Parallel MIMO Multiuser RF Channel Measurements and Analysis with Focus on Impulse Response and MMSE Gain Variations with Antenna Placement

by

Wesley A. Miller

B.Sc.E., University of New Brunswick, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science in Engineering

In the Graduate Academic Unit of Electrical and Computer Engineering

Supervisor(s):	Brent R. Petersen, Ph.D., Electrical and Computer Engineering
	Bruce G. Colpitts, Ph.D., Electrical and Computer Engineering
Examining Board:	Dennis F. Lovely, Ph.D., Electrical and Computer Engineering
	Ian L. Veach, Electrical and Computer Engineering
External Examiner:	Ali Ghorbani, Ph.D., Computer Science

This thesis is accepted

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

April 2010

To my parents for their consistently sound advice and support.

Abstract

A multiple channel antenna testbed is constructed and configured for analysis of channel impulse responses. Transmit signals are generated using an Altera® Stratix® II field programmable gate array (FPGA) and hardware-modulated onto the radio frequency (RF) carrier. Received RF signals are down-converted to an intermediate frequency carrier using a commercial RF front end, sampled at FPGA analog-to-digital converters, and software-demodulated offline using MATLAB®. Matched sliding correlators are applied to the received signals to yield the channel impulse responses, which are then placed into a least-mean squares (LMS) adaptive filter simulation to evaluate minimum mean squared error (MMSE) performance for several indoor and outdoor antenna layouts in the 1900 (MHz) Personal Communications Services C2 band as per Standard Radio System Plan 510. MMSE values from simulation show improvement for the indoor far-spacing case over the indoor close-spacing case and the outdoor cases.

Acknowledgements

I would like to thank my supervisors Brent Petersen and Bruce Colpitts for their giving their tireless effort and significant expertise in making the achievements of this thesis possible.

Many thanks also go to the ECE administrative staff, Denise Burke, Karen Annett, and Shelley Cormier, as well as the CNSR Project coordinator, Lori Tozer, for their kind support throughout project development and marked efficiency in processing numerous purchase order forms. My thanks go out to Brian Guidry for construction of transmitter and receiver antenna stands in addition to cabinet bulkhead. His work made it possible to safely carry out system testing in a wide range of environments while still ensuring system device protection.

I would like to thank the ECE Technical Staff, with specific thanks to Bruce Miller for his assistance in routing several printed circuit boards and aiding with the timely acquisition of new equipment, Michael Morton for his assistance in RS-232 communications and finding/ordering required commercial hardware to specification, Nicholas Fowler for taking part in wiring amplifiers and printed circuit board construction, and Kevin Hanscom for his generous loan of coaxial cable connectors and adapters. Special thanks to Ryan Jenning for taking part in wiring amplifiers, helping with moving equipment when system radio channel testing was being carried out for indoor and outdoor locations, logging data from channel measurements to PC, and helping with securing equipment to the antenna stands.

I would like to thank Yi Fu and Florin Burlacu for their part in helping to carry out an abundance of system radio channel testing for both indoor and outdoor locations, as well as generating a database of complex channel impulse responses for these corresponding locations. This research was supported by the Atlantic Innovation Fund from the Atlantic Canada Opportunities Agency, and by Bell Aliant, our industrial partner.

Table of Contents

List of Tables

List of Figures

List of Symbols, Nomenclature or Abbreviations

ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
AWGN	Additive White Gaussian Noise
BER	Bit-Error Rate
BPSK	Binary Phase Shift Key
CIR	Channel Impulse Response
COS	Code-Operated Switch
COTS	Commercial Off-The-Shelf
DAC	Digital-to-Analog Converter
dBc	Decibels with respect to carrier
dBm	Decibels with respect to a milliWatt
dBi	Decibels with respect to isotropic radiator
dBW	Decibels with respect to a Watt
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input/Output
I	Inphase
IF	Intermediate Frequency
ISM	Industrial, Scientific, and Medical

LNA	Low Noise Amplifier
LRS	Linear Recursive Sequence
LVTTL	Low-Voltage Transistor-Transistor Logic
MSym/s	Mega-Symbols Per Second
MIMO	Multiple Input Multiple Output
MMSE	Minimum Mean Squared Error
MSPS	Mega-Samples Per Second
PCB	Printed Circuit Board
PCS	Personal Communication Services
PLL	Phase-Locked Loop
PN	Pseudo-Random Noise
Q	Quadrature
QAM	Quadrature Amplitude Modulation
RF	Radio Frequency
RMS	Root-Mean-Squared
RSS	Radio Standards Specification
SNR	Signal-To-Noise-Ratio
TTL	Transistor-Transistor Logic
V_{DC}	Volt Direct Current
V_{pp}	Volt Peak-to-Peak

Chapter 1

Introduction

There has been a significant amount of simulation-based research in the field of multiple input multiple output (MIMO) systems in the last decade, but recently the research interest has shifted to incorporate more real-world prototyping. At the time of this writing, quite a few universities and research institutions have already designed, constructed, and tested real world MIMO applications, but no such implementation of this scale previously existed at University of New Brunswick (Fredericton). In this thesis, which is part of a larger project called Communication Networks and Services Research (CNSR), a RF antenna testbed was constructed and implemented to support 16 wireless channels operating in parallel, and the system performance was evaluated on the basis of radio channel impulse response measurements and corresponding minimum mean squared error (MMSE) estimates.

1.1 Background and Literature Review

A channel impulse response (CIR) is the measure of a radio communications channel when an electromagnetic impulse is transmitted over air. In this respect, the radio propagation channel can be viewed as a time and spatially varying filter which distorts the transmitted signal of interest. The "filter" characteristics are mainly determined by both the reflection of electromagnetic waves off other objects and the movement of reflective surfaces in and around the path of propagation (multipath), as well as the movement of the transmitter and/or receiver. The purpose of multiple impulse response measurements is to quantify the varying filter characteristics of the communication channel with respect to both time and space. With respect to MIMO systems, channel impulse responses can be used for the investigation and quantification of improved channel capacity and system performance over traditional single input single output (SISO) systems. In the past decade, there has been significant work and achievements in the area of MIMO research. For example, the work done by David Gesbert et al. [3] examined different classes of proposed algorithms and techniques which attempt to realise the various benefits of MIMO including spatial multiplexing and spacetime coding schemes, and also investigates the integration of MIMO algorithms into current commercial standards such as third generation wireless (3G) and wireless local area network (WLAN).

One of the key methods in CIR estimations is the use of cross-correlation. Sharif et al. [4] found that, using pseudo-random noise (PN) sequences for signal generation, cross-correlating the received signal with a copy of the transmitted signal is a valid estimation technique for obtaining CIR. Most systems that utilise PN sequences in this manner use a specific type of PN sequence known as *maximum length* sequences. This specific type of sequence is quite useful in tracking impulse responses for a given number of repeat intervals of transmitted PN sequences.

1.2 Thesis Objective and Contributions

The objective of this thesis was to develop an RF antenna testbed with 16 wireless channels operating in parallel for examination of the effects of antenna positioning and separation on the system channel impulse responses and corresponding channel matrix MMSE estimations. This work is an continuation of the work done by J. Andy Harriman et al. [5], at which stage the testbed included two Altera® Stratix® II field programmable gate arrays (FPGAs), and other commercial off-the-shelf (COTS) equipment such as 4 external ADC daughter boards, a signal generator, Hittite wideband quadrature modulators, 2400 (MHz) ISM band antennas, AOR wideband receivers, and various lengths of RG-58 coaxial cable. A custom signal-conditioning printed circuit board (PCB), designed and developed by Harriman, was also constructed and installed on one of the FPGAs in order to perform voltage level translations on the signals being passed from FPGA general purpose input output (GPIO) pins to the Hittite modulator. Custom PCBs were also developed for DC biasing the Hittite modulators. One FPGA was used to generate the PN sequences for transmission and both FPGAs were used to acquire and process the four receiver intermediate frequency (IF) signals, with each FPGA supporting connections for two external analog-to-digital converters (ADCs). Harriman demonstrated successful single channel transmission, recovery, and demodulation (both in FPGA software and MATLAB®) to verify system operation.

In this work however, extra hardware (including splitters, amplifiers for both the transmitters and receivers, additional Hittite modulators, extra lengths of RG-58 coaxial cable, and 1900 (MHz) transmitter and receiver antennas) was purchased/developed and installed to reconfigure the testbed for simultaneous operation of four transmitters and four receivers at 1900 (MHz). Each transmit station Hittite modulator RF output was amplified by approximately 15 (dB) before transmission via RF antenna, and the radio IF outputs were amplified by approximately 25 (dB) before being sampled by FPGA ADCs. CIR matrices were implicitly obtained after baseband signal processing then placed in a LMS adaptive filter simulation to determine whether the obtained matrix could be used in a communications system. Eight tripods were each fitted with wooden platforms and used to house the equipment for each of four transmitters and four receivers. The equipment and cabling were secured onto the tripod stand by cable ties in effort to safeguard and ease the movement of the equipment during field testing. Four lengths of low-loss coaxial cable were also purchased for providing each transmitter with a high frequency local oscillator signal up to 90 (m) away from signal source generator. A testbed of this nature lends itself to many practical areas of communications research including multi-channel and MIMO system capacity measurements in a variety of test environments, signalling wavelength antenna placement (SWAP) gain analysis, and fourth-generation mobile communication technologies.

This system currently consists of eight antennas, one antenna for each of four transmitter stations and one for each of four receiver stations, and is configured for operation in the 1900 (MHz) PCS C2 band. Each transmitter station sends out a signal at 2.5 Mega-Symbols Per Second (MSym/s) on a carrier frequency of 1902.5 MHz, and each radio receiver is tuned to the same carrier frequency for proper signal reception. Each radio receiver down-converts its received high-frequency carrier to a 10.7 MHz IF signal, which is then passed to the FPGA for baseband signal processing. In any decentralized system, whether it is the receivers, the transmitters, or both that are decentralized, there is usually the need for pilot symbols for transmitter-receiver synchronization. However, as this system is a centralized approach, pilot symbols are not required. The FPGA, which is used for signal generation, and the radio receivers, which are used for reception, are all phase-locked to a common 10 MHz standard from a signal generator. It should also be noted that the system can be expanded to support more transmitters, via utilization of additional or different FPGAs, and more receivers, via additional analog-to-digital converters (ADCs) on the FPGA(s), for future research experiments.

1.3 Thesis Layout

The rest of the material in this thesis is presented in the four chapters which follow this section. Chapter 2 describes the various system modules, along with system architecture, as well as the actual equipment used for system testing, verification and debugging. Chapter 3 details the chosen method of signal generation and transmission, including transmission signal sequences, receiver frequency standard distribution and configuration, hardware design encapsulation, and baseband signal processing technique. Chapter 4 outlines the various test configurations used for system verification and results from effect of antenna placement and transmission sequence variation. Chapter 5 gives a summation of the thesis findings and intended future work.

Chapter 2

System Design and Layout

In this chapter, we describe the various design modules and hardware architecture, and also describe the selection of equipment used for testing and debugging system operation.

2.1 System Modules and Hardware Architecture

The RF antenna testbed system design has been divided into three main modules: transmit, receive and central control. During module construction it was also found that it was important to implement hardware encapsulation, with regard to cable connector points, in order to decrease cable pull on the connectors and to ease the process of system debugging. System module and hardware encapsulation details are given in the following subsections.

2.1.1 Transmit

Each transmit module consists of a large Antenna Factory FO1710-8, 8 (dBi) antenna, a Hittite HMC497LP4 wideband quadrature modulator (mounted on a evaluation PCB) with 5 (V) power supply, and a $\mathrm{Mini}-\mathrm{Circuits}^{®}$ ZX60-6013E+ low-noise amplifier (LNA) with 12 (V) power supply, all mounted on camera tripod with triangular wooden platform installed. The device interconnections are made using short lengths of RG-58 coaxial cable. Figure 2.1 illustrates a block diagram of setup and Figure 2.2 shows an actual photograph of a transmit station. The F01710-8 antenna data sheet [6] states it has an operating frequency range of 1710–1990 (MHz), and is therefore suitable for transmission in the 1900 (MHz) PCS C2 band. This vertically-polarized antenna also has a gain of 8 (dBi) which is oriented to maximize radiation at the horizontal plane. The HMC497LP4 quadrature modulator accepts baseband input for a frequency range of DC–700 (MHz), and local oscillator (LO) input between 450–4000 (MHz) [7]. The modulator uses quadrature amplitude modulation (QAM) to modulate the incoming inphase (I) and quadrature (Q) baseband signals supplied from the FPGA, onto the incoming LO signal which is supplied from an Aeroflex signal generator. The $\mathrm{Mini-Circuits}^{®}$ ZX60-6013E+ LNA, which has an operating frequency range of 20–6000 (MHz), is used to amplify the modulated RF output to the transmit antenna. However, since the FPGA uses the 0–3.3 (V) low-voltage transistor-transistor logic (LVTTL) standard and the modulator only accepts baseband input between 1.4 (V_{DC}) and 1.6 (V_{DC}), signal conditioning circuitry had to be implemented to interface signals from FPGA to the modulator. The signal conditioning circuitry PCB used in this project was developed by Harriman et al. [5], in which an illustration and schematic of said PCB is given. It mounts on top of FPGA general purpose input/output (GPIO) pins and is fitted with SMA connectors to facilitate connection with, and signal transmission along, SMA-connectorized RG-58 coaxial cable.

2.1.2 Receive

Each receiver module consists of an antenna for operation in the 1900 (MHz) frequency range, an AOR AR5000A wideband radio receiver, and a $\mathrm{Mini-Circuits}^{®}$ ZHL-

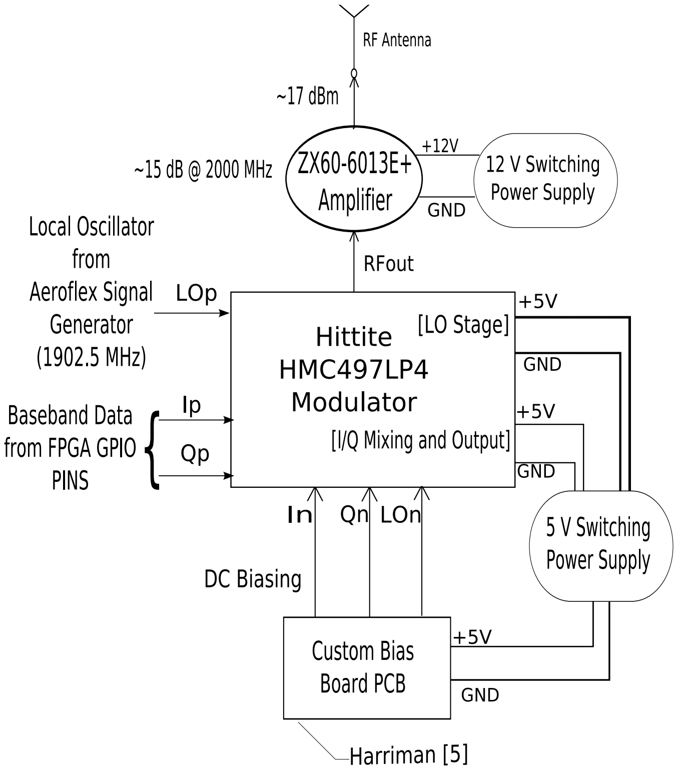
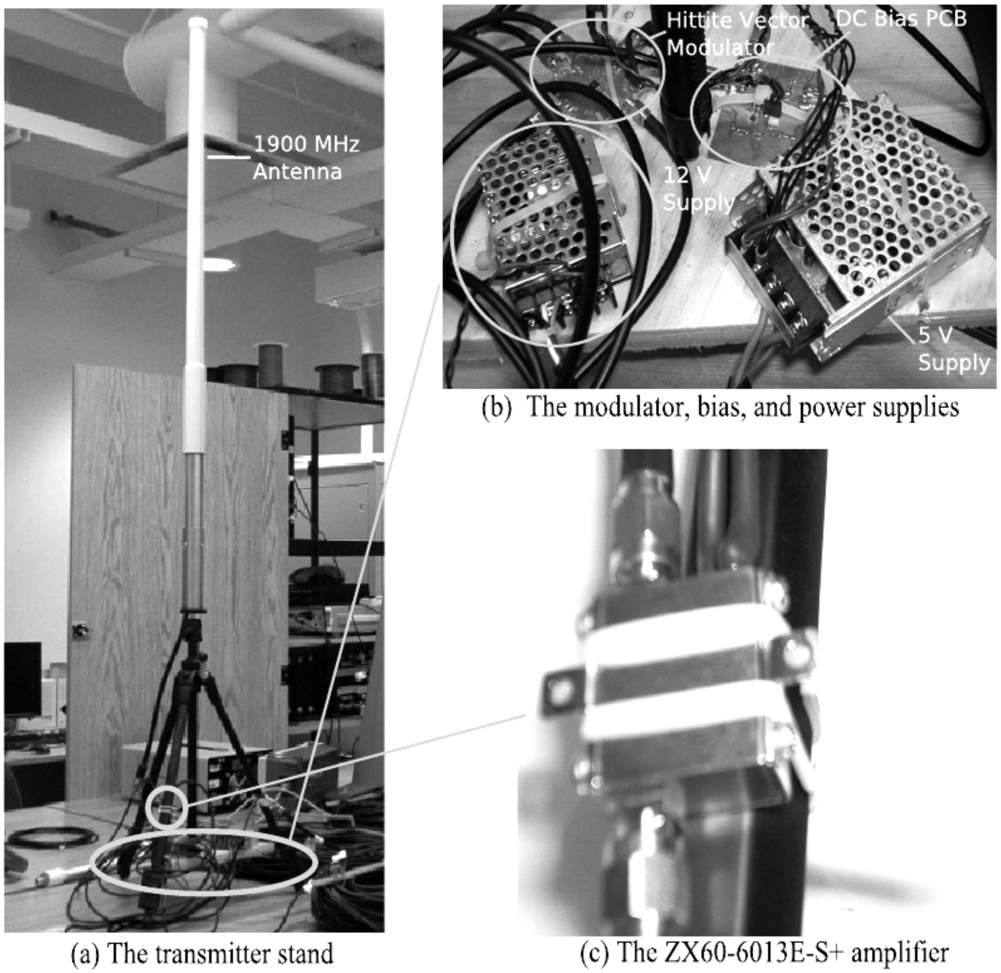


Figure 2.1: Transmitter Station Block Diagram



(b) The modulator, bias, and power supplies

(a) The transmitter stand

(c) The ZX60-6013E-S+ amplifier

Figure 2.2: Transmitter Station Photograph

6A medium power amplifier, all mounted on a camera tripod stand with a circular wooden platform installed. Figure 2.3 shows a block diagram of the setup and Figure 2.4 shows a photograph of the actual setup. During the channel measurement phase, two different sets of 1900 (MHz) antennas were used. Initially, system testing and debugging was carried out with a Nearson Inc. quarter-wave, 0 (dBi), rubber duck antenna with SMA plug connector installed at each receiver, which is shown in Figure 2.3. These tri-band antennas are specified for operation in the 890–960 (MHz), 1710–1810 (MHz), and 1850–1990 (MHz) bands. However, prior to the outdoor testing phase of the thesis, the decision was made to change these antennas with the aim of achieving improved received signal strength and quality. As a result, the outdoor tests described later in Chapter 4 were carried out using a quarter-wave, 3 (dBi), whip antenna with SMA plug connector installed at each receiver. The AR5000A receiver has a tunable frequency range of 0.01–2600 (MHz), which makes it suitable for operating in the 1900 (MHz) PCS C2 band. The receiver has an input connection for an external frequency standard (which in this case comes from the same Aeroflex signal generator that supplies the FPGA clock), an input connection for RF signal from receiver antenna, as well as an output connection for intermediate frequency (IF) output. The ZHL-6A amplifier has an operating frequency range of 0.0025–500 (MHz), stated as having a typical gain of 25 (dB) at 2000 (MHz) and 1 (dB) compression point of +22 (dB) [8]. It is used to amplify the IF output from the radio before propagation along the 90-(m) RG-58 coax to an FPGA ADC input. The total attenuation of this 10.7 (MHz) signal along the 90-(m) RG-58 coax was found to be approximately 4.4 (dB), but, without amplification, it was found that the IF output from each radio was of significantly low amplitude (in the order of 30 ($\mathrm{mV_{pp}}$)), making it necessary to use an amplifier in order to get observable signal at the other end of the 90-(m) cable run connected to the FPGA ADC input.

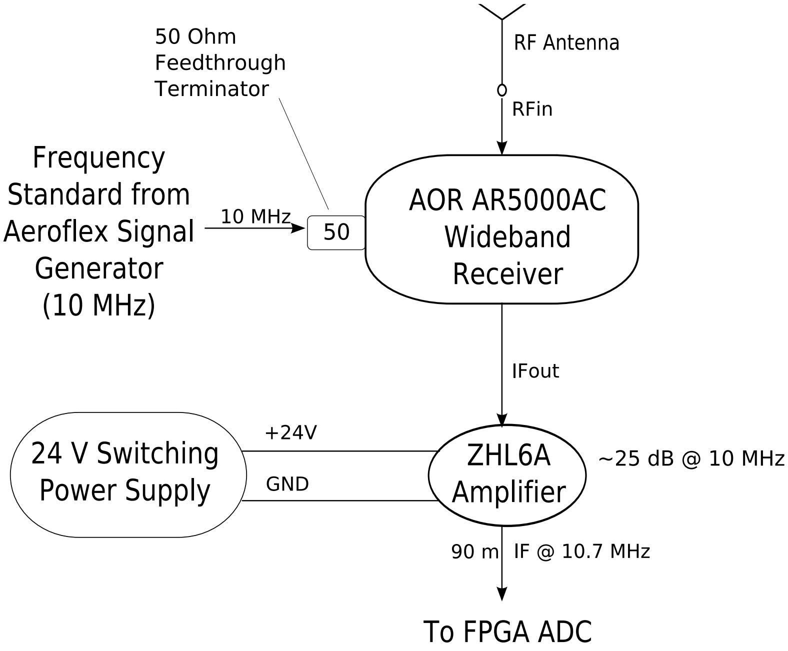
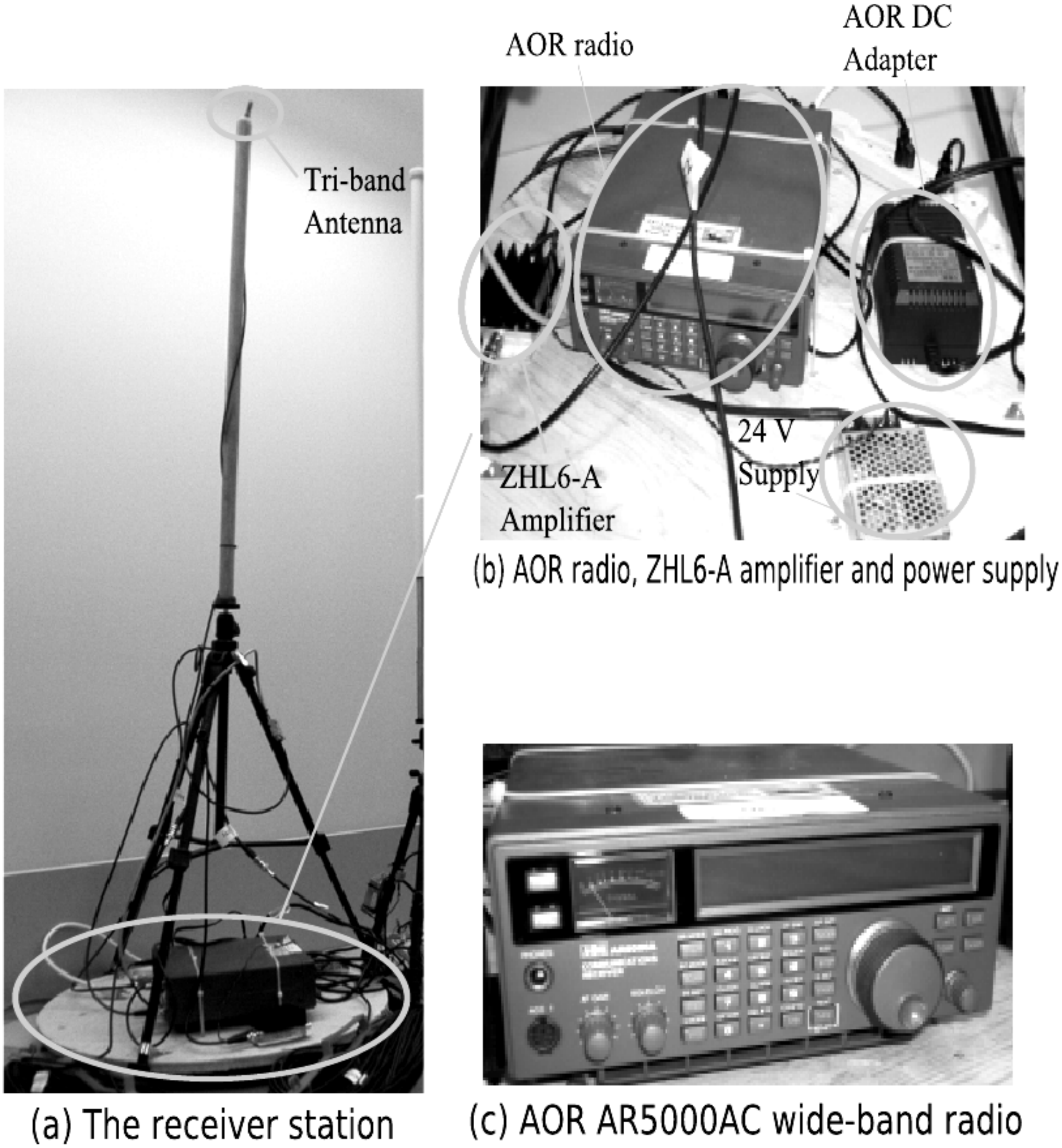


Figure 2.3: Receiver Station Block Diagram



(a) The receiver station

(b) AOR radio, ZHL6-A amplifier and power supply

(c) AOR AR5000AC wide-band radio

Figure 2.4: Receiver Station Photograph

2.1.3 Central Control

This module consists of all the devices which will effect system control and facilitate communication amongst devices on both the transmit and receive side. It consists of one Altera® Stratix® II FPGA with two ADC daughter boards, an Aeroflex signal generator, eight ZX60-6013E+ LNAs, one ZHL-6A amplifier, and two power splitters. A block diagram showing all inter-device connections is given in Figure 2.5, and a photograph is shown in Figure 2.6. The signal generator provides a high-stability 10 MHz frequency standard which is amplified and split to supply a clock signal to each radio's phase-locked loop (PLL), as well as the FPGA's PLL, in order to frequency-lock the system. The generator also supplies the LO signal which is split four ways to supply the modulator at each transmitter.

