An Intelligent Physical Layer For Cognitive Radio Networks

Aveek Dutta, Jeffrey Fifield, Graham Schelle, Dirk Grunwald, Douglas Sicker
University of Colorado, Boulder
Boulder, CO
{Aveek.Dutta, Jeff.Fifield, Graham.Schelle, Dirk.Grunwald, Douglas.Sicker}@colorado.edu

ABSTRACT

In this paper, we present an intelligent physical layer for cognitive mesh networks. It is well recognized that wireless mesh networks suffer from the inherent property of per hop delay attributed to store and forward routing and channel contention. We show that an intelligent physical layer coupled with efficient traffic engineering and channel allocation mechanism will reduce latency. In this paper, we discuss the evolution of an OFDM receiver, with sufficient software control to aid reconfigurability, capable of receiving and decoding information on different set of subcarriers, and also capable of switching the incoming signals to a different part of the available spectrum on the fly. Equipped with this enhanced receiver we propose a mechanism for wireless wormhole routing, which employs frequency domain switching between subchannels where each subchannel is defined by a set of subcarriers. The OFDM receiver handles three primitives: transmit, receive and relay rather than just transmit or receive. Instead of a contention based, store and forward routing, a relay oriented physical layer has been proposed to reduce latency. The processing pipeline at an intermediate node no longer involves higher layer processing, and the hardware relays the incoming signal on-the-fly to a different part of the spectrum allowing for a full duplex transmission as the transmitter can relay signals while it is receiving on a different subchannel.

1. INTRODUCTION

Today, 802.11a/g and its variants have been adopted as a standard for wireless data communication and are a part of day to day life for many. However, for multi-hop wireless networks using 802.11a/g, the maximum data rates supported by the standard are not achieved in practice because of the substantial routing and processing delay experienced at every node. At each node in a path, a packet must be received, stored, processed and queued before being forwarded to its destination. Contending for the medium and the actual transmission incur additional delays. Researchers in [1] envision an intelligent physical layer to facilitate low latency mobile ad-hoc networks (MANETs). Motivated by the need for hardware that can be used to build these low latency MANETs, a platform has been designed that not only acts as a conventional 802.11a/g transceiver but also intelligently uses spectrum by employing frequency domain switching and adaptive modulation techniques. In addition to relaying signals using conventional store and forward methods, the transceiver has the ability to route packets at the physical layer, reducing the required processing significantly. Instead of traversing up and down the network stack for every relayed packet, the physical layer relay requires only the packet detection, equalization, demodulation, frequency switching, and modulation steps. Although reduced contention does not follow directly from this design, such a goal can be achieved by using this design along with preassigned channels, scheduling and resource allocation mechanisms similar to those found in TDMA networks. This requires an access control mechanism that resides in the physical layer and that is be able to make decisions on path access with little or no intervention from the MAC layer. The proposed hardware should support multiple access techniques like OFDMA to further reduce latency.

Use of orthogonal multi-carrier modulation techniques like OFDM along with a frequency agile transceiver enables every node in the network to make full use of the spectrum. Multi-carrier communication also introduces the idea of subchannels where a predefined number of subcarriers are grouped to form a subchannel. Thus every transceiver can be thought of as a switch or relay that can receive in one subchannel and transmit in another subchannel based on the channel conditions. This kind of frequency agility requires a frequency agile transmitter as well as a frequency agile receiver. A software controlled OFDM transmitter designed by researchers in [2] successfully demonstrated the design and performance of a frequency agile transmitter. The corresponding receiver and relay design is presented in this paper.

In the quest for new research platforms that are robust and that allow real time reconfigurability, field programmable gate array (FPGA) based hardware has proven to be a good choice. Software equivalents and hybrid CPU/FPGA designs have also proved to be beneficial. Fast interfaces between FPGA hardware and conventional CPUs in the form of PCI and PCI Express coupled with the processing power of FPGAs make these types of systems formidable platforms for developing next generation transceivers. As a final step, our transmitter, receiver, and relay, along with a PC host interface have been designed to fit onto a Virtex-IV FPGA

Digital Object Identifier: 10.4108/ICST.WICON2008.4984
http://dx.doi.org/10.4108/ICST.WICON2008.4984

with software control for easy reconfigurability.

The rest of the paper is organized as follows. Section 2 contains related work in the field of OFDM receiver design in reconfigurable platforms, section 3 describes the hardware design aspects for various processing blocks, and in section 4 we present the design modifications necessary to implement wormhole routing. Section 5 deals with the performance of the proposed hardware and presents experimental results. Section 6 presents future directions and challenges in practical implementations of such hardware from the standpoint of protocols and software defined radio architectures. We conclude in section 7.

2. RELATED WORK

We start by citing previous works implementing OFDM transceivers in FPGAs. Most of the instances of hardware implementation employ a hybrid design using the processing power of either the host PC or the embedded processor core in the FPGA. Basic OFDM modem implementation has been around for while. Dick and Harris [3] describe algorithms and designs for packet detection, timing and equalization that can be implemented with relatively low complexity in FPGAs. Speth et al. [4, 5] is one of the earliest to formally lay out the design aspects and algorithms for an OFDM receiver.

Troya et al. [6] describe an ASIC implementation of an OFDM receiver. The WARP project at Rice University [7] is another FPGA based development platform. They have implemented a complete OFDM transmit and receive chain in the FPGA. As in our work, the WARP platform is programmed using Simulink and System Generator. Fifield et al. [2] have already implemented a frequency agile OFDM transmitter that is capable of efficient utilization of the available spectrum by selectively transmitting in some of the subcarriers rather than all subcarriers as done in 802.11a/g [8]. Wouters et al. [9] proposed a novel OFDM wireless modem with adaptive loading and their design is also optimized into an ASIC which supports HYPERLAN and IEEE 802.11a. The KUAR platform from Kansas University [10], is also a complete implementation of OFDM physical layer. They also implement a NC-OFDM where certain subchannels can be selectively and dynamically disabled.

