LoRaBee: Cross-Technology Communication from LoRa to ZigBee via Payload Encoding

Junyang Shi, Di Mu, Mo Sha
Department of Computer Science
State University of New York at Binghamton
{jshi28, dmu1, msha}@binghamton.edu

***Abstract*—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 (LPWAN) 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 paper, 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 (RSS). Experimental results show that our LoRaBee provides reliable CTC communication from LoRa to ZigBee with the throughput of up to 281.61bps in the Sub-1 GHz bands.**

***Index Terms*—Cross-technology Communication, LoRa, ZigBee, Internet of Things**

I. 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 [1]. Many wireless technologies (e.g., ZigBee, WiFi, and Bluetooth) are readily available to form the networks which 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 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 [2]. 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 [3]. A recent study [4] shows that adding low-power wide-area network (LPWAN) radios such as LoRa [5] 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.

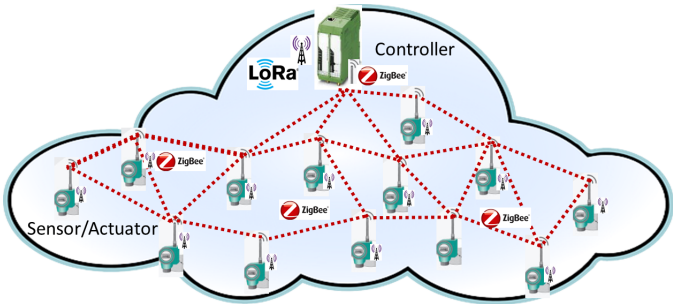


Fig. 1. A LPWMN equipping with LPWAN radios.

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 [6]–[21]. 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. In this paper, 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[1]. 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

[1]In this paper, we focus on enabling the unidirectional data delivery from LoRa to ZigBee and leave the MAC and routing protocol designs such as acknowledgement delivery and collision avoidance for future work.

strength (RSS). This design ensures full compatibility with the commercial off-the-shelf (COTS) LoRa and ZigBee devices. Specifically, this paper makes the following contributions:

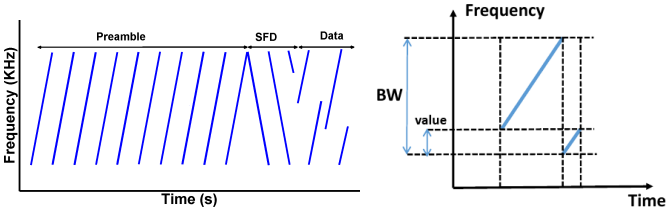
- To our knowledge, this is the first paper to investigate CTC from LoRa to ZigBee in the Sub-1 GHz bands, distinguished with previous work pertaining to CTC among WiFi, ZigBee, and Bluetooth devices in the 2.4 GHz band.
- This paper performs an empirical study that investigates the characteristics of LoRa from a CTC's point of view and provides a set of new observations.
- This paper 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.
- 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 281.61bps[2] in Sub-1 GHz bands.

The remainder of the paper is organized as follows. Section II reviews the related work and Section III discusses the background of LoRa and ZigBee. Section IV introduces our empirical study. Sections V presents the design of our LoRaBee. Section VI evaluates LoRaBee and Section VII concludes the paper.

II. 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 [6]–[9], [11]–[23]. 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 [18]. Zhang et al. developed GapSense which leverages the sequences of energy bursts to modulate symbol [15]. Kim et al. proposed FreeBee which adjusts the appearance of WiFi beacons in the time dimension to transmit modulated data [9], [10]. Yin et al. designed C-Morse which controls the presence of data traffic to deliver information [11]. 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 [6]. Li et al. developed WEBee which uses WiFi packets to directly emulate the ZigBee signals in the physical-layer [12]. Yin et al. proposed to use the presence and absence of energy profiles to convey information among heterogeneous wireless devices [19]. More recently, Zheng et al. developed StripComm which is an interference-aware CTC modulation and demodulation scheme [8]. Guo et al. developed ZigFi that uses channel state information to convey data from ZigBee to WiFi [7]. Yu et al. [22] and Hao et al. [23] proposed to use CTC for clock synchronization. Jiang et al. developed SymBee that achieves symbol-level CTC from ZigBee to WiFi [13]. Jiang et al. [14] and Chi et al. [20] enabled the CTC between ZigBee and Bluetooth devices. In contrast to previous studies on CTC among ZigBee, WiFi, and Bluetooth devices in the 2.4 GHz ISM bands, this paper 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.

