Implementation of Physical-layer Network Coding

Lu Lu*, Taotao Wang*, Soung Chang Liew* and Shengli Zhang†
*Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong.
†Department of Communication Engineering, Shenzhen University, China.
email:{ll007, wtt011, soung}@ie.cuhk.edu.hk, zsl@szu.edu.cn

***Abstract*—This paper presents the first implementation of a two-way relay network based on the principle of physical-layer network coding. To date, only a simplified version of physical-layer network coding (PNC), called analog network coding (ANC), has been successfully implemented. The advantage of ANC is that it is simple to implement; the disadvantage, on the other hand, is that the relay amplifies the noise along with the signal before forwarding the signal. PNC systems in which the relay performs XOR or other denoising PNC mappings of the received signal have the potential for significantly better performance. However, their implementation also poses many challenges. For example, the relay must be able to deal with symbol and carrier-phase asynchronies of the simultaneous signals received from the two end nodes, and the relay must perform channel estimation before decoding. We investigate a PNC implementation in the frequency domain, referred to as FPNC, to tackle these challenges. FPNC is based on OFDM. In FPNC, XOR mapping is performed on the OFDM samples in each subcarrier rather than on the samples in the time domain. We implement FPNC on the universal soft radio peripheral (USRP) platform. Our implementation requires only moderate modifications of the packet preamble design of 802.11a/g OFDM PHY. With the help of the cyclic prefix (CP) in OFDM, symbol asynchrony and the multi-path fading effects can be dealt with simultaneously in a similar fashion. Our experimental results show that symbol-synchronous and symbol-asynchronous FPNC have essentially the same BER performance, for both channel-coded and unchannel-coded FPNC.**

***Index Terms*—physical-layer network coding, network coding implementation, software radio**

I. Introduction

In this paper, we present the first implementation of physical-layer network coding (PNC) on the software radio platform. We believe this prototyping effort moves the concept of PNC a step toward reality. Our implementation work also exposes and raises some interesting issues for further research.

PNC, first proposed in [1], is a subfield of network coding [2] that is attracting much attention recently. The simplest system in which PNC can be applied is the two-way relay channel (TWRC), in which two end nodes A and B exchange information with the help of a relay node R in the middle, as illustrated in Fig. 1. Compared with the conventional relay system, PNC could double the throughput of TWRC by reducing the time slots for the exchange of one packet from four to two [1]. In PNC, in the first time slot, end nodes A and B send signals simultaneously to relay R; in the second time slot, relay R processes the superimposed signals and maps them to a network-coded packet for broadcast to the end nodes. From the network-coded packet, each end node then makes use of its self information to extract the packet from the other end node [1], [3], [4].

Fig. 1. System model for physical-layer network coding.

Prior to this paper, only a simplified version of PNC, called analog network coding (ANC) [5], has been successfully implemented. The advantage of ANC is that it is simple to implement; the disadvantage, on the other hand, is that the relay amplifies the noise along with the signal before forwarding the signal, causing error propagation.

To our best knowledge, the implementation of the original PNC based on XOR mapping as in [1] has not been demonstrated, even though it could have significantly better performance. A reason is that the implementation of XOR PNC poses a number of challenges. For example, the relay must be able to deal with symbol and carrier-phase asynchronies of the simultaneous signals received from the two end nodes, and the relay must perform channel estimation before decoding.

This paper presents a PNC implementation in the frequency domain, referred to as FPNC, to tackle these challenges. In particular, FPNC is based on OFDM, and XOR mapping is performed on OFDM samples in each subcarrier rather than the samples in the time domain. We implement FPNC on the universal soft radio peripheral (USRP) platform. Our implementation requires only moderate modifications of the packet preamble design of 802.11a/g OFDM PHY. With the help the cyclic prefix (CP) in OFDM, symbol asynchrony and the multi-path fading effects can be dealt with simultaneously in a similar fashion. Our experimental results show that symbol-synchronous and symbol-asynchronous FPNC have nearly the same BER performance, for both channel-coded and unchannel-coded FPNC.

The remainder of this paper is organized as follows: Section II overviews the challenges of PNC. Section III presents the FPNC frame format design. Section IV addresses the key implementation challenges. Experimental results are given in

This work is supported by AoE grant E-02/08 and the General Research Funds Project Number 414911, established under the University Grant Committee of the Hong Kong Special Administrative Region, China, and the CUHK Direct Grant 2050464. This work is also supported by NSF of China (Project No. 60902016), NSF of Guangdong (Project No. 10151806001000003) and NSF of Shenzhen (Project No. JC201005250034A).

Section V. Finally, Section VI concludes this paper.

II. Challenges

In the following, we overview the challenges of PNC, and the implementation approaches taken by us to tackle them:

A. Synchronization

A key issue in PNC is how to deal with the asynchrony between the signals transmitted simultaneously by the two end nodes [6]. That is, symbols transmitted by the two end nodes could arrive at the receiver with symbol misalignment. For theoretical research, perfect synchronization is usually assumed [1], [4], [7]. This assumption is demanding to achieve in real systems.