The subchannel allocation mechanism in multicarrier communication using OFDM has been studied for some time now and is a critical part in 802.16a based systems. Zhu et al. [11] describe an adaptive subchannel allocation method in a OFDM communication system. Although channel allocation does not form a part of this work, it is worth mentioning because the receiver proposed here has the capability to adapt to dynamic channel allocation from a control unit, usually a part of the path access control abstraction. As envisioned by Ramanathan [1], a receiver capable of switching waveforms from one subchannel to other on the fly will substantially reduce the latency in existing wireless networks. Simultaneous transmit and receive can be done only if the receiving subchannels and the transmitting subchannels do not interfere. This is particularly difficult in existing wireless networks because the packet detection and synchronization still depends on the physical layer preamble, which is transmitted using all the subcarriers available. One apparent solution to this problem is to use the preambles to periodically synchronize the transmitter and the receiver and use TDMA slots to communicate. Although this is a feasible solution for one hop communication, it introduces a lot of overhead and delay as the number of hops increases. So synchronization and packet detection still forms an integral part of the receiver which needs further work and certainly calls for changes in existing protocols. Therefore, in the future, physical layers will need to evolve to handle three primitives - receive, discard or relay/re-broadcast after switching the subchannels instead of two, either keep it or discard it because currently the routing decisions are not a part of the physical layer.

3. OFDM RECEIVER DESIGN

This chapter presents the design and implementation of an OFDM receiver for a FPGA. It also describes the various algorithms that have been implemented in the hardware to successfully decode 802.11a/g packets.

3.1 System Parameters

For this implementation we use the 802.11a/g [8] physical layer specification with a few changes. While keeping the subcarrier spacing constant, the sampling frequency has been increased 4 times to 80 MHz and the FFT/IFFT size to 256 instead of 20MHz and 64pt FFT/IFFT respectively. This allows us to accommodate a more subcarriers with the same bandwidth, providing greater throughput at the expense of a higher clock frequency for the baseband receiver. For this implementation we chose to use only 64 of the 256 subcarriers and the receiver has been designed around this assumption. Since the design operates at a higher clock frequency, the number of samples processed per unit time is four times greater than in conventional 802.11a/g receivers.

3.2 The Radio Components

The OFDM transceiver components consist of the following: 1) Radio frontend from Fidelity Comtech, Inc - This radio is responsible for up/down conversion to/from the 2.4GHz ISM band. Gain control is also a part of this unit which can be controlled by software on the host computer. 2) Xilinx ExtremeDSP development kit IV manufactured by Nallatech - The ExtremeDSP board includes a Virtex IV equipped with a PCI/USB interface and two sets of A/D and D/A converters.

3.3 Packet Detection and Synchronization

The packet detection of an OFDM receiver utilizes the periodic nature of the short preamble. Every 802.11a/g packet has a short preamble of 10 symbols each 64 samples long. Schmidl and Cox [12] explain a packet detection algorithm which computes a ratio of the autocorrelation energy between the n^{th} and $(n+64)^{th}$ samples and the raw signal energy in that interval.

The decision metric is obtained as the ratio between the two by using a threshold test. Both of these correlations can be implemented using a recursive summing unit or a single stage CIC (Cascaded Integrator Comb) Filter expressed by the iterative eq. (1).

$$y(n) = y(n) + x(n) - x(n-D) \tag{1}$$

Fig. 1 shows the output of the packet detector. The plateau shape of the output shows the autocorrelation energy which starts to build up and remains constant after 2 symbol periods. The third trace shows the raw signal energy

Digital Object Identifier: 10.4108/ICST.WICON2008.4984
http://dx.doi.org/10.4108/ICST.WICON2008.4984

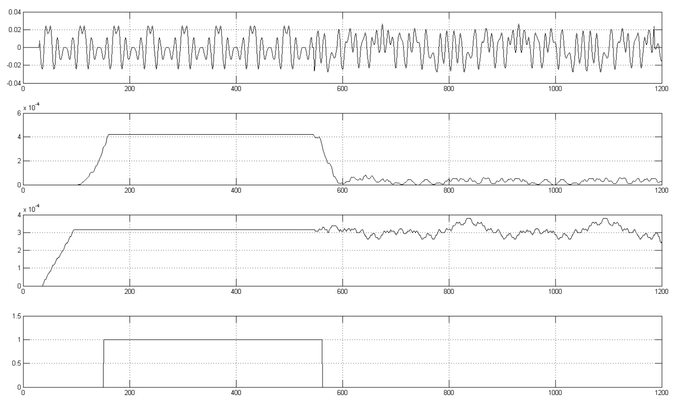


Figure 1: Packet Detector Output

during the autocorrelation phase which also goes constant for the short preamble duration.In the presence of noise, the plateaus will not be flat but will have inclination with jagged edges. The scaling factor for the signal energy needs to be dynamically adjusted to adapt to the changing signal energy or else this might lead to false detection or no detection at all.

3.4 Carrier Frequency Offset Correction

Since the transmitter and the receiver clocks are powered by different crystal oscillators, there will always be an offset between the two. At the receiver, during down-conversion, the oscillator will try to tune to the desired center frequency. Due to oscillator drifts, let there be an offset of δf. Therefore, after down-conversion we get,

$$r_n = s_n e^{-j2\pi\delta f n T_s} \quad \text{where} \quad \delta f = f_{rx} \sim f_{tx} \tag{2}$$

Since the preambles are periodic with periodicity D = 64 sample for short preambles and D = 256, we can use this property to extract the frequency offset δf as follows,

$$Z = \sum_{n=1}^{L-1} r_n . r_{n+D}^{*} = e^{-j2\pi\delta f D T_s} \sum_{n=1}^{L-1} |s_n|^2 \tag{3}$$

L and D are the integration length and periodicity which are different for short and long preambles. The longer the integration time the better the frequency estimate. Therefore for short preamble, L = 128 and D = 64, where as for long preamble L = 256 and D = 256. The coarse estimate is updated twice during the short preamble and then the estimate is made finer twice at the end of the two long preambles. The frequency offset can directly be computed using the autocorrelation energy in eq. (3) as follows,

$$\delta f = \frac{1}{2\pi D T_s} \arctan(Z) \tag{4}$$

Translating the algorithm in the hardware involves the following design block/steps:

- Detect the short preambles to trigger the change of integration time (from L = 128 to L = 256).
- Calculate the frequency offset by calculating the correlation energy and performing the arctan operation.
- Employ a phase accumulator to latch on to the correct phase at the end of the coarse and fine estimates.
- Finally, use a frequency synthesizer to generate the desired frequency equivalent to the calculated phase offset.

