IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 2, 2022, accepted May 16, 2022, date of publication May 20, 2022, date of current version May 26, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3176609

An Innovative FPGA Implementations of the Secure Frequency Hopping Communication System Based on the Improved ZUC Algorithm

WANG JINPENG[1], ZHANG TENG[1], ZHANG BO[1], JEREMY-GILLBANKS[2], AND ZHAO XIN[1], (Member, IEEE)

[1]School of Information Science & Engineering, Dalian Polytechnic University, Dalian, Liaoning 116034, China
[2]School of Electronic, Electrical and Computer Engineering, The University of Western Australia (M350), Perth, WA 6009, Australia

Corresponding author: Wang Jinpeng (wangjp@dlpu.edu.cn)

This work was supported in part by the Project of the National Natural Science Foundation of China under Grant 61402069; in part by the Fundamental Research Funds for the Central Universities under Grant 3132016317; in part by the 2017 and 2020 Projects of the Natural Science Foundation of Liaoning Province under Grant 20170540059 and Grant J2020018; in part by the General Project of Liaoning Education Department, in 2016, under Grant 2016J205; and in part by the General Project of the National Social Science Fund under Grant 2019AG00482.

ABSTRACT Frequency Hopping (FH) is a significant anti-interception and anti-interference, so some wireless communication systems extensively use it. Then it can offer better security performance of the system via combining cryptographic algorithms with FH. This paper presents a detailed theoretical design and hardware implementation of an evolutionary ZUC stream cipher algorithm on FPGA. Using some encryption algorithms can improve the security of the FH system by increasing its complexity, including SNOW3g, ZUC algorithm, Chaos system, and other cipher methods. ZUC (named for Zu Chongzhi, a famous mathematician in ancient China) is bitstream encryption that builds the core of the 3GPP integrity algorithm 128-EIA3 and the 3GPP confidentiality algorithm 128-EEA3, providing authentic security services in 4G, even in 5G (IMT-2020) to some extent. The article presented an evolutionary three-layers structure of the ZUC FH system, which innovatively applied the permutation polynomial algorithm and an evolutionary DES cipher algorithm. The introduction of that new DES and the permutation polynomial can efficiently increase the security performance of the ZUC system but also avoid some shortcomings of the UHF (Ultra High Frequency) communication. The experiment of this research designed software by coding VHDL language and implemented hardware using a XILINX Virtex-5 FPGA. The finished experiment gave, analyzed, and compared all results according to the performance of the proposed system and hardware resources. The tests for evaluating the randomness and security denote that the constructed sequences have better performance, including the randomness, linear complexity, and hamming correlations, and pass the NIST test finally. The experimental results show that this improved ZUC cipher stream algorithm proposed in the article can provide a high throughput of 2.08 Gbps under a 65MHz clock frequency. The authors will focus their future work on applying the evolutionary S-box and the algorithm with better security performance.

INDEX TERMS Frequency hoppin, ZUC, FPG, NIS.

I. INTRODUCTION

The 5G or 6G mobile cellular networks require new frequency bands and techniques to meet the explosively fast growth of mobile data traffic [1]. Only to keep using the current microwave bands cannot meet these requirements, and the millimeter-wave (mm-Wave) technique should be used [2], [3]. Although some researchers argue that mm-Wave may not be reasonable for the cellular network owing to poor penetration through obstacles and huge near-field loss, a few implemented measurements have shown the contrary results [4]. As found that mm-Wave cellular communications are feasible for those cells with radii on the order of 150-200 meters in densely deployed, provided that they

The associate editor coordinating the review of this manuscript and approving it for publication was Jiankang Zhang.

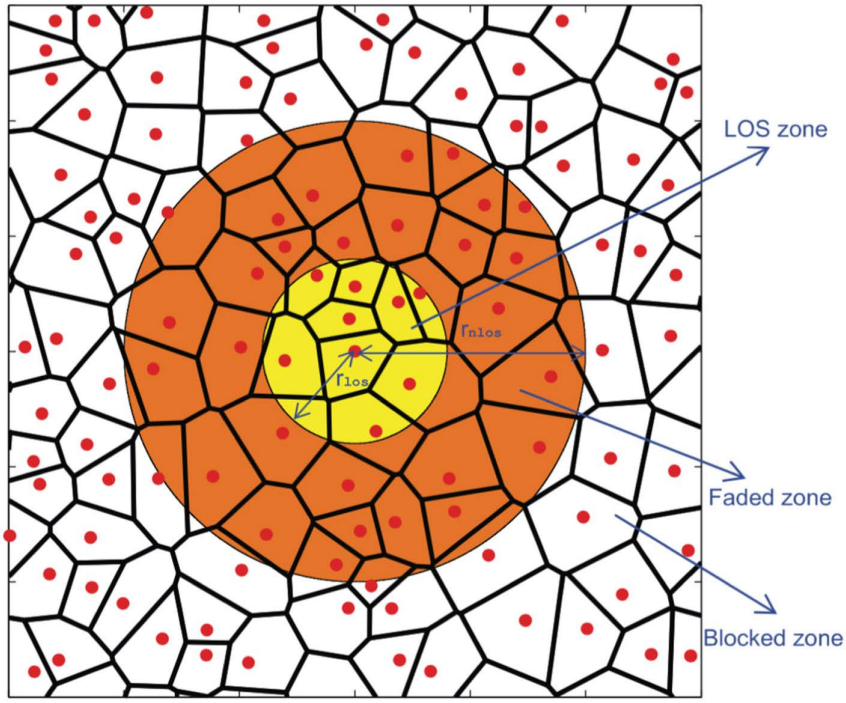


FIGURE 1. A typical wireless communication network. Red dots represent the BSs, while those tick-black lines symbolize the polygon cells. This network is made up of three specific areas: a) LOS area is the inner circle; b) Faded or NLOS area is those annular regions with orange color; c) Blocked area is those regions which is outside the colored circle.

have sufficient support by enough beamforming gain between the mobile ends and the BSs (Base Stations) that they serve [5].

