Channel Sounding for the Masses: Low Complexity GNU 802.11b Channel Impulse Response Estimation

Dustin Maas, Mohammad H. Firooz, Junxing Zhang, Neal Patwari, and Sneha K. Kasera

Abstract

New techniques in cross-layer wireless networks are building demand for *ubiquitous channel sounding*, that is, the capability to measure channel impulse response (CIR) with any standard wireless network and node. Towards that goal, we present a software-defined IEEE 802.11b receiver and CIR measurement system with little additional computational complexity compared to 802.11b reception alone. The system implementation, using the universal software radio peripheral (USRP) and GNU Radio, is described and compared to previous work. We validate the CIR measurement system and present the results of a measurement campaign which measures millions of CIRs between WiFi access points and a mobile receiver in urban and suburban areas.

I. Introduction

Channel impulse response (CIR) measurements have long held importance for communication system design [1], [2], [3]. The CIR describes the spreading, or echoing, that occurs when an impulse is sent through a channel. This spreading in time can lead to inter-symbol interference (ISI), and frequency-selective or narrow band fading, depending on the symbol bandwidth. A knowledge of the CIR characteristics enables system designers to ensure that ISI does not dominate and hence lead to an excessive irreducible bit error rate [4]. Multipath channels can also be used to increase the bit rate and reliability of multiple-input multiple-output (MIMO) communications systems. Accurate MIMO channel models can be built from CIR measurements [5], and can be used to improve MIMO system design [6]. In general, measurements of CIR in wireless networks have become increasingly important to determine the real-world performance of many new technologies.

In addition, new cross-layer wireless networking technologies use measurements of the multipath channel for purposes of environmental awareness and security, such as fingerprint-based localization [7], RF-based multistatic

D. Maas and N. Patwari are with the Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, USA. M.H. Firooz is with the Dept. of Electrical Engineering, University of Washington, Seattle, USA. S.K. Kasera is with the School of Computing, University of Utah, Salt Lake City, USA. J. Zhang is with the Inner Mongolia University College of Computer Science. This material is based upon work supported by the National Science Foundation under Grant Nos. #0855261 and #0831490, and a University of Utah Research Foundation TCP Grant. Correspondence email: npatwari@ece.utah.edu.

radar [8], location distinction [9], secret key establishment [10]. These applications require CIR measurements to be performed in real time using commercial wireless devices, as opposed to with specialized measurement equipment or in post-processing. Typical commercial wireless devices use the received signal in a demodulator to estimate the transmitted bits, but then discard the received signal samples. Information about the channel (besides the received signal strength) is not forwarded to higher networking layers, nor can it be estimated from the demodulated bits. For the mentioned applications to be viable, future commercial wireless devices must be able to rapidly calculate CIR information.

In this paper, we present the design of an inexpensive CIR measurement system. It is built upon GNU Radio, an open source framework for software-defined radio [11], [12], and the universal software radio peripheral (USRP), an open-source transceiver platform [13]. Compared to signal analyzers and oscilloscopes (a 3-GHz vector network analyzer (VNA) can cost US $20,000), our system is low cost. The cost of the proposed system is US$975 [14], which enables large-scale deployment as might be seen in a typical WiFi deployment. Our system works seamlessly with standard PHY layer signals from commercial 802.11b wireless devices. Essentially, our system provides an 802.11b receiver with the additional capability of CIR estimation.

However, this paper is not limited in scope to the USRP – the implementation presented enables practical CIR estimation in hardware with strict computational limitations, such as FPGAs or application-specific integrated circuits (ASICs). It will not be feasible to compute a CIR estimate with commercial 802.11 hardware unless the computational complexity is low. The open-source platform chosen is an advantage, we believe, because it is likely to lead to cooperative improvement in the system capabilities and large-scale adoption. Providing a system implementation that works within the limitations of the hardware platform is, in part, a demonstration of the feasibility of the approach in future commercial systems.

The CIR measurement system we present in this paper for 802.11b is similar to sliding correlator channel sounding method in which a known pseudo noise (PN) signal is generated and continuously transmitted from a transmitter to the receiver [1], [15]. However, our work is different from the existing method in the following four significant ways:

1) The PN sequence in 802.11b is fixed and not designed for high dynamic range CIR estimation.
2) Devices transmitting in 802.11b send PN-coded symbols modulated with data; modulation is undesirable from the perspective of CIR estimation.
3) Unlike sliding correlator measurement systems, which calculate the full cross-correlation signal after thousands of PN signal periods, our system calculates the full cross-correlation signal during each PN signal period [15].
4) No specialized transmitter is required, as any standard 802.11b transmitter (*e.g.*, laptop or access point) may be used.

Note that IEEE 802.11b devices must support two mandatory bit rates (1 Mbps and 2 Mbps) and may optionally support two higher rates (5.5 Mbps and 11 Mbps) as specified in [16]. In this work, for simplicity, we only consider the standard rates. We note that the start of any 802.11b packet and some 802.11g packets (the first 192 symbols,

known as the PLCP[1] frame), are sent at either the 1 or 2 Mbps rate. Thus many packet sources exist which our system implementation can utilize for CIR estimation.

