



sigfox

Kommunikation fürs IoT

Pi and More 10

Nico Maas



Nico Maas

IT Systemelektroniker

Bachelor of Science

mail@nico-maas.de

www.nico-maas.de

@nmaas87



- **I. Einführung**
 - Was ist Sigfox
 - Welche Besonderheiten hat Sigfox?
 - Wie ist der Verbreitungsgrad?
- **II. Demo mit Wisol EVK**
 - Evaluation Kit besorgen (hier: Wisol EVK EVBSFM10R Rev.3)
 - Evaluation Kit registrieren
 - Callback erstellen
 - Nachrichten senden
- **III. Abschluss**
 - Fragen
 - Quellenangaben



Dieser Vortrag wurde in keinem Bereich von Sigfox finanziell unterstützt oder gesponsert.

Allerdings wurden von Wisol Evaluation Kits bereitgestellt um echte Erfahrung mit dem Sigfox Netzwerk und der Wisol Hardware zu erlangen.

Vielen Dank an Wisol!



I. Einführung



■ **Sigfox:**

- Baut keine Chips oder fertige Produkte
- Design vom Protokoll, Betrieb vom Carrier Network und Lizenzierung

■ **Prinzipien:**

- Low Power Wide Area Network
- Low Cost
- Low energy consumption
- Ease of use
- Long range
- Operated ("Carrier Network")
- Frequency-independent (ISM Band)
- Embedded subscriber identification (DEVID + PAC [Porting Authorization Code])
- Penetration (Structures / Anti-Jam -> Ultra Narrow Band)

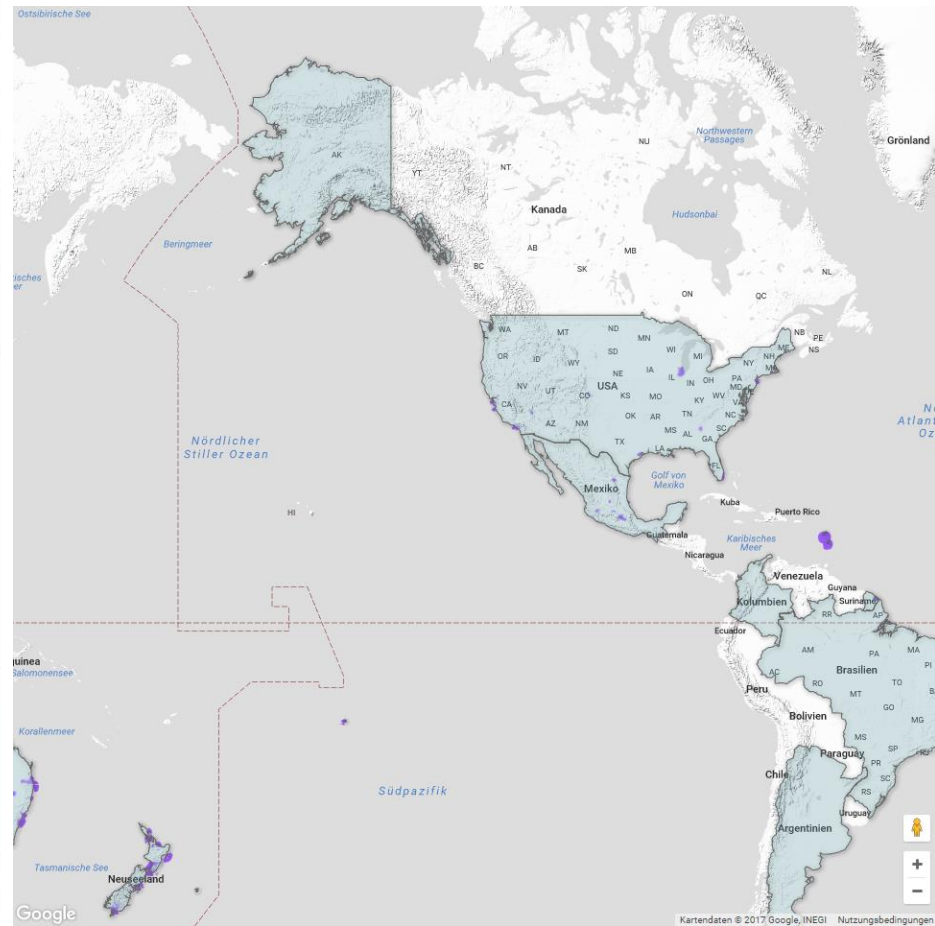
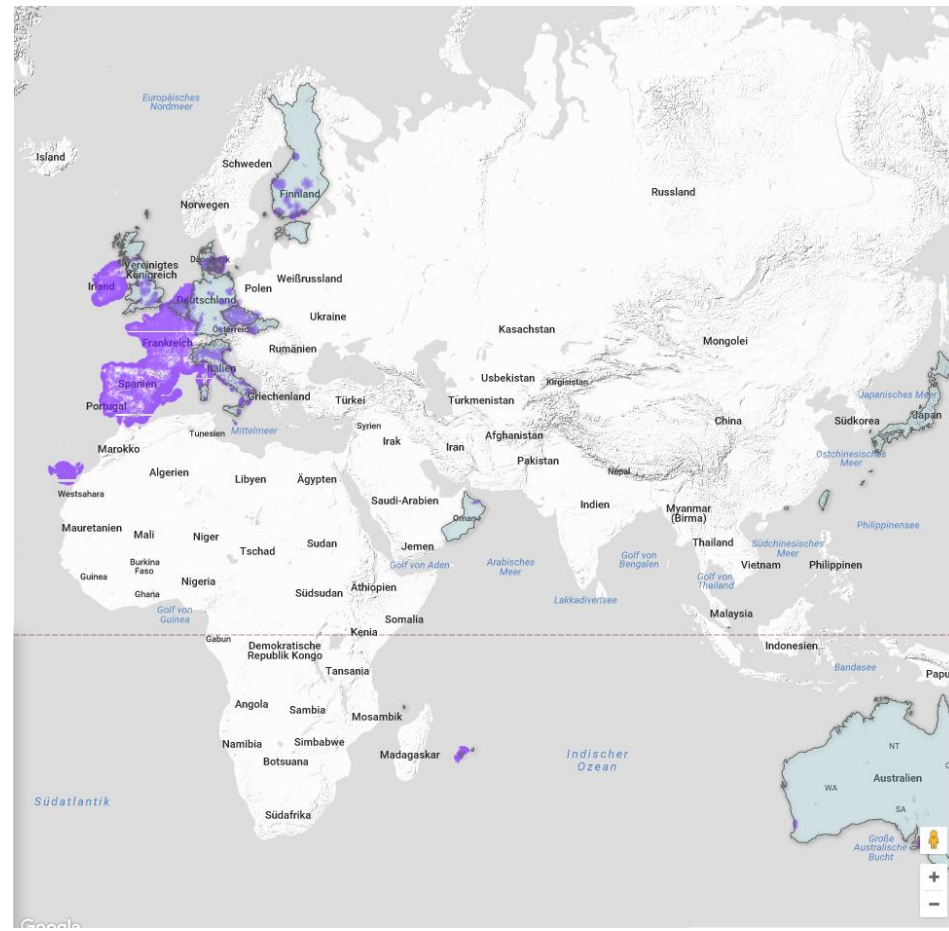