2.1.4 Hardware Encapsulation

In order to decrease coaxial cable connector wear and tear and avoid regularly interfering with individual equipment connections on a station, it was decided that each transmit and receive station would have mounted external coaxial connection points for the 90-(m) cables that connect the various devices on a given station back to the central control hub. To do this, each of several 1-(m) lengths of cable were connected to the connector of a given station device requiring direct connection to the central control hub, and the other end of the short length cable (opposite gender to first connector) is mounted on the tripod to facilitate connection to the respective 90-(m) line to/from central control. Also, a bulkhead was installed at the central control hub and used to interface the devices inside the cabinet to all transmit and receive stations. The bulkhead consists of 32 SMA female-female connectors, and four RS-232 female-female connectors. The SMA connectors are for connecting each transmitter's 90-(m) I and Q lines (from FPGA), as well as 90-(m) low-loss LO lines

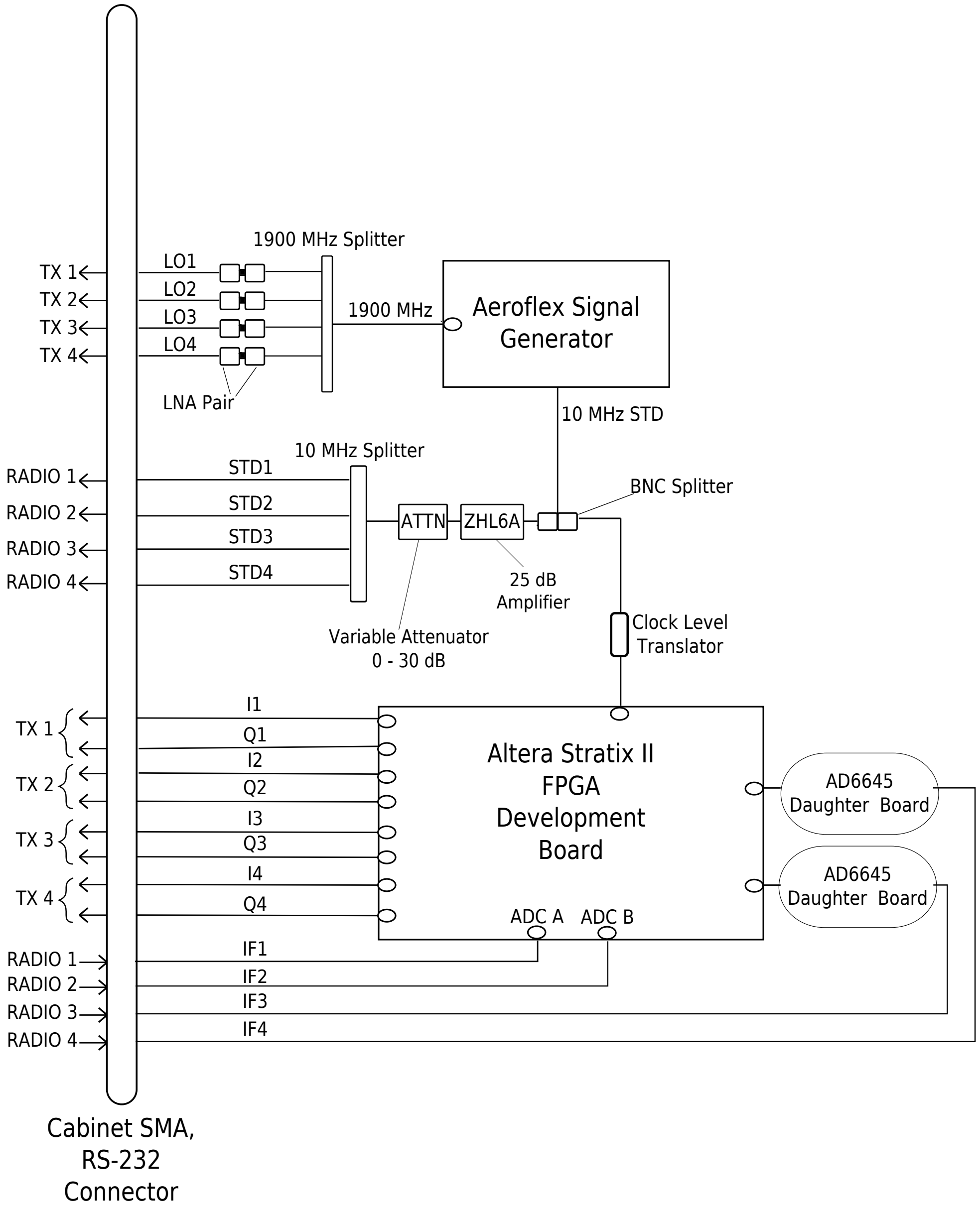


Figure 2.5: Central Station Block Diagram

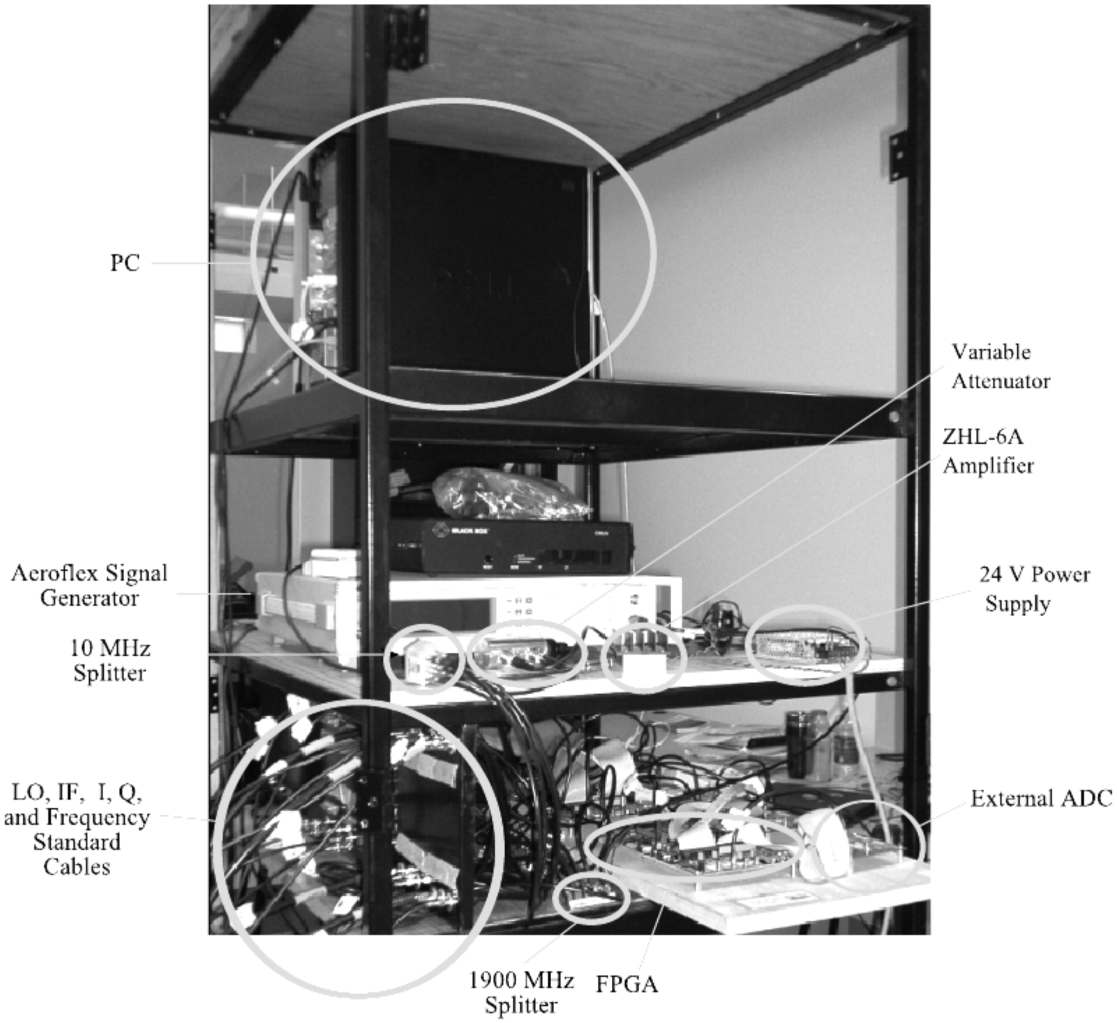


Figure 2.6: Central Station Photograph

(from signal generator) to their respective transmitter stations, as well as connecting each receiver's IF output and 10 (MHz) standard lines to their respective receivers.

2.2 Equipment for System Testing and Debugging

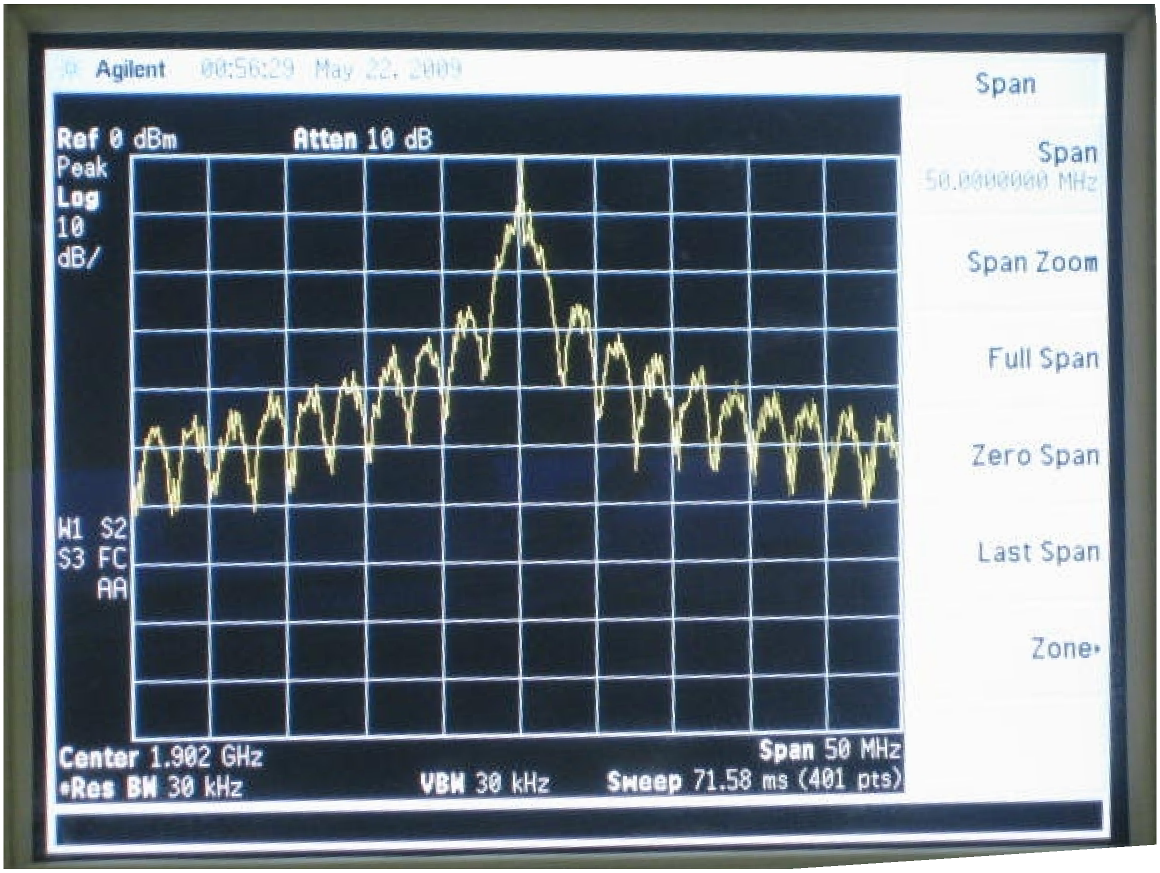
This section describes some of the equipment that was used to perform tests in effort to verify proper device and inter-device functioning, as well as to perform any necessary debugging, at the various stages of project development.

2.2.1 Tektronics® TDS2002B Oscilloscope

The Tektronics® TDS2002B oscilloscope was of significant use in the debugging process, including taking voltage level measurements at FPGA pin outputs, verifying proper clock signals at radio and FPGA external clock inputs, temporary monitoring of IF signal from radio to FPGA ADC to observe and confirm reception of modulated signal, and verifying correct FPGA signal conditioning circuit and custom bias board voltage levels before installing inter-device connections.

2.2.2 Agilent® HP E4402B Spectrum Analyser

The Agilent® HP E4402B spectrum analyser was used to carry out spectrum analysis for RF transmissions within compliance of Standard Radio System Plan (SRSP)-510 [9] and Radio Standards Specifications (RSS)-133 [10]. SRSP-510 is a document written by Industry Canada that specifies PCS minimum technical requirements for efficient utilization of the bands 1850-1915 (MHz) and 1930-1995 (MHz), while RSS-133 sets out requirements for the certification of transmitters and receivers used to provide PCS in these same two bands. The analyser is stated to have a maximum RF input power rating of +30 (dBm) (1 Watt) and a frequency sweep range of 9 (kHz)–3 (GHz), which makes it very suitable for the nature of the system measurements in this project. The RF output of the modulator is monitored on the E4402B Spectrum Analyzer, and the data, along with a screen shot, is saved on the floppy and transferred to PC for inclusion with MATLAB®analysis. An example of such an RF screen shot is given in Figure 2.7. The frequency domain of the RF waveform is a sinc function, which is expected as the baseband PN sequences in the time domain are rectangular waveforms.



Agilent 00:56:29 May 22, 2009

Ref 0 dBm Atten 10 dB

Peak

Log

10

dB/

W1 S2

S3 FC

AA

Center 1.902 GHz Span 50 MHz

•Res BW 30 kHz VBW 30 kHz Sweep 71.58 ms (401 pts)

Span 50.0000000 MHz

Figure 2.7: HP E4402B spectrum analyser RF screen shot

Chapter 3

System Implementation

In this chapter we explore the actual details of system implementation, including method of signal generation and transmission, frequency standard distribution for receiver PLLs, and the particular baseband signal processing technique that was used for signal recovery.

3.1 Signal Generation and Transmission

The baseband signals which were generated for transmission are known as *maximum length* PN sequences, and, these sequences, along with LO signals for corresponding transmit stations, are transported to each of the transmit station modulator input ports. Also, calculations were carried out on RF spectrum measurements to estimate the highest RF power and baseband signal bandwidth that could be employed while maintaining compliance with spectrum bandwidth and out-of-band emissions as per SRSP-510 [9] and RSS-133 [10].

3.1.1 Maximum Length Pseudo-Random Noise Sequences

As per Golomb [11], a sequence $\{\alpha_\mathrm{k}\}$ is a maximum length PN sequence if and only if it is a binary sequence which satisfies a linear recurrence

$$\alpha_\mathrm{k} = \sum_{i=1}^{n} \mathrm{c_i}\ \alpha_\mathrm{k-i}\ (\mathrm{mod}\ 2) \tag{3.1}$$

and has period $p = 2^n - 1$, where n is referred to as the degree of the PN sequence $\{\alpha_\mathrm{k}\}$, and the *characteristic polynomial* of the sequence

$$f(x) = 1 + \sum \mathrm{c_i}\ \mathrm{x^i}\ (\mathrm{mod}\ 2) \tag{3.2}$$

is irreducible. For $f(x)$ to be prime, it is a necessary and sufficient condition that $f(x)$ divide $1 - \mathrm{x^m}$ for $m \geq p$ only. A list of prime characteristic polynomials of 9th degree were taken from Petersen *Tables of Irreducible Polynomials over GF(2)* [1], and used as tap generator polynomials for each of the inphase and quadrature signals to each transmitter. Linear recursive sequence (LRS) generators were coded in Verilog hardware description language (HDL) and used in Quartus® II FPGA software to produce the I/Q pairs (from FPGA GPIO pins) for transmission. The LRS Verilog HDL code is given in Appendix C.

3.1.2 Transmission Data Calculations

This subsection includes calculations for estimating the minimum and maximum PN sequence lengths (i.e. number of samples) that can be transmitted using this antenna testbed based on maximum Tx-Rx antenna separation, estimated indoor root-mean-square (RMS) delay spread, number of required periods of transmitted sig-

Table 3.1: Polynomial Table [1]

Primitive Polynomial	Verilog HDL Generator Tap Equivalent [LSB,..,MSB]
$x^8 + x^7 + x^6 + x^5 + x^4 + 1$	9'b100011111
$x^8 + x^7 + x^6 + x^3 + x^2 + x + 1$	9'b111000111
$x^8 + x^7 + x^5 + x^4 + x^2 + 1$	9'b101011011
$x^8 + x^7 + x^5 + x^4 + x^3 + 1$	9'b100111011
$x^8 + x^7 + x^6 + x^5 + x^2 + 1$	9'b101001111
$x^8 + x^7 + x^6 + x^3 + x^2$	9'b010000111
$x^8 + x^7 + x^4 + x^3$	9'b000110011
$x^8 + x^7 + x^3 + 1$	9'b100100011

nal for observation and analysis, IF bandwidth, and the IF ADC sampling frequency. It is noted that the minimum number of transmit samples are governed by system multipath resolution requirements and the maximum number is ultimately limited by the coherence time [12]. For any multi-user MIMO communication system, it is important that multipath be resolved in order to track small-scale changes in the channel and facilitate good user identification performance. RMS delay spread is the difference in arrival times of the shortest and the longest of the multiple paths taken by a transmitted signal during propagation from transmitter to receiver, and the line-of-sight (LOS) component represents the shortest path between transmitter and receiver. Given that the transmitter station lines (“I”, ”Q”, and “LO”) and receiver station lines (“IF” and “STD”) are 90 (m) in length, a transmitter and receiver may be separated by a maximum line-of-sight (LOS) distance of 180 (m) as shown in Figure 3.1, which demonstrates the LOS component and the first dominant reflected signal component from transmitter to receiver using 2-dimensional (2-D) analysis. Analysis of a typical 3 × 4 × 5 right-angled triangle was extended to find the total reflected path at an LOS separation of 180 (m):

$$LOS\ at\ 5\ (\mathrm{m}) : x + y = 3 + 4 = 7\ (\mathrm{m}) \tag{3.3}$$

$$LOS\ at\ 180\ (\mathrm{m}) : x + y = 7\ \frac{180}{5} = 252\ (\mathrm{m}) \tag{3.4}$$

For the case of 180-(m) LOS separation, (3.4) indicates that there is a theoretical difference of 72 (m) between LOS and first dominant reflected signal paths, which corresponds with a time difference of 72 (m)/(3×10^8) (m/s) = 240 (ns) between LOS and reflected components per cycle. Therefore, 3 cycles will have total delay spread, σ_{total}, of approximately 720 (ns). Given that the IF is sampled at 40 (MHz), the corresponding number of Tx samples, $\mathrm{N_{min}}$, is found as follows:

$$\mathrm{N_{min}} = F_s \left(\frac{\mathrm{samples}}{\mathrm{s}}\right) \sigma_{\mathrm{total}}\ (\mathrm{s}) \tag{3.5}$$

$$= 40 \times 10^6 \left(\frac{\mathrm{samples}}{\mathrm{s}}\right) 720 \times 10^{-9}\ (\mathrm{s}) \tag{3.6}$$

$$= 29\ (\mathrm{samples}) \tag{3.7}$$

$$\tag{3.8}$$

Therefore, a minimum of $\mathrm{N_{min}}$ = 29 (samples) (or PN length of 29 bits) per period must be transmitted in order for multipath to be resolved at the maximum Tx-Rx antenna LOS separation of 180 (m).

As mentioned previously, coherence time is the upper limit on the number of transmit samples [12]. Coherence time is inversely proportional to Doppler frequency, which represents frequency and phase shifts in transmitted signals at receivers as users and/or objects move around in the path of propagation, or, more specifically:

$$\tau_{\mathrm{coh}} = K\ \frac{1}{\mathrm{f_{Doppler}}} \tag{3.9}$$

where τ_{coh} is the coherence time, K is an arbitrary factor, and $\mathrm{f}_{\mathrm{Doppler}}$ is the estimated Doppler frequency. If a typical user moves, or if an object is moving in the propagation path at an average speed, $\mathrm{v}_{\mathrm{user}}$, of 5 (km/hr), then the 1902.5 (MHz) link Doppler frequency, $\mathrm{f}_{\mathrm{Doppler}}$, can be found as

$$\frac{\mathrm{v}_{\mathrm{user}}}{c} = \frac{\mathrm{f}_{\mathrm{Doppler}}}{\mathrm{f}_{\mathrm{Carrier}}} \tag{3.10}$$

$$\mathrm{f}_{\mathrm{Doppler}} = \mathrm{f}_{\mathrm{Carrier}} \, \frac{\mathrm{v}_{\mathrm{user}}}{c} \tag{3.11}$$

$$= 1902.5 \times 10^6 \, \frac{5 \left(\frac{\mathrm{km}}{\mathrm{hr}}\right) 10^3 \left(\frac{\mathrm{m}}{\mathrm{km}}\right) \frac{1}{3600} \left(\frac{\mathrm{hr}}{\mathrm{s}}\right)}{3 \times 10^8 \left(\frac{\mathrm{m}}{\mathrm{s}}\right)} \tag{3.12}$$

$$= 8.81 \text{ (Hz)} \tag{3.13}$$

Using (3.9) with $K = \frac{1}{10}$ and $\mathrm{f}_{\mathrm{Doppler}} = 8.81$ (Hz) yields $\tau_{\mathrm{coh}} \approx 11$ (ms). The chip length, $\mathrm{T_c}$, of each baseband signal at frequency, B_s, of 2.5 (MHz) is found to be

$$\begin{aligned} \mathrm{T_c} &= \frac{1}{B_s} \\ &= \frac{1}{2.5 \times 10^6} \\ &= 400 \text{ (ns)} \end{aligned}$$

and the condition for total number of coherent symbols, $\mathrm{N}_{\mathrm{coh}}$, arriving within coherence time window, τ_{coh}, is such that

$$\mathrm{N}_{\mathrm{coh}} \; \mathrm{T_c} < \tau_{\mathrm{coh}} \tag{3.14}$$

Substituting the above calculated values for $\mathrm{T_c}$ and τ_{coh} in (3.14)

$$\mathrm{N_{coh}}\ (400 \times 10^{-9}) \left(\frac{\mathrm{s}}{\mathrm{sample}}\right) < 11 \times 10^{-3}\ (\mathrm{s}) \tag{3.15}$$

$$\Rightarrow \mathrm{N_{coh}} < 25000\ (\mathrm{samples}) \tag{3.16}$$

$$\Rightarrow \mathrm{N_{coh}} < 16384\ (\mathrm{samples}) \tag{3.17}$$

$$\Rightarrow \mathrm{N_{coh}} = 2^{14}\ (\mathrm{samples}) \tag{3.18}$$

which corresponds to a maximum PN sequence length, $\mathrm{N_{max}}$, of 2^{14} bits or LRS generator tap polynomials of no more than 14 bits wide. In summary, the shortest PN code that can be utilized while maintaining multipath resolution capability is approximately 2^5 bits and the longest code that can be used for while maintaining data capture in the estimated coherence timing window is 2^{14} bits. However, it should be noted that while these code length estimates indicate absolute minimum and maximum values, they are still only theoretical as they do not take device speed and memory limitations into account, among other factors, especially for the case of the upper limit. For the select configuration in this thesis, the true upper limit in PN sequence length was found to be 2^9 bits, which will be explained later in Section 3.3.

3.1.3 Spectrum Simulation and Analysis

As previously mentioned, this system is currently configured for RF operation using the generously-loaned Bell Aliant 1900 (MHz) spectrum in the Personal Communications Services (PCS) C2 block as per the band plan in SRSP-510 [9]. The lower and upper subbands in this band plan are separated by 80 (MHz), and this thesis uses the 1900–1905 (MHz) (lower) and 1980–1985 (MHz) (upper) subbands for RF transmission. For all subsequently presented test results and calculations in this thesis, the 1902.5 (MHz) (centre of lower subband) carrier was used as the main RF link. Given that the modulator accepts both I and Q signals for baseband input, the

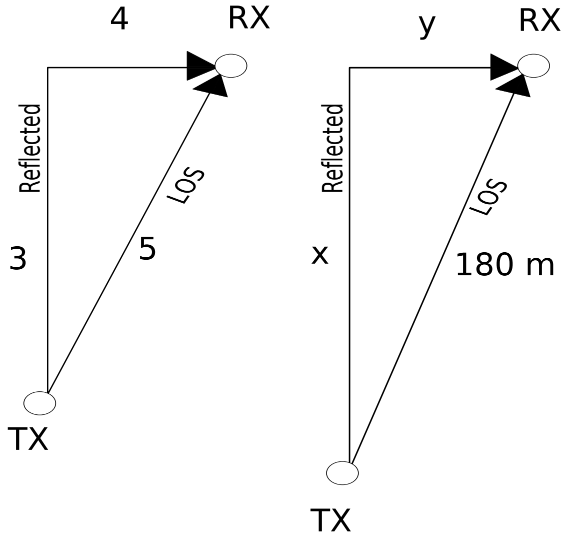


Figure 3.1: 2-D Plane Analysis of Tx-Rx Multipath with Reflected Signal at 180 (m) LOS

I and Q signals were each chosen to be 2.5 (MHz), so as to have the spectrum limited to either of the 5 (MHz) lower or upper subbands of the C2 block. In addition, further analysis had to be conducted in order to find the maximum allowed transmit power per subband in compliance with SRSP-510 and RSS-133. According to section 2.3 of RSS-133, bandwidth is measured as the width of the signal between two points about the carrier, with one point above the carrier and one point below, such that all signals outside those points are attenuated at least 20 (dB) below transmit power (-20 dBc). In addition to this definition of bandwidth, out of block emissions also had to be considered. As per section 6.5.1 (a) (i) in RSS-133, in the first 1 (MHz) bands immediately outside and adjacent to the equipment's operating frequency block, the power of emissions per any 1 % of the emission bandwidth shall be attenuated below the transmitter output power P (in Watts) by at least $43 + 10\log_{10} P$ (dB). MATLAB®simulation source code was developed for this analysis (see Section B.1), and, given the above constraints in RSS-133, the maximum allowed transmitter output power was found to be +17.53 (dBm). Given this finding, and that the Hittite modulators were each outputting between 0 (dBm) and 1 (dBm) RF power for the current setup, it was decided to use an amplifier on each RF output with a nominal gain of approximately 15 (dBm) at 1900 (MHz), which made the ZX60-6013E+ amplifier a very suitable choice for the purpose.

3.1.4 Transmit Amplifiers and Local Oscillator Distribution

At the modulator, the LO input power must be in the range ± 6 (dBm). Using regular RG-58 Belden cable to send the LO signal to a transmitter would result in significant attenuation (see Table 3.2), and a very high power amplifier would have to be used to adjust the LO power to the required input range. The cable used was the Jefa Tech Low Loss 400 50 Ohm cable assembly, a close alternative to Belden LMR-400, with only slight performance improvement over the Belden LMR-400 as indicated in the Low Loss 400 data sheet [13]. Given that the system RF is 1902.5 (MHz) (or 1982.5 (MHz) for forward link), the nearest stated attenuation factors at 2000 (MHz) were used in calculating the total attenuation along the 90-(m) cable runs in Table 3.2.

The Low Loss 400 is stated as having a total attenuation of approximately 18 dB, but this is loss due to cable only. Connector losses at splitter, bulkhead, and transmit stations also contribute to total attenuation in signal as it propagates from source generator output to modulator input. The total signal attenuation at 1902.5 MHz was found to be 26.28 (dB) (see Section A.1 for specific test conditions and more test frequencies). To overcome this attenuation, a pair of ZX60-6013E+ LNAs are used after each split, each theoretically giving 15 (dB) amplification for a total of 30 (dB) per LO signal. Actual experiment showed the total amplification to be closer to 28 (dB) per LO signal, a decrease in gain which may have been a result of the second amplifier operating at compression, but the resulting distortion would not be a significant issue for the LO signal so this occurrence is still acceptable. Now, with 28 (dB) of gain supplied by the amplifier pair and a source generator RF level setting of +3 (dBm), the resulting received signal strength of approximately 1.7 (dBm) was observed at the modulator input, thus achieving the required modulator input range on ±6 (dBm). Also, each amplifier requires a 12 (V) power supply. As each 12 (V) power supply could support up to four amplifiers, two (2) power supplies were used at the central processing station to power all four amplifier pairs in preference to installing a power supply at each of four transmit stations.

Table 3.2: Theoretical Coaxial Cable Attenuation

Cable Type	Attenuation at 2 GHz (dB/30 m)	Total Attenuation over 90 m(dB)
Belden RG58 U	24.95	74.85
Low Loss 400	5.97	17.91

3.2 Receivers

In this section, the system frequency standard distribution is described in more detail, the radio configuration is given, and the chosen baseband processing technique is explained with the aid of mathematical analysis and graphical demonstrations.

3.2.1 Frequency Standard Distribution

In a MIMO system where the transmitters and receivers are decentralized, there is need for what is known as pilot symbols [14, 15, 16]. These are reference symbols which are transmitted with signal phase information and used by the receivers to track the phase of incoming RF signals for use in successful demodulation, but at the expense of decreased spectral efficiency due to the associated pilot transmission overhead. However, since the intent was to implement the testbed as a centralised communication system (single clock source common to all transmitters and receivers), a high-stability frequency standard (less than 1 (ppm)) of 10 (MHz) was generated from the Aeroflex signal generator and distributed in effort to make the frequency standard common to all transmitters (FPGA I/Q baseband data clock) and receivers (AOR radio external frequency standard inputs). Figure 3.2 below demonstrates the frequency standard distribution. The receivers are spaced 90 (m) apart to take advantage of SWAP gain [5] [17]. However, this spacing means that the frequency standard being generated from the signal generator must be split five ways to supply the FPGA and four radios with clock signals. Supplying clock to the FPGA is simple in that the signal generator and FPGA are close together inside the cabinet. However, as the radios are each 90 (m) away, this means splitting that same clock signal four more ways and sending it down each of four 90-(m) RG-58 coaxial cables, which causes significant attenuation in the original signal. The previous developer had indeed found that it was necessary to use an amplifier before splitting the clock signal for the radios.

As a result, a Mini – Circuits® ZHL-6A 25 (dB) amplifier, with 24 (V) power supply, was used in conjunction with a Mini – Circuits® 4-way splitter to supply each radio with a clock signal with enough amplitude to lock each radio's PLL during operation, as also demonstrated in Figure 3.2.

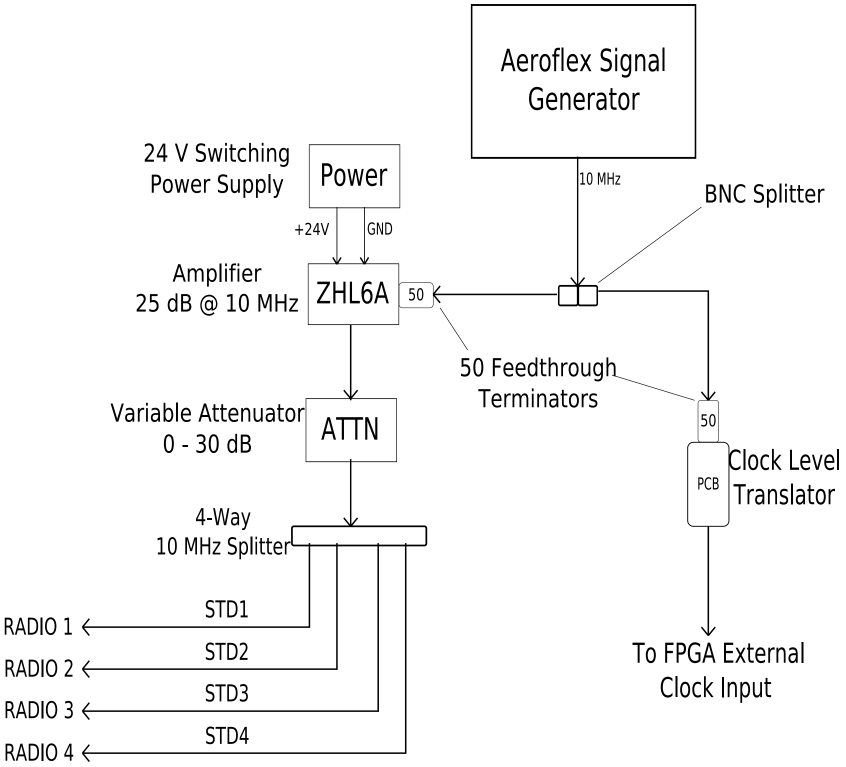


Figure 3.2: High-Stability Frequency Standard Distribution

3.2.2 AOR AR5000A Receiver Configuration

Figure 3.3 is a photograph of one of four AR5000A Wideband radio receivers used for receiving the RF signals. Each receiver downconverts received modulated RF signals down to an intermediate frequency stage of 10.7 (MHz)± 5 (MHz). This downconversion allows the signal to be sampled by an ADC at a lower speed, the minimum being the Nyquist sampling rate at 21.4 (MSPS) for sampling the signal centred at 10.7 (MHz). However, the analog bandwidth into the ADC has to span the whole frequency range, thus the minimum ADC sampling rate must be 15.7 (MHz) × 2 = 31.4 (MHz).

