Channel Sounding with Software Defined Radio

Harry W.H. Jones*, Pawel A. Dmochowski*, Paul D. Teal*
*School of Engineering and Computer Science, Victoria University of Wellington
NEW ZEALAND.
Email: harry.jones@ecs.vuw.ac.nz, pawel.dmochowski@ecs.vuw.ac.nz, paul.teal@ecs.vuw.ac.nz

Abstract: We present an investigation into the channel sounding of wireless communications systems using a Software Defined Radio (SDR) platform. Using the Universal Software Defined Radio (USRP), in conjunction with the GNURadio SDR system, we implemented and evaluated channel sounding capabilities using a variety of signal sequences. The final implementation utilized the IEEE 802.11a Orthogonal Frequency Division Multiplexing (OFDM) preamble for the tasks of carrier frequency and timing estimation, as well as channel sounding. We show that the system is able to characterize 1.8 MHz of spectrum reliably, with a resolution of 34 kHz. The bandwidth restrictions caused by the hardware bottlenecks were the main source of limitation.

Keywords: Channel sounding, Software defined radio, GNURadio, USRP

1 INTRODUCTION

In all mobile communication systems, it is essential to take into account the rapidly changing wireless environment. An estimate of the channel impulse response (CIR) or transfer function (CTF) can be used to equalise its effects at the receiver and to optimize transmission. The title given to the estimation of these characteristics is *channel sounding*. Providing accurate channel state information (CSI) to the transmitter can further enhance the use of the channel, although some form of prediction may be required to compensate for the latency in providing this information. Multiple-Input Multiple-Output (MIMO) systems particularly rely on reliable CSI, and some MIMO schemes require CSI at the transmitter.

Modern communications systems are required to be highly adaptive, due to the changing conditions of the wireless environment. Parameters within a system that may be adapted include modulation, power allocation and coding technique. Many functionalities, such as modulation, pulse shaping, constellation mapping and coding are typically done in hardware. Such architectures lack the ability to adapt during operation, and also complicate the task of system re-design. Software defined radio (SDR) offers a solution to a number of the issues described above by abstracting the tasks of constellation mapping, coding and pulse shaping from a hardware-based issue to a software one. This allows the engineer to change the specifications of a system by simply re-writing code, not by creating a new piece of hardware.

The aim of the research elucidated in this paper is to utilize these new devices and the powerful GNURadio framework to build a fully functional Single-Input Single-Output (SISO) channel sounder, and also to investigate the possibility of implementing a MIMO system. The final SISO sounder is able to characterise 2 MHz of unique bandwidth with a resolution of 38 kHz, utilizing the IEEE 80211.a Orthogonal Frequency Division Multiplexing (OFDM) preamble.

2 SOFTWARE DEFINED RADIO

A generic SDR requires only a Radio Frequency (RF) front end and a high-speed FPGA processor for digital signal processing. The basic idea is that the computer does most of the waveform specific manipulations, while the hardware provides sampling and amplification capabilities. Purely software radios are not yet used in real systems, but a progressively higher proportion of transceiver functionality is becoming software based. Software defined radios are often used in testing and prototyping new wireless systems. There are still some drawbacks to these devices however, mainly to do with limited hardware capability. ADC/DAC devices able to sample bandpass signals are generally not price effective, leading to many SDRs implementing a superheterodyne down conversion before digitisation. It is also difficult for these devices to sample

low power signals and thus a system normally requires a low-noise amplification stage before digitisation. There is also the bottleneck between the hardware front end and the software platform itself, in this research a USB cable. However, as the technology improves one can see software defined radio becoming the future of communications. This is due to the fact that the powerful software abstraction offered by SDR allows a robust system to be built easily depending on circumstance.

3 Implementation

The Universal Software Radio Peripheral [1, 2] is designed to allow users to program a high bandwidth radio using software. It does only the essential parts of the hardware front end, namely sampling conversion and RF amplification. The sampling conversion is done in both an Altera Cyclone FPGA and an AD9862BST ADC/DAC chip. The USRP offers up to 4 RF front ends, which are implemented on *daughterboards*. The front end is based on a superheterodyne structure. The USRP is designed to work with the GNURadio [1] system, which is an open-source platform.