This paper focuses on the implementation of asynchronous PNC. To deal with asynchrony, our FPNC implementation makes use of OFDM to lengthen the symbol duration within each subcarrier. Then, independent XOR PNC mapping is performed within each subcarrier. Specifically, we have the following frequency domain FPNC setup.

1) FPNC Frequency Domain Setup: Suppose that there are M_A and M_B paths from nodes A and B to relay R with delays $\tau_A^0 < \tau_A^1 < \cdots < \tau_A^{M_A-1}$ and $\tau_B^0 < \tau_B^1 < \cdots < \tau_B^{M_B-1}$. Define C as the length of the CP. We require that FPNC delay spread $\triangleq \max\left(\tau_A^{M_A-1}, \tau_B^{M_B-1}\right) \leq C$. This requirement is referred to as the *Delay-Spread-Within-CP requirement*. The received signal at subcarrier k is[1]

$$Y[k] = H_A[k]X_A[k] + H_B[k]X_B[k] + W[k], \qquad (1)$$

where $k = 0, \ldots, N-1$ represents the subcarrier index in one OFDM symbol. Note that the time-domain delay spread has been incorporated into $H_A[k]$ and $H_B[k]$. In FPNC, we map $Y[k]$ for each subcarrier k into the XOR, $X_A[k] \oplus X_B[k]$. This will be detailed in Section IV-C. The main point here is in (1), the signals of different subcarriers k are isolated from each other, and we only need to perform PNC mapping within each subcarrier.

Our discussion thus far has assumed the absence of CFO. When there is CFO, inter-carrier interference (ICI) may occur, and this will be further discussed in Section IV-A.

B. Channel Estimation

For good performance of asynchronous PNC, the relay must have the knowledge of the uplink channel state information (CSI). This has been the assumption in many prior works on PNC (e.g., [1], [9]). This means that in implementation, the relay will need to estimate the channel gains. Most channel estimation techniques for the OFDM system assume point-to-point communication in which only one channel needs to be estimated. In PNC, the relay needs to estimate two channels based on simultaneous reception of signals (and preambles) from the two end nodes. This poses the following two problems in PNC that do not exist in point-to-point communication:

1) Channel Gains: Channel estimation in a point-to-point OFDM system (e.g., 802.11 [10]) is generally facilitated by training symbols and pilots in the transmitted signal. If used unaltered in the PNC system, the training symbols and pilots from the two end nodes may overlap at the relay, complicating the task of channel estimation. In our implementation, we solve this problem by assigning orthogonal training symbols and pilots to the end nodes. The details will be given in Section IV.

2) CFOs: It is well known that carrier frequency offset (CFO) between the transmitter and the receiver can cause inter-subcarrier interference (ICI) if left uncorrected. In a point-to-point system, CFO can be estimated and compensated for. In PNC, we have two CFOs at the relay, one with respect to each end node. Even if the two CFOs can be estimated perfectly, their effects cannot be both compensated for totally; the total elimination of the ICI of one end node will inevitably lead to a larger ICI for the other end node. To strike a balance, our solution is to compensate for the mean of the two CFOs (i.e., compensate for $(\mathrm{CFO}_A + \mathrm{CFO}_B)/2$). The details will be elaborated in Section IV.

C. Joint Channel Decoding and Network Coding

For reliable communication in a practical PNC system, channel coding needs to be incorporated. This paper considers link-by-link channel-coded PNC, in which the relay maps the overlapped channel-coded symbols of the two end nodes [4], [11] to the XOR of the source symbols[2]; after that, the relay channel-encodes the XOR source symbols to channel-coded symbols for forwarding to the end nodes. Such a link-by-link channel-coded PNC system has better performance than an end-to-end channel-coded PNC system [4], [11].

In our FPNC design, we adopt the convolution code as defined in the 802.11 a/g standard. The relay first maps the overlapped channel-coded symbols to their XOR on a symbol-by-symbol basis. After that it cleans up the noise by (i) channel-decoding the XOR channel-coded symbols to the XOR source symbols, and then (ii) re-channel-coding the XOR source symbols to the XOR channel-coded symbols for forwarding to the two end nodes.

III. FPNC Frame Format

This section focuses on the PHY frame design to enable asynchronous operation, channel estimation, and frequency offset compensation in FPNC. As previously mentioned, the asynchronous operation requires the PNC delay spread to be within CP. To ensure this, a simple MAC protocol as follows

[1]Note here that the FPNC frequency domain expression is a general result that takes into account the multi-path channels and also the symbol shift (which is embeded in the multipath delays $\tau_A^{M_A-1}$ and and $\tau_B^{M_B-1}$). Interested readers are referred to our technical report [8] for a rigorous derivation.

[2]This process is called Channel-decoding-Network-Coding (CNC) in [11] because it does two things: channel decoding and network coding. Unlike the traditional multiuser detection (MUD) in which the goal is to recover the individual source information from the two end nodes, CNC aims to recover the XOR of the source information during the channel decoding process. CNC is a component in link-by-link channel-coded PNC critical for its performance [4], [11].

