

© 2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

X. Liu, Z. Chi, W. Wang, Y. Yao, P. Hao and T. Zhu, "High-Granularity Modulation for OFDM Backscatter," in IEEE/ACM Transactions on Networking, doi: 10.1109/TNET.2023.3286880.

<https://doi.org/10.1109/TNET.2023.3286880>

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

High-Granularity Modulation for OFDM Backscatter

Xin Liu¹, Zicheng Chi², *Member, IEEE*, Wei Wang³, *Member, IEEE*, Yao Yao, Pei Hao, and Ting Zhu⁴

Abstract—Orthogonal frequency-division multiplexing (OFDM) has been widely used in WiFi, LTE, and adopted in 5G. Recently, researchers have proposed multiple OFDM-based WiFi backscatter systems that use the same underlying design principle (i.e., codeword translation) at the OFDM symbol-level to transmit the tag data. However, since the phase error correction in WiFi receivers can eliminate the phase offset created by a tag, the codeword translation requires specific WiFi receivers that can disable the phase error correction. As a result, phase error is introduced into the decoding procedure of the codeword translation, which significantly increases the tag data decoding error. To address this issue, we designed a novel OFDM backscatter called TScatter, which uses high-granularity sample-level modulation to avoid the phase offset created by a tag being eliminated by phase error correction. Moreover, by taking advantage of the phase error correction, our system is able to work in more dynamic environments. Our design also has two advantages: much lower bit error rate (BER) and higher throughput. We conducted extensive evaluations under different scenarios. The experimental results show that TScatter has i) three to four orders of magnitude lower BER when its throughput is similar to the latest OFDM backscatter system MOXcatter; or ii) more than 212 times higher throughput when its BER is similar to MOXcatter. Our design is generic and has the potential to be applied to backscatter other OFDM signals (e.g., LTE and 5G).

Index Terms—Internet of Things (IoT), passive communication, backscatter, OFDM.

I. INTRODUCTION

BY REFLECTING ambient signals, backscatter systems conduct passive communication which can provide low power consumption, low cost, and ubiquitous connectivity for Internet of Things (IoT) devices to support various applications (e.g., smart buildings and smart health). To achieve ubiquitous connectivity and leverage the advantages of the ambient signals, researchers have developed various backscatter systems that reflect ambient signals from TV or frequency

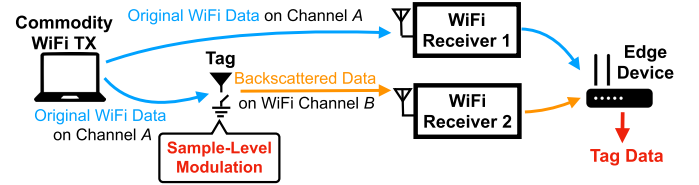


Fig. 1. A general architecture of OFDM-based WiFi backscatter systems [6], [7], [8]. Different from existing systems that modulate the tag data at the symbol-level, our backscatter system uses the sample-level modulation.

modulation (FM) radio towers, LoRa, WiFi, or LTE [1], [2], [3], [4], [5]. Therefore, these backscatter systems are complimentary to each other and have their own unique advantages in terms of energy efficiency, throughput, deployment cost, etc.

In this paper, we mainly focus on the design of the WiFi backscatter due to i) pervasively available WiFi signals inside buildings; ii) numerous WiFi devices; and iii) abundant applications supported by WiFi. All of these can significantly increase the adoption of our developed techniques. The pioneer work on WiFi backscatter [9] achieved up to 1 kbps throughput and 2.1 m range. Follow up works improved the performance of WiFi backscatter by using customized full-duplex access points [10] or standard 802.11 b/g/n devices [6], [7], [8]. Since the majority of the WiFi signals in buildings are using an advanced modulation scheme – OFDM, researchers recently proposed a general architecture of OFDM-based WiFi backscatter systems (shown in Figure 1) to leverage productive data communication from the surrounding WiFi devices. This architecture has demonstrated to be effective in supporting the smart offices application, in which WiFi receivers can be connected by Ethernet backhaul to decode the tag data [6], [7], [8].

However, these systems [6], [7], [8] require specific WiFi receivers that can disable the phase error correction. This is because the phase error correction can eliminate the phase offset created by the tag, and cause incorrect tag data decoding. On the other hand, the phase error correction is very important for the WiFi demodulation because phase errors in the WiFi systems may be dynamically changing due to the changes of environment (e.g., temperature of the oscillators and object movement). Therefore, WiFi protocols always arrange a certain number of pilot subcarriers in each OFDM symbol to track and correct phase errors. Without the phase error correction, the WiFi demodulation error will increase. Similarly, disabling the phase error correction will also introduce the phase error into the demodulation procedure of the codeword translation, which will increase the tag decoding error.

Manuscript received 29 June 2022; revised 18 April 2023; accepted 31 May 2023; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor C. Peng. This work was supported by NSF under grants CNS-2316605, CNS-2305246, and CNS-2127881. (Corresponding author: Ting Zhu.)

Xin Liu and Ting Zhu are with the Department of Computer Science and Electrical Engineering, The Ohio State University, Columbus, OH 43210 USA (e-mail: liu.10663@osu.edu; zhu.3445@osu.edu).

Zicheng Chi is with the Department of Electrical Engineering and Computer Science, Cleveland State University, Cleveland, OH 44115 USA (e-mail: z.chi@csuohio.edu).

Wei Wang is with the Department of Computer Science, Saint Louis University, St. Louis, MO 63103 USA (e-mail: wei.wang.5@slu.edu).

Yao Yao and Pei Hao are with the Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore, MD 21250 USA (e-mail: of90379@umbc.edu; phao1@umbc.edu).

Digital Object Identifier 10.1109/TNET.2023.3286880

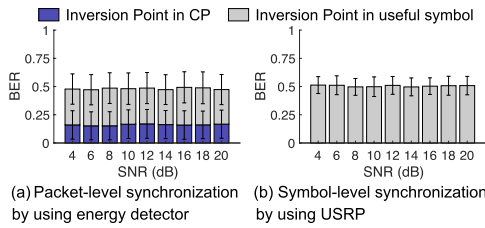


Fig. 2. High BER identified in experiments with a total number of 128,000 data points transmitted.

To demonstrate the limitation of these OFDM-based WiFi backscatter systems, we rebuilt these systems using USRPs that ran the standard 802.11g stack which contains the phase error correction. We identified the high BER which happens even when the signal to noise ratio (SNR) is high (shown in Figure 2). By using a low-power energy detector to measure the signal strength in the air as introduced in previous systems, the tag is synchronized with the WiFi sender at the packet-level. From the result (Figure 2(a)), we can observe that the total BER is around 50% even with a high SNR. Since a tag embeds the tag data by inverting the phase of the incoming signal, the inversion point can settle anywhere in a symbol with packet-level synchronization. After conducting a thorough analysis, we found that one portion of the inversion points (in blue) settle in cyclic prefixes (used to prevent inter symbol interference), which causes decoding failures because the cyclic prefix is removed at the receiver. The other portion of the inversion points (in gray) settle in useful symbols (used to carry real WiFi data), however, this portion will also cause the bit error. We also conducted symbol-level synchronization and found that the blue part is eliminated with a precise synchronization (shown in Figure 2(b)). However, the BER is still very high across different SNRs.

To address this issue, we conducted comprehensive studies and designed a new OFDM backscatter system called TScatter to achieve lower BER and higher throughput. We propose a sample-level modulation scheme, in which the phase offsets on pilot subcarriers are very different from that on the data subcarriers. Thus, the phase offsets on the data subcarriers cannot be eliminated by the phase error correction. Therefore, we can extract the tag data while the phase error correction is present. Moreover, since the phase error correction is present, our system is able to work in more dynamic environments and achieve much lower BER.

Our sample-level backscatter system has a lot of benefits. On one hand, the system provides high reliability allowing tags to work in various scenarios, such as non-line-of-sight or underground. On the other hand, our TScatter can be configured in high throughput mode with Mbps level throughput to support IoT edge computing applications such as ubiquitous surveillance in smart buildings. Further more, high throughput provides high energy efficiency (bit/joule). Given the same amount of harvested energy, our backscatter system can transmit more data than existing ambient WiFi backscatter systems. Moreover, higher energy efficiency also provides another benefit – given the same amount of data to be transmitted, our backscatter system needs much less energy. Therefore, our backscatter system can be deployed further away from

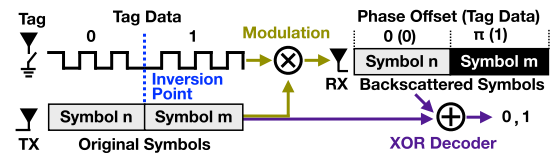


Fig. 3. Overview of the existing OFDM backscatter technique.

the wireless energy transferring sources than the latest WiFi backscatter systems [6], [7], [8].

The main contributions of this paper are as follows:

- We verified existing OFDM-based WiFi backscatter systems and redesigned the system with a novel sample-level modulation technique, which can retrieve tag data from the process of phase error correction in modern OFDM-based wireless communication system. Potentially, this technique can be applied to LTE and 5G.
- To demodulate the high granularity modulated data, we built demodulation models that capture the WiFi demodulation procedure and derive a minimization function to estimate the tag data. We further enhanced our design to support different WiFi modulation schemes. The evaluation results demonstrate the effectiveness of our design.
- We built a hardware prototype of our proposed backscatter system that contains a low power FPGA and a simple RF switch. We also designed a tag IC for estimating the power consumption. Our empirical results show that TScatter has i) three to four orders of magnitude lower BER when its throughput is similar to the latest OFDM backscatter system MOXcatter [8]; or ii) more than 212 times higher throughput when its BER is similar to MOXcatter.

II. BACKGROUND OF EXISTING OFDM BACKSCATTER

To fully reveal the impact of phase error correction on the codeword translation technique, it is necessary to first understand how the technique works. Generally, current OFDM backscatter techniques mainly consist of three parts: (i) the excitation OFDM signal generation on the sender side; (ii) the phase modulation on the tag side; and (iii) the codeword decoding on the receiver side.