Each receiver was given the same operating configuration as outlined in Table 3.3 below, and a block diagram of the AOR receiver is given in Appendix A. Examination of the AOR block diagram shown in Section A.2 reveals that the IF output is taken after the second stage but before the third, and, as per the block diagram, no AGC is applied to this second stage, indicating that the radios only downconvert and filter

Figure 3.3: AOR AR5000A Wideband Receiver

the received RF signals to IF. Each radio also has an option for clock source, with choice between the internal 12.8 (MHz) PLL, and external frequency standard. As the decision was made to frequency lock the receivers and transmitters to a high-stability frequency standard, the external standard option is selected. The IF filter bandwidth setting ranges from 6 (kHz)–220 (kHz), and outputs filtered signals to the audio output port. Typically, a narrower filter bandwidth setting (e.g. 6 (kHz)) is chosen for narrowband signals, and a wider filter BW setting (e.g. 110 (kHz) or 220 (kHz)) is used for wideband signals. For this thesis, however, the monitored IF signal is taken before the filter stage, so changing the filter settings has no significant effect on the resulting IF output. A filter setting of 15 (kHz) was arbitrarily chosen. The RF is tuned to 1902.5 (MHz), and "*ANT 1*" selected to accept RF signal from the receiver antenna. The radio has five variable frequency oscillators (VFO) selection options, each with a different purpose as stated in the radio operating manual [2].

Table 3.3: AOR AR5000A Configuration

Configuration Parameter	Setting
Clock	FRQ STD EXT
Tune Freq	1902.5 MHz
Mode	FM
IF BW	15 KHz
Antenna	ANT1
VFO	A

3.3 Baseband Signal Processing

In this section, the chosen method of demodulation is described and demonstrated via mathematical analysis and graphical aid. Each sampled IF signal is processed to extract complex baseband signal in the form $\mathrm{I(n)} + \mathrm{jQ(n)}$, which are then convolved with *matched filters* constructed from transmit I/Q to yield the sixteen (16) channel impulses that result from a 4-transmitter, 4-receiver configuration.

3.3.1 IF Demodulation

Each IF signal is sampled as shown in Figure 3.4, which is SignalTap® II screen shot zoomed in to show one period of baseband I and Q transmit signals. The IF signal from Radio 1, or “IF 1”, is “ext_adc1_data”, “IF 2” is “ext_adc2_data”, “IF 3” is “onboard_adc1_data”, and “IF 4” is “onboard_adc2_data”. The PN sequences generated at the time of transmission are also recorded as “i1_tx”, “q1_tx”, . . . , “i4_tx”, and “q4_tx”. Note that each of the two vertical time bars mark the start of a new period, and that there is 8176 samples between them. This is because the SignalTap® II clock is at 40 (MHz) and the baseband I and Q signals are at 2.5 (MHz), thus upsampling the displayed I/Q by a factor of 40 (MHz)/2.5 (MHz) = 16. Therefore, dividing the given number of samples by this factor of 16 yields 8176/16 = 511 samples, or 2^9, which is the actual length of each PN sequence. This coincides with the maximum length PN sequence requirement for the 9-bit LRS generator tap polynomials.

A diagram depicting the utilized IF demodulation technique is given in Figure 3.5. As previously mentioned, the IF signal is sampled at the FPGA ADC at a sampling frequency, $\mathrm{f}_{\mathrm{sadc}} = 40$ (MHz) so as to satisfy the Nyquist sampling requirements of the IF signal. The output of the DC remove block is the difference between the signal amplitude at each sample point and the mean amplitude value. The signal is then passed through a bandpass filter, which filters the IF for the expected 5.7 (MHz)–15.7 (MHz) range, and split two ways. One path is multiplied with a sinusoid $\sqrt{2}\ \cos(2\ \pi\ \mathrm{f}_{\mathrm{IF}}\ n + \phi_{\mathrm{est}})$, the other path is multiplied with another sinusoid $-\sqrt{2}\ \sin(2\ \pi\ \mathrm{f}_{\mathrm{IF}}\ n + \phi_{\mathrm{est}})$, and both resultant signals are downsampled to baseband to yield the estimated baseband received inphase, $\hat{\mathrm{i}}_{\mathrm{bb}}$, and quadrature, $\hat{\mathrm{q}}_{\mathrm{bb}}$, signals, respectively. These demodulating sinusoids are at the IF frequency, f_{IF}. The value of ϕ_{est}, after previous constellation analysis of received I and Q signals for demodulator clock adjustment was found to be approximately +23° for an RF transmission test with one transmitter and one receiver. However, it should be noted that demodulator

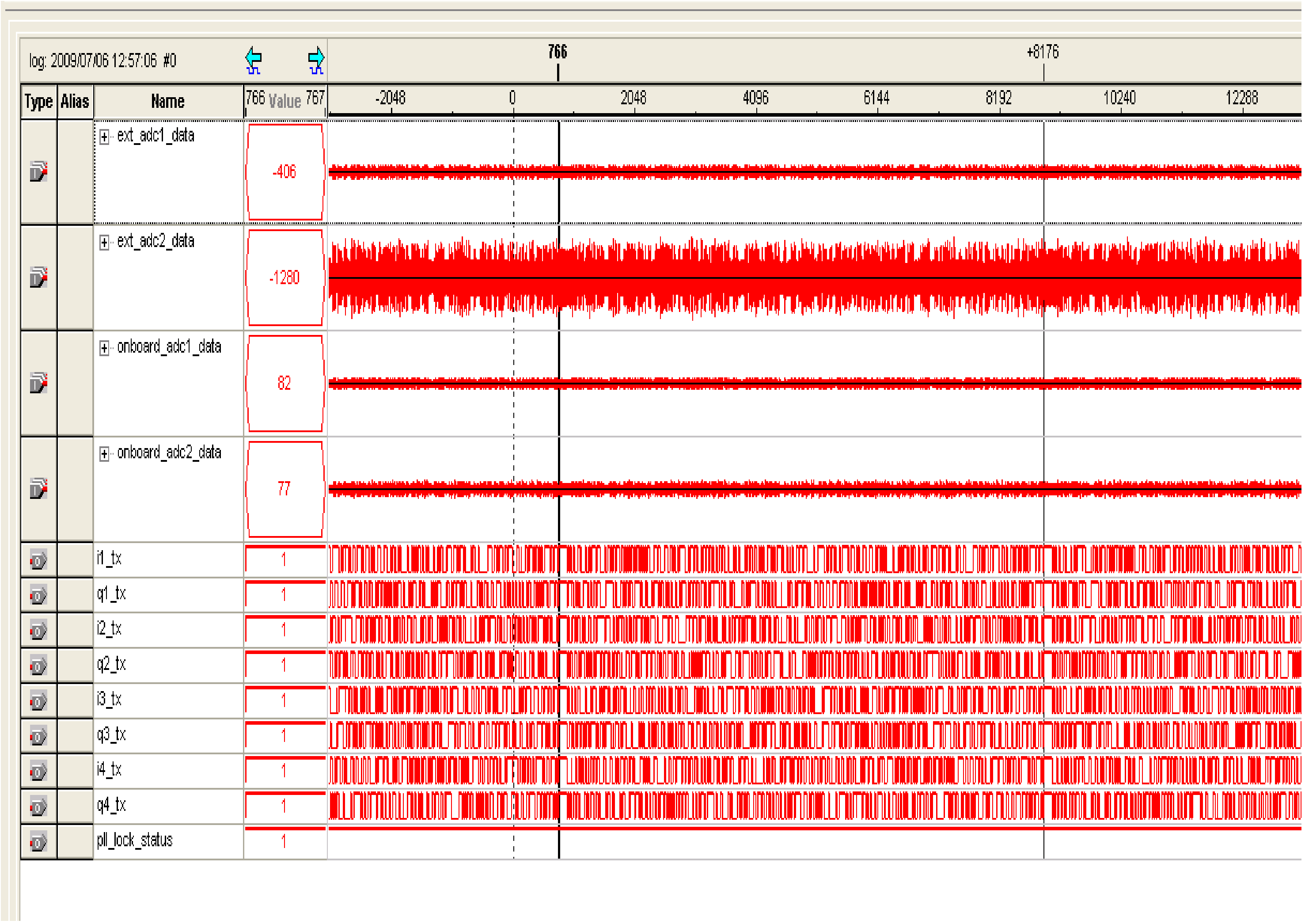


Figure 3.4: IF signals with I/Q Sequence Generation and Store

phase-tracking is not necessary at this stage as the system is frequency-locked and a phase mismatch here, caused by propagation delay of RF from transmitter to receiver, merely results in a rotation of the demodulated I/Q constellation, which has no real effect on observation of the matched sliding correlator outputs. The quadrature signal estimate is then made imaginary and combined with the inphase signal estimate to yield the total complex received signal estimate $\hat{\mathrm{s}}_{\mathrm{bb}}(n) = \hat{\mathrm{i}}_{\mathrm{bb}}(n) + j\hat{\mathrm{q}}_{\mathrm{bb}}(n)$.

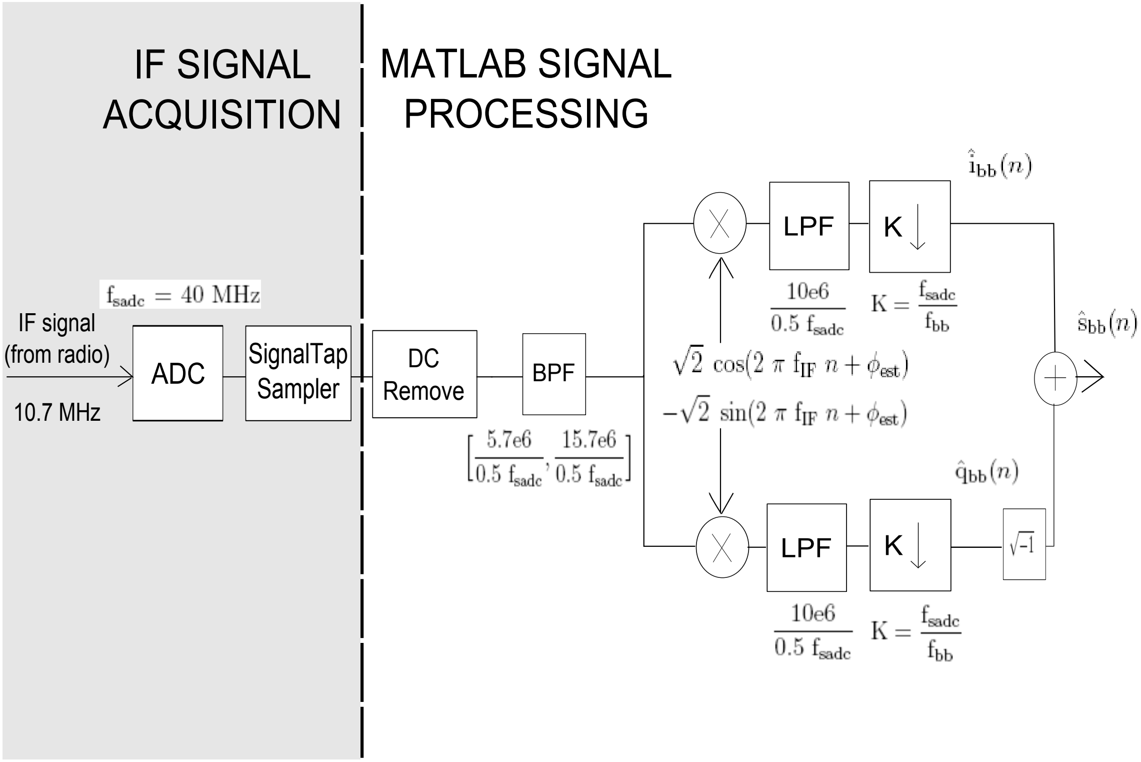

IF SIGNAL ACQUISITION
MATLAB SIGNAL PROCESSING
$\mathrm{f}_{\mathrm{sadc}} = 40$ MHz
IF signal (from radio)
10.7 MHz
ADC
SignalTap Sampler
DC Remove
BPF
$\left[\frac{5.7e6}{0.5\ \mathrm{f}_{\mathrm{sadc}}}, \frac{15.7e6}{0.5\ \mathrm{f}_{\mathrm{sadc}}}\right]$
$\sqrt{2}\ \cos(2\ \pi\ \mathrm{f}_{\mathrm{IF}}\ n + \phi_{\mathrm{est}})$
$-\sqrt{2}\ \sin(2\ \pi\ \mathrm{f}_{\mathrm{IF}}\ n + \phi_{\mathrm{est}})$
LPF
K ↓
$\frac{10e6}{0.5\ \mathrm{f}_{\mathrm{sadc}}}$
$\mathrm{K} = \frac{\mathrm{f}_{\mathrm{sadc}}}{\mathrm{f}_{\mathrm{bb}}}$
$\hat{\mathrm{i}}_{\mathrm{bb}}(n)$
LPF
K ↓
$\frac{10e6}{0.5\ \mathrm{f}_{\mathrm{sadc}}}$
$\mathrm{K} = \frac{\mathrm{f}_{\mathrm{sadc}}}{\mathrm{f}_{\mathrm{bb}}}$
$\hat{\mathrm{q}}_{\mathrm{bb}}(n)$
$\sqrt{-1}$
$\hat{\mathrm{s}}_{\mathrm{bb}}(n)$

Figure 3.5: IF Demodulation

3.3.2 Matched Filter Sliding Correlators

A bank of four *matched-filter* sliding correlators are applied to each of the four received signals in order to implicitly obtain the sixteen (16) channel impulse responses. A matched filter is used for optimum detection of a known signal. Proakis text [18] gives this description of a matched filter.

A filter whose impulse response $h(t) = s(T - t)$, where $s(t)$ is assumed to be confined to the time interval $0 \leqslant t \leqslant T$, is called the *matched filter* to the signal $s(t)$. The response of $h(t) = s(T - t)$ to the signal $s(t)$ is

$$y(t) = \int_0^t s(\tau)s(T - t + \tau)\ \mathrm{d}\tau \tag{3.19}$$

It can be seen from (3.19) above that $y(t)$ is essentially a time-autocorrelation function of $s(t)$, and that $y(t)$ is an even function of t, which attains a peak at $t = T$. Therefore, the correlation of M periods of complex n^{th} degree maximum length PN sequences will have a peak at each value $k(2^n - 1)$ for $k = 1, 2, \ldots, 2M$, with the maximum peak at $k = M$. For this thesis, impulse response points, h_{ij}, were recorded for six points about the centre of the matched filter outputs, $k = M$, for the i^{th} receiver and j^{th} transmitter. Figure 3.6 shows four matched filters, $\mathrm{y_j}^*(N - n)$, being applied to the i^{th} received baseband signal estimate, $\hat{\mathrm{s}}_{\mathrm{i}}(n)$, to yield the baseband impulse responses $\mathrm{h}_{\mathrm{ij}}(n)$ for $j = 1, 2, 3, 4$. N is the total number of transmission bits recorded and observed at the received end per RF transmission. For the case where each original baseband PN sequence pair for the j^{th} transmitter, $\mathrm{i_j}(n)$ and $\mathrm{q_j}(n)$, is chosen to be 511 (or $2^9 - 1$) bits long and $M = 4$ periods are stored, $\mathrm{N} = 2044$ bits. Note that $\mathrm{y_j}(n) = \mathrm{i_j}(n) + j\mathrm{q_j}(n)$. An explanation is given in subsection 3.3.3 on deriving the actual upper limit on the number of bits used for each transmission.

3.3.3 System Issues and Limitations

During system development, several important issues were encountered and addressed, with the main issue being the actual limit on the number of bits that can be used for each transmission. It was found that the transmission bit limit depended on the chosen method of baseband signal processing, with the option of implementing online or offline baseband signal processing. Online baseband signal processing

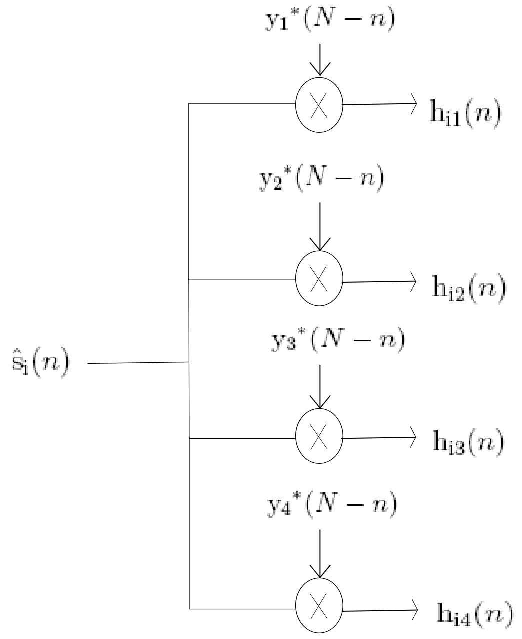


Figure 3.6: Generating Impulse Responses with Matched Filters

involves using the FPGA software, Quartus® II, to process the received IF using both Altera® intellectual property (IP) mega-functions and user-defined Verilog HDL functions, while offline processing requires using MATLAB®scripts to process the IF signals sampled at the FPGA ADCs. The advantage of implementing online processing is such that real-time observation of received signal estimates $\hat{i}_{bb}(n)$ and $\hat{q}_{bb}(n)$ are enabled via SignalTap® II logic analyzer, but at the expense of significant added memory overhead in SignalTap® II due to usage of the IP mega-functions, which, for the 4 × 4 case, includes 8 finite impulse response (FIR) filters (one for each I and Q estimate), four numerically-controlled oscillators (to generate demodulator sinusoids for each I/Q pair), and a PLL which multiplies/divides the 10 (MHz) standard at the FPGA external clock input to supply clock signals to the ADCs, NCOs, FIRs, and LRS generators. For the Altera® Stratix® II EPS21080 DSP development kit, the maximum number of stored samples in SignalTap® II is 128 K, but including all the above mega-functions in Quartus® II and monitoring the four incoming IF signals with the four I/Q PN sequence pairs reduces that number to 8 K, which significantly reduces the number of transmission bits that can be recorded and stored. Since online processing requires the use of the IP mega-functions, the decision was made to use offline processing in MATLAB®, only using a PLL to provide clocks to the FPGA ADCs and user-defined LRS generators in Verilog HDL to generate the I/Q sequences in Quartus® II, and using SignalTap® II to observe only the four sampled IF signals and four I/Q sequence pairs. With this approach, the maximum number of available storage samples in SignalTap® II increased to 32 K. It was found that increasing the sampling frequency results in less available storage samples for observation, and so, in order to maximize number of storage samples, the FPGA ADC and SignalTap® II sampling frequencies are chosen to be 40 (MHz), which is a convenient multiple of 2.5 above the Nyquist rate of 31.4 (MHz). Now with the SignalTap® II clock operating at 40 (MHz), when it samples the original I/Q baseband sequence, which is at 2.5 (MHz), the sequence is up-sampled by a factor of 16. Therefore, for 32 K (2^{15}) available samples, this results in the condition that the number of recorded samples per transmission, N_{tx}, from the baseband point of view, must satisfy

$$\begin{aligned} \# \; of \;\; samples \; per \; transmission &\leq \# \; available \; samples/upsample \; factor \quad (3.20)\\ &\leq 2^{15}/16 \quad (3.21)\\ &\leq 2048(samples/period) \quad (3.22) \end{aligned}$$

Therefore, all the baseband PN sequences combined must be no longer than 2048 bits per RF transmission. The desire, however, was to utilise the periodic nature of these maximal length sequences in order to observe multiple complex baseband impulses from received and processed IF signals, then to select several points about the central response. As a result, the decision was made to record a minimum of three periods per transmission. The 2048-sample block is conveniently split into four smaller blocks of 512 bits each, thus resulting in each maximum length PN sequence having 511 bits per period. Given this finding, LRS generators were developed using Verilog HDL in Quartus® II using 9-bit tap polynomials as indicated back in subsection 3.1.1.

Chapter 4

System Measurements and Results

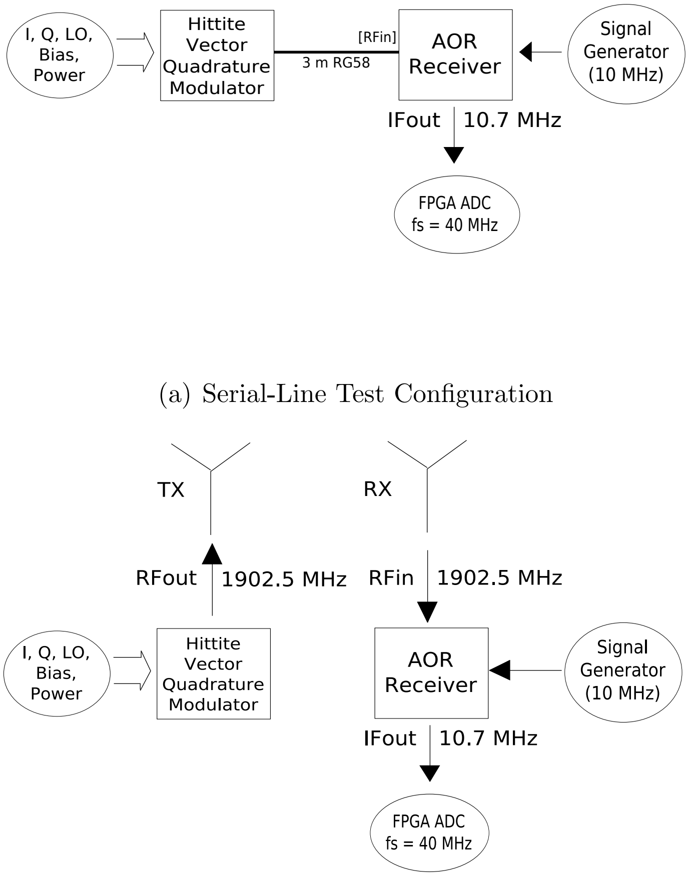
This chapter describes the various testing/debugging configurations and presents an MMSE comparison for two different indoor antenna layouts, and three different outdoor antenna layouts.

4.1 Testing Configurations

In effort to debug wireless transmission for multiple parallel channels, it was decided to do serial tests with each transmitter and receiver to verify operation before implementing the full 4×4 implementation. In addition, the channel impulse sets obtained from the serial configurations would serve as the "ideal" case upon which to compare the channel impulses for each channel in the 4×4 wireless case.

4.1.1 Serial-Line and Serial-Wireless Configurations

For the serial-line configuration, only one transmitter and one receiver was enabled. The transmitter and receiver are connected using a 3-(m) SMA Male N-Type Male connector terminated cable between the RF output of the modulator at transmitter and the radio receiver's antenna input (see Figure 4.1(a)), while for the serial-wireless configuration, the RF output of the modulator at transmitter and the radio receiver's antenna input were each connected to an antenna for operation in either the 1900 MHz PCS C2 band or ISM 2400 MHz band at a separation of 3 (m) (see Figure 4.1(b)). The received signal was processed to yield the channel impulse response which is then compared with the impulse response obtained from corresponding serial-line test for similarity, as well as the parallel-wireless impulse responses obtained for subsequent M × N (M transmitters and N receivers) testing configurations.



(a) Serial-Line Test Configuration

(b) Serial-Wireless Test Configuration

Figure 4.1: Serial Test Configurations (a) Serial-Line, (b) Serial-Wireless

4.2 Antenna Layouts with Corresponding MMSE Approximations

The transmit and receive antennas were set out for two different indoor layout plans and three different outdoor layout plans. For the indoor layouts, “Indoor Layout 1” is used for indoor analysis of 4 × 4 arrangement with close LOS antenna spacing while “Indoor Layout 2” is used for indoor analysis of 4 × 4 case with increased antenna spacing and scatterers (e.g. windows, metal objects) in the paths of signal propagation. In addition to these 4 × 4 cases, 3 × 4 tests were carried out in order to investigate the effect of reducing the *rank* of the channel matrix for both indoor layouts. As per Gesbert et. al [3], the rank of the channel matrix is defined as the number of independent equations to be solved for in the obtained channel matrix, or more clearly, min(N_{tx},N_{rx}), where N_{tx} is the number of transmitters and N_{rx} is the number of receivers that were active for the measured transmission. Hence, the rank for a 4 × 4 case would be 4, and the rank for a 3 × 4 case is 3. This reduction in rank means that there will be less independent equations to solve for in the obtained channel matrix, thus decreasing matrix complexity and making it easier to invert the channel at the receiver. Also, using more receivers than transmitters has the effect of increasing spatial diversity, thus enhancing received signal quality. As a result, it was expected that the 3 × 4 cases would provide improved MMSE performance over the full-rank 4 × 4 cases, and tests were subsequently carried out to confirm this expectation.

For the outdoor layouts, “Outdoor Layout 1” is used for outdoor analysis for the 4 × 4 case and a 2 × 2 case with close LOS antenna spacing, “Outdoor Layout 2” is used for outdoor analysis for the 4 × 4 case and a 2 × 2 case with increased antenna spacing and scatterers (e.g. trees, buildings) in the paths of signal propagation, and “Outdoor Layout 3” is used for outdoor analysis for a 2 × 2 case with further increase in antenna separation and change in transmitter and receiver antenna arrangement. The channel impulse responses obtained from the matched sliding correlator outputs were put in an LMS adaptive filter simulation, using six points about the centre of each impulse response, in order to examine user learning curve performance and average MMSE for all users. The purpose of the simulation was to the evaluate the channel matrix and use this evaluation to determine if one could build a communications system with that matrix present. The source code for the LMS adaptive filter simulation is given in Section B.3, and a high-level block diagram illustrating how the measured channel matrices are placed in the simulation model is given in Figure 4.2.

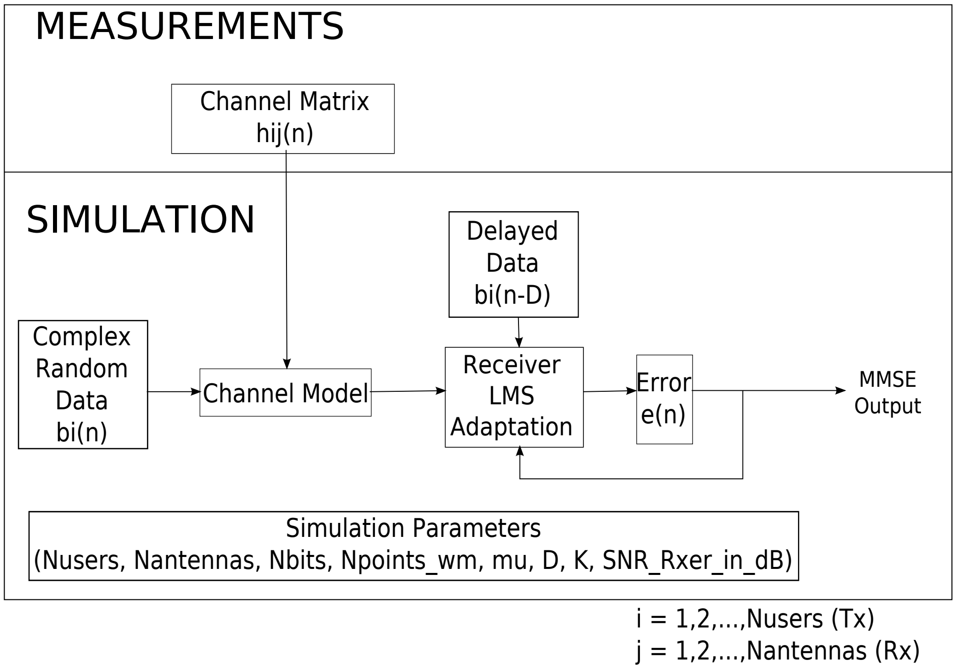


Figure 4.2: LMS Adaptive Filter Simulation with Measured Channel Information

The simulation uses the LMS algorithm:

$$\mathrm{w}_{\mathrm{n+1}} = \mathrm{w}_{\mathrm{n}} + \mu \ \mathrm{e}_{\mathrm{n}} \ \mathrm{r}_{\mathrm{n}} \tag{4.1}$$

where $\mathrm{w_n}$ is the adaptive filter tap weight, $\mathrm{e_n}$ is the filter error estimate between filter training data $\mathrm{d_n}$ and filter estimate $\hat{\mathrm{d}}_\mathrm{n}$, $\mathrm{r_n}$ is the received signal, and μ is an adaptation constant used to adjust the adaptation step-size. In this simulation, the channel impulse response matrix obtained from each $\mathrm{M - transmitter \times N - receiver}$ configuration is entered in place of the simulation channel matrix, and the simulation parameters are adjusted for optimum convergence. Foschini et. al [19] have shown that, for coherent detection of QAM symbols, the bit-error rate (BER) is upper-bounded by MSE

$$\mathrm{P_e} \leq \; 2\; exp\Big\{ -\frac{1}{MSE}\left[1-\frac{MSE}{2\;\sigma^2(L)}\right]^2\Big\} \tag{4.2}$$

$$\approx \; 2\; exp\Big\{ -\frac{1}{MSE}\Big\},\; \mathrm{N_o} \;\rightarrow\; 0 \tag{4.3}$$

where $\mathrm{P_e}$ is the probability or bit error, $\mathrm{N_o}$ is the noise spectral density, σ^2 is the variance of the transmitted QAM symbols, and L is the maximum number of data levels assumed by the transmitted QAM symbols. The MSE in this sense, is the optimal MSE obtained from selecting of an adaptively-developed optimal front-end filter(with complex impulse response), and an appropriate canceller. Given the above finding for the relationship between BER and MSE, the average user (transmitter) MMSE value, obtained as simulation output after placing the measured channel matrix into the simulation, was used as the basis of comparison for channel characteristics in the various antenna layouts. For 4 × 4 MMSE analysis in the following subsections, the filter parameters that gave optimum convergence are given in Table 4.1, and for 2 × 2 MMSE analysis, the same parameters are used with the exception of “Nusers” and “Nantennas” both being a value of 2. A value of 1 for the spreading factor, K, means no spread-spectrum is employed and flat-fading is assumed. The number of bits for transmission, Nbits, is at a high value of 2^{18} bits, which means a longer training time for the filters, but better convergence results than for a lower value. The optimum filter training data decoding delay, D, was found to be 15 samples, and the step-size, μ, that gave best simulation convergence was found to be 2^{-9}.