3.1 Hardware

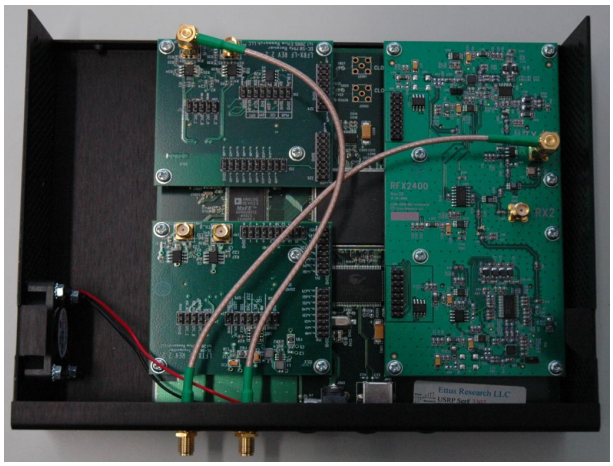

Figure 1: USRP motherboard with the low frequency receive and transmit daughterboards shown on the left, and the RFX2400 shown on the right.

The USRP motherboard, with 2 low frequency daughterboards and one high frequency transceiver connected, is shown in Figure 1. The USRP transmits and receives baseband data from the computer via a USB2.0 connection. A maximum rate of 32MB/s is able to be transmitted across the USB2.0 connection [1]. This limit was not always achieved due to the limited hardware of the host PC.

Figure 2 shows the ADC/DAC and FPGA sections of the USRP. For transmitted data, baseband data from

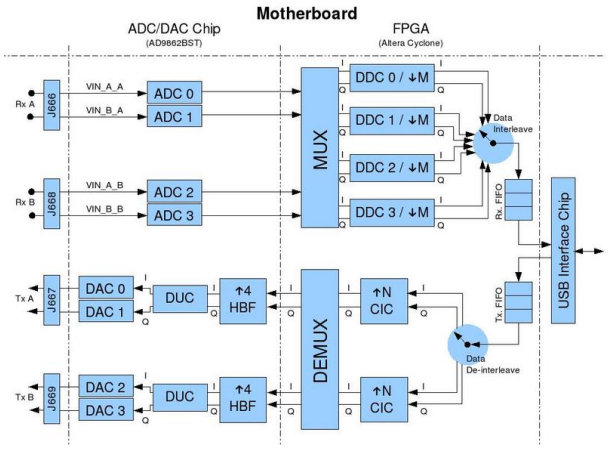

Figure 2: ADC/DAC and FPGA sections of the USRP, showing the sampling conversion stages and the multiplexer used for routing data. [1]

the computer at a certain sampling rate is transferred to the USRP. It is then routed by a demultiplexer to an up-converter and onto a digital to analogue conversion stage sampling at 128MS/s, before being transmitted via a daughterboard front end. On the receive end, the USRP receives the analogue waveform based at an Intermediate Frequency (IF) from the daughterboard and digitizes this at 64MS/s. The data is then routed by a multiplexer to a down-conversion stage, in order for it to be transmitted across the USB interface. The down-conversion is done on the FPGA embedded within the USRP, which implements Finite Impulse Response (FIR) Cascaded Integrator-Comb (CIC) and Half-Band (HB) filters in order to change sampling rate and to avoid aliasing. The up-converter consists of a CIC stage within the FPGA, as well as a final HB stage implemented in the DAC chip. The conversion stages are based on the standard I/Q quadrature multiplier.

The CIC and HB filters of the sampling conversion stages attenuate the spectrum of the baseband waveform near the Nyquist limit significantly. Figure 3 shows the magnitude response of the CIC-HBF receive filters, with a decimation rate of 32. This was kept for the rest of the research, as it avoided data corruption. Utilizing complex I/Q receiver sampling, this leads to a maximum sounding spectrum of 2 MHz.

The daughterboards act as the analogue radio frequency front end of the USRP. The USRP motherboard offers 4 different slots, two for receive and two for transmit. Each slot has access to two of the AD/DA slots on the AD9862BST. The RFX2400, which has a bandwidth between 2.4–2.5 GHz, was used for the final channel sounder.