The Sender: To produce the OFDM symbol, the sender conducts an Inverse Discrete Fourier Transform (IDFT) on its subcarriers in the frequency domain. The output samples of the corresponding IDFT will form a single OFDM symbol $S(t)$ in the time domain, which can be represented as:

$$S(t) = \text{IDFT} [\mathcal{X}_k] \quad (1)$$

where $\{\mathcal{X}_k\}$ are the subcarriers. Multiple symbols are concatenated to create the final excitation OFDM signal.

The Tag: Existing OFDM-based WiFi backscatter systems allow a tag to convey information by inverting the phase of OFDM symbols in the time domain [6], [7], [8]. An example is shown in Figure 3, the backscatter tag uses zero phase offset to transmit data zero and a phase inversion to transmit data one. Then, the backscattered symbol $B(t)$ is given by:

$$B(t) = \begin{cases} S(t)e^{j0} & \text{Tag data 0} \\ S(t)e^{j\pi} & \text{Tag data 1} \end{cases} \quad (2)$$

To improve the data rate, the tag can create additional phase offsets to convey more information, which is shown below:

$$\mathcal{B}(t) = \begin{cases} S(t)e^{j0} & \text{Tag data 00} \\ S(t)e^{j\frac{\pi}{2}} & \text{Tag data 01} \\ S(t)e^{j\pi} & \text{Tag data 10} \\ S(t)e^{j\frac{3\pi}{2}} & \text{Tag data 11} \end{cases} \quad (3)$$

We note that to achieve above codeword translations (Equation 2 and 3), the tag needs to synchronize the tag data with the symbol. Formally, we define the point that concatenates two different tag data as the tag data **inversion point**. As shown in Figure 3, the tag data inversion point (from 0 to 1) is aligned with the beginning of the OFDM symbol m .

The Receiver: A Discrete Fourier Transform (DFT) is performed to convert the backscattered symbols to the frequency-domain backscattered subcarriers. Because of the linearity of the DFT, the operation of the phase change in the time domain corresponds to the frequency multiplication. Therefore, the backscattered subcarriers can be represented as:

$$\mathcal{X}_k e^{j\delta} = \text{DFT} [\mathcal{B}(t)] \quad (4)$$

where δ is the phase offset. Then, we can decode the tag data by computing the XOR operation of backscattered subcarriers and original subcarriers:

$$\mathcal{X}_k e^{j\delta} \oplus \mathcal{X}_k = \begin{cases} \text{Tag data 0 } \delta = 0 \\ \text{Tag data 1 } \delta = \pi \end{cases} \quad (5)$$

III. WHY EXISTING OFDM BACKSCATTER SYSTEMS DISABLE PHASE ERROR CORRECTION?

The underlying assumption in the codeword translation is that backscatter systems can be free of the effect of the phase error. However, we point out that failure to consider the effect of phase error will significantly increase the BER of OFDM backscatter systems. Therefore, in this section, we first investigate influences of the phase error correction on codeword translation. Then, we extensively analyze all the scenarios that will affect the BER of the codeword translation. At last, we outline the desired properties to reduce the BER of OFDM backscatter systems.

A. Tag Data Is Eliminated

In real-world scenarios, due to the changes of environment, the OFDM receiver is required to perform the phase error correction to eliminate the phase error in the signal. This process also eliminates the phase offset created by the tag.

Specifically, the phase error correction is to use available pilot subcarriers $\{\mathcal{X}_p\}$ to track the phase error. Since the phase error can be simply modeled as a constant multiplicative component across a symbol (e.g., $e^{j\phi}$ [11]), the backscattered subcarrier in Equation 4 should be $\mathcal{X}_k e^{j(\delta+\phi)}$ while the backscattered pilot subcarrier should be $\mathcal{X}_p e^{j(\delta+\phi)}$. Then, the phase error \mathcal{H} can be computed by comparing backscattered pilot subcarriers and original pilot subcarriers:

$$\mathcal{H} = \frac{\sum \mathcal{X}_p e^{j(\delta+\phi)}}{\sum \mathcal{X}_p} = e^{j(\delta+\phi)} \quad (6)$$

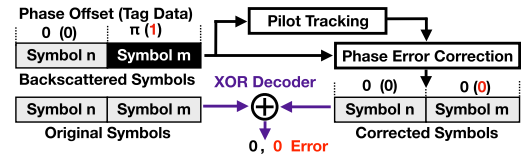


Fig. 4. The phase offset π is eliminated by phase error correction, which causes tag data 1 to be mistakenly decoded as tag data 0.

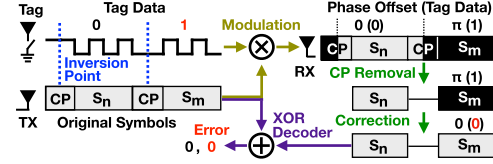


Fig. 5. When the inversion point is located in the CP, the phase offset π representing tag data 1 can still be eliminated.

Finally, the backscattered subcarriers can be corrected as:

$$\mathcal{X}_k e^{j(\delta+\phi)} \cdot \mathcal{H}^{-1} = \mathcal{X}_k \quad (7)$$

The key observation from the above expression is that the phase error $e^{j\phi}$, as well as the phase offsets $e^{j\delta}$ from the codeword translation, are eliminated from subcarriers. In other words, as shown in Figure 4, if the tag transmits arbitrary data, the XOR decoder may never output data 1. This is because the codeword translation makes the pilot subcarriers have the same phase offset as other subcarriers, which makes it feasible to eliminate phase offsets by the phase error correction. As a result, since phase offsets are eliminated, tag data 1 will be mistakenly decoded as 0. The effect of phase error correction will become more severe when the tag increases its data rate. As shown in Equation 3, the tag uses four phase offsets to double the data rate. However, due to the phase error correction, the receiver may not extract these phases offsets correctly. In this case, the tag data 01, 10, 11 will be mistakenly decoded as 00.

The above analysis mainly focuses on the scenario that the inversion point is perfectly aligned with the beginning of the symbol. In following sections, we will show that even when the inversion point is not aligned with the beginning of the symbol, the receiver may still face high BER.

B. Inversion Point in Cyclic Prefix

The XOR decoder cannot extract the tag data correctly when the inversion point is within the cyclic prefix (CP). Because the OFDM receiver will first remove CP to prevent intersymbol interference introduced by the multipath effect, the inversion point in the CP is removed as well. As shown in Figure 5, after the CP removal, although the useful symbol S_m still has the phase offset π , the phase error correction will eliminate the phase offset, which causes the decoding error.

We conduct an experiment to show the effect of the inversion point in the cyclic prefix. In this experiment, an 802.11g OFDM-based WiFi physical layer is implemented on the USRP B210. In order to precisely control the synchronization between the OFDM symbol and the tag data, the tag is connected to the USRP using two wires: one for common ground and another for signal. When the USRP transmits the

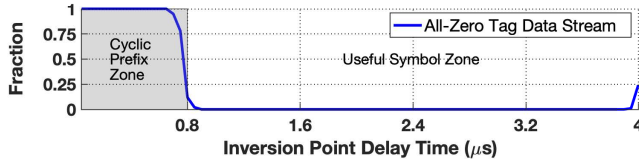


Fig. 6. Experiments show that if the inversion point delay time is within the CP zone, whatever data stream the tag sends will be decoded as all-zero.

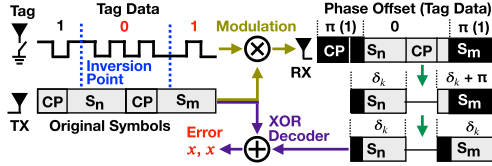


Fig. 7. When the inversion point is located in the useful symbol, the decoding result is the same and thus causes the decoding error.

WiFi signal to the air, it will also transmit the starting signal through the wire to tell the tag to embed tag data in the WiFi signal. By modifying the delay after the starting signal, we can synchronize the tag data inversion point with the different timings in the OFDM symbol duration ($4\mu\text{s}$). The original tag data is an arbitrary data stream. However, as shown in Figure 6, when the inversion points are within the cyclic prefix zone, the decoded tag data are all zeros, because the phase offsets of the tag data 1 are eliminated. When the inversion points are within the useful symbol zone, the decoded tag data is not all zeros. However, as we will discuss in Section III-C, it may still have a high BER.

C. Inversion Point in Useful Symbol

In this section, we discuss the case where the inversion point lies within the useful symbol. As depicted in Figure 7, the phase offsets in symbols S_n and S_m are counter to each other, resulting in opposing phase offsets δ_k and $\delta_k + \pi$ on the subcarriers, where k is the subcarrier index. Notably, the π in $\delta_k + \pi$ is a common phase offset among the S_m subcarriers, which can be removed by phase error correction. Once removed, each S_m subcarrier is left with only δ_k , aligning the phase offsets of S_n . Consequently, they share the same decoding result, causing the errors. It should be noted that δ_k varies across subcarriers, causing the decoding result x to be either 1 or 0.

D. Desired Properties and Our Solution

From the analysis in Section III-B and III-C, the OFDM backscatter should satisfy the following two properties to avoid the decoding error caused by the phase error correction:

- 1) The tag data inversion point should fall into the useful symbol.
- 2) The tag data should be represented by phase offsets that cannot be eliminated.

Since the backscatter tag is a low power device, we cannot use a high power circuit to conduct precise synchronization between the inversion point and the useful symbol. To overcome this challenge, we explore a high granularity modulation scheme, sample-level modulation. By doing this, the phase offsets on the pilot subcarriers are different from that on the

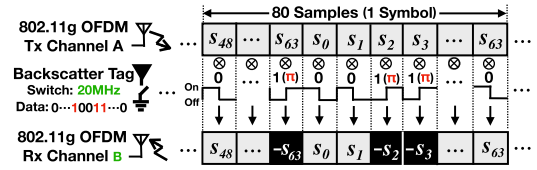


Fig. 8. When a switching cycle is equal to a sample duration, the switching action can appropriately change the phase of the sample.

data subcarriers. Thus, the phase offsets on the data subcarriers cannot be eliminated by the phase error correction.

To satisfy the second property, we utilize orthogonal coding technique for reliable data transfer. Specifically, to reduce the BER and achieve high reliability, we propose orthogonal pseudo-random noise (PN) sequences to represent the tag data. With the help of the sample-level modulation, each data bit in the PN sequence can represent a phase offset. As a result, the sequences of phase offsets are also orthogonal and the tag data can avoid being eliminated.

