LEGO-Fi: Transmitter-Transparent CTC With Cross-Demapping

Xiuzhen Guo, *Student Member, IEEE*, Yuan He, *Senior Member, IEEE*, Xiaolong Zheng, *Member, IEEE*, Zihao Yu, *Student Member, IEEE*, and Yunhao Liu, *Fellow, IEEE*

***Abstract*—Cross-Technology Communication (CTC) is an emerging technique that enables direct communication across different wireless technologies. The state-of-the-art works in this area propose physical-level CTC, in which the transmitters emulate signals that follow the receiver's standard. Physical-level CTC means considerable processing complexity at the transmitter, which does not apply to the communication from a low-end transmitter to a high-end receiver. This article proposes LEGO-Fi, which supports the CTC from ZigBee to WiFi and Bluetooth to WiFi. LEGO-Fi is a transmitter-transparent CTC technique, which leaves the processing complexity solely at the receiver side and therefore, makes a critical advance toward bidirectional high-throughput CTC. The key technique inside is cross-demapping, which stems from two key technical insights: 1) a ZigBee/Bluetooth packet leaves distinguishable features when passing the WiFi modules and 2) compared to ZigBee's/Bluetooth's simple encoding and modulation schemes, the rich processing capacity of WiFi offers extra flexibility to process a ZigBee/Bluetooth packet. The evaluation results show that LEGO-Fi achieves a throughput of 213.6 Kb/s in practice, which is, respectively, 13000×, 1200×, and 7.5× faster than FreeBee, ZigFi, and SymBee, the three existing ZigBee-to-WiFi CTC approaches. For the CTC from Bluetooth to WiFi, LEGO-Fi achieves a throughput of 904.1 Kb/s.**

***Index Terms*—Cross-demapping, cross-technology, ZigBee-to-WiFi.**

I. Introduction

THE PROLIFERATION of Internet-of-Things (IoT) applications brings about the increasing dense deployments of various wireless devices. The coexistence of heterogeneous wireless technologies becomes a common situation, where the interplay across technologies draws a lot of research attention [1]–[5]. Under such circumstances, cross-technology communication (CTC) emerges to enable direct communication among devices that follow different communication standards [6]–[10].

With CTC, the heterogeneous wireless devices can build a new communication channel to coordinate with each other, so that wireless interference and collisions will be appropriately handled [11]–[13]. The ZigBee sensors installed in a smart home can directly report the sensory data to a WiFi device, which may further share the data via high-speed WiFi network. The front-end sensors and controllers in a smart factory can closely collaborate with each other by directly transmitting commands and data, so as to meet the requirement of real-time applications [14], [15]. CTC not only provides a new way to manage wireless networks but also enhances the ability of interoperation and in-situ data exchange among heterogeneous devices.

To realize CTC, one will meet two major challenges: 1) the incompatibility between technologies and 2) the asymmetry of device capacity. Early works in this area propose packet-level CTC, which manipulate transmitted packets and use the packet length [16], received signal strength (RSS) [17]–[20], transmission timings [21], [22], and other features [23], [24] as the information carrier. Recent works propose physical-level CTC. WEBee [25] proposes the technique of physical-level emulation. It uses the high-speed WiFi radio to emulate the desired signals of the low-speed ZigBee radio. BlueBee [26] emulates legitimate ZigBee frames with a Bluetooth radio by using the phase differences between samples. Physical-level CTC significantly enhances the throughput, compared with packet-level CTC.

In spite of the promising progress, we still face a critical gap toward the bidirectional high-throughput CTC. The reason is that the state-of-the-art physical-level emulation technique incurs considerable processing cost at the transmitter, preventing it from being applied to the reverse direction, i.e., from a low-end device to a high-end device. Recently, Wang *et al.* [27] proposed a symbol-level CTC via payload encoding and recycled the idle listening result of WiFi to decode ZigBee signals. Other proposals [21], [22], [24] for CTC from ZigBee to WiFi all stay at the packet level.

Taking CTC from ZigBee to WiFi as an example, we list the critical challenges as follows. First, the bandwidth of ZigBee (2 MHz) is much narrower than that of WiFi (20 MHz). Given the fixed sampling period at a WiFi receiver, only a small portion of the ZigBee signals can be received, which means inevitable information loss. Second, ZigBee and WiFi have totally different packet formats. Passing a ZigBee preamble

Manuscript received September 26, 2020; revised December 1, 2020; accepted January 9, 2021. Date of publication February 2, 2021; date of current version April 7, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1003000; in part by the Research and Development Project of Key Core Technology and Generic Technology in Shanxi Province under Grant 2020XXX007; and in part by the National Science Fund of China under Grant 61772306 and Grant 62072050. *(Corresponding author: Yuan He.)*

Xiuzhen Guo, Yuan He, Zihao Yu, and Yunhao Liu are with the School of Software and BNRist, Tsinghua University, Beijing 100084, China (e-mail: guoxz16@mails.tsinghua.edu.cn; he@greenorbs.com; zh-yu17@mails.tsinghua.edu.cn).

Xiaolong Zheng is with the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100088, China (e-mail: zhengxiaolong.zxl@gmail.com).

Digital Object Identifier 10.1109/JIOT.2021.3054669

into the packet detection module of WiFi will always result in failure. Third, ZigBee does not comply with the symbol synchronization mechanism of WiFi. Even if the ZigBee packet entered the symbol synchronization process of WiFi, the synchronization will be erroneous. Last but not least, ZigBee and WiFi adopt different modulation schemes. For the quadrature demodulator of WiFi, a signal from ZigBee [modulated by direct sequence spread spectrum (DSSS) and OQPSK] contains undesirable time delay and phase offset between its I-phase and Q-phase. Directly decoding a ZigBee signal on WiFi appears to be infeasible.

In order to address the above challenges, we, in this article, propose transmitter-transparent CTC with cross-demapping, and implement it as LEGO-Fi. LEGO-Fi conceptually achieves a throughput of 250 kb/s, the ceiling speed of standard ZigBee communication. The design of LEGO-Fi stems from two key technical insights: 1) a ZigBee packet leaves distinguishable features when passing the WiFi modules and 2) compared to ZigBee's simple encoding and modulation schemes, the rich processing capacity of WiFi offers extra flexibility to process a ZigBee packet.

LEGO-Fi reuses standard WiFi modules to achieve ZigBee reception. Specifically, we devise a lightweight downsampling technique to bridge the bandwidth gap between ZigBee and WiFi. The periodicity of ZigBee preambles is preserved after passing WiFi's short preamble detection, which in turn acts as the important basis for WiFi to identify ZigBee packets. LEGO-Fi reuses the WiFi long preamble detection module to locate the start of frame delimiter (SFD) in a ZigBee packet and realizes symbol synchronization. The WiFi quadrature demodulator is reused to obtain the phase shift sequences of ZigBee signals. The key technique in LEGO-Fi is cross-demapping, which translates the CTC decoding task into a pattern identification problem. Our contributions are summarized as follows.

1) We propose LEGO-Fi, a transmitter-transparent CTC for ZigBee reception at WiFi. LEGO-Fi leaves the processing complexity at the receiver and conceptually achieve a throughput up to 250 kb/s, the ceiling speed of standard ZigBee communication. To the best of our knowledge, LEGO-Fi is the first work on physical-level CTC that unleashes the full communication capacity from a low-end device to a high-end device (e.g., ZigBee to WiFi).
2) In LEGO-Fi, we propose a key concept called cross-demapping, which leverages the mapping between the distinguishable patterns of ZigBee symbols to decode ZigBee signals. Highlights of our design mainly include: a) the lightweight downsampling technique to bridge the bandwidth gap between ZigBee and WiFi; b) exploiting the periodicity of ZigBee preambles to detect ZigBee packet at WiFi; c) the symbol synchronization scheme built on top of the SFD structure; and d) the matching filter-based cross-demapping scheme.
3) We implement LEGO-Fi on the USRP platform, and evaluate its performance under a wide range of settings. The results demonstrate that LEGO-Fi achieves a throughput of 213.6 kb/s in practice, which is, respectively, 13000× and 1200× faster than FreeBee and ZigFi, the two existing ZigBee-to-WiFi CTC approaches.

The remainder of this article is organized as follows. Section II discusses related works. Section III introduces the background and motivation of this work. We elaborate on our design in Section IV. Section V presents the evaluation results. We conclude this work in Section VI.

II. Related Works

As we mentioned in the first section, the incompatibility between technologies and the asymmetry of device capacity are the two major challenges of CTC. According to the way to cope with the challenges, we can classify the existing works into two categories: 1) packet-level CTC and 2) physical-level CTC.

