

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45

LoRa Specification

Copyright © 2015 LoRa Alliance, Inc. All rights reserved.

NOTICE OF USE AND DISCLOSURE

Copyright © LoRa Alliance, Inc. (2015). All Rights Reserved.

The information within this document is the property of the LoRa Alliance (“The Alliance”) and its use and disclosure are subject to LoRa Alliance Corporate Bylaws, Intellectual Property Rights (IPR) Policy and Membership Agreements.

Elements of LoRa Alliance specifications may be subject to third party intellectual property rights, including without limitation, patent, copyright or trademark rights (such a third party may or may not be a member of LoRa Alliance). The Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

This document and the information contained herein are provided on an “AS IS” basis and THE ALLIANCE DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO (A) ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OF THIRD PARTIES (INCLUDING WITHOUT LIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING PATENT, COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT.

IN NO EVENT WILL THE ALLIANCE BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OF BUSINESS, OR FOR ANY OTHER DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR IN TORT, IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The above notice and this paragraph must be included on all copies of this document that are made.

LoRa Alliance, Inc.
2400 Camino Ramon, Suite 375
San Ramon, CA 94583

Note: All Company, brand and product names may be trademarks that are the sole property of their respective owners.



LoRaWAN™ Specification

Authors:

N. Sornin (Semtech), M. Luis (Semtech), T. Eirich (IBM), T. Kramp (IBM),
O.Hersent (Actility)

Version: V1.0

Date: 2015 January

Status: Released

Important note: This is a candidate specification for the
LoRa™ Alliance protocol named LoRaWAN™.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

| | | |
|----|---|----|
| 1 | Contents | |
| 2 | 1 Introduction | 7 |
| 3 | 1.1 LoRaWAN Classes | 7 |
| 4 | 1.2 Conventions | 8 |
| 5 | 2 Introduction on LoRaWAN options | 9 |
| 6 | 2.1 LoRaWAN Classes | 9 |
| 7 | 2.2 Specification scope | 10 |
| 8 | Class A – All end-devices | 11 |
| 9 | 3 Physical Message Formats | 12 |
| 10 | 3.1 Uplink Messages..... | 12 |
| 11 | 3.2 Downlink Messages | 12 |
| 12 | 3.3 Receive Windows..... | 12 |
| 13 | 3.3.1 First receive window channel, data rate, and start | 13 |
| 14 | 3.3.2 Second receive window channel, data rate, and start | 13 |
| 15 | 3.3.3 Receive window duration..... | 13 |
| 16 | 3.3.4 Receiver activity during the receive windows..... | 13 |
| 17 | 3.3.5 Network sending a message to an end-device | 13 |
| 18 | 3.3.6 Important notice on receive windows | 14 |
| 19 | 3.3.7 Receiving or transmitting other protocols..... | 14 |
| 20 | 4 MAC Message Formats | 15 |
| 21 | 4.1 MAC Layer (PHYPayload)..... | 15 |
| 22 | 4.2 MAC Header (MHDR field)..... | 15 |
| 23 | 4.2.1 Message type (MType bit field)..... | 16 |
| 24 | 4.2.2 Major version of data message (Major bit field) | 16 |
| 25 | 4.3 MAC Payload of Data Messages (MACPayload)..... | 17 |
| 26 | 4.3.1 Frame header (FHDR)..... | 17 |
| 27 | 4.3.2 Port field (FPort) | 20 |
| 28 | 4.3.3 MAC Frame Payload Encryption (FRMPayload) | 20 |
| 29 | 4.4 Message Integrity Code (MIC)..... | 21 |
| 30 | 5 MAC Commands..... | 22 |
| 31 | 5.1 Link Check commands (<i>LinkCheckReq</i> , <i>LinkCheckAns</i>) | 23 |
| 32 | 5.2 Link ADR commands (<i>LinkADRReq</i> , <i>LinkADRAns</i>)..... | 23 |
| 33 | 5.3 End-Device Transmit Duty Cycle (<i>DutyCycleReq</i> , <i>DutyCycleAns</i>)..... | 25 |
| 34 | 5.4 Receive Windows Parameters (<i>RXParamSetupReq</i> , <i>RXParamSetupAns</i>) | 26 |
| 35 | 5.5 End-Device Status (<i>DevStatusReq</i> , <i>DevStatusAns</i>) | 27 |
| 36 | 5.6 Creation / Modification of a Channel (<i>NewChannelReq</i> , <i>NewChannelAns</i>) | 27 |
| 37 | 5.7 Setting delay between TX and RX (<i>RXTimingSetupReq</i> , <i>RXTimingSetupAns</i>) | 28 |
| 38 | 6 End-Device Activation | 30 |
| 39 | 6.1 Data Stored in the End-device after Activation | 30 |
| 40 | 6.1.1 End-device address (DevAddr)..... | 30 |
| 41 | 6.1.2 Application identifier (AppEUI)..... | 30 |
| 42 | 6.1.3 Network session key (NwkSKey) | 30 |
| 43 | 6.1.4 Application session key (AppSKey) | 30 |
| 44 | 6.2 Over-the-Air Activation | 30 |
| 45 | 6.2.1 End-device identifier (DevEUI) | 31 |
| 46 | 6.2.2 Application key (AppKey) | 31 |
| 47 | 6.2.3 Join procedure..... | 31 |
| 48 | 6.2.4 Join-request message | 31 |
| 49 | 6.2.5 Join-accept message..... | 32 |
| 50 | 6.3 Activation by Personalization | 33 |

| | | | |
|----|--------|---|----|
| 1 | 7 | Physical Layer | 34 |
| 2 | 7.1 | EU 863-870MHz ISM Band | 34 |
| 3 | 7.1.1 | EU863-870 Preamble Format..... | 34 |
| 4 | 7.1.2 | EU863-870 ISM Band channel frequencies | 34 |
| 5 | 7.1.3 | EU863-870 Data Rate and End-point Output Power encoding | 35 |
| 6 | 7.1.4 | EU863-870 JoinAccept CFList..... | 35 |
| 7 | 7.1.5 | EU863-870 LinkAdrReq command | 36 |
| 8 | 7.1.6 | EU863-870 Maximum payload size | 36 |
| 9 | 7.1.7 | EU863-870 Receive windows..... | 37 |
| 10 | 7.1.8 | EU863-870 Default Settings | 37 |
| 11 | 7.2 | US 902-928MHz ISM Band | 39 |
| 12 | 7.2.1 | US902-928 Preamble Format..... | 39 |
| 13 | 7.2.2 | US902-928 Channel Frequencies | 39 |
| 14 | 7.2.3 | US902-928 Data Rate and End-point Output Power encoding | 39 |
| 15 | 7.2.4 | US902-928 JoinResp CFList | 40 |
| 16 | 7.2.5 | US902-928 LinkAdrReq command | 40 |
| 17 | 7.2.6 | US902-928 Maximum payload size | 41 |
| 18 | 7.2.7 | US902-928 Receive windows..... | 42 |
| 19 | 7.2.8 | US902-928 Default Settings | 42 |
| 20 | 7.3 | China 779-787MHz ISM Band..... | 43 |
| 21 | 7.3.1 | CN779-787 Preamble Format..... | 43 |
| 22 | 7.3.2 | CN779-787 ISM Band channel frequencies..... | 43 |
| 23 | 7.3.3 | CN779-787 Data Rate and End-point Output Power encoding | 44 |
| 24 | 7.3.4 | CN779-787 JoinAccept CFList | 44 |
| 25 | 7.3.5 | CN779-787 LinkAdrReq command | 45 |
| 26 | 7.3.6 | CN779-787 Maximum payload size | 45 |
| 27 | 7.3.7 | CN779-787 Receive windows..... | 46 |
| 28 | 7.3.8 | CN779-787 Default Settings | 46 |
| 29 | 7.4 | EU 433MHz ISM Band | 47 |
| 30 | 7.4.1 | EU433 Preamble Format..... | 47 |
| 31 | 7.4.2 | EU433 ISM Band channel frequencies | 47 |
| 32 | 7.4.3 | EU433 Data Rate and End-point Output Power encoding | 48 |
| 33 | 7.4.4 | EU433 JoinAccept CFList..... | 48 |
| 34 | 7.4.5 | EU433 LinkAdrReq command | 49 |
| 35 | 7.4.6 | EU433 Maximum payload size | 49 |
| 36 | 7.4.7 | EU433 Receive windows..... | 50 |
| 37 | 7.4.8 | EU433 Default Settings | 50 |
| 38 | | Class B – Beacon | 51 |
| 39 | 8 | Introduction to Class B..... | 52 |
| 40 | 9 | Principle of synchronous network initiated downlink (Class-B option)..... | 53 |
| 41 | 10 | Uplink frame in Class B mode | 55 |
| 42 | 11 | Downlink Ping frame format (Class B option) | 56 |
| 43 | 11.1 | Physical frame format | 56 |
| 44 | 11.2 | Unicast & Multicast MAC messages..... | 56 |
| 45 | 11.2.1 | Unicast MAC message format | 56 |
| 46 | 11.2.2 | Multicast MAC message format..... | 56 |
| 47 | 12 | Beacon acquisition and tracking..... | 57 |
| 48 | 12.1 | Minimal beacon-less operation time | 57 |
| 49 | 12.2 | Extension of beacon-less operation upon reception | 57 |
| 50 | 12.3 | Minimizing timing drift..... | 57 |
| 51 | 13 | Class B Downlink slot timing | 58 |
| 52 | 13.1 | Definitions | 58 |
| 53 | 13.2 | Slot randomization | 59 |

| | | | |
|----|--------|--|----|
| 1 | 14 | Class B MAC commands | 60 |
| 2 | 14.1 | PingSlotInfoReq | 60 |
| 3 | 14.2 | BeaconFreqReq | 61 |
| 4 | 14.3 | PingSlotChannelReq | 61 |
| 5 | 14.4 | BeaconTimingReq | 62 |
| 6 | 14.5 | BeaconTimingAns | 63 |
| 7 | 15 | Beaconing (Class B option) | 64 |
| 8 | 15.1 | Beacon physical layer | 64 |
| 9 | 15.1.1 | EU 863-870MHz ISM Band | 64 |
| 10 | 15.1.2 | US 902-928MHz ISM Band | 64 |
| 11 | 15.2 | Beacon frame content | 65 |
| 12 | 15.3 | Beacon GwSpecific field format | 66 |
| 13 | 15.3.1 | Gateway GPS coordinate:InfoDesc = 0, 1 or 2 | 67 |
| 14 | 15.4 | Beaconing precise timing | 67 |
| 15 | 15.5 | Network downlink route update requirements | 67 |
| 16 | 16 | Class B unicast & multicast downlink channel frequencies | 69 |
| 17 | 16.1 | EU 863-870MHz ISM Band | 69 |
| 18 | 16.2 | US 902-928MHz ISM Band | 69 |
| 19 | | Class C – Continuously listening | 70 |
| 20 | 17 | Class C: Continuously listening end-device | 71 |
| 21 | 17.1 | Second receive window duration for Class C | 71 |
| 22 | 17.2 | Class C Multicast downlinks | 71 |
| 23 | | Support information | 72 |
| 24 | 18 | Examples and Application Information | 73 |
| 25 | 18.1 | Uplink Timing Diagram for Confirmed Data Messages | 73 |
| 26 | 18.2 | The third ACK frame in this example also carries an application payload. A downlink frame can carry any combination of ACK, MAC control commands and payload. Downlink Diagram for Confirmed Data Messages | 73 |
| 27 | 18.3 | Downlink Timing for Frame-Pending Messages | 74 |
| 28 | 18.4 | Data-Rate Adaptation during Message Retransmissions | 76 |
| 29 | 19 | Recommendation on contract to be provided to the network server by the end- device provider at the time of provisioning | 77 |
| 30 | 20 | Recommendation on finding the locally used channels | 78 |
| 31 | 21 | Revisions | 79 |
| 32 | 21.1 | Revision 1.0 | 79 |
| 33 | 22 | Glossary | 80 |
| 34 | 23 | Bibliography | 81 |
| 35 | 23.1 | References | 81 |
| 36 | 24 | NOTICE OF USE AND DISCLOSURE | 82 |
| 37 | | | |
| 38 | | | |
| 39 | | | |
| 40 | | | |

41 Tables

| | | |
|----|--|----|
| 42 | Table 1: MAC message types | 16 |
| 43 | Table 2: Major list | 16 |
| 44 | Table 3: FPort list | 20 |
| 45 | Table 4: MAC commands | 22 |
| 46 | Table 5: Channel state table | 24 |
| 47 | Table 6: LinkADRAAns status bits signification | 25 |
| 48 | Table 7: RX2SetupAns status bits signification | 26 |
| 49 | Table 8: Battery level decoding | 27 |
| 50 | Table 9: NewChannelAns status bits signification | 28 |

| | | |
|----|---|----|
| 1 | Table 10: Del mapping table | 29 |
| 2 | Table 11: EU863-870 synch words | 34 |
| 3 | Table 12: EU863-870 default channels | 34 |
| 4 | Table 13: EU863-870 JoinReq Channel List | 35 |
| 5 | Table 14: Data rate and TX power table..... | 35 |
| 6 | Table 15: ChMaskCntl value table..... | 36 |
| 7 | Table 16: EU863-870 maximum payload size | 37 |
| 8 | Table 17 : EU863-870 maximum payload size (not repeater compatible)..... | 37 |
| 9 | Table 18: Data rate and TX power table (Rem: DR4 is identical to DR12, DR8..13 must be implemented in end-devices and are reserved for future applications) | 40 |
| 10 | Table 19: ChMaskCntl value table..... | 40 |
| 11 | Table 20: US902-928 maximum payload size (repeater compatible)..... | 41 |
| 12 | Table 21 : US902-928 maximum payload size (not repeater compatible)..... | 41 |
| 13 | Table 22: Data rate mapping..... | 42 |
| 14 | Table 23: CN779-787 synch words | 43 |
| 15 | Table 24: CN780 JoinReq Channel List | 43 |
| 16 | Table 25: Data rate and TX power table..... | 44 |
| 17 | Table 26: ChMaskCntl value table..... | 45 |
| 18 | Table 27: CN780 maximum payload size | 45 |
| 19 | Table 28 : CN780 maximum payload size (not repeater compatible)..... | 46 |
| 20 | Table 29: EU433 synch words | 47 |
| 21 | Table 30: EU433 JoinReq Channel List | 47 |
| 22 | Table 31: Data rate and TX power table..... | 48 |
| 23 | Table 32: ChMaskCntl value table..... | 49 |
| 24 | Table 33: EU433 maximum payload size | 49 |
| 25 | Table 34 : EU433 maximum payload size (not repeater compatible)..... | 50 |
| 26 | Table 35: Beacon timing | 58 |
| 27 | | |
| 28 | | |

29 Figures

| | | |
|----|---|----|
| 30 | Figure 1: LoRaWAN Classes | 9 |
| 31 | Figure 2: Uplink PHY structure..... | 12 |
| 32 | Figure 3: Downlink PHY structure | 12 |
| 33 | Figure 4: End-device receive slot timing..... | 13 |
| 34 | Figure 5: Radio PHY structure (CRC* is only available on uplink messages) | 15 |
| 35 | Figure 6: PHY payload structure | 15 |
| 36 | Figure 7: MAC payload structure..... | 15 |
| 37 | Figure 8: Frame header structure..... | 15 |
| 38 | Figure 9: LoRa message format elements..... | 15 |
| 39 | Figure 10: US902-928 channel frequencies | 39 |
| 40 | Figure 11: Beacon reception slot and ping slots | 54 |
| 41 | Figure 12 : beacon-less temporary operation | 57 |
| 42 | Figure 13: Beacon timing | 58 |
| 43 | Figure 14: Class C end-device receive slot timing..... | 71 |
| 44 | Figure 15: Uplink timing diagram for confirmed data messages | 73 |
| 45 | Figure 16: Downlink timing diagram for confirmed data messages..... | 74 |
| 46 | Figure 17: Downlink timing diagram for frame-pending messages, example 1 | 75 |
| 47 | Figure 18: Downlink timing diagram for frame-pending messages, example 2 | 75 |
| 48 | Figure 19: Downlink timing diagram for frame-pending messages, example 3 | 75 |
| 49 | | |

1 Introduction

This document describes the LoRaWAN™ network protocol which is optimized for battery-powered end-devices that may be either mobile or mounted at a fixed location.

LoRaWAN networks typically are laid out in a star-of-stars topology in which **gateways**¹ relay messages between **end-devices**² and a central **network server** at the backend. Gateways are connected to the network server via standard IP connections while end-devices use single-hop LoRa™ or FSK communication to one or many gateways.³ All communication is generally bi-directional, although uplink communication from an end-device to the network server is expected to be the predominant traffic.

Communication between end-devices and gateways is spread out on different **frequency channels** and **data rates**. The selection of the data rate is a trade-off between communication range and message duration, communications with different data rates do not interfere with each other. LoRa data rates range from 0.3 kbps to 50 kbps. To maximize both battery life of the end-devices and overall network capacity, the LoRa network infrastructure can manage the data rate and RF output for each end-device individually by means of an **adaptive data rate** (ADR) scheme.

End-devices may transmit on any channel available at any time, using any available data rate, as long as the following rules are respected:

- The end-device changes channel in a pseudo-random fashion for every transmission. The resulting frequency diversity makes the system more robust to interferences.
- The end-device respects the maximum transmit duty cycle relative to the sub-band used and local regulations.
- The end-device respects the maximum transmit duration (or dwell time) relative to the sub-band used and local regulations.

Note: Maximum transmit duty-cycle and dwell time per sub-band are region specific and are defined in the Chapter 6.

1.1 LoRaWAN Classes

All LoRaWAN devices implement at least the Class A functionality as described in this document. In addition they may implement options named Class B, Class C as also described in this document or others to be defined. In all cases, they must remain compatible with Class A.

¹ Gateways are also known as **concentrators** or **base stations**.

² End-devices are also known as **nodes**.

³ Support for intermediate elements – repeaters – is not described in the document, however payload restrictions for encapsulation overhead are included in this specification. A repeater is defined as using LoRaWAN as its backhaul mechanism.

1 1.2 Conventions

2 MAC commands are written *LinkCheckReq*, bits and bit fields are written **FRMPayload**,
3 constants are written RECEIVE_DELAY1, variables are written *N*.

4 In this document,

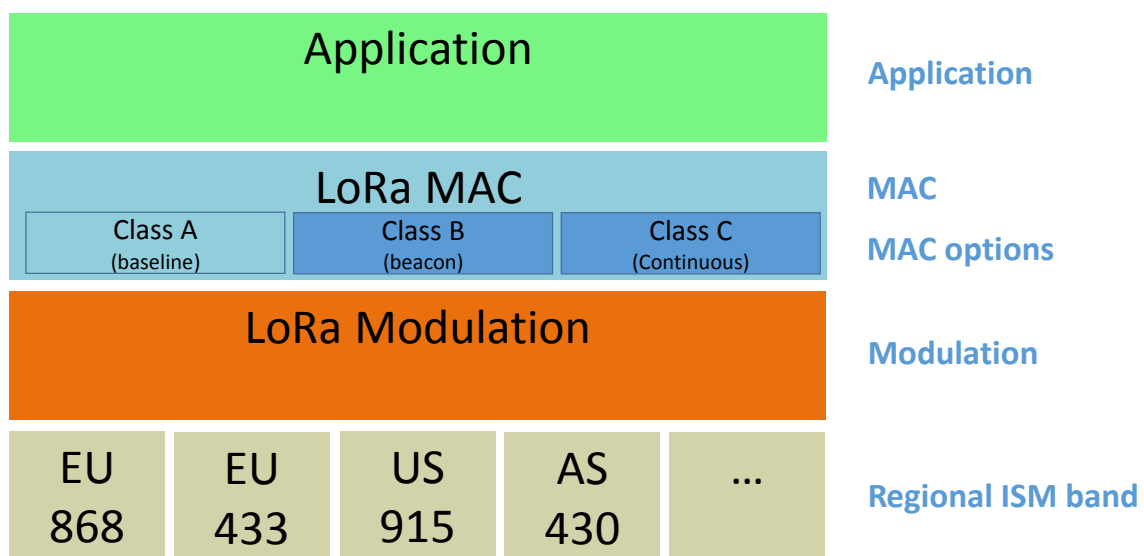
- 5 • The octet order for all multi-octet fields is little endian and
- 6 • EUI are 64 bits integers and are transmitted as little endian.

2 Introduction on LoRaWAN options

LoRa™ is a wireless modulation for long-range low-power low-data-rate applications developed by Semtech. Devices implementing more than Class A are generally named “higher Class end-devices” in this document.

