Aufbau und Bewertung eines low-cost Energiemessstellenkonzeptes als LoRa-Netzwerk zur Energiekennzahlenaggregation im Produktionsumfeld

Evaluation of a low-cost energy metering concept implemented as a LoRa Network for energy key performance indicator aggregation in a production environment

Bachelor-Thesis

Autor: Lindsay Philip Stebe

Matrikelnummer: 1525267

Betreuer: Dipl.-Ing Johannes Sossenheimer, M. Sc. | Prof. Dr.-Ing. Matthias Weigold

Abgabe: Darmstadt, 24.11.2021

TECHNISCHE UNIVERSITÄT DARMSTADT

TU DARMSTADT

Bachelorthesis
für
Lindsay Philip Stebe | 1525267

TECHNISCHE UNIVERSITÄT DARMSTADT

Institut für Produktionsmanagement, Technologie und Werkzeugmaschinen

Fachbereich Maschinenbau

Otto-Berndt-Str. 2
64287 Darmstadt

Telefon: +49 6151 16-20080
Telefax: +49 6151 16-20087

Thema:
Aufbau und Bewertung eines low-cost Energiemessstellenkonzepts als LoRa-Netzwerk zur Energiekennzahlenaggregation im Produktionsumfeld

Topic:
Evaluation of a low-cost energy metering concept implemented as a LoRa network for energy key performance indicator aggregation in a production environment

---

Um Energieflüsse in Unternehmen aufzuzeigen, bietet sich das systematisches Energiemanagement an, mit dem u.a. die Energieeffizienz kontinuierlich verbessert werden kann. Die DIN EN ISO 50001 zu Energiemanagementsystemen fordert Unternehmen zu einer kontinuierlichen Messung, Überwachung und Bewertung des Energieverbrauchs auf. Eines der größten Hemmnisse bei der flächendeckenden energetischen Transparenzschaffung ist neben den Kosten für die Sensoren, vor allem das fehlende Wissen wie ein wirtschaftlicher oder energetischer Nutzen aus den gewonnenen Energiedaten generiert werden kann.
In der Arbeit soll mit Mikrokontrollern ein LoRa-basiertes, low-cost Energiemessstellenkonzept für verschiedene Produktionsmaschinen und Infrastrukturanalagen aufgebaut werden. Hierfür sollen bestehende Mikrokontroller LoRa-fähig gemacht werden. Aufbauend hierauf soll eine sinnvolle, teilweise kabellose Netzwerkarchitektur für Energiemessgeräte erarbeitet werden. Die Arbeit baut auf Vorarbeiten auf, in welchen kostengünstige Hardware zur Energiedatenerfassung und optischen Zustandsdatenerfassung auf Basis von Raspberry Pis entwickelt wurde. Die Energie- und Zustandsdaten sollen übertragen und sinnvoll zu Kennzahlen aggregiert werden.
Das Konzept soll an verschiedenen Maschinen und Versorgungsanlagen der ETA-Fabrik erprobt und validiert werden.

In dieser Arbeit sind im Detail folgende Aufgabenteile geplant:

- Recherche zum Stand der Wissenschaft und Technik
  (1) zu möglichen **Zielen** von **Energiedatenerfassung** und **Energiekennzahlbildung**
  (2) zu LoRa-basierten **Energiemessstellenkonzepten** für die Industrie und deren Kosten
  (3) von LoRa- bzw. IP-Netzwerk-basierter Energiedatenerfassung für industrielle Produktionssysteme
- Identifikation von **Anforderungen** an Energiemessstellen mit IP-Netzwerk- bzw. LoRa-basierter Anbindung
- Erprobung und Bewertung unterschiedlicher LoRa-Hardware für Mikrokontroller inkl. der Datenanbindung an übergeordnete Managementplattformen (z.B. EnEffCo / Loriot)
- Sinnvolle Aggregation essenzieller **Messwerte** zu energetischen und produktionstechnischen **Kennzahlen**
- Aufbau eines LoRa-basierten **Energiemessstellennetzwerkes**, der sowohl die RasPI-basierten Vorarbeiten, als auch noch nicht betrachtete Mikrokontroller einbezieht.
- **Implementierung** des low-cost Messstellenkonzepts an mehreren Produktionsmaschinen und Versorgungsanlagen der ETA-Fabrik und Bewertung des zeitlichen und kostentechnischen Aufwands.

Beginn: 19.05.2021
Umfang: 12 CP; 360 h
Betreuer: Dipl.-Ing. Johannes Sossenheimer, M.Sc.

Prof. Dr.-Ing. M. Weigold

Erklärung zur Abschlussarbeit gemäß §22 APB und § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Lindsay Stebe, die vorliegende Bachelor-Thesis gemäß § 22 Abs. 7 APB der TU Darmstadt ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs.2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung gemäß § 23 Abs. 7 APB überein.

Darmstadt, den 22.11.2021

(Ort, Datum) (Unterschrift)

Folgende Einverständniserklärung ist unabhängig vom Prüfungsverfahren zur Bachelorprüfung (ein Exemplar verbleibt bei den Prüfungsakten) und ohne Einfluss auf die Bewertung der Bachelor-Thesis.

Ich bin damit

einverstanden / ~~nicht einverstanden~~,

dass die Bachelor-Thesis in den Bibliotheksbestand der TU Darmstadt aufgenommen wird und öffentlich zugänglich gemacht wird.

Darmstadt, den 22.11.2021

(Ort, Datum) (Unterschrift)

Die TU Darmstadt bittet Sie im Interesse eines freien Informationsaustausches, ihr Urheberrecht an der Arbeit zu wissenschaftlichen Zwecken nutzen zu dürfen. Sie können die Nutzung Ihres Urheberrechts durch die TU Darmstadt ohne Angabe von Gründen und ohne nachteilige Folgen für die Bewertung der Arbeit verweigern.

Ich bin damit

einverstanden / ~~nicht einverstanden~~,

dass die TU Darmstadt das Urheberrecht an meiner Bachelor-Thesis zu wissenschaftlichen Zwecken nutzen kann.

Darmstadt, den 22.11.2021

(Ort, Datum) (Unterschrift)

Kurzfassung & Abstract

Stichwörter: Raspberry Pi, Arduino, JavaScript, Loridane, energy metering, KPI aggregation, LoRa network.

Die vorliegende Arbeit befasst sich mit der Entwicklung, Implementierung und Evaluierung eines drahtlosen Energiemessstellenkonzeptes an Produktionsmaschinen. Im Fokus liegt dabei die Hardwareauswahl, die Implementierung des Software-Codes der verschiedenen Netzwerkinstanzen und die Nutzbarmachung der aggregierten Daten. Dazu gehören das Erarbeiten von Anforderungen an Energiemessstellen, sowie an eine LoRa-Netzwerkstruktur und die bestmögliche Umsetzung unter Berücksichtigung von Kostenoptimierung und Machbarkeit. Die Energiemessstellen werden mit LoRa-fähigen Microcontrollern realisiert, die ihren Nachrichtenstrom an ein, aus derselben Hardware bestehendes, Gateway senden, welches die Daten wiederum über WLAN an einen Raspberry Pi basierten Server weiterleitet. Dafür kommen Microcontroller zum Einsatz, die auf dem Espressif ESP32 Chipsatz und einem Semtech SX1276 LoRa Transceiver basieren. Ergebnis der Implementierung ist die Erweiterung des UDP-basierten LoRa Funks um ein Zeitscheibensystem nach dem „fixed TDMA"-Prinzip, um ein desynchronisiertes Senden der Endgeräte zu ermöglichen. Dies verhindert die zeitliche Überlappung von Funksignalen und lässt so eine Erhöhung der Übertragungswahrscheinlichkeit der Datenpakete zu.

The work on hand concerns the development, the implementation and evaluation of a cordless energy measurement approach for production machines. In focus is selecting adequate hardware as well as implementing software code for individual network instances and the aggregation and utilization of measurement data. This includes acquiring the specifications on energy metering devices in this context and the requirements regarding a LoRa network structure for an optimum implementation accounting cost optimization and feasibility. The energy metering devices are realized with LoRa-capable microcontrollers which transmit data to a gateway based on the identical low-cost one-channel LoRa/microcontroller hardware which then forwards the datasets via MQTT and Wi-Fi to a Raspberry Pi based server. For this purpose, microcontrollers on an ESP32 basis with a Semtech SX1267 LoRa transceiver are employed. The objective of this implementation is the extension of UDP-based LoRa radio with a time disk system that complies with the principles of a fixed TDMA approach but benefits from the efficient implementation of a relative time system without the necessity of dealing with absolute timestamps. This way any device as a part of the network can transmit its information in a predefined timeframe which substantially reduces signal overlapping and therefore packet loss as a principle. Thereby the maximum network capacity and the packet delivery rate increases.

Inhaltsverzeichnis

ERKLÄRUNG ZUR ABSCHLUSSARBEIT GEMÄß §22 APB UND § 23 ABS. 7 APB DER TU DARMSTADT ..... **I**

KURZFASSUNG & ABSTRACT ..... **II**

INHALTSVERZEICHNIS ..... **III**

ABBILDUNGSVERZEICHNIS ..... **VII**

TABELLENVERZEICHNIS ..... **VII**

CODEVERZEICHNIS ..... **IX**

SYMBOLVERZEICHNIS UND GLOSSAR ..... **X**

1 EINLEITUNG ..... **- 1 -**

1.1 Problemstellung ..... **- 1 -**

1.2 Lösungsweg ..... **- 2 -**

1.3 Aufbau der Arbeit ..... **- 2 -**

2 GRUNDLAGEN UND STAND DER TECHNIK ..... **- 3 -**

2.1 Notwendigkeit der Verbrauchsdatenerfassung ..... **- 3 -**

2.2 Technische Grundlagen ..... **- 3 -**

2.2.1 Einordnung LoRa ..... - 3 -

2.2.2 Funktionsweise ..... - 4 -

2.2.3 Chip ..... - 8 -

2.2.4 Chirp ..... - 8 -

2.2.5 Spreading Factor ..... - 9 -

2.2.6 Coding Rate ..... - 10 -

2.2.7 Protokoll ..... - 11 -

2.2.8 Signalinversion ..... - 12 -

2.2.9 Antenne ..... - 12 -

2.3 Marktstandards auf LoRa-Basis ..... **- 17 -**

2.3.1 LoRaWAN ..... - 17 -

2.3.2 TheThingsNetwork und TheThingsStack ..... - 17 -

2.3.3 Aspekte des LoRaWAN-Ansatzes ..... - 20 -

2.4 Energiemesstechnik auf LoRaWAN Basis ..... - 22 -
2.4.1 Vergleich verschiedener Übermittlungstechnologien ..... - 23 -
2.4.2 LoRa-fähige Energiemesssyteme und Erweiterungen ..... - 24 -
2.4.3 MID Zertifikation ..... - 25 -

2.5 Voraussetzungen für die Aggregation von Kennzahlen ..... - 26 -
2.5.1 Generierung von Rohdaten ..... - 26 -
2.5.2 Berechnung von Kennzahlen ..... - 27 -

2.6 Rechtliche Rahmenbedingungen im SRD-Funkband ..... - 28 -

2.7 Zusammenfassung des Forschungsstandes und der Marktsituation ..... - 29 -

3 ANFORDERUNGEN AN EIN LORA-MESSSTELLENSYSTEM ..... - 30 -

3.1 Anforderungen durch die Messstellenumgebung ..... - 30 -

3.2 Konstruktive Anforderungen ..... - 30 -

3.3 Informationstechnische Anforderungen ..... - 31 -

3.4 Anforderungsliste ..... - 32 -

4 KONZEPTION ..... - 33 -

4.1 Netzwerkstruktur ..... - 33 -

4.2 Hardwareauswahl ..... - 34 -

4.3 Funktionelles Konzept ..... - 35 -

4.4 Datenübermittlung ..... - 36 -

4.5 Auswahl der Softwareumgebung ..... - 37 -

5 IMPLEMENTIERUNG LORIDANE – LONG RANGE INDEPENDENT DATA AGGREGATION NETWORK ..... - 38 -

5.1 Verwendete Software ..... - 38 -
5.1.1 Messstellen (Nodes) ..... - 38 -
5.1.2 Gateway ..... - 39 -
5.1.3 Software für den Raspberry Pi-basierten Server ..... - 40 -

5.2 Installation der Server Software ..... - 41 -
5.2.1 Automatisierte Installation ..... - 41 -
5.2.2 Import von Loridane in eine bestehende NodeRed-Instanz ..... - 43 -

5.3 Installation der Firmware auf Heltec Wireless Sticks ..... - 45 -

5.4 Elektrische Beschaltung der Heltec Boards ..... - 46 -
5.4.1 Beschaltung als Gateway ..... - 46 -
5.4.2 Beschaltung als Node ..... - 47 -

5.5 Funktionsbeschreibung der Loridane Software ..... - 48 -
5.5.1 Informationstechnische Netzwerkparameter ..... - 48 -
5.5.2 Signalpfad Uplink ..... - 51 -
5.5.3 Signalpfad Downlink ..... - 52 -
5.5.4 Datenbank ..... - 52 -
5.5.5 Confirmations ..... - 54 -
5.5.6 Acknowledgement ..... - 55 -
5.5.7 Admission ..... - 55 -
5.5.8 Zeitscheibe ..... - 55 -
5.5.9 Verschlüsselung der Nutzdaten ..... - 61 -
5.5.10 Grafische Benutzeroberfläche ..... - 62 -

5.6 Berücksichtigung der rechtlichen Rahmenbedingungen ..... - 70 -

5.7 Maximale Anzahl von Netzwerkteilnehmern ..... - 71 -

5.8 Maximale Paketübermittlungsrate ..... - 72 -

5.9 Modulationsmodi bei Sende- und Empfangsvorgang ..... - 73 -

6 EVALUATION DER IMPLEMENTIERUNG VON LORIDANE ..... - 74 -

6.1 Langzeitversuch ..... - 74 -
6.1.1 Interpretation der Graphen ..... - 75 -

6.2 Evaluation der Netzwerkperformanz und -Zuverlässigkeit ..... - 76 -
6.2.1 Testprinzip ..... - 76 -
6.2.2 Analyse der Daten aus dem Langzeitversuch ..... - 76 -
6.2.3 Analyse der Daten aus dem Belastungstest ..... - 77 -
6.2.4 Versuchsergebnisse ..... - 78 -

6.3 Kostenstruktur ..... - 79 -

6.4 Einsatzgebiete für Loridane ..... - 81 -

7 FAZIT UND AUSBLICK ..... - 82 -

ANHANG ..... A

ANHANG: ELEKTRONISCHE DATEN ..... B

8 QUELLENVERWEIS: PRODUKTLINKS ..... P

9 QUELLENVERWEIS: GENANNTE UND VERWENDETE SOFTWARE ..... Q

10 LITERATURVERZEICHNIS ..... S

Abbildungsverzeichnis

Abbildung 1 CSS-Modulationsschema mit 125kHz Bandbreite ..... - 5 -
Abbildung 2 Gegenüberstellung konventionelle Modulationsverfahren ..... - 6 -
Abbildung 3 LoRa Signalspektrum (izinto GmbH 2020), geringfügig verändert ..... - 7 -
Abbildung 4 Vergleich SF7 und SF9 ..... - 9 -
Abbildung 5 Protokollstruktur LoRa (Semtech Corp. 2015b, S. 29) ..... - 11 -
Abbildung 6 868 MHz Antennen (v.l.n.r., Helix (THB GmbH 2021), Ground-Plane) ..... - 13 -
Abbildung 7 Abstrahlcharakteristik der optimierten Antenne ..... - 14 -
Abbildung 8 S_{11} und VSWR ..... - 15 -
Abbildung 9 A-resonante Schwingung in Resonator und Radial ..... - 15 -
Abbildung 10 Widerstand und Blindwiderstand der Antenne ..... - 16 -
Abbildung 11 Struktur eines LoRaWAN Netzwerkes.jpg (Linnemann et al. 2019, S. 26) ..... - 17 -
Abbildung 12 Übersicht LoRaWAN Geräteklassen (Linnemann et al. 2019, S. 27) ..... - 19 -
Abbildung 13 Paketübertragungssicherheit nach Rate (Magrin et al. 2019) ..... - 21 -
Abbildung 14 LoRa-Erweiterungen für Energiemesssysteme [PL9/PL1/PL8] ..... - 25 -
Abbildung 15 Einfachste Netzwerkstruktur ..... - 33 -
Abbildung 16 Heltec Wireless Stick (Heltec Automation 2021) ..... - 34 -
Abbildung 17 Multi-Star-Netzwerk ..... - 36 -
Abbildung 18 Schema der Softwareinstanzen des Servers ..... - 41 -
Abbildung 19 Konfiguration von Loridane auf den MQTT Broker ..... - 42 -
Abbildung 20 Import in eine bestehende Node Red Instanz ..... - 44 -
Abbildung 21 Beschaltung des Heltec Wireless Stick ..... - 47 -
Abbildung 22 Loridane Signalpfad Uplink (größere Darstellung Anhang IV) ..... - 51 -
Abbildung 23 Datenübermittlung nach Zeitscheibe ..... - 58 -
Abbildung 24 Loridane GUI Tab-Menü ..... - 62 -
Abbildung 25 Loridane GUI Tab Messages ..... - 63 -
Abbildung 26 Loridane GUI Tab Global Config ..... - 64 -
Abbildung 27 Loridane GUI Tab Configure Node ..... - 66 -
Abbildung 28 Loridane GUI Tab System ..... - 67 -
Abbildung 29 Loridane GUI Generate Mac ..... - 68 -
Abbildung 30 Loridane GUI WiFi Credentials ..... - 69 -
Abbildung 31 Maximale Paketübermittlungsrate nach Gateways und Spreading Factor ..... - 72 -
Abbildung 32 Elektrischer Wirkverbrauchszähler ORNO OR-WE-520 ..... - 74 -
Abbildung 33 Langzeitaufnahme mit Loridane (Ausschnitt) ..... - 75 -
Abbildung 34 Vergleich der Wahrscheinlichkeiten für erfolgreiche Paketübermittlung bezogen auf die Uplinkrate ..... - 79 -

Tabellenverzeichnis

Tabelle 1 Geometrische Parameter der optimierten Antenne ..... - 14 -
Tabelle 2 Gegenüberstellung einiger Datenübertragungstechniken ..... - 23 -
Tabelle 3 LoRa-fähige Energiezähler und LoRa-Erweiterungen für Bestandszähler ..... - 24 -
Tabelle 4 Genauigkeitsklasse MID Zählwerke +5°C bis +30°C (Richtlinie 2014/32/EU, S. 210) ..... - 25 -
Tabelle 5 Anforderungen an ein drahtloses Energiemesssystem ..... - 32 -
Tabelle 6 Energieversorgung des Heltec Wireless Stick (Heltec Automation 2021) ..... - 46 -
Tabelle 7 Konfigurationsflaggen für Nodes und Gateways ..... - 50 -
Tabelle 8 Versuchsdaten Loridane Langzeit- und Belastungstest ..... - 76 -
Tabelle 9 Fehlerkategorien der Langzeit-Datenanalyse ..... - 77 -
Tabelle 10 Versuchsergebnisse ..... - 78 -
Tabelle 11 Kostenschätzung LoRaWAN für eine Installation an 10 Produktionsmaschinen ..... - 80 -
Tabelle 12 Kostenschätzung Loridane für eine Installation an 10 Produktionsmaschinen ..... - 80 -

Codeverzeichnis

Code 1 Präprozessorliterale zur Grundkonfiguration (C) ..... - 39 -
Code 2 Installation der Loridane-Software auf dem Raspberry Pi (Shell) ..... - 42 -
Code 3 Globales Kontextobjekt (JavaScript) ..... - 44 -
Code 4 Nomenklatur der UID (txt) ..... - 48 -
Code 5 Uplink-Protokoll (txt) ..... - 49 -
Code 6 Downlink Protokoll zur globalen Konfiguration (txt) ..... - 50 -
Code 7 Konfiguration eines einzelnen Gerätes (txt) ..... - 51 -
Code 8 JSON Objekt zur Übermittlung der Uplinks an den Server (JSON) ..... - 52 -
Code 9 Loridane Ordnerstruktur (txt) ..... - 52 -
Code 10 Datenbankformat (JSON) ..... - 53 -
Code 11 Acknowledgement (txt) ..... - 55 -
Code 12 Ausrollen der Time Disk (txt) ..... - 57 -
Code 13 Berechnung der Zeitscheibenlänge (JavaScript) ..... - 59 -
Code 14 Algorithmus zum Berechnen des Zeitscheibensegmentes auf dem Node (C++) ..... - 60 -
Code 15 Evaluation des Sendeintervalls auf dem Node (C++) ..... - 61 -
Code 16 Evaluation des Zeitslots auf dem Node (C++) ..... - 61 -
Code 17 Nutzbare Funkkanäle (JSON) ..... - 70 -
Code 18 Beispiel Testdatensätze (txt) ..... - 77 -
Code 19 Ausgabe eines Testergebnisses (txt) ..... - 78 -

Symbolverzeichnis und Glossar

Basissystem

Die erste Spalte der folgenden Liste zeigt die im Text verwendeten Symbole für die auftretenden physikalischen und mathematischen Größen und Einheiten. In der zweiten Spalte wird die Bedeutung des Symbols beschrieben.

Zahlenformat	**Weise**
binär	10011010b
hexadezimal	0x9A
dezimal	154, ohne Prä- oder Suffix

Symbol	**Bedeutung**
r	Radius
B_{max}	höchste prozentuale Kanalbelegung
BW	Bandbreite eines Funkkanals
c_{time}	Zeitkonstante
CR	Coding Rate
CRC	boolscher Wert
D	mittlerer Leiterabstand
d	Leiterdurchmesser
DE	boolscher Wert
e, E	Fehler
ε_{rL}	Dielektrizitätskonstante der Luft
I	Elektrischer Strom
i	Impulswertigkeit elektrischer Arbeit
I_{min}	kleinstes Zeitintervall
I_{sync}	Synchronisationszeitintervall
IH	boolscher Wert
L_{arith}	Paketverlust aufgrund arithmetischer Fehler
l_{max}	größte Leitungslänge
L_{sum}	Gesamtpaketverlust
L_{ul}	Empfangspaketverlust
n_g	Anzahl Gateways
n_n	Anzahl Nodes

n_u	Uplinkkapazität
ns	Nanosekunde
P	Leistung
P_0	emmitierte Leistung eines Senders
P_{noise}	Rauschleistung
P_{nutz}	Nutzleistung
P_{refl}	reflektierte elektromagnetische Leistung
φ	Phasenwinkelverschiebung
R_B	Bitrate (Brutto)
R_{Braw}	Rohbitrate ohne Protokolloverhead (Netto)
R_{pd}	Paketübermittlungsrate
S	diskrete Frequenzschritte
SF	Spreading Factor
$T_{i_{mess}}$	Messintervall
t_{pd}	Leitungslaufzeit
t_s	Schwellzeit eines Signals
T_{sn}	nächster Sendzeitpunkt
τ	
TD_{size}	Gesamtlänge einer Zeitscheibe
TOA	Länge der ununterbrochenen Sendedauer
TOA_b	Sendezeit für ein bit
TOA_{mean}	gemittelte TOA
U	elektrische Spannung
$ULPDR$	Wahrscheinlichkeit einer erfolgreichen Übertragung
$VSWR$	elektromagnetisches Reflexionsverhältnis
W_{slot}	Zeitslotlänge

Einheiten	Bedeutung
GWh	Gigawattstunde
dBi	Dezibel (isotrop), Vergleichswert zu einer idealen Antenne mit allseitig gleichmäßiger Abstrahlcharakteristik im Gegensatz zu einer Antenne mit Richtwirkung
dBm	Dezibelmilliwatt, dekadischer Logarithmus der an einer Stelle im Signalpfad vorhandenen Sendeleistung/induzierten Empfangsleistung
kWh	Kilowattstunden
W	Watt
sym	Symbol [bit]
imp	Impulsanzahl
h	Stunde
s	Sekunde
Ω	Ohm
msg	Anzahl Nachirchten
mon	Monat

Abkürzung	Bedeutung
ADR	Adaptive Datenrate
AES	Advanced Encryption Standard
ASCII	American Standart Code for Information Interchange
ASK	Amplitude SK
CSS	Chirp-Spread-Spectrum
CSV	Comma Separated Values
ED	End Device
ETA-Fabrik	Modell- und Forschungsfabrik der PTW-Arbeitsgruppe „Energietechnologie und Anwendung in der Produktion“
FSK	Frequency SK
GPIO	General Purpose Input/Output
HAL	Hardware Abstraction Layer
HWS	Heltec Wireless Stick
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
IP	Internet-Protokol
ISM	Industrial Scientifical Medical
JS	JavaScript
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LAN	Local Area Network
LoRa	Ein „**Lo**ng **Ra**nge“- Funkstandard der international lizenzfreie Mittelfrequenzen im Sub-Gigaherz-Band nutzt, vgl. IEEE 802.15.4

LoRaWAN	**Lo**ng **Ra**nge **W**ide **A**rea **N**etwork, ein von dem Industriekonsortium „LoRaWAN-Alliance“ entwickelter Netzwerkstandard auf Basis von LoRa
LORIDANE	Zusammenfassende Beschreibung der gesamten hier entwickelten Netzwerktopologie bestehend aus LoRa-Messstellen / Nodes, Gateways und dem Raspberry Pi Server, akr.: **Lo**ng **R**ange **I**ndependent **D**ata **A**ggregation **Ne**twork
LTE	Long Term Evolution
MID	Measurement Instruments Directive
MQTT	Message Queuing Telemetry Transfer
MyC	Micro Controller
NB	Narrow-Band
OPC/UA	Open Platform Communication Unified Architecture
OSI	Open Systems Interconnection
PSK	Phase SK
PTW	Institut für Produktionsmanagement, -Technologie und Werkzeugmaschinen an der TU Darmstadt
Raspberry Pi	Linux-Computer-Hardware der Raspberry Pi-Foundation©
RSSI	Received Signal Strength
RTC	Real Time Clock
SF	Spreading Factor, Spreizfaktor
SFTP	Secure File Transfer Protocol
SK	Shift-Keying-Verfahren
SNR	Signal to noise ratio
SoC	System on a Chip
SRD	Short Range Device
SSH	Secure Shell
SSID	Service Set Identifier
SWR	Standing Wave Ratio
TDMA	Time Division Multiple Access
TOA	Time On Air
TTN	The Things Network
TU	Technische Universität
txt	Textformat
UDP	User Datagram Protocol
UID	Unique Identifier
ULPDR	Upload Packet Delivery Ratio
USB	Universal Serial Bus
VSWR	Voltage Standing Wave Ratio (spannungsbezogene SWR)
WAN	Wide Area Network
WiFi	Wireless Fidelity == WLAN
WLAN	Wireless Local Area Network, drahtloser Funkstandard, IEEE 802.11
WWW	World Wide Web

Indizes und Verweise	Bedeutung
(Autor/Hrsg., Jahr, Seite)	Literaturverweis
[SL]	Verweis auf das Softwareverzeichnis
[PL]	Verweis auf das Produktverzeichnis
(5.4.1)	Verweis auf einen Abschnitt in dieser Arbeit

1 Einleitung

1.1 Problemstellung

Nach DIN EN ISO 500001 sind Unternehmen dazu angehalten und ab einem Jahresenergieverbrauch von mehr als 5 GWh dazu verpflichtet, ihre Energieverbräuche zu überwachen und transparent zu erfassen. Das zentrale Argument hierfür ist der Klimaschutz als Resultat eines verbesserten Monitorings und dem sich daraus ergebenden Potenzial zur Effizienzsteigerung. Daneben können, zum Beispiel, Energie- und damit Kosteneinsparungen, das Erlangen einer Zertifizierung, die Möglichkeit zum Verkauf überzähliger CO_2-Zertifikate oder Prestigegründe, im Sinne der Werbung mit nachhaltigeren Produktionsstätten, Anlass zur Implementierung eines zentralisierten und automatisierten Energiemesskonzeptes sein.

Das zunehmende Auftreten der IT-Werkzeuge der Industrie 4.0 in den technischen Branchen fördert die Möglichkeiten, aber auch die Notwendigkeit, Prozessoptimierungen in Anbetracht der Energieflüsse vorzunehmen. Derartige Optimierungen verlangen sowohl eine zeitlich hinreichende Auflösung des Energiemonitorings als auch eine engmaschige Beobachtung, bezogen auf einzelne Maschinen oder auf Teilprozesse innerhalb der Maschine. Hinsichtlich dessen ergeben sich zwei Herausforderungen. Ein manuelles Ablesen der Verbrauchsstellen und händisches Einpflegen der Werte in einen Report, zum Beispiel in Form einer Tabellenkalkulation, ist ebenso unpraktikabel wie eine Verdrahtung der zahlreichen Messtellen über Ethernet oder Datenbussysteme. Des Weiteren sind viele Produktionsstätten räumlich zu sehr ausgedehnt, um eine flächige WLAN-Abdeckung zu realisieren oder die Unternehmen sind an verschiedenen Standorten tätig. Um diese Anwendungen abzudecken, ist Software und entsprechend reichweitenstarke Hardware verfügbar, etwa auf Basis der LoRa-Funktechnologie. Allerdings ist einerseits die übersichtliche Auswahl an Hard- und Software kostenintensiv, sodass vor allem für kleine und Kleinstunternehmen ein Investitionsrisiko vermeintlich abzuwägen ist. Andererseits ist die vorhandene Software darauf ausgelegt einen globalen Standard zur Nutzung von LoRa im Rahmen eines Wide Area Network (WAN) zu setzen. Die sich dadurch ergebende zentralisierte Implementierung und Datenverarbeitung ist daher vielseitig einsetzbar, jedoch für bestimmte Zwecke, unter anderem für ein Energiemonitoring wie oben beschrieben, ungeeignet.

1.2 Lösungsweg

In dieser Arbeit wird ein wegbereitender Vorschlag in Form einer realen Implementierung erarbeitet, der sich der vorgenannten Problemstellung widmet. Im Zuge dessen wird ein Messstellen- und Netzwerkkonzept auf Basis der LoRa-Technologie implementiert und versuchsweise an Energiemesstechnik installiert. Kerninhalt der Implementierung ist die

- Anbindung von LoRa-fähigen Microcontrollern an elektrische Wirkverbrauchsmesser,
- Drahtlose Übermittlung
    - der Messwerte an ein LoRa Gateway[1],
    - der Datensätze von einem Gateway an einen Raspberry Pi basierten Server,
- Speicherung in einer Datenbank,
- Übermittlung / zur Verfügung stellen der aggregierten Datenbasis an eine Drittanbieter-Software, die ein Monitoring und Reporting ermöglicht.

1.3 Aufbau der Arbeit

In Kapitel 2 wird zunächst eine technologische Einführung bezüglich der LoRa-Technik und der bereits am Markt erhältlichen LoRaWAN-Technologie vorgenommen. Es folgt eine Ergebnisdarstellung der Produktrecherche sowie eine Gegenüberstellung und Erörterung der den Ansätzen eigenen Vor- und Nachteilen. Im Rahmen dessen werden in Kapitel 3 die Anforderungen an ein drahtloses Energiemessstellenkonzept beschrieben und diese in Kapitel 4 in eine Konzeption überführt. In Kapitel 5 wird die Realisierung des LORIDANE-Konzeptes im Detail erläutert. Kern des Kapitels 6 ist die Beschreibung der Testimplementierung an elektrischen Wirkverbrauchszählern und die Evaluation der Funktionalität des Systems. In Kapitel 7 wird ein Ausblick gefasst und das Potential der Lösung aufgezeigt.

[1] Gateway hier als physikalische Vermittlungsstelle zwischen LoRa und WLAN

2 Grundlagen und Stand der Technik

In diesem Kapitel wird in die relevanten technischen Grundlagen von LoRa und herkömmlichen Funksystemen eingeführt. Im Anschluss folgt eine auf die Nutzung für Energiemesssysteme bezogene Bewertung der Marktsituation und eine Diskussion der Vor- und Nachteile der einzelnen Technologien.

2.1 Notwendigkeit der Verbrauchsdatenerfassung

Neben der in (1.1) genannten normativen Notwendigkeit zur Verbrauchsdatenerfassung besteht in einem Betrieb mit verschiedenen Ressourcenumsätzen unter Umständen der Wunsch nach einem Verbrauchsmonitoring für die betriebsinterne Quantifizierung von Vorgängen und Umsätzen, sowohl unter Umweltschutz- als auch unter Kostenaspekten (Weyand et al. 2021, S. 264). Als eine mögliche Methodik dies zu realisieren, steht die MFCA (Material Flow Cost Analysis) zur Verfügung, die neben dem materiellen Ressourcenverbrauch auch den elektrischen Energieverbrauch bilanziert (Weyand et al. 2021, S. 266). Das vierschrittige Verfahren umfasst neben der Zieldefinition als Begründung für ein Ressourcenmonitoring auch die Suche nach zentralen Verbrauchsstellen („Hotspots“), die Festlegung auf zu analysierende Verbrauchsarten und die Implementierung einer Messstrategie sowie deren Umsetzung (Weyand et al. 2021, S. 266). Zumindest die beiden letzteren Schritte erfordern dabei den Einsatz von Messtechnik. Damit soll eine Datengrundlage geschaffen werden, um neben der reinen Erfassung des Ressourceneinsatzes auch den schlussendlichen Nutzen zu evaluieren (Weyand et al. 2021, S. 266). Dies erfordert den Einsatz von Messtechnik, die implementierbar, wirtschaftlich, hinreichend genau und verfügbar ist. Die Erfüllung der Voraussetzungen zur Implementierung eines solchen Messsystems, bezogen auf den elektrischen Energieumsatz, ist das Ziel der vorliegenden Arbeit.

2.2 Technische Grundlagen

2.2.1 Einordnung LoRa

Der Begriff LoRa umfasst die physikalische Ebene einer Funktechnologie zum Datenaustausch zwischen kompatiblen Systemen. Das Kürzel impliziert die vorrangige Stärke dieser Technologie – Long Range. Während die weit am Markt verbreiteten drahtlosen Übertragungstechnologien WLAN, Bluetooth und LTE auf die Datenübertragung mit großen Datenraten und kleinen Latenzzeiten optimiert sind, ergeben sich für die Übertragung von Telemetriedaten andere Anforderungen. Hier sind zumeist Datenpakete mit einer Größe von wenigen Bytes zu versenden. Hinzu kommt gegebenenfalls Bedarf an einer größeren Reichweite, der mit den oben genannten Technologien nicht unbedingt zu decken ist (Galinina et al. 2020, S. 256).

Dabei auftretende Zielkonflikte ergeben sich durch die Gegenüberstellung der Parameter Reichweite, Energieverbrauch und Datenrate, aber auch von Datensicherheit, Empfängersensitivität und Sendeleistung.

Um eine bis dato vorhandene Lücke zu schließen und Internet of Things (IoT)-Netzwerke sinnvoll möglich zu machen, hat die Semtech Corporation die LoRa Technologie entwickelt und im Jahr 2010 zum Patent angemeldet (Patent Nr.: US-7791415-B2). Im Jahr 2015 begründete Semtech mitunter den gemeinnützigen Verein LoRa Alliance, der seither mit der Schaffung von, diese Technik betreffenden, Standards betraut ist.

2.2.2 Funktionsweise

Die Emission von Funksignalen ist global stark reglementiert. Daher werden für LoRa lizenzfreie Frequenzbänder verwendet. In Europa wird dazu das 868Mhz-Band[2] genutzt. Ein auf dieser Mittelfrequenz liegendes Frequenzband kann im Vergleich zu anderen digital verwendeten Bändern als mittel- bis langwelliges Band eingeordnet werden. Dabei ist die Verwechslungsgefahr mit den sich über viele Jahrzehnte eingebürgerten Begriffen zu beachten. So liegt beispielsweise das VHF[3] Band um 50 MHz und der WiFi Standard IEEE 802.11ad um 60 GHz. Ein Zielkonflikt in der Wahl der Mittelfrequenz liegt in der Abwägung zwischen Signalreichweite, der Durchdringungsfähigkeit für verschiedene Stoffe und der erreichbaren Datenrate.

Die Datenrate nimmt mit zunehmender Kanalbandbreite proportional zu (Rembold 2017, S. 8), während sich die Dämpfung im Signalweg im Sub-Terahertz-Bereich durch atmosphärische Eigenschaften ebenfalls erhöht. Dies liegt zum Beispiel in der Resonanzabsorption (Rembold 2017, S. 37) häufiger atmosphärischer Stoffe begründet. Da das Übermitteln von Binärdaten im Kontext des IoT selten unter idealen Umständen, wie mit einer direkten Sichtverbindung und im Vakuum, stattfindet, liegt hierauf ein besonderes Augenmerk. Für die Telemetriedatenübertragung in urbaner Umgebung und damit auch beim Energiemonitoring im Produktionsumfeld wären durch Stahlbeton oder metallische Hindernisse sehr hohe Übertragungsfrequenzen ungeeignet (Rembold 2017, S. 20). Die Eindringtiefe des elektromagnetischen Feldes der Strahlung im hier betrachteten Spektrum beträgt in konduktiven Materialien ohnehin nur etwa einen Mikrometer (Schmüser 2012, S. 63). Daher gewinnen Reflexions- und Beugungseffekte an Relevanz. Da die Wellenlängen im Sub-Gigaherz-Bereich[4] eher in der Größenordnung der Hindernisse in der Produktionsumgebung entsprechen (Rembold 2017, S. 4), ergeben sich hierdurch positive Auswirkungen auf die Strahlenbeugung (Hering et al. 2012, S. 484) und damit auch indirekt auf die Reichweite, insbesondere dann, wenn zwischen Sender und Empfänger keine direkte Sichtverbindung besteht (Rembold 2017, S. 33). Eine Verwendung des SRD[5]-Bandes um 868Mhz (Bundesnetzagentur Vfg 133 / 2019 2019, S. 1) bietet sich für derartige Zwecke an, da es einen nutzbaren Kompromiss zwischen der physikalischen Robustheit des Signals und der Datenrate bietet.

Um digitale Nutzsignale mittels Funkwellen zu übertragen, bedarf es einer Technik, die diese Daten in einer zu definierenden Weise auf das Funksignal moduliert. LoRa verwendet dazu das

[2] 863 bis 869 MHz. USA 915MHz, China 920 MHz

[3] Very High Frequency, z. B. Radio und terrestrisches TV

[4] Bei 868 MHz circa 32 cm, bei 2,4GHz (WLAN) circa 10cm

[5] „non-specific Short-Range Devices "

Chirp-Spread-Spectrum-Verfahren CSS (Semtech Corp. 2015b, S. 1). Namensgebend für diese Modulationstechnik ist der zeitliche Frequenzverlauf bei der Übertragung eines Einzelsymbols, der durch seinen graduellen Verlauf zwischen Trägerfrequenz und Oberfrequenz an das Grillen-Zirpen[6] erinnert. Abbildung 1 stellt fünf Zyklen des Zirpens mit einer Trägerfrequenz von 868MHz und einer Bandbreite von 125kHz dar.

Hier wird der Hauptunterschied des CSS-Modulationsschemas zu konventionellen Verfahren, wie dem Frequency-Shift-Keying FSK, Phase-Shift-Keying PSK und Amplitude-Shift-Keying ASK deutlich. Alle Shift-Keying-Schemata haben gemeinsam, dass ein binärer Nutzdatenwert über eine binäre Änderung des Signals codiert wird. Bei CSS findet hingegen ein gradueller Frequenzübergang Anwendung.

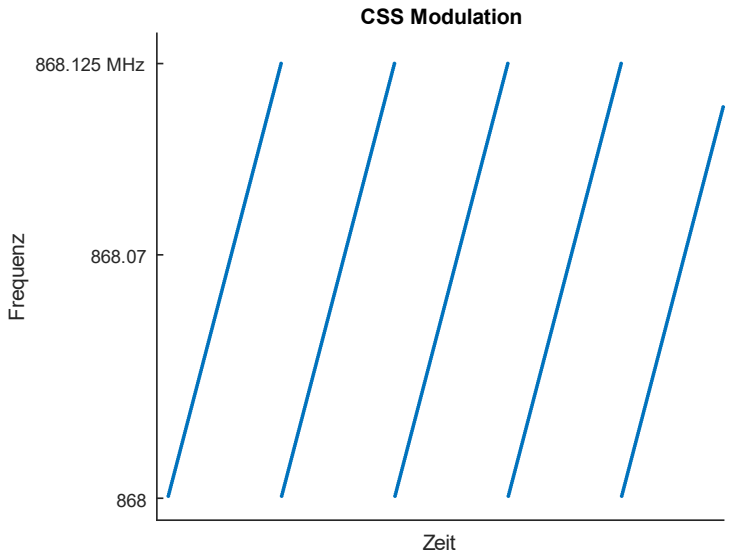


Abbildung 1 CSS-Modulationsschema mit 125kHz Bandbreite

Die Codierung der Nutzdaten erfolgt über die Detektion der Anfangs- und Endfrequenz relativ zur Trägerfrequenz. In Abbildung 2 ist die Funktionsweise der vier genannten Verfahren qualitativ dargestellt.

Die Graphen zeigen schematisch die Übertragung von zwei verschiedenen Symbolen. In der linken Hälfte ist die Übertragung eines Nutzwertes von 0b, während auf der rechten Seite ein Nutzwertsignal für 1b dargestellt ist.

Soll mit der ASK Modulation eine Null übertragen werden, so wird die Sendeleistung auf einen Wert verringert, die einem zu definierenden Bruchteil der Nennsendeleistung entspricht (John Anthes 2007, S. 9). Empfängerseitig bedingt dies die Verwendung der Amplitude der Präambel als Referenzwert. Sie darf sich im Verlauf der Übertragung eines Datensatzes nicht so erheblich ändern, dass ein 0b-Signal über den definierten Threshold steigt und ein 1b-Signal unter den Threshold fällt.

[6] engl. “to chirp”

Dies hätte einen Bitfehler in der empfängerseitigen Signalinterpretation zur Folge, der eine Nachricht potenziell unbrauchbar macht.

