



Willkommen

zum

LoRaWAN –/ TTN– HandsOn

Beuth Hochschule für Technik, Berlin – 21./22.11.2018

Dipl.-Ing. (FH) Tasso Mulzer

Nach Vorarbeiten von:

Dr. Christian Hammel, Technologiestiftung Berlin

Dr. Benjamin Seibel, Technologiestiftung Berlin

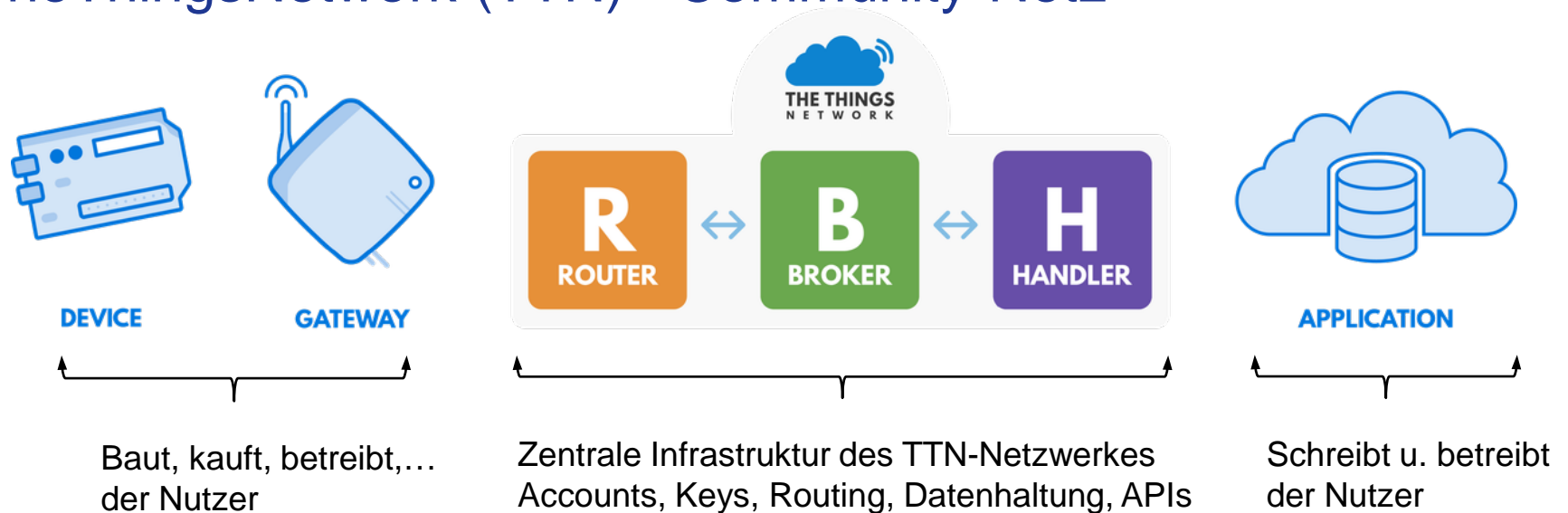


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



**TECHNOLOGIE
STIFTUNG
BERLIN**

TheThingsNetwork (TTN) - Community-Netz



Die zentrale Struktur kann jeder kostenlos nutzen. Sie bietet Geräte-Keys, Nutzerverwaltung, Paketrouting, End-zu-End-Verschlüsselung (AES-128) und API (HTTP, MQTT).

Kostenpflichtig: Leistungen der TTN Industries wie private Server, Integration in Kundensysteme,...

