PHYSICAL LAYER WATERMARKING OF BINARY PHASE-SHIFT KEYED SIGNALS USING STANDARD GNU RADIO BLOCKS

By

BRUCE LEBOLD

Bachelor of Science in Electrical Engineering

Oklahoma State University

Stillwater, OK

2009

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2011

PHYSICAL LAYER WATERMARKING OF BINARY

PHASE-SHIFT KEYED SIGNALS

USING STANDARD GNU RADIO BLOCKS

Thesis Approved:

Dr. George Scheets

Thesis Advisor

Dr. Keith Teague

Dr. Damon Chandler

Dr. Mark E. Payton

Dean of the Graduate College

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

The development and implementation of a communications system physical layer watermarking method is described within this paper. The results of the watermarking method are analyzed for viability. In Chapter I the need for physical layer watermarking is discussed and the approach taken is introduced. Chapter II describes some of the previous work done in the area of physical layer watermarking. Chapter III outlines the method by which the new watermarking method is developed and tested. In Chapter IV the test results are detailed and analyzed. The thesis will be concluded and possible improvements will be discussed in Chapter V.

1.1 Background

Security concerns are inherent to wireless transmissions. The physically open nature of wireless transmission and reception makes it more vulnerable to several threats. Previously, encryption has been relied upon to secure wirelessly transmitted data. Encryption secures data by restricting who can access the data within the transmission. However, if there is to be no restriction on who may access the data, encryption is not an option. Recent technological advances have sparked research into physical layer watermarking.

Watermarking has previously been used widely in multimedia as a method of copyright protection where data can be authenticated for a user and tracked if a user distributes the data. Typically, watermarking is performed at the application layer where data is manipulated in an imperceptible way to add hidden information before modulating and transmitting the signal. Very careful consideration must be taken as to the type of data being transmitted to determine the best method of watermarking. Otherwise the watermark will degrade the information making it unusable.

Physical layer watermarking methods watermark, or add a hidden information channel, to the physical layer of a signal. Instead of altering the data being transmitted, the transmission itself is altered. Watermarking in the physical layer does not directly alter the data being transmitted. Any degradation caused to the host digital signal is seen as an increased bit error rate. This means no knowledge of the type of data contained by the host signal is necessary to implant or decode a physical layer watermark.

The recent advent of Digital Television (DTV) combined with the ongoing development of Software Defined Radio (SDR) has led to the current evolution of spectrum allocation rules by the FCC. Each broadcast television channel is allocated 6 MHz of bandwidth. Even in some urban areas only around ten channels are occupied, reflecting a 20% usage of the 300 MHz of spectrum reserved for broadcast DTV. This encouraged the FCC's recent decision to allow unlicensed use of unused broadcast TV spectrum in hopes that new technologies, such as "super Wi-Fi hot spots" will be spurred. [1] [2] With the development of new technologies, the decision may also alleviate the shortage of spectrum available for current mobile broadband users. A new wireless standard, IEEE 802.22, is being developed to take advantage of these new rules and provide a standard for future SDR networking devices. [3] SDR networks are an area where physical layer authentication may prove useful.

1.2 Importance

There are certain situations in which physical layer authentication of a transmission could add an important layer of security to a transmission. With current technologies, the main benefit of watermarking would be the authentication of a signal. The new technologies being developed with SDR could have many uses for watermarking including frequency prioritization, authentication, and data multiplexing.

Emergency broadcasts over TV or radio need to be publically available for anyone to receive. This means it would be possible for someone to broadcast a fake announcement. If this were to occur, it may take some time to determine it was a faked broadcast. Any delay in responding to a fake emergency broadcast could have great negative effect – allowing for the onset of panic. Watermarking emergency broadcasts would allow concerned parties to immediately identify a fake broadcast and update the public with corrected information.

In critical networks, such as military or other networks containing sensitive information, it may be desirable to add a layer of authentication that is not available through the use of encryption. In most wireless networks, all users ultimately have access to all transmitted data. This gives an attacker with network access the ability to impersonate another user. Impersonation attacks, such as MAC address spoofing are a common method of gaining access to restricted network assets and poses a security risk. [4] Watermarking could provide specific network devices a higher level of trust or priority within a larger network of users that would normally be treated with the same level of scrutiny. For example, if a specific server should only provide access to one client on a larger network, the client could be outfitted with a network device that could implant a watermark on the signal. Then only connections from that physical client could be authenticated.

The development of SDR has brought two new authentication scenarios that would benefit from physical layer watermarking. In both, there are incumbent users that are given frequency priority. The first would be a “Trust a Few” scenario – where some users may try to impersonate an incumbent. The other scenario would be “Trust Everyone.” In this scenario, all users would be trusted to obey the incumbency rules.

IEEE 802.22 devices represent a “Trust a Few” scenario. As a condition for sharing the broadcast TV spectrum, devices that operate under the new FCC rules must, using GPS data, check a geographic database for incumbent users. Devices may also be developed that do not check the geographic database but instead must sense any incumbent users of any channel before attempting to transmit on that channel. SDR devices operating in this manner would be put through stringent testing to verify they correctly sense incumbent users with low probability of false negative according to the FCC rules [2]. TV broadcasters and wireless microphones are each given incumbent status under the rules.

If a broadcast is detected at or above the incumbent threshold, the spectrum sensing SDR device may not transmit in a frequency range that could interfere with the incumbent broadcast. For example an incumbent ATSC broadcast on channel N would prevent the spectrum sensing SDR device from transmitting on channels N, N-1, or N+1. The SDR device would need to cease transmission and scan the spectrum every several seconds to ensure only minimal interference to a new or returning incumbent broadcaster. When an incumbent signal is not present at or above the threshold, the SDR device may transmit at the allowable EIRP of up to 4 watts depending on the type of device (portable or stationary, antenna type, etc…) [1]. The following sensing threshold requirements are from the FCC document 10-174 [2]:

> *(c) Sensing requirements.*
> *(1) Detection threshold.*
> *(i) The required detection thresholds are:*
> *(A) ATSC digital TV signals: -114 dBm, averaged over a 6 MHz bandwidth;*
> *(B) NTSC analog TV signals: -114 dBm, averaged over a 100 kHz bandwidth;*
> *(C) Low power auxiliary, including wireless microphone, signals: -107 dBm, averaged over a 200 kHz bandwidth.*

Due to the incumbency rules, a designer or user of a wireless device may have an interest in illegitimately mimicking the transmission of a TV broadcast channel. This would allow that device to be protected from unwanted interference from spectrum sensing devices that it may otherwise not be protected from under Part 15 of the unlicensed radio rules. Transmitter mimicking could also be used to hog multiple channels on the spectrum. A clever designer could effectively disable transmissions from every spectrum sensing SDR device in range by creating false TV stations. SDR devices that relied on spectrum sensing to follow incumbency rules would have to cease transmission on any detected TV channels within several seconds of detecting the false TV stations.

If TV broadcast mimicking becomes a problem, the solution may be to require legitimate broadcasters to watermark their signal. A public key encryption system could be used to generate the watermark. This system would give everyone a public key with the ability to decode and verify the watermark, but only a broadcaster would know the private key used to encode a valid watermark.

As previously mentioned, there may also be “Trust Everyone” situations where users simply need to be identified for priorities. One example of a “Trust Everyone” scenario would be a military radio system. Different radio technologies may be in use and all users are expected to operate appropriately, adhering to frequency incumbency rules. With SDR, frequency sharing is possible amongst many types of communications. In addition, the users of a frequency can be authenticated when needed.

As required with SDR devices sharing TV broadcast spectrum, all military radios would need to periodically scan the channel in use for incumbent users. A high priority signal transmission should almost immediately cause the ceasing of all other transmission on that frequency. So the scan may need to be performed every several milliseconds. If an officer using a two-way radio attempts to issue critical commands, any other service using that frequency must quickly stop and possibly select a different frequency – be it a video transmission or other two-way radio users. If the video communication equipment was using a different data protocol or using a different encryption standard it may not be able to determine a high priority transmission had been received by the data alone. Watermarking the physical transmission would allow some side data, the watermark, to be understood without being aware of the encryption scheme or data protocol in use by the host signal.

Watermark-aware radios would be able to authenticate users while unaware systems are unable to detect the presence of the watermark. The data received by unaware systems is unchanged. Attackers may try to inhibit the use of a frequency with false radio transmissions. Using a physical layer watermarking system, the false transmissions could be identified. Any questionable transmission that did not include a watermark generated with an assigned ID would quickly be identified by watermark aware receivers.

Another use for physical layer watermarking comes from the notion that watermarking is simply a way to hide data within a stream of data. With a properly developed watermarking method, the watermark might be usable as a multiplexer rather than an authentication method. Currently, wireless networking devices use different sub-carrier modulation schemes, ranging from BPSK to 64QAM depending on signal strength. Subject to channel capacity constraints, watermarking could allow each sub-carrier data stream to carry multiple data streams. This might allow more configuration to more efficiently use the bandwidth of each channel. In a situation where two users had excess signal quality, two data streams could be combined. For example, the data for a user at moderate range and high bitrate, using QPSK, and the data for a user at short range and low bitrate, using BPSK, could be combined into one QPSK signal with a watermark - the information for the moderate range user stored in the host QPSK signal and the information for the short range user stored in a watermark. The main user, that of the host signal, would not need any modification to their receiver making the method backwards compatible with current hardware. The modified modulation scheme could be used with both aware and unaware receivers.

1.3 Contribution

This document discusses the development of a watermarking method that could potentially be applied to current digital communications equipment as well as many developing SDR technologies. The physical layer watermarking method will not rely on each device understanding the data protocol of other devices, but instead it will only require understanding the physical attributes of the RF transmission such as modulation type and bitrate. The physical layer watermarking method will be configurable to balance host signal degradation with watermark strength and be implementable using either current hardware or SDR devices.

This paper will provide a description of the signal manipulation that occurs to implant a watermark, provide testing results, and analyze the watermarking method for viability by comparing it to previous methods. It will not detail the entire implementation of the watermarking method into a usable system, but it will suggest a general implementation.

The remainder of the paper proceeds as follows: in Chapter II the previous works in physical layer watermarking are introduced; Chapter III discusses the development, implementation, and testing procedures; Chapter IV displays and provides analysis on the results of the watermarking method; and Chapter V concludes the paper.

CHAPTER II

REVIEW OF LITERATURE

In the past, watermarking has seen widespread use in media. Watermarking has been used to imbed copyright information, track media distribution, authenticate data, and discretely transmit data. Due to the attacks that watermarks face, the majority of research on watermarking involves finding ways to imbed the watermark into data in a manner such that altering it will also make the watermarked data unusable. [5] Until recently, physical layer watermarking has seen limited research. It would have been costly to test and implement, but the recent development of SDR and push for spectrum sharing has made physical layer watermarking economically feasible and encouraged research in the topic.

2.1 Previous Work

In [6] the idea of super-imposing a pseudo-random ID onto the ATSC data stream in Digital TV (DTV) transmissions is discussed. A method for identifying transmitters is established; however it does not anticipate that unauthorized users may attempt to mimic a DTV broadcaster. The method is such that “transmitter identification is realized by conducting a cross-correlation function between the received signal and the possible pseudo random sequences.” Anyone who can determine what random number generator is being used may be able to recreate the broadcasters ID. In addition, the paper does not address the possibility of an attacker replaying the broadcast to mimic a DTV broadcaster.

The method in [7] addresses the potential for watermarking signals in 802.11 wireless networking. Specifically OFDM transmissions are simulated using two watermarking approaches. One approach, called constellation dithering, layers a second, low power QPSK signal on each OFDM payload sub-carrier. The other approach, called baud dithering, creates a timing jitter within each data symbol. The two methods in [7] are primarily used to identify a given wireless node. That is, the OFDM signal as a whole is analyzed to verify the presence of the watermark. The paper showed the use of watermark bitrates of around one hundredth of the OFDM symbol rate. These methods could be used to authenticate individual sub-carriers, but the watermark bit rate would be reduced even further.

The watermarking method in [8] outlines a scenario that presents the importance of physical layer watermarking. Four users are considered: Alice (transmitter), Carol (unaware receiver), Bob (aware receiver), and Eve (aware receiver; active adversary). Alice transmits a watermarked signal using a secret key. Carol and Bob both receive the signal, but only Bob is aware of the watermarking scheme and has the secret key. Eve receives the signal and is aware of the watermarking scheme, but does not know the secret key. Eve may also transmit a signal attempt to fool Bob into believing she is Alice. The presence of an adversary creates a need for the secret key as well as a time-varying watermark tag, so that a given tag is only accepted for a given set of data one time. The paper also analyzes the stealth, robustness, and security of its method.

The work in [8] is extended to multicarrier systems in [9] and [10]. The merits of spreading a watermark over several carriers in an OFDM transmission are discussed in [9], and it is shown that spreading a watermark over many low power carriers, rather than a few high power carriers, is most beneficial to the stealth and robustness of the watermark. [10] introduces several methods of watermarking an OFDM transmission and discusses the trade-offs between message throughput and watermark stealth and robustness that occurs with each method. The work in [10] shows that a watermarking technique where watermark to message power is allocated equally across non-zero carriers performs the best in terms of stealth, robustness, security, and required computation power. The information provided in [9] and [10] would be important to future work with the proposed watermarking method. While this thesis discusses only the watermarking of a single carrier, it may be desirable to extend the watermark to have a specific OFDM implementation where the watermark power is spread across several carriers.

2.2 Works Employed

The new physical layer watermarking method proposed by this thesis makes use of the adversary model and configurability from [8] as a main consideration for development. Stealth, robustness, and security are also considered. The method proposed by this paper can be considered an alternative technique to implement the constellation dithering of [7], but unlike [7] it can be implemented and tested using standard software defined radio building blocks or readily available hardware. The previous physical layer watermarking methods researched in [6] [7] [11] seem to have found acceptable methods of implanting a watermark that could be adjusted to reduce noise-like interference to an acceptable level. However, the methods lack configurability, require hardware changes, or, should a software define radio be used for implementation, the use of non-standard building blocks for the software radio. Most previous methods are only simulated since implementation would be difficult. In [7] the method was applied to a prototype system, but the results were only analyzed by reporting a transmitted image had not been visually degraded.

In addition to the previous work on physical layer watermarking, [3] provides an introduction to the IEEE 802.22 standard based on SDR. The document provides knowledge of how spectrum sensing devices may need to react to their environment. It also provides insight into potential problems with spectrum sharing.

The research of this thesis will have several goals. The proposed method of physical layer watermarking will use the previous work as a model to create a real-world testable physical layer watermarking approach. In addition, the developed method will be configurable and provide a user with a method usable for secure authentication.

The next chapter discusses the methodology of the development of the physical layer watermarking method and its implementation and testing.

CHAPTER III

METHODOLOGY

The development and testing of the new watermarking method is discussed in the following four sections. The objective is discussed first, as it had a major impact on the development of the watermarking method. The development and implementation are then discussed before the procedure for testing the watermarking method is outlined.

3.1 Objective

Viability is the primary objective of the new watermarking method. There is great importance placed on the ability for the developed method to be simple and cost effective to implement on current digital systems as well as SDR systems. This led to the use of Ettus Research USRP [12] software radios for development and testing of the watermarking method. The USRP's are intended to act as a customizable radio transmitter or receiver. Rather than designing a radio to perform a specific function, computer software controls the signal to be modulated allowing one device to transmit or receive a multitude of different modulation types at any center frequency, given the proper daughter board for that frequency. The USRP device can transmit a signal with up to 8MHz of bandwidth. A software package, GNU Radio [13], is used to control the USRP radios. The software contains blocks that recreate common RF hardware such as modulators, synchronizers, demodulators, amplifiers, and filters. Using the blocks, a bit stream can be modulated and modified before being transmitted at the selected center frequency. A key goal of this thesis was to use as many readily available GNU Radio blocks as possible in the design of the proposed physical layer watermarking method.

The use of USRP radios allowed for quick implementation of actual radio transmission and reception. Test data could be transmitted, received, and analyzed in a real world, multipath environment.

Proper use of the adversary model from [8] maintains a viable security model by ensuring the watermark is resistant to attacks. Stealth and robustness are also very important – stealth to ensure the unaware receiver does not suffer from interference and robustness to ensure the watermark can be read by aware receivers.

It is important that the physical layer watermarking method is developed to be usable with as many technologies as possible so that spectrum sensing devices can then detect authorized users of many types of transmitters without decoding the data streams.

3.2 Method Development

Originally the goal of this thesis was to create a physical layer watermark that added some small signature onto the transmission. This would have been some slight alteration to the frequency, constellation, or timing of the signal. This would need to be a minute change to maintain stealth, likely only producible and decodable by special purpose SDR systems.

After the investigation into previous physical layer watermarking methods, it was clear there was a need for a method that was not only compatible with SDR but could be implemented with pre-existing hardware and software blocks, such that a special purpose system would not be required.

A typical M-PSK modulator transmits $\log_2(M)$ bits of information per symbol. For example a Binary PSK, or 2-PSK, symbol would transmit one bit of data. Likewise, an 8-PSK symbol would transmit three bits of data. The plot in Figure 1 shows a constellation diagram for a gray-coded 8-PSK symbol.

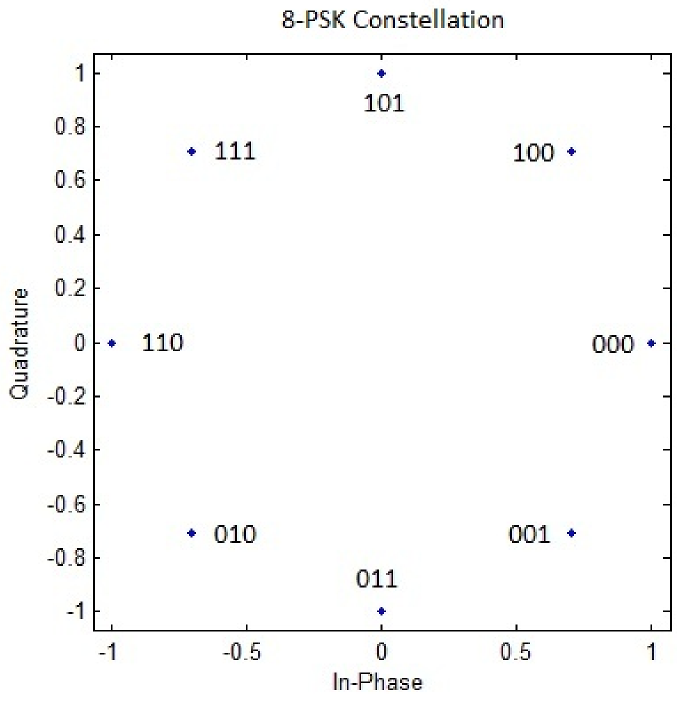


Figure 1 - Gray Coded 8-PSK Constellation

Each symbol transmitted is one of the constellation points and reflects three bits of data. Each symbol is separated by forty-five degrees.

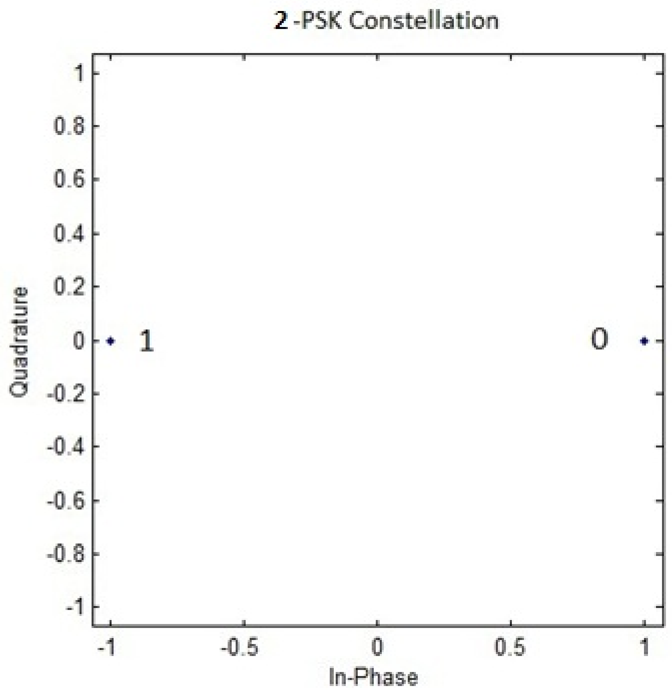


Figure 2 - 2-PSK (BPSK) Constellation

A 2-PSK constellation is shown in Figure 2. The 2-PSK symbol can transmit only one bit per symbol, and each possible symbol is separated by one hundred eighty degrees.