2.1 LoRaWAN Classes

A LoRa network distinguishes between a basic LoRaWAN (named Class A) and optional features (Class B, Class C ...):



8
9 **Figure 1: LoRaWAN Classes**

- 10 • **Bi-directional end-devices (Class A):** End-devices of Class A allow for bi-
11 directional communications whereby each end-device’s uplink transmission is
12 followed by two short downlink receive windows. The transmission slot scheduled by
13 the end-device is based on its own communication needs with a small variation
14 based on a random time basis (ALOHA-type of protocol). This Class A operation is
15 the lowest power end-device system for applications that only require downlink
16 communication from the server shortly after the end-device has sent an uplink
17 transmission. Downlink communications from the server at any other time will have to
18 wait until the next scheduled uplink.
- 19 • **Bi-directional end-devices with scheduled receive slots (Class B):** End-devices
20 of Class B allow for more receive slots. In addition to the Class A random receive
21 windows, Class B devices open extra receive windows at scheduled times. In order
22 for the End-device to open it receive window at the scheduled time it receives a time
23 synchronized Beacon from the gateway. This allows the server to know when the
24 end-device is listening.
- 25 • **Bi-directional end-devices with maximal receive slots (Class C):** End-devices of
26 Class C have nearly continuously open receive windows, only closed when
27 transmitting. Class C end-device will use more power to operate than Class A or
28 Class B but they offer the lowest latency for server to end-device communication.

1 **2.2 Specification scope**

2 This LoRaWAN specification describes the additional functions differentiating an end-device
3 higher Class from one of Class A. A higher Class end-device shall also implement all the
4 functionality described in the LoRaWAN Class A specification.

5 **NOTE:** Physical message format, MAC message format, and other
6 parts of this specification that are common to both end-devices of
7 Class A and higher Classes are described only in the LoRaWAN
8 Class A specification to avoid redundancy.

1 **CLASS A – ALL END-DEVICES**

- 2 All LoRaWAN end-devices must implement Class A features.

1 **3 Physical Message Formats**

2 The LoRa terminology distinguishes between uplink and downlink messages.

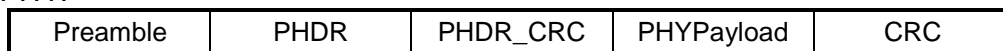
3 **3.1 Uplink Messages**

4 **Uplink messages** are sent by end-devices to the network server relayed by one or many
5 gateways.

6 Uplink messages use the LoRa radio packet explicit mode in which the LoRa physical
7 header (**PHDR**) plus a header CRC (**PHDR_CRC**) are included.¹ The integrity of the payload
8 is protected by a CRC.

9 The **PHDR**, **PHDR_CRC** and payload **CRC** fields are inserted by the radio transceiver.

10 *Uplink PHY:*



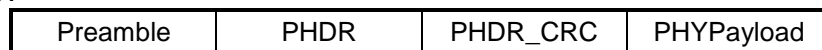
11 **Figure 2: Uplink PHY structure**

12 **3.2 Downlink Messages**

13 Each **downlink message** is sent by the network server to only one end-device and is
14 relayed by a single gateway.²

15 Downlink messages use the radio packet explicit mode in which the LoRa physical header
16 (**PHDR**) and a header CRC (**PHDR_CRC**) are included.³

17 *Downlink PHY:*



18 **Figure 3: Downlink PHY structure**

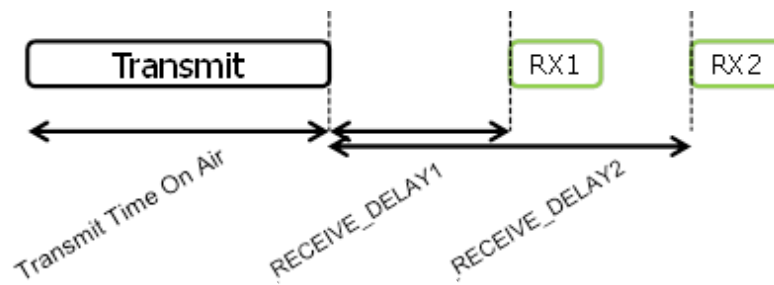
19 **3.3 Receive Windows**

20 Following each uplink transmission the end-device opens two short receive windows. The
21 receive window start times is a configured periods are the end of the transmission of the last
22 uplink bit.

¹ See the LoRa radio transceiver datasheet for a description of LoRa radio packet implicit/explicit modes.

² This specification does not describe the transmission of multicast messages from a network server to many end-devices.

³ No payload integrity check is done at this level to keep messages as short as possible with minimum impact on any duty-cycle limitations of the ISM bands used.



1
2 **Figure 4: End-device receive slot timing.**

3 **3.3.1 First receive window channel, data rate, and start**

4 The first receive window RX1 uses the same frequency channel as the uplink and a data
5 rate that is a function of the data rate used for the uplink. RX1 opens RECEIVE_DELAY1¹
6 seconds (+/- 20 microseconds) after the end of the uplink modulation. The relationship
7 between uplink and RX1 slot downlink data rate is region specific and detailed in the Section
8 7. By default the first receive window datarate is identical to the datarate of the last uplink.

9 **3.3.2 Second receive window channel, data rate, and start**

10 The second receive window RX2 uses a fixed configurable frequency and data rate and
11 opens RECEIVE_DELAY2¹ seconds (+/- 20 microseconds) after the end of the uplink
12 modulation. The frequency and data rate used can be modified through MAC commands
13 (see Section 5). The default frequency and data rate to use are region specific and detailed
14 in the Section 7 .

15 **3.3.3 Receive window duration**

16 The length of a receive window must be at least the time required by the end-device's radio
17 transceiver to effectively detect a downlink preamble.

18 **3.3.4 Receiver activity during the receive windows**

19 If a preamble is detected during one of the receive windows, the radio receiver stays active
20 until the downlink frame is demodulated. If a frame was detected and subsequently
21 demodulated during the first receive window and the frame was intended for this end-device
22 after address and MIC (message integrity code) checks, the end-device does not open the
23 second receive window.

24 **3.3.5 Network sending a message to an end-device**

25 If the network intends to transmit a downlink to an end-device, it will always initiate the
26 transmission precisely at the beginning of one of those two receive windows.

¹ RECEIVE_DELAY1 and RECEIVE_DELAY2 are described in Chapter 6.

1 **3.3.6 Important notice on receive windows**

2 An end-device shall not transmit another uplink message before it either has received a
3 downlink message in the first or second receive window of the previous transmission, or the
4 second receive window of the previous transmission is expired.

5 **3.3.7 Receiving or transmitting other protocols**

6 The node may listen or transmit other protocols or do any transactions between the
7 LoRaWAN transmission and reception windows, as long as the end-device remains
8 compatible with the local regulation and compliant with the LoRaWAN specification.

4 MAC Message Formats

All LoRa uplink and downlink messages carry a PHY payload (**Payload**) starting with a single-octet MAC header (**MHDR**), followed by a MAC payload (**MACPayload**)¹, and ending with a 4-octet message integrity code (**MIC**).

Radio PHY layer:

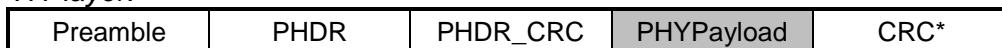


Figure 5: Radio PHY structure (CRC* is only available on uplink messages)

PHYPayload:

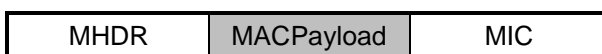


Figure 6: PHY payload structure

MACPayload:

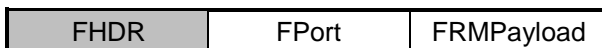


Figure 7: MAC payload structure

FHDR:



Figure 8: Frame header structure

Figure 9: LoRa message format elements

4.1 MAC Layer (PHYPayload)

| | | | |
|--------------|------|------------|-----|
| Size (bytes) | 1 | 1..M | 4 |
| PHYPayload | MHDR | MACPayload | MIC |

The maximum length (*M*) of the **MACPayload** field is region specific and is specified in Chapter 6.

4.2 MAC Header (MHDR field)

| | | | |
|-----------|-------|------|-------|
| Bit# | 7..5 | 4..2 | 1..0 |
| MHDR bits | MType | RFU | Major |

The MAC header specifies the message type (**MType**) and according to which major version (**Major**) of the frame format of the LoRaWAN layer specification the frame has been encoded.

¹ Maximum payload size is detailed in the Chapter 6.

1 4.2.1 Message type (MType bit field)

2 The LoRaWAN distinguishes between six different MAC message types: **join request**, **join**
 3 **accept**, **unconfirmed data up/down**, and **confirmed data up/down**.

| MType | Description |
|-------|-----------------------|
| 000 | Join Request |
| 001 | Join Accept |
| 010 | Unconfirmed Data Up |
| 011 | Unconfirmed Data Down |
| 100 | Confirmed Data Up |
| 101 | Confirmed Data Down |
| 110 | RFU |
| 111 | Proprietary |

4 **Table 1: MAC message types**

5 4.2.1.1 Join-request and join-accept messages

6 The join-request and join-accept messages are used by the over-the-air activation procedure
 7 described in Chapter 6.2.

8 4.2.1.2 Data messages

9 Data messages are used to transfer both MAC commands and application data, which can
 10 be combined together in a single message. A **confirmed-data message** has to be
 11 acknowledged by the receiver, whereas an **unconfirmed-data message** does not require
 12 an acknowledgment.¹ **Proprietary messages** can be used to implement non-standard
 13 message formats that are not interoperable with standard messages but must only be used
 14 among devices that have a common understanding of the proprietary extensions.

15 Message integrity is ensured in different ways for different message types and is described
 16 per message type below.

17 4.2.2 Major version of data message (Major bit field)

| Major bits | Description |
|------------|-------------|
| 00 | LoRaWAN R1 |
| 01..11 | RFU |

19 **Table 2: Major list**

20 **Note:** The Major version specifies the format of the messages
 21 exchanged in the join procedure (see Chapter 6.2) and the first four
 22 bytes of the MAC Payload as described in Chapter 4. For each major
 23 version, end-devices may implement different minor versions of the
 24 frame format. The minor version used by an end-device must be made
 25 known to the network server beforehand using out of band messages
 26 (e.g., as part of the device personalization information).

¹ A detailed timing diagram of the acknowledge mechanism is given in Section 18.

1 4.3 MAC Payload of Data Messages (MACPayload)

2 The MAC payload of the data messages, a so-called “data frame”, contains a frame header
 3 (FHDR) followed by an optional port field (FPort) and an optional frame payload field
 4 (FRMPayload).

5 4.3.1 Frame header (FHDR)

6 The FHDR contains the short device address of the end-device (DevAddr), a frame control
 7 octet (FCtrl), a 2-octets frame counter (FCnt), and up to 15 octets of frame options (FOpts)
 8 used to transport MAC commands.
 9

| Size (bytes) | 4 | 1 | 2 | 0..15 |
|--------------|---------|-------|------|-------|
| FHDR | DevAddr | FCtrl | FCnt | FOpts |

10 For downlink frames the FCtrl content of the frame header is:

| Bit# | 7 | 6 | 5 | 4 | [3..0] |
|------------|-----|-----------|-----|----------|----------|
| FCtrl bits | ADR | ADRACKReq | ACK | FPending | FOptsLen |

11 For uplink frames the FCtrl content of the frame header is:

| Bit# | 7 | 6 | 5 | 4 | [3..0] |
|------------|-----|-----------|-----|-----|----------|
| FCtrl bits | ADR | ADRACKReq | ACK | RFU | FOptsLen |

12 4.3.1.1 Adaptive data rate control in frame header (ADR, ADRACKReq in FCtrl)

13 LoRa network allows the end-devices to individually use any of the possible data rates. This
 14 feature is used by the LoRaWAN to adapt and optimize the data rate of static end-devices.
 15 This is referred to as Adaptive Data Rate (ADR) and when this is enabled the network will be
 16 optimized to use the fastest data rate possible.

17 Mobile end-devices should use their fixed default data rate as data rate management is not
 18 practical when the moving end-device causes fast changes in the radio environment.

19 If the **ADR** bit is set, the network will control the data rate of the end-device through the
 20 appropriate MAC commands. If the **ADR** bit is not set, the network will not attempt to control
 21 the data rate of the end-device regardless of the received signal quality. The **ADR** bit may
 22 be set and unset by the end-device or the Network on demand. However, whenever
 23 possible, the ADR scheme should be enabled to increase the battery life of the end-device
 24 and maximize the network capacity.

25 **Note:** Even mobile end-devices are actually immobile most of the time.
 26 So depending on its state of mobility, an end-device can request the
 27 network to optimize its data rate using ADR.

28 If an end-device whose data rate is optimized by the network to use a data rate higher than
 29 its default data rate, it periodically needs to validate that the network still receives the uplink
 30 frames. Each time the uplink frame counter is incremented (for each new uplink, repeated
 31 transmissions do not increase the counter), the device increments an ADR_ACK_CNT
 32 counter. After ADR_ACK_LIMIT uplinks (ADR_ACK_CNT >= ADR_ACK_LIMIT) without any
 33 downlink response, it sets the ADR acknowledgment request bit (**ADRACKReq**). The
 34 network is required to respond with a downlink frame within the time set by the
 35 ADR_ACK_DELAY, any received downlink frame following an uplink frame resets the
 36 ADR_ACK_CNT counter. The downlink **ACK** bit does not need to be set as any response
 37 during the receive slot of the end-device indicates that the gateway has still received the
 38 uplinks from this device. If no reply is received within the next ADR_ACK_DELAY uplinks

1 (i.e., after a total of `ADR_ACK_LIMIT + ADR_ACK_DELAY`), the end-device may try to
2 regain connectivity by switching to the next lower data rate that provides a longer radio
3 range. The end-device will further lower its data rate step by step every time
4 `ADR_ACK_LIMIT` is reached. The **ADRACKReq** shall not be set if the device uses its
5 default data rate because in that case no action can be taken to improve the link range.

6 **Note:** Not requesting an immediate response to an ADR
7 acknowledgement request provides flexibility to the network to
8 optimally schedule its downlinks.
9

10 **Note:** In uplink transmissions the **ADRACKReq** bit is set if
11 `ADR_ACK_CNT >= ADR_ACK_LIMIT` and the current data-rate is
12 greater than the device defined minimum data rate, it is cleared in
13 other conditions.

14 4.3.1.2 Message acknowledge bit and acknowledgement procedure (ACK in FCtrl)

15 When receiving a *confirmed data* message, the receiver shall respond with a data frame that
16 has the acknowledgment bit (**ACK**) set. If the sender is an end-device, the network will send
17 the acknowledgement using one of the receive windows opened by the end-device after the
18 send operation. If the sender is a gateway, the end-device transmits an acknowledgment at
19 its own discretion.

20 Acknowledgements are only sent in response to the latest message received and are never
21 retransmitted.

22

23 **Note:** To allow the end-devices to be as simple as possible and have
24 as few states as possible it may transmit an explicit (possibly empty)
25 acknowledgement data message immediately after the reception of a
26 data message requiring a confirmation. Alternatively the end-device
27 may defer the transmission of an acknowledgement to piggyback it
28 with its next data message.

29 4.3.1.3 Retransmission procedure

30 The number of retransmissions (and their timing) for the same message where an
31 acknowledgment is requested but not received is at the discretion of the end device and
32 may be different for each end-device, it can also be set or adjusted from the network server.

33 **Note:** Some example timing diagrams of the acknowledge mechanism
34 are given in Chapter 18.

35

36 **Note:** If an end-device has reached its maximum number of
37 retransmissions without receiving an acknowledgment, it can try to re-
38 gain connectivity by moving to a lower data rate with longer reach. It is
39 up to the end-device to retransmit the message again or to forfeit that
40 message and move on.

41

42 **Note:** If the network server has reached its maximum number of
43 retransmissions without receiving an acknowledgment, it will generally
44 consider the end-device as unreachable until it receives further

1 messages from the end-device. It is up to the network server to
2 retransmit the message once connectivity to the end-device in question
3 is regained or to forfeit that message and move on.

4
5 **Note:** The recommended data rate back-off strategy during re-
6 transmissions is described in Chapter 18.4

7 **4.3.1.4 Frame pending bit (FPending in FCtrl, downlink only)**

8 The frame pending bit (**FPending**) is only used in downlink communication, indicating that
9 the gateway has more data pending to be sent and therefore asking the end-device to open
10 another receive window as soon as possible by sending another uplink message.

11 The exact use of **FPending** bit is described in Chapter 18.3.

12 **4.3.1.5 Frame counter (FCnt)**

13 Each end-device has two frame counters to keep track of the number of data frames sent
14 uplink to the network server (FCntUp), incremented by the end-device and received by the
15 end-device downlink from the network server (FCntDown), which is incremented by the
16 network server. The network server tracks the uplink frame counter and generates the
17 downlink counter for each end-device. After a join accept, the frame counters on the end-
18 device and the frame counters on the network server for that end-device are reset to 0.
19 Subsequently FCntUp and FCntDown are incremented at the sender side by 1 for each data
20 frame sent in the respective direction. At the receiver side, the corresponding counter is
21 kept in sync with the value received provided the value received has incremented compared
22 to the current counter value and is less than the value specified by MAX_FCNT_GAP¹ after
23 considering counter rollovers. If this difference is greater than the value of MAX_FCNT_GAP
24 then too many data frames have been lost then subsequent will be discarded.

25 The LoRaWAN allows the use of either 16-bits or 32-bits frame counters. The network side
26 needs to be informed out-of-band about the width of the frame counter implemented in a
27 given end-device. If a 16-bits frame counter is used, the **FCnt** field can be used directly as
28 the counter value, possibly extended by leading zero octets if required. If a 32-bits frame
29 counter is used, the **FCnt** field corresponds to the least-significant 16 bits of the 32-bits
30 frame counter (i.e., FCntUp for data frames sent uplink and FCntDown for data frames sent
31 downlink).

32 The end-device shall not reuse the same FCntUp value, except for retransmission, with the
33 same application and network session keys.

34 **Note:** Since the **FCnt** field carries only the least-significant 16 bits of
35 the 32-bits frame counter, the server must infer the 16 most-significant
36 bits of the frame counter from the observation of the traffic.

37 **4.3.1.6 Frame options (FOptsLen in FCtrl, FOpts)**

38 The frame-options length field (**FOptsLen**) in **FCtrl** byte denotes the actual length of the
39 frame options field (**FOpts**) included in the frame.

¹ Actual value for MAX_FCNT_GAP, RECEIVE_DELAY1 and RECEIVE_DELAY2 can be found at 7.1.7 for EU863-870 or 7.2.7 for US902-928.

1 **FOpts** transport MAC commands of a maximum length of 15 octets that are piggybacked
2 onto data frames; see Chapter 4.4 for a list of valid MAC commands.

3 If **FOptsLen** is 0, the **FOpts** field is absent. If **FOptsLen** is different from 0, i.e. if MAC
4 commands are present in the **FOpts** field, the port 0 cannot be used (**FPort** must be either
5 not present or different from 0).

6 MAC commands cannot be simultaneously present in the payload field and the frame
7 options field.

8 4.3.2 Port field (FPort)

9 If the frame payload field is not empty, the port field must be present. If present, an **FPort**
10 value of 0 indicates that the **FRMPayload** contains MAC commands only; see Chapter 4.4
11 for a list of valid MAC commands. **FPort** values 1..223 (0x01..0xDF) are application-
12 specific. **FPort** values 224..255 (0xE0..0xFF) are reserved for future standardized
13 application extensions.

14

| Size (bytes) | 7..23 | 0..1 | 0..N |
|-------------------|-------|-------|------------|
| MACPayload | FHDR | FPort | FRMPayload |

15 N is the number of octets of the application payload. The valid range for N is region specific
16 and is defined in Section 7

17 N should be equal or smaller than:

$$18 \quad N \leq M - 1 - (\text{length of FHDR in octets})$$

19 where M is the maximum MAC payload length.

20 4.3.3 MAC Frame Payload Encryption (FRMPayload)

21 If a data frame carries a payload, **FRMPayload** must be encrypted before the message
22 integrity code (**MIC**) is calculated.

23 The encryption scheme used is based on the generic algorithm described in IEEE
24 802.15.4/2006 Annex B [IEEE802154] using AES with a key length of 128 bits.

25 As a default, the encryption/decryption is done by the LoRaWAN layer for all FPort. The
26 encryption/decryption may be done above the LoRaWAN layer for specific FPorts except 0,
27 if this is more convenient for the application. The information concerning which FPort from
28 which node is encrypted/decrypted outside of the LoRaWAN layer must be communicated to
29 the server using an out-of-band channel (see Section 19).

30 4.3.3.1 Encryption in LoRaWAN

31 The key K used depends on the FPort of the data message:

32

| FPort | K |
|--------|---------|
| 0 | NwkSKey |
| 1..255 | AppSKey |

33 **Table 3: FPort list**

34 The fields encrypted are:

$$35 \quad pld = \text{FRMPayload}$$

36 For each data message, the algorithm defines a sequence of Blocks A_i for $i = 1..k$ with $k =$
37 $\text{ceil}(\text{len}(pld) / 16)$:

1

| Size (bytes) | 1 | 4 | 1 | 4 | 4 | 1 | 1 |
|--------------|------|----------|-----|---------|--------------------|------|-----|
| A_i | 0x01 | 4 x 0x00 | Dir | DevAddr | FCntUp or FCntDown | 0x00 | i |

2

 3 The direction field (**Dir**) is 0 for uplink frames and 1 for downlink frames.

 4 The blocks A_i are encrypted to get a sequence S of blocks S_i :

5

 6 $S_i = \text{aes128_encrypt}(K, A_i)$ for $i = 1..k$

 7 $S = S_1 | S_2 | .. | S_k$

8 Encryption and decryption of the payload is done by truncating

9

 10 $(pld | \text{pad}_{16}) \text{ xor } S$

 11 to the first $\text{len}(pld)$ octets.

