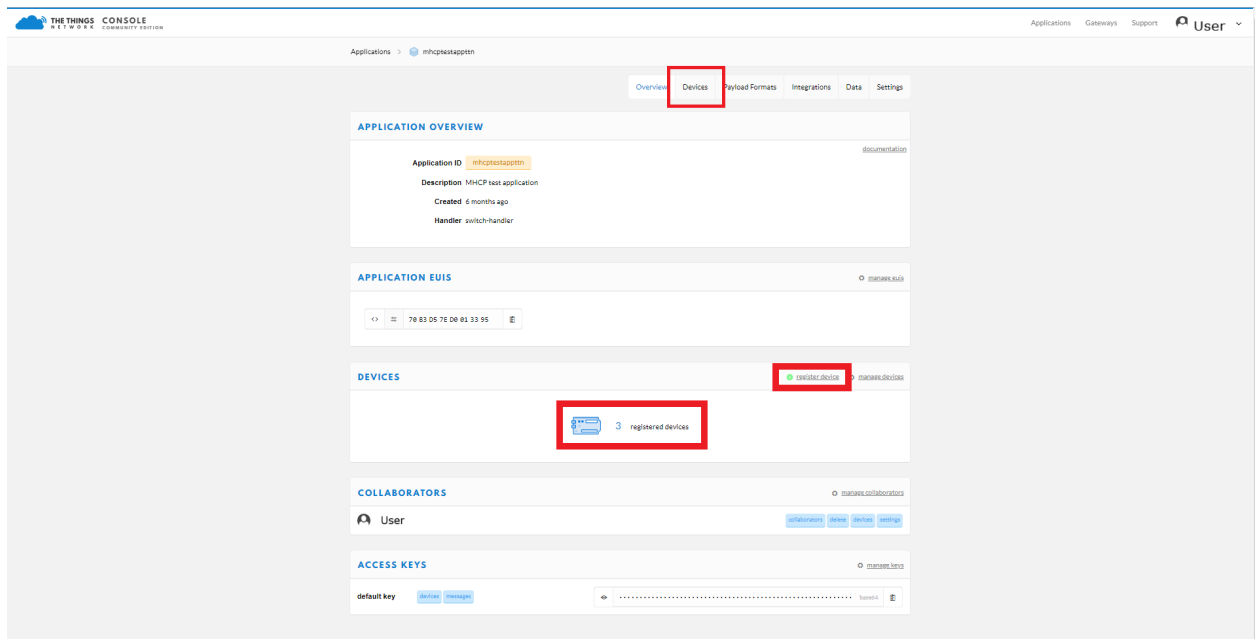
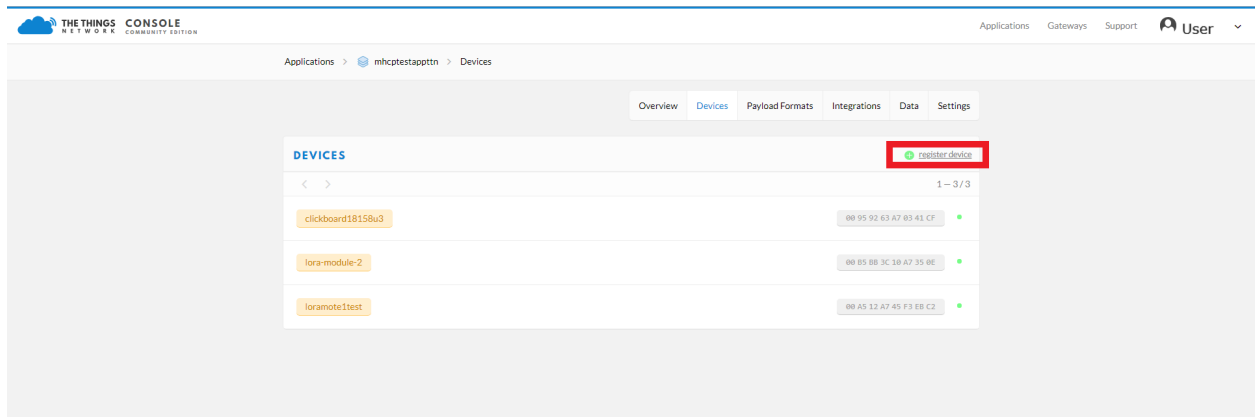


Figure 3-7. Devices Tab



In the following screen, add a name to the device. Press the button to generate a device EUI and then click on **Register**.

Figure 3-8. New Device Tab

The screenshot shows the 'REGISTER DEVICE' form in the The Things Console. The form has tabs for Overview, Devices, Payload Formats, Integrations, Data, and Settings. The 'REGISTER DEVICE' tab is active. It contains the following fields:

- Device ID:** A text input field with a description: 'This is the unique identifier for the device in this app. The device ID will be immutable.'
- Device EUI:** A text input field with a description: 'The device EUI is the unique identifier for this device on the network. You can change the EUI later.' A red box highlights the '0x' prefix.
- App Key:** A text input field with a description: 'The App Key will be used to secure the communication between you device and the network.' A note below the field says 'this field will be generated'.
- App EUI:** A text input field with a value of '78 E3 D5 7E D8 01 33 95'.

At the bottom right, there are 'Cancel' and 'Register' buttons. The 'Register' button is highlighted with a red box.

Now click on the device in the **Device** tab and its overview will open.

Figure 3-9. Device Overview

The screenshot shows the 'DEVICE OVERVIEW' for device 'lora-module-2'. The 'Overview' tab is active. It displays the following information:

- Application ID:** mhctestapp1tn
- Device ID:** lora-module-2
- Activation Method:** OTAA (highlighted with a red box)
- Keys required for OTAA:**
 - Device EUI:** 00 85 88 3C 18 A7 35 0E
 - Application EUI:** 78 E3 D5 7E D8 01 33 95
 - App Key:** (highlighted with a red box)
- Keys required for ABP:**
 - Device Address:** 25 85 24 89
 - Network Session Key:** (highlighted with a red box)
 - App Session Key:** (highlighted with a red box)
- Status:** 50 seconds ago
- Frames up:** 1 (with a link to 'reset frame counters')
- Frames down:** 2

The most important information can be found in this tab. The Activation Method field shows the currently configured method. The keys for both methods can also be found here. The Status field shows the time the device last communicated with the server.

In order to change the activation method, go to the **Settings** tab in the top right corner.

Figure 3-10. Device Settings

The screenshot shows the 'Device Settings' page in the The Things Console. The left sidebar has 'DEVICE SETTINGS' with 'General' and 'Location' tabs. The main area has 'SETTINGS' with the following fields:

- Description:** A human-readable description of the device.
- Device EUI:** The serial number of your radio module, similar to a MAC address. Value: 00 B5 BB 3C 10 A7 35 0E.
- Application EUI:** Value: 70 B3D5 7E D0 01 33 95.
- Activation Method:** Two buttons, 'OTAA' and 'ABP'. The 'OTAA' button is highlighted with a red box.
- App Key:** The key your device will use to set up sessions with the network. Value: B8 A4 73 34 FF 70 03 35 DD 5C B6 47 DB FB D5 52.
- Frame Counter Width:** Two buttons, '16 bit' and '32 bit'.
- Frame Counter Checks:** A checkbox that is checked.

