# DATA CHUNKING IN QUASI-SYNCHRONOUS DS-CDMA

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Trevor Dalke

May 2014

COMMITTEE MEMBERSHIP

| | |
|---|---|
| TITLE: | Data Chunking in Quasi-Synchronous DS-CDMA |
| AUTHOR: | Trevor Dalke |
| DATE SUBMITTED: | May 2014 |
| COMMITTEE CHAIR: | Dr. Vladimir Prodanov<br>Assistant Professor<br>Electrical Engineering |
| COMMITTEE MEMBER: | Dr. Dennis Derickson<br>Department Chair<br>Electrical Engineering |
| COMMITTEE MEMBER: | Dr. Jane Zhang<br>Associate Professor<br>Electrical Engineering |

ABSTRACT

Data Chunking in Quasi-Synchronous DS-CDMA

Trevor Dalke

DS-CDMA is a popular multiple access technique used in many mobile networks to efficiently share channel resources between users in a cell. Synchronization between users maximizes the user capacity of these systems. However, it is difficult to perfectly synchronize users in the reverse link due to the geographic diversity of mobile users in the cell. As a result, most commercial DS-CDMA networks utilize an asynchronous reverse link resulting in a reduced user capacity. A possible compromise to increase the user capacity in the reverse link is to implement a quasi-synchronous timing scheme, a timing scheme in which users are allowed to be slightly out of synchronization. This paper suggests a possible way to implement a quasi-synchronous DS-CDMA reverse link using the method of "data chunking". The basic premise is derived by making a link between TDMA and synchronous DS-CDMA. By considering some basic TDMA limitations, a proposed "data chunked" quasi-synchronous DS-CDMA system is derived from a TDMA system. The effects of such a system are compared to those of a chip interleaved system. MATLAB simulations are performed to analyze the performance of the system in the presence of small synchronization errors between users. Implementation of guard bands is explored to further reduce errors due to imperfect synchronization between users.

TABLE OF CONTENTS

LIST OF TABLES ..... viii
LIST OF FIGURES ..... ix
1 Introduction ..... 1
2 Background ..... 3
2.1 Cellular Network Overview ..... 3
2.2 Comparison of Multiple Access Techniques ..... 4
2.2.1 Channel Resources ..... 4
2.2.2 Time Division Multiple Access ..... 4
2.2.3 Frequency Division Multiple Access ..... 5
2.2.4 Code Division Multiple Access ..... 6
2.3 DS-CDMA Overview ..... 7
2.3.1 DS-CDMA System ..... 7
2.3.2 Spread Spectrum Nature of DS-CDMA ..... 10
2.3.3 DS-CDMA Timing ..... 11
2.3.4 Multiple Access Interference ..... 12
2.4 DS-CDMA Characteristics ..... 13
2.4.1 Privacy ..... 13
2.4.2 Jam Resistance ..... 13
2.4.3 Frequency Reuse ..... 13
2.4.4 Soft Handoff ..... 14
2.4.5 Resistance to Multipath Fading ..... 14
2.4.6 Power Control ..... 16
3 Implementation of DS-CDMA ..... 18
3.1 Properties of Spreading Codes ..... 18
3.1.1 Cross Correlation ..... 18
3.1.2 Autocorrelation ..... 19
3.2 Forward Link Orthogonal Spreading Codes ..... 19

3.2.1 Walsh Code Generation ..... 19
3.2.2 Example Synchronous DS-CDMA System ..... 21
3.2.3 Walsh Cross Correlation ..... 23
3.3 Reverse Link Orthogonal Spreading Codes ..... 23
3.3.1 Asynchronous Spreading Codes ..... 24
3.3.2 Quasi-Synchronous Spreading Codes ..... 24
3.4 Forward Error Correction ..... 24
3.5 Interleaving ..... 25
3.5.1 Basic Concept ..... 25
3.5.2 Block Interleaving ..... 26
3.5.3 Chip Interleaving ..... 26
3.6 Block Diagram of a Commercial DS-CDMA System ..... 27
3.6.1 Forward Link ..... 28
3.6.2 Reverse Link ..... 29
4 Proposed System ..... 30
4.1 Derivation ..... 30
4.1.1 TDMA to CDMA ..... 30
4.1.2 Chunking in TDMA ..... 33
4.1.3 Chunking in DS-CDMA ..... 35
4.1.4 Relation to Chip Interleaving ..... 35
4.2 Implementation ..... 36
4.2.1 Transmitter Design ..... 36
4.2.2 Receiver Design ..... 38
4.3 Example Baseband Chunking System ..... 39
4.4 Possible Benefits of Chunking ..... 43
4.4.1 Resistance to Small Synchronization Errors between Users ..... 43
4.4.2 Resistance to Burst Interference ..... 44
4.4.3 Flexibility ..... 44

5 Simulation Strategy ..... 45
5.1 Assumptions ..... 45
5.2 Simulation Model ..... 46
5.3 General Practices ..... 47
6 Simulation Results ..... 48
6.1 Walsh Cover Cross Correlation ..... 48
6.2 Spreading Properties ..... 49
6.2.1 Spectrum of Non-Chunked System ..... 49
6.2.2 Spectrum of a Chunked System ..... 50
6.3 Simulations of Complete System ..... 53
6.3.1 Error Distributions ..... 53
6.3.2 Effect of increasing Shift Magnitude ..... 56
6.3.3 Error Magnitude vs. Symbol Position ..... 58
6.3.4 Bit Error Rate vs. Symbol Number ..... 61
6.3.5 Implementation of Guard Bands ..... 65
7 Conclusion ..... 67
7.1 Summary of Results ..... 67
7.2 Future Work ..... 69
References ..... 70
Appendix A: Glossary of Terms ..... 73
Appendix B: Matlab Code ..... 78
System Functions ..... 78
Additional Functions ..... 81
Simulations ..... 82
Appendix C: Analysis of Senior Project Design ..... 91

LIST OF TABLES

Table 1: Modern Mobile Wireless Standards [4 - 6] ..... 2
Table 2: DS-CDMA Encoding Example ..... 22
Table 3: DS-CDMA Decoding Example ..... 22
Table 4: 2 User TDMA Signal with 1 Symbol Time Slots ..... 30
Table 5: 2 User Sum and Difference Signal ..... 31
Table 6: 4 User DS-CDMA System ..... 32
Table 7: Codes for 4 User System ..... 32
Table 8: TDMA with Guard Bands ..... 33
Table 9: TDMA with 2 Symbol Chunks ..... 34
Table 10: TDMA with 3 Symbol Chunks ..... 34
Table 11: DS-CDMA with 3 Symbol Chunks ..... 35
Table 12: All possible chunks for a 2x4 set of messages ..... 40
Table 13: Compressing and Repeating Chunks ..... 40
Table 14: Walsh Cover for Different Chunk Sizes ..... 41
Table 15: Encoding and Superposition in Channel ..... 42
Table 16: Decoding the Chunked DS-CDMA Signal ..... 43

LIST OF FIGURES

Figure 1: Forecasted Growth of Mobile Traffic [2] ..... 1
Figure 2: Cellular Network Layout [11] ..... 3
Figure 3: TDMA Frame Structure [14] ..... 5
Figure 4: TDMA vs. FDMA [15] ..... 6
Figure 5: CDMA [15] ..... 7
Figure 6: DS-CDMA Block Diagram ..... 8
Figure 7: Applying the Spreading Code[18] ..... 9
Figure 8: Correlation Receiver to Recover $s1(t)$ ..... 9
Figure 9: Spreading the Spectrum [19] ..... 11
Figure 10: Multipath [15] ..... 15
Figure 11: Rake Receiver ..... 16
Figure 12: Walsh Code Generation [30] ..... 20
Figure 13: 64 Code Walsh Set ..... 21
Figure 14: Cross Correlation of 64 User Walsh Set ..... 23
Figure 15: Interleaving a 16 Symbol Message ..... 25
Figure 16: Packet Loss in Interleaved System ..... 26
Figure 17: Block Interleaving Transmitter ..... 26
Figure 18: Chip Interleaving Transmitter ..... 27
Figure 19: IS-95 Forward Channel [36] ..... 28
Figure 20: IS-95 Reverse Channel [36] ..... 29
Figure 21: Chip Interleaving Due to Chunking ..... 36
Figure 22: Chunking Transmitter ..... 37
Figure 23: 4 Symbol Compress and Repeat Block ..... 37
Figure 24: Summing Revolver with 2 Users and 4 Symbol chunks ..... 39
Figure 25: System Block Diagram ..... 46
Figure 26: Cyclic Cross Correlation between Chunked Walsh Codes ..... 48
Figure 27: Compressed Message Spectrum ..... 49
Figure 28: Encoded Message Spectrum vs. Walsh Code Spectrum ..... 50
Figure 29: Walsh Code Spectrum ..... 51
Figure 30: Chunked Encoded Message Spectrum ..... 52

Figure 31: Error Distributions for Varying Chunk Sizes ..... 53
Figure 32: Mean Error vs. Chunk Size ..... 54
Figure 33: Error Variance vs. Chunk Size ..... 55
Figure 34: Bit error rate vs. Chunk Size ..... 56
Figure 35: Error Variance vs. Maximum Shift Size ..... 57
Figure 36: Bit error rate vs. Maximum Shift Size ..... 58
Figure 37: Error Magnitude by Position for Maximum Allowed Shift = 1 Chip ..... 59
Figure 38: Error Magnitude by Position for Maximum Allowed Shift = 2 Chips ..... 60
Figure 39: Bit Error Rate vs. Position, Chunk Size = 1 Symbol ..... 62
Figure 40: Bit Error Rate vs. Position, Chunk Size = 4 Symbols ..... 63
Figure 41: Received Bit Error Rate vs. Position, Chunk Size = 16 Symbols ..... 64
Figure 42: Bit Error Rate vs. Position, Chunk Size = 64 ..... 65
Figure 43: Fraction of Quarantined Symbols vs. Chunk Size ..... 66

# 1 Introduction

Over the past decade, there has been an explosion in mobile data consumption driven by the development of inexpensive, widely available wireless devices [1]. Innovations in wireless technology have allowed for the development of advanced cellular networks capable of supporting rapidly increasing traffic. As shown in figure 1, mobile traffic is expected to continue its rapid growth for the foreseeable future. Maintaining this rate of growth will be a technological challenge that will require continued innovation of advanced cellular networks.

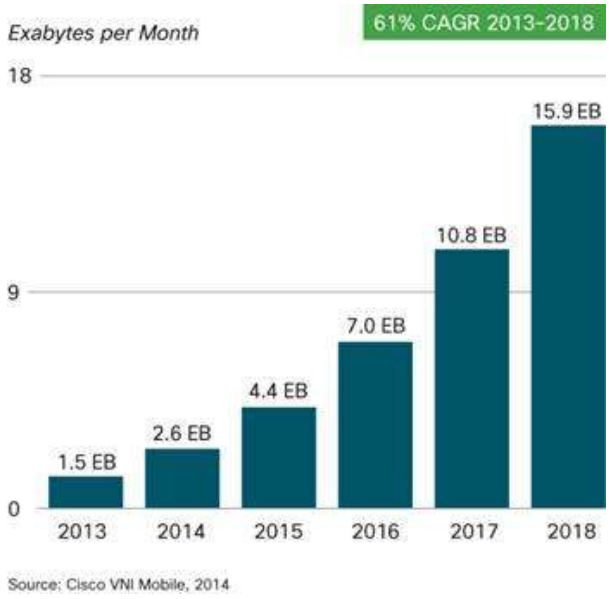


*Figure 1: Forecasted Growth of Mobile Traffic [2]*

Efficiently sharing the network resources between many users is critical to maximizing the capacity of a network. The methods used to accomplish this are known as multiple access techniques, as they allow many users to communicate within a single cell. The continued growth of mobile data consumption is largely dependent on the continued improvement of these techniques.

DS-CDMA (Direct Sequence Code Division Multiple Access) has been the dominant multiple access technique in recent years. It is currently used in many wireless systems, and will likely continue to be used for a long time. Table 1 shows the multiple access techniques used in modern systems, and other relevant information about the systems. Modern variants of DS-CDMA systems include the CDMA2000 and HSPA+ (WCDMA) standards [3].

| Name | Modulation | Channel BW (MHz) | Downstream (Mbit/s) | Upstream (Mbit/s) |
|---|---|---|---|---|
| EDGE (2.9G) | TDMA/FDMA | .2 | 1 | .3 |
| CDMA2000 (3G) | DS-CDMA | 1.25 | 2.45<br>3.1 | .15<br>1.8 |
| HSPA+ (3.5G) | DS-CDMA | 5 | 21<br>42<br>84<br>672 | 5.8<br>11.5<br>22<br>168 |
| LTE (4G) | OFDMA | 1.4,3,5,10,15,20 | 100<br>150<br>300 | 50<br>75 |

*Table 1: Modern Mobile Wireless Standards [4 - 6]*

While OFDMA has gained favor over DS-CDMA for 4G systems, DS-CDMA based systems are still a strong contender for future wireless standards, and continue to be developed [7]. Some of the more promising DS-CDMA based methods being developed include multicarrier CDMA (MC-CDMA) [8] and collaborative CDMA [9]. These potential systems offer advantages that cannot be achieved through current LTE technology. The objective of this paper is to develop a possible improvement to existing and future DS-CDMA based systems, and to investigate the characteristics of such a system.

# 2 Background

## 2.1 Cellular Network Overview

Cellular networks are broken up into non-overlapping hexagonal geographic areas called cells. Within each cell is a base station. This base station manages and communicates with the devices within its cell. Neighboring base stations also communicate with each other to prevent interference between cells. A typical cell structure is shown in figure 2. Cells may be designed to support hundreds of users [10].

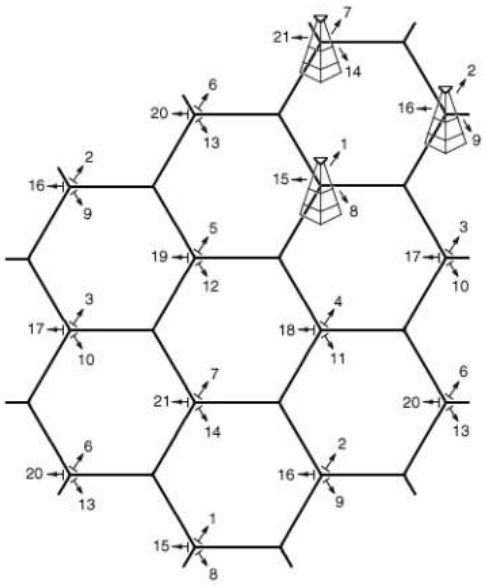


*Figure 2: Cellular Network Layout [11]*

This communication is bidirectional, with all communications from the base station to the user known as the downlink or forward link, and communications from the user to the base station known as the uplink or reverse link. These communications occur over many different frequency channels using multiple access techniques to avoid interference between users. Because mobile devices have a much more limited transmitter and receiver, most communication strategies involve the base station doing

the majority of the work. As a result, the forward and reverse links are often asymmetric [10].

## 2.2 Comparison of Multiple Access Techniques

### 2.2.1 Channel Resources

Multiple access methods are multiplexing schemes, designed to efficiently share the capacity of a channel between many users. The two main components of the transmission that can be shared among the users are the frequency band and time allocations. The frequency spectrum over which mobile devices can communicate is limited to between about 300MHz and 3GHz. The Federal Communications Committee (FCC) regulates usage of these frequencies, assigning different frequency bands to different parties [10]. Because mobile devices generally do not transmit data and receive data continuously, time is considered a resource that can be shared. Multiple access schemes can take advantage of this by scheduling mobile devices so as not to interfere, and dynamically allocating channel resources to active devices.

Naturally, the two simplest ways to share the channel resources are to either separate users in the time domain as in TDMA (Time Division Multiple Access), or in the frequency domain as in FDMA (Frequency Division Multiple Access). Many practical systems, such as LTE and GSM, utilize variations on one or both of these techniques [12, 13]. To illustrate how each of these work, consider a channel shared by K users that has a total bandwidth $W_t$ transmitting over a "frame" with a time period $T_t$.

### 2.2.2 Time Division Multiple Access

In a TDMA scheme, each frame's time period is divided up into K time slots of period $T=T_t/K$. Each time slot is assigned to a different user, so that only one user is

utilizing the channel at a time. Because there is no overlapping between users, all users in this system are orthogonal and do not interfere with each other. This allows them to be easily separated at the receiver. Since all users transmit over equal time periods at the same data rate, they each occupy the same bandwidth $W=W_t$ [13]. A typical TDMA frame structure is shown in figure 3.

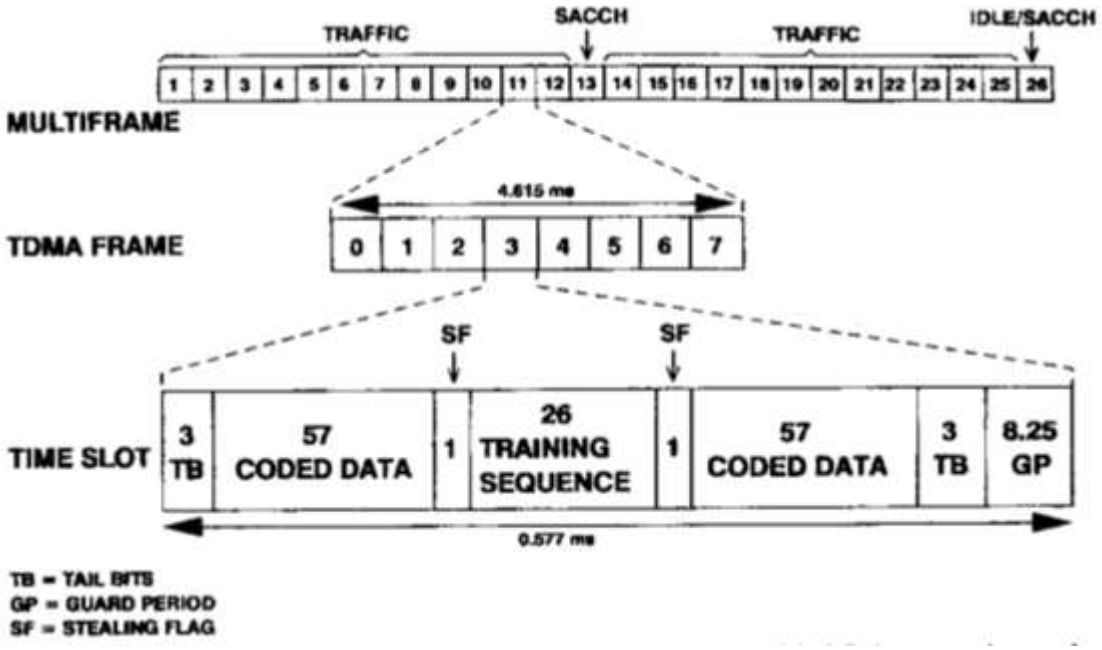


*Figure 3: TDMA Frame Structure [14]*

### 2.2.3 Frequency Division Multiple Access

FDMA is the contrast strategy to TDMA. Instead of sharing the time resource, FDMA users share the frequency resource. This is done by partitioning the channel of the cell into K non-overlapping sub-channels of bandwidth $W=W_t/K$. Each user is assigned its own unique sub-channel, over which it can transmit continuously, utilizing the entire time resource $T=T_t$. Because the sub-channels do not overlap, users are orthogonal and do

not interfere with each other [15]. A comparison of the structure of TDMA and FDMA is shown in figure 4.

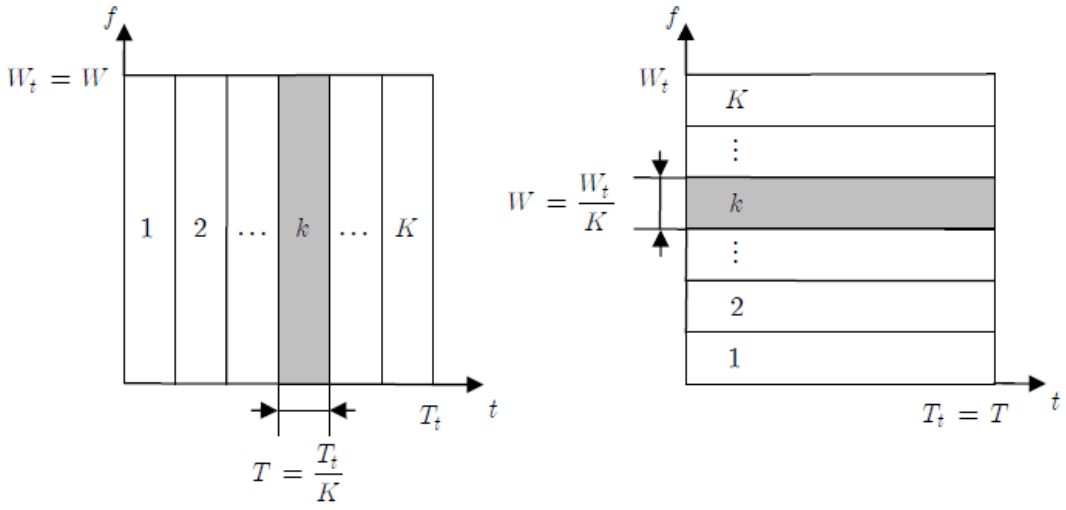


*Figure 4: TDMA vs. FDMA [15]*

### 2.2.4 Code Division Multiple Access

CDMA (Code Division Multiple Access) approaches this problem differently, with each user occupying the entire time interval $T=T_t$, over the entire bandwidth $W=W_t$. Instead of separating users through either time or frequency, users are separated through orthogonality between the messages themselves. This orthogonality is granted by applying a unique orthogonal code to each user. The orthogonality between users effectively separates users into K sub-channels like those used in FDMA. However, these sub-channels are all over the same bandwidth and are therefore stacked on top of one another like in TDMA [16]. The orthogonal channel structure of CDMA is represented in figure 5.

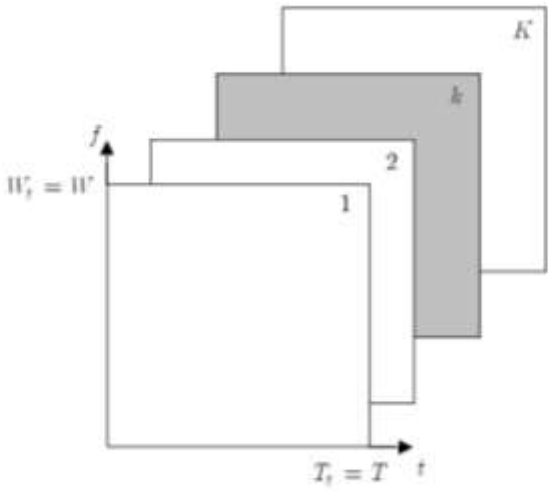


*Figure 5: CDMA [15]*

## 2.3 DS-CDMA Overview

### 2.3.1 DS-CDMA System

The most popular types of CDMA used in mobile communications is known as Direct Sequence CDMA (DS-CDMA). This type of CDMA will be the focus of this paper. A basic DS-CDMA link is shown in figure 6. It consists of the transmitter, channel and receiver.