IV. SAMPLE-LEVEL MODULATION

Our sample-level modulation has two design goals: i) embedding the tag data in the OFDM signal at the sample-level, and ii) minimizing the interference from the original band channel. In order to achieve the first design goal, we propose to use the tag to change the phase of any sample in the OFDM symbol. To achieve the second design goal, the tag needs to shift the center frequency of the backscattered samples to the adjacent band channel. Figure 8 shows how TScatter leverages the switching action to change the phase of the sample. For the sake of simplicity, we first assume that the on and off edges of switching actions are aligned with the samples. We discuss the misalignment situation in Section VI-A.

A fundamental basis for our sample-level modulation is that the OFDM symbol can be divided into the sample-level, whose duration is much shorter than the symbol-level. When 802.11g OFDM generates a symbol in the transmitter, the IDFT converts the 64 subcarriers into a 64-sample sequence. The samples can be represented by rewriting Equation 1 with the definition of IDFT:

$$S_n = \text{IDFT} [X_k] = \frac{1}{64} \sum_{k=0}^{63} X_k e^{j2\pi kn/64} \quad (8)$$

where, $n \in \{0, \dots, 63\}$, S_n denotes the n 'th sample and X_k denotes the k 'th subcarrier. To create a CP, the last 16 samples $[S_{48}, \dots, S_{63}]$ are replicated in front of the 64 samples. Hence, a symbol comprises 80 samples.

When the tag toggles its RF switch to backscatter the OFDM signals, it essentially uses square waves to modulate the phases of the samples. We use θ_n to represent the initial phase of the n 'th square wave \mathcal{W}_n . Then, \mathcal{W}_n can be represented using a Fourier series as follows [12]:

$$\mathcal{W}_n(f_s, \theta_n) = 0.5 + \frac{2}{\pi} \sum_{m=1,3,5,\dots}^{\infty} \frac{1}{m} \cos(2\pi m f_s t + \theta_n) \quad (9)$$

where, f_s is the toggling frequency of the switch.

Since each symbol is $4\mu\text{s}$ long and contains 80 samples, when the tag toggles its switch at $f_s = 20\text{ MHz}$, the switching cycle ($0.05\mu\text{s} = \frac{1}{20\text{ MHz}}$) will be equal to the sample duration ($0.05\mu\text{s}$). Therefore, the switching action can appropriately modulate the phase of the sample. The n 'th backscattered sample can be calculated by multiplying the n 'th sample and the 1st harmonic of the n 'th square wave:

$$\begin{aligned} S_n \mathcal{W}_n^{m=1} &= \frac{2}{\pi} S_n \cos(2\pi f_s t + \theta_n) \\ &= \frac{1}{\pi} \{ S_n e^{j\theta_n} e^{j2\pi f_s t} + S_n e^{-j\theta_n} e^{-j2\pi f_s t} \} \quad (10) \end{aligned}$$

Summary: From Equation 10, one can observe that our two design goals are achieved. First, there are two backscattered samples $S_n e^{j\theta_n}$ and $S_n e^{-j\theta_n}$. They are formed by changing the phase of the original sample S_n by θ_n and $-\theta_n$, respectively. If we change θ_n according to the tag data, the tag data is embedded into the sample. For example, if the tag wants to use the 4-phase scheme ($0, \pi/2, \pi$ and $3\pi/2$) to transmit the tag data, we can define the tag data '00' as $\theta_n = 0$, '01' as $\theta_n = \pi/2$, '10' as $\theta_n = \pi$ and '11' as $\theta_n = 3\pi/2$. Second, the two backscattered samples are multiplied by $e^{j2\pi f_s t}$ and $e^{-j2\pi f_s t}$, respectively, which means the backscattered samples are shifted by $f_s = \pm 20\text{ MHz}$ into the adjacent band channels and thus isolated from the original band channel. Therefore, an OFDM receiver can obtain the backscattered samples without the interference from the original band channel [13]. For these two sidebands, one is desired and the other is unwanted and wasted. The unwanted sideband can be easily eliminated by making the signal have a negative copy on the unwanted sideband as introduced in HitchHike [14].

V. TAG CODING SCHEME

In this section, we discuss the coding methods on the tag, which aim to avoid the inversion point being removed by the cyclic prefix removal and the phase offset being eliminated by the phase error correction, and make TScatter achieve highly reliable backscatter communication.

A. Surviving From Cyclic Prefix

Since the receiver removes the cyclic prefix (i.e., the first 16 samples of each symbol) directly prior to the procedures in the OFDM module, a portion of the tag data may be removed as well because the tag may embed this data in the cyclic prefix. Figure 9(a) shows an example that the tag data d_1 to d_{16} are lost due to the CP removal.

A naive solution is to utilize an envelope detector to detect the start of the cyclic prefix and embed the tag data in the useful symbol. However, it is unlikely to obtain an accurate CP detection using a packet-level detection on the low-power envelope detector. This is because achieving sample-level synchronization accuracy with a low-power envelope detector approach can be challenging due to reduced sensitivity as bandwidth increases [15]. Instead, TScatter simply embeds 40 bits of tag data twice on 80 samples of a symbol (shown in Figure 9(b)). After the CP removal, there still exists a copy of 40 bits of data. By doing this, the tag does not have to

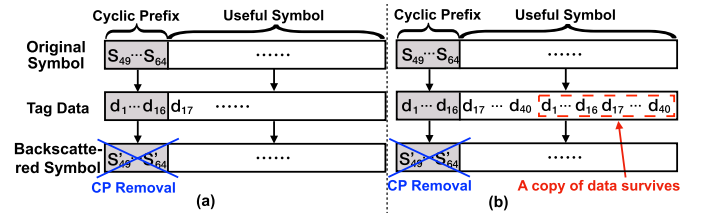


Fig. 9. In (a), the tag data in the CP is lost due to its removal. In (b), the tag double-embeds 40 bits of data, ensuring data survival post-CP removal.

rely on the synchronization accuracy of the envelope detector to avoid the CP. This approach can be easily implemented on the tag without extra energy cost or computing resources.

At the receiver side, to decode the 40 bits of tag data in each symbol, we need to know which sample in the symbol is the starting sample of the 40-sample sequence. To find the starting sample, we designed a header (shown in Section VI-B.1) with a duration of two symbols. The header is predefined as a flag sequence. Despite part of the flag sequence being removed by the CP removal, a backscattered symbol remains, modulated solely by the flag sequence. We incorporate a sliding-window algorithm into the decoding process (see Section VI-C) to identify the best-matching flag sequence segment for this symbol's decoding. In addition, we adopt a strategy inspired by the block-pilot symbol method, which involves repeatedly sending the predefined header after a certain number of symbols to re-identify the starting tag modulation sample. If the discrepancy between consecutive estimations is greater than one sample, we utilize the multiple values obtained from these estimations to correct the erroneous one, ensuring the accurate identification of the starting tag modulation sample throughout the transmission process. This method precisely locates the starting sample, enhancing TScatter system's performance and reliability. When the decoding matches the predefined two-symbol trailer, it marks the tag data's end.

B. Coding Scheme for Low BER

To avoid the phase offsets being eliminated, TScatter uses predefined nearly orthogonal pseudo-random noise (PN) sequences (shown in Table I) to represent the tag data. As shown in Figure 10, the tag data is divided into multiple groups. Each group will be spread to a special 40-bit long PN sequence. Then each bit in the PN sequence is transmitted by using the sample-level modulation. With the help of the sample-level modulation, each bit in the PN sequence can represent a phase offset. As a result, the sequences of phase offsets are orthogonal to each other and the tag data can avoid being eliminated. As described in Section V-A, to avoid being removed by the CP removal, each sequence is transmitted twice. To reduce the coding complexity on the tag, we implant a lookup table of the sequences to map the tag data to the corresponding PN sequence.

Utilizing orthogonal PN sequences to represent tag data offers three advantages. First, they boost the effective SNR of backscattered signals at the receiver due to their orthogonality. Second, the predefined sequences lower computational complexity, enabling direct sequence matching instead of

TABLE I
MAP TAG DATA TO PN SEQUENCE

No.	Tag Data	PN Sequence
1	0000	01101001111101101100110011110101001010
2	0001	1001010101110100010000010111110001010
3	0010	10001010101001101010011111001010011000
4	0011	111100000100011101011101010000010001111
5	0100	0010100001000100100011011001111011110111
6	0101	1101011101101000110001111000011001100001
7	0110	0101110011101010011010000100100101110101
8	0111	001001111110100101101100110000100110100
9	1000	0101001101110100011100000101011101011100
10	1001	11001100110100101101001100110001100111100
11	1010	1010001110111001010101010011000100011110
12	1011	1110101111001001100111011100111010011001
13	1100	0001001011001011010011101111010011100001
14	1101	1000110101100111000001001110100001110111
15	1110	01100110011001011010110010001101100011
16	1111	001111000111010011101010000111100100010

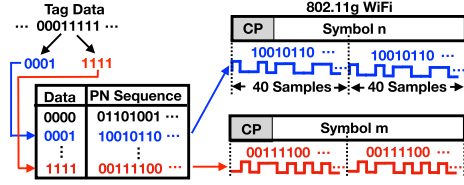


Fig. 10. TScatter uses PN sequences to represent the tag data.

estimation in decoding. Lastly, they allow TScatter's use in lower order OFDM modulation schemes like BPSK or QPSK (see Section VI-C).

C. Coding Scheme for High Throughput

High throughput coding combines convolutional code and predefined data for higher order modulation schemes like 64QAM or 16QAM. The convolutional code is a 2/3 rate feed-forward encoder. Predefined data boosts estimation accuracy by restricting the estimation result to a smaller range during decoding (detailed in Section VI-C). For 64QAM, of the 40-sample sequence, 32 samples embed convolutional coded tag data, and the remaining 8 are modulated by predefined data. For 16QAM, 18 samples embed convolutional coded tag data and 22 embed predefined data.

VI. DECODING OF TAG DATA

One benefit of our sample-level modulation is that it does not require an accurate synchronization with the OFDM sender because the phase offset caused by a lack of accurate synchronization can be corrected by the phase error correction. In this section, we first analyze the phase offset, then describe the OFDM demodulation model and mathematically demonstrate how the phase offset is corrected by the OFDM receiver. In the end, we describe how to decode the tag data.

