Visvesvaraya Technological University, Belagavi.

PROJECT REPORT

on

"XILINX Implementation of Gold Code Generator"

Project Report submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Engineering

in

Electronics and Communication Engineering

For the academic year 2019-2020

Submitted by

USN	Name
1CR16EC116	Praveen Yadav
1CR16EC137	Roshan C Thomas
1CR16EC142	Sagar T.H
1CR16EC192	Vinayaka R Karanji

Under the guidance of

Jyoti M Roogi

Assistant Professor

Department of ECE

CMRIT, Bengaluru

Department of Electronics and Communication Engineering

CMR Institute of Technology, Bengaluru – 560 037

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to Certify that the dissertation work **"XILINX Implementation of Gold Code Generator"** carried out by Praveen Yadav, Roshan C Thomas, Sagar T.H, Vinayaka R Karanji USN: 1CR16EC116, 1CR16EC137, 1CR16EC142, 1CR16EC192 bonafide students of **CMRIT** in partial fulfillment for the award of **Bachelor of Engineering** in **Electronics and Communication Engineering** of the **Visvesvaraya Technological University, Belagavi,** during the academic year **2019-20**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said degree.

Signature of Guide	Signature of HOD	Signature of Principal
_____	_____	_____
Jyoti M Roogi, Assistant Professor, Dept. of ECE., CMRIT, Bengaluru.	Dr. R. Elumalai Head of the Department, Dept. of ECE., CMRIT, Bengaluru.	Dr. Sanjay Jain Principal, CMRIT, Bengaluru.

External Viva

Name of Examiners	Signature & date
1.	
2	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose consistent guidance and encouragement crowned our efforts with success.

We consider it as our privilege to express the gratitude to all those who guided in the completion of the project.

We express our gratitude to Principal, **Dr. Sanjay Jain,** for having provided us the golden opportunity to undertake this project work in their esteemed organization.

We sincerely thank Dr. R. Elumalai**,** HOD**,** Department of Electronics and Communication Engineering, CMR Institute of Technology for the immense support given to us.

We express our gratitude to our project guide Mrs. Jyoti M Roogi**,** Assistant Professor, for her support, guidance and suggestions throughout the project work.

Last but not the least, heartful thanks to our parents and friends for their support.

Above all, we thank the Lord Almighty for His grace on us to succeed in this endeavour.

Abstract

This project proposes a new architecture for the implementation of an n-input XOR gate in ultrahigh-speed applications. The new circuitry makes it possible to let the conventional sequence logic with XOR feedback, such as Gold code generator, to work up to ten-gigabit per second. This project will systematically describe the principle of substituting a set of differential switches for an XOR gate. The proposed circuitry is demonstrated in a five-stage Gold code generator is implemented. The simulation results show that the delay of the proposed four-input XOR gate is so much improved as to let the five-stage (5,3) and (5,4,3,2) Gold code generator to work up to 10 Gb/s.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table 3.2 Seeding table

CHAPTER-1
INTRODUCTION

As a result of the rapidly growing volume of data transfer in telecommunication networks, data streams for digital video, high-definition television (HDTV), and color graphics are requiring higher and higher bandwidth. The design of ultrahigh-speed circuits for multi-gigabit data transfers becomes more and more important either in wireless or in optical communications. At the same time, the advantage of combining the baseband and the RF front-end on single CMOS chip for cost saving is strongly desired for highly integrated systems-on-chip (SoC) applications. Since the improvement of fT, the CMOS integrated circuits take over the territories thus far claimed by GaAs and InP devices. CMOS CML (current mode logic) circuits are suitable for the very high-speed applications. Most implementations of them are based on the combinational logic to avoid the speed degradation of feedback delay even in case of PRBS (Pseudo-Random Bit Sequence) or LFSR (Linear Feedback Shift Registers) generator which are typical applications of the logic with XOR feedback.

Although there are some works that deal with the ultrahigh-speed logic with feedback, they focus only on the divider which output signal is directly fed back to the input terminal. In other words, they are the special case of the logic with feedback because there is no delay in the feedback path. This project tries to solve the delay limitation of the sequence logic with XOR feedback. We propose a new architecture for the implementation of an n-input XOR gate in ultrahigh-speed applications.

Gold codes are a set of specific sequences found in systems employing spread spectrum or code-division multiple access (CDMA) techniques. These systems are often used in communications equipment such as cellular telephones, global positioning systems (GPS), and Very Small Aperture Satellite Terminals (VSATS). Gold codes have cross-correlation properties necessary in a multi-user environment, where one code must be distinguished from several codes existing in the same spectrum. This application note describes the implementation of the Gold Code Generator reference design, based on the 3rd Generation Partnership Project (3GPP) specifications for the WCDMA Universal

Mobile Telecommunications Systems (UMTS) uplink channel. Using the time division multiplexing (TDM) technique, the gold code generator is able to generate 32 unique codes simultaneously and efficiently, using the same resources of a single code generator.

1.1 BACKGROUND OVERVIEW

Gold code generators have applicability for a number of communication systems, in particular, for spread spectrum wireless communications. In spread spectrum transmissions, the circuit artificially spreads the transmitted signals bandwidth by modulating an information signal, either in phase or frequency, with a pseudo-random sequence that occurs at a greater rate than that required for the data alone. During signal reception, the receiver synchronizes an internal pseudo-random generator to the pseudo-random sequence of the transmitted signal to fully recover the available power and decode the message. Most direct sequence spread spectrum systems pseudo-randomly modulate the phase of the RF carrier signal 10 times or greater than the rate required for the data transmission. This results in a signal spectrum which is much broader than would be occupied if the RF carrier signal were modulated by only the data stream. Frequency hopping systems use the Gold code generator to implement frequency hops within the spread spectrum range.

Matching Gold code generators at the transmitter and receiver allow the correlation and recovery of the information signal. Other transmitted signals with different pseudo-random codes can be transmitted in the same bandwidth since the correlation between the different pseudo-random codes is quite low. The transmissions using the different pseudo-random codes will tend not to significantly interfere with one another.

The pseudorandom noise (PN) sequences are a series of 1’s and 0’s which lack any definite pattern, and look statistically independent and uniformly distributed. The sequences are deterministic, but exhibit noise properties similar to randomness. The PN sequence generator is usually made up of shift registers with feedback. By linearly combining elements from taps of the shift register and feeding them back to the

input of the generator, you can obtain a sequence of much longer repeat length using the same number of delay elements in the shift register. Hence, these blocks are also referred to as linear feedback shift registers (LFSR). The length of the shift register, the number of taps, and their positions in the LFSR, are important to generate PN sequences with desirable autocorrelation and cross-correlation properties.
One preferred way of implementing a pseudo-random sequence is to combine the outputs of two linear feedback shift registers. Such a pseudo-random generator is called a gold code generator.

1.2 HISTORY

Gold Code is named after Robert Gold. The Gold Codes are the launch codes for nuclear weapons provided to the Presidents of the United States in their role as Commander-in-Chief of the United States armed forces. A Gold code, also known as Gold sequence, is a type of binary sequence, used in telecommunication (CDMA) and satellite navigation (GPS). It refers to a special set of binary Random (Pseudo Random) sequence in which the correlation among member sequences is very small. Due to this property (small correlation), this is widely used for various wireless communication system as a scrambling code. These codes possess pseudo random properties ensuring both favourable auto and cross correlation characteristics. Such codes are used in direct sequence spread spectrum systems such as UMTS in order to describe channels across the radio interface.

1.3 GENERAL THEORY

1.3.1 Understanding

One of the applications of spread spectrum systems is to provide a means other than frequency division multiple access (FDMA) or time division multiple access (TDMA) of sharing the scarce channel resources. When channel resources are shared using spread spectrum techniques, a method known as code division multiple access (CDMA), all users are permitted to transmit simultaneously using the same band of frequencies. Therefore, code division multiple accessing does not require the time synchronization needed in TDMA nor the many filters required in FDMA. Users are each assigned a different spreading code so that they can be separated in the receiver de-spreading process. A

goal of the spread spectrum designer for a multiple access system is to find a set of spreading codes or waveforms such that as many users as possible can use a band of frequencies with as little mutual interference as possible.

Here, the receiver de-spreading operation is a correlation operation with the spreading code of the desired transmitter. Ideally, a received signal that has been spread using a different spreading code will not be de-spread and will cause minimal interference in the desired signal. The specific amount of interference from a user employing a different spreading code is related to the cross-correlation between the two spreading codes and the power level of the two signals.

Unfortunately, the ideal spreading code would be an infinite sequence of equally likely random binary digits. Hence, the use of an infinite random sequence implies infinite storage in both the transmitter and receiver. This is clearly not possible, so that the periodic pseudo random codes (PN codes) are always employed.

Gold codes are specific PN codes and allow construction of families of 2n - 1 codes from pairs of n-stage shift registers in which all codes have well defined cross-correlation characteristics. In other words, Gold codes are combinations of maximal-length(m) codes, which are by far the most widely used in multiple access systems.

The Gold codes introduced in this thesis were invented in 1964 at the Magnavox Corporation specifically for multiple-access applications of spread spectrum. The first spread spectrum modem to employ Gold codes was MX-170 and the first developmental models were demonstrated in 1964[13]. Relatively large sets of Gold codes exists which have well controlled cross-correlation properties.

Spread spectrum communication techniques by using Gold codes have been recognized as a viable method to gain an advantage in interference environments. Many new military-oriented systems have been initiated and some civil systems have been attempted.

1.3.2 Correlation Property

This is the most fundamental theory when we think about Gold code. As we discussed before, CDMA that allows for the simultaneous operation of many signals at the same carrier frequency is a multiplexing technique in which many carriers, all at essentially the same frequency, send different PN type codes that have the property that

the cross-correlation is low between any pair of the codes. That is, the transmitted signals are of the form

$$S_i(t) = (\sqrt{2p})\, d_i(t)\, G_i(t) \sin(w_o t + \theta)$$

i = 1,….,N.

where

p is transmitted power

di(t) is the i^{th} data signal (±1)

Gi(t) is the i^{th} Gold code

wo is the carrier radian frequency

e is the carrier phase

Further, it is required that

$$\left| \int_0^T G_i(t)\, G_i(t-KTs)\, dt \right| << \int_0^T G_j^2(t) dt$$

$$i \neq j \text{ for arbitrary } k$$

so that the i^{th} channel will not interfere with the j^{th} channel for any time shift between them. This interference problem extends to acquisition, tracking, and data demodulation. The correlation properties of the code sequences used in spread spectrum communications depend on code type, length, chip rate, and even the chip-by-chip structure of the particular code being used. Both autocorrelation and cross-correlation are of interest in communication system design.

1.3.3 Autocorrelation

Autocorrelation, in general, is defined as the integral

$$\theta(r) = \int_{-\infty}^{+\infty} g(t)g(t-r)\, dt$$

which is a measure of the similarity between a signal and a phase-shifted replica of itself. An autocorrelation over all phase shift(t-r) of the signal, where is one-chip intervals.

Autocorrelation is of most interest in choosing code sequences that give the least probability of a false synchronization. In a communications system designed for maximum sensitivity it is no mean task to discriminate between correlation peaks in a poorly chosen code. Therefore the designer should investigate the code he or she uses carefully, even if that code is one of the relatively safe m-sequences. Statements such as our extremely long 127-bit chip code sequence assures noise like properties, which have been observed in the literature on spread spectrum systems, exhibit a lack of investigation.

Here, we introduce a term for the property of code sequences, pairs of sequences, or a sequence and another signal that determines a receiver's ability to recognize the proper point of code synchronization. This property is called the index of discrimination (ID) and denotes the difference in correlation between the fully correlated (perfectly synchronized) code and the minor peak of autocorrelation or of cross-correlations. A particular code will then have separate ID values for autocorrelation and cross-correlation with non coded signals. The Niger the ID value, the better the code.

Code sequence autocorrelation is expressed as the number of agreements minus the number of disagreements when the code or codes are compared chip by chip. The following example shows autocorrelation for all shifts of a three stage shift register generator, generating a seven-chip maximal linear code:

$S = S_1\ S_2\ S_3\ ,\ S_3 + S_2 \rightarrow S_1\ ,\ L(S) \rightarrow R(S)$

Reference sequence : 0111001

Shift	Sequence	Agreements	Disagreements	A-D
0	0111001	7	0	7
1	1011100	3	4	-1
2	0101110	3	4	-1
3	0010111	3	4	-1
4	1001011	3	4	-1
5	1100101	3	4	-1
6	1110010	3	4	-1

Table 1.1 Example shows autocorrelation for all shifts of a three stage shift register generator

Here, the net correlation A-D is -1 for all except zero-shift or synchronous condition and 2n - 1 = 7 for the zero-shift condition. This is typical of all m-sequences. That is, the autocorrelation spectrum for an m-sequences is two valued

N occurs 1 time

-1 occurs N-1 times

In the region between the zero and plus or minus one chip shifts, correlation increases linearly so that the autocorrelation function for an m-sequence is triangular as shown in Figure 2.1.