12 4.3.3.2 Encryption above the LoRaWAN layer

 13 If the layers above LoRaWAN provide pre-encrypted FRMPayload to LoRaWAN on selected
 14 ports (but not on FPort 0 which is reserved for MAC commands), LoRaWAN transfers
 15 FRMPayload from MACPayload to the application and FRMPayload from the application to
 16 MACPayload without any modification of FRMPayload.

17

18 4.4 Message Integrity Code (MIC)

 19 The message integrity code (**MIC**) is calculated over all the fields in the message.

20

 21 $msg = \text{MHDR} | \text{FHDR} | \text{FPort} | \text{FRMPayload}$

 22 whereby $\text{len}(msg)$ denotes the length of the message in octets.

 23 The **MIC** is calculated as follows [RFC4493]:

24

 25 $cmac = \text{aes128_cmac}(\text{NwkSKey}, B_0 | msg)$

 26 $\text{MIC} = cmac[0..3]$

27

 28 whereby the block B_0 is defined as follows:

| Size (bytes) | 1 | 4 | 1 | 4 | 4 | 1 | 1 |
|--------------|------|----------|-----|---------|--------------------|------|-------------------|
| B_0 | 0x49 | 4 x 0x00 | Dir | DevAddr | FCntUp or FCntDown | 0x00 | $\text{len}(msg)$ |

29

 30 The direction field (**Dir**) is 0 for uplink frames and 1 for downlink frames.

5 MAC Commands

For network administration, a set of MAC commands may be exchanged exclusively between the network server and the MAC layer on an end-device. MAC layer commands are never visible to the application or the application server or the application running on the end-device.

A single data frame can contain any sequence of MAC commands, either piggybacked in the **FOpts** field or, when sent as a separate data frame, in the **FRMPayload** field with the **FPort** field being set to 0. Piggybacked MAC commands are always sent without encryption and must not exceed 15 octets. MAC commands sent as **FRMPayload** are always encrypted and must not exceed the maximum **FRMPayload** length.

Note: MAC commands whose content shall not be disclosed to an eavesdropper must be sent in the **FRMPayload** of a separate data message.

A MAC command consists of a command identifier (**CID**) of 1 octet followed by a possibly empty command-specific sequence of octets.

| CID | Command | Transmitted by | | Short Description |
|--------------|-------------------------|----------------|---------|--|
| | | End-device | Gateway | |
| 0x02 | <i>LinkCheckReq</i> | x | | Used by an end-device to validate its connectivity to a network. |
| 0x02 | <i>LinkCheckAns</i> | | x | Answer to LinkCheckReq command. Contains the received signal power estimation indicating to the end-device the quality of reception (link margin). |
| 0x03 | <i>LinkADRReq</i> | | x | Requests the end-device to change data rate, transmit power, repetition rate or channel. |
| 0x03 | <i>LinkADRAns</i> | x | | Acknowledges the LinkRateReq. |
| 0x04 | <i>DutyCycleReq</i> | | x | Sets the maximum aggregated transmit duty-cycle of a device |
| 0x04 | <i>DutyCycleAns</i> | x | | Acknowledges a DutyCycleReq command |
| 0x05 | <i>RXParamSetupReq</i> | | x | Sets the reception slots parameters |
| 0x05 | <i>RXParamSetupAns</i> | x | | Acknowledges a RXSetupReq command |
| 0x06 | <i>DevStatusReq</i> | | x | Requests the status of the end-device |
| 0x06 | <i>DevStatusAns</i> | x | | Returns the status of the end-device, namely its battery level and its demodulation margin |
| 0x07 | <i>NewChannelReq</i> | | x | Creates or modifies the definition of a radio channel |
| 0x07 | <i>NewChannelAns</i> | x | | Acknowledges a NewChannelReq command |
| 0x08 | <i>RXTimingSetupReq</i> | | x | Sets the timing of the of the reception slots |
| 0x08 | <i>RXTimingSetupAns</i> | x | | Acknowledge RXTimingSetupReq command |
| 0x80 to 0xFF | Proprietary | x | x | Reserved for proprietary network command extensions |

Table 4: MAC commands

Note: The length of a MAC command is not explicitly given and must be implicitly known by the MAC implementation. Therefore unknown MAC commands cannot be skipped and the first unknown MAC command terminates the processing of the MAC command sequence. It is therefore advisable to order MAC commands according to the

1 version of the LoRaWAN specification which has introduced a MAC
 2 command for the first time. This way all MAC commands up to the
 3 version of the LoRaWAN specification implemented can be processed
 4 even in the presence of MAC commands specified only in a version of
 5 the LoRaWAN specification newer than that implemented.

6
 7 **Note:** Any values adjusted by the network server (e.g., RX2, new or
 8 adjusted channels definitions) remain only valid until the next join of
 9 the end-device. Therefore after each successful join procedure the
 10 end-device uses the default parameters again and it is up to the
 11 network server to re-adjust the values again as needed.

12 5.1 Link Check commands (*LinkCheckReq*, *LinkCheckAns*)

13
 14 With the **LinkCheckReq** command, an end-device may validate its connectivity with the
 15 network. The command has no payload.

16 When a **LinkCheckReq** is received by the network server via one or multiple gateways, it
 17 responds with a **LinkCheckAns** command.

| Size (bytes) | 1 | 1 |
|-----------------------------|--------|-------|
| LinkCheckAns Payload | Margin | GwCnt |

18
 19 The demodulation margin (**Margin**) is an 8-bit unsigned integer in the range of 0..254
 20 indicating the link margin in dB of the last successfully received **LinkCheckReq** command.
 21 A value of “0” means that the frame was received at the demodulation floor (0 dB or no
 22 margin) while a value of “20”, for example, means that the frame reached the gateway 20 dB
 23 above the demodulation floor. Value “255” is reserved.

24 The gateway count (**GwCnt**) is the number of gateways that successfully received the last
 25 **LinkCheckReq** command.

26 5.2 Link ADR commands (*LinkADRReq*, *LinkADRAAns*)

27
 28 With the **LinkADRReq** command, the network server requests an end-device to perform a
 29 rate adaptation.

| Size (bytes) | 1 | 2 | 1 |
|---------------------------|------------------|--------|------------|
| LinkADRReq Payload | DataRate_TXPower | ChMask | Redundancy |

| Bits | [7:4] | [3:0] |
|-------------------------|----------|---------|
| DataRate_TXPower | DataRate | TXPower |

30
 31
 32
 33 The requested data rate (**DataRate**) and TX output power (**TXPower**) are region-specific
 34 and are encoded as indicated in Chapter 7.

1 The channel mask (**ChMask**) encodes the channels usable for uplink access as follows with
 2 bit 0 corresponding to the LSB:

| Bit# | Usable channels |
|------|-----------------|
| 0 | Channel 1 |
| 1 | Channel 2 |
| .. | .. |
| 15 | Channel 16 |

3 **Table 5: Channel state table**

4 A bit in the **ChMask** field set to 1 means that the corresponding channel can be used for
 5 uplink transmissions if this channel allows the data rate currently used by the end-device. A
 6 bit set to 0 means the corresponding channels should be avoided.

| Bits | 7 | [6:4] | [3:0] |
|------------------------|-----|------------|-------|
| Redundancy bits | RFU | ChMaskCntl | NbRep |

8 In the Redundancy bits the number of repetitions (**NbRep**) field is the number of repetition
 9 for each uplink message. This applies only to “unconfirmed” uplink frames. The default value
 10 is 1. The valid range is [1:15]. If **NbRep**==0 is received the end-device should use the
 11 default value. This field can be used by the network manager to control the redundancy of
 12 the node uplinks to obtain a given Quality of Service. The end-device performs frequency
 13 hopping as usual between repeated transmissions, it does wait after each repetition until the
 14 receive windows have expired.

15 The channel mask control (**ChMaskCntl**) field controls the interpretation of the previously
 16 defined **ChMask** bit mask. This field will only be non-zero values in networks where more
 17 than 16 channels are implemented. It controls the block of 16 channels to which the
 18 **ChMask** applies. It can also be used to globally turn on or off all channels using specific
 19 modulation. This field usage is region specific and is defined in Chapter 7.

20 The channel frequencies are region-specific and they are defined in Chapter 6. An end-
 21 device answers to a **LinkADRReq** with a **LinkADRAns** command.

| Size (bytes) | 1 |
|---------------------------|--------|
| LinkADRAns Payload | Status |

23
 24

| Bits | [7:3] | 2 | 1 | 0 |
|--------------------|-------|-----------|---------------|---------------------|
| Status bits | RFU | Power ACK | Data rate ACK | Channel mask ACK |

25

1 The *LinkADRAns Status* bits have the following meaning:

2

| | Bit = 0 | Bit = 1 |
|-------------------------|---|--|
| Channel mask ACK | The channel mask sent enables a yet undefined channel. The command was discarded and the end-device state was not changed. | The channel mask sent was successfully interpreted. All currently defined channel states were set according to the mask. |
| Data rate ACK | The data rate requested is unknown to the end-device or is not possible given the channel mask provided (not supported by any of the enabled channels). The command was discarded and the end-device state was not changed. | The data rate was successfully set. |
| Power ACK | The requested power level is not implemented in the device. The command was discarded and the end-device state was not changed. | The power level was successfully set. |

3 **Table 6: LinkADRAns status bits signification**

4 If any of those three bits equals 0, the command did not succeed and the node has kept the
5 previous state.

6 **5.3 End-Device Transmit Duty Cycle (*DutyCycleReq*, *DutyCycleAns*)**

7 The *DutyCycleReq* command is used by the network coordinator to limit the maximum
8 aggregated transmit duty cycle of an end-device. The aggregated transmit duty cycle
9 corresponds to the transmit duty cycle over all sub-bands.

10

| Size (bytes) | 1 |
|-----------------------------|-----------|
| DutyCycleReq Payload | MaxDCycle |

11

12 The maximum end-device transmit duty cycle allowed is:

$$aggregated\ duty\ cycle = \frac{1}{2^{MaxDCycle}}$$

13 The valid range for **MaxDutyCycle** is [0 : 15]. A value of 0 corresponds to “no duty cycle
14 limitation” except the one set by the regional regulation.

15 A value of 255 means that the end-device shall become silent immediately. It is equivalent to
16 remotely switching off the end-device.

17 An end-device answers to a *DutyCycleReq* with a *DutyCycleAns* command. The
18 *DutyCycleAns* MAC reply does not contain any payload.

5.4 Receive Windows Parameters (*RXParamSetupReq*, *RXParamSetupAns*)

The *RXParamSetupReq* command allows a change to the frequency and the data rate set for the second receive window (RX2) following each uplink. The command also allows to program an offset between the uplink and the RX1 slot downlink data rates.

| Size (bytes) | 1 | 3 |
|----------------------------|------------|-----------|
| RX2SetupReq Payload | DLsettings | Frequency |

| Bits | 7 | 6:4 | 3:0 |
|-------------------|-----|-------------|-------------|
| DLsettings | RFU | RX1DRoffset | RX2DataRate |

The *RX1DRoffset* field sets the offset between the uplink data rate and the downlink data rate used to communicate with the end-device on the first reception slot (RX1). As a default this offset is 0.. The offset is used to take into account maximum power density constraints for base stations in some regions and to balance the uplink and downlink radio link margins.

The data rate (*RX2DataRate*) field defines the data rate of a downlink using the second receive window following the same convention as the *LinkADRReq* command (0 means DR0/125kHz for example). The frequency (**Frequency**) field corresponds to the frequency of the channel used for the second receive window, whereby the frequency is coded following the convention defined in the *NewChannelReq* command.

The *RXParamSetupAns* command is used by the end-device to acknowledge the reception of *RXParamSetupReq* command. The payload contains a single status byte.

| Size (bytes) | 1 |
|----------------------------|--------|
| RX2SetupAns Payload | Status |

The status (**Status**) bits have the following meaning.

| Bits | 7:3 | 2 | 1 | 0 |
|--------------------|-----|-----------------|-------------------|-------------|
| Status bits | RFU | RX1DRoffset ACK | RX2 Data rate ACK | Channel ACK |

| | Bit = 0 | Bit = 1 |
|--------------------------|---|---|
| Channel ACK | The frequency requested is not usable by the end-device. | RX2 slot channel was successfully set |
| RX2 Data rate ACK | The data rate requested is unknown to the end-device. | RX2 slot data rate was successfully set |
| RX1DRoffset ACK | the uplink/downlink data rate offset for RX1 slot is not in the allowed range | RX1DRoffset was successfully set |

Table 7: RX2SetupAns status bits signification

If either of the 3 bits is equal to 0, the command did not succeed and the previous parameters are kept.

1 5.5 End-Device Status (*DevStatusReq*, *DevStatusAns*)

2 With the ***DevStatusReq*** command a network server may request status information from an
 3 end-device. The command has no payload. If a ***DevStatusReq*** is received by an end-
 4 device, it responds with a ***DevStatusAns*** command.

| Size (bytes) | 1 | 1 |
|-----------------------------|---------|--------|
| DevStatusAns Payload | Battery | Margin |

5 The battery level (**Battery**) reported is encoded as follows:

| Battery | Description |
|---------|--|
| 0 | The end-device is connected to an external power source. |
| 1..254 | The battery level, 1 being at minimum and 254 being at maximum |
| 255 | The end-device was not able to measure the battery level. |

6 **Table 8: Battery level decoding**

7 The margin (**Margin**) is the demodulation signal-to-noise ratio in dB rounded to the nearest
 8 integer value for the last successfully received ***DevStatusReq*** command. It is a signed
 9 integer of 6 bits with a minimum value of -32 and a maximum value of 31.

| Bits | 7:6 | 5:0 |
|---------------|-----|--------|
| Status | RFU | Margin |

10 5.6 Creation / Modification of a Channel (*NewChannelReq*, 11 *NewChannelAns*)

12
 13 The ***NewChannelReq*** command can be used to either modify the parameters of an existing
 14 channel or to create a new one. The command sets the center frequency of the new channel
 15 and the range of data rates usable on this channel:

| Size (bytes) | 1 | 3 | 1 |
|------------------------------|---------|------|---------|
| NewChannelReq Payload | ChIndex | Freq | DrRange |

16
 17 The channel index (**ChIndex**) is the index of the channel being created or modified.
 18 Depending on the region and frequency band used, the LoRaWAN specification imposes
 19 default channels which must be common to all devices and cannot be modified by the
 20 ***NewChannelReq*** command (cf. Chapter 6). If the number of default channels is N , the
 21 default channels go from 0 to $N-1$, and the acceptable range for **ChIndex** is N to 15. A
 22 device must be able to handle at least 16 different channel definitions. In certain region the
 23 device may have to store more than 16 channel definitions.

24
 25 The frequency (**Freq**) field is a 24 bits unsigned integer. The actual channel frequency in Hz
 26 is $100 \times \mathbf{Freq}$ whereby values representing frequencies below 100 MHz are reserved for
 27 future use. This allows setting the frequency of a channel anywhere between 100 MHz to
 28 1.67 GHz in 100 Hz steps. A **Freq** value of 0 disables the channel. The end-device has to
 29 check that the frequency is actually allowed by its radio hardware and return an error
 30 otherwise.

31
 32 The data-rate range (**DrRange**) field specifies the data-rate range allowed for this channel.
 33 The field is split in two 4-bit indexes:

| Bits | 7:4 | 3:0 |
|----------------|-------|-------|
| DrRange | MaxDR | MinDR |

1
 2 Following the convention defined in Section 5.2 the minimum data rate (**MinDR**) subfield
 3 designate the lowest data rate allowed on this channel. For example 0 designates DR0 / 125
 4 kHz. Similarly, the maximum data rate (**MaxDR**) designates the highest data rate. For
 5 example, DrRange = 0x77 means that only 50 kbps GFSK is allowed on a channel and
 6 DrRange = 0x50 means that DR0 / 125 kHz to DR5 / 125 kHz are supported.

7 The newly defined channel is enabled and can immediately be used for communication.

8 The end-device acknowledges the reception of a **NewChannelReq** by sending back a
 9 **NewChannelAns** command. The payload of this message contains the following
 10 information:

| Size (bytes) | 1 |
|------------------------------|--------|
| NewChannelAns Payload | Status |

11
 12 The status (**Status**) bits have the following meaning:

| Bits | 7:2 | 1 | 0 |
|---------------|-----|--------------------|----------------------|
| Status | RFU | Data rate range ok | Channel frequency ok |

| | Bit = 0 | Bit = 1 |
|-----------------------------|---|--|
| Data rate range ok | The designated data rate range exceeds the ones currently defined for this end-device | The data rate range is compatible with the possibilities of the end-device |
| Channel frequency ok | The device cannot use this frequency | The device is able to use this frequency. |

13
 14 **Table 9: NewChannelAns status bits signification**

15 If either of those 2 bits equals 0, the command did not succeed and the new channel has not
 16 been created.

17 **5.7 Setting delay between TX and RX (*RXTimingSetupReq*, 18 *RXTimingSetupAns*)**

19 The **RXTimingSetupReq** command allows configuring the delay between the end of the TX
 20 uplink and the opening of the first reception slot. The second reception slot opens one
 21 second after the first reception slot.

| Size (bytes) | 1 |
|---------------------------------|----------|
| RXTimingSetupReq Payload | Settings |

22
 23 The delay (**Delay**) field specifies the delay. The field is split in two 4-bit indexes:

| Bits | 7:4 | 3:0 |
|-----------------|-----|-----|
| Settings | RFU | Del |

24
 25 The delay is expressed in seconds. **Del** 0 is mapped on 1 s.

| Del | Delay [s] |
|-----|-----------|
| 0 | 1 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| .. | .. |

| | |
|----|----|
| 15 | 15 |
|----|----|

1 [Table 10: Del mapping table](#)

2

3 An end device answers *RXTimingSetupReq* with *RXTimingSetupAns* with no payload.

1 **6 End-Device Activation**

2 To participate in a LoRaWAN network, each end-device has to be personalized and
3 activated.

4 Activation of an end-device can be achieved in two ways, either via **Over-The-Air**
5 **Activation** (OTAA) when an end-device is deployed or reset, or via **Activation By**
6 **Personalization** (ABP) in which the two steps of end-device personalization and activation
7 are done as one step.

8 **6.1 Data Stored in the End-device after Activation**

9 After activation, the following information is stored in the end-device: a device address
10 (**DevAddr**), an application identifier (**AppEUI**), a network session key (**NwkSKey**), and an
11 application session key (**AppSKey**).

12 **6.1.1 End-device address (DevAddr)**

13 The **DevAddr** consists of 32 bits identifies the end-device within the current network. Its
14 format is as follows:

| Bit# | [31..25] | [24..0] |
|---------------------|----------|---------|
| DevAddr bits | NwkID | NwkAddr |

15 The most significant 7 bits are used as network identifier (**NwkID**) to separate addresses of
16 territorially overlapping networks of different network operators and to remedy roaming
17 issues. The least significant 25 bits, the network address (**NwkAddr**) of the end-device, can
18 be arbitrarily assigned by the network manager.

19 **6.1.2 Application identifier (AppEUI)**

20 The **AppEUI** is a global application ID in IEEE EUI64 address space that uniquely identifies
21 the application provider (i.e., owner) of the end-device.

22 The **AppEUI** is stored in the end-device before the activation procedure is executed.

23 **6.1.3 Network session key (NwkSKey)**

24 The **NwkSKey** is a **network session key** specific for the end-device. It is used by both the
25 network server and the end-device to calculate and verify the **MIC** (message integrity code)
26 of all data messages to ensure data integrity. It is further used to encrypt and decrypt the
27 payload field of a MAC-only data messages.

28 **6.1.4 Application session key (AppSKey)**

29 The **AppSKey** is an **application session key** specific for the end-device. It is used by both
30 the network server and the end-device to encrypt and decrypt the payload field of
31 application-specific data messages. It is also used to calculate and verify an application-
32 level **MIC** that may be included in the payload of application-specific data messages.

33 **6.2 Over-the-Air Activation**

34 For over-the-air activation, end-devices must follow a join procedure prior to participating in
35 data exchanges with the network server. An end-device has to go through a new join
36 procedure every time it has lost the session context information.

1 The join procedure requires the end-device to be personalized with the following information
 2 before its starts the join procedure: a globally unique end-device identifier (**DevEUI**), the
 3 application identifier (**AppEUI**), and an AES-128 key (**AppKey**).