In a similar, but independent project, a channel sounder for 802.11b applications is reported by Jemai and Kürner in [17], which begins by recording the samples of a 192-bit segment of the 802.11b signal onto a PC. Then, the signal is despread and demodulated on the PC. Next, the transmitted signal for the 192-bit segment is recreated using the demodulated bits. Finally, the recorded received signal and recreated transmitted signal are convolved. Since both have many samples, the cross-correlation consumes significant PC computation time, on the order of $NB\log(NB)$, where N is the number of samples per bit, and B is the number of bits used. In comparison, our system involves PC computation on the order of NB. The system proposed in [17] uses proprietary software and VHDL implementations (ComBlock products from Mobile Satellite Services Inc.), while our implementation uses open-source hardware and software with a wide user base that utilizes and contributes to the code library. As an open source platform, our code has been downloaded from our website 1140 times since its first posting.

Our specific contributions to 802.11b CIR estimation system research are summarized as follows:

1) We provide an implementation of an 802.11b FPGA matched filtering method, the first, to our knowledge, to be presented for the USRP-based GNU radio framework.
2) We provide a method to estimate the CIR from a modulated 802.11b signal. In particular, we use the output of the receiver's matched filter, which allows a lower-complexity CIR estimate compared to [17].
3) We perform extensive measurements, in both lab-controlled and real-world multipath channels.

II. Analysis Methods

In this section, we present a detailed analytical framework for CIR estimation using received 802.11b signals. We describe an 802.11b signal, how it is impacted by a multipath channel, and how the proposed system estimates both: (1) the transmitted data, and (2) the amplitudes and delays of the multipath in the channel. This signal framework is used throughout this paper.

A. Transmitted Signal

The 802.11b physical layer uses direct-sequence spread spectrum (DSSS) modulation with symbol duration of $T_s = 1\mu$s. This transmitted symbol stream is multiplied by a pseudo-noise (PN) code signal, which also has duration T_s. Denoting the PN code signal as $c(t)$ and the jth transmitted data symbol as b_j, the transmitted signal in baseband is given by

$$s(t) \quad = \quad \sum_j b_j c(t - jT_s). \tag{1}$$

Note that b_j generally takes complex values, because data symbols may be modulated either using differential binary phase-shift keying (DBPSK) or differential quadrature phase-shift keying (DQPSK). Although our work is developed and tested for DBPSK, it is readily extendible to DQPSK.

[1]Physical Layer Convergence Procedure

The PN code in 802.11b is called the Barker code. This code consists of eleven *chips*, each with duration $T_c = T_s/11$ μs, thus "spreading" the bandwidth of the transmitted signal to eleven times the original bandwidth. The Barker code signal is a modulated sequence of pulses,

$$c(t) \quad = \quad \sum_{i=0}^{10} c_i p(t - iT_c), \tag{2}$$

where $p(t)$ is the pulse shape, and $c_i \in \{+1, -1\}$ as given in [16]. The pulse shape is chosen to meet the bandwidth limitations imposed by the 802.11b standard, but the precise shape of $p(t)$ is left to the designer. In this paper, when it is necessary to use a particular pulse shape, we have chosen to use a square root raised cosine (SRRC) pulse with roll-off factor $\alpha = 0.5$, which meets the spectral mask requirements and represents a good trade-off between temporal and frequency domain characteristics [18].

B. Received Signal

Because of the multipath radio channel, many copies of the transmitted signal arrive at the receiver with different time delay, amplitude, and phase. The multipath channel filter is modeled as [19]:

$$h(t) = \sum_{l=0}^{L-1} \alpha_l \delta(t - \tau_l), \tag{3}$$

where L is the total number of multipath components, $\alpha_l = |\alpha_l| e^{j\angle\alpha_l}$ is the complex amplitude gain of the lth multipath, τ_l is the delay of the lth multipath, and $\delta(\cdot)$ is the Dirac delta function. Since we are only interested in the relative time delay of each multipath, we let $\tau_0 = 0$, and then τ_l is the additional delay compared to the first arriving multipath.

The received signal $r(t)$ is the convolution of the transmitted signal and the channel filter. Applying (3) and (1),

$$r(t) = s(t) \star h(t) = \sum_{l=0}^{L-1} \sum_{j} \alpha_l b_j c(t - \tau_l - jT_s). \tag{4}$$

An 802.11b receiver "de-spreads" the signal, *i.e.*, performs matched filtering with the PN code signal $c(t)$ from (2), which results in signal $q(t)$,

$$q(t) = r(t) \star c(-t) = \sum_{l=0}^{L-1} \alpha_l \sum_{j} b_j R_c(t - \tau_l - jT_s), \tag{5}$$

where $R_c(t) = c(t) \star c(-t)$ and $R_c(0)$ is the energy in the signal $c(t)$, which we denote $\mathcal{E}_c$.

Standard 802.11b receivers must perform despreading, *i.e.*, the calculation of $q(t)$, in order to perform demodulation. We propose that $q(t)$ can be used directly in CIR estimation as well. By using an output that existing 802.11b receivers compute, we make it more feasible for future 802.11b receivers to estimate CIR without significant additional computational complexity.

We note that it is possible to estimate CIR from all symbols comprising the PLCP preamble and header. If the PLCP is known *a priori*, the reception range can be significantly increased by correlating with the entire PLCP, rather than $c(t)$. In this case, the "energy per bit" is essentially increased by a factor of 48, a 17 dB increase. In

this work, we present a CIR measurement system that works with any 802.11b transmitter, thus we cannot know the PLCP ahead of time. Further, correlating with the entire PLCP adds computational complexity.

The above formulation has not included interference. Inevitably, some packets will be unable to be received due to low SINR, and thus the CIR will not be estimated. Further, the SINR can be estimated from a received packet, and CIR estimates can be dropped if the desired SINR is not achieved.

