

Seminararbeit: Lorawan

Tobias Sigmann

25. Mai 2019

Inhaltsverzeichnis

1	Einführung in Lora	4
2	Aufbau eines Lora-Netzwerk	4
2.1	Gateway	5
2.2	Netzwerkserver	6
2.3	Join-Server	6
2.4	End-Gerät	7
3	LoraWan Funktionsweise	7
3.1	Schichtenmodell	7
3.2	Netzwerkbeitritt	8
3.2.1	OTAA	8
3.2.2	ABP	10
3.3	Protokoll	10
3.3.1	MAC-Kommandos	11
3.3.2	LoRa-Paketstruktur	12
3.4	Übertragungsart	14
3.4.1	Adaptive Data Rate	15
4	Lora Geräte Klassen	16
4.1	Klasse A	16
4.2	Klasse B	17
4.2.1	Klassenwechsel A nach B	18
4.2.2	Betrieb	18
4.2.3	Singel / Multicast	19
4.2.4	Beacon	20
4.3	Klasse C	21
5	Sicherheit	21
5.0.1	Schlüssel	22
5.0.2	Zähler	23

6	Live-Beispiel	23
7	Ausblick	23

1 Einführung in Lora

Lora ist ein Low Power, Wide Area (LPWA) Netzwerkprotokoll und somit sehr gut für batteriebetriebene kabellose Geräte geeignet. Deswegen wird Lora auch oft im Internet of Things (IoT) Bereich verwendet. Mittels der bidirektionalen Kommunikation ist es möglich Daten und Befehle über weite Strecken zu übertragen. Leider leidet darunter die Geschwindigkeit, sodass sich Lora nicht als WLAN Ersatz eignet. Trotzdem können zwischen 0.3 und 50 kbps erreicht werden. In Europa werden 863 MHz bis 870 MHz verwendet. Allerdings variiert der Frequenzbereich für andere Kontinente. Je nach Bedingungen können so bis zu 20km entfernte Endgeräte erkannt und mit diesen kommuniziert werden. Es ist sogar möglich den Standort des Gerätes zu bestimmen.

Eine Alternative zu Lora ist Sigfox, hierauf werde ich nicht weiter eingehen. LoRa-WAN 1.1

[Tec15](Optimiert für Batterie Kapazität(Teilnehmer) Reichweite, Kosten mehrjährige Batterielaufzeit, kleine Datenmengen, große Reichweite, LPWAN (Low Power WAN)

Kriterien für Lora: Netzwerk Architektur, Reichweite, Batterielaufzeit, Interferenzrobustheit, Anzahl Knoten, Sicherheit, bidirektionale Kommunikation, verschiedene Anwendungsunterstützung

Orientiert für Mobile Adressierbare Endgeräte)

[AVTP⁺17](alternativen: Sigfox, Ingenu, Dash7

Klassen Kompromiss zwischen Reichweite, Performance(Latenz/ Durchsatz) und Energiebedarf

Energiesparend durch ADR (Adaptive Daten Rate))

Es wird folgen: Was ist lora, wo und wofür wird es benutzt, wie weit kann man senden und wie schnell...

2 Aufbau eines Lora-Netzwerk

Lora wird auch deswegen gerne für IoT-Geräte verwendet, weil der Netzwerkaufbau ermöglicht die über Lora verwendeten Daten im Internet abzurufen und so ohne weiteres

das Gerät mit dem Internet zu verbinden. Um die von den End-Geräten gesendeten LoRa-Pakete auf IP/TCP Pakete umzusetzen wird ein Gateway benötigt, das auf der einen Seite LoRa-Pakete empfängt/sendet und auf der anderen Seite TCP/IP Pakete verwendet. Das Gateway implementiert aber keinerlei Logik. Hierzu ist ein Netzwerkservers zuständig der durch die Gateways das Netzwerk kontrolliert und steuert. Gleichzeitig stellt er die Verbindung zu einem Applikationsserver her, in dem er die vom Gateway empfangenen Daten weiterleitet.

Der Applikationsserver ist zuständig den die gesendete Nachrichten zu verarbeiten und gegebenenfalls selbst welche an die Endgeräte zu senden.

Diese Architektur wurde gewählt um die Laufzeit der Akku betriebenen Endgeräte, Anzahl der Endgeräte, Qualität Signals und Sicherheit des Netzwerkes möglichst hoch zu halten. [Tec15, S. 8 ff.]

2.1 Gateway

Das Teilnetz das aus dem Gateway und mehreren LoRa-Endgeräten besteht ist Sternförmig aufbau. Jedes Endgeräten kommuniziert direkt mit dem Gateway. Diese Art der Kommunikation wird auch “Single-Hop-Connection” zu Deutsch (Einfacher-Sprung-Verbindung) genannt, da die gesendeten Daten ohne Umwege an das Gateway gesendet werden. Jedes Gateway ist mit mindestens einem Netzwerkservers verbunden.

Ein Endgerät kann gleichzeitig an mehreren Gateways senden. Der Netzwerkservers ist zuständig die Pakete auf Duplikate zu überprüfen und nur einmalig an die Applikationsservers zu senden. Ein weiterer Vorteil ist das kein Übergabe der Endgeräte bei Standortwechsel zu andern Gateways nötig ist. Dadurch müssen die Gateways mit vielen Endgeräten kommuniziert. Um diese hohe Endgeräteanzahl zu ermöglichen wurde darauf verzichtet mit jedem Endgerät einzelne zu kommunizieren und stattdessen auf eine Parallele Kommunikation gesetzt. Hierzu werden adaptive Datenraten und Mehrkanal-Multi-Modem-Transceiver verwendet.

Durch die genannt Eigenschaften der Gateways wird eine gute Skalierbarkeit erzielt. Dadurch können neue Gateways die Anzahl der Endgeräte um das 6 bis 8-fach erhöhen.vgl. [Tec15, S.10]

2.2 Netzwerkserver

Der NetzwerkServer ist das “Herzstück“ eines jeden Lora-Netzwerkes. Er kann mit mehreren Gateways und mehreren Applikationsserver verbunden sein.

Die wichtigste Aufgabe des Netzwerksserver ist das Steuern des LoRa-Teils des Netzwerkes. Der Server verwaltet jedes Endgerät separat indem es mit ihm den zu verwenden Funkkanal Aushandelt und die Datenrate kontrolliert wenn ADR(Adaptiv Data Rate) verwendet wird. Außerdem ist er bei dem Netzwerkbeitritt eines Endgerätes .beteiligt.

