1-1-2001

Construction of a random signal with a specific Psd and a uniform Pdf

Venkatraghavan V Bringi
University of Nevada, Las Vegas

CONSTRUCTION OF A RANDOM SIGNAL WITH A SPECIFIC PSD AND A UNIFORM PDF

by

Venkatraghavan Bringi

Bachelor of Engineering
University of Madras, India
1998

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree
Department of Electrical and Computer Engineering
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
August 2001

UMI Number: 1406386

UMI®

UMI Microform 1406386

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

UNLV

Thesis Approval

The Graduate College
University of Nevada, Las Vegas

June 25, 2001

The Thesis prepared by

Venkatraghavan Bringi

Entitled

Construction Of A Random Signal With A Specific PSD And A Uniform PDF

is approved in partial fulfillment of the requirements for the degree of

Masters of Science in Electrical Engineering

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Examination Committee Member

Graduate College Faculty Representative

PR/1017-53/1-00

ABSTRACT

CONSTRUCTION OF A RANDOM SIGNAL WITH A SPECIFIC PSD
AND A UNIFORM PDF

by

Venkatraghavan Bringi

Dr. Peter Stubberud, Examination Committee Chair
Associate Professor of Electrical and Computer Engineering
University of Nevada, Las Vegas

The performance of a dynamic element matching (DEM) flash digital to analog converter (DAC) can be improved by controlling the DEM DAC's interconnection network with a random signal that has a specific power spectral density (PSD) and a uniform probability distribution function (PDF). Many algorithms exist for generating a random signal with a white PSD and a uniform PDF, but there exists only one algorithm for generating a random signal with a specific PSD and a particular PDF. For DEM DAC applications, the random signal must be generated at the speed of the DEM DAC. However, a real time implementation of this existing algorithm is too computation intensive for a typical DEM DAC. In this thesis, an algorithm that constructs a uniformly distributed random signal with a specific PSD is developed. This uniformly distributed colored random signal is implemented using a finite state machine (FSM) and Linear Feedback Shift Registers (LFSRs).

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

It is my pleasure to gratefully acknowledge the mentoring of Dr. Peter Stubberud, Associate Professor of Electrical Engineering at the University of Nevada, Las Vegas. It has been a privilege and an honor to receive his guidance during the period of my thesis work.

A special acknowledgement is due to Dr. Evangelos Yfantis, Professor of Computer Science at the University of Nevada, Las Vegas, for his invaluable suggestions. The time I spent with him was very inspiring and of immense help to my research work. I express my heartfelt gratitude to Dr. Jay Karmarkar for spending his precious time in discussing the thesis. I appreciate and thank Dr. Rama Venkat and Dr. Eugene McGaugh for reviewing this thesis. I would like to acknowledge the entire faculty and staff at the Graduate College and the Department of Electrical and Computer Engineering for their cooperation.

I share this thesis and all my academic successes with my family. I thank them for their encouragement and support. Last but not the least, I thank and appreciate my friends for their encouragement in completion of this work.

CHAPTER 1

INTRODUCTION

Flash digital to analog converters (DACs) depend on matched components for converting a digital signal to an analog signal. In practice, perfectly matched components are impossible to fabricate. Even matched components on integrated circuits exhibit mismatch errors such as linear gradient mismatch errors, geometric mismatch errors and dynamic mismatch errors. The errors due to mismatched components add a nonlinear transformation, called integral nonlinearity (INL), to a flash DAC's linear transformation [1] and reduce the DAC's performance.

A signal processing algorithm called dynamic element matching (DEM) has been used to reduce the effects of component mismatches in DACs [2] thereby improving their performance. DEM algorithms reduce the effects of mismatched components by rearranging dynamically the interconnections of mismatched components so that the time averages of the equivalent components at each of the component positions are almost equal [2; 3]. In a flash DEM DAC, an interconnection network dynamically rearranges the mapping between the digital input signal and the mismatched unit DACs so that the time averages of the activated unit DAC outputs are almost equal and the time averages of the deactivated unit DAC outputs are almost equal [2; 3; 4; 5]. If the interconnection network's control signal is deterministic, the mapping between the digital input signal and the mismatched unit DACs is deterministic, and the DAC is said to be a deterministic

DAC [3]. Similarly, if the interconnection network's control signal is stochastic, the mapping between the digital input signal and the mismatched unit DACs is stochastic, and the DAC is said to be a stochastic DAC [3]. Many stochastic DEM DACs require a stochastic control signal with a uniform probability distribution function (PDF) and a particular power spectral density (PSD) to control the DEM algorithm's interconnection network [3].

In general, a linear system can shape a random signal's PSD to approximate a desired PSD; however a linear system typically cannot generate a signal with a desired PDF. Many other algorithms [6; 7] exist for generating a stochastic signal with a uniform PDF and a white PSD, only one algorithm [8] exists for generating a stochastic signal with a specific PDF and a particular PSD. The existing algorithm [8] generates a random signal with a specific PSD and a specific probability distribution using a linear system. This algorithm shapes a signal's PDF by representing the desired PDF by a set of approximation coefficients, determining an output/input relation that expresses the variation of the PDF while it is passing through the linear system, and then determining a relationship between the approximation coefficients and the output/input relation. An input signal is then constructed from an independent, identically distributed (i.i.d) uniform or Gausian process and a non-linear characteristic.

DEM DACs require a random signal that has the same sampling rate as the DAC. A real time implementation of the above algorithm requires a large amount of computation. Thus, a real time implementation of this algorithm is not practical for DEM DACs. Therefore, a simple method is required to construct a uniformly distributed random signal with a specific PSD for controlling a DEM DAC's interconnection network.

The PSD of a random signal [9; 10] can be shaped using a linear filter [11; 12; 13] and the PDF of a random signal can be shaped using a nonlinear transformation. Figure 1-1 shows the block diagram of such a system. In Figure 1-1, a white random signal, $w(n)$,

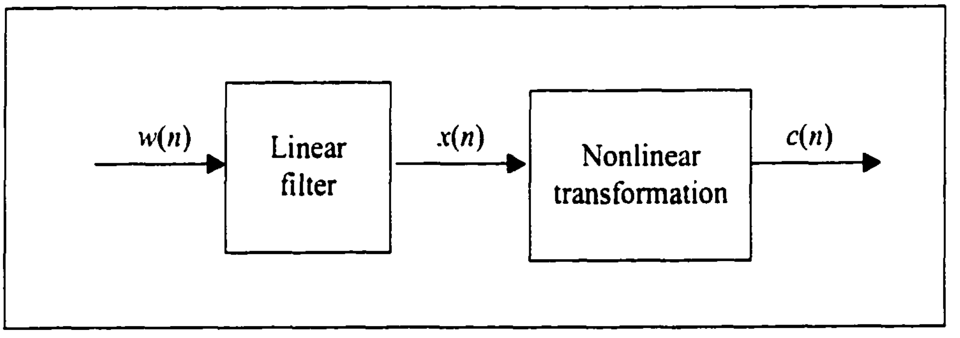


Figure 1-1. Block diagram of a system that generates a colored signal with a specific distribution.

is filtered using a linear filter that shapes the signal's PSD. The PDF of the filtered signal, $x(n)$, is shaped by a nonlinear transformation. The resulting random signal, $c(n)$, will be an appropriate uniformly distributed colored signal if the nonlinear transformation does not significantly alter the spectrum of $x(n)$. For example, a lowpass filtered random signal, $x(n)$, can be generated by passing a white random signal, $w(n)$, through a lowpass filter. The PDF of the colored random signal, $x(n)$, can be shaped using a nonlinear transformation. If the nonlinear transformation does not significantly alter the spectrum of $x(n)$, then the output, $c(n)$, is a colored random signal with a specific distribution. This approach is used in this thesis to generate a uniformly distributed random signal with a specific PSD.

In this thesis, it is also shown that the two most significant bits of a B bit colored binary random signal have more influence on the signal's PSD than the B–2 least significant bits. Therefore, a B bit colored random signal can be adequately generated by generating the two most significant bits as colored bits and the $B-2$ least significant bits as white bits. The two most significant bits are generated using a finite state machine (FSM) and the $B-2$ least significant bits are generated using Linear Feedback Shift Registers (LFSRs) [14; 15; 16]. The real time implementation that generates a uniformly distributed colored random signal using a FSM and LFSRs requires less hardware when compared to the hardware requirements for generating all the B bits as colored bits. Therefore, the real time implementation can be operated at the clock speed of a DEM DAC.

CHAPTER2

BACKGROUND INFORMATION

The algorithm developed in this thesis constructs a uniformly distributed random signal with a specific power spectral density (PSD). The PSD of the random signal is shaped by filtering a white random signal using a linear phase filter and then filtering the resulting signal by a nonlinear filter that shapes the PDF. The uniformly distributed colored random signal is generated using a finite state machine (FSM) and Linear Feedback Shift Registers (LFSRs) [14; 15; 16]. The FSM is used to generate the two most significant bits (MSBs) of the uniformly distributed colored random signal and the remaining bits are generated using LFSRs. Because LFSRs are very simple hardware structures, they are well suited for real time implementations.

2.1 A DEM Flash DAC Architecture

Figure 2-1 shows the architecture of a B bit DEM flash DAC. The DAC's input signal, $x(n)$, is a B bit digital signal where $x_0 \leq x(n) \leq x_0 + 2^B - 1$. The natural binary converter transforms the digital input signal, $x(n)$, into the B bit natural binary signal, $\chi(n)$, where $\chi(n) = x(n) - x_0$ which implies that $0 \leq \chi(n) \leq 2^B - 1$. The modified thermometer coder converts the natural binary coded signal, $\chi(n)$, into a 2^B bit modified thermometer encoded signal, $t(n)$. The interconnection network, controlled by the signal $c(n)$, connects the 2^B bits of the modified thermometer encoded signal, $t(n)$, to the 2^B unit DACs. The control signal, $c(n)$, can be a deterministic signal or a stochastic signal. The interconnection network's output, $g(n)$, activates $\chi(n)$, or $x(n) - x_0$, unit DACs and

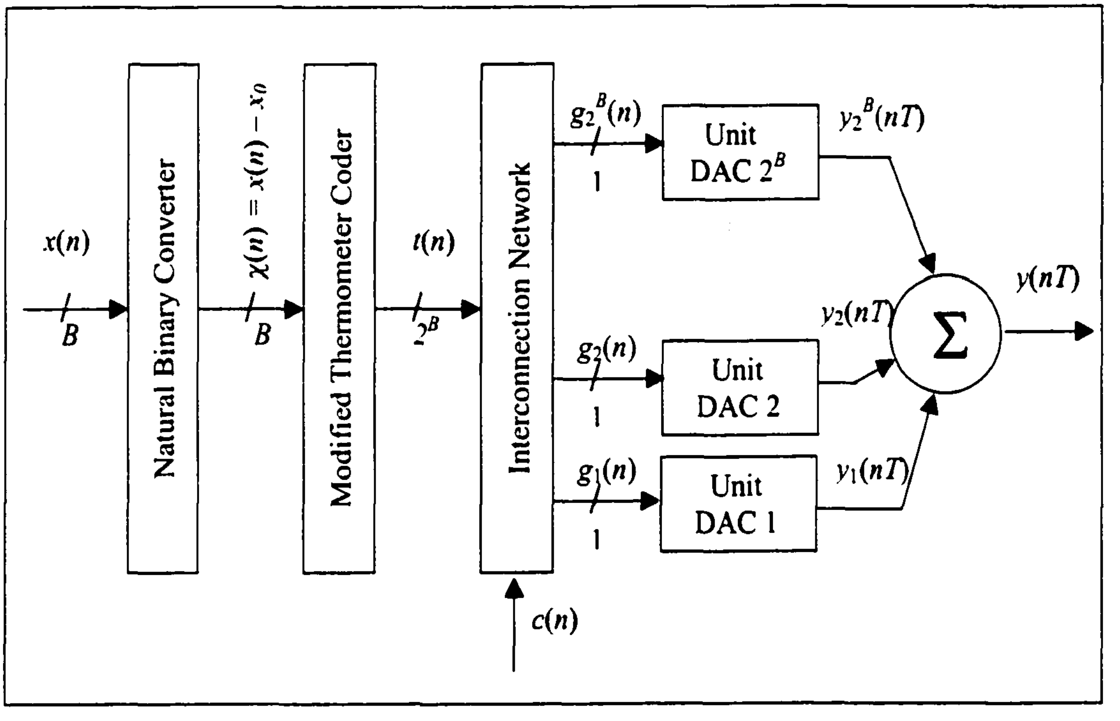


Figure 2-1. A B bit dynamic element matching flash DAC architecture.

deactivates the remaining 2^B-$\chi(n)$, or 2^B-$x(n) + x_0$, unit DACs irrespective of the control signal, $c(n)$. If each activated unit DAC generates an analog signal, a, and each deactivated unit DAC generates an analog signal d, then the DAC's quantization step sizes or code widths, q, are the difference between a and d, that is $q = a\text{-}d$. The DAC's output, $y(nT)$, which is the sum of all the unit DAC outputs, can be written as

$$y(nT) \;=\; a\chi(n) + d[2^B\text{-}\chi(n)]$$

$$= (a\text{-}d)\chi(n) + d2^B$$

$$= q[x(n) - x_0] + d2^B \qquad (2\text{-}1)$$

The mismatched components between each of the unit DACs prevent the analog output of the activated unit DACs as well as the deactivated unit DACs from being identical. Therefore, the DAC's quantization step sizes are not constant which results in degradation of the DAC's performance. To improve the DAC's performance, the DEM DAC's interconnection network dynamically alters the mapping between the input signal, $x(n)$, and the mismatched unit DACs so that the time averages of the activated unit DAC outputs are nearly equal. If the interconnection network's control signal, $c(n)$, is deterministic, the mapping between the DAC's input signal, $x(n)$, and the 2^B unit DACs is deterministic, and the DAC is said to as a deterministic DEM DAC. Similarly, if the interconnection network's control signal, $c(n)$, is stochastic, the mapping between the DAC's input signal, $x(n)$, and the 2^B unit DACs is stochastic, and the DAC is said to as a stochastic DEM DAC.

2.2 Linear Feedback Shift Register

In this thesis, a uniformly distributed random signal with a specific PSD is implemented using a finite state machine (FSM) and Linear Feedback Shift Registers (LFSRs) [14; 15; 16]. LFSRs can generate a uniformly distributed white random signal of any length using D flip-flops and XOR gates.

An LFSR of length n consists of n registers (or n stages) which are numbered 0, 1, … , n-1 and each register is capable of storing one bit. A clock controls the data movement between the registers. During each clock pulse, the content of stage i is moved to stage i-

1 for each i, $1 \leq i \leq n$-1. Before moving the content of stage i to stage i-1, the new content of stage n-1 is first calculated by modulo 2 addition of the contents of a subset of the set of stages {0, 1, ... , n-1}. The elements of this subset are often called taps. The most commonly used register is a D flip-flop and modulo 2 addition is performed using an XOR gate. The length of the binary sequence generated using the LFSR depends upon the number of taps and the initial state of the registers (the state of the registers during the first clock cycle).

In an LFSR, a polynomial is used to represent a binary code [14; 15; 16; 17; 18; 19]. To illustrate, consider a polynomial, $f(x)$, of degree n where

$$f(x) = 1 + a_1x^1 + \ldots. + a_{n-1}x^{n-1} + a_nx^n, \qquad (2\text{-}2)$$

$a_k \in \{0,1\}$ and $1 \leq k \leq n$. This polynomial determines the taps of the LFSR [14; 15; 16; 17; 18; 19]. Figure 2-2 shows an LFSR of length n. In Figure 2-2, the polynomial associated with the LFSR is $f(x) = 1 + a_1x^1 + \ldots. + a_{n-1}x^{n-1} + a_nx^n$ and the number of stages in the LFSR is equal to the degree of the polynomial, $f(x)$. In Figure 2-2, s_j, $j \geq n$, is the feedback bit and is the new content of stage n-1 during each clock pulse. The feedback bit, s_j, is calculated by modulo 2 addition of taps. If the initial content of stage i is $s_i \in \{0, 1\}$ for each i, $0 \leq i \leq n$-1, then $[s_{n-1}, \ldots., s_1, s_0]$ is called the initial state of the LFSR. The output, s, of stage 0 is the random sequence, $\{s_0, s_1, s_2, \ldots.\}$, generated by the LFSR and it is uniquely determined by the recursion

$$s_j = (a_1s_{j-1} \oplus a_2s_{j-2} \oplus \ldots.\oplus a_ns_{j-n}) \text{ for } j \geq n \qquad (2\text{-}3)$$

where $\oplus$ is modulo 2 operator.

The polynomial, $f(x)$, is called an irreducible polynomial if $f(x)$ cannot be factored or written as a product of two polynomials [15; 16]. Every irreducible polynomial, with coefficients 0 or 1 and of degree $n > 1$, divides the polynomial $1 - x^r$ where $r = 2^n - 1$, with a zero remainder. If $f(x)$ is an irreducible polynomial of degree n, then the LFSR generates a sequence with a period of length $2^n - 1$ [15; 16] and the sequence is said to be

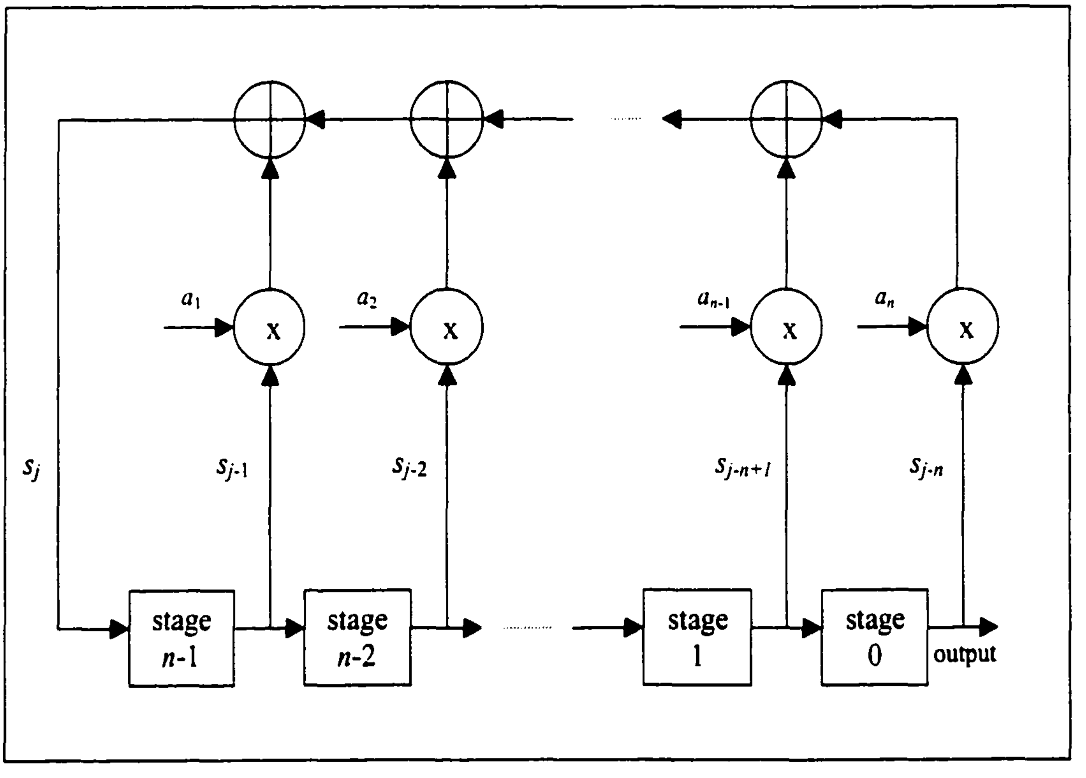


Figure 2-2 An LFSR of length n.

a *maximal sequence*. The output of an LFSR has a period of length l, where $1 \leq l \leq 2^n - 1$ and the period length, l, depends on the polynomial associated with the LFSR. In a *maximal sequence* of length $2^n - 1$, there are $2^{n-1} - 1$ zeros and 2^{n-1} ones [15; 16]. Figure 2-3 shows the almost uniform distribution of a *maximal sequence* of length $2^n - 1$.

For example, consider a polynomial

$$f(x) = 1 + x + x^3. \tag{2-4}$$

The polynomial in Equation (2-4) is an irreducible polynomial because $f(x)$ cannot be written as a product of two polynomials. Also, the polynomial, $f(x) = 1+x+x^3$, divides the polynomial $1-x^7$ with a zero remainder; that is,

$$(1-x^7)/(1+x+x^3) = (1-x)(1+x^2+x^3) \tag{2-5}$$

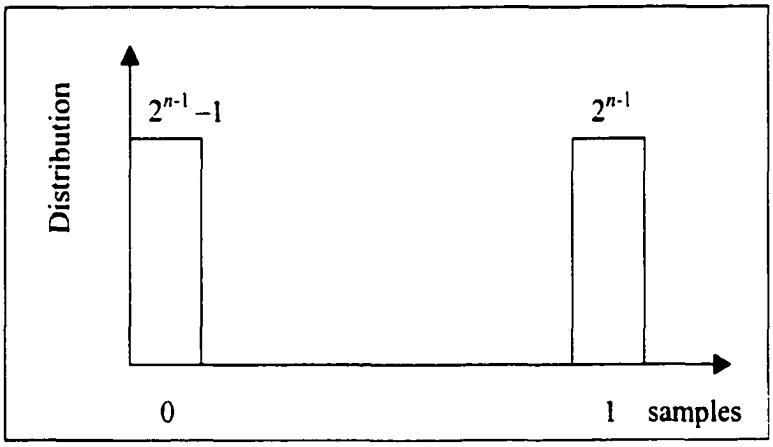


Figure 2-3. Distribution of a *maximal sequence.*

Because the polynomial, $f(x) = 1+x+x^3$, is an irreducible polynomial, the period of the generated sequence will be 2^3-1, which is 7, and the sequence is said to be a *maximal sequence*. Figure 2-4 shows the three-stage LFSR that generates a binary sequence with a period of length 2^3-1, which is 7. The output, s, of stage 0 is determined by the recursion $s_j = (s_{j-1} \oplus s_{j-3})$ for $j \geq 3$. If the initial state of the LFSR is [1 1 1], then Figure 2-5 shows the binary sequence generated using the three-stage LFSR. This figure shows that the generated sequence has a period of 2^3-1, which is 7, and the output of stage 2 and stage 1 is same as the output of stage 0 except for a delay. Therefore, the output of stage 2 and stage 1 is a phase-shifted replica of the output of stage 0. Because the generated sequence is a *maximal sequence*, there are $2^{3-1}-1$ zeros and 2^{3-1} ones in the sequence. Figure 2-6 shows the distribution of the sequence generated by stage 0.

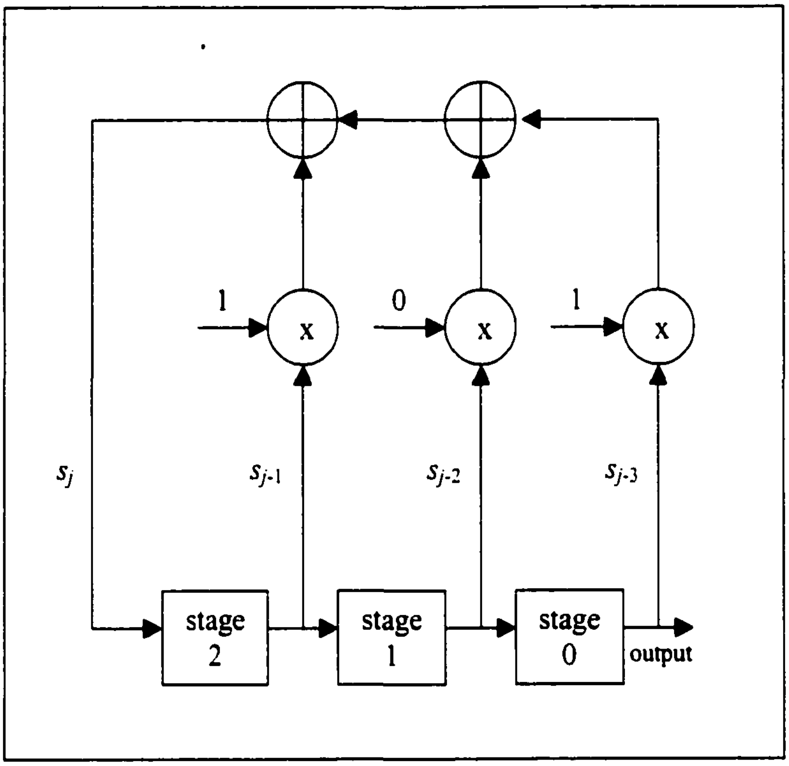


Figure 2-4 An LFSR of length 3.

stage 2	stage 1	stage 0
1	1	1
0	1	1
1	0	1
0	1	0
0	0	1
1	0	0
1	1	0
1	1	1

Figure 2-5. Sequence generated using the three-stage LFSR.

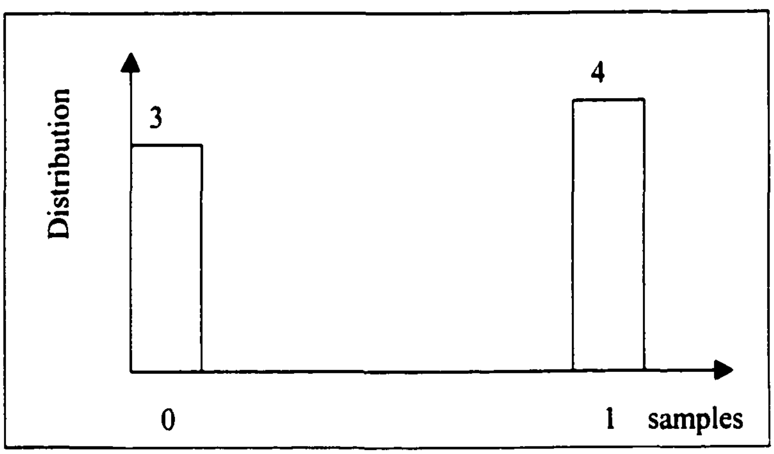


Figure 2-6. Distribution of the sequence generated by stage 0.

2.3 Correlation Function and Covariance function

The randomness of a sequence can be measured using the correlation function and the covariance function of the sequence.

If $x(n_1)$ and $x(n_2)$ are two random variables at time instant n_1 and n_2, then the correlation function, R_x, is defined as

$$R_x(n_1, n_2) = E[x(n_1)x(n_2)] \tag{2-6}$$

where $E[x(n_1)x(n_2)]$ is the expected value of the product of $x(n_1)$ and $x(n_2)$. If the random signal is assumed to be a wide-sense stationary signal, then the correlation function, R_x, becomes independent of the time origin and depends only on the time difference between n_2 and n_1. Therefore, for a wide sense stationary signal, Equation (2-6) can be written as

$$R_x(\tau) = E[x(n_1)x(n_1 + \tau)] \tag{2-7}$$

where $R_x(\tau)$ is the correlation function of a wide-sense stationary signal and $\tau = n_2\text{-}n_1$. If $\bar{x}(n_1)$ and $\bar{x}(n_2)$ are the mean or the expected value of the random variables $x(n_1)$ and $x(n_2)$ respectively, then the covariance function, C_x, is defined as

$$C_x(n_1, n_2) = E\{\, [x(n_1)\text{-}\ \bar{x}(n_1)]\ [x(n_2)\text{-}\ \bar{x}(n_2)]\ \}. \tag{2-8}$$

If the random signal is assumed to be a wide-sense stationary signal, then the correlation function, C_x, becomes independent of the time origin and depends only on the time difference between n_2 and n_1. Therefore, for a wide sense stationary signal, Equation (2-8) can be written as

$$C_x(\tau) = E\{\, [x(n_1)\text{-}\bar{x}(n_1)]\ [x(n_1+\tau)\text{-}\bar{x}(n_1+\tau)]\ \} \tag{2-9}$$

where $C_x(\tau)$ is the covariance function of a wide-sense stationary signal, $\tau = n_2\text{-}n_1$. Equations (2-8) and (2-9) show that the covariance function, $C_x(\tau)$, is same as the correlation function, $R_x(\tau)$, if the expected value or mean of the random variables is zero. If the mean is nonzero, $C_x(\tau) = E[x(n_1)x(n_1+\tau)] + E[\bar{x}(n_1)\bar{x}(n_1+\tau)] - E[x(n_1)\bar{x}(n_1+\tau)] - E[\bar{x}(n_1)x(n_1+\tau)]$ and the covariance function is a shifted version of the correlation function.

The power spectral density, $S_x(e^{j\omega})$, of the random variable, $x(n)$, is defined as the Fourier Transform of the correlation function, $R_x(\tau)$; that is,

$$S_x(e^{j\omega}) = \mathbf{F}\{R_x(\tau)\}$$

$$= \sum_{\tau=-\infty}^{\infty} R_x(\tau)\, e^{-j\omega\tau}. \qquad (2\text{-}10)$$

If $x(n)$ is assumed to be an ergodic signal where the time averages are equivalent to ensemble averages, then the autocorrelation, $R_x(\tau)$, of the random variable $x(n)$ can be written as

$$R_x(\tau) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n1=-N}^{N} x(n_1)\, x(n_1+\tau) \qquad (2\text{-}11)$$

and therefore the PSD, $S_x(e^{j\omega})$, can be written as

$$S_x(e^{j\omega}) = \mathbf{F}\{R_x(\tau)\}$$

$$= \sum_{\tau=-\infty}^{\infty} \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n1=-N}^{N} x(n_1)\, x(n_1+\tau)\, e^{-j\omega\tau} \qquad (2\text{-}12)$$

Interchanging the summations, Equation (2-12) can be written as

$$S_x(e^{j\omega}) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n1=-N}^{N} x(n_1) \sum_{\tau=-\infty}^{\infty} x(n_1+\tau)\, e^{-j\omega\tau} \qquad (2\text{-}13)$$

Substituting $m = n_1+\tau$ in (2-13),

$$S_x(e^{j\omega}) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n1=-N}^{N} x(n_1)\, e^{j\omega n1}\; X(e^{j\omega}) \qquad (2\text{-}14)$$

$$= \lim_{N \to \infty} \frac{1}{2N+1} X(e^{-j\omega}) X(e^{j\omega}) \tag{2-15}$$

If $x(n)$ is real then,

$$|X(e^{-j\omega})| = |X(e^{j\omega})|. \tag{2-16}$$

Substituting Equation (2-16) in Equation (2-15),

$$S_x(e^{j\omega}) = \lim_{N \to \infty} \frac{1}{2N+1} |X(e^{j\omega})|^2 \tag{2-17}$$

If N is a finite value, then Equation (2-17) can be written as

$$S_x(e^{j\omega}) = \frac{|X(e^{j\omega})|^2}{2N+1}$$

$$= \frac{|X(e^{j\omega})|^2}{T} \tag{2-18}$$

where $T = 2N+1$.

