# Ein Arduino-basierter Radioaktivitäts-Messknoten für das LoRaWAN IOT-Netzwerk „The Things Network“ mit dem PiGI-Geigerzähler-Modul

Bernd Laquai, 8.12.2018

Leider ist das Geigerzähler-Modul für den unter /1/ beschriebenen LoRaWAN Messknoten zumindest temporär nicht mehr lieferbar. Die unter /2/ beschrieben Alternative ist sicher eine interessante Alternative für Bastler, aber wer gerne einen Messknoten ganz zügig aus fertigen Modulen zusammensetzen möchte, für den gibt es in der Zwischenzeit auch noch eine einfache Ersatzlösung: Das Apollo-NG PiGI-Aufsteckboard, das ursprünglich für den Raspberry Pi entwickelt wurde und als Open Source Projekt veröffentlicht wurde. Es eignet sich genauso für den Arduino.

Das PiGI-Modul ist mit ca. 5mA Stromaufnahme für 450V Betriebsspannung des Zählrohrs sehr sparsam und kann daher auch vom Mikrocontroller-Board versorgt werden. Neben der Bereitstellung der Hochspannung, die in einem weiten Bereich an einem Poti eingestellt werden kann, enthält es auch einen Zählimpulsverstärker, der die analogen Zählimpulse in digitale wandelt, so dass ein Mikrocontroller z.B. mit einem Interrupt darauf reagieren kann.

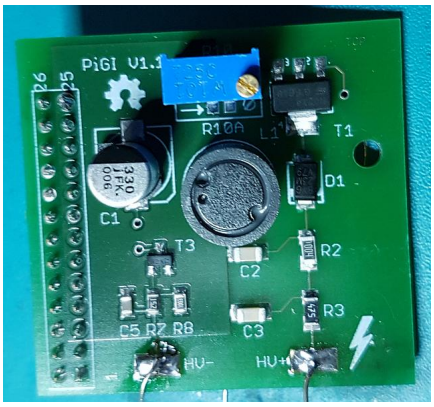


Abb. 1: Das PiGI- Geiger-Müller Zähler-Modul

Dieses Board wurde recht erfolgreich von der Regionalgruppe Aachen des „Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung e.V.“ (FiFF) eingesetzt um das TDRM Messnetz (Tihange Doel Radiation Monitoring) aufzubauen. Über ein Mitglied dieser Gruppe kann das Board auch als fertiges Modul bezogen werden ( eMail: PeterKa@TreeDev.eu ).

Die Funktion des Moduls ist etwas trickreich und nicht gleich auf Anhieb zu verstehen. Kern der Schaltung ist auch wieder ein Step-Up Konverter, bestehend aus dem Transistor T1, der Induktivität L1, der Diode D1 und der Kapazität C2.

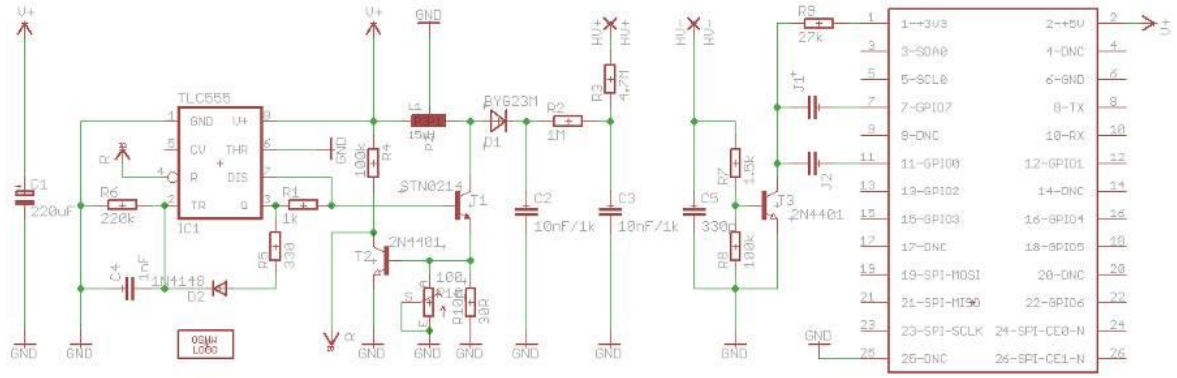


Abb. 2: Schaltplan des PiGI-Aufsteckmoduls

Der Hochspannungskapazität C2 ist noch ein weiteres RC-Siebglied, bestehend aus R2 und C3, nachgeschaltet um etwaige Spikes in der Hochspannung zu unterdrücken. R3 ist ein Anodenvorwiderstand mit 4.7MΩ, der den Strom durch das Zählrohr begrenzt und für viele gängige Zählrohre passend ist. Das Geiger-Müller Zählrohr wird an HV+ mit der Anode und an HV-mit der Kathode angeschlossen.

