Florida State University Libraries

Electronic Theses, Treatises and Dissertations

The Graduate School

2003

Passive Detection Suppression of Cyclostationary Phase Coded Waveforms

Mohsin M. Benghuzzi

THE FLORIDA STATE UNIVERSITY
FAMU-FSU COLLEGE OF ENGINEERING

PASSIVE DETECTION SUPPRESSION
OF
CYCLOSTATIONARY PHASE CODED WAVEFORMS

By

MOHSIN M. BENGHUZZI

A Dissertation submitted to the
Department of Electrical and Computer Engineering
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Degree Awarded:
Spring Semester, 2003

The members of the Committee approve the dissertation of Mohsin M. Benghuzzi defended on January 31, 2003.

Frank Gross
Professor Directing Dissertation

James Simpson
Outside Committee Member

Simon Foo
Committee Member

Rodney Roberts
Committee Member

Approved:

Reginald Perry, Chair
Department of Electrical and Computer Engineering

ACKNOWLEDGEMENTS

My sincere thanks to my committee members Dr. Frank Gross, Dr. James Simpson, Dr. Simon Foo, Dr. Rodney Roberts. To the professor directing my dissertation, Dr. Frank Gross, no amount of gratitude will suffice. Dr. Gross has been a source of inspiration throughout my Ph.D. endeavor and is the reason that I completed the degree.

I would also like to thank the past and present members of the Sensor Systems Research Lab for their help and support. A special thanks to Shantanu Joshi for his invaluable discussions on radar theory and the MATLAB application.

TABLE OF CONTENTS

LIST OF FIGURES	vi
ABSTRACT	ix
CHAPTER 1 – INTRODUCTION	1
CHAPTER 2 – STOCHASTIC PROCESSES BASIC CONCEPTS	4
The Random Variable	4
The Distribution Function	5
The Density Function	5
Mean of a Random Variable	6
The Stochastic Process	7
The Autocorrelation Function	9
The Cross-Correlation Function	11
The Power Spectral Density	11
Discrete-Time Stochastic Processes	13
Time-Average of a Stochastic Sequence	13
Time-Autocorrelation of a Stochastic Sequence	14
Time Cross-Correlation of a Stochastic Sequence	14
Stationarity	15
CHAPTER 3 – CYCLOSTATIONARITY	17
Cyclostationary Signals	18
First and Second-Order Periodicity in Cyclostationary Signals	20
Hidden Periodicity in Cyclostationary Signals	21
An Example of Hidden Periodicity in Cyclostationary Signals	21
CHAPTER 4 – CODED WAVEFORMS	26
Basic Radar Concepts	26
Uncoded Pulsed Radar	28
Pulse Compression	29
Waveform Coding	31
Binary Phase Codes	32
Complementary Phase Codes	36
Golay Codes	38
Welch Codes	40
Detectability of Features in Coded Waveforms	42

CHAPTER 5 – PASSIVE DETECTORS	44
Square-Law Detector	45
Fourth-Law Detector	47
Delay-and-Multiply Detector	47
CHAPTER 6 – PASSIVE DETECTION ENERGY SUPPRESSION	51
Simulation of the WELT Code Waveform	51
Generation of Spectral Lines	53
WELT Code Filter Design	56
Energy Suppression Results	66
Future Research	77
CHAPTER 7 – CONCLUSION	82
APPENDIX A	87
APPENDIX B	90
APPENDIX C	92
APPENDIX D	94
APPENDIX E	97
APPENDIX F	100
APPENDIX G	104
APPENDIX H	108
APPENDIX I	116
APPENDIX J	124
APPENDIX K	132
APPENDIX L	139
REFERENCES	144
BIOGRAPHICAL SKETCH	148

LIST OF FIGURES

Figure 2.1 - An Ensemble of a Continuous Stochastic Process	8
Figure 2.2 - An Ensemble of a Continuous Stochastic Sequence	10
Figure 3.1 - (a) Power Spectral Density (PSD) of a Low-Pass Signal. (b) PSD of an AM Signal. (c) PSD of a Squared Low-Pass Signal. (d) PSD of a Squared AM Signal	22
Figure 4.1 - Elements of a Simple Radar	27
Figure 4.2 - Typical Transmitted Radar Radio Frequency Pulses	27
Figure 4.3 - An Uncoded Pulse Waveform and its Autocorrelation	29
Figure 4.4 - Pulse Compression. (a) One Transmit Pulse with Higher Peak Power level. (b) A Set of Sub-pulses with Lower Peak Power Level	30
Figure 4.5 - Binary Coded Signals. Top – Modulating Binary Code. Bottom – Corresponding Phase Coded cw Signal	32
Figure 4.6 - A Barker Coded Waveform of Length 13	34
Figure 4.7 - Barker Codes	35
Figure 4.8 - The Autocorrelation of a Barker Coded Waveform of Length 13	35
Figure 4.9 - Complementary Codes A and B	36
Figure 4.10 - Autocorrelation Functions for Complementary Codes A and B	37
Figure 4.11 - Sum of the Autocorrelation Functions of Codes A and B	37
Figure 4.12 - A 64 Element Golay Coded Waveform and its Spectral Contents	40
Figure 4.13 - A 64 Element Welty Coded Waveform and its Spectral Contents	42
Figure 5.1 - Block Diagram of the Square-Law Detector	46
Figure 5.2 - Block Diagram of the Fourth-Law Detector	48
Figure 5.3 - Block Diagram of the Delay and Multiply Detector	49
Figure 6.1 – (a) Square-Law Transformation. (b) Fourth-Law Transformation. (c) Delay-and-Multiply Transformation	53

Figure 6.2 - (a) PSD of a Welter Coded cw Waveform. (b) PSD of the Welter Coded cw Waveform When Passed through a Squarer. (c) PSD of (a) with Magnitude in dB. (d) PSD of (b) with Magnitude in dB ..	54
Figure 6.3 - (a) PSD in dB of a Welter Coded cw Waveform. (b) PSD of the Welter Coded cw Waveform When Passed through a 4 th -Law Detector ..	55
Figure 6.4 – A Unit Height Pulse of Duration τ ..	57
Figure 6.5 – Welter Code Consisting of Delayed Unit Height Pulses ..	58
Figure 6.6 – (a) Multiplicative Terms in the Equation For the Fourier Transform of the Square of the Filtered Welter Coded cw Waveform. (b) Approximation of the Entire Spectrum of the Filtered Welter Coded cw Waveform ..	61
Figure 6.7 – Spectrum of Unfiltered and Notch Filtered Welter Code ..	65
Figure 6.8 – Top: Unfiltered Welter Code. Bottom: Filtered Welter Code, $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$..	67
Figure 6.9 – Top: Composite Autocorrelation of the Two Unfiltered Complementary Welter Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$..	68
Figure 6.10 – Top: Spectra of the Square of the Unsuppressed Welter Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a Welter Code Filter of $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$..	69
Figure 6.11 – Top: Unfiltered Welter Code. Bottom: Filtered Welter Code, $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$..	70
Figure 6.12 – Top: Composite Autocorrelation of the Two Unfiltered Complementary Welter Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$..	71
Figure 6.13 – Top: Spectra of the Square of the Unsuppressed Welter Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a Welter Code Filter of $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$..	72
Figure 6.14 – Top: Unfiltered Welter Code. Bottom: Filtered Welter Code, $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$..	73
Figure 6.15 – Top: Composite Autocorrelation of the Two Unfiltered Complementary Welter Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$..	74

Figure 6.16 – Top: Spectra of the Square of the Unsuppressed Welter Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a Welter Code Filter of $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$	75
Figure 6.17 – Percentage Average Power Reduction in the Welter Coded cw Waveform Reference Bandwidth Versus the Normalized Welter Code Notch Filter Bandwidth, B_n	76
Figure 6.18 - Top: Welter Code. Center: $\frac{1}{2}$ Chip Delayed Welter Code. Bottom: $\frac{1}{2}$ Chip Delayed-and-Multiplied Welter Code.	79
Figure 6.19 - Top: PSD of the Welter Code. Bottom: PSD of $\frac{1}{2}$ Chip Delayed-and-Multiplied Welter Code	80

ABSTRACT

Common military radar systems transmit sophisticated pulse compression waveforms for range information. Although much progress has been made in suppressing the platform radar cross section to avoid detection by enemy active radar, little has been done to protect the platform from being detected due to its own on board emissions. An aircraft can have great stealth properties but have a “loud” pulse compression waveform and is therefore liable to detection by enemy passive receivers.

A technique that reduces pulse peak power levels and maintains high range-resolution is known as pulse compression. Pulse compression is a coding technique in which a radar pulse of duration T is subdivided into N sub-pulses. This technique reduces transmitted power, and therefore, reduces detectability by distributing power over time. A matched filter is used at the receiver to reduce (compress) the effective width of the reflected coded pulse.

A disadvantage of using coded waveforms is that the transmitted signal has detectable features. These features can be easily detected by enemy passive detection equipment. Pulse coding generates features such as carrier frequency, chip rate, and frequency of hopping. In particular, presence of carrier frequency in coded waveforms gives rise to cyclostationary properties. Cyclostationary properties in a phase coded waveform are exhibited in the form of spectral lines in the transformed signal generated

using a nonlinear transformation. This study focuses on suppression of cyclostationary properties present in a transformed Welts coded cw waveform with the nonlinear transformation being that of the quadratic type.

In this dissertation, energy suppression of the cyclostationary spectral lines generated in the power spectral density of the square of the Welts coded cw signal is achieved by filtering the Welts code itself. The filter utilized is of the notch type that is centered around the dc frequency. Three various bandwidths of the energy suppression Welts code notch filter are employed. The bandwidths are chosen in relation to the first null of the Welts code sinc spectrum and are set at: $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right), \frac{1}{50} \cdot \left(\frac{2}{\tau}\right), \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, where the first null of the Welts code sinc spectrum is at $\pm \left(\frac{1}{\tau}\right)$ with τ being the code's pulse width.

Spectral line energy suppression is determined by comparing average power within a frequency bandwidth after the square of the filtered and unfiltered Welts coded cw waveforms. The squared cw waveform reference bandwidth within which average power is compared is chosen to be equal to the bandwidth of the main-lobe in the Welts code spectrum, that is $\left(\frac{2}{\tau}\right)$, or 6 MHz, and is centered around twice the carrier frequency, or around 24 MHz. Average power within the reference bandwidth of the unfiltered Welts coded cw waveform is measured to be 10.9 dBm.

Filtering using the first Welts code notch filter bandwidth of $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$, or 60 kHz results in no code distortion. Filtering using this bandwidth also causes no degradation in

the WELTI code's desirable complementary characteristics as the composite autocorrelation's narrow central peak is retained and the time side-lobes still cancel. The average power within the CW waveform reference bandwidth, for this notch filter bandwidth case, is measured to be 8.98 dBm, which is a reduction of approximately 36.6% from the average power for the unsuppressed CW waveform.

Filtering using the second energy suppression WELTI code notch filter bandwidth of $\frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$, or 120 kHz results in slight code distortion; nonetheless, the overall shape of the WELTI code is maintained. For this bandwidth case also, filtering causes no degradation in the WELTI code composite autocorrelation's narrow central peak characteristic, and the time side-lobes still virtually cancel. The average power within the CW waveform reference bandwidth is measured to be 7.32 dBm, which is a reduction of approximately 56.5 % from the average power for the unsuppressed CW waveform.

Filtering using the third energy suppression WELTI code notch filter bandwidth of $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, or 240 kHz resulted in highly noticeable code distortion. For this bandwidth case, the narrow central peak characteristic also remained unchanged, however, no complete side-lobe cancellation resulted. The average power within the CW waveform reference bandwidth was measured to be 4.47 dBm, which is a reduction of approximately 77.4 % from the average power for the unsuppressed CW waveform.

CHAPTER 1

INTRODUCTION

In a traditional uncoded pulsed radar system, an electromagnetic pulse of width T is radiated in a direction of interest. If an object lies in the same direction, some of the energy will be reflected back toward the transmitter. A receiver is used to detect this returned pulse and an estimate of the range to the object can be made. The returned power is inversely proportional to the range to the fourth power. For a reasonable range-resolution, a fairly high power would need to be transmitted in a very short time. This type of pulse can be easily detected by enemy radar.

A technique that reduces pulse peak power levels and maintains high range-resolution is known as pulse compression. Pulse compression is a coding technique in which a radar pulse of duration T is subdivided into N sub-pulses. This technique reduces transmitted power, and therefore, reduces detectability by distributing power over time. A matched filter is used at the receiver to reduce (compress) the effective width of the reflected coded pulse.

A disadvantage of using coded waveforms in radar is that the transmitted signal has detectable features. These features can be easily detected by enemy passive detection equipment. Pulse coding generates features such as carrier frequency, chip rate, and frequency of hopping. In particular, presence of a carrier frequency in coded waveforms

gives rise to cyclostationary properties. Presence of the cyclostationary properties in a pulse coded waveform is what renders the waveform an enemy detectable signal. This dissertation focuses on suppression of cyclostationary properties present in pulse coded waveforms, more specifically, on suppression of cyclostationary properties present in coded waveforms when the pulse code employed is of the Welty type.

To familiarize the reader with the basic concepts needed for the study of cyclostationarity, basic concepts of random variables, stochastic processes, and stationarity are first reviewed in Chapter 2. A discussion of cyclostationarity then follows in Chapter 3. For a complete understanding of the concept of cyclostationarity, various definitions of the concept are presented. First and second-order periodicity in cyclostationary signals are discussed next. A non-trivial example of first-order hidden periodicity in cyclostationary signals is then given.

In Chapter 4, the concepts of pulse compression and waveform coding in radar are introduced. Complementary phase codes, which are a subclass of binary phase codes, are then discussed. Discussion of the two well-known complementary codes, the Golay code and the Welty code, is next presented. Detectability of features in coded waveforms is then discussed in the chapter. Finally, feature detectors are introduced. In Chapter 5, the three commonly used feature detectors are presented. The three detectors are the square-law detector, the fourth-law detector, and the delay-and-multiply detector. The coded waveform feature that each of these receivers is capable of detecting is then discussed. Detection steps within each of these three receivers are finally presented.

In Chapter 6, simulation of a complementary pair of Welts coded cw waveforms is performed using MATLAB routines. Generation of cyclostationary spectral lines in the square of a Welts coded cw signal due to the presence of the carrier frequency is discussed. The filtering technique to be used for energy suppression of the cyclostationary spectral lines generated in the square of the Welts coded cw waveform is then presented. Filtering is performed on the Welts code itself, and therefore, the desired Welts code filter characteristics are derived. It is determined that the filter to be utilized should be of the notch type.

Next, cyclostationary spectral lines suppression results are presented in the chapter. Three various bandwidths of the energy suppression Welts code notch filter are employed. The bandwidths are chosen in relation to the first null of the Welts code sinc spectrum. Thus, the notch filter bandwidths are chosen as: $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right), \frac{1}{50} \cdot \left(\frac{2}{\tau}\right), \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, where the first nulls of the Welts code sinc spectrum are at $\pm \left(\frac{1}{\tau}\right)$ with τ being the code's pulse width. Finally, opportunities for further research are discussed in the chapter, in particular, opportunities to investigate generation of spectral lines due to, this time, the presence of second-order periodicity in cyclostationary signals, i.e., due to the presence of second-order periodicity in delayed-and-multiplied Welts codes.

CHAPTER 2

STOCHASTIC PROCESSES BASIC CONCEPTS

In this chapter, random variables, stochastic processes, and stationarity basic concepts are first discussed in order to familiarize the reader with the basic concepts needed for the study of cyclostationarity. A discussion of cyclostationarity then follows.

The Random Variable

A random variable (RV), will be denoted by the upper case letter X , is a rule for assigning to every outcome s of an experiment specified by the space S a real number $X(s)$ [2]. The experiment X , for example, could be n numbers of tosses of a coin with the sample space being $S=\{\text{head, tail}\}$ and the real number $X(s)$ is the total number of occurrence of the outcome ‘head.’ Stated differently, a random variable is considered to be a function that maps all elements of the space S into points on the real line or some parts thereof. A RV X may, therefore, be any function. It is not necessary that outcomes in S map uniquely onto the set of real numbers. More than one outcome in S may map onto a single value of X . The converse, however, is not true. That is, every outcome in S must correspond to one and only one value of X .

The Distribution Function

A concept that is of interest in the study of random variables is the probability that the RV X is less than a given number x . The probability $P\{X \leq x\}$ is the probability of the event $\{X \leq x\}$ [3]. This function of x , denoted $F(x)$, is called the probability distribution function or simply the distribution function of the RV X . Thus,

$$F(x) = P\{X \leq x\} \quad (2.1)$$

For two random variables $X_1 = X(t_1)$ and $X_2 = X(t_2)$, the 2nd order joint distribution function is:

$$F(x_1, x_2; t_1, t_2) = P\{X(t_1) \leq x_1, X(t_2) \leq x_2\} \quad (2.2)$$

For N random variables $X_i = X(t_i)$, $i = 1, 2, \dots, N$, the N^{th} order joint distribution function is

$$F(x_1, \dots, x_N; t_1, \dots, t_N) = P\{X(t_1) \leq x_1, \dots, X(t_N) \leq x_N\} \quad (2.3)$$

The Density Function

The derivative of the distribution function, denoted $f(x)$, is called the probability density function or simply the density function of the RV X . Thus,

$$f(x) = \frac{dF(x)}{dx} \quad (2.4)$$

Some of the common density functions are the Normal or Gaussian, the Binomial, the Poisson, and the Rayleigh densities.