Figure 1 shows a typical network. There are BSs in the mode and some mobile nodes surrounded via a polygon cell. Y_j symbolizes the j^{th} base station and its position. If the referenced BS is Y_0 and the coordinate system is selected, it locates at the origin, i.e., $Y_0 = 0$. To place randomly other BSs of the network. For example, these BSs can be placed in position based on some known places of an existing network or synthesized through one stochastic-geometry model. In the following parts, when the distribution of the BSs must be in use, the authors assumed that they were from a Uniform Clustering Process (UCP) [11] of the intensity λbs. Each BS can be around one by one exclusion zone with a radius *of* $r_{\min}$.

With the rapid development of modern communication technology and electronic equipment, communication is more convenient and faster. At the same time, it is also facing more and more interference and threats [6], [7]. Therefore, frequency hopping (FH) communication has become one of the significant directions of the development of modern communication technology because of its good anti-interference performance and multiple access networking ability [8]. It has wide usage in many fields, such as Bluetooth [9], smart home [10], etc. Compared with the traditional communication with a fixed carrier frequency, the transmitter and receiver of FH communication change the carrier frequency through a group of pseudo-random sequences synchronously. This group of pseudo-random sequences is called FH sequences. The design of FH sequences plays a decisive role in the security performance of the communication system.

With a lot of 5G (IMT-2020) used instances have emerged in the last two years, research on the 5G physical layer is an expected topic on multiple access technologies which can offer reliability, low latency (Autonomous vehicles), and Enhanced mobile bandwidth (Gigabit/second bandwidth) and user density (IoT applications). Attributed to progress in SIC (Successive Interference Cancellation) non-orthogonal multiple access (NOMA) techniques, particularly Multiuser Share Access (MUSA) and Sparse Code Multiple Access (SCMA), have been considered as a leading candidate to achieve high user density [11].

The main idea of the NOMA system is that multiple users can simultaneously apply the same Resource Element (RE, i.e., time-frequency resource) and then be separated using SIC. Furthermore, a short spreading code usually from a codebook belongs to available codes is assigned for every user in SCMA. Because of the large number of users and short-codes, it is too hard to implement completely orthogonal codes (which the specific design in a conventional CDMA system). Introducing FH into SCMA [12] is another method to make SIC much more efficient and decrease interference. Traditional methods utilize the FH multiple Access technologies for combatting narrow-band fading. Its idea is about FH introducing sufficient frequency separation between those

consecutive time slots so that it is bigger than the coherence bandwidth and constructing an independent fading mobile communication environment. Virtually, each hop should be greater than the coherence bandwidth once the time slots are definitely within the coherence time [13].

As a crucial part of the 128-EIA3 and the 128-EEA3, using the ZUC algorithm can provide integrity and confidentiality in the wireless transmissions of the LTE. The GSM association and 3GPP have developed this kit of protocols.

The regulation of the ZUC algorithm was first published on June 18, 2010 [14]. To ensure the security of the ZUC algorithm, it was in evaluation via the global public assessment and modified during initialization on January 4, 2011 [15], [16]. The revised version of the ZUC was not open for public evaluation until June 2011.

Nowadays, the design and implementation of hardware for cryptographic methods play a significant role in many application areas because of its good performance of the high throughput. Therefore, if the cryptographic algorithms can be implemented on hardware and gain greater efficiency or not has been become an important and hot topic [17]–[20]. Presently, there is not much research on hardware implementations of the ZUC. In many cases, the initial file provided by designers of the submitted algorithm only includes the software implementation of the ZUC. Because the LFSRs (Linear Feedback Shift Registers) are pretty efficient on hardware and are the main building blocks of the cipher, it is possible to implement the ZUC on hardware and achieve higher throughput.

Nowadays, the hardware implementation for cryptographic algorithms plays a significant role in some application areas owing to its high throughput. Therefore, whether a cipher method can be implemented in hardware and obtain greater efficiency becomes an important and hot topic [21]–[23]. By now, there are seldom research results on hardware implementation for ZUC. Sometimes, the original files gotten from designers of the published algorithms only include the software implementation for ZUC. Because the LFSRs (Linear Feedback Shift Registers) are very efficient in hardware and the main building parts of the cipher, it is possible and feasible to implement ZUC in hardware and obtain higher throughput.

Nearly, people started a period of a widespread upsurge in the study of chaos theory. Chaos is a physical phenomenon sensitive to initial values with high complexity. Therefore, people have shown great interest in the chaos in the past decades. Chaos [24] has gradually developed from the climate to other fields, such as information science, biology, engineering, finance, etc. So far, under the research of people, the chaotic system has been widely developed, from the original Lorentz system, to produce many different chaotic systems, including hyperchaotic system [25], [26], discrete chaotic system [27]–[29], memristor, chaotic system [30] and so on. Using chaos can improve the complexity of the FH system.

Some current RIP (Radio Interface Protection) algorithms for LTE [31], 128-EEA1 for confidentiality, as well 128-EIA1 for integrity have been contrived by the SAGE (Security Algorithms Group of Experts)/ETSI [32]. 128-EEA1 and 128-EIA1 are according to SNOW3G stream cipher [33], [34]. In addition, the GSM Association, together with the 3GPP, 3rd Generation Partnership Project, designates the second set of algorithms, 128-EEA2 and 128-EIA2 [35], based on AES block cipher [36], [37]. At last, the GSM association with 3GPP specifies the third set of algorithms for integrity and confidentiality, including 128-EIA3 and 128-EEA3, respectively [38]. Both two ciphers are according to the ZUC stream cipher [39]. The key reason for the new ciphers is that many countries will utilize the LTE. However, Chinese regulations do not accept these algorithms used in China because they were not products of China. Nevertheless, China allows the ZUC to be in China.

The traditional sequence construction methods for the FH system, such as m sequence, RS code sequence, and other ways, have their disadvantages of low complexity. Of course, there are several FH sequences proposed in recent years, just like the chaotic system and the improved m sequence. Although these presented new algorithms have overcome some shortcomings of the conventional method and have high complexity, they have disadvantages. The chaotic frequency hopping needs an introduction disturbed by an m sequence, and the improved m sequence requires its more complicated generated hardware equipment.

Therefore, this paper proposed an evolutionary three-layers structure of the ZUC FH system, which innovatively applied the permutation polynomial algorithm and an evolutionary DES cipher algorithm. Furthermore, using the hardware-based on FPGA can implement this presented algorithm with an acceptable medium complexity.