Bei diesem Modul wird das Zählimpulssignal kathodenseitig abgenommen. Es muss also beachtet werden, dass die Kathode nicht auf Masse liegt, sondern das schwache Zählimpulssignal trägt. Daher sollte das Zählrohr nicht in die Nähe von Quellen kommen, die Störstrahlung emittieren wie z.B. Handy, Motoren o.ä., und die gesamte Außenseite des Zählrohrs sollte auch mit keinem Leiter in Berührung kommen. Die Zählimpulse erzeugen am Widerstand R8 einen Spannungsabfall und werden vom Transistor T3 verstärkt. Der Kollektorwiderstand von T3 muss auf der Spannung, mit dem das IO-Interface des Mikrocontrollers betrieben werden soll liegen. Haben die IO‘s wie beim RasPi einen High-Pegel von 3.3V muss R8 auf 3.3V gelegt sein. Beim Arduino Uno oder Leonardo haben die IO‘s einen High Pegel von 5V, daher muss in diesem Fall R8 an 5V angeschlossen werden. Die eigentlichen Zählimpulse sind beim PiGI-Board „low-aktiv“ und werden über den Jumper J1 (das Symbol dafür im Schaltplan ist keine Kapazität!) und Pin 7 der Steckerleiste an den Mikrocontroller angekoppelt (der Jumper J2 an Pin 11 der Steckerleiste ist in der Regel auf dem Board nicht bestückt).

Getaktet wird der Hochspannungstransistor T1 vom CMOS Timer-Baustein TLC555. Allerdings ist in die Pulserzeugung auch der Emitterwiderstand R10 involviert (nur der Poti, der Festwiderstand ist nicht bestückt). An diesem entsteht ein Spannungsabfall, der zum Strom durch die Spule und durch den Hochspannungstransistor T1 proportional ist. Dieser steigt von Null aus gemäß $i(t)=-1/L*U_0*t$ linear an, wenn T1 durchgeschaltet ist. Sobald dieser Spannungsabfall an R10 groß genug ist, schaltet auch T2 durch, welcher den Reset auslöst, so dass der Ausgang am TLC555 von bisher High auf Low geht. Nach einer gewissen Zeit, springt der Timer dann wieder zurück auf High und der nächste Schaltzyklus beginnt. Da die Größe des Spannungsabfalls den Rücksetz-Zeitpunkt bestimmt, kann über den Poti die Dauer des Stromflusses durch die Spule sowie die Frequenz der Pulse und damit auch die Spannung, welche der Step-Up Konverter erzeugt, eingestellt werden.