4 The **AppEUI** is described above in 6.1.2.

5 **Note:** For over-the-air-activation, end-devices are not personalized
 6 with any kind of network key. Instead, whenever an end-device joins a
 7 network, a network session key specific for that end-device is derived
 8 to encrypt and verify transmissions at the network level. This way,
 9 roaming of end-devices between networks of different providers is
 10 facilitated. Using both a network session key and an application
 11 session key further allows federated network servers in which
 12 application data cannot be read or tampered with by the network
 13 provider.

14 6.2.1 End-device identifier (DevEUI)

15 The **DevEUI** is a global end-device ID in IEEE EUI64 address space that uniquely identifies
 16 the end-device.

17 6.2.2 Application key (AppKey)

18 The **AppKey** is an AES-128 application key specific for the end-device that is assigned by
 19 the application owner to the end-device and most likely derived from an application-specific
 20 root key exclusively known to and under the control of the application provider.¹ Whenever
 21 an end-device joins a network via over-the-air activation, the AppKey is used to derive the
 22 session keys NwkSKey and AppSKey specific for that end-device to encrypt and verify
 23 network communication and application data.

24 6.2.3 Join procedure

25 From an end-device's point of view, the join procedure consists of two MAC messages
 26 exchanged with the server, namely a **join request** and a **join accept**.

27 6.2.4 Join-request message

28 The join procedure is always initiated from the end-device by sending a join-request
 29 message.

| Size (bytes) | 8 | 8 | 2 |
|--------------|--------|--------|----------|
| Join Request | AppEUI | DevEUI | DevNonce |

31 The join-request message contains the **AppEUI** and **DevEUI** of the end-device followed by a
 32 **nonce** of 2 octets (**DevNonce**).

33 **DevNonce** is a random value.² For each end-device, the network server keeps track of a
 34 certain number of **DevNonce** values used by the end-device in the past, and ignores join
 35 requests with any of these **DevNonce** values from that end-device.

1. Since all end-devices end up with unrelated application keys specific for each end-device, extracting the AppKey from an end-device only compromises this one end-device.

² The DevNonce can be extracted by issuing a sequence of RSSI measurements under the assumption that the quality of randomness fulfills the criteria of true randomness

Note: This mechanism prevents replay attacks by sending previously recorded join-request messages with the intention of disconnecting the respective end-device from the network.

The message integrity code (**MIC**) value (see Chapter 4 for MAC message description) for a join-request message is calculated as follows:¹

$$cmac = aes128_cmac(AppKey, MHDR | AppEUI | DevEUI | DevNonce)$$

$$MIC = cmac[0..3]$$

The join-request message is not encrypted.

6.2.5 Join-accept message

The network server will respond to the **join-request** message with a **join-accept** message if the end-device is permitted to join a network. The join-accept message is sent like a normal downlink but uses delays JOIN_ACCEPT_DELAY1 or JOIN_ACCEPT_DELAY2 (instead of RECEIVE_DELAY1 and RECEIVE_DELAY2, respectively). The channel frequency and data rate used for these two receive windows are identical to the one used for the RX1 and RX2 receive windows described in the “receive windows” section of the “Physical Layer” chapter

No response is given to the end-device if the join request is not accepted.

The join-accept message contains an application nonce (**AppNonce**) of 3 octets, a network identifier (**NetID**), an end-device address (**DevAddr**), a delay between TX and RX (**RxDelay**) and an optional list of channel frequencies (**CFList**) for the network the end-device is joining. The CFList option is region specific and is defined in Section 7.

| Size (bytes) | 3 | 3 | 4 | 1 | 1 | (16) Optional |
|--------------|----------|-------|---------|------------|---------|---------------|
| Join Accept | AppNonce | NetID | DevAddr | DLSettings | RxDelay | CFList |

The **AppNonce** is a random value or some form of unique ID provided by the network server and used by the end-device to derive the two session keys **NwkSKey** and **AppSKey** as follows:³

$$NwkSKey = aes128_encrypt(AppKey, 0x01 | AppNonce | NetID | DevNonce | pad_{16})$$

$$AppSKey = aes128_encrypt(AppKey, 0x02 | AppNonce | NetID | DevNonce | pad_{16})$$

The MIC value for a join-accept message is calculated as follows:⁴

$$cmac = aes128_cmac(AppKey, MHDR | AppNonce | NetID | DevAddr | RFU | RxDelay | CFList)$$

$$MIC = cmac[0..3]$$

The join-accept message itself is encrypted with the **AppKey** as follows:

$$aes128_decrypt(AppKey, AppNonce | NetID | DevAddr | RFU | RxDelay | CFList | MIC)$$

Note: The network server uses an AES decrypt operation in ECB mode to encrypt the join-accept message so that the end-device can use an AES encrypt operation to decrypt the message. This way an end-device only has to implement AES encrypt but not AES decrypt.

¹ [RFC4493]

³ The pad_{16} function appends zero octets so that the length of the data is a multiple of 16.

⁴ [RFC4493]

1 **Note:** Establishing these two session keys allows for a federated
 2 network server infrastructure in which network operators are not able
 3 to eavesdrop on application data. In such a setting, the application
 4 provider must support the network operator in the process of an end-
 5 device actually joining the network and establishing the NwkSKey for
 6 the end-device. At the same time the application provider commits to
 7 the network operator that it will take the charges for any traffic incurred
 8 by the end-device and retains full control over the AppSKey used for
 9 protecting its application data.

10 The format of the **NetID** is as follows: The seven LSB of the **NetID** are called **NwkID** and
 11 match the seven MSB of the short address of an end-device as described before.
 12 Neighboring or overlapping networks must have different **NwkIDs**. The remaining 17 MSB
 13 can be freely chosen by the network operator.

14 The DLsettings field contains the downlink configuration:

15

| Bits | 7 | 6:4 | 3:0 |
|-------------------|-----|-------------|---------------|
| DLsettings | RFU | RX1DRoffset | RX2 Data rate |

16

17 The RX1DRoffset field sets the offset between the uplink data rate and the downlink data
 18 rate used to communicate with the end-device on the first reception slot (RX1). By default
 19 this offset is 0. The downlink data rate is always lower or equal than the uplink data rate. The
 20 offset is used to take into account maximum power density constraints for base stations in
 21 some regions and to balance the uplink and downlink radio link margins.

22 The actual relationship between the uplink and downlink data rate is region specific and
 23 detailed in the “Physical Layer” section

24 The delay **RxDelay** follows the same convention as the **Delay** field in the
 25 **RXTimingSetupReq** command.

26 **6.3 Activation by Personalization**

27 Under certain circumstances, end-devices can be activated by personalization. Activation by
 28 personalization directly ties an end-device to a specific network by-passing the **join request**
 29 - **join accept** procedure.

30 Activating an end-device by personalization means that the **DevAddr** and the two session
 31 keys **NwkSKey** and **AppSKey** are directly stored into the end-device instead of the **DevEUI**,
 32 **AppEUI** and the **AppKey**. The end-device is equipped with the required information for
 33 participating in a specific LoRa network when started.

34 Each device should have a unique set of NwkSKey and AppSKey. Compromising the keys
 35 of one device shouldn't compromise the security of the communications of other devices.
 36 The process to build those keys should be such that the keys cannot be derived in any way
 37 from publicly available information (like the node address for example).

1 7 Physical Layer

2 7.1 EU 863-870MHz ISM Band

3 7.1.1 EU863-870 Preamble Format

4 The following synchronization words should be used:

| 5 Modulation | Sync word | Preamble length |
|--------------|-----------|-----------------|
| LORA | 0x34 | 8 symbols |
| GFSK | 0xC194C1 | 5 bytes |

6 **Table 11: EU863-870 synch words**

7 7.1.2 EU863-870 ISM Band channel frequencies

8 In Europe, radio spectrum allocation in the ISM band is defined by ETSI [EN300.220].

9 The network channels can be freely attributed by the network operator. However the three
10 following default channels must be implemented in every EU868MHz end-device. Those
11 channels are the minimum set that all network gateways should always be listening on.

12

| Modulation | Bandwidth [kHz] | Channel Frequency [MHz] | FSK Bitrate or LoRa DR / Bitrate | Nb Channels | Duty cycle |
|------------|-----------------|----------------------------|----------------------------------|-------------|------------|
| LoRa | 125 | 868.10 868.30 868.50 | DR0 to DR5 / 0.3-5 kbps | 3 | <1% |

13 **Table 12: EU863-870 default channels**

14 In order to access the physical medium the ETSI regulations impose some restrictions such
15 maximum time the transmitter can be on or the maximum time a transmitter can transmit per
16 hour. The ETSI regulations allow the choice of using either a duty-cycle limitation or a so-
17 called **Listen Before Talk Adaptive Frequency Agility** (LBT AFA) transmissions
18 management. The current LoRaWAN specification exclusively uses duty-cycled limited
19 transmissions to comply with the ETSI regulations.

20 The LoRaWAN enforces a per sub-band duty-cycle limitation. Each time a frame is
21 transmitted in a given sub-band, the time of emission and the on-air duration of the frame
22 are recorded for this sub-band. The same sub-band cannot be used again during the next
23 T_{off} seconds where:

$$T_{off_{subband}} = \frac{TimeOnAir}{DutyCycle_{subband}} - TimeOnAir$$

24 During the unavailable time of a given sub-band, the device may still be able to transmit on
25 another sub-band. If all sub-bands are unavailable, the device has to wait before any further
26 transmission. The device adapts its channel hopping sequence according to the sub-band
27 availability.

28 Example: A device just transmitted a 0.5 s long frame on one default channel. This channel
29 is in a sub-band allowing 1% duty-cycle. Therefore this whole sub-band (868 – 868.6) will be
30 unavailable for 49.5 s.

1 EU868MHz ISM band end-devices should use the following default parameters

- 2 • Default radiated transmit output power: 14 dBm

3 EU868Mhz end-devices should be capable of operating in the 863 to 870 MHz frequency
 4 band and should feature a channel data structure to store the parameters of at least 16
 5 channels. A channel data structure corresponds to a frequency and a set of data rates
 6 usable on this frequency.

7 The first three channels correspond to 868.1, 868.3, and 868.5 MHz / DR0 to DR5 and must
 8 be implemented in every end-device. Those default channels cannot be modified through
 9 the **NewChannelReq** command and guarantee a minimal common channel set between
 10 end-devices and network gateways.

11 The following table gives the list of frequencies that should be used by end-devices to
 12 broadcast the JoinReq message. The JoinReq message transmit duty-cycle should never
 13 exceed 0.1%

14

| Modulation | Bandwidth [kHz] | Channel Frequency [MHz] | FSK Bitrate or LoRa DR / Bitrate | Nb Channels | Duty cycle |
|------------|-----------------|--|----------------------------------|-------------|------------|
| LoRa | 125 | 864.10 864.30 864.50 868.10 868.30 868.50 | DR0 – DR5 / 0.3-5 kbps | 6 | <0.1% |

15 [Table 13: EU863-870 JoinReq Channel List](#)

16 7.1.3 EU863-870 Data Rate and End-point Output Power encoding

17 The following encoding is used for Data Rate (DR) and End-point Output Power (TXPower)
 18 in the EU863-870 band:

| DataRate | Configuration | Indicative physical bit rate [bit/s] | TXPower | Configuration |
|----------|----------------------|--------------------------------------|---------|-----------------------|
| 0 | LoRa: SF12 / 125 kHz | 250 | 0 | 20 dBm (if supported) |
| 1 | LoRa: SF11 / 125 kHz | 440 | 1 | 14 dBm |
| 2 | LoRa: SF10 / 125 kHz | 980 | 2 | 11 dBm |
| 3 | LoRa: SF9 / 125 kHz | 1760 | 3 | 8 dBm |
| 4 | LoRa: SF8 / 125 kHz | 3125 | 4 | 5 dBm |
| 5 | LoRa: SF7 / 125 kHz | 5470 | 5 | 2 dBm |
| 6 | LoRa: SF7 / 250 kHz | 11000 | 6..15 | RFU |
| 7 | FSK: 50 kbps | 50000 | | |
| 8..15 | RFU | | | |

19 [Table 14: Data rate and TX power table](#)

20 7.1.4 EU863-870 JoinAccept CFList

21

22 The EU 863-870 ISM band LoRaWAN implements an optional **channel frequency list**
 23 (CFList) of 16 octets in the JoinAccept message.

1 In this case the **CFList** is a list of five channel frequencies for the channels four to eight
 2 whereby each frequency is encoded as a 24 bits unsigned integer (three octets). All these
 3 channels are usable for DR0 to DR5 125kHz LoRa modulation. The list of frequencies is
 4 followed by a single RFU octet for a total of 16 octets.

5

| Size (bytes) | 3 | 3 | 3 | 3 | 3 | 1 |
|---------------|----------|----------|----------|----------|----------|-----|
| CFList | Freq Ch4 | Freq Ch5 | Freq Ch6 | Freq CN7 | Freq Ch8 | RFU |

6 The actual channel frequency in Hz is 100 x frequency whereby values representing
 7 frequencies below 100 Mhz are reserved for future use. This allows setting the frequency of
 8 a channel anywhere between 100 MHz to 1.67 GHz in 100 Hz steps. Unused channels have
 9 a frequency value of 0. The **CFList** is optional and its presence can be detected by the
 10 length of the join-accept message. If present, the **CFList** replaces all the previous channels
 11 stored in the end-device apart from the three default channels as defined in Chapter 6. The
 12 newly defined channels are immediately enabled and usable by the end-device for
 13 communication.

14 7.1.5 EU863-870 LinkAdrReq command

15 The EU863-870 LoRaWAN only supports a maximum of 16 channels. When **ChMaskCntl**
 16 field is 0 the ChMask field individually enables/disables each of the 16 channels.

17

| ChMaskCntl | ChMask applies to |
|------------|---|
| 0 | Channels 1 to 16 |
| 1 | RFU |
| .. | .. |
| 4 | RFU |
| 5 | RFU |
| 6 | All channels ON The device should enable all currently defined channels independently of the ChMask field value. |
| 7 | RFU |

18 **Table 15: ChMaskCntl value table**

19 If the ChMask field value is one of values meaning RFU, the end-device should reject the
 20 command and unset the “**Channel mask ACK**” bit in its response.

21 7.1.6 EU863-870 Maximum payload size

22 The maximum **MACPayload** size length (M) is given by the following table. It is derived from
 23 limitation of the PHY layer depending on the effective modulation rate used taking into
 24 account a possible repeater encapsulation layer. The maximum application payload length in
 25 the absence of the optional **FOpt** control field (N) is also given for information only. The
 26 value of N might be smaller if the **FOpt** field is not empty.

27

| DataRate | M | N |
|----------|-----|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 230 | 222 |

| | | |
|------|-------------|-----|
| 5 | 230 | 222 |
| 6 | 230 | 222 |
| 7 | 230 | 222 |
| 8:15 | Not defined | |

1 **Table 16: EU863-870 maximum payload size**

2 If the end-device will never operate with a repeater then the maximum application payload
 3 length in the absence of the optional **FOpt** control field should be:

4

| DataRate | M | N |
|----------|-------------|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 250 | 242 |
| 5 | 250 | 242 |
| 6 | 250 | 242 |
| 7 | 250 | 242 |
| 8:15 | Not defined | |

5 **Table 17 : EU863-870 maximum payload size (not repeater compatible)**

6 **7.1.7 EU863-870 Receive windows**

7 The RX1 receive window uses the same channel than the preceding uplink. The data rate is
 8 a function of the uplink data rate and the RX1DROffset as given by the following table. The
 9 allowed values for RX1DROffset are in the [0:5] range. Values in the [6:7] range are
 10 reserved for future use.

11

12

| RX1DROffset Upstream data rate | 0 | 1 | 2 | 3 | 4 | 5 |
|-----------------------------------|----------------------------------|-----|-----|-----|-----|-----|
| | Downstream data rate in RX1 slot | | | | | |
| DR0 | DR0 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR1 | DR1 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR2 | DR2 | DR1 | DR0 | DR0 | DR0 | DR0 |
| DR3 | DR3 | DR2 | DR1 | DR0 | DR0 | DR0 |
| DR4 | DR4 | DR3 | DR2 | DR1 | DR0 | DR0 |
| DR5 | DR5 | DR4 | DR3 | DR2 | DR1 | DR0 |
| DR6 | DR6 | DR5 | DR4 | DR3 | DR2 | DR1 |
| DR7 | DR7 | DR6 | DR5 | DR4 | DR3 | DR2 |

13

14 The RX2 receive window uses a fixed frequency and data rate. The default parameters are
 15 869.525 MHz / DR0 (SF12, 125 kHz)

16 **7.1.8 EU863-870 Default Settings**

17 The following parameters are recommended values for the EU863-870Mhz band.

18 RECEIVE_DELAY1 1 s
 19 RECEIVE_DELAY2 2 s (must be RECEIVE_DELAY1 + 1s)
 20 JOIN_ACCEPT_DELAY1 5 s

| | | |
|----|---|--|
| 1 | JOIN_ACCEPT_DELAY2 | 6 s |
| 2 | MAX_FCNT_GAP | 16384 |
| 3 | ADR_ACK_LIMIT | 64 |
| 4 | ADR_ACK_DELAY | 32 |
| 5 | ACK_TIMEOUT | 2 +/- 1 s (random delay between 1 and 3 seconds) |
| 6 | If the actual parameter values implemented in the end-device are different from those default | |
| 7 | values (for example the end-device uses a longer RECEIVE_DELAY1 and | |
| 8 | RECEIVE_DELAY2 latency), those parameters must be communicated to the network | |
| 9 | server using an out-of-band channel during the end-device commissioning process. The | |
| 10 | network server may not accept parameters different from those default values. | |
| 11 | | |

1 7.2 US 902-928MHz ISM Band

2 7.2.1 US902-928 Preamble Format

3 The following synchronization words should be used:

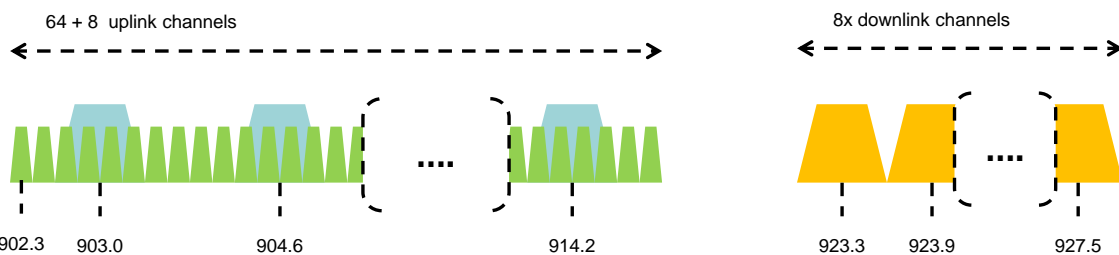
| 4 Modulation | Sync word | Preamble length |
|--------------|-----------|-----------------|
| LORA | 0x34 | 8 symbols |

5 LoRaWAN does not make use of GFSK modulation in the US902-928 ISM band.

6 7.2.2 US902-928 Channel Frequencies

7 The 915 MHz ISM Band shall be divided into the following channel plans.

- 8 • Upstream – 64 channels numbered 0 to 63 utilizing LoRa 125 kHz BW varying from
- 9 DR0 to DR3 starting at 902.3 MHz and incrementing linearly by 200 kHz to 914.9
- 10 MHz
- 11 • Upstream – 8 channels numbered 64 to 71 utilizing LoRa 500 kHz BW at DR4
- 12 starting at 903.0 MHz and incrementing linearly by 1.6 MHz to 914.2 MHz
- 13 • Downstream – 8 channels numbered 0 to 7 utilizing LoRa 500 kHz BW at DR10 to
- 14 DR13) starting at 923.3 MHz and incrementing linearly by 600 kHz to 927.5 MHz
- 15



16
17 **Figure 10: US902-928 channel frequencies**

18 915 MHz ISM band end-devices should use the following default parameters:

- 19 • Default radiated transmit output power: 20 dBm
- 20 ○ Devices, when transmitting with 125 kHz BW may use a maximum of
- 21 +30 dBm. The transmission should never last more than 400 ms.
- 22 ○ Devices, when transmitting with 500 kHz BW may use a maximum of
- 23 +26 dBm

24 US902-928 end-devices should be capable of operating in the 902 to 928 MHz frequency

25 band and should feature a channel data structure to store the parameters of 72 channels. A

26 channel data structure corresponds to a frequency and a set of data rates usable on this

27 frequency.

28 If using the over-the-air activation procedure, the end-device should broadcast the JoinReq

29 message alternatively on a random 125 kHz channel amongst the 64 channels defined using

30 **DR0** and a random 500 kHz channel amongst the 8 channels defined using **DR4**. The end-

31 device should change channel for every transmission.