A. Phase Offset Analysis

The phase offset can be characterized by three factors:

- *Frequency Offset*. In the WiFi receiver, the sample received at frequency f_c is down-converted to baseband using a local

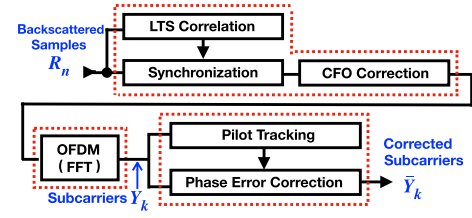


Fig. 11. The demodulation process in 802.11g WiFi receiver from the backscattered samples to the corrected subcarriers.

carrier frequency. The offset between f_c and the local frequency results in two effects: carrier frequency offset (CFO) and sampling frequency offset (SFO). CFO introduces a α_{cfo} phase offset to samples during down-conversion, while SFO adds a α_{sfo} phase offset to OFDM subcarriers post-DFT [16].

- *Envelope Detection Delay*. The tag uses an envelope detector to detect WiFi transmission, but uncertainty in delayed response prevents exact identification of sample start. Consequently, switching actions and samples aren't edge-aligned. This can be attributed to a time delay τ causing a phase offset $\beta = 2\pi f_s \tau$ [17].
- *Wireless Channel*. The wireless channel introduces a phase offset of ϕ to the samples.

Thus, a backscattered sample can be expressed as follows:

$$\mathcal{R}_n = e^{j\alpha_{cfo}} e^{j(\beta+\phi)} e^{j\theta_n} \mathcal{S}_n \quad (11)$$

In Sections VI-B.1 and VI-B.2, we show that CFO and SFO-induced phase offsets can be estimated and corrected. Section VI-B.3 demonstrates that phase offsets from envelope detection delay and wireless channel can be eliminated, not affecting data subcarrier values.

B. Demodulation Model

TScatter's demodulation aims to decode tag data on OFDM subcarriers. To comprehend how the OFDM receiver translates backscattered samples into subcarriers, we dissect the 802.11g receiver's workflow into three sections (depicted in Figure 11), for which we construct models.

1) *Synchronization & CFO Correction*: In this section, we construct a mathematical model for synchronization and CFO correction in Figure 11. Upon recognizing WiFi transmission, the receiver uses the preamble to synchronize symbol timing and correct CFO. Thus, we begin by examining how the tag synchronizes the WiFi sender and receiver.

Our synchronization solution is depicted in Figure 12. The preamble includes two long training sample sequences, each composed of 64 samples. By identifying two peak correlations between the received samples and training pattern, the WiFi receiver pinpoints the symbol's start. To maintain training sample values, TScatter solely reflects the preamble using continuous square waves upon detecting WiFi transmission, refraining from embedding data. After square waves, TScatter begins modulating WiFi samples.

One may ask whether the phase offset (described in Section VI-A) will change the preamble value, therefore affects the synchronization. We note that TScatter does not

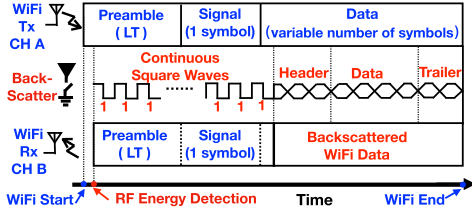


Fig. 12. Synchronization between the WiFi sender and the WiFi receiver. The backscatter packet contains four parts: Continuous Square Waves (CSW), Header (in Section V-A), Data and Trailer (in Section V-A). The CSW, a series of square waves, reflects the preamble without embedding tag data.

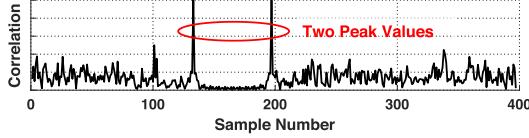


Fig. 13. By obtaining the indexes of two peak values, the WiFi receiver can achieve a precise synchronization.

change the preamble sample's value. Without loss of generality, let us assume the phases of continuous square waves are $\theta_n = 0$ and the training pattern value is \overline{LT}_i . Then, the positions of two peak values of the correlation between the received samples \mathcal{R}_l and the training pattern \overline{LT}_i are:

$$\begin{aligned} (l_1, l_2) &= \underset{l}{\operatorname{argmax}_2} \left\| \sum_{i=0}^{63} (\mathcal{R}_{l+i} * \overline{LT}_i) \right\|_2 \\ &= \underset{l}{\operatorname{argmax}_2} \left\| e^{j(\beta+\phi)} e^{j*0} \sum_{i=0}^{63} (\mathcal{S}_{l+i} * \overline{LT}_i) \right\|_2 \\ &= \underset{l}{\operatorname{argmax}_2} \left\| \sum_{i=0}^{63} (\mathcal{S}_{l+i} * \overline{LT}_i) \right\|_2 \end{aligned} \quad (12)$$

where, argmax_2 provides the two sample numbers (l_1, l_2) for two peak correlation values.

Equation 12 proves that the positions of two peak correlation values are only related to the original sample values \mathcal{S}_l and are not affected by the phase offset. Therefore, the phase offset will not affect the synchronization. Our experimental result (shown in Figure 13) also illustrates that when the tag uses continuous square waves to reflect the WiFi signals, it does not interfere with the positions of two peak correlation values, and the receiver can achieve a precise synchronization.

In the WiFi receiver, the received sample at frequency f_c is down-converted to baseband with a local carrier frequency $(1+\varepsilon)f_c$. At the receiver side, the frequency offset εf_c causes two phenomenas: CFO and SFO. The CFO introduces an extra phase offset of $\alpha = 2\pi\varepsilon\frac{f_c}{f_c}$ on the samples during each down-conversion procedure. When the l 'th sample of the frame is down-converted, the phase offset applied to the sample is accumulated to $\alpha_{cfo} = l\alpha$. Equation 11 can be rewritten as:

$$\mathcal{R}_{l,n} = e^{jl\alpha} e^{j(\beta+\phi)} e^{j\theta_n} \mathcal{S}_n \quad (13)$$

where l and n point to the same sample. Since a WiFi frame contains multiple symbols, l is the sample number in the frame, while n is the sample number in the symbol.

Since the tag does not affect the synchronization, the CFO can be estimated accurately. An estimation of the CFO can be obtained simultaneously when the synchronization is achieved.

According to the WiFi specification [18], the WiFi receiver uses LT samples to estimate the CFO. The starting LT sample number is computed as l_1 . Let us denote the t 'th LT sample as \mathcal{R}_{l_1+t} . The CFO estimator is given by:

$$\alpha = \frac{1}{64} \angle \left(\sum_{t=0}^{63} \mathcal{R}_{l_1+t}^* \mathcal{R}_{l_1+t+64} \right) \quad (14)$$

To correct the CFO, the received sample $\mathcal{R}_{l,n}$ is multiplied by $e^{-jl\alpha}$. Through the synchronization, the WiFi receiver learns the sample number l . Thus, the received sample can be updated to the corrected sample \mathcal{S}_n :

$$\mathcal{R}_n = \mathcal{R}_{l,n} * e^{-jl\alpha} = e^{j(\beta+\phi)} e^{j\theta_n} \mathcal{S}_n e^{-jl\alpha} e^{-jl\alpha} \quad (15)$$

Equation 15 shows that the impact of CFO is canceled by the CFO correction module in the WiFi receiver.

2) *Modeling OFDM and SFO Correction*: The 80 backscattered samples of each symbol first go through the CP removal module, where the first 16 samples are removed. Then the remaining 64 samples are applied to the DFT to generate the complex values of OFDM subcarriers [19]. The subcarrier's value can be derived as:

$$Y_k = \text{DFT} [\mathcal{R}_n] = e^{j(\beta+\phi)} \sum_{n=0}^{63} e^{j\theta_n} \mathcal{S}_n e^{-j2\pi nk/64} \quad (16)$$

where, $k \in \{0, \dots, 63\}$ and Y_k is the complex value of the k 'th OFDM subcarrier. Equation 16 shows that the tag data is transferred from the backscattered samples in time domain to the subcarriers in frequency domain by OFDM.

Like the CFO, an SFO also exists, which causes a phase offset to OFDM subcarriers after the DFT:

$$\alpha_{sfo} = -2\pi\varepsilon r k \frac{N + N_g}{N} \quad (17)$$

where, $N = 64$ is the DFT size, $N_g = 16$ is the cyclic prefix length, r is the symbol number which is learned through the synchronization, k is the subcarrier number and ε can be calculated from the estimated CFO ($\varepsilon = \frac{\alpha_{cfo}}{2\pi f_c}$). The OFDM subcarriers are corrupted as $Y_k e^{j\alpha_{sfo}}$.

From Equation 17, we know the key to correct the SFO is to estimate CFO accurately. In Section VI-B.1, we have demonstrated the tag does not change the preamble value so that the receiver can achieve a precise CFO estimation. The output subcarriers are multiplied by $e^{-j\alpha_{sfo}}$ (i.e., $Y_k = Y_k e^{j\alpha_{sfo}} e^{-j\alpha_{sfo}}$) to correct the offset.

3) *Modeling Phase Error Correction*: While the tag data is transferred into the subcarriers, the phase offset $\beta + \phi$ is also transferred into the subcarriers (shown in Equation 16), which may increase the tag data decoding error. We had an important observation that the OFDM receiver can leverage pilot subcarriers to eliminate the phase offset. In this section, we mathematically demonstrate the procedure.

The phase error estimation is performed by calculating the rotating phase of the four pilot subcarriers. In the sender, the four pilot subcarriers are $\{\mathcal{X}_{11}, \mathcal{X}_{25}, \mathcal{X}_{39}, \mathcal{X}_{53}\}$ and their initial values are $\{1, 1, 1, -1\}$ [20]. In the receiver, we use $\{Y_{11}, Y_{25}, Y_{39}, Y_{53}\}$ (shown in Equation 16) to represent

the four pilot subcarriers. The rotating angle of the pilot subcarriers can be estimated as:

$$\begin{aligned}\Psi &= \angle(Y_{11} + Y_{25} + Y_{39} - Y_{53}) - \angle(\mathcal{X}_{11} + \mathcal{X}_{25} + \mathcal{X}_{39} - \mathcal{X}_{53}) \\ &= \beta + \phi + \angle\Gamma\end{aligned}\quad (18)$$

where,

$$\Gamma = \sum_{n=0}^{63} e^{j\theta_n} \mathcal{S}_n \left(\sum_{k=11,25,39} e^{-j2\pi nk/64} - e^{-j2\pi n \frac{53}{64}} \right) \quad (19)$$

From Equation 18 and 19, we observe that a new phase offset $\angle\Gamma$ is introduced due to the backscatter modulation on the pilot subcarriers. Equation 18 also illustrates that although the backscatter modulation change the pilot subcarriers, the phase offset $\beta + \phi$ is isolated from the tag data.