Weiterhin überprüft er die empfangen Pakete auf ihre Korrektheit, Integrität und filtert Duplikate, die durch das Empfangen der gleichen Übertragung von einem Endgerät an verschiedenen Gateways, verursacht wurden. Dabei ermittelt er auch die Gateways, die den besten empfang zu den jeweiligen Endgeräten hat und nutzt dieses um Daten an die Endgeräte zu senden.

Es ist nicht immer möglich Daten direkt zu senden, da die Endgeräte nur manchmal empfangsbereit sind. Um die Applikationsserver zu entlasten, puffert der Netzwerkserver die Daten und sendet diese zum nächst möglichem Zeitpunkt.

Eine weitere sehr Wichtige Ausgabe ist es eine API für den Applikationsserver bereitzustellen um eine einfache und schnelle Kommunikation zu ermöglichen.

2.3 Join-Server

Der Server kann mit mehreren Netzwerkservern verbunden werden und jeder Netzwerkserver kann mehrere Join-Server haben.

Ein Join-Server wird benötigt um den Beitritt mittels OTAA zu ermöglichen. Mehr zu OTAA kann in dem Kappitel OTAA gelesen werden. Wenn ein Endgerät dem Netzwerk beitreten möchte, leitend der Netzwerkserver die Anfragen an den Join-Server weiter. Dieser führt dann die nötige schritte des Beitritts aus wie z.B. ableiten von Schlüsseln oder Senden der nötigen Einstellungen. Um dies zu tun muss ihm der NwkKey und der AppKey bekannt sein, da diese zum verschlüsseln der Nachrichten verwendet werden aber aus Sicherheitsgründen nie über das Netzt übertragen werden dürfen. [SOR17b, S. 9 f.]

2.4 End-Gerät

Endgeräte sind Geräte die Informationen mittels LoRa empfangen oder senden. Jedes Endgerät ist mit einem bestimmten Applikationsserver verbunden.

Jedes Endgerät muss zur korrekten Funktion mehrere wichtige Informationen speichern.

- DevEUI: Globale Endgeräte_ID die eindeutig für jedes Endgerät definiert ist. Vergleichbar mit der MAC-Adresse eines TCP/IP Gerätes.
- JoinEUI: Globale Adresse des Join-Servers an den die Anfrage gehen soll. Wird nur für OTAA Geräte benötigt.
- NwkKey und AppKey: Werden verwendet um spätere Schlüssel abzuleiten und die Kommunikation während der Beitrittsprozedur in ein Netzwerk abzusichern. Dafür müssen sie sowohl dem Join-Server als auch dem Endgerät bekannt sein da sie nie übertragen werden.

[SOR17c, S.47 ff.]

3 LoraWan Funktionsweise

Im folgenden Kapitel wird näher auf die Funktionsweise von LoRaWAN eingegangen. Speziell, liegt der Fokus auf dem Netzwerkeitritt, das verwendete Protokoll und wie die Daten physikalisch Übertragen werden.

3.1 Schichtenmodell

Das Schichtenmodell lässt sich in zwei Teile unterteilen. Der LoRa Teil ist der unterste und kümmert sich um die physikalische Übertragung der Pakete und der LoRaWAN Teil des Modells ist für die Steuerung des Netzwerkes, Implementierung der LoRaWAN-Klassen und das Überprüfen / Verschlüsseln der Daten zuständig.

Die unterste Schicht des LoRa Teils ist für die Anwendung der Richtigen Frequenzen zuständig. In Europa muss das ISM-Band 868 verwendet werden in den Vereinigten Staaten wird das Band 915 verwendet. [Tec15, S.7]

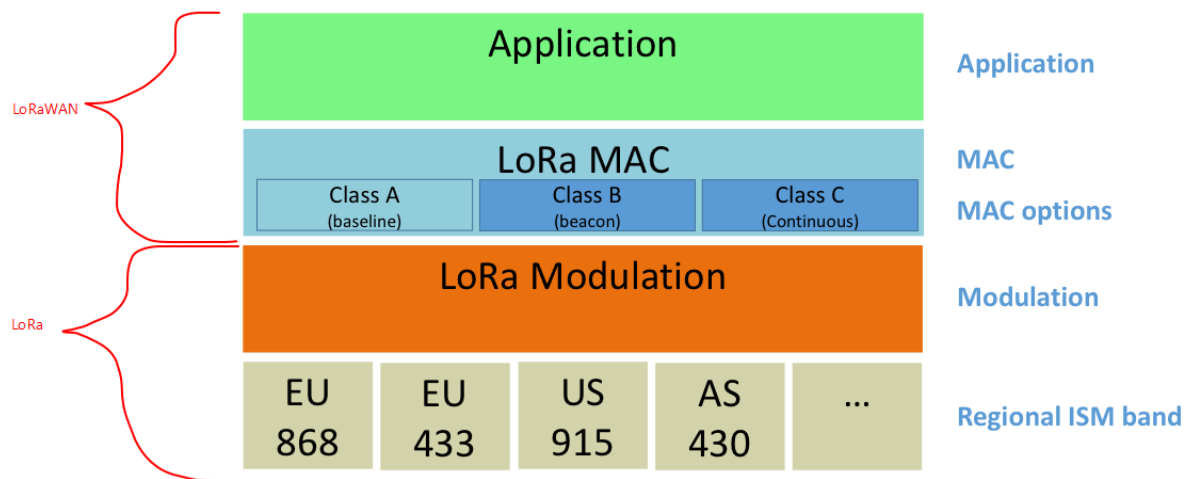


Abbildung 1: LoRaStack [Tec15, S.7]

Die darüber liegende Schicht heißt LoRa Modulation und kümmert sich darum dass die Pakete so in die Frequenz "moduliert" werden, dass der Empfänger diese korrekt und effizient empfangen und wiederherstellen kann. Mehr dazu im Kapitel Übertragungsart

Über der LoRa Modulation Schicht liegt die erste LoRaWan Schicht, LoRa MAC. AC steht für "Media Access Protokoll". Dieses Protokoll wird verwendet um das Loranetz zu steuern und wie der Name schon sagt, Daten zu übertragen. Diese Schicht ist außerdem für die Implementierung der einzelnen Endgeräteklassen und für das Übertragen der Steuerungskommandos zuständig. Mehr zu den Klassen kann im Kapitel `namerefsec:klassen` und im Kapitel Protokoll gelesen werden.

Die oberste Schicht nennt sich Applikationsschicht und ist dafür zuständig die Nutzdaten einer Nachricht passend zu verpacken, zu verschlüsseln und zu authentifizieren.

3.2 Netzwerkbeitritt

End-Geräte sind immer bestimmten Netzwerken zugeordnet. Es gibt zwei Wege um ein neue End-geräte zu einem bestehenden Netzwerk hinzuzufügen.