3.5 Long Correlation and Packet Timing

Packet timing is obtained from the long preamble. The long preamble consists of two symbols, each 256 samples long preceded by a 128 sample long cyclic prefix. Correlation is performed with a local copy of the long preamble. To reduce design complexity and eliminate floating point multiplication we decompose the correlator to simple logical operations as follows:

1. The "sign" bit of the I and Q samples is used instead of the actual values to eliminate any floating point operation.
2. The local copy of the long preamble also consists of the sign of the time domain signal.
3. Since the objective of correlation is to search for an exact set of samples as stored in the receiver, a XNOR operation is used to compare the likelihood between the incoming signal and the local copy of the long preamble. Whenever the sign of the input sample matches that of the local copy the output is a '1' else '0'.

The output of the long correlator is shown in fig. 2. The two peaks mark the end of the two long preamble symbols. The second peak is detected and marks the end of the packet preamble and start of the "signal" symbol. This signal is used to trigger the FFT block.

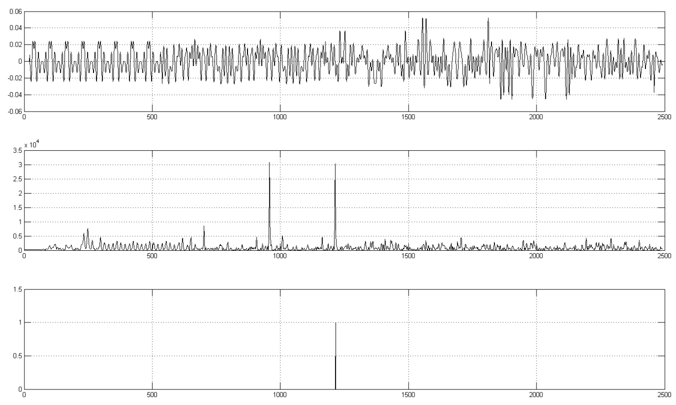


Figure 2: Long Correlator Output

3.6 De-prefix and FFT

Every OFDM symbol carries a 64 sample long cyclic prefix which are removed from every OFDM symbol before performing the FFT. The FFT engine used here is the IP block from Xilinx, which is a part of the Xilinx System Generator development platform. The FFT trigger input from the long correlator is used to load data into the FFT engine. The effect of deviations in this timing has been discussed in [3], [5], [4], [6]. Therefore, this timing is of paramount importance when it comes to successfully decoding an OFDM symbol. The FFT engine is operated in "pipelined streaming mode" [13] and has been optimized for speed. However the latency in the FFT engine is significant compared to the processing delay of the entire receiver pipeline. Detailed analysis of latency has been done in section 5. The output of the FFT produces the frequency domain representation of the OFDM symbol and the subcarriers are indexed from -128 to 127 for a 256 point FFT.

3.7 Equalization

The equalizer is an integral part of the receiver. Many equalization algorithms have been discussed in [3], [5], [4], [6], [14], [9], and [15]. The equalizer is responsible for correcting any phase and magnitude errors in the signal introduced by the time-varying, frequency selective wireless channel. Pilot subcarriers, inserted according to a predetermined order, are often used to estimate the channel. The channel estimate for the intermediate subcarriers can be obtained by interpolation between the pilots. For 802.11a/g four pilot tones are inserted in subcarriers [-21 -7 7 21] and are used to estimates the channel. Let there be N_p pilots in one OFDM symbol uniformly inserted S subcarriers apart, where $S = 14$. The receiver knows the pilot locations $\bar{P} = [P_k]^T$, (k = 0,…,$N_P - 1$), the pilot values $\bar{X}^P = [P_k]^T$ (k = 0,…,$N_P - 1$), and the received signal $\bar{Y}$. Given this information, a least squares estimate can be made at the pilot location given by,

$$\hat{H}^p_{LS} = \left[\frac{Y(P_0)}{X^p_0} \cdots \frac{Y(P_{N_p-1})}{X^p_{N_p-1}} \right]^T$$

The task here is to estimate the channel condition at the data subcarriers given the LS estimates at the pilot subcarriers $\hat{H}^p_{LS}$ and the received signal $\bar{Y}$. A piece-wise linear interpolation is done to estimate the channel between two pilots given by,

$$\hat{H}(kS+t) = \hat{H}^p_{LS}(k) + \left[\hat{H}^p_{LS}(k+1) - \hat{H}^p_{LS}(k) \right] . \frac{t}{s}, \quad 0 \le t \le S$$

Linear interpolation is easy to implement in hardware as it requires one accumulator unit to perform the task. However, second order interpolation involving three or more pilots could also be used at the expense of more hardware resources. Therefore, from a design stand point, the trade-off between performance and resource is a key factor to be considered.