LoRaWAN-Sprech: Device = Mote = Node = Endgerät; Gateway = Basisstation (= Router)

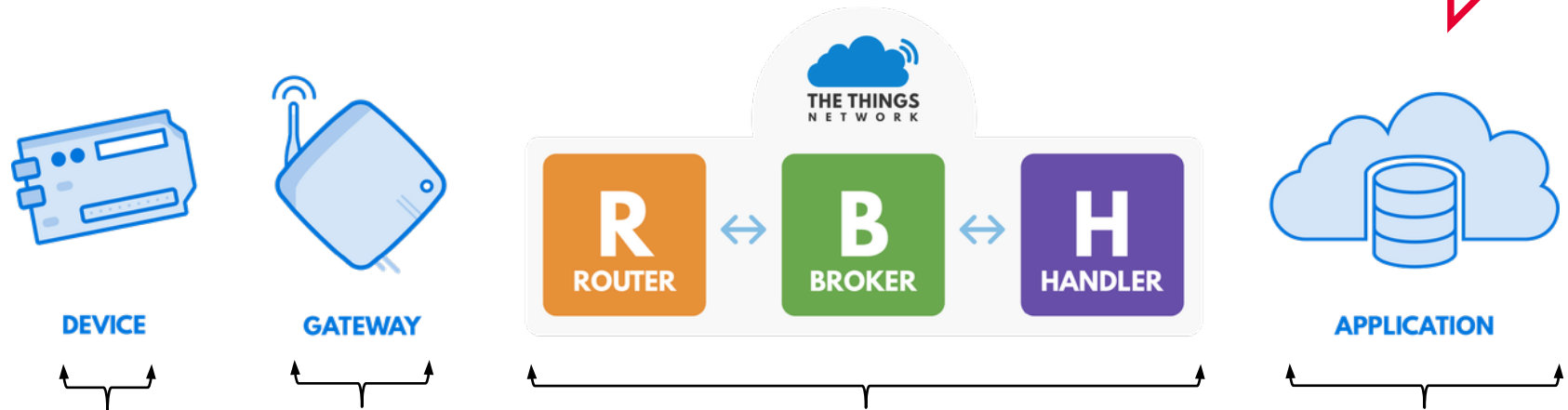
Funkregulierung: 1% Airtime, also Sendeintervalle nicht beliebig verkürzen!

Stand:

- 500 Communities weltweit, 42.000 User, 4.000 Gateways, 20.000 Applications, Schwerpunkte NL, BE, CH, UK
- TTN-Berlin: 60 Contributors, 45 Gateways

Worum geht es heute? Was machen wir?

Wir übertragen selbst gemessene Daten vom Arduino ins Netz (uplink)



Unser
Arduino!

Vorhanden

Das benutzen wir einfach.

Hier spielt die Musik,
aber heute nicht.

Wir schließen einen Sensor an.
Wir messen eine Temperatur.
Wir schalten eine LED ein.
Wir melden das Gerät bei TTN an.
Wir senden die Temperatur ins TTN.
Wir nutzen Beispielcode.

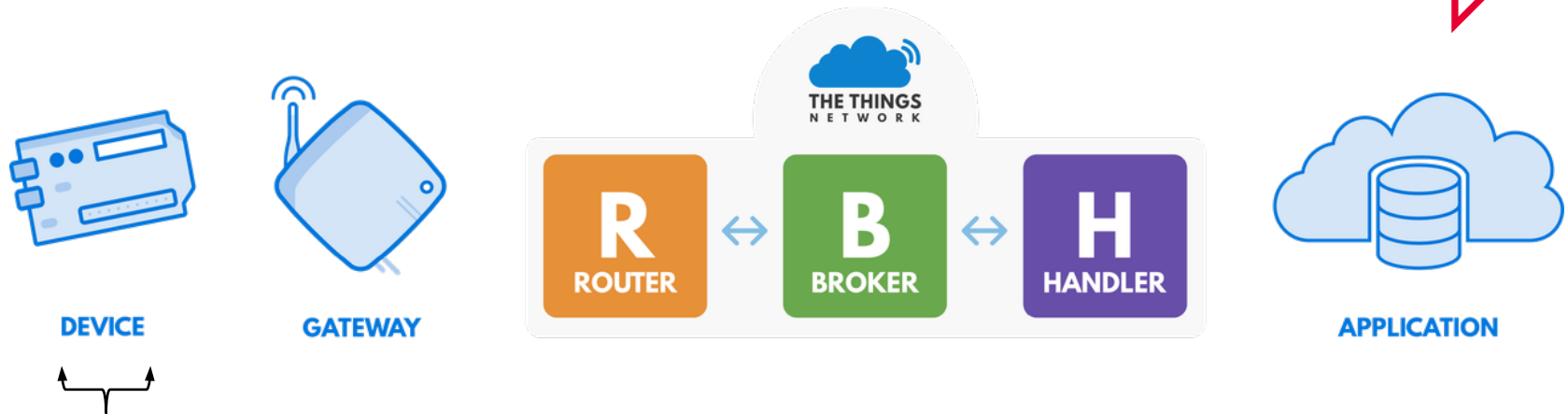
Teilnehmer sollten einen
Account haben, damit sie ihr
node anmelden und auf ihre
Daten zugreifen können.