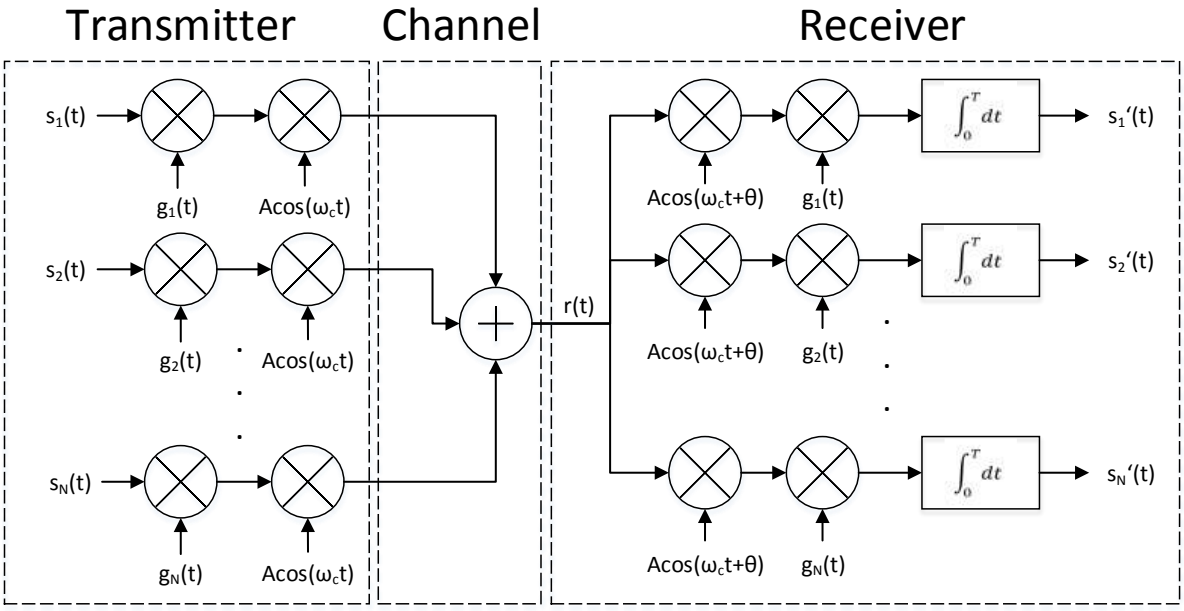


*Figure 6: DS-CDMA Block Diagram*

In this system, there are N users, each transmitting a baseband message: $s_1(t), s_2(t), \dots, s_N(t)$. They are separated by multiplying each signal by a unique orthogonal spreading code: $g_1(t), g_2(t), \dots, g_N(t)$. This results in the signals: $s_1(t)g_1(t), s_2(t)g_2(t), \dots, s_N(t)g_N(t)$. These spreading codes are applied at a much higher bit rate than the bit rate of the original signals, "chipping" the signal in the time domain and spreading the spectrum in the frequency domain [17]. The bit rate of the spreading codes is known as the chip rate. This process is represented in the time domain in figure 7.

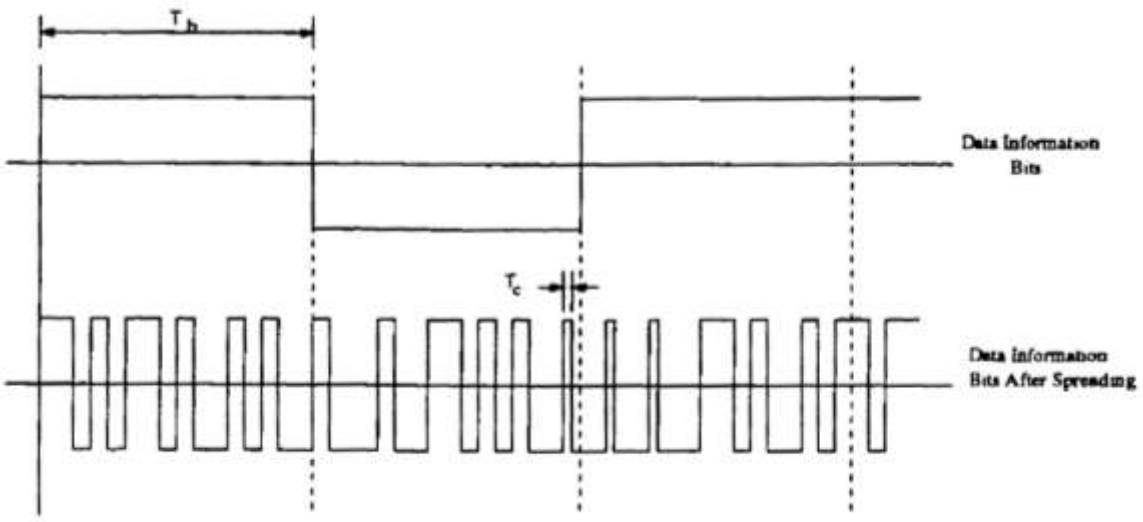


*Figure 7: Applying the Spreading Code[18]*

These spread signals are BPSK modulated up to the carrier frequency, then transmitted. In the channel, the modulated signals sum together forming the single composite signal $r(t)$.

If the desired signal to be received is $s_1(t)$, then the basic necessary receiver necessary to recover message is shown in figure 8. This type of receiver is known as a correlation receiver.

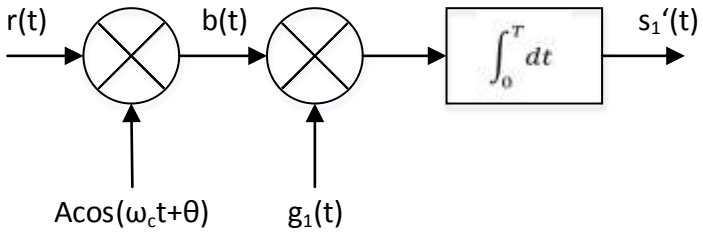


*Figure 8: Correlation Receiver to Recover $s_1(t)$*

The receiver first down-converts the received signal, $r(t)$, recovering the baseband signal: $b(t) = s_1(t)g_1(t) + s_2(t)g_2(t) + \cdots + s_N(t)g_N(t)$. To decode each message,

the receiver multiplies the baseband signal by the appropriate spreading code. If the desired signal to be decoded is $s_1(t)$, then $b(t)$ is multiplied by $g_1(t)$ resulting in the composite signal: $b(t)g_1(t) = s_1(t)g_1(t)^2 + s_2(t)g_1(t)g_2(t) + \cdots + s_N(t)g_1(t)g_N(t)$. The resulting signal is integrated one symbol period at a time, to recover the signal $s_1{}'(t)$:

$$s_1'(t) = \int_0^T b(t)g_1(t)dt$$

$$= \int_0^T s_1(t)g_1^2(t) + s_2(t)g_1(t)g_2(t) + \cdots + s_N(t)g_1(t)g_N(t)dt$$

$$= s_1(t)\int_0^T g_1^2(t)dt + s_2(t)\int_0^T g_1(t)g_2(t)dt + \cdots + s_N(t)\int_0^T g_1(t)g_2(t)dt$$

($s_1(t), s_2(t), \ldots, s_N(t)$ constant over integration interval)

Spreading codes are chosen to be orthogonal to each other, meaning that $\int_0^T g_i^2(t)dt = 1$, and $\int_0^T g_i g_j(t)dt = 0$ for $i \neq j$. Therefore:

$$s_1'(t) = s_1(t)$$

The base station receiver has many branches to decode all of the users' transmitted signals, while a mobile receiver will typically only decode a single message [17]. Because noise is generally uncorrelated to the signals it is almost completely rejected by the correlation receiver.

### 2.3.2 Spread Spectrum Nature of DS-CDMA

The use of spreading codes in DS-CDMA make it a member of the spread spectrum modulation family. The ratio of the chip rate of the spreading code to the

symbol rate of the baseband message signal in a spread spectrum system is known as the spreading factor or the processing gain $G_p$. The spectrum of the signal is spread in the frequency domain by the processing gain. Because power remains constant during the process of spreading, the amplitude of the spread message decreases by $1/G_p$ [19]. This spreading of the spectrum is shown in figure 9.

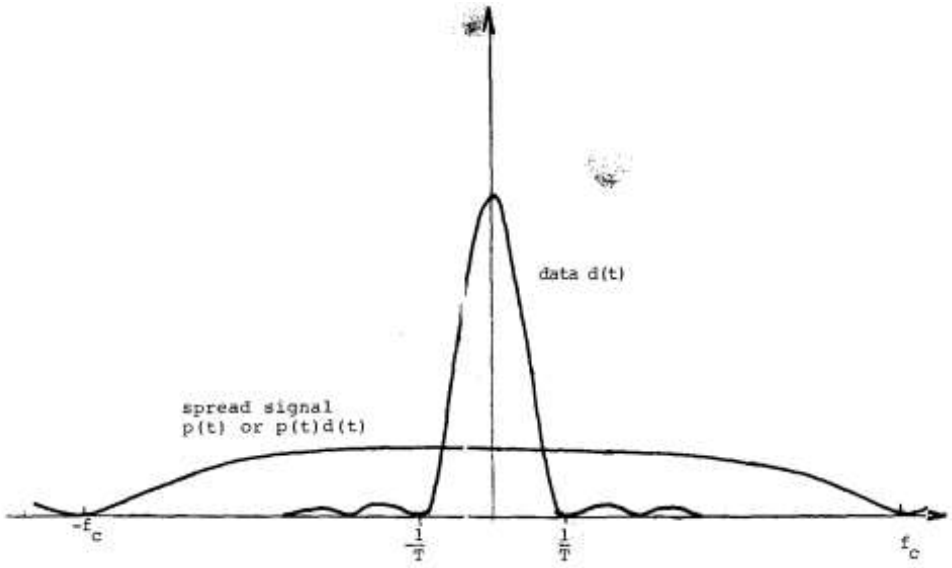


*Figure 9: Spreading the Spectrum [19]*

De-spreading the signal at the receiver by multiplying the signal by the correct spreading code at the receiver restores the original narrow band signal. Using the incorrect spreading code to de-spread the signal does not restore the original signal.

### 2.3.3 DS-CDMA Timing

The type of timing used in DS-CDMA has a large impact on the design of the system. There are two primary types of timing schemes, synchronous and asynchronous. Synchronous systems utilize spreading codes that are perfectly orthogonal, resulting in minimal interference between users. However, synchronous systems can be difficult to

implement because they require precise timing. In most systems, the forward link is synchronous due to all of the transmitted signals originating from the same point simultaneously. In the reverse link, synchronization between users is difficult because of the spatial diversity and less accurate clocks used on mobile devices. The reverse link is therefore usually operated asynchronously, where the users transmit independently of each other. This comes at the cost of increased interference between users, limiting the capacity of the system [20].

### 2.3.4 Multiple Access Interference

The primary reason that synchronous systems are preferred to asynchronous systems is that synchronous systems exhibit less interference between users. This is because synchronous systems utilize perfectly orthogonal spreading codes, while asynchronous systems utilize spreading codes that have finite correlation [16]. This finite correlation causes interference between users. This interference is known as multiple access interference (MAI). While MAI is generally small for most asynchronous spreading codes, it increases with each user. Given the process gain $G_p$ and the number of users in the cell $M$, the received signal to interference ratio $E_b/I_0$ is:

$$\frac{E_b}{I_0} = \frac{G_p}{M-1} \quad [17]$$

MAI is the primary source of interference in the reverse link, and is what ultimately limits the number of users in the cell. Because MAI gradually increases with the number of users in the cell, all users in the cell experience a gradual decline in performance as users are added to the system [21].

## 2.4 DS-CDMA Characteristics

### 2.4.1 Privacy

CDMA systems have inherent privacy built into them. Only authorized users are able to receive the signal, because picking up the signal requires knowledge of the correct spreading code. Attempting to decode the signal using an incorrect spreading code does not restore the original signal, because it is generally not correlated to the signal. The spreading code cannot be easily guessed, because it is typically a very long pseudorandom sequence. Most DS-CDMA standards use a code known as the long code, a PN (pseudonoise) code with a period of $2^{42} - 1$ bits to provide this privacy [22].

### 2.4.2 Jam Resistance

Jamming of a signal is the process of intentionally interfering with a transmission. This is typically done by transmitting a narrowband jamming signal near the frequency of the signal to be jammed. However, this strategy is ineffective for spread spectrum systems because the jamming signal is generally uncorrelated to the signal. Because of this, the power of the jamming signal is spread at the receiver. So while it does interfere with the received signal, the power of the jamming signal over the bandwidth of the jammed signal after de-spreading the signal is very low [17].

### 2.4.3 Frequency Reuse

DS-CDMA has a frequency reuse factor of 100%. This means that neighboring cells can operate over the same frequencies without interference. This is due to the fact that in addition to separating signals within a cell, codes can be applied to separate signals from neighboring cells. The frequency reuse factor of TDMA and FDMA systems is less than 100%, meaning that neighboring cells cannot operate over the same

frequencies unless they are spaced sufficiently far apart. This is because there is no way to separate interfering TDMA or FDMA signals from neighboring cells. As a result, properly designed multicellular DS-CDMA systems generally have a much higher user capacity than pure TDMA or FDMA systems [21].

### 2.4.4 Soft Handoff

In cellular systems it is often necessary to transfer mobile users from one cell into another through the process known as a handoff. The preferred type of handoff is known as a soft handoff. A soft handoff is a seamless handoff in which the mobile device connects to a new cell before terminating connection to the previous cell. This provides a seamless transition that is undetectable to the user. The alternative, the hard handoff, breaks the old connection before connecting to the new cell. This results in an unfavorable abrupt change in signal strength.

Due to the fact that neighboring cells operate over the same frequencies, soft handoffs are much easier to implement in DS-CDMA systems than in other types of systems. Additionally, soft handoffs have the added benefit of increasing the user capacity of the reverse link. For these reasons, soft handoffs are standard in most DS-CDMA systems. This is in contrast to most other systems, where soft handoffs are rarely implemented due to them being much more difficult to implement. These other systems must instead use hard handoffs [23].

### 2.4.5 Resistance to Multipath Fading

Mobile channels tend to exhibit large amounts of multipath fading, destructive interference from delayed and attenuated versions of the original signal. Multipath is caused by reflections and atmospheric effects, as shown in figure 10.

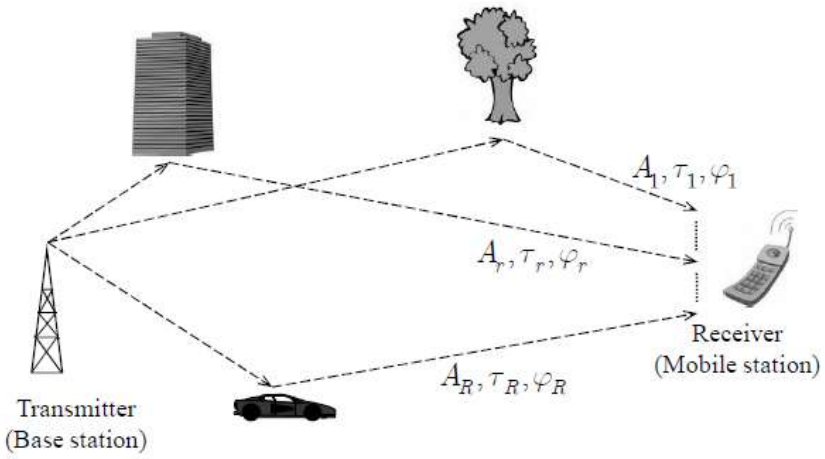


*Figure 10: Multipath [15]*

Multipath fading can be detrimental to many types of systems. However, DS-CDMA is resistant to multipath fading because the multipath signals are generally poorly correlated with the original message. They are therefore mostly rejected by the correlation receiver.

Power lost to multipath can be recovered through the use of a modified correlation receiver known as a rake receiver. A common rake receiver architecture designed to recover a single message is shown in figure 11 [24].

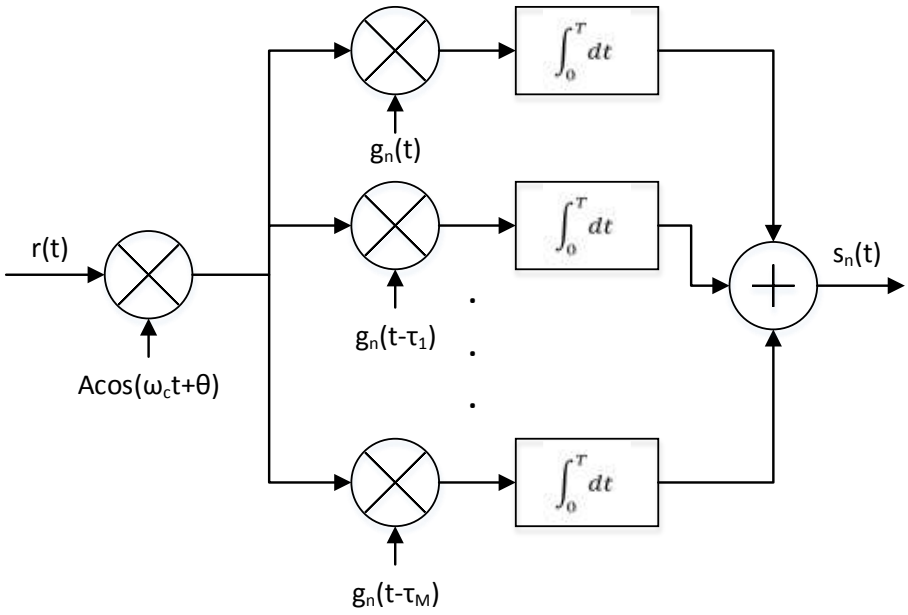


*Figure 11: Rake Receiver*

The rake receiver has many fingers to decode a single message and its multipath components. To accomplish this, each finger applies the same spreading code shifted by a different amount in time. The recovered multipath components, as well as the principle component are appropriately shifted then summed together. This results in a much stronger recovered signal [25].

### 2.4.6 Power Control

Because of the varying location of users with respect to the base station, the signals transmitted by mobile users tend to attenuate varying amounts. Signals transmitted by users near the base station tend to receive little attenuation while far away users tend to be greatly attenuated. If all of the mobile devices were to transmit using the same power, the signals received by the base station would vary in amplitude. While this is not typically a problem for TDMA or FDMA, DS-CDMA relies upon all received

signals having roughly the same power level. The solution to this problem is for the base station to carefully control the transmit power of each of the mobile device to ensure that each signal is roughly equal in amplitude when received by the base station [26].

# 3 Implementation of DS-CDMA

While the concept behind DS-CDMA is relatively simple, many important design considerations must be taken into account to maximize the efficiency of the system. Several of the important components of a practical DS-CDMA system will be discussed in the following sections.

## 3.1 Properties of Spreading Codes

The interference between users in DS-CDMA is primarily a function of the spreading codes used in the system. These properties can be quantified through the correlational properties of the codes.

### 3.1.1 Cross Correlation

The cross correlation between two real valued functions, X and Y, is defined to be: $R_{XY}(\tau) = \int_{-\infty}^{\infty} X(t)Y(t+\tau)\,d\tau$ [27]. It provides a statistical method for comparing the similarity between two signals as they are shifted past one another. For a particular delay, $\tau$, the value of the cross correlation gives a measure of similarity. When the cross correlation is equal to zero, the two functions are orthogonal. The cross correlation properties of spreading codes are important, because the cross correlation properties of the spreading code are inherited by the spread message. Multiple access interference (MAI) is directly related to the cross correlation between users. To minimize the MAI of the system, the cross correlation between spreading codes in a set should be minimized [17].

### 3.1.2 Autocorrelation

The autocorrelation function is simply the cross correlation between a function and itself: $R_x(\tau) = \int_{-\infty}^{\infty} X(t)X(t+\tau)\,d\tau$ [27]. As such, the autocorrelation function always has a peak at $\tau = 0$. A spreading code that has an autocorrelation peak at $\tau = 0$, and is zero everywhere else is uncorrelated with delayed copies of itself. A spreading code with these properties is highly desirable for the purposes of signal identification and rejection of multipath interference. The type of spreading code that closely approximates this desired autocorrelation function is the PN (Pseudonoise) code, a deterministically generated signal that has the properties of a random sequence [28].

## 3.2 Forward Link Orthogonal Spreading Codes

The forward link is nearly always synchronous. The most common type of orthogonal spreading codes used in synchronous systems are Walsh codes. Walsh codes possess the property of being perfectly orthogonal when synchronized.

### 3.2.1 Walsh Code Generation

The Walsh code is a set of 2^N codes, 2^N symbols long. They are created from the Hadamard matrices. The lowest order Hadamard matrix is $H_1 = 1$. Each higher level of the Hadamard matrix is recursively generated through the Hadamard Transform: $H_{N+1} = \begin{bmatrix} H_N & H_N \\ H_N & \overline{H_N} \end{bmatrix}$, where $\overline{H_N}$ is the inverse of $H_N$. Through this process, the second order Hadamard matrix is generated: $H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$. The corresponding Walsh codes are simply the rows of the Hadamard matrix. So from this matrix we obtain the Walsh codes $W(2,1) = [1 \quad 1]$ and $W(2,2) = [1 \quad -1]$. This process of Walsh code generation can

be represented by the tree diagram in figure 12. It can be continued to create an arbitrarily large number of Walsh codes [29].

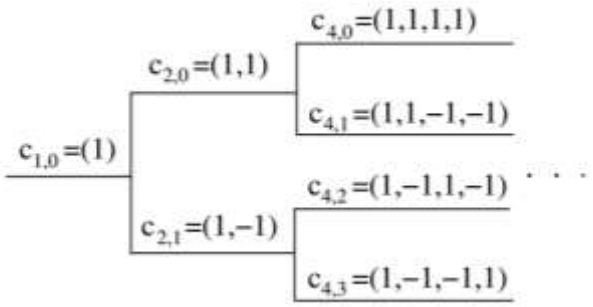


*Figure 12: Walsh Code Generation [30]*

In practice however, other less computationally intensive methods are often used to construct these Walsh codes, such as the Fast-Walsh Fourier Transform [31]. The ease of Walsh code generation is one of the reasons that the Walsh code is the preferred orthogonal spreading code. The CDMA2000 standard uses a set of 256 Walsh codes, and the older standard IS-95 uses 64 Walsh codes. Figure 10, below, is a representation of all codes used in a 64x64 set, as would be used in the IS-95 standard. "1" is represented by a white square, and "-1" is represented by a black square [26].

