2013

Implementations of Multi-Antenna Systems in a USRP Based SDR using Simulink

Wesley G. Rumble
West Virginia University

Implementations of Multi-Antenna Systems in a USRP Based SDR using Simulink

by

Wesley G. Rumble

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Electrical Engineering

Daryl Reynolds, Ph.D., Chair
Matthew C.Valenti, Ph.D.
Vinod Kulathamani, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2012

Keywords: SDR, MIMO, MATLAB, USRP, Wireless

Abstract

Implementations of Multi-Antenna Systems in a USRP Based SDR using Simulink

by

Wesley G. Rumble
Master of Science in Electrical Engineering

West Virginia University

Daryl Reynolds, Ph.D., Chair

Software Radio systems are one of the most versatile and cost effective systems for prototyping communications techniques in use today. Through the use of minimal hardware, typically only an RF front end, an SDR allows for as much of the signal processing as possible to be done via software. The benefits of utilizing software for signal processing in place of hardware are numerous. In some cases, like this work, parameters of operation may need to be dynamic or the objective of the work is academic research of new techniques. In some cases like these, swapping out costly signal processing hardware for signal processing software makes more sense. Additionally, the use of multi-antenna systems is becoming more and more common. In mid-2011, Mathworks released support for standard SISO (single input, single output) communications with the NI USRP (National Instruments Universal Software Radio Peripheral) but has yet to release support for any SIMO, MISO, or MIMO applications. In this work, we produce a testbed for proof of SIMO concepts and performance improvements in MATLAB and Simulink along with outlining preliminary work in implementing the MISO method of Alamouti STBC. The processes used to implement the discussed techniques will be outlined and discussed including results from testing with the USRPs in Simulink.

Acknowledgments

My journey at WVU has been one of many challenges and accomplishments. Through that I have made many connections with many great people, a list which would take too long to list. Despite that, a few stand above the rest in helping with my success. First and foremost, I would like to thank my advisor Dr. Daryl Reynolds. If it were not for his countless hours of discussing and directing, I would have never achieved the goals we set before me upon embarking on this degree. Thanks to his endless dedication and patience to both my degree work and the work of the greater project, I always had a highly knowledgeable leader at the helm to guide me in the right direction all along.

I would also like to say thanks to Dr. Matthew Valenti. In addition to serving on my committee, I also was able to take several of Dr. Valenti's courses. Through all of his courses, he strives to help his students make the connection between the classroom and the real world. In my case, he helped me to be able to both see and better understand the fundamentals described in class in my work, which was a tremendous help along the way.

While MATLAB/Simulink is a very well documented software package, there will always be problems that come up that support can't fix. When I hit those blocks along the way, I was able to find help from Mike McLernon, a Development Manager at Mathworks who has served as primary contact for all inquiries involving Mathworks support package for the USRP. I would like to thank him for his numerous emails throughout the last year through which he has expedited the development of this work with excellent support and incredible communication.

Last but certainly not least, I'd like to thank my family. My parents have always pushed me to work at my highest potential and been there to support me at both the high and low points of my academic career. Also, without the prayers and support of both of them along with my wonderful wife Stacy, I wouldn't have had the prayerful and emotional support to drive through the tasks put before me to finish this work and degree.

Contents

List of Figures

Notation

We use the following notation and symbols throughout this thesis.

$USRP$	:	Universal Software Radio Peripheral
SDR	:	Software Defined Radio
$SIMO$	:	Single Input Multiple Output
$MISO$	:	Multiple Input Single Output
$MIMO$	:	Multi ple Input Multiple Output
EGC	:	Equal Gain Combining
SD	:	Selection Diversity
MRC	:	Maximal Ratio Combining
$MIMO$	:	Multiple Input Multiple Output
$STBC$	:	Space Time Block Code
ISI	:	Inter Symbol Interference
BER	:	Bit Error Rate
PLL	:	Phase Locked Loop

Chapter 1

Introduction

In an ever growing face paced wireless world, the need for affordable and effective means of testing new and improved transmission methods is a never ending one. Due to the high cost of RF equipment and the limitations that fixed parameter hardware impose, the idea of a highly dynamic system for testing new methods presents itself as highly desirable. In 1984, a team at E-Systems (now known as Raytheon) created what they coined as a Software Radio. Their original model was a digital baseband receiver with software covering the demodulation and filtering along with array processors for accessing shared memory.[1] From that original receiver design to the first transceiver developed in Germany in 1988 [2] and beyond, the world of software radio has never looked back. Simply put, the idea of software radio revolves around minimizing the amount of hardware in a system and replacing all removed hardware with a software equivalent. Typically, the flow of an SDR is as follows. Data is read in through and RF front in, processed through an analog to digital converter (ADC) and lastly through an FPGA prior to being fed out of the radio and into the computer. Most SDR interfaces communicate from radio to computer/software host via USB or ethernet.

When it comes to academic work in this area, most work is commonly done involving the USRP, created by Ettus Research [3], or the WARP produced by Rice University.[4] [5] In mid-2011, Mathworks released support for standard SISO (single input, single output) communications with the NI USRP (National Instruments Universal Software Radio Peripheral) but has yet to release support for any SIMO, MISO, or MIMO applications. For this work, a testbed for proof of SIMO concepts and performance improvements in MATLAB

and Simulink was produced. In addition to that, preliminary work with implementing the MISO method of Alamouti STBC was also performed.

1.1 Project Timeline

1.1.1 Early Work in GNURadio

My work on this project has been very dynamic. The initial goal placed before me was to implement SIMO, MISO, and eventually MIMO techniques within our USRP based SDR testbed. In addition to that, I was tasked with a few other smaller requested tasks for sponsor demonstrations. When assigned to this project, I was directed to use the NI USRP N210's purchased for the project along with the open source GNURadio SDR block set to implement these methods. The first step was to work through the simple SISO analog voice transmission and digital file transfer (text files). Following that it was decided to implement a full duplex real time streaming video transfer. This part of the project took a bit of time, the better part of two months, but it too was successfully completed. Research into the work being done in GNURadio proved that SIMO,MISO, and MIMO techniques had been successfully implemented to date [6] [7], so reproducing these methods presented itself as a valid first step. At this point, I moved on to implementing 1×2 EGC with the USRPs in GNURadio. This proved to be fairly straightforward with the use of the Ettus Research MIMO cable, a proprietary cable distributed by Ettus to synchronize the operation of two USRP units.

It was not long after completing the EGC SIMO work in GNURadio that I was permitted to attend the GNURadio conference in Philadelphia, PA. It was at this conference that we learned of Mathworks' recently developed and released support for SDR operations within MATLAB/Simulink. This came as somewhat of a surprise because it was the first we'd heard about this, but it also was very intriguing because of my previous experience with MATLAB and the excellent support Mathworks provides for it. One major problem that was encountered with GNURadio was that while it was fairly self sustained, the support for it was purely user based and minimal. The idea of having the vast Mathworks support base

along with the apparent simplicity of adding other colleagues MATLAB/Simulink work on top of my radio system served as enough motivation to switch software platforms for the project.

1.1.2 Work in MATLAB/Simulink

Once the change to MATLAB/Simulink was made, there was another period of time spent familiarizing myself with the USRP support provided from Mathworks. This meant working through the provided demos and then recreating the analog sound and digital data transfers which were the initial demos developed in GNURadio. This did not take as long as it it did in GNURadio thanks to the solid documentation and support provided with the USRP support from Mathworks. The next step in my work was to recreate the full duplex streaming video demo, which is when the issue of synchronization came to the forefront. A similar video transfer demo was created in Simulink and proved to transmit streaming video but the image output at the receiver was not reconstructed correctly from bit to image form. It became apparent that there was an issue with time synchronization between transmitter and receiver. The received image was segmented into four pieces, nearly perfect quadrants to be exact. The issue was that quadrants one and three and two and four had been flopped in some manner, which was inconsistent on an iteration to iteration basis.

Initial attempts to synchronize transmitter and receiver were crude and proved to be unsuccessful. Although it was known fact that the issue was triggering the receiver to record and display the output at the appropriate time. The issue encountered when attempting to do this was there was no accurate and dependable way to easily find the beginning of an image sequence. It was at this point that I reached out for support from Mathworks on the issue and learned of their involvement in the area of synchronization correction, which is discussed in the next chapter.

Initial contact with Mathworks about the encountered synchronization issues revealed that they too had encountered these issues when they tried to start sending streaming data. My contact there, Mike McLernon, informed me that they were preparing to release a QPSK synchronization demo for the USRPs. Thankfully, he was kind enough to send me

a prerelease copy of the demo to start working through. This demo was a self supported simulator which didn't actually use the USRPs but modeled the rest of the system to emulate the USRP system. This proved to be very helpful because it gave me an insight into the development of their support and gave me a few extra weeks to work with the synchronization blocks minus the USRPs prior to the release of the official package which uses the USRPs. Once the release came out I was able to proceed forward with the data transfer and start working on the SIMO,MISO, and MIMO techniques.

Once the issue of synchronization was under control (discussed in detail in chapter 5) the work on implementing SIMO techniques was started. While working in GNURadio I was able to implement EGC successfully and proved its performance improvement over that of the standard 1×1 SISO transmission system. Due to this, when I started working on implementing the SIMO methods in MATLAB/Simulink, it seemed fitting that I would start with EGC. Following that I moved on to SD and finally MRC before preparing a document tracking the results of that work. The body of that document serves as the primary body of the SIMO Testing chapter.

The implementation of the SIMO methods served as the brunt of my work on this project. Despite the fact that the major synchronization issues were dealt with once the MATLAB support was received, there were still several issues encountered along the way. This was to be expected, but the delays they caused could not be predicted, obviously, so this portion of the work took a little longer than expected. Despite this, the SIMO work was completed in time to start work on the MISO Alamouti STBC method. The Alamouti STBC method work is the last stage of my involvement on this project. I have worked extensively trying to include the MATLAB/Simulink provided Alamouti encoder and combiner blocks within Simulink into the Mathworks provided QPSK sync demo. The biggest issue with this part of the work was the need for a channel gain estimate, which will be discussed more in detail later on.

This order of events serves as a comprehensive list of each stage of my work throughout my time working on this project. Whist working on these implementations, many new things were discovered along with a lot of new inadvertently learned skills both MATLAB wise and wireless communications wise as well.

1.2 Thesis Outline

In this thesis, we take a look at various stages and accomplishments of an SDR implementation. More specifically, the focus is on multiple antenna transmission techniques and their performance in the SDR testbed. The goal of these works was to prove the performance gains of the discussed methods and to serve as the groundwork for further multiple antenna implementation research.

Considering the ever growing market for SDR products and research, the addition of MATLAB/Simulink functionality and support for the USRP presents itself as a highly useful tool. Due to the newness of this support, functionality for multiple antennas doesn't currently exist. The addition of this functionality to the current support would be very beneficial to the USRPs capabilities in MATLAB and Simulink. As stated, the initial support only supports SISO (single input, single output) communications, and its lack of support for further work with SIMO, MISO, and MIMO techniques provided the inspiration for this work. Based on an exhaustive literature search and communication with the USRP support developers at Mathworks it is believed that this work is the first successful SIMO/MISO implementation with the USRP and MATLAB/Simulink.