C. CIR Estimation

The estimation of CIR from a received 802.11b signal is complicated by the modulated data $\{b_j\}$. That is, the PN code signal is modulated with data, presumably unknown to the receiver until after demodulation. For example, for BPSK, $b_j \in \{-1, +1\}$. In this section we first present the (unrealistic) case of an unmodulated signal, *i.e.*, where $b_j = 1$ for all j. We then describe how we estimate the CIR from a modulated 802.11b signal.

First, for an ideal unmodulated signal, (5) would simplify to

$$q(t) = \sum_{l=0}^{L-1} \alpha_l R_{pn}(t - \tau_l), \quad \text{where } R_{pn}(t) = \sum_j R_c(t - jT_s). \tag{6}$$

Here, R_{pn} is the correlation of a PN code signal with a repeating PN code signal with period T_s. The Barker code has the property that this correlation function $R_{pn}(t)$ peaks at $t = 0$ and integer multiples of T_s and is almost constant in between those peaks [20]. Figure 1(a) shows the signal $q(t)$ when there is exactly $L = 1$ path with amplitude $\alpha_0 = 1$. As multipath components correspond to time-delayed versions of $q(t)$, the almost constant correlation in between peaks makes it possible to identify multipath contributions even when their magnitude $|\alpha_l|$ is small.

When dealing with modulated signals, the correlation $q(t)$ may not be nearly constant between peaks, making low-amplitude multipath components harder to identify. In Figure 1(b), we use the transmitted symbols $\mathbf{b} = [1, 1, -1, 1, 1]$, and plot the correlation output signal $q(t)$ from (5), for the case that $L = 1$ and $\alpha_0 = 1$. Note that the normalized amplitude of $q(t)$ between the 2nd and 3rd peaks, and between the 3rd and 4th peaks, rapidly change between $\pm 1/11$. These periods of varying correlation correspond to the times in between changes in symbol values b_j. When $b_j \neq b_{j+1}$, the value of $q(t)$ for $jT_s < t < (j+1)T_s$ is not almost constant.

However, note that when $b_j = b_{j+1}$, there is a nearly constant $-1/11$ correlation value in between the two peaks at jT_s and $(j+1)T_s$. *When subsequent symbols are identical, the almost constant correlation value in $q(t)$ can be exploited for improved CIR estimation*. To avoid the negative impact of symbol modulation, we use the correlator output signal $q(t)$ whenever the symbol value b_j repeats.

To be explicit, define two correlation functions, $R_o(t)$ and $R_s(t)$ (shown in Figure 2), as:

$$\begin{aligned} R_o(t) &= (R_c(t) - R_c(t - T_s)) I_{(0,T_s)} \\ R_s(t) &= (R_c(t) + R_c(t - T_s)) I_{(0,T_s)} \end{aligned} \tag{7}$$

where $I_{(0,T_s)}(t)$ has value one at interval $(0, T_s)$ and zero elsewhere. We also define two subsets, $J_s = \{j : b_j = b_{j+1}\}$ for symbol integers j when the next symbol value repeats, and $J_o = \{j : b_j \neq b_{j+1}\}$. Then we can write

(5) as

$$q(t) = \sum_{j \in J_s} b_j \sum_{l=0}^{L-1} \alpha_l R_s(t - jT_s - \tau_l) + \sum_{j \in J_o} b_j \sum_{l=0}^{L-1} \alpha_l R_o(t - jT_s - \tau_l). \quad (8)$$

This version of $q(t)$ contains terms $R_s(\cdot)$ and $R_o(\cdot)$ that have support only over one symbol period. We estimate the CIR by averaging only the symbol periods of $q(t)$ that correspond to repeated symbol values:

$$\hat{h}(t) = \frac{1}{|J_s|} \sum_{j \in J_s} b_j q(t - jT_s) I_{(0,T_s)}(t) \approx \sum_{l=0}^{L-1} \alpha_l R_s(t - \tau_l) \quad (9)$$

Essentially, the channel estimator in (9) averages together only the impulse responses estimated during periods when the symbol value is not switching and thus the correlation function is nearly constant. Note that symbol values b_j do not affect $\hat{h}(t)$. In the ideal case, the channel estimate is a sum of time-delayed, attenuated, and phase-shifted versions of $R_s(t)$. However, in a given hardware implementation, $R_s(t)$ may be affected by other filters, known or unknown, in the receiver chain. If the overall filter of the receiver chain is unknown, it may be beneficial to estimate $R_s(t)$ using a known channel, *i.e.*, an interference-free cabled connection between the transmitter and receiver. We employ this method to generate an estimate of $R_s(t)$ from a single packet, which we call $\hat{R}_s(t)$.

The CIR estimate $\hat{h}(t)$ in (9) is a convolution of the true CIR in (3) with $\hat{R}_s(t)$, which has a zero-to-zero pulse width of approximately 188 ns. Since multipath arrive more closely spaced than 188 ns, the complex-valued, time-delayed pulse shapes $\hat{R}_s(t - \tau_l)$ overlap in time, making it difficult to visually inspect $\hat{h}(t)$ to identify multipath arrival delays.

We apply a deconvolution procedure based on [21] to estimate multipath time delays. This procedure is described in detail in [22]. In short, we discretize the CIR and write the measurement as a linear combination of the CIR amplitudes. Then, we solve a quadratic optimization problem using the well known convex optimization software [23] to perform the inversion.