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



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

Fig. 2. LoRa modulation.

III. Background

A. LoRa Overview

LPWANs are emerging as a new paradigm in the field of IoT connectivity [24]. LoRa is an industry LPWAN technology which has been initiated by Semtech 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 which 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 I lists the possible values for

TABLE I
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

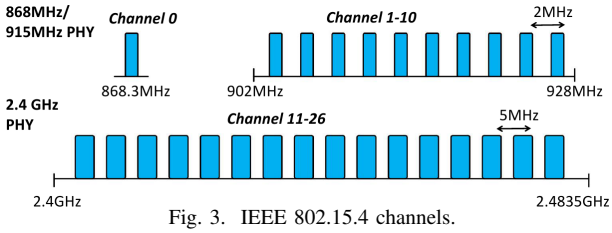


Fig. 3. IEEE 802.15.4 channels.

each parameter in the United States. f_c determines the central carrier frequency for data transmission[3]. BW determines the magnitude of frequency variation ($f_{max} - f_{min}$), representing the channel width. Each chirp consists of 2^{SF} chips which can carry SF bits of data. The time duration of one LoRa chirp is:

$$T_{chirp} = \frac{2^{SF}}{BW} \quad (1)$$

CR uses the Hamming code [25] 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, "0x6A" may be input into the LoRa transceiver to carry 106.

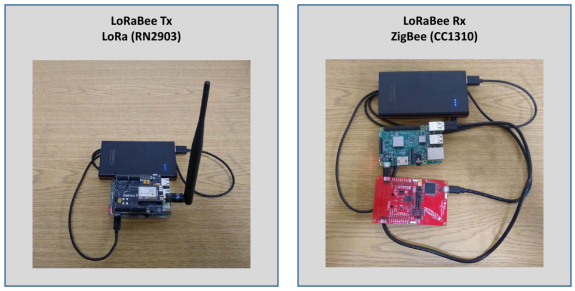
B. ZigBee Overview

ZigBee is based on the IEEE 802.15.4 standard, 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 [26] and Silicon Labs EFR32MG12P433F1024GM48 [27]) support operating in both Sub-1 GHz and 2.4 GHz bands. ZigBee uses Binary Phase Shift Keying (BPSK) modulation, which provides the throughput of up to 40kbps in the Sub-1 GHz bands.

IV. 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 guidelines for our CTC design. We perform the experiments with two Raspberry Pi 3 Model B [28]: one integrating with a SX1272 LoRa shield [29] containing a Microchip RN2903 radio [30], which is compatible with LoRa, and the other integrating with a TI CC1310 launchpad [31], which is compatible with ZigBee. Figure 4 shows the hardware.

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



(a) CTC sender. (b) CTC receiver.
Fig. 4. Hardware.

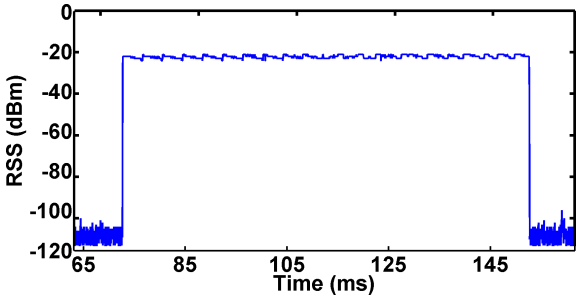


Fig. 5. An example RSS trace measured by ZigBee when LoRa and ZigBee channels overlap completely. ZigBee operates on channel 6 with the central frequency of 916 MHz. LoRa transmits a packet with the content of 0x00 using the same central frequency with $BW = 250$ KHz, $SF = 10$, $CR = 4/5$, and $CRC = off$.