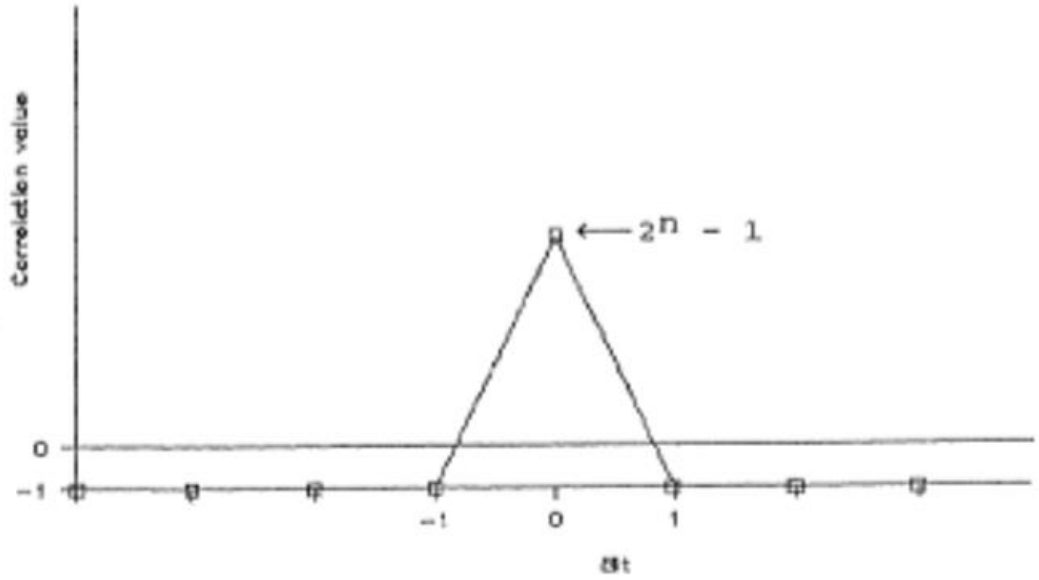


Figure 1.1 M-sequence autocorrelation function

This characteristic autocorrelation is used to great advantage in communication and ranging systems. Two communicators may operate simultaneously, for instance, if their codes are phase-shifted more than one chip. In a ranging system a range measurement is ensured of being accurate within one chip by using the correlation peak as the marker for measurement. This may be accomplished by setting the correlation detector in such a way that it recognizes the level associated with +1-chip synchronization and does not recognize the lower level.

When codes other than in-sequences are used autocorrelation properties may be markedly different from those of the m-sequences. Figure 2.2 illustrates a typical autocorrelation function for a non maximal code. The minor correlation peaks are dependent on the actual code used and are used by partial correlations of the code with a phase-shifted replica of itself. When such minor correlation occur, a receiving

system's ability to synchronize may be impaired because it must discriminate between the major (±l chip) and minor correlation peaks, and the margin of discrimination is reduced.

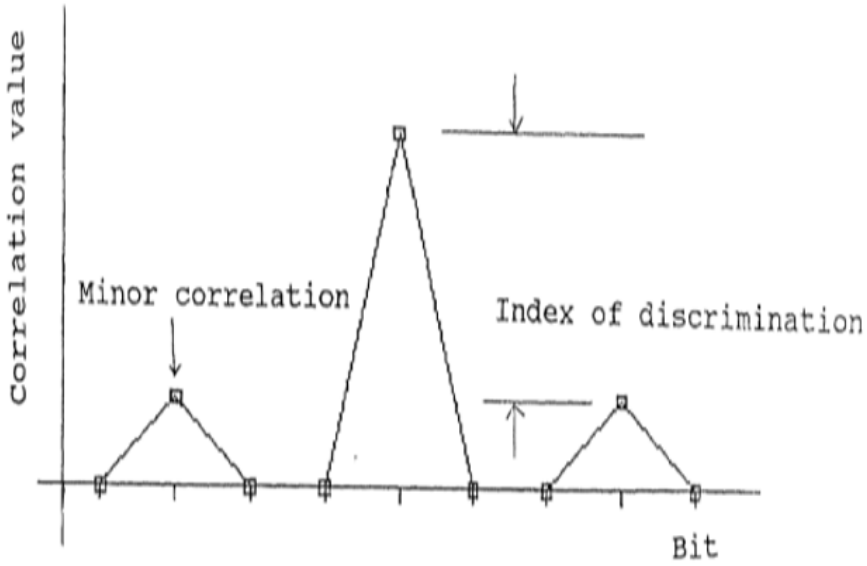


Figure 1.2 Typical non maximal code autocorrelation function

For purposes of illustration, let us consider the five-stage shift register generator (SRG) shown in Figure 2.3.
If feedback is taken from stages five and three, the code sequence output is:
....11111000110111010100001001011 00....(31 chip)

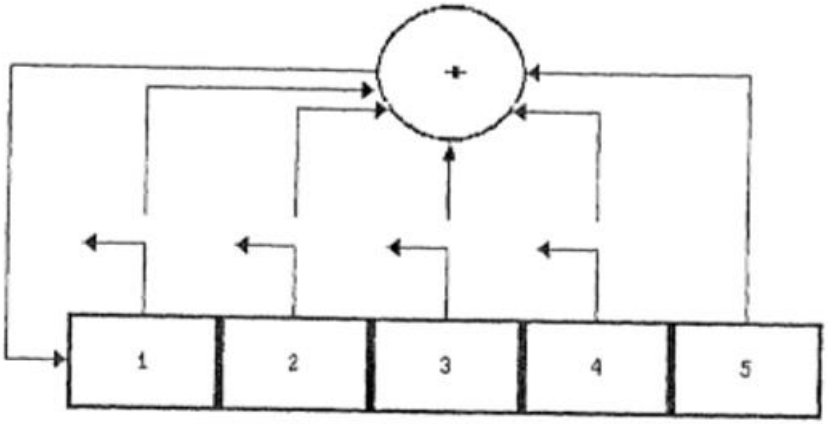


Figure 1.3 Variable tap five-stage SRG

The autocorrelation of this sequence is shown in Figure 2.1; its maximum value is 2n - 1 = 31 and its ID_{auto} = 32. This ID value is, as

expected, typical of all linear maximal sequences (of which this is an example) for which ID_{auto} is always equal to 2n.

Now if we modify the feedback to come from stages five and four, one possible output sequence is only 21 bits long:
...111110000100011001010...(21 chip)

This is an example of a non maximal linear sequence that is less than 2n - 1 chip long. There are two other non maximal linear sequences available from this same feedback configuration whose lengths are seven and three chip:
...1001110... and ...101...

The initial start vector contained in the register determines which of the sequences is generated. For this region greater care is necessary when non maximal sequences are used, both to ensure that the initial start vector is correct (or at least is one of the allowable states) and that noise does not cause the register to go to a state outside the desired set. (In such a case the output code could suddenly change from one sequence to another.)

1.3.3.1 Periodic Autocorrelation

We noticed that m-sequences are certain binary sequences of length $N = 2^n - 1$,where n is the number of shift register stages.

A possible representation of the autocorrelation function is

$$\theta(r) = A - D = N - 2D$$

where A is the number of places that the code a0, ….. , a_{N-1} with period N and the cyclic shift agree, and D is the number of places where they disagree, so that A + D = N.

An m-sequence has the property that a period of the sequence contains 2^{n-1} is and $2^{n-1} - 1$ 0s because there are 2^{n-1} even numbers ending in 1, and $2^{n-1} - 1$ odd numbers in the same range with binary representation ending in 0.

The autocorrelation function for m-sequences is defined by

$$\theta(0) = N \quad r=0$$
$$\theta(r) = -1 \quad 1<r<N-1$$

where the periodic autocorrelation function θ(r) is defined by

$$\theta(r) = \sum_{l=0}^{N-1} a_l \, a_{l+r}$$

For synchronization purposes this periodic autocorrelation function is ideal. In fact it can be shown that this is the best possible autocorrelation

function of any binary sequence of length 2^{n-1} in the sense of minimizing max $\theta(r)$, $0 < r < N$.

1.3.3.2 Aperiodic Autocorrelation

To complete the description of autocorrelation properties of m-sequences, calculation of the aperiodic function is also needed.

The aperiodic autocorrelation depend strongly on the phase of sequences. That is to say, sidelobes and possible multipath interference will depend on the initial starting condition of the shift registers producing the spreading codes. This gives a multitude of codes for any given sequence period N corresponding to the N possible choices of phase for each of the cyclically distinct m-sequences. The result is roughly 106 possible codes of periods 31 to 4095 (there are 474 m-sequences of periods 31 to 4095), and there is a need for some criteria for sorting out the 'best' codes from this large selection.

1.4 Cross-correlation

Cross-correlation is of interest in several area such as CDMA systems(or any code addressed system) in which response of the receiver to any signal other than the proper addressing sequence is not allowable, and anti jamming systems that may employ codes with extremely low cross-correlation as well as unambiguous autocorrelation.

Cross-correlation is the measure of similarity between two different code sequences. The only difference between autocorrelation and cross-correlation is that in the general convolution integral for autocorrelation a different term is substituted

$$\theta_{(cross)} = \int_{-\infty}^{+\infty} f(t)\, q(t-r)\, dt$$

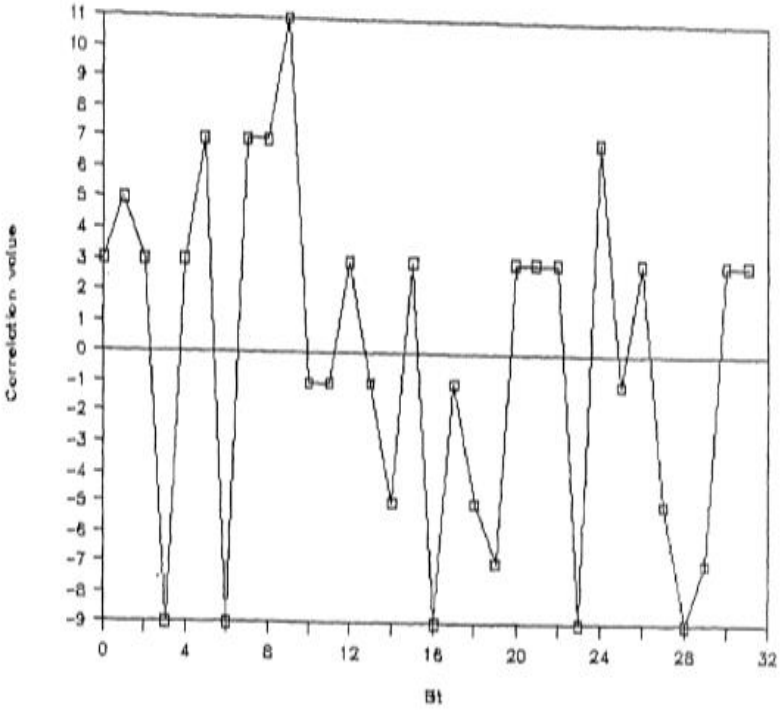
Cross-correlation for different code sequences can be tabulated by generating a comparison table and curve of agreements minus disagreements.

That is, it is expressed as the number of agreements minus the number of disagreements when the code or codes are compared chip by chip the same as for autocorrelation.

But even the linear maximal sequences are not immune to cross-correlation problems, though they are, in general, the best available. It is also of some interest to note, even when the the codes used exhibit excellent crosscorrelation properties when averaged over their entire

length, that short-term crosscorrelations, which are quite effective in disrupting communications, can (and do) occur.

Here, we have restricted our consideration in this section to integration over a long period (- 04 to m). This is essentially (for our simple case) the same as integrating over the code length, for the codes spoken of here repeat at intervals of 2^{n-1} chips. We hasten to point out that integration (as in a synchronization detector) over a period less than that of the code used allows short-term correlations; that is, a short pattern occurring in two different codes or twice in the same code could appear as a legitimate code synchronization when the integration period does not significantly exceed the pattern period.