The sampled measurement is written as,

$$\hat{h}[n] \quad = \quad \sum_{l=0}^{L-1} \alpha_l \hat{R}_s(n\, T_s - \tau_l) + w[n] \quad (10)$$

where $w[n]$ is measurement noise, assumed to be i.i.d. Gaussian. Equation (10) can be written as $\hat{\mathbf{h}} = \hat{\mathbf{R}}_s \boldsymbol{\alpha} + \mathbf{w}$, where $[\hat{\mathbf{R}}_s]_{k,l} = \hat{R}_s(kT_s - \tau_l)$ is an $M \times L$ matrix, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_L]^T$, and $\mathbf{w}$ is an M-dimensional noise vector. An estimate of $\boldsymbol{\alpha}$ is the solution to the following optimization problem [22], [21],

$$\hat{\boldsymbol{\alpha}} = \min_{\boldsymbol{\alpha}} \|\hat{\mathbf{h}} - \hat{\mathbf{R}}_\mathbf{s} \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \quad (11)$$

where λ is a fixed parameter, the inverse of a Lagrange multiplier [24], which is set as discussed in [21].

III. Implementation

In this section, we present the system implementation of the WiFi-based receiver and CIR estimator using the proposed FPGA matched filtering method on a USRP. The USRP receiver path consists of a 64 MS/s (million samples per second) 12-bit ADC, an Altera Cyclone FPGA, and a USB controller. The USB 2.0 bandwidth is not

sufficient to stream an 802.11b signal sampled at the Nyquist rate. However, the rate limitations of a USB 2.0 link do not limit transfer of 802.11b *symbol information*, since symbols are sent at 1 Msymbols/s [25]. The spreading via DSSS adds no information, but causes the RF bandwidth to increase by a factor of 11. To reduce the received signal to samples at 1 MS/s, we must first despread on the FPGA. After despreading, symbol decisions can be made using only one sample per symbol, and, as we will show, a subset of samples per symbol can be used for CIR estimation.

A broad overview of the signal processing steps is shown in Figure 3. We first reduce the sampled data $r(t)$ to 32 MS/s. Then, we despread as described in (5). The output $q(t)$ has a sample rate of 32 MS/s, however, not every sample is necessary, so we send only samples near the peaks in $q(t)$, as described in detail in this section.

The result is that the average data rate sent to the PC is within the rate limitations of a USB 2.0 link. The PC then performs the symbol detection and bit decoding operations as specified in the IEEE 802.11b standard. Our receiver implementation can consistently receive 802.11b packets sent at the 2 Mbps rate, and the reception range is up to 20 meters.

We compare our implementation to previous work [26], which we call the *bandwidth reduction method*. In this method, $r(t)$ is filtered and downsampled to a 8 MHz RF bandwidth, smaller than the RF bandwidth of the DSSS signal. Then, the samples are at a rate low enough to be transfered over USB to be processed on a PC. The downsampling reduces the range of the receiver, as we show in Section IV.

The main computational challenge in the proposed method is the implementation of matched filtering on the FPGA. We propose a computationally-efficient method to implement the 802.11b matched filter, valid for the strict limitations of the given FPGA, or any computationally limited ASIC or FPGA implementation. We describe three ways in which the implementation reduces computational complexity and data rate yet still provides a high-capability system implementation: multiplication reduction; use of two memories; and peak selection.

1) Multiplication Reduction: A direct implementation of the matched filter in (5) would require 32 multiplies and additions per sample. We reduce the complexity as follows. Figure 4 shows $c(t)$ and its sampled version, $c(iT_s)$. We quantize each sample of the PN code signal, $c(iT_s)$, and denote the quantized values as $c_q(iT_s)$. In our implementation, we chose quantization to five bits, in a trade-off between resolution and multiplier space complexity.

Some values of $|c(iT_s)|$ are similar enough, that when quantized to five bits, $|c_q(iT_s)| = |c_q(jT_s)|$ for some $j \neq i$. Since summation is simpler than multiplication in an FPGA, it saves both time and complexity to first add (or subtract) samples with identical $|c_q|$ value, and then multiply the sum by its $|c_q|$ value.

$$q(nT_s) = \sum_{i=0}^{31} c_q(iT_s) r((n-i)T_s) = \sum_{g=1}^{15} c_g \left[\sum_{i \in S_g} \text{sgn}\left\{c_q(iT_s)\right\} r((n-i)T_s) \right] \tag{12}$$

where $q(nT_s)$ is the nth sample of the match filter output $q(t)$, S_g is the set of all indexes in the gth group, $\text{sgn}\{\cdot\}$ is the signum function, and c_g is the multiplicative factor $c_q(iT_s)$ for all $i \in S_g$. The S_g and c_g for each group g are listed in Table I. Using this rearrangement, we require 15 multiplications, instead of the 32 that would be required in a direct implementation.

2) Two Memories: An FPGA requires parallelization in order to complete the several additions and multiplications required at each new sampling time. Our implementation allows two clock cycles (clock rate of 64 MHz) per sampling time (sampling rate of 32 MS/s). During these two clock cycles, we must perform addition and multiplication as described above, and shift samples to allow space for the new incoming signal sample.

For this purpose, we use two 32-length arrays, which we refer to as mem and bmem. When a new sample is received, it is located at mem[0] while mem[1] to mem[31] are filled by bmem[0] to bmem[30]. In the next cycle, mem[0] to mem[31] are put in bmem[0] to bmem[31]. This process is depicted in Figure 5. As explained in previous paragraph, we first add the data in bmem, by group g, which is completed in one cycle. Then, multiplication by group multiplier c_g is performed, and the results summed.

