Implementation and Experimental Results of Superposition Coding on Software Radio

R. K. Ganti, Z. Gong, M. Haenggi, C. Lee, S. Srinivasa, D. Tisza, S. Vanka and P. Vizi
{rganti,zgong,mhaenggi,clee14,ssriniv1,dtisza,svanka,pvizi}@nd.edu
Network Communications & Information Processing Lab
Department of Electrical Engineering, University of Notre Dame
Notre Dame, IN 46556, USA

***Abstract*—Superposition coding is a well-known capacity-achieving coding scheme for stochastically degraded broadcast channels. Although well-studied in theory, it is important to understand issues that arise when implementing this scheme in a practical setting. In this paper, we present a software-radio based design of a superposition coding system on the GNU Radio platform with the Universal Software Radio Peripheral acting as the transceiver frontend. We also study the packet error performance and discuss some issues that arise in its implementation.**

***Keywords*– GNU Radio, Orthogonal Frequency Division Multiplexing, Software-Defined Radio, Superposition Coding, Universal Software Radio Peripheral.**

I. Introduction and Motivation

The past few decades have witnessed several technological advances in wireless communication. Key research breakthroughs such as low-density parity-check (LDPC) and turbo codes, multiple-input multiple output (MIMO), orthogonal frequency division multiplexing (OFDM), and code-division multiple access (CDMA) have been incorporated into several communications standards. In order to meet the ever-growing demand for wireless services, there has also been an increased research focus on implementing multi-user communication techniques that could form the basis for the next-generation of wireless technologies.

One such technique of potential benefit in scenarios that require a single transmitter to communicate with multiple receivers (e.g., a cellular downlink) is Superposition Coding (SC), which is known to be the optimal (capacity-achieving) communication strategy for stochastically degraded[1] broadcast channels [1]. In principle, SC works by exploiting the relative disparities in channel quality to different receivers to optimally allocate transmit power among them, while using the entire bandwidth for each receiver. Employing Successive Decoding (SD), receivers decode the messages meant for all those receivers whose channel quality is poorer than theirs, before decoding their own message. However, the information-theoretic framework does not consider several issues of practical importance - finite design complexity, hardware non-idealities (e.g., carrier frequency offset, phase noise) or higher layer issues like medium access. This motivates the experimental study of SC in an attempt to understand its practical limitations. To the best of our knowledge, this is the first such effort.

Additionally, the SC system design and implementation has the following benefits.

1) The SD procedure is also beneficial in other scenarios. For example, it is the optimal decoding scheme for the multiple-access channel (e.g., cellular uplink) and has been shown to improve link throughput in ad hoc scenarios (see e.g., [2]). Thus, our SD module in the receiver can be reused for these scenarios.
2) SC sets the foundation to experiment with more sophisticated medium access schemes that are point-to-multipoint, or even multipoint-to-multipoint

Conventionally, the testing of a theoretical idea such as the SC technique in practice would require the design of new hardware and would involve a considerable design effort. Such a long development cycle, however, is not conducive to rapid prototyping and verification of new protocol designs. An alternative design paradigm is the so-called software-defined radio (SDR) [3], where the design flow is mostly in software, thus making it useful for fast prototyping of new communication techniques by leveraging the inherent flexibility of software-based systems compared to their hardware-based counterparts.

GNU Radio is an open source software development toolkit that provides the signal processing blocks to implement SDRs, and it interfaces with a Universal Software Radio Peripheral (USRP) board that serves as an analog and RF front-end [4]. It has been used as a cognitive radio platform [5]–[8] or a prototyping system [9], [10]. Some notable examples of USRP-based testbeds are UT Austin's Hydra [11] and the mesh network platform by Bell Labs and Microsoft Research [12]. Recently, the MAC layer for an SC-based physical layer was implemented [13]. However, to the best of our knowledge, the USRP implementation of the SC PHY layer is yet to be investigated.

In this paper, we describe the realization of OFDM-based SC on GNU Radio and the study of SC through experiments. We focus on using GNU Radio/USRP as a platform to evaluate and compare the performance of SC and a time-division multiplexing (TDM) system.

[1]Degraded implies that the channel qualities to the individual users can be ordered.