Prior to discussing the implementation work, we will first take a look at the theory behind each method. All three SIMO methods and the one MISO method implemented will be discussed and illustrated. While the theory of these methods is far from being new or novel, their implementation in a testbed such as this is, and will therefore be discussed in detail. One major issue faced in this work was the variation in the wireless channel and its effect on the performance of this system. Initial work showed there was a lack of synchronization of the data stream within the receiver. The received signal was broken, with the beginning of the output data vector shifting spatially on an iteration by iteration basis. This leads us to the final segment of the preface for our work; synchronization. In the synchronization chapter there will be a detailed outline of the Mathworks provided QPSK synchronization [8] receiver and the modifications I had to make to it for it to perform as needed to output BER results.

After this sequence of outlines and details as to the basis and support of the testbed, we

will move on to the actual testbed configuration and the methods' implementation within said testbed. The specifics of the aforementioned multiple antenna methods' implementations will be described and their results discussed. In addition to the Simulink models executed to generate the output data, some MATLAB scripts were also needed to perform the necessary BER calculations. Those scripts will be discussed in this section and will be attached within the appendices.

Lastly, the MISO work I have worked on will be outlined and analyzed. The attempted MISO method, Alamouti STBC, has proved to be more difficult than previously expected. While it's feasibility has not been completely ruled out, after working on it intensively, the issues which pose a major hindrance to the implementations success became quite apparent. The work which was done and the issues encountered will be discussed in detail. Following that an outline of the proposed future work on this grant will be discussed along with the conclusions drawn from this work as a whole.

Chapter 2

Overview of Multiple Antenna Techniques

The wireless channel is arguably one of the most inconsistent variables in the world of communications. The level of variation a given wireless channel can experience is virtually unlimited dependent on the environment in which is is contained. Due to this, quality of transmission in a given wireless system can be either negligibly effected or seriously damaged prior to its reception at the receiver. The most prevalent cause of signal damage in wireless is fading. Fading can be caused by many factors, but is generally defined as deviation of the attenuation of a signal over a given propagation medium. This can be due to reflections of the signal while traveling through the channel along with the inevitable signal attenuation, delay, and phase shift of the wireless channel. Fading can be classified as either slow or fast fading, depending on the rate in which the channel is changing. For this work, a slow fading channel was assumed following real time monitoring of the channel during testing. A common method for improving communication performance in the wireless field today is the practice of using multiple antenna techniques at the transmitter and/or receiver. This can be done using a variety of methods, which are classified as one of the following: SIMO, MISO, or MIMO. For this work, the primary focus was on implementing SIMO methods. In addition to that, some time was spent on implementing MISO methods; Alamouti space time block code, to be specific. The details of the implementation of these methods will be described later, while this section serves merely as a theoretical outline of these methods.

2.1 SIMO Techniques

As stated, one well documented method for combatting the effects of fading is to add receiver diversity through the use of SIMO techniques. The use of multiple receivers/antennas allows for multiple versions of a transmitted signal to be received via multiple paths through the wireless channel. By accounting for these multiple paths in different ways, depending on the SIMO method, the performance of a given system is bound to improve. When it comes to techniques such as these, there are three common options used in practice. These methods are called selection diversity (SD), equal gain combining (EGC), and maximal ratio combining (MRC), in order of lowest performance gain to the highest. It is well known that all three of these SIMO techniques provide full diversity.

2.1.1 Selection Diversity

The first method used, selection diversity, is the most straightforward of the SIMO methods. Simply put, in a SD receiver, the two received signals are taken and compared to each other. The signal with the highest received signal channel gain is chosen as the signal to be demodulated for analysis. The flow of this method's Simulink implementation is shown in figure 2.1.

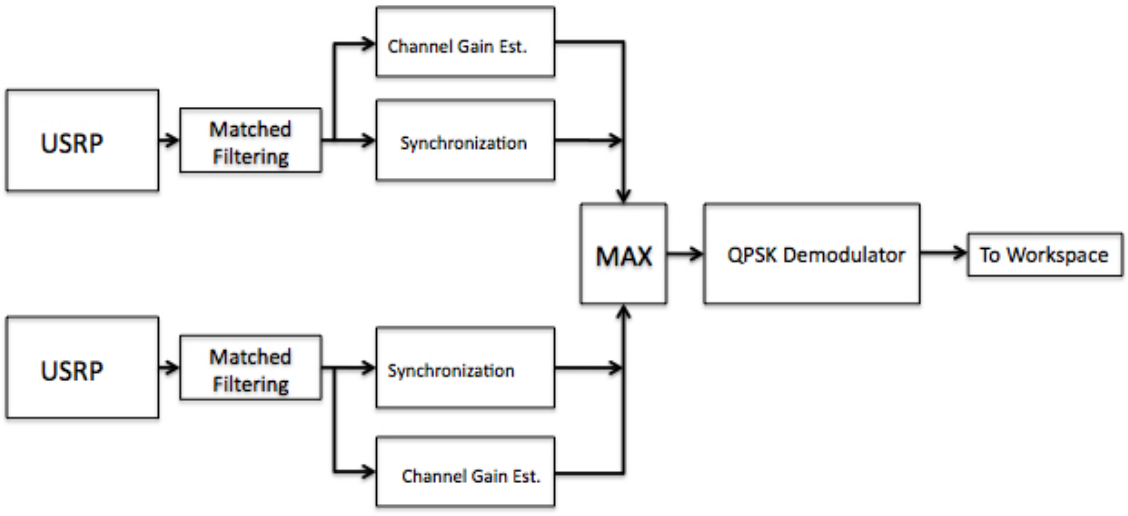


Figure 2.1: Flowgraph of SD Receiver Simulink Model

2.1.2 Equal Gain Combining

The second method examined in this work was equal gain combining. This method does not take into account the gain of either receivers channel but instead adds the matched filter outputs prior to demodulation. This method maximizes the QPSK symbols' matched filter outputs distance from the origin which makes the decision for the QPSK demodulator much easier. The flow of this method's Simulink implementation is shown in figure 2.2.

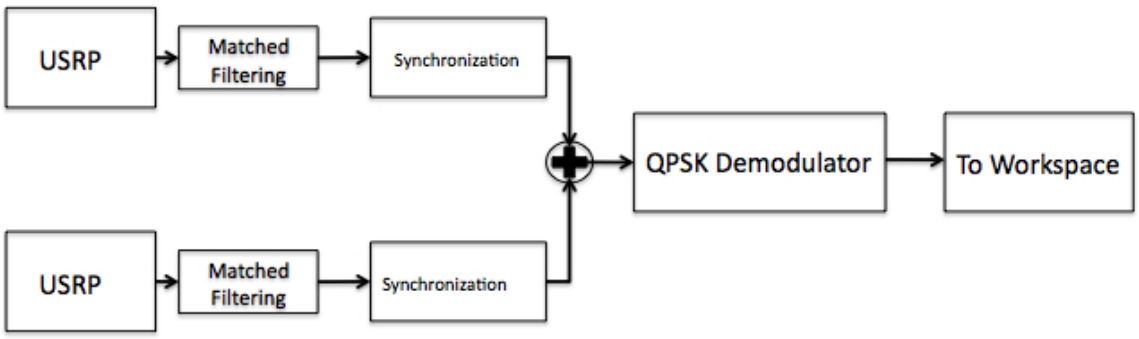


Figure 2.2: Flowgraph of EGC Receiver Simulink model.

2.1.3 Maximal Ratio Combining

The third and final SIMO technique implemented for this work was maximal ratio combining. This method effectively takes SD and EGC and combines their methods to both boost the signals' strength but weigh the signal streams based on the gain of their respective channels as well. In this method, the channel gain of each stream is determined. Then each signal is multiplied by its respective gain. Finally the signals are added prior to demodulation. The flow of this method's Simulink implementation is shown in figure 2.3.

2.2 MISO Techniques

Another common technique for improving communication performance in the wireless field today is the use of multiple transmit antennas. This case is slightly more complex than the SIMO methods discussed. This is due to the fact that when you have two or more

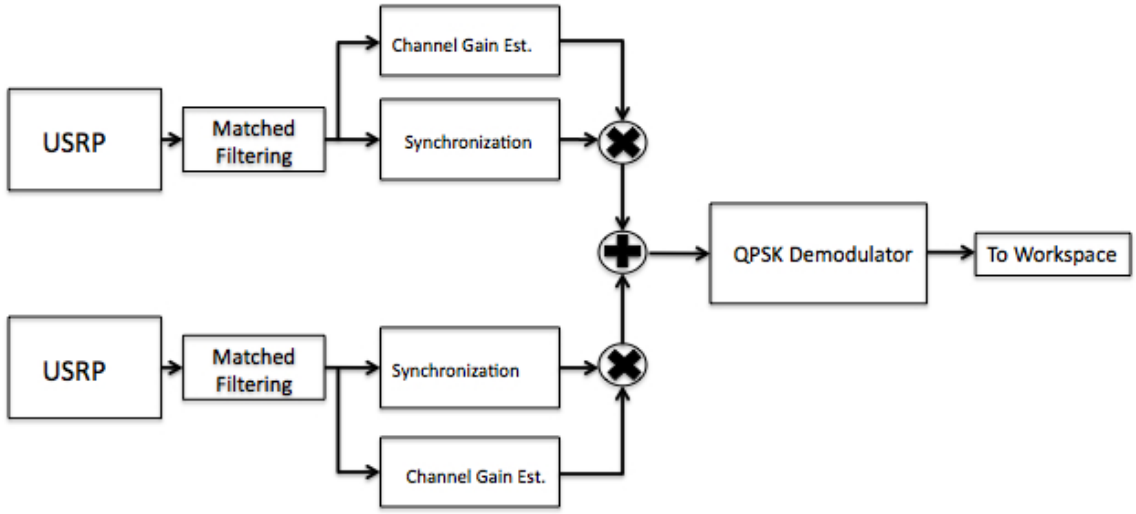


Figure 2.3: Flowgraph of MRC Receiver Simulink model.

transmitters all transmitting on the same frequency at the same time, something has to be done to ensure that the receiver can distinguish between the incoming signals. In the case studied in this work, the Alamouti STBC method, an orthogonal space time block coding method is used.

The Alamouti method of STBC uses an orthogonal mixing matrix to transmit two symbols per time slot in order to achieve a full rate and diversity. The transmitted data stream is multiplied by a coding matrix just before being transmitted. The Alamouti coding matrix looks like this:

$$C = \begin{pmatrix} c_1 & c_2 \\ -c_2^* & c_1^* \end{pmatrix}$$

The Alamouti code is a full rate code which produces a BER performance on the same level as MRC. The end result of this coding method is that using an accurate channel gain estimate at the receiver end, there will be perfect orthogonality between symbols after receiver processing. In this case, each symbol transmitted is transmitted twice, once per timeslot. It should be noted that this form of STBC is the only orthogonal STBC that is full rate, or rate-1. This means that the system can achieve full diversity gain without sacrificing data rate. A flowgraph of this method is shown in figure 2.4.