The activation method can be set from here as well as adding a description, writing a new device EUI or looking at the application key.

After the device has been registered and set, everything is configured from the TTN side and will work with a configured device. The transmitted data can be checked in the **Data** tab in the Device Overview.

3.4 Peripheral Configuration

First, all the peripherals and the main clock must be configured in Atmel START. To do this, a new Atmel START project must be created by choosing the AVR-IoT WG Board as a starting point:

1. Open Atmel Studio.
2. Click on *File → New → Atmel Start Project*.
3. In the search bar, type AVR-IoT and select the ATmega4808 AVR-IoT WG Board.

Figure 3-11. Project Selection

The screenshot shows the 'CREATE NEW PROJECT' dialog in Atmel Studio. The 'RESULTS' section displays a table of search results for '4808'.

Name	Architecture	Package	Pins	Flash	SRAM
ATmega4808-XFR	AVR	SSOP28	28	48 KB	6 KB
ATmega4808-MFR	AVR	VQFN32	32	48 KB	6 KB
ATmega4808-AFR	AVR	TQFP32	32	48 KB	6 KB
ATmega4808 AVR IoT WG					

For the purpose of this application, a system clock of 5 MHz will be used. This ensures a power-saving profile is used, while delivering the appropriate performance for LoRaWAN communication. The ADC is used to read data from the light sensor. The I²C peripheral is used for reading data from the temperature sensor.

One USART instance (ASYNC) communicates with the LoRa module at 57600 baud rate, while the other USART instance is used for communication through the COM port to the PC or other devices connected to the USB at 9600 baud rate. Timer/Counter B is used for waking up the microcontroller from Sleep.

3.4.1 Main Clock Control. CLKCTRL

In the CLKCTRL, the configuration for the main clock must be made.

Hardware settings

CLKSEL (Main clock source): 20 MHz internal oscillator

PEN (Prescaler enable): On

PDIV (Prescaler division): 4

32 kHz internal ULP oscillator configuration enable: On

Run in Standby: On

Figure 3-12. Clock Configuration

MY SOFTWARE COMPONENTS ?

GENERAL	COMPONENT SETTINGS	COMPONENT SIGNALS
i User guide	Driver: Drivers:CLKCTRL:Init	
DRIVERS:CLKCTRL:INIT (CLKCTRL) CONFIGURATION ON CLKCTRL ↻		
MAIN CLOCK CONFIGURATION Enable: ☒ OSCULP32K DIV 32 Enable: ☒		
CLKSEL: Main Clock Source: ? 20MHz Internal Oscillator (OSC20M) ▼	CLKSEL: Main Clock Source: 32KHz Internal Ultra Low Power Oscillator ▼	
PEN: Prescaler enable: ? ☒	16MHZ OSCILLATOR CONFIGURATION Enable: ☒	
PDIV: Prescaler division: ? 4 ▼	RUNSTDBY: Run standby: ? ☐	
LOCKEN: lock enable: ? ☐	32.768KHZ CRYSTAL OSCILLATOR CONFIGURATION Enable: ☐	
CLKOUT: System clock out: ? ☐	TCD0 CLOCK CONFIGURATION Enable: ☒	
20MHZ OSCILLATOR CONFIGURATION Enable: ☒	CLKSEL: clock select: ? 20MHz Internal Oscillator (OSC20M) ▼	
RUNSTDBY: Run standby: ? ☐	SYNCPRES: Synchronization prescaler: ? 1 ▼	
32KHZ INTERNAL ULP OSCILLATOR CONFIGURATION Enable: ☒		
RUNSTDBY: Run standby: ? ☒		
EXTERNAL CLOCK CONFIGURATION Enable: ☒		
External Oscillator Frequency: 32768 ▼ dec v		
SYSTEM CLOCKS Enable: ☒		
CPU clock source: Main Clock (CLK_MAIN) ▼		
RAM clock source: Main Clock (CLK_MAIN) ▼		
NVM clock source: Main Clock (CLK_MAIN) ▼		

3.4.2 ADC

Configure the ADC to read the output of the light sensor.

Hardware settings

Driver: Drivers:ADC:Basic

AIN/5: PD5 On

RESSEL: ADC resolution: 10-bit mode

MUXPOS: Analog channel selection bits: ADC input pin 5

PRESC: Clock prescaler: CLK_PER divided by 16

REFSEL: Reference selection: V_{DD}

Figure 3-13. ADC Configuration

ADC_0

MCCode basic driver: Peripheral initialization + API to support simple ADC use-case

GENERAL

User guide

Rename component

Remove component

COMPONENT SETTINGS

Driver: Drivers:ADC:Basic

CLOCKS

ADC: Main Clock (CLK_MAIN) (5 MHz)

COMPONENT SIGNALS

AIN/0: ☐ PD0 (PORT/P/10)

AIN/1: ☐ PD1 (PORT/P/11)

AIN/2: ☐ PD2 (PORT/P/12)

AIN/3: ☐ PD3 (PORT/P/13)

AIN/4: ☐ PD4

AIN/5: ☒ PD5

AIN/6: ☐ PD6 (LORA2_CLICK_0/CTS PIN)

AIN/7: ☐ PD7 (PORT/P/17)

AIN/12: ☐ PF2 (PORT/P/22)

AIN/13: ☐ PF3 (PORT/P/23)

AIN/14: ☐ PF4 (PORT/P/24)

AIN/15: ☐ PF5 (PORT/P/25)

DRIVERS:ADC:ADC0 (ADC BASIC) CONFIGURATION ON ADC0

ADVANCED CONFIGURATION

DBGRUN: Debug run: ☐

RUNSTBY: Run standby mode: ☐

RESSEL: ADC Resolution: 10-bit mode

INPUT CONFIGURATION

MUXPOS: Analog Channel Selection Bits: ADC input pin 5

INTERRUPT CONFIGURATION

RESRDY: Result Ready Interrupt Enable: ☐

BASIC CONFIGURATION

ENABLE: ADC Enable: ☒

PRESC: Clock Prescaler: CLK_PER divided by 16

REFSEL: Reference Selection: VDD

SAMPLING CONFIGURATION

Enable: ☐

3.4.3 USART to PC

This is the USART instance that provides the interface with the computer or the device connected at the USB port. The USART instance will be named Terminal.

Hardware settings

Rename component: Terminal

Driver: Drivers:USART:Basic

Mode: Async IRQ mode

Instance: USART2

RXD: PF1

TXD: PF0

Printf support: On

RX buffer size: 8

TX buffer size: 8

RXEN: Receiver enable: On

TXEN: Transmitter enable: On

PMODE: Parity mode: No parity

SBMODE: Stop Bit mode: 1 Stop bit

CHSIZE: Character size: Character size: 8 bits

Baud rate: 9600

Figure 3-14. USART Terminal Configuration

3.4.4 Timer TCB

This is the timer that will keep counting during Sleep in order to wake up the microcontroller and the RN2483 module. It will be configured to provide a 1-minute period between transmissions.

The timer is in Periodic Interrupt mode, which sends an interrupt request every time its value becomes bigger than the value in the CCMP field. The timer uses the system clock in order to work while the microcontroller is in Sleep.