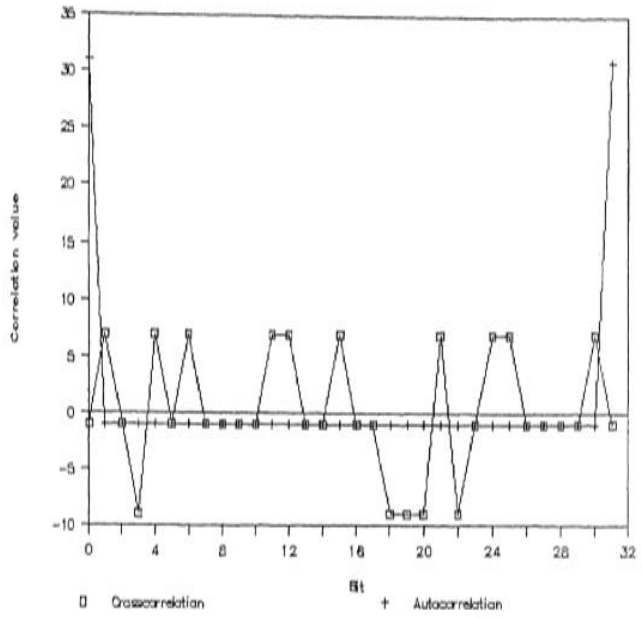
Shift	A	D	A-D	Shift	A	D	A-D
0	17	14	3	16	11	20	-9
1	18	13	5	17	15	16	-1
2	17	14	3	18	13	18	-5
3	11	20	-9	19	12	19	-7
4	17	14	3	20	17	14	3
5	19	12	7	21	17	14	3
6	11	20	-9	22	17	14	3
7	19	12	7	23	11	20	-9
8	19	12	7	24	19	12	7
9	21	10	11	25	15	16	-1
10	15	16	-1	26	17	14	3
11	15	16	-1	27	13	18	-5
12	17	14	3	28	11	20	-9
13	15	16	-1	29	12	19	-7
14	13	18	-5	30	17	14	3
15	17	14	3	31	17	14	3

Figure 1.4 Comparative autocorrelation and cross-correlation for 31 chip mirror image m-sequence

The autocorrelation for curve of the [5,3] code shows a zero-shift correlation value of 31. For the [5,3] and [5,2] codes cross-correlated, however, the peak value is 11. It gives an index of discrimination of 20 and is 37.5% less than the autocorrelation value. The [5,3] and [5,2] codes are images; that is, one is the same as the other, but generated in reverse order. Cross-correlation of the [5,3] and [5,4,3,2] code is lower than that for the image codes, but is still such that the peak cross-

correlation value is seven, a value that occurs at 10 different shift positions.

The significant point is that these particular pairs of code sequences are not capable of operating in the same link if the transmitted power from either transmitter exceeds the other enough to raise the peak cross-correlation to a value near peak autocorrelation. Of course, such short codes should not be used, but the comparison is reasonably representative of operation even with much longer sequences used in code division multiplexing or other multiple access applications



Shift	A	D	A-D	Shift	A	D	A-D
0	15	16	-1	16	15	16	-1
1	19	12	7	17	15	16	-1
2	15	16	-1	18	11	20	-9
3	11	20	-9	19	11	20	-9
4	19	12	7	20	11	20	-9
5	15	16	-1	21	19	12	7
6	19	12	7	22	11	20	-9
7	15	16	-1	23	15	16	-1
8	15	16	-1	24	19	12	7
9	15	16	-1	25	19	12	7
10	15	16	-1	26	15	16	-1
11	19	12	7	27	15	16	-1
12	19	12	7	28	15	16	-1
13	15	16	-1	29	15	16	-1
14	15	16	-1	30	19	12	7
15	19	12	7	31	15	16	-1

Figure 1.5 Comparative autocorrelation and cross-correlation for 31 chip sequence (no image)

1.5 Cross-correlation spectra in Gold codes

The Gold code sequences are of great utility when cross-correlation is a prime consideration. Their real advantages lies in that for every code in a set of 2^n - 1 codes, each of length 2^n - 1, cross-correlation values are well defined, and a system can be designed to operate within this definition.

The Gold code sets to be defined shortly have a cross-correlation spectrum which is three-valued.

The cross-correlation spectrum of pairs of m-sequences can be three-valued, four-valued, or possibly many-valued, where those three values are -t(n), -1, [t(n) - 2]

where

$$t(n) = 1 + 2^{(n+1)} / 2 \quad \text{for n is odd}$$
$$1 + 2^{(n+2)} / 2 \quad \text{for n is even}$$

where the code period $N = 2^n - 1$, are called preferred pairs of m-sequences. Finding preferred pairs of m-sequences is necessary in defining sets of Gold codes. The following conditions are sufficient to define a preferred pair b and b' of m-sequences:

1.6 Algorithm for Gold code selection

Any code can be represented by a polynomial, where the binary codes are represented by a polynomial of the form 1 + AX + BX2 + CX3 +….. + ZXn Here, each coefficient(A, B, Z) is either 0 or 1, each term of the polynomial (except for the first, 1) corresponds to a stage of a binary shift register, and there are n stages in the register. That is, each term in the polynomial containing an X corresponds one-to-one with a stage in a binary shift register. The feedback connections in the code generator are defined by the terms in the polynomial whose coefficient is 1. For instance, a code generator whose characteristic polynomial is 1 + 1X + 1X2 + 1X3 + 1X7 would have seven stages with feedback taken from its first, second, third, and seventh stages. Here, we can express this code as [7, 3, 2, 1] = 1 + X + X2 + X3 + X7 Linear maximal codes, in which we have major interest, have characteristic polynomials that are primitive. That is, the primitive or non-factorable polynomials each define a different linear maximal code. Fortunately, the tables of polynomials mentioned in Appendix F both define primitive polynomials and provide information that allows proper selection of pairs of codes for use in generating Gold codes.

The Gold-derived algorithm for selection of preferred pairs requires the use of code tables that list the polynomial roots. The algorithm is used as follows:
1. Select a polynomial of the proper degree from the table(an n-stage shift register requires an nth degree polynomial).
2. Read the number(k) in the polynomial roots column associated with the polynomial selected.
3. If the code generator has an odd number of stages, then calculate 2^k + 1. If the number of stages is even, calculate $2^{(k+2)/2}$ + 1.
4. The number calculated in step 3 is the polynomial root of a second code that completes a preferred pair.

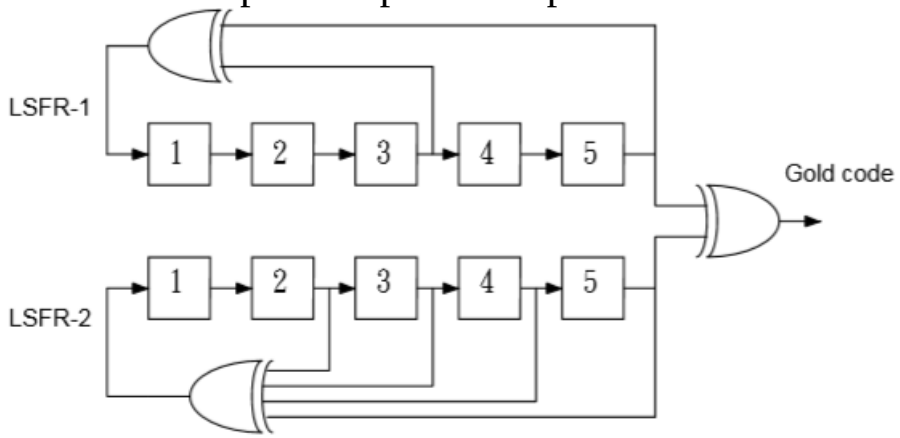


Figure 1.6 The block diagram of the five-stage (5,3) and (5,4,3,2) Gold code generator

Figure 1.4 shows a five-stage Gold code generator consisting of LFSR-1 and LFSR-2. Both LFSR include five latches, or in other word, five shift registers in which data are always shifted from left to right. Number 3 and 5 of LFSR-1 are connected to the XOR gate and then becomes the input of the shift registers. We denote it as (5, 3). Following the above discussion, LFSR-2 in Figure 1.4 can be denoted as (5, 4, 3, 2).

1.7 Generation of general Gold codes

Gold code sequence generators are useful because of the large number of codes they supply, although they require only one pair of feedback tap sets. A bonus awarded on the basis of the use of these codes is that only a few sets of feedback connection are required for each simple shift register generator (SSRG) while retaining the capacity to generate a large number of codes. The single-tap SSRG is the fastest configuration possible. Thus the Gold code sequences are potentially available at rates

equal to the capacity of the fastest SSRG. The Gold codes are generated by modulo-2 addition of a pair of maximal linear sequences as shown in Figure 3.1. The code sequences are added chip-by-chip by synchronous clocking. Here, code 1 and code 2 are the same length. Thus the two code generators maintain the same phase 48
relationship, and the codes generated are the same length as the two base codes, which are added together but are non maximal. A specific example is shown in Figure 3.2.

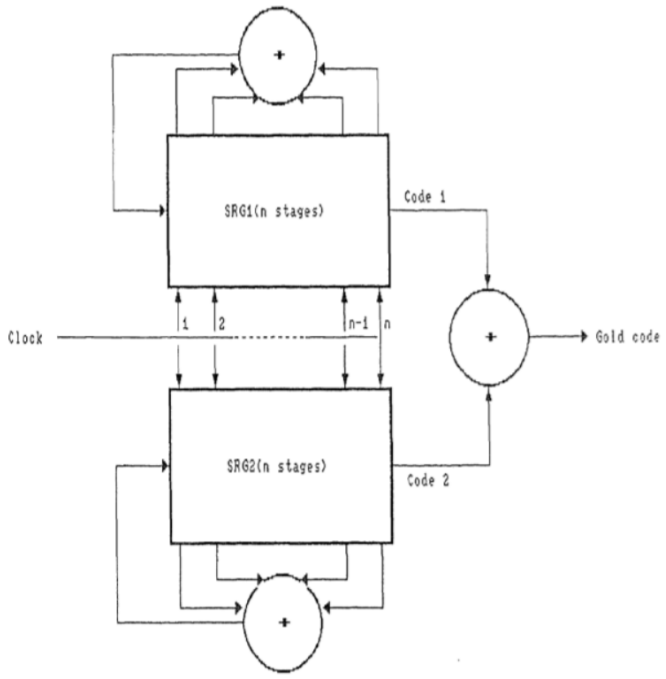


Figure 1.7 Configuration of Gold code generator

The shift and add property of maximal sequences tells us that any maximal sequence added to a phase-shifted replica of itself(any integral number of bits) produces a different phase shift as an output. Here the same operation is performed, with the new sequence having the same length as those being added, and non maximal. Furthermore, every change in phase position between the two generators causes a new sequence to be generated. To see this advantage, consider the following example. Given a five-stage sequence generator, we choose a set of feedback taps, [5,3] and [5,4,2,1] from the reference table in Appendix F. Here, there are only six feedback sets available for the five-stage register and half are images of the other half. If more than six 31-chip codes are needed, we cannot get them from our five-stage register. Therefore, we use two five-stage sequence generators connected in the

Gold code sequence generator configuration, as shown in Figure 3.2. Table 3.3 also shows the modulo - 2 -combined Gold codes produced by combining the two output maximal codes with different initial offsets; that is, the two code generators are started with initial conditions offset by various amounts to give different output codes. The all-ones vector is set into both registers as an initial condition. In addition one, three, and five-chip shifts(from all-ones vector) are also shown in initial conditions:

