Achieving Receiver-Side Cross-Technology Communication with Cross-Decoding

Wenchao Jiang
Department of Computer Science and Engineering,
University of Minnesota
jiang832@umn.edu

Song Min Kim*
Department of Computer Science,
George Mason University
song@gmu.edu

Zhijun Li
School of Computer Science and Technology,
Harbin Institute of Technology
lizhijun.hit@gmail.com

Tian He*
Department of Computer Science and Engineering,
University of Minnesota
tianhe@umn.edu

ABSTRACT

Cross-technology Communication (CTC) is a key technique to explore the full capacity of heterogeneous wireless. The latest CTC designs explore the PHY-layer to reach the standards' maximum rate, but leaving a critical gap to practicality – existing PHY-layer CTCs are commonly transmitter-side techniques requiring a high-end transmitter (with a high degree of freedom in signal manipulation) to emulate the receiver signal closely. This inherently limits the reverse direction (low-end to high-end) communication. We present XBee, a unique receiver-side CTC that fills in the gap and makes a critical step towards achieving CTC bidirectionality. XBee is demonstrated as a ZigBee to BLE communication, where the key innovation lies in the unique mechanism of cross-technology decoding, or *cross-decoding* in short, which interprets a ZigBee frame only by carefully observing the bit patterns obtained at the BLE receiver. Technically, XBee counterintuitively explores the sampling offset to overcome the intrinsic challenge due to BLE's narrower bandwidth (1MHz) than ZigBee (2MHz). Extensive implementation and evaluation on USRP and commodity devices reaches 250 kbps under 85% reliability, a 15,000x improvement over state-of-the-art ZigBee to BLE communication, and comparable with the latest PHY-layer CTCs to achieve CTC bidirectionality.

*Song Min Kim and Tian He are co-corresponding authors.

MobiCom '18, October 29-November 2, 2018, New Delhi, India

ACM ISBN 978-1-4503-5903-0/18/10…$15.00
https://doi.org/10.1145/3241539.3241547

CCS CONCEPTS

• **Networks** → *Wireless personal area networks*;

KEYWORDS

Cross-technology Communication; Bluetooth Low Energy; ZigBee; Internet of Things

ACM Reference Format:
Wenchao Jiang, Song Min Kim, Zhijun Li, and Tian He. 2018. Achieving Receiver-Side Cross-Technology Communication with Cross-Decoding. In *The 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18), October 29-November 2, 2018, New Delhi, India.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3241539.3241547

1 INTRODUCTION

Over the last decade, we have witnessed the explosive growth of wireless technologies in diversity (e.g., WiFi, ZigBee, and Bluetooth) as well as in density, to satisfy various communication and service requirements under different environments. With the upcoming Internet of Things (IoT) area, the body of wireless devices is anticipated to reach 20 billion by the year 2020 [15].

Under the highly diversified and dense wireless habitat, the connectivity between specialized heterogeneous wireless technologies offers a great opportunity for advanced services [25]. To this end, researchers recently propose cross-technology communication (CTC) technique which enables direct connection between heterogeneities only using commodity devices. In the literature, existing CTC works can be categorized as packet-level CTC and PHY-layer CTC – Initial CTC solutions are restricted by the coarse-grained packet-level information and thus their throughputs are restricted to a few tens of bps. More recently, researchers take advantage of the PHY-layer information and propose PHY-layer

transmitter-side CTC with signal emulation. More specifically, by manipulating the payload of a packet in the software [22, 26], a wireless transmitter (e.g., WiFi or Bluetooth) is able to generate a receiver (e.g., ZigBee) compliant packet. Works in this category yield vastly increased rate reaching the limits defined by the standards.

Despite the significant improvement in the data rate and thus practicality, we note that transmitter-side CTC with signal emulation is applicable mainly for when the transmitter is a high-end platform (e.g., WiFi → ZigBee in [26]). This is because powerful radios support sophisticated modulations offering higher degrees of freedom in waveform control with greater capability in signal emulation. However, due to the asymmetric nature of CTC in terms of the transmitter and receiver, the reverse communication, i.e., from a low-end transmitter to a high-end receiver, is limited. This calls for a new fundamental technique that effectively enables the PHY-layer CTC in the case of low-end transmitter incapable of signal emulation due to the radio or computational limits.

In this paper, we propose the first receiver-side CTC, which aims at moving the complexity to the receiver side, as opposed to the transmitter-side CTC with signal emulation. This is inspired by the observation that the bit stream yield by any native demodulator reflects some universal and intrinsic properties of the waveform in the air, such as the amplitude, frequency, or phase. In addition, different modulation techniques are intrinsically related. For example, the frequency-shift keying and phase shift keying are tied because the frequency is the time derivative of phase [4]. Therefore a frequency-shift keying demodulator is able to cross-decode phase-shift keying signal with an upper layer interpreter to recover the phase information from the demodulated bits, which indicates frequency shifts. Specifically, this work proposes XBee, a receiver-side CTC at Bluetooth Low Energy (BLE) for cross-decoding ZigBee packets. At the transmitter side, native ZigBee packets equivalent to homogeneous communication are sent; At the receiver side, the BLE receiver is able to cross-decode every ZigBee symbols upon the bits yielding by the native BLE demodulator. This is particularly challenging to achieve on commercial platforms, which hide physical layer signal via abstraction. Through the adoption of cross-decoding, XBee first achieves PHY-layer CTC from a low-end device to a high-end device. Its data rate is 15,000x higher than the existing packet-level ZigBee to BLE CTCs, and comparable with the state-of-the-art PHY-layer CTCs through signal emulation, enabling CTC bidirectionality. The technical contribution of XBee can be summarized as:

- We design and implement XBee, a PHY-layer CTC uniquely based on cross-decoding at the receiver side. This is achieved solely by a careful examination of the bit patterns which are observable on commercial BLE devices, which enables XBee to operate without any hardware or firmware modification. Most importantly, the transmitter side stays the same as in homogeneous communication.

- Interestingly, XBee *explores* the opportunity within the sampling offset, to overcome the intrinsic uncertainty in cross-decoding. This is counterintuitive as sampling offset is detrimental to decoding, and is compensated in normal communication. XBee also features link layer designs including scheduling protocol that ensures compatibility with the BLE standard as well as non-disruptiveness to other devices in the BLE network.

- We implement XBee for extensive evaluations on its rate, reliability under various environment and parameter setting. Our experiment results have shown that XBee can achieve $250kbps$ with 85% accuracy, increasing the data rate of the existing packet-level CTCs by 15,000x, and comparable to the state-of-the-art PHY-layer CTCs.

2 MOTIVATION

Spectrum sharing is becoming even more prevalent with the explosively growing body of wireless devices and standards, as well as expanding open spectrum – from traditional 5GHz, 2.4GHz, and 900MHz bands to 600MHz, TV spectrum, and 7GHz high-frequency which turned unlicensed recently between 2014-2016 [12–14]. Spectrum crowded with diverse wireless technologies with incompatible physical layers inevitably leads to cross-technology interference (CTI), which becomes one of the root causes of network performance degradation.

To this end, connectivity among heterogeneous technologies is critical to alleviate CTI and, at the same time, explore the potential of cross-technology collaboration. In other words, to draw the full capability of wireless-rich IoT. Recently, researchers propose the cross-technology communication (CTC) aiming at building the direct communication among heterogeneous wireless technologies. Existing CTC works can be categorized as packet-level CTC and the PHY-layer CTC. Packet-level CTC works have intrinsic limitations, while within PHY-layer CTC, all existing works are based on transmitter-side signal emulation.

• **Limitations of Packet-Level CTC.** Earlier set of CTC designs use packet level information, such as the packet duration [5, 37], beacon interval [25], data traffic pattern [10, 21, 34], and energy amplitude [18] to convey messages across technologies. Such approaches, due to the coarse packet-level granularity, have intrinsic limitations in the data rate confined to a few tens bps at the highest.

• **Limitations of Existing PHY-layer CTC.** To overcome these limits, the latest CTC designs [22, 26] utilize fine-grained physical layer information for high-speed CTC that approach the maximum rate defined by the standard. Technically, PHY-layer CTC introduced until now are based on transmitter-side signal emulation, where the transmitter approximates the target waveform by exploring the signal degree of freedom offered by the transmitter's modulator. For example in [26], a WiFi to ZigBee PHY-layer CTC is proposed. The WiFi transmitter carefully selects payload where the corresponding OFMD QAM constellation points approximate that of the ZigBee's OQPSK signal so that the emulated signal can be demodulated by the ZigBee receiver with its native demodulator.

Despite PHY-layer CTC's significant advancement in the data rate, the technique of signal emulation – which the current PHY-layer CTC designs are commonly dependent upon – applicable only for a higher-end transmitter to a lower-end receiver scenario. This is because sophisticated radios in higher-end systems offer higher degrees of freedom in modulation. In other words, they support assembling (i.e., modulating) complicated signals (e.g.., OFDM QAM in WiFi), and therefore are more capable of emulating simpler signals desired at lower-end receivers (e.g., OQPSK in ZigBee). Due to this technical reason and the asymmetric nature of CTC, PHY-layer CTC in reverse direction, from a lower-end transmitter to a higher-end receiver, is difficult to achieve through the known technique (i.e., signal emulation) and remains an open issue.