3.2.1 OTAA

Die sicherste aber auch aufwendigste Methode um ein End-Gerät mit einem Netzwerk zu verbinden heißt OTAA "Over-the-Air Activation". Hierbei muss jedes Mal wenn einem Netzwerk beigetreten werden soll die Join-Prozedur ausgeführt werden. Hierfür müssen

folgende 4 Konstanten Vorgegeben werden. DevEUI, JionEUI , NwkKey, AppKey.

Näheres zu dem DevEUI und JoinEUI kann im Kapitel ?? gefunden werden.

Der NWKSKY ist für die Verschlüsselung der Datenpakete bis zu Gateway zuständig. Auch dieser Key wird vom Netzwerkserver erzeugt und muss manuell in den Code eingetragen werden. [GAS17, S.3]

Der letzte Wert heißt APPSKY und sichert die Kommunikation vom End-Gerät zu dem Applikationsserver ab. Der Schlüssel wird genau wie der NWKSKY vom Netzwerkserver erzeugt und verwaltet. [GAS17, S.3]

Als erstes muss das End-Gerät eine Join- oder Rejoin-Nachricht senden. Die Nachricht besteht aus der JoinEUI, dem DevEUI und einer DevNonce. Mit der DevNonce sollen Replayattack verhindert werden. Diese Nonce ist das beim ersten Join-Request 0 und sollte sich bei jedem Join-Request erhöhen. Unter anderem deswegen muss sie auch dann noch gespeichert werden wenn kein Strom zur Verfügung steht um die Nonce nicht nach jedem ausschalten zurückzusetzen. Falls von dem gleichen Endgerät eine Join-Request mit einer zu kleinen DevNonce gesendet wird, wird die Nachricht ignoriert und es ist nicht möglich dem Netzwerk beizutreten.

Die Accept Nachricht besteht aus einer JoinNonce, einem NetzwerkID Net_ID, einer Geräteadresse DevAddr, einem Einstellungsfeld DLSettings , einer Zeitangabe wie lange zukünftig auf eine Antwort nach dem senden gewartet werden muss, hier RxDelay und einer optionalen Liste an Netzwerkparameter CFList.

Die JoinNonce wird verwendet um Replayataken zu verhindern und muss größer sein als die zuletzt gesendete um von dem Endgerät verarbeitend zu werden. Außerdem wird die Nonce benutzt um Schlüssel wie AppSKY herzuleiten. Für jedes Endgerät wird eine eigene Join Nonce geführt, sie sollte sich nicht wiederholen. Jedes Endgerät merkt sich die letzte JoinNonce und tritt auch nur bei wenn diese größer ist als die letzte empfangene.

Die Join-Accept Nachricht wird vom Endgerät nach JOIN_ACCEPT_DELAY1 oder JOIN_ACCEPT_DELAY2 nach dem Senden des Request erwartet. Sollte die Join-Accept Nachricht zu einem andern Zeitpunkt gesendet werden, wird diese nicht empfangen, da das Endgerät nicht empfangsbereit ist.

Mehr Informationen zu den Ableitungen der Schlüssel finden Sie in dem Kapitel Sicherheit.

3.2.2 ABP

Die einfachste Art des Beitritts heißt ABP was für “Activation by Personalization” zu Deutsch “Aktivierung durch Personalisierung” steht. Hierbei muss lediglich vor Inbetriebnahme des End-Gerätes 5 Konstanten definiert Werden.

Als erstes muss die DevAdr(Geräteadresse) angegeben werden. Diese Adresse existiert nur einmal im Netzwerk und wird verwendet um das Endgerät zu identifizieren. Die Adresse wird vom Netzwerkservers erzeugt und muss Manuel von dort in den Programmcode eingetragen werden,

Die anderen 4 Werte sind die Schlüssel die verwendet werden um die Kommunikation zu verschlüsseln. FNwkSIntKey, SNwkSIntKey, NwkSEncKey und AppSKey. Dies hat den Vorteil das die Join-Prozdeur übersprungen werden kann. Allerdings werden immer dieselben schlüssel verwendet was zu einem Sicherheitsproblem werden kann.

Nach Beitritt muss das ResetInd Mac Kommando im FOpt Feld gesendet werden gesendet. Dieses Kommando teilt dem Netzwerkservers mit, dass das Endgerät online ist und das die Standard Übertragungsart, Frequenz und Chanel benutzt werden.

Das Netzwerk muss mit einem ResetConf-Kommando antworten. In diesem Teilt es dem Endgerät die unterstützten LoRa-Versionen mit. Danach kann die Kommunikation beginnen.

3.3 Protokoll

Das LoRaWAN Protokoll ist optimiert für Batteriebetrieoben Endgeräte die drahtlos kommunizieren möchten. Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte. Die Modulationstechnik und eine Adaptive Dtanenrate (ADR). Auch die Öne-Hopärchitektur trät zur energieeffizienz bei. Die Art wie LoRa siganle Moduliert wird in Kapittel Übertragungsart besprochen. [CB⁺17, S,1 f]

Damit der Netzwerkservers das LoRa-Netzt steuern kann wurden, wurden Mac-Kommandos eingesetzt. Mit diesen Kommandos lassen sich wie schon gesehen, dem Netzwerk beitreten, mit dem Endgerät kommunizieren und Frequenzen, Kanäle und vieles mehr zu steuern. Da die Kommandos nur für den Netzwerkservers und die Endgeräte von Bedeutung sind, werden diese nicht an den Applikationsservers gesendet sonder vom Netzwerkservers her-

CID	Kommandoname	Funktion
21	22	23
31	32	33

ausgefiltert. Im Folgenden wird näher auf die MAC-Kommandos und die Paketstruktur eingegangen. Ein Uplink ist ein Paket dass vom Endgerät, das Bildlich gesprochen “unter“ dem Gateway sitzt, an das Gateway gesendet wird. Ein Downlink ist dem entspricht ein Paket das vom Gateway an den Server gesendet wird.

3.3.1 MAC-Kommandos

MAC-Kommandos werden ausschließlich für die Steuerung und Pflege des Netzwerkes verwendet. steht für “ Medium Access Control“. Da der Netzwerkeserver das Netzwerk steuert werden diese Kommandos nur zwischen dem Netzwerkeserver und dem Endgerät verwendet. Der Applikationsserver bekommt von den Kommando nichts mit.

MAC-Kommandos werden mittels eines “command identifier“ (CID) gesendet. Das ist eine 8 Bit Zahl die das MAC-Kommando repräsentiert. Nach dem CID folgen mögliche variablen die dem Kommando mitgegeben werden. Die Anzahl und Länge der Variablen ist nicht beschränkt.