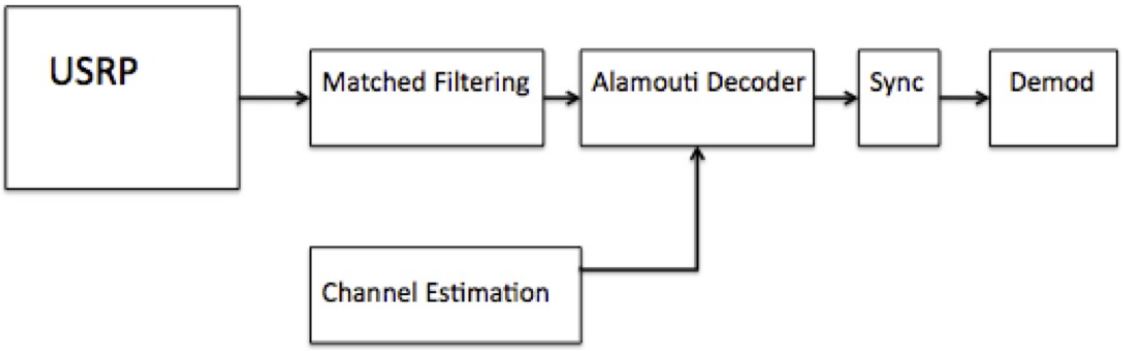


Figure 2.4: Flowgraph of MISO Receiver Simulink model.

The key with this method is an accurate channel gain estimate for decoding. The vitality of this will be discussed in chapter six, as it was a major obstacle encountered during this portion of my work.

2.3 Summary of Multiple Antenna Methods

Given the lack of prior implementations of multiple antenna techniques with the USRP in MATLAB/Simulink, these simple yet effective methods presented themselves as the first step in producing multiple antenna implementations within our testbed. A successful implementation of these methods should prove to lay the groundwork for numerous more complicated and beneficial methods down the road. Also, due to the newness of the support, these methods presented good modifications to the QPSK receiver from Mathworks which helped me learn how they work and how they could be tweaked to support these multiple antenna methods and future work in the area as well.

Chapter 3

Synchronization

3.1 Synchronization Overview

As previously mentioned, during initial SISO 1×1 testing with the USRPs in MATLAB/Simulink, the received data streams were not synchronized in that they could not detect the beginning of a packet of data and would commence data recording in the middle of a packet. This became evident during testing of our video transfer demo we were asked to prepare. The sequence of image frames of the video were transferred in real time but due to this lack of synchronization the output image was segmented into quadrants like a puzzle. Also, the issue of synchronizing data streams between the two receiver USRPs in the SIMO method work became apparent as well during initial testing.

Upon noticing these issues, work was commenced on creating some sort of synchronization segment of the receiver. During this initial stage of testing, I was still working with sending images taken from webcam video. The aim of my first few attempts at synchronizing consecutive images was to concatenate the image with a segment of ones or zeros which the receiver could seek out to set as the beginning of an image and therefore begin recording the output data stream at this point. As simple as this theory may sound, it did not work due to the variance within the wireless channel. Regardless of the length of the portion of ones or zeros concatenated at the beginning of each image, the receiver could not accurately distinguish the vector of ones or zeros due to the fact that the initial vector was not received intact and therefore undetectable. Thankfully I did not spend much time trying this method

because Mathworks had encountered the same problem and were preparing a full QPSK synchronization model for release.

Once these synchronization issues were uncovered, I decided to utilize one of the biggest reasons we switched software packages for this work; Mathworks' full service support. I posted my problem on the Mathworks message boards and within hours had an email from one of their production managers who works on the USRP support. He informed me of their work on a QPSK synchronization model [8] forwarded me a copy of the QPSK demo prior to their release, as mentioned in the project timeline. While this version of the demo was a self contained simulator with no USRP blocks, it was a little less condensed than the official release and gave me a good look into the receiver itself and how it works. This proved to be useful when some modifications were needed to be made to the receiver in order to produce the performance metrics needed for analysis.

Changes to the models were needed for access to the receiver in order to access the matched filter outputs of the receiver for channel gain measurements. Also, access to the synchronized data streams prior to demodulation for the accumulation and equalization stages for EGC and MRC was needed. Some minor modifications were needed for the overall structure of the QPSK receiver demo model due to the addition of the second USRP receiver for the 1×2 cases as well. In the initial model the synchronization receiver was set up as an enabled subsystem which requires a trigger signal from the USRP to start recording data as to not record bad data prior to the arrival of information from the USRP. The problem encountered was that the addition of the second USRP meant that the receiver needed to only operate when both USRPs were outputting data, otherwise the receiver would record bad data from one USRP while the other was recording or vice versa. Through the use of a logical block in Simulink the receiver was set up to only record when both USRPs were recording data which enabled the rest of the testing to be conducted successfully. The other major change to the model (block diagram shown in figure 4.1) was that instead of descrambling the output message and then selecting only the 105 bits of each packet which contained the "hello world" message from the transmitter, the entire received data vector was output to the workstation for analysis.

3.2 QPSK Synchronization Receiver

While the receiver block diagram shown in figure 4.1 may look somewhat complicated, the process set forth by the Mathworks support's receiver is actually very straightforward and more or less adaptable to one's needs based on what they are trying to do. In my case, as previously mentioned, the only modifications I had to make to this receiver was to remove the last block prior to output that selected the 105 "hello world" bits that are output to the workspace in the original demo as a visual output. This modification was made within the "Data Decoding" block in figure 4.1. As for the model itself, the flow of how it works and fixes the problem of synchronization is as follows.

Prior to discussing the operation of the receiver, it needs to be reiterated that this model is contained within a larger enabled subsystem which is triggered by the port on the USRP block that acts as a data trigger when the USRP is actually recording and outputting good data and not noise. This enables the receiver to only record, resynchronize, and output good data as well. Once the receiver is triggered and this section of the receiver is enabled, the data is first read in and broken into frames. Following this stage, the signal is routed through an automatic gain control block which fixes the amplitude of each sample. This block was only used in the 1×1 SISO case and was removed for the 1×2 SIMO cases so as to not effect the improvements of the SIMO methods. The next step in the receiver is the raised cosine receive filter, which just serves as the inverse of the raised cosine transmit filter used in the transmitter model to shape the transmitted signal in an effort improve performance by reducing inter-symbol interference (ISI). From this point on serves as the actual synchronization steps in the receiver.

The synchronization steps of the receiver are broken into four main parts, each of which contain smaller subsystem parameters which can be accessed by opening the block within Simuilnk. Those stages of the receiver are not depicted here for the sake of conciseness however they will be discussed. The first step in the process is the course frequency compensation. This block finds the frequency offset between receiver and transmitter and corrects the frequency offset before passing it along to the next step. The next stage in the receiver is the fine frequency compensation. The first step of this block is an "unbuffer" block which

breaks the vectors down and reads each value through as scalars. Similar to the course compensation, this step finds the phase offset instead of the frequency offset. The phase offset is monitored and corrected through the use of a phase locked loop, or PLL. The PLL runs continuously and passes the signal on to the next stage. The third stage of the resynchronization is the timing recovery block. Within this stage the signal is routed through a timing recovery PLL. This stage implements a MATLAB based scalar timing recovery which passes the data through the timing recovery PLL and then passes it back through a modified buffer in order to output the data back in vector form for the last stage of the model. The fourth and final stage of the data decoding block. This block is the most complicated but provides a tremendous amount of help in the process. Before discussing the operation of this block, it is important to note the one key feature of the transmitter model supplied by Mathworks that needs to be discussed. The only difference between my initial transmitter model and the provided one from Mathworks was that they add a double Barker code, 26 bits in length, to the beginning of each packet of data. Similar to my attempt of adding a vector of zeros or ones to each packet, this method works a lot more effectively. This brings us back to the data decoding block of the receiver. The first stage of this block uses the Barker code to compare against the received data and calculate the delay of the channel. Once this is complete the system aligns the data with the detected frame head and resolves any possible phase ambiguity caused by the fine frequency compensation block. Lastly, this stage demodulates the data and outputs it to the workstation by way of a column vector.

When the system is used as described, it performs quite well. The issue of synchronization is dealt with and the system outputs data that can be analyzed for BER performance. This system was modified a few times to change the message being transmitted in an attempt to do some image transfer again but for the sake of implementing the SIMO/MISO methods, the original message provided with the Mathworks QPSK demo was used.

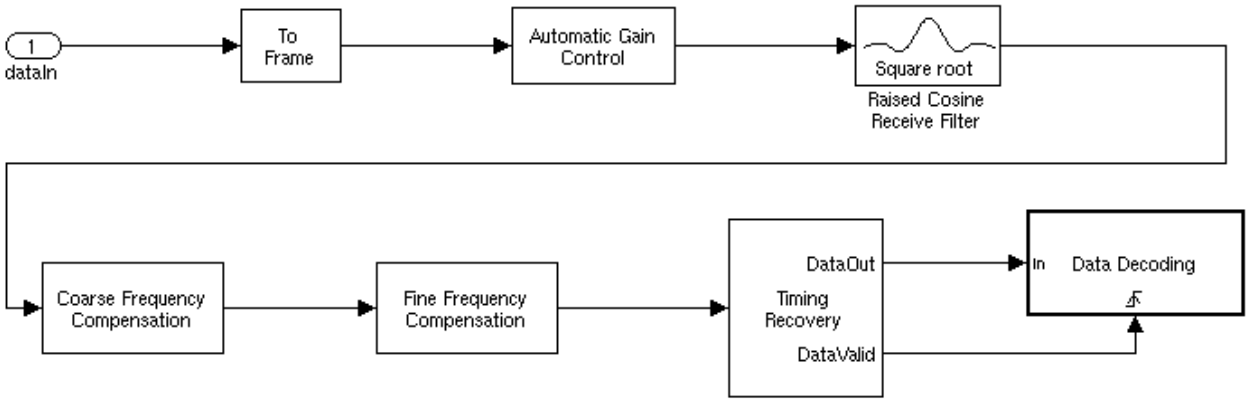


Figure 3.1: Screenshot of Simulink QSPK Sync Receiver Block Diagram

Chapter 4

SIMO Testing

4.1 Testbed Description

In this section, both the USRP testbed setup and software used are described. First the focus will be the USRP and its specifications [9]. The details for our hardware configuration and initialization protocols will be discussed along with the challenges of adding a second USRP in MATLAB/Simulink. Then the Simulink models and MATLAB scripts used to produce the results will be discussed.