The second and Nth-order density functions are similarly obtained from the derivatives of the appropriate distribution functions:

$$f(x_1, x_2; t_1, t_2) = \frac{\partial^2 F(x_1, x_2; t_1, t_2)}{\partial x_1 \partial x_2} \quad (2.5)$$

$$f(x_1, \dots, x_N; t_1, \dots, t_N) = \frac{\partial^2 F(x_1, \dots, x_N; t_1, \dots, t_N)}{\partial x_1 \dots \partial x_N} \quad (2.6)$$

Mean of a Random Variable

The expected value or mean of a random variable X , denoted $E[X]$, is by definition the integral

$$E[X] = \int_{-\infty}^{\infty} x f(x) dx \quad (2.7)$$

where $f(x)$ is the density function of X .

If X happens to be discrete with values x_i having probabilities $P(x_i)$, the density function then becomes

$$f(x) = \sum_i P(x_i) \delta(x - x_i) \quad (2.8)$$

where $\delta(\cdot)$ is the impulse function. Inserting in Equation (2.7) and using the identity

$$\int_{-\infty}^{\infty} x \delta(x - x_i) dx = x_i \quad (2.9)$$

we obtain

$$E[X] = \sum_i x_i P(x_i) \quad (2.10)$$

for a discrete random variable.

The Stochastic Process

The concept of a stochastic process (also referred to as a random process) is based on enlarging the random variable concept to include time [3]. A stochastic process $X(t, s)$ is a rule for assigning to every outcome s of an experiment S a function $X(t, s)$ [1]. Therefore, a stochastic process $X(t, s)$ is a family of time functions depending on the parameter s . The domain of s is the set of all outcomes and the domain of t is the set of real numbers.

Figure 2.1 - An Ensemble of a Continuous Stochastic Process.

As mentioned earlier, a stochastic process $X(t, s)$ represents a family or ensemble of time functions when both t and s are variables. Figure 2.1 illustrates a few members of an ensemble. Each member time function is called a sample function of the process. When t is a variable and s is fixed at a specific value (outcome), a stochastic process represents a single time function [2]. When t is fixed and s is a variable, a stochastic process in this case represents a random variable. It should be noted that the notation $X(t)$ is customary used instead of $X(t, s)$ to represent a stochastic process omitting its dependence on s .

Stochastic processes are classified according to the characteristics of time and the random variable $X = X(t)$, i.e., whether X or t , or both are discrete [41]. A class that is of importance in this study is one for which X is continuous but t has only discrete values, shown in Figure 2.2. A stochastic process of such characteristics is called a stochastic sequence.

The Autocorrelation Function

The autocorrelation function, denoted $R_{xx}(t_1, t_2)$, of a stochastic process $X(t)$ is the expected value of the product $X(t_1)X(t_2)$:

$$R_{xx}(t_1, t_2) = E[X(t_1)X(t_2)] = \iint_{-\infty}^{\infty} x_1 x_2 f(x_1, x_2; t_1, t_2) dx_1 dx_2 \quad (2.11)$$

Figure 2.2 - An Ensemble of a Continuous Stochastic Sequence.

where $f(x_1, x_2; t_1, t_2)$ is the second-order joint density function. The subscript in the denotation is customarily dropped and the autocorrelation is simply expressed as $R(t_1, t_2)$. The autocorrelation function is commonly used as a measure of resemblance (correlation) of a process with itself as time is varied.

The Cross-Correlation Function

The cross-correlation function, denoted $R_{xy}(t_1, t_2)$, of two stochastic process $X(t)$ and $Y(t)$ is the function:

$$R_{xy}(t_1, t_2) = E[X(t_1)Y(t_2)] = \iint_{-\infty}^{\infty} xy f_{xy}(x, y; t_1, t_2) dx dy \quad (2.12)$$

and is commonly used as a measure of resemblance (correlation) of one process, $X(t)$, with another, $Y(t)$, as time is varied.

The Power Spectral Density

The power spectral density (also known as the power density spectrum) of a stochastic signal $X(t)$ describes the way in which relative signal power is distributed with frequency. An important property of the power spectral density (PSD) is that its inverse Fourier transform is the time average of the autocorrelation function [13], that is

$$\frac{1}{2\pi} \int_{-\infty}^{\infty} S_{xx}(w) e^{jw\tau} dw = A[R(t, t + \tau)] \quad (2.13)$$

Where $A[\cdot]$ is the time average, $S_{xx}(w)$ denotes the auto-power spectral density of $X(t)$ and τ is any time interval $t_1 - t_2$. The word auto in the name and the subscript in the denotation are customarily dropped and the PSD is expressed as $S(w)$. For the important case where the stochastic signal is at least wide-sense stationary (WSS processes are discussed in the next section), $A[R(t, t + \tau)] = R(\tau)$ and we get

$$R(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(w) e^{jw\tau} dw \quad (2.14)$$

or

$$S(w) = \int_{-\infty}^{\infty} R(\tau) e^{-jw\tau} d\tau \quad (2.15)$$

Equation (2.15) states that the PSD of a wide-sense stationary stochastic process is the Fourier transform of its autocorrelation function [6]. A point also worth mentioning here is that the average power, $R(0)$, of a wide-sense stationary stochastic signal $X(t)$ is obtained by integrating its $S(w)$ over all frequencies [1]. This is consistent with the fact that $S(w)$ is the “density of power” of $X(t)$ at the frequency w .

Discrete-Time Stochastic Processes

As defined earlier, a discrete-time stochastic process, known as a stochastic sequence, is one for which time has only discrete values. A stochastic sequence of length N is given by

$$X(n) = \sum_{m=-\infty}^{\infty} X(m) \delta(n - m) \quad (2.16)$$

where $X(m)$ is the amplitude of the corresponding continuous process, $X(t)$, at the time location specified by the Dirac function $\delta(n - m)$.

The quantities that are often encountered in stochastic sequence manipulations are the time-average, time autocorrelation, and time cross-correlation of a sequence. These quantities are briefly described next.

Time-Average of a Stochastic Sequence

For a stochastic sequence of length N , the time-average is:

$$\langle X(n) \rangle = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N X(n) \quad (2.17)$$

Time-Autocorrelation of a Stochastic Sequence

The autocorrelation function of a stochastic process as defined by Equation (2.11) is the statistical or ensemble average (i.e., expectation) of the product of $X(t_1)$ and $X(t_2)$ [16]. In terms of time average, on the other hand, the (time) autocorrelation function for a continuous stochastic process is defined as

$$R(t_1, t_2) = \langle X(t_1) X(t_2) \rangle = \lim_{T \rightarrow \infty} \frac{1}{T} \int_0^T X(t_1) X(t_2) dt \quad (2.18)$$

For a stochastic sequence of length N , the time-autocorrelation function is given by

$$R(m_1, m_2) = \langle X(m_1) X(m_2) \rangle = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{m=0}^{N-1} X(m_1) X(m_2) \quad (2.19)$$

Time Cross-Correlation of a Stochastic Sequence

The time cross-correlation function for a continuous stochastic process is defined as

$$R_{XY}(t_1, t_2) = \lim_{T \rightarrow \infty} \frac{1}{T} \int_0^T X(t_1) Y(t_2) dt \quad (2.20)$$

and the time cross-correlation function of a stochastic sequence is given by

$$R_{XY}(m_1, m_2) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{m=0}^{N-1} X(m_1) Y(m_2) \quad (2.21)$$

Stationarity

As stated before, a stochastic process becomes a random variable when time is fixed at a specific value. The resulting random variable will have statistical properties, such as a mean value, moments, variance, that are related to its probability density function. Generally, the N random variables obtained from the process for N time instants will have statistical properties related to their N -dimensional joint density function [3].

A stochastic process $X(t)$ is said to be stationary if all its statistical properties do not change with time [41]. This means that the process $X(t)$ and $X(t + \tau)$ have the same statistics for any τ . Processes that are not stationary are called non-stationary [3]. Furthermore, a process is called stationary to order one if its first-order density function does not change with time [2]. That is,

$$f(x, t_1) = f(x, t_1 + \Delta) \quad (2.22)$$

for any t_1 and Δ . Generally, a process is stationary to order N if its N th-order density function does not change with time. That is,

$$f(x_1, \dots, x_N; t_1, \dots, t_N) = f(x_1, \dots, x_N; t_1 + \Delta, \dots, t_N + \Delta) \quad (2.23)$$

for all $t_1, \dots, t_N$ and Δ . It should be noted that a process stationary to all orders $N = 1, 2, \dots$, is called strict-sense stationary.

Quantities that are often encountered in solving practical problems are the autocorrelation function and mean value of a stochastic process. Problem solutions are generally simplified if these quantities are not dependent on absolute time [16]. Of course, determining that a process is stationary to order two, that is,

$$f(x_1, x_2; t_1, t_2) = f(x_1, x_2; t_1 + \Delta, t_2 + \Delta) \quad (2.24)$$

is sufficient to guarantee that its autocorrelation function and mean value are independent of time. However, second-order stationarity is often more restrictive than necessary. A more relaxed and useful form of stationarity is the wide-sense stationarity. A process $X(t)$ is said to be wide-sense stationary if its mean is constant:

$$E[X(t)] = \text{constant}$$

and its autocorrelation function depends only on $\tau = t_1 - t_2$:

$$E[X(t)X(t + \tau)] = R(\tau)$$

It should be noted that a process stationary to order two is wide-sense stationary, however, the converse is not necessarily true [20].

CHAPTER 3

CYCLOSTATIONARITY

A process $X(t)$ is called strict-sense cyclostationary (SSCS) with period T if its statistical properties are invariant to a shift of the origin by integral multiples of T [22], or equivalently, the N th-order density function of the process must be such that

$$f(x_1, \dots, x_N; t_1+mT, \dots, t_N+mT) = f(x_1, \dots, x_N; t_1, \dots, t_N) \quad (3.1)$$

for every integer m .

A process $X(t)$ is called wide-sense cyclostationary (WSCS) if

$$E[X(t)] = \text{constant}$$

and

$$R(t_1+mT, t_2+mT) = R(t_1, t_2) \quad (3.2)$$

for every integer m .

It follows from the definitions above that if $X(t)$ is SSCS, it is also WSCS. A definition of cyclostationary signals that is more useful in signal processing will be given in the next section.

Cyclostationary processes are a subclass of non-stationary processes [2]. They have more in common with stationary processes than do other subclasses of non-stationary processes. The common structure shared by stationary and cyclostationary processes suggests that important theorems and special theories for stationary processes can be extended or generalized to cyclostationary processes [12].

Cyclostationary Signals

A useful definition of cyclostationarity is that a signal is cyclostationary (wide-sense, WS) if and only if there exist some n th-order nonlinear transformation of the signal that will generate finite-strength additive sine-wave components, i.e., a transformation that will generate spectral lines [4]. The most general time-invariant quadratic transformation of a real time-series $X(t)$ is simply a linear combination of delay products

$$Y(t) = \sum_{\tau_1, \tau_2} h(\tau_1, \tau_2) X(t - \tau_1) X(t - \tau_2) \quad (3.3)$$

for some weighting function $h(\tau_1, \tau_2)$.

As a simple nontrivial example, if there exists a quadratic transformation of a signal (like the squared signal or the product of the signal with a delayed version of itself) that will generate spectral lines then the signal is said to be cyclostationary of order 2. It should be noted that, in contrast to the definition of cyclostationarity just given, for stationary signals, only a spectral line at frequency zero can be generated.

The focus of the study of cyclostationarity has always been on exploiting the redundancy and sine-wave generation properties of cyclostationary signals to perform difficult signal processing tasks. The signals of interest often consist of combinations (additive, multiplicative, and other types) of stationary, periodic, and polyperiodic signals.

A cyclostationary signal is further described as a signal with a frequency spectrum that is correlated with a shifted version of itself [22]. For the case in which $n = 2$, the definition of a cyclostationary signal means that a signal $X(t)$ is cyclostationary with cycle frequency α if and only if at least some of its delay-product waveforms, $Y(t) = X(t)X(t-\tau)$ for some delay τ , exhibit a spectral line at frequency α . The cycle frequency α indicates the frequency at which a signal exhibits stationary properties, i.e., the frequency at which the same statistical properties of a signal are exhibited. It should be noted that all cycle frequencies α for which a signal is cyclostationary are multiples of a single fundamental frequency (equal to the reciprocal of a fundamental period, defined at the beginning of this chapter as T .) If not all cycle frequencies are multiples of a single fundamental frequency, then the signal is said to be polycyclostationary [4]. This means that there is more than one statistical periodicity (cyclostationarity) present in the signal.

First and Second-Order Periodicity in Cyclostationary Signals

A discrete-time signal $X(t)$, for $t = 0, \pm 1, \pm 2, \pm 3, \dots$, contains a finite-strength additive sine-wave component with frequency α if the Fourier coefficient associated with α is not zero [22]. If the sine-wave component is of the form

$$a \cos(2\pi\alpha t + \theta) \quad \alpha \neq 0 \quad (3.4)$$

then, the Fourier coefficient

$$C_x^\alpha \equiv \langle X(t) e^{-j2\pi\alpha t} \rangle = \lim_{Z \rightarrow \infty} \frac{1}{2Z+1} \sum_{t=-Z}^Z X(t) e^{-j2\pi\alpha t} \quad (3.5)$$

must not be zero, where $\langle \cdot \rangle$ indicates the time averaging operation. In this case, the power spectral density (PSD) of $X(t)$ includes a spectral line at $f = \alpha$ and its image $f = -\alpha$. Signals such as $X(t)$ containing spectral lines in the PSD are said to exhibit first-order periodicity, with frequency α . More precisely, a signal is said to exhibit first-order periodicity if and only if the PSD of the signal contains spectral lines.

A second form of periodicity that is often encountered is second-order periodicity. A signal $X(t)$ is said to exhibit second-order periodicity if and only if the PSD of the delay-product signal $X(t - \tau_1) X(t - \tau_2)$ for some delays τ_1 and τ_2 contains spectral lines at some nonzero frequencies α [28].

Hidden Periodicity in Cyclostationary Signals

If the sine-wave component of Equation (3.4) above is weak relative to $X(t)$, it might not be evident from visual inspection of $X(t)$ that it contains a periodic component, and therefore, it is said to contain hidden periodicity. However, because of the associated spectral lines, hidden periodicity can be detected and in some cases exploited.

This study is concerned with signals that contain more subtle types of hidden periodicity than just the first-order type. That is, the concern is with signals that give rise to spectral lines not in the PSD of the signal, but in the PSD of the signal converted into one with first-order periodicity by a nonlinear time-invariant transformation. In particular, the focus is on the types of hidden periodicity that can be converted into the first-order type by a quadratic transformation.

An Example of Hidden Periodicity in Cyclostationary Signals

An example of a signal containing subtle type of hidden periodicity is an amplitude modulated (AM) signal [4]. Let $r(t)$ be a random low-pass signal, a typical of such is a low-pass filtered thermal noise with the PSD $S_r(f)$ as shown in Figure 3.1(a). It should be noted that, as depicted in the figure, $S_r(f)$ contains no spectral lines at $f = 0$. This is due to the fact that for such signals, the dc component (the Fourier coefficient associated with $f = 0$) is zero, i.e.,

$$C_r^0 = \langle r(t) \rangle = 0 \quad (3.6)$$

Figure 3.1 - (a) Power Spectral Density (PSD) of a Low-Pass Signal. (b) PSD of an AM Signal. (c) PSD of a Squared Low-Pass Signal. (d) PSD of a Squared AM Signal.

If $r(t)$ is used to modulate the amplitude of a sine-wave, the AM signal, $A(t)$, is then

$$A(t) = r(t) \cos(2\pi f_0 t) \quad (3.7)$$

The PSD of the AM signal, $S_A(f)$, is shown in Figure 3.1(b) and is given by

$$S_A(f) = \frac{1}{4} S_r(f + f_0) + \frac{1}{4} S_r(f - f_0) \quad (3.8)$$

$S_A(f)$, as shown in Figure 3.1(b), contains no spectral lines and this is due to $S_r(f)$ itself not having spectral lines, as previously indicated. If we now square $A(t)$, we obtain

$$B(t) = A^2(t) = r^2(t) \cos^2(2\pi f_0 t) \quad (3.9)$$

Using the identity

$$\cos^2(2\pi f_0 t) = \frac{1}{2} + \frac{1}{2} \cos(4\pi f_0 t) \quad (3.10)$$

we have

$$B(t) = \frac{1}{2} [r^2(t) + r^2(t) \cos(4\pi f_0 t)]$$

or

$$B(t) = \frac{1}{2} [R(t) + R(t) \cos(4\pi f_0 t)] \quad (3.11)$$

where

$$R(t) = r^2(t) \quad (3.12)$$

Since $R(t)$ is non-negative (and not equal to zero since $r(t)$ is not equal to zero), its dc value must be positive, i.e., the sine-wave component at $f = 0$ ($\alpha = 0$), as previously defined by Equation (3.5) and given by

$$C_R^0 = \langle R^2(t) \rangle \quad (3.13)$$

is greater than zero. Consequently, the PSD of $R(t)$, $S_R(t)$, contains a spectral line at $f = 0$, as indicated in Figure 3.1(c).