Das PSK basiert auf der Verschiebung der Phase des Signals bei gleicher Amplitude und Frequenz (Peebles 1987). Bei einer Trägerfrequenz von 868MHz bedeutet das eine Verschiebung der Amplitudenmaxima um 57.6 x10^{-9} Sekunden und ist damit anfällig für Signalstörungen durch beispielsweise Interferenzen, wenn andere Teilnehmer auf derselben Frequenz senden. (Ma et al. 2002, S. 11)

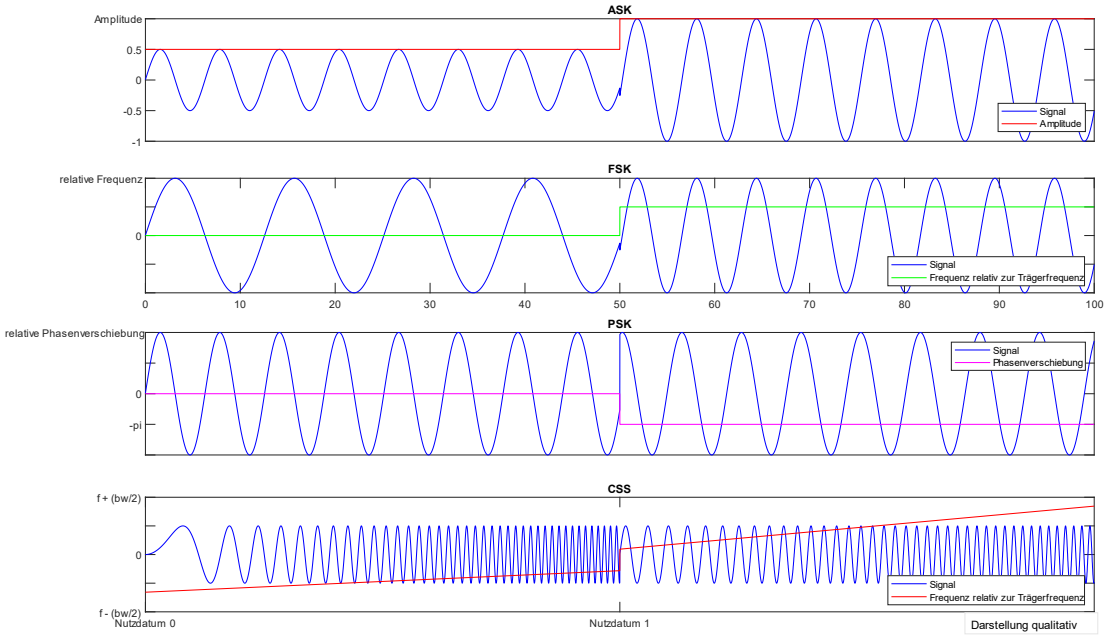


Abbildung 2 Gegenüberstellung konventionelle Modulationsverfahren

Bei einer FSK-Modulation ist, wie bei CSS, die Frequenz Informationsträger und die relative Frequenzhöhe - die Bandbreite - ist maßgeblich, da die Nutzdaten über einen Frequenzsprung codiert sind. Das für LoRa verwendete CSS basiert auf einer nicht binären Codierung über die Länge eines Zirpens, wie Abbildung 2 zeigt. Die Zeitabfolge der Signale ist von unten nach oben zu lesen.

Dabei wird die Frequenz nicht kontinuierlich erhöht, sondern ist jeweils für das verwendete Frequenzband durch 2^{19} Schritte gequantelt, da LoRa Chips über einen 19 bit Synthesizer verfügen (Semtech Corp. 2015b, S. 82). Damit ergibt sich für die in Europa verwendeten Frequenzen des von Semtech so definierten Band 1 von 862 bis 1020MHz

$$S = \frac{(1020-862)x10^6 Hz}{2^{19} Schritte} = 301.36 \frac{Hz}{Schritt}. \tag{2-1}$$

So ließe sich für einen FSK-Modulator mit zwei festen Frequenzen eine einfachste Bauart von zwei Schwingkreisen ableiten, zwischen welchen hin und her geschalten wird, um Binärsignale zu erzeugen (Peebles 1987). Die Erzeugung der Phasenverschiebung in einem PSK-Modulator um üblicherweise 180° geschieht durch Umpolen der Anregung.

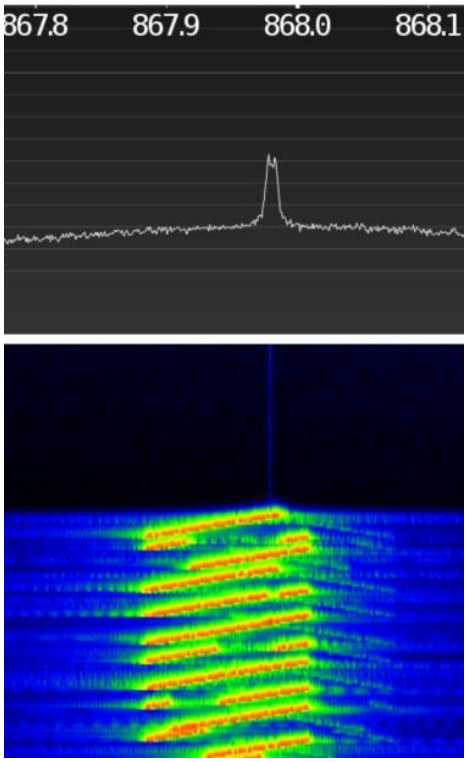


Abbildung 3 LoRa Signalspektrum (izinto GmbH 2020), geringfügig verändert

Da der LoRa-Modulator unter anderem sowohl verschiedene FSK-Moden (Semtech Corp. 2015b, S. 1) als auch Spread-Spectrum Moden unterstützt, ist der Aufbau dieser Modulatoren ungleich komplexer.

In der Praxis wird in modernen Systemen bei Anwendungen mit geringen Anforderungen an die Datenrate, Datenintegrität und Reichweite zumeist ein FSK Verfahren eingesetzt, zum Beispiel für Garagentorantriebe. Das beruht auf der beschriebenen relativen konstruktiven Simplizität dieser Systeme. Diese einfache Bauform geht mit Einschränkungen, bezogen auf Parameter, einher, die relevant für die Reichweite und die Signalrobustheit sind.

Hier ist insbesondere die Empfängersensitivität zu nennen. Beispielhaft für ein ASK-Verfahren ist eine Amplitudendifferenz von 4.77dB zwischen 0b und 1b Signal. Es ist offensichtlich, dass der Empfänger bei einem Signal-Rauschabstand (SNR[7]) von 4.77dB nicht zwischen dem Hintergrundrauschen und dem 0b Nutzsignal unterscheiden kann, da die Amplitude als Informationsträger neben dem Rauschen physikalisch nicht mehr aufzulösen ist. In der Praxis erzeugt ein solcher Empfänger bereits bei einem SNR von 14 dB etwa ein Prozent Bitfehler (John Anthes 2007, S. 10). Damit würde bei einer exemplarischen Nutzlast bestehend aus einer Identifikationsnummer (14 Byte) und einem telemetrischen Nutzwert (8 Byte), also einer Nachrichtenlänge von

[7] SNR – engl. signal to noise ratio

$$n_n = 24\,Byte \cdot 8\frac{bit}{Byte} = 192\,bit \tag{2-2}$$

jede der Nachrichten unbrauchbar, sofern keine Informationsredundanz vorliegt, da statistisch 1.9 Bitfehler pro Datensatz auftreten. Obwohl die empfängerseitige Signalleistung des 0b Signals nach

$$\frac{P_{nutz}}{P_{noise}} = 10^{\left(\frac{14\,dB}{10}\right)} = 25.12 \tag{2-3}$$

noch beim etwa 25-fachen der Rauschleistung liegt.

Hierbei wird der CSS-Ansatz eminent relevant, da eine Detektion und informationstechnische Auflösung des Signals nicht ausschließlich über die relative Frequenzhöhe, sondern den Gradienten der Frequenz erfolgt. Es liegt die Idee zugrunde, die Art des Zirpens auf Seiten des Modulators und des Demodulators vor der Datenübertragung festzulegen, um in definierter Weise nur das Zirpen mit einem bestimmten Frequenzgradienten zu detektieren. Dem wird die LoRa-Technik mit der Einführung verschiedener Stellparameter gerecht und kann dadurch die Empfängersensitivität signifikant verbessern (Semtech Corp. 2015a, S. 23). So lässt sich die Auswertung eines Signals mit einem SNR von -7.5dB (Semtech Corp. 2015b, S. 27) bei der höchsten Datenrate (SF7) (s. 2.2.5) realisieren, obwohl die Signalleistung nur etwa 18% der Leistung des Hintergrundrauschens entspricht. Dies lässt den Umkehrschluss zu, dass die Sendeleistung bei gleicher Entfernung zwischen Empfänger und Sender und konstantem Rauschen gegenüber einem konventionellem Modulationsverfahren deutlich verringert werden kann, oder bei gleicher Sendeleistung eine deutlich größere Entfernung zu überbrücken ist.

Um die Funktionsweise des CSS-Verfahrens im Detail wieder zu geben, werden im Folgenden einige notwendige Begriffe eingeführt.

2.2.3 Chip

Wie in Abschnitt 2.2.2 erläutert, findet bei der Informationsübertragung via CSS ein gradueller Frequenzübergang Anwendung. Im Zuge dessen wird ein Nutzwertsignal mit festgelegter Bandbreite in einzelne Frequenzimpulse (Chips) unterteilt. Die Chiprate entspricht der Bandbreitenfrequenz. Für das Beispiel aus Abbildung 1 mit 125kHz Bandbreite bedeutet dies, dass 125000 Chips pro Sekunde versendet werden, deren Frequenzhöhe sich jeweils zu den Vorhergehenden ändert. Die Änderung beträgt hierbei mindestens einen Schritt nach Gleichung 2-1.

2.2.4 Chirp

Ein Chirp setzt sich aus den einzelnen Chips zusammen, die einem Bandbreitendurchlauf entsprechen. Im genannten Beispiel sind dies ebenfalls 125000 bei 125kHz Bandbreite.

Ein Chirp beginnt nicht obligatorisch bei der Trägerfrequenz, sondern kann bei definierten Bruchteilen zwischen der Trägerfrequenz und der Summe aus Träger- und Bandbreitenfrequenz beginnen. Diese Quantelung ist abhängig vom Spreading Factor SF.

2.2.5 Spreading Factor

Üblich für LoRa Modulationen sind ganzzahlige Spreizfaktoren zwischen 7 und 12. Die Signalbandbreite wird dazu in n Frequenzbänder aufgeteilt, wenn sich n aus

$$n = 2^{SF}; \quad SF = 7, \dots, 12 \tag{2-4}$$

berechnet. Hierdurch wird erreicht, dass mit einem Chirp nicht nur 0b und 1b codiert werden kann, sondern $2^{SF} bit$. Für einen Chirp wird die Bandbreite immer einmalig und vollständig durchlaufen. Beginnt der Chirp oberhalb der Trägerfrequenz, wird der Chirp beendet, indem die Frequenz bei Erreichen der Oberfrequenz wieder herunter auf die Trägerfrequenz springt und bis zur Startfrequenz weitergeführt wird. Bei der Verwendung eines SF von 7 ergeben sich somit 2^7 - 1 (= 127) mögliche Startfrequenzen für den Chirp. Auf diese Weise kann mit einem Symbol ein 7 Bit codierter Wert übertragen werden. Abbildung 4 zeigt aufeinander folgende Zirp-Zyklen mit den Spreizfaktoren 7 und 9 für die Darstellung von 0000000b und 000000000b (Null in 7Bit- und 9Bit-Binärdarstellung).

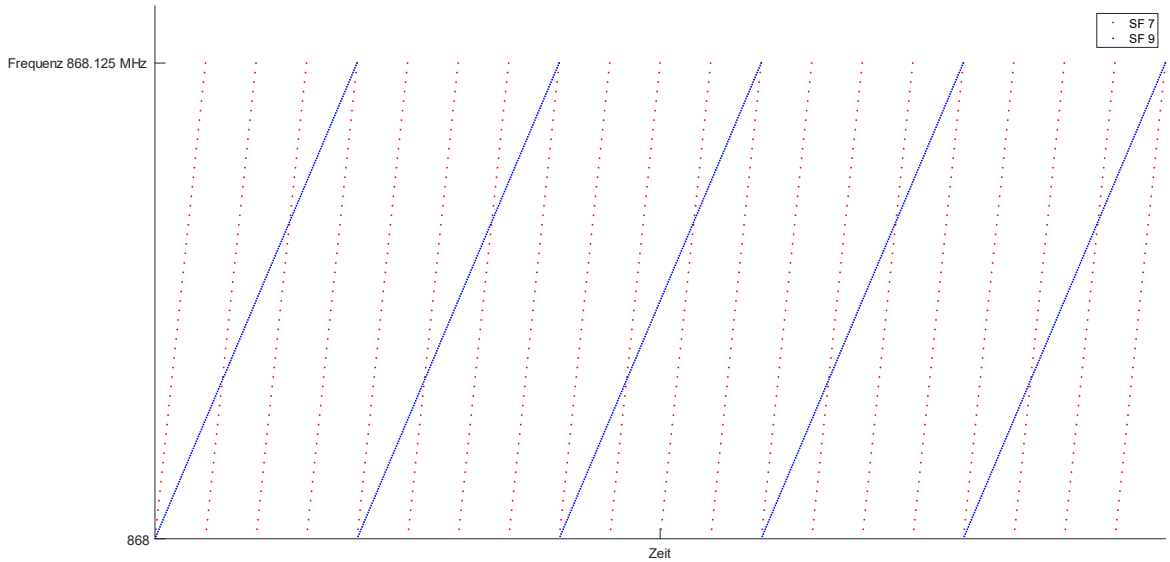


Abbildung 4 Vergleich SF7 und SF9

Hier wird deutlich, dass die Wahl des SF einen direkten Einfluss auf die Datenrate hat. Da ein Chip pro Hertz Bandbreite versendet wird, benötigt ein Chirp bei der Wahl des SF9 (SF_1) nach

$$c_{time}\tau = 2^{SF_0 - SF_1} \cdot \tau = 4\,\tau \tag{2-5}$$

gegenüber dem SF7 (SF_0), genau die vierfache Zeit. Daraus folgt, dass sich die Datenrate mit einer Erhöhung des SF auf den nächsten ganzzahligen Wert halbiert.

Die Symbolrate R_S lässt sich folglich aus

$$R_S = \frac{BW}{2^{SF}} \left[\frac{sym}{sek}\right] \tag{2-6}$$

mit der Kanalbandbreite BW in Hertz bestimmen.

Da die Symbolbreite dem Spreading Factor entspricht, ergibt sich somit für die Rohbitrate R_{Braw}

$$R_{Braw} = \frac{SF \cdot BW}{2^{SF}} \left[\frac{bit}{sek}\right]. \tag{2-7}$$

2.2.6 Coding Rate

In Abschnitt 2.2.2 wurde bereits das Auftreten von Bitfehlern in der empfängerseitigen Signalinterpretation und die damit einhergehende Notwendigkeit einer Datenredundanz beschrieben.

Bei LoRa wird diese Redundanz in Form einer Coding Rate berücksichtigt. Eine vollständige Redundanz ließe sich erreichen, indem jedes Byte zweimal versendet wird. Damit ergäbe sich die Ausnutzung der durch die Parametrierung des Transceivers festgelegten Datenrate zu 4/8. Ein weiterer möglicher Ansatz ist die Nutzung eines zusätzlichen Paritäts-Bits als Prüf-Bit. Ein Paritäts-Bit gibt an, ob die Anzahl der vorangegangenen 4 Bits, welche mit einer 1b besetzt sind, ungerade (Parität ist 1b) oder gerade (0b) ist. Die Ausnutzung der Datenrate betrüge in diesem Fall 4/5. Dies entspricht dem, für den Großteil der getätigten LoRa-Übertragungen, genutzten Wert und ist für LoRaWAN-Anwendungen Standard. Semtech gibt in seinem Skriptum zur Funktionsweise der LoRa Modulation eine Definition der Coding Rate CR mit

$$CR = \frac{4}{x};\ \mathrm{x} = 4, \dots, 8 \tag{2-8}$$

an. Mit der Anzahl der Bits x, die für 4 Bit Nutzdaten übertragen werden müssen (Semtech Corp. 2015a, S. 9–10).

Für die standardmäßige CR von 4/5 bedeutet das eine Verringerung der Rohbitrate R_{Braw} um den Faktor 4/5 zur Nutzbitrate

$$R_B = CR \cdot R_{Braw} = \frac{SF \cdot CR \cdot BW}{2^{SF}} \left[\frac{bit}{sek}\right], \tag{2-9}$$

beziehungsweise ihrem Reziprok, der Time-On-Air (TOA) für ein Bit TOA_b.

$$TOA_b = \frac{1}{R_B} = \frac{1}{CR \cdot R_{Braw}} = \frac{2^{SF}}{SF \cdot CR \cdot BW} \left[\frac{sek}{bit}\right] \tag{2-10}$$

Die gesetzlichen und informationstechnischen Gründe für die Notwendigkeit einer genauen Kenntnis der benötigten Sendezeit werden in den folgenden Abschnitten näher betrachtet.

2.2.7 Protokoll

Neben den Nutzdaten und den Bits zum Zwecke der Vorwärtsfehlerkorrektur aus der CR wird ein LoRa-Datenpaket um eine Präambel und einen Header erweitert.

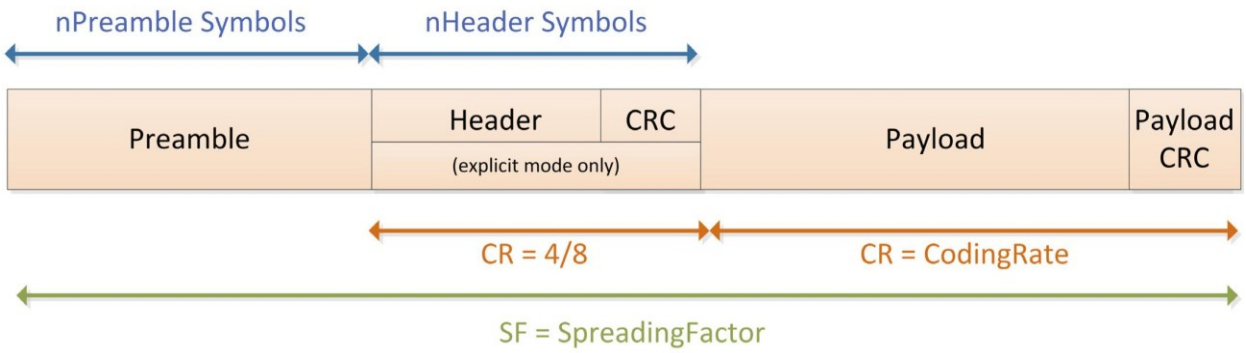


Abbildung 5 Protokollstruktur LoRa (Semtech Corp. 2015b, S. 29)

Der Schwerpunkt der folgenden Betrachtungen liegt auf den Standardeinstellungen eines LoRa Transceivers, wenngleich eine individuelle Parametrierung in weiten Grenzen möglich ist.

Abbildung 5 zeigt schematisch die Grundstruktur eines LoRa-Datenpaketes mit folgenden Eigenschaften der einzelnen Datensätze (Semtech Corp. 2015b, S. 29): Die

- **Präambel**
  dient der Synchronisation zwischen Sender und Empfänger und wird durch ein 1Byte langes Sync-Word angeführt. Der
- **Header (Expliziter Modus)**
  enthält Informationen über die Länge der Nutzinformation, die verwendete CR und optional eine Prüfsumme. Die
- **Payload**
  entspricht dem Nutzdatenwert und der
- **CRC (Cyclic Redundancy Check)**
  ist eine optionale Prüfsumme.

Die Länge der Präambel ist werksseitig auf 12 Symbole voreingestellt (Semtech Corp. 2015b, S. 29). Sie enthält als erstes Byte einen Zahlenwert - dem Sync-Word - zwischen 0x00 und 0xFF der zwischen Sender und Empfänger gleichermaßen vorkonfiguriert werden muss. Andernfalls würde empfängerseitig eine weitere Auswertung des Signals unterlassen. Dies bietet den Vorteil, dass voneinander unabhängig betriebene LoRa-Systeme die Nachrichten anderer Systeme ignorieren können.

Der Header kann sowohl implizit als auch explizit konfiguriert werden. An dieser Stelle soll nur der explizite Modus betrachtet werden. Dieser enthält die Information über die Nutzdatenlänge in Bytes und die CR von 4/5 bis 4/8.

So ergibt sich die tatsächliche Datensatzgröße zu

$$n = 8 + \frac{8PL - 4SF + 28 + 16CRC - 20IH}{4SF - 8DE} \cdot (CR + 4)\ [sym] \tag{2-11}$$

und vereinfacht sich unter der genannten Annahme der Standardparameter, wenn ein Header verwendet wird (IH = 0), keine Optimierung für geringe Datenraten (DE = 0) Anwendung findet und auf die Prüfsummen (CRC = 0) verzichtet wird (Semtech Corp. 2015b, S. 31) zu

$$n = 8 + \frac{8PL - 4SF + 28}{4SF} \cdot (CR + 4)\ [sym] \cdot SF\ \left[\frac{bit}{sym}\right]. \tag{2-12}$$

Beispielhaft für ein übertragenes Datum mit einer Identifikationsnummer (14byte) und einer 8byte Payload in Form eines Byte Buffers (insg. 24byte), ergibt sich so eine absolute Datensatzgröße von

$$\begin{aligned} n &= \left(8sym + \frac{8 \cdot 24\ - 4 \cdot 7 + 28}{4 \cdot 7} \cdot 5sym\right) \cdot 7\frac{bit}{sym} \\ &= 295{,}96bit \end{aligned} \tag{2-13}$$

welche auf 296Bit aufzurunden ist, für einen Spreading Factor 7 und somit eine Sendedauer von

$$T_S = 296bit \cdot \frac{2^7}{7 \cdot 125000\frac{bit}{s}} = 0.0433s = 43.3\ ms. \tag{2-14}$$

(2-13) in (2-10)

2.2.8 Signalinversion

LoRa-Module werden üblicherweise in einem Modus betrieben, der vorsieht, dass ein Chirp von einer tieferen zu einer höheren Frequenz ausgeführt wird (upchirp). Entsprechend findet eine Demodulation durch die Überlagerung eines umgekehrten Frequenzmusters statt (downchirp). Ein konfigurierbarer „InvertIQ"-Modus (Semtech Corp. 2015b, S. 114) kann durch setzen eines Registereintrages aktiviert werden und invertiert die Modulationslogik. Invertierte Signale sind für andere Geräte in Reichweite und einer Konfiguration auf den normalen Modus nicht detektierbar. Dies gilt ebenfalls vice versa.

2.2.9 Antenne

Die bestmögliche Ausnutzung der Sendeleistung und der Empfängersensitivität ist nur dann gegeben, wenn alle Teilnehmer eine, auf den Frequenzbereich und den Aufstellungsort angepasste, Antenne verwenden. Im Handel sind verschiedene Antennentypen mit unterschiedlichen Charakteristika für das SRD-Band erhältlich. Das breiteste Angebot ist für, mit Kunststoff gekapselten, Helix-Antennen (s. Abbildung 6, links) gegeben. Diese bieten die Vorteile einer geringen Bauraumnutzung und günstigen Herstellung. Nachteile der Helix-Bauform sind eine nur teilweise Ausnutzung der elektromagnetischen Leistung durch ihre Breitbandigkeit und die Notwendigkeit eine, im Vergleich zur Wellenlänge große, Reflexionsfläche unterhalb der Antenne bereitstellen zu müssen. Aufgrund der breiten Frequenzakzeptanz werden häufig baugleiche Antennen für den Europäischen Markt (868MHz) und andere Märkte bis 920Mhz angeboten. Entsprechend groß ist die

am Empfänger induzierte Rauschleistung durch Fremdsignale und damit die Empfindlichkeit gegenüber Störquellen.

Eine lineare Polarisierung der elektromagnetischen Wellen und eine parallele Aufstellung der Antennen bietet eine bessere Ausnutzung der Signalleistung. Des Weiteren ist für eine flexible relative Aufstellbarkeit eine isotrope Charakteristik der Abstrahlung gegenüber einer Richtcharakteristik, zumindest empfängerseitig, vorzuziehen. Dies ist im Falle einer einfachsten Bauart mit einer Ground-Plane-Antenne mit vier Radialen (Abbildung 6, rechts) zu erreichen.

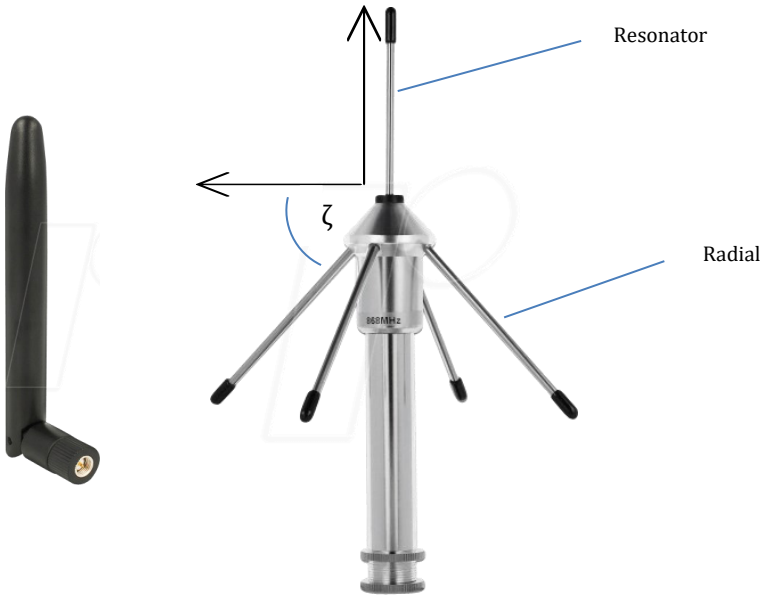


Abbildung 6 868 MHz Antennen (v.l.n.r., Helix (THB GmbH 2021), Ground-Plane)

Antennen solcher Bauart schlagen durch die aufwendigere Herstellung mit einem höheren Anschaffungspreis (ab ca. 50 € (Ground-Plane) gegenüber ab ca. 10€ (Helix)) zu Buche. Um eine für den LoRa-Frequenzbereich optimierte Antenne mit den geforderten Eigenschaften zu erreichen, wird hier mit dem MatLab-Antenna-Designer [SL1] die numerische Optimierung einer Ground-Plane-Antenne durchgeführt. Ziel der Optimierung ist die Generierung einer Antennengeometrie, die möglichst schmal-bandig ist und sich der gewünschten kugelförmigen Abstrahlcharakteristik möglichst nähert (s. Abbildung 7).

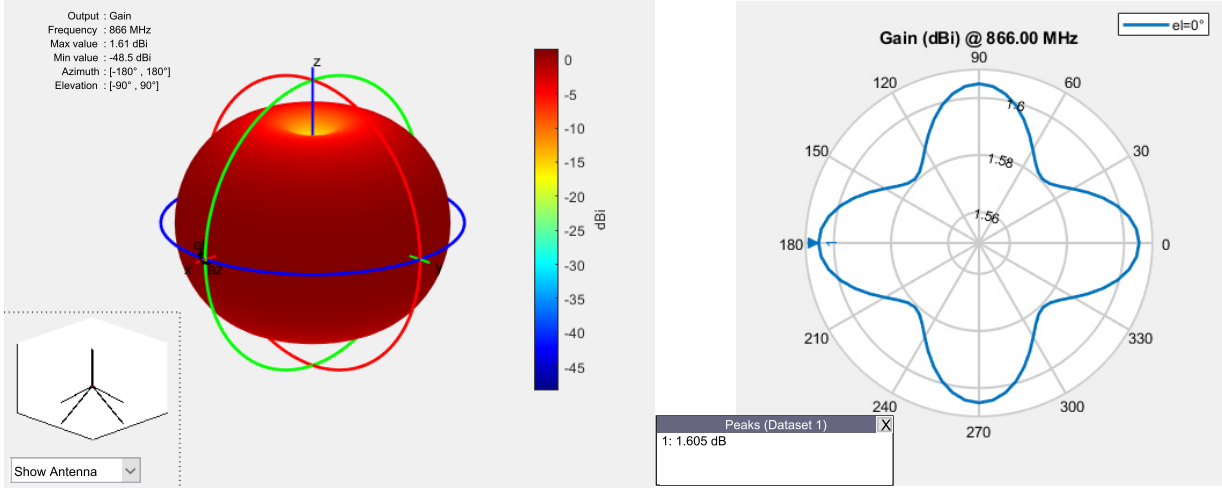


Abbildung 7 Abstrahlcharakteristik der optimierten Antenne

Das hierdurch erreichte verbesserte Bandpassverhalten der Antenne erhöht die isolierte Akzeptanz des geforderten Frequenzbereiches. Währenddessen ist die Ansprechbarkeit, außer an den Antennenpolen, sehr homogen. Die einfache Verfügbarkeit des Grundmaterials wird erreicht, indem die Optimierung mit Materialkennwerten durchgeführt wurde, die handelsüblichem einadrigem Verlege-Stromkabel aus Kupfer mit 1.5mm² Leiterquerschnitt entspricht. Mit den gegebenen Parametern ergibt sich die Antennengeometrie anhand der in Tabelle 1 dargestellten Werte.

Parameter	Wert
Leiterquerschnitt	1.5mm²
Resonatorlänge	7.74cm
Radiallänge	10.23cm
Radialenwinkel ζ (s. Abbildung 6)	33.5°

Tabelle 1 Geometrische Parameter der optimierten Antenne

Zur Generierung der Daten wurde in MatLab das Objekt einer Ground-Plane Antenne mit den Parametern für das Frequenzband von 863Mhz bis 869MHz und einer Mittelfrequenz von 866 MHz erstellt. Wie in der Praxis üblich zielt die Optimierung auf die Generierung einer $\lambda/4$ Antenne ab. Der Resonator hat damit das Viertel der Länge eines Leiters, für dessen Materialkonstanten das Potenzial bei Anregungsfrequenz eine Periode, vom Fußpunkt zur Spitze und zurück, durchläuft. Für einen frequenzabgestimmten Resonator und gegebener Materialkonstante (hier Kupfer) ist dessen Länge als Festparameter zu definieren (Resonanzfall), der aber einem Verkürzungsfaktor (Kark 2014, S. 267) unterliegt. Zur Optimierung, durch den in MatLab implementierten SADEA (surrogate model assisted differential evolution for antenna synthesis)-Algorithmus, mit 100 Iterationen, wurde als Hauptparameter die Erreichung eines ausgeprägten Bandpassverhaltens der Antenne festgelegt. Als Designparameter sind somit die Länge der Radiale und der Radialenwinkel definiert. Der erste Vorschlag des Algorithmus für die Radiallänge von 10.24cm wurde händisch auf 10.23cm verkleinert, fixiert und die Optimierung erneut durchgeführt.

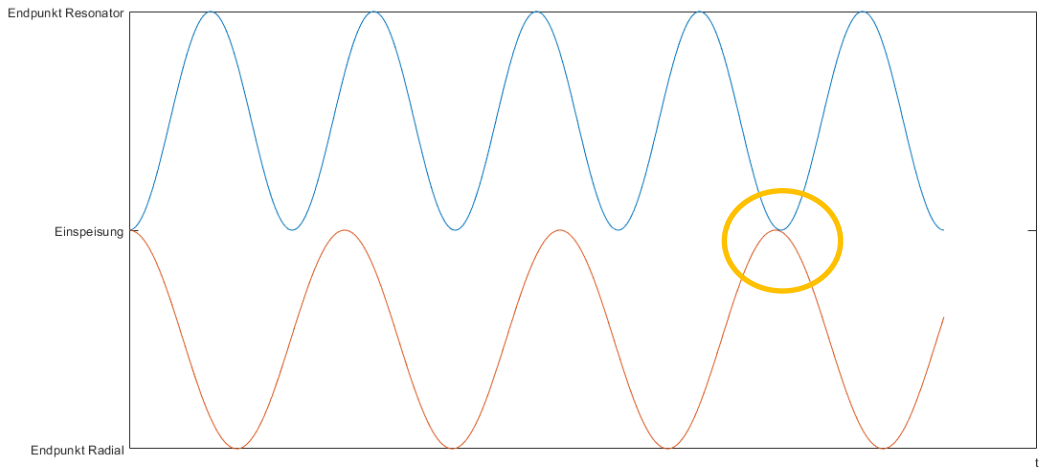


Abbildung 9 A-resonante Schwingung in Resonator und Radial

Damit ist erreicht, dass das Längenverhältnis von 774/1023 zwischen Resonator und Radial irrational ist und zwischen Zähler und Nenner Teilerfremdheit besteht. Folglich treffen sich die gegensätzlichen Amplitudenmaxima des, in den Leitern oszillierenden, Potentials nicht im Fußpunkt (s. Abbildung 9) und verringert den Betrag der maximal zu erwartenden Reflexion des elektromagnetischen Feldes in den Transceiver. Ebenfalls bringt dies eine kleine, inhärente Verbesserung des Stehwellenverhältnisses der Antenne mit sich. Grundlegend dafür ist die Annahme, dass, wie in (Kark 2014, S. 159) beschrieben, der Einfluss der Reflexion am Einspeisepunkt maximal ist und sich direkt auf das Stehwellenverhältnis auswirkt. Ausschlaggebend hierfür ist der Prozentuale Anteil der, in den Transceiver, zurückreflektierten Leistung S_{11}.

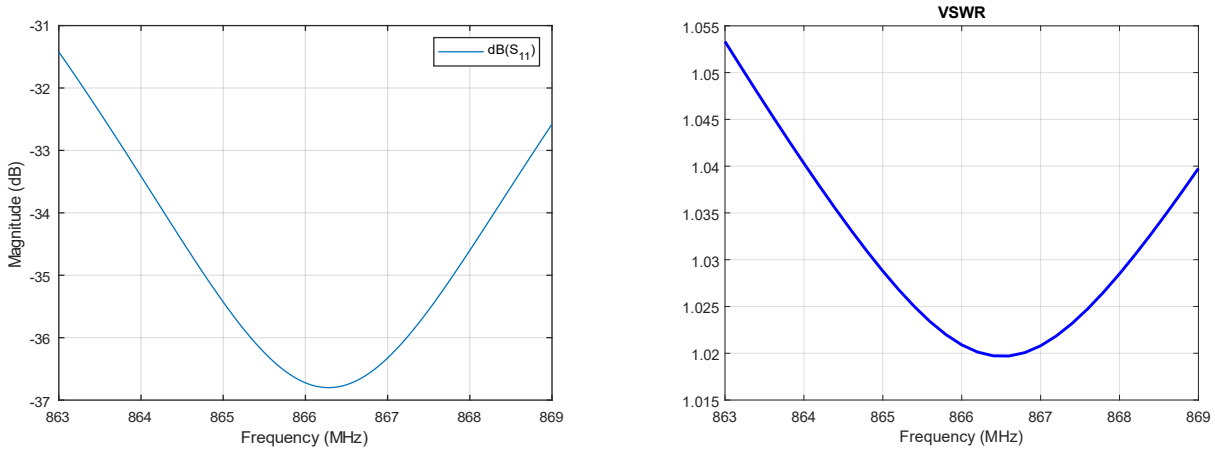


Abbildung 8 S_{11} und VSWR

Die rückreflektierte Leistungsanteil (Abbildung 8, links) wird, im definierten Frequenzband, mit einem Maximum bei 863 MHz genähert und beträgt somit etwa

$$\frac{P_{refl}}{P_0} = 10^{\left(-\frac{31.5\,dB}{10}\right)} = 7.08 \cdot 10^{-4} \tag{2-15}$$

oder weniger.

Direkt hieraus ergibt sich die VSWR (Voltage Standing Wave Ratio) nach (Kark 2014, S. 448) aus

$$VSWR(f) = \frac{1 + |S_{11}(f)|}{1 - |S_{11}(f)|} \tag{2-16}$$

für die gegebene Antenne zwischen 1.02 und 1.053 (Abbildung 8, rechts). Aufgrund der Breitbandigkeit kommerzieller Antennen sind diese in den Datenblättern zumeist mit SWRs <2.0 angegeben, wie zum Beispiel in (RAKwireless Technology Co. Ltd. 2021, S. 2). Bezüglich des theoretischen Idealfalls einer SWR von 1.0 ist hier daher eine erhebliche Verbesserung erreicht.

Die Impedanz der Antenne zeigt sich abhängig von der Winkelung der Radialen zum Resonator. Dies ergibt sich durch die Änderung der Ausbreitungskonstante und des Nahfeld-Ablöseverhaltens für unterschiedliche Winkelungen. So stellt sich die beste erreichbare Lösung, unter den getroffenen Voraussetzungen, mit einem Kompromiss dar. Entsprechend des Datenblattes des LoRa-Transceivers (Semtech Corp. 2015b, S. 14) ist eine Antennenimpedanz von 50 Ohm anzustreben. Die hiesige Lösung trägt dieser Forderung Rechnung, indem es gelingt, die gekoppelten geometrischen Eigenschaften so zu parametrieren, dass sich die Impedanz etwas oberhalb 50 Ohm und der Blindwiederstand nahe 0 Ohm einstellt (Abbildung 10).

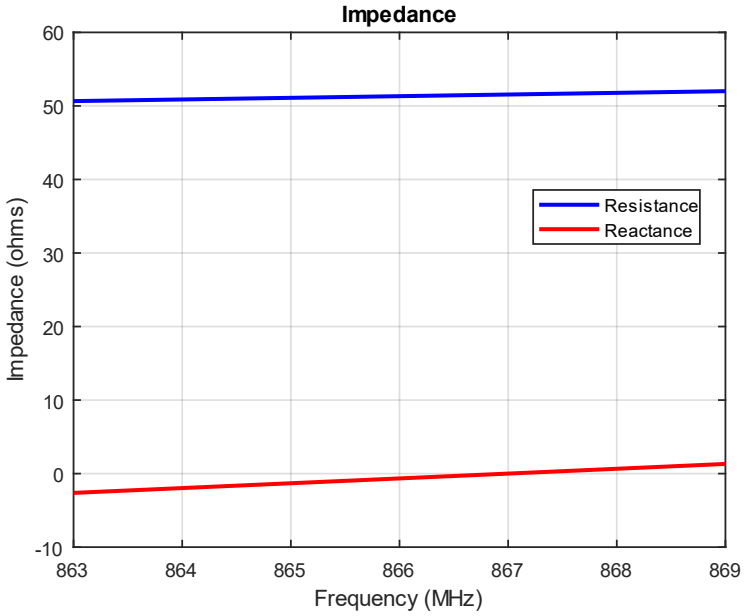


Abbildung 10 Widerstand und Blindwiderstand der Antenne

2.3 Marktstandards auf LoRa-Basis

In diesem Abschnitt wird die Leistungsfähigkeit von am Markt existierenden, LoRa basierten IoT Netzwerken am Beispiel LoRaWAN beschrieben.

2.3.1 LoRaWAN

Um die LoRa-Technik sind mehrere Standards entstanden, die eine Implementierung für kommerzielle und private Anwendungen bieten. Zum Herstellen eines Vergleichs zu der in dieser Arbeit erreichten Lösung „Loridane“ (s. Abschnitt 4 und 5), wird im Folgenden die weit verbreitete LoRaWAN-Lösung von TheThingsIndustries©, namens TheThingsNetwork (TTN), kurz vorgestellt.

2.3.2 TheThingsNetwork und TheThingsStack

Bei TTN ist die LoRa-Technik für ein Triplett aus Hosts, Gateways und Backendservern realisiert und nutzt die Fähigkeit von LoRa, große Distanzen mit verhältnismäßig kleinem Energieaufwand für das Versenden von Nachrichten abzudecken (Lourdes Peñalver et al. 2021, S. 222). Das Funktionsschema ist in Abbildung 11 dargestellt.

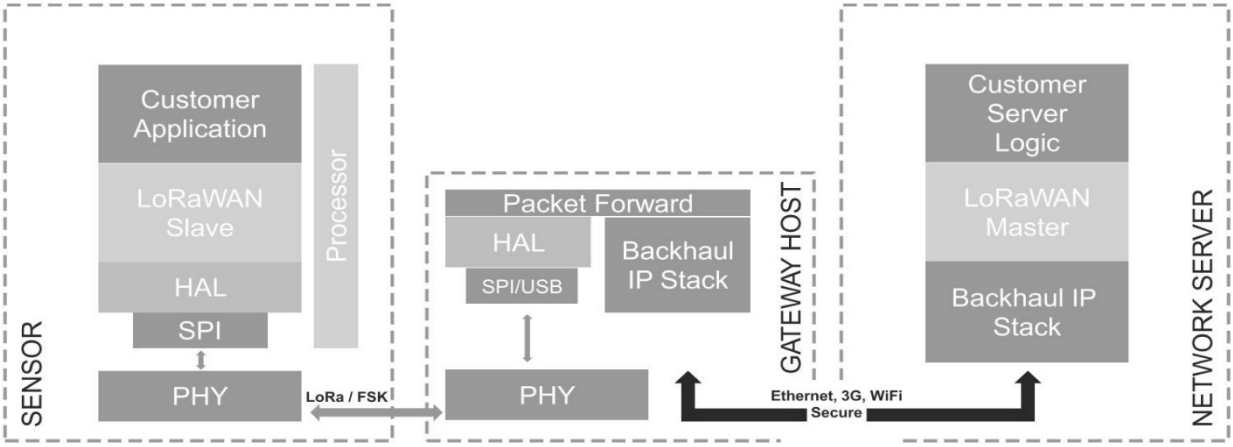


Abbildung 11 Struktur eines LoRaWAN Netzwerkes.jpg (Linnemann et al. 2019, S. 26)

Der Host ist zumeist ein Microcontroller, der durch eine erweiterte Hardware, z.B. durch Anbringung eines, wie in Abschnitt 2.2 beschriebenen SX1276 Chips Datensätze via LoRa versenden und empfangen kann. Diese Endgeräte sind sterntopologisch in Reichweite eines Gateway-Hubs aufgestellt. Im Sinne des zellularen Ansatzes eines Mobilfunknetzwerkes können auch mehrere Gateways Sterntopologien aufspannen, um größere, auch sich gegebenenfalls überlappende Bereiche abzudecken (Linnemann et al. 2019, S. 7). Eine Übertragung des Datensatzes findet statt, indem das Signal vom Gateway, welches als Empfangseinheit dient, transcodiert wird und das Datenpaket über ein IP-Netzwerk ((W)LAN oder WWW) an einen Backendserver weiterleitet.