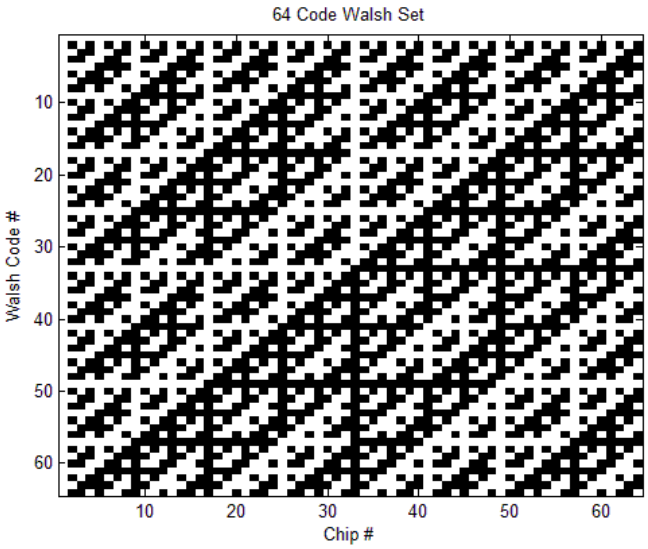


*Figure 13: 64 Code Walsh Set*

When used as a spreading code, the Walsh code is periodically applied at a fast rate so that each message symbol is multiplied by an entire Walsh code. For the IS-95 standard, this means that the Walsh codes are applied at a rate equal to 64 times the symbol rate. The rate at which the Walsh code is applied is known as the chip rate. Since the chip rate is 64 times the symbol rate in this case, the spectrum of these messages are effectively spread by a factor of 64. This repeated Walsh code is known as the Walsh cover, and can be used to spread a message of any arbitrary length [32].

### 3.2.2 Example Synchronous DS-CDMA System

To demonstrate how the Walsh code can be used as an orthogonal spreading code in synchronous DS-CDMA, consider the transmission of the two messages M1 and M2. We follow the steps covered previously to encode the messages, as shown in table 2. Each message is first multiplied by the Walsh cover. In this case, since there are two signals the Walsh cover is simply the repetition of:

$$W_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

After multiplying the signals by the Walsh cover, the signals transmitted then summed in the channel.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Message Signals** | | | | | | | |
| **M1** | 1 | | -1 | | -1 | | 1 | |
| **M2** | 1 | | 1 | | -1 | | 1 | |
| | **Message Signals Sampled at Chip Rate** | | | | | | | |
| **M1** | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| **M2** | 1 | 1 | 1 | 1 | -1 | -1 | 1 | 1 |
| | **Generate Walsh Cover** | | | | | | | |
| **W1** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **W2** | 1 | -1 | 1 | -1 | 1 | -1 | 1 | -1 |
| | **Multiply Each Message by Walsh Cover** | | | | | | | |
| **M1*W1** | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| **M2*W2** | 1 | -1 | 1 | -1 | -1 | 1 | 1 | -1 |
| | **Superposition of Encoded Messages** | | | | | | | |
| **Σ** | 2 | 0 | 0 | -2 | -2 | 0 | 2 | 0 |

*Table 2: DS-CDMA Encoding Example*

To decode the messages we use a correlation receiver, which multiplies the received signal by the appropriate Walsh code then integrates the result one symbol period at a time. This process is shown in table 3.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Multiply Received Signal by Walsh Cover** | | | | | | | |
| **Σ*W1** | 2 | 0 | 0 | -2 | -2 | 0 | 2 | 0 |
| **Σ*W2** | 2 | 0 | 0 | 2 | -2 | 0 | 2 | 0 |
| | **Integrate Over Symbol Period** | | | | | | | |
| **M1'** | 1 | | -1 | | -1 | | 1 | |
| **M2'** | 1 | | 1 | | -1 | | 1 | |

*Table 3: DS-CDMA Decoding Example*

### 3.2.3 Walsh Cross Correlation

To illustrate the properties of the Walsh codes, the average normalized cyclic cross correlation between codes in a 64 Walsh set plotted in MATLAB is shown in figure 14.

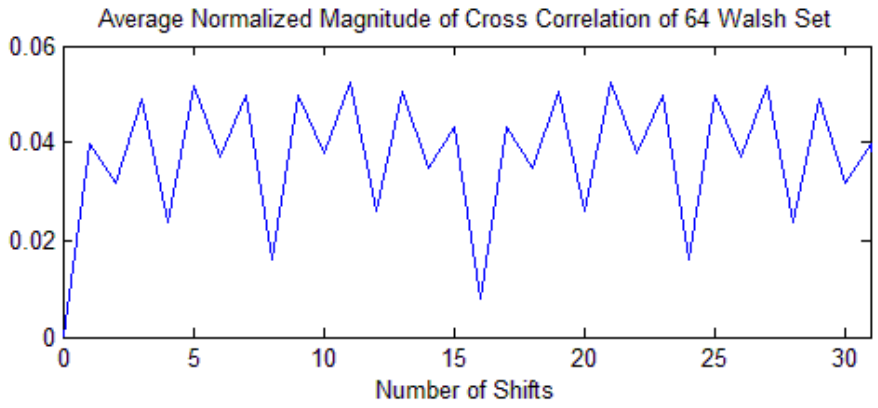


*Figure 14: Cross Correlation of 64 User Walsh Set*

This cross correlation shows that the Walsh codes are indeed perfectly orthogonal to each other when synchronized. However, the cross correlation becomes finite and relatively large in systems that have any synchronization error between users. This confirms that any system that uses Walsh codes must require strict timing requirements to ensure complete synchronization between users. Even a difference of 1 chip between Walsh codes results in significant correlation between users [33].

## 3.3 Reverse Link Orthogonal Spreading Codes

Generally, the reverse link cannot be perfectly synchronized. As a result, some other timing scheme must be used. Typically these systems are either asynchronous, but quasi-synchronous reverse links have also been proposed. Each of these two cases utilize different spreading codes.

### 3.3.1 Asynchronous Spreading Codes

The reverse link in most DS-CDMA systems is asynchronous, meaning that users transmit data at timings completely independently of each other. The asynchronous reverse link in IS-95 for example uses PN codes to separate users because they have low constant cross correlation with each other. There are many other types of asynchronous spreading codes that can be used in this type of system. As previously discussed, the disadvantage of using asynchronous spreading codes is that they always have a finite cross correlation between each other, leading to MAI [28].

### 3.3.2 Quasi-Synchronous Spreading Codes

An alternative to using an asynchronous reverse link is a quasi-synchronous reverse link. Such systems operate synchronously using spreading codes that are orthogonal when synchronized, and lose their orthogonality in a gradual manner. This allows the system to operate synchronously, but with some tolerance for synchronization errors between users. One proposed set of spreading codes is that of the scrambled Walsh code as suggested in [20]. This paper will explore the implementation of a quasi-synchronous reverse link using Walsh codes.

## 3.4 Forward Error Correction

Even in well-designed systems, there will be a finite number of errors resulting from the imperfect nature of communication channels. Because errors lead to a degradation of the user experience, nearly all digital communication systems employ forward error correction (FEC) to detect and correct these types of errors at the receiver. FEC is a type of channel coding, in which messages are encoded in a redundant way through the use of an error correction code (ECC). The resulting redundant bits are used

by the receiver to detect and correct errors. In cases where FEC is unable to completely correct the errors, the packet must be retransmitted through an automatic repeat request (ARQ). This results in large variable delays, and is therefore highly undesirable [34].

## 3.5 Interleaving

### 3.5.1 Basic Concept

FEC can only effectively correct errors that are sufficiently spaced apart. FEC schemes are therefore ineffective at correcting long strings of errors, known as burst errors. The solution to this problem is to spread burst errors across the message by increasing the time diversity of the signal. This is done by mixing the bits of the signal in a predictable way before transmission (interleaving), then restoring them back to their original order (de-interleaving) at the receiver. An example of interleaving a message is shown in figure 15.

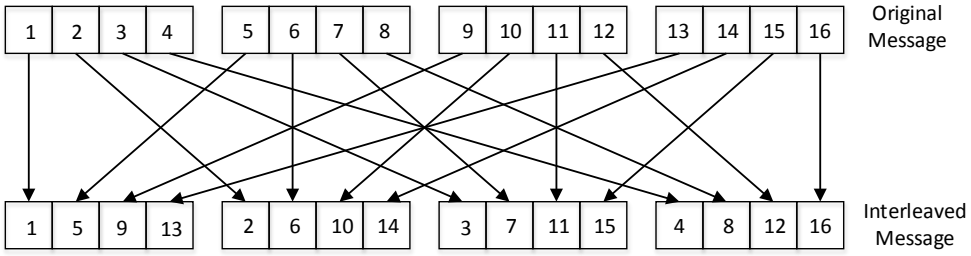


*Figure 15: Interleaving a 16 Symbol Message*

Now consider the case in which an entire packet is lost. If the original signal had been transmitted, then FEC would likely be insufficient because FEC is not capable of correcting long strings of errors. However, if the interleaved signal is transmitted then de-interleaved at the receiver this spreads the errors across the signal as in figure 16. This

case should be easily corrected through FEC due to the spreading out of the errors across the message [35].

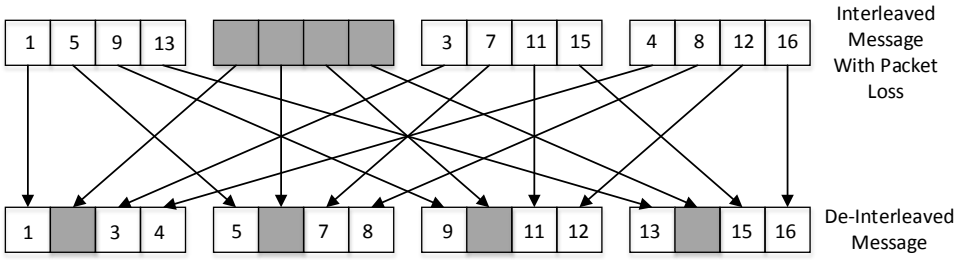


*Figure 16: Packet Loss in Interleaved System*

### 3.5.2 Block Interleaving

There are two primary types of interleaving that can be used in DS-CDMA systems. Block interleaving refers to using an interleaver before spreading the signal, as in figure 17. This is the type of interleaving used in the majority of DS-CDMA systems, such as the IS-95 standard. De-interleaving is performed at the receiver after de-spreading the message using a correlation receiver [18].

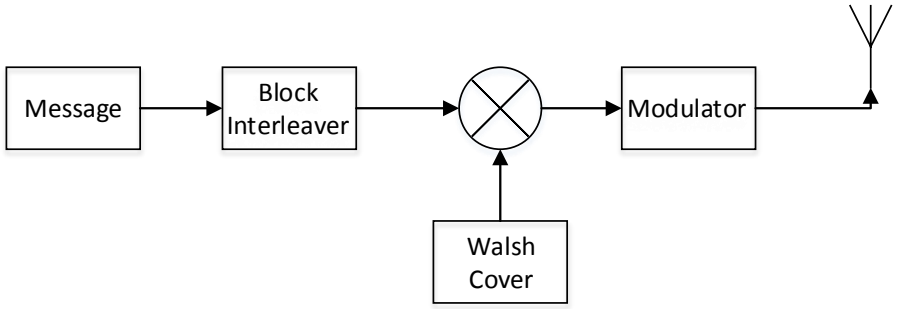


*Figure 17: Block Interleaving Transmitter*

### 3.5.3 Chip Interleaving

Chip interleaving refers to using an interleaver after spreading the signal, as shown in figure 18. De-interleaving is performed at the receiver before de-spreading the system using the correlation receiver. Chip interleaving is much more effective when it

comes to mitigating burst errors, because it is able to interleave to a greater extent than block interleaving leading to greater time diversity. However, the hardware required to chip interleave is more difficult to implement because it is required to operate on a signal with a much higher bit rate [18].

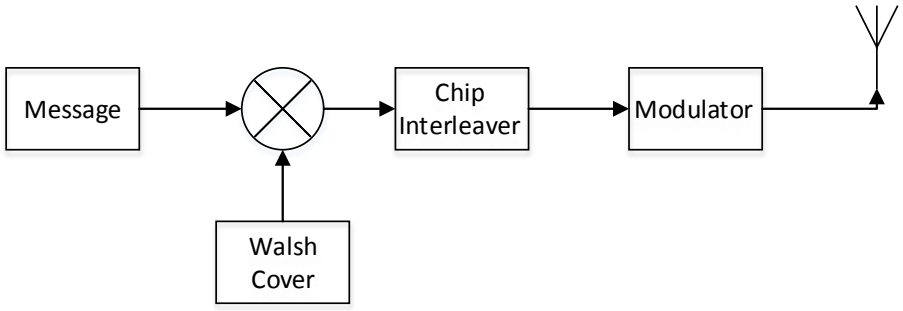


*Figure 18: Chip Interleaving Transmitter*

## 3.6 Block Diagram of a Commercial DS-CDMA System

Now that the components have been discussed, a commercial DS-CDMA system can be analyzed. While many DS-CDMA standards exist, they are generally pretty similar. This paper will mostly consider the IS-95 standard, as it is the basis from which most other DS-CDMA standards were formed. Most properties of IS-95 can be generalized to the other types of systems. The block diagrams of the forward and reverse links are shown in figures 19 and 20. The actual systems are very complex, and many details will not be discussed.

### 3.6.1 Forward Link

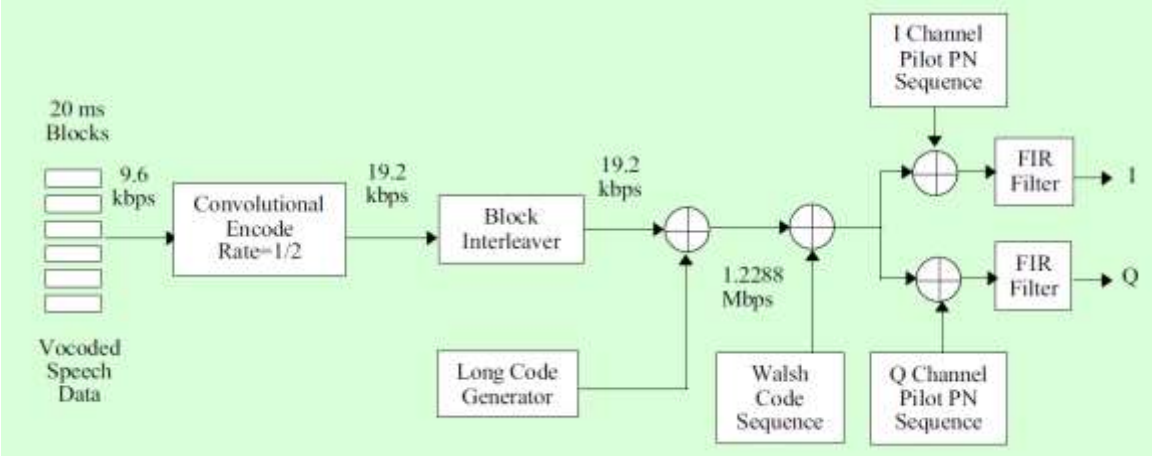


*Figure 19: IS-95 Forward Channel [36]*

The forward channel of the IS-95 standard is synchronous DS-CDMA. The signals to be transmitted are broken into 20ms blocks at 9.6kbps. Forward error correction is applied to each of these signals through convolutional encoding, then block interleaving. This leaves the signal at a data rate of 19.2kbps. There are therefore 19.2kbps*20ms = 384 bits per block. The forward channel is synchronous, so the Walsh code is used to create the orthogonal channels. Each signal is assigned a unique code from the set of 64 Walsh codes. Because there are 64 Walsh codes, the Walsh cover is applied at a chip rate 64*19.2kbps = 1.2288Mbps.

The signal is also spread by the long code and two short codes in quadrature. The long code is a PN code that has a period of $2^{42} - 1$ chips. This code is used to separate users in different base stations. The short code is a PN code that has a period of $2^{15} - 1$ chips. All of spreading codes are applied at 1.2288Mbps. The output signals are then filtered, and transmitted using QPSK modulation. Timing information is transmitted

through the use of a pilot channel and a sync channel. These channels are used to help the mobile receiver coherently demodulate the signal [32].

### 3.6.2 Reverse Link

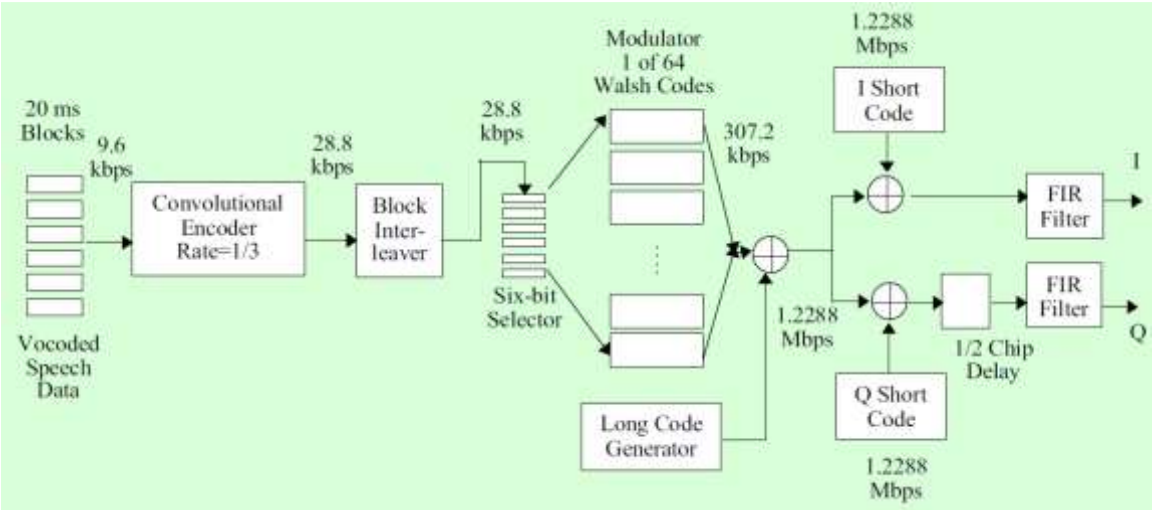


*Figure 20: IS-95 Reverse Channel [36]*

The reverse link is similar to the forward link, but with a few changes. The main difference is that the reverse link is asynchronous. While the Walsh code is used in the reverse link, it is not used for spreading. It is instead used for 64-ary modulation. The users are instead separated entirely by short codes. No timing information is transmitted in the reverse link. Instead, the timing information is extracted from the timing offset of the spreading codes [32].

# 4 Proposed System

## 4.1 Derivation

### 4.1.1 TDMA to CDMA

A basic synchronous DS-CDMA system can be developed from a simple TDMA system if a few considerations are observed. Consider the two user system, consisting of User A and User B. The simplest TDMA scheme for these two users is to simply take turns transmitting one symbol at a time as in table 4. In order to maintain the original data rate of each signal, the TDMA signal is transmitted at twice the symbol rate. The TDMA signal therefore requires twice the bandwidth of the original messages.

| User A | a1 | | a2 | | a3 | | a4 | |
|---|---|---|---|---|---|---|---|---|
| User B | b1 | | b2 | | b3 | | b4 | |
| $TDMA_A$ | a1 | | a2 | | a3 | | a4 | |
| $TDMA_B$ | | b1 | | b2 | | b3 | | b4 |

*Table 4: 2 User TDMA Signal with 1 Symbol Time Slots*

One of its primary problems of such a system is its sensitivity to burst interference. If one or more time slots within a frame of the TDMA signal are completely wiped out due to any source of burst interference, one or more symbols between several users are completely lost because all of the information associated with the symbol is lost. This problem gets worse as the number of users increases due to the time slots getting smaller with a greater number of users.

It would be preferable to develop a multiple access system in which burst interference is distributed among the users. This would ensure that each user faces an equal, less severe loss of signal when faced with burst interference than in the original TDMA signal. To accomplish this, we can transmit each symbol across multiple timeslots

on top of each other at the same rate as the TDMA signal. In order to do this we must maintain orthogonality between users.

A common way to orthogonally combine two signals is to code the bits of the signal in sum and difference pairs. This allows the two individual signals to occupy one channel, while still being separable. This method is commonly used in stereo transmission, where the left (L) and right (R) channels are in bit pairs coded as L+R and L-R [37]. This process can be followed for the TDMA signal to combine user A and user B by treating them as the left and right channels, as shown in table 5.

| User A | a1 | | a2 | | a3 | | a4 | |
|---|---|---|---|---|---|---|---|---|
| User B | b1 | | b2 | | b3 | | b4 | |
| $\Sigma\Delta_{AB}$ | a1+b1 | a1-b1 | a2+b2 | a2-b2 | a3+b3 | a3-b3 | a4+b4 | a4-b4 |

*Table 5: 2 User Sum and Difference Signal*

Upon inspection, it is clear that we have constructed a basic synchronous DS-CDMA system using Walsh codes. User A's symbols were each multiplied by the code 1,1 and User B's symbols were multiplied by the code 1,-1. Thus, we have used a set of Walsh codes:

$$W_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

To consider how such a system with a larger number of users would work, consider the first symbol of a 4 user system. To establish orthogonality between users, we first make A and B orthogonal to each other and C and D orthogonal to each other. This is done by finding sum and difference between users A and B, and between users C and D as was done previously. To make the two resulting signals orthogonal, we can simply repeat this process by finding the sum and difference between these new signals, as shown in table 6.

| | | | | |
|---|---|---|---|---|
| User A | a1 | | | |
| User B | b1 | | | |
| User C | c1 | | | |
| User D | d1 | | | |
| $\Sigma\Delta_{AB}$ | a1+b1 | | a1-b1 | |
| $\Sigma\Delta_{CD}$ | c1+d1 | | c1-d1 | |
| $\Sigma\Delta_{ABCD}$ | (a1+b1)+(c1+b1) | (a1+b1)-(c1+d1) | (a1-b1)+(c1-d1) | (a1-b1)-(c1-d1) |

*Table 6: 4 User DS-CDMA System*

If we observe this resulting signal, we have created a set of four orthogonal codes. These codes are represented in table 7.

| $\Sigma\Delta_{ABCD}$ | (a1+b1)+(c1+d1) | (a1+b1)-(c1+d1) | (a1-b1)+(c1-d1) | (a1-b1)-(c1-d1) |
|---|---|---|---|---|
| Code for A | 1 | 1 | 1 | 1 |
| Code for B | 1 | 1 | -1 | -1 |
| Code for C | 1 | -1 | 1 | -1 |
| Code for D | 1 | -1 | -1 | 1 |

*Table 7: Codes for 4 User System*

Once again, through the basic process of finding the sum and difference, we have found another set of Walsh codes:

$$W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

This process can be continued to develop a synchronous DS-CDMA signal for an arbitrary number of users, and the appropriate Walsh codes. By converting the TDMA system into a DS-CDMA system, we have solved the original problem of weakness to burst interference by implementing a system that has the “burden sharing” property. This exercise demonstrates the similarity between TDMA and synchronous DS-CDMA systems. We would like to further study the characteristics of TDMA so that we can better understand the characteristics of DS-CDMA systems. This will potentially allow us

to apply solutions to TDMA problems to a synchronous DS-CDMA system in an attempt to improve it.

### 4.1.2 Chunking in TDMA

Synchronization between users is a major concern in TDMA systems and synchronous DS-CDMA systems. Users in perfectly synchronized TDMA systems do not interfere because users only transmit during their assigned time slot. When synchronization is not perfect, symbols between users begin overlapping resulting in interference. This problem becomes much worse in systems with more users, ultimately limiting the capacity of the system [13].

