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

Bernd Laquai, 28.10.2018

Das Internet der Dinge (Internet-of-Things, IoT) ist derzeit in aller Munde, und in der Tat findet derzeit eine rasante Entwicklung in der Elektronik und im Netzaufbau statt um es in der Praxis auch tatsächlich anwendbar zu machen. Man kann auch sagen, dass der Wettbewerb der Dienste-Anbieter schon in vollem Gange ist. IoT-Netzwerke und Dienste, welche auf der neuen Technologie aufbauen, schießen aus dem Boden wie Pilze, so dass man kaum noch einen Überblick mehr hat, wo die Unterschiede liegen. Ein Merkmal ist jedoch allen diesen Netzwerken gemeinsam: Es sind meist Funknetzwerke für die Datenübertragung, die mit sehr niedriger Leistung auf der Teilnehmerseite arbeiten und nur geringe Daten-Bandbreiten zur Verfügung stellen (Bytes pro Minute), dafür aber Reichweiten im Kilometerbereich erreichen. Architektonisch sind sie so aufgebaut, dass viele Sensorknoten ihre Daten zu vergleichsweise wenigen Gateways (ähnlich einem WLAN-Router) funken, welche die empfangenen Daten ins Internet an die Server des IoT-Netzwerk-Providers weiterlenken. Im Netzwerk-Backend des Providers werden mehrfach empfangene Pakete aussortiert und geordnet. Danach werden die Daten je nach Wunsch des Sensorknoten-Betreibers entweder an „normale“ Internet-Server weitergereicht oder an weiterverarbeitende Dienste z.B. zur Darstellung auf Webseiten oder zum Versand von Benachrichtigungen (z.B. per E-Mail) übergeben.

Die Tatsache, dass diese neuen Funk-Technologien mit sehr niedrigem Leistungsverbrauch hohe Reichweiten erreicht, ermöglicht vor allem den Aufbau von dezentralen Netzen mit vielen kleinen IoT-Netzknoten auch an Stellen in der Umwelt, wo ein WLAN Router nicht so ohne weiteres hinreicht. Von daher sind diese Funknetze vor allem für eine datensparsame Umweltüberwachung geeignet, wo flächige verteilte Sensornetzwerke ihre Messdaten autonom erfassen und komprimiert über ihre Funkanbindung durch das Internet in eine Datenbank zur Visualisierung und zur weiteren Nachbearbeitung übertragen können.

Hier soll diese neue IoT-Technologie am Beispiel eines Sensorknotens für die Erfassung der Umwelt-Radioaktivität und der Datenübertragung über das LoRaWAN (Long Range Wide Area Network) zum „The Things Network (TTN)“ Netzwerk-Provider dargestellt werden. Zur Visualisierung und Auswertung der Daten soll die IoT-Plattform „Cayenne“ von der Firma Mydevices verwendet werden. Sowohl die Dienste des TTN, wie auch in gewissem Umfang die Dienste von Mydevices sind kostenfrei, sofern die Anwendung keinen kommerziellen Hintergrund hat. Im Gegensatz dazu sind die übrigen bereits nutzbaren IoT-Netzwerke mit nennenswerter Verbreitung in Europa, wie beispielsweise Sigfox oder LTE IoT-NB (Telekom), ähnlich wie der normale Mobilfunk, immer mit gewissen Kosten verbunden.

Der hier vorgestellte LoRaGeiger Netzwerkknoten besteht aus einem Geigerzählermodul SBM20-Driver von 4N-Galaxy mit digitalem Zählimpulsausgang und einem „The Things Uno“ von TTN. Der The Things Uno ist im Prinzip ein Arduino-Leonardo Derivat mit einem RN2483 LoRa Funk-Modul vom Hersteller Microchip. Das LoRa Funk-Modul selbst besteht aus einem Chipsatz, der einen Microcontroller enthält, welcher den LoRaWan Protokoll-Stack abwickelt und den eigentlichen Funk-Transceiver-Chip, welcher das von der Firma Semtech patentierte Chirp-Modulationsverfahren für den LoRa Standard implementiert. An dieser Stelle ist wichtig zu erwähnen, dass der im RN2483 LoRa Funkmodul enthaltene Mikrocontroller die Kommunikation zum Arduino hin über eine UART abwickelt, welche eine serielle Schnittstelle im RS232 Stil implementiert. Im Gegensatz dazu verwendet der neue Arduino MKRWAN 1300, der ein LoRa-Funk-Modul des Herstellers Murata enthält, die SPI Schnittstelle zur Kommunikation mit dem SAMD-Controller auf dem Main Board, was

zu gewissen Inkompatibilitäten mit der derzeitigen Bibliothek von TTN führt. Bei der Auswahl des Main Boards ist also etwas Vorsicht geboten, von daher ist man mit dem The Things Uno, der vom TTN selbst entwickelt wurde, eher auf der sicheren Seite, auch wenn das Design nicht mehr so ganz das modernste ist.

Bevor man sich jedoch entscheidet, einen LoRaWAN Netzknoten zu entwickeln, empfiehlt es sich die Netzwerkabdeckung in der Gegend, in welcher der Knoten aufgestellt werden, soll zu prüfen. Da die LoRaWAN Gateways in der Regel von ambitionierten Bastlern und Sponsoren der Open Source Community oder von Hochschulen aufgestellt werden und der Netzwerkaufbau erst begonnen hat, deckt das Netzwerk zumeist nur die größeren oder besonders innovativen Städte ab. Die beste Möglichkeit der Prüfung bietet das TTN-Mapper Portal https://ttnmapper.org/. Hier werden die von Freiwilligen erfassten Daten zur Netzabdeckung dargestellt, welche die Empfangssignalstärke mit Hilfe von mobilen IoT-Netzknoten vermessen. Die auf dem Portal dargestellten Graphen zeigen mit farblich codierten Strahlen an, wie gut das Signal der Netzknoten an unterschiedlichen Orten von den verschiedenen Gateways empfangen wurde.