After a symbol is transmitted it becomes distorted by channel fading, additive noise, and interference from other devices. The demodulator matches each received symbol to the closest possible symbol for the M-PSK constellation in use. For example, a single 2-PSK symbol could be shifted by anything less than ninety degrees in either direction and still potentially be correctly demodulated. The shift is considered to be caused by noise and interference. However, unexpected shifts in phase, causing larger phase variance, can hinder the ability of a receiver to lock to the phase of a signal as shown in [14].

The proposed physical layer watermarking method will shift each symbol by a fraction of the angle that would normally cause a symbol change, using standard GNU Radio blocks. Effectively, two bit streams will be transmitted within one symbol stream. The main signal is modulated normally, and the watermark is added by shifting the phase of the modulated signal. To an unaware receiver the shift will appear to be phase noise. The effective power level of the watermark can be adjusted by changing the angle by which the phase is changed. A smaller angle will allow the main signal to retain more of its effective signal to noise ratio (SNR) and decrease the effective SNR of the watermark.

The security will be left to the user to develop as it will differ depending on application. A watermark for each block of data could be created using a secret key, the data, and the time to form a watermark that is not reproducible by someone without the secret key.

The proposed method allows a watermarked signal to be created by shifting the phase of the modulated signal using SDR. SDR applications could shift the phase by any amount. Alternatively, the watermarked signal can be created by modulating a specially coded bit stream that is fed to a higher order PSK modulator. A 2-PSK signal with a forty-five degree shifted watermark could be created using an 8-PSK modulator. Per Figure 1, transmitting 0 with 2-PSK modulator is the same as transmitting 000 with an 8-PSK modulator. Likewise, transmitting a 1 with a 2-PSK modulator is that same as transmitting a 110 with an 8-PSK modulator. When a watermark 1 is present the 2-PSK symbol will be shifted by forty-five degrees. This action can be produced with an 8-PSK modulator. An example of a bit stream encoded with a watermark is shown in Table 1.

Example Watermark Encoding

Main Signal	0	0	1	0	1	1	1	1
Watermark	1	0	0	0	1	0	1	0
8-PSK Encoded	100	000	110	000	010	110	010	110

Table 1 - Example of 8-PSK Watermark Encoding

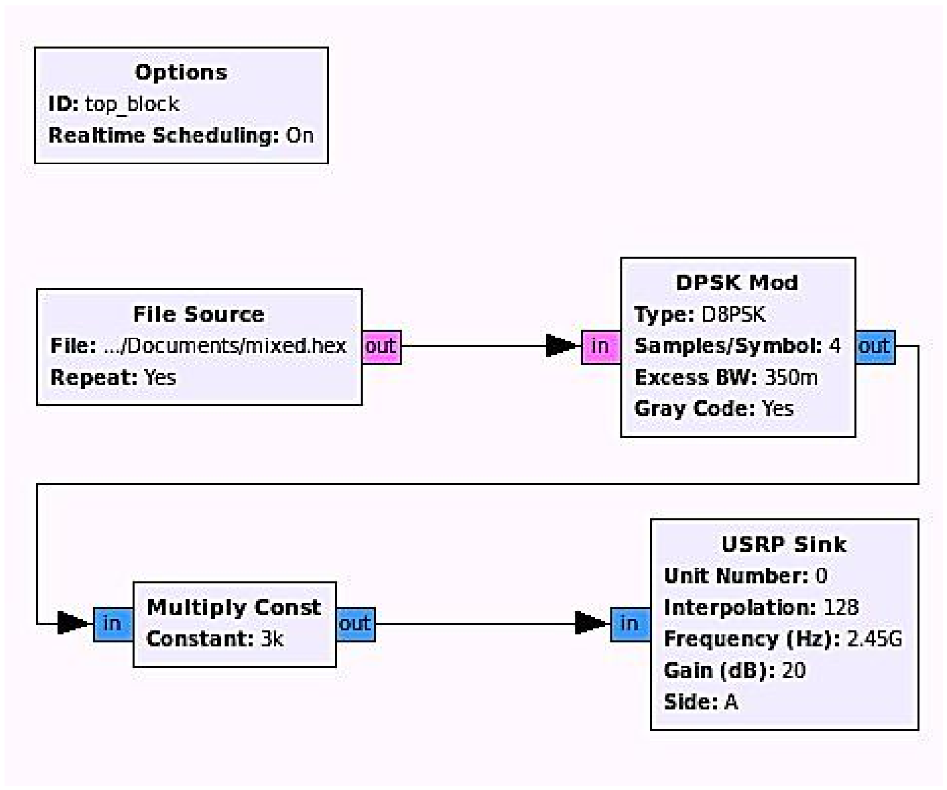
The unaware receiver will try to decode the main signal, recognizing the watermark as noise. An aware receiver using SDR could decode the watermark signal with the required demodulator. It would then decode the "noise" by analyzing the additional phase difference of each symbol. Using this method, the aware receiver would also be able to decode the main signal and the watermark by simply passing the received signal into an M-PSK demodulator of the watermark order.

Unlike the watermarking methods in [6] [7] [8] [11], the proposed method can be implemented using M-PSK modulators and demodulators. The method could be extended to QAM systems as well. Instead of changing the phase angle, the I or Q channel would be altered by one step at a higher QAM order. A 4-QAM signal could be watermarked with a rectangular 16-QAM or higher modulator.

3.3 Implementation

The watermarking scheme was implemented using GNU Radio blocks and Matlab. Matlab is used to encode a randomly generated bit stream with a randomly generated watermark. Two different random bit streams were created: one representing the host signal's data and the other representing the watermark. Two hundred fifty-six bits were used for each. GNU Radio was used to create the flow diagrams that controlled the USRP software radios.

The highest order pair currently available within GNU Radio for testing the watermarking method is 2-PSK and 8-PSK. The encoded bit stream is therefore modulated using an 8-PSK modulator. Three bits of data are used to represent each pair of data and watermark bits. The GNU Radio block flow diagram for the transmitter is shown in Figure 3.



Options
ID: top_block
Realtime Scheduling: On

File Source
File: .../Documents/mixed.hex
Repeat: Yes

DPSK Mod
Type: D8PSK
Samples/Symbol: 4
Excess BW: 350m
Gray Code: Yes

Multiply Const
Constant: 3k

USRP Sink
Unit Number: 0
Interpolation: 128
Frequency (Hz): 2.45G
Gain (dB): 20
Side: A

Figure 3 - GNU Radio Flow Diagram for Watermarked Signal Transmitter

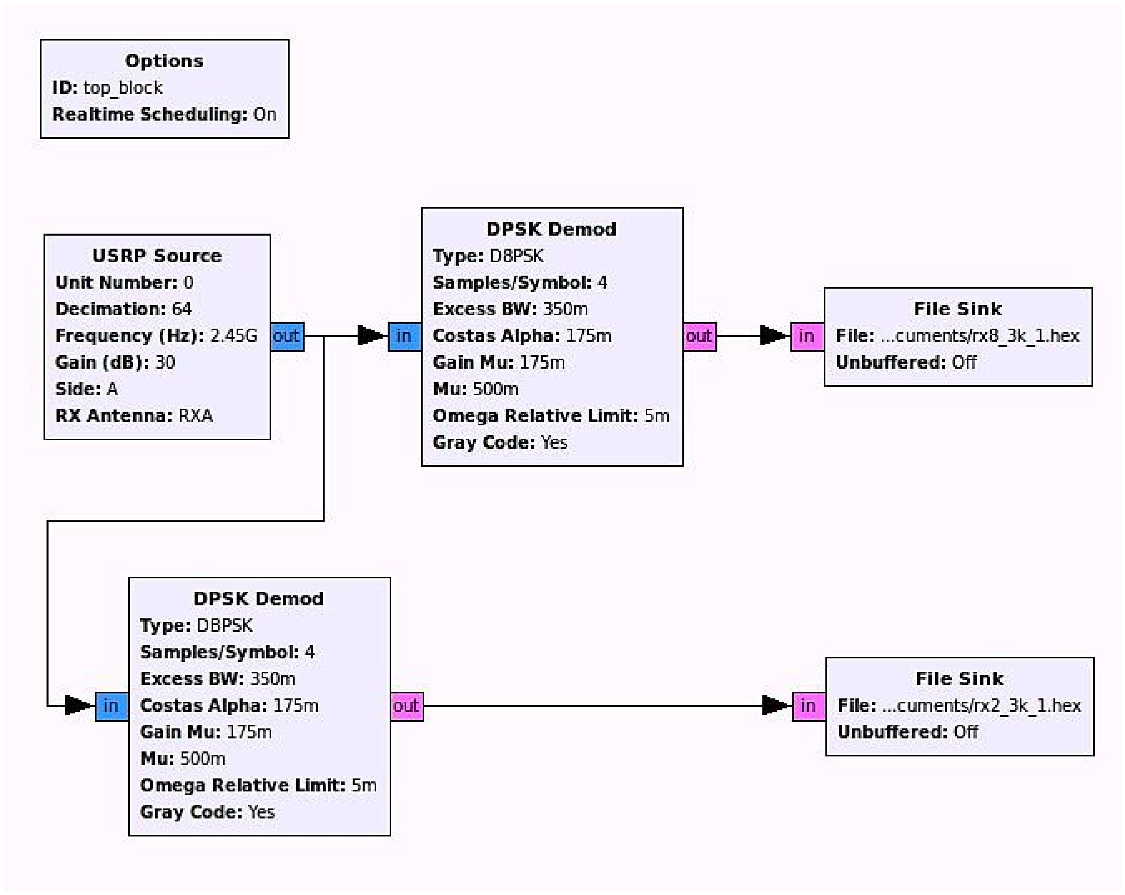
The settings of each block are described below:

- **Options –** ID is an identifier for the flowchart and has no effect; Realtime Scheduling should be on to prevent buffering when the CPU load is too high.
- **File Source –** File is the location of the file to be read for transmission; Repeat should be set to Yes to transmit the bit stream from the file repeatedly. If set to No the block flow will only transmit as many bits as are stored in the file.
- **DPSK Mod –** Type is the modulation type to be used – D8PSK is used for the watermark encoded bit stream, DBPSK would be used to transmit the original bit stream without a watermark; Samples/Symbol controls the number of samples output from the block per M-ary symbol created and 4 is used for all tested transmissions; Excess Bandwidth controls the amount of bandwidth used to transmit the signal and the default of 0.350 is used for all transmissions; Gray Code should be set to yes so that a symbol constellation pattern is used such that misinterpreting the received symbol by only one decision region will only cause one bit to be in error, minimizing the BER.
- **Multiply Const –** Constant sets the amplitude of the signal to be transmitted and can range from 0 to about 32000. Settings of 1500 and 3000 were used for the two power levels transmitted. The values of 1500 and 3000 map to a transmission power of approximately 0.21 mW and 0.83 mW respectively.
- **USRP Sink –** Unit Number is used if multiple USRP's are used on one computer so it is set to the default of 0 since only one USRP is in use on each computer; Interpolation reduces the number of samples taken from previous blocks for transmission – the USRP will take 128,000,000/Interpolation samples per second and Interpolation can range from 4 to 256; Frequency (Hz) sets the center

frequency of the transmission; Gain (dB) has no effect with the USRP daughterboards in use; Side is used to select which daughterboard filters are in use and the default A is used.

Matlab is used to encode the 8-PSK bit stream as shown previously in Table 1. For the watermark aware transmitter the two 256 bit streams are encoded into one 768 bit stream stored in a file named ‘mixed.hex’. The encoded bit stream is fed, by the File Source block, repeatedly to the DPSK Mod block at 750,000 bits per second. The DPSK Mod block then creates the 8-PSK modulating signal. Each group of three bits is turned into one symbol and four samples are created per symbol, so the output is 1,000,000 samples per second. The signal is then interpolated to bring the sample rate to 128,000,000 samples per second and fed to the attached USRP radio via the USRP Sink block. The USRP then broadcasts the signal using a 2.45 GHz center frequency.

The transmitted signal is received by another USRP. The receiver block flow diagram is shown in Figure 4. The top most File Sink block represents the output a watermark aware receiver would process, and the bottom most File Sink block represents the output that watermark unaware receiver would process.


Options
ID: top_block
Realtime Scheduling: On

USRP Source
Unit Number: 0
Decimation: 64
Frequency (Hz): 2.45G
Gain (dB): 30
Side: A
RX Antenna: RXA

DPSK Demod
Type: D8PSK
Samples/Symbol: 4
Excess BW: 350m
Costas Alpha: 175m
Gain Mu: 175m
Mu: 500m
Omega Relative Limit: 5m
Gray Code: Yes

File Sink
File: ...cuments/rx8_3k_1.hex
Unbuffered: Off

DPSK Demod
Type: DBPSK
Samples/Symbol: 4
Excess BW: 350m
Costas Alpha: 175m
Gain Mu: 175m
Mu: 500m
Omega Relative Limit: 5m
Gray Code: Yes

File Sink
File: ...cuments/rx2_3k_1.hex
Unbuffered: Off

Figure 4 - GNU Radio Flow Diagram for Aware and Unaware Receivers

The settings of each block are described below:

- **Options –** ID is an identifier for the flowchart and has no effect; Realtime Scheduling should be on to prevent buffering when the CPU load is too high.
- **USRP Source -** Unit Number is used if multiple USRP's are used on one computer so it is set to the default of 0 since only one USRP is in use on each computer; Decimation reduces the number of samples from the received signal that is output to following blocks – the USRP Source block will output 64,000,000/Decimation samples per second and Decimation can range from 2 to

128; Frequency (Hz) sets the center frequency of the receiver; Gain (dB) has no effect with the USRP daughterboards in use; Side is used to select which daughterboard filters are in use and the default A is used. Rx Antenna selects the input to be used, and the default RXA is used.

- **DPSK Demod -** Type is the modulation type to be used – D8PSK is used for the aware receiver. DBPSK is used for the unaware receiver; Samples/Symbol controls the number of samples output from the block per M-ary symbol created, and 4 is used to match the transmission; Excess Bandwidth controls the amount of bandwidth used to transmit the signal, and the default of 0.350 is used to match the transmission; Gray Code should be set to yes so that a symbol constellation pattern is used such that misinterpreting the received symbol by only one decision region will only cause one bit to be in error, minimizing the BER; Costas Alpha, Gain Mu, Mu, and Omega Relative Limit are synchronization and timing parameters used to demodulate the signal, and all are set to their default except Costas Alpha which was set to 0.050 to minimize synchronization errors.
- **File Sink –** File is the location the demodulated bit stream is stored.

Any signal received by the USRP radio is fed to the GNU Radio application via the USRP Source block. The signal is shifted to baseband, decimated and output to two DPSK Demod blocks. A decimation of 64 is used to make the block output 1,000,000 samples per second to match the sample rate of the signal before transmission. The DPSK Demod block filters, applies automatic gain control, and synchronizes the signal with a costas loop. The block also decodes the received signal constellation to a bit stream. The bit stream is stored to a file using the File Sink block. One DPSK Demod block represents an aware receiver using a D8PSK demodulator. The other represents an unaware receiver using a DBPSK demodulator.

The aware receiver records three bits for each received symbol, representing both the message and watermark, and the unaware receiver records one bit per symbol, representing the message. The recorded bit streams can then be used appropriately. The unaware receiver would use that received data as usual. The aware receiver would analyze the data, time, and received watermark to determine whether the signal was authentic or not.

3.4 Testing Setup

The proposed watermarking method assumes the user will develop a secure, block-based watermark that works best for their system and watermark power level. For each block of host signal bits, the same number of watermark bits can be sent, or the watermark bit rate could be reduced below that of the host signal. If the former, the watermark bit error rate (BER) will likely be higher than that main signal's BER, so some error correction may need to be used for the watermark data, reducing the effective bitrate of the watermark.

To test the new physical layer watermarking method, the BER was tested in a real world wireless environment. The tests were done in an office environment at ranges from eight to thirty feet. At some locations, large metal obstacles were between the transmitter and receiver. This created the type of multipath environment that most wireless devices must be able to operate within.

The watermarking method was tested at seven distances. The receiver was left at one location while the transmitter was relocated for all distances. Positions were marked with tape with so testing locations were not changed during development. Identical, omnidirectional antennas were used on both radios and remained oriented the same direction relative to the radio for all tests. The transmitter power remained constant for each location. A map of the environment is shown below in Figure 5.

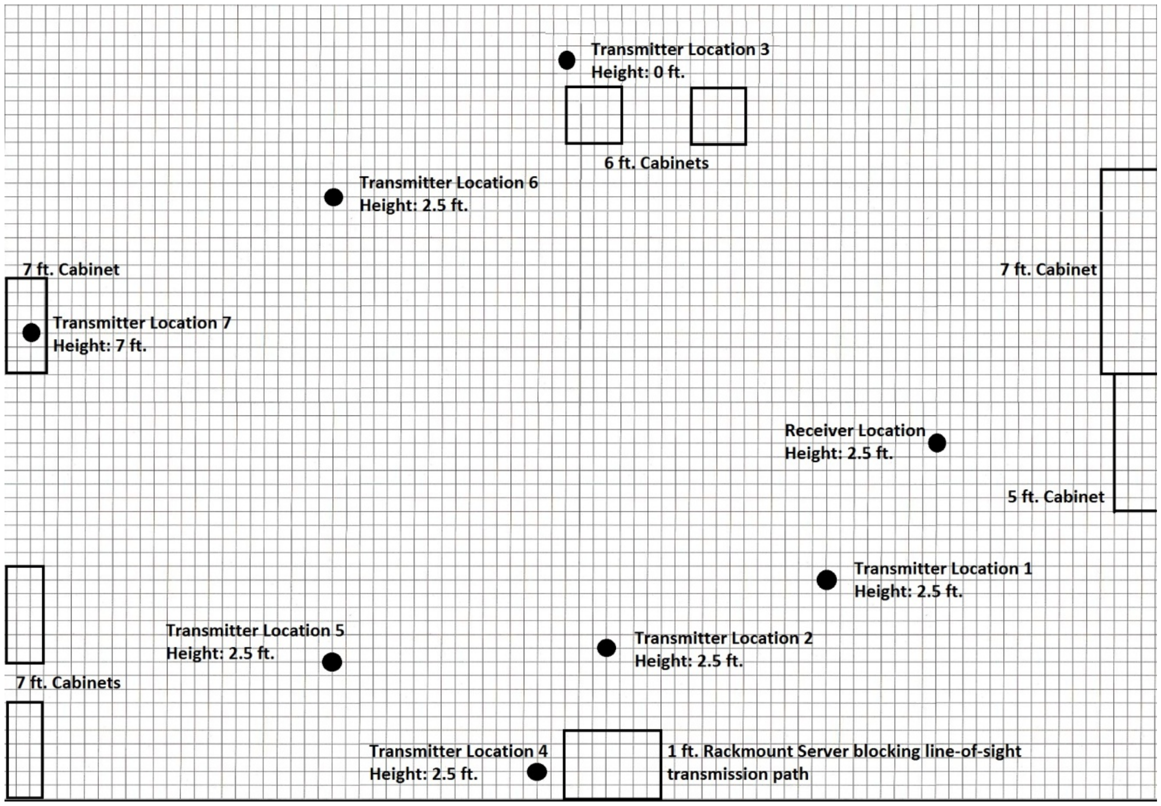


Figure 5 - Test Environment

The map shows the approximate location of each transmitter location, the receiver location, and large metal objects likely to cause multipath interference. Each grid block is 6 inches by 6 inches. The rectangular room was 29 x 42 feet.

The following list identifies the four important BER's to identify at each distance:

- Bit error rate for unaware reception of non-watermarked signal
- Bit error rate for unaware reception of watermarked main/host signal
- Bit error rate for aware reception of watermark
- Bit error rate for aware reception of the watermarked main/host signal

With this information, the viability for the watermark can be determined. In terms of the BER for the unaware receiver, the DBPSK and D8PSK pair is the worst case that should be used for this watermarking method. This gives a great deal of power to the watermark and in so will deteriorate the reception of the host signal significantly. In terms of the watermark BER, the pair is the best case. The D8PSK watermark modulation provides the most watermark power possible, using standard MPSK offsets, without completely degrading the host signal. A DQPSK modulation would provide more watermark power, but it would cause the minimum expected BER of the unaware receiver to be 25%. Using a D16PSK watermark would decrease the host signal degradation and increase the watermark BER.

