mZig: Enabling Multi-Packet Reception in ZigBee

Linghe Kong[1,2]
[1]McGill University, Canada
[2]Shanghai Jiao Tong University, China
linghe.kong@mail.mcgill.ca

Xue Liu
McGill University, Canada
xueliu@cs.mcgill.ca

ABSTRACT

This paper presents mZig, a novel physical layer design that enables a receiver to simultaneously decode multiple packets from different transmitters in ZigBee. As a low-power and low-cost wireless protocol, the promising ZigBee has been widely used in sensor networks, cyber-physical systems, and smart buildings. Since ZigBee based networks usually adopt tree or cluster topology, the convergecast scenarios are common in which multiple transmitters need to send packets to one receiver. For example, in a smart home, all appliances report data to one control plane via ZigBee. However, concurrent transmissions in convergecast lead to the severe collision problem. The conventional ZigBee avoids collisions using backoff time, which introduces additional time overhead. Advanced methods resolve collisions instead of avoidance, in which the state-of-the-art ZigZag resolves one m-packet collision requiring m retransmissions. We propose mZig to resolve one m-packet collision by this collision itself, so the theoretical throughput is improved m-fold. Leveraging the unique features in ZigBee's physical layer including its chip rate, half-sine pulse shaping and O-QPSK modulation, mZig subtly decomposes multiple packets from one collision in baseband signal processing. The practical factors of noise, multipath, and frequency offset are taken into account in mZig design. We implement mZig on USRPs and establish a seven-node testbed. Experiment results demonstrate that mZig can receive up to four concurrent packets in our testbed. The throughput of mZig is 4.5x of the conventional ZigBee and 3.2x of ZigZag in the convergecast with four or more transmitters.

Categories and Subject Descriptors

C.2.1 [**Computer-Communications Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Design, Experimentation, Performance

1. INTRODUCTION

Based on IEEE 802.15.4, ZigBee [2] is a competitive wireless technology that has draw extensive interests by academia and industry. Different from high-power and high-bitrate WiFi [50] or 4G/5G [29], ZigBee focuses on the field of low-power, low-cost, and low-bitrate communications, which has been widely used in sensor networks [41, 51], cyber-physical systems [48], and smart buildings [38]. In 2013, the new version ZigBee, ZigBee Smart Energy V2 [3], was published. The number of smart devices equipped with ZigBee communication modules is poised to increase dramatically.

ZigBee based networks usually adopt tree or cluster topology [19, 26], in which multiple transmitters (TXs) need to send packets to one receiver (RX), known as convergecast [13, 42]. The convergecast is fundamental for plenty of applications. For example, in a smart home, hundreds of sensors and appliances report data to one control plane via a ZigBee [38]. Some other convergecast examples in recent studies include: data collection [20], neighbor discovery [8], ACK for multicast [9], and link correlation estimation [41].

In such convergecast scenarios, there exist severe collisions due to multiple concurrent transmissions. In ZigBee products, carrier sense multiple access (CSMA) [31, 35] is the conventional solution to avoid collisions, which exploits backoff time to divide transmissions into different time slots. The main drawback of CSMA is to introduce additional time overhead. Moreover, CSMA fails to avoid collisions in the case of hidden terminals [40]. The advanced methods such as interference cancellation [17] and constructive interference [10] resolve collisions instead of avoiding them. In the collision resolution category, the state-of-the-art method is ZigZag, which leverages time offsets among collisions to separate multiple packets. Nevertheless, to separate an m-packet collision, ZigZag requires m retransmissions to form collisions with different offsets. Thus, the upper bound of ZigZag's throughput is equal to that of one-TX one-RX communication without collisions.

To further improve the throughput and address the collision problem better in convergecast, we propose mZig to enable multi-packet reception (MPR) in ZigBee. The goal of mZig is to decompose m concurrent packets from one collision directly. Hence, the theoretical throughput of mZig is m-fold than the current best ZigZag. It is very challenging to achieve this amazing result in practice. The secret of mZig is from the physical layer of ZigBee.

In the physical layer, ZigBee shows three unique features on its chip, where one chip is the smallest unit carrying information in ZigBee's baseband signal. (i) Every chip is oversampled by multiple samples, because the sampling rate of off-the-shelf ADCs is much higher than the chip rate. (ii) The waveform of a chip is known, because ZigBee adopts half-sine pulse shaping on every chip. (iii) The amplitude of every chip is nearly the same in one packet, because O-QPSK modulates chips by different phase but not amplitude. These features provide opportunities to develop mZig.

To see how mZig works, consider the convergecast scenario

MobiCom'15, September 07 - 11, 2015, Paris, France
 ISBN 978-1-4503-3619-2/15/09... $15.00
DOI: http://dx.doi.org/10.1145/2789168.2790104

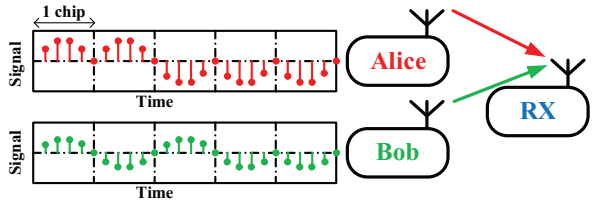

Figure 1: A convergecast scenario.

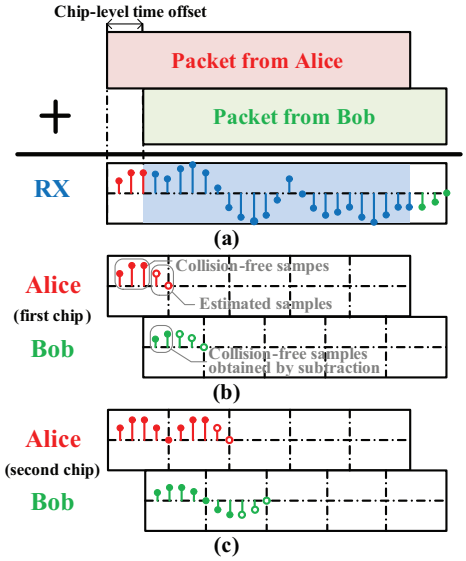

Figure 2: An example of decomposing a two-packet collision with chip-level time offset.

in Fig. 1, where Alice and Bob send packets simultaneously to RX, causing a collision. Alice sends five chips '11000' and Bob sends '10100', respectively. According to ZigBee, a chip '1' is a positive half-sine shaping, and a chip '0' is a negative half-sine. In this example, every chip has five samples. The baseband signals of these two packets are shown in Fig. 1.

When two packets arrive at RX with chip-level time offset, mZig decomposes two packets leveraging the features of oversampling and known shaping. An example of collided packet at RX is shown in Fig. 2(a), where two packets cannot be distinguished because their baseband signals are overlapped. When the time offset can be detected, we find the first three samples are collision-free. Using these three samples and the known half-sine shaping, we can estimate the next two samples and form the first chip of Alice as shown in Fig. 2(b). Then, we subtract this estimated chip from the collided packet. Another two collision-free samples are obtained, which can be used to estimate the first chip for Bob. Repeating the operations of subtraction and estimation as shown in Fig. 2(c), mZig decomposes the collided packet into two packets chip-by-chip.

When two packets arrive at RX synchronously, mZig decomposes packets leveraging the amplitude difference. The collided packet is shown in Fig. 3, where all chips are overlapped without time offset, so the above method using collision-free samples cannot be applied. If there is only one TX, the received signal will have two amplitude levels for chip '1' and '0', respectively. Furthermore, if there are m TXs, there will be 2^m levels of amplitude combinations. In Fig. 3, four different amplitude levels can be counted, so we can estimate that there are two packets in this collision. Denote these four amplitude levels by L_i, where $L_1 > L_2 > L_3 > L_4$.

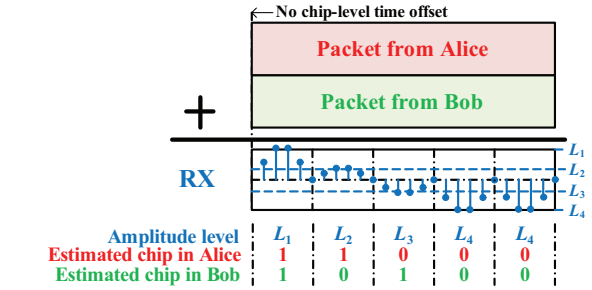

Figure 3: An example of decomposing a two-packet collision without chip-level time offset.

In addition, denote the amplitudes of Alice and Bob by α and β, respectively. Assume $\alpha > \beta$, we can build a mapping relationship that $L_1 = \alpha + \beta$, $L_2 = \alpha - \beta$, $L_3 = -\alpha + \beta$, and $L_4 = -\alpha - \beta$. Based on such relationships, every chip can be decomposed by its amplitude. For example, the third chip at RX reaches L_3, so the third chip for Alice is '0' and the third chip for Bob is '1' according to $L_3 = -\alpha + \beta$.

The main contributions of this paper are as follows:

- We design mZig, a novel physical layer technique on receiver to enable multi-packet reception in ZigBee. The core design of mZig leverages the unique features of ZigBee to decompose multiple packets from one collision chip-by-chip. To enhance the performance of mZig, we carefully address a series of practical issues including noise, multipath, and frequency offset. We theoretically derive the maximal value of m as $\lfloor \frac{S}{2 \times C} \rfloor$, where m is the number of concurrent transmissions, S is the sampling rate, and C is the chip rate.
- We implement mZig on USRPs, and build a seven-node testbed. Experiment results demonstrate that mZig can receive up to four concurrent transmissions with low bit error using 32Msps sampling rate. The throughput of mZig achieves 4.5x of the conventional ZigBee and 3.2x of ZigZag in the convergecast with four or more transmitters. More simulations are conducted to reveal the impacts of chip-level time offset, noises, multipath, and frequency offset, which are uncontrollable parameters in experiments. In simulation, we also transplants mZig on Bluetooth Low Energy (BLE), another key protocol in low-power communications.

The proposed mZig can decompose multi-packet collision without additional requirements such as synchronization or packet length. However, mZig is not omni-directional. For example, it fails when low SNR. In such a case, since the design of mZig is orthogonal to existing collision resolution techniques such as SIC [32], ZigZag [15], and full duplex [21], they can work complementarily for addressing collisions.

2. PRELIMINARY

Since mZig is a physical layer design, we firstly review the physical layer of conventional ZigBee in this section. Then, we summarize the unique features of ZigBee, which provide the opportunity to develop mZig.

