WIDE: Physical-level CTC via Digital Emulation

Xiuzhen Guo, Yuan He, Jia Zhang, Haotian Jiang
School of Software & BNRist, Tsinghua University
{guoxz16,jia-zhan15,jht15}@mails.tsinghua.edu.cn,heyuan@mail.tsinghua.edu.cn

ABSTRACT

Cross-Technology Communication (CTC) is an emerging technique that enables direct communication across different wireless technologies. Recent works achieve physical-level CTC by emulating the standard time-domain waveform of the receiver. This method faces the challenges of inherent unreliability due to the imperfect emulation. Different from analog emulation, we propose a novel concept named digital emulation, which stems from the following insight: The receiver relies on the phase shift to decode symbols rather than the shape of analog time-domain waveform. There are lots of phase sequences which satisfy the requirement of phase shift. The distortions of these phase sequences after WiFi emulation are different. We have the opportunity to select an appropriate phase sequence with the relatively small emulation errors to achieve a reliable CTC. The key point of digital emulation is generic and applicable to a set of CTCs, where the transmitter has a wider bandwidth for emulation and the receiver decoding is based on the phase shift. In this paper, we implement our proposal as WIDE, a physical-level CTC via digital emulation from WiFi to ZigBee. We conduct extensive experiments to evaluate the performance of WIDE. The results show that WIDE significantly improves the Packet Reception Ratio (PRR) from 41.7% to 86.2%, which is 2× of WEBee's, an existing representative physical-level CTC.

CCS CONCEPTS

• **Networks → Network protocols**.

KEYWORDS

Cross-Technology Communication; Digital emulation

ACM Reference Format:
Xiuzhen Guo, Yuan He, Jia Zhang, Haotian Jiang. 2019. WIDE: Physical-level CTC via Digital Emulation. In *The 18th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2019) (IPSN '19), April 16–18, 2019, Montreal, QC, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3302506.3310388

1 INTRODUCTION

The proliferation of Internet of Things (IoT) applications brings about the increasingly dense deployments of various wireless devices [17, 24, 25, 27, 39, 40], which causes a more serious coexistence of heterogeneous wireless technologies [9, 26, 28, 30, 42]. Under such circumstances, cross-technology communication (CTC) is an emerging technique to enable direct communication among devices that follow different communication standards [10, 29, 32, 43]. 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 [2, 20, 36, 37]. CTC not only provides a new way to manage wireless networks, but also enhances the ability of in-situ data exchange for emerging IoT applications (e.g. industrial surveillance and smart home), where seamless data collection and interoperation are desired [1, 4, 30, 35]. In addition, compared with traditional gateways, CTC avoids the hardware cost and deployment complexity [14, 19, 34].

Early CTC works establish communication channels based on the packet-level, which manipulate transmitted packets and use the packet length [38], the received signal strength [3, 13, 33, 41], or the transmission timings [16, 18] as the information carrier. Recent works propose physical-level CTC. WEBee [21] uses the high-speed WiFi radio to emulate the standard half sine waveform of the low-speed ZigBee radio by carefully selecting the payload of the WiFi packet. BlueBee [23] modifies the payload of BLE to emulate the signal of ZigBee. XBee [15] realizes CTC from ZigBee to BLE based on cross-demapping, which decodes the ZigBee packet by observing the bit patterns obtained at the BLE receiver.

Almost all existing physical-level CTCs are realized by analog emulation method, namely that the sender emulates the standard time-domain waveform of the receiver. Whereas, the analog emulation-based CTCs may not be suitable for applications that requires high reliability (e.g. data dissemination and reliable network flooding) due to the limited Packet Reception Ratio (PRR). This is because the analog emulated signal cannot perfectly match the desired signal. The protocol standard of the sender is different from that of the receiver and the hardware restrictions also affect the emulation result. So there are inevitable distortions between the desired waveform and the emulated waveform. To realize high reliability in the practical applications, the emulated packets have to be retransmitted. As a result, the efficiency and throughput of analog emulation-based CTC degrade.

We find that the decoding of the ZigBee receiver doesn't rely on the specific shape of the time-domain waveform. Intrinsically, the ZigBee receiver decodes data based on the binary phase shift between the sampling points. As a result, in addition to the standard half sine waveform, other types of waveforms can also be properly decoded as long as the phase shift sequences generated by these waveforms satisfy the requirement of the binary phase shift sequence at the receiver. Therefore, different from analog emulation, we propose a novel concept **Digital Emulation** for physical-level CTC. Instead of emulating the standard time-domain waveform of the receiver, we directly emulate the phase shift. There are lots of phase sequences whose phase shift sequences satisfy the requirement of the receiver for correct decoding. These phase sequences can be emulated by constructing different payloads of the sender and the emulation errors of these phase sequences are different. Therefore, we have the opportunity to select an appropriate phase sequence with relatively small emulation errors to achieve a reliable CTC.

The concept of digital emulation is generic and applicable to a set of CTCs, where the transmitter has a wider bandwidth for emulation and the receiver decoding is based on the phase shift. In this paper, we implement our proposal as WIDE, a physical-level CTC via digital emulation from WiFi to ZigBee. Instead of emulating the standard half sine waveform of ZigBee, WIDE selects an appropriate phase sequence to emulate the phase shift sequence directly, which makes the decoded binary phase shift sequence at the ZigBee receiver more accurate and robust.

Specifically, we first select the square wave as a basic unit to generate a set of ladder shaped phase sequences. The corresponding phase shift sequence of the ladder shaped phase sequence is stable within a demodulation period and satisfies the requirement of a ZigBee symbol. WiFi modifies the content of the payload to accomplish the process of emulation [21], which makes the phase shift of the payload resembles that of the desired phase shift. Then we adopt a greedy algorithm to generate the initial phase sequence. In addition, we analyze the errors caused by Cyclic Prefix (CP) during the WiFi emulation and propose an algorithm named Secondary Adjustment based on FEedback (SAFE) to further optimize the phase sequence. In this way, we can get the appropriate phase sequence for phase shift emulation. The ZigBee packet reception ratio of WIDE can be improved from 41.7% to 86.2%, which is 2× of WEBee's, an representative physical-level CTC. Our contributions are summarized as follows.

- We propose a novel concept, digital emulation, for physical-level CTC. Instead of emulating the standard time-domain waveform of the receiver, we select an appropriate phase sequence to emulate the phase shift of the receiver directly. Without modifying the firmware or hardware of both WiFi and ZigBee devices, WEBee is a transparent design that can be easily deployed in existing WiFi infrastructure with broad applicability. The method of digital emulation is generic and applicable to a set of CTCs, where the transmitter has a wider bandwidth for emulation and the receiver decoding is based on the phase shift.
- We design WIDE, a physical-level CTC via digital emulation from WiFi to ZigBee. In WIDE, we address several challenges, including the phase sequence generation and the phase sequence optimization, to select an appropriate phase sequence for phase shift emulation.
- We implement WIDE on both the USRP N210 platform and the commodity device. The experimental results demonstrate that WIDE achieves high reliable CTC from WiFi to ZigBee. WIDE improves the Packet Reception Ratio (PRR) of ZigBee packets from 41.7% to 86.2%, which is 2× of WEBee's, an existing representative physical-level CTC.

The rest of this paper is organized as follows. Section 2 discusses related works. Section 3 compares the analog emulation and the digital emulation. We elaborate on our design in Section 4. Section 5 presents the evaluation results. We conclude this work in Section 6.

2 RELATED WORKS

The incompatibility between technologies and the asymmetry of device capacity are the two major challenges of CTC. According to the method to cope with the challenges, we can classify the existing works into two categories: packet-level CTC and physical-level CTC.

Packet-level CTC. By manipulating the packets as information carrier, packet-level CTC builds an accessible side channel for CTC, such as the received signal strength [3, 13, 33, 41], the packet length [38], the transmission timings [16, 18], and the channel state information [7, 11]. FreeBee [18] embeds symbols into beacons by shifting their transmission timings. The date rate of FreeBee is limited by the beacon rate which is usually 102.4ms per beacon for commercial WiFi devices, however. Other works propose the energy profile as a new information carrier to exchange the data without a gateway. Esense [3] is the first work that uses energy sampling realizing data transmission from the WiFi to the ZigBee device. It aims at building an alphabet of implicit data using the packet duration information. Since the communication channel is intrinsically noisy, it is not a trivial to reduce the harmful impact of noise. The impact of noise on CTC throughput is analyzed in WiZig [13], which adjusts the transmission power to encode multiple bits. StripComm [41] is a novel interference resilient CTC tailored to the coexisting environment. StripComm leverages the idea of Manchester Coding and proposes a novel interference-aware coding mechanism. HoWiEs [38] controls the WiFi packet length and encode bits by the length of packet on-air time. C-Morse [33] uses the combination of the short WiFi packets and the long WiFi packets with short intervals to construct the recognizable energy patterns at the ZigBee receiver. EMF [6] leverages the independency among different window sizes for embedding different pieces of information in a string of existing packets. B2W2 [7] and ZigFi [11] exploit the feature of Channel State Information (CSI) to realize communication from BLE to WiFi and ZigBee to WiFi respectively. 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. Physical-level CTC aims at creating compliance across technologies and building the CTC channel right at the physical layer [19, 31]. WEBee [21] proposes physical-level emulation, which uses the high-speed WiFi radio to emulate the standard ZigBee time-domain signals of the low-speed ZigBee radio. Specifically, WEBee chooses the payload of a WiFi frame so that a portion of this WiFi frame is recognized by commodity ZigBee devices transparently as a legitimate ZigBee frame. In order to improve the reliability of WEBee, TwinBee [5] proposes a chip-combining coding scheme to recover chip errors introduced by imperfect signal emulation. LongBee [22] is another improved CTC work of WEBee, which extends the communication range of CTC to support long-range IoT applications. In terms of signal emulation, LongBee works similar to WEBee. Moreover, LongBee combines the high transmission power of WiFi and the fine receiving sensitivity of ZigBee together to increase the CTC communication