Die MAC-Kommandos müssen zwingend ausgeführt werden. Damit der Netzwerkeserver weiß welche Kommandos ausgeführt wurden müssen die Nachrichten in derselbe Reihenfolge wie sie angekommen sind bestätigt oder beantwortet werden. Alle Kommandos die in einem Frame gesendet wurden, müssen in einem bestätigt / beantwortet werden. Um die Reihenfolge zu speichern existiert ein Puffer in den die MAC-Kommando-Antworten eingetragen werden. Dieser Puffer wird beim nächsten Uplink mit gesendet. Falls es vorkommt, dass der Puffer zu klein ist werden keine weiteren Antworten in den Puffer eingetragen. Deswegen muss der Netzwerkeserver vorher die Maximale gröÙer der Antwort berechne und die MAC-Kommandos dementsprechend in Frames aufteilen. [SOR17c, S.29 ff.]

Im Folgenden ist eine Liste aller MAC-Kommandos angegeben.

3.3.2 LoRa-Paketstruktur

Die Paketstruktur kommt wie beim ISO/OSI Schichtenmodell durch das “durchlaufen“ des Stacks zustande. Da im Folgenden die Paketstruktur vom Groben ins Feine Betrachtend wird, werden hier als erstes die Felder der Modulationsschicht betrachtet.

Jedes Pakete besteht aus grundlegend aus 2 Felder Präambel und PHYPayload Falls es sich um einen Uplink-Paket handelt wird noch ein CRC Code hinzugefügt: Preamble, PHYPayload, CRC. In diesem Fall spricht man von einem implizit Paket oder von dem implizitem Modus. Implizit Modus bedeute dass es kein Payload Header gibt, der Felderlängen oder CRC längenangebe angibt. Diese sind somit feste zuvor definierte. Im expliziten Modus werden noch 2 Felder hinzugefügt, PHDR und PHDR_CRC. Somit sieht ein expliziteste Paket folgendermaßen aus: Preamble, PHDR, PHDR_CRC, PHYPayload. Auch hier gilt, im Falle eines Uplink-Paketes wird am Ende ein CRC Feld angefügt. Somit ergibt sich folgende Paketstruktur: Preamble, PHDR, PHDR_CRC, PHYPayload, CRC.

Die Preamble ist dafür gedacht dem Empfänger mitzuteilen dass gleich Datengesendet werden. Deswegen wird hier nur ein Signal gesendet das ohne Informationen ist, aber von dem Empfänger wahrgenommen werden kann.

Da Teile des LoRaWAN Protokolls geschützt sind, finden sich über die PHDR und PHDR_CRC Felder kaum Informationen. Allerdings geht hervor, dass der PHDR die Länge des PHYPayloads und die Zieladresse beinhalten sollte. Das PHDR_CRC Feld wird benutzt um sicherzustellen dass die empfangenen Werte korrekt sind. Dies wird mittels des CRC Verfahrens überprüft.

Wie schon mehrfach erwähnt wird in Uplink-Nachrichten ein zusätzliches CRC Feld verwendet. CRC steht für Cyclisch Redundanz Check und wird verwendet um die Korrektheit der Nachricht zu bestätigen. PHDR, PHDR_CRC und das CRC Feld werden automatisch vom dem Funktransceiver (Modul aus Empfänger und Sender) hinzugefügt.

Die LoRa MAC ebene fügt nun das PHYPayload Feld ein. PHYPayload steht für Physikalische Payload. Es gibt 3 Mögliche PHYPayloads Entweder wird ein MACPayload eingefügt, Join-Rejoin-Request oder aber es werden die Join-Accept Nachricht darin transportiert. Um die Daten bzw. die MAC Kommandos richtig auswerten zu können und um die Korrektheit überprüfen zu können werden einige Headers und zusätzliche Felder benötigt. Deswegen lässt sich das Feld PHYPayload weiter unterteilen in MHDR und

MACPayload. Für den Fall das der MACPayload eine Join-rejoin oder MACPayload Nachricht ist, wir noch ein MIC Feld hinzugefügt(MHDR, MACPayload, MIC). MIC steht für Message Integrity Code und wird verwendet um die Korrektheit der des MACPayloads und des MHDR festzustellen.

Das MHDR Feld beschreibt wie die Daten im MACPayload Feld zu deuten sind. Wieder wird dieses Feld in Unterfelder Unterteilt. MType, RFU und Major heißen die Unterfelder. Das MType Feld beschreibt die Art der Nachricht. z.B: kann hier angegeben werden ob es sich um Datennachrichten, Join-Nachrichten, ... handelt. RFU steht für "Reserved for Future Usag" zu Deutsch "für zukünftige verwendung reservier".Daher kann dieses Feld in der version 1.1 und niedriger ignoriert werden. Das Major Unterfeld wird verwendet um das LoRa-Version der Nachricht zu definieren. Momentan ist nur der Wert 0 Definiert. 0 Steht für LoRaWan R1. Die restlichen werde sind für zukünftige Updates reserviert.

Mit der Unterteilung des MACPayload springen wir in dem LoRaStack noch eine ebene höher, in die Applikationsschicht. Enthalten im MACPayload Feld sind der Frameheader FHDR, der Frame Port FPort und der Frame Payload FRMPayload. Daten die gesendet werden sollen befinden sich in dem FRMPayload Feld. Wenn keine Daten gesendet werden, kann das FRMPayload Feld auch MAC-Kommandos enthalten. In dem Feld FPorts wird angegeben an welchen Port und somit an welche Teilapplikation des Applikationsserver die Daten geleitet werden sollen. Es gibt einige feste Ports. Port 0 ist reserviert um MAC-Kommandos im FRMPayload Feld entgegenzunehmen. Die Ports 0x01 bis 0xDF sind Anwendungsspezifische Ports und Port 0xE0 ist für das LoRaWAN Test Layer Protokoll reserviert. Falls ein anderer Port als die geraden genannten angegeben wird, wird die Nachricht verworfen.

Erneut kann der FHDR "Frame Header" in einzelne Felder unterteilt werden in DevAddr, FCtrl, FCnt, FOpts. In dem Feld DevAddr wird die Zieladresse der Nachricht vermerkt. Im Feld FCnt (Frame Counter) wird der jeweilige counterwert für die bisher gezählten Nachrichten übermittelt. Hamit schützt man sich vor Replayattacks. Mehr zu den Counter kann im Kapitel Sicherheit gelesen werden. Im FOpt Feld können bis zu 5 MAC Kommandos parallel zu Daten übermittelt werden. Die Anzahl kommt auf die Menge der mitgelieferten Variablen an.