Since 48 out of the 64 subcarriers are used to carry the payload data ($k \in \{6, \dots, 58\}$ and $\notin \{11, 25, 32, 39, 53\}$), the OFDM receiver multiplies 48 data subcarriers by the conjugate of rotating phase $e^{-j\Psi}$ to correct the phase offset:

$$\begin{aligned}\bar{Y}_k &= e^{-j\Psi} Y_k \\ &= e^{-j\angle\Gamma} \underbrace{e^{-j(\beta+\phi)}}_{\cancel{e^{-j(\beta+\phi)}}} \underbrace{e^{j(\beta+\phi)}}_{\cancel{e^{j(\beta+\phi)}}} \sum_{n=0}^{63} e^{j\theta_n} \mathcal{S}_n e^{-j2\pi nk/64}\end{aligned}\quad (20)$$

where,

$$e^{-j\angle\Gamma} = \frac{\text{Re}\{\Gamma\} - j * \text{Im}\{\Gamma\}}{\sqrt{\text{Re}\{\Gamma\}^2 + \text{Im}\{\Gamma\}^2}} \quad (21)$$

Equation 20 demonstrates that the phase offset $\beta + \phi$ is eliminated and does not affect the tag data decoding. On the other hand, Equation 20 also demonstrates that the values of data subcarriers were affected twice: The first one happens during the backscatter modulation stage, when the values of the data subcarriers were changed according to the tag data; the second one happens during the phase error correction stage, when the values of data subcarriers were corrected by the phase offset calculated from pilot subcarriers. Although Γ exists in Equation 20, Equation 19 shows that the only unknown value in Γ is the tag data. Therefore, we can leverage Equation 20 to decode the tag data without being affected by the phase offset $\beta + \phi$.

C. Decoding Tag Data

Equation 20 shows that the phase change on each subcarrier is different and determined by all 64 backscattered samples. If the tag data on any backscattered sample change, the phase change on each subcarrier changes. Therefore, we must use all subcarriers' information to decode each tag data.

We observe that the inputs of Equation 20 are the tag data θ_n and the original OFDM sample \mathcal{S}_n , while the outputs of Equation 20 are the corrected data subcarrier \bar{Y}_k . Hence, if we obtain the values of \mathcal{S}_n and \bar{Y}_k , we can decode θ_n .

The values of original OFDM sample \mathcal{S}_n can be calculated by using Equation 8. However, the challenge is that we cannot obtain the real value of \bar{Y}_k . Normally, OFDM backscatter uses the coded data linear transform technique in [6], [7], and [8] to track the phase change on the subcarriers. If we follow the same technique to get the subcarrier values, we find that the

subcarrier values deviate from \bar{Y}_k . This is because the OFDM receiver maps \bar{Y}_k to the nearest QAM points (we assume \vec{Y}_k).

Since \vec{Y}_k are the nearest QAM points of \bar{Y}_k , we can estimate the tag data θ_n by calculating the minimum Euclidean distance between the corrected data subcarrier values \bar{Y}_k and their mapped points \vec{Y}_k :

$$\begin{aligned}\begin{bmatrix} \tilde{\theta}_0 \\ \vdots \\ \tilde{\theta}_{63} \end{bmatrix} &= \arg \min_{[\theta_0, \dots, \theta_{63}]} \sum_{\substack{k=6 \\ k \neq 11, 25, 32, 39, 53}}^{58} \|\vec{Y}_k - \bar{Y}_k\|_2 \\ &= \arg \min_{[\theta_0, \dots, \theta_{63}]} \sum_{\substack{k=6 \\ k \neq 11, 25, 32, 39, 53}}^{58} \|\vec{Y}_k - e^{-j\angle\Gamma} \sum_{n=0}^{63} e^{j\theta_n} \mathcal{S}_n e^{-j2\pi nk/64}\|_2\end{aligned}\quad (22)$$

where, k are the indexes of the data subcarriers, \vec{Y}_k are provided by the receiver and Γ can be represented using Equation 19. The only unknown value in Equation 22 is the tag data θ_n . Therefore, we can leverage Equation 22 to decode the tag data. We note that when OFDM employs BPSK or QPSK, the QAM mapping points become sparser. In high throughput mode, utilizing the calculation of the minimum Euclidean distance to infer multiple unknown tag data can result in higher BER. In such cases, it is recommended to use Low BER mode, as this mode only has a few known predefined PN sequences representing fixed tag data. Furthermore, in Low BER mode, the process involves directly finding the best-match sequence rather than estimating the sequence. This approach, combined with using a single PN sequence to modulate multiple WiFi symbols to transmit one tag data, can further mitigate the errors caused by the sparsity of QAM mapping points.

Since each tag data θ_n has limited values (i.e., either 0 or π), Equation 22 is a constrained linear least-squares problem. We note that the left side of Equation 22 is 64 unknown values $(\tilde{\theta}_0, \dots, \tilde{\theta}_{63})$, while we only have 48 data subcarrier values \bar{Y}_k ($k \in \{6, \dots, 58\}$ and $\notin \{11, 25, 32, 39, 53\}$) on the right side. That means Equation 22 is not full row-rank. Mathematically, to determine a unique solution using Equation 22, we need to minimize the number of unknown values on left side to 48. A simple way to do this is that the number of unknown tag data in each symbol is no more than 48 and the remaining tag data are predefined. For example, we predefine $e^{j\theta_{48, \dots, 63}} = e^{j0} = 1$ and calculate the unknown data $e^{j\theta_{0, \dots, 47}}$. Therefore, the left side (the unknown tag data) and the right side (the data subcarrier values) have the same size, and Equation 22 is now full row-rank. In fact, Equation 22 illustrates that the maximum number of samples used to store the unknown tag data in each symbol is 48. This is because we rely on the data subcarriers' values to calculate the unknown tag data and each symbol provides only 48 data subcarriers.

Equation 22 is solved by using the matrix decomposition, whose computational complexity is $O(n^3)$, where n is the number of the unknown tag data. When n gets to the maximum value (i.e., 48), there are $n^3 = 110592$ floating-point operations. A low power edge computing platform Jetson Nano [21] (only cost \$99) implemented on an ARM A57

processor with 472G floating-point operations per second, can solve the problem in $\frac{110592}{472 \times 10^9} = 0.25 \mu\text{s}$, which is less than a symbol duration $4 \mu\text{s}$.

D. Decoding Copied Data

The coding scheme in Section V-A necessitates that no more than 40 tag data are embedded in each symbol. Since Equation 22 supports solving up to 48 unknowns, it can naturally handle the 40 unknowns. It is worth noting that in Section V-A, we utilized an example with two copies of data within one symbol to briefly explain the surviving from cyclic prefix. However, we do not mandate that these two copies of data are synchronized with the WiFi symbol, meaning they don't have to fall within the same symbol. Due to tag's inability to achieve fine synchronization with WiFi, the first copy of data may span across two symbols. Nevertheless, we can still decode the data. If we can decode the portion of data in the previous symbol, the copy of this portion in the current symbol becomes known. In this case, within the current symbol, there are still only 40 unknowns, and none of them are lost due to the cyclic prefix removal. Since each group of data has a portion that spans across the previous symbol, the key to solving this non-synchronous issue is to decode the first group of data that spans across the corresponding previous symbol. This group of data is adjacent to the predefined header. Consequently, in the first symbol containing unknowns, there will be either a maximum of 40 unknowns or fewer, with the remaining data being part of the predefined header, making decoding possible. Once this portion of data is decoded, the rest of the data can be decoded sequentially.

VII. LITE VERSION OF TSCATTER

In prior works [6], [7], [8], the codeword translation works only when the phase error correction is disabled. In this section, we demonstrate that TScatter can also use the codeword translation at the symbol level without disabling the phase error correction and introduce the lite version of TScatter.

Prior works use the symbol-level modulation (i.e., phase change once per symbol) to realize the codeword translation. Based on our observation from Section III-C, when the inversion point settles in the useful symbol, the corrected subcarriers are different from the original ones. Since the symbol-level modulation is essentially a special case of the sample-level modulation, it is possible to use the codeword translation to achieve the basic idea of TScatter without disabling the phase error correction. We call this approach Lite TScatter because it can be implemented efficiently.

Specifically, the whole procedure of Lite TScatter is as follows: before the tag transmits the data, it first transmits a long all-one (or all-zero) sequence to reflect the preamble and several symbols. Since there is no phase change in the long sequence, the corrected subcarriers are the same as the original ones. Then, the tag transmits the first data. The first data is predefined, and its value should be different from that of the long sequence. For example, if the long sequence is all-one, the first data should be defined as zero. Since there is

a phase change between the long sequence and the first data, we can find the first symbol which is different from the original one. Next, the tag can transmit arbitrary data. If the corrected subcarriers are the same as the original ones, it implies there is no phase change and the tag data should be equal to the last one. If the corrected subcarriers are different from the original ones, it implies there is a phase change and the data should be different from the last one. Therefore, we can decode the tag data one by one. By modifying the codeword translation decoder in Equation 5, the decoding function of Lite TScatter is as follows:

$$\text{Tag data } d_n = \begin{cases} d_{n-1} & \mathcal{X}_k e^{j\delta_k} \oplus \mathcal{X}_k = 0 \\ 1 - d_{n-1} & \mathcal{X}_k e^{j\delta_k} \oplus \mathcal{X}_k \neq 0 \\ \text{Predefined } n = 1 \end{cases} \quad (23)$$

Equation 23 illustrates that Lite TScatter reuses the existing OFDM backscatter systems, retaining the symbol level modulation and XOR-based decoding. This makes it possible to be implemented on commodity devices. However, it approaches the OFDM backscatter communication from a sample level modulation perspective, resulting in a different decoding algorithm compared to existing OFDM backscatter systems. Existing OFDM backscatter systems necessitate disabling phase error correction, decoding based on whether a symbol's phase rotation is 0 or 180 degrees. However, as phase errors can quickly accumulate, decoding inaccuracies become inevitable when the phase rotation deviates from the expected values. On the other hand, Lite TScatter allows phase error correction, keeping phase error consistently zero. Its decoding process determines if the phase rotation is 0 or non-zero degrees, a significantly broader range compared to 0 or 180 degrees, thereby reducing the BER.