In this article, the authors take the merits of the properties of LFSRs for implementing the novel proposed stream cipher ZUC on an FPGA (Field Programmable Gate Array) and an attempt to optimize this implementation. We presented an evolutionary three-layers structure of the ZUC FH system which innovatively applied the permutation polynomial algorithm and an evolutionary DES cipher algorithm. The introduction of that new DES and the permutation polynomial can efficiently increase the security performance of the ZUC system but also avoid some shortcomings of the UHF communication. The experimental results gained in this paper denote that this ZUC implementation is a resilient, flexible and robust solution for the 4G of LTE usage, even for 5G to some extent. The implementation of the FPGA presented in our experiment can achieve a 2.08 Gbps throughput under a clock frequency of 65 MHz.

The arrangement of the rest of the paper is as follows: Section 2 shows the Fundamentals of the ZUC Algorithm; section 3 gives the newly proposed evolutionary three-layers structure of the ZUC FH system in this article; Section 4 shows the implementations of the proposed ZUC Cipher Algorithm on FPGA in detail; Section 5 presents the experimental Results, Analysis, and Comparisons; At last, Section 6 describes the whole conclusions of this research.

II. FUNDAMENTALS OF THE ZUC ALGORITHM

ZUC is a word-oriented stream cipher firstly presented via the Data Assurance and DACAS (Communication Security Research Center) of the Chinese Academy of Sciences [40]. It can take a 128-bit Initial Key (IK) and a 128-bit Initial Vector (IV) as input and a Key-stream of the 32-bit words as outputs (so each 32-bit word is called a key-word in it). The Key-stream can adapt to encrypt the plain text.

DES, Data Encryption Standard, is a block algorithm determined as FIPS (Federal data Processing Standard) by the National Bureau of Standards of the federal government of America. DES algorithm using key encryption was widely in use internationally after it authorized utilization in unclassified government communications. In some types of literature, to be different from DES as a standard, Des is also known as DEA (Data Encryption Algorithm) when it is an algorithm.

Based on the specification of the ZUC [41], ZUC contains three logical layers (as shown in Figure 2). The bottom layer is a procedure of a nonlinear function F; the middle layer is a process of the BR (Bit-Reorganization), and the top layer is an LFSR (linear feedback shift register) of 16 stages.

A. THE PROCEDURE OF AN LFSR WITH 16 STAGES

The LFSR (Linear Feedback Shift Register) has 16 31-bit registers $(S_0, S_1, \cdots, S_{15})$, and every memory $S_i(0 \leq i \leq 15)$ is limited to get the value from the following dataset: $\{1, 2, 3, \ldots, 2^{31} - 1\}$. The LFSRs have two modes of operations, including the work mode and initialization mode.

As shown in Algorithm 1 below, the LFSR doesn't receive any input in the work mode. It denotes that other parts of the ZUC work independently with the LFSR, which enlightens the authors that when we acquire S16 for each clock pulse, the shift register operates shifts for every clock cycle, which means that it can produce a 32-bit key per clock cycle.

Algorithm 1 The LFSR in the Work Mode

{Begin
$S_{16} = \{2^{15}S_{15} + 2^{17}S_{13} + 2^{21}S_{10} + 2^{20}S_4 +$
$(1 + 2^8)S_0\}$
$mod(2^{31} - 1)$; //Generate S_{16}.
If $S_{16} = 0$ then
set $S_{16} = 2^{31} - 1$; //
Initialization again.
else
$(S_1, S_2, \cdots, S_{15}, S_{16}) \rightarrow$
$(S_0, S_1, \cdots, S_{14}, S_{15})$
Endif;
End;}

NOTE Please: All denotations "//" of the tables in this paper mean an explanation of the program before it.

In the initialization mode, the LFSR first receives a 31-bit input word u achieved by getting rid of the right-most bit from 32-bit output W of that nonlinear function F ($u = W \gg 1$). As shown in Algorithm 2 below, the mode of the initialization can work as the following steps:

Algorithm 2 The LFSR in the Initialization Mode

{Input: u
Begin
$v = \{2^{15}S_{15} + 2^{17}S_{13} + 2^{21}S_{10} + 2^{20}S_4 + (1+$
$2^8)S_0\} mod(2^{31} - 1)$; //Set v.
If $S_{16} = 0$ then
set $S_{16} = 2^{31} - 1$; // Initialization
again.
else
$(S_1, S_2, \cdots, S_{15}, S_{16}) \rightarrow (S_0, S_1, \cdots, S_{14}, S_{15})$
Endif;
End;}

B. THE PROCESS OF THE BR (BIT-REORGANIZATION)

The middle layer of the ZUC is the process of the BR (Bit-Reorganization). And assuming that the eight registers of the LFSR stand for respectively S_0, S_2, S_5, S_7, S_9, S_{11}, S_{14}, *and* S_{15}, etc.

Algorithm 3 The BR Process

{Begin
$X_0 = S_{15H}||S_{14L}$; // The assignment to the first 32-bits word X_0.
$X_1 = S_{11L}||S_{9H}$; // The assignment to the second 32-bits word X_1.
$X_2 = S_{7L}||S_{5H}$; // The assignment to the third 32-bits word X_2.
$X_3 = S_{2L}||S_{0H}$; // The assignment to the fourth 32-bits word X_3.
End;}

Based on Algorithm 3 above, the BR generates four words with 32-bits: X_0, X_1, X_2, and X_3. Then the former three words are passed into the next layer, i.e., the bottom layer (the nonlinear function F). For realizing the concatenation of signals, in contrast to software implementations, the operation on hardware works easy, only changing the order of the wires, which costs little time to accomplish. Therefore, the BR process should better operate with the nonlinear function F together to save clock cycles.

C. THE PROCEDURE OF THE NONLINEAR FUNCTION F

There are two 32-bits registers R_1, R_2, two 32-bits logical gates of X-NOR, two 32-bits modulo 2^{32} adders in the Nonlinear Function $\boldsymbol{F}$. In addition, Nonlinear Function $\boldsymbol{F}$ also has three inputs X_0, X_1, X_2, and one 31-bits output W. Those inputs come from the outputs of the last layer, i.e., the process of the BR. Finally, it has a left cyclical shifter of 16 positions, two 32×32 S-boxes, as well as two linear transformation functions L_1, L_2. Algorithm 4 describes the process of nonlinear function F in detail as below:

In Algorithm 4 above, S is an S-box with 32×32; L_1 and L_2 are linear transformations defined and written as Equations (1), (2), as shown at the bottom of the next page, respectively.