3) Peak Selection: The output of the FIR filter, $q(nT_s)$, has a 32 MS/s rate. With a sampling period of 31.25 ns, we capture 344 ns (much more than the typical excess delay for short-range channels [27]) of the signal within 11 samples. These 11 samples per symbol can be reliably transfered via USB.

The peak selection algorithm selects 11 out of each 32 samples per symbol as follows. First, the FPGA computes the power values $|q(nT_s)|^2$, $n = 1, \ldots, 32$. The index of samples with maximum power is denoted $n_{max} = \text{argmax}_n |q(nT_s)|^2$. The FPGA sends through the USB the samples from three samples before to seven samples after the peak power sample, *i.e.*, $\{q((n_{max}+i)T_s)\}_{i=-3}^{7}$.

Note that the proposed CIR measurement system finds samples near the maximum-power peak, not necessarily the line-of-sight (LOS) path. In a non-LOS dominant channel, if the LOS path arrives within three samples prior to the maximum-power peak, the proposed system records the full CIR.

IV. Experimental Results

In all cases, we load an Ettus Research USRP (rev 4.5) with the code described in Section II-C. The RF front end is a RFX2400 daughterboard (rev 30), also from Ettus Research. The antenna is a 2400-2480 MHz sleeve dipole antenna with omnidirectional pattern in the horizontal plane and a 3 dBi gain. The USRP is connected to a Dell Inspiron laptop running Python and Matlab. The Python (GNU radio) code collects data from the USB, demodulates the packet data, and writes to a file. The Matlab code performs the averaging required in (9) and then displays and stores the impulse response estimate $\hat{h}(t)$. From the stored $\hat{h}(t)$, the deconvolution described in Section II-C is performed in post-processing.

A. Demodulator

We do not proceed with CIR estimation when packet data does not pass the CRC test. Equally important, the MAC address of a transmitter is included in the packet header, and this is necessary to distinguish packets originating from different transmitters. In this section, we measure the packet reception rate (PRR) of the implemented 802.11b CIR measurement system in an interference-free environment.

We configure a test transmitter, a D-Link 802.11b wireless router (model DI-614+), to broadcast a beacon packet at a basic rate (1 or 2 Mbps) every 200 ms (5 packets/sec). The router and receiver are placed in a shielded anechoic

chamber and separated by 6.0 meters. The packet reception rate is recorded for three minutes, and experiment repeated four times. The implementation presented in this paper receives an average of 724 packets, while the bandwidth reduction method receives an average of 454. The results show that the FPGA matched filtering method outperforms the bandwidth reduction method by successfully demodulating 1.6 times more packets.

B. Channel Measurement

In this section we first perform two experimental validations on our implementation using known channels between the transmitter and the receiver. Then, we perform an experimental measurement campaign to measure a large number of CIRs in outdoor areas in and around Salt Lake City, Utah. We provide measurement results and summarize the measured delay characteristics.

1) Validation: To validate the CIR estimation system, we create two channels with known impulse response out of RF hardware and cable, with which we connect the wireless router (transmitter) and receiver.

Single-path: The transmitter is connected to an attenuator, whose output is connected via cable to the receiver. We record several measured CIR estimates $\hat{h}_1(t)$. Figure 6-(a) shows three measurements $\hat{h}_1(t)$ and the estimated CIR for a single-path channel, $\hat{R}_s(t)$. Since $\hat{h}_1(t)$ is nearly identical to $\hat{R}_s(t)$, it is apparent that the channel has only one path, *i.e.* $L = 1$. Figure 6-(b) presents the deconvolved CIR estimate from (11).

Double-path: In the next experiment, the transmitter cable is connected to a RF splitter with two outputs, one connected to a short cable (1.5m), and another to a long cable (25.9m). We first measure the CIR using a vector network analyzer, from which we find that the difference in delay between the two paths is 122 ns. The amplitude difference between the two paths is measured to be 9.5 dB by using a LadyBug power sensor (LB479A). Figure 6-(a) shows the CIR measurements for $\hat{h}_2(t)$. As can be seen, $\hat{h}_2(t)$ is consistently higher in amplitude than $\hat{R}_s(t)$ between the samples 6 through 10, indicative of later-arriving multipath power. The deconvolution algorithm of (11) is applied and the resulting estimate in plotted in Figure 6-(c).

The results clearly show two paths, the later paths with a 125 ns relative delay and between 10 and 12 dB less received power. In the results where the estimated power of the second path is above -20 dB relative to the path with maximum power (the same noise level we use for the calculation of the dispersion statistics), we find we are able to achieve a standard deviation of 0.69 radians for the difference in phase between multipath components corresponding to the first and second paths. Additionally, the standard deviation of the relative power of the two multipath components is 3.5 dB. These statistics are good considering the hardware synchronization issues, phase noise, and the coarse sampling period for the CIR estimates.

Observation of Figures 6(b) and 6(c), as well as many other deconvolution results lead us to the conclusion that the dynamic range for the CIR measurement system is at least 20 dB, which is expected since the PN coding gain of the Barker code is $20 \log_{10} 11 \approx 20.8$ dB.

C. Drive-Test CIR Measurement Campaign

We use our system to measure CIRs in three residential, two commercial, and one downtown area in Salt Lake City. The residential areas are comprised of one to three story single-family homes and apartment buildings. The commercial areas include streets near strip malls, low-rise office buildings, and heavy vehicle traffic. The downtown area is an urban canyon of high-rise office buildings on both sides of the streets. In each area, the receiver antenna is on the outside of a vehicle that drives at typical speeds on city streets. In the course of six five-minute drive-test measurements, a total of three million CIR measurements are recorded. Figure 7 presents a typical deconvolved CIR estimates $\hat{\boldsymbol{\alpha}}$ from each area.