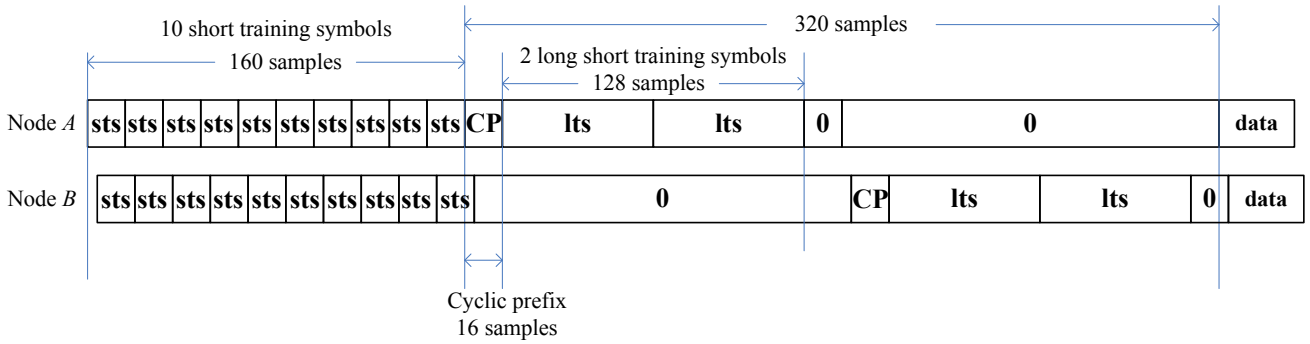

Fig. 2. FPNC preamble format.

could be used to trigger near-simultaneous transmissions by the two end nodes. The relay could send a short polling frame (similar to the "beacon frame" in 802.11 that contains only 10 Bytes) to the end nodes. Upon receiving the polling frame, the end nodes then transmit. With this method, the symbols would arrive at the relay with a relative delay offset of $|RTT_A - RTT_B|$, where RTT is the round trip time, including the propagation delay and the processing time at the end nodes. This delay offset is not harmful to our system as long as the sample misalignment of two end nodes is within the CP length.

Given this loose synchronization, our training symbols and pilot designs described below can then be used to facilitate channel estimation and frequency offset compensation in F-PNC. We modify the PHY preamble design of 802.11a/g for FPNC. The overall FPNC frame format is shown in Fig. 2. The functions of the different components in the PHY preamble are described in the next few subsections.

A. FPNC Short Training Symbol

In 802.11, the short training symbol (STS) sequence contains 160 time-domain samples, in which 16 samples form one STS unit (sts) for a total of 10 identical units, as shown in Fig. 2. FPNC adopts the same STS sequence as in 802.11, as illustrated in Fig. 2. The STS sequence is used by the relay node to perform the sample timing recovery on the received frame. In particular, the relay node applies a cross-correlation to locate the sample boundary for the long training symbols that follow the STS sequence. The normalized cross-correlation is defined as follows:

$$Z[n] = \frac{\left|\sum_{i=0}^{L-1} (sts^*[i] y_R[n+i])\right|}{\sum_{i=0}^{L-1} (y_R[n+i] y_R^*[n+i])}, \quad (2)$$

where n is the received sample index, $y_R[n]$ is the n-th sample at the relay R, and $L = 16$ is the length of each sts. For FPNC, this cross-correlation will result in 20 peaks over the STS sequences (see Fig. 3) of the two frames if the frames are not synchronized. From this profile of peaks, we can identify the last two peaks. If the Delay-Spread-Within-CP requirement is satisfied, then the last two peaks must be the last peaks of

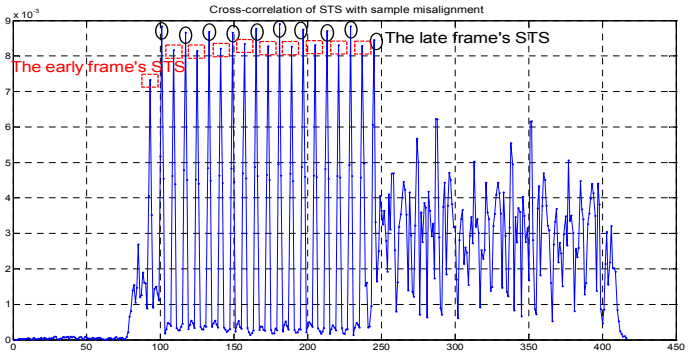

Fig. 3. Cross-correlation of the STS for the uplink of FPNC.

A and B, respectively. This is because the CP as well as the sts are of 16 samples in length. From there, we could locate the boundaries of the long training symbol (LTS) of A and B that follow. Note that when the STS sequences of nodes A and B overlap exactly, we will have ten peaks only. In this case, the LTS boundaries of A and B also overlap exactly, and we simply use the last peak to identify the common boundary.

B. FPNC Long Training Symbol

With reference to Fig. 2, the 802.11 LTS sequence contains 160 time-domain samples in which there is a CP followed by two identical LTS units, lts. The receiver uses the LTS sequence to perform channel estimation and CFO compensation.