Packet-Level CTC: Packet-related features are relatively easy to access, even when the packets come from a different communication technology. By manipulating the packets as information carrier, packet-level CTC builds a mutually accessible side channel. Depending on the communication contexts, the existing works utilize various packet related features, such as the RSS [17]–[20], [28], [29], packet length [16], transmission timings [21], [22], and channel state information (CSI) [23], [24], [30]. FreeBee [21] shifts the transmission timings of beacons to embed symbols. WiZig [18] adjusts the transmission power and grouping size of packets to encode multiple bits simultaneously. StripComm [19] proposes an interference-resilient CTC scheme that exploits Manchester Coding and interference cancellation technique. B2W2 [23] exploits the feature of overlapped channels to realize communication from BLE to WiFi. ZigFi [24] deliberately piggybacks the ZigBee transmission over a WiFi transmission. It leverages the CSI in the overlapped WiFi packet to build a side channel from ZigBee. DopplerFi [31] enables a two-way communication channel between BLE and Wi-Fi by injecting artificial Doppler shifts, which can be decoded by the CSI and the Gaussian frequency shift keying demodulator.

Generally speaking, the manipulation of packet transmissions can convey data while masking the aforementioned incompatibility and asymmetry. The throughput of packet-level CTC, however, is bounded by the granularity of packet manipulation, which is at the magnitude of millisecond.

Physical-Level CTC: Differing from the packet-level schemes, physical-level CTC aims at creating compliance across technologies and building the CTC channel right at the physical layer. WEBee [25] proposes physical-level emulation. It sets the payload of a WiFi frame so that a portion of this WiFi frame is recognized as a legitimate ZigBee frame by the receiver. Using a similar method, BlueBee [26] modifies the payload of BLE to emulate the signal of ZigBee. XBee [6] proposes CTC from ZigBee to BLE, which decodes a ZigBee packet by observing the bit patterns perceived at the receiver. Decoding the ZigBee signal is realized by the cross-demapping process based on a preconfigured mapping table. Recent works focus more on the the reliability and applicability of CTC.

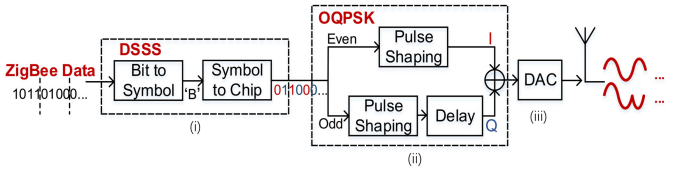


Fig. 1. Workflow of a ZigBee transmitter.

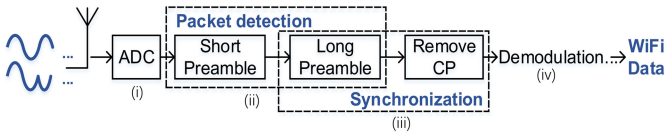


Fig. 2. Workflow of a WiFi receiver.

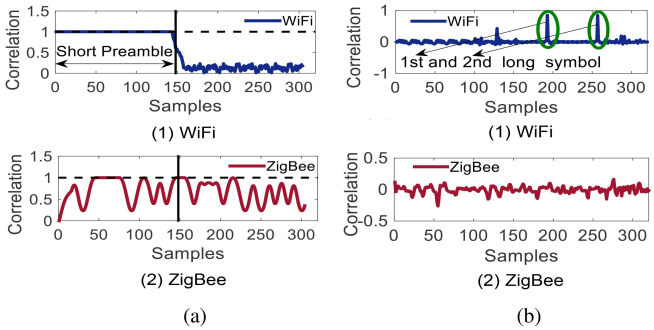


Fig. 3. (a) Short preamble detection result of WiFi and ZigBee. (b) Long preamble detection result of WiFi and ZigBee.

TwinBee [32] proposes a chip-combining coding scheme to recover chip errors introduced by imperfect signal emulation. LongBee [33] extends the communication range from WiFi to ZigBee by concentrating the effective transmission power and improving the receiver sensitivity. WIDE [34] is the latest physical-level CTC. Instead of emulating the desired phase with the WiFi QAM points, WIDE modifies the payload of WiFi to modulate the phase shift of signals, which enhances the communication reliability. Software-defined radio (SDR)-Lite [35] is an SDR receiver that empowers commodity WiFi to retrieve the in-phase and quadrature of an ambient signal and supports the in-depth analysis for CTC.

Compared with the published INFOCOM version [36], we first introduce the challenges and opportunities of physical-level CTC from ZigBee to WiFi in Section III, which helps the reader to better understand the design of LEGO-Fi. We further extend the design of LEGO-Fi to support the Bluetooth signal reception at the WiFi device in Section IV-E. We add more experiments to evaluate the performance of LEGO-Fi. Specifically, we evaluate the performance of each LEGO-Fi components in Section V-B, and we add the comparison of LEGO-Fi with SymBee in Section V-C. In Section V-D, we study the impact of ZigBee payload length, evaluate the performance of LEGO-Fi in different environments, and evaluate LEGO-Fi performance under mobility. Moreover, we add the latest CTC works as the references and update in Section II.

III. Background & Motivation

This section starts with a brief introduction to the workflows of a ZigBee transmitter and a WiFi receiver. This introduction helps us to better understand the underlying challenges of physical-level CTC between ZigBee and WiFi.

A. ZigBee Transmitter

Fig. 1 shows how a ZigBee transmitter (IEEE 802.15.4) works.

1) Every four data bits are mapped into a symbol and each symbol is then mapped into a 32-chip sequence. This process is the so-called DSSS. The bit rate and chip rate are 250 kb/s and 2Mchip/s, respectively, following the IEEE 802.15.4 standard.
2) The chip sequences are modulated by OQPSK with half sin pulse shaping. The time offset between I-phase and Q-phase is 0.5 μs, which is the reciprocal of the chip rate.
3) The modulated ZigBee signals are converted by digital analog conversion (DAC) and sent via the RF radio.

B. WiFi Receiver

The WiFi receiver works on 5 GHz and we take IEEE 802.11 g/n as an example. The workflow of a WiFi receiver is shown in Fig. 2.

1) The WiFi Rx radio digitalizes the received signals by analogy digital conversion (ADC).
2) The discrete samples of signals are sent to the packet detection module, which sequentially executes short preamble detection and long preamble detection. The detection result tells whether the received frame is a legitimate WiFi frame.
3) If a legitimate WiFi frame is identified, the receiver conducts symbol synchronization and decodes the WiFi data bits. Otherwise, the WiFi receiver discards the signals.

C. Challenges

Due to the mismatch in the bandwidths of ZigBee and WiFi, ZigBee packets cannot pass the WiFi packet detection module. For intuitive observation on the consequence, we manually feed both WiFi and ZigBee preambles to the short and long preamble detection modules of WiFi. From Fig. 3, we can see that a WiFi preamble passing the short preamble detection module presents consistently large values of correlation, while a ZigBee preamble presents apparent variances. The mismatch of bandwidth results in the mismatch of sampling rate of WiFi and ZigBee, which makes it impossible for the ZigBee preamble to pass the WiFi preamble detection module. Due to the mismatch of sampling rate, there are only a part of ZigBee preamble that can be processed within the sampling window (320 samples) of the WiFi preamble detection module. This part of ZigBee preamble cannot be detected by the WiFi preamble detection module because of the broken periodicity. Therefore, the downsampling is performed so that the pattern of ZigBee preamble can be caught in the WiFi preamble detection module.

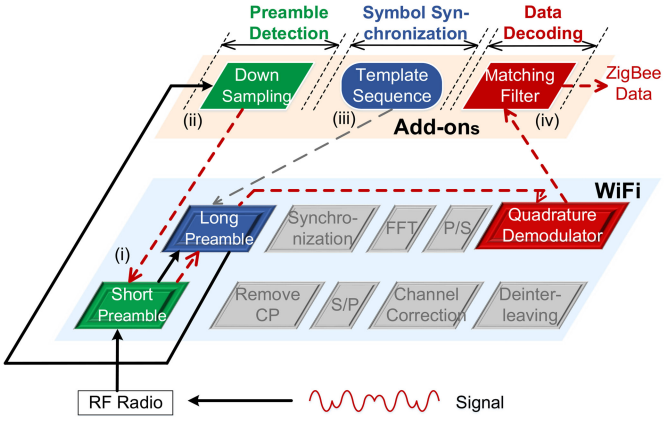


Fig. 4. Framework of LEGO-Fi.