The timer is only enabled during Sleep and will count at the frequency of the low frequency oscillator divided by 2, which is 16.384 kHz. It is configured to send an interrupt request every 60 seconds. The value in CCMP can be changed in order to have different times between transmissions.

Hardware settings

Driver: Drivers:TCB:Init

Instance: TCB0

ENABLE: Enable: On

RUNSTBY: Run Standby: On

CLKSEL: Clock select: CLK_PER/2 (From prescaler)

CNTMODE: Timer mode: Periodic interrupt

CCMP: Compare or capture: 0x3c00 – this is the compare value required in order for the timer to overflow and generate an interrupt every 60 seconds.

Include ISR harness in driver_isr.c: On

CAPT: Capture or time-out: On

Figure 3-15. TCB Configuration

TIMER_0
TCB Init driver

GENERAL

- User guide
- Rename component
- Remove component

COMPONENT SETTINGS

Driver: Drivers:TCB:Init

Instance: TCB0

CLOCKS

TCB: Main Clock (CLK_MAIN) (5 MHz)

COMPONENT SIGNALS

WO/0: ----

DRIVERS:TCB:INIT (INIT) CONFIGURATION ON TCB0

CONFIGURATION

ENABLE: Enable: ☒

RUNSTBY: Run Standby: ☒

SYNCUPD: Synchronize Update: ☐

ASYNC: Asynchronous Enable: ☐

DBG RUN: Debug Run: ☐

CLKSEL: Clock Select: CLK_PER/2 (From Prescaler)

CNTMODE: Timer Mode: Periodic Interrupt

INTERRUPT CONFIGURATION

Include ISR harness in driver_isr.c: ☒

CAPT: Capture or Timeout: ☒

EVENT CONFIGURATION

EDGE: Event Edge: ☐

CAPTEI: Event Input Enable: ☐

FILTER: Input Capture Noise Cancellation Filter: ☐

COUNT AND COMPARE/CAPTURE

CCMP: Compare or Capture: 0x3c00 hex v

CNT: Count: 0x0 hex v

PIN OUTPUT CONFIGURATION

CCMPEN: Pin Output Enable: ☐

CCMPINIT: Pin Initial State: ☐

3.4.5 I²C

This peripheral is used for communicating with the temperature sensor on the board.

Hardware settings

Driver: Drivers:I²C:Master

Mode: Interrupt

SCL: PA3

SDA: PA2

SCL Frequency: 100000

Trise: 0

ENABLE: Enable TWI master: On

Figure 3-16. I²C Configuration

3.4.6 LORA2 CLICK

A LORA2 CLICK driver must be configured. This middleware has a dependency for a USART instance which will be used to communicate with the RN2483 module.

Hardware settings

USART Async: USART_ASYNC

CTS PIN: PD6

HwReset PIN: PA0

RTS PIN: PC3

Figure 3-17. LoRa2 Click Configuration

LORA2_CLICK_0
Driver for LORA2 clickboard

GENERAL

- User guide
- Rename component
- Remove component

COMPONENT SETTINGS

Variant: Click:Wireless:LoRa2

DEPENDENCIES

USART Async: USART_ASYNC

If you can't find your component in the list make sure you have selected the correct **variant** for this component, and also correct **driver** and/or **mode** for the component you wish to select.

CTS PIN: PD6

HwReset PIN: PA0

RTS PIN: PC3

LORA2 CLICKBOARD CONFIGURATION

EXAMPLE CONFIGURATION

Generate Example: ☒

3.4.7 USART Async to RN2483 Module

This is the USART instance that will be used to send commands to the LoRa module and receive response messages from it. The USART instance will be named LoRa.

Hardware settings

Rename component: LoRa

Driver: Drivers:USART:Basic

Mode: Async IRQ mode

Instance: USART1

RXD: PC1

TXD: PC0

Printf support: Off

RX buffer size: 8

TX buffer size: 8

RXEN: Receiver enable: On

TXEN: Transmitter enable: On

PMODE: Parity mode: No parity

SBMODE: Stop Bit mode: 1 Stop bit

CHSIZE: Character size: Character size: 8 bits

Baud rate: 57600

Figure 3-18. USART LoRa Configuration

USART_ASYNC
UART driver in IRQ mode

GENERAL

- User guide
- Rename component
- Remove component

COMPONENT SETTINGS

Driver: Drivers:USART:Basic

Mode: Async IRQ Mode

Instance: USART1

COMPONENT SIGNALS

RXD: PC1

TXD: PC0

CLOCKS

USART: Main Clock (CLK_MAIN) (5 MHz)

DRIVERS:USART: BASIC (ASYNC IRQ MODE) CONFIGURATION ON USART1

DRIVER CONFIGURATION

Printf support: ☐

RX Buffer Size: 8

TX Buffer Size: 8

BASIC CONFIGURATION

RXEN: Receiver enable: ☒

TXEN: Transmitter Enable: ☒

PMODE: Parity Mode: No Parity

SBMODE: Stop Bit Mode: 1 stop bit

CHSIZE: Character Size: Character size: 8 bit

Baud Rate: 57600 dec v

ADVANCED CONFIGURATION

LBME: Loop-back Mode Enable: ☐

DBG RUN: Debug Run: ☐

3.4.8 CPUINT

Global interrupts need to be enabled in order to wake up from Sleep.

Hardware settings

CPU_SREG: Global Interrupt Enable: On

3.4.9 PINMUX

Several pins have to be renamed and configured as digital output in order to have LED functionality. The pins need to have the exact user label in order for the program to compile.

- PD0 named ERR_LED:
 - User label: LED_RED
 - Initial level: High
 - Pin Mode: Digital output
- PD1 named DATA_LED:
 - User label: LED_YELLOW
 - Initial level: High
 - Pin Mode: Digital output
- PD2 named CONN_LED:
 - User label: LED_GREEN
 - Initial level: High
 - Pin Mode: Digital output
- PD3 named WIFI_LED:
 - User label: LED_BLUE
 - Initial level: High

- Pin Mode: Digital output
- 5. PF3 named WINC_EN:
 - User label: PF3
 - Initial level: High
 - Pin Mode: Digital output

3.5 Getting the Keys from TTN

In order to use the TTN server register, the LoRa device has to be registered to an application on the server.

After the previous steps have been completed, TTN will provide the required keys for Over-the-Air Authentication (OTAA) or Activation by Personalization (ABP).

In the `lora_keys.h` file, there are two lines for defining the authentication type. For OTAA authentication, the define OTAA line must be uncommented and the ABP line commented, while for ABP authentication the opposite must be true.

```
#define OTAA      1
//#define ABP    1
```

3.5.1 Over-the-Air Authentication

The device added to the TTN application must be configured as OTAA. TTN will provide the following information that needs to be passed to the RN2483 module: Device EUI, Application EUI and App Key. They can be found in the overview for the device.

Figure 3-19. OTAA TTN Credentials

The screenshot shows three credential fields from the TTN interface:

- Device EUI:** A text box containing the hexadecimal value `00 B5 BB 3C 10 A7 35 0E`.
- Application EUI:** A text box containing the hexadecimal value `70 B3 D5 7E D0 01 33 95`.
- App Key:** A text box containing a series of dots, indicating a masked key value.