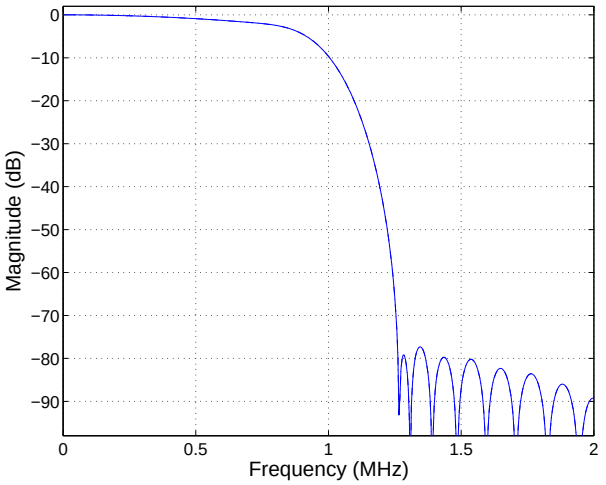

Figure 3: Combined receiver CIC-HBF filter response from DC–2MHz, with a decimation rate of 32. [3]

3.2 Software

The system was run using the NetBSD operating system, in conjunction with the GNURadio platform [1]. The platform is an open source library that contains a substantial library of prewritten C++ signal processing blocks. The blocks, representing sources, sinks and processing blocks have input and output ports connected together via the high level language Python. The GNURadio platform was utilized in conjunction with MATLAB, which performed all offline signal processing.

4 Implementation

4.1 Synchronisation techniques

The synchronisation of carrier frequency offset and timing are critical to the operation of the channel sounder. Carrier frequency offset (CFO) is caused by the inaccuracy of the receiver and transmitter oscillator clocks, as well as by the Doppler shift [4] caused by relative movement in the system. This causes the received signal to exhibit a low frequency modulation at baseband, even after the full down-conversion process. The basic effect of the CFO is that the demodulated signal is not centered at DC, rather it is centred at a certain δf, which is a function of the Doppler shift and oscillator offset. It is thus essential to take this into account when decoding data. In the project, a Minimum Mean Square Error (MMSE) estimate for the CFO named Moose's method [5, 6] was used.

Similarly, it is important to know the optimal timing instant to reduce the effect of any intersymbol inteference on the received signal. A popular algorithm used in OFDM systems to both detect received packets and to estimate the initial sample number of the packet is the Schmidl-Cox detection algorithm [7, 6].

Moose's method Moose's method is based around taking a L length sliding window correlation of the received data, where L is the repetition length of a certain sequence. A minimum of two repeated sequences are required to obtain an estimate of the CFO. In the project, this sequence was the 64-bin OFDM preamble used in the IEEE 802.11a standard [8]. The L length, L lag sliding window correlation product is defined as

$$P_l = r_{l+L}^H \times r_l \tag{1}$$

where the received sequence of length L, L ahead of sample l is correlated with the delayed sequence at sample l. This measures the phase offset between successive samples and by doing so over the whole sequence, one obtains a rate of change of phase. The MMSE estimate for the carrier frequency offset in Hz is given by [5]

$$\hat{\nu} = \frac{1}{2\pi LT} \arctan \frac{\Re(P_l)}{\Im(P_l)}. \tag{2}$$

An example of Moose's method, in the 2.4–2.5 GHz range, is shown in Figure 4. The algorithm estimates a carrier frequency of approximately 8.6 kHz, within the ±5ppm oscillator frequency uncertainty for 2.45 GHz quoted for the USRP [1].