Lite TScatter's BER is higher than the low BER mode of TScatter (explained in Section V-B), as it only has one inversion point every 4 WiFi symbols, potentially not affecting subcarrier phase if near or within the cyclic prefix zone. To lower the BER, we can add more inversion points over a certain duration, ensuring phase change. For instance, using one PN sequence from Table I to modify the symbol when transmitting '1', and an all-zero sequence for '0'. This enables effective XOR operations, still making Lite TScatter feasible for commodity WiFi devices.

VIII. ACCESS CONTROL

Our TScatter system is built upon the open-source backscatter platform HitchHike [14], which provides solutions for addressing WiFi congestion, scaling to multiple tags, and waking up by ambient signals through its access control mechanisms. TScatter inherits and aligns with its access control mechanisms.

Specifically, in TScatter's communication process, the WiFi transmitter first secures the wireless channel by sending RTS-to-CTS messages. These messages secure two WiFi channels, one for normal WiFi transmission and another for backscatter. This approach eliminates the need for the tag to sense and occupy the wireless channel, as it would consume too much power. Once the channel is secured, the WiFi transmitter

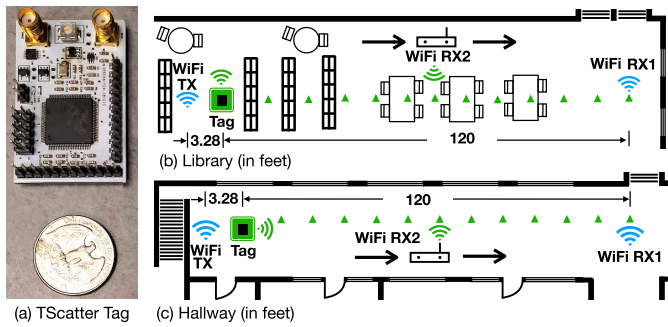


Fig. 14. Implementation and experimental setup.

can wake up and synchronize with the tag. To achieve synchronization, the WiFi transmitter sends a series of short WiFi packets in predetermined packet slots. The presence or absence of packets in these predefined slots represents the data transmitted to the tag using On-Off Keying (OOK) modulation. By employing a low-power envelope detector, the tag can identify the sequences of ones and zeros transmitted by the WiFi transmitter. When the tag decodes a predefined sequence from the WiFi transmitter, it knows it can embed backscattered bits onto the next incoming WiFi packet. By leveraging this access control mechanism, TScatter can effectively avoid WiFi congestion, scale up to multiple tags, and prevent waking up due to ambient signals.

IX. EVALUATION

In this section, we present the implementation of TScatter and show the experimental results of our extensive evaluation.

A. Implementation

TScatter (Figure 14(a)) is implemented on an open-source backscatter platform [22], comprising a Microsemi AGLN250 low power FPGA and an ADG902 RF reflective switch. For comparison purposes, we've implemented two TScatter variants: Lite and Full. Lite TScatter extends existing OFDM backscatter systems, maintaining symbol level modulation and XOR-based decoding. Full TScatter introduces sample level modulation and employs minimum Euclidean distance calculation for decoding QAM mapping data. It has two sub-modes: Low BER and High Throughput. The Low BER mode uses **binary** and **nibble** (4-bit) coding schemes (Section V-A), and High Throughput mode uses **2-phase** and **4-phase** modulation schemes (Section IV).

To conduct fair comparisons, the distance between the WiFi transmitter and the backscatter tag is 3.28 feet (i.e., 1 m), which is the same as the one in FreeRider [7]. The WiFi transmitter is implemented on a ThinkPad T420s laptop which transmits the IEEE 802.11g compatible signals [23]. The WiFi transmitter utilizes 64QAM as its modulation scheme by default. In Section IX-G, we evaluate TScatter's performance with different WiFi modulation schemes.

To extensively evaluate TScatter's performance, the experiments were conducted in three different scenarios: **Library** (Figure 14(b)), **Hallway** (Figure 14(c)), and **Stadium**. The WiFi receivers are implemented using DELL XPS 9550 laptop

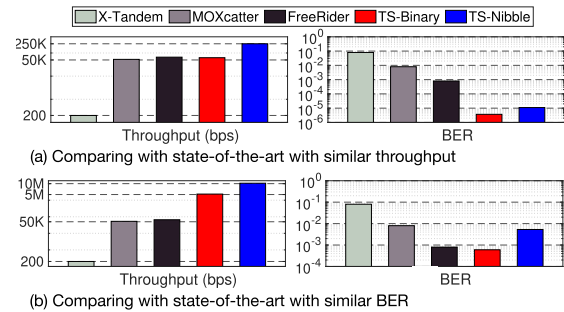


Fig. 15. Comparing with state-of-the-art.

and USRP-B210 with 802.11g PHY layer. WiFi receiver 1 is used to obtain the original WiFi information. WiFi receiver 2 listens for the backscattered WiFi signal. To assess the system performance, we use the following two metrics:

Bit Error Rate (BER): The BER is defined as the ratio of the number of bit errors to the total number of transmitted bits. In our experiments, for a communication distance to be considered effective, the BER must be lower than 1%.

Throughput: The throughput is defined as the number of correctly demodulated data bits at the PHY layer, calculated as $(\text{Total bits} * (1 - \text{BER}))/\text{Time}$.

B. Comparing With State-of-the-Art

We first compare TScatter's performance with the best results from recent OFDM-based WiFi backscatter systems (X-Tandem [6], MOXcatter [8], FreeRider [7]). Figure 15(a) shows TScatter's significantly lower BER compared to other solutions at similar throughput. Specifically, TS-Nibble and TS-Binary achieve approximately 250 kbps and 62.5 kbps respectively. Figure 15(b) presents throughput comparisons, where TScatter achieves 10.61 Mbps using 4-phase modulation. The main reason TScatter achieves orders of magnitude better performance is that it takes advantage of the phase error correction. State-of-the-art solutions require specific commodity NICs that can disable the phase error correction. However, the phase error correction is very important for the OFDM demodulation because the phase error may be dynamically changing due to the changes of environment (e.g., temperature of the oscillators and object movement). Without the phase error correction, the OFDM demodulation error will increase. Similarly, disabling phase error correction will also introduce phase error into the decoding procedure of the codeword translation, which will increase the tag data decoding error. In contrast, TScatter takes the phase error correction into consideration. On the demodulation side, TScatter builds the demodulation model from the backscattered samples to the corrected subcarriers that can not only correct the phase error due to the dynamic environments but also decode the tag data by estimating the minimal Euclidean distance rather than by using XOR. It also utilizes sample-level modulation to apply more robust coding methods to improve BER or throughput. As a result, TScatter's BERs are around 10^2 , 10^3 , and 10^4 times lower than FreeRider, MOXcatter, and X-Tandem, respectively. Similarly, its throughput is 53,050,

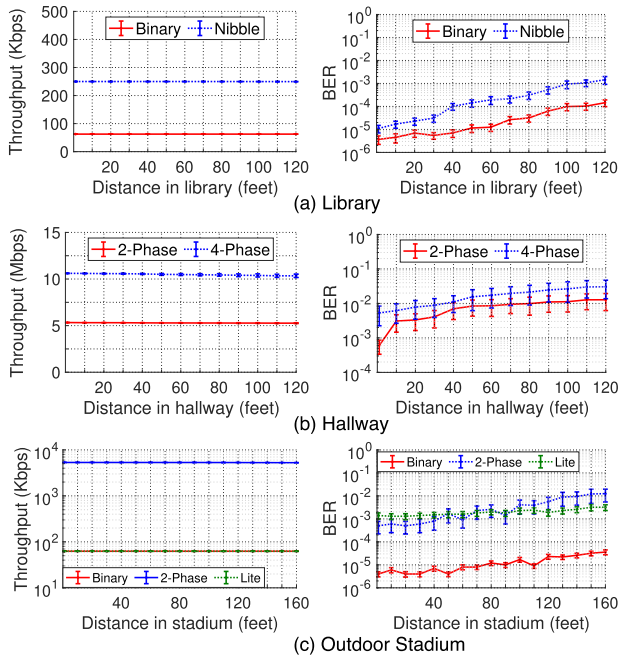


Fig. 16. Throughput and BER in library, hallway and outdoor stadium.

212, and 176 times higher than X-Tandem, MOXcatter, and FreeRider, respectively.

C. Library

To understand how TScatter works in a multipath rich environment, we conducted the experiments in a library. Since there are a lot of shelves, tables, and chairs in the library, these obstacles not only block the direct line-of-sight transmission path but create more multipath effects as well. In this setting, the backscatter tag and WiFi transmitter are deployed on the shelves. We move the WiFi receiver from shelves to tables to vary the multipath environment.

Figure 16(a) show the throughput and BER over different communication distances, respectively. As we can observe from these figures, the throughput of TScatter is similar over the change of distances while the BER increases slowly. When the communication distance reaches 120 feet, the average BERs of TScatter are still around 10^{-3} and 10^{-4} for Nibble and Binary, respectively. Since the multipath effect will become more severe as the distance increases in the library scenario, we can conclude that TScatter is resistant to the wireless channel interference. Because the demodulation process takes the phase error introduced by the wireless channel into consideration, TScatter shows great advantages in this scenario.

D. Hallway

In this section, we evaluate TScatter in the hallway scenario. As shown in Figure 14(c), the hallway is on the second floor of an academic building. This scenario is selected to reflect TScatter's real-world performance with less environmental impact because the hallway is relatively spacious. In this scenario, the backscatter tag and the WiFi receiver were kept

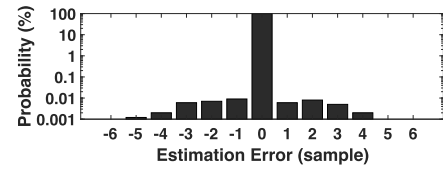


Fig. 17. The distribution of starting tag modulation sample detection error (the y-axis is in log scale). Our method achieves 98% estimation accuracy, i.e., the probability of a 0-sample estimation error is 98%. We can utilize the multiple estimations to correct the erroneous one.

in line-of-sight. The distance between the tag and the receiver is changing from one foot to 120 feet.