In equations (1, 2) above, the critical path is the computation of $W_1 = R_1 \boxplus X_1$ in the procedure of Nonlinear function $\boldsymbol{F}$, where $\boxplus$ represents a modulo 2^{32} addition. The rest

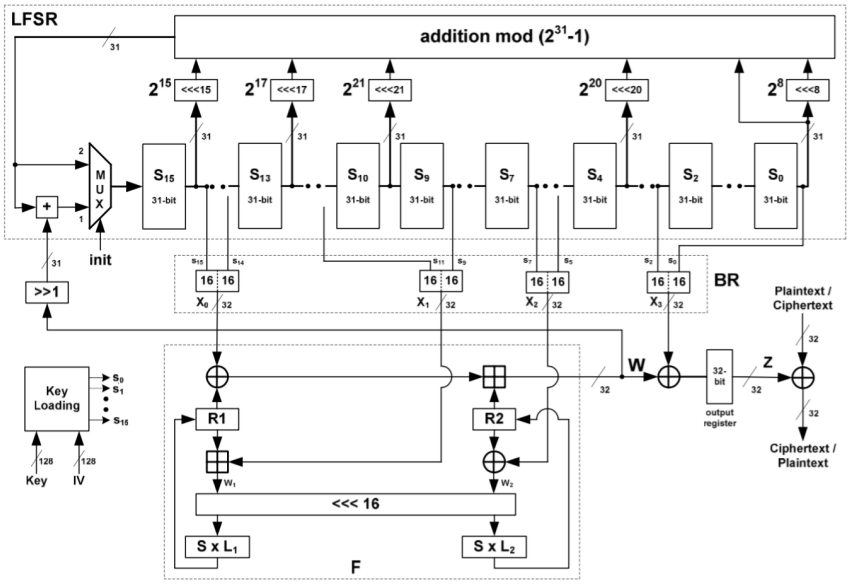


FIGURE 2. The structure of a traditional ZUC algorithm.

Algorithm 4 The Procedure of the Nonlinear Function $\boldsymbol{F}$

{Input: X_0, X_1, X_2
Begin
$W = (X_0 \oplus R_1) \boxplus R_2$; // W is the 32-bits output word of $\boldsymbol{F}$;
$W_1 = R_1 \boxplus X_1$; //Operator $\boxplus$ is modulo 2^{32} adder;
$W_2 = R_2 \oplus X_2$; // Operator $\oplus$ is the XOR-gates;
$R_1 = S\,(L_1\,(W_{1L}||W_{2H}))$; // R is the is the 32-bits register;
$R_2 = S\,(L_2\,(W_{2L}||W_{1H}))$; // Operator || is the OR-gates;
End;

operations of the Nonlinear Function $\boldsymbol{F}$ are negligible because they cost little time in contrast to this 2^{32} modulo adder. Therefore, the authors suppose that finishing the operation of the BR and the Nonlinear Function F simultaneously in a clock cycle. It denotes ZUC can produce a 32-bits key in each clock cycle once the LFSR completes the update per clock cycle.

D. THE IMPLEMENTATION OF ZUC

The implementation of ZUC includes two steps: the first step for initializing and the second step for working. Within the initializing step, to complete this initialization, the cipher method operates the following stages 32 times:

1. BR (); // Call Algorithm 2: Bit-Reorganization.
2. $w = F(X_0, X_1, X_2)$; // Assignment for w.
3. LFSR in the Initialization mode ($w \gg 1$); // Call Algorithm 2.

When accomplishing the first step of initializing, this algorithm will go into the second step for the working stage. During the beginning of this step, the algorithm throws away output W of the Nonlinear Function F and operates the following process once:

1. BR (); // Call Algorithm 2: Bit-Reorganization.
2. $F(X_0, X_1, X_2)$; // Assignment for w.
3. The LFSR in the work mode (); // Call Algorithm 1.

After finishing the two steps above, the algorithm runs into a new stage for generating a key-stream, which means the method operates the following steps once for every iteration and generates output with 32-bits word:

1. BR (); // Call Algorithm 2: Bit-Reorganization.
2. $Z = F(X_0, X_1, X_2)$; // Assignment for Z.
3. The LFSR in the work mode (); // Call Algorithm 1.

Expression (3) below may be the most time-consuming part of the LFSR process, so the critical path of the LFSR procedure is about the computation of expression (3):

$$\left\{2^{15}S_{15} + 2^{17}S_{13} + 2^{21}S_{10} + 2^{20}S_4 + \left(12^8\right)S_0\right\} mod\left(2^{31}-1\right) \quad (3)$$

In the expression above, S is an S-box with 32 × 32; L1 and L2 are linear transformations defined before and written as Equations (1) and (2).

The efficiency of the ZUC execution on FPGA relies upon costs of the significant ways of the whole method, i.e., those

$$L_1(X) = X \oplus (X <<<_{32} 2) \oplus (X <<<_{32} 10) \oplus (X <<<_{32} 18) \oplus (X <<<_{32} 24) \quad (1)$$

$$L_2(X) = X \oplus (X <<<_{32} 8) \oplus (X <<<_{32} 14) \oplus (X <<<_{32} 22) \oplus (X <<<_{32} 30) \quad (2)$$

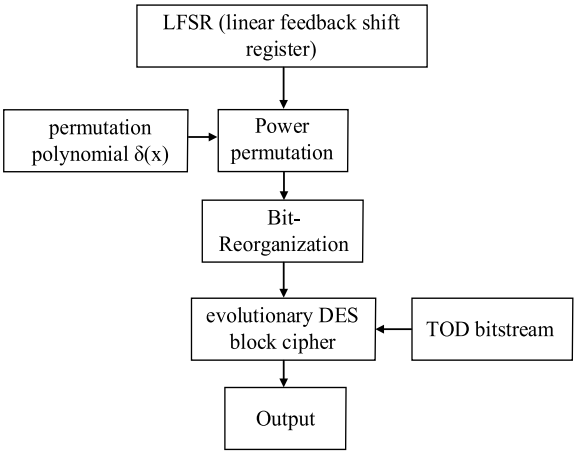


FIGURE 3. The proposed evolutionary three-layer architecture of the ZUC FH system. NOTE Please: here, TOD (Transit-Oriented Development) is 64 bits.

maximum values of the critical path between that of the procedure of BR, Nonlinear Function F, and LSFR. As for Equation (3), there are five modulo $(2^{31}-1)$ additions while implementing an update operation in the procedure of the LFSR per time. Without any optimization over the computation of Formula (3), although completing an update operation in the process of the LFSR per clock cycle, this critical path tends to be too long.