- **ISM Band (868 MHz Europa / 915 MHz USA)**
 - unterschiedliche Zonen:
 - RCZ1 Europe
 - RCZ2 US
 - RCZ3 TBD
 - RCZ4 Australia/New Zealand
- **Reichweite: 30 – 50 km (ländliche Gegend), 3 – 10 km (Stadt)**
- **Max. Reichweite: +100 km bei LoS**
- **Ca. 1 Millionen "Things" pro Station**
- **Energiebedarf pro Station 1/1000 im Vergleich zu GSM**
- **Tx: <50mA, für wenige Sekunden, 25mW, 14dB / Standby: ~ μ A**



- **Bidirektional**
- **Durchsatz: 100 bits / sec**
- **Uplink**
 - Payload: 12 Bytes = 96 Bit
 - Nachrichten: 140 / Tag
- **Downlink**
 - Payload: 8 Bytes = 64 Bit
 - Nachrichten: bis zu 4 / Tag
 - "Due to ETSI regulation, [...] devices are limited to 4 [...] messages a day."
(<http://makers.sigfox.com/getting-started/>)
- **Jede Kommunikation wird vom Client initialisiert**
- **RESTful API (JSON Payloads)**

Einführung: Verbreitung





II. Demo mit Wisol EVK



- **I. Evaluation Kit besorgen**
- **II. Evaluation Kit registrieren**
- **III. Callback erstellen**
- **IV. Nachrichten senden**

Demo: I. Evaluation Kit besorgen



■ Partner System anmelden:

- <https://partners.sigfox.com/>

SIGFOX PARTNER NETWORK
« Connecting the dots to make the IoT happen »

GET STARTED ON IOT WITH SIGFOX

Discover all Sigfox Ready products and contact the relevant partners for your IoT project

[FIND AN EXISTING SOLUTION](#)

[BUILD A NEW DEVICE](#)