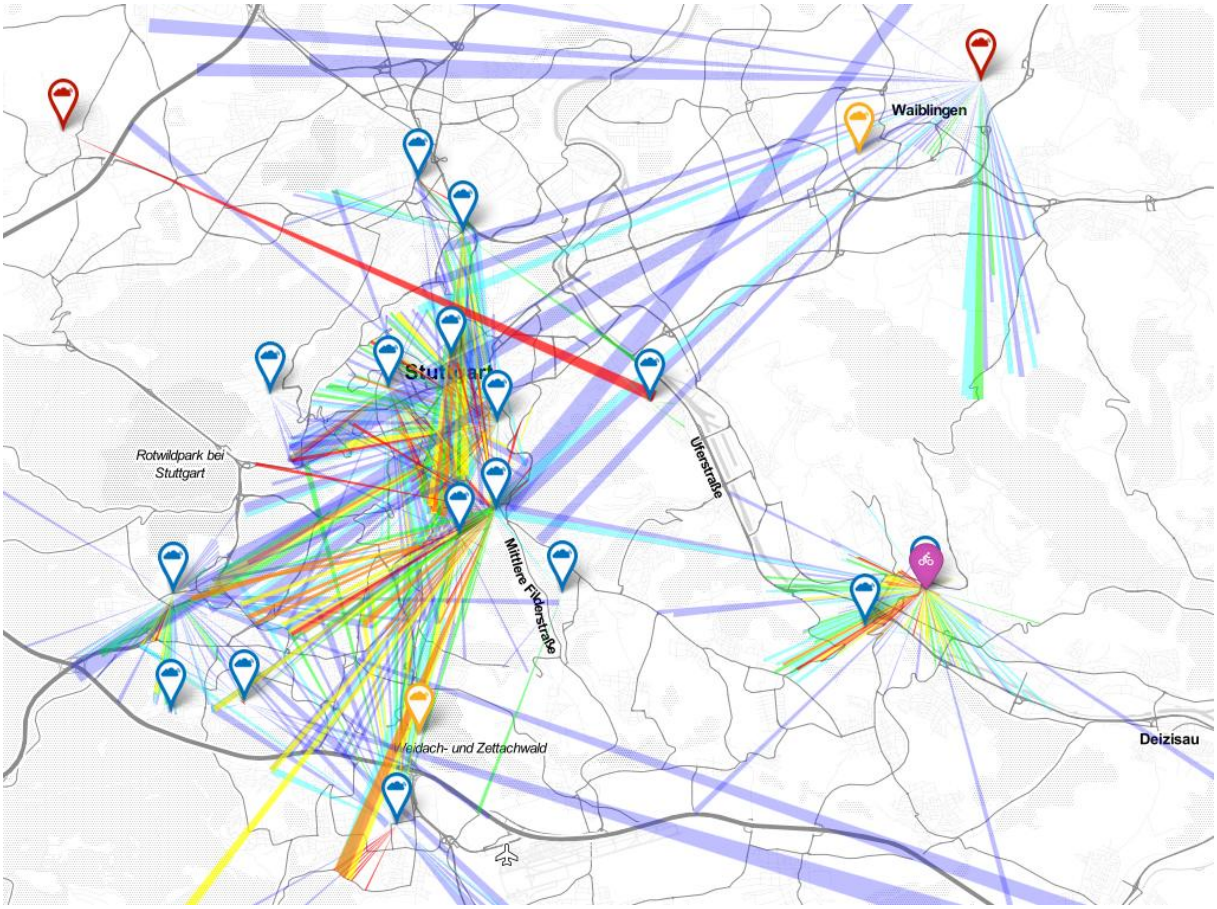

Abb. 1: Der derzeitige Status der TTN-LoRaWAN Netzabdeckung in der Gegend um Stuttgart

Für das Mapping wird in der Regel ein Mobiltelefon benutzt, auf welcher die ttmapper App läuft. Die App verwendet die Standortdaten des Telefons und übermittelt diese zusammen mit der Kennung für den mobilen IoT-Netzknoten an einen Server des ttnmapper Portals. Gleichzeitig sendet der IoT-Netzknoten eine beliebige Nachricht an das TTN Netzwerk, welches die empfangenden Gateways und deren Position ebenfalls an das ttnmapper Portal weitergibt. Nun geht man davon aus, dass sich der mobile IoT-Netzknoten und das Mobiltelefon an ein und derselben Stelle befinden und kann so die Empfangssignalstärke am jeweiligen Gateway dem Ort des Mobiltelefons zuordnen.

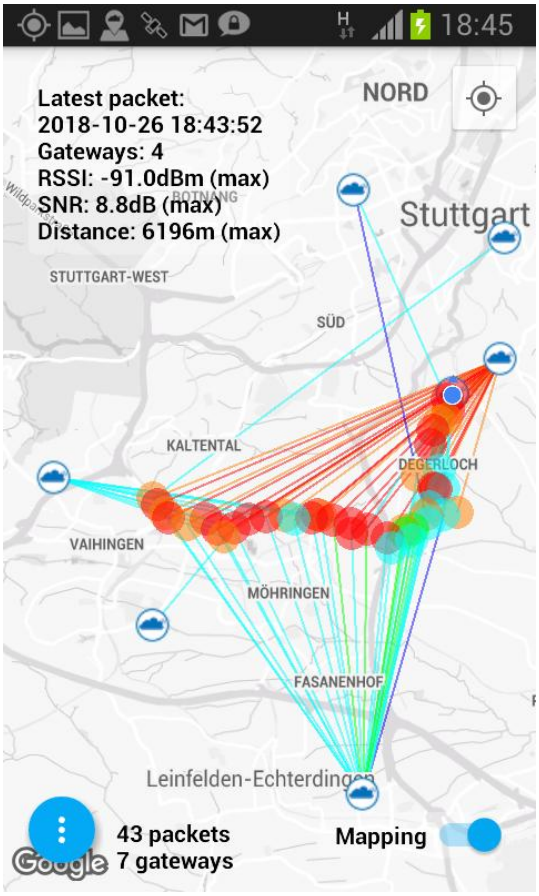

Abb. 2: Mapping Ergebnis eines mobilen TTN LoRaWAN Netzknotens im Süden Stuttgarts. Die blauen Wolken stellen die Position der Gateways und deren Empfangssignalstärke in farbigen Strahlen dar.