Wenn wir Zeit haben,
lesen wir unsere
Daten mit einem
Node-Red Viewer.

Wie melde ich den Arduino an?
Wo sehe ich meine Daten?

Was genau soll unser Device machen?

Wir übertragen selbst gemessene Daten vom Arduino ins Netz (uplink)



Unser Arduino

- Misst die aktuelle Temperatur mit einem Sensor
- Blinkt mit einer Kontroll-LED, wenn die Temperatur über 30° ist, damit wir direkt sehen können, ob unsere Schaltung und unser Code funktionieren
- Überträgt die Temperatur (genauer: Bytes) ins TTN (genauer: zum TTN-Broker)

TTN

- Zeigt uns in der Ansicht „Console“ unsere Temperatur an (wenn wir die Bytes in brauchbare Daten zurückverwandelt (decodiert) haben)

Application

- Verwertet die Daten – speichert / rechnet / steuert / visualisiert.

Board zusammenbauen

1. LoraRaWAN-Shield auf den Arduino montieren

(so aufstecken, dass er gewaltfrei passt und dass die Anschlüsse auf Arduino und Shield in Deckung sind)

2. ANTENNE ANSCHRAUBEN !!!!!

(NIE ohne Antenne betreiben, das kann das Board zerstören)

Arduino-Code

1. **Arduino-Code:** Die IDE kompiliert den C-ähnlichen Code in Maschinencode, der dann auf den Arduino geladen wird. Deshalb kann man Code auch nicht vom Arduino herunterkopieren.
2. **Programmaufbau:** Es gibt immer mindestens drei Teile,
 - einen Header, in dem includes, Variablen u. ä. definiert werden,
 - einen Teil „`void setup () {}`“, der genau einmal abgearbeitet wird und
 - einen Teil „`void loop () {}`“, der ständig wiederholt wird.Man kann an beliebiger Stelle eigene Funktionen ergänzen, die sich aus anderen Programmteilen heraus aufrufen lassen.
3. **Syntax**
Funktionen: `void FUNKTIONSNAME (Rückgaben) {Code der Funktion}`
Befehlszeilen: enden immer mit `;`
Kommentar: wird mit `//` eingeleitet und nicht kompiliert
4. **Progrämmchen (Sketches)**
Sind einfache Textdateien und enden auf `.ino`
5. **Beispiele für heute:** <https://github.com/tmulzer/workshops>

Kommunikation Lappi / Arduino testen

1. Arduino-IDE starten

2. Arduino über USB-Kabel anschließen

(NIE ohne Antenne!)

3. Kommunikationstest

„Werkzeuge | Board | Mega“, „Werkzeuge | Prozessor | Mega2560“; „Werkz. | Port | \$richtigerPort“;

„Werkzeuge | Boardinformationen holen“

Wenn etwas kommt: alles richtig angeschlossen,

Wenn nicht oder Fehler: richtiger Port? Strom am Arduino? USB-Anschluss freigegeben (Linux)?