In Abbildung 11 ist diese Struktur schematisch dargestellt und gliedert sich von links nach rechts wie folgt.

- Sensor:
  Mit dem im vorhergehenden Absatz als Host bezeichneten Gerät werden durch eine proprietäre Software des Herstellers[8] Werte eines Hardwaresensors abgefragt und verwertet, sowie auf eine Datenstruktur transcodiert, die dem LoRaWAN-Protokoll entspricht. Das HAL (Hardware Abstraction Layer) ist ein Programmteil oder eine Softwarebibliothek, die eine funktionserfüllende Ansteuerung des, via serieller Schnittstelle angebundenen, LoRa Transceivers ermöglicht. Auf physikalischer Ebene (PHY, hier LoRa-Funk) wird dann der Datensatz an das Gateway (Abbildung in der Mitte) übertragen.

- Gateway Host:
  Im Gateway wird keine inhaltliche Veränderung des Datensatzes vorgenommen, sondern vom LoRa Transceiver empfangen und auf ein IP-Stack übertragen (z.B. Ethernet, WLAN). Da hier unterschiedliche Protokolle Anwendung finden muss der Nutzwert vom LoRa-Datensatz getrennt und für IP-Netzwerke neu codiert werden.

- Network Server:
  Die Übertragung des Datensatzes findet nun nach dem Internet-Protokoll über das lokale Netzwerk oder das WWW statt. Im Programm des Network-Servers wird die Authentifikation (s.u. DevEUI) des Hosts vorgenommen und dem Datenstrom für die vorkonfigurierte Applikation zugeführt. Die Applikation stellt dem Endbenutzer den Datensatz via eines graphischen Web-Interfaces zur Verfügung.

Hier wird deutlich, dass ein im LoRa-Kontext meist als Gateway bezeichnetes Gerät, nach den formalen Kriterien des OSI-Schichtenmodells (Open Systems Interconnection, ISO/IEC 7498-1:1994) keine Gateway-, sondern eine Hubfunktionalität bereitstellt, da es bereits auf der physikalischen Schicht (OSI1) agiert aber keine - für Gateways typischen - Verwaltungsaufgaben auf der Sitzungsschicht (OSI5) oder einem höheren Layer übernimmt.

Der Datensatz wird auf dem Server decodiert. Zum einen von einer binären Codierung in Cayenne Low Power Payload (myDevices Inc. 2021) auf Base64 und zum anderen wird die AES128-Verschlüsselung (NIST Computer Security Division (CSD) 2001) aufgelöst. Letzteres geschieht durch zwei zentral abgespeicherte Kennzahlen. Die DevEUI (Device Extended Unique Identifier) ist eine weltweit nur einmalig, von der LoRaWAN Alliance vergebene Identifikationskennung mit 24 Byte Länge. Sie wird ergänzt durch einen AppKEY (Application Key), der 16 Byte umfasst und als Vektor für die Nachrichtenver- und Nachrichtenentschlüsselung client- und serverseitig dient. Dazu wird durch den Endbenutzer auf der Website von TTN für jedes oder eine Gruppe von Endgeräten ein AppKEY für Server und Endgerät vereinbart und die DevEUI auf dem Server manuell hinterlegt, um den Client dort zu authentifizieren. Bislang sind etwa 21500 Gateways weltweit online (Stand 14.08.2021). Allerdings sind diese Gateways nicht Eigentum des TTN, sondern gehören gewerblichen oder privaten Betreibern. Diese möchten gegebenenfalls an ihrem jeweiligen Standort eine

[8] meist embedded C/C++, selten microPython

Anbindung an das TTN schaffen, beispielsweise, um eigene LoRaWAN-Geräte an das Netzwerk anzuschließen. Sie müssen dann jedoch gleichzeitig eine Internetanbindung zur Verfügung stellen, die eine Gateway-Server-Kommunikation ermöglicht. Dazu werden die Gateways mit einer, für TTN proprietären, Packet-Forwarder-Software bespielt.

LoRaWAN-Zertifizierte Gateways besitzen zumeist zehn parallele Demodulationspfade. Somit können sie acht Kombinationen für Frequenzen und Spreizfaktoren abhören und auf zwei Kanälen senden.

Endgeräte (end-devices, EDs) senden, bedingt durch ihre Hardwarearchitektur, stets nur auf einer Frequenz und einem Spreizfaktor zu einer Zeit. Sie können die Parameter aber dynamisch zwischen den Übertragungen ändern. Für EDs sind indes drei verschiedene Geräteklassen vorgesehen:

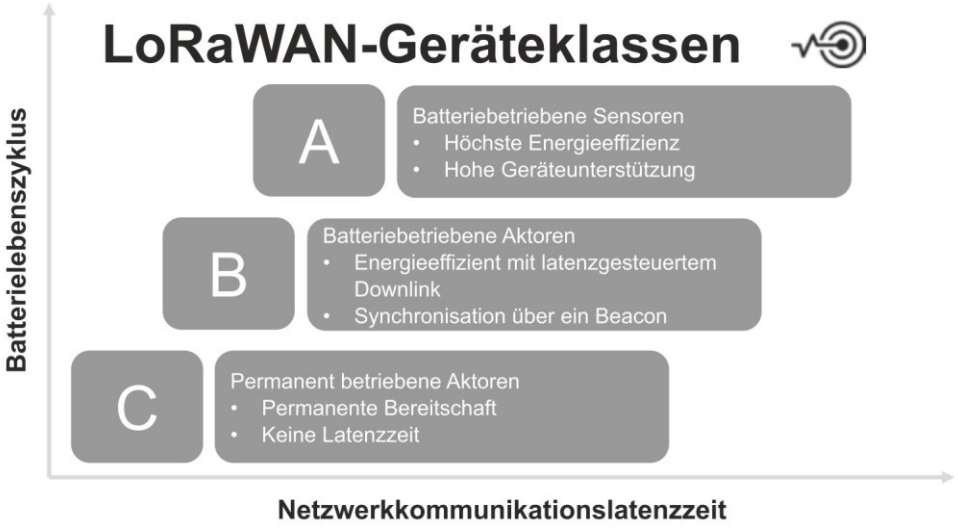


Abbildung 12 Übersicht LoRaWAN Geräteklassen (Linnemann et al. 2019, S. 27)

Class A- Devices:
Class A- Devices schalten den größten Teil der Zeit auf einen Sleep-Mode, während dessen ein sehr geringer Betriebsstrom von einigen Mikroampere typisch ist. Zu dieser Zeit können sie weder Sensoren auslesen und weder Nachrichten empfangen noch senden. Sie verbleiben nach dem Absenden einer Nachricht wenige Sekunden im Empfangsmodus, um gegebenenfalls Konfigurationsnachrichten zu erhalten.

Class B- Devices:
Class B- Devices sind darüber hinaus in regelmäßigen Zeitabständen empfangsbereit, auch wenn sie selbst keine Nachricht versendet haben. Dadurch erhöht sich der mittlere Energieverbrauch und die potenzielle Batterielaufzeit verringert sich gegenüber einem Class A-Gerät. Im Gegenzug verringert sich aber die Latenzzeit. Auch Class B-Devices sind nicht geeignet, um Sensoren mit kurzen Intervallen abzufragen oder Nachrichten zu versenden.

Class C- Devices:
Class C- Devices finden Anwendung, wenn für einen dauerhaften Betrieb Netzstrom oder eine Kombination aus Batteriebetrieb und beispielsweise Photovoltaiktechnik zur Verfügung steht, um eine ununterbrochene Energieversorgung zu gewährleisten. Somit können Sensoren in beliebig kurzen Intervallen ausgelesen und häufiger Daten versendet und empfangen werden.

Allen drei Geräteklassen bleibt gemeinsam, dass sie Datensätze nicht zu voneinander abhängig definierten Zeitpunkten absetzen und damit Signalüberlappungen vermehrt auftreten können.

2.3.3 Aspekte des LoRaWAN-Ansatzes

Die Netzwerkarchitektur des LoRaWAN-Ansatzes hat einige entscheidende Vor- und Nachteile, je nach verfolgtem Anwendungszweck. Diese sind zum einen Teil beabsichtigt und bedingen sich durch das gewählte Protokoll, zum anderen Teil begründen sie sich in den Eigenschaften der technischen Basis von LoRa oder sind via media zur Lösung von Zielkonflikten.

Der entscheidende Vorteil eines globalen teiloffenen Multi-Star-Netzwerkes ist, dass jedes Endgerät weltweit LoRaWAN-Nachrichten an jedes Gateway übertragen kann, dass in Signalreichweite ist und der Nachrichteninhalt dem Benutzer über die Web-Applikation zur Verfügung gestellt werden kann.

Die Endgeräte können in vielen Anwendungsfällen mit Batterien betrieben werden, wenn keine kurzen Sendeintervalle oder dauerhaft hohe Sendeleistungen benötigt werden. Durch Techniken zur Implementierung von adaptiven Datenraten und Sendeleistungen der EDs wird über dies ein Großteil des Potentials der Kanalbelegung genutzt. Darüber hinaus unterstützt das LoRaWAN-Protokoll das Frequency Hopping, was den EDs die Möglichkeit einräumt die Sendeträgerfrequenz dynamisch zu ändern.

Hardware-Entwickler, die EDs nach den Standards der LoRaWAN-Alliance vermarkten, bieten dem Endnutzer einen verhältnismäßig einfachen Einstieg, da die EDs vorkonfiguriert sind und sich in Betrieb nehmen lassen, ohne selbst Software-Code schreiben zu müssen. Es muss allerdings in der Webapplikation von TTN ein JavaScript-basierter Decoder eingepflegt werden, um die Datenpakete auszuwerten. Diesen geben die Hersteller der EDs jedoch zumeist vor. Die Einrichtung eines Gateways hingegen, ist ohne zumindest grundlegende Kenntnisse in der Verwendung kommandozeilenbasierter Betriebssysteme kaum möglich, da diese im Regelfall auf einem, via Secure-Shell (SSH) verwalteten, Linux Computer basieren (Headless[9]-Betrieb).

Entscheidende Nachteile ergeben sich invers aus den Vorteilen. Datenraten oberhalb von 50 kBit/s sind mit LoRa nicht zu bewerkstelligen. Echtzeit-Anwendungen sind nur mit Klasse-C-EDs zu realisieren, diese sind durch eine feste Stromversorgung charakterisiert. Eine Verwendung der LoRa EDs ist nicht ohne Gateways möglich und eine Internetanbindung dieser ist – ohne Weiteres – obligatorisch. TTN führt in seiner Dokumentation darüber hinaus einige selbst auferlegte Einschränkungen auf. Die Verwendung von serverseitig innervierten Aktoren ist kaum erwünscht,

[9] ohne direkt angeschlossene Ein-/Ausgabe-Peripherie

da die Dokumentation von TTN explizit zum „unterlassen [von down-link-Nachrichten[10]], wenn möglich“ (The Things Industries 2021) auffordert. Dies gilt im selben Kontext auch für „Confirmations“, also für gekoppelte bidirektionale Kommunikation zur Empfangsbestätigung. Darüber hinaus beschränkt eine „Fair-Use-Policy“ (The Things Industries 2021) den Einsatz von EDs auf eine Übertragungszeit von bis zu 30 Minuten pro 24 Stunden.

LoRaWAN EDs versenden ihre Nachrichten auf Basis des UDP (User Datagram Protocol). Das entbehrt zwar der Notwendigkeit die EDs aktiv mit dem Netzwerk zu verbinden, nachdem sie einmalig admittiert wurden, sorgt aber auch für eine Ungeregeltheit des Datenverkehrs. Steigt daher im Abdeckungsbereich eines Gateways die Anzahl der EDs, steigt damit auch die Wahrscheinlichkeit der Überschneidung von Nachrichten. Dabei ist beachtenswert, dass dies nicht nur die eigenen EDs berücksichtigt, sondern auch solche die ein Dritter im Empfangsbereich des Gateways betreibt.

Magrin et al. haben LoRaWAN Übertragungen anhand verschiedener Parameter untersucht und dabei die Übertragungssicherheit in Abhängigkeit der Anzahl von EDs in Reichweite eines LoRa-Gateways quantifiziert.

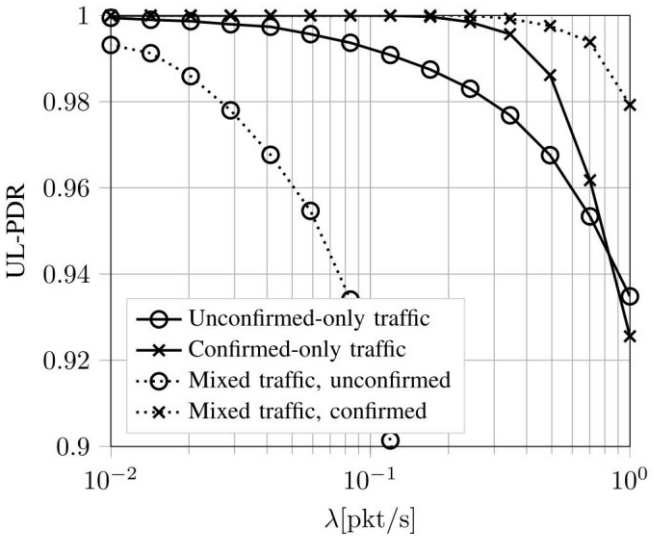


Abbildung 13 Paketübertragungssicherheit nach Rate (Magrin et al. 2019)

Mit der zweiten Kurve von unten ist der prozentuale Anteil erfolgreicher Übertragung gegen die zeitliche Häufigkeit für nicht konfirmierte Nachrichten mit einer Nutzwertgröße von 54 Byte aufgetragen. Der sich aus der Statistik der zu erwartenden Signalüberlappungen ergebende logistische Verlauf, weist bei einer Nachrichtenrate von einer Nachricht pro Sekunde, die durchschnittlich innerhalb der Empfangsreichweite eines Gateways versendet wird, noch eine UL-PDR (upload packet delivery ratio) von etwa 93.5 % auf. Dies entspricht einem Paketverlust von 6,5 % auf einen Gateway-Hub mit 8 parallelen Empfangspfaden (entspr. Uplink-Pfade) in 3 Kanälen. Für „unconfirmed-only traffic“ besteht weder eine Priorisierung für den Nachrichtenempfang noch für das Nachrichtenversenden auf dem Gateway.

[10] Nachrichten die vom Server an das ED gerichtet sind

Treffen zwei Nachrichten zeitgleich bei einem Gateway mit einem Kanal ein, kann keine pauschale Aussage darüber getroffen werden ob, und falls, welche Nachricht vom Gateway empfangbar ist. Dies ist von den Faktoren

- Spreading Factor,
- Übertragungsfrequenz,
- Empfangene Signalleistung, und damit
- Entfernung zwischen ED und Gateway,
- Aktivierung der adaptiven Datenrate (ADR),
- bauliche Gegebenheiten und dem
- Aufstellungsort

abhängig.

Unter der Annahme, dass ein Gateway auf zwei Frequenzen konfiguriert ist, auf welchen Nachrichten eintreffen, entsteht *keine* Kollision. Auch, wenn dies auf zwei verschiedene SF bei gleicher Frequenz zutrifft. Die ganzzahligen SF sind mit Bedacht so gewählt. Sie sind „inhärent Orthogonal“ (Semtech Corp.) und ein Fremdsignal tritt für den Demodulator nur als Rauschen auf.

Treffen jedoch zwei Nachrichten auf derselben Frequenz mit demselben SF ein, ist die Empfangbarkeit von der Signalleistung abhängig. Ist der Unterschied größer als 6 dB wird das stärkere Signal empfangen; bei einer ähnlicheren Leistung gehen beide Nachrichten verloren. Dies betrifft also zwei EDs, wenn bei gleicher Sendeleistung eines weniger als doppelt so weit vom Gateway entfernt steht, als das andere. Betrachtet man dazu als Beispiel eine telemetrische Temperaturmessung eines Raumes oder eines wenig dynamischen Prozesses, wäre eine einzelne Nachricht gegebenenfalls entbehrlich. Analysiert man anhand der Daten aber dynamische Systeme oder Prozesse, oder bedarf es der Erfüllung eines anderen datenkritischen Zweckes, etwa das Schalten eines Aktors, die zeitkritische Übertragung des Impulses eines Bewegungsmelders oder die engmaschige Beobachtung des Energieverbrauchs einer Maschine, kann ein anteiliger Verlust von Nachrichten, der die Datenintegrität stört, zum Verlust ganzer Datensatzfolgen führen.

2.4 Energiemesstechnik auf LoRaWAN Basis

Derzeit stehen bereits verschiedene Geräte am Markt zur Verfügung, die zur Einbindung von Energiezählern in IoT-Netzwerke dienen. Einige Energiemessgeräte sind selbst schon IoT-fähig.

Die weiteste Verbreitung finden hierbei Geräte, die eine Messwertbereitstellung über konventionelle Datenübertragungstechniken erreichen. Die Auswahl an Energiemessstellen, die jüngere Übertragungstechnologien unterstützen, ist ungleich geringer. Im Folgenden soll eine Gegenüberstellung dieser Technologien die bestehenden Möglichkeiten veranschaulichen. Die Spalte „Tauglichkeit“ soll eine Eignung des jeweiligen Systems für die Energiedatenerfassung darstellen. Nach dieser Skala wäre die ideale Übertragungstechnik mit vier Punkten eine drahtlose, mit großer Datenrate und Reichweite, die ohne zusätzliche Empfangshardware auskommt und bei der keine Übermittlungsgebühren anfallen.

2.4.1 Vergleich verschiedener Übermittlungstechnologien

Technologie	Drahtlos	Datenrate	Reichweite	Übermittlungskosten	Hub	Tauglichkeit
Ethernet	--[11]	++	100m	++	++	2
WLAN	++	++	20m	++	++	2.7
Modbus	--	++	1000m	++	-	1.6
Wireless MBUS	++	-	20m	++	--	1.5
ZigBee	++	+	30m	++	--	1.8
Bluetooth	++	+	10m	++	-	2
GSM	++	+	1000m	--	++	2
NB-IoT	++	+	10km	--	++	3.2
LoRa	++	0	10km	++	--	3

Tabelle 2 Gegenüberstellung einiger Datenübertragungstechniken

Für den Einsatz von IoT-Technik zum Zweck der Energiemessung ist, aufgrund des Datenaufkommens und der Kürze der Übermittlungsintervalle, den Technologien der Vorzug zu gewähren, die keine Kosten für einzelne Datenpakete verursachen. Darüber hinaus spielt zur Abdeckung größerer Industrieanlagen die Reichweite eine entscheidende Rolle. Da die Datenpakete mit einigen Bytes recht klein sind (vgl. 5.5.1.3), wäre der Mehrwert durch große Datenraten nicht herausragend. Hingegen dessen ist die Möglichkeit einer drahtlosen Anbindung der Messstellen beachtenswert, da die zu verlegende Kabellänge bei drahtgebundenen Systemen mit jeder Messstelle zunimmt.

Aufgrund der weiten Verbreitung von IP basierten Netzwerken ist es erwartbar, dass am Markt Messtechnik mit WLAN- und/oder Ethernet Anschluss verfügbar ist, zum Beispiel [PL13]. Die in Abschnitt 2.2.2 getroffenen Aussagen zur Wahl des Frequenzbandes für IoT-Applikationen finden hier allerdings weiterhin Beachtung. Sie haben uneingeschränkte Gültigkeit für ZigBee und Bluetooth, da diese Geräte ebenfalls das 2.4Ghz-Band nutzen (IEEE 802.15).

Die Narrow Band (NB)-IoT Architekturen werden von kommerziellen Netzbetreibern zur Verfügung gestellt, daher fallen hier Folgekosten, per Verbindung oder Tarifweise, an. Für GSM basierte Dienste wird das bereits ausgebaute Mobilfunknetz verwendet. In jedem Fall ist es jedoch nicht ohne weiteres möglich, ein unabhängiges, privates System ohne Kommunikation über das Internet zu implementieren. Für die hiesigen Zwecke kondensiert sich daher die Auswahl auf die IoT-Sensorik im lizenzfreien SRD-Band.

[11] -- entspricht -2, ++ entspricht +2, Alle Werte wurden zum Errechnen der Versatilität auf das Intervall [0; 4] normiert und anschließend arithmetisch gemittelt, wobei die Reichweite doppelte Gewichtung hat

2.4.2 LoRa-fähige Energiemesssyteme und Erweiterungen

Hersteller	Typ	Funktion	Index	Preis in €, ca.
Adeunis	ARF8200AA	Stromsensor	[PL1]	150
Adeunis	ARF8230AA	Impulszähler	[PL2]	130
Adeunis	ARF8240AA	Modbus Slave	[PL3]	77
Dragino	RS485-LN	Modbus Slave	[PL5]	150
Eastron	SDM630-Lora	Energiezähler	[PL10]	430
Elsys	ELT-1-LITE	Impulszähler	[PL4]	270
Heyuan	IOT-L2S-D1	Modbus Slave	[PL6]	-
Heyuan	HYFW	Modbus Slave	[PL7]	-
Lobaro	EDL21	Impulszähler	[PL9]	80
SDC	SDM104E	Energiezähler	[PL8]	eingestellt

Tabelle 3 LoRa-fähige Energiezähler und LoRa-Erweiterungen für Bestandszähler

Beim Gros der verfügbaren Geräte handelt es sich nicht um Energiezähler im eigentlichen Sinn. Zumeist werden reine Strommessgeräte angeboten, die aufgrund der potenziell schwankenden Netzspannung nicht zur Energieverbrauchserfassung geeignet sind. Andere Geräte sind Ergänzungen zu bestehenden Energiezählern (Tabelle 3 Zeilen: 2-7,9). Wenn diese mit Impulsausgängen – elektronisch oder optisch – ausgerüstet sind, kann eine Erfassung der umgesetzten Energie als Zeitintegral der Leistung erfolgen. Viele digitale Energiezähler schalten zwischen 500- und 5000-mal per Kilowattstunde eine Signalleitung für einige Millisekunden durch[12]. Liegt an dieser eine Spannung an, kann das Signal mit einem Microcontroller impulsweise erfasst werden. Zeitlich etwas höher aufgelöste Messungen lassen Zähler mit einer Modbus (RS485) Schnittstelle zu (Tabelle 3 Zeilen:3,4,7,8). Hier können über ein serielles Signal zumeist Werte wie Momentanleistung, Zählerstand, Netzspannung, Momentanstrom und verschiedene Parameter der Spannungsquelle abgefragt werden.

[12] S0 Schnittstelle – vgl. Norm: DIN EN 62053-31 (VDE 0418-3-31):1999-04

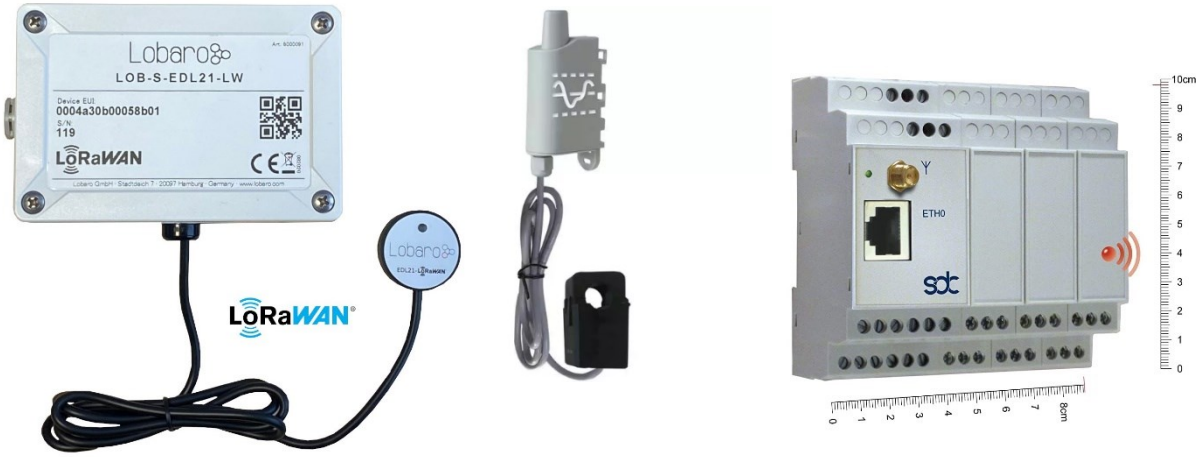


Abbildung 14 LoRa-Erweiterungen für Energiemesssysteme [PL9/PL1/PL8]

Die Auswertung und vor allem die Drahtlose Übertragung des seriellen Datenstroms bedarf dann zusätzlicher Hardware (s. „Host“ 2.3.2).

Produkte die Energieverbrauchsmessung und LoRa-Datenübertragung vereinen sind am Markt selten zu finden (Tabelle 3, Zeilen: 8,10). Diese können mit der Installation von zusätzlichen Gateway-Hubs an LoRaWAN-Netzwerke wie TTN/TTS, Loriot.io, et cetera (s. 2.3.1) angebunden werden.

2.4.3 MID Zertifikation

Ein Vorteil bei der Verwendung von Energiezählern, wie sie üblicherweise für Unterzählschränke oder Hausanschlüsse verwendet werden, ist, dass sie als geeicht mit einer MID (Measurement Instruments Directive) -Zertifikation erhältlich sind. Diese geht auf die EU-Direktive 2014/32/EU zurück und regelt die Anforderungen an Messgeräte verschiedener Art, darunter auch elektrische Wirkverbrauchszähler. Im Sinne dieser Richtlinie gelten solche Zähler als für eine Verbrauchsabrechnung hinreichend genau. Sie sind daher mit höheren Anforderungen an ihre Genauigkeit beaufschlagt als andere marktübliche Messeinrichtungen und zu diesem Zweck in die drei Genauigkeitsklassen A, B und C mit aufsteigender Genauigkeit aufgeteilt.

Die Genauigkeit der Zähler wird nach der Breite der Fehlergrenzen bei Messung der Stromstärke festgelegt:

Zählerklasse	A	B	C
$I_{min} \leq I \leq I_{tr}$	3.5%	2%	1%
$I_{tr} \leq I \leq I_{max}$	3.5%	2%	0.7%

Tabelle 4 Genauigkeitsklasse MID Zählwerke +5°C bis +30°C (Richtlinie 2014/32/EU, S. 210)

Dabei ist $I = I_{min}$die Stromstärke oberhalb derer die Abweichung gerade in der maximal zulässigen Fehlergrenze liegt, ab I_{tr} liegt die Fehlergrenze gerade innerhalb der minimalen Grenze für die angegebene Klasse, bei $I \geq I_{max}$ liegt die Fehlergrenze außerhalb des zulässigen Bereichs (Richtlinie

2014/32/EU, S. 209). Es wird nur die Wirkleistung erfasst, wobei die Ist-Netzspannung um 10%, die Netzfrequenz um 2% abweichen darf.

Der maximal zulässige Messfehler berechnet sich nach

$$e_{ges} = e(I) \cdot \sqrt{\left(\delta e^2(T, I, \cos\varphi) + \delta e^2(U, I, \cos\varphi) + \delta e^2(f, I, \cos\varphi)\right)}. \tag{2-17}$$

Die Werte für den Wurzelterm sind im Zertifikat vom Hersteller des Messwerkes tabellarisch für verschiedene Temperaturbereiche und Leistungsfaktoren in Abhängigkeit des Phasenwinkels φ anzugeben.

So kann beispielsweise für einen Zähler der Genauigkeitsklasse B, hier einem Wirkverbrauchszähler der Typs ONRO OR-WE-520 [PL14], bei einem maximalen Fehler der Strommessung von 2%, der bei schwellender Temperatur, schwankender Messpannung und schwankender Netzfrequenz (Worstcase) und einem Leistungsfaktor von 1 betrieben wird ein Einflussfaktor von 0.39 angenommen werden (ORNO-Logistic Sp. Z O.O 2014, S. 6). Die größte zulässige Abweichung der gemessenen Wirkarbeit zur realen Leistung betrüge dann bei 30kW: $(0.39 * 2\%) * 30kW = 234W$.

2.5 Voraussetzungen für die Aggregation von Kennzahlen

Unter den in Abschnitt 2.4 beschriebenen Voraussetzungen für die Verwendung von Messtechnik zur Erfassung von elektrischer Wirkenergie, muss die Annahme getroffen werden, dass ausschließlich ein absoluter Zählerstand des Messwerkes oder etwa wattstundenweise Impulse zur Auswertung des Verbrauchs verlässlich zur Verfügung stehen.

2.5.1 Generierung von Rohdaten

Beispielhaft für einen 3- Phasigen Energiezähler ist die Ausgabe eines S0-Impulses pro 1.25 Wh (800 Impulse pro kWh) Energieumsatz. Diese Werte sind geräteabhängig und finden sich als Aufdruck auf dem Gerät (vgl. Abbildung 32). Die Leistungsgröße liegt daher in zeitlich integrierter Form vor, sodass eine Erfassung der mittleren Leistung, von zum Beispiel 30 Watt, alle 150 Sekunden nach

$$P = \frac{1\,imp * 1.25\frac{Wh}{imp}}{\Delta t} \cdot 3.6 \cdot 10^6 \frac{ms}{h} \tag{2-18}$$

mittels dem Differenzenquotient zwischen zwei Zählerständen erfolgt. Dabei ist Δt die vergangene Zeit zwischen dem letzten und dem vorletzten Impuls in Millisekunden und der Zähler stets eine Gerätekonstante. Die erwartbare Zeitdauer zwischen zwei Impulsen bei einer mittleren Leistungsabnahme berechnet sich aus

$$\Delta T = \frac{1}{P \cdot i}, \tag{2-19}$$

im genannten Beispiel zu

$$\Delta T = \frac{3.6 \cdot 10^3 \frac{s}{h}}{30W \cdot 0.8 \frac{imp}{Wh}} = 150s. \tag{2-20}$$

Damit ist sowohl eine Zählerstanderfassung als auch eine Erfassung der mittleren verbrauchten Leistung etwa einmal pro zweieinhalb Minuten möglich.

Betrachte man nun eine Maschine im Betrieb, bei etwa 20 kW elektrischer Leistungsabnahme, sinkt das Erfassungsintervall auf

$$\Delta T = \frac{3.6 \cdot 10^3 \frac{s}{h}}{20000W \cdot 0.8 \frac{imp}{Wh}} = 0.225s = 225ms. \tag{2-21}$$

Demnach verbessert sich die zeitliche Auflösung mit größerer Leistung und die Impulshäufigkeit der Messeinrichtung sollte entsprechend der zu beobachtenden Anlage ausgewählt werden. Stattdessen wäre ein Messsystem zu wählen, das eine andere, leistungsunabhängige Datenanbindung bietet[13]. Allerdings ist es gegebenenfalls nicht notwendig, während der Ruhe- oder Rüstzeit einer Maschine, eine höhere als die angegebene zeitliche Auflösung zu erreichen. Während des Fertigungsbetriebs ist diese dann hoch genug, um betrachtungsrelevante Prozesse zu analysieren. Dazu sollte eine Mindestabtastrate nach dem Shannon-Theorem möglich sein. Dies entspräche mindestens 2 Abtastungen pro Prozess. In jedem Fall ist, durch eine geeignete Programmierung des Mikrocontrollers, eine Einhaltung der geforderten, arithmetischen Genauigkeit trotz Mittelung des Leistungsumsatzes zu gewährleisten.

2.5.2 Berechnung von Kennzahlen

Mit den erfassbaren Daten aus dem vorhergehenden Abschnitt ist eine hinreichend genaue Kenntnis über den kumulierten Energieumsatz und damit auch dem Leistungsbedarf eines Prozesses gewährleistet, sodass durch Speicherung der Daten eine energiebezogene Prozessanalyse a posteriori stattfinden kann. Ziel der Analyse ist die Berechnung und Aggregation von Kennzahlen, die Aufschluss über die energetische Prozessgüte und das Optimierungspotential geben.

Solche Kennzahlen sind zum Beispiel:

- Mittlerer Energieumsatz pro Anlage
  Der mittlere Energieumsatz pro Anlage bildet den Verbrauch einer einzelnen Produktionsanlage über ein beliebiges Zeitfenster ab.

- Kumulierter mittlerer Energieumsatz

  Gemittelte Umsätze werden über einen Anlagenpark oder einen Produktionsabschnitt aufaddiert.

- Energiekosten per Maschinenzeit

[13] Z.B Modbus s. Abschnitt 2.4

Unter Angabe des Stromtarifes werden die Energieverbräuche in Energiekosten überführt.

- Energieverbrauch pro Bauteil

  Mit Berücksichtigung der Produktionsrate wird jedem produzierten Bauteil ein individueller Energierumsatz zugeschrieben.

- Energiekosten pro Bauteil

  Die Berechnung erfolgt analog.

- CO_2 Emission

  Wenn der bezogene Energiemix bekannt ist, wird basierend auf einem Tabellenwerk oder einem Umweltindikator jedem Prozess, Anlage oder jedem Bauteil eine CO_2-Emission zugeschrieben.

- Ruhezeit / Produktionszeit

Letztere Kennzahlen können ohne Berücksichtigung der Produktionsplanung Aufschluss über den Istzustand von Ruhe-, Produktions- und Wartungszeiten der Maschine geben. Eine Erkennung des Anlagenzustandes erfordert allerdings Kenntnis über die typischen Verbräuche, während der Prozesse, die durch Lösung von beispielsweise (informationstechnischen) Klassifikations- / Separationsproblemen gewonnen werden müssen, sofern keine dauerhafte manuelle Erfassung geschehen soll.

Als Kennzahlen sind hier Key Performance Indicators (KPIs) zu verstehen, die sich direkt aus der Kenntnis des Wirkverbrauchs einer Anlage und der Zeitdauer ableiten lassen. Dies kann auch unter Angabe weiterer, einfacher numerischer Parameter oder durch statistische Analysen geschehen. Diese sind von einem Messsystem zu berechnen, abzuspeichern und für weitere Analysen zur Verfügung zu stellen. Darüber hinaus bedingen sie eine Abtastung mit hinreichender zeitlicher und qualitativer Auflösung, um aussagekräftige Datensätze zu erhalten. Der Erhalt dieser aussagekräftigen Daten bedarf eines zuverlässigen Übermittlungsverfahrens, das mit marktüblichen Langstreckenübertragungstechniken schwerlich umzusetzen ist, und der Generierung zuverlässiger Daten durch entsprechende Messtechnik.

2.6 Rechtliche Rahmenbedingungen im SRD-Funkband

Das SRD-Band ist lizenzfrei und kann daher für private und kommerzielle Übertragungen verwendet werden. Dennoch regelt die Bundesnetzagentur für Deutschland die Verwendung und Kanalbelegung in den Frequenznutzungsbestimmungen der (Vfg 12/2020, vormals Vfg 133/2019). Daraus resultierend ergeben sich für einzelne Kanäle zwischen 863 MHz und 870 MHz verschiedene maximale Kanalbelegungen. Diese betragen zwischen 0.1% und 10% je nach Kanal. So wird angegeben, wie lange die Sendezeit gegenüber der Ruhezeit eines einzelnen Netzwerkgerätes oder Senders, gemittelt über eine Stunde, höchstens sein darf (Vfg 12/2020, 225-8). Zwei Subbänder sind für Alarmanlagen reserviert und daher für ISM-Anwendungen (industrial, scientifical, medical) nicht freigegeben und daher in jedem Fall zu berücksichtigen (s. 5.6).

2.7 Zusammenfassung des Forschungsstandes und der Marktsituation

Der in den vorhergehenden Abschnitten beschriebene Stand der Technik zeigt, dass keine auf Langstreckenfunk basierende Netzwerktechnologie am Markt zur Verfügung steht, die eine zuverlässige Signalübertragung unter den im Folgenden genannten Umständen ermöglicht.

Der Signalpfad soll nicht das Internet als obligatorisches Übertragungsmedium nutzen. Dazu gehören auch Übertragungen, die über Mobilfunkprovider abgewickelt werden. Zum einen ist bei Internetübertragungen die Abhängigkeit von einem Drittanbieter gegeben. Dies schlägt sich im Aufkommen von Übertragungskosten nieder oder erschwert erheblich den Einsatz solcher Technologien an Orten, an denen keine Internetverbindung hergestellt werden kann oder aus Datenschutzgründen nicht hergestellt werden soll. Zum anderen ergeben sich hieraus längere Signallaufzeiten.

Sind viele Endgeräte im Empfangsbereich eines Gateways, kann es zu Signalüberlappungen kommen, die einen Signalempfang verhindern. Daher besteht die Notwendigkeit diese zu unterbinden. Die LoRaWAN-Technologie bietet bisher keine zugängliche Implementierung, die dies ermöglicht. Darüber hinaus ist ein inhärenter Zielkonflikt mit der Eigenschaft der LoRaWANs gegeben, dass alle Endgeräte zu nicht exakt definierten Zeitpunkten senden. Dieser Zielkonflikt umfasst die Notwendigkeit mit einer steigenden Anzahl an Endgeräten mehr Gateways lokal aufzustellen, obwohl es bezüglich der Signalreichweite nicht nötig wäre, um die Empfangswahrscheinlichkeit für einen Datensatz zu erhöhen oder sich auf eine begrenzte Anzahl Endgeräte zu beschränken und den Verlust von Datensätzen in Kauf zu nehmen. Mit anderen Funktechnologien wie zum Beispiel WLAN kann ohne zusätzlich angebrachte Empfangstechnik keine omnidirektionale Abdeckung größerer Flächen oder über Gebäude hinweg dargestellt werden. Gleichermaßen ist eine Datenanbindung über drahtgebundene Techniken mit einem höheren Installationsaufwand verbunden und eine Flexibilität, die vergleichbar mit der einer drahtlosen Technik ist, nicht gegeben.

Zum Zweck des Energiemonitorings ist am Markt keine hinreichende Auswahl an Messtechnik verfügbar, die diese Eigenschaften verbindet und erfordert daher eine Entkoppelung von Mess- und Übertragungseinheit, die es aufgrund einer flexiblen Hardwareauswahl ermöglicht, Daten zu erfassen und an eine zentrale Instanz zu übertragen. Eine Umsetzung unter Beachtung der oben genannten Umstände bedarf somit eines neuen Verfahrens. Daraus ergeben sich die Anforderungen an ein Langstreckenfunkübertragungsverfahren für telemetrische Zustandsdaten laut den Beschreibungen im folgenden Abschnitt.

3 Anforderungen an ein LoRa-Messstellensystem

Im folgenden Abschnitt werden die Anforderungen an das Energiemessstellennetzwerk identifiziert und diese in eine Konzeption überführt. Dies geschieht anhand der Aufarbeitung der Erkenntnisse über die Eigenschaften von LoRa(-WAN) und der Kenntnis über die am Markt verfügbaren Geräten aus den vorhergehenden Abschnitten.

3.1 Anforderungen durch die Messstellenumgebung

Bei einem Einsatz von Mikroelektronik unmittelbar an einer Produktionsmaschine können verschiedene Einflüsse durch die Umgebung auf die Elektronik identifiziert werden. Jedoch soll bei den Überlegungen der Umstand mit einbezogen werden, dass abgetrennte Bereiche an einer Maschine zur Versorgung mit elektrischer Leistung vorhanden sein müssen, beispielsweise in Form eines Schaltschrankes oder Sicherungskastens (DIN EN 60439-1 (VDE 0660-500)). Dabei muss gewährleistet sein, dass die vorhandene Leistungselektronik bereits entsprechend ihrer Zertifizierung vor den Einflüssen aus ihrer Umgebung geschützt ist. Des Weiteren sind alle mit Spannung beaufschlagten Kontaktflächen berührungssicher auszuführen (DIN EN 50274 (VDE 0660-514)). Dies gilt ebenfalls für die Versorgungsleitung des Mikrocontrollers.

3.2 Konstruktive Anforderungen

Da in Schaltschränken eine Klickmontage mit tragschienengeeigneten Geräten[14] üblich ist, ist für die Unterbringung des Microcontrollers (MyC) ein Leergehäuse, das mit auf der Tragschiene anzubringen ist, die Lösung der Wahl. Dieses sollte möglichst platzsparend sein und dem MyC dennoch genügend Raum lassen, um die entstehende Wärme so abzuführen, dass sich keine nennenswerte Lebensdauereinschränkung für den MyC ergibt. Hierbei tritt ein Zielkonflikt mit der Antennenanbringung auf. Da Schaltkästen meist aus Metall gefertigt sind, wird dadurch die Reichweite des Funksignals gegebenenfalls erheblich eingeschränkt. Dann ist abzuwägen, wie die Unterbringung der Antenne außerhalb des Schaltschrankes zu realisieren ist.

Um die Energieversorgung für den MyC zu realisieren, ist modellabhängig eine Versorgungsspannung zwischen 3.3V und 7V (ca. 2 Watt) zur Verfügung zu stellen. Ist an vorhandener Elektronik keine Versorgungsspannung dieser Größe abzugreifen, muss ein entsprechendes Netzteil auf der Tragschiene angebracht und mit Netzspannung gespeist werden.

[14] DIN EN 60715:2018-07, TH35-Schiene

3.3 Informationstechnische Anforderungen

Die Datenrate und die Verarbeitungsrate des LoRa Netzwerkes mit allen Komponenten muss so ausgelegt sein, dass sich eine höhere zeitliche Auflösung der zu messenden Größen als mit herkömmlicher Long Range Funktechnik darstellen lässt. Es ist zu gewährleisten, jeden zu beobachtenden Prozess mindestens dreimal abtasten zu können (zwei Messintervalle). Damit entspricht das kleinste Abtastintervall der halben Prozessdauer. Dies ist mit LoRa Technik so umzusetzen, dass die potentielle Paketübertragungsrate und die Übertragungswahrscheinlichkeit höchstmöglich liegen. Der Datenintegrität ist mit einer ausreichend sicheren Verschlüsselung Rechnung zu tragen, um gegebenenfalls versendete sensible Daten vor Lauschangriffen zu schützen.

Um Interferenzen mit anderen Systemen zu vermeiden, die sich in der Nähe im Betrieb befinden, ist zu gewährleisten, dass die Sendefrequenz – innerhalb des zur Verfügung stehenden Spektrums – frei wählbar ist. Eine freie Wahl der Signalbandbreite zu ermöglichen, vergrößert gegebenenfalls die Möglichkeiten mit dem System. Dies würde einen weiteren Freiheitsgrad in der Konfiguration erfordern. Die Sinnhaftigkeit dessen ist daher im Versuch mit einer realen Implementierung zu überprüfen.

