LoRaSync: energy efficient synchronization for scalable LoRaWAN

Laurent Chasserat, Nicola Accettura, Pascal Berthou

LoRaSync: energy efficient synchronization for scalable LoRaWAN

Laurent Chasserat, Nicola Accettura and Pascal Berthou
LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France
Email: {firstname.lastname}@laas.fr

***Abstract*—Low-Power Wide-Area Networks (LPWAN) connect a large number of battery-powered wireless devices over long distances. Among them, Long Range Wide Area Networks (LoRaWAN) implement a Pure ALOHA medium access scheme to save device energy by minimizing the radio usage. However, frame collisions restrain the network scalability when the traffic load increases. In this context, synchronization is used to exploit the available bandwidth more efficiently by controlling the timing of frame transmissions and reducing the collision probability. Remarkably, the network throughput is increased at the cost of an extra energy demand due to the inherent overhead. In that, the entailed network scenario is still supposed to address low power applications. This paper timely presents *LoRaSync*, an energy-efficient synchronization scheme designed for LoRa networks of any size. An accurate clock drift model was established based on measurements made on real cheap devices, and leveraged to support the design of *LoRaSync*. Our mechanism has been used to evaluate the same ALOHA-based random access but on a time-slotted basis, thus increasing the maximum achievable throughput compared to the legacy access. Throughput and energy efficiency models are established to evaluate the performances of a *LoRaSync*-operated network. These models are validated with a simulation environment mimicking large-scale deployments, and then used to determine the most energy efficient slot size for any traffic load. As a final proof of concept, *LoRaSync* has been implemented and tested on a LoRa testbed to demonstrate the feasibility of our solution on real hardware.**

***Index Terms*—LoRaWAN, LPWAN, MAC Protocols, Energy Efficiency, Scalability, Machine-to-Machine Communications, Ad-Hoc Networks, IoT**

I. Introduction

By providing cheap devices with low power and long range communication capabilities [1], Low Power Wide Area Networks (LPWAN) keep gaining relevance in industrial and research environments. In particular, Long Range Wide Area Networks (LoRaWANs) have emerged as a very promising technology to support a wide range of distributed sensing applications [2]. The increasing availability of cheap wireless devices and the consequent growth of communication traffic raises scalability issues that have triggered a great research interest [3]–[7].

The reason behind LoRaWAN's scalability limitations lies in the simplicity of its medium access scheme. Indeed, to send information through the Internet, a node forges and immediately transmits a link layer frame without checking if the radio channel is free. This simple access scheme is the very well-known Pure ALOHA MAC (Medium Access Control) protocol [8]. Herein the channel throughput is limited to a maximum of 18% of the available bandwidth due to frame collisions.

Given that this strategy insures very poor network performances for high traffic loads, synchronization mechanisms have been explored as means to reduce the frame collision probability and increase the maximum LoRaWAN throughput [9], [10]. As a matter of fact, leveraging time synchronous slots to setup a very simple Slotted ALOHA random access scheme [11] doubles the maximum achievable throughput compared to its basic asynchronous version (*i.e.*, Pure ALOHA). More interestingly, a synchronization mechanism allows the development of even more sophisticated MAC layer schemes capable of handling various traffic requirements. For instance, a scheduled access could be designed to insure even collision-free communications. In order to share a common time reference across the network, all devices need to periodically receive timing information to cope with the natural drift of their embedded clock. To do so, they need to frequently switch on their radio to listen to incoming frames. This technique consumes a considerable amount of the available energy in the feeding batteries. Therefore, energy efficiency should be prioritized when designing the synchronization mechanism.

With this approach in mind, we introduced *Class S* [18] as an extension of the LoRaWAN *Class B*, leveraging a beacon-based synchronization scheme to setup uplink transmission timeslots. Simulations showed that this solution could enable a slotted access over LoRa. However, a beacon skipping mechanism was required to limit the device radio usage and ensure energy efficiency. In this sense, we timely present the *LoRaSync* synchronization scheme, as a means to implement *Class S* on real cheap low power devices. More specifically, we measured the clock skew of typical LoRa hardware to evaluate how fast devices drift away from a given time reference. Based on this preliminary experimental evaluation, we designed *LoRaSync* with slots large enough to cope with clock errors. In that, the slot size and the maximum clock drift are used to compute the maximum amount of beacons that may be safely skipped while keeping devices synchronized. Such a beacon skipping mechanism is crucial to minimize the power consumption due to frame receptions, and eventually maximize the lifetime of feeding batteries.

In order to assess the performances of large-scale *LoRaSync* networks, throughput and energy efficiency models have been established. These models notably account for slot size variations, reception slot enlargements and the proposed

Synchronization mechanism	Synchronization strategy	In-band synchronization	Scalable to any number of devices	Real-hardware implementation	LoRaWAN compliant
Piyare *et al.* [12]	Low-power wake-up receivers		x	x	
Beltramelli *et al.* [13]	FM-RDS signals		x	x	
Polonelli *et al.* [9]	Acknowledgements	x		x	x
Haxhibeqiri, Garrido *et al.* [10] [14]	Acknowledgements			x	x
Abdelfadeel *et al.* [7]	Beaconing	x	x		
Reynders *et al.* [15]	Beaconing	x	x		
Zorbas *et al.* [16]	Beaconing	x	x	x	
Tessaro *et al.* [17]	Beaconing	x	x	x	
LoRaSync	Beaconing	x	x	x	x

TABLE I: Comparison of related works about LoRa synchronization

beacon skipping mechanism. They additionally have been validated through the LoRaWAN-sim simulation environment. Furthermore, the impact of the *LoRaSync* slot size on the overall energy efficiency of the network has been evaluated. Remarkably, the slot size has a twofold impact on the network behavior. On the one hand, increasing the slot length reduces the number of slots that fit into the slotframe, thus reducing the maximum achievable throughput. On the other hand, using large slots also allows devices to stay synchronized for longer time periods without receiving beacons, thus reducing their overall power consumption. As a consequence, we discovered that setting up the proper slot size value is functional to fit the best trade-off between throughput and power consumption. We therefore provide a methodology to determine the most energy efficient slot size for any traffic load.

In order to further prove the soundness of the conceived *LoRaSync* mechanism, a real-world testbed has been setup and used to implement our synchronization scheme. This proof-of-concept demonstrates that the timing error can successfully be controlled despite the low-quality clocks found on such devices.