One possible solution to this problem is to include guard bands, intervals between time slots to help prevent interference between users. The duration of these intervals should be longer than the uncertainty in the synchronization error so as to prevent inter-symbol interference. However, this strategy becomes problematic with larger synchronization uncertainty as the guard bands become large relative to the time slot intervals. This results in a significant reduction of the effective data rate, making this method unfeasible for some channels. Table 8 shows the previously used TDMA example with the addition of guard bands between users.

| User A | a1 | | | a2 | | | a3 | | | a4 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| User B | b1 | | | b2 | | | b3 | | | b4 | | |
| $TDMA_A$ | a1 | | | a2 | | | a3 | | | a4 | | |
| $TDMA_B$ | | | b1 | | | b2 | | | b3 | | | b4 |

*Table 8: TDMA with Guard Bands*

Since the guard band interval is limited by the time slot interval, most TDMA systems use larger time slots and each user transmits many symbols, a “chunk” of data, per time slot. This has the effect of reducing the total number of time slots, and the

number of guard bands required. Because fewer guard bands are necessary, this would also allow for the guard bands to be larger, allowing for more uncertainty in the synchronization between users. An example of chunked TDMA, where each chunk consists of 2 symbols is shown in table 9.

| User A | a1 | | a2 | | a3 | | a4 | |
|---|---|---|---|---|---|---|---|---|
| User B | b1 | | b2 | | b3 | | b4 | |
| $TDMA_A$ | | a1 | a2 | | | a3 | a4 | |
| $TDMA_B$ | | | | b1 | b2 | | | b3 |

*Table 9: TDMA with 2 Symbol Chunks*

The chunked TDMA signal requires fewer guard bands because it requires fewer frames to transmit the same amount of data. It can therefore support larger guard bands while still maintaining a reasonable data rate. The cost of doing this is additional latency. In this example, User B experiences 3 times as much latency as the non-chunked case.

This can be extended to larger chunk sizes. Table 10 shows the previous TDMA signal with 3 symbol chunks.

| User A | a1 | | a2 | | a3 | | a4 | |
|---|---|---|---|---|---|---|---|---|
| User B | b1 | | b2 | | b3 | | b4 | |
| $TDMA_A$ | | | a1 | a2 | a3 | | | |
| $TDMA_B$ | | | | | | b1 | b2 | b3 |

*Table 10: TDMA with 3 Symbol Chunks*

In this case, User B faces 5 times as much latency as the non-chunked case. From these examples, it is clear that some users in chunked TDMA face large amounts of latency. It can be shown that maximum delay experienced due to this phenomenon is approximately equal to the frame period of the signal. Because a frame consists of one time slot from each user, the frame period is dependent on the number of users, and duration of each

time slot. As a result, the maximum latency of a chunked TDMA signal becomes very large with a larger number of users and a larger chunk size.

This latency is a limiting factor in practical TDMA systems. For example, GSM TDMA channels have frames consisting of 456 bits transmitted at a data rate of 22.8 kbps. This means that each channel has an inherent minimum latency of 456/22800 = 20 ms, without even considering other sources of delay. This is a fundamental limitation of TDMA systems [13].

### 4.1.3 Chunking in DS-CDMA

To improve this latency, we can transform this chunked TDMA signal into a synchronous DS-CDMA signal by applying the sum and difference strategy used previously, considering each of the chunks to be a unit as shown in table 11.

| User A | a1 | | a2 | | a3 | | a4 | |
|---|---|---|---|---|---|---|---|---|
| User B | b1 | | b2 | | b3 | | b4 | |
| $Chunk_A$ | | | a1 | a2 | a3 | a1 | a2 | a3 |
| $Chunk_B$ | | | b1 | b2 | b3 | b1 | b2 | b3 |
| CDMA | | | a1+b1 | a2+b2 | a3+b3 | a1-b1 | a2-b2 | a3-b3 |
| A Code | | | 1 | | | 1 | | |
| B Code | | | 1 | | | -1 | | |

*Table 11: DS-CDMA with 3 Symbol Chunks*

Notice that the latency of this implementation is the same for all users and is independent of the chunk size. It is only dependent on the number of users. In fact, the latency is simply equal to the symbol period of the original message. Because of this, DS-CDMA can use any chunk size while maintaining a low amount of latency for all users.

### 4.1.4 Relation to Chip Interleaving

This method of chunking essentially chip interleaves the DS-CDMA system. This can be realized by comparing the DS-CDMA system without chunking to one with

chunking. Figure 21 shows the chip interleaved nature of the 3 symbol chunked DS-CDMA signal.

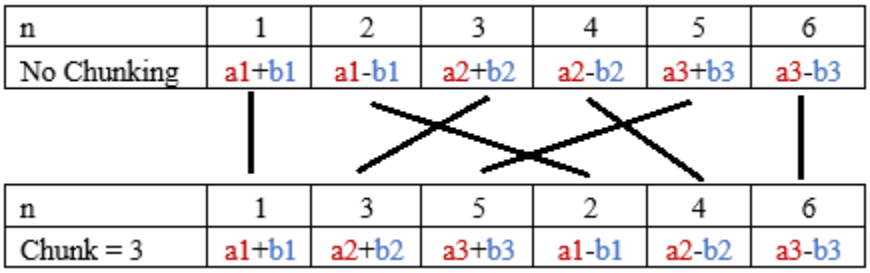


*Figure 21: Chip Interleaving Due to Chunking*

In this example, it is clear that none of the data to be transmitted is altered. It is merely rearranged in a predictable fashion. By chunking the data, we have essentially created a chip interleaved DS-CDMA system without the use of a dedicated chip interleaver. This interleaving only occurs across the symbols contained in one chunk. Since the chunk size is only 3 symbols in this case, only 3 symbols get interleaved at a time. This result makes clear that if we wish to maximize the time diversity of the system, we should use the largest possible chunk size.

## 4.2 Implementation

### 4.2.1 Transmitter Design

The transmitter required to make the chunked DS-CDMA system work is a slight modification of a basic DS-CDMA transmitter. It takes the message to be encoded, then compresses and repeats the signal one chunk at a time to make the chunked signal. This signal is then multiplied by the Walsh cover, a Walsh code that repeats once per each compressed chunk. A simplified block diagram of the chunking process is shown in figure 22.

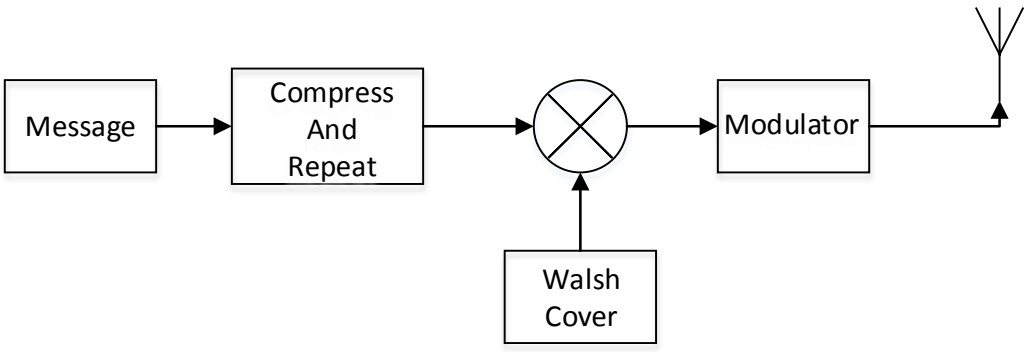


*Figure 22: Chunking Transmitter*

To implement this system, we need the compress and repeat block. A possible implementation is shown in figure 23. Consider the case in which there are N users, which a desired chunk size M.

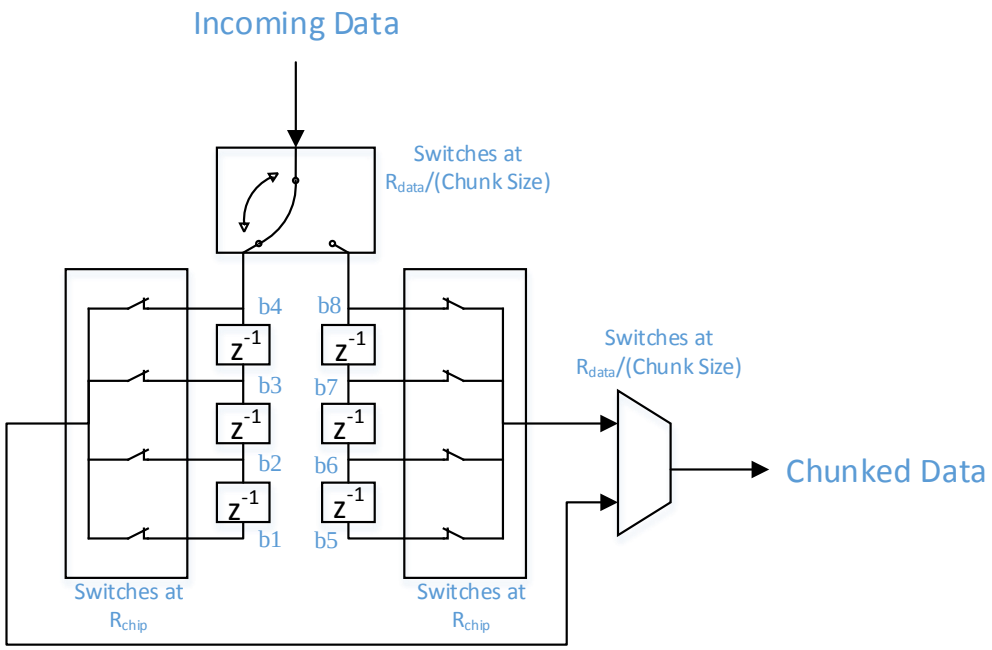


*Figure 23: 4 Symbol Compress and Repeat Block*

This block can be created using two circular buffers of length M, and a 2 to 1 mux. One register loads chunks of data at the symbol rate, while the other rotates at a fast rate, $R_{chip}$.

The faster rate is equal to the chip rate, the original data rate of the message times the number of users in the system. The two registers continually alternate roles. The 2 to 1 multiplexer samples the rotating register to create the chunked signal. This example can create chunks up to 4 symbols long. The basic idea can be extended to arbitrarily large chunk sizes.

### 4.2.2 Receiver Design

The receiver necessary to decode this is a variation on the correlation receiver. It multiplies each signal by the appropriate Walsh cover, at the same data rate originally used. This results in N total received messages. These messages are still chunked, and at the chip rate. To decode these messages, we use a "summing revolver" of length M. This device sums N bits M chips apart of the received baseband signal $b(k)$ to recover the decoded signal $d_i(n)$ using the appropriate Walsh cover $W_i(k)$as expressed:

$$d_i(n) = \frac{1}{N}\sum_{k=0}^{N-1} W_i(kM+n)\, b(kM+n), \qquad 0 \le n < M$$

After decoding M symbols of the signal, the summing revolver must be advanced by $(M-1) * N$ chips. The process is repeated until the entire signal is recovered. The process is visualized in figure 24 for a system with a chunk size M=4 symbols and N=2 users.

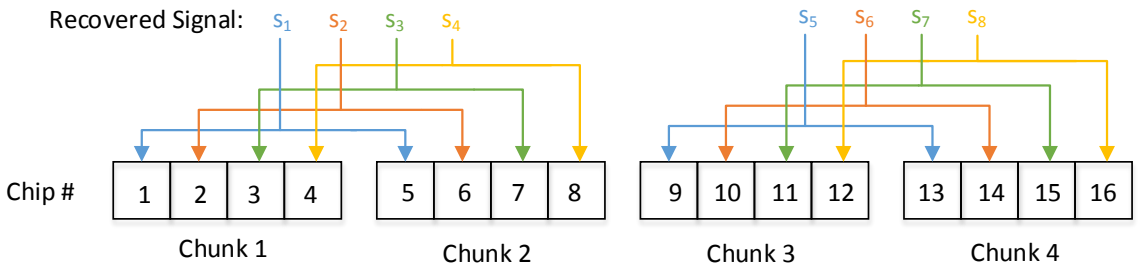


*Figure 24: Summing Revolver with 2 Users and 4 Symbol chunks*

In this example, M=4 and N=2. Therefore, the signal is broken up into M=4 chip blocks. Since N=2 users, each symbol is composed of two chips. This diagram depicts the two chips that must be integrated at a time to recover each symbol. Note that each chip is only integrated once.

## 4.3 Example Baseband Chunking System

To demonstrate the full process in a baseband system as will later be simulated, consider the two signals M1 and M2 transmitted by two different users. Since these signals are only 4 symbols, there are only three ways that they can be divided up into equal sized chunks. These options are shown below, table 12. In this example, the symbols are color coded for clarity.

| Breaking the Message Signals into Equal Sized Chunks | | | | |
|---|---|---|---|---|
| | **Message Signals** | | | |
| **M(1)** | 1 | -1 | -1 | 1 |
| **M(2)** | 1 | 1 | -1 | 1 |
| | **Chunk Size = 1 Symbol** | | | |
| | *Chunk 1* | *Chunk 2* | *Chunk 3* | *Chunk 4* |
| **$M_1$(1)** | 1 | -1 | -1 | 1 |
| **$M_1$(2)** | 1 | 1 | -1 | 1 |
| | **Chunk Size = 2 Symbols** | | | |
| | *Chunk 1* | | *Chunk 2* | |
| **$M_2$(1)** | 1 | -1 | -1 | 1 |
| **$M_2$(2)** | 1 | 1 | -1 | 1 |
| | **Chunk Size = 4 Symbols** | | | |
| | *Chunk 1* | | | |
| **$M_4$(1)** | 1 | -1 | -1 | 1 |
| **$M_4$(2)** | 1 | 1 | -1 | 1 |

*Table 12: All possible chunks for a 2x4 set of messages*

Each of these chunks is compressed and repeated to make the chunked signals R(1) and R(2). Since there are only two messages being encoded, they are compressed by a factor of 2. This process is shown in table 13.

| Compressing and Repeating Each Chunk | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Chunk Size = 1 Symbol** | | | | | | | |
| | *Chunk 1* | *Chunk 1* | *Chunk 2* | *Chunk 2* | *Chunk 3* | *Chunk 3* | *Chunk 4* | *Chunk 4* |
| **$R_1$(1)** | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| **$R_1$(2)** | 1 | 1 | 1 | 1 | -1 | -1 | 1 | 1 |
| | **Chunk Size = 2 Symbol** | | | | | | | |
| | *Chunk 1* | | *Chunk 1* | | *Chunk 2* | | *Chunk 2* | |
| **$R_2$(1)** | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 |
| **$R_2$(2)** | 1 | 1 | 1 | 1 | -1 | 1 | -1 | 1 |
| | **Chunk Size = 4 Symbols** | | | | | | | |
| | *Chunk 1* | | | | *Chunk 1* | | | |
| **$R_4$(1)** | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 |
| **$R_4$(2)** | 1 | 1 | -1 | 1 | 1 | 1 | -1 | 1 |

*Table 13: Compressing and Repeating Chunks*

Each compressed and repeated section gets multiplied by the Walsh code. Since there are two signals in this case, we use:

$$W = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

This Walsh code is applied over each chunk. Therefore, the Walsh code must be applied at a different bit rate for each chunk size. This repeated Walsh code is referred to as the Walsh cover, and is shown in table 14.

| **Generating Walsh Cover** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Chunk Size =1 Symbol, Walsh Compressed and Repeated 4 times** | | | | | | | |
| **$W_1(1)$** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **$W_1(2)$** | 1 | -1 | 1 | -1 | 1 | -1 | 1 | -1 |
| | **Chunk Size =2 Symbols, Walsh Compressed and Repeated 2 times** | | | | | | | |
| **$W_2(1)$** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **$W_2(2)$** | 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 |
| | **Chunk Size =4 Symbols, Walsh Compressed and Repeated 1 time** | | | | | | | |
| **$W_4(1)$** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **$W_4(2)$** | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 |

*Table 14: Walsh Cover for Different Chunk Sizes*

Each message is multiplied by the appropriate Walsh cover, then transmitted. Under ideal conditions, in a perfect channel, the baseband signals sum together with no offset regardless of the carrier frequency. This process can be seen in table 15.

| Encoding and Superposition in Channel | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Chunk Size =1 Symbol | | | | | | | |
| $R_1(1)^* W_1(1)$ | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| $R_1(2)^* W_1(2)$ | 1 | -1 | 1 | -1 | -1 | 1 | 1 | -1 |
| $\Sigma_1$ | 2 | 0 | 0 | -2 | -2 | 0 | 2 | 0 |
| | Chunk Size =2 Symbols | | | | | | | |
| $R_2(1)^* W_2(1)$ | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 |
| $R_2(2)^* W_2(2)$ | 1 | 1 | -1 | -1 | -1 | 1 | 1 | -1 |
| $\Sigma_2$ | 2 | 0 | 0 | -2 | -2 | 2 | 0 | 0 |
| | Chunk Size =4 Symbols | | | | | | | |
| $R_4(1)^* W_4(1)$ | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 |
| $R_4(2)^* W_4(2)$ | 1 | 1 | -1 | 1 | -1 | -1 | 1 | -1 |
| $\Sigma_4$ | 2 | 0 | -2 | 2 | 0 | -2 | 0 | 0 |

*Table 15: Encoding and Superposition in Channel*

This composite signal Σ is received at the receiver. To recover each of the two messages, the signal is multiplied by the same Walsh cover as was used for encoding. The resulting signals are integrated using the summing revolver. In this example, the chips that are integrated together are chips of the same color. The result of the decoding process is shown in table 16.

| Decoding | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Chunk Size =1 Symbol** | | | | | | | |
| | *Chunk 1* | *Chunk 1* | *Chunk 2* | *Chunk 2* | *Chunk 3* | *Chunk 3* | *Chunk 4* | *Chunk 4* |
| **$\Sigma_1$* $W_1$(1)** | 2 | 0 | 0 | -2 | -2 | 0 | 2 | 0 |
| **$\Sigma_1$* $W_1$(2)** | 2 | 0 | 0 | 2 | -2 | 0 | 2 | 0 |
| **$M_1$'(1)** | 1 | | -1 | | -1 | | 1 | |
| **$M_1$'(2)** | 1 | | 1 | | -1 | | 1 | |
| | **Chunk Size =2 Symbols** | | | | | | | |
| | *Chunk 1* | | *Chunk 1* | | *Chunk 2* | | *Chunk 2* | |
| **$\Sigma_2$* $W_2$(1)** | 2 | 0 | 0 | -2 | -2 | 2 | 0 | 0 |
| **$\Sigma_2$* $W_2$(2)** | 2 | 0 | 0 | 2 | -2 | 2 | 0 | 0 |
| **$M_2$'(1)** | 1 | | -1 | | -1 | | 1 | |
| **$M_2$'(2)** | 1 | | 1 | | -1 | | 1 | |
| | **Chunk Size =4 Symbols** | | | | | | | |
| | *Chunk 1* | | | | *Chunk 1* | | | |
| **$\Sigma_4$* $W_4$(1)** | 2 | 0 | -2 | 2 | 0 | -2 | 0 | 0 |
| **$\Sigma_4$* $W_4$(2)** | 2 | 0 | -2 | 2 | 0 | 2 | 0 | 0 |
| **$M_4$'(1)** | 1 | | -1 | | -1 | | 1 | |
| **$M_4$'(2)** | 1 | | 1 | | -1 | | 1 | |

*Table 16: Decoding the Chunked DS-CDMA Signal*

This system has demonstrated the basic method of encoding and decoding the chunked DS-CDMA system. This general method can be extended to any chunk size, the only requirement is that the chunks are of equal size.

## 4.4 Possible Benefits of Chunking

### 4.4.1 Resistance to Small Synchronization Errors between Users

Because the Walsh code is applied over chunks rather than individual symbols, the Walsh cover used in DS-CDMA is much more slowly varying. This suggests that chunked Walsh covers should lose orthogonality to each other in a more gradual manner. This means that multiple access interference should be small for small shifts between users. Chunking may also offer some protection to symbols near the center of the chunks, as only the edges of the chunks should interfere in the presence of small shifts. If this is

true, then the addition of guard bands at the edges of the chunks should further decrease the bit error rate of the system.

### 4.4.2 Resistance to Burst Interference

Chunking the signal adds time diversity to the system, because it breaks the symbols up and spreads them throughout the signal. Burst interference therefore only results in the loss of a piece of several symbols rather than the complete loss of a few symbols. Loss of a small piece of a symbol should not be a problem, as it is only one of many pieces. Combined with forward error correction, the system should be strong against burst interference.

### 4.4.3 Flexibility

This method has a lot of flexibility built into it because of its ability to operate with many different chunk sizes. In the case with the 1 symbol chunks, the process is identical to that of a system with no interleaving. Other degrees of interleaving are obtained by simply varying the size of the chunks, and the rate at which the Walsh code is applied. Maximum interleaving is achieved by simply using a chunk size equal to the length of the signal. Decoding the signal can be done with a slightly modified correlation receiver.

# 5 Simulation Strategy

## 5.1 Assumptions

To evaluate the impact that data chunking has on a DS-CDMA system, a baseband DS-CDMA system was simulated in MATLAB. The primary goal of these simulations was to test the proposed system in a quasi-synchronous environment to evaluate its viability. In this simulation, only the Walsh code was used for spreading. Additional codes that may be used in a practical system were not considered. This is because the orthogonality between users in a synchronous system is due to spreading by the Walsh codes. Any spreading caused by other spreading codes should have no significant effect on the orthogonality between users.

For these simulations the channel was considered to be ideal, with no losses or multipath effects. The only non-ideality considered was random chip offsets between the transmitted messages. This is to simulate a system with random synchronization errors. For a maximum allowed shift $a$, each signal was allowed to shift by $[-a, -a + 1, \dots, 0, \dots, a - 1, a]$ chips with respect to a perfect system with no shift. The distribution of shifts was assumed to be uniform, with $a$ assumed to be an integer. These simulations will test the effect of increasing the maximum allowed shift on the error distribution at the receiver versus the chunk size of the data. It is assumed in these simulations that these relative shifts are random, but that the receiver knows what they are. This is a reasonable assumption because in real CDMA systems the timing information is encoded within the signals to allow the receiver to properly synchronize the signals.

## 5.2 Simulation Model

Under these assumptions, a basic DS-CDMA system was created in MATLAB to simulate the effect of chunking on a system in which the users are not perfectly synchronized. The block diagram of the system is shown in figure 25.