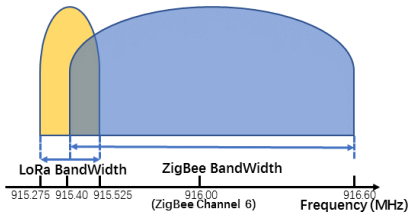
A. Energy Profiling of LoRa Signals on ZigBee

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

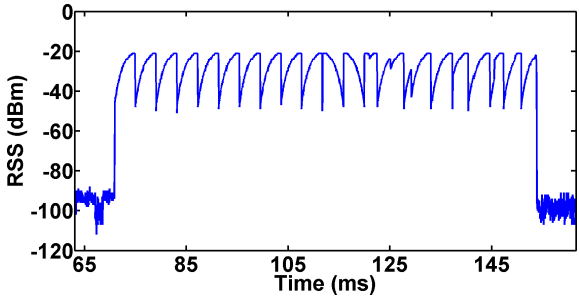
Observation 1*: ZigBee can capture the energy emission from LoRa, but cannot detect the individual LoRa chirps when the LoRa and ZigBee channels overlap completely.*

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 68.60ms, the RSS measured by ZigBee immediately increases and varies from -51dBm to -21dBm 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*



(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 0x00 using the central frequency of 915.4 MHz with $BW = 250$ KHz, $SF = 10$, $CR = 4/5$, and $CRC = off$.

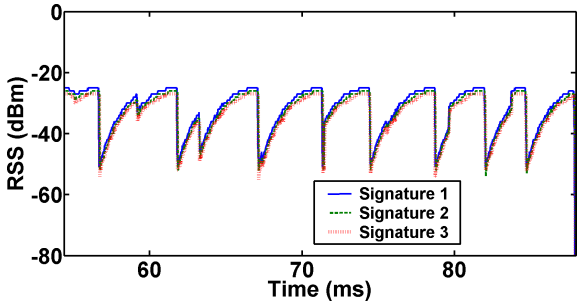


Fig. 7. RSS signatures measured by ZigBee when LoRa transmits packets with the same payload, which contains one byte (0x01).

its RSS measurements when the LoRa and ZigBee channels overlap partially.

The Observation 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.

B. LoRa Payload Encoding

In this set of experiments, we investigate the feasibility of decoding the LoRa packet payload from the measured

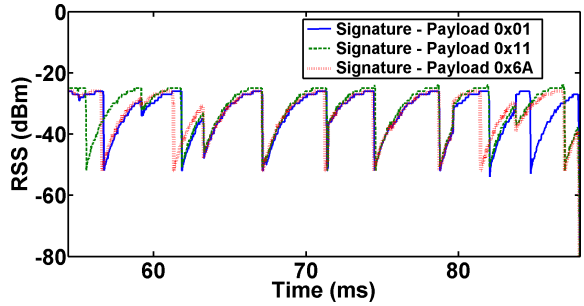


Fig. 8. RSS signatures measured by ZigBee when LoRa transmits 0x01, 0x11, and 0x6A, respectively.

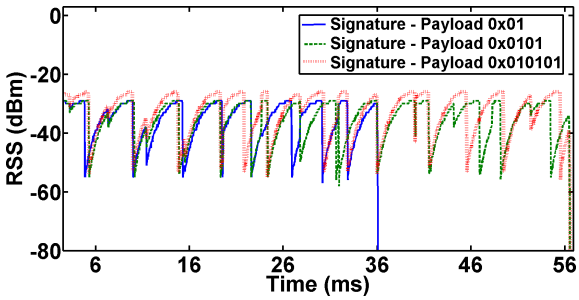


Fig. 9. RSS signatures measured by ZigBee when LoRa transmits 0x01, 0x0101, and 0x010101, respectively.

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 0x01 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 0x01, 0x11, and 0x6A, respectively. The differences between the three RSS signatures are noticeable. Please note that LoRa preprocesses data by performing data whitening (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 0x01, 0x0101, and 0x010101, respectively. The RSS signatures are completely different. This is because LoRa rearranges the bits in the packet payload before transmitting them. The bytes in the packet payload are not directly concatenated, resulting in the distinct RSS signatures.

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

The Observation 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

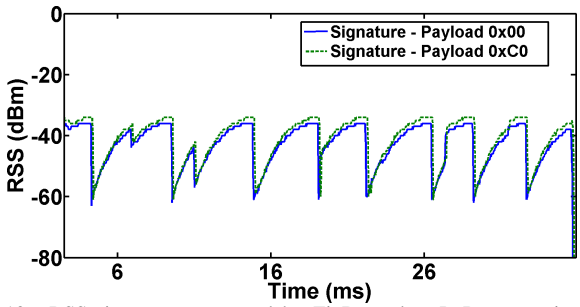


Fig. 10. RSS signatures measured by ZigBee when LoRa transmits 0x00, and 0xC0, respectively.