Matlab was used to randomly generate two individual two hundred fifty-six bit data streams: one to represent a host data stream and one to represent a watermark. Matlab was then used to create a seven hundred sixty-eight bit stream – the watermark encoded bit stream. The bit streams can be seen in Appendix E. The Matlab code that encodes the watermarked 8-PSK bit stream can be seen in Appendix A.

The watermark encoded bit stream is mapped onto a D8PSK waveform, transmitted from each location, received, demodulated, and stored in files. To store the bit streams, the receiver block flow is run for about sixty seconds. Matlab is used to analyze the received bit streams. The received bit streams are compared with the original transmitted bit streams to find each of the three BER’s. The Matlab code to find the unaware message BER and watermark BER is located in Appendix B. It should be noted that the first several bits of each bit stream are dropped to eliminate any data that is recorded before synchronization occurs.

The watermark aware receiver code, in effect, slides the original bit stream block across the entire received bit stream. At each position, the number of matching bits is determined. Because of the randomness of the bit streams, the matching bits will always be around fifty percent of the length of the original bit stream until the block is in line with the same received data block, assuming the block length is long enough. In this manner the received data indices are recorded that correspond to the start of a new block of data. This allows the code to ensure there received data has the correct number of bits per block. Synchronization errors can cause bits to be dropped or added, which would cause problems when calculating the BER. If there is one extra bit in a block, the last bit is removed. Otherwise, the block is padded with zeros to ensure it is the correct length.

After removing the first bits from the beginning of the received data and generating blocks with the correct number of bits, the BER is calculated by moving the transmit block from the beginning of the received data to the end two hundred fifty-six bits at a time. All non-matching bits are summed and divided by the length of the received data to find the BER. The code used to find the bit error rates can be found in Appendix B and C.

The BER’s are recorded and analyzed to determine the predictability and viability of the watermarking method. The bit error rate for unaware reception of non-watermarked signal is used as the control for the multipath environment.

The next chapter shows and analyzes the results of testing the proposed watermarking method.

CHAPTER IV

FINDINGS

The results of the watermarking method are discussed in this chapter and compared to the expectations. The viability of the watermarking method is also discussed.

4.1 Expectations

The proposed watermarking method will be evaluated based on bit error rate (BER) which can be fairly straightforward to predict for PSK systems. However the BER is more difficult to predict for modified DPSK systems the method tested. Figure 6 shows the theoretical BER for several M-ary DPSK signals in an AWGN environment. The figure was generated using the bertool GUI in Matlab. The GUI uses the berawgn function which is documented [15] as using the following closed form approximation to generate the plot:

$$P_b = \frac{1}{k}\left(\sum_{i=1}^{M/2} (w_i^{'}) A_i \right)$$

where $w_i^{'} = w_i + w_{M-i}$, $w_{M/2}^{'} = w_{M/2}$, w_i is the Hamming weight of bits assigned to symbol *i*, M represents the number of symbols, k represents the number bits per symbol, and:

$$A_i = F\left((2i+1)\frac{\pi}{M}\right) - F\left((2i-1)\frac{\pi}{M}\right)$$
$$F(\psi) = -\frac{\sin\psi}{4\pi}\int_{-\pi/2}^{\pi/2}\frac{\exp\left(-kE_b/N_0(1-\cos\psi\cos t)\right)}{1-\cos\psi\cos t}dt$$

A special case for M=2 is used:

$$P_b = \frac{1}{2}\exp\left(-\frac{E_b}{N_0}\right) \qquad (1)$$

This simplified expression for the M=2 case (DBPSK) can also be found in [16].

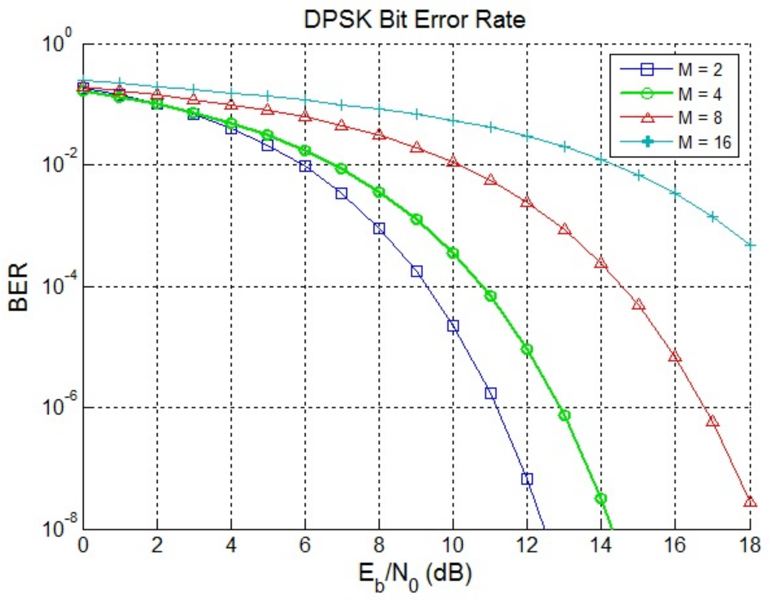


Figure 6 - DPSK Bit Error Rate for Several M-ary Signals

Theoretically a DBPSK signal with an E_b/N_o of 11.1 dB has a BER of about 10^{-6}. A D8PSK signal with the same E_b/N_o has a BER of about $5x10^{-3}$. This increase in BER is acceptable in some systems because the raw bitrate is increased by a factor of three.

The proposed watermarking system will be tested by encoding a D8PSK watermark onto a DBPSK signal. Therefore an unaware receiver will demodulate with a DBPSK GNU Radio block and the aware receiver will demodulate with a D8PSK block. The expected performance of the watermarking method can be analyzed by examining how the constellation is changed by encoding a watermark.

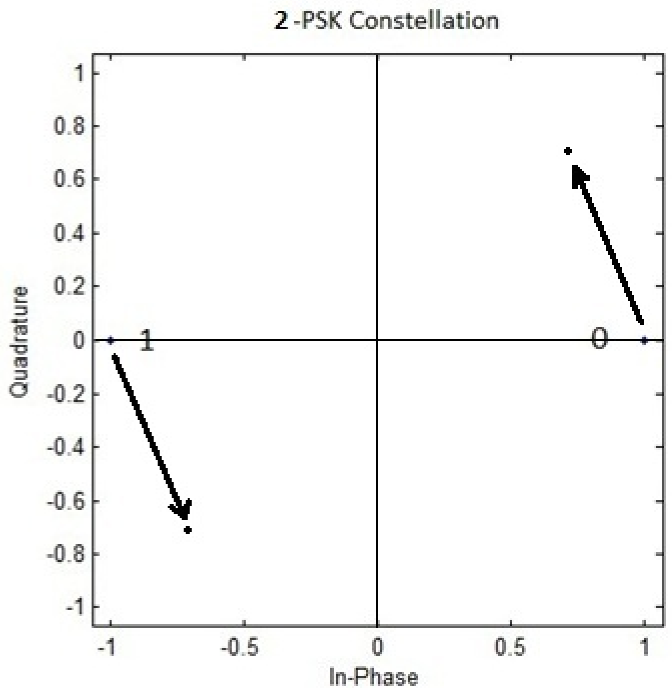


Figure 7 - Effect of Watermark Logic 1 on BPSK Constellation

The constellation change is shown in Figure 7. A watermark bit of 1 will shift the phase by forty-five degrees. Since DPSK is being used, it is important to consider the bit to bit constellation change. Because only a watermark bit of 1 changes the constellation, there are four distinct situations that need to be examined:

- Watermark changes from 0 to 1
- Watermark changes from 1 to 0
- Watermark remains 0
- Watermark remains 1

With a random set of data each of these situations has an equal (25%) probability of occurring for each transmitted symbol. A ‘watermark change from 0 to 1’ or a ‘watermark remains 1’ situation will cause a forty-five degree phase change that is unexpected by the unaware receiver. This means 50% of watermarked symbols are altered before transmission. To evaluate the effect this has on the BER of the received signal, the constellation of transmitted symbols is examined analytically and visually using trigonometry.

Analytically, the general form of bit rate error for a BPSK signal can be used to determine the effect of altering the symbols used to transmit each bit. The bit error rate of a BPSK signal using a matched filter detector is [16]:

$$BER = Q\left(\sqrt{\frac{A^2T}{N_O}\left(\frac{1-\cos\theta}{2}\right)}\right) = Q\left(\sqrt{\frac{2E_b}{N_O}\left(\frac{1-\cos\theta}{2}\right)}\right) \quad (2)$$

where θ is the number of degrees each symbol is separated by. The formula assumes that the matched filter detector uses an optimum threshold, one that is in the middle of each possible symbol, when determining which symbol was transmitted. The formula (2) for BPSK shows how the effective power of a transmission is reduced by changing the separation angle of symbols. As the separation angle decreases, the distance from the decision line decreases, increasing the probability that noise will make the symbol in error. Several examples of possible symbol angle separations are shown in Figure 8.

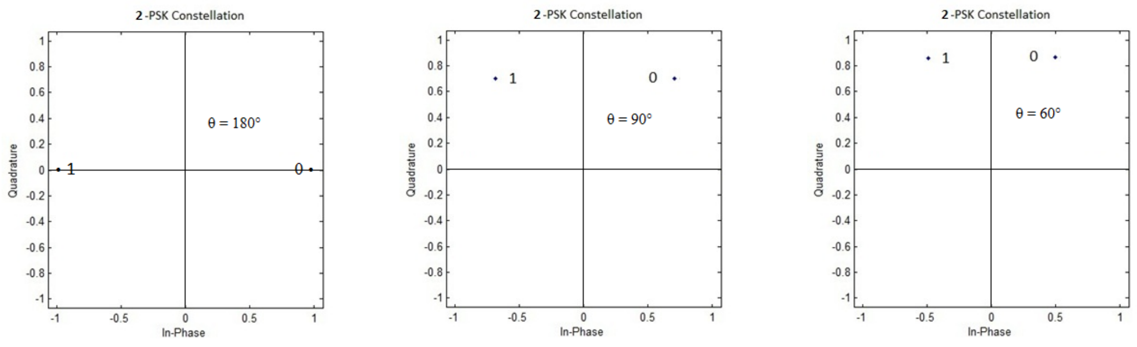


Figure 8 - BPSK Constellations With Varied Symbol Seperation

Given that each symbol is transmitted at the same amplitude, shown as 1, the BER formula (2) yields the maximum effective power, P_{max}, when θ is 180 degrees. With a 90 degree separation the effective power is $P_{max}/2$, and with a 60 degree separation the effective power is $P_{max}/4$. The BER formula (2) assumes the two symbol constellation points are equidistant from the decision line (the y-axis) and only considers the shortest distance to the decision line, so the effective power could also be found using trigonometry. To find the effective amplitude of the modified symbols, the distance of each symbol from the decision line is found. The effective amplitude with 180 degree separation is 1.00, with 90 degree separation it is 0.707, and with 60 degree separation it is 0.50. The effective power for the three is therefore proportional to 1.00, 0.50, and 0.25 respectively. The trigonometric analysis yields the same results to determine effective power and can be extended to the symbol modification used by the proposed watermarking method if implemented in a PSK system. However the proposed watermarking method has been implemented using DPSK which requires additional analysis to estimate the expected BER for the watermarked signal.

The proposed watermarking method causes half of the host signal bits to be transmitted at an altered phase angle. The DBPSK receiver compares the phase of the current symbol to the phase of the previous received symbol. The zero degree reference is set at the phase angle of the previous symbol. That previous symbol can be thought of as the 0 point in Figure 7. The decision line for the current symbol is the quadrature axis (y-axis). Anything left of the axis is a 1 and anything to the right is a 0. This means a phase change of -90 to 90 degrees will be interpreted as 0 and a phase change of 90 to 270 degrees will be interpreted as 1. Moving parallel to the quadrature axis would have no effect on the decision of the BPSK receiver, but for a DBPSK receiver the movement causes some additional error. A DPSK system treats the previous received symbol, specifically its phase angle, as the reference for the current symbol. If the previous symbol's phase angle is changed 20 degrees due to the effects of noise, the decision line for the current symbol will be rotated 20 degrees. This rotation will be unaccounted for in the transmission of the current symbol. This has the effect of increasing the probability of a bit error by creating symbol decision regions that are less than optimal. The change is shown in Figure 9.

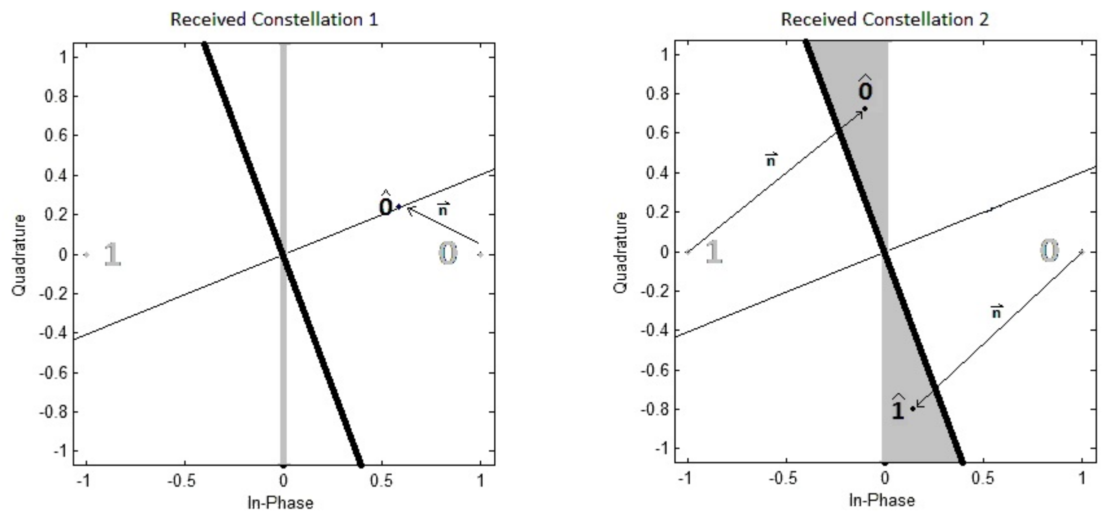


Figure 9 – Example DBPSK Decision Region Change Due to Noise

The left side of Figure 9 shows the effect of the first received symbol. A DBPSK transmitter may be transmitting only symbols with a 0 or 180 degree phase so the ideal decision line is the y-axis. A symbol is shown being received with a 20 degree phase due to noise. The DBPSK receiver will use this phase as a reference, rotating the decision line 20 degrees. The shaded regions on the right side of the figure would now be decoded in error. A received signal vector in the top shaded area would be decoded as a 0 when it is actually a 1, while a vector in the bottom shaded area would be decoded as a 1 when it is actually a 0. If the first symbol had not been affected by noise, the second symbol could have had a noise induced phase change of -90 to 90 degrees without a bit error. With the changed decision line, the second symbol can have a noise induced phase change of -70 to 110 degrees without an error. This range is less than optimal and will result in a higher probability of bit error.

4.1.1 Estimation Methods

For a PSK system, the most straightforward method of estimating the BER for normal signals or modified signals is to find the probability of bit error in the same manner as equation (2). In vector space, for BPSK, a symbol is transmitted at X = -1 to represent a 1 bit and X = 1 to represent a 0 bit as shown in Figure 10. The spikes here represent delta functions, each with probability ½.

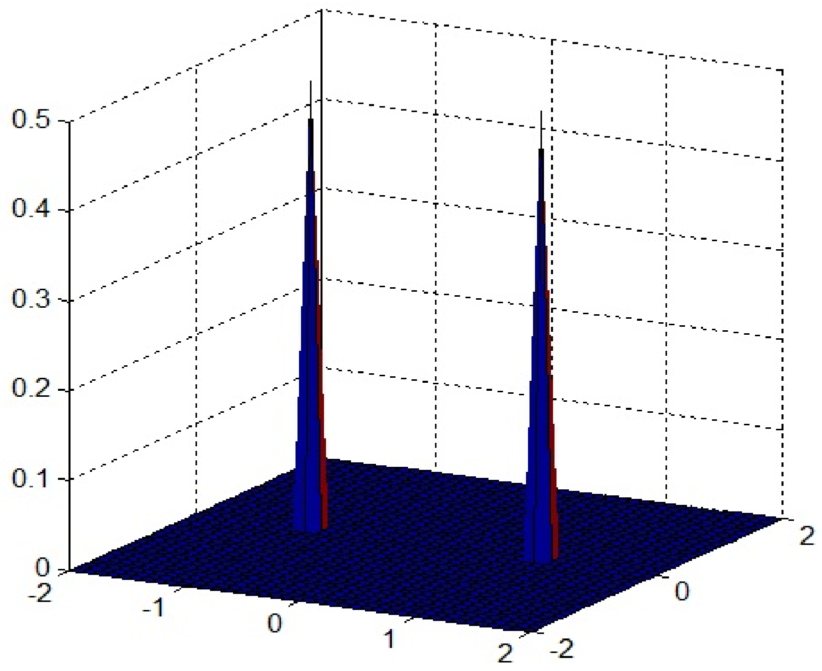


Figure 10 - Transmitted BPSK Spectrum

For equation (2) it is assumed the noise added by the channel is additive white Gaussian noise (AWGN). Noise has the effect of adding a vector of random orientation and length to the symbol. Then the distribution of the received signal is similar to the Matlab plot shown in Figure 11. The ratio of A^2/σ^2 is proportional to E_b/N_o. A, the amplitude, controls how far the centers of the Gaussian volumes are apart. σ^2, the variance, sets the spread of the Gaussian volume.

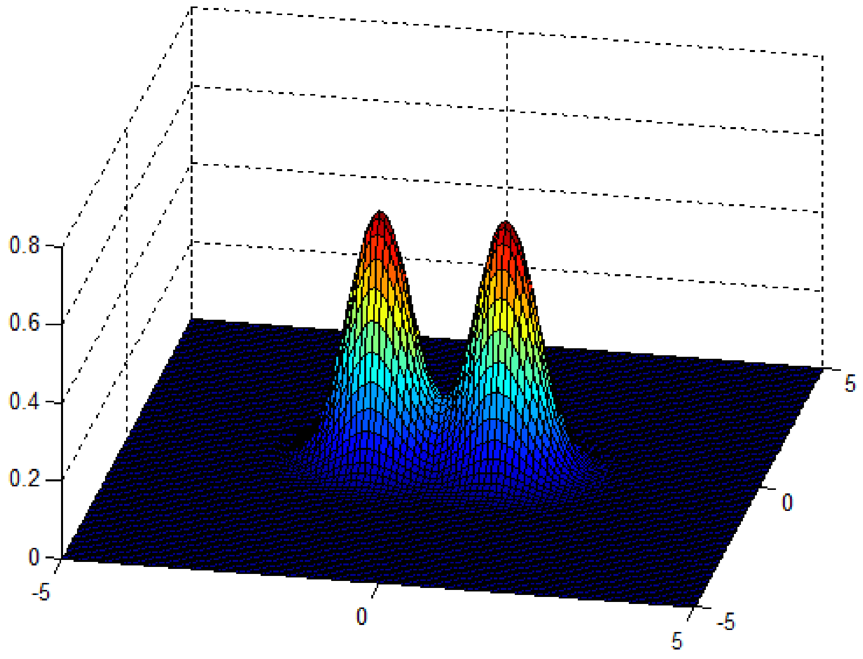


Figure 11 - Received BPSK Distribution

To find the BER, the volume of each Gaussian is integrated in the incorrect decision regions. For this BPSK example there are two regions. They are separated by the Y axis at 0 which runs between the two peaks. Assuming negative logic, for a Logic 1, the BER is found by integrating the volume of the left Gaussian that is on the right side of the decision line. And for the Logic 0, the BER is found by integrating the volume of the right Gaussian on the left side of the decision line. If transmission symbols are modified, the BER can be determined by placing Gaussian volumes at the correct location to represent the change. This vector model can be used for QPSK and higher modulation orders as well. Each modulation order will have different decision regions. This method works for coherent PSK assuming the receiver generates an accurate phase estimate from the received signal, which will occur under normal operation. It will not predict the DPSK BER however because of the difference in how a DPSK demodulator makes decisions.

As has been noted previously, DPSK can demodulate non-coherently because it does not lock to one reference phase. Instead each received symbol's phase angle is compared to the last. This means each received symbol determines the decision regions for the next symbol. The received noise will cause the decision regions to change randomly. The decision lines will rotate about the origin with some random distribution. This distribution can be used to predict the number of bit errors at each E_b/N_o level.