4. Testcode hochladen

Testcode der TSB (kommunikationstest_lappi_arduino.ino) in die IDE laden; „Sketch | hochladen“

Keine Fehlermeldungen? Alles prima!

5. Test

„Werkzeuge | serieller Monitor“:

Wenn bei 115200 baud sinnvoller Output kommt, ist alles prima.

Arduino bei TTN anmelden, Daten senden und decodieren

1. Bei TTN einloggen (wer keinen Account hat, muss sich einen anlegen)

TTN-Konsole:

Anwendung anlegen,

device anlegen,

Settings, ABP, generiert Schlüssel und Device-ID

2. Arduino IDE starten

TSB-Mustercode (temperatur_basteltreff.ino) in die IDE laden und auf den Arduino laden (die Schlüssel im Beispiel sind fake und produzieren Fehler)

Die Schlüssel Network Session Key, Appskey und Devicekey (≠ EUI !) in den Mustercode einpflegen (als hex, wird mit <-> erzeugt und mit dem „Arduino“-Tool konvertiert).

Eine beliebige interne ID könnt ihr einbauen, wenn ihr mehrere Nodes auseinanderhalten wollt.

3. Code hochladen

Wenn eure Schlüssel richtig eingegeben sind, dann seht ihr im seriellen Monitor Daten, die ihr erwartet und in eurer Konsole Daten, die noch nicht sinnvoll sind.

Wenn der Sensor lange genug in der Hand war (>30°) geht die LED an.

4. Payload-Decoder einrichten

Damit die gesendeten Daten wieder in eine sinnvolle Form kommen, legt ihr über die TTN-Konsole einen payload-decoder an. Code dafür (javascript) steht als Kommentar im Arduino-Testcode.

5. Code verstehen

versuchen wir direkt im Code, dafür sind Kommentare im Code.

Kommunikation Arduino / Gateway testen

1. Testprogramm laden

„Datei | Öffnen... | ttn-name.ino“

2. LMIC (LoRaWAN-Bibliothek einbinden)

„Sketch | Bibliothek einbinden“ | „MCCI LoRaWAN LMIC library“

3. Konfiguration für Europa

„Eigene Dokumente\Arduino\libraries\MCCI_LoRaWAN_LMIC_library\project_config\lmic_project_config.h“:
#define CFG_eu868 1 aktiv, #define CFG_us915 1 mit //auskommentieren.

4. Testcode anpassen:

1. <Name>

2. <NwkSKey>

3. <AppSKey>

4. <DevAddr>

... evtl einfacher aus der TTN-Konsole kopieren!

5. Testcode hochladen

Testcode der BHT (ttn-name.ino) in die IDE laden; „Sketch | hochladen“
Keine Fehlermeldungen? Alles prima!

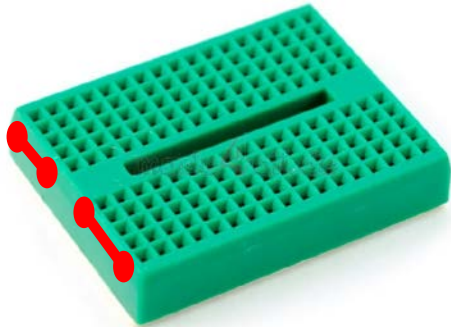
6. Test


Wenn eure Daten in der TTN-Console ankommen, ist alles prima.