Das Letzte Feld das in Unterfelder unterteilt werden kann ist das FCtrl Feld. Hier wird das Verhalten des Gerätes gesteuert sowie Nachrichten bestätigt. Es gibt leichte Unterschiede für ein Uplink und für Downlink Nachrichten. Beide Nachrichtentypen haben ein ADR, ein ACK und ein FOptsLen Feld. Im ADR wird definiert ob der Sendende bereit ist im Modus "Adaptive Data Rate" Daten zu senden, siehe Kapitel Adaptive Data Rate. Mit dem ACK Feld können empfangene Nachrichten bestätigt werden. Ob Nachrichten bestätigt werden müssen steht im MType Feld (Confirmed Data). In dem FOptsLen Feld wird die Länge des FOpts Feldes mitsamt des Headers eingetragen. Wenn das FOptsLen 0 ist, ist kein FOpts Feld vorhanden.

Ein Downlinkpaket hat zusätzlich ein RFU Feld das nicht verwendet wird und ein FPending Feld. In diesem Feld kann das Gateway bzw. der Netzwerkservers dem Endgerät mitteilen, dass noch mehr Daten zu senden sind.

Dahingegen hat ein Uplinkpaket ein ClassB Feld indem das Endgerät dem Gateway mitteilt, dass es gerne auf Funktionsklasse B wechseln würde und ein ADRACKReq Feld. Dieses Feld wird verwendet um zu überprüfen ob das Netzwerk noch antwortet. Die genaue Funktionsweise ist im Kapitel Adaptive Data Rate erklärt.

Eine noch genauere Darlegung der LoRa-Paketstruktur kann in den LoRaWA 1.1 Specification [SOR17c] gefunden werden.

3.4 Übertragungsart

Um die entstandenen Pakete in Signale umzusetzen und diese effizient und gleichzeitig übertragen zu können nutzt LoRa Chirp-Spread-Spectrum (CSS). Hierbei werden die Frequenz über eine gewisse Zeit hinweg verändert. Durch messen in welche richtung, ansteigen oder abfallen, die Frequenz verändert wird, können 1 und 0 codiert werden. Man spricht bei einem Bit von einem Chirp-Impuls. Durch aneinanderreihen der verschiedenen Impulse ist es möglich mehrere Bits nacheinander zu übertragen. Das entstandene Signal wird auch als Sub-Chirp bezeichnet. Durch verwenden von unterschiedlichen ansteigezeiten und abfallzeiten ist es möglich mehrere Signale auf der selben Frequenz zu übertragen ohne dass die Signale sich gegenseitig stören. Dies nennt man auch Spreading Factor. Außerdem kann die Parallelität durch verschiedene Frequenzbereiche verbessert werden. CSS ist besonders für große Reichweiten geeignet und somit auch bestens für LoRa. Am besten ist

das Signal wenn das Endgerät nahe am Gateway ist. Je weiter es entfernt desto schlechter wird das Signal. Um die Kommunikation trotzdem zu ermöglichen wird der “Spreading Factor“ erhöht. Dies hat auch den Vorteil dass der Energieaufwand gering gehalten werden kann. Analog wie Menschen auf einer Party nicht immer versuchen lauter sondern besonders langsam und deutlich sprechen. [SOR17a]

Diese Aufteilung durch den Spreading Factor und die Frequenz werden Channels erzeugt. Channels können beliebig benutzt werden. es gibt allerdings müssen zwei Regeln zu beachtet werden:

1. Channels werden per Pseudozufallszahl geändert
2. Sendezeit erfüllt die Regionalen Bestimmungen

Das Aloha Protokoll wird verwendet um festzustellen wann gesendet werden soll. Dabei wird einfach gesendet wenn Daten zum Senden vorhanden sind. Wenn nun zwei Sender gleichzeitig auf dem selben Channel senden möchten kommt es zu einer Kollision. Dadurch kann das Gateway die empfangenen Daten nicht mehr auswerten und die Daten müssen erneut übertragen werden. Deswegen warten beide Endgeräte eine zufällige, unterschiedliche Zeit ab bevor sie erneut senden.

3.4.1 Adaptive Data Rate

Adaptive Data Rate oder kurz ADR wird verwendet um immer die optimalste Senderate und die optimale Sendepower für das Endgerät zu finden und so schnellstmöglich die Daten zu senden. ADR kann nur verwendet werden wenn im FHDR Feld des LoRa-Paketes das ADR Bit gesetzt ist, siehe Protokoll. Die Steuerung durch ADR findet durch den Netzwerkservers statt. Sobald der Netzwerkservers bereit ist, setzt er das Bit im Downlink-Paket. Ist das Endgerät ebenfalls bereit setzt es ebenfalls das Bit und ADR kann verwendet werden. Falls es nicht möglich sein sollte ADR zu verwenden sollte es durch das Applikationslayer gesteuert werden.

Die Steuerung findet durch spezielle MAC-Kommandos statt. Standardgemäß wird die höchste Übertragungsstärke verwendet und die geringste Übertragungsrate. Falls diese gedrosselt werden soll wird vom Netzwerkservers das LinkADRReq MAC –Kommando

benutzt. Mit diesem wird das Endgerät informiert, dass es die Übertragungsstärke, Übertragungsrate oder den Übertragungskanal ändern soll. Die Werte sind in den Parameter codiert. Sobald die Werte geändert wurden, muss periodisch überprüft werden ob das Netzwerk die Nachrichten noch bekommt. Deswegen wird jedes Mal wenn der wenn ein Uplink empfangen wird, wird der `ADR_ACK_CNT` Zähler erhöht. Wenn dieser Zähler ein gewissen schwellenwert (`ADR_ACK_Limit`) überschreitet, wird das `ADRACK-Req` Bit im Uplink gesetzt. Dieses signalisiert den Netzwerkserver das er mit einem ein Nachricht senden muss um die Verbindung zu bestätigen. Falls dieser Downlink nicht in `ADR_ACK_Delay` Frames empfangen wird, wird zuerst die Übertragungsstärke auf das Maximum gesetzt. Falls möglich wird außerdem die Datenrate verringert um die Reichweite zu erhöhen. Die Datenrate wird solange weiter, jede `ADR_ACK_Delay` Frames, verringert bis diese minimal ist. Falls diese schon minimal ist müssen alle Kanäle benutzt werden. Dies wird solange probiert biss eine Verbindung hergestellt werden kann. [SOR17c, S.19 f]

