Enabling Cross-technology Communication from LoRa to ZigBee via Payload Encoding in Sub-1 GHz Bands

JUNYANG SHI, DI MU, and MO SHA, State University of New York at Binghamton

Low-power wireless mesh networks (LPWMNs) have been widely used in wireless monitoring and control applications. Although LPWMNs work satisfactorily most of the time thanks to decades of research, they are often complex, inelastic to change, and difficult to manage once the networks are deployed. Moreover, the deliveries of control commands, especially those carrying urgent information such as emergency alarms, suffer long delay, since the messages must go through the hop-by-hop transport. Recent studies show that adding low-power wide-area network radios such as LoRa onto the LPWMN devices (e.g., ZigBee) effectively overcomes the limitation. However, users have shown a marked reluctance to embrace the new heterogeneous communication approach because of the cost of hardware modification. In this article, we introduce LoRaBee, a novel LoRa to ZigBee cross-technology communication (CTC) approach, which leverages the energy emission in the Sub-1 GHz bands as the carrier to deliver information. Although LoRa and ZigBee adopt distinct modulation techniques, LoRaBee sends information from LoRa to ZigBee by putting specific bytes in the payload of legitimate LoRa packets. The bytes are selected such that the corresponding LoRa chirps can be recognized by the ZigBee devices through sampling the received signal strength. Experimental results show that our LoRaBee provides reliable CTC communication from LoRa to ZigBee with the throughput of up to 281.61 bps in the Sub-1 GHz bands.

CCS Concepts: • **Networks → Network protocol design**; **Sensor networks**; *Network management;*

Additional Key Words and Phrases: Cross-technology communication, LoRa, ZigBee, low-power wireless mesh network

ACM Reference format:
Junyang Shi, Di Mu, and Mo Sha. 2021. Enabling Cross-technology Communication from LoRa to ZigBee via Payload Encoding in Sub-1 GHz Bands. *ACM Trans. Sen. Netw.* 18, 1, Article 6 (September 2021), 26 pages.
https://doi.org/10.1145/3470452

1 INTRODUCTION

The **Internet of Things (IoT)** refers to a broad vision whereby things such as everyday objects, places, and environments are connected to each other via the Internet [27]. Many wireless technologies (e.g., ZigBee [45], WiFi [35], and Bluetooth [2]) are readily available to form the networks that connect those things for various IoT applications. Many of those networks follow the **low-power wireless mesh network (LPWMN)** paradigm and have been widely deployed for

Part of this article was published in Proceedings of the ICNP [29].
This work was supported in part by the National Science Foundation under grants CNS-1657275 and CNS-2046538.

Authors' address: J. Shi, D. Mu, and M. Sha (corresponding author), State University of New York at Binghamton, 4400 Vestal Parkway East, Binghamton, NY, 13902, USA; emails: {jshi28, dmu1, msha}@binghamton.edu.
Authors current address: Mo Sha is currently with the Knight Foundation School of Computing and Information Sciences at Florida International University, 11200 SW 8th St, Miami, FL 33199; email: msha@fiu.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Association for Computing Machinery.

1550-4859/2021/09-ART6 $15.00
https://doi.org/10.1145/3470452

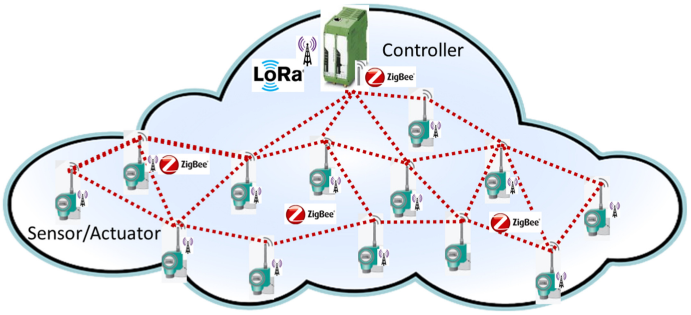


Fig. 1. A LPWMN equipping with LPWAN radios.

monitoring and control applications. For instance, sensors and actuators equipped with ZigBee radios have been used for a decade in industrial facilities, such as steel mills, oil refineries, and chemical plants, to monitor and control automation processes [24]. A multi-hop LPWMN connects sensors and forwards sensor readings to a control room where a controller sends commands to actuators. Although LPWMNs work satisfactorily most of the time thanks to decades of research, they are often complex, inelastic to change, and difficult to manage once the networks are deployed. Moreover, the deliveries of control commands, especially those carrying urgent information such as emergency alarms, suffer long delay, since the messages have to go through the hop-by-hop transport [21]. A recent study [10] shows that adding **low-power wide-area network (LPWAN)** radios such as LoRa [23] onto the LPWMN devices (e.g., ZigBee) effectively overcomes the limitation of LPWMNs, since the key messages can be transmitted from the controller to the sensors and actuators through the direct long-distance links, as Figure 1 shows. However, the industry practitioners have shown a marked reluctance to embrace the new heterogeneous communication approach because of the cost of hardware modification.

Cross-technology communication (CTC) technologies have been seeing appreciable advancement in recent years. Significant efforts have been made to enable the direct communication among ZigBee, WiFi, and Bluetooth devices in the 2.4 GHz **industrial, scientific, and medical radio (ISM)** bands [5, 6, 11, 12, 17–19, 22, 33, 34, 37, 38, 40–43]. Unfortunately, those existing solutions are not directly applicable to send messages from LoRa to ZigBee because of the unique characteristics of LoRa in the Sub-1 GHz bands. However, there has been increasing interest in using ZigBee in the Sub-1 GHz bands. After the ZigBee Alliance announced its ZigBee PRO 2017 with the dual-band option [46], the ZigBee devices that operate in the sub-1 GHz bands have been widely deployed for manufacturing, smart homes, and smart cities. For instance, the devices based on the Zigbee PRO 2017 are part of Europe's biggest engineering projects today and the Zigbee PRO-based solutions are being deployed across the United Kingdom, which has a government mandate to roll out smart meters to approximately 30 million homes by 2020 [3]. In this article, we introduce LoRaBee, a novel LoRa to ZigBee CTC approach, which leverages the energy emission in the Sub-1 GHz bands as the carrier to deliver information. The highlight of LoRaBee design lies in its simplicity and compatibility. Although LoRa and ZigBee adopt distinct modulation techniques, LoRaBee sends information from LoRa to ZigBee by putting specific bytes in the payload of legitimate LoRa packets, namely payload encoding. The bytes are selected such that the corresponding LoRa chirps can be recognized by the ZigBee devices through sampling the **received signal strength (RSS)**. This design ensures full compatibility with the **commercial off-the-shelf (COTS)** LoRa and ZigBee devices. A LoRa base station can disseminate the network management messages, time synchronization beacons, and urgent information to ZigBee devices through our CTC approach. Specifically, this article makes the following contributions:

- To our knowledge, this is the first article to investigate CTC from LoRa to ZigBee in the Sub-1-GHz bands, distinguished with previous work pertaining to CTC among WiFi, ZigBee, and Bluetooth devices in the 2.4 GHz band.
- This article performs an empirical study that investigates the characteristics of LoRa from a CTC's point of view and provides a set of new observations.
- This article introduces LoRaBee, a novel LoRa to ZigBee CTC approach. By elaborately tuning the LoRa's central carrier frequency and packet payload, a ZigBee device is capable of decoding the information carried by the LoRa chirps by purely sampling the RSS. LoRaBee does not require any hardware modification.
- This article presents a new **Time Division Multiple Access- (TDMA)** based **medium access control (MAC)** protocol, which allows a LoRa device to time synchronize and deliver information to a network of ZigBee devices through LoRaBee.
- LoRaBee has been implemented and tested on real hardware. Experimental results show that LoRaBee provides reliable CTC communication from LoRa to ZigBee with the throughput of up to 281.61 bps[1] in Sub-1 GHz bands.

The remainder of the article is organized as follows. Section 2 reviews the related work and Section 3 discusses the background of LoRa and ZigBee. Section 4 introduces our empirical study. Sections 5 and 6 present the design of our LoRaBee and MAC protocol. Section 7 evaluates LoRaBee and Section 8 concludes the article.

2 RELATED WORKS

There has been increasing interest in developing CTC technologies in recent years. Significant efforts have been made to enable the direct communication among ZigBee, WiFi, and Bluetooth devices in the 2.4 GHz ISM bands [5, 6, 11, 12, 16–18, 22, 33, 34, 37–43]. The key idea of packet level CTC is that heterogeneous wireless devices operating in the shared spectrum need to sense the presence of signal via channel energy detection, such as RSS and channel state information. Most of the CTC technologies leverage the energy intensity, gap between energy appearance, and duration of radio energy to modulate data. For instance, Chebrolu et al. proposed to enable the communication from WiFi to ZigBee devices based on sensing and interpreting energy profiles and convey information by modulating the WiFi energy duration to construct an alphabet set [5]. Zhang et al. developed GapSense, which leverages the sequences of energy bursts to modulate symbol [41]. Kim et al. proposed FreeBee, which adjusts the appearance of WiFi beacons in the time dimension to transmit modulated data [18, 19]. Yin et al. designed C-Morse, which controls the presence of data traffic to deliver information [38]. Guo et al. designed a CTC technique that employs modulation techniques in both the amplitude and temporal dimensions to optimize the throughput over a noisy channel [12]. Li et al. developed WEBee, which uses WiFi packets to directly emulate the ZigBee signals in the physical-layer [22]. Yin et al. proposed to use the presence and absence of energy profiles to convey information among heterogeneous wireless devices [37]. More recently, Zheng et al. developed StripComm, which is an interference-aware CTC modulation and demodulation scheme [43]. Gawlowicz et al. designed LtFi that allows direct communication between **LTE in Unlicensed (LTE-U)** and WiFi devices, which can enable collaboration between co-located LTE-U and WiFi networks to mitigate interference [9, 47]. Zheng et al. designed a transparent cross-technology opportunistic forwarding method to mitigate Cross-Technology Interference [44]. Guo et al. developed ZigFi that uses channel state information to convey data

[1]As a comparison for the throughput value, a LoRa device pair provides a throughput of up to 11 kbps under the same settings.

from ZigBee to WiFi [11]. Yu et al. [39] and Hao et al. [16] proposed to use CTC for clock synchronization. Jiang et al. developed SymBee that achieves symbol-level CTC from ZigBee to WiFi [33]. Jiang et al. [17] and Chi et al. [6] enabled the CTC between ZigBee and Bluetooth devices. Wang et al. [32] leveraged the CTC technology to turn the pervasively-deployed WiFi access point into a multi-user transmitter, which transmits different packets to multiple ZigBee devices in parallel. Xia et al. [36] extended the communication range from ZigBee to WiFi and established symmetric CTC over asymmetric channels. In contrast to previous studies on CTC among ZigBee, WiFi, and Bluetooth devices in the 2.4 GHz ISM bands, this article investigates the characteristics of LoRa in the Sub-1 GHz bands; to our knowledge, it represents the first systematic study on CTC from LoRa to ZigBee. Our work is therefore orthogonal and complementary.