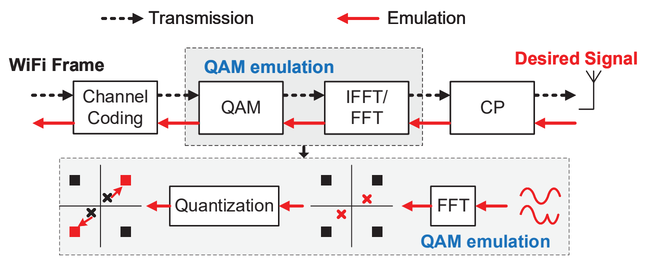


Figure 1: The workflow of analog emulation

range significantly. LEGO-Fi [12] achieves physical-level CTC from ZigBee to WiFi by leveraging cross-demapping, which stems two key technique insights. First, a ZigBee packet leaves distinguishable features when passing the WiFi modules. Second, compared to ZigBee's simple encoding and modulation schemes, the rich processing capacity of WiFi offers extra flexibility to process a ZigBee packet. PMC [8] enables parallel communication to multiple ZigBee and WiFi devices. BlueBee [23] modifies the payload of BLE to emulate the signal of ZigBee. XBee [15] realizes CTC from ZigBee to BLE based on cross-decoding, which decodes a ZigBee packet by observing the bit patterns obtained at the BLE receiver. Scylla [14] is a software control CTC which allows multiple wireless stacks to coexist on top of a single radio chip, thereby simultaneously offering multiple communication interfaces.

Almost all existing physical-level CTCs are based on the analog emulation, where the sender emulates the standard time-domain waveform of the receiver. Due to the incompatibility of different protocol standards and the hardware restrictions, the analog emulated signal can not perfectly match the desired waveform, which results in low PRR. The limited reliability may restrict its wider applications. In this paper, we propose a physical-level CTC via digital emulation, where the sender emulates the phase shift of the receiver directly. There are lots of phase sequences which satisfy the requirement of phase shift. We have the opportunity to select an appropriate phase sequence with the relatively small emulation errors to achieve a reliable CTC. We compare the analog emulation and the digital emulation in the following section.

3 ANALOG EMULATION VS. DIGITAL EMULATION

In this Section, we compare the analog emulation and the digital emulation. We introduce the workflow and limitation of the physical-level CTC via analog emulation. We further introduce the motivation and benefit of the physical-level CTC via digital emulation.

3.1 Analog Emulation

3.1.1 The workflow of analog emulation. In order to achieve the physical-level CTC via analog emulation, the payload of a WiFi frame is elaborately selected to construct a legitimate ZigBee frame via emulating the ZigBee standard half sine waveform closely. A ZigBee symbol with $16\mu s$ has to be segmented and each $4\mu s$-segment is emulated by a WiFi symbol. The process of analog emulation is shown in Fig .1. The desired ZigBee signal is fed into the FFT module and we select the nearest QAM constellation points to construct the payload and emulate the ZigBee signal. After selecting the payload, the process of WiFi transmission is a reverse direction. The WiFi sender adds the cyclic prefixing (CP) to the time-domain signal and then sends it by using the RF radio, just like sending the normal WiFi signal. The entire procedure is transparent to the hardware layer of WiFi device and all of the modification is conducted on the software layer.

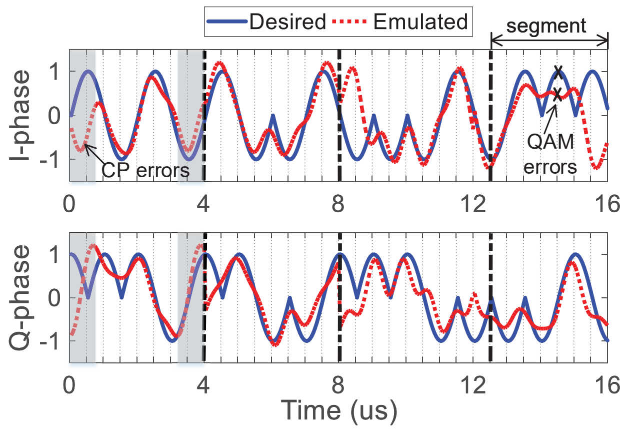


Figure 2: Desired and emulated signals via analog emulation

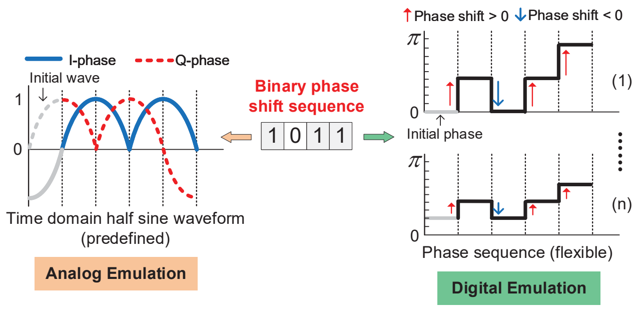


Figure 3: The comparison of analog emulation and digital emulation

3.1.2 The limitation of analog emulation. Due to the incompatibility of different protocol standards and the hardware restrictions, the analog emulated signal cannot perfectly match the desired signal. The analog emulation result of ZigBee symbol "0" is shown in Fig. 2. We can find that the emulated signals have distortion compared with the standard ZigBee half sine signals. As for analog emulation, there are mainly two types of intrinsic errors.

QAM emulation errors. QAM emulation is the core of analog emulation, where the standard ZigBee time-domain signals are fed into the FFT of WiFi to find the corresponding QAM constellation points. WiFi's predefined QAM points are limited and discrete, so time-domain signals of ZigBee may not be perfectly mapped to these QAM points predefined by WiFi. In addition, WiFi has 64 subcarriers and there are only seven subcarriers overlapping with ZigBee. As a result, only seven nearest QAM points with

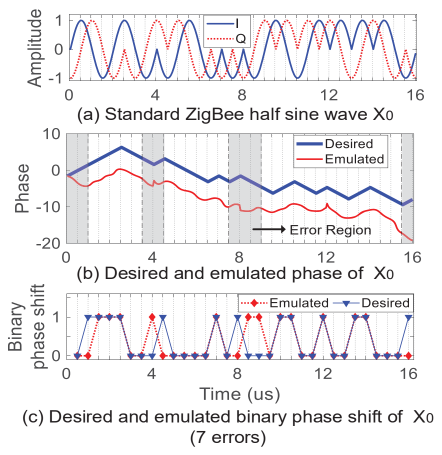


Figure 4: The emulation result of the standard half sine wave

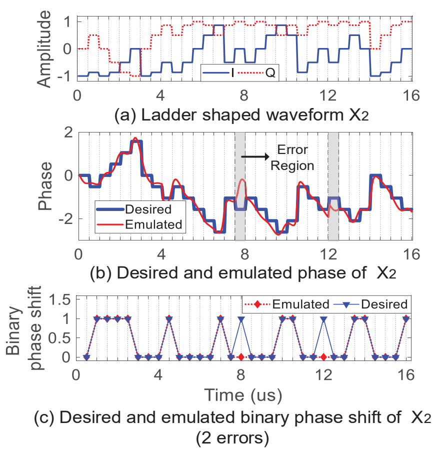


Figure 5: The emulation result of a ladder shaped waveform

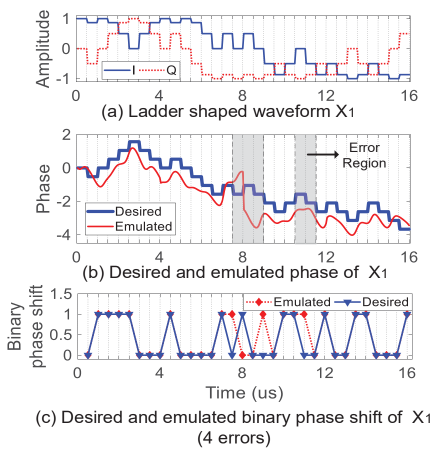


Figure 6: The emulation result of another ladder shaped waveform

the minimum Euclid Distance to the FFT coefficients are selected to emulate the standard ZigBee time-domain signals. When the ZigBee receiver demodulates the emulated signal, quantization errors cannot be avoided.

CP errors. Another source of emulation errors comes from the WiFi's cyclic prefixing (CP). CP is a 0.8us guard interval in each WiFi symbol, which is copied from the right WiFi symbol and pasted into the left of this symbol. In this way, the front segment of WiFi signals is same with the end segment of WiFi signals. Whereas, there is no such repetition in ZigBee signals. As a result, the CP errors of emulated signals are also out of the control. Furthermore, the desired signals are predefined and fixed. So the above analog emulation errors are inevitable.