For FPNC, in order to estimate two uplink channel gains, we design the LTS so that it contains twice the length of LTS in 802.11a/g, as shown in Fig. 2. In Fig. 2, we intentionally show the case in which the LTS sequences of the two end nodes are not exactly synchronized. Note that we change the 802.11 LTS design by shortening its original CP length from 32 to 16 to make sure that the two lts units of B will not overlap with the data of A that follows under the condition that the delay spread is less than the CP length of 16. This does not impose additional requirement on the delay spread, since the CP of the data OFDM symbols in 802.11a/g (and FPNC) have only 16 samples anyway (i.e., the delay spread must be within 16 samples anyway). Section IV will detail the CFO compensation and channel estimation methods for our implementation.

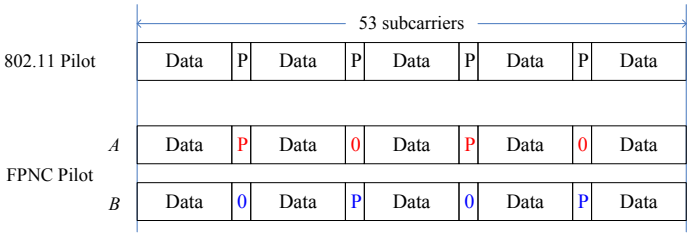

Fig. 4. Pilot design for FPNC (frequency domain).

C. FPNC Pilot

There are four pilots for each OFDM symbol in 802.11, as shown in Fig. 4. The four pilots are used to fine-tune the channel gains estimated from LTS. In a frame, there are multiple OFDM symbols, but only one LTS in the beginning. In practice, the channel condition may have changed by the time the later OFDM symbols arrive at the receiver. That is, the original channel gains as estimated by LTS may not be accurate anymore for the later OFDM symbols. The pilots are used to track such channel changes.

We design the FPNC pilots of nodes A and B by nulling certain pilots to introduce orthogonality between them, as shown in Fig. 4. As will be detailed in Section IV-B, this allows us to track the channel gains of A and B separately in a disjoint manner in FPNC. We conducted some experiments for a point-to-point communication system using the two-pilot design rather than the four-pilot design. We find that for our linear interpolation channel tracking scheme described in Section IV-B, the BER performances of the two-pilot and four-pilot designs are comparable for BPSK- and QPSK-modulated systems.

IV. Addressing Key Implementation Challenges

We next present our methods for carrier frequency offset compensation, channel estimation, and FPNC mapping, assuming the use of the PHY frame format presented in Section III.

A. FPNC Carrier Frequency Offset (CFO) Compensation

For the uplink phase, when there are CFOs, the received frames at relay R will suffer from time-varying phase asynchronies. We need to compensate for the CFOs to alleviate inter-carrier interference (ICI) among data on different subcarriers. In particular, there are two CFOs for the uplink phase of FPNC system. Estimating and compensating for the two CFOs simultaneously are demanding [12], [13]. In our FPNC implementation, we first calibrate the carrier frequencies at nodes A and B with reference to the relay R via beacon frame to make sure that the oscillator frequencies are roughly equal to each other. For CFO compensation, we first estimate the two independent CFOs (namely CFO_A and CFO_B) caused by the carrier frequency offsets between nodes A and B and relay R, respectively. We then compensate for the mean of the two CFOs (i.e., $CFO_{PNC} = (CFO_A + CFO_B)/2$). This may not be the most optimal approach, but it is simple and can mitigate the ICI caused by CFO to some extent. Further details of this scheme and its performance can be found in our technical report [8].

After compensation, our received data in the time domain is given by

$$\tilde{y}_R[n] = y_R[n]e^{-jn\tilde{\phi}}, \tag{3}$$

where $y_R[n]$ is the received samples and $\tilde{\phi}$ is the is the CFO compensation. In the frequency domain, we have

$$\tilde{Y}_R[k] = DFT(\tilde{y}_R[n]). \tag{4}$$

We emphasize that the computation complexity of FPNC CFO compensation is exactly the same as that of point-to-point communication.

B. FPNC Channel Estimation

In this subsection, we present the channel estimation and tracking method for FPNC. Note that CFO compensation was performed on the time-domain signal. For channel estimation, however, we are interested in the channel gains for different subcarriers in the frequency domain. This means that channel estimation will be performed after DFT. Thus, in the following we look at the signal after CP removal and DFT.

For FPNC channel estimation, we use the LTS to obtain a first estimate. Pilots are used to obtain additional estimates for later OFDM symbols within the same frame. In the following, we consider channel estimation of $H_A[k]$. Estimation of $H_B[k]$ is performed similarly.

For channel estimation based on LTS, define one FPNC LTS unit of node A (i.e., with respect to Fig. 2, one unit is lts_A) in the frequency domain as $X_A^{LTS}[k]$, where $k = 0, \ldots, N-1$. Based on the first unit of lts_A the received frequency domain LTS_A (i.e., $\tilde{Y}_R^{LTS_A}[k] = DFT\left(\tilde{y}_R^{LTS_A}[n]\right)$), we perform channel estimation of $H_A[k]$ as follows:

$$\hat{H}_A[k] = \frac{\tilde{Y}_R^{LTS_A}[k]}{X_A^{LTS}[k]}. \tag{5}$$

As mentioned in Section III, each LTS contains two identical units in our design. The uplink channel gain $H_A[k]$ between node A and relay R is estimated by taking the average of the two units results

$$\tilde{H}_A[k] = \left(\hat{H}_A[k] + \hat{H}_A[k+N]\right)/2. \tag{6}$$

In general, the channel may have changed from the first OFDM symbol to the last OFDM symbol within the same frame. The estimate based on LTS in (6) applies only for the earlier symbols. Pilots are used to track the channel changes for later symbols. Our pilot design was shown in Fig. 4. In each FPNC OFDM symbol, there are two pilots per end node. Note from Fig. 4 that the two pilots of node A and the two pilots are node B are positioned at different subcarriers and non-overlapping in the frequency domain. Therefore, we could separately track the changes in $H_A[k]$ and $H_B[k]$. In the following, we consider the tracking of $H_A[k]$. Tracking of $H_B[k]$ can be done similarly.