2.1 Physical Layer of conventional ZigBee

The ZigBee standard [2] specifies its operation in 2.4GHz (worldwide), 915MHz (America) and 868MHz (Europe) ISM bands. This paper focuses on the most widely used 2.4GHz ZigBee. Sixteen channels are allocated to ZigBee in this band, where the bandwidth of each channel is 2MHz. The bit rate is 250kbps. The block diagram of physical layer in conventional ZigBee is illustrated in Fig. 4, including one

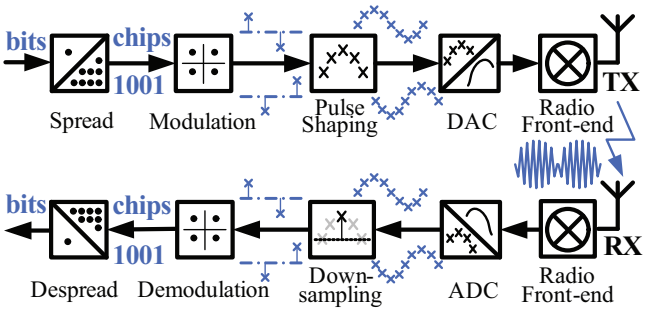

Figure 4: The block diagram of physical layer in conventional ZigBee.

Table 1: Bit to chip spreading

Bits: $b_0b_1b_2b_3$	**Chips**: $c_0c_1c_2\cdots c_{31}$
0000	11011001110000110101001000101110
1000	11101101100111000011010100100010
0100	00101110110110011100001101010010
⋮	⋮
1111	11001001011000000111011110111000

transmitter (TX) and one receiver (RX).

In the physical layer, a TX sends a packet through five steps: spread, modulation, pulse shaping, digital-analog conversion (DAC), and radio front-end.

First, ZigBee spreads bits into chips by direct-sequence spread spectrum (DSSS). According to the given bit-to-chip spreading relationship as shown in Tab. 1, every four bits are spread to the specified 32 chips. A chip is the smallest unit carrying information in ZigBee. The chip rate is $C = 2$Mchip/s, which is equal to the bandwidth.

Second, ZigBee modulates chips onto I-Q phases using offset quadrature phase-shift keying (O-QPSK). As shown in Fig. 5, the chips $c_0, c_2, \cdots$ are modulated onto I phase and the chips $c_1, c_3, \cdots$ are modulated onto Q phase one-by-one. The chip rate of each phase is $C/2 = 1$Mchip/s. Hence, the duration of each chip is $\frac{1\text{s}}{1\text{Mchip/s}} = 1\mu$s. The 'O' in O-QPSK expresses that a half chip time offset, *i.e.*, $1\mu\text{s}/2 = 0.5\mu$s, exists between I phase and Q phase.

Third, ZigBee adopts the half-sine pulse shaping to shape a chip into baseband samples. These samples form a digital waveform. A chip '1' is shaped to a positive half-sine and a chip '0' is shaped to a negative half-sine as shown in Fig. 6. The duration of a chip's waveform keeps 1μs. In one packet, the amplitudes of all chips are the same, which depends on the selected transmission power.

Fourth, DAC converts the digital baseband waveform into the analog baseband waveform.

Fifth, the radio front-end up-converts the baseband waveform to 2.4GHz carrier and sends it out.

The physical layer of RX is nearly the inverse of TX as shown in Fig. 4. To receive a packet, the RX down-converts the received signal to an analog baseband waveform, converts the analog waveform to a digital one, down-samples and demodulates the digital waveform into chips, and despreads chips into bits. For ease of understanding, we only draw the major modules of RX in Fig. 4. More detailed RX modules such as phase tracking can refer to [2].

2.2 Features of ZigBee Chips

From the conventional ZigBee, we observe that there are three unique features in the chips.

• **Oversampling**. Since the sampling rate of recent analog-digital conversions (ADCs) is usually higher than ZigBee's chip rate, a chip at RX is sampled by multiple samples. For instance, USRP B210 [1] has a 61.44Msps ADC, so every chip has no less than $\lfloor \frac{61.44\text{Msps}}{1\text{Mchip/s}} \rfloor = 61$ samples.

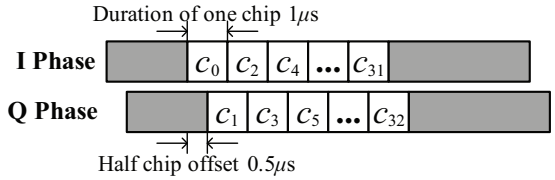

Figure 5: O-QPSK modulation.

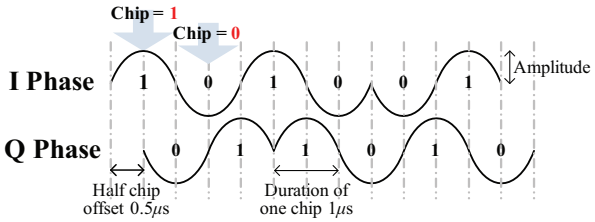

Figure 6: Half-sine pulse shaping.

• **Known shaping**. As all chips are shaped by half-sine at TX, the shapes of received chips are known. Although these chips are interfered by noises in wireless channels, their basic shapes are maintained.

• **Uniform amplitude**. Unlike ASK or QAM [5], which operates the modulation by different levels of amplitude, ZigBee adopts O-QPSK modulation at TX. As a result, the amplitudes of all chips in one packet are equivalent.

In the following, we will show how to leverage these features to design mZig.

3. CORE DESIGN OF MZIG

The multi-packet collision primer is presented in this section, and collisions are classified into two categories: collisions with or without chip-level time offset. Then, the core designs of mZig are introduced to resolve the collisions in these two categories respectively.

3.1 Multi-Packet Collision Primer

A wireless signal is typically represented as discrete complex values in baseband [37]. As introduced in §2.1, chips in ZigBee are shaped to be a sequence of samples in baseband. Denote $X[n]$ to be the complex value of the n-th baseband sample at TX. The received signal can also be represented as a sequence of samples spaced by the sampling interval $\Delta = 1/S$, where S is the sample rate. If the transmitted sample is $X[n]$, the received sample is formulated by

$$Y[n] = HX[n] + W[n], \tag{1}$$

where $H = he^{\gamma}$ is the channel parameter, whose magnitude h refers to the channel attenuation and its angle γ is a phase shift depending on the distance between the TX and the RX, and $W[n]$ is the noise.

If two TXs Alice and Bob transmit concurrently, the signals of their packets are added up in air. Then, the received sample can be expressed as:

$$Y[n] = H_A X_A[n] + H_B X_B[n] + W[n], \tag{2}$$

where H_A and H_B are the channel parameters of Alice and Bob, respectively. $X_A[n]$ and $X_B[n]$ are the transmitted samples of Alice and Bob.

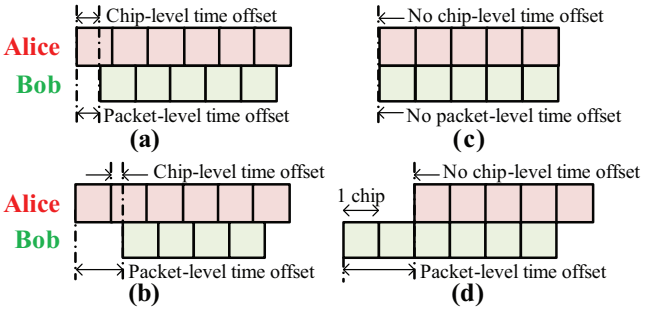

Figure 7: Examples of collisions with or without chip-level time offset.

In the one-TX one-RX case, since the channel parameter H can be estimated [45], the RX is able to resolve $X[n]$ using the received $Y[n]$ and the given Eq. (1), as long as the signal-to-noise ratio (SNR) is large enough. Thus, the transmitted packet is successfully decoded.

In contrast, in the two-TX one-RX case, though H_A, H_B, and $Y[n]$ are known in Eq. (2), two unknown variables $X_A[n]$ and $X_B[n]$ cannot be resolved simultaneously with only one equation. Hence, conventional wireless systems cannot deal with multi-packet collision. To resolve $X_A[n]$ and $X_B[n]$, it is necessary to explore other available information.

3.2 Two Categories of Multi-Packet Collision

In this paper, collisions are classified into two categories according to the chip-level time offset (CTO) among multiple packets in a collision. In the rest part of §3, §4 and §5, we use the two-packet collision (instead of multi-packet) and only the I phase baseband samples (instead of I-Q phases) as examples to explain the mZig design easily.

• **Collision with chip-level time offset**. A collision in this category shows that any chip in packet Alice does not align with any chip in packet Bob. Mathematically, a chip-level time offset exists when

$$\tau_{AB} = |T_{AB}| \ \mathbf{mod} \ \frac{1}{C} \neq 0, \tag{3}$$

where τ_{AB} is the chip-level time offset, $|T_{AB}|$ is the packet-level time offset between Alice and Bob in a collision, **mod** is the modulus operator, and $1/C$ is the duration of a chip. Some examples of collisions w/ CTO are shown in Fig. 7(a)(b).

• **Collision without chip-level time offset**. A collision in this category is that the overlapped chips between Alice and Bob are aligned. Such a collision can be judged by

$$\tau_{AB} = |T_{AB}| \ \mathbf{mod} \ \frac{1}{C} = 0. \tag{4}$$

Some examples of collisions w/o CTO are shown in Fig. 7(c)(d). In particular, even two packets have a packet-level time offset, it is possible that they have no chip-level time offset as shown in Fig. 7(d).

3.3 CrossIC Design

Cross Interference Cancellation (CrossIC) is one core design in mZig, which leverages the unique features of 'oversampling' and 'known shaping' to decompose multiple packets in a collision with chip-level time offset. The basic idea of CrossIC is to use collision-free samples and known shaping to estimate next some samples within a chip duration.

Denote λ to be the number of samples in one chip, where $\lambda = S/(C/2)$, where $C/2$ is the chip rate of I phase. Since CTO τ_{AB} is able to be detected (refer to §4.1), the number of samples k in CTO can be calculated by $k = \lfloor \tau_{AB} S \rfloor$, τ_{AB} is the duration of CTO and S is the sampling rate. When two packets are detected in one collision, we denote A and B as the packets from Alice and Bob, respectively. As the analysis in §3.1, two unknown variable $X_A[n]$ and $X_B[n]$ cannot be resolved simultaneously in Eq. (2) with only one known sample $Y[n]$. In order to decompose $X_A[n]$ and $X_B[n]$, CrossIC is to introduce the shaping relationship as a new information, which leverages k collision-free samples $X_A[n-1], X_A[n-2], \cdots, X_A[n-k]$ to estimate $X_A[n]$. Then, $X_B[n]$ in Eq. (2) is able to be calculated.