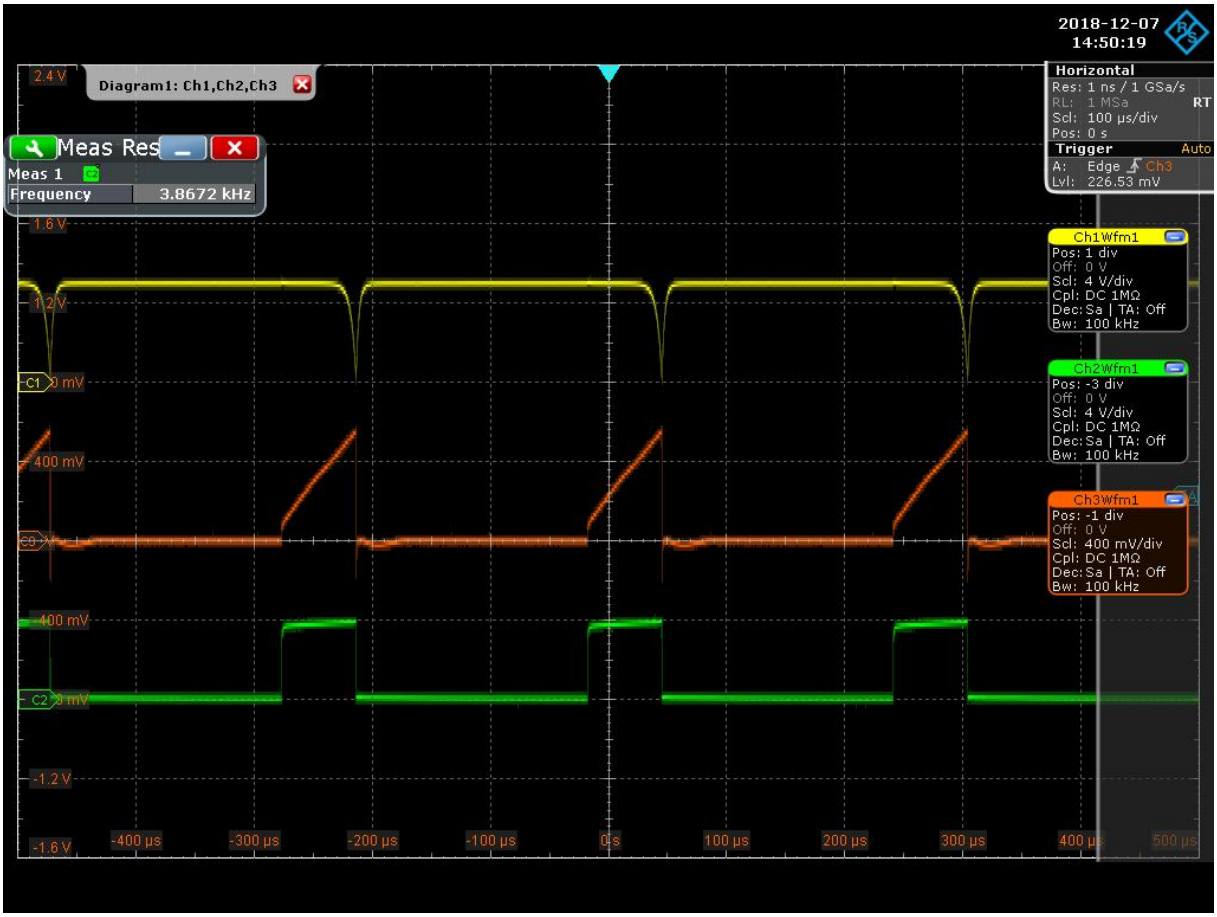


Abb. 3: Steuersignale am PiGI-Board, gelb: Reset-Signal am TLC 555 (Kollektor von T2), orange: Spannung an R10 (Basisspannung an T2), grün: Output am TLC555

In der Originalanwendung, zusammen mit dem RasPi, passt der Steckverbinder direkt zu seinem Gegenstück auf dem Controller-Board und kann daher ganz einfach aufgesteckt werden. Setzt man das PiGI-Board für den Arduino ein, muss man es als „Break Out“ Modul betrachten und den Steckverbinder per Kabel an die Stiftleiste eines Arduino anschließen. Wenn man einen Arduino mit 5V IO‘s nutzen möchte, muss der Pin1 am PiGI-Board mit dem 5V Anschluss des Arduino-Board verbunden werden, bei einem neueren Arduino mit 3.3V IO‘s (z.B. Maker Familie oder Arduino Due) muss man den Pin 1 am PiGI-Board mit dem 3.3V Anschluss des Arduino-Board verbinden, damit der High-Pegel des Zählimpulsausgangs zum Controller passt. Der Zählimpulsausgang selbst liegt am PiGI-Board auf Pin Nr. 7 und muss beim Arduino-Board auf den passenden Interrupteingang für den Interrupt 0, der im Arduino-Sketch verwendet wird, gelegt werden. In dem hier verwendeten Board mit LoRa-Funkmodul, dem „The Things Uno“ von TTN ist das der Pin 3, da es sich bei diesem Board um eine „Arduino Leonardo“-Architektur handelt (Atmel 32U4 basiert). In der Abbildung ist zu beachten, dass der Stecker auf der Rückseite sitzt, das Board aber von der Top Seite dargestellt ist (Pin 1 oben links am Stecker).