As for the long preamble detection, Fig. 3(b)(1) shows that the correlation curve of a WiFi preamble has two obvious peaks, which correspond to the starts of two long training symbols. For a ZigBee preamble, the correlation values are seemingly random, as shown in Fig. 3(a)(2). Hence, the WiFi packet detection module always discards a ZigBee packet, as it is designed to do. Moreover, the incompatibility of the modulation schemes between ZigBee (OQPSK) and WiFi (OFDM and mQAM) makes it impossible to directly decode ZigBee packets at WiFi.

IV. Design

LEGO-Fi is a transmitter-transparent CTC for ZigBee reception at WiFi. The design of LEGO-Fi stems from two key technical insights: 1) a ZigBee packet leaves distinguishable features when passing the WiFi modules and 2) compared to ZigBee's simple encoding and modulation schemes, the rich processing capacity of WiFi offers extra flexibility to process a ZigBee packet. LEGO-Fi focuses on the CTC from ZigBee to WiFi. We select the method of WiZig [18], which leverages the energy of WiFi packets to transmit control messages from WiFi to ZigBee. In this section, we will first present an overview of LEGO-Fi and then introduce the design details.

A. Overview

LEGO-Fi leaves the processing complexity at the receiver and reuses the standard WiFi modules for the ZigBee reception. In LEGO-Fi, we propose a key concept called cross-demapping, which leverages the mapping between the phase shift sequences and ZigBee symbols to decode ZigBee signals. The framework of LEGO-Fi is shown in Fig. 4. The received signals will be down converted and digitalized into discrete samples by the WiFi RF radio. The samples are then sent to the processing modules to decode specific data bits as the following workflow.

1) The received samples are first sent to the WiFi preamble detection module to decide whether the received samples are WiFi signals. If the WiFi preamble is detected, LEGO-Fi will not process them and let the WiFi modules continue decoding the packets as the traditional WiFi receiver does.
2) Otherwise, LEGO-Fi takes over the samples and checks whether they are from ZigBee. The received samples are first processed by downsampling to obtain complete ZigBee preambles. Then, LEGO-Fi reuses the WiFi short preamble detection module to detect the periodic ZigBee preamble. If not detected, LEGO-Fi discards the samples.
3) If the periodic ZigBee preamble is detected, LEGO-Fi conducts the symbol synchronization to segment each ZigBee symbol. LEGO-Fi feeds the SFD template of ZigBee packets into the WiFi long preamble detection module to locate the SFD and accomplish the symbol synchronization.
4) Then, the samples are forwarded to the WiFi quadrature demodulator to obtain the phase shift of OQPSK signals. LEGO-Fi uses the matching filter to identify the phase shift sequences. The ZigBee signal can be decoded by the mapping between phase shift sequences and ZigBee symbols, and ZigBee symbols and data bits.

B. ZigBee Preamble Detection

ZigBee preamble detection is a prerequisite of receiving ZigBee packets at WiFi. In Section III, we have shown that even though the ZigBee packets cannot be directly detected, the periodicity of the ZigBee preamble is still reserved after passing the WiFi preamble detection module. But due to the asymmetry bandwidths, given the fixed sampling period, the WiFi receiver can only process a portion of the ZigBee signals, leading to the inevitable information loss.

LEGO-Fi devises a lightweight downsampling technique to bridge the bandwidth gap between ZigBee and WiFi. Then, LEGO-Fi can obtain more samples in the fixed sampling window, without distorting the ZigBee signals. Then, LEGO-Fi reuses the WiFi short preamble detection module to check the existence of periodicity in the information-recovered samples. If periodicity is found, a ZigBee preamble is detected. Otherwise, LEGO-Fi discards the samples and processes the next received signals.

1) Downsampling: The duration time of the WiFi preamble is 16 μs and the sampling rate is 20 MHz at a WiFi receiver, which means that the WiFi packet preamble detection module processes only 320 samples at once. But the duration of a complete ZigBee preamble is 128 μs. Hence, only 1/8 of the ZigBee preamble can be processed in the WiFi packet detection window T_1, which causes information loss. The processing result of a ZigBee preamble in T_1 is shown in Fig. 5(a). We cannot find any periodicity.

To detect the periodicity, more information needs to be included into the WiFi detection window. Therefore, we adopt downsampling and reduce the sampling rate to 2.5 MHz to capture the whole ZigBee preamble. The periodicity is easy to find in the processing results, as shown in Fig. 5(c).

Whereas, such a low rate will result in the distortion of ZigBee signals. According to the Nyquist Theorem, the sampling rate must be at least twice of the highest analog frequency component to fully recover the signal. A sampling rate of 2.5 MHz cannot recover the ZigBee signals whose

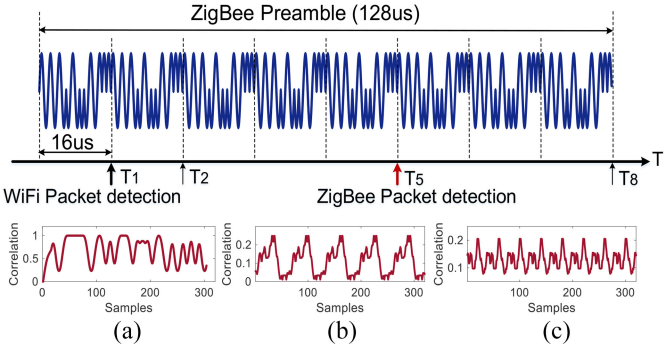


Fig. 5. Diagram of ZigBee preamble detection. (a) T = T1. (b) T = T5. (c) T = T8.

highest rate is 2 MHz. Hence, we adjust the sampling rate to 4 MHz. During the WiFi detection window, 5/8 of the ZigBee preamble can be captured. The processing results when downsampling to 4 MHz are shown in Fig. 5(b). We can find even though the periodic preamble is not completely included, it is feasible to identify the periodicity.

2) Periodicity Detection: The downsampled samples are sent to the WiFi short preamble detection module to examine the existence of periodicity. The delay correlation algorithm adopted by the WiFi short preamble detection module can be reused. If there is a ZigBee packet, the 320-sample detection window consists of five repeated symbol "0000" and each symbol has 64 samples. The delay correlation result of ZigBee preamble is

$$c_n = \sum_{k=0}^{l} r_{n+k} r^*_{n+k+16} = \sum_{k=0}^{l} r_{n+k+64} r^*_{n+k+16+64} = c_{n+64} \quad (1)$$

where r_n is the received signal and r_{n+16} is the signal delayed by 16 samples. l is the length of the moving window, which is 64 for common WiFi receivers. Since the repeated period of the ZigBee preamble is also 64, c_n, the result of performing the WiFi short preamble detection module on ZigBee preamble should also be periodic, as shown in Fig. 6(a)(1). We then feed the sequence of c_n into the WiFi short preamble detection module again to examine the existence of periodicity.

The processing result should be

$$s_n = \sum_{k=0}^{l} c_{n+k} c^*_{n+k+16}. \quad (2)$$

Fig. 6(a)(2) presents the processing result of the c_n sequence in Fig. 6(a)(1).

Specifically, LEGO-Fi adopts uniformly spaced sampling to implement downsampling and extracts one sample from every five received samples. When the number of extracted samples cumulates to 320, the WiFi detection window size, LEGO-Fi reuses the WiFi short preamble detection twice to get s_n. If there is a subsequence of s_n and each value in the subsequence is larger than the threshold, LEGO-Fi determines there are ZigBee signals. The threshold is empirically set at 0.5.

C. Symbol Synchronization

Symbol synchronization is essential for a communication system to get the start and the end of a symbol. WiFi mainly relies on the long training sequence for symbol synchronization. Due to the difference in packet formats of ZigBee and WiFi, the synchronization mechanism at WiFi receiver does not work for ZigBee packets. Even though ZigBee packets do not have the long training sequence, it has the SFD to indicate the start of the packet. The SFD is defined as "0×A7." Because of the DSSS mechanism, the symbol "0×A7" is mapped to a specific chip sequence, as shown in Fig. 6(b)(1). LEGO-Fi leverages the certainty of SFD to locate it. LEGO-Fi calculates the moving correlation between the known SFD chip sequence and the received signal, as follows:

$$\text{corr}_i = \sum_{k=0}^{127} r_{i+k} x^*_k, \quad i = 1, 2, \ldots, n - 127 \quad (3)$$

where x_k is the template SFD chip sequence, and r_n is the received signal. The length of moving correlation window is 128, equal to the size of SFD. As shown in Fig. 6(b)(1), the start of SFD is the position with the maximum value of the correlation results. If the maximum value $\text{corr}_{\max}$ is larger than a threshold, the SFD can be accurately located to the *max*-th position. If no corr_i is larger than the threshold, the SFD localization fails and LEGO-Fi will drop the packet. The threshold is empirically set at 20.