Signalüberlappungen sind mit Priorität zu verhindern, da sie einen negativen Einfluss auf die Paketzustellungsrate und damit der Fehlersicherheit des Systems haben.

Die empfangenen Daten sind dem Anwender des Systems zur Verfügung zu stellen. Dazu gehört eine lokale (nicht WWW-basierte) Speicherung der Daten. Dazu ist ausreichend Speicherplatz für einen relevanten Beobachtungszeitraum zur Verfügung zu stellen. Bestenfalls hat der Anwender die Möglichkeit, Manipulationen an relevanten Netzwerkparametern über eine graphische Benutzeroberfläche vorzunehmen. Diese soll Bedienungsfehler möglichst filtern und gleichfalls eine freie Konfiguration ermöglichen.

Die Software-Installation des Grundsystems muss von einem geschulten Anwender binnen einer Stunde zu realisieren sein. Es ist zu gewährleisten, dass das Bespielen der Endgeräte (Nodes/Hosts) mit einer vorkonfigurierten Firmware möglich ist, um sie dem Anwendungsfall anzupassen. Wünschenswert ist eine Konfiguration der Firmware über eine Serielle Datenleitung (USB) oder drahtlos über das Netzwerk selbst. Der zeitliche Umfang für die Installation darf dabei den Aufwand für ein kommerzielles Produkt nicht übersteigen.

3.4 Anforderungsliste

Die sich aus den vorhergehenden Abschnitten ergebenden Anforderungen an das Messtellenkonzept fassen sich in der Tabelle unten zusammen.

Anforderungsliste Loridane				Baugruppe: Node(N), Gateway(G), Server(S) , Alle(A)	
Gliederung	FF BF ZF W	**Anforderungen**			
		Nr.	Bezeichnung	Werte, Daten, Erläuterung	BG
Umgebung	BF0	1	Temperatur	Produktionsumgebung; -10 bis 50°C	N,G
	BF0	2		Büroumgebung	S
	BF0	3	Luftfeuchte	Entsprechend verwendeter Hardware	A
	BF-	4	Bauraum	Hutschienenmontage < 6 TE	N
	BF+	5	Hermetik	> IP 54	N,G
	W	6		Eigensicher	N,G
	FF	7		Berührungssicher	N,G
Konstruktiv	ZF	8	Stromversorgung	via Maschinenversorgung	N
	BF-	9	Datenleitung	< 1m	N
	FF	10	Elektrische Spannung	Logic Level entsprechend MyC Hardware	N
	FF	11		Versorgung über Netz	A
	ZF	12	Antennenposition	integriert in Hutschienengehäuse	N
	ZF	13		an/in Gehäuse	G
	ZF	14	Sensorik	Externer Strom und Spannungssensor	N
Informationstechnisch	BF+	15	Datenrate	> 1 Nachricht/min / Node	A
	W	16		adaptiv	N,G
	FF	17	Abtastrate	ausreichend für MID Zertifizierung Klasse B 2%	N
	FF	18	Funkreichweite	ETA abdeckend	N,G
	W	19		> FF	N,G
	FF	20	Genauigkeit	wie MID Zertifizierung Klasse B 2%	N
	FF	21	Datenverkehr	Verschlüsselt AES128	N,G
	FF	22	Frequenz	im Band frei wählbar 863-869MHz	N,G
	FF	23	Bandbreite	statisch	N,G
	W	24		dynamisch	N,G
	FF	25	Sendereihenfolge	asynchron	N,G
	FF	26	Datenspeicherung	lokal	S
	ZF	27	Interface	grafisch	S
	BF-	28	Installationsaufwand SW	<1h	A
	W	29	Firmware Update	OTA	N,G
	ZF	30		Flash by Server, Drahtgebunden	N,G
	FF	31		Manuell via IDE	N,G
§	FF	32	Funkkanalbelegung	Höchstens wie gestattet nach Vfg 133/2019	N,G

Tabelle 5 Anforderungen an ein drahtloses Energiemesssystem

Legende zu Tabelle 5:
FF.: Festforderung,
BF.: Bereichsforderung („+“: Abweichung nach oben ist wünschenswert, „-“ vice versa, „0“ mit festgelegten Grenzen),
W.: Wunsch

4 Konzeption

Im Sinne der Umsetzung eines low-cost Ansatzes soll die, zur Erfüllung der Anforderungen ausgewählte Hardware günstig und leicht verfügbar sein. Es steht jedoch keine Software am Markt zur Verfügung, mit der eine solches Messtellenkonzept umsetzbar wäre. Daher wird hier eine eigene Implementierung vorgenommen (Abschnitt 5).

4.1 Netzwerkstruktur

Die Netzwerkstruktur soll grundlegend der des LoRaWAN-Ansatzes entsprechen. Insofern werden drei Baugruppen entwickelt (s. 2.3.2).

Nodes:

Endgeräte welche die Energiezähler auslesen. Der Datensatz wird aufbereitet und via LoRa versendet.

Gateway-Hub:

Das Gateway empfängt die LoRa Signale der Nodes. Eine spätere Auswertung der Daten und Bewertung der Systemfunktionalität bedarf einiger zusätzlicher Information über den Übertragungsprozess, die das Gateway dem Datensatz hinzufügt, bevor es ihn an den Server weiterreicht.

Server:

Der Server ist das Rückgrat des Netzwerkes und übernimmt alle Aufgaben, die zur Steuerung der Netzwerkfunktionen, automatisch oder manuell, notwendig sind. Dies beinhaltet ebenfalls die Steuerung und Konfiguration der Nodes, das Bereitstellen des User-Interfaces, die Datenverarbeitungslogik, die Datenspeicherung und die Anbindung an das vorhandene IP-Netzwerk.

Abbildung 15 Einfachste Netzwerkstruktur

4.2 Hardwareauswahl

Eine kostengünstige Variante von LoRa-Chips wurde bereits in Abschnitt 2.2 beschrieben. Es sind Entwicklungsplatinen verschiedener Hersteller verfügbar, die sowohl einen Semtech SX1276 LoRa-Chip als auch einen MyC zur Steuerung enthalten. Der SX1276 [PL11] ist im unteren Preissegment angesiedelt, da er zu einer Zeit ausschließlich empfangen oder senden und ausschließlich eine Frequenz und einen Spreading Factor bedienen kann. Damit ist er als Transceiver-Modul in einem Node geeignet, benötigt einen kleinen Bauraum und ist energiesparend. Um die Anforderungen an die Datenverarbeitungsrate zu erfüllen, wird als MyC hier ein ESP32 [PL12] der Firma Espressif verwendet. Dieser bietet zwei getrennte Rechenwerke, wobei eines die Verwaltung des integrierten IP-Stack übernimmt, das ermöglicht, den MyC gleichzeitig mit einem WLAN-Netzwerk zu verbinden und so die für das Gateway notwendige bidirektionale Konnektivität bereitzustellen. Das andere Rechenwerk verarbeitet dabei gleichzeitig das proprietäre Betriebsprogramm. Somit ist auch die Verwendbarkeit als Gateway gegeben und die Anforderungen an die Hardware können durch ein Gerät abgebildet werden. Das erfordert den Zukauf von nur einem Gerätetyp.

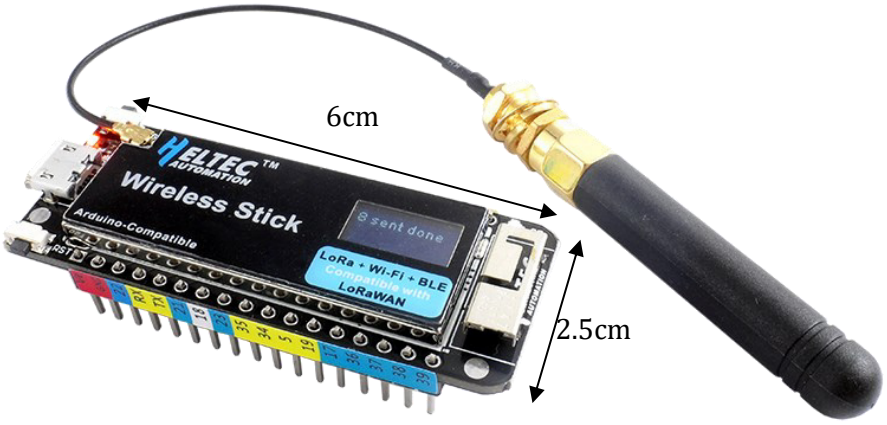


Abbildung 16 Heltec Wireless Stick (Heltec Automation 2021)

In Abbildung 15 ist die LoRa-Kommunikation über blaue, die WLAN-Kommunikation zwischen Gateway und Server über rote Wellen schematisiert. Die Kombination aus ESP32 und SX1276 ist auf Entwicklungsplatinen von mehreren Herstellern verfügbar. Um die bestmögliche Übertragungssicherheit zu gewährleisten und zur Verhinderung von Signalinterferenzen sollten LoRa- und WLAN-Funktionalität voneinander elektromagnetisch entkoppelt sein. Der Hersteller Heltec-Automations führt unter seiner Kernmarke Heltec ein für den hiesigen Zweck sehr gut geeignetes Modul [PL16].

In Abbildung 16 ist ein Heltec Wireless Stick, der im Folgenden, entsprechend seiner Funktion, nur als „Node“ oder „Gateway“ bezeichnet wird abgebildet. Das Gerät findet hier mit unterschiedlicher Softwareimplementierung für beide Geräte als Hardwarebasis Anwendung. Zur elektromagnetischen Schirmung von LoRa- und WLAN-Modul sind die Komponenten unter, auf die elektrische Masse gelegten, Stahlblechen untergebracht.

Die im unteren Bildteil sichtbaren GPIOs sind über die Platine herausgeführte Ein- und Ausgänge des ESP32 MyC. Sie sind softwaredefiniert und können verschiedene Funktionalitäten darstellen. Darunter digitale/analoge Ein- und Ausgänge, sowohl im kontinuierlichen als auch im pulsweitenmodulierten Betrieb, zur qualitativen elektrischen Kapazitätsmessung und in

Kombination zur Kommunikation über verschiedene Standards wie als serieller Master oder Slave, als I²C-Host oder als Bus-Teilnehmer/Host. Des Weiteren steht eine USB-Schnittstelle zur Verfügung, über die sowohl seriell kommuniziert werden als auch die Programmierung erfolgen kann.

Als Server wird ein Raspberry Pi Model 4 verwendet. Dieser wird headless (s. 2.3.3) betrieben und fungiert als logisches Backend der Implementierung. Aus diesem Grund wird keine weitere Hardware benötigt und keine Internetanbindung oder ein externer Serverdienst.

4.3 Funktionelles Konzept

Die Systemsteuerung und damit alle Betriebsfunktionen des Systems werden mit einem nutzereigenen Notebook, Tablet oder Smartphone über das lokale Netzwerk erledigt. Dazu stellt der Raspberry Pi-Server einen gerouteten WLAN-Zugriffspunkt zur Verfügung, über den auch das Gateway verbunden wird. Das ermöglicht eine flexible Aufstellung des Gateways innerhalb der WLAN-Reichweite und gleichzeitig eine günstige Aufstellung in Bezug auf den LoRa-Empfang. Ziel dieser proprietären Implementierung ist es, die Funktionalität des LoRa Netzwerkes von internetbasierten Diensten zu entkoppeln, falls es erforderlich ist, und gleichzeitig die Möglichkeit zu schaffen, alle LoRa-basierten Geräte so zu kontrollieren, dass eine Erfüllung der vorgenannten Anforderungen möglich wird. Dazu sind zeitkritische Prozesse nötig, die mit dem marktüblichen LoRaWAN-Ansatz zum Teil nicht realisierbar oder die zum anderen Teil noch Gegenstand aktueller Forschung sind, z.B. (Polonelli et al. 2019). Diese Ansätze sind in kommerziellen LoRaWAN Anwendungen bisher nicht implementiert[15]. Des Weiteren basieren sie auf einem absoluten (epochenbasierten) Zeitsystem, was bei Netzwerken, die sich topologisch über größere Entfernungen und über das Internet erstrecken, einem großen Aufwand zur Synchronisation der Endgeräte bedarf. Hier soll dem begegnet werden, indem eine Zeitsynchronisation basierend auf einem relativen Zeitsystem eingeführt wird. Dies soll ermöglichen, den Nodes ein Zeitscheibensegment zuzuweisen, indem sie Datensätze übermitteln dürfen und so Signalkollisionen aktiv zu verhindern. Dem liegt die Idee des Time Division Multiple Access (TDMA)-Ansatzes zugrunde. Das Ziel ist es, die Nodes innerhalb eines relativen Zeitsystems (der Zeitscheibe) so präzise miteinander zu Synchronisieren, dass eine Aufteilung der Zeitscheibe in Segmente für alle Nodes und die aktive Zuweisung eines Zeitscheibensegmentes zu jedem Node eine ideale Desynchronisation der Datenübermittlung ermöglicht. Mit solcher Software ausgestattete Nodes sind dann als Impulszähler an einer S0-Schnittstelle (s. 2.4.2) verwendbar. Die Impulse können nicht nur gezählt, sondern auch Rechnungen mit kleinem arithmetischem Aufwand direkt auf dem Node durchgeführt werden. Im Rahmen des dem Node zugewiesenen Zeitslots und nach Konfiguration eines Mindestintervalls sendet jeder Node einen Datensatz an das Gateway-Hub. Der so übermittelte Datensatz wird im lokalen Speicher des Servers abgelegt. Diese Daten stehen dem Nutzer über eine IP-Netzwerkverbindung zur weiteren (automatisierten) Verarbeitung zur Verfügung. Die Konfiguration der Rahmenparameter, darunter die LoRa-Netzwerkparameter und die Art der Datenverarbeitung, legt der Nutzer über ein graphisches Interface fest. Das Interface stellt der Server in Form einer Web-Applikation zur Verfügung.

[15] Stand: September 2021

4.4 Datenübermittlung

Die auf LoRa basierende Datenübertragung zwischen Nodes und Gateways erfordert im Signalpfad zu einem späteren Zeitpunkt eine Instanz, die die Übersetzung des Protokolls und des Signalweges zu serieller oder Internet-Protocol basierter Übermittlung vornimmt. Aufgrund der flexiblen Nutzung soll daher die Kommunikation zwischen Gateway und Server über WLAN erfolgen. Es ist sinnhaft die Übersetzung auf das IP so bald als möglich, nach der LoRa-Übermittlung, vorzunehmen. Eine dazwischen geschaltete Internetverbindung ist fakultativ, darf aber, um den Anforderungen zu genügen, nicht obligatorisch sein. Als Übertragungsziel über WLAN stehen verschiedene Möglichkeiten zur Auswahl:

- POST-Requests an eine vom Server zur Verfügung gestellte REST API (Representational State Transfer Application Programming Interface)
- serielle UDP Datenströme mit proprietärem Protokoll
- Datenstrombasierte Übertragung via z.B. OPC/UA (Open Platform Communication Unified Architecture)
- Publish-/Subscribe-basierte Datenübermittlung durch z.B. MQTT (Message Queuing Telemetry Transfer)

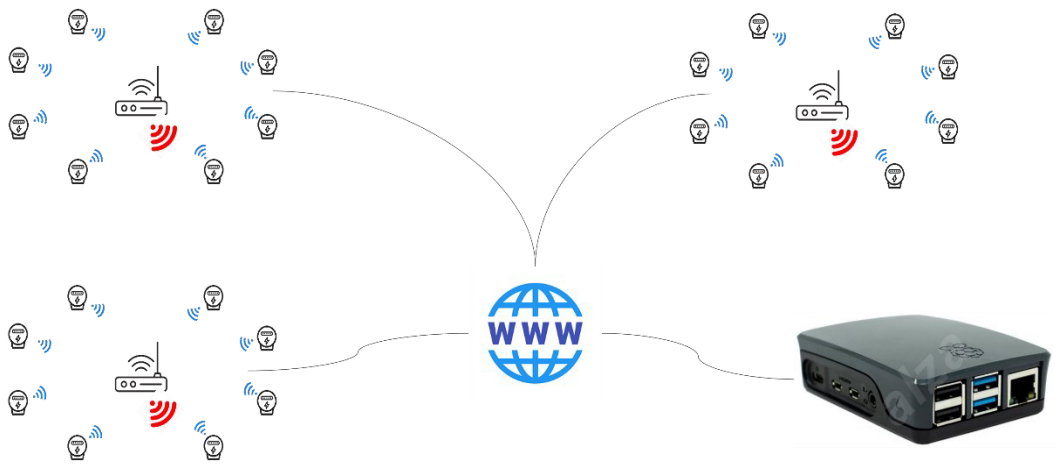


Abbildung 17 Multi-Star-Netzwerk

Eine Implementierung mit MQTT (MQTT-V3.1.1-os) hat, neben der Verfügbarkeit von freien Bibliotheken, vor allem für zu Arduino kompatiblen MyC und der guten Performanz, den Vorteil einen ungerichteten Datenverkehr zu ermöglichen. Somit ist der Datenstrom eines Gateways ohne weiteres Zutun von mehreren Clients empfangbar und umgekehrt der Datenstrom mehrerer Gateways an einen Server zu richten. Dies ergibt sich aus dem Umstand, dass nur ein Server (MQTT-Broker) die Datenströme (Publishings) verwaltet und jeder authentifizierte Client den Empfang des Datenstroms anfordern darf (Subskription). Gleichzeitig wird so ein Multi-Star Netzwerk ermöglicht (Abbildung 17), wobei anstatt des WWW das lokale Netzwerk als Mittler auftreten kann.

4.5 Auswahl der Softwareumgebung

Für die Nodes und Gateways ist aufgrund der vorliegenden Hardwarearchitektur und der Struktur des Betriebssystems der ESP32 MyC eine Implementierung in C/C++ obligatorisch. Die Entwicklung der Firmware kann daher in Arduino erfolgen. Die Arduino IDE (Integrated Development Environment) verbindet die Funktionalität einer Entwicklungsumgebung und der Datenschnittstelle zu den, zu programmierenden, MyC.

Die Software für den Server ist aufgrund der Verfügbarkeit von Laufzeitumgebungen für nahezu alle Programmiersprachen auf einem Raspberry Pi frei wählbar und entscheidet sich nach der Eignung für den hiesigen Zweck und dem Implementierungsaufwand. Im Rahmen des IoT ist serverseitig einer eventgetriebenen Programmierung der Vorzug gegenüber einem klassischen rein prozedural imperativen Programmaufbau zu geben. Denn die Datenverarbeitung erfolgt reaktiv bei Dateneingang. Des Weiteren ist anforderungsgemäß eine graphische Benutzeroberfläche zur Server- und Netzwerksteuerung zu erstellen. Dafür bietet Node.js [SL2] als Laufzeitumgebung und daher eine Codierung des Programms in JavaScript erhebliche Vorteile. Für den Entwicklungsschritt ist es darüber hinaus ein Zugewinn, dass mit NodeRed [SL3] eine Entwicklungsumgebung existiert, die für die Implementierung von eventbasierten JavaScript-Anwendungen im Kontext der Verarbeitung von IoT-Events spezialisiert ist. Für die Erstellung einer graphischen Oberfläche stehen darüber hinaus für NodeRed Module zur Verfügung, die direkt aus der webbasierten IDE einzubinden sind. Somit können alle geforderten Funktionalitäten auf diesem Weg realisiert werden.

5 Implementierung Loridane – Long Range Independent Data Aggregation Network

Dieser Abschnitt beschreibt die Softwareimplementierung für die drei Netzwerkinstanzen Node, Gateway-Hub und Server. In diesem Rahmen wird eine Funktionsbeschreibung der erstellten NodeRed- und JavaScript-Module sowie der LoRa Gerätesoftware und die erforderlichen Schritte zur Installation eines lauffähigen Systems erläutert.

5.1 Verwendete Software

Die Loridane-Software ist für einzelne Betriebsfunktionen auf Bibliotheken und freier Software basiert, die Voraussetzung zur Verwendbarkeit von Loridane sind.

5.1.1 Messstellen (Nodes)

Die Softwareimplementierung der Nodes erfolgt in C/C++. Im Rahmen der vorliegenden Arbeit wird die Entwicklung der Node-Software, das Kompilieren und das Überspielen auf das ESP32 SoC über die Arduino IDE [SL4] vorgenommen. Zur Erfüllung aller Funktionen sind drei zusätzliche C++ -Bibliotheken erforderlich:

- „arduino-LoRa“ [SL5],
- „esp32-encrypt“ [SL6],
- „PubSubClient“ [SL7].

Um die Kompatibilität der Arduino IDE mit dem ESP32 herzustellen, wird darüber hinaus das von der Espressif Corp. zur Verfügung gestellte Hardware-Abstraction-Layer benötigt [SL8].

Das Betriebsprogramm für die Nodes wurde für die vorliegende Arbeit erstellt. Es ist modular aufgebaut und gliedert sich in vier einzelne Funktionsteile:

1. node_S0_loridane.ino[16],
2. credentials.h,
3. functions.h,
4. sensorS0.h.

Modul 1 ist das Hauptmodul des Programms. Hier werden die zum Betrieb notwendigen Module inkludiert, notwendige Variablen definiert und alle nötigen Funktionen initialisiert. Nach Beendigung aller initialen Aufgaben, geht das Programm in eine Endlosschleife über, in der alle Betriebsfunktionen des Nodes ausgeführt werden.

Modul 2 (credentials.h) dient der Initialisierung gerätespezifischer Parameter. Dazu gehört zum einen die Definition der seriellen GPIOs an denen das LoRa-Modul angeschlossen ist, zum anderen das setzen der initialen Sendeleistung des LoRa-Transceivers, der Sendefrequenz und des Spreading

[16] https://github.com/lstebe/loridaneArduino

Factors sowie die Definition der Device UID als einzigartige Hardwareadresse. Entsprechend Abschnitt 5.5.9 wird hier das Passwort für die Verschlüsselung der Nutzdaten gesetzt.

```
// Nutzdaten werden verschlüsselt übertragen, wenn definiert
#define ENCRYPT

// Serielles Debugging ist möglich, wenn definiert
#define DEBUG

// Erhält der Node kein Acknowledgement auf der voreingestellten Frequenz durchläuft
// er einen Funkbandscan, wenn NICHT definiert
#define FREQFIX
```

Code 1 Präprozessorliterale zur Grundkonfiguration (C)

Des Weiteren sind drei Präprozessorliterale nach Code 1 definierbar, die vor Kompilierung der Software grundlegende Einstellungen ermöglichen. Der Compiler lädt nur die notwendigen Bibliotheken auf den Node.

In Modul 3 (functions.h) sind alle Betriebsfunktionen implementiert, die zum laufenden Betrieb, nicht aber für die Initialisierung notwendig sind.

Modul 4 (sensorS0.h) enthält Definitionen und Funktionen die Sensorspezifisch sind und ist wiederum viergeteilt. Um zu gewährleisten, dass zukünftig auch andere Sensoren mit wenig Aufwand einzubinden sind, sind in Modul 4 alle sensorbezogenen Initialisierungen und Funktionen sowie sensorspezifische Koeffizienten und Rechenoperatoren integriert. Damit kann das Sensormodul softwareseitig ausgetauscht und nach der Initialisierung über die Funktionen für das Sensor Setup, das Sensorauslesen und nachgängigen Operationen angesprochen werden.

5.1.2 Gateway

Für das Gateway und die Nodes, dient ein baugleiches ESP32/SX1276-LoRa-Modul als Hardware. Jedoch wird ein anderes Betriebsprogramm verwendet, das aus sieben Einzelmodulen besteht:

1. loridane_gateway.ino[17],
2. init.h,
3. functions.h,
4. credentials.h,
5. receive.h,
6. send.h,
7. serialinput.h.

Die Aufteilung der Module 1-3 nach Betriebsfunktionen ist analog zu den Node-Modulen in Abschnitt 5.1. Die Funktionen unterscheiden sich.

In Modul 4 (credentials.h) werden Konstanten und Variablen definiert, welche die Authentifikation für das lokale WLAN und die Verbindung zum MQTT-Broker, sowie die Adressierung ermöglichen.

[17] https://github.com/lstebe/loridaneArduino

Modul 5 (receive.h) hält die Funktionen, welche zum Empfang von LoRa-Nachrichten und Modul 6 (send.h) solche, die analog zum Versand benötigt werden.

Um das Gateway im laufenden Betrieb auf ein neues WLAN und einen anderen MQTT-Broker konfigurieren zu können, wurde mit Modul 7 (serialinput.h) einen Parser für serielle Datenströme über USB implementiert. Wird das Gateway über seine USB-Schnittstelle direkt vom Raspberry Pi Server mit Strom versorgt, kann so die Datenleitung parallel zur Konfiguration des Gateways verwendet werden und es entfällt die Notwendigkeit, das Betriebsprogramm mit anderen Parametern erneut zu Kompilieren und aufzuspielen.

5.1.3 Software für den Raspberry Pi-basierten Server

Für den, im Folgenden nur Server genannten, Raspberry Pi ist per se jedes Betriebssystem geeignet, das auf einem Debian Linux Kernel basiert. Die Implementierung von Loridane wurde mit einer aktuellen Version, des durch die RasperryPi-Foundation quelloffen zur Verfügung gestellten, RaspberryPi OS Lite [SL9] durchgeführt. Um alle benötigten Funktionalitäten des Servers darzustellen, ist die im Folgenden aufgelistete, zusätzliche Software nötig:

Für den Betrieb des Loridane-Programms:

- Node.js (JavaScript Backend-Laufzeitumgebung),
    - Zuzüglich folgender Module aus dem Node Paket Manager (npm):
        - node-red-dashboard [SL3],
        - node-red-contrib-ui-table [SL3],
        - node-red-contrib-opcua [SL18],
        - cryptojs (Verschlüsselung) [SL2],
        - crypto (Hash-Generierung) [SL2],
- NodeRed [SL3] (Loridane Ausführungsumgebung),
- Loridane[18], die in dieser Arbeit erstellte Serversoftware.

Für die Darstellung der MQTT-Broker Funktionalität auf dem Raspberry Pi:

- Mosquitto [SL10] (MQTT-Broker).

Um den Raspberry Pi wahlweise als WLAN-Accesspoint zu betreiben nach (RaspberryPi Fd.):

- Hostapd [SL11] (WLAN-Zugriffspunkt),
- Dnsmasq [SL12] (DNS- und DHCP-Funktionalität),
- netfilter-persistent [SL13],
- iptables [SL14].

Die Installation eines lauffähigen Gesamtsystems ist mit etwa 80 Einzelbefehlen zur Eingabe im Linux-Terminal, für die oben aufgeführten Punkte, komplex. Daher wurde das Linux-Shell-Skript

[18] https://github.com/lstebe/setupLoridane

loridaneinstall.sh erstellt, das alle automatisierbaren Operationen selbstständig durchführt und nur für die Passworteingabe und grundlegende Konfigurationen einige manuelle Eingaben erfordert.

5.2 Installation der Server Software

Während auf der Node- und der Gateway-Hardware nur ein Systemprogramm in Betrieb ist, ergibt sich für den Server das folgende, schematische Softwaresetup.

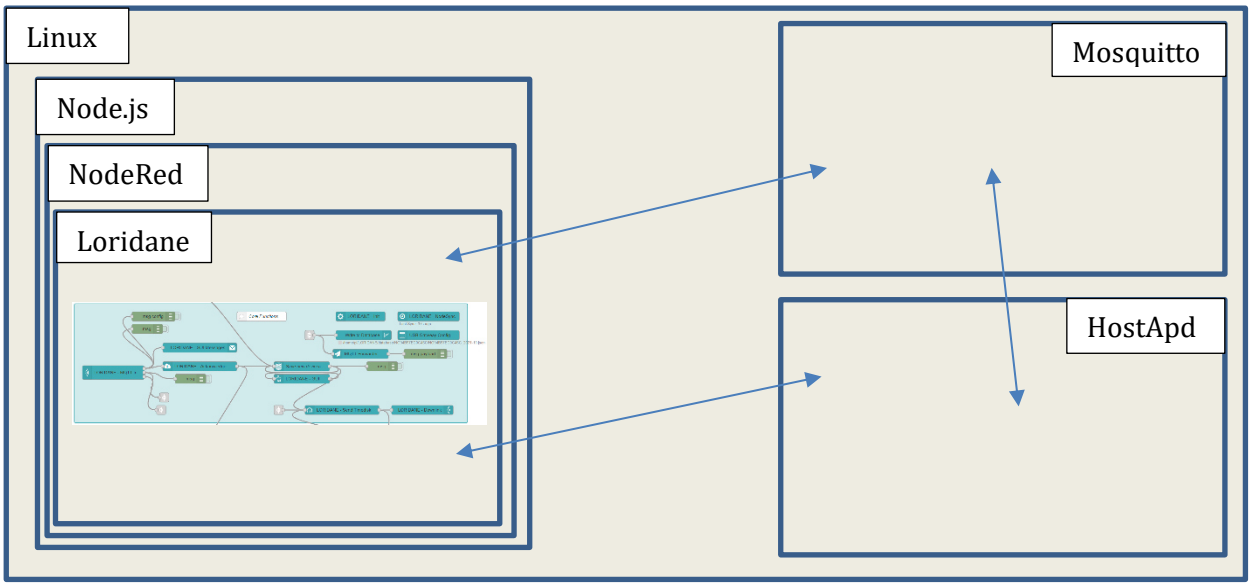


Abbildung 18 Schema der Softwareinstanzen des Servers

Um den Loridane-Server auf einem Raspberry Pi zu installieren, stehen zwei Möglichkeiten zur Disposition:

1. Automatisierte Installation auf einen neu aufgesetzten Raspberry Pi (5.2.1),
2. Import von Loridane in eine bestehende NodeRed Umgebung (5.2.2).

5.2.1 Automatisierte Installation

Requisite:

- RaspberryPi 4,
- microSD Karte >32GB,
- Vorinstalliertes RaspberryPi OS (Lite, bevorzugt),
- LAN-Anbindung und aktivierter SSH-Zugriff oder direkter Terminalzugriff,
- Internetanbindung oder den Installationsordner „setupLoridane“, der alle Installationsdateien enthält,
- ein SFTP-Client Programm, welches Datenzugriff auf den Server ermöglicht.

Annahme für die folgenden 4 Schritte ist die Verwendung des Linux-Standardbenutzers „pi“.

```
#! /bin/sh
## Kann als Skript ausgeführt werden

# Aktuelle Position: Homeverzeichnis: /home/pi
# Der User "pi" muss über sudo -Rechte verfügen

#Installation von git zum Herunterladen von Loridane
sudo apt-get install git -y

#Erstellen einer lokalen Kopie des Loridane Installations-Repositorys
git clone https://github.com/lstebe/setupLoridane.git

## Statt eines git-clone kann der Ordner manuell in das Homeverzeichnis kopiert werden

#Wechsel in den Installationsordner
cd /setupLoridane

#Ausführen des Loridane Installationsskriptes
sudo sh loridaneinstall.sh

# Installation erfolgt automatisch, teilweise mit grafischen Instruktionen, bedarf
aber einzelner manueller Eingaben.
# Ausführungszeit ca. 20 min bei Vollinstallation
exit 0
#---Ende---#
```

Code 2 Installation der Loridane-Software auf dem Raspberry Pi (Shell)

Hiernach ist die Installation serverseitig bereits erfolgt.

Nach dem Neustart müssen in Loridane der Benutzername und das Passwort zur Verbindung mit dem MQTT Broker hinterlegt werden.

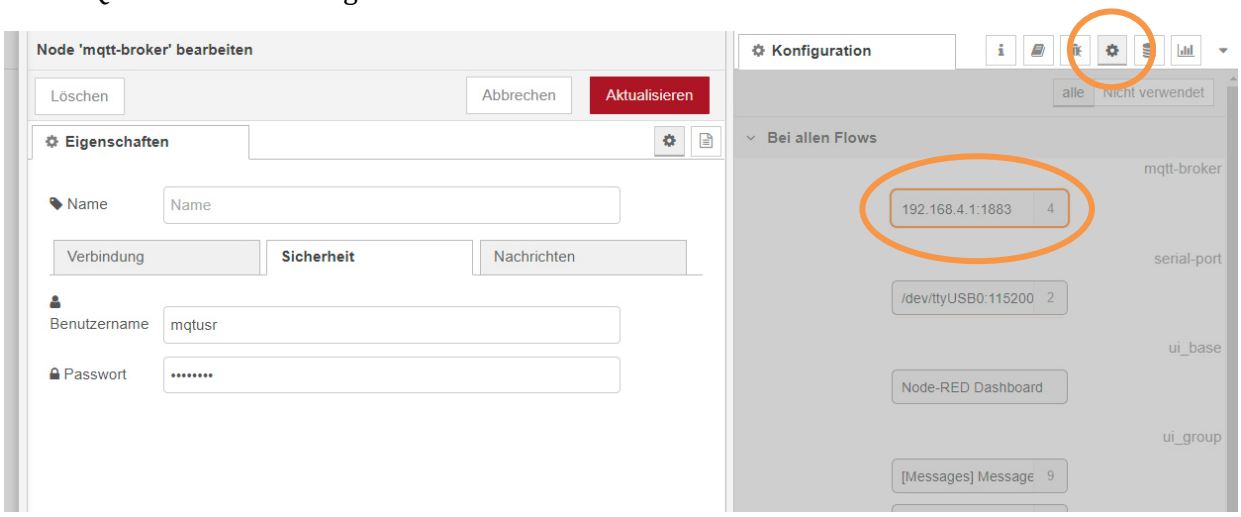


Abbildung 19 Konfiguration von Loridane auf den MQTT Broker

Zum Öffnen der MQTT-Konfiguration wird die Benutzeroberfläche der Ausführungsumgebung für Loridane „Node Red“ geöffnet. Dies geschieht einmalig über einen Webbrowser an einem Desktop-PC.

Es bestehen dazu zwei Möglichkeiten:

Via LAN, im gleichen Netzwerk wie der Server:

- Eingabe in die Browseradressleiste
  http://IP.Adresse.des.RaspberryPI:1880

Via WLAN:

- Verbinden mit dem Access Point des RaspberryPi über die, während der Installation hinterlegten WLAN-Zugangsdaten, z.B.: SSID: „LORIDANE-01“, Passwort: „superSecret123“

- Eingabe in die Browseradressleiste
  http://192.168.4.1:1880
  Wobei anstelle der „4“ (Default-Wert) die richtige, in der Installation vergebene, Subnetzkennung verwendet wird.

Die graphische Oberfläche wird bei der Installation mit einem Passwortschutz versehen. Daher ist auf Nachfrage eine Anmeldung mit

Benutzer: loridaneadmin und
Passwort: loridaneadmin

nötig.

Ebenfalls wird während der Installation für den MQTT Broker automatisch ein Benutzer „mqtusr“ mit den notwendigen Berechtigungen angelegt. Das Passwort vergibt der Anwender während der Installation. Um die Benutzerdaten in NodeRed zu hinterlegen, wird das Konfigurationsmenü in der rechten Seitenleiste geöffnet (Abbildung 19 Konfiguration von Loridane auf den MQTT Broker, kleiner Kreis), und nach Öffnen der MQTT-Konfiguration (Abbildung 19, großer Kreis) Benutzername und Passwort eingetragen. Die Bestätigung erfolgt durch einen Klick auf „Aktualisieren“ und einem anschließenden Klick auf

.

Sodann ist Loridane fertig konfiguriert und lauffähig. Das Browserfenster kann geschlossen werden.

Es ist zu empfehlen, die Zugangsdaten zur Node Red Weboberfläche im Nachgang zu ändern, wie in (OpenJS Fd. 2021b) oder die Oberfläche zu deaktivieren, wie in (OpenJS Fd. 2021a).

5.2.2 Import von Loridane in eine bestehende NodeRed-Instanz

Anstatt der automatischen Installation, besteht ebenso die Möglichkeit Loridane in eine bereits vorhandene Node Red Instanz zu integrieren. In diesem Fall öffnet der Anwender in einem Webbrowser die eigene Node-Red-Administrationsoberfläche (s. 5.2.1) und in einem zweiten Browser Tab die Datei loridaneimport.json aus dem Loridane GitHub Repository[19].

[19] https://github.com/lstebe/setupLoridane/blob/main/loridaneimport.json

Es ist zu beachten, dass Loridane im globalen Kontext des NodeRed-Flows ein Konfigurationsobjekt anlegt. Dieses wird beschrieben und gelesen via:

```
//Lesen
global.get('LORIDANE["<property>"]');

//Schreiben
global.set('LORIDANE["<property>"]',value);
```

Code 3 Globales Kontextobjekt (JavaScript)

Daher ist das globale Keyword „LORIDANE“ für die Verwendung durch Loridane reserviert und freizuhalten.

Für den Import ist es nicht notwendig das setupLoridane Repository gänzlich zu laden, sondern ausreichend innerhalb des Repositorys zur oben genannten Datei zu navigieren und den Inhalt in die Zwischenablage zu kopieren (s. Abbildung 20). Der Import erfolgt dann über das Burger-Menü in der Node Red Oberfläche, im Fenster rechts oben.

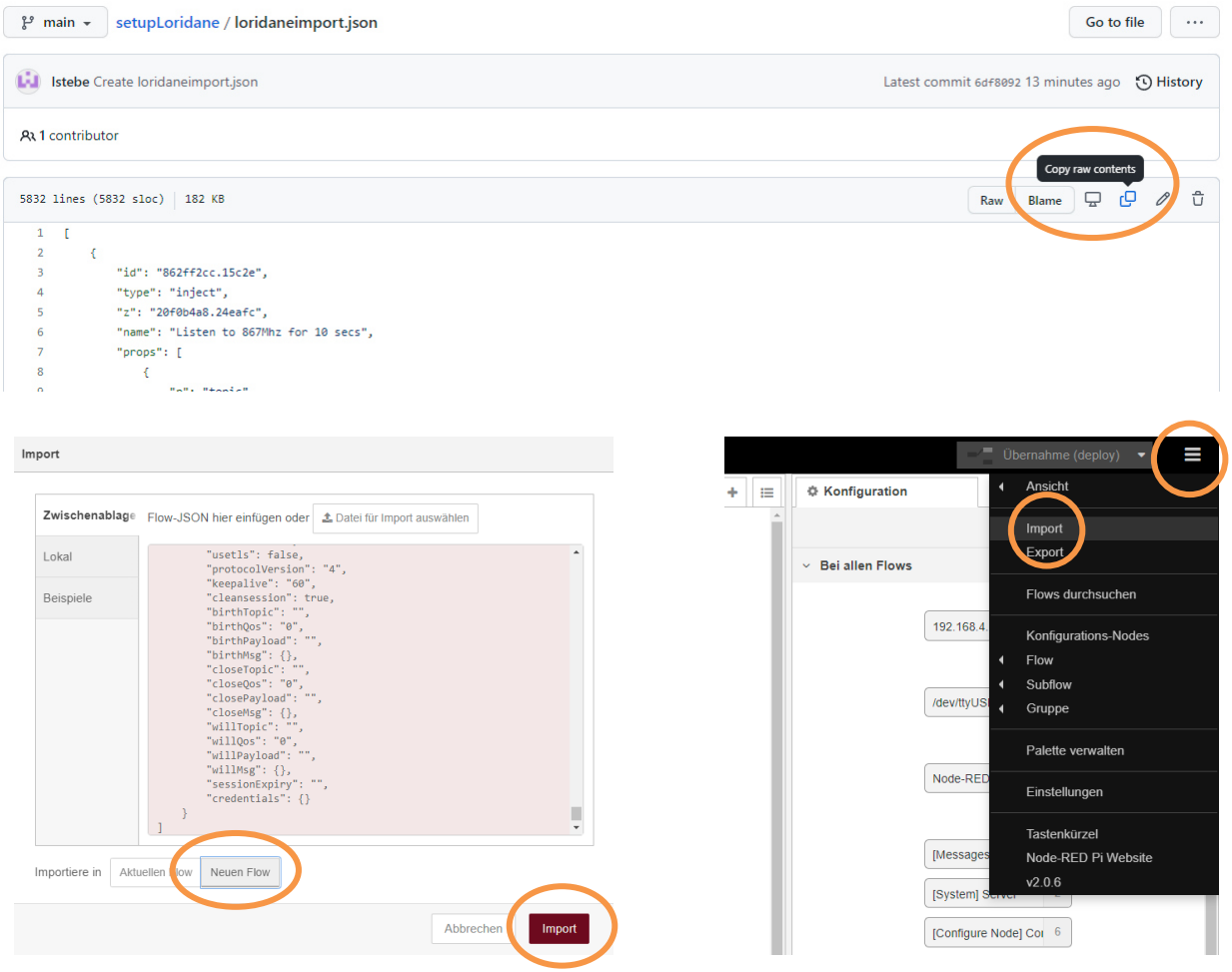


Abbildung 20 Import in eine Bestehende Node Red Instanz

In diesem Fall muss, für die Anbindung des Gateways, ein WLAN-Netzwerk mit Zugriff auf den Server sowie ein MQTT Broker verfügbar sein. Die Konfiguration erfolgt analog zu Abbildung 19.

Alternativ kann das Installationsskript aus (5.2.1) verwendet werden, um auf dem Raspberry Pi nachträglich einen WLAN Access Point und einen MQTT Broker zu installieren. Dazu wird das Skript in gleicher Weise ausgeführt, aber der Anwender überspringt bei Aufforderung die Installation von NodeRed. Die Node.js Modulabhängigkeiten gemäß (5.1.3) sind dann manuell zu installieren.

5.3 Installation der Firmware auf Heltec Wireless Sticks

Zur Installation des Betriebsprogramms auf den Nodes für die Messstellen, wird die Arduino IDE benötigt.

Zur Vorbereitung der auf einem Desktop Computer installierten und geöffneten Arduino IDE sind folgende Schritte vorzunehmen:

- Unter Datei > Voreinstellungen > Zusätzliche Boardverwalter-URLs:
  Hinzufügen des Links für ESP32 SoCs:
  https://dl.espressif.com/dl/package_esp32_index.json.

- Unter Werkzeuge > Bibliotheken Verwalten… :
  Installation von „LoRa“ und „PubSubClient“ (s. 5.1).