From these premises, this contribution is organized as follows. Section II describes some related works about synchronization in LoRa networks, while showing how *LoRaSync* introduces for the first time an energy-efficient and scalable synchronized access for LoRa communications. Then, Section III introduces the *LoRaSync* design, based on real clock skew measurements. Section IV presents and validates models for the throughput, power consumption and energy efficiency in *LoRaSync*-enabled networks, that are then leveraged to determine the most energy efficient slot sizes. For the sake of completeness, a preliminary evaluation of *LoRaSync* in real LoRa networks is presented in Section V through the description of a proof-of-concept implementation. Finally, Section VI discloses concluding remarks and research perspectives.

II. Related works

A LoRaWAN network relies on gateways to gather frames transmitted by low-power end devices and forward them to a data collection server. The research community has raised strong concerns regarding the scalability of such deployments, leveraging propagation models [19] and extensive simulation campaigns [5]. Indeed, the simplistic Pure ALOHA access scheme used in LoRaWAN dwarfs the network capacity. Therefore, the following contributions have proposed MAC layer improvements to increase the maximum achievable throughput, leveraging time-synchronization to make better use of the available bandwidth.

A first approach to seamlessly synchronize devices without impacting the LoRa communications is the use of out-of-band communications. In [12], Piyare *et. al.* propose an on-demand Time Division Multiple Access (TDMA) scheme that leverages another low-power communication technology (*i.e.*, micro-Watt wake-up receivers) to activate all devices and provide them with a common time reference before starting LoRa data exchanges. Similarly, Beltramelli *et. al.* [13] propose the use of FM-RDS signals to achieve synchronization in LoRa networks. The main drawback of these strategies is that they need LoRa devices to be equipped with additional hardware components, thus making such a solution not usable with current products available on the market.

Another popular strategy is to use individual acknowledgments to synchronize devices upon demand. For instance in [9], Polonelli *et. al.* implemented a Slotted ALOHA scheme on real hardware, using *Class A* reception windows to share time information through acknowledgments. In [10], the synchronization frame additionally carries a timeslot assignment schedule to organize the transmission of all devices based on their traffic needs. Notably, Bloom filters are used to reduce the size of this extended downlink frame and enhance the efficiency of this mechanism. This work is then leveraged in [14] to setup a transmission scheduling protocol for LoRaWAN. These contributions account for the clock drift of devices and are supported by real-hardware implementations. This shows the feasibility of the strategy, but also reveals that a re-synchronization is periodically required. On this regard, the gateway's Duty Cycle limitations bound the network size that can be handled by this mechanism [20]. Besides, most LoRa gateways are half-duplex [21], meaning that they cannot transmit and receive frames at the same time. In that, the downlink traffic will considerably increase if many nodes require an ack-based synchronization. The gateway will therefore be unreachable during the transmission of these messages, which will eventually affect the uplink throughput.

These issues are avoided when using beacons to advertise the network and synchronize end devices. Such an approach scales with any network size, since the reception of a beacon frame can serve as time source for all devices listening to

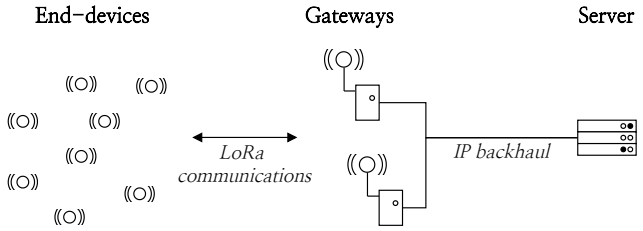


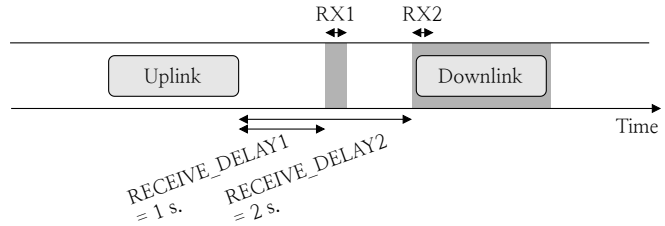
Fig. 1: LoRaWAN topology

it on the air. That is why we chose the LoRaWAN *Class B* beaconing mechanism as a basis to design *Class S* [18], an access scheme allowing slotted uplink communications. One of the key conclusions of this contribution was that a beacon-skipping mechanism was highly desired in order to save battery on the low-power LoRa devices. Several contributions [7], [15]–[17] have explored beacon-based synchronization, but all fail to address this matter. In details, [7] leverages beacons to share the settings of a fine-grained scheduling algorithm designed for bulk data collection. Alternatively, this same mechanism is used in [15] to group transmissions with similar RSSI and SF within the same timeslots. This allows to reduce the impact of the capture effect, notably improving the network fairness. With TS-LoRa [16], Zorbas *et. al.* tackle the problem of autonomous slot assignment. SACK (synchronization/acknowledgement) packets are introduced to serve both as synchronization beacon and group acknowledgement. The viability of this protocol is asserted with a real-hardware implementation, yet it introduces considerable changes to the default protocol and increases the energy consumption by 50%. Finally, beacon synchronization was once again used in [17] with a focus on industrial networks featured with periodic, predictable traffic. Clock drift measurements and corrections are notably emphasized in this last contribution.

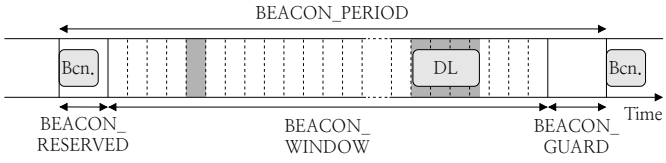
In this landscape, we therefore introduce *LoRaSync*, a scalable beaconing-based synchronization approach designed for LoRa networks that focuses on energy efficiency and requires minimal changes to the current standard. It implements a beacon-skipping mechanism relying on real clock drift measurements in order to space out the synchronization events as much as possible without creating inter-slot transmission overlaps, thus keeping the device power consumption to the minimum. To the best or our knowledge, *LoRaSync* is the first synchronization scheme designed to maximize energy efficiency, that additionally works for any network size.

III. LoRaSync requirements and design

With *LoRaSync*, we propose an energy-efficient and scalable synchronization approach tailored for LoRa deployments. It is designed to support the development of time-slotted access schemes for such networks, with an effort to minimize the device power consumption. This Section first provides relevant background on the LoRa technology and an experimental evaluation of the clock drift on low-cost LoRa devices. This information is then leveraged to present the *LoRaSync* components in details, namely *(i)* the synchronization strategy, *(ii)* a slotframe structure resilient to the device clock skew and *(iii)* the beacon-skipping mechanism designed to save battery power.