4 Lora Geräte Klassen

Um maximal Energie zu sparen aber trotzdem die Möglichkeit dass die Endgeräte agile Daten empfangen können wurden die Geräteklassen eingeführt. Das Hauptmerkmal der Klassen sind die unterschiedlichen Empfangsmodien. Es gibt 3 Klassen, A, B und C. Die Klasse A muss standartgemäß von jedem Endgerät implementiert werden. B und C sind Optional und müssen nicht vorhanden sein Alle Geräte die mehr als Klasse A unterstützen werden als “higher Class end-devices “ genannt. [SOR17c, S.10]

4.1 Klasse A

Klasse A wird auch (All end-Devicec) genannt, zeichnet sich durch den geringsten Stromverbrauch aus. Die Kommunikation wird von dem Endgeräte gestartet werden. Das bietet die Möglichkeit dass das Endgerät, wenn keine Daten gesendet werden müssen, in einen sehr sparsamen Schlafmodus wechselt. Um das Endgeräte nicht zum “aufwachen“ zwingen zu müssen, wurde auf einen Hardbeat oder ähnliches verzichtet. Dadurch kann das Endgerät so lange “schlafen“ wie es möchte. Klasse A erlaubt außerdem das das Endgerät andere

Protokolle verwendet solange es keine LoRa Daten sendet oder empfängt. [SOR17c, S.11 ff.] Das Endgerät startet die Kommunikation in dem es Daten an das Gateway sendet. Daraufhin hat das Gateway die Möglichkeit 2 mal Daten zum Endgeräte senden. Die Empfangsfenster werden RX1 und RX2 genannt.

Die Empfangsfenster RX1 und RX2 müssen mindestens solange geöffnet bleiben das sie eine beginnende Übertragung feststellen können. Falls keine Übertragung empfangen wird, wird das Empfangsfenster wieder geschlossen. Empfangsfenster RX1 wird nach RECIEV_DELAY1 zeiteinheiten \pm 20msec nach Beendigung des Uplinks geöffnet. Es wird die selbe Frequenz und Datenrate verwendet die auch bei den Downlink verwendet wurde. Wenn festgestellt in RX1 festgestellt wurde das keine Weiteren Daten mehr empfangen werden müssen kann auf das öffnen des RX2 Fensters auch verzichtet werden.

RX2 wird nach RECIEV_DELAY2 Zeiteinheiten \pm 20msec nach Beendigung des Uplinks geöffnet. Allerdings ist die Datenrate und Frequenz fest. Nur mittels spezieller MAC Kommandos kann dies verändert werden.

Für die Join-Prozedur wird immer die Klasse A verwendet. Die verwendete Frequenz entspricht der Empfangsfenster RX1 oder RX2.

4.2 Klasse B

Die Klasse B (B für BEACON) bietet bidirektionale Kommunikation mit einer deterministischem Downlink Latenz. Um diese Latenz zu gewährleisten, muss die Kommunikation synchronisiert ablaufen. Außerdem muss festgestellt werden, ob das Endgerät bzw. das Gateway noch in Reichweite ist. Dies wird mittels eines periodischem Beacon ermittelt. Dieser Beacon wird regelmäßig vom Gateway gesendet und dient der Synchronisation der Endgeräte. Zeitpunkten gesendet werde realisiert. Die Latenz ist einstellbar und kann bis zu 128 Sekunden. [SOR17c, S.66 ff.]

Die Endgeräte öffnen in regelmäßigen Abständen ein Empfangsfenster das Pingslot genannt wird. Ein Downlink der in einem Pingslot gesendet wird, wird Ping genannt. Da immer das Gateway mit dem besten empfang die Daten an das Gateway sendet, muss das Endgerät selbständig feststellen wenn es einen Beacon mit einer unbekannten ID bekommt und durch eine Uplink dem Server mitteilen das es sich in einer neuen Umgebung ist. Dadurch lernt der Server wo sich das Endgerät befindet und kann das

Gateway mit dem besten Empfang wählen. Obwohl das Endgerät durch die periodischen Beacon nicht “schalfe“ kann, ist die Klasse B für den Batteriebetrieb gedacht.

4.2.1 Klassenwechsel A nach B

Um einen Wechsel überhaupt zu ermöglichen muss der Netzwerksender die Standard Pingslot Periode, die Pingslot Datenrate und den Pingslot Kanal kennen. Alle Endgeräte treten in Klasse A dem Netzwerk bei. Das wechseln in die Klasse B wird durch folgenden Prozess realisiert.

Als erstes muss das Programm des Endgerätes beim LoRaWAN Layer anfragen ob es möglich ist in Klasse B zu wechseln. Der Layer sucht nun nach einem Beacon. Wird ein Beacon entdeckt, wird die BEACON_LOCKED Serviceprimitive zurückgeliefert. Wenn kein Beacon empfangen wurde wird die BEACON_NOT_FOUND primitive zurückgegeben. Um diesen Prozess zu beschleunigen kann das DeviceTimeReq MAC-Kommando verwendet werden. Damit wird das Gateway aufgefordert einen Beacon zu senden. Nun kann das Endgerät in den Modus B wechseln.

Als Zweites setzt der MAC Layer des Endgerätes das Class B Bit im FCtrl Feld des Uplinks auf 1. Der MAC Layer ist auch verantwortlich die Pingslot und für die Beacons Empfangsfenster zu öffnen. Dabei muss mit der größten möglichen Abweichung der internen Uhr gerechnet werden und dementsprechend die Empfangsfenster angepasst werden. Diese darf pro Beacon nicht mehr als $\pm 1.3\text{msec}$ liegen. [SOR17c, S.73] Der Inhalt des empfangenen Beacon wird mit der Signalstärke dem Programm des Endgerätes zur weiteren Verarbeitung gesendet. Damit kann z.B. die innere Uhr nachgestellt werden oder Ortswechsel festgestellt werden.

4.2.2 Betrieb

Damit der Netzwerksender dem Endgerät mitteilen kann dass die Pingslot Frequenz und die Datenrate geändert werden soll gibt es den PingSlotChannelReq Mac-Kommando. Die neuen Werte sind in den Argumenten enthalten.

Das Endgerät kann die Periode der Pingslot zu einer beliebigen Zeit ändern. Ist dies der Fall, so muss das Endgerät in Klasse A wechseln und mittels dem MAC-Kommando PingSlotChannelReq die geänderte Periode mitteilen. Danach kann zurück in die Klasse

B gewechselt werden.

Falls einige länger als 2 Stunden kein Beacon empfangen wird, kann die Synchronisation mit dem Netzwerk verloren gehen. Dadurch funktioniert die Kommunikation in Klasse B nicht mehr und es wird in Klasse A gewechselt. Nun kann versucht werden eine Verbindung mit der Klasse A aufzubauen. Das Programm des Endgerätes kann versuchen wieder in Klasse B zu wechseln. Dieser Prozess kann sich immer wieder wiederholen.