3.2 Digital Emulation

3.2.1 The feasibility of digital emulation. We find that the decoding of the ZigBee receiver doesn't directly rely on the specific shape of waveform. Intrinsically, ZigBee uses phase shift to modulate symbols. ZigBee outputs "1" if the phase shift is bigger than 0° and otherwise outputs "0". After collecting 32 binary phase shifts, the ZigBee receiver maps this binary phase shift sequence into a 4-bit symbol, according to DSSS process.

Based on the finding that the receiver decoding is based on the phase shift, we propose digital emulation, where the sender directly produces proper sequence of phase shift for emulation. As shown in Fig. 3, there are lots of phase sequences which satisfy the requirement of binary phase shift sequence of ZigBee. Different phase sequences correspond to different waveforms. WiFi can construct different payload to emulate these waveforms. So in addition to the standard ZigBee half sine waveform, other types of waveforms can also be correctly decoded as long as these waveforms have the same binary phase shift sequences.

We conduct several experiments to verify the feasibility of digital emulation. The standard half sine waveform of ZigBee symbol "F" is shown in Fig. 4(a). The emulated phase sequence and the binary phase shift sequence are shown in Fig. 4(b) and Fig. 4(c) respectively. Due to the emulation distortion, except the first and the last, there are 6 wrong binary phase shift values. Another ladder shaped waveform is shown in Fig. 5(a). Its corresponding desired phase sequence and emulated phase sequence are shown in Fig. 5(b). The decoded binary phase shift sequence is shown in Fig. 5(c). We find that there are only 2 wrong bits, which can be easily mapped to the ZigBee symbol "F" according to DSSS.

3.2.2 The flexibility and challenges of digital emulation. Compared with analog emulation, digital emulation is more flexible and robust. The phase sequence with desirable phase shift sequence is not unique. Although the binary phase shift sequence of a ZigBee symbol is predefined and fixed, there are lots of phase sequences which satisfy the requirement of the binary phase shift sequence. Different phase sequences correspond to different time-domain waveforms. The performance of WiFi to emulate different phase sequence based on QAM emulation is different. Therefore, we have the opportunity to reduce the QAM errors and CP errors of WiFi emulation by selecting an appropriate phase sequence. But the selection of an appropriate phase sequence is challenging. Not all phase sequences that satisfy the binary phase shift requirement will have a better emulation result. Another ladder shaped waveform is shown in Fig. 6(a). Its phase sequence and decoding result are shown in Fig. 6(b) and Fig. 6(c) respectively. There are 4 wrong binary phase shift values except the first and the last. So how to select an appropriate phase sequence for WiFi emulation remains a challenging task and needs further study.

4 DESIGN

In this section, we will first present an overview of WIDE and then introduce the design details.

4.1 Overview

The framework of WIDE is shown in Fig. 7 and the workflow of WIDE is as follows.

(i) **Phase sequence generation:** First, WIDE selects the square wave as a basic unit to generate a set of ladder shaped phase sequences, which satisfy the binary phase shift requirement. We

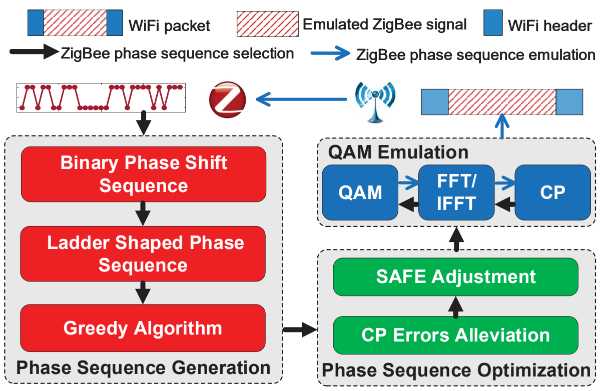

WiFi packet
Emulated ZigBee signal
WiFi header
ZigBee phase sequence selection
ZigBee phase sequence emulation
Binary Phase Shift Sequence
Ladder Shaped Phase Sequence
Greedy Algorithm
Phase Sequence Generation
QAM Emulation
QAM
FFT/ IFFT
CP
SAFE Adjustment
CP Errors Alleviation
Phase Sequence Optimization

Figure 7: The framework of WIDE

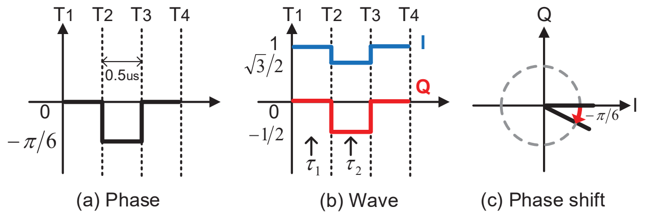


Figure 8: The basic idea of WIDE

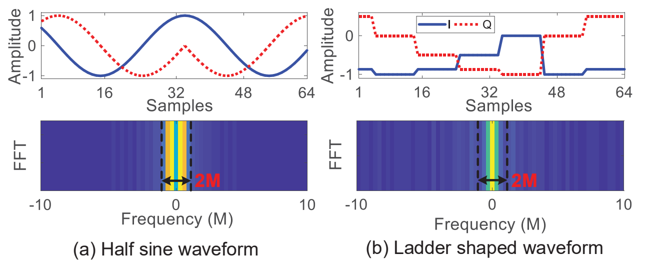


Figure 9: The FFT results of half sine waveform and ladder shaped waveform

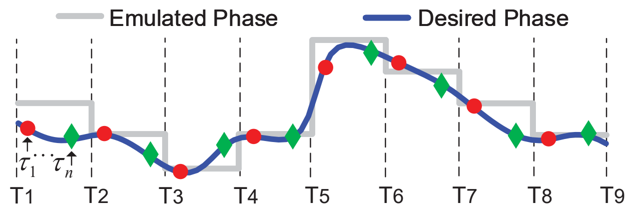


Figure 10: The illustration of SHD

propose the metric Statistical Hamming Distance (SHD) to quantify the degree of distortion between the desired phase sequence and the emulated phase sequence. Then we adopt a greedy algorithm to generate an initial phase sequence for phase shift emulation.

(ii) **Phase sequence optimization:** We analyze the CP errors after WiFi emulation and alleviate the CP errors by slightly adjusting the phase at some specific positions. Furthermore, WIDE proposes an algorithm named Secondary Adjustment based on FEedback (SAFE), which adjusts the phase sequence again according to the feedback result of WiFi emulation. In this way, we get an appropriate phase sequence for WiFi emulation.

(iii) **Phase sequence emulation:** The phase sequence corresponds to a desired waveform. The desired waveform is fed into the FFT module and we select the nearest QAM constellation points to construct the WiFi payload and emulate the desired waveform. After emulation, the phase sequence of WiFi payload resembles to desired phase sequence. WiFi header, preamble, and tail are ignored by the ZigBee receiver. Then the WiFi payload can be considered as a legitimate ZigBee frame and ZigBee symbols can be decoded successfully.

Therefore, the key point of the digital emulation is the selection of the appropriate phase sequence. Among 16 ZigBee symbols, each of them corresponds to a binary phase shift sequence. We need to select 16 appropriate phase sequences to realize the digital emulation for ZigBee symbols. We operate the process of selecting phase sequence locally and then get a mapping table from symbol to phase sequence. This mapping table can be loaded on the WiFi device prior to running WIDE so that the WiFi device is able to emulate these phase sequences by elaborately construct the payload. We introduce the design techniques, including phase sequence generation and phase sequence optimization more clearly as follows.

4.2 Phase Sequence Generation

4.2.1 Waveform unit. First, we need to select an wave as a basic unit to generate the phase sequence which satisfied the phase shift requirement of the receiver. Specifically, each ZigBee symbol corresponds to a 32-bit binary phase shift sequence and the ZigBee receiver demodulates the phase shift every $0.5\mu s$. If the binary phase shift is 1, the phase within two demodulation periods needs to increase and vice versa. For example, we vary the phase from 0 to $-\frac{\pi}{6}$ within $T_1 - T_3$ as shown in Fig. 8(a) and itąŕs corresponding time-domain waveform is shown in Fig. 8(b). There are two samples at τ_1 and τ_2, which satisfy that $T_1 \le \tau_1 \le T_2$, $T_2 \le \tau_2 \le T_3$, and $\tau_2 - \tau_1 = 0.5\mu s$. The phase shift between these two samples is $-\frac{\pi}{6}$, which is lower than $0°$ and the corresponding binary phase shift is 0, as shown in Fig. 8(c).

In order to guarantee the phase shift within a demodulation period is stable, we select the square wave as a basic unit to generate the phase sequence and the corresponding waveform. The frequency component required for emulating the square wave is higher than the sine wave, however, this problem is not difficult to handle. What we need to emulate is a ladder shaped waveform as shown in Fig. 9(b), as the binary phase shift sequence is composed of many consecutive "0" or "1". From the FFT results of the standard half sine waveform and the ladder shaped waveform, we can find that the frequency components of the ladder shaped waveform are also concentrated in 2M. Therefore, it is feasible to emulate the ladder shaped waveform with the limited number of subcarriers within 2M bandwidth at the WiFi sender.