4.1.1 Hardware - USRP

The USRP is a hardware platform designed for rapid design and implementation of flexible SDR systems. [3] Chosen because of its high level of customization at a low cost in comparison to other options on the market [3] [10], the USRP has proven to be a good choice for our work. Containing only an RF front end complete with ADCs and DACs, it allows the rest of the signal processing to be within Simulink. Specifically used in this work was the USRP N210, which is part of Ettus' highest class of USRPs. The general work flow of each USRP, is as follows. The WBX daughterboard was used for this work as the RF frontend to receive RF signal which is then converted to IF (and vice versa for the TX case). The WBX daughterboard is a transceiver rated for operation from 50-2200 MHz with a peak output of 100 mW and a noise Figure of 5db [5]. At this point the signal reaches the motherboard, which where it is sampled by an ADC. The N210 is equipped with

dual 14 bit ADCs which operate at 100 MSPS, and dual 16 bit DACs which operate at 400 MSPS. From this point it is converted to baseband by a digital downconverter (DDC) on the onboard FPGA, the Spartan 3A-DSP 3400. Once the signal is at baseband, it is transmitted to the host computer via a gigabit ethernet connection. The transmitter structure is simply the reverse of the receiver which uses a digital to analog converter (DAC) and a digital upconverter (DUC).

Each USRP was equipped with one VERT900 omnidirectional whip antenna which is a dual band antenna rated for 824-960 MHz and 1710-1990 MHz. Due to these hardware specifications, a frequency of 1.8 GHz was chosen for all experiments.

Ettus also has an optional MIMO cable which is designed to link a master and slave USRP together to communicate to the base computer via one ethernet connection. Despite the lack of support for this cable or the use of multiple USRPs with Simulink at this point, testing to see if the use of this cable was feasible was conducted. While the slave USRP was able to successfully receive data, the output of the channel gain estimation showed that the degradation of the signal due to the channel gain was much higher than that of the master. Once the MIMO cable was tested a second configuration which used separate ethernet cards for each receiver was used. The channel gain estimates for either USRP were much closer in magnitude in this case and the BER results were better. As a result of this, the independent dual ethernet configuration was used.

One last obstacle that was encountered in setting up the dual receiver USRPs was that the USRP units come preprogrammed with the default IP address of 192.168.10.2. This goes in connection to the default ethernet card IP of 192.168.10.1. The problem encountered was that to use two different ethernet ports one would have to have a different IP address, and in turn so would the corresponding USRP. The Mathworks support for the USRPs does not provide means to change the IP of a USRP, so a USRP net burner file from Ettus' UHD support package was needed to manually change the USRP IP address. Once this was done, the dual receiver USRPs were able to be used simultaneously.

4.1.2 Software - Simulink

The basis for the software used in this work was the provided QPSK Synchronization demos from Mathworks for Simulink [8]. Those demos provide a self contained synchronizing QPSK receiver, so no changes were needed in relation to the synchronization level of the receiver itself. In addition to the modified QPSK demo models from Mathworks, some additional MATLAB scripts (described in the next section) were needed for the alignment (spatial synchronization of received data streams), and resizing (to allow comparison to transmitted data stream) of the received signals. The QPSK transmitter and receiver include the use of Barker Codes to assist in the synchronization of the received data which was used in these scripts to align the beginning of the two received data streams for the SIMO cases. During testing it was observed that immediately following the initialization of the receiver that both USRPs recorded an inconsistent amount of bad data, typically on the order of 5000 bits. To remove this variance from the equation it was decided to remove all recorded bits prior to the first successful recorded Barker code in the received signal.

4.1.3 Software - MATLAB Scripts

In order to perform the needed BER calculations, a few MATLAB scripts had to be created for post reception analysis. Once the receiver model executed each time during testing, the output data was saved to the MATLAB workspace for post-processing. Each SIMO technique (SD, EGC, and MRC) had their respective calculation script, each with its own minor modifications for their specific case. All MATLAB scripts used for these calculations can be found in appendix A.

Because the addition of the received signals was performed within the Simulink model for the EGC case, it did not require additional steps beyond the general flow discussed below. The general flow of each script was as follows. The received data stream was read in, and the first successfully received Barker code was sought out and found. All bits in the received stream prior to that code's beginning were removed. From that point, the received data was compared to the transmitted stream.

In the SD implementation's case, there were a few steps of processing prior to the afore-

mentioned BER calculation which were required for SD. The two received signal stream's channel gain was measured and the stream which encountered the stronger channel was chosen for calculation. In order to do this, the QPSK demodulation had to be removed from the Simulink model and added to the MATLAB script so it could be performed after this selection took place. From there the stream was passed through the previously discussed calculations.

As for the MRC case, it was slightly more complicated. This case, similar to the SD case, also calculated the channel gains each signal encountered. From there, the received signal streams were multiplied by their respective channel gains so as to weight the signals based on the quality of their respective channel, as is the procedure for MRC. As was the case in SD, in MRC the QPSK demodulator was again removed from the Simulink model and added to the MATLAB script following the weighting and addition of the streams. The final step of this script performed the mentioned addition of streams (similar to EGC) and the BER calculations were performed.

4.2 Results

This section outlines the testing procedures used and the results they provided. The goal of these experiments was to measure and analyze the bit error rate (BER) performance of the 1×1 and 1×2 USRP testbed configurations as a function of the transmit gain. For these experiments, the transmit gain is determined by the transmitter gain parameter within the Simulink USRP TX block. This parameter varies the gain of the analog amplifier within the USRP unit. Testing was also done at three different distances to see where the most dynamic range of BER performance was achieved. Testing was conducted on all 3 SIMO options simultaneously in order for accurate comparison of the three methods performance to each other.

4.2.1 Theoretical Performance

Prior to working on producing results from the SIMO USRP implementations, a set of theoretical measurements was needed for comparison's sake. The results generated and

plotted in this section cannot be compared directly to the results from the USRP testing, but the characteristics of the methods' curves can be directly compared. The theoretical measurements could not able to be directly compared to the USRP testing results because the theoretical BER results are plotted against SNR (E_b/N_o) while the USRP BER results are plotted against transmitter gain values. This procedure was used because the actual SNR for the USRPs could not be accurately calculated during testing because the transmitter parameters only allow for variation of transmitter gain. Having that said, the curves plotted in Figure 4.1 were generated using the following equations. The exact error probability for a 1×1 system using BPSK modulation operating in a Rayleigh flat fading environment is [11]

$$P_e = \frac{1}{2}\left(1 - \left(1 + \frac{1}{E_b/N_o}\right)^{-1/2}\right). \tag{4.1}$$

Similarly, the error probability for selection diversity can be written [11]

$$P_{e,\mathrm{SD}} = \frac{1}{2}\left(1 - 2\left(1 + \frac{1}{E_b/N_o}\right)^{-\frac{1}{2}} + \left(1 + \frac{2}{E_b/N_o}\right)^{-\frac{1}{2}}\right). \tag{4.2}$$

The next method tested was equal gain combining, whose bit error probability is [12]

$$P_{e,\mathrm{EGC}} = \frac{1}{2}\left[1 - \frac{\sqrt{\frac{E_b}{N_o}\left(\frac{E_b}{N_o} + 2\right)}}{E_b/N_o + 1}\right]. \tag{4.3}$$

Lastly, the bit error probability for maximal ratio combinging is [13]

$$P_{e,\mathrm{MRC}} = p^2(1 + 2(1 - p)), \tag{4.4}$$

where p is defined as

$$p = \left(\frac{1}{2} - \frac{1}{2}\left(1 + \frac{1}{E_b/N_o}\right)^{-\frac{1}{2}}\right). \tag{4.5}$$

As far as this work is concerned, the generation of this theoretical results were made assuming a flat fading Rayleigh multipath channel using QPSK modulation. The curves

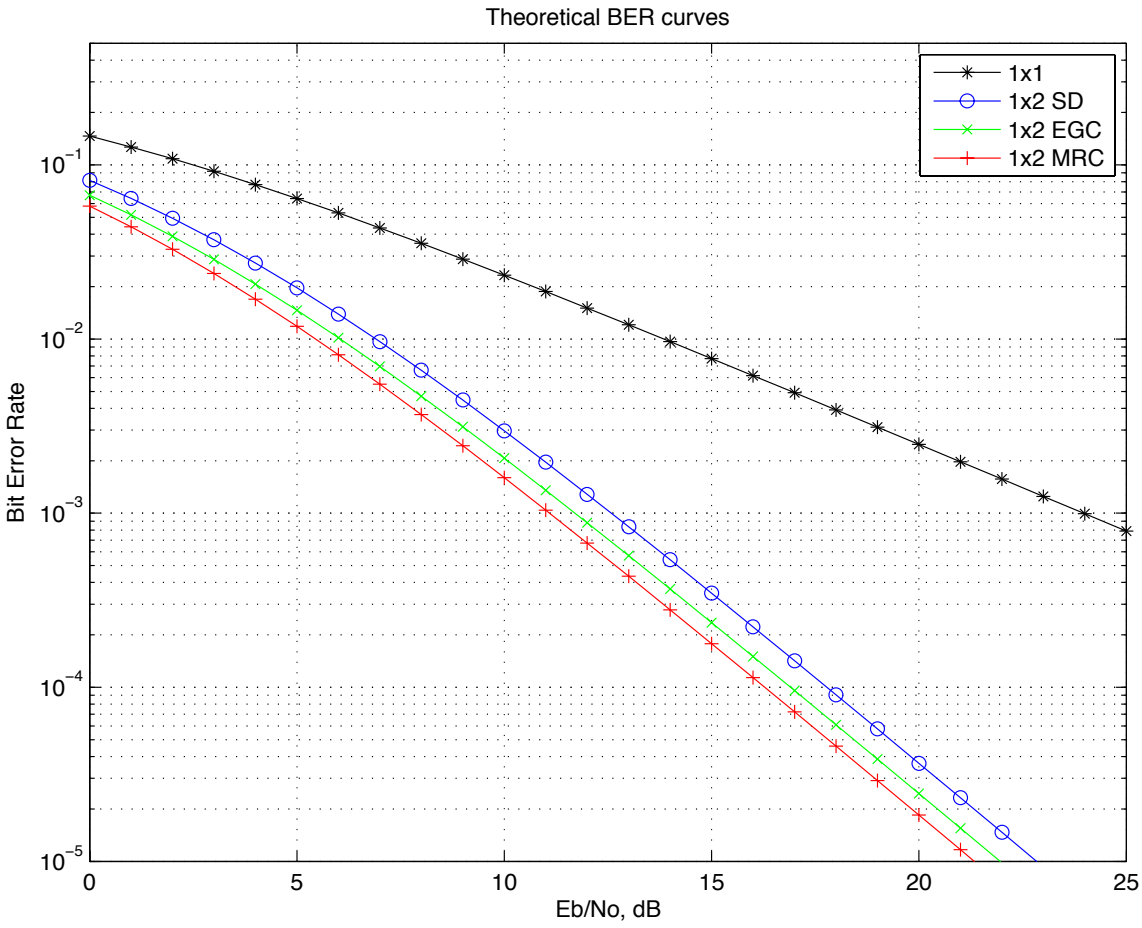
generated and plotted in Figure 4.1 are as expected. The behavior of the curves themselves and in relation to each other are as expected given the projected behavior of these methods. With this set of theoretical data, I was able to move on to generating simulation data for comparison.