To create estimations for the BER of the proposed watermark system, a Monte Carlo simulation of the system was created. Two individual Monte Carlo simulations were performed at multiple E_b/N_o levels:

- Phase difference error distribution of symbol with no unexpected shift
- Phase difference error distribution of symbol with unexpected 45 degree shift

A DPSK constellation is decoded by taking the phase difference of the current and previous symbols. A DBPSK decoder will expect 0 or 180 degree phase differences in ideal, no noise situations. To estimate the BER based on the effects of AWGN and watermarking, the simulation finds the estimated distribution of phase difference errors seen by a receiver. That is, it finds the phase difference caused only by AWGN and, in the case of the unaware receiver, the watermark. To simplify the simulation, the phase difference contribution from symbol phase changes is not considered; the simulation assumes every transmitted message symbol is a Logic 0. Due to symmetry, the results are the same as if random message 0's and 1's were transmitted.

To implement the first simulation, with no unexpected watermark shift, two arrays were created to represent the I and Q channels of 100,000,000 symbols. Both were made up of statistically independent, Gaussian distributed, randomly generated numbers with a variance of 1. The I channel's Gaussian distribution had a mean of the simulated received signal peak amplitude and the Q channel's Gaussian distribution had a mean of 0. The two arrays were then combined into one array of complex numbers: the I channel as real and the Q channel as imaginary. The vector array at this point would represent a series of unwatermarked symbols, contaminated with additive white Gaussian noise (AWGN), at the receiver.

To implement the second simulation, another array of equal length was generated. This time the vector array was created without the addition of noise. Every sample in the Q channel was equal to 0 and every sample of the I channel was equal to the simulated signal's peak amplitude. A random set of 1's and 0's of the same length was then generated to determine where to add an unexpected 45 degree shift. At each chosen location that sample and all future samples were shifted by 45 degrees by multiplying the symbols by: $\left(\frac{1}{\sqrt{2}}+\frac{1}{\sqrt{2}}i\right)$. This had the effect of rotating the current and all future samples by 45 degrees without changing their amplitude. This simulates the behavior of the transmitter of a watermarked signal. At this point, the vector array represented the transmitted signal. Then a vector array of 0 mean, normally distributed, Gaussian noise was added to both the I and Q channel of the array. Now the vector array represented the received signal. Both sets of data are then analyzed to create an expected phase distribution. The Matlab code for the Monte Carlo simulation is located in Appendix D.

For each set of random symbols (shifted and non-shifted), another array of data was created containing the phase change of every adjacent pair of symbols, simulating a DPSK receiver. The phase change included any unexpected effects from random noise or watermarking. Example histograms showing the distribution of phase angle differences are shown for both simulations in Figure 12.

In the simulation, the signal vector lengths represent the peak voltage A of a zero mean sinusoid DPSK signal, with an average power of $A^2/2$ watts. The two statistically independent Gaussian distributed noise values added to the I and Q channels generate noise power equal to the sum of power in the I and Q channels [17]. The SNR of the input signal for these simulations is therefore $A^2/(4\sigma^2)$, where σ^2 is the variance of the Gaussian noise. The variance is held constant at 1. A is varied to simulate multiple SNR values.

The simulation was performed in a manner similar to the model used in the derivation of MDPSK BER expressions in [18]. Gaussian noise was added to a vector representation of a received signal. In [18] a special form of SNR called "instantaneous SNR" was defined as $P^2/(2\sigma^2)$, where P was the length of the vector used to represent a sinusoidal signal. Instantaneous SNR was therefore equal to twice the traditional SNR. To compare the signal SNR with E_b/N_o [18] defined a "Symbol SNR" equal to the "Instantaneous SNR". The E_b/N_o was found as:

$$E_b/N_o = \text{(Symbol SNR)/(\# bits per symbol)}$$

E_b/N_o was equal to the Symbol SNR per bit, which was then used to determine bit error rates for MDPSK signals.

The Monte Carlo simulation produced bit error rates in accordance with the E_b/N_o figure in [18]. The controls of the simulation, a typical DBPSK and typical D8PSK signal, confirmed that:

$$E_b/N_o = \text{Bit SNR} = \text{(Symbol SNR)/(\# bits per symbol)} = \text{(SNR * 2)/(\# bits per symbol)}$$

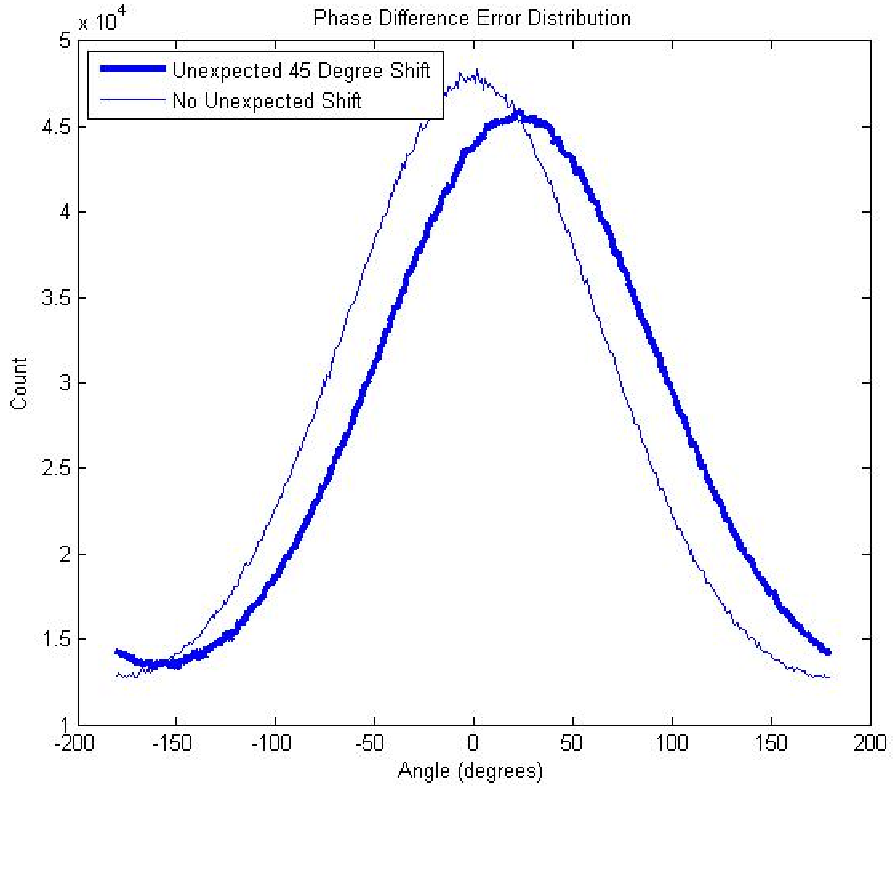


Figure 12 – Example Phase Difference Error Distribution

Figure 12 shows the phase difference error distributions of two signals received with equal E_b/N_o. One signal is a typical signal with AWGN contamination. The other has an unexpected 45 degree shift added to half of its symbols before AWGN is applied. The unexpected shift distorts the phase difference error distribution; shifting at what degree the most phase errors occur. It is also noticeable that the number of symbols that have an error over 90 or below -90 degrees is increased. This will lead to an increase in BER for a DBPSK receiver.

With no shift added, the simulation gives the phase difference error distribution of a typical DPSK system. And in the case of the proposed watermarking method it also gives the phase error distribution of a watermarked signal as received by an aware receiver. Because the aware receiver will be expecting 45 degree shifts to represent the watermark, there is no need to consider these shifts in the phase as errors. The no shift added simulation can then be used to estimate the following:

- BER of typical DBPSK signal
- BER of watermarked message by an aware receiver
- BER of watermark by an aware receiver
- BER of typical D8PSK signal

With the shift added simulation, the phase difference error distribution of a watermarked signal received by an unaware receiver is found. This simulation can be used to find the BER of a watermarked message by an unaware receiver.

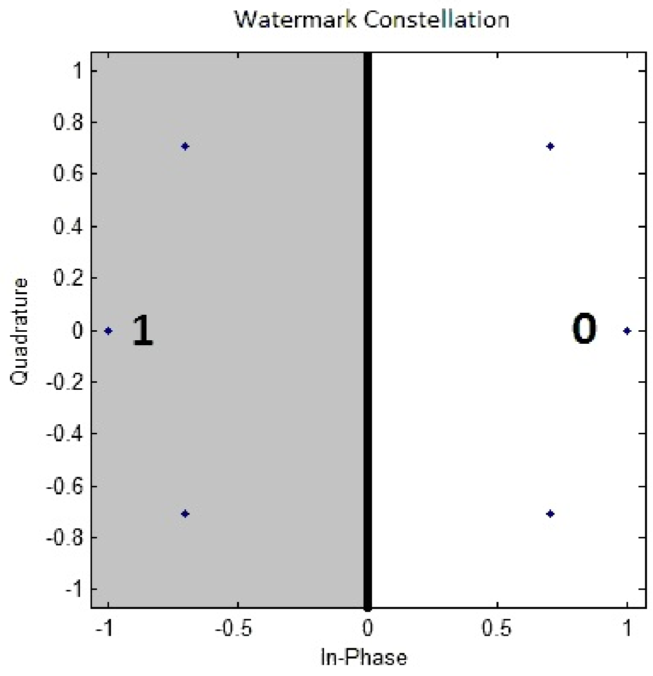


Figure 13 - Unaware Message Regions (Current Phase – Previous Phase)

For a normal DBPSK signal, the phase error regions of 90 to 270 degrees cause one bit error. A logic 0 will be decoded as a 1 and a logic 1 will be decoded as a 0. To estimate the BER of an ordinary DBPSK signal the non-shifted phase difference error histogram is summed from 90 to 270 degrees and divided by 100,000,000: the number of bits in error divided by the total number of bits.

A BER estimate for a typical D8PSK transmission can then be found by summing the total number of incorrect bits that would be produced for each incorrect 45 degree wide region. Figure 15 below shows these regions. Each D8PSK symbol has 7 possible incorrect regions. Assuming a Gray Code is used to minimize the number of bit errors should a symbol error occur, the phase error regions are listed below with the number of bits in error:

- 22.5 – 67.5 degrees: 1 bit
- 67.5 – 112.5 degrees: 2 bits
- 112.5 – 157.5 degrees: 3 bits
- 157.5 – 202.5 degrees: 2 bits
- 202.5 – 247.5 degrees: 1 bit
- 247.5 – 292.5 degrees: 2 bits
- 292.5 – 337.5 degrees: 1 bit

These phase error regions are negative for some D8PSK symbols and positive for others, but the phase difference error distribution is symmetric about 0 for a received signal with no unexpected shift (other than AWGN) – so these ranges can be used for any possible D8PSK symbol. The number of symbols in each phase error region in the Monte Carlo simulations is multiplied by the number of bits in error in that region. The total is then divided by 300,000,000, the number of bits transmitted by the 100,000,000 D8PSK symbols simulated.

The results of the Monte Carlo simulations of typical DBPSK and D8PSK signals with AWGN are shown below.

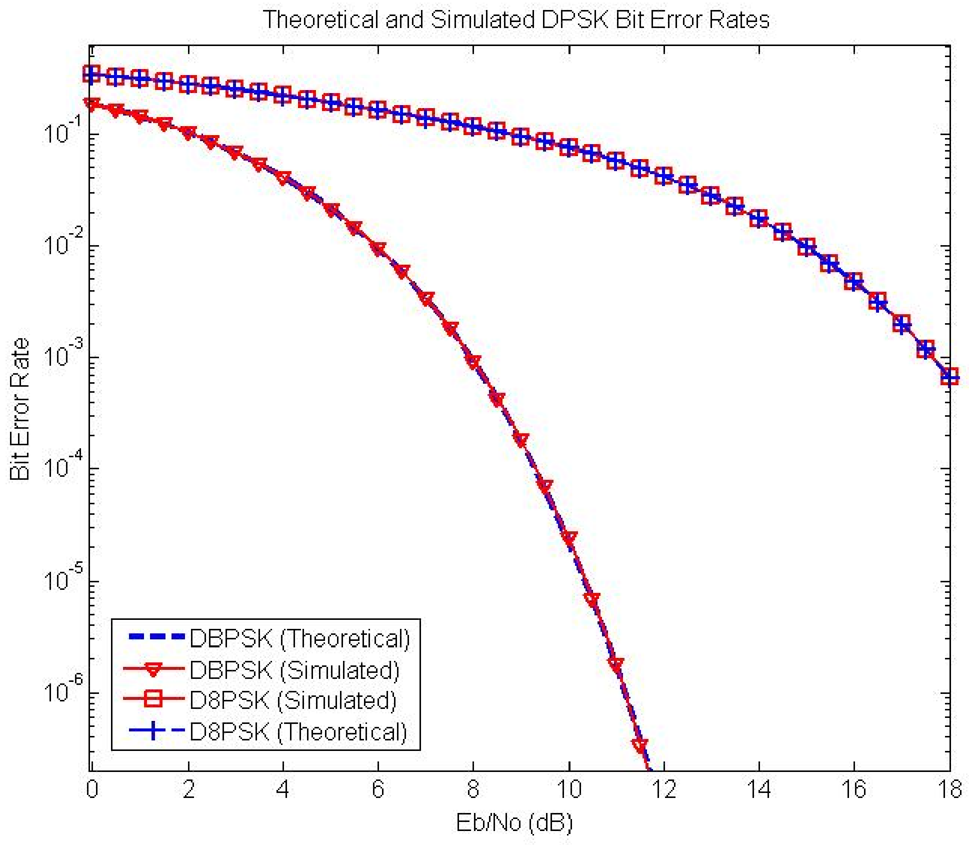


Figure 14 - Monte Carlo Simulation Results for Typical DBPSK and D8PSK Signals

Figure 14 shows the results of the Monte Carlo simulation for typical DBPSK and D8PSK signals. The simulation results show that, since E_b/N_o equals half the simulated SNR for DBPSK simulations, and E_b/N_o equals 1/6 of the simulated SNR for D8PSK simulations, the method can accurately predict bit error rates for standard modulation techniques, and hence can also be expected to do so for the proposed watermarking method's testing range. Note that for D8PSK, when the symbol energy equals the bit energy of a DBPSK signal, the D8PSK bit energy is 1/3rd that of DBPSK as the D8PSK symbol energy must be spread over three bits.

As BER's drop down towards 10^{-6} the simulation loses accuracy due to the limited number of samples.

A BER estimate of a watermarked DBPSK signal received by an unaware receiver is found by summing the bits over the same ranges (90 to 180 and -180 to -90) using the shifted phase difference histogram.

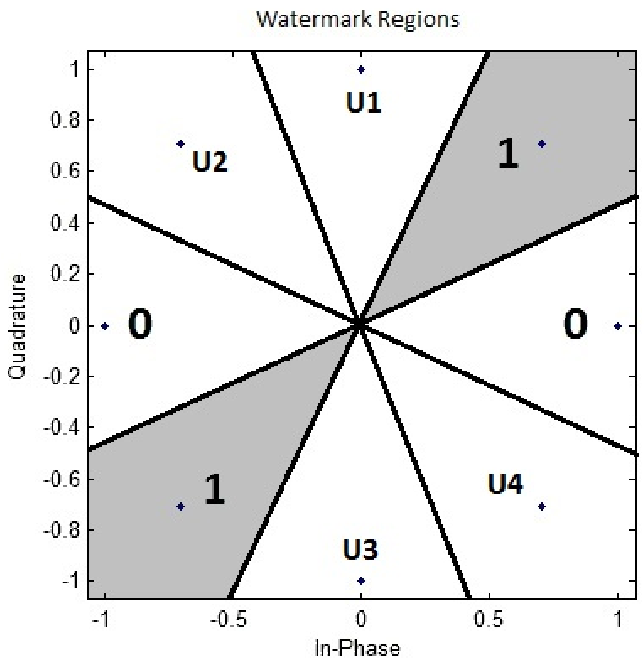


Figure 15 – D8PSK Decision Regions

Figure 15 shows the phase difference regions for D8PSK symbols. The implemented watermarking transmitter used four of the eight possible D8PSK symbols. Ideal watermark logic 1 regions are labeled with a 1 and ideal logic 0 regions are labeled with a 0. Regions U1 - U4 are regions that will only be received when there is a D8PSK symbol error. Because errors are more likely to occur in the nearest regions, the receiver watermark decision regions can be optimized as shown in Figure 16.

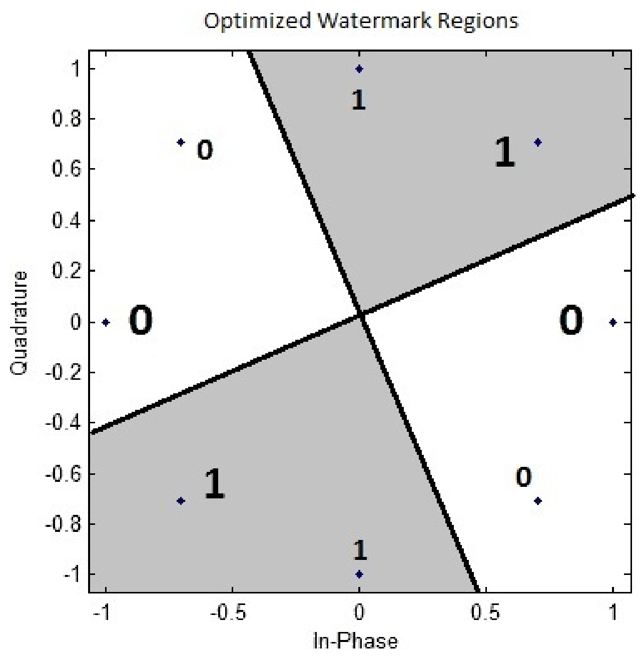


Figure 16 - Optimized Aware Watermark Regions

The unknown regions are set to be recognized as the nearest valid watermark symbol. For example, if a watermark aware D8PSK receiver identifies the received difference vector as being in region U1, this region is closer to a watermark 1 than a watermark 0, and hence should be interpreted as likely being a watermark logic 1. This optimization leaves two phase difference error regions that will cause any watermark bit to be in error: 22.5 to 112.5 degrees and -157.5 to -67.5 degrees. Summing these regions in the phase difference error histogram yields the estimated BER for the watermark received by an aware receiver.

In the same sense that the watermark symbol regions are optimized, the original message symbol regions for an aware receiver can be optimized. Figure 17 shows the optimized regions.

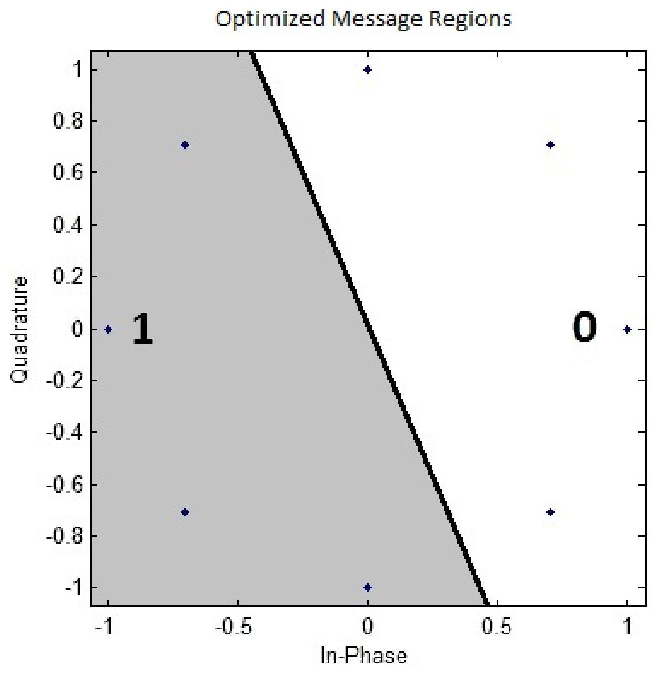


Figure 17 - Optimized Aware Message Regions

For an unwatermarked DBPSK message being received by a D8PSK receiver, the regions marked as U1 and U3 in Figure 15 (centered at 90 and -90 degrees) are unknown. They have an equally likely chance of being received after a 0 or 180 degree phase change. However since the watermarked message is sometimes shifted 45 degrees, U1 will more likely be the result of a message 0 and the U3 the result of a message 1. The BER for a watermarked message decoded by an aware receiver can then be estimated by summing the non-shifted phase difference error histogram in the ranges 112.5 to 180 and -180 to -67.5 degrees.