Using the correlation function and the covariance function, the randomness of the binary sequence generated using the three-stage LFSR in Section 2.2, can be measured. Figure 2-7 shows the correlation function, $R_x(\tau)$. This figure shows that the maximum correlation occurs at $\tau = 7$. Figure 2-8 shows the covariance function, $C_x(\tau)$. This figure is same as the correlation function but with a zero mean. If the mean is nonzero, then the sequence will have a DC shift in the power spectrum. If this DC shift is not desired, then the covariance function is used instead of the correlation function. Figure 2-9 shows the PSD and the distribution of ones and zeros of the generated sequence. Because the sequence is a *maximal sequence*, it has 4 ones and 3 zeros, as expected. The PSD shows that the sequence generated using the LFSR is almost white because the PSD is present over the entire frequency spectrum. The PSD is not constant over the entire spectrum because the length of the generated sequence is very small. If the length of the sequence generated using the LFSR is very long (in the order of thousands), then the PSD becomes more constant over the entire frequency spectrum.

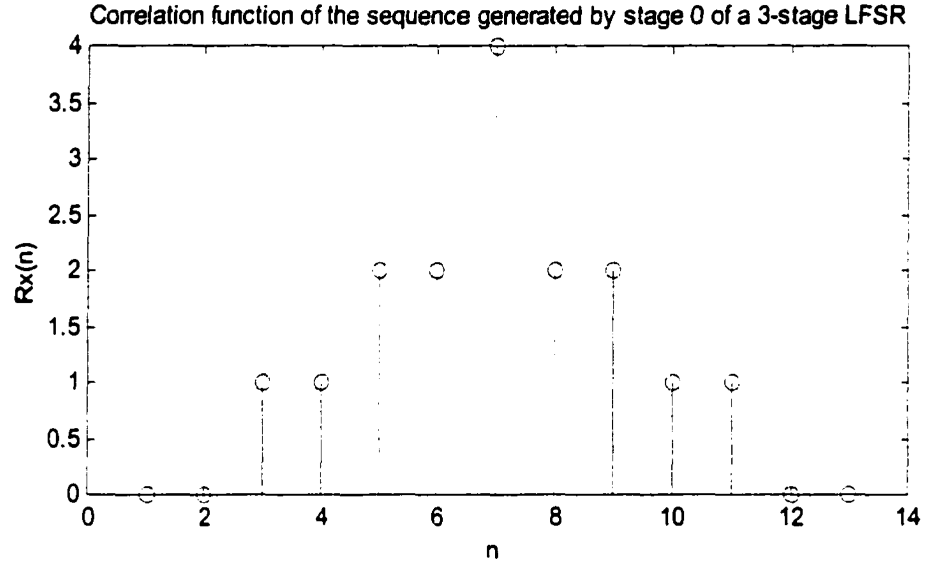


Figure 2-7. Correlation function of the sequence generated by stage 0 of a three-stage LFSR.

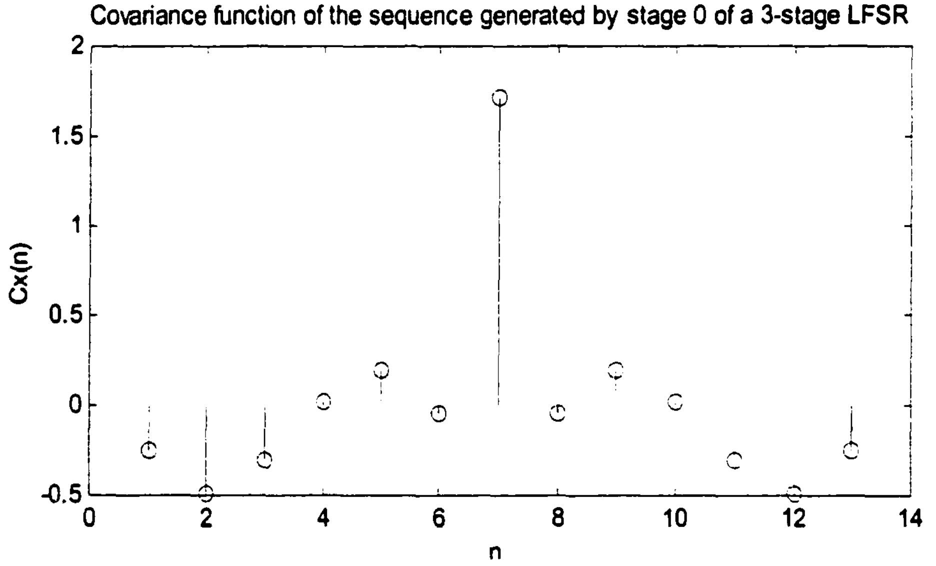


Figure 2-8. Covariance function of the sequence generated by stage 0 of a three-stage LFSR.

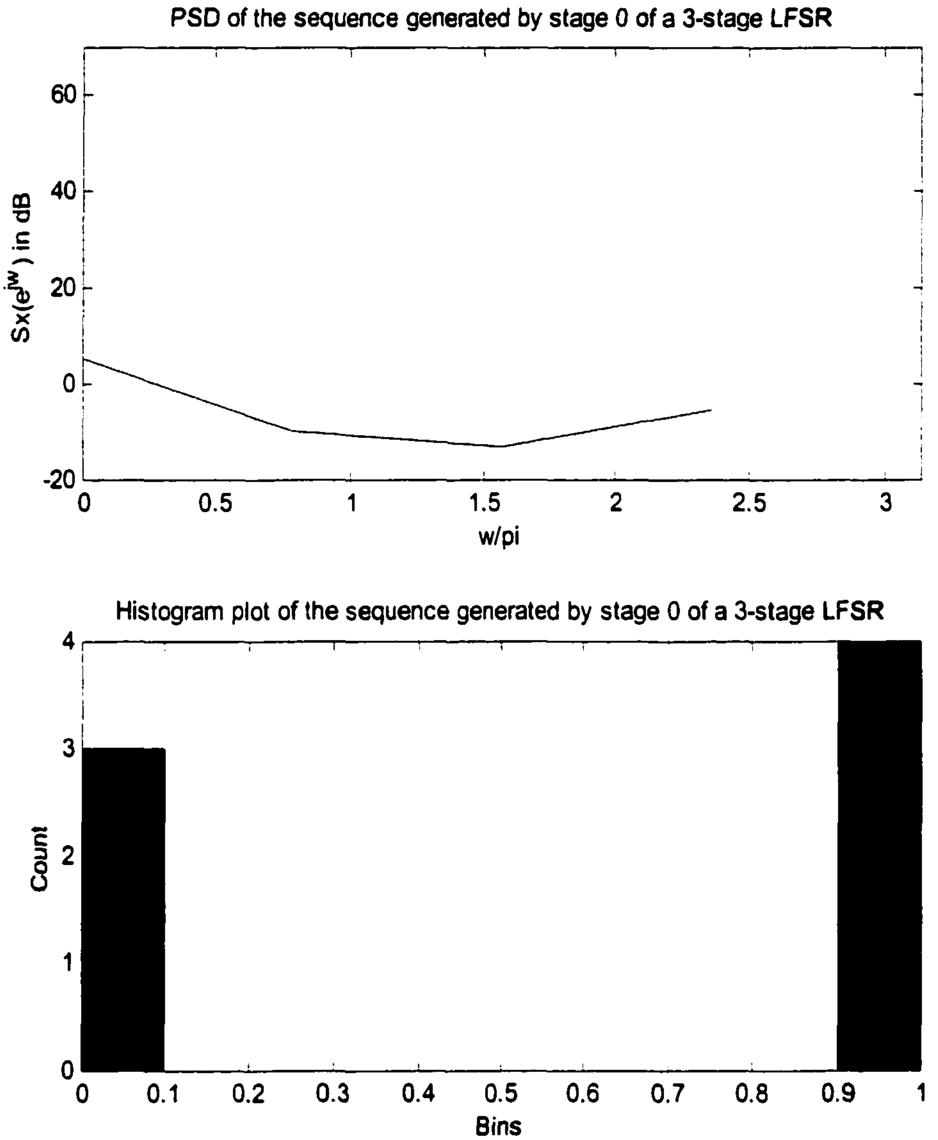


Figure 2-9. PSD and histogram plot of the sequence generated by stage 0 of a three-stage LFSR.

CHAPTER 3

METHODOLOGY OF CONSTRUCTION OF A UNIFORMLY DISTRIBUTED COLORED RANDOM SIGNAL

In a flash DEM DAC, many interconnection networks dynamically rearrange the mapping between the digital input signal and the mismatched unit DACs so that the time averages of the activated unit DAC outputs are equal and the time averages of the deactivated unit DAC outputs are equal. For these interconnection networks, the required control signal is a uniformly distributed random signal with a specific power spectral density (PSD) [3; 4]. Figure 3-1 shows the probability distribution function (PDF) of a control signal that is uniformly distributed between α and β. In this thesis, an algorithm

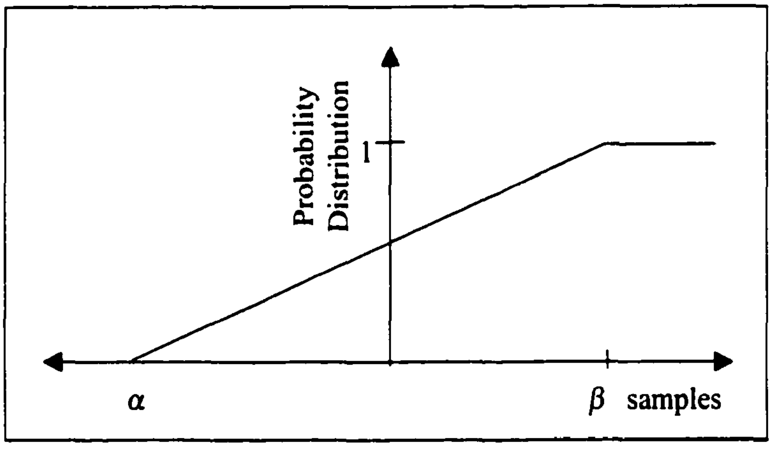


Figure 3-1. PDF of a uniformly distributed control signal.

is developed for constructing a uniformly distributed random signal with a specific PSD. Figure 3-2 shows the block diagram of this algorithm that generates a uniformly distributed colored random signal, $c(n)$, by filtering a white random signal, $w(n)$, using

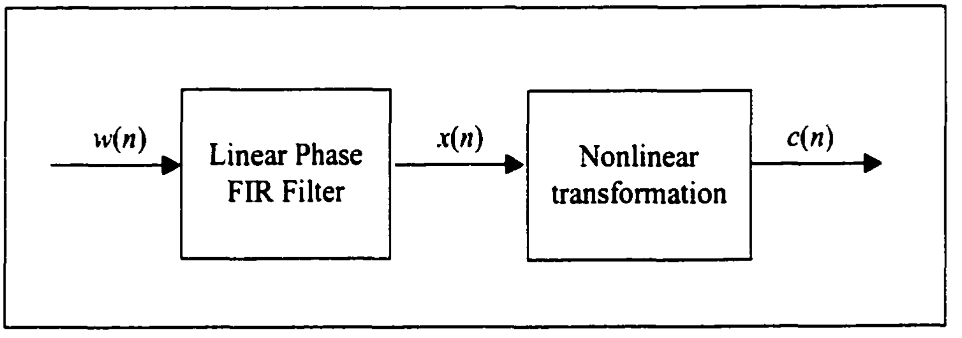


Figure 3-2. Block diagram of a system that generates a uniformly distributed colored random signal.

a linear phase FIR filter that shapes the PSD and then filtering the colored random signal, $x(n)$, using a nonlinear filter that shapes the PDF.

3.1 Transformation of a random signal

The PSD of a random signal can be shaped by filtering a white random signal using a linear phase FIR filter. The frequency response, $H(e^{j\omega})$, of the filter in Figure 3-3 can be used to describe any frequency selection filter with a single passband having a lower cutoff frequency of ω_l and an upper cutoff frequency of ω_u. Because the phase of the filter is zero, $H(e^{j\omega})$ has complex conjugate symmetric about $\omega = 0$. Therefore the filter's impulse response, $h(n)$, is real. Using the inverse Fourier Transform, the filter's impulse response, $h(n)$, can be written in terms of ω_l and ω_u as

$$h(n) = (1/2\pi) \int_{-\pi}^{\pi} H(e^{j\omega})\, e^{j\omega}\, d\omega$$

$$= \begin{cases} (1/\pi n)\,[\, \sin(\omega_u n) - \sin(\omega_l n)\,] & \text{when } n \neq 0 \\ (1/\pi)\,[\omega_u - \omega_l\,] & \text{when } n = 0 \end{cases} \qquad (3\text{-}1)$$

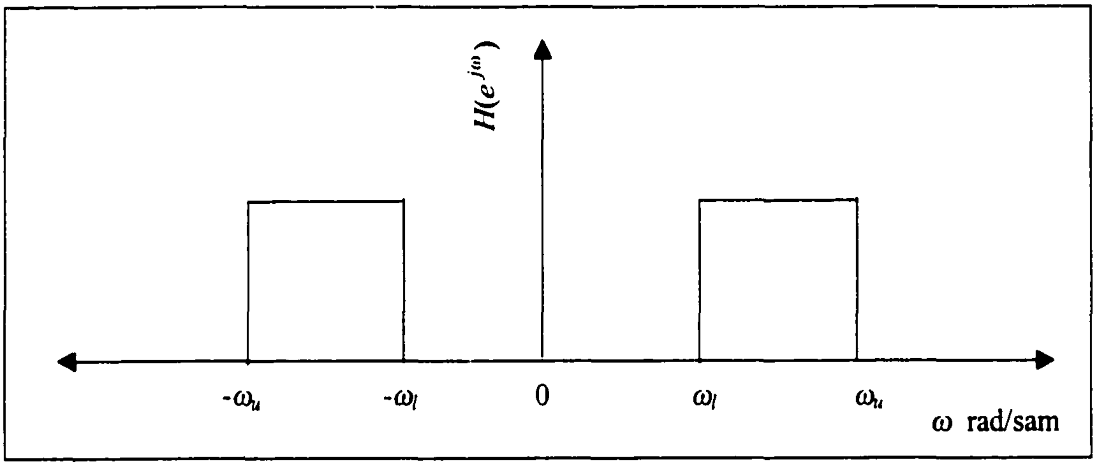


Figure 3-3. Frequency response of the filter.

If the phase is non-zero, then the frequency response, $H(e^{j\omega})$, can be written as

$$H(e^{j\omega}) = H(\omega)\, e^{j\theta(\omega)} \qquad (3\text{-}2)$$

where $H(\omega)$ is the zero phase frequency response of the system and

$$\theta(\omega) = \frac{(N\text{-}1)}{2}\, \omega \qquad (3\text{-}3)$$

where N is the length of the impulse response, $h(n)$, of the filter.

For example, if the required random signal is a bandlimited signal with a lower cutoff frequency of ω_l and an upper cutoff frequency of ω_u, then a white random signal would be filtered using a linear phase FIR filter that has the frequency spectrum, $H(e^{j\omega})$, as shown in Figure 3-3. In Equation (3-1), if $\omega_l = 0$ and $\omega_u \neq \pi$, then the linear phase FIR filter is a lowpass filter and if $\omega_l \neq 0$ and $\omega_u = \pi$, then the linear phase FIR filter is a highpass filter.

If $w(n)$ is a white random input signal, then the filtered signal, $x(n)$, is

$$x(n) = w(n) * h(n)$$

$$= \sum_{k=-\infty}^{\infty} w(k)\, h(n-k) \qquad (3\text{-}4)$$

where * represents the convolution of the white random signal, $w(n)$, with $h(n)$. If $R_w(n)$ is the autocorrelation of $w(n)$ then the PSD, $S_w(e^{j\omega})$, of the white random signal, $w(n)$, is

$$S_w(e^{j\omega}) = \mathbf{F}\{R_w(n)\}$$

$$= \sum_{n=-\infty}^{\infty} R_w(n)\, e^{-j\omega n} \qquad (3\text{-}5)$$

where $\mathbf{F}\{R_w(n)\}$ is the Fourier Transform of $R_w(n)$. If the white random signal, $w(n)$, is an ergodic signal having a finite time period, T, then the PSD, $S_w(e^{j\omega})$, is

$$S_w(e^{j\omega}) = \frac{|\,W(e^{j\omega})\,|^2}{T} \qquad (3\text{-}6)$$

Because $w(n)$ is a white signal, the PSD, $S_w(e^{j\omega})$, of $w(n)$ will be constant over the entire frequency spectrum.

If $x(n)$ is also an ergodic signal having a finite time period, T, then the PSD, $S_x(e^{j\omega})$, of the colored random signal, $x(n)$, is

$$S_x(e^{j\omega}) = \frac{|\,X(e^{j\omega})\,|^2}{T} \qquad (3\text{-}7)$$

$$= \frac{|\,W(e^{j\omega})\,H(e^{j\omega})\,|^2}{T}$$

The Central Limit Theorem, when expressed as a property of convolutions, states that the convolution operation of a large number of positive functions is approximately a normal function [9]. Because $x(n) = \sum_{k=-\infty}^{\infty} w(k)\, h(n-k)$, then the colored random signal, $x(n)$, is an approximately normally distributed signal whether $w(n)$ is a uniformly distributed or normally distributed white signal. To transform the colored random signal's, $x(n)$'s, normal distribution to a uniform distribution, $x(n)$ is filtered using its distribution function; that is,

$$c(n) = F_{x(n)}(x(n)) \tag{3-8}$$

where $c(n)$ is the resulting filtered random signal and $F_{x(n)}(x(n))$ is the distribution function of the normally distributed colored random signal, $x(n)$. Because $x(n)$ is normally distributed,

$$F_{x(n)}(x_o) = \begin{cases} (1/2) + erf(x_o) & x_o \geq 0 \\ (1/2) - erf(x_o) & x_o < 0 \end{cases} \tag{3-9}$$

where *erf* is the error function which is defined as

$$erf(x_o) = (1/sqrt(2\pi)) \int_0^{xo} exp(-t^2/2)\, dt \tag{3-10}$$

To show that the filtered random signal, $c(n)$, is uniformly distributed, let $c_o = F_{x(n)}(x_o)$ and let $F_{c(n)}(c_o)$ be the distribution function of the new random signal; that is,

$$F_{c(n)}(c_o) = P(c(n) \leq c_o) \tag{3-11}$$

Because $F_{x(n)}(x_o)$ is monotonic, $c(n) \leq c_o$ if and only if $x(n) \leq x_o$. Therefore,

$$F_{c(n)}(c_o) = P(c(n) \leq c_o)$$

$$= P(x(n) \leq x_o)$$

$$= F_{x(n)}(x_o)$$

$$= c_o \tag{3-12}$$

Equation (3-12) shows that the distribution function, $F_{c(n)}(c_o)$, is equal to c_o, which implies that $c(n)$ is uniformly distributed. The uniformly distributed random signal, $c(n)$, exists in the same interval as that of the normally distributed colored random signal, $x(n)$.

The frequency spectrum, $C(e^{j\omega})$, of the uniformly distributed random signal, $c(n)$, is

$$C(e^{j\omega}) = \mathbf{F}\ \{c(n)\}$$

$$= \mathbf{F}\ \{F_{x(n)}(\ x(n))\ \} \qquad \{\text{ from Equation (3-8) }\}$$

$$= \sum_{n=-\infty}^{\infty} F_{x(n)}(x(n))\, e^{-j\omega n} \tag{3-13}$$

Equation (3-13) shows that the frequency spectrum, $C(e^{j\omega})$, of the uniformly distributed random signal is the Fourier Transform of the distribution function, $F_{x(n)}(x(n))$, of the normally distributed colored random signal, $x(n)$. It is very difficult to get a closed form expression for Equation (3-13) because the distribution function, $F_{x(n)}(x(n))$, of the normally distributed colored random signal, $x(n)$, is represented in terms of the error function, *erf*, and it exists in a finite interval.

Instead of finding a closed form expression for the Fourier Transform of an error function, it is simulated in Matlab. From the simulation it is found that the frequency spectrum, $C(e^{j\omega})$, exists in the same interval as that of the frequency spectrum, $X(e^{j\omega})$, of the normally distributed colored random signal, $x(n)$. If $R_c(n)$ is the autocorrelation of $c(n)$, then the PSD, $S_c(e^{j\omega})$, of $c(n)$ is

$$S_c(e^{j\omega}) = \mathbf{F}\{R_c(n)\}$$

$$= \sum_{n=-\infty}^{\infty} R_c(n)\, e^{-j\omega n} \tag{3-14}$$

where $\mathbf{F}\{R_c(n)\}$ is the Fourier Transform of $R_c(n)$. Because $c(n)$ is assumed to be an ergodic signal having a finite time period, T, the PSD, $S_c(e^{j\omega})$, is

$$S_c(e^{j\omega}) = \frac{|C(e^{j\omega})|^2}{T} \tag{3-15}$$

Because $C(e^{j\omega})$ exists in the same interval as that of $X(e^{j\omega})$, the PSD, $S_c(e^{j\omega})$, of $c(n)$ will also exist in the same interval as that of the PSD, $S_x(e^{j\omega})$, of the normally distributed colored random signal, $x(n)$. Therefore, $c(n)$ is a uniformly distributed colored random signal.

As described, a uniformly distributed colored random signal can be constructed by filtering a uniformly distributed or normally distributed white random signal with a linear phase FIR filter and then filtering the resulting output by a nonlinear filter, the error function. The result is a uniformly distributed colored random signal, $c(n)$, and this random signal can be used in a B bit DEM DAC as interconnection network's control signal. Appendix A contains MATLAB code for the algorithm.

3.2 Examples to illustrate the effectiveness of the algorithm

To illustrate the effectiveness of the algorithm, several uniformly distributed colored random signals are generated with different spectral densities. Figure 3-4 shows the impulse response, $h(n)$, and the frequency spectrum, $H(e^{j\omega})$, (in dB) of a lowpass filter with a cutoff frequency of $\pi/3$ and of order 99. Figure 3-5 shows the autocorrelation, $R_w(n)$, and the PSD, $S_w(e^{j\omega})$, of a normally distributed white random signal, $w(n)$, with zero mean and of length 925. Because the random input signal is white, the PSD, $S_w(e^{j\omega})$, is present over the entire frequency spectrum. The normally distributed colored random signal, $x(n)$, is obtained by filtering the normally distributed white random signal, $w(n)$, using the linear phase FIR filter, $h(n)$. Figure 3-6 shows the autocorrelation, $R_x(n)$, and the PSD, $S_x(e^{j\omega})$, of the normally distributed colored random signal, $x(n)$, of length 1024 and with a cutoff frequency of $\pi/3$. The uniformly distributed colored random signal, $c(n)$, is obtained by a nonlinear transformation of the normally distributed colored random signal, $x(n)$. Figure 3-7 shows the autocorrelation, $R_c(n)$, and the PSD, $S_c(e^{j\omega})$, of the uniformly distributed colored random signal, $c(n)$, of length 1024 and with a cutoff frequency of $\pi/3$. Figure 3-7 also shows that the PSD, $S_c(e^{j\omega})$, of $c(n)$ exists in the same interval as that of the PSD, $S_x(e^{j\omega})$, of the normally distributed colored random signal, $x(n)$. Figure 3-8 shows the histogram plot of the normally distributed white random signal, $w(n)$, normally distributed colored random signal, $x(n)$, and uniformly distributed colored random signal, $c(n)$. Figure 3-9 shows the frequency spectrum of the normally distributed white random signal, $w(n)$, normally distributed colored random signal, $x(n)$, and uniformly distributed colored random signal, $c(n)$, with a cutoff frequency of $\pi/3$. Figures 3-10 – 3-15 show the simulation results of the algorithm for constructing a uniformly distributed bandlimited random signal with a lower cutoff frequency of $\pi/3$ and an upper cutoff frequency of $3\pi/4$. Figures 3-16 – 3-21 show the simulation results of the algorithm for constructing a uniformly distributed highpass filtered random signal with a cutoff frequency of $3\pi/4$.

3.3 Examples to illustrate the effects of the design parameters

The examples in this section illustrate the effects that the length of a white random input signal, $w(n)$, and the length of a linear phase FIR filter's impulse response have on the PSD, $S_c(e^{j\omega})$, of the uniformly distributed colored random signal, $c(n)$.

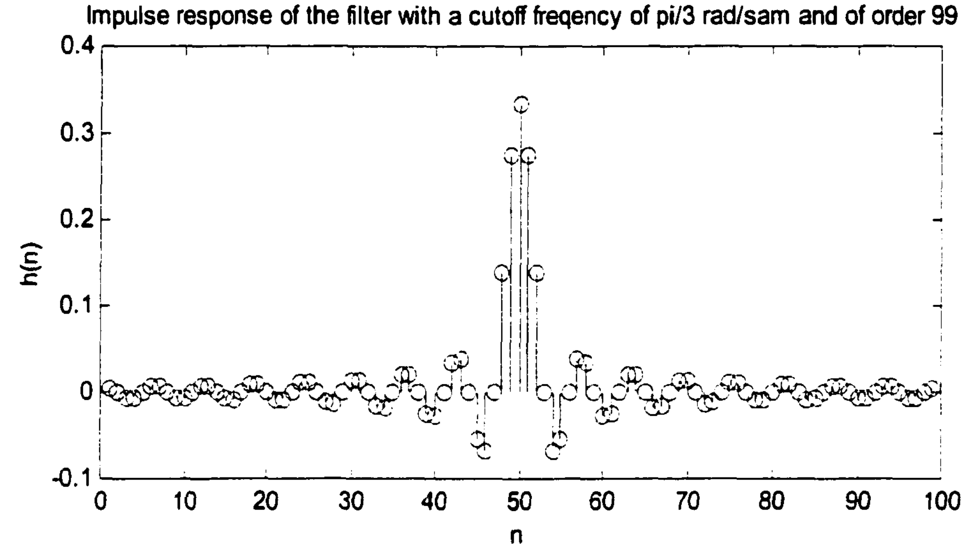
A random signal is white if it has a constant PSD over the entire frequency spectrum. For finite length signals, the magnitude of the spectrum depends upon the length of the random signal. If the length of the white random signal is small (in the order of hundreds), then the PSD is not constant over the entire frequency spectrum. If the length of the white random signal is very long (in the order of thousands), then the PSD is constant over the entire frequency spectrum.

The PSD of the white random signal, $w(n)$, influences the power spectrum of the constructed uniformly distributed colored random signal, $c(n)$. To study the influence of the length of the normally distributed white random signal, $w(n)$, the algorithm is simulated with $w(n)$ having lengths 223 and 2015. Figure 3-22 shows the PSD of $w(n)$ for lengths 223 and 2015. Figure 3-23 shows the corresponding PSD of the uniformly distributed colored random signal, $c(n)$. Figure 3-22 shows that for $w(n)$ of length 223, $S_w(e^{j\omega})$ varies between -20dB and 15dB and for $w(n)$ of length 2015, $S_w(e^{j\omega})$ varies between -9dB and 6dB. Figure 3-23 shows that for $c(n)$ of length 256, $S_c(e^{j\omega})$ in the passband varies between 0dB and 47dB and for $c(n)$ of length 2048, $S_c(e^{j\omega})$ in the passband varies between 26dB and 38dB.

The length of the normally distributed white random signal, $w(n)$, influences the PSD of the normally distributed colored random signal, $x(n)$, which in turn influences the PSD of the uniformly distributed colored random signal, $c(n)$. Therefore, the PSD of the output random signal, $c(n)$, will be constant if the length of the normally distributed white random signal, $w(n)$, is considerably long.

The length of the linear phase FIR filter's impulse response influences the transition at the discontinuity of the PSD, $S_c(e^{j\omega})$, of the colored random signal. The transition from passband to stopband of the filter will be sharp if the length of the linear phase FIR filter's impulse response or the filter order is high [11]. If the transition region of the filter is sharp, then the transition at the discontinuity of $S_c(e^{j\omega})$ will also be sharp.

Figure 3-24 shows the impulse response, $h(n)$, and the frequency response, $H(e^{j\omega})$, of a lowpass filter having $h(n)$ of length 11 and with a cutoff frequency of $\pi/2$. Figure 3-25 shows the impulse response, $h(n)$, and the frequency response, $H(e^{j\omega})$, of a lowpass filter having $h(n)$ of length 63 and with a cutoff frequency of $\pi/2$. Figures (3-24) and (3-25) show that the transition region of the filter having $h(n)$ of length 63 is sharper than the transition region of the filter having $h(n)$ of length 11. Figure 3-26 shows the corresponding PSD, $S_c(e^{j\omega})$, of the uniformly distributed colored random signal, $c(n)$. The transition at the discontinuity of $S_c(e^{j\omega})$ generated by filtering a white random signal using a linear phase FIR filter having an impulse response of length 63 is much sharper than the transition at the discontinuity of $S_c(e^{j\omega})$ generated by filtering a white random signal using a linear phase FIR filter having an impulse response of length 11. So, the transition at the discontinuity of $S_c(e^{j\omega})$ will be sharper if the length of the linear phase FIR filter's impulse response is large.