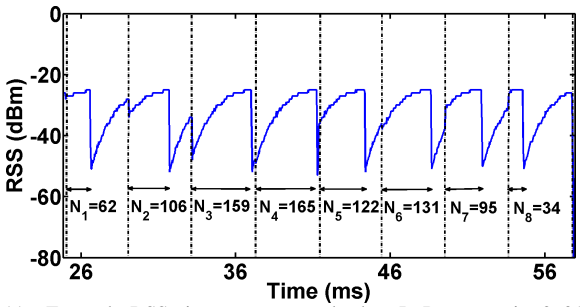


Fig. 11. Example RSS signature captured when LoRa transmits 0x01 with eight LoRa chirps. The time duration between the start of LoRa data chirps and their corresponding RSS drop are marked.

payload varying from 0x00 to 0xFF. 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 0x00 and 0xC0, respectively.

Observation 5*: A ZigBee device may not be able to distinguish all possible bytes (0x00-0xFF) which LoRa carries due to its insufficient RSS sampling accuracy.*

The 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.

C. 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 which 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.

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 II 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.

TABLE II
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
0x01	62	106	159	165	122	131	95	34
0x2F	66	88	152	163	128	134	95	36
0x33	16	87	161	167	124	133	72	132
0x34	21	104	151	167	124	134	72	132
0xFF	13	85	170	170	126	132	94	132

Observation 6*: The eight[4] numbers of RSS samples which capture the sudden RSS value drops can be used as the feature to identify the RSS signature.*

V. 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.

[4]LoRa may use more than eight data chirps to carry one byte in its packet payload. The number is decided by Eq.2 (see Section V-B).

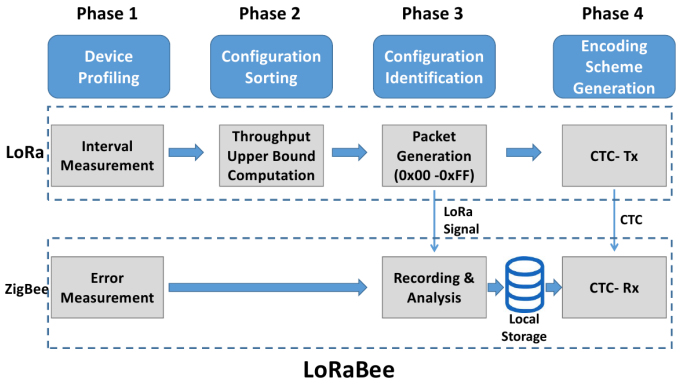
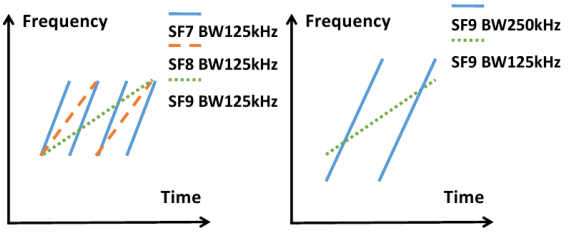


Fig. 12. LoRaBee design overview.



(a) Impact of SF. (b) Impact of BW.

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

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 V-A). In the second phase, LoRaBee computes and sorts the upper bound of theoretical throughput from LoRa to ZigBee under different LoRa configurations (Section V-B). In the third phase, LoRaBee identifies the LoRa configuration which provides the maximum actual throughput (Section V-C). In the final phase, LoRaBee generates the encoding scheme for the given devices (Section V-D).

A. 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 (see Section V-C). 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 V-B.

B. 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 Eq. 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 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:

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

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:

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

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 V-A), the upper bound of theoretical throughput from LoRa to ZigBee, which LoRaBee provides, is:

$$D_{bound} = \frac{8}{T_s + T_g} \quad (4)$$

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

With Eq. 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 (0x00-0xFF) from its measured RSS features. According to our Observation 5 in Section IV, 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 which provides the maximum D_{actual} (see Section V-C).

