Implementation and performance evaluation of multi level pseudo random sequence generator

Hadeer Hussein Ali[1], Hadi T. Ziboon[2], Ashwaq Q. Hameed[3]
[1]Department of Medical Instrumentation Techniques Engineering, Al-Salam University College, Baghdad, Iraq
[2]Department of Computer Engineering, Al-Salam University College, Baghdad, Iraq
[3]Department of Electrical Engineering, University of Technology, Baghdad, Iraq

Article Info

Article history:

Received Nov 9, 2021
Revised Jun 23, 2022
Accepted Nov 22, 2022

Keywords:

Adaptive
FPGA
MLPN
Random sequence circular

ABSTRACT

In this paper, introduce a proposed multi-level pseudo-random sequence generator (MLPN). Characterized by its flexibility in changing generated pseudo noise (PN) sequence according to a key between transmitter and receiver. Also, introduce derive of the mathematical model for the MLPN generator. This method is called multi-level because it uses more than PN sequence arranged as levels to generation the pseudo-random sequence. This work introduces a graphical method describe the data processing through MLPN generation. This MLPN sequence can be changed according to changing the key between transmitter and receiver. The MLPN provides different pseudo-random sequence lengths. This work provides the ability to implement MLPN practically in more than one method such as microcontroller or field programmable gate array (FPGA). In this paper discusses MLPN performance using MATLAB as compared with golden PN sequence generator with different modulation schemes such as binary phase-shift keying (BPSK), quadrature phase-shift keying (QPSK), and quadrature amplitude modulation 16QAM. The simulation results show that MLPN performs almost likely golden PN sequence but sure with advantage its flexibility to change generated MLPN between transmitter and receiver. The MLPN sequence is applicable in the same field of PN sequence applications such as code division multiple access (CDMA), spread spectrum system (SSS), and data scrambling.

Corresponding Author:

Hadeer Hussein Ali
Department of Computer Engineering, Al-Salam University College
Baghdad City, Al-Hussein Sec.881, Branch:48, House: 17, Iraq
Email: hadeer.hussein2018@gmail.com

1. INTRODUCTION

Main challenges in telecommunication systems, data transmission at a high rate while maintaining user security [1], as a result of the usage of spread spectrum technology in telecommunication systems, direct sequence code division multiple access (DS-CDMA) systems have been considerably increased. Because of its important characteristics, such as low power spectral density [2], wide bandwidth, low interception probability, transmitted signal under noise level, and robustness against multipath fading, anti-interference capability [3], the direct-sequence spread spectrum (DSSS) technique has been widely used in secured digital communications. To spread the bandwidth and transmit the signal under the noise level in DS-CDMA systems [4], the transmitters use periodic pseudo noise (PN) sequences and modulate the baseband signal before transmission [5].

Many applications, such as security applications and sophisticated communication systems, require random sequence frames [6]. One of the most well-known methods to generate the pseudo-random sequence is a linear feedback shift register (LFSR) [7]. It uses a set of shift registers connected in serial to create the random series [8]. This method LFSR is considered low complexity and could be implemented either using simple shift registers or using field programmable gate array (FPGA) [9]. The random number generation (RNG) was introduced to use in security applications and in cryptography [10], there is a real need for a random sequence with reasonable randomness probability to use in data ciphering [5]. Here, mathematics starts to get its role in random sequence generation. Many researchers start a competition to provide a sequence with the best randomness probability [11]. For example, using a gaussian random number generator (GRNG) algorithm to generate a random number sequence and examine the behavior of random numbers generated in the region of the Gaussian probability density function [12]. Also, pseudo-random sequence generations find their way into database applications; it introduced a generation algorithm with 2,000 database applications [13]. Also, time-variant recursions of accumulators are used to generate the random sequence, this method is characterized by low complexity uses the low resource of hardware [14]. Random sequence generations wildly used in communication applications, audio is used as a source to generate a random sequence with reasonable randomness probability [15]. The pseudo-random number generation also uses quadratic irrationals to produce a generator has fulfilled in the application of cryptography [16]. FPGA technology issued to implement the random sequence generator because of its flexibility in hardware implementation and high speed [17]. The security application takes benefit from the random sequence in modern cypher applications such as rabbit cypher [18]. The generated pseudo-random sequence using a computer may suffer from the good arbitrary start point [19], but the rest of the random series has deterministic properties or also parodic properties [19]. Quantum number generation introduced to use as a random number generator with excellent randomness properties over typical random number generator [20].

In this paper, we focus on using a random sequence as the adaptive sequence between both the transmitter and receiver in a communication application. In the next section, introduce the proposed multi-level pseudo-random sequence generator (MLPN) mathematical model.

2. PN SEQUENCE

PN is a sequence of binary values, such as ±1, that looks to be random but is actually absolutely deterministic and periodic, LFSR are commonly used to create PN sequences [21]. The number of shifts register stages determines the length of the PN sequence. The maximum allowable PN sequence length, N is provided by (2) assuming there are m shift registers used [22]:

$$N = 2^n - 1 \tag{1}$$

Such a sequence is referred to as a maximal length sequence**.** The recursive formula is used to construct the sequence a_i according to the formula [7]:

$$a_i = C_1 a_{i-1} + C_2 a_{i-2} + \cdots C_n a_{i-n} = \sum_{k=1}^{n} C_k a_{i-k} \tag{2}$$

Additions and multiplications in the above formula represent the modulo-2, and all terms are binary (0 or 1). The connection vectors C_1, C_2,........... C_n sets the major properties of the generated sequence by defining the LFSR sequence generator's characteristic polynomial. The PN code produced by the LFSR with n flip-flops is periodic. The main constraints of PN sequence.

PN sequence represent statistical random sequence and includes several characteristics such as nearly equal numbers of zeros and ones. The shifted versions of the sequence have a very poor correlation and cross-correlation with other signals, such as interference and noise, is extremely low. The main disadvantage of PN codes is their small code family size, which means they only serve a small number of mobile users [23].

3. GOLDEN PN SEQUENCES

In 1967 and1968, Gold suggested the gold sequences. It refers to a particular type of binary random (Pseudo Random) sequence in which the correlation between member sequences is extremely low, the golden are made by XOR two m sequences of the same length. Gold code are generated by XOR pair of PN sequence. For a gold sequence of length $N = 2^n - 1$, two L linear feedback shift register LFSR of length $2^n - 1$ are used. Gold sequences provide superior cross-correlation qualities than maximum length LFSR sequences when the LFSRs are chosen suitably. The gold code has the advantage of a uniform and bounded cross correlation between the two codes [24].