Figure 18 shows the BER estimates produced by the Monte Carlo estimation method.

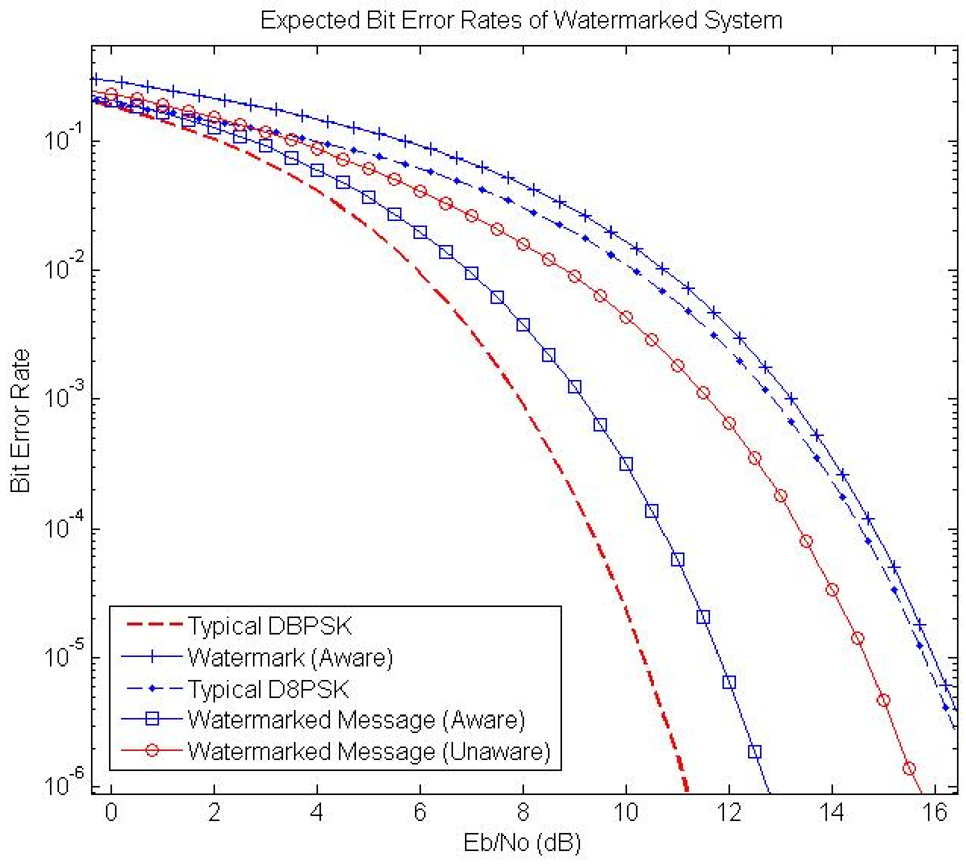


Figure 18 - Monte Carlo Simulation Estimated Bit Error Rates

The simulation shows that the watermark will significantly degrade the original message signal. A D8PSK watermark will cause around 4 dB in loss to a DBPSK host signal. An aware receiver is able to decode the message with less apparent signal degradation by using optimized decision regions.

The watermark BER, as might be expected from a rough estimation, is about 3/2 times the BER of a typical D8PSK signal. This occurs because even though the watermark is implemented via D8PSK, there is only one bit per watermark symbol as opposed to three bits per symbol which would be expected in a normal D8PSK signal. Most D8PSK symbol errors result in one of the three bits being decoded in error. This accounts for the factor of three. About half of the symbol errors that occur will cause that watermark bit to be in error. Most symbol errors are due to an error one symbol region away. The watermark symbol regions are set up so that a symbol error one region to one phase direction will cause a bit error, but an error one region in the other phase direction will not. This accounts for the factor of ½.

The Monte Carlo BER estimations will be compared with the experimental results in the results section.

4.2 Results

The results of the proposed watermarking method are shown in this section. As described in the ‘Testing Setup’ section in Chapter III, the watermark method was tested at each of the seven locations. The transmissions were performed at 250,000 symbols per second. Data was collected by running the transmitter block flow diagram at each of the seven locations. The receiver block flow diagram was run for approximately one minute so that approximately 15,000,000 data symbols were collected for each of the three tests for each location. The collected symbols reflected about 15,000,000 bits for each of the following tests: unaware reception of the un-watermarked signal, unaware reception of the watermarked signal, and aware reception of the watermark. The BER for each set of data was then determined by comparing the known, repeated sequence to the received data as previously discussed in the Testing Setup section. Plots were made for each test to show the BER at each location.

Two transmission power levels were used to test the physical layer watermarking method. The XCVR2450 daughterboard in use with the USRP supports a transmit power up to 100 mW (20 dBm) [12]. The signal amplitude is controlled by the Multiply Const block in the block flow diagram. The value can range from 0 to 32767 (-∞ to 20 dBm). The work in [19] showing the measured received power from a USRP transmission verifies that the transmission amplitude varies approximately linearly with the multiplier. Each time the constant is doubled, the power received is increased by a factor of four (6 dB).

To estimate the transmission power used in the physical layer watermarking experiment, the transmitter multiplier values used (1500 and 3000) were compared to the maximum value of 32767. The power gain caused by a change in amplitude is:

$$G_p = 20 \times log_{10}\left(\frac{A_2}{A_1}\right) \quad (4)$$

where A_1 and A_2 are the original amplitude and new amplitude respectively. So when the Multiply Const block is set to 1500 (A_2 = 1500, A_1 = 32767) the power gain is -26.8 dB, and when set to 3000 (A_2 = 3000, A_1 = 32767) the power gain is -20.8 dB. The approximate transmit power can be found by applying these power gains to the maximum transmit power of the daughterboard. Therefore the approximate transmit powers for the 1500 and 3000 levels are -6.8 dBm (0.21 mW) and -0.8 dBm (0.83 mW) respectively.

Because the spectrum in use is shared by Wi-Fi and other devices, interference occasionally sometimes caused the noise floor to be increased or the signal to be overridden by another. Figure 19 shows a received spectrum, averaged, with the peak amplitude held by the plot.

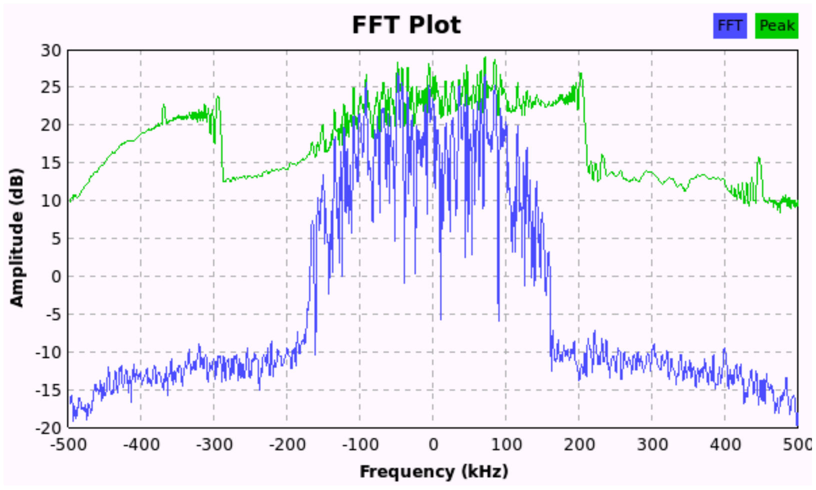


Figure 19 - Location 7 FFT With Peak Amplitude Held Constant

The above figure shows that interference levels can be as high as the received signal. A typical transmission from testing is shown by the blue line. The green line shows the maximum levels seen by the receiver over the one minute test period. The interference combined with the effect of multipath fading caused some irregularities in the BER of tested data.

The first set of data was taken at the lower power level of ~0.21 mW to show the results of the proposed watermarking method near the edge of failure. Figure 20 shows the BER of the host signal without an encoded watermark. Figure 21 shows the BER of the watermarked signal demodulated by the unaware receiver. Figure 22 shows the BER of the watermark when demodulated by the aware receiver. Figure 23 shows the BER of the watermarked signal demodulated by the aware receiver.

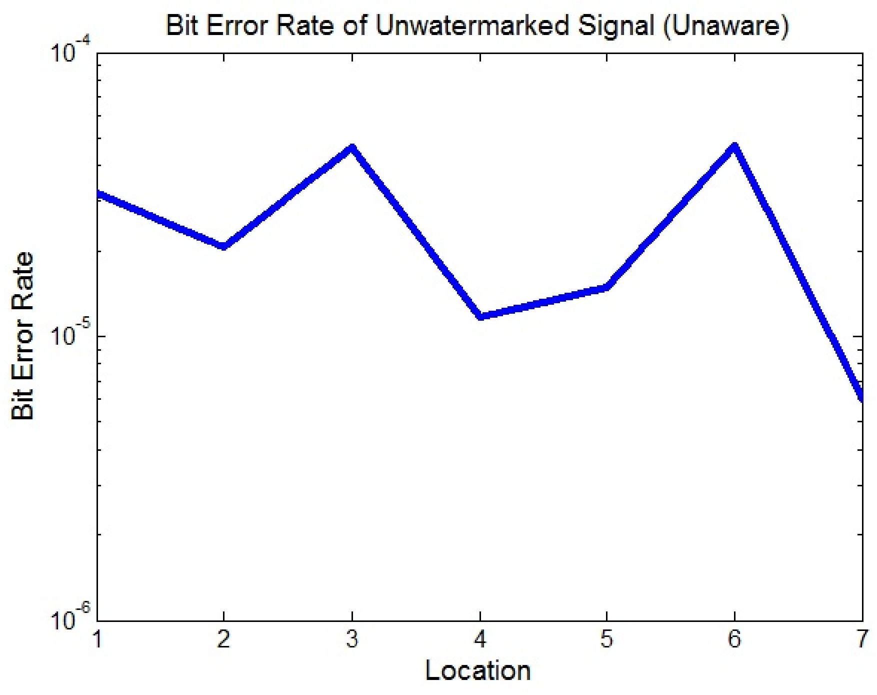


Figure 20 - Low Power, Unwatermarked: Unaware Receiver Message BER

The BER of the unwatermarked signal serves as the control for the tested transmissions. Using the BER calculated from the observed control transmission data for each location, an estimated E_b/N_o can be determined using the theoretical BER for DPSK signals as plotted in Figure 6. The estimated E_b/N_o figure for the 0.21 mW unwatermarked transmission for each location is as follows:

- Location 1: 9.85 dB
- Location 2: 10.05 dB
- Location 3: 9.60 dB
- Location 4: 10.30 dB
- Location 5: 10.15 dB
- Location 6: 9.60 dB
- Location 7: 10.55 dB

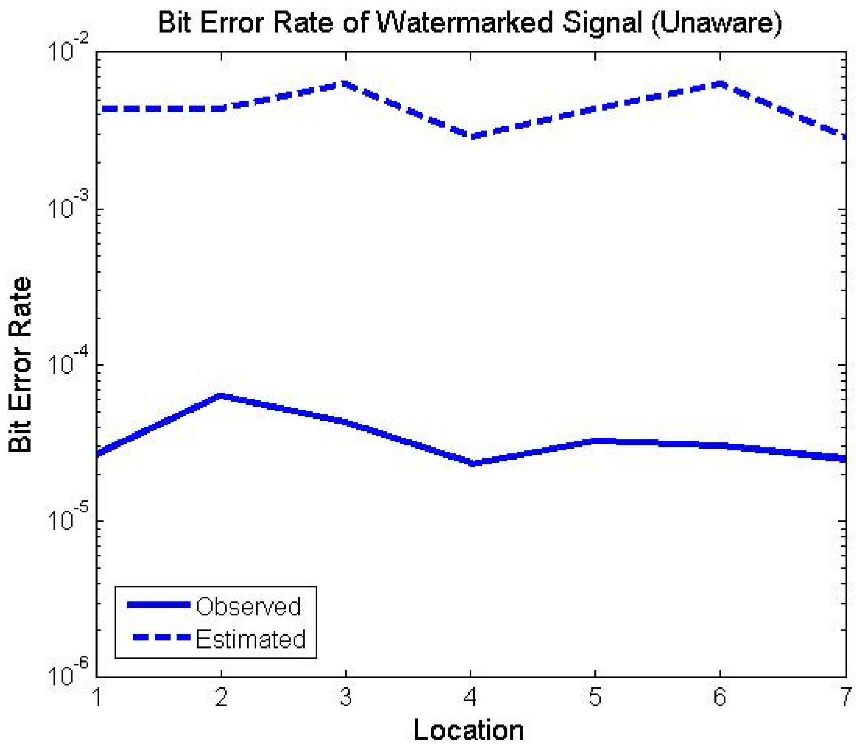


Figure 21 - Low Power, Watermarked: Unaware Receiver Message BER

The BER of the message after watermarking, as decoded by an unaware receiver, is shown in Figure 21. The line marked ‘Observed’ shows the BER of the message as tested and calculated experimentally using the USRP radios and Matlab. The ‘Estimated’ line shows the BER that would be expected based on the Monte Carlo estimation discussed previously, and the Eb/No estimates determined immediately above.

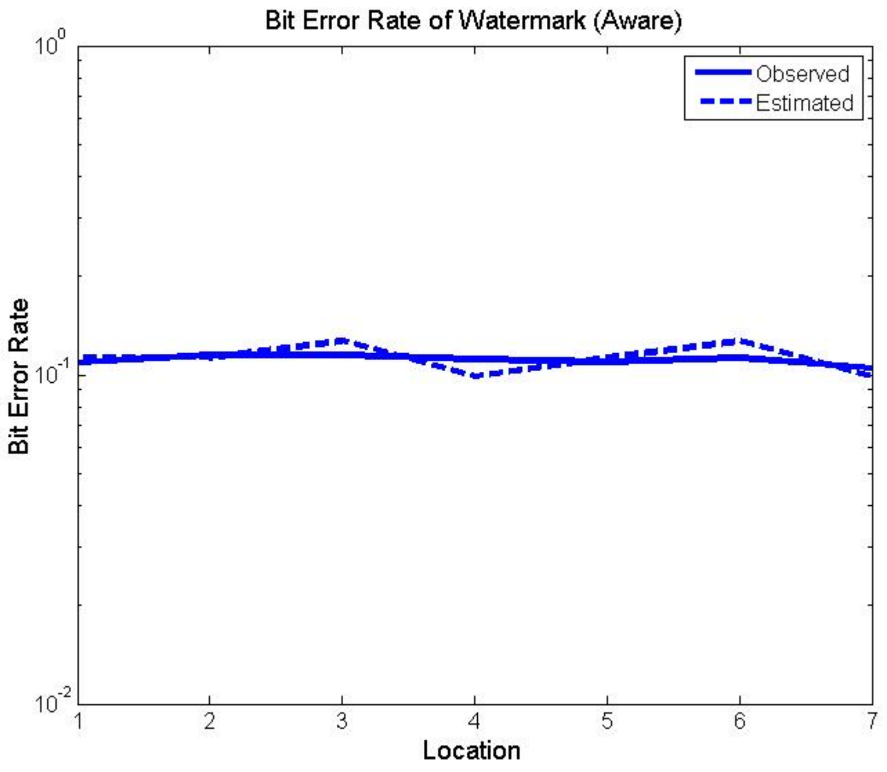


Figure 22 - Low Power, Watermarked: Aware Receiver Watermark BER

An estimated BER for the watermark is made using the listed E_b/N_o levels for each location and the Monte Carlo estimation. Figure 22 shows that the observed watermark

BER is close to the expected BER. In the low power test, the watermark performs very close to the estimate.

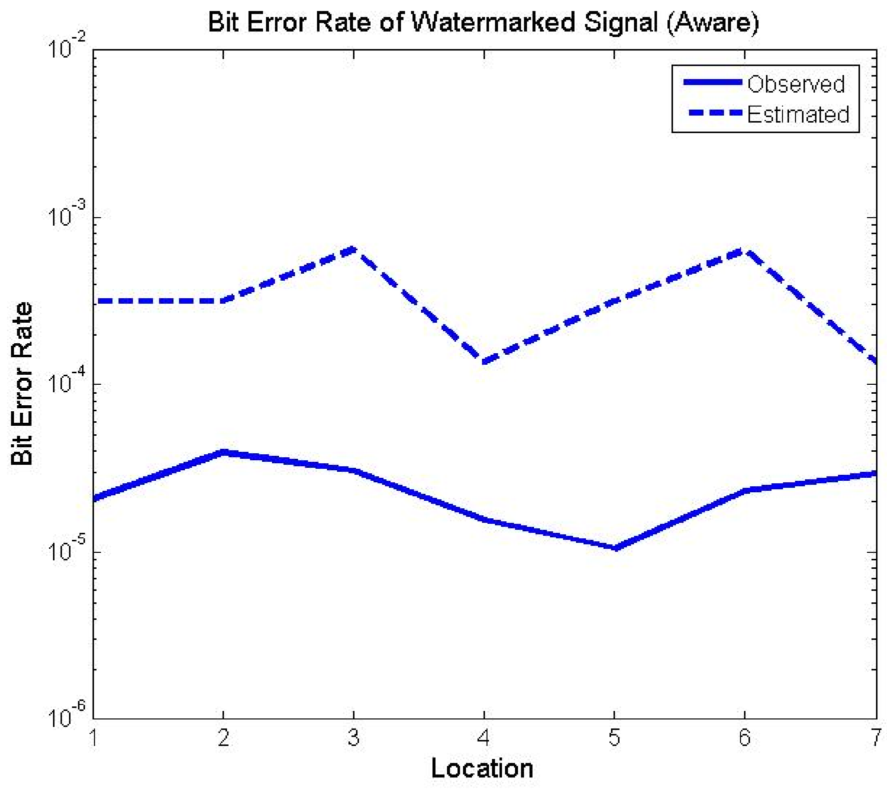


Figure 23 - Low Power, Watermarked: Aware Receiver Message BER

Figure 23 shows the estimated and observed BER of the host signal message as decoded by an aware receiver. Like the unaware receiver, it performs better than expected.

A second set of data was taken at a higher power level of 0.83 mW to further test the ability of the watermarking method. Figure 24 shows the BER of the host signal without an encoded watermark. Figure 25 shows the BER of the watermarked signal demodulated by the unaware receiver. Figure 26 shows the BER of the watermark demodulated by the aware receiver. Figure 27 shows the BER of the watermarked signal demodulated by the aware receiver.