(a) LoRaWAN *Class A*



(b) LoRaWAN *Class B*

(c) LoRaWAN *Class C*

Fig. 2: LoRaWAN classes medium access

A. Background on LoRa and LoRaWAN

The LoRa LPWAN technology is well known for its capability to gather data over wide deployment areas. The term LoRa specifically designates the physical layer, while LoRaWAN is the Medium Access Control (MAC) layer that was designed on the top of it. The proprietary LoRa modulation relies on a Chirp Spread Spectrum (CSS) technique, which has proven to be resilient to multi-path interference and channel fading [22]. It allows transmissions to reach up to 10 km ranges under ideal conditions [23], with a relatively low power operation. A key parameter of this technique is the Spreading Factor (SF), that can be associated to the chirp sweeping time [24]. A bigger SF results in a longer transmission range, but also in a bigger Time on Air (ToA). High SFs are therefore featured with a low data rate and high energy consumption. In the scope of this contribution, we focus on using the smallest SF (SF7) because it results in the shortest ToA, and therefore in the minimal energy consumption.

The LoRa physical layer modulation is proprietary, however the upper layers are open and well documented [25]. In particular, the LoRa Alliance provides a common view on the MAC layer implementation through the LoRaWAN specification. This standard presents a network topology in which nodes communicate with gateways via LoRa communications, and gateways interact with servers through an IP backhaul. This architecture is represented in Figure 1.

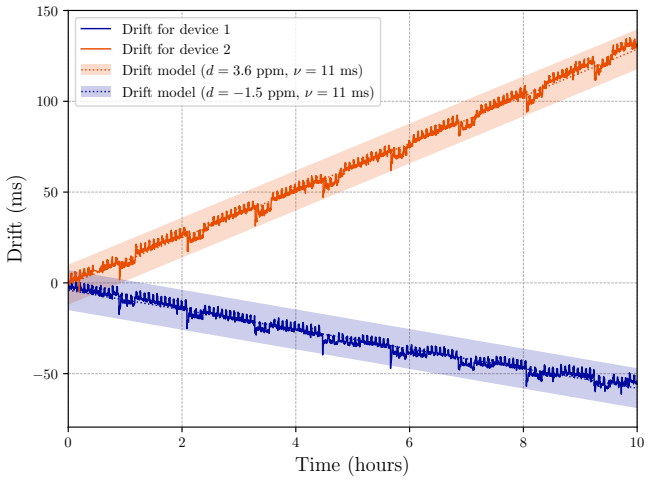


Fig. 3: Device drift measurements

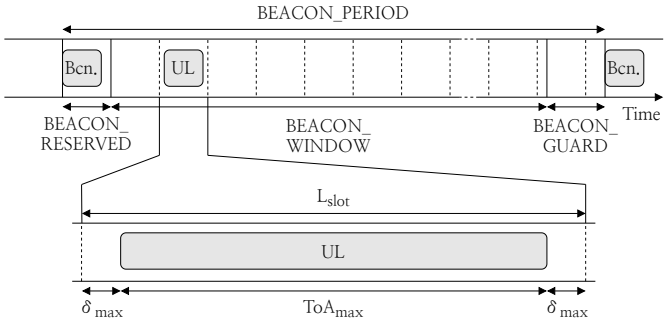


Fig. 4: LoRaSync beacon period and slot structure.

TABLE II: Slotframe intervals

Interval	Length
BEACON_PERIOD	128 s.
BEACON_RESERVED	2.12 s.
BEACON_WINDOW	122.88 s.
BEACON_GUARD	3 s.

LoRaWAN defines several device classes with specific communication modes able to fit different types of applications. *Class A* is the default scheme that all devices must implement. In this mode, the uplink (device to server) transmissions are carried-out in a Pure ALOHA manner, and followed by two 30 ms reception slots (*i.e.*, RX1 and RX2), providing the server with an opportunity to return a downlink message. This behavior is depicted in Figure 2a. In other words, if the server has a pending message for a given node, it will wait for an indeterminate time before having a transmission opening. *Class B* was then introduced to reduce the downlink delay. *Class B* devices operate as *Class A* ones for uplink transmissions, but they also periodically open additional reception slots, offering frequent downlink opportunities. To do so, they need to synchronize to the network by listening to beacons broadcast by the gateways, as shown in Figure 2b. Finally, *Class C* devices (Figure 2c) must continuously listen to possible incoming frames when they are not transmitting. This mode obviously consumes a significant amount of power and is reserved for certain critical actuators.

B. Device drift measurement and modeling

The *LoRaSync* synchronization mechanism aims at efficiently correcting the natural clock drift of cheap wireless devices to allow slotted uplink communications. Therefore, it is necessary to understand how this drift can be modeled before tackling the *LoRaSync* design. To do so, we use the testbed presented in Section V, and program LoPy devices[1] to periodically transmit their local time. By comparing the intervals between the subsequent transmissions as seen by the gateway (which benefits from a very accurate GPS time), and the estimation of the same intervals done by the devices, we are able to evaluate the variations of the timing error. This clock skew has been plot in Figure 3 for two devices.

[1]Pycom website: https://pycom.io/product/lopy4/

In most related works found in the literature [9], [13], [17], the clock drift is modeled as a linear function with the assumption that the sensor temperature is constant. This is referred to as the Simple Skew Model (SKM) [26], and it is verified in our experiment as we can see on Figure 3 that the device drift follows a linear trend on the long term. For cheap devices equipped with low-quality clock crystals, a 20 parts per million (ppm) worst-case drift coefficient is typically assumed.

However, it should also be noted that the clock skew displays a substantial noise around this overall trend, which must be estimated and is accounted for when designing the *LoRaSync* margins. This finding has been mentioned in [27] but is rarely considered in practice, even though for such cheap hardware it can result in significant errors for short time spans. As a result, we define a clock drift interval model in which d is the drift coefficient in ppm, and ν the maximum deviation from the linear model due to noise. Therefore, we are assured that a device featured with such parameters which hasn't been synchronized for a duration of t will experience a drift comprised within the interval $(d \cdot t) \pm \nu$. Such model has been plot next to the real drift on Figure 3, with $\nu = 11$ ms and d adapted to the drift coefficient of each of the two devices, respectively -1.5 and 3.6 ppm. It shows that the interval is respected during the duration of the experiment, and that this model can reasonably be used to design *LoRaSync*. Besides, the worst-case 20 ppm drift hypothesis is checked.