Table 4.1: Adaptive Filter Simulation Parameters

Parameter	Description	Value
Nusers	Number of active transmitters	4
Nantennas	Number of active receivers	4
μ	Adaptation constant	2^{-9}
K	Spreading Factor	1
Nbits	Number of transmission bits used for simulation	2^{18}
D	Filter training data decoding delay	15

4.2.1 Indoor Layout 1 (IL1)

Figure 4.3 demonstrates the antenna setup for the chosen configuration in which the antennas are considered close to each other, and Table 4.2 shows the antenna separation distances in meters. Four periods of transmitted signals are captured and sampled at each FPGA ADC, via IF, and processed to yield baseband data and channel impulse responses. The channel impulse responses are then placed into the LMS adaptive filter simulation to yield user error curves and thus gauge system performance. After signal processing and simulation run, the mean MMSE for all users in this layout, MMSE_{IL1}, was found to be -3.55 (dB). Figure 4.4 shows the curves from top to bottom for User 1 to User 4, respectively. The term "User" here represents a particular transmitter. User 1 performance is very weak at around -1.50 (dB), indicating a possible saturation at the receivers, thus not being able to detect User 1's signal due to the presence of four different transmit signals in such close proximity. Users 2 and 3 are slightly better at -4.30 (dB) and -3.40 (dB) respectively, and User 4 is acceptable at -5.00 (dB) . These results seem to coincide with the finding by Zhu et al. [20] that the channel matrix is usually very hard to invert for close LOS antenna spacing.

Given that User 1 could not be identified from the above 4 × 4 case, a 3 × 4 test case was carried out in which transmit 1 was disabled, leaving all the other transmitters and receivers active. The obtained error curves are given in Figure 4.5, and the mean MMSE for all users, $\text{MMSE}_{\text{IL1}_{3x4}}$, was found to be -14.18 (dB). This is indeed a significant improvement over the 4 × 4 test case, but at the compromise of supporting one less user.

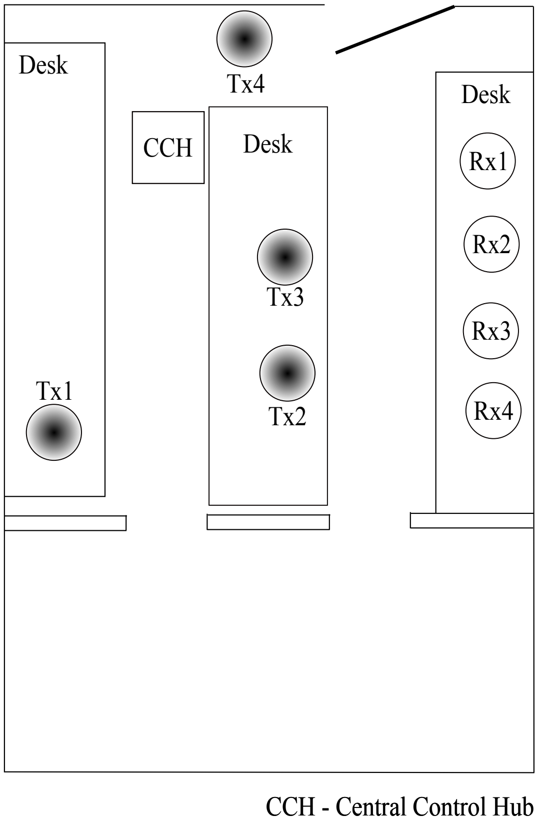


Figure 4.3: Indoor Layout 1 in room ITB214

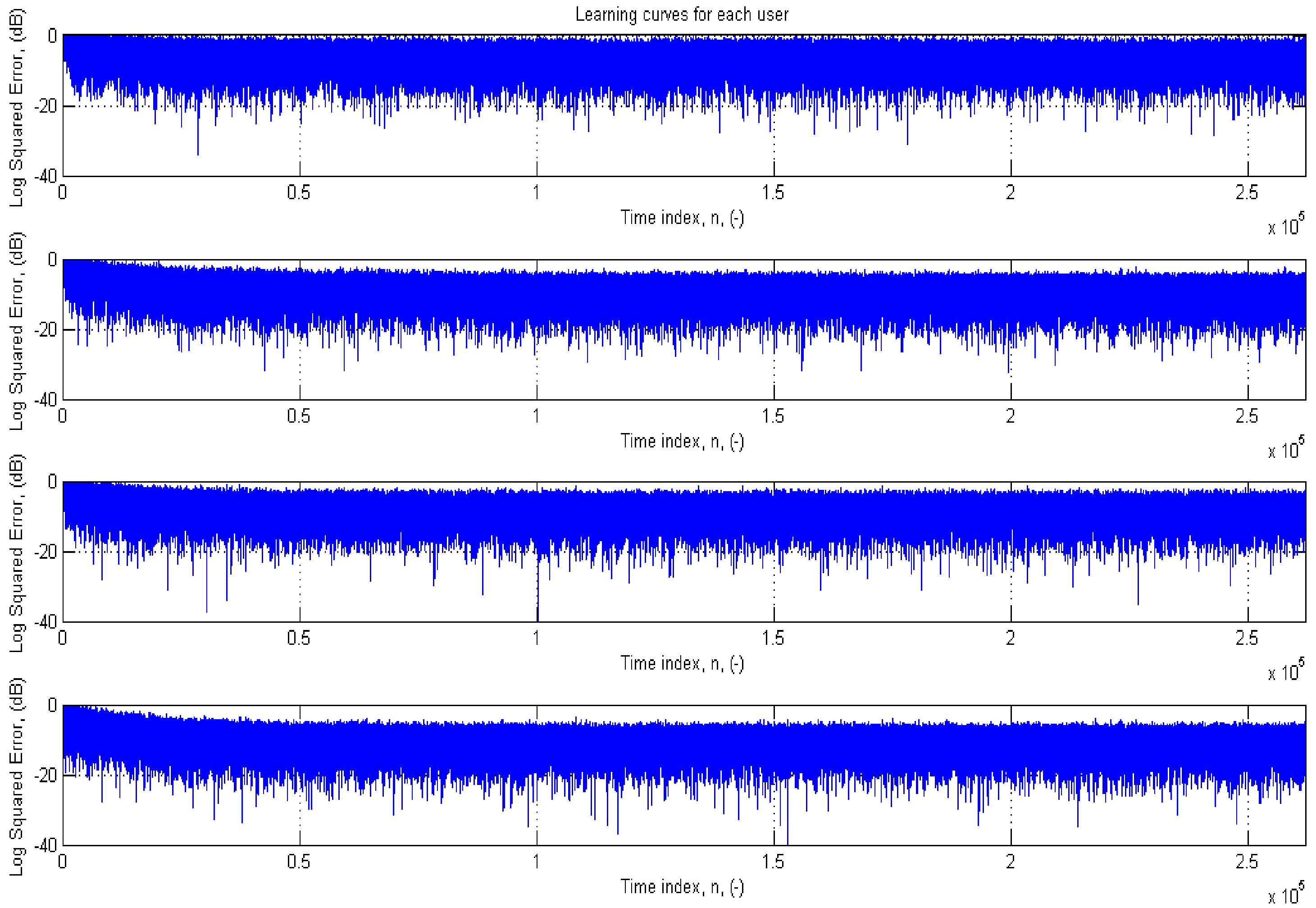


Figure 4.4: User Learning Curves for Indoor Layout 1 on 1902.5 MHz

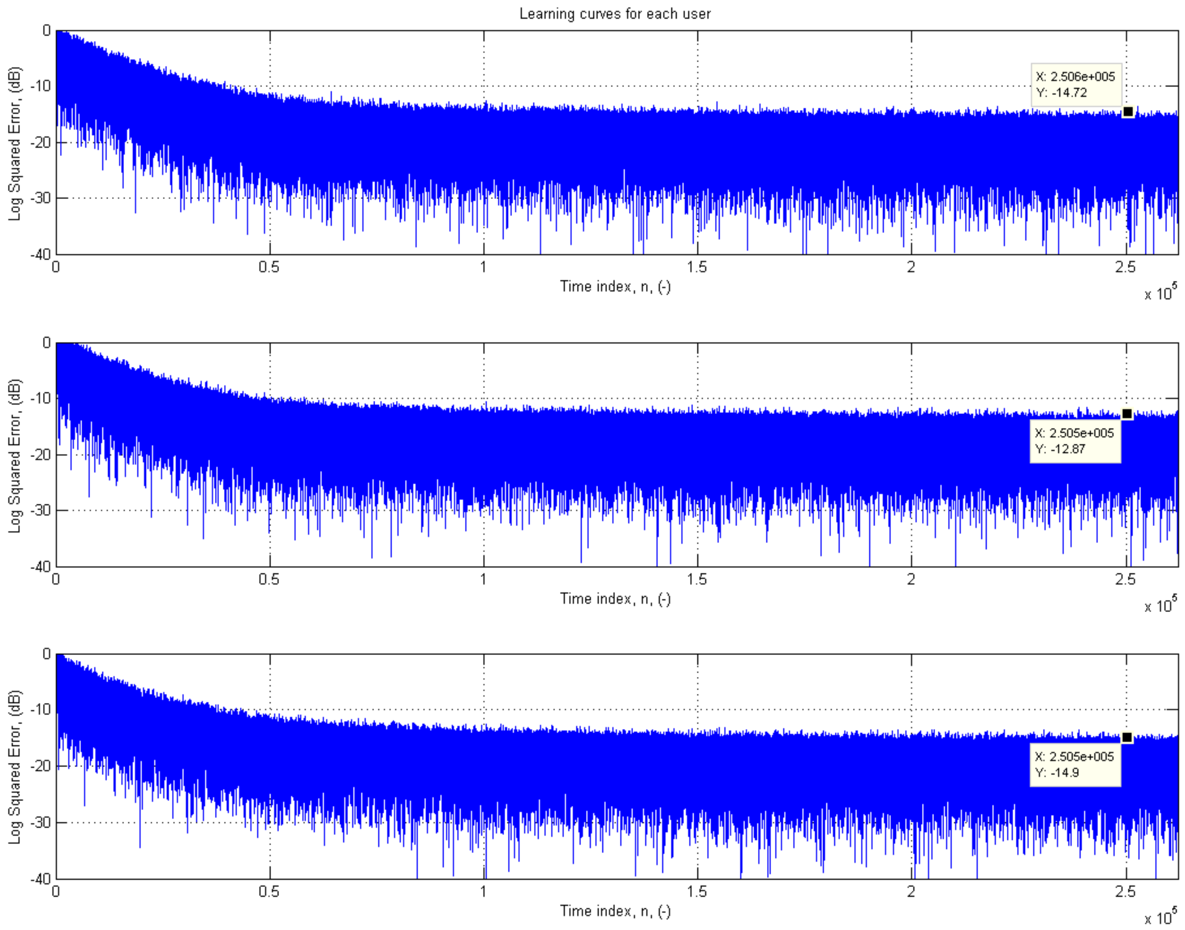


Figure 4.5: User Learning Curves for Indoor Layout 1 on 1902.5 MHz for 3x4 case

Table 4.2: Antenna Separation Distances (in m.) for Indoor Layout 1

	RX1	RX2	RX3	RX4
TX1	7.01	5.80	5.50	5.50
TX2	4.27	3.05	2.74	3.05
TX3	3.35	2.83	3.05	3.70
TX4	3.66	4.57	4.88	5.94

4.2.2 Indoor Layout 2 (IL2)

Figure 4.6 demonstrates the antenna setup for the chosen configuration in which the antennas are further separated, and Table 4.3 shows the separation distances in feet. After signal processing and LMS simulation, the MMSE for this layout, $\mathrm{MMSE}_{\mathrm{IL2}}$, was found to be -5.08 (dB). Figure 4.7 shows the curves from top to bottom for User 1 to User 4, respectively. Similar to the case for IL1, User 1 performance is weak at -1.50 (dB), User 2 and User 3 performance appear to improve at -7.00 (dB) and -7.50 (dB) respectively, and User 4 performance is acceptable at -4.3 (dB). This result seems to agree with the finding by Zhu et al. [20] that the channel matrix should become less complex and easier to invert with increase in antenna separation and multipath for MIMO channels.

Once again, given the weak identification of the User 1 signal from the above 4×4 case, a 3×4 test case was carried out in which transmit 1 was disabled, leaving all the other transmitters and receivers active. The obtained error curves are given in Figure 4.8, and the mean MMSE for all users, $\mathrm{MMSE}_{\mathrm{IL2}_{3\mathrm{x}4}}$, was found to be -17 (dB). This finding indicates an improvement of approximately 3 (dB) over $\mathrm{MMSE}_{\mathrm{IL1}_{3\mathrm{x}4}}$.

Table 4.3: Antenna Separation Distances (in m.) for Indoor Layout 2

	RX1	RX2	RX3	RX4
TX1	4.57	27.4	27.7	30.0
TX2	19.2	3.35	3.96	11.6
TX3	30.0	8.70	3.66	5.03
TX4	31.7	22.6	17.4	9.14

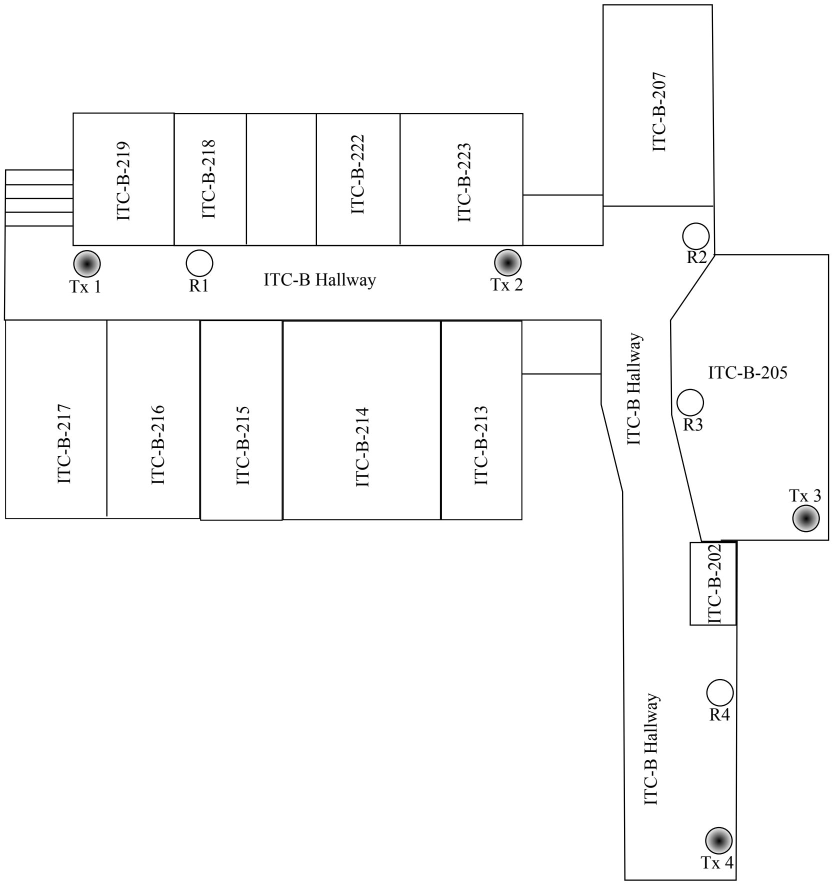


Figure 4.6: Indoor Layout 2 in room ITB214

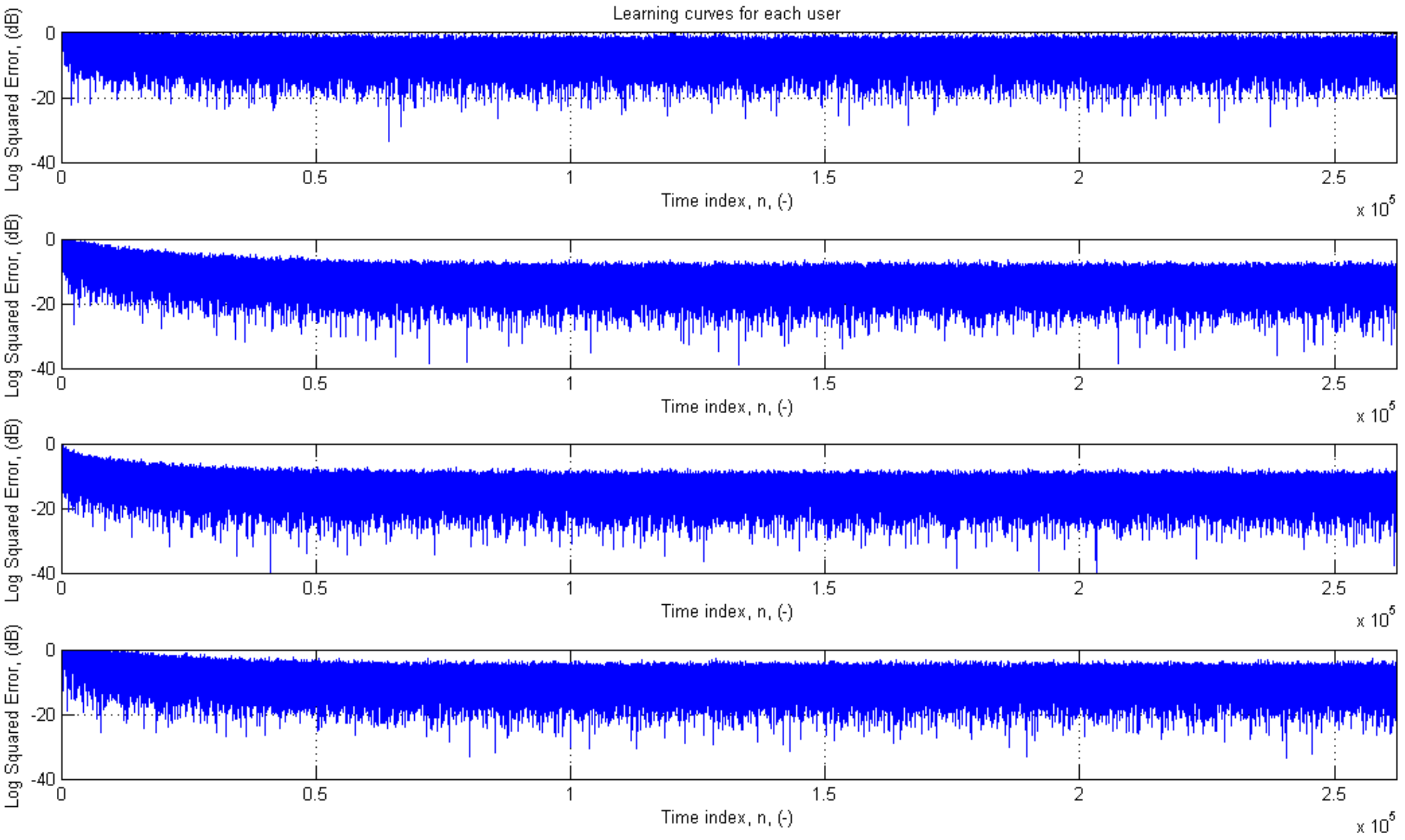


Figure 4.7: User Learning Curves for Indoor Layout 2 on 1902.5 MHz for 4x4 case

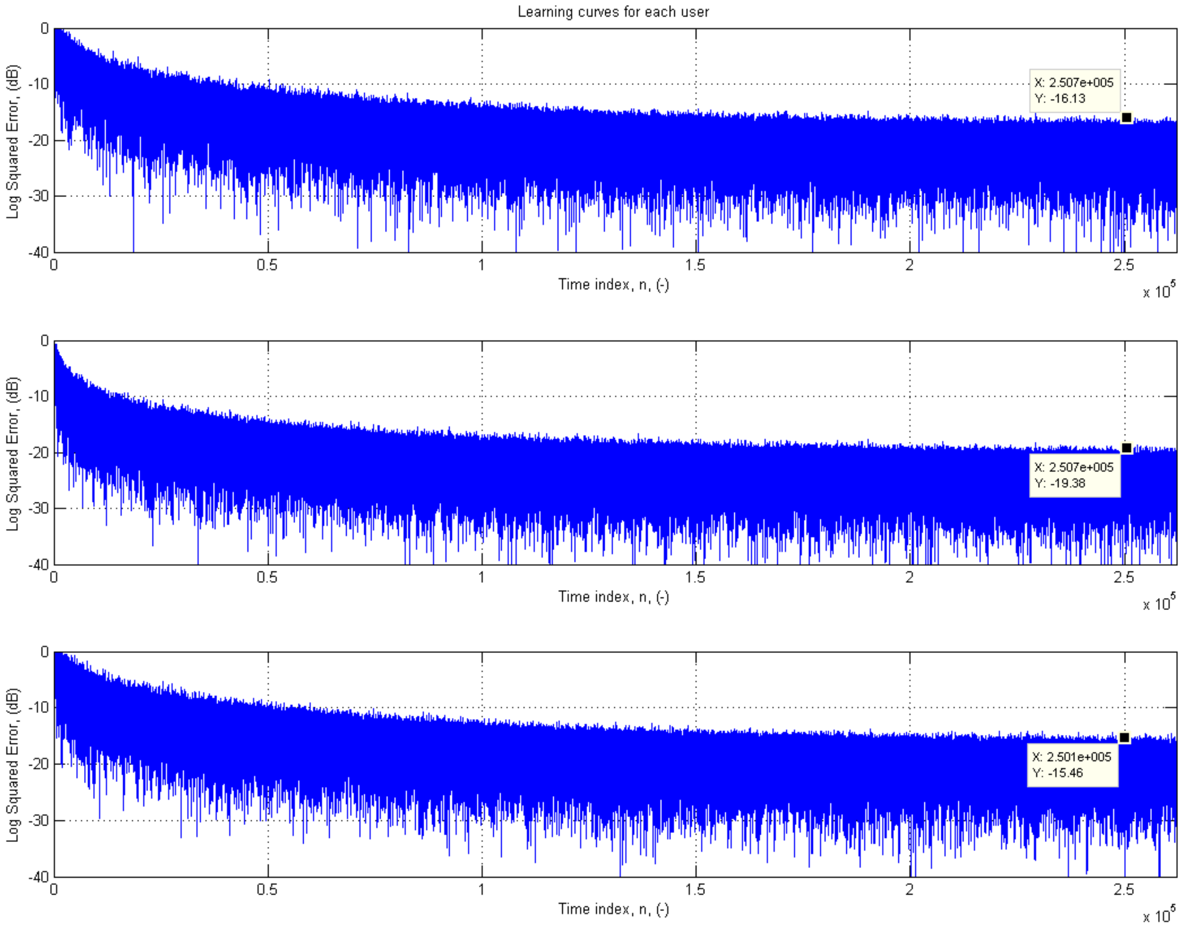


Figure 4.8: User Learning Curves for Indoor Layout 2 on 1902.5 MHz for 3x4 case

4.2.3 Outdoor Layout 1 (OL1)

Figure 4.9 shows the antenna positions for outdoor layout 1, an outdoor location in Fredericton, New Brunswick, taken from a Google Earth® snapshot with indicated scale.

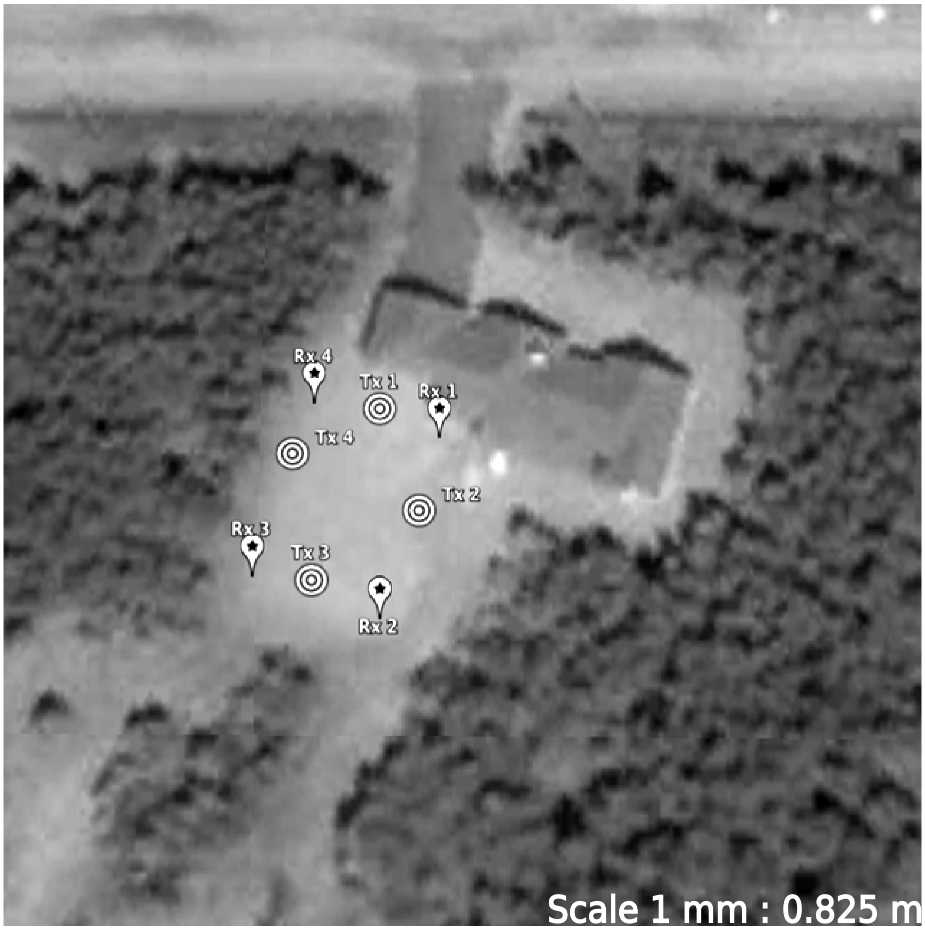


Figure 4.9: Outdoor Layout 1 on 1902.5 MHz

The learning curves for each user in this 4 × 4 configuration are given in Figure 4.10. After signal processing and LMS simulation, the MMSE for this layout, MMSE_{OL1}, was found to be -4.45 (dB). User 1 is acceptable at -3.00 (dB), and Users 2, 3, and 4 are good at -4.6 (dB), -5.7 (dB), and -4.5 (dB) respectively. With such a very small margin of difference between MMSE_{IL2} and MMSE_{OL1}, it is not possible to conclude whether there is a difference in overall performance between Indoor Layout

2 and Outdoor Layout 1. Users 2 and 3 performance for Indoor Layout 2 is better than Outdoor Layout 1, but users 1 and 4 perform better in Outdoor Layout 1 than in Indoor Layout 2.

Learning curves are also given for a 2 × 2 case of this layout in Figure 4.11, using transmitters marked "Tx2" and "Tx3" and receivers marked "Rx2" and "Rx3" as per Figure 4.9. After signal processing and LMS simulation, the MMSE for this layout, $\mathrm{MMSE}_{\mathrm{OL1}_{2\mathrm{x}2}}$, was found to be -24.1 (dB). This sort of improvement is expected as now with only two users and two receive antennas active, the channel matrix becomes significantly less complex and easier to invert.

4.2.4 Outdoor Layout 2 (OL2)

Figure 4.12 shows the antenna positions for outdoor layout 2, which is at the same location as outdoor layout 1 with indicated scale. "Tx5", "Tx6","Tx7", and "Tx8" in the this photograph are actually Tx 1, Tx2, Tx3, and Tx4 respectively.

User learning curves are given in Figure 4.13. After signal processing and LMS simulation, the MMSE for this layout, $\mathrm{MMSE}_{\mathrm{OL2}}$, was found to be -2.86 (dB), and the learning curves show the correspondingly weak performance for all users. These results were expected as the impulse responses recovered for this layout were of very low magnitude, with transmitter 4 practically appearing to be off. One possible explanation is that at the same level of RF transmit power and this increased range of antenna separation, along with a building shadowing one receiver from the other transmitters, results in a channel matrix in which all the channels have very small magnitudes thus making it very difficult to invert the channel and detect multiple users.