Schaltungen aufbauen

Kontroll-LED: Arduino und Breadboard:

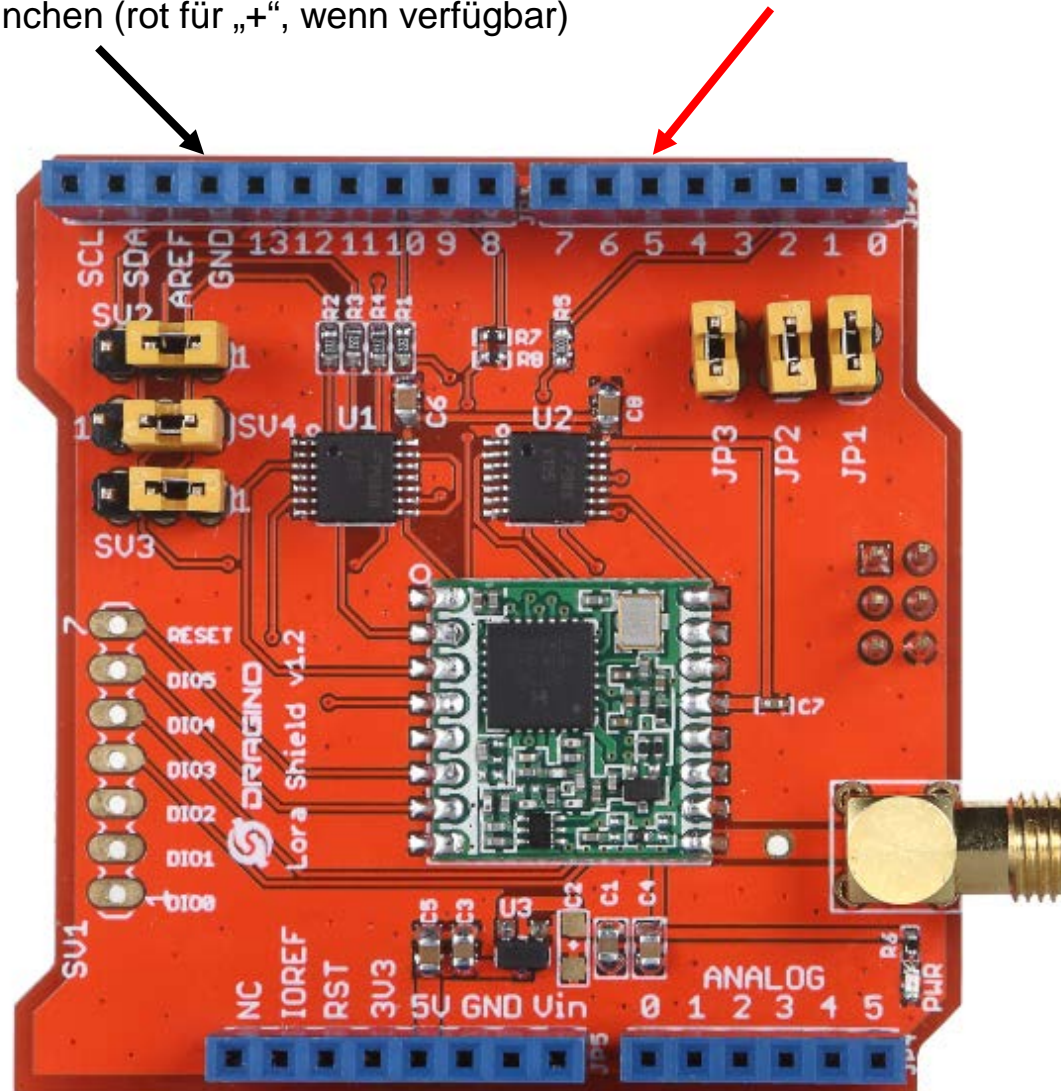
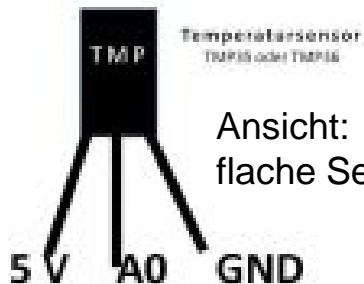
- Pin GND (auf der „Digitalseite“ vom Arduino → Vorwiderstand → kurzes LED-Bein (sw. Kabel, wenn da)
- Pin 5 (Arduino, Digitale Seite) → langes LED-Beinchen (rot für „+“, wenn verfügbar)



 Diese 5 Anschlüsse sind verbunden.