Copy this information to the `lora_keys.h` file at the declarations of the commands for the RN2483 module, like in the following example.

```
#ifndef OTAA
const char my_deveui[] = "00B5BB3C10A7350E";
const char my_appui[] = "70B3D57ED0013395";
const char my_appkey[] = "B8A4.....";
```

The `'rn2903_SendString()'` function is compatible with RN2483 as the USART communication is the same for both modules. The middleware for the LoRa click was created for the North American version (RN2903) and the function names use that version, but there is no difference in the communication interface between the two, so the functions work perfectly for both of them.

OTAA is the preferred and the most secure method to connect to TTN. The device will perform a join procedure where a dynamic device address (DevAddr) is assigned to it and security keys will be exchanged.

3.5.2 Activation by Personalization (ABP)

The device added to TTN must be configured as ABP. TTN will provide all the required information: Device Address, Network Session Key, App Session Key. This information can be found in the device overview.

Figure 3-20. ABP TTN Credentials

The screenshot shows three credential fields for an ABP device:

- Device Address:** 26 05 28 0C
- Network Session Key:** Masked with dots (.....6D)
- App Session Key:** Masked with dots (.....93)

Copy this information to the `lora_keys.h` file in the declaration of the commands for the RN2483 module, as in the following example.

```
#elif ABP
const char my_devaddr[] = "26052B8D";
const char my_nwkskey[] = "4AE4A.....6D";
const char my_appskey[] = "D545F.....93";
```

This activation method skips the join procedure but has some security concerns.

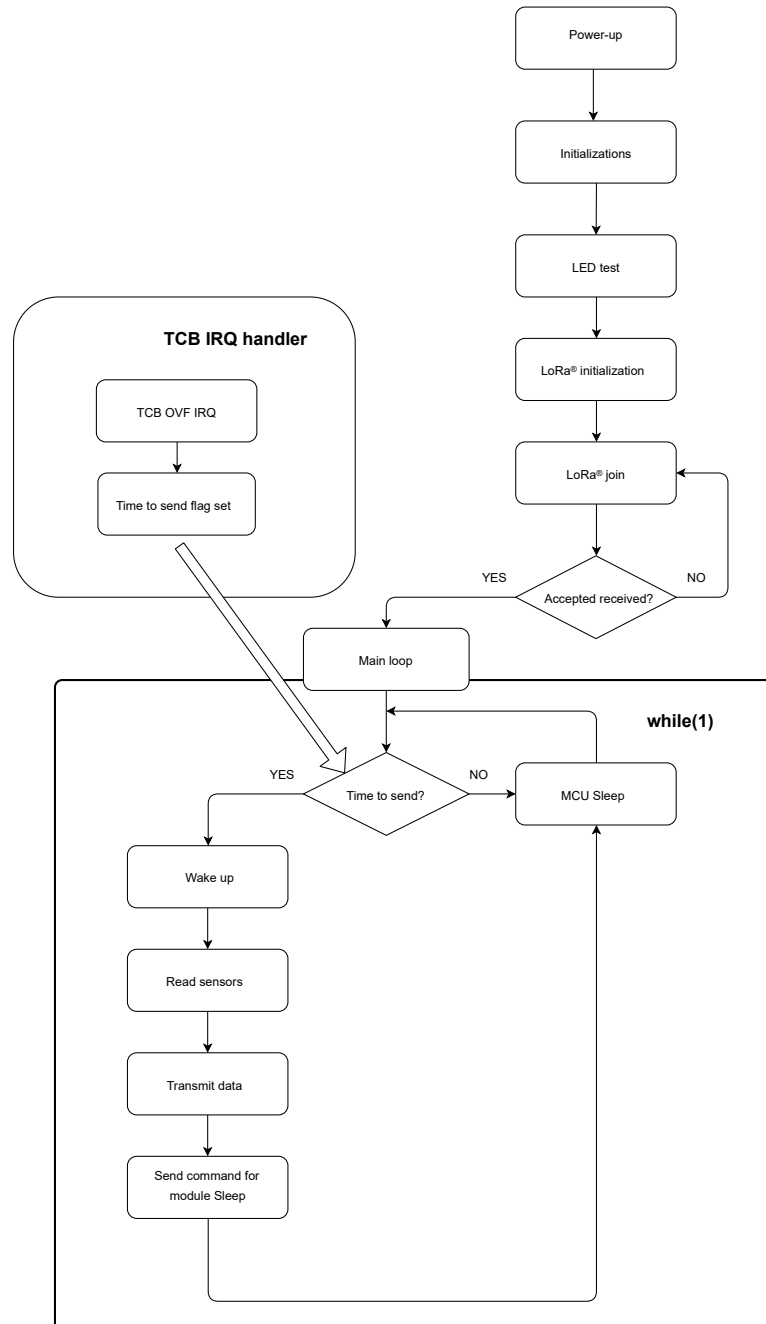
3.6 Data Rate and Spread Factor Modulation (SFM)

Data rate correlates to the amount of time the device spends transmitting data. A lower data rate means a higher SFM and the possibility of transmitting at larger distances, but also comes with the disadvantage of requiring a longer time between messages to fit in the duty cycle limit. For transmitting every minute, the highest data rate of five is recommended.

4. Functional Description of the Application

The application starts by trying to do a join operation until the accepted message is received. In the case of a 'denied' response, the join operation is repeated.

Figure 4-1. Application Flowchart



After the device joins the network, it will start sending sensor data every minute to the server. While the device is sending data, the yellow LED will stay on. When the device is not in use, a Sleep command will be sent to it, after which the microcontroller will also enter Sleep.

In the Initializations phase, the peripherals will be configured according to the user defined settings. The following modules are configured during this step:

1. The main 20 MHz clock with a 4 prescaler setting (5 MHz) and the 32 kHz internal oscillator with a 64 prescaler setting which comes out to about 500 Hz.
2. All the pins for the LEDs, switches and other functions (CTS, RTS, etc.)
3. The Timer B peripheral is configured for a period of 60 seconds during Sleep.
4. Sleep control initialization.
5. Communication peripherals: USART_0, USART ASYNC, I²C.
6. The ADC.
7. Enables the global and peripheral interrupts.

After these steps are completed, an LED test is performed by lighting and closing each LED for a short time.

The LoRa Initialization phase starts by resetting the RN2483 module, setting the required keys, registering the ISRCallback function for the timer and trying to join the network until an accepted message is received. During all these, the blue LED will stay turned on to signify this is happening.

After the accepted message has been received, the microcontroller will enter an infinite loop where it stays in Sleep for one minute and then sends the sensor data to the server.

As a part of the Sleep mode operation, the system clock source will be switched to the Internal Ultra-Low Power 32.768 kHz Oscillator (OSCULP32K). The same clock source will also be used for Timer B.

When it is time to send, the timer is disabled to prevent interrupts, the system clock is switched back to the 20 MHz Internal Oscillator (OSC20M). The firmware will send the wake-up command and will wait for the 'ok' response from the RN2483 module. Next, the sensors are read and the command for transmitting them is sent to the RN2483 module. After the transmission finishes, the LoRa module is put back to sleep for a minute, the clock is switched to the ultra-low power oscillator and Timer B is re-enabled. While the transmission is happening, the yellow LED will stay on to notify the user that a transmission is ongoing.