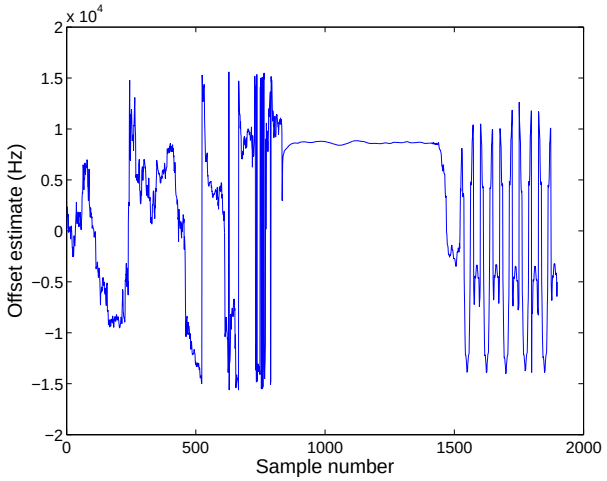

Figure 4: Moose's method estimates a CFO of 8.6 kHz as the packet is received at sample number 900.

Schmidl-Cox detection algorithm The Schmidl-Cox detection algorithm measures the energy difference between two successive repeated sequences and the current one. It uses the sliding window of (1) and divides it by the energy of the later sequence, given by

$$R_l = r_{l+L}^H \times r_{l+L}. \tag{3}$$

The Schmidl-Cox detection merit is then given by [7]

$$\rho_{sc} = \frac{|P_l|}{R_l} \tag{4}$$

where figure of merit ρ_{sc} will increase from 0 to 1 as the sliding window correlator encompasses a larger portion of the two repeated L length sequences. The detection merit will reach unity when the correlator encompasses both of the L length sequences, as P_l will be the same energy as R_l. Thus, the value of l that corresponds to the unity level is the initial sample of the packet. With this known, timing synchronisation is achieved. An example of the Schmidl-Cox detection algorithm output is shown in Figure 5. The detection merit rises to unity as a packet is received.

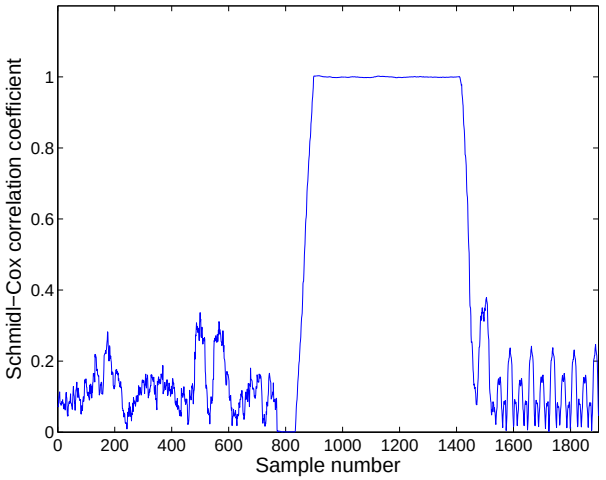

Figure 5: Schmidl-Cox detection algorithm increases from 0 to 1 as the packet is received at sample number 900

4.2 Channel sounding approach

Initial investigations focused on a time-domain cross-correlation [9] technique to find the CIR of a coaxial cable wired channel, utilizing the low frequency daughterboards. The sounding signals used were complex chirps and pseudo-random noise. At an early stage in the research, the response of Figure 3 was not fully understood and thus corrupted received signals. Without knowledge of the internal USRP filter response, the research moved toward a frequency-domain approach to channel sounding. This involved calculating and averaging the CTF, given by $H(\omega) = \frac{Y(\omega)}{X(\omega)}$, in conjunction with IEEE 80211.a OFDM preambles. The effect of the USRP filters was eventually understood and taken into account. It was now possible to implement a wireless system, utilizing the 2.4–2.5 GHz daughterboards. In order to reduce the spectral attenuation near the Nyquist bandwidth, the preambles were interpolated in software via zero-padding. Utilizing a high quality coaxial cable and the low frequency daughterboards, a wired equalisation was performed to take into account the action of the USRP filters. The assumption that the filter effect is similar on the low and high frequency boards is a valid one since they share the same ADC/DAC paths. The response of the the USRP was assumed to have linear phase, due to the FIR CIC/HB filters. Thus only the magnitude response was used to equalise the effects of the transmit and receive filters.

The final channel sounding approach was based on a waveform with three distinct sections, similar to the IEEE 80211.a OFDM standard. Figure 6 shows the received, CFO compensated, waveform.