The PSD for $B(t)$ is given by

$$S_B(f) = \frac{1}{4} [S_R(f) + \frac{1}{4} S_R(f + 2f_0) + \frac{1}{4} S_R(f - 2f_0)] \quad (3.14)$$

and, as shown in Figure 3.1(d), contains spectral lines at $f = \pm 2f_0$ as well as at $f = 0$.

Thus, by performing a quadratic transformation (the square) on $A(t)$ we have converted the hidden periodicity resulting from the sine-wave factor $\cos(2\pi f_0 t)$ into first-order periodicity with associated spectral lines in the PSD. The existence, and therefore the detection, of the associated spectral lines can be used to signal the presence of $A(t)$.

The transformation into first-order periodicity described above is easily seen if $r(t)$ is a random binary sequence that switches back and forth between $+1$ and -1 , since we then have

$$R(t) = 1$$

and therefore, $B(t)$ in Equation (3.9) is a periodic signal given by

$$\begin{aligned} B(t) &= \frac{1}{2} [R(t) + R(t) \cos(4\pi f_0 t)] \\ &= \frac{1}{2} + \frac{1}{2} \cos(4\pi f_0 t) \end{aligned} \tag{3.15}$$

It is important to note that the results of Equation (3.15) are obtained also when $r(t)$ is of the Welts code form, as will be discussed later. This is because the Welts code is a pseudo-noise sequence that also switches back and forth between $+1$ and -1 .

CHAPTER 4

CODED WAVEFORMS

Basic Radar Concepts

Radar detects the presence of objects by detecting reflected radio waves (frequencies). Objects reflect radio frequencies in a similar manner as they do light [23]. Radio waves and light are similar in the sense that they are both flow of electromagnetic energy. The only difference between the two is that light has much higher frequencies. The reflected radio frequencies are scattered in all directions. Object detection occurs when a significant portion of it is scattered back in the direction from which it originally emanated, i.e., back in the direction of the radar receiver.

A radar system generally consists of five elements [23]: a transmitter, a receiver tuned to the transmitter's frequency, two antennas, and a display device. One of the antennas is for transmission and one is for reception, however, the transmitter and receiver normally share a common antenna (Figure 4.1). The process of detecting the presence of a target begins by the antenna radiating the radio frequencies generated by the transmitter. In the mean time, the reflected frequencies (target echoes) is picked up by the antenna and passed to the receiver. If an echo is detected by the receiver, target detection and location is noted on the radar display device.

Figure 4.1 - Elements of a Simple Radar.

To prevent the radar transmitter from interfering with the reception of the reflected signal, the radio frequencies are usually transmitted in pulses (Figure 4.2), and the receiver is turned off during transmission.

Figure 4.2 - Typical Transmitted Radar Radio Frequency Pulses.

In order to detect a target, the radio frequency beam is swept in a systematic manner through the terrain in which targets are expected to be present. The target will not be detected unless its echoes are strong enough to be discerned above background noise. This is accomplished by allowing detection confirmation to occur only when target echoes signal level exceeds a predetermined threshold. It should be noted that the strength of a target's echoes is inversely proportional to the target's distance to the fourth power ($1/R^4$) [8]. The range to the target can be estimated by utilizing the time difference between radar pulse transmission and reception. The distance to the target is calculated as the speed of the pulse (speed of light) multiplied by half the time difference between transmission and detection of the reflected pulse.

Uncoded Pulsed Radar

In a traditional uncoded pulsed radar system, the transmitted radio frequencies are in the form of one single rectangular pulse of width T . The receiver would typically be a correlation receiver, which compares the received signal to the transmitted signal. A rectangular pulse and its autocorrelation is shown in Figure 4.3. Although a returned pulse would be attenuated and dispersed, the correlation output of the receiver would still be of similar form to the autocorrelation of the transmitted pulse [36].

Figure 4.3 - An Uncoded Pulse Waveform and its Autocorrelation.

Pulse Compression

Pulse compression is a radar technique used to reduce pulse detectability by enemy receivers (basics of pulse compression can be found in *Gross, et al* [5].) Pulse compression is demonstrated in Figure 4.4. In this technique, a radar pulse of duration T is subdivided before transmission into N sub-pulses each of width τ . The Subdivided pulse is compressed in that performing the autocorrelation at the receiver “compresses” the transmit pulse back into a narrow pulse [5].

The subdivided pulses are typically of equal duration and each has a particular phase. The bandwidth, B , of this waveform of N sub-pulses is roughly the bandwidth of an individual sub-pulse. Each sub-pulse is referred to as a chip with the chip rate being $1/\tau$. The chip rate is the frequency of occurrence of each sub-pulse.

Figure 4.4 - Pulse Compression. (a) One Transmit Pulse with Higher Peak Power level. (b) A Set of Sub-Pulses with Lower Peak Power Level.

Pulse sub-division achieves the benefits of high range-resolution without the need to resort to a short pulse [8]. This technique of using a long modulated pulse to obtain the resolution of a short pulse, but with the energy of a long pulse, reduces peak power levels by distributing power overtime. The purpose of this power distribution technique is to reduce pulse detectability by enemy radar.

A quantity that is associated with pulse compression is the pulse compression ratio [5]. It is a measure of the amount of pulse compression. Pulse compression ratio is defined as T/τ or BT where B , as was defined earlier, is the bandwidth of the pulse. The

amount of compression achieved is dependent upon the type of coded pulse used. The limitation on the compression ratio (T/τ) is determined by the chip rate, and therefore, is approximately determined by the coded pulse bandwidth B .

Waveform Coding

Discrete waveform coding, or simply waveform coding, is a radar pulse compression technique. Discrete coded waveforms, in general, embody ordered sequences that are impressed, at fixed time intervals, on the amplitude, frequency, or phase of a continuous-wave (cw) carrier [11]. The general description for discrete coded waveforms is given by

$$\begin{aligned} \psi(t) &= \sum_{n=1}^N a_n p_n(t) \cos[(\omega_0 + \omega_n)t + \theta_n], & 0 \leq t \leq N\tau \\ &= 0, & \text{elsewhere} \end{aligned} \tag{4.1}$$

where $n = 1, 2, \dots, N$, and $p(t)$ is a unit height pulse of fixed duration τ such that

$$\begin{aligned} p_n(t) &= p(t - [n - 1]\tau) = 1, & (n - 1)\tau \leq t \leq n\tau \\ &= 0, & \text{elsewhere} \end{aligned} \tag{4.2}$$

ω_0 in Equation (4.1) is the carrier frequency and a_n , ω_n , θ_n are respectively the amplitude, frequency, and phase modulation impressed on the cw carrier. It should be noted at this point that this study is concerned with discrete coded waveforms of the phase-modulated type, i.e., when only phase is impressed upon the cw carrier to produce θ_n .

Binary Phase Codes

Binary phase coding, often referred to simply as binary coding or phase coding, is a form of discrete waveform coding in which the phase of each sub-pulse is made either 0 or π resulting in the sub-pulses having amplitude of either +1 or -1 [5]. An example of a phase coded waveform and its corresponding cw signal is shown in Figure 4.5.

Figure 4.5 - Binary Coded Signals. Top – Modulating Binary Code. Bottom – Corresponding Phase Coded cw Signal.

Phase coded waveforms are obtained from the general description of Equation (4.1) by setting $a_n = 1$ and deleting ω_n (i.e., eliminating amplitude and frequency modulation components.) The resulting equation is

$$\begin{aligned}\psi(t) &= \sum_{n=1}^N p_n(t) \cos(\omega_o t + \theta_n), & 0 \leq t \leq N\tau \\ &= 0, & \text{elsewhere}\end{aligned}\tag{4.3}$$

where $\theta_n = 0, \pi$.

Use of phase coded waveforms in radar results in significant pulse compression ratio improvement when compared to the use of other coded waveforms of the same length T . Improved pulse compression ratio translates into the autocorrelation of the coded waveform having a much narrower central peak, resulting in the signal reflected to the radar receiver being easier distinguished from noise. As mentioned earlier, the amount of compression achieved is dependent upon the type of code used to modulate the cw. The limitation on the compression ratio (BT) is approximately determined by the coded waveform bandwidth B , and since phase coded waveforms have large bandwidths compared to other coded waveforms, this results in phase coded waveforms having superior compression ratios [5].

An example of phase coded waveforms is the Barker phase code. Barker code of length 13 is shown in Figure 4.6. The unique advantage of the Barker code is that the time side-lobes of its autocorrelation function are all of equal height, a characteristic making reflected waveforms easier to distinguish by receivers. The disadvantage of the Barker code is that there exist no more than nine possible sequences of the code (listed in

Figure 4.6 - A Barker Coded Waveform of Length 13.

Figure 4.7) with the longest code only having 13 elements [5]. Since pulse compression ratio is inversely proportional to the number of elements in a code, use of the Barker code is severely restrictive. As can be seen in Figure 4.8, the time side-lobes of the autocorrelation of the longest possible Barker code are of equal height and are restricted to $1/N = 1/13$ of the main-lobe amplitude.

Code Length	Code Elements
2	+ - , + +
3	+ + -
4	+ + - + , + + + -
5	+ + + - +
7	+ + + - - + -
11	+ + + - - - + - - + -
13	+ + + + + - - + + - + - +

Figure 4.7 - Barker Codes

Figure 4.8 - The Autocorrelation of a Barker Coded Waveform of Length 13.

Complementary Phase Codes

There exist a class of binary phase codes known as complementary codes. Complementary codes are codes in which a pair of equal-length codes have the property that the time side-lobes cancel when one algebraically adds the autocorrelation functions of the pair of codes [5]. Code A and code B, for example, are said to be a complementary pair if each side-lobe in the autocorrelation function of code A is the negative of the corresponding side-lobe in the autocorrelation of code B. Only the side-lobes and not the peaks of one are the negative of the other, and therefore, the sum yields a larger peak and no side-lobes. A pair of complementary codes A and B, each 8 chips long, is shown in Figure 4.9. Figure 4.10 shows the corresponding complementary autocorrelation

Figure 4.9 - Complementary Codes A and B.

Figure 4.10 - Autocorrelation Functions for Complementary Codes A and B.

Figure 4.11 - Sum of the Autocorrelation Functions of Codes A and B.

functions. The composite autocorrelation function after addition is shown in Figure 4.11. It is clearly seen how the time side-lobes complement each other and result in total cancellation. However, in practice, noise and Doppler shifts will cause the autocorrelation functions of the complementary set to lose their perfectly complementary property resulting in the side-lobes not being completely cancelled [5].

There are a number of classes of complementary code sets, each with different autocorrelation functions and performance properties. Two different complementary code sets will be discussed next, the Golay code and the Welty code.

Golay Codes

Marcell Golay developed a method to generate a complementary pair of codes [18]. Golay defined his complementary pair as a pair of codes with one code having the same number of like elements separated by a given spacing as the other code has dislike elements separated by the same spacing. For example, the sequence (1,1,-1,1) has two dislike pairs of adjacent elements and the sequence (1,-1,-1,-1) has two like pairs of adjacent elements. Also, there is one dislike pair of elements separated by one element in the first sequence and there is one like pair separated by one element in the second sequence. These two sequences are a Golay complementary pair.

A Golay complementary pair of length 2^N can be generated by combining a Golay complementary pair of length 2^{N-1} in the following manner [5]. Let a and b represent the two sequences of the Golay complementary pair of length 2^{N-1} . A new Golay complementary pair of length 2^N represented by c and d are generated by the following

relation: $c=(a, b)$ and $d=(a, \underline{b})$ where the comma indicates the second is appended to the first and the underline indicates the binary complement of the sequence, replacing 1 with -1 and -1 with 1. For example, if $a=(1,1,-1,1)$ and $b=(1,-1,-1,-1)$ are a complementary pair of length 4, then the complementary pair of length 8 would be $c=(1,1,-1,1,1,-1,-1,-1)$ and $d=(1,1,-1,1,-1,1,1,1)$. This process could be repeated as many times as needed to produce a pair of codes with lengths equal to any power of two. Additional complementary pairs can be produced by applying certain transformations. This set of transformations when applied changes one or both of the sequences but leaves their lengths and complementary properties unchanged.

There are four transformations that produce a new Golay complementary pair without affecting their complementary property [5]. The first is to simply reverse the order of the bits in one or both of the sequences. The second transformation is to replace one or both of the sequences with their binary complements. The third transformation is to replace alternate elements in both of the sequences with their binary complements. An example of this transformation would be to replace the complementary set of sequences $(1,-1,1,1,1,-1,-1,-1)$ and $(1,-1,1,1,-1,1,1,1)$ with $(-1,-1,-1,1,-1,-1,1,-1)$ and $(1,1,1,-1,-1,-1,1,-1)$. Notice that the odd positioned element in the first sequence were altered and the even positioned elements in the second sequence were altered. The last transformation is to replace odd positioned elements in both sequences. For example, the sequences $(1,1,-1,1)$ and $(1,-1,-1,-1)$ could be replaced with the sequences $(-1,1,1,1)$ and $(-1,-1,1,-1)$. It should be noted that any combination of these transformations could be applied in any order.

Use of Golay code is desirable because their complementary property results in ideal pulse compression characteristics, and because of their ease of generation. An example of a 64 chip Golay code is shown in Figure 4.12.

Figure 4.12 - A 64 Element Golay Coded Waveform and its Spectral Contents.

Walti Codes

George Walti also created a set of binary codes that have the complementary property [19]. Walti's code set consists of 2^N codes of length 2^N . Each code in the set is labeled according to the form D_j^k , where the length of the code is 2^k , and j is an integer from 0 to 2^k-1 that determines the specific code in the set. A set of Walti codes of length 2^N is generated from the set of Walti codes of length 2^{N-1} .

Walti code sets of any size can be generated from the fundamental set of Walti codes $D_0^1=(1,1)$ and $D_1^1=(1,-1)$ [5]. The two codes, D_0^1 and D_1^1 , are partitioned into halves as follows: $D_0^1=(w,x)$ and $D_1^1=(y,z)$. The set of four codes of length four are generated as follows: $D_0^2=(w,x,w,\underline{x})$, $D_1^2=(w,x,\underline{w},x)$, $D_2^2=(y,z,y,\underline{z})$ and $D_3^2=(y,z,\underline{y},z)$ where the underlines indicate the binary complement of the sequence. The new set of four codes would then be $D_0^2=(1,1,1,-1)$, $D_1^2=(1,1,-1,1)$, $D_2^2=(1,-1,1,1)$ and $D_3^2=(1,-1,-1,-1)$. Each member of the new set of codes can then be partitioned into halves and used to generate a set of eight codes of length eight. This procedure can be repeated until the desired code length is obtained. Once the code set is obtained, a pair of codes with the complementary property is chosen from the set. The codes D_i^k and D_j^k are said to be mates if the condition $|i-j|=2^{k-1}$ is true [5]. Any two mates from the code set have the complementary property and form a complementary pair of codes. From the previous example, codes D_0^2 and D_2^2 are mates and codes D_1^2 and D_3^2 are mates. It should be noted that code D_3^2 is equal to the binary complement of code D_0^2 written in reverse order. Also, code D_2^2 is equal to code D_1^2 written in reverse order. The complementary property of a Walti code and its mate also remains unchanged under the same four transformations described by Golay. An example of a 64 chip Walti code is shown in Figure 4.13. The focus of this study will be on the Walti code.

Figure 4.13 - A 64 Element WELT Coded Waveform and its Spectral Contents.

Detectability of Features in Coded Waveforms

A disadvantage of using coded waveforms is that the transmitted signals have detectable and exploitable features. Coded waveforms achieve superior pulse compression but create new higher order features which an interceptor can use to discriminate signals from noise. Coded waveforms generate features such as chip rate, carrier frequency and frequency hop rate [5]. Therefore, a coded waveform lowers the probability of detection through pulse compression but is still detectable because of the features present in the signal. The focus of this study is to reduce the detectability of one

of the features, namely, the carrier frequency feature in a coded waveform. The coded waveform that this study will be concerned with is the Welty binary phase coded waveform.

A class of detectors, called feature detectors, that is common in the communication world is capable of detecting the higher order features present in a coded waveform. Discussion of these feature detectors follows.

CHAPTER 5

PASSIVE DETECTORS

Until recently, detection of penetrating aircraft has been accomplished by means of line-of-sight detection between aircraft and a ground-based radar system, in which illumination of aircraft by radar was necessary [5]. Now, however, detection and target locating can be accomplished without power emanation by deploying passive receiver systems.

Passive detectors are devices designed to detect the presence of emitted radiation without relying on target illumination first. Detection is achieved by the reception of active radar system's emissions. These detectors have no emissions of their own, and therefore, it is difficult to determine detector location or determine whether detection has occurred [5]. Passive detectors are also simple and inexpensive to construct, so by wide deployment, they can become an effective mean of monitoring a very large area. Passive detectors are also referred to as feature detectors because in addition to their signal detection capability they are designed to detect and exploit features in the radar signal. These features can include frequency hop rate, chip rate, carrier frequency, and slew rate. Feature detectors can detect features in a radar waveform without knowledge of the actual generating code. Even though feature detectors are used to detect, locate and identify a signal source, they cannot actually decode the content or information in the

radar signal. The three widely used feature detectors are the square-law, fourth-law, delay-and-multiply detectors [5]. Each of these detectors is discussed below.

