

# Seminararbeit: Lorawan

Tobias Sigmann

21. Mai 2019

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung in Lora</b>	<b>3</b>
<b>2</b>	<b>Aufbau eines Lora-Netzwerk</b>	<b>4</b>
2.1	Gateway . . . . .	4
2.2	Netzwerkserver . . . . .	6
2.3	Applicationsserver . . . . .	6
2.4	Join-Server . . . . .	7
2.5	End-Gerät . . . . .	7
<b>3</b>	<b>LoraWan Funktionsweise</b>	<b>7</b>
3.1	Schichtenmodell . . . . .	8
3.2	Netzwerkbeitritt . . . . .	9
3.2.1	OTAA . . . . .	9
3.2.2	ABP . . . . .	10
3.3	Protokoll . . . . .	11
3.4	Übertragungsart . . . . .	15
3.4.1	Adaptive Data Rate . . . . .	16
<b>4</b>	<b>Lora Geräte Klassen</b>	<b>17</b>
4.1	Klasse A . . . . .	18
4.2	Klasse B . . . . .	20
4.2.1	Klassenwechsel A nach B . . . . .	21
4.2.2	Betrieb . . . . .	22
4.2.3	Singel / Multicast . . . . .	23
4.2.4	Beacon . . . . .	24
4.3	Klasse C . . . . .	25
4.3.1	Wechsel von A nach C . . . . .	26
<b>5</b>	<b>Sicherheit</b>	<b>26</b>
<b>6</b>	<b>Live-Beispiel</b>	<b>26</b>
<b>7</b>	<b>Ausblick</b>	<b>26</b>

# 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. LoRaWAN 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<sup>+</sup>17](alternativen: Sigfox, Ingenu, Dash7

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

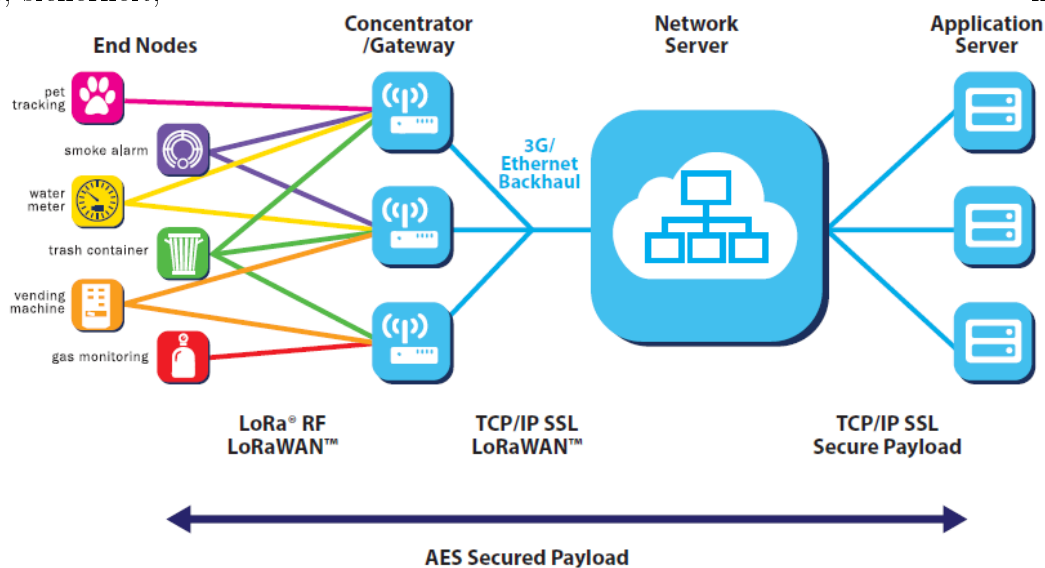
Energiesparend durch ADR (Adaptive Daten Rate) )

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 Logic. Hierzu ist ein Netzwerkservers zuständig der durch die Gateways das Netzwerk kontrolliert und steuert. Gleichzeitig stellt er die Verbindung zu einem Applicationsserver her an den er die vom Gateway empfangenen Daten sendet und von dem auch Daten an die Endgeräte, wieder über das Gateway, gesendet werden. Diese Architektur wurde gewählt um die Laufzeit der End-Geräte, Anzahl der End-Geräte, Qualität des Signals und Sicherheit des Netzwerkes möglichst hoch zu halten.

[Tec15] (Architektur hat großen Einfluss auf Batterie, Anzahl Teilnehmer, Qualität, Sicherheit, ...)



### 2.1 Gateway

Das Teilnetz, das aus dem Gateway und mehreren LoRa-End-Geräten besteht, ist sternförmig aufgebaut. Jedes End-Gerät 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 einem Netzwerkservers verbunden.

Ein Endgerät kann gleichzeitig an mehreren Gateways senden. Der Netzwerkservers ist zuständig die Pakete auf Duplikate zu überprüfen und nur solche nur einmalig an die Applikation zu senden. Ein weiterer Vorteil ist das kein Handover nötig ist, da alle Gateways fähig sind die Daten des Endgeräts zu verarbeiten bzw. weiterzusenden.

Durch die Sternförmige Architektur des Netzes und die Fähigkeit von allen in der Reichweite befindlichen Endgeräte Daten zu empfangen, muss ein Gateway mit vielen End-Geräten kommuniziert. Da es nicht möglich ist mit jedem Gerät nacheinander zu kommunizieren, muss dies gleichzeitig geschehen. Hierzu werden adaptive Datenraten und Mehrkanal-Multi-Modem-Transceiver verwendet um eine hohe End-Geräteanzahl zu ermöglichen. Außerdem hängt die Anzahl der Teilnehmer davon ab wie geschickt die Kanäle gewählt wurden, welche Datenraten verwendet werden und wie lange gesendet werden muss um die Daten zu senden (man spricht auch von der "Time-On-Air").

Durch die genannten Eigenschaften der Gateways wird auch eine gute Skalierbarkeit erzielt. Dadurch kann ein neues Gateway die Anzahl der Knoten um das 6 bis 8-fach erhöhen.