Demo: I. Evaluation Kit besorgen

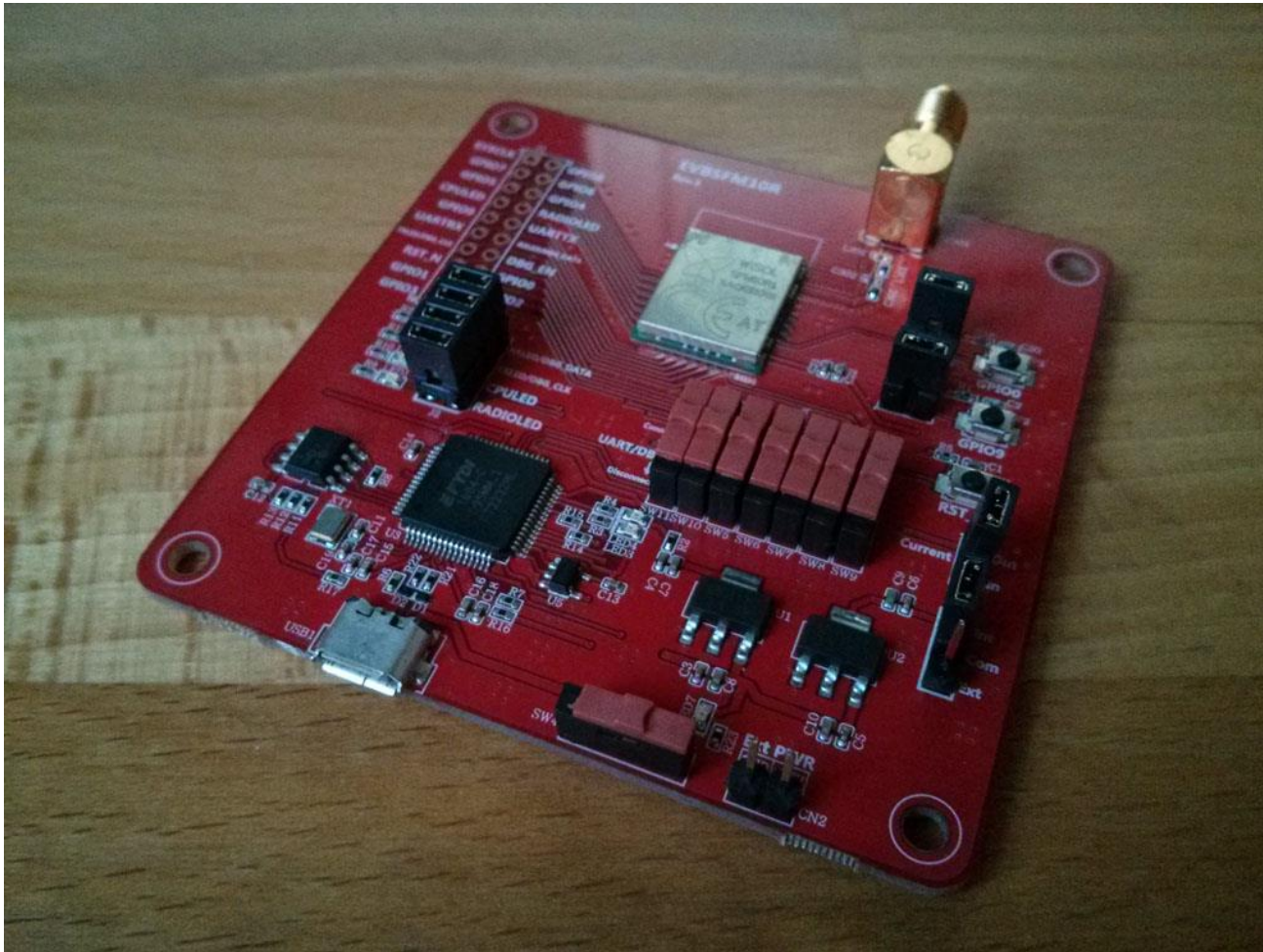


- **Richtigen Supplier finden + Evalkit für die RCZ1 Zone (Europa)**
- **ggf. bei deren Website anmelden**
- **Lieferbedingungen klären und bestellen**
- **Spaß mit Zoll und Co KG (Einfuhrzölle und Liefergebühren betrachten!)**