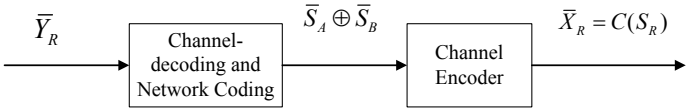

Fig. 5. Link-by-link channel-coded PNC, including channel-decoding and network coding (CNC) process and channel encoding.

Let k' and k'' denote the subcarriers occupied by the two pilots of A. Consider OFDM symbol m. Let $\tilde{Y}_R^m[k']$ and $\tilde{Y}_R^m[k'']$ be the received signal in the frequency domain. Because the pilots of A and B do not overlap, $\tilde{Y}_R^m[k']$ and $\tilde{Y}_R^m[k'']$ contain only signals related to the pilots of A. We first multiply $\tilde{Y}_R^m[k']$ and $\tilde{Y}_R^m[k'']$ by $(\tilde{H}_A[k'])^{-1}$ and $(\tilde{H}_A[k''])^{-1}$ obtained from (6), respectively. Let $P_A[k']$ and $P_A[k'']$ be the two pilots. Then, we compute

$$\begin{aligned}\Delta\tilde{H}_A^m[k'] &= (\tilde{H}_A[k'])^{-1}\tilde{Y}_R^m[k'] \Big/ P_A[k'],\\ \Delta\tilde{H}_A^m[k''] &= (\tilde{H}_A[k''])^{-1}\tilde{Y}_R^m[k''] \Big/ P_A[k''].\end{aligned} \tag{7}$$

After that, we perform linear fitting to obtain $\Delta\tilde{H}_A^m[k]$ for $k \neq k', k''$, as follows:

$$\Delta\tilde{H}_A^m[k] = \Delta\tilde{H}_A^m[k'] + \left(\frac{\Delta\tilde{H}_A^m[k''] - \Delta\tilde{H}_A^m[k']}{k'' - k'}\right)(k - k'). \tag{8}$$

To obtain the final channel estimation for the m-th OFDM symbol, we compute

$$H_A^m[k] = \tilde{H}_A[k] \cdot \Delta\tilde{H}_A^m[k]. \tag{9}$$

C. FPNC Mapping

For reliable communication, channel coding should be used. Channel coding in PNC systems can be either done on an end-to-end basis or a link-by-link basis [4], [11]. The latter generally has better performance because the relay performs channel decoding to remove noise before forwarding the network-coded signal.

The basic idea in link-by-link channel-coded PNC is shown in Fig. 5. It consists of two parts. Let $\bar{Y}_R$ denote the vector representing the overall channel-coded overlapped frames received by relay R. The operation performed by the first part is referred to as the Channel-decoding and Network-Coding (CNC) process in [11]. It maps $\bar{Y}_R$ to $\bar{S}_A \oplus \bar{S}_B$, where $\bar{S}_A$ and $\bar{S}_B$ are the vectors of source symbols from nodes A and B, respectively, and the $\oplus$ operation represents symbol-by-symbol XOR operation across corresponding symbols in $\bar{S}_A$ and $\bar{S}_B$. Note that the number of symbols in $\bar{Y}_R$ is more than the number of symbols in $\bar{S}_A \oplus \bar{S}_B$ because of channel coding. Importantly, CNC involves both channel decoding and network coding. In particular, CNC channel-decodes the received signal $\bar{Y}_R$ not to $\bar{S}_A$ and $\bar{S}_B$ individually, but to their XOR. The second part can be just any conventional channel coding operation that channel code $\bar{S}_A \oplus \bar{S}_B$ to $\bar{X}_R = C(\bar{S}_A \oplus \bar{S}_B)$ for broadcast to nodes A and B, where $C(*)$ is the channel coding operation.

As mentioned in [11] and [4], the CNC component is unique to the PNC system, and different designs can have different performance and different implementation complexity. We refer the interested readers to [4] for a discussion on different CNC designs.