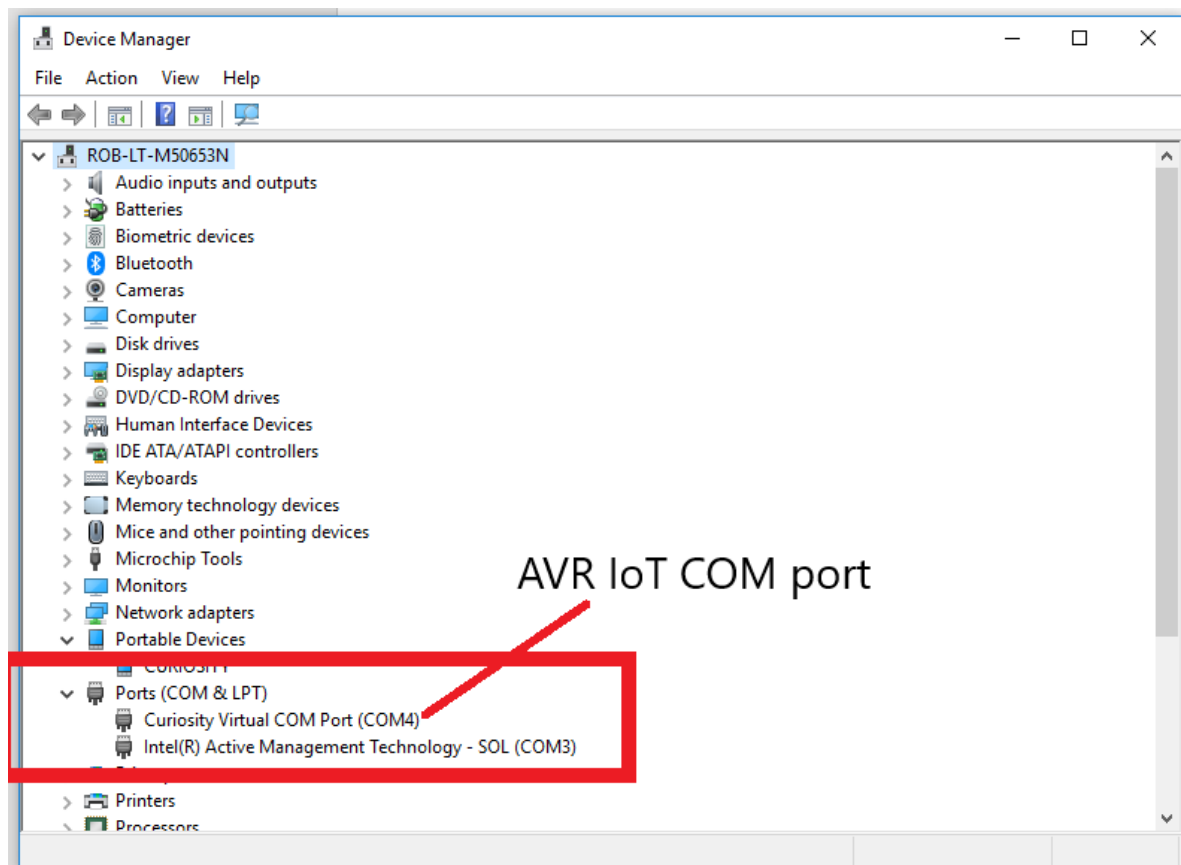
5. Serial Communication with the RN2483 Device

In order to receive messages through the USART interface from the RN2483 device, a serial communication software is required. Any program will work, but this application note will explain the use of Tera Term, a free software that can be downloaded from [here](#).

Follow these steps to connect the terminal to the board:

1. Make sure the board is connected to the PC via the Micro-USB cable. The COM port which the board has been connected to is needed at this point. To find out the COM port, access the Device Manager window and search for the board. To open Device Manager, first open Windows® Start, type Device Manager and click on it.
2. Scroll down through the list of devices and expand 'Ports (COM & LPT)'.

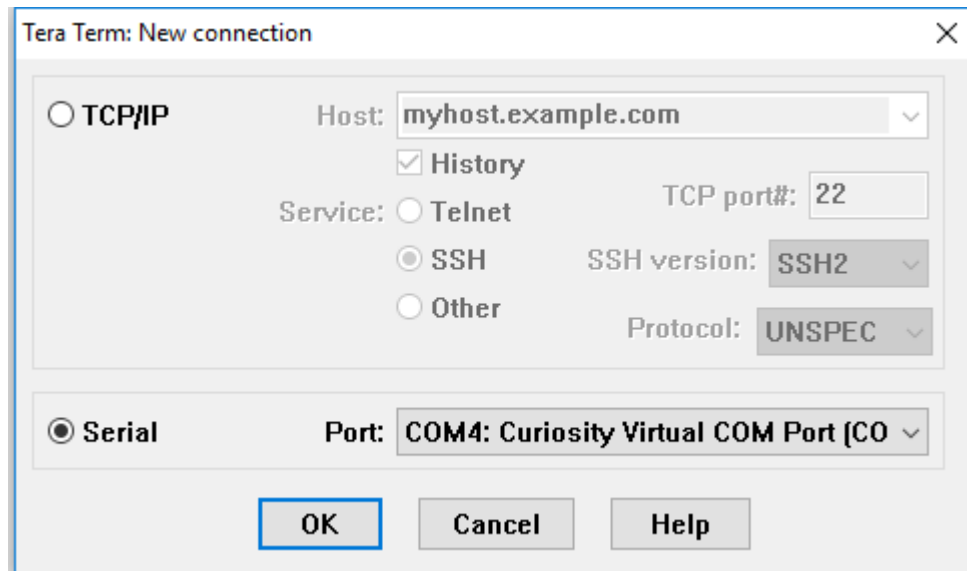
Figure 5-1. Device Manager



Note: The port to which the AVR-IoT WG Board is connected contains 'Curiosity' in its name (e.g., COM4 in the example above). Keep the Device Manager window open.

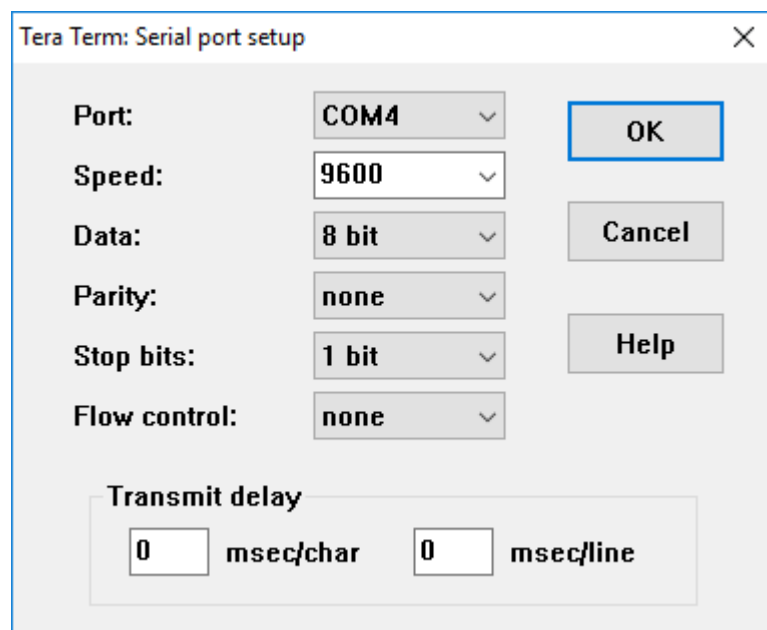
3. Open Tera Term (search for its shortcut on the PC desktop or in Windows Start panel).
4. Click **Serial** and select the correct COM port of the AVR-IoT WG Board in the Port drop-down menu; click **OK**.