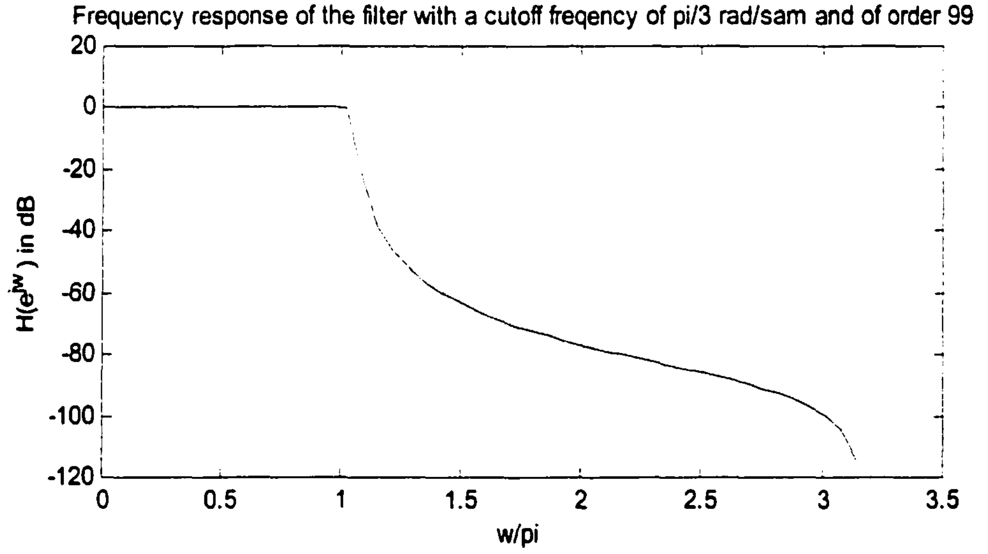


Figure 3-4. Impulse response and frequency response of the filter with a cutoff frequency of $\pi/3$ rad/sam and of order 99.

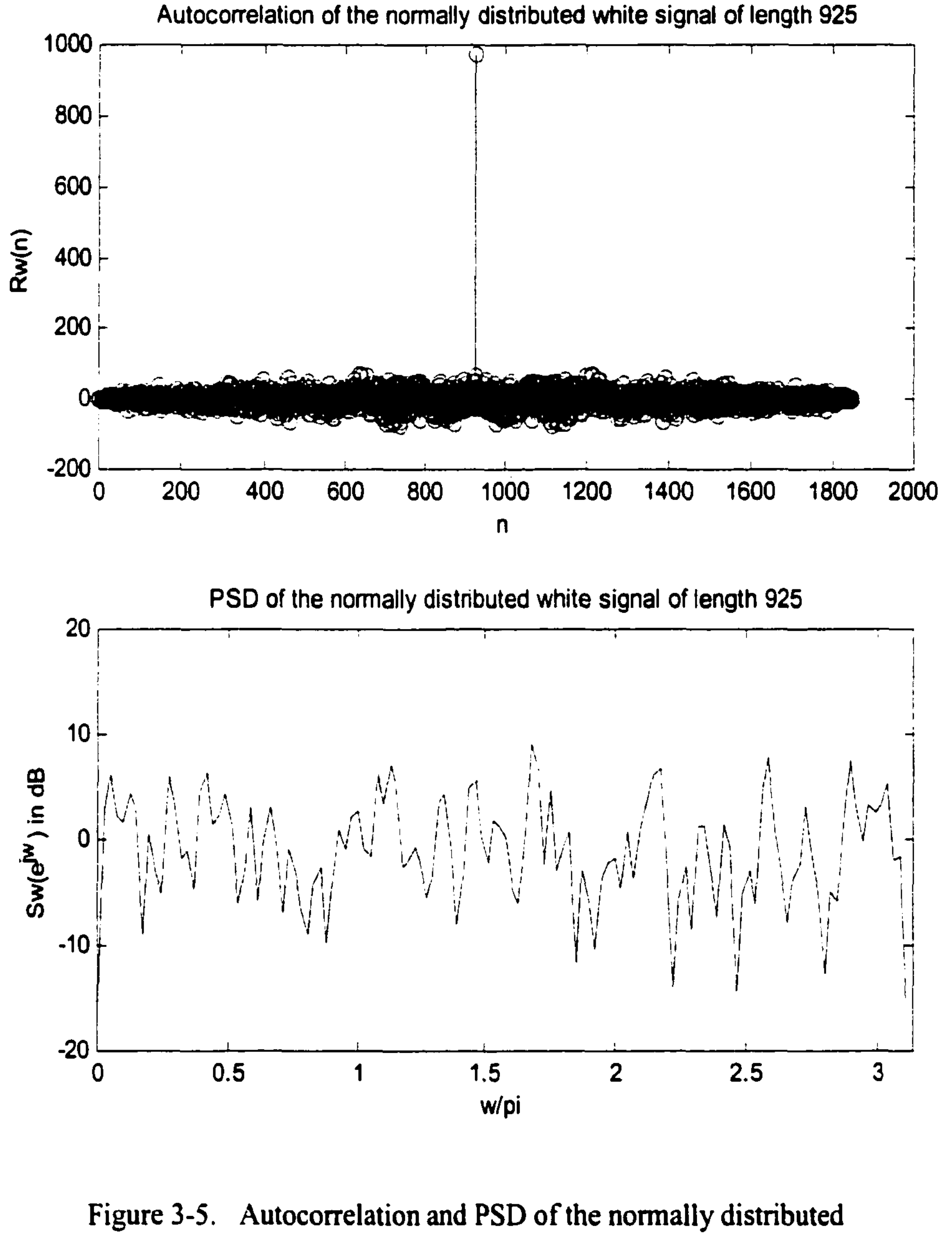


Figure 3-5. Autocorrelation and PSD of the normally distributed white signal of length 925.

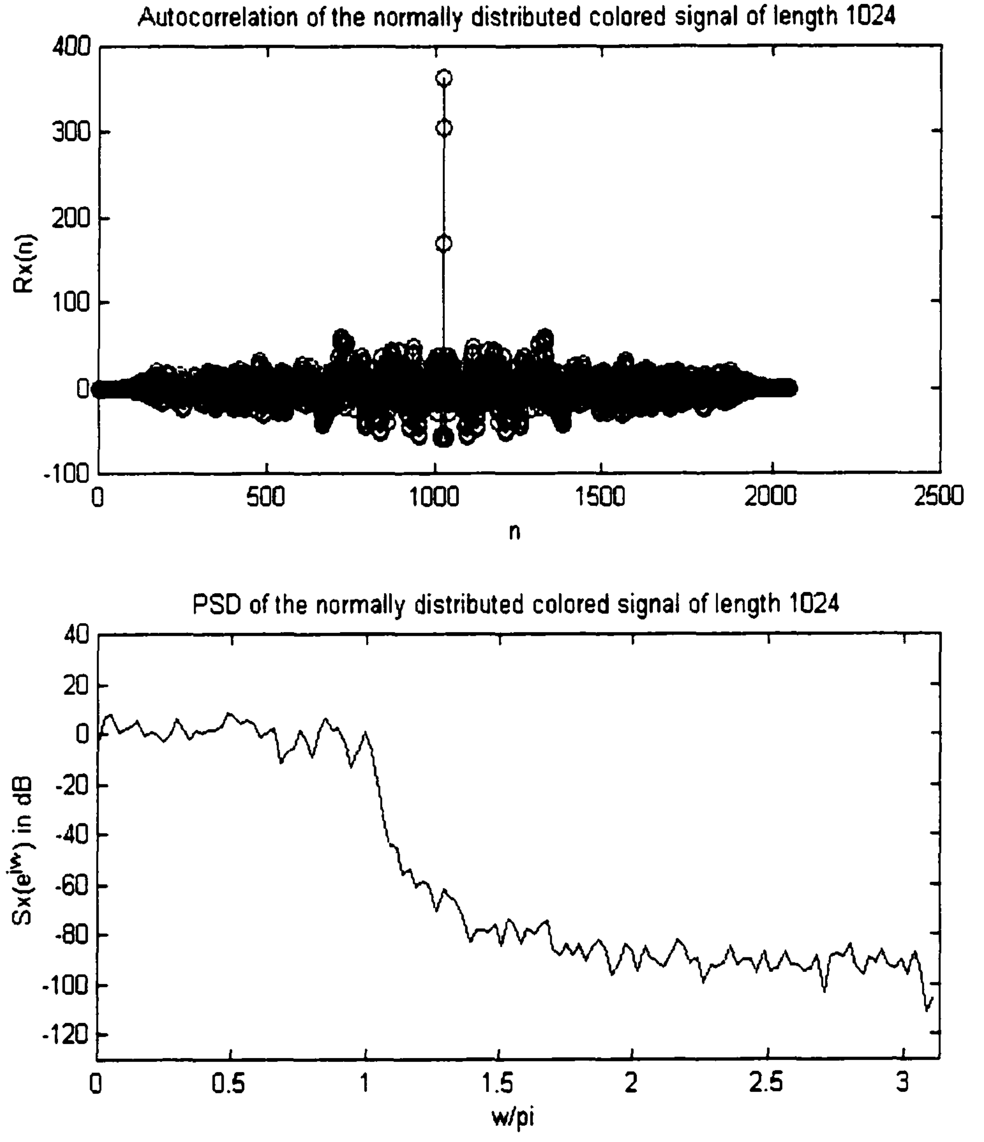


Figure 3-6. Autocorrelation and PSD of the normally distributed colored signal of length 1024 and with a cutoff frequency of $\pi/3$ rad/sam.

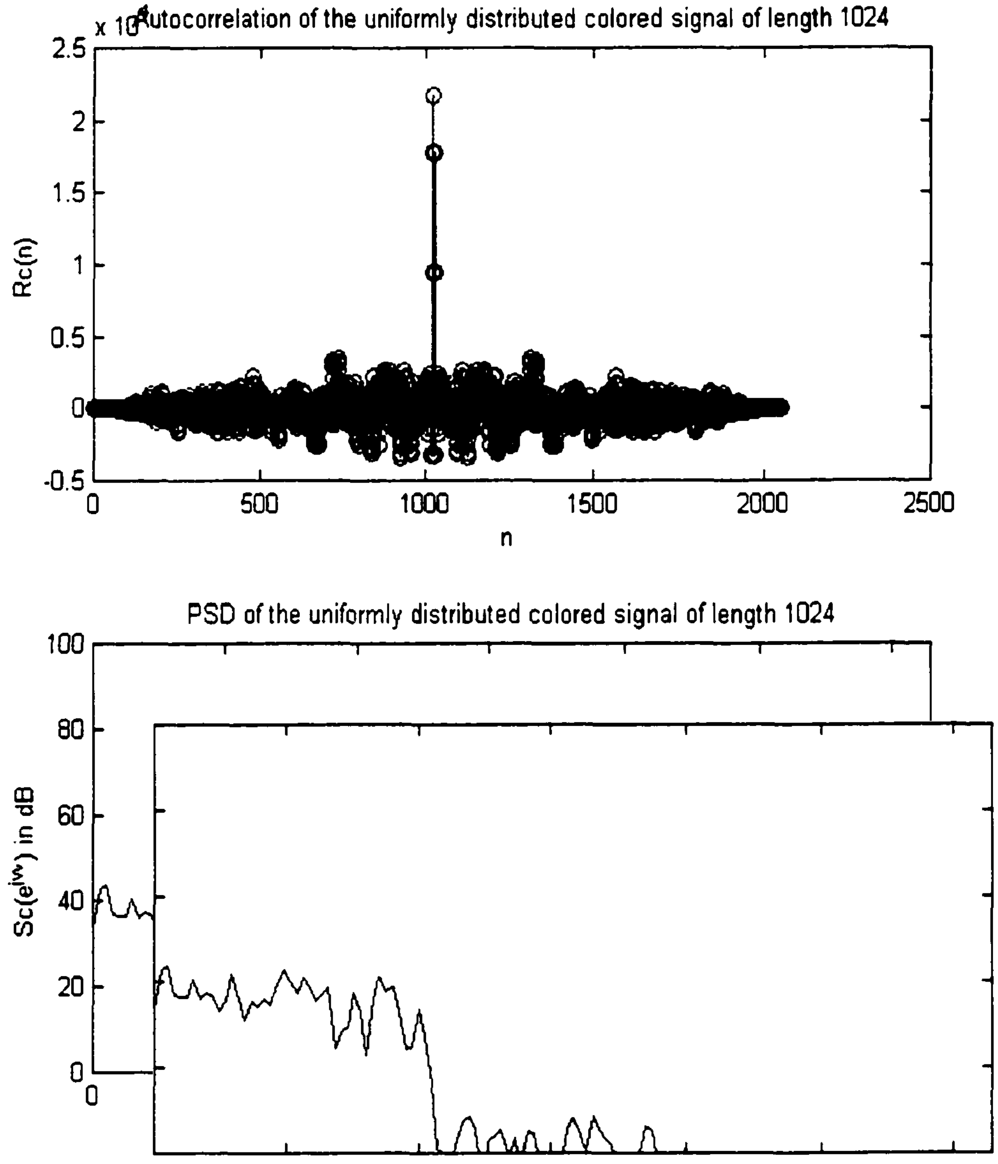


Figure 3-7. Autocorrelation and PSD of the uniformly distributed colored signal of length 1024 and with a cutoff frequency of π/3 rad/sam.

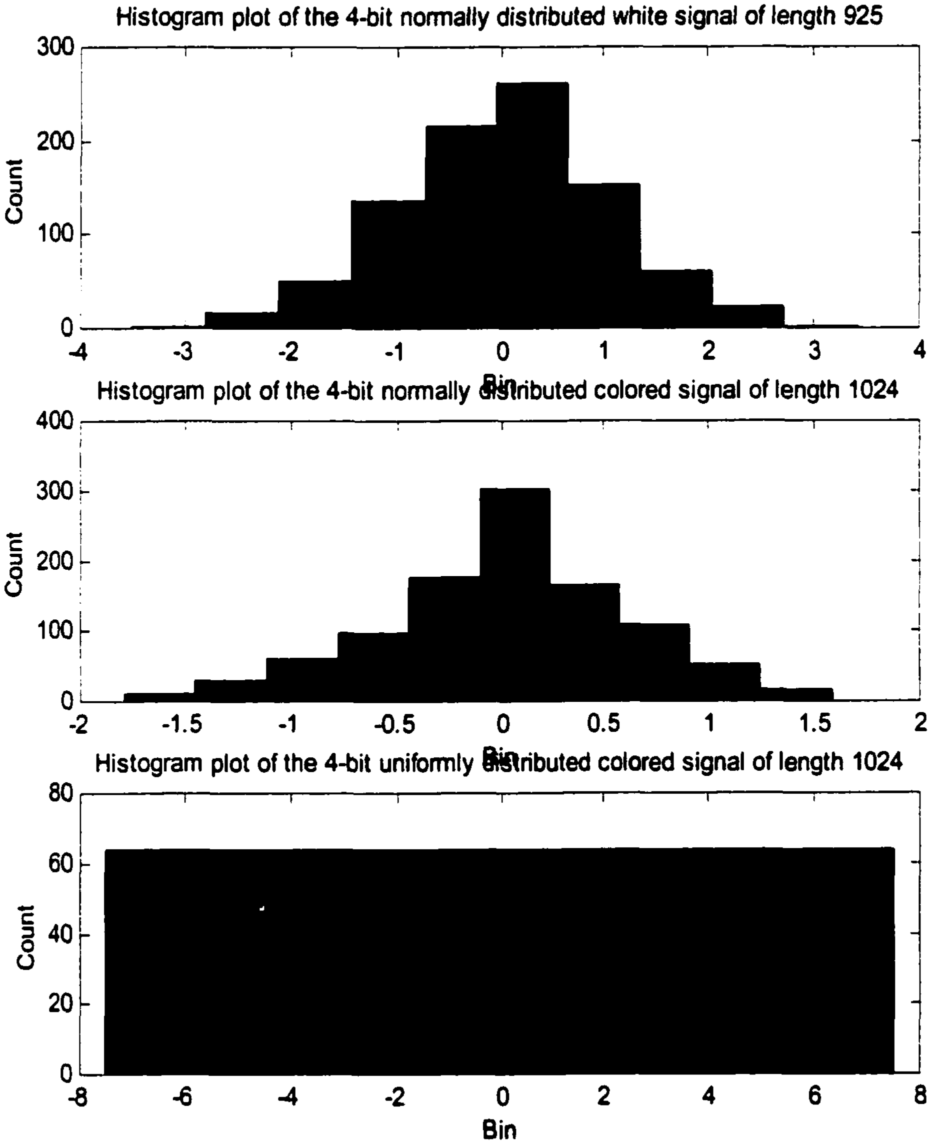


Figure 3-8. Histogram plot of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $\pi/3$ rad/sam.

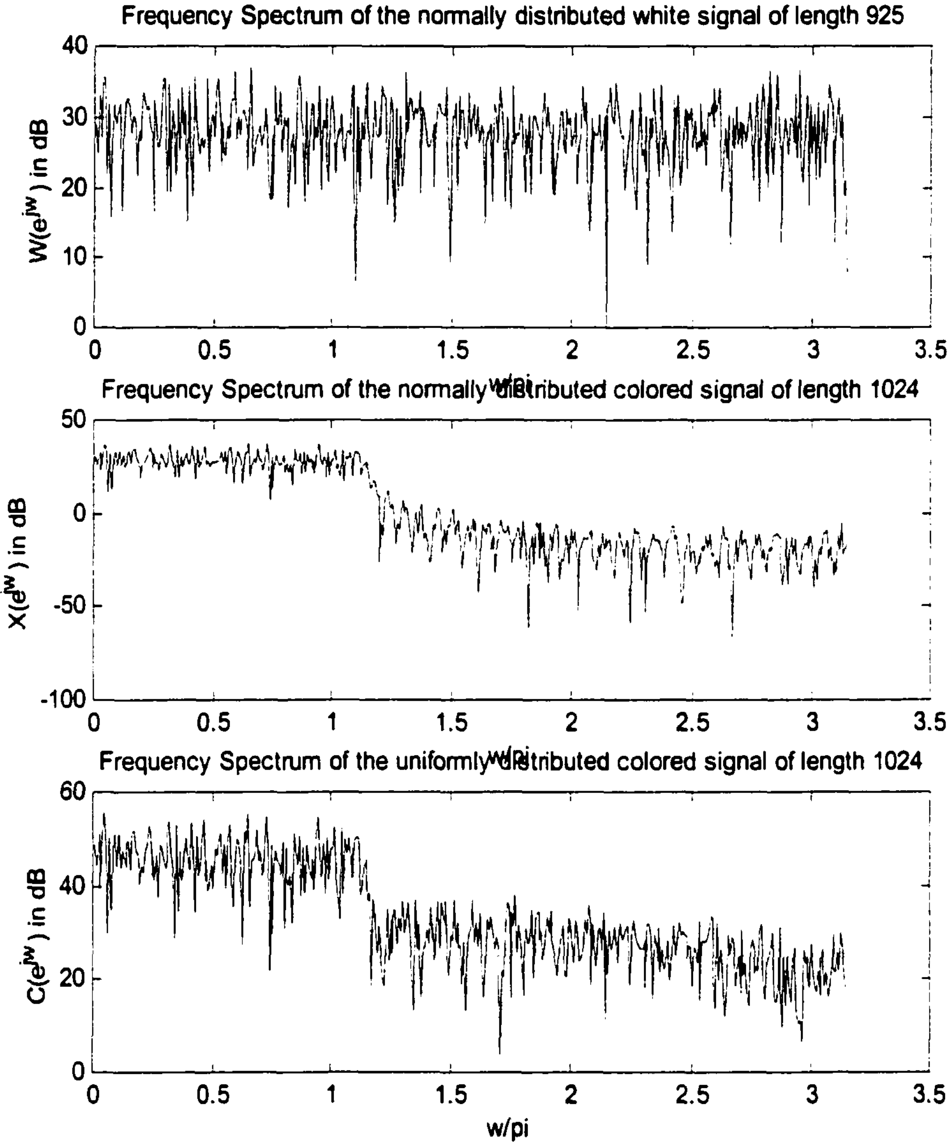
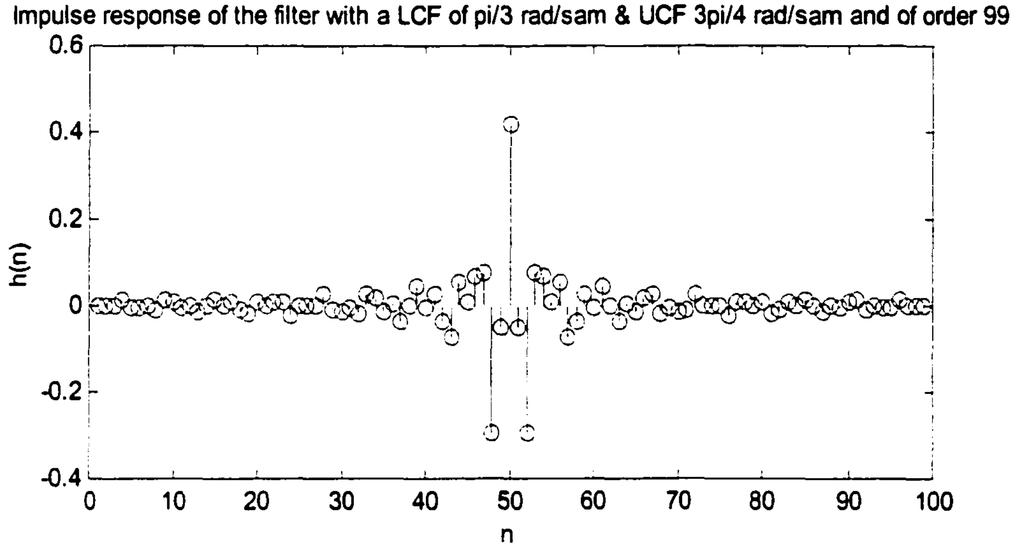


Figure 3-9. Frequency spectrum of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $\pi/3$ rad/sam



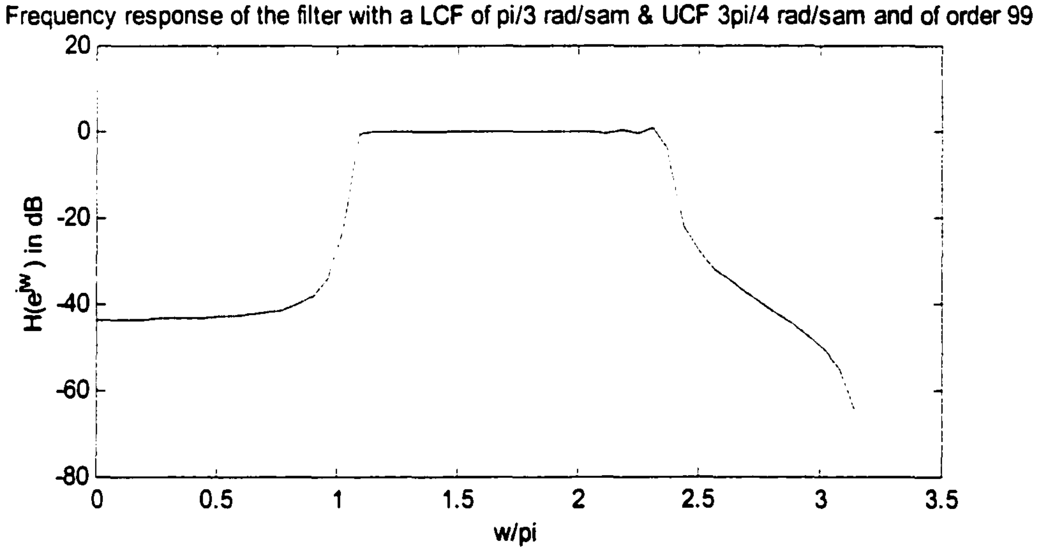


Figure 3-10. Impulse response and frequency response of the filter with a lower cutoff frequency of $\pi/3$ rad/sam & an upper cutoff frequency of $3\pi/4$ rad/sam and of order 99.

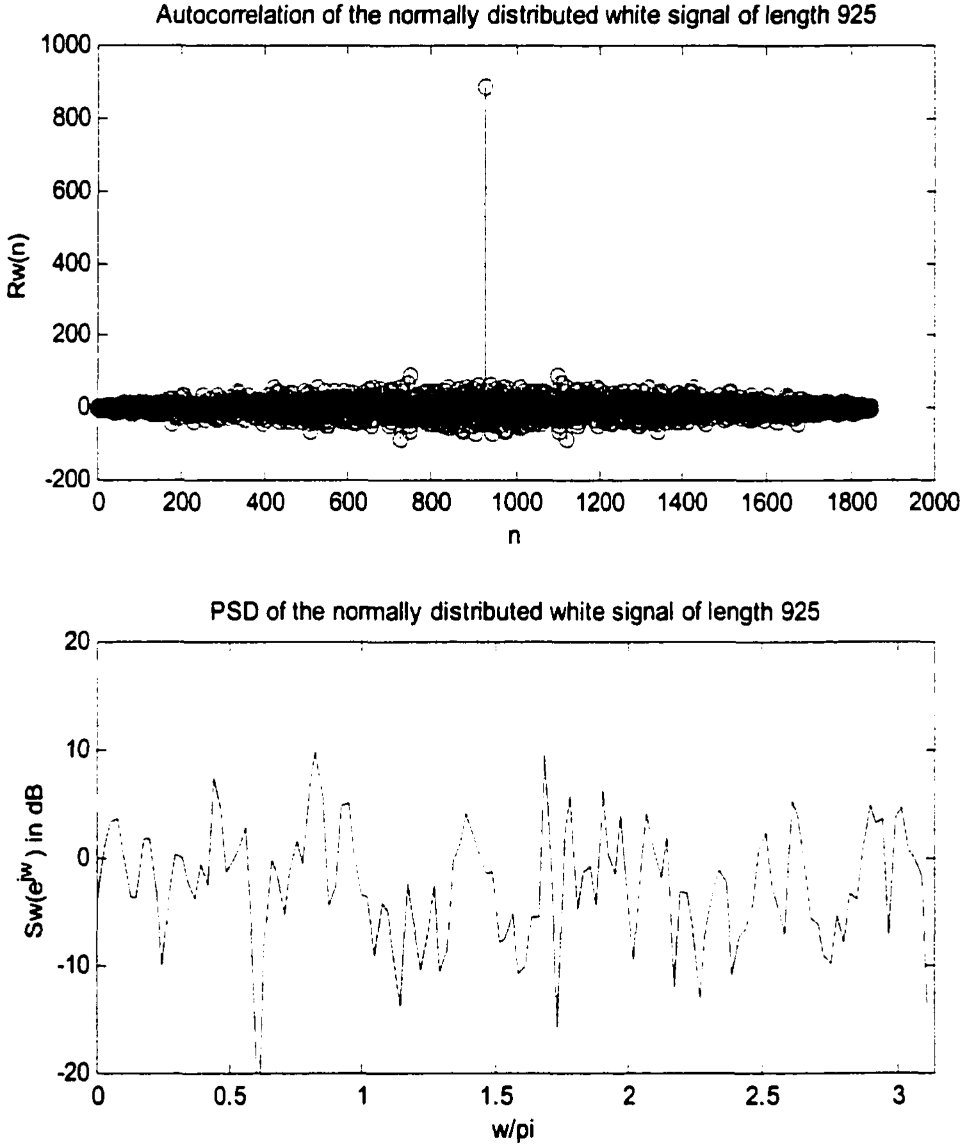


Figure 3-11. Autocorrelation and PSD of the normally distributed white signal of length 925.

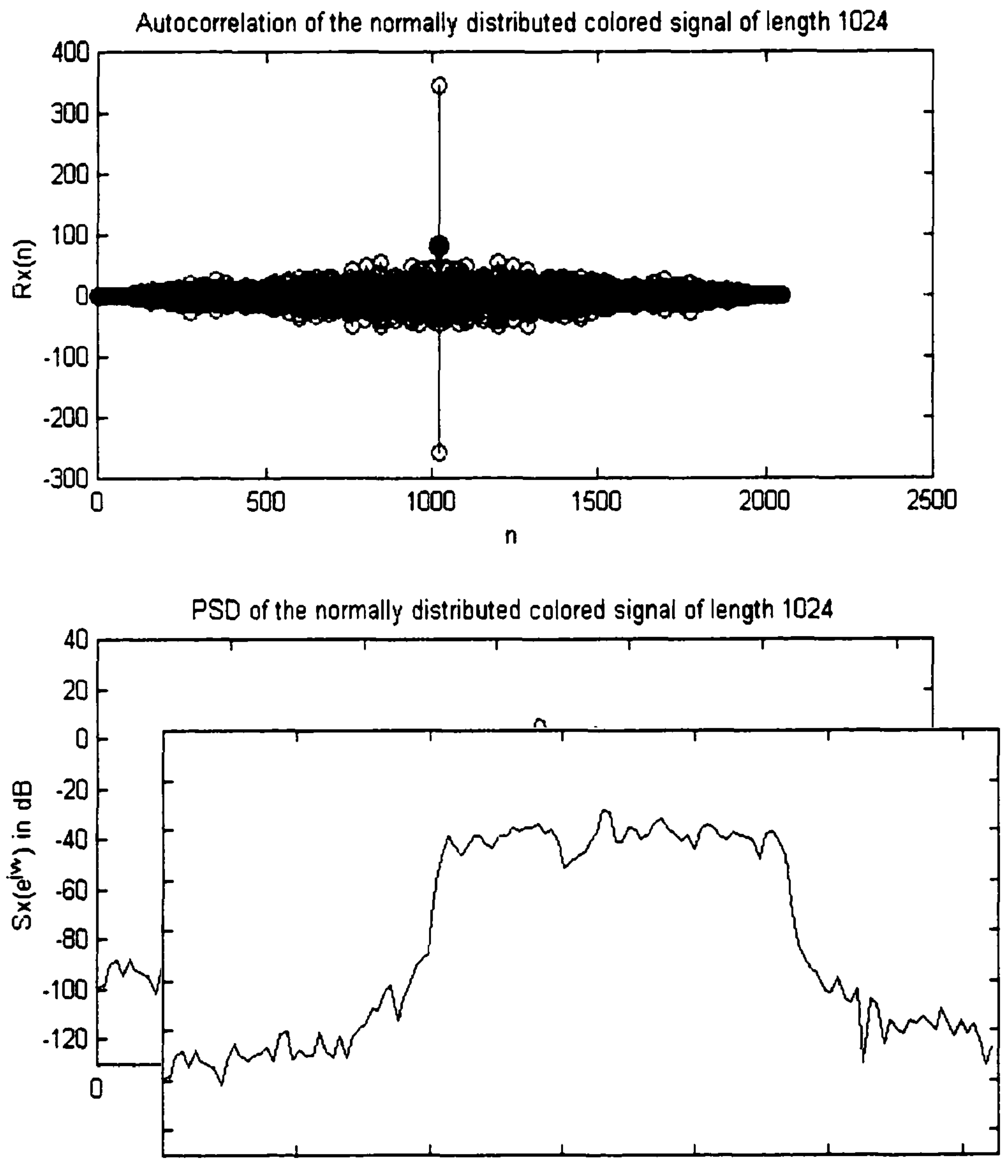


Figure 3-12. Autocorrelation and PSD of the normally distributed colored signal of length 1024 and with a lower cutoff frequency of π/3 rad/sam & an upper cutoff frequency of 3π/4 rad/sam.