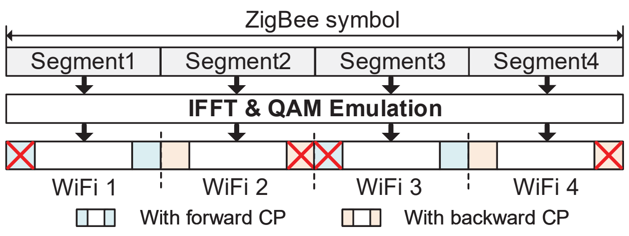


Figure 11: The emulation result with CP errors

4.2.2 SHD Metric. ZigBee decodes its bits based on binary phase shift sequence of the signal waveform. Digital emulation aims at choosing an appropriate phase sequence to make the decoded binary phase shift sequence of ZigBee after WiFi emulation more accurate and robust. Whereas, due to the limited availability of subcarriers, QAM quantization errors, and the impact of CP, there are inevitable distortion between the desired waveform and the emulated waveform. To quantify the degree of distortion helps us optimize the phase sequence and reduce the emulation errors.

As shown in Fig. 10, the desired phase sequence and the emulated phase sequence are separately shown by the gray line and the blue line. The hamming distance between the decoded binary phase shift sequence and the predefined binary phase shift sequence can be used to characterize the distortion between the emulated phase sequence and the desired phase sequence.

Whereas, the hamming distance changes with the variation of the position of the first sample. The demodulation period of the binary phase shift is $0.5\mu s$. The ZigBee receiver can decode different binary phase shift sequences when the first sample starts at different positions. As a result, we propose Statistical Hamming Distance (SHD) to quantify the degree of distortion between the desired phase sequence and the emulated phase sequence. SHD is used as an objective function to select a phase sequence with minimal emulation errors regardless of the sampling positions. The start position of the sampling obeys uniform distribution. We suppose the start position of sampling can be $\tau_1, \tau_2, ..., \tau_n$ and the corresponding hamming distance is $H_1, H_2, .., H_n$. SHD is defined as

$$SHD = \frac{1}{n}(H_1 + H_2 + ... + H_n) \tag{1}$$

The larger n is, the more convincing that the SHD can characterize the degree of distortion. We conduct many experiments and find that the performance gains from increasing the choice n are limited. We set n at 5 since this configuration already meets the requirement.

4.2.3 Phase sequence initialization. We suppose that the binary phase shift sequence of a ZigBee symbol is

$$\Delta\Phi = \{\Delta\Phi_1, \Delta\Phi_2, ..., \Delta\Phi_{n-1}, \Delta\Phi_n\}, n = 1, 2, ..., 32 \tag{2}$$

The phase sequence is

$$\Phi = \{\Phi_1, \Phi_2, ..., \Phi_{n-1}, \Phi_n\}, n = 1, 2, ..., 32 \tag{3}$$

If the initial phase is φ and the phase shift between two consecutive phases is $\Delta\varphi$, the phase sequence in Eq. (3) can be generated by

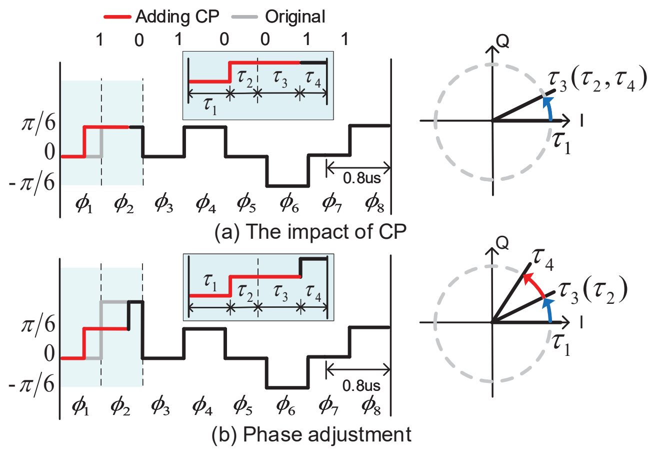


Figure 12: Forward CP and $\Delta\Phi 1 = \Delta\Phi 7$

$$\Phi_j = \begin{cases} \varphi & j=1 \\ \Phi_{j-1}+\Delta\varphi & j=2, ..., 32 \quad (\Delta\Phi_{j-1}=1) \\ \Phi_{j-1}-\Delta\varphi & j=2, ..., 32 \quad (\Delta\Phi_{j-1}=0) \end{cases} \tag{4}$$

The waveform corresponds to this phase sequence is

$$x(n) = I(n) + Q(n) = cos(\Phi_n) + i * sin(\Phi_n), n = 1, 2, ..., 32 \tag{5}$$

According to Eq. (4), there are two factors that affect the phase sequence. One is the initial phase and the other one is the phase shift between two phases. The initial phase φ can be any value in $[0, 2\pi)$ and the phase shift $\Delta\varphi$ can be any value in $[0, \pi)$. Many sets of $(\varphi, \Delta\varphi)$ will generate different phase sequences. In order to simplify this problem and reduce the computation cost, we discretize the value of φ and $\Delta\varphi$. Therefore, the optimization function is

$$\min \ SHD \tag{6}$$

$$s.t \begin{cases} \varphi = m * \frac{1}{12}\pi & m = 0, 1, 2, ..., 23 \\ \Delta\varphi = n * \frac{1}{12}\pi & n = 0, 1, 2, ..., 11 \end{cases} \tag{7}$$

We adopt a greedy algorithm to get the appropriate set of $(\varphi, \Delta\varphi)$ and generate the initial phase sequence Φ.

4.3 Phase Sequence Optimization

4.3.1 CP errors alleviation. CP has a harmful impact on the WiFi emulation and we mainly focus on alleviating the CP errors in this section.

A ZigBee symbol is 16 μs and it has to be segmented as 4 segments before emulation. Each segment corresponding to a 8-bit phase sequence can be emulated by a WiFi symbol. CP is copied from the right of the WiFi symbol and pasted into (overwrite) the left of the symbol. A selective boundary flipping method proposed in WEBee [21] can be used to alleviate the harmful impact of CP. As shown in Fig. 11, the left of segment2 and segment4 is copied to the right before the WiFi emulation. In this way, this method can disperse the impact of CP to the left/right-most and middle boundaries. The CP in segment1/3 is named as forward CP and the CP in segment2/4 is named as backward CP. Forward CP affects the first binary phase

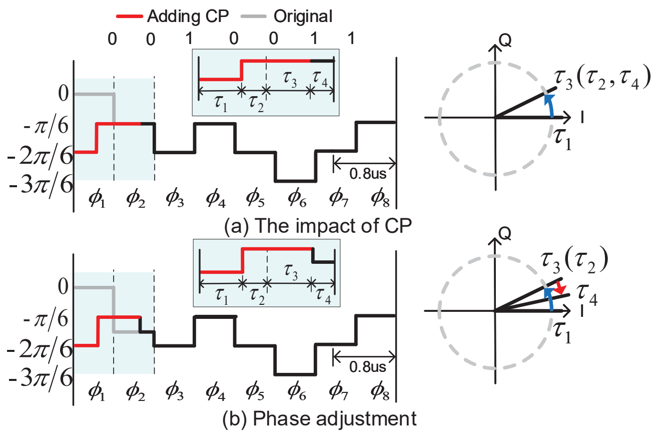


Figure 13: Forward CP and ΔΦ1 ≠ ΔΦ7

shift value ΔΦ1 and backward CP affects the last binary phase shift value ΔΦ7 in a ZigBee segment.

We adjust the phase at the specific position of a ZigBee segment to further alleviate the harmful impact of CP. The specific optimization method is related to the type of the CP and the requirement of the binary phase shift values ΔΦ1 and ΔΦ7. Next, we take the forward CP as an example to analyze the impact of CP and propose the corresponding optimization method.

ΔΦ1 = ΔΦ7 & Forward CP. We suppose the binary phase shift sequence is $\Delta\Phi = \{\Delta\Phi_1, \Delta\Phi_2, ..., \Delta\Phi_7\} = \{1010011\}$, the original phase sequence $\Phi = \{\Phi_1, \Phi_2, ..., \Phi_8\}$ is shown by the gray/black line in the left of Fig. 12(a). If there is a forward CP, the phase sequence after adding CP is shown by the red/black line in the left of Fig. 12(a). We analyze the demodulation result of the samples affected by the CP as shown in the right of Fig. 12(a). We suppose a demodulation period is divided into two time slots, τ_1 and τ_2. When the first sample is within in τ_1, the second sample is within in τ_3 and the demodulated result is 1. Whereas, if the first sample is within in τ_2, the second sample is within in τ_4 and the demodulated result is wrong. The duration of τ_1 and τ_2 is 0.3 μs and 0.2 μs respectively, so the original error probability (EP) is 0.4.

In order to correct the demodulation error, the phase within τ_4 needs to be larger than the phase within τ_2. It is worth noting that phase within τ_4 is original Φ_2 and the phase within τ_2 is original Φ_8. So we increase original Φ_2 by ΔΦ to make the new Φ_2 larger than original Φ_8. The adjustment result is shown by the gray/black line in the left of Fig. 12(b) and the phase sequence after adding CP is shown by the red/black line in the left of Fig. 12(b). In this way, as shown in the right of Fig. 12(b), the demodulation result of the samples affected by the CP is right no matter where the the first sample is.