Demo: I. Evaluation Kit besorgen



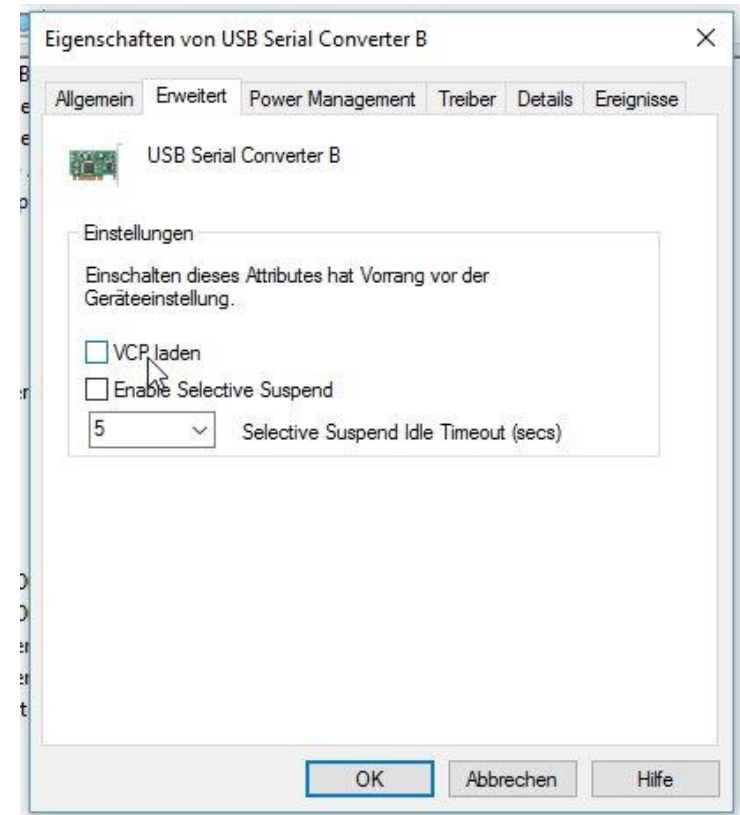
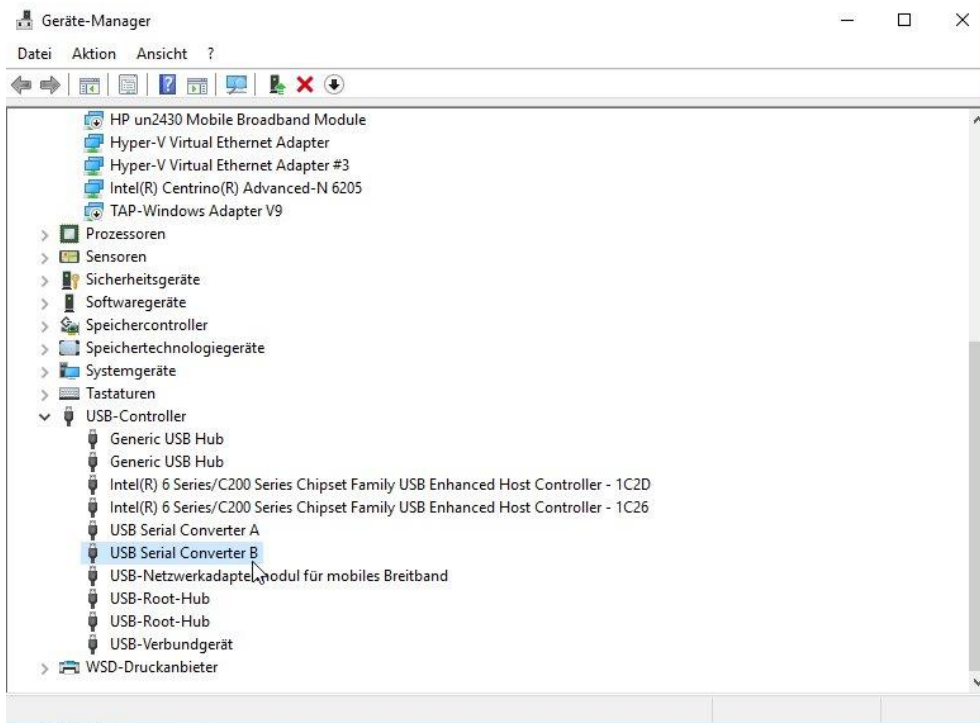
- **Wisol EVBSFM10R1**



Demo: II. Evaluation Kit registrieren



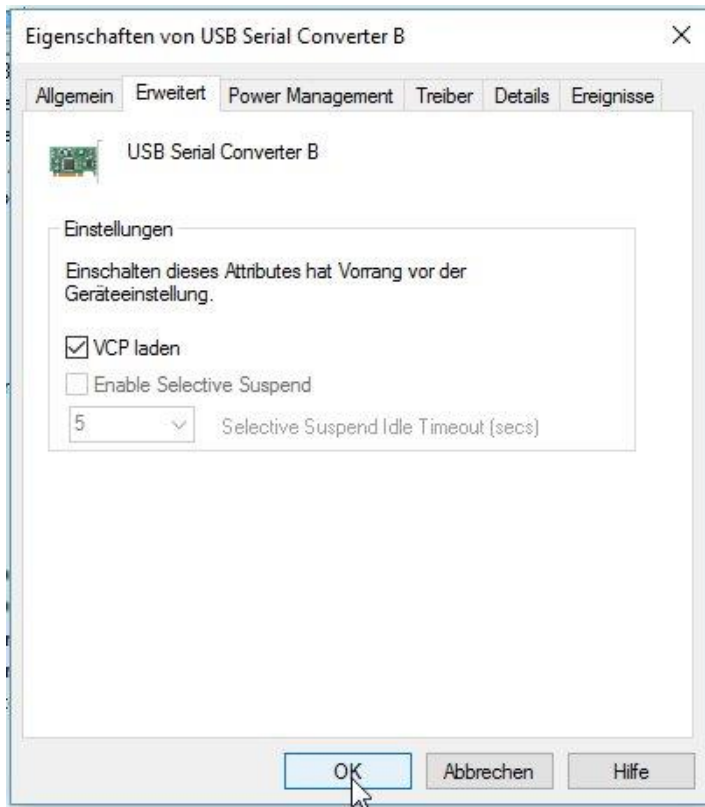
- Treiber herunterladen und installieren
- VCP aktivieren