LoRaBee enables the CTC from LoRa to ZigBee in the Sub-1 GHz bands by encoding the LoRa packet payload with specific bytes, whose corresponding LoRa chirps can be detected by the ZigBee device through sampling the RSS. Specifically, the ZigBee device detects the sudden RSS value drop caused by each LoRa chirp and uses the time intervals of all RSS value drops to decode information. Unfortunately, LoRaBee is not applicable in the 2.4 GHz band, because the LoRa device transmits much faster and the ZigBee device is not capable of sampling RSS frequent enough to detect individual LoRa chirps. To enable the CTC from LoRa to ZigBee in the 2.4 GHz, we have developed another approach, which encodes information using the sizes of the LoRa payloads [28]. The ZigBee device uses the number of the consecutive RSS values higher than a threshold to decode information. Such an approach [28] encodes less information in each time unit, which results in lower throughput if applied in the Sub-1 GHz bands compared to LoRaBee. Therefore, our two CTC approaches reported in this article and [28] are complementary with each other.

3 BACKGROUND

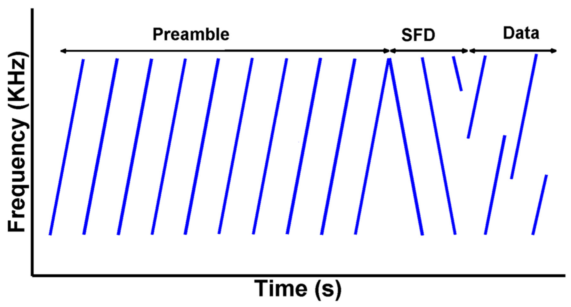
3.1 LoRa Overview

LPWANs are emerging as a new paradigm in the field of IoT connectivity [31]. LoRa is an industry LPWAN technology that has been initiated by Semtech [7] to build scalable IoT networks. LoRa provides a radio modulation scheme, which leverages **chirp spread spectrum (CSS)** modulation to deliver data. LoRa utilizes the unlicensed ISM bands and incorporates a variation of CSS technique to encode information.

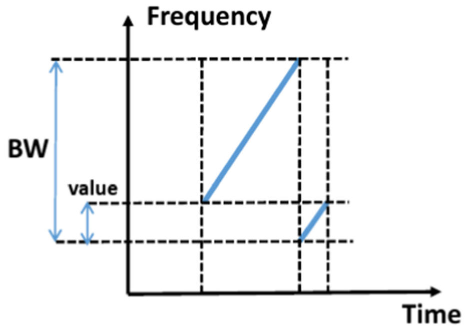
Modulation technique: LoRa employs the CSS modulation to modulate signals. It uses frequency chirps with a constantly increasing (upchirp) or decreasing (downchirp) frequency that sweeps through a predefined bandwidth. Figure 2(a) plots an example LoRa transmission with multiple chirps in the frequency variation over time. The first 10 upchirps are preamble whose frequency starts from the minimum frequency (f_{min}) to the maximum frequency (f_{max}). They are followed by 2.25 downchirps annotated as **Start Frame Delimiter (SFD)** that goes from f_{max} to f_{min}. The rest chirps carry data. The modulated data chirps start at different frequency positions represent different encoded bits. When each data chirp reaches f_{max}, it wraps around and starts from f_{min}, as Figure 2(a) shows. In other words, LoRa uses different starting frequency of the chirp signal to encode different information. As Figure 2(b) shows, the value in the y-axis represents the encoded bits. More LoRa chirps are concatenated to represent more data bits.

Key physical-layer parameters: LoRa allows users to change the central carrier frequency (f_c), frequency bandwidth (BW), spreading factor (SF), coding rate (CR), and cyclic redundancy check (CRC). Table 1 lists the possible values for each parameter in the United States. f_c determines the central carrier frequency for data transmission.[2] BW determines the magnitude of frequency

[2]LoRa can also operate in 2.4 GHz, but provides much shorter link distance. In this article, we focus on investigating the CTC in the Sub-1 GHz bands.



(a) A LoRa transmission with upchirps, downchirps and data chirps.



(b) A single LoRa data chirp.

Fig. 2. LoRa modulation.

Table 1. Key LoRa Physical-layer Parameters

Parameter	Options
f_c	between 902 MHz to 928 MHz
SF	7, 8, 9, 10, 11, 12
BW(KHz)	125, 250, 500
CR	4/5, 4/6, 4/7, 4/8
CRC	on or off

variation ($f_{max} - f_{min}$), representing the channel width. Each chirp consists of 2^{SF} chips that can carry SF bits of data. The time duration of one LoRa chirp is as follows:

$$T_{chirp} = \frac{2^{SF}}{BW}. \tag{1}$$

CR uses the Hamming code [15] to provide redundancy and correct error bits. This number refers to the proportion of the transmitted bits that actually carry information. LoRa allows users to enable the CRC check.

Input: The LoRa transceivers provided by Semtech only accept hexadecimal strings as input. The upper layer protocols must translate their data into the hexadecimal format. For instance, "$0 \times 6A$" may be input into the LoRa transceiver to carry 106.

3.2 ZigBee Overview

ZigBee is based on the IEEE 802.15.4 standard [1], which specifies to operate in the Sub-1 GHz and 2.4 GHz ISM bands. Figure 3 plots the channels defined in different frequencies. The channel 1–10 overlaps the LoRa's operating frequencies in the Sub-1 GHz bands with the channel width of 1.2 MHz, while the channel 11–26 operates in the 2.4 GHz band. Many COTS ZigBee radios (e.g., TI CC1352R [30] and Silicon Labs EFR32MG12P433F1024GM48 [20]) support operating in both Sub-1 GHz and 2.4 GHz bands. ZigBee uses Binary Phase Shift Keying modulation, which provides the throughput of up to 40 kbps in the Sub-1 GHz bands.

4 EMPIRICAL STUDY

In this section, we introduce our empirical study that investigates the characteristics of LoRa communication from a CTC's point of view and present a series of observations that provide

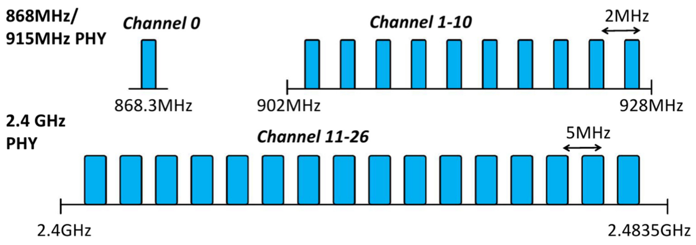
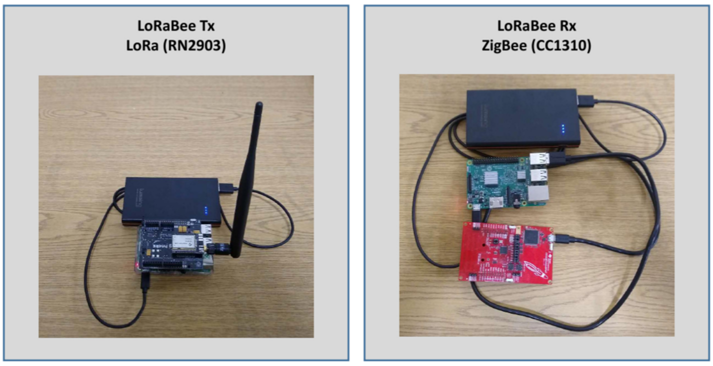


Fig. 3. IEEE 802.15.4 channels.



(a) CTC sender. (b) CTC receiver.

Fig. 4. Hardware.

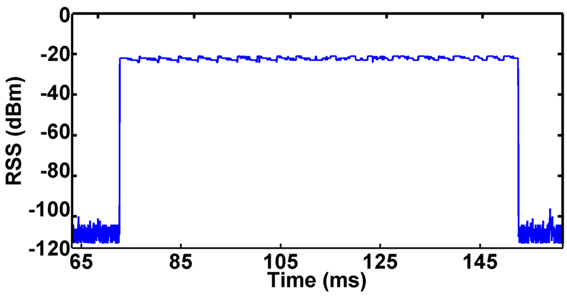
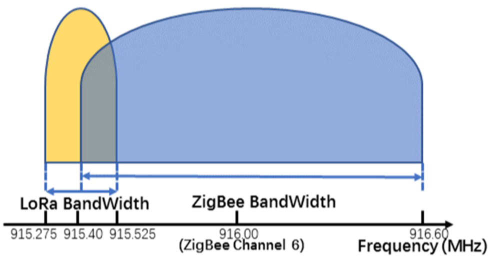


Fig. 5. An example RSS trace measured by ZigBee when LoRa and ZigBee channels overlap completely. ZigBee operates on channel 6 with the central frequency of 916 MHz. LoRa transmits a packet with the content of 0×00 using the same central frequency with $BW = 250$ KHz, $SF = 10$, $CR = 4/5$, and $CRC = off$.

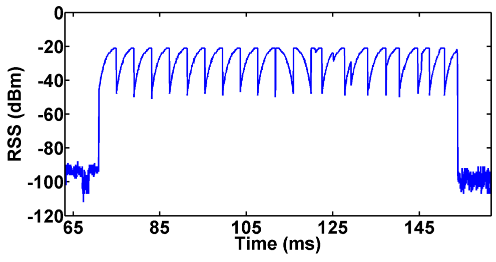
guidelines for our CTC design. We perform the experiments with two Raspberry Pi 3 Model B [26]: one integrating with a SX1272 LoRa shield [14] containing a Microchip RN2903 radio [25], which is compatible with LoRa, and the other integrating with a TI CC1310 launchpad [4], which is compatible with ZigBee. The RSS sampling rate of the TI CC1310 launchpad is 41.50 kHz. Figure 4 shows the hardware.

4.1 Energy Profiling of LoRa Signals on ZigBee

In this set of experiments, we measure the energy emission from LoRa on ZigBee. We first configure LoRa to operate completely overlapping the ZigBee channel. Figure 5 plots an example RSS trace



(a) LoRa and ZigBee channels overlap partially.



(b) Example RSS trace.