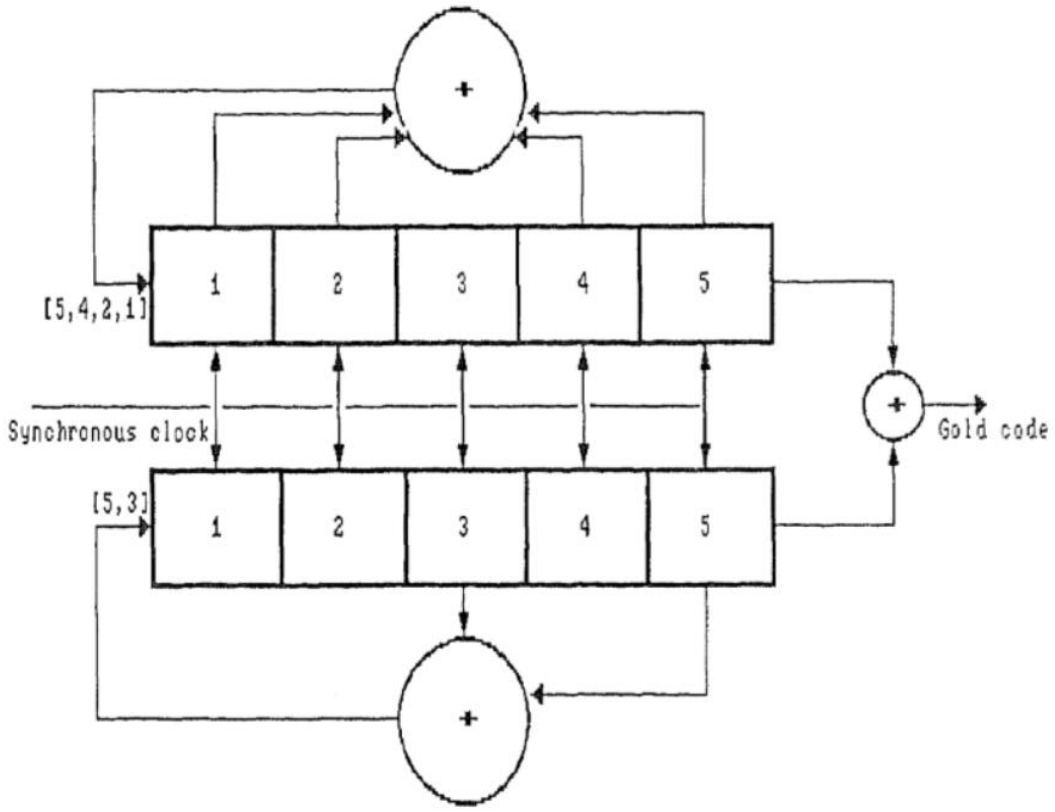


Figure 1.8 Gold code generation

[5,4,2,1]	11111011001110000110101001000 10
[5,3]	11111000110111010100001001011 00
Zero-shift combination	00000011111001010010100000011 10
One-chip-shift combination	00001010100000101110111011111 011
Three-chip-shift combination	00111101110100100111100010001 01
Five-chip-shift combination	11100000100100000100001011110 1

Figure 1.9 Modulo-2-combined Gold code

Any shift in initial conditions from 0 to 30 chips can be used (A 31-chip shift is the same as the zero shift). Thus, from this Gold sequence generator, 33 maximal-length codes are available. Extending this demonstration, we can show that any two-register Gold code generator of length n can generate 2n - 1 m-sequences(length 2n - 1) plus the two maximal base sequences. A multiple-register Gold code generator can generate (2n - 1)r non maximal sequences of length 2n - 1 plus r m-sequences of the same length where r is the number of registers and n is register length. In addition to their advantage in generating large numbers of codes, the Gold codes may be chosen so that over a set of codes available from a given generator the cross correlation between the codes is uniform and bounded. Thus the Gold codes are attractive for applications in which a number of code-division-multiplexed signals are to be used. The same guarantee of bounded cross correlation is impossible for m-sequences of the same length.

CHAPTER-2
LITERATURE-REVIEW

2.1 Wireless Communication Principles and Practice
-Theodore S Rappaport

A source which provided us with the basic concepts involved in Code Division Multiple Access technique which is an area of application of Gold Codes.

In Code Division Multiple Access systems, a narrowband message signal is multiplied by a very large bandwidth signal called the spreading signal. The spreading signal is a pseudo-noise code sequence that has a chip rate which is orders of magnitudes greater than the data rate of the message. All users in CDMA system use the same carrier frequency and may transmit simultaneously. Each user has his own pseudorandom codeword which is approximately orthogonal to all other codewords. The receiver performs a time correlation operation to detect only the specific desired codeword. All other codewords appear as noise due to decorrelation. For detection of the message signal, the receiver needs to know the codeword used by the transmitter. Each user operates independently with no knowledge of the other users.
The features of CDMA include the following:

- Many users of a CDMA system share the same frequency.
- Channel data rates are very high in CDMA systems. Consequently, the symbol (chip) duration is very short and usually much less than the channel delay spread. Since PN sequences have low autocorrelation, multipath which is delayed by more than a chip will appear as noise.

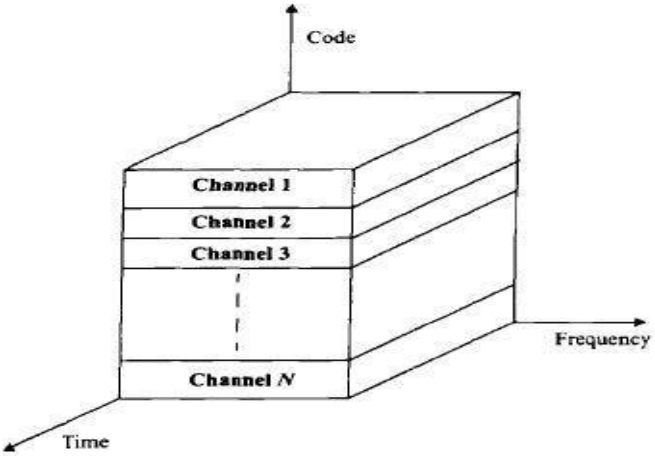


Figure 2.1 The pseudorandom code used can be the Gold Code.

2.2 Design of Pseudo Random Number Generator using Linear Feedback Shift Register

-Shabbir Hassan, M. U. Bokhari

A source which provided us with reference on how to generate a pseudo random code.

In order to secure data in a communication channel we use several encryption algorithms, keys and mathematical tools. All these attributes can be easily implemented using mathematical concepts that were dealt with in this paper. Due to the simple structure and implementation of LFSR, it is widely used in network communication and industries for generating pseudo random sequences. LFSR has the widest implementation in networking and cryptography. Generally, LFSRs are constructed by D Flip-Flop and two input XOR gates. It can be implemented in two ways, first is Fibonacci implementation and other is D Flip-Flop implementation.

A Mealy machine, Autonomous Linear Feedback Shift Registers (LFSR), Pseudo Random number Generator, Polynomial Sequence Generator, Pseudo Random Pattern Generator or simply an LFSR comprises of two parts: (i) clocked storage elements (Flip-Flop) and (ii) a feedback path. The number of storage elements gives us the degree of the LFSR. In other words, an LFSR with m Flip-Flops is said to be of degree m. The possible feedback paths compute the input for the left most as XOR or XNOR sum of certain Flip-Flop in the shift register. The internal value of LFSR is called initial fill, initial vector or

a seed (in mathematical terminology) and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current or previous states. The output of the LFSR is one bit at each clock. Likewise, the register has a finite number of possible states, it must eventually enter a repeating cycle excluding all zeroes pattern, i.e., once it reaches its final state; it will traverse the sequence exactly as before. However, an LFSR with a well-chosen feedback function can produce a sequence of bits which appears random and has a very long cycle.

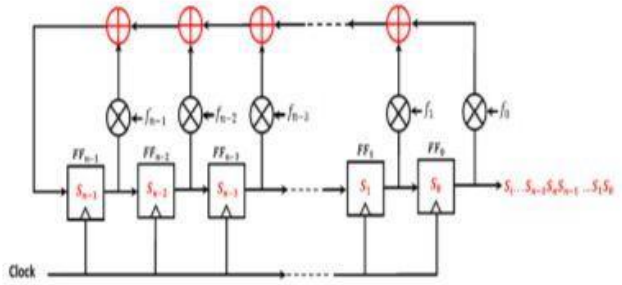


Figure 2.2 LFSR

2.3 Gold Codes lecture Fall 2014

-Prof David Money Harris

A source which provided all necessary details about Gold Codes.

Gold codes are a pseudorandom sequence of 0's and 1's that are commonly used in communication systems and devices like GPS and cell phones.

They are important in applications like CDMA since they are easy to generate in hardware and software and minimally jam other Gold codes generated by other sources. They have properties that resemble that of random noise.

Communication systems need a set of bit sequences that are easy to generate on hardware and software and have a low cross correlation with other sequences in the set. This makes it easier to tell them apart even when they are corrupted by noise. Gold codes are such a class of 2^N-1 sequences of length 2^N-1.

To define a Gold code: i) Decide the characteristic polynomial for both the LFSR's; ii)State the seed for second LFSR; iii) Always choose a seed of the form 0000….1 for the first LFSR.

In CDMA, all phones transmit on all frequencies simultaneously and each phone has it's own 15-bit Gold Code.

2.4 ALTERA Gold Code Generator Reference Design

- Carnegie Mellon School of Computer Science

A source which gave us an idea on a reference design upon which we could brainstorm and work on a design of our own.

The scrambling codes used in 3G CDMA wireless systems are based on Gold codes. Gold codes are obtained by combining two PN sequences and modulo-2 adding, or XORing, the output together. These codes have specific cross-correlation properties, to allow as many users as possible, with minimum interference. Using a set of polynomials, you can construct the PN sequences (also known as m-sequences). The reference design used in this paper utilizes a set of specific primitive polynomials over Galois Field 2 as described in the 3rd Generation Partnership Project (3GPP) Technical Specification 25.213.

The first sequence uses the following polynomial:

$X^{25} + X^3 + 1$

The second sequence uses the following polynomial:

$X^{25} + X^3 + X^2 + X + 1$

The use of 25-stage LFSRs was required by the specifications for this design.

The final result is a long scrambling code, $C_{long,1n}$ and $C_{long,2n}$, generated by summing (using modulo-two addition) the outputs of two PN code sequence generators.

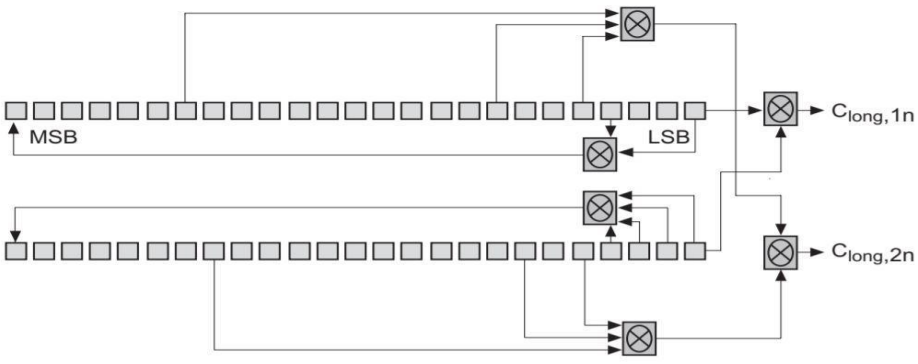


Figure 2.3 Gold Code Generator

CHAPTER-3
SOFTWARE DESCRIPTION

3.1 XILINX ISE

Xilinx ISE (Integrated Synthesis Environment) is a software tool produced by Xilinx for synthesis and analysis of HDL designs, enabling the developer to synthesize ("compile") their designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer. Xilinx ISE is a design environment for FPGA products from Xilinx, and is tightly-coupled to the architecture of such chips, and cannot be used with FPGA products from other vendors. The Xilinx ISE is primarily used for circuit synthesis and design, while ISIM or the ModelSim logic simulator is used for system-level testing. Other components shipped with the Xilinx ISE include the Embedded Development Kit (EDK), a Software Development Kit (SDK) and ChipScope Pro.

3.2 OVERVIEW OF XILINX

Xilinx, Inc. is an American technology company that develops highly flexible and adaptive processing platforms. The company invented the field-programmable gate array (FPGA), programmable system-on-chips (SoCs), and the adaptive compute acceleration platform (ACAP). It is the semiconductor company that created the first fabless manufacturing model.[4][5] Xilinx's products are used across many industries and technologies, including the data center, wired & wireless communications, AI/ML, automotive, industrial, consumer, aerospace and defense and Broadcast & Pro-AV.

Co-founded by Ross Freeman, Bernard Vonderschmitt, and James V Barnett II in 1984, the company went public on the NASDAQ in 1989. Xilinx's FPGAs have been used for the ALICE (A Large Ion Collider Experiment) at the CERN European laboratory on the French-Swiss border to map and disentangle the trajectories of thousands of subatomic particles. Xilinx FPGAs can run a regular embedded OS (such as Linux or vxWorks) and can implement processor peripherals in programmable logic. The Virtex-II Pro, Virtex-4, Virtex-5, and Virtex-6 FPGA families, which include up to two embedded IBM PowerPC cores, are targeted to the needs of system-on-chip (SoC) designers.