Der Aufbau des LoRaGeiger IoT-Netzknotens kann nun ganz einfach durch Ankopplung eines Geigerzählermoduls an den The Things Uno erfolgen, den man so programmiert, dass er einerseits die digitalen Zählpulse des Geigerzählermoduls auswertet (die Zählrate berechnet) und andrerseits das Zählratenergebnis über das LoRaWAN Modul an das TTN Netzwerk sendet. Innerhalb der TTN Plattform wird der IoT-Netzknoten innerhalb des Benutzerkontos angemeldet und festgelegt, was das TTN Netzwerk-Backend mit den empfangenen Zählraten-Daten tun soll. Prinzipiell können die vom TTN empfangenen Zählratenwerte bereits über ein Browserfenster als Zahlen im Hex-Format beobachtet werden. Da es für Umwelt-Radioaktivitätsmessungen, wie auch für andere Daten sinnvoll ist, den zeitlichen Verlauf des gemessenen Sensorsignals anzuzeigen, empfiehlt es sich diese noch einmal an einen weiteren Internetbasierten Dienst weiterzuschicken, der die Aufgabe der grafischen Darstellung im Browserfenster übernimmt. Alternativ dazu kann man die Daten natürlich auch an einen eigenen Webserver weiterschicken, auf dem ein Skript (z.B. ein in PHP geschriebenes) die Daten entgegennimmt und in einer eigenen Datenbank einträgt (z.B. mittels SQL). Mit einem weiteren Skript könnten diese Daten dann ebenfalls, auf Anfrage eines Browsers hin, in eine Grafik gezeichnet oder textuell ausgegeben werden. Die serverseitige Programmierung und das Bereithalten der Datenbank werden einem aber vollständig abgenommen, wenn man den Weg über einen Dienst wie Cayenne wählt. Dafür ist aber die verfügbare Flexibilität bei der Darstellung etwas stärker eingeschränkt.

Die Firma Mydevices (https://mydevices.com/) bietet die Visualisierung von Daten aus IoT-Netzen mit dem Cayenne-Tool als Dienstleistung an, die für private Kleinanwendungen bisher noch kostenlos und zeitlich nicht beschränkt ist. Gleichzeitig hat auch der TTN-Netzprovider eine Schnittstelle zu der

Visualisierungsoberfläche Cayenne als eine mögliche Variante der Datenweitergabe (die sogenannte Integration) implementiert. Wählt man diese im Benutzerbereich für eine Klasse von Netzknoten (Application) aus, und hat man im Netzknoten die entsprechenden Aufrufe an die Cayenne Bibliotheks-Schnittstelle (API) eingebaut, dann erzeugt dies nach einer einfachen Konfiguration der Oberfläche gleich die gewünschte Grafik an. Außerdem bietet Cayenne noch den Export der Daten nach Excel an. Schließlich kann man noch Trigger-Punkte setzen, so dass Cayenne automatisch, z.B. nach Überschreiten eines Schwellwerts, eine E-Mail zur Warnung verschickt, solange das Internet zu dem Zeitpunkt noch ordnungsgemäß funktioniert (was zumindest bei einem Super-Gau in einem benachbarten Kernkraftwerk durchaus angezweifelt werden sollte).

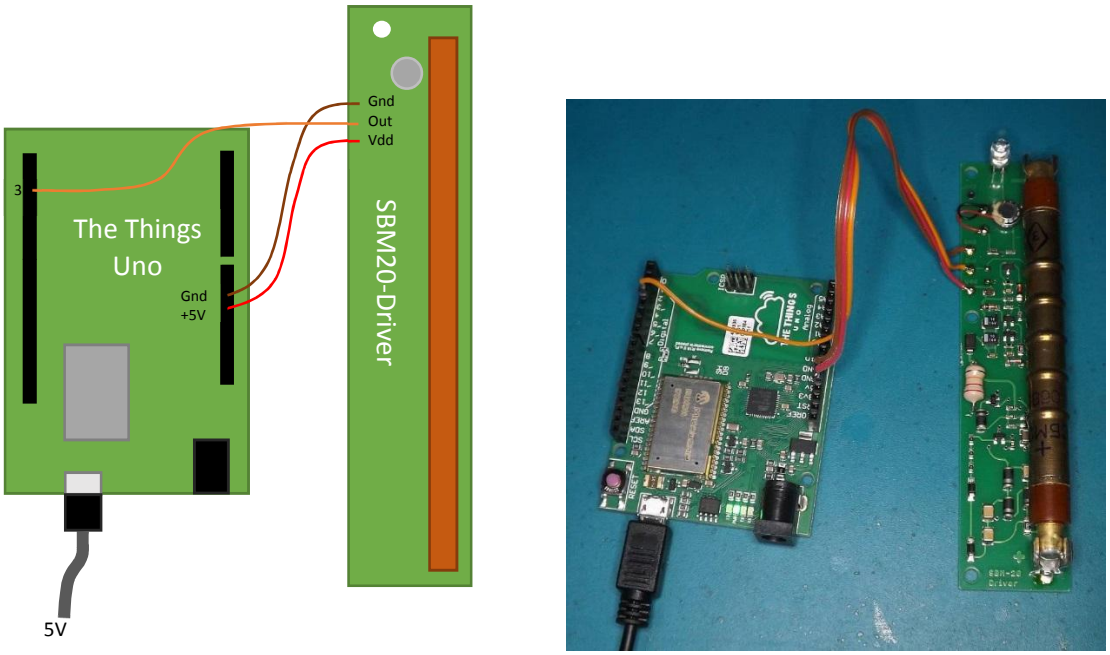

Abb. 3: Schematischer und realer Aufbau des LoRaGeiger Netzknotzens für das TTN

Der Aufbau der LoRaGeiger Netzknoten-Hardware ist gegen den Aufwand des Herstellens der Kommunikationswege im Netzwerk ziemlich trivial. Verwendet man ein Geigerzählermodul wie den SBM20-Driver von 4N-Galaxy, braucht man zum Betrieb lediglich eine Spannungsversorgung von 5V, die man am The Things Uno an der Arduino Uno kompatiblen Stiftleiste abgreifen kann. Das The Things Uno Main-Board lässt sich entweder über die Spannungsversorgungsbuchse (7-12V) und den eingebauten Linearregler versorgen oder aber man speist eine geregelte Spannung von 5V über den USB-Port von einem 5V Stecker-Netzteil oder einer anderen Stromversorgungseinheit ein (z.B. einer kleinen autonomen Solaranlage). Der digitale Zählimpulsausgang des Geigerzähler-Moduls muss zusätzlich noch mit einem digitalen Pin des The Things Uno verbunden werden, welcher einem Interrupt Eingang zugeordnet ist, welcher die Zählpulse registriert.

An dieser Stelle ist Achtsamkeit geboten. Der The Things Uno von TTN ist eigentlich weniger zum Arduino Uno kompatibel (atmega 328p mit separatem USB Chip) als viel mehr zum Arduino Leonardo, welcher auf dem Atmel Mega 32U4 Controller (mit eingebauter USB Schnittstelle) basiert. Damit sind die Interrupt-Eingänge auf andere Pins verteilt als beim Arduino Uno. In diesem Beispiel wird der Interrupt 0 verwendet, welcher beim The Things Uno wie auch beim Arduino Leonardo auf dem digitalen Pin 3 der Stiftleiste zu liegen kommt. Mit diesem Pin ist der Zählimpulsausgang des SBM20-Driver zu verbinden.