Fig. 6. An example RSS trace measured by ZigBee when the LoRa and ZigBee channels overlap partially. ZigBee operates on channel 6 with the central frequency of 916 MHz. LoRa transmits a packet with the content of 0×00 using the central frequency of 915.4 MHz with $BW = 250$ KHz, $SF = 10$, $CR = 4/5$, and $CRC = off$.

measured by ZigBee when the LoRa and ZigBee channels overlap completely. As Figure 5 shows, when LoRa begins to transmit at 72.65 ms, the RSS measured by ZigBee immediately increases from −112 dBm to −21 dBm. The RSS values vary slightly within the range of $[-24, -21]$dBm during the LoRa transmission (from 72.65 ms to 155.59 ms).

Observation 1. *ZigBee can capture the energy emission from LoRa, but cannot detect the individual LoRa chirps when the LoRa and ZigBee channels overlap completely.*

We then shift the central frequency of LoRa, making the LoRa and ZigBee channels overlap partially. Figure 6(a) shows the frequency settings of LoRa and ZigBee, making a half of the LoRa channel locate outside the ZigBee channel, and Figure 6(b) plots an example RSS trace. As Figure 6(b) shows, when LoRa begins to transmit at 68.60 ms, the RSS measured by ZigBee immediately increases and varies from −51 dBm to −21 dBm during the transmission of each LoRa chirp. ZigBee not only detects the LoRa transmission but also captures the transmissions of individual LoRa chirps including the first 10 upchirps for preamble, the 2.25 downchirps for SFD, and the eight modulated data chirps.

Observation 2. *ZigBee can detect the upchirps for preamble, the downchirps for SFD, and the modulated data chirps from its RSS measurements when the LoRa and ZigBee channels overlap partially.*

Observations 1 and 2 motivates LoRaBee to elaborately tune the central frequency of LoRa, making its channel partially overlap the ZigBee channel, to enable the CTC from LoRa to ZigBee.

4.2 LoRa Payload Encoding

In this set of experiments, we investigate the feasibility of decoding the LoRa packet payload from the measured RSS values on ZigBee. We name the measured RSS trace, representing the LoRa modulated data chirps in a packet, as a RSS signature. First, we configure LoRa to transmit the packets with the same payload and examine whether ZigBee always captures the same RSS signature. Figure 7 shows three example RSS signatures when LoRa transmits 0×01 repeatedly. From here, we only plot the data chirps and omit the upchirps and downchirps for preamble and SFD. We observe that the RSS signatures are always identical to each other when LoRa transmits the same payload and obtain the same observation after repeating the experiments with different packet payloads.

We then configure LoRa to transmit different data bytes. Figure 8 shows three RSS signatures when LoRa transmits 0×01, 0×11, and $0 \times 6A$, respectively. The differences between the three RSS signatures are noticeable. Please note that LoRa preprocesses data by performing data whitening

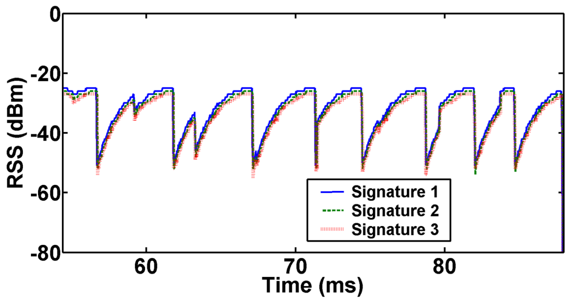


Fig. 7. RSS signatures measured by ZigBee when LoRa transmits packets with the same payload, which contains one byte (0 × 01).

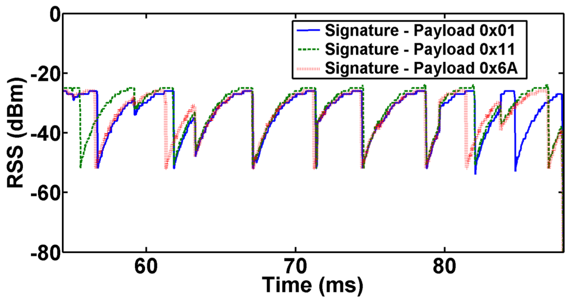


Fig. 8. RSS signatures measured by ZigBee when LoRa transmits 0 × 01, 0 × 11, and 0 × 6A, respectively.

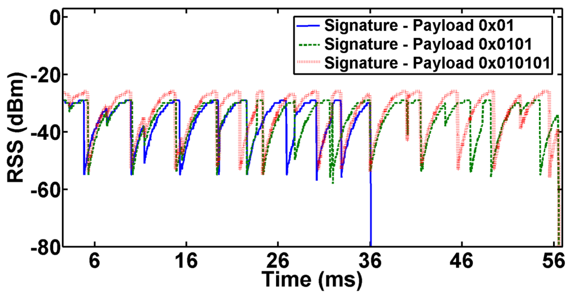


Fig. 9. RSS signatures measured by ZigBee when LoRa transmits 0 × 01, 0 × 0101, and 0 × 010101, respectively.

(introducing randomness), adding error correction bits, interleaving (adding scrambled bits), and adding chirp gray indexing for error tolerance enhancement before transmitting it. Therefore, the actual data transmitted by LoRa is encoded and scrambled from the original one. Although the encoding procedure of LoRa is closed source, the consistent mapping from the input data to the generated LoRa chirps is observed empirically.

Observation 3. *It is feasible to decode the LoRa payload from the measured RSS signature on ZigBee, since the mapping from the input data to the generated LoRa chirps is consistent.*

We also configure LoRa to carry the same byte multiple times in its packet payload and observe the RSS signature. Figure 9 plots three RSS signatures when LoRa transmits 0 × 01, 0 × 0101, and 0 × 010101, respectively. The RSS signatures are completely different. This is because LoRa

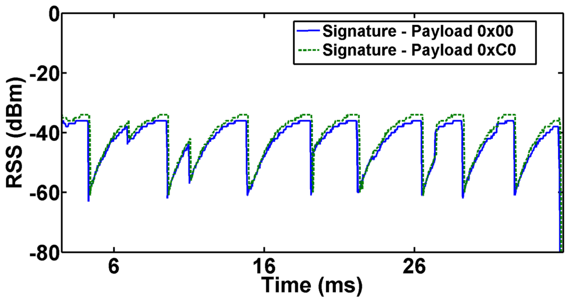


Fig. 10. RSS signatures measured by ZigBee when LoRa transmits 0×00, and $0 \times C0$, respectively.

rearranges the bits in the packet payload before transmitting them. The bytes in the packet payload are not directly concatenated, resulting in distinct RSS signatures.

Observation 4. *When LoRa carries the same byte multiple times in its packet payload, the resulting RSS signatures are different.*

Observations 3 and 4 motivate LoRaBee to send information from LoRa to ZigBee by putting a single byte in the payload of each legitimate LoRa packet. The byte is selected such that the corresponding LoRa chirps can be recognized by the ZigBee devices through sampling the RSS.

Finally, we configure LoRa to transmit all possible 1-byte payload varying from 0×00 to $0 \times FF$. We observe that some RSS signatures are indistinguishable by ZigBee due to its insufficient RSS sampling accuracy. Figure 10 shows two example RSS signatures measured by ZigBee when LoRa transmits 0×00 and $0 \times C0$, respectively.

Observation 5. *A ZigBee device may not be able to distinguish all possible bytes (0×00-$0 \times FF$) that LoRa carries due to its insufficient RSS sampling accuracy.*

Observation 5 motivates LoRaBee to generate a tailored encoding scheme for the given ZigBee device with the consideration of its hardware limitation. The encoding scheme only uses those data bytes whose RSS signatures are distinguishable by the ZigBee device to carry the CTC data. Therefore, LoRaBee may transmit more bits to carry the desired data.

4.3 Feature Selection

To enable the LoRa payload encoding, we need to correlate the data byte in the LoRa payload to the resulting RSS signature. The naive approach would be to map the byte to the entire RSS signature and let the ZigBee and LoRa devices store the mapping. At runtime, the ZigBee device can run a sequence matching algorithm to decode the information by comparing the measured RSS signature against all stored ones. However, this method suffers four major problems. First, it requires the LoRa and ZigBee devices to store all RSS sampling points, resulting in large memory consumption. Second, iterating through all RSS signatures introduces significant computation overhead and long delay. Third, the RSS values measured by the ZigBee device are not very accurate, which may introduce some sequence matching errors. Fourth, the measured RSS values depend on the distance between the LoRa and ZigBee devices. Thus, every ZigBee device must record the RSS signatures and perform the calibration, which maps each LoRa payload value to its own measured RSS signature, introducing significant overhead. The abovementioned problems motivate us to identify a lightweight feature that can be easily extracted from the RSS signature and used reliably to decode the LoRa packet payload. The selected feature must not depend on the distance between the LoRa and ZigBee devices. Therefore, only one ZigBee device in the network performs the calibration and then shares the mapping between LoRa payload values and RSS signatures to other devices.

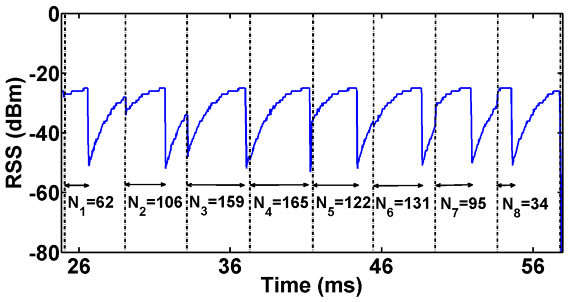


Fig. 11. Example RSS signature captured when LoRa transmits 0×01 with eight LoRa chirps. The time duration between the start of LoRa data chirps and their corresponding RSS drop are marked.

Table 2. The Eight Number of RSS Samples N_i between the Starts of Data Chirps and the Sudden RSS Value Drops (SF = 10, BW = 250 KHz, CR = 4/5, and CRC = off)

Payload	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8
0×01	62	106	159	165	122	131	95	34
$0 \times 2F$	66	88	152	163	128	134	95	36
0×33	16	87	161	167	124	133	72	132
0×34	21	104	151	167	124	134	72	132
$0 \times FF$	13	85	170	170	126	132	94	132

We observe that there always exists a sudden drop in the measured RSS values during the transmission of each LoRa chirp. This is because the LoRa's CSS modulation requires the radio to gradually increase its operating frequency and wrap around to f_{min} when it reaches f_{max}. When the LoRa and ZigBee channels overlap partially, the RSS measurement experiences a significant decrease when LoRa begins to use the frequency located outside the ZigBee channel. Since LoRa uses the different starting frequency of data chirp signal to encode different information, the time of those sudden drops in the RSS measurements depends on the data in the LoRa packet payload. Figure 11 plots an example of RSS signature with the marked time duration between the starts of data chirps and their corresponding sudden RSS value decreases. Our ZigBee device generates 177 RSS samples during the transmission of a LoRa chirp. We mark the number of RSS samples between the start of each data chirp and the sudden RSS value drop N_i ($i \in [1, 8]$) in Figure 11. It is important to note that this feature neither relies on the absolute RSS values nor depends on the distance between the LoRa and ZigBee devices. Therefore, only one ZigBee device in the network performs the calibration and then shares the mapping between LoRa payload values and RSS signatures to other devices. Table 2 lists some example N_i records when LoRa transmits different bytes. The 10 LoRa upchirps for preamble are used by ZigBee to synchronize its clock and identify the start of each LoRa data chirp. Please note that a low RSS sampling rate of RSS may decrease the number of distinguishable RSS signatures and the ZigBee device can detect all LoRa signatures when its RSS sampling rate is larger than the LoRa chip rate.