• **The Need for Receiver-Side Cross-decoding.** This work is motivated by the fundamental but missing piece – enabling PHY-layer CTC from lower-end transmitted to higher-end receiver – which is the key technique to achieve bidirectional communication in PHY-layer CTC. This is achieved by enabling cross-decoding of the native transmitter packet at the receiver. For example, a commercial BLE device (receiver) runs a mechanism that interprets the message within an unmodified ZigBee (transmitter) packet. In other words, cross-decoding is pushing the complexity to the receiver side (conversely to the signal emulation) which is the higher-end. Along with the previous designs, this is the key to accomplishing PHY-layer CTC in all directions; thus bringing true ubiquitous connectivity among heterogeneous wireless systems and further, enabling cross-technology channel negotiations and advanced collaborations in practice.

To make our description specific, in the paper, we focus on the cross-decoding of ZigBee packets at a BLE receiver, which is a missing piece left by existing transmitter-side signal emulation [22]. The CTC between ZigBee and Bluetooth, the two most popular technologies in IoT will trigger a lot of interesting applications as illustrated in Fig. 1, such as

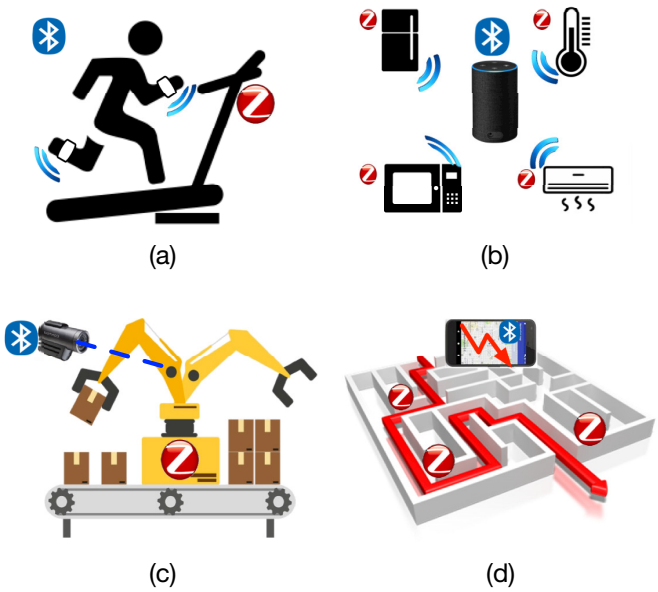


Figure 1: IoT applications with the cooperation of heterogeneous wireless technologies.

(a) In the gym, workout equipment attached with ZigBee radio can communicate with the wearable Bluetooth devices to make customized workout plan; (b) In the smart home, smart devices with ZigBee radio are able to associate to the Bluetooth speaker to play essential messages; (c) In the factory, ZigBee sensors can notify a Bluetooth camera to monitor the pipeline when abnormal events are detected; (d) In the indoor navigation, the ZigBee landmarks can help smartphones achieve fine-grained navigation.

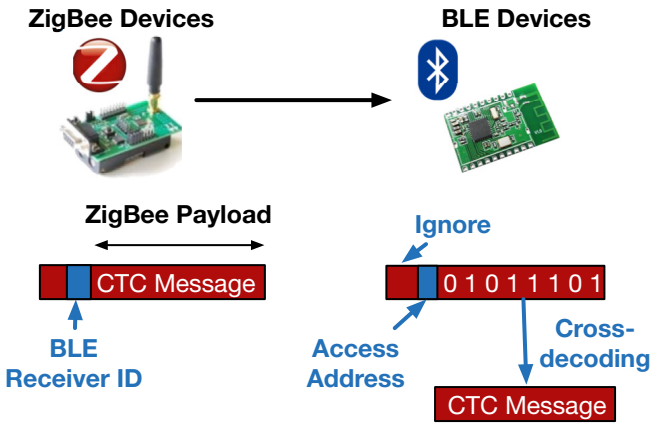


Figure 2: XBee overview: ZigBee payload embeds receiver ID that is captured as the access address at the BLE receiver. The following part of the ZigBee payload is cross-decoded by the BLE for CTC message.

3 XBEE IN A NUTSHELL

• **Overview**. XBee is a PHY-layer CTC supporting CTC message from ZigBee to BLE with receiver-side cross-decoding. By pushing the complexity to the receiver side, i.e., the BLE side, XBee supports transmitting CTC messages in native ZigBee symbols. The BLE receiver simply demodulates all the input ZigBee signal into BLE bits. Then a cross-decoding module will interpret the demodulated bits into original ZigBee symbols to recover the CTC message. To trigger the cross-decoding as well as specify the receiver, a specific ZigBee symbol sequence is chosen to work as a BLE receiver ID, as illustrated in Fig. 2, which will match a manipulated BLE access address at the receiver side.

• **Unique features** XBee is the first PHY-layer CTC based on receiver-side cross-decoding. By pushing the complexity to the receiver, it covers the essential but missing piece left by the state-of-the-art PHY-layer CTC based on transmitter-side signal emulation, paving the way to accomplish PHY-layer CTC in all directions. In addition, XBee is friendly to the transmitter, for the messages are in native ZigBee symbols. Finally, XBee needs no hardware or firmware modification at either the transmitter or receiver, making it easy to deploy onto millions of existing ZigBee and BLE devices.

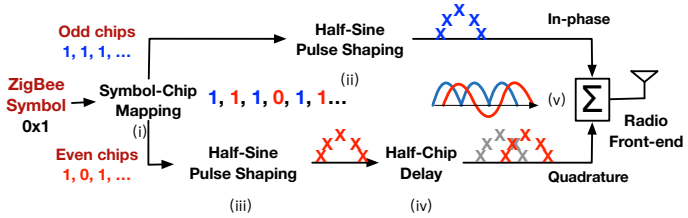


Figure 3: ZigBee as the transmitter.

4 XBEE DESIGN

This section presents the details and insights of XBee.

4.1 Background

To provide the necessary technical backgound, we first discuss the designs of ZigBee transmitter and the BLE receiver and how they operate.

Symbol (4 bits)	Chip Sequence (32 bits)
0x0	11011001110000110101001000101110
0x1	11101101100111000011010100100010
...	...
0xF	11001001011000000111011110111000

Table 1: Symbol-to-chip mapping as defined in the ZigBee (802.15.4) standard

• **ZigBee Transmitter** ZigBee adopts direct sequence spread spectrum (DSSS) and offset quadrature phase-shift keying (OQPSK) in its modulation. In Fig. 3, we illustrate the whole procedure from a ZigBee symbol to the transmitted I/Q signal in the air from step (i) to (v). A ZigBee symbol is the minimum unit of a ZigBee frame. Each ZigBee symbol contains 4-bit information, i.e., from symbol '0x0' to symbol '0xF'. In the PHY layer, a ZigBee symbol first goes through DSSS, where each ZigBee symbol will be extended to 32 chips according to the symbol-chip mapping table, i.e., Table 1, in the IEEE 802.15.4 standard, as illustrated in step (i). Then the 32 chips will go through the OQPSK modulation, where the odd chips are allocated to the in-phase and the even chips are allocated to the quadrature. Both the in-phase and quadrature chip sequences will go through a half-sine pulse shaping module, as illustrated in step (ii) and (iii), to shape the chips to a sinusoidal wave. What unique in OQPSK is that the quadrature chip sequence will further have a half-chip delay, as illustrated in step (iv). Finally, the in-phase and quadrature signals are merged and transmitted to the air.

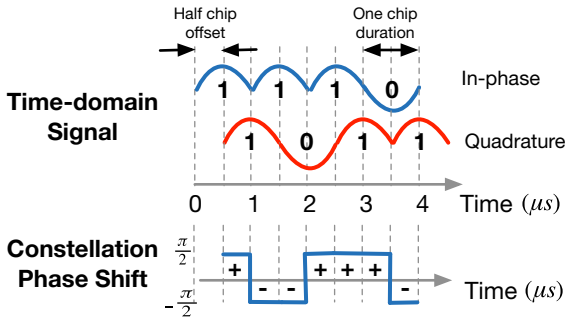


Figure 4: Time-domain signal and phase shift of ZigBee symbol '1'.

The transmitted ZigBee signal shows particular property in the constellation. In Fig. 4, the time-domain signal of the first 8 chips of ZigBee symbol '1' is plotted. Each ZigBee chip lasts $1\mu s$. In the constellation, the change in phase between consecutive samples, referred to as the *phase shift*, is calculated. Due to the half-chip offset in OQPSK, the phase shifts will only take two values, $\frac{\pi}{2}$ or $-\frac{\pi}{2}$, representing positive or negative phase shifts.