This 32 × 32 S-box S consists of four mini S-boxes with 8 × 8, which means that S = (S_0, S_1, S_2, S_3) and $S_0 = S_2$, $S_1 = S_3$. An official encryption specification has definitions of the S_0 and S_1. Both the L_1 and L_2 are linear transformations with 32-bit words to 32-bit words.

As for a cipher operation, the encryption key loading process firstly expands the initial vector and the initial key into 16 of 31 bits integers as an initial stage of the LFSR, then implements the initialization step and working mode. The first stage operates an initialization for IK or IV and clocks the cipher with no generating output. Whereas, in the work mode, the second step generates a 32 bits word of outputting signal for every clock cycle.

III. PRESENTED ARCHITECTURE

A. THE NEWLY PROPOSED EVOLUTIONARY THREE-LAYERS STRUCTURE OF THE ZUC FH SYSTEM

As shown in Figure 3, this paper improves the three-layer architecture based on the Structure of the traditional ZUC Algorithm mentioned in Section II as follows: 1. the method introduces a permutation polynomial δ (x) in the first layer; 2. the middle layer keeps no change; 3. The third layer utilizes an evolutionary DES cipher algorithm, and it can better ensure security using this more updatable and secure evolutionary S-box.

Using a TOD bitstream describes the file containing the complete internal configuration status of the FPGA, including wiring, logical resources, and I/O settings. There are two steps during FPGA power on or the subsequent FPGA reconfiguration: 1. Reading the TOD bitstream is from external nonvolatile memory such as the flash memory; 2. loading it into the internal configuration SRAM (Single address Radom Accessing Memory) is through using the processing of an FPGA configuration controller. Therefore, this paper needs a TOD bitstream as the input to be a restriction condition for boosting the function of DES.

Compared with traditional methods, as shown in figure 3 above, there are two improved parts in the proposed structure: one is the introduction of the permutation polynomial, and the other one is to evolute the conventional DES cipher algorithm.

B. THE PERMUTATION POLYNOMIAL

Once a sequence only possesses a low linear complexity, attackers can easily reconstruct the shortest length of the LFSR and the feedback logic of this sequence via using a simple algorithm just like the BM (Boyer Moore) [10]. Consequently, an FH sequence must have a higher linear complexity to ensure its performance of safety.

The permutation polynomials δ (x) on finite fields can be applied to improve this linear complexity to increase the linear complexity of an FH sequence. During the stage of use of permutation polynomial δ (x), a (Q + 1)/2 power operation is added to the original sequence set (i.e., the increased multiplication times are about log2((Q + 1)/2), and then added to the original sequence. This type of permutation polynomial can not only maintain the optimal Hamming correlation of the transformed sequence but increase the linear complexity of the sequence. Therefore, the set of FH sequences obtained in this way has high linear complexity and is easy implementation in engineering.

Let p be an odd prime number, $q = p^r$, where r is a positive integer. Suppose α is a generator of GF $(q^m)^*$ (where GF is a new signature function defined in the Galois field.), odd number m ≥ 3, $n = (q^m - 1)/2$, and integer d satisfy gcd $(d, q^m - 1) = 1$. If $\beta = \alpha^{2d}, \forall\alpha \in GF(q^m)$, then the sequence $\boldsymbol{S}_a$ can be defined as follows:

$$\boldsymbol{S}_\alpha = \left(\mathrm{Tr}(\alpha), \mathrm{Tr}(\alpha\beta), \cdots, \mathrm{Tr}\left(\alpha\beta^{n-1}\right)\right) \tag{4}$$

In equation (4), Where $\mathrm{T_r}(x) = x + x^q + \cdots + x^{q^{m-1}}$ is the trace function, GF $(q^m) \longrightarrow$ GF (q); $\boldsymbol{S}_\alpha$ is an optimal FH sequence with $\left((q^m - 1)/2, \left(q^{m-1} - 1\right)/2; q\right)$, and its linear complexity is m. The complexity of this sequence is pretty low due to its period of $(q^m - 1)/2$.

However, Permutation polynomials can produce FH sequences with higher complexity. Using equation (4) can obtain the sequence $\boldsymbol{S}_\alpha$ if $b = \left(c^2 + 1\right)\left(c^2 - 1\right), c \in$ GF (q), $c \neq 0, c^2 \neq 1$, and assuming $\delta(x) = x^{(q+1)/2} + bx$, then:

$$\delta(S_\alpha(t)) = \mathrm{Tr}\left(\alpha\beta^t\right)^{\left(\alpha\beta^t\right)q+\frac{1}{2}} b\mathrm{Tr}\left(\alpha\beta^t\right), \quad 0 \leq t \leq \left(q^m - 1\right)/2 - 1 \tag{5}$$

Based on equation (5), together with three given conditions, including '$q = p^r$,' '$(q+1)/2$ can be written as $(q+1)/2 = \sum_{i=0}^{r-1} \eta_i p_i$,' and '$0 \leq \eta_i < p$,' expression (6)