Observation 6. *The eight*[3] *numbers of RSS samples that capture the sudden RSS value drops can be used as the feature to identify the RSS signature.*

[3]LoRa may use more than eight data chirps to carry one byte in its packet payload. The number is decided by Equation (2) (see Section 5.2).

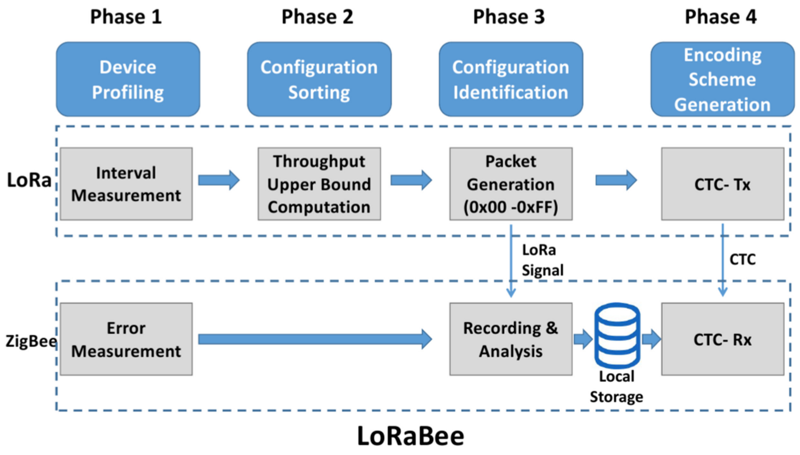


Fig. 12. LoRaBee design overview.

5 LORABEE DESIGN

In this section, we introduce the design of our LoRaBee. Figure 12 shows the overview of how LoRaBee generates the encoding scheme for the given LoRa and ZigBee devices. The process consists of four phases including *Device Profiling*, *Configuration Sorting*, *Configuration Identification*, and *Encoding Scheme Generation*.

In the first phase, LoRaBee measures the hardware and software capabilities of the given ZigBee and LoRa devices (Section 5.1). In the second phase, LoRaBee computes and sorts the upper bound of theoretical throughput from LoRa to ZigBee under different LoRa configurations (Section 5.2). In the third phase, LoRaBee identifies the LoRa configuration that provides the maximum actual throughput (Section 5.3). In the final phase, LoRaBee generates the encoding scheme for the given devices (Section 5.4).

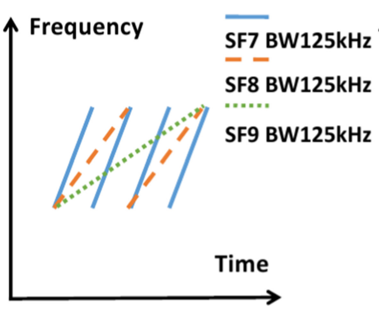
5.1 Device Profiling

LoRaBee first controls the LoRa and ZigBee devices to perform experiments that quantify the inaccuracy of feature measurements. Specifically, the LoRa device transmits the same packet multiple times, while the ZigBee device records the feature ($\{N_i | 1 \leq i \leq N_{chirp}\}$) of each RSS signature, i.e., the number of RSS samples (N_i) between the start of the ith data chirp and the following sudden RSS value drop. The maximum variation of those features, denoted as $var(N)$, is recorded by LoRaBee to serve as the guard space among RSS signatures[4] (see Section 5.3). LoRaBee then measures the minimal time interval T_g (software delay) between two consecutive packets transmitted by the given LoRa device. T_g is used to compute the upper bound of theoretical throughput from LoRa to ZigBee in Section 5.2.

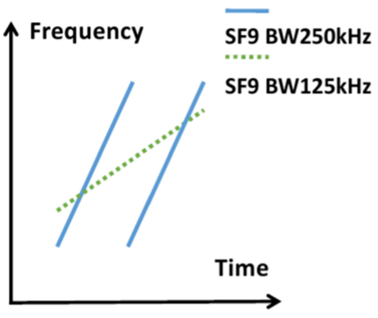
5.2 Configuration Sorting

The selection of LoRa physical-layer parameters including SF, BW, CR, and CRC, namely a LoRa configuration, makes a significant impact on the CTC throughput. According to Equation (1), the time duration of transmitting a LoRa chirp is decided by SF and BW. As Figure 13 shows, the time duration of transmitting a LoRa chirp doubles every time SF increases by one, while it is reduced by half when BW doubles. LoRa transmits the chirps faster when using a smaller SF and a larger BW. CR and CRC decide how many chirps LoRa uses to transmit a data byte. Either adding more

[4]We use the maximum variation of the LoRa chirp's features $var(N)$ as the guard space to achieve high CTC reliability.



(a) Impact of *SF*.



(b) Impact of *BW*.

Fig. 13. Impact of *SF* and *BW* on the time duration of transmitting LoRa chirps.

redundancy by using a smaller *CR* or enabling the *CRC* check (adding 16 bits) reduces the LoRa throughput. The selection of those parameters also makes a significant impact on how many RSS features can be distinguished by the ZigBee device.

The number of data chirps (N_{chirp}) in each LoRa packet can be calculated as follows:

$$N_{chirp} = 8 + max\left(\left\lceil\frac{8PL - 4SF + 8 + CRC + H}{4(SF - DE)}\right\rceil * \frac{4}{CR}, 0\right), \tag{2}$$

where PL is the LoRa payload size in bytes, CRC is either 16 if the CRC check is enabled or 0 otherwise, H is the size of LoRa packet header, and DE is either 2 if $SF \in \{11, 12\}$ or 0 otherwise. Thus, the on-air time of a LoRa packet (T_s) can be calculated as follows:

$$T_s = (N_{chirp} + 12.25) * \frac{2^{SF}}{BW}, \tag{3}$$

where $N_{chirp} + 12.25$ represents the total number of LoRa chirps carrying the packet.

With the minimal inter-packet time interval T_g (see Section 5.1), the upper bound of theoretical throughput from LoRa to ZigBee, which LoRaBee provides, is

$$D_{bound} = \frac{8}{T_s + T_g}, \tag{4}$$

where 8 is the multiplication of the time (1 s) and the number of bits (8 bits) in each packet.

With Equations (2), (3), and (4), LoRaBee can compute the upper bound of throughput D_{bound}, which it provides under each LoRa configuration ($6*3*4*2 = 144$ configurations in total). LoRaBee then sorts all configurations based on their D_{bound} values in the descending order (denoted as $\{D_{bound}[i] | 1 \leq i \leq 144\}$).

Please note that the D_{bound} values are calculated with the assumption that the ZigBee device can distinguish all possible bytes (0×00-$0 \times FF$) from its measured RSS features. According to our Observation 5 in Section 4, a ZigBee device may not be able to distinguish all of them due to its insufficient RSS sampling accuracy. Therefore, LoRaBee must compute the actual throughput D_{actual} under different configurations and then identify the best one that provides the maximum D_{actual} (see Section 5.3).

5.3 Configuration Identification

Since a ZigBee device may not be able to distinguish all possible bytes from its measured RSS features, LoRaBee defines

$$D_{actual}[i] = \alpha_i * D_{bound}[i], \tag{5}$$

ALGORITHM 1: Configuration Identification Algorithm

```
Input  : $\{D_{bound}[i] | 1 \le i \le 144\}$
Output : $D_{max}$, $index$, $Feature_{select}[][]$
$D_{max} = 0$, $index = 0$, $Feature_{select}[][]=\{0\}$;
for $i = 1; i \le 144; i++$ do
    Run Algorithm 2 to get $\alpha_i$ and $Feature_i[][]$;
    $D_{actual}[i] = \alpha_i * D_{bound}[i]$
    if $D_{actual}[i] > D_{max}$ then
        $index$ = i;
        $D_{max} = D_{actual}[i]$;
        Copy $Feature_i[][]$ to $Feature_{select}[][]$;
    end
    if $D_{max} \ge D_{bound}[i+1]$ then
        Output $index$, $D_{max}$, and $Feature_{select}[][]$;
        break;
    end
end
```

where $\alpha_i \in (0, 1]$ denotes the throughput loss ratio and i represents one of the 144 LoRa configurations. Algorithm 1 shows our configuration identification algorithm. The input of Algorithm 1 is the sorted throughput upper bound $\{D_{bound}[i] | 1 \le i \le 144\}$, obtained from Configuration Sorting (Section 5.2). The output of Algorithm 1 contains the maximum actual throughput (D_{max}), the index of the selected configuration ($index$), and the corresponding RSS distinguishable features ($Feature_{select}[][]$). Algorithm 1 first initializes D_{max}, $index$, and $Feature_{select}[][]$ to zero (line 1). Then it computes α_i by running Algorithm 2 and $D_{actual}[i]$ (lines 3 and 4) under each configuration i until D_{max} is not less than $D_{bound}[i+1]$ (lines 10–13). The loop terminates, since the rest configurations cannot provide higher throughput. Because the maximum actual throughput is already larger than or equal to the rest theoretical throughput upper bound values. This design is to reduce overhead.