Figure 5-2. Tera Term New Connection



- Click on the **Setup** tab and then on **Serial Port**. Set the correct serial communication parameters as described in the picture below. Confirm settings by clicking **OK**.

Figure 5-3. Serial Port Configuration



After these steps have been completed, the device can send messages describing if it executed the commands correctly. The next section will describe what these messages mean. The communication is only one way. The user is not able to send commands to the RN2483 device.

6. Responses from RN2483 Device through USART

There are several responses the device will send during normal operation. This is a list detailing all of them. For more information about the commands that can be received, check [RN2483 LoRa Technology Module Command Reference User's Guide](#).

RN2483 X.Y.Z MMM DD YYYY HH:MM:SS

X.Y.Z is the firmware version and the rest are, in order: month, day, year, hour, minutes, seconds.

This message is displayed after a system Reset. It provides information about the device.

OK

This message is received after every command sent to the device, to confirm that the command was interpreted correctly.

invalid_param

This message is received instead of 'ok' when a command was sent incorrectly. If the application code from this application note is used, this message will never appear.

Accepted/Denied

This message is received after attempting to authenticate on the TTN server after a join operation. This only appears after powering on the device. 'Accepted' is the response that confirms a successful join to the network.

'Denied' can mean several things, including no functional gateway in the area or that the authentication keys are wrong. The 'denied' message can appear a few times even if the device is set up properly for reasons beyond the user's control. If this happens, the device attempts the join operation again until 'accepted' is received. Generally, if the 'denied' message is received more than three times in a row, the device will need to be restarted. If the problem persists, the gateway and the keys need to be checked. Another problem might be the choice of handler in the TTN menu.

mac_tx_ok

This message represents a successful transmission to the server. This is the most common response for information transmission.

mac_err

This error does not affect the normal operation of the device and is thrown in different situations that can vary from the message being 'lost', to the gateway not being able to send an acknowledge response. The gateway has to obey the 1% duty cycle as well, so that is the most common reason for receiving this error, but it will not affect the transmission of data.

no_free_ch

This error means that the hardware duty cycle limit has been reached. The device will try to send again until a transmission is executed successfully. This error could potentially appear more frequently for smaller data rates as the messages spend more time on air. This can be solved by increasing the time between transmissions or moving to a higher data rate.

This information is useful for troubleshooting and the responses from the device provide an easy way to check if everything is in order.

Figure 6-1. Communication Example Between PC and RN2483

```
accepted
ok
mac_tx_ok
ok
ok
mac_tx_ok
ok
ok
ok
mac_tx_ok
ok
ok
ok
mac_tx_ok
ok
ok
mac_err
ok
ok
mac tx ok
```

7. Node.js Installation and Application

In order to better visualize the data from the sensors, a node.js application was created that displays them in a graphic format. In order to use the application, node.js and a few dependencies have to be installed on the user's machine.

This guide will explain how to do this:

1. Download the installer for node.js from this [link](#) and run it to completion.
2. Test if it has installed correctly. Open one of Command Prompt, Windows PowerShell or the preferred command line shell. Type `node -v` and a version will appear if everything has been installed correctly.
3. Download and install Git from [here](#).
4. Navigate to the folder where the Data Visualizer is saved and hold shift and press on right click. Select the option to open PowerShell in that directory. Alternatively, the `cd` command can be used to navigate to the correct folder.
5. Run the `npm i` command to install the required node modules.
6. Optional: Run the `npm audit fix` command if there are any vulnerabilities.

These are all the steps necessary to configure the visualizer.

Figure 7-1. PowerShell Commands

```

Administrator: Windows PowerShell
PS C:\My stuff\The Things Network> npm i
> grpc@1.16.0 install C:\My stuff\The Things Network\node_modules\grpc
> node-pre-gyp install --fallback-to-build --library=static_library

node-pre-gyp WARN
[grpc] Success: "C:\My stuff\The Things Network\node_modules\grpc\src\node\extension_binary\node-v64-win32-x64-unknown\g
rpc_node.node" is installed via remote
npm WARN thttnmyapp@1.0.0 No description
npm WARN thttnmyapp@1.0.0 No repository field.

added 321 packages from 198 contributors and audited 922 packages in 16.997s
found 2 moderate severity vulnerabilities
  run 'npm audit fix' to fix them, or 'npm audit' for details
PS C:\My stuff\The Things Network> npm audit fix
> grpc@1.19.0 install C:\My stuff\The Things Network\node_modules\grpc
> node-pre-gyp install --fallback-to-build --library=static_library

node-pre-gyp WARN
[grpc] Success: "C:\My stuff\The Things Network\node_modules\grpc\src\node\extension_binary\node-v64-win32-x64-unknown\g
rpc_node.node" is installed via remote
npm WARN thttnmyapp@1.0.0 No description
npm WARN thttnmyapp@1.0.0 No repository field.

4 ttn@2.3.2
added 51 packages from 25 contributors, removed 8 packages and updated 27 packages in 128.48s
fixed 2 of 2 vulnerabilities in 770 scanned packages
PS C:\My stuff\The Things Network> node index.js
listening on *:3000
a user connected
user disconnected
a user connected
{ app_id: 'mncptestappttn',
  dev_id: 'lora-module-2',
  hardware_serial: '00B5BB3C10A7350E',
  port: 1,
  counter: 12,
  confirmed: true,
  is_retry: true,
  payload_raw: <Buffer 0d 1b>,
  metadata:
    { time: '2019-04-11T10:42:48.436452969z',
      frequency: 868.3,
      modulation: 'LORA',
      data_rate: 'SF10BW125',
      airtime: 329728000,
      coding_rate: '4/5',
      gateways: [ [Object] ] } }

```

The connection keys must be taken from the TTN console. Navigate to the **Applications** tab and scroll down to the **Access Key** tab. That value needs to be copied to the `accessKey` variable in the `index.js`

file. The file can be changed with any text editor. The appID variable must also be changed to the user's application ID which can be found on the same tab.