In order to compare different multipath channels and to develop some general design guidelines for wireless systems, parameters that grossly quantify the multipath channel are used. The time dispersive properties of wide band multipath channels are most commonly quantified by their mean excess delay $\bar{\tau}$ and RMS delay spread σ_τ, as defined in [27]. Table II presents the average mean excess delay, average RMS delay spread, and maximum RMS delay spread of the measured channel responses for each area.

Delay spreads depend strongly on path length, antenna height, frequency, and environment. However, previous work has shown that, in general, rural and suburban delay spreads are smaller than in urban or dense urban areas [28], [29], [30], [31]. Our results in Table II show a similar trend, since the residential and commercial areas can be considered suburban and have lower average RMS delay spreads than the urban downtown area. One of the few studies of RMS delay spreads for indoor-to-outdoor channels near 2.4 GHz reported average RMS delay spreads of 27-44 ns [32], but the studied path lengths were about 330 meters, significantly longer than one would expect from 802.11b path lengths.

V. Conclusion

Future wireless networks are envisioned that rely on the real-time estimation of CIR from received WiFi packets for the purposes of cross-layer security, localization, and environmental imaging. We present a CIR estimation system using an inexpensive and open source hardware and software platform to enable these emerging areas of research. We show how accurate CIR estimation can be performed using a resource-constrained FPGA, which provides a proof-of-concept for future commercial devices.

Future work should address MIMO (*e.g.*, 802.11n) CIR measurement using a bank of synchronized software radios. Low complexity MIMO CIR measurement will likely benefit the development of future cross-layer techniques in multiple-antenna wireless networks.

References

[1] D. C. Cox, “Delay Doppler characteristics of multipath propagation at 910 MHz in a suburban mobile radio environment,” *IEEE Trans. Antenna and Propagation*, vol. 20, no. 5, pp. 625–635, Sept. 1972.

[2] G. L. Turin, F. D. Clapp, T. L. Johnston, S. B. Fine, and D. Lavry, “A statistical model of urban multipath propagation,” *IEEE Trans. Vehicular Technology*, vol. 21, no. 1, pp. 1–9, Feb. 1972.

[3] D. M. Devasirvatham, "Time delay spread and signal level measurements of 850 MHz radio waves in building environments," *IEEE Trans. Antenna and Propagation*, vol. 34, no. 11, pp. 1300–1305, Nov. 1986.

[4] A. Kemp and S. Barton, "The impact of delay spread on irreducible errors for wideband channels on industrial sites," *Wireless Personal Communications*, vol. 34, no. 3, pp. 307–319, 2005.

[5] H. Xu, D. Chizhik, H. Huang, and R. Valenzuela, "A wave-based wideband MIMO channel modeling technique," in *Proc. 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 4, 2002.

[6] J.-M. Molina-Garcia-Pardo, J.-V. Rodriguez, and L. Juan-Llacer, "MIMO channel sounder based on two network analyzers," *IEEE Trans. Instrumentation and Measurments*, vol. 57, no. 9, Sept. 2008.

[7] D. Lee, K. Sowerby, and M. Neve, "Extracting fine multipath detail from measured data at 5.8 GHz," in *Proc. 59th IEEE Vehicular Technology Conference*, vol. 1, 2004, pp. 74–78.

[8] M. C. Wicks, B. Himed, J. Bracken, H. Bascom, and J. Clancy, "Ultra narrow band adaptive tomographic radar," in *Proc. 1st IEEE Int. Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, Dec. 2005, pp. 36–39.

[9] J. Zhang, M. H. Firooz, N. Patwari, and S. K. Kasera, "Advancing wireless link signatures for location distinction," in *Proc. 14th Int. Conf. Mobile Computing and Networking (MobiCom'08)*, Sept. 2008, pp. 26–37.

[10] B. Azimi-Sadjadi, A. Kiayias, A. Mercado, and B. Yener, "Robust key generation from signal envelopes in wireless networks," in *Proc. 14th ACM Conf. Computer and Communications Security*, Nov. 2007, pp. 401–410.

[11] E. Blossom, "GNU radio: Tools for exploring the radio frequency spectrum," *Linux Journal*, no. 112, p. 4, Jun. 2004.

[12] G. D. Troxel, E. Blossom, S. Boswell, A. Caro, I. Castineyra, A. Colvin, T. Dreier, J. Evans, N. Goffee, K. Haigh, and et al., "Adaptive dynamic radio open-source intelligent team (ADROIT): Cognitively-controlled collaboration among SDR nodes," in *Proc. IEEE Workshop on Networking Technologies for Software Defined Radio Networks*, Sept. 2006, pp. 8–17.

[13] Ettus Research LLC. [Online]. Available: http://www.ettus.com/

[14] M. Firooz, D. Maas, and N. Patwari. [Online]. Available: http://span.ece.utah.edu/pmwiki/pmwiki.php?n=Main.80211bReceiver

[15] R. J. Pirkl and G. D. Durgin, "Optimal sliding correlator channel sounder design," *IEEE Trans. Wireless Communications*, vol. 7, no. 9, pp. 3488–3497, Sept. 2008.

[16] "IEEE Std 802.11b, Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Higher-speed physical layer extension in the 2.4 GHz band," 1999.

[17] J. Jemai and T. Kurner, "Broadband WLAN channel sounder for IEEE 802.11b," *IEEE Trans. Vehicular Technology*, vol. 57, no. 6, pp. 3381–3392, Nov. 2008.