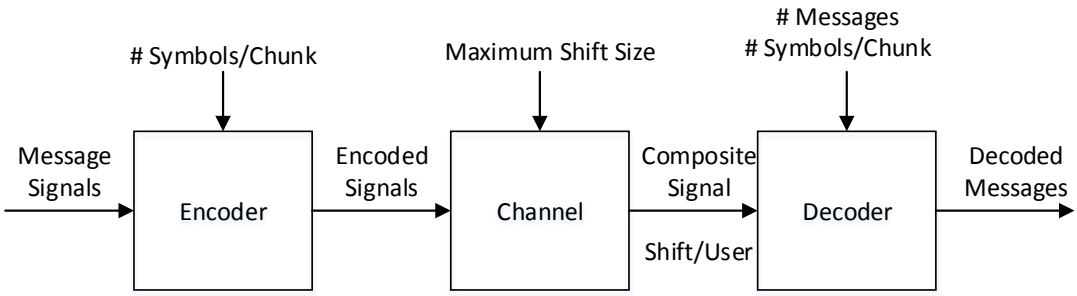


*Figure 25: System Block Diagram*

The encoder takes a set of binary finite length message signals represented by -1's and 1's, and encodes them in chunks using the Walsh code. The number of messages must be a power of 2, due to the number of Walsh codes being a power of 2. There is no restriction on the length of the messages. The number of symbols per chunk must be a factor of the length of the message signals. The output of the encoder is a set of chunked DS-CDMA encoded messages.

The channel takes these chunked DS-CDMA encoded messages from the encoder, and randomly circularly shifts them within the range specified by maximum shift size. It then sums all of the shifted encoded messages to create a composite signal. All of the random shift values and the composite signal are then output to the decoder.

The decoder receives the composite signal and shift information, along with the number of symbols per chunk and number of messages. Using this information, it

decodes each message. This is done by shifting the composite signal to undo any shifting done in the channel, then multiplying it by the appropriate Walsh sequence. Each signal is then integrated using the summing revolver to decode the original messages.

## 5.3 General Practices

All simulations used a set of 64 users to match the IS-95 standard. The length of the message blocks was arbitrarily chosen to be 64 symbols. Because of the highly divisible nature of 64, this allowed for many different chunk sizes to be tested. The message signals were random set of -1's and 1's, with an equal probability of each. When finding the bit error rate, simulations generally followed the rule that a symbol $\geq 0$ is a 1, while a symbol $< 0$ is a -1. Simulations were generally repeated many times to gather representative data.

# 6 Simulation Results

Using the simulation model, many simulations were performed to determine the characteristics of the system in a quasi-synchronous environment.

## 6.1 Walsh Cover Cross Correlation

The cross correlations between all combinations of Walsh covers in a 64 user system were found for several different chunk sizes. The magnitude of these cross correlations were averaged together and normalized resulting in figure 26.

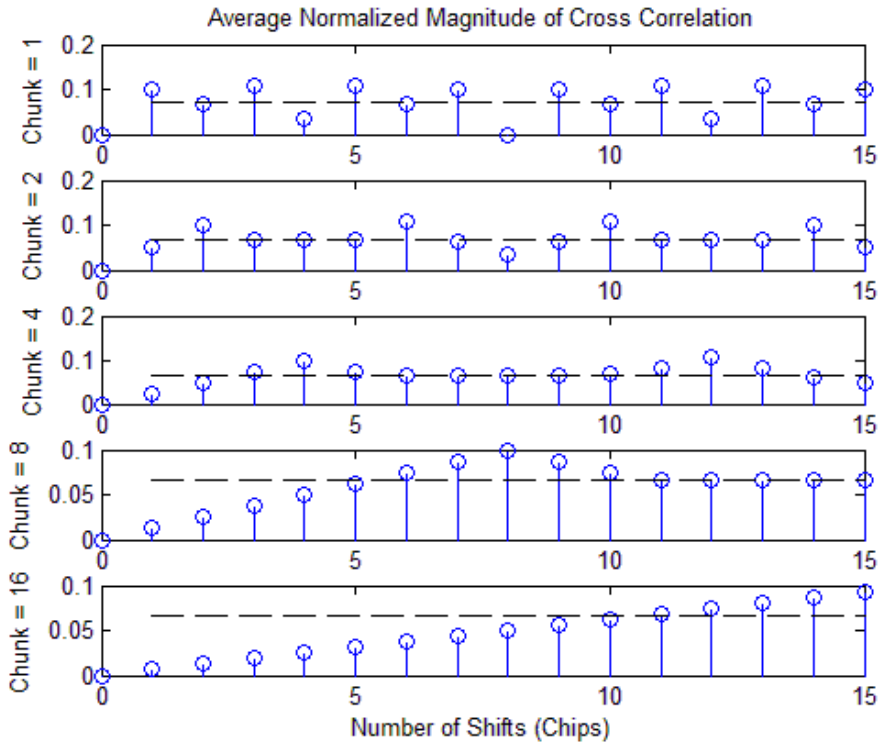


*Figure 26: Cyclic Cross Correlation between Chunked Walsh Codes*

From this plot, we can clearly see that the Walsh cover used in a system with no chunking has poor cross correlational characteristics when there is any time offset between users. Chunked Walsh covers clearly have superior cross correlations at small

delays. As the chunk size increases, the rate at which the Walsh covers lose orthogonality decreases. This promises manageable amounts of multiple access interference, given that the chunk size is large enough to accommodate the magnitude of the synchronization errors.

## 6.2 Spreading Properties

### 6.2.1 Spectrum of Non-Chunked System

To characterize the spreading properties of the system, the spectral characteristics of the encoding process were found using the FFT (Fast Fourier Transform). First, the system was run with no chunking, chunk size = 1 symbol. Without chunking, the message is not compressed and repeated. The spectrum of a typical pseudorandom message is shown in figure 27.

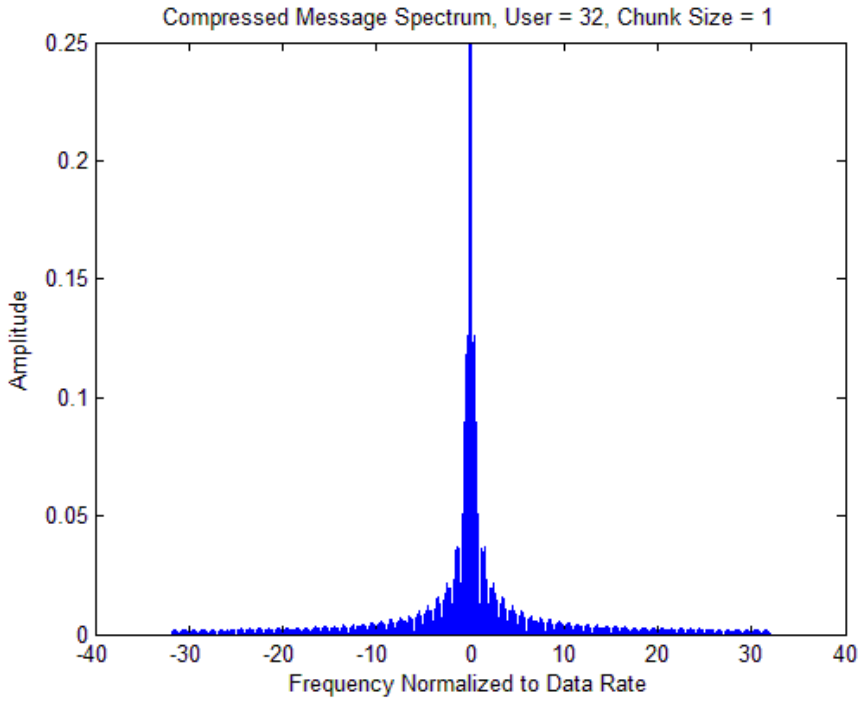


*Figure 27: Compressed Message Spectrum*

This signal is then spread by an arbitrary Walsh cover. In this case, the Walsh cover is applied at 64 times the bit rate of the message. To accomplish this, the Walsh cover must be repeated for every symbol. This makes the Walsh cover highly periodic, resulting in spreading of the message signal to only a few discrete locations. The resulting encoded message spectrum and the spectrum of the Walsh cover are shown in figure 28.

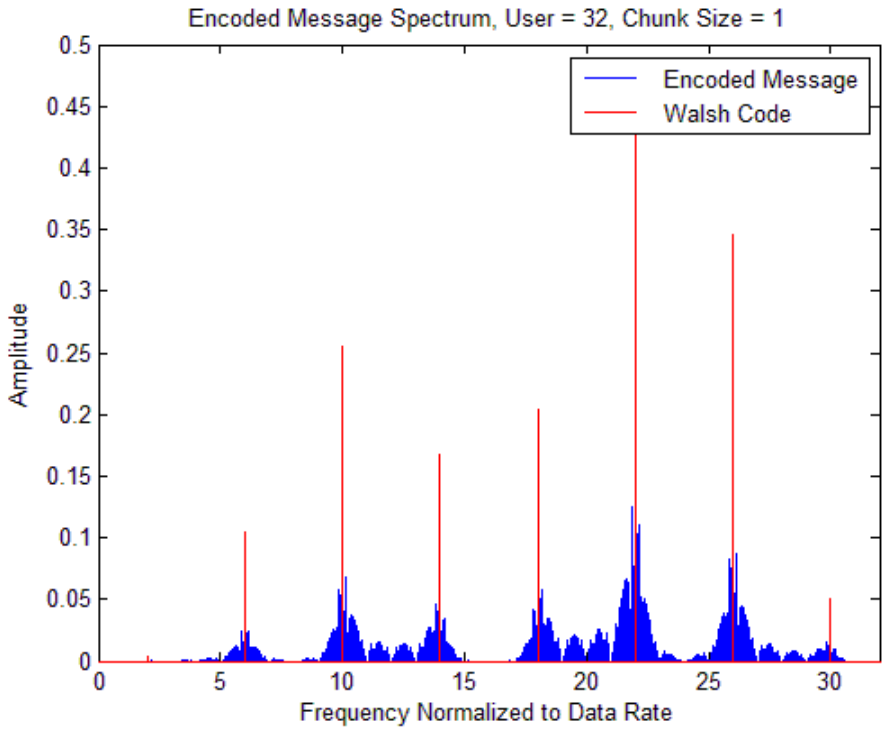


*Figure 28: Encoded Message Spectrum vs. Walsh Code Spectrum*

This case would be considered poorly spread, because the signal is concentrated at specific frequencies rather than being uniformly distributed across the bandwidth. This makes the non-chunked system vulnerable to frequency selective interference.

### 6.2.2 Spectrum of a Chunked System

With a large chunk size, the Walsh cover is applied at a much lower data rate and the signal itself is compressed and repeated at a much higher data rate. Since the message

is a much higher data rate than the Walsh cover, it can be viewed as the Walsh cover being spread by the message. Figure 29 shows the spectrum of an arbitrary Walsh cover in this case.

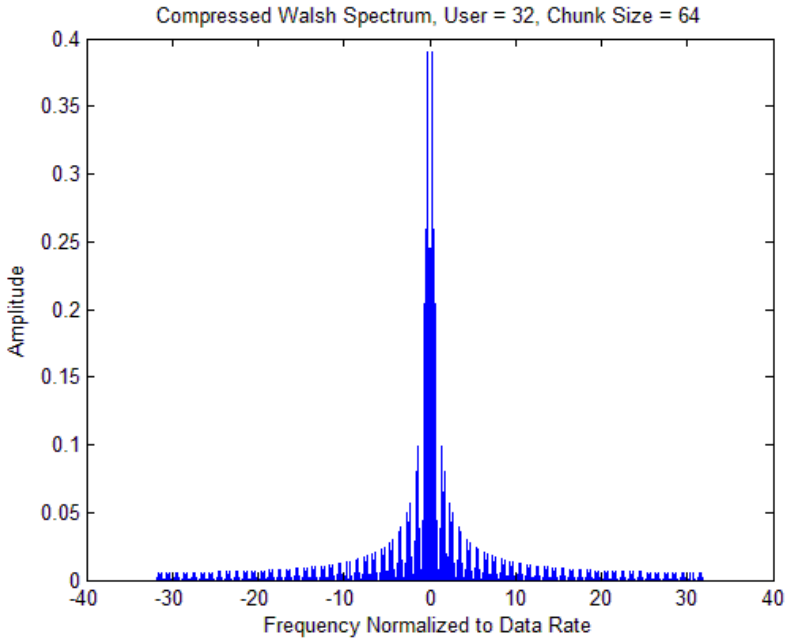


*Figure 29: Walsh Code Spectrum*

This signal is spread by the compressed and repeated message, as shown in figure 30.

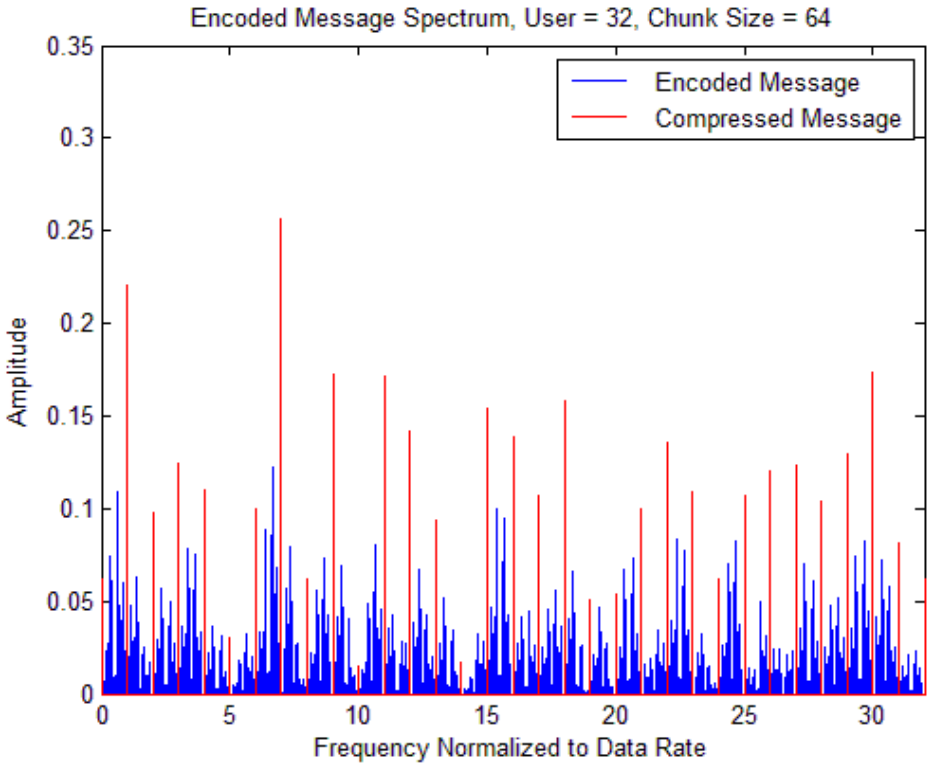


*Figure 30: Chunked Encoded Message Spectrum*

The message is still repeated, making it appear as a periodic message. However, the message is much longer and much more random than the Walsh code, so it has much more uniform frequency spectrum. The result is that the Walsh code is spread to many discrete locations. This results in a much more uniformly spread spectrum. This system is much more frequency diverse than the system with no chunking, meaning that it should be more resilient to all forms of frequency selective interference.

## 6.3 Simulations of Complete System

### 6.3.1 Error Distributions

Using the developed MATLAB simulation, the error distribution of the system with various chunk sizes and a random shift magnitude of 1 chip was found. This means that each message within the set can shift by either -1,0,1 chips. These simulations were performed with 64 users, transmitting messages in blocks 64 symbols long. Each simulation was run 100 times to collect more data points. Figure 31 plots the received error magnitude distribution that resulted from this simulation using the raw values of the received data. The error is simply the difference between the decoded symbol and the encoded symbol.

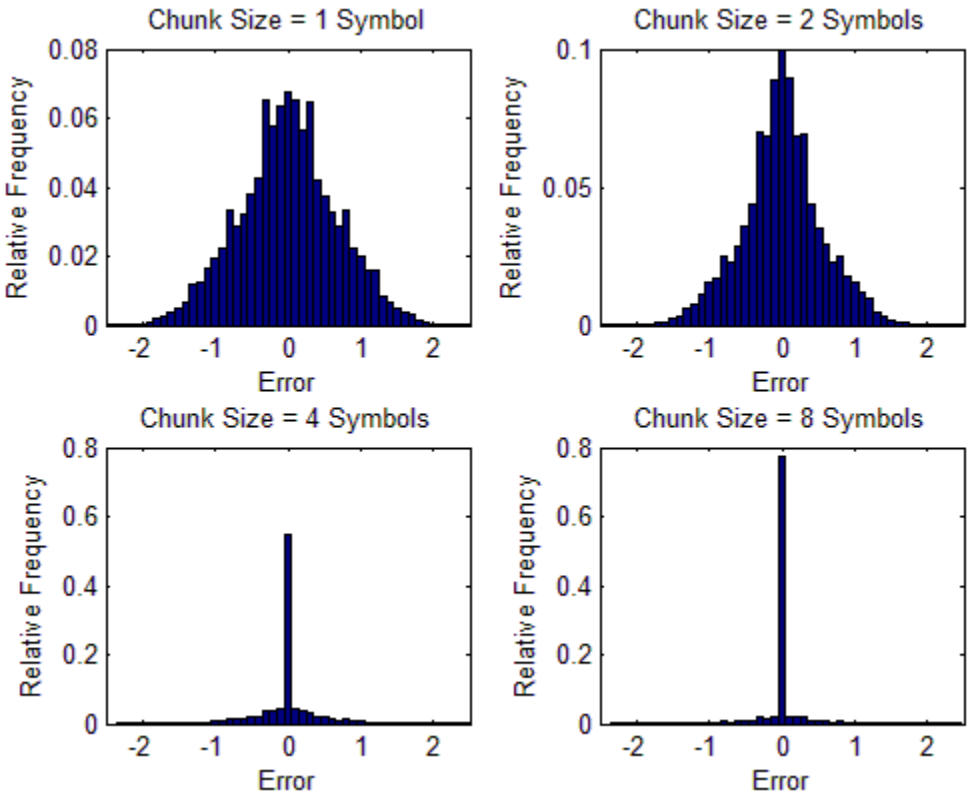


*Figure 31: Error Distributions for Varying Chunk Sizes*

These plots appear to follow an approximately Gaussian distribution with a mean of roughly zero. As the chunk size increases, we can clearly see that the variance of the distribution greatly decreases. This suggests that the bit error rate must greatly improve as the chunk size increases. To compare the properties of these simulations the mean and variance of the error magnitudes vs. chunk size were plotted in figures 32 and 33.

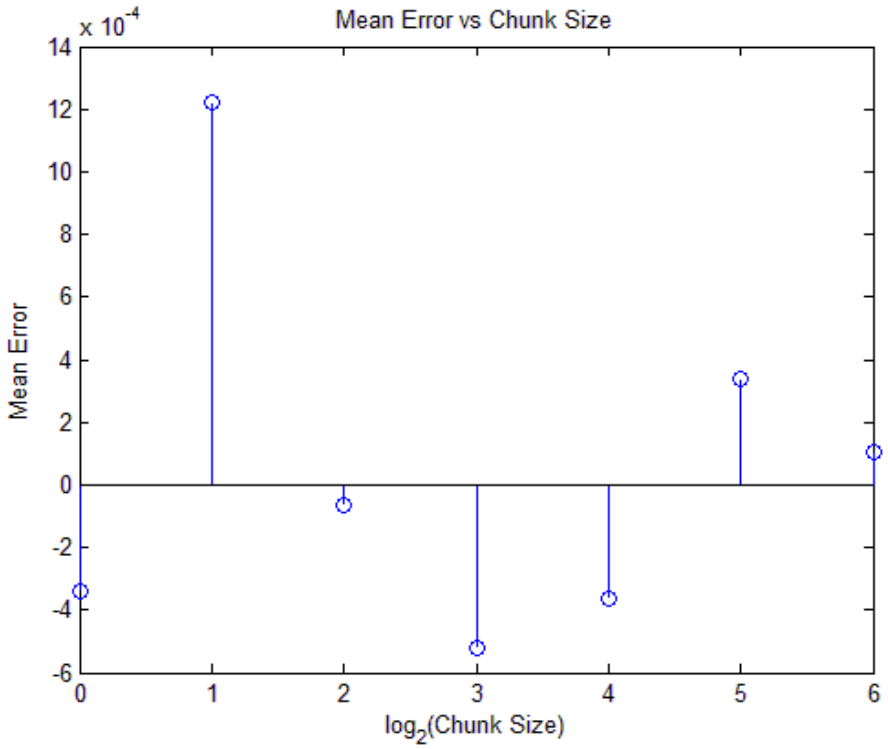


*Figure 32: Mean Error vs. Chunk Size*

This plot confirms that the mean error is small for all of the simulated cases. There does not appear to be a relationship between the chunk size and mean error. This suggests that this chunking does not introduce any source of systematic error that skews the results in either direction.

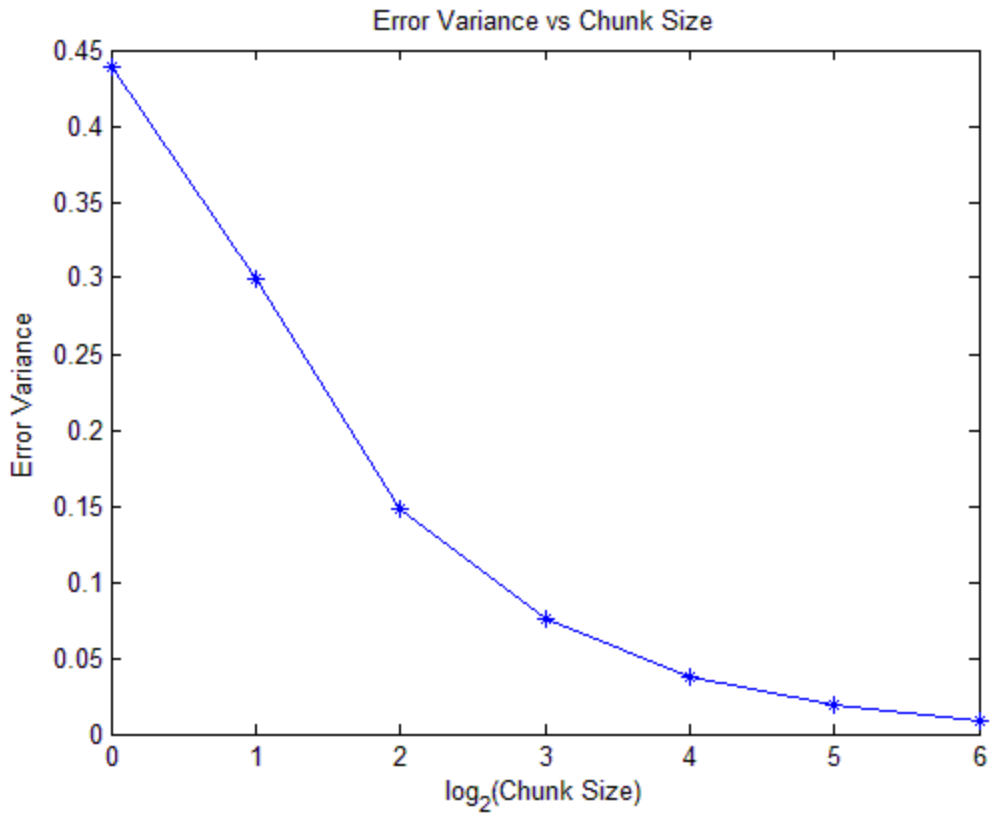


*Figure 33: Error Variance vs. Chunk Size*

The error variance clearly decreases rapidly as the chunk size increases as was observed in the previous plots. With the exception of the first point, each doubling of the chunk size appears to halve the error variance. A decreasing error variance, with a constant mean suggests that chunking the data should improve the bit error rate.