Algorithm 2 shows the process that computes α_i under each configuration i. The input of Algorithm 2 is the maximum variations of RSS signature features $var(N)$ (see Section 5.1). LoRaBee first coordinates the LoRa and ZigBee devices to run control experiments to collect all RSS signature features $\{F[m][n] | 1 \le m \le 256, 1 \le n \le N_{chirp}\}$, storing N_{chirp} records for each possible data byte. Specifically, the LoRa device transmits packets each of which contains a byte from 0×00 to $0 \times$ FF. During the transmission of each LoRa packet, the ZigBee device records the numbers of RSS samples N_i between the starts of data chirps and the sudden RSS value drops. After obtaining $F[][]$, LoRaBee performs a similarity test to compute α_i (lines 3–21). In Algorithm 2, the outside loop goes through all the elements in $F[][]$ (lines 3–21). The inside loop checks whether the current feature is indistinguishable from the features that have already been selected (lines 4–14). If not, then the feature is added into $Feature_i[][]$ (lines 15–19). Otherwise, it is discarded. Each element in $Feature_i[][]$ stores the mapping from a LoRa payload byte (stored in $Feature_i[][0]$) to its corresponding N_{chirp} feature values in $Feature_i[][1]$, $Feature_i[][2], \ldots, Feature_i[][N_{chirp}]$. The actual number of bits that can be carried by in each LoRa packet to ZigBee depends on the size of $Feature_i[][n]$ array (denoted as $size$). Algorithm 2 computes α_i as follows:

$$\alpha = \frac{D_{actual}}{D_{bound}} = \frac{\lfloor \log_2 size \rfloor}{8}, \tag{6}$$

ALGORITHM 2: α_i and $Feature_i[][]$ Computation Algorithm

```
Input  : $var(N)$
Output : $\alpha_i$, $Feature_i[][]$
1  Run experiments to collect RSS signature features $F[][]$ under the current configuration;
2  $size = 0$, $count = 0$, $flag = true$, $Feature_i[][] = \{0\}$;
3  for $k = 1; k \leq 256; k++$ do
4      for $j = 1; j \leq size; j++$ do
5          for $l = 1; l \leq N_{chirp}; l++$ do
6              if $| Feature_i[j][l] - F[k][l] | \leq var(N)$ then
7                  $count++$;
8              end
9          end
10         if $count == N_{chirp}$ then
11             $flag = false$;
12         end
13         $count = 0$;
14     end
15     if $flag == true$ then
16         Copy $F[k][]$ to $Feature_i[size][]$;
17         $Feature_i[size][0] = k$;
18         $size++$;
19     end
20     $flag = ture$;
21 end
22 $\alpha_i = \frac{\lfloor \log_2^{size} \rfloor}{8}$;
23 Output $\alpha_i$ and $Feature_i[][]$;
```

where $\lfloor \log_2 size \rfloor$ represents the number of bits that can be carried in each LoRa 1-byte packet by LoRaBee. Algorithm 2 outputs α_i and $Feature_i[][]$, which are used by Algorithm 1.

5.4 Encoding Scheme Generation

After finding the LoRa configuration that provides the maximum throughput, LoRaBee starts to generate the encoding scheme. Since only $size$ bytes among 256 possible ones (0×00-$0 \times$ FF) can be distinguished by the ZigBee device, LoRaBee uses the first $2^{\lfloor \log_2^{size} \rfloor}$ distinguishable bytes to transmit the decimal values between 0 and $2^{\lfloor \log_2^{size} \rfloor} - 1$ with $\lfloor \log_2^{size} \rfloor$ bits. Therefore, LoRaBee uses the first $2^{\lfloor \log_2^{size} \rfloor}$ values in $Feature_{select}[][0]$ to encode data.

At runtime, LoRaBee first performs the segmentation by dividing the input data into pieces, each of which has $\lfloor \log_2^{size} \rfloor$ bits, and then transmits those pieces one by one. The LoRa and ZigBee devices use $Feature_{select}[][]$ to encode and decode the information. For example, the LoRa device puts the value $Feature_{select}[x][0]$ in the packet payload if it wants to transmit x, while the ZigBee device decodes x when it detects the match between the measured RSS feature and $\{Feature_{select}[x][i] | 1 \leq i \leq N_{chirp}\}$. LoRaBee reassembles the data pieces at the ZigBee device. The encoding scheme can be encrypted and shared between ZigBee devices, which prevents the adversaries from obtaining the CTC information.

Because of signal attenuation and interference, the ZigBee device may get some wrong values in the RSS signature feature. LoRaBee may still be able to decode the information by using the rest N_i. Algorithm 3 shows the algorithm that is used by LoRaBee to decode information.

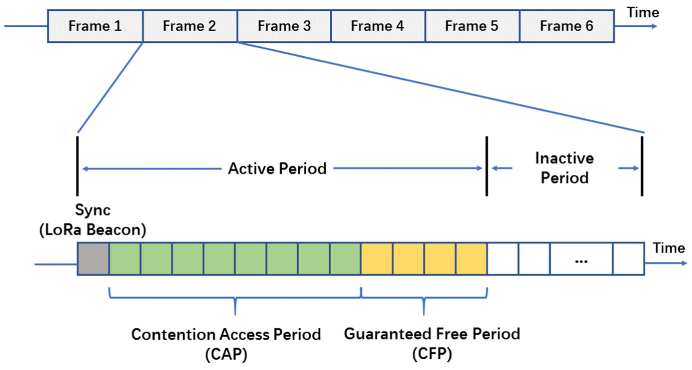


Fig. 14. TDMA frame structure.

6 CTC-BASED MAC PROTOCOL

In this section, we introduce the design of our CTC-based MAC protocol, which allows the LoRa to deliver data to a network of ZigBee devices using LoRaBee. We first present our TDMA frame structure design, and then discuss our CTC-based time synchronization method that replaces the flooding-based time synchronization method used in the ZigBee network. Finally, we present our scheduling approach that assigns time slots for time synchronization, unicast and broadcast from LoRa to ZigBee, and transmissions between ZigBee devices.

6.1 TDMA Frame Structure

Our TDMA frame structure, as Figure 14 shows, is designed to take advantage of the CTC from LoRa to ZigBee. We develop our TDMA frame structure by extending the IEEE 802.15.4 MAC. Time is divided into fixed-length time slots and several time slots are grouped together to form a slotframe. Slotframes are concatenated and repeat over time. The LoRa device uses the first time slot in each slotframe to broadcast its beacon.[5] In addition to broadcasting information, the beacons are used for time synchronization (see Section 6.2). All packet deliveries from LoRa/ZigBee to ZigBee take place in the active period that consists of **contention access period (CAP)** and **contention free period (CFP)**. To support CTC from LoRa to ZigBee, the length of a time slot (T_{slot}) must be

$$T_{slot} \geq T_{guard} + (N_{chirp} + 12.25) * \frac{2^{SF}}{BW}, \tag{7}$$

where T_{guard} is the guard time that addresses the clock drift issue between devices and $(N_{chirp} + 12.25) * \frac{2^{SF}}{BW}$ denotes the on-air time of a LoRa packet. The time slots in CFP can only be used by their owners and all devices can compete the use of the time slots in CAP in a CSMA fashion. Our CTC-based MAC protocol supports both unicast and broadcast. The transmission scheduling will be presented in Section 6.3.

6.2 Time Synchronization

Time synchronization is critical for all TDMA-based MAC protocols. The flooding-based method requires all devices in the network to periodically flood beacons, which can be used to adjust their local clocks. This largely increases the energy consumption and occupies a significant amount of

[5]Our MAC protocol can be used to support multiple LoRa devices that are time-synchronized. The first time slot in each slotframe is assigned to a single LoRa device to address the contention issue.

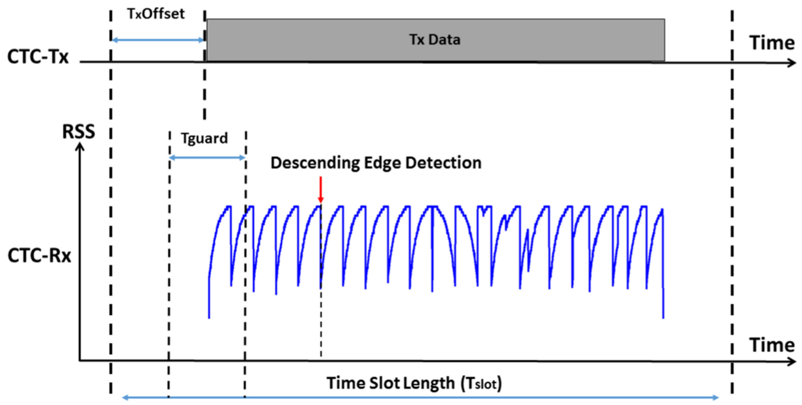


Fig. 15. Example RSS signature recorded in a time slot.

ALGORITHM 3: LoRaBee Decoding Algorithm

```
Input  : Input feature ($Input[]$)
Output : Decoded Result ($R$)
$flag = true$;
for $j = 1; j \le m; j++$ do
    for $l = 1; l \le N_{chirp}; l++$ do
        if $| Feature_{select}[j][l] - Input[l] | > var(N)$ then
            $flag = false$;
        end
    end
    if $flag == true$ then
        $R = j$;
        Output decoded result $R$;
        break;
    end
    $flag = true$;
end
```

time slots. To reduce the time synchronization overhead, we develop a method to use the CTC signals generated by the LoRa device for time synchronization. Specifically, we use the LoRa preamble (i.e., the first several upchirps) for time synchronization instead of the LoRa data chirps. The LoRa data chirps can be used to broadcast information to ZigBee devices. Figure 15 shows an example RSS signature measured in a time slot by a ZigBee device when the LoRa device delivers a beacon. The time slot length is T_{slot} and the guard time T_{guard} is used to accommodate the possible clock drift between receiving two consecutive beacons. The LoRa device transmits after the time offset $TxOffset$ in each time slot scheduled to transmit a beacon. Each ZigBee device records the time when it captures the descending edge of the preamble ($T_{descending}$) as the reference for time synchronization. The preamble length is a configurable parameter in LoRa ranging from 6 to 65,535 symbols. In our implementation, we use the fifth descending edge for time synchronization, as shown in Figure 15. The time when the current slotframe starts (T_b) can be calculated as follows:

$$T_b = T_{descending} - TxOffset - 5 * \frac{2^{SF}}{BW}, \tag{8}$$

where $T_{descending}$ is the time when the ZigBee devices capture the descending edge, $TxOffset$ is the transmission offset, and $\frac{2^{SF}}{BW}$ is the time duration of a single LoRa upchirp. Each ZigBee device can use Equation (8) to derive when the slotframe starts and then adjust its clock by comparing the global time with local time.

We use an example to illustrate the effectiveness of our method on reducing the time synchronization overhead. Orchestra [8] uses a flooding-based method for time synchronization, thus it has to reserve n time slots to flood beacons across the network consisting of n ZigBee devices in each slotframe. While using our method, it only needs to use one time slot to deliver a LoRa beacon for time synchronization.

6.3 Transmission Scheduling

We develop a scheduling approach that assigns time slots for time synchronization, unicast and broadcast from LoRa to ZigBee, and transmissions between ZigBee devices. Specifically, our scheduling approach assigns dedicated time slots on a fixed channel to deliver LoRa beacons and transmissions from LoRa to ZigBee. The ZigBee devices can use the rest channels and time slots for their transmissions. Here are the key scheduling rules of our approach:

Assigning Time Slots to Deliver LoRa Beacons: When a ZigBee device attempts to join the network, it first switches to the dedicated channel that is partially overlapped with the LoRa channel, and samples the RSS in the air to capture the LoRa beacon for time synchronization. In our implementation, we use the first time slot in the active period to deliver the LoRa beacon.