- Unter Werkzeuge > Board > Boardverwalter… :
  Installation von „esp32“ von „espressif systems“

Nach dem Setzen der Voreinstellungen ist die Arduino IDE neu zu starten.

Die Betriebssoftware für die Nodes und das Gateway wird ebenfalls über das Loridane GitHub Repository bezogen[20] und auf die lokale Festplatte kopiert. Es enthält die beiden Ordner

- gateway_loridane,
- node_S0_loridane

mit der arduinokompatiblen Software.

Die enthaltene *.ino -Datei ist direkt mit der Arduino IDE zu öffnen. Nach der korrekten Auswahl des Boards „Heltec Wireless Stick“ (Werkzeuge > Boards > ESP32 Arduino) und des USB-Ports (Werkzeuge > Port), kann die Firmware überspielt werden (Sketch > Hochladen).

Vor dem Überspielen der Gateway Software sind in der Datei credentials.h die Zugangsdaten bei den Variablen

- ssid,
- password,
- mqtt_server,
- mqtt_usr und mqtt_pwd einzutragen.

[20] https://github.com/lstebe/loridaneArduino

Vor dem Überspielen der Node Software ist unter credentials.h die variable aes_key mit einem 16-stelligen alphanumerischen Passwort zur Verschlüsselung der Nutzdaten abzulegen.

5.4 Elektrische Beschaltung der Heltec Boards

Im folgenden Abschnitt wird die anwendungsabhängige elektrische Beschaltung des Heltec Wireless Stick (HWS) als Gateway und als Node beschrieben.

5.4.1 Beschaltung als Gateway

Wird das Heltec Board als Gateway verwendet, findet die Kommunikation ausschließlich drahtlos statt. Daher bestehen vier Optionen zur Energieversorgung, welche in Tabelle 6 aufgelistet sind.

Schnittstelle	Strombedarf	V_{min}	V_{Typ}	V_{max}
USB	<500mA	4.7	5	6
3.3V Pin	<150mA	2.7	3.3	4.2
5V Pin	<500mA	4.7	5	6
Lithium Batterie/Akku	<250mA	3.3	3.7	4.2

Tabelle 6 Energieversorgung des Heltec Wireless Stick (Heltec Automation 2021)

Bei einer Verwendung des HWS als Gateway wird mit einem Lithium-Ion Akku mit einer nominellen Kapazität von 1 Ah (3.7V), eine Laufzeit von circa sechs Stunden erreicht. Dies ist nicht sinnvoll und soll daher nicht weiter betrachtet werden.

Für das Gateway ist eine Energieversorgung mit einem handelsüblichen USB-Ladegerät möglich. Andernfalls ist eine Beschaltungsmöglichkeit in Abbildung 21 aufgezeigt.

5.4.2 Beschaltung als Node

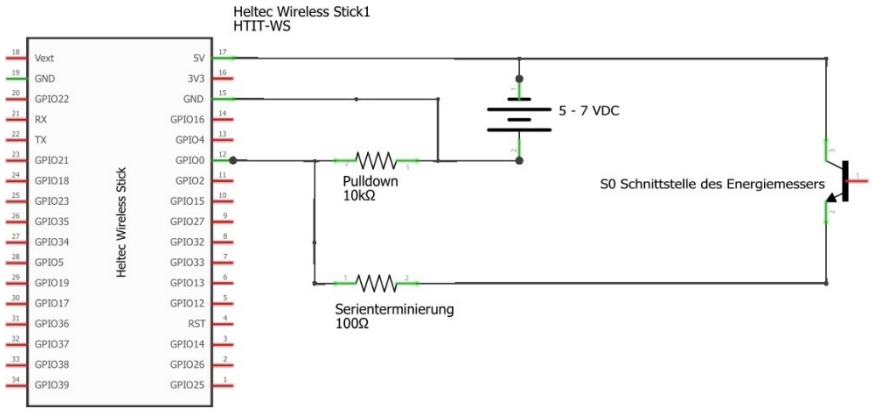


Abbildung 21 Beschaltung des Heltec Wireless Stick

Neben der Energieversorgung ist für eine Verwendung des HWS als Übermittlungssystem an der S0 Schnittstelle eines Energiezählers zusätzlich eine Signalverkabelung nötig, die Abbildung 21 zeigt.

Eine geeignete Versorgungsspannung für die Elektronik ist nach Tabelle 6 per USB oder am 5V Pin des HWS zu gewährleisten. Die Pins für elektrische Masse (GND) und der 5V Pin sind mit der USB-Buchse parallelgeschaltet. Daher kann die Versorgung analog über beide Wege erfolgen. Die S0 Schnittstelle ist als NPN-Transistor auf der Ausgangsseite eines Optokopplers zu verstehen, dessen Kollektor (S0+) und Emitter (S0-) als Signalausgang zur Verfügung stehen. Daher ist die S0 Schnittstelle potentialfrei, sodass S0+ mit der Versorgungsspannung des HWS zu beaufschlagen ist. Der Emitter schaltet daraufhin die Versorgungsspannung bei einem S0 Impuls auf einen freien GPIO Pin durch. So lässt sich der Impuls am HWS als High-Signal detektieren. Um Spannungsschwankungen und damit Fehlauswertungen des GPIO Pins zu vermeiden, wird der Pin (Pin 12 in Abbildung 21) mit Hilfe eines 10kΩ Widerstandes in Pulldownkonfiguration auf Masse gelegt. So ist sichergestellt, dass der Pin auf Massepotenzial liegt, sofern die Schnittstelle keine Impulse emittiert. Der hier abgebildete Pulldownwiderstand kann extern angebracht werden, ist aber im HWS vorhanden und damit auch mittels Programmierung zuschaltbar. Die Loridane Software für Nodes berücksichtigt dies. Zur Eliminierung von Störungen ist zwischen dem S0-Emitter und dem GPIO Pin ein Widerstand zur Serienterminierung eingefügt. Dieser ist nicht immer obligatorisch. Werden jedoch Signalleitungen verwendet die länger sind als nach Gleichung 5-1 berechnet oder ist die Flankensteigung des Impulssignals sehr steil, soll zum einen der initiale Stromfluss limitiert und zum anderen die mögliche Wellenausbreitung im Kabel durch den Spannungsimpuls unterbunden werden. Nach (Hering et al. 2014, S. 633) ist die maximale Leitungslänge für die Signalleitung durch

$$\mathrm{l}_{\max} = \frac{t_s}{2\, t_{pd}} \tag{5-1}$$

gegeben. Nimmt man für die Schaltflankenzeit 5ns und für die Leitungslaufzeit

$$\mathrm{t}_{\mathrm{pd}} = \frac{1}{60\% \cdot c} = \frac{1}{60\% \cdot 30\,\frac{cm}{ns}} = 0.055\,\frac{ns}{cm} \tag{5-2}$$

an, liegt eine Leitung mit einer Länge, die größer als

$$l_{max} = \frac{5ns}{0.055\frac{ns}{cm}} \approx 90cm \tag{5-3}$$

ist, im kritischen Bereich. Daher wird hier zur Serienterminierung ein Emitterwiderstand vorgesehen, der nach (Hering et al. 2014, S. 635) bei etwa

$$\mathrm{Z} = \frac{120\Omega}{\sqrt{\varepsilon_{rL}}} \cdot \ln\left(\frac{D}{d}\right) = \frac{120\Omega}{\sqrt{1.0005}} \cdot \ln\left(\frac{0.005}{0.001}\right) \approx 200\Omega \tag{5-4}$$

liegt, und damit so groß ist wie der Intrinsic-Widerstand des Signalleiters (0.6mm² Kupfer, verdrillter „Klingeldraht“).

Andernfalls könnte die durch den Signalimpuls induzierte Schwellspannung dazu führen, dass Schwingungen im Kabel entstehen, die um die Auswertungsgrenze für ein digitales High- und Lowsignal oszillieren und damit Messfehler verursachen.

5.5 Funktionsbeschreibung der Loridane Software

In diesem Abschnitt werden die Eigenschaften der Loridane-Software für den Server, das Gateway und die Nodes beschrieben. Mit Loridane ist eine Netzwerkarchitektur für LoRa-fähige Endstellen zur Impulszählung an elektrischen Wirkverbrauchsmessern implementiert. Das realisierte System erfüllt die Konzeptionierung aus Abschnitt 4.

5.5.1 Informationstechnische Netzwerkparameter

Für die Umsetzung des Loridane Netzwerkes ist die Einführung einiger Netzwerkparameter essenziell, die für die Kommunikation und Datenspeicherung und -Verarbeitung benötigt werden. Diese Parameter werden im Folgenden eingeführt.

5.5.1.1 Gerätekennung

Den Netzwerkinstanzen Node und Gateway werden Unique Identifier (UID) zugewiesen, um sie vor dem Server zu authentifizieren und die Datenquellen zu identifizieren. Da die hier verwendete Hardware neben dem LoRa- auch über einen WLAN-Chip verfügt, ist sie bereits mit einer MAC-Adresse (Media Access Code) codiert. Diese wird bei der Initialisierung oder bei einem Neustart des MyC aus der Hardware ausgelesen. Je nach Art des Betriebsmodus (Node/Gateway) wird der MAC-Adresse ein NO (Node) oder GW (Gateway) vorangestellt, sodass sich eine UID der folgenden Form ergibt.

```
// Node
NO94B97EC0C46C

//Gateway
GW94B97EC0C31C
```

Code 4 Nomenklatur der UID (txt)

Serverseitig kann so bei jeder eingehenden Nachricht der Gerätetyp und die Geräteidentifikation aufgeschlüsselt werden. Mit dem Präfix (NO/GW) hat eine UID hier stets eine Länge von vierzehn Zeichen. Nach dem Präfix folgt eine ASCII-Zeichenrepräsentation von 6 Byte in Hexadezimaldarstellung (0x00 bis 0xFF) stets ohne Trennzeichen. So wird die via LoRa zu übertragende Datenmenge um redundante Information verkleinert.

5.5.1.2 Standard Parameter

Nodes und Gateways sind bei Betriebsstart auf eine Frequenz von 867.0 MHz und SF 7 eingestellt. So ist gewährleistet, dass alle Geräte auf den gleichen Netzwerkparametern starten. Das Sync-Word (s. 2.2.7) ist in Loridane stets 0xF3 und fest voreingestellt[21]. Die Kanalbandbreite ist für Loridane auf 125 kHz fixiert. Die Coding Rate beträgt stets 4/5.

5.5.1.3 Uplink

Nachrichten, die von einem Endgerät versendet und an den Server gerichtet sind, werden als Uplink-Nachricht bezeichnet. Die Bezeichnung folgt hier so den Begrifflichkeiten von LoRaWANs. Für Uplinks in Loridane ist eine definierte Struktur vorgesehen, um gateway- und serverseitig eine automatisierte Auswertbarkeit sicherzustellen, unabhängig vom Anwendungszweck und des Datentyps. Dazu wurde die Struktur wie im Folgenden beschrieben festgelegt.

```
// Datensatz Uplink allgemeine Struktur
NO94B97EC0C46CDATUM1;DATUM2;DATUMN

//Beispiel NO<UID>Wattstunden;Watt
NO94B97EC0C46C3.75;1282
```

Code 5 Uplink-Protokoll (txt)

Auf die UID des Nodes folgt ohne weiteres Zeichen das erste Datum. Ein Trennzeichen ist hier obsolet, da die Länge der UID für alle Geräte gleich ist. Die Konkatenation weiterer Daten erfolgt durch eine Trennung mit Semikola (;). Per se ist durch diese Form des Protokolls die Anzahl der versendbaren Daten nicht begrenzt. Allerdings muss dem Umstand Rechnung getragen werden, dass LoRa Transceiver höchstens 255 Byte in einem Sendevorgang verarbeiten können. Zusätzlich ist die Loridane-Software so parametriert, dass die Breite eines Übertragungsfensters bei SF7 102 ms beträgt. Theoretisch wäre in dieser Zeit eine Übertragung von 62 Byte möglich. Aufgrund einer zu erwartenden Ungenauigkeit der RTCs (Real Time Clocks) der ESP32 SoCs und dem Berechnungsaufwand, sind an dieser Stelle 20% Vulnerabilitätszeitraum berücksichtigt, sodass unter Realbedingungen 51Byte sicher (s. Abschnitt 5) übertragen werden können. Das obige Beispiel in (Code 5) nutzt diese Kapazität mit 24 Byte etwa zur Hälfte.

[21] Geräte in kommerziellen LoRaWAN-Anwendungen verwenden 0x34 und sind damit von Loridane entkoppelt

5.5.1.4 Downlink

Als Downlinks sind Nachrichten zu verstehen, die vom Server an ein oder mehrere Gateways oder Nodes gerichtet sind. Gründe innerhalb des Netzwerkes die Möglichkeit zu serverseitig initiierter Kommunikation einzuräumen sind zum Beispiel die drahtlose Konfiguration von Endgeräten, das arbiträre Anstoßen einer Sensormessung oder, jenseits der Datenaggregation, die Übermittelung von Steuerbefehlen an Aktorik. Für Loridane wurde ein Downlink-Protokoll implementiert, dass sich an dem Schema für Uplinks orientiert. Alle Downlinks die gleichzeitig mehrere Endgeräte adressieren, entsprechen der folgenden Syntax:

```
// Allgemeine Syntax der Downlink-Flaggen: Parameter >p<, Adressat >a<
pa:value1;pa:value2; …

//Beispiel Frequenz der Nodes auf 867MHz, Spreading Factor 7, Sendleistung auf 14dBm
cn:fn:8670;sf:7;tn:14;
```

Code 6 Downlink Protokoll zur globalen Konfiguration (txt)

Zu diesem Zweck sind auf dem Server und auf den Nodes und Gateways Parser implementiert, welche die Konfigurationsflaggen (Tabelle 7) verarbeiten. Die erste Stelle gibt dabei an, welcher Parameter manipuliert werden soll. Die zweite Stelle steht für den Adressat: Nodes („n“) oder Gateways („g“).

Tabelle 7 Konfigurationsflaggen für Nodes und Gateways

Stelle	Flagge	Wertebereich	Quantelung	Bedeutung
Node	cn:	--	--	Alle Nodes werden konfiguriert
Gateway	cg:	--	--	Alle Gateways werden konfiguriert
Node	fn:	8630 - 8688	1	Frequenz der Nodes (10^5Hz)
Node	sn:	7 - 12	1	Spreading Factor der Nodes
Node	tn:	4 - 20	1	Sendeleistung der Nodes (dBm)
Node	iv:	>5000	1	(Mindest-) Sendeintervall (Millisekunden)
Node	BL:	1 || 0	0,1	Blockiere Sendevorgang auf allen Nodes
Gateway	fg:	8630 - 8688	1	Frequenz der GWs (10^5Hz)
Gateway	sg:	7 - 12	1	Spreading Factor der GWs
Gateway	tg:	4 - 20	1	Sendeleistung der GWs (dBm)

Über eine globale Netzwerkkonfiguration hinaus ist es möglich, einzelne Geräte direkt zu adressieren. In diesem Fall beginnt das Buffer, welches die Konfiguration enthält mit der UID des jeweiligen Gerätes.

```
//Beispiel: Frequenz des Nodes auf 867MHz, Spreading Factor 7, Sendeintervall auf
35 Sekunden
NO84CCA84ADF70fn:8670;sn:7;iv:35000;
```

Code 7 Konfiguration eines einzelnen Gerätes (txt)

5.5.2 Signalpfad Uplink

In Abbildung 22 ist der Signalweg eines Uplinks, respektive eine vom Node an den Server adressierte Nachricht, zu sehen.

Die an der S0 Schnittstelle der elektrischen Wirkverbrauchszähler angeschlossen Nodes werden mit der Firmware auf ein Sendeintervall von 10 Minuten vorkonfiguriert. Detektiert ein Node in dieser Zeit Impulse, werden diese aufaddiert und mit einer Konstanten (Wh / Impuls) multipliziert, um den Wirkverbrauch im betrachteten Zeitintervall zu kumulieren. Aus der Zeitspanne zwischen den eingehenden Impulsen wird die Leistung berechnet. Der Energieumsatz in Wattstunden und die Leistung in Watt versendet der Node via LoRa nach dem Schema in Code 5.

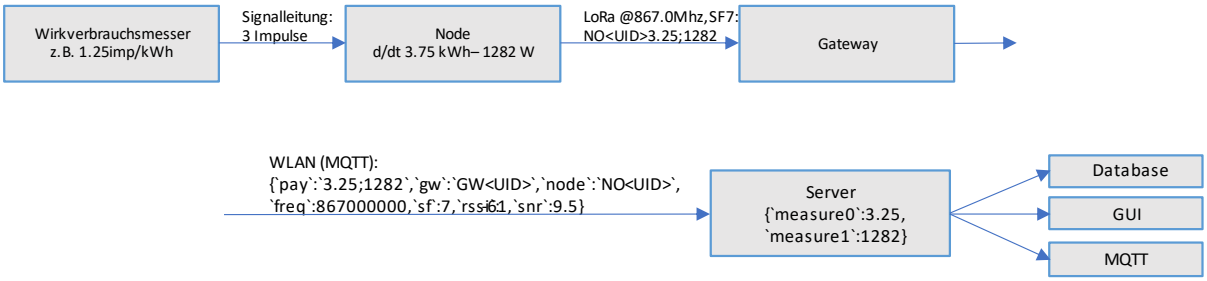


Abbildung 22 Loridane Signalpfad Uplink (größere Darstellung Anhang IV)

Das Gateway empfängt die Daten des Nodes via LoRa. Nach dem erfolgreichen Empfang werden aus dem Transceiver zusätzlich zu den Nutzdaten einige Kennzahlen des Empfangsvorgangs ausgelesen. Im Einzelnen die

- received signal strength (RSSI),
- signal to noise ratio (SNR),
- tatsächliche Frequenz und der
- Spreading Factor (SF).

Auf dem Gateway werden diese Daten in ein stringifiziertes JSON Objekt (JavaScript Object Notation, s. Code 8) zusammengefasst und dieses via WLAN an den MQTT Broker übertragen. Der Loridane-Server, welcher das zugehörige MQTT-Topic lora/gateway/# als Client subskribiert, parst den JSON-String in ein Objekt und kann die Daten so weiterverarbeiten (siehe Programmablaufplan Anhang V).

```
//Stringifiziertes JSON Objekt des Gateways an den Server
{"payload":"NO94B97EC0C46C1.25;56.13","gw":"GWF008D1C8D658","rssi":-63,
"snr":9.500000,"sf":7,"freq":867000000}
```

Code 8 JSON Objekt zur Übermittlung der Uplinks an den Server (JSON)

5.5.3 Signalpfad Downlink

Für Downlink-Nachrichten entspricht das Schema der umgekehrten Reihenfolge wie für Uplinks. Die einzige Ausnahme ist das MQTT-Topic, auf dem der Server an die Gateways publiziert. Die Gateways subskribieren stets ausschließlich das Topic lora/GW<eigeneUID>/#. So kann jedes Gateway einzeln angesprochen werden. Bei Änderungen der Konfiguration durch den Nutzer über das GUI (5.5.10), werden diese durch die Serversoftware automatisch an das korrekte Topic des entsprechenden Gateways ausgespielt.

5.5.4 Datenbank

```
.LORIDANE
├── config
│   ├── loridaneConfig.json
│   └── memCacheLoridane.json
└── database
    ├── NO84CCA84ADF70
    │   ├── NO84CCA84ADF70_2021-10.json
    │   └── NO84CCA84ADF70_2021-11.json
    ├── NO94B97EC0C46C
    │   ├── NO94B97EC0C46C_2021-10.json
    │   └── NO94B97EC0C46C_2021-11.json
    ├── NOF008D1C8D658
    │   └── NOF008D1C8D658_2021-10.json
    └── NOF008D1C8DC20
        └── NOF008D1C8DC20_2021-10.json
```

Code 9 Loridane Ordnerstruktur (txt)

Zur Speicherung der Messwerte und zum Halten der Konfigurationsdaten wird während der Installation ein Verzeichnis im Home-Ordner des Users angelegt. Der Name des Ordners ist stets `LORIDANE/`. Er enthält zwei Unterordner config/ und database/.

5.5.4.1 Initiale Konfiguration

Bei der Installation wird in den Ordner config/ einmalig eine JSON-Datei loridaneConfig.json mit initialen Voreinstellungen kopiert (Anhang I). Bei jedem Start von Loridane werden hieraus die Dateipfade und Sicherheitsschlüssel geladen. Alle Konfigurationen der Endgeräte, sowie Trivialnamen und Verbindungseinstellungen werden in der Datei memCacheLoridane.json gespeichert (Anhang II). Diese wird automatisch erstellt oder neu geschrieben, wenn Konfigurationsänderungen durch den Nutzer erfolgen oder neue Netzwerkgeräte erkannt werden. Hierdurch sind alle Einstellungen auch über Serverneustarts hinaus persistent. Loridane lädt die Datei bei Programmstart oder fällt, falls sie nicht existiert, auf die Basiskonfiguration zurück. Manuelle Änderungen an der Konfiguration vorzunehmen ist möglich. Dies erfordert einen Neustart

von NodeRed und damit der Loridane Software oder ein erneutes Laden der Konfiguration nach 5.5.10.5.

5.5.4.2 Speicherung der Messdaten

```
{
"measure0":"21.25",                                                    z.B.[Wh]
"measure1":"17689.00",                                                 z.B.[Watt]
"time":"4.11.2021,17:03:09",
"timestamp":1636041789128,
"uid":"NO84CCA84ADF70"
}
```

Code 10 Datenbankformat (JSON)

Für jeden Node im Netzwerk wird unter database/ ein Unterordner angelegt, der nach der UID des Nodes benannt ist. Empfängt der Server Daten, werden diese zeilenweise für jeden Monat in eine JSON-Datei geschrieben (s. Code 9). Die Dateien werden automatisiert für jeden Monat erstellt und entsprechend benannt. Existiert eine Datei bereits wird sie nicht überschrieben. Die Daten werden ungeachtet ihres Formats an die Datei angehängt. Nutzdaten, die entsprechend des Uplink-Protokolls mit Semikola getrennt sind, werden nummeriert und in ein Objekt geschrieben. Die Objektklasse zur Datenablage in der JSON Datenbank hat eine Form wie in (Code 10).

Da die Datensätze zeilenweise in eine Datei geschrieben werden, geht der ursprüngliche Charakter einer JSON-Datei verloren. Der Standard (ECMA-404, S. 3) bedingt, dass nur ein Objekt oder ein Array in eine Datei geschrieben werden darf. Eine Folge von Objekten müsste sich daher, um dem Standard zu entsprechen, in einem Array befinden. Für die Datenspeicherung in diesem partikulären Fall ist zeilenweises JSON allerdings konvenient, da nicht komplexe Datenbanksysteme auf dem Raspberry Pi installiert werden müssen, die zusätzlich zur Echtzeitdatenverarbeitung Rechenkapazität beanspruchen. Darüber hinaus kann auf diese Weise in der Datei auf Umklammerung und Trennzeichen verzichtet werden. Hingegen ist für das JSON-Format sowohl eine gute maschinelle Verarbeitung mit JavaScript als auch eine gute Lesbarkeit der Rohdaten für Menschen gegeben. Dies erfordert einen Algorithmus zum Export der proprietären Rohdaten zu anderen Systemen.

5.5.4.3 Export der Messdaten

Um zu gewährleisten, dass die Messwerte auch mit Energie-Effizienz-Controlling-Software (z.B. EnEffCo© [SL15]) oder anderen Datenbanksystemen verarbeitet werden können, ist in Loridane ein Algorithmus implementiert, der das proprietäre Datenformat effizient verarbeitet und für jeden Node als CSV-Datei (comma separated values) exportiert. Die Bezeichnung der CSV-Dateien erfolgt analog zur Bezeichnung der JSON-Dateien in (5.5.4). Ablageort ist der Ordner /database. Dazu wird /database rekursiv nach JSON-Dateien durchsucht und alle gefundenen Dateien exportiert. Die Herausforderung besteht in der Umsetzung der Verarbeitung von Datensätzen mit entsprechendem Umfang auf dem Raspberry Pi 4, sodass die Anforderungen vor allem die Arbeitsspeicherkapazität nicht übersteigen und dennoch eine Konvertierung in begrenzter Zeit erfolgen kann.

Da jede JSON-Datei alle Messwerte eines Nodes für einen Monat enthält, ist zu erwarten, dass sie n Datensätze enthält, mit

$$\mathrm{n} = \frac{2.5 \cdot 10^6 \frac{s}{mon}}{I_{UL}\ [\frac{s}{msg}]} \tag{5-5}$$

und I_{UL}, dem Sendeintervall jedes Nodes. Nun sei der hypothetische Fall eines Sendeintervalls von 1800ms Sekunden je Node angenommen. Dieser darf aus rechtlichen Gründen bezüglich der Allgemeinzuteilung der Kanalnutzung im SRD-Band nicht eintreten und beträgt 200% der erlaubten Kanalbelegung (s. 5.6). Es konnte gezeigt werden, dass der Algorithmus eine JSON-Datei dieser Größe mit entsprechend 1.4 Mio. Datensätzen zu jeweils 5 Datenpunkten[22] in 25 Sekunden verarbeiten kann. Die Zeitkomplexität ist dann direkt proportional zur Anzahl der Datensätze, respektive der Nodes im Netzwerk, wenn der Export einmal monatlich erfolgt. Auf diese Weise kann die Notwendigkeit zur Nutzung einer zusätzlichen Desktop-Software verhindert werden. Die Software wird mit allen Funktionen somit autark auf dem Rasperry Pi betrieben. Ebenfalls ist somit sichergestellt, dass bei Verwendung des empfohlenen Betriebssystems gemäß (s. 5.2.1), alle Funktionen auf dem Raspberry Pi 4 mit 1GB Arbeitsspeicher, dem mithin günstigsten Modell, ausgeführt werden können. Dies ist unter Verwendung eines komplexeren Datenbanksystems nicht darstellbar und wurde im Versuch mit der TSDB (time series data base) InfluxDB [SL16] evaluiert, die mit ihrer Optimierung auf Zeitreihendaten den hier geforderten Funktionsumfang, aber ebenso die Hardwareanforderungen an den Raspberry Pi in Minimalkonfiguration übertrifft, wenn ein CSV-Export wie oben beschrieben stattfinden soll. Dann muss ein Raspberry Pi mit mindestens 2GB RAM verwendet werden.

Standardmäßig verwendet Microsoft Excel [SL17] als Trennzeichen (engl. „delimiter“) für CSV das Semikolon. Für den Datenexport mit anderem Trennzeichen, kann in der Konfigurationsdatei loridaneConfig.json unter `delimiter`: jedes ASCII-Zeichen gesetzt werden.

Ein Export der Daten in Echtzeit ist möglich und standardmäßig aktiviert. Jede Nachricht wird im Format der Datenbankeinträge über das Topic loridane/forwarded/ an den integrierten MQTT-Broker übergeben und ist von dort aus subskribierbar. Darüber hinaus wird mit der Installation ein OPC/UA Client [SL18] zur Verfügung gestellt, der den Port für Echtzeitdaten mit der Weiterleitung via MQTT teilt. Aufgrund der sich in der Art und Komplexität stark unterscheidenden Parametrierung, ist der OPC/UA Client an der entsprechenden Stelle in Loridane[23] vorbereitet, muss aber händisch im NodeRed Editor (5.2.1) parametriert werden.

5.5.5 Confirmations

Bei der Übertragung von Konfigurationen an die Nodes (downlink), kann es durch Übermittlungsfehler dazu kommen, dass der Datensatz nicht korrekt interpretiert werden kann. Um

[22] Dies entspricht 133MB Textdaten

[23] Function ID: c0a3aa7bc8ce6775

zu vermeiden, dass Nodes und Gateway hiernach nicht dieselben Funkeinstellungen verwenden, werden Konfigurationsnachrichten von den Nodes stets mit einem „+“ im Nutzdatensatz konfirmiert. Bleibt die Bestätigung von, dem Server bekannten Nodes aus, wird die Einstellung des Gateways zurückgesetzt und die Konfiguration bis zu dreimal erneut gesendet. Erst dann gilt die Verbindung zum Node als verloren.

5.5.6 Acknowledgement

Nach einem ersten Start oder Reset verfügen Nodes in Loridane über drei hartkodierte Parameter. Diese sind die jeweilige UID sowie die initiale Sendefrequenz und der Spreizfaktor. Die Nodes sind im Server in diesem Moment gegebenenfalls von einer vorherigen Verbindung bekannt. Falls nicht, müssen sie manuell admittiert werden (5.5.7). Nach dem Herstellen der Stromversorgung versenden die Nodes, ungeachtet ihres Admissionsstatus, in Abständen von etwa 3 Sekunden eine Acknowledgement-Anfrage. Diese enthält die UID zwei Mal in der Form:

```
//Acknowledgement-Anfrage eines Nodes
N094B97EC0C46CN094B97EC0C46C
```

Code 11 Acknowledgement (txt)

Der Server überprüft jede eingehende Nachricht darauf, ob der Nutzdatenwert der UID entspricht, ob UID und Nutzdaten gleich sind und ob es sich um eine valide UID handelt (5.5.1.1). Ist die Nachricht somit als Acknowledgementanfrage identifiziert, wird über das empfangende Gateway ein Konfigurationsdatum versendet.

5.5.7 Admission

Ist die UID des Nodes auf dem Server nicht hinterlegt, wird der Anwender über das grafische Interface einmalig informiert und bezüglich der Bestätigung oder Ablehnung des Nodes befragt. Fällt die Bestätigung positiv aus, wird für den Node ein Konfigurationsobjekt angelegt. Andernfalls wird die UID reversibel in eine Ignorierliste verschoben, um zukünftige Anfragen zu verhindern.

5.5.8 Zeitscheibe

Die Implementierung der Zeitscheibe ist eine weitere Eigenschaft, die Loridane von bereits existenten LoRaWAN Lösungen unterscheidet.

5.5.8.1 Zweck der Zeitscheibe

Wie in (2.2.2, 2.3.3) beschrieben, besteht in einer Netzwerktopologie, die auf ungerichtetem Datenverkehr beruht und innerhalb derer die Teilnehmer zu arbiträren Zeitpunkten senden, eine erhöhte Wahrscheinlichkeit, dass sich Funksignale zeitlich überlappen. Durch Interferenzen und der Superposition dieser Signale, sind diese empfängerseitig unter Umständen nicht mehr auswertbar. Die Verteilung von definierten Zeitintervallen an Netzwerkgeräte, um zu gewährleisten, dass die Sendevorgänge stets zeitlich scharf voneinander getrennt stattfinden, ist eine offensichtliche Lösung

für dieses Problem und bereits Gegenstand der Forschung. Beispielsweise beschreiben (Degesys et al. 2007, S. 15) unter anderem eine Möglichkeit zur Implementierung eines „Desync-TDMA“-Verfahrens. Der Ansatz des „fixed-TDMA“ weist dazu jedem Netzwerkgerät ein festes Zeitfenster zu.

5.5.8.2 Ansätze zum Fixed TDMA

Die Zuweisung eines Sendezeitfensters (Slots) kann durch ein absolutes Zeitsystem erfolgen, indem die Nodes in Kenntnis über die aktuelle Epochenzeit und ihren jeweils zugewiesenen Slots sind (Mirabella et al. 2018, S. 112). Für Loridane wird eine andere Lösung gewählt, die berücksichtigt, dass alle Netzwerkteilnehmer nur einen Kanal belegen. Dementsprechend findet zur Verminderung von Overhead und damit zur Verbesserung der Performanz, ein proprietäres, relatives Zeitsystem Anwendung.

5.5.8.3 Voraussetzungen für das relative Zeitsystem

Alle Geräte in Loridane verfügen über eine eigene RTC und damit über die Kenntnis der vergangenen Millisekunden seit ihrem letzten Neustart, nicht aber über die absolute Uhrzeit (Espressif Systems Co., Ltd., S. 26). Hier wird daher ein Algorithmus implementiert, der die Nodes befähigt den jeweiligen Slot in der Zeitscheibe mit wenigen Parametern selbst zu berechnen. Dazu beachtet jede Netzwerkinstanz folgende Regeln, die in der jeweiligen Software hartkodiert sind:

- Ohne Acknowledgement darf ein Node in den ersten zehn Minuten ab Start, ausschließlich Acknowledgementanfragen senden.
- Acknowledgementanfragen beantwortet der Server zum nächstmöglichen Zeitpunkt.
- Eine Neuzuteilung der Slots erfolgt frühestens zehn Sekunden nachdem ein neuer Node admittiert wurde.
- Ein Acknowledgement für einen Node wird ungeachtet dessen Admissionsstatus vergeben, um weitere Sendevorgänge zu undefinierten Zeitpunkten zu vermeiden.
- Jede Instanz in Loridane erhält einen Slot von gleicher Länge.
- Nachrichten, die einen Slot zuweisen, sind stets an eine einzige Instanz adressiert.
- Zeitgebend für die Synchronisation der Instanzen sind Nachrichten vom Server, die entweder
    - Synchronisationsnachrichten oder
    - globale Konfigurationsnachrichten sind.
- Als Synchronisationszeitpunkt gilt der Moment, in dem die Netzwerkinstanzen die Nachricht empfangen und als Konfigurationsnachricht identifiziert haben.
- Eine erneute Synchronisation erfolgt spätestens nach 233 Sekunden.

5.5.8.4 Funktionsweise des Zeitscheibensystems

Der Server ist als einzige Netzwerkinstanz über die Anzahl der verbundenen Geräte in Kenntnis. Daher werden die Slots durch den Server vergeben. Zuzüglich einer Vulnerabilitätszugabe vergibt der Server Slots für den Versand von 51 Bytes je Gerät. Die Länge dieses Zeitintervalls wird nach Gleichung 2-13 berechnet. Es werden dazu die Anzahl der bekannten Nodes und Gateways addiert und mit der Sendezeit je Gerät multipliziert. Die Zeitscheibe wird stets vollständig ausgespielt. Der Versand erfolgt je Node über das Gateway, welches als Letztes eine Nachricht des Nodes empfangen oder beim Empfang das stärkste Empfangssignal hatte.

```
// Ausrollen der Time Disk für 3 Nodes und 2 Gateways
// Protokoll: NO<UID>td:<index>;<gesamte-Zeitscheibenlänge>;

// Topic: lora/GW94B97EC0C31C
cn:BL:1
NO94B97EC0C46Ctd:0;509;
NOF008D1C8DC20td:1;509;
NO84CCA84ADF70td:2;509;
GW94B97EC0C31Ctd:3;509;
GWF008D1C8D658td:4;509;

// Topic: lora/GWF008D1C8D658
cn:BL:1

// Topic:  lora/GW94B97EC0C31C
cn:BL:0

// Topic: lora/GWF008D1C8D658
cn:BL:0
```

Code 12 Ausrollen der Time Disk (txt)

In Code 12 ist das Gateway mit der Kennung GWxxxx1C beispielsweise im Server für alle Nodes als das nächstgelegene Gateway hinterlegt. Daher erfolgt der Versand der Zeitscheibe ausschließlich über das zu diesem Gateway gehörige Topic. Da es sich bei den Nachrichten für die Zeitscheibensegmente nicht um globale, sondern um individuelle Konfigurationen handelt, werden diese nicht als Synchronisationsnachricht interpretiert. Anschließend werden alle Nodes mit einer globalen Nachricht ("cn:" - **c**onfigure **n**odes) über beide Gateways neu synchronisiert.

Das Ausspielen der Zeitscheibe ausschließlich innerhalb der Gateway Downlink-Slot würde für eine große Anzahl Nodes sehr viel Zeit in Anspruch nehmen. Daher wird für diesen Zweck der Nachrichtenversand auf allen Nodes mit dem Befehl cn:BL:1 unterdrückt und erst nach dem Ausspielen der Zeitscheibe wieder freigegeben.

Die Nodes und Gateways haben ebenfalls Gleichung 2-13 hinterlegt und können ihren Slot aufgrund der Kenntnis über die Parameter Spreading Factor, Gesamtlänge der Zeitscheibe (hier: 509ms) und Position innerhalb der Zeitscheibe (hier: 0 – 4) selbst berechnen. Dies geschieht unter minimaler Ausnutzung der Kanalbelegung durch die Verwendung sehr weniger, kurzer Nachrichten und ist daher effizienter als die Synchronisation auf ein absolutes Zeitsystem und zusätzlicher Zuteilung der Zeitscheibensegmente.

Abbildung 23 zeigt schematisch das zeitliche Vorgehen bei der Datenübermittlung. Zur Zeitscheibe gehören alle Nodes und Gateways, die mit einem Server verbunden sind. Jeder Slot wird nur einmalig vergeben, da die Überlappung des Empfangsbereichs zweier Gateways nichts auszuschließen ist. Dem Beispiel aus Code 12 entsprächen die Nodes in der Abbildung Slot 0, Slot1 und Slot 2. Die beiden Gateways halten Slot n+1 und Slot n+2 (Σ 5 Teilnehmer). Seien zu Beginn von Slot 0 alle Teilnehmer synchronisiert worden, beginnt ab diesem Punkt die Zeitscheibe für 509ms und mit allen ganzzahligen Vielfachen davon bis zur nächsten Synchronisation. Damit ergibt sich der nächste Sendezeitpunkt T_{sn} für Node n in Zeitscheibe k, ab dem Synchronisationszeitpunkt $T = 0$ zu

$$T_{sn} = k \cdot 509ms + n \cdot 101.8ms\,; \quad \forall\, k,n \in \mathbb{N}\,. \tag{5-6}$$

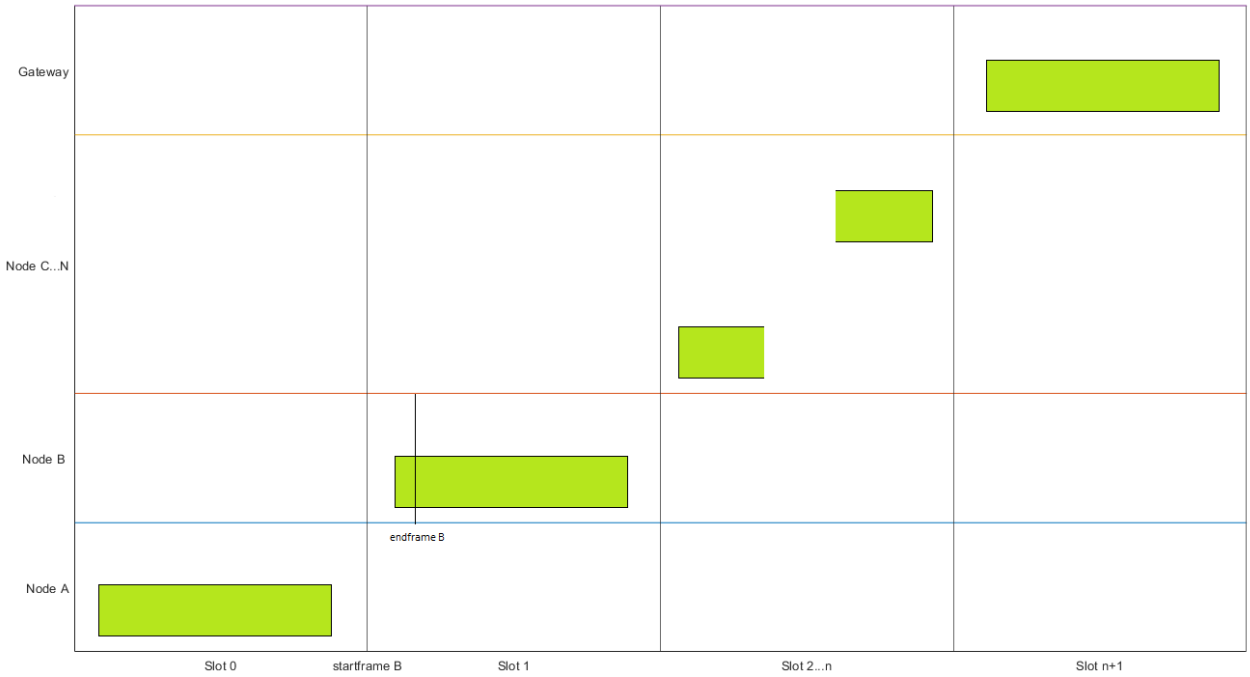


Abbildung 23 Datenübermittlung nach Zeitscheibe

In der Abbildung ist ebenfalls für Node B (Slot 1) das Startframe und Endframe eingezeichnet. Während das Startframe per definitionem auf dem Beginn des Slots liegt, ist das Endframe so gewählt, dass der Nachrichtenversand den folgenden Slot nicht tangiert, wenn der Node zu diesem spätestmöglichen Zeitpunkt sendet.

5.5.8.5 Algorithmus

Der Algorithmus für die Berechnung der Zeitscheibensegmente folgt einer strengen Logik. Der Spreading Factor und die damit einhergehende Datenrate sowie die maximale Nachrichtenlänge ist auf dem Node und dem Server bereits parametriert.

5.5.8.6 Erstellung der Zeitscheibe auf dem Server

Zum Erstellen der Zeitscheibe lädt der Server die Anzahl der Loridane Teilnehmer aus der globalen Konfiguration (Anhang II) und berechnet sie anhand Gleichung 2-13 und der Anzahl bekannter Netzwerkteilnehmer.

```
const devices = global.get("LORIDANE.devices");
var devicecount = devices.gw.length + devices.nodes.length;
sf = devices.gw[0].sf;

//Gleichtung 2-13
timedisksize = Math.ceil(
                 ((8 + ((400 - 4*sf + 28)/(4*sf)*5)) * Math.pow(2,sf)/(1e5)*1000)*devicecount
                );

msg.payload = {size:timedisksize,devicecount:devicecount};
global.set(["LORIDANE.timedisk.size","LORIDANE.timedisk.count"],[timedisksize,devicecount]);
return msg;
```

Code 13 Berechnung der Zeitscheibenlänge (JavaScript)

Im folgenden Schritt wird die Zeitscheibe aufgeteilt (Anhang III, H) und wie in (Code 12) an die Funktion übergeben, die eine Weiterleitung an den MQTT-Broker durchführt. Die Übergabe an den Broker ist zeitkritisch, da eine weitere Verzögerung der Nachrichten ab diesem Punkt nicht mehr möglich ist. Somit muss der Server bereits beim Ausspielen der Zeitscheibe die Nachrichten zeitlich so verteilen, dass per Zeitscheibenumlauf nur so viele Nachrichten an das Gateway übergeben werden, wie in einem Slot versendet werden können. Betrachtet man dazu die Datensätze zur Slotzuweisung am Beispiel von (Code 12), sind für jede Nachricht höchstens 26 Byte Nutzdaten zu erwarten, unter der Voraussetzung, dass die Anzahl der Netzwerkgeräte 99 und die Gesamtdauer der Zeitscheibe 99999 ms nicht übersteigt. Damit kann das Gateway höchstens zwei Nachrichten pro Slot mit einer Slotzuweisung übertragen. Eine größere Zeitscheibe ist möglich. In diesem Fall versendet der Server nur noch eine Zuweisung pro Slot. Das genaue Timing wird dann im Folgeschritt durch eine Funktion bestimmt (Anhang V, L).