Based on the oversampling and the known shaping features, multiple samples forming a half-sine pulse shaping can be represented by

$$\begin{aligned} H_A X_A[i] &= c\ \alpha \sin(\pi f t), \\ &= c\ \alpha \sin(\pi f \Delta (i - n + k + 1)), \end{aligned} \tag{5}$$

where $H_A X_A[i]$ is the value of i-th sample, c is the sign indicator (+1 for chip '1' and -1 for chip '0'), α is the amplitude of a chip in packet A, $\sin(\pi f t)$ presents the half-sine waveform, $f = C/2 = 1$Mchip/s is the chip rate of I phase, t is the duration of one chip whose range is $0 \leq t \leq 1\mu$s, $t = \Delta(i - n + k + 1)$ bridges t to the i-th sample, Δ is the time interval between two samples, and $(i - n + k + 1)$ aligns the i-th sample to the range $0 \leq t \leq 1\mu$s.

CrossIC decomposes a collision chip-by-chip in a cross manner, *i.e.*, one chip for Alice, then one chip for Bob. For every chip, CrossIC have two steps: extraction and estimation. Using the example in Fig. 2, we explain how CrossIC works in detail.

• Extraction A: extract k collision-free samples in the first chip of A. When the packet-level time offset T_{AB} is detected, the chip-level time offset is $\tau_{AB} = T_{AB} \mathbf{mod} \frac{1}{C}$. Thus, the number of collision-free samples $k = \lfloor \tau_{AB} S \rfloor$ can be calculated. In Fig. 2(b), k=3 collision-free samples are extracted.

• Estimation A: use k collision-free samples to estimate the next $(\lambda - k)$ samples in the first chip of A. We discuss this estimation step in two cases.

Case I ($\tau_{AB} = |T_{AB}| \neq 0$): When the chip-level time offset is equal to the packet-level time offset as shown in Fig. 7(a), there is no pre-knowledge of α. Hence, in Eq. (5), two variables c and α are unknown. The extracted collision-free samples are used to not only judge c but also estimate α. Given k collision-free samples, c is determined by

$$c = \begin{cases} 1, & \text{if} \sum_{i=n-k}^{n-1} H_A X_A[i] > 0; \\ -1, & \text{if} \sum_{i=n-k}^{n-1} H_A X_A[i] < 0. \end{cases} \tag{6}$$

When the sum of k samples is larger than 0, this chip is '1' and the waveform is a positive half-sine, so $c = 1$. Otherwise, the chip is '0' with a negative half-sine, so $c = -1$.

Only having c is inadequate to decompose the next chips. The value of α is required to be known as well. Substituting any collision-free sample into Eq. (5), we obtain one result of α. However, because collision-free samples suffer from noise in wireless channels, k different values of α are obtained by k collision-free samples. To approach the real value, we adopt the average value $\widetilde{\alpha}$ as the estimated result:

$$\widetilde{\alpha} = \frac{\sum_{i=n-k}^{n-1} H_A X_A[i]}{c \sum_{i=n-k}^{n-1} \sin(\pi f \Delta (i - n + k + 1))}. \tag{7}$$

Substituting c and $\widetilde{\alpha}$ into Eq. (5), we can calculate the val-

ues of next $(\lambda - k)$ samples of A such as $X_A[n], X_A[n+1], \cdots, X_A[n+\lambda-k-1]$. In Fig. 2(b), $(\lambda - k) = 2$ samples of the first chip for Alice are estimated.

Case II ($\tau_{AB} \neq 0$ and $\tau_{AB} < |T_{AB}|$): When the chip-level time offset is not equal to the packet-level time offset as shown in Fig. 7(b), there are some collision-free chips at the beginning of packet. The solution for Case II is simpler than Case I, because α can be measured from the collision-free chips directly. With k collision-free samples, we just need to determine c by Eq. (6). And the next $(\lambda - k)$ samples of A can be estimated as the same method in Case I.

• Extraction B: extract $(\lambda - k)$ collision-free samples in the first chip of B. When the $(\lambda - k)$ samples $H_A X_A[i]$ $(i = n, n+1, \cdots, n+\lambda-k-1)$ are estimated, the $(\lambda - k)$ samples $H_B X_B[i]$ can be calculated by $Y[i] - H_A X_A[i]$, (we ignore noise W in this section and will discuss it in §4.2). Then, these subtracted samples $H_B X_B[i]$ become collision-free. In Fig. 2(b), $(\lambda - k) = 2$ samples of Bob are collision-free after the substraction.

• Estimation B: use $(\lambda - k)$ collision-free samples to estimate the next k samples in the first chip of B. Similarly, the estimation B step is similar to the estimation A, where the explicit equations are as follows.

$$H_B X_B[i] = c\ \beta \sin(\pi f \Delta (i - n + 1)). \tag{8}$$

$$c = \begin{cases} 1, & \text{if} \sum_{i=n}^{n+\lambda-k-1} H_B X_B[i] > 0; \\ -1, & \text{if} \sum_{i=n}^{n+\lambda-k-1} H_B X_B[i] < 0. \end{cases} \tag{9}$$

$$\widetilde{\beta} = \frac{\sum_{i=n}^{n+\lambda-k-1} H_B X_B[i]}{c \sum_{i=n}^{n+\lambda-k-1} \sin(\pi f \Delta (i - n + 1))}. \tag{10}$$

After the estimation by above three equations, the values of next k samples of B can be calculated. In Fig. 2(b), $k = 2$ samples of the first chip for Bob are estimated.

Repeating the extraction and the estimation iteratively as shown in Fig. 2(c), mZig decomposes the collided packet into two packets A and B chip-by-chip.

The estimation errors of $\widetilde{\alpha}$ and $\widetilde{\beta}$ depend on the signal-noise ratio (SNR). Assume the average noise on every sample is $\overline{W}$. In the conventional ZigBee, the SNR of Alice's chip is

$$\frac{\text{Signal}}{\text{Noise}} = \frac{\sum_{i=1}^{\lambda} \alpha \sin(\pi f \Delta i)}{\overline{W}\lambda}. \tag{11}$$

In CrossIC, the SNRs of Alice's chip and Bob's chip are

$$\frac{\sum_{i=1}^{k} \alpha \sin(\pi f \Delta i)}{\overline{W}k} \quad \text{and} \quad \frac{\sum_{i=k+1}^{\lambda} \beta \sin(\pi f \Delta i)}{\overline{W}(\lambda - k)}, \tag{12}$$

respectively. Compared with ZigBee, the SNRs for chip estimation in CrossIC are a little reduced, which will slightly increase the estimation errors. More impacts of SNRs will be shown in simulation §8.2.

Furthermore, the backward CrossIC is able to operate from the last chip to the first one for a collided packet. In Fig.7(a), we find that there is not only a CTO at the first chip but also a CTO at the last chip. Hence, the decomposition by CrossIC on two directions is symmetric. One more decomposed result is obtained by the backward CrossIC, which is a double check to reduce the estimation errors.

3.4 AmpCoD Design

Amplitude Combination based Decomposition (AmpCoD) is another core design in mZig, which leverages the unique feature of 'uniform amplitude' to decompose packets in a collision w/o CTO. Since there is no CTO, CrossIC fails to work due to no collision-free samples. To this end, AmpCoD is proposed to leverage the different amplitude combinations to decompose every collided chip into two original chips.

Table 2: Four amplitude combinations in a two-packet collision without chip-level time offset.

Alice	chip='1'	'1'	'0'	'0'
Bob	chip='1'	'0'	'1'	'0'
Amplitude	$\alpha+\beta$	$\alpha-\beta$	$-\alpha+\beta$	$-\alpha-\beta$

Consider if only Alice transmits her packet, each received chip at RX is a half-sine waveform with the amplitude either α when chip='1' or $-\alpha$ when chip='0'. If only Bob transmits his packet, the amplitude is either β or $-\beta$. When they transmit concurrently, their signals are added up in air. In the scenario of two-packet collision w/o CTO, their amplitudes form $2^2 = 4$ combinations as listed in Tab. 2. AmpCoD requires that the amplitude difference exists between two packets, *i.e.*, $|\alpha - \beta| > \max(\epsilon, \overline{W})$, where ϵ is the amplitude unit of ADC (*e.g.*, the amplitude unit of an 8-bit ADC is $(\alpha+\beta)/2^8$.) and $\overline{W}$ is the average noise.

AmpCoD also have two steps: amplitude statistic and chip identification. Using the example in Fig. 3, we explain how AmpCoD works in detail.

• Amplitude statistic: count the number of different amplitude levels, estimate the number of packets, and build the relationship between amplitudes. We discuss this statistic step in two cases.

Case III ($\tau_{AB} = |T_{AB}| = 0$): When neither the chip-level nor the packet-level has any time offset as shown in Fig. 7(c), the waveform of a collided chip is still half-sine and there is no pre-knowledge of α or β. AmpCoD needs to count the number of different amplitudes of all chips in the collided packet. If four different levels can be found in the statistics, AmpCoD can determine that the collision is a two-packet collision. Denote these four levels by $L_1, \cdots, L_4$, where $L_1 > L_2 > L_3 > L_4$. Without loss of generality, we assume $\alpha > \beta$. Using Tab. 2, AmpCoD can build a mapping relationship between Ls and amplitude combinations

$$\begin{cases} L_1 = \alpha + \beta; & L_3 = -\alpha + \beta; \\ L_2 = \alpha - \beta; & L_4 = -\alpha - \beta. \end{cases} \tag{13}$$

Case IV ($\tau_{AB} = 0$ and $|T_{AB}| \neq 0$): When a collision has no CTO but has packet-level time offset as shown in Fig. 7(d), α or/and β can be obtained from the collision-free chips. For example, in Fig. 7(d), α can be obtained from the last chip and β can be obtained from the first two chips. In this case, the procedures including the counting of different levels and the estimation of the number of collided packets can be skipped. Then, the mapping relationship in Eq. (13) can be built by known α and β directly.