Learning curves are also given for a 2 × 2 case of this layout in Figure 4.14, using transmitters marked "Tx5" and "Tx8" and receivers marked "Rx6" and "Rx7" as per Figure 4.12. After signal processing and LMS simulation, the MMSE for

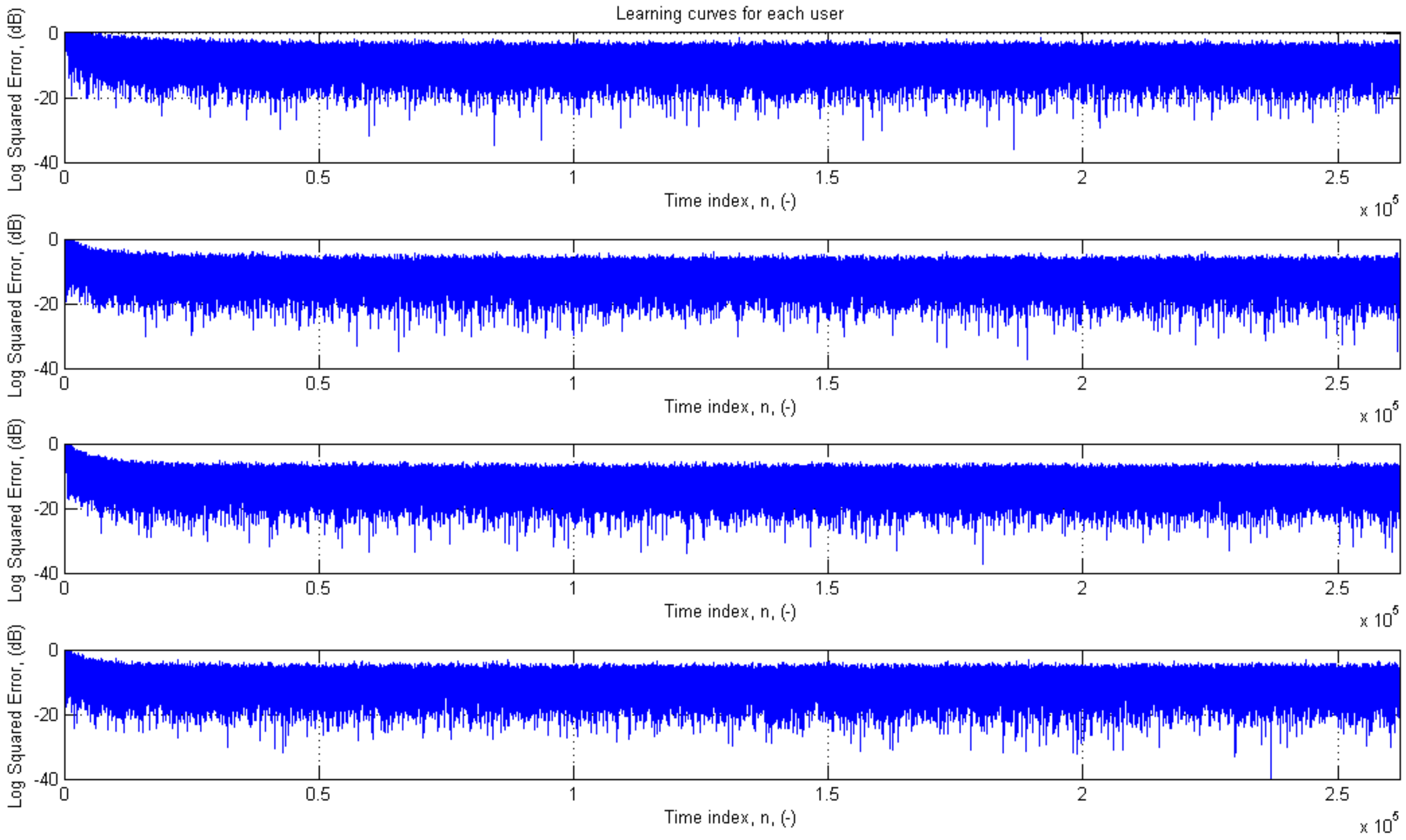


Figure 4.10: User Learning Curves for Outdoor Layout 1 on 1902.5 MHz for 4x4 case

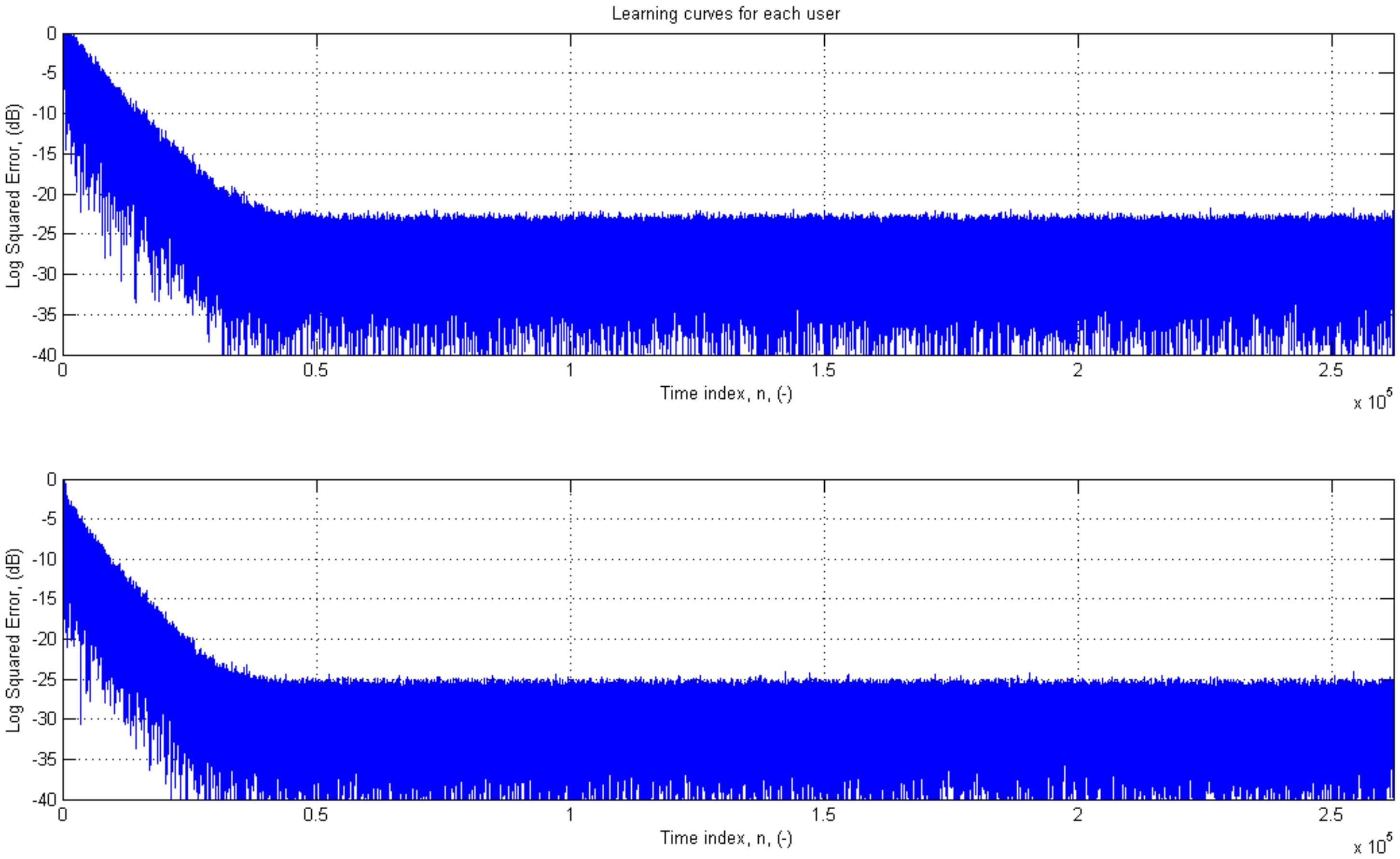


Figure 4.11: User Learning Curves for Outdoor Layout 1 on 1902.5 MHz for 2x2 case

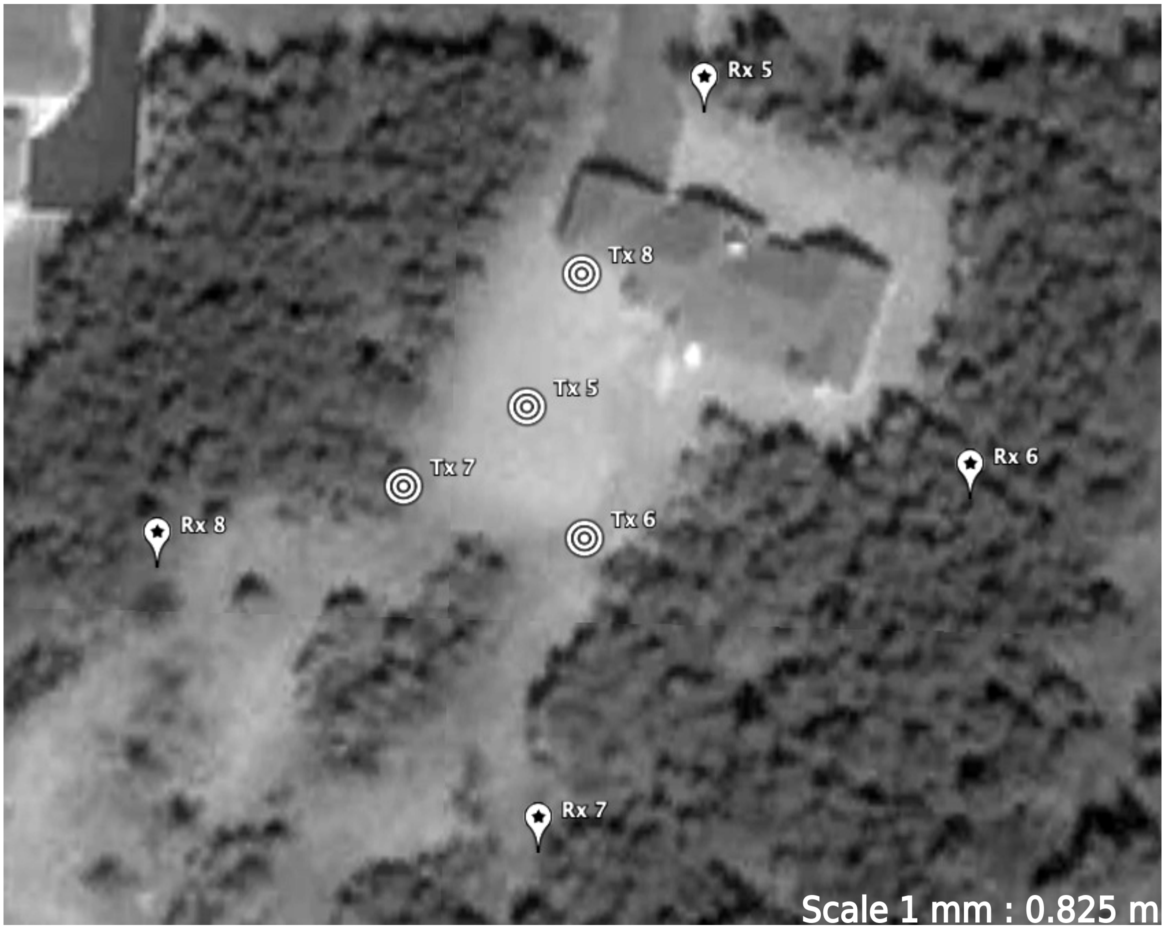


Figure 4.12: Outdoor Layout 2 on 1902.5 MHz

this layout, $\text{MMSE}_{\text{OL2}_{2\text{x}2}}$, was found to be -9.1 (dB), which again is an expected improvement over the 4 × 4 case due to reduction in channel matrix complexity.

4.2.5 Outdoor Layout 3 (OL3)

Figure 4.15 shows outdoor layout 3, with indicated scale. User learning curves are given in Figure 4.16 for a 2 × 2 case, using transmitters marked "Tx9" and "Tx11" and receivers marked "Rx10" and "Rx11" as per Figure 4.15. After simulation, the MSE for this two-user case, $\text{MMSE}_{\text{OL3}_{2\text{x}2}}$, was found to be approximately -11.64 (dB). This exhibits some improvement over the $\text{MMSE}_{\text{OL1}_{2\text{x}2}}$, possibly indicating a point where gains are starting to become noticeable with increase in antenna spacing.

4.2.6 Layout Analysis Summary

This section summarizes the findings for all layouts analyzed in the preceding subsections. Table 4.4 is a summary of the MMSE measurements obtained for each of the antenna layouts.

Table 4.4: MMSE Summary for all Layouts at 1902.5 MHz RF

Layout	Configuration	MMSE (dB)
Indoor Layout 1	4x4	-3.55
Indoor Layout 2	4x4	-5.08
Indoor Layout 1	3x4	-14.18
Indoor Layout 2	3x4	-17
Outdoor Layout 1	4x4	-4.45
Outdoor Layout 2	4x4	-2.86
Outdoor Layout 1	2x2	-24.1
Outdoor Layout 2	2x2	-9.1
Outdoor Layout 3	2x2	-11.64

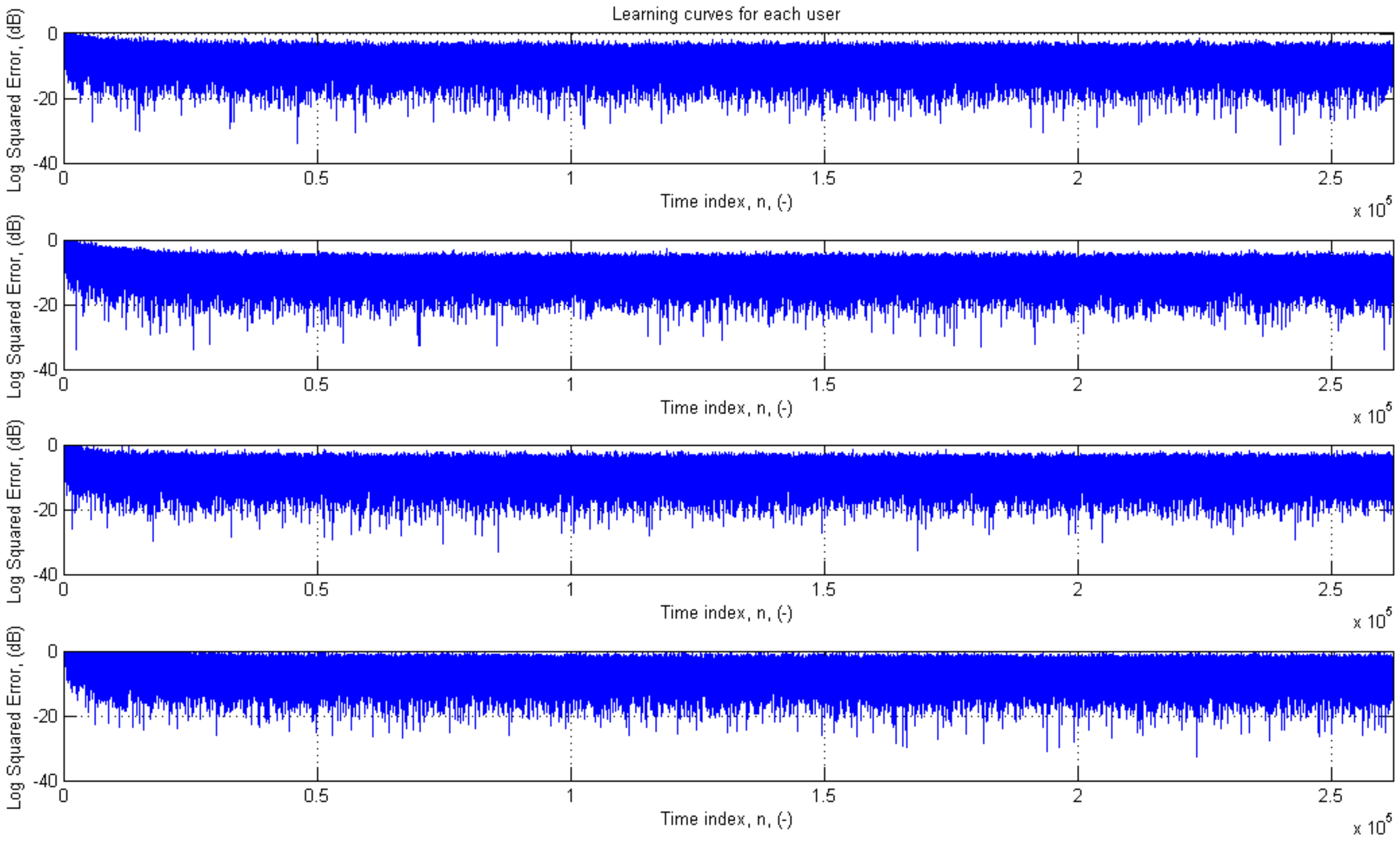


Figure 4.13: User Learning Curves for Outdoor Layout 2 on 1902.5 MHz for 4x4 case

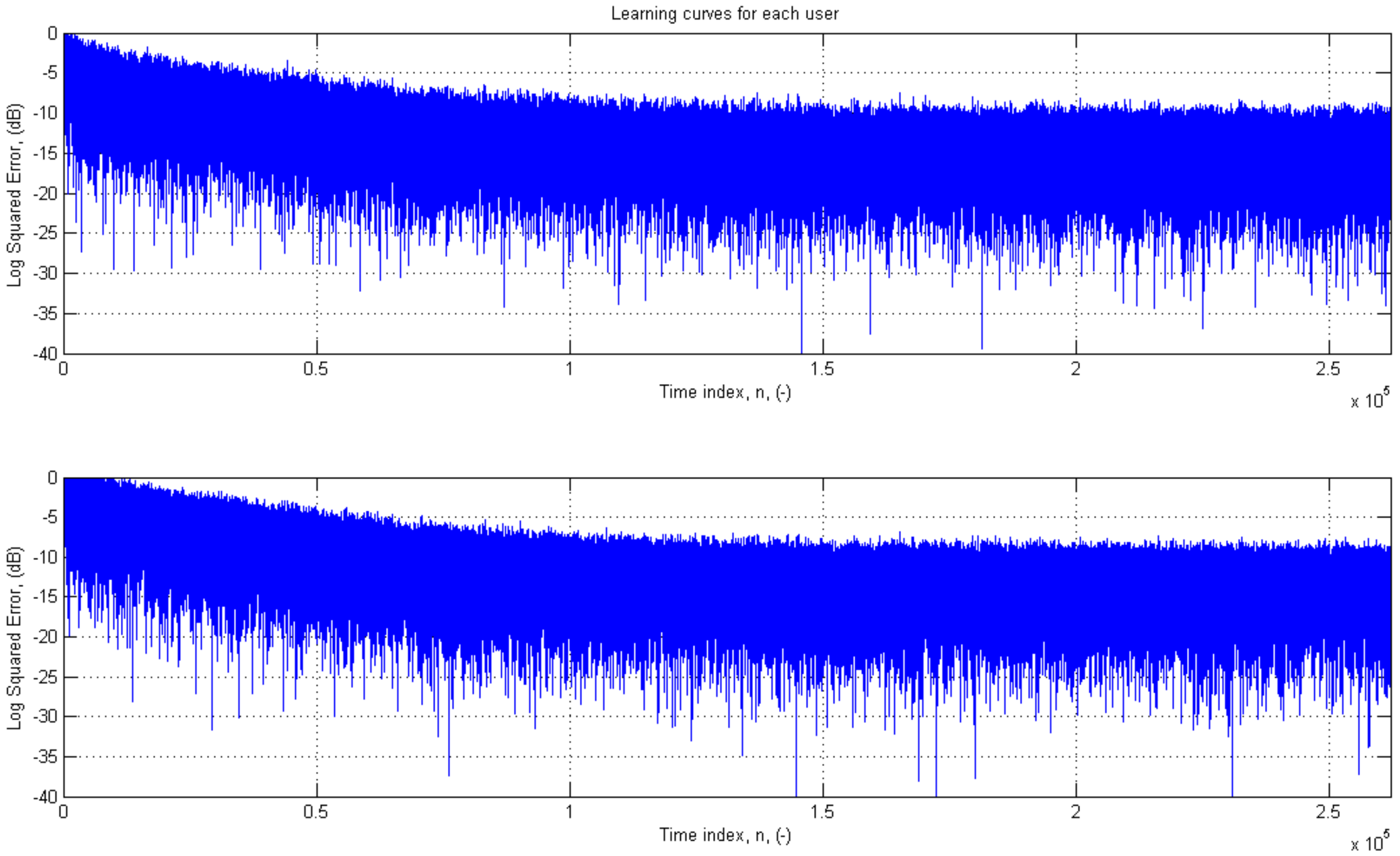


Figure 4.14: User Learning Curves for Outdoor Layout 2 on 1902.5 MHz for 2x2 case

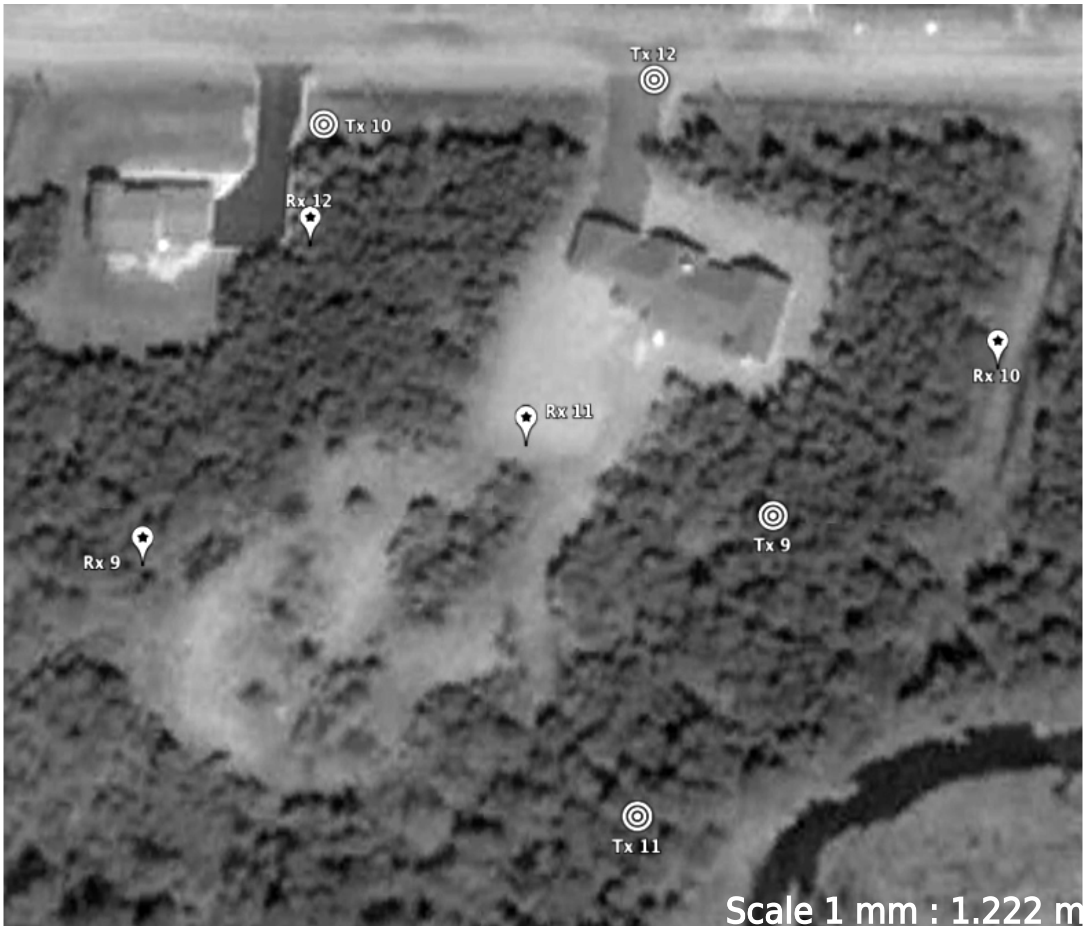


Figure 4.15: Outdoor Layout 3 on 1902.5 MHz

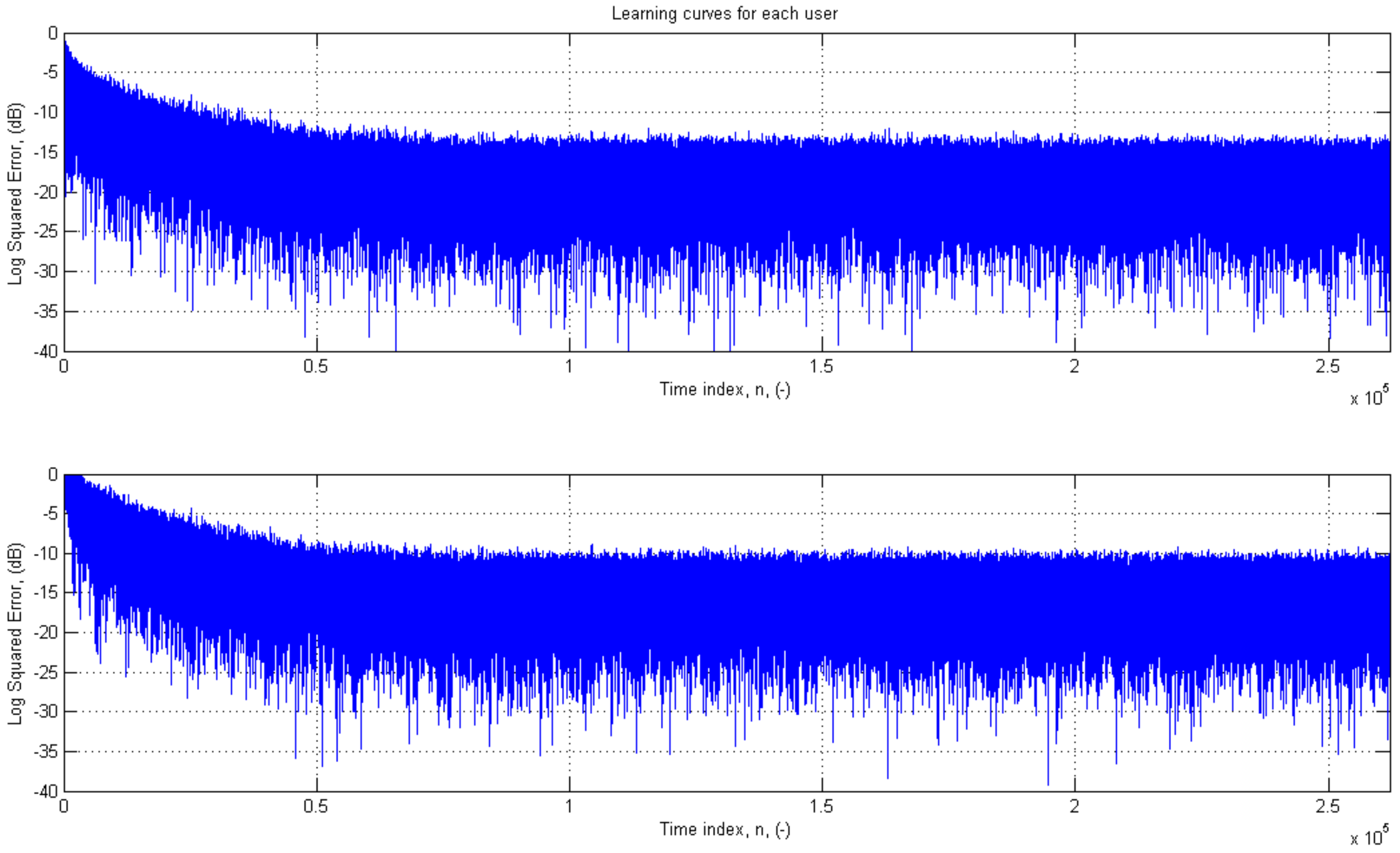


Figure 4.16: User Learning Curves for Outdoor Layout 3 on 1902.5 MHz for 2x2 case

Chapter 5

Summary and Future Work

This section includes a summary of experiment findings and future work suggestions for this thesis.

5.1 Summary of Results

For the indoor test cases, the full rank configuration yielded an improvement in MMSE of approximately 2.5 (dB) for increased antenna separation, while the reduced rank configurations yielded an improvement in MMSE of approximately 3 (dB) for increased antenna separation. It was also found that the reduced rank 3×4 case yielded an overall 11 (dB) MMSE improvement over the full rank 3×4 case, but, in eliminating a user, the overall number of transmitted symbols were decreased. The outdoor tests demonstrated a pattern in which the MMSE performance appeared to actually degrade with increase in antenna separation, with the exception of $\text{MMSE}_{\text{OL3}_{\text{2x2}}}$ showing a 2.5 (dB) improvement over the $\text{MMSE}_{\text{OL2}_{\text{2x2}}}$. There is some discrepancy with the outdoor testing however. It was noted that outdoor layout 1 yields a significantly improved MMSE over both outdoor layouts 2 and 3. One suggestion is that, even at 90 (m), there may still not be enough range on the antenna separations for observing symbol wavelength gains when transmitting only with 5 (MHz) bandwidth. At symbol rate, $\mathrm{f_T} = 2.5$ (MSym/s), the symbol wavelength, λ, is 120 (m). This means that within this range the channels would still be experiencing flat random fading, thus not allowing for observation of performance improvement from reduced spatial correlation. Another suggestion is that the absence of AGC at the receive end may have detracted from the layout comparisons. The received signal strength was observed as decreasing with corresponding increase in antenna separation, thus indeed showing degraded performance as signal is now closer to noise floor. However, if AGC was applied and signal strength dependence on distance was eliminated, channel performance is now being examined purely on the basis of spatial correlation thus forming a better basis of comparison.

5.2 Future Work

It is suggested that future outdoor RF transmission tests be carried out on a frequency which allows for wider bandwidth, for example, the 2.4 (GHz) ISM band. However, there is still the limitation that the radio IF signals are only 10 (MHz), thus limiting the baseband symbol rate to 5 (MSym/s) and so the antennas would still have to be separated by a minimum of $3 \times 10^8 (m/s)/5 \times 10^6 (Hz) = 60\ (m)$.

Another suggestion for future work stems from one of two approaches. One may either obtain an FPGA development platform that has eight digital-to-analog converters (DAC) so that pulse shaping and raised-cosines may be used for the transmitter sequences instead of rectangular PN sequences, thus allowing for more efficient use of transmission bandwidth using the current Hittite quadrature modulators, or one may employ a different method of modulation in which the total complex baseband signal for each transmit pair, $\mathrm{i_n(t) + jq_n(t), n = 1, 2, 3, 4}$, is directly modulated onto the RF carrier, thus only requiring four DACs for baseband pulse-shaping.

Changing the type of antennas on the transmitters and receivers may also be a future venture. The antennas used in this project were monopoles (vertically polarized), and, for the purposes for this thesis, were sufficient for the proposed channel measurement technique, but studies have been carried out indicating that significant MIMO performance gains may be achieved using antennas with dual or multiple polarization [21].