Gold and Kasami demonstrated that the cross correlation only has three possible values for certain well-chosen m-sequences, namely-1, -t, and t-2. The terms "preferred sequences" refer to two of these types of sequences. The length of the LFSR utilized here is purely dependent on *t*. The length of the LFSR utilized here is purely dependent on *t* [25]. Figure 1 explain block diagram of golden code sequence generator:

$$\text{If m is odd } t = 2^{(m+1)/2} + 1 \tag{3}$$

$$\text{If m is even } t = 2^{(m+2)/2} + 1 \tag{4}$$

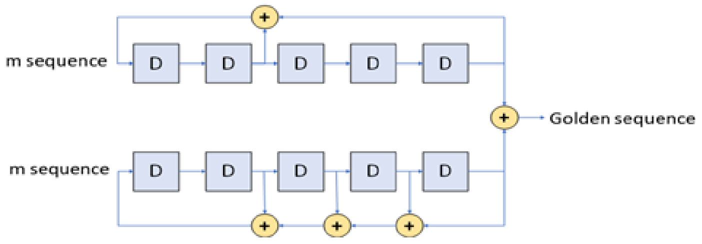


Figure 1. Golden PN sequence generator

In comparison to its PN sequence, gold codes have a larger code family size, which gives them more importance, so that gold codes overcome the drawback of PN codes' small code family size [22].

4. MULTI-LEVEL PSEUDO RANDOM SEQUENCE GENERATOR MATHEMATICAL MODEL

To understand the need for proposed MLPN sequence generator. Golden PN sequence has a good characteristic but with drawback its fixed, in this paper, introduce MLPN sequence pass the problem of changing the PN sequence used between the transmitter and receiver. The golden PN sequence is fixed. So, if we use it to process the data between the transmitter and receiver there is a chance for an unauthorized intruder to discover the used golden PN sequence and then the transmitted data. The proposed MLPN sequence pass this problem by a proposed design based on a mathematical model used to generate a PN sequence a cording a pre-defined Key between transmitter and receiver. This key can change with a large number of possibilities between transmitter and receiver. MLPN sequence need a special design of levels organized as net to discover the original data. MLPN sequence gives level of security for transmitted data. So, the unauthorized intruder needs the key and the MLPN sequence Net to discover the transmitted data. In this article discuss the mathematical model used to generate MLPN sequence. This discussion covers the following:

a. Introduce the mathematical model for the proposed MLPN sequence generator.
b. The mathematical model gives the facility of changing generating PN key according to presenting between transmitter and receiver, so it provides an adaptive process. And produce different pseudo-random sequence at each different key.
c. The MLPN generator gives the ability to change the random sequence length according to the presenting parameter.

Consider the MLPN sequence as rectangular (or window) as shown in Figure 2 and each row called level; these levels have shifted version of reference golden code (G_1). Shifted versions of golden code takes the last element and rotate it to first location and push the other elements to the right side so:
Golden code PN sequence vector.

$$\text{G}_1=[G_1\,(1).G_1\,(2)\,\ldots.\;G_1(1{:}\,n-1).G_1\,(n)\,] \tag{5}$$

Shifted golden code by one bit,

$$G_2 = [G_1\,(n).G_1\,(1).G_1\,(2)\,\ldots.\;G_1(1{:}\,n-1)] \tag{6}$$

By the same way define other shifted versions of golden sequences to generate,

$$G_3 = [G_2\,(n).G_2\,(1).G_2\,(2)\,\ldots.\;G_2(1{:}\,n-1)] \tag{7}$$

$$G_4 = [G_3\,(n).G_3\,(1).G_3\,(2)\,\ldots.\;G_3(1{:}\,n-1)] \tag{8}$$

Where n is number of golden code elements. Each shifted version of golden code represents level. So, from her comes the name multiple level. Because we will use these multiple levels in PN sequence generation. If we define a pair which indicate the start and end of the movements along the levels. Where each shifted version of golden sequence called level so:

$$Pair = \big(S(1).E(1)\big) \tag{9}$$

Where $S(1)$ represent start of transition on level, $E(1)$ represent end of transition on level. Each pair give two bits, one from start state and the second from the end state. Group of pairs, if we combine more than single pair in a group, we can construct what so called group of pairs, this group of pairs will represent the key of MLPN sequence generator.

$$P = \big\{\big(S(1).E(1)\big).\big(S(2).E(2)\big).\big(S(3).E(3)\big) \ldots \big(S(N).E(N)\big)\big\} \tag{10}$$

Where N=number of pairs.

$$P_i = \lim_{i=1\to N}\big\{\big(S(i).E(i)\big)\big\} \tag{11}$$

Where $i = index\ number\ of\ pair$. So, if we refer to the key as a mathematical representation of window, the key of seven bits and two steps with four levels could be in the form:

$$n\text{=number of bits}=2^{(\text{number of D registers})}\text{-}1=7 \tag{12}$$

$$P_1 = \lim_{i=1\to\frac{n+1}{l}}\big\{\big(S(i).\ E(i)\big)\big\} \qquad \text{first group for first step} \tag{13}$$

$$P_2 = \lim_{i=\frac{n+1}{l}+1\to\frac{n+1}{2}}\big\{\big(S(i).E(i)\big)\big\} \qquad \text{second group for second step} \tag{14}$$

Where l represent number of levels, each group represent half of the golden code sequence,

$$P_1 \text{ takes 1, 2 } states=\{\big(S(1).E(1)\big)\ \big(S(2).E(2)\big)\} \tag{15}$$

$$P_2 \text{ takes 3, 4 states}=\{\big(S(3).E(3)\big)\ \big(S(4).E(4)\big)\} \tag{16}$$

Step length=number of levels
The probability of start $S(i)$ and end $E(i)$ of each group according to the window diagram in Figure 3 is equal to:
For first step we can consider four start probability with end state as:
1-First level, R represent the result vector
The start states
$R_1(1) = G_1(1)\ at\ S\ (1) = 1$
The end states

$$\left.\begin{aligned} R_1(2) &= G_1(2)\ at\ E(1) = 1\\ R_1(2) &= G_2(2)\ at\ E(1) = 2\\ R_1(2) &= G_3(3)\ at\ E(1) = 3\\ R_1(2) &= G_4(4)\ at\ E(1) = 4\end{aligned}\right\} \quad \text{First state end options} \tag{17}$$

2-Second level
the start states
$R_1(3) = G_2(1)\ at\ S\ (2)\ =\ 2$
The end states

$$\left.\begin{aligned} R_1(4) &= G_1(2)\ at\ E(2) = 1\\ R_1(4) &= G_2(2)\ at\ E(2) = 2\\ R_1(4) &= G_3(3)\ at\ E(2) = 3\\ R_1(4) &= G_4(3)\ at\ E(2) = 4\end{aligned}\right\} \quad \text{second state end options} \tag{18}$$