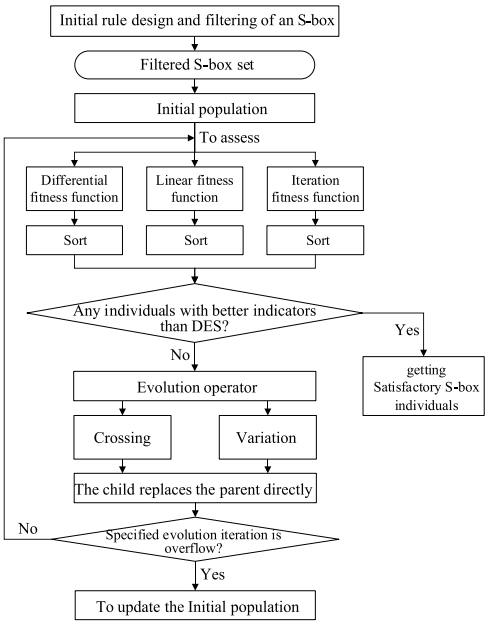


FIGURE 4. A flowchart for the evolutionary strategy of the S-box.

can be derived:

$$\mathrm{Tr}\left(\alpha\beta^t\right)^{\frac{q+1}{2}} = \left(\sum_{j=0}^{m-1}\left(\alpha\beta^t\right)^{q^j}\right)^{\frac{q+1}{2}} = \prod_{i=0}^{r-1}\left(\sum_{j=0}^{m-1}\left(\alpha^{p^i}\beta^{p^i t}\right)^{q^j}\right)^{\eta_i} \quad (6)$$

Furthermore, since the formula

$$\sum_{\lambda_{i,0}+\lambda_{i,1}+\cdots+\lambda_{i,m-1}} \frac{\eta_i!}{\lambda_{i,0}!\lambda_{i,1}!\cdots\lambda_{i,m-1}!}\left(\alpha^{p^i}\beta^{p^i t}\right) \times \sum_{j=0}^{m-1} q^j\lambda_{i,j}$$

can simplify as $\left(\sum_{j=0}^{m-1}\left(\alpha^{p^i}\beta^{p^i t}\right)^{q^j}\right)^{\eta_i}$, therefore, expanding equation (6) above can get expression (7), as shown at the bottom of the next page.

If considering $f(x)$ as a characteristic polynomial, its root is a linear sequence $a = a_0 \cdot a_1 a_2 \cdots, a_n$ and only if there is a set of coefficients $\lambda_1, \lambda_2, \cdots, \lambda_n$ such that $a = \lambda_1\alpha_1^k + \lambda_2\alpha_2^k + \cdots + \lambda_n\alpha_n^k$, then the linear complexity of this sequence equals the number of non-zero coefficients in equation (7).

Therefore, it is necessary to calculate the number of the different modules q^{m-1} of coefficient β in equation (7). According to the different values of $\lambda_{i,j}$, $\lambda'_{i,j}$, there is an expression as below:

$$\sum_{j=0}^{m-1} q^j \sum_{i=0}^{\gamma-1}\lambda_{i,j}p^i = \sum_{j=0}^{m-1} q^j \sum_{i=0}^{\gamma-1}\lambda'_{i,j}p^i \bmod (q^m-1) \quad (8)$$

Because $\lambda_{i,j} \leqq \eta_i$ and $\lambda'_{i,j} \leqq \eta_i$, thus while $q > 3$, there is an Inequality $0 < \frac{(q+1)}{2} = \sum_{i=0}^{\gamma-1}\eta_i p^i < q-1$, which means both $\sum_{j=0}^{m-1} q^j \sum_{i=0}^{\gamma-1}\lambda_{i,j}p^i$ and $\sum_{j=0}^{m-1} q^j \sum_{i=0}^{\gamma-1}\lambda'_{i,j}p^i$ are less than $q^m - 1$. Consequently, after eliminating $\mathrm{mod}(q^m - 1)$, expanding equation (8) and taking the operation of the modulo q on both sides to achieve the expression below:

$$\lambda_{0,0} + \lambda_{1,0}p + \cdots + \lambda_{\gamma-1,0}p^{\gamma-1} = \lambda'_{0,0} + \lambda'_{1,0}p + \cdots + \lambda'_{\gamma-1,0}p^{\gamma-1} \bmod q \quad (9)$$

As seen from equation (9) that both sides of it are much less than q, so eliminating the $\mathrm{mod}\, q$ can get $\lambda_{\gamma-1,0} = \lambda'_{\gamma-1,0}$. Because the coefficients β of equation (7) are different from each other, thus the power of β in the sequence $b\mathrm{T_r}\left(\alpha\beta^t\right)$ and the coefficients β of equation (7) are also different from each other. Then, this paper calculates the total number of possibilities of different coefficients through a formula of the combination number, the linear complexity of $\delta(s_a)$ is as expression (10) below:

$$\begin{bmatrix} m + \dfrac{(p+1)}{2} - 1 \\ \dfrac{(p+1)}{2} \end{bmatrix}\begin{bmatrix} m + \dfrac{(p-1)}{2} - 1 \\ \dfrac{(p-1)}{2} \end{bmatrix}^{\gamma-1} + m \quad (10)$$

In equation (10) above, the square bracket is a formula of the combination number, i.e., $\begin{bmatrix} m \\ n \end{bmatrix}$: $C_n^m = \frac{n!}{m!(n-m)!}$.

Consequently, using permutation polynomial $\delta(\mathrm{x})$ can efficiently increase the linear complexity of the optimal FH sequence with low complexity.

C. THE ALGORITHM OF THE EVOLUTIONARY DES BLOCK CIPHER

Commonly, there are two principles of a block cipher design to resist the statistical analysis from other's encryption systems, including confusion and diffusion. Principle Confusion is to make a relationship between the statistical characteristics of the ciphertext and the value of the key as complex as possible; principle Diffusion is to act the influence of each plaintext to more output ciphertext bits as quickly as possible and make the influence of each key expand to more ciphertext bits as fast as possible.

Due to the advantages of the algorithm of the DES cipher, including its fast encryption, decryption speed, and good security [55], this method is a well-known block cipher by now widely used in many fields and industries that need encryption. Its security performance mainly depends on the core component S-box. Its security performance primarily depends on the core component S-box. S-box is the only nonlinear substitution transformation component in DES cipher. This S-box plays a role in confusion or complication in the encryption process, so to increase security.