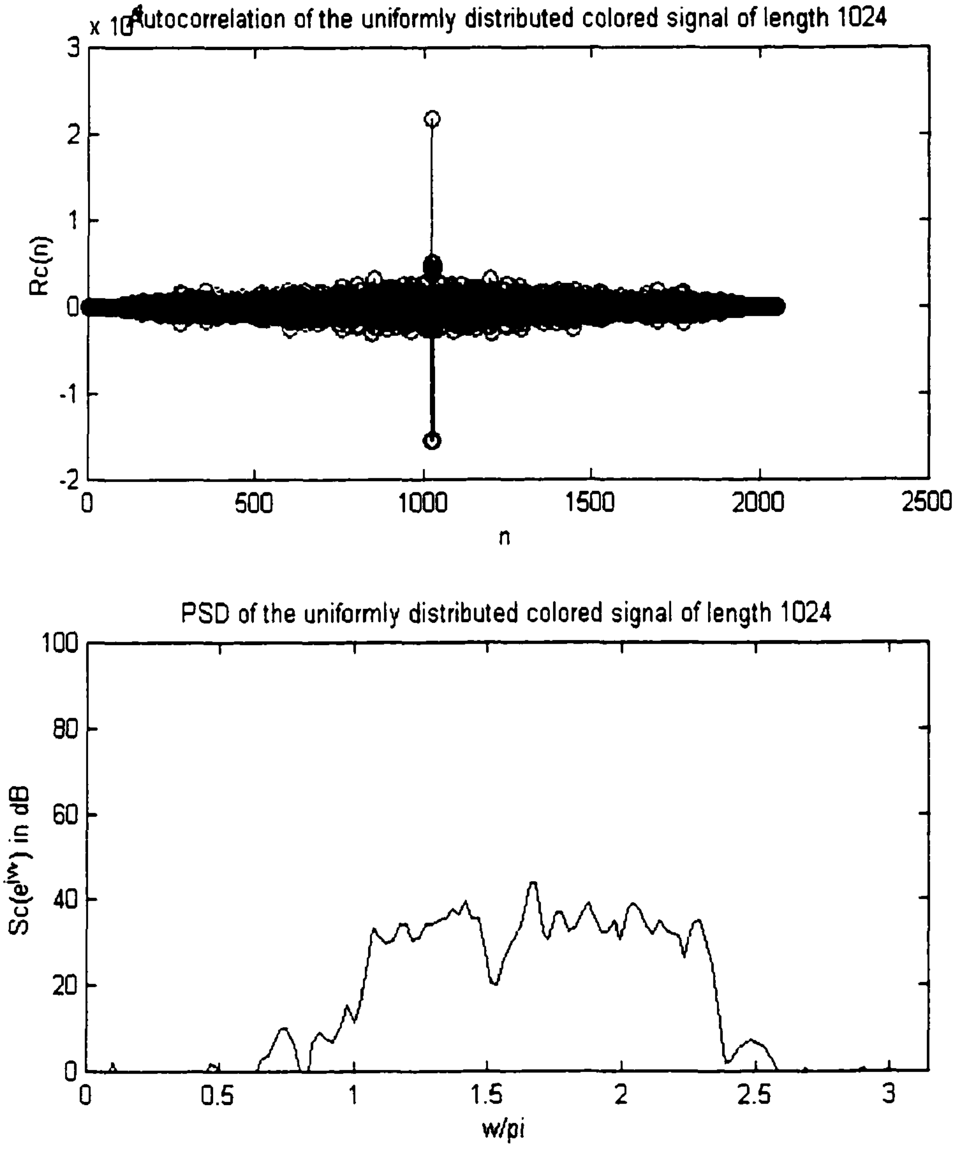


Figure 3-13. Autocorrelation and PSD of the uniformly distributed colored signal of length 1024 and with a lower cutoff frequency of $\pi/3$ rad/sam & an upper cutoff frequency of $3\pi/4$ rad/sam.

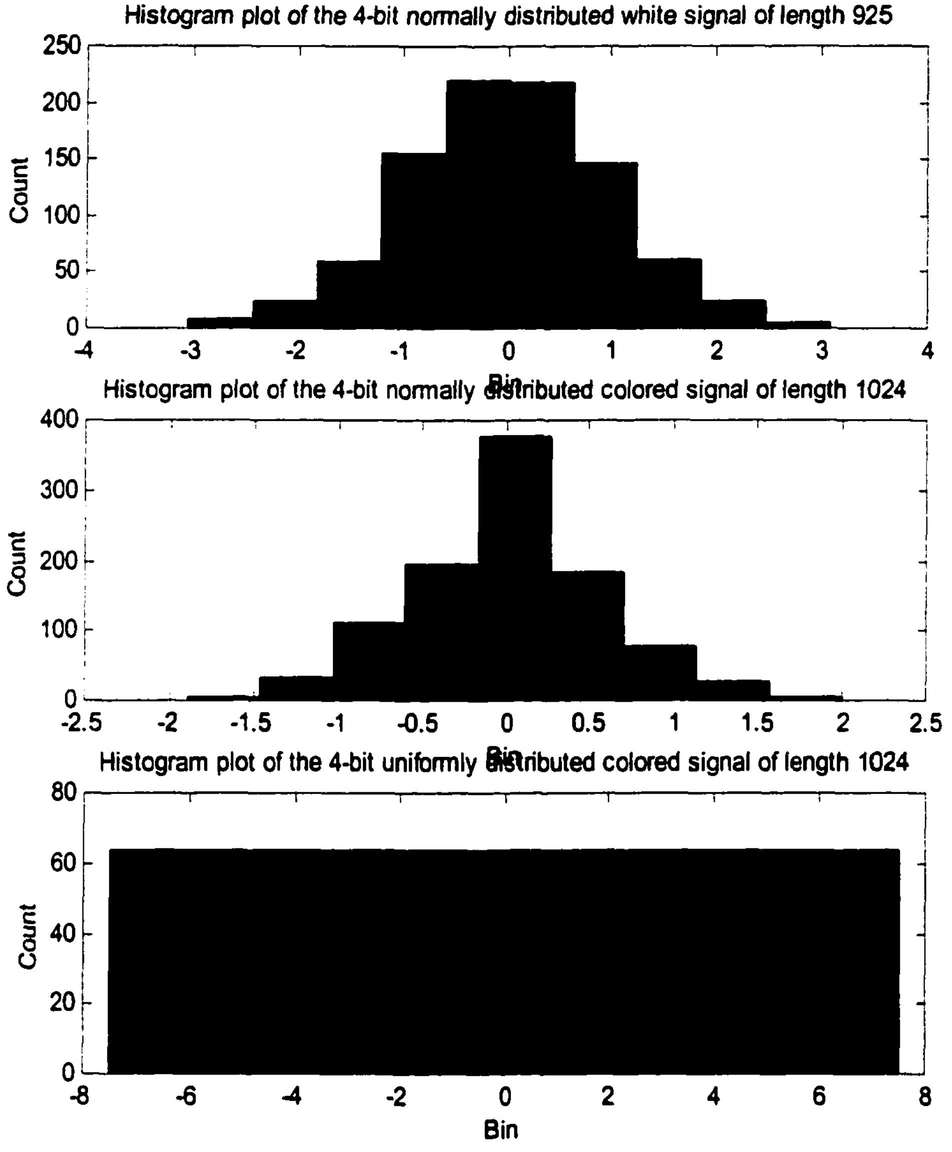


Figure 3-14. Histogram plot of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $\pi/3$ rad/sam & $3\pi/4$ rad/sam.

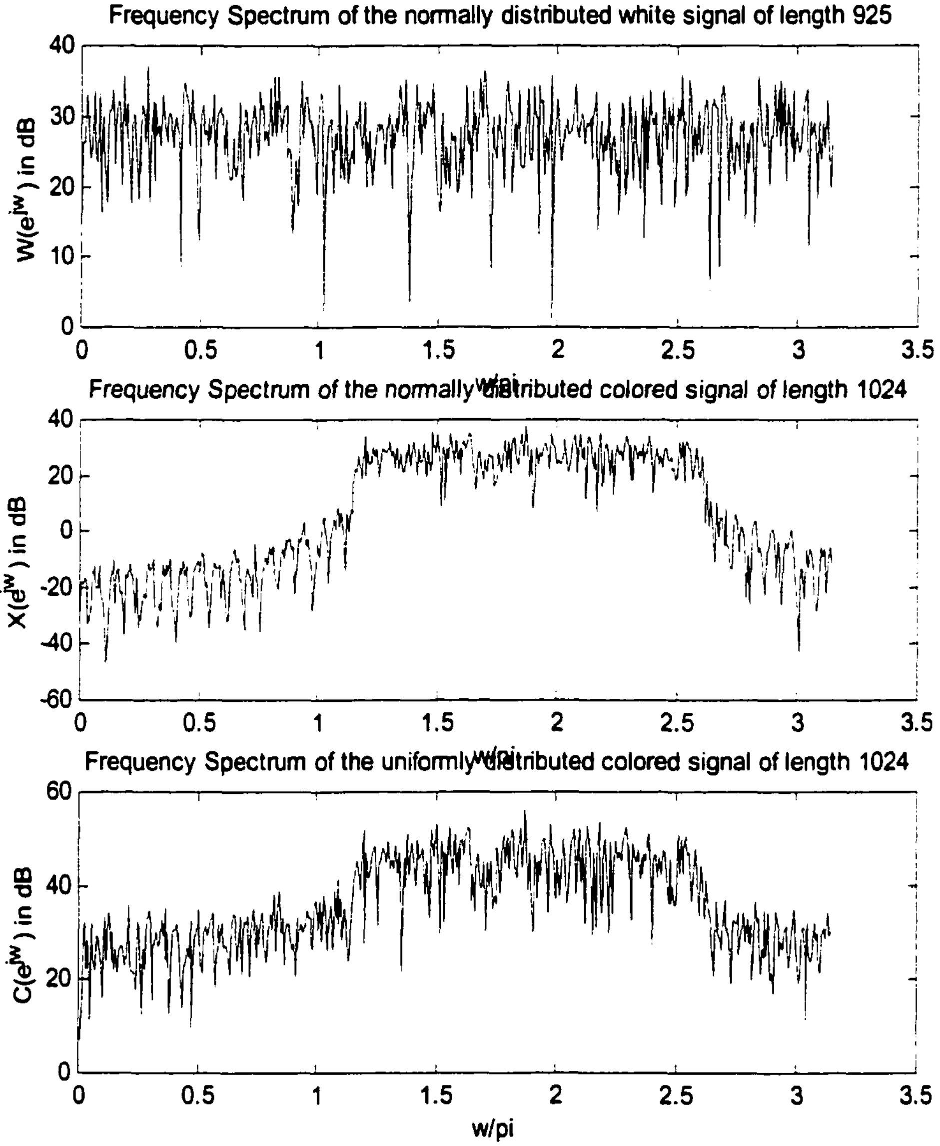
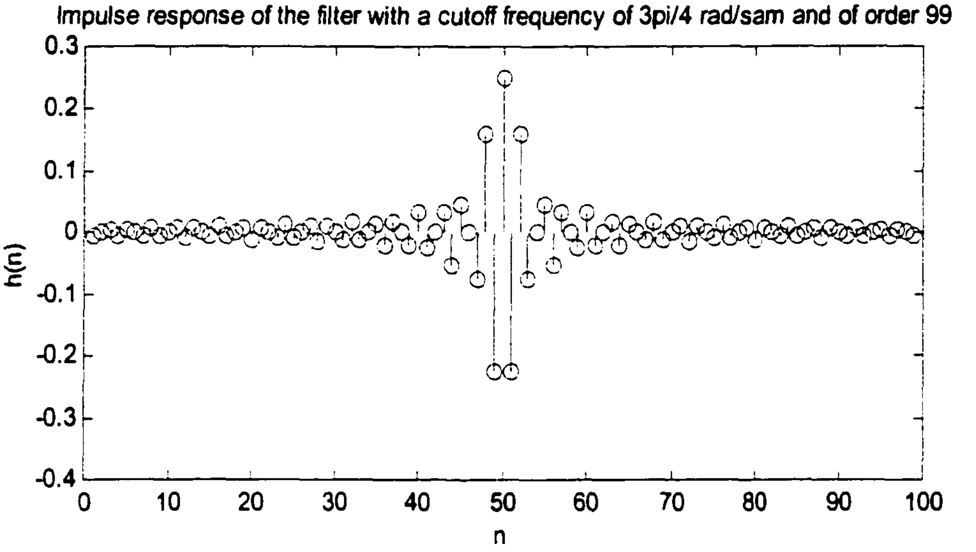


Figure 3-15. Frequency Spectrum of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $\pi/3$ rad/sam and $3\pi/4$ rad/sam.



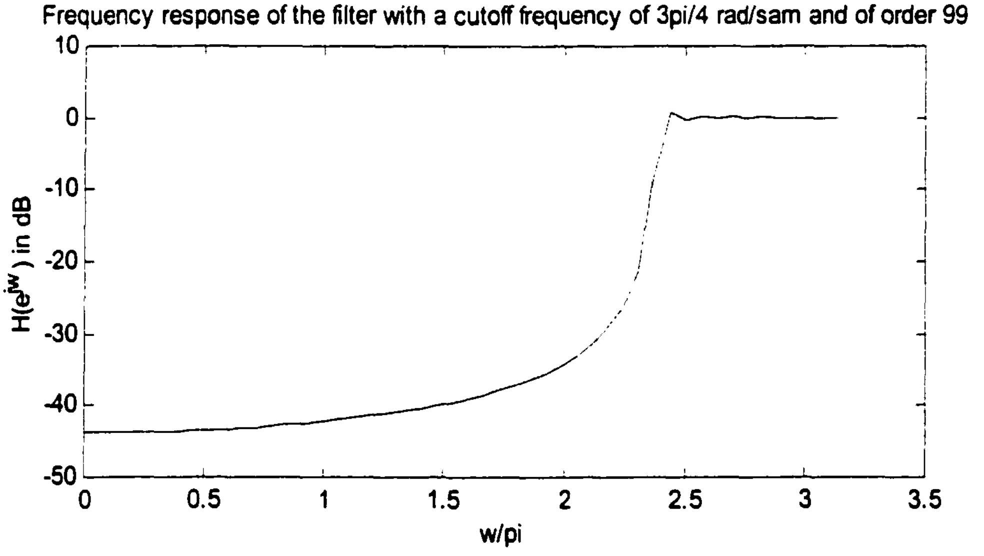


Figure 3-16. Impulse response and frequency response of the filter with a cutoff frequency of $3\pi/4$ rad/sam and of order 99.

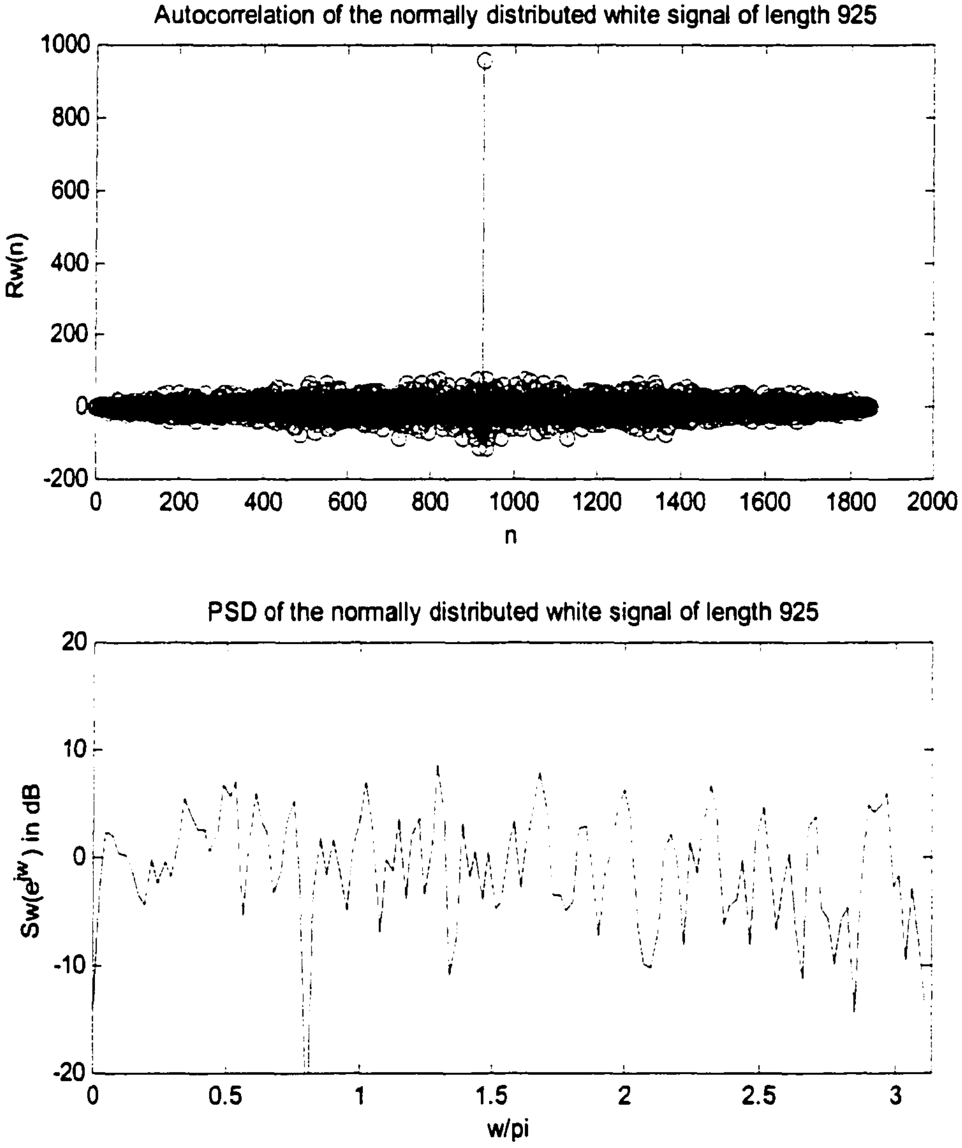


Figure 3-17. Autocorrelation and PSD of the normally distributed white signal of length 925.

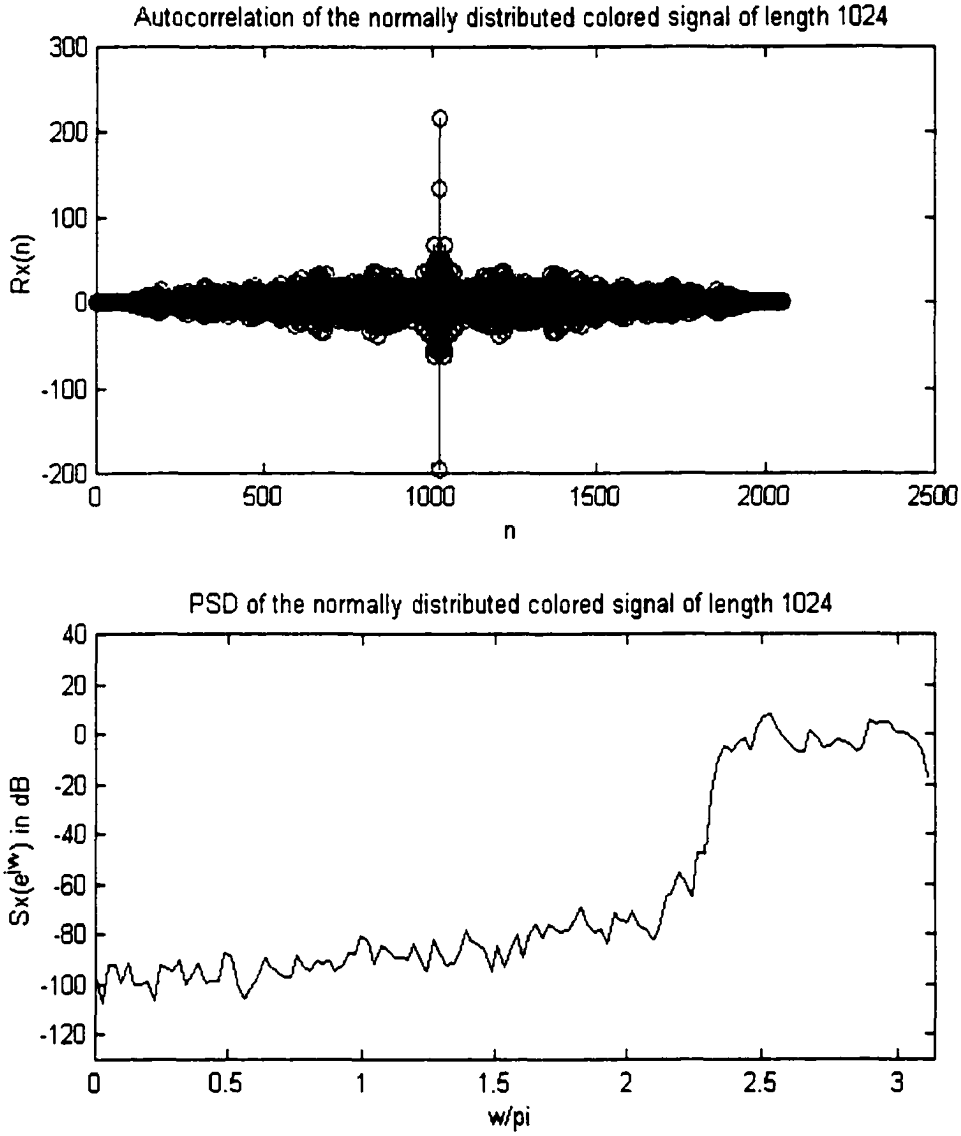


Figure 3-18. Autocorrelation and PSD of the normally distributed colored signal of length 1024 and with a cutoff frequency of $3\pi/4$ rad/sam.

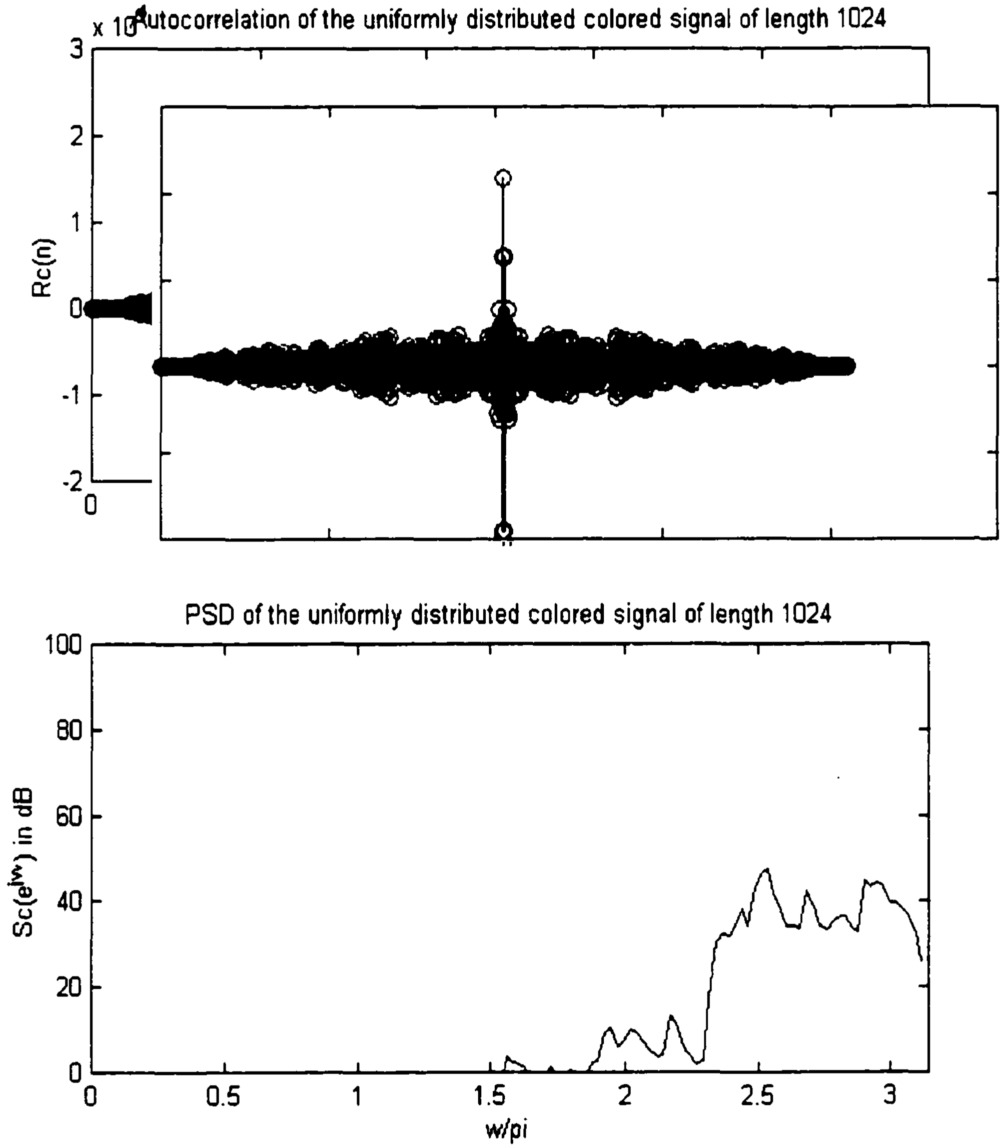


Figure 3-19. Autocorrelation and PSD of the uniformly distributed colored signal of length 1024 and with a cutoff frequency of $3\pi/4$ rad/sam.

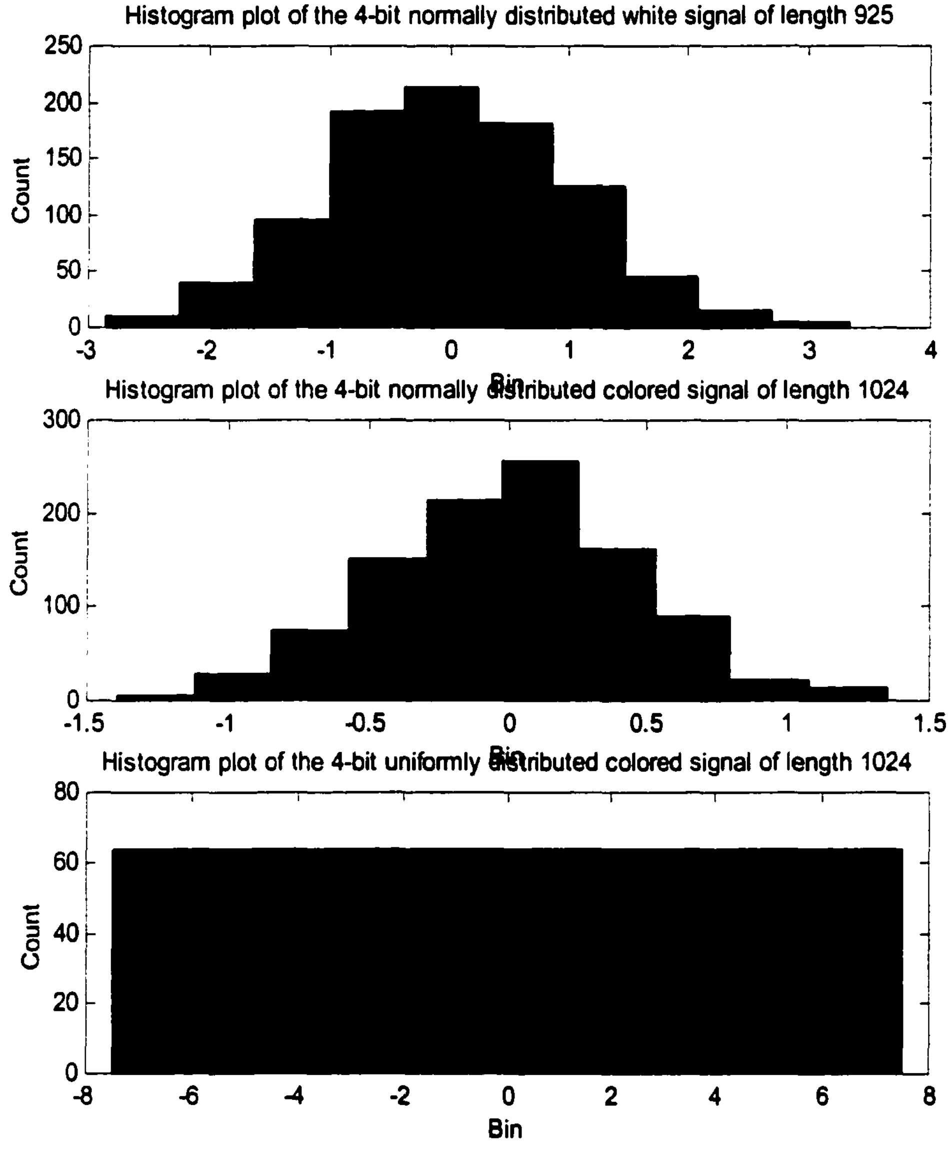


Figure 3-20. Histogram plot of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $3\pi/4$ rad/sam.