3-Third level
the start states
$R_1(5) = G_3(1)\ at\ S(3) = 3$
The end states

$$\left.\begin{array}{l} R_1(6) = G_1(3)\ at\ E(3) = 1 \\ R_1(6) = G_2(2)\ at\ E(3) = 2 \\ R_1(6) = G_3(2)\ at\ E(3) = 3 \\ R_1(6) = G_4(2)\ at\ E(3) = 4 \end{array}\right\} \quad \text{Third state end options} \qquad (19)$$

4-Forth level
the start states
$R_1(7) = G_4(1)\ at\ S(4) = 4$
The end states

$$\left.\begin{array}{l} R_1(8) = G_1(4)\ at\ E(4) = 1 \\ R_1(8) = G_2(3)\ at\ E(4) = 2 \\ R_1(8) = G_3(2)\ at\ E(4) = 3 \\ R_1(8) = G_4(2)\ at\ E(4) = 4 \end{array}\right\} \quad \text{Fourth state end options} \qquad (20)$$

Note: we can apply key pair as desired, for example there is possibility to apply 4 pairs in first step, but sure we can choose two only. Or even multiple pairs with random order. Just keep in mind each pair gives two bits, the first from start point and the second in the end point. The window has flexibility in generation MLPN sequence. Note for the second step, the same window repeated the probability of second window by itself is the same of first step. Just consider shift in position by number of levels. For example, start of first level will be in location 4 instead of location one of first step and so on. Refer to Figure 3 gives good understanding of state transition on window diagram. In following illustrative example how to construct and generate MLPN sequence. Consider golden sequence has PN vector as shown:
$G_1 = \{0.1.1.0.1.0.1.0\}$ the shifted version of this golden PN sequence will be
$G_2 = \{0.0.1.1.0.1.0.1\}$, $G_3 = \{1.0.0.1.1.0.1.0\}$ and $G_4 = \{0.1.0.0.1.1.0.1\}$ respectively.
Figure 4 show the window diagram of four levels and two steps.
If we consider two groups of keys. First group consist of two pairs only and so the second group
$P_1 = \{(3.2).(4.1)\}$ Gives from step1 $R_1 = [1.0.0.0]$
$P_2 = \{(3.1).(3.4)\}$ Gives from step2 $R_2 = [1.0.1.1]$
R $_{\text{total}}$= $[R1\ R2] = \{1.0.0.0.1.0.1.1\}$
The result MLPN sequence = $\{1.0.0.0.1.0.1.1\}$
Figure 2 explain MLPN with four level

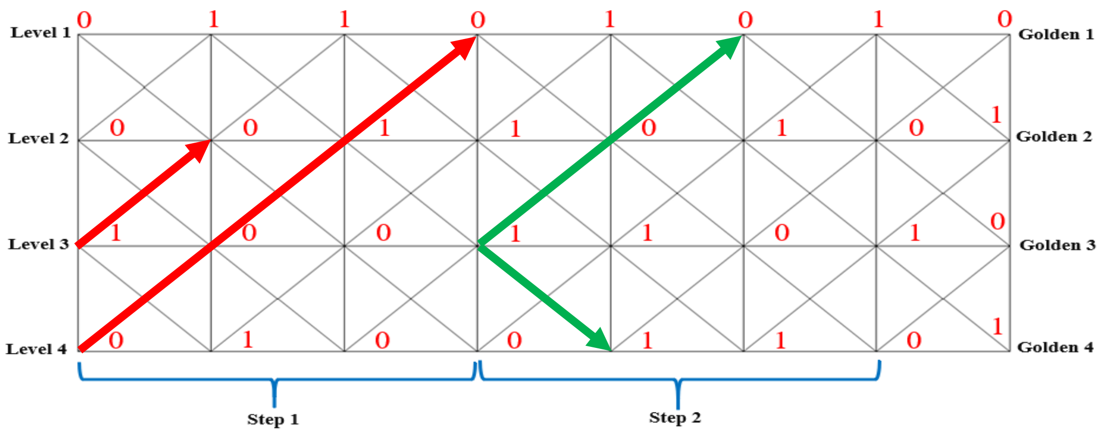


Figure 2. Window diagram four levels with two steps

Discuss properties of MLPN sequence. MLPN sequence has some properties, these properties as:
a. $G_1.\ G_2.\ G_3.\cdots G_i$ the shifted version of PN sequence can arranged in any desired order so shifted levels can distribute randomly to increase randomness.

b. Increase number of levels as desired, but the single pair still gives two bits only related to start state and end state, the step length sure will increase with increasing number of levels.
c. The pairs group represent the secret key between the transmitter and the receiver, these groups can be chosen as desired between transmitter and receiver and sure changed easily. Each change gives different MLPN sequence.
d. Generated PN by this method can be applied easily to any application use PN sequence in its construction.
e. The first level G_1 can be golden sequence or any other PN sequence. Also, each level can use any random generated PN in each level. So, each level may include its own PN sequence not always the shifted version.

The pairs group represent the secret key between the transmitter and the receiver, these groups can be chosen as desired between transmitter and receiver and sure changed easily. Each change gives different MLPN sequence.