Square-Law Detector

This detector is used to detect the radar pulse carrier frequency by means of a non-linear square-law device. The squaring operation concentrates the energy at d.c. and twice the carrier frequency. Filters centered around twice the carrier frequency can then be used to reject noise and detect the desired signal. Information on the radar coded waveform will not be available, but sufficient energy, however, will be present in the carrier frequency for it to be detected. A block diagram of the square-law detector is shown in Figure 5.1. The incoming radar signal is first selected and amplified by a radio-frequency (RF) filter tuned to the desired carrier frequency. Next, a frequency converter comprised of a multiplier (mixer) and a local oscillator translates the RF output down to an intermediate-frequency (IF) band that is then filtered out by an IF filter. The IF output is then fed to the main component of the detector that is comprised of a squaring device. In the remaining detector stages, the signal is detected using envelope detector, integrated, and a decision whether detection occurred is then made.

Figure 5.1 - Block Diagram of the Square-Law Detector.

Fourth-Law Detector

This detector is also used to detect the radar pulse carrier frequency by means of a non-linear transformation device. The fourth-order operation concentrates the energy at d.c. and four times the carrier frequency. Filters can then again be used to reject noise and detect the desired signal. Information on the radar coded waveform also will not be available, but sufficient energy will be present in the carrier frequency for it to be detected. A block diagram of the fourth-law detector is shown in Figure 5.2. The components of the detector are essentially the same as the ones in the square-law detector, except for the non-linear transformation device. In order to produce a fourth order transformation effect, in this case, the squaring operation is performed twice.

Delay-and-Multiply Detector

This detector is used to detect the chip rate associated with the radar coded waveform. The delay and multiply operation preserves chip rate information and mixes the carrier up to twice the frequency [5]. The preservation of chip rate information is achieved by selecting the delay built into the detector to be one half of the expected coded waveform chip width. This will result in energy concentration at the chip rate fundamental frequency, in addition to energy concentration at twice the carrier frequency. Next stage noise rejection filters, in this case, should be centered around the expected

Figure 5.2 - Block Diagram of the Fourth-Law Detector.

Figure 5.3 - Block Diagram of the Delay and Multiply Detector.

chip rate fundamental frequency and not twice the carrier frequency. A block diagram of the delay-and-multiply detector is shown in Figure 5.3. The components of the detector are again essentially the same as the ones in the square-law detector, except for the non-linear transformation device. In this case, a delay τ is added before multiplying the signal by itself.

CHAPTER 6

PASSIVE DETECTION ENERGY SUPPRESSION

Use of coded waveforms, as previously mentioned, reduces pulse detectability by reducing peak power levels in the transmitted signal. However, the drawback of using coded waveforms is that the transmitted signal has detectable and exploitable features such as carrier frequency, chip rate, and frequency hop rate [5]. The focus of this study is on reducing the detectability of the carrier frequency feature in phase coded waveforms, specifically, on reducing the detectability of carrier frequency feature in a Welts coded cw waveform.

Simulation of the Welts Coded Waveform

Simulation of a complementary pair of Welts coded cw waveforms is performed using MATLAB routines. The routines used are listed in Appendices A to L. Beginning with the initial complementary pair $a=[1,-1,1,1]$ and $b=[-1,-1,-1,1]$, the routines first generate a complementary set of Welts codes of length 16 chips. The codes are then sampled, with each chip having 64 samples. The sample frequency was chosen to be

192 MHz. The chip frequency (i.e., $\frac{1}{\tau}$, where τ is the chip width) was selected to be 3MHz. It should be noted that in order to produce an accurate Fast Fourier transform of the cw waveform in MATLAB, a complete cycle (period) of the code is generated. That is, once the Welty code is generated, the code is padded with zeros to produce a full period.

Once a set of complementary Welty codes is produced, a sampled carrier frequency is next generated. The carrier frequency, f_c , used in this study to generate the coded cw signal was chosen to be 12MHz. In order to simplify the MATLAB plots generated, the carrier frequency was chosen to be in the MHz range and not in the typical GHz range for radar RF carrier frequencies. Such simplification has no effect on the MATLAB results obtained in this study. Finally, The Welty coded cw waveforms are generated by multiplying the Welty codes with the carrier frequency. The coded cw waveforms are then passed through a simplified square-law feature detector. The only component in the detector (Figure 5.1) that is of significance in this research is the multiplier, or equivalently, the square transformation, and thus, the square-law feature detector was simplified and simulated in MATLAB as depicted in Figure 6.1(a). Similarly, the fourth-law and delay-and-multiply detectors were simulated as shown in Figures 6.1(b) and (c) respectively.

Figure 6.1 – (a) Square-Law Transformation. (b) Fourth-Law Transformation. (c) Delay-and-Multiply Transformation.

Generation of Spectral Lines

The presence of the carrier frequency in a coded cw signal results in the signal having cyclostationary properties. Figure 6.2(a) shows the PSD of a Welty coded (16 chips) cw signal prior to passing the signal through a square-law feature detector (i.e., prior to passing the signal through a quadratic transformation.) Figure 6.2(b), on the other hand, shows the PSD of the coded waveform after passing the waveform through the detector.

It is clearly evident from Figure 6.2(b) that passing the waveform through the detector, i.e., performing a non-linear square transformation as mandated by the

(a)

(b)

(c)

(d)

Figure 6.2 - (a) PSD of a WELTI Coded cw Waveform. (b) PSD of the WELTI Coded cw Waveform when Passed through a Squarer. (c) PSD of (a) with Magnitude in dB. (d) PSD of (b) with Magnitude in dB.

definition of cyclostationarity, results in the generation of finite-strength sine-wave components (spectral lines), and therefore, results in the waveform being a cyclostationary signal of order 2. The sine-wave components are at $f=2f_c=24$ MHz and $f=0$, as expected from Equation (3.14). Figures (c) and (d) are the plots of the PSDs in (a) and (b) with the magnitude shown in dB.

Generation of the dc spectral component in the signal spectrum is trivial in that it provides no carrier frequency information, and therefore, is of no concern in this study. Furthermore, since the signal gives rise to spectral lines not in its PSD, but in the PSD of the transformed signal, then by definition, a hidden first-order periodicity is present in the WELTI coded CW waveform. This hidden periodicity can be detected and exploited by a

Figure 6.3 - (a) PSD in dB of a WELTI Coded CW Waveform. (b) PSD of the WELTI Coded CW Waveform When Passed through a 4th-Law Detector.

simple enemy feature detector.

Hidden periodicity in the WELTI coded CW waveform can also be converted to yield spectral lines when the signal is passed through a fourth-law detector. Figures 6.3(a) and (b) show respectively the PSDs in dB of the signal before and after passing the waveform through the detector. The spectral lines, in this case, are at $f=2f_c$, $f=4f_c$, and $f=0$.

The goal of this research is to attempt to suppress the hidden periodicity present in the WELTI coded CW signal. Suppression of periodicity will be attempted by filtering out frequency components present in the actual WELTI code itself. Filtering of certain frequency components in the WELTI code will result in energy suppression in the neighborhood of twice the carrier frequency present in the square of the coded CW waveform. It should be noted that any filtering of frequency components should cause no degradation of the narrow central peak characteristic of the coded signal's autocorrelation, i.e., pulse compression ratio superiority of the WELTI coded CW signal should be preserved. It is expected that trade-off will occur between obtaining higher degree of passive detection energy suppression and retaining higher pulse compression ratio superiority.

WELTI Code Filter Design

As discussed earlier, suppression of the carrier frequency feature in the squared WELTI coded CW waveform will be attempted by suppressing frequency components in the WELTI code itself. Suppression of frequency components within a WELTI code frequency

bandwidth will result in energy suppression. The energy suppressed coded cw waveform will then be generated by modulating the carrier frequency with the new filtered Welter code.

In this section, the characteristics of the Welter code energy suppression filter will be determined. This will be accomplished by first determining the relationship between the Welter code and the unit height pulse.

A unit height pulse of fixed duration τ , shown in Figure 6.4, is given by

$$p(t) = u(t) - u(t - \tau) \quad (6.1)$$

where $u(t)$ is the unit step function. The Fourier transform of $p(t)$ can be easily derived to be

$$P(f) = \mathcal{F}\{p(t)\} = \tau e^{-j\pi f\tau} \frac{\sin(\pi f\tau)}{\pi f\tau} \quad (6.2)$$

Figure 6.4 – A Unit Height Pulse of Duration τ .

The Welts code in terms of $p(t)$, as shown in Figure 6.5, is defined as

$$w(t) = \sum_{n=1}^N a_n p(t - [n - 1] \tau), \quad a_n = \pm 1 \quad (6.3)$$

Figure 6.5 – Welts Code Consisting of Delayed Unit Height Pulses.

The Fourier transform of $w(t)$ is

$$\begin{aligned} W(f) &= \mathcal{F}\{w(t)\} = \sum_{n=1}^N a_n \mathcal{F}\{p(t - [n - 1] \tau)\} \\ &= \sum_{n=1}^N a_n \int_{-\infty}^{\infty} p(t - [n - 1] \tau) e^{-j2\pi f t} dt \end{aligned} \quad (6.4)$$

Upon integration, $W(f)$ reduces to

$$W(f) = P(f) \sum_{n=1}^N a_n e^{-j2\pi f(n-1)\tau} \quad (6.5)$$

Next, the relationship in the frequency domain between the filtered and unfiltered WELT code is expressed in terms of the WELT code's filter transfer function. Denoting the filtered WELT code as $m(t)$, the Fourier transform of $m(t)$ is then

$$M(f) = \mathcal{F}\{m(t)\} = H(f) W(f) \quad (6.6)$$

where $H(f)$ is the transfer function of the desired energy suppression filter. Substituting Equation (6.5) into Equation (6.6), we then have

$$M(f) = H(f) P(f) \sum_{n=1}^N a_n e^{-j2\pi f(n-1)\tau} \quad (6.7)$$

When we modulate a carrier with the filtered WELT code $m(t)$ we get

$$s(t) = m(t) \cos 2\pi f_c t \quad (6.8)$$

Squaring of $s(t)$ results in the formation of the unwanted spectral lines, and therefore, the square of Equation (6.8) needs to be closely examined to determine the appropriate energy filtering scheme.

The square of Equation (6.8), $s^2(t)$, can be written in the frequency domain as

$$\mathcal{F}\{s^2(t)\} = \mathcal{F}\{s(t)\} \otimes \mathcal{F}\{s(t)\} = S(f) \otimes S(f) \quad (6.9)$$

where the operation $\otimes$ denotes convolution and $S(f)$ is the Fourier transform of $s(t)$. The convolution can be expressed as,

$$R(\alpha) = S(f) \otimes S(f) = \int_{-\infty}^{\infty} S(f) S(\alpha - f) df \quad (6.10)$$

We can take the Fourier transform of Equation (6.8) to obtain

$$\begin{aligned} S(f) &= \mathcal{F}\{s(t)\} = \mathcal{F}\{m(t)\cos 2\pi f_c t\} \\ &= \frac{1}{2} [M(f - f_c) + M(f + f_c)] \end{aligned} \quad (6.11)$$

Substituting Equation (6.11) into Equation (6.10), we have

$$R(\alpha) = \frac{1}{4} \int_{-\infty}^{\infty} [M(f - f_c) + M(f + f_c)] [M(\alpha - f - f_c) + M(\alpha - f + f_c)] df$$

which can be rewritten as

$$R(\alpha) = \frac{1}{4} \int_{-\infty}^{\infty} [M(f - f_c) + M(f + f_c)] [M(\{f_c + \alpha\} - f) + M(\{-f_c + \alpha\} - f)] df \quad (6.12)$$

If we let X_1 and X_2 represent, respectively, the first and second bracketed terms in the above equation, then X_1 and X_2 can be depicted as shown in Figure 6.6(a). The integral of the product $[X_1][X_2]$, i.e., $R(\alpha)$, is depicted in Figure 6.6(b). From Figure 6.6(b), it is

Figure 6.6 – (a) Multiplicative Terms in the Equation For the Fourier Transform of the Square of the Filtered WELT Coded CW Waveform. (b) Approximation of the Entire Spectrum of the Filtered WELT Coded CW Waveform.

easily concluded that $R(\alpha)$ may be seen as the sum of three terms. If we denote the left spectrum of $R(\alpha)$ by $X_L(\alpha)$, the center spectrum by $X_C(\alpha)$, and the right spectrum by $X_R(\alpha)$, then $R(\alpha)$ can be viewed as

$$R(\alpha) = X_L(\alpha) + X_C(\alpha) + X_R(\alpha) \quad (6.13)$$

The frequency components at the output of the square-law detector that are of interest, i.e., that are to be suppressed, are those in the neighborhood of $\alpha \approx \pm 2f_c$. Since $X_L(\alpha)$ is the mirror image of $X_R(\alpha)$, then for the purpose of problem simplification and without loss of generality, frequency components of interest in only the right positive spectrum $X_R(\alpha)$ are considered, and therefore, the frequency components that will be examined are those in the neighborhood of $\alpha \approx 2f_c$. Expanding Equation (6.12), we get

$$\begin{aligned} R(\alpha) = \frac{1}{4} \int_{-\infty}^{\infty} [& M(f - f_c) M(\{f_c + \alpha\} - f) + M(f - f_c) M(\{-f_c + \alpha\} - f) \\ & + M(f + f_c) M(\{f_c + \alpha\} - f) + M(f + f_c) M(\{-f_c + \alpha\} - f)] df \end{aligned} \quad (6.14)$$

Substituting $\alpha = 2f_c$ in the above equation, we have

$$\begin{aligned} R(2f_c) = \frac{1}{4} \int_{-\infty}^{\infty} [& M(f - f_c) M(3f_c - f) + M(f - f_c) M(f_c - f) \\ & + M(f + f_c) M(3f_c - f) + M(f + f_c) M(f_c - f)] df \end{aligned} \quad (6.15)$$

and since we are only considering $X_R(\alpha)$, the only term then having significant contribution upon integration of Equation (6.15) is the second term, $M(f - f_c) M(f_c - f)$. All other products in the integral are nearly zero. Therefore, $R(2f_c)$ reduces to

$$X_R(2f_c) \approx \frac{1}{4} \int_{-\infty}^{\infty} M(f - f_c) M(f_c - f) df \quad (6.16)$$

and for any α near $2f_c$, $X_R(2f_c)$ becomes

$$X_R(\alpha) \approx \frac{1}{4} \int_{-\infty}^{\infty} M(f - f_c) M(-f_c + \alpha - f) df \quad (6.17)$$

Substituting $M(f)$ of Equation (6.7) into Equation (6.17), $X_R(\alpha)$ then approximately becomes

$$X_R(\alpha) \approx \frac{1}{4} \int_{-\infty}^{\infty} [H(f - f_c) P(f - f_c) \sum_{n=1}^N a_n e^{-j2\pi(f-f_c)(n-1)\tau}] \\ [H(-f - f_c + \alpha) P(-f - f_c + \alpha) \sum_{n=1}^N a_n e^{j2\pi(-f-f_c+\alpha)(n-1)\tau}] df$$

or

$$X_R(\alpha) \approx \frac{1}{4} \int_{-\infty}^{\infty} H(f - f_c) P(f - f_c) H(-f - f_c + \alpha) P(-f - f_c + \alpha) \\ \left[\sum_{n=1}^N a_n e^{-j2\pi(f-f_c)(n-1)\tau} \right] \left[\sum_{n=1}^N a_n e^{j2\pi(-f-f_c+\alpha)(n-1)\tau} \right] df \quad (6.18)$$

For maximum energy suppression in the Welts coded cw signal, Equation (6.18) must be minimized. The summations in the brackets are the phase terms for the Welts code and should not be altered in any manner in order to preserve the delay characteristic of each of the pulses in the code. Therefore, what only needs to be examined in Equation (6.18) is the product of the Fourier transform of the desired Welts code filter with the Fourier transform of the unit height pulse. Frequency components of substantial energy contribution in the Fourier transform of a pulse, that is, in the spectrum of the sinc function, are those located within the frequency spectrum's main-lobe, in particular, those in the frequency neighborhood centered around the main-lobe peak. Hence, for maximum energy suppression, the filter chosen should be a notch filter. A notch filter is easily constructed and its bandwidth can be easily varied as needed, and when filtering a frequency spectrum of the Welts code's type, a notch filter will produce maximum energy suppression results. Figure 6.7 depicts the frequency spectrum of the Welts code in the top plot and the same spectrum after filtering with a notch filter in the bottom plot.

Three various bandwidths of the energy suppression notch filter will be employed. The bandwidths will be chosen in relation to the first null of the Welts code sinc spectrum. Thus, the notch filter bandwidths will be chosen as: $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$,

$\frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$, $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, where the first nulls of the Welts code sinc spectrum are at $\pm \left(\frac{1}{\tau}\right)$, or

at $\pm 3\text{MHz}$. The center frequency of the three notch filters is the dc frequency. It is important to note that once the Welts code is filtered, peak power in the code will be restored through appropriate normalization. This normalization is necessary in order to accurately compare energy suppression. Energy suppression will be determined by

Figure 6.7 – Spectrum of Unfiltered and Notch Filtered WELTI Code.

comparing average power within a frequency bandwidth in the square of the filtered and unfiltered WELTI coded cw waveform. The squared cw waveform bandwidth within which peak power is to be compared is chosen to be equal to the bandwidth of the main-lobe in the WELTI code spectrum, that is $\left(\frac{2}{\tau}\right)$, or 6 MHz, and will be centered around twice the carrier frequency, or around 24 MHz. As was discussed in Chapter 2 and can be concluded from Equation (2.14) (average power is defined as the autocorrelation $R(\tau)$ when $\tau=0$), the average power within a bandwidth will be calculated by integrating the PSD over the appropriate range of frequencies.