4.2.2 Testing Data Acquisition

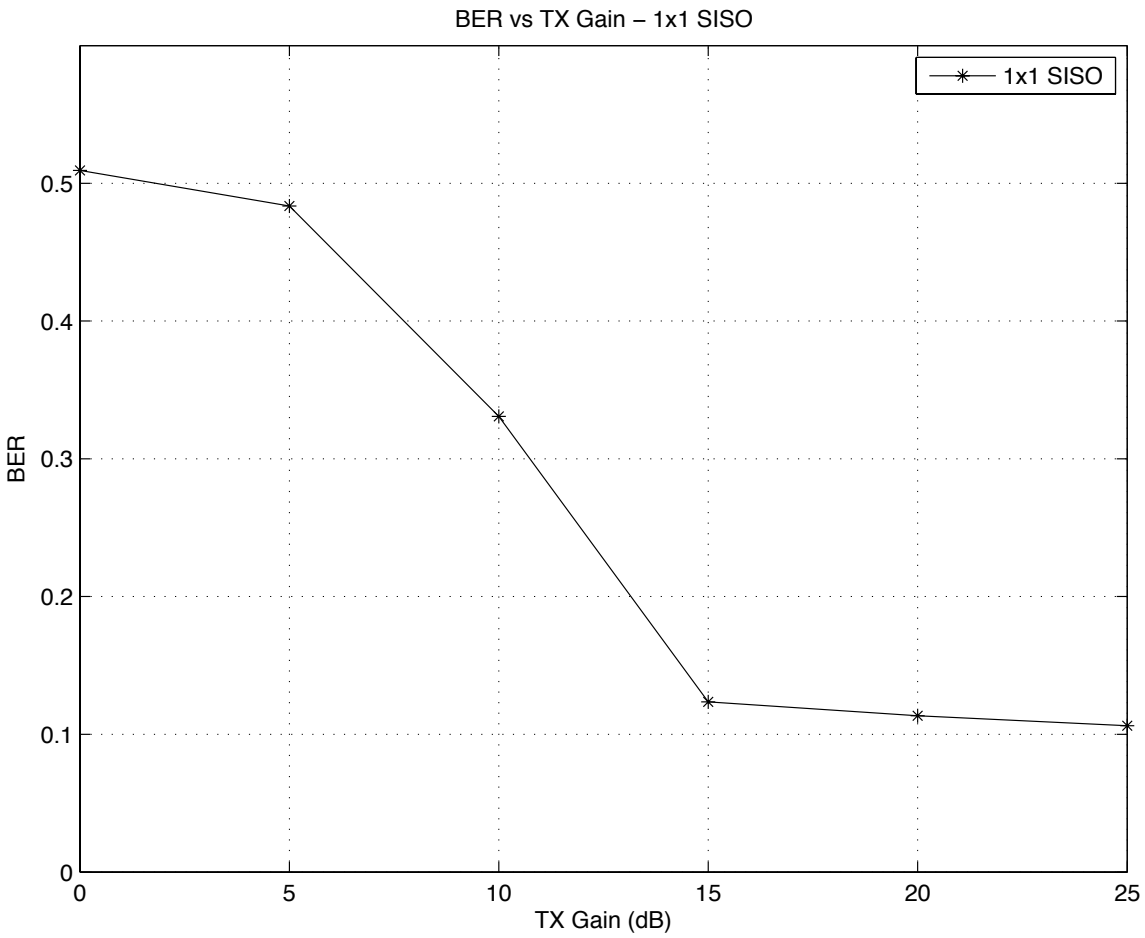
Experiments for this work were conducted at distances of 7.5, 15, and 22.5 feet. This was decided after testing the USRPs at various distances after which it became clear the most dynamic range of BER was obtained at a distance of 15 feet. When tested at distances closer than 15 feet the gain could not be reduced low enough to obtain high error rates. Beyond this distance the received signal was too weak for the receiver from 0-10 dB, and from 10-25 dB it provided rather poor results. After observing this behavior, the data from testing at 15 feet was chosen as the most conclusive data for the testing. Plots of testing at distance of 7.5 and 22.5 feet are shown in Figures 4.4 and 4.5 for reference. These tests were performed using the default RX setting on the USRP N210 which uses the RF2 antenna (TX/RX) port for both TX and RX because of the half duplex nature of this testing. It was noted during testing that use of the dedicated RX2 input of the RF1 antenna port extended the range of the USRPs communication but it was decided to use RF2 because of the dynamic range of its performance at 15 feet.

BER was recorded from tests conducted at gain settings increasing in 2.5 dB increments from 0-25 (25dB is the maximum dB gain of the USRP for the TX side of the WBX board). As for the receiver(s), the RX gain was set to 15 and left constant for all experiments. This was chosen because it is a median value in the range for the RX side of the WBX board (0-38 dB). All experiments were done with a direct line of sight between transmitter and receiver(s) and the channel was assumed to be flat fading. No equalization or RAKE reception was used. Figure 4.1 shows the plot of the theoretical curves for the 1×1 SISO and 1×2 SIMO cases which were used for comparison sake during the USRP testing. It can be noted in all the provided plots that the performance tapers off from 15 dB on. Seeing as this is not expected in a typical BER plot, it was initially a concern. Testing

was repeated multiple times times to monitor for this tapering action. This characteristic proved to be consistent of these plots through the entire course of this work. An inquiry to Ettus Research was made in regards to this behavior and they indicated that the gain compression of the analog amplifier in the USRP would be the most likely cause. This gain compression causes the linearity of the plot to cease and the form of the curve becomes nonlinear, as noticeable in the provided plots. Therefore, it was decided that this behavior must be due to the gain compression of the analog amplifier in the USRP. [h!]