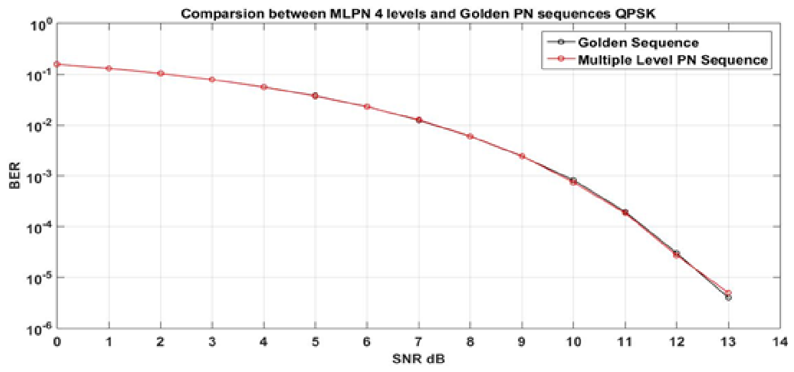


Figure 3. MLPN 4 levels, golden sequence with QPSK modulation

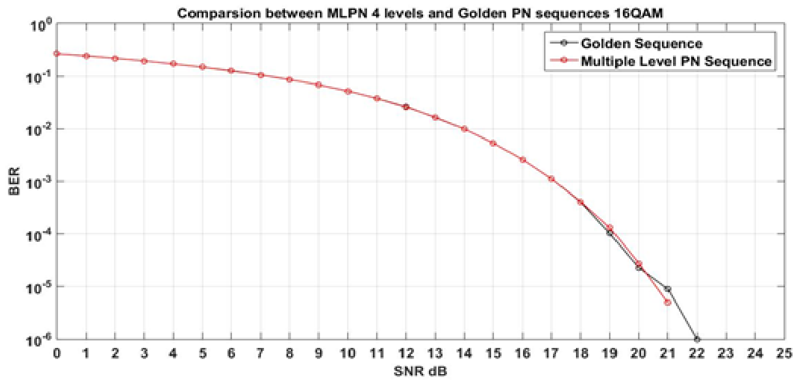


Figure 4. MLPN 4 levels, golden sequence with 16QAM modulation

5. CORRELATION

The correlation properties of PN codes are significant in CDMA code design because they determine not only the level of channel access interference [6]. which contains interference from other users of the channel as well as self-interference due to multi-path propagation, but also the code acquisition properties. The concept of correlation is determining the degree of similarity between two sets of data. It has a range of - 1 to 1 as its definition. A partial correlation value is shown by the other value. The correlation factor can be categorized in two types Auto correlation and cross correlation. In auto correlation the matching of sequence with all phase shifts of itself is known as auto correlation. pure random data must have a correlation value around to 0 about any auto correlations with a phase angle other than zero, the auto correlation formula is defining as:

$$R(\tau) = \frac{1}{N}\sum_{k=1}^{N} B_k B_{k-\tau} \qquad where\ \tau = 0.N.2N \tag{21}$$

Cross correlation compares two sequences from different source instead of comparing a shifted copy of a sequence with itself. The cross correlation between two sequence A and B is defining as:

$$R_{A.B}(\tau) = \frac{1}{N}\sum_{k=1}^{N} A_k B_{k-\tau} \qquad where\ \tau = 0.N.2N \tag{22}$$

In next paragraph discuss the performance of MLPN sequence generator as compared with Golden sequence using MATLAB software package. Also consider the auto-correlation properties for each generated PN sequence.

6. MLPN SEQUENCE PERFORMANCE SIMULATION

MLPN sequence generated using the window diagram. In previous paragraphs discuss the mathematical model, generation with illustrated examples. Here discusses the performance of generated MLPN sequence as compared with good performance well known golden code. The tool used to test the performance is MATLAB software package. Actually, the test includes programming a communication system deals with PN sequence. This communication system tested with different modulation schemes. The modulation schemes include BPSK, QPSK, and 16QAM. Figure 3 explain BER perform of MLPN at 4 levels with QPSK.

The system uses PN sequence generator to generate either golden sequence or MLPN sequence. This depends on our choice during simulation. Also, for modulation schemes, the simulation uses one of these modulation schemes during simulation, this depends on the case under test. So, this simple communication system will test golden sequence and MLPN sequence under same modulation scheme. The cases under test with its parameter listed in Table 1. Note GP abbreviation refers to golden sequence polynomial. Golden sequence has two generating polynomials. The D refers to D- register. The initial condition of D registers is shown in Table 1 for each case. Note golden sequence is periodic, it means the generation polynomial used to generate 8 bits is the same for generation 16 bit, because its periodic the bits frame repeat itself. We can take the golden sequence length as needed in simulation. For MLPN sequence simulation consider two types one based on four levels with two steps and two groups, each group consist of two pairs, each pair gives two bits. So over all gives 8-bits. The second MLPN consist of five levels with four steps. Each step will operate with its own group. Each group designed with two pairs. Recall each pair gives two bits. So over all we generate 16-bit MLPN sequence. The simulation results listed in Figures 5-8. The comparison between all systems summarized in Table 2. And followed by comment on results. Figure 4 refers to comparison of BER performance of MLPN 4 with golden at 16QAM. Figure 5 refers to comparison of BER performance of MLPN 5 with golden at BPSK. Figure 6 refers to comparison of BER performance of MLPN 5 with golden at QPSK. Figure 7 refers to comparison of BER performance of MLPN 5 with golden at 16QAM.

Table 1. Sequence parameter of MLPN, golden sequence with different types of modulation schemes

MLPN sequence parameter	Golden sequence parameter	Modulation scheme
G1=[1,1,0,0,0,0,1,1]	First polynomial:	1-BPSK
4 Levels, two steps.	$GP_1=D^4+D^3+1$.	2-QPSK
$P_1=\{(4, 1), (3, 2)\}$	Initial GP_1=[0,0,0,1].	3-16QAM
$P_2=\{(3, 1), (4, 3)\}$	$GP_2=D^4+D^2+1$.	
MLPN length=8 bit	Initial GP_2=[0,0,0,1].	
	Golden length=8 bit.	
G1=[1,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,1]	First polynomial:	1-BPSK
5 Levels, four steps	$GP_1=D^4+D^3+1$.	2-QPSK
$P_1=\{(1, 4), (3, 1)\}$,	Initial GP_1=[0,0,0,1].	3-16QAM
$P_2=\{(2, 4), (4, 3)\}$	$GP_2=D^4+D^2+1$.	
$P_3=\{(5, 1), (2, 5)\}$	Initial GP_2=[0,0,0,1].	
$P_4=\{(4, 3), (5, 3)\}$	Golden length=16 bit.	
MLPN length=16 bit.		