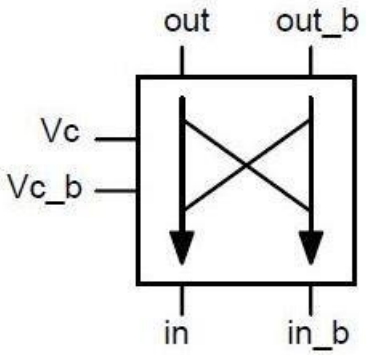
3.3 DESIGN

- We start the design by breaking the entire project into smaller parts in a bottom-up fashion.
- The basic building block of this project is the differential switch which is constructed using four NMOS transistors.
- A two input XOR is constructed by connecting two of these differential switches in cascade.
- Similarly, an n-input XOR is constructed by connecting n differential switches in cascade.
- An LFSR is constructed by making use of a number of shift registers that are connected in cascade and consisting of a feedback that depends on the characteristic equation. What outputs are to be fed back depends on the characteristic equation. The outputs will be XORed using the n-input XOR that we constructed above.
- The Gold code is generated by finally XORing (2-input XOR) the outputs of two LFSR's is constructed in the above manner.

We shall discuss each of these units in detail:

3.3.1 Differential Switch

It is organized to have two-input (in, in_b), two-output (out, out_b), and two control voltage (Vc, Vc_b) terminals.



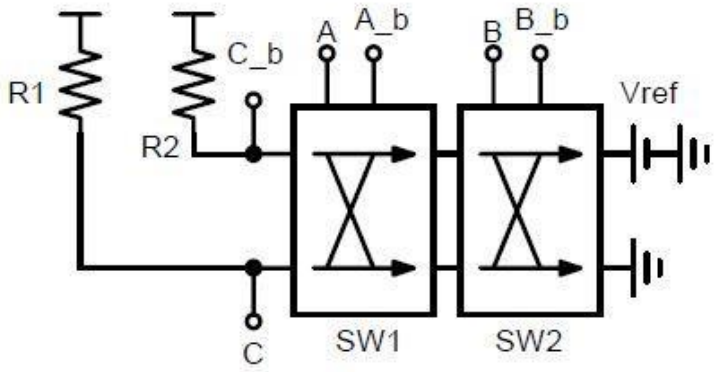
When the control voltage is "low", or in other words, (low, high) or (0,1) voltage signals are applied to the (Vc, Vc_b) terminals, the proposed switch will stay in OFF status.

Hence, the input terminals are connected straight to the output

terminals without exchange in this situation.
On the other hand, if the control voltage is “high”, or in other words, we apply (high, low) or (1,0) voltage signals to the (Vc, Vc_b) terminals, then the proposed switch is transferred to the exchange position.
The “in” terminal is connected to “out_b” terminal while “in_b” is connected to “out”.
For better understanding, we can remember that the signal connected at the input terminal of the switch will be bypassed (without exchange) to the output terminal if the control voltage is 0 (Vc = 0, Vc_b = 1). Reversely the output signal will be exchanged if the control voltage is 1 (Vc = 1, Vc_b =0).

Two-input XOR

In order to create a two-input XOR gate, two differential switches, two resistors, and one reference voltage source should be connected (A, A_b) and (B, B_b) denote two complementary inputs of the XOR gate, and (C, C_b) denotes the complementary output of it.

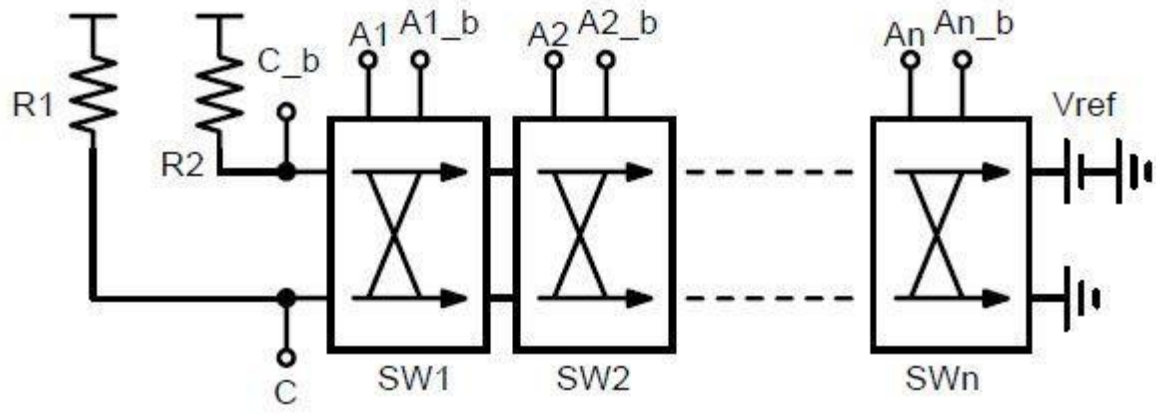


In case of A = 0 and B = 0 (A_b = 1 and B_b = 1), both switches are in OFF status, and two resistors are connected directly to the voltage source and the ground respectively without any exchange. Hence, the output will be C = 0 (ground) and C_b = 1 (Vref).
Alternatively, in case of A = 1 and B = 0, it is comprehensive that SW1 is turned ON (exchange position) and SW2 still stays at OFF. Then the connection being exchanged in SW1 makes R1 be connected to Vref and R2 to the ground. The result of this connection is C = 1 and C_b = 0.
The rest we can come to is that this functions incredibly similar to a two input XOR. We can observe the table below for reference.

A	B	SW1	SW2	C	C_b
0	0	OFF	OFF	0	1
1	0	ON	OFF	1	0
0	1	OFF	ON	1	0
1	1	ON	ON	0	1

N-input XOR

XOR is a slow gate, especially when the number of input terminals is increased. This delay always prevents the logic with XOR feedback from the very high-speed applications. We propose a solution for the high-speed n-input XOR gate.



It consists of n blocks of the above differential switches connected in series. Ai and Ai_b denote the i-th complementary inputs. C and C_b still denote the outputs of the gate.

To help us understand, let $X_i = -1$ denotes "exchange" while $X_i = 1$ denotes "direct connection" of each switch. The equilibrium exchange X_{total} of the series of switches can be expressed as

$$X_{total} = X_1.X_2.X_3.\ldots\ldots.X_n$$

It is comprehensive that even-number of exchanges of the switches has the same effect as direct connection because the even order of (-1) is equal to 1. Only odd number of exchanges can make the equilibrium connection be exchanged.

Hence, terminal C will be connected to the ground if the equilibrium connection is direct and the result is C=0. Reversely if the equilibrium connection is exchanged then C is connected to Vref and the output of the gate is C=1. This result satisfies the need of an n-input XOR gate

because the XOR of even number of 1s is 0, and the XOR of odd number of 1s is 1.

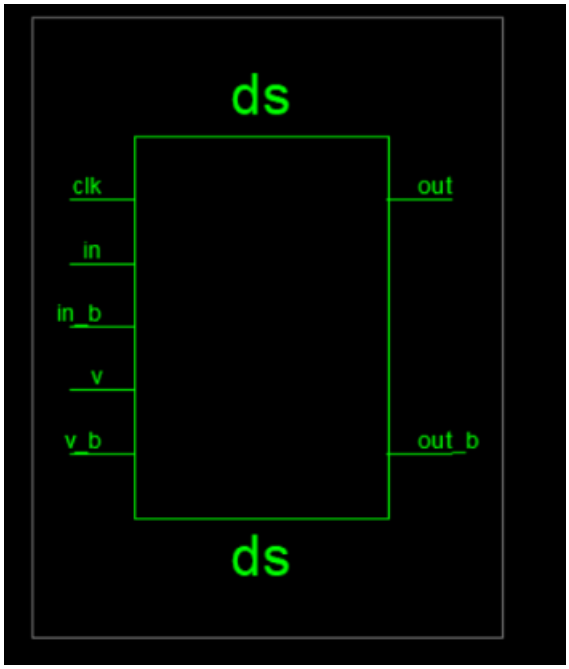
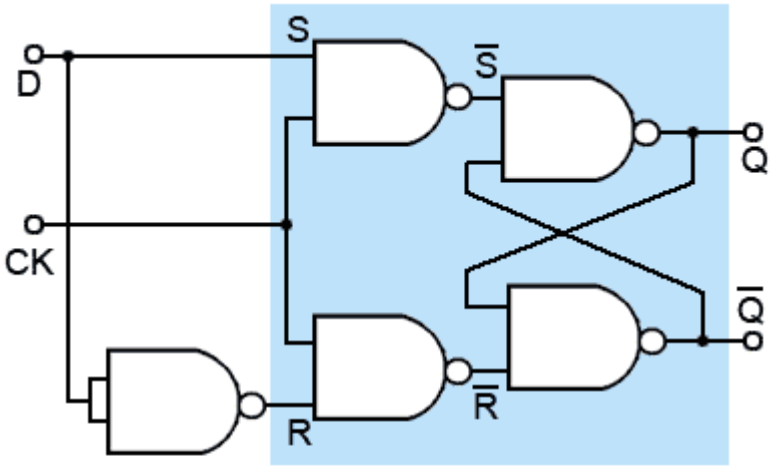


Figure 3.1 Schematic of Differential Switch

3.3.2 D Flip flop

The D flip-flop tracks the input, making transitions with match those of the input D. The D stands for "data"; this flip-flop stores the value that is on the data line. It can be thought of as a basic memory cell. A D flip-flop can be made from a set/reset flip-flop by tying the set to the reset through an inverter. The result may be clocked.



Inputs		Outputs	
CK	D	Q	$\overline{Q}$
0	X	No change	
1	0	0	1
1	1	1	0

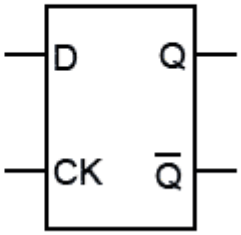


Figure 3.2 Level Triggered D Type Flip-flop

D Type Flip-flop

The major drawback of the SR flip-flop (i.e. its indeterminate output and non-allowed logic states) described in Digital Electronics is overcome by the D type flip-flop. This flip-flop, shown in Fig. 3.2 together with its truth table and a typical schematic circuit symbol, may be called a Data flip-flop because of its ability to 'latch' and remember data, or a Delay flip-flop because latching and remembering data can be used to create a delay in the progress of that data through a circuit. To avoid the ambiguity in the title therefore, it is usually known simply as the D Type. The simplest form of D Type flip-flop is basically a high activated SR type with an additional inverter to ensure that the S and R inputs cannot both be high or both low at the same time. This simple modification prevents both the indeterminate and non-allowed states of the SR flip-flop. The S and R inputs are now replaced by a single D input, and all D type flip-flops have a clock input.

Operation

As long as the clock input is low, changes at the D input make no difference to the outputs. The truth table in Fig. 3.2 shows this as a 'don't care' state (X). The basic D Type flip-flop shown in Fig. 5.3.1 is called a level triggered D Type flip-flop because whether the D input is active or not depends on the logic level of the clock input.

Provided that the CK input is high (at logic 1), then whichever logic state is at D will appear at output Q and (unlike the SR flip-flops) Q is always the inverse of Q).

In Fig. 3.2, if D = 1, then S must be 1 and R must be 0, therefore Q is SET to 1.

Alternatively, If D = 0 then R must be 1 and S must be 0, causing Q to be reset to 0.

The Data Latch

The name Data Latch refers to a D Type flip-flop that is level triggered, as the data (1 or 0) appearing at D can be held or 'latched' at any time whilst the CK input is at a high level (logic 1).As can be seen from the timing diagram shown in Fig 3.3, if the data at D changes during this time, the Q output assumes the same logic level as the D.

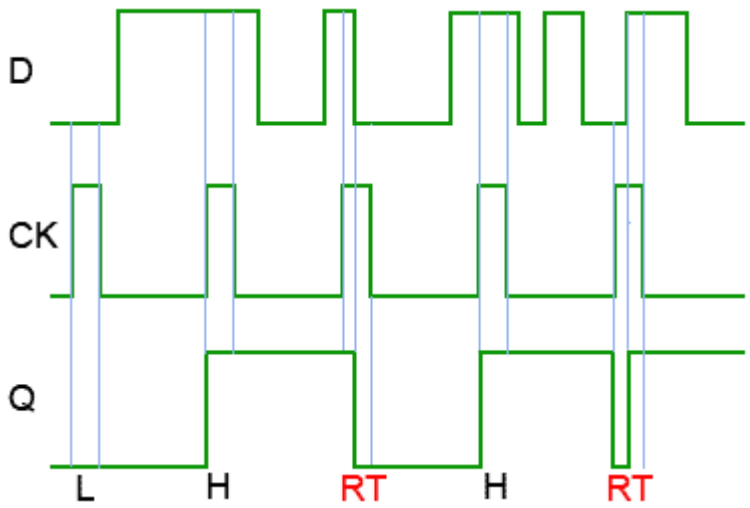


Figure 3.3 Timing Diagram for a Level Triggered D Type Flip-flop

The Edge Triggered D Type Flip-flop

Fortunately ripple though can be largely prevented by using the Edge Triggered D Type flip-flop illustrated in Fig 3.3.