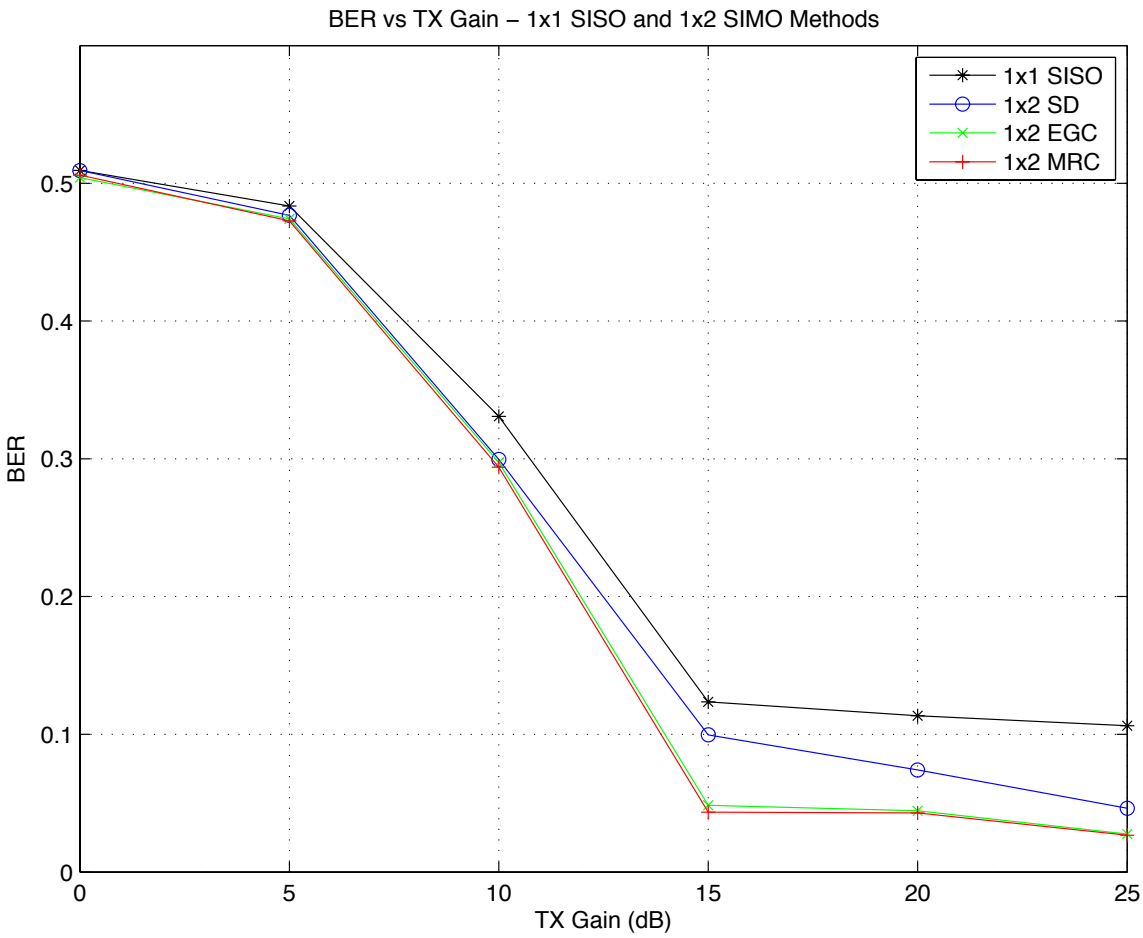


Theoretical BER vs TX Gain for 1×1 SISO and 1×2 SD, EGC, and MRC configurations



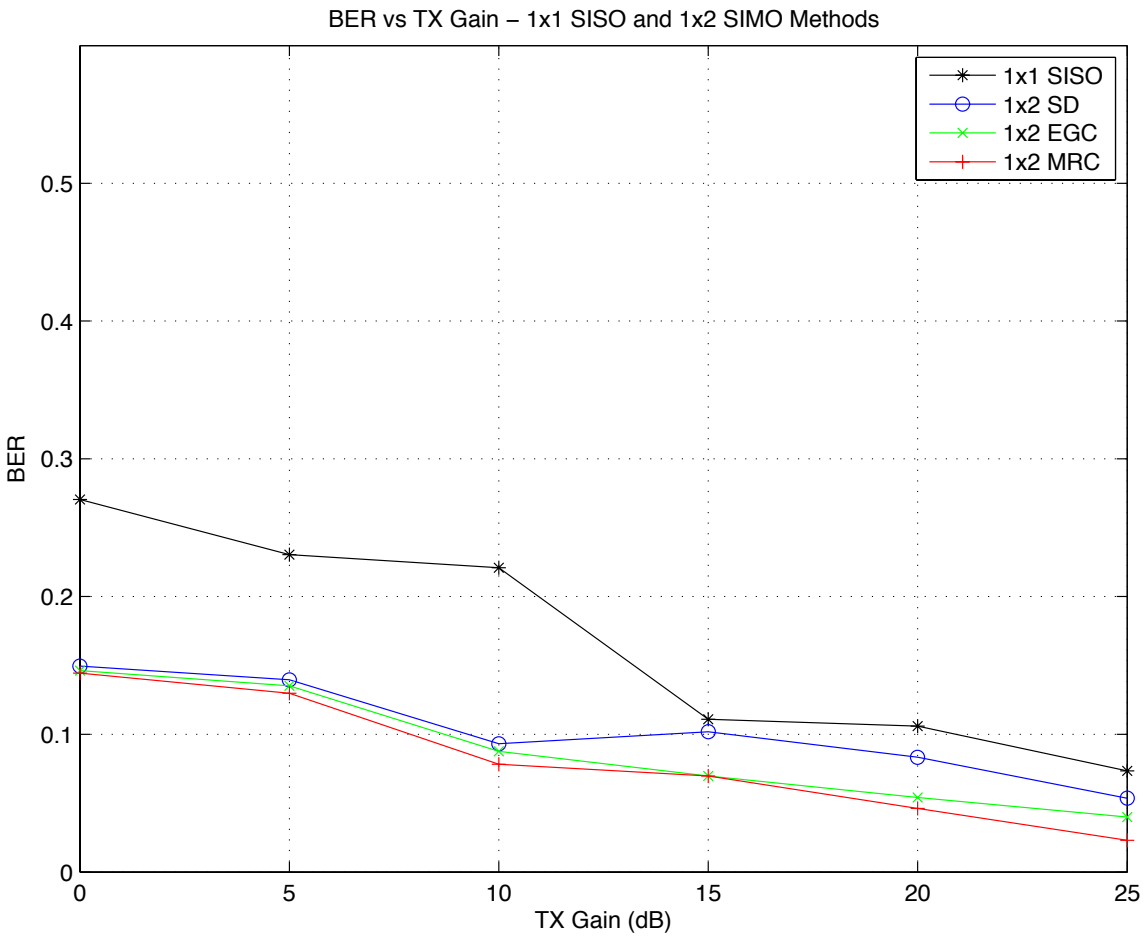
[h!]

BER results for 1×1 SISO configuration



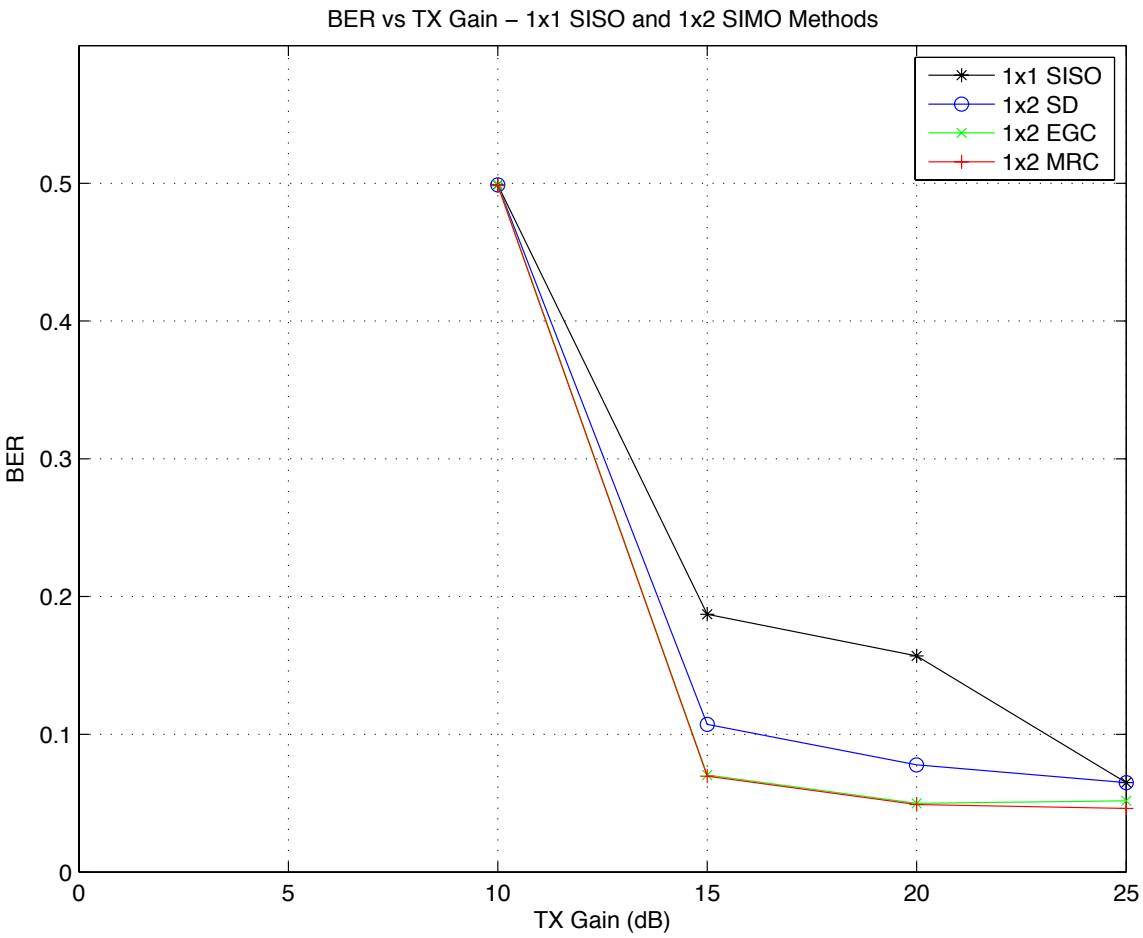
[h!]

BER results for SD,EGC, and MRC 1×2 configurations at distance of 15 feet



[h!]

BER results for SD,EGC, and MRC 1×2 configurations at distance of 7.5 feet



[h!] BER results for SD,EGC, and MRC 1×2 configurations at distance of 22.5 feet

4.2.3 1×2 - Selection Diversity (SD) Case

The first of the SIMO techniques tested in the USRP testbed was selection diversity. The flow of this configuration is shown in Figure 4.1. In order to use this method the magnitude of each received signal vector had to be monitored and measured prior to demodulation to determine which signal had encountered the stronger channel. This step in the process is represented by the channel gain estimation block in the Figure 4.1. Once this was determined the corresponding demodulated signal stream was chosen, on which the BER calculations were performed.

The results of this method's testing is represented by the blue '$\circ$' line in Figure 4.3. As expected, this method shows a slight performance gain over the standard 1×1 setup. The

performance gain is less because only one signal is used in SD. The other signal's energy is ignored in this case, and is not used at all.

4.2.4 1×2 - Equal Gain Combining (EGC) Case

The next technique tested for this work was the EGC method. The flow of this configuration is shown in Figure 4.2. Based on theoretical calculations, it was expected that this method would perform better than SD, which it does. Each received signal stream was accessed just prior to demodulation and the signals were accumulated, and demodulated.

The results of this method's testing is represented by the green 'x' line in Figure 4.3. Due to the fact that the QPSK modulation scheme was used, there was only one point per quadrant in the modulation constellation. Thanks to this, the boosting of each received signals amplitude from the origin meant that there was a more defined difference between respective points in the received vector, making it easier for the decoder to make the correct bit decisions, hence the performance gain.

4.2.5 1×2 - Maximal Ratio Combining (MRC)

The last technique tested in these experiments was MRC. The flowgraph is for this configuration is shown in Figure 4.3. Basically utilizing elements from SD and EGC, MRC takes the measured channel gains (represented by the channel gain estimation block in the flowgraph) from each receive channel and multiplies each channel's received signal by that value in an effort to weigh each signal by its respective level of accuracy. This weighing of the signals makes it so the signal with a better quality channel is weighed greater than that of the weaker channel. It then accumulates the signals like EGC prior to demodulation.

The results of this method's testing is represented by the red '+' line in Figure 4.3. As expected based on the theoretical results this method provided the greatest performance improvement of the SIMO techniques althought it is minimal in comparison to EGC.

4.2.6 Summary of Results

Overall, the results of the SIMO methods' testing followed the expected performance curve as defined by the theoretical results in Figure 4.1. All three methods showed a noticeable performance improvement over that of the 1×1 SISO case. This presented itself as conclusive evidence that the methods used to perform the SIMO method testing provided a performance gain relative to the performance of the SISO configuration.

Chapter 5

Alamouti Work

Once the implementations of the SIMO methods were completed, the focus shifted to implementing MISO methods; specifically, the Alamouti space time block code method. As previously mentioned, an Alamouti encoder/combiner blockset exists within Simulink and was used for this portion of the work. The original 1×1 SISO transmitter model was taken and adapted for this part. The Alamouti encoder block was added just after the QPSK modulator and before the USRP transmitter block. Also, a selector block was added to break the two columns of the transmit signal and route them to their respective USRP block. Similarly, on the receiver end, the Alamouti combiner block had to be added before the QPSK demodulator block.

Unfortunately, a working version of the Alamouti coded system was not successfully created and tested for a myriad of reasons. While the Alamouti blockset works well within the simulated system when the USRPs and QPSK sync portions are not included, the addition of the actual wireless channel along with the USRPs and QPSK sync portions introduce some major variables that could not be compensated for successfully. This is not to say that work on this topic is not worth continuing, it was just not completed in time for this work. Time was spent attacking this problem from various angles but regardless of strategy, the following issues were too much to overcome.

Aside from some minor logistical issues revolving around where the Alamouti combiner block was to be actually placed within the receiver model, the major issue encountered that ultimately halted progress on this part was the channel gain input required by the

Alamouti combiner. The channel gain input of the Alamouti combiner is used to generate the orthogonal mixing matrix needed to correctly decode the received data. If the channel gain input is not correct or at least close to it, the mixing matrix generated and used for decoding will not be orthogonal and will therefore incorrectly decode the data. The problem with this within the context of this work was that there exists no reliable way of measuring or approximating the channel gain of the system and therefore there is no way to feed a correct channel gain estimate to the combiner for decoding.

The initial method tested during this stage was a sort of batch processing. A separate model was generated which merely measured the channel gain of the system based on comparing the received signal to the transmitted signal. Then once a set of channel gains was recorded it was read back in to the Alamout receiver and fed into the channel gain input of the Alamouti combiner block. This method rotated back and forth between recording the channel gain and then receiving the signal transmitted by the 2×1 transmitter model. While this method sounds like it would at least work partially in some capacity on paper, it did not work well enough to show acceptable BER performance during testing and analysis.

Overall, the tested methods for implementing Alamouti STBC into the MATLAB/Simulink USRP testbed were not successful. Due to the high variance in the wireless channel and the inaccuracy of the channel gain estimation, an acceptable level of performance was not able to be achieved. It is believed that batch processing method was not acceptable for channel gain estimation and therefore caused the mixing matrix of the Alamouti combiner to not be orthogonal which in turn negates the whole point of the Alamouti STBC. This method is still a possibility for implementation in the USRP testbed however some substantial work on channel gain estimation needs to be done in order to develop a usable method in order to have accurate channel gain estimates for the Alamouti combiner block within the receiver.