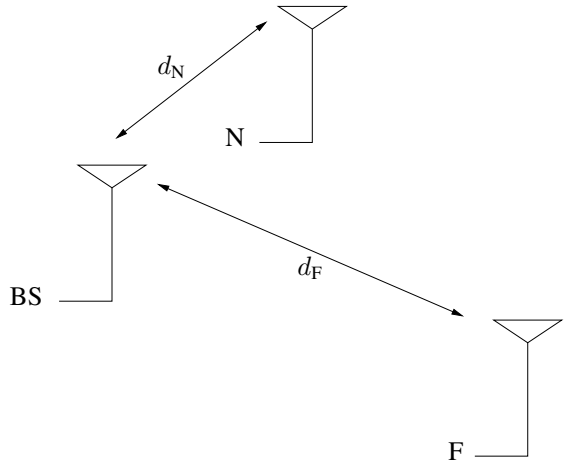

Fig. 1. The base station wishes to communicate with two users N and F, which are located at distances d_{N} and d_{F} away, $d_{\mathrm{N}} \ll d_{\mathrm{F}}$.

II. How Does Superposition Coding Work?

Consider a base station (BS) that wishes to communicate with two users (see Fig. 1). Now, suppose that one of the users, N (who we call the "near" user) is much closer to the BS than the other, F (the "far" user). Such a scenario can arise, for example, in a cellular system where one user is close to the base station and the other near the cell edge. In information theory, this scenario is modeled as a *stochastically degraded* Gaussian broadcast channel (BC) [1].

The capacity region for the stochastically degraded Gaussian BC, defined as the set of all simultaneously achievable communication rate pairs, is known. Furthermore, every point in the capacity region may be achieved using SC at the transmitter, and *Successive Decoding* (SD) at the near user. Accordingly, the transmitter superposes and simultaneously transmits the encoded messages for both users. The near user, who has a better channel can always decode messages meant for the far user and then effectively cancel out the interference due to the signal(s) meant for the far user, before decoding its own messages. The far user decodes its message by treating the near user message as noise. This approach can be extended to a K-user downlink channel ($K \geq 2$).

Formally, denote by X_{N} (X_{F}) the encoded near (far) user's signal, each with unit power. Let the total power of the base station BS be unity, of which a fraction α (resp. $1-\alpha$) is given to the far (resp. near) user. Since the channel from BS to N is AWGN, the near user observes

$$Y_{\mathrm{N}} = \sqrt{\alpha} X_{\mathrm{F}} + \sqrt{1-\alpha} X_{\mathrm{N}} + W_{\mathrm{N}},$$

where W_{N} denotes AWGN (with variance σ_{N}^2). Since the channel from BS to N is stronger than that from BS to F, the near user N can decode the far user's message. After canceling out the interference αX_{F}, the near user observes

$$Y_{\mathrm{N}} - \sqrt{\alpha} X_{\mathrm{F}} = \sqrt{1-\alpha} X_{\mathrm{N}} + W_{\mathrm{N}},$$

which may be used to decode its own message. The far user, on the other hand, can only decode its own message but not the near user's message.

Employing SC along with interference cancellation, the following rate pair $(R_{\mathrm{N}}, R_{\mathrm{F}})$ can be achieved. The right hand side terms together represent the boundary of the capacity region [1]:

$$R_{\mathrm{N}} < \log_2 \left(1 + \frac{1-\alpha}{\sigma_{\mathrm{N}}^2}\right) \quad \text{bps/Hz}$$

and

$$R_{\mathrm{F}} < \log_2 \left(1 + \frac{\alpha}{1-\alpha+\sigma_{\mathrm{F}}^2}\right) \quad \text{bps/Hz},$$

where $\sigma_{\mathrm{F}}^2 > \sigma_{\mathrm{N}}^2$ represents the AWGN variance at the far user. In particular, SC can provide a very reasonable rate to the near user, while achieving a rate close to the single-user capacity for the far user.

III. Implementation of SC

We now describe the top-level architecture of our system and the physical layer frame structure. A detailed evaluation of the performance of this architecture can be found in [14].