Demo: II. Evaluation Kit registrieren



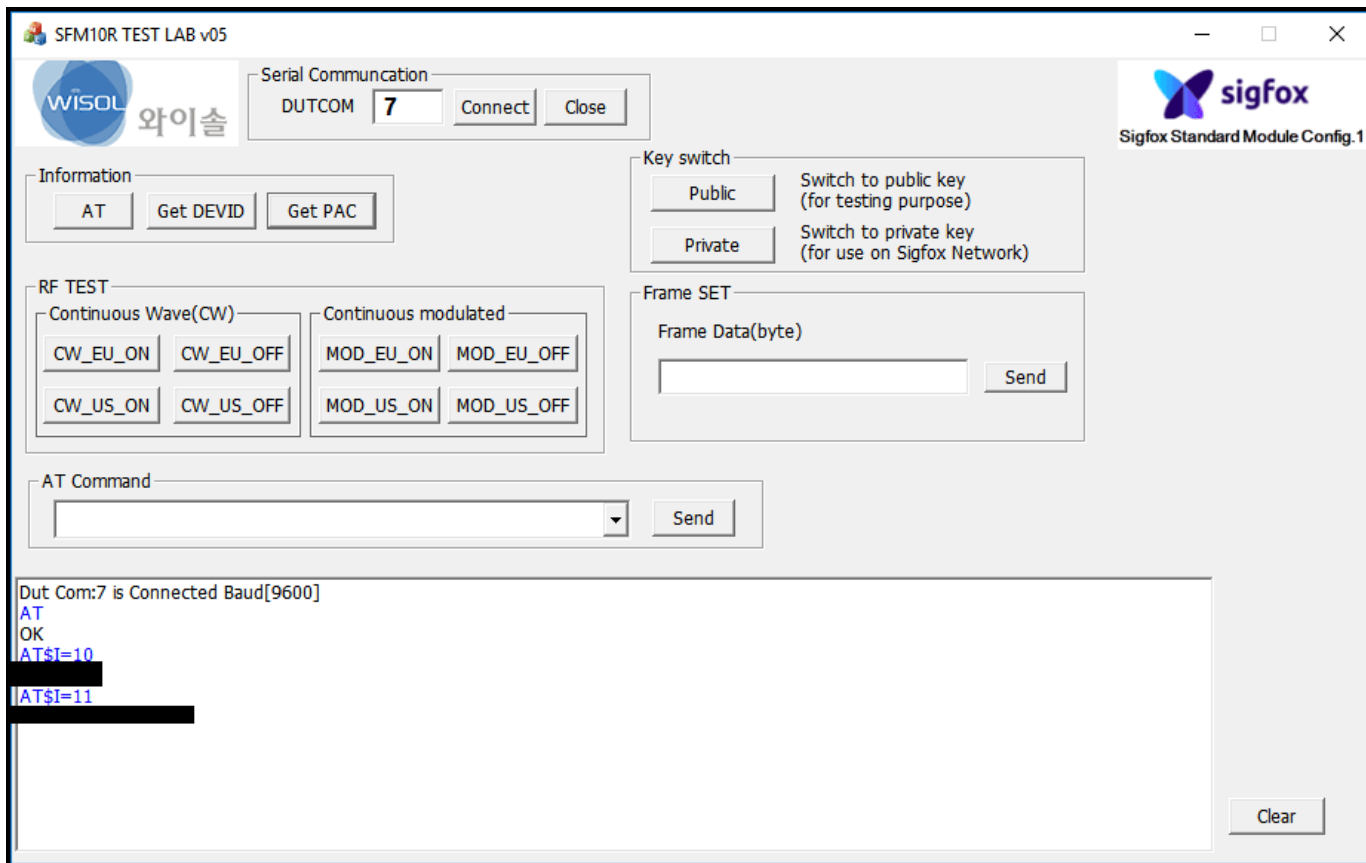
- Evalboard nochmal abstecken und neu anschließen
- Gerät erscheint als Serial Anschluss



Demo: II. Evaluation Kit registrieren



- Testsoftware herunterladen und starten
- COM Nummer eingetragen, verbinden und DEVID und PAC auslesen



Demo: II. Evaluation Kit registrieren



- Auf "Getting Started" Website gehen:
 - <http://makers.sigfox.com/getting-started/>

A screenshot of the 'Getting Started' page from the Sigfox website. The page has a purple header with the text 'Getting Started' in white. Below the header, there is a paragraph of text, a list of five links, and another paragraph of text. The background of the page is white with a faint purple circuit board pattern.

Getting Started

This getting started guide will walk you through sending your first Sigfox message, setting up basic callbacks and configuring the downlink protocol. If you're new to Sigfox and want to jump start your Sigfox development, you'll find everything you need here! This guide will start by assuming you are using an external processor/MCU to communicate with the Sigfox module.

You'll find information here about the following:

- [Registering a Sigfox Device](#)
- [Sending a Message](#)
- [Creating a Callback](#)
- [Requesting a Downlink Message](#)
- [Sigfox Projects](#)

If you have any questions or concerns that are outside of the range of this guide, do not hesitate to contact us through any means via the details at the bottom of the page. We're active on Twitter and will try to respond to any questions or problems you have!