• Chip identification: identify every chip in the collided packet to be multiple chips according to the built relationship. When a chip's level approaches L_i, both case III and case IV identify this chip using Eq. (13). For example, the amplitude of the fifth chip in Fig. 3 approaches L_4, this chip can be decomposed to be a chip '0' for Alice because of $-\alpha$ and a chip '0' for Bob because of $-\beta$.

4. DESIGN ENHANCEMENT FOR MZIG

With only the core design in §3, mZig cannot provide a satisfactory performance because several practical issues

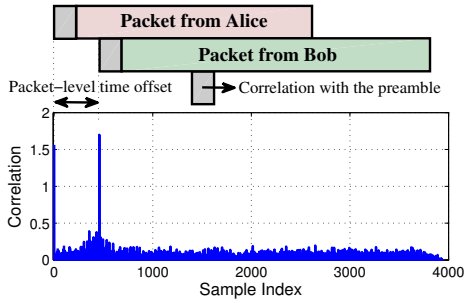

Figure 8: Time offset detection for collision

largely affect the decomposition, such as noise, multipath, and frequency offset. This section presents the design enhancement for mZig in order to mitigate these effects.

4.1 Time Offset Detection

Collision detection and time offset detection are required in practical mZig.

To detect a collision with packet-level time offset, we adopt the similar method in [15]. Any ZigBee packet starts with a preamble including 32 bits of '0's [2]. A collision can be detected by this known preamble using correlation calculation. The known preamble has $L = 32 \times \frac{32}{4} \times \lambda = 256\lambda$ samples, where $\frac{32}{4}$ is the redundancy of bit-to-chip spreading, and λ is the number of samples in a chip. Align these L samples with the first L received samples and compute their correlation. Then, shift the alignment to the next sample and re-compute. Repeat this process until the end of the packet. The correlation result is near zero except when the preamble is perfectly aligned with the beginning of one certain packet. An example of the correlation result for a two-packet collision is shown in Fig. 8. A two-packet collision is detected if there are two spikes in the result. One spike is at the beginning of the collided packet, and the position of the second spike indicates the start of the second packet. Hence, the packet-level time offset $|T_{AB}|$ is detected by the distance between two spikes. Then, the chip-level time offset τ_{AB} can be calculated by Eq. (4). In addition, an m-packet collision is detected if there are m spikes in the result.

To detect a collision without packet-level time offset, we propose a detection method, namely 2^m Amplitude Levels (2MAL). When a two-packet collision with $T_{AB} = 0$, the first 256 chips have two levels of amplitudes (*i.e.*, $\alpha+\beta$ and $-\alpha-\beta$) because they have the same 32-bit '0's preamble. In addition, it is possible that there are some collision-free chips in the end of the packet. These collision-free chips can be detected by having two levels of amplitudes, where these two amplitudes are in the range of preamble's two amplitudes because either $|\alpha|$ or $|\beta|$ is $< |\alpha+\beta|$. However, in the other chips, four levels of amplitudes can be found as discussion in Tab. 2. Hence, 2MAL works as follows: if two-level amplitudes are counted in the first 256 chips and 2^m-level amplitudes are counted in the collided chips, an m-packet collision is detected without packet-level time offset.

4.2 Anti-Noise Design

Noises are inevitable in real wireless channels. In order to reduce the negative effect of noises, we propose the anti-noise design to enhance the performance of mZig.

Because the noises are time-varying (*e.g.*, the common noise model follows the Gaussian distribution $W \sim \mathcal{N}(0, \sigma^2)$), the effects of noises on every chip are different. The anti-noise design leverages the concept of average noises $\overline{W}$, so the effect of noises on different chips trends to the same.

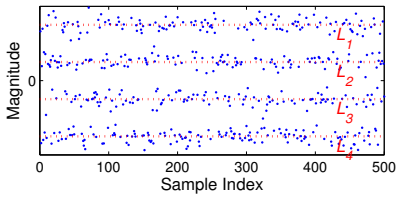

Figure 9: Measured amplitudes of samples.

In CrossIC, the estimation of α (or β) is chip-independent. Thus, the estimated $\widetilde{\alpha}$ of every chip is different when noises $W[n]$s exist. However, the original αs should be the same due to the uniform amplitude feature. After all $\widetilde{\alpha}$s are obtained by CrossIC, the anti-noise design averages all $\widetilde{\alpha}$s to be an $\overline{\alpha}$, in which the noises $W[n]$s are also averaged to be a $\overline{W}$. The effect of noises can be largely reduced by taking $\overline{\alpha}$ as the known amplitude and re-operating CrossIC.

In AmpCoD and 2MAL, the determination of L in the amplitude statistic step is also affected by noises. For example, a original sample should be L_1 but the measured sample is $L_1 + W[n]$. The measured samples of amplitudes in a collision w/o CTO are shown in Fig. 9. The anti-noise design in AmpCoD and 2MAL also resorts to the average concept in order to reduce the effect of noises in α and β estimations. There are four steps in the anti-noise design. (i) Use the channel estimation [18] to get a rough $\overline{W}$. (ii) Use a typical classification method (*e.g.*, SVM [6]) with parameter $\overline{W}$ to classify samples and get the number of amplitude levels. (iii) If the number of amplitude levels is a power of 2^m, an m-packet collision is detected. (iv) Average all amplitudes belonging to one level to get the estimated amplitude of this level as shown in Fig. 9. In addition, the relationship between levels can be used to assist the determination. For example, in Eq. (13), $L_1 = -L_4$ and $L_2 = -L_3$.

4.3 Multipath Filter

In conventional one-TX one-RX ZigBee communications, since the chip rate is low, 2Mchip/s, the multipath effect is not severe (much smaller than WiFi, whose bandwidth is 20MHz). Thus, the redundancy of bit-to-chip spreading is adequate to against multipath. However, in the multi-TX one-RX convergecast scenario, multiple transmissions suffer from different multipath effects and their resultant effect is severe on accurate decomposition. As a result, we design a multipath filter method to reduce the multipath effect.

This method also resorts to the result of channel estimation [18]. The mutlipath feature of any channel between one certain TX to RX can be modeled by a sequences of impulse responses. For example, the multipath feature between Alice and RX is shown in Fig. 10. When a chip '1' is determined, its multipath effect can be calculated by the chip's waveform convolving with the impulse responses as shown in Fig. 10. Assume the channel estimations of Alice-to-RX and Bob-to-RX are known, whenever a chip is estimated, (i) match the amplitude to the channel estimation result in order to know which channel this chip belongs to, (ii) calculate the multipath effect by convolution, and (iii) reduce the estimated multipath effect of this chip from the collided packet. The multipath filter operates as a plug-in step in CrossIC/AmpCoD chip-by-chip during the decomposition.

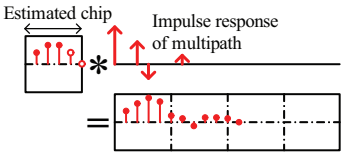

Figure 10: Multipath filter.

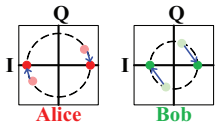

Figure 11: Frequency offset compensation.

4.4 Frequency Offset Compensation

A frequency offset always exists between two commercial radios. Such an offset causes a linear displacement in the phase of the received signal. Typically, a RX estimates the frequency offset by phase tracking and then compensates for it. The frequency offset can be tracked using any prior collision-free packet [15, 34].

In the one-TX one-RX scenario, the compensation of frequency offset is directly applied on the received packet at RX. However, in the convergecast scenario, *e.g.*, two-TX and one-RX, the conventional method fails because the frequency offsets of Alice-to-RX and Bob-to-RX are different. Mathematically,

$$Y[n] = H_A X_A[n] e^{j2\pi n \delta_A \Delta} + H_B X_B[n] e^{j2\pi n \delta_B \Delta} + W[n], \quad (14)$$

where δ_A and δ_B are the frequency offsets of Alice-to-RX and Bob-to-RX, respectively. Our solution compensates the frequency offset in a chip-by-chip manner. In CrossIC, there are two existing steps for every chip: extraction and estimation. With the compensation, the number of steps is extended to three. The new second step is to compensate the frequency offset on the extracted samples. For example, the frequency offsets of Alice and Bob are tracked as shown in Fig. 11. After the step of Extraction A in §3.3, the new second step compensates the tracked offset (Alice-to-RX) on the collision-free samples. The step of Estimation A operates then. Similarly, decomposing a chip for Bob follows the steps: Extraction B, Compensation B, and Estimation B.

5. MZIG ANALYSIS

This section analyzes the capability of mZig, the scope of mZig, and the transplant of mZig to Bluetooth.

5.1 Beyond Two-Packet Collision

The proposed mZig is easy to extend from two-packet collision to m-packet collision. We use three-packet collision as an example. More concurrent transmissions can adopt the same extension method.

The correlation method proposed in §4.1 can detect m-packet collision w/ CTO by m spikes. If there are three spikes detected, a three-packet collision w/ CTO are determined. Using these three spikes, the packet-level time offsets T_{AB}, T_{BC}, T_{CA} can be calculated. Thus, the CTOs τ_{AB}, τ_{BC}, τ_{CA} are also available. Assume that the samples in τ_{AB} are collision-free at the beginning of the collision. CrossIC (§3.3) operates as: extract these samples, and the first chip of Alice is estimated. Then, extract the samples in τ_{BC}, the first chip of Bob can be estimated. At last, extract the samples in τ_{CA}, the first chip of Carol can be estimated. Iteratively operate the extraction and estimation chip-by-chip, a three-packet collision w/ CTO can be decomposed.

The 2MAL method in §4.1 can detect m-packet collision w/o CTO by 2^m amplitude levels. For example, if there are eight different amplitude levels detected by 2MAL, a three-packet collision w/o CTO can be determined. Denote the eight levels by $L_1, L_2 \cdots, L_8$. Assume that $L_1 > L_2 > \cdots > L_8$ and $\alpha > \beta > \zeta > 0$, where ζ is the chip amplitude of the third TX Carol. With the top-2 levels $L_1 = \alpha + \beta + \zeta$ and $L_2 = \alpha + \beta - \zeta$, we have $\zeta = (L_1 - L_2)/2$. Similarly, with L_1 and $L3$, the value of β is determined. With known L_1, β and ζ, we can determine α. Then, the mapping relationship between the amplitude levels and chips can be built. One mapping example is listed in Tab. 3. Based on this table, chips can be decomposed by AmpCoD (§3.4).