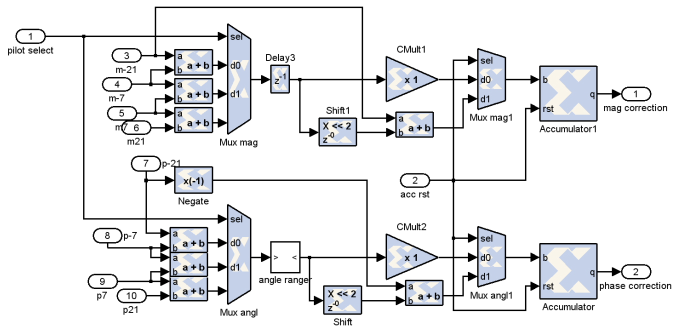


Figure 3: Linear Interpolation Equalizer

The interpolation block is shown in fig.3. The constant multiplier multiplies the difference between two pilots by 1/S = 1/14 and the accumulator accumulates the results for every t∈ [1 : S] and adds to the previous pilot estimate. The output provides the channel estimates for all the data subcarriers. Once the channel estimates of the subcarriers are obtained at the output of the interpolator the received subcarriers $\bar{Y}(k)$ are corrected as follows,

$$angle\left[\hat{X}(k)\right] = angle\left[Y(k)\right] - angle\left[H_e(k)\right] \tag{5}$$

$$mag\left[\hat{X}(k)\right] = \frac{mag\left[Y(k)\right]}{angle\left[H_e(k)\right]} \tag{6}$$

Where $H_e(k)$ are the channel estimates obtained from the linear interpolator block. The I and Q signals are computed from the magnitude and angle as follows,

$$I_{corrected} = mag\left[\hat{X}(k)\right] . \cos(angle[\hat{X}(k)]) \tag{7}$$

$$Q_{corrected} = mag\left[\hat{X}(k)\right] . \sin(angle[\hat{X}(k)]) \tag{8}$$

Fig. 4 shows the magnitude and phase interpolation of a BPSK modulated OFDM symbol. The red colored samples show that the pilots have been changed by the time varying frequency selective channel. With the prior knowledge of the pilots we know that the magnitude of the pilots as transmitted is “1” for all pilots and phase of the pilots are $[0, 0, 0, \pi]$. Fig. 4 also shows that the magnitude and phase of the pilots are restored to their original transmitted values. The uneven magnitude is largely due to limited precision in a fixed point design.

3.8 Demodulation

The demodulator follows the equalizer. The demodulator is a maximum likelihood (ML) baseband demodulator which performs a threshold test as per the symbol energy specified in 802.11a/g specification [8]. Decision boundaries are given by the perpendicular bisector of the line joining the two symbols.

The opposite symmetry of the constellation, except for the sign bit b_0, helps in reducing the search space for ML decoding by using two sets of threshold tests on the I and Q signals. The output of the demodulator is always a 6 bit number which is the maximum number of bits per subcarrier for a 64QAM modulation. The proper number of information bits are extracted in the de-interleaver, thus ensuring that only the information bits are extracted. A key point to this demodulator is a feedback loop from the signal symbol decoder. The signal symbol contains the information about the modulation and coding rate of the entire payload that follows the signal symbol. A feedback loop requires buffering of samples until the signal symbol is decoded.

3.9 De-interleaver, De-puncture, Viterbi Decoder and De-scrambler

3.9.1 *De-interleaver*

The interleaver is a block interleaver with a block size corresponding to the number of information bits in a single OFDM symbol denoted by N_{CBPS}. The implementation is done using M-Code and is a combination of slicers that rearrange the ordering of the bits in a specified order and then concatenate them back into a signal vector.

3.9.2 *De-puncture*

There are two types of puncturing in 802.11a/g. The specification manual [8] shows the detailed process of puncturing

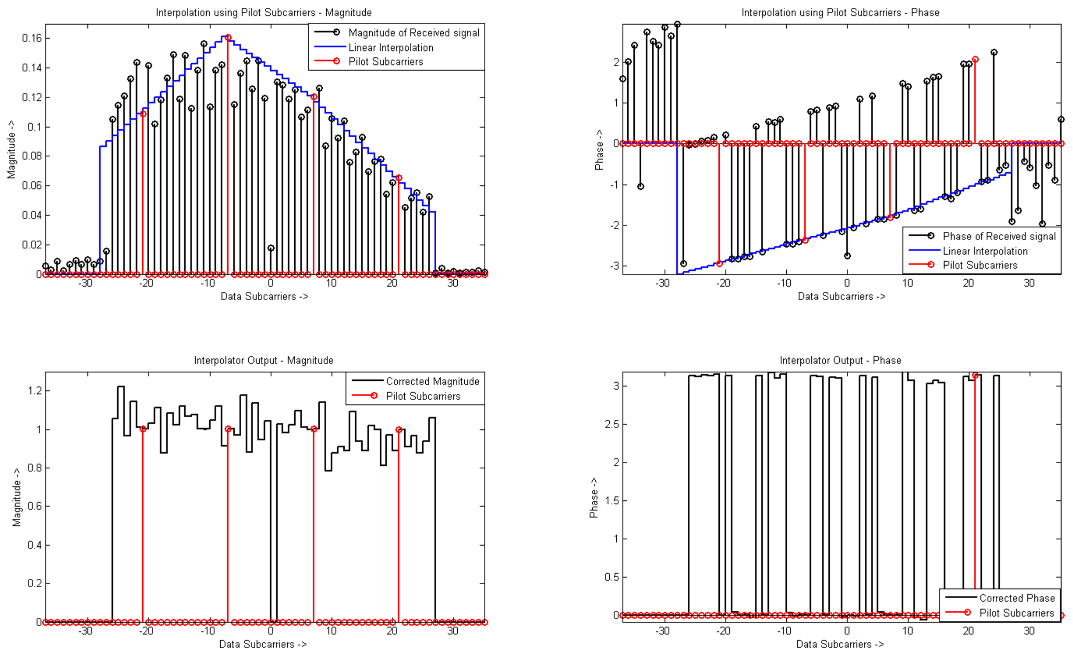


Figure 4: Linear Interpolation Equalizer Output

and de-puncturing the data bits. The de-puncture block is also implemented using M-Code, which inserts the dummy bits at appropriate indices.

3.9.3 Viterbi Decoder

The Viterbi Decoder uses trellis based decoding. For implementation, we have used an open source verilog implementation of the trellis based Viterbi decoder [16]. The block has been optimized for memory utilization and speed. However, additional hardware is required for this block to work in the form of additional buffering and control signals.

3.9.4 De-scrambler

The de-scrambler is just the inverse of the scrambling operation as mentioned in the specification [8]. The same scrambler unit is used to scramble and de-scramble the data. Scramblers are essentially a chain of shift registers with appropriate taps XOR-ed together to form the scrambled data.