C. LoRaSync design

Building a slotted access requires signaling messages to be exchanged in the network to share a common time reference among all devices. To do so, *LoRaSync* exploits the LoRaWAN *Class B* beaconing mechanism. Therefore, this scheme can run on typical LoRa hardware with minimal updates. A synchronization event is defined as a tuple $(ts_{\text{bcn}}, cnt_{\text{bcn}})$, where ts_{bcn} is the beacon's UTC timestamp and cnt_{bcn} is the

device's internal counter value at the beginning of the beacon reception. Such event is saved by the device at each beacon reception. UTC timestamps are defined as the total number of microseconds elapsed since the Unix epoch. Device counters are incremented every microsecond as well, therefore it is possible to add and substract counter values to timestamps. Hence, the timestamp *ts* associated with any counter value *cnt* may be computed with the following equation:

$$ts(cnt) = ts_{\text{bcn}} + (cnt - cnt_{\text{bcn}}) \tag{1}$$

The precision of this estimator depends on both the internal clock quality and the time elapsed since the last synchronization event. Indeed, indicating the device's worst-case clock drift with d (expressed in parts per million, ppm) and ν the noise margin, the worst-case timing error *err* may be computed for any counter value *cnt* with the following equation:

$$err(cnt,\ d,\ \nu) = (cnt - cnt_{\text{bcn}}) \cdot d + \nu \tag{2}$$

This worst-case error is notably used to enlarge the width of the reception slot each time a device needs to receive a beacon. This feature allows the radio to be in a listening state when the beacon is sent, regardless of its current clock misalignment. *LoRaSync* implements *Class S* [18], meaning that the beacon window is divided into timeslots dedicated to uplink transmissions. As a means to preserve bidirectional communications, *Class S* transmissions are followed by the RX1 and RX2 reception slots just like *Class A* ones. In order to adapt *Class S* to real hardware constraints, a maximum clock offset threshold δ_{max} is defined. The slot size computation thus accounts for δ_{max} and the maximum frame Time on Air (ToA) allowed:

$$L_{\text{slot}} = ToA_{\text{max}} + 2 \cdot \delta_{\text{max}} \tag{3}$$

With such a slot size, a device may not trigger a transmission that overlaps over two different slots as long as the current clock skew of this device is smaller than δ_{max}. The associated slotframe layout within the *BEACON_PERIOD* is pictured in figure 4. It is divided in three intervals: *BEACON_RESERVED* which is dedicated to the beacon reception, *BEACON_WINDOW* in which slots are layed-out, with the last slot overlapping onto *BEACON_GUARD*. These intervals are identical to the ones used in the LoRaWAN *Class B* (*c.f.* Table II) in order to facilitate the implementation of *Class S*. According to the scheme pictured so far, the number of slots in the slotframe is:

$$n_{\text{slots}} = \left\lceil \frac{BEACON\_WINDOW}{L_{\text{slot}}} \right\rceil \tag{4}$$

A beacon-skipping mechanism has been implemented in order to ameliorate the energy efficiency of the protocol. In particular, n_{skip} is defined as the maximum number of beacons that can be skipped by a device before it needs to get synchronized again. Given the worst-case drift coefficient d, the noise margin ν and the maximum offset allowed δ_{max}, n_{skip} is equal to the biggest $k \in \mathbb{N}$ that fits the following inequality:

$$(k+1) \cdot BEACON\_PERIOD \cdot d + \nu \leq \delta_{\text{max}} \tag{5}$$

This means that increasing the maximum offset allowed (and thus the slot size) reduces the synchronization beacon periodicity, eventually decreasing the device power consumption induced by beacon receptions.

IV. Modeling and optimization

This section presents throughput, power consumption and energy efficiency models. For the sake of comparison, both *Class A* (*i.e.* Pure ALOHA) and a Slotted ALOHA scheme running on top of *LoRaSync* will be assessed. Let us first remind the most usual ALOHA throughput modeling with a finite number of devices n, as presented in [11]. The number of frames generated by any device during a duration t is modeled by a Poisson distribution of parameter λ. Besides, we make the hypothesis that all frames in the network have the same Time on Air (ToA) and do not require an acknowledgment. Additionally, Pure ALOHA devices do not use any kind of buffering, and slotted ones use a buffer of size 1 that simply allows them to wait for the start of the next slot. If we define this ToA as the elementary time unit of our modeling, then the probability p that a pure ALOHA device generates a frame during such time unit follows the cumulative distribution function of the exponential law:

$$p = p[X \leq 1] = 1 - e^{-\lambda} \tag{6}$$

The probability that exactly k among n devices generate a frame during a time unit can then be derived with the binomial coefficient:

$$P[k \text{ in } n] = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k} \tag{7}$$

For Pure ALOHA, the average throughput T_p (in erlangs) is equal to the probability that exactly one device generates a frame during a time unit, and that none of the other $(n-1)$ devices do during the previous one:

$$T_p = P[1 \text{ in } n] \cdot P[0 \text{ in } (n-1)] = np(1-p)^{2(n-1)} \tag{8}$$

In order to adapt this modeling to *LoRaSync*, we introduce q as the probability that exactly one device generates a frame for the duration of a slot. We once again leverage the exponential law:

$$q = p\Big[X \leq \frac{L_{\text{slot}}}{ToA_{\text{max}}}\Big] = 1 - e^{-\lambda \frac{L_{\text{slot}}}{ToA_{\text{max}}}} \tag{9}$$

Besides, the k_{s} coefficient is introduced as a means to represent the actual transmission time available per slotframe:

$$k_{\text{s}} = \frac{n_{\text{slots}} \cdot ToA_{\text{max}}}{BEACON\_PERIOD} \tag{10}$$

Using once again the binomial coefficient to compute the probability that exactly one device generates a frame during the duration of a slot, the average slotted ALOHA throughput T_s (in erlangs) is equal to:

$$T_s = k_s n q (1-q)^{n-1} \tag{11}$$

In order to model the energy efficiency of the devices, their power consumption (P) must first be computed. This is done

by considering its value in the transmission, reception and sleep states separately. We note P_{TX}, P_{RX} and P_{SLEEP} the power consumption of a typical SX1276 LoRa transceiver in transmission, reception and sleeping states respectively, when equipped with a 3.3V battery voltage [28]. It is computed with the transceiver's supply current in the considered state, respectively 20, 10.8 and $0.2 \cdot 10^{-3}$ mA. Our modeling accounts for the LoRaWAN 30 ms. reception slots following the uplink transmissions (*i.e.*, RX1 and RX2). For this purpose, we introduce the slot listening rate ρ_s:

$$\rho_s = \lambda \cdot \frac{2 \cdot 0.03}{ToA_{\text{max}}} \tag{12}$$

The overall network power consumption therefore is:

$$P_p = \left[\lambda P_{\text{TX}} + \rho_s P_{\text{RX}} + (1 - \lambda - \rho_s) P_{\text{SLEEP}}\right] \cdot n \tag{13}$$