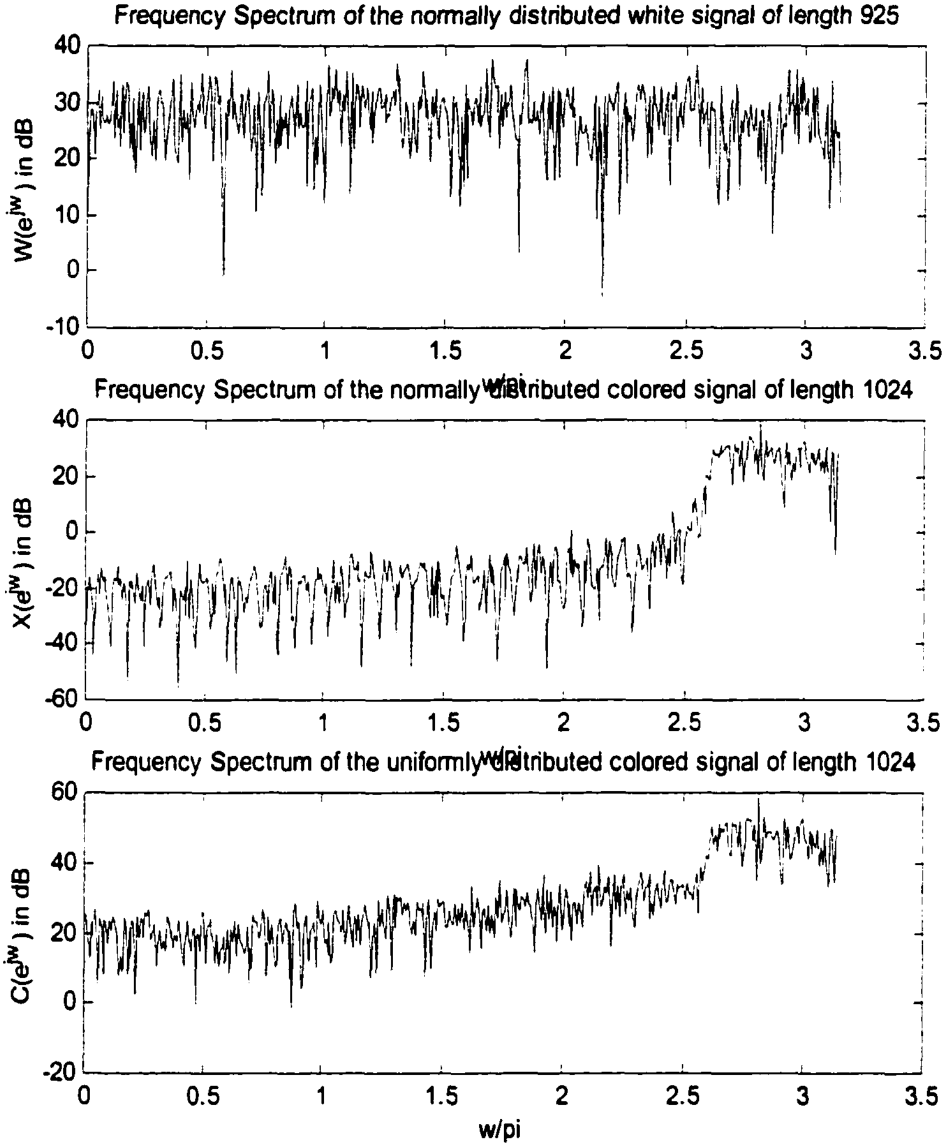
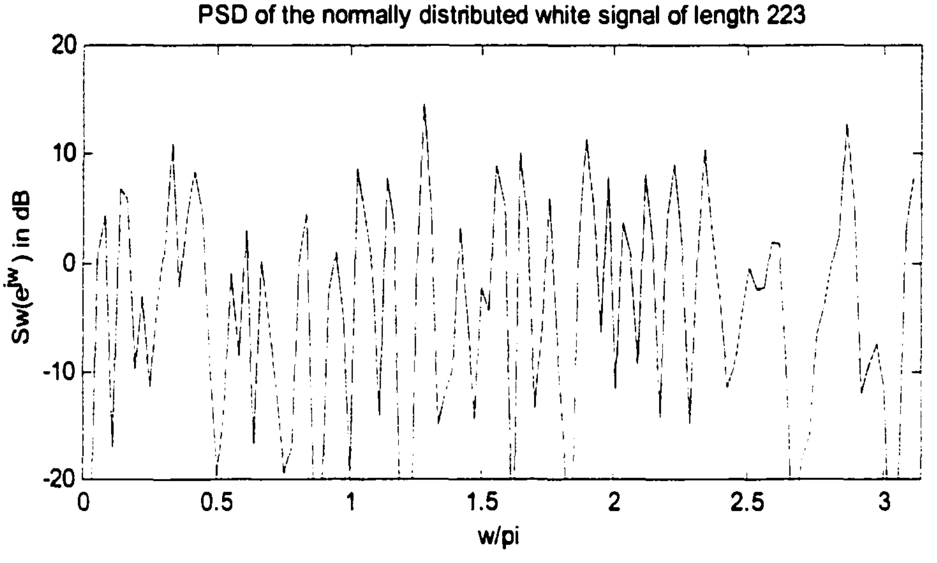


Figure 3-21. Frequency spectrum of the normally distributed white signal, normally distributed colored signal and uniformly distributed colored signal with a cutoff frequency of $3\pi/4$ rad/sam.



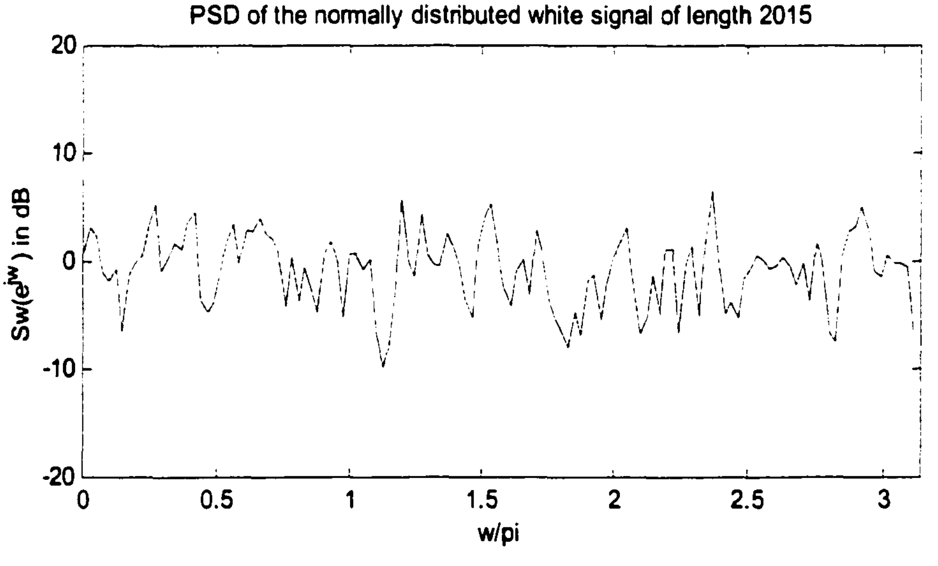
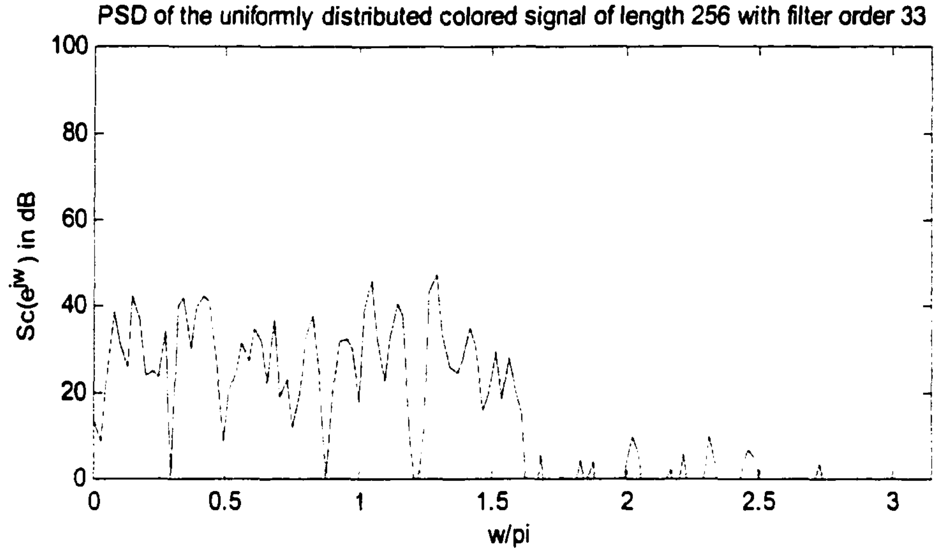


Figure 3-22. PSD of the normally distributed white signal of lengths 223 and 2015.



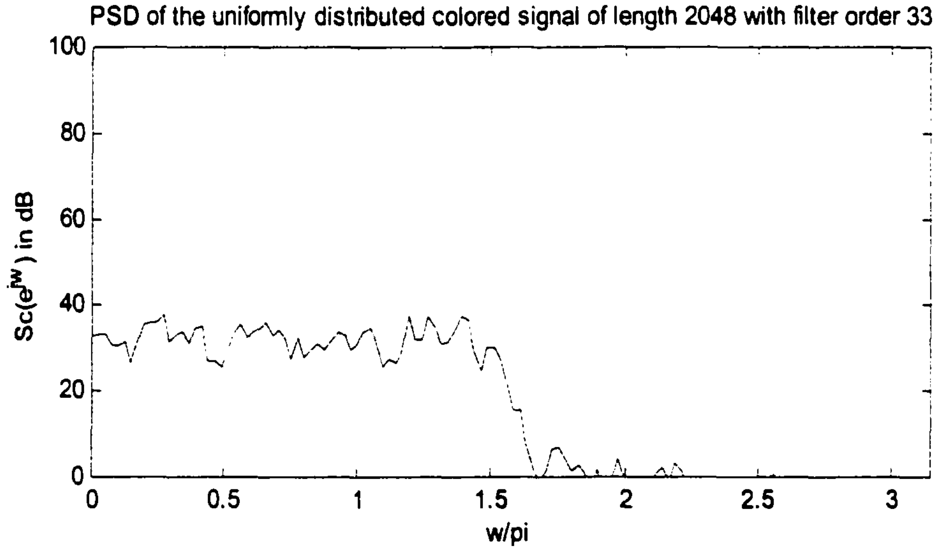


Figure 3-23. PSD of the uniformly distributed colored signal of lengths 223 and 2015 and with a cutoff frequency of $\pi/2$ rad/sam.

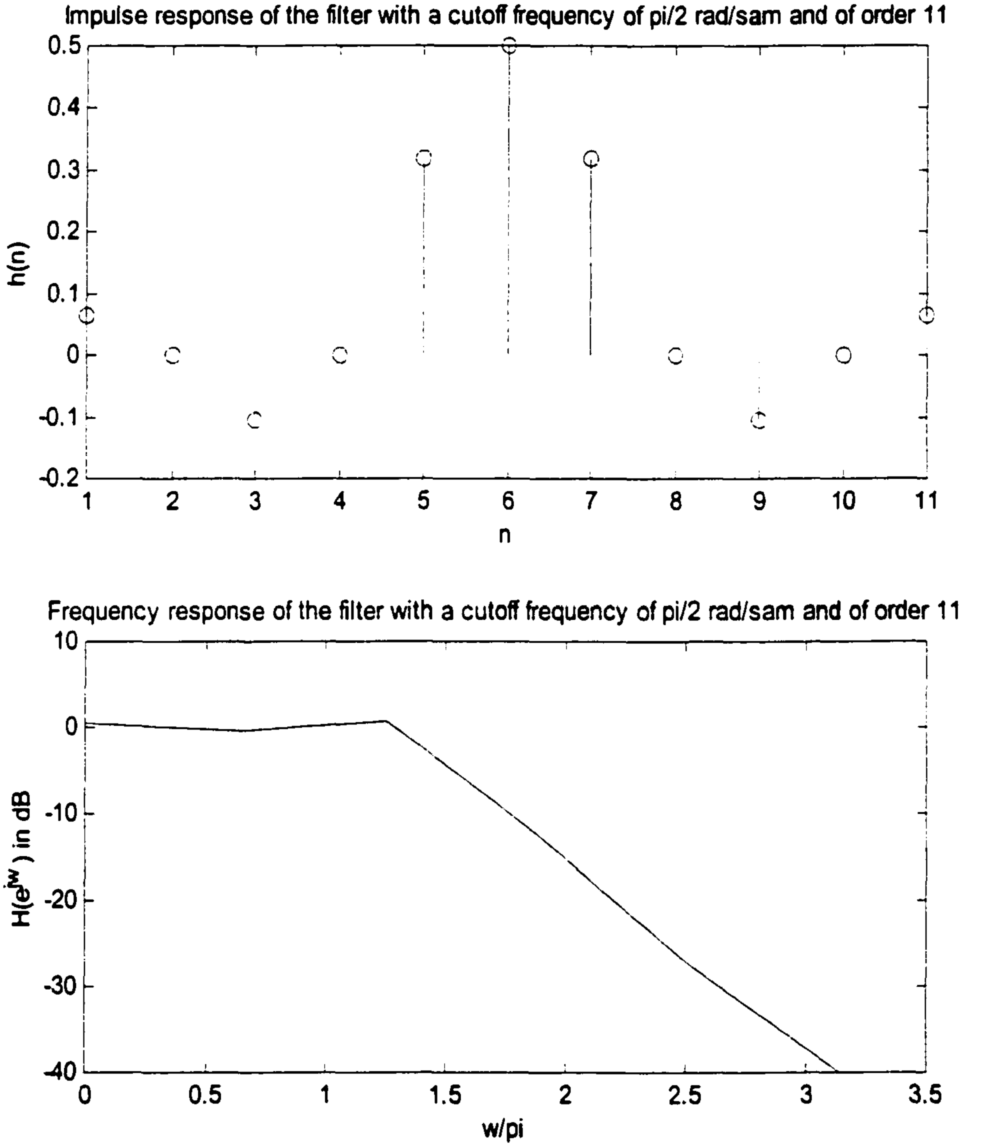


Figure 3-24. Impulse response and frequency response of the filter with a cutoff frequency of $\pi/2$ rad/sam and of order 11.

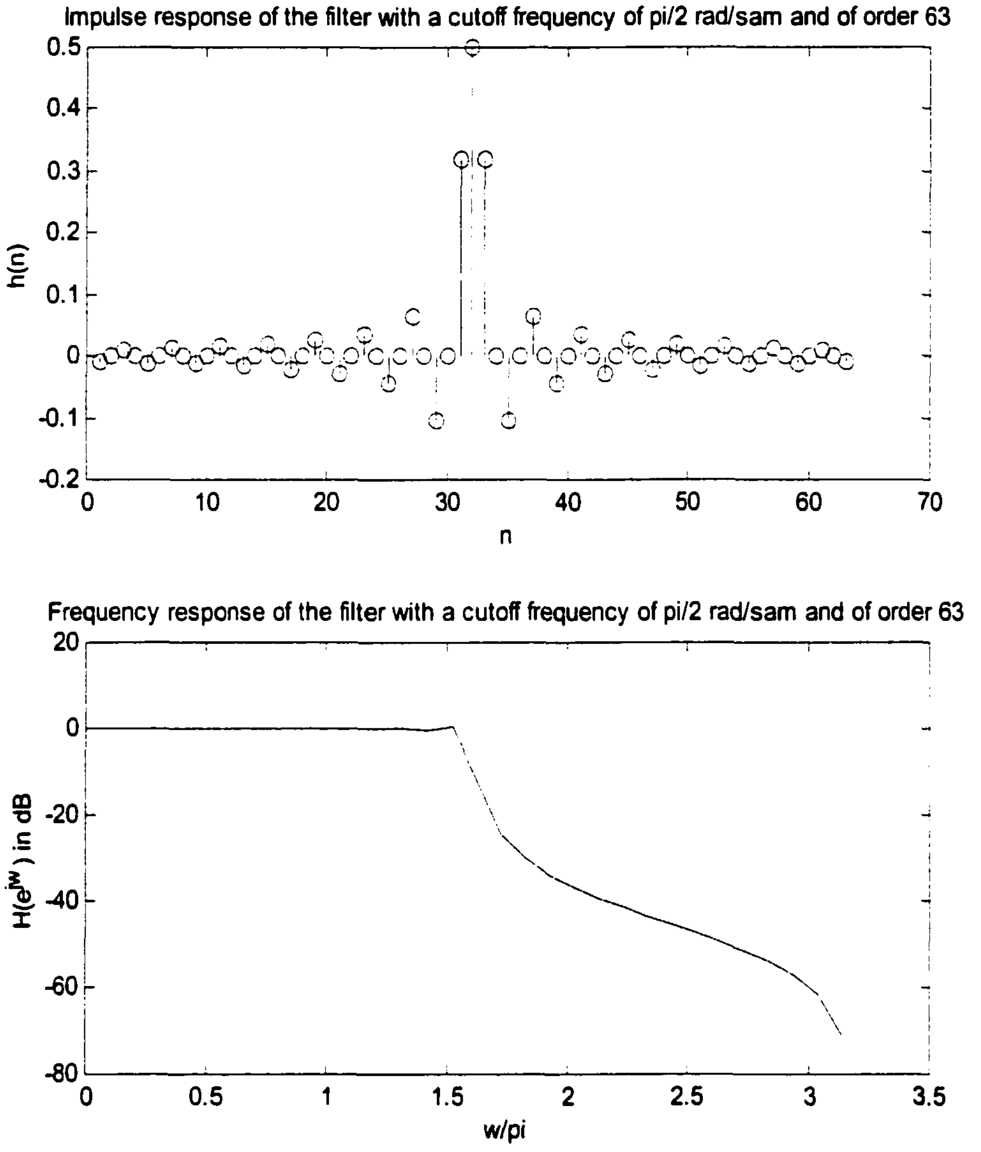


Figure 3-25. Impulse response and frequency response of the filter with a cutoff frequency at π/2 rad/sam and of order 63.

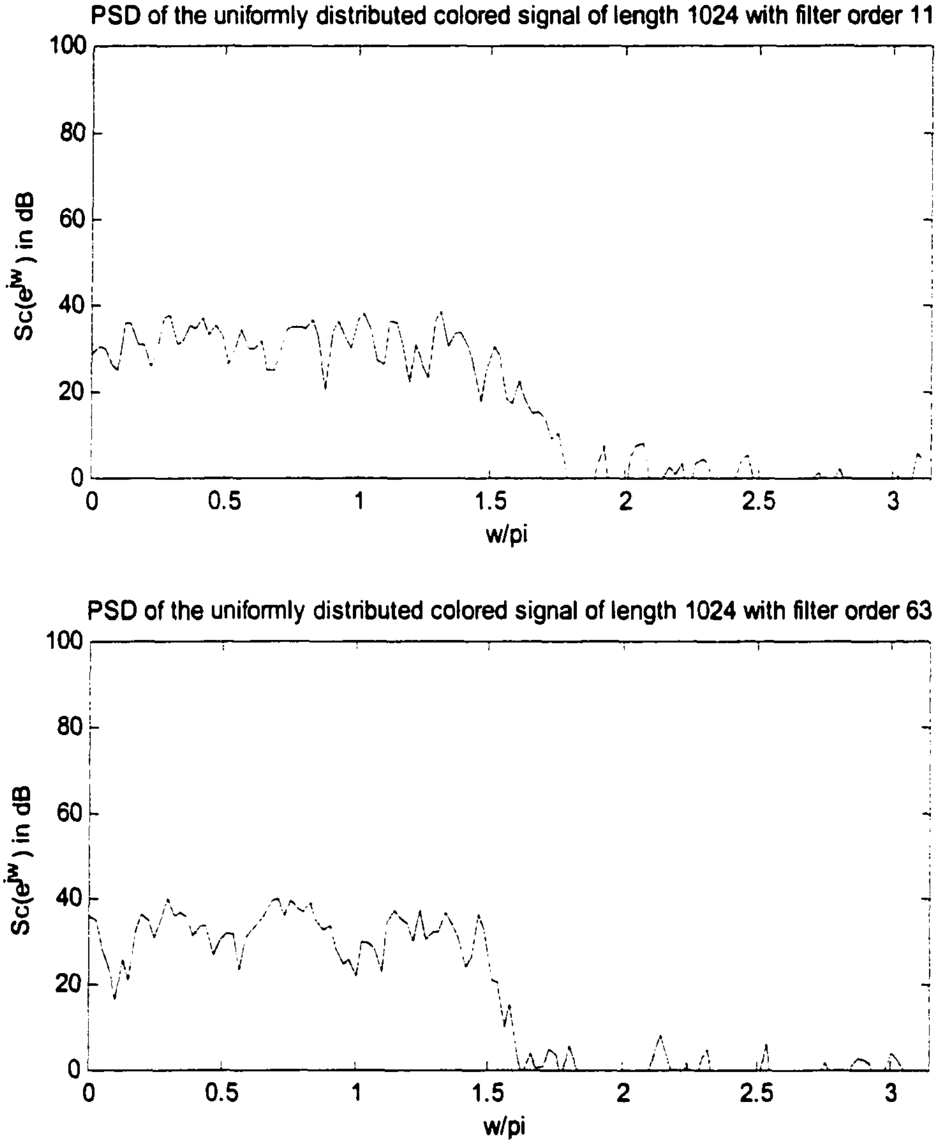


Figure 3-26. PSD of the uniformly distributed colored signal of length 1024 with a cutoff frequency of π/2 rad/sam and of order 11 & 63.

CHAPTER 4

HARDWARE GENERATION

A B bit uniformly distributed white random signal can be efficiently generated using Linear Feedback Shift Registers (LFSRs) [14; 15; 16]; however the literature does not have an efficient method for generating a B bit uniformly distributed colored random signal. One method of generating a B bit uniformly distributed colored random signal is using a finite state machine (FSM). If a FSM has to generate all the B bits of a uniformly distributed colored random signal then the hardware requirements become very large. In this chapter, it is shown that the two most significant bits (MSBs) of a B bit uniformly distributed colored random signal influence the signal's PSD more than the B–2 least significant bits (LSBs). Therefore, when generating a B bit uniformly distributed colored random signal, it is often sufficient to generate the two MSBs as colored bits using a FSM and the remaining B–2 LSBs as white bits using LFSRs. This method reduces the hardware requirements for a real time implementation on chip when compared to the hardware requirements for generating all the B bits as colored bits using a FSM.

4.1 PSD of the individual bits of a uniformly distributed colored random signal

To show that the two MSBs of a B bit uniformly distributed colored random signal influence the signal's PSD more than the B–2 LSBs, the power in the individual bit sequence of a B bit uniformly distributed colored random signal is calculated. If $0 \le w(n) < 1$, then $w(n)$ can be written as

$$w(n) = \sum_{r=1}^{B} w_r(n)\, 2^{-r} \tag{4-1}$$

where $w_r(n)$ is the rth bit of $w(n)$, $w_1(n)$ is the most significant bit (MSB) of $w(n)$ and $w_B(n)$ is the least significant bit (LSB) of $w(n)$. The normally distributed colored signal, $x(n)$, is

$$x(n) = w(n) * h(n)$$

$$= \sum_{k=-\infty}^{\infty} w(k)\, h(n-k) \tag{4-2}$$

where $h(n)$ is the impulse response of a linear phase FIR filter and $w(n)$ is the normally distributed white random signal. Substituting Equation (4-1) in Equation (4-2),

$$x(n) = \sum_{r=1}^{B} w_r(n)\, 2^{-r} * h(n)$$

$$= \sum_{k=-\infty}^{\infty} \left\{ h(n-k) \sum_{r=1}^{B} w_r(k)\, 2^{-r} \right\} \tag{4-3}$$

Interchanging the summations, Equation (4-3) can be written as

$$x(n) = \sum_{r=1}^{B} 2^{-r} \sum_{k=-\infty}^{\infty} h(n-k)\, w_r(k)$$

$$= \sum_{r=1}^{B} 2^{-r}\, x_r(n) \tag{4-4}$$

where $x_r(n) = \sum_{k=-\infty}^{\infty} h(n-k)\, w_r(k)$ and in general, $x_r(n) \notin \{0,1\}$. If $x_r(n)$, $1 \le r \le B$, is converted to a binary number, $Q[x_r(n)]$, by rounding, then $Q[x_r(n)]$ can be written as

$$Q[x_r(n)] = x_r(n) + \sum_{j=1}^{r} e_j(n) \tag{4-5}$$

where $-2^{-j}/2 \le e_j(n) \le 2^{-j}/2$, $e_j(n)$, is the rounding error of the jth bit and each $e_j(n)$ is assumed to be a uniformly distributed, zero mean wide-sense stationary white noise. Figure 4-1 shows the uniform distribution of the rounding error, $e_j(n)$, of the jth bit.

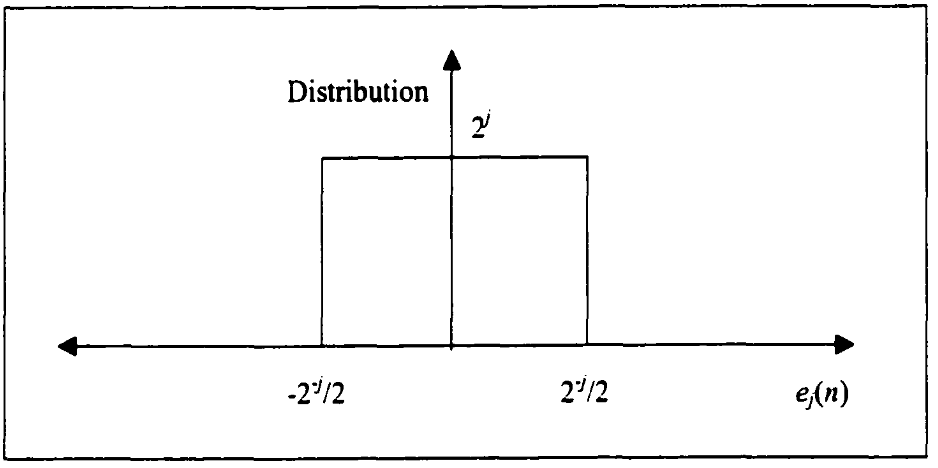


Figure 4-1 Uniform distribution of $e_j(n)$.

Equation (4-5) shows that the rounding error of each bit propagates to the succeeding bits.

For example, when $r = 1$

$$Q[x_1(n)] = x_1(n) + e_1(n) \tag{4-6}$$

where $-2^{-1}/2 \le e_1(n) \le 2^{-1}/2$, and when $r = 2$

$$Q[x_2(n)] = x_2(n) + e_1(n) + e_2(n) \tag{4-7}$$

where $-2^{-2}/2 \le e_2(n) \le 2^{-2}/2$, and when $r = B$

$$Q[x_B(n)] = x_B(n) + \sum_{j=1}^{B} e_j(n) \tag{4-8}$$

where $-2^{-B}/2 \leq e_B(n) \leq 2^{-B}/2$. These equations show that the rounding error of each bit propagates to the succeeding bits. The rounding error of the LSB is $e_1(n) + e_2(n) + \ldots + e_B(n)$ and it is the contribution of rounding error from all the previous bits. If the rounding error of the jth bit, $e_j(n)$, $1 \leq j \leq B$, is assumed to be a zero mean wide-sense stationary white noise then the sum, $e(n) = e_1(n) + e_2(n) + \ldots + e_B(n)$, is also a zero mean wide-sense stationary white noise.

The PSD, $S_{Q[xr]}(e^{j\omega})$, of $Q[x_r(n)]$ can be written as

$$S_{Q[xr]}(e^{j\omega}) = \mathbf{F}\{R_{xr}(n)\} + \sum_{j=1}^{r} \mathbf{F}\{R_{ej}(n)\}$$

$$= S_{xr}(e^{j\omega}) + \sum_{j=1}^{r} S_{ej}(e^{j\omega}) \tag{4-9}$$

where $R_{xr}(n)$ is the autocorrelation of $x_r(n)$, $S_{xr}(e^{j\omega})$ is the PSD of $x_r(n)$, $R_{ej}(n)$ is the autocorrelation of $e_j(n)$ and $S_{ej}(e^{j\omega})$ is the PSD of $e_j(n)$. Because $e_j(n)$ is assumed to be a zero mean wide-sense stationary white noise, the PSD, $S_{ej}(e^{j\omega})$, of $e_j(n)$ is constant over the entire frequency spectrum.