C. Configuration Identification

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

$$D_{actual}[i] = \alpha_i * D_{bound}[i] \quad (5)$$

Algorithm 1: Configuration Identification Algorithm

Input : $\{D_{bound}[i]|1 \leq i \leq 144\}$
Output: D_{max}, $index$, $Feature_{select}[][]$

1 $D_{max} = 0$, $index = 0$, $Feature_{select}[][]$={0};
2 **for** $i = 1; i \leq 144; i++$ **do**
3 Run Algorithm 2 to get α_i and $Feature_i[][]$;
4 $D_{actual}[i] = \alpha_i * D_{bound}[i]$
5 **if** $D_{actual}[i] > D_{max}$ **then**
6 $index$ = i;
7 $D_{max} = D_{actual}[i]$;
8 Copy $Feature_i[][]$ to $Feature_{select}[][]$;
9 **end**
10 **if** $D_{max} \geq D_{bound}[i+1]$ **then**
11 Output $index$, D_{max}, and $Feature_{select}[][]$;
12 break;
13 **end**
14 **end**

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

Input : $var(N)$
Output: α_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 α_i and $Feature_i[][]$;

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 $\{D_{bound}[i]|1 \leq i \leq 144\}$, obtained from Configuration Sorting (Section V-B). 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 $D_{actual}[i]$ (line 3–4) under each configuration i until D_{max} is not less than $D_{bound}[i+1]$ (line 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 which computes α_i under each configuration i. The input of Algorithm 2 is the maximum variations of RSS signature features $var(N)$ (see Section V-A). LoRaBee first coordinates the LoRa and ZigBee devices to run control experiments to collect all RSS signature features $\{F[m][n]|1 \leq m \leq 256, 1 \leq n \leq N_{chirp}\}$, storing N_{chirp} records for each possible data byte. Specifically, the LoRa device transmits packets each of which contains a byte from 0x00 to 0xFF. 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 (line 3–21). In Algorithm 2, the outside loop goes through all the elements in $F[][]$ (line 3–21). The inside loop checks whether the current feature is indistinguishable from the features which have already been selected (line 4–14). If not, the feature is added into $Feature_i[][]$ (line 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 $Feature_i[][1]$, $Feature_i[][2]$, ... , $Feature_i[][N_{chirp}]$. The actual number of bits which 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:

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

where $\lfloor \log_2 size \rfloor$ represents the number of bits which 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.

D. Encoding Scheme Generation

After finding the LoRa configuration which provides the maximum throughput, LoRaBee starts to generate the encoding scheme. Since only $size$ bytes among 256 possible ones (0x00-0xFF) 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.

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 which is used by LoRaBee to decode information.

Algorithm 3: LoRaBee Decoding Algorithm

```
Input  : Input feature (Input[])
Output: Decoded Result (R)
flag = true;
for j = 1; j ≤ m; j++ do
    for l = 1; l ≤ 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
```

VI. 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. Finally, We study the impact of retransmissions on LoRaBee.

A. Microbenchmark Experiments

In the Device Profiling phase, LoRaBee coordinates the LoRa and ZigBee devices to perform control experiments to measure the variations of the features extracted from the RSS signatures. Figure 14 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. LoRaBee also measures the minimum inter-packet time interval (T_g) between two consecutive LoRa packets. T_g of our LoRa device is 8.33ms. With those two parameters, LoRaBee

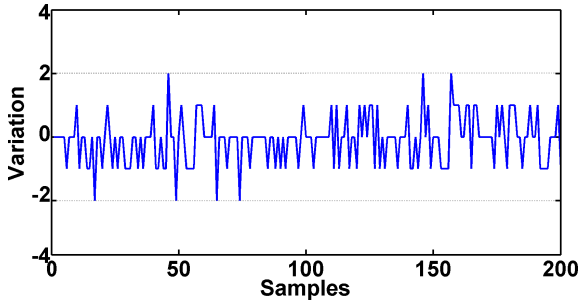


Fig. 14. Variations of measured RSS signature features from the median value. $var(N) = 2$.