For the *Class S* model we need to additionally consider the time spent in a receiving state for beacon reception purposes. We define the device beacon reception period T_{bcn} as:

$$T_{\text{bcn}} = BEACON\_PERIOD \cdot (n_{\text{skip}} + 1) \tag{14}$$

The beacon time on air is noted ToA_{bcn}. With *LoRaSync*, the beacon reception window is enlarged to cope for the worst-case drift that can occur during T_{bcn} with a drift coefficient d and a noise margin ν. Indeed, the device transmission may be shifted by an offset ranging from $-d \cdot T_{\text{bcn}} - \nu$ to $d \cdot T_{\text{bcn}} + \nu$. At the end of the beacon reception, the device switches its radio to sleep mode, so that the elapsed listening time may range from ToA_{bcn} to $ToA_{\text{bcn}} + 2 \cdot (d \cdot T_{\text{bcn}} + \nu)$ depending on the transmission offset. We assume that the drift coefficient d of all devices is uniformly distributed in the interval $\pm d$, therefore the average time spent in listening mode is $d \cdot T_{\text{bcn}} + \nu$. As a result, the beacon listening rate ρ_b is:

$$\rho_b = \frac{ToA_{\text{bcn}} + d \cdot T_{\text{bcn}} + \nu}{T_{\text{bcn}}} \tag{15}$$

The power consumption model can then be expressed as such:

$$P_s = \left[(\rho_s + \rho_b) P_{\text{RX}} + \lambda P_{\text{TX}} + (1 - \rho_s - \rho_b - \lambda) P_{\text{SLEEP}}\right] \cdot n \tag{16}$$

The energy efficiency (E) is expressed in Bytes per Joule and can be obtained by dividing the throughput by the power consumption. The throughput must however be expressed in bytes per second first, which is achieved by the term $\frac{bytes_{\text{pkt}}}{ToA_{\text{pkt}}}$ in the equations below. The energy efficiency models for Pure and Slotted ALOHA, respectively E_p and E_s, are therefore:

$$E_p = \frac{T_p}{P_p} \cdot \frac{bytes_{\text{pkt}}}{ToA_{\text{pkt}}} \tag{17}$$

$$E_s = \frac{T_s}{P_s} \cdot \frac{bytes_{\text{pkt}}}{ToA_{\text{pkt}}} \tag{18}$$

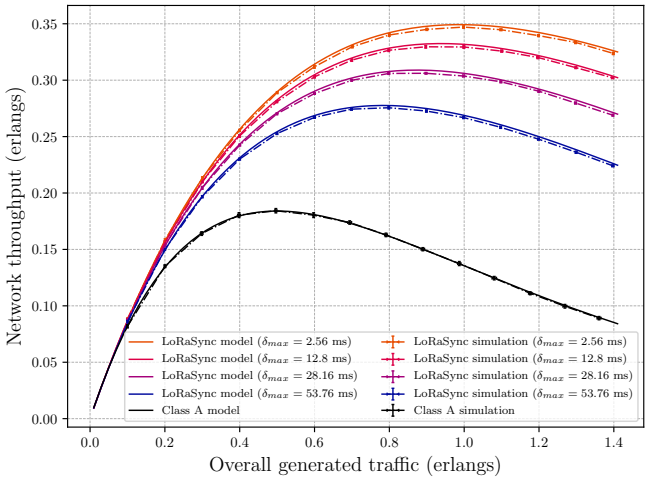


Fig. 5: Throughput model and simulation results

A. Model validation and analysis

LoRaSync targets large-scale deployments that experience scalability issues with the legacy *Class A*. Given the difficulty to approach such a large scale scenario with a research testbed, the models presented above have been validated using the LoRaWAN-Sim simulation environment to instantiate a network of 2000 devices. This analysis is completed by a proof-of-concept implementation on a few devices (*c.f.* Section V) to demonstrate the feasibility of our approach. The LoRaWAN *Class A* and a Slotted ALOHA access over *LoRaSync* have been replicated in the simulator. Additionally, a clock drift feature has been implemented according to the model derived from real measurements in Section III-B. The transmission parameters have been set to SF 7, a coding rate of $4/5$ with explicit header and cyclic redundancy check enabled. All frames carry the maximum payload size allowed allowed by the LoRaWAN specification (*i.e.* 250 bytes) to minimize the amount of overhead transmitted by the devices, which results in a ToA of 389.376 ms. *LoRaSync* was set up with a worst-case drift coefficient $d = 20$ ppm, which is a typical value for low-quality crystals found in such cheap hardware, and a noise margin $\nu = 11$ ms as measured in Section III-B. Throughout this analysis, error bars represent 99% confidence intervals along the x and y axes computed with Student's t law.

The simulated throughput for Pure and Slotted ALOHA are compared to the models in Figure 5, and the energy efficiency equivalent is displayed in Figure 6. In all cases, the model curves match the simulation results, showing the consistency between the modeling and the protocol implementation on LoRaWAN-Sim. In each plot *Class A* is compared to *LoRaSync* for several slot sizes. This comparison notably shows that the models faithfully capture the impact of the transmission margin size on the overall performances, which will be relevant for the analysis of Section IV-B.

When focusing on the observed performances, Figure 5 also shows the good operation of the *LoRaSync* synchronization,

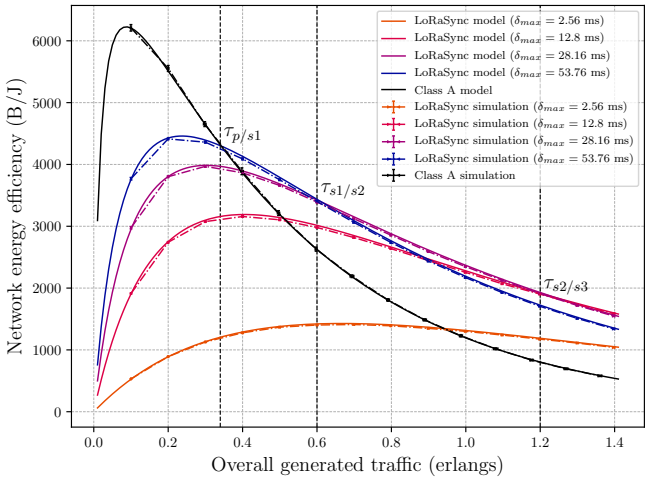


Fig. 6: Energy efficiency model and simulation results

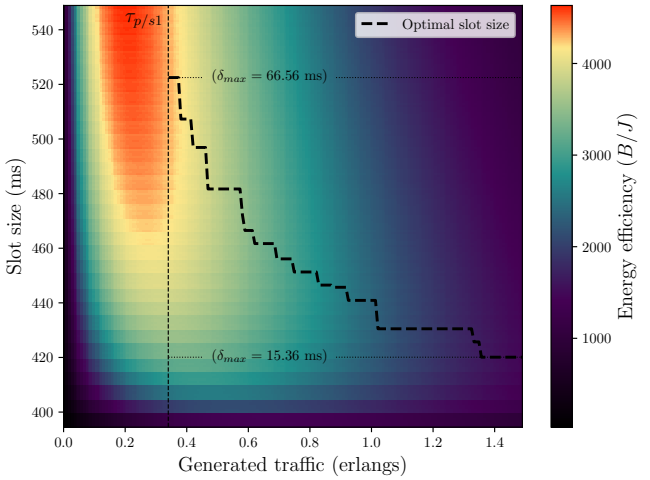


Fig. 7: Energy efficiency as a function of the slot size (2000 devices)