The above process can be realized by the WiFi long preamble detection module. The SFD sequence needs to be added as the template sequence.

D. ZigBee Data Decoding

Existing demodulation methods supported by WiFi cannot decode OQPSK signals due to the undesirable time delay and phase offset between the I-phase and Q-phase. Even though directly decoding ZigBee packets is infeasible, a ZigBee packet leaves distinguishable features when passing the WiFi modules. Hence, we translate the ZigBee decoding problem into the pattern identification problem.

First, LEGO-Fi reuses WiFi quadrature demodulator to calculate the product of the signal and the conjugate delayed signal. There is no need to separate I/Q signals. Then, LEGO-Fi divides the phase shift sequence into segments with a length of 64 samples. Each segment corresponds to a ZigBee symbol to be decoded. Fig. 6(c) shows the phase shift sequence of ZigBee symbol "1111". To remove the influence of dynamic channel condition, LEGO-Fi quantizes the received ZigBee samples of each segment based on a threshold filter. When the value of phase shift is greater than or equal to 0, the phase shift is quantized to 1. Otherwise, it is quantized to −1.

LEGO-Fi then leverages the matching filter to find which standard phase shift sequence is the most similar one with the current quantized phase shift segment. We mainly consider two kinds of channel noise. One of them is random Gaussian white noise, and the other is burst interference. The random Gaussian white noise will affect the phase values of all sampling points. Other burst interference signals in the environment will affect the distortion of some sampling points. The matching filter is used to calculate the similarity between the received phase sequence and the template phase sequence.

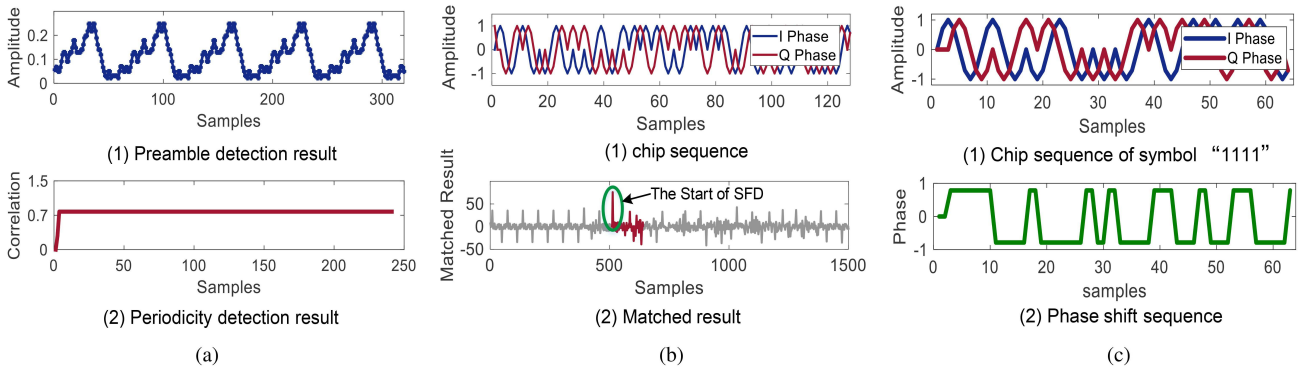


Fig. 6. (a) Detection result of ZigBee preamble and the periodicity detection result. (b) Chip sequence of SFD and the correlation result with received signals. (c) Phase shift sequence (take ZigBee symbol "1111" as an example).

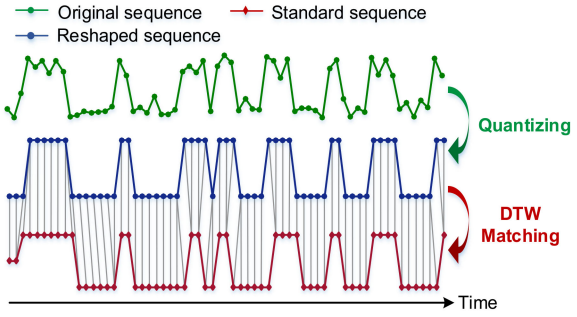


Fig. 7. Matching filter-based decoding.

Due to the randomness and burst of the channel, the received phase sequence is easy to be affected by background noise, multipath environment, and interference signal, thus causing distortion. Moreover, the received phase sequence is random and the number of sampling points may not be the same each decoding window. In these complex cases, the distance (or similarity) between the received phase sequence and the template phase sequence cannot be effectively calculated by the traditional Euclidean distance. Therefore, when there are phase distortions and length variation of the received phase sequence, we use dynamic time wrapping (DTW) as the matching filter to judge the sequence similarity and improve the reliability of decoding. It is worth noticing that DTW is not the only method that meets the requirement here. Other techniques, e.g., cross-correlation, can also be used as the matching filter.

Fig. 7 shows an example. Suppose that the 16 standard phase shift sequences are $\{Y_1(n), Y_2(n), \ldots, Y_{16}(n)\}$. For the jth received phase shift segment, we obtain a cost set $\{\text{cost}_1^j, \text{cost}_2^j, \ldots, \text{cost}_{16}^j\}$ after DTW, where cost_i^j is the similarity between the ith standard phase shift sequence and the jth received phase shift segment. The sequence with the minimal matching cost is the most similar sequence. Therefore, we can find the corresponding *symbol* according to the following equation:

$$\text{symbol} = \text{Index}(\min(\text{cost}_i), \quad i = 1, 2, \ldots, 16). \tag{4}$$

After learning the symbol id, LEGO-Fi can infer the four data bits based on the mapping relationship between symbols and data bits. In this way, LEGO-Fi achieves the ZigBee reception at WiFi successfully at the physical level.

E. CTC From Bluetooth to WiFi

In this section, we use the method of LEGO-Fi to achieve the Bluetooth signal reception at the WiFi receiver. Frequency overlapping is the precondition of CTC. In order to achieve the communication from Bluetooth to WiFi, Bluetooth needs to transmit signals in a fixed channel overlapped with WiFi, which is supported by the broadcast mode of Bluetooth. On the one hand, there is no downsampling operation when the WiFi device processes Bluetooth signals, because the duration of Bluetooth preamble is much shorter than that of ZigBee preamble. On the other hand, a new template sequence is added for Bluetooth symbol synchronization due to the difference packet formats of Bluetooth and ZigBee. The processing of Bluetooth reception at the WiFi modules has three steps: 1) Bluetooth preamble detection; 2) Bluetooth symbol synchronization; and 3) Bluetooth phase shift sequence identification.

Bluetooth Preamble Detection: The Bluetooth preamble consists of one byte "0×AA" (10101010). The duration of the Bluetooth preamble is 8 μs, which is equal to the duration of a WiFi short preamble duration. Within the detection window, the Bluetooth preamble consists of four repeated segments "10" and each segment has 40 samples. Without the operation of downsampling, we reuse WiFi short preamble detection module with the delay correlation as

$$s_n = \sum_{k=0}^{l} r_{n+k} r_{n+k+40}^* \tag{5}$$

where r_n is the received signal, r_{n+40} is the signal delayed by 40 samples, and l is the length of a moving window.

When the Bluetooth preamble is sent to the WiFi short preamble detection module, the detection result shown in Fig. 8 presents consistently large values of correlation, which is similar with the detection result shown in Fig. 3(a). So the Bluetooth signals will be reserved when passing the WiFi modules.

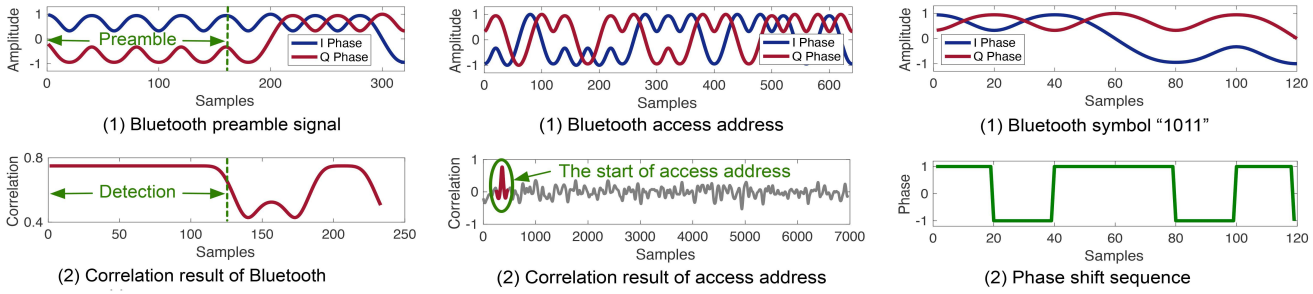


Fig. 8. (a) Detection result of Bluetooth preamble. (b) Detection result of Bluetooth access address. (c) Bluetooth phase shift sequence (take Bluetooth "1011" as an example).