[18] B. Farhang-Boroujerny, *Signal Processing Techniques for Software Radios*. Lulu Press Inc., 2008.

[19] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[20] N. Levanon and E. Mozeson, *Radar Signals*. John Wiley & Sons Inc., 2004.

[21] J. Fuchs, "Multipath time-delay detection and estimation," *IEEE Trans. Signal Processing*, vol. 47, no. 1, pp. 237–243, Jan. 1999.

[22] M. Firooz, D. Maas, J. Zhang, N. Patwari, and S. Kasera, "Channel sounding for the masses: Low complexity gnu 802.11b channel impulse response estimation," *arXiv:1007.3476v1 [cs.CR]*, July 2010.

[23] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 1.21," http://cvxr.com/cvx, Apr. 2011.

[24] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Addison-Wesley Reading, MA, 1973.

[25] IEEE wireless standards zone. Http://standards.ieee.org/wireless/. [Online]. Available: http://standards.ieee.org/wireless/

[26] D. Sumorok. bbn_80211b_rx.py, version 1.7. [Online]: http://acert.ir.bbn.com/viewvc/adroitgrdevel/. [Online]. Available: http://acert.ir.bbn.com/viewvc/adroitgrdevel/adroitgrdevel/gr-bbn/src/examples/

[27] T. S. Rappaport, *Wireless Communications: Princiles and Practice*. Prentice-Hall Inc., New Jersey, 1996.

[28] J. Jorgensen, I. Kostanic, and W. Foose, "Application of channel sounding to CDMA PCS design at 1900 MHz," in *Proc. 47th IEEE Vehicular Technology Conference*, vol. 3, May 1997, pp. 1937–1941.

[29] L. Greenstein, V. Erceg, Y. Yeh, and M. Clark, "A new path-gain/delay-spread propagation model for digital cellular channels," *IEEE Trans. Vehicular Technology*, vol. 46, no. 2, pp. 477–485, May 1997.

[30] X. Zhao, J. Kivinen, P. Vainikainen, and K. Skog, "Propagation characteristics for wideband outdoor mobile communications at 5.3 GHz," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 3, pp. 507–514, Apr. 2002.

[31] E. Sousa, V. Jovanovic, and C. Daigneault, "Delay spread measurements for the digital cellular channel in toronto," *IEEE Trans. Vehicular Technology*, vol. 43, no. 4, pp. 837–847, Nov. 1994.

[32] G. Durgin, V. Kukshya, and T. Rappaport, "Wideband measurements of angle and delay dispersion for outdoor and indoor peer-to-peer radio channels at 1920 MHz," *IEEE Trans. Antennas and Propagation*, vol. 51, no. 5, pp. 936–944, may 2003.

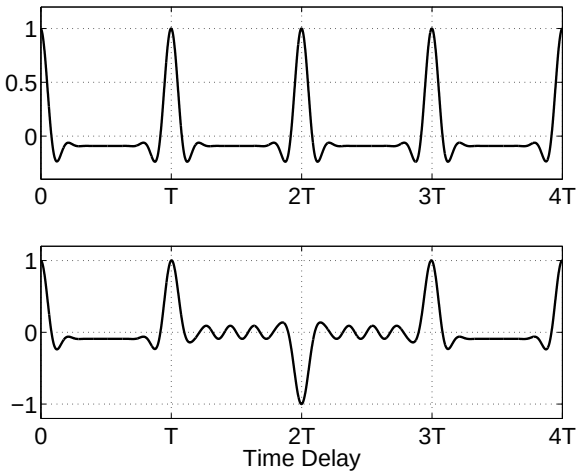

Fig. 1. Correlation output signal $q(t)$ in one-path channel ($L = 1$ and $\alpha_0 = 1$) when (Top) receiving an unmodulated signal (*i.e.*, $\mathbf{b} = [1, 1, 1, 1, 1]$) (Bottom) receiving a signal modulated with $\mathbf{b} = [1, 1, -1, 1, 1]$.

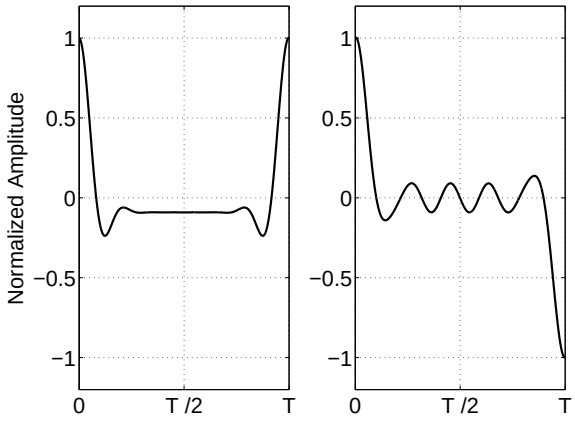

Fig. 2. Normalized symbol-period length correlation functions (Left) $R_s(t)$ and (Right) $R_o(t)$, both given in (8).

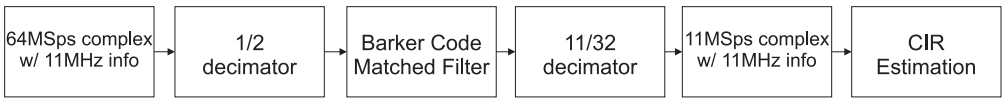

Fig. 3. Block diagram of FPGA matched filtering method.