Der SMB20-Driver enthält die erforderliche Hochspannungserzeugung für das russische SBM20 Geiger-Müller Zählrohr und den Zählimpulsverstärker, welcher die Zählimpulse in kurze digitale Pulse

umformt. Ein großer Vorteil des SBM20-Driver Moduls ist, dass es auch bei höheren Zählraten nur einen sehr geringen Stromverbrauch aufweist (ca. 2mA). Das SBM20 Zählrohr ist für Radioaktivitätsmessungen in der Umwelt gut geeignet. Es liefert bei normaler natürlicher Strahlung des Untergrunds und bei normaler kosmischer Strahlung etwa 30 Zählpulse pro Minute. Da man aus rein statistischen Gründen 100 Zählpulse benötigt um die Zählratenstreuung bei 10% zu halten (Poisson-Statistik), folgt daraus, dass man unter Normalbedingungen für das SBM20 Zählrohr eine Messdauer pro Messwert von etwa 3 Minuten einkalkulieren sollte. Dieses Messintervall passt auch gut zu der „Fair Access Policy“ des TTN, welches festlegt, in welchem Abstand wie viele Bytes gesendet werden dürfen um das Netzwerk nicht zu überlasten. Wie bei ähnlichen Anwendungen bereits diskutiert, empfiehlt es sich aus statistischen Gründen die Anzahl von 100 Pulsen als konstant vorzugeben und dafür die Zeit zu messen um die Zählrate zu bestimmen (und nicht umgekehrt die Messzeit vorzugeben). Damit sinkt dann auch die Messdauer bei höheren Radioaktivitätswerten. Aus diesem Grund ist es zusätzlich sinnvoll eine Mindestwartezeit von 1 Minute vorzusehen, bevor ein neuer Messwert über das LoRaWAN Netzwert gesendet wird um die „Fair Access Policy“ des TTN auch bei einem „Störfall“ noch sicher nicht zu verletzen.

Man muss sich allerdings klarmachen, dass bei der Überwachung der Umweltradioaktivität, anders als zum Beispiel bei Feinstaubmessungen, die Radioaktivitätswerte normalerweise nicht schwanken. Die einzige Schwankung, die man unter Normalbedingungen in Messungen wahrnehmen kann, und die man auch zur kontinuierlichen Funktionsüberprüfung nutzen sollte, ist die, welche durch die Deposition von Radonzerfallsprodukten mit schlagartig eintretendem starkem Regen entsteht. Die dabei auf dem Boden abgelagerten Radionuklide mit kurzer Halbwertszeit lassen die messbare Umweltradioaktivität um einen Faktor 2-3 zu Beginn des eintretenden Niederschlags steigen. Daher sollte das System so bemessen sein, dass es diese Regenpeaks sicher nachweisen kann. Bei einer Impulsvorgabe von 100 Pulsen und der dazugehörigen statistischen Streuung von 10% kann man davon ausgehen, dass das SBM20-Driver Modul bei Messzeiten größer 3 Minuten eine ausreichend niedrige Nachweisgrenze für die Regenpeaks zur laufenden Funktionskontrolle erreicht. Dies setzt allerdings noch voraus, dass das der Radioaktivitäts-Messknoten etwa 1m über dem Boden (am besten Grasboden) in einem für Beta- und Gammastrahlung transparenten Gehäuse aufgestellt wird (z.B. in einem kurzen Kunststoff-Drainage-Rohr aus dem Baumarkt, beidseitig verschlossen mit Blindstopfen).

Neben der Verbindung des Geigerzählermoduls mit dem The Things Uno und der Stromversorgung ist keine weitere Verbindung mehr nötig. Das The Things Uno Main-Board hat eine auf die Platine aufgedruckte Antenne, die in den meisten Fällen eine ausreichende Sendeleistung garantiert. Allerdings kann die Platine mit einer Bestückungsänderung auch mit einer externen Antenne betrieben werden, für den Fall, dass außergewöhnlich Reichweiten erforderlich sind, oder die Verbindung zu einem Gateway stark durch umliegende Bebauung oder hohe Geländehindernisse beeinträchtigt ist. Dazu muss am Antennenanschluss des LoRa-Funkmoduls ein Jumperwiderstand ausgelötet und dafür eine uFL-Antennenbuchse aufgelötet werden, die dann mit dem Antennenkabel einer 868MHz Antenne verbunden wird.

Sobald die alle Verbindungen hergestellt sind, muss der The Things Uno noch in der Arduino Entwicklungsumgebung programmiert werden. Das dafür erforderliche Programm (Sketch) im Anhang kann leicht aus dem bereits besprochenen generischen Programm für die Verwendung des SBM20-Driver Moduls mit einem Arduino abgeleitet werden. Was speziell noch hinzugefügt werden muss, sind die Anweisungen zum Einbinden der Bibliothek für das The Things Network (TheThingsNetwork.h) und für die Cayenne Low Power Payload (LPP) API zur Visualisierung der Daten (CayenneLPP.h). Des Weiteren müssen zwei serielle Schnittstellen definiert werden, zum einen debugSerial und zum zweiten loraSerial. Die Schnittstelle debugSerial ist identisch mit der Serial Schnittstelle in der Arduino Umgebung und dient der Ausgabe von LoRaWAN Debugging Informationen in der Arduino

Entwicklungsumgebung. Die Definition von debugSerial als Serial ermöglicht der TheThingsNetwork Bibliothek Statusmeldungen auf dem Serial Monitor auszugeben. Die Definition von loraSerial als Serial1 bedeutet aber, dass dem Objekt ttn, welchem die Klasse TheThingsNetwork zugewiesen wird, mit dem Lora Modul über dessen UART kommunizieren kann. Serial1 wird also im Innern der TheThingsNetwork Bibliothek als Software Serial Schnittstelle implementiert. Die Deklaration von Serial1 müsste also bei Verwendung eines anderen LoRa Funkmoduls, das auch über eine UART mit anderen Pins angesprochen werden soll, noch mit einer weiteren Anweisung für eine andere Software Serial Schnittstelle explizit überschrieben werden. Bei Verwendung eines The Things Uno ist dieses Thema jedoch bereits in der Bibliothek erledigt.