The power, $P_{Q[xr]}$, in $Q[x_r(n)]$ is

$$P_{Q[xr]} = (1/2\pi)\int_{-\pi}^{\pi} S_{xr}(e^{j\omega})\,d\omega + \sum_{j=1}^{r} (1/2\pi)\int_{-\pi}^{\pi} S_{ej}(e^{j\omega})\,d\omega$$

$$= P_{xr} + \sum_{j=1}^{r} P_{ej} \tag{4-10}$$

Because $e_j(n)$ is assumed to be a uniformly distributed, zero mean wide-sense stationary white noise, the power, P_{ej}, in $e_j(n)$ is the variance of $e_j(n)$ [10]; that is,

$$P_{ej} = (1/2\pi) \int_{-\pi}^{\pi} S_{ej}(e^{j\omega})\, d\omega$$

$$= E[e_j^2(n)] \qquad (4\text{-}11)$$

The variance of $e_j(n)$ is

$$E[e_j^2(n)] = \int_{-2^{-j}/2}^{2^{-j}/2} e_j^2(n)\, 2^j\, de(n)$$

$$= 2^{-2j}/12 \qquad (4\text{-}12)$$

From Equations (4-11) and (4-12), the power, P_{ej}, in $e_j(n)$ is

$$P_{ej} = 2^{-2j}/12 \qquad (4\text{-}13)$$

In a B-bit uniformly distributed random signal that has a period of length l, where l = 2^m, each bit sequence is uniformly distributed with 2^{m-1} zeros and 2^{m-1} ones [15]. Therefore, the power, $P_{Q[xr]}$, in the rth bit, $1 \le r \le B$, is

$$P_{Q[xr]} = E\,[\, Q[x_r(n)]^2\,]$$

$$= (1/2^m) \sum_{n=1}^{2^{m-1}} 0 \; + (1/2^m) \sum_{n=1}^{2^{m-1}} (2^{-r})^2$$

$$= (2^{-2r}\, 2^{m-1})/2^m$$

$$= 2^{-2r}/2 \qquad (4\text{-}14)$$

Substituting Equations (4-13) and (4-14) in Equation (4-10),

$$2^{-2r}/2 = P_{xr} + \sum_{j=1}^{r} 2^{-2j}/12 \qquad (4\text{-}15)$$

Equation (4-15) shows that the power in the rounding error of each bit is added to the succeeding bits. Therefore, the power, P_{e1}, in the rounding error of the MSB is the smallest and it is $2^{-2}/12$. The power in the rounding error of the LSB is the largest and it is the sum of the power in the rounding error of all the bits. The signal power, $P_{Q[x1]}$, in the MSB is the largest and it is $2^{-2}/2$. The signal power, $P_{Q[xB]}$, in the LSB is the smallest and it is $2^{-2B}/2$. Therefore, the power in the rounding error of the MSB is much smaller than the power in $Q[x_1(n)]$ and $S_{Q[x1]}(e^{j\omega})$ is similar to $S_{x1}(e^{j\omega})$. The power in the rounding error of the LSB is the contribution of the power in the rounding error of all the previous bits and it is greater than the power in $Q[x_B(n)]$. Therefore, $S_{Q[xB]}(e^{j\omega})$ is constant over the entire frequency spectrum and $Q[x_B(n)]$ looks like a white bit. For example, when $r = 1$

$$P_{Q[x1]} = P_{x1} + P_{e1} \qquad (4\text{-}16)$$

$$P_{Q[x1]} = 2^{-2}/2 = 0.125$$

$$P_{e1} = 2^{-2}/12 = 0.0208$$

where $P_{Q[x1]}$ is the power in the MSB, $Q[x_1(n)]$. Because the power, P_{e1}, in the rounding error of the MSB is smaller than the signal power, the MSB looks like a colored bit. When $r = 2$

$$P_{Q[x2]} = P_{x2} + P_{e1} + P_{e2} \qquad (4\text{-}17)$$

$$P_{Q[x1]} = 2^{-4}/2 = 0.03125$$

$$P_{e1} + P_{e2} = 2^{-2}/12 + 2^{-4}/12 = 0.026$$

where $P_{Q[x2]}$ is the power in the second MSB, $Q[x_2(n)]$. The power in the rounding error of the MSB propagates to the second MSB and the power in the rounding error of the second MSB is $P_{e1} + P_{e2}$. Because $P_{e1} + P_{e2} < P_{x2}$, the second LSB also looks like a colored bit. When $r = 3$

$$P_{Q[x3]} = P_{x3} + P_{e1} + P_{e2} + P_{e3} \qquad (4\text{-}18)$$

$$P_{Q[x3]} = 2^{-6}/2 = 0.0078$$

$$P_{e1} + P_{e2} + P_{e3} = 2^{-2}/12 + 2^{-4}/12 + 2^{-6}/12$$
$$= 0.0273$$

where $P_{Q[x3]}$ is the power in the third MSB, $Q[x_3(n)]$. The power in the rounding error of the third MSB is $P_{e1} + P_{e2} + P_{e3}$ and it is the contribution of the power in the rounding error from all the previous bits. Because $P_{e1} + P_{e2} + P_{e3} >> P_{x3}$, the third MSB looks like a white bit and $S_{Q[x3]}(e^{j\omega})$ is constant over the entire frequency spectrum. For $r \geq 3$, the power in the rth bit is smaller than the power in the rounding error of that bit. Therefore, the PSD of the rth bit is constant over the entire frequency spectrum and the rth bit appears like a white bit. Figure 4-2 shows the signal power in the rth bit and the power in the rounding error of that bit (in this figure, the continuous line corresponds to the signal power and the dashed line corresponds to the power in the rounding error). This figure shows that signal power in the two MSBs is larger than the power in the rounding error of the corresponding bit. For $r \geq 3$, the signal power is smaller than the power in the rounding error of the rth bit. Therefore, for $r \geq 3$, the rth bit looks like a white bit.

Equations (4-4) – (4-18) are also applicable to the uniformly distributed colored random signal, $c(n)$, because Section 3.1 showed that the nonlinear transformation shapes the distribution of the random signal and does not significantly alter the spectrum. Therefore, if the uniformly distributed colored signal, $c(n)$, is converted to a B bit binary signal $c_r(n)$, then only the two MSBs are colored and influence the signal's PSD more than the remaining B-2 LSBs. Because the power in the rounding error of the B-2 LSBs is larger than the power in the signal, the B-2 LSBs look like white bits. Therefore, a B bit uniformly distributed colored random signal can be adequately generated by generating the two MSBs as colored bits and the remaining B–2 LSBs as white bits.

To illustrate, a uniformly distributed highpass filtered random signal that is bandlimited between $3\pi/4$ and π and has a length of 1024 is generated. The colored random signal is converted to a four-bit binary signal. Figure 4-3 shows that the PSD of the MSB is colored because the power in the MSB is larger than the power in the rounding error of the MSB. Similarly, Figure 4-4 shows that the PSD of the second bit is also colored because the power in the second MSB exceeds the power in the rounding error of the second MSB. Figure 4-5 and Figure 4-6 show that the PSD of the third bit

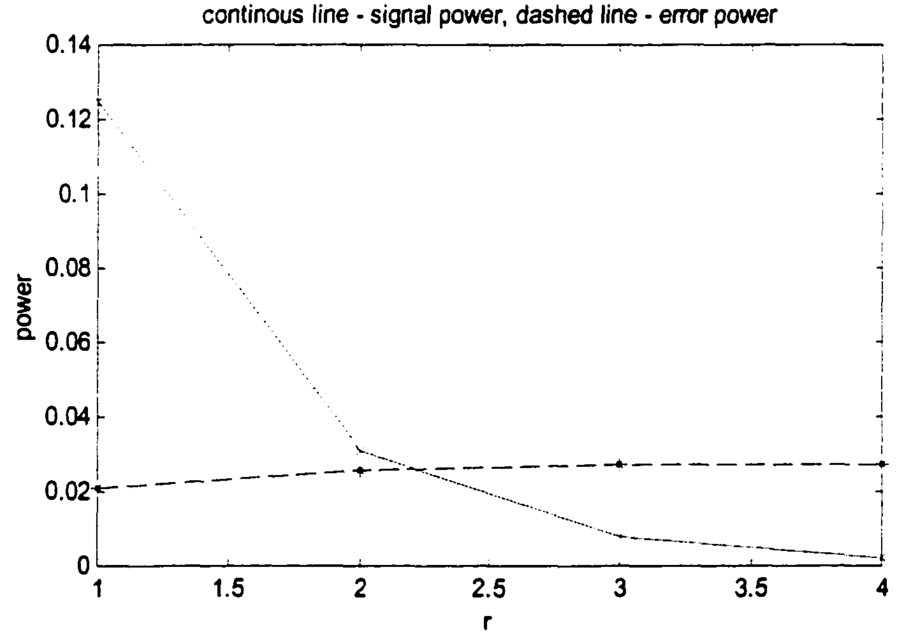


Figure 4-2 The signal power in the rth bit and the power in the rounding error of the rth bit.

and the fourth bit are uniform over the entire frequency spectrum because the power in these bits is smaller than the power in the rounding error of the corresponding bits. Therefore, a four- bit uniformly distributed colored random signal can be adequately generated by generating the two MSBs as colored bits and the remaining 2 LSBs as white bits.

4.2 Hardware design using a FSM

Section 4.1 showed that a uniformly distributed colored random signal can be adequately generated by generating the two MSBs as colored bits and the remaining B-2 LSBs as white bits. The B-2 LSBs can be generated on chip using Linear Feed Back Shift Registers (LFSRs) and the two MSBs can be generated using a finite state machine (FSM).

In this thesis, FSMs that generate the two MSBs are designed in VHDL [20; 21]. The uniformly distributed colored random signal constructed by the algorithm is assumed to be periodic with a period l, where l is the length of the constructed random signal. Therefore, the FSM is designed with l states and the output at the mth state is the mth two-bit word of the random signal, where $1 \leq m \leq l$. For example, consider a four-bit uniformly distributed highpass filtered random signal that is bandlimited between $3\pi/4$ and π and has a length of 256. The two MSBs of the uniformly distributed highpass filtered random signal are colored and are generated using a FSM. Because the length of the random signal is 256, the FSM will have 256 states and the output at the mth state will be the mth two-bit word of the random signal, where $1 \leq m \leq 256$. Figure 4-7 shows a FSM with 256 states, designed to generate the two MSBs of the four-bit uniformly distributed colored random signal. This figure shows that at the rising edge of the clock, the FSM changes from the mth state to $(m+1)$th state and the output at the mth state is the mth two-bit word of the random signal. A C program was written to generate the state machine VHDL code. Appendix A contains the C program listing that generates the state machine VHDL code.

4.3 Linear Feedback Shift Register implementation

Section 4.2 shows the generation of the two MSBs of a uniformly distributed colored random signal using a FSM. The remaining B-2 LSBs can be generated on chip using Linear Feed Back Shift Registers (LFSRs). The B-2 LSBs can also be generated using a FSM, but the hardware requirements for an LFSR implementation are much less than the hardware requirements for a FSM implementation. Therefore the LFSR implementation is preferred for generating the B-2 white bits.

The approach for the LFSR implementation is to use a polynomial associated with the LFSR for generating each white bit of the uniformly distributed colored random signal. For B-2 white bits of the random signal, B-2 polynomials are required. Each polynomial should generate a sequence that has a length that is identical to the length of the random signal constructed by the algorithm. For example, in Section 4.2, the two MSBs of the uniformly distributed highpass filtered random signal that is bandlimited between $3\pi/4$ and π and has a length of 256 are generated using a FSM. The remaining 2 LSBs are generated using LFSRs. Two polynomials, one for the third bit and the other for the fourth bit, are required for the LFSR implementation. Each polynomial should generate a sequence that has a length of 256. Because an n-stage LFSR can generate a sequence of length 2^n at most, a sequence that has a length of 256 requires an eight-stage shift register for the LFSR implementation. The irreducible polynomials associated with an eight-stage LFSR are [14]:

1. $1+x^2+x^3+x^4+x^8$
2. $1+x^3+x^5+x^6+x^8$
3. $1+x^2+x^5+x^6+x^8$
4. $1+x+x^3+x^5+x^8$
5. $1+x+x^5+x^6+x^8$
6. $1+x+x^6+x^7+x^8$
7. $1+x+x^2+x^5+x^6+x^7+x^8$
8. $1+x+x^2+x^3+x^4+x^6+x^8$

Because the above polynomials are irreducible polynomials, they can generate sequences with a period of length 2^8-1, which is 255. Any two of the above polynomials can be used to generate the two LSBs of the random signal. The PSD of the sequence generated using the LFSR will be white [14; 15; 16] and uniformly distributed with $2^{8-1}-1$ zeros and 2^{8-1} ones. Figure 4-8 shows the PSD and the histogram plot of the LSB sequence of the uniformly distributed highpass filtered random signal that is generated using an eight-stage LFSR. This figure shows that the PSD of the binary sequence generated using an LFSR is white and uniformly distributed with $2^{8-1}-1$ zeros and 2^{8-1} ones. Figure 4-9 shows that the four-bit random signal generated using the FSM and LFSRs is a uniformly distributed highpass filtered random signal that is bandlimited between $3\pi/4$ and π. Therefore, a uniformly distributed colored random signal can be adequately generated by generating the two MSBs as colored bits and the B-2 LSBs as white bits.

In general, the period of the random signal can be increased if each bit of the random signal is generated with a different period. The period of the random signal will be very long if the MSB is generated with a long period and the succeeding bits are generated with decreasing periods. For example, in a four-bit uniformly distributed colored random signal, the period of the random signal increases if the MSB is generated with a period of 2^n-1, the second MSB is generated with a period of 2^{n-1}-1, the third MSB is generated with a period of 2^{n-2}-1 and the LSB is generated with a period of 2^{n-3}-1. This method can be followed to increase the period of a uniformly distributed colored random signal generated using a FSM and LFSRs. But the disadvantage is that the hardware requirements for a FSM implementation grow exponentially with the period of the random sequence. Therefore, if the two MSBs are generated with a longer period than the succeeding bits, then the hardware requirements for a FSM implementation increase such that the real time implementation on chip becomes too large for a typical DEM DAC. To keep the hardware requirements as small as possible, the two MSBs are generated with a small period. The *B*-2 LSB bits that are generated using LFSRs can have longer periods because increasing their periods increase the hardware requirements by few registers and gates. The trade-off for this method of generating the random signal is that the period of the random signal is just greater than the period of the MSB. But, this method is still advantageous because the side correlations are decreased without much increase in the hardware requirements for a real time implementation on chip.

For example, the two MSBs of a four-bit uniformly distributed highpass filtered random signal are generated using a FSM with a period of 256. The third MSB is generated using an LFSR with a period of 511 and the LSB is also generated using an LFSR but with a period of 1023. Figure 4-10 and Figure 4-11 show the autocovariance and the PSD of the 2 MSB sequences generated using a FSM. Because the period of the sequence is 256, the autocovariance of the sequence has four side correlations and a main correlation. Figure 4-12 shows the autocovariance and the PSD of the third MSB sequence generated using an LFSR. Because the period of the third MSB sequence is 511, the autocovariance of the sequence has 2 side correlations and a main correlation. Figure 4-13 shows the autocovariance and the PSD of the LSB sequence generated using an LFSR. Because the period of the LSB sequence is 1023, the autocovariance of the sequence has no side correlation. Figure 4-14 shows the PSD and the histogram plot of the uniformly distributed colored random signal of length 1023 generated using a FSM and different length LFSRs. This figure shows that the random signal is a uniformly distributed high pass filtered signal that is bandlimited between $3\pi/4$ and π. Figure 4-15 shows the autocovariance of the uniformly distributed colored random signal of length 1023 generated using a FSM and different length LFSRs. Figure 4-16 shows the autocovariance of the uniformly distributed colored random signal of length 1023 generated with a period of 256. Figures 4-13 and 4-14 show that the side correlations of the uniformly distributed colored random signal of length 1023 generated using a FSM and different length LFSRs are lesser than the side correlations of the uniformly distributed colored random signal of length 1023 generated with a period of 256.

Therefore, generating a uniformly distributed colored random signal by generating the two MSBs with a small period and the succeeding B-2 LSBs with increasing periods reduce the side correlations of the random signal without much increase in the hardware requirements for a real time implementation on chip for a DEM DAC application.

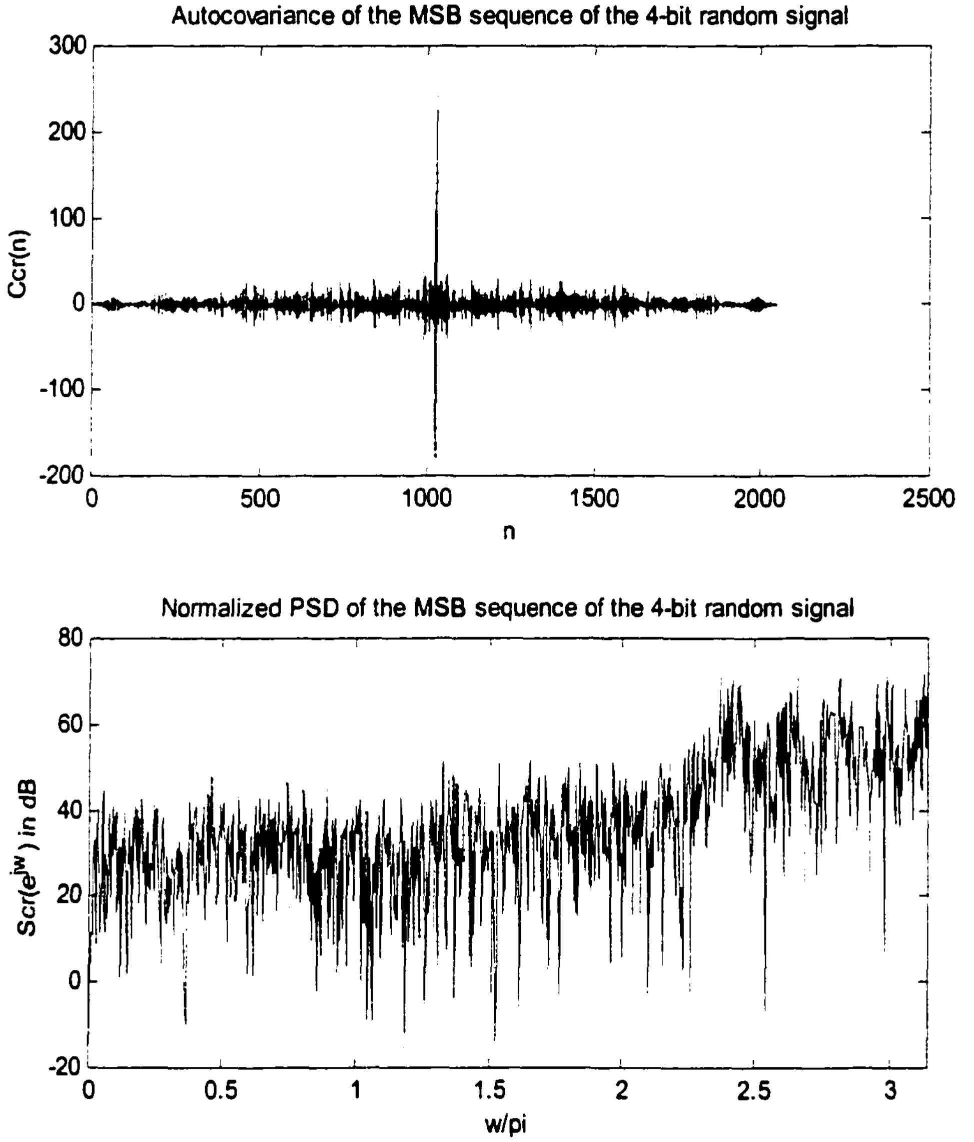


Figure 4-3. Autocovariance and PSD of the MSB sequence of the four-bit random signal that is bandlimited between $3\pi/4$ rad/sam and π rad/sam.

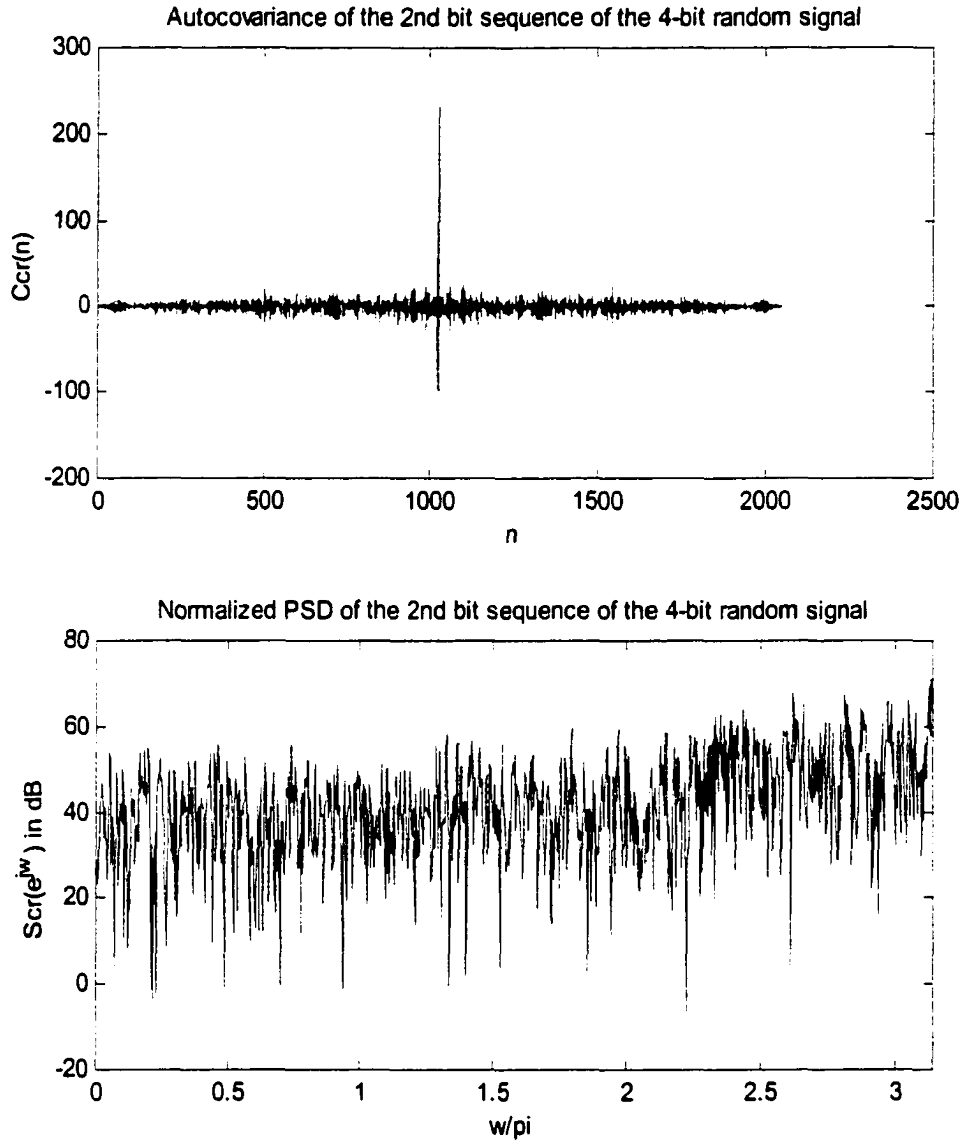


Figure 4-4. Autocovariance and PSD of the 2nd bit sequence of the four-bit random signal that is bandlimited between $3\pi/4$ rad/sam and π rad/sam.

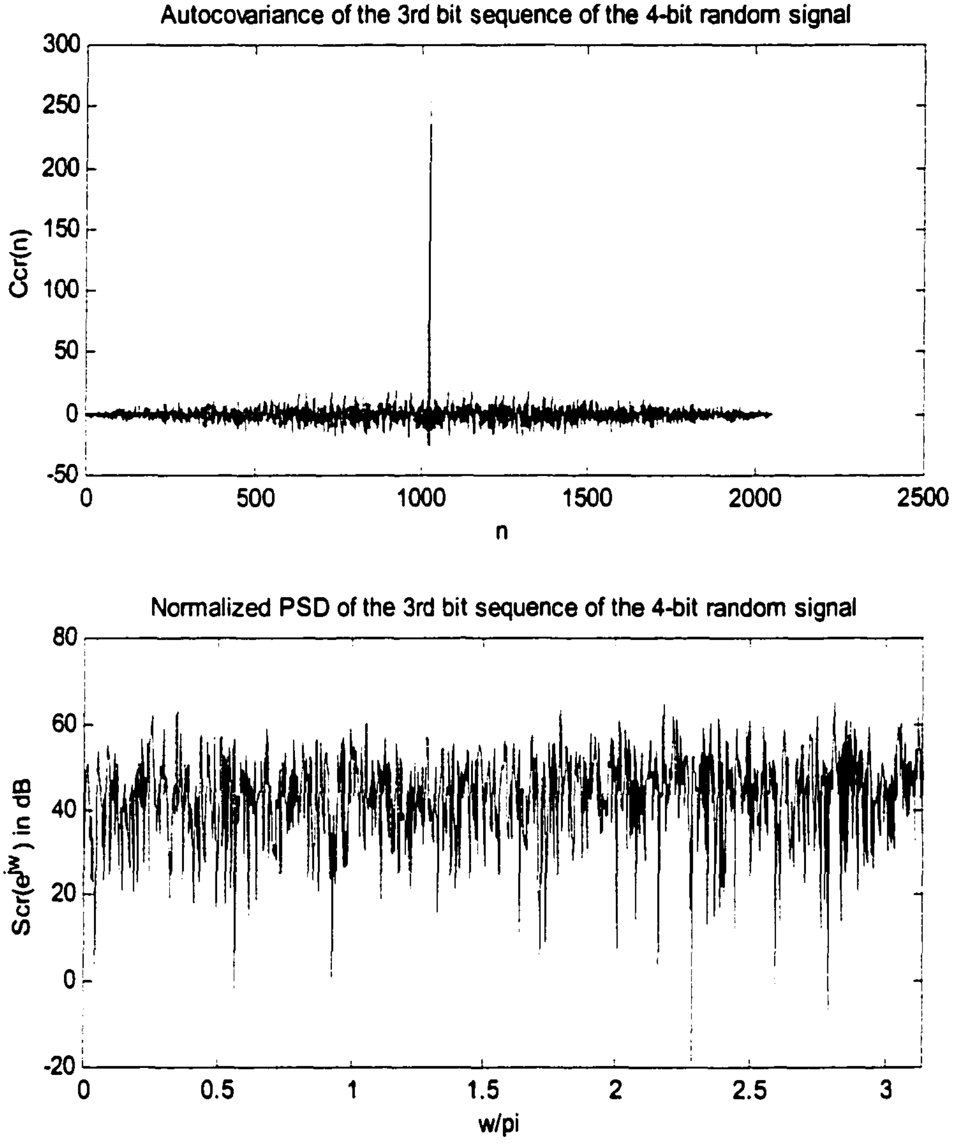


Figure 4-5. Autocovariance and PSD of the 3rd bit sequence of the four-bit random signal that is bandlimited between $3\pi/4$ rad/sam and π rad/sam.

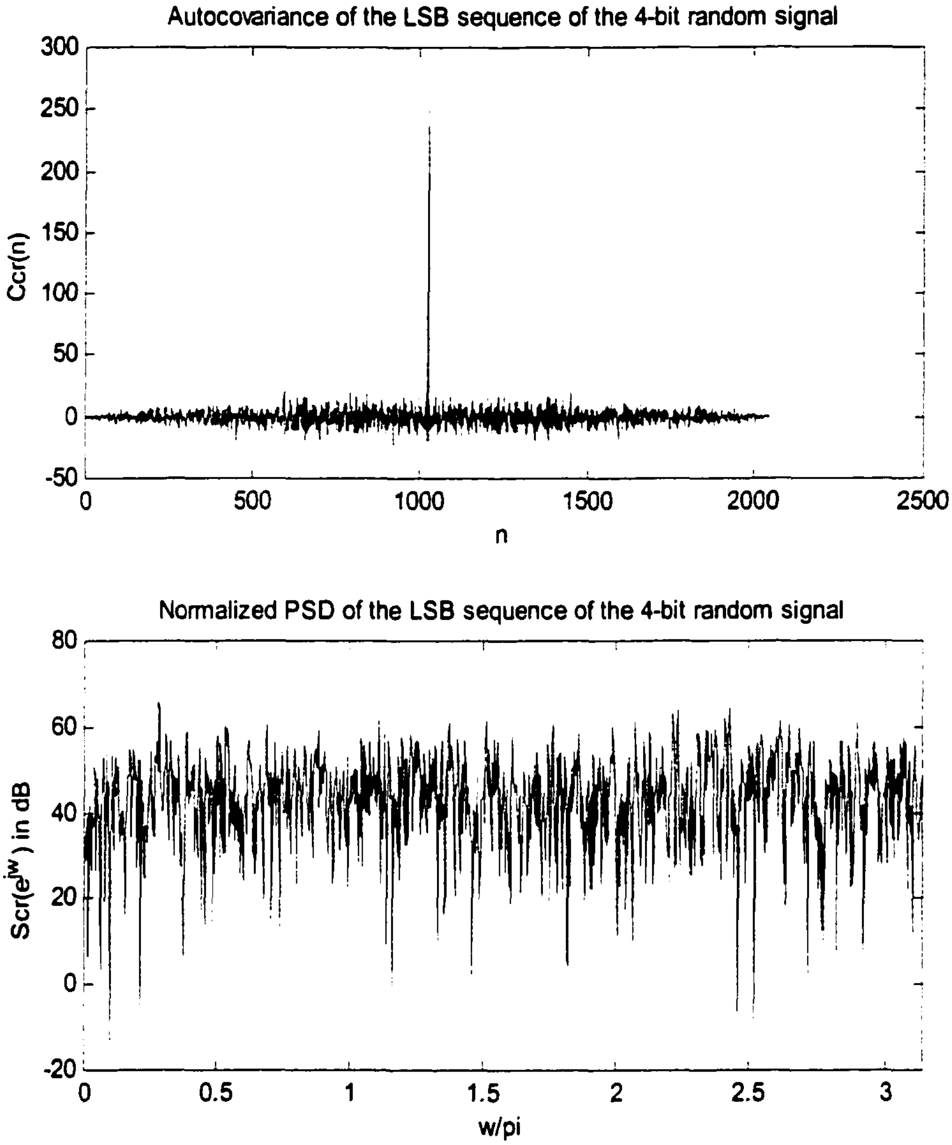


Figure 4-6. Autocovariance and PSD of the LSB sequence of the four-bit random signal that is bandlimited between $3\pi/4$ rad/sam and π rad/sam.