[Tec15] (Meistens wird ein netzförmiges Netz aufgebaut. Knoten leiten Nachrichten weiter => größere Reichweite aber kompliziert, erlaubt weniger Teilnehmer und energieaufwändig).

Lora Sternförmig => Energie-effizient, Knoten senden direkt an Gateways. Gateways senden an Server, Server muss doppelte Pakete filtern, Sicherheitscheck, ACK über bestes Gateway senden, datenrate anpassen.

Keine Handover

Gateway müssen viele Geräte handeln da Stern. erreichen durch (adaptive Datenrate, multi channel/multi modem transceiver) mehrere Nachrichten auch verschiedenen Channels gleichzeitig empfangen

Wichtige Faktoren (anz. channels, datenrate (time on air), payload länge, Sendehäufigkeit)

Skaliert sehr gut => gemacht für große Nutzerzahlen Neues gateway kann Knoten 6-8 x verbessern ) [SOR17]( Applikation Server -> Zentraler Server(leitend Pakete weiter) -> Gateway(wandelt lorawan in ip Pakete um) -> Endgerät/Knoten )

## 2.2 Netzwerkserver

Der Netzwekserver ist das "Herzstück eines jeden Lora-Netzwerkes. Im fallen viele Aufgaben zu. Er kann mit mehreren Gateways und mehreren Applications-server verbunden sein.

Die wichtigste Aufgabe ist das Steuern des Lora-Teils des Netzwerkes. Der Server verwaltet jedes End-Gerät separat indem es mit ihm den zu verwendenden Kanal aushandelt und die datenrate adaptiv kontrolliert wenn ADR(Adaptiv Data Rate) verwendet wird. Weiterhin überprüft er die empfangen Pakete auf ihre Korrektheit und Integrität und filtert Duplikate ,die durch das Empfangen des gleichen Signales an verschiedene Gateways, verursacht wurden. Dabei ermittelt er auch das Gateway das den besten empfang zu dem End-Gerät hat und nutzt dieses um Daten an das endgerät zu senden. Es ist nicht immer möglich Daten direkt zu senden. Um die Applikation-Server zu entlasten puffert der Netzwerkserver die Daten und sendet sie zum nächst möglichen Zeitpunkt zu senden. Desweiteren muss er die empfangenen Pakete "bestätigen und leitet join requests an den joiserver weiter

Eine weitere sehr Wichtige Ausgabe ist es eine API für den Applikations-server bereitzustellen um eine einfache und schnelle kommunikation zu ermöglichen.

[SOR17]( Sicherheit(zähler, ...), leitet pakete weiter, filtert pakete, merher gateway via ip verbunden, kontrolliert datenrate, kanäle, adaptiv data rate api.

## 2.3 Applicationsserver

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

## 2.4 Join-Server

Ein Join-Server wird benötigt um den Beitritt mittels OTAA zu ermöglichen. Der Server kann an mehrere Netzwerkserver verbunden sein und jeder Netzwerkserver kann mehrere Join server haben. Wenn ein Endgerät dem Netzwerk beitreten möchte leitet der Netzwerkserver die anfragen an den JoinServer weiter. Dieser führt dann die nötigen schritte des Beitritts aus wie z.B. ableiten von Schlüsseln. 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 Netz gesendet werden.

## 2.5 End-Gerät

Endgeräte sind Geräte die Informationen mittels Lora empfangen oder senden. Jedes End-Gerät ist mit einem bestimmten Applikationsserver verbunden.

Jedes Endgerät muss zur korrektur funktion mehrere wichtige Informationen Speichern.

ist das  
wichtig?

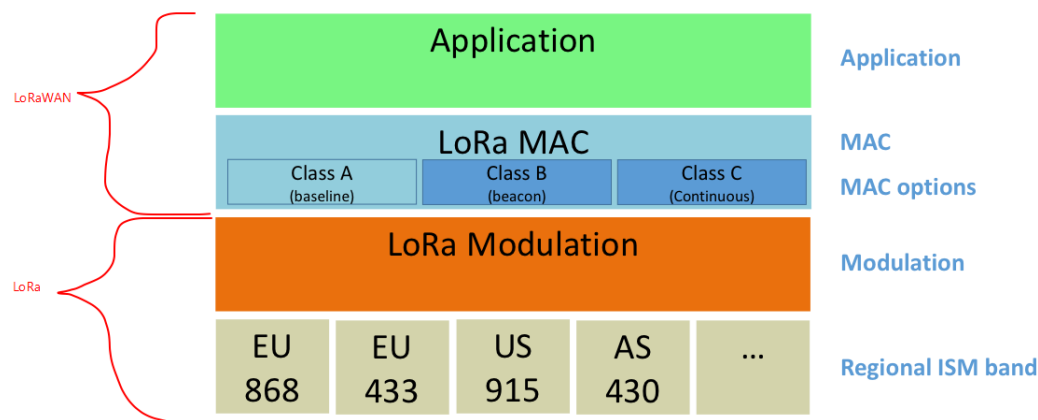
- DevEUI: Globale endgerätid die eindeutig für jedes endgerät definiert ist. (wie mac eines computers)
- JoinEUI: Globale Adresse des JoinServers an den die anfrage gehen soll. wird nur für OTAA geräte benötigt.
- NwkKey und AppKey: Werden verwendet um spätere Schlüssel abzuleiten und die kommunikation während des Joinens abzusichern. Dafür müssen sie sowohl dem Join-Server als auch dem Endgerät bekannt sein da sie nie übertragen werden.

## 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. [SOR17] (Geschwindigkeit ist kompromiss zwischen abstand/geschw. die untersch freuen-

zen bzw. Geschwindigkeiten beeinflussen sich nicht gegenseitig => keine Interferenz  
 Die Datenrate ist einstellbar, jedoch wird die Reichweite bei höherer Datenrate gemindert. Ein Vorteil von LoRa ist, dass die einzelnen Datenraten nicht interferieren und so jedes Endgerät seine eigene Datenrate unabhängig von den anderen verwenden kann. Außerdem wird die Datenrate und die Sendeleistung für jedes Gerät separat gesteuert (ADR, Adaptive Data Rate)

### 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 und Verschlüsseln der Daten zuständig.

Die unterste Schicht des LoRa Teils moduliert die verwendeten Frequenzen. In Europa muss das ISM-Band 868 verwendet werden, in den Vereinigten Staaten wird das Band 915 verwendet.