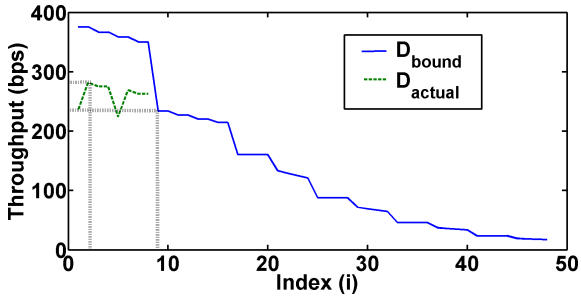


Fig. 15. Theoretical upper bound throughput D_{bound} vs. actual throughput D_{actual}.

can compute the theoretical upper bound throughput $D_{bound}[i]$ under each LoRa configuration i in the Configuration Sorting phase. Table III lists the computed D_{bound} values under each LoRa SF, CR, and CRC combination[5].

After obtaining the D_{bound} values, LoRaBee runs control experiments to measure the actual throughput (D_{actual}) in the Configuration Identification phase. To reduce the experimental 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 larger than the maximum D_{actual} under the first i configurations. Figure 15 plots the theoretical throughput upper bound D_{bound} and the actual throughput D_{actual} under different configurations. LoRaBee finds the maximum throughput of 281.61bps 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 among WiFi, Bluetooth, and ZigBee, since LoRa provides much lower physical bit rates ranging from 250bps to 11kbps under various configurations Table IV lists the number of distinguishable RSS signatures and D_{actual} under the first eight LoRa configurations. As Table IV 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 the second configuration. Table V lists five pairs of indistinguishable RSS signature features whose differences are smaller than the error range $var(N) = 2$.

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

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

TABLE IV
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

TABLE V
SIMILAR RSS SIGNATURE FEATURES COLLECTED WHEN $SF = 7$, $BW = 250$ KHZ, $CR = 4/5$, AND $CRC = on$.

Payload	RSS Signature Features N_i
0x00	13 8 16 14 12 12 16 17 17 17 01 12 17
0x06	13 8 16 14 12 12 16 17 17 16 01 11 17
0x1B	12 7 15 14 11 11 15 18 17 16 01 05 03
0x1D	12 7 15 14 12 11 15 18 17 17 01 05 03
0x30	12 7 15 13 11 11 15 19 18 17 01 11 07
0x33	13 7 16 13 12 11 15 18 18 17 01 11 07
0xAA	12 6 15 13 11 10 14 16 16 15 01 16 16
0xAF	13 6 15 12 11 10 14 17 16 15 01 16 16
0xE0	12 7 15 13 11 11 15 16 17 16 10 11 17
0xF3	13 7 15 13 12 11 15 16 16 16 11 11 17

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.

B. 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 16 shows the boxplot of 200 measurements. On average, the LoRa device consumes 0.33ms to encode a packet, while the ZigBee device uses 4.66ms 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 which 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.

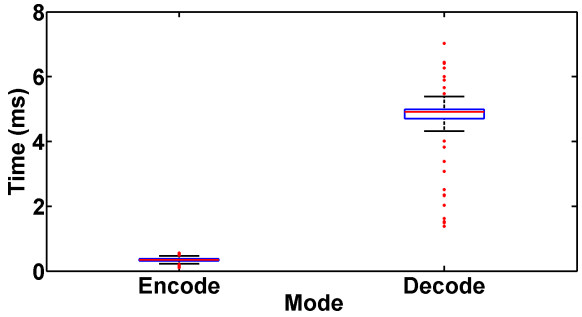
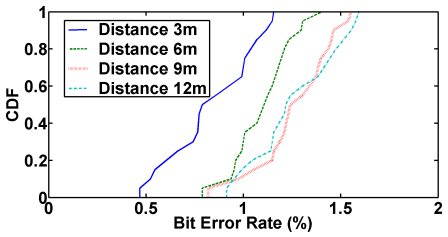


Fig. 16. Box plot of the time consumed by LoRaBee to encode and decode information. The results are gathered from 200 experimental runs. Central red mark in box indicates median; bottom and top of box represent the 25th percentile (q_1) and 75th percentile (q_2); crosses indicate outliers ($x > q_2 + 1.5*(q_2-q_1)$ or $x < q_1-1.5*(q_2-q_1)$); whiskers indicate range excluding outliers.