Figure 16(b) shows the throughput of our TScatter system using different modulation schemes (i.e., 2-phase and 4-phase). Overall, TScatter can achieve a maximum of 10.61 Mbps and the throughput is stable over different communication distances because it has a high granularity modulation scheme. The BER under different communication distances. We can observe that the BER for the 2-phase modulation scheme is very low. The average BER is lower than 1% when the distance increases to 70 feet. Even when the distance increases to 120 feet, the average BER is still around 1%. This is because our demodulation model captures the impact of the phase error correction on the received subcarriers.

E. Outdoor Stadium

To understand the performance of TScatter with different modes (i.e., low BER, high throughput, and lite version) in the same scenario, we conduct experiments in an outdoor stadium. The distance between the tag and the receiver is changing from one foot to 160 feet.

Figure 16(c) shows the throughput over different communication distances. The high throughput mode (2-phase) outperforms the low BER mode (binary) and Lite TScatter because it has a high granularity modulation scheme. When the receiver is 160 feet away from the tag, the throughput of binary, 2-phase, and Lite TScatter are around 62.5 kbps, 5.26 Mbps and 62.3 kbps, respectively. The BERs at different distances. The BER of low BER mode and Lite TScatter is more stable than that of high throughput mode despite the fact that the signal strength degrades across distance. Specifically, the BER of low BER mode is much less than that of the other two modes. Therefore, the low BER mode has the significant advantage on providing reliable backscatter communications. When the distance reaches 160 feet, the average BERs of binary, 2-phase, and Lite TScatter are still around 10^{-5} , 10^{-2} and 10^{-3} , respectively.

F. Accuracy of Starting Tag Modulation Sample Detection

As discussed in Section V-A, accurately estimating the starting tag modulation sample is crucial for demodulating the tag data. In this section, we evaluate the accuracy of our method in estimating the starting tag modulation sample. Figure 17 shows the distribution of estimation errors for the starting tag modulation sample. Our results indicate that our method achieves a high estimation accuracy: 98% of the estimations have zero-sample errors. The remaining estimations

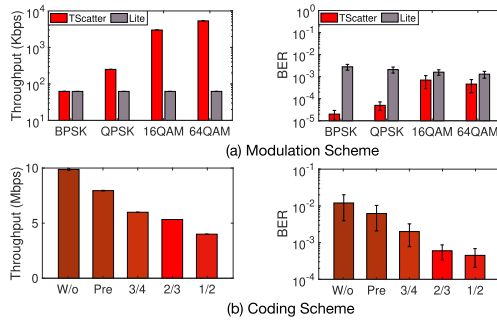


Fig. 18. Impact of OFDM Modulation Scheme and Tag Coding Scheme.

demonstrate minimal errors, with 0.63% and 0.66% of them having errors of ± 1 sample and ± 2 samples, respectively. Then we can utilize the multiple values obtained from these estimations to correct the erroneous one, ensuring the accurate identification of the starting tag modulation sample throughout the transmission process.

G. Impact of OFDM Modulation Scheme

In Section V-B and V-C, we introduced how TScatter deals with lower and higher level WiFi OFDM modulation schemes respectively. In this section, we compare sample-level modulation TScatter's performance (in terms of throughput and BER with low BER or high throughput coding scheme) with Lite TScatter (which uses symbol level modulation) under different WiFi OFDM modulation schemes (i.e., BPSK, QPSK, 16QAM, and 64QAM). Figure 18(a) shows the throughput result. We observe that the throughput of Lite TScatter is almost the same under different OFDM modulation schemes because one bit of data is embedded in per symbol no matter what the OFDM modulation scheme is. For TScatter, the throughput increases while the OFDM modulation scheme changes from the lowest level (i.e., BPSK) to the highest level (i.e., 64QAM). At 64QAM, the throughput reaches 5.33 Mbps. The reason of this growing trend is that higher level OFDM modulation schemes can provide higher resolution to decode the tag data, which can accommodate more tag data per symbol.

Figure 18(a) also shows the result of BER versus OFDM modulation scheme. We can observe that TScatter's BER is lower than Lite TScatter's across all kinds of modulation schemes even when the OFDM transmitter utilizes BPSK. This is because the demodulation model captures the impact of the phase error correction on the received subcarriers. Moreover, TScatter leverages the sample-level modulation to apply more robust coding methods to further improve the BER.

H. Impact of Tag Coding Scheme

In Section V-C, we have described TScatter's coding schemes for high throughput. Figure 18(b) shows TScatter's performance with different code rates. "W/o" denotes the results without using coding scheme. "Pre" denotes the results that TScatter only uses predefined data without coding. "3/4", "2/3", and "1/2" denote the results that TScatter uses both predefined data and coding with a code rate at

3/4, 2/3, and 1/2, respectively. Since the experiments show similar trends, we show the results with the 2-phase modulation scheme in the hallway scenario. The WiFi transmitter utilizes 64QAM.

When no coding scheme is utilized, the whole 40-sample sequence is used to store the unknown tag data (described in Section V-A). Consequently, TScatter has the highest throughput as well as the highest BER among all coding schemes. The reason of the high BER is that the corrected data subcarrier values deviate from their mapping points, which cause estimation errors in Equation 22. To reduce the estimation errors, we replace 8 unknown tag data with the predefined data to constrict the estimation result to a smaller range of values. Figure 18(b) also shows that the BER decreases to below 1%. To achieve a more reliable communication, we combine the predefined data and the convolutional code to generate the tag data. When the code rate is up to 2/3, the BER is lower than 0.1% and the throughput is as high as 5.33 Mbps, which provides a reliable high throughput communication.

I. Energy Consumption Analysis

We designed an integrated circuit for TScatter's digital processing module and conducted a simulation using Cadence Spectre for TSMC 65nm process. In the power consumption simulation, multiple factors are considered including Vdd (1.6 V \sim 2 V), temperature (25 $^{\circ}$ C), system clock frequency, and power mode.

The IC design consists of four modules: clock, energy detector, control module and RF transistor. TScatter first detects the WiFi packet by using the energy detector. When the WiFi packet is detected, the clock synthesizes a 20 MHz frequency for the control module. Then the control module reads the data from the sensor, latches and codes the sensor data, and toggles the RF transistor according to the coded data. Since the power consumption of the energy detector and the RF transistor is very low (around 0.3 μ W), we mainly introduce the implementation details of the clock and the control module:

Clock. The bottleneck of the power consumption of the TScatter system is the clock. We use a ring oscillator to synthesize a 20 MHz frequency. The frequency accuracy is sensitive to temperature change. We add a thermistor to design a temperature compensation circuit to compensate for the frequency drift. Moreover, we add a trimming pad to correct the frequency. The simulation result shows that the power consumption of the ring oscillator is 2.7 μ W.

Control module. Our control module comprises a cache circuit for sensor data storage, a coding circuit to encode the data, and an inverter acting as a fan-out to toggle the RF transistor. The cache is built using basic cache cells, which consist of a D latch and a transmission gate. By linking the D latch output to the transmission gate input, a cache cell capable of simultaneous read and write is formed. We utilize latches instead of flash due to their low power consumption even at 20 MHz. The coding circuit includes a lookup table for low BER mode, featuring 16 PN sequences (Table I), constructed from basic logic gates, and a convolutional encoder for high throughput mode, composed of 7 latches and 3 XOR gates.

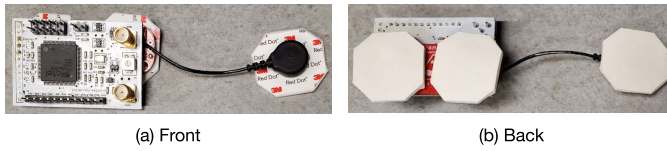


Fig. 19. EMG connected TScatter tag.

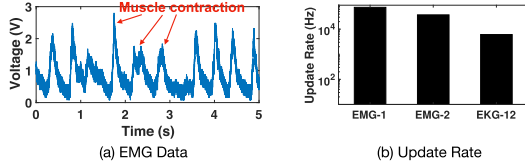


Fig. 20. (a) displays the EMG readings during dumbbell lifts with the system attached to the arm. (b) shows the average update rate when the system is linked with 1-channel EMG, 2-channel EMG, and 12-channel EKG.

The power consumption of the control module, according to simulations, is around $6.2 \mu\text{W}$.

From the above analysis, the overall power consumption of TScatter is $30.2 \mu\text{W}$, which aligns with other OFDM backscatter systems. This is mainly because the most power-hungry part of the backscatter tag is the clock. No matter the sample-level modulation or the symbol-level modulation, they all need to shift the incoming WiFi signal to the adjacent band channel, which can be done by toggling the RF switch at a specific frequency. The minimum value of the frequency equals a WiFi channel bandwidth, i.e., 20 MHz. Hence, both modulation types operate at 20 MHz. Although TScatter performs the convolutional coding in high throughput mode, the encoder which is constructed by 7 D latches and 3 XOR gates has a very low power consumption even when it works at 20 MHz. Therefore, the energy consumptions of TScatter and other OFDM backscatter systems are similar.

X. APPLICATION

In this section, we show that biometric measurements (such as EMG and EKG) can be passively wirelessly transmitted in real-time by using TScatter. Figure 19 presents a prototype of an EMG sensor connected to the TScatter tag. Utilizing foam electrodes, the tag and sensor can be secured to a muscle for recording its electrical activity. Figure 20(a) displays data from the system monitoring arm activities, with each peak indicating a muscle contraction. Figure 20(b) presents the average update rate under various configurations: i) EMG-1, where a 1-channel EMG monitors sleep abnormalities on a leg muscle; ii) EMG-2, where a 2-channel EMG monitors fitness levels on the upper and lower arm; and iii) EKG-12, where a 12-channel EKG tests TScatter's compatibility with multi-channel EKGs. With a common sampling rate of 1 kHz and a 24-bit resolution, the needed bit rate for 12-channel EKG is 288 kbps. Despite wireless signal attenuation due to the human body and upper layer overheads (e.g., ACKs), TScatter, capable of offering around 5-10 Mbps throughput with different modulations, should suffice for 12-channel EKG measurements. We note that in our implementation, we use an EKG sensor for data collection and TScatter solely for

passive wireless transmission, ensuring no amplitude difference between the EKG sensor data and the transmitted data.