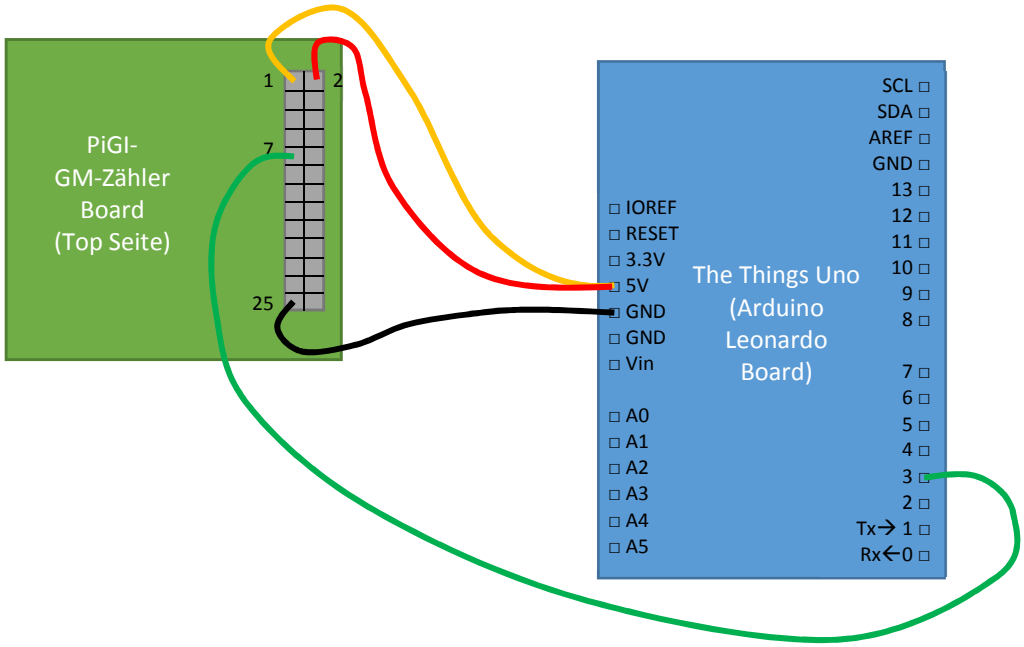


Abb. 4: Anschluss des PiGI-GM-Zählermoduls an den Arduino-Leonardo (The Things Uno).

Wenn man nur eine minimale Geiger-Müller Zählrohr Anwendung auf dem Arduino laufen lassen möchte, dann müsste man an das PiGI-Board lediglich noch ein Zählrohr anschließen (Anode an HV+, Kathode an HV- und könnte den folgenden einfachen Sketch verwenden:

```
#define MAXCNT 100
#define CalFactor 1

volatile int counter = 0;
unsigned long oldTime = 0;

void setup()
{
  Serial.begin(9600);
  attachInterrupt(0, count, FALLING);
}

void loop()
{
  unsigned long time;
  unsigned long dt;
  float rate;

  if (counter == MAXCNT) {
    detachInterrupt(0);
    time = millis();
    dt = time-oldTime;
    rate = (float)MAXCNT*60.0*1000.0/(float)dt/CalFactor;
    Serial.println(round(rate));
    delay(100);

    oldTime = millis();
    counter = 0;
    attachInterrupt(0, count, FALLING);
  }
}

void count()
{
  counter++;
}
```

Dieser Sketch gibt lediglich die aus 100 gezählten Impulsen berechnete Zählrate auf dem seriellen Monitor aus. Dazu ordnet er dem Zählimpulseingang die fallende Flanke des Interrupt 0 zu und definiert, dass falls ein Zählimpuls eintrifft, die Interrupt-Handler-Routine count() ausgeführt wird. Diese zählt einfach die Pulse in der Variablen counter hoch. Ansonsten dreht sich das Programm in der Funktion loop() endlos. Wenn die maximale Zahl Pulse, die für die Berechnung der Zählrate verwendet wird (MAXCNT) erreicht ist, findet eine Zählratenberechnung statt. Dazu wird die vergangene Zeit mit millis() abgefragt und die Differenz zur letzten Zeitabfrage gebildet. Die Rate ergibt sich dann durch Division von MAXCNT durch die Zeitdifferenz unter Berücksichtigung der Zeiteinheit (millis liefert die Zeit in Millisekunden). Als Ausgabe entsteht dann die Zählrate in Counts per Minute (cpm).

Soll die Applikation nun die Zählrate über das LoRaWAN in ein Messnetz übertragen, dann ist die Verwendung des „The Things Uno“ die einfachste Lösung. Dieses Board enthält ein LoRaWAN Funkmodul und es wird eine Bibliothek bereitgestellt, mit der die Daten direkt über LoRaWAN in das freie IoT Netz des Providers „The Things Network“ (TTN) übertragen werden können. Wie man dazu vorgehen muss, ist in /1/ ausführlich beschrieben.