ΔΦ1 ≠ ΔΦ7 & Forward CP. Whereas, not all the CP errors of phase sequence can be corrected. If $\Delta\Phi_1$ is different with $\Delta\Phi_7$, the phase sequence after adding CP is shown by the red/black line in the left of Fig. 13(a). We can find that no matter the first sample is within τ_1 or τ_2, the demodulation result is wrong. Due to the different binary phase shift requirement of $\Delta\Phi_1$ and $\Delta\Phi_7$, the phase within τ_1 is always larger than the phase within τ_3. So the demodulation

CP Type	$\Delta\Phi_1$	$\Delta\Phi_7$	EP	Φ_2	Φ_7
Forward	1	1	0	$\Phi_8 + \Delta\Phi$	
	1	0	0.6		
	0	0	0	$\Phi_8 - \Delta\Phi$	
	0	1	0.6		
Backward	0	0	0		$\Phi_1 + \Delta\Phi$
	1	0	0.6		
	1	1	0		$\Phi_1 - \Delta\Phi$
	1	0	0.6		

Table 1: The optimization of CP errors

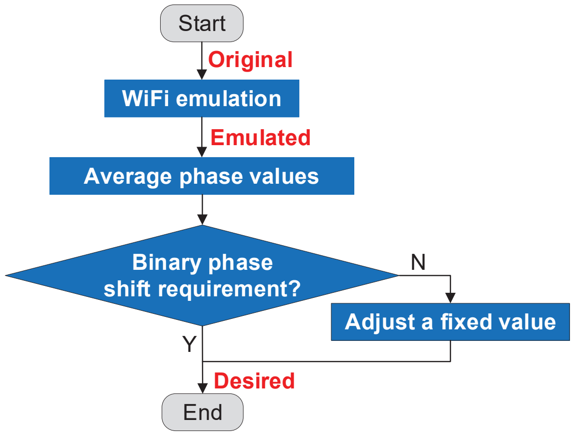


Figure 14: The workflow of SAFE

error when the first sample is within τ_1 is inevitable. Whereas, we can decrease original Φ_2 by ΔΦ to make new Φ_2 smaller than original Φ_8. The adjustment result is shown as Fig. 13(b). If the first sample is within in τ_2, the second sample is within in τ_4 and the demodulated result is 0. If the first sample is within in τ_1, the second sample is within in τ_3, the demodulated result is wrong and this error is inevitable. The duration of τ_1 and τ_2 is 0.3 μs and 0.2 μs respectively, so EP can be decreased to 0.6.

The condition of backward CP is similar and we omit the specific analysis due to the limited space.

In summary, there are totally 16 CP optimization conditions, including forward CP or backward CP, same or different $\Delta\Phi_1$ and $\Delta\Phi_7$. The optimization of all cases is shown in Table. 1. When $\Delta\Phi_1$ is equal to $\Delta\Phi_7$, the EP caused by CP can be eliminated. When $\Delta\Phi_1$ is different to $\Delta\Phi_7$, the EP caused by CP can also be reduced to 0.6. In this way, the harmful impact of CP can be effectively alleviated.

4.3.2 *SAFE Algorithm.*

In order to further optimize the phase sequence, we propose an algorithm named Secondary Adjustment based on FEedback (SAFE).

After the process of phase sequence generation in subsection 4.2, we select an initial phase sequence by greedy algorithm. We adjust the phase values at the specific positions and get a new phase sequence to alleviate the harmful impact of CP. The phase sequence corresponds to a ladder shaped waveform and WiFi constructs the payload to emulate this waveform closely. Due to the limited

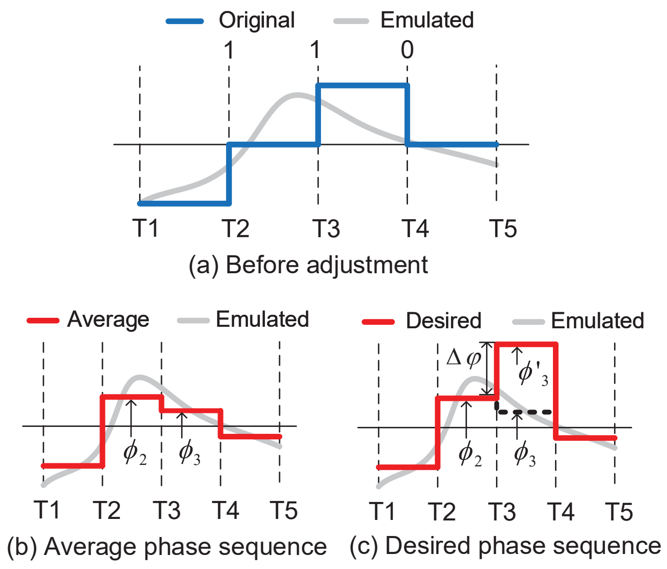


Figure 15: An example of SAFE

available QAM points and QAM quantization errors, the emulated phase sequence has distortions compared with the desired phase sequence. We adopt SAFE algorithm shown in Fig. 14 to further adjust the phase sequence according to the feedback result after WiFi emulation. The process of SAFE algorithm is as follows.

(1) The phase sequence generated by the previous steps is fed into the WiFi emulation modules shown in Fig. 1 and we can obtain the emulated phase sequence.

(2) We calculate the average phase value of the emulated phase sequence within a demodulation period as a new phase value to construct an adjusted phase sequence.

(3) We verify whether the adjusted phase sequence meets the binary phase shift requirement of the ZigBee symbol. If the phase shift sequence of the adjusted phase sequence is right, the adjusted phase sequence is the desired phase sequence. If the phase shift sequence of the adjusted phase sequence contradicts the requirement of the ZigBee symbol, we directly further adjust a fixed phase shift based on the previous adjusted phase sequence until zigbee's decoding requirement is met.

We take Fig. 15(a) as an example. The original phase sequence generated by previous steps and the emulated phase sequence after WiFi emulation are shown in the blue and gray lines respectively. We calculate the average phase value within each demodulation period of the emulated phase sequence as a new phase value. As shown in Fig. 15(b), the new calculated phase values are $\Phi_1, \Phi_2, \Phi_3, \Phi_4$. Whereas, the new phase value obtained by this method may contradict the requirement of the binary phase shift. The average phase value Φ_3 of the emulated phase sequence within (T_3, T_4) is lower than the average phase value Φ_2 of the emulated phase sequence within (T_2, T_3). Whereas, the required binary phase shift is "1". The phase sequence is contradict with the phase shift requirement of ZigBee decoding.

In this condition, we directly adjust an fixed phase shift based on the previous phase value. As shown in Fig. 15(c), we adjust the

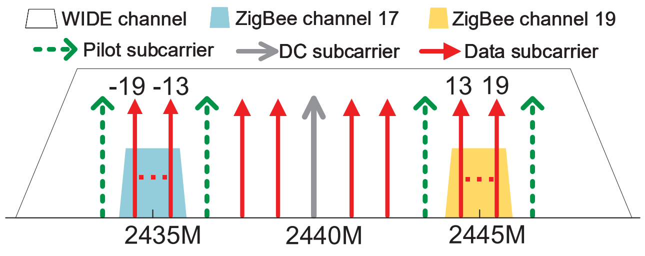


Figure 16: Channel Mapping for parallel communication

phase value within (T_3, T_4) as Φ'_3, where $\Phi'_3 = \Phi_2 + \Delta\varphi$ and $\Delta\varphi$ is selected by the previous greedy algorithm. In this way, we obtain the desired phase sequence for WiFi emulation.

After the phase sequence generation and the phase sequence optimization, we select an appropriate phase sequence with relatively small emulation errors. WIDE constructs the payload of WiFi sender to emulate the phase sequence and realizes the physical-level CTC from WiFi to ZigBee.

4.4 Parallel Communication

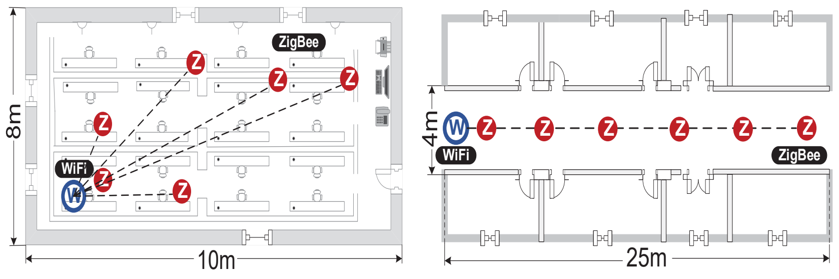
As we all know, a WiFi channel overlaps with several ZigBee channels. WIDE can support parallel CTC from WiFi to ZigBee with the channel mapping scheme. The central frequency of a WiFi channel is set as 2440MHz, the two regions of WiFi subcarriers [-13 to -19] and [13 to 19] can be utilized to achieve two parallel CTC with standard ZigBee channel 17 and channel 19 as shown in Fig. 16. We note many commodity WiFi radios (e.g., Atheros AR9485, AR5112, and AR2425) can set their central frequency.

5 EVALUATION

In this section, we conduct extensive experiments to evaluate the performance of WIDE. We compare WIDE with WEBee, the representative physical-level CTC from WiFi to ZigBee. WIDE can be implemented directly with commodity devices. The USRP N210 devices are used only for evaluation purpose to measure low-level PHY information, such as Hamming distance and symbol error rate (SER).