Table 3: Eight amplitude levels in a three-packet collision without chip-level time offset.

Alice	'1'	'1'	'1'	'1'	'0'	'0'	'0'	'0'
Bob	'1'	'1'	'0'	'0'	'1'	'1'	'0'	'0'
Carol	'1'	'0'	'1'	'0'	'1'	'0'	'1'	'0'
Level	L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_8

The hybrid m-packet collision is also possible. For example, in a three-packet collision, packets A and B have a CTO while packets A and C have no CTO. In such a case, CrossIC is firstly adopted to decompose the overlapped signal into two group: B and AC, while AC is considered as one packet here because they have no CTO. After B is decomposed out, packets A and C can be decomposed by AmpCoD.

5.2 Error Propagation

Up to now, we have described the system assuming correct decoding. In practice, the estimation of one certain chip may be error. For example, a chip '1' is falsely estimated to be '0'. If a chip error occurs during decomposition, it may affect later chips because of the iterative decomposition manner. Using the analysis in ZigZag [15], we have the result that the error propagation in mZig dies exponentially fast as ZigZag. More error propagation analysis in iterative decoding method can refer to SigSag [36]. We will show the impact of error propagation in performance evaluation §7.2.

5.3 Scope

The proposed mZig adopts a best effort design. In the absence of collisions, mZig acts like the conventional ZigBee. However, when collisions occur, mZig attempts to decompose and decode them.

The theoretical upper bound of concurrent transmission in mZig is $M = \max(m) = \lfloor \frac{S}{2 \times C} \rfloor$, where C is the chip rate, and 2 indicates that at least two collision-free samples are needed to estimate one chip. Using only one collision-free sample may result in estimation failure because Eq. (5) cannot be successfully resolved if $X_A[i] = 0$. However, when there are two successive samples, the half-sine pulse shaping can guarantee that at least one sample is nonzero. From the theoretical upper bound, we observe that the capability of mZig is mainly determined by the sampling rate of the ADC in wireless device. The practical concurrent capability is smaller than the theoretical one due to noises and estimation errors. We will show the experimental results in §7.2.

Most collision patterns are able to be addressed by mZig. Nevertheless, mZig is not omni-directional. For example, AmpCoD fails when any two packets have the same amplitude, in which the levels of amplitude combinations $(\alpha - \beta)$ and $(-\alpha + \beta)$ cannot be distinguished. However, the mZig design is orthogonal to other collision resolution techniques. If the failure of mZig occurs, other alternative techniques can be triggered to resolve the collision.

The proposed mZig is a customized technique for ZigBee, which leverages ZigBee's features in physical layer. mZig

Table 4: Comparison of physical layer features between Bluetooth Low Energy (BLE) and ZigBee.

	BLE	ZigBee
Required features for mZig		
Oversampling	Yes, 1Msymbol/s	Yes, 2Mchip/s
Know shaping	Yes, Gaussian	Yes, Half-sine
Uniform amplitude	Yes, GFSK	Yes, O-QPSK
Other physical layer features		
Baseband unit	Symbol	Chip
Bitrate	1Mbps	250kbps
Spreading technique	FHSS	DSSS

cannot be applied in WiFi because their physical layer features are totally different. For example, WiFi adopts the raised cosine pulse shaping (inter-symbol interferences exist in this shaping method, so the chip waveform is not independent) and the QAM modulation (multiple levels of amplitudes in one packet). However, mZig is able to extend to other wireless protocol, which have the similar features.

5.4 Transplant of mZig in Bluetooth

Bluetooth Low Energy (BLE) [4] is another key protocol focusing on the field of low-power communications. The core design of mZig can be easily transplanted to BLE, because BLE has the similar physical layer features as ZigBee. In Tab. 4, we compare the physical layer features between BLE and ZigBee. We find that (i) BLE has the oversampling feature, because its symbol rate is 1Msymbol/s, where symbol is the smallest unit carrying information in BLE's baseband. If an off-the-shelf ADC has 61.44Msps, the samples of a symbol is more than 60. (ii) BLE has the known shaping feature. Based on the Gaussian pulse shaping, the waveform of every symbol in BLE is known and the Gaussian shaping is symbol-independent. (iii) BLE has the uniform amplitude feature. Since BLE adopts GFSK modulation, which modulates samples by frequency but not amplitude, the amplitudes of symbols in one packet are the same.

Several other physical layer features are different between BLE and ZigBee, so the enhancement designs for BLE are different from ZigBee. Since this paper mainly focuses on ZigBee, we do not implement mZig in BLE. However, we simulate the performance of mZig in BLE in §8.2.

6. IMPLEMENTATION

We implement mZig on USRP and build a seven-node testbed. This section presents the details of implementation including the physical (PHY) layer, the media access control (MAC) layer, and the prototype.

6.1 Physical Layer Development

TX side: mZig requires no change on the PHY of TXs. Hence, any TX in our prototype adopts the conventional PHY of ZigBee as shown in Fig. 4.

RX side: mZig requires some lightweight changes on the PHY of RX, including a new decomposition module, namely Decomposition module for mZig (DmZig), and M parallel decoding lines as shown in Fig. 12.

The functionality of DmZig is to decompose the received baseband samples of a collision into m sequences of collision-free samples. The flow chart of DmZig module is illustrated in Fig. 13. The main procedure of DmZig is as follows. (i) DmZig detects whether there is a collision in the baseband samples using the correlation and the 2MAL methods (§4.1). If there is no collision, this sequence of collision-free samples

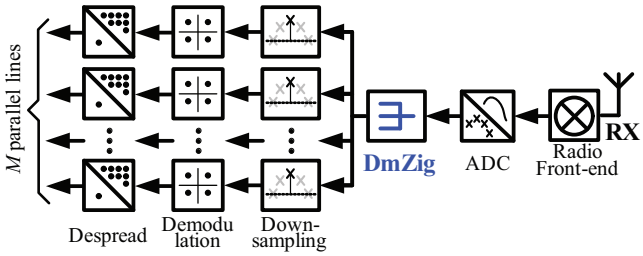

Figure 12: Physical layer of an RX with mZig.

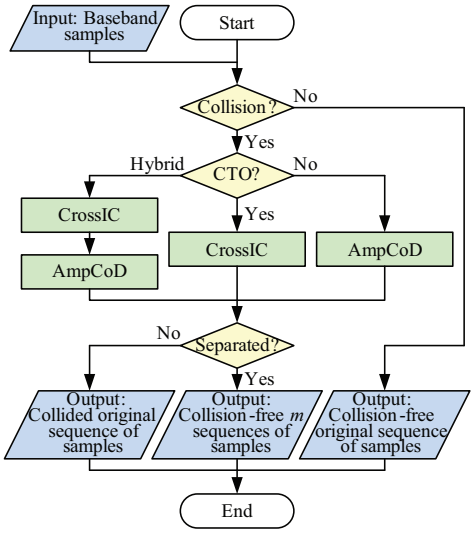

Figure 13: Flow chart in DmZig module.

are output directly. If a collision is detected, DmZig needs to know how many concurrent packets, *i.e.*, m, in this collision and the collision type of these m packets. (ii) The correlation and the 2MAL methods (§4.1) are used to detect the collision type. If the collision is detected w/o CTO, AmpCoD (§3.4) is adopted to decompose the samples. If the collision is detected w/ CTO, CrossIC (§3.3) is adopted. If a hybrid collision (§5.1) is detected, samples are decomposed by CrossIC and AmpCoD in a serial manner. (iii) There are three kinds of outputs in DmZig as shown in Fig. 12. If there is no collision, the output is the original baseband samples, and one decoding line can deal with them; if the collision can be decomposed into m sequences, the output is m sequences of samples, and m decoding lines can deal with them; if the collision can be detected but cannot be decomposed, the output includes a flag of 'collision' and the original samples, then the other alternative collision resolution techniques [15, 17] will be triggered.

With the core and the enhancement designs together, the full version of CrossIC implemented in PHY includes:

- Collision-free samples extraction (§3.3),
- Frequency offset compensation (§4.4),
- Chip estimation (§3.3),
- Multipath filter (§4.3).

Above four steps are iteratively operated. After operating these four steps from the first to the last chip, CrossIC needs to do the anti-noise method (§4.2) and repeats above four steps again for reducing the effect of noises. In addition, the backward CrossIC (§3.3) is used for double check in the end.

The full version of AmpCoD includes:

- Amplitude statistic (§3.4),
- Anti-noise design (§4.2),

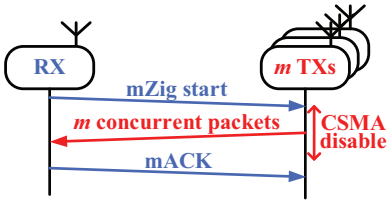

Figure 14: A basic MAC for mZig

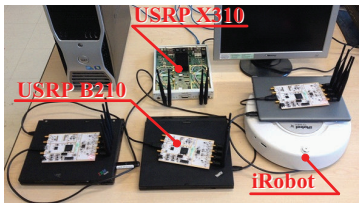

Figure 15: Testbed

• Chip identification (§3.4).

The functionality of M parallel decoding lines is to decode m sequences of collision-free samples (output of DmZig module) into bits in a parallel manner, so that the time consumption on decoding is reduced and the MPR is achieved. Since the number of concurrent transmissions m is not known a priori, we set the value of M is equal to the theoretical upper bound of m, where $M = \max(m) = \lfloor \frac{S}{2\times C} \rfloor$. When the DmZig module outputs m sequences of collision-free samples, m (where $m \leq M$) decoding lines are activated and decode m sequences respectively. The implementation complexity of M decoding lines is light, which is just M copies of the conventional decoding line.

6.2 Media Access Control Development

Although MAC design is not our focus in this paper, a basic MAC is still needed to be developed. The conventional MAC cannot be directly applied to mZig due to two problems: (i) The conventional MAC includes CSMA/CA. With CSMA/CA, multiple TXs cannot transmit concurrently. (ii) The conventional MAC includes the acknowledgement (ACK), which is sent by RX to TX for confirming the successful packet reception. But the conventional ACK is not suitable for the multi-TX one-RX convergecast.

A basic MAC for mZig is developed in our implementation, which is shown in Fig. 14, including three steps.