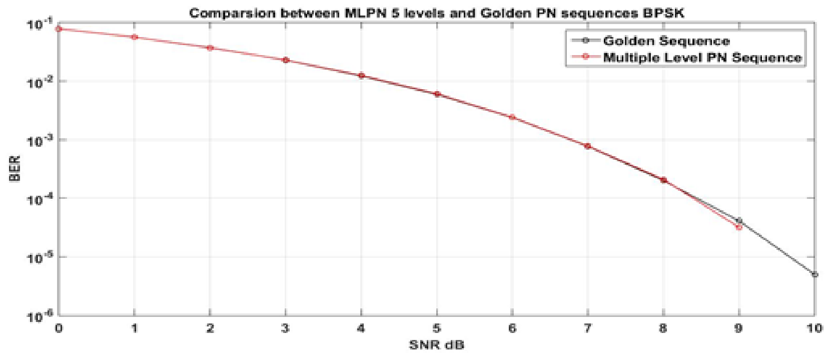


Figure 5. MLPN 5 levels, the golden sequence with BPSK modulation

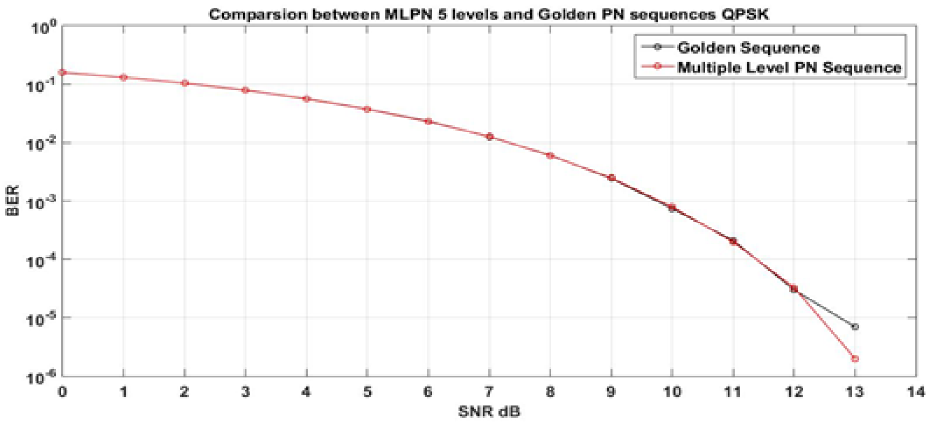


Figure 6. MLPN 5 levels, golden sequence with QPSK modulation

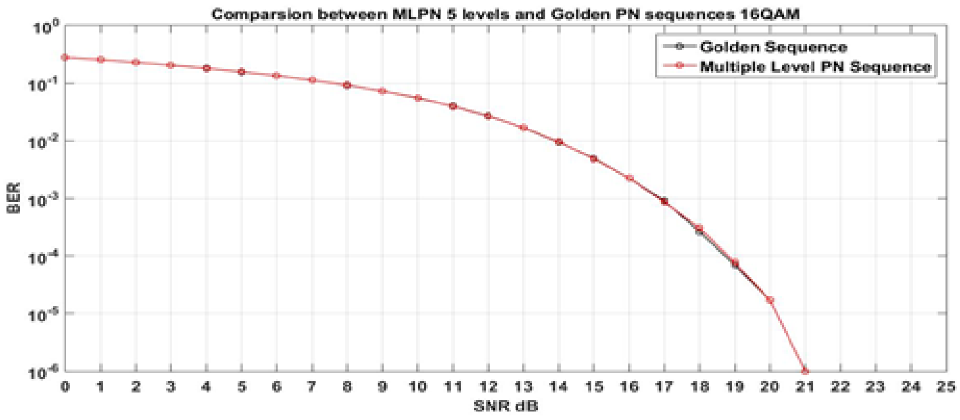


Figure 7. MLPN 5 levels, golden sequence with 16QAM modulation

Table 2. Simulation results comparison between MLPN and golden sequence

	MLPN 4 levels		Golden sequence	
Modulation	SNR	BER	SNR	BER
BPSK	10	4×10^{-06}	10	3×10^{-06}
QPSK	13	5×10^{-06}	13	4×10^{-06}
16QAM	21	5×10^{-06}	22	2.3×10^{-05}
	MLPN 5 levels		Golden sequence	
Modulation	SNR	BER	SNR	BER
BPSK	9	3.2×10^{-05}	10	5×10^{-06}
QPSK	13	2×10^{-06}	13	7×10^{-06}
16QAM	21	1×10^{-06}	20	1×10^{-06}

Referring to simulation results shown in Table 2. It shows the proposed MLPN sequence gives a good performance as comparison with the golden code sequence. It actually the same performance for a large range of SNR. The small difference at the end of the bit error rate (BER) performance curve. So, the performance almost the same. But from the other side of view. The MLPN sequence has more flexibility in changing the sequence, by changing the generation key. While for golden sequence it's fixed. The MLPN sequence gives a new PN sequence when changing the generation key. It does not represent the shifted version of the previous MLPN sequence. But it generates a new sequence depending on the Key and the pairs included inside groups. This property gives the proposed MLPN sequence advantage over the common golden code sequence. Its application can include data security too, because its PN sequence changed with key. The probability of finding new sequence depends on number of steps. An increasing number of levels and number of steps with number of groups gives a good probability to generate a large range of MLPN sequences.

The auto correlation for MLPN sequence of four levels and golden sequence is shown in Figure 8. The cross correlation between two sequences MLPN and golden with 8-bits in length is shown in Figure 9.

The auto correlation show mirror at the center of 8 for both MLPN and golden sequence. The correlation shows the similarity between two sequences. Also, the auto correlation and cross correlation for MLPN and golden sequence for 16-bit length are shown in Figure 10 and Figure 11 respectively.