To confirm this, the bit error rate was calculated for the same system, figure 34. To calculate an approximate bit error rate, received symbols greater than or equal to 0 were assigned a value of "1", and received symbols less than 0 were assigned a value of "-1".

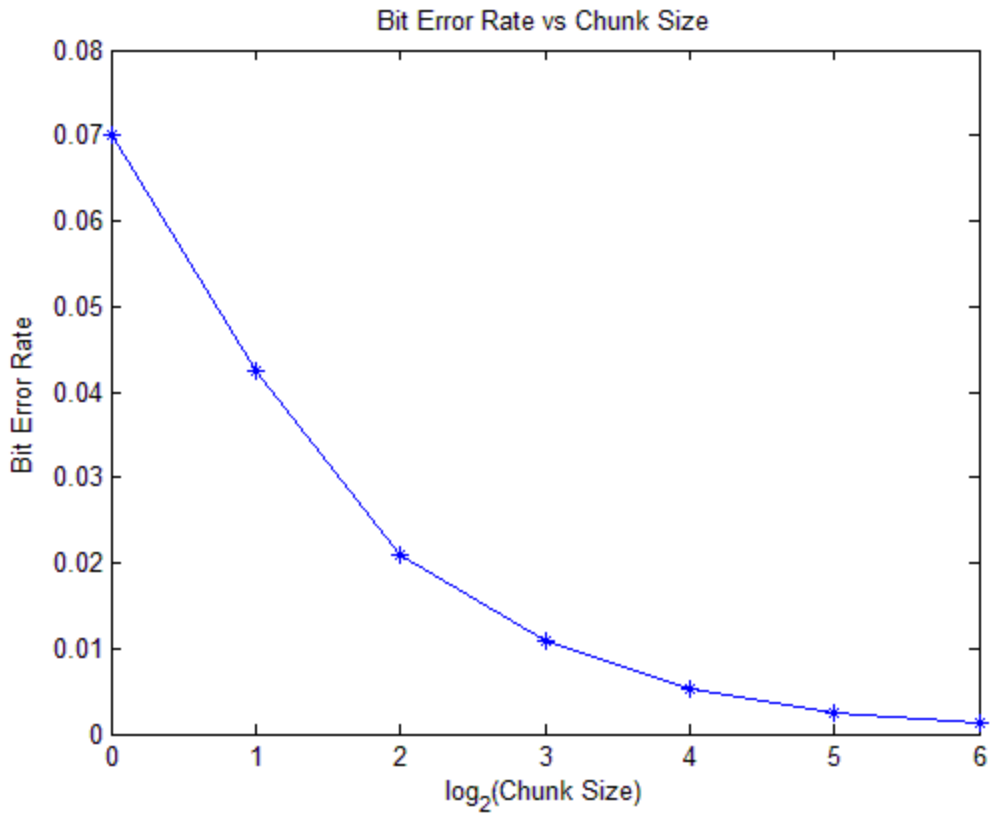


*Figure 34: Bit error rate vs. Chunk Size*

This plot shows that the bit error rate does decrease as the chunk size increases, as was predicted previously. Here we can see that the bit error rate drops from about 7.2% in a non-chunked system down to about 0.1% in a system with a chunk size of 64. We can see a trend that the error rate roughly halves every time the chunk size doubles. These results suggest that using a large chunk size greatly reduces the effect of synchronization errors.

### 6.3.2 Effect of increasing Shift Magnitude

To further explore the effect of synchronization errors, the same simulations were performed. However, this time the size of the shift was allowed to be greater than 1 chip. In this simulation, each user transmitted with a random shift ranging from –a to a, where a is the maximum shift size in chips. The maximum shift was varied from 0 to 64 chips,

for many different signals with different chunk sizes. The error variance of these received distributions was plotted versus the maximum shift size in figure 35.

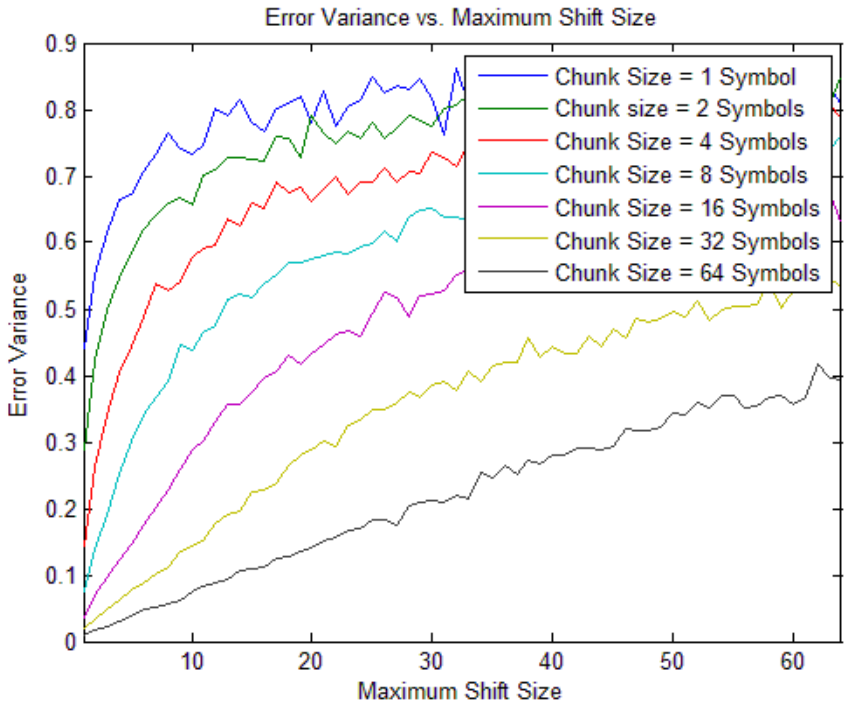


*Figure 35: Error Variance vs. Maximum Shift Size*

The error variance is clearly related to the chunk size, with the larger chunk sizes having lower error variance for any maximum allowed shift size. This suggests a lower corresponding bit error rate for all maximum shifts for these chunked signals.

To confirm this, the bit error rate was plotted under the same conditions in figure 36.

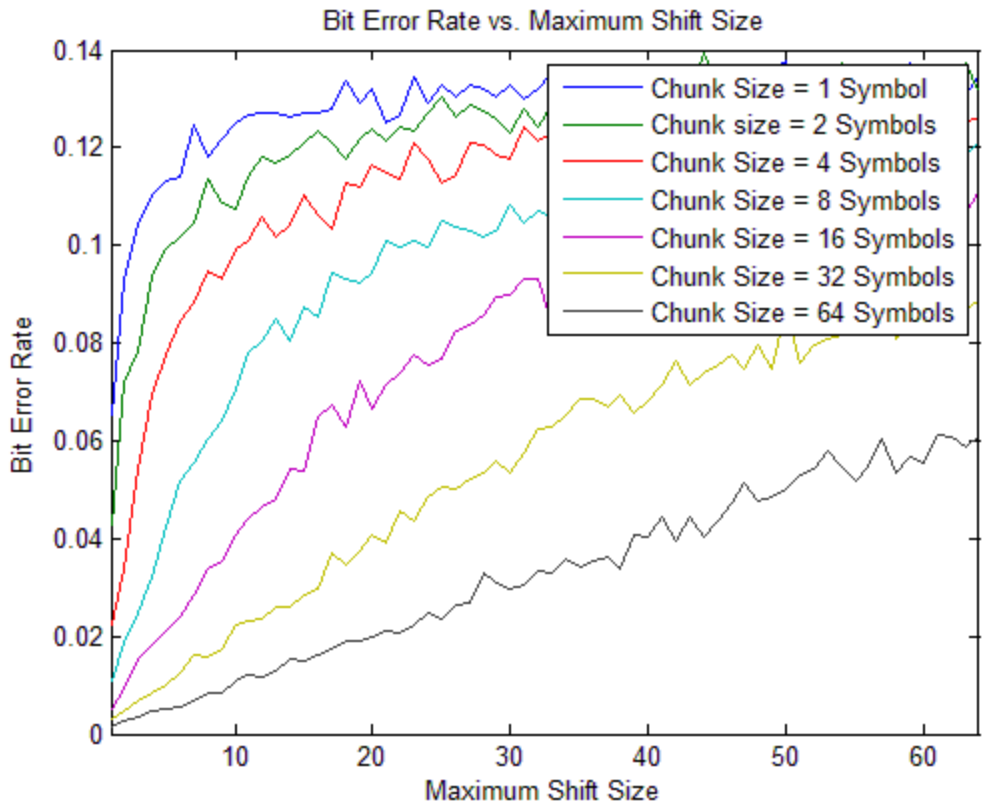


*Figure 36: Bit error rate vs. Maximum Shift Size*

As expected, the system with the largest chunk size had the lowest bit error rate for all shift sizes. It also appears as though the system with no interleaving had the worst performance. From the data, it would appear that there is a linear trend between the bit error rate and the maximum shift size. This linear trend only continues until the bit error rate is approximately 13%, after which the bit error rate appears to remain constant. As the chunk size decreases the slope of the curve appears to increase, with the system with no chunking having the steepest slope.

### 6.3.3 Error Magnitude vs. Symbol Position

Since it was clear that the chunked systems appear to have reduced sensitivity to small synchronization errors, we would like to investigate the location of these errors to

determine if there are any patterns. Any patterns in the distributions may suggest the possibility of adding guard bands. To determine this, the simulations were performed again, and each symbol was evaluated individually. This was repeated for several different maximum shifts and chunk sizes.

With the shift confined to a maximum of 1 chip, the error magnitude is plotted for each symbol position from a single run in in figure 37.

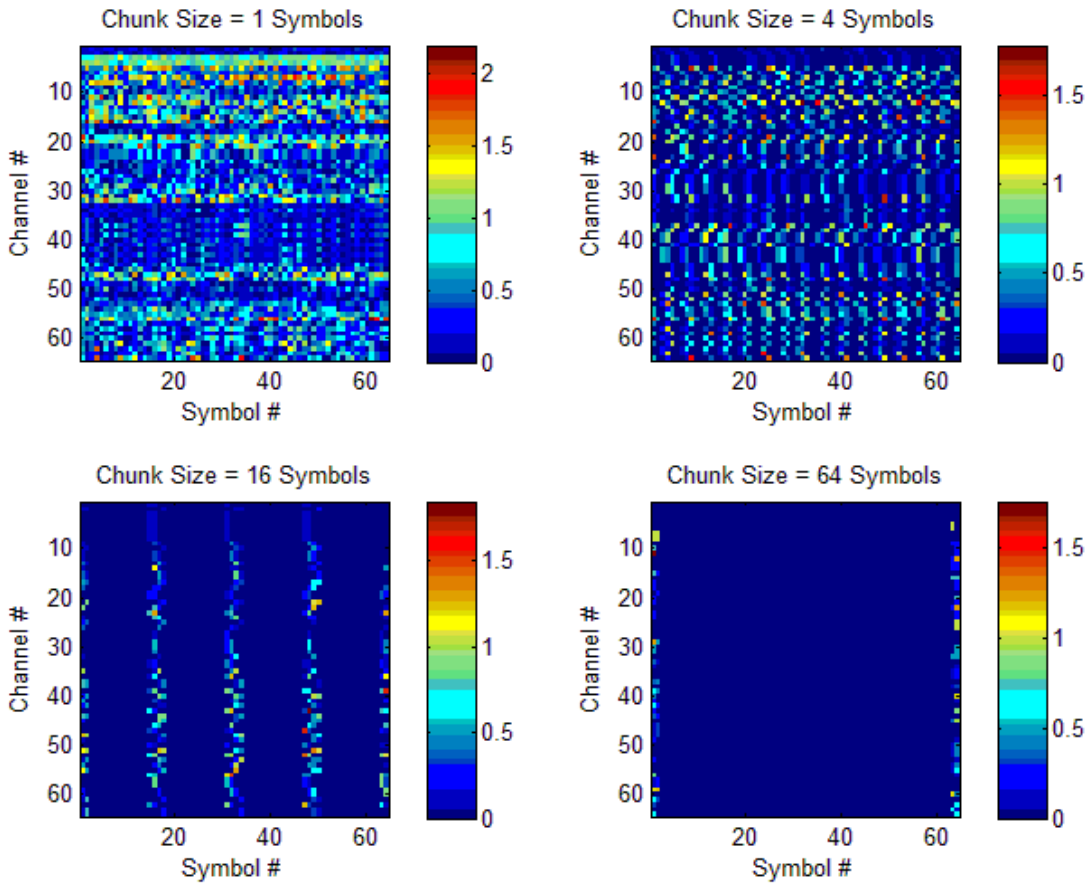


*Figure 37: Error Magnitude by Position for Maximum Allowed Shift = 1 Chip*

With a chunk size of 1 symbol, there is no clear pattern in magnitude of the errors. However, for larger chunks there does appear to be a pattern. The 4 symbol chunked

signals have a slight pattern, while the 16 and 64 symbol chunks clearly have clusters of high error bits. These clusters are equally spaced, and only appear to be 4 symbols wide. With the exception of a few channels, all channels appear to have roughly the same error distribution.

To determine whether this pattern holds true in systems that allow for greater synchronization errors, the simulation was repeated. However, this time the signals were allowed to shift by up to 2 chips. The results of this simulation are shown in figure 38.

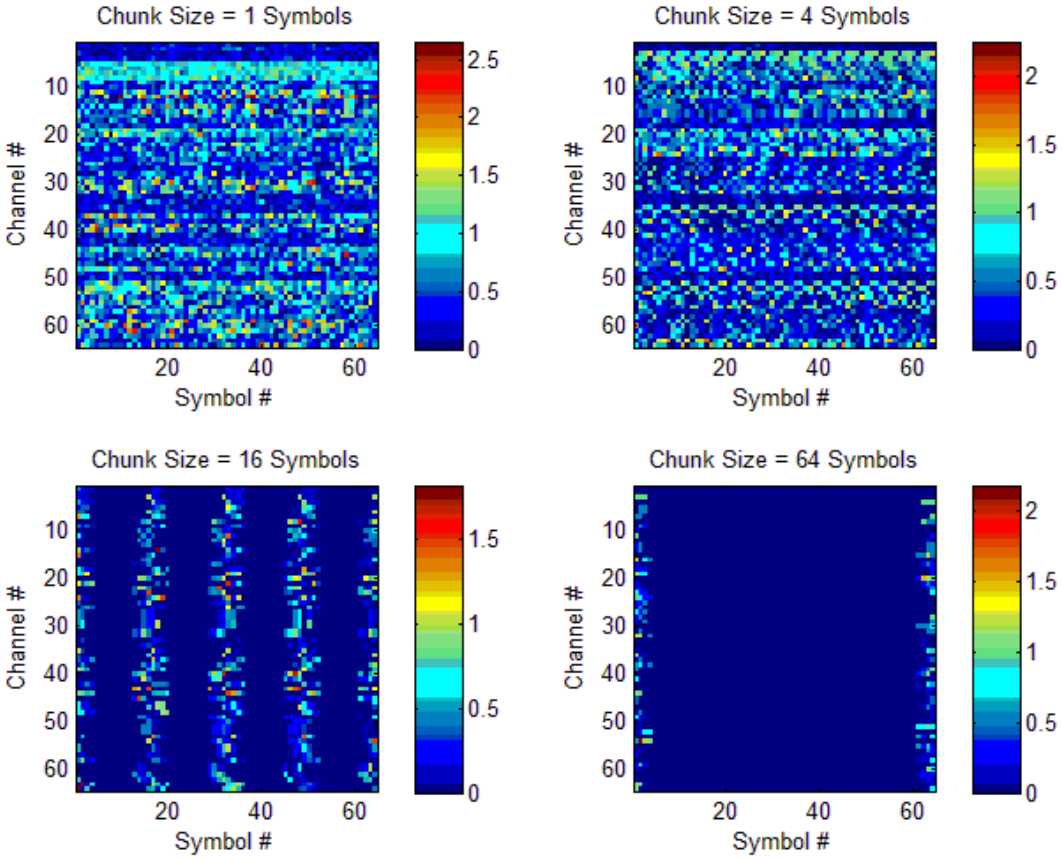


*Figure 38: Error Magnitude by Position for Maximum Allowed Shift = 2 Chips*

These plots show the same general pattern as the previous figure. However, the clusters of high error symbols have clearly grown larger. All patterns in the 4 symbol chunk plot appear to have disappeared, as the clusters overlapped each other.

### 6.3.4 Bit Error Rate vs. Symbol Number

Because the errors seem be dependent entirely on the symbol number and not the channel number, the simulation was run again. This time, the bit error rate for each symbol position was calculated. This was done by running the simulation 100 times, and considering all of the errors for each symbol position without consideration for the channel number. Because there are 64 channels, that means that each bit error rate had a total of 64*100=6400 bits from which the calculations were performed. The results of this simulation for several different maximum shifts and chunk sizes are shown in figures 39-42.

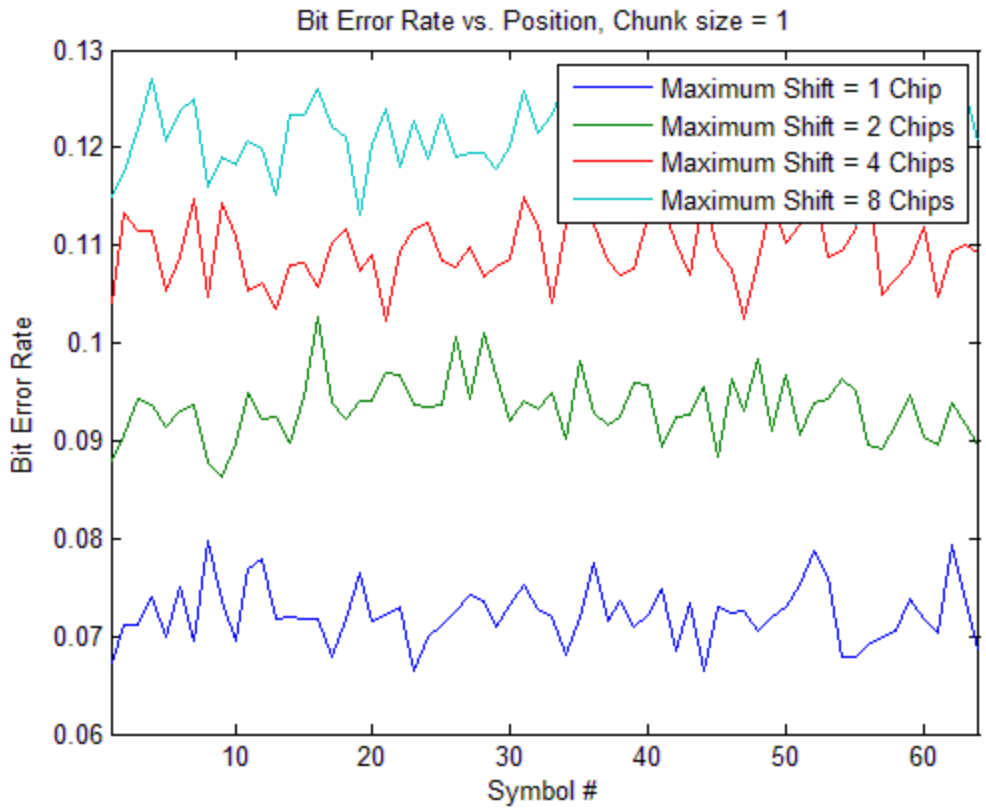


*Figure 39: Bit Error Rate vs. Position, Chunk Size = 1 Symbol*

For the system with no chunking, the bit error rate is constant across the messages and increases as the maximum shift is increased. In this case, the bit error rate is relatively high for all of the shifts. This suggests that the non-chunked system would behave poorly in a quasi-synchronous environment. The fact that the bit error rate is constant across the entire set of messages means that guard bands would be of no use in this case.

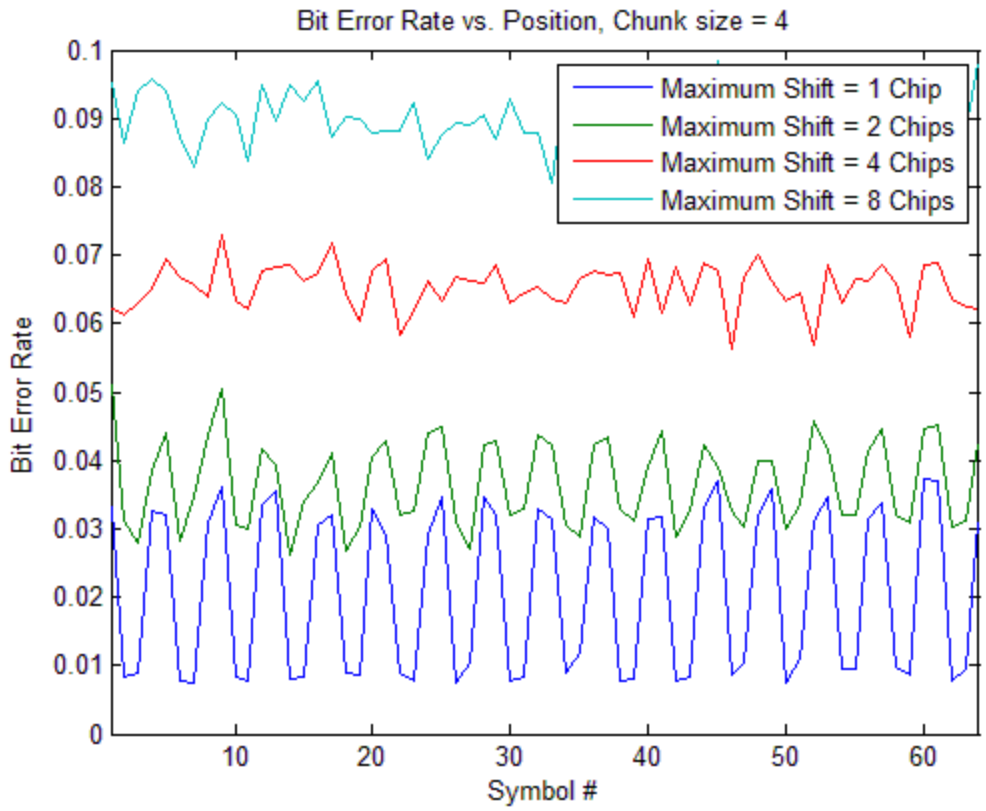


*Figure 40: Bit Error Rate vs. Position, Chunk Size = 4 Symbols*

With a chunk size of 4 symbols, the bit error rate is clearly much lower for all of the maximum shifts than the non-chunked case. The bit error rate still appears to be uniform across the messages. However, for the lower allowed maximum shifts, there is a new pattern emerging. For the maximum shift of 1 and 2 chips, the bit error rate is clearly dependent on the symbol position. For the maximum shift equal to 1 chip for example, it appears as the bit error rate alternates between 1% and 3%.