As shown in figure 4, the overall strategy of evolutionary S-box design is: first, to use the existing design criteria, especially those with clear quantitative regular, to design and generate the initial population and reduce the sample space of the S-box; second, to filter and sort the randomly chosen initial population based on the pre-determined cryptographic indexes; Lastly, through the various indexes, the attained optimal or non-inferior solutions should undergo an evolutionary

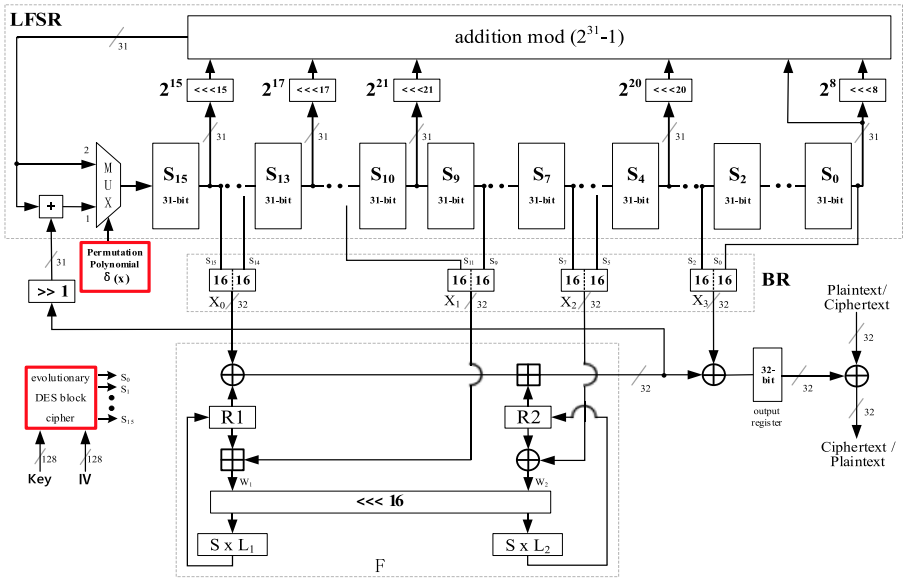


FIGURE 5. The proposed architecture of the ZUC Cipher algorithm on FPGA.

compromise. If this compromise is at fault, the algorithm will begin a new round of evolution until obtaining the demanded S-box. Figure 4 shows the evolution process.

The adopted variation strategy in the evolutionary process are listed as follows:

(1). Randomly change the arrangement order of two s-boxes;

(2). Randomly select an S-box for column transformation or row transformation;

(3). Randomly rearrange the arrangement order of eight S-boxes.

The adopted crossing strategy in the evolutionary process are listed as follows:

(1). Randomly generate an 8-bit binary data and exchange two S-boxes by determining whether one of these bits is 1.;

(2). Generate three random numbers of less than 8: N1, N2, and m, and exchange the data between N1 and Ni + m of the first S-box and N2 to N2 + m of the second S-box.

Moreover, the algorithm will put new individuals into the group for every specific evolutionary algebra to better search in the global range.

IV. IMPLEMENTATIONS OF ZUC CIPHER ALGORITHM ON FPGA

As researched in Section II before, Figure 2 shows that the traditional ZUC algorithm has an architecture of three layers. The upper layer is a 16 level LFSR on a finite field GF (2^{31}-1); The bottom layer is a procedure of a nonlinear function $\boldsymbol{F}$; the middle layer is a process of the BR. In this part, implementations of the presented evolutionary structure on FPGA in this paper (as shown in figure 5) are listed in detail as follows:

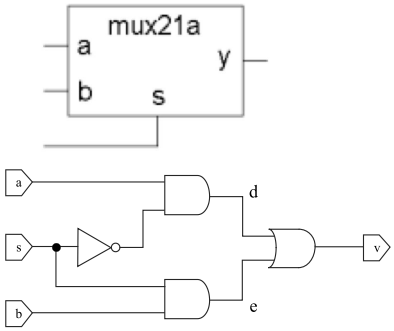


FIGURE 6. The circuit of a Multiplexer.

A. THE IMPLEMENTATION OF THE PROPOSED ZUC CIPHER ALGORITHM ON FPGA

This study explored whether the ZUC cipher works on a current hardware device or not for efficient application over LTE networks. Figure 5 illustrates the proposed hardware architecture of the ZUC Cipher Algorithm on FPGA.

The presented system has as basic I/O interfaces a 32-bits cipher-text/plaintext output and a 32-bits plaintext/cipher-text input. Additionally, the system possesses two inputs, a key, a 128-bits initialization value, and IV, a 128-bit secret key, and upholds the initialization step, the working step, and the key-stream producing stage.

$$\sum_{\sum_{j=0}^{m-1}\lambda_{0,j}=\eta_0}\cdots\sum_{\sum_{j=0}^{m-1}\lambda_{\gamma-1,j}=\eta_{\gamma-1}}\prod_{i=0}^{\gamma-1}\left(\frac{\eta_i!}{\lambda_{i,0}!\lambda_{i,1}!\cdots\lambda_{i,m-1}!}\alpha\sum_{j=0}^{m-1}q^j\sum_{i=0}^{\gamma-1}\lambda_{i,j}p^i\beta^{\left(\sum_{j=0}^{m-1}q^j\sum_{i=0}^{\gamma-1}\lambda_{i,j}p^i\right)^t}\right) \quad (7)$$

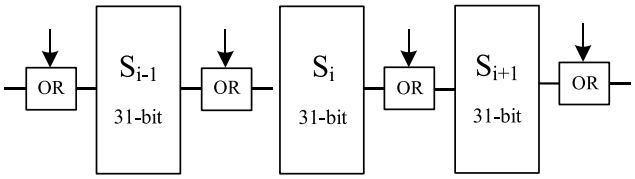
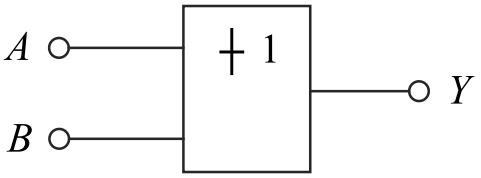
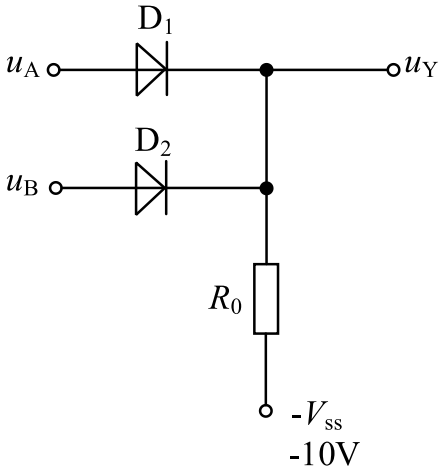


FIGURE 7. The linking component 31-bits OR-gates among the LFSR cells.