• The RX broadcasts an 'mZig start' message to inform that $\widetilde{M}$ TXs can transmit concurrently in the following time window, where the duration of window is given by this RX, and $\widetilde{M}$ is a number given by the customized scheduler.

• When TXs receive the 'mZig start' message, they disable their CSMA/CA and send packets during the window. CSMA/CA is re-opened after the window time is expired.

• When the window time is expired, the RX stops to receive packets, and sends an ACK to all TXs. This ACK for mZig is named mACK, which include ACKs for $\widetilde{M}$ TXs with all successfully received packets during the window.

In particular, we design an online scheduler built-in the MAC to give an estimated $\widetilde{M}$, where $\widetilde{M}$ is the practical capability of concurrent transmissions. If m TXs send packets concurrently and $m > \widetilde{M}$, mZig fails to decompose the collisions. In order to deal with the $m > \widetilde{M}$ cases, we design this scheduler to assign only $\widetilde{M}$ TXs can transmit in the following window. However, the value of $\widetilde{M}$ is not fixed, because it is varying according to the dynamic environment. This scheduler estimates the value of $\widetilde{M}$ in an online manner according to the bit error rate (BER) of all decoded packets. When BER$<<\xi$, $\widetilde{M}$ is increased; when BER $>\xi$, $\widetilde{M}$ is reduced; when BER$\leq\xi$ but not too far, $\widetilde{M}$ maintains its current value. In our experiment, we set $\xi = 10^{-3}$, which is a common setting in wireless communication to determine the successful reception of a packet. We set BER$<<\xi$ as BER$< 10^{-4}$. The initial $\widetilde{M}$ is set as the theoretical M.

The other required functionalities such as phase tracking and channel estimation are the same as the typical ZigBee.

6.3 Prototype Development

We develop the prototype of mZig as follows.

Hardware: The hardware includes one USRP X310, six USRP B210s, and six iRobots. We develop mZig RX in USRP X310, which is a fixed but powerful software defined radio (SDR) device. USRP X310 is linked to a desktop. We develop mZig TXs in USRP B210s, which are portable SDRs, and power supplied by USB 3.0 port. USRP B210s are linked to laptops. Six iRobots can randomly move in a plane with speed less than 0.4m/s. These iRobots carry the laptops and USRP B210s to test not only static but also the dynamic channel effects on mZig in our experiment. Partial hardware devices are shown in Fig. 15.

We select USRP as mZig RX because of its high sampling-rate ADC. Some commercial ZigBee devices such as TelosB adopt 4Msps sampling rate, which are inadequate to operate CrossIC in mZig. On the other hand, both USRP and TelosB can serve as mZig TXs. In this work, we select USRPs as TXs so that the transmission samples can be logged at TX sides for chip-level comparison.

Software: GnuRadio is the software for developing mZig.

Testbed: Our testbed includes seven mZig nodes in a general office, whose area is $7.5\times6.8\text{m}^2$. Six TXs and one RX build a convergecast topology.

7. EXPERIMENT

Using the seven-node testbed, we conduct experiments to verify the feasibility of mZig and evaluate its performance.

7.1 Experiment Setting

Configuration:

• Sampling rate. The default sampling rate is set S=32Msps, which is 16x of the chip rate.

• Transmission power. We set Tx_Gain to be 70 in GnuRadio, so the transmission power is 0dB (USRP B210 is a uncalibrated device on its output power, so this setting is just 'near' 0dB according to the specification [1]), where 0dB (1mW) is the default TX power conditioned by ZigBee standard. With such a power setting, the RX can receive any TX's transmission in our $7.5\times6.8\text{m}^2$ office space.

• Channel selection. In an office environment, there are some WiFi and Bluetooth radios in 2.4GHz band. To avoid the effect of coexistence [11, 47, 50], we select the Channel 26 in our mZig experiment, which is a non-overlapping channel with WiFi. In addition, Bluetooth [4] adopts FHSS technology. Its working frequency quickly hops 1600 times per second. There is only one time channel overlap in 79 times. A comparative study [33] reveals that Bluetooth does not disturb ZigBee in most instances.

• Packet length. Each transmitting packet consists of a 32-bit preamble and a random payload of 200, 400, 600, 800,

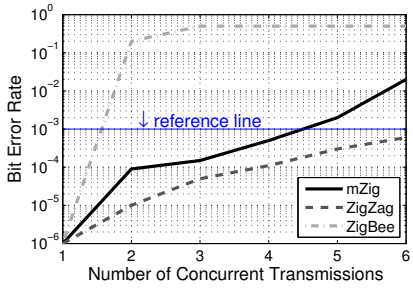

Figure 16: Comparison of different schemes on BER.

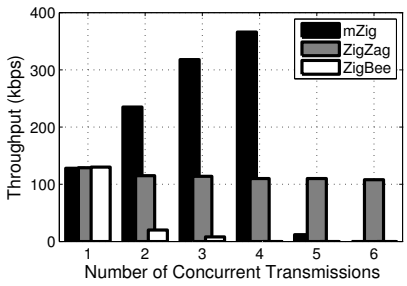

Figure 17: Comparison of schemes on throughput (without MAC).

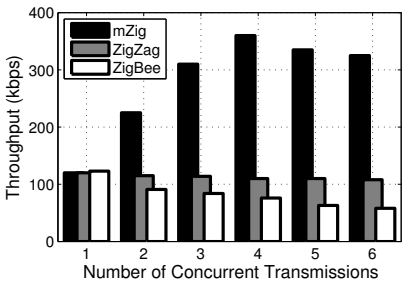

Figure 18: Comparison of schemes on throughput (with MAC).

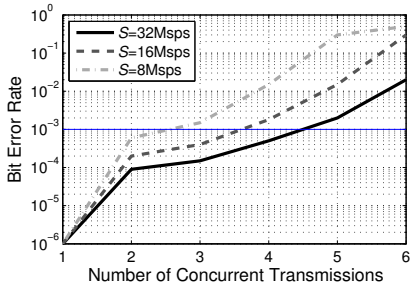

Figure 19: Impact of sampling rate

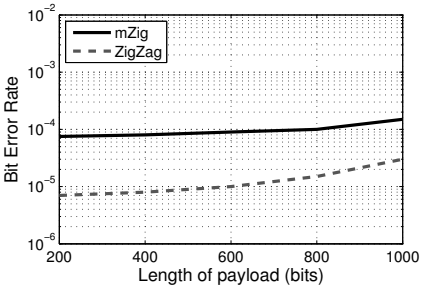

Figure 20: Impact of packet length.

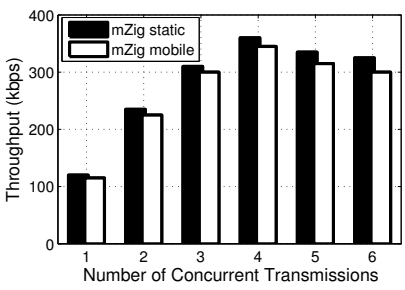

Figure 21: Impact of mobility.

Table 5: Probabilities of different collisions.

m TXs	w/ CTO	w/o CTO	hybrid	collision-free
2	93.4%	5.9%	0%	0.7%
3	82.3%	0.4%	17.2%	0.1%
4	65.7%	0%	34.3%	0%
5	50.3%	0%	49.7%	0%
6	35.8%	0%	64.2%	0%

or 1000 bits, which satisfies the payload length requirement of ZigBee, whose limitation is no more than 1064 bits.

Compared schemes:

• mZig: is the proposed design in this paper.

• ZigZag [15]: is the state-of-the-art design for collision resolution, which is also the closest work to our mZig. ZigZag resolves an m-packet collision using m collided packets.

• ZigBee [2]: is the conventional ZigBee protocol. ZigBee exploits CSMA/CA in its MAC to avoid collisions. ACK and retransmission mechanisms are used to guarantee the successful packet reception.

Metrics:

• Bit Error Rate (BER): is the rate of incorrect bits to all transmitted bits. We consider a packet to be correctly received if its BER is less than 10^{-3}. This setting is in accordance with typical wireless design [15].

• Throughput: is the average received bits in one second. A higher throughput indicates that more data can be received in unit time and more TXs are allowed to transmit concurrently. The throughput is the key metric to express the advantage of mZig.

7.2 Experiment Result

First, the probability statistics of different collisions in our experiments are shown in Tab. 5. We observe that most collisions are w/ CTO and hybrid. Thus, both CrossIC and AmpCoD in mZig are important to tackle different types of collisions. Moreover, the performance of CrossIC is the bottleneck of M, because most collisions have CTOs.

To verify the feasibility of mZig, we start the experiment with a varying number of TXs m from 1 to 6. In this experiment, MACs are disabled (neither scheduling in mZig nor CSMA in ZigBee) to show the decoding capability of PHYs only. Fig.16 compares BERs of three schemes with varying m. The conventional ZigBee cannot decode collisions. When m becomes 2, its BER increases to more than 2×10^{-1}. Since CrossIC in mZig leverages collision-free samples to estimate chips, more TXs lead to fewer available collision-free samples. Hence, the effect of noises on mZig increases with m. When $m \geq 5$, mZig's BER is larger than the reference line 10^{-3}. **This experiment demonstrates that the feasibility of mZig and its MPR capacity in this experimental environment is** $m = 4$. The BER of ZigZag also increases with m, but its trend is smoother than mZig. This result shows that its decoding capacity is over than 6 TXs. Although ZigZag is better than mZig on BER, mZig is better than ZigZag on throughput.

Fig.17 shows the throughput of three schemes, where MACs are still disabled. TXs transmit packets with a random interval between 1 to 4ms. When m=1, three schemes achieve a similar throughput, which is about 129kbps (smaller than the bitrate 250kbps due to channel estimation, interval, and so on). mZig largely improves the throughput with m due to its MPR capability. When $m = 2, 3, 4$, its throughput achieves 235, 318, 366kbps, respectively. However, mZig's PHY cannot decompose ≥5-packet collision, so throughput drops sharply when $m = 5$. ZigZag requires retransmissions to build m collided packets, and then m TX's packets can be resolved. Its throughput nearly keeps a constant with a little decrease with m. ZigBee's PHY cannot resolve collisions, so when $m \geq 2$, it nearly has no throughput.