A. Platform

We adopt a software-centric paradigm to implement our design. All the signal processing algorithms at the baseband are implemented digitally using GNU Radio (revision 10923) on a general-purpose computer. This approach also allows us to use several built-in libraries that come with the GNU Radio package, including the USRP interface, which acts as the RF and analog frontend.

The design uses OFDM as the transmission scheme. OFDM offers a high degree of bandwidth scalability and implementation flexibility, especially given its relative ease of time- and frequency-synchronization, channel estimation and equalizer design [15].

B. System Parameters and Packet Structure

The system parameters used in the experiment are provided in Table I. It can be shown that for fixed coding rates and M-ary QAM modulations to both users, choosing $\alpha = 0.8$ is optimal in terms of maximizing the reliability to the near user [14]. The physical layer frame structure is depicted in Fig. 5. The preamble sequence is used primarily for (coarse) frequency and timing synchronization and is essentially a pseudo-random training sequence of length 24 μs (TS1) repeated twice. The channel estimation symbols are employed for performing equalization and are formed by repeating another pseudo-random training sequence (TS2) of length 16 symbols (including cyclic prefix) three times. The payload is sent over the 8 subcarriers reserved for data transmission. Four subcarriers are reserved for pilots to aid in fine frequency offset compensation, and the remaining four are guard subcarriers.

Center Frequency	903 MHz
System Bandwidth	1 MHz
Payload Size	0.5kBytes (incl. 4-Byte CRC)
Transmission Scheme	16-tone OFDM
Data Tones	8
Pilot Tones	4
Null Tones	4
CP Length	4μs
Far User Modulation	BPSK
Near User Modulation	BPSK
Gen. Poly. for Conv. Code	$[133, 171]$
Far User Code Rate	1/2
Near User Code Rate	1/2
Power allocation α	0.8

TABLE I
PARAMETERS USED IN THE EXPERIMENT.

C. Transmitter Operation

The transmitter block diagram is shown in Fig. 2 (see next page). The payloads (near and far user data in bits) are provided to the physical layer by the link layer. Each user's payload is assigned a modulation type, code rate, and the fraction of the transmit power. As shown in Fig. 2, these payloads are separately encoded based on their respective parameters prior to time-domain conversion. In time domain, the signals are weighted by the fraction of power allocation factors ($\sqrt{1-\alpha}$ for the near user signal, and $\sqrt{\alpha}$ for the far user signal), added and transmitted. The link layer interacts with the physical layer through Unix sockets, while the physical layer is implemented as a single GNU Radio signal processing block. The transmitter interacts with the USRP via GNU Radio driver software.

D. Receiver Operation

The receiver block diagram is shown in Fig. 3. The receiver front end is implemented in hardware using the USRP, while the physical layer is implemented using a single GNU Radio signal processing block. The USRP downconverts and digitizes signals in the frequency band of operation. Initially, all blocks in the receiver except the timing and frequency synchronization block are inactive. This block essentially consists of a windowed correlator. Our algorithm is similar to the well-known Schmidl-Cox algorithm [16]. If the signal in the correlator window satisfies correlation properties, this block declares a valid packet along with the location of the channel estimation training sequence (see Fig. 3). Moreover, our preamble structure allows us to use the phase of the correlator to estimate the frequency offset with respect to the transmitter. Since this estimate can have a large variance, we call this the Coarse Frequency Estimate. The coarse frequency correction unit uses this coarse estimate supplied by the synchronization unit to correct all future samples in the packet for frequency offset. Using the location of the channel estimation training sequence, the coarse-offset corrected samples are now fed to the channel estimator, which in turn estimates the channel. In this implementation we have assumed that the receiver knows the modulation, coding and power allocation information, via a reliable channel like the Ethernet. Hence after the channel estimation is complete, the receiver is now ready to decode the payload(s).

Since the locations of the OFDM symbols are known in time, the cyclic prefix (CP) is removed from the incoming samples, and blocks of 16 (equal to the number of subcarriers) samples are buffered and processed by the FFT routine. Once in frequency domain, the pilot subcarriers are identified and using the channel estimates obtained earlier, the pilot tones are used to estimate and correct the payload symbols for any residual (i.e., "fine") frequency offset. These offset-corrected symbols are equalized using the channel estimates to yield hard decision code bits (or their reliabilities for soft-decoding). Until this point, the receiver operation is quite similar to a typical IEEE 802.11a receiver.