Um auch innerhalb der minimal 2 Stunden in den kein Beacon empfangen wurde einen Kommunikation zu ermöglichen wird jedes Mal wenn ein Beacon verloren geht in den Beacon-Less Modus gewechselt. Dieser Modus orientiert sich ausschließlich an der internen Uhr. Um die Abweichung auszugleichen werden die Empfangsfenster immer früher begonnen und immer später beendet. Das bedeutet einen höheren Energieverbrauch aber auch eine höhere Wahrscheinlichkeit noch Daten zu empfangen obwohl die Uhren des Gateways und des Endgerätes auseinanderlaufen.

4.2.3 Singel / Multicast

In Klasse B können die Nachrichten als Singelcast oder als Multicast Nachrichten verwendet werden. Eine Singelcast Nachricht wird an des Endgerät das im DevAfdR Feld der Nachricht codiert ist gesendet. Im Multicastmodus wird das Paket an mehrere Endgeräte gesendet. Damit diese möglich ist müssen sich die Endgeräte dieselbe Multicast Adresse und die dazugehörigen Schlüssel teilen. Durch verschiedene Multicastadresse ist es möglich sogenannte Multicastgruppen zu erzeugen die nicht alle sondern nur ein Teil aller Endgeräte beinhalten. LoRaWAN gibt allerdings keine Methode vor wie die Adressen und Schlüssel verteilt werden. Diese Aufgabe muss in der Applikationsebene sprich im Programm der Endgeräte bzw. des Applikationsserver oder die Schlüssel und Adresse werden fest in den Programmcode programmiert.

In Multicastnachrichten sind keine MAC-Kommandos erlaubt. Nur Daten dürfen als Multicastnachrichten übertragen werden. Dies wurde eingeführt da Multicastnachrichten nicht dieselbe Robustheit wie Singelcastnachrichten haben. Die Nachrichten dürfen nicht bestätigt werden da sonst eine große Last für das Netz entstehen würde. [SOR17c, S.84]

4.2.4 Beacon

Wie schon erwähnt werden der Beacons verwendet um das Endgerät mit dem Netzwerk zu synchronisieren. Deswegen werden die Beacon Periodisch gesendet. Die Zeit zwischen zwei Beacon wir BEACKON_Period genannt. Die Endgeräte öffnen Empfangsfenster um diese Beacons zu empfangen. Ein Beacon zu übertragen dauert BEACON_RESERVED lange. Das Beacon-Empfangsfenster wird BEACKON_GUARD früher geöffnet um sicher zu stellen das Beacon auch wirklich erkannt werden kann. Außerdem wird die Beacon_GUARD benutzt um sicherzustellen dass kein Pingslot mehr geöffnet ist. Deswegen muss diese Beacon_GUARD mindestens so lang sein wie ein maximaler Pingslot. Das hat den Vorteil, dass nicht darauf gedacht werden muss wann ein Pingslot geöffnet wurde, da er im Zweifelsfall fertig ist bevor ein Beacon empfangen wird. Vergleichbar ist dies mit der Distributed Coordination die bei WLAN Anwendung findet.

Während versucht wird ein Beacon zu empfangen kann kein Pingslot geöffnet werden

Um Synchronisierungen durch die Beacon zu vermeiden, wie “alle Endgeräte wollen sofort nach den Beacon senden wollen“, wird mittels zufälliger Wartezeiten und zufälliger Pingslotperiode verhindert. Durch den Zufall wird die Wahrscheinlichkeit deiner Kollision geringer.

Beacons haben ihr eigenes Paketformat. Diese Pakete sind immer gleich lang. Dadurch kann auf Header verzichtet werden was auch der Verarbeitungsgeschwindigkeit zugutekommt. Wie auch ein normales LoRa-Paket, so besteht auch das erste Feld des Beacon-Paketes aus der Preamble. Danach folgt der BCNPayload. Der BCNPayload(Beacon Payload) lässt sich unterteilen in RFU, Time, CRC, GWSpecific, RFU, CRC. Die zwei CRC Felder weisen schon auf die logische Unterteilung in zwei hältern hin. Der erste Teil enthält Beacon spezifische Informationen (time und CRC). In dem Time Feld ist die Zeit seit 00:00:00, 01.01.1980 Modulo 2^{32} enthalten. das CRC Feld wird verwendet um die Korrektheit des Zeit und des RFU Feldes zu versichern. RFU steht wieder für “Reserved for Future Usage“ und wird nicht verwendet. Die andere Hälfte ist Gatewayspezifisch. Sie enthält das GwSpecific Feld und ein RFU Feld . Beide Fleder sind durch ein zweites CRC Feld abgesichert ist. Das GwSpezific Feld lässt sich unterteilen in InfoDesc und Info Felder. Das InfoDesc gibt an auf was sich das Infofeld bezieht. 0 gibt an das die GPS-Koordinaten der ersten Antenne folgen. 1 steht für die GPS-Koordinaten der zweiten Antenne und 2

steht für die GPS-Koordinaten der dritten Antenne. Die Werte 3 bis 127 sind noch solange sich im Info Feld Koordinaten enthalten kann dieses unterteilt werden in Längen- und Breitengrad.

Auch Klasse A kann den Beacon nutzen, um herauszufinden von welchem Gateway es gerade Daten empfängt und um eventuelle Standortwechsel festzustellen.

In Europa werden die Beacons auf einer festen Frequenz übertragen die sich nicht ändert. Nur über das MAC-Kommando PingSlotChannelReq.

4.3 Klasse C

C steht für CONTINUOUSLY Listening. Wie der Name schon sagt ist hier dauernd ein Empfangsfenster geöffnet. Dafür wird es ermöglicht fast Latzen frei zu übertragen. Dies bedeutet aber auch dass der Stromverbrauch am höchsten ist und somit die Klasse C nicht für den Batteriebetrieb geeignet.

Geräte die Klasse C implementieren sollen aus nicht die Klasse B implementieren das es sonst zu Fehlern kommen kann.

Diese Klasse verwendet die gleichen Empfangsfenster mit der gleichen Frequenz wie in Klasse A. Der große Unterschied besteht allerdings darin dass RX2 immer dann geöffnet ist wenn nicht gerade Daten an das Gateway gesendet werden oder RX1 geöffnet ist.

Auch in Klasse C ist es, wie in B, möglich Multicastnachrichten zu senden. Hierbei gelten die gleichen Regeln wie bei Klasse B.