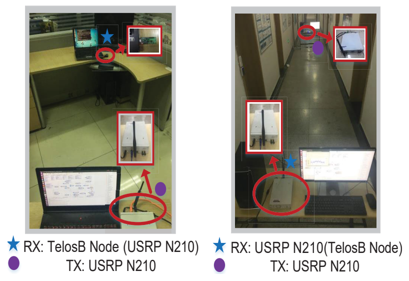
5.1 Experiment Setup

The WIDE transmitter is a USRP N210 device with 802.11 b/g PHY. The WIDE receivers is a USRP N210 device with 802.15.4 PHY. During experiments,each emulated ZigBee packet consists of four bytes of preamble (0x00000000), a byte of start of frame delimiter (SFD) (0xA7), two bytes of packet length, variable bytes payload. For fair comparison, the composition of WEBee packet is the same as that of WIDE. We set the ZigBee channel at 19 and set the central frequency of the WiFi channel at 2440MHz, which can be realized by many commodity WiFi devices (e.g. Atheros AR9485, AR5112, and AR2425). Our evaluation include symbol error rate (SER), packet reception ratio (PRR), and goodput. To ensure statistical validity, we obtain the average result of 10 experiments, each of which sends 1,000 WIDE packets under a wide range of settings including indoor/hallway, short/long distance, and mobile scenarios.



(a) Lab (b) Hallway

Figure 17: The floor plans of the lab and the hallway



(a) Lab (b) Hallway

Figure 18: Experiment settings in the lab and the hallway

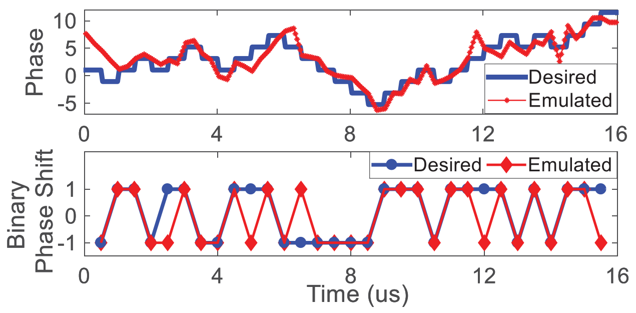


Figure 19: Emulated phase sequence

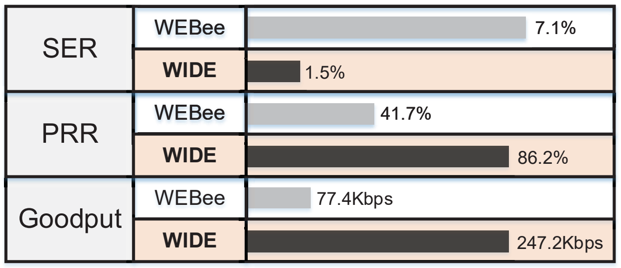


Figure 20: Overall performance comparison

5.2 Emulated Phase Sequence

First, we observe the desired phase sequence and emulated phase sequence to verify the feasibility of digital emulation. As shown in Fig. 19, we can find that the emulated phase sequence resembles the desired phase sequence with limited distortions. The decoded binary phase shift of the emulated phase sequence only has four wrong bits (except the first and the last bits). The limited number of error bits can be tolerated by the mechanism of ZigBee DSSS decoding. This emulated phase sequence can be decoded successfully. Therefore, the digital emulation method is feasible for the physical-level CTC.

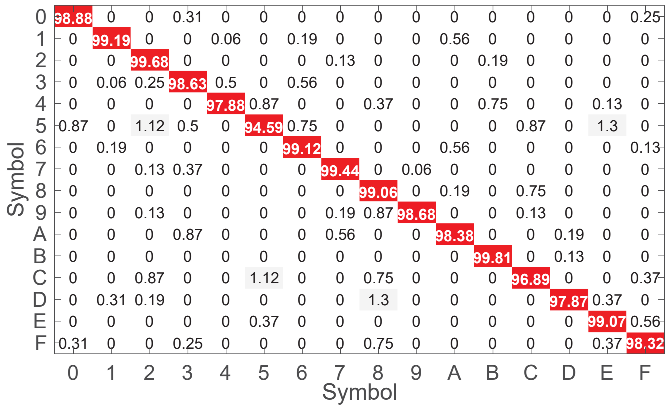
5.3 Overall Performance Comparison

We conduct experiments to compare the overall performance of WIDE and WEBee in practice. The distance between the WiFi sender and the ZigBee receiver is 4m. The ZigBee payload is 16 bytes and includes all 16 different symbols. The experiments are conducted in our lab as shown in Fig. 18(a), with a consistent ambient environment and a similar network interference condition.

The comparison results are shown in Fig. 20. First, the SER of WEBee is 7.1% and the SER of WIDE is 1.5%. The reason is that the WiFi emulation result of the ladder shaped waveform is better than the half sine waveform, which reduces the symbol decoding errors. We further evaluate the decoding accuracy for all different ZigBee symbols and the evaluation results are shown in Fig. 21. The average decoding accuracy of different symbols varies from 94.59% to 99.81%. Because the phase sequence of each ZigBee symbol is different, the emulation result of WiFi is also different. Furthermore, Fig. 22 shows the Hamming Distance between the decoded binary phase shift sequence and the predefined binary phase shift sequence when we adopt WEBee and WIDE. For all ZigBee symbols, the Hamming Distance of WIDE is much lower than WEBee. The commercial ZigBee device sets a threshold to tolerate a certain number of chip errors, which by default is 12. This threshold is relaxed to 20 in WEBee, while WIDE has no need to modify this threshold.

At the same time, the PRR of WIDE can be up to 86.1%, while the PRR of WEBee is 41.7%. This is also because that the better emulation result improves the possibility of preamble detection and header synchronization for the ZigBee signals. In addition, the PRR of both WEBee and WIDE is also related to the number of packet transmissions, a parameter to trade off between throughput and reliability. Fig. 24 illustrates the PRR under different transmission numbers. The PRR of WIDE exceeds 80% when the CTC packets are sent at a rate of 250Kbps (only 1 transmission), while WEBee achieves the 80% PRR when the CTC packets are sent at a rate of 83.3Kbps (3 transmissions).

WIDE and WEBee are both the physical-level CTC methods, so the theoretical throughput of WIDE and WEBee can both be the ceiling speed of standard ZigBee communication. Due to the different SER and PRR of ZigBee decoding, the goodputs of WEBee and WIDE are 77.4Kbps and 247.2Kbps, respectively. It is worth noting that the performance of WEBee realized by our evaluation is worse than [21], this is because we don't adopt repeated preamble



Symbol	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	98.88	0	0	0.31	0	0	0	0	0	0	0	0	0	0	0	0.25
1	0	99.19	0	0	0.06	0	0.19	0	0	0	0.56	0	0	0	0	0
2	0	0	99.68	0	0	0	0	0.13	0	0	0	0.19	0	0	0	0
3	0	0.06	0.25	98.63	0.5	0	0.56	0	0	0	0	0	0	0	0	0
4	0	0	0	0	97.88	0.87	0	0	0.37	0	0	0.75	0	0	0.13	0
5	0.87	0	1.12	0.5	0	94.59	0.75	0	0	0	0	0	0.87	0	1.3	0
6	0	0.19	0	0	0	0	99.12	0	0	0	0.56	0	0	0	0	0.13
7	0	0	0.13	0.37	0	0	0	99.44	0	0.06	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	99.06	0	0.19	0	0.75	0	0	0
9	0	0	0.13	0	0	0	0	0.19	0.87	98.68	0	0	0.13	0	0	0
A	0	0	0	0.87	0	0	0	0.56	0	0	98.38	0	0	0.19	0	0
B	0	0	0	0	0	0	0	0	0	0	0	99.81	0	0.13	0	0
C	0	0	0.87	0	0	1.12	0	0	0.75	0	0	0	96.89	0	0	0.37
D	0	0.31	0.19	0	0	0	0	0	1.3	0	0	0	0	97.87	0.37	0
E	0	0	0	0	0	0.37	0	0	0	0	0	0	0	0	99.07	0.56
F	0.31	0	0	0.25	0	0	0	0	0.75	0	0	0	0	0	0.37	98.32

Figure 21: Decoding accuracy for different symbols

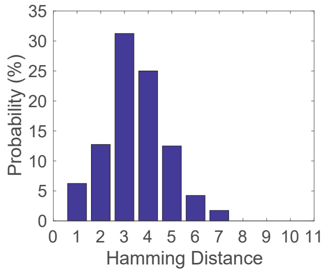


Figure 22: Hamming Distance of WIDE symbol

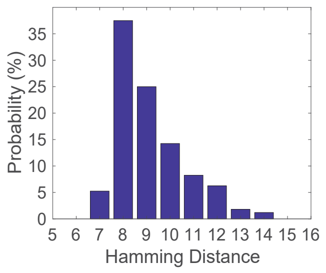


Figure 23: Hamming Distance of WEBee symboln

protection and link coding. Whereas, it doesn't affect the comparison of WIDE and WEBee when we conduct all the experiments under the same settings.

5.4 WIDE Performance under Different Settings

In this subsection, we vary the ZigBee payload length, the distance between the sender and the receiver, and operating environments to study their impacts on WIDE in terms of SER and PRR. We also evaluate WIDE in mobile scenarios.