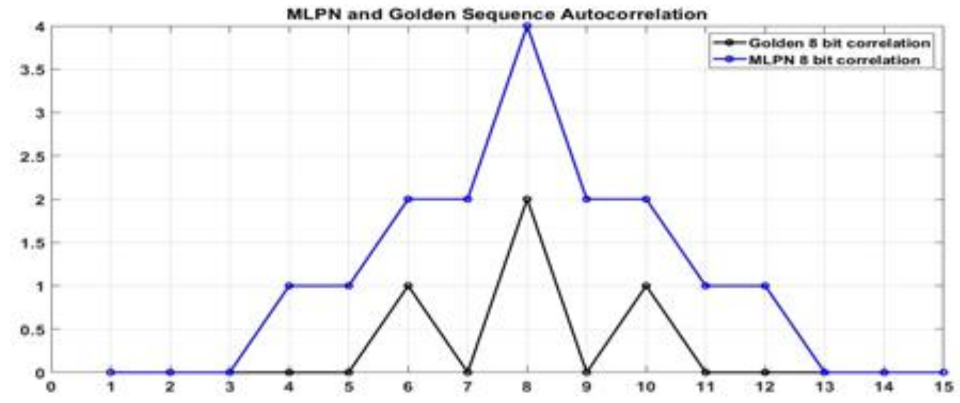


Figure 8. Autocorrelation MLPN and golden sequence 8-bit length

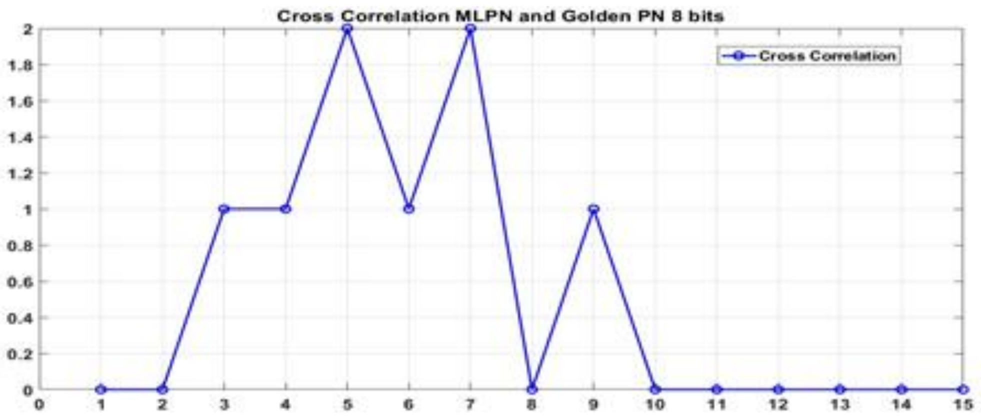


Figure 9. Cross correlation MLPN and golden sequence 8-bit length

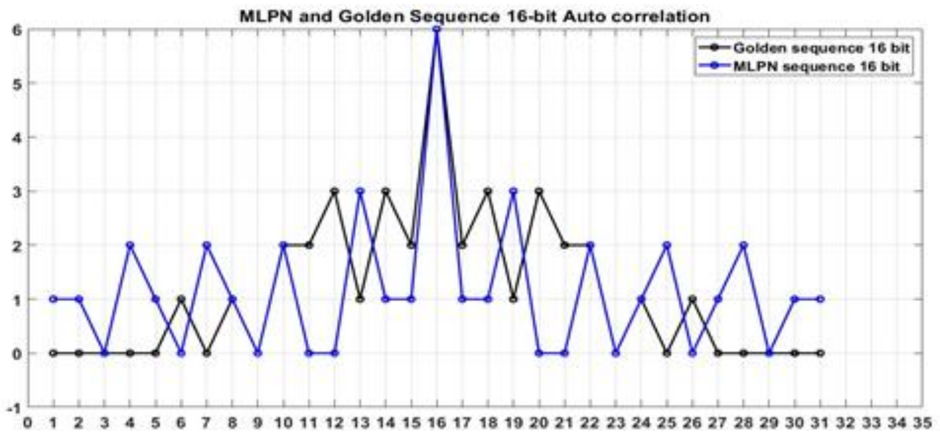


Figure 10. Auto correlation MLPN and golden sequence 16-bit length

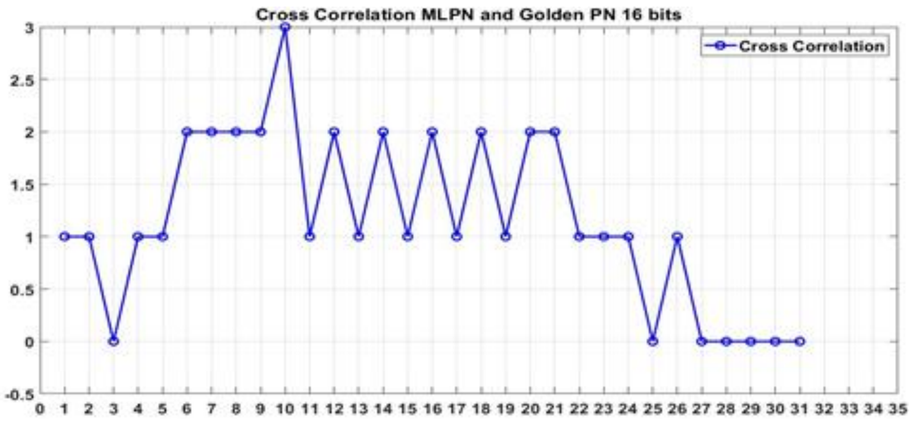
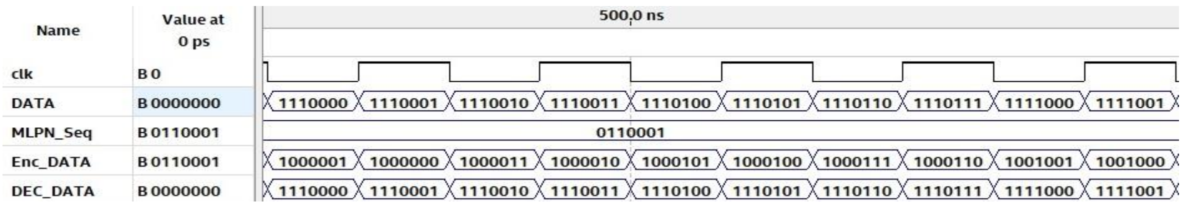


Figure 11. Cross correlation MLPN and golden sequence 16-bit length

7. MLPN SEQUENCE GENERATOR FPGA IMPLEMENTATION

MLPN sequence generator designed to give flexible changed PN sequence. The mathematical model that describes the generator is quite simple; it represents the operation followed to generate the sequence. The transmitter side and receiver side could quickly agree with the same key to generate the random sequence. MLPN can implemented using microcontroller or FPGA. The FPGA choice provides the ability to implement the MLPN sequence generator with high speed in operation. The MLPN sequence generator implementation depends on the mathematical model, because it describes the mathematical operations used to generate the PN sequence according a key. This mathematical operation programmed using very high-speed description language VHDL. Figure 12 show the Universal test waveform of the MLPN sequence. Figure 13 shows block diagram of MLPN sequence FGPA implementation. Figure 14 show block diagram of FPGA implementation. Figure 14 show flow chart steps to implement MLPN sequence generator using VHDL language. The implementation uses Quartus software with Cyclone IV EP4CE6E22C8N FPGA from Altera.

The universal waveform test is used to verify the implementation, test its operation with variable inputs. The design includes 7 bits MLPN sequence XOR with input data for the encoded data vector then XOR again with the encoded data vector to restore the input data vector. The decoded data vector sure exactly the same input data vector. The simulation shows the Implementation of MLPN sequence generator works without any mistakes. Figure 15 show the MLPN sequence netlist view.



Name	Value at 0 ps	500,0 ns									
clk	B 0										
DATA	B 0000000	1110000	1110001	1110010	1110011	1110100	1110101	1110110	1110111	1111000	1111001
MLPN_Seq	B 0110001	0110001									
Enc_DATA	B 0110001	1000001	1000000	1000011	1000010	1000101	1000100	1000111	1000110	1001001	1001000
DEC_DATA	B 0000000	1110000	1110001	1110010	1110011	1110100	1110101	1110110	1110111	1111000	1111001

Figure 12. Universal waveform test of MLPN sequence generator FPGA implementation

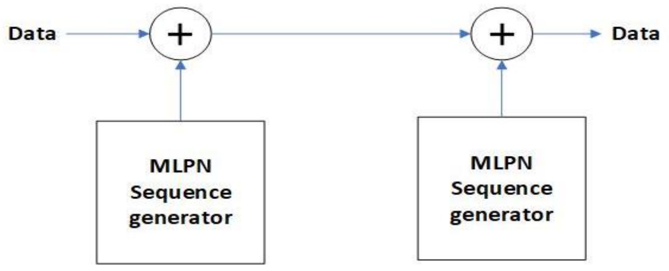


Figure 13. MLPN sequence FGPA implementation block diagram

The simulation and test of the MLPN sequence generator show the possibility to implement it using FPGA. But keep in mind the simple and clear mathematical model permit to of use other technologies to implement MLPN sequence generator. For example, referring to the microcontroller. It uses integrated development environment IDE in design and program variable systems. Its language characterized by its easy and powerful C language. Program MLPN sequence generator not so hard job, actually its very similar to simulating MLPN using MATLAB with m files extension type. This gives the facility and ability to use MLPN sequence generator with variable applications.