32 7.2.3 US902-928 Data Rate and End-point Output Power encoding

33 The following encoding is used for Data Rate (**DR**) and End-point Output Power (**TXPower**)

34 in the US902-928 band:

1

| DataRate | Configuration | Indicative physical bit rate [bit/sec] | TXPower | Configuration |
|----------|----------------------|--|---------|--------------------|
| 0 | LoRa: SF10 / 125 kHz | 980 | 0 | 30 dBm – 2*TXpower |
| 1 | LoRa: SF9 / 125 kHz | 1760 | 1 | 28 dBm |
| 2 | LoRa: SF8 / 125 kHz | 3125 | 2 | 26 dBm |
| 3 | LoRa: SF7 / 125 kHz | 5470 | 3 : 9 | |
| 4 | LoRa: SF8 / 500 kHz | 12500 | 10 | 10 dBm |
| 5:7 | RFU | | 11:15 | RFU |
| 8 | LoRa: SF12 / 500 kHz | 980 | | |
| 9 | LoRa: SF11 / 500 kHz | 1760 | | |
| 10 | LoRa: SF10 / 500 kHz | 3900 | | |
| 11 | LoRa: SF9 / 500 kHz | 7000 | | |
| 12 | LoRa: SF8 / 500 kHz | 12500 | | |
| 13 | LoRa: SF7 / 500 kHz | 21900 | | |
| 14:15 | RFU | | | |

2 **Table 18: Data rate and TX power table (Rem: DR4 is identical to DR12, DR8..13 must be implemented in**
 3 **end-devices and are reserved for future applications)**

4 **7.2.4 US902-928 JoinResp CFList**

5 The US902-928 LoRaWAN does not support the use of the optional **CFList** appended to the
 6 JoinResp message. If the **CFList** is not empty it is ignored by the end-device.

7 **7.2.5 US902-928 LinkAdrReq command**

8 For the US902-928 version the **ChMaskCntl** field of the **LinkADRReq** command has the
 9 following meaning:

10

| ChMaskCntl | ChMask applies to |
|------------|--|
| 0 | Channels 0 to 15 |
| 1 | Channels 16 to 31 |
| .. | .. |
| 4 | Channels 64 to 71 |
| 5 | RFU |
| 6 | All 125 kHz ON ChMask applies to channels 65 to 72 |
| 7 | All 125 kHz OFF ChMask applies to channels 65 to 72 |

11 **Table 19: ChMaskCntl value table**

12 If **ChMaskCntl** = 6 (resp 7) then 125 kHz channels are enabled (resp disabled).
 13 Simultaneously the channels 64 to 71 are set according to the **ChMask** bit mask.

14

15

16

17

Note: FCC regulation requires hopping over at least 50 channels when using maximum output power. It is possible to have end-devices with less channels (at least six 125 kHz channels) when limiting the end-device transmit power to 21 dBm.

1 7.2.6 US902-928 Maximum payload size

2 The maximum **MACPayload** size length (M) is given by the following table. It is derived from
 3 the maximum allowed transmission time at the PHY layer taking into account a possible
 4 repeater encapsulation. The maximum application payload length in the absence of the
 5 optional **FOpt** MAC control field (N) is also given for information only. The value of N might
 6 be smaller if the **FOpt** field is not empty:
 7

| DataRate | M | N |
|----------|-------------|-----|
| 0 | 19 | 11 |
| 1 | 61 | 53 |
| 2 | 137 | 129 |
| 3 | 250 | 242 |
| 4 | 250 | 242 |
| 5:7 | Not defined | |
| 8 | 41 | 33 |
| 9 | 117 | 109 |
| 10 | 230 | 222 |
| 11 | 230 | 222 |
| 12 | 230 | 222 |
| 13 | 230 | 222 |
| 14:15 | Not defined | |

8 **Table 20: US902-928 maximum payload size (repeater compatible)**

9 The greyed lines correspond to the data rates that may be used by an end-device behind a
 10 repeater.

11 If the end-device will never operate under a repeater then the maximum application payload
 12 length in the absence of the optional **FOpt** control field should be:
 13

| DataRate | M | N |
|----------|-------------|-----|
| 0 | 19 | 11 |
| 1 | 61 | 53 |
| 2 | 137 | 129 |
| 3 | 250 | 242 |
| 4 | 250 | 242 |
| 5:7 | Not defined | |
| 8 | 61 | 53 |
| 9 | 137 | 129 |
| 10 | 250 | 242 |
| 11 | 250 | 242 |
| 12 | 250 | 242 |
| 13 | 250 | 242 |
| 14:15 | Not defined | |

14 **Table 21 : US902-928 maximum payload size (not repeater compatible)**

7.2.7 US902-928 Receive windows

- The RX1 receive channel is a function of the upstream channel used to initiate the data exchange. The RX1 receive channel can be determined as follows.
 - RX1 Channel Number = Transmit Channel Number modulo 8
- The RX1 window data rate depends on the transmit data rate (see Table 22 below).
- The RX2 (second receive window) settings uses a fixed data rate and frequency. Default parameters are 923.3Mhz / DR8

| Upstream data rate RX1DROffset | Downstream data rate | | | |
|-----------------------------------|----------------------|------|------|------|
| | 0 | 1 | 2 | 3 |
| DR0 | DR10 | DR9 | DR8 | DR8 |
| DR1 | DR11 | DR10 | DR9 | DR8 |
| DR2 | DR12 | DR11 | DR10 | DR9 |
| DR3 | DR13 | DR12 | DR11 | DR10 |
| DR4 | DR13 | DR13 | DR12 | DR11 |
| DR8 | DR8 | DR8 | DR8 | DR8 |
| DR9 | DR9 | DR8 | DR8 | DR8 |
| DR10 | DR10 | DR9 | DR8 | DR8 |
| DR11 | DR11 | DR10 | DR9 | DR8 |
| DR12 | DR12 | DR11 | DR10 | DR9 |
| DR13 | DR13 | DR12 | DR11 | DR10 |

Table 22: Data rate mapping

The allowed values for RX1DROffset are in the [0:3] range. Values in the range [4:7] are reserved for future use.

7.2.8 US902-928 Default Settings

The following parameters are recommended values for the US902-928 band.

| | |
|--------------------|--|
| RECEIVE_DELAY1 | 1 s |
| RECEIVE_DELAY2 | 2 s (must be RECEIVE_DELAY1 + 1s) |
| JOIN_ACCEPT_DELAY1 | 5 s |
| JOIN_ACCEPT_DELAY2 | 6 s |
| MAX_FCNT_GAP | 16384 |
| ADR_ACK_LIMIT | 64 |
| ADR_ACK_DELAY | 32 |
| ACK_TIMEOUT | 2 +/- 1 s (random delay between 1 and 3 seconds) |

If the actual parameter values implemented in the end-device are different from those default values (for example the end-device uses a longer RECEIVE_DELAY1 & 2 latency), those parameters must be communicated to the network server using an out-of-band channel during the end-device commissioning process. The network server may not accept parameters different from those default values.

1 7.3 China 779-787MHz ISM Band

2 7.3.1 CN779-787 Preamble Format

3 The following synchronization words should be used :

| 4 Modulation | 5 Sync word | 6 Preamble length |
|--------------|-------------|-------------------|
| LORA | 0x34 | 8 symbols |
| GFSK | 0xC194C1 | 5 bytes |

7 **Table 23: CN779-787 synch words**

8 7.3.2 CN779-787 ISM Band channel frequencies

9

10 The LoRaWAN can be used in the Chinese 779-787MHz band as long as the radio device EIRP is less than 10mW (or 10dBm).

11 The end-device transmit duty-cycle should be lower than 1%.

12 The LoRaWAN channels center frequency can be in the following range:

- 13 • Minimum frequency : 779.5Mhz
- 14 • Maximum frequency : 786.5 MHz

15 CN780Mhz end-devices should be capable of operating in the 779 to 787 MHz frequency band and should feature a channel data structure to store the parameters of at least 16 channels. A channel data structure corresponds to a frequency and a set of data rates usable on this frequency.

16 The first three channels correspond to 779.5, 779.7 and 779.9 MHz with DR0 to DR5 and must be implemented in every end-device. Those default channels cannot be modified through the **NewChannelReq** command and guarantee a minimal common channel set between end-devices and gateways of all networks. Other channels can be freely distributed across the allowed frequency range on a network per network basis.

17 The following table gives the list of frequencies that should be used by end-devices to broadcast the JoinReq message. The JoinReq message transmit duty-cycle should never exceed 0.1%

18

| 19 Modulation | 20 Bandwidth [kHz] | 21 Channel Frequency [MHz] | 22 FSK Bitrate or LoRa DR / Bitrate | 23 Nb Channels | 24 Duty cycle |
|---------------|--------------------|--|-------------------------------------|----------------|---------------|
| LoRa | 125 | 779.5 779.7 779.9 780.5 780.7 780.9 | DR0 – DR5 / 0.3-5 kbps | 6 | <0.1% |

25 **Table 24: CN780 JoinReq Channel List**

1 7.3.3 CN779-787 Data Rate and End-point Output Power encoding

2 The following encoding is used for Data Rate (DR) and End-point Output Power (TXPower)
3 in the CN780 band:

| DataRate | Configuration | Indicative physical bit rate [bit/s] | TXPower | Configuration |
|----------|----------------------|--------------------------------------|---------|---------------|
| 0 | LoRa: SF12 / 125 kHz | 250 | 0 | 10 dBm |
| 1 | LoRa: SF11 / 125 kHz | 440 | 1 | 7 dBm |
| 2 | LoRa: SF10 / 125 kHz | 980 | 2 | 4 dBm |
| 3 | LoRa: SF9 / 125 kHz | 1760 | 3 | 1 dBm |
| 4 | LoRa: SF8 / 125 kHz | 3125 | 4 | -2 dBm |
| 5 | LoRa: SF7 / 125 kHz | 5470 | 5 | -5 dBm |
| 6 | LoRa: SF7 / 250 kHz | 11000 | 6..15 | RFU |
| 7 | FSK: 50 kbps | 50000 | | |
| 8..15 | RFU | | | |

4 **Table 25: Data rate and TX power table**

5 7.3.4 CN779-787 JoinAccept CFList

6 The CN780 ISM band LoRaWAN implements an optional **channel frequency list** (CFList) of
7 16 octets in the JoinAccept message.

8 In this case the CFList is a list of five channel frequencies for the channels four to eight
9 whereby each frequency is encoded as a 24 bits unsigned integer (three octets). All these
10 channels are usable for DR0 to DR5 125kHz LoRa modulation. The list of frequencies is
11 followed by a single RFU octet for a total of 16 octets.

| Size (bytes) | 3 | 3 | 3 | 3 | 3 | 1 |
|---------------|----------|----------|----------|----------|----------|-----|
| CFList | Freq Ch4 | Freq Ch5 | Freq Ch6 | Freq CN7 | Freq Ch8 | RFU |

13 The actual channel frequency in Hz is 100 x frequency whereby values representing
14 frequencies below 100 Mhz are reserved for future use. This allows setting the frequency of
15 a channel anywhere between 100 MHz to 1.67 GHz in 100 Hz steps. Unused channels have
16 a frequency value of 0. The **CFList** is optional and its presence can be detected by the
17 length of the join-accept message. If present, the **CFList** replaces all the previous channels
18 stored in the end-device apart from the three default channels as defined in Chapter 6.

19 The newly defined channels are immediately enabled and usable by the end-device for
20 communication.

7.3.5 CN779-787 LinkAdrReq command

The CN780 LoRaWAN only supports a maximum of 16 channels. When **ChMaskCntl** field is 0 the ChMask field individually enables/disables each of the 16 channels.

| ChMaskCntl | ChMask applies to |
|------------|---|
| 0 | Channels 1 to 16 |
| 1 | RFU |
| .. | .. |
| 4 | RFU |
| 5 | RFU |
| 6 | All channels ON The device should enable all currently defined channels independently of the ChMask field value. |
| 7 | RFU |

Table 26: ChMaskCntl value table

If the ChMask field value is one of values meaning RFU, then end-device should reject the command and unset the “**Channel mask ACK**” bit in its response.

7.3.6 CN779-787 Maximum payload size

The maximum **MACPayload** size length (M) is given by the following table. It is derived from limitation of the PHY layer depending on the effective modulation rate used taking into account a possible repeater encapsulation layer. The maximum application payload length in the absence of the optional **FOpt** control field (N) is also given for information only. The value of N might be smaller if the **FOpt** field is not empty:

| DataRate | M | N |
|----------|-------------|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 230 | 222 |
| 5 | 230 | 222 |
| 6 | 250 | 242 |
| 7 | 230 | 222 |
| 8:15 | Not defined | |

Table 27: CN780 maximum payload size

If the end-device will never operate with a repeater then the maximum application payload length in the absence of the optional **FOpt** control field should be:

| DataRate | M | N |
|----------|-----|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 250 | 242 |
| 5 | 250 | 242 |
| 6 | 250 | 242 |
| 7 | 250 | 242 |

| | |
|------|-------------|
| 8:15 | Not defined |
|------|-------------|

 1 **Table 28 : CN780 maximum payload size (not repeater compatible)**

 2 **7.3.7 CN779-787 Receive windows**

 3 The RX1 receive window uses the same channel than the preceding uplink. The data rate is
 4 a function of the uplink data rate and the RX1DROffset as given by the following table. The
 5 allowed values for RX1DROffset are in the [0:5] range. Values in the range [6:7] are
 6 reserved for future use

 7

| RX1DROffset Upstream data rate | 0 | 1 | 2 | 3 | 4 | 5 |
|-----------------------------------|----------------------------------|-----|-----|-----|-----|-----|
| | Downstream data rate in RX1 slot | | | | | |
| DR0 | DR0 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR1 | DR1 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR2 | DR2 | DR1 | DR0 | DR0 | DR0 | DR0 |
| DR3 | DR3 | DR2 | DR1 | DR0 | DR0 | DR0 |
| DR4 | DR4 | DR3 | DR2 | DR1 | DR0 | DR0 |
| DR5 | DR5 | DR4 | DR3 | DR2 | DR1 | DR0 |
| DR6 | DR6 | DR5 | DR4 | DR3 | DR2 | DR1 |
| DR7 | DR7 | DR6 | DR5 | DR4 | DR3 | DR2 |

8

 9 The RX2 receive window uses a fixed frequency and data rate. The default parameters are
 10 786 MHz / DR0.

 11 **7.3.8 CN779-787 Default Settings**

12 The following parameters are recommended values for the CN779-787MHz band.

| | |
|-----------------------|--|
| 13 RECEIVE_DELAY1 | 1 s |
| 14 RECEIVE_DELAY2 | 2 s (must be RECEIVE_DELAY1 + 1s) |
| 15 JOIN_ACCEPT_DELAY1 | 5 s |
| 16 JOIN_ACCEPT_DELAY2 | 6 s |
| 17 MAX_FCNT_GAP | 16384 |
| 18 ADR_ACK_LIMIT | 64 |
| 19 ADR_ACK_DELAY | 32 |
| 20 ACK_TIMEOUT | 2 +/- 1 s (random delay between 1 and 3 seconds) |

 21 If the actual parameter values implemented in the end-device are different from those default
 22 values (for example the end-device uses a longer RECEIVE_DELAY1 and
 23 RECEIVE_DELAY2 latency), those parameters must be communicated to the network
 24 server using an out-of-band channel during the end-device commissioning process. The
 25 network server may not accept parameters different from those default values.

1

2 7.4 EU 433MHz ISM Band

3 7.4.1 EU433 Preamble Format

4 The following synchronization words should be used :

5

| Modulation | Sync word | Preamble length |
|------------|-----------|-----------------|
| LORA | 0x34 | 8 symbols |
| GFSK | 0xC194C1 | 5 bytes |

6 [Table 29: EU433 synch words](#)

7 7.4.2 EU433 ISM Band channel frequencies

8 The LoRaWAN can be used in the ETSI 433-434 MHz band as long as the radio device
9 EIRP is less than 10 mW (or 10 dBm).

10 The end-device transmit duty-cycle should be lower than 1%¹.

11 The LoRaWAN channels center frequency can be in the following range:

- 12 • Minimum frequency : 433.175 MHz
- 13 • Maximum frequency : 434.665 MHz

14 EU433 end-devices should be capable of operating in the 433.05 to 434.79 MHz frequency
15 band and should feature a channel data structure to store the parameters of at least 16
16 channels. A channel data structure corresponds to a frequency and a set of data rates
17 usable on this frequency.

18 The first three channels correspond to 433.175, 433.375 and 433.575 MHz with DR0 to DR5
19 and must be implemented in every end-device. Those default channels cannot be modified
20 through the **NewChannelReq** command and guarantee a minimal common channel set
21 between end-devices and gateways of all networks. Other channels can be freely distributed
22 across the allowed frequency range on a network per network basis.

23 The following table gives the list of frequencies that should be used by end-devices to
24 broadcast the JoinReq message. The JoinReq message transmit duty-cycle should never
25 exceed 0.1%

26

| Modulation | Bandwidth [kHz] | Channel Frequency [MHz] | FSK Bitrate or LoRa DR / Bitrate | Nb Channels | Duty cycle |
|------------|-----------------|-------------------------------|----------------------------------|-------------|------------|
| LoRa | 125 | 433.175 433.375 433.575 | DR0 – DR5 / 0.3-5 kbps | 3 | <1% |

27 [Table 30: EU433 JoinReq Channel List](#)

¹ The EN300220 ETSI standard limits to 10% the maximum transmit duty-cycle in the 433MHz ISM band. The LoRaWAN requires a 1% transmit duty-cycle lower than the legal limit to avoid network congestion.

1 7.4.3 EU433 Data Rate and End-point Output Power encoding

2 The following encoding is used for Data Rate (DR) and End-point Output Power (TXPower)
3 in the EU433 band:

| DataRate | Configuration | Indicative physical bit rate [bit/s] | TXPower | Configuration |
|----------|----------------------|--------------------------------------|---------|---------------|
| 0 | LoRa: SF12 / 125 kHz | 250 | 0 | 10 dBm |
| 1 | LoRa: SF11 / 125 kHz | 440 | 1 | 7 dBm |
| 2 | LoRa: SF10 / 125 kHz | 980 | 2 | 4 dBm |
| 3 | LoRa: SF9 / 125 kHz | 1760 | 3 | 1 dBm |
| 4 | LoRa: SF8 / 125 kHz | 3125 | 4 | -2 dBm |
| 5 | LoRa: SF7 / 125 kHz | 5470 | 5 | -5 dBm |
| 6 | LoRa: SF7 / 250 kHz | 11000 | 6..15 | RFU |
| 7 | FSK: 50 kbps | 50000 | | |
| 8..15 | RFU | | | |

4 **Table 31: Data rate and TX power table**

5 7.4.4 EU433 JoinAccept CFList

6
7 The EU433 ISM band LoRaWAN implements an optional **channel frequency list** (CFList) of
8 16 octets in the JoinAccept message.

9 In this case the CFList is a list of five channel frequencies for the channels four to eight
10 whereby each frequency is encoded as a 24 bits unsigned integer (three octets). All these
11 channels are usable for DR0 to DR5 125 kHz LoRa modulation. The list of frequencies is
12 followed by a single RFU octet for a total of 16 octets.

13

| Size (bytes) | 3 | 3 | 3 | 3 | 3 | 1 |
|---------------|----------|----------|----------|----------|----------|-----|
| CFList | Freq Ch4 | Freq Ch5 | Freq Ch6 | Freq CN7 | Freq Ch8 | RFU |

14 The actual channel frequency in Hz is 100 x frequency whereby values representing
15 frequencies below 100 Mhz are reserved for future use. This allows setting the frequency of
16 a channel anywhere between 100 MHz to 1.67 GHz in 100 Hz steps. Unused channels have
17 a frequency value of 0. The **CFList** is optional and its presence can be detected by the
18 length of the join-accept message. If present, the **CFList** replaces all the previous channels
19 stored in the end-device apart from the three default channels as defined in Chapter 6.

20 The newly defined channels are immediately enabled and usable by the end-device for
21 communication.

1 7.4.5 EU433 LinkAdrReq command

2 The CN780 LoRaWAN only supports a maximum of 16 channels. When **ChMaskCntl** field is
 3 0 the ChMask field individually enables/disables each of the 16 channels.

4

| ChMaskCntl | ChMask applies to |
|------------|---|
| 0 | Channels 1 to 16 |
| 1 | RFU |
| .. | .. |
| 4 | RFU |
| 5 | RFU |
| 6 | All channels ON The device should enable all currently defined channels independently of the ChMask field value. |
| 7 | RFU |

5 **Table 32: ChMaskCntl value table**

6 If the ChMask field value is one of the values meaning RFU, then end-device should reject
 7 the command and unset the “**Channel mask ACK**” bit in its response.