It may be good to implement some form of AGC at the receive end to form an improved basis of comparison for various antenna configurations, especially for the case of far antenna separations. This may be done by using a digitally programmable variable attenuator in conjunction with another amplifier (with gain range 15–25 (dB)) placed between each IF amplifier output and FPGA ADC input, thus requiring four attenuators and four additional amplifiers in total. A Verilog HDL routine may be implemented such that it accepts a specific FPGA ADC signal (e.g. “IF1”) as an input, then, upon comparison of ADC IF output signal peak level with user-defined thresholds, outputs transistor-transistor logic (TTL) levels, via FPGA GPIO pins, to the corresponding IF attenuator control circuit in order to select the various attenuation settings. Essentially, the attenuation is increased for higher IF peak levels and decreased for lower IF peak levels to effectively present a “constant” signal level to each ADC. One such attenuator is the 50P-1130 solid state 50 Ω programmable attenuator from JFW Industries Inc. (see manual [22]). This small, lightweight device has an operating frequency range of 1–200 (MHz) and attenuation range from 0–63.5 (dB) in steps of 0.5 (dB), making it a valid choice for installation into the current testbed design. This attenuator model comes stock with SMA or BNC female connectors, but custom connector orders may also be made with JFW Industries. The attenuator control circuit is interfaced with via a DE-9 connector, and the pin out with TTL levels for corresponding attenuation settings is given in the attenuator manual [22]. The Altera® Stratix® II EP2S180 FPGA development board is only fitted with one DE-9 connector, however, so either new control circuit data connections must be constructed from any free pins on the board or a different FPGA development board must be used. Currently, the J26, J27, and J28 expansion prototype connectors are not in use, and there are enough pins to create three more connections. Once control circuit data connections are arranged for all four attenuators, four of the comparator routines may be implemented for simultaneous dynamic control of each IF attenuator. The received signal strength should now be virtually independent of antenna separation distance. Conversely, a digitally programmable variable amplifier, such as the Analog Devices AD8260, may be used. The AD8260 consists of a high current driver and a digitally programmable variable gain amplifier, and has a gain range of -6–+24 (dB) in steps of 3 (dB). The amplifier settings are controlled via a 4-bit parallel interface, and, similar to the programmable attenuator, the amplifier gain settings may be driven with logic from FPGA pins after using a comparator routine to continually determine the required amplification for the IF signal of interest.

References

[1] W. W. Peterson, *Error-Correcting Codes.* Cambridge, Massachusetts: M.I.T Press, Massachusetts Institute of Technology, 1961.

[2] AOR, *AR5000 Operating Manual.* AOR Ltd., Tokyo, Japan.

[3] D. Gesbert, M. Shafi, D.-S. Shiu, P. Smith, and A. Naguib, "From theory to practice: an overview of MIMO space-time coded wireless systems," *Selected Areas in Communications, IEEE Journal on*, vol. 21, pp. 281–302, April 2003.

[4] Z. Sharif and A. Sha'ameri, "The application of cross correlation technique for estimating impulse response and frequency response of wireless communication channel," in *Research and Development, 2007. SCOReD 2007. 5th Student Conference*, (Selangor, Malaysia), pp. 1–5, Dec. 2007.

[5] J. Harriman, B. Petersen, and M. Kaye, "A reconfigurable four-channel transceiver testbed with signalling-wavelength-spaced antennas under centralised FPGA control," in *CNSR. Proceedings of the 4th Annual*, (Moncton, NB), pp. 311–313, May 2006.

[6] Antenna Factory, *F01710-8 Datasheet.* Antenna Factory, Inc., Schaumburg, Illinois, USA.

[7] Hittite, *HMC497LP4 Direct Quadrature Modulator Datasheet.* Hittite Microwave Corporation, Chelmsford, MA, USA.

[8] Mini-Circuits, *Plug-in & Coaxial Broadband, Linear Amplifiers Datasheet.* Mini-Circuits, Brooklyn, NY, USA.

[9] Standard Radio System Plan (SRSP-510), *Technical Requirements for Personal Communications Services in the bands 1850-1915 MHz and 1930-1995 MHz.* Industry Canada, issue 4 ed., February 2008.

[10] Radio Standards Specifications - 133 (RS-133), *2 GHz Personal Communications Services (PCS).* Industry Canada, issue 4 ed., February 2008.

[11] S. W. Golomb, *Shift Register Sequences.* San Francisco, California: Holden-Day, Inc., 1967.

[12] S. Catreux, V. Erceg, D. Gesbert, and J. Heath, R.W., "Adaptive modulation and MIMO coding for broadband wireless data networks," *Communications Magazine, IEEE*, vol. 40, pp. 108–115, Jun 2002.

[13] Jefa Tech, *RF Flexible Low Loss Cable - 50 Ohms Datasheet.* Jefa Tech Wireless Technology Solutions, Maryland, USA.

[14] G. Auer and I. Cosovic, "Pilot design for multi-user MIMO," in *Acoustics, Speech and Signal Processing. ICASSP. IEEE International Conference*, pp. 3621–3624, April 2009.

[15] L. Xin, L. Guangming, and W. Jianhua, "Preamble time-domain least-squares channel estimation algorithm for MIMO-OFDMA systems," in *Intelligent Networks and Intelligent Systems, 2008. ICINIS '08. First International Conference*, pp. 225–228, Nov. 2008.

[16] C. Pirak, K. Wang, Z.J.and Liu, and S. Jitapunkul, "A data-bearing approach for pilot-embedding frameworks in space-time coded mimo systems," *Signal Processing, IEEE Transactions on*, vol. 54, pp. 3966–3979, Oct. 2006.

[17] V. Polu, B. Colpitts, and B. Petersen, “Symbol-wavelength MMSE gain in a multi-antenna UWB system,” in *Communication Networks and Services Research (CNSR). Proceedings of the 4th Annual*, (Moncton, NB), pp. 95–99, May 2006.

[18] J. G. Proakis, *Digital Communications*. New York, NY, USA: McGraw-Hill Book Company, 1983.

[19] G. J. Foschini and J. Salz, “Digital communication over fading radio channels,” *Bell Syst. Tech. J.*, vol. 62, pp. 429–456, Feb. 1983.

[20] G. Zhu, B. Petersen, and B. Colpitts, “Signalling wavelength in an antenna array for space-time wireless over line-of-sight channels,” in *Communication Networks and Services Research (CNSR). Proceedings of the 3rd Annual*, (Halifax, N.S, Canada), pp. 69–73, May 2005.

[21] L. Liu, W. Hong, H. Wang, G. Yang, N. Zhang, H. Zhao, J. Chang, C. Yu, X. Yu, H. Tang, H. Zhu, and L. Tian, “Characterization of line-of-sight MIMO channel for fixed wireless communications,” *Antennas and Wireless Propagation Letters, IEEE*, vol. 6, pp. 36–39, 2007.

[22] J. Industries, *50P-130 Solid State Programmable Attenuator*. JFW Industries, Inc., Indianapolis, IN, USA, November 2007.

Appendix A

System Equipment

A.1 Low Loss 400 Cable Attenuation

The signal generator is set to 0 (dBm), its RF output connected to a 4-way RF splitter using a 4-(ft) length of RG-58 cable, and each RF splitter output port is connected to a back-to-back pair of ZX60-6013E+ amplifiers using a 1-(ft) length of RG-58 cable to each port. The output from each pair is connected to the bulkhead via a 3-(ft) length of RG-58, and the signal at each bulkhead connector is connected to the HP E4402B spectrum analyser via a 300-(ft) length of Jefa Tech® Low Loss 400 coaxial cable. The signal strength is monitored on the HP E4402B spectrum analyser for various settings of signal generator frequency as stated in Table A.1.

Table A.1: Measured Signal Strength Along Low Loss Cable

Carrier Frequency (MHz)	Received Signal Power (dBm)
0.01	3.09
1	-0.52
10	-7.33
100	-10.61
1902.5	-26.28
2500	-29.92

A.2 AOR Radio Block Diagram

Figure A.1 below is a block diagram of the AOR AR5000A wideband receiver. A switch is in place for selecting between the internal 12 (MHz) and external 10 (MHz) frequency standard, and the selected standard drives the PLLs for each filtering stage. The IF output is taken at the output of the second filtering stage, before onboard AGC is implemented, hence there is no AGC applied to IF output.

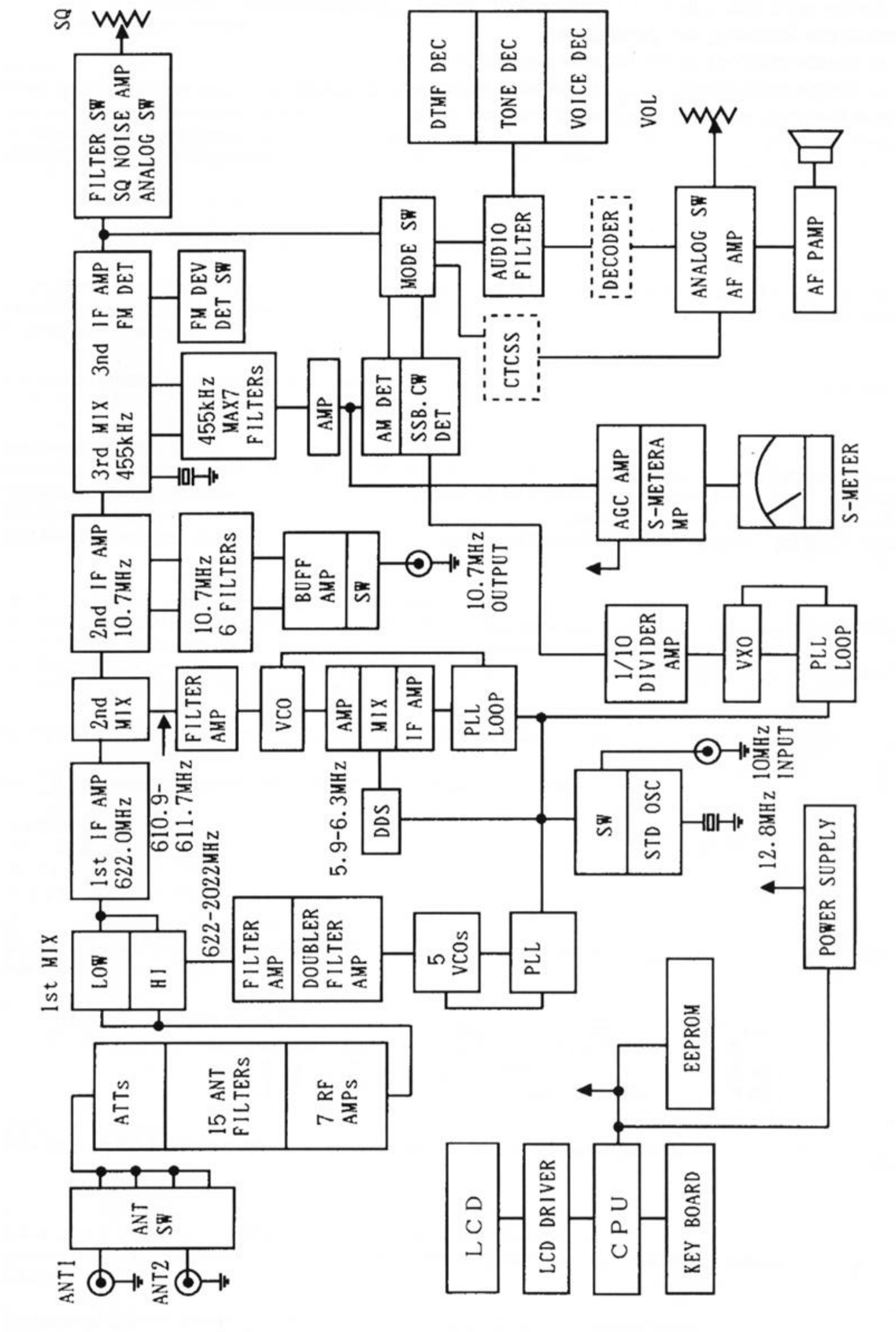


Figure A.1: AOR AR5000A Wideband Receiver Schematic [2]

Appendix B

MATLAB® Simulation Source Code

B.1 Transmit Power Limit Analysis

The function “wm_spectral_mask.m” determines the maximum transmit power than can be employed for complying with RSS-133 requirements, and it uses “wm_numint.m” to perform numerical integration. In the function “wm_numint.m”, the weights were obtained from text [?].

```
%
% wm_spectral_mask.m
%
% Spectral Mask for Compliance with RSS-133
% 1/T is to 2.5e6  as 0.02/T is to 50 (kHz)
T = 1/2.5e6 ;
l1 = -40 * (1/T) ;
l2 =  40 * (1/T) ;
N1 =  100 ;
%
order = 6 ;
Iexact = 1 ;
I1 = wm_numint('wm_pss',l1,l2,order,N1)
N2 = 2*N1 ;
I2 = wm_numint('wm_pss',l1,l2,order,N2)
relative_error_percent = ((I1-I2)/I2)*100
%
Ia = I2 ;
l1 =   0 * (1/T) ;
l2 =  40 * (1/T) ;
N1 =  100 ;
%
```

```
order = 6 ;
Iexact = 1 ;
I1 = wm_numint('wm_pss',l1,l2,order,N1)
N2 = 2*N1 ;
I2 = wm_numint('wm_pss',l1,l2,order,N2)
relative_error_percent = ((I1-I2)/I2)*100
%
Ib = I2 ;
l1 =  1.38 * (1/T) ;
l2 =  1.40 * (1/T) ;
N1 =  100 ;
%
order = 6 ;
Iexact = 1 ;
I1 = wm_numint('wm_pss',l1,l2,order,N1)
N2 = 2*N1 ;
I2 = wm_numint('wm_pss',l1,l2,order,N2)
relative_error_percent = ((I1-I2)/I2)*100
%
Ic = I2 ;
Rdb = 10 * log10 (Ic/Ia)
A = - Rdb ;
P  =  10 ^ ( (A-43)/10 )
PdBm  =  10 * log10(P/(10^(-3)))
-----
function I=wm_numint(F,l1,l2,order,N)
%WM_NUMINT I=wm_numint(F,l1,l2,order,N) approximates the integral of F between
%    l1 and l2 by N applications of a polynomial approximation of order
%    given by order.  In other words if order=1 a trapizoidal rule is
%    used.  If order=2, Simpson's rule is used.  And so on.
%    1 <= order <= 12.
%    Example:
% l1 = 0 ;  l2 = 20 ;  order = 6 ;
% N =  5 ;
% Iexact = exp(l2) - exp(l1) ;
% I1 = wm_numint('exp',l1,l2,order,N) ;
% N = 10 ;
% I2 = wm_numint('exp',l1,l2,order,N) ;
% relative_error_percent = ((I1-I2)/I2)*100

% Author: Wesley A. Miller  wes.mill@unb.ca
% Date Created:     July 22, 2009
if (l1<=l2)&(N>=1)&(fix(N)==N)&(length(F)>=1)&((1<=order)&(order<=12))& ...
   (fix(order)==order),

%
   if order == 1,
      weights0 = [1 1];
      denominator = 1;
   elseif order == 2,
      weights0 = [1 4 1];
      denominator = 3;
   elseif order == 3,
      weights0 = [1 3 3 1];
```

```
     denominator = 4;
  elseif order == 4,
     weights0 = [7 32 12 32 7];
     denominator = 45;
  elseif order == 5,
     weights0 = [19 75 50 50 75 19];
     denominator = 144;
  elseif order == 6,
     weights0 = [41 216 27 272 27 216 41];
     denominator = 420;
  elseif order == 7,
     weights0 = [751 3577 1323 2989 2989 1323 3577 751];
     denominator = 8640;
  elseif order == 8,
     weights0 = [989 5888 -928 10496 -4540 10496 -928 5888 989];
     denominator = 14175;
  elseif order == 9,
     weights0 = [2857 15741 1080 19344 5778 5778 19344 1080 15741 2857];
     denominator = 44800;
  elseif order == 10,
     weights0 = [16067 106300 -48525 272400 -260550 427368 -260550 272400 ...
-48525 106300 16067];
     denominator = 299376;
  elseif order == 11,
     weights0 = [2171465 13486539 -3237113 25226685 -9595542 15493566 ...
 15493566 -9595542 25226685 -3237113 13486539 2171465 ];
     denominator = 43545600;
  elseif order == 12,
     weights0 = [1364651 9903168 -7587864 35725120 -51491295 87516288 ...
 -87797136 ...
 87516288 -51491295 35725120 -7587864 9903168 1364651 ];
     denominator = 31531500;
  end

  if N==1,
     weights = weights0;
  else
     head = weights0(1:order);
     if order == 1,
        middleslice = [(weights0(1,1)+weights0(1,order+1))];
     else
        middleslice = [(weights0(1,1)+weights0(1,order+1)) weights0(2:order)];
     end
     maker=[];
     for i=1:order,
        maker(i,[i:order:order*(N-1)])=ones(1,N-1);
     end
     middle = middleslice * maker;
     clear middleslice
     clear maker
     tail = weights0(1,order+1);
     weights = [head middle tail];
     clear middle
```

```
   end

   delx=(l2-l1)/(order*N);
   x=l1:delx:l2;
   if length(x) == (order*N), x = [x x(order*N)+delx]; end
   y=feval(F,x);
   clear x
   ny=length(y);

   I = ((order*delx)/(2*denominator)) * sum(weights.*y);
end
-----
function y=wm_pss(f)
%
% Example:
%
% f = -5e6 : 1e5 : 5e6 ;
% y = wm_pss(f) ;
% plot(f,y)
T = 1 / 2.5e6 ;
y = T * sinc ( T * f ) .* sinc ( T * f ) ;
-----
```