Demo: II. Evaluation Kit registrieren



■ Registrierung

- <https://backend.sigfox.com/activate>

sigfox Lost password

Dev Kit Activation

SIGFOX has partnered with a large number of companies that provide development kits or evaluation boards in order to test and prototype on SIGFOX network. Most of them come with an included subscription. By using this form you'll be able to activate your subscription and create an account on SIGFOX backend.

Choose your kit provider

Atmel cooking hacks éolane HIDn SEEK

NXP NEMEUS ON QUICKSAND

ON Semiconductor®

Radiocrfts SNOOC ST SMART@EVERYTHING

Embedded Wireless Solutions Société Nationale des Objets Connectés life.augmented

SNOOTLAB TEXAS INSTRUMENTS airboard WISOL

Demo: II. Evaluation Kit registrieren



- **DEVID und PAC eingeben**

Dev Kit Activation

Activate your SIGFOX subscription included with your Wisol kit.

Pick your country Device information Account details

DEVICE ID (HEX)

PAC

BACK **NEXT**

Demo: II. Evaluation Kit registrieren



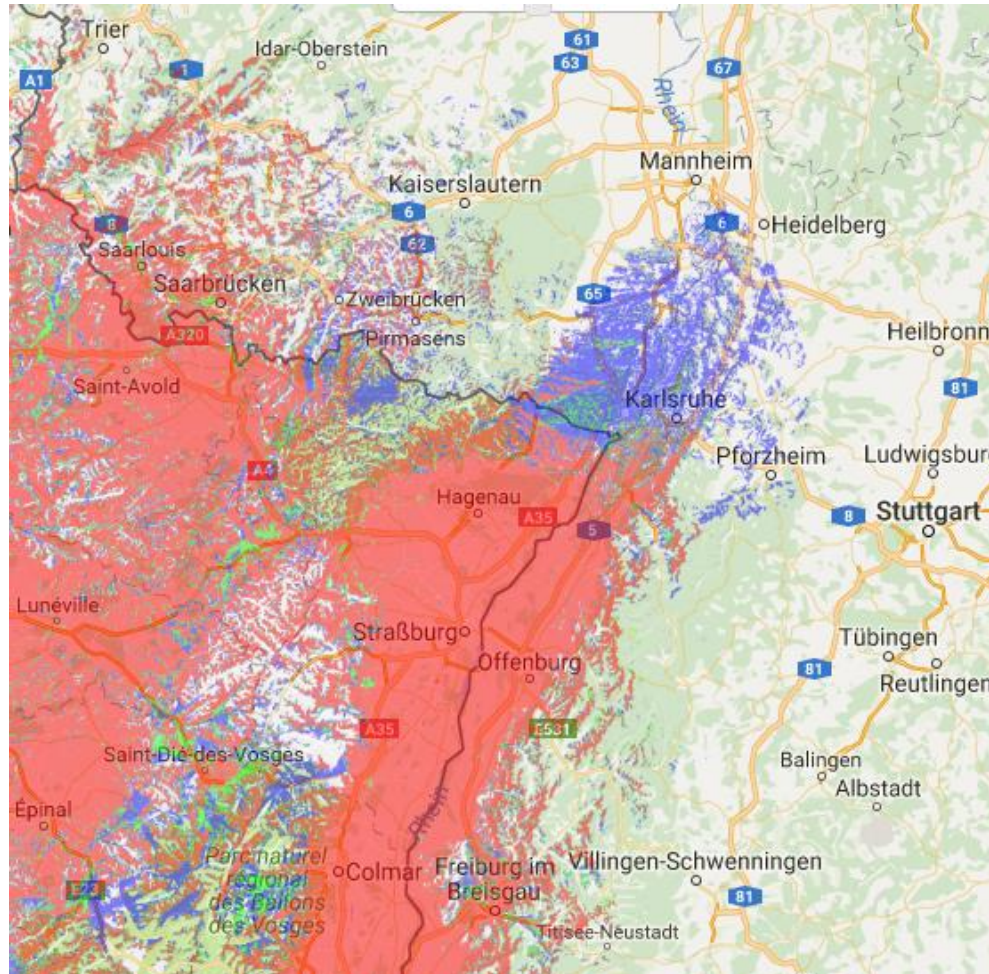
■ Provider auswählen

The screenshot shows the 'Dev Kit Activation' page on the Sigfox website. The page is titled 'Dev Kit Activation' and includes a sub-header 'Activate your SIGFOX subscription included with your Wisol kit.' Below this, there are three tabs: 'Pick your country', 'Device information', and 'Account details'. The 'Pick your country' tab is active, displaying a grid of providers for various countries. The providers are arranged in a grid with country flags above their logos. The countries shown are Australia, Belgium, Czech Republic, Finland, France, Ireland, Italy, Luxembourg, Netherlands, New Zealand, Portugal, and Slovakia. The providers listed include thinxtra, ENGIE, SimpleCell, ConnectedFinland, sigfox, VT NETWORKS, NETROTTER, rms.lu, qereq, thinxtra, NARROW NET, SimpleCell, cellnex, and arqiva. At the bottom of the page, there is a link: 'SIGFOX has been rolled-out in your country and you can't find it in the list? [Contact-us](#)'.