5 Sicherheit

Sicherheit in netzwerkfähigen Systemen ist ein sehr wichtiges und heiß diskutiertes Thema. Da alle LoRa-Geräte über das Medium "Luft" die Daten übertragen, sind diese auch für alle empfangbar. Deswegen ist es wichtig die Daten zu verschlüsseln um zu verhindern dass Dritte die Daten mitlesen. genauso wichtig die Daten zu Authentifizierung und somit zu überprüfen wer die Daten gesendet hat. Auch verschlüsselte Daten können von Dritten mitgelesen werden. Allerdings können diese nichts mit den verschlüsselten Daten anfangen. Das einzige was Dritte mit den Daten tun können ist eine sogenannte Replayattack durchzuführen in dem sie das empfangene Datenpaket nochmal senden und so versuchen

die Kommunikation zu stören oder Fehlinformationen zu streuen. Um Replayattack entgegenzuwirken, werden Zähler oder Nonce mit den Daten mitgeschickt. Im folgenden Text wird betrachtend wie LoRa diese Sicherheitsfeatures implementiert hat.

5.0.1 Schlüssel

Schon der Beitritt eines Endgerätes muss verschlüsselt sein. Dafür werden zwei verschiedene Schlüssel verwendet. Die Join-Nachricht wird mit dem JSIntKey(Joining Session Encryption Key) verschlüsselt. Der Schlüssel wird mittels dem der aes128 Verschlüsselung generiert. Speziell wird der NWkKey mit der deinem 16Byte langem Wort verschlüsselt. Das Wort besteht am anfang aus der Konstant 0x06, danach folgt die DevEUI und zum Schluss wird mit =en aufgefüllt. Die Accept-Nachricht wird ebenfalls verschlüsselt. Allerdings wird hier der JSEncKey(Joining Session Encryption Key) verwendet. Dieser wird auf die Gleiche weise erzeugt, allerdings wird bei dem Verschlüsselungswort 0x06 durch 0x05 ersetzt. JSIntKey wird außerdem verwendet um den MIC-Code in der Join-Nachricht zu berechnen.

Alle MAC-Kommandos die an Port 0 gesendet werden, werden separat verschlüsselt. Dies bietet eine Höhere Sicherheit, selbst wenn Daten entschlüsselt werden können, so bleibt es unmöglich falsche MAC-Kommandos zu verschlüsseln und zu einem Endgerät zu senden. Hierzu wird der Network session encryption key (NwkSEncKey) verwendet. Dieser wird auf die selbe weise erzeugt wie JSIntKey. Allerdings besteht hier das Schlüsselwort aus 0x04, der JoinNonce, der JoinEUI und dem DevNonce. Diese Werte werden bei dem Beitritt mittels OTAA erzeugt. Siehe: ??.

Um die Integrität der Daten zu bestimmen werden zwei schlüssel verwendet. Der FNwkSIntKey oder lang Forwarding Network session integrity key wird benutzt um den MIC-Code für die Integritätsbestimmung abzuleiten. Auch dieser Schlüssel wird wie JSIntKey abgeleitet aber das Schlüsselwort besteht aus 0x01, der JoinNonce, der JoinEUI und DevNonce. Der SNwkSIntKey(Serving Network session integrity key) stellt das gegenstück da und wird verwendet den MIC-Code zu testen ob nichts verändert wurde. Hier besteht das Schlüsselwort aus 0x03, der JoinNonce, der JoinEUI und der DevNonce.

JSEncKey, JSIntKey, FNwkSIntKey, SNwkSIntKey, NwkSEncKey sind für jedes Endgerät unterschiedlich.

Momentan sind nur die MAC-Kommentare vor dem Mitlesen geschützt. Natürlich müssen auch die mitgelieferten Daten geschützt werden. Dies geschieht unter Verwendung des AppSKeys. Der Applikation Session Key wird von dem Applikationsserver und dem Endgerät verwendet. Dieser Schlüssel wird vom Programmierer vorgegeben, da er mit dem Schlüssel des Applikationsserver übereinstimmen muss. [SOR17c, S.50 ff]

5.0.2 Zähler

Jedes Endgerät hat drei verschiedene Counter. Ein Framecounter für Uplinkframes FCntUp und zwei für Downlinkframes NFCntDown und AFCntDown.

Der FCntUp counter wird für jeden Uplink der gesendet wird erhöht. Der stand wird in das FCnt feld der Nachricht mit eingefügt. Da das Gateway den selben zähler hat, kann es erkennen ob die Nachricht schon einmal gesendet wurde in dem es nur Nachrichten mit einem größeren FCnt Wert als der eigene FCntUp Zähler enthält.

Genau so wird mit den Downlink counter verfahren, bloß auf dem Endgerät. Im NFCntDown Zähler werden die MAC-Kommando Nachrichten gezählt. AFCntDown wird für alle anderen Nachrichten verwendet. [?, S.22 ff]

6 Live-Beispiel

wenn vorhanden.

7 Ausblick

Literatur

- [AVTP⁺17] ADELANTADO, FERRAN, XAVIER VILAJOSANA, PERE Tuset-PEIRO, BORJA MARTINEZ, JOAN MELIÀ-SEGUÍ und THOMAS WATTEYNE: *Understanding the Limits of LoRaWAN*. <https://ieeexplore.ieee.org/abstract/document/8030482>, September 2017. Eingesehen am 09.04.2019.
- [CB⁺17] CHEONG, PHUI SAN, JOHAN BERGS, , CHRIS HAWINKEL und JEROEN FAMAHEY: *Comparison of LoRaWAN Classes and their Power Consumption*. <https://ieeexplore.ieee.org/abstract/document/8240313>, November 2017. Eingesehen am 09.04.2019.
- [GAS17] GEMALTO, ACTILITY und SEMTECH: *LoRaWANTM SECURITY WHITE PAPER PREPARED FOR THE LoRa ALLIANCETM*. <https://lora-alliance.org/resource-hub/lora-alliance-security-whitepaper>, Februar 2017. Eingesehen am 09.04.2019.
- [SOR17a] SORNIN, N. (Herausgeber): *Exploring LoRa and LoRaWAN(todo)*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [SOR17b] SORNIN, N. (Herausgeber): *LoRaWANTM 1.1 Backend(todo2)*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [SOR17c] SORNIN, N. (Herausgeber): *LoRaWANTM 1.1 Specification*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [Tec15] TECHNICALMARKETINGWORKGROUP1: *A technical overview of LoRa® and LoRaWANTM*. <https://lora-alliance.org/resource-hub/what-lorawantm>, November 2015. Eingesehen am 09.04.2019.