Energy Suppression Results

As noted earlier, energy suppression results are to be compared when the Welte code notch filter employed is of three various bandwidths. Energy suppression results are determined by calculating the average power within a frequency bandwidth in the square of the filtered and unfiltered Welte coded cw waveform. That cw waveform bandwidth, as indicated previously, is $\left(\frac{2}{\tau}\right)$, or 6 MHz. For simplicity, this bandwidth will be referred to as the cw waveform reference bandwidth.

First the average power within the cw waveform reference bandwidth was determined when the Welte code was unfiltered. That average power was measured to be 12.4 mW, or equivalently, 10.9 dBm. This average power remains the same for the three Welte code notch filter bandwidth cases. Next, the Welte code was filtered by employing each of the three various notch filter bandwidths. Peak power normalization was then performed on the filtered Welte code in order to restore peak power's original value. The filtered normalized Welte code was then multiplied by the carrier frequency to obtain the energy suppressed coded cw waveform. The energy suppressed cw waveform was then squared and average power was measured within the 6 MHz reference bandwidth of the suppressed squared cw waveform. This average power measurement was repeated for the three Welte code notch filter bandwidth cases.

Figure 6.8 shows the unfiltered 16 chip Welte code in the top and the filtered code in the bottom. The energy suppression notch filter bandwidth in this case is $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$, or

Figure 6.8 – Top: Unfiltered Welte Code. Bottom: Filtered Welte Code, $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$.

60 kHz. It is clearly evident from the figure that the filtered Welte code remained essentially unchanged and that filtering resulted in only slight code distortion. The amplitude of the code seems to have been slightly reduced.

The composite autocorrelations for the unfiltered and filtered Welte codes were then compared. The top plot in Figure 6.9 shows the composite autocorrelation for the unfiltered Welte code and the bottom plot shows that for the filtered code. As seen from the bottom plot, the shape of the central peak remains virtually unchanged and the time side-lobes still nearly cancelled. This clearly indicates that the complementary characteristics of the Welte code pair were retained. Therefore, we can easily conclude that for this choice of notch filter bandwidth, filtering caused only slight degradation of the Welte code's composite autocorrelation and that higher pulse compression ratio superiority was retained.

Figure 6.9 – Top: Composite Autocorrelation of the Two Unfiltered Complementary WELTI Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$.

The PSD of the square of the WELTI coded cw waveform was next plotted and examined. Figure 6.10 shows the PSD of the square of the unsuppressed WELTI coded cw waveform (top plot) and the PSD of the square of the energy suppressed coded cw waveform (bottom plot.) As seen from the bottom plot, the spectral line at $f=24$ MHz was suppressed in that the PSD peak value was noticeably reduced. The PSD peak value was reduced by a factor of approximately 2/5 when compared to the peak value in the top plot. The average power within the cw waveform reference bandwidth was measured to be 7.90 mW, or equivalently, 8.98 dBm. This is a reduction of approximately 36.6 % from the average power for the unsuppressed cw waveform.

Figure 6.10 – Top: Spectra of the Square of the Unsuppressed Wेल्टी Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a Wेल्टी Code Filter of $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$.

Next, filtering of the Wेल्टी code was performed using the second energy suppression notch filter bandwidth of $\frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$, or 120 kHz. Figure 6.11 shows the unfiltered Wेल्टी code in the top and the filtered code in the bottom. For this notch filter bandwidth case, as seen from the bottom plot, filtering resulted in slight code alteration. Filtering seems to have resulted in a slight amplitude reduction for approximately the first half of the code's duty cycle. The overall shape of the Wेल्टी code, nonetheless, was maintained.

Figure 6.11 – Top: Unfiltered Weltsi Code. Bottom: Filtered Weltsi Code, $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$.

The composite autocorrelations for the unfiltered and filtered Weltsi codes again for this notch filter bandwidth case were compared. The top plot in Figure 6.12 shows the composite autocorrelation for the unfiltered Weltsi code and the bottom plot shows that for the filtered code. It is seen from the bottom plot that the shape of the central peak for this choice of bandwidth remains nearly unchanged. The time side-lobes still virtually cancelled, even though, the existence of more visible side-lobe residue renders cancellation less than that obtained for the 60 kHz bandwidth case. Therefore, for this 120 kHz bandwidth case, it still can be easily concluded that filtering caused no severe

Figure 6.12 – Top: Composite Autocorrelation of the Two Unfiltered Complementary Welfi Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$.

degradation of the Welfi code's composite autocorrelation and that higher pulse compression ratio superiority was still retained.

The PSD of the square of the Welfi coded cw waveform for this notch filter case again was plotted and examined. Figure 6.13 shows the PSD of the square of the unsuppressed Welfi coded cw waveform (top plot) and the PSD of the square of the energy suppressed coded cw waveform (bottom plot.) As evident from the bottom plot, the spectral line at $f=24$ MHz was further suppressed in that more reduction of the peak value resulted when the plot is compared to that for the 60 kHz notch filter bandwidth case in the bottom of Figure 6.10. The PSD peak value in the bottom of Figure 6.13 was

Figure 6.13 – Top: Spectra of the Square of the Unsuppressed WELTI Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a WELTI Code Filter of $BW = \frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$.

reduced by a factor of approximately 3/5 when compared to the peak value for the unsuppressed waveform in the top plot. The average power within the cw waveform reference bandwidth for this notch filter bandwidth case was measured to be 5.40 mW, or equivalently, 7.32 dBm. This is a reduction of approximately 56.5 % from the average power for the unsuppressed cw waveform.

Finally, filtering of the WELTI code was performed using the third energy suppression notch filter bandwidth of $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, or 240 kHz. Figure 6.14 shows the unfiltered WELTI code in the top and the filtered code in the bottom. For this notch filter

Figure 6.14 – Top: Unfiltered Welts Code. Bottom: Filtered Welts Code, $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$.

bandwidth case, as seen from the bottom plot, filtering resulted in highly noticeable code distortion. Filtering seems to have resulted in even more amplitude reduction than that resulted in the 120 kHz notch filter bandwidth case, and the amplitude reduction in this case appears to be for the entire code's duty cycle.

The composite autocorrelations for the unfiltered and filtered Welts codes for this notch filter bandwidth case were again compared. The top plot in Figure 6.15 shows the composite autocorrelation for the unfiltered Welts code and the bottom plot shows that for the filtered code. Again, it is seen from the bottom plot that the shape of the central peak for this choice of bandwidth remains virtually unchanged. However, no complete side-lobe cancellation is obtained in this case also. Side-lobe residue is more evident in

Figure 6.15 – Top: Composite Autocorrelation of the Two Unfiltered Complementary Welfi Codes. Bottom: Composite Autocorrelation of the Two Complementary Codes when the Codes are Filtered with a $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$.

this bandwidth case than it was in the 120 kHz bandwidth case. Nonetheless, severe degradation of the Welfi code's composite autocorrelation still did not result, and the higher pulse compression ratio superiority of the code was still retained.

Again, the PSD of the square of the Welfi coded cw waveform for this notch filter case was plotted and examined. Figure 6.16 shows the PSD of the square of the unsuppressed Welfi coded cw waveform (top plot) and the PSD of the square of the energy suppressed coded cw waveform (bottom plot.) Again, as evident from the bottom plot, the spectral line at $f=24$ MHz was further suppressed in that more reduction of the PSD peak value resulted when the plot is compared to that for the 120 kHz notch filter bandwidth case in the bottom of Figure 6.13. The PSD peak value in the bottom of

Figure 6.16 – Top: Spectra of the Square of the Unsuppressed Welter Coded cw Waveform. Bottom: Spectra of the Square of the Suppressed Coded cw Waveform. Suppression is with a Welter Code Filter of $BW = \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$.

Figure 6.16 was reduced by approximately 4/5 when compared to the peak value for the unsuppressed waveform in the top plot. The average power within the cw waveform reference bandwidth for this notch filter bandwidth case was measured to be 2.80 mW, or equivalently, 4.47 dBm. The average power reduction in this case is approximately 77.4% from the average power for the unsuppressed cw waveform.

To obtain a better understanding of the relationship between energy suppression in the cw waveform reference bandwidth and the Welter code notch filter bandwidth, a plot was generated depicting reference bandwidth percentage average power reduction versus notch filter bandwidth. Figure 6.17 shows the plot. To obtain a generalized plot,

the bandwidth scale is plotted in its normalized form where frequency was divided by the bandwidth of the main-lobe in the WELTI code spectrum, that is, the bandwidth axis was divided by $\left(\frac{2}{\tau}\right)$, or 6 MHz. The notch filter bandwidth was varied from 0 up to 600 kHz in increments of $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$, or 60 kHz. In the equivalent normalized scale, the normalized filter bandwidth, B_n , was varied from 0 up to 0.1 in increments of $\frac{1}{100}$. It should be noted that the average power percentage reductions for the three bandwidth

Figure 6.17 – Percentage Average Power Reduction in the WELTI Coded cw Waveform Reference Bandwidth Versus the Normalized WELTI Code Notch Filter Bandwidth, B_n .

cases discussed earlier have been re-plotted and denoted on the curve with small circles.

As can be seen from the figure, the plot obtained is of the exponential type. It is apparent that reference bandwidth average power decreases exponentially up to a notch filter bandwidth of approximately 400 kHz, or equivalently, up to a normalized bandwidth of approximately 0.07. Beyond that bandwidth, there appears to be no additional power reduction, as percentage reduction remained constant at approximately 83 %.

In summary, as the WELTI code notch filter bandwidth was increased, energy suppression, accordingly, increased in that the measured average power within the WELTI coded CW waveform reference bandwidth decreased. As the notch filter bandwidth approached 400 kHz, average power reduction remained unchanged at approximately 83%. Also, as was expected, a trade off occurred between energy, or spectral line, suppression and the retention of the desirable complementary characteristics of the WELTI code's autocorrelation. As the WELTI code notch filter bandwidth was increased, the time-side lobes still nearly cancelled, except as the filter bandwidth approached $\frac{1}{25}$ of the bandwidth of the first null of the WELTI code sinc spectrum, or $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, where highly noticeable side-lobe residue began resulting.

Future Research

Filtering of the WELTI code was performed in this research to suppress cyclostationary spectral lines that are produced in the square of a WELTI coded CW

waveform. Filtering of the Welty code should also produce similar suppression results of cyclostationary spectral lines in a fourth-ordered Welty coded cw signal, i.e., in a cw waveform that has undergone a fourth-order non-linear transformation.

The focus of this dissertation also was on suppression of cyclostationarity in a Welty coded cw waveform when the code employed is of the Welty type. A second opportunity for further research on suppression of cyclostationarity arises when the code employed is instead of the Golay type (for Golay code discussion, see Chapter 4.) Since both Golay and Welty codes belong to the same class of codes, i.e., both are complementary phase codes, the technique of choice to be used in the suppression of cyclostationarity in a squared or fourth-ordered Golay coded cw waveform should similarly be that of filtering the Golay code itself and observing average power suppression in the transformed Golay coded cw waveform, as was done when the code employed was of the Welty type. Average power suppression results for the squared or fourth-ordered Golay coded cw waveform should be similar to those obtained in this dissertation when the code employed was of the Welty type.

In Chapter 3, two forms of periodicity in cyclostationary signals were discussed, first and second-order periodicity. First-order periodicity was defined as the arise of spectral lines in the transformed signal when the non-linear transformation $Y(t) = X(t)X(t-\tau)$ is performed on the signal with the delay τ being 0, i.e., when $Y(t) = X^2(t)$, or the non-linear transformation is simply the square of the signal. Suppression of cyclostationarity due to such periodicity was attempted in this research. An opportunity for further study that is of greater consequence is to investigate second-order periodicity

in cyclostationary signals. In this case, the delay τ in the delay-product signal $Y(t)=X(t)X(t-\tau)$ is of a nonzero value. This delay-and-multiply transformation also results in the generation of spectral lines in the transformed cyclostationary signal. The information contained in the spectral lines generated in a delayed-and-multiplied phase coded signal is the code's chip width, or equivalently, the code's chip rate. This chip rate information, as was discussed in Chapter 5, can be easily detected and exploited by an enemy delay-and-multiply feature detector.

Figure 6.18 shows a 16 chip WELT code in the top, the code with a delay of $\frac{1}{2}$ chip in the center, and the $\frac{1}{2}$ chip delayed-and-multiplied code in the bottom. The bottom plot

Figure 6.18 – Top: WELT Code. Center: $\frac{1}{2}$ Chip Delayed WELT Code. Bottom: $\frac{1}{2}$ Chip Delayed-and-Multiplied WELT Code.

indicates that the delay-and-multiply operation resulted in a code with a chip width that is half of that for the original Welter code, or equivalently, resulted in a chip rate frequency twice of that for the original Welter code, i.e., a new chip rate frequency of 6 MHz.

Figure 6.19 shows the PSD for the Welter code in the top and the PSD for the $\frac{1}{2}$ chip delayed-and-multiplied code in the bottom. In both PSD plots, only the lower part of the frequency spectrum is shown. This is to allow for closer observation and comparison of the characteristics of the spectra, in particular, the characteristics of the delay-and-multiply spectrum.

It is clearly evident from the delay-and-multiply PSD plot in the bottom of Figure 6.19 that the same frequency spectrum is periodically repeated. The plot also shows that

Figure 6.19 – Top: PSD of the Welter Code. Bottom: PSD of $\frac{1}{2}$ Chip Delayed-and-Multiplied Welter Code.

at the center of each of the repeated spectra, there is a clearly well defined sine-wave component, i.e., a clearly well defined spectral line. Furthermore, these central spectral lines, as can be seen from the plot, are separated by a frequency of 6 MHz. This frequency separation is identical to the chip rate frequency previously noted for the delayed-and-multiplied code in the bottom plot of Figure 6.18, i.e., this frequency separation is twice the chip rate for the original Welts code. This indicates that a delayed-and-multiplied operation can easily result in the determination of a Welts code's chip rate. Using a delay-and-multiply feature detector, as depicted in Figure 5.3, a delay τ that would result in obtaining spectral line frequency separation equivalent to that in the bottom of Figure 6.19 can be easily determined. Knowing the necessary delay τ would in turn allow for the determination of a Welts code's chip rate. Knowing the chip rate of a transmitted Welts code may provide an enemy the opportunity for signal exploitation. Therefore, further research may be conducted to investigate how chip rate information may be extracted from the repeated central spectral lines present in the spectrum of a delayed-and-multiplied Welts code.

CHAPTER 7

CONCLUSION

Use of binary phase coding, in contrast to use of uncoded pulsed radar signals, reduces peak power levels in transmitted signals, and therefore, reduces signal detectability by enemy radar. Use of binary phase coding also results in reflected signals that are easier distinguished from noise by the radar receiver due to superior pulse compression ratio.

A class of binary phase codes of superior characteristics is the complementary phase codes. Complementary codes possess the characteristic that when one algebraically adds the autocorrelation functions of a pair of complementary codes, the time side-lobes cancel. This autocorrelation function side-lobe cancellation enhances reflected signal distinction from noise by the receiver. The two widely known complementary codes are the Golay and Welty codes.

A disadvantage of using waveform coding is that the transmitted signal has detectable and exploitable features. These features can be detected with simple enemy passive receivers. A feature that was of particular interest in this dissertation is the carrier frequency feature. Presence of carrier frequency feature in a coded waveform gives rise to cyclostationary properties. These cyclostationary properties can be easily detected with the use of a nonlinear transformation device. Cyclostationary properties in

a coded waveform are exhibited in the form of spectral lines in the transformed signal generated using the nonlinear transformation device. This study focused on suppression of cyclostationary properties present in the WELTI coded CW waveform, more specifically, on energy suppression of spectral lines in a transformed WELTI coded CW waveform with the nonlinear transformation being the square of the waveform.

Cyclostationary spectral lines present in the PSD of the square of the WELTI coded CW signal were suppressed by filtering out frequency components present in the actual WELTI code itself. For maximum energy suppression, the filter employed was chosen to be of the notch type. The frequency components filtered out are components around the DC frequency. Three different notch filter bandwidths were employed in filtering the WELTI code, each having the DC frequency as bandwidth center frequency. The bandwidths were chosen in relation to the first null of the WELTI code sinc spectrum, where, in terms of the WELTI code's pulse width τ , the first nulls of the code sinc spectrum are at $\pm \left(\frac{1}{\tau}\right)$, or ± 3 MHz. Therefore, the notch filter bandwidths were chosen

as: $BW = \frac{1}{100} \cdot \left(\frac{2}{\tau}\right), \frac{1}{50} \cdot \left(\frac{2}{\tau}\right), \frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, or 60 kHz, 120 kHz, and 240 kHz respectively.