The clock pulse applied to the flip-flop is reduced to a very narrow positive going clock pulse of only about 45ns duration, by using an AND gate and applying the clock pulse directly to input ‘a’ but delaying its arrival at input ‘b’ by passing it through 3 inverters. This inverts the pulse and also delays it by three propagation delays, (about 15ns per inverter gate for 74HC series gates). The AND gate therefore produces logic 1 at its output only for the 45ns when both ‘a’ and ‘b’ are at logic 1 after the rising edge of the clock pulse.

Synchronous and Asynchronous Inputs

A further refinement in Figure 3.4 is the addition of two further inputs SET and RESET, which are actually the original S and R inputs of the basic low activated SR flip-flop.

Notice that there is now a subtle difference between the active low Set (S) and Reset (R) inputs, and the D input. The D input is SYNCHRONOUS, that is its action is synchronised with the clock, but the S and R inputs are ASYNCHRONOUS i.e. their action is NOT synchronised with the clock. The SET and RESET inputs in Fig 5.3.4 are ‘low activated’, which is shown by the inversion circles at the S and R inputs to indicate that they are really S and R.

The flip-flop is positive edge triggered, which is shown on the CK input in Fig 5.3.4 by the wedge symbol. A wedge accompanied by an inversion circle would indicate negative (falling) edge triggering, though this is generally not used on D Type flip-flops.

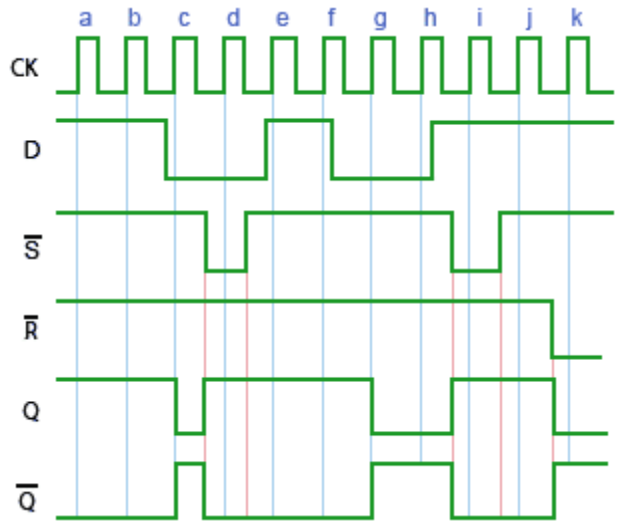


Figure 3.4 Typical Schematic Symbols for D Type Edge Triggered Flip-Flops

Timing Diagram

The 'Edge triggered D type flip-flop with asynchronous preset and clear capability', although developed from the basic SR flip-flop becomes a very versatile flip-flop with many uses. A timing diagram illustrating the action of a positive edge triggered device is shown in Fig. 5.3.5.

At the positive going edges of clock pulses a and b, the D input is high so Q is also high.

Just before pulse c the D input goes low, so at the positive going edge of pulse c, Q goes low.

Between pulses c and d the asynchronous S input goes low and immediately sets Q high.

The flip-flop then ignores pulse d while S is low, but as S returns high, and D has also returned to its high state before pulse e, Q remains high during pulse e.

At the positive going edge of pulse h, the low level of input D remains, keeping Q low, but between pulses h and i, the S input goes low, overriding any action of D and immediately making Q high.

D is still high at the positive going edge of pulse f, and because the flip-flop is positive edge triggered, the change in the logic level of D during

pulse f is ignored until the positive going edge of pulse g, which resets Q to its low level.

Clock pulse i is again ignored, due to S being in its active low state and Q remains high, under the control of S until just before pulse j. At the positive going edge of pulse j, input D regains control, but as D is high and Q is already high, no change in output Q occurs.

Finally, just before pulse k, the asynchronous reset input (R) goes low and resets Q to its low level (logic 0), which again causes the D input to be ignored.

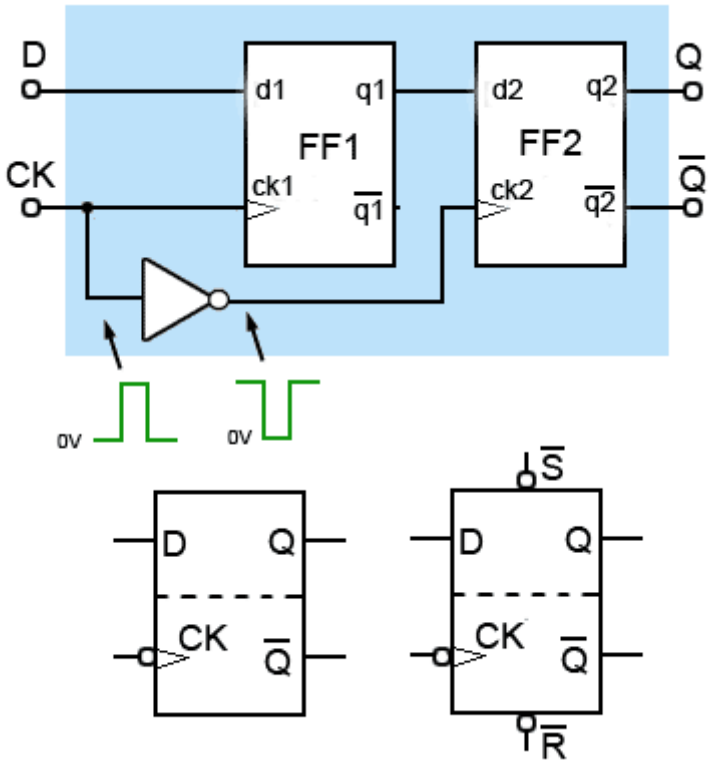


Figure 3.5 The D Type Master Slave Flip-flop

The D Type Master Slave Flip-Flop

Yet a further version of the D Type flip-flop is shown in Fig. 5.3.6 where two D type flip-flops are incorporated in a single device, this is the D type master-slave flip-flop. Circuit symbols for the master-slave device are very similar to those for edge triggered flip-flops, but are now divided into two sections by a dotted line, as also illustrated in Fig 5.3.6.

FF1 (the master flip-flop) is a positive edge triggered device, and an inverted version of the CK pulse is fed from the main CK input to FF2 (the slave), also positive edge triggered. Notice that although the clock inputs on the circuit symbols suggest that this is a negative edge triggered device, data is actually taken into FF1 on the POSITIVE going edge of the CK pulse. The data also of course appears at q1 at this time, but as the CK pulse is inverted at ck2, FF2 is seeing a falling edge at the same time, so ignores the data on d2.

After the positive going edge of the external CK pulse, FF1 ignores any further data at D, and at the negative going edge of the external CK pulse, the data being held at q1 is taken into the d2 input of FF2 which now sees a positive going edge of the inverted CK pulse. Therefore data is taken into D at the positive going (rising) edge of the CK pulse, and then appears at Q at the negative going (falling) edge of the CK pulse.

Considering the master slave flip-flop as a single device, the relationship between the clock (CK) input and the Q output does look rather like a negative edge triggered device, as any change in the output occurs at the falling edge of the clock pulse. However, as illustrated in Figure 3.6 this is not really negative edge triggering, because the data appearing at Q as the clock pulse returns to logic 0, is actually the data that was present at input D at the RISING edge of the CK pulse. Any further changes that may occur in data at the D input during the clock pulse are ignored. D type master-slave flip-flops are also available with asynchronous S and R inputs making it a very versatile device indeed.

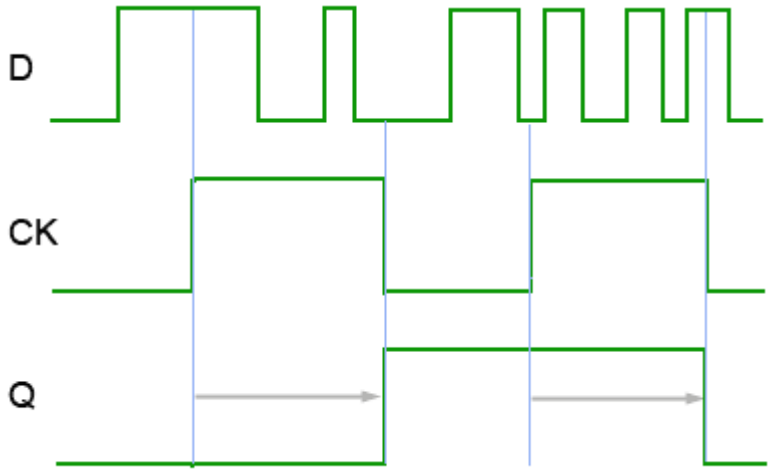


Figure 3.6 Timing Diagram for a D Type Master-Slave Flip-flop

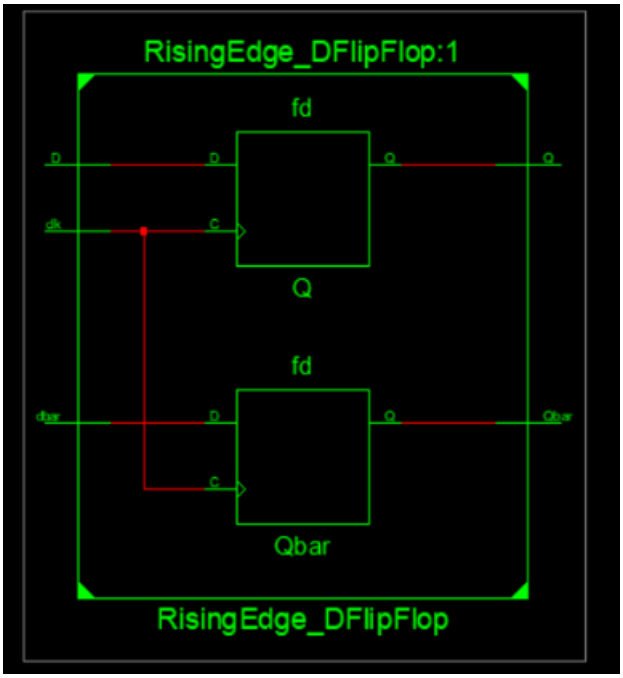
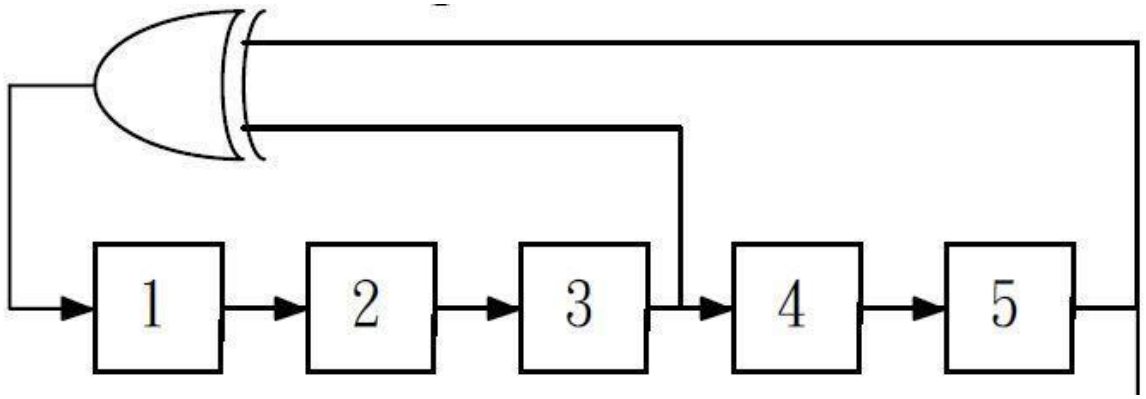


Figure 3.7 Schematic of D flip flop

3.3.3 Linear Feedback Shift Registers

An LFSR is constructed by connecting a number of shift registers in cascade and the outputs of each shift register will be provided/not provided as feedback depending on a characteristic equation.
A shift register is a unit which will hold a value and simply shift contents from its input to output depending on a clock signal.

The shift register is nothing but a D-Flipflop which is constructed using NAND gates as shown below:
A number of such D-Flipflops will be connected in cascade and feedbacks will be tapped based on characteristic equation resulting in the following structure.



Where 1,2,3,4,5 represent the D-Flipflops as discussed above and the XOR gate is the n-input XOR that we developed earlier.