Demo: II. Evaluation Kit registrieren



- **Provider auswählen**



Demo: II. Evaluation Kit registrieren



Account anlegen

Dev Kit Activation

Activate your SIGFOX subscription included with your Wisol kit.



Pick your country

Device information

Account details

Account creation

Already have an account? [Sign in](#)

FIRST NAME *

LAST NAME *

EMAIL ADDRESS *

TIMEZONE

UTC ▼

POSITION

COMPANY NAME *

COMPANY ADDRESS

BACK

SUBSCRIBE

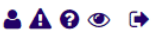
Demo: II. Evaluation Kit registrieren



Übersicht



DEVICE **DEVICE TYPE** USER GROUP



INFORMATION

LOCATION

ASSOCIATED DEVICES

DEVICES BEING TRANSFERRED

STATISTICS

EVENT CONFIGURATION

CALLBACKS

Device type 'Wisol Nico Maas kit' - Associated devices

Export devices Id/Pac

Id

State

Average SNR (all)

Last seen from date

Count: 1 / 1

page 1

Average Rssi	Average SNR	Communication status	Device type	Id	Last seen	Name	Token state
			Wisol Nico Maas kit		N/A		

page 1

Demo: III. Callback erstellen



■ Auf Device Type klicken, Callbacks, New

sigfox DEVICE **DEVICE TYPE** USER GROUP ▼ [User] [Alert] [Help] [Share]

CALLBACKS

Device type 'Wisol Nico Maas kit' - New Callback

Create callbacks to connect SIGFOX cloud to your server/platform.
A callback is a custom http request containing your device(s) data, along with other variables, sent to a given server/platform when the aforesaid device(s) message is received by SIGFOX cloud.

- Custom callback**
Creates a new callback from SIGFOX cloud to **your own server**. This is the "default" callback type. You can create a full custom request (http method, content type, headers, etc).
- AWS IoT**
AWS IoT is a managed cloud platform that lets connected devices easily and securely interact with cloud applications and other devices. AWS IoT can support billions of devices and trillions of messages, and can process and route those messages to AWS endpoints and to other devices reliably and securely.
- AWS Kinesis**
Amazon Kinesis is a platform for streaming data on AWS, offering powerful services to make it easy to load and analyze streaming data, and also providing the ability for you to build custom streaming data applications for specialized needs.
- Microsoft Azure™ Event hub**
Event Hubs is an event processing service that provides event and telemetry ingress to the cloud at massive scale, with low latency and high reliability. This service is especially useful for: application instrumentation, user experience or workflow processing, Internet of Things (IoT) scenarios.
- Microsoft Azure™ IoT hub**
Azure IoT Hub is a fully managed service that enables reliable and secure communications between millions of IoT devices and a solution back end. Azure IoT Hub enables secure communications using per-device security credentials and access control. Note that the devices are **automatically created** on the IoT hub if needed.

Demo: III. Callback erstellen



■ Callback definieren (<https://backend.sigfox.com/apidocs/callback>)

Callbacks

Type

Channel

Send duplicate

Custom payload config

URL syntax: [http://host/path?id={device}&time={time}&key1={var1}&key2={var2}...](#)
Available variables: device, time, duplicate, snr, station, data, avgSnr, lat, lng, rssi, seqNumber
Custom variables:

Url pattern

Use HTTP Method

Send SNI (Server Name Indication) for SSL/TLS connections

Headers

header	value
--------	-------

Content type

Body

```
{
  "deviceId": "{device}",
  "time": "{time}",
  "duplicate": "{duplicate}",
  "snr": "{snr}",
  "station": "{station}",
  "data": "{data}",
  "avgSnr": "{avgSnr}",
  "lat": "{lat}",
  "lng": "{lng}",
  "rssi": "{rssi}",
  "seqNumber": "{seqNumber}"
}
```