Fig.18 also shows the throughput of three schemes, but MACs are enabled. For mZig, the throughput of m from 1 to 4 has only small change (about 121kbps) because of ACKs. However, when $m = 5$ and 6, the throughput is soared up to 330kbps. This result reveals the advantage of the scheduler, which learns the RX's MPR capability $\widetilde{M}$=4, and schedules 4 concurrent transmissions in every window. *i.e.*, even there are 5 TXs, only 4 TXs are assigned to transmit. Thus, the

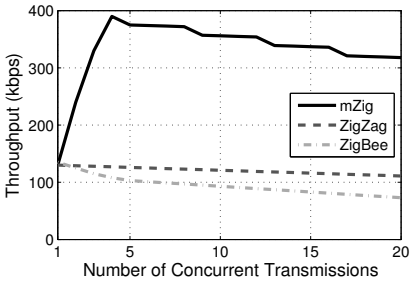

Figure 22: Throughput simulation.

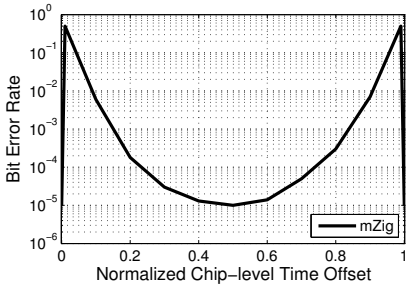

Figure 23: Impact of CTO.

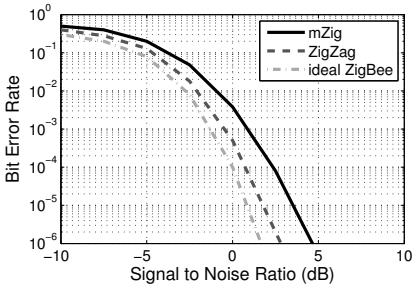

Figure 24: Impact of SNR.

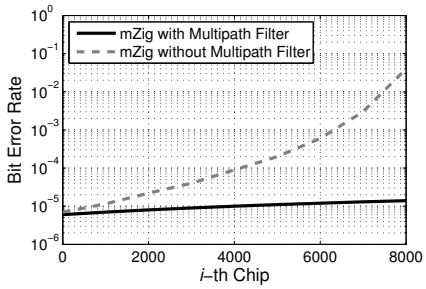

Figure 25: Impact of multipath.

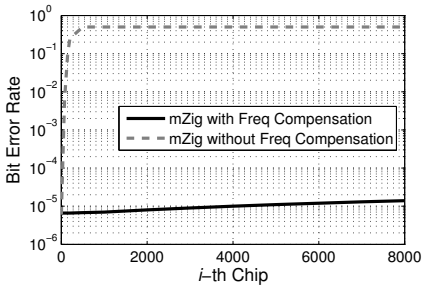

Figure 26: Impact of freq. offset.

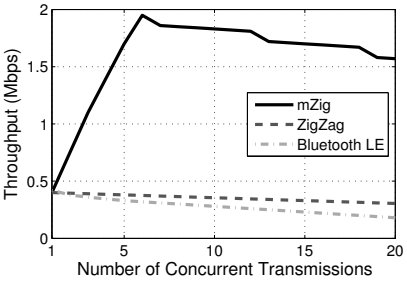

Figure 27: BLE simulation.

throughput of $m = 5$ and 6 is close to $m = 4$. A little decrease is caused by the scheduling overhead. ZigZag also changes slightly compared with Fig.17. Since MAC of ZigBee leverages CSMA to avoid collisions, only one TX can transmit packet in one time slot. We set the MAC parameters according to the ZigBee standard [2], *e.g.*, the maximal backoff is 4 and the ACK waiting duration is set as 120 symbol period. Fig.18 shows that there is some throughput in ZigBee, but it is still smaller than ZigZag. ZigBee's performance decreases with m because more TXs lead to more backoff. **This experiment demonstrates that the throughput is improved up to 3.2x by mZig compared with the state-of-the-art ZigZag, 4.5x compared with the conventional ZigBee**.

To understand the impacts of some practical factors on mZig, we conduct the following experiments.

Fig.19 mines the relation between the sampling rate and the BER in mZig. A larger sampling rate indicates more samples in one chip. *e.g.*, every chip has $\lambda = 16$ samples at 32Msps sampling rate, but $\lambda = 2$ at 4Msps. In addition, more TXs lead to smaller number of collision-free samples. For example, when $m = 2$, 32Msps has average $k = 16/2 = 8$ collision-free samples. But 4Msps has only $k = 0.5$, which is inadequate to decompose collisions w/ CTO. Fig.19 shows that the BER of mZig is gradually improved with the growth of sampling rate. When S=32Msps, the theoretical maximal $M = \lfloor 32M/(2 \times 2M) \rfloor = 8$. However, since the noises in office environment, the practical maximal $\widetilde{M} = 4$, where its BER is still smaller than 10^{-3}. When S=16Msps, $\widetilde{M} = 3$.

Fig. 20 shows the impact of different packet length, where the concurrent transmissions is set as m=2. The payload is varied from 200 to 1000 bits. We observe that BERs of both ZigZag and mZig slightly increase with the payload, which indicates that both schemes have error propagation problem, but not severe due to the limited packet length.

Fig. 21 shows the impact of mobility in mZig. Two cases are compared. In the static case, seven nodes are fixed and form a star topology, where six TXs have the same distance to the RX. In the mobile case, six iRobots randomly move in the office carrying laptops and USRPs. The speed of iRobots is set as 0.2m/s. The result in Fig. 21 is a long-term average. We find that in the mobile case, the throughput is a little lower than the static one. The reason is that the mobility increases the BER. In addition, some links are occasionally weak due to mobility.

8. SIMULATION

In the experiments, several parameters are uncontrollable, such as chip-level time offsets, noises, multipath, and frequency offset. In order to understand the impacts of these parameters, we conduct extensive simulations.

8.1 Simulation Setting

Our simulations inherit most settings from our experiments. Moreover, we amend several settings including: (i) 20 TXs and one RX form a convergecast scenario. (ii) A packet arrives at the RX with a random time point. So all collision types can be evaluated and the number of collision-free samples k is also a random number. (iii) The noise is set as a 3dBm AWGN. Three paths are set in our multipath model, where the power rate is 0.7:0.2:0.1. The frequency offset is a random angle.

8.2 Simulation Result

Our experiment performs based on a seven-node testbed. To investigate the throughput of mZig in a denser scenario, this simulation conducts based on 20 TXs and one-RX. Fig. 22 demonstrates (i) the experimental and the simulated throughput are similar. (ii) mZig can maintain the throughput at 300kbps when $m > 4$. Similarly, ZigZag and ZigBee also maintain their performance on throughput as the trends of constant with a little decrease.

Fig. 23 shows the impact of chip-level time offset. In this simulation, m is set to be 2. The X-axis presents the normalized time offset to the duration of one chip. Fig. 23 indicates that the best BER is achieved when the chip-level time offset is half of a chip duration. In this case, any chip is identified by the same number of collision-free samples. If a chip

has λ=16 samples, when m=2, the maximal collision-free samples for both Alice and Bob is its average 16/2=8.

Fig. 24 shows the BER comparison of different schemes in different SNRs when m=2. Since ZigBee has no capability to address even a two-packet collision, its BER is not plotted in this figure. Instead, we plot a reference, which is the BER of ideal ZigBee decoding a collision-free packet. In Fig. 24, ZigZag achieves the similar BER of reference. Since mZig may estimate a certain chip with very few samples depending on the CTO, the impact of SNR on its estimation is large. Hence, the BER of mZig is not good as the other two, but still within the range of 3dB.

Fig. 25 shows the impact of multipath when $m = 2$. The X-axis presents the first chip to the 8000-th chip (1000 bits are spread to 8000 chips). Since mZig operates iteratively from the first to the last chip, without the multipath filter, the value of BER gradually increases. On the contrary, with the multipath filter, a low BER is maintained. This result demonstrates the functionality of our multipath filter.

Fig. 26 shows the impact of frequency offset when $m = 2$. Without the frequency offset compensation, chips cannot be estimated correctly. For example, the BER becomes larger than 10^{-3} when just decomposing to the 4-th chip, and then quickly soars to 50%. However, with the compensation, mZig keeps a low BER. This result demonstrates the functionality of our frequency offset compensation for mZig.

In the last simulation shown in Fig. 27, we transplant mZig in Bluetooth low energy (BLE). This simulation demonstrates the feasibility of mZig in BLE. We observe that mZig achieves the throughput of 1.9Mbps when $m = 6$, which is 5.4x of BLE, and 4.7x of ZigZag. The trends of the curves and the improvement ratio for BLE are similar to the results for ZigBee in Fig. 22.

9. RELATED WORK

Convergecast cannot bypass the collision problem, which attracts extensive studies to tackle this problem from different directions. We classify them into two categories.

Collision avoidance: The conventional ZigBee adopts CSMA [14, 35] to address collisions, which avoids collisions by random backoff time and retransmission. This method fails in some scenarios such as hidden terminals. Field tests [7] show that over 10% packet loss due to collisions in WLAN. In addition, the backoff time mechanism increases the delay by scheduling transmissions into different time slots. Even some recent scheduling studies [28] aim to reduce the time consumption, the theoretical upper bound of this approach is m time slots to transmit m packets.

RTS-CTS [46] is the supplement for CSMA/CA, which addresses the hidden terminal problem by handshake. However, the handshake introduces additional overheads. Experiments [22] show that RTS-CTS significantly reduces the overall throughput. Thus, common wireless devices disable RTS-CTS by default.

Different from the collision avoidance by backoff or handshake, mZig decomposes the collisions directly. So the delay is removed and the throughput is increased.

Collision resolution: Advanced researches advocate to resolve collisions for realizing MPR.

Capture effect [25] separates collisions by requiring significant power difference among multiple packets. But this requirement cannot be guaranteed due to no prior control.

Successive interference cancellation (SIC) [17, 30, 32] resolves collisions by distinct powers or pre-coded signatures. Similar to capture effect, SIC also demands prior scheduling and known users. Thus, most SICs are designed for cellular networks [30], which have the base stations to do the scheduling. SIC has another major limitation: TXs have to operate at a data rate much lower than the maximal one.

Network coding [44], analog network coding [23], XORs [24], interference alignment [16] and full duplex [21] decode collisions by subtracting some known packets from the collided packet. These techniques are not suitable for convergecast, where packets from different TXs are unknown a priori.

PIP [27] presents a customized code to achieve parallel communications. Buzz [39] and virtual full duplex [49] exploits compressive sensing to recover concurrent transmissions from collisions. The computational overhead of these methods is high, which may increase the transmission delay.