5.5.8.7 Berechnung des Zeitslots auf den Nodes

Erhält der Node eine LoRa-Nachricht nach (Code 12) mit dem individuellen Index und der Gesamtlänge der Zeitscheibe, können die Berechnungen wie folgt durchgeführt werden.

```
/* global Scope */
unsigned long int tdsize; // Time Disk Size in total
unsigned long int startframe; // Slots starttime
unsigned long int endframe; //endtime for beginning transmission
unsigned long int nowtime = millis(); //current systemtime in ms since Start


/*relevant part of parser foo*/

bool IRAM_ATTR onDownlink(String LRpayload) { // Input: NO>UID<td:>diskex<;>tdsize<;
  // read packet
  String msghandle;
  msghandle = LRpayload; // LRpayload is destructive var, copy
  int PL = 3; // Search Pattern Length
  /*
  ...shortened
  */
  int diskex = msghandle.indexOf("td:"); // index if timedisk
  int syncindex = msghandle.indexOf("sync"); // index if syncmsg
  if (!isConfig(cindex, msghandle)) {
    return false;
  }
  confirmflag = true;
  /*
  ... shortened
  */

  if (diskex != -1) {
 //if msg contains td:<number>;<number; . e.g. td:3;509; -> Frame 3 in 509ms (305 - 407ms)
    String tdcountstr = msghandle.substring(diskex + PL, msghandle.indexOf(";", diskex));
    int newdiskex = msghandle.indexOf(";", diskex) + 1;
    String tdsizestr = msghandle.substring(newdiskex, msghandle.indexOf(";", newdiskex));
    unsigned tdcount = tdcountstr.toInt();
    tdsize = tdsizestr.toInt(); //509ms
    //Gleichung 2-13
    int duration = ceil((8 + ((400 - 4 * SF + 28) / (4 * SF) * 5)) * pow(2, SF) / (1e2));
    startframe = duration * tdcount; //305ms
    endframe = duration * (tdcount + 1) - 51; // 407 – 51 ms

  }
  if (syncindex != -1) { //if cn: -msg contains "sync"
    synctime = millis();
    confirmflag = false;
  }
 /*
  ...
  */
  return true;
}
```

Code 14 Algorithmus zum Berechnen des Zeitscheibensegmentes auf dem Node (C++)

Nach der Evaluation des Erhalts einer Zeitscheibe, berechnet der Node die Länge eines Zeitscheibensegmentes (duration). Dies ist mit derselben Rundung wie bei der Erstellung der Zeitscheibe auf dem Server verbunden (ceil()) und bezweckt die konservative Generierung eines ganzzahligen Wertes. Anhand des Index des Zeitslots (tdcount) berechnet der Node den Beginn (startframe) und das Ende (endframe) seiner individuellen Sendezeit. Die Variable `endframe` ist um die Sendedauer des größtmöglichen Datenpaketes nach vorne verschoben, sodass eine Signalüberlagerung mit dem Node im nächsten Slot ausgeschlossen werden kann. Hiermit gibt sie

nicht das Ende des Zeitfensters an, sondern den spätestmöglichen Sendebeginn. Die Kenntnis der Gesamtlänge der Zeitscheibe (tdsize) wird erst zur Evaluation zum Sendezeitpunkt benötigt.

Sind im Node Daten gespeichert, die auf Versand warten, ist zu überprüfen, ob das Mindestintervall zwischen zwei Sendevorgänge abgelaufen ist (sendinterval).

```
unsigned long int nowtime = millis(); // ms since system start

bool maysend(unsigned long nowtime) {
  if (nowtime - lastTimeSend >= sendinterval && inFrame()) {
    return true;
  }
  return false;
}
```

Code 15 Evaluation des Sendeintervalls auf dem Node (C++)

Ist dies positiv evaluiert, ist zusätzlich festzustellen, ob innerhalb der Zeitscheibe auch der Zeitslot zwischen startframe und endframe liegt, sodass die Nachricht abgesetzt werden kann. Die Vulnerabilitätszugabe auf den Zeitslot ist so bemessen, dass die Funktion inFrame() innerhalb eines Evaluationszyklus ([2.5 ; 11] ms) mindestens einmal `true` zurückgeben kann.

```
bool inFrame() {
  nowtime = millis(); // ms since system start
  unsigned int progress = (nowtime - synctime) % tdsize;
  if (progress >= startframe && progress < endframe) {
    return true;
  }
  return false;
}
```

Code 16 Evaluation des Zeitslots auf dem Node (C++)

Dabei wird der Zeitunterschied zwischen dem letzten Synchronisationszeitpunkt (synctime) und der Jetztzeit (nowtime) geprüft. Das Modulo aus Jetztzeit und Zeitscheibenlänge gibt den Fortschritt in der aktuellen Zeitscheibe an. Ist der Fortschritt erstmalig größer als startframe, ist der Sendezeitpunkt erreicht. Hieraus folgt, dass der Node den Zeitslot bis zur nächsten Nachricht abwartet, auch wenn das vorgegebene Sendeintervall kürzer wäre als die Zeitscheibe. Dabei sind alle Zeitkritischen Funktionen mit hohem Berechnungsaufwand als IRAM_ATTR ausgeführt. Damit werden sie nicht bei Funktionsaufruf aus dem Flashspeicher geladen, sondern direkt bei Systemstart im RAM abgelegt, wodurch kürzere Zugriffzeiten ermöglicht werden (Semtech Corp. 2021).

5.5.9 Verschlüsselung der Nutzdaten

Die Nutzdaten werden auf dem Node nach dem AES128 Standard im ECB-Modus (Electronic Code Book) ohne Padding verschlüsselt. So ist vor Überspielen der Node Software auf den Heltec Wireless Stick die Variable aes_key im Modul credentials.h und auf dem Server in loridaneConfig.json einzutragen (s. 5.3). Da die Nutzdaten verschlüsselt vom Gateway zum Server weitergeleitet werden, ist die MQTT Übertragung zusätzlich zu Benutzername und Passwort nicht abgesichert. Die Entschlüsselung erfolgt direkt auf dem Server und es liegt somit eine end-to-end-Verschlüsselung vor. Der Server erkennt verschlüsselte Nutzdaten automatisch, da sie nicht dem festgelegten Protokoll entsprechen. Wenn versendete Daten nicht sensibel sind, kann der Nutzer

unverschlüsselte Übertragungen veranlassen, indem er im Modul credentials.h die Schrägstriche vor //#define ENCRYPT hinzufügt.

5.5.10 Grafische Benutzeroberfläche

Die Parametrierung der Netzwerkinstanzen erfolgt über eine Benutzeroberfläche die als Web-Applikation zur Verfügung gestellt wird. Auf sie kann von Browsern auf Desktop- und Mobilgeräten zugegriffen werden.

Wenn der Server über Ethernet mit dem LAN verbunden ist, geschieht dies über die URL: loridane.gw:1880/ui oder <IP.Adresse.des.PI>:1880/ui.

Falls der Zugriff über das eigene WLAN-Netzwerk des Servers erfolgt, entspricht die IP-Adresse des Servers derjenigen während der Installation vergebenen Subnetzkennung. Wurde der Standardwert nicht verändert, ist dies 192.168.4.1:1880/ui.

Die Benutzeroberfläche besteht aus sechs Tabs, die über das Burger Menü links-oben zu erreichen sind.

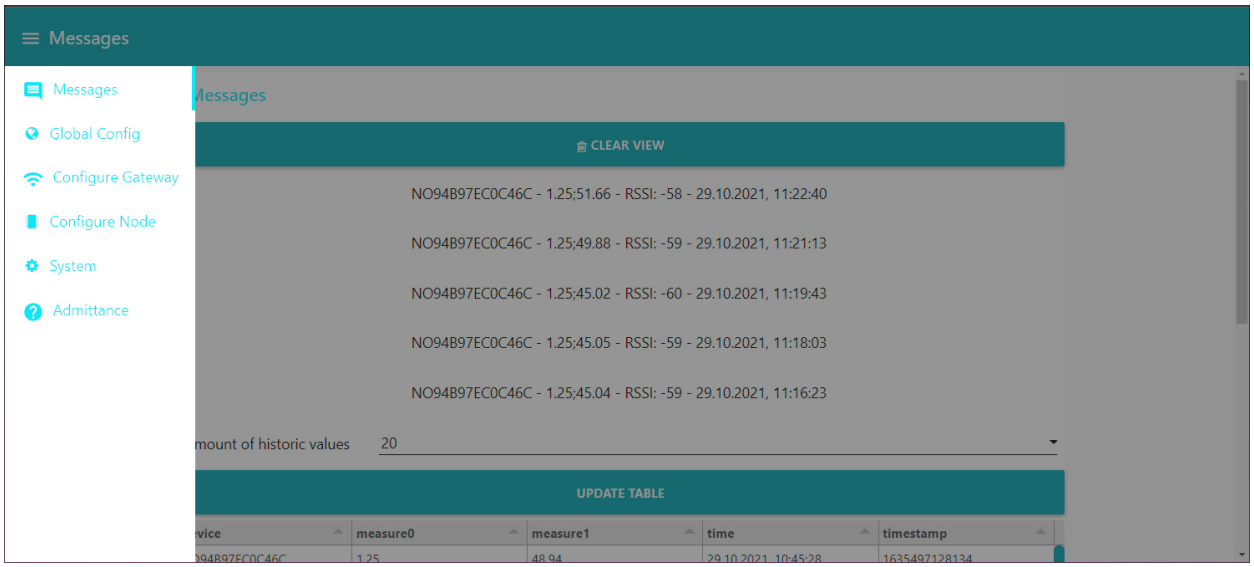


Abbildung 24 Loridane GUI Tab-Menü

Die Bezeichnungen der Tabs sind:

- Messages,
- Global Config,
- Configure Gateway,
- Configure Node,
- System und
- Admittance.

Die Funktionen der Tabs gliedern sich wie im Folgenden beschrieben.

5.5.10.1 Tab: Messages

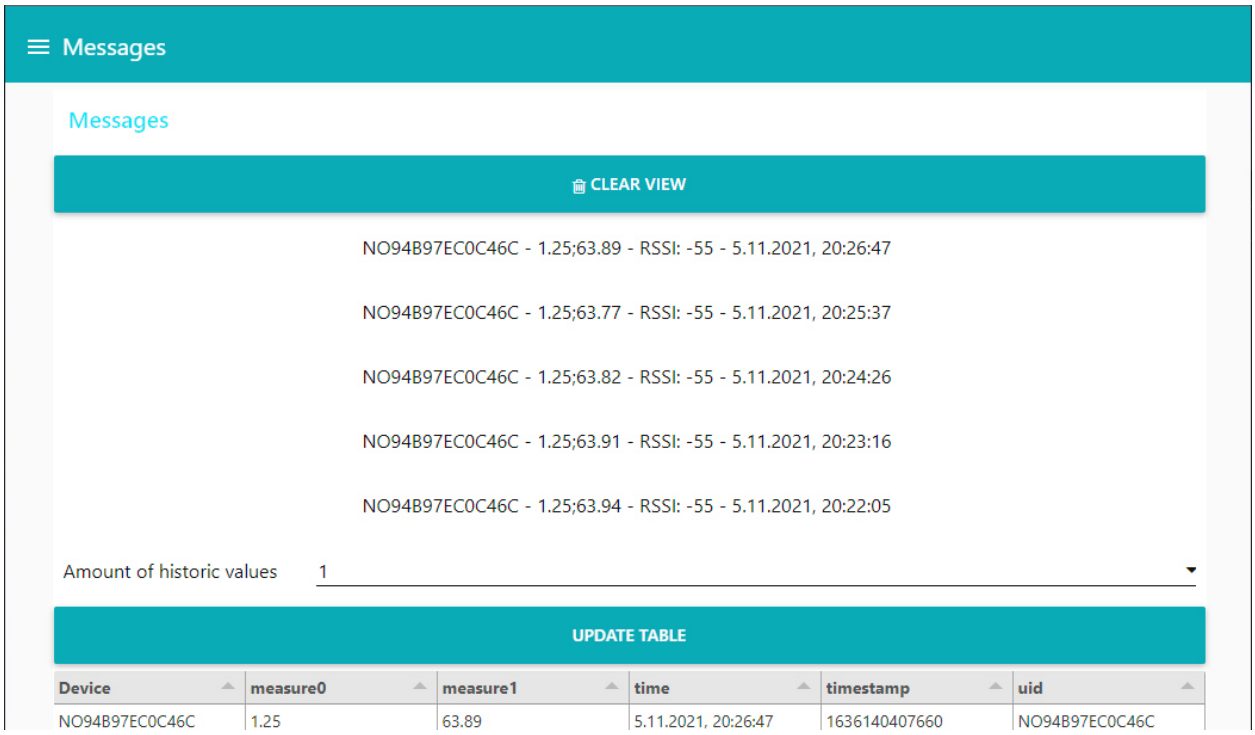


Abbildung 25 Loridane GUI Tab Messages

Im Hauptfenster werden im oberen Teil die letzten fünf Datensätze angezeigt, die auf dem Server eingegangen sind. Neben der UID des versendenden Nodes, der Uhrzeit und der Signalstärke, wird der übertragene Nutzdatensatz in unverarbeiteter Form als String angezeigt. So kann festgestellt werden, ob das Datenformat korrekt ist und erwartbare Werte übertragen werden.

Darunter ist eine Tabelle im Format einer Tabellenkalkulation dargestellt. Die hier angezeigten Werte werden direkt aus der Datenbank gelesen. Daher ist hier die Aufteilung der semikolagetrennten Datensätze in „measures“ erkennbar. Mit dem Drop-Down-Menü kann der Anwender festlegen, wie viele historische Datensätze aus der Datenbank angezeigt werden sollen. Die Tabelle aktualisiert sich dabei frühestens nach 30 Sekunden selbstständig, wenn Nachrichten eingehen oder wenn der Anwender die Auswahl ändert.

5.5.10.2 Tab: Global Config

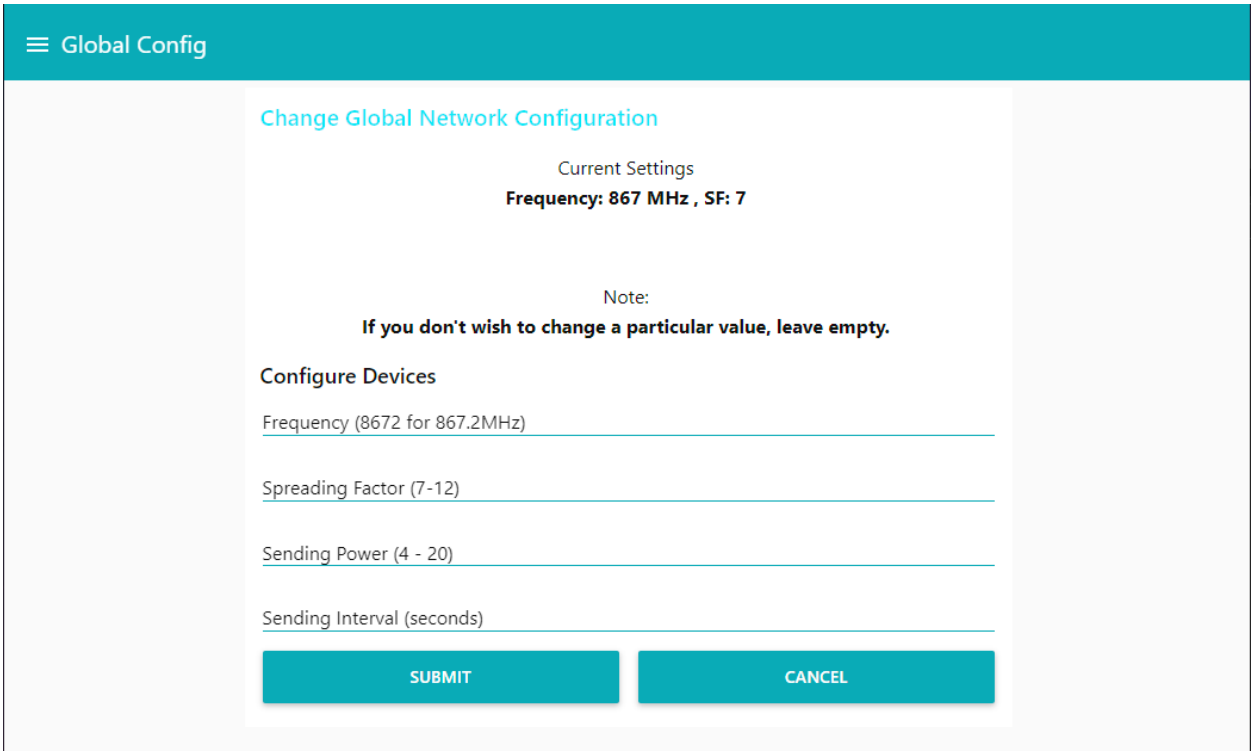
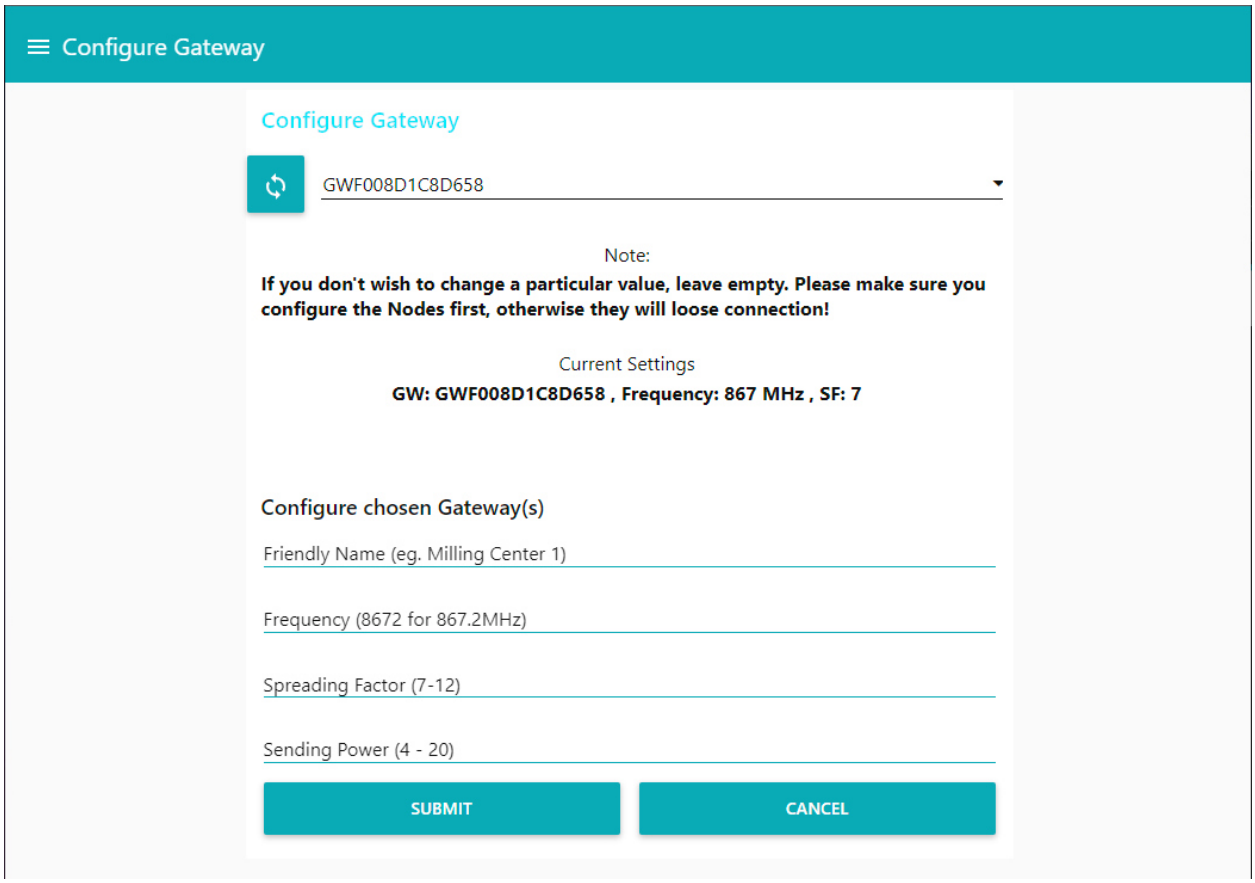


Abbildung 26 Loridane GUI Tab Global Config

Im Großteil der Anwendungsfälle wird es gewünscht sein, eine globale Konfiguration über das ganze Netzwerk auszuspielen. Dazu wird die Eingabemaske im Tab „Global Config“ verwendet. Soll für das Netzwerk die Frequenz, der Spreading Factor, die Sendeleistung oder das Sendeintervall der Nodes geändert werden, geschieht dies hier mit einer Eingabe für alle Gateways und Nodes, die sich in Loridane befinden.

5.5.10.3 Tab: Configure Gateway



Falls an einem Loridane Server mehrere Subnetze eingerichtet werden sollen, ist es möglich, ein oder mehrere Gateways unabhängig von den anderen zu parametrieren. Dies geschieht analog zu (5.5.10.2). Über das Dropdown Menü werden dazu die Gateways, deren Einstellungen verändert werden sollen, ausgewählt. Da die Nodes an den Server, nicht aber am Gateway angebunden sind, führt eine Änderung der Gatewayeinstellungen prinzipbedingt dazu, dass das Gateway keine Einstellungen mehr nachträglich an Nodes übertragen kann. Daher muss im ersten Schritt die Konfiguration der Nodes erfolgen.

5.5.10.4 Tab: Configure Node

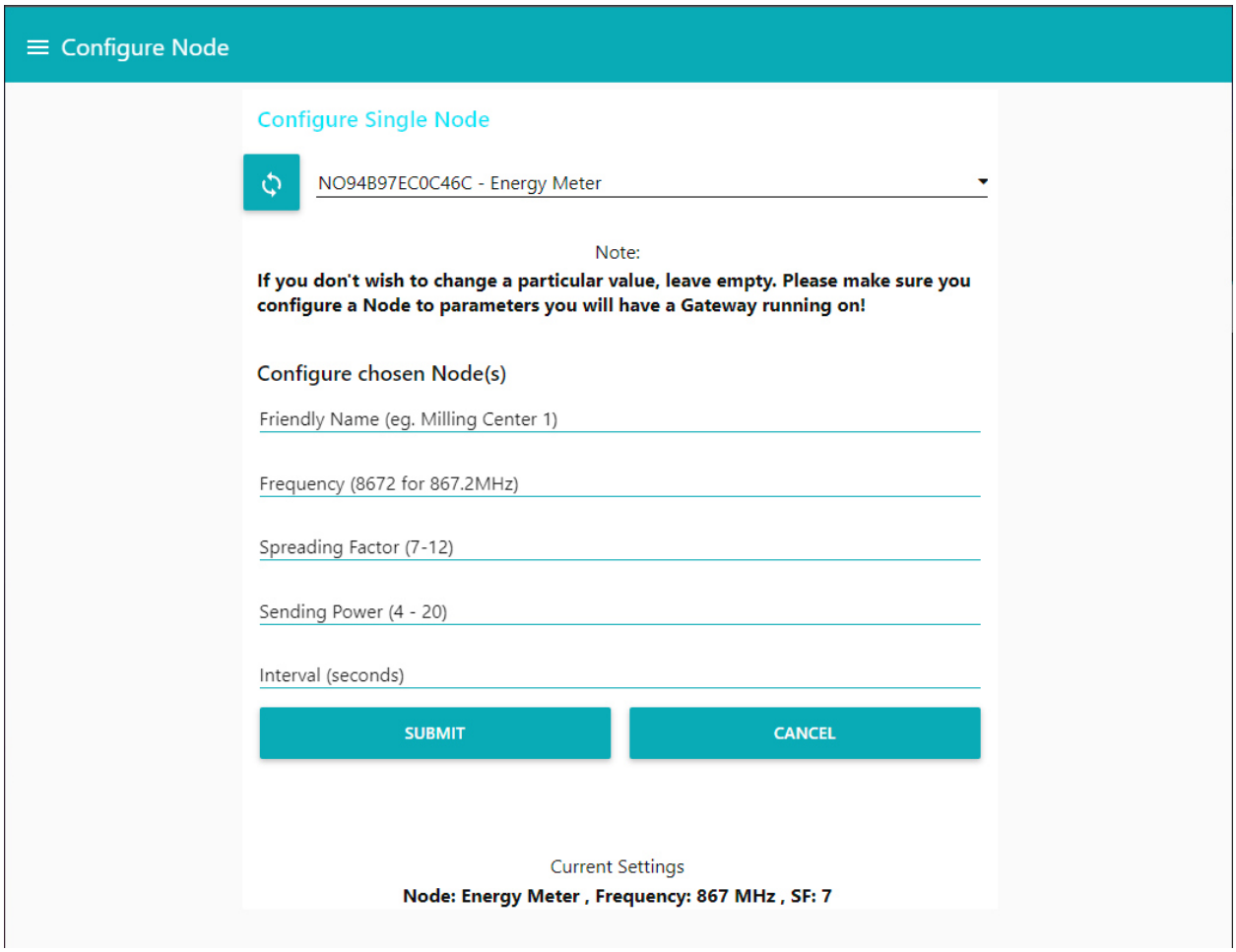


Abbildung 27 Loridane GUI Tab Configure Node

Entsprechend der Ausführungen zur Änderung von Gatewayeinstellungen, ist dies für einen oder mehrere Nodes möglich. So wird die Option geschaffen an einem Standort zum Beispiel mehrere Subnetze in Multistar-Konfiguration (vgl. 4.4) zu erstellen oder diese Subnetze an verschiedenen Standorten mit den lokal idealen Einstellungen zu betreiben. Für die Parametrierung einzelner Geräte oder einer Gruppe ist immer zuerst die Konfiguration der Nodes und danach die Konfiguration des Gateways (s. 5.5.10.3) vorzunehmen.

5.5.10.5 Tab: System

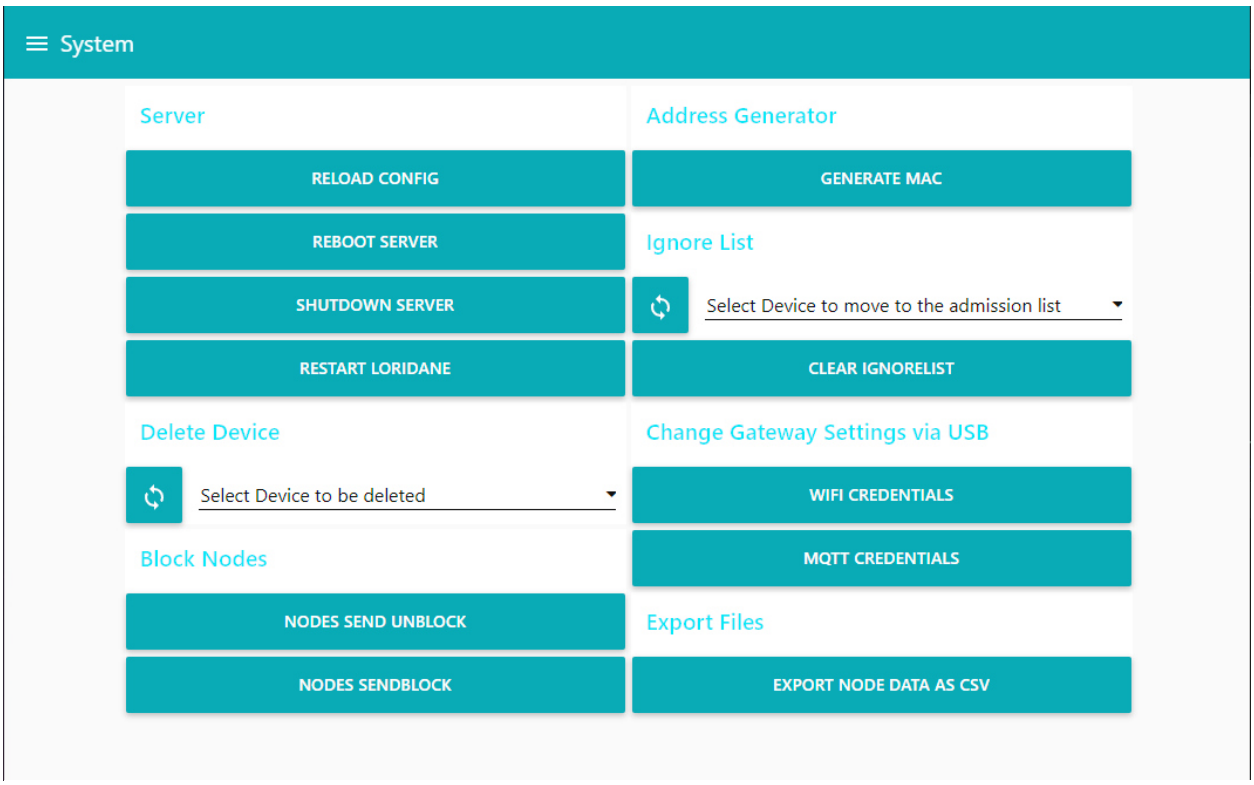


Abbildung 28 Loridane GUI Tab System

Das Tab „System“ ist in mehrere Gruppen gegliedert deren Funktionen durch einen Mausklick auf die entsprechende Schaltfläche ausgeführt werden können.

5.5.10.5.1 Server

Die Gruppe „Server“ beinhaltet die Möglichkeit den Server neu zu starten, gänzlich herunterzufahren oder nur das Loridane-Programm neu zu starten, zum Beispiel nach einer manuellen Änderung in memCacheLoridane.json oder loridaneConfig.json. Ein Druck auf eine dieser Schaltflächen muss über den folgenden Prompt bestätigt werden. Ein Aktionsabbruch ist ebenfalls möglich.

5.5.10.5.2 Delete Device

In „Delete Device“ besteht die Möglichkeit die vorhandenen Netzwerkgeräten aus dem Programmspeicher in ein Drop-Down-Menü zu laden. Von hier aus ist die Auswahl eines oder mehrere Geräte möglich, deren Konfiguration gelöscht werden soll, um sie damit aus dem Loridane-Netzwerk zu entfernen. Sind in der Datenbank noch gespeicherte Messwerte zu diesen Geräten vorhanden bleiben diese bestehen.

5.5.10.5.3 Export Files

Unter „Export Files“ ist es möglich alle JSON-Dateien die im Ordner /database liegen in kommagetrennte Werte (*.csv-Dateien) zu konvertieren. Damit können die Daten in viele

Programme importiert werden. So zum Beispiel in Monitoring-Software oder andere Datenbanksysteme (s. 5.5.4).

5.5.10.5.4 Address Generator

Für zukünftige Implementierungen von Nodes für Loridane, kann es gewünscht oder auf aufgrund der benötigten Sensorik oder Umgebungsbedingungen notwendig sein, andere MyC/LoRa-Chip Kombinationen zu wählen. Die hier verwendeten Heltec Wireless Sticks stellen nur eine Option dar. Um dem Umstand Rechnung zu tragen, dass andere MyC kein integriertes IP-Stack und damit keine hart kodierte MAC-Adresse besitzen, ist unter „Address Generator“ die Möglichkeit gegeben UIDs zu erzeugen, die mit Loridane kompatibel sind. Dazu wird die Schaltfläche „Generate Mac“ genutzt. In Loridane ist ein Algorithmus hinterlegt, der dabei ein zufällig erstelltes Buffer von sechs Byte Länge erzeugt und diese in stringifizierten Hexadezimalzahlen ausgibt.

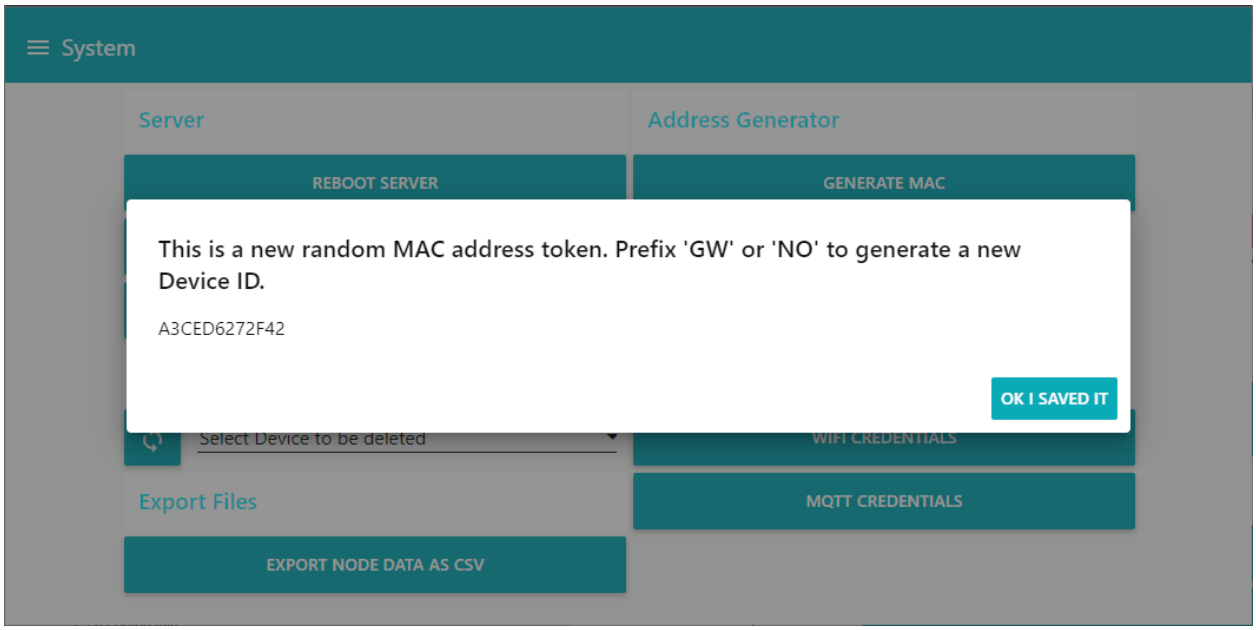


Abbildung 29 Loridane GUI Generate Mac

Für den ausgegebenen String wird überprüft, ob bereits ein Gerät dieser UID mit Loridane gekoppelt ist. Für diesen unwahrscheinlichen Fall wird eine neue Zeichenfolge erzeugt. Andernfalls wird sie wie in Abbildung 29 auf der Benutzeroberfläche angezeigt. Durch Voranstellen des Präfixes „NO“ entsteht daraus eine zur Hardwarekodierung nutzbare UID der Form NOA3CED6272F42. Der Vorgang ist beliebig oft wiederholbar, da die Hardwareadresse nur dann in die Datenbank übernommen wird, wenn der Server tatsächlich eine Nachricht von dieser UID enthält.

5.5.10.5.5 Ignore List

Eine weiteres Drop-Down-Menü zeigt sich in (Abbildung 28) unter „Ignore List“. Wenn bereits vorhandene UIDs unter „Delete Device“ vom Server entfernt wurden, ist es gewünscht von diesen kein weiteres Zutrittsgesuch zum Loridane (s. nächster Abschnitt) zuzulassen. Somit werden gelöschte UIDs in eine Ignorierliste verschoben, um deren zukünftige Nachrichten zu blockieren. Soll

der Node wieder in Loridane admittiert werden, kann er dazu aus der „Ignore List“ wieder in der „Admission List“ abgespeichert werden.

5.5.10.5.6 Change Gateway Setting via USB

Da das Gateway via WLAN und MQTT mit dem Server kommuniziert, muss eine Möglichkeit gegeben sein die Zugangsdaten zu parametrieren. Dies geschieht in einem ersten Schritt mit dem Hartkodieren der Zugangsdaten bei der Gatewayinstallation wie unter (5.3). Dennoch muss die Software für einen Wechsel des Netzwerkes nicht neu aufgespielt werden. Sofern das Gateway direkt an einer USB-Schnittstelle des Servers angeschlossen ist, kann die Serversoftware seriell und kabelgebunden mit dem Gateway kommunizieren. Hierfür dienen die Schaltflächen „WiFi Credentials“ und „MQTT Credentials“ in (Abbildung 28). Wenn in loridaneConfig.json Zugangsdaten hinterlegt sind, müssen diese nicht von Hand eingetragen werden.

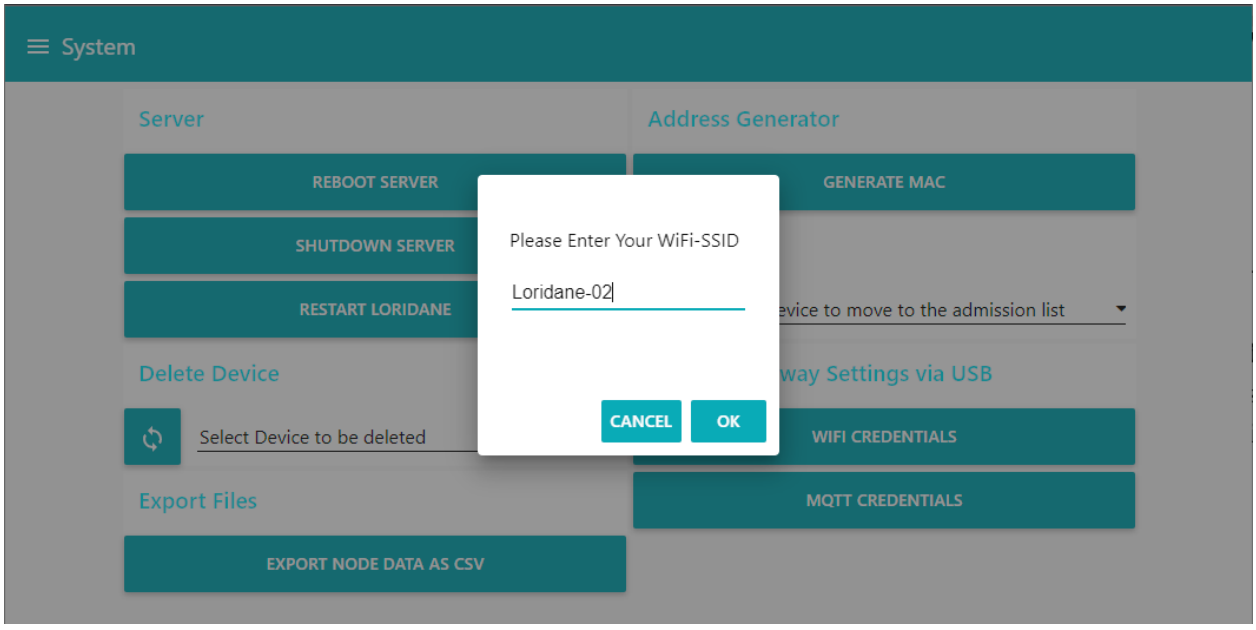


Abbildung 30 Loridane GUI WiFi Credentials

Einige Sekunden nach einem Druck auf „WiFi-Credentials“ öffnet sich eine Maske zur Eingabe des Netzwerknamens. Nach Bestätigung und Speichern der Information wird automatisch ein weiteres Fenster zu Eingabe des Passwortes geöffnet. Für die Einstellung der Zugangsdaten zum MQTT-Broker wird analog verfahren. Dafür ist die Kenntnis der IP-Adresse des Brokers erforderlich. Auf eine drahtlose Konfigurierbarkeit des Gateways wird aus Sicherheitsgründen bewusst verzichtet.

5.5.10.5.7 Block Nodes

Mit der Schaltfläche „Nodes Send Block“ ist es möglich den Sendevorgang auf allen Nodes zu verzögern bis sie wieder mit „Nodes Send Unblock“ freigegeben werden. Dazu wird eine Nachricht an die Nodes übermittelt, die wie beim Ausspielen der Zeitscheibe den Sendevorgang unterdrückt. Dabei bleiben alle berechneten Werte bis zur Freigabe erhalten.

5.5.10.6 Tab: Admittance

Der Tab „Admittance“ hält ein Drop-Down-Menü. Hier wird eine Liste von UIDs angezeigt, die ein Zugangsgesuch an den Server gesendet haben. Dieses kann hier bestätigt werden, woraufhin auf dem Server eine Konfiguration für das jeweilige Gerät angelegt wird. Nicht admittierte Geräte werden in die Ignorierliste verschoben.