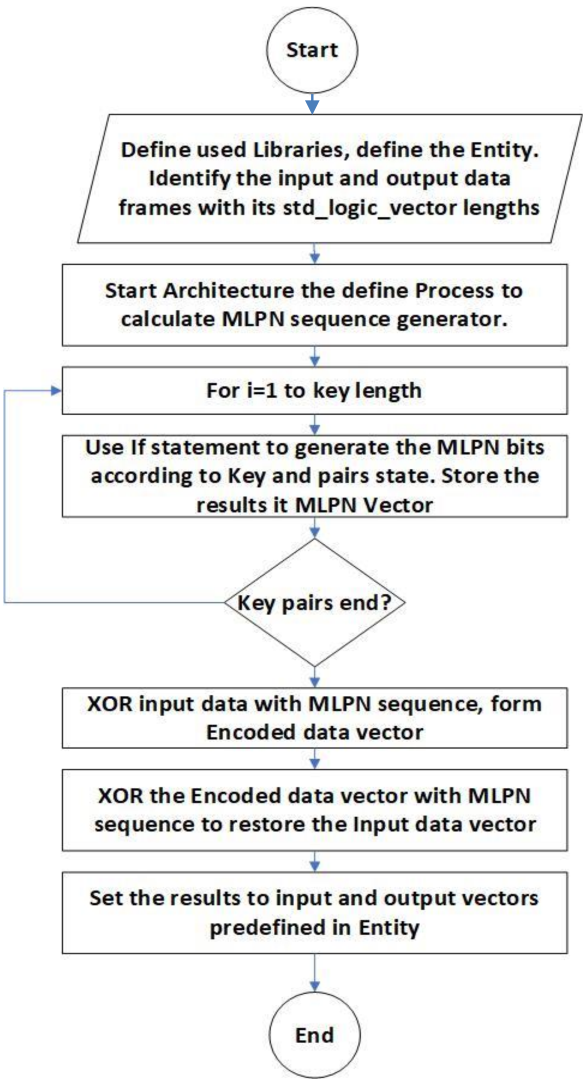

Start
Define used Libraries, define the Entity. Identify the input and output data frames with its std_logic_vector lengths
Start Architecture the define Process to calculate MLPN sequence generator.
For i=1 to key length
Use If statement to generate the MLPN bits according to Key and pairs state. Store the results it MLPN Vector
Key pairs end?
XOR input data with MLPN sequence, form Encoded data vector
XOR the Encoded data vector with MLPN sequence to restore the Input data vector
Set the results to input and output vectors predefined in Entity
End

Figure 14. Flow chart FPGA implementation of MLPN sequence generator using VHDL language

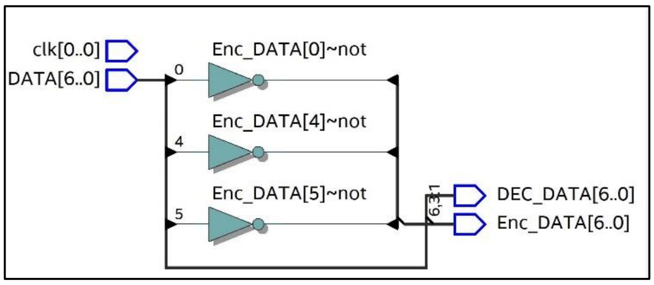


Figure 15. MLPN sequence generator FPGA netlist view

8. CONCLUSION

In this paper introduce a proposal pseudo random sequence generator, introduce its mathematical model and discuss its operation with illustrative examples. This MLPN sequence generator characterized by its simple mathematical model, can generate different random sequence lengths according to the key. Also, it can provide adaptive operation with agreeing between transmitter and receiver. This paper also introduces FPGA implementation using Altera FPGA, to show this MLPN sequence generator is applicable. In this paper discuss two MLPN generators, one with four levels and the second with five levels. But the design can consist of multiple levels with a different number of steps. Use MATLAB to comparison between MLPN and well-known golden sequence, the BER curves show MLPN has good performance likely golden sequence for a large range SNR. The construction of MLPN decided by the designer according to application. And take into consideration the complexity of the target design. Finally, this MLPN sequence generator can be used in security applications, especially with systems that not need sophisticated design with the accepted security level.

REFERENCES

[1] A. Sharma, T. Anwar, and D. K. Sharma, "Simulation of Gold Code Sequences for Spread Spectrum Application," *International Journal of Engineering and Technical Research (IJETR)*, no. 7, pp. 129–132, 2014.

[2] P. Nagaradjane, A. Sai Sarathi Vasan, L. Krishnan, and A. Venkataswamy, "Adaptive Co-Channel Interference Suppression Technique for Multi-User MIMO MC DS/CDMA Systems," *International Journal of Communications, Network and System Sciences*, vol. 02, no. 09, pp. 822–826, 2009, doi: 10.4236/ijcns.2009.29095.

[3] Z. Shi, Y. Zhang, Z. Qian, X. Sun, and X. Ma, "Compressive narrowband interference detection and parameter estimation in direct sequence spread spectrum communication," *IET Signal Process.*, no. June, pp. 1–12, 2021, doi: 10.1049/sil2.12075.

[4] M. N. Haque and M. K. Shil, "Performance Analysis of Multiuser MC-CDMA System for Nakagami-m, Rayleigh and Rician Fading Channel Using a New Set of Spreading Codes," *International Journal of Electrical Engineering and Computer Science (EEACS)*, vol. 3, pp. 97–104, 2021.

[5] P. Chen and A. Duncan, "Scrambled Direct-Sequence Spread-Spectrum Underwater Acoustic Communication System," *Editors*, vol. 2021, pp. 1–5, 2021.

[6] A. H. Khaleel and I. Q. Abduljaleel, "A novel technique for speech encryption based on k-means clustering and quantum chaotic map," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 160–170, 2021, doi: 10.11591/eei.v10i1.2405.

[7] Z. Chen, Z. Niu, Y. Sang, and C. Wu, "Arithmetic Autocorrelation of Binary m-Sequences," pp. 1–9, 2021, [Online]. Available: http://arxiv.org/abs/2111.11176.

[8] Z. Lin, Z. Ni, L. Kuang, C. Jiang, B. Liu and Z. Huang, "A Rapid PN Code Acquisition Method for Low Spreading Factor Satellite Communication Systems," in *IEEE Communications Letters*, vol. 25, no. 11, pp. 3664-3668, Nov. 2021, doi: 10.1109/LCOMM.2021.3053011.