5.4.1 Impact of ZigBee payload length. We study the impact of payload length on WIDE and WEBee. We change the payload length of ZigBee from 8 bytes to 24 bytes. The distance between the WiFi sender and the ZigBee receiver is 4m. Fig. 25 and Fig. 26 show the evaluation results of SER and PRR respectively. We can find that the SER increases with the increase of payload length since that the longer payload brings more accumulated errors. When the payload length is 8 bytes, the SER of WIDE and WEBee is 1.21% and 5.41% respectively. When the payload length increases to 24 bytes, the SER of WIDE and WEBee increases to 2.17% and 9.14% respectively. Whereas, the SER of WIDE is still much lower than that of WEBee.

The PRR of ZigBee packets relies on the preamble detection, header synchronization, and the CRC result. So a single symbol error may lead to a packet loss. When a packet has variable length payload, the longer payload it has, the easier to lose packets. Fig.

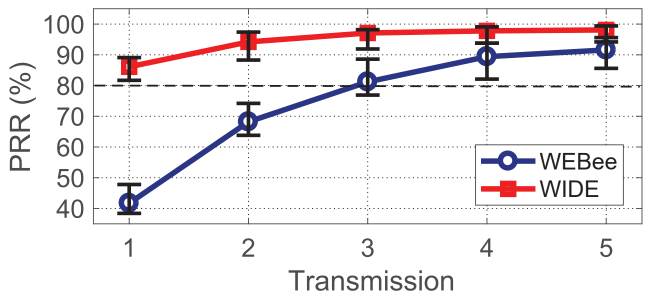


Figure 24: PRR with the times of transmission

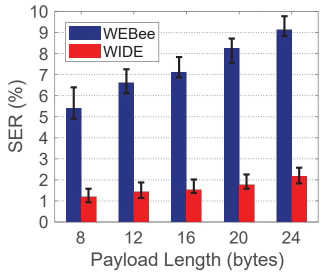


Figure 25: SER with different ZigBee payload lengths

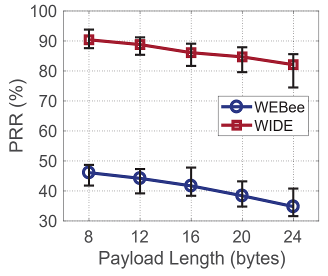


Figure 26: PRR with different ZigBee payload lengths

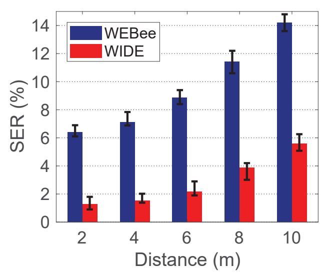


Figure 27: SER with different distances

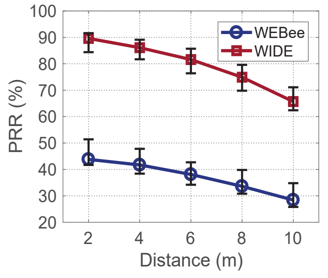


Figure 28: PRR with different distances

26 shows the PRR with the variation of payload length. We can find that the PRR of WEBee decreases sharply with the increase of payload length. When the payload length is 24 bytes, the PRR of WEBee decreases to 34.1%. Whereas, the PRR of WIDE only decreases to 82.1%.

The results reveal that the payload length has an influence on the performance of WIDE and WEBee, but WIDE can still achieve relatively reliable performance when increasing the payload length.

5.4.2 Impact of distance. We then study the impact of distance between the WiFi sender and the ZigBee receiver. We conduct this experiment in the lab and vary the distance from 1m to 10m, as shown in Fig.17(a). The ZigBee payload length is 16 bytes.

Fig. 27 shows the SER with the variation of distance. We can find that the SER increases with the increase of distance. When the distance is 2m, the SER of WIDE and WEBee is 1.28% and 6.41% respectively. When the distance increases to 10m, the SER of WIDE

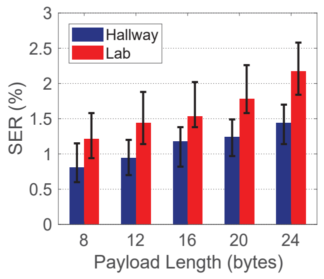


Figure 29: SER (hallway) with different ZigBee payload lengths

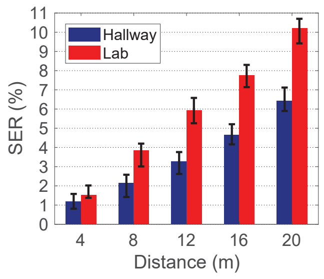


Figure 30: SER (hallway) with different distances

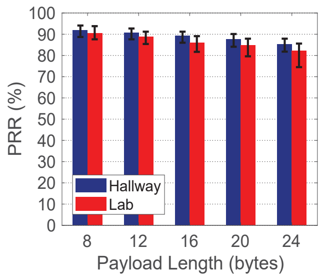


Figure 31: PRR (hallway) with different ZigBee payload lengths

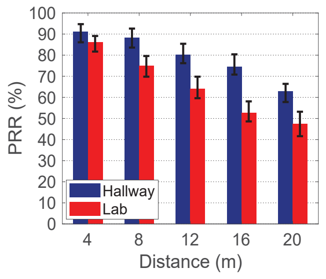


Figure 32: PRR (hallway) with different distances

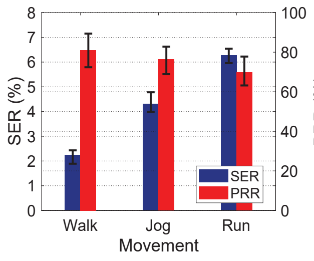


Figure 33: WIDE performance under mobility

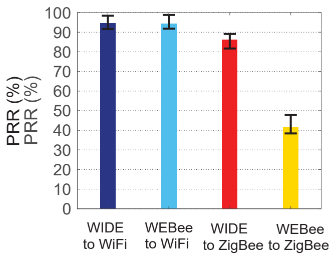


Figure 34: PRR of WiFi and ZigBee

and WEBee increases to 5.58% and 15.24% respectively. Because the longer the distance, the lower the SNR. This results in the worse attenuation of the signal amplitude and the phase distortion. Whereas, the SER of WIDE is still more stable and lower than that of WEBee, which is due to the good emulation for the ladder shaped waveform.

The PRR with the variation of distance is shown in Fig. 28. We can find that the PRR of WEBee decreases sharply with the increase of distance. When the distance is 10m, the PRR of WEBee decreases to 28.4%. The PRR of WIDE decreases to 65.7%. With the increase of distance, the success rate of ZigBee preamble detection and header synchronization will decrease. In addition, the increase of the SER also makes CRC easier to fail.

The results reveal that the distance has an influence on the performance of WIDE and WEBee, but WIDE can still achieve relatively reliable performance when increasing the distance. In addition, we want to clarify that WIDE is not restricted in use at short range. Indeed, WIDE doesn't affect the communication distance of the WiFi device and the ZigBee device. The limited communication range in the experiments is just because of the limited space of the real lab, where we carried out the experiments.

5.4.3 Impact of environment. We evaluate WIDE in different environments. We conduct the experiments to compare the SER and PRR of WIDE in the lab and the hallway as shown in Fig. 18(a) and Fig. 18(b). Fig. 29 and Fig. 30 present the SER of WIDE in two environments. We can find that the SER of WIDE in the hallway is lower than the SER of WIDE in the lab. This is because the environment in the lab is more complicated than that in the hallway, which leads to more serious multipath influences on the received signals and results in higher decoding errors. We can also find that the SER increases with the increase of payload length. For example, when the payload length is 24 bytes, the SER of WIDE in the lab and hallway is 2.17% and 1.44% respectively. Similarly, the SER increases with the increase of distance. For example, when the distance is 10m, the SER of WIDE in the lab and hallway is 10.24% and 6.42% respectively.

Fig. 31 and Fig. 32 present the PRR of WIDE in two environments. We can find that the PRR of WIDE in the hallway is higher than the PRR of WIDE in the lab. This is because the environment of hallway is cleaner and the SNR is higher, which improves the possibility of preamble detection and header synchronization for the ZigBee signals. We can also find that the PRR decreases with the increase of payload length. For example, when the payload length is 24 bytes, the PRR of WIDE in the lab and hallway is 82.1% and 85.2% respectively. Similarly, the PRR decreases with the increase of distance. For example, when the distance is 20m, the PRR of WIDE in the lab and hallway is 47.4% and 62.8% respectively.

5.4.4 Impact of mobility. We also evaluate the performance of WIDE under mobility. In the experiments, the WIDE sender transmits packets to emulate ZigBee signals. The ZigBee payload length is 16 bytes. A volunteer carrying the ZigBee receiver walks, jogs, and runs at a speed of 1 m/s, 2 m/s, and 4 m/s, respectively. Fig. 33 shows the SER and PRR of WIDE with varying speeds. The SER increases and the PRR decreases with the increase of the speed. When the moving speed is 4m/s, the SER and the PRR of WIDE is 6.28% and 68.9%, which is still acceptable.