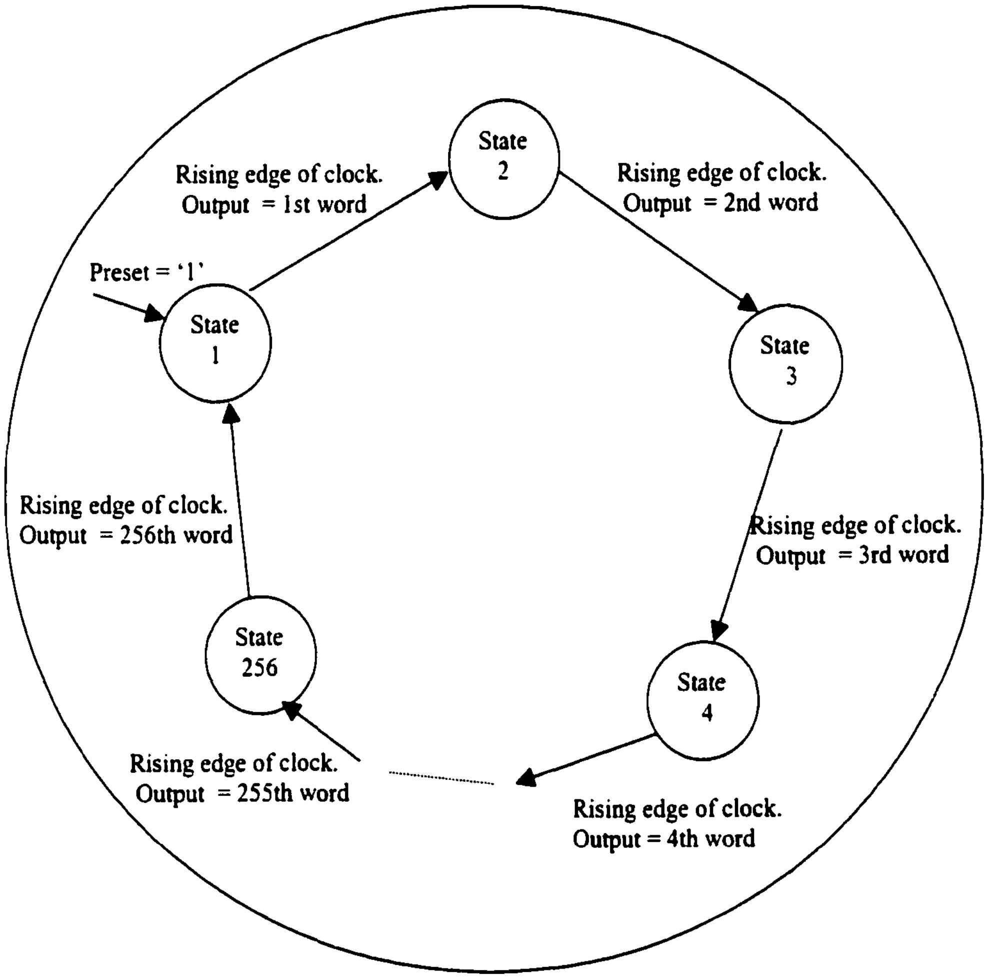


Figure 4-7. FSM that generates the two MSBs of a uniformly distributed colored random signal of length 256.

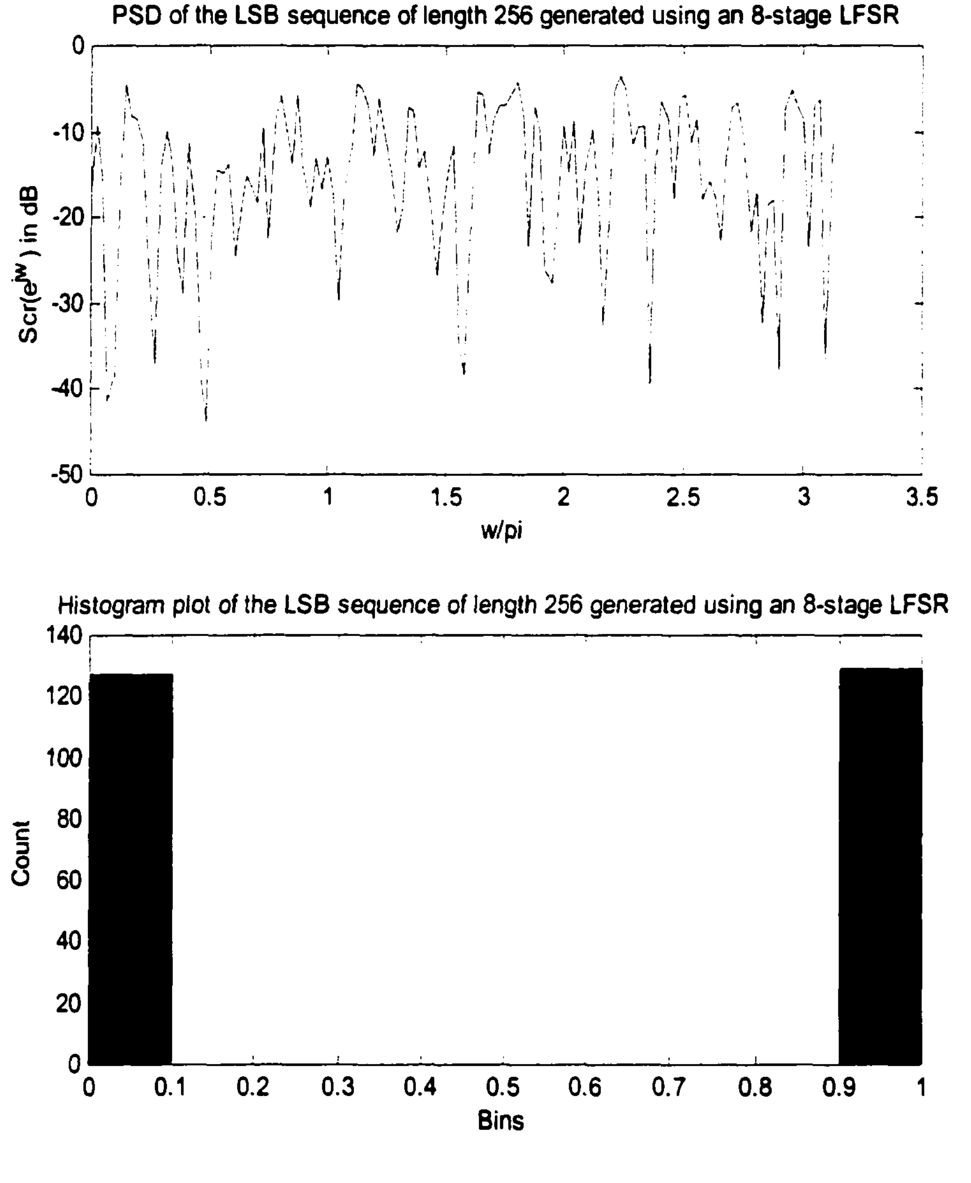


Figure 4-8. PSD and histogram plot of the LSB sequence of length 256 generated using an eight-stage LFSR.

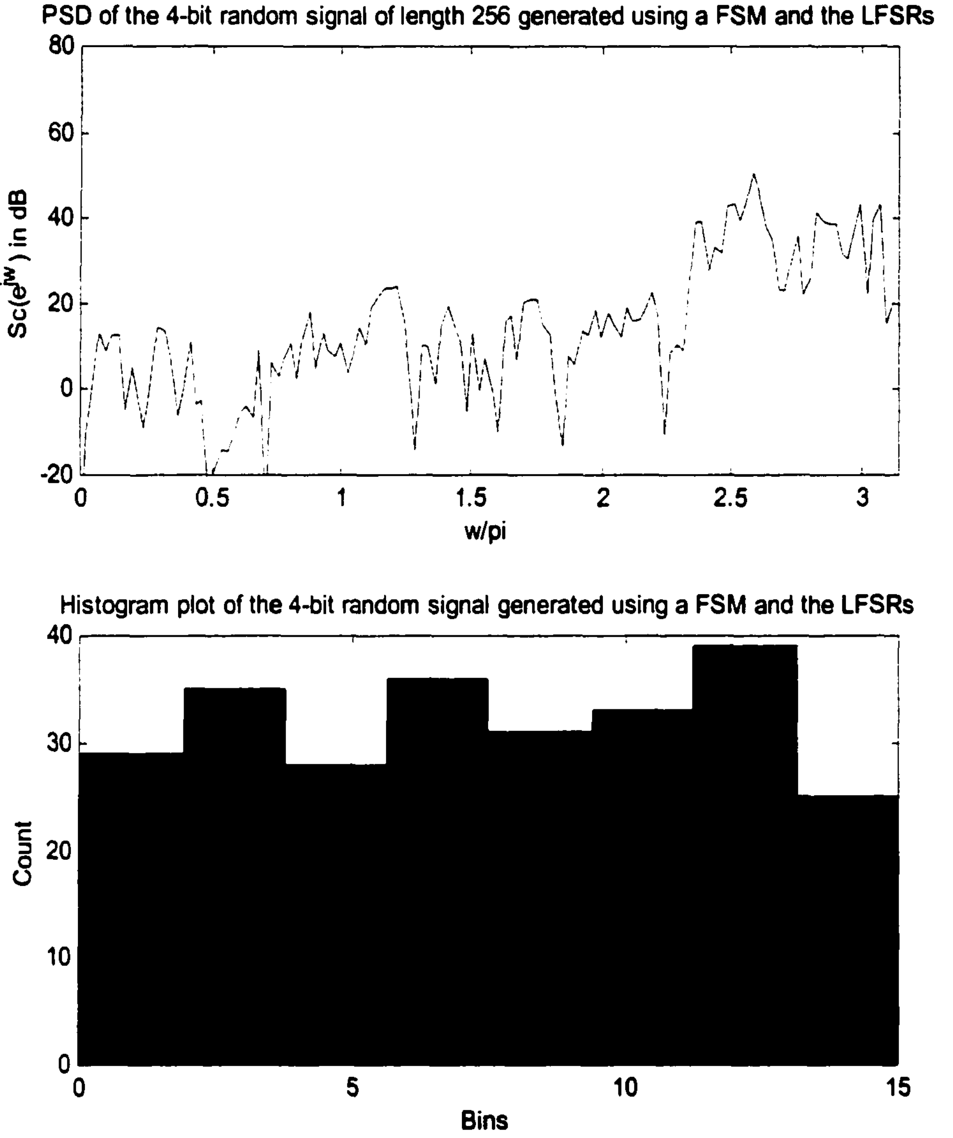
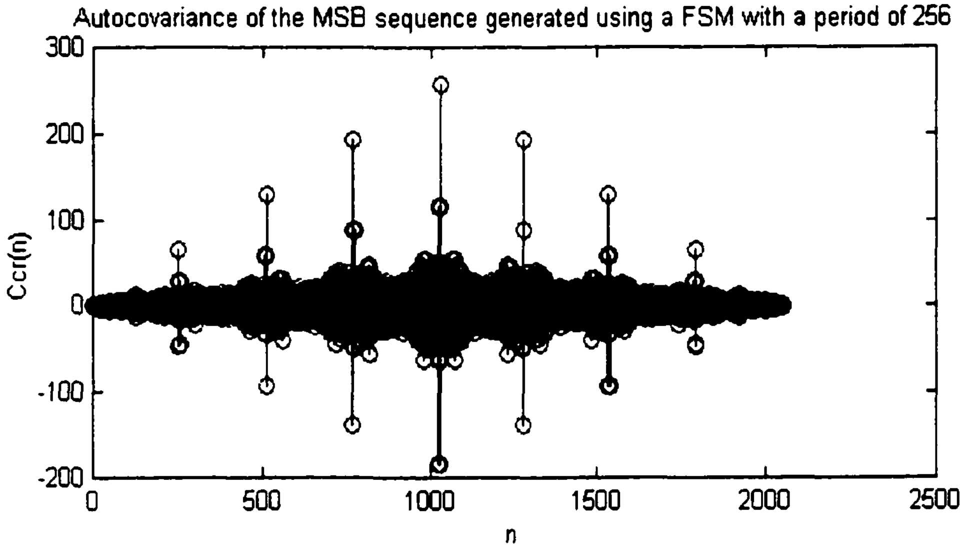


Figure 4-9. PSD and histogram plot of the four-bit random signal of length 256 and with a cutoff frequency $3\pi/4$ generated using a FSM and LFSRs.



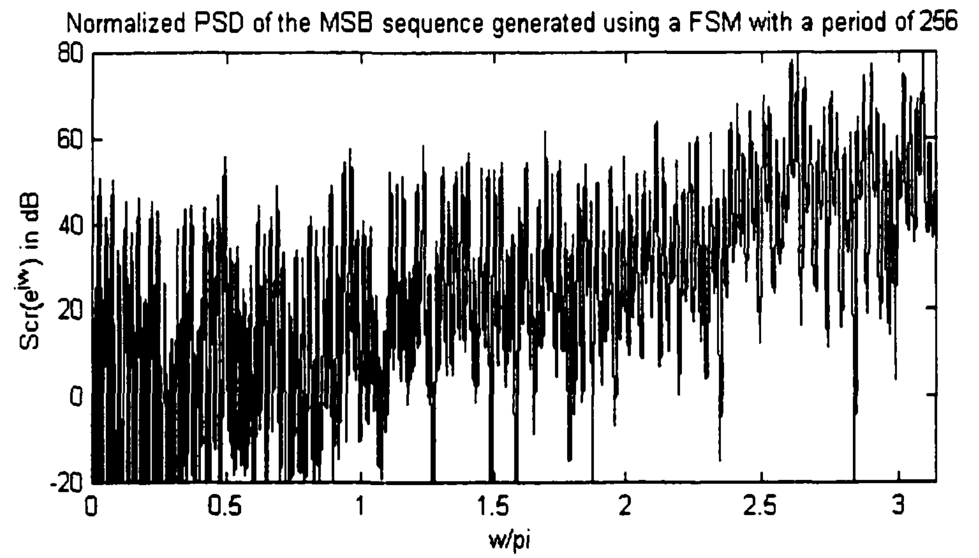


Figure 4-10. Autocovariance and PSD of the MSB sequence of a four- bit random signal generated using a FSM with a period of 256.

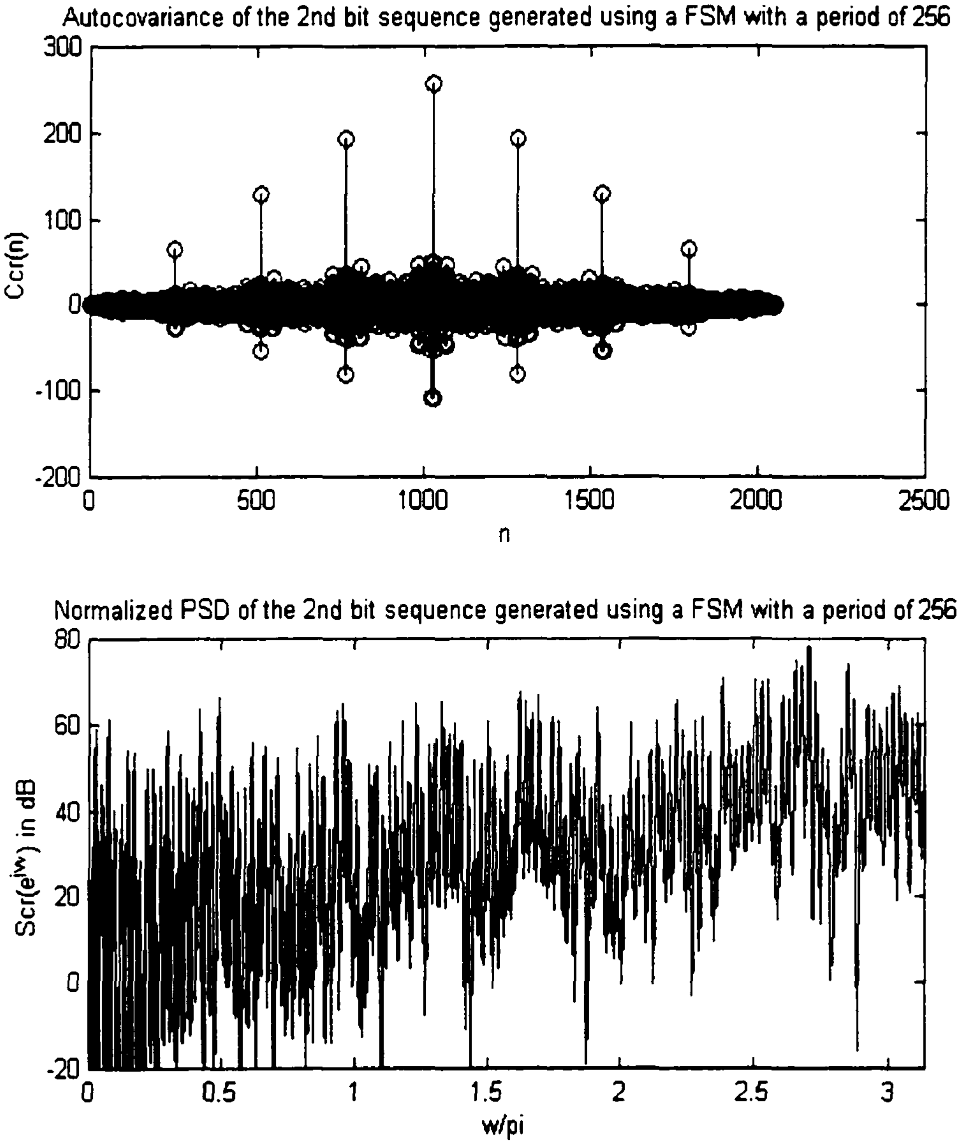
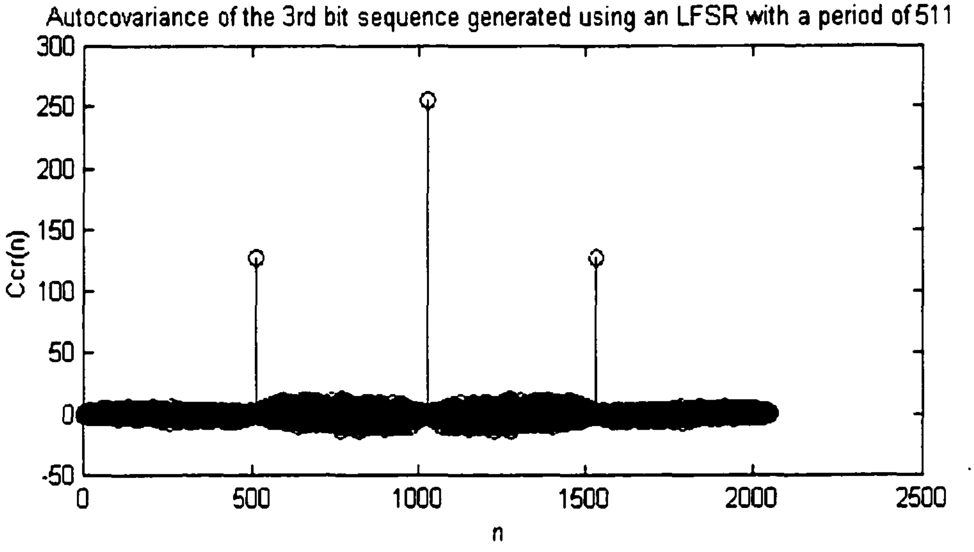


Figure 4-11. Autocovariance and PSD of the 2nd MSB sequence of a four-bit random signal generated using a FSM with a period of 256.



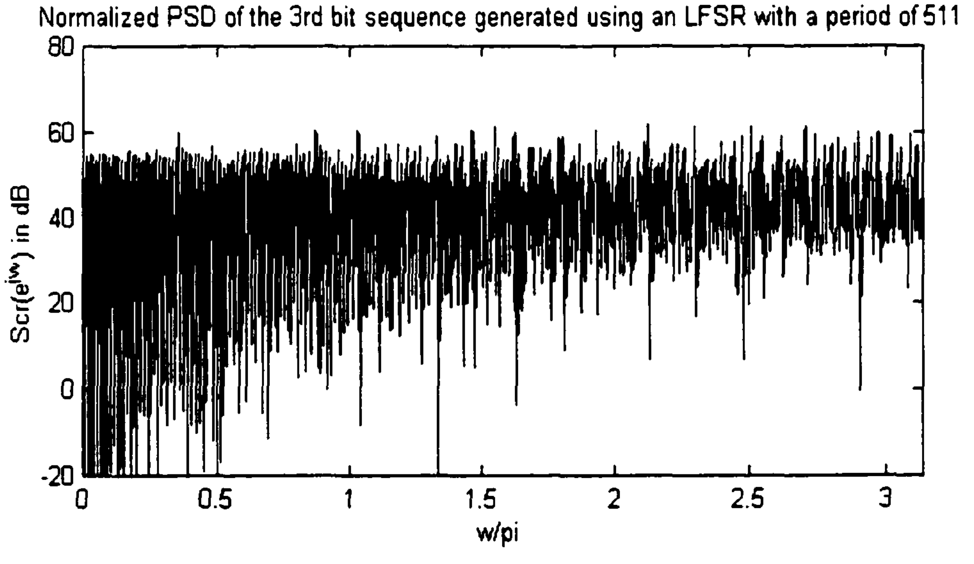


Figure 4-12. Autocovariance and PSD of the 3rd MSB sequence of a four-bit random signal generated using an LFSR with a period of 511.

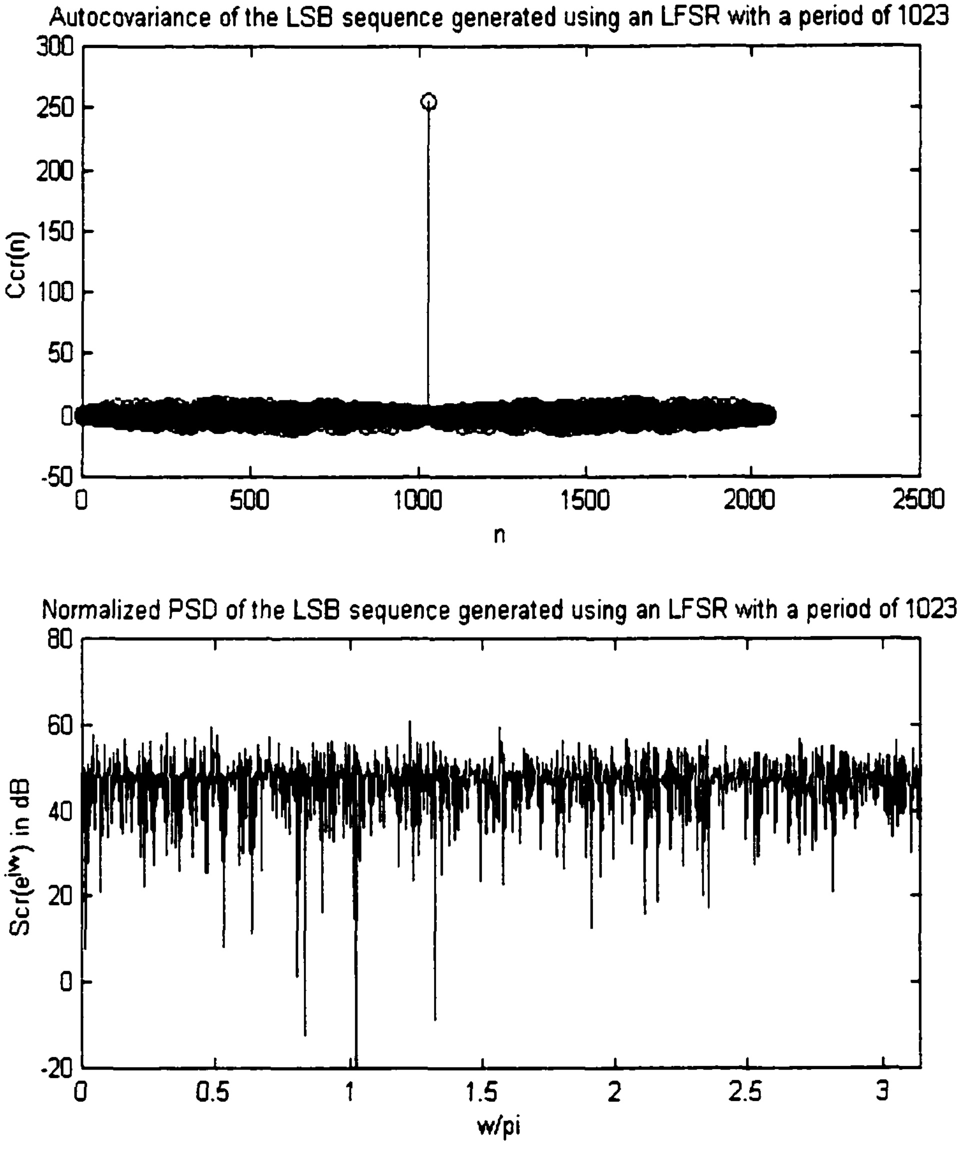
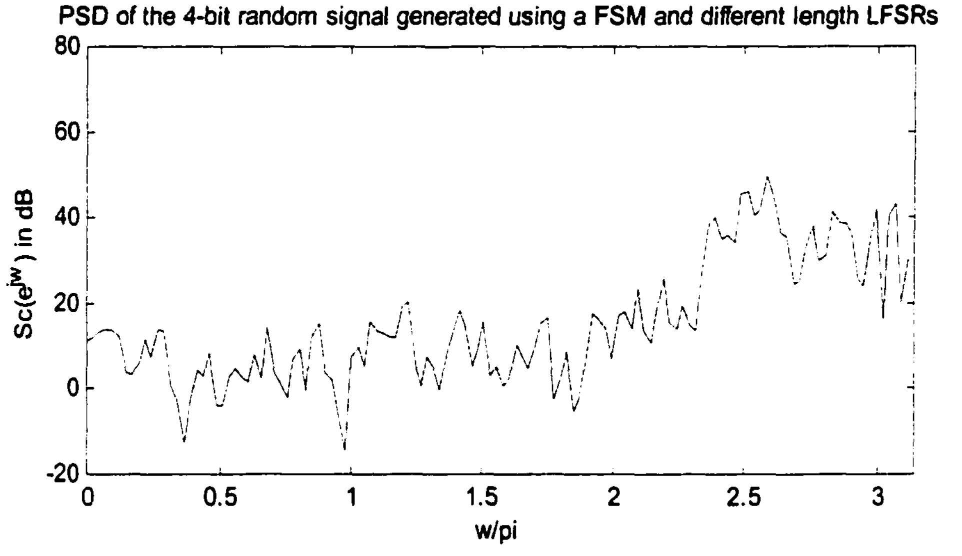


Figure 4-13. Autocovariance and PSD of the LSB sequence of a four-bit random signal generated using an LFSR with a period of 1023.



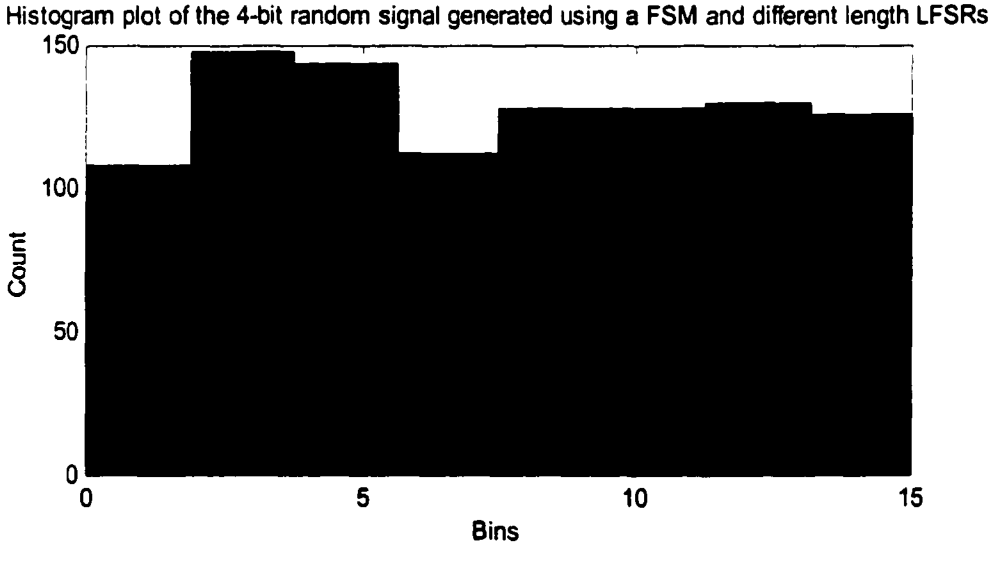


Figure 4-14. PSD and histogram plot of the uniformly distributed highpass filtered random signal generated using a FSM and different length LFSRs that is bandlimited between $3\pi/4$ rad/sam and π rad/sam.

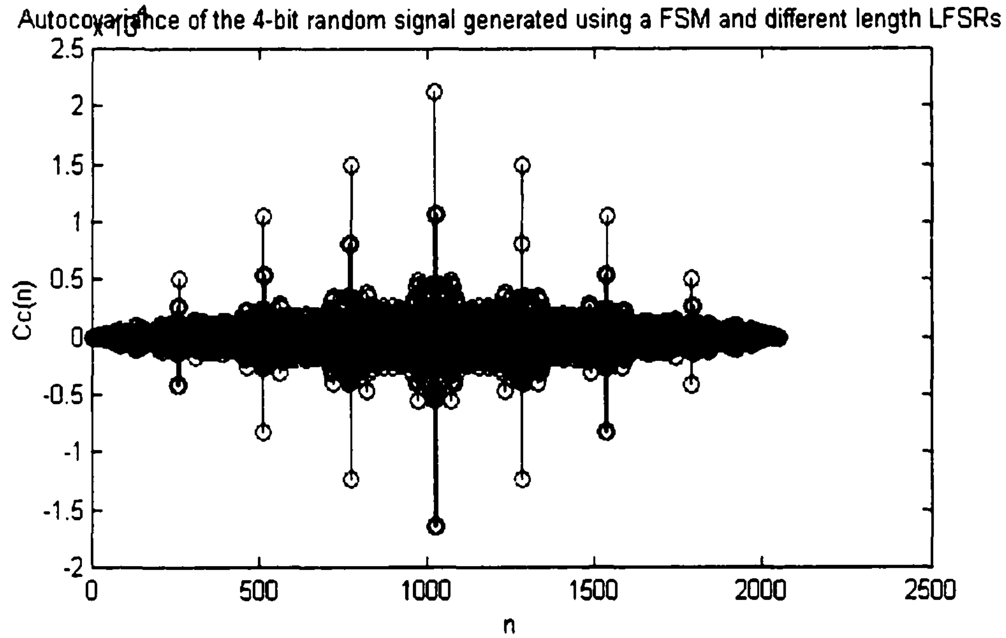


Figure 4-15. Autocovariance of the uniformly distributed highpass filtered random signal generated using a FSM and different length LFSRs .