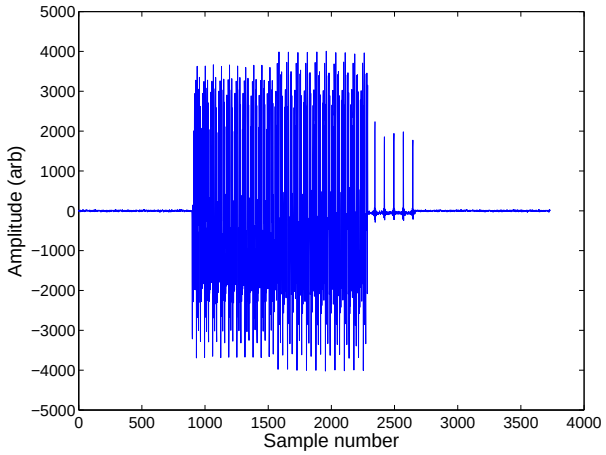

Figure 6: CFO derotated received waveform starting at sample number 900, showing the three distinct sections of the sounding sequence.

The first section consists of 10×64-bin 80211.a OFDM preambles encoded with BPSK. This was used for the synchronisation algorithms explained above. The MATLAB script applies both algorithms to the received data. It initially finds a detection peak in the Schmidl-Cox algorithm output, and then uses the averaged estimate of the CFO from Moose's method to correct the residual baseband oscillation. The 64-bin preamble was used because it offered an accurate CFO estimate and avoided the possible aliasing effects of Moose's method. The second section is used for calculating and averaging $H(\omega)$. It consists of the same 64-bin preambles as before, but these are interpolated up to a certain % of the Nyquist bandwidth before transmission. The preamble is initially zero padded to a certain percentage of Nyquist bandwidth in the frequency domain, before being transformed into the time domain via the Inverse Fast Fourier Transform. Only 53 bins of the preamble were utilized, due to the original spectrum consisting of numerous high frequency nulls. Thus to sound a bandwidth of 70%, the length of one preamble was 75. This waveform was then repeated 10 times, as in the first section. Figure 7 shows the spectrally flat 53-bin preamble, interpolated to 70% of the available bandwidth. The null DC component for power reduction is clearly visible and is seen in all subsequent results. The first preamble is used as a cyclic prefix, while the next 9 are used to

calculated $H(\omega)$. This also naturally takes into account the possible initial phase offset between the transmitter and receiver systems, as this is averaged over numerous runs. The third section consists of 5×53-bin QPSK encoded OFDM data symbols, used to ascertain whether an accurate estimate of the CTF has been found. As with the second section, these symbols are interpolated up to the same bandwidth of the sounding signal. The first symbol is used as a cyclic prefix, with the next 4 being used to check the CTF. The script applies a zero-forcing equalisation to the received synchronised data, then ascertains whether the correct QPSK data has been received with a threshold detection device.

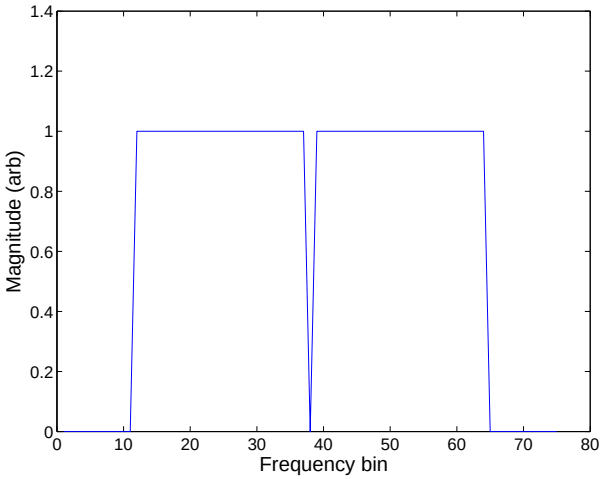

Figure 7: Spectrum of 53-bin 80211.a OFDM preamble interpolated to 75 bins, 70% of the original bandwidth.