Figure 3.8 The structure shown above is a (5,3) LFSR.

Taps And Seeds In LFSR

- Bits fed back are called the taps.
- LFSR taps are described by a characteristic polynomial.
- Ex:$1+x^3+x^5$
- In above example Taps in columns 3 and 5.
- 1 is not a tap but corresponds to the input of the first bit x0.
- The initial contents of the LFSR are called the seeds.
- EX:00001

Shift Register Sequence

- A shift register sequence is the pattern in the msb
- Sequence: 10000100101100111110001101110 10

Step	Q1	Q2	Q3	Q4	Q5	Step	Q1	Q2	Q3	Q4	Q5
0	0	0	0	0	1	16	0	0	1	1	1
1	1	0	0	0	0	17	0	0	0	1	1
2	0	1	0	0	0	18	1	0	0	0	1
3	0	0	1	0	0	19	1	1	0	0	0
4	1	0	0	1	0	20	0	1	1	0	0
5	0	1	0	0	1	21	1	0	1	1	0
6	1	0	1	0	0	22	1	1	0	1	1
7	1	1	0	1	0	23	1	1	1	0	1
8	0	1	1	0	1	24	0	1	1	1	0
9	0	0	1	1	0	25	1	0	1	1	1
10	1	0	0	1	1	26	0	1	0	1	1
11	1	1	0	0	1	27	1	0	1	0	1
12	1	1	1	0	0	28	0	1	0	1	0
13	1	1	1	1	0	29	0	0	1	0	1
14	1	1	1	1	1	30	0	0	0	1	0
15	0	1	1	1	1						

Table 3.1 Shift register sequence table

Seeding

- Different seeds give shifted version of the sequence
- Seed 00010:Sequence 010000100101100111110001101110 1

Step	Q1	Q2	Q3	Q4	Q5	Step	Q1	Q2	Q3	Q4	Q5	
0	0	0	0	0	1	16	0	0	1	1	1	
1	1	0	0	0	0	17	0	0	0	1	1	
2	0	1	0	0	0	18	1	0	0	0	1	
3	0	0	1	0	0	19	1	1	0	0	0	
4	1	0	0	1	0	20	0	1	1	0	0	
5	0	1	0	0	1	21	1	0	1	1	0	
6	1	0	1	0	0	22	1	1	0	1	1	
7	1	1	0	1	0	23	1	1	1	0	1	
8	0	1	1	0	1	24	0	1	1	1	0	
9	0	0	1	1	0	25	1	0	1	1	1	
10	1	0	0	1	1	26	0	1	0	1	1	
11	1	1	0	0	1	27	1	0	1	0	1	
12	1	1	1	0	0	28	0	1	0	1	0	
13	1	1	1	1	0	29	0	0	1	0	1	
14	1	1	1	1	1	30	0	0	0	1	0	←Seed
15	0	1	1	1	1							

Table 3.2 Seeding table

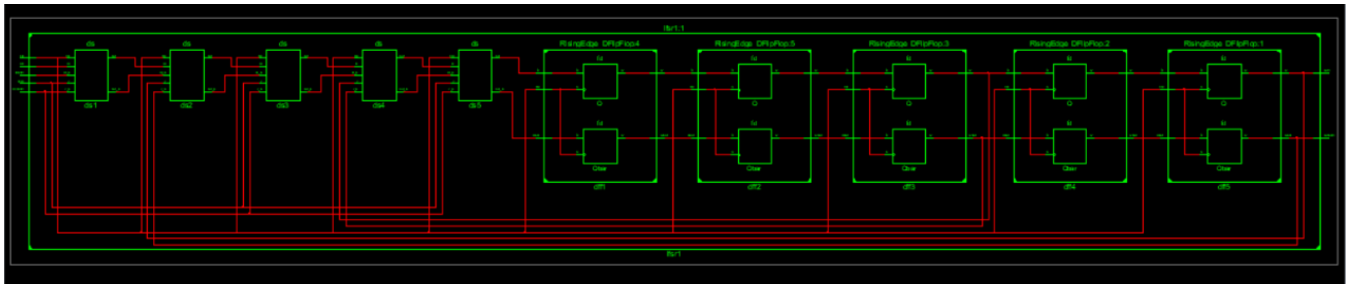


Figure 3.9 Schematic of Linear feedback shift Register(LFSR)

3.3.4 Gold Code Generator

A Gold Code generator is constructed using two equal length LFSRs which usually have different characteristic equations. The outputs of these two LFSRs are then XORed together using a two input XOR.
The whole purpose of using XORs in the project is to generate a code which seems totally random in nature like noise.
The selection of proper characteristic equations is essential to ensure that the Gold Code Generator functions up to our expectations.
The final structure looks like:

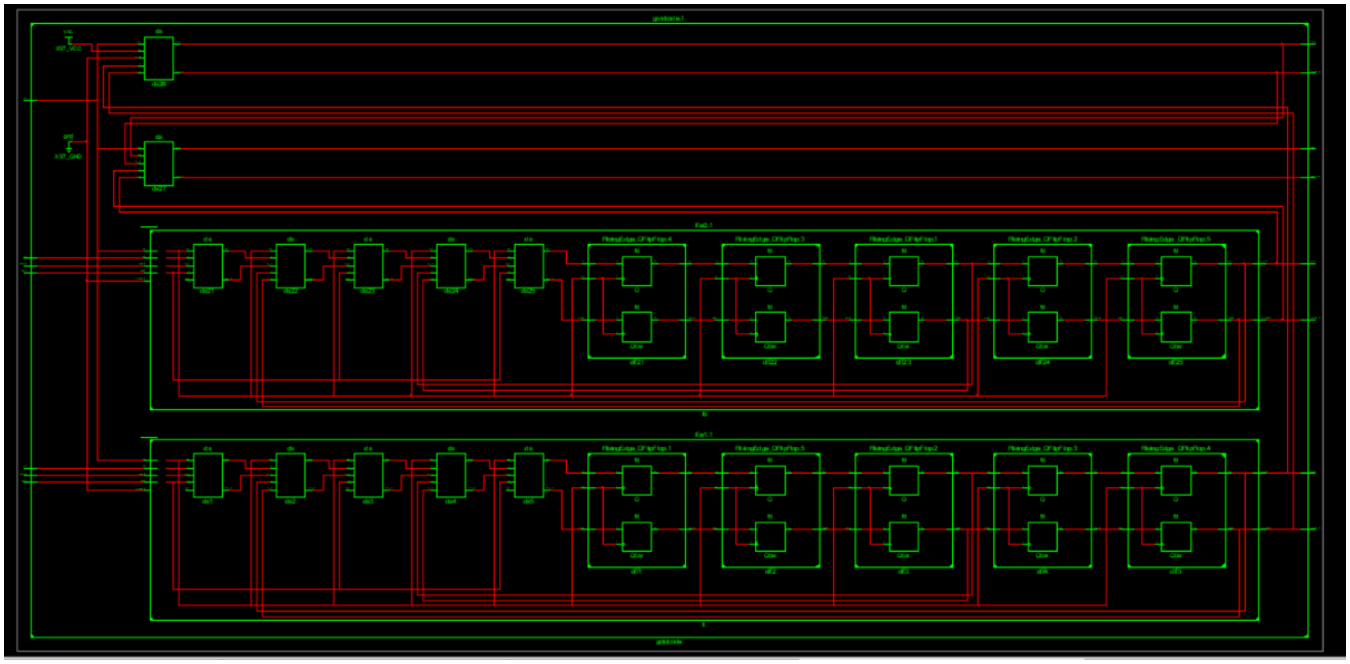


Figure 4.0 Schematic of Gold code Generator

CHAPTER-4
RESULTS

The simulation tool used in this work is ADS of Agilent. Besides typical-typical corner case, fast-fast and slow-slow cases are both checked. The generator in fast-fast case can work over 10 GHz while in slow-slow case can work only below 9.5 GHz. The implemented Gold code generator is also checked at lower frequency such as 1 GHz, 1 MHz, and even 1 KHz. The simulation results show that the generator works well at 1 GHz and 1 MHz, but not at 1 KHz. Then it works again after the improvement of the rising and falling time. These results show that the implemented generator has good stability. They also prove the proposed architecture of multi-input XOR gate is fast enough in ten-gigabit applications.

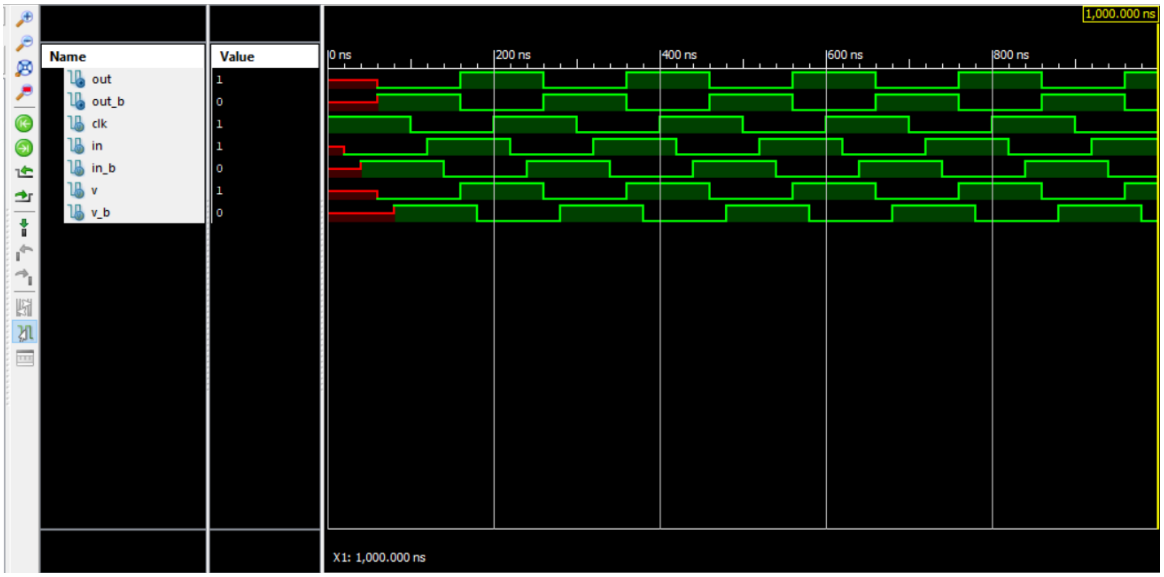


Figure 4.1 Output Waveforms of Differential Switch

The above waveforms show the operation of two Differential Switches operating as a Xor gate. The peak to peak value of five switches is found to be lower than a single switch which proves that the delay can be reduced using switches.

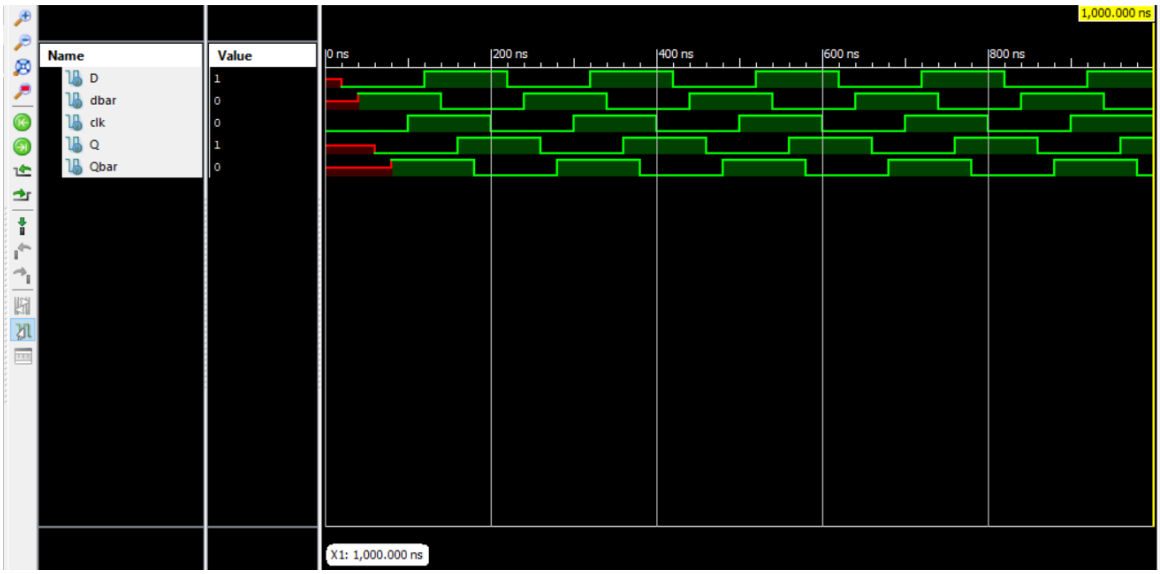


Figure 4.2 Output Waveforms Of D flip flop

The above waveforms show that D flip flop with complementary inputs produces the shift in the inputs to the outputs when the clock is applied.