since the slotted access allows to nearly double the maximum achievable throughput as expected. Regarding energy efficiency we find the result already presented in [29], which is that *Class A* performs better for low generation rates and the slotted access becomes preferable when traffic rate is higher than the threshold $\tau_{p/s1}$. With this network configuration, and if we consider 53.76 ms to be the upper bound to δ_{max}, $\tau_{p/s1}$ is equal to 0.34 erlangs. This value strongly depends on the hypothesis that devices are featured with a 20 ppm. drift, which is a pessimistic estimation as we saw in Figure 3. Using devices of higher quality will naturally increase the Slotted ALOHA efficiency and move the crossing point towards a lower rate. More interestingly, Figure 6 shows that the value of δ_{max}, and therefore the slot size, has a strong impact on the network energy efficiency. Indeed, for a generated traffic range of 0.34 to 0.6 erlangs the maximum energy efficiency is attained when $\delta_{\text{max}} = 53.76$ ms, then the $\delta_{\text{max}} = 28.16$ ms curve is the best until 1.2 erlangs and finally $\delta_{\text{max}} = 12.8$ ms prevails above that point. These intervals are represented by the $\tau_{s1/s2}$ and $\tau_{s2/s3}$ thresholds in Figure 6. In order to further explore this finding, the next part sheds some light on the link between slot size and energy efficiency.

B. Slot size optimization

This part aims at evaluating the impact of the *LoRaSync* slot size on the network performances. Indeed the slot margin length δ_{max} has an effect on the maximum achievable throughput, but also on the synchronization periodicity which in turn alters the device power consumption. This makes it a relevant parameter to consider when optimizing energy efficiency. In order to represent a realistic scenario, the models presented in the previous Section has been used to analyze the same large-scale deployment of 2000 devices.

From equation 3 we know that the slot size depends on δ_{max}, which represents the maximum device drift allowed. Increasing δ_{max} will result in larger slots, ultimately decreasing the number of slots per slotframe and the overall throughput. On the other hand the bigger δ_{max}, the more devices are able to drift. This means that they are able to skip more beacons. Each additional beacon skipped reduces the overall device power consumption. Energy efficiency, defined as the ratio between the throughput and power consumption, thus strongly depends on the slot size. It has been plot as a function of the slot size and generated traffic in Figure 7.

At each rate, the slot size associated with the maximum energy efficiency is represented by the black dashed line. The line is only traced for generation rates bigger than the $\tau_{p/s1}$ threshold introduced in Figure 6. Indeed the asynchronous Pure ALOHA access should be preferred in terms of energy efficiency for rates lower than this threshold, it is therefore useless to consider the slot size there. This line shows that the optimal slot size decreases as the network load increases. Remarkably, with the same maximum ToA, smaller slots mean that devices are allowed to drift of a smaller amount. As a result a shorter synchronization period is required, which implies consuming more power in beacon receptions. Such short margins are therefore only suitable for high transmission rates. Conversely, wide slots allow to space out synchronization events, but reduce the number of slots available per slotframe. This allows to save power at the cost of a reduced maximum achievable throughput, which is not a problem when the traffic load is low. In a nutshell the more traffic there is, the more the margins should be reduced.

All in all, this result shows that the slot size should ideally be adapted to the traffic load in order to maximize energy efficiency. We presented in [29] the TREMA mechanism to dynamically switch between Pure and Slotted ALOHA depending on the probed traffic conditions. Interestingly, this work could easily be extended to dynamically adapt the slot size to the generated load in order to further optimize the energy efficiency of the network.

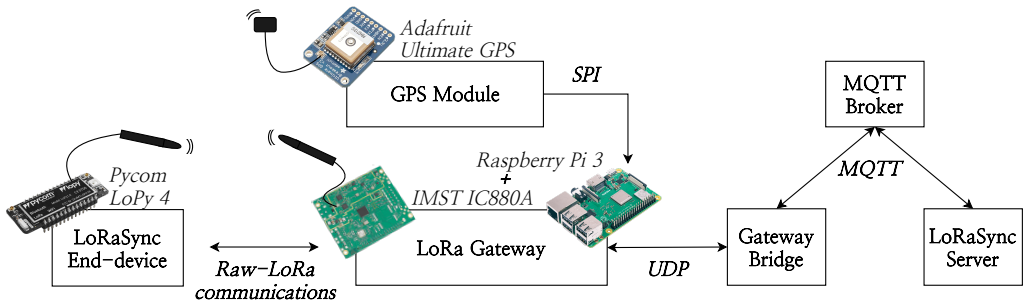


Fig. 8: Experimental testbed infrastructure.

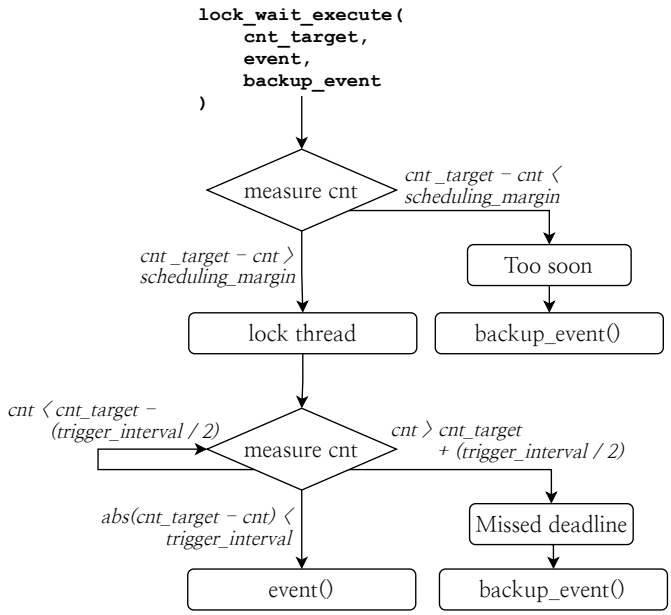


Fig. 9: `lock_wait_execute` flowchart.