Assigning Time Slots for Unicasting: A hash function is used to allow different ZigBee devices to listen to the LoRa CTC packets in different dedicated time slots. The time slot used by a ZigBee device to receive CTC is determined by its unique node **id (ID)**. The LoRa device uses the sth time slot to send information to the ZigBee device with ID:

$$s = (ID - ID_{min})\%(CFP_{end} - CFP_{begin} + 1) + CFP_{begin}, \tag{9}$$

where CFP_{begin} is the slot number of the first time slot in CFP, CFP_{end} is the slot number of the last time slot in CFP, and ID_{min} is the minimal node id in the network. We only use the CFP time slots for CTC unicast.

Assigning Time Slots for Broadcasting: The last time slot in CFP is reserved for broadcast. All Zigbee devices can switch to the dedicated channel and wake up at the same time to receive CTC signals broadcasted by the LoRa device.

Assigning Time Slots for ZigBee Transmissions: The ZigBee devices use the rest channels for transmission between them. During CAP time slots, all ZigBee devices compete for channel access in a CSMA fashion. The unoccupied CFP time slots can also be used for contention-free communication.

7 EVALUATION

To validate the efficiency of our LoRaBee in enabling the CTC from LoRa to ZigBee, we perform a series of experiments. We first perform microbenchmark experiments to validate our design and evaluate the capability of LoRaBee to effectively identify the best LoRa configuration, which provides the maximum throughput. We also evaluate the efficiency of LoRaBee's encoding and decoding processes. We then perform experiments to quantify the **bit error rate (BER)** of LoRaBee under different link distances in indoor and outdoor environments and repeat the experiments under controlled interference. Then, we study the impact of retransmissions on LoRaBee.

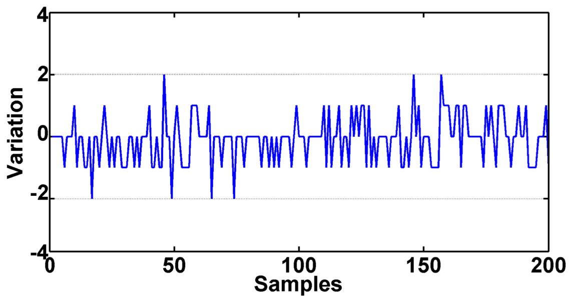


Fig. 16. Variations of measured RSS signature features from the median value. $var(N) = 2$.

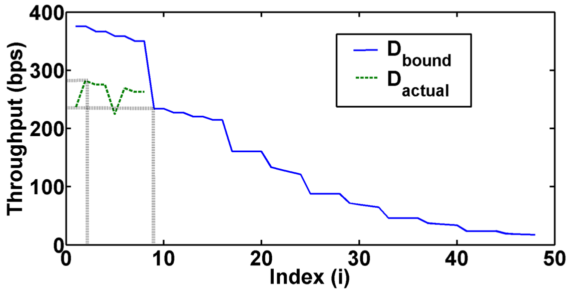


Fig. 17. Theoretical upper bound throughput D_{bound} vs. actual throughput D_{actual}.

7.1 Microbenchmark Experiments

In the Device Profiling phase, LoRaBee coordinates the LoRa and ZigBee devices to perform controlled experiments to measure the variations of the features extracted from the RSS signatures. Figure 16 plots some example variations deviating from the median value measured on our ZigBee device. We observe that the maximum variation $var(N)$ is 2 from all traces and using a smaller value for $var(N)$ significantly increases the bit error rate. For example, the BER increases to 21.60% when $var(N)$ is set to 1. LoRaBee also measures the minimum inter-packet time interval (T_g) between two consecutive LoRa packets. T_g of our LoRa device is 8.33 ms. With those two parameters, LoRaBee can compute the theoretical upper bound throughput $D_{bound}[i]$ under each LoRa configuration i in the Configuration Sorting phase. Table 3 lists the computed D_{bound} values under each LoRa SF, CR, and CRC combination.[6]

After obtaining the D_{bound} values, LoRaBee runs control experiments to measure the actual throughput (D_{actual}) in the Configuration Identification phase. According to Equation (5), the throughput loss ratio α_i is less than or equal to 1. The actual throughput cannot be higher than the theoretical upper bound. To reduce overhead, LoRaBee examines the LoRa configurations based on their D_{bound} values in the descending order and stops the experiments if $D_{bound}[i + 1]$ is not greater than the maximum D_{actual} under the first i configurations. Figure 17 plots the theoretical throughput upper bound D_{bound} and the actual throughput D_{actual} under different configurations. LoRaBee finds the maximum throughput of 281.61 bps when LoRa uses the second configuration ($SF = 7$, $CRC = on$, $CR = 4/5$, $BW = 250$ kHz). LoRaBee stops the measurements after obtaining D_{actual} under the eighth configuration, since the rest configurations cannot provide higher throughput. Please note that the CTC throughput from LoRa to ZigBee is lower than the ones

[6]We omit the values when BW is 125 kHz or 500 kHz due to the page limit.

Table 3. Theoretical Throughput Upper Bound D_{bound} under Different LoRa Configurations When BW Is 250 KHz

SF	CRC	CR	D_{bound} (bps)	Index	SF	CRC	CR	D_{bound} (bps)	Index
7	off	4/5	375.48	1	10	off	4/5	87.60	25
7	on	4/5	375.48	2	10	off	4/6	87.60	26
7	off	4/6	366.67	3	10	off	4/7	87.60	27
7	on	4/6	366.67	4	10	off	4/8	87.60	28
7	off	4/7	358.26	5	10	on	4/5	71.55	29
7	on	4/7	358.26	6	10	on	4/6	69.02	30
7	off	4/8	350.23	7	10	on	4/7	66.67	31
7	on	4/8	350.23	8	10	on	4/8	64.47	32
8	off	4/5	233.68	9	11	off	4/5	45.91	33
8	on	4/5	233.68	10	11	off	4/6	45.91	34
8	off	4/6	226.89	11	11	off	4/7	45.91	35
8	on	4/6	226.89	12	11	off	4/8	45.91	36
8	off	4/7	220.49	13	11	on	4/5	37.17	37
8	on	4/7	220.49	14	11	on	4/6	35.81	38
8	off	4/8	214.44	15	11	on	4/7	34.54	39
8	on	4/8	214.44	16	11	on	4/8	33.36	40
9	off	4/5	160.48	17	12	off	4/5	23.52	41
9	off	4/6	160.48	18	12	off	4/6	23.52	42
9	off	4/7	160.48	19	12	off	4/7	23.52	43
9	off	4/8	160.48	20	12	off	4/8	23.52	44
9	on	4/5	133.13	21	12	on	4/5	18.95	45
9	on	4/6	128.75	22	12	on	4/6	18.25	46
9	on	4/7	124.64	23	12	on	4/7	17.59	47
9	on	4/8	120.78	24	12	on	4/8	16.98	48

Table 4. D_{actual} under the First Eight LoRa Configurations

Config	Distinguishable RSS signatures	D_{actual} (bps)
1	59/256	234.67
2	72/256	281.61
3	70/256	275.00
4	96/256	275.00
5	61/256	223.91
6	102/256	268.69
7	87/256	262.67
8	107/256	262.67

among WiFi, Bluetooth, and ZigBee, since LoRa provides much lower physical bit rates ranging from 250 bps to 11 kbps under various configurations. Table 4 lists the number of distinguishable RSS signatures and D_{actual} under the first eight LoRa configurations. As Table 4 shows, many RSS signatures are not distinguishable due to the insufficient RSS sampling accuracy of the ZigBee device. For example, the ZigBee device can only identify 72 among 256 RSS signatures under

Table 5. Similar RSS Signature Features Collected When $SF = 7$, $BW = 250$ KHz, $CR = 4/5$, and $CRC = on$

Payload	RSS Signature Features N_i
0×00	13 8 16 14 12 12 16 17 17 17 01 12 17
0×06	13 8 16 14 12 12 16 17 17 16 01 11 17
$0 \times 1B$	12 7 15 14 11 11 15 18 17 16 01 05 03
$0 \times 1D$	12 7 15 14 12 11 15 18 17 17 01 05 03
0×30	12 7 15 13 11 11 15 19 18 17 01 11 07
0×33	13 7 16 13 12 11 15 18 18 17 01 11 07
$0 \times AA$	12 6 15 13 11 10 14 16 16 15 01 16 16
$0 \times AF$	13 6 15 12 11 10 14 17 16 15 01 16 16
$0 \times E0$	12 7 15 13 11 11 15 16 17 16 10 11 17
$0 \times F3$	13 7 15 13 12 11 15 16 16 16 11 11 17

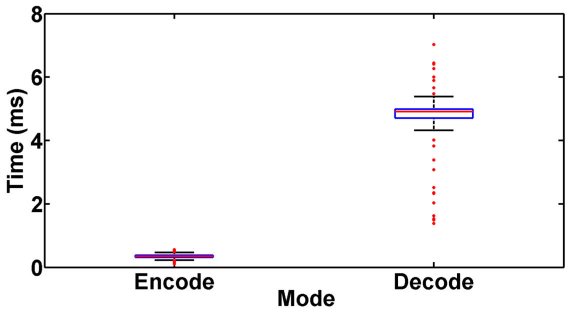


Fig. 18. Boxplot of the time consumed by LoRaBee to encode and decode information. The results are gathered from 200 experimental runs. Central red mark in box indicates median; bottom and top of box represent the 25th percentile (q_1) and 75th percentile (q_2); crosses indicate outliers ($x > q_2 + 1.5 * (q_2 - q_1)$ or $x < q_1 - 1.5 * (q_2 - q_1)$); whiskers indicate range excluding outliers.

the second configuration. Table 5 lists five pairs of indistinguishable RSS signature features whose differences are smaller than the error range $var(N) = 2$.

The results gathered from our microbenchmark experiments not only demonstrate the correctness of our LoRaBee design but also show that LoRaBee can efficiently identify the LoRa configuration, which provides the maximum throughput.

7.2 Encoding and Decoding Efficiency

We also measure the time consumed by LoRaBee to encode and decode the information on the LoRa and ZigBee devices. Figure 18 shows the boxplot of 200 measurements. On average, the LoRa device consumes 0.33 ms to encode a packet, while the ZigBee device uses 4.66 ms to extract the features from the measured RSS samples and decode information from them. The LoRa packet transmission time is not included in the result. The fast encoding and decoding speeds benefit from the lightweight feature that can be easily and accurately extracted from the RSS signature, demonstrating the efficiency of LoRaBee. Please note that the ZigBee device consumes a similar amount of power on sampling the RSS values and receiving packets. Thus, the energy consumption increase caused by LoRaBee is marginal.