Die darüberliegende 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 kann. Mehr dazu im Kapitel Übertragungsart.

Über der LoRa Modulationsschicht liegt die erste LoRaWAN-Schicht, LoRa MAC. Diese Schicht ist für die Implementierung der einzelnen Endgeräteklassen und für das Übertragen der Steuerkommandos zuständig. Mehr zu den Klassen kann im Kapitel LoRa Geräte Klassen und im Kapitel Protokoll gelesen werden.



Die oberste schiecht nennt sich Applikationsschicht und ist dafür zuständig die Nutzdaten einer NACHricht passend zu verpacken zu verschlüsseln und zu authentifizieren.

[Tec15](Application, Lora MAC, MacOptins(Classes), LoraModialtion(Regionales ISB))

## 3.2 Netzwerkbeitritt

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

[SOR17](wei arten OTAA(Over the air activation), ABP(Activation by Personalization) Jedes gerät hat eine vorgegebene DevEUI (wie Mac adresse eines PCs), JoinEUI muss angegeben werden und adressiert den Join server

)

ABP steht für Äctivation by Personalizationünd bedeutet Wörtlich über- setzt Aktivierung durch Personalisierung.

Was  
macht

der?  
wohin

da-

mit?  
umshreiben

!!!!

### 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 folgene 4 Werte Vorgegeben werden. DevEUI, JionEUI , NwkKey, AppKey(Verschlüsselung des join requests).

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 replayattacs verhindert werden. Sie ist das beim ersten Join request 0 und sollte sich bei jedem Join-Request erhöhen. Außerdem muss sie auch dann noch gespeichert werden wenn kein Strom zur verfügung speht. Falls von dem gelichen endgerät eine Join-Reguest mit einer zu kleinen DevNonce empfangen wird, wird die NACHricht ignoriert und es ist nocht möglich dem Netzwerk beizutreten.

Die Accept Nachricht besteht aus eier JoinNonce, einem NetzwerkID Net\_ID,

einer Geräteadresse DevAddr, einer einstellungsfeld DLSettings , einer angabe wie lange auf eine antwort nach dem senden gewartet werden muss RxDelay und einer optionalen liste an Netzwerkparamerter CFList. Die JoinNonce wird außerdem benutzt im schlüssel wie AppSKey, ... herzuleiten. Für jedes Endgerät wird eine eigene JoinAccept 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.

Der NWKSEKEY 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.

Der letzte Wert heißt APPSKEY und sichert die kommunikation vom Endgerät zu dem Applikationsserver ab. Der Schlüssel wird genau wie der NWKSEKEY vom Netzwerkserver erzeugt und verwaltet.

Mehr Informationen zu den verschiedenen Schlüsseln finden Sie in dem Kapitel Sicherheit.

Wenn der Netzwerkserver den Beitritt des Endgerätes erlaubt sendet er eine Join-Accept nachricht zurück. Das Endgerät erwartet die nachricht nach JOIN\_ACCEPT\_DELAY1 oder JOIN\_ACCEPT\_DELAY2 nach dem Senden des Request. Sollte die Join-Accept nachricht zu einem anderen zeitpunkt gesendet werden, wird diese nicht empfangen weil das Endgerät nicht empfangsbereit ist. Die Nachricht enthält einstellungen für das Endgerät sowie die Id des Netzwerkes und die neue Adresse für das Endgerät. Um replayatacken zu verhindern enthält die nachricht zusätzlich eine JoinNonce. Diese wird für jedes endgerät separat geführt und muss größer sein als die zuletzt gesendete.

### 3.2.2 ABP

Die einfachste Art des Beitritts heist ABP was für Activation by Personalization zu deutsch Aktivierung durch Personalisierung steht. Hierbei muss lediglich vor inbetriebnahme des End-Gerätes 3 Konstanten definiert werden. Manche Hersteller "brennen" diese drei Werte fest in den chip ein, sodass er nicht geändert werden kann. Falls es nicht möglich ist dem hersteller die gewünschten

werte zukommen zu lassen, sind solche End-Geräte nicht für den Beitritt mittels ABP geeignet.

Als erstes muss die DeviceAddress angegeben werden. Diese Adresse existiert nur einmal im Netzwerk und wird verwendet um das Endgerät zu identifizieren. Die Adresse wird vom Netzwerkserver erzeugt und muss manuell von dort kopiert werden. Die verwendete Frequenz entspricht der RX1 bzw. RX2 aus dem Kapitel Klasse A.

Mit Hilfe dieser 3 Werte kann die Join-Request - Join-Accept-Prozedur übersprungen werden. Daher kann das Gerät direkt einen LoRa-Netz beitreten wenn es angeschaltet wird und muss nicht erst alle Schlüssel neu ableiten und aushandeln. Allerdings ist diese Methode deswegen weniger sicher, da immer die selben Schlüssel verwendet werden.

Nach der Verbindung muss das ResetInd Mac command im FOpt-Feld gesendet werden solange bis ein ResetConf erhalten wird. Nun ist das Gerät im Netzwerk und kann unter der eingestellten Adresse und mit den eingestellten Schlüsseln arbeiten.

Sobald dem Netzwerk erfolgreich beigetreten wurde werden die benötigten Schlüssel aus den vorher gesetzten Werten abgeleitet. Genaueres dazu in Kapitel Sicherheit.