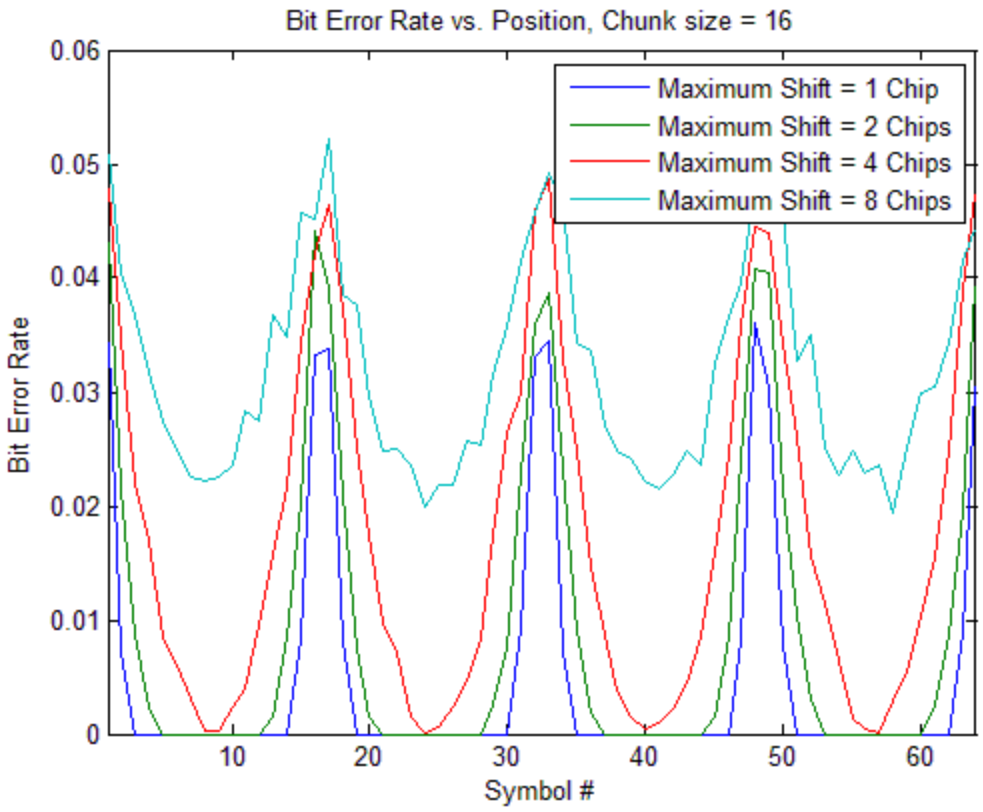


*Figure 41: Received Bit Error Rate vs. Position, Chunk Size = 16 Symbols*

With a chunk size of 16, this figure is clearly demonstrating a strong pattern for all of the maximum shift sizes. There are even some locations where the bit error rate is equal to zero. For all of the maximum shift sizes, there is a peak in the bit error rate at 4 locations located 16 symbols apart from each other. The width and maximum bit error rate of each peak appear to increase with the maximum chip sizes.

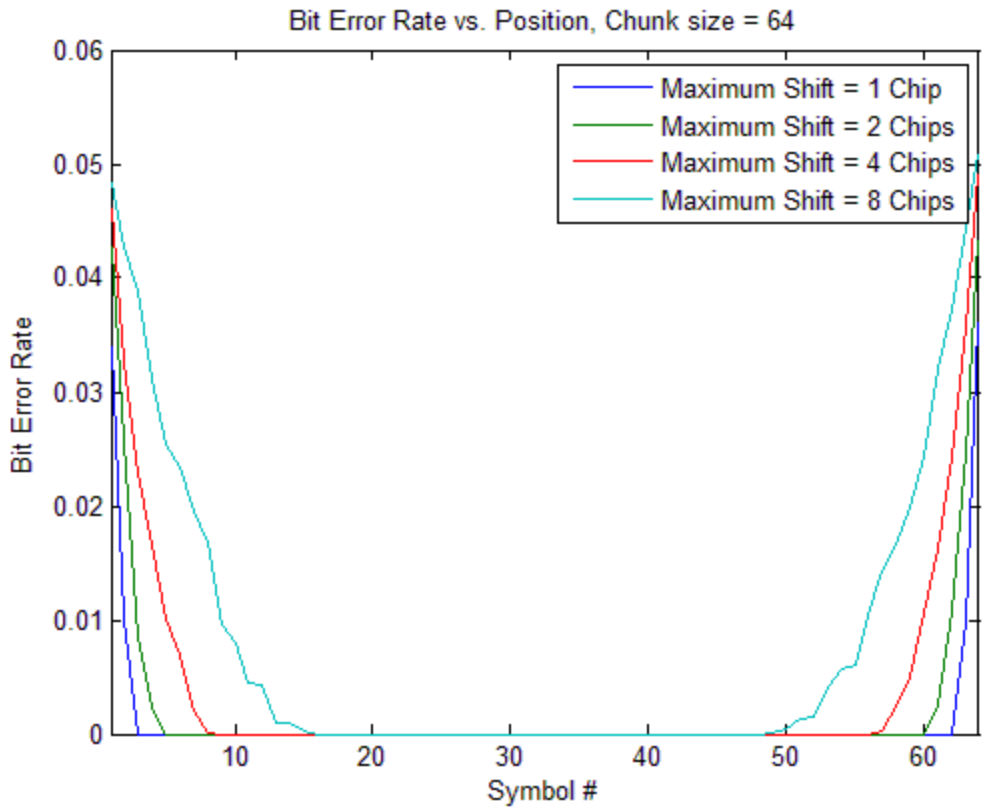


*Figure 42: Bit Error Rate vs. Position, Chunk Size = 64*

With a chunk size of 64 symbols, most of the signal is completely free of errors for all of these maximum shift sizes. Only the edges of the messages appear to be vulnerable to error, and these error rates are relatively small, about 4.5%.

### 6.3.5 Implementation of Guard Bands

Since we can clearly identify the locations of the errors based upon the chunk size and the maximum shift, we can quarantine certain symbol positions to create "guard bands" to eliminate all error due to synchronization. That is, we can place redundant information in the symbol positions that are prone to error. These symbols can be checked at the receiver as part of forward error correction, but would not be relied upon for actual data.

The plots from the previous two sections show that the errors only occur at the edges of the chunks. From the previous plots, it is established that these vulnerable edges are 2*(Max Shift) symbols wide. The factor of 2 is because each signal can shift left or right by up to the maximum shift. Because each chunk has two edges, the total number of vulnerable symbols is equal to 4*(Max Shift) symbols per chunk. The fraction of symbols that must be quarantined vs. the chunk size is plotted in figure 42.

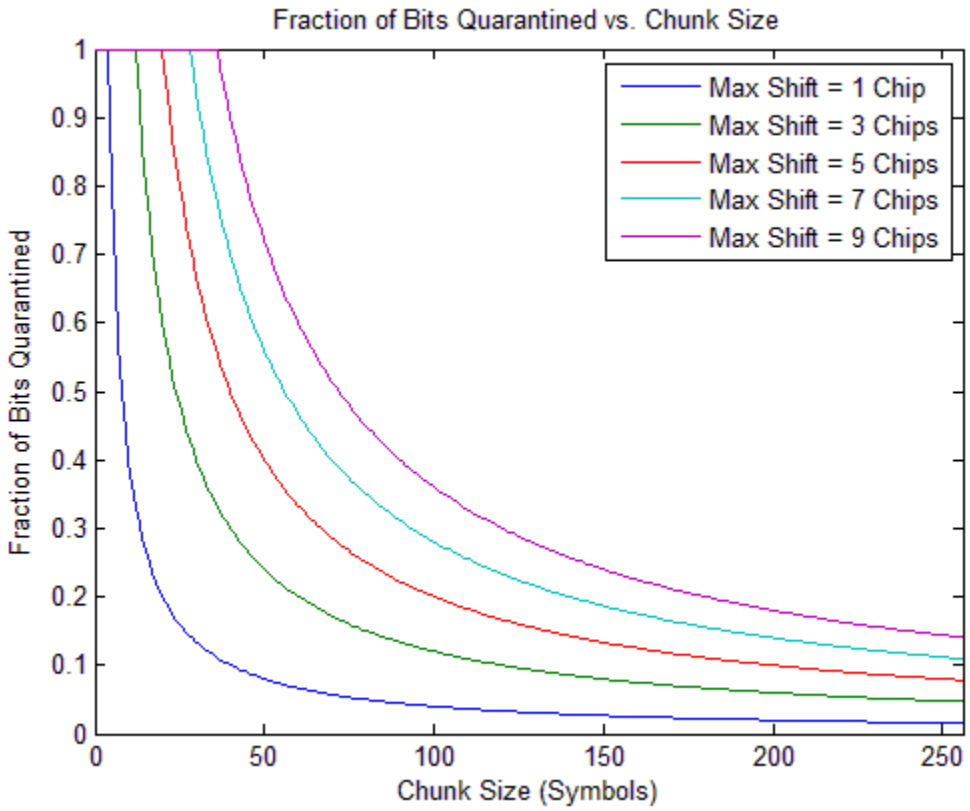


*Figure 43: Fraction of Quarantined Symbols vs. Chunk Size*

This plot demonstrates that creating guard bands costs a lot of symbols in systems with small chunk sizes, or large shift uncertainty. For best results, the largest chunk size should be used. However, even relatively large chunks have significant data loss to guard bands when the maximum allowed shift is more than a few chips.

# 7 Conclusion

## 7.1 Summary of Results

We have shown that there is a fundamental link between TDMA and synchronized DS-CDMA, and the development of the Walsh codes used in this technique. This link has been exploited to develop a system that takes advantage of a TDMA solution to burst error in a synchronous DS-CDMA system. In so doing we have created a system that is robust to burst error, but does not face the latency problems that a similar TDMA system would face. This new type of system applies synchronous DS-CDMA techniques over variable sized chunks of data to create a sort of interleaving. This new system is flexible, and should be relatively easy to implement.

Possible implementations of the transmitter and receiver for this type of system were explored to demonstrate the relative simplicity of implementing this type of system. Both the transmitter and receiver only require a few changes to allow the full range of chunking to be possible. This type of system is more general than a synchronized DS-CDMA system with no chunking, as it can easily switch the size of the chunks, even allowing it to transmit without chunking the data. This allows for the possibility of varying the chunk size depending on the channel conditions.

This system was shown to result in a form of chip interleaving, a popular method to increase the time diversity of a signal. This type of interleaving has many benefits such as increasing the resistance of the signal to burst error, multipath, and synchronization errors. The main downside of any system that uses chip interleaving is increased complexity of the transmitter and receiver, as it requires an interleaver and de-interleaver. However, this method that we have developed is likely less complex than a conventional

chip interleaved system as it does not require these two components. It simply modifies the way in which the Walsh code and message signal is applied to the system.

To investigate possible benefits of this type of system, a baseband system in which users were not perfectly synchronized was simulated in MATLAB. The simulations showed that the bit error rate drops linearly with the chunk size for any given maximum shift. This result was confirmed to be true for larger shifts. This result suggests that the best performance of this system in the face of synchronization errors is with the largest possible chunk size. However, even with a large chunk size this system is overwhelmed by errors larger than about half a symbol period.

The final simulations showed that the locations of errors due to errors in synchronization are completely predictable. The errors always occur on the edges of the chunks and the size of the sections that can have errors is simply equal to twice the maximum shift size. From this information, it is clear that we can create guard bands that would completely eliminate all synchronization errors for a given worst case synchronization. This can be done by simply placing extra redundant bits in the vulnerable symbol positions so that none of the actual message is corrupted by synchronization errors. We have therefore shown that this system may be suitable for a quasi-synchronous reverse link.

This system has numerous other advantages due its chip interleaving nature. Other papers have considered the effect of chip interleaving with respect to multipath fading [18] and other phenomena. With regard to these, there is generally a significant reduction in errors when using a DS-CDMA system that incorporates chip interleaving. The only real drawback to this type of a system is that it requires some additional transmitter and

receiver complexity. However, compared to other interleaved DS-CDMA systems this system is of similar complexity.

## 7.2 Future Work

Further simulations can be performed to investigate other effects on the chunked system. Such simulations could investigate the performance of the system in the presence of multipath, and other channel effects including burst errors. The system could also be tested in the face of different methods of jamming and frequency selective interference, or unequal power levels of received signals. The system could be designed and tested in hardware using the components described.

# References


[1] J. Hoebeke, I. Moerman, B. Dhoedt, and P. Demeester. “An overview of mobile ad hoc networks: applications and challenges”, In Proceedings of the 43rd European Telecommunications Congress, FITCE2004, Ghent, Belgium, November 2004.

[2] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018,” white paper, Cisco, Feb. 2014.

[3] T. Farley, “Mobile Telephone History”, Telektronikk, April 2005.

[4] D. D. Muzic, “Capabilities and impacts of edge evolution toward seamless wireless networks,” in MIPRO 2009, 2009.

[5] Q. Bi, R. Brown, D. Cui, A. Gandhi, C.-Y. Huang, and S. Vitebsky, “Performance of 1EV-DO third generation wireless high speed data system,” Bell Labs J., vol. 7, no. 3, pp. 97–107, Jan. 2003.

[6] B. McWilliams, Y. Le Pezennec and G. Connins, "HSPA+ (2100 MHz) vs LTE (2600 MHz) spectral efficiency and latency comparison," in XVth International Telecommunications Network Strategy and Planning Symposium (NETWORKS), 2012.

[7] S. Hossain, “5G wireless communication systems” American Journal of Engineering Research (AJER) e-ISSN: 2320-0847 p-ISSN: 2320-0936 Volume-02, Issue-10, pp-344-353.

[8] K. Tigga, S. Shinghai, "A review of innovative techniques in MC-CDMA for future 4G." *International Journal of Engineering*, vol. 5, no. 11, Nov. 2013.

[9] I. Shakya, F. Ali, and E. Stipidis, “High user capacity collaborative CDMA,” IET Commun., vol. 5, no. 3, pp. 307-319, Feb. 2011.

[10] D. M. Pozar, *Microwave and RF Design of Wireless Systems*. New York: Wiley, 2001.

[11] C. Marvin, *When Old Technologies Were New*. New York : Oxford University Press, 1988.

[12] 3rd Generation Partnership Project, “UTRA-UTRAN Long Term Evolution (LTE) and 3GPP System Architecture Evolution (SAE)”, 2006.

[13] A. Mehrotra, *GSM System Engineering*. Boston, MA: Artech House, 1996.

[14] K. Raith and J. Uddenfeldt, “Capacity of digital cellular TDMA systems,” IEEE Trans. Veh. Technol., vol. 40, pp. 323-332, May 1991.

[15] A. Dudkov, "Chip and signature interleaving in DS CDMA Systems.", Ph.D thesis, Tampere University of Technology, 2009.

[16] A. J. Viterbi, *CDMA: Principles of Spread Spectrum Communication.* Reading, MA: Addison-Wesley, 1995.

[17] B. Sklar, *Digital Communications Fundamentals and Applications*, Prentice Hall PTR, 2001.

[18] L. Lim, *Chip interleaving for CDMA cellular systems*. University of Toronto, 1999.

[19] R. L. Pickholtz, D. L. Schilling, and L. B. Millstein, “Theory of spread spectrum communications—a tutorial,” IEEE Trans. Commun., vol. COMM-30, pp. 855–884, 1982.

[20] K. Choi and T. Han, “Orthogonal spreading code for quasi-synchronous CDMA based on scrambled Walsh sequence,” in Proc. 2006 IEEE Globecom, pp. 1–4.

[21] K. S. Gilhousen, I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, Jr., and C. E. Wheatley, III, “On the capacity of a cellular CDMA system,” IEEE Trans. Veh. Technol., vol. 40, pp. 303–312, May 1991.

[22] Muxiang Zhang, Christopher Carroll, and Agnes Hui Chan., “Analysis of IS-95 CDMA voice privacy”. In Selected Areas in Cryptography, pages 1–13, 2000.

[23] A. J. Viterbi, A. M. Viterbi, K. S. Gilhousen, and E. Zehavi, “Soft handoff extends CDMA cell coverage and increases reverse link capacity,” IEEEJ. Select. Areas Commun., vol. 12, pp. 1281-1288, Oct. 1994.

[24] A. Duel-Hallen, J. Holtzman, and Z. Zvonar, "Multi-User Detection for CDMA Systems," /E€€ Pers. Commun., vol. 2, no. 2, pp. 46-58, Apr. 1995.

[25] G. Aliftiras, “Receiver implementation for a CDMA cellular system”, Ph.D thesis, Virginia Polytechnic Institute and State University, July 1996.

[26] H. Holma, A. Toskala, *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications,* John Wiley and Sons, 2004

[27] A. Ambardar, *Analog and Digital Signal Processing*. Pacific Grove, California: Brooks-Cole, 1999. Print.

[28] R. C. Dixon, *Spread Spectrum Systems with Commercial Applications*, Wiley, 1994.

[29] M. Pal, S. Chattopadhyay, “A novel orthogonal minimum cross-correlation spreading code in CDMA system”, in Proc. Emerging Trends in Robotics and Communication Technologies (INTERACT), Chennai, 2010.

[30] E. Hardouin and C. Laot. "A chip-interleaving pattern retaining orthogonality in DS-CDMA Systems: application to the multicode downlink". in Proc. Vehicular Technology Conference (VTC), Florida. USA. vol. 4. pp. 2401- 2405. Oct. 2003.

[31] J.L. Shanks, “Computation of the fast Walsh-Fourier transform,” IEEE Trans. Computers, vol. 18, pp. 457-459, 1969.

[32] D. P. Whipple, "The CDMA Standard," Applied Microwave & Wireless, pp. 24-37, Spring 1994.

[33] J. Abouei, “A set of cyclic orthogonal codes acquired from Walsh-Hadamard matrix.” Presented at the 34th Int. Math. Conf., Shahrood, Iran, Sep. 2003.

[34] C. Lin, C. Shieh, N. Chilamkurti, C. Ke, W. Hwang, “A RED-FEC mechanism for video transmission over WLAN”, IEEE Transactions on Broadcasting, vol. 54 no. 3, pp. 517–524, Sep. 2008.

[35] L.-J. Chen, T. Sun, M. Y. Sanadidi, and M. Gerla, “Improving wireless link throughput via interleaved FEC,” inProc. 9th IEEE Symp. Computers and Communications (ISCC), Alexandria, Egypt, 2004.

[36] Y. N. Han, H. G. Bahk, and S. Yang, “CDMA mobile system overview: introduction, background, and system concepts,” ETRI J., vol. 19, no. 3, pp. 83–97, 1997.

[37] J.D. Johnston, ‘Perceptual transform coding of wideband stereo signals’, Proc. ICASSP, 1989, Glasgow, pp. 1993-1996.

# Appendix A: Glossary of Terms

**Autocorrelation:** Multiplication of a signal with a time-delayed replica of itself, with product integration. $\psi_{auto} = \int f(t)\, f(t-\tau) dt$ [28].

**Baseband:** The basic information channel for a communication system. For voice systems approximately 3 kHz low pass baseband bandwidths are usually required [28].

**Biphase:** A term used to signify two phase (±90°) phase shift keying [28].

**Carrier:** A term used to identify the center frequency of a signal, whether modulated or not [28].

**CDMA:** Code division multiple access. Any signal multiplexing technique using codes to separate signals [28].

**Channel:** A means of one way transmission. A defined sequence of periods (e.g. timeslots) in a TDMA system: a defined frequency band in an FDMA system; a defined sequence of periods and frequency bands in a frequency hopped system [13].

**Cell:** The area of radio coverage locally defined as seen by the mobile station with a base station identity code wand uniquely defined by the network with a call global identification [13].

**Cell Coverage Area:** Area within which a defined quality of reception is provided, that is, a planned radio coverage of a cell [13].

**Chip:** The output of a code generator during one clock interval [28].

**Clock:** The frequency or chip rate reference used to set the rate of code generation in a spread spectrum system [28].

**Codes:** *Composite*: A code made up of two or more distinct codes, usually generated by modulo-2 addition. *Gold:* A particular type of composite code oriented to multiple

access. *JPL*: A particular type of composite code oriented to ranging and rapid synch acquisition. *Linear*: A class of codes in which only linear operations are allowed for generation (e.g. for binary codes only modulo-2 addition or subtraction is allowed). *Maximal*: Codes whose length is $2^N - 1$, where N is the generator length. *Nonlinear*: A class of codes in which nonlinear operations are performed, all codes other than linear. These codes encompass the truly secure codes. *Nonmaximal*: All codes other than maximal. *Quasiorthogonal*: A term used to describe codes with good mutual cross correlation properties. *Synchopated*: Composite codes generated with nonsynchronous or nonsimultaneous chip timing [28].

**Coherent:** A synchronized, phased matched condition between a receiver's reference and the desired signal [28].

**Correlation:** The degree of agreement between a pair of signals [28].

**Correlator:** The section of a spread spectrum system in which a received signal and the local reference are compared for agreement. The desired synchronized signal is despread and undesired signals are spread [28].

**Cross Correlation:** The degree of agreement between two unlike signals, defined to be: $\psi_{cross} = \int f(t)\, g(t-\tau) dt$ [28].

**De-Interleaving:** The process of restoring an interleaved signal [18].

**Direct Sequence:** A form of modulation wherein a code sequence is used to directly modulate a carrier, usually by phase shift keying [28].

**Downlink:** Physical link from BS toward the MS (BS transmits, MS receives) [13].

**FDMA:** Frequency division multiple access. A signal multiplexing technique using frequency to separate signals.

**Frequency Synthesis:** Generation of multiple frequencies from one or a few sources. See direct and indirect synthesis categories [28].

**Guard Period:** Period at beginning and end of timeslot during which MS transmission is attenuated [13].

**Heterodyne:** A process in which a signal is mixed with another or multiplied with it, usually for the purpose of frequency translation [28].

**Integrate and Dump:** A form of synchronous detector that integrates over an information bit period and is discharged (dumped) at the end of that period. The integrator is sampled just before discharge to determine its state. When synchronized with a received signal, it is a matched filter [28].

**Interference:** Any signal that tends to hamper the normal reception of a desired signal. Equivalent to jamming except considered non-hostile in origin [28].

**Interleaving:** Mixing the source signal in a predictable way [18].

**Jammer:** A source of deliberate interference, usually hostile [28].

**Jamming:** The signal produced by the jammer, intended to interfere with reception of a desired signal [28].

**Jamming Margin ($M_j$):** The amount of interference a system is able to withstand while producing the required output signal to noise and bit error rates [28].

***m*-ary:** Used to designate signal forms other than binary; multilevel signal formats such as ternary (3), and quaternary (4) [28].

**Matched Filter:** The ideal filter; designed for reception of a specific signal, passing the signal with minimum loss while passing minimum noise [28].

**Maximal Sequences:** A type of code sequence which is the longest that can be generated by a feedback code generator. For binary code generation ts length is $2^N - 1$, where N is the generator length in chips [28].