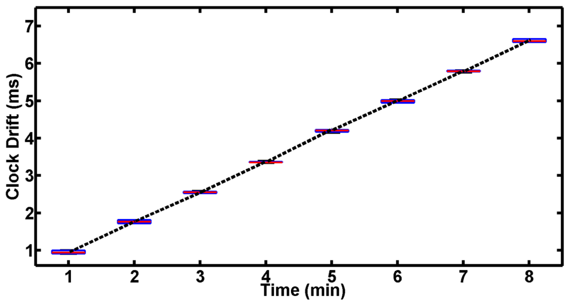


Fig. 19. Time desynchronized over time.

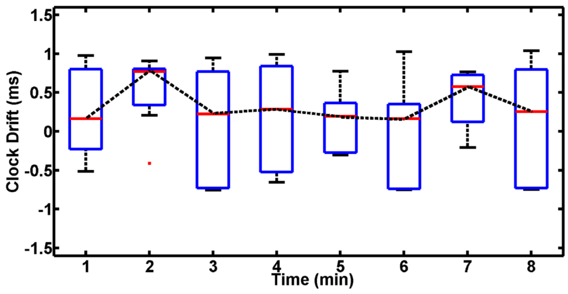


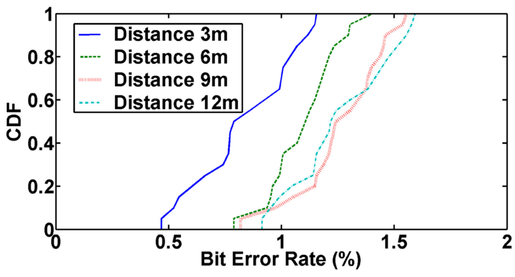
Fig. 20. Time Synchronization error between two ZigBee devices over time.

7.3 Time Synchronization Accuracy

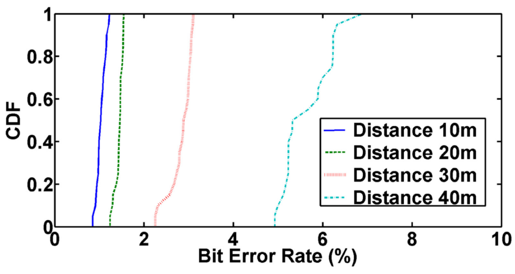
To validate our design of using LoRa signals to time synchronize ZigBee devices, we perform experiments to measure the time synchronization errors. In our experiments, we set the time slot length (T_{slot}) to 30 ms including a 3 ms guard time (T_{guard}). The time slot length is sufficient for a LoRa packet transmission (see Equation (7)). We configure LoRa to use $SF = 7$, $CRC = on$, $CR = 4/5$, $BW = 250$ kHz to maximize its throughput and transmits a beacon in every three seconds. The slotframe consists of 100 time slots. We use the first time slot in each slotframe to transmit the LoRa beacon. As Figure 19 plots, two ZigBee devices desynchronize in time very fast. After three minutes, the time difference is larger than the guard time. With the help of LoRa beacons, the median synchronization error is up to 0.77 ms and the maximum synchronization error is 1.03 ms, smaller than the half of our designed guard time ($T_{guard}/2$), as Figure 20 shows. The experimental results show that using LoRa signals can effectively keep those devices time synchronized.

7.4 Bit Error Rate

We then measure the BER under the best LoRa configuration, which provides the maximum throughput. We generate 1,500 random bytes in the hexadecimal format using an online random byte generator [13] and run LoRaBee to deliver them. We vary the distance between our LoRa and ZigBee devices ranging from 3 to 12 m in an indoor corridor and run the experiments for 20 times under each distance. Figure 21(a) plots the **cumulative distribution function (CDF)** of BER in an indoor environment. The maximum BER is 1.13% and the average is 0.82% when the devices are 3 m apart. The maximum BER slightly increases to 1.41%, 1.55%, and 1.59% when the link distance increases to 6, 9, and 12 m, respectively. The average BER under those four distances are 0.82%, 1.11%, 1.26%, and 1.28%. The slow increases indicate that the signal attenuation has a small impact on BER.



(a) Indoor environment.



(b) Outdoor environment.

Fig. 21. BER measurements in indoor and outdoor environments.

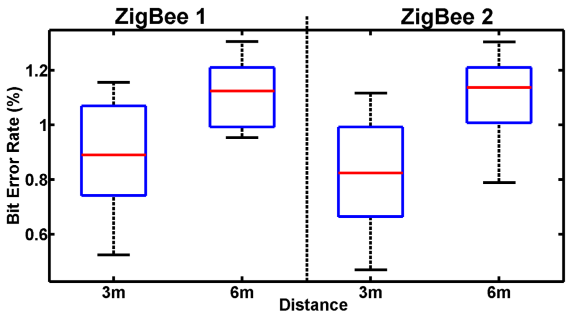


Fig. 22. Boxplot of the BER when a LoRa device transmits packets to multiple ZigBee devices.

We repeat the experiments in an outdoor environment. Figure 21(b) shows the CDF of BER. Similarly, we observe that BER increases with increasing distance. The average BERs are 1.05%, 1.44%, 2.86%, and 5.67% when the link distances are 10, 20, 30, and 40m, respectively. From the results, we can observe that BER increases slowly with link distance, indicating that signal attenuation slightly affects LoRaBee's performance. The results also show that LoRaBee demonstrates an acceptable performance ($BER \leq 1.61\%$). We repeat the experiments using devices with different battery levels and in different days with different temperature and humidity and observe little impact from those factors. LoRaBee always provides stable performance.

We also measure the BER when a LoRa device transmits packets to multiple ZigBee devices. Figure 22 plots the Boxplot of the BER when the LoRa device unicasts 500 bytes to two ZigBee devices three or six meters away. We schedule the CTC transmissions to use two different time slots based on our scheduling rules specified in Section 6.3 and repeat the experiments for 10 times. The median BERs measured on two devices are 0.89% and 0.83% when the LoRa and ZigBee devices are 3 m apart and they increase to 1.12% and 1.14% when the device distance is increased to 6 m. We then measure the BER when two LoRa devices transmit packets to a single ZigBee device in different slotframe. Figure 23 plots the CDF of the BER whe the LoRa devices transmit 500 bytes to one ZigBee device. Each LoRa device is 3 m away from the ZigBee device. Dedicated time slots have been assigned for CTC transmission and we repeat the experiments for 10 times. The median BERs measured are 1.05% and 0.97% for each LoRa device transmitting CTC packets to the ZigBee, respectively. The results show that our MAC protocol can successfully effectively avoid collisions and maintains high CTC reliability.

7.5 Impact of Interference

We also study the impact of interference on the BER of LoRaBee. We set up two pairs of LoRa and ZigBee devices: one pair in an indoor corridor and the other in an outdoor open space. We

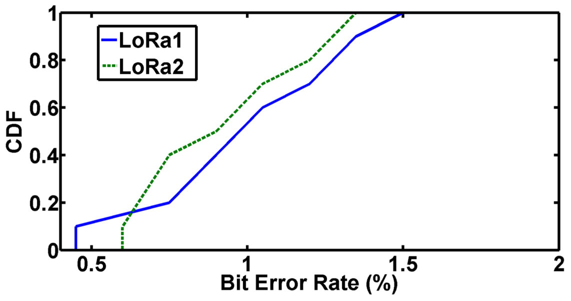


Fig. 23. CDF of the BER when two LoRa devices transmit packets to a ZigBee device.

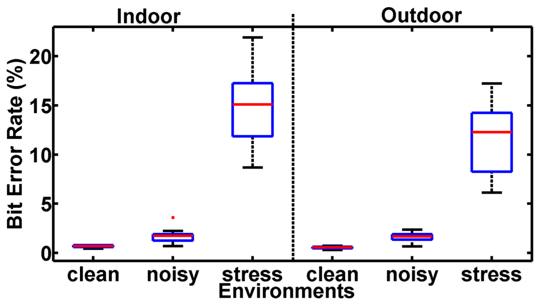
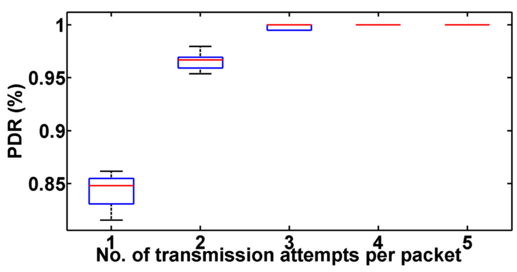


Fig. 24. Box plot of the BER of LoRaBee in the clean, noisy, and stress testing environments.

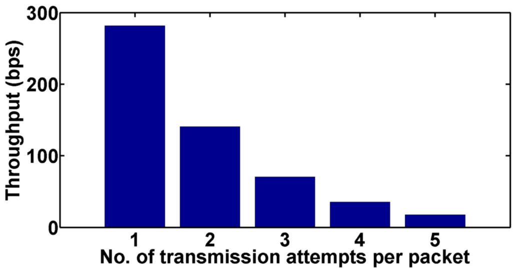
configure a TI CC1310 launchpad to generate controlled interference by transmitting back-to-back 64-Byte ZigBee packets in the central frequency of 915.6 MHz and vary the distance between the interferer and the LoRa and ZigBee device pair to create different interference conditions: clean, noisy, and stress test. The distance between the interferer and the LoRa and ZigBee device pair is 15, 8, and 5 m under the clean, noisy, and stress test, respectively. We measure the BER when the LoRa device transmits 500 bytes to the ZigBee device and repeats the experiments 10 times under each condition. Figure 24 shows the Boxplot of BER when the LoRa and ZigBee devices are three meter away. In the indoor environment, LoRaBee achieves median BERs of 0.67%, 1.72%, and 15.10% in clean, noisy, and stress test environments, respectively. In the outdoor environment, LoRaBee achieves median BERs of 0.54%, 1.66%, and 12.28% in clean, noisy, and stress test environments, respectively. The results show that LoRaBee consistently provides low BERs under moderate interference. The significant increases on BERs under strong interference emphasize the importance of employing an appropriate MAC protocol (e.g., a TDMA-based MAC) when using LoRaBee.

7.6 Impact of Retransmissions

To evaluate the impact of retransmissions on LoRaBee, we have performed the experiments with different number of transmission attempts. Figure 25(b) shows the performance of LoRaBee with different number of transmission attempts per packet when the devices are 6m apart in the corridor. As Figure 25(a) shows, the retransmissions successfully improve the median **packet delivery ratio (PDR)** from 81.54% to 100% when three transmission attempts are scheduled for each packet. All PDRs become 100% when four transmission attempts are scheduled for each packet. As Figure 25(b), the throughput decreases with more transmission attempts. The results show that the retransmissions effectively enhance the link reliability at the cost of reduced throughput.



(a) Reliability.



(b) Throughput.

Fig. 25. Performance with different No. of transmission attempts per packet.