Dem Objekt lpp muss noch die Klasse CayenneLLP zugeordnet werden und zwar mit dem Parameter 51, was die maximale Anzahl übertragbaren Bytes darstellt. Im Setup werden die beiden Seriellen Schnittstellen initialisiert und danach wird das TTN Netzwerkprotokoll angewiesen dem Netzwerk beizutreten (join mit „Over the Air Activation“). Schließlich wird noch der Interrupt für das Geigerzähler Modul aktiviert.

In der loop() Schleife wird wie im generischen SBM20-Driver Programm die Zählrate berechnet (hier nach Vorgabe von 100 Pulsen, die gezählt sein müssen), dann erfolgt das Aufsetzen der Cayenne Befehle zur Visualisierung mit lpp.reset() und lpp.addLuminosity(1,rate). Da Cayenne bisher noch keine physikalische Größe für die Radioaktivität kennt, wurde hier stellvertretend die Größe Luminosity in Lux zweckentfremdet, d.h. was in Cayenne dargestellt wird, ist die Zählrate in Counts Per Minute (cpm) oder falls eine Kalibrierung (CalFactor) vorliegt die Ortsdosisleistung in µSv/h. Für die Bestimmung der Zählrate wird bei Auftreten eines Interrupts (fallende Flanke des low-aktiven Zählimpulses) die Interrupt Handler Routine count() ausgeführt, welches den Zähler hochzählt. Sobald dieser die in MAXCNT vorgegebene Zählimpulszahl überschreitet, wird zunächst der Interrupt deaktiviert und dann die Zählrate aus der verstrichenen Zeit berechnet. Nach der Übertragung der Zählrate in das LoRaWAN Netzwerk wird der Zählpuls-Zähler wieder zurückgesetzt und der Interrupt wieder aktiviert.

Das kurze Programm benötigt nur etwa 50% des verfügbaren Speicherplatzes des Controllers auf dem The Things Uno und 40% des dynamischen Speichers. Damit könnten also bei Bedarf noch einige weitere Aktionen vom Controller abgewickelt werden, wie z.B. eine einfache Akku-Laderegelung für eine solare Stromversorgung.

Der Gesamt-Stromverbrauch des SBM20-Driver Moduls und dem The Things Uno Main-Board liegt etwa bei 40mA bei 5V (etwa 200mW) und ist damit noch nicht besonders stromsparend im Sinne heutiger Low-Power Technologien. Allein die helle, grüne Power-On LED wird schon alleine etwa 15mA davon benötigen. Auf der anderen Seite erlaubt die Verfügbarkeit der Cayenne Integration auch eine besonders schnelle Implementierung der Online Radioaktivitäts-Überwachung. Bis diese Bibliothek in vergleichbarer Form auch für die neue stromsparendere Arduino MKR Familie zur Verfügung steht, wird vermutlich noch einige Zeit vergehen. Darüber hinaus ist die weitere Reduktion des Stromverbrauchs nur mit profunden Kenntnissen der verschiedenen Stromsparmöglichkeiten in den unterschiedlichen Sleep-Modi der moderneren Controller möglich, was einer zügigen Implementierung oft auch im Wege steht. Ganz grundsätzlich bestünde aber auch dafür noch Einiges an zusätzlichem Einspar-Potential für den Energieverbrauch.

Bevor nun Daten in das TTN Netzwerk übertragen werden können, muss die Anmeldung bei The Things Network erfolgen. Dazu muss zunächst ein Account eingerichtet werden und eine Application hinzufügen werden. Dazu begibt man sich nach der Anmeldung auf die Seite „Console“. Unter

Applications kann man nun verschiedene Applications hinzufügen. Unter einer Application versteht man eine Anwendung für mehrere Netzwerk-Knoten. Wenn man also eine oder mehrere Radioaktivitäts-Messstationen als Netzwerk-Knoten betreiben will, dann würde man die Application vielleicht mit „Radioaktivitäts-Messnetzwerk“ bezeichnen und als Application Id den Namen „radnet“ vergeben.