The near user uses the successive decoding procedure to decode its payload bits (see Fig. 3, next page). This procedure involves decoding the far user bits, re-encoding and modulating the payload to reconstruct the far user component for each subcarrier of the received signal. This is subtracted subcarrier-wise from the received signal; its payload is then extracted from the residual signal. In contrast, the far user decodes its payload bits by simply treating the near user signal as interference. In both cases, the decoded packets are sent to their respective queues. The MAC checks the CRC for each received packet and declares an error if the CRC fails.

IV. EXPERIMENTAL RESULTS

The experimental setup consists of a transmitter and two (near and far) receivers. Each transceiver consists of a software radio along with a USRP2 and a PC. Fig. 4 shows the experimental setup with the relative locations of the three USRP2's.

A. PER versus SNR

Fig. 6 plots the Packet Error Rate (PER) versus SNR performance of the SC system for both the near and far users when $\alpha = 0.8$. Both the far- and near-user payloads are taken to be 0.5 kBytes long, and the code rate for both users is $1/2$ (see Table I). The experiment was conducted indoors in our laboratory. Since more power is allocated to the far user, the far user has better PER. The figure shows that our SC system gives very good performance (PER achieves 10^{-2}). Clearly, the far user has a lower PER than the near user (for the same SNR).

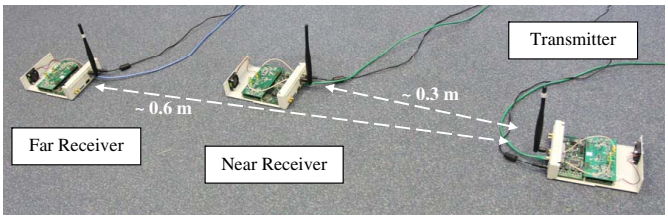

Fig. 4. Depiction of the relative locations of the radios in the implementation set-up. These distances were chosen so that both the receivers lie in the far-field of the transmitter and observe different SNRs.

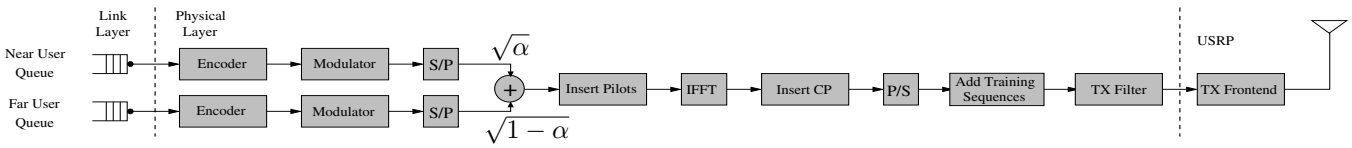

Fig. 2. Block diagram of the transmitter. The Serial-to-Parallel (S/P) function accepts serial data at its input and outputs data as blocks of the desired size (in this case the number of subcarriers, 16). The Parallel-to-Serial (P/S) function inverts this operation. CP stands for Cyclic Prefix.

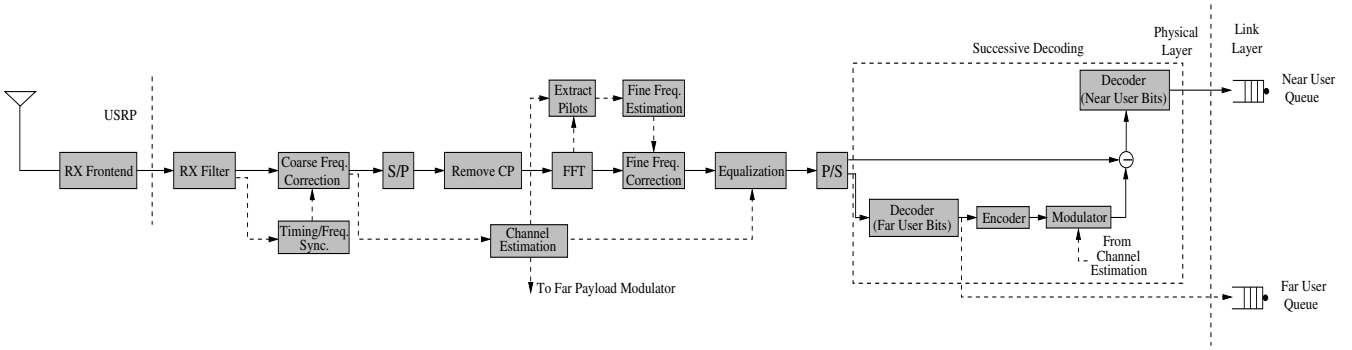

Fig. 3. Block diagram of the receiver.