3.10 Signal Symbol Decoder

The signal symbol is the first symbol in the physical layer data payload. This header symbol contains all of the information required to decode the rest of the frame. It contains 24 control bits divided into various fields as defined in [8]. The decoder is a set of 24 single bit registers connected in a chain and controlled by a clock that counts from 0 to 23. At the end of the 23^{rd} count the outputs are latched into the appropriate output ports, one for each field in the signal symbol.

4. OFDM RECEIVER - PHYSICAL LAYER ROUTER

4.1 Frequency Agile Receiver

In this section, the OFDM receiver has been customized to be able to decode the information bits from a certain set of subcarriers, called subchannels. For this implementation the following points are to be taken into consideration:

- To make the transmission frequency agile, 52 subcarriers are split into two subchannels, subchannel #1 has subcarriers -26 to -1 and subchannel #2 has subcarriers 1 to 26. The information bits from two users are processed separately for encoding, interleaving and puncturing and then placed on two subchannels.
- The signal symbol is the key to decode on subchannels, as it contains the information about modulation type and coding rate of the packet and its length. Since the signal symbol is still transmitted on all subcarriers it should be decoded as any other standard OFDM symbol and its information should be used to change the receiver pipeline to tune to a particular subchannel.
- The equalizer needs to be changed to equalize on one subchannel using a fewer number of pilots. Using linear interpolation and using fewer number of pilots will lead to a lower precision in the equalized signal. For subchannel #1 pilots $[-21, -7]$ are used and for subchannel #2 pilots $[7, 21]$ are used.
- For simplicity the encoding and length of the data from two users are kept same. This structure lets us use the same signal symbol configuration without adding additional fields/symbols for two separate users. The signal symbol can be modified to accommodate the encoding information of two separate users.
- Since the de-interleaver requirs data bits on all subcarriers, as that is how it was encoded at the transmitter, two successive OFDM symbols with data bits on either subchannel #1 or subchannel #2 are to be merged to form 52 subcarriers and then fed to the de-interleaver. Therefore the de-interleaver and the Viterbi decoder

remain unchanged for a full channel decoding or for a subchannel decoding. This helps in reducing the requirement of control signaling for all the receiver blocks. Another way of doing this would be to change the transmitter to interleave and encode on only the set of subcarriers allotted to a particular user rather than performing the encoding on all subcarriers. This will require additional control signals at the transmitter as well as at the receiver which involves additional hardware.

- Control signaling to tune receiver to a particular subchannel can be done in two ways. One way is to embed that information in the signal symbol which is to be decoded first to extract information required to decode the packet. Otherwise it can be controlled by the MAC layer in the form of a software control, if we can think of some kind of out of band signaling involved before the actual transmission begins which sets up the receiver pipeline accordingly.

4.2 Frequency Domain Switch/Router

In this section we present an addition to the existing OFDM receiver so that it can work as a frequency domain switch. This allows the receiver to work as a full-duplex transceiver rather than a conventional half-duplex transceiver. As claimed by [17] we can use the physical layer to route packets. But in order to do this we need a platform that supports the subchannel switching with control signals either embedded in the received packet or using software control from the host controller. Fig.5 shows the schematic for a frequency domain switch. This type of full duplex transceiver requires the use of two sets of radio front-ends. With advances in fabrication technology cost of front-ends will certainly not be a hindrance to this architecture. Using the receiver and the transmitter as a pipeline without any feedback loop and negligible turnaround time between the receive and transmit mode, latency in multihop mesh network will reduce greatly as also mentioned in [1]. With this configuration of the receiver, a path access mechanism can be designed in a higher layer that will be responsible in setting up the path and allocating subchannels and then the intermediate nodes can act as a pipeline to switch incoming packets on-the-fly onto another subchannel without mutual interference while using a full duplex mode of communication.

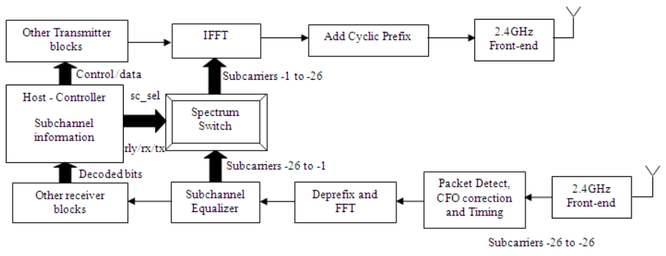


Figure 5: Frequency Domain Switch

Designing such a frequency switch has an additional challenge due to the fact we have adapted an existing 802.11a/g design and the fact that we continue to support 802.11a/g in the new design. The preamble is still used for packet detection and CFO estimation and the preamble and the signal symbol are transmitted using the full spectrum. This imposes a restriction on using the receiver as a full duplex relay because the relayed signal going out of the transmitter block cannot use the full spectrum to transmit the preamble and the signal symbol required to detect and decode the relayed packet in the next hop. This is because the receiver pipeline is still receiving data on at least one of the subchannels while it is transmitting on the other.

A possible and convincing solution to this situation is to change the preamble and the signal symbol to transmit only in the subchannel that is transmitting and not all the subcarriers. To do this either the sequence of the long and short preamble symbols needs to be changed or it has to be spread onto multiple symbols on the transmitting subchannels only.

Another plausible solution is to use some sort of a label so that the receiver is able to detect and decode the packet. This type of feature can be found in Multiprotocol Label Switching.

Lastly, if we can use the transceiver in a TDMA environment, then it would eliminate the packet detect and synchronization procedure. Not only would it eliminate all the additional hardware required for packet detection and packet timing, it would also allow the relayed signals to be transmitted without interference and also decoded at the next hop. But this method would require prior handshaking between the transmitter and the receiver to align their respective clocks. Also clock drifts play a crucial role in TDMA networks. Since OFDM is a FFT based modulation scheme, exact alignment of the FFT window at the receiver is very important. Therefore performance of TDMA networks and eliminating the packet detect and synchronization procedure is something to be investigated and then implemented using this transceiver.