### 3.3 Protokoll

Das LoRaWAN Protokoll ist optimiert für batteriebetriebene Endgeräte die drahtlos kommunizieren möchten. Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte. Die Modulationstechnik und eine Adaptive Datenrate (ADR). Auch die One-Hop-Architektur trägt zur Energieeffizienz bei. Die Art wie LoRa-Signale moduliert wird in Kapitel Übertragungsart besprochen. Um die genannten Eigenschaften und das LoRa-Netzwerk zu steuern werden sogenannte MAC commands verwendet. Diese werden vom Netzwerkserver oder von einem Endgerät gesendet. MAC steht hierbei für "Media Access Protokoll" und bietet die Möglichkeit die Kommunikation mit den Endgeräten, Frequenzen, Kanälen und vieles mehr zu steuern. Da die Kommandos nur für den

Netzwerkserver und die Endgeräte von Bedeutung sind, werden diese nicht an den Applikationsserver gesendet sondern am Netzwerkserver herausgefiltert. Im folgenden wird näher auf die MAC Kommandos und die Paketstruktur eingegangen.

Jedes Paket besteht aus grundlegend aus 2 Feldern (Preamble, PHY Payload). Falls es sich um ein Uplinkpaket handelt wird noch ein CRC Code hinzugefügt (Preamble, PHY Payload, CRC). In diesem Fall spricht man von einem impliziten Paket oder von dem impliziten Modus. Impliziter Modus bedeutet, dass es kein Payload, Codierungsrate und der CRC Längenangabe gibt und diese somit eine feste zuvor definierte Länge haben. Im expliziten Modus werden noch 2 Felder hinzugefügt, PHDR und PHDR\_CRC. Somit sieht ein explizites Paket folgendermaßen aus (Preamble, PHDR, PHDR\_CRC, PHY Payload). Auch hier gilt, in allen Fällen eines Uplinkpaketes wird am Ende ein CRC Feld angefügt => (Preamble, PHDR, PHDR\_CRC, PHY Payload, 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 wird.

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 PHY Payloads und die Zieladresse beinhalten sollte. Das PHDR\_CRC Feld wird benutzt, um sicherzustellen, dass die empfangenen Werte korrekt sind mittels des CRC Verfahrens.

Wie schon mehrfach erwähnt wird in Uplinknachrichten ein zusätzliches CRC Feld verwendet. CRC steht für Cyclic Redundancy Check und wird verwendet, um die Korrektheit der Nachricht zu bestätigen. PHDR, PHDR\_CRC und das CRC Feld werden automatisch vom dem Funktransceiver (Modul als Empfänger und Sender) hinzugefügt.

Die bis jetzt behandelten Felder des LoRa Paketes wurden alle von der LoRa Modulationsebene erstellt.

Die darüberliegende Ebene "LoRa MAC" fügt nun das PHY Payload Feld ein. PHY Payload steht für Physikalische Payload. Es gibt 3 mögliche PHY Payloads. Entweder wird ein MAC Payload eingefügt oder es werden Join-Rejoin was heißt PHY Payload?

request oder aber es wird 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 Headders und zusätzliche felder benötigt. Deswegen lässt sich das Feld weiter unterteilen in (MHDR, MACPayload). Für den Fall das der MACPayload keine 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 Unterfelder MHDR | FHDR | FPort | FRMPayload festzustellen. Diese unbekannten felder werde im laufe des kapitells noch behandelt.

Das MHDR Feld beschreibt wie die Daten im MACPayload Feld zu deuten sind. Wieder wird dieses Feld in Unterfelder Unterteilt. MTType, 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 Usage" Deutsch "für zukünftige verwendung reserviert". Daher kann dieses Feld in der version 1.1 und niedriger ignoriert werden. Im Major Unterfeld wird verwendet um das Format 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 den 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 Datengesendet werden, kann das FRMPayload Feld auch MAC kommandos enthalten. In dem Feld FPorts wird angegeben an welchen port und somit an welche teilapplikation die Daten geleidet werden. Es gib einige feste Ports. z.B. Port 0 Zeit an das das FRMPayload Feld MAC commands endthält, 0x01 bis 0xDF sind Anwendungsspezifische Ports und Port 244 ist für das LoRaWan Test Layer protokoll reserviert. Falls ein andere Port als die geraden genannten angegeben wird, wird die nachricht verwerfen. Erneut kann der FHDR "Frame Header" in einzelne Felder unterteilt werden (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. Damit schützt man sich vor replay Attacks. Im FOpts feld können bis zu 5 MAC kommandos parallel zu Dtaen übermittelt werden. Die Anzahl kommt auf die mege der mitgelieferten variablen an . Das Letzte feld das in Unterfelder unterteilt wird ist das FCTRL feld. Hier wird das verhalten des Gerätes gesteuert sowie nachrichten acknowledged. Es gibt leichte unterschiede für ein Up-Link und für Down-link 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 Adaptive Data Rate. Mit dem Ack Feld können empfangene nachrichten markiert werden. Ob Nachrichten bestätigt werden müssen steht im MType feld. In dem FOptsLen feld wird die länge des FOpts feldes mitsamt des Headers eingetragen .

Ein Downlinkpaket hat zusätzlich ein RFU feld das nicht verwendet wird und ein FPending feld. In diesem feld kann das Gateway bzw der Netzwerks server dem Endgerät mitteilen, dass noch mehr Daten zu senden sind und mehr empfangsfenster geöffnet werden müssen.

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. .

[SOR17]( Mac commands werden benutzt um geräte zu steuern => frequenzen zu ändern, ... Application wird diese nie erhalten, läuft zwischen netzwerkserver und lora gerät ab. Verschlüsselt hier oder da. aufbau: 1byte command, x byte extra data. müssen vom empfangenden acknowledged werden. Reihenfolge ist zu beachten. Alle nachrichten in einem frame müssen auch in einem frame ack werden. => Macbuffer ermöglicht dies. Wenn buffer überläuft werden die ältesten ack.

CRC usw wurden von sender erstellt und eingefügt. )

[CB<sup>+</sup>17]( Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte. Die Modulationstechnik und eine Adaptive Datenrate (ADR) )

counter  
in  
gerät  
ein-  
führen  
und  
leicht  
erklä-  
ren  
wie  
das  
mit  
den  
coun-  
ter  
geht  
länge  
rein-  
schrei-  
ben?  
und  
er-  
klärt  
das  
mac  
va-  
riablen  
haben  
kön-  
nen  
da  
steht  
noch  
was  
von  
port  
und

### 3.4 Übertragungsart

Um die Enstandenen Pakete in Signale umzusezenun didese effizient und gleichzeitig übertragen zu können nutzt LoRa Chirp Spread Spectrum (CSS). Hierbei werden die Frequenz über eine gewisse Zeit hinweg verändert. Durch erkennen in welche richtig, ansetigen oder Abfalleb, die frequenz verändert wird, können 1 und 0 Codiert werden. Man spricht bei einem Bit von einem Chrip-Impuls. Durch aneinanderreihe der verschiedenen impulse ist es möglich mehere Bits nacheinander zu übertragen. Das entstandene Signal wird auch al Sub-Chrip bezeichnet. Durch verwenden von Unterschiedlichen anstiegs und abfalszeiten ist es möglich mehere Signale auf der Selben frequenz zu übertragen ohne das die Signale sich gegenseitig stören. Dies nennt man Spread Factor. Auserdem kann die Paralelität durch verschiedene Frequenzbereiche verbessert werden. CSS ist besonders für große reichweiten geeignet und somit auch bestens für Lo- nachprüfen ra. Am besten ist das Signal wenn das endgerät nahe am Gateway ist. Je weiter es entfert desto schlechter wird das signal. Um die kommunikation trotzdem zu ermöglichen wird der Spredding Factor erhöht. Dies hat auch den Vorteil dass der Energieaufwand gering gehalten wird. Analog wie Menschen auf einer Party nicht immer versuchen lauter zu sprechen, sonder auch versuchen besonders langsam und deutlich zu sprecehn.

Mann sichbrcht auch von Channels. CHannels können beliebeig benutzt werde. es gibt alerdings zwei regeln zu beachtn.

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 der selbern Frequenz senden möchten kommt es zu einer kollision. Dadurch kann das Gateway die empfangenen Dta- ne nicht mehr auswerden. Deswegen warten beide Endgeräte eine zufällige, unterschiedliche Zeit abe bist sie erneut senden.

[SOR17]( Knoten können zu jeder Zeit, auf beliebigen Kanälen, beliebig

schnell, beliebig lange senden, solange folgende regeln befolgt werden.

- Channels werden per Pseudozufallszahl geändert
- Sendezeit erfüllt die Regionalen Bestimmungen

)

[AVTP<sup>+</sup>17]( Chrip Signal => Zeitliche Änderung in Trägerfrequenz(höhere Frequenz als Datenrate)(positiv chrip/negativ chrip)

Datensignal wird in Chrip Signal moduliert. Resultierende Signal ist breitbandiger als Datensignal. Maximale Datenrate auch mit Rauschen erreichbar.

Durch orthogonale SSpread Factor"mehrere Signale auf einem Chanel ) [Tec15](normal FSK, schon sehr effcient. Lora "chirp spread spectrum odulation". Ist wie FSk aber größere Rechiweite, robuster. Stammt aus dem Militär/raumfahrt.Lora als erstes für kommerziellen billigen Einsatz.

Spread spectrum => signale sind Ortohonal für versch. spreizraten, fakto koreliert mit datenrate => verschiedene Datenraten auf einem Kanal

Nähere Geräte sind schneller => höhere Datenrate => kürzee übertragungsdauer und lassen somit merh zeit für andere, => bessere Batterielaufzeit. Deswegen sidn symetrische up/downlinks nötig.) Frequenzhopping, spread spectrum, code-channels

soll ich  
ver-  
hält-

### 3.4.1 Adaptive Data Rate

Adaprive Data Rate oder kurz ADR wird verwedent 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 LoraPaketes das ADR Bit gesetzt ist, siehe Protokoll. Die Steuerung duch ADR findet durch den Netzwerkserver statt. Sobald der Netzwerkserver bereit ist, stzt er das bit im downlink. IST das endgerät ebenfalls bereit setzt es ebenfalls das bit und ADR kann verwendet werden. Flass es nicht möglich sein sollte ADR zu verwenen sollte es durch das Applikaionslayer gesteuert werden.

niss  
zwi-  
schen  
data-  
rat sf  
und  
ener-  
gie  
rein-

Die Steuerung dfindet durch spezielle MAC kommandos statt. Standartgemäs wird die höchset übertragunsstärke verwedent allerdings auch die gerings-

ma-  
chen?  
warum?



te Übertragungsrate. Falls diese gedrosselt werden soll wird vom Netzwerkserver das **LinkADRReq** MAC command benutzt. mIt diesem wird das Endgerät informiert, dass es die data rate, transmit power, repetition rate or channel ändern soll. Was auf welchen wert geändert werden soll wird in die Parameter codiert. Sobald die Werte geändert wurden, muss periodisch überprüft werden ob das Netzwerk die Nachrichten noch bekommt. Deswegen wird jedes mal wenn der uplinkframecount erhöht wird, wird der ADR\_ACK\_CNT counter verwendet. Wenn dieser counter ein gewissen schwellenwert (ADR\_ACK\_Limit) überschreitet, iwrd das ADRACKReq bit gesetzt. Dieses signalisiert den Netzwerkserver das er mit einem Uplink ein Downlonk senden muss um die Verbindung zu bestätigen. Falls dieser Downlink nicht in ADR\_ACK\_Delay frames empfangen wird, wird zuerst die übertragungsstärke auf max 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 siese schon minimal ist müssen alle chanles wiederverwenet werden. Dies wird solange probiert biss eine verwbindung hergestellt werden kann.

[AVTP<sup>+</sup>17]( Datenrate und Funkfrequenz(RF) werde passend zum Abstand angepasst

nahe Knoten => hohe Datenrate => kurze Sendezeit => weniger RF-Power kann nach Bedarf geändert werden

=> immer möglichst schnelle senden => weniger Energie

) [CB<sup>+</sup>17](crip signal)

## 4 Lora Geräte Klassen

Um maximal energie zu sparen aber trotzdem die möglichkeit dass die endgeräte agiel 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 A können werden als "high class Joinen nur in A beschrieben?

End-Devices" genannt. wohin

[SOR17](Geräte müssen mindestens A können, alle die mehr können werden auch "high class End-Devices" genannt) Vielleicht zu klein => in anderes Kapitel stopfen. Bei mehrfacher Übertragung wird nicht erhöht mit counter?

Die Endgeräte sind je nach Kommunikationsart/Protokoll Art in drei Klassen (A, B und C) unterteilt.

Jede Klasse hat 3 Counter FCntUP(Pro uplink ++), FCNTDown(pro downlink außer port 0 => mach), AFCntDown(port ungleich 0 dann ++) (nur beschreiben wie diese grob funktionieren) Zähler sollen nicht flüchtig sein (Batteriewechseln kein reset) bei neuverbinden müssen alle Counter auf 0 gesetzt werden. Counter müssen auf beiden Seiten gleich gehalten werden (Synchron geführt) Wenn Nachricht empfangen ist muss der darin enthaltene Counter größer sein als der eigene.

die Counter Werte sollen so weit wie möglich nur einmal verwendet werden.

) [Tec15](Asynchrone Knoten wegen Batterie => Event/Scheduler gesteuert verwendet ALOHA)

Normal Netze müssen sich synchronisieren und Nachrichten abrufen. Lora partiell nicht => laut GSMA 3 bis 5 fach effizienter)

zur besseren Anpassung/ Anpassung an Batterie

EU: 10 Kanäle (8: 250bps bis 5.5kbps) (1: FSK 50kbps) (high rate Lora 114kbps)  
)

## 4.1 Klasse A

Klasse A wird auch (All end-Device) genannt und zeichnet sich durch sehr geringer Stromverbrauch aus. Die Kommunikation kann bidirektional stattfinden, allerdings muss die Kommunikation von dem Endgerät gestartet werden. Das bietet die Möglichkeit dass das Endgerät, wenn keine Daten gesendet werden müssen, in einen sehr sparsamen Schlafmodus wechselt. Um das Endgerät nicht zum Aufwachen zwingen zu müssen, wurde auf einen Hardbeat oder ähnliches verzichtet. Dadurch kann das Endgerät so lange schlafen wie es möchte.

Somit ist die Klasse A auch die potenziell Stromsparende Endgeräteklasse. Die Klasse A erlaubt außerdem das das Endgerät andere Protokolle schickt solange es keine LoRa Daten sendet oder empfängt.

Das Endgerät startet die Kommunikation in dem es Daten an das Gateway sendet(uplink). Daraufhin hat das Gateway die Möglichkeit 2 mal Daten zum Endgeräte senden(downlink). Die Downlinkfenster werden RX1 und RX2 genannt. Da die Kommunikation asynchron stattfindet, muss das Endgerät warten bis die uplinkphase abgeschlossen ist.

Die empfangsfenster RX1 und RX2 müssen mindestens solange geöffnet bleiben das sie eine beginnende Übertragung feststellen können. Falls keine Übertragung empfangen wird, wird das Fenster wieder geschlossen. Anderenfalls werden die Daten empfangen. Das Empfangsfenster RX1 wird nach RECIEV\_DELAY1 zeiteinheiten  $\pm 20\text{msec}$  nach beendigung des Uplinks geöffnet. Es wird die selbe Frequenz und Datenrate verwendet die auch bei den uplink 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  $\pm 20\text{msec}$  nach beendigung des Uplinks geöffnet. Allerdings ist die Datenrate und Frequenz fest. Nur Mittels spezieller MAC commands kann dies verändert werden.

Für alle join / rejoin Aktivitäten wird immer die Klasse A verwendet . Schon

[SOR17]( radio packet explicit mode, vom Gateway(1) zum Knoten(1), erklärt ausgelöst vom Netzwerkservers, auch Multikasts möglich, (Preamble, PHDR, oder PHDR\_CRC, PHYPayload) Um Nachricht kurz zu halten kein CRC am Ende, woanach Receiver\_Delay1 / Receiver\_Delay2 kann empfangen werden (rx1, rx2) ders

Fenster müssen lange genug für Preamble auf bleiben=> wenn erkannt wird empfangen wenn nicht Fenster wieder zu. Es darf nur gesendet werden wenn beide Fenster zu sind. ==>Es ist auch erlaubt andere Protokolle zu sprechen wenn nicht gesendet oder gehört wird.<== ) [SOR17]( Frequenz abhängig von Uplinkfrequenz, Datenrate abhängig von Uplinkdatenrate, wird nach Receiver\_Delay 1  $\pm 20\text{ msec}$  erwartet, Datenrate auch abhängig von Regionalen Regeln, Standard: Datenrate = Uplinkdatenrate ) [SOR17]( feste Fre-

quenz/Datenrate, nach Delay2 +/- 20 msec, Frequenz/Datenrate mittels MAC änderbar ) [SOR17]( Öffnungslänge muss für Preamble ausreichen, nach RX1 + MIC und Authentizitätscheck muss nicht zwingen RX2 geöffnet werden, Sender muss in einem der beiden Fenster stattfinden, Falls Downlink über beide Fenster => Frames müssen gleich sein. Knoten dürfen nicht während empfangen/ zwischen RX1 und RX2 senden, andere Protokolle dürfen gesprochen werden wenn gesendet werden darf )

## 4.2 Klasse B

Die Klasse B (B für BEACON) bietet bidirektionale Kommunikation mit einer deterministischen downlink Latenz. Um diese Latenz zu gewährleisten, muss die Kommunikation Synchron ablaufen. Außerdem muss festgestellt werden, ob das Endgerät bzw. das Gateway noch in Reichweite ist. Dies wird mittels eines periodischen "beacon" durchgeführt. Dieser Beacon wird regelmäßig vom Gateway gesendet und dient der Synchronisation der Endgeräte. Zeitpunkten gesendet werden realisiert. Die Latenz ist einstellbar und kann bis zu 128 Sekunden. 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 selbstständig feststellen wenn es einen Beacon mit einer unbekannten ID bekommt und durch einen Uplink dem Server mitteilen dass es 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 "beacons" nicht schlafen kann, ist die Klasse B für den Batteriebetrieb gedacht.

[SOR17](wird verwendet wenn mehr Bedarf für Empfangsfenster ist. Hierzu ist ein Synchronsignal nötig=> zu bestimmten Zeiten kann damit empfangen werden Gateway sendet Beacon für Synchronisation. Um Daten empfangen zu werden werden Empfangsslots => Pingslots verwendet, werden periodisch geöffnet und mittels Beacon synchronisiert. Normalerweise werden diese schnell

geschlossen außer es wird etwas empfangen. Gateway dessen beacon benutzt wird, wird nach empfangsqualität ausgewählt. Wenn neuer/unbekannter Beacon von einem anderen Gateway empfangen wird, wird der netzwerkserver benachrichtet und dieser entscheidet welcher verwendet wird (passt rote an).

Das Netzwerk muss die standard ping-slot periode Datenrate und kanal kennen.

Um ein gerät auf klasse B zu kommen muss erst von Klasse A gewechselt werden.

Entgeräte müssen Netzwerkserver über position informieren. Dies kann über eine leere nachricht passieren oder eine normale (uplink).

Das beacon und die enthaltenen daten werden an die applikation geschickt. Der server kann den beacon auswerten. zwischen beacon und uplink wird random time verwendet um kolisionen zu verhindern. änderungen an ping-slot periode .. muss mitgeteilt werden. Hierzu ist klasse A nötig => wechel zu A, wechel zu B. Nachschuen  
wie ge-  
nau

Beacon wird genutzt um clockdrift auszugleichen. Wenn kein beacon empfangen wird => Beaconless mode. Dieser wird bis zu 2 stunden beibehalten. Reines verlassen auf interne Uhr. Wenn beacon empfangen wird, wird zeit zurückgesetzt. ) das  
funk-  
tio-  
niert

#### 4.2.1 Klassenwechsel A nach B

Um einen Wechsel überhaupt zu ermöglichen muss der Netzwerkserver die default ping-slot periodem die ping-slot datenrate und den Ping-slot channel 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 be- erklären

schleunigen kann das DeviceTimeReq MAC kommando verwendet werden. Damit wird das Gateway aufgefordert ein beacon zu senden. Nun kann das endgerät in den modus B wechseln.

Als Nächstes setzt der MAC Layer des endgerätes das Class B Bit im FCtrl field des Uplinks auf 1. Dadurch ist er auch verantwortlich die Ping slots und für die Beacons zu öffnen. Dabei muss mit der größt 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. Der Inhalt der empfangenen Beacons wird mit der Signalstärke und das Programm des Endgerätes zur weiteren Verarbeitung gesendet. Damit kann z.B. dem LoRaWAN layer angewiesen werden die Uhr nachzustellen.

[SOR17] (Endgerät fordert LoRaWAN layer an. Layer sucht beacon. Mac command DeviceTimeReq um schneller beacon zu bekommen nutzen. Danach wird das ClassB field auf 1 gesetzt. Bei den geöffneten fenstern wird der maximal mögliche clockdrift berücksichtigt. Downlink läuft wie bei A ab.

)

#### 4.2.2 Betrieb

Damit der Netzwerkserver dem Endgerät mitteilen kann dass die pingslots frequenz und/oder die Datenrate geändert werden soll gibt es den PingSlotChannelReq Mac kommando. Die werden sind in den argumenten enthalten.

Das Endgerät kann die Periode der Pingslots zu einer beliebigen Zeit ändern. Ist dies der Fall, so muss das Endgerät in Klasse A wechseln mit mittels dem MAC kommando PingSlotChannelReq die geänderte periode mitteilen. wird danach kann zurück in Klasse B gewechselt werden. andere

Falls einige länger als 2 Stunden kein Beacon empfangen wird, kann die gespeicher synchronisation mit dem Netzwerk verloren gehen. Dadurch funktioniert die Kommunikation in Klasse B nicht mehr und es wird in Klasse A gewechselt. Da 1/s sich nun die Kommunikationsstrategie verändert muss mit einem Uplink in dem  $\Rightarrow \text{hz}$  das ClassB Field 0 ist, der Netzwerkserver informiert werden. 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 maximal 2 Stunden in denen kein Beacon empfangen wurde eine 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 den Drift 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.

drift?

### 4.2.3 Singel / Multicast

Die Downlink der Klasse B unterscheidet sich nicht von denen der Klasse A. Allerdings kann sich der Frequenzplan unterscheiden.

In Klasse B können die Nachrichten als Singelcast oder als Multicast Nachrichten verwendet werden. Eine Singelcast Nachricht wird an das Gerät das im DevAddr der Nachricht codiert ist gesendet. Im Multicastmodus wird das Paket an alle Endgeräte gesendet. Damit das möglich ist müssen sich die Geräte die selbe Multicast Adresse und die dazugehörigen Schlüssel teilen. Durch verschiedene Multicastadressen 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 also in der Applikationsebene sprich im Programm der Endgeräte oder direkt bei der Personalisierung (Programmierung) erledigt werden.

In Multicastadressen sind keine MAC Kommandos erlaubt. Nur Daten dürfen als Multicastnachricht übertragen werden. Dies wurde eingeführt da Multicastnachrichten nicht die selbe Robustheit wie Singelcastnachrichten haben. Die Nachrichten dürfen nicht acknowledged werden. Das Fpending zeigt an dass mehr unconfirmed Multicastnachrichten zu senden sind. [SOR17] (separate Adresse für Multicast / confirmed) festgelegt durch Layer oder manuell für Gruppenmulticast Nicht für MAC confirmed

geeignet, )

#### 4.2.4 Beacon

Wie schon erwähnt wird der Beacon verwendet um das Endgerät mit dem Netzwerk zu synchronisieren. Deswegen wird dieser Periodisch gesendet. Die Zeit zwischen zwei Beacons wird `BEACON_Period` genannt. Die Endgeräte öffnen Empfangsfenster um diese Beacons zu empfangen. Ein Beacon zu übertragen dauert `BEACON_RESERVED` lange. Das Beacon wird `Beacon_GUARD` früher geöffnet um sicher zu stellen das Beacon auch wirklich zu empfangen. Während versucht wird ein Beacon zu empfangen kann kein pingslot geöffnet werden. Ausserdem wird die `Beacon_GUARD` benutzt um sicherzustellen das kein Ping slot mehr geöffnet ist. Deswegen muss diese `Beacon_GUARD` mindestens so lang sein wie ein maximaler pingslot. Ein weiterer vorlesungsbezug? vorteil ist, dass nicht darauf geachtet werden muss wann ein pingslot geöffnet wird, da er sowieso im zweifelsfall fertig ist bevor ein beacon empfangen wird.

Um Synchronisierungen durch die Beacons zu vermeiden, wie alle Endgeräte wollen sofort nach dem beacon senden wollen, wird mittels zufälliger warte, pingslot zeiten und zufälliger pingslotanzahlen verhindert.

Beacons haben ihr eigenes Paketformat. Diese Pakete sind immer gleich lang. Dadurch kann auf header verzichtet werden was auch der Geschwindigkeit der verarbeitung zu gute kommt. Wie auch ein normales LoRaPaket, so besteht auch das erste Feld des Beaconpaketes aus der Preamble nur das die des Beaconpaketes länger dauert was ein bemerkenswerter der übertragung wahrscheinlicher macht. Danach folgt nur noch der BCNPayload. Der BCNPayload lässt sich unterteilen in RFU, Time, CRC, GWSpecific, RFU, CRC. Die zwei CRC Felder weisen schon auf die logische unterteilung in zwei hälften hin. Der erste Teil enthält beacon spezifische informationen (time und CRC). In dem Timefeld ist die zeit seit 00:00:00, Sunday 6th of January 1980 (start of the GPS epoch) modulo  $2^{32}$  enthalten. das CRC feld wird verwendet um die korrektheit des Zeit und des RFU Feldes zu versichern. Die andere hälfte ist Gatewayspezifisch. Sie enthält das GWSpecific feld und ein RFU feld das auch durch ein



zweites CRC feld abgesichert ist. Das GwSpecific feld lässt sich unterteilen in InfoDesc und Info felder. Das InfoDesc gibt an auf was sich das Infofeld bezieht. 0 GPS coordinate of the gateway first antenna 1 GPS coordinate of the gateway second antenna 2 GPS coordinate of the gateway third antenna 3:127 RFU 128:255 Reserved for custom network specific broadcasts. Sonst kann sich im infofeld koordinaten enthalten kann dieses unterteilt werden in Längen und Breitengrad.

Auch Klasse A kann den beacon somit nutzen um herauszufinden von welchem gateway es gerade Daten empfängt und um somit eventuelle Standortwechsel festzustellen.

In Europa werden die Beacons auf einer festen Frequenz übertragen die sich nicht ändert außer über das MAC kommando PingSlotChannelReq. Auf anderen Kontinenten kann es sein dass Frequenzhopping angewendet wird.

regionale  
para-  
meter  
er-  
wähnt?

### 4.3 Klasse C

C steht für CONTINUOUSLY Listening. Wie der Name schon sagt wird hier unaufhörlich ein empfängssender geöffnet. Dadurch wird es ermöglicht fast Latenzfrei zu übertragen. Dies bedeutet aber auch dass der Stromverbrauch am höchsten ist und somit nicht für den Batteriebetrieb geeignet. Das Gateway kann immer Daten senden außer wenn das Endgerät gerade Daten sendet. Hier sind Geschwindigkeit von bis zu 50mb möglich.

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 den gleichen Funktionen 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. Also auch während. Außerdem stehen die gleichen MAC kommandos und zwei zusätzliche zur Verfügung.

suche  
normal

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

net?

[SOR17] (öffnet RX1 und RX2 Fenster wie in Klasse A. Immer wenn nicht

gesendet wird oder RX1 offen ist, ist RX2 offen. Multicast ist auch möglich. )

#### 4.3.1 Wechsel von A nach C

Da es kein ClassC Fled in einem LoRapaket gibt, wurde für das umschalten in ClassC mode MAc kommandos eingeführt. Das endgerät sendet das Device-ModeInd commando. ALs parameter kann es 0 für Klasse A und 2 Für klasse C angeben. Der Netzerkserver kann mit DeviceModeConf welches den wert der klasse enthät indas gewächselt wurde.

## 5 Sicherheit

Lora bietet die end-to-end Sicherheit an, indem es die Signale zweimal verschlüsselt.

Die erste Verschlüsselung dient dazu die gesendeten Daten vor eventuellen Mithörern zu verschlüsseln. Die Verschlüsselung geschieht mit einem 128-bit Network-Session-Key.

Die zweite Verschlüsselung wird bis zur endgültigen Weiterverarbeitung der Daten auf z.B. einen Server verwendet und ist ein 128 bit Application-Session-Key.

Das zur Verschlüsselung verwendete Protokoll ist AES. Auch zu Authentifizierung und zur Überprüfung der Integrität wir AES verwendet. [GAS17] [Tec15](Applikationsverschlüsselung(schutz der Daten for mitlesen) Netzwerk(Autentiizierung der Knoten) AFS, Key Exnage IEEE EU164) [SOR17](symetrischer Schlüssel => nur einer benötigt, Sessionkey ist abgeleited von Knoten-rootkey. JoinServer setllt verbindung der Keys her. )

## 6 Live-Beispiel

wenn vorhanden.

## 7 Ausblick

# Literatur

- [AVTP<sup>+</sup>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<sup>+</sup>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: *LoRaWAN<sup>TM</sup> SECURITY WHITE PAPER PREPARED FOR THE LoRa ALLIANCE<sup>TM</sup>*. <https://lora-alliance.org/resource-hub/lora-alliance-security-whitepaper>, Februar 2017. Eingesehen am 09.04.2019.
- [SOR17] SORNIN, N. (Herausgeber): *LoRaWAN<sup>TM</sup> 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 LoRaWAN<sup>TM</sup>*. <https://lora-alliance.org/resource-hub/what-lorawantm>, November 2015. Eingesehen am 09.04.2019.