Sensor am Arduino anschließen:

- GND an GND (Analogseite)
- 5V an 5V (Analogseite)
- Messausgang auf Analogeingang 0

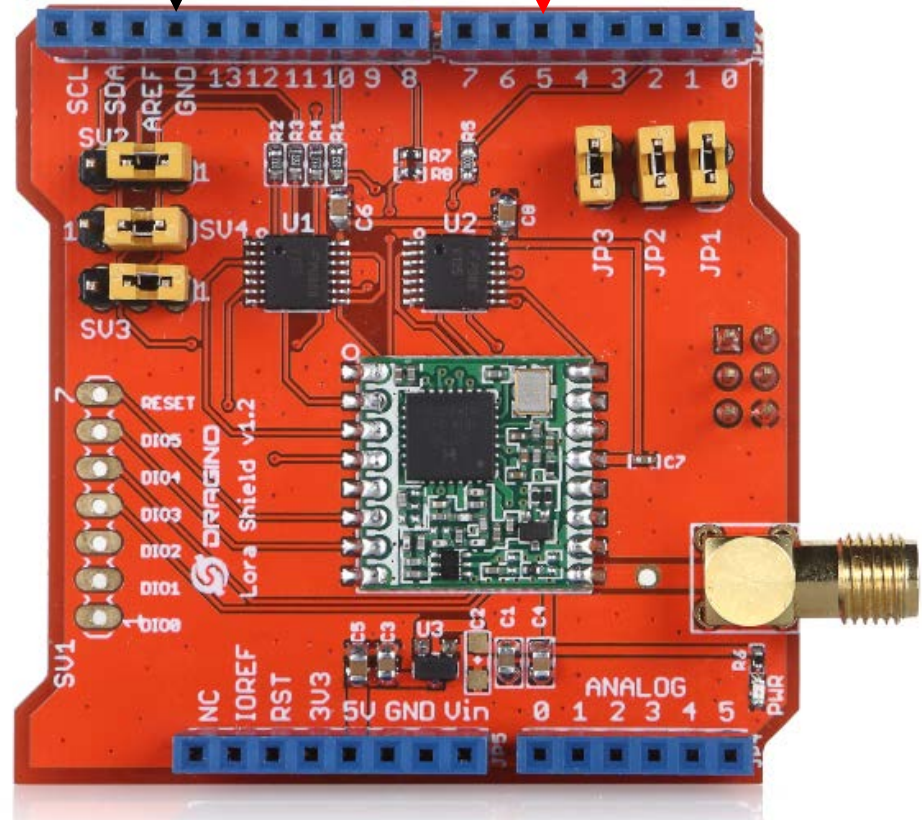
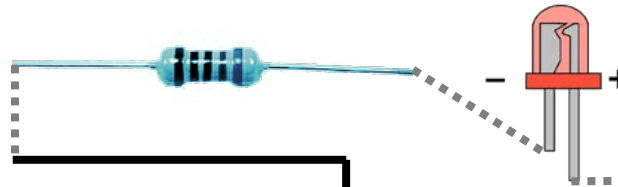