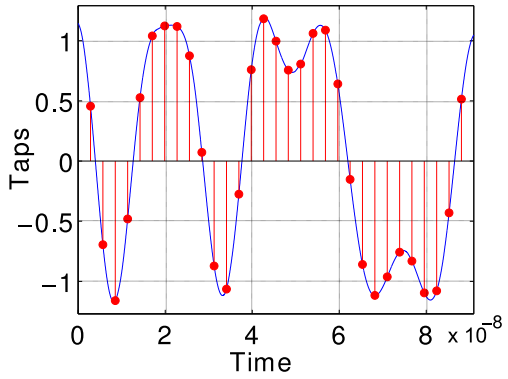

Fig. 4. Samples (•) of PN code signal $c(t)$, or equivalently, taps of the matched filter.

TABLE I

INDICES BY GROUP g AND THE GROUP'S MULTIPLIER VALUE c_g.

g	Multiplier c_g	Index Set S_g
1	19	$\{16, 28\}$
2	18	$\{3, 7, 23, 24, 31\}$
3	17	$\{11, 12, 19, 22, 15\}$
4	16	$\{25\}$
5	15	$\{6\}$
6	14	$\{8, 20, 22\}$
7	13	$\{4, 13\}$
8	12	$\{5, 14, 17\}$
9	11	$\{29\}$
10	10	$\{10\}$
11	8	$\{0, 26, 27\}$
12	7	$\{1, 30\}$
13	4	$\{18\}$
14	2	$\{9\}$
15	1	$\{21\}$

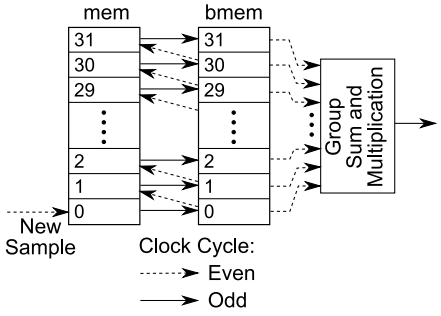

Fig. 5. Memories mem and bmem are used to accept a new sample, and shift data, in two cycles, to allow for summation and multiplication.

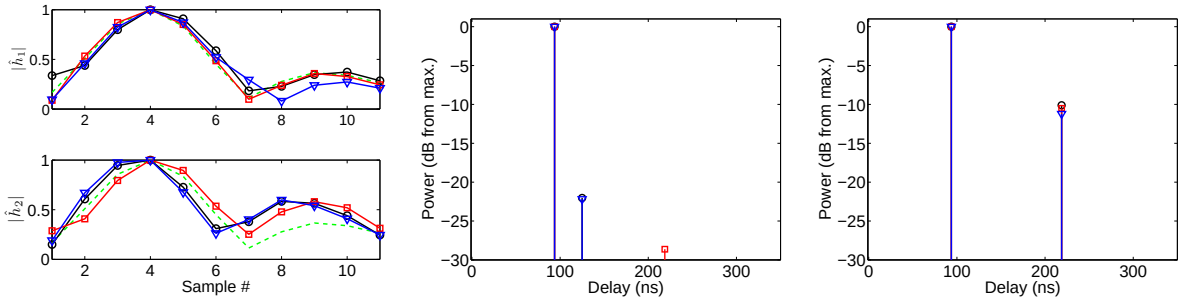

Fig. 6. Single- and double-path results: (a) $|\hat{h}|$ for single-path (Top) and double-path (Bottom), both showing three different measurements (solid marked lines) and empirical CIR $\hat{R}_s[n]$ (dashed line); Corresponding deconvolved $\hat{\boldsymbol{\alpha}}$ for (b) single-path; and (c) double-path.

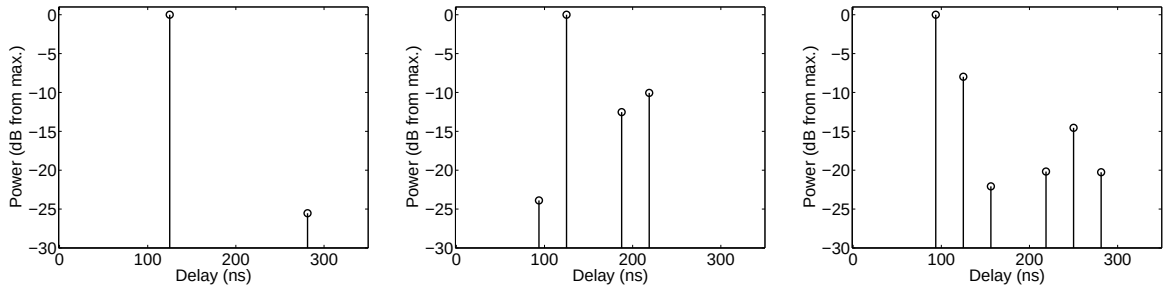

Fig. 7. Typical CIR, $\hat{\boldsymbol{\alpha}}$, measured in (a) residential, (b) commercial, and (c) downtown areas.

TABLE II

RMS DELAY SPREAD AND MEAN EXCESS DELAY STATISTICS FOR RESIDENTIAL (RES.), COMMERCIAL (COM.), AND DOWNTOWN (DT) AREAS.

	Res. 1	Res. 2	Res. 3	Com. 1	Com. 2	DT
Average $\bar{\tau}$ (ns)	7.1	36.7	7.4	6.4	17.7	48.2
Average σ_τ (ns)	7.0	23.7	7.4	6.5	16.9	30.6
Max. σ_τ (ns)	47.4	86.8	35.2	22.8	80.7	88.6