Der Sketch von oben muss dazu wie folgt erweitert werden:

```
#include <TheThingsNetwork.h>
#include <CayenneLPP.h>
#define MAXCNT 100
#define CalFactor 1


// Set your AppEUI and AppKey
const char *appEui = "<App EUI eintragen>";
const char *appKey = "<App Key eintragen>";

#define loraSerial Serial1
#define debugSerial Serial

// Replace REPLACE_ME with TTN_FP_EU868 or TTN_FP_US915
#define freqPlan TTN_FP_EU868

TheThingsNetwork ttn(loraSerial, debugSerial, freqPlan);
CayenneLPP lpp(51);

volatile int counter = 0;
unsigned long oldTime = 0;

void setup()
{
  loraSerial.begin(57600);
  debugSerial.begin(9600);

  // Wait a maximum of 10s for Serial Monitor
  while (!debugSerial && millis() < 10000)
    ;

  debugSerial.println("-- STATUS");
  ttn.showStatus();

  debugSerial.println("-- JOIN");
  ttn.join(appEui, appKey);

  attachInterrupt(0, count, FALLING);
}

void loop()
{
  unsigned long time;
  unsigned long dt;
  float rate;

  if (counter == MAXCNT) {
    detachInterrupt(0);
    time = millis();
    dt = time-oldTime;
    rate = (float)MAXCNT*60.0*1000.0/(float)dt/CalFactor;
```

```
    debugSerial.println(round(rate));

    lpp.reset();
    lpp.addLuminosity(1, rate);
    ttn.sendBytes(lpp.getBuffer(), lpp.getSize());
    delay(60000);

    oldTime = millis();
    counter = 0;
    attachInterrupt(0, count, FALLING);
  }
}

void count()
{
  counter++;
}
```

Dieser Sketch ist nun mit Aufrufen der „The Things Network“-Bibliothek und der Cayenne-Bibliothek erweitert. Außerdem setzt er eine Software-Serial Schnittstelle unter loraSerial auf (Serial1 beim Leonardo). Der normale Serial Monitor (Serial) wird als debugSerial erklärt. Sobald die Zählrate berechnet und auf dem Serial Monitor ausgegeben ist, wird zunächst die Formatierung der Zählrate für die webbasierte Visualisierungsplattform Cayenne vorgenommen, an die der TTN-Gateway die Daten weiterreicht. Dazu dient die Cayenne lpp Methode. Da lpp bisher keine Zählraten und Geiger-Müller Zählrohre kennt, wird hierzu der Helligkeit (Luminosity) als Größe in Cayenne missbraucht. Später, bei der Einrichtung des Cayenne Dashboards kann das wieder mit Count Rate überschrieben werden. Die Methode ttn() mit der Funktion sendBytes schickt die Zählrate schließlich über das LoRaWAN an das TTN Backend, das, wenn die entsprechende Cayenne Integration im User Bereich aufgesetzt ist, die Daten dann an mydevices.com weiterreicht, wo die Daten im Cayenne Dashboard des Anwenders entsprechend visualisiert werden (für die detaillierte Konfiguration siehe /1/).

Abb. 5 zeigt einen solchen Aufbau mit dem PiGI-Board und dem The Things Uno. An das PiGI-Board wurde ein Zählrohr des Typs FZH 76V von Frieseke & Höpfner angeschlossen. Es hat gegenüber einem russischen SBM-20 eine deutlich niedrigere Zählrate, da es deutlich kleiner ist (etwa 80cpm pro µSv/h U nat.). Die Tasse mit Uranglasur bringt es aber auf etwa 800cpm bei diesem Zählrohr. Abb. 6 zeigt das zugehörige Cayenne Dashboard, wo zunächst die Tasse und danach die Nullrate zu sehen ist.