Chapter 6

Conclusions

In an ever growing wireless world, the days of simple 1×1 wireless communication systems are coming to an end. Multiple antenna transmission techniques are not a new idea by any means, but in the context of this research area, these implementations are believed to be. While the performance of the discussed SIMO and MISO methods has been proven in practice, theoretical results don't do much for improving real world applications. That brings us to the motivation for this work. Point to point wireless communication systems are vital in numerous applications in nearly every market. Specifically for this work, it is the desire of the grant sponsors to eventually use these methods in their military radio projects. Proving the methods described in this document serves as a building block for real world implementation in their systems.

As previously mentioned, the initial support from Mathworks made no effort to deal with time synchronization of the transmitted signals, which posed a major problem early on. Thanks to the QPSK synchronization package released by Mathworks, minimal time was exerted trying to fix the synchronization issues which meant I was able to move on to the SIMO/MISO implementations sooner. Having that said, the synchronization receiver could pose some problems for future work. It is set up to be fairly self contained, which is good if the user just wants it to work. The problem surfaces when modifications to the receiver are necessary. Due to its hierarchical nature of embedded subsystems, the user must be very cautious when making modifications because the model is very sensitive. Additions or subtractions to the receiver tended to cause sample time issues or recording issues during

this work. Other than that, the supplied synchronization receiver worked very well and was a tremendous asset to this work.

The SIMO method implementation was an interesting process. As stated, it was decided that EGC would be the first method implemented because it was successfully implemented in GNURadio and had been proven to show performance improvements. The EGC case serves as a prime example of a system that required modifications to the QPSK receiver. Thankfully the modifications for this were simple and just required a duplication of the original receiver with an addition of signals just prior to demodulation. In hindsight, the EGC implementation was the smoothest of the three SIMO methods. When it came to SD and MRC, both were taken on simultaneously. Due to the fact that for both these cases the channel gain had to be monitored, the signal processing could not be done in real time. The received signals had to be compared to the transmitted signal in order to determine the channel gain of each signal. For the SD case the signal with the stronger channel was chosen for demodulation while the MRC case weighted each signal based on its gain prior to demodulation. The implementations of these two methods was a little more involved than that of the EGC method however they too ended up providing the expected results. Overall, the SIMO methods implemented provided good data and helped us to draw some important conclusions. As stated, the output was as expected, however some of the things encountered along the way were in some ways more important in reference to the future of the project than the data itself. Extensive work with the QPSK receiver during the SIMO implementations provided actual application driven reasons to modify the synchronization receiver. This, in turn, provided us with useful insight to what future work could entail when it comes to working with the Mathworks synchronization support.

The time spent on the MISO Alamouti STBC implementation was not enough to achieve the goal, unfortunately, however this does not mean useful results were not obtained. While the SIMO methods revealed how the user can expect to have to adapt and deal with errors caused by modifications to the QPSK synchronization receiver, the Alamouti case shed light on other issues that still exist with the USRP support. The success of the Alamouti implementation, and any other multiple antenna technique reliant on channel estimation, revolves around developing a working method of channel estimation. Having used the Alamouti en-

coder/combiner blocks within the single computer simulators I used throughout my work to test the methods implemented, I feel strongly that once a working channel estimator is developed, the Alamouti implementation should not be far behind.

As I take a step back and observe all the work that has taken place over the last two years on this project, both the lists of achievements and roadblocks surprise me. As I stated before, the goals set forth for the SIMO implementations were reached as expected. This was a very proud achievement, but the additional skills developed through working with the software, and the roadblocks encountered and surpassed could potentially surpass the achievements of the data. Those gains, along with the information learned through working on the Alamouti implementation serve as vital stepping stones to further progress in this area. When you add those factors to the first successful SIMO implementations with the USRP in MATLAB/Simulink, the end result is a successful study in multiple antenna techniques and their implementation in a USRP based SDR testbed.

References

[1] P. Johnson, "New Research Lab Leads to Unique Radio Receiver," *E-Systems Team*, vol. 5, no. 4, pp. 6–7, 1985.

[2] P. Hoeher and H. Lang, "Coded=8PSK modem for fixed and mobile satellite services based on DSP," *Proc. First Int. Workshop on Digital Signal Processing Techniques Applied to Space Communications*, pp. 117–123, 1988.

[3] "USRP N210 Product Listing," https://www.ettus.com/product/details/UN210-KIT, June 2012.

[4] Rice University, "WARPLab Framework Overview - Rice University Wireless Open-Access Research Platform (WARP)," warp.rice.edu/trac/wiki/WARPLab, June 2012.

[5] K. Amiri, Yang Sun, P. Murphy, Cavallaro Hunter C., and A J.R. Sabharwal, "WARP, a Unified Wireless Network Testbed for Education and Research," *IEEE International Conference on Microelectronic Systems Education*, pp. 53–54, 2007.

[6] Xiaolong Li, Weihong Hu, H. Yousefi'zadeh, and A. Qureshi, "A case study of a MIMO SDR implementation," *IEEE MILCOM*, pp. 1–7, 2008.

[7] Jin Zhang, Juncheng Jia, Qian Zhang, and E.M.K. Lo, "Implementation and Evaluation of Cooperative Communication Schemes in Software-Defined Radio Testbed," *IEEE INFOCOM*.

[8] "USRP Hardware Support from Mathworks," http://www.mathworks.com/discovery/sdr/usrp.html, June 2012.

[9] "USRP N210 Datasheet," https://www.ettus.com/content/files/2987_Ettus_N200-210_DS_FINAL_1.27.12_1.pdf, June 2012.

[10] "WARP Order Price List," http://mangocomm.com/price-list, June 2012.

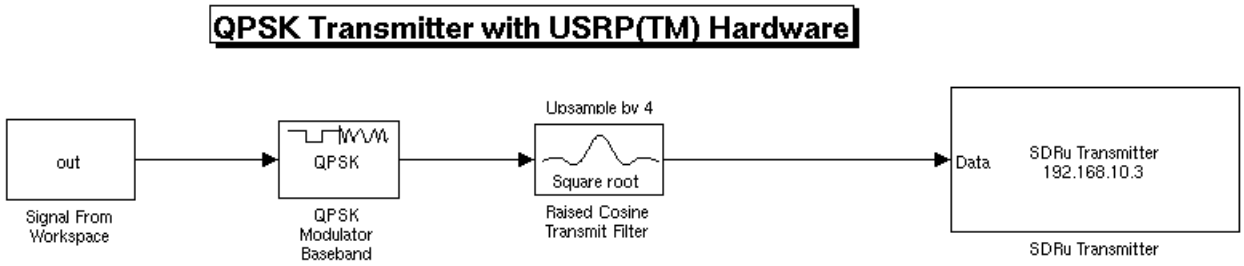
[11] "Selection Diversity," http://www.dsplog.com/2008/09/06/receiver-diversity-selection-diversity/, Sept. 2008.

[12] "Equal Gain Combining," http://www.dsplog.com/2008/09/19/equal-gain-combining/, Sept. 2008.

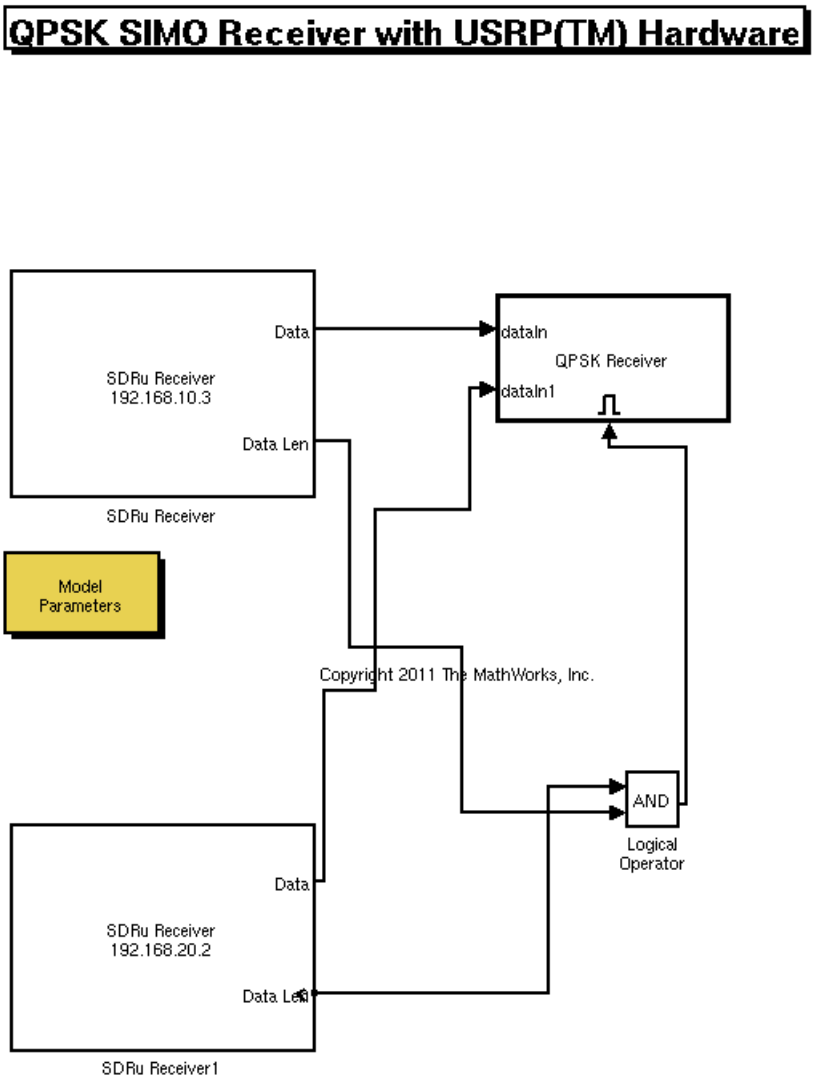
[13] "Maximal Ratio Combining," http://www.dsplog.com/2008/09/28/maximal-ratio-combining/, Sept. 2008.