8 7.4.6 EU433 Maximum payload size

9 The maximum **MACPayload** size length (M) is given by the following table. It is derived from
 10 limitation of the PHY layer depending on the effective modulation rate used taking into
 11 account a possible repeater encapsulation layer. The maximum application payload length in
 12 the absence of the optional **FOpt** control field (N) is also given for information only. The
 13 value of N might be smaller if the **FOpt** field is not empty:

14

| DataRate | M | N |
|----------|-------------|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 230 | 222 |
| 5 | 230 | 222 |
| 6 | 230 | 222 |
| 7 | 230 | 222 |
| 8:15 | Not defined | |

15 **Table 33: EU433 maximum payload size**

16 If the end-device will never operate with a repeater then the maximum application payload
 17 length in the absence of the optional **FOpt** control field should be:

18

| DataRate | M | N |
|----------|-----|-----|
| 0 | 59 | 51 |
| 1 | 59 | 51 |
| 2 | 59 | 51 |
| 3 | 123 | 115 |
| 4 | 250 | 242 |
| 5 | 250 | 242 |
| 6 | 250 | 242 |
| 7 | 250 | 242 |

| | |
|------|-------------|
| 8:15 | Not defined |
|------|-------------|

Table 34 : EU433 maximum payload size (not repeater compatible)

7.4.7 EU433 Receive windows

The RX1 receive window uses the same channel than the preceding uplink. The data rate is a function of the uplink data rate and the RX1DROffset as given by the following table. The allowed values for RX1DROffset are in the [0:5] range. Values in the range [6:7] are reserved for future use.

| RX1DROffset Upstream data rate | 0 | 1 | 2 | 3 | 4 | 5 |
|-----------------------------------|----------------------------------|-----|-----|-----|-----|-----|
| | Downstream data rate in RX1 slot | | | | | |
| DR0 | DR0 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR1 | DR1 | DR0 | DR0 | DR0 | DR0 | DR0 |
| DR2 | DR2 | DR1 | DR0 | DR0 | DR0 | DR0 |
| DR3 | DR3 | DR2 | DR1 | DR0 | DR0 | DR0 |
| DR4 | DR4 | DR3 | DR2 | DR1 | DR0 | DR0 |
| DR5 | DR5 | DR4 | DR3 | DR2 | DR1 | DR0 |
| DR6 | DR6 | DR5 | DR4 | DR3 | DR2 | DR1 |
| DR7 | DR7 | DR6 | DR5 | DR4 | DR3 | DR2 |

The RX2 receive window uses a fixed frequency and data rate. The default parameters are 434.665MHz / DR0 (SF12 , 125kHz)

7.4.8 EU433 Default Settings

The following parameters are recommended values for the EU863-870Mhz band.

| | |
|--------------------|--|
| RECEIVE_DELAY1 | 1 s |
| RECEIVE_DELAY2 | 2 s (must be RECEIVE_DELAY1 + 1s) |
| JOIN_ACCEPT_DELAY1 | 5 s |
| JOIN_ACCEPT_DELAY2 | 6 s |
| MAX_FCNT_GAP | 16384 |
| ADR_ACK_LIMIT | 64 |
| ADR_ACK_DELAY | 32 |
| ACK_TIMEOUT | 2 +/- 1 s (random delay between 1 and 3 seconds) |

If the actual parameter values implemented in the end-device are different from those default values (for example the end-device uses a longer RECEIVE_DELAY1 & 2 latency) , those parameters must be communicated to the network server using an out-of-band channel during the end-device commissioning process. The network server may not accept parameters different from those default values.

1

CLASS B – BEACON

1 **8 Introduction to Class B**

2 This section describes the LoRaWAN Class B layer which is optimized for battery-powered
3 end-devices that may be either mobile or mounted at a fixed location.

4 End-devices should implement Class B operation when there is a requirement to open
5 receive windows at fixed time intervals for the purpose of enabling server initiated downlink
6 messages.

7 LoRaWAN Class B option adds a synchronized reception window on the end-device.

8 One of the limitations of LoRaWAN Class A is the Aloha method of sending data from the
9 end-device; it does not allow for a known reaction time when the customer application or the
10 server wants to address the end-device. The purpose of Class B is to have an end-device
11 available for reception on a predictable time, in addition to the reception windows that
12 follows the random uplink transmission from the end-device of Class A. Class B is achieved
13 by having the gateway sending a beacon on a regular basis to synchronize the all the end-
14 devices in the network so that the end-device can opening a short extra reception window
15 (called “ping slot”) at a predictable time during a periodic time slot.

16 **Note:** The decision to switch from Class A to Class B comes from the
17 application layer of the end-device. If this class A to Class B switch
18 needs to be controlled from the network side, the customer application
19 must use one of the end-device’s Class A uplinks to send back a
20 downlink to the application layer, and it needs the application layer on
21 the end-device to recognize this request – this process is not managed
22 at the LoRaWAN level.

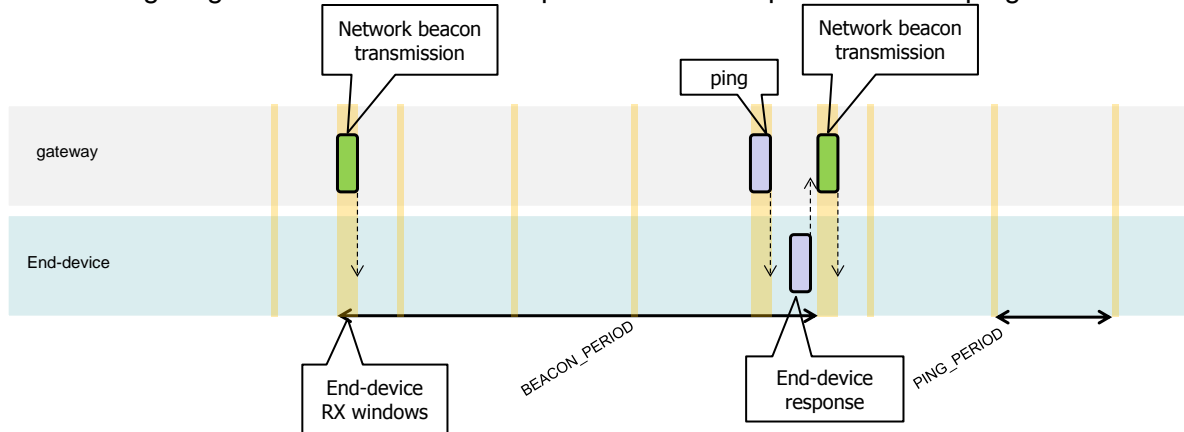
9 Principle of synchronous network initiated downlink (Class-B option)

For a network to support end-devices of Class B, all gateways must synchronously broadcast a beacon providing a timing reference to the end-devices. Based on this timing reference the end-devices can periodically open receive windows, hereafter called “ping slots”, which can be used by the network infrastructure to initiate a downlink communication. A network initiated downlink using one of these ping slots is called a “ping”. The gateway chosen to initiate this downlink communication is selected by the network server based on the signal quality indicators of the last uplink of the end-device. For this reason, if an end-device moves and detects a change in the identity advertised in the received beacon, it must send an uplink to the network server so that the server can update the downlink routing path database.

All end-devices start and join the network as end-devices of Class A. The end-device application can then decide to switch to Class B. This is done through the following process:

- The end-device application requests the LoRaWAN layer to switch to Class B mode. The LoRaWAN layer in the end-device searches for a beacon and returns either a BEACON_LOCKED service primitive to the application if a network beacon was found and locked or a BEACON_NOT_FOUND service primitive. To accelerate the beacon discovery the LoRaWAN layer may use the “BeaconTimingReq” message described later.
- Based on the beacon strength and the battery life constraints, the end-device application selects a ping slot data rate and periodicity, this is then requested them from the end-device LoRaWAN layer.
- Once in Class B mode, the MAC layer sets to 1 the *Class B* bit of the FCTRL field of every uplink frame transmitted. This bit signals to the server that the device has switched to Class B. The MAC layer will autonomously schedule a reception slot for each beacon and each ping slot. When the beacon reception is successful the end-device LoRaWAN layer forwards the beacon content to the application together with the measured radio signal strength. The end-device LoRaWAN layer takes into account the maximum possible clock drift in the scheduling of the beacon reception slot and ping slots. When a downlink is successfully demodulated during a ping slot, it is processed similarly to a downlink as described in the LoRaWAN Class A specification.
- A mobile end-device must periodically inform the network server of its location to update the downlink route. This is done by transmitting a normal (possibly empty) “unconfirmed” or “confirmed” uplink. The end-device LoRaWAN layer will appropriately set the *Class B* bit to 1. Optimally this can be done more efficiently if the application detects that the node is moving by analyzing the beacon content. In that case the end-device must apply a random delay (as defined in Section 15.5 between the beacon reception and the uplink transmission to avoid systematic uplink collisions.
- If no beacon has been received for a given period (as defined in Section 12.2), the synchronization with the network is lost. The MAC layer must inform the application layer that it has switched back to Class A. As a consequence the end-device LoRaWAN layer stops setting the *Class B* bit in all uplinks and this informs the network server that the end-device is no longer in Class B mode. The end-device application can try to switch back to Class B periodically. This will restart this process starting with a beacon search.

1 The following diagram illustrates the concept of beacon reception slots and ping slots.



2
3 **Figure 11: Beacon reception slot and ping slots**

4 In this example, given the beacon period is 128 s, the end-device also opens a ping
5 reception slot every 32 s. Most of the time this ping slot is not used by the server and
6 therefore the end-device reception window is closed as soon as the radio transceiver has
7 assessed that no preamble is present on the radio channel. If a preamble is detected the
8 radio transceiver will stay on until the downlink frame is demodulated. The MAC layer will
9 then process the frame, check that its address field matches the end-device address and
10 that the Message Integrity Check is valid before forwarding it to the application layer.

1 **10 Uplink frame in Class B mode**

2 The uplink frames in Class B mode are same as the Class A uplinks with the exception of
 3 the RFU bit in the FCtrl field in the Frame header. In the Class A uplink this bit is unused
 4 (RFU). This bit is used for Class B uplinks.

5

| Bit# | 7 | 6 | 5 | 4 | 3..0 |
|-------|-----|-----------|-----|---------|----------|
| FCtrl | ADR | ADRACKReq | ACK | Class B | FOptsLen |

6

7 The *Class B* bit set to 1 in an uplink signals the network server that the device as switched to
 8 Class B mode and is now ready to receive scheduled downlink pings.

9

10 The signification of the FPending bit for downlink is unaltered and still signals that one or
 11 more downlink frames are queued for this device in the server and that the device should
 12 keep is receiver on as described in the Class A specification.

13

1 **11 Downlink Ping frame format (Class B option)**

2 **11.1 Physical frame format**

3 A downlink Ping uses the same format as a Class A downlink frame but might follow a
4 different channel frequency plan.

5 **11.2 Unicast & Multicast MAC messages**

6 Messages can be “unicast” or “multicast”. Unicast messages are sent to a single end-device
7 and multicast messages are sent to multiple end-devices. All devices of a multicast group
8 must share the same multicast address and associated encryption keys. The LoRaWAN
9 Class B specification does not specify means to remotely setup such a multicast group or
10 securely distribute the required multicast key material. This must either be performed during
11 the node personalization or through the application layer.

12 **11.2.1 Unicast MAC message format**

13 The MAC payload of a unicast downlink **Ping** uses the format defined in the Class A
14 specification. It is processed by the end-device in exactly the same way. The same frame
15 counter is used and incremented whether the downlink uses a Class B ping slot or a Class A
16 “piggy-back” slot.

17 **11.2.2 Multicast MAC message format**

18 The Multicast frames share most of the unicast frame format with a few exceptions:

- 19 • They are not allowed to carry MAC commands, neither in the **FOpt** field, nor in the
20 payload on port 0 because a multicast downlink does not have the same
21 authentication robustness as a unicast frame.
- 22 • The **ACK** and **ADRACKReq** bits must be zero. The **MType** field must carry the value
23 for Unconfirmed Data Down.
- 24 • The **FPending** bit indicates there is more multicast data to be sent. If it is set the
25 next multicast receive slot will carry a data frame. If it is not set the next slot may or
26 may not carry data. This bit can be used by end-devices to evaluate priorities for
27 conflicting reception slots.

28

1 **12 Beacon acquisition and tracking**

2 Before switching from Class A to Class B, the end-device must first receive one of the
3 network beacons to align his internal timing reference with the network.

4 Once in Class B, the end-device must periodically search and receive a network beacon to
5 cancel any drift of its internal clock time base, relative to the network timing.

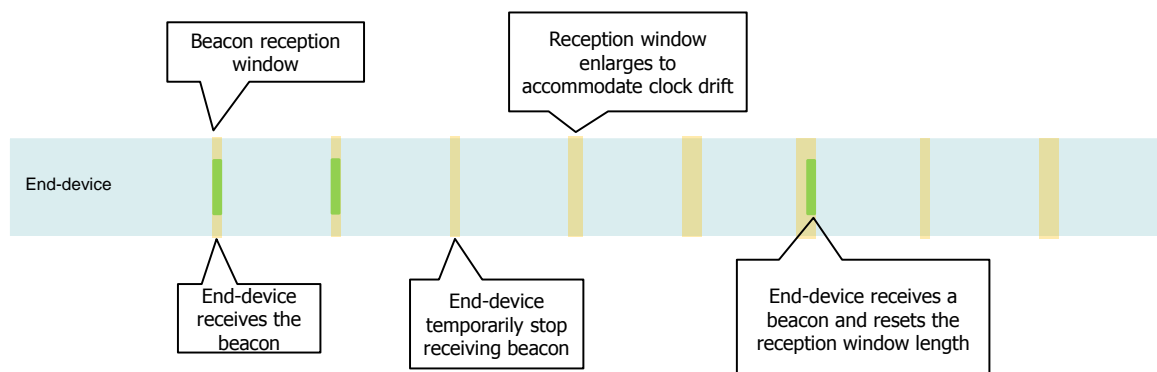
6 A Class B device may be temporarily unable to receive beacons (out of range from the
7 network gateways, presence of interference, ..). In this event, the end-device has to
8 gradually widen its beacon and ping slots reception windows to take into account a possible
9 drift of its internal clock.

10 **Note:** For example, a device which internal clock is defined with a +/-
11 10ppm precision may drift by +/-1.3mSec every beacon period.

12 **12.1 Minimal beacon-less operation time**

13 In the event of beacon loss, a device shall be capable of maintaining Class B operation for 2
14 hours (120 minutes) after it received the last beacon. This temporary Class B operation
15 without beacon is called “beacon-less” operation. It relies on the end-device’s own clock to
16 keep timing.

17 During beacon-less operation, unicast, multicast and beacon reception slots must all be
18 progressively expanded to accommodate the end-device’s possible clock drift.
19



20 **Figure 12 : beacon-less temporary operation**

22 **12.2 Extension of beacon-less operation upon reception**

23 During this 120 minutes time interval the reception of any beacon directed to the end-device,
24 should extend the Class B beacon-less operation further by another 120 minutes as it allows
25 to correct any timing drift and reset the receive slots duration.

26 **12.3 Minimizing timing drift**

27 The end-devices may use the beacon’s (when available) precise periodicity to calibrate their
28 internal clock and therefore reduce the initial clock frequency imprecision. As the timing
29 oscillator’s exhibit a predictable temperature frequency shift, the use of a temperature
30 sensor could enable further minimization of the timing drift.

13 Class B Downlink slot timing

13.1 Definitions

To operate successfully in Class B the end-device must open reception slots at precise instants relative to the infrastructure beacon. This section defines the required timing.

The interval between the start of two successive beacons is called the beacon period. The beacon frame transmission is aligned with the beginning of the BEACON_RESERVED interval. Each beacon is preceded by a guard time interval where no ping slot can be placed. The length of the guard interval corresponds to the time on air of the longest allowed frame. This is to insure that a downlink initiated during a ping slot just before the guard time will always have time to complete without colliding with the beacon transmission. The usable time interval for ping slot therefore spans from the end of the beacon reserved time interval to the beginning of the next beacon guard interval.

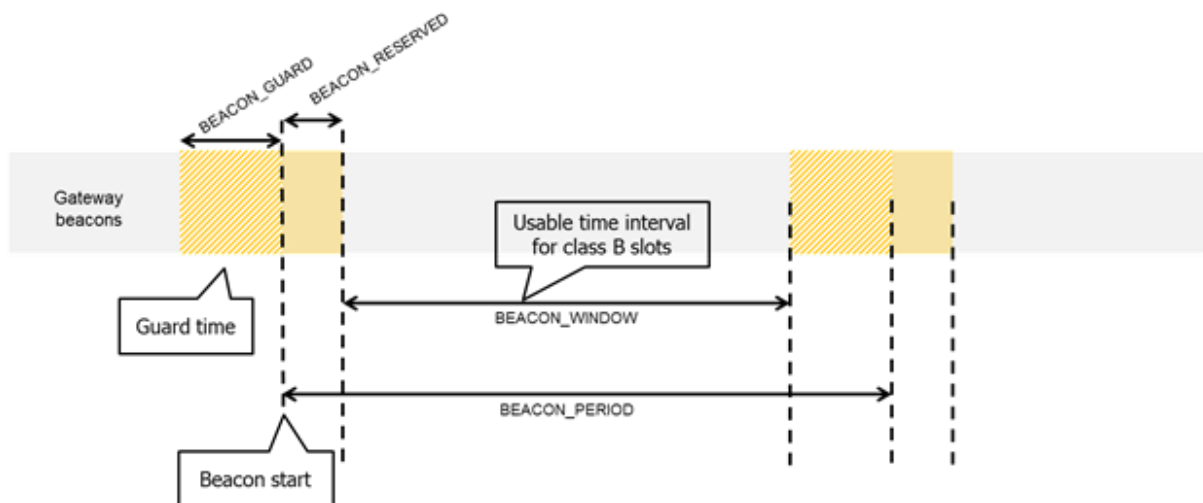


Figure 13: Beacon timing

| | |
|-----------------|-----------|
| Beacon_period | 128 s |
| Beacon_reserved | 2.120 s |
| Beacon_guard | 3.000 s |
| Beacon-window | 122.880 s |

Table 35: Beacon timing

15

The beacon frame time on air is actually much shorter than the beacon reserved time interval to allow appending network management broadcast frames in the future.

The beacon window interval is divided into $2^{12} = 4096$ ping slots of 30 ms each numbered from 0 to 4095.

An end-device using the slot number N must turn on its receiver exactly T_{on} seconds after the start of the beacon where:

$$T_{on} = beacon_reserved + N * 30 \text{ ms}$$

N is called the *slot index*.

The latest ping slot starts at $beacon_reserved + 4095 * 30 \text{ ms} = 124\,970 \text{ ms}$ after the beacon start or 3030 ms before the beginning of the next beacon.

25

1 13.2 Slot randomization

2 To avoid systematic collisions or over-hearing problems the slot index is randomized and
3 changed at every beacon period.

4 The following parameters are used:

5

| | |
|-------------------|---|
| DevAddr | Device 32 bit network unicast or multicast address |
| <i>pingNb</i> | Number of ping slots per beacon period. This must be a power of 2 integer: $pingNb = 2^k$ where $1 \leq k \leq 7$ |
| <i>pingPeriod</i> | Period of the device receiver wake-up expressed in number of slots: $pingPeriod = 2^{12} / pingNb$ |
| <i>pingOffset</i> | Randomized offset computed at each beacon period start. Values can range from 0 to (pingPeriod-1) |
| <i>beaconTime</i> | The time carried in the field BCNPayload . Time of the immediately preceding beacon frame |
| <i>slotLen</i> | Length of a unit ping slot = 30 ms |

6

7 At each beacon period the end-device and the server compute a new pseudo-random offset
8 to align the reception slots. An AES encryption with a fixed key of all zeros is used to
9 randomize:

10 $Key = 16 \times 0x00$

11 $Rand = aes128_encrypt(Key, beaconTime \parallel DevAddr \parallel pad16)$

12 $pingOffset = (Rand[0] + Rand[1] \times 256) \text{ modulo } pingPeriod$

13 The slots used for this beacon period will be:

14 $pingOffset + N \times pingPeriod$ with $N=[0:pingNb-1]$

15 The node therefore opens receive slots starting at :

| | |
|------------|---|
| First slot | Beacon_reserved + pingOffset x slotLen |
| Slot 2 | Beacon_reserved + (pingOffset + pingPeriod) x slotLen |
| Slot 3 | Beacon_reserved + (pingOffset + 2 x pingPeriod) x slotLen |
| ... | ... |

16 If the end-device serves simultaneously a unicast and one or more multicast slots this
17 computation is performed multiple times at the beginning of a new beacon period. Once for
18 the unicast address (the node network address) and once for each multicast group address.

19 In the case where a multicast ping slot and a unicast ping slot collide and cannot be served
20 by the end-device receiver then the end-device should preferentially listen to the multicast
21 slot. If there is a collision between multicast reception slots the FPending bit of the previous
22 multicast frame can be used to set a preference.

23 The randomization scheme prevents a systematic collision between unicast and multicast
24 slots. If collisions happen during a beacon period then it is unlikely to occur again during the
25 next beacon period.

1 14 Class B MAC commands

2 All commands described in the Class A specification shall be implemented in Class B
3 devices. The Class B specification adds the following MAC commands.