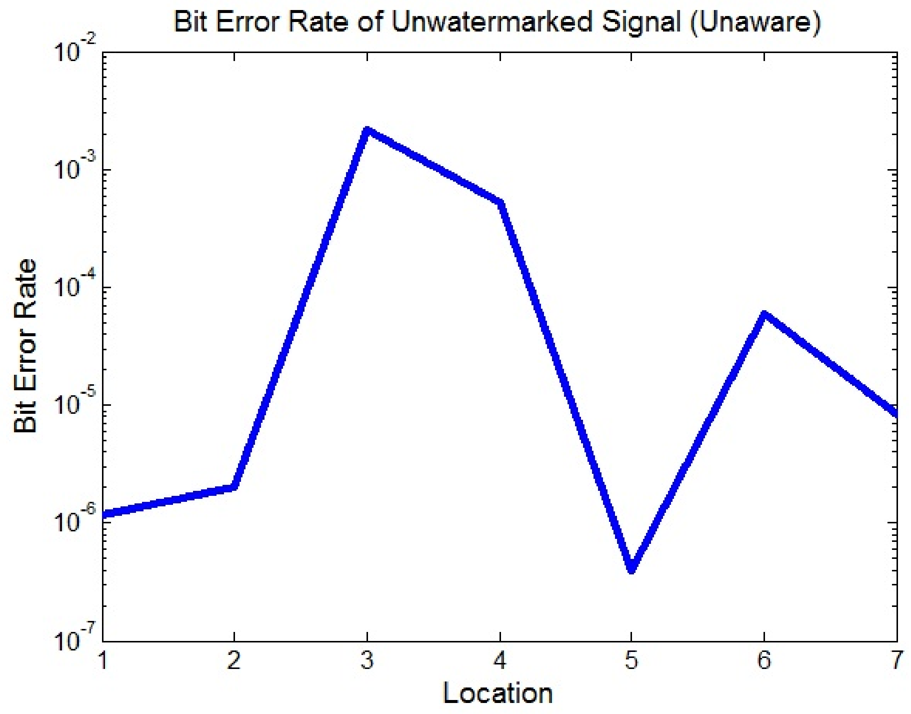


Figure 24 - High Power, Unwatermarked: Unaware Receiver Message BER

It should be noted from Figure 24 that the control signal's BER did not improve as much as would be expected by the 6dB increase in E_b/N_o that would be expected by the transmission power increase. The estimated E_b/N_o figures, based on the observed BER, for the 0.83 mW unwatermarked transmission for each location are as follows:

- Location 1: 11.15 dB
- Location 2: 11.00dB
- Location 3: 7.20 dB
- Location 4: 8.20 dB

- Location 5: 11.45 dB
- Location 6: 9.50 dB
- Location 7: 10.55 dB

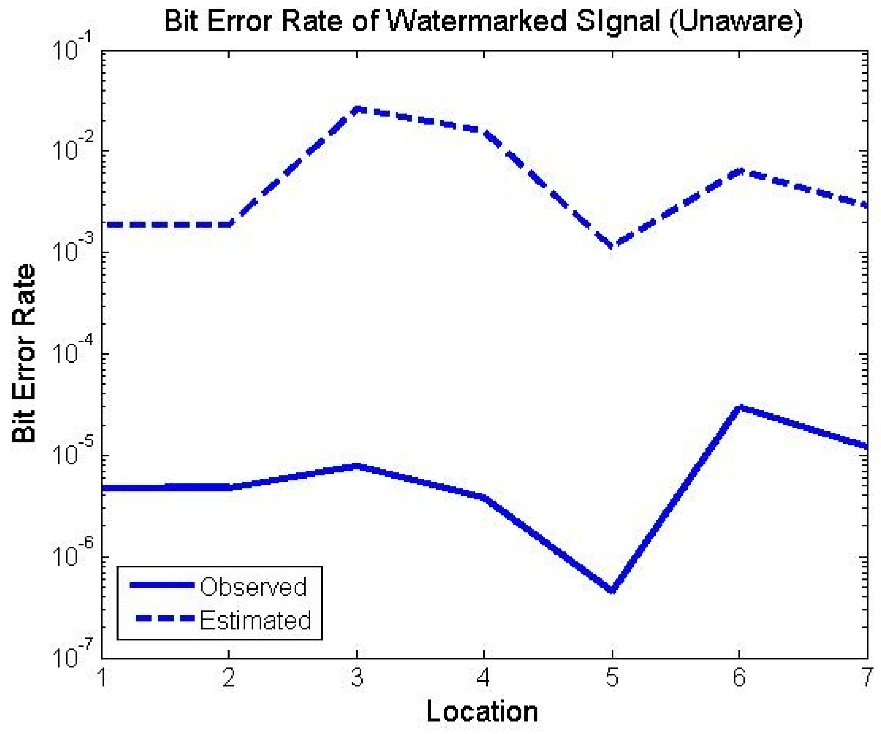


Figure 25 - High Power, Watermarked: Unaware Receiver Message BER

Figure 25 shows that at a higher power level, the watermark continues to only slightly degrade the host signal while the estimation predicts a greater degradation. The BER of the watermarked message is again lower than expected.

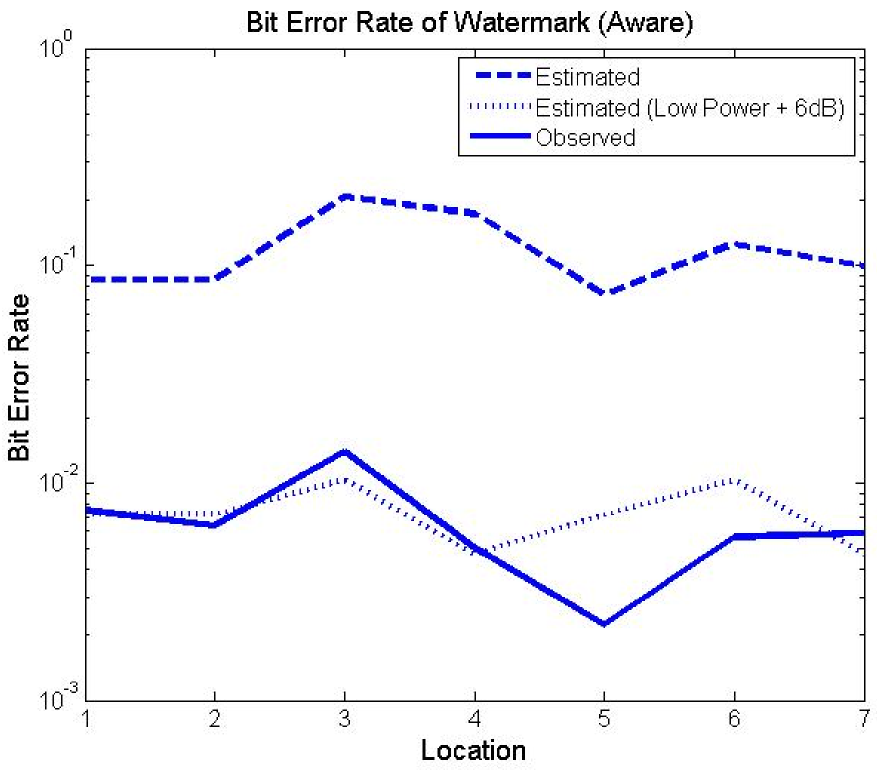


Figure 26 - High Power, Watermarked: Aware Receiver Watermark BER

Figure 26 shows the observed BER of the watermark and two estimated bit error rates using the Monte Carlo method. The first uses the E_b/N_o levels from the high power control/unwatermarked signal to estimate the BER. The second uses the E_b/N_o levels from the low power test and adds 6 dB to account for the 6 dB increase in transmit power. The low power + 6 dB estimate is very close to the observed BER. A probable cause for the discrepancy with respect to the high power Eb/No estimates is that, in the uncontrolled testing environment used, random traffic from WiFi users in the building distorted the estimated high power Eb/No values.

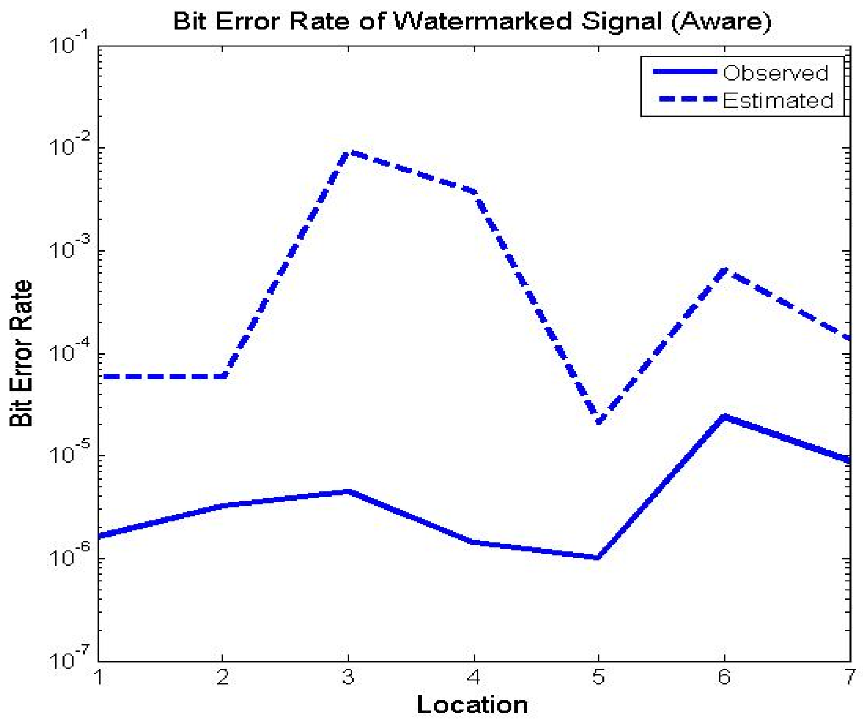


Figure 27 - High Power, Watermarked: Aware Receiver Message BER

Figure 27 shows the observed and estimated BER of the watermarked message as decoded by the aware receiver.

The unexpectedly high BER of the high power control signal, the better than estimated performance of the watermarked message signal, and the requirement to use the low power control signal with the Monte Carlo estimate to produce an accurate BER estimate show that the signal quality, and therefore BER, has deteriorated due to some unknown causes. Because of the relatively low number of samples tested to find the experimental BER for each test, a short burst of errors could easily cause an increased observed BER.

To verify this deterioration and further show the performance of the watermarking method, another set of test data was taken at a single location using several power levels. The results are shown below.

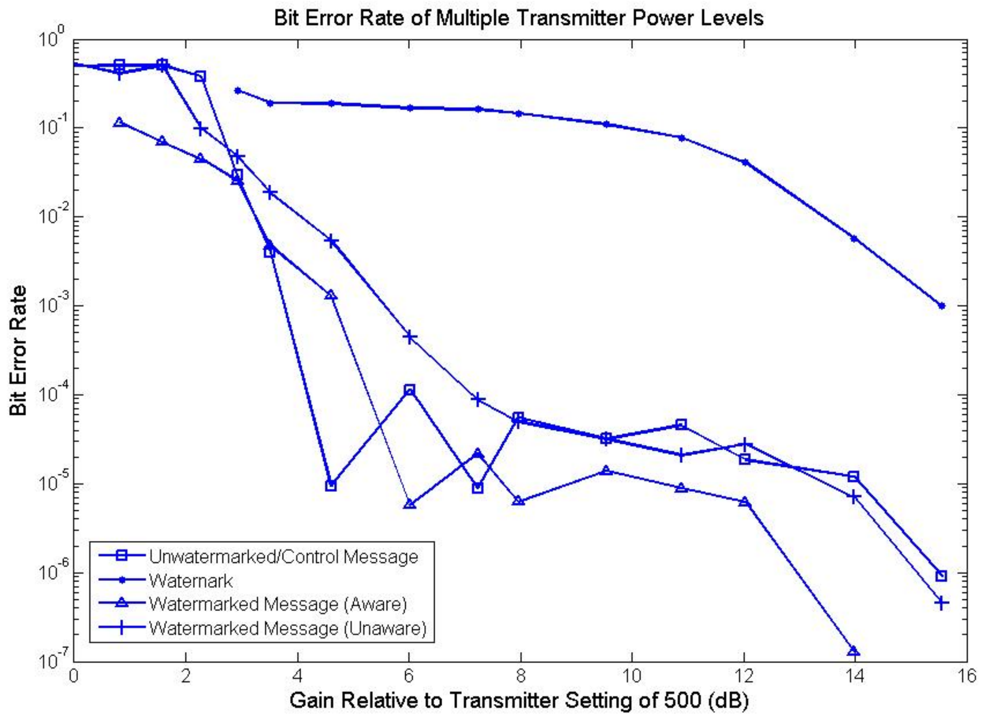


Figure 28 - BER of All Tests Using Varied Transmission Powers at Location 6

Figure 28 shows the performance of the watermarking method as only the transmission power level is changed. The position is unchanged. The performance shows the waterfall pattern expected by the simulations until the BER reaches the 10^{-5} region. At this point, the unknown sources of interference, likely Wi-Fi equipment, appear to be creating a BER floor by causing short bursts of high powered interference. Since only 15,000,000 bits are evaluated, it takes only 150 bits in error to cause a BER of 10^{-5}. Over the one minute time period these samples are collected, it is likely to receive at least one burst of interference capable of increasing the BER.

4.3 Analysis

The results indicate the proposed watermarking method may be useful in future and existing systems. Figures in [20] show that BPSK signals in 802.11b systems may operate at SNR levels such that the BER is in the range of 10^{-6} to 10^{-3}. The proposed watermarking method was tested and shows promise where the BER for the DBPSK signal is in the 10^{-6} to 10^{-4} range or lower. To analyze the watermarking method's performance, the host signal and watermark signal are examined. The proposed method is also compared to a similar method that has been previously proposed.

4.3.1 Host Signal

The unaware receiver will receive the original message with some degradation as expected. It is difficult to determine a precise degree by which the watermark encoding degrades the signal in real world tests. However from the data gathered in analyzing the BER of each received signal an approximation can be made. To approximate the signal degradation, the BER for watermarked and host signals were compared – each received by an unaware, DBPSK receiver. The receiver BER of the watermarked signal and non-watermarked are compared using Figures 21 and 25.

In comparing the BER of the non-watermarked host signals to the watermarked message signals, the watermarked signal appears to have suffered anywhere from no change to 4 dB in loss. The real world tests showed loss comparable to the -4 dB estimate shown by the Monte Carlo simulations. The effective loss may be different depending on the watermark implementation by the user. The loss in signal quality would result in some reduction in transmission range.

4.3.2 Watermark

At low power, the BER of the received watermark is, as expected, much higher than the BER of the host signal. For the 21 mW transmission, about 10% of watermark bits would be received incorrectly in the tested environment. This would be an unacceptable level for almost any use. Reducing the effective bit rate of the watermark by repeating watermarked bits would effectively increase the bit time and therefore E_b/N_o, improving the watermark BER.

Increasing the transmission power of the signal by a factor or four decreases the BER of the D8PSK watermark as expected. The power increase is equivalent to an approximate 6dB increase in E_b/N_o. The D8PSK watermark BER improvement from around $2x10^{-1}$ to around $2x10^{-2}$ with the increase in power reflects the expected 6 dB improvement in E_b/N_o. The Monte Carlo simulation accurately predicts the BER of the received watermark given a correct host signal E_b/N_o level.

Based on the test results, the effectiveness of the watermark will depend on the scheme developed by the user. For example a block based watermark where a 256 bit watermark was generated for each 256 bits of raw data may require a 90% watermark match. With a BER of 0.10, few legitimate blocks of data would be authenticated, and the majority would not. This could lead to a great deal of lost data in a one-way communication channel. However reducing the number of watermark bits required per host signal data bit could significantly decrease the watermark BER.

4.3.3 Method Comparison

There is one method proposed in [7] that use a similar watermarking method as the one proposed by this thesis. The proposed method uses a different form of constellation dithering to embed a separate watermark data stream within an OFDM carrier. Additionally the results are measured by comparing BER to SNR level, which is comparable to the use of BER to analyze the method proposed in this paper. Figure 29 shows the effect of the constellation dithering watermark method on the BER of the host signal at different watermark strengths. Alpha is the power level at which the watermark is implanted, where *alpha + payload power = 1*.

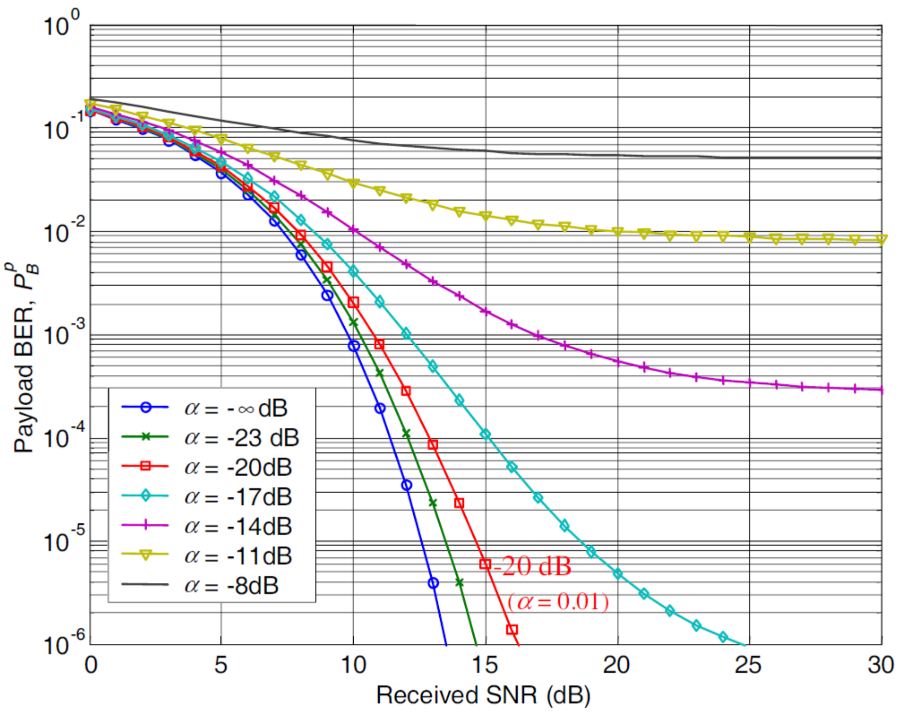


Figure 29 - OFDM Watermarking Signal Quality [7]

From the figure it is observed that at a power level for which the original host signal has a BER of 10^{-6}, a host signal watermarked with a signal 23 dB weaker, decoded by an unaware receiver will have a BER of at least 10^{-5}. The lowest simulated power of this watermarking method creates about a 1dB equivalent degradation to the SNR of the signal in the range a BPSK signal would be used.

The simulations in [7] used a signal bandwidth of 1 MHz using a baseband sample rate of 1.27 Msps. The main signal would be able to transmit a BPSK signal with a 614 kbps bit rate or a QPSK signal with a 1.27 Mbps bit rate. The simulations used a watermark bitrate ranging from 1.99 to 12.5 kbps with a QPSK host signal, so it was tested with a watermark bit rate to host signal bit rate ratio of 1:638 to 1:101. In comparison, the method proposed by this thesis was tested with a signal bandwidth of 1 MHz, main signal bit rate of 250 kbps, and a watermark bit rate to host signal bit rate ratio of 1:1; the proposed physical layer watermarking method was tested with equal host signal and watermark bit rates. The user of the proposed watermarking system would be able lower the effective watermark bit rate to decrease the watermark BER to much lower levels than achieved in the tests.

As tested, the host signal suffers significant degradation that might be unacceptable in some systems. To show the effect of lowering the watermark power, more plots were created using the Monte Carlo simulation of the proposed watermarking system.

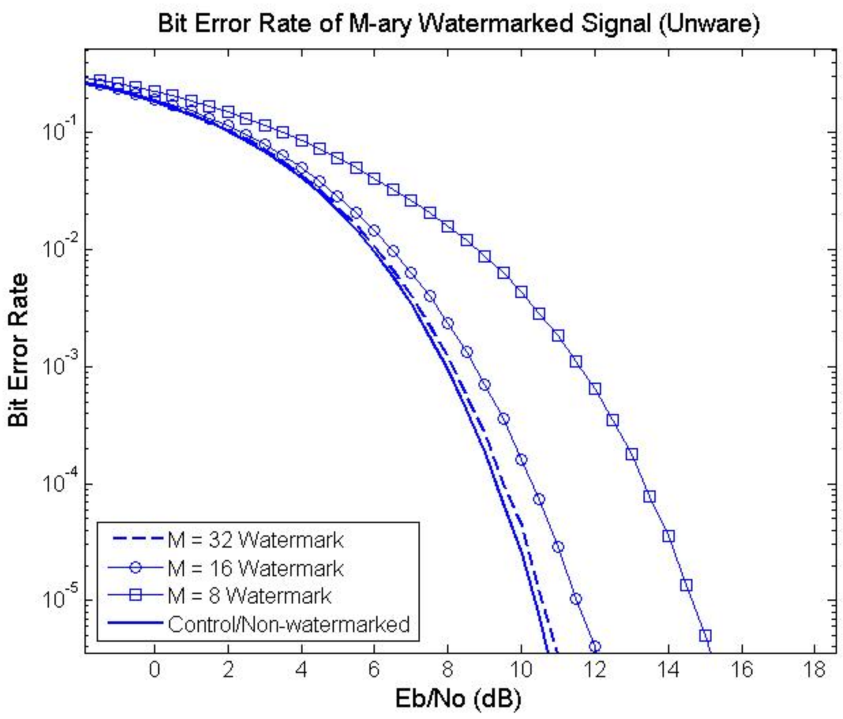


Figure 30 - Effect on Signal Degradation with Decreased Watermark Power

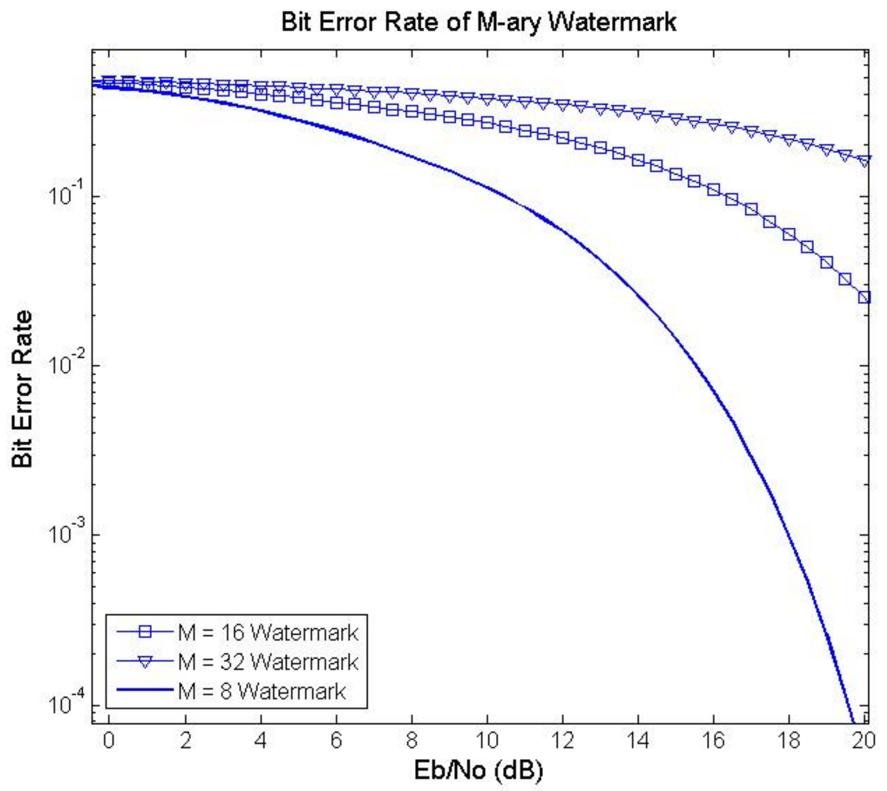


Figure 31 - Effect on Watermark BER with Decreased Watermark Power

Figure 30 shows the effect on the host signal of reducing the watermark power. Using a D16PSK modulator would reduce the signal degradation to about -1 dB. A D32PSK modulator implementation would further reduce the signal degradation: to about -0.25 dB. Figure 31 shows the effect on the watermark signal quality when the watermark power is reduced. The watermark BER will be increased significantly. Recall that the watermark bit rate may be decreased to improve the watermark BER.