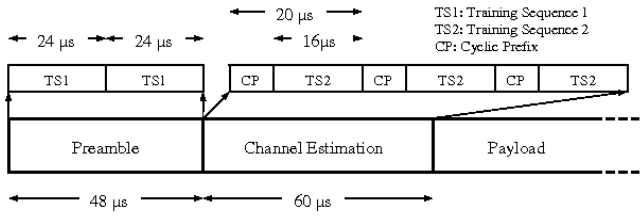

Fig. 5. Physical layer frame structure. The preamble sequence is chosen to allow for robust on-air detection of the frame, along with estimation of the timing and frequency offsets. The training sequence TS2 that follows is repeated to improve noise-averaging, thereby improving the accuracy of the channel estimates. The payload is decoded based on the modulation type, code rate and the fraction of power assigned to each user by the transmitter. These parameters are assumed to be known to the receivers via a reliable channel (e.g., Ethernet).

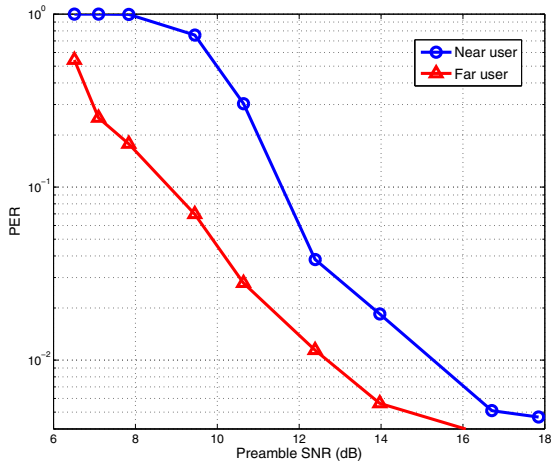

Fig. 6. PER versus SNR for the near and far users for $\alpha = 0.8$. Note that the SNR is defined as the measured preamble power divided by the noise power. Since the far user is allocated more power than the near user, its PER is lower. Both the far and the near user achieve a good performance (PER<10^{-2}) at moderate SNR.

B. Lessons Learned

We now briefly overview some issues involved in implementing the SC system using GNU Radio.

- The GNU Radio scheduler was designed for a flow-based framework, i.e., each block operates on a stream of data rather than packets of data. This makes the design of a frame-based system quite challenging. Moreover, implementing feedback loops between signal processing blocks is cumbersome. One solution is to implement all the functions in one big GNU Radio block, but this sacrifices the reusability of the functional blocks. From our experience, implementing all the functions in one GNU Radio block significantly simplifies the design.
- The effective bandwidth of the USRP1 (the first generation USRP) is much smaller than that set by the user. The cause of this problem lies in the highly non-ideal transmit path implementation of the USRP1. The DAC's on the transmit path are designed to operate at a fixed frequency of 128 MHz [4]. Therefore, any digitally synthesized signal at a lower bandwidth to be input to the DAC's must be interpolated to 128 MHz. However, we observed that the USRP1 uses a rather simplistic scheme to implement this interpolation, so it shows a poor passband response. Such a frequency response causes significant degradation of sub-carrier SNR as one moves away from the DC subcarrier. Similar problems were reported recently elsewhere [17]. The solution is to implement a software-based filter and do part of the interpolation in software. In USRP2 (the

second generation USRP), this problem seems to have been alleviated, and that is why we use USRP2 for the experiments.

V. Conclusions

We have presented a design that leverages the flexibility of the GNU Radio/USRP platform to implement physical layer of a communication system that uses superposition coding. The performance of the implementation was measured using packet error rates for each of the streams. We found that the limitations of the USRP hardware have implications for the design of training sequences, and the software environment imposes constraints on the degree of modularity allowed in the system implementation.