Bluetooth Symbol Synchronization: In order to get the start position of the Bluetooth payload, the access address in a Bluetooth packet can be used to achieve synchronization. Different Bluetooth packets have the same fixed access address, which consists of 8 B "0×8E89BED6." LEGO-Fi calculates the moving correlation between the known access address and the received signal. The detection result is shown in Fig. 8 and the start of access address is the position with the maximum value of the correlation results.

Bluetooth Phase Shift Sequence Identification: We suppose that 4 Bluetooth bits make up a Bluetooth symbol and there are 16 (2^4) different Bluetooth symbols. Fig. 8 shows the phase shift sequence of Bluetooth symbol "1011." A matching filter based on DTW can also be used to identify these phase shift sequences.

F. Discussion

With the proliferation of IoT devices in all kinds of scenarios, we may foresee the ever increasing needs of CTC in wireless systems. Supporting CTC with commercial wireless devices is very likely to be a default practice in the near future. As a software implementation on USRP, LEGO-Fi maximizes the reuse of standard WiFi modules and sheds lights on future hardware implementation of CTC-compatible WiFi devices. Note that in the design of LEGO-Fi, standard WiFi modules are reused as much as possible. As long as the basic root privilege of each module (i.e., the interoperability of the interface and the register) is provided, LEGO-Fi can decode ZigBee signals by rewiring the WiFi modules and adjusting the control logic of the data flow. According to the scheme of LEGO-Fi, the current design of WiFi devices can obtain the new ability of CTC, with limited modification to the circuit, while all the functions and properties of a WiFi device are completely preserved.

The most calculations in our work are completed by the WiFi module, and such calculations will consume the resources on the WiFi device. For example, DTW uses the dynamic programming to calculate the similarity between the received phase sequence and the template phase sequence with the computational complexity of $O(n^2)$. Compared with the decoding process of WiFi packets, including the short preamble detection, long preamble detection, FFT, QAM demapping, and deinterleave, the decoding process of LEOGO-Fi symbols is relatively simple.

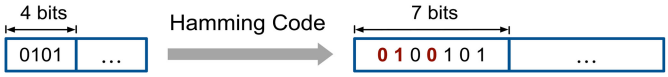


Fig. 9. Illustration of hamming coding.

G. Reliability Enhancement

The link coding mechanism is a potential method to correct decoding errors and enhance the reliability of decoding. Specifically, Hamming Code (7.4) can be used to recover the decoding errors. As shown in Fig. 9, we add 3-b check data to every 4-b data and there 3-b check data can correct 1-b error data. The Hamming Code (7, 4) has the capacity of error detection rate of (1/4), which is sufficient to handle the decoding errors.

V. Evaluation

In this section, we conduct extensive experiments to evaluate the performance of LEGO-Fi. The LEGO-Fi transmitter is the commercial TelosB node with CC2420 and the prototype of LEGO-Fi is implemented on a USRP-RIO platform.

A. Experimental Setup

For comparison, we implement three state-of-the-art works: 1) FreeBee [21]; 2) ZigFi [24]; and 3) SymBee [27]. FreeBee embeds CTC symbols by the shifting transmission timing of a beacon. FreeBee detects the packet by RSS and uses folding to determine the shifts of the transmission timings. In the default setting of FreeBee, a 6-b symbol can be detected when folding five beacons. The beacon rate is 20 beacon/s, as same as the default setting of FreeBee in [21]. ZigFi piggybacks ZigBee packets over WiFi data packets and leverages the changes of CSI to convey CTC symbols. The CSI variations of eight packets can encode 1-b symbol. SymBee uses the specific ZigBee symbols to yield unique patterns when WiFi conducts idle listening. ZigBee symbol combination of 6 and 7 is used to convey CTC symbol 0 while ZigBee symbol combination of E and F represents CTC symbol 1.

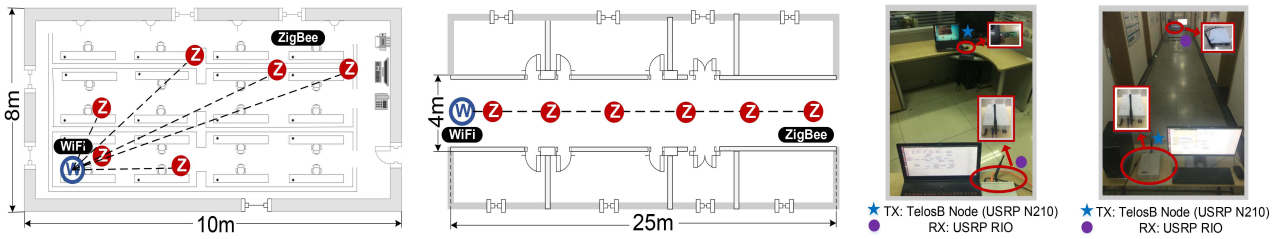


Fig. 10. (a) Floor plans of the lab. (b) Floor plans of the hallway. (c) Experiment settings in the lab. (d) Experiment settings in the hallway.

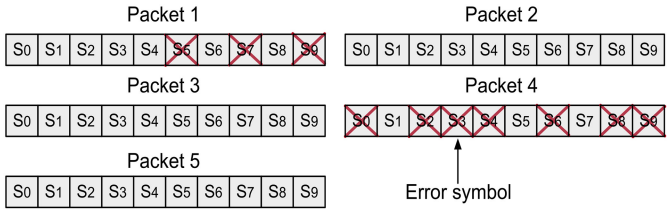


Fig. 11. Definition of SER and PRR.

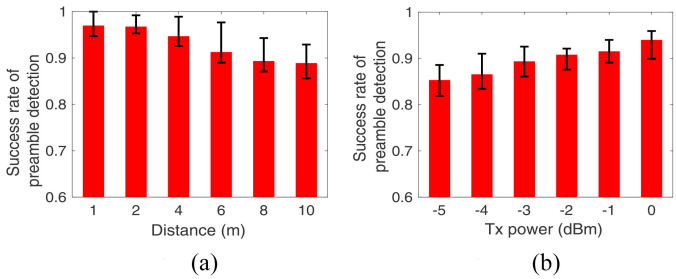


Fig. 12. Success rate of preamble detection. (a) Distance. (b) Power.

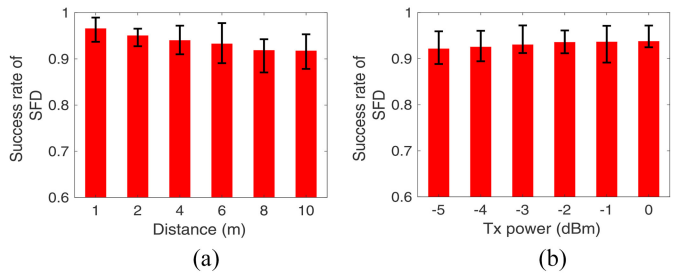


Fig. 13. Success rate of SFD localization. (a) Distance. (b) Power.

Since the implementation of ZigFi relies on an existing WiFi link, a commercial WiFi device is used to transmit WiFi packets with a packet length of 145 B. The data packet rate is set to 2000 packets/s, as same as the default setting in [24]. Another WiFi device installed Intel 5300 NIC and CSITool software platform is used as the ZigFi receiver with the sampling rate of 2 KHz.

Unless otherwise specified, the transmission power of TelosB node is set to 0 dBm. The channel is set to ZigBee channel 23 and the receiving channel is set to WiFi channel 11. In LEGO-Fi, the packet size is 24 B. The payload covers all the 16 ZigBee symbols and the number of each symbol is equal. The experiments are conducted in two environments: 1) our lab on campus and 2) the hallway of a building. The floor plans are shown in Fig. 10(a) and (b), and the corresponding settings are shown in Fig. 10(a) and (b).