[9] K. N. Devika and R. Bhakthavatchalu, "Design of reconfigurable LFSR for VLSI IC testing in ASIC and FPGA," *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017, pp. 0928-0932, doi: 10.1109/ICCSP.2017.8286506.

[10] A. Rukhin, J. Soto, and J. Nechvatal, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," *Nist Spec. Publ.*, vol. 22, no. April, pp. 1/1--G/1, 2010, [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/rng/documents/SP800-22rev1a.pdf.

[11] M. Jafari Barani, P. Ayubi, M. Yousefi Valandar, and B. Y. Irani, "A new Pseudo random number generator based on generalized Newton complex map with dynamic key," *Journal of Information Security and Applications*, vol. 53, p. 102509, 2020, doi: 10.1016/j.jisa.2020.102509.

[12] J. S. Malik and A. Hemani, "Gaussian random number generation: A survey on hardware architectures," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–37, 2016, doi: 10.1145/2980052.

[13] Z. Liu, M. Huang and S. Zhu, "The Design and Implementation of a Pseudo Random Number Generation Algorithm," *2009 International Conference on Computational Intelligence and Natural Computing*, 2009, pp. 126-129, doi: 10.1109/CINC.2009.242.

[14] V. R. Gonzalez-Diaz, F. Pareschi, G. Setti and F. Maloberti, "A Pseudorandom Number Generator Based on Time-Variant Recursion of Accumulators," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, no. 9, pp. 580-584, Sept. 2011, doi: 10.1109/TCSII.2011.2161165.

[15] C. Ryan, M. Kshirsagar, G. Vaidya, A. Cunningham, and R. Sivaraman, "Design of a Cryptographically Secure Pseudo Random Number Generator with Grammatical Evolution," *Scientific reports*, vol. 12, 2021, doi: 10.1038/s41598-022-11613-x.

[16] T. R. Devi, "Importance of Cryptography in Network Security," *2013 International Conference on Communication Systems and Network Technologies*, 2013, pp. 462-467, doi: 10.1109/CSNT.2013.102.

[17] M. Bakiri, C. Guyeux, J. F. Couchot, and A. K. Oudjida, "Survey on hardware implementation of random number generators on FPGA: Theory and experimental analyses," *Computer Science Review*, vol. 27, pp. 135–153, 2018, doi: 10.1016/j.cosrev.2018.01.002.

[18] A. V. Tutueva, E. G. Nepomuceno, A. I. Karimov, V. S. Andreev, and D. N. Butusov, "Adaptive chaotic maps and their application to pseudo-random numbers generation," *Chaos, Solitons and Fractals*, vol. 133, 2020, doi: 10.1016/j.chaos.2020.109615.

[19] R. Prabhu, R. Nagarajan, N. Karthick, and S. Suresh, "Implementation of Direct Sequence Spread Spectrum Communication System Using FPGA," *International Journal of Advanced Engineering, Management and Science (IJAEMS)*, vol. 3, no. 5, pp. 488–496, 2017, doi: 10.24001/ijaems.3.5.13.

[20] X. Ma, X. Yuan, Z. Cao, B. Qi, and Z. Zhang, "Quantum random number generation," *npj Quantum Information*, vol. 2, no. 1, pp. 1–9, 2016, doi: 10.1038/npjqi.2016.21.

[21] M. F. Diagana and S. B. Gueye, "Analysis, Design, and Test of CDMA LFSR with Offset Mask Using Standard ICs," *Engineering*, vol. 08, no. 04, pp. 226–231, 2016, doi: 10.4236/eng.2016.84019.

[22] M. B. Mollah and M. R. Islam, "Comparative analysis of Gold Codes with PN codes using correlation property in CDMA technology," *2012 International Conference on Computer Communication and Informatics*, 2012, pp. 1-6, doi: 10.1109/ICCCI.2012.6158894.

[23] S. D. Cardell, A. Fúster-Sabater, and V. Requena, "Interleaving shifted versions of a PN-sequence," *Mathematics*, vol. 9, no. 6, pp. 1–23, 2021, doi: 10.3390/math9060687.

[24] E. G. Figueiredo, U. P. Clival, B. A. A. Rtery, N. R. Crawford, P. D, and E. T. Al, "C Omparative a Nalysis of a Nterior P

Etrosectomy," *Neurosurgery*, vol. 58, no. February, pp. 13–21, 2006.
[25] N. Chaitanya, R. Hrishikesh, S. Sb, L. S. Kartheek Raja and Y. N. Kumar Reddy, "Overview on CDMA networks and Implementation of CDMA Spread Spectrum Techniques using MATLAB," *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 2020, pp. 524-529, doi: 10.1109/ICCES48766.2020.9138015.

BIOGRAPHIES OF AUTHORS

Hadeer Hussein Ali was born in Baghdad, Iraq. she received her B.Sc. degree in Electrical Engineering in 2012 and his M.Sc. degree in Communication Engineering in 2016 and currently Ph.D. student degree in Electronic and Communication Engineering from University Technology, Iraq. Her research interest includes communication systems, SDR, multiuser MIMO. Now she has been lecturer at Al-Salam University College, Iraq. She can be contacted at email: hadeer.hussein2108@gmail.com.

Hadi T. Zaboon was born in Baghdad, Iraq. He received her B.Sc. degree in Electrical Engineering in 1973 and his M.Sc. degree in Communication Engineering in 1978 and Ph.D. degree in 1984 in Electrical and Communication Engineering from University of Aston, Birmingham, UK. He is Head of Al-Salam University College, Iraq. His research activities are electronic, controller, radar, communication systems, SDR, OFDM, MIMO systems, FBMC. Now he has been an Prof. at Al-Salam University College, Iraq. He can be contacted at email: hadi.t.ziboon@alsalam.edu.iq.

Ashwaq Q. Hameed was born in Baghdad, Iraq. She received her B.Sc. degree in Electrical Engineering in 1997 and his M.Sc. degree in Communication Engineering in 1999 and Ph.D. degree in 2003 in Communication Engineering from University of Technology, Baghdad, Iraq. She is faculty member of Electrical Engineering Department-University of Technology. Her research activities are mobile wireless communication systems, multicarrier OFDMA, Q-OSTFBC MIMO systems, speech signal processing, e-learning, radar systems. Co-author of three books: 1- ANFIS technique for Identification of Digitally Modulated Signals Using MATLAB, 2-Leaser Fundamentals and 3-Small and Special Electrical Motors and Their Control Technique. Now she has been an Assist. Prof. at the University of Technology, Iraq, Baghdad. She can be contacted at email: 50058@uotechnology.edu.iq.