8 CONCLUSIONS

In this article, we present LoRaBee, a novel CTC approach to enable the direct communication from LoRa to ZigBee. By elaborately tuning the LoRa's central carrier frequency and packet payload, a ZigBee device can decode the LoRa chirps by simply sensing the RSS. An empirical study has been performed to investigate the characteristics of LoRa communication from a CTC's point of view and a series of insights are distilled to guide our LoRaBee design. LoRaBee has been implemented and tested on real hardware. Experimental results show that our LoRaBee provides reliable CTC communication from LoRa to ZigBee with the throughput of up to 281.61 bps in the Sub-1 GHz bands in indoor and outdoor environments, which is enough for a LoRa base station to disseminate network management and urgent control messages to ZigBee devices, such as the periodic network management beacons transmitted by Orchestra [8] and the control messages generated by the coupled water tank monitoring system [21].

REFERENCES

[1] 802.15.4e. 2020. IEEE 802.15.4e WPAN Task Group. Retrieved from http://www.ieee802.org/15/pub/TG4e.html.
[2] Bluetooth. 1998. Bluetooth Technology. Retrieved from https://www.bluetooth.com/.
[3] businesswire. 2017. Multi-Band IoT Mesh Network Technology for Massive IoT Deployments. Retrieved from https://www.businesswire.com/news/home/20170621005029/en/The-Zigbee-Alliance-Introduces-First-Multi-Band-IoT-Mesh-Network-Technology-for-Massive-IoT-Deployments.
[4] CC1310. 2018. CC1310. Retrieved from http://www.ti.com/lit/ds/symlink/cc1310.pdf.
[5] K. Chebrolu and A. Dhekne. 2009. Esense: Communication through energy sensing. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom'09)*. Association for Computing Machinery, New York, NY, 85–96.
[6] Zicheng Chi, Yan Li, Hongyu Sun, Yao Yao, Zheng Lu, and Ting Zhu. 2016. B2W2: N-Way concurrent communication for IoT devices. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems (SenSys'16)*. Association for Computing Machinery, New York, NY, 245–258. https://doi.org/10.1145/2994551.2994561
[7] Semtech Corporation. 1960. Semtech. Retrieved from https://www.semtech.com/.
[8] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. 2015. Orchestra: Robust mesh networks through autonomously scheduled TSCH. In *Proceedings of the ACM Conference on Embedded Network Sensor Systems (Sensys'15)*. Association for Computing Machinery, New York, NY, 337–350.
[9] Piotr Gawlowicz, Anatolij Zubow, and Adam Wolisz. 2018. Enabling cross-technology communication between LTE unlicensed and WiFi. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'18)*. IEEE, 144–152.
[10] Chaojie Gu, Rui Tan, and Xin Lou. 2019. One-hop out-of-band control planes for multi-hop wireless sensor networks. *ACM Trans. Sens. Netw.* 15, 4 (Jul. 2019), 29. https://doi.org/10.1145/3342100
[11] Xiuzhen Guo, Yuan He, Xiaolong Zheng, Liangcheng Yu, and Omprakash Gnawali. 2018. ZIGFI: Harnessing channel state information for cross-technology communication. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'18)*. IEEE, 360–368.

[12] Xiuzhen Guo, Xiaolong Zheng, and Yuan He. 2017. WiZig: Cross-technology energy communication over a noisy channel. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'17)*. IEEE, 1–9.

[13] Mads Haahr. 1998. Random.org. Retrieved from https://www.random.org/bytes/.

[14] Cooking Hacks. 2016. SX1272 LoRa Shield for Raspberry Pi. Retrieved from https://www.cooking-hacks.com/sx1272-lora-shield-for-raspberry-pi-900-mhz.

[15] Richard Wesley Hamming. 1950. Error detecting and error correcting codes. *Bell Syst. Techn. J.* 2, 29 (1950), 147–160.

[16] T. Hao, R. Zhou, G. Xing, M. W. Mutka, and J. Chen. 2014. WizSync: Exploiting Wi-Fi infrastructure for clock synchronization in wireless sensor networks. *IEEE Trans. Mobile Comput.* 13, 6 (Jun. 2014), 1379–1392.

[17] Wenchao Jiang, Song Min Kim, Zhijun Li, and Tian He. 2018. Achieving receiver-side cross-technology communication with cross-decoding. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom'18)*. Association for Computing Machinery, New York, NY, 639–652. https://doi.org/10.1145/3241539.3241547

[18] Song Min Kim and Tian He. 2015. Freebee: Cross-technology communication via free side-channel. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom'15)*. Association for Computing Machinery, New York, NY, 317–330.

[19] S. M. Kim, S. Ishida, S. Wang, and T. He. 2017. Free side-channel cross-technology communication in wireless networks. *IEEE/ACM Trans. Netw.* 25, 5 (2017), 2974–2987.

[20] Silicon Labs. 2017. Silicon Labs EFR32MG12P433F1024GM48. Retrived from https://www.silabs.com/products/wireless/mesh-networking/efr32mg-mighty-gecko-zigbee-thread-soc/device.efr32mg12p433f1024gm48.

[21] Bo Li, Lanshun Nie, Chengjie Wu, Humberto Gonzalez, and Chenyang Lu. 2015. Incorporating emergency alarms in reliable wireless process control. In *Proceedings of the ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS'15)*. Association for Computing Machinery, New York, NY, 218–227.

[22] Zhijun Li and Tian He. 2017. WEBee: Physical-layer cross-technology communication via emulation. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom'17)*. Association for Computing Machinery, New York, NY, 2–14.

[23] LoRa. 2015. LoRa Alliance. Retrieved from https://lora-alliance.org/.

[24] Chenyang Lu, Abusayeed Saifullah, Bo Li, Mo Sha, Humberto Gonzalez, Dolvara Gunatilaka, Chengjie Wu, Lanshun Nie, and Yixin Chen. 2016. Real-time wireless sensor-actuator networks for industrial cyber-physical systems. *Proc. IEEE Spec. Issue Industr. Cyber Phys. Syst.* 104, 5 (2016), 1013–1024.

[25] Microchip. 2015. RN2903. Retrieved from http://ww1.microchip.com/downloads/en/DeviceDoc/50002390E.pdf.

[26] Raspberry Pi. 2016. Raspberry Pi 3 Model B. Retrieved from https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[27] Michael E. Porter and James E. Heppelmann. 2014. How smart, connected products are transforming competition. *Harv. Bus. Rev.* 92, 11 (2014), 64–88.

[28] Junyang Shi, Xingjian Chen, and Mo Sha. 2019. Enabling direct messaging from lora to ZigBee in the 2.4-GHz band for industrial wireless networks. In *Proceedings of the 2019 IEEE International Conference on Industrial Internet (ICII'19)*. IEEE, 180–189.

[29] Junyang Shi, Di Mu, and Mo Sha. 2019. LoRaBee: Cross-Technology Communication from LoRa to ZigBee via Payload Encoding. In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*. IEEE, Chicago, IL, USA, USA, 1–11.

[30] TICC1352R. 2018. TI CC1352R LaunchPad. Retrieved from http://www.ti.com/tool/launchxl-cc1352r1.

[31] H. Wang and A. O. Fapojuwo. 2017. A survey of enabling technologies of low power and long range machine-to-machine communications. *IEEE Commun. Surv. Tutor.* 19, 4 (2017), 2621–2639.

[32] Shuai Wang, Woojae Jeong, Jinhwan Jung, and Song Min Kim. 2020. X-MIMO: Cross-technology multi-user MIMO. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys'20)*. ACM, New York, NY, 218–231.

[33] S. Wang, S. M. Kim, and T. He. 2018. Symbol-level cross-technology communication via payload encoding. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS'18)*. IEEE, 500–510.

[34] S. Wang, Z. Yin, S. M. Kim, and T. He. 2017. Achieving spectrum efficient communication under cross-technology interference. In *Proceedings of the International Conference on Computer Communication and Networks (ICCCN'17)*. IEEE, 1–8.

[35] WiFi. 2000. WiFi Alliance. Retrieved from https://www.wi-fi.org/.

[36] Dan Xia, Xiaolong Zheng, Liang Liu, Chaoyu Wang, and Huadong Ma. 2020. c-Chirp: Towards symmetric cross-technology communication over asymmetric channels. In *Proceedings of the IEEE International Conference on Sensing, Communication, and Networking (SECON'20)*. IEEE, 1–9.

[37] S. Yin, Q. Li, and O. Gnawali. 2015. Interconnecting WiFi devices with IEEE 802.15.4 devices without using a gateway. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS'15)*. IEEE, 127–136.

[38] Zhimeng Yin, Wenchao Jiang, Song Min Kim, and Tian He. 2017. C-Morse: Cross-technology communication with transparent morse coding. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'17)*. IEEE, 1–9.

[39] Zihao Yu, Chengkun Jiang, Yuan He, Xiaolong Zheng, and Xiuzhen Guo. 2018. Crocs: Cross-technology clock synchronization for WiFi and zigbee. In *Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*. Junction Publishing, 135–144.

[40] Xinyu Zhang and Kang G. Shin. 2013. Cooperative carrier signaling: Harmonizing coexisting WPAN and WLAN devices. *IEEE/ACM Trans. Netw.* 21, 2 (2013), 426–439. https://doi.org/10.1109/TNET.2012.2200499

[41] X. Zhang and K. G. Shin. 2013. Gap Sense: Lightweight coordination of heterogeneous wireless devices. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'13)*. IEEE, 3094–3101.

[42] Yifan Zhang and Qun Li. 2013. HoWiES: A holistic approach to zigbee assisted WiFi energy savings in mobile devices. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'13)*. IEEE,1366–1374.

[43] Xiaolong Zheng, Yuan He, and Xiuzhen Guo. 2018. StripComm: Interference-resilient cross-technology communication in coexisting environments. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'18)*. IEEE, 171–179.

[44] Xiaolong Zheng, Dan Xia, Xiuzhen Guo, Liang Liu, Yuan He, and Huadong Ma. 2020. Portal: Transparent cross-technology opportunistic forwarding for low-power wireless networks. In *Proceedings of the 21st International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (Mobihoc'20)*. ACM, New York, NY, 241–250.

[45] ZigBee. 2002. Zigbee Alliance. Retrieved from https://zigbeealliance.org/.

[46] ZigBee. 2017. Zigbee Alliance Introduces Zigbee PRO 2017. Retrieved from https://www.automation.com/en-us/products/product03/zigbee-alliance-introduces-zigbee-pro-2017.

[47] A. Zubow, P. Gawlowicz, and S. Bayhan. 2018. On practical coexistence gaps in space for LTE-U/WiFi coexistence. In *Proceedings of the 24th European Wireless Conference*. VDE, 1–8.

Received November 2020; revised April 2021; accepted June 2021