In this paper, we choose a design that is amenable to simple implementation, as shown in Fig. 6. We refer to this CNC design as XOR-CD. In this design, any linear channel code can be used. In our implementation, we choose to use the convolutional code. In XOR-CD, the channel-decoding and network coding operations in CNC are performed in a disjoint manner. As shown in Fig. 6, based on the CFO-compensated $\tilde{Y}_R[k]$ obtained as in (4), we obtain the overall vector $\bar{Y}_R = (Y_R[k])_{k=0,1,...}$. We then perform symbol-wise PNC mapping to get an estimate for the the channel-coded XOR vector $\bar{X}_A \oplus \bar{X}_B = (X_A[k] \oplus X_B[k])_{k=0,1,...}$, where $\bar{X}_A = (X_A[k])_{k=0,1,...}$ and $\bar{X}_B = (X_B[k])_{k=0,1,...}$ are the channel-coded vectors from A and B, respectively. We assume the same linear channel code is used at nodes A, B, and R. Note that since we adopt the convolutional code, $C(*)$ is linear. Therefore, we have $\bar{X}_A \oplus \bar{X}_B = C(\bar{S}_A) \oplus C(\bar{S}_B) = C(\bar{S}_A \oplus \bar{S}_B)$, and thus the same Viterbi channel decoder as used in a conventional point-to-point communication link can be used in the second block of Fig. 6.

The mapping in the first block in Fig. 6 could be performed as follows. Based on the channel gains estimated in (9), we could perform the XOR mapping for the k-th subcarrier in the m-th OFDM symbol (assuming BPSK modulation) according to the decision rule below:

$$\exp\left\{-\frac{|Y_R^m[k]-H_A^m[k]-H_B^m[k]|^2}{2\sigma^2}\right\} + \exp\left\{-\frac{|Y_R^m[k]+H_A^m[k]+H_B^m[k]|^2}{2\sigma^2}\right\} \underset{X_R^m[k]=-1}{\overset{X_R^m[k]=1}{\gtrless}} \exp\left\{-\frac{|Y_R^m[k]+H_A^m[k]-H_B^m[k]|^2}{2\sigma^2}\right\} + \exp\left\{-\frac{|Y_R^m[k]-H_A^m[k]+H_B^m[k]|^2}{2\sigma^2}\right\}, \tag{10}$$

where we have assumed Gaussian noise with variance σ^2. The computation complexity in (10)[3], however, is large. In our implementation, we adopt a simple "log-max approximation" [15] (i.e., $\log(\sum_i \exp(z_i)) \approx \max_i z_i$) that yields the following decision rule:

$$\min\left\{|Y_R^m[k] - H_A^m[k] - H_B^m[k]|^2, |Y_R^m[k] + H_A^m[k] + H_B^m[k]|^2\right\} \underset{X_R^m[k]=1}{\overset{X_R^m[k]=-1}{\gtrless}} \min\left\{|Y_R^m[k] + H_A^m[k] - H_B^k[k]|^2, |Y_R^m[k] - H_A^m[k] + H_B^m[k]|^2\right\}. \tag{11}$$

This decision rule can also be interpreted as in Table I, where

$$U \triangleq \underset{U\in\{\pm H_A^m[k] \pm H_B^m[k]\}}{\arg\min}\left\{|Y_R^m[k] - U|^2\right\}. \tag{12}$$

Note here that this decision rule could be used even for non-Gaussian noise. This is because (12) corresponds to finding the nearest point in the constellation map (constructed by combining the two end nodes' channel gains).

[3]Note that (10) is similar to (7) in Ref. [14], except that here we allow for the possibility that $|H_A^m[k]| \neq |H_B^m[k]|$

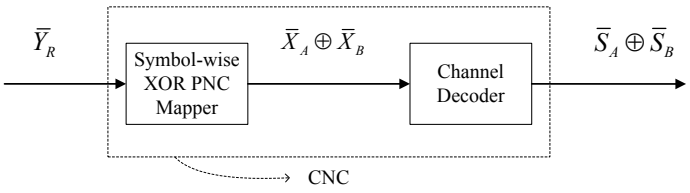

Fig. 6. XOR-CD design for CNC.

TABLE I
XOR MAPPING WITH BPSK MODULATION IN FPNC.

$U = \arg\min_{U \in \{\pm H_A^m[k] \pm H_B^m[k]\}} \left\{ \left\| Y_R^m[k] - U \right\|^2 \right\}$	$X_R^m[k] = X_A^m[k] \oplus X_B^m[k]$
$H_A^m[k] + H_B^m[k]$	1
$H_A^m[k] - H_B^m[k]$	-1
$-H_A^m[k] + H_B^m[k]$	-1
$-H_A^m[k] - H_B^m[k]$	1

Based on the XORed samples detected using the decision rule of Table I, we then perform the channel decoding to get the XORed source samples. In our implementation, we use a Viterbi decoder with hard input and hard output. In general, a soft Viterbi algorithm could also be used for potentially better BER performance [16].

V. EXPERIMENTAL RESULTS

This section presents details of our FPNC implementation over the software radio platform and the experimental results.

A. FPNC Implementation over Software Radio Platform

We implement FPNC in a 3-node GNU Radio testbed, with Software Defined Radio (SDR). The topology is shown in Fig. 1. Each node is a commodity PC connected to a USRP GNU radio [17].

*1) **Hardware**:* We use the Universal Software Radio Peripheral (USRP) [18] as our radio hardware. Specifically, we use the XCVR2450 daughterboard operating in the 2.4/5GHz range as our RF frontend. We use the USRP1 motherboard for baseband data processing. The largest bandwidth that USRP1 could support is 8MHz. In our experiment, we use only use half of the total bandwidth for FPNC (i.e., 4MHz bandwidth).