Ok Cancel



■ Ggf. Custom payload definieren

Custom payload
config



Custom message type decoding grammar

The "custom format" grammar is as follows :

```
Format = field_def [ " " field_def ] * ;
field_def = field_name ":" byte_index ":" type_def ;
field_name = (alpha | digit | "#" | "_" ) * ;
byte_index = [ digit ] * ;
type_def = bool_def | char_def | float_def | uint_def ;
bool_def = "bool:" ("0" | "1" | "2" | "3" | "4" | "5" | "6" | "7") ;
char_def = "char:" length ;
float_def = "float:" ("32" | "64") [ ":" ( "little-endian" | "big-endian" ) ] ;
uint_def = "uint:" ("8" | "16" | "24" | "32") [ ":" ( "little-endian" | "big-endian" ) ] ;
int_def = "int:" ("8" | "16" | "24" | "32" | "40" | "48" | "56" | "64") [ ":" ( "little-endian" | "big-endian" ) ] ;
length = number * ;
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

A field is defined by its name, its position in the message bytes, its length and its type :

- the field name is an identifier including letters, digits and the '-' and '_' characters.
- the byte index is the offset in the message buffer where the field is to be read from, starting at zero. If omitted, the position used is the current byte for boolean fields and the next byte for all other types. For the first field, an omitted position means zero (start of the message buffer)

Next comes the type name and parameters, which varies depending on the type :

- **boolean** : parameter is the bit position in the target byte
- **char** : parameter is the number of bytes to gather in a string
- **float** : parameters are the length in bits of the value, which can be either 32 or 64 bits, and optionally the endianness for multi-bytes floats. Default is big endian. Decoding is done according to the IEEE 754 standard.
- **uint** (unsigned integer) : parameters are the number of bits to include in the value, and optionally the endianness for multi-bytes integers. Default is big endian.
- **int** (signed integer) : parameters are the number of bits to include in the value, and optionally the endianness for multi-bytes integers. Default is big endian.

Examples :

Format	Message (in hex)	Result
int1::uint:8 int2::uint:8	1234	{ int1: 0x12, int2: 0x34 }
b1::bool:7 b2::bool:6 i1:1:uint:16	C01234	{ b1: true, b2: true, i1: 0x1234 }
b1::bool:7 b2::bool:6 i1:1:uint:16:little-endian	801234	{ b1: true, b2: false, i1: 0x3412 }
b1::bool:7 b2::bool:6 i1:1:uint:16:little-endian i2::uint:8	80123456	{ b1: true, b2: false, i1: 0x3412, i2:0x56 }
str::char:6 i1::uint:16 i2::uint:32	41424344454601234567890A	{ str: "ABCDEF", i1: 0x123, i2:0x4567890A }



- **Callback auf Server programmieren**

```
<?php
header('Content-type: application/json');
$json = file_get_contents('php://input');
$arr = json_decode($json,true);

function writeFile_Sigfox($write){
    $fh = fopen('sigfox.txt', 'a') or die("can't open file");
    fwrite($fh, $write);
    fclose($fh);
}

writeFile_Sigfox($json);
writeFile_Sigfox($arr['deviceId']);
?>
```

Demo: IV. Nachrichten senden



- Testsoftware starten, verbinden, Private Key auswählen, Nachricht eingeben und senden




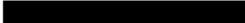
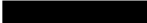

The screenshot displays the SFM10R TEST LAB v05 software interface, which is used for testing and sending messages. The interface is divided into several sections:

- Serial Communication:** Includes a DUTCOM field set to 7, with Connect and Close buttons.
- Information:** Contains buttons for AT, Get DEVID, and Get PAC.
- Key switch:** Features Public and Private buttons. The Private button is selected, with a note: "Switch to private key (for use on Sigfox Network)".
- RF TEST:** Divided into Continuous Wave (CW) and Continuous modulated sections, each with On/Off buttons for EU and US frequencies.
- AT Command:** A text input field for entering AT commands.
- Frame SET:** Includes a Frame Data (byte) input field containing "0123CAFE" and a Send button.
- Terminal:** A large text area at the bottom showing the output of the AT command: "Dut Com:7 is Connected Baud[9600] AT OK".

The interface also features logos for WISOL (와이솔) and sigfox (Sigfox Standard Module Config.1).



■ Nachricht wird in Callback empfangen

Average Rssi	Average SNR	Communication status	Device type	Id	Last seen	Name	Token state
 -115.89	 33.25				2017-01-02 10:05:09		

```
{  
  "deviceId":"xxxxxx",  
  "time":"1483395345",  
  "duplicate":"false",  
  "snr":"9.63",  
  "station":"xxxx",  
  "data":"0123cafe",  
  "avgSnr":"23.91",  
  "lat":"50.0",  
  "lng":"6.0",  
  "rssi":"-137.00",  
  "seqNumber":"82"  
}
```

15,93 km



III. Abschluss



■ Vorteile Sigfox:

- ideal für kleine Sensoren
- verbraucht wenig Strom -> lange Standby Zeiten

■ Nachteile Sigfox:

- Keine Bestätigung des Nachrichten Empfanges
- Nur beschränkte Möglichkeit zum Sensor selbst zu kommunizieren (4 Nachrichten pro Tag)
- Kosten / Subscription relativ undurchsichtig (zw. 14€ und 1€ pro Gerät / Jahr, ohne Gewähr)
- Security?

Fragen?



Vielen Dank für Ihre Aufmerksamkeit!



- Bilder, soweit nicht anders angegeben, von sigfox.com oder selbst erstellt
- <http://www.eejournal.com/archives/articles/20141103-sigfox/>
- <http://www.rfwireless-world.com/Terminology/SIGFOX-technology-basics.html>
- <https://github.com/sigfox/makers-tour/blob/master/presentation/2016/presentation.pdf>
- https://radiocrafts.com/uploads/AN018_SIGFOX_Frequently_Asked_Questions_2_1.pdf
- <https://developers.thethings.io/serialization-formats-sigfox.html>
- <http://makers.sigfox.com/getting-started/>