5 Results

The channel sounding system was set up in an small office, with two stationary USRPs separated by 3m. An example of the averaged magnitude CTF at 2.45 GHz is shown in Figure 8. This depicts 70% of the available 2 MHz spectrum being sounded. It includes the channel sounding result with and without the equalisation of the internal USRP filters. As shown, a significant amount of the bandwidth is attenuated by the CIC and HB filters on both transmit and receive ends. The equalised CTF is shown to exhibit flat fading in this particular region of spectrum. The phase response over this same bandwidth is shown in Figure 9. It is clearly linear, leading to the conclusion that there was no significant distortion through the channel.

In order to ascertain the validity of the CTF estimate, the constellation diagram of the received QPSK encoded OFDM was plotted. An example of this is presented in Figure 10, showing the correct received data. The channel sounder was able to consistently sound up to 90% of the Nyquist bandwidth of the system. A fully automated system was programmed to allow the user to define the

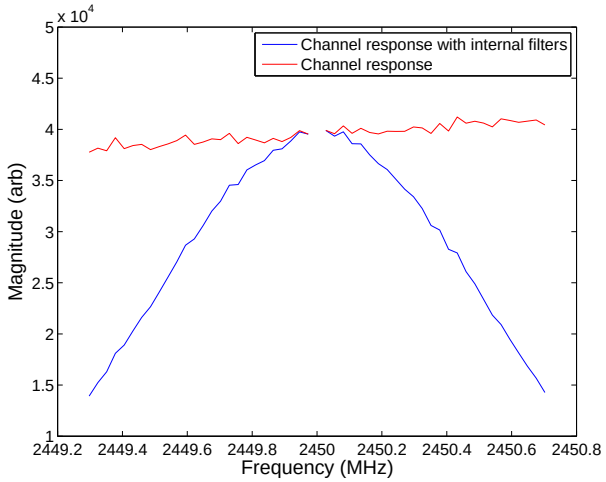

Figure 8: Magnitude CTF response centered at 2.45 GHz, with 70% of available bandwidth being sounded.

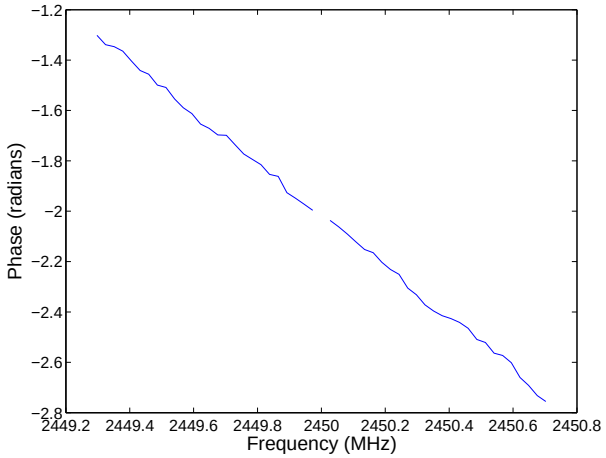

Figure 9: Phase response of CTF centered at 2.45 GHz, with 70% of available bandwidth being sounded.

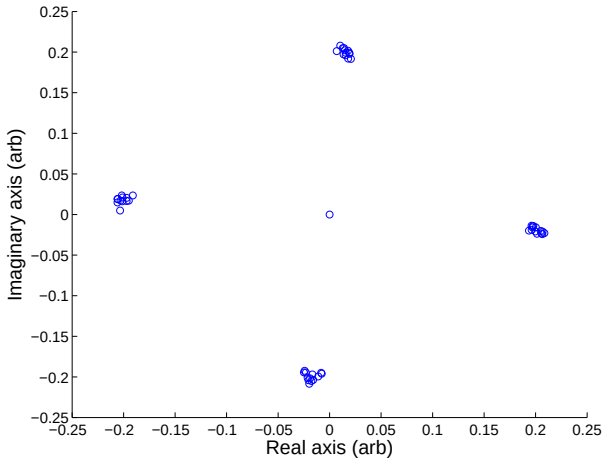

Figure 10: Constellation diagram of equalised received QPSK encoded OFDM data sequence

data encoded in the third section of the packet, while creating the correct waveforms for 5 different bandwidths. The wired response of the USRP filters to these bandwidths was also measured, in order to equalise their response.