Spectral line energy suppression was determined by comparing average power within a frequency bandwidth in the square of the filtered and unfiltered WELTI coded CW waveform. The squared CW waveform bandwidth within which average power was compared was chosen to be equal to the bandwidth of the main-lobe in the WELTI code spectrum, that is $\left(\frac{2}{\tau}\right)$, or 6 MHz, and was centered around twice the carrier frequency, or

around 24 MHz. This bandwidth was referred to as the cw waveform reference bandwidth.

First, the average power within the cw waveform reference bandwidth was determined when the WELTI code was unfiltered. That average power was measured to be 12.4 mW, or equivalently 10.9 dBm. The WELTI code was then filtered using the first energy suppression notch filter bandwidth of 60 kHz. Filtering with the 60 kHz bandwidth resulted in slight code distortion. Also, filtering caused only slight degradation in the WELTI code's desirable complementary characteristics as the composite autocorrelation's narrow central peak was virtually unchanged and the time side-lobes still nearly cancelled. Also for this notch filter bandwidth case, the peak value (spectral line) in the PSD of the square of the cw waveform was reduced by a factor of approximately $2/5$ when compared to the peak value for the unsuppressed cw waveform. The average power within the cw waveform reference bandwidth was measured to be 7.90 mW, or equivalently 8.98 dBm. This is a reduction of approximately 36.6 % from the average power for the unsuppressed cw waveform.

Next, filtering of the WELTI code was performed using the second energy suppression notch filter bandwidth of 120 kHz. Filtering with the 120 kHz bandwidth also resulted in code distortion; nonetheless, the overall shape of the WELTI code was maintained. For this choice of bandwidth also, filtering caused no severe degradation in the WELTI code's complementary characteristics as the composite autocorrelation's narrow central peak was nearly unchanged and the time side-lobes still virtually cancelled. The peak value in the PSD of the square of the cw waveform, for this bandwidth case, was reduced by a factor of approximately $3/5$ when compared to the

peak value for the unsuppressed cw waveform. The average power within the cw waveform reference bandwidth was measured to be 5.40 mW, or equivalently 7.32 dBm. This is a reduction of approximately 56.5 % from the average power for the unsuppressed cw waveform.

Finally, filtering of the Welty code was performed using the third energy suppression notch filter bandwidth of 240 kHz. For this notch filter bandwidth case, filtering resulted in highly noticeable code distortion. As for the code's composite autocorrelation, the narrow central peak, for this choice of bandwidth also, remained virtually unchanged. Side-lobe cancellation, however, was the least for this case as more residue was evident. The peak value in the PSD of the square of the cw waveform was reduced by a factor of approximately 4/5 when compared to the peak value for the unsuppressed cw waveform. The average power within the cw waveform reference bandwidth was measured to be 2.80 mW, or equivalently 4.47 dBm. This is a reduction of approximately 77.4 % from the average power for the unsuppressed cw waveform.

In this dissertation, suppression of cyclostationarity due to first-order periodicity in Welty coded cw waveforms was attempted. An opportunity for further study is to investigate second-order periodicity in cyclostationary signals. In such case, the delay τ in the delay-product signal $Y(t) = X(t)X(t-\tau)$ is of a nonzero value. This delay-and-multiply transformation also results in the generation of spectral lines. The information contained in the spectral lines generated in a delayed-and-multiplied phase coded signal is the code's chip rate. This chip rate information can be easily detected and exploited with the use of simple enemy delay-and-multiply feature detector. Therefore, further

research may be conducted to investigate how chip rate information can be extracted from the spectral lines present in the spectrum of a delayed-and-multiplied WELT code.

APPENDIX A

COMPLEMENTARY CODE GENERATOR ROUTINE

This routine generates a pair of complementary codes. A set of an initial pair of length four is used to generate the complementary pair of codes. Each of the codes is 8 chips long. Each chip in a code is sampled 64 times.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Complementary Codes Generator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  L -- number of chips in the Complementary codes
%  a, b -- complementary codes
%  awav, bwav -- complementary codes sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L=8;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Complementary Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a=[1,-1,1,1];
b=[-1,-1,-1,1];
awav=[0];
bwav=[0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Complementary codes each of length 8
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n=log2(L/2);
```

```

for i=2:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample Complementary Codes, 64 samples per each chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:length(a),
    for j=1:64
        awav=[awav,a(i)];
        bwav=[bwav,b(i)];
    end
end

awav(:,1)=[];
bwav(:,1)=[];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Coded Waveforms
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;

t = linspace(1,8,512);

subplot(1,2,1);
plot(t,awav);
axis([min(t) max(t) -2 2]);
xlabel('Time');
ylabel('Amplitude');
title('Complementary Code A');

subplot(1,2,2);
plot(t,bwav);
axis([min(t) max(t) -2 2]);
xlabel('Time');
title('Complementary Code B');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain and plot the autocorrelation of each of the complementary codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

x1=xcorr(awav);
x2=xcorr(bwav);

t = linspace(-8,8,1023);
figure;
subplot(1,2,1);
plot(t,x1);
axis([min(t) max(t) -200 1.2*max(x1)]);
xlabel('Time');
ylabel('Amplitude');
title('Autocorrelation for Code A');

```

```

subplot(1,2,2);
plot(t,x2);
axis([min(t) max(t) -200 1.2*max(x2)]);
xlabel('Time');
title('Autocorrelation for Code B');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain and plot the sum of the autocorrelations of the complementary codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

x3=x1+x2;
figure;
plot(t,x3);
axis([min(t) max(t) -200 1.2*max(x3)]);
xlabel('Time');
ylabel('Amplitude');
title('Composite Autocorrelation');

```

APPENDIX B

RECTANGULAR PULSE GENERATOR ROUTINE

This routine generates a rectangular pulse. It also generates and plots the autocorrelation of the pulse.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Rectangular Pulse Generator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  N -- Length of Pulse
%  a -- The rectangular Pulse
%  x1 -- The Autocorrelation of the Rectangular Pulse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Rectangular Pulse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

N=1152;
a=[zeros(1,N/3) ones(1,N/3) zeros(1,N/3)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Rectangular Pulse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

t = linspace(0,3,N);
subplot(1,2,1);
plot(t,a);
```

```

axis([min(t) max(t) -0.5 1.5]);
xlabel('Time');
ylabel('Amplitude');
title('Uncoded Rectangular Pulse');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the Autocorrelation of the Rectangular Pulse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

x1=xcorr(a);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot the Autocorrelation of the Rectangular Pulse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

t = linspace(-2,2,(2*N)-1);
subplot(1,2,2);
plot(t,x1/32);
axis([min(t) max(t) -5 15]);
xlabel('Time');
title('Autocorrelation of Uncoded Pulse');

```

APPENDIX C

BARKER CODE GENERATOR ROUTINE

This routine generates a Barker code. The code is 13 elements long. Each element is sampled 64 times. The autocorrelation of the code is also generated.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Barker Code Generator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  b -- Barker code
%  L -- number of chips in the Barker code
%  Barker -- Barker code sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Barker code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

b=[1,1,1,1,1,-1,-1,1,1,-1,1,-1,1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Barker Code of 64 samples per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L=length(b);

for i=1:L,
    c=[b(i)];
    for j=1:63,
```



```

        c=[c,b(i)];
    end
    if i==1
        Barker=c;
    else
        Barker=[Barker,c];
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Barker Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
t = linspace(1,13,832);
plot(t,Barker);
axis([min(t) max(t) -2 2]);
xlabel('Time');
ylabel('Amplitude');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot the Autocorrelation of the Barker Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x1=xcorr(Barker);
figure;
t = linspace(-13,13,1663);
plot(t,x1);
axis([min(t) max(t) min(x1) 1.2*max(x1)]);
xlabel('Time');
ylabel('Amplitude');

```

APPENDIX D

WELTI CODE GENERATOR ROUTINE

This routine generates a pair of Welty codes. A set of an initial pair of length two is used to generate the pair of Welty codes. Each of the codes is 64 chips long. Each chip in a code is sampled 64 times. The Fourier transforms of the pair of codes is also generated and plotted.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Welty Code Generator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% L -- number of chips in the Golay code
% a, b -- Welty complementary codes
% awav -- Welty code sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L=64;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Welty complementary Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a=[1,1];
b=[-1,1];
awav=[0];
bwav=[0];
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate WELTI Codes of 64 elements
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

n=log2(L);
for i=1:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample WELTI Code a, # of samples=64x64=4096
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

for i=1:length(a),
    for j=1:64
        awav=[awav,a(i)];
    end
end

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot coded waveform awav
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

figure;
subplot(1,2,1);
t = linspace(1,64,4097);
plot(t,awav);
axis([min(t) max(t) -1.5 1.5]);
xlabel('Time(unit chip width)');
ylabel('Amplitude');
title('64 Element WELTI Code');

```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and plot the Fourier Transform of the coded waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
fawav=abs(fft(awav));
subplot(1,2,2);
plot(fawav);
axis([0,255,0,700]);
xlabel('Frequency');
title('Spectrum of 64 Element WELTI Code');
```

APPENDIX E

GOLAY CODE GENERATOR ROUTINE

This routine generates a pair of Golay codes. A set of an initial pair of length two is used to generate the pair of Golay codes. Each of the codes is 64 chips long. Each chip in a code is sampled 64 times. The Fourier transforms of the pair of codes is also generated and plotted.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Golay Code Generator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% L -- number of chips in the Golay code
% a, b -- Golay complementary codes
% awav -- Golay code sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L=64;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Golay complementary Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a=[1,1,-1,1];
b=-[1,-1,-1,-1];
awav=[0];
bwav=[0];
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Golay codes of 64 elements
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

n=log2(L);
for i=2:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample Golay Code a, # of samples=64x64=4096
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

for i=1:length(a),
    for j=1:64
        awav=[awav,a(i)];
    end
end

```

```

awav(:,1)=[];

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Coded waveform awav
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

figure;
t = linspace(1,64,4096);
subplot(1,2,1);
plot(t,awav);
axis([min(t) max(t) -1.5 1.5]);
xlabel('Time(unit chip width)');
ylabel('Amplitude');
title('64 Element Golay Code');

```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot the Fourier Transform of the Coded Waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
fawav=abs(fft(awav));
subplot(1,2,2);
plot(fawav);
axis([0,255,0,700]);
xlabel('Frequency');
title('Spectrum of 64 Element Golay Code');
```

APPENDIX F

ROUTINE FOR THE GENERATION OF THE SQUARE OF THE WELTI CODED WAVEFORM

A Welty code of 64 chips is first generated. Each chip in the code is sampled 64 times. The Welty code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. A Welty coded cw waveform is then generated by multiplying the periodic pulse by a carrier frequency. The square of the cw waveform is then generated.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Generation of the Square of the Welty Coded Waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  L -- number of chips in the Welty code
%  a, b -- Welty complementary codes
%  awav -- Welty code sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
L=64;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Welty Complementary Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
a=[1,1];
b=[-1,1];
awav=[0];
bwav=[0];
```



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate WELTI Codes of 64 elements
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n=log2(L);
for i=2:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

awav=a(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample WELTI Code a, # of samples=64x64=4096
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:length(a)
    samples=64;
    if i==1,
        samples=63;
    end
    for j=1:samples
        awav=[awav,a(i)];
        bwav=[bwav,b(i)];
    end
end

N=length(awav);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create a Periodic WELTI Code Waveform by Padding awav with 0's.
% Length of New Pulse is 5N.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

zeros=[0];
p=(4*N)-1;
for j=1:p
    zeros=[zeros,0];
end

```

```

awav=[awav,zeros];
bwav=[bwav,zeros];
N=length(awav);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate A WELT Coded cw Waveform
% The Carrier Frequency is 12MHz
% The Carrier is first sampled at fs=192MHz, then Multiplied by the Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

fs=192E06;
fc=12E06;
t=0:(1/fs):(N-1)*(1/fs);
carrier=cos(2*pi*fc*t);
awav=awav.*carrier;

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot the PSD of the WELT Coded cw Waveform. PSD is normalized
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

figure;
y1=abs(fft(awav));
Py1= y1.*conj(y1)/N;
Py1=Py1/max(Py1);
fd=(0:N-1)*(fs/N);
subplot(1,2,1);
plot(fd,Py1);
axis([0 30E06 0 1.5]);
xlabel('Hz');
ylabel('Magnitude');
title('PSD of WELT Coded Waveform');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the Square of the WELT Coded cw Waveform.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

awav=awav.^2;

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot the PSD of the Squared Signal. The Squared Signal is
% Normalized and Divided by 2 to Match Axis For Unsquared Signal.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

y2=abs(fft(awav));
Py2= y2.*conj(y2)/N;
Py2=Py2/max(Py2);
fd=(0:N-1)*(fs/N);
subplot(1,2,2);
plot(fd,Py2);
axis([0 30E06 0 1.5]);
xlabel('Hz');
title('PSD of Square of Coded Waveform');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot, in dB, the PSD of the Welty Coded cw Waveform and the Squared Signal.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

Py1db=10.*log(Py1);
Py2db=10.*log(Py2);
figure;

subplot(1,2,1);
plot(fd,Py1db);
axis([0 30E06 -60 1]);
xlabel('Hz');
ylabel('Magnitude In dB');
title('PSD of Welty Coded Waveform');

```

```

subplot(1,2,2);
plot(fd,Py2db/2);
axis([0 30E06 -60 1]);
xlabel('Hz');
title('PSD of Square of Coded Waveform');

```

APPENDIX G

ROUTINE FOR THE GENERATION OF THE FOURTH ORDERED WELTI CODED WAVEFORM

A Welty Code of 64 chips is first generated. Each chip in the code is sampled 64 times. The Welty Code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. A Welty Coded cw waveform is then generated by multiplying the periodic pulse by a carrier frequency. The fourth ordered Welty coded cw waveform is then generated.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Generation of the Fourth Ordered Welty Coded Waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  L -- number of chips in the Welty code
%  a, b -- Welty complementary codes
%  awav -- Welty code sampled 64 times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L=64;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial Welty Complementary Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a=[1,1];
b=[-1,1];
awav=[0];
bwav=[0];
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Welter Codes of 64 elements
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

n=log2(L/2);
for i=2:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

```

```

awav=a(1);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample Welter Code a, # of samples=64x64=4096
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

for i=1:length(a)
    samples=64;
    if i==1,
        samples=63;
    end
    for j=1:samples
        awav=[awav,a(i)];
        bwav=[bwav,b(i)];
    end
end

```

```

N=length(awav);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create a Periodic Welter Code Waveform by Padding awav with 0's.
% Length of New Pulse is 5N.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

zeros=[0];

p=(4*N)-1;
for j=1:p
    zeros=[zeros,0];
end

```

```

awav=[awav,zeros];
bwav=[bwav,zeros];
N=length(awav);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate A WELTI Coded cw Waveform
% The Carrier Frequency is 12MHz
% The Carrier is first sampled at fs=192MHz, then Multiplied by the Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fs=192E06;
fc=12E06;
t=0:(1/fs):(N-1)*(1/fs);

carrier=cos(2*pi*fc*t);
awav=awav.*carrier;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot In dB the PSD of the WELTI Coded cw Waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y1=abs(fft(awav));
Py1= y1.*conj(y1)/N;
Py1=Py1/max(Py1);
Py1db=10.*log(Py1);
fd=(0:N-1)*(fs/N);
subplot(1,2,1);
plot(fd,Py1db);
axis([0 50E06 -60 1]);
xlabel('Hz');
ylabel('Magnitude In dB');
title('PSD of WELTI Coded Waveform');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the Fourth Ordered WELTI Coded cw Waveform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

awav=awav.^4;

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot, In dB, the PSD of the Fourth Ordered Signal
% Normalized and Divided by 2 to Match Axis For Unsquared Signal.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

y2=abs(fft(awav));
Py2= y2.*conj(y2)/N;
Py2=Py2/max(Py2);
Py2db=10.*log(Py2);
fd=(0:N-1)*(fs/N);
subplot(1,2,2);
plot(fd,Py2db/2);
axis([0 50E06 -60 1]);
xlabel('Hz');
title('PSD of Fourth Ordered Waveform');

```

APPENDIX H

ROUTINE FOR FILTERING

THE WELTI CODE WITH A BANDWIDTH OF $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$

A Welty code of 16 chip is first generated. The Welty code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. The Welty code is then filtered utilizing a notch filter of bandwidth $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$, or 60 kHz, where τ is the codes pulse width. For accurate comparison of energy suppression, the peak energy value in the filtered Welty code is restored, through normalization, to its original value in the unfiltered code. An energy suppressed Welty coded cw waveform is then generated by multiplying the filtered Welty code with the carrier frequency. Average power is then measured within a reference bandwidth, $\left(\frac{2}{\tau}\right)$, in the suppressed Welty coded cw waveform. Next, average power within the reference bandwidths in the unsuppressed and suppressed cw waveforms is compared. Finally, the composite autocorrelations for the unfiltered and filtered Welty codes are also plotted and compared.


```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

wa = [1 1];
wb = [1 -1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 2:log2(Nchips),
    c = [wa,-1.*wb];
    d = [wa,wb];
    wa = c;
    wb = d;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample codes nsamples times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:length(wa),
    awav = [awav,wa(i)*ones(1,nsamples)];
    bwav = [bwav,wb(i)*ones(1,nsamples)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create periodic WELTI codes
% Obtain the composite autocorrelation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aunfilt=[awav,zeros(1,(p-1)*num)];
bunfilt=[bwav,zeros(1,(p-1)*num)];

y1u=conv(aunfilt,fliplr(aunfilt));
z1u=conv(bunfilt,fliplr(bunfilt));
compu=y1u+z1u;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in aunfilt
% Find energy after square law no carrier
% Find maximum peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

ea=sum(aunfilt.^2)/N % Energy in the unfiltered Welter code
eamax=max(aunfilt.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Welter code
% Plot Fourier transform of Welter code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (microsec)');
ylabel('Amplitude');
title('16 Chip Welter Code')

fwava=fft(aunfilt)/N;
fwavb=fft(bunfilt)/N;
fwava(1)=0;

subplot(2,1,2);
plot(f,abs(fwava)/max(abs(fwava)));
axis([0 12 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('16 Chip Welter Code Spectra')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Add 12 MHz carrier
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

f0=12;
carrier = cos(2*pi*f0*t);
acar = carrier.*aunfilt;
bcar = carrier.*bunfilt;
wav=acar;

fwavu=fft(acar)/length(acar);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squa=acar.^2;
squb=bcar.^2;