4.4 Viability

The physical layer watermarking method proposed in this thesis offers comparable performance to a previously developed method as well as the potential to be implemented in several different types of systems. The method is also configurable with ability to balance watermark stealth with robustness.

In physical layer watermarking stealth is the measure of how little the host signal is affected, to the unaware receiver, by watermarking. Stealth of the proposed watermarking method, like in other methods, can be increased by decreasing the power of the watermark. This will decrease the range at which the watermark can be received. Any attacker aware of the watermarking method within watermark reception range would be able to decode the raw watermark bit stream. The user of the watermarking method would be able to create a secure watermark bit stream based on their needs. The security of the transmitter authentication would depend on the user's choice of watermark generation method. A tag generated based on a secret key as well as time such as in [8] would be a likely choice.

The robustness of the proposed watermarking method also depends on the watermark strength and effective watermark bit rate. At the tested strength and bit rate, the watermark would be robust enough in some situations. However, most systems would require a significantly lower BER which may be achieved by lowering the effective watermark bit rate through bit repetition or by incorporating Forward Error Correction parity bits. In this method robustness can be maintained by lowering the effective watermark bitrate as the watermark power or received signal power is reduced.

4.4.1 Configurability

The following list shows the configurations available for the proposed watermarking method and the descriptions for their use follow.

- Watermark power adjustment (phase offset level)
- Watermark bit rate (bit repetition)
- Watermark bit rate (unused bit times)

The stealth of the watermark can be increased by decreasing the phase offset that indicates a watermark logic 1. For example, a D16PSK modulator could be used instead of the tested D8PSK modulator. This would have the effect of reducing the signal degradation, or increasing stealth, seen by the unaware receiver while increasing the BER of the watermark decoded by the aware receiver.

The watermark BER can be reduced without decreasing stealth by lowering the watermark bit rate through bit repetition. Watermark bits could be repeated several times, effectively increasing the bit time while reducing the bit rate. Ideally the watermark bits would be repeated in a known, interleaved pattern to prevent bursts of interference from causing excessive bit errors. A reduced watermark BER is possible if fewer watermark bits are required for each raw data block. For example, a watermarking system with a reduced watermark bit rate might apply a 256 bit watermark to every 8192 bit block of data. By increasing the watermark bit time the effective E_b/N_o of each watermark bit could then be increased by nearly 32 or 15dB, greatly decreasing the watermark BER. The user could then require 99% of watermark bits to match for authentication, increasing security.

The stealth of the watermark can also be increased by lowering the watermark bit rate. To do so, the bit rate would be lowered by padding the watermark bit stream with logic 0's. Since only watermark logic 1's alter the host signal, padding the watermark bit stream with logic 0's will reduced the signal degradation seen by the unaware receiver. For example the watermark bit rate could be reduced by half by inserting a 0 watermark bit after every valid watermark bit. This would reduce the number of altered host signal symbols by half.

4.4.2 Implementation

A goal of the proposed watermarking method is to provide an easily implementable method of authenticating wireless data on the physical layer. The proposed method could be added to some existing systems while leaving unaware transmitters and receivers unchanged and possibly only requiring software changes to aware transmitters and receivers. As tested, the watermark would likely only be usable by aware receivers very close to the receiver or where transmissions were at very high power levels. However the configurability of the proposed watermarking systems ensures the system is viable in a wider range of systems. A watermark used purely for authentication would not necessarily require a watermark bit rate equal to the payload bit rate. The proposed watermarking method creates a system in which the end user of the method can choose the effective bit rate as required. The watermark bitrate can be as high as the payload bitrate, as tested, or lowered to reduce watermark BER. The digital RF free space link equation, which can be found in literature and [21], is as follows:

$$(Eb/NO) = Pt + Gt + (mi = 83.41 \; or \; km = 87.56) - 20log(d) - 20log(f) + Gr - R + 228.6 - T0Sys - Lo - M \qquad (5)$$

where all units are dB and R is the bitrate.

In an authentication only watermarking system, the watermark bit rate to payload bit rate ratio might be reduced to 1:100 as seen in [7] while maintaining the same 4 dB host signal degradation. From the RF link equation, this could improve the effective E_b/N_o of the watermark by up to 20 dB. This would cause a significant improvement in the watermark BER. The watermark power could then be lowered to match the host signal degradation seen in [7]. Comparing the proposed watermarking method with that in [7] shows the method has the potential compete with the performance of current physical layer watermarking methods. And unlike other methods, the proposed method can be implemented using common, pre-existing hardware.

The proposed watermarking method might also allow a user to imbed a separate data channel within a BPSK or QPSK signal giving a secondary user access to data. This may allow a greater ability to configure a wireless network for spectral efficiency. This is not to say channel capacity could be increased, but that transmission may be altered to add additional bit rate thresholds without switching sub-carrier modulation schemes. Figure 32 shows the diminishing benefit of increased received SNR for IEEE 802.11b modulation schemes.

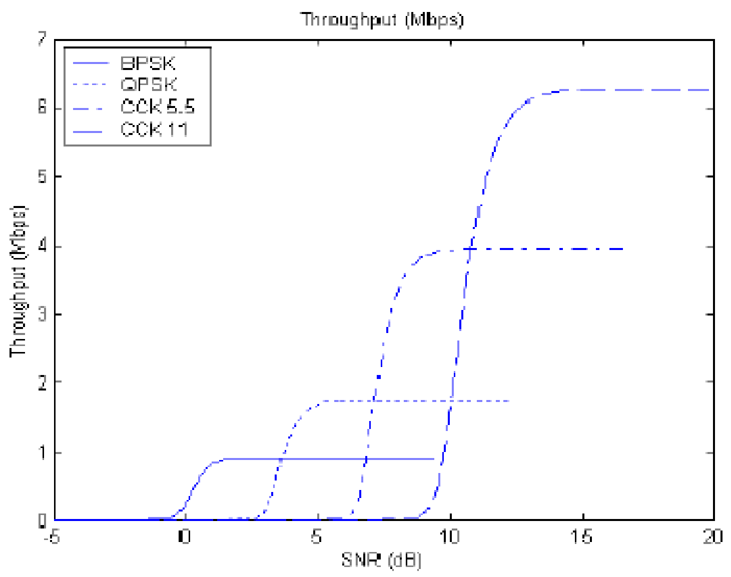


Figure 32 - IEEE 802.11b Throughput vs. SNR [22]

There is a significant SNR increase for each modulation scheme that can occur where little increased throughput occurs but switching to a higher order modulation scheme would decrease throughput. For example, the range 2.0 – 3.5 dB for BPSK shows little increase in throughput, but these SNR's cannot support a switch to QPSK. Potentially a watermark signal added to the BPSK signal could allow additional, perhaps surreptitious, communication without significantly decreasing network throughput of the over endowed BPSK signal.

One of the main benefits of the proposed watermarking method is the ability to implement it using readily available technology. The method, as tested, can be fully implemented by encoding a bit stream, transmitting with a D8PSK modulator, and receiving with DBPSK (unaware) and D8PSK (aware) demodulators. The watermarked bit streams can be encoded and decoded with fairly simple data processing. Lower strength watermarks can be applied by using 16- or higher DPSK modulators and demodulators.

The proposed watermarking method can be extended to DQPSK signals as well as MPSK and QAM signals. In each, the method would require the watermark be encoded and transmitted with a modulator of higher order. For PSK signals the minimum watermark order would be 4*M where M is the order of the host signal. For example a QPSK signal would need to be watermarked with at least a 16-PSK modulator. A QAM-4 signal could be watermarked with a QAM-16 modulator, but higher QAM orders may require different watermark pairings.

The next chapter concludes this thesis and provides suggestions for future work on the proposed physical layer watermarking method.

CHAPTER V

CONCLUSION

5.1 Conclusion

A physical layer watermarking method using standard GNU Radio blocks has been proposed and tested. The results show that it has the potential to compete with other physical layer watermarking methods with respect to potential security and host signal degradation. The watermarking method is very configurable, allowing the user to balance watermark stealth and robustness to suit their needs. Additionally, this physical layer watermarking method can be implemented using readily available technology. Some systems would be able to implement the watermarking method without any hardware modification.

The nature of the watermarking method also allows it to be used as an additional data channel, making it suitable for stealthy transmissions of secret data. Using this watermarking method allows a user to add a secret data transmission onto a pre-existing transmission. The method might also be used to increase spectral efficiency in wireless networks when bit rate thresholds are reducing efficiency.

The proposed watermarking method was tested in a real world environment and performed with similar results to simulations.

5.2 Future Work

Implementation and testing should be performed for higher order DPSK modulators as well as MPSK and QAM modulators. The existing GNU Radio blocks did not allow for this. QAM blocks are being developed, so in the future, testing with GNU Radio blocks should be possible. Alternatively, it may be possible to use Matlab to generate samples of a PSK or QAM modulated signal, transmit and receive the samples using the USRP radios, and demodulate the received signal with Matlab.

The effects of changing the effective watermark bit rate should be analyzed to determine the most efficient bit rates for given levels of signal degradation. The watermark bit time can be increased which may greatly reduce the watermark BER. By repeating each watermark bit several times in a known interleaved pattern, the bit time can be increased while reducing the watermark's vulnerability to bursts of interference. Another method of reducing the watermark bit rate would involve inserting 0 bits into known locations in the watermark bit stream. This would reduce the number of altered host signal bits, lowering the effective power of the watermark without changing the modulation scheme required to produce the watermarked signal. The two methods of bit rate reduction should be tested in use together and separately. FEC could also be used in addition to lowered bit rates to decrease the watermark BER further.

The proposed method could also be modified to allowing bitrates above the symbol rate to be transmitted in the watermark channel when conditions provide a high received SNR. Using a 32-PSK watermark on a BPSK host signal would allow a watermark bitrate of twice the symbol rate by allowing a watermark constellation change of 11.25, 22.5, or

33.75 degrees. The host signal degradation would be roughly the same as using a 16-PSK watermark to implant a watermark at the symbol rate.

Bibliography

[1] U.S. Government Printing Office, "Federal Communications Commission: Unlicensed Operation in the TV Broadcast Bands; Final Rule," *Federal Register*, vol. 75, no. 233, pp. 75813-75843, December 2010.

[2] Federal Communications Commission, FCC 10-174, 2010, SECOND MEMORANDUM OPINION AND ORDER: Unlicensed Operation in the TV Broadcast Bands.

[3] Carlos Cordeiro, Kiran Challapali, and Dagnachew Birru, "IEEE 802.22: An Introduction to the First Wireless Standard Based on Cognitive Radios," *Journal of Communications*, vol. 1, no. 1, pp. 38-47, April 2006.

[4] Richa Bansal, Siddharth Tiwari, and Divya Bansal, "Non-cryptographic methods of MAC spoof detection in wireless LAN," in *International Conference on Networks*, New Delhi, 2008, pp. 1-6.

[5] R. Chandramouli, Nasir Memon, and Majid Rabbani, "Digital Watermarking," *Encyclopedia of Imaging Science and Technology.*\, 2002.

[6] Xianbin Wang, Yiyan Wu, and Bernard Caron, "Transmitter Identification Using Embedded Pseudo Random Sequences," *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 244-252, September 2004.

[7] John E. Kleider, Steve Gifford, Scott Chuprun, and Bruce Fette, "Radio Frequency Watermarking for OFDM Wireless Networks," in *IEEE Internation Conference on*

Acoustics, Speech, and Signal Processing, Montreal, 2004, pp. 397-400.

[8] Paul L. Yu, John S. Baras, and Brian M. Sadler, "Physical-Layer Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 1, pp. 38-51, March 2008.

[9] Paul L. Yu, John S. Baras, and Brian M. Sadler, "Multicarrier Authentication at the Physical Layer," in *2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Newport Beach, 2008, pp. 1-6.

[10] Paul L. Yu, John S. Baras, and Brian M. Sadler, "Power Allocation Tradeoffs in Multicarrier Authentication Systems," in *2009 Sarnoff Symposium*, Princeton, 2009, pp. 1-5.

[11] Nate Goergen, T. Clancy, and Timothy R. Newmann, "Physical Layer Authentication Watermarks Through Synthetic Channel Emulation," in *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum*, Singapore, 2010, pp. 1-7.

[12] Ettus Research LLC. (2011, May) Ettus Research. [Online]. http://www.ettus.com/

[13] Eric Blossom. (2010, June) GNU Radio. [Online]. http://gnuradio.org/redmine/wiki/gnuradio

[14] Pan Liu and Yeheskel Bar-Ness, "Closed-Form Expressions for BER Performance in OFDM Systems with Phase Noise," in *IEEE International Conference on Communications*, Istanbul, 2006, pp. 5366-5370.

[15] The Mathworks Inc. (2011, May) Mathworks. [Online]. http://www.mathworks.com/help/toolbox/comm/ug/bsvzixi.html#bq423zy

[16] Bernard Sklar, *Digital Communications: Fundamentals and Applications*, 2nd ed., Rose Keman, Ed. Upper Saddle River, United States: Prentice Hall, 2000.

[17] Unnikrishna Pillai and Papoulis Athanasios, *Probability Random Variables, and Stochastic Processes*, 4th ed. United States of America: McGraw-Hill, 2001.

[18] R. F. Pawula, S. O. Rice, and J. H. Roberts, "Distribution of the Phase Angle Between Two Vectors Perturbed by Gaussian Noise," *IEEE Transactions on Communications*, vol. 30, no. 8, pp. 1828-1841, August 1982.

[19] Kaleem Ahmad, Uwe Meier, Andreas Pape, Halina Kwasnicka, and Bjoern Griese, "A Generic Cognitive Radio for Evaluating Coexistence Optimized Industrial Automation Systems," in *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, Rome, 2009, pp. 1-3.

[20] J. Mikulka and S. Hanus, "CCK and Barker Coding Implementation in IEEE 802.11b Standard," in *17th International Conference Radioelektronika*, Brno, 2007.

[21] George Scheets, Modern Communications, 2010, Lecture notes and course materials.

[22] Javier Pavon and Sunghyun Choi, "Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strenth Measurement," in *IEEE International*

Conference on Communications, Anchorage, 2003, pp. 1108-1113.

[23] Jim Pearce. (2009) Spread Spectrum Scene. [Online]. http://www.sss-mag.com/ebn0.html

APPENDICES

The following appendices contain the binary information transmitted, and the Matlab code used analyze the received data.

Appendix A

```
function [ mixed ] = mix_wm_sig(tx_fname_sig, tx_fname_wm, sig_format,
wm_format)
% Bruce Lebold, 01/04/2011
% Calculate the PBE of rx signal using a repeated tx binary file and
the
% received binary file.
%
% For tx_ and rx_format variables: 0 sets each 1 byte
% to reflect 1 bit of data, i.e. each byte can be either 0 or 1. 1 sets
% each byte to reflect 8 bits of data.
%
% The variable start_bits sets the number of bits used to find the
start of
% good data in the rx file. The first start_bits bits of the tx file
will
% be aligned in the rx file.
%
% Set wm to 0 if received signal data and 1 if wm data was received in
3
% bytes per bit format.


%% Store files to arrays
fclose('all');

fid_tx = fopen(tx_fname_sig,'r');
tx = fread(fid_tx);
fclose('all');

if sig_format == 1
    tx_temp = zeros(8*length(tx),1);
    for i=1:1:length(tx)
        tx_temp((i-1)*8+1) = boolean(bitand(tx(i),128));
        tx_temp((i-1)*8+2) = boolean(bitand(tx(i),64));
        tx_temp((i-1)*8+3) = boolean(bitand(tx(i),32));
        tx_temp((i-1)*8+4) = boolean(bitand(tx(i),16));
        tx_temp((i-1)*8+5) = boolean(bitand(tx(i),8));
        tx_temp((i-1)*8+6) = boolean(bitand(tx(i),4));
        tx_temp((i-1)*8+7) = boolean(bitand(tx(i),2));
        tx_temp((i-1)*8+8) = boolean(bitand(tx(i),1));
    end

    tx = tx_temp;
end

sig = tx;

fid_tx = fopen(tx_fname_wm,'r');
tx = fread(fid_tx);
fclose('all');

if wm_format == 1
    tx_temp = zeros(8*length(tx),1);
    for i=1:1:length(tx)
        tx_temp((i-1)*8+1) = boolean(bitand(tx(i),128));
        tx_temp((i-1)*8+2) = boolean(bitand(tx(i),64));
        tx_temp((i-1)*8+3) = boolean(bitand(tx(i),32));
        tx_temp((i-1)*8+4) = boolean(bitand(tx(i),16));
        tx_temp((i-1)*8+5) = boolean(bitand(tx(i),8));
        tx_temp((i-1)*8+6) = boolean(bitand(tx(i),4));
        tx_temp((i-1)*8+7) = boolean(bitand(tx(i),2));
        tx_temp((i-1)*8+8) = boolean(bitand(tx(i),1));
    end

    tx = tx_temp;
end

wm = tx;

%% Assign 8-PSK symbols

% form possible 8-PSK bit combinations
poss = [0 0 0; 0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 1 1; 1 0 1; 1 0 0];
last_sym = 1;
mixed = zeros(3*length(sig),1);

% Assign 8-PSK bit pattern based on host and watermark bit
for i = 1:1:length(sig)
    r = 1;

    wmindex = mod(i-1, 256) + 1;

    if [sig(i) wm(wmindex)] == [0 0]
        last_sym = 1;
        mixed((i-1)*3+1:i*3) = poss(last_sym,:);

    elseif [sig(i) wm(wmindex)] == [1 0]
        last_sym = 5;
        mixed((i-1)*3+1:i*3) = poss(last_sym,:);
    elseif [sig(i) wm(wmindex)] == [1 1]
        last_sym = 5 + r;
        mixed((i-1)*3+1:i*3) = poss(last_sym,:);

    elseif [sig(i) wm(wmindex)] == [0 1]
        last_sym = mod(r,8)+1;
        mixed((i-1)*3+1:i*3) = poss(last_sym,:);

    end

end

% Store to array
mixed_temp = zeros(length(mixed)/8,1);
for i=1:1:length(mixed_temp)
    mixed_temp(i) = 128*mixed((i-1)*8+1) + 64*mixed((i-1)*8+2) +
32*mixed((i-1)*8+3) + 16*mixed((i-1)*8+4) + 8*mixed((i-1)*8+5) +
4*mixed((i-1)*8+6) + 2*mixed((i-1)*8+7) + mixed((i-1)*8+8);

end

mixed = mixed_temp;
```