Figure 7-2. TTN Access Key

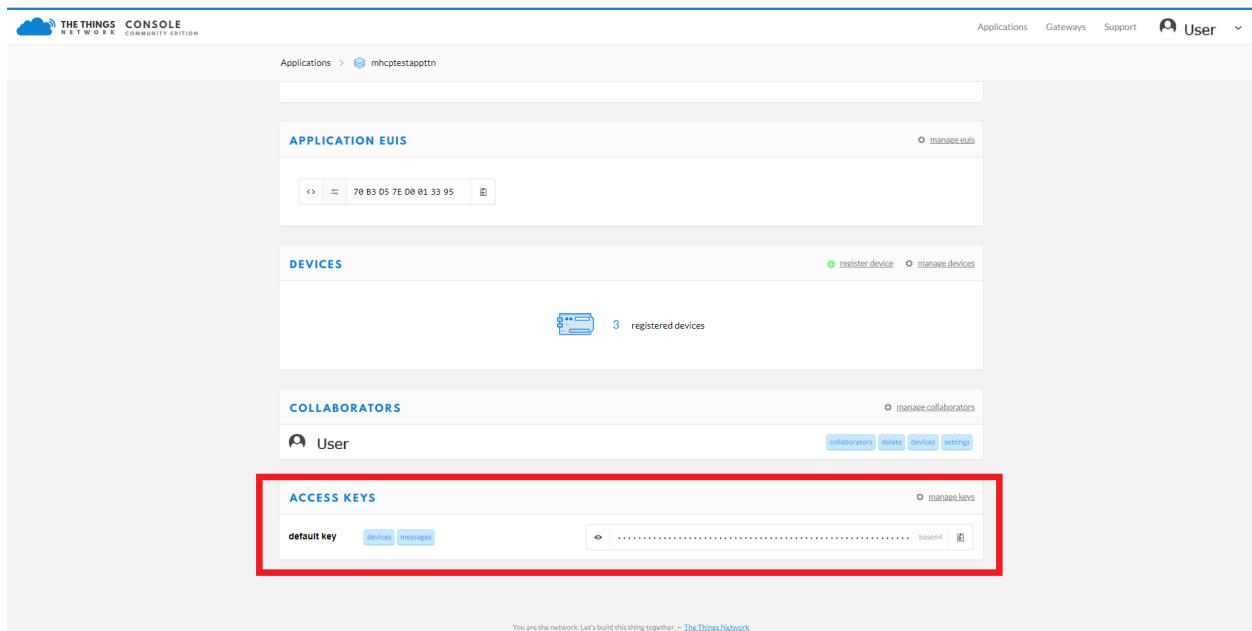
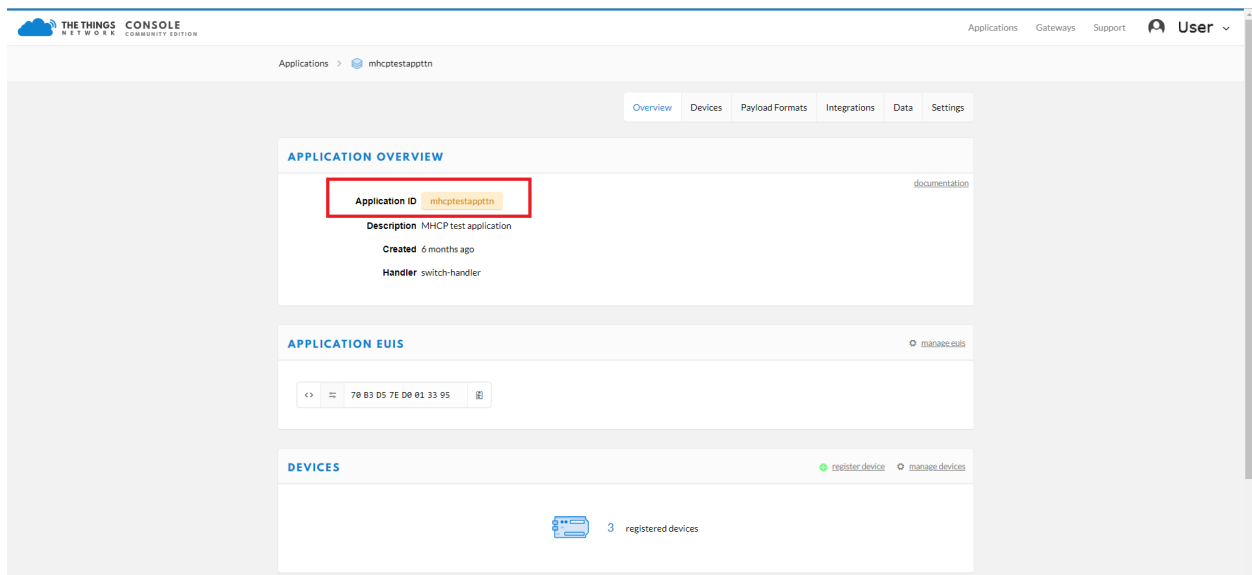


Figure 7-3. TTN Application ID



Code example:

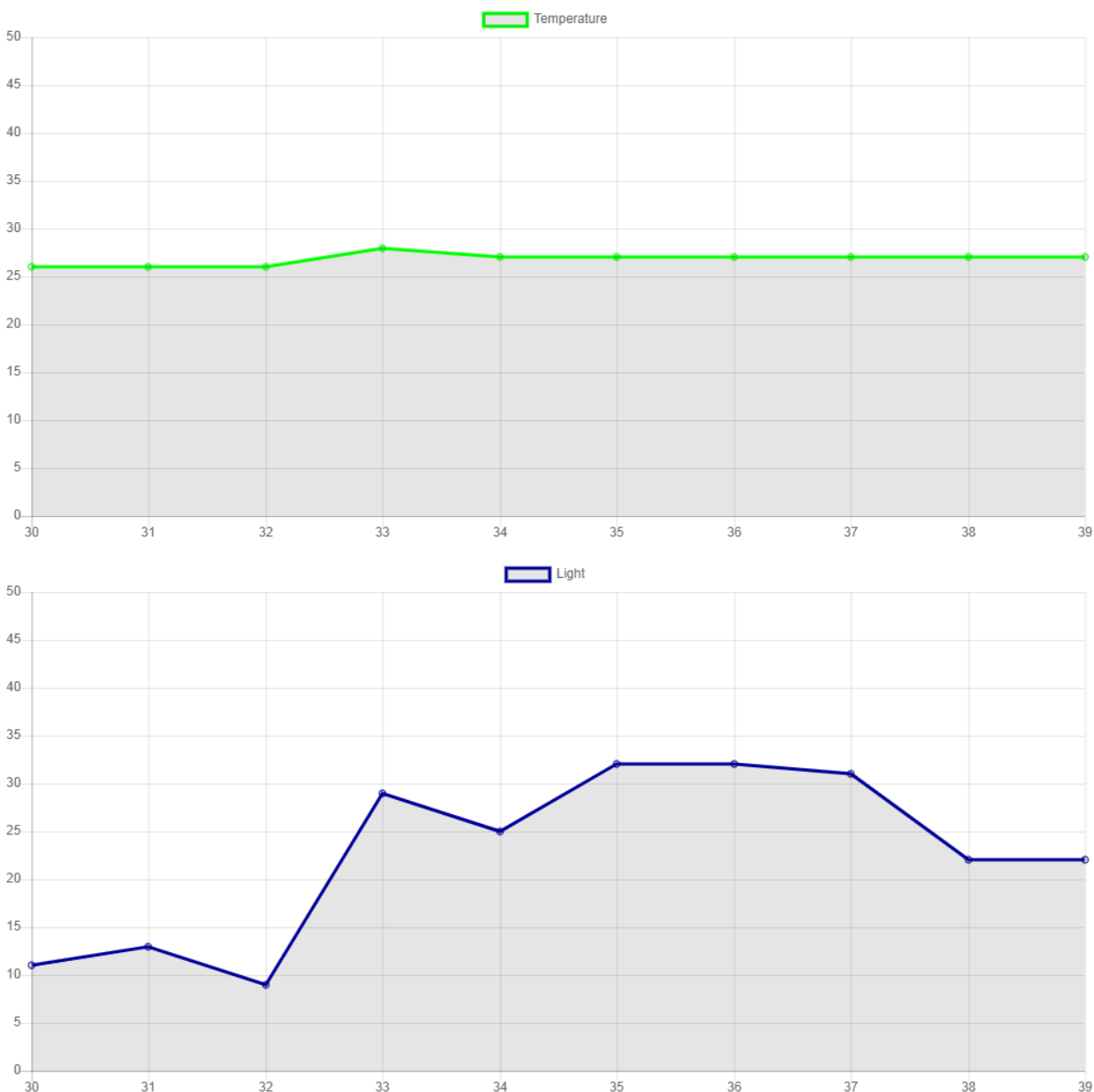
```
var appID = "mhcptestapptn"
var accessKey = "ttn-account-v2.8JjxCed1rGvYUtCT5a41IO41fv9DCbSLsdIApbn5q54"
```