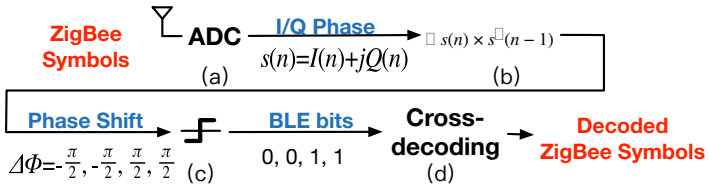


Figure 5: BLE as the receiver.

• **BLE receiver** In the PHY layer, BLE adopts the Gaussian Frequency Shift Keying (GFSK), where each BLE bit indicates either a positive or a negative frequency shift. As illustrated in Fig. 5, at the demodulator, BLE adopts the quadrature demodulator to detect frequency shifts through the phase shifts [1][1]. More specifically, a BLE receiver first samples the channel, i.e., gets the complex I/Q samples $s(n) = I(n) + jQ(n)$, in step (a) and feeds to its demodulator. The demodulator calculates the change in phase, i.e., the *phase shifts*, between consecutive I/Q samples to figure out the signal frequency shift. More specifically, the BLE demodulator uses the formula $arctan(s(n) \times s^*(n-1))$, where $s^*(n-1)$ is the conjugate of $s(n-1)$ in step (b). In step (c), the phase shifts are quantized to be BLE bit 0 or 1 according to the *sign* of phase shifts to be negative or positive. Finally, the cross-decoding block, illustrated in step (d), interprets the BLE bits yielding from the native demodulator to ZigBee symbols. Since each BLE bit lasts $1\mu s$, while each ZigBee symbol lasts $16\mu s$, 16 BLE bits are interpreted as one ZigBee symbol.

4.2 Opportunities and Challenges

Cross-decoding is feasible due to the following two technical insights. First, the phase shift is the intrinsic feature of phase modulated signal such as ZigBee signal and is also used at the BLE receiver to figure out signal frequency shift. Second, at the BLE receiver, the phase shifts are quantized so that only the *sign* (+/-) of the phase shifts matters, which brings a lot of freedom in cross-decoding.

However, the challenges come from the fact that the BLE bandwidth is 1MHz, only half that of ZigBee. The low bandwidth is corresponding to the low sample rate, or equivalently the larger sample interval according to the Nyquist theorem. As a result, BLE receiver is not able to get full ZigBee symbol information from sampling. Later on, we will see this makes one ZigBee symbol corresponds to multiple possible BLE bit sequences at the BLE receiver. How to figure out and deal with the uncertainty are challenging issues in cross-decoding.

4.3 Cross-decoding

XBee's core technique of cross-decoding interprets ZigBee packet only from the bit patterns obtained at the BLE receiver, making the design fully compatible with commercial devices. To achieve this, we first offer insights on BLE output bits when it is fed with different ZigBee signals, which can be inversely applied to derive ZigBee chips (and thus symbols) – i.e., cross-decoding. The limited bandwidth of BLE (1MHz) compared to ZigBee (2MHz) makes BLE bits only partially reflect ZigBee signal.

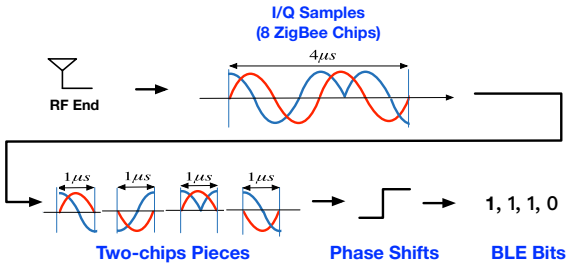


Figure 6: BLE receiver yields a bit for every $1us$ corresponding to two ZigBee chips.

We illustrate cross-decoding with a walk-through example in Fig. 6 with 8 ZigBee chips lasting $4\mu s$. At the BLE receiver, ZigBee chips are first cut into four $1\mu s$ pieces. We refer to each piece as the *two-chip piece* for its containing the phase shift information of two ZigBee chips. A two-chip piece will finally be demodulated as a single BLE bit '1' or '0' according to the accumulated phase shift.

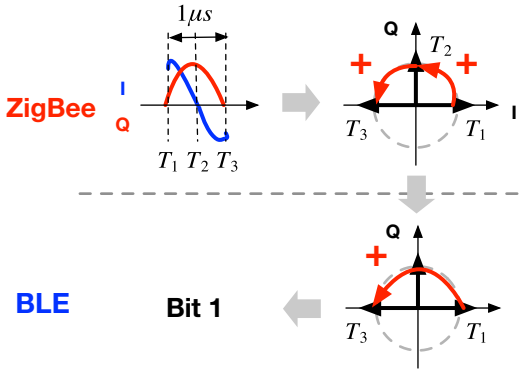


Figure 7: BLE bit when phase shifts in two chips are consistent. Phase shift of '++' yields bit '1' at the BLE receiver. Likewise, '−−' yields '0'.

We now take a closer look at the demodulation of the two-chip pieces. We first study the case when the two chips incur consecutive two positive (++) phase shifts, as shown in Fig. 7. In the figure, ZigBee has a $\frac{\pi}{2}$ phase shift every $0.5us$, from T_1 to T_2 and from T_2 to T_3. The BLE receiver, however, with its bandwidth limited to 1MHz, samples at only T_1 and T_3. The accumulated phase shift from T_1 to T_3 is positive (i.e., π), so BLE outputs bit '1'. Similarly, BLE conveys bit '0' upon two chips with both negative (−−) phase shifts. This demonstrates the clear relationship between consistent two-chip pieces and the BLE bits, which, however, does not hold when phase shifts due to the two chips are inconsistent (i.e., '+-' or '-+').

[1] Note that frequency is the derivative of phase, so a frequency shift keying $s(t) = Acos(2\pi(f \pm \Delta f)t)$ is equivalent to a phase shift keying of $s(t) = Acos(2\pi f t \pm \Phi(t))$, where $\Phi(t) = 2\pi\Delta f t$.

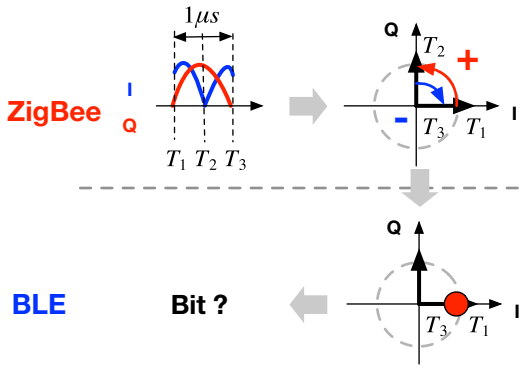


Figure 8: Inconsistent phase shifts leave BLE bit undetermined.

• **The impact of inconsistent phase shift:** We illustrate the inconsistent phase shift case, i.e., one positive and one negative, in Fig. 8. In the figure, the phase shifts from T_1 to T_2 and T_2 to T_3 are different in the constellation, which makes the overall phase shift 0 theoretically. In this case, we are not able to determine the final BLE bit, which we refer to as the *undetermined bits*. However, in a practical system, the phase shift will not stay at 0, factors such as the unsynchronized transmitter and receiver or the signal distortion in the air, will break the balance and make the final phase shift prone to either the positive or the negative side.

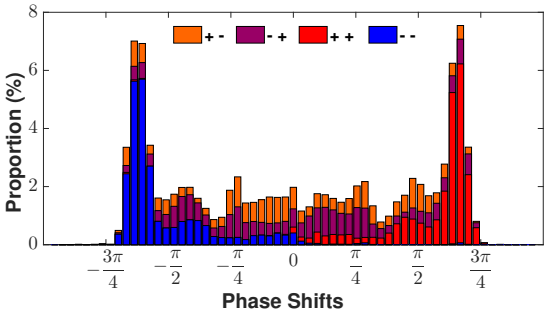


Figure 9: Phase shift distribution of all four groups of two-chips pieces.

As a proof of concept, in Fig. 9, we use USRP BLE receiver to demodulate over 1000 random ZigBee symbols, and outputs the phase shift of each two-chip piece before interpreting them into BLE bits. Then we plot the distribution of the phase shift of the four possible groups of the ZigBee two-chip pieces, according to the combinations of '+' and '-', i.e., '++', '+-', '-+', and '- -'. From the figure we can see, almost all the phase shifts of the '++' and '- -' groups are positive and negative respectively, which will be uniquely demodulated as BLE bits '1' and '0' respectively. The other two groups cannot be uniquely demodulated, because they distribute on both the positive and negative sides. In addition, the phase shifts of the group '++' and '- -' are mostly over $\pi/2$ or less than $-\pi/2$, while that of the other two groups accumulate near $\pm\pi/4$. That is because consistent phase shifts (i.e., '++' and '- -') will sum up, while inconsistent phase shifts (i.e., '+-' and '-+') will cancel out each other.