5.6 Berücksichtigung der rechtlichen Rahmenbedingungen

```
{
    "allowedFreq":
        [
            {"min":863,"max":864.875,"factor":1000},
            {"min":865,"max":868.475,"factor":100},
            {"min":868.7,"max":869.025,"factor":1000},
            {"min":869.4,"max":869.525,"factor":1000},
            {"min":869.7,"max":869.875,"factor":100}
        ],
        "freq":{
            "min":863,
            "max":869.875
        }
}
```

Code 17 Nutzbare Funkkanäle (JSON)

Die Einhaltung von Vorgaben zur Nutzung lizenzfreier Funkfrequenzen durch die Bundesnetzagentur (s. 2.6) ist durch den Betreiber von Funksystemen sicherzustellen. Unter Berücksichtigung der technischen Eigenschaften der SX1276 LoRa-Trasceiver, ergeben sich nach (Bundesnetzagentur Vfg 133 / 2019 2019) maximale Sendezeiten und damit Mindestsendeintervalle für die Nodes in Loridane. Die von den Nodes erzeugten Datensätze haben eine nicht fest definierte Größe und das Sendeintervall ist dynamisch ausgelegt. Darüber hinaus ergibt sich durch die Parametrierbarkeit des Spreading Factor in Loridane ein komplexer Zusammenhang, der für den Anwender bezüglich der genutzten Sendezeit nicht ohne weiteres transparent ist. In kommerziellen LoRaWAN Anwendungen wird dem begegnet, indem die Fair-Use-Policy (s. 2.3.3) die Sendeintervalle konservativ beschränkt, um die rechtlichen Vorgaben nicht auszureizen. Das liegt unter anderem darin begründet, dass LoRaWAN-Nodes global standortunabhängig sein sollen und daher nicht in einer, mit Loridane vergleichbaren, Form mit dem Netzwerk gekoppelt sind. Aufgrund der engen Verbindung zwischen Endgeräten und Server in Loridane ist es möglich, die Sendeintervalle und -Dauer, die „Time-On-Air“ (TOA) der Endgeräte genau zu erfassen. Dazu wird durch den Server für jede eingehende Nachricht die TOA berechnet und über den Beobachtungszeitraum von einer Stunde gemittelt. Damit ist bekannt, wie hoch die zeitliche Kanalbelegung durch Loridane Nodes ist und ohnehin besteht Kenntnis über die Frequenz sowie den Spreading Factor. Des Weiteren ist in Loridane eine Tabelle der nutzbaren Kanäle und Kanalbelegungen für das europäische SRD-Band hinterlegt, wodurch gesperrte Kanäle unter Inbezugnahme der Signalbandbreite berücksichtigt werden. Wählt der Anwender eine Frequenz oder ein Sendeintervall, das nach rechtlichen Bedingungen nicht wählbar ist, wird durch die Benutzeroberfläche ein Hinweis gegeben, dass die Parametrierung nicht durchgeführt werden kann und ein zulässiger Vorschlag unterbreitet. Dies gilt *nicht* für eine nachträgliche Erhöhung des Spreading Factor. Jedoch wird das aktuell kleinstmögliche

Übertragungsintervall im Tab „Messages“ angezeigt. Damit ist das Intervall für einen höheren Spreading Factor nach Gleichung 2-5 zu berechnen.

Wählt der Anwender hingegen eine erlaubte Frequenz, wird anhand der gemittelten Nachrichtenlänge das kleinste mögliche Sendeintervall berechnet und angezeigt. Dazu dienen die Daten aus Code 17. Für den ersten Kanal liegt hier die maximal erlaubte Kanalbelegung pro Gerät bei $B_{max} = 0.1\%$ (Bundesnetzargentur, Vfg.). Daraus ergibt sich ein Mindestintervall von

$$I_{min} = TOA_{mean} \cdot \frac{100\%}{B_{max}} = TOA_{mean} \cdot 1000 \tag{5-7}$$

und analog für die Kanäle mit einer anderen Höchstbelegung.

5.7 Maximale Anzahl von Netzwerkteilnehmern

Die maximale Teilnehmerzahl ist bei Systemen, die auf einem IPv4-Adressraum basieren auf 256 begrenzt, wenn keine Subnetze verwendet werden. Der Adressraum reicht von 0.0.0.0 bis 255.255.255.255, wobei die letzte Stelle lokal vergeben wird. Durch den MAC-basierten 6 Byte-Adressraum in Loridane sind theoretisch höchstens 256^6 Adressen zu vergeben. Dies wird allerdings nicht dem realen Anwendungsfall von Loridane entsprechen. Im Versuch zeigt sich, dass die ESP32 SoC, aufgrund des Fehlgangs der Systemuhr, die Zeitscheibenzuweisung zuverlässig einhalten können, wenn sie spätestens nach zehn Minuten synchronisiert werden. Per definitionem steht je Zeitscheibenumlauf ein Slot je Gateway zur Verfügung, weshalb die Zeitscheibenlänge auf 10 Minuten zu begrenzen ist. In diesem Fall stehen

$$n_n = \left\lfloor \frac{TD_{size}}{W_{slot}} - n_g \right\rfloor \tag{5-8}$$

Slots für Uplinks zur Verfügung. Mit der Zeitscheibenlänge $TD_{size} = 10min$, der Slotbreite $W_{slot} = 101.8\ ms$ und einem Gateway ($n_g = 1$) ergibt sich eine maximale Anzahl von Nodes n_n

$$n_n = \left\lfloor \frac{6 \cdot 10^5\ ms}{101.8\ ms} - 1 \right\rfloor = 5892. \tag{5-9}$$

Die Loridane Implementierung sieht als Standard ein Synchronisationsintervall von 233 Sekunden vor. Wenn keine größere Anzahl an Netzwerkteilnehmern notwendig ist, ergibt sich nach Gleichung 5-9 n_n zu 2287 und einem kleinstmöglichen Übertragungsintervall von 233 Sekunden je Node entsprechend TD_{size}.

Sind weniger Nodes als die maximal Möglichen im Netzwerk und wird daher die Zeitscheibe innerhalb eines Synchronisationszyklus mehrmals durchlaufen, sind ebenfalls mehr Slots für das Gateway vorhanden. Die Uplinkkapazität n_u beträgt dann nicht 2287, sondern

$$n_u = \left\lfloor \frac{I_{sync}}{(n_n + n_g) \cdot W_{slot}} \cdot n_n \right\rfloor. \tag{5-10}$$

Im Beispiel mit 50 Nodes beträgt sie

$$n_u = \left\lfloor \frac{233000ms}{(50+1)\cdot 101.8\,ms} \cdot 50 \right\rfloor = 2243. \tag{5-11}$$

Daher liegt n_u etwas niedriger als n_n, aber es können bis zu 44 Nachrichten pro Node in einem Synchronisationsintervall übermittelt werden, wodurch das Sendeintervall auf 5.2 Sekunden verringert werden kann.

Der Standardwert für das Synchronisationsitervall von 233 Sekunden ist unter Beachtung folgender Voraussetzungen begründet gewählt:

- Die Synchronisation ist zeitlich ausreichend genau
- Es ist eine ausreichende Anzahl Slots für eine, voraussichtlich in den meisten Anwendungsfällen, ausreichende Anzahl an Netzwerkteilnehmern verfügbar.
- Der Anwender kann das Sendeintervall auf ganzzahlige Sekunden einstellen und 233 ist eine Primzahl. Daher werden in der Zeitscheibe, in der die Synchronisation stattfinden soll, statistisch gesehen und prinzipbedingt die wenigsten Uplinks übermittelt und die Wahrscheinlichkeit einer erfolgreichen Synchronisation ist maximal.

Das Synchronisationsintervall ist unter Beachtung der oben genannten Einschränkungen in loridaneConfig.json unter `syncInterval`: in Sekundenschritten frei einstellbar.

5.8 Maximale Paketübermittlungsrate

Die Länge der Zeitscheibe und damit auch die Paketübermittelungsrate (R_{pd}) ist abhängig vom netzwerkweit eingestellten Spreading Factor und der verhältnismäßigen Anzahl der Netzwerkteilnehmer. Da je Zeitscheibe ebenfalls ein Downlink-Slot je Gateway vergeben wird bedingt eine Erhöhung der Anzahl der Gateways eine anteilige Verringerung der Uplink-Slots.

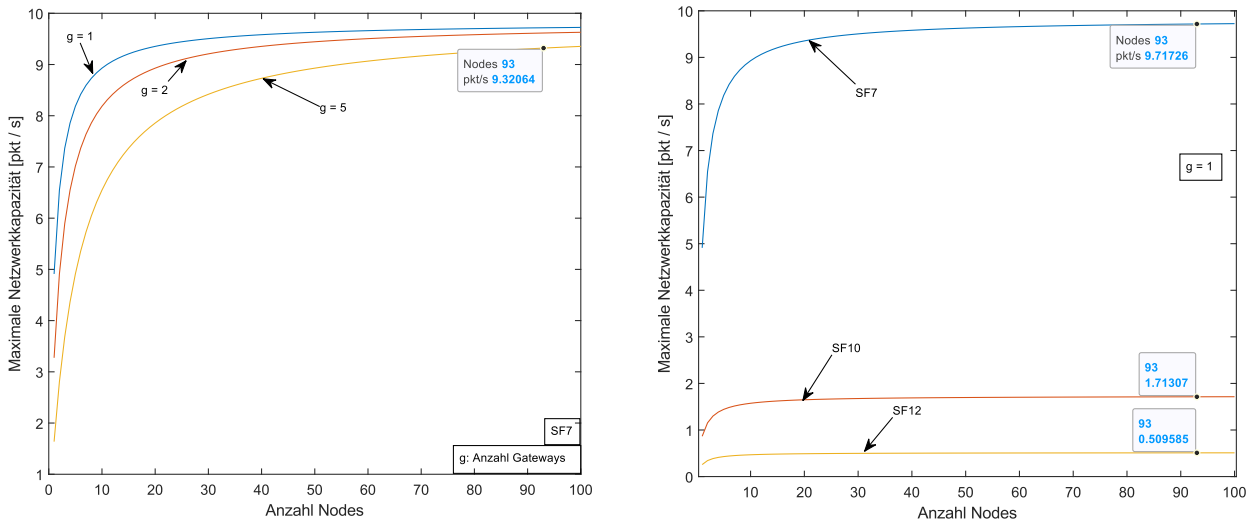


Abbildung 31 Maximale Paketübermittlungsrate nach Gateways und Spreading Factor

Wenn das Netzwerk auf der höchsten Datenrate (SF7) betrieben wird und nur ein Gateway (g = 1) vorhanden ist, stellt sich die höchste Paketübermittlungsrate ein. Ab 20 Nodes im Netzwerk stehen circa 9.5 Uplink-Slots je Sekunde zur Verfügung (Abbildung 31, links). Eine Erhöhung des Spreading Factor auf den nächsthöheren Wert verringert die Daten- und damit auch die Paketübermittlungsrate um die Hälfte (Gleichung 2-5)(Abbildung 31, rechts). So beträgt die Länge eines Slots bei SF12 fast 2 Sekunden. Aus diesem Grund ist der Spreading Factor so niedrig wie möglich zu wählen. Eine Erhöhung ist dann sinnvoll, wenn eine Ausnutzung der höheren Empfängersensitivität (Semtech Corp. 2015b, S. 20) bei höherem SF zwecks einer Verlängerung der Reichweite notwendig wird.

5.9 Modulationsmodi bei Sende- und Empfangsvorgang

In Loridane senden alle Nodes nach der regulären Modulationsart (2.2.8) von LoRa (upchirp). Die Gateways sind während aller Uplink-Slots im regulären Modus auf Empfangsbereitschaft. Gateways nutzen bei der Datenübermittlung den InvertIQ-Modus (downchirp). Die Nodes empfangen damit alle Nachrichten mit invertierter Modulation außerhalb ihres eigenen Sendezeitfensters. Die Nachrichten anderer Nodes erhalten sie damit nicht. Nodes und Gateways befinden sich immer im Empfangsmodus und verlassen diesen erst kurz vor dem Nachrichtenversand. Nach dem Absenden der Nachricht wird unmittelbar zurück in den Empfangsmodus gewechselt, losgelöst davon ob das Sendezeitfenster bereits beendet ist oder nicht.

6 Evaluation der Implementierung von Loridane

In diesem Abschnitt wird zunächst eine Evaluation der Implementierung der Nodes an einem Wirkverbrauchszähler und anschließend die Bewertung der Funktionalität des Loridane Netzwerkes erläutert.

6.1 Langzeitversuch

Abbildung 32 Elektrischer Wirkverbrauchszähler ORNO OR-WE-520 [PL14]

Die Funktionalität der Loridane Nodes als LoRa-fähige Impulszähler ist über einen längeren Zeitraum zu überprüfen. Dazu wurde ein Datenserver mit einem 3-phasigen Wirkverbrauchszähler des Typs ORNO OR-WE-520 (s.2.4.3)(Abbildung 32) ausgestattet. Hierbei handelt es sich um einen Wirkverbrauchszähler, der über keine Ethernet- oder Busanbindung verfügt. Es steht ebenfalls keine Funktionalität zur Analyse der Netzqualität zu Verfügung. Jedoch ist eine S0-konforme Schnittstelle vorhanden, die als Datenquelle zur Impulserfassung hinreichend ist. Der Zähler ist darüber hinaus mit Genauigkeitsklasse B MID-Zertifiziert (ORNO-Logistic Sp. Z O.O 2014). Die Belastbarkeit beträgt 80A je Phase, wodurch sich eine maximale symmetrische Anschlussleistung von 55.2 kW ergibt. Nach Datenblatt beträgt die Impulswertigkeit des Gerätes 1.25 Wh/imp und die Impulsdauer 90 ms (ORNO-Logistic Sp. Z O.O 2021).

Der Zähler wurde über mehrere Wochen an der Energieversorgung eines Dateiservers betrieben, um an den Loridane-Server einen Datensatz mit repräsentativem Umfang zu übermitteln.

Der in Abbildung 33 ausschnittsweise abgebildete Datensatz hat einen Umfang von 15761 Samples. Das dynamisch eingestellte Sendeintervall betrug dabei größer oder gleich 60 Sekunden. Eine vergrößerte Ansicht liegt in Anhang IX (S. N).

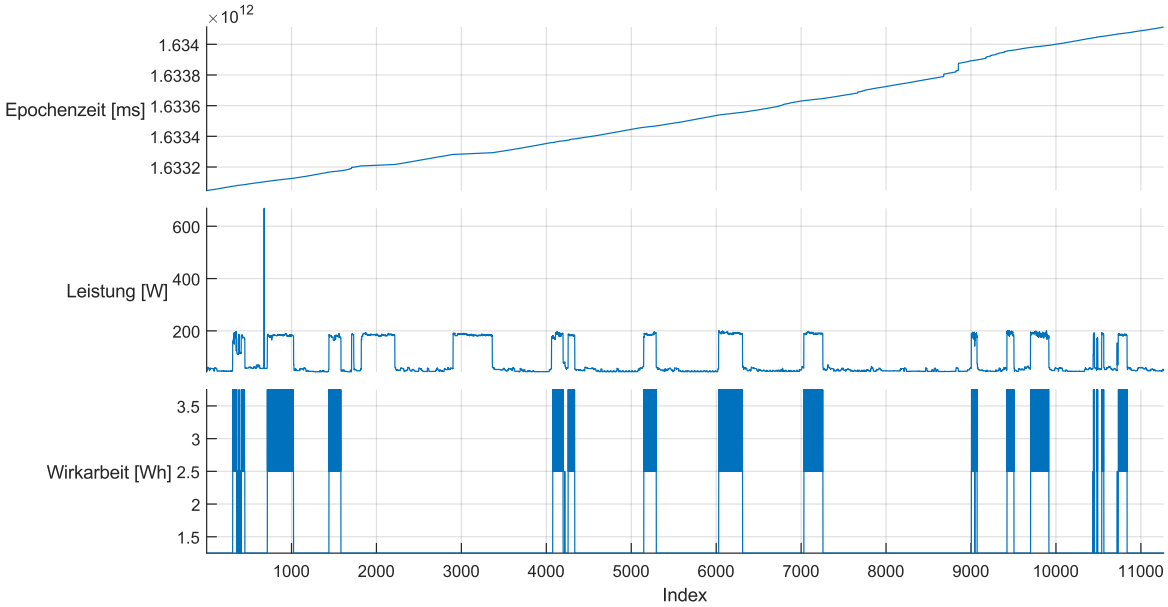


Abbildung 33 Langzeitaufnahme mit Loridane (Ausschnitt)

Mit den Graphen sind die vom Node übertragenen Parameter Leistung und Wirkarbeit über ihre Indizes aufgetragen. Die Epochenzeit wurde den Datensätzen vom Loridane-Server hinzugefügt und entspricht einem Unix-Zeitstempel[24]. Der zeitliche Umfang des Datensatzes beträgt 17.5 Tage. Dabei sind die vier Prozesse Leerlauf, Lese- und Schreibvorgang sowie Durchführung von Berechnungen zu identifizieren. Bei letzterem Prozess ist der Leistungsunterschied deutlich erkennbar (~180W), die Auflösung der aus Intervall und Wirkarbeit berechnete Leistung ist aber hoch genug, um auch die Festplattenzugriffe aus dem Standby zu identifizieren (Anhang IX).

6.1.1 Interpretation der Graphen

Fällt die Anschlussleistung unter 75 Watt ab, ist das Intervall zwischen zwei Impulsen länger als 60 Sekunden, da dann weniger als 1.25 Wh/min umgesetzt werden. Der Node verzögert in diesem Fall die Übermittlung bis zum nächsten Impuls. Die Schwankung in der Steigung des Graphen der Epochenzeit ist daraus abzuleiten. Augenfällig ist der Sprung in der Epochenzeit bei Index 8860. Hier hat das Gateway die Netzwerkverbindung zum Server für 12.6 Stunden verloren. Der dem zugrunde liegende Softwarefehler wurde im Nachgang behoben.

Der Leistungsgraph zeigt einen Ausreißer bei Index 678. Es liegt die Vermutung nahe, dass ein Impuls durch den Node zweimal gezählt wurde. Spezifikationsgemäß beträgt die Impulslänge der ORNO-Zählers 90ms. Die Länge des Sperrintervalls auf dem Node beträgt 110ms ab der ersten High-Evaluation des Pins. Es ist im Nachgang nicht zu rekonstruieren, ob der Energiezähler einen zu langen Impuls ausgegeben hat oder ein Fehler im Signalpfad vorliegt. Die Einführung der Serienterminierung (s. 5.4.2) minimiert diese Problematik auf ein statistisch nicht relevantes Maß (vgl. folgender Abschnitt).

[24] Millisekunden seit 01.01.1970 00:00:00.000 UTC

Im unteren Graphen ist der Arbeitsumsatz je Sendeintervall aufgetragen. Entsprechend der Impulswertigkeit und im Rahmen der Genauigkeit des ORNO-Zählers geben die Daten den tatsächlich gemessenen Arbeitsumsatz an und unterliegen daher keiner Unsicherheit bezüglich arithmetischer Operationen auf dem Node.

6.2 Evaluation der Netzwerkperformanz und -Zuverlässigkeit

Basierend auf den Daten des Langzeitversuchs und weiteren Versuchszyklen wird in diesem Abschnitt die Leistungsfähigkeit von Loridane quantifiziert.

6.2.1 Testprinzip

Es soll überprüft werden, mit welcher Konfidenz Daten über das Loridane Netzwerk übertragen werden können. Ebenso ist zu überprüfen, ob die Netzwerkkapazität den theoretischen Vorhersagen aus Abschnitt 5 genügt. Dazu wurden mehrere Testläufe durchgeführt.

Anzahl der Nodes n_n	Anzahl der Gatways n_g	Ist-Paketübermittlungsrate R_{pd} [pkt/s]
1	1	0.011
3	1	1.965
3	1	5.893
3	1	6.530

Tabelle 8 Versuchsdaten Loridane Langzeit- und Belastungstest

Tabelle 8 zeigt die Netzwerkbelastung und die Parameter der Übertragungen. Die Daten aus Zeile 1 der Tabelle stammen aus der Analyse des Langzeitversuchs (s. 6.1).

Zur Generierung der Daten in Zeile 2 bis 4 wurden drei Nodes und ein Gateway an Loridane angebunden und in getrennten Versuchsreihen zwischen 1.965 und 6.53 Pakete je Sekunde an den Server übermittelt. Die LoRa-Einstellung war dabei stets 867.0 MHz und SF7, während die Sendeleistung und örtliche Aufstellung der Nodes so gewählt wurde, dass sich am Gateway eine RSSI von rund -100 dBm einstellt. Dies entspricht einem realitätsnahen Betriebsszenario, unter der Annahme, dass der Spreading Factor so klein wie möglich gewählt wird, aber die RSSI mindestens 20dB oberhalb der bestenfalls erwartbaren Empfängersensitivität liegt. In allen Versuchen lag die SNR zwischen 8.75 und 9.5 dB.

6.2.2 Analyse der Daten aus dem Langzeitversuch

Während des Langzeitversuchs wurden tatsächliche Messwerte, keine Emulierten, übertragen. Diese waren stets entsprechend des Uplink-Protokolls gepackt (vgl. Code 5) und übertrugen daher die Node UID, die Wirkleistung und die berechnete mittlere Leistung im Intervall. Aufgrund dessen kann es auf dem Node sowohl zu arithmetischen Fehlern als auch zu Fehlern bei der Impulsauswertung kommen. Um diese Datensätze als verloren zu berücksichtigen, ist in Tabelle 10 der prozentuale Paketverlust durch arithmetische Operationen auf dem Node als L_{arith} eingeführt. Verluste, die

aufgrund von Übertragungsfehlern in der Funkstrecke zustande kommen sind als L_{UL} kategorisiert. Dazu wurden die Datenpunkte aus der Loridane Datenbank exportiert und mittels eines Python Skripts analysiert. Die Zuordnung eines Datenpunktes zu der jeweiligen Fehlerkategorie erfolgt, wenn:

E_{arith}	E_{UL}
die Datenpunkte außerhalb des Erwartungsintervalls liegen.	die empfangene UID nicht korrekt ist.
keine Korrelation zwischen Zeitintervall, Wirkarbeit und Leistung besteht.	der Datensatz Nicht-ASCII-Zeichen enthält.
der Leistungswert die Größe inf[25] annimmt.	der Datensatz fehlt.
der Wirkarbeitswert kein Vielfaches von 1.25 ist.	der Datensatz andere Zeichen als Dezimalpunkte und Ziffern enthält.

Tabelle 9 Fehlerkategorien der Langzeit-Datenanalyse

Ist der Datensatz vorhanden und es besteht keine Korrelation der Datenpunkte wird der Fehler hier als arithmetisch kategorisiert. Besteht keine Korrelation von Wirkarbeit und Leistung zur vergangenen Intervalllänge gilt der Datensatz dazwischen als, bei Empfang, verloren (E_{UL}). Korrelation besteht dann, wenn sich der Zeitstempel des Datensatzes innerhalb einer ε-Umgebung von 2% der erwartbaren Intervalllänge seit dem letzten Datensatz befindet.

6.2.3 Analyse der Daten aus dem Belastungstest

Die Quantifizierung des Anteils erfolgreich übertragener Datensätze im Belastungstest erfolgt nach einem ähnlichen Schema wie bei der Langzeitanalyse. Für Tests bei hohen Paketübermittlungsraten, die an der Obergrenze der Netzwerkkapazität von Loridane liegen, stehen aufgrund der großen Anzahl an benötigten Nodes keine Realdaten zur Verfügung. Daher wurde auf den Nodes eine Testsoftware verwendet. Diese gleicht der Software für S0-Schnittstellen, außer bei der Generierung der Sensorwerte, völlig. Die Messwerte sind vom Node emuliert. Dazu wird als Wert für die Wirkarbeit ein, paketweise inkrementierter, Zähler verwendet, und als Leistungswert ein numerisch generierter Zufallswert in Gleitkommaformat zwischen 1.00 und 24000.00 (vgl. Code 18).

```
// Beispiel Realwert
NO94B97EC0C46C3.75;202.76

//Beispiele emuliert
NO94B97EC0C46C123;23412.00
NO94B97EC0C46C124;3235.86
```

Code 18 Beispiel Testdatensätze (txt)

[25] Zahldarstellung für „unendlich“ in C/C++

Aus informationstechnologischer Sicht sind die Datensätze strukturell identisch und somit dazu geeignet die UL-PDR in vergleichbarer Weise zu bestimmen. Allerdings entfällt bei den emulierten Werten das Auftreten von arithmetischen Fehlern. Aufgrund des Zählers im jeweils ersten Datenpunkt kann jedoch ermittelt werden, welche Pakete nicht auswertbar oder aufgrund von Fehlern in der Funkstrecke verloren sind.

6.2.4 Versuchsergebnisse

Das verwendete Python-Skript zur Analyse der Datensätze gibt eine Textdatei der Form

```
Lost Packages:
154 of 15671
which equals
0.9827069108544445 %

The amount of arithmetic Errors is
85 of 15671
which equals
0.5424031650819986 %
Total: 1.525110075936443 %
```

Code 19 Ausgabe eines Testergebnisses (txt)

aus. In Tabelle 10 sind die Ergebnisse numerisch dargestellt.

Die Paketübermittlungsrate von 6.53 pkt/s entspricht dem prinzipbedingten Maximum nach Abschnitt 5.8 für 3 Nodes und ein Gateway. In Abschnitt 2.3.3 ist der Bezug zur Leistungsfähigkeit von LoRaWAN ohne Implementierung eines Synchronisationssystems hergestellt, die auf der mathematischen Modellierung von Magrin et al. basieren.

n_n	n_g	R_{pd} [pkt/s]	$RSSI$ [dBm]	L_{arith} [%]	L_{UL}[%]	L_{sum} [%]	$ULPDR$ [%]	n_{data}
1	1	0.011	-57 [-64; -53]	0.542	0.983	1.5	99.017	15761
3	1	1.965	-101 [-103; -95]	--	3.1	3.1	96.9	1748
3	1	5.893	-102 [-104; -97]	--	3.7	3.7	96.3	7115
3	1	6.530	-102 [-104; -97]	--	6.5	6.5	93.5	1024

Tabelle 10 Versuchsergebnisse

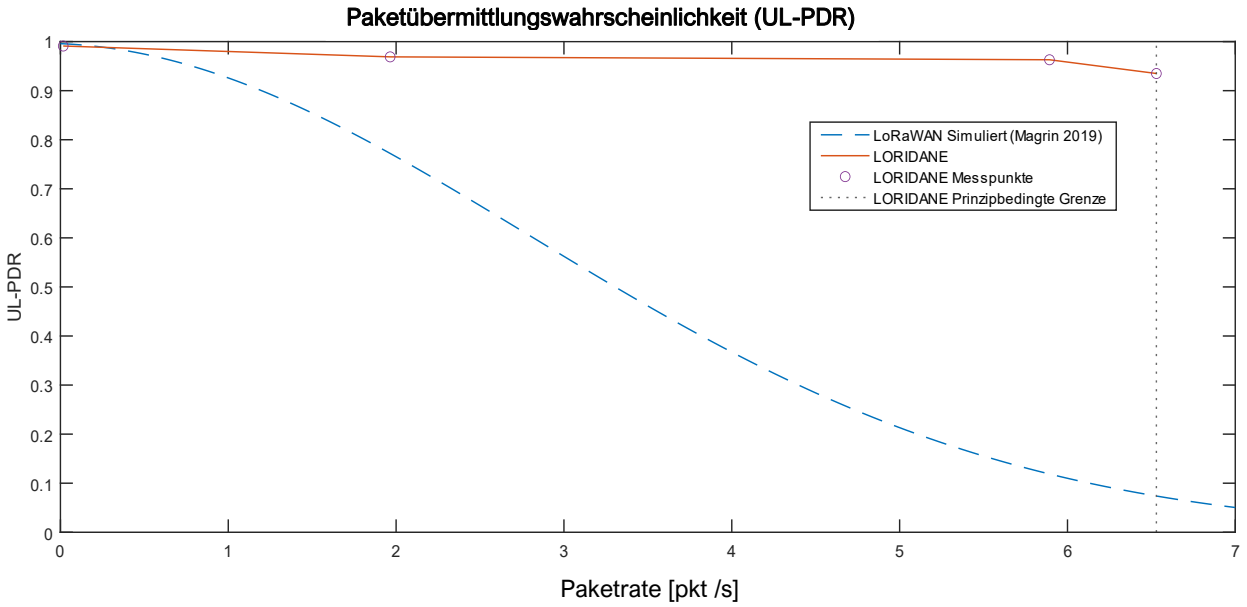


Abbildung 34 Vergleich der Wahrscheinlichkeiten für erfolgreiche Paketübermittlung bezogen auf die Uplinkrate

Abbildung 34 zeigt die Kurve für LoRaWAN-Uplinks ohne Konfirmierung der Nachricht qualitativ nach (Magrin et al. 2021, S. 28). Die zugrundeliegende Parametrierung der Übertragungsart, ohne Empfangskonfirmierung und ohne Wiederholung, entspricht einem Uplink-Vorgang in Loridane. Der für Loridane erstellte Graph zeigt eine lineare Verknüpfung der Ist-Werte. Diese sind in der Abbildung durch die vier Messpunkte dargestellt. Es ist zu beachten, dass die Empfangshardware der Untersuchungen von Magrin et al., entsprechend des Graphen in der Abbildung acht parallele Empfangspfade exklusive der Downloadpfade verwendet (Magrin et al. 2019, S. 4). Das Loridane-Gateway verwendet *einen* gemeinsamen Pfad für Uplink und Downlink.

So zeigt sich, dass Übertragungsfehler mit dieser Zeitscheibenimplementierung nicht verhindert, aber deutlich vermindert werden können.

6.3 Kostenstruktur

Im Beispiel erfolgt hier die Aufstellung der voraussichtlichen Kosten in einer Kostenschätzung für eine Installation von zehn LoRaWAN Impulszählern, einem LoRaWAN-Gateway unter Verwendung eines kostenfreien LoRaWAN Serverdienstes (Tabelle 11) in Gegenüberstellung mit einer Lösung von zehn Loridane Nodes mit S0-Software, einem Loridane-Gateway und einem Loridane-Server (Tabelle 12). Elektrische Wirkverbrauchszähler werden für beide Implementierungen gleichermaßen benötigt und sind daher von der Betrachtung ausgenommen. Die Summe aus direkten und indirekten Personalkosten zur Konfektionierung der Loridane-Hardware wird zu 42€/h veranschlagt. Für die Einrichtung von Software wird dieser Satz bei beiden Systemen ebenfalls zu Grunde gelegt.

Kostenschätzung LoRaWAN		10 Nodes					
Kostengruppe	Kostenart	Einheit	Wert €	Anzahl	Faktor	Gesamt €	Gruppensumme
Personal	Einrichtung Serverdienst	10min	7,00 €	1,00	6	42,00 €	
	Einrichtung Nodesoftware	10min	7,00 €	10,00	0,5	35,00 €	
	Einrichtung Gatewaysoftware	10min	7,00 €	1,00	3	21,00 €	
	Einrichtung Hilfssoftware	10min	7,00 €	1,00	3	21,00 €	
	Überspielen Nodesoftware	10min	7,00 €	0,00	0	0,00 €	
	Überspielen Gateway Software	10min	7,00 €	1,00	9	63,00 €	
	Lötarbeiten	10min	7,00 €	0,00	0	0,00 €	
	Bohrarbeiten	10min	7,00 €	0,00	0	0,00 €	182,00 €
Primäres Material	LoRaWAN Impulszähler, z.B. [PL2]	1Stk.	110,00 €	10,00	1	1.100,00 €	
	Gateway, z.B. [PL15]	1Stk.	399,00 €	1,00	1	399,00 €	
	Signalkabel	1m	2,00 €	10,00	1	20,00 €	1.519,00 €
Hilfsmaterial	Pauschal	1Stk.	1,00 €	11,00	1	11,00 €	11,00 €
Summe:						1.712,00 €	

Tabelle 11 Kostenschätzung LoRaWAN für eine Installation an 10 Produktionsmaschinen

Kostenschätzung Loridane		10 Nodes					
Kostengruppe	Kostenart	Einheit	Wert €	Anzahl	Faktor	Gesamt €	Gruppensumme
Personal	Einrichtung Server	10min	7,00 €	1,00	6	42,00 €	
	Einrichtung Nodesoftware	10min	7,00 €	0,00	0	- €	
	Einrichtung Gatewaysoftware	10min	7,00 €	1,00	1	7,00 €	
	Einrichtung Hilfssoftware	10min	7,00 €	1,00	2	14,00 €	
	Überspielen Nodesoftware	10min	7,00 €	10,00	0,2	14,00 €	
	Überspielen Gateway Software	10min	7,00 €	1,00	1	7,00 €	
	Lötarbeiten	10min	7,00 €	10,00	0,6	42,00 €	
	Bohrarbeiten	10min	7,00 €	10,00	0,3	21,00 €	147,00 €
Primäres Material	ESP32/LoRa Boards	1Stk.	30,00 €	11,00	1	330,00 €	
	Gehäuse	1Stk.	10,00 €	11,00	1	110,00 €	
	Raspberry Server	1Stk.	65,00 €	1,00	1	65,00 €	
	Signalkabel	1m	2,00 €	10,00	1	20,00 €	525,00 €
Hilfsmaterial	Pauschal	1Stk.	1,00 €	11,00	1	11,00 €	11,00 €
Summe:						683,00 €	

Tabelle 12 Kostenschätzung Loridane für eine Installation an 10 Produktionsmaschinen

Unter Annahme der oben genannten Voraussetzungen und den Kostenschätzungen betragen die Installationskosten für Loridane nicht ganz 40% einer Installation mit marktüblichen LoRaWAN-Produkten. Trotz des konservativ hoch eingeschätzten Stundenlohns und dem hohen Anteil an Eigenleistung / Auftragsarbeit bei der Konfektionierung der Hardware, ergibt sich für keine Anzahl an Installationsstellen ein Break-Even. Der Kostenvorteil bleibt auf der Seite von Loridane mit einem asymptotischen Kostenverhältnis von circa 45% für eine große Anzahl Nodes.

6.4 Einsatzgebiete für Loridane

Die Protokollstruktur von Loridane ist so gewählt, dass sowohl numerische als auch alphanumerische Nutzdaten übertragen werden können. Dies ermöglicht einen Einsatz über die Übermittlung von elektrischen Verbrauchsdaten hinaus. Ebenso ist der Versand von Zustandsdaten verschiedener Art möglich. Der modulare Aufbau der Node-Software sieht vor, dass externe Sensorik angebracht werden kann und eine Ergänzung der Software möglich ist. Obwohl Downlink-Nachrichten vorgesehen und prinzipiell zu gegebener Zeit gewollt sind, liegt der Fokus bei Loridane auf der Datenaggregation durch Uplinks. Aufgrund der Netzwerkeigenschaft völlig entkoppelt vom Internet betrieben werden zu können, ist eine Topologie nach dem Prinzip des Loridane-Netzwerkes auch dort einsetzbar, wo schwerlich oder keine Internetanbindung zur Verfügung steht. An entfernten Außenstellen oder auf großen Schiffen, und vom ausschließlich industriellen und wissenschaftlichen Kontext losgelöst beispielsweise zum Datensammeln in Gebäudekomplexen oder in Zugangskontrollen.

7 Fazit und Ausblick

Es ließ sich mit Loridane in einer realen Implementierung zeigen, dass es möglich ist die Wahrscheinlichkeit von nicht-konfirmierten, erfolgreichen Funkdatenübertragungen gegenüber marktüblichen LoRa-Verfahren erheblich zu steigern und damit gleichzeitig die Verlässlichkeit bezüglich des Datenempfangs zu erhöhen. Nichtsdestotrotz steht Loridane nicht in direkter Konkurrenz zu LoRaWAN-Anwendungen, die bezüglich der Mobilität der Nodes und der Fähigkeit zu Langzeitbatteriebetrieb gewisse Vorteile bieten. Vielmehr ist mit Loridane eine Lösung dargestellt, die in einem lokal statischen Umfeld hochleistungsfähig ist. Dabei kann auf Drittanbieter und Clouddienste im Betrieb völlig verzichtet werden. Dennoch ist es möglich einen Server für mehrere über das Internet verbundene Standorte zu betreiben. Damit gelingt es viele positive Eigenschaften aus beiden Ansätzen zusammenzubringen. Darüber hinaus ist der Schwierigkeitsgrad in der Installation, aufgrund vieler Automatismen, nicht erheblich von kommerziellen Lösungen abweichend, aber an mancher Stelle verbessernd. Dabei wurden alle Fest- oder Bereichsforderungen (s.3.4) erfüllt oder übertroffen. Die Wunschforderungen 24, 29 und 30 waren bei der späteren Realisierung begründet infrage zu stellen. Eine dynamische Bandbreitenparametrierung (W24) ergibt keine nennenswerten Vorteile in Bezug auf die schlussendlich realisierte Netzwerkkapazität, die mit der gewählten Einstellung in vielen Anwendungsfällen voraussichtlich nicht vollständig genutzt wird. Die Fähigkeit zur drahtlosen Durchführung von Firmwareupdates (W29) lässt Sicherheitsbedenken entstehen, deren Notwendigkeit mit ihren Vorteilen abzuwägen war. Die Fähigkeit des Servers zu implementieren die Node- und Gateway Software zum Überspielen auf die MyC-Hardware bereitzuhalten (ZF30) würde bedingen, dass der Anwender die individuellen Einstellungen in einer kommandozeilenbasierten Umgebung vornehmen müsste. Bei der Implementierung von Loridane wurde bewusst, wo möglich, auf diese Notwendigkeit verzichtet und einer intuitiven Bedienbarkeit der Vorzug gegeben. Die Vorkonfiguration der Netzwerkteilnehmer über eine grafische Oberfläche an einem Desktop-Computer lässt dies eher zu.

Dennoch ist eine hohe Funkreichweite durch die LoRa-Technik möglich. In vielen Fällen bedeutet das ebenso eine große räumliche Streuung der Signale über den eigenen Standort hinaus. Das erfordert einen verantwortungsvollen Umgang mit den Fähigkeiten eines hochkapazitiven Netzwerkes, in einem, wie in diesem Fall gegeben, lizenzfreien Funkband. So ist für die Parametrierung der einzelnen Teilnehmer stets abzuwägen, wie kurz das Sendeintervall eines Nodes sein muss, um aussagekräftige Daten zu erhalten und welche Sendeleistung für eine zuverlässige Übertragung notwendig ist.

Ein Angriffspunkt zur Optimierung zum vermehrten Versand von Instruktionen via Downlink bestünde in der Zuweisung längerer Zeitslots oder die Zuweisung mehrerer Downlink-Slots an die Gateways je Zeitscheibe. Dadurch würde Loridane um die Fähigkeit erweitert bidirektionale Maschine-zu-Maschine-Kommunikation bei Bedarf zu ermöglichen.

Die Verwendung der Loridane-Software zur Erfassung anderer Zustandsdaten als dem elektrischen Wirkverbrauch würde bedingen, dass die Node-Software um Module zur Verwendung anderer Sensoren erweitert wird. Dazu gehören zum Beispiel Füllstandssensoren für Vorratsbehältnisse, Volumenstromsensoren für flüssige Verbrauchsgüter und Druckluft. Die Serversoftware kann die übermittelten numerischen Werte ungeachtet ihres Entstehungszusammenhangs verarbeiten. Damit

wird dem Anspruch genügt ein effektives Mittel zu implementieren, um Umsatzmonitoring künftig für alle stationär automatisiert erfassbaren Verbrauchsgüter zu realisieren.

Nun besteht mit Loridane die Möglichkeit zur effektiven Datenübertragung via LoRa-Funk. Die Herausforderung besteht darin, die gesammelten Daten zu verarbeiten, um ebenso effektiv Nutzen aus ihnen ziehen zu können. Aufgrund der Möglichkeit mit Loridane zeitlich hochaufgelöste energetische Prozessüberwachung zu betreiben und der guten Wirtschaftlichkeit einer Installation, ist der Hemmschuh ein weiteres Stück gelöst, nicht nur Daten großflächig und, dem IoT-Prinzip folgend, „smart“ zu erfassen, sondern die Datengrundlage vermehrt auf ein verstärkt energieeffizientes Handeln rückzukoppeln und in eine automatisierte effiziente Produktionsplanung sowie in eine starke Entscheidungsgrundlage zu überführen.