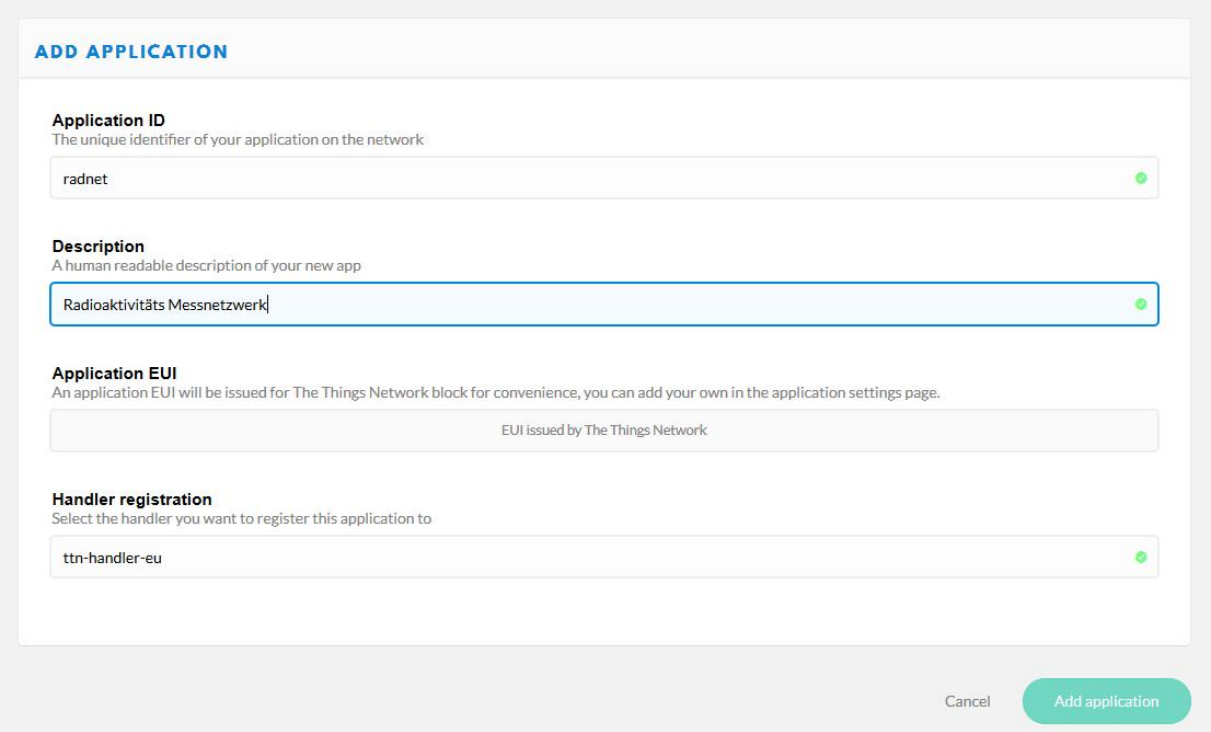

Abb. 4: Registrierung einer Application bei TTN

Die Application EUI ist ein Schlüssel für die Application und wird beim Hinzufügen automatisch vergeben. Danach muss man den Netzwerk-Knoten zunächst probehalber in Betrieb nehmen. In den meisten Fällen liefert das Programm, welches auf dem Controller bei Auslieferung vorinstalliert ist, zu Beginn die sogenannte Device EUI, also den Schlüssel für den Netzwerk-Knoten. Das sieht bei einem neu ausgelieferten The Things Uno im Serial Monitor etwa so aus:

```
Device Information

EUI: 0004A30B001AAAAA
Battery: 3273
AppEUI: 0000000000000000
DevEUI: 0004A30B001AAAAA
Data Rate: 0
RX Delay 1: 1000
RX Delay 2: 2000

Use the EUI to register the device for OTAA
-----------------------------------------
```

Diese Information kopiert man sich am besten in eine Datei und speichert sie unter einem Namen, dem man auf der Platine vermerkt ab. Den Wert unter DevEUI kopiert man in die Zwischenablage und fügt sie unter Device EUI ein, nachdem man im Fenster der radnet Application bei Devices auf „register device“ geklickt hat. Dann vergibt man dem Netzknoten unter Device Id noch einen Namen z.B. „radnode1“ und registriert das Device so.

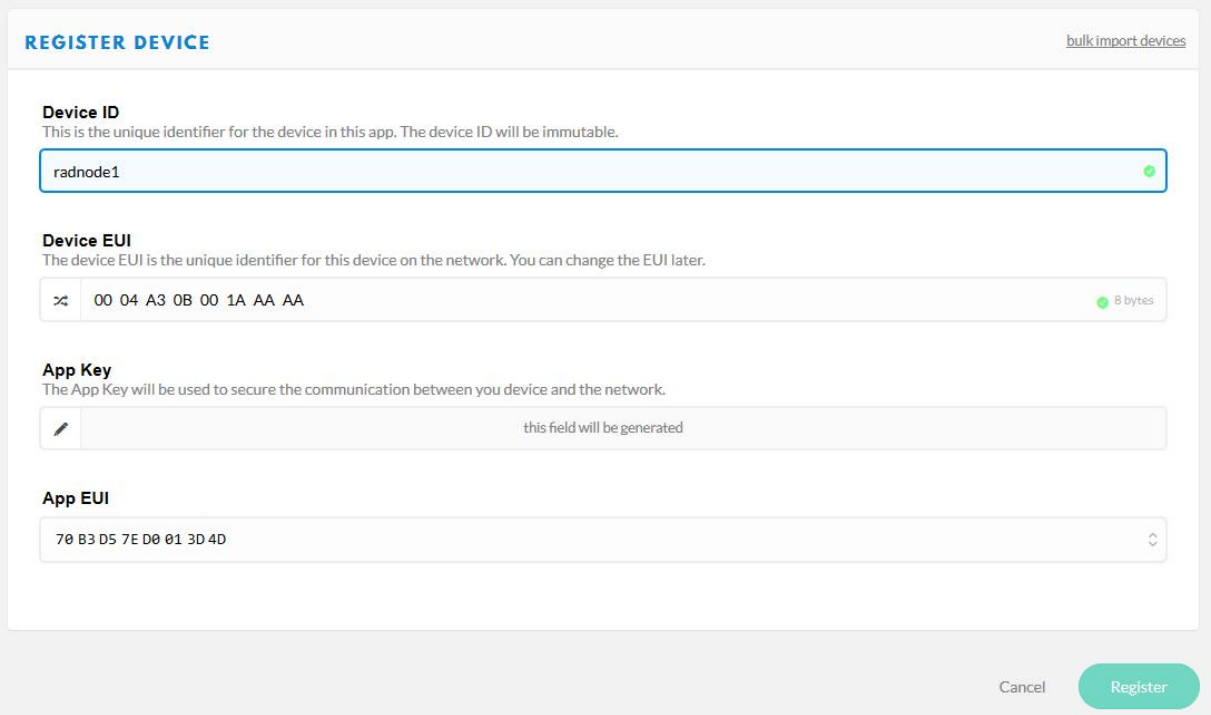

Abb. 5: Registrierung eines LoRaGeiger IoT-Netzknotens bei TTN.

Ganz unten auf der TTN-Seite eines registrierten Device ist ein „Example Code“ Fragment zu finden:

const char *appEui = "70B3D57ED00AAAAA";

const char *appKey = "9E58DF9414E4EDBE2C2E82E12ABAAAAA";

Diesen Code kopiert man in die Zwischenablage und fügt die Schlüssel-Nummern an der Stelle im Arduino Programmcode des Netzknotens ein:

```
// Set your AppEUI and AppKey
const char *appEui = "70B3D57ED00AAAAA";
const char *appKey = "9E58DF9414E4EDBE2C2E82E12ABAAAAA";
```

Dieses Vorgehen ist nötig, damit die versendeten Pakete des Netzknotens an der richtigen Stelle im Netzwerk-Backend herauskommen und die Verschlüsselung der Daten richtig funktioniert.

Nun muss das Arduino Programm mit den richtigen Schlüssel-Nummern versehen auf den The Things Uno hochgeladen werden. Danach beginnt dann der Netzknoten zu arbeiten und man kann die Daten, die empfangen wurden, unter dem Reiter „Data“ im Device Fenster von radnode1 überprüfen. Die Datenbytes werden dabei im hexadezimalen Format als Payload angezeigt.

Bis dahin erreichen die gesendeten Daten also nur das Netzwerk-Backend des TTN-Netzwerks. Eine aussagekräftige Visualisierung der Daten oder ein Export ist aber an dieser Stelle bisher nicht möglich. Um das zu erreichen, kann jetzt zusätzlich eine Integration zur Application hinzugefügt werden, welche sich auf die Daten aller Netzknoten dieser Application auswirkt. Unter dem Menüpunkt Applications->radnet-> Integrations kann Cayenne als Ziel ausgewählt werden. Um genau zu sein, handelt es sich hier um die Anwendung des Cayenne-LPP Protokolls für das Weiterleiten der Daten an den Dienstleister mydevices.com .