B.2 Baseband Demodulation

This section contains the MATLAB functions used for baseband demodulation of the sampled IF signals. Function "sbin2dec.m" converts the binary data from SignalTap® II output into decimal equivalent for use in MATLAB®, while "WM_signal_extract_avg.m" processes up to four IF signals to recover inphase and quadrature signals from each IF, then cross-correlates these quadrature components with matched filters (developed using the sequences transmitted by the FPGA during IF signal acquisition) to implicitly yield the impulse responses. The impulses are then averaged "log_number" times before saving them in a matrix.

```
function [sRes, uRes] = sbin2dec(sbin)
% Function takes an array of signed binary values and converts it to
% its signed and unsigned decimal equivalent
N = max(size(sbin(1,:))); % N = number of bits in
                          % the binary array (= 14 for current data)
if(N<=1)
    disp('Error: The number of bits must be greater than or equal to 1.');
    result = NaN;
    return
end
%%
%Conversion to signed decimal
k=1;
sRes = zeros(size(sbin(:,1)));
for j=1:1:size(sbin,1),
    if sbin(j,1), %MSB is set
    sRes(k) = 0-2^N;
    end
    for i=2:1:N,
        sRes(k) = sRes(k) + (sbin(j,i)* 2^(N-i));
    end
    k=k+1;
end
%Invert MSB
for j=1:1:size(sbin,1),
    if(sbin(j,1))
        sbin(j,1) = 0;
    else
        sbin(j,1) = 1;
    end
end
%%
%Conversion to unsigned decimal
k=1;
uRes = zeros(size(sbin(:,1)));
for j=1:1:size(sbin,1),
    for i=1:1:N,
        uRes(k) = uRes(k)+sbin(j,i)*2^(N-i);
    end
    k=k+1;
end
-----
%
% WM_signal_extract_avg
%
% Extracts the ADC data from SignalTap .csv file, processes four
% Intermediate Frequency (IF) signals to extract inphase and quadrature
% signals for each IF, then uses generated matched filter(s) to yield
% impulse response(s). The impulses are then averaged "log_number" times.
% The generated matched filter is comprised of the same sequences
% transmitted by the FPGA, and the IF signals are actual data from FPGA
% ADCs used to sample incoming radio IF signals.
% SignalTap clock frequency  = 40 MHz
% FPGA ADC sampling frequency = 40 MHz
% I/Q clock frequency = 2.5 MHz
%
% Author: Wesley A. Miller wes.mill@unb.ca
% Date Created: May 26, 2009
% Date Modified: Nov 29, 2009
clear all;clc;
log_number = 125; %number of data logs
ir_tx1_set = 0;
ir_tx2_set = 0;
ir_tx3_set = 0;
ir_tx4_set = 0;

h11set = 0;
h12set = 0;
h13set = 0;
h14set = 0;

h21set = 0;
h22set = 0;
h23set = 0;
```

```
h24set = 0;

h31set = 0;
h32set = 0;
h33set = 0;
h34set = 0;

h41set = 0;
h42set = 0;
h43set = 0;
h44set = 0;

for i=1:log_number %data log loop
str = ['C:\Documents and Settings\labuser\Desktop\Data Analysis...
\Data_close_layout4\october_01_2009_1tx_ITB214_4rx_ITB214_(layout4)tr'...
num2str(i) '.csv'];
M = importdata(str,',',9);
time = M.data(:,1);
% adcBin1 = M.data(:,3:16); % ADC1 = IF1
% adcBin2 = M.data(:,18:31); % ADC2 = IF2
% adcBin3 = M.data(:,33:44); % onboardADC1
% adcBin4 = M.data(:,46:57); % onboardADC2

% Edit as of W.M. July 14, 2009
adcBin1 = M.data(:,3:14); % onboardADC1
adcBin2 = M.data(:,16:27); % onboardADC2
adcBin3 = M.data(:,29:42); % ADC1 = IF1
adcBin4 = M.data(:,44:57); % ADC2 = IF2
%
% FPGA-generated I and Q
%
i1_tx = M.data(:,58);
q1_tx = M.data(:,59);
i2_tx = M.data(:,60);
q2_tx = M.data(:,61);
i3_tx = M.data(:,62);
q3_tx = M.data(:,63);
i4_tx = M.data(:,64);
q4_tx = M.data(:,65);
%
% Convert signed binary ADC data to unsigned decimal for work in MATLAB
%
[adc1_sDec, adc1_uDec] = sbin2dec(adcBin1);
[adc2_sDec, adc2_uDec] = sbin2dec(adcBin2);
[adc3_sDec, adc3_uDec] = sbin2dec(adcBin3);
[adc4_sDec, adc4_uDec] = sbin2dec(adcBin4);
fs_adc = 40e6; % ADC Sampling Frequency of IF at 40 MHz
fs_stp = 40e6; % SignalTap clock at 40 MHz
fi = 10.7e6; % intermediate frequency is 10.7 MHz
fs = 2.5e6; % baseband I and Q frequency at 2.5 MHz
fs_adc_bb_ratio = fix(fs_adc/fs); % used for downsampling from ADC
                                  %sampling frequency to baseband
phase_est = 23; % estimated phase of carrier frequency in degrees
phase_est_rad = phase_est*(pi/180); % carrier phase in radians
```

```
i1_tx_ds = downsample(i1_tx,fs_adc_bb_ratio);
q1_tx_ds = downsample(q1_tx,fs_adc_bb_ratio);
i2_tx_ds = downsample(i2_tx,fs_adc_bb_ratio);
q2_tx_ds = downsample(q2_tx,fs_adc_bb_ratio);
i3_tx_ds = downsample(i3_tx,fs_adc_bb_ratio);
q3_tx_ds = downsample(q3_tx,fs_adc_bb_ratio);
i4_tx_ds = downsample(i4_tx,fs_adc_bb_ratio);
q4_tx_ds = downsample(q4_tx,fs_adc_bb_ratio);
impplotset = [1805:2305];
flag_plotset = 0; % 0: plot whole sequence, 1: plot subset of I/Q sequences
flag_impplotset = 1; % 0: plot whole impulse response 1: plot subset
%
% flag_plot_spectra
% 0: do not plot processed IF spectra, 1: plot processed IF spectra
flag_plot_spectra = 0;
x1 = adc1_sDec - mean(adc1_sDec);
x2 = adc2_sDec - mean(adc2_sDec);
x3 = adc3_sDec - mean(adc3_sDec);
x4 = adc4_sDec - mean(adc4_sDec);
%
% Processing IF1 signal
%
X1=fft(x1);
y1 = x1(1:32769);
Y1=fft(y1);
%
% Bandpass filtering on [5.7 MHz, 15.7 MHz]
%
fbplow = 5.7e6; % 5.7 MHz
fbphigh = 15.7e6; % 15.7 MHz
fbplratio = fbplow/(0.5*fs_adc);
fbphratio = fbphigh/(0.5*fs_adc);
h1=fir1(64,[fbplratio fbphratio]); % 64-Tap Bandpass (5.7 MHz - 15.7 MHz)
                                   % FIR filter
y2=conv(y1,h1);
Y2=fft(y2);
[nrows_y2 ncols_y2] = size(y2);
tdemod = (0:(nrows_y2-1))*(1/fs_adc); % time sample for IF demodulation
yc=sqrt(2)*cos(2*pi*fi*tdemod+phase_est_rad); % cosine to be mixed
                                              % with inphase signal
ys=-sqrt(2)*sin(2*pi*fi*tdemod+phase_est_rad); % negative sine to be
                                               % mixed with quadrature signal
zc=y2.*(yc’); % IF1 signal mixed with cosine to yield inphase component
zs=y2.*(ys’); % IF1 signal mixed with negative sine to
              % yield quadrature component
Zc=fft(zc);  % inphase1 fft
Zs=fft(zs);  % quadrature1 fft
flp = 10e6;   % 10 MHz cutoff for low pass filter
flpratio = flp/(0.5*fs_adc);
h2=fir1(64,flpratio,’low’); % 64-Tap Lowpass FIR Filter
zfc=conv(zc,h2); % inphase1 output from lpf
Zfc=fft(zfc); % fft inphase1 output from lpf
zfs=conv(zs,h2); % quadrature1 output from lpf
Zfs=fft(zfs); % fft quadrature1 output from lpf
```

```
%
% Downsample inphase1 and quadrature1 by f_adc_bb_ratio to baseband
%
i1_rxdemod = downsample(zfc, fs_adc_bb_ratio);
q1_rxdemod = downsample(zfs, fs_adc_bb_ratio);
[nrows_i1rxdemod,ncols_i1rxdemod] = size(i1_rxdemod);
[nrows_q1rxdemod,ncols_q1rxdemod] = size(q1_rxdemod);
%
% Averaging over data to yield logic I and Q
%
i_mid1 = sum(i1_rxdemod)/nrows_i1rxdemod;
q_mid1 = sum(q1_rxdemod)/nrows_q1rxdemod;
demod_i1 = (i1_rxdemod > i_mid1);
demod_q1 = (q1_rxdemod > q_mid1);
[nrows_demodi1, ncols_demodi1]= size(demod_i1);
[nrows_demodq1, ncols_demodq1]= size(demod_q1);
%
% Select a subset of points to plot for logic I/Q
%
if flag_plotset == 1,
    plotset = [200:700]; % used to select demod I/Q samples for display
else
    plotset = [1:nrows_demodi1-1];
end
if flag_plot_spectra == 1,
    %
    % Plotting spectra of IF1, mixed I and Q, and separate I/Q from IF1
    %
    figure;
    plot(time, x1); title('Sampled IF1 Signal');
    figure;
    subplot(421)
    f2=[0:(length(Y1)-1)]*(fs_adc/length(Y1));
    plot(f2,10*log10(abs(Y1)))
    xlabel('Spectrum of sampled IF1 signal (MHz)')
    subplot(422)
    f3=[0:(length(Y2)-1)]*(fs_adc/length(Y2));
    plot(f3,10*log10(abs(Y2)))
    xlabel('Spectrum of IF1 signal after bandpass filtering (MHz)')
    subplot(423)
    f4=[0:(length(Zc)-1)]*(fs_adc/length(Zc));
    plot(f4,10*log10(abs(Zc)))
    xlabel('Spectrum of IF1 signal mixed with cosine)
    subplot(424)
    f5=[0:(length(Zs)-1)]*(fs_adc/length(Zs));
    plot(f5,10*log10(abs(Zs)))
    xlabel('Spectrum of IF1 signal mixed with negative sine)
    subplot(425)
    f6=[0:(length(Zfc)-1)]*(fs_adc/length(Zfc));
    plot(f6,10*log10(abs(Zfc)))
    xlabel('Spectrum of lowpass filtered inphase signal (MHz)')
    subplot(426)
    f7=[0:(length(Zfs)-1)]*(fs_adc/length(Zfs));
    plot(f7,10*log10(abs(Zfs)))
    xlabel('Spectrum of lowpass filtered quadrature signal (MHz)')
    subplot(427)
    stairs(demod_i1(plotset),'g-', 'LineWidth',1);
    xlabel('Recovered I Signal from IF1');
    subplot(428)
    stairs(demod_q1(plotset),'g-', 'LineWidth',1);
    xlabel('Recovered Q Signal from IF1');
end % end if(flag_plot_spectra)
% END IF1 signal processing
%
% Processing IF2 signal
%
X2=fft(x2);
N=length(x2);
y3=x2(1:32769);
Y3=fft(y3);
%
% Bandpass filtering on [5.7 MHz, 15.7 MHz]
%
fbplow2 = 5.7e6; % 5.7 MHz
fbphigh2 = 15.7e6; % 15.7 MHz
fbplratio2 = fbplow2/(0.5*fs_adc);
fbphratio2 = fbphigh2/(0.5*fs_adc);
h3=fir1(64,[fbplratio2 fbphratio2]); % 64-Tap Bandpass (5.7 MHz - 15.7 MHz)
                                     % FIR filter
y4=conv(y3,h3);
Y4=fft(y4);
[nrows_y4 ncols_y4] = size(y4);
tdemod2 = (0:(nrows_y4-1))*(1/fs_adc); % time sample for IF demodulation
yc2=sqrt(2)*cos(2*pi*fi*tdemod2+phase_est_rad); % cosine to be mixed
                                                % with inphase signal
ys2=-sqrt(2)*sin(2*pi*fi*tdemod2+phase_est_rad); % negative sine to be mixed
                                                 % with quadrature signal
zc2=y4.*(yc2'); % Bandpassed IF2 signal mixed with cosine
                % to yield inphase component
zs2=y4.*(ys2'); % Bandpassed IF2 signal mixed with negative
                % sine to yield quadrature component
Zc2=fft(zc2);  % inphase2 fft
Zs2=fft(zs2);  % quadrature2 fft
flp2 = 10e6;   % 10 MHz cutoff for low pass filter
flpratio2 = flp2/(0.5*fs_adc);
h4=fir1(64,flpratio2,'low'); % 64-Tap Lowpass FIR Filter
zfc2=conv(zc2,h4); % inphase2 output from lpf
Zfc2=fft(zfc2); % fft inphase2 output from lpf
zfs2=conv(zs2,h4); % quadrature2 output from lpf
Zfs2=fft(zfs2); % fft quadrature2 output from lpf
%
% Downsample inphase2 and quadrature2 by f_adc_bb_ratio
%
i2_rxdemod = downsample(zfc2, fs_adc_bb_ratio);
q2_rxdemod = downsample(zfs2, fs_adc_bb_ratio);
[nrows_i2rxdemod,ncols_i2rxdemod] = size(i2_rxdemod);
[nrows_q2rxdemod,ncols_q2rxdemod] = size(q2_rxdemod);
%
```

```
% Average data to yield logic I and Q
%
i_mid2 = sum(i2_rxdemod)/nrows_i2rxdemod;
q_mid2 = sum(q2_rxdemod)/nrows_q2rxdemod;
demod_i2 = (i2_rxdemod > i_mid2);
demod_q2 = (q2_rxdemod > q_mid2);
[nrows_demodi2, ncols_demodi2]= size(demod_i2);
[nrows_demodq2, ncols_demodq2]= size(demod_q2);
% Select a subset of points to plot %
if flag_plotset == 1,
    plotset = [200:700]; % used to select demod I/Q samples for display
else
    plotset = [1:nrows_demodi2-1];
end
if flag_plot_spectra == 1,
    %
    % Plotting spectra and recovered I/Q from IF2
    %
    figure;
    plot(time, x2); title('Sampled IF2 Signal');
    figure;
    subplot(421)
    f8=[0:(length(Y3)-1)]*(fs_adc/length(Y3));
    plot(f8,10*log10(abs(Y3)))
    xlabel('Spectrum of sampled IF2 signal (MHz)')
    subplot(422)
    f9=[0:(length(Y4)-1)]*(fs_adc/length(Y4));
    plot(f9,10*log10(abs(Y4)))
    xlabel('Spectrum of IF2 signal after bandpass filtering (MHz)')
    subplot(423)
    f10=[0:(length(Zc2)-1)]*(fs_adc/length(Zc2));
    plot(f10,10*log10(abs(Zc2)))
    xlabel('Spectrum of IF2 signal mixed with cosine)
    subplot(424)
    f11=[0:(length(Zs2)-1)]*(fs_adc/length(Zs2));
    plot(f11,10*log10(abs(Zs2)))
    xlabel('Spectrum of IF2 signal mixed with negative sine)
    subplot(425)
    f12=[0:(length(Zfc2)-1)]*(fs_adc/length(Zfc2));
    plot(f12,10*log10(abs(Zfc2)))
    xlabel('Spectrum of lowpass filtered inphase2 signal (MHz)')
    subplot(426)
    f13=[0:(length(Zfs2)-1)]*(fs_adc/length(Zfs2));
    plot(f13,10*log10((abs(Zfs2))))
    xlabel('Spectrum of lowpass filtered quadrature2 signal (MHz)')
    subplot(427)
    stairs(demod_i2(plotset),'g-','LineWidth',1);
    xlabel('Recovered I signal from IF2');
    subplot(428)
    stairs(demod_q2(plotset),'g-','LineWidth',1);
    xlabel('Recovered Q Signal from IF2');
end % end if(flag_plot_spectra)
% END IF2 signal processing
%
```

```
% Processing IF3 signal
%
X3=fft(x3);
N=length(x3);
y5=x3(1:32769);
Y5=fft(y5);
%
% Bandpass filtering on [5.7 MHz, 15.7 MHz]
%
fbplow3 = 5.7e6; % 5.7 MHz
fbphigh3 = 15.7e6; % 15.7 MHz
fbplratio3 = fbplow3/(0.5*fs_adc);
fbphratio3 = fbphigh3/(0.5*fs_adc);
h5=fir1(64,[fbplratio3 fbphratio3]); % 64-Tap Bandpass (5.7 MHz - 15.7 MHz)
                                     % FIR filter
y6=conv(y5,h5);
Y6=fft(y6);
[nrows_y6 ncols_y6] = size(y6);
tdemod3 = (0:(nrows_y6-1))*(1/fs_adc); % time sample for IF demodulation
yc3=sqrt(2)*cos(2*pi*fi*tdemod3+phase_est_rad); % cosine to be mixed
                                                % with inphase signal
ys3=-sqrt(2)*sin(2*pi*fi*tdemod3+phase_est_rad); % negative sine to be mixed
                                                 % with quadrature signal

zc3=y6.*(yc3'); % Bandpassed IF2 signal mixed with cosine
                % to yield inphase component
zs3=y6.*(ys3'); % Bandpassed IF2 signal mixed with negative
                % sine to yield quadrature component
Zc3=fft(zc3);  % inphase3 fft
Zs3=fft(zs3);  % quadrature3 fft
flp3 = 10e6;   % 10 MHz cutoff for low pass filter
flpratio3 = flp3/(0.5*fs_adc);
h6=fir1(64,flpratio3,'low'); % 64-Tap Lowpass FIR Filter
zfc3=conv(zc3,h6); % inphase3 output from lpf
Zfc3=fft(zfc3); % fft inphase3 output from lpf
zfs3=conv(zs3,h6); % quadrature3 output from lpf
Zfs3=fft(zfs3); % fft quadrature3 output from lpf
%
% Downsample inphase3 and quadrature3 by f_adc_bb_ratio
%
i3_rxdemod = downsample(zfc3, fs_adc_bb_ratio);
q3_rxdemod = downsample(zfs3, fs_adc_bb_ratio);
[nrows_i3rxdemod,ncols_i3rxdemod] = size(i3_rxdemod);
[nrows_q3rxdemod,ncols_q3rxdemod] = size(q3_rxdemod);
%
% Average data to yield logic I and Q
%
i_mid3 = sum(i3_rxdemod)/nrows_i3rxdemod;
q_mid3 = sum(q3_rxdemod)/nrows_q3rxdemod;
demod_i3 = (i3_rxdemod > i_mid3);
demod_q3 = (q3_rxdemod > q_mid3);
[nrows_demodi3, ncols_demodi3]= size(demod_i3);
[nrows_demodq3, ncols_demodq3]= size(demod_q3);
%
```

```
% Select a subset of points to plot for logic I/Q
%
if flag_plotset == 1,
    plotset = [200:700]; % used to select demod I/Q samples for display
else
    plotset = [1:nrows_demodi3-1];
end
if flag_plot_spectra == 1,
    %
    % Plotting spectra and recovered I/Q from IF3
    %
    figure;
    plot(time, x3); title('Sampled IF3 Signal');
    figure;
    subplot(421)
    f14=[0:(length(Y5)-1)]*(fs_adc/length(Y5));
    plot(f14,10*log10(abs(Y5)))
    xlabel('Spectrum of sampled IF3 signal (MHz)')
    subplot(422)
    f15=[0:(length(Y6)-1)]*(fs_adc/length(Y6));
    plot(f15,10*log10(abs(Y6)))
    xlabel('Spectrum of IF3 signal after bandpass filtering (MHz)')
    subplot(423)
    f16=[0:(length(Zc3)-1)]*(fs_adc/length(Zc3));
    plot(f16,10*log10(abs(Zc3)))
    xlabel('Spectrum of IF3 signal mixed with cosine)
    subplot(424)
    f17=[0:(length(Zs3)-1)]*(fs_adc/length(Zs3));
    plot(f17,10*log10(abs(Zs3)))
    xlabel('Spectrum of IF3 signal mixed with negative sine)
    subplot(425)
    f18=[0:(length(Zfc3)-1)]*(fs_adc/length(Zfc3));
    plot(f18,10*log10(abs(Zfc3)))
    xlabel('Spectrum of lowpass filtered inphase3 signal (MHz)')
    subplot(426)
    f19=[0:(length(Zfs3)-1)]*(fs_adc/length(Zfs3));
    plot(f19,10*log10((abs(Zfs3))))
    xlabel('Spectrum of lowpass filtered quadrature3 signal (MHz)')
    subplot(427)
    stairs(demod_i3(plotset),'g-','LineWidth',1);
    xlabel('Recovered I signal from IF3');
    subplot(428)
    stairs(demod_q3(plotset),'g-','LineWidth',1);
    xlabel('Recovered Q Signal from IF3');
end % end if(flag_plot_spectra)
% END IF3 processing

%
% Processing IF4 signal
%
X4=fft(x4);
N=length(x4);
%y7=x4(1:65537);
y7=x4(1:32769);
```

```
Y7=fft(y7);
%
% Bandpass filtering on [5.7 MHz, 15.7 MHz]
%
fbplow4 = 5.7e6; % 5.7 MHz
fbphigh4 = 15.7e6; % 15.7 MHz
fbplratio4 = fbplow4/(0.5*fs_adc);
fbphratio4 = fbphigh4/(0.5*fs_adc);
h7=fir1(64,[fbplratio4 fbphratio4]); % 64-Tap Bandpass (5.7 MHz - 15.7 MHz)
                                     % FIR filter
y8=conv(y7,h7);
Y8=fft(y8);
[nrows_y8 ncols_y8] = size(y8);
tdemod4 = (0:(nrows_y8-1))*(1/fs_adc); % time sample for IF demodulation
yc4=sqrt(2)*cos(2*pi*fi*tdemod4+phase_est_rad); % cosine to be mixed
                                                % with inphase signal
ys4=-sqrt(2)*sin(2*pi*fi*tdemod4+phase_est_rad); % negative sine to be mixed
                                                 % with quadrature signal
zc4=y8.*(yc4’); % Bandpassed IF2 signal mixed with cosine
                % to yield inphase component
zs4=y8.*(ys4’); % Bandpassed IF2 signal mixed with negative
                % sine to yield quadrature component
Zc4=fft(zc4);  % inphase4 fft
Zs4=fft(zs4);  % quadrature4 fft
flp4 = 10e6;   % 10 MHz cutoff for low pass filter
flpratio4 = flp4/(0.5*fs_adc);
h8=fir1(64,flpratio4,’low’); % 64-Tap Lowpass FIR Filter
zfc4=conv(zc4,h8); % inphase4 output from lpf
Zfc4=fft(zfc4); % fft inphase4 output from lpf
zfs4=conv(zs4,h8); % quadrature4 output from lpf
Zfs4=fft(zfs4); % fft quadrature4 output from lpf
%
% Downsample inphase4 and quadrature4 by f_adc_bb_ratio
%
i4_rxdemod = downsample(zfc4, fs_adc_bb_ratio);
q4_rxdemod = downsample(zfs4, fs_adc_bb_ratio);
[nrows_i4rxdemod,ncols_i4rxdemod] = size(i4_rxdemod);
[nrows_q4rxdemod,ncols_q4rxdemod] = size(q4_rxdemod);
%
% Average data to yield logic I and Q
%
i_mid4 = sum(i4_rxdemod)/nrows_i4rxdemod;
q_mid4 = sum(q4_rxdemod)/nrows_q4rxdemod;
demod_i4 = (i4_rxdemod > i_mid4);
demod_q4 = (q4_rxdemod > q_mid4);
[nrows_demodi4, ncols_demodi4]= size(demod_i4);
[nrows_demodq4, ncols_demodq4]= size(demod_q4);
%
% Select a subset of points to plot for logic I/Q
%
if flag_plotset == 1,
    plotset = [200:700];
else
    plotset = [1:nrows_demodi4-1];
```

```
end
if flag_plot_spectra == 1,
    %
    % Plotting spectra and recovered I/Q from IF4
    %
    figure;
    plot(time, x4); title(’Sampled IF4 Signal’);
    figure;
    subplot(421)
    f20=[0:(length(Y7)-1)]*(fs_adc/length(Y7));
    plot(f20,10*log10(abs(Y7)))
    xlabel(’Spectrum of sampled IF4 signal (MHz)’)
    subplot(422)
    f21=[0:(length(Y8)-1)]*(fs_adc/length(Y8));
    plot(f21,10*log10(abs(Y8)))
    xlabel(’Spectrum of IF4 signal after bandpass filtering (MHz)’)
    subplot(423)
    f22=[0:(length(Zc4)-1)]*(fs_adc/length(Zc4));
    plot(f22,10*log10(abs(Zc4)))
    xlabel(’Spectrum of IF4 signal mixed with cosine)
    subplot(424)
    f23=[0:(length(Zs4)-1)]*(fs_adc/length(Zs4));
    plot(f23,10*log10(abs(Zs4)))
    xlabel(’Spectrum of IF4 signal mixed with negative sine)
    subplot(425)
    f24=[0:(length(Zfc4)-1)]*(fs_adc/length(Zfc4));
    plot(f24,10*log10(abs(Zfc4)))
    xlabel(’Spectrum of lowpass filtered inphase4 signal (MHz)’)
    subplot(426)
    f25=[0:(length(Zfs4)-1)]*(fs_adc/length(Zfs4));
    plot(f25,10*log10((abs(Zfs4))))
    xlabel(’Spectrum of lowpass filtered quadrature4 signal (MHz)’)
    subplot(427)
    stairs(demod_i4(plotset),’g-’,’LineWidth’,1);
    xlabel(’Recovered I signal from IF4’);
    subplot(428)
    stairs(demod_q4(plotset),’g-’,’LineWidth’,1);
    xlabel(’Recovered Q Signal from IF4’);
end % end if(flag_plot_spectra)
% END IF4 processing
%
% Create sequence of {-1,+1} from downsampled transmitted data from
% SignalTap clock frequency of 40 MHz
%
i1_tx_seq = 2 * i1_tx_ds - 1;
q1_tx_seq = 2 * q1_tx_ds - 1;
i2_tx_seq = 2 * i2_tx_ds - 1;
i3_tx_seq = 2 * i3_tx_ds - 1;
i4_tx_seq = 2 * i4_tx_ds - 1;
q2_tx_seq = 2 * q2_tx_ds - 1;
q3_tx_seq = 2 * q3_tx_ds - 1;
q4_tx_seq = 2 * q4_tx_ds - 1;
%
% Transmitted complex signals
```

```
%
tx1_cplx_seq = i1_tx_seq + 1i * q1_tx_seq;
tx2_cplx_seq = i2_tx_seq + 1i * q2_tx_seq;
tx3_cplx_seq = i3_tx_seq + 1i * q3_tx_seq;
tx4_cplx_seq = i4_tx_seq + 1i * q4_tx_seq;
%
% Matched sliding correlators
%
matched_cplx_filter1 = conj(tx1_cplx_seq(((length(tx1_cplx_seq):-1:1)))) ;
matched_cplx_filter2 = conj(tx2_cplx_seq(((length(tx2_cplx_seq):-1:1)))) ;
matched_cplx_filter3 = conj(tx3_cplx_seq(((length(tx3_cplx_seq):-1:1)))) ;
matched_cplx_filter4 = conj(tx4_cplx_seq(((length(tx4_cplx_seq):-1:1)))) ;
%
% Received complex signals
%
rx1_cplx_seq = i1_rxdemod + 1i * q1_rxdemod;
rx2_cplx_seq = i2_rxdemod + 1i * q2_rxdemod;
rx3_cplx_seq = i3_rxdemod + 1i * q3_rxdemod;
rx4_cplx_seq = i4_rxdemod + 1i * q4_rxdemod;
%
% Passing received signals through matched sliding correlators
%
ir_tx1 = conv(matched_cplx_filter1, tx1_cplx_seq);
h11 = conv(matched_cplx_filter1, rx1_cplx_seq);
h12 = conv(matched_cplx_filter1, rx2_cplx_seq);
h13 = conv(matched_cplx_filter1, rx3_cplx_seq);
h14 = conv(matched_cplx_filter1, rx4_cplx_seq);
ir_tx2 = conv(matched_cplx_filter2, tx2_cplx_seq);
h21 = conv(matched_cplx_filter2, rx1_cplx_seq);
h22 = conv(matched_cplx_filter2, rx2_cplx_seq);
h23 = conv(matched_cplx_filter2, rx3_cplx_seq);
h24 = conv(matched_cplx_filter2, rx4_cplx_seq);
ir_tx3 = conv(matched_cplx_filter3, tx3_cplx_seq);
h31 = conv(matched_cplx_filter3, rx1_cplx_seq);
h32 = conv(matched_cplx_filter3, rx2_cplx_seq);
h33 = conv(matched_cplx_filter3, rx3_cplx_seq);
h34 = conv(matched_cplx_filter3, rx4_cplx_seq);
ir_tx4 = conv(matched_cplx_filter4, tx4_cplx_seq);
h41 = conv(matched_cplx_filter4, rx1_cplx_seq);
h42 = conv(matched_cplx_filter4, rx2_cplx_seq);
h43 = conv(matched_cplx_filter4, rx3_cplx_seq);
h44 = conv(matched_cplx_filter4, rx4_cplx_seq);

    ir_tx1_set = ir_tx1_set + abs(ir_tx1(impplotset)).^2;
    ir_tx2_set = ir_tx2_set + abs(ir_tx2(impplotset)).^2;
    ir_tx3_set = ir_tx3_set + abs(ir_tx3(impplotset)).^2;
    ir_tx4_set = ir_tx4_set + abs(ir_tx4(impplotset)).^2;
    h11set = h11set + abs(h11(impplotset)).^2;
    h12set = h12set + abs(h12(impplotset)).^2;
    h13set = h13set + abs(h13(impplotset)).^2;
    h14set = h14set + abs(h14(impplotset)).^2;

    h21set = h21set + abs(h21(impplotset)).^2;
    h22set = h22set + abs(h22(impplotset)).^2;
    h23set = h23set + abs(h23(impplotset)).^2;
    h24set = h24set + abs(h24(impplotset)).^2;

    h31set = h31set + abs(h31(impplotset)).^2;
    h32set = h32set + abs(h32(impplotset)).^2;
    h33set = h33set + abs(h33(impplotset)).^2;
    h34set = h34set + abs(h34(impplotset)).^2;

    h41set = h41set + abs(h41(impplotset)).^2;
    h42set = h42set + abs(h42(impplotset)).^2;
    h43set = h43set + abs(h43(impplotset)).^2;
        h44set = h44set + abs(h44(impplotset)).^2;

end %end datalog loop

    ir_tx1_set = ir_tx1_set./log_number;
    ir_tx2_set = ir_tx2_set./log_number;
    ir_tx3_set = ir_tx3_set./log_number;
    ir_tx4_set = ir_tx4_set./log_number;

    h11set = h11set./log_number;
    h12set = h12set./log_number;
    h13set = h13set./log_number;
    h14set = h14set./log_number;

    h21set = h21set./log_number;
    h22set = h22set./log_number;
    h23set = h23set./log_number;
    h24set = h24set./log_number;

    h31set = h31set./log_number;
    h32set = h32set./log_number;
    h33set = h33set./log_number;
    h34set = h34set./log_number;

    h41set = h41set./log_number;
    h42set = h42set./log_number;
    h43set = h43set./log_number;
    h44set = h44set./log_number;
%
% Plotting Impulse Responses
%
    figure;
    subplot(321)
    stem(impplotset, ir_tx1_set, ’o’) ; grid ; title(’mag(ir_tx1.^2)’);
    subplot(322)
    stem(impplotset, ir_tx1_set, ’o’) ; grid ; title(’mag(ir_tx1.^2)’);
    subplot(323)
    stem(impplotset, h11set, ’o’) ; grid ; title(’mag(h11.^2)’);
    subplot(324)
    stem(impplotset, h12set, ’o’) ; grid ; title(’mag(h12.^2)’);
    subplot(325)
    stem(impplotset, h13set, ’o’) ; grid ; title(’mag(h13.^2)’);
    subplot(326)
```

```
stem(impplotset, h14set, 'o') ; grid ; title('mag(h14.^2)');
figure;
subplot(321)
stem(impplotset, ir_tx2_set, 'o') ; grid ; title('mag(ir_tx2.^2)');
subplot(322)
stem(impplotset, ir_tx2_set, 'o') ; grid ; title('mag(ir_tx2.^2)');
subplot(323)
stem(impplotset, h21set, 'o') ; grid ; title('mag(h21.^2)');
subplot(324)
stem(impplotset, h22set, 'o') ; grid ; title('mag(h22.^2)');
subplot(325)
stem(impplotset, h23set, 'o') ; grid ; title('mag(h23.^2)');
subplot(326)
stem(impplotset, h24set, 'o') ; grid ; title('mag(h24.^2)');
figure;
subplot(321)
stem(impplotset, ir_tx3_set, 'o') ; grid ; title('mag(ir_tx3.^2)');
subplot(322)
stem(impplotset, ir_tx3_set, 'o') ; grid ; title('mag(ir_tx3.^2)');
subplot(323)
stem(impplotset, h31set, 'o') ; grid ; title('mag(h31.^2)');
subplot(324)
stem(impplotset, h32set, 'o') ; grid ; title('mag(h32.^2)');
subplot(325)
stem(impplotset, h33set, 'o') ; grid ; title('mag(h33.^2)');
subplot(326)
stem(impplotset, h34set, 'o') ; grid ; title('mag(h34.^2)');
figure;
subplot(321)
stem(impplotset, ir_tx4_set, 'o') ; grid ; title('mag(ir_tx4.^2)');
subplot(322)
stem(impplotset, ir_tx4_set, 'o') ; grid ; title('mag(ir_tx4.^2)');
subplot(323)
stem(impplotset, h41set, 'o') ; grid ; title('mag(h41.^2)');
subplot(324)
stem(impplotset, h42set, 'o') ; grid ; title('mag(h42.^2)');
subplot(325)
stem(impplotset, h43set, 'o') ; grid ; title('mag(h43.^2)');
subplot(326)
stem(impplotset, h44set, 'o') ; grid ; title('mag(h44.^2)');
```