Multiple-Input Single-Output (MISO) and Single-Input Multiple-Output (SIMO) systems were programmed with the GNURadio platform. These were tested but no further investigation was done into MIMO channel sounding due to time constraints. We view the use of orthogonal sets of sequences for this purpose as essential for further research.

6 Conclusions and further work

The project aim was to build a fully functional channel sounder utilizing the USRP and the GNURadio software platform. Initially, complex chirps and pseudo-random noise were used to sound a coaxial cable wired channel. The internal filter response was not understood and thus corrupted the results. Due to the unknown problem of the internal filter action of the USRPs, a move to the IEEE 802.11a OFDM preamble followed. This also lead to the utilization of the high frequency daughterboards of the USRP for a wireless implementation.

The final form of the channel sounding waveform consisted of three distinct sections. The first used the 80211.a preamble for a robust synchronisation algorithm, involving the Schmidl-Cox detection merit and Moose's method for carrier frequency offset estimation. The second was for calculating and averaging the CTF, utilizing the same preamble. The third section was an appended QPSK encoded OFDM data waveform, for testing the validity of the CTF. The response of the USRP filters was equalised utilizing a high quality coaxial cable, allowing the actual CTF to be measured. By checking the received QPSK data with a threshold decision device, the validity of the estimated CTF was verified. Although not significant enough to cause an erroneous decision, the data occasionally exhibited a phase rotation of up to $\frac{\pi}{8}$. This was due to the inaccuracy of Moose's method and the phase drift between the CTF averaging section and the received data. The hardware setup had two main bottlenecks, namely the USB cable and the PC itself. This lead to a minimum decimation rate of 32 with a sampling frequency of 64MS/s, leading to a maximum sounding bandwidth of 2 MHz. Numerous percentiles of that bandwidth were able to be sounded with the system, depending on the relative importance of range versus precision. Using the full bandwidth and the 53-bin IEEE preamble leads to a resolution of 38 kHz. This implementation was not as reliable as the system that sounded 90% of the available Nyquist bandwidth, leading to a sounding range of 1.8 MHz and resolution of 34 kHz. With this system operational, a fully functional channel sounder for use in the 2.4–2.5 GHz range was built using the USRP and the GNURadio platform.

The following extensions and improvement are envisaged as a continuation of this research:

- Implementation of a GNURadio C++ block for rapid detection using the Schmidl-Cox detection algorithm.
- Implement the channel sounder in the FPGA itself, bypassing the USB bottleneck and reducing the effect of the internal USRP filters.
- Upgrade computer hardware to allow lower interpolation and decimation rates in the current implementation.
- Upgrade to the USRP2 [1] platform, which improves all the ADC and DAC paths to allow far higher bandwidths. This system is based on an Ethernet interface, rather than USB.
- Use the XCVR2450 daughterboards to sound the 5 GHz band.
- Investigate other carrier frequency offset and detection methods that could improve accuracy.
- Mobile channel prediction.

References

[1] `http://gnuradio.org/trac`.

[2] `http://www.ettus.com/`.

[3] F. A. Hamza, "The USRP under 1.5X Magnifying Lens!," 2008.

[4] J. G. Proakis, *Digital Communications*. McGraw-Hill, 2001.

[5] P. H. Moose, "A Technique for Orthogonal Frequency Division Multiplexing Frequency Offset Correction," *IEEE Transactions of Communications, Vol. 42, No. 10*, 1994.

[6] A. J. Coulson, "Maximum Likelihood Synchronisation for OFDM using a Pilot Symbol: Algorithms," *IEEE Journal On Selected Areas In Communications, Vol. 19, No. 12*, 2001.

[7] T. M. Schmidl and D. C. Cox, "Robust Frequency and Timing Synchronization for OFDM," *IEEE Transactions of Communications, Vol. 45, No. 12*, 1997.

[8] `http://www.ieee802.org/11/`.

[9] R. J. Pirkl and G. D. Durgin, "Optimal Sliding Correlator Channel Sounder Design," *IEEE Transactions on Wireless Communications Vol.7, No.9*, 2008.