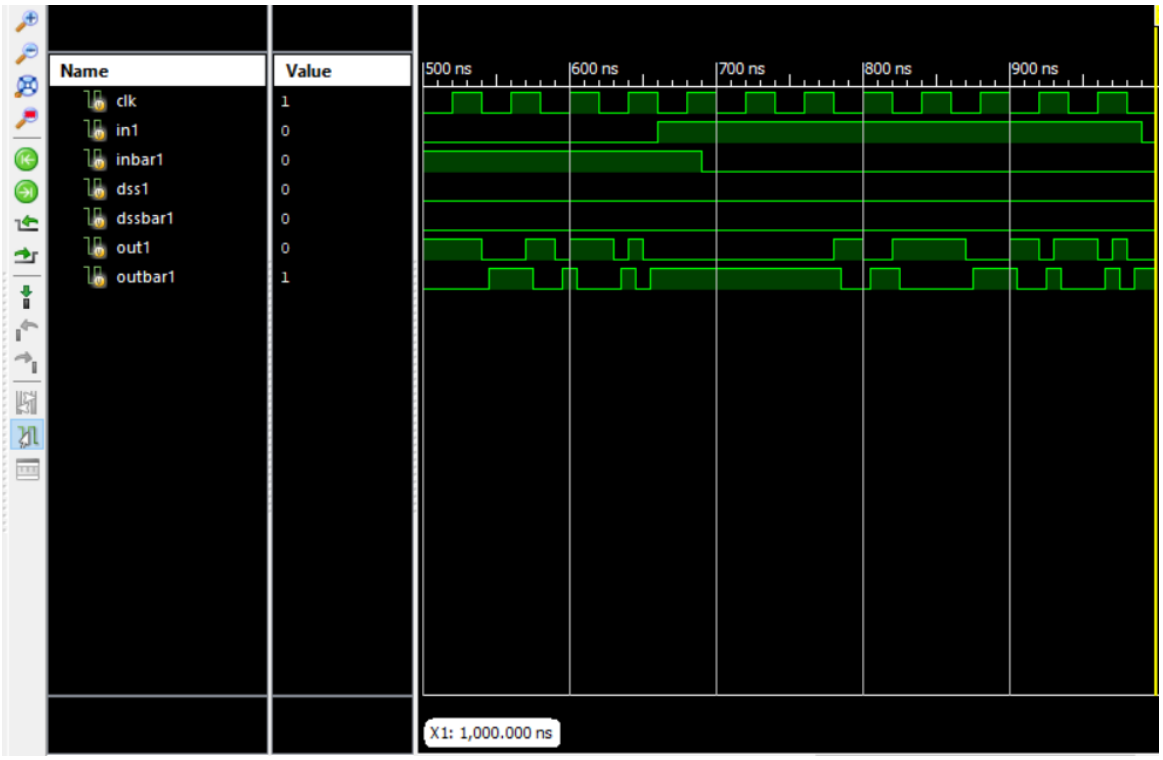


Figure 4.3 Output Waveforms Of Linear feedback shift Register(LFSR)

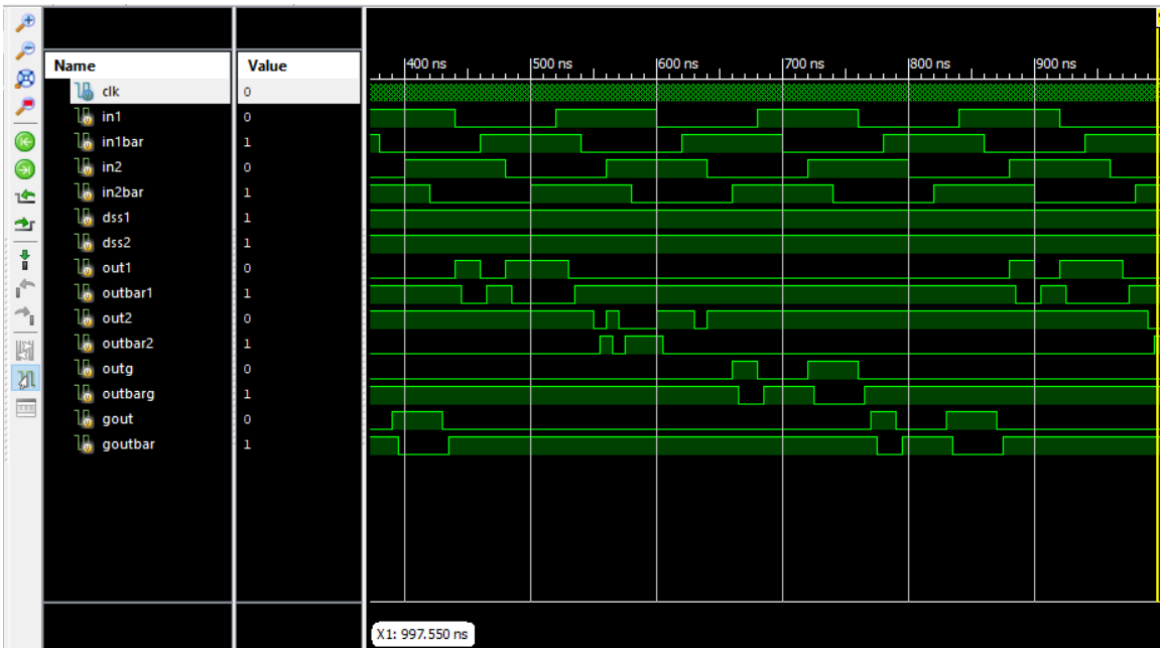


Figure 4.4 Output Waveforms Of Gold code Generator

LFSR1 has tapping at register 3 and 5, while LFSR2 has tapping at registers 2, 3, 4 and 5. Both the LFSR generate a pseudo random sequence. The output bits from LFSR are Xor'ed to obtain a gold code sequence.

CHAPTER 5
APPLICATIONS OF GOLD CODE GENERATOR

5.1 GPS

- 24 satellites orbit earth
- Atleast 6 are visible in the unobstructed sky at anytime
- All satellites broadcast 10-bit Gold Codes
- All share a 1.575 GHz carrier
- 1.023MHz code rate
- 1023 bits / sequence ---> repeats every 1ms
- Each satellite jams all of the others
- Thermal noise exceeds strength of all satellites combined
- But satellites are identified by correlation
- 50 Hz data rate
- Transmitted signal may be inverted based on data value.

5.2 BEACONS

- Eight LED beacons on the E11 playing field
- Beacon b (b = 1...8) flashes GC ($1+x^2+x^3+x^4+x^5$, $1+x^3+x^5$, b)
- 4 KHz data rate (250 microseconds / bit)
- Sequence is
- Normal when the beacon is WHITE
- Inverted when the beacon is GREEN
- Detect beacons using a phototransistor on your bot l
- Produces a voltage related to the light intensity
- Principles of operation to be described later.

 Identifying a Beacon

1.Read 31 phototransistor samples at exactly 4 KHz
2. Compute average value
3. Convert readings to binary by comparing to average

4. Correlate against each of 31 offsets for each of 8 beacons
5. If correlation exceeds a threshold, report beacon found
6. Improve accuracy by taking more than 31 samples.

5.3 CDMA

- Code Division Multiple Access (cell phones)
- All phones transmit on all frequencies simultaneously
- Each uses its own 15-bit (length 32767) Gold Code
- Identify the phone by correlating against its Gold Code
- Developed by Qualcomm
- Replaces Time Division Multiple Access
- Where each user gets a time slot (TDMA)
- Better quality reception when spectrum is not completely full
- Central to 3G and 4G wireless systems.

CHAPTER 6
CONCLUSIONS AND SCOPE FOR FUTURE WORK

6.1 CONCLUSION

In conclusion, the Gold Code Generator reference design is efficient because the implementation allows time sharing. In reusing the same resources, the code generator is able to produce 32 separate codes simultaneously. The circuit realization is very promising regarding the issues of versatility, reconfigurability and compactness. Gold sequences are constructed from pairs of preferred m-sequences of the same length by modulo-2 addition. It relies on flip-flops, which are serially interconnected and properly tapped to an XOR gate used for feedback. The correlation property of binary sequences and also the autocorrelation and cross-correlation properties for periodic and aperiodic sequences. In addition, we have included the correlation properties of the Gold codes. This verifies that a gold code can be produced in the optical domain in a straightforward manner, just like its electronic counterpart, without complex modifications in its standard structural form. The design can be used and extended for diverse applications for which gold code is required. This reference design presents a building block that will help shorten the development time of your W-CDMA System. The gold code generator was simulated using a standard XILINX tool.

6.2 FUTURE WORK

1. Design of a high speed,high precision reduced size lfsr.
2. Comparison with existing lfsr.
3. Designing of a multi bit Gold code generator

REFERENCES

1. E. Laskin and S. P. Voinigescu , "A 60 mW per Lane, 4x23-Gb/s 27-1 PRBS Generator", IEEE Journal of Solid-State Circuits, vol. 41, no. 10, pp2198-2208, Oct. 2006.

2. Y. -H. Peng and L. -H. Lu, "A 16-GHz Triple-Modulus PhaseSwitching Prescaler and Its Application to a 15-GHz Frequency Synthesizer in 0.18-um CMOS", IEEE Transactions on Microwave Theory and Techniques, vol. 55, no. 1, pp.44-51, Jan. 2007.

3. P. Heydari and R. Mohanavelu, "Design of Ultrahigh-Speed LowVoltage CMOS CML Buffers and Latches", IEEE Transactions on Very Large Scale Integration (VLSI) System, vol. 12, no. 10, pp.10811093, Oct. 2004.

4. J. Lee and B. Razavi, "A 40-Gb/s Clock and Data Recovery Circuit in 0.18-um CMOS Technology", IEEE Journal of Solid-State Circuits, vol. 38, no. 12, pp.2181-2191, Dec. 2003.

5. J. David, "Verification of CML circuits used in PLL contexts with Verilog-AMS", in Proc. Of IEEE International Behavioral Modeling and Simulation Conference, Session 4, pp. 97-102, Oct. 2006.

6. D. Perels, R. Bischoff, J. Biveroni, M. Bruehwiler, A. Burg, N. Felber, W. Fichtner, "Programmable Code Generator for Software Defined Radio", in Proc. of the of the 35th IEEE Asilomar Conf. on Signals, Systems, and Computers, pp. 2156-2160, Nov. 2003.

7. B. Forouzandeh, A. S. Seyedi, "Comparing the Performance of a LowPower High Speed Flip-Flop in Bulk and SOI Technologies", in Proc. of International Conference on Mixed Design of Integrated Circuits and Systems, pp. 251-255, June 2006.

8. P. Heydari and R. Mohanavelu, "A 40-GHz Flip-Flop-Based Frequency Divider", IEEE Transactions on Circuits and System--II: Express briefs, vol. 53, no. 12, Dec. 2006.

9. S.H. Rasouli, A. Amirabadi, A.Seyedi, A. Afzali-Kusha, "Double edge triggered feedbacked flipflop in sub 100 nm technology", in Proc. of the 2006 conference on Asia South Pacific design automation, pp. 297- 302, Jan. 2006.

10. M. George, M. Hamid, and A. Miller, "Gold Code Generators in Virtex Devices", Xilinx application note, XAPP217 (v1.1), Jan. 2001.

11. System performance criteria in CDMA networks using gold codes.
12. Gold Code Generator Reference Design.
13. Optimized Method for Generating and Acquiring GPS Gold Codes.
14. Simulation of Gold Code Sequences for Spread Spectrum Application.
15. TOAD-BASED ALL-OPTICAL GOLD CODE GENERATOR.
16. Wireless Communication Principles and Practice-Theodore S Rappaport.
17. Design of Pseudo Random Number Generator using Linear Feedback Shift Register-Shabbir Hassan, M. U. Bokhari.
18. Gold Codes lecture Fall 2014-Prof David Money Harris.
19. ALTERA Gold Code Generator Reference Design- Carnegie Mellon School of Computer Science
20. J. K. Holmes, Coherent Spread-Spectrum Systems, John Wiley and sons Inc., New York 1982.

APPENDIX A