```

```

fwavsu=fft(squa)/length(squa);
fwavsu(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define first filter cutoff frequency and filter Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nnn=1/100;
fc=nnn*chipfreq;
fcc=fc*1E-6;
fss=fs*1E-6;
mfa=(f > fcc & f < (fss-fcc)).*fwava;
mfb=(f > fcc & f < (fss-fcc)).*fwavb;

w1a=real(ifft(mfa)).*abs(aunfilt);
w1b=real(ifft(mfb)).*abs(aunfilt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in filtered Welti code
% Find energy in filtered Welti code after square law no carrier
% Find maximum peak power in filtered Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ew1a=sum(w1a.^2)/N;
ew1amax=max(w1a.^2);

ew1b=sum(w1b.^2)/N;
ew1bmax=max(w1b.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Normalize peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w1a=sqrt(eamax/ew1amax)*w1a;
w1b=sqrt(eamax/ew1bmax)*w1b;

esf=sum(w1a.^2)/N % Welti energy after normalization
mfas=fft(w1a);

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
title('Unfiltered Welti Code')

```

```

ylabel('Amplitude');
subplot(2,1,2);
plot(t,w1a);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (\musec)');
ylabel('Amplitude');
title('Filtered Welte Code')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compare original spectrum with scaled filtered spectrum
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(f,abs(fwava)/max(abs(fwava)));
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Unfiltered Welte Code Spectra');

subplot(2,1,2);
plot(f,abs(mfa)/max(abs(mfa)));
xlabel('Frequency (MHz)');
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Filtered Welte Code Spectra');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate energy in the two spectra
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

eoriginal=sum(abs(fwava).^2)/N % Energy in original Welte code
efiltered=sum(abs(mfas).^2)/N % Energy in filtered scaled Welte code

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain and Plot composite autocorrelation before and
% after filtering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y1f=conv(w1a,fliplr(w1a));
z1f=conv(w1b,fliplr(w1b));
num2=length(w1a);
ta=T*1E6*[-(num2-1):(num2-1)]/(num2-1);
compf=y1f+z1f;
figure;
subplot(2,1,1)

```

```

plot(ta,compu/max(compu));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
ylabel('Amplitude');
title('Composite Autocorrelation Before Filtering')
subplot(2,1,2)
plot(ta,compf/max(compf));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
xlabel('Delay Tau (\musec)');
ylabel('Amplitude');
title('Composite Autocorrelation After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier) after wave filtering
% Plot squared filtered signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

acarf = carrier.*w1a;
bcarf = carrier.*w1b;

figure;
fwavf=fft(acarf)/length(acarf);
plot(f,(abs(fwavf)/max(abs(fwavf))));
axis([0 40 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of Coded cw Waveform after Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find output power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squaf=acarf.^2;
squbf=bcarf.^2;

fwavsf=fft(squaf)/length(squaf);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pspecf=abs(fwavsf).^2;
pspecu=abs(fwavsu).^2;
pspecumax=max(pspecu)/4;

```

```

figure;
subplot(2,1,1);
plot(f,(pspecu/pspecumax));
axis([18 30 0 1]);
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform Before Filtering')

subplot(2,1,2)
plot(f,(pspecf/pspecumax));
axis([18 30 0 1]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate total energy in band around twice carrier frequency
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fl=2*f0-chipfreq*1E-6;
fh=2*f0+chipfreq*1E-6;
specun=(f > fl & f < fh).*pspecu;
specf=(f > fl & f < fh).*pspecf;
mmax=max(specun);

energyun=sum(specun) % Energy in BW of square of unfiltered waveform
energyf=sum(specf) % Energy in BW of square of filtered waveform

ratioenergy=100*(abs(energyun-energyf)/energyun)

```

APPENDIX I

ROUTINE FOR FILTERING

THE WELTI CODE WITH A BANDWIDTH OF $\frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$

A Welty code of 16 chip is first generated. The Welty code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. The Welty code is then filtered utilizing a notch filter of bandwidth $\frac{1}{50} \cdot \left(\frac{2}{\tau}\right)$, or 120 kHz, where τ is the codes pulse width. For accurate comparison of energy suppression, the peak energy value in the filtered Welty code is restored, through normalization, to its original value in the unfiltered code. An energy suppressed Welty coded cw waveform is then generated by multiplying the filtered Welty code with the carrier frequency. Average power is then measured within a reference bandwidth, $\left(\frac{2}{\tau}\right)$, in the suppressed Welty coded cw waveform. Next, average power within the reference bandwidths in the unsuppressed and suppressed cw waveforms is compared. Finally, the composite autocorrelations for the unfiltered and filtered Welty codes are also plotted and compared.


```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

wa = [1 1];
wb = [1 -1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 2:log2(Nchips),
    c = [wa,-1.*wb];
    d = [wa,wb];
    wa = c;
    wb = d;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample codes nsamples times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:length(wa),
    awav = [awav,wa(i)*ones(1,nsamples)];
    bwav = [bwav,wb(i)*ones(1,nsamples)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create periodic WELTI codes
% Obtain the composite autocorrelation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aunfilt=[awav,zeros(1,(p-1)*num)];
bunfilt=[bwav,zeros(1,(p-1)*num)];

y1u=conv(aunfilt,fliplr(aunfilt));
z1u=conv(bunfilt,fliplr(bunfilt));
compu=y1u+z1u;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in aunfilt
% Find energy after square law no carrier
% Find maximum peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

ea=sum(aunfilt.^2)/N % Energy in the unfiltered Welter code
eamax=max(aunfilt.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Welter code
% Plot Fourier transform of Welter code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (microsec)');
ylabel('Amplitude');
title('16 Chip Welter Code')

fwava=fft(aunfilt)/N;
fwavb=fft(bunfilt)/N;
fwava(1)=0;

subplot(2,1,2);
plot(f,abs(fwava)/max(abs(fwava)));
axis([0 12 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('16 Chip Welter Code Spectra')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Add 12 MHz carrier
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

f0=12;
carrier = cos(2*pi*f0*t);
acar = carrier.*aunfilt;
bcar = carrier.*bunfilt;
wav=acar;

fwavu=fft(acar)/length(acar);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squa=acar.^2;
squb=bcar.^2;

```

```

fwavsu=fft(squa)/length(squa);
fwavsu(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define first filter cutoff frequency and filter Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nnn=1/50;
fc=nnn*chipfreq;
fcc=fc*1E-6;
fss=fs*1E-6;
mfa=(f > fcc & f < (fss-fcc)).*fwava;
mfb=(f > fcc & f < (fss-fcc)).*fwavb;

w1a=real(ifft(mfa)).*abs(aunfilt);
w1b=real(ifft(mfb)).*abs(aunfilt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in filtered Welti code
% Find energy in filtered Welti code after square law no carrier
% Find maximum peak power in filtered Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ew1a=sum(w1a.^2)/N;
ew1amax=max(w1a.^2);

ew1b=sum(w1b.^2)/N;
ew1bmax=max(w1b.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Normalize peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w1a=sqrt(eamax/ew1amax)*w1a;
w1b=sqrt(eamax/ew1bmax)*w1b;

esf=sum(w1a.^2)/N % Welti energy after normalization
mfas=fft(w1a);

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
title('Unfiltered Welti Code')

```

```

ylabel('Amplitude');
subplot(2,1,2);
plot(t,w1a);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (\musec)');
ylabel('Amplitude');
title('Filtered Welte Code')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compare original spectrum with scaled filtered spectrum
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(f,abs(fwava)/max(abs(fwava)));
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Unfiltered Welte Code Spectra');

subplot(2,1,2);
plot(f,abs(mfa)/max(abs(mfa)));
xlabel('Frequency (MHz)');
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Filtered Welte Code Spectra');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate energy in the two spectra
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

eoriginal=sum(abs(fwava).^2)/N % Energy in original Welte code
efiltered=sum(abs(mfas).^2)/N % Energy in filtered scaled Welte code

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain and Plot composite autocorrelation before and
% after filtering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y1f=conv(w1a,fliplr(w1a));
z1f=conv(w1b,fliplr(w1b));
num2=length(w1a);
ta=T*1E6*[-(num2-1):(num2-1)]/(num2-1);
compf=y1f+z1f;
figure;
subplot(2,1,1)

```

```

plot(ta,compu/max(compu));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
ylabel('Amplitude');
title('Composite Autocorrelation Before Filtering')
subplot(2,1,2)
plot(ta,compf/max(compf));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
xlabel('Delay Tau (\musec)');
ylabel('Amplitude');
title('Composite Autocorrelation After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier) after wave filtering
% Plot squared filtered signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

acarf = carrier.*w1a;
bcarf = carrier.*w1b;

figure;
fwavf=fft(acarf)/length(acarf);
plot(f,(abs(fwavf)/max(abs(fwavf))));
axis([0 40 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of Coded cw Waveform after Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find output power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squaf=acarf.^2;
squbf=bcarf.^2;

fwavsf=fft(squaf)/length(squaf);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pspecf=abs(fwavsf).^2;
pspecu=abs(fwavsu).^2;
pspecumax=max(pspecu)/4;

```

```

figure;
subplot(2,1,1);
plot(f,(pspecu/pspecumax));
axis([18 30 0 1]);
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform Before Filtering')

subplot(2,1,2)
plot(f,(pspecf/pspecumax));
axis([18 30 0 1]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate total energy in band around twice carrier frequency
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fl=2*f0-chipfreq*1E-6;
fh=2*f0+chipfreq*1E-6;
specun=(f > fl & f < fh).*pspecu;
specf=(f > fl & f < fh).*pspecf;
mmax=max(specun);

energyun=sum(specun) % Energy in BW of square of unfiltered waveform
energyf=sum(specf) % Energy in BW of square of filtered waveform

ratioenergy=100*(abs(energyun-energyf)/energyun)

```

APPENDIX J

ROUTINE FOR FILTERING

THE WELTI CODE WITH A BANDWIDTH OF $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$

A Welty code of 16 chip is first generated. The Welty code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. The Welty code is then filtered utilizing a notch filter of bandwidth $\frac{1}{25} \cdot \left(\frac{2}{\tau}\right)$, or 240 kHz, where τ is the codes pulse width. For accurate comparison of energy suppression, the peak energy value in the filtered Welty code is restored, through normalization, to its original value in the unfiltered code. An energy suppressed Welty coded cw waveform is then generated by multiplying the filtered Welty code with the carrier frequency. Average power is then measured within a reference bandwidth, $\left(\frac{2}{\tau}\right)$, in the suppressed Welty coded cw waveform. Next, average power within the reference bandwidths in the unsuppressed and suppressed cw waveforms is compared. Finally, the composite autocorrelations for the unfiltered and filtered Welty codes are also plotted and compared.


```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

wa = [1 1];
wb = [1 -1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 2:log2(Nchips),
    c = [wa,-1.*wb];
    d = [wa,wb];
    wa = c;
    wb = d;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample codes nsamples times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:length(wa),
    awav = [awav,wa(i)*ones(1,nsamples)];
    bwav = [bwav,wb(i)*ones(1,nsamples)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create periodic WELTI codes
% Obtain the composite autocorrelation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aunfilt=[awav,zeros(1,(p-1)*num)];
bunfilt=[bwav,zeros(1,(p-1)*num)];

y1u=conv(aunfilt,fliplr(aunfilt));
z1u=conv(bunfilt,fliplr(bunfilt));
compu=y1u+z1u;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in aunfilt
% Find energy after square law no carrier
% Find maximum peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

ea=sum(aunfilt.^2)/N % Energy in the unfiltered Welter code
eamax=max(aunfilt.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot Welter code
% Plot Fourier transform of Welter code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (microsec)');
ylabel('Amplitude');
title('16 Chip Welter Code')

fwava=fft(aunfilt)/N;
fwavb=fft(bunfilt)/N;
fwava(1)=0;

subplot(2,1,2);
plot(f,abs(fwava)/max(abs(fwava)));
axis([0 12 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('16 Chip Welter Code Spectra')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Add 12 MHz carrier
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

f0=12;
carrier = cos(2*pi*f0*t);
acar = carrier.*aunfilt;
bcar = carrier.*bunfilt;
wav=acar;

fwavu=fft(acar)/length(acar);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squa=acar.^2;
squb=bcar.^2;

```

```

fwavsu=fft(squa)/length(squa);
fwavsu(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define first filter cutoff frequency and filter Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nnn=1/25;
fc=nnn*chipfreq;
fcc=fc*1E-6;
fss=fs*1E-6;
mfa=(f > fcc & f < (fss-fcc)).*fwava;
mfb=(f > fcc & f < (fss-fcc)).*fwavb;

w1a=real(ifft(mfa)).*abs(aunfilt);
w1b=real(ifft(mfb)).*abs(aunfilt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in filtered Welti code
% Find energy in filtered Welti code after square law no carrier
% Find maximum peak power in filtered Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ew1a=sum(w1a.^2)/N;
ew1amax=max(w1a.^2);

ew1b=sum(w1b.^2)/N;
ew1bmax=max(w1b.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Normalize peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w1a=sqrt(eamax/ew1amax)*w1a;
w1b=sqrt(eamax/ew1bmax)*w1b;

esf=sum(w1a.^2)/N % Welti energy after normalization
mfas=fft(w1a);

figure;
subplot(2,1,1);
plot(t,aunfilt);
axis([0 max(t) -1.5 1.5]);
title('Unfiltered Welti Code')

```

```

ylabel('Amplitude');
subplot(2,1,2);
plot(t,w1a);
axis([0 max(t) -1.5 1.5]);
xlabel('Time (\musec)');
ylabel('Amplitude');
title('Filtered Welte Code')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compare original spectrum with scaled filtered spectrum
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure;
subplot(2,1,1);
plot(f,abs(fwava)/max(abs(fwava)));
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Unfiltered Welte Code Spectra');

subplot(2,1,2);
plot(f,abs(mfa)/max(abs(mfa)));
xlabel('Frequency (MHz)');
ylabel('Magnitude');
axis([0 12 0 1.2]);
title('Filtered Welte Code Spectra');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate energy in the two spectra
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

eoriginal=sum(abs(fwava).^2)/N % Energy in original Welte code
efiltered=sum(abs(mfas).^2)/N % Energy in filtered scaled Welte code

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain and Plot composite autocorrelation before and
% after filtering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y1f=conv(w1a,fliplr(w1a));
z1f=conv(w1b,fliplr(w1b));
num2=length(w1a);
ta=T*1E6*[-(num2-1):(num2-1)]/(num2-1);
compf=y1f+z1f;
figure;
subplot(2,1,1)

```

```

plot(ta,compu/max(compu));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
ylabel('Amplitude');
title('Composite Autocorrelation Before Filtering')
subplot(2,1,2)
plot(ta,compf/max(compf));
axis([-max(ta)/2 max(ta)/2 -.5 1.2]);
xlabel('Delay Tau (\musec)');
ylabel('Amplitude');
title('Composite Autocorrelation After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier) after wave filtering
% Plot squared filtered signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

acarf = carrier.*w1a;
bcarf = carrier.*w1b;

figure;
fwavf=fft(acarf)/length(acarf);
plot(f,(abs(fwavf)/max(abs(fwavf))));
axis([0 40 0 1.2]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of Coded cw Waveform after Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find output power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squaf=acarf.^2;
squbf=bcarf.^2;

fwavsf=fft(squaf)/length(squaf);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pspecf=abs(fwavsf).^2;
pspecu=abs(fwavsu).^2;
pspecumax=max(pspecu)/4;

```

```

figure;
subplot(2,1,1);
plot(f,(pspecu/pspecumax));
axis([18 30 0 1]);
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform Before Filtering')

subplot(2,1,2)
plot(f,(pspecf/pspecumax));
axis([18 30 0 1]);
xlabel('Frequency (MHz)');
ylabel('Magnitude');
title('Spectra of the Square of Coded cw Waveform After Filtering')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate total energy in band around twice carrier frequency
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fl=2*f0-chipfreq*1E-6;
fh=2*f0+chipfreq*1E-6;
specun=(f > fl & f < fh).*pspecu;
specf=(f > fl & f < fh).*pspecf;
mmax=max(specun);

energyun=sum(specun) % Energy in BW of square of unfiltered waveform
energyf=sum(specf) % Energy in BW of square of filtered waveform

ratioenergy=100*(abs(energyun-energyf)/energyun)