We take symbol error ratio (SER) and packet error ratio (PRR) as the key metrics to evaluate LEGO-Fi's performance. As shown in Fig. 11, SER is the error ratio of the number of corrupted symbols to the total number of transmitted symbols, rather than the probability of a symbol to be corrupted. The corrupted symbols may be distributed in one or more packets. Packet reception ratio (PRR) is the ratio of the number of correctly decoded packets to the total number of packets. When a packet fails the CRC check, even if there is only one corrupted symbol, the packet will be lost.

B. Benchmarks

LEGO-Fi consists of three components: 1) ZigBee preamble detection; 2) SFD localization; and 3) matching filter-based decoding. In this section, we evaluate the performance of the LEGO-Fi components to demonstrate the effectiveness of reusing WiFi modules.

1) Preamble Detection: We first measure the success rate of ZigBee preamble detection in LEGO-Fi. As shown in Fig. 12(a), the success rate of ZigBee preamble detection decreases with the increase of the distance between the sender and the receiver. When the distance is 10 m with the transmission power of 0 dBm, the success rate of preamble detection is 0.889. The success rate of ZigBee preamble detection increases with the increase of the transmission power as shown in Fig. 12(b). When the transmission power is -5 dBm with the distance of 5 m, the success rate of preamble detection is 0.853.

2) SFD Localization: We then study the success rate of SFD localization when varying the distance and the transmission power. Given the transmission power of 0 dBm, the success rate decreases from 0.966 to 0.918 when the distance increases from 1 to 10 m as shown in Fig. 13(a). Given the distance of 5 m, the success rate increases from 0.922 to 0.938 when the transmission power increases from -5 to 0 dBm as shown in Fig. 13(b). Compared to the results of preamble detection, we can find that the success rate of SFD localization is a little higher. This is because the symbol synchronization mechanism of WiFi long preamble detection is similar with the SFD localization. Whereas, the detection of ZigBee preamble needs to downsampling a longer sequence and reuses WiFi short preamble detection module twice.

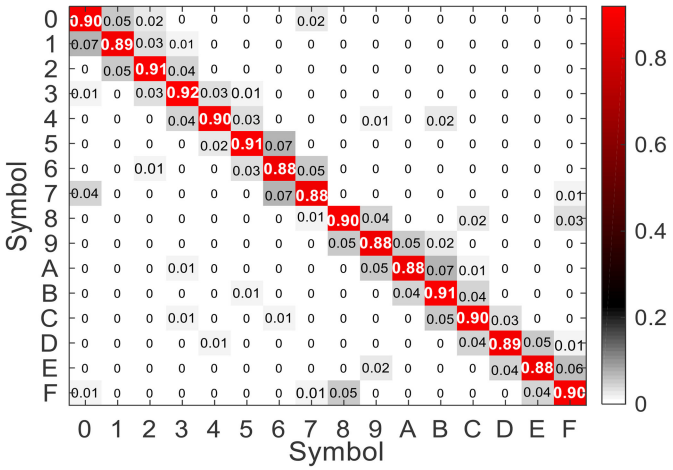


Fig. 14. Decoding accuracy for different symbols.

3) Matching Filter-Based Decoding: We evaluate the performance of matching filter-based decoding under two settings. The confusion matrix of the average decoding accuracy is shown in Fig. 14. Overall, the average decoding accuracy of different symbols varies from 0.88 to 0.92, with an average accuracy of 0.90 for all symbols. The decoding errors mainly occur between adjacent symbols. The reason behind this phenomenon is that the chip sequences of ZigBee symbols are related to each other through cyclic shift and conjugation and the adjacent chip sequences are similar.

C. Overall Performance Comparison

First, we analyze and compare the theoretical capacity of LEGO-Fi and three the-state-of-art works: 1) FreeBee [21]; 2) ZigFi [24]; and 3) SymBee [27]. The capacity of FreeBee depends on the beacon rate. In the default setting of FreeBee, the beacon rate is 20 beacons/s and five beacons can modulate 6 b. Therefore, the theoretical capacity of FreeBee = 24 bps = 20 beacons/s × 6/5 b/beacon. For ZigFi, the capacity is decided by the packet transmission rate. In the default setting of ZigFi, the packet transmission rate is 2000 packets/s and eight packets can modulate 1 b. Hence, the theoretical capacity of ZigFi = 250 bps = 2000 packets/s × 1/8 b/beacon. For SymBee, the capacity relies on the duration of ZigBee symbol. In the default setting of SymBee, one CTC symbol consists of two ZigBee symbols and each ZigBee symbol lasts 4 μs. Hence, the theoretical capacity of SymBee = 31.25 kb/s = 1 s÷8 μs. Since LEGO-Fi is a physical-level CTC that allows the ZigBee transmitter sends standard ZigBee packet, the theoretical capacity of LEGO-Fi is same with the capacity of ZigBee, which is 250 kb/s.

We then conduct experiments to compare the performance of three works in practice. The distance between the ZigBee sender and the WiFi receiver is 5 m. All settings are same with the default settings. The experiments of three works are conducted in our lab, with a consistent ambient environment and a similar network interference condition.

The comparison results are shown in Fig. 15. FreeBee achieves a throughput of 17.2 bps with a SER of 0.114. ZigFi

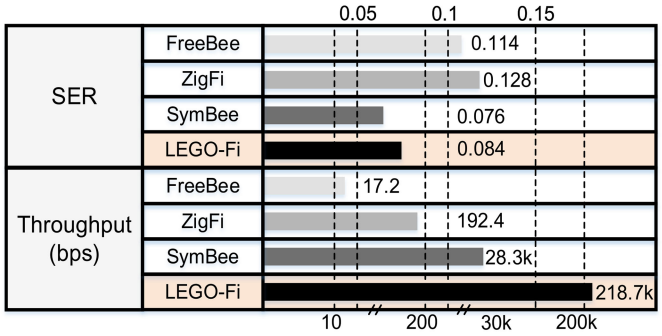


Fig. 15. Overall performance comparison.

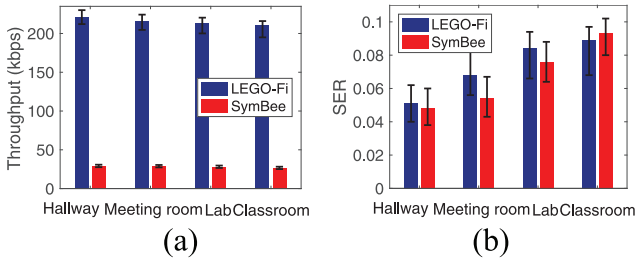


Fig. 16. Comparison of LEGO-Fi and SymBee in different scenarios. (a) Throughput. (b) SER.

achieves a throughput of 192.4 bps with a SER of 0.128. SymBee achieves a throughput of 28.3 kb/s with a SER of 0.076. The measured performance of both FreeBee, ZigFi, and SymBee is close to the reported. The throughput of LEGO-Fi is up to 213.6 kb/s, which is 13000×, 1200×, and 7.5× higher than that of FreeBee, ZigFi, and SymBee, respectively. The SER of LEGO-Fi is 0.084, which is lower than FreeBee and ZigFi, revealing the physical-level manipulation is more reliable than packet-level controls.

We also observe the SER and throughput of LEGO-Fi and SymBee in four different scenarios: 1) hallway; 2) meeting room; 3) lab; and 4) classroom. The SER of LEGO-Fi is slightly higher than SymBee shown in Fig. 16(a), and the throughput of LEGO-Fi is much better than SymBee shown in Fig. 16(b). The experiment results show that LEGO-Fi has better practical performance than SymBee.

D. Performance Under Different Settings

In this section, we vary the distance between the sender and the receiver, the transmission power and the payload length to study their impacts on LEGO-Fi in terms of PRR, SER, and throughput. We also evaluate LEGO-Fi in mobile scenarios.

1) Impact of Transmission Power and Distance: The RSS depends on both the transmission power and the distance between the transmitter and receiver. We plot the PRR of LEGO-Fi under different transmission powers and distances in Fig. 17. We can find the PRR varies in the range of 0.778–0.937. Even when the transmission is low (−5 dBm) and the distance is far (10 m), the PRR of LEGO-Fi is still above 0.77.