*2) **Software**:* The software for baseband signal processing is based on the open source of GNURadio project [17]. We build our system by modifying the 802.11g transmitter implementation in the FTW project [19]. The FTW project [20], however, does not have a 802.11g receiver. Therefore, we develop our own OFDM receiver, designed specifically to tackle various issues in the FPNC system, such as CFO estimation and compensation, channel estimation, and CNC processing as presented in Section IV.

B. Experimental Results

We conduct our experiments over the channel one of 802.11g, with 2.412GHz being the central frequency. For each transmitter power level (we vary the SNR from 5dB to 20dB), we transmit 1000 packets (using BPSK modulation) and examine the resulting BER performance. Both the symbol-synchronous and symbol-asynchronous cases are investigated. The packet length is 1500Bytes, which is a normal Ethernet frame size.

1) Time-Synchronous FPNC versus Time-Asynchronous F-PNC: In Section II, we show that as long as the *Delay-Spread-Within-CP requirement* is satisfied, FPNC will not have asynchrony in the frequency domain. Of interest is whether this reduces the asynchrony penalty in practice. In our first set of experiments, we investigate this issue. We study both unchannel-coded as well as channel-coded FPNC systems.

To create different levels of time asynchrony, we adjust the positions of the end nodes. One of the set-ups corresponds to the perfectly synchronized case (the STS correlation has only ten peaks in the perfectly synchronized case: see Section III. Fig. 7(a) shows the BER-SNR curves for the synchronous case, and Fig. 7(b) shows the curves for the asynchronous case with eight samples offset between the early and late frames. Note that this asynchrony still satisfies the Delay-Spread-Within-CP requirement because the CP has of 16 samples. We find that the performance results of the asynchronous cases with other time offset to be similar, and we therefore present the results of the eight-sample offset only.

From Fig. 7(a) and (b), we see that the asynchronous FPNC has essentially the same BER performance as that of the synchronous FPNC. Hence, we conclude that FPNC is robust against time asynchrony as far as BER performance is concerned.

2) FPNC versus point-to-point transmission: To better analyze the BER performance of FPNC, we benchmark it with a standard point-to-point transmission, in which we only allow node A to communication with relay R. It is shown in Fig. 7(c) that the unchannel-coded BER performance loss for FPNC is 4-5 dB, while its channel-coded BER performance loss is less than 3 dB, relative to point-to-point transmission. If we use the guideline that the common decodable 802.11 link usually works at an SNR regime that is higher than 20 dB [5], [21], we can conclude that our FPNC implementation has very good performance in this regime (with BER lower than 10^{-5}) that it could nearly double the throughput of a TWRC compared to the traditional scheduling method (i.e., FPNC reduces the needed time slots for TWRC from four to two).

VI. CONCLUSION AND FUTURE WORK

This paper presents the first implementation of a PNC system in which the relay performs the XOR mapping on the simultaneously received signals as originally envisioned in [1]. In particular, in our implementation, the XOR mapping is performed in the frequency domain of an OFDM PNC system. We refer to the OFDM PNC system as FPNC. The

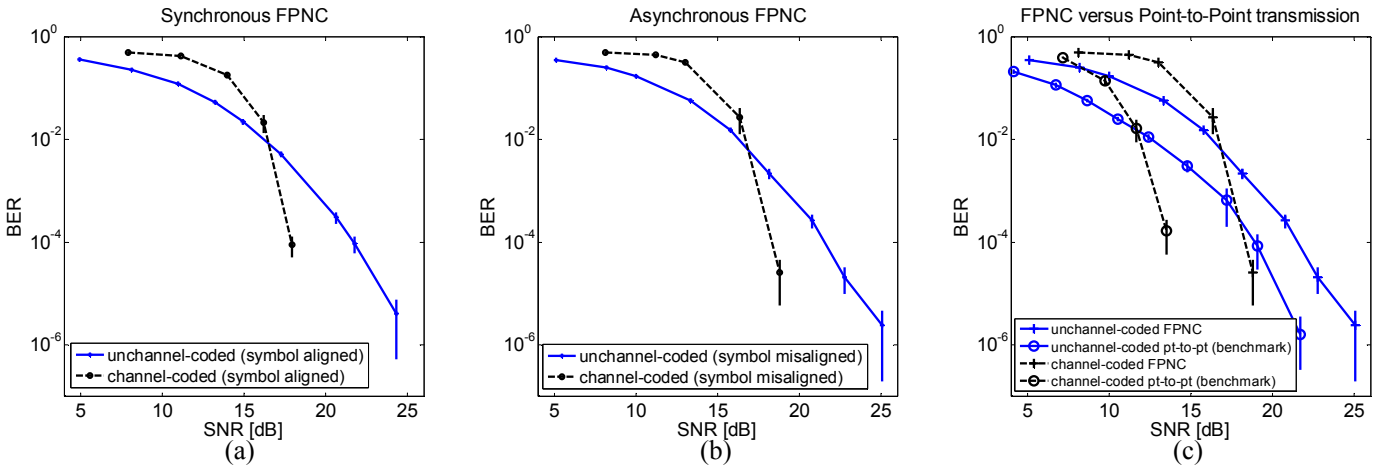

Fig. 7. Experimental results for FPNC: (a) BER of FPNC with synchronous samples; (b) BER of FPNC with asynchronous samples; (c) BER comparison of FPNC versus a point-to-point transmission (as a benchmark). The 95% confidence intervals are marked in the figures. Note that the BER of FPNC here is related to whether the XOR bit is decoded correctly, not whether the individual bits from the two end nodes are decoded correctly.