In order to run the application, open a command line shell in its folder and run the command `node index.js`. A local server will be started that will take the information from TTN and display it in the browser at the address 'localhost:3000'. The shell must be kept open for the application to continue to function.

To reach the page, open the preferred Internet browser and type in the address bar 'localhost:3000' and press Enter.

The webpage contains two graphs: one for temperature and one for light level. Both graphs display 10 data points at a time, discarding the last one when a new one comes in. The graph adds a new data point every minute with the value added being the last received one from the server. The webpage has to be open in order for data points to be added to the graph.

Figure 7-4. Data Graphs



8. Conclusion

Adding LoRa connectivity to the AVR-IoT WG Board through a Click board is simple and can be done by following the steps detailed in this application note. This allows the user to communicate at large distances with low power consumption.

9. References

1. RN2483 LoRa Technology Module Command Reference User's Guide: <https://ww1.microchip.com/downloads/en/DeviceDoc/RN2483-LoRa-Technology-Module-Command-Reference-User-Guide-40001784G.pdf>
2. The Things Network:
 - <https://www.thethingsnetwork.org/docs/gateways/>
 - <https://www.thethingsnetwork.org/docs/gateways/gateway/>
 - <https://account.thethingsnetwork.org/register>
3. MikroElektronika LoRa click: <https://www.mikroe.com/lora-rf-click>
4. A Technical Overview of LoRa and LoRaWAN: https://www.tuv.com/media/corporate/products_1/electronic_components_and_lasers/TUeV_Rheinland_Overview_LoRa_and_LoRaWANtmp.pdf
5. Node.js download: <https://nodejs.org/en/>
6. Tera Term download: <https://tera-term.en.lo4d.com/>

The Microchip Website

Microchip provides online support via our website at <http://www.microchip.com/>. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to <http://www.microchip.com/pcn> and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the web site at: <http://www.microchip.com/support>

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TempTrackr, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, FlashTec, Hyper Speed Control, HyperLight Load, IntelliMOS, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, Vite, WinPath, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BlueSky, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestlC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2019, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-4578-4

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit <http://www.microchip.com/quality>.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com	Australia - Sydney Tel: 61-2-9868-6733 China - Beijing Tel: 86-10-8569-7000 China - Chengdu Tel: 86-28-8665-5511 China - Chongqing Tel: 86-23-8980-9588 China - Dongguan Tel: 86-769-8702-9880 China - Guangzhou Tel: 86-20-8755-8029 China - Hangzhou Tel: 86-571-8792-8115 China - Hong Kong SAR Tel: 852-2943-5100 China - Nanjing Tel: 86-25-8473-2460 China - Qingdao Tel: 86-532-8502-7355 China - Shanghai Tel: 86-21-3326-8000 China - Shenyang Tel: 86-24-2334-2829 China - Shenzhen Tel: 86-755-8864-2200 China - Suzhou Tel: 86-186-6233-1526 China - Wuhan Tel: 86-27-5980-5300 China - Xian Tel: 86-29-8833-7252 China - Xiamen Tel: 86-592-2388138 China - Zhuhai Tel: 86-756-3210040	India - Bangalore Tel: 91-80-3090-4444 India - New Delhi Tel: 91-11-4160-8631 India - Pune Tel: 91-20-4121-0141 Japan - Osaka Tel: 81-6-6152-7160 Japan - Tokyo Tel: 81-3-6880-3770 Korea - Daegu Tel: 82-53-744-4301 Korea - Seoul Tel: 82-2-554-7200 Malaysia - Kuala Lumpur Tel: 60-3-7651-7906 Malaysia - Penang Tel: 60-4-227-8870 Philippines - Manila Tel: 63-2-634-9065 Singapore Tel: 65-6334-8870 Taiwan - Hsin Chu Tel: 886-3-577-8366 Taiwan - Kaohsiung Tel: 886-7-213-7830 Taiwan - Taipei Tel: 886-2-2508-8600 Thailand - Bangkok Tel: 66-2-694-1351 Vietnam - Ho Chi Minh Tel: 84-28-5448-2100	Austria - Wels Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 Denmark - Copenhagen Tel: 45-4450-2828 Fax: 45-4485-2829 Finland - Espoo Tel: 358-9-4520-820 France - Paris Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 Germany - Garching Tel: 49-8931-9700 Germany - Haan Tel: 49-2129-3766400 Germany - Heilbronn Tel: 49-7131-72400 Germany - Karlsruhe Tel: 49-721-625370 Germany - Munich Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 Germany - Rosenheim Tel: 49-8031-354-560 Israel - Ra'anana Tel: 972-9-744-7705 Italy - Milan Tel: 39-0331-742611 Fax: 39-0331-466781 Italy - Padova Tel: 39-049-7625286 Netherlands - Drunen Tel: 31-416-690399 Fax: 31-416-690340 Norway - Trondheim Tel: 47-72884388 Poland - Warsaw Tel: 48-22-3325737 Romania - Bucharest Tel: 40-21-407-87-50 Spain - Madrid Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 Sweden - Gothenberg Tel: 46-31-704-60-40 Sweden - Stockholm Tel: 46-8-5090-4654 UK - Wokingham Tel: 44-118-921-5800 Fax: 44-118-921-5820
Atlanta Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 Austin, TX Tel: 512-257-3370 Boston Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 Chicago Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 Dallas Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 Detroit Novi, MI Tel: 248-848-4000 Houston, TX Tel: 281-894-5983 Indianapolis Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 Los Angeles Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 Raleigh, NC Tel: 919-844-7510 New York, NY Tel: 631-435-6000 San Jose, CA Tel: 408-735-9110 Tel: 408-436-4270 Canada - Toronto Tel: 905-695-1980 Fax: 905-695-2078			