5.5 Impact on WiFi Reception

WIDE leverages the WiFi payload to accomplish the digital emulation, but WIDE doesn't modify the preamble or header of the WiFi standard. Therefore, the WIDE sender indeed transmits a WiFi packet, which can be received by the WiFi receiver. In this section, we conduct experiments to study the impact of WIDE on the WiFi receptions. The WiFi sender transmits packets, with the payload emulating a ZigBee packets with the payload length of 16 bytes. The distance between the WiFi sender and the WiFi receiver is 4m. We measure the PRR of the WiFi and ZigBee. Experimental results are shown in Fig. 34. We can find that in both WIDE and

WEBee, the PRR of WiFi is similar. The results validate that WIDE has negligible influence on WiFi receptions.

6 CONCLUSION

In this paper, we propose WIDE, a physical-level CTC from WiFi to ZigBee via digital emulation. Instead of emulating the standard ZigBee half sine waveform, WIDE selects an appropriate non-standard waveform to emulate ZigBee binary phase shift sequence, which is the essence of ZigBee decoding. Compared with analog emulation, digital emulation can adjust the phase sequence at the same time of satisfying the requirement of the binary phase shift sequence, which offers flexibility to achieve a more reliable CTC. We conduct extensive experiments to evaluate the performance of WIDE. The results show that WIDE significantly improves the Packet Reception Ratio (PRR) from 41.7% to 86.2%, which is 2× of WEBee's, an existing representative physical-level CTC. To the best of our knowledge, WIDE is the first work that leverages digital emulation to achieve physical-level CTC. Without loss of generality, the method of digital emulation is generic and applicable to a set of CTCs, where the transmitter has a wider bandwidth for emulation and the receiver decoding is based on the phase shift.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers and the shepherd for their valuable comments and helpful suggestions. This work was supported by National Key R&D Program of China No. 2017YFB1003000 and National Nature Science Foundation of China No. 61772306.

REFERENCES

[1] Z. An, L. Yang, and Q. Lin. Cross-frequency communication: Near-field identification of uhf rfids with wifi. In *MobiCom*. ACM, 2018.

[2] Y. Chae, S. Wang, and S. M. Kim. Exploiting wifi guard band for safeguarded zigbee. In *Sensys*. ACM, 2018.

[3] K. Chebrolu and A. Dhekne. Esense: communication through energy sensing. In *MobiCom*. ACM, 2009.

[4] G. Chen and W. Dong. Jamcloak: Reactive jamming attack over cross-technology communication links. In *ICNP*. IEEE, 2018.

[5] Y. Chen, Z. Li, and T. He. Twinbee: Relable physical-layer cross-technology communication with symbol-level coding. In *INFOCOM*. IEEE, 2018.

[6] Z. Chi, Z. Huang, Y. Yao, T. Xie, H. Sun, and T. Zhu. Emf: Embedding multiple flows of information in existing traffic for concurrent communication among heterogeneous iot devices. In *INFOCOM*. IEEE, 2017.

[7] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu. B2w2: N-way concurrent communication for iot devices. In *SenSys*. ACM, 2016.

[8] Z. Chi, Y. Li, Y. Yao, and T. Zhu. Pmc: Parallel multi-protocol communication to heterogeneous iot radios within a single wifi channel. In *ICNP*. IEEE, 2017.

[9] J. Dhivvya, S. N. Rao, and S. Simi. Towards maximizing throughput and coverage of a novel heterogeneous maritime communication network. In *MobiHoc*. ACM, 2017.

[10] S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the rf smog: Making 802.11 robust to cross-technology interference. In *SIGCOMM*. ACM, 2011.

[11] X. Guo, Y. He, X. Zheng, L. Yu, and O. Gnawali. Zigfi: Harnessing channel state information for cross-technology communication. In *INFOCOM*. IEEE, 2018.

[12] X. Guo, Y. He, X. Zheng, Z. Yu, and Y. Liu. Lego-fi: Transmitter-transparent ctc with cross-demapping. In *INFOCOM*. IEEE, 2019.

[13] X. Guo, X. Zheng, and Y. He. Wizig: Cross-technology energy communication over a noisy channel. In *INFOCOM*. IEEE, 2017.

[14] H. Iqbal, M. H. Alizai, I. A. Qazi, O. Landsiedel, and Z. A. Uzmi. Scylla: interleaving multiple iot stacks on a single radio. In *CoNext*. ACM, 2018.

[15] W. Jiang, S. M. Kim, Z. Li, and T. He. Achieving receiver-side cross-technology communication with cross-decoding. In *MobiCom*. ACM, 2018.

[16] W. Jiang, Z. Yin, S. M. Kim, and T. He. Transparent cross-technology communication over data traffic. In *INFOCOM*. IEEE, 2017.

[17] M. Jin, Y. He, X. Zheng, Y. He, D. Fang, D. Xu, T. Xing, and X. Chen. Smoggy-link: Fingerprinting interference for predictable wireless concurrency. In *ICNP*. IEEE, 2016.

[18] S. M. Kim and T. He. Freebee: Cross-technology communication via free side-channel. In *MobiCom*. ACM, 2015.

[19] Y. Li, Z. Chi, X. Liu, and T. Zhu. Chiron: Concurrent high throughput communication for iot devices. In *MobiSys*. ACM, 2018.

[20] Y. Li, Z. Chi, X. Liu, and T. Zhu. Passivezigbee: Enabling zigbee transmissions using wifi. In *Sensys*. ACM, 2018.

[21] Z. Li and T. He. Webee: Physical-layer cross-technology communication via emulation. In *MobiCom*. ACM, 2017.

[22] Z. Li and T. He. Longbee: Enabling long-range cross-technology communication. In *INFOCOM*. IEEE, 2018.

[23] Z. Li, W. Jiang, and T. He. Bluebee: Physical-layer cross-technology communication via emulation. In *SenSys*. ACM, 2017.

[24] Z. Li, Y. Xie, M. Li, and K. Jamieson. Recitation: Rehearsing wireless packet reception in software. In *MobiCom*. ACM, 2015.

[25] C. J. M. Liang, K. Chen, N. B. Priyantha, J. Liu, and F. Zhao. Rushnet: practical traffic prioritization for saturated wireless sensor networks. In *SenSys*. ACM, 2014.

[26] A. Saifullah, M. Rahman, D. Ismail, C. Lu, R. Chandra, and J. Liu. Snow: Sensor network over white spaces. In *SenSys*. ACM, 2016.

[27] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu. Stpp: Spatial-temporal phase profiling-based method for relative rfid tag localization. *IEEE/ACM Transactions on Networking*, 25(1):596–609, 2017.

[28] R. K. Sheshadri, K. Sundaresan, E. Chai, A. Khojastepour, S. Rangarajan, and D. Koutsonikolas. Blu: Blue-printing interference for robust lte access in unlicensed spectrum. In *CoNEXT*. ACM, 2017.

[29] R. K. Sheshadri, K. Sundaresan, E. Chai, S. Rangarajan, and D. Koutsonikolas. Eli: Empowering lte with interference awareness in unlicensed spectrum. In *ICNP*. IEEE, 2018.

[30] 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 *MobiHoc*. ACM, 2015.

[31] S. Wang, Z. Yin, Z. Li, and T. He. Networking support for physical-layer cross-technology communication. In *ICNP*. IEEE, 2018.

[32] Y. Yan, P. Yang, X. Li, T. Yue, L. Zhang, and L. You. Zimo: building cross-technology mimo to harmonize zigbee smog with wifi flash without intervention. In *MobiCom*. ACM, 2013.

[33] Z. Yin, W. Jiang, S. M. Kim, and T. He. C-morse: Cross-technology communication with transparent morse coding. In *INFOCOM*. IEEE, 2017.

[34] Z. Yin, Z. Li, S. M. Kim, and T. He. Explicit channel coordination via cross-technology communication. In *MobiSys*. ACM, 2018.

[35] Z. Yu, C. Jiang, Y. He, X. Zheng, and X. Guo. Crocs: Cross-technology clock synchronization for wifi and zigbee. In *EWSN*. ACM, 2018.

[36] W. Zeng, A. Arora, and K. Srinivasan. Low power counting via collaborative wireless communications. In *IPSN*. ACM/IEEE, 2013.

[37] X. Zhang and G. S. Kang. Enabling coexistence of heterogeneous wireless systems:case for zigbee and wifi. In *MobiHoc*. ACM, 2011.

[38] Y. Zhang and Q. Li. Howies: A holistic approach to zigbee assisted wifi energy savings in mobile devices. In *INFOCOM*. IEEE, 2013.

[39] Z. Zhao, W. Dong, G. Chen, G. Min, T. Gu, and J. Bu. Embracing corruption burstiness: Fast error recovery for zigbee under wi-fi interference. *IEEE Transactions on Mobile Computing*, 16(9):2518–2530, 2017.

[40] X. Zheng, Z. Cao, J. Wang, Y. He, and Y. Liu. Zisense: towards interference resilient duty cycling in wireless sensor networks. In *SenSys*. ACM, 2014.

[41] X. Zheng, Y. He, and X. Guo. Stripcomm: Interference-resilient cross-technology communication in coexisting environments. In *INFOCOM*. IEEE, 2018.

[42] X. Zheng, J. Li, H. Gao, and Z. Cai. Capacity of wireless networks with multiple types of multicast sessions. In *MobiHoc*. ACM, 2014.

[43] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma. Zifi: wireless lan discovery via zigbee interference signatures. In *MobiCom*. ACM, 2010.