Appendix B

```
function [ local index count pbe pbe_s] = wm_pbe_local(tx_fname,
rx_fname, tx_format, rx_format, start_bits, wm)
% Bruce Lebold, 01/04/2011
% Calculate the PBE of rx signal using a repeated tx binary file and
the
% received binary file.
%
% For tx_ and rx_format variables: 0 sets each 1 byte
% to reflect 1 bit of data, i.e. each byte can be either 0 or 1. 1 sets
% each byte to reflect 8 bits of data.
%
% The variable start_bits sets the number of bits used to find the
start of
% good data in the rx file. The first start_bits bits of the tx file
will
% be aligned in the rx file.
%
% Set wm to 0 if received signal data and 1 if wm data was received in
3
% bytes per bit format.





%% Store files to arrays

fclose('all');

pbe_s = 1;

fid_tx = fopen(tx_fname, 'r');
fid_rx = fopen(rx_fname, 'r');

tx = fread(fid_tx);
rx = fread(fid_rx);
rx=rx(1:length(rx)-mod(length(rx),3));
fclose('all');

if tx_format == 1
    tx_temp = zeros(8*length(tx),1);
    for i=1:1:length(tx)
        tx_temp((i-1)*8+1) = boolean(bitand(tx(i),128));
        tx_temp((i-1)*8+2) = boolean(bitand(tx(i),64));
        tx_temp((i-1)*8+3) = boolean(bitand(tx(i),32));
        tx_temp((i-1)*8+4) = boolean(bitand(tx(i),16));
        tx_temp((i-1)*8+5) = boolean(bitand(tx(i),8));
        tx_temp((i-1)*8+6) = boolean(bitand(tx(i),4));
        tx_temp((i-1)*8+7) = boolean(bitand(tx(i),2));
        tx_temp((i-1)*8+8) = boolean(bitand(tx(i),1));
    end

    tx = tx_temp;
```

```
end

if rx_format == 1
    rx_temp = zeros(8*length(rx),1);
    for i=1:1:length(rx)
        rx_temp((i-1)*8+1) = boolean(bitand(rx(i),128));
        rx_temp((i-1)*8+2) = boolean(bitand(rx(i),64));
        rx_temp((i-1)*8+3) = boolean(bitand(rx(i),32));
        rx_temp((i-1)*8+4) = boolean(bitand(rx(i),16));
        rx_temp((i-1)*8+5) = boolean(bitand(rx(i),8));
        rx_temp((i-1)*8+6) = boolean(bitand(rx(i),4));
        rx_temp((i-1)*8+7) = boolean(bitand(rx(i),2));
        rx_temp((i-1)*8+8) = boolean(bitand(rx(i),1));
    end

    rx = rx_temp;
end

%% If a watermark is being decoded, decide what the watermark bit is

if wm == 1

    rx_3 = zeros(length(rx)/3, 1);
    poss = [0 0 0; 0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 1 1; 1 0 1; 1 0 0];


    for i=1:1:length(rx_3)

        if rx((i-1)*3+1:i*3) == poss(1,:)'
            rx_3(i) = 1;

        elseif rx((i-1)*3+1:i*3) == poss(2,:)'
            rx_3(i) = 2;

        elseif rx((i-1)*3+1:i*3) == poss(3,:)'
            rx_3(i) = 3;

        elseif rx((i-1)*3+1:i*3) == poss(4,:)'
            rx_3(i) = 4;

        elseif rx((i-1)*3+1:i*3) == poss(5,:)'
            rx_3(i) = 5;

        elseif rx((i-1)*3+1:i*3) == poss(6,:)'
            rx_3(i) = 6;

        elseif rx((i-1)*3+1:i*3) == poss(7,:)'
            rx_3(i) = 7;

        elseif rx((i-1)*3+1:i*3) == poss(8,:)'
            rx_3(i) = 8;

        end
    end

    rx = zeros(length(rx_3),1);

    for i=1:1:length(rx_3)
        if rx_3(i) == 1 || rx_3(i) == 5 || rx_3(i) == 4 || rx_3(i) == 8
            rx(i) = 0;
        else rx(i) = 1;
        end
    end

end

%% Create array of local pbes and find gaps
local = zeros(length(rx),1);
count = 1;

for i=1:length(rx)-length(tx)
    local(i) = sum(tx ~= rx(i:i+length(tx)-1))/length(tx);
end

index = [];

for i = 1:length(local)-length(tx)
    if local(i) < .25
        index(count) = i;
        count = count + 1;
    end
end

gapind = [];
count = 1;
for i=2:length(index)
    if index(i)-index(i-1) ~= length(tx)
        gapind(count) = i;
        count = count + 1;
    end
end

gapind % print the gap locations
length(rx)/length(tx) % print the number of blocks
length(index) % how many blocks there should have been

accum = 0;
count = 0;
rx(1:index(1)-1) = [];
accum = index(1)-1;
count = accum;

%% Pad the gaps
for i=2:length(index)

    if mod(index(i)-index(i-1), length(tx)) == 1
```

```
        rx(index(i) - accum - 1) = [];
        accum = accum + 1;
        count = count + 1;

    elseif mod(index(i)-index(i-1), length(tx)) ~= 0

        add = length(tx) - mod(index(i)-index(i-1), length(tx));
        temp = [rx(1:index(i) - accum - 1); zeros(add, 1); rx(index(i)-
accum:length(rx))];
        rx = temp;
        accum = accum - add;
        count = count + add;
    end

end

%% Calculate pbe
errors = 0;

for i=1:length(rx)
   if rx(i) ~= tx(mod(i-1, length(tx)) + 1)
       errors = errors+1;
   end
end

pbe = errors/length(rx);
```

Appendix C

```
% Find the BER of the original message using an aware receiver. Feed
the
% original message bit stream to tx_fname and the received D8PSK bit
stream
% to rx_fname.

function [ local index count pbe rx_3] = wm_pbe_aware(tx_fname,
rx_fname)

%% Store files to arrays

fclose('all');

fid_tx = fopen(tx_fname, 'r');
fid_rx = fopen(rx_fname, 'r');

tx = fread(fid_tx);
rx = fread(fid_rx);
rx=rx(1:length(rx)-mod(length(rx),3));
fclose('all');

tx_temp = zeros(8*length(tx),1);
for i=1:1:length(tx)
    tx_temp((i-1)*8+1) = boolean(bitand(tx(i),128));
    tx_temp((i-1)*8+2) = boolean(bitand(tx(i),64));
    tx_temp((i-1)*8+3) = boolean(bitand(tx(i),32));
    tx_temp((i-1)*8+4) = boolean(bitand(tx(i),16));
    tx_temp((i-1)*8+5) = boolean(bitand(tx(i),8));
    tx_temp((i-1)*8+6) = boolean(bitand(tx(i),4));
    tx_temp((i-1)*8+7) = boolean(bitand(tx(i),2));
    tx_temp((i-1)*8+8) = boolean(bitand(tx(i),1));
end

tx = tx_temp;

rx_3 = zeros(length(rx)/3, 1);

    % Determine symbol number
    poss = [0 0 0; 0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 1 1; 1 0 1; 1 0 0];


    for i=1:1:length(rx_3)

        if rx((i-1)*3+1:i*3) == poss(1,:)'
            rx_3(i) = 1;

        elseif rx((i-1)*3+1:i*3) == poss(2,:)'
            rx_3(i) = 2;

        elseif rx((i-1)*3+1:i*3) == poss(3,:)'
            rx_3(i) = 3;
        elseif rx((i-1)*3+1:i*3) == poss(4,:)'
            rx_3(i) = 4;

        elseif rx((i-1)*3+1:i*3) == poss(5,:)'
            rx_3(i) = 5;

        elseif rx((i-1)*3+1:i*3) == poss(6,:)'
            rx_3(i) = 6;

        elseif rx((i-1)*3+1:i*3) == poss(7,:)'
            rx_3(i) = 7;

        elseif rx((i-1)*3+1:i*3) == poss(8,:)'
            rx_3(i) = 8;

        end

    end

    rx = zeros(length(rx_3),1);

    % assign message bit based on symbol number.
    for i=1:1:length(rx_3)
        if rx_3(i) == 1 || rx_3(i) == 2 || rx_3(i) == 3 || rx_3(i) == 8
            rx(i) = 0;
        else rx(i) = 1;
        end
    end

%% Create array of local pbes and find gaps
local = zeros(length(rx),1);
count = 1;

for i=1:length(rx)-length(tx)
    local(i) = sum(tx ~= rx(i:i+length(tx)-1))/length(tx);
end

index = [];

for i = 1:length(local)-length(tx)
    if local(i) < .25
        index(count) = i;
        count = count + 1;
    end
end

gapind = [];
count = 1;
for i=2:length(index)
    if index(i)-index(i-1) ~= length(tx)
        gapind(count) = i;
        count = count + 1;
    end
```

```
end

gapind % print out the locations of gaps
length(rx)/length(tx) % how many blocks there should have been
length(index) % how many blocks there were

accum = 0;
count = 0;
rx(1:index(1)-1) = [];
accum = index(1)-1;
count = accum;

%% Pad the gaps
for i=2:length(index)

    if mod(index(i)-index(i-1), length(tx)) == 1
        rx(index(i) - accum - 1) = [];
        accum = accum + 1;
        count = count + 1;

    elseif mod(index(i)-index(i-1), length(tx)) ~= 0

        add = length(tx) - mod(index(i)-index(i-1), length(tx));
        temp = [rx(1:index(i) - accum - 1); zeros(add, 1); rx(index(i)-
accum:length(rx))];
        rx = temp;
        accum = accum - add;
        count = count + add;
    end

end

%% Calculate pbe
errors = 0;

for i=1:length(rx)
   if rx(i) ~= tx(mod(i-1, length(tx)) + 1)
       errors = errors+1;
   end
end

pbe = errors/length(rx);
```

Appendix D

```
% This codes provides the Monte Carlo simulation used to create the
phase
% difference error distributions used to estimate the BER of the
% watermarking method's various received signals as well as typical
DBPSK
% and D8PSK signals. This code assumes 45 degree watermark.


%% Generate Shifted Phase Difference Error Distribution and PBE

% Create random array of 0's and 1's signifying where watermark shifts
are
% located
a=round(rand(10000000,1));

% create array to store how many times each vector is shifted
b=zeros(10000000,1);

% Set each element to the number of shifts required by incrementing
when
% the array of shift locations is 1
b(1,1)=a(1);
for i=2:length(a)
    b(i,1)= b(i-1,1)+a(i);
end

% mod 8 the number of shifts since 8 shifts brings it back to 0 degrees
b = mod(b, 8);

% generate an array of ones to initialize the shift vectors
c = ones(10000000,1);

% shift each vector by 45 degrees the determined number of times
for i=1:length(c)
    for j=1:b(i)
        % for each time a vector should be shifted, multiply the result
        % (starting with 1) by a 45 degree vector with amplitude 1.
        c(i) = c(i)*((1/sqrt(2))+(1/sqrt(2))*1i);
    end
end

% The first set is calculated with signal vector amplitude 1
% Each additional loop increases the the signal vector AMPLITUDE by .25
dB
for j=0:.25:14

    be1 = 0;

    % for each vector length, 10,000,000 samples are generated 10 times
for
    % a total of 100,000,000 samples
    rounds = 10;
    for i=1:rounds
        % multiply the vector length by the shifting vector to get the
        % transmitted signal
        iq = c.*(10^(j/10));

        % add noise to get the recived signal
        iq = iq + (randn(10000000,1) + randn(10000000,1)*1i);

        % find the angle between each adjacent pair of received symbols
and
        % convert to degrees (will range from -180 to 180)
        diffiq = iq./circshift(iq,[1 1]);
        diffang = angle(diffiq)*180/pi;

        % count number of errors for Unaware Message Reception
        be1 = be1 + length(find(diffang > 90)) + length(find(diffang <
-90));

    end

    % caluculate Unaware Message Reception PBE
    pbe1s(j*4+1) = be1/(rounds*10000000)


end



%% Generate Non-shifted Phase Difference Error Distribution and PBE's

% The first set is calculated with signal vector amplitude 1
% Each additional loop increases the the signal vector AMPLITUDE by .25
dB
for j=0:.25:14

    be1 = 0;
    be2 = 0;
    be3 = 0;
    be4 = 0;

    rounds = 10;

    for i=1:rounds
        % create the signal vector based on vector length value
        iq = (10^(j/10));

        % add noise to vector
        iq = iq + (randn(10000000,1) + randn(10000000,1)*1i);

        % find the angle between each adjacent pair of received symbols
and
        % convert to degrees (will range from -180 to 180)
        diffiq = iq./circshift(iq,[1 1]);
        diffang = angle(diffiq)*180/pi;

        % count number of errors for DBPSK
        be1 = be1 + length(find(diffang > 90)) + length(find(diffang <
-90));

        % count number of errors for Watermark Reception
        be2 = be2 + length(find((diffang > 22.5 & diffang < 112.5))) +
length(find((diffang > -157.5 & diffang < -67.5)));

        % count number of errors for Aware Message Reception
        be3 = be3 + length(find(diffang > 112.5 & diffang < 180)) +
length(find(diffang < -67.5 & diffang > -180));

        % count number of errors for D8PSK
        be4 = be4 + length(find(diffang > 22.5 & diffang < 67.5)) +
2*length(find(diffang > 67.5 & diffang < 112.5)) ...
            + 3*length(find(diffang > 112.5 & diffang < 157.5)) +
2*length(find(diffang > 157.5)) ...
            + length(find(diffang < -22.5 & diffang > -67.5)) +
2*length(find(diffang < -67.5 & diffang > -112.5)) ...
            + length(find(diffang < -112.5 & diffang > -157.5)) +
2*length(find(diffang < -157.5));

    end

    % caluculate DBPSK PBE
    pbe1(j*4+1) = be1/(rounds*10000000);

    % caluculate Watermark Reception PBE
    pbe2(j*4+1) = be2/(rounds*10000000);

    % caluculate Aware Message Reception PBE
    pbe3(j*4+1) = be3/(rounds*10000000);

    % caluculate D8PSK PBE
    pbe4(j*4+1) = be4/(3*(rounds*10000000));

end
```

Appendix E

The bit streams shown were transmitted in order as read from left to right, top to bottom.

Host Signal Bit Stream

1	1	0	1	0	0	0	1	0	1	0
0	1	1	0	0	1	1	1	0	0	0
1	1	1	0	1	1	1	1	0	0	0
1	0	1	1	1	1	1	0	1	1	0
0	0	1	0	0	1	1	1	0	1	0
0	0	1	1	0	0	1	0	1	1	0
0	0	1	0	0	0	1	1	0	0	0
1	0	0	0	0	1	0	0	0	0	0
1	0	1	0	0	1	1	1	1	0	0
1	0	1	0	1	0	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1	1
1	0	1	1	0	0	0	0	1	1	0
1	1	1	0	0	0	0	0	0	0	0
1	0	0	1	0	1	1	0	1	0	1
0	0	1	0	0	1	1	0	0	1	1
1	1	1	1	1	1	0	0	1	0	0
1	1	0	0	1	1	0	0	1	1	0
1	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	0

1	0	0	1	0	0	1	0	0	1	0
1	0	1	0	0	0	0	1	0	1	0
0	1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	1	0
1	0	1								

Watermark Bit Stream

1	0	1	1	1	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	0	1
1	1	1	1	1	1	0	1	1	0	0
1	0	1	0	0	0	0	0	0	0	1
0	0	1	1	1	1	0	0	1	0	1
0	0	1	0	1	1	1	0	1	1	0
0	1	0	1	0	1	0	0	1	0	1
1	1	1	0	0	0	1	0	1	0	0
0	0	1	0	0	1	1	1	1	1	0
1	0	0	1	1	0	1	0	0	1	1
0	1	0	1	1	0	0	0	1	0	1
0	1	0	0	1	1	0	1	1	0	1
0	0	0	0	1	1	0	0	0	0	1
1	0	1	0	1	1	1	1	1	0	0

1	1	0	0	0	0	1	0	1	0	0
1	0	1	0	0	0	0	0	0	0	1
0	1	0	0	1	0	0	0	1	0	1
0	0	0	1	1	1	0	0	1	0	1
1	1	0	1	0	1	0	0	1	1	1
1	1	1	1	1	0	0	0	0	0	1
1	1	1	1	1	0	1	1	1	1	0
0	0	0	1	0	1	0	0	0	0	1
1	0	1	1	1	1	1	1	0	1	0
1	0	0								

Mixed Bit Stream

1	1	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	0	0	1	1
1	0	0	0	0	0	1	1	1	0	1
1	1	0	0	0	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	0	0
1	1	1	1	0	0	1	1	1	0	1

1	1	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	1	1	1	0	0
1	1	1	0	0	0	1	0	0	1	1
1	0	1	1	0	0	0	0	1	1	0
1	1	1	0	0	1	1	1	0	0	0
1	0	0	1	0	0	1	0	0	0	1
1	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	1	1
1	0	0	0	1	1	0	0	0	1	1
1	0	1	1	1	1	1	1	1	1	0
1	1	1	0	0	0	0	0	1	1	1
0	1	1	0	0	0	1	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0
1	1	0	0	0	1	0	0	1	1	1
1	0	0	1	1	1	1	0	0	1	0
0	1	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	1	0	0	0
1	1	1	1	0	0	0	0	0	0	1
1	1	0	0	1	0	0	0	0	0	0
1	1	1	0	0	1	1	1	0	0	0
1	0	0	1	0	0	0	1	1	1	1
1	1	1	1	0	0	0	1	0	0	1
1	1	1	0	0	1	1	1	1	1	1

0	1	1	1	0	0	1	1	1	1	0
0	1	1	1	1	0	0	0	0	0	1
1	1	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	0	1	1	1	1
1	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	0	1
1	0	0	0	0	1	1	0	1	1	1
1	1	0	1	1	1	0	0	0	0	0
0	0	0	1	0	0	1	1	1	1	0
0	1	1	1	0	0	0	0	0	0	1
0	0	1	0	0	1	1	1	1	1	1
1	0	0	1	1	1	1	1	1	1	1
1	0	1	1	0	1	1	0	1	1	0
0	0	0	1	1	1	0	0	0	1	1
1	0	0	0	0	0	1				

VITA

BRUCE LEBOLD

Candidate for the degree of

Master of Science

Thesis: PHYSICAL LAYER WATERMARKING OF BINARY PHASE-SHIFT KEYED SIGNALS USING STANDARD GNU RADIO BLOCKS

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Master of Science in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in July 2011.

Completed the requirements for the Bachelor of Science in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in 2009.

Experience:

Research Assistant for Dr. George Scheets, Oklahoma State University, Stillwater, OK, May 2010 – May 2011.

Research Assistant for Dr. Alan Cheville, Oklahoma State University, Stillwater, OK, May 2009 – August 2010.

Professional Memberships:

Student Member of the IEEE, 2009 – 2011

Name: Bruce Maxwell Lebold Date of Degree: July, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: PHYSICAL LAYER WATERMARKING OF BINARY PHASE-SHIFT KEYED SIGNALS USING STANDARD GNU RADIO BLOCKS

Pages in Study: 98 Candidate for the Degree of Master of Science

Major Field: Electrical Engineering

Scope and Method of Study:
This thesis discussed the development, implementation, simulation, and testing of a physical layer watermarking method. The method was to use pre-existing GNU Radio building blocks. The main goal of the project was to implement a watermarking method using GNU Radio with the USRP software radios which could also be implemented using standard communications hardware so implementation on SDR systems as well as pre-existing communications systems was possible.

Simulations of the physical layer watermarking system were created using a Monte Carlo method. The generation of a probability distribution of phase difference error was appropriate to analyze the expected performance of the DPSK watermarking system developed.

Testing was performed in a realistic office environment where interference in the tested frequency band was common. A stationary receiver gathered data from a transmitter at various locations and power levels. The bit error rate of the gathered data was determined to analyze performance.

Findings and Conclusions:
While the testing in a real world environment had a limited range of valid analysis due to limited sampling time and interference, the results were comparable with the simulations.

Testing and simulations showed the proposed physical layer watermarking method has the potential to compete with the performance of other authentication focused watermarking methods. In addition, the proposed method could be used to provide a separate, possible secretive, data channel under certain circumstances.

An important benefit of the proposed watermarking method is its ability to be implemented in many SDR or traditional communication systems with no hardware modifications.

ADVISER’S APPROVAL: Dr. George Scheets