Schaltungen aufbauen: LED mit Breadboard verkabeln

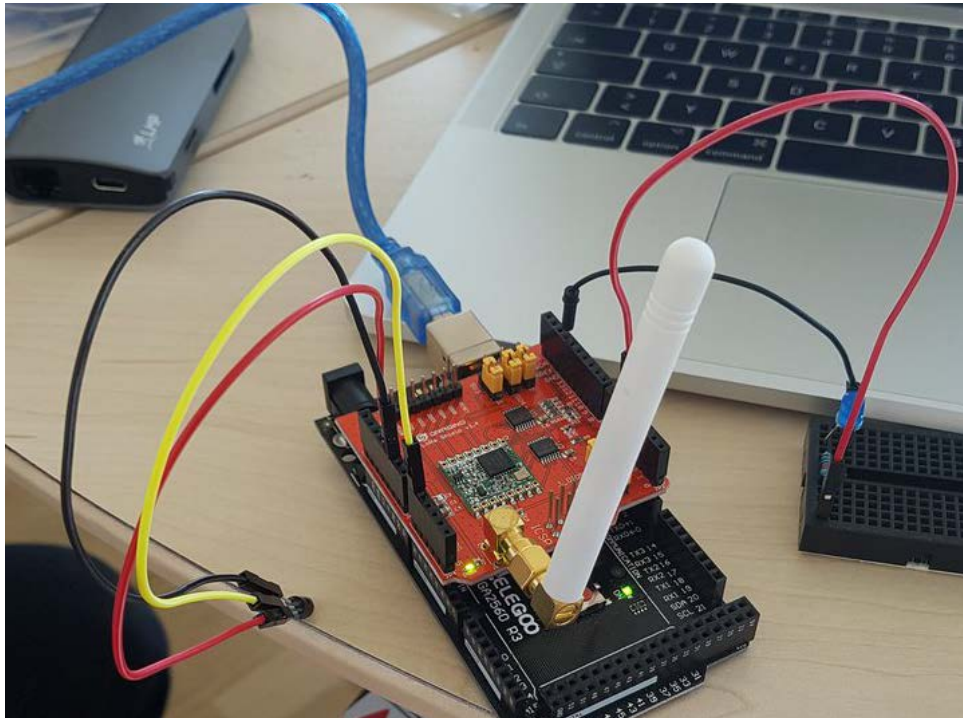
Kontroll-LED: Arduino und Breadboard:

- Pin GND (auf der „Digitalseite“ vom Arduino → Vorwiderstand → kurzes LED-Bein (sw. Kabel, wenn da)
- Pin 5 (Arduino, Digitale Seite) → langes LED-Beinchen (rot für „+“)

..... im Breadboard

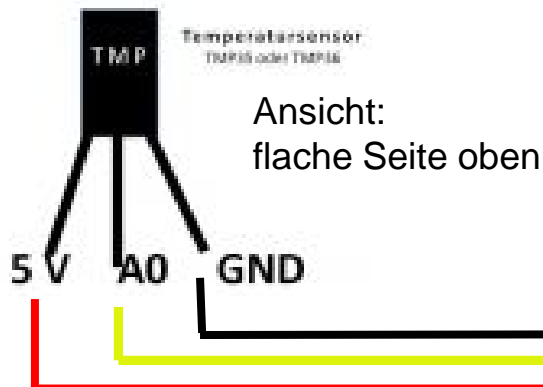
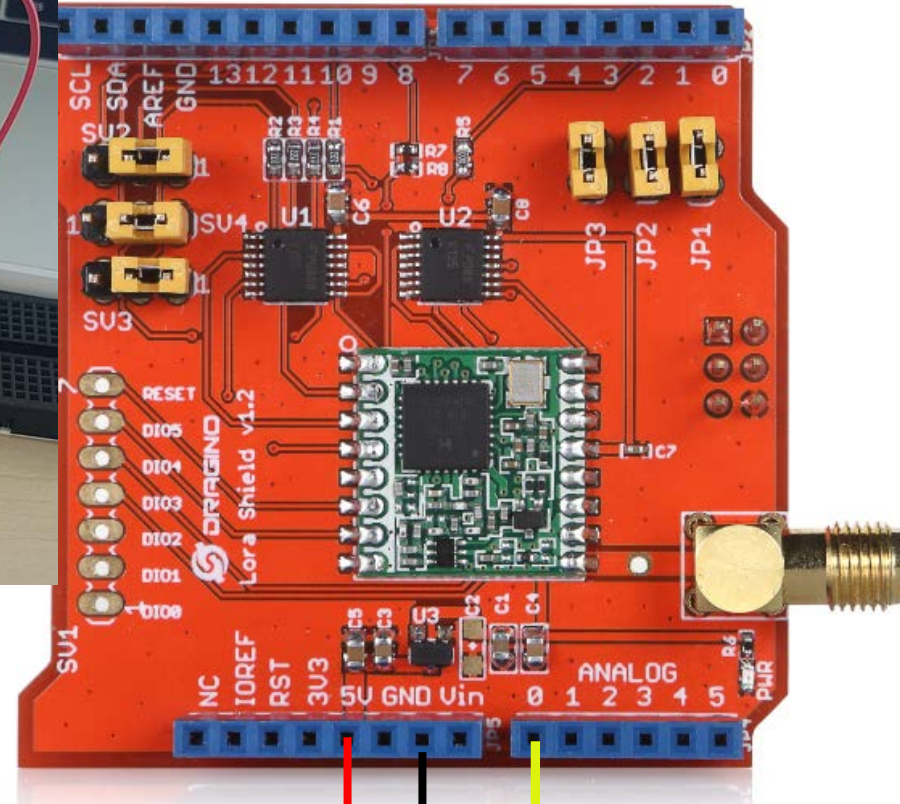