V. Proof of Concept on Real Hardware

In order to complement the previous theoretical analysis, *LoRaSync* was implemented on a small-scale research testbed. This proof-of-concept demonstrates that our mechanism is capable of bounding the synchronization error despite real hardware constraints. We additionally provide hands-on insights on how to handle the hardware and software challenges faced when synchronizing low-cost LoRa devices.

A. Architectural challenges

A LoRa testbed was set up according to the typical LPWAN architecture, *i.e.* a server, a gateway and devices, as depicted in Figure 8.

The server side mimicks the ChirpStack architecture, in which a MQTT broker centralizes the communications of all gateways. The ChirpStack gateway bridge allows to translate the MQTT streams into UDP datagrams understandable by the gateways. Finally, a custom *LoRaSync* server was developed to control and monitor the network with MQTT messages. The gateway is composed of a Raspberry Pi 3 running the Semtech Packet Forwarder software, used in conjunction with an IMST IC880A LoRa concentrator. The testbed devices are LoPy 4 chips developed by Pycom, offering quick prototyping capabilities for a relatively low cost.

By default, LoRa gateways are only capable of triggering downlink transmissions *(i)* immediately, or *(ii)* after a short delay following an uplink transmission (*Class A* RX windows). In order to allow timestamp-based downlink transmissions, an Adafruit Ultimate GPS chip was connected to the Raspberry Pi through Serial Peripheral Interface (SPI). It provides time and location information to the gateway program, and also sends a direct Pulse Per Second (PPS) signal to the concentrator enabling the triggering of transmissions with a 10 nanoseconds precision [30]. This feature is mandatory in order to enable the periodic broadcasting of the *LoRaSync* synchronization beacons.

B. Software challenges

The main challenge faced when programming *LoRaSync* on LoPy devices was to schedule event executions with absolute timestamps, even though the embedded time library only allows callbacks after relative delays. To achieve this goal, the device logic was first split onto four threads, each assigned to a specific task: *(i)* message generation, *(ii)* frame reception, *(iii)* handling of received messages, and *(iv)* frame transmission. In this way, events requiring a precise timing such as frame transmissions and receptions may lock the CPU when needed, while the other tasks may proceed their execution freely during non-critical time.

Additionally, a software overlay was developed in order to handle critical method executions. Once synchronized, a node is able to estimate its current timestamp with its internal counter according to equation (1). When a timestamp event is scheduled, relative callbacks can then used to trigger a call to the `lock_wait_execute` method, capable of handling time-critical events. This method is in charge of locking the hardware resources for the current thread, and waiting until the internal counter falls within a precise confidence interval around the target timestamp (*c.f.* Figure 9). It then triggers the event execution with a ~ 100 microseconds precision. In

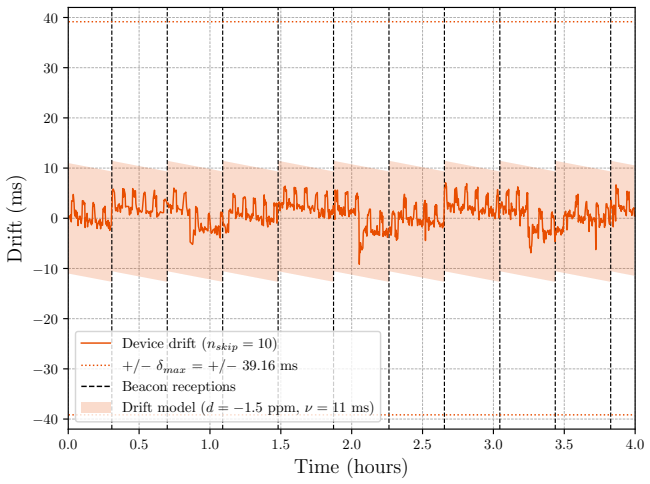


Fig. 10: Device drift measurements

the case where resources could not be locked in time for the critical method call, a backup event passed as parameter is executed in order to correct the failure.

Once the device is synchronized, this triggering process allows to track beacons by turning the radio on just before the next broadcast. It also enables uplink timestamp-based transmissions, which ultimately allows to implement the *Class S* timeslots.

C. Clock correction demonstration and discussion

In order to demonstrate the good operation of *LoRaSync*, the clock drift of a periodically synchronized device is plot in Figure 10. This device is setup with $\delta_{\text{max}} = 39.16$ ms, a worst-case drift coefficient $d = 20$ ppm and a noise margin $\nu = 11$ ms. This setting results in $n_{\text{skip}} = 10$, meaning that a beacon is received every $11 \times 128 = 1408$ seconds. Beacon receptions are represented on the graph by black vertical dashed lines.

Here it is clear that the device never crosses the maximum drift allowed line $\delta_{\text{max}} = 39.16$. The margin here appears particularly wide because the actual drift coefficient of the chosen device is $d = -1.5$ ppm, which is very small compared to the worst-case estimation of 20 ppm. The drift model interval has been plot over the actual drift in order to highlight the synchronization events. It shows that in this particular case, the noise plays a bigger role in clock errors than the linear component of the drift.

When evaluating this implementation with several devices, we found that the capture effect greatly impacted the network throughput. This physical-layer phenomenon will therefore be integrated to our performance models in a future contribution in order to picture the behavior of real-life LoRa networks even more faithfully.

All in all, this proof of concept shows that the synchronization error is kept below the worst-case drift threshold δ_{max}. *LoRaSync* therefore operates as expected in a realistic context.

VI. Conclusion and perspectives

In this paper we introduced *LoRaSync*, a synchronization scheme designed for LoRa networks that aims at maximizing the device energy efficiency. Synchronization was used to setup a Slotted ALOHA access providing capacity improvements compared to the legacy Pure ALOHA scheme. *LoRaSync* slots may however be leveraged to implement any slotted MAC protocol. Synchronization is achieved through beacon broadcasts, and is therefore scalable to any network size. The *LoRaSync* slots contain margins to cope for the worst-case device clock drift, ensuring a reliable operation even on low-cost hardware. The margin size is directly linked to a beacon skipping mechanism that reduces the power consumption due to beacon receptions to the minimum.

Throughput, power consumption and energy efficiency models were established to assess the performances of the Slotted ALOHA scheme built on top of *LoRaSync*. This access scheme was compared to the asynchronous Pure ALOHA access, and the LoRaWAN-Sim simulator was used to verify the consistency of the results. Such modeling ultimately allows to find the most efficient slot size for any traffic load. Results show that the more traffic is generated, the smaller margins should be in order to maximize energy efficiency at all times. *LoRaSync* was implemented on an experimental testbed, showing its capability to successfully correct the timing errors and keep the clock drift below the maximum threshold allowed even on a real-hardware setup.

