LoRaWAN Application Layer Clock Synchronization Specification v1.0.0

NOTICE OF USE AND DISCLOSURE

LoRa Alliance™
5177 Brandin Court
Fremont, CA 94538
United States

LoRaWAN Application Layer Clock Synchronization Specification

Authored by the FUOTA Working Group of the LoRa Alliance Technical Committee

Technical Committee Chairs:
N.SORNIN (Semtech), A.YEGIN (Actility)

Working Group Chairs:
J.CATALANO (Kerlink), N.SORNIN (Semtech)

Editor:
J.CATALANO (Kerlink)

Contributors:
J.CATALANO (Kerlink), J-P.COUPIGNY (STMicroelectronics), N.SORNIN (Semtech), J.STOKKING (The Things Network Foundation)

Version: v1.0.0
Date: September 10, 2018
Status: Final release

Contents

Tables

1 Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", “MAY", and “OPTIONAL" in this document are to be interpreted as described in RFC 2119.

The octet order over the air for all multi-octet fields is little endian (Least significant byte is sent first).

2 Introduction

This document proposes an application layer messaging package running over LoRaWAN to synchronize the real-time clock of an end-device to the network's GPS clock with second accuracy. Synchronizing the end-device(s) clock is very useful of many applications like:

- Get all end-devices of a multicast group switching to classC temporarily and synchronously at the beginning of the slot
- Get many sensors to synchronously perform a measurement (get water meter reading of all meters at midnight every day for example)
- Enabling end-devices to transmit time-stamped events (the door was opened this morning at 8:00AM) with a unified clock

This package is useful for end-devices which do not have access to other accurate time source. An end-device using LoRaWAN 1.1 or above SHOULD use DeviceTimeReq MAC command instead of this package. ClassB end-devices have a more efficient way of synchronizing their clock, the classB network beacon. They SHOULD NOT use this package and directly use the beacon time information. End-devices with an accurate external clock source (e.g.: GPS) SHOULD use that clock source instead.

All messages described in this document are transported as application layer messages. As such, all unicast messages (uplink or downlink) are encrypted by the LoRaWAN MAC layer using the end-device's AppSKey.

The package uses a dedicated port to separate its traffic from the rest of the applicative traffic.

3 Clock synchronization Message Package

The identifier of the clock synchronization package is 1. The version of this package is version 1.

The following messages are sent to/from each end-device individually using Unicast uplink or downlink on a port specifically used for the clock synchronization package. The default port value is 202. These messages MUST NOT be sent using multicast. If these messages are received on a multicast address the end-device MUST drop them silently.

All unicast control messages use the same format:

Command1	Command1 Payload	Command2	Command2 payload	….

A message MAY carry more than one command. The length of each command's payload is fixed and a function of the command. Commands are executed from first to last. Each command MUST be individually acknowledged by the end-device.

The following table summarizes the list of the clock synchronization messages

CID	Command name	Transmitted by		Short Description
		End-device	server	
0x00	***PackageVersionReq***		x	Used by the AS to request the package version implemented by the end-device
0x00	***PackageVersionAns***	x		Conveys the answer to PackageVersionReq
0x01	***AppTimeReq***	x		Used by end-device to request clock correction
0x01	***AppTimeAns***		x	Conveys the clock timing correction
0x02	***DeviceAppTimePeriodicityReq***		x	Used by the application server for 2 purposes: Set the periodicity at which the end-device shall transmit AppTimeReq messages and request an immediate transmission of end-device time
0x02	***DeviceAppTimePeriodicityAns***	x		
0x03	***ForceDeviceResyncReq***		x	Used by the application server to the end-device to trigger a clock resynchronization.

Table 1: Clock Synchronization messages summary

3.1 PackageVersionReq & Ans

The ***PackageVersionReq*** command has no payload.
The end-device answers with a ***PackageVersionAns*** command with the following payload.

Field	PackageIdentifier	PackageVersion
Size (bytes)	1	1

Table 2: PackageVersionAns

PackageIdentifier uniquely identifies the package. For the “clock synchronization package” this identifier is 1.
PackageVersion corresponds to the version of the package specification implemented by the end-device.

3.2 AppTimeReq & Ans

The **AppTimeReq** message is transmitted by the end-device to request a clock correction from the application server. The message is meant to be transmitted periodically by the end-device. The default periodicity is a function of the accuracy required by the application and the maximum clock drift speed of the end-device.
This message SHALL only be transmitted a single time with a given DeviceTime payload, as the network reception time stamp will be used by the application server to compute the require clock correction. Therefore the “clock synchronization” package SHALL first temporarily disable ADR and set NbTrans=1 before transmitting this message, then revert the MAC layer to the previous state.

The **AppTimeReq** command has the following payload.

Field	DeviceTime	Param
Size (bytes)	4	1

Table 3: AppTimeReq

Where*:*

Param Fields	RFU	AnsRequired	TokenReq
Size (bits)	3bits	1bit	4bits

Table 4: Param fields

DeviceTime is the current end-device clock and is expressed as the time in seconds since 00:00:00, Sunday 6th of January 1980 (start of the GPS epoch) modulo 2^32. Note that this is the same format as the Time field in the beacon frame. The time is captured immediately before transmitting the radio packet. The processing delay between the clock time capture and the transmission of the packet should be minimized. The intent is to provide second accurate timing therefore the delay SHALL be < 250mSec.