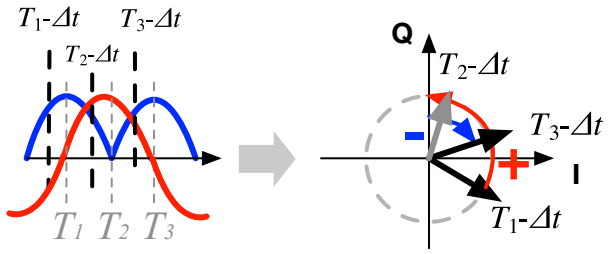
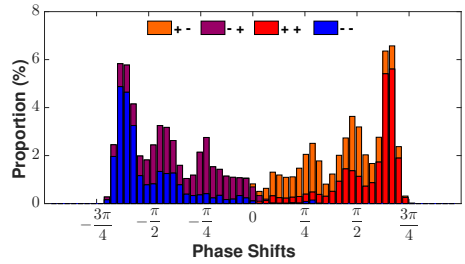


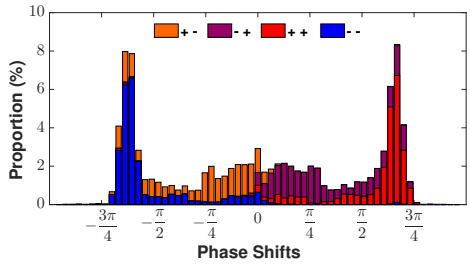
Figure 10: The impact of sample offset on the BLE bit sequence.

• **Exploring the sample offset:** The actual output of undetermined bits largely depends on the sample offset between the transmitter and receiver, when they are not well synchronized. The sample offset may become prone to either the left or the right of the two-chips piece boundary. Without loss of generality, in Fig. 10, we illustrate the case when the samples have a left offset from the boundary T_1, T_2, and T_3 by Δt. In the constellation, the unsynchronized samples extend the duration of positive phase shifts, i.e., the red arrow from $T_1 - \Delta t$ to T_2, and shorten that of the negative phase shift, i.e., the blue arrow from T_2 to $T_3 - \Delta t$. So the positive phase shift dominates and makes the two-chip piece demodulated as BLE bit '1'. How about the other undetermined two-chip pieces? We do not need to analyze them one by one. Due to the relatively stable sample interval during a short time, i.e., within a ZigBee packet, all the samples share the same sample offset and can be derived accordingly. The stable sample interval also greatly reduces the computation cost. For n undetermined bits, we do not need to calculate 2^n variations, but only 2, either all the samples have a left offset or all have a right offset.

As a proof of concept, in Fig. 11a and 11b, we illustrate the impact on phase shift distribution given the knowledge of sample offset. Here use the same data as in Fig. 9, but we first separate the data based on the sample offset of a packet to be left (Fig. 11a) or right (Fig. 11b). We can see the ZigBee two-chip pieces groups '+ -' and '- +' can now be uniquely demodulated. More specifically, the overall phase shift is positive if a two-chip piece '+ -' offset to the left and negative if it offsets to the right. Vice versa for the '- +' case. The results back up our finding that it is the sample offset that affects the actual output of undetermined two-chip pieces.



(a) Left sample offset.



(b) Right sample offset.

Figure 11: Distribution of the four groups of two-chips pieces with different sample offset

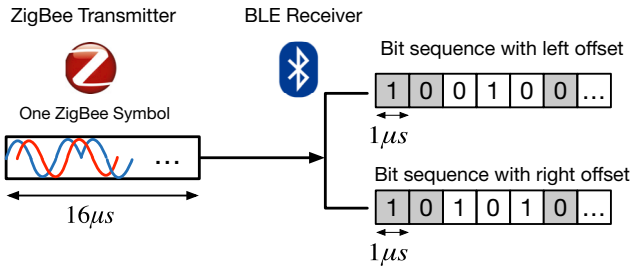


Figure 12: BLE receiver yields two different bit sequences for any ZigBee symbol due to right/left sample offset. The two sequences share determined bits (gray box), where the bits are flipped for undetermined bits (white box).

• **Cross-decoding at Commodity BLE Receiver:** Above discussion showed how a ZigBee symbol may yield two different bit sequences depending on (left or right) sample offset at the BLE receiver. In this part of the section, we illustrate how the original ZigBee symbols are recovered (i.e., cross-decoded) from the two-bit sequences. We initiate the step by step description with an example in Fig. 12; the determined bits (from consistent phase shift) are marked as grays boxes, where they remain the same regardless of the left/right offset. Conversely, undetermined bits (from inconsistent phase shift), indicated as white boxes, depends on sample offset. In other words, the two-bit sequences yield from left/right sample offset share the same bits in gray boxes, where those in white boxes are flipped (0↔1). Recall that there are 16 ZigBee symbols; since each symbol may yield two-bit sequences at the BLE receiver, cross-decoding turns out to be simply mapping each of the 32 possible bit sequences to a most likely ZigBee symbol. This is formulated as two matrices that map bit sequences to the corresponding ZigBee symbols, where one matrix holds the mapping for 16-bit sequences under left sample offset while the other holds those under right sample offset.

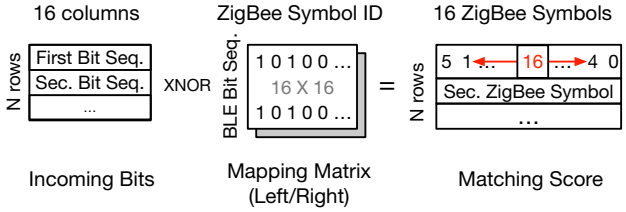


Figure 13: ZigBee symbol ↔ BLE bit seq. mapping.

Fig. 13 illustrates the process of cross-decoding via matrix mapping. The bit sequence output at the commercial BLE receiver is cut into small chunks of length 16 to match the duration of a ZigBee symbol (i.e., 16us). As the received bits are naturally prone to error in practice, each 16 bit sequence is compared with the ideal bit sequence corresponding to the ZigBee symbols under left/right sample offset – in other words, each column of the left/right mapping matrix. Specifically, this is computed by XNOR between the input bit sequence and the ideal bit sequence, which yields the number of bit match between the two. The ZigBee symbol that corresponds to the ideal bit sequence with the maximum bit match is the result of cross-decoding. From cross-decoded ZigBee symbols, XBee extracts the message including frame header, payload, and FCS. We note that mapping matrices are constant as therefore XBee computationally lightweight, enabling non-disruptive operation under low-end BLE devices. Specifically, the computational cost is only $O(n)$, where n is the number of input bits. We measure the XBee energy consumption in Sec. 7.5.

To summarize, XBee cross-decodes ZigBee symbols directly from the bits retrieved from a commercial BLE device. By smartly leveraging sampling offset, an unavoidable phenomenon in practice, XBee effectively recovers original ZigBee symbols under very light-weight computation. That is, surprisingly, XBee successfully receive 2MHz-wide ZigBee signal using BLE hardware constrained to 1MHz bandwidth.

5 HOW XBEE WORKS

In this section, we will discuss how to apply the cross-decoding technique at the BLE receiver along with all the BLE protocol constraints such as packet detection, data whitening, and scheduling.

5.1 XBee Packet Detection

According to the BLE frame format, a 32-bit access address is attached ahead of any BLE frame, which is used to identify the BLE frame in a specific BLE connection. Its value can be determined by the user and shared between the transmitter and receiver in the association process.

In XBee, for each ZigBee-to-BLE connection, we assign a unique receiver ID at the ZigBee transmitter before any CTC message, as illustrated in Fig. 2. This receiver ID is in native ZigBee symbol, and lasts 2-symbol long, corresponding to 32 BLE bits. At the BLE receiver side, we use the 32 BLE bit sequence corresponding to the 'receiver ID' as the access address. Then the ZigBee packet can be recognized by the BLE receiver as a valid BLE packet and go further into the demodulation and cross-decoding.

In addition, as we mentioned earlier, a receiver ID in ZigBee symbols will correspond to two BLE bit sequences depending on the sample offset. In the access address detection, we allow bit sequences matching either access addresses due to different sample offsets to pass. By looking at which access address matches the receiver ID, we are able to figure out whether there exists a left offset or right offset between the transmitter and receiver and choose the corresponding mapping matrix for cross-decoding.

5.2 Reverse Data Whitening

Till now, we have assumed that we have direct access to the BLE bit stream from the native demodulator for the simplicity of description. However, in real BLE platform, we only have access to the payload bytes. Between the raw BLE bytes and the payload bytes, for the security issue, there is a scrambler layer, known as the data whitening. The data whitening process in BLE is through a linear-feedback shift register (LFSR) shown in Fig. 14. More specifically, the LFSR circuit will output a scramble seed which is used 'whiten' the received bytes by doing the XOR operation with them. The scramble seed is initialized as the channel number (i.e., from 0 to 39) and change iteratively after each byte through the formula $x^7 + x^4 + 1$ as shown in Fig. 14. To recover the raw BLE bytes from the payload bytes, we need to generate the same sequence of BLE scramble seeds and XOR them byte by byte with the BLE payload bytes.