Constructive interference [10, 12, 43] is used to receive multiple synchronized transmissions of a same packet. Although constructive interference increases the transmission reliability and reduces the power consumption in ZigBee, its major limitation is that all packets must have the same content. Hence, it is not appropriate for the convergecast, where multiple TXs send different packets.

ZigZag [15] is the most related work to this paper. If a two-packets collision occurs, ZigZag demands two TXs retransmit packets to form another collision. Then, ZigZag separates two original packets in virtue of the different time offsets in two collisions. In this way, ZigZag requires m collided packets to decode m-packet collision. Theoretically, mZig improves the throughput m-fold than ZigZag because only one collided packet is adequate to realize MPR in mZig.

In comparison, mZig decomposes a collision based on the physical layer features of ZigBee and the collision itself, while requiring no pre-coding, priori control, known packets or retransmissions. Besides, mZig is lightweight on implementation and computation. Furthermore, mZig can work with other collision resolution techniques such as ZigZag, XORs, and full duplex complementarily.

10. CONCLUSION

This paper presents mZig, a physical layer design to enable multi-packet reception in ZigBee. Leveraging the physical layer features of ZigBee, mZig decomposes multiple packets from a collision chip-by-chip. Our core contribution is a novel MPR technique customized for ZigBee. Through both theoretical analysis and performance evaluation, we show that mZig embraces the collision, receives the concurrent transmissions, and increases the throughput in convergecast.

We believe mZig has wider implications for wireless design than explored in this paper. For example, mZig PHY motivates a more aggressive MAC in ZigBee to increase throughput by exploiting concurrent transmissions. Moreover, mZig can be extended to other PSK and FSK based wireless networks such as cellular, satellite, and GPS systems.

Acknowledgements

This research was supported in part by the NSERC Discovery Grant 341823, NSERC Collaborative Research and Development Grant CRDPJ418713, Canada Foundation for Innovation (CFI)'s John R. Evans Leaders Fund 23090, NSFC Grant 61303202, and China Postdoctoral Science Foundation Grant 2014M560334.

11. REFERENCES

[1] USRP B210 data sheet. https://www.ettus.com/content/files/b200-b210_spec_sheet.pdf.

[2] ZigBee specification. http://www.zigbee.org.

[3] ZigBee smart energy profile 2 application protocol standard. http://www.zigbee.org/Standards/ZigBeeSmartEnergy/SmartEnergyProfile2.aspx, 2013.

[4] Bluetooth specification. https://www.bluetooth.org/en-us/specification/adopted-specifications, 2014.

[5] M. T. Abuelma'atti. New ASK/FSK/PSK/QAM wave generator using multiple-output operational transconductance amplifiers. *IEEE Transactions on Circuits and Systems*, 48(4):487–490, 2001.

[6] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

[7] Y. Cheng, J. Bellardo, P. Benkö, A. Snoeren, G. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In *ACM SIGCOMM*, 2006.

[8] R. Cohen and B. Kapchits. Continuous neighbor discovery in asynchronous sensor networks. *IEEE/ACM Transactions on Networking*, 19(1):69–79, 2011.

[9] J. Crowcroft and K. Paliwoda. A multicast transport protocol. In *ACM SIGCOMM Computer Communication Review*, volume 18, pages 247–256, 1988.

[10] M. Doddavenkatappa, M. C. Chan, and B. Leong. Splash: Fast data dissemination with constructive interference in wireless sensor networks. In *USENIX NSDI*, 2013.

[11] J. Fang, K. Tan, Y. Zhang, S. Chen, L. Shi, J. Zhang, Y. Zhang, and Z. Tan. Fine-grained channel access in wireless LAN. *IEEE/ACM Transactions on Networking*, 21(3):772 – 787, 2013.

[12] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *ACM IPSN*, 2011.

[13] L. Fu, Y. Qin, X. Wang, and X. Liu. Throughput and delay analysis for convergecast with MIMO in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(4):768–775, 2012.

[14] M. Garetto, T. Salonidis, and E. Knightly. Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks. *IEEE/ACM Transactions on Networking*, 16(4):864–877, 2008.

[15] S. Gollakota and D. Katabi. Zigzag decoding: combating hidden terminals in wireless networks. In *ACM SIGCOMM*, 2008.

[16] S. Gollakota, S. D. Perli, and D. Katabi. Interference alignment and cancellation. *ACM SIGCOMM Computer Communication Review*, 39(4):159–170, 2009.

[17] D. Halperin, T. Anderson, and D. Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless LANs. In *ACM MOBICOM*, 2008.

[18] J. Haupt, W. U. Bajwa, G. Raz, and R. Nowak. Toeplitz compressed sensing matrices with applications to sparse channel estimation. *IEEE Transactions on Information Theory*, 56(11):5862–5875, 2010.

[19] Y.-K. Huang, A.-C. Pang, P.-C. Hsiu, W. Zhuang, and P. Liu. Distributed throughput optimization for ZigBee cluster-tree networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(3):513–520, 2012.

[20] O. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi. Fast data collection in tree-based wireless sensor networks. *IEEE Transactions on Mobile Computing*, 11(1):86–99, 2012.

[21] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha. Practical, real-time, full duplex wireless. In *ACM MOBICOM*, 2011.

[22] G. Judd and P. Steenkiste. Using emulation to understand and improve wireless networks and applications. In *USENIX NSDI*, 2005.

[23] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: analog network coding. In *ACM SIGCOMM*, 2007.

[24] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. XORs in the air: practical wireless network coding. In *ACM SIGCOMM*, 2006.

[25] J. Kim and J. Lee. Capture effects of wireless CSMA/CA protocols in rayleigh and shadow fading channels. *IEEE Transactions on Vehicular Technology*, 48(4):1277–1286, 1999.

[26] T. Kim, S. H. Kim, J. Yang, S.-e. Yoo, and D. Kim. Neighbor table based shortcut tree routing in ZigBee wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 25(3), 2014.

[27] L. Kong, L. He, Y. Gu, M. Y. Wu, and T. He. A parallel identification protocol for RFID systems. In *IEEE INFOCOM*, 2014.

[28] J. Kwak, C.-H. Lee, and D. Y. Eun. A high-order markov chain based scheduling algorithm for low delay in csma networks. In *IEEE INFOCOM*, 2014.

[29] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet. Openairinterface: A flexible platform for 5g research. *ACM SIGCOMM Computer Communication Review*, 44(5):33–38, 2014.

[30] P. Patel and J. Holtzman. Analysis of a simple successive interference cancellation scheme in a DS/CDMA system. *IEEE Journal on Selected Areas in Communications*, 12(5):796–807, 1994.

[31] S. Sen, R. Roy Choudhury, and S. Nelakuditi. CSMA/CN: Carrier sense multiple access with collision notification. In *ACM MOBICOM*, 2010.

[32] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi. Successive interference cancellation: Carving out mac layer opportunities. *IEEE Transactions on Mobile Computing*, 12(2):346–357, 2013.

[33] A. Sikora and V. F. Groza. Coexistence of IEEE 802.15.4 with other systems in the 2.4 GHz-ISM-Band. In *IEEE IMTC*, 2005.

[34] P. Stoica and O. Besson. Training sequence design for frequency offset and frequency-selective channel estimation. *IEEE Transactions on Communications*, 51(11):1910–1917, 2003.

[35] Y. Tay, K. Jamieson, and H. Balakrishnan.

Collision-minimizing CSMA and its applications to wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1048–1057, 2004.

[36] A. Tehrani, A. Dimakis, and M. Neely. Sigsag: Iterative detection through soft message-passing. In *IEEE INFOCOM*, 2011.

[37] D. Tse and P. Viswanath. *Fundamentals of wireless communication*. Cambridge university press, 2005.

[38] H. Y. Tung, K. F. Tsang, K. T. Chui, H. C. Tung, H. R. Chi, G. P. Hancke, and K. F. Man. The generic design of a high-traffic advanced metering infrastructure using ZigBee. *IEEE Transactions on Industrial Informatics*, 10(1):836–844, 2014.

[39] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk. Efficient and reliable low-power backscatter networks. *ACM SIGCOMM Computer Communication Review*, 42(4):61–72, 2012.

[40] L. Wang, K. Wu, and M. Hamdi. Combating hidden and exposed terminal problems in wireless networks. *IEEE Transactions on Wireless Communications*, 11(11):4204–4213, 2012.

[41] S. Wang, S. M. Kim, Y. Liu, G. Tan, and T. He. Corlayer: A transparent link correlation layer for energy efficient broadcast. In *ACM MOBICOM*, 2013.

[42] X. Wang, L. Fu, X. Tian, Y. Bei, Q. Peng, X. Gan, H. Yu, and J. Liu. Converge cast: on the capacity and delay tradeoffs. *IEEE Transactions on Mobile Computing*, 11(6):970–982, 2012.

[43] Y. Wang, Y. Liu, Y. He, X. Y. Li, and D. Cheng. Disco: Improving packet delivery via deliberate synchronized constructive interference. *IEEE Transactions on Parallel and Distributed Systems*, 26(3):713 – 723, 2015.

[44] Q. Xiang, H. Zhang, J. Wang, G. Xing, S. Lin, and X. Liu. On optimal diversity in network-coding-based routing in wireless networks. In *IEEE INFOCOM*, 2015.

[45] X. Xie, M. Peng, B. Zhao, W. Wang, and Y. Hua. Maximum a posteriori based channel estimation strategy for two-way relaying channels. *IEEE Transactions on Wireless Communications*, 13(1):450–463, 2014.

[46] K. Xu, M. Gerla, and S. Bae. Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks. *Elsevier Ad Hoc Networks*, 1(1):107–123, 2003.

[47] Y. Yan, P. Yang, X. Li, Y. Tao, L. Zhang, and L. You. Zimo: Building cross-technology MIMO to harmonize ZigBee smog with WiFi flash without intervention. In *ACM MOBICOM*, 2013.

[48] P. Yi, A. Iwayemi, and C. Zhou. Developing ZigBee deployment guideline under WiFi interference for smart grid applications. *IEEE Transactions on Smart Grid*, 2(1):110–120, 2011.

[49] L. Zhang and D. Guo. Virtual full duplex wireless broadcasting via compressed sensing. *IEEE/ACM Transactions on Networking*, 22(5):1659 – 1671, 2014.

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

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