B.3 LMS Adaptive Filter Simulation

The following functions are LMS adaptive filter simulations, with the first function supporting four transmitters (or "users") and four receivers ("antennas") and the second function supporting two transmitters and two receivers. A set of channel impulse responses may be entered as a matrix (thus characterizing the radio channel) into either simulation which is then processed to yield individual user error curves and MMSE estimations to evaluate channel conditions.

```
%
% Author:      Wesley A. Miller wes.mill@unb.ca
% Date Created:     July 29, 2009
% Date Modified:    December 13, 2009
```

```
%
% The number of users is selectable.
% The spread spectrum expansion factor is selectable.
% The number of receiver antennas is selectable.
% 4-QAM modulation
% Complex baseband
% Obtained channel impules are used as radio channels
% Phase shifted downsampler
% LMS receiver
% Desired SNR at the receiver output
clear ;
clc; close all;
clear functions;
% load Oct_16_2009_4x4_19025M_close_matrix ;
% load Oct_19_2009_4x4_19025M_far_matrix ;
% load Oct_16_2009_4x4_19825M_close_matrix ;
% load Oct_19_2009_4x4_19825M_far_matrix ;
load dec_2009_impulses_close_far_veryfar;
C = outmatrixFar; % Figure 5.7 - Outdoor Layout 2
% C = outmatrixClose; % Figure 5.6 - Outdoor Layout 1

% 4 Tx and 4 Rx measurements, spread spectrum, maxmimum linear capacity
Nusers           =  4 ;    % number of users, must be 2, 3, 4, ...
Nantennas        =  4 ;    % number of receiver antennas
K                =  1 ;    % spreading factor, must be 1, 2, 3, ...
Nbits            =  2^18  ; % number of bits
D                = 15 ;  % Decoding delay, must be 0,1,2,...
DownSamplePhase  =  K-1 ; % phase delay of the receiver downsampler,
                          % an integer
Npoints_Wm       =  20 ; % number of points in w, must be 1,2,3,...
mu               =  2^(-9) ;  % LMS adaptation constant
SNR_Rxer_in_dB   =  40 ;  % (dB)

% Noise variance at each receiver antenna
sigmanm          =  10^(-(SNR_Rxer_in_dB/10)) ;

% Scale noise variance to match changes in simulation parameters
% and maintain a constant SNR at the receiver outputs
sigmanm          =  sigmanm * sqrt( Nantennas ) ;

% Initialize matrices for the transmitted information for all users.
bnall = [] ;
dnall = [] ;
amall = [] ;
pmall = [] ;
fmall = [] ;
for u = 1:Nusers ,
    % transmitter data, 4-QAM
    bn  =       ( 2 * ( rand(1,Nbits) < 0.5 ) - 1 )  +  ...
        1i * ( 2 * ( rand(1,Nbits) < 0.5 ) - 1 )         ;
    bn  =  (1/sqrt(2)) * bn ;
    % delayed data
    bDn  =  [ zeros(1,D) bn ] ;
    dn   =  bDn ;
    dn   =  dn ( [ 1 : (length(dn)-D)] ) ;
    % upsampled data
    am  =  reshape ( [ bn ; zeros(K-1,Nbits) ] , 1 , Nbits*K ) ;
    % spread spectrum spreading code
    %  generated randomly, complex, unit energy
    pm  =       ( 2 * ( rand(1,K) < 0.5 ) - 1 )  +  ...
        1i * ( 2 * ( rand(1,K) < 0.5 ) - 1 )         ;
    pm  =  pm / sqrt(sum(abs(pm).^2)) ;
    pm = 1 ;  %  Set each user to have no spreading code
    % transmitter output
    fm  =  conv ( pm , am ) ;
    % Save matrices with the transmitted information for all users.
    bnall  =  [ bnall ; bn ] ;
    dnall  =  [ dnall ; dn ] ;
    amall  =  [ amall ; am ] ;
    pmall  =  [ pmall ; pm ] ;
    fmall  =  [ fmall ; fm ] ;
end
% The channel has Nusers inputs and Nantennas outputs.
% Initialize matrices for the channel information for all users and antennas.
%     The number of rows in gmall is Nantennas.
cmall = [] ;
small = [] ;
gmall = [] ;
% Use 6 points around centre impulse response from each channel impulse
% response to form channel matrix with zero padding on each end
c11 = [ zeros(1,2) C(1,1:6) zeros(1,2) ] ;
c21 = [ zeros(1,2) C(2,1:6) zeros(1,2) ] ;
c31 = [ zeros(1,2) C(3,1:6) zeros(1,2) ] ;
c41 = [ zeros(1,2) C(4,1:6) zeros(1,2) ] ;
c12 = [ zeros(1,2) C(5,1:6) zeros(1,2) ] ;
c22 = [ zeros(1,2) C(6,1:6) zeros(1,2) ] ;
c32 = [ zeros(1,2) C(7,1:6) zeros(1,2) ] ;
c42 = [ zeros(1,2) C(8,1:6) zeros(1,2) ] ;
c13 = [ zeros(1,2) C(9,1:6) zeros(1,2) ] ;
c23 = [ zeros(1,2) C(10,1:6) zeros(1,2) ] ;
c33 = [ zeros(1,2) C(11,1:6) zeros(1,2) ] ;
c43 = [ zeros(1,2) C(12,1:6) zeros(1,2) ] ;
c14 = [ zeros(1,2) C(13,1:6) zeros(1,2) ] ;
c24 = [ zeros(1,2) C(14,1:6) zeros(1,2) ] ;
c34 = [ zeros(1,2) C(15,1:6) zeros(1,2) ] ;
c44 = [ zeros(1,2) C(16,1:6) zeros(1,2) ] ;
c11 = [ zeros(1,2) C(1,1:6) zeros(1,2) ] ;
c21 = [ zeros(1,2) C(2,1:6) zeros(1,2) ] ;
c12 = [ zeros(1,2) C(3,1:6) zeros(1,2) ] ;
c22 = [ zeros(1,2) C(4,1:6) zeros(1,2) ] ;

for a = 1:Nantennas ,
    gmaall = [] ;
    for u = 1:Nusers ,
        fm  =  fmall(u,:) ;
        if ( ( a == 1 ) && ( u == 1 ) )
            cm = c11 ;
        end
```

```
if ( ( a == 1 ) && ( u == 2 ) )
    cm = c12 ;
end
if ( ( a == 1 ) && ( u == 3 ) )
    cm = c13 ;
end
if ( ( a == 1 ) && ( u == 4 ) )
    cm = c14 ;
end
if ( ( a == 2 ) && ( u == 1 ) )
    cm = c21 ;
end
if ( ( a == 2 ) && ( u == 2 ) )
    cm = c22 ;
end
if ( ( a == 2 ) && ( u == 3 ) )
    cm = c23 ;
end
if ( ( a == 2 ) && ( u == 4 ) )
    cm = c24 ;
end
%
if ( ( a == 3 ) && ( u == 1 ) )
    cm = c31 ;
end
if ( ( a == 3 ) && ( u == 2 ) )
    cm = c32 ;
end
if ( ( a == 3 ) && ( u == 3 ) )
    cm = c33 ;
end
if ( ( a == 3 ) && ( u == 4 ) )
    cm = c34 ;
end
%
if ( ( a == 4 ) && ( u == 1 ) )
    cm = c41 ;
end
if ( ( a == 4 ) && ( u == 2 ) )
    cm = c42 ;
end
if ( ( a == 4 ) && ( u == 3 ) )
    cm = c43 ;
end
if ( ( a == 4 ) && ( u == 4 ) )
    cm = c44 ;
end
% channel output
sm  =  conv ( cm , fm ) ;
% Save matrices with the channel information for
% all users and antennas.
cmall  =  [ cmall ; cm ] ;
small  =  [ small ; sm ] ;
% Save matrix with received information
```

```
        gmaall = [ gmaall ; sm ] ;
    end
    gma  =  sum(gmaall,1) ;

    % Normalize the signal at the antenna output
    % to have unity power.  This simulates perfect
    % automatic gain control (AGC)
    gma  =  gma / sqrt(mean(abs(gma).^2)) ;

    % Save matrix with received information
    gmall = [ gmall ; gma ] ;
end

% Generate complex noise
% Initialize noise matrix
%     The number of rows in nmall is Nantennas.
nmall = [] ;

for a = 1:Nantennas ,
    nm      =        randn(1,length(sm))  +  ...
        1i *  randn(1,length(sm))          ;
    nm      =  sigmanm  *  nm ;

    % Save noise matrix
    nmall  =  [ nmall ; nm ] ;
end

% Find the receiver input.
%     The number of rows in rmall is Nantennas.
rmall  =  gmall + nmall ;

% Receiver
[Nrows,Ncolumns]  =  size(rmall) ;
Nsamples = Ncolumns ;

% Initialize matrices for the receiver information for all users.
umall = [] ;
Wmall = [] ;
Rmall = [] ;
dhatnall = [] ;
eall = [] ;

for u = 1:Nusers ,
    % Extract the signals for user u
    bn  =  bnall(u,:) ;
    dn  =  dnall(u,:) ;
    % Approximately pre-size the array sizes, for reduced processing time
    um                =  zeros(1,Nsamples) ;
    errors_recordedn  =  0 * bn ;
    dhatns            =  0 * bn ;
    % Initialize the adaptive filter
    Wm  =  zeros(Nantennas,Npoints_Wm) + 1i*zeros(Nantennas,Npoints_Wm) ;
    Rm  =  zeros(Nantennas,Npoints_Wm) + 1i*zeros(Nantennas,Npoints_Wm) ;
    Rm  =  [ rmall(:,1) Rm(:,[1:(Npoints_Wm-1)]) ] ;
    n = 0 ;
    for m  =  0 : (Nsamples-2) ,
        % regular FIR filter processing
        % obtain filter output
        um(1+ m)  =  sum ( sum ( Wm .* Rm ) ) ;
        % downsample
        if ( mod(m-DownSamplePhase,K) == 0 ) ,
            dhatn  =  um(1+ m) ;
            dhatns(1+ n)  =  dhatn ;
            % adapt the filter provided the delayed data exists
            if  ( 1 <= (1+ n) )  &  ( (1+ n) <= length(dn) )  ,
                % adapt the filter provided the delayed data is present
                if  ( dn(1+ n) ˜= 0 )  ,
                    en  =  dn(1+ n) - dhatn ;
                    % errors_recordedn(1+ n-D) = en ;
                    errors_recordedn(1+ n  ) = en ;
                    Wm   =   Wm  +  mu * en * conj(Rm) ;
                end
            end
            n  =  n + 1 ;
        end
        % shift FIR filter state
        Rm  =  [ rmall(:,1+ m+1)  Rm(:,[1:(Npoints_Wm-1)]) ] ;
    end  % for m  =  0 : (Nsamples-2) ,
    % Save matrices with the received information for all users.
    umall  =  [ umall ; um ] ;
    Wmall  =  [ Wmall ; reshape(Wm,1,Nantennas*Npoints_Wm) ] ;
    Rmall  =  [ Rmall ; reshape(Rm,1,Nantennas*Npoints_Wm) ] ;
    dhatnall  =  [ dhatnall ; dhatns ] ;
    eall  =  [ eall ; errors_recordedn ] ;
end  % for u = 1:Nusers ,

% Display results

% Coefficients Nantennas by Nusers
% clf ;
figure(1) ;
for u = 1:Nusers ,
    Wm  =  reshape(Wmall(u,:),Nantennas,Npoints_Wm) ;
    for a = 1:Nantennas ,
        subplot(Nantennas,Nusers,Nusers*(a-1)+u) ;
        iWm = [ 0 : (Npoints_Wm-1) ] ;
        stem(iWm,abs(Wm(a,:)).ˆ2,’o’) ;
        % ylabel(’Tap Energy, |wm|ˆ2’) ;
        % xlabel(’Index, i’) ;
        % stem(iWm,angle(Wm),’o’) ;
        % ylabel(’tap angle, angle(wm)’) ;
        % xlabel(’Index, i’) ;
        if ( Nusers*(a-1)+u ) == 1 ,
            title(’Coefficients (Nantennas by Nusers), W’) ;
        end
    end  % for a = 1:Nantennas ,
end  % for u = 1:Nusers ,
% Learning curves for each user
figure(2) ;
for u = 1:Nusers ,
    errors_recordedn  =  eall(u,:) ;
    subplot(Nusers,1,u) ;
    i_se_db  =  0 : (length(errors_recordedn)-1) ;
    se_db  =  10 * log10(abs(errors_recordedn)+eps) ;
    plot(i_se_db,se_db) ;
    grid ;
    axis ( [ 0 (Nbits-1) -SNR_Rxer_in_dB 0] ) ;
    ylabel('Log Squared Error, (dB)') ;
    xlabel('Time index, n, (-)') ;
    if u == 1 ,
        title('Learning curves for each user') ;
    end
end  % for u = 1:Nusers ,

% Presume the second half of the learning curve is converged and find the MSE
% based on the values in the second half of the simulation.
% iforavg  =  [ (fix(length(errors_recordedn)/2)) : length(errors_recordedn) ] ;
% squarederrors  =  reshape( eall(:,iforavg) , 1 , Nusers*length(iforavg) ) ;
% msedB  =  10*log10( mean( abs(squarederrors).^2 ) ) ;
% disp(sprintf('The MSE after convergence for all users is %0.2f (dB).',msedB));
-----
%
% Author:      Wesley A. Miller wes.mill@unb.ca
% Date Created:     July 29, 2009
% Date Modified:    December 13, 2009
%
% The number of users is selectable.
% The spread spectrum expansion factor is selectable.
% The number of receiver antennas is selectable.
% 4-QAM modulation
% Complex baseband
% Obtained channel impules are used as radio channels
% Phase shifted downsampler
% LMS receiver
% Desired SNR at the receiver output
clear ;
clc; close all;
clear functions;
% load Oct_16_2009_4x4_19025M_close_matrix ;
% load Oct_19_2009_4x4_19025M_far_matrix ;
% load Oct_16_2009_4x4_19825M_close_matrix ;
% load Oct_19_2009_4x4_19825M_far_matrix ;
% load dec_2009_outdoorimpulses;
load dec_2009_impulses_close_far_veryfar;
% C = outmatrix ;
% C = outmatrixFar; % Figure 5.7 - Outdoor Layout 2
% C = outmatrixClose; % Figure 5.6 - Outdoor Layout 1
% C = outmatrixClose2x2; % Figure 5.6 - Outdoor Layout 1
% C = outmatrixFar2x2; % Figure 5.7 - Outdoor Layout 2
% C = outmatrixVeryFar; % Figure 5.8 - Outdoor Layout 3
C = outmatrixLayout5_2x2;
```

```
%
% 2 Tx and 2 Rx measurements, spread spectrum, maxmimum linear capacity
Nusers           =  2 ;    % number of users, must be 2, 3, 4, ...
Nantennas        =  2 ;    % number of receiver antennas
K                =  1 ;    % spreading factor, must be 1, 2, 3, ...
Nbits            =  2^18  ; % (-), number of bits
D                = 15 ;  % (T-spaced samples), Decoding delay, must be 0,1,2,...
DownSamplePhase  =  K-1 ; % phase delay of the receiver downsampler,
% an integer
Npoints_Wm       =  20 ; % (-), number of points in w, must be 1,2,3,...
mu               =  2^(-9) ;  % (-), LMS adaptation constant
SNR_Rxer_in_dB   =  40 ;  % (dB)
% Noise variance at each receiver antenna
sigmanm          =  10^(-(SNR_Rxer_in_dB/10)) ;
% Scale noise variance to match changes in simulation parameters
% and maintain a constant SNR at the receiver outputs
sigmanm          =  sigmanm * sqrt( Nantennas ) ;
% Initialize matrices for the transmitted information for all users.
bnall = [] ;
dnall = [] ;
amall = [] ;
pmall = [] ;
fmall = [] ;
for u = 1:Nusers ,
    % transmitter data, 4-QAM
    bn  =       ( 2 * ( rand(1,Nbits) < 0.5 ) - 1 )  +  ...
        1i * ( 2 * ( rand(1,Nbits) < 0.5 ) - 1 )         ;
    bn  =  (1/sqrt(2)) * bn ;
    % delayed data
    bDn  =  [ zeros(1,D) bn ] ;
    dn   =  bDn ;
    dn   =  dn ( [ 1 : (length(dn)-D)] ) ;
    % upsampled data
    am  =  reshape ( [ bn ; zeros(K-1,Nbits) ] , 1 , Nbits*K ) ;
    % spread spectrum spreading code
    %  generated randomly, complex, unit energy
    pm  =       ( 2 * ( rand(1,K) < 0.5 ) - 1 )  +  ...
        1i * ( 2 * ( rand(1,K) < 0.5 ) - 1 )         ;
    pm  =  pm / sqrt(sum(abs(pm).^2)) ;
    pm = 1 ;  %  Set each user to have no spreading code
    % transmitter output
    fm  =  conv ( pm , am ) ;
    % Save matrices with the transmitted information for all users.
    bnall  =  [ bnall ; bn ] ;
    dnall  =  [ dnall ; dn ] ;
    amall  =  [ amall ; am ] ;
    pmall  =  [ pmall ; pm ] ;
    fmall  =  [ fmall ; fm ] ;
end
% The channel has Nusers inputs and Nantennas outputs.

% Initialize matrices for the channel information for all users and antennas.
%     The number of rows in gmall is Nantennas.
cmall = [] ;
```

```
small = [] ;
gmall = [] ;
% Use 6 points around centre impulse response from each channel impulse
% response to form channel matrix with zero padding on each end
c11 = [ zeros(1,2) C(1,1:6) zeros(1,2) ] ;
c21 = [ zeros(1,2) C(2,1:6) zeros(1,2) ] ;
c12 = [ zeros(1,2) C(3,1:6) zeros(1,2) ] ;
c22 = [ zeros(1,2) C(4,1:6) zeros(1,2) ] ;
for a = 1:Nantennas ,
    gmaall = [] ;
    for u = 1:Nusers ,
        fm  =  fmall(u,:) ;
        if ( ( a == 1 ) && ( u == 1 ) )
            cm = c11 ;
        end
        if ( ( a == 1 ) && ( u == 2 ) )
            cm = c12 ;
        end
        if ( ( a == 2 ) && ( u == 1 ) )
            cm = c21 ;
        end
        if ( ( a == 2 ) && ( u == 2 ) )
            cm = c22 ;
        end
        % channel output
        sm  =  conv ( cm , fm ) ;
        % Save matrices with the channel information for
        % all users and antennas.
        cmall  =  [ cmall ; cm ] ;
        small  =  [ small ; sm ] ;
        % Save matrix with received information
        gmaall = [ gmaall ; sm ] ;
    end
    gma  =  sum(gmaall,1) ;
    % Normalize the signal at the antenna output
    % to have unity power.  This simulates perfect
    % automatic gain control (AGC)
    gma  =  gma / sqrt(mean(abs(gma).^2)) ;
    % Save matrix with received information
    gmall = [ gmall ; gma ] ;
end
% Generate complex noise
% Initialize noise matrix
%     The number of rows in nmall is Nantennas.
nmall = [] ;
for a = 1:Nantennas ,
    nm      =        randn(1,length(sm))  +  ...
        1i *  randn(1,length(sm))          ;
    nm      =  sigmanm  *  nm ;
    % Save noise matrix
    nmall  =  [ nmall ; nm ] ;
end
% Find the receiver input.
%     The number of rows in rmall is Nantennas.
```

```
rmall  =  gmall + nmall ;

% Receiver
[Nrows,Ncolumns]  =  size(rmall) ;
Nsamples = Ncolumns ;

% Initialize matrices for the recevier information for all users.
umall = [] ;
Wmall = [] ;
Rmall = [] ;
dhatnall = [] ;
eall = [] ;
for u = 1:Nusers ,
    % Extract the signals for user u
    bn  =  bnall(u,:) ;
    dn  =  dnall(u,:) ;
    % Approximately pre-size the array sizes, for reduced processing time
    um                =  zeros(1,Nsamples) ;
    errors_recordedn  =  0 * bn ;
    dhatns            =  0 * bn ;
    % Initialize the adaptive filter
    Wm  =  zeros(Nantennas,Npoints_Wm) + 1i*zeros(Nantennas,Npoints_Wm) ;
    Rm  =  zeros(Nantennas,Npoints_Wm) + 1i*zeros(Nantennas,Npoints_Wm) ;
    Rm  =  [ rmall(:,1) Rm(:,[1:(Npoints_Wm-1)]) ] ;
    n = 0 ;
    for m  =  0 : (Nsamples-2) ,
        % regular FIR filter processing
        % obtain filter output
        um(1+ m)  =  sum ( sum ( Wm .* Rm ) ) ;
        % downsample
        if ( mod(m-DownSamplePhase,K) == 0 ) ,
            dhatn  =  um(1+ m) ;
            dhatns(1+ n)  =  dhatn ;
            % adapt the filter provided the delayed data exists
            if  ( 1 <= (1+ n) )  &  ( (1+ n) <= length(dn) )  ,
                % adapt the filter provided the delayed data is present
                if  ( dn(1+ n) ~= 0 )  ,
                    en  =  dn(1+ n) - dhatn ;
                    % errors_recordedn(1+ n-D) = en ;
                    errors_recordedn(1+ n  ) = en ;
                    Wm   =   Wm  +  mu * en * conj(Rm) ;
                end
            end
            n  =  n + 1 ;
        end
        % shift FIR filter state
        Rm  =  [ rmall(:,1+ m+1)  Rm(:,[1:(Npoints_Wm-1)]) ] ;
    end  % for m  =  0 : (Nsamples-2) ,
    % Save matrices with the received information for all users.
    umall  =  [ umall ; um ] ;
    Wmall  =  [ Wmall ; reshape(Wm,1,Nantennas*Npoints_Wm) ] ;
    Rmall  =  [ Rmall ; reshape(Rm,1,Nantennas*Npoints_Wm) ] ;
    dhatnall  =  [ dhatnall ; dhatns ] ;
    eall  =  [ eall ; errors_recordedn ] ;
```

```
end  % for u = 1:Nusers ,

% Display results

% Coefficients Nantennas by Nusers
% clf ;
figure(1) ;
for u = 1:Nusers ,
    Wm  =  reshape(Wmall(u,:),Nantennas,Npoints_Wm) ;
    for a = 1:Nantennas ,
        subplot(Nantennas,Nusers,Nusers*(a-1)+u) ;
        iWm = [ 0 : (Npoints_Wm-1) ] ;
        stem(iWm,abs(Wm(a,:)).^2,'o') ;
        % ylabel('Tap Energy, |wm|^2') ;
        % xlabel('Index, i') ;
        % stem(iWm,angle(Wm),'o') ;
        % ylabel('tap angle, angle(wm)') ;
        % xlabel('Index, i') ;
        if ( Nusers*(a-1)+u ) == 1 ,
            title('Coefficients (Nantennas by Nusers), W') ;
        end
    end  % for a = 1:Nantennas ,
end  % for u = 1:Nusers ,
% Learning curves for each user
figure(2) ;
for u = 1:Nusers ,
    errors_recordedn  =  eall(u,:) ;
    subplot(Nusers,1,u) ;
    i_se_db  =  0 : (length(errors_recordedn)-1) ;
    se_db  =  10 * log10(abs(errors_recordedn)+eps) ;
    plot(i_se_db,se_db) ;
    grid ;
    axis ( [ 0 (Nbits-1) -SNR_Rxer_in_dB 0] ) ;
     ylabel('Log Squared Error, (dB)') ;
     xlabel('Time index, n, (-)') ;
    if u == 1 ,
        title('Learning curves for each user') ;
    end
end  % for u = 1:Nusers ,
% Presume the second half of the learning curve is converged and find the MSE
% based on the values in the second half of the simulation.
iforavg  =  [ (fix(length(errors_recordedn)/2)) : length(errors_recordedn) ] ;
squarederrors  =  reshape( eall(:,iforavg) , 1 , Nusers*length(iforavg) ) ;
msedB  =  10*log10( mean( abs(squarederrors).^2 ) ) ;
disp(sprintf('The MSE after convergence for all users is %0.2f (dB).',msedB));
-----
```

Appendix C

Quartus$^{\circledR}$ II Verilog HDL Code and Quartus Block Diagrams

The following sections containt the Quartus block diagram files and Verilog HDL code that was used to distribute FPGA clock, program the FPGA for transmit signal generation, and sample incoming IF signals from AOR AR5000A radio receivers.

C.1 Quartus Block Diagrams

This section contains images of the Quartus block diagram files used to distribute the 10 (MHz) frequency standard supplied to the FPGA external clock input, program the FPGA for generating the four I/Q transmit signal pairs, and sample the four received IF signals.

C.1.1 FPGA Clock Distribution: Quartus PLL

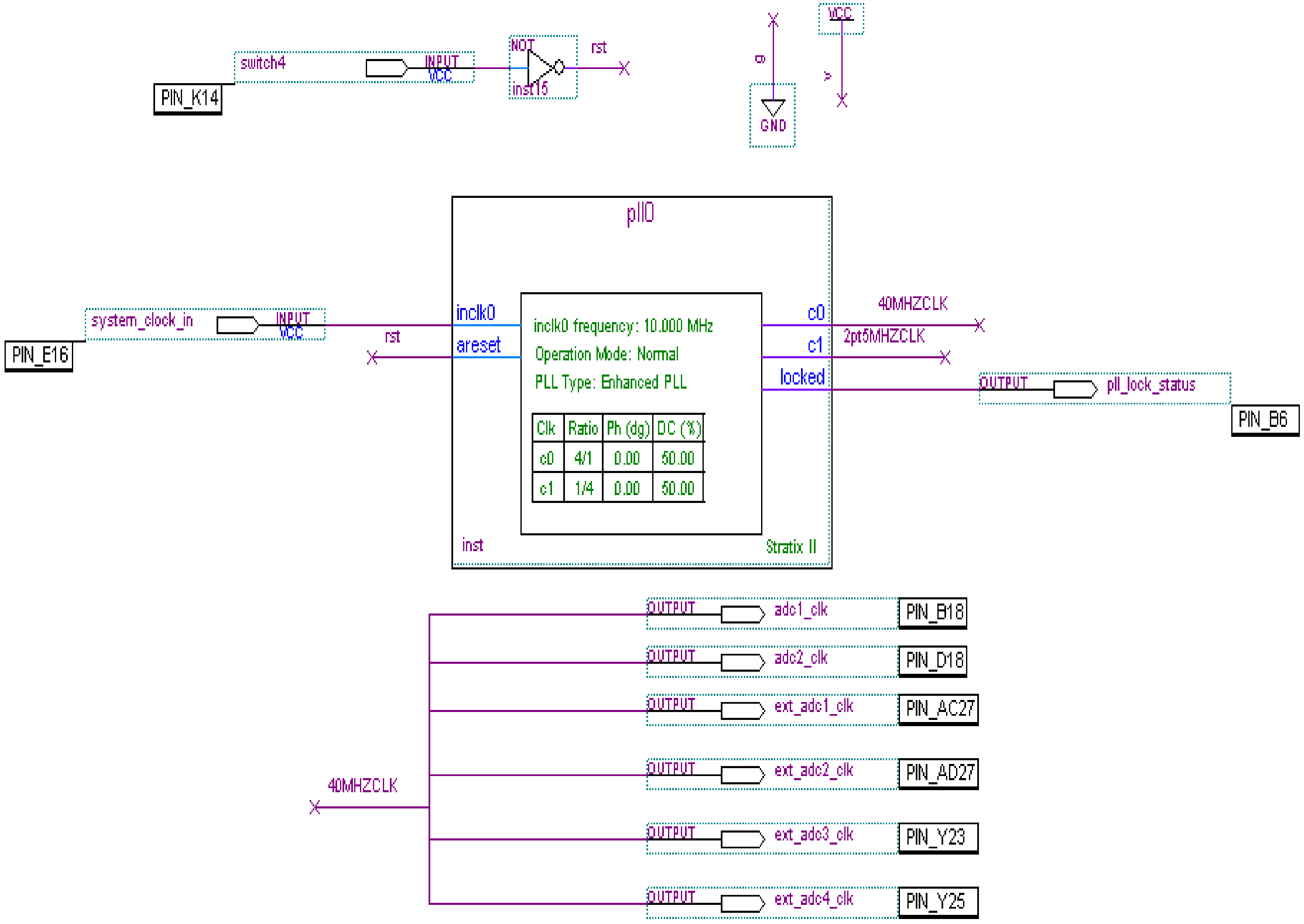


Figure C.1: Quartus PLL

C.1.2 Transmit Side: Polynomial Taps and LRS Generators

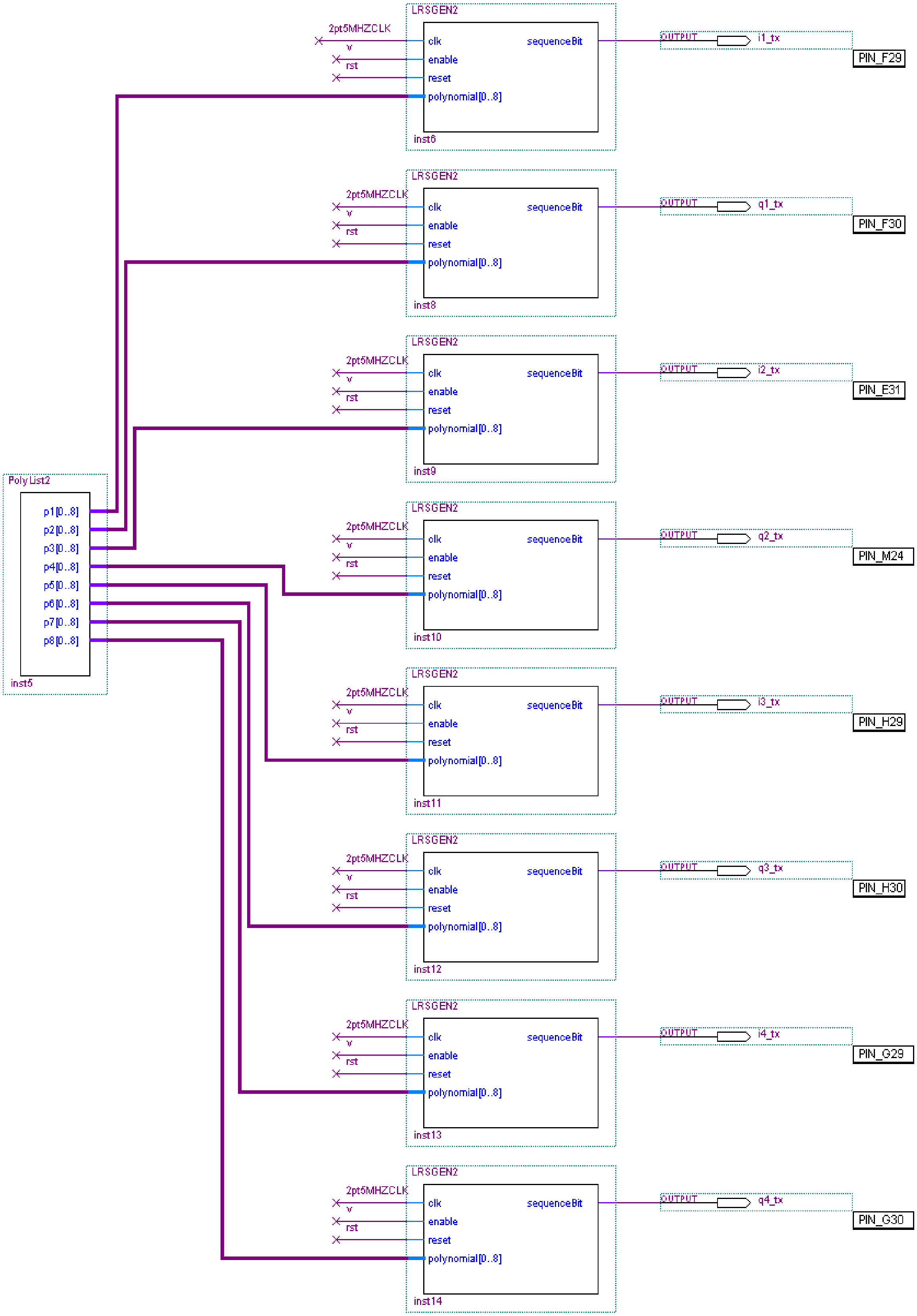


Figure C.2: Polynomial Taps and LRS Generators for I/Q Pairs

C.1.3 Receive Side: Quartus ADC Pin Assignments

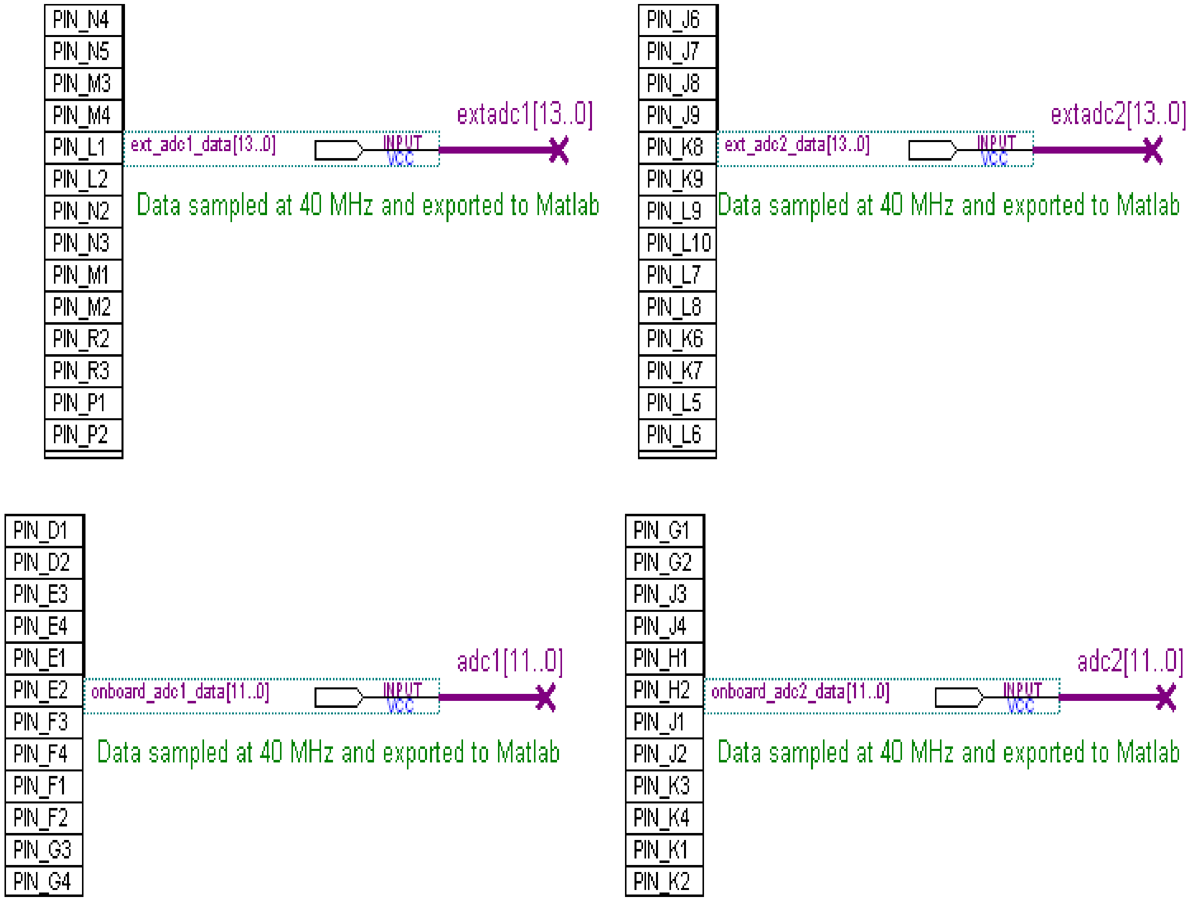


Figure C.3: Quartus ADC Pin Assignments

C.2 Verilog HDL Code

This section contains Verilog HDL code used to program the FPGA and generate the 4 I/Q transmit signal pairs.

C.2.1 Polynomial Taps

```
module PolyList2(p1,
p2,
p3,
p4,
p5,
p6,
p7,
p8);
output [0:8] p1;
output [0:8] p2;
output [0:8] p3;
output [0:8] p4;
output [0:8] p5;
output [0:8] p6;
output [0:8] p7;
output [0:8] p8;
/*
9-bit LRS polynomials (this is not an exhaustive list)
All are primitive with linearly independent roots and reciprocal
polynomial also has linearly independent roots as per Peterson (1961).
// assign pn = 9'b(LSB,..,MSB)
*/
assign p1 = 9'b100011111;
assign p2 = 9'b111000111;
assign p3 = 9'b101011011;
assign p4 = 9'b100111011;
assign p5 = 9'b101001111;
assign p6 = 9'b010000111;
assign p7 = 9'b000110011;
assign p8 = 9'b100100011;
endmodule
```

C.2.2 Linear Recursive Sequence Generator

```
module LRSGEN2(clk,
enable,
reset,
polynomial,
sequenceBit);
input clk;
input enable;
input reset;
input [0:8] polynomial;
output sequenceBit;
reg [0:8] statecurrent;
reg [0:8] temp;
reg sequenceBit;
always@(posedge clk or posedge reset)
begin
if(reset)
begin
sequenceBit = 1'b0; //initial sequence bit is 0
statecurrent = 11'b111111111; //intialize current state to all ones
end
else if(enable)
begin
sequenceBit = statecurrent[8]; //PN sequence bit is MSB in current state
                                   //vector
      temp = statecurrent; // temp variable to store current state vector -
                           // analogous to "shiftedstate"
/* statenext = [sequenceBit shiftedstate]; */
//input bit is addition modulo 2 of current state and polynomial vectors
statecurrent[0] = (temp[0]&polynomial[0])^(polynomial[1]&temp[1])
                       ^(polynomial[2]&temp[2])^(polynomial[3]&temp[3])
                       ^(polynomial[4]&temp[4])^(polynomial[5]&temp[5])
                       ^(polynomial[6]&temp[6])^(polynomial[7]&temp[7])
                       ^(polynomial[8]&temp[8]);
statecurrent[1:8] = temp[0:7];
  end
end
endmodule
```

Vita

Candidate's full name: Wesley Alexander Miller

University attended: University of New Brunswick, B.Sc.E., (2006)

Publications:

Wesley A. Miller, Brent R. Petersen, Bruce G. Colpitts, "Parallel MIMO Channel Measurement Architecture," *IEEE Proceedings of the 7th Annual Communications Networks Services and Research Conference (CNSR 2009)*, (Moncton, N.B., Canada), pp.437-439, 11-13 May 2009