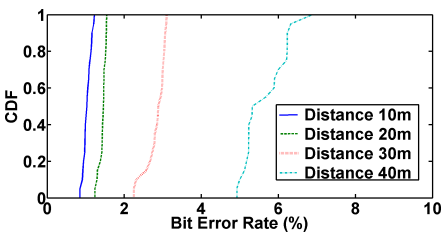
C. Bit Error Rate

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 [32] and run LoRaBee to deliver them. We vary the distance between our LoRa and ZigBee devices ranging from 3m to 12m in an indoor corridor and run the experiments for 20 times under each distance. Figure 17(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 3m apart. The maximum BER slightly increases to 1.41%, 1.55%, and 1.59% when the link distance increases to 6m, 9m, and 12m, 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.

We repeat the experiments in an outdoor environment. Figure 17(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



(a) Indoor environment.



(b) Outdoor environment.

Fig. 17. BER measurements in indoor and outdoor environments.

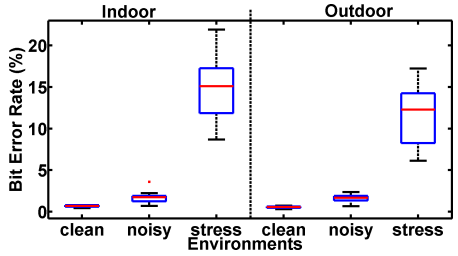


Fig. 18. Box plot of the BER of LoRaBee in the clean, noisy, and stress testing environments.

distances are 10m, 20m, 30m, 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 ($BER \leq 1.61\%$).

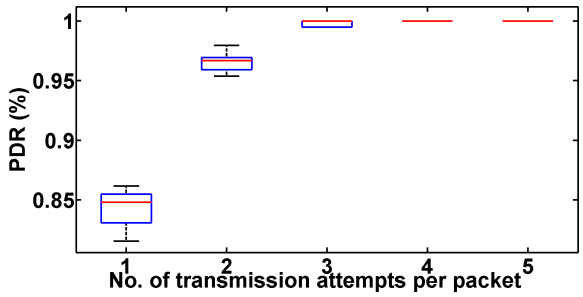
We also repeat the experiments using devices with different battery levels and in different days with different temperature and humidity and observe little impact from those factors. LoRaBee always provides stable performance.

D. Impact of Interference

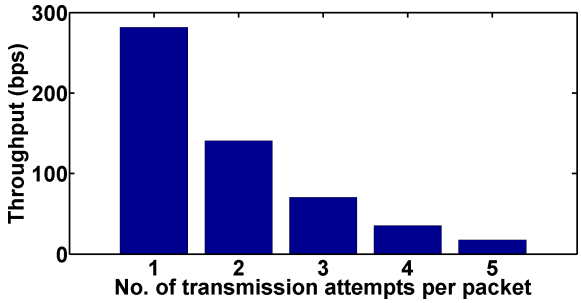
We also study the impact of interference on the BER of LoRaBee. We set up two pairs of LoRa and ZigBee devices: one pair in an indoor corridor and the other in an outdoor open space. We configure a TI CC1310 launchpad to generate controlled interference by transmitting back-to-back 64-Byte ZigBee packets in the central frequency of 915.6Mhz and vary the distance between the interferer and the LoRa and ZigBee device pair to create different interference conditions: clean, noisy, and stress test. 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 18 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 medium access control (MAC) protocol (e.g., a TDMA-based MAC) when using LoRaBee.

E. Impact of Retransmissions

Finally, we evaluate the impact of retransmissions on LoRaBee. Figure 19(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 19(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



(a) Reliability.



(b) Throughput.

Fig. 19. Performance with different No. of transmission attempts per packet.

for each packet. As Figure 19(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.

VII. Conclusions

In this paper, 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. Experimental results show that our LoRaBee provides reliable CTC communication from LoRa to ZigBee with the throughput of up to 281.61bps in the Sub-1 GHz bands in indoor and outdoor environments.

Acknowledgment

This work was supported by the NSF through grant CRII-1657275 (NeTS).

REFERENCES

[1] M. E. Porter and J. E. Heppelmann, "How smart, connected products are transforming competition," *Harvard Business Review*, vol. 92, no. 11, pp. 64–88, 2014.

[2] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems," *Proceedings of the IEEE, Special Issue on Industrial Cyber Physical Systems*, vol. 104, no. 5, 2016.