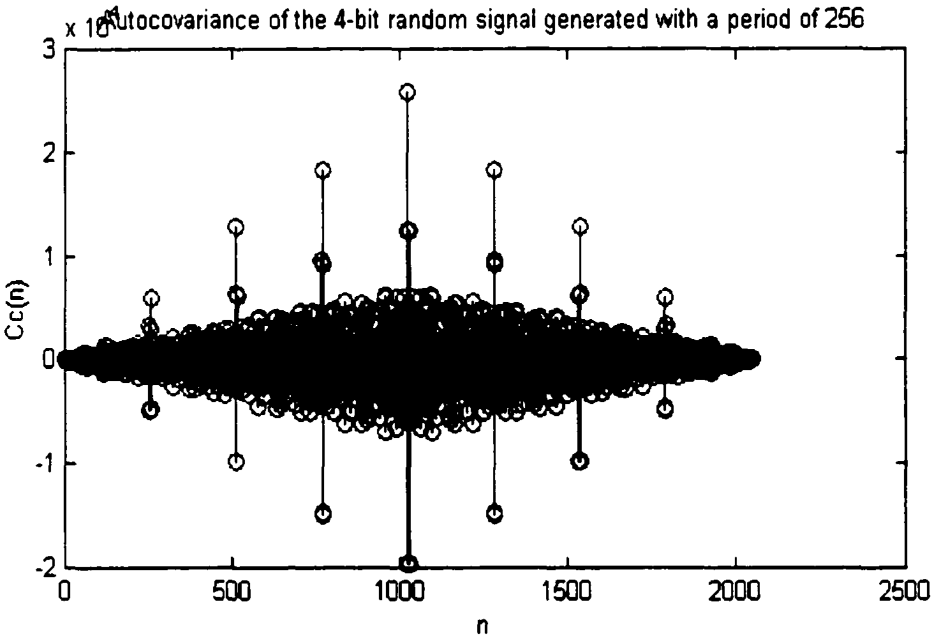


Figure 4-16. Autocovariance of the uniformly distributed highpass filtered random signal generated with a period of 256.

CHAPTER 5

CONCLUSION

In this thesis, an algorithm that can generate a uniformly distributed colored random signal in real time is developed. The uniformly distributed colored random signal can be generated with different power spectral densities like a lowpass filtered random signal, a bandpass filtered random signal or a highpass filtered random signal. Several examples illustrate the effects of the parameters such as the length of the white random input signal and the length of the linear phase FIR filter's impulse response have on the generated colored random signal. The examples illustrate that if the length of the white input random signal is long (in the order of thousands) then the PSD of the output random signal is more constant, and if the length of the linear phase FIR filter's impulse response is long, then the transition at the discontinuity of the power spectrum is very sharp.

In this thesis, it is shown that in a B bit uniformly distributed colored random signal, the two most significant bits (MSBs) have a more significant influence on the signal's PSD than the remaining B–2 least significant bits (LSBs). Therefore, a B bit uniformly distributed colored random signal can be adequately generated by generating the two MSBs as colored bits and the remaining B–2 LSBs as white bits. The two MSBs are generated as colored bits using a finite state machine (FSM) and the B–2 LSBs are generated as white bits using Linear Feedback Shift Registers (LFSRs). The disadvantage of a FSM design is that the hardware requirements for the FSM implementation grow exponentially with the period of the random signal. To keep the hardware requirements low, the two MSBs are generated with a small period.

The B-2 LSBs can also be generated using a FSM, but the hardware requirements for an LFSR implementation are much less than the hardware requirements for a FSM implementation. Therefore, the LFSR implementation is preferred for generating the B-2 white bits. The tradeoff for simple hardware requirements is the influence of the LFSR generated sequence on the uniform distribution of the resulting random signal. The LFSR generated sequence influences the distribution of the random signal because using an LFSR it is very difficult to generate exactly the same bit sequence constructed by the algorithm. For a uniformly distributed binary sequence of length l with $(l-1)/2 + 1$ ones and $(l-1)/2$ zeros, $l! \,/\, ((l-1)/2)!\ ((l+1)/2)!$ unique sequences, each of length l can be generated. But polynomials do not exist to generate all the $l! \,/\, ((l-1)/2)!\ ((l+1)/2)!$ sequences. Therefore, it is very difficult to generate exactly the same bit sequence using the LFSR. But, this does not significantly alter the distribution of the colored random signal because the two MSBs make the most significant contribution for the uniform distribution of a random signal. Because the two MSBs are generated using a FSM, they are exactly same as the sequence constructed by the algorithm. Therefore, the uniform distribution of the random signal is not significantly altered. For example, in a three-bit uniformly distributed random signal, changing the LSB sequence does not significantly alter the uniform distribution of the random signal as long as the individual bit sequence is uniformly distributed. Because the LFSR generated sequence is uniformly distributed with $l/2$ zeros and $l/2+1$ ones, it does not significantly alter the distribution of the random signal.

In this thesis, it is also shown that generating a uniformly distributed colored random signal by generating the two MSBs with a small period and the succeeding B-2 LSBs with increasing periods reduce the side correlations of the random signal without much increase in the hardware requirements for a real time implementation on chip. This is because the two MSBs are generated using a FSM and the B-2 LSBs are generated using LFSRs.

The advantage of this real time implementation for generating a uniformly distributed colored random signal is that it does not require large computation. Because of this, the real time implementation can be operated at the clock speed of a DEM DAC. Therefore, the generated uniformly distributed colored random signal can be used in a DEM DAC as interconnection network's control signal. Many alternative approaches were tried during the research work but this algorithm was reported because it is far simpler than the others.

APPENDIX

PROGRAM LISTING

```
% ***************************************************
% ***             Filter Specifications           ***
% ***                                             ***
% ***   w_low     -  Low Pass Cut off Frequency   ***
% ***                of the Desired PSD           ***
% ***   w_up      -  High Pass Cut off Frequency  ***
% ***                of the Desired PSD           ***
% ***   fil_order -  Order of the filter          ***
% ***   w_low = 0 - LPF,   w_up = pi - HPF        ***
% ***************************************************

w_low = 3*pi/4;
w_up = pi;
fil_order = 33;


% *************************************************
% *** Calling the function filter_design that   ***
% *** that generates the time domain components ***
% *** of the filter                             ***
% ***                                           ***
% ***       h - Time domain components of the   ***
% ***       filter                              ***
% ***   H - Frequency response of the           ***
% ***       filter                              ***
% ***  psdhmat - Power Spectral Density of the  ***
% ***            filter using matlab command PSD***
% *************************************************

h = filter_design(w_low,w_up,fil_order);
H = abs(fft(h));

psdhmat = psd(h);


% *****************************************************
% * Parameters Specifications for the random signal  *
% *                                                   *
% * len      - length of the random signal            *
% * no_bits  - Number of bits for binary conversion  *
% *****************************************************

bit_len = 8;
len = 2^bit_len-(fil_order-1);
no_bits = 4;

val = round((len-1)/2);
```

```
f = (0:val)/val*pi;

% ****************************************************
% * Generation of Normally Distributed White Signal  *
% *                                                  *
% * nw       - normally distributed white signal of  *
% *            length len with zero mean and         *
% *            unit variance                         *
% * NW       - Frequency Spectrum of nw              *
% * corrnw   - Autocorrelation of nw                 *
% * psdnw    - Power Spectral Density of nw using    *
% *            corrnw (psdnw -- FFT --> corrnw)      *
% * psdnwmat - Power Spectral Density of nw using    *
% *            using the matlab command PSD          *
% * meannw   - mean of nw                            *
% * varnw    - variance of nw                        *
% ****************************************************

nw = randn(len,1);
NW = abs(fft(nw));

psdnwf = (NW.^2);
psdnwf = psdnwf/max(psdnwf);

corrnw = xcorr(nw);
psdnw = abs(fft(corrnw));

% Normalizing the Power Spectral density
psdnw = psdnw/max(psdnw);

psdnwmat = psd(nw);

meannw = mean(nw);
varnw = cov(nw);


% ****************************************************
% *Generation of Normally Distributed colored Signal *
% *    by filtering nw using the designed filter     *
% *                                                  *
% * nc  - normally distributed colored signal of     *
% *       length len                                 *
% * NC  - Frequency Spectrum of nc                   *
% * corrnc   - Autocorrelation of nc                 *
% * psdnc    - Power Spectral Density of nc using    *
% *            corrnc (psdnc -- FFT --> corrnc)      *
```

```
% * psdncmat - Power Spectral Density of nw using    *
% *            using the matlab command PSD           *
% * meannc   - mean of nc                             *
% * varnc    - variance of nc                         *
% ******************************************************

nc = conv(h,nw);
lennc = length(nc);
%val2 = (lennc-1)/2;
%f2 = (0:val2)/val2*pi;

NC = abs(fft(nc));

psdncf = NC.^2;
psdncf = psdncf/max(psdncf);

corrnc = xcorr(nc);
psdnc = abs(fft(corrnc));

% Normalizing the Power Spectral density
psdnc = psdnc/max(psdnc);

psdncmat = psd(nc);

meannc = mean(nc);
varnc = cov(nc);


% ******************************************************
% * Calling the function that converts the normally  *
% *     distributed colored signal to uniformly       *
% *               distributed signal                  *
% *                                                   *
% * ucb - Uniformly distributed colored signal in    *
% *       binary form using no_bits                   *
% * ucwm- Uniformly distributed colored signal        *
% *       with nonzero mean                           *
% * meanucwm   - mean of nc                           *
% * varucwm    - variance of nc                       *
% ******************************************************

[ucb] = normal2uniform(nc,no_bits);

ucwm = bin2dec(ucb);

meanucwm = mean(ucwm);
varucwm = cov(ucwm);
```

```
% ******************************************************
% *** Generation of Uniformly Distributed colored  ***
% ***           signal with zero mean              ***
% ***                                              ***
% *** uc     - uniformly distributed colored signal ***
% ***          with zero mean                      ***
% *** UC     - Frequency Spectrum of uc            ***
% *** corruc   - Autocorrelation of uc             ***
% *** psduc    - Power Spectral Density of uc using***
% ***            corruc (psduc -- FFT --> corruc)  ***
% *** psducmat - Power Spectral Density of uc using***
% ***            using the matlab command PSD      ***
% *** meanuc   - mean of uc                        ***
% *** varuc    - variance of uc                    ***
% ******************************************************

uc = ucwm - meanucwm;
UC = abs(fft(uc));

psducf = UC.^2;
psducf = psducf/max(psducf);

corruc = xcorr(uc);
psduc = abs(fft(corruc));

% Normalizing the Power Spectral density
psduc = psduc/max(psduc);

psducmat = psd(uc);

meanuc = mean(uc);
varuc = cov(uc);


% ****************************
% *** Plotting the figures ***
% ****************************


val1 = (fil_order-1)/2;
f1 = (0:val1)/val1*pi;

lenpsdhmat = length(psdhmat);
fh = (0:lenpsdhmat-1)/lenpsdhmat*pi;
```

```
figure(301);
subplot(211);
stem(h);
title('Impulse response of the filter with a cutoff
frequency of pi/2 rad/sam and of order 63');
xlabel('n');
ylabel('h(n)');

subplot(212);
plot(f1, 20*log10(H(1:val1+1)));
title('Frequency response of the filter with a cutoff
frequency of pi/2 rad/sam and of order 63');
xlabel('w/pi');
ylabel('H(e^j^w) in dB');

%figure(311);
%subplot(313);
%plot(fh, 20*log10(psdhmat));
%title('PSD of the filter with a cutoff freqency at 3pi/4
and filter order 99');
%xlabel('w/pi');
%ylabel('PSD - Sh in dB');


figure(302);

subplot(311);
plot(f, 20*log10(NW(1:val+1)));
title('Frequency Spectrum of the normally distributed white
signal of length 925');
xlabel('w/pi');
ylabel('W(e^j^w) in dB');

subplot(312);
plot(f, 20*log10(NC(1:val+1)));
title('Frequency Spectrum of the normally distributed
colored signal of length 1024');
xlabel('w/pi');
ylabel('X(e^j^w) in dB');

subplot(313);
plot(f, 20*log10(UC(1:val+1)));
title('Frequency Spectrum of the uniformly distributed
colored signal of length 1024');
xlabel('w/pi');
ylabel('C(e^j^w) in dB');
```

```
lenpsdnwmat = length(psdnwmat);
fnw = (0:lenpsdnwmat-1)/lenpsdnwmat*pi;

lenpsdncmat = length(psdncmat);
fnc = (0:lenpsdncmat-1)/lenpsdncmat*pi;

lenpsducmat = length(psducmat);
fuc = (0:lenpsducmat-1)/lenpsducmat*pi;

figure(304);

%subplot(211);
%stem(corrnw);
%title('Autocorrelation of the normally distributed white
signal of length 925');
%xlabel('n');
%ylabel('Rw(n)');

subplot(212);
plot(fnw, 20*log10(psdnwmat));
axis([0 pi -20 20]);
title('PSD of the normally distributed white signal of
length 2015');
xlabel('w/pi');
ylabel('Sw(e^j^w) in dB');

figure(305);

%subplot(211);
%stem(corrnc);
%title('Autocorrelation of the normally distributed colored
signal of length 1024');
%xlabel('n');
%ylabel('Rx(n)');

subplot(212);
plot(fnc, 20*log10(psdncmat));
title('PSD of the normally distributed colored signal of
length 1024');
axis([ 0 pi -130 40]);
xlabel('w/pi');
ylabel('Sx(e^j^w) in dB');

figure(306);

%subplot(211);
%stem(corruc);
```

```
%title('Autocorrelation of the uniformly distributed
colored signal of length 1024');
%xlabel('n');
%ylabel('Rc(n)');

subplot(212);
plot(fuc, 20*log10(psducmat));
axis([ 0 pi 0 100]);
title('PSD of the uniformly distributed colored signal of
length 1024 with filter order 63');
xlabel('w/pi');
ylabel('Sc(e^j^w) in dB');


figure(307);

subplot(311);
hist(nw);
title('Histogram plot of the 4-bit normally distributed
white signal of length 925');
xlabel('Bin');
ylabel('Count');

subplot(312);
hist(nc);
title('Histogram plot of the 4-bit normally distributed
colored signal of length 1024');
xlabel('Bin');
ylabel('Count');

subplot(313);
hist(uc,2^no_bits);
title('Histogram plot of the 4-bit uniformly distributed
colored signal of length 1024');
xlabel('Bin');
ylabel('Count');
```

```
% ************************************************
% ***  Function for generating the impulse     ***
% ***        response of the filter            ***
% ************************************************


function [h] = filter_design(w_low,w_up,fil_order)


% ********************************************
% ***       Parameter Description          ***
% ***                                      ***
% ***  h  -  Time domain components of the ***
% ***        filter                        ***
% *** w_low - Low Pass Cut off Frequency   ***
% ***         of the Desired PSD           ***
% *** w_up - High Pass Cut off Frequency   ***
% ***        of the Desired PSD            ***
% ***  fil_order  -  Order of the filter   ***
% ***    w_low = 0 - LPF, wh = pi - HPF    ***
% ********************************************


n = -(fil_order-1)/2 :1: (fil_order-1)/2;
j=1;

for k = -(fil_order-1)/2:-1

     h(j) = (sin(k*w_up)-sin(k*w_low))/(pi*k);
     j=j+1;

end

h(j) = (w_up-w_low)/pi;
j=j+1;

for k = 1:(fil_order-1)/2

     h(j) = (sin(k*w_up)-sin(k*w_low))/(pi*k);
     j=j+1;

end
```

```
% *********************************************
% ***    Function for converting normally   ***
% *** distibuted real sequence to uniformly***
% ***      distributed binary sequence     ***
% *********************************************


function [uni] = normal2uniform(nor, no_bits)


% ***************************************************
% ***           Parameter description            ***
% ***                                            ***
% *** nor - Normally distributed real sequence   ***
% *** uni - Uniformly distributed binary sequence***
% *** no_bits - Number of bits for binary        ***
% ***           representation of the real number***
% ***************************************************

norlen = length(nor);
quantlen = 2^no_bits;
count = floor(norlen/quantlen);

i = 0:quantlen-1;
quant = dec2bin(i,no_bits);

[norasc,pos] = sort(nor);

j = 1;
num = 1;

for m = 1:norlen

     if num > count
          j = min(j+1,quantlen);
          num = 1;
     end
     uni(pos(m),:) = quant(j,:);
     num = num+1;

end

y1 = bin2dec(uni);
```

```
%figure(1);
%subplot(211);
%hist(nor);

%subplot(212);
%hist(y1,quantlen);
```

```
% ************************************************
% ***  Function for histogram plot of each bit ***
% *** of the binary random sequence obtained   ***
% *** from the decimal sequence                ***
% ************************************************


function [hist_seq] = hist_binary(bin_seq)


% ******************************************
% ***       Parameter Description        ***
% ***                                    ***
% *** bin_seq - Binary sequence          ***
% *** hist_seq- Binary sequence for which***
% ***           histogram can be plotted ***
% ******************************************

[bin_seqlen, bin_seqbits] = size(bin_seq);


% ************************************
% *** MSB - Left most Bit          ***
% ***       b1 (i=1 in iteration)  ***
% *** LSB - Right most Bit - bm    ***
% ************************************

for i = 1:bin_seqbits

     hist_seq(:,i) = bin2dec(bin_seq(:,i));
     figure(100);
     subplot(bin_seqbits,1,i);
     hist(hist_seq(:,i));
     title('Histogram plot of each bit of the binary signal
     - MSB first, LSB last');
     xlabel('Bins');
     ylabel('Count');

end
```

```
% ************************************************
% *** Function for finding the Autocovariance   ***
% *** and power spectral density of each bit of ***
% *** the binary random sequence obtained from  ***
% *** the decimal sequence                      ***
% ************************************************

function [covb, psdb] = bits_psd(bin_seq);


% ***************************************************
% ***              Parameter Description          ***
% ***                                             ***
% *** bin_seq - Binary sequence                   ***
% *** covb -  Matrix in which each column has the ***
% ***         autocovariance of the corresponding ***
% ***         bit of the binary sequence          ***
% *** psdb -  Matrix in which each column has the ***
% ***         power spectral density of the       ***
% ***         corresponding bit of the binary     ***
% ***         sequence                            ***
% ***************************************************


[bin_seqlen, bin_seqbits] = size(bin_seq);

% *************************************
% *** MSB - Left most Bit           ***
% ***       b1 (i=1 in iteration)   ***
% *** LSB - Right most Bit - bm     ***
% *************************************

for i = 1:bin_seqbits

   b = bin_seq(:,i);
   covb(:,i) = xcov(b);
   psdb(:,i) = abs(fft(covb(:,i)));

   lenpsdb = length(psdb(:,i));
   lenpsdbt = round((lenpsdb-1)/2);
   fr = (0:lenpsdbt)/lenpsdbt * pi;

   figure(10+i);
   subplot(211);
```

```
    plot(covb(:,i));
    title('Autocovariance of each bit of the binary
    sequence - MSB first, LSB last');
    xlabel('Lag');
    ylabel('Cb');

    subplot(212);
    plot(fr,psdb(1:lenpsdbt+1,i));
    title('Normalized PSD of each bit of the binary
    sequence - MSB first, LSB last');
    xlabel('Normallized frequency');
    ylabel('PSD - Sb in db');

end
```

```
% ********************************************
% ***  Function for generating the Random  ***
% *** sequence using Linear Feed back Shift***
% ***           Register (LFSR)            ***
% ********************************************

function [y] = lfsrgeneration(degree,taps)

%***************************************************
% ***          Parameter description            ***
% *** degree - Degree of the shift register      ***
% ***          polynomial generating the sequence ***
% *** taps   - Tap positions of the LFSR         ***
% *** y      - Generated random sequence         ***
% **************************************************

% *** Initial state of the LFSR ***
x = ones(1,degree);

true = 1;
i = 1;
tapsize = length(taps);

while (true ~= 0 & i < (2^degree))

    i = i+1;
    for j = 2:degree

        x(i,j)=x(i-1,j-1);

    end

    temp = 0;

    % *** Feedback using xor gate ***
    for j = 1:tapsize

        temp = temp + x(i-1,taps(j));

    end

    % *** x - generated binary random sequence ***
    x(i,1) = mod(temp,2);
```

```
        repeat = x(1,:) ~= x(i,:);
        true = sum(repeat);

end


%*** Extracting one of the columns from the binary
     sequence ***
y = x(:,1);
```

Program name : statemachine.cpp

```
/* C-Program to generate the state diagram VHDL description
of a random signal. The binary random signal should be in
.txt format and the generated VHDL file will be in .vhd
format.*/

#include<stdio.h>
#include<math.h>

#define vectorlength 127
#define nobits 7

main()
{
  FILE *f1,*f2;

  char *str[40];
  char txtfile[20];
  char vhdlfile[20];
  char rpy[nobits];
  char tmp;

  int stateno[vectorlength+1];
  int i,j;

  /* Initializing the array stateno */
  for (i=0;i<vectorlength;i++)
    {
      stateno[i] = i+1;
    }

  stateno[i] = 1;

  /* Getting the txt file name where the truth table is
     stored */

  printf("The truth table is stored as text file\n\n");
  printf("Enter the name of this file with a .txt
          extension\n");
  scanf("%s",txtfile);

  /* Getting the vhdl file name where the output is
     stored */

  printf("The output file should be stored as a .vhd
          file\n\n");
```

```
printf("\nEnter the name of the file into which it
           writes\n");
scanf("%s",vhdlfile);

str[1]  = "library IEEE;";
str[2]  = "use IEEE.std_logic_1164.all;";
str[3]  = "use IEEE.std_logic_arith.all;";
str[4]  = "use IEEE.std_logic_unsigned.all;";

str[5]  = "entity statemachine is";
str[6]  = "port (";
str[7]  = "clk : in std_logic;";
str[8]  = "preset : in std_logic;";
str[9]  = "yrp : out std_logic_vector(";
str[10] = "downto 0)";
str[11] = ");";
str[12] = "end statemachine;";

str[13] = "architecture fsm of statemachine is";
str[14] = "type states is (";
str[15] = "s";
str[16] = "signal presentstate: states;";
str[17] = "begin";
str[18] = "process(clk)";

str[19] = "if clk = '1' then";
str[20] = "if reset = '1' then";
str[21] = "presentstate <=";
str[22] = "yrp <=";
str[23] = "else";
str[24] = "end if;";
str[25] = ";";

str[26] = "case presentstate is";
str[27] = "when";
str[28] = " =>";
str[29] = "when others => null;";
str[30] = "end case;";

str[31] = "end process;";
str[32] = "end fsm;";

/* Opening the txt file in read mode */

// f1=fopen("y127.txt", "r");

f1=fopen(txtfile, "r");
```

```
rewind(f1);

/* Opening the vhdlfile in write mode */
// f2=fopen("try.vhd", "w");
f2=fopen(vhdlfile, "w");

/* Writing the intial statements to the vhdlfile */

fprintf(f2, "%s\n", str[1]);
fprintf(f2, "%s\n", str[2]);
fprintf(f2, "%s\n", str[3]);
fprintf(f2, "%s\n\n", str[4]);

fprintf(f2, "%s\n", str[5]);
fprintf(f2, "\t%s\n", str[6]);

fprintf(f2, "\t\t%s\n", str[7]);
fprintf(f2, "\t\t%s\n", str[8]);
fprintf(f2, "\t\t%s", str[9]);
fprintf(f2, "%d %s", nobits-1, str[10]);
/* this is for the downto statement for o/p yrp */

fprintf(f2, "%s\n", str[11]);
fprintf(f2, "%s\n\n", str[12]);

fprintf(f2, "%s\n\n", str[13]);
fprintf(f2, "\t%s ", str[14]);

for(i=0;i<vectorlength-1;i++)
  {
    fprintf(f2, "%s%d, ", str[15], stateno[i]);
  }

fprintf(f2, "%s%d", str[15], stateno[i]);
fprintf(f2, "%s\n", str[11]);
fprintf(f2, "\t%s\n\n", str[16]);

fprintf(f2, "%s\n\n", str[17]);
fprintf(f2, "%s\n", str[18]);
fprintf(f2, "%s\n", str[17]);
fprintf(f2, "\t%s\n", str[19]);
fprintf(f2, "\t\t%s\n", str[20]);
fprintf(f2, "\t\t\t%s %s%d%s\n", str[21], str[15],
        stateno[0], str[25]);

fseek(f1, 5, 0);
for(j=0;j<nobits;j++)
```

```
        {
            fscanf(f1, "%c",&rpy[j]);
            fseek(f1, 5, 1);
        }
    fprintf(f2,"\t\t\t%s \"%.7s\" %s\n", str[22], rpy,
            str[25]);

    fprintf(f2, "\t\t%s\n", str[23]);
    fprintf(f2, "\t\t\t%s\n", str[26]);
    rewind(f1);
    fseek(f1, 5, 0);

    for(i=0;i<vectorlength;i++)
      {
         fprintf(f2, "\t\t\t\t%s %s%d %s\n", str[27],
                 str[15], stateno[i], str[28]);
         for(j=0;j<nobits;j++)
        {
            fscanf(f1, "%c",&rpy[j]);
            fseek(f1, 5, 1);
        }

         fscanf(f1,"\n");
         fprintf(f2, "\t\t\t\t\t%s \"%.7s\" %s\n",
                 str[22], rpy, str[25]);
         /* change here according to the number of
            bits used. */
         fprintf(f2, "\t\t\t\t\t%s\n", str[19]);
         fprintf(f2, "\t\t\t\t\t\t%s %s%d%s\n", str[21],
                 str[15], stateno[i+1], str[25]);
         fprintf(f2, "\t\t\t\t\t%s\n", str[24]);
      }

    fprintf(f2, "\t\t\t\t%s \n", str[29]);
    fprintf(f2, "\t\t\t%s\n", str[30]);
    fprintf(f2, "\t\t%s\n", str[24]);
    fprintf(f2, "\t%s\n", str[24]);
    fprintf(f2, "%s\n", str[31]);
    fprintf(f2, "%s\n", str[32]);

    fclose(f1);
    fclose(f2);

}
```

BIBLIOGRAPHY

[1] P. Stubberud and J.W. Bruce, "An analysis of harmonic distortion and integral nonlinearity errors in DACs," *Proc. of the Midwest Symposium on Circuits and Systems*, August 1999.

[2] R.J. Van de Plaasche, "Dynamic element matching for high accuracy monolithic D/A converters," *IEEE J. Solid-State Circuits*, December 1976.

[3] P. Stubberud and J.W. Bruce, "An analysis of Dynamic Element Matching Flash Digital to Analog Converters," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, February 2001.

[4] J.W. Bruce and P. Stubberud, "Generalized cube networks for implementing dynamic element matching network," *Proc. of the Midwest Symposium on Circuits and Systems*, August 1998.

[5] P. Stubberud, J.W. Bruce and B. Steadman, "A DAC architecture with a hardware efficient dynamic element matching network," *Proc. Second International Workshop on Design of Mixed-Mode Integrated Circuits and Applications*, 1998.

[6] Donald E. Knuth, "The art of computer programming, vol2, Seminumerical Algorithms," Third Edition, Addison-Wesley, 1997.

[7] Martina F. Schollmeyer and William H. Tranter, "Noise Generators for the Simulation of Digital Communication Systems," *Proc. of the 24th annual symposium on Simulation*, April 1991.

[8] Klemens Letsch and Rolf Matzner, "On the Construction of a Random Process With Given Power Spectral Density Function," *Proc. of the Midwest Symposium on Circuits and Systems*, August 1998.

[9] Athanasios Papoulis, "Probability, random variables, and stochastic processes," Third Edition, McGraw-Hill, 1991.

[10] George R. Cooper and Clare D. McGillem, "Probabilistic methods of Signal and System Analysis," Third Edition, Oxford University Press, 1999.

[11] Alan V. Oppenheim and Ronald W. Schafer, "Discrete-time signal processing," Prentice-Hall, 1989.

[12] S. Lawrence Marple, Jr., "Digital Spectral Analysis with Applications," Prentice-Hall, 1987.

[13] David R. Smith, "Digital Transmission Systems," Van Nostrand Reinhold, 1993.

[14] Robert C. Dixon, "Spread Spectrum Systems," John Wiley & Sons, 1984.

[15] Solomon W. Golomb, "Shift Register Sequences," Aegean Park Press, 1982.

[16] A. Menezes, P. van Oorschot and S. Vansone, "Handbook of Applied Cryptography," CRC Press, 1997.

[17] Harald Niederreiter, "Random Number Generation and Quasi-Monte Carlo Methods," SIAM, 1992.

[18] Rudolf Lidl and Harald Niederreiter, "Encyclopedia of mathematics and its applications 20 - Finite Fields," Cambridge University Press, 1997.

[19] Christy J. Olsen, "Spectral Analysis of Random Number Generators," Masters Thesis, University of Nevada, Las Vegas, 1992.

[20] Kevin Skahill, "VHDL for Programmable logic," Addison-Wesley, 1996.

[21] Zainalabedin Navabi, "VHDL : analysis and modeling of digital systems," McGraw-Hill, 1998.

VITA

Graduate College
University of Nevada, Las Vegas

Venkatraghavan Bringi

Local Address:
4236 Grove Circle Apt1
Las Vegas, Nevada 89119

Home Address:
21 Nethaji Street
Cheyyar, Tamil Nadu, India 604 407

Degree:
Bachelor of Engineering, Electronics and Communication Engineering, 1998
University of Madras, Chennai

Special Honors and Awards:
Received gold medal for topping the 1994-1998 Electronics and Communication Engineering batch during undergraduate Study.

Member of τβπ.

Publications:
Venkatraghavan .B .V and Sundararajan .S, "Neural Networks Based Isolated Words Recognition," submitted to a national conference held at P.S.G College of Engineering, India, 1998.

Thesis Title:
Construction of a Random Signal with a Specific PSD and a Uniform PDF.

Thesis Examination Committee:
Chairperson, Dr. Peter Stubberud, Ph. D.
Committee Member, Dr. Rama Venkat, Ph. D.
Committee Member, Dr. Eugene McGaugh, Ph. D.
Graduate Faculty Representative, Dr. Evangelos Yfantis, Ph. D.