Schaltungen aufbauen: Sensor



Sensor am Arduino anschließen:

- GND an GND (Analogseite)
- 5V an 5V (Analogseite)
- Messausgang auf Analogeingang 0



Daten ansehen und prüfen

1. Direkt bei TTN

TTN-Konsole: Bei eurer Application / Eurem Device könnt ihr die Daten live sehen und wenn der Decoder funktioniert, werden sie auch entschlüsselt.

2. Von TTN auslesen

über http: geht, sprengt hier aber schnell den Rahmen

über MQTT: geht, sprengt hier aber schnell den Rahmen

über Node-Red: fertiges Beispiel steht unter:

<http://141.64.71.71:1880/#flow/2>



Cooler Anwendungen? Eigene Ideen?

Wer mehr ausprobieren mag, ist herzlich willkommen.

Wer gerne Leiterplatten, Mechanik, Elektronik, Informatik, Mechatronik, ... bauen, entwickeln, austüfteln, basteln ... und zusammenbringen möchte, ist hier im FVM-Labor richtig. Gerne auch wieder zusammen mit der TSB.

<http://labor.beuth-hochschule.de/fvm/>

Tasso Mulzer - tasso.mulzer@beuth-hochschule.de

[@Knurpsl](#)

Beispiele für schicke LoRaWAN-Anwendungen von Bürgern:

- Umweltsensoren: <https://skopjepulse.mk/>
- Badewasser im Dortmund-Ems-Kanal: <http://wiekaltistderkanal.de/ui/#/0>

Wer mit der Technologiestiftung Bildungsmaterial und Unterlagen für die Hackingbox IoT entwickeln mag, ist dort ebenfalls herzlich willkommen.

Carolin Clausnitzer - clausnitzer@technologiestiftung-berlin.de

[@caroclausnitzer](#)

Anhang

1. Arduino-IDE / Linux / Zugriff auf serielle Schnittstelle

Herausfinden, zu welcher group die Schnittstellenuser gehören:

`ls -l /dev/ttyUSB*` oder `ls -l /dev/ttyACM*` liefert etwas in der Art:

```
crw-rw---- 1 root tty 188, 0  5 apr 23.01 ttyUSB0
```

Die Angabe in der vierten Spalte (hier „tty“) zeigt, welche Nutzergruppe Zugriff hat.

Dieser Gruppe müssen wir noch beitreten (<username> ist unser Linux-Benutzername):

```
sudo usermod -a -G tty <username>
```

Jetzt noch mal neu starten, dann sollte der Zugriff klappen.