TokenReq is a 4 bits counter initially set to 0. TokenReq is incremented (modulo 16) each time the end-device receives and processes successfully an AppTimeAns message.

If the *AnsRequired* bit is set to 1 the end-device expects an answer whether its clock is well synchronized or not. If this bit is set to 0, this signals to the AS that it only needs to answer if the end-device clock is de-synchronized.

The application server MAY respond to the **AppTimeReq** command with an **AppTimeAns** with the following payload:

Field	TimeCorrection	Param
Size (bytes)	4	1

Table 5: AppTimeAns

Where*:*

Param Fields	RFU	TokenAns
Size (bits)	4bits	4bits

Table 6: Param fields

TimeCorrection is a signed 32-bit integer, stipulating the time delta correction in seconds.

If the *AnsRequired* bit is 0 the application server MAY respond if the end-device indicated current clock timing drifts above a certain application specific threshold. If the end-device's clock is well synchronized, the application server does not need to answer. The application server uses the network time stamp of the uplink frame to compute the required timing correction.
If the *AnsRequired* bit is 1 the application server SHOULD respond to the **AppTimeReq** command. Not responding to the end-device very probably triggers a retransmission of AppTimeReq by the end-device until it receives an answer. This retransmission strategy is application specific.

When the application server answers:
TokenAns MUST match the *TokenReq* value of the AppTimeReq message which is being answered. If the *TokenAns* & *TokenReq* fields do not match the end-device SHALL ignore the AppTimeAns message.

If the two tokens match, then the end-device SHALL increment its TokenReq internal counter (modulo 16) and TimeCorrection MUST be added to the current end-device clock to be synchronous with the network clock. The end-device SHALL immediately perform the correction on its clock. Any following transmission of the **AppTimeReq** message SHALL reflect the timing correction and the incremented *TokenReq* value to avoid unnecessary downlinks.

3.3 DeviceAppTimePeriodicityReq & Ans

Each end-device's application MAY come with a different default periodicity for the transmission of the **AppTimeReq** message.

The **DeviceAppTimePeriodicityReq** command is used by the application server to modify this periodicity and/or get an instant reading of the end-device's clock value. The message payload is:

Field	Periodicity
Size (bytes)	1

Table 7: DeviceAppTimePeriodicityReq

Where:

Periodicity Fields	RFU	Period
Size (bits)	4bits	4bits

Table 8: DeviceAppTimePeriodicityReq Periodicity field

Period encodes the periodicity of the **AppTimeReq** transmissions. The actual periodicity in seconds is $128.2^{Period} \pm rand(30)$ where $rand(30)$ is a random integer in the +/-30sec range varying with each transmission.

The end-device responds with the **DeviceAppTimePeriodicityAns** message containing the following payload.

Field	Status	Time
Size (bytes)	1	4

Table 9: DeviceAppTimePeriodicityAns

Where:

Status Fields	RFU	NotSupported
Size (bits)	7bits	1bit

Table 10: DeviceAppTimePeriodicityAns Status field

NotSupported bit is set to 1 if the end-device's application does not accept a periodicity set by the application server and manages the clock synchronization process and periodicity itself.

Time is the current end-device's clock time captured immediately before the transmission of the radio message.

3.4 ForceDeviceResyncReq

The **ForceDeviceResyncReq** message is transmitted by the application server to the end-device to trigger a clock resynchronization.

> An example of condition that may trigger this transmission is the McClassCSessionAns message sent by the end-device in response to a classC setup command from the application server. If the server detects that the end-device's clock is not well synchronized it should force the end-device to re-synchronize its clock else the end-device will miss the multicast slot.

272 The **ForceDeviceResyncReq** command has a single byte payload.
273

Field	ForceConf
Size (bytes)	1

274 **Table 11: ForceDeviceResyncReq**

275 Where*:*
276

ForceConf Fields	RFU	NbTransmissions
Size (bits)	5bits	3bits

277 **Table 12: ForceConf fields**

278 There is no **ForceDeviceResyncAns** message. The end-device responds by sending up to
279 *NbTransmissions* **AppTimeReq** messages with the *AnsRequired* bit set to 0. The end-
280 device stops re-transmissions of the **AppTimeReq** if a valid **AppTimeAns** is received. If the
281 *NbTransmissions* field is 0, the command SHALL be silently discarded.
282 The delay between consecutive transmissions of the **AppTimeReq** is application specific.
283

4 Glossary

AS	Application Server
GPS	Global Positioning System
TBD	To Be Done

5 Bibliography

5.1 References

[LoRaWAN 1.0.2]: LoRaWAN™ 1.0.2 Specification, LoRa Alliance, July 2016

[LoRaWAN 1.1]: LoRaWAN™ 1.1 Specification, LoRa Alliance, October 11, 2017

6 NOTICE OF USE AND DISCLOSURE

LoRa Alliance™
5177 Brandin Court
Fremont, CA 94538
United States