(a). the ENTITY of a 31-bits OR gate.



(b). the ARCHITECTURE of a 31-bits OR gate.

FIGURE 8. The circuit of a 31-bits OR gate (i.g., one output from two inputs).

B. THE IMPLEMENTATION OF THE 31-BITS MULTIPLEXER

The Data Selector (also known as Multiplexer, i.e., MUX) is a device that can choose a signal as the output from multiple analog or digital signals in electronic technology, especially in the digital circuit. Using this 31-bit selector is corresponding to its application area of LFSR module. A Multiplexer with 2^n inputs has n selectable input-outputs and can select one of these signals as output through a control terminal. Using the Multiplexer mainly can increase the amount of the sending data through a network within a certain amount of bandwidth and period. This data selector enables multiple signals to share a common resource or device, just like a transmission line or an A/D (analog-to-digital) converter, which does not need to equip a device for each inputting signal. Figure 6. shows the circuit of a Multiplexer (taking a One out of two multi-channel Multiplexer as an example, i.e., MUX21a). Using the Multiplexer mainly can increase the amount of the sending data through a network within a certain amount of bandwidth and period.

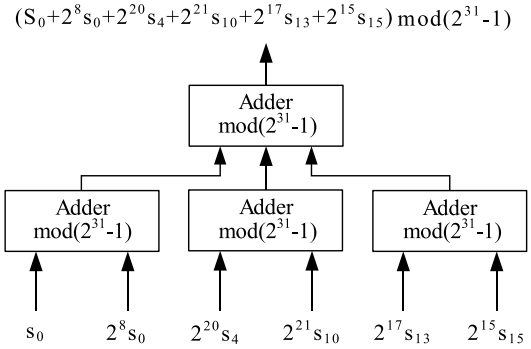


FIGURE 9. The implementation of the feedback logic circuit.

C. THE IMPLEMENTATION OF THE 31-BITS OR GATE AMONG THOSE LFSR CELLS

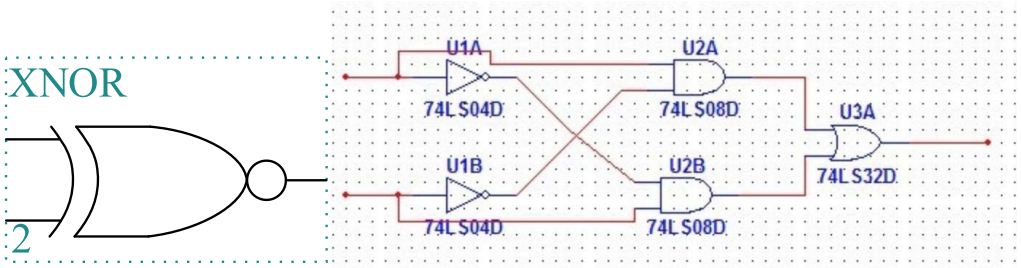
Firstly, the key loading part utilizes a 240-bit D constant, $D = d_0||d_1||\ldots||d_{15}$ (where $0 \le d_i \le 15$ are predefined), and then together with IV as well Key, generates 16 31-bit substrings based on a specific rule of $s_i = k_i||d_i||iv_i$ (where $0 \le i \le 15$ is also predefined). The iv_i and k_i are the 16 bytes of the IV and the Key, where iv_0 and k_0 are the most important values. The 31-bits LFSR applies these substrings as the initial value of the cells $s_0, s_1, \ldots, s_{15}$, respectively. As shown in Figure 7 below, the LFSRs can parallel load the substrings as their initial values through the OR- gates. The OR- gates are forced by zeros while fetching these values.

An OR gate, also known as an OR circuit or logic sum circuit, is a logic device, which means if one of several conditions is satisfied, an event will occur, we usually also call it "or" logic relationship. A digital circuit with an "or" logic relationship has multiple inputs and one output. Its output is a high level (i.e., logic "1") as long as one of these inputs is the high level, as well its output is a low level (i.e., logic "0") only when all inputs are the low level. Figure 8. gives the implementation of ENTITY and ARCHITECTURE of the OR gate based on the FPGA of the CPLD (Complex Programmable Logic Device) as below:

Algorithm 5 The Procedure of the Nonlinear Function $\boldsymbol{F}$

```
{LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY or2a IS
    PORT (a, b: IN STD_LOGIC;
          c: OUT STD_LOGIC);
    END ENTITY or2a;
ARCHITECTURE one OF or2a IS
    BEGIN
     c <= a OR b;
END ARCHITECTURE one;
```

Besides Figure 8, algorithm 5. This research also programs the OR gate using VHDL (Verilog Hardware Description Language.



(a). the ENTITY of a 31-bits XOR gate. (b). the ARCHITECTURE of a 31-bits XOR gate.

FIGURE 10. The circuit of a 31-bits XOR gate (i.e., one output from two inputs).

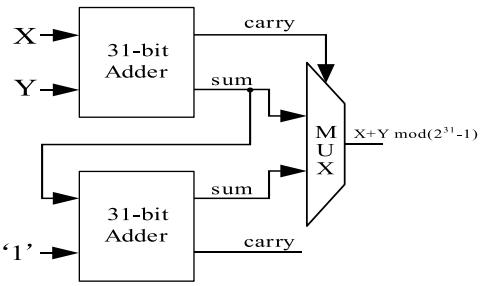


FIGURE 11. The architecture of adder mod (2^{31}-1).

D. THE IMPLEMENTATION OF T IMPLEMENTATION OF THE FEEDBACK LOGICAL CIRCUIT

As seen in Figure 5, the operation of the presented ZUC Cipher Algorithm starts with the initial parallel loading of those values of the LFSR. Furthermore, the initial values of the two registers of R1 and R2 are zero. And then, during the stage of