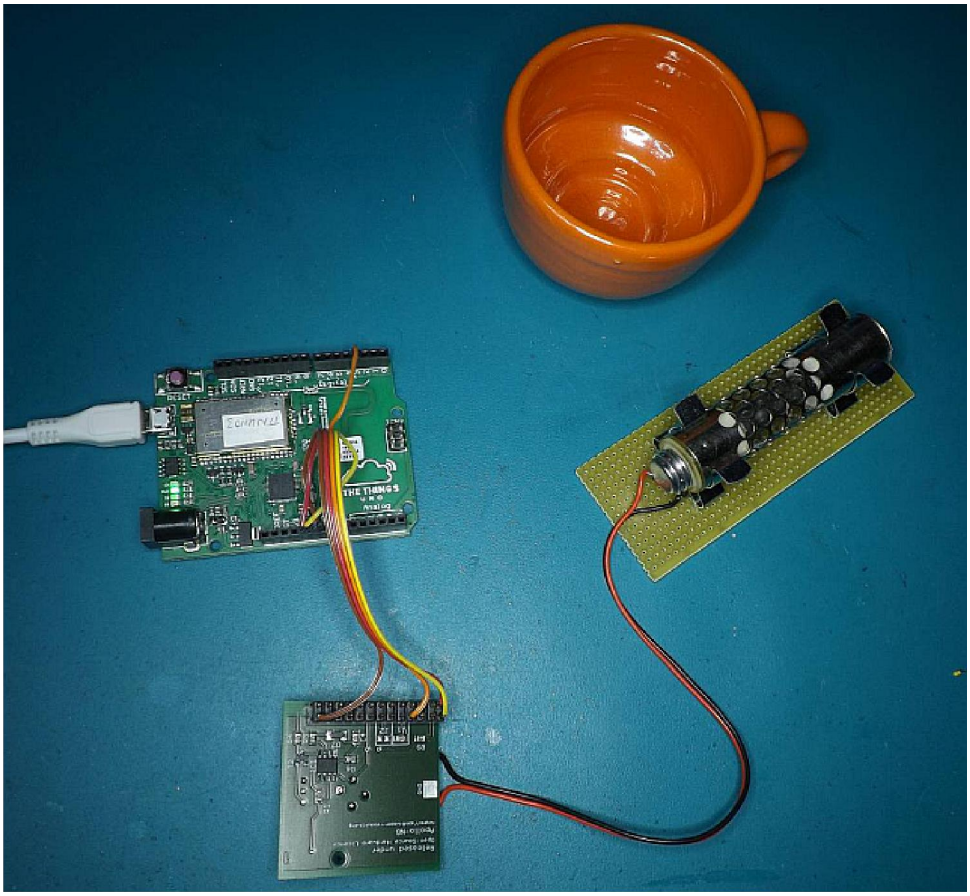

Abb. 5: Ein kompletter Beispiel-Aufbau eines LoRaWAN-TTN Messknotens für Radioaktivität mit dem PiGI Board am Arduino (The Things Uno), die Tasse mit Uranglasur dient als Prüfstrahler


Abb. 6: Zugehörige Messung visualisiert in Cayenne, zu Beginn die Tasse, danach die Nullrate

**Links und Literatur**


/1/ Ein Umwelt-Radioaktivitäts-Messknoten für das LoRaWAN IOT-Netzwerk „The Things Network“

http://opengeiger.de/LoRaGeigerTTN.pdf

/2/ IsiGeiger: Ein Geiger-Müller Zählrohr-Modul für Mikrocontroller-Anwendungen

http://opengeiger.de/IsiGeigerDoku.pdf

/3/ Universelle Bestimmung der Zählimpulsrate von Sensoren für Radioaktivität mit dem Arduino

http://www.opengeiger.de/UnivZaehlerArduino.pdf

/4/ TDRM Messnetz (Tihange Doel Radiation Monitoring)

https://tdrm.fiff.de/index.php

/5/ Apollo-NG PiGI-Projekt

https://apollo.open-resource.org/lab:pigi

/6/ The Things Uno Datasheet

https://www.thethingsnetwork.org/docs/devices/uno/

/7/ Kaliumchlorid als Kochsalzersatz und als Prüfstrahler

https://de.wikipedia.org/wiki/Kaliumchlorid

/8/ Bezugsquelle für den „The Things Uno“

Geschäftskunden über Farnell.de

https://de.farnell.com/the-things-network/ttn-un-868/the-things-uno-eu/dp/2675815?st=The%20Things%20Uno

Privatkunden über z.B. develektro

https://www.develektro.com/THE-THINGS-NETWORK-TTN-UN-868-THE-THINGS-UNO-EU

/9/ Bezugsquelle für das fertige PiGI-Modul eMail: PeterKa@TreeDev.eu