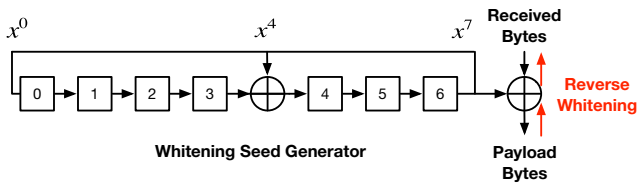


Figure 14: Reverse BLE data whitening

5.3 XBee Scheduling

Two specific difference between ZigBee and BLE network are that BLE devices need association and will do frequency hopping. To break the barrier between ZigBee and BLE, we need to design the MAC layer association and channel scheduling methods.

We start with a brief introduction of how BLE association and frequency hopping works. The 40 BLE channels, i.e., channel 0 to 39, are classified as the 3 advertising channels, i.e., 37, 38, 39, and 37 data channels. Before communication, two BLE devices must first associate at one of the three advertising channels. More specifically, one BLE device, referred to as the BLE slave device, will keep broadcasting its availability for connection through an advertising packet. Another device, referred to as the BLE master device, will choose to connect to the BLE slave device by replying 'request connection' message and necessary connection parameters, such as the channel hopping increment in frequency and the hopping interval in time. Then they will exchange packets following the associated channel hopping schedule as illustrated in Fig. 15.

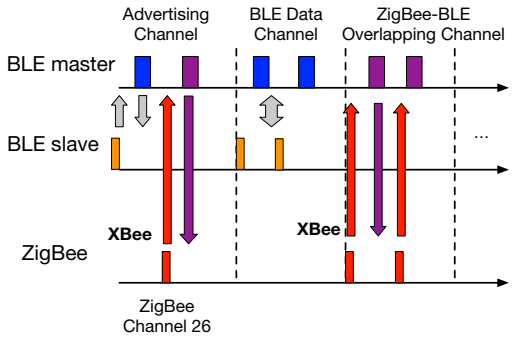


Figure 15: A BLE master device communicates with a ZigBee device while connecting to BLE slave devices.

Similarly, in XBee, the association protocol between a ZigBee device and the BLE network is designed so that the ZigBee device can connect to existing BLE network. A ZigBee device will start by broadcasting its availability for connection with a specific receiver ID at ZigBee channel 26, which is also BLE broadcasting channel 39. If the BLE master device is willing to connect to a ZigBee device, it will listen for a specific access address corresponding to the ZigBee receiver ID, besides the normal access address for BLE advertising packets. After receiving the request from the ZigBee device, the BLE master device will use signal emulation technique [22] to reply necessary connection information, such as the hopping increment and hopping interval in a ZigBee compliant frame. The ZigBee device can either choose to follow BLE's hopping schedule on the ZigBee-BLE overlapping channels if it supports frequency hopping, e.g., the WirelessHart ZigBee

protocol, or stay on one overlapping channel and wait for the arrival of the BLE device at most after 37 hops [20].

The join of a ZigBee device in the BLE network will not disrupt normal BLE connection. That is because the schedule of BLE communication is in a slotted manner. In other words, all associated devices to a BLE master devices will be given separate time slots, so that the BLE master device can communicate with normal BLE devices while keeping the connection with the ZigBee device through CTC. Thus we have achieved a hybrid network with both ZigBee and BLE devices working harmoniously.

6 DISCUSSION

6.1 Receiver ID Protection

Recall that the ZigBee transmitter attaches a receiver ID ahead of any CTC message, which will be recognized as the BLE access address. The successful detection of the receiver ID is critical to BLE cross-decoding, because otherwise the whole packet will be regarded as noise and discarded. To protect the receiver ID, we simply repeat it multiple times. For example, in Fig. 27, we repeat the receiver ID two times, and the BLE receiver is able to identify the ZigBee packet if it successfully detects any of the receiver IDs. The cross-decoding will start after the last repeat of receiver ID.

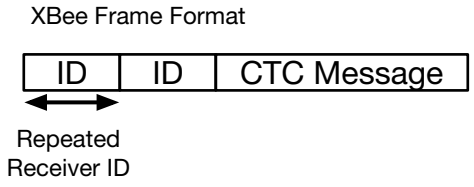


Figure 16: Reliable XBee with repeated receiver ID

6.2 Receiver Oversampling

In XBee, one challenge is the low bandwidth and the corresponding low sample rate at the BLE receiver side. However, we note that in commodity devices the receiver may oversample the channel, e.g., sample at $2MHz$ instead of $1MHz$, to make the system more robust. Even in that case, our arguments still satisfy, because the BLE demodulator will only yield bit stream at $1MHz$ no matter how fast it samples at the PHY layer, which can not fully represent the phase shift information in the ZigBee two-chips pieces.

7 EVALUATION

In this section, we compare the performance of XBee with the state of the art and evaluate its performance under various settings.

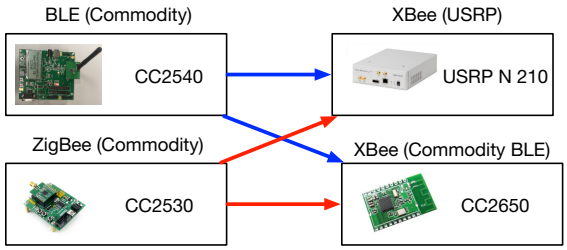


Figure 17: Experiment setting for XBee

7.1 Platform Setting

Fig. 17 demonstrates platforms for XBee evaluation. We have implemented XBee on USRP N210 with BLE PHY, as well as on commercial off-the-shelf (COTS) BLE CC2650 evaluation board. We also use COTS ZigBee and BLE with CC2530 and CC2540 boards, respectively, as transmitters. All experiments are repeated multiple times for statistical results.

7.2 Data Rate

We first evaluate the data rate of XBee in comparison to the standard ZigBee and state-of-the-art CTC techniques. The study spans both packet-leve CTC designs [21, 25, 34] and more recent PHY-layer CTCs [22, 26].

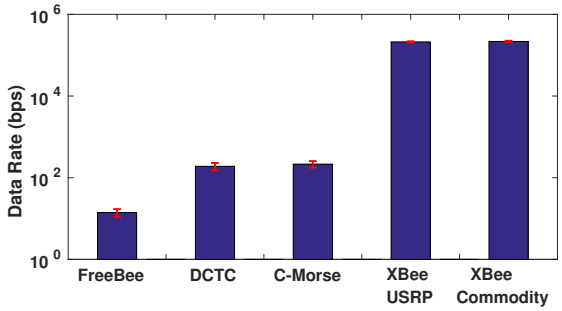


Figure 18: XBee data rate compared with three state-of-the-art packet level CTCs

• **Versus packet-level CTC:** We first compare XBee's data rate with three packet level CTCs, i.e., FreeBee[25], DCTC [21], and C-Morse [34]. As shown in Fig. 18, the data rate of FreeBee with $14bps$ where DCTC and C-Morse have rates of $190bps$ and $215bps$. Such designs use coarse-grained packet-level information (e.g., packet timing, patterns, etc.) and thus are intrinsically limited in performance. XBee, by directly utilizing the physical layer information, reaches significantly higher rate. XBee on USRP and commodity chip can achieve a data rate of $212kbps$ and $217kbps$ respectively, which outperforms FreeBee by over 15,000×, and DCTC and C-Morse by over three orders of magnitude. This demonstrates the practicality of XBee over packet-level competitors.

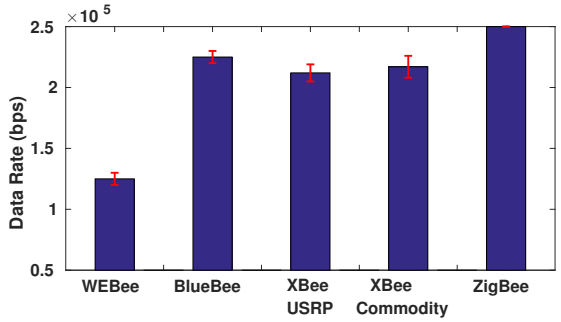


Figure 19: XBee data rate VS. state-of-the-art PHY-layer CTCs

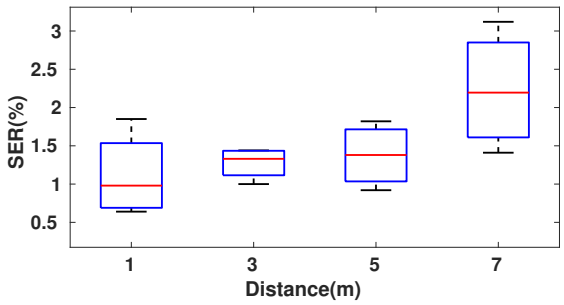


Figure 20: SER with different distance.