4

| CID | Command | Transmitted by | | Short Description |
|------|----------------------------------|----------------|---------|--|
| | | End-device | Gateway | |
| 0x10 | <i>PingSlotInfoReq</i> | x | | Used by the end-device to communicate the ping unicast slot data rate and periodicity to the network server |
| 0x10 | <i>PingSlotInfoAns</i> | | x | Used by the network to acknowledge a PingInfoSlotReq command |
| 0x11 | <i>PingSlotChannelReq</i> | | x | Used by the network server to set the unicast ping channel of an end-device |
| 0x11 | <i>PingSlotFreqAns</i> | x | | Used by the end-device to acknowledge a <i>PingSlotChannelReq</i> command |
| 0x12 | <i>BeaconTimingReq</i> | x | | Used by end-device to request next beacon timing & channel to network |
| 0x12 | <i>BeaconTimingAns</i> | | x | Used by network to answer a <i>BeaconTimingReq</i> uplink |
| 0x13 | <i>BeaconFreqReq</i> | | x | Command used by the network server to modify the frequency at which the end-device expects to receive beacon broadcast |
| 0x13 | <i>BeaconFreqAns</i> | x | | Used by the end-device to acknowledge a BeaconFreqReq command |

5 14.1 PingSlotInfoReq

6 With the ***PingSlotInfoReq*** command an end-device informs the server of its unicast ping
7 slot periodicity and expected data rate. This command must only be used to inform the
8 server of the parameters of a UNICAST ping slot. A multicast slot is entirely defined by the
9 application and should not use this command.

10

| | |
|--------------------------------|-------------------------|
| Size (bytes) | 1 |
| PingSlotInfoReq Payload | Periodicity & data rate |

11

| Bit# | 7 | [6:4] | [3:0] |
|------------------------------------|-----|-------------|-----------|
| Periodicity & data rate | RFU | Periodicity | Data rate |

12 The **Periodicity** subfield is an unsigned 3 bits integer encoding the ping slot period currently
13 used by the end-device using the following equation.

$$14 \quad pingSlotPeriod = 2^{Periodicity} \text{ in seconds}$$

- 15 • **Periodicity** = 0 means that the end-device opens a ping slot every second
- 16 • **Periodicity** = 7, every 128 seconds which is the maximum ping period supported by
17 the LoRaWAN Class B specification.

18 The **Data rate** subfield encodes the data rate at which the end-point expects any ping. This
19 uses the same encoding scheme that the ***LinkAdrReq*** command described in the Class A
20 specification.

- 1 The server needs to be aware of the end-device ping slot periodicity or expected data rate
 2 else Class B downlinks will not happen successfully. For that purpose the **PingSlotInfoReq**
 3 MAC command **must be acknowledged** with a **PingSlotInfoAns** before the device can
 4 switch from class A to Class B. To change its ping slot scheduling or data rate a device
 5 should first revert to Class A , send the new parameters through a **PingSlotInfoReq**
 6 command and get an acknowledge from the server through a **PinSlotInfoAns** . It can then
 7 switch back to Class B with the new parameters.
- 8 This command can be concatenated with any other MAC command in the **FHDRFOpt** field
 9 as described in the Class A specification frame format.

10 **14.2 BeaconFreqReq**

- 11 This command is sent by the server to the end-device to modify the frequency on which this
 12 end-device expects the beacon.

13

| Octets | 3 |
|---------------------------------------|-----------|
| PingSlotChannelReqPay load | Frequency |

14

- 15 The Frequency coding is identical to the **NewChannelReq** MAC command defined in the
 16 Class A.

- 17 **Frequency** is a 24bits unsigned integer. The actual beacon channel frequency in Hz is 100
 18 x frequ. This allows defining the beacon channel anywhere between 100 MHz to 1.67 GHz
 19 by 100 Hz step. The end-device has to check that the frequency is actually allowed by its
 20 radio hardware and return an error otherwise.

- 21 A valid non-zero Frequency will force the device to listen to the beacon on a fixed frequency
 22 channel even if the default behavior specifies a frequency hopping beacon (i.e US ISM
 23 band).

- 24 A value of 0 instructs the end-device to use the default beacon frequency plan as defined in
 25 the “Beacon physical layer” section. Where applicable the device resumes frequency
 26 hopping beacon search.

27 **14.3 PingSlotChannelReq**

- 28 This command is sent by the server to the end-device to modify the frequency on which this
 29 end-device expects the downlink pings.

30

| Octets | 3 | 1 |
|---------------------------------------|-----------|---------|
| PingSlotChannelReq Payload | Frequency | DrRange |

31

- 32 The Frequency coding is identical to the **NewChannelReq** MAC command defined in the
 33 Class A.

- 34 **Frequency** is a 24bits unsigned integer. The actual ping channel frequency in Hz is 100 x
 35 frequ. This allows defining the ping channel anywhere between 100MHz to 1.67GHz by
 36 100Hz step. The end-device has to check that the frequency is actually allowed by its radio
 37 hardware and return an error otherwise.

- 38 A value of 0 instructs the end-device to use the default frequency plan.

1 **DrRange** is the data rate range allowed on this channel. This byte is split in two 4-bit
2 indexes.

| | | |
|----------------|---------------|---------------|
| Bits | 7:4 | 3:0 |
| DrRange | Max data rate | Min data rate |

4
5 Following the convention defined in the “Physical layer” section of the Class A specification,
6 the “Min data rate” subfield designates the lowest data rate allowed on this channel. For
7 example 0 designates DR0 / 125 kHz in the EU physical layer. Similarly “Max data rate”
8 designates the highest data rate. For example in the EU spec, DrRange = 0x77 means that
9 only 50 kbps GFSK is allowed on a channel and DrRange = 0x50 means that DR0 / 125 kHz
10 to DR5 / 125 kHz are supported.

11 Upon reception of this command the end-device answers with a **PingSlotFreqAns**
12 message. The MAC payload of this message contains the following information:

| | |
|--------------------------------|--------|
| Size (bytes) | 1 |
| pingSlotFreqAns Payload | Status |

14 The **Status** bits have the following meaning:

| | | | |
|---------------|-----|-----------------------|-------------------------|
| Bits | 7:2 | 1 | 0 |
| Status | RFU | Data rate range ok | Channel frequency ok |

| | Bit = 0 | Bit = 1 |
|-----------------------------|---|--|
| Data rate range ok | The designated data rate range exceeds the ones currently defined for this end device, the previous range is kept | The data rate range is compatible with the possibilities of the end device |
| Channel frequency ok | The device cannot use this frequency, the previous ping frequency is kept | The device is able to use this frequency. |

16 14.4 BeaconTimingReq

17 This command is sent by the end-device to request the next beacon timing and channel.
18 This MAC command has no payload. The **BeaconTimingReq** & **BeaconTimingAns**
19 mechanism is only meant to accelerate the initial beacon search to lower the end-device
20 energy requirement.

21 The network may answer only a limited number of requests per a given time period. An end-
22 device must not expect that **BeaconTimingReq** is answered immediately with a
23 **BeaconTimingAns**. Class A end-devices wanting to switch to Class B should not transmit
24 more than one **BeaconTimingReq** per hour.

25 End-devices requiring a fast beacon lock must implement an autonomous beacon finding
26 algorithm.

1 14.5 BeaconTimingAns

2 This command is sent by the network to answer a **BeaconInfoReq** request.

| Size (bytes) | 2 | 1 |
|-----------------------------|-------|---------|
| BeaconInfoReqPayload | Delay | Channel |

3 The “**Delay**” field is a 16bits unsigned integer. If the remaining time between the end of the
 4 current downlink frame and the start of the next beacon frame is noted *RTime* then:

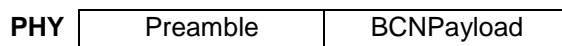
5 $30\text{ ms} \times (\mathbf{Delay}+1) > RTime \geq 30\text{ ms} \times \mathbf{Delay}$

6 In networks where the beacon uses alternatively several channels, the “**Channel**” field is the
 7 index of the beaoning channel on which the next beacon will be broadcasted. For networks
 8 where the beacon broadcast frequency is fixed then this field content is 0.
 9

1 15 Beaconing (Class B option)

2 15.1 Beacon physical layer

3 Besides relaying messages between end-devices and network servers, all gateways
 4 participate in providing a time-synchronization mechanisms by sending beacons at regular
 5 fixed intervals configurable per network (BEACON_INTERVAL). All beacons are transmitted
 6 in radio packet implicit mode, that is, without a LoRa physical header and with no CRC being
 7 appended by the radio.



9 The beacon Preamble begins with (a longer than default) 10 unmodulated symbols. This
 10 allows end-devices to implement a low power duty-cycled beacon search.

11 The beacon frame length is tightly coupled to the operation of the radio Physical layer.
 12 Therefore the actual frame length might change from one region implementation to another.
 13 The changing fields are highlighted in **Bold** in the following sections.

14 15.1.1 EU 863-870MHz ISM Band

15 The beacons are transmitted using the following settings

| | | |
|-----------|-------------|--|
| DR | 3 | Corresponds to SF9 spreading factor with 125 kHz BW |
| CR | 1 | Coding rate = 4/5 |
| frequency | 869.525 MHz | This is the recommended frequency allowing +27 dBm EIRP. Network operators may use a different frequency as long as ETSI compliance is achieved |

16 The beacon frame content is:

| | | | | | |
|---------------------|-------|------|------------|------------|------------|
| Size (bytes) | 3 | 4 | 1 | 7 | 2 |
| BCNPayload | NetID | Time | CRC | GwSpecific | CRC |

17 15.1.2 US 902-928MHz ISM Band

18 The beacons are transmitted using the following settings:

| | | |
|-------------|--|---|
| DR | 10 | Corresponds to SF10 spreading factor with 500kHz bw |
| CR | 1 | Coding rate = 4/5 |
| frequencies | 923.3 to 927.5MHz with 600kHz steps | Beaconing is performed on the same channel that normal downstream traffic as defined in the Class A specification |

19 The downstream channel used for a given beacon is:

20
$$\text{Channel} = \left[\text{floor} \left(\frac{\text{beacon_time}}{\text{beacon_period}} \right) \right] \text{ modulo } 8$$

- 21
- 22 • whereby beacon_time is the integer value of the 4 bytes "Time" field of the beacon frame
 - 23 • whereby beacon_period is the periodicity of beacons , 128 seconds
 - 24 • whereby $\text{floor}(x)$ designates rounding to the integer immediately inferior to x
- 25

Example: the first beacon will be transmitted on 923.3Mhz , the second on 923.9MHz, the 9th beacon will be on 923.3Mhz again.

| Beacon channel nb | Frequency [MHz] |
|-------------------|-----------------|
| 0 | 923.3 |
| 1 | 923.9 |
| 2 | 924.5 |
| 3 | 925.1 |
| 4 | 925.7 |
| 5 | 926.3 |
| 6 | 926.9 |
| 7 | 927.5 |

The beacon frame content is:

| Size (bytes) | 3 | 4 | 2 | 7 | 1 | 2 |
|-------------------|-------|------|------------|------------|------------|------------|
| BCNPayload | NetID | Time | CRC | GwSpecific | RFU | CRC |

15.2 Beacon frame content

The beacon payload **BCNPayload** consists of a network common part and a gateway-specific part.

| Size (bytes) | 3 | 4 | 1/2 | 7 | 0/1 | 2 |
|-------------------|-------|------|-----|------------|-----|-----|
| BCNPayload | NetID | Time | CRC | GwSpecific | RFU | CRC |

The network common part contains a network identifier **NetID** to uniquely identify the network for which the beacon is sent, and a timestamp **Time** in seconds since 00:00:00 [Coordinated Universal Time](#) (UTC), 1 January 1970. The integrity of the beacon's network common part is protected by an 8 or 16 bits CRC depending on PHY layer parameters. The CRC-16 is computed on the NetID+Time fields as defined in the IEEE 802.15.4-2003 section 7.2.1.8. When an 8 bits CRC is required then the 8 LSBs of the computed CRC-16 are used.

For example: This is a valid EU868 beacon frame:

AA BB CC | 00 00 02 CC | 7E | 00 | 01 20 00 | 00 81 03 | DE 55

Bytes are transmitted left to right. The corresponding field values are:

| Field | NetID | Time | CRC | InfoDesc | lat | long | CRC |
|------------------|--------|----------|------------|----------|--------|--------|-------------|
| Value Hex | CCBBAA | CC020000 | 7E | 0 | 002001 | 038100 | 55DE |

The CRC-16 of the NetID+Time fields is 0xC87E but only the 8LSBs are used in that case

The seven LSB of the **NetID** are called **NwkID** and match the seven MSB of the short address of an end-device. Neighboring or overlapping networks **must have** different **NwkIDs**.

1 The gateway specific part provides additional information regarding the gateway sending a
 2 beacon and therefore may differ for each gateway. The RFU field when applicable (region
 3 specific) should be equal to 0. The optional part is protected by a CRC-16 computed on the
 4 GwSpecific+RFU fields. The CRC-16 definition is the same as for the mandatory part.

5 For example: This is a valid US900 beacon:

| Field | NetID | Time | CRC | InfoDesc | lat | long | RFU | CRC |
|-----------|--------|----------|-------------|----------|--------|--------|-----------|-------------|
| Value Hex | CCBBAA | CC020000 | C87E | 0 | 002001 | 038100 | 00 | D450 |

6 Over the air the bytes are sent in the following order:

7 AA BB CC | 00 00 02 CC | 7E C8 | 00 | 01 20 00 | 00 81 03 | 00 | 50 D4

8 Listening and synchronizing to the network common part is sufficient to operate a stationary
 9 end-device in Class B mode. A mobile end-device should also demodulate the gateway
 10 specific part of the beacon to be able to signal to the network server whenever he is moving
 11 from one cell to another.

12 **Note:** As mentioned before, all gateways send their beacon at exactly
 13 the same point in time (i.e., time-synchronously) so that for network
 14 common part there are no visible on-air collisions for a listening end-
 15 device even if the end-device simultaneously receives beacons from
 16 several gateways. With respect to the gateway specific part, collision
 17 occurs but an end-device within the proximity of more than one
 18 gateway will still be able to decode the strongest beacon with high
 19 probability.

20 15.3 Beacon GwSpecific field format

21 The content of the **GwSpecific** field is as follow:

| Size (bytes) | 1 | 6 |
|-------------------|----------|------|
| GwSpecific | InfoDesc | Info |

22 The information descriptor **InfoDesc** describes how the information field **Info** shall be
 23 interpreted.

24

| InfoDesc | Meaning |
|----------|---|
| 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 |
| | |

25 For a single omnidirectional antenna gateway the **InfoDesc** value is 0 when broadcasting
 26 GPS coordinates. For a site featuring 3 sectorized antennas for example, the first antenna
 27 broadcasts the beacon with **InfoDesc** equals 0, the second antenna with **InfoDesc** field
 28 equals 1, etc ...

1 15.3.1 Gateway GPS coordinate: InfoDesc = 0, 1 or 2

2 For **InfoDesc** = 0, 1 or 2, the content of the **Info** field encodes the GPS coordinates of the
3 antenna broadcasting the beacon

| Size (bytes) | 3 | 3 |
|--------------|-----|-----|
| Info | Lat | Lng |

4 The latitude and longitude fields (**Lat** and **Lng**, respectively) encode the geographical
5 location of the gateway as follows:

- 6 • The north-south latitude is encoded using a signed 24 bit word where -2^{23}
7 corresponds to 90° south (the South Pole) and 2^{23} corresponds to 90° north (the
8 North Pole). The equator corresponds to 0.
- 9 • The east-west longitude is encoded using a signed 24 bit word where -
10 2^{23} corresponds to 180° west and 2^{23} corresponds to 180° east. The Greenwich
11 meridian corresponds to 0.

12 15.4 Beacons precise timing

13 The beacon is sent every 128 seconds starting at 00:00:00 Coordinated Universal Time
14 (UTC), 1 January 1970 plus **NwkID** plus TBeaconDelay. Therefore the beacon is sent at
15 $B_T = k * 128 + \mathbf{NwkID} + T\text{BeaconDelay}$

16 seconds after 00:00:00 Coordinated Universal Time (UTC), 1 January 1970

17 whereby k is the smallest integer for which

$$18 \quad k * 128 + \mathbf{NwkID} > T$$

19 whereby

20 T = seconds since 00:00:00 Coordinated Universal Time (UTC), 1 January 1970.

21 **Note:** T is not (!) Unix time. Similar to GPS time and unlike Unix time,
22 T is strictly monotonically increasing and is not influenced by leap
23 seconds.

24 Whereby TBeaconDelay is a network specific delay in the [0:50] ms range.

25 TBeaconDelay may vary from one network to another and is meant to allow a slight
26 transmission delay of the gateways. TBeaconDelay must be the same for all gateways of a
27 given network. TBeaconDelay must be smaller than 50 ms. All end-devices ping slots use
28 the beacon transmission time as a timing reference, therefore the network server as to take
29 TBeaconDelay into account when scheduling the class B downlinks.
30

31 15.5 Network downlink route update requirements

32 When the network attempts to communicate with an end-device using a Class B downlink
33 slot, it transmits the downlink from the gateway which was closest to the end-device when
34 the last uplink was received. Therefore the network server needs to keep track of the rough
35 position of every Class B device.

36 Whenever a Class B device moves and changes cell, it needs to communicate with the
37 network server in order to update its downlink route. This update can be performed simply
38 by sending a “confirmed” or “unconfirmed” uplink, possibly without applicative payload.

39 The end-device has the choice between 2 basic strategies:

- 1 • Systematic periodic uplink: simplest method that doesn't require demodulation of the
2 "gateway specific" field of the beacon. Only applicable to slowly moving or stationery
3 end-devices. There are no requirements on those periodic uplinks.
4 • Uplink on cell change: The end-device demodulates the "gateway specific" field of
5 the beacon, detects that the ID of the gateway broadcasting the beacon it
6 demodulates has changed, and sends an uplink. In that case the device should
7 respect a pseudo random delay in the [0:120] seconds range between the beacon
8 demodulation and the uplink transmission. This is required to insure that the uplinks
9 of multiple Class B devices entering or leaving a cell during the same beacon period
10 will not systematically occur at the same time immediately after the beacon
11 broadcast.
- 12 Failure to report cell change will result in Class B downlink being temporary not operational.
13 The network server may have to wait for the next end-device uplink to transmit downlink
14 traffic.
15
16

1 **16 Class B unicast & multicast downlink channel frequencies**

2 **16.1 EU 863-870MHz ISM Band**

3 All unicast&multicastClass B downlinks use a single frequency channel defined by the
4 “**PingSlotChannelReq**” MAC command. The default frequency is 869.525MHz

5 **16.2 US 902-928MHz ISM Band**

6 By default Class B downlinks use a channel function of the Time field of the last beacon (see
7 Beacon Frame content) and the DevAddr.

$$\text{Class B downlink channel} = \left[\text{DevAddr} + \text{floor} \left(\frac{\text{Beacon_Time}}{\text{Beacon_period}} \right) \right] \text{ modulo } 8$$

- 8 • Whereby Beacon_Time is the 32 bit Time field of the current beacon period
- 9 • Beacon_period is the length of the beacon period (defined as 128sec in the
10 specification)
- 11 • Floor designates rounding to the immediately lower integer value
- 12 • DevAddr is the 32 bits network address of the device

13 Class B downlinks therefore hop across 8 channels in the ISM band and all Class B end-
14 devices are equally spread amongst the 8 downlink channels.

15 If the “**PingSlotChannelReq**” command with a valid non-zero argument is used to set the
16 Class B downlink frequency then all subsequent ping slots should be opened on this single
17 frequency independently of the last beacon frequency.

18 If the “**PingSlotChannelReq**” command with a zero argument is sent, the end-device
19 should resume the default frequency plan, id Class B ping slots hopping across 8 channels.

20 The underlying idea is to allow network operators to configure end-devices to use a single
21 proprietary dedicated frequency band for the Class B downlinks if available, and to keep as
22 much frequency diversity as possible when the ISM band is used.

23

1

CLASS C – CONTINUOUSLY LISTENING

1 **17 Class C: Continuously listening end-device**

2 The end-devices implanting the Class C option are used for applications that have sufficient
3 power available and thus do not need to minimize reception time.

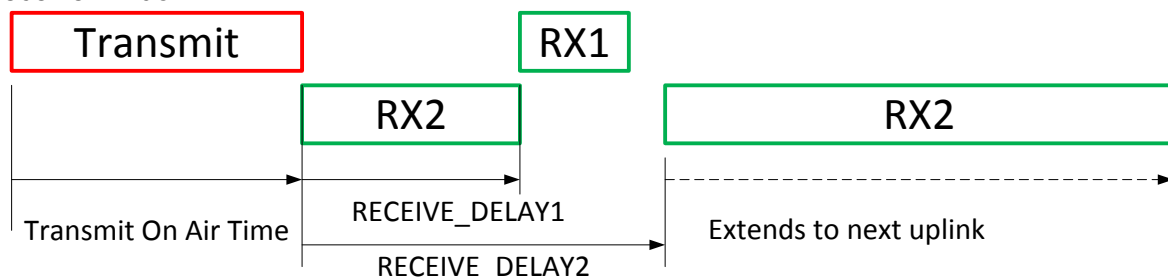
4 Class C end-devices cannot implement Class B option.