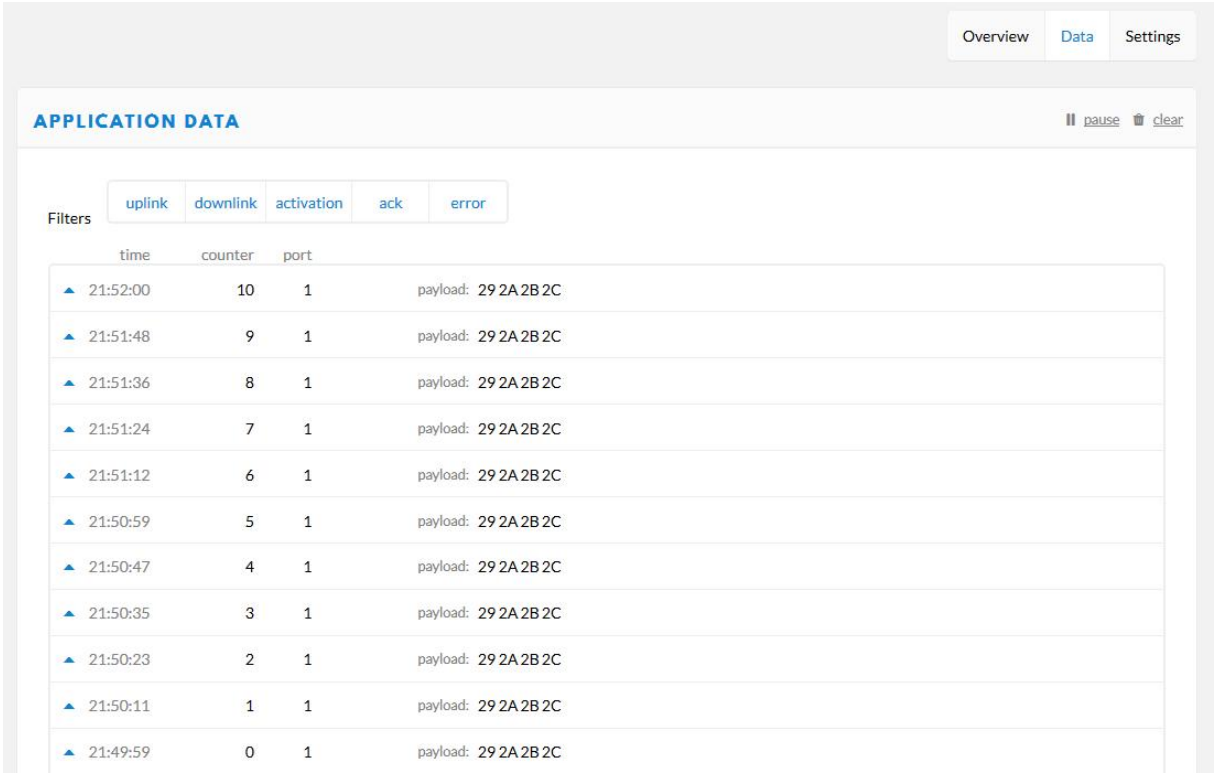

Abb. 6: Beobachtung der vom TTN Netzwerk-Backend empfangenen Daten

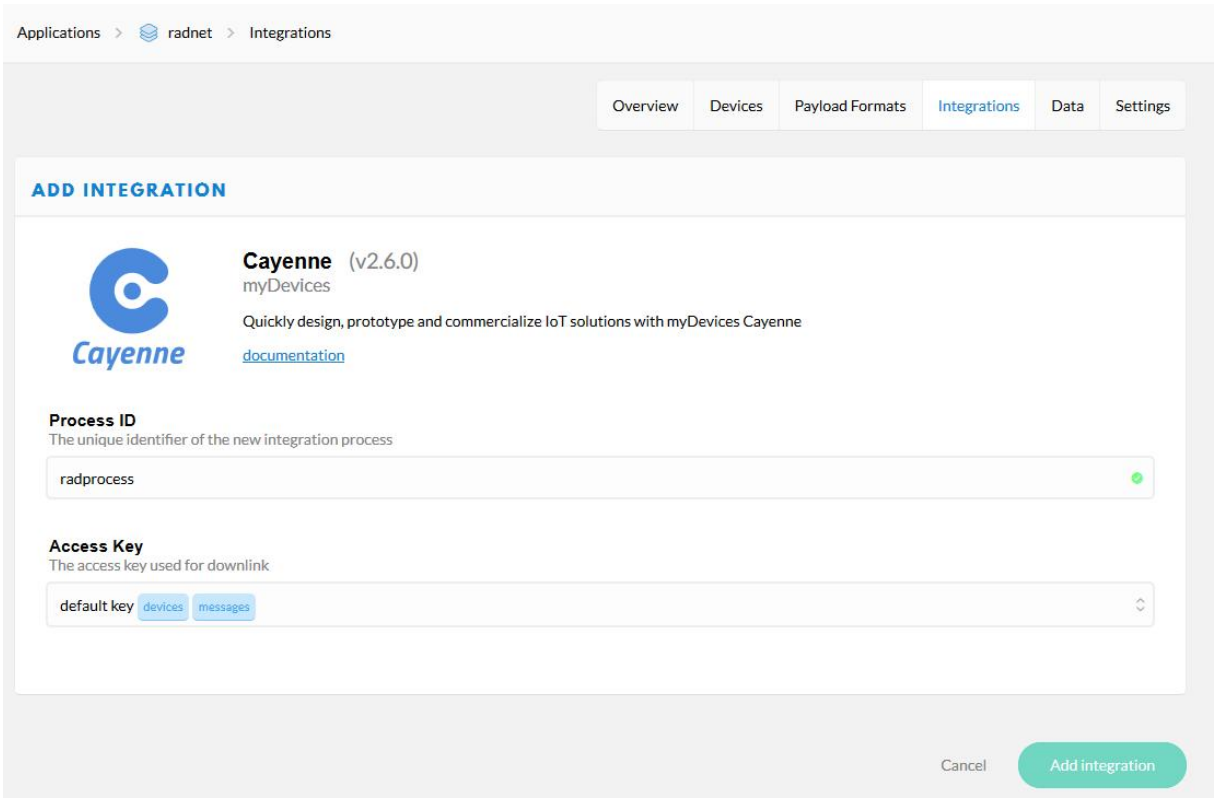

Abb. 7: Hinzufügen einer Cayenne Integration

Als nächstes muss man sich unter mydevices.com einen Account anlegen. Dann fügt man unter „Add new...“ Device/Widget -> Lora -> The Things Network -> The Things Uno den Netzwerk-Knoten hinzu (ganz unten in der langen Liste). In dem Fenster, welches nun am rechten Rand des Browsers aufgeht,

gibt man ebenfalls die Device EUI als Schlüssel Nummer ein. Entscheidend ist hier auch, dass über den Eingabefeldern steht: „This device uses Cayenne LPP“