VI. Acknowledgment

The support of NSF (grant CNS 04-47869) and DTRA (grant N00164-07-8510) is gratefully acknowledged. The authors would also like to thank Dr. Robert Heath for helpful discussions.

References

[1] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed., John Wiley & Sons, Inc., 2006.

[2] S. P. Weber, J. G. Andrews, X. Yang, G. de Veciana, "Transmission Capacity of Wireless Ad Hoc Networks With Successive Interference Cancellation," *IEEE Trans. Info. Theory*, vol. 53, No. 8, pp. 2799-2814, 2007.

[3] J. Mitola III, "Software radios: Survey, critical evaluation and future directions," *IEEE Aerospace and Electronic Systems Magazine*, vol. 8, no. 4, pp. 25-36, 1993.

[4] *GNU Radio*, http://www.gnu.org/software/gnuradio/.

[5] F. Ge, Q. Chen, Y. Wang, C. W. Bostian, T. W. Rondeau, and B. Le, "Cognitive radio: from spectrum sharing to adaptive learning and reconfiguration," in *Proceedings of IEEE Aerospace Conference*, pp. 1-10, Mar. 2008.

[6] C. Sokolowski, M. Petrova, A. de Baynast, and P. Mahonen, "Cognitive radio testbed: exploiting limited feedback in tomorrow's wireless communication," in *Proceedings of IEEE International Conference on Networks Communications Workshops*, pp. 493-498, May 2008.

[7] R. Dhar, G. George, A. Malani, and P. Steenkiste, "Supporting integrated MAC and PHY software development for the USRP SDR," in *Proceedings of 1st IEEE Workshop on Networking Technologies for Software Defined Radio Networks*, pp. 68-77, Sept. 2006.

[8] M. L. Dickens, B. P. Dunn, and J. N. Laneman, "Design and implementation of a portable software radio," *IEEE Communications Mag*azine, vol. 46, no. 8, pp. 58-66, 2008.

[9] X. Li, W. Hu, H. Yousefizadeh, and A. Qureshi, "A case study of a MIMO SDR implementation," in *Proceedings of IEEE Military Communications Conference (MILCOM)*, pp. 1-7, Nov. 2008.

[10] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa, "On the performance of IEEE 802.11 under jamming," in *Proceedings of The 27th IEEE Conference on Computer Communications (INFOCOM)*, pp. 1265-1273, Apr. 2008.

[11] K. Mandke, S.-H. Choi, G. Kim, R. Grant, R. C. Daniels, W. Kim, R. W. Heath, and S. M. Nettles, "Early results on Hydra: a flexible MAC/PHY multihop testbed," in *Proceedings of IEEE 65th Vehicular Technology Conference (VTC2007-Spring)*, pp. 1896-1900, Apr. 2007.

[12] R. Alimi, L. Li, R. Ramjee, H. Viswanathan, and Y. R. Yang, "iPack: in-network packet mixing for high throughput wireless mesh networks," in *Proceedings of The 27th IEEE Conference on Computer Communications (INFOCOM)*, pp. 66-70, Apr. 2008.

[13] L. E. Li, R. Alimi, R. Ramjee, J. Shi, Y. Sun, H. Viswanathan, and Y. R. Yang, "Superposition coding for wireless mesh networks ," in *Proceedings of the 13th ACM International Conference on Mobile Computing and Networking (MobiCom '07)*, pp. 330-333, Sept. 2007.

[14] R. K. Ganti, Z. Gong, M. Haenggi, C. Lee, S. Srinivasa, D. Tisza, S. Vanka and P. Vizi, "Superposition Coding on a Software-Defined Radio: Design, Implementation and Performance Evaluation", *in preparation*. Available online at http:www.nd.edu/~mhaenggi/pubs/tcom10.pdf.

[15] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*, Sams, 2001.

[16] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, pp. 1613-1621, 1997.

[17] K. Mandke, R. C. Daniels, S. M. Nettles, and R. W. Heath, Jr., "On the challenges of building a multi-antenna software defined packet radio," *Proceedings of the SDR 08 Technical Conference and Product Exposition*, Washington, D.C., Oct. 2008.