4.2.1 Design and Implementation

Since switching subcarriers involves rearrangement of the subcarriers, buffering of samples is an integral part of the design. For this implementation we keep the same transmitter design as described in section 4.1 with two subchannels. The selection of the subchannel is controlled by the host controller. The host controller also has control signals to either receive the packet or relay it towards the destination. Decoding can be performed on either subchannels as shown in section 4.1 or on all subcarriers as done in conventional 802.11a/g compliant OFDM receivers.

The equalizer performs equalization only on the subchannel selected as mentioned in section 4.1. The equalization is required to restore the modulation levels of the subcarriers. Otherwise, the destination node will not be able to equalize the the cumulative effect of multiple frequency selective channel over which the signal has been relayed through multiple intermediate nodes. For IFFT, the IFFT unit from the transmitter is used. This calls for additional control signals for the transmitter as well to distinguish between the data coming in from an originating node or a relayed signal.

The spectrum of the incoming signal and the operation of the switch is shown in Fig.6. The top figure shows the incoming signal having data on all subcarriers. The second one shows that only information contained in subchannel #2 has been separated and equalized. This signal with data subchannel #0 is fed to the frequency switch and switched over to subchannel #1 as shown in third figure, followed by transmission by the transmitter front-end.

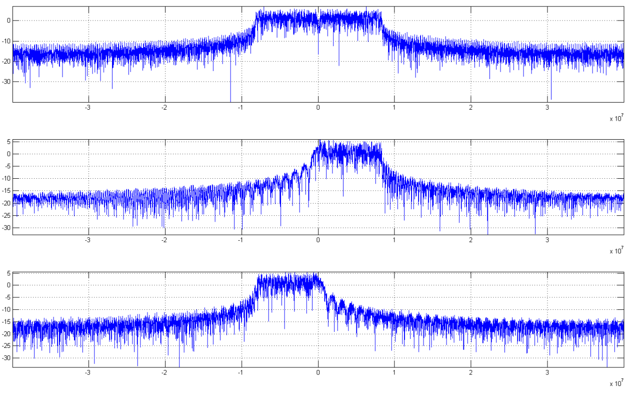


Figure 6: Input and Output Spectrum for Switch

5. PERFORMACE ANALYSIS

This section shows some key results and performance of different receiver subsystems. To show this we have used actual 802.11a/g packets captured in an indoor environment from a non-line-of-sight access point placed at an approximate distance of 15m using a suitable front-end radio to obtain the baseband samples. The receiver has also been tested with different types of modulation.

5.1 Packet Detection and Timing

In section 3.3 and section 3.5 the functioning of the packet detect and timing has been detailed. Fig. 1 and Fig. 2 shows the output of the packet detect and timing in absense of noise. Fig. 7 shows the output of the packet detection and timing blocks in presence of noise. Although the energy detector shows multiple detection, we actually do not care about those because it is the first rising edge that detects the packets and latched for setting up the entire receiver pipeline. It is clear that the shape of the short preambles symbols are far from ideal. But since the packet detection uses the periodicity of the preambles, autocorrelation still exists which is seen in the second subfigure. Therefore it is the relative energy that is important in packet detection and not just the absolute energy in the signal.

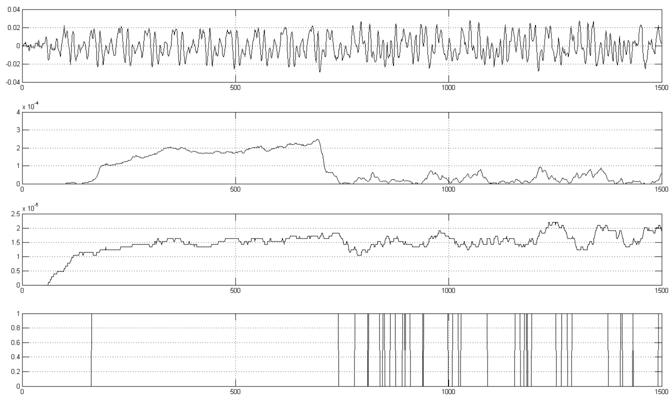
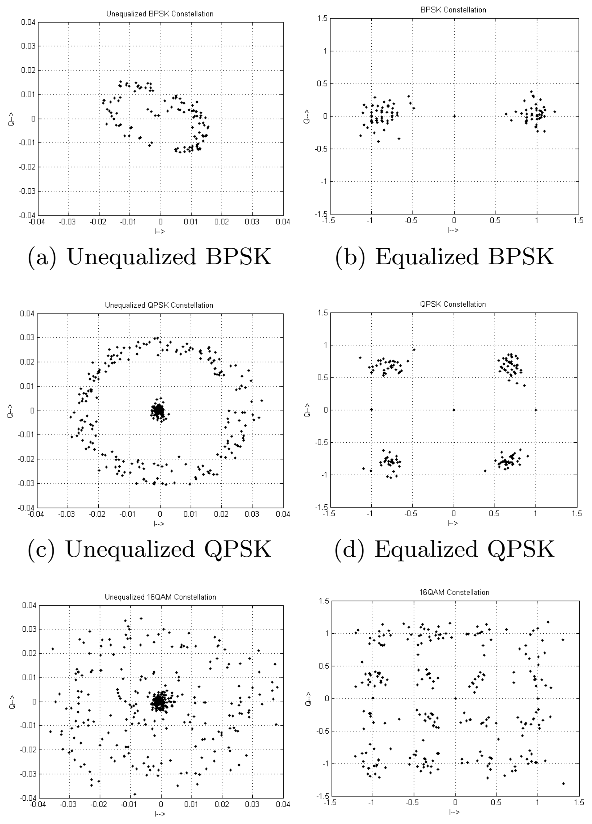


Figure 7: Packet Detection in Presence of Channel Noise

5.2 Equalizer and Channel Estimation

5.2.1 Equalizer Performance

Fig. 8 shows the constellations for BPSK, QPSK and 16QAM modulations. It is seen that BPSK and QPSK are decoded with high accuracy due to their large noise margin only if they have the required signal to noise ratio and have enough signal power to pass the threshold test of the packet detect and timing blocks. As the constellations get tighter, the margin of error reduces and decoding becomes more and more difficult. But the interleaver, FEC and scrambler still allows us to decode information bits under poor channel condition. For equalization the linear interpolation technique has been employed.