2) Impact of Payload Length: We then study the impact of payload length on LEGO-Fi, ZigFi, and FreeBee. We change the payload length of ZigBee from 24 to 120 B. The packet

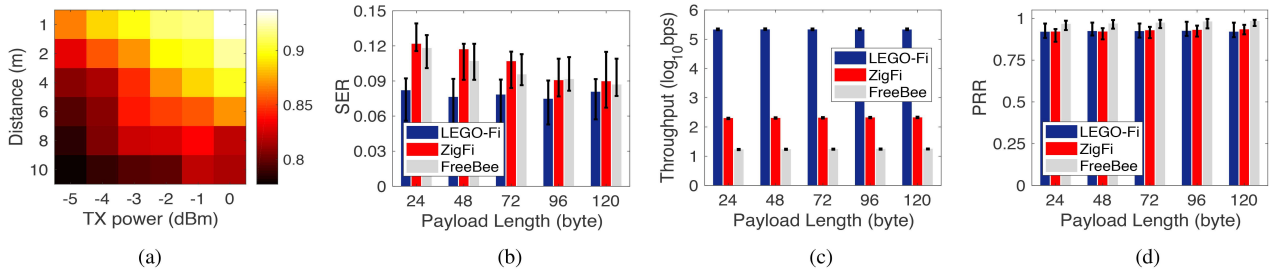


Fig. 17. (a) PRR vs. Tx power & Distance of LEGO-Fi. (b) SER with different ZigBee payload lengths. (c) Throughput with different ZigBee payload lengths. (d) PRR with different ZigBee payload lengths.

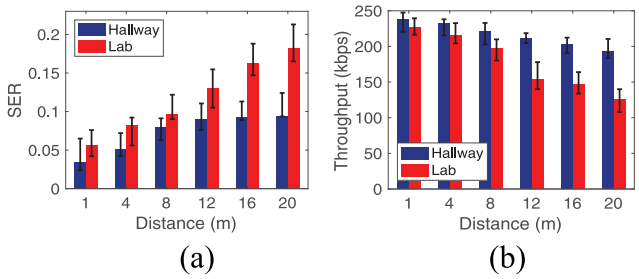


Fig. 18. LEGO-Fi performance in the lab and the hallway. (a) SER. (b) Throughput.

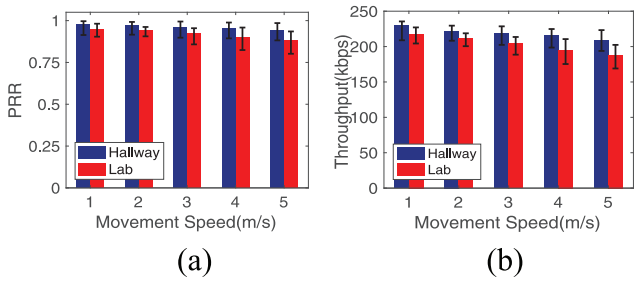


Fig. 19. LEGO-Fi performance under mobility. (a) PRR. (b) Throughput.

transmission power is 0 dBm and the distance between the sender and the receiver is 2 m.

The PRR of LEGO-Fi relies on ZigBee preamble periodicity and SFD detection instead of the whole packet. Hence, the impact of payload length on the PRR of LEGO-Fi is relatively small. Whereas, ZigFi and FreeBee rely on a relatively long payload to reliably detect the RSSI or CSI variations. From Fig. 17(a), we can find the SER of LEGO-Fi is relatively stable, but both FreeBee and ZigFi have larger SERs when the payload length is short. Throughput and PRR show similar results, as shown in Fig. 17(b) and (c). The results reveal that FreeBee and ZigFi, the packet-level CTC, prefer longer ZigBee packets but LEGO-Fi can receive the standard ZigBee packets without any requirement on the packet length.

3) Impact of Environment: We evaluate LEGO-Fi in different environments. We conduct the experiments in another larger lab and the hallway. Fig. 18(a) and (b) presents the SER and the throughput of LEGO-Fi in two environments, respectively. We can find that when the distance increases, the SER increases and the throughput degrades in both environments. When the distance is 20 m, the SER in the hallway and the lab is 0.098 and 0.182, respectively. When the distance is 20 m, the throughput of LEGO-Fi in the hallway is 194.8 kb/s and only 125.1 kb/s in the lab. The increasing speed of the SER and dropping speed of the throughput in the lab are much faster than the ones in the hallway. This is because the environment in the lab is more complicated than that in the hallway, leading to more serious multipath influences on the received signals.

4) Impact of Mobility: We also evaluate the LEGO-Fi performance with a mobile ZigBee sender in two sites: 1) the lab and 2) the hallway. In the experiments, the ZigBee sender transmits ZigBee packets at a power of 0 dBm. The packet length is 24 B. A volunteer carrying the ZigBee node moves at a speed from 1 to 5 m/s in a straight line. We control the movement of this volunteer smoothly to ensure the fixed speed of the ZigBee node. For each setting, the experiment is repeated 50 times. The duration of all the experiments is twenty minutes.

Fig. 19(a) and (b) shows that the PRR and the throughput of LEGO-Fi with varying speeds. Both the PRR and the throughput decrease with the increase of speed. Whereas, even when the moving speed is 5 m/s, the performance of LEGO-Fi is still acceptable, achieving the throughput of 195.1 and 170.8 kb/s with the PRR of 0.891 and 0.832, in the hallway and the lab, respectively.

E. Performance of LEGO-Fi When Receiving Bluetooth Signals

We conduct experiments to evaluate the performance of LEGO-Fi when receiving Bluetooth signals in four different scenarios: 1) hallway; 2) meeting room; 3) lab; and 4) classroom. The Bluetooth node (CC2650) transmits Bluetooth packets with the length of 40 B. The Bluetooth channel is set at Bluetooth channel 38 and the WiFi channel is set at WiFi channel 4. The distance between the Bluetooth node and the WiFi device varies from 2 to 10 m. The experimental results are shown in Fig. 20(a) and (b). The SER of LEGO-Fi in the hallway, meeting room, lab, and classroom is 0.046, 0.062, 0.068, and 0.073 when the distance is 2 m. The theoretical throughput of LEGO-Fi is 1 Mb/s. As shown in Fig. 20(b), the throughput of LEGO-Fi in the hallway, meeting room, lab, and classroom is 934.8, 920.4, 904.1, and 899.2 kb/s. Moreover, the SER of LEGO-Fi increases and the throughput of LEGO-Fi decreases with the increase of distance.

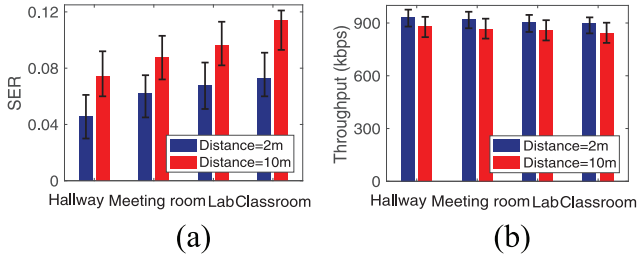


Fig. 20. LEGO-Fi performance when receiving Bluetooth signals. (a) SER. (b) Throughput.

VI. Conclusion

CTC has great potential in IoT applications. Our work focuses on "the shortest plank" of CTC, i.e., communication from ZigBee to WiFi. Our proposal called LEGO-Fi unleashes the full communication capacity from ZigBee to WiFi. LEGO-Fi is a transmitter-transparent CTC, which leaves the processing complexity at the receiver. LEGO-Fi reuses the standard WiFi modules and presents cross-demapping to decode ZigBee signals. The main take away from this article is twofolded: 1) the asymmetry in devices' capacity offers extra flexibility to process CTC signals and 2) besides packet-level metrics, the unique protocol features also provide important basis in building the side channel for CTC. In the future, we will also try to deploy LEGO-Fi in real-world applications.

References

[1] T. Zachariah, N. Klugman, B. Campbell, J. Adkins, N. Jackson, and P. Dutta, "The Internet of Things has a gateway problem," in *Proc. ACM HotMobile*, 2015, pp. 27–32.

[2] C. J. M. Liang, K. Chen, N. B. Priyantha, J. Liu, and F. Zhao, "RushNet: Practical traffic prioritization for saturated wireless sensor networks," in *Proc. ACM SenSys*, 2014, pp. 105–118.

[3] X. Zhang and K. G. Shin, "Enabling coexistence of heterogeneous wireless systems: Case for ZigBee and WiFi," in *Proc. ACM MobiHoc*, 2011, p. 6.

[4] W. Zeng, A. Arora, and K. Srinivasan, "Low power counting via collaborative wireless communications," in *Proc. ACM/IEEE IPSN*, 2013, pp. 43–54.

[5] S. Singh, K. Sundaresan, S. V. Krishnamurthy, X. Zhang, A. Khojastepour, and S. Rangarajan, "TRINITY: A practical transmitter cooperation framework to handle heterogeneous user profiles in wireless networks," in *Proc. ACM MobiHoc*, 2015, pp. 297–306.