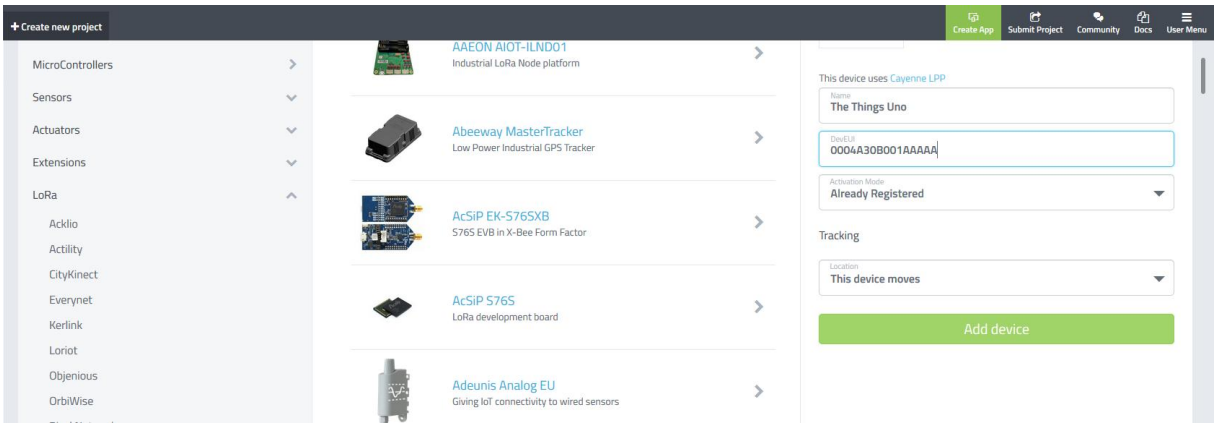

Abb. 8: Auswählen des IoT-Netzknoten Typs bei Cayenne

Drückt man nun „Add Device“, dann entsteht ein Fenster mit der Meldung: „Your dashboard appears when Cayenne receives data from this device. Set up your device to transmit more frequently to speed up the process.” Sobald also der LoRaGeiger IoT-Netzknoten Daten sendet, erscheint die Anzeige für die Zählrate als Luminosity. Klickt man auf das Balkendiagramm-Symbol rechts oben im Feld Luminosity, dann erscheint ein Graph für die Anzeige der Werte in einem einstellbaren Zeitraum. In diesem Fenster kann auch der Export nach Excel angeclickt werden. Cayenne ermöglich darüber hinaus noch etliche weitere nette Optionen für die Visualisierung der Daten.

Als Prüfung der Funktionalität im Niedrigdosisbereich kann man nun etwa 100g Kaliumchlorid (KCl) in der Apotheke als Niedrigdosis-Prüfstrahler kaufen. Da das natürliche Kalium, das auch in vielen Lebensmitteln enthalten ist, etwa 0.012% des radioaktiven Isotops K-40 enthält, beträgt die Aktivität von 100g KCl rein rechnerisch etwa 1635Bq. Zum Vergleich schreibt die Tscherobylverordnung vor, dass 1kg Wildfleisch nicht mehr als 600Bq enthalten dürfen, wenn es zum Verzehr geeignet sein soll. Stellt man nun einen runden Behälter mit 100g Kaliumchlorid neben das Geiger-Müller Zählrohr des LoRaGeiger-Messknotens, dann muss sich eine deutlich sichtbare Erhöhung der Zählrate ergeben, wenn alles richtig funktioniert. Man muss allerdings davon ausgehen, dass von einem runden hohen Behälter mit KCl weniger als ein Viertel der Strahlung in Richtung des Zählrohrs geht und das SBM20-Zählrohr von den Beta- Strahlungsquanten und den Gamma-Strahlungsquanten mit 1452keV Energie des K-40 auch nicht alle erkennt. Deswegen liegt der Unterschied zur Nullrate ohne KCl doch erheblich unter der Rate der Zerfallsakte im Volumen des KCl-Prüfstrahlers von 1600 pro Sekunde. Typischerweise erhöht sich bei diesem Experiment mit 100g KCl die Zählrate nur um etwa 30 cpm, was durchaus vergleichbar zu einem „guten“ Regenpeak ist.

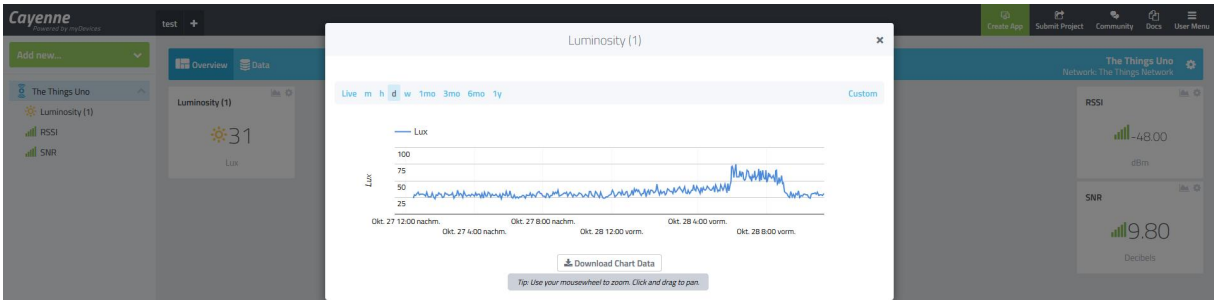

Abb. 9: Anzeige der Zählrate des LoRaGeiger in Cayenne als Luminosity-Graph mit der Einheit Lux, Test der Funktionalität am Ende der Messung mit 100g KCl aus der Apotheke (diätischer Kochsalz-Ersatz)

Anhang

Listing für den The Things Uno:

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

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

#define loraSerial Serial1
#define debugSerial Serial

#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);

  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++;
}
```

Links und Literatur

/1/ Universelle Bestimmung der Zählimpulsrate von Sensoren für Radioaktivität mit dem Arduino

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

/2/ 4N-Galaxy: Geigerzähler, SBM-20 Driver Interface (Prod. Nr. #R02)

http://www.4n-gx.de/R02_de.html

/3/ The Things Uno Datasheet

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

/4/ Kaliumchlorid als Kochsalzersatz und als Prüfstrahler

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