Preliminary testbed results have pushed us to integrate the capture effect to the presented performance models in a future contribution. Besides, the TREMA switching mechanism allowing to select synchronous or asynchronous access schemes depending on the traffic load should be implemented, and improved to also adapt the slot size. Finally, all these contributions should be extended to operate in multi-gateway scenarios as well.

References

[1] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, Mar. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959517302953

[2] M. Rizzi, P. Ferrari, A. Flammini, and E. Sisinni, "Evaluation of the IoT LoRaWAN Solution for Distributed Measurement Applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 12, pp. 3340–3349, Dec. 2017, conference Name: IEEE Transactions on Instrumentation and Measurement.

[3] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa Low-Power Wide-Area Networks Scale?" in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '16. New York, NY, USA: ACM, 2016, pp. 59–67, event-place: Malta, Malta. [Online]. Available: http://doi.acm.org/10.1145/2988287.2989163

[4] K. Mikhaylov, J. Petäjäjärvi, and J. Janhunen, "On LoRaWAN scalability: Empirical evaluation of susceptibility to inter-network interference," in *2017 European Conference on Networks and Communications (EuCNC)*, Jun. 2017, pp. 1–6.

[5] F. Van den Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Scalability Analysis of Large-Scale LoRaWAN Networks in ns-3," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2186–2198, Dec. 2017, conference Name: IEEE Internet of Things Journal.

[6] J. Haxhibeqiri, F. Van den Abeele, I. Moerman, and J. Hoebeke, "LoRa Scalability: A Simulation Model Based on Interference Measurements," *Sensors*, vol. 17, no. 6, p. 1193, Jun. 2017, number: 6 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1424-8220/17/6/1193

[7] K. Q. Abdelfadeel, D. Zorbas, V. Cionca, B. O'Flynn, and D. Pesch, "FREE - Fine-grained Scheduling for Reliable and Energy Efficient Data Collection in LoRaWAN," *arXiv:1812.05744 [cs]*, Dec. 2018.

[8] N. Abramson, "THE ALOHA SYSTEM: another alternative for computer communications," in *Proc. of AFIPS '70 (Fall)*, ser. AFIPS '70 (Fall). Houston, Texas: ACM, Nov. 1970, pp. 281–285.

[9] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment," *Sensors*, vol. 19, no. 4, p. 838, Jan. 2019.

[10] J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Low Overhead Scheduling of LoRa Transmissions for Improved Scalability," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3097–3109, Apr. 2019.

[11] R. Rom and M. Sidi, "Multiple Access Protocols: Performance and Analysis," *SIGMETRICS Perform. Evaluation Rev.*, vol. 18, p. 11, 1991.

[12] R. Piyare, A. L. Murphy, M. Magno, and L. Benini, "On-Demand TDMA for Energy Efficient Data Collection with LoRa and Wake-up Receiver," in *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct. 2018, pp. 1–4, iSSN: 2160-4886.

[13] L. Beltramelli, A. Mahmood, P. Osterberg, M. Gidlund, P. Ferrari, and E. Sisinni, "Energy efficiency of slotted lorawan communication with out-of-band synchronization," *IEEE Transactions on Instrumentation and Measurement*, p. 1–1, 2021.

[14] C. Garrido-Hidalgo, J. Haxhibeqiri, B. Moons, J. Hoebeke, T. Olivares, F. J. Ramirez, and A. Fernández-Caballero, "LoRaWAN Scheduling: From Concept to Implementation," *IEEE Internet of Things Journal*, 2021.

[15] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, and S. Pollin, "Improving Reliability and Scalability of LoRaWANs Through Lightweight Scheduling," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1830–1842, Jun. 2018.

[16] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, and D. Pesch, "TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things," *Computer Communications*, vol. 153, pp. 1–10, Mar. 2020.

[17] L. Tessaro, C. Raffaldi, M. Rossi, and D. Brunelli, "Lightweight Synchronization Algorithm with Self-Calibration for Industrial LORA Sensor Networks," in *2018 Workshop on Metrology for Industry 4.0 and IoT*, Apr. 2018, pp. 259–263.

[18] L. Chasserat, N. Accettura, and P. Berthou, "Short: Achieving energy efficiency in dense LoRaWANs through TDMA," in *IEEE WoWMoM 2020*, Cork, Ireland, Aug. 2020.

[19] O. Georgiou and U. Raza, "Low Power Wide Area Network Analysis: Can LoRa Scale?" *IEEE Wireless Communications Letters*, vol. 6, no. 2, pp. 162–165, Apr. 2017.

[20] A.-I. Pop, U. Raza, P. Kulkarni, and M. Sooriyabandara, "Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks?" *arXiv:1704.04174 [cs]*, Dec. 2017.

[21] V. Di Vincenzo, M. Heusse, and B. Tourancheau, "Improving Downlink Scalability in LoRaWAN," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–7, iSSN: 1938-1883.

[22] S. El-Khamy, S. Shaaban, and E. Thabet, "Frequency-hopped multi-user chirp modulation (FH/M-CM) for multipath fading channels," in *Proceedings of the Sixteenth National Radio Science Conference. NRSC'99 (IEEE Cat. No.99EX249)*, Feb. 1999.

[23] K. Mikhaylov, J. Petäjäjärvi, and T. Haenninen, "Analysis of capacity and scalability of the lora low power wide area network technology," in *European Wireless 2016; 22th European Wireless Conference*, ser. Proceedings of EW '16, Oulu, Finland, May 2016, pp. 1–6.

[24] M. Knight and B. Seeber, "Decoding LoRa: Realizing a Modern LPWAN with SDR," *Proc. of the GNU Radio Conference*, vol. 1, no. 1, Sep. 2016.

[25] L. Alliance, "LoRaWAN® L2 1.0.4 Specification (TS001-1.0.4)," Oct. 2020.

[26] D. Veitch, S. Babu, and A. Pàsztor, "Robust synchronization of software clocks across the internet," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, ser. IMC '04. New York, NY, USA: Association for Computing Machinery, Oct. 2004, pp. 219–232. [Online]. Available: https://doi.org/10.1145/1028788.1028817

[27] A. Mallat and L. Vandendorpe, "Joint estimation of the time delay and the clock drift and offset using UWB signals," in *2014 IEEE International Conference on Communications (ICC)*, Jun. 2014, pp. 5474–5480, iSSN: 1938-1883.

[28] Semtech Corporation, "SX1276/77/78/79 datasheet," Jan. 2019.

[29] L. Chasserat, N. Accettura, B. Prabhu, and P. Berthou, "TREMA: A traffic-aware energy efficient MAC protocol to adapt the LoRaWAN capacity," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, Jul. 2021, pp. 1–6, iSSN: 2637-9430.

[30] A. Industries, "Adafruit Ultimate GPS Breakout v3 datasheet," 2021.