implementation of FPNC requires us to tackle a number of implementation challenges, including carrier frequency offset (CFO) compensation, channel estimation, and FPNC mapping. A major advantage of FPNC compared with PNC in the time domain is that FPNC can deal with the different arrival times of the signals from the two end nodes in a natural way. To validate the advantage of FPNC, we present experimental results showing that time-domain symbol asynchrony does not cause performance degradation in FPNC.

Going forward, there are many rooms for improvement in our FPNC implementation. In this paper, when faced with alternative design choices, we opt for implementation simplicity than performance superiority. For example, we choose to use a simple PNC mapping method called XOR-CD in this paper, which is simple to implement but has inferior performance compared with other known methods [4] in the low SNR regime. In addition, our implementation exercise reveals a number of problems with no good theoretical solutions yet, and further theoretical analysis is needed; in such cases, we use simple heuristics to tackle the problems. For example, CFO compensation for FPNC is an area that is not well understood yet, because we have to deal with CFOs of more than one transmitter relative to the receiver. In this paper, we simply compensate for the mean of the CFOs of the two end nodes. Better methods await further theoretical studies. Last but not least, we base our design on the 802.11 standard to a large extent with only moderate modifications. If we do not limit our design within the framework of 802.11, there could be other alternatives with potentially better performance.

REFERENCES

[1] S. Zhang, S. C. Liew, and P. P. Lam, "Hot topic: Physical layer network coding," in *Proc. ACM MOBICOM*, 2006.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Infomation Theory*, vol. 46, pp. 1204–1216, 2000.

[3] S. Zhang, S. C. Liew, and L. Lu, "Physical layer network coding schemes over finite and infinite fields," in *Proc. IEEE Glob. Telecom. Conf. (GLOBECOM)*, 2008.

[4] S. C. Liew, S. Zhang, and L. Lu, "Physical-layer network coding: Tutorial, survey, and beyond," submitted to *Physical Communication,* available at: http://arxiv.org/abs/1105.4261.

[5] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: Analog network coding," in *Proc. ACM SIGCOMM*, 2007.

[6] L. Lu and S. C. Liew, "Asynchronous physical-layer network coding," to appear in *IEEE Trans. Wireless Commun.*

[7] P. Popovski and H. Yomo, "Physical network coding in two-way wireless relay channels," in *Proc. IEEE Int. Conf. on Comm. (ICC)*, 2007.

[8] L. Lu, T. Wang, S. C. Liew, and S. Zhang, "Implementation of physical-layer network coding," *Technical Report,* available at http://arxiv.org/abs/1105.3416.

[9] F. Rossetto and M. Zorzi, "On the design of practical asynchronous physical layer network coding," in *IEEE 10th Workshop on SPAWC*, 2009.

[10] IEEE 802.11-2007, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," http://standards.ieee.org/getieee802/download/802.11-2007.pdf.

[11] S. Zhang and S. C. Liew, "Channel coding and decoding in a relay system operated with physical-layer network coding," *IEEE Jour. Select. Areas in Comm.*, vol. 27, no. 5, pp. 788–796, Jun. 2009.

[12] H. Mehrpouyan and S. Blostein, "Bounds and algorithms for multiple frequency offset estimation in cooperative networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 4, pp. 1300–1311, 2011.

[13] X. Li, F. Ng, and T. Han, "Carrier frequency offset mitigation in asynchronous cooperative OFDM transmissions," *IEEE Trans. Signal Proccessing*, vol. 56, no. 2, pp. 675–685, 2008.

[14] L. Lu, S. C. Liew, and S. Zhang, "Optimal decoding algorithm for asynchronous physical-layer network coding," in *Proc. IEEE Int. Conf. on Comm. (ICC)*, Jun. 2011.

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[16] B. Sklar, *Digital Communications: Fundamentals and Applications (2nd Edition)*. Prentice Hall PTR, 2003.

[17] G. FSF., "GNU Radio - Gnu FSF Project," http://gnuradio.org/redmine/wiki/gnuradio.

[18] Ettus Inc., "Universal software radio peripheral," http://www.ettus.com/.

[19] A. Costantini, P. Fuxjaeger, D. Valerio, P. Castiglione, and G. Zacheo, "FTW IEEE802.11a/g/p OFDM frame encoder," https://www.cgran.org/wiki/ftw80211ofdmtx.

[20] P. Fuxjër, A. Costantini, D. Valerio, P. Castiglione, G. Zacheo, T. Zemen, and F. Ricciato, "IEEE 802.11p transmission using gnuradio," in *Proc. IEEE WSR*, 2010.

[21] J. Geier, "SNR cutoff recommendations," http://www.wi-fiplanet.com/tutorials/article.php/3468771.