XI. RELATED WORK

Recently, backscatter has played an attractive role in the passive communication field because of its significant performance on power consumption. Various innovative techniques have been proposed to facilitate diverse applications [24], [25], [26]. For instance, ReMix [24] overcomes human body surface interference to backscatter from deep tissue devices, Living IoT [25] enhances smart farming by attaching backscatter devices to live insects, and PAB [26] supports long-term oceanic applications by backscattering acoustic signals underwater.

To leverage existing infrastructures, such as TV band [1], FM radio [2], LoRa [3], [27], Bluetooth [12], [28], ZigBee [29], and WiFi [4], [6], [7], [8], [9], [10], [13], [14], [30], [31], [32], and LTE [5], researchers have also explored various solutions. Specifically, in the OFDM WiFi backscatter field, WiFi Backscatter [9] modulates CSI and RSSI measurements at OFDM receivers to convey information. BackFi [10] uses customized full-duplex devices to clear interference during OFDM signal backscattering. FS-Backscatter [13] shifts backscattered OFDM signals to an adjacent band to minimize interference. FreeRider [7] introduces the OFDM-based codeword translation technique, which piggybacks tag data by altering the phase of backscattered OFDM symbols. WiTAG: [32] sends data by selectively interfering with subframes in an aggregated frame, allowing compliant communication over modern 802.11n and 802.11ac networks, without needing hardware or software modifications to any devices. Building on FreeRider, MOX-catter [8], VMscatter [31], and X-Tandem [6] develops MIMO OFDM backscatter systems and a multi-hop OFDM backscatter system, respectively. Diverging from existing backscatter systems, TScatter pioneers sample-level coding and subcarrier-level decoding using OFDM waveforms, achieving significantly lower BER or higher throughput than current approaches.

To address the synchronization issue of the OFDM backscatter systems, the latest work SyncScatter [15] built a customized amplifier into the integrated circuit to achieve fine synchronization. TScatter, on the other hand, adopts a sample-level modulation perspective to provide a more in-depth understanding of synchronization issues and leverages coarse synchronization for backscatter communication. It is worth noting that TScatter is complementary to SyncScatter, as the latter employs symbol-level modulation, which necessitates disabling phase error correction during communication. By integrating both, TScatter could help SyncScatter retain phase error correction, reducing BER and enhancing compatibility with standard WiFi devices.

XII. DISCUSSION AND CONCLUSION

Although our system was tested using USRPs, we believe that our Lite TScatter has the potential to be deployed on commodity WiFi devices with minimal modification, as it

retains symbol-level modulation and XOR-based decoding while serving as a natural extension of existing OFDM backscatter systems. Based on the description of prior works and WiFi specifications [6], [7], [8], [18], [33], implementing Full TScatter on commodity devices would require addressing the following engineering challenges: 1) As backscattered WiFi packets are modified, commodity receivers may discard them due to CRC errors. Similar to other WiFi backscatter systems, TScatter could employ a WiFi card in monitor mode to receive packets even with bad checksums. 2) Full TScatter decoding entails calculating minimum Euclidean distance using QAM mapping data, composed of discrete binary bits. We believe that future WiFi chips will provide interfaces to read these data bits, enhancing Full TScatter's accessibility on commodity devices. Our primary contribution is a novel OFDM backscatter design principle, sample-level modulation. Our comprehensive evaluations reveal that TScatter in real-world scenarios can achieve significantly lower BER or much higher throughput than existing approaches [6], [7], [8].

REFERENCES

- [1] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: Wireless communication out of thin air," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2013, pp. 1–12.
- [2] A. Wang, V. Iyer, V. Talla, J. R. Smith, and S. Gollakota, "FM backscatter: Enabling connected cities and smart fabrics," in *Proc. 14th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Mar. 2017, pp. 243–258.
- [3] M. Hesar, A. Najafi, and S. Gollakota, "NetScatter: Enabling large-scale backscatter networks," in *Proc. 16th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Feb. 2019, pp. 271–283.
- [4] B. Kellogg, V. Talla, S. Gollakota, and J. R. Smith, "Passive Wi-Fi: Bringing low power to Wi-Fi transmissions," in *Proc. 13th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Mar. 2016, pp. 151–164.
- [5] Z. Chi, X. Liu, W. Wang, Y. Yao, and T. Zhu, "Leveraging ambient LTE traffic for ubiquitous passive communication," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2020, pp. 172–185.
- [6] J. Zhao, W. Gong, and J. Liu, "X-Tandem: Towards multi-hop backscatter communication with commodity WiFi," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Oct./Nov. 2018, pp. 1–15.
- [7] P. Zhang, C. Josephson, D. Bharadia, and S. Katti, "FreeRider: Backscatter communication using commodity radios," in *Proc. 13th Int. Conf. Emerg. Netw. Exp. Technol.*, Nov. 2017, pp. 389–401.
- [8] J. Zhao, W. Gong, and J. Liu, "Spatial stream backscatter using commodity WiFi," in *Proc. 16th Annu. Int. Conf. Mobile Syst. (MobiSys)*, Jun. 2018, pp. 1–13.
- [9] B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall, "Wi-Fi backscatter: Internet connectivity for RF-powered devices," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2014, pp. 1–12.
- [10] D. Bharadia, K. R. Joshi, M. Kotaru, and S. Katti, "BackFi: High throughput WiFi backscatter," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2015, pp. 283–296.
- [11] D. Vasisht, S. Kumar, H. Rahul, and D. Katabi, "Eliminating channel feedback in next-generation cellular networks," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2016, pp. 1–14.
- [12] V. Iyer, V. Talla, B. Kellogg, S. Gollakota, and J. Smith, "Inter-technology backscatter: Towards Internet connectivity for implanted devices," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2016, pp. 1–14.
- [13] P. Zhang, M. Rostami, P. Hu, and D. Ganesan, "Enabling practical backscatter communication for on-body sensors," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2016, pp. 1–14.
- [14] P. Zhang, D. Bharadia, K. Joshi, and S. Katti, "HitchHike: Practical backscatter using commodity WiFi," in *Proc. 14th ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Nov. 2016, pp. 1–13.
- [15] M. Dunna, M. Meng, P.-H. Wang, C. Zhang, P. Mercier, and D. Bharadia, "SyncScatter: Enabling WiFi like synchronization and range for WiFi backscatter communication," in *Proc. 18th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Apr. 2021, pp. 923–937.
- [16] Y. Yao et al., "Aegis: An interference-negligible RF sensing shield," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 1718–1726.
- [17] W. Wang, T. Xie, X. Liu, and T. Zhu, "ECT: Exploiting cross-technology concurrent transmission for reducing packet delivery delay in IoT networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 369–377.
- [18] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2016, Dec. 2016.
- [19] W. Wang, X. Liu, Y. Yao, Y. Pan, Z. Chi, and T. Zhu, "CRF: Coexistent routing and flooding using WiFi packets in heterogeneous IoT networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 19–27.
- [20] Y. Li, Z. Chi, X. Liu, and T. Zhu, "Chiron: Concurrent high throughput communication for IoT devices," in *Proc. 16th Annu. Int. Conf. Mobile Syst. (MobiSys)*, Jun. 2018, pp. 1–13.
- [21] NVIDIA. (2019). *Jetson Nano Brings AI Computing to Everyone*. [Online]. Available: <https://devblogs.nvidia.com/jetson-nano-ai-computing/>
- [22] C. Xu and P. Zhang, "Open-source software and hardware platforms for building backscatter systems," *GetMobile, Mobile Comp. Commun.*, vol. 23, no. 1, pp. 16–20, Jul. 2019.
- [23] W. Wang, X. Liu, Y. Yao, and T. Zhu, "Exploiting WiFi AP for simultaneous data dissemination among WiFi and ZigBee devices," in *Proc. IEEE 29th Int. Conf. Netw. Protocols (ICNP)*, Nov. 2021, pp. 1–11.
- [24] D. Vasisht, G. Zhang, O. Abari, H.-M. Lu, J. Flanz, and D. Katabi, "In-body backscatter communication and localization," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2018.
- [25] V. Iyer, R. Nandakumar, A. Wang, S. B. Fuller, and S. Gollakota, "Living IoT: A flying wireless platform on live insects," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Oct. 2019, pp. 1–15.
- [26] J. Jang and F. Adib, "Underwater backscatter networking," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Aug. 2019, pp. 1–13.
- [27] V. Talla, M. Hesar, B. Kellogg, A. Najafi, J. R. Smith, and S. Gollakota, "LoRa backscatter: Enabling the vision of ubiquitous connectivity," in *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 1, no. 3, Sep. 2017, pp. 1–24.
- [28] M. Zhang, S. Chen, J. Zhao, and W. Gong, "Commodity-level BLE backscatter," in *Proc. 19th Annu. Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, Jun. 2021, pp. 402–414.
- [29] Y. Li, Z. Chi, X. Liu, and T. Zhu, "Passive-ZigBee: Enabling ZigBee communication in IoT networks with 1000X+ less power consumption," in *Proc. 16th ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Nov. 2018, pp. 159–171.
- [30] R. Zhao et al., "OFDMA-enabled Wi-Fi backscatter," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Oct. 2019, pp. 1–15.
- [31] X. Liu, Z. Chi, W. Wang, Y. Yao, and T. Zhu, "VMscatter: A versatile MIMO backscatter," in *Proc. 17th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Feb. 2020, pp. 895–909.
- [32] A. Abedi, F. Dehbashi, M. H. Mazaheri, O. Abari, and T. Brecht, "WiTAG: Seamless WiFi backscatter communication," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM)*, Jul. 2020, pp. 240–252.
- [33] Z. Chi, Y. Li, X. Liu, Y. Yao, Y. Zhang, and T. Zhu, "Parallel inclusive communication for connecting heterogeneous IoT devices at the edge," in *Proc. 17th Conf. Embedded Netw. Sensor Syst. (SenSys)*, Nov. 2019, pp. 205–218.