• Versus state-of-the-art PHY-layer CTC: The recent advances in CTC introduced PHY-layer designs based on signal emulation, namely WEBee and BlueBee [22, 26]. Work along this line commonly leverage the transmitter's high degree of freedom in signal manipulation, to generate waveform closely follows that of the receiver technology. XBee is a new PHY-layer CTC taking the unique approach of cross-decoding, which, by transferring the complexity to the receiver side, enables CTC under transmitter with a limited degree of freedom (i.e., low-end RF). We compare the performance of XBee to the state-of-the-art PHY-layer CTC designs by measuring how closely they approach the ZigBee standard data rate of 250Kbps. Fig. 19 shows that WEBee and BlueBee achieve $125kbps$ and $225kbps$, respectively. XBee, by reaching $217kbps$, outperforms WEBee by 1.7× and is comparable to BlueBee. In contrast to XBee, BlueBee's receiver's bandwidth (2MHz ZigBee) is wider than that of the sender (1MHz BLE). BlueBee benefits from this to retrieve more fine-grained phase information, thus reaching slightly higher performance. The result validates that XBee, by only utilizing receiver-side technique (i.e., cross-decoding), achieves performance similar to the state-of-the-art PHY-layer CTCs that rely on sophisticated transmitter-side signal processing. This is an indication that XBee successfully fills in the gap towards CTC bidirectionality accompanied with known PHY-layer CTCs, as experimentally evaluated in Sec. 7.6.

7.3 Symbol Error Rate

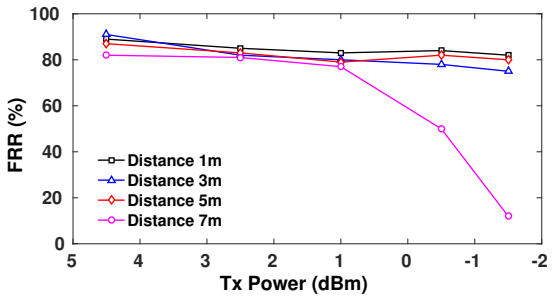
Here we evaluate the ZigBee symbol error rate (SER) via cross-decoding.

In Fig. 20, we study the ZigBee SER when a $4.3dBm$ commodity ZigBee Tx is put $[1m, 7m]$ away from a USRP BLE receiver. We find the average SER is about 1% at $1m$ and gradually increases to about 2.2% at 7m. That's because BLE signal is usually low in Tx power, and will quickly attenuate in the air.

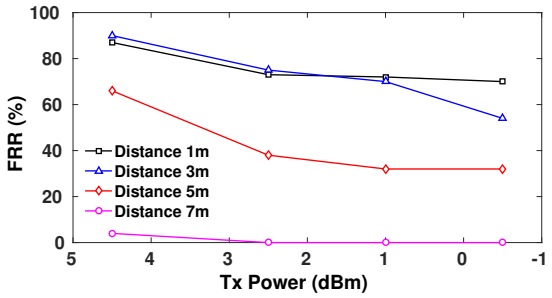
7.4 Frame Reception Ratio

In this part of the section, we study the impacts of the factors that affect the frame reception ratio (FRR) of XBee.

• Impact of Distance & Tx Power Like any other communication systems, XBee's performance is affected by distance and Tx power. The show their impact, FRR at the distance range of $[1m, 7m]$, and Tx power of $[4.5dBm, -1.5dBm]$. Both XBee implementation on USRP and commodity BLE devices are tested for completeness.



(a) USRP.



(b) Commodity device.

Figure 21: FRR with different Tx power and distance.

Fig. 21a reveals that the FRR of XBee on USRP gradually decreases from 90% to 80% with an increase in distance and the decrease in Tx power when the distance is within $5m$. A sharp decrease occurs at the distance of $7m$, at a low Tx power of $1dBm$ or lower. In Fig. 21b, XBee on the commodity device

reaches FRR over 60% when the distance is shorter than $3m$. The performance on commodity devices is worse than that of the USRP. While the details of commodity hardware are hidden, we believe that the performance gap is mainly due to cheap RF components in commodity BLE with low antenna gains and inaccurate phase detection, which is subject to change for different hardware and vendors. With the aim to provide stable and reproducible results, and to offer thorough analysis and deep understanding, we use USRP in the following parts of the section (unless otherwise stated).

Tx Power (dBm)	-22	-16	-10	-3	1	4.5
Rx RSS (dBm)	-75	-71	-64	-57	-53	-50

Table 2: Rx RSS changes linearly with the Tx power.

To understand the relationship between the FRR and the Rx RSS, we studied the relationship between the Tx power and Rx RSS. In the experiment, we use the CC2530 ZigBee device with the CC2540 BLE device. We connect the Tx and Rx with a cable with a $30dBm$ attenuator. In Table. 2, we show the relationship between the Tx power and the Rx RSS. We find that throughout the $30dB$ Tx power range, the Tx power almost changes linearly with the Rx RSS. Combined with the results in Fig. 21, we can tell the relationship between the FRR and the Rx RSS.

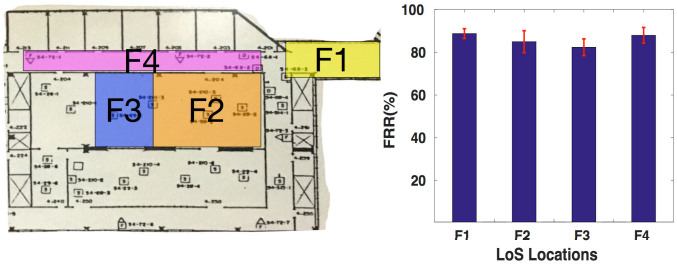


Figure 22: LoS scenarios.

• **LoS Locations** We evaluate the FRR in various locations within a university building, including the lobby (F1), a meeting room with minimal obstacles (F2), a lab (F3), and the hallway (F4). The distance between the ZigBee transmitter and the XBee receiver was kept at $1m$ while maintaining the line of sight (LoS). As illustrated in Fig. 22, the average FRR in the lobby and the hallway are both over 85%, while falling as low as 80% in the lab environment. This is because the lab is crowded with many WiFi (e.g., laptops) and Bluetooth (e.g., wireless mouse) devices causing strong interference.

• **NLoS Locations** We also study the performance of XBee under various realistic NLoS scenarios, where the ZigBee transmitter is sitting in a drawer, in a pocket, on the desk covered by some paper, or obstructed by the human body. And a USRP BLE receiver is put $1m$ away. We note that this

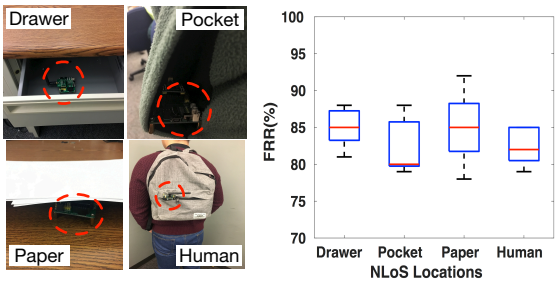


Figure 23: NLoS scenarios.

experiment was conducted in a large meeting room with mostly open space (F2 in Fig. 22) to factor out other channel effects (e.g., multipath). Fig. 23 depicts the results – While the pocket scenario shows the highest impact, FRR is kept at a reasonable level of above 80%, which demonstrates XBee's reliability under various NLoS scenarios in practice.

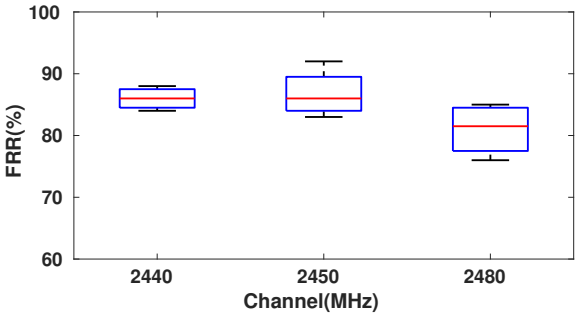


Figure 24: Interference on different channels.

• **Impact of Channel** XBee operates on all the overlapping channels between ZigBee and BLE, ranging from [$2410MHz$, $2420MHz$, ..., $2480MHz$]. Among them we choose three representative channels: one is overlapping with the WiFi channel, i.e., $2440MHz$, another is the BLE advertising channel, i.e., $2480MHz$, and the third is a relative clean channel, i.e., $2450MHz$, as illustrated in Fig. 24. We compare the impact of different sources of wireless interference, i.e., WiFi and BLE advertising packets. From Fig. 24, we find the average FRR of XBee on channels $2440MHz$ and $2450MHz$ are both over 85%, while the FRR on channel $2480MHz$ is slightly lower, around 80%. We believe that is due to the huge amount of BLE advertising packets on the channel $2480MHz$ interfering with the cross-decoding.

• **Cross-decoding Under Low Noise** In this experiment, we factor out the effect of the wireless channel to provide insights into the performance limits of cross-decoding. To do completely remove wireless noise, we connect the ZigBee transmitter and the USRP XBee receiver through a cable. We also measure low noise scenario over the air, by putting the transmitter and the receiver pair side by side. Fig. 25

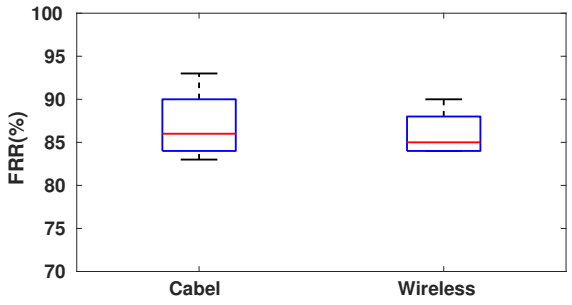


Figure 25: Low noise scenarios: cable and over the air

demonstrates that, in both cases, FRR was kept similar at 85%, which is effectively the maximum performance of XBee under the ideal channel.