(a) Unequalized BPSK (b) Equalized BPSK

(c) Unequalized QPSK (d) Equalized QPSK

(e) Unequalized 16QAM (f) Equalized 16QAM

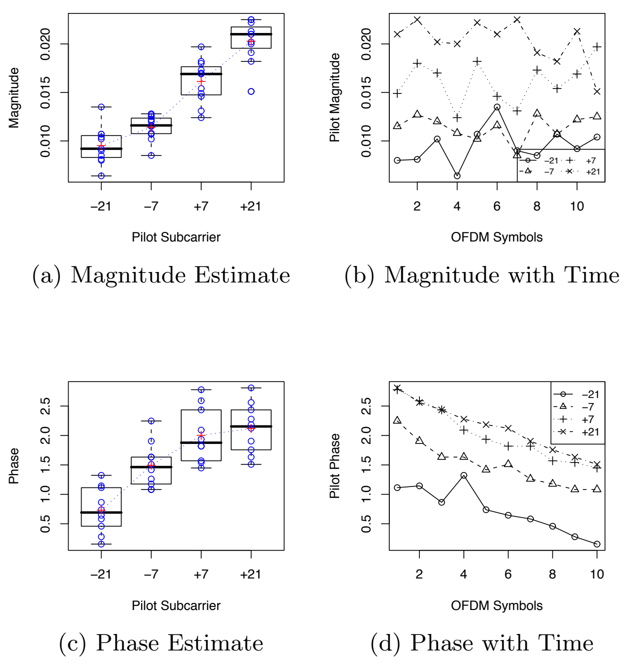
Figure 8: Equalizer Performance

5.2.2 Channel Estimation

In order to provide a better picture of the channel used for the experiments we plot the channel estimates as given by the pilots. Fig. 9 shows the channel estimates for a BPSK modulated pilot subcarriers for 11 OFDM symbols = 44μs. the variance for four different pilots is certainly not the same and also their ranges are different. It is of interest to note that the channel changes not only over multiple symbols but also within one symbol.

This exercise has been carried out to assess the need for the pilot symbols in every OFDM symbol. Since pilot subcarriers occupy certain parts of the spectrum it would be

Digital Object Identifier: 10.4108/ICST.WICON2008.4984
http://dx.doi.org/10.4108/ICST.WICON2008.4984



(a) Magnitude Estimate (b) Magnitude with Time

(c) Phase Estimate (d) Phase with Time

Figure 9: Behavior of Pilot Subcarriers

helpful if we can still decode the information bits without the pilots. This is helpful when we split the spectrum into smaller subchannels then with the current setup a minimum ratio is to be maintained between subcarriers and pilots. Another method that has been investigated in [15] is to use the long sequence of known subcarriers at the begining of the signal symbol. This known sequence can be used to estimate the channel over all subcarriers. But as shown in Fig. 9(b) and Fig 9(d) the pilots vary differently over time and the channel is independent from one symbol to the other. Therefore we cannot use the channel estimate from one symbol to equalize a different symbol.

5.3 Hardware Utilization

The overall hardware utilization is shown in table 1 for the FPGA platform chosen for this implementation. The following functional blocks have been included in the transceiver: 1) The OFDM Transmitter, 2) The OFDM Receiver, 3) Frequency Switch, 4) Subchannel Receiver, 5) PCI Interface and DMA transfer Logic, 6) ZBT Memory Module required for the transmitter [2]

The receiver block can be further broken down to smaller subsystems and hardware utilization of individual subsystem has been shown in table 2. The equalizer consumes the maximum hardware because of the use of cordic arctan, division and sin/cos IPs. Additional control hardware is required to accomodate the full spectrum equalizer as well as the subchannel equalizer.

5.4 Receiver Latency

Receiver latency is required to determine the turnaround time for the receiver, which also affects the latency in the network. Identifying latency in different blocks helps to identify bottlenecks in the pipeline and motivate design. It seems that the FFT core has the maximum latency. Usually for any practical transceiver, the minimum time that is required for the MAC/PHY to receive the last symbol of a frame at the air interface process the frame and respond with the first symbol on the air interface of the response frame is of great interest. This includes receiver side Phy layer processing delay + MAC processing delay + Transmitter side processing delay + PCI transfer delay for both Rx and Tx + Front-end radio hardware delay. If we disregard the MAC processing delay and the PCI transfer delay then we can summarize the following:

1. Receiver side: Difference between the last symbol received at the air interface to last bit transfered to host = 14.83μsec.

2. Transmitter side: Difference between the FIFO read signal to the first analog sample out from the DAC = 11.68 μsec.

3. Time to relay a signal on the fly after switchng subchannels = 18.8 μsec. This includes the time for FFT and then time required to perform the IFFT at the transmiter side.

4. Key note: The FFT/IFFT module consumes the bulk of the latency = 7.4 μsec. x 2 (for Tx and Rx) = 14.8μsec.

It is observed that most clock cycles are consumed by the FFT/IFFT unit and other than that the latency is attributed largely to various buffering elements required for proper functioning of the pipeline. In order to further reduce latency we need to use better pipelined cores with faster cycle times.