**Modem:** Abbreviation for modulator-demodulator [28].

**Noise Figure:** The amount of noise added (in dB) by a signal handling stage to that existing at its input. Usually applied to a first RF stage to signify the amount of noise output over the noise input [28].

**Orthogonal:** Used to signify signals that are mutually transparent or noninterfering. Frequency and amplitude modulation are ideally considered orthogonal [28].

**Preamble:** A special code sequence used for acquisition of synchronization in spread spectrum systems. Usually short compared with codes used in communication [28].

**Process Gain ($G_P$):** The gain or S/N improvement enjoyed by a spread spectrum system due to coherent band spreading and remapping of the desired signal, expressed by the equation: $\frac{BW_{RF}}{R_{info}} = G_p = TW$ [28].

**Propagation:** The process by which a signal proceeds from the transmitter to a receiver [28].

**Pseudonoise:** A term used to signify any of a group of code sequences that exhibit noise like properties. Also sometimes used as a name for systems that employ pseudonoise code modulation [28].

**PSK:** Abbreviation for phase shift keying [28].

**QPSK:** Four phase, phase shift keyed signal format. Phase shifts occur at 0°, ±90°, and 180°, depending on the data being transmitted [28].

**Remapping:** The process of correlation in which a spread spectrum desired signal is transformed into a coherent narrowband signal and undesired signals are transformed into wide bandwidths [28].

**Search:** A process by which spread spectrum systems acquire synchronization [28].

**Secure:** A term used to signify that a signal is protected from both inadvertent and deliberate attempts at decoding the signal. For all practical purposes a secure signal can be decoded only by possessing the proper key [28].

**Service Provider:** The organization through which the subscriber obtains telecommunication services. This may be a network operator or possibly a separate body [13].

**Sequence:** A code or train of chips [28].

**Shift Register:** A sequentially connected group of delay elements used to generate code sequences in spread spectrum systems [28].

**Spread Spectrum:** Any of a group of modulation formats in which an RF bandwidth much wider than necessary is used to transmit an information signal so that a signal to noise improvement may be gained in the process [28].

**Synchronization:** Timing agreement or the act of gaining timing agreement between spread spectrum transmitter and its receiving system [28].

**TDMA:** Time division multiple access. A signal multiplexing technique using time division to separate signals [28].

# Appendix B: Matlab Code

## System Functions

### modulation_test.m

```
function
decoded_signal=modulation_test(message_signal,chunk_size,max_shift)
%function
decoded_signal=modulation_test(message_signal,chunk_size,max_shift)

%Simulates a baseband chunked DS-CDMA system with random channel shifts

%Inputs:
%message signal: set of messages to be transmitted
%chunk_size: number of symbols per chunk
%max_shift: maximum allowed chip shift in channel

%Outputs:
%decoded_signal: signal decoded at receiver

[num_messages,message_len]=size(message_signal);

%Modulate
encoded_signal=encoder(message_signal,chunk_size);

%Send through channel
[channel_signal,shift]=channel(encoded_signal,max_shift);

%Decode received signal
decoded_signal=decoder(channel_signal,chunk_size,num_messages,shift);

end
```

### expand.m

```
function expanded_signal = expand(input_signal,chunk_size,num_dup)
%function expanded_signal = expand(input_signal,chunk_size,num_dup)

%Compress and repeat the signal

%Input arguments:
%input_signal: two dimensional matrix to be expanded horizontally
%chunk_size: length of input signal to be duplicated at a time
%num_dup: number of times for each length of the input signal to be
%         duplicated

%Output arguments:
%expanded_signal: input_signal repeated num_dup times in sections of
%                 length chunk_size
```

```
[m n]=size(input_signal);

for k=1:n/chunk_size
    A(:,:,k)=input_signal(:,(k-1)*chunk_size+1:k*chunk_size);
end

expanded_signal=reshape(repmat(A,1,num_dup),m,n*num_dup);
end
```

### encoder.m

```
function encoded_signal=encoder(message_signals,chunk_size)
%function encoded_signal=encoder(message_signals,chunk_size)

%Compress and repeat the signal in chunks

%Inputs:
%message_signals: the message to be encoded
%chunk_size: length of signal to be compressed and repeated at a time

%Outputs:
%encoded_signal: sum of the modulated signals

[num_messages,len_messages]=size(message_signals);

%Compress and repeat messages
expanded_message=expand(message_signals,chunk_size,num_messages);

%Create Walsh cover
walsh_sequence=hadamard(num_messages);
A=expand((walsh_sequence),1,chunk_size);
walsh_cover=repmat(A,1,length(expanded_message)/length(A));

%Modulate each signal
encoded_signal=expanded_message.*walsh_cover;
end
```

### channel.m

```
function [channel_signal,shift]=channel(encoded_signal,max_shift)
%function [channel_signal,shift]=channel(encoded_signal,max_shift)

%Shift each signal by a random number of chips, then sum the resulting
%signals

%Input Arguments:
%encoded signal: superposition of modulated signals
%max_shift: maximum shift size

%Output Arguments:
%signal: encoded signal, with the effects of multipath applied

%Calculate random shift for each message
```

```
[num_messages,~]=size(encoded_signal);
shift=(randi(2*max_shift+1,num_messages,1)-1-max_shift);

%Shift each message
for k=unique(shift(shift~=0))'

encoded_signal(shift==k,:)=circshift(encoded_signal(shift==k,:),[0,k]);
end

%Sum all messages in channel
channel_signal=sum(encoded_signal);

end
```

**decoder.m**

```
function
decoded_signal=decoder(inc_signal,chunk_size,num_messages,shift)
%function
decoded_signal=decoder(inc_signal,chunk_size,num_messages,shift)

%Takes a chunked DS-CDMA signal, and decodes it.

%Inputs:
%inc_signal: encoded signal to be decoded
%chunk_size: length of signal to be repeated at a time
%num_messages: number of times to duplicate the signal
%num_messages: number of encoded messages

%Outputs:
%decoded_signals: decoded signal, should be equal to the original
message
%                 signal

%Duplicate the signal
signal=repmat(inc_signal, num_messages, 1);

%Unshift signals
for k=unique(shift(shift~=0))'
    signal(shift==k,:)=circshift(signal(shift==k,:),[0,-k]);
end

%Generate Walsh cover
walsh_sequence=hadamard(num_messages);
A=expand((walsh_sequence),1,chunk_size);
expanded_walsh=repmat(A,1,length(inc_signal)/length(A));

%Despread
z=signal.*expanded_walsh;

%Shift and dump
shifted_signal=[];
indices=[];
for l=0:num_messages*chunk_size:length(inc_signal)-1
```

```
    for k=1:chunk_size
        index=[k+l:chunk_size:num_messages*chunk_size+l];
        indices=[indices,index];
    end
end

%Integrate and dump
shifted_signal=z(:,indices);
x=reshape(shifted_signal,num_messages,num_messages,length(inc_signal)/n
um_messages);
decoded_signal=reshape(sum(x,2),num_messages,length(inc_signal)/num_mes
sages)/num_messages;

end
```

## Additional Functions

### source.m

```
function x = source(M,N)
%function x = source(M,N)

%Generate random binary signals

%Inputs
%M: Desired number of messages
%N: Desired length of messages

%Output
%x: M Randomly generated message signals of length N
%    message signals, represented by 1's and -1's

x = 2.*randi([0,1],M,N)-1;

end
```

### fcxcorr.m

```
function [xc] = fcxcorr(u1,u2)
%[xc] = fcxcorr(u1,u2)

%Uses fft to calculate the normalized circular cross correlation of two
periodic
%signal vectors. Both inputs must be the same size.

%Input arguments:
%u1: First input vector
%u2: Second input vector

%Output arguments:
%xc: normalized circular cross correlation between u1 and u2
```

```
xc=ifft(fft(u1).*conj(fft(u2)))/(norm(u1)*norm(u2));
```

## Simulations

### walsh_64.m

```
%Walsh_64.m
%64 Code Walsh Set Representation

%Generate Plot
figure(1)
imagesc(hadamard(64))
colormap(gray(2))

%Label Plot
xlabel('Chip #')
ylabel('Walsh Code #')
title('64 Code Walsh Set')
```

### walsh_corr.m

```
%walsh_corr.m
%Compute the normalized cross correlation between all the walsh codes
%in a set, and plot their average magnitude vs shift

clear

%Generate Walsh codes
walsh_sequence=hadamard(64);

%Compute correlations between all walsh codes
k=1;
for m=1:length(walsh_sequence)
    for n=1:m-1

fcxcorrelation(k,:)=fcxcorr(walsh_sequence(m,:),walsh_sequence(n,:));
        k=k+1;
    end
end


%Plot results
figure(1)
plot([0:63],mean(abs(fcxcorrelation),1))
title('Average Normalized Magnitude of Cross Correlation of 64 Walsh
Set')
xlabel('Number of Shifts')
xlim([0,31])
```

### chunked_corr.m

```
%Compute the normalized cross correlation between all the walsh codes,
%and plot them

clear

%Generate set of 16 Walsh Codes
walsh_sequence=hadamard(16);

for l=0:4

    %Generate Walsh covers for various chunk sizes
    chunk_size=2^l;
    A=expand((walsh_sequence),1,chunk_size);
    walsh_cover=repmat(A,1,length(walsh_sequence)/chunk_size);

    %Calculate cross correlations between Walsh covers
    k=1;
    for m=1:length(walsh_sequence)
        for n=1:m-1
            walsh_corr(k,:)=fcxcorr(walsh_cover(m,:),walsh_cover(n,:));
            k=k+1;
        end
    end

    %Plot average normalized cross correlation between Walsh covers
    figure(1)
    subplot(5,1,l+1)
    stem([0:length(walsh_cover)-1],mean(abs(walsh_corr),1))
    hold on
    plot(ones(1,16)*mean(mean(abs(walsh_corr),1)),'--k')
    xlim([0,15])
    ylabel(['Chunk = ',num2str(2^l)])
    if l==0
        title('Average Normalized Magnitude of Cross Correlation')
    end

end

xlabel('Number of Shifts (Chips)')
```

### encoded_spectrums.m

```
%encoded_spectrums.m
%Plot the spectrums of encoding signals

%% Without Chunking
%Plot Spectrums of System with 1 Symbol Chunks

chunk_size=1;

%Generate Messages
num_messages=64;
```

```
message_len=64;
message=source(num_messages,message_len);

%Parameters
user=32;
N=message_len*num_messages;

%Compress and Repeat Message
expanded_message=expand(message,chunk_size,num_messages);

%Generate Walsh Cover
walsh_sequence=hadamard(num_messages);
A=expand((walsh_sequence),1,chunk_size);
expanded_walsh=repmat(A,1,length(expanded_message)/length(A));

%Encode Signals Using Walsh Cover
encoded_signal=expanded_message.*expanded_walsh;

%Calculate Relevant FFT's
fft_message=fft(expanded_message,N,2)/length(expanded_message);
fft_walsh=fft(expanded_walsh,N,2)/length(expanded_walsh);
fft_encoded=fft(encoded_signal,N,2)/length(encoded_signal);

%Plot Compressed Message Spectrum
figure(1)
F=num_messages*(-(N-1)/2/N:1/N:(N-1)/2/N);
stem(F,abs(circshift(fft_message(user,:),[0,length(fft_message)/2])),'m
arker','none');
title(['Compressed Message Spectrum, User = ',num2str(user),', Chunk
Size = ',num2str(chunk_size)])
xlabel('Frequency Normalized to Data Rate')
ylabel('Amplitude')

%Plot Encoded Message Spectrum
figure(2)
F=num_messages*(0:1/N:(N-1)/N);
stem(F,abs(fft_encoded(user,:)),'marker','none')
hold on
stem(F,circshift(abs(fft_walsh(user,:)),[0,length(fft_walsh)/2]),'r','m
arker','none')
hold off
title(['Encoded Message Spectrum, User = ',num2str(user),', Chunk Size
= ',num2str(chunk_size)])
legend('Encoded Message','Compressed Walsh')
xlabel('Frequency Normalized to Data Rate')
ylabel('Amplitude')
xlim(num_messages*[0 .5])

%% With Chunking
%Plot Spectrums of DS-CDMA System with 64 Symbol Chunks

chunk_size=64;

%Compress and Repeat Message
expanded_message=expand(message,chunk_size,num_messages);
```

```
%Generate Walsh Cover
walsh_sequence=hadamard(num_messages);
A=expand((walsh_sequence),1,chunk_size);
expanded_walsh=repmat(A,1,length(expanded_message)/length(A));

%Encode Signals Using Walsh Cover
encoded_signal=expanded_message.*expanded_walsh;

%Calculate Relevant FFT's
fft_message=fft(expanded_message,N,2)/length(expanded_message);
fft_walsh=fft(expanded_walsh,N,2)/length(expanded_walsh);
fft_encoded=fft(encoded_signal,N,2)/length(encoded_signal);

%Plot Compressed Walsh Spectrum
figure(3)
F=num_messages*(-(N-1)/2/N:1/N:(N-1)/2/N);
stem(F,circshift(abs(fft_walsh(user,:)),[0,length(fft_walsh)/2]),'marke
r','none')
title(['Compressed Walsh Spectrum, User = ',num2str(user),', Chunk Size
= ',num2str(chunk_size)])
xlabel('Frequency Normalized to Data Rate')
ylabel('Amplitude')

%Plot Encoded Message Spectrum
figure(4)
F=num_messages*(0:1/N:(N-1)/N);
stem(F,abs(fft_encoded(user,:)),'marker','none')
hold on
stem(F,abs(fft_message(user,:)),'r','marker','none')
hold off
title(['Encoded Message Spectrum, User = ',num2str(user),', Chunk Size
= ',num2str(chunk_size)])
legend('Encoded Message','Compressed Message')
xlabel('Frequency Normalized to Data Rate')
ylabel('Amplitude')
xlim(num_messages*[0 .5])
```

### error_distributions.m

```
%error_distributions.m
%Error Distributions with Maximum Shift = 1 chip
%Simulates the system, plots received error histograms
%Plots mean, variance, and bit error rate vs. maximum shift

clear
shift=1;

for n=1:7
    for k=1:100
        %Generate random set of messages
        message(:,:,k)=source(64,64);

        %Run chunked DS-CDMA system
```

```
            decoded(:,:,k)=modulation_test(message(:,:,k),2^(n-1),shift);
        end

        %Compare decoded and original signals
        error(n,:)=decoded(:)-message(:);

        %Determine received values
        received(decoded<0)=-1;
        received(decoded>=0)=1;

        %Plot histograms
        figure(1)
        if n<=4
            subplot(2,2,n)
            x=-2.5:.1:2.5;
            [C,x]=hist(error(n,:),x);
            C=C./sum(C);
            bar(x,C,1)
            if n==1
                title('Chunk Size = 1 Symbol')
            else
                title(['Chunk Size = ',num2str(2^(n-1)),' Symbols'])
            end
            xlabel('Error')
            ylabel('Relative Frequency')
            xlim([-2.5,2.5])
        end

        %Find Bit Error Rate
        BER(1,n)=1-sum(received(:)==message(:))/length(message(:));
    end

    %Calculate error statistics
    mean_error=mean(error');
    var_error=var(error');

    %Plot Bit error rate
    figure(8)
    plot([0:6],BER,'-*')
    title('Bit Error Rate vs Chunk Size')
    xlabel('log_2(Chunk Size)')
    ylabel('Bit Error Rate')

    %Plot error variance
    figure(9)
    plot([0:6],var_error,'-*')
    title('Error Variance vs Chunk Size')
    xlabel('log_2(Chunk Size)')
    ylabel('Error Variance')

    %Plot mean error
    figure(10)
    stem([0:6],mean_error)
    title('Mean Error vs Chunk Size')
    ylabel('Mean Error')
```

```
xlabel('log_2(Chunk Size)')
```

### error_vs_shift.m

```
%error_vs_shift.m
%Plots the variance and bit error rate vs. maximum shift for various
%chunk sizes

clear

for n=0:6
    for shift=0:64
        for k=1:10
            %Generate messages
            message(:,:,k)=source(64,64);

            %Run chunked DS-CDMA system
            decoded(:,:,k)=modulation_test(message(:,:,k),2^n,shift);
        end

        %Find error variance
        error=decoded(:)-message(:);
        error_var(shift+1,n+1)=var(error);
        error_mean(shift+1,n+1)=mean(error);

        %Find bit error rate
        received(decoded<0)=-1;
        received(decoded>=0)=1;
        Bit_Error_Rate(shift+1,n+1)=1-
sum(received(:)==message(:))/length(message(:));
    end
end

%Plot Bit Error Rate vs. Shift
figure(1)
plot(0:shift,Bit_Error_Rate)
legend('Chunk Size = 1 Symbol','Chunk size = 2 Symbols','Chunk Size = 4
Symbols','Chunk Size = 8 Symbols','Chunk Size = 16 Symbols','Chunk Size
= 32 Symbols','Chunk Size = 64 Symbols')
title('Bit Error Rate vs. Maximum Shift Size')
xlabel('Maximum Shift Size')
ylabel('Bit Error Rate')
xlim([1,64])


%Plot Error Variance vs. Shift
figure(2)
plot(0:shift,error_var)
legend('Chunk Size = 1 Symbol','Chunk size = 2 Symbols','Chunk Size = 4
Symbols','Chunk Size = 8 Symbols','Chunk Size = 16 Symbols','Chunk Size
= 32 Symbols','Chunk Size = 64 Symbols')
title('Error Variance vs. Maximum Shift Size')
xlabel('Maximum Shift Size')
ylabel('Error Variance')
xlim([1,64])
```

**error_vs_2d_position.m**

```
%error_vs_2d_position.m
%Simulates a chunked DS-CDMA system, and plots the resulting error
%magnitudes vs channel and symbol position
clear
max_shift = 1;

for L=0:1
    max_shift=2^L;

    for n=0:6

        %Generate random Set of messages
        message(:,:,n+1)=source(64,64);

        %Run chunked DS-CDMA system
         decoded(:,:,n+1) =
modulation_test(message(:,:,n+1),2^n,max_shift);

        %Compare decoded and encoded signals
        error(:,:,n+1)=message(:,:,n+1)-decoded(:,:,n+1);
    end

    %Plot Results
    k=1;
    for n=1:2:7
        figure (L+1)
        subplot(2,2,k)
        imagesc((abs(error(:,:,n))))
        colorbar
        title(['Chunk Size = ',num2str(2^(n-1)),' Symbols'])
        xlabel('Symbol #')
        ylabel('Channel #')
        k=k+1;
    end
end
```

**error_vs_position.m**

```
%error_vs_position.m
%Simulates a DS-CDMA system, and plots the received error vs the symbol
%position

clear

runs=100;

for L=0:3

    shift=2.^L;

    for k=1:runs
```

```
        for n=1:2:7
            %Create Messages
            message(:,:,n,k)=source(64,64);

            %Transmit and Receive Messages
            decoded(:,:,n,k)=modulation_test(message(:,:,n,k),2^(n-
1),shift);

            %Determine which bits are correct
            for m=1:64
                received(decoded(m,:,n,k)<0)=-1;
                received(decoded(m,:,n,k)>=0)=1;
                correctbit(m,:,n,k)=received==message(m,:,n,k);
            end

        end
    end

    %Calculate Bit Error Rates
    BER(:,L+1,:)=1-sum(sum(correctbit,1),4)/64/runs;
end

%Plot Bit Error Rates
for n=1:2:7
    figure(n)
    plot(BER(:,:,n))
    xlim([1,64])
    title(['Bit Error Rate vs. Position, Chunk size = ',num2str(2^(n-
1))])
    legend('Maximum Shift = 1 Chip','Maximum Shift = 2 Chips','Maximum
Shift = 4 Chips','Maximum Shift = 8 Chips')
    xlabel('Symbol #')
    ylabel('Bit Error Rate')
end
```

## guard_bands.m

```
%guard_bands.m
%Plots the fraction of bits lost to guard bands for various shift sizes
and
%chunk sizes

clear

%Generate Functions
n=256;
for x=1:10
    z(x,:)=[ones(1,4*x),4*x./(4*x:n)];
end

%Plot results
figure(1)
```

```
plot([0:n],z((1:2:10),:))
xlim([0,n])
title('Fraction of Bits Quarantined vs. Chunk Size')
legend('Max Shift = 1 Chip','Max Shift = 3 Chips','Max Shift = 5
Chips','Max Shift = 7 Chips','Max Shift = 9 Chips')
xlabel('Chunk Size (Symbols)')
ylabel('Fraction of Bits Quarantined')
```

# Appendix C: Analysis of Senior Project Design

**Project Title:** Data Chunking in Quasi-Synchronous DS-CDMA

**Student's Name:** Trevor Dalke **Student's Signature:**

**Advisor's Name:** Dr. Vladimir Prodanov **Advisor's Initials:** **Date:**

**Summary of Functional Requirements:**

This project describes the process of "data chunking" in quasi-synchronous DS-CDMA that could be implemented in commercial mobile networks. Possible implementations of data chunking in the reverse link are explored. The performance of such a system is simulated in MATLAB where small random shifts between users are permitted.

**Primary Constraints:**

The system was tested in a multiuser environment through simulations. A reasonable model had to be developed to discover the performance of the system. The final model that was decided upon was a baseband model, with randomly distributed synchronization errors between users. It was assumed that the decoder had knowledge of these synchronization errors, and used them to decode the messages. A more complex channel model turned out to not be necessary for the testing of synchronization errors in this case.

To gather statistically meaningful results, the simulations were repeated many times. Due to hardware and software limitations, these simulations were time consuming. The speed of these simulations were greatly improved by optimizing the MATLAB code.

This involved “vectorizing” as much as possible to reduce the number of loops used in the simulation. The result was a substantial reduction in simulation time, allowing for more thorough simulations to be performed.

**Economic:**

The project did not have any direct costs, as it made use of previously owned software, without any hardware costs. This project probably took on the order of 300 hours of time to develop and test. I was the first person to work on this project.

**If manufactured on a commercial basis:**

If this product were to be implemented, it would require significant modifications to the transmitter of CDMA mobile devices and the receiver of CDMA base stations. The base station receiver could be designed to be reverse compatible with current CDMA devices. Modifying the base stations in the network would be the most expensive part of the implementation, as all of the base stations in the network would require modification. Manufacturing the modified transmitter on the mobile devices would not be any more expensive than manufacturing the current transmitter.

**Environmental:**

This system should not have any significant effect on the impact that operating a CDMA network has on the environment.

**Manufacturability:**

Because this system is similar to existing designs, it could use existing manufacturing equipment and processes without any significant change in manufacturing costs.

**Sustainability:**

Implementation of this system should not have any impact on the sustainability of current mobile networks.

**Ethical:**

There are no known ethical issues with this system.

**Health and Safety:**

There are no known safety concerns with the design, manufacture or use of this project.

**Social and Political:**

There are no known social or political concerns with the design, manufacture or use of this project.

**Development:**

This project required heavy use of MATLAB for simulations. I had already had previous experience with MATLAB; however, I learned a great deal from doing these simulations. I learned of many new MATLAB functions, as well as tricks to speed up the existing code to minimize simulation time.