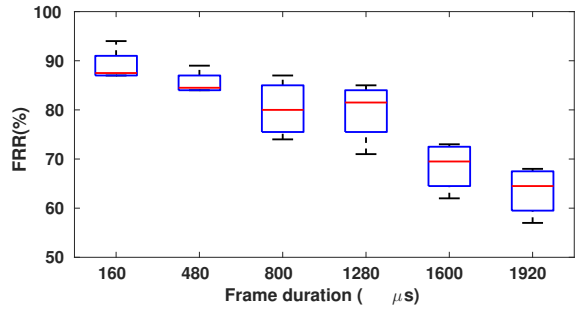


Figure 26: FRR under varying frame durations.

• **Impact of Packet Duration** Compared to BLE 4.0 standard which restricts payload size to be 33 bytes at maximum, the newly introduced BLE 4.2 allows a much longer payload of up to 251 bytes. This provides the opportunity to cross-decode long ZigBee frames. This naturally leads to a question of the impact of the frame length on XBee's performance. To investigate this, we first place ZigBee (CC2530) transmitter nearby (i.e., 1m) the USRP XBee receiver to minimize channel effects. The antenna gains are $30dB$ at the Tx and $10dB$ at the Rx. Under this setting, we increase the frame duration from $160\mu s$ to $1920\mu s$, corresponding to 20 and 240 BLE payload bytes, which closely approaches the BLE 4.2 limit. As shown in Fig. 26, the FRR of XBee decreases with the frame duration, from 90% to 66% due to a higher chance of corruption, as in any wireless communication designs. It shows XBee's FRR is kept a reasonable level of 65% under a long frame size of $1920\mu s$ defined by the latest BLE 4.2 standard – indicating XBee is able to support long and bursty data delivery.

• **Repeated Receiver ID** Recall that receiver ID corresponds to the preamble in BLE. Therefore, successful detection of the receiver ID is critical for XBee to successfully receive frames (i.e., cross-decode). To this end, we evaluate how the repetition of the receiver ID affects FRR. Fig. 27 demonstrates

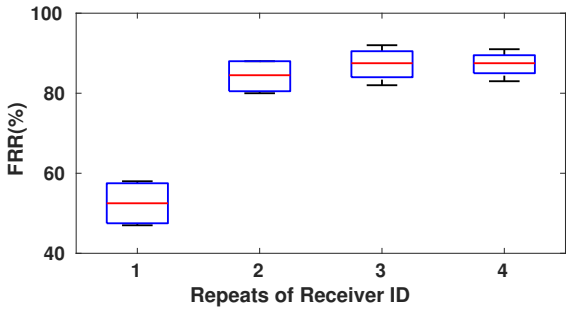


Figure 27: ZigBee frame with repeats of receiver ID

the increase in FRR with more repetition of receiver ID. A single receiver ID (i.e., repeat = 1) yields FRR=50%, however, appending two receiver IDs (i.e., repeat = 2) quickly increases the FRR to 85%. Increasing the repetition over two does not offer notable gain while increasing the overhead. Therefore XBee uses two receiver IDs by default.

7.5 Energy Consumption

Here we study the energy consumption of XBee on the commodity CC2650 BLE board. Recall that the additional energy consumption brought by XBee is the cross-decoding through matrix multiplication in Fig. 13. To measure the energy cost, we measure the average computation time the BLE board takes to cross-decode each ZigBee symbol, then multiplied by the average power of the CC2650 BLE board. Our measurement finds that it takes $174\mu s$ on average to cross-decode a ZigBee symbol. The average power consumption of the board is $4.35mW$ when active [24]. So it takes $0.7\mu J$ to cross-decode a ZigBee symbol.

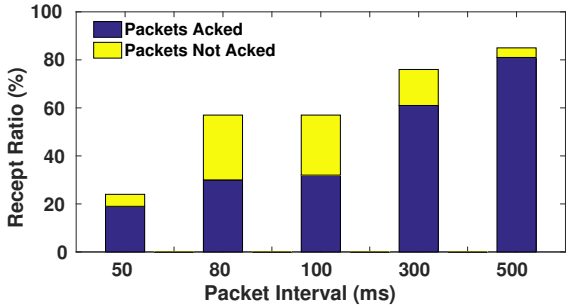


Figure 28: Bidirectional CTC between ZigBee and BLE

7.6 Application: Bidirectional CTC

XBee is a receiver-side design, which can be applied alongside existing transmitter-side PHY-CTC to achieve bidirectionality – an essential function for multiple critical aspects of networking. We demonstrate the feasibility of bi-directional CTC, with a case study of ZigBee and BLE. This is achieved

by utilizing [22], a state-of-the-art BLE to ZigBee CTC technique, with XBee. Specifically, we implement an acknowledge mechanism via the following two steps. First, a commodity ZigBee device sends native ZigBee frames to a BLE receiver. Upon correct cross-decoding at the BLE receiver via XBee, the BLE device sends back cross-technology ACKs through signal emulation in [22]. The packet interval at the ZigBee transmitter is set between [50ms, 500ms]. The result depicted in Fig. 28 shows that the frame reception and ACK rates are both low when under short transmission interval. They improve as the frame interval grows larger. This is because the USRP-based receiver has a delay in receiving, processing, and sending back ACK, ranging between [50ms, 100ms]. The long delay causes ACKs to collide with the coming ZigBee to BLE packets. When the frame interval is large enough, e.g., 500ms, XBee correctly cross-decodes 85% of the frames, and acknowledges 95% of the decoded frames, thereby successfully demonstrating bidirectional CTC in practice. We believe this will be a key enabler to sophisticated cross-technology protocol designs at the upper layers.

8 RELATED WORK

The coexistence of large amount of heterogeneous wireless technologies on the open 2.4GHz ISM band has become a common network scenario in recent decades. The coexistence does increase the competition and cross-technology interference (CTI) among them [2, 3, 6–8, 19, 23, 28–31], but at the same time provides opportunities for cross-technology communication (CTC) [5, 9–11, 16–18, 21, 27, 32–35, 37**?** , 38].

• **Packet-level CTC** Initial CTC designs focuse on exploring packet-level information for CTC, such as the packet duration [5, 37], interval [25], energy pattern[10, 21, 34], and energy level [11, 18, 38]. For example, Esense [5], and HoWiES [37] embed the WiFi to ZigBee CTC message within multiple dedicated WiFi packets via specific packet durations. The packet duration can be distinguished from background noises at the receiver side. GSense[36] attaches customized preambles ahead of heterogeneous wireless packets and exchange CTC information through the gaps between the customized preambles. FreeBee [25] is a free channel design which takes advantage of the mandatory WiFi beacons and embed CTC within the WiFi beacon intervals. B^2W^2 [11] is a CTC design from Bluetooth to WiFi by modulating the energy level of Bluetooth packets and demodulated through WiFi CSI at the receiver side. C-Morse [34] constructs a series of Morse code like long and short WiFi packets that can be demodulated at the ZigBee receiver. StripComm [38] introduces an interference-resilient CTC which adopts an interference-resilient coding scheme that contains both presence and absence of packets in one symbol to improve CTC data rate against strong interference. Despite the various packet-level information these CTC designs based on, CTC works in this category are intrinsically restricted in data rate due to the sparse packet-level information in the air.

• **Physical-layer CTC** PHY-layer CTCs are the recent advances. Existing PHY-layer CTC works commonly take advantage of the high-end transmitters to emulate the signal compliant to the low-end receiver[9, 22, 26, 27], e.g., $WiFi \rightarrow ZigBee$ or $BLE \rightarrow ZigBee$. WEBee [26] is the first to propose PHY-layer CTC through the signal emulation techniques where a commodity WiFi transmitter emulates a ZigBee standard compliant packet by carefully choosing the WiFi payload. It first achieves CTC in the PHY-layer with close to ZigBee standard data rate. BlueBee [22] is the first to implement signal emulation in low power devices, e.g., from a commodity BLE transmitter to a ZigBee receiver, by exploring the opportunities in the signal phase shifts. TwinBee [9] recovers the intrinsic errors brought in WiFi signal emulation by exploiting ZigBee chip-level error patterns and proposes a chip-combining decoding method. LongBee [27] improves the transmission range of a WiFi signal emulation by exploring the transmitter side channel bandwidth.

We find the PHY-layer CTCs till now are commonly based on the signal emulation technique. Despite their significant advancement in the data rate, the technique of signal emulation is applicable only for the higher-end transmitter to the lower-end receiver scenario. This is because powerful radios support sophisticated modulations offering higher degrees of freedom in waveform control but not vice versa. Compared with them, XBee is the first PHY-layer CTC based on the receiver side cross-decoding. It is the fundamental but missing piece to enable PHY-layer CTC from the lower-end transmitter to the higher-end receiver to accomplish PHY-layer CTC in all directions. It is the building block to enable cross-technology channel negotiations and advanced collaborations in practice.