[6] W. Jiang, S. Min Kim, Z. Li, and T. He, "Achieving receiver-side cross-technology communication with cross-decoding," in *Proc. ACM MobiCom*, 2018, pp. 639–652

[7] Z. Chi, Y. Li, Y. Yao, and T. Zhu, "PMC: Parallel multi-protocol communication to heterogeneous iot radios within a single WiFi channel," in *Proc. IEEE ICNP*, 2017, pp. 1–10.

[8] Z. Yin, Z. Li, S. Min Kim, and T. He, "Explicit channel coordination via cross-technology communication," in *Proc. ACM MobiSys*, 2018, pp. 178–190.

[9] X. Zheng, D. Xia, X. Guo, L. Liu, Y. He, and H. Ma, "Portal: Transparent cross-technology opportunistic forwarding for low-power wireless networks," in *Proc. ACM MobiHoc*, 2020, pp. 241–250.

[10] D. Xia, X. Zheng, L. Liu, C. Wang, and H. Ma, "c-Chirp: Towards symmetric cross-technology communication over asymmetric channels," in *Proc. IEEE SECON*, 2020, pp. 1–9.

[11] Y. Yan, P. Yang, X.-Y. Li, T. Yue, Z. Lan, and L. You, "ZIMO: Building cross-technology mimo to harmonize zigbee smog with wifi flash without intervention," in *Proc. ACM MobiCom*, 2013, pp. 465–476.

[12] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma, "ZiFi: Wireless LAN discovery via zigbee interference signatures," in *Proc. ACM MobiCom*, 2010, pp. 49–60.

[13] C. Ma, J. Liu, X. Tian, H. Yu, Y. Cui, and X. Wang, "Interference exploitation in D2D-enabled cellular networks: A secrecy perspective," in *Proc. IEEE Trans. Commun.*, vol. 63, no. 1, pp. 229–242, Jan. 2015.

[14] S. Hu *et al.*, "Data acquisition for real-time decision-making under freshness constraints," in *Proc. IEEE RTSS*, 2015, pp. 185–194.

[15] Q. Pu *et al.*, "Low latency geo-distributed data analytics," in *Proc. ACM SIGCOMM*, 2015, pp. 421–434.

[16] Y. Zhang and Q. Li, "HoWiES: A holistic approach to ZigBee assisted WiFi energy savings in mobile devices," in *Proc. IEEE INFOCOM*, 2013, pp. 1366–1374.

[17] K. Chebrolu and A. Dhekne, "Esense: Communication through energy sensing," in *Proc. ACM MobiCom*, 2009, pp. 85–96.

[18] X. Guo, X. Zheng, and Y. He, "WiZig: Cross-technology energy communication over a noisy channel," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[19] X. Zheng, Y. He, and X. Guo, "StripComm: Interference-resilient cross-technology communication in coexisting environments," in *Proc. IEEE INFOCOM*, 2018, pp. 171–179.

[20] Z. Yin, W. Jiang, S. M. Kim, and T. He, "C-Morse: Cross-technology communication with transparent morse coding," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[21] S. M. Kim and T. He, "FreeBee: Cross-technology communication via free side-channel," in *Proc. ACM MobiCom*, 2015, pp. 317–330.

[22] W. Jiang, Z. Yin, S. M. Kim, and T. He, "Transparent cross-technology communication over data traffic," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[23] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu, "B2w2: N-way concurrent communication for IoT devices," in *Proc. ACM SenSys*, 2016, pp. 245–258

[24] X. Guo, Y. He, X. Zheng, L. Yu, and O. Gnawali, "ZigFi: Harnessing channel state information for cross-technology communication," in *Proc. IEEE INFOCOM*, 2018, pp. 360–368.

[25] Z. Li and T. He, "WEBee: Physical-layer cross-technology communication via emulation," in *Proc. ACM MobiCom*, 2017, pp. 2–14.

[26] W. Jiang, Z. Yin, R. Liu, Z. Li, S. M. Kim, and T. He, "BlueBee: A 10, 000x faster cross-technology communication via PHY emulation," in *Proc. ACM SenSys*, 2017, pp. 1–13.

[27] S. Wang, S. M. Kim, and T. He, "Symbol-level cross-technology communication via payload encoding," in *Proc. IEEE ICDCS*, 2018, pp. 500–510.

[28] X. Guo, Y. He, and X. Zheng, "WiZig: Cross-technology energy communication over a noisy channel," *IEEE/ACM Trans. Netw.*, vol. 28, no. 6, pp. 2449–2460, Dec. 2020.

[29] X. Guo *et al.*, "Aloba: Rethinking ON-OFF keying modulation for ambient lora backscatter," in *Proc. ACM SenSys*, 2020, pp. 192–204.

[30] X. Guo, Y. He, X. Zheng, L. Yu, and O. Gnawali, "ZigFi: Harnessing channel state information for cross-technology communication," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 301–311, Feb. 2020.

[31] W. Wang, S. He, L. Sun, T. Jiang, and Q. Zhang, "Cross-technology communications for heterogeneous IoT devices through artificial doppler shifts," *IEEE Trans. Wireless Commun.*, vol. 18, no. 2, pp. 796–806, Feb. 2019.

[32] Y. Chen, Z. Li, and T. He, "TwinBee: Relable physical-layer cross-technology communication with symbol-level coding," in *Procs. IEEE INFOCOM*, 2018, pp. 153–161.

[33] Z. Li and T. He, "LongBee: Enabling long-range cross-technology communication," in *Procs. IEEE INFOCOM*, 2018, pp. 162–170.

[34] X. Guo, Y. He, J. Zhang, and H. Jiang, "Wide: Physical-level CTC via digital emulation," in *Proc. IEEE/ACM IPSN*, 2019, pp. 49–60.

[35] W. Jeong *et al.*, "SDR receiver using commodity wifi via physical-layer signal reconstruction," in *Proc. ACM MobiCom*, 2020, pp. 32:1–32:14.

[36] X. Guo, Y. He, X. Zheng, Z. Yu, and Y. Liu, "LEGO-Fi: Transmitter-transparent CTC with cross-demapping," in *Proc. INFOCOM*, 2019, pp. 2125–2133.

Xiuzhen Guo received the B.E. degree from the School of Electronic and Information Engineering, Southwest University, Chongqing, China, in 2016. She is currently pursuing the Ph.D. degree with Tsinghua University, Beijing, China.

Her research interests include wireless network co-existence and cross technology communication.

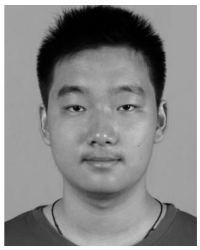
Yuan He received the B.E. degree from the University of Science and Technology of China, Hefei, China, the M.E. degree from the Institute of Software, Chinese Academy of Sciences, Beijing, China, and the Ph.D. degree from Hong Kong University of Science and Technology, Hong Kong, in 2003, 2006, and 2010, respectively.

He is an Associate Professor with the School of Software and TNLIST, Tsinghua University, Beijing. His research interests include wireless networks, Internet of Things, and pervasive and mobile computing.

Dr. He is a member of ACM.

Xiaolong Zheng received the B.E. degree from the Dalian University of Technology, Dalian, China, in 2011, and the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong, in 2015.

He is currently an Associate Professor with the School of Computer Science and Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing, China. From September 2015 to August 2018, he was a Postdoctoral Researcher with Tsinghua University, Beijing. His research interests include Internet of Things, wireless networks, and intelligent computing.



Zihao Yu is currently pursuing the Ph.D. degree with the School of Software, Tsinghua University, Beijing, China, under the supervision of Prof. Y. He.

His main research interests are in the field of cross technology communication.

Yunhao Liu (Fellow, IEEE) received the B.S. degree from Automation Department, Tsinghua University, Beijing, China, and the M.S. and Ph.D. degrees in computer science and engineering from Michigan State University, East Lansing, MI, USA, in 1995 and 2004, respectively.

He is currently an MSU Foundation Professor and the Chairperson of Department of Computer Science and Engineering, Michigan State University, and holds Chang Jiang Chair Professorship with Tsinghua University. His research interests include sensor network and pervasive computing, peer-to-peer computing, IOT, and supply chain.

Dr. Liu is currently serves as the Editor-in-Chief of *ACM Transactions on Sensor Networks*. He is an ACM Distinguished Speaker. He is a Fellow of ACM.