```

APPENDIX K

ROUTINE FOR PLOTTING

WELTI CODE FILTER BANDWIDTH

VS

PERCENTAGE AVERAGE POWER REDUCTION

This routine plots a curve of the Welty code notch filter bandwidth versus percentage average power reduction in the reference bandwidth of the suppressed Welty coded cw waveform. Welty code filtering is performed in the same manner as in the routines of Appendices H, I, and J. Filtering is performed within a loop that iterates the notch filter bandwidth in increments of $\frac{1}{100} \cdot \left(\frac{2}{\tau}\right)$, or 60 kHz, where τ is the Welty code's chip width. The bandwidth scale is plotted in its normalized form where frequency was divided by the bandwidth of the main-lobe in the Welty code spectrum, that is, the bandwidth axis was divided by $\left(\frac{2}{\tau}\right)$, or 6 MHz. The notch filter bandwidth was varied from 0 up to 600 kHz. In the equivalent normalized scale, the normalized filter bandwidth was varied from 0 up to 0.1 in increments of $\frac{1}{100}$.


```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Plotting of Welter Code Filter BW
%                               vs
%                               Percentage Average Power Reduction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% a, b          -- Welter codes
% awav, bwav    -- Welter codes sampled 64 times per chip and multiplied
%               by a 12 MHz carrier (192 MHz sample frequency).
% awavf, bwavf  -- Welter codes sampled and filtered with a Notch
%               Filter.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adjustable Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Nchips        -- number of chips in the Golay coded pair
%               (must be a power of two)
% fs            -- sample frequency
% Tau           -- Pulse Width
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

duty = 1/5;
p = 1/duty;
chipfreq = 3E6;
Tau = 1/chipfreq;
Nchips = 16;
nsamples = 64;
delt = 1/(nsamples*chipfreq);
fs = 1/delt;
num = Nchips*nsamples;

N = Nchips*nsamples/duty;
T = delt*N;
f = 1E-6*(0:N-1)*fs/(N-1);
t = 1E6*(0:N-1)*T/(N-1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initializing variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

awav = [];
bwav = [];

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

wa = [1 1];
wb = [1 -1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 2:log2(Nchips),
    c = [wa,-1.*wb];
    d = [wa,wb];
    wa = c;
    wb = d;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample codes nsamples times per chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:length(wa),
    awav = [awav,wa(i)*ones(1,nsamples)];
    bwav = [bwav,wb(i)*ones(1,nsamples)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create periodic WELTI codes
% Obtain the composite autocorrelation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aunfilt=[awav,zeros(1,(p-1)*num)];
bunfilt=[bwav,zeros(1,(p-1)*num)];

y1u=conv(aunfilt,fliplr(aunfilt));
z1u=conv(bunfilt,fliplr(bunfilt));
compu=y1u+z1u;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in aunfilt
% Find energy after square law no carrier
% Find maximum peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

ea=sum(aunfilt.^2)/N % Energy in the unfiltered Welti code
eamax=max(aunfilt.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain Fourier transform of Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fwava=fft(aunfilt)/N;
fwavb=fft(bunfilt)/N;
fwava(1)=0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Add 12 MHZ carrier
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

f0=12;
carrier = cos(2*pi*f0*t);
acar = carrier.*aunfilt;
bcar = carrier.*bunfilt;
wav=acar;

fwavu=fft(acar)/length(acar);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squa=acar.^2;
squb=bcar.^2;
e4un=sum(squa.^2)/N
fwavsu=fft(squa)/length(squa);
fwavsu(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define first filter cutoff frequency and
% filter Welti code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fff(1)=0;
ratioenergy(1)=0;

```

```

for j=2:11;

nnn=(j-1)/100;
fc=nnn*chipfreq;
fff(j)=2*fc*1E-6;
fcc=fc*1E-6;
fss=fs*1E-6;
mfa=(f > fcc & f < (fss-fcc)).*fwava;
mfb=(f > fcc & f < (fss-fcc)).*fwavb;

w1a=real(ifft(mfa)).*abs(aunfilt);
w1b=real(ifft(mfb)).*abs(aunfilt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find energy in filtered WELTI code
% Find energy in filtered WELTI code after square law no carrier
% Find maximum peak power in filtered WELTI code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ew1a=sum(w1a.^2)/N;
ew1amax=max(w1a.^2);

ew1b=sum(w1b.^2)/N;
ew1bmax=max(w1b.^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Normalize peak power
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w1a=sqrt(eamax/ew1amax)*w1a;
w1b=sqrt(eamax/ew1bmax)*w1b;

esf=sum(w1a.^2)/N; % WELTI energy after normalization
mfas=fft(w1a);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate energy in the two spectra
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

eoriginal=sum(abs(fwava).^2)/N; % Energy in original WELTI code
efiltered=sum(abs(mfas).^2)/N; % Energy in filtered scaled WELTI code

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain composite autocorrelation before and
% after filtering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y1f=conv(w1a,fliplr(w1a));
z1f=conv(w1b,fliplr(w1b));
num2=length(w1a);
ta=T*1E6*[-(num2-1):(num2-1)]/(num2-1);
compf=y1f+z1f;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Square (wave + carrier) after wave filtering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

acarf = carrier.*w1a;
bcarf = carrier.*w1b;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Find output power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

squaf=acarf.^2;
squbf=bcarf.^2;

fwavsf=fft(squaf)/length(squaf);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the power spectral density
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pspecf=abs(fwavsf).^2;
pspecu=abs(fwavsu).^2;
pspecumax=max(pspecu)/4;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate total energy in band around twice carrier frequency
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fl=2*f0-chipfreq*1E-6;
fh=2*f0+chipfreq*1E-6;
specun=(f > fl & f < fh).*pspecu;

```

```

specf=(f > fl & f < fh).*pspecf;
mmax=max(specun);

energyun=sum(specun) % Energy in BW of square of unfiltered waveform
energyf=sum(specf) % Energy in BW of square of filtered waveform

ratioenergy(j)=100*(abs(energyun-energyf)/energyun);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Normalize frequency scale and plot BW vs percentage
% average power reduction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

BWn=fff/6;

figure;
plot(BWn,ratioenergy);
xlabel('Normalized Bandwidth')
ylabel('Percentage Average Power Reduction');
axis([0 max(BWn) 0 100]);
title('Walti Code Filter BW vs Percentage Average Power Reduction');
hold on;
plot(BWn(2),ratioenergy(2),'o',BWn(3),ratioenergy(3),'o',BWn(5),ratioenergy(5),'o');
hold off;

```

APPENDIX L

ROUTINE FOR PERFORMING A DELAY-AND-MULTIPLY OPERATION ON THE WELTI CODE

A WELTI code of 16 chips is first generated. The WELTI code is then transformed into a periodic pulse by padding it with zeros. The length of the new pulse is five times the original pulse. A $\frac{1}{2}$ chip delayed code is next generated. The delay-and-multiply operation is then performed by multiplying the delayed code with the undelayed one. Finally, the PSD of the delayed-and-multiplied waveform is obtained and plotted for characteristic observation.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       This Program Performs a Delay and Multiply Operation
%       on a 16 Chip WELTI Code
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Initial the Two Complementary WELTI Pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a=[1,1];
b=[-1,1];
```

```

awav=[0];
bwav=[0];

L=16;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the Welter Codes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n=log2(L);
for i=2:n,
    c=[a,-1.*b];
    d=[a,b];
    a=c;
    b=d;
end

awav=a(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sample the Welter Codes - 64 Samples Per Chip
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:length(a)
    samples=64;
    if i==1,
        samples=63;
    end
    for j=1:samples
        awav=[awav,a(i)];
        bwav=[bwav,b(i)];
    end
end

N1=length(awav);
awav1=awav;

```



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create Periodic WELTI Code Waveforms by Padding awav and bwav with 0's.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

p=(4*N1);

zer=zeros(1,p);
awav=[awav,zer];
bwav=[bwav,zer];
N2=length(awav);
fs=192E06;
ts=1/fs;
t=0:(ts):(N2-1)*(ts);
f=(0:N2-1)*(fs/N2);

```

```

figure;
subplot(3,1,1);
plot(t,awav);
axis([0 1E-5 -1.5 1.5]);
ylabel('Amplitude');
title('WELTI Code')

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the PSD of WELTI Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

y=abs(fft(awav));
Py= y.*conj(y)/N2;
PydB=10.*log10(Py);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate Delayed WELTI Code and Plot the Delayed Code.
% The Delay Code is 1/2 Chip Delayed.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

samples=64;
delay2=samples/2;
X=zeros(1,delay2);
Y=[X,awav1];
p2=p-delay2;
Z2=zeros(1,p2);
delawav2=[Y,Z2];

```

```

subplot(3,1,2);
plot(t,delawav2);
axis([0 1E-5 -1.5 1.5]);
ylabel('Amplitude');
title('1/2 Chip Delayed WELTI Code');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate and Plot the Delayed-and-Multiplied Signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

pawav2=awav.*delawav2;
subplot(3,1,3);
plot(t,pawav2);
axis([0 1E-5 -1.5 1.5]);
xlabel('Time (sec)');
ylabel('Amplitude');
title('Delayed-and-Multiplied WELTI Code');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot the PSD of the WELTI Code
% Generate and Plot the PSD of the Delayed-and-Multiplied Signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

figure;
subplot(2,1,1);
plot(f,10*Py);
axis([0 3600E4 0 25]);
ylabel('Magnitude');
title('PSD of the WELTI Code');

```

```

y2=abs(fft(pawav2));
Py2= y2.*conj(y2)/N2;
subplot(2,1,2);
plot(f,10*Py2);
axis([0 3600E4 0 25]);
xlabel('Hz');
ylabel('Magnitude');
title('PSD of 1/2 Chip Delayed-and-Multiplied WELTI Code');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot in dB the PSD of the WELTI Code
% Generate and Plot in dB the PSD of the Delayed-and-Multiplied Signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

figure;
PydB=10.*log10(Py);
subplot(2,1,1);
plot(f,10*PydB);
axis([0 3600E4 -1000 200]);
ylabel('Magnitude in dB');
title('PSD in dB of the WELTI Code');

```

```

Py2dB=10.*log10(Py2);
subplot(2,1,2);
plot(f,10*Py2dB);
axis([0 3600E4 -1000 200]);
xlabel('Hz');
ylabel('Magnitude in dB');
title('PSD in dB of 1/2 Chip Delayed-and-Multiplied WELTI Code');

```

REFERENCES

- [1] A. Leon-Garcia, *Probability and Random Processes*, Addison-Wesley, New York, 1989.
- [2] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, New York, 1985.
- [3] P. Peebles, *Probability, Random Variables, and Random Signal Principles*, McGraw-Hill, New York, 1980.
- [4] W. A. Gardner, *Cyclostationarity in Communications and Signal Processing*, IEEE Press, 1994.
- [5] F. B. Gross, D. Soileau and L. R. Kremer, "Investigation of Optimal Codes or Code Sets for Enhanced Radar Performance," Report for U.S. Special Operations Command, Contract No. USZA22-97-K-0001, December 1997.
- [6] A. B. Carlson, *Communication Systems*, McGraw-Hill, New York, 1986.
- [7] H. Ogura, "Spectral Representation of a Periodic Nonstationary Random Process," *IEEE Transactions on Information Theory*, vol. IT-17, pp. 143-149, March 1971.
- [8] M. Skolnik, *Introduction to Radar Systems*, McGraw-Hill, New York, 1980.
- [9] R. Gold, "Optimal Binary Sequences for Spread Spectrum Multiplexing," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 619-621, October 1967.
- [10] M. Skolnik, *Radar Handbook*, McGraw-Hill, New York, 1990.
- [11] C. E. Cook and M. Bernfeld, *Radar Signals – An Introduction to Theory and Applications*, Academic Press, New York, 1967.
- [12] W. A. Gardner and L. E. Franks, "Characterization of Cyclostationary Random Signal Processes," *IEEE Transactions on Information Theory*, vol. IT-21, no. 1, January 1975.
- [13] A.V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

- [14] W. A. Gardner and L. E. Franks, "Estimation for Cyclostationary Random Processes," in *Proceedings of 9th Annual Allerton Conference for Circuit and System Theory*, 1971, pp. 222-231.
- [15] C. C. Tseng, "Complementary Sets of Sequences," *IEEE Transactions on Information Theory*, vol. IT-18, pp.644-652, September 1972.
- [16] R. D. Strum and D. E. Kirk, *Discrete Systems and Digital Signal Processing*, Addison-Wesley, New York, 1988.
- [17] R. A. Dillard, "Detectability of Spread-Spectrum Signals," *IEEE Transactions on Aerospace and Electronics Systems*, AES-15, pp. 526-537, July 1979.
- [18] M. J. E. Golay, "Complementary Series," *IRE Transactions on Information Theory*, April 1961.
- [19] G. R. Welty, "Quaternary Codes for Pulsed Radar," *IRE Transactions on Information Theory*, June 1960.
- [20] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume II, Detection Theory*, Prentice-Hall, Upper Saddle River, NJ, 1998.
- [21] J. O. Mutton, "Advanced Pulse Compression Techniques," *Hughes Aircraft Company*, NAECON '75 RECORD, pp. 141-148, 1975.
- [22] W. A. Gardner, "Exploitation of Spectral Redundancy in Cyclostationary Signals," *IEEE Signal Processing Magazine*, vol. 8, pp. 14-36, 1988.
- [23] G. W. Stimson, *Introduction to Airborne Radar*, SciTech Publishing, Mendham, NJ, 1998.
- [24] E. G. Gladyshev, "Periodically and Almost Periodically Correlated Random Processes with Continuous Time Parameter," *Theory Prob. Appl.*, vol. 8, pp. 173-177, 1963.
- [25] W. A. Gardner, "Signal Interception: A Unifying Theoretical Framework for Feature Detection," *IEEE Transactions on Communications*, vol. COM-36, pp. 897-906, 1991.
- [26] W. A. Gardner and C. M. Spooner, "Signal Interception: Performance Advantages of Cyclic-Feature Detectors," *IEEE Transactions on Communications*, vol. 40, No. 1, 1992.

- [27] Z. Ding, "Characteristics of Band-Limited Channels unidentifiable From Second-Order Cyclostationary Statistics," *IEEE Transactions on Signal Processing*, vol. 3, pp. 150-152, May 1996.
- [28] G. D. Zivanovic and W. A. Gardner, "Degrees of Cyclostationarity and Their Application to Signal Detection and Estimation," *IEEE Transactions on Signal Processing*, vol. 22, pp. 287-297, March 1991.
- [29] J. W. Betz, "An Information Theoretic Measure of Cyclostationarity," The MITRE Corporation, The Second Workshop on Cyclostationary Signals, Monterey, CA, July 31-August 2, 1994.
- [30] E. Brookner, *Radar Technology*, Artech House, Dedham, Massachusetts, 1977.
- [31] J. I. Marcum, "A Statistical Theory of Target Detection by Pulsed Radar," *IRE Transactions on Information Theory*, vol. IT-6, No. 2, April 1960.
- [32] D. K. Barton, *Radar System Analysis*, Artech House, Dedham, Massachusetts, 1976.
- [33] G. V. Trunk, "Comparison of Collapsing Losses in Linear Square-Law Detectors," *Proceedings of the IEEE*, vol. 60, No. 6, June 1972.
- [34] J. V. DiFranco and W. L. Rubin, *Radar Detection*, Prentice-Hall, Englewood Cliffs, New Jersey, 1968.
- [35] D. K. Barton, "Simple Procedures for Radar Detection Calculation," *IEEE Transactions on Aerospace and Electronics Systems*, vol. AES-5, No. 5, September 1969.
- [36] D. P. Meyer and H. A. Mayer, *Radar Target Detection*, Academic Press, New York, 1973.
- [37] J. D. Wolf, G. M. Lee and C. E. Suvo, "Radar Waveform Synthesis by Mean-square Optimization Techniques," *IEEE Transactions on Aerospace and Electronics Systems*, July 1969.
- [38] D. K. Barton and H. R. Ward, *Handbook of Radar Measurement*, Prentice-Hall, Englewood Cliffs, New Jersey, 1969.
- [39] L. F. Fehlnner, "Marcum's and Swerling's Data on Target Detection by a Pulsed Radar," *Applied Physics Laboratory*, Report TG-451, The John Hopkins University, Silver Springs, Maryland, July 1962.

- [40] A. W. Rihaczek, "Radar Signal Design for Target Resolution," *Proceedings of the IEEE*, vol. 53, No. 2, pp. 116-128, February 1965.
- [41] P. M. Woodward, *Probability and Information Theory, with Applications to Radar*, Pergamon Press, London, 1953.
- [42] T. H. Powell and A. I. Sinsky, "A Time Sidelobe Reduction Technique for Small Time-Bandwidth Chirp," *IEEE Transactions on Aerospace and Electronics Systems*, May 1974.
- [43] R. S. Berkowitz, *Modern Radar*, John Wiley & Sons, New York, 1965.

BIOGRAPHICAL SKETCH

Mohsin Mansour Benghuzzi received his B.A. degree in Physics and Mathematics with a minor in Computer Science in 1984 and his M.S. degree in Electrical Engineering in 1989. He received his Ph.D. in Electrical Engineering in 2003. Mr. Benghuzzi currently works in the telecommunications industry and is certified by the State of Florida as a Professional Engineer.