9 CONCLUSION

In summary, XBee is the first receiver-side PHY-layer CTC with cross-decoding. By moving the complexity to the receiver side, it covers the fundamental but missing piece to enable PHY-layer CTC from lower-end transmitted to higher-end receiver to accomplish PHY-layer CTC in all directions, paving the way for advanced cross-technology channel negotiations and collaborations in practice.

ACKNOWLEDGEMENT

This work was supported in part by the NSF CNS-1525235, NSF CNS-1718456, NSF CNS-1717059, and NSF China 61672196. We sincerely thank our anonymous shepherd and anonymous reviewers for their valuable comments and feedback.

REFERENCES

[1] [n. d.]. cc2420 data sheet. http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[2] Fadel Adib, Swarun Kumar, Omid Aryan, Shyamnath Gollakota, and Dina Katabi. 2013. Interference alignment by motion. In *Proceedings of ACM MobiCom, 2013*.

[3] Paramvir Bahl, Ranveer Chandra, Thomas Moscibroda, Rohan Murty, and Matt Welsh. 2009. White space networking with wi-fi like connectivity. *ACM SIGCOMM Computer Communication Review* 39, 4 (2009), 27–38.

[4] G. Ballou. 2005. *Handbook for Sound Engineers*. Focal. https://books.google.com/books?id=y0d9VA0lkogC

[5] Kameswari Chebrolu and Ashutosh Dhekne. 2009. Esense: communication through energy sensing. In *Proceedings of ACM MobiCom 2009*.

[6] Bo Chen, Yue Qiao, Ouyang Zhang, and Kannan Srinivasan. 2015. Airexpress: Enabling seamless in-band wireless multi-hop transmission. In *Proceedings of ACM MobiCom, 2015*.

[7] Bo Chen, Vivek Yenamandra, and Kannan Srinivasan. 2015. Interference alignment using shadow channel. In *Proceedings of IEEE INFOCOM, 2015*.

[8] Lin Chen, Ruolin Fan, Kaigui Bian, Mario Gerla, Tao Wang, and Xiaoming Li. 2015. On heterogeneous neighbor discovery in wireless sensor networks. In *Proceedings of IEEE INFOCOM, 2015*.

[9] Yongrui Chen, Zhijun Li, and Tian He. 2018. TwinBee: Reliable Physical-Layer Cross-Technology Communication with Symbol-Level Coding. In *Proceedings of IEEE INFOCOM 2018*.

[10] Zicheng Chi, Zhichuan Huang, Yao Yao, Tiantian Xie, Hongyu Sun, and Ting Zhu. [n. d.]. EMF: Embedding Multiple Flows of Information in Existing Traffic for Concurrent Communication among Heterogeneous IoT Devices. In *Proceedings of IEEE INFOCOM 2017*.

[11] 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 ACM Sensys 2016*.

[12] FCC [n. d.]. *Fact Sheet: Spectrum Frontiers Order To Identify, Open Up Vast Amounts Of New High-Band Spectrum For Next Generation (5g) Wireless Broadband*. FCC. Available at https://apps.fcc.gov/edocs_public/attachmatch/DOC-340310A1.pdf.

[13] FCC. 2014. Increased Availability of Spectrum for Unlicensed Uses in the 5 GHz Band. Available at https://www.fcc.gov/document/fcc-increases-5ghz-spectrum-wi-fi-other-unlicensed-uses.

[14] FCC. 2015. Rules for unlicensed operations in the TV and the 600 MHz band. Available at https://www.fcc.gov/document/fcc-adopts-rules-unlicensed-services-tv-and-600-mhz-bands.

[15] Inc Gartner. 2016. *Gartner Report*. Available at urlhttp://cloudtimes.org/2013/12/20/gartner-theinternet-of-things-will-grow-30-times-to-26-billion-by-2020/.

[16] Piotr Gawłowicz, Anatolij Zubow, and Adam Wolisz. 2018. Enabling Cross-technology Communication between LTE Unlicensed and WiFi. In *Proceedings of IEEE INFOCOM 2018*.

[17] Xiuzhen Guo, Yuan He, Xiaolong Zheng, Liangcheng Yu, and Omprakash Gnawali. 2018. ZigFi: Harnessing Channel State Information for Cross-Technology Communication. In *Proceedings of IEEE INFOCOM 2018*.

[18] Xiuzhen Guo, Xiaolong Zheng, and Yuan He. [n. d.]. WiZig: Cross-Technology Energy Communication over a Noisy Channel. In *Proceedings of IEEE INFOCOM 2017*.

[19] Tian Hao, Ruogu Zhou, Guoliang Xing, Matt W Mutka, and Jiming Chen. 2014. Wizsync: Exploiting wi-fi infrastructure for clock synchronization in wireless sensor networks. *IEEE Transactions on mobile computing* 13, 6 (2014), 1379–1392.

[20] Robin Heydon. [n. d.]. *Bluetooth low energy: the developer's handbook*. Vol. 1.

[21] Wenchao Jiang, Zhimeng Yin, Song Min Kim, and Tian He. 2017. Transparent Cross-technology Communication over Data Traffic. In *Proceedings of IEEE INFOCOM, 2017*.

[22] Wenchao Jiang, Zhimeng Yin, Ruofeng Liu, Song Min Kim, Zhijun Li, and Tian He. 2017. BlueBee: a 10,000x Faster Cross-Technology Communication via PHY Emulation. In *Proceedings of ACM Sensys 2017*.

[23] Tao Jin, Guevara Noubir, and Bo Sheng. 2011. WiZi-Cloud: Application-transparent dual ZigBee-WiFi radios for low power internet access. In *INFOCOM*. 1593–1601.

[24] Christin Lee Joakim Lindh and Marie Hernes. 2017. *Measuring Bluetooth Low Energy Power Consumption*. http://www.ti.com/lit/an/swra478c/swra478c.pdf

[25] Song Min Kim and Tian He. 2015. FreeBee: Crosstechnology Communication via Free Sidechannel. In *Proceedings of ACM MOBICOM 2015*.

[26] Zhijun Li and Tian He. 2017. WEBee: Physical-Layer Cross-Technology Communication via Emulation. In *Proceedings of ACM MobiCom 2017*.

[27] Zhijun Li and Tian He. 2018. LongBee: Enabling Long-Range Cross-Technology Communication. In *Proceedings of IEEE INFOCOM 2018*.

[28] Rajesh Mahindra, Hari Viswanathan, Karthik Sundaresan, Mustafa Y Arslan, and Sampath Rangarajan. 2014. A practical traffic management system for integrated LTE-WiFi networks. In *Proceedings of ACM MobiCom 2014*.

[29] Georgios Nikolaidis, Mark Handley, Kyle Jamieson, and Brad Karp. 2015. COPA: cooperative power allocation for interfering wireless networks. In *Proceedings of ACM CoNEXT 2015*.

[30] Božidar Radunović, Ranveer Chandra, and Dinan Gunawardena. 2012. Weeble: Enabling low-power nodes to coexist with high-power nodes in white space networks. In *Proceedings of ACM CoNEXT 2012*.

[31] Saravana Rathinakumar, Bozidar Radunovic, and Mahesh K Marina. 2016. CPRecycle: Recycling Cyclic Prefix for Versatile Interference Mitigation in OFDM based Wireless Systems. In *Proceedings of ACM CoNEXT 2016*.

[32] Shuai Wang, Song Min Kim, and Tian He. 2018. Symbol-level Cross-technology Communication via Payload Encoding. In *Proceedings of IEEE ICDCS 2018*.

[33] Wei Wang, Tiantian Xie, Xin Liu, and Ting Zhu. 2018. ECT: Exploiting Cross-Technology Concurrent Transmission for Reducing Packet Delivery Delay in IoT Networks. In *Proceedings of IEEE INFOCOM 2018*.

[34] Zhimeng Yin, Wenchao Jiang, Song Min Kim, and Tian He. 2017. C-Morse: Cross-technology Communication with Transparent Morse Coding. In *Proceedings of IEEE INFOCOM 2017*.

[35] Zhimeng Yin, Zhijun Li, Song Min Kim, and Tian He. 2018. Explicit Channel Coordination via Cross-technology Communication. In *Proceedings of ACM MobiSys 2018*.

[36] Xinyu Zhang and Kang G Shin. 2013. Gap sense: Lightweight coordination of heterogeneous wireless devices. In *Proceedings of IEEE INFOCOM 2013*.

[37] Yifan Zhang and Qun Li. 2013. HoWiES: A holistic approach to ZigBee assisted WiFi energy savings in mobile devices. In *Proceedings of IEEE INFOCOM 2013*.

[38] Xiaolong Zheng, Yuan He, and Xiuzhen Guo. 2018. StripComm: Interference-Resilient Cross-Technology Communication in Coexisting Environments. In *Proceedings of IEEE INFOCOM 2018*.