Table 1: Transceiver Hardware Utilization

Parameter(#)	**Count**	**Util.**
Slices	14343 out of 15360	93%
Slice Flip Flops	21837 out of 30720	71%
Total 4 input LUTs	24834 out of 30720	80%
Used as Logic	17493	
Used as shift registers	5992	
Used as RAMs	128	
FIFO16/RAMB16s	107 out of 192	55%
DCM_ADVs	3 out of 8	37%
DSP48s	114 out of 192	59%

Table 2: Receiver Hardware Utilization

Subsystem	**Slice**	**FF**	**LUT**	**Emb Mult**
Pkt_det & Timing	2553	2837	2968	11
CFO Correction	1556	2218	2540	22
Deprefix and FFT	2427	3482	3280	46
Equalizer	3023	3373	5275	1
Data Buffer	834	1473	1496	0
Demodulator	144	129	250	0
Deint, Viterbi, etc	1311	947	2031	0
Freq Switch	244	265	413	0
Total	**12092**	**14724**	**18253**	**80**

6. FUTURE WORK

The transceiver described in this paper has the potential to open new avenues in future research in the field of wireless mesh networks. A relay oriented physical layer hardware allows us to route packets with little or no involvement of the higher layers. In order to accomplish this, interference aware channel assignment mechanism and channel adaptation over multiple hops is very important. Also, scalability of the existing design is very important towards an actual deployment of a low latency mesh network. Therefore, we can lay out the following as possible additions to the existing design: 1) Customize the receiver to decode on all 256 subcarriers. For this we need to define pilots for FFT size greater than 64. 2) Embed sub-channel information in the signal field or define new protocols. Also define preambles for each sub-channel for synchronization without interfering with relayed signals. 3) Define channel assignment algorithms and path access mechanisms. 4) A very promising but challenging alternative would be to use TDMA to synchronize the transmitter and the receiver.

7. CONCLUSION

In this paper a complete design and implementation of an OFDM receiver in FPGA has been described. Performance analysis shows that the hardware works as good as a commercial signal analyzer with an OFDM decoder. We have also shown that using a small number of subcarriers compared to the FFT size and equalization with pilots based linear interpolation equalizer does not provide the best results as the channel starts to degrade or the constellation used gets tighter. However, the real challenge in multichannel cut through switching over multiple hops is the synchronization and signal detection not to mention the need of equalization. The synchronization and packet detection can be substituted with out of band handshaking with very accurate clocks at both ends. However, the narrow margin of error in the alignment of the FFT window between the transmitter and the receiver does pose a challenge in actual deployment of such system. Equalization has proved to be another important operation in the receiver or the relay chain. As mentioned in section 5.2.2, even for static nodes channel characteristics cannot be assumed to be constant over the duration of the packet transmission. Therefore, equalization still needs to be done on a symbol-by-symbol basis to ensure correct decoding of the information bits. To conclude, we also understand that in order for the transceiver to work as a physical layer router we need an equally intelligent path access control mechanism and interference aware channel assignment algorithms to route packets on-the-fly without involvement of higher layers.

8. REFERENCES

[1] R. Ramanathan, "Challenges: A radically new architecture for next generation mobile ad hoc networks," in *MOBICOM*, 2005, pp. 132–139.

[2] J. Fifield, P. Kasemir, D. Grunwald, and D. Sicker, "Experiences with a platform for frequency agile techniques," in *DYSPAN*, 2007.

[3] C. Dick and F. Harris, "FPGA implementation of an OFDM PHY," in *Signals, Systems and Computers, 2003. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 1, 9-12 Nov. 2003, pp. 905–909 Vol.1.

[4] M. Speth, S. Fechtel, G. Fock, and H. Meyr, "Optimum receiver design for wireless broad-band systems using OFDM - I," in *Communications, IEEE Transactions on*, vol. 47, Nov 1999, pp. 1668–1677.

[5] ——, "Optimum receiver design for OFDM-based broadband transmission - II. A case study," in *Communications, IEEE Transactions on*, vol. 49, Apr 2001, pp. 571–578.

[6] A. Troya, K. Maharatna, M. Krstic, E. Grass, and R. Kraemer, "OFDM synchronizer implementation for an IEEE 802.11(a) compliant modem," in *IASTED International Conference on Wireless and Optical Communication, 2002*. IASTED, 2002, pp. 152–157.

[7] "Rice university. WARP - wireless open-access research platform." [Online]. Available: http://warp.rice.edu

[8] "IEEE." [Online]. Available: http://standards.ieee.org/

[9] M. Wouters, G. Vanwijnsberghe, P. Van Wesemael, T. Huybrechts, and S. Thoen, "Real time implementation on fpga of an ofdm based wireless lan modem extended with adaptive loading," in *Solid-State Circuits Conference, 2002. ESSCIRC 2002. Proceedings of the 28th European*, 24-26 Sept. 2002, pp. 531–534.

[10] J. D. Guffey, A. M. Wyglinski, and G. J. Minden, "Agile radio implementation of ofdm physical layer for dynamic spectrum access research," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 26-30 Nov. 2007, pp. 4051–4055.

[11] J. Zhu, B. Bing, Y. G. Li, and J. Xu, "An adaptive subchannel allocation algorithm for ofdm-based wireless home networks," in *Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE*, 5-8 Jan. 2004, pp. 352–356.

[12] T. Schmidl and D. Cox, "Low-overhead, low-complexity [burst] synchronization for ofdm," in *Communications, 1996. ICC 96, Conference Record, Converging Technologies for Tomorrow's Applications. 1996 IEEE International Conference on*, vol. 3, 23-27 Jun 1996, pp. 1301–1306 vol.3.

[13] "Xilinx System Generator for DSP." [Online]. Available: http://www.xilinx.com/

[14] S. Coleri, M. Ergen, A. Puri, and A. Bahai, "Channel estimation techniques based on pilot arrangement in ofdm systems," in *Broadcasting, IEEE Transactions on*, vol. 48, Sep 2002, pp. 223–229.

[15] M. Serra, P. Marti, and J. Carrabina, "Implementation of a channel equalizer for ofdm wireless lans," in *Rapid System Prototyping, 2004. Proceedings. 15th IEEE International Workshop on*, 28-30 June 2004, pp. 232–238.

[16] "Perl script for viterbi decoder -." [Online]. Available: http://viterbi-gen.sourceforge.net/

[17] R. McTasney, D. Grunwald, and D. Sicker, "Low-Latency Multichannel Wireless Mesh Networks," in *Proceedings of the 16th International Conference on Computer Communications and Networks 2007 ICCCN 2007*. New York, NY, USA: IEEE, 2007, pp. 1082–1087.