Anhang

Anhang: Elektronische Daten

Datei	Programm	Beschreibung
\Ausarbeitung 20211124_StebeLindsay_Thesis_Loridane.pdf	Adobe Reader	Dieses Dokument
\Datenblätter janitza-bhb-umg96rm-20-250V-de.pdf	Adobe Reader	Bedienungsanleitung Janitza UMG96RM
\Datenblätter janitza-bhb-umg96rmpn-de.pdf	Adobe Reader	Datenblatt / Anleitung Janitza UMG96RM
\Datenblätter OR-WE-513,516,517,520_manual PL_EN_DE.pdf	Adobe Reader	Datenblatt ORNO Wirkverbrauchszähler
\Datenblätter OR-WE-520_Certyfikat_MID_ModB.pdf	Adobe Reader	MID Zertifikat ORNO Wirkverbrauchszähler
\Datenblätter sx1276_77_78_79.pdf	Adobe Reader	Datenblatt SX1267 LoRaTransceiver
\Hilfsprogramme\Matlab loratoa.m	Matlab	Matlab-Function, implementiert Gleichung 2-13 und berechnet die Time on Air für LoRa Pakete
\Hilfsprogramme\Matlab packetps.m	Matlab	Matlab-Script, berechnet maximale Paketübermittlungsraten in Loridane
Software Loridane\Arduino\dummy_node_loridane dummy_node_loridane.ino	Arduino IDE	"Dummy"-Testsoftware für Loridane-Nodes. Messwerte sind emuliert.
Software Loridane\Arduino\gateway_loridane gateway_loridane.ino	Arduino IDE	Betriebssoftware für das Loridane-Gateway.
Software Loridane\Arduino\node_S0_loridane node_S0_loridane.ino	Arduino IDE	Betriebssoftware für Loridane-Nodes als Impulszähler an S0-Schnittstellen
Server\Installationsordner\setupLoridane fetchNR.sh	Linux Terminal (Shell)	Shell-Skript, lädt die aktuelle Version von Node Red. Wird automatisch von loridaneinstall.sh ausgeführt
Server\Installationsordner\setupLoridane flows_loridane.json	Node Red	Hält das Loridane Programm zur Ausführung in Node Red
Server\Installationsordner\setupLoridane flows_loridane_cred.json	Node Red	Hält verschlüsselte Standardzugangsdaten
Server\Installationsordner\setupLoridane loridaneConfig.json	Loridane	Ist Basiskonfiguration für die Loridane Software, wird bei Installation auf den Server kopiert.
Server\Installationsordner\setupLoridane loridaneinstall.sh	Linux Terminal (Shell)	Ist das Installationsskript für die Loridane-Software, für alle Hilfsprogramme und Abhängigkeiten
Server\Installationsordner\setupLoridane settings.js	Loridane	Hält die korrekte Grundkonfiguration für Node Red zur Ausführung des Loridane Programms
Server\Text Exports LoridaneServer_backend_logic.pdf	Adobe Reader	Enthält keine ausführbare Software, aber enthält die Serverlogik ohne User-Interface-Funktionen zur Ansicht ohne Node Red Editor

Dateistruktur der elektronischen Anhänge:

```
├──Ausarbeitung
│   └──20211124_StebeLindsay_Thesis_Loridane
├──Datenblätter
│       esp32_datasheet_en.pdf
│       janitza-bhb-umg96rm-20-250V-de.pdf
│       janitza-bhb-umg96rmpn-de.pdf
│       OR-WE-513,516,517,520_manual PL_EN_DE.pdf
│       OR-WE-520_Certyfikat_MID_ModB.pdf
│       sx1276_77_78_79.pdf
│
├──Hilfsprogramme
│   └──Matlab
│           loratoa.m
│           packetps.m
│
└──Software Loridane
    ├──Arduino
    │   │   Analog zu Github repository
    │   │
    │   ├──dummy_node_loridane
    │   │   │   cipher.cpp
    │   │   │   credentials.h
    │   │   │   dummy_node_loridane.ino
    │   │   │   encrypt.h
    │   │   │   functions.h
    │   │   │   init.h
    │   │   │   sensor.h
    │   │   │
    │   │   └──data
    │   ├──gateway_loridane
    │   │       credentials.h
    │   │       functions.h
    │   │       gateway_loridane.ino
    │   │       init.h
    │   │       persist.h
    │   │       receive.h
    │   │       send.h
    │   │       serialinput.h
    │   │
    │   └──node_S0_loridane
    │           cipher.cpp
    │           credentials.h
    │           encrypt.h
    │           functions.h
    │           node_S0_loridane.ino
    │           sensorS0.h
    │           │
    │           └──data
    └──Server
        ├──Installationsordner
        │   │   Analog zu Github repository
        │   │
        │   └──setupLoridane
        │           fetchNR.sh
        │           flows_loridane.json
        │           flows_loridane_cred.json
        │           LICENSE
        │           loridaneConfig.json
        │           loridaneinstall.sh
        │           README.md
        │           settings.js
        │
        └──Text Exports
            Diese Dateien sind nicht ausführbar!
            LoridaneServer_backend_logic.js
            LoridaneServer_backend_logic.pdf
            loridane_server_version_hash.txt
```

ANHANG I: LORIDANECONFIG.JSON

```
{
"_comment": "the homepath to your Loridane should be set here, this can not be empty, close the
directories with a / please",
"path":{
    "config":"/home/pi/LORIDANE/config/",
    "database":"/home/pi/LORIDANE/database/"
    },
"_comment":"set here if the payloads of the lora nodes should be encrypted or not, key and
initalisation vector must be a 16byte hex-tring and the same in your nodes",
"encryption":{
    "engaged":true,
    "key":"abcdefghijklmnop"
    },
"_comment":"The initial frequency the gateway should be listening to, this overrides the gateway
setting",
"freq":867e6,
"csvDelimiter":";",
"syncInterval":233,
"_comment":"If the Gateway is attached to USB at the server the ui configuration can be automated,
then add credentials and set the flag to true. Data is NOT encrypted",
"gatewayConfig":{
    "automatic":false,
    "ssid":"Loridane-02",
    "wpwd":"a-password",
    "broker":"192.168.4.1",
    "user":"mqtusr",
    "mpwd":"a-password"
    }
}
```

ANHANG II: MEMCACHELORIDANE.JSON

memcacheLoridane.json und globaler Scope


```
{
    "LinuxUsername": "pi",
    "blockONstart": false,
    "classes": {},
    "funcs": {
      "math": {}
    },
    "values": {
      "allowedFreq": [
        {
          "min": 863,
          "max": 864.875,
          "factor": 1000
        },
        {
          "min": 865,
          "max": 868.475,
          "factor": 100
        },
        {
          "min": 868.7,
          "max": 869.025,
          "factor": 1000
        },
        {
          "min": 869.4,
          "max": 869.525,
          "factor": 1000
        },
        {
          "min": 869.7,
          "max": 869.875,
          "factor": 100
        }
      ],
      "freq": {
        "min": 863,
        "max": 869.875
      }
    },
    "admit": [],
    "devices": {
      "gw": [
        {
          "uid": "GW94B97EC0C31C",
          "type": "GW",
          "freq": 867000000,
          "sf": 7,
          "friendlyname": "Gateway Outdoor",
          "interval": 0,
          "lastSeen": 1635512093184
        },
        {
          "uid": "GWF008D1C8D658",
          "type": "GW",
          "freq": 867000000,
          "sf": 7,
          "interval": 0,
          "lastSeen": 1635941007651,
          "nextGW": "GWF008D1C8D658"
        }
      ],
```

```
    "nodes": [
        {
          "uid": "NO94B97EC0C46C",
          "type": "energy",
          "freq": 867000000,
          "sf": 7,
          "interval": 78613,
          "nextGW": "GWF008D1C8D658",
          "tx": 20,
          "lastinterval": 78835,
          "lastSeen": 1635941007651,
          "friendlyname": "Energy Meter"
        },
        {
          "uid": "NOF008D1C8DC20",
          "type": "energy",
          "freq": 867000000,
          "sf": 7,
          "interval": 600321,
          "nextGW": "GW94B97EC0C31C",
          "tx": 20,
          "lastinterval": 600617,
          "lastSeen": 1635448194210
        },
        {
          "uid": "NO84CCA84ADF70",
          "type": "energy",
          "freq": 867000000,
          "sf": 7,
          "interval": 0,
          "nextGW": "GWF008D1C8D658",
          "tx": 20,
          "lastinterval": 0,
          "lastSeen": 1635779323285
        }
      ]
    },
    "timedisk": {
      "size": 509,
      "count": 5,
      "disk": {
        "NO94B97EC0C46C": {
          "uid": "NO94B97EC0C46C",
          "slot": 0,
          "tdsize": 509
        },
        "NO84CCA84ADF70": {
          "uid": "NO84CCA84ADF70",
          "slot": 2,
          "tdsize": 509
        },
        "NOF008D1E8D698": {
          "uid": "NOF008D1E8D698",
          "slot": 3,
          "tdsize": 611
        },
        "NOF008D1C8DC20": {
          "uid": "NOF008D1C8DC20",
          "slot": 1,
          "tdsize": 509
        },
        "NOF008D1E8D658": {
          "uid": "NOF008D1E8D658",
          "slot": 5,
          "tdsize": 814
        },
        "NO84CCA82ADF70": {

        …
```

```
"NO84CCA84ADF70": {
        "lastreading": {
          "measure0": "NO84CCA84ADF70",
          "time": "1.11.2021, 16:08:43",
          "timestamp": 1635779323285,
          "uid": "NO84CCA84ADF70"
        }
      },
      "NO14B93EC0C46C": {
        "lastreading": {
          "measure0": "1.25",
          "measure1": "55.53",
          "time": "12.10.2021, 03:28:00",
          "timestamp": 1634002080174
        }
      },
      "NO14B97EC0C46C": {
        "lastreading": {
          "measure0": "1.25",
          "measure1": "44.20",
          "time": "19.10.2021, 10:50:48",
          "timestamp": 1634633448670
        }
      },
      "NO94B93EC0C46C": {
        "lastreading": {
          "measure0": "1.25",
          "measure1": "25847.9",
          "time": "23.10.2021, 00:44:34",
          "timestamp": 1634942674924
        }
      },
      "NOF008D1C8D658": {
        "lastreading": {
          "measure0": "NOF008D1C8D658",
          "time": "29.10.2021, 10:28:34",
          "timestamp": 1635496114823
        }
      },
      "NOF008D1E8D658": {
        "lastreading": {
          "measure0": "3.75",
          "measure1": "21608.80",
          "time": "28.10.2021, 18:13:16",
          "timestamp": 1635437596426
        }
      },
      "NO84CCA82ADF70": {
        "lastreading": {
          "measure0": "395",
          "measure1": "12584.03",
          "time": "28.10.2021, 17:43:32",
          "timestamp": 1635435812706
        }
      }
    }
  }
```

Anhang III: Zeitscheibengerechtes Versenden von Downlinks

```
function sendmsg(msg){
    node.send(msg)
}
function getSendTime(UID){
    const findGW = global.get("LORIDANE.funcs.findGW");
    const gws = global.get("LORIDANE.devices.gw") || [0];
    let amountOfGws = gws.length; //defaults to 1
    const indexGW = findGW(UID);
    let sendtime = size - timeOnAir(50) * (amountOfGws-indexGW);
    return sendtime;
}
function timeOnAir(bytes){
    let toa = Math.ceil((8 + ((bytes*8 - 4 * sf + 28) / (4 * sf) * 5)) * (2 ** sf) / (1e2));
    return toa;
}
//-----------------------------------------------------------------
// load needed data from context
const now = Date.now();
const sf = global.get("LORIDANE.devices.gw[0].sf")||7;
const TS_SYNC = [19,34,61,111,206,386];
let lastmsg = context.get("lastmsg")||0;
let startDisk = context.get("startDisk")||lastmsg;
let timedisk = global.get("LORIDANE.timedisk")||{size:1500,count:5};
let size = timedisk.size;
let timeinframe = (now-startDisk) % size;

let UID = msg.topic.substring(msg.topic.indexOf("GW"), msg.topic.indexOf("GW") + 14);
let sendtime = getSendTime(UID);
let untilsend = size - sendtime - timeinframe;

// check when the timeframe for the gateway is reached
if(timeinframe > sendtime - timeOnAir(30) || untilsend < 0){
    untilsend = 2 * size - timeinframe - sendtime;
    setTimeout(sendmsg,untilsend,msg);
}else{
    setTimeout(sendmsg,untilsend,msg);
    node.status({text:"Delayed msg for "+untilsend+" ms"});
}
context.set("lastmsg",now + untilsend);

//check if in the to-be-confirmed-array is old data
let confirm = global.get("LORIDANE.confirm")||[];
if (confirm.length > 0){
    for(var con in confirm){
        if(now - confirm[con].time >= 10*size){
            confirm.splice(con,1);
        }
    }
}

//save outgoing msgs to the confirmation scope
if(msg.payload.startsWith("NO")){
    obj = {uid: msg.payload.substring(0,14), out:msg.payload, time: now};
    confirm.push(obj);
}
//except sync msgs
if(msg.payload.includes("cn:")){
    if (!msg.payload.includes("sync")){
        nodes = global.get("LORIDANE.devices.nodes");
        for(var nd in nodes){
            obj = {uid: nodes[nd].uid, out:msg.payload, time: now};
            confirm.push(obj);
        }
    }else{
        context.set("startDisk",now+untilsend+TS_SYNC[sf-7]);
    }

}
global.set("LORIDANE.confirm",confirm);

node.status({text:"Delayed msg for "+untilsend+" ms"});
setTimeout(() => node.status({text:""}),3000);
return;
```

Anhang IV: Signalpfade im Loridane-Netzwerk

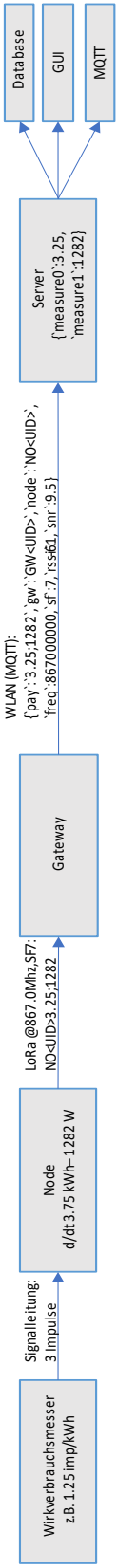


Abbildung 23 Loridane Signalpfad Uplink

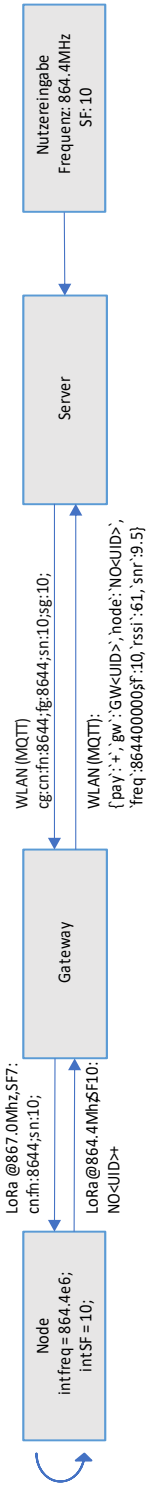
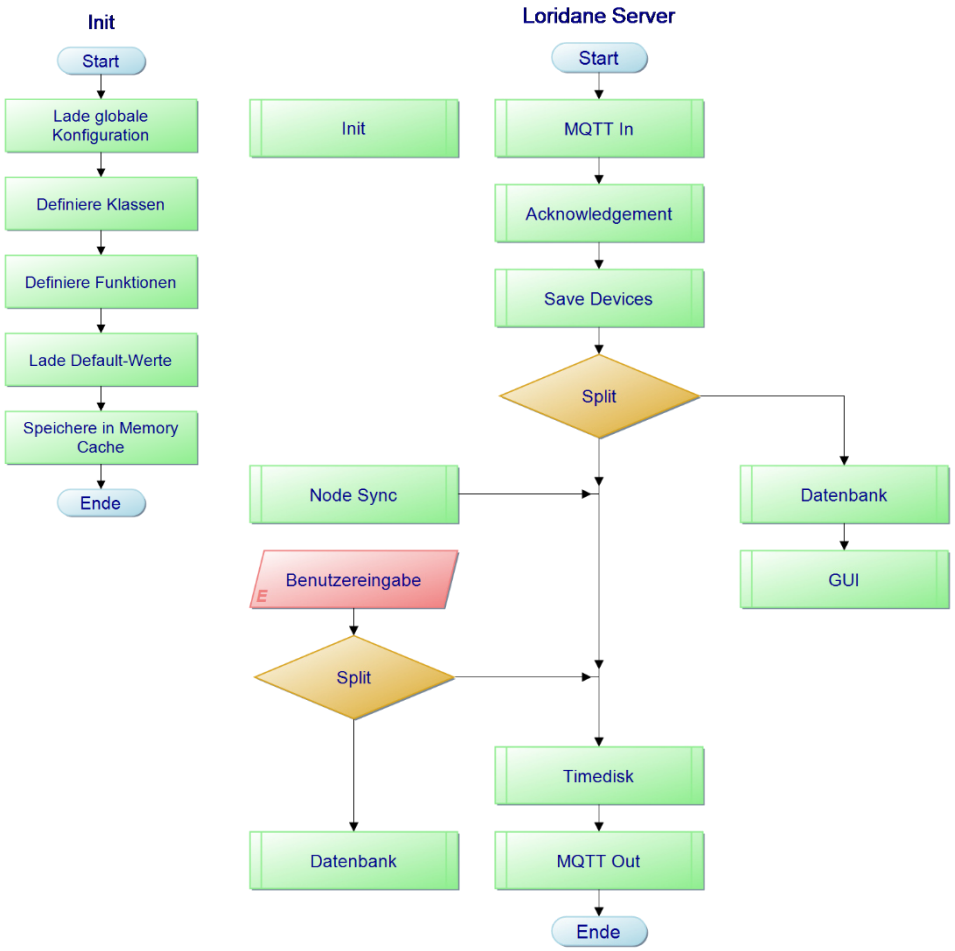
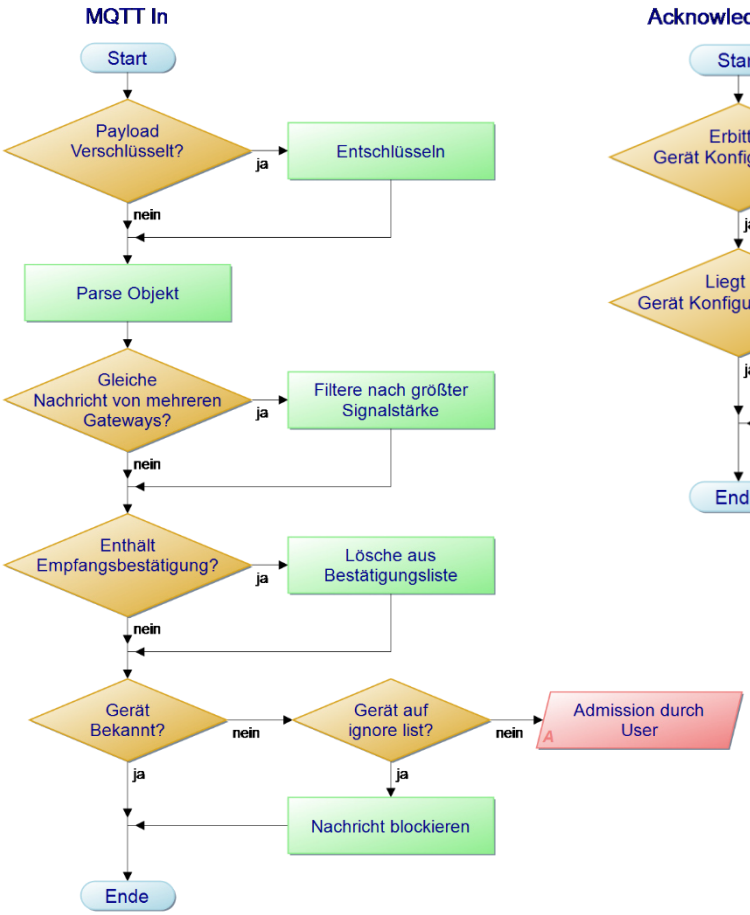
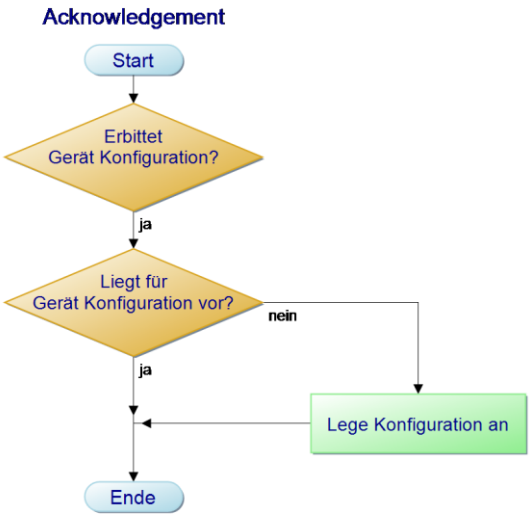


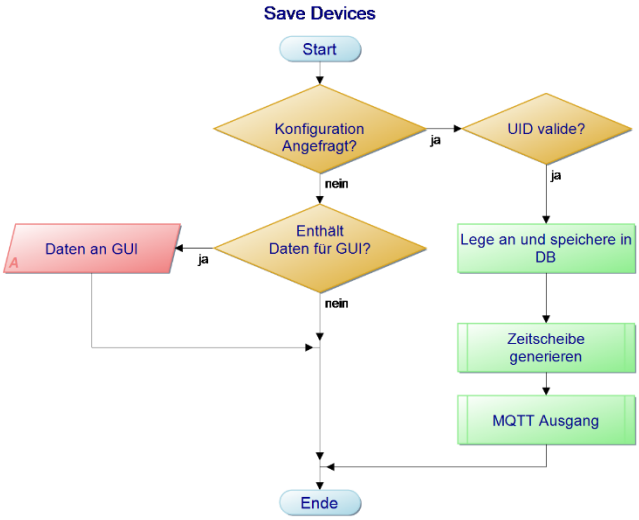
Abbildung 24 Loridane Signalpfad Downlink

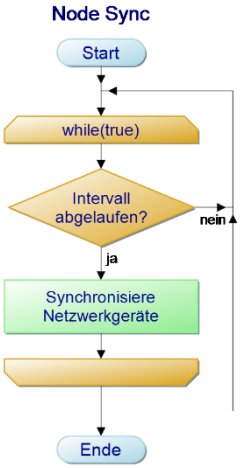
Anhang V: Programmablaufplan Loridane Server

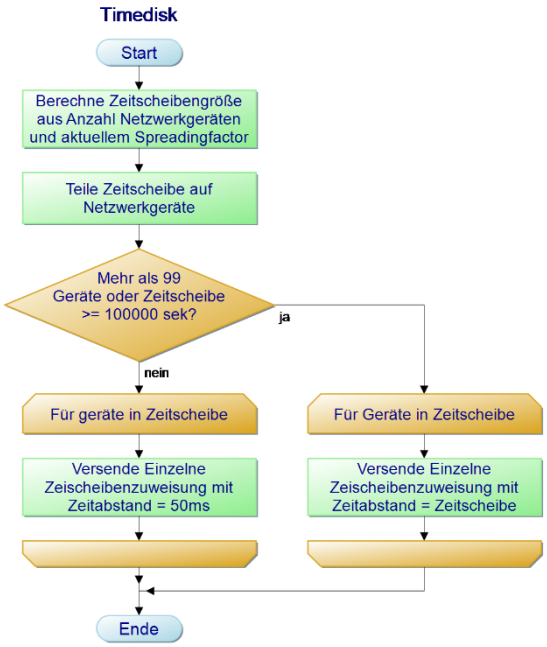


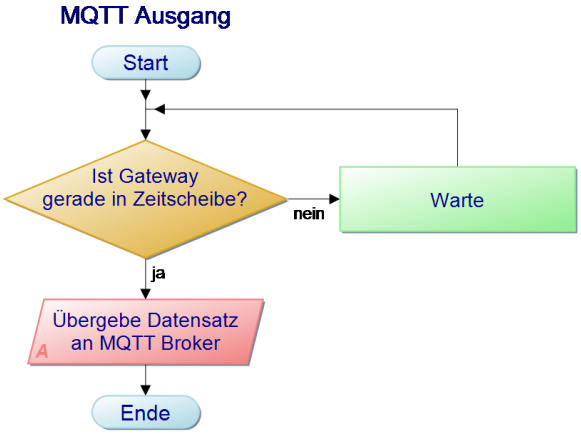

MQTT In
Start
Payload Verschlüsselt?
ja
Entschlüsseln
nein
Parse Objekt
Gleiche Nachricht von mehreren Gateways?
ja
Filtere nach größter Signalstärke
nein
Enthält Empfangsbestätigung?
ja
Lösche aus Bestätigungsliste
nein
Gerät Bekannt?
nein
Gerät auf ignore list?
nein
Admission durch User
A
ja
ja
Nachricht blockieren
Ende


Acknowledgement
Start
Erbittet Gerät Konfiguration?
ja
Liegt für Gerät Konfiguration vor?
nein
ja
Lege Konfiguration an
Ende


Save Devices
Start
Konfiguration Angefragt?
ja
UID valide?
nein
ja
Daten an GUI
A
ja
Enthält Daten für GUI?
Lege an und speichere in DB
nein
Zeitscheibe generieren
MQTT Ausgang
Ende


Node Sync
Start
while(true)
Intervall abgelaufen?
nein
ja
Synchronisiere Netzwerkgeräte
Ende

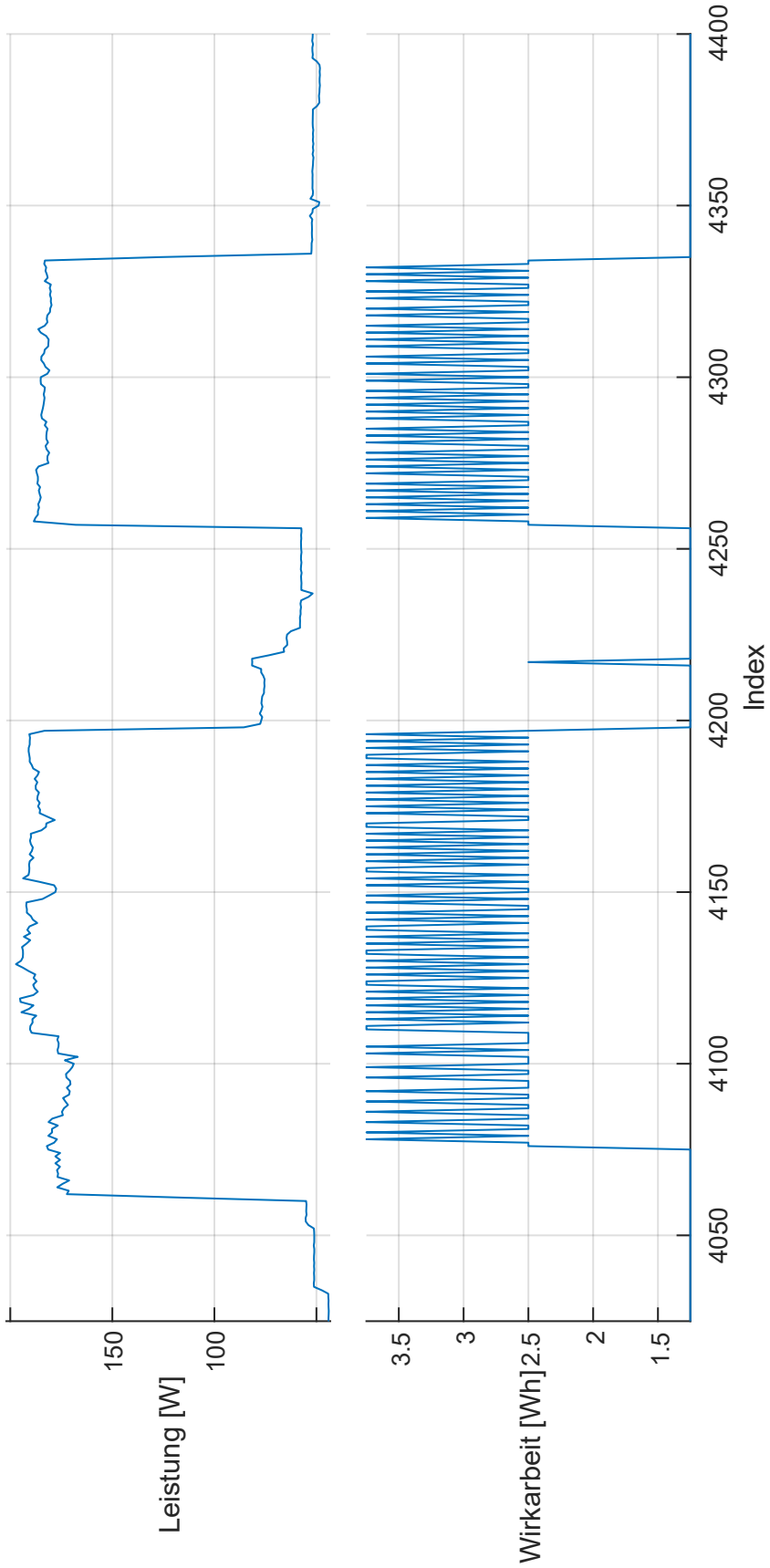

Timedisk
Start
Berechne Zeitscheibengröße aus Anzahl Netzwerkgeräten und aktuellem Spreadingfactor
Teile Zeitscheibe auf Netzwerkgeräte
Mehr als 99 Geräte oder Zeitscheibe >= 100000 sek?
ja
nein
Für geräte in Zeitscheibe
Für Geräte in Zeitscheibe
Versende Einzelne Zeischeibenzuweisung mit Zeitabstand = 50ms
Versende Einzelne Zeischeibenzuweisung mit Zeitabstand = Zeitscheibe
Ende


MQTT Ausgang
Start
Ist Gateway gerade in Zeitscheibe?
nein
Warte
ja
Übergebe Datensatz an MQTT Broker
A
Ende

Anhang VIII: Einflussfaktoren ORNO WE520 3-Phasen Wirkverbrauchszähler gemäß MID Zertifikat (ORNO-Logistic Sp. Z O.O 2014), Auszug

		Influence Factors for Temperature. Frequency & Voltage					
Current	PF Cos	-25°C	-10°C	5°C	30°C	40°C	55°C
Imin	1.0	0.65	0.60	0.25	0.16	0.31	0.36
Itr	1.0	0.59	0.46	0.42	0.42	0.48	0.60
10Itr	1.0	0.68	0.54	0.42	0.39	0.50	0.68
Imax	1.0	0.67	0.53	0.44	0.49	0.61	0.83
Itr	0.5ind	0.43	0.37	0.29	0.06	0.15	0.31
10Itr	0.5ind	0.48	0.41	0.38	0.45	0.60	0.77
Imax	0.5ind	0.45	0.34	0.26	0.34	0.47	0.65
Itr	0.8cap	0.60	0.45	0.30	0.27	0.40	0.52
10Itr	0.8cap	1.27	1.17	1.00	0.67	0.60	0.56
Imax	0.8cap	0.68	0.47	0.37	0.39	0.48	0.69
L1							
Itr	1.0	1.02	0.89	0.56	0.20	0.32	0.60
10Itr	1.0	1.01	0.77	0.51	0.17	0.34	0.64
Imax	1.0	1.04	0.76	0.48	0.16	0.33	0.64
Itr	0.5ind	1.12	0.68	0.44	0.25	0.39	0.56
10Itr	0.5ind	1.01	0.59	0.42	0.21	0.41	0.67
Imax	0.5ind	0.77	0.52	0.27	0.24	0.42	0.69
L2							
Itr	1.0	0.62	0.47	0.35	0.16	0.14	0.21
10Itr	1.0	0.49	0.40	0.31	0.12	0.13	0.25
Imax	1.0	0.44	0.32	0.18	0.11	0.20	0.38
Itr	0.5ind	0.37	0.35	0.24	0.12	0.13	0.17
10Itr	0.5ind	0.25	0.32	0.36	0.23	0.18	0.28
Imax	0.5ind	0.28	0.19	0.15	0.12	0.20	0.32
L3							
Itr	1.0	0.80	0.66	0.41	0.29	0.25	0.42
10Itr	1.0	0.82	0.67	0.46	0.20	0.24	0.42
Imax	1.0	0.68	0.49	0.30	0.15	0.29	0.55
Itr	0.5ind	0.33	0.33	0.23	0.22	0.22	0.46
10Itr	0.5ind	0.52	0.22	0.12	0.28	0.24	0.45
Imax	0.5ind	0.42	0.31	0.19	0.19	0.36	0.58

Anhang IX: Ausschnitt des Langzeitversuchs (6.25 Stunden)



8 Quellenverweis: Produktlinks

Name	Index	Link
ARF8200AA	[PL1]	https://www.adeunis.com/en/produit/analog-wired-sensor-interface/
ARF8230AA	[PL2]	https://www.adeunis.com/en/produit/pulse-impulse-interface/
ARF8240AA	[PL3]	https://www.adeunis.com/en/produit/modbus-interface-for-modbus-slaves/
ELT-1-LITE	[PL4]	https://www.elsys.se/shop/product/elt-1-lite/?v=f003c44deab6
RS485-LN	[PL5]	https://www.dragino.com/products/lora-lorawan-end-node/item/154-rs485-ln.html
IOT-L2S-D1	[PL6]	https://www.heyuanintel.com/products/LoRaWAN_Module.html
HYFW	[PL7]	https://www.heyuanintel.com/products/Modbus_Energy_Meter.html
SDM104E	[PL8]	http://www.smart-data-communication.com/sdc-products-services-3/sdc-smart-data-meter-e-2
EDL21	[PL9]	https://www.lobaro.com/portfolio/electricity-meter-edl21-over-lorawan-bridge/
SDM630-Lora	[PL10]	https://www.eastroneurope.com/products/view/sdm630mct-lorawan-mid
Semtech SX1276	[PL11]	https://www.semtech.com/products/wireless-rf/lora-core/sx1276
ESP32	[PL12]	https://www.espressif.com/en/products/socs/esp32
Janitza UMG96RM	[PL13]	https://www.janitza.de/umg-96rm.html
ONRO OR-WE-520	[PL14]	https://orno.pl/de/produkt/1083/3-phasen-stromzaehler-80a-mid-3-module-din-th-35mm
Kerlik iFemtoCell	[PL15]	https://www.kerlink.com/product/wirnet-ifemtocell/
Heltec Wireless Stick	[PL16]	https://heltec.org/project/wireless-stick/

Alle Links zuletzt geprüft am 11.10.2021

9 Quellenverweis: Genannte und verwendete Software

Name	Index	Link	Version	Autor/Hrsg.
MatLab Antenna Designer	[SL1]	https://de.mathworks.com/help/antenna/gs/antenna-design-and-analysis-using-antenna-designer-app.html	R2021b	The MathWorks Inc.
Node.js	[SL2]	https://nodejs.org/en/	14.18.1	nodeJS Fd.
Node Red	[SL3]	https://nodered.org/	2.0.6	nodeJS Fd.
Arduino IDE	[SL4]	https://www.arduino.cc/en/software	1.8	Arduino LLC
Arduino LoRa	[SL5]	https://github.com/sandeepmistry/arduino-LoRa	0.8.0	Sandeep Mistry
Esp32-encrypt	[SL6]	https://github.com/josephpal/esp32-Encrypt	--	Franz Joseph Pal
PubSubClient	[SL7]	https://github.com/knolleary/pubsubclient	2.8	Nick O`Leary
ESP32 HAL	[SL8]	https://dl.espressif.com/dl/package_esp32_index.json	1.0	Espressif Systems Co., Ltd.
RaspberryPi OS Lite	[SL9]	https://www.raspberrypi.com/software/	5.10	RaspberryPI Fd.
Mosquitto	[SL10]	https://mosquitto.org/	2.0.13	Eclipse Fd, Inc.
HostApd	[SL11]	https://man.openbsd.org/hostapd.8	2.2.7	(Open Source)
Dnsmasq	[SL12]	https://thekelleys.org.uk/dnsmasq/doc.html	2.80	(Open Source)
netfilter-persistent	[SL13]	https://www.netfilter.org/	1.0.11	(Open Source)
IpTables	[SL14]	https://www.netfilter.org/projects/iptables/downloads.html#iptables-1.8.7	1.8.4	(Open Source)
EnEffCo	[SL15]	https://www.eneffco.de/	--	ÖKOTEC Energiemanagement GmbH
influxDB	[SL16]	https://www.influxdata.com/	2.1.1	InfluxData Inc.
Microsoft Excel	[SL17]	https://www.microsoft.com/de-de/microsoft-365/excel	Office365	Microsoft Corporation
node-red-contrib-opcua	[SL18]	https://flows.nodered.org/node/node-red-contrib-opcua	0.2.244	Mika Karaila

node-red-contrib-ui-table	[SL19]	https://flows.nodered.org/node/node-red-node-ui-table	0.3.11	Nick O`Leary

Alle Links zuletzt geprüft am 10.11.2021

10 Literaturverzeichnis

Bundesnetzagentur Vfg 133 / 2019 (2019): Allgemeinzuteilung von Frequenzen zur Nutzung durch Funkanwendungen geringer Reichweite (SRD).

Degesys, Julius; Ian Rose; Ankit Patel; Radhika Nagpal (2007): Proceedings of the Sixth International Symposium on Information Processing in Sensor Networks ; April 25 - 27, 2007, Cambridge, Massachusetts, USA. DESYNC: Self-Organizing Desynchronization and TDMA on Wireless Sensor Networks, S. 11–20.

Espressif Systems Co., Ltd. (Hg.): Datenblatt ESP32. Online verfügbar unter https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf, zuletzt geprüft am 09.11.2021.

Galinina, Olga; Andreev, Sergey; Balandin, Sergey; Koucheryavy, Yevgeni (Hg.) (2020): Internet of Things, Smart Spaces, and Next Generation Networks and Systems. 20th International Conference NEW2AN 2020 and 13th Conference ruSMART 2020 St. Petersburg Russia August 26–28 2020 Proceedings Part I. 1st ed. 2020. Cham: Springer International Publishing; Imprint: Springer (Springer eBook Collection, 12525).

Heltec Automation (Hg.) (2021): Produktseite Heltec Wireless Stick. Online verfügbar unter https://heltec.org/project/wireless-stick/, zuletzt geprüft am 26.10.2021.

Hering, Ekbert; Bressler, Klaus; Gutekunst, Jürgen (2014): Elektronik für Ingenieure und Naturwissenschaftler. Berlin, Heidelberg: Springer Berlin Heidelberg.

Hering, Ekbert; Martin, Rolf; Stohrer, Martin (2012): Physik für Ingenieure. Berlin, Heidelberg: Springer Berlin Heidelberg.

John Anthes (2007): OOK, ASK and FSK Modulation in the Presence of an Interfering signal. Hg. v. Murata Manufacturing Co., Ltd. Carrollton, Texas, USA. Online verfügbar unter https://wireless.murata.com/media/products/apnotes/ook.pdf, zuletzt geprüft am 21.10.2021.

Kark, Klaus W. (2014): Antennen und Strahlungsfelder. Wiesbaden: Springer Fachmedien Wiesbaden.

Linnemann, Marcel; Sommer, Alexander; Leufkes, Ralf (2019): Einsatzpotentiale von LoRaWAN in der Energiewirtschaft. Praxisbuch zu Technik Anwendung und regulatorischen Randbedingungen. Wiesbaden: Springer Vieweg (Springer eBook Collection).

Lourdes Peñalver; Lorena Parra; Ozgur Akan; Paolo Bellavista; Jiannong Cao; Geoffrey Coulson et al. (Hg.) (2021): Industrial IoT Technologies and Applications: Springer Nature Switzerland.

Ma, Yao; Lim, Teng Loon; Pasupathy, S. (2002): Error probability for coherent and differential PSK over arbitrary Rician fading channels with multiple cochannel interferers. Hg. v. IEEE Xplore (3).

Magrin, Davide; Capuzzo, Martina; Zanella, Andrea (2019): A Thorough Study of LoRaWAN Performance Under Different Parameter Settings. Hg. v. IEEE (1).

Magrin, Davide; Capuzzo, Martina; Zanella, Andrea; Zorzi, Michele (2021): A Configurable Mathematical Model for Single-Gateway LoRaWAN Performance Analysis. 4. Aufl. Hg. v. IEEE.

Mirabella, O.; Brischetto, M.; Raucea, A.; Sindoni, P. (Hg.) (2018): Dynamic Continuous clock Synchronization for IEEE 802.15.4 based sensor networks. Lightweight synchronization algorithm with self-calibration for Industrial LoRa Sensor Networks. Unter Mitarbeit von Luca Tessaro, Cristiano Raffaldi, Maurizio Rossi, Davide Brunelli. Department of Industrial Engineering, University of Trento, Trento, Italy: IEEE.

MQTT-V3.1.1-os, 29.10.2014: MQTT v.3.1.1 Specification. Online verfügbar unter http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf, zuletzt geprüft am 21.11.2021.

myDevices Inc. (Hg.) (2021): Cayenne Documentation. Low Power Payload. Online verfügbar unter https://developers.mydevices.com/cayenne/docs/lora/#lora-cayenne-low-power-payload, zuletzt geprüft am 10.11.2021.

NIST Computer Security Division (CSD) (2001): FIPS 197, Advanced Encryption Standard (AES). Announcing the ADVANCED ENCRYPTION STANDARD. Hg. v. Federal Information Processing Standards (192). Online verfügbar unter https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf, zuletzt aktualisiert am 26.11.2001, zuletzt geprüft am 10.11.2021.

OpenJS Fd. (Hg.) (2021a): Node Red Configuration. OpenJS Fd. Online verfügbar unter https://nodered.org/docs/user-guide/runtime/configuration, zuletzt geprüft am 21.10.2021.

OpenJS Fd. (Hg.) (2021b): Securing Node Red. OpenJS Fd. Online verfügbar unter https://nodered.org/docs/user-guide/runtime/securing-node-red, zuletzt geprüft am 21.10.2021.

ORNO-Logistic Sp. Z O.O (2014): OR-WE-520 Zertifikat MID ModB. Gliwice, Polen (0120/SGS0396). Online verfügbar unter https://b2b.orno.pl/download-resource/26068/, zuletzt geprüft am 24.08.2021.

ORNO-Logistic Sp. Z O.O (Hg.) (2021): Bedienungsanleitung OR-WE-513,516,517,520. Online verfügbar unter https://b2b.orno.pl/download-resource/26057/, zuletzt aktualisiert am 30.01.2021, zuletzt geprüft am 10.11.2021.

Peebles, Peyton Zimmerman (1987): Digital communication systems. Englewood Cliffs, NJ, USA: Prentice/Hall.

Polonelli, Tommaso; Brunelli, Davide; Marzocchi, Achille; Benini, Luca (2019): Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment. In: *Sensors (Basel, Switzerland)* 19 (4). DOI: 10.3390/s19040838.

RAKwireless Technology Co. Ltd. (Hg.) (2021): Datenblatt RAK 868-930 MHz LoRa Antenna. Online verfügbar unter https://cdn-reichelt.de/documents/datenblatt/E910/RAK_W113_DB-EN.pdf, zuletzt geprüft am 20.11.2021.

RaspberryPi Fd. (Hg.): Setting up a Raspberry Pi as a routed wireless access point. Online verfügbar unter https://www.raspberrypi.org/documentation/configuration/wireless/access-point-routed.md, zuletzt geprüft am 10.06.2021.

Rembold, Bernhard (2017): Wellenausbreitung. Wiesbaden: Springer Fachmedien Wiesbaden.

Richtlinie 2014/32/EU, 26.02.2014: Richtlinie 2014/32/EU des Europäischen Parlaments und des Rates vom 26. Februar 2014 zur Harmonisierung der Rechtsvorschriften der Mitgliedstaaten über die Bereitstellung von Messgeräten auf dem Markt (Neufassung), zuletzt geprüft am 25.08.2021.

Schmüser, Peter (2012): Theoretische Physik für Studierende des Lehramts 2. Elektrodynamik und Spezielle Relativitätstheorie. Berlin: Springer (Springer-Lehrbuch, / Peter Schmüser ; Bd. 2).

Semtech Corp. (Hg.): LoRa and LoRaWAN: Documentation. Online verfügbar unter https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan, zuletzt geprüft am 03.10.2021.

Semtech Corp. (Hg.) (2015a): AN1200.22 LoRa™ Modulation Basics. 2. Aufl. (AN1200.22). Online verfügbar unter https://www.frugalprototype.com/wp-content/uploads/2016/08/an1200.22.pdf, zuletzt geprüft am 09.11.2021.

Semtech Corp. (Hg.) (2015b): Datenblatt SX1276. Online verfügbar unter https://www.semtech.com/products/wireless-rf/lora-core/sx1276#download-resources, zuletzt geprüft am 31.08.2021.

Semtech Corp. (Hg.) (2021): Documentation Memory Types. Online verfügbar unter https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/memory-types.html, zuletzt geprüft am 09.11.2021.

ECMA-404, 2017: The JSON Data Interchange Syntax.

The Things Industries (Hg.) (2021): Documentation The Things LoRaWAN Network. Limitations. Online verfügbar unter https://www.thethingsnetwork.org/docs/lorawan/limitations/, zuletzt geprüft am 09.11.2021.

Weyand, Astrid; Rommel, Carmen; Zeulner, Julia; Sossenheimer, Johannes; Weigold, Matthias; Abele, Eberhard (2021): Method to increase resource efficiency in production with the use of MFCA. In: *Procedia CIRP* 28 (98), S. 264–269. DOI: 10.1016/j.procir.2021.01.101.