[3] B. Li, L. Nie, C. Wu, H. Gonzalez, and C. Lu, "Incorporating Emergency Alarms in Reliable Wireless Process Control," in *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2015.

[4] C. Gu, R. Tan, X. Lou, and D. Niyato, "One-Hop Out-of-Band Control Planes for Low-Power Multi-Hop Wireless Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[5] LoRa. [Online]. Available: https://lora-alliance.org/

[6] X. Guo, X. Zheng, and Y. He, "WiZig: Cross-Technology Energy Communication over a Noisy Channel," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[7] X. Guo, Y. He, X. Zheng, L. Yu, and O. Gnawali, "ZIGFI: Harnessing Channel State Information for Cross-Technology Communication," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[8] X. Zheng, Y. He, and X. Guo, "StripComm: Interference-Resilient Cross-Technology Communication in Coexisting Environments," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[9] S. M. Kim and T. He, "Freebee: Cross-Technology Communication via Free Side-Channel," in *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2015.

[10] S. M. Kim, S. Ishida, S. Wang, and T. He, "Free Side-Channel Cross-Technology Communication in Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2974–2987, 2017.

[11] Z. Yin, W. Jiang, S. M. Kim, and T. He, "C-Morse: Cross-Technology Communication with Transparent Morse Coding," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[12] Z. Li and T. He, "WEBee: Physical-Layer Cross-Technology Communication via Emulation," in *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

[13] S. Wang, S. M. Kim, and T. He, "Symbol-Level Cross-Technology Communication via Payload Encoding," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2018.

[14] W. Jiang, S. M. Kim, Z. Li, and T. He, "Achieving Receiver-Side Cross-Technology Communication with Cross-Decoding," in *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2018.

[15] X. Zhang and K. G. Shin, "Gap Sense: Lightweight Coordination of Heterogeneous Wireless Devices," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013.

[16] Y. Zhang and Q. Li, "HoWiES: A Holistic Approach to ZigBee Assisted WiFi Energy Savings in Mobile Devices," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013.

[17] X. Zhang and K. G. Shin, "Cooperative Carrier Signaling: Harmonizing Coexisting WPAN and WLAN Devices," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, 2013.

[18] K. Chebrolu and A. Dhekne, "Esense: Communication through Energy Sensing," in *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[19] S. Yin, Q. Li, and O. Gnawali, "Interconnecting WiFi Devices with IEEE 802.15.4 Devices without Using a Gateway," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2015.

[20] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu, "B2W2: N-Way Concurrent Communication for IoT Devices," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2016.

[21] S. Wang, Z. Yin, S. M. Kim, and T. He, "Achieving Spectrum Efficient Communication under Cross-Technology Interference," in *International Conference on Computer Communication and Networks (ICCCN)*, 2017.

[22] Z. Yu, C. Jiang, Y. He, X. Zheng, and X. Guo, "Crocs: Cross-Technology Clock Synchronization for WiFi and ZigBee," in *International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2018.

[23] T. Hao, R. Zhou, G. Xing, M. W. Mutka, and J. Chen, "WizSync: Exploiting Wi-Fi Infrastructure for Clock Synchronization in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 6, pp. 1379–1392, June 2014.

[24] H. Wang and A. O. Fapojuwo, "A Survey of Enabling Technologies of Low Power and Long Range Machine-to-Machine Communications," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2621–2639, 2017.

[25] R. W. Hamming, "Error Detecting and Error Correcting Codes," *Bell System Technical Journal*, vol. 2, no. 29, 1950.

[26] TI CC1352R. [Online]. Available: http://www.ti.com/tool/launchxl-cc1352r1

[27] Silicon Labs EFR32MG12P433F1024GM48. [Online]. Available: https://www.silabs.com/products/wireless/mesh-networking/efr32mg-mighty-gecko-zigbee-thread-soc/device.efr32mg12p433f1024gm48

[28] Raspberry Pi 3 Model B. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[29] SX1272 LoRa Shield for Raspberry Pi. [Online]. Available: https://www.cooking-hacks.com/sx1272-lora-shield-for-raspberry-pi-900-mhz

[30] RN2903. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/50002390E.pdf

[31] CC1310. [Online]. Available: http://www.ti.com/lit/ds/symlink/cc1310.pdf

[32] Random.org. [Online]. Available: https://www.random.org/bytes/