A Model Screenshots

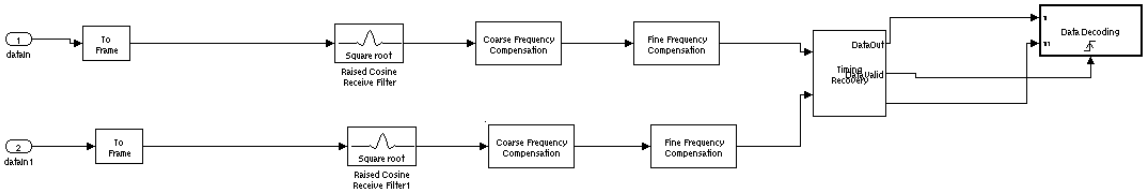
A.1 USRP Transmitter Model



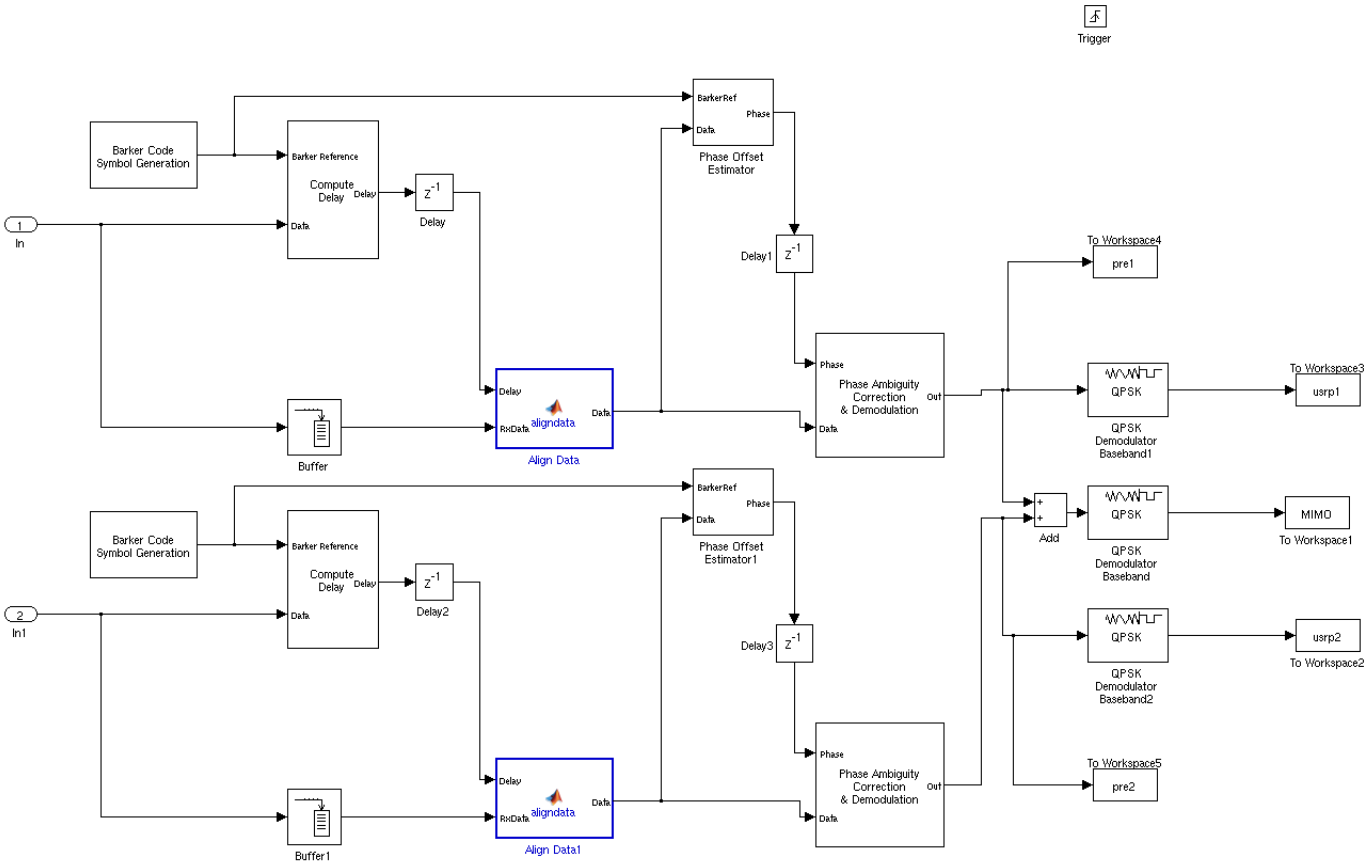
A.2 USRP SIMO Receiver Model



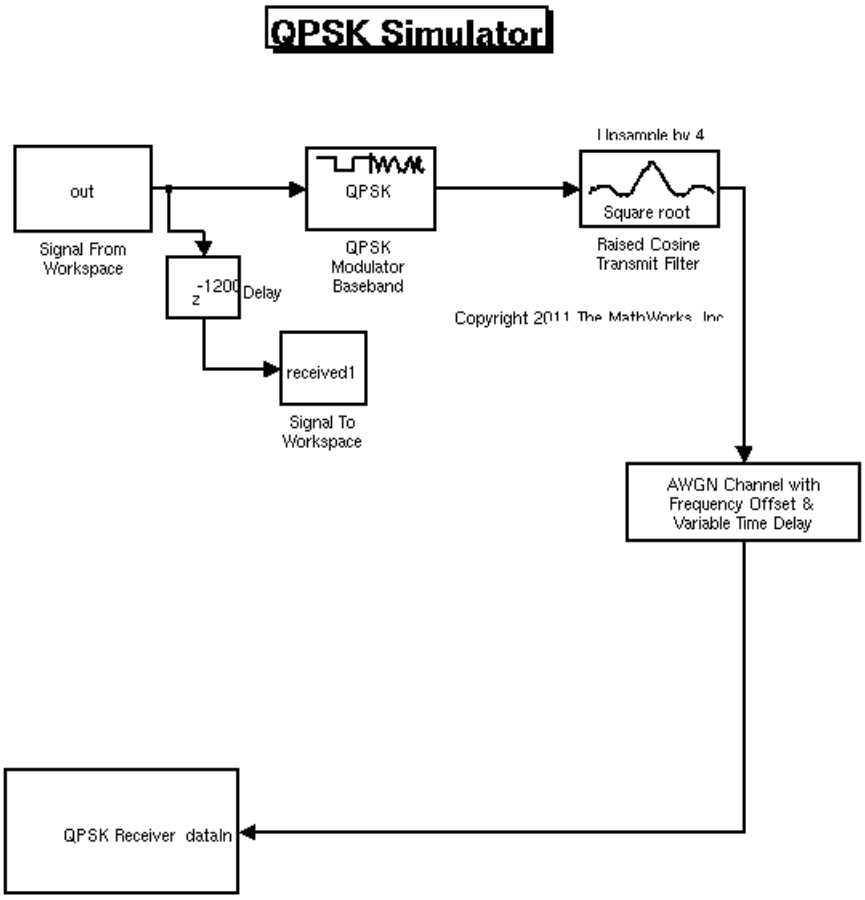
A.3 USRP SIMO Sync Receiver Model



A.4 USRP SIMO Receiver Data Decoder



A.5 QPSK Simulator Model



B MATLAB Scripts

B.1 EGC BER Calculator

```
u1 = num2str(usrp1)';
u2 = num2str(usrp2)';
m1 = num2str(MIMO)';

barker = '11111111110000111100110011';

ind1 = min(findstr(barker,u1));
ind2 = min(findstr(barker,u2));
ind3 = min(findstr(barker,m1));
indices = [ind1 ind2 ind3 ];
location = max(indices)
s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);



sizes = [s1(1) s2(1) m1(1) ]
usrp1 = usrp1(location:s1(1));
usrp2 = usrp2(location:s2(1));
MIMO = MIMO(location:m1(1));

s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);

sizes = [s1(1) s2(1) m1(1)]

newsizes = floor(sizes/200)*200;
upper = min(newsizes)

usrp1 = usrp1(1:upper);
usrp2 = usrp2(1:upper);
```

```
MIMO = MIMO(1:upper);



upper = min(sizes)/200;

x = [];
for i = 1:upper
    x = vertcat(x,out);
    i = i+1;
end

usrp1error0(j) = (sum(usrp1~=x)) / (upper*200)
usrp2error0(j) = (sum(usrp2~=x)) / (upper*200)
MIMOerror0(j) = (sum(MIMO~=x)) / (upper*200)
j=j+1;
```

B.2 SD BER Calculator

```


u1 = num2str(usrp1)';
u2 = num2str(usrp2)';
m1 = num2str(MIMO)';

barker = '11111111110000111100110011';

ind1 = min(findstr(barker,u1));
ind2 = min(findstr(barker,u2));
ind3 = min(findstr(barker,m1));
indices = [ind1 ind2 ind3 ];
location = max(indices);
s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);
```

```
sizes = [s1(1) s2(1) m1(1) ];
usrp1 = usrp1(location:s1(1));
usrp2 = usrp2(location:s2(1));
MIMO = MIMO(location:m1(1));

s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);

sizes = [s1(1) s2(1) m1(1)];

newsizes = floor(sizes/200)*200;
upper = min(newsizes);

usrp1 = usrp1(1:upper);
usrp2 = usrp2(1:upper);
MIMO = MIMO(1:upper);



upper = min(sizes)/200;

x = [];
for i = 1:upper
    x = vertcat(x,out);
    i = i+1;
end
Chan1 = mean(abs(Raw2/(1+j)))
Chan2 = mean(abs(Raw / (1+j)))
if Chan1 > Chan2
    1
    SDerror25(j) = (sum(usrp1~=x)) / (upper*200);
else
    2
    SDerror25(j) = (sum(usrp2~=x)) / (upper*200);
end
SDerror25
SDusrp1error25(j) = (sum(usrp1~=x)) / (upper*200)
SDusrp2error25(j) = (sum(usrp2~=x)) / (upper*200)%MIMOerror22_5(j) = (sum(MIMO~=x)) / (upp
```

```
j=j+1
```

B.3 MRC BER Calculator

```
Chan1 = mean(abs(Raw2/(1+j)));
Chan2 = mean(abs(Raw / (1+j)));

stream1 = Chan1 * Raw3;
stream2 = Chan2 * Raw4;

MIMO = stream1 + stream2;



hdemod = comm.QPSKDemodulator('PhaseOffset',pi/4,'BitOutput',true);

MIMO = step(hdemod,MIMO);
u1 = num2str(usrp1)';
u2 = num2str(usrp2)';
m1 = num2str(MIMO)';

barker = '11111111110000111100110011';

ind1 = min(findstr(barker,u1));
ind2 = min(findstr(barker,u2));
ind3 = min(findstr(barker,m1));
indices = [ind1 ind2 ind3 ];
location = max(indices);
s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);



sizes = [s1(1) s2(1) m1(1) ];
usrp1 = usrp1(location:s1(1));
usrp2 = usrp2(location:s2(1));
```

```
MIMO = MIMO(location:m1(1));

s1 = size(usrp1);
s2 = size(usrp2);
m1 = size(MIMO);

sizes = [s1(1) s2(1) m1(1)];

newsizes = floor(sizes/200)*200;
upper = min(newsizes);

usrp1 = usrp1(1:upper);
usrp2 = usrp2(1:upper);
MIMO = MIMO(1:upper);



upper = min(sizes)/200;

x = [];
for i = 1:upper
    x = vertcat(x,out);
    i = i+1;
end









MRCusrp1error25(j) = (sum(usrp1~=x)) / (upper*200)


MRCusrp2error25(j) = (sum(usrp2~=x)) / (upper*200)
MRCMIMOerror25(j) = (sum(MIMO~=x))/(upper*200)
j = j+1
```