5 The Class C end-device will listen with RX2 windows parameters as often as possible. The
6 end-device listens on RX2 when it is not either (a) sending or (b) receiving on RX1,
7 according to Class A definition. To do so, it will open a short window on RX2 parameters
8 between the end of the uplink transmission and the beginning of the RX1 reception window
9 and it will switch to RX2 reception parameters as soon as the RX1 reception window is
10 closed; the RX2 reception window will remain open until the end-device has to send another
11 message.

12 **Note:** There is not specific message for a node to tell the server that it
13 is a Class C node. It is up to the application on server side to know that
14 it manages Class C nodes based on the contract passed during the
15 join procedure.

16 **17.1 Second receive window duration for Class C**

17 Class C devices implement the same two receive windows as Class A devices, but they do
18 not close RX2 window until they need to send again. Therefore they may receive a downlink
19 in the RX2 window at nearly any time. A short listening window on RX2 frequency and data
20 rate is also opened between the end of the transmission and the beginning of the RX1
21 receive window.



22 **Figure 14: Class C end-device receive slot timing.**

24 **17.2 Class C Multicast downlinks**

25 Similarly to Class B, Class C devices may receive multicast downlink frames. The multicast
26 address and associated network session key and application session key must come from
27 the application layer. The same limitations apply for Class C multicast downlink frames:

- 28 • They are not allowed to carry MAC commands, neither in the **FOpt** field, nor in the
29 payload on port 0 because a multicast downlink does not have the same
30 authentication robustness as a unicast frame.
- 31 • The **ACK** and **ADRACKReq** bits must be zero. The **MType** field must carry the value
32 for Unconfirmed Data Down.
- 33 • The **FPending** bit indicates there is more multicast data to be sent. Given that a
34 Class C device keeps its receiver active most of the time, the **FPending** bit does not
35 trigger any specific behavior of the end-device.

SUPPORT INFORMATION

1
2
3

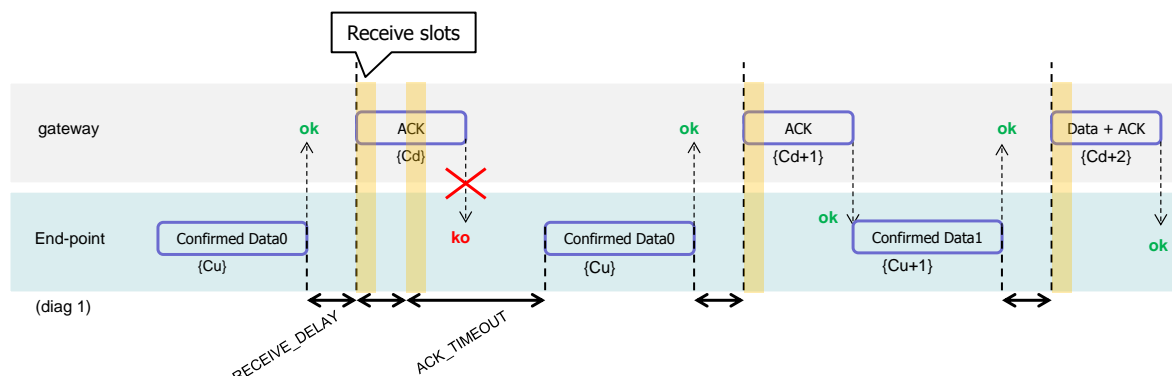
This sub-section is only a recommendation.

1 **18 Examples and Application Information**

2 Examples are illustrations of the LoRaWAN spec for information, but they are not part of the
3 formal specification.

4 **18.1 Uplink Timing Diagram for Confirmed Data Messages**

5 The following diagram illustrates the steps followed by an end-device trying to transmit two
6 confirmed data frames (Data0 and Data1):
7



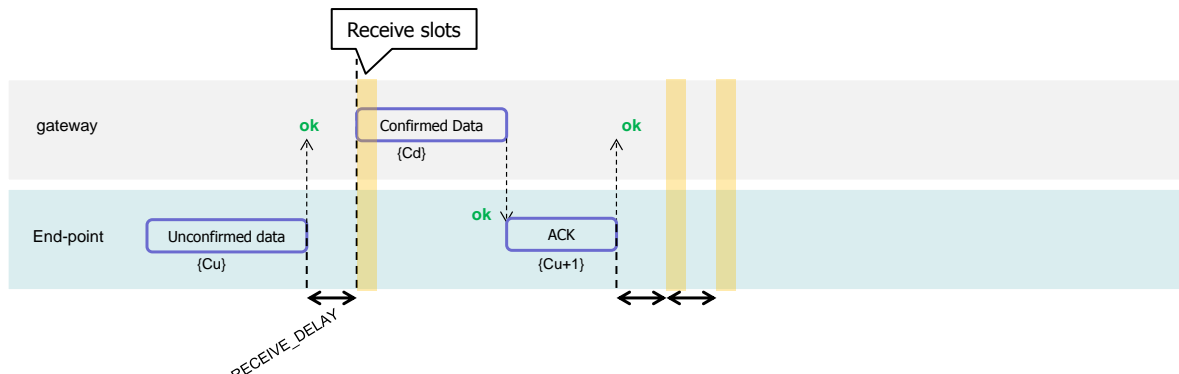
8
9 **Figure 15: Uplink timing diagram for confirmed data messages**

10 The end-device first transmits a confirmed data frame containing the Data0 payload at an
11 arbitrary instant and on an arbitrary channel. The frame counter Cu is simply derived by
12 adding 1 to the previous uplink frame counter. The network receives the frame and
13 generates a downlink frame with the ACK bit set exactly RECEIVE_DELAY1 seconds later,
14 using the first receive window of the end-device. This downlink frame uses the same data
15 rate and the same channel as the Data0 uplink. The downlink frame counter Cd is also
16 derived by adding 1 to the last downlink towards that specific end-device. If there is no
17 downlink payload pending the network shall generate a frame without a payload. In this
18 example the frame carrying the ACK bit is not received.

19 If an end-point does not receive a frame with the ACK bit set in one of the two receive
20 windows immediately following the uplink transmission it may resend the same frame with
21 the same payload and frame counter again at least ACK_TIMEOUT seconds after the
22 second reception window. This resend must be done on another channel and must obey the
23 duty cycle limitation as any other normal transmission. If this time the network receives the
24 ACK downlink during its first receive window, as soon as the ACK frame is demodulated, the
25 end-device is free to transmit a new frame on a new channel.

26 **18.2 The third ACK frame in this example also carries an application**
27 **payload. A downlink frame can carry any combination of ACK, MAC**
28 **control commands and payload. Downlink Diagram for Confirmed**
29 **Data Messages**

30 The following diagram illustrates the basic sequence of a “confirmed” downlink.
31
32



1
2 **Figure 16: Downlink timing diagram for confirmed data messages**

3 The frame exchange is initiated by the end-device transmitting an “unconfirmed” application
4 payload or any other frame on channel A. The network uses the downlink receive window to
5 transmit a “confirmed” data frame towards the end-device on the same channel A. Upon
6 reception of this data frame requiring an acknowledgement, the end-device transmits a
7 frame with the ACK bit set at its own discretion. This frame might also contain piggybacked
8 data or MAC commands as its payload. This ACK uplink is treated like any standard uplink,
9 and as such is transmitted on a random channel that might be different from channel A.

10 **Note:** To allow the end-devices to be as simple as possible and have
11 keep as few states as possible it may transmit an explicit (possibly
12 empty) acknowledgement data message immediately after the
13 reception of a data message requiring an acknowledgment.
14 Alternatively the end-device may defer the transmission of an
15 acknowledgement to piggyback it with its next data message.

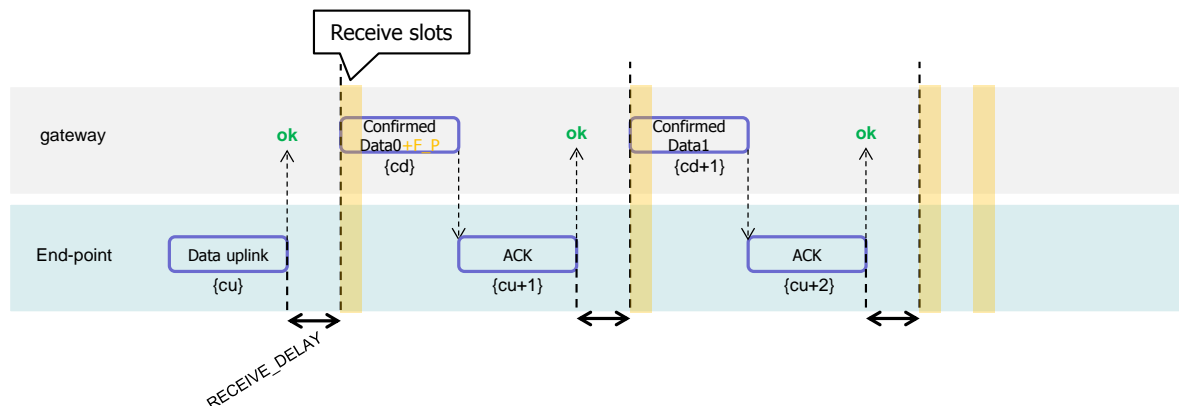
16 **18.3 Downlink Timing for Frame-Pending Messages**

17 The next diagram illustrates the use of the **frame pending** (FPending) bit on a downlink.
18 The FPending bit can only be set on a downlink frame and informs the end-device that the
19 network has several frames pending for him; the bit is ignored for all uplink frames.

20 If a frame with the FPending bit set requires an acknowledgement, the end-device shall do
21 so as described before. If no acknowledgment is required, the end-device may send an
22 empty data message to open additional receive windows at its own discretion, or wait until it
23 has some data to transmit itself and open receive windows as usual.

24 **Note:** The FPending bit is independent to the acknowledgment
25 scheme.

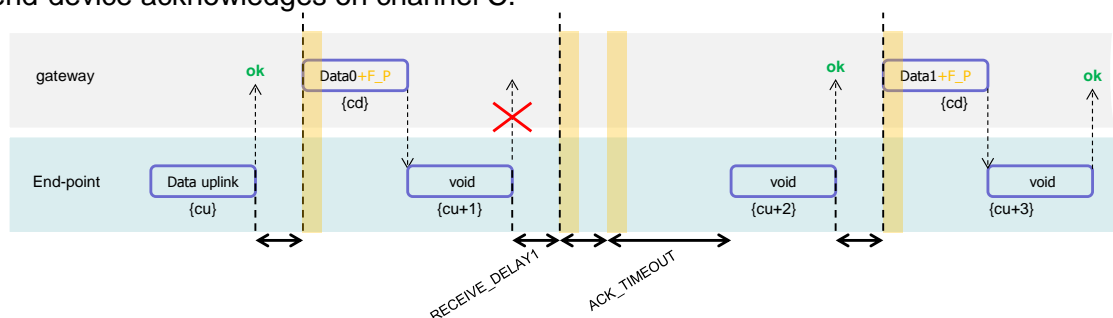
(*) F_P means 'frame pending' bit set



26

1 **Figure 17: Downlink timing diagram for frame-pending messages, example 1**

2 In this example the network has two confirmed data frames to transmit to the end-device.
 3 The frame exchange is initiated by the end-device via a normal “unconfirmed” uplink
 4 message on channel A. The network uses the first receive window to transmit the Data0 with
 5 the bit FPending set as a confirmed data message. The device acknowledges the reception
 6 of the frame by transmitting back an empty frame with the ACK bit set on a new channel B.
 7 RECEIVE_DELAY1 seconds later, the network transmits the second frame Data1 on
 8 channel B, again using a confirmed data message but with the FPending bit cleared. The
 9 end-device acknowledges on channel C.

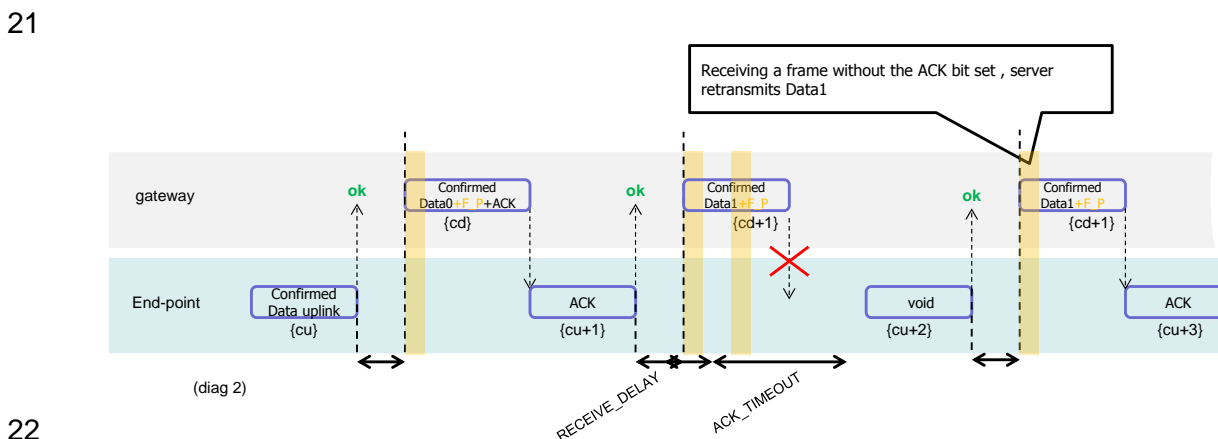


10 **Figure 18: Downlink timing diagram for frame-pending messages, example 2**

12 In this example, the downlink frames are “unconfirmed” frames, the end-device does not
 13 need to send back and acknowledge. Receiving the Data0 unconfirmed frame with the
 14 FPending bit set the end-device sends an empty data frame. This first uplink is not received
 15 by the network. If no downlink is received during the two receive windows, the network has
 16 to wait for the next spontaneous uplink of the end-device to retry the transfer. The end-
 17 device can speed up the procedure by sending a new empty data frame.

18 **Note:** An acknowledgement is never sent twice.

19 The FPending bit, the ACK bit, and payload data can all be present in the same downlink.
 20 For example, the following frame exchange is perfectly valid.



22 **Figure 19: Downlink timing diagram for frame-pending messages, example 3**

24 The end-device sends a “confirmed data” uplink. The network can answer with a confirmed
 25 downlink containing Data + ACK + “Frame pending” then the exchange continues as
 26 previously described.

1 18.4 Data-Rate Adaptation during Message Retransmissions

2 When an end-device attempts the transmission of a “confirmed” frame toward the network it
 3 expects to receive an acknowledgement in one of the subsequent reception slot. In the
 4 absence of the acknowledgement it will try to re-transmit the same data again. This re-
 5 transmission happens on a new frequency channel, but can also happen at a different data
 6 rate (preferable lower) than the previous one. It is strongly recommended to adopt the
 7 following re-transmission strategy.

8 The first transmission of the “confirmed” frame happens with a data rate DR.
 9

| Transmission nb | Data Rate |
|-----------------|-----------------------|
| 1 (first) | DR |
| 2 | DR |
| 3 | $\max(\text{DR}-1,0)$ |
| 4 | $\max(\text{DR}-1,0)$ |
| 5 | $\max(\text{DR}-2,0)$ |
| 6 | $\max(\text{DR}-2,0)$ |
| 7 | $\max(\text{DR}-3,0)$ |
| 8 | $\max(\text{DR}-3,0)$ |

10 The Data Rate $\max(a,b)$ stands for maximum of a and b values.

11 If after a recommended 8 transmissions, the frame has not been acknowledged the MAC
 12 layer should return an error code to the application layer.

13 **Note:** For each re-transmission, the frequency channel is selected
 14 randomly as for normal transmissions.

15 Any further transmission uses the last data rate used.

16 For example if an end-device sends a “confirmed” frame first using DR5 and has to
 17 retransmit 3 times (twice at DR5 then twice at DR4), the next frame transmitted will use DR4

18 Other example, if an end-device sends a “confirmed” frame first using DR5 and does not
 19 receive an acknowledge after 8 transmissions (2 at DR5, 2 at DR4, ... , 2 at DR2), and the
 20 application of this end-device re-initiates a “confirmed” transmission a little later, the first two
 21 transmission will be tentatively at DR2, then switch to DR1, then to DR0.

1 **19 Recommendation on contract to be provided to the network**
2 **server by the end-device provider at the time of provisioning**

3 Configuration data related to the end-device and its characteristics must be known by the
4 network server at the time of provisioning. –This provisioned data is called the “contract”.
5 This contract cannot be provided by the end-device and must be supplied by the end-device
6 provider using another channel (out-of-band communication).

7 This end-device contract is stored in the network server. It can be used by the application
8 server and the network controller to adapt the algorithms.

9 This data will include:

- 10 • End-device specific radio parameters (device frequency range, device maximal
11 output power, device communication settings - RECEIVE_DELAY1,
12 RECEIVE_DELAY2)
- 13 • Application type (Alarm, Metering, Asset Tracking, Supervision, Network Control)

1 20 Recommendation on finding the locally used channels

2 End-devices that can be activated in territories that are using different frequencies for
3 LoRaWAN will have to identify what frequencies are supported for join message at their
4 current location before they send any message. The following methods are proposed:

- 5 • A GPS enabled end-device can use its GPS location to identify which frequency
6 band to use.
- 7 • End-device can search for a beacon and use its frequency to identify its region
- 8 • End-device can search for a beacon and if this one is sending the antenna GPS
9 coordinate, it can use this to identify its region
- 10 • End-device can search for a beacon and if this one is sending a list of join
11 frequencies, it can use this to send its join message

1 21 Revisions**2 21.1 Revision 1.0**

- 3 • Draft version of LoRaWAN

4

1 22 Glossary

| | | |
|----|----------|---|
| 2 | | |
| 3 | ADR | Adaptive Data Rate |
| 4 | AES | Advanced Encryption Standard |
| 5 | AFA | Adaptive Frequency Agility |
| 6 | AR | Acknowledgement Request |
| 7 | CBC | Cipher Block Chaining |
| 8 | CMAC | Cipher-based Message Authentication Code |
| 9 | CR | Coding Rate |
| 10 | CRC | Cyclic Redundancy Check |
| 11 | DR | Data Rate |
| 12 | ECB | Electronic Code Book |
| 13 | ETSI | European Telecommunications Standards Institute |
| 14 | EIRP | Equivalent Isotropically Radiated Power |
| 15 | FSK | Frequency Shift Keying modulation technique |
| 16 | GPRS | General Packet Radio Service |
| 17 | HAL | Hardware Abstraction Layer |
| 18 | IP | Internet Protocol |
| 19 | LBT | Listen Before Talk |
| 20 | LoRa™ | Long Range modulation technique |
| 21 | LoRaWAN™ | Long Range Network protocol |
| 22 | MAC | Medium Access Control |
| 23 | MIC | Message Integrity Code |
| 24 | RF | Radio Frequency |
| 25 | RFU | Reserved for Future Usage |
| 26 | Rx | Receiver |
| 27 | RSSI | Received Signal Strength Indicator |
| 28 | SF | Spreading Factor |
| 29 | SNR | Signal Noise Ratio |
| 30 | SPI | Serial Peripheral Interface |
| 31 | SSL | Secure Socket Layer |
| 32 | Tx | Transmitter |
| 33 | USB | Universal Serial Bus |
| 34 | | |

1 **23 Bibliography**

2 **23.1 References**

- 3 [IEEE802154]: IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-
- 4 Rate Wireless Personal Area Networks (LR-WPANs), IEEE Std 802.15.4TM-2011 (Revision
- 5 of IEEE Std 802.15.4-2006), September 2011.
- 6 [RFC4493]: The AES-CMAC Algorithm, June 2006.

1 **24 NOTICE OF USE AND DISCLOSURE**

2 Copyright © LoRa Alliance, Inc. (2015). All Rights Reserved.

3 The information within this document is the property of the LoRa Alliance (“The Alliance”) and its use and
4 disclosure are subject to LoRa Alliance Corporate Bylaws, Intellectual Property Rights (IPR) Policy and
5 Membership Agreements.

6 Elements of LoRa Alliance specifications may be subject to third party intellectual property rights, including
7 without limitation, patent, copyright or trademark rights (such a third party may or may not be a member of LoRa
8 Alliance). The Alliance is not responsible and shall not be held responsible in any manner for identifying or failing
9 to identify any or all such third party intellectual property rights.

10 This document and the information contained herein are provided on an “AS IS” basis and THE ALLIANCE
11 DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO (A) ANY
12 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OF THIRD
13 PARTIES (INCLUDING WITHOUT LIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING
14 PATENT, COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF
15 MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NONINFRINGEMENT.

16 IN NO EVENT WILL THE ALLIANCE BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS
17 OF USE OF DATA, INTERRUPTION OF BUSINESS, OR FOR ANY OTHER DIRECT, INDIRECT, SPECIAL OR
18 EXEMPLARY, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR
19 IN TORT, IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF
20 ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

21 The above notice and this paragraph must be included on all copies of this document that are made.

22 LoRa Alliance, Inc.

23 2400 Camino Ramon, Suite 375

24 San Ramon, CA 94583

25 *Note: All Company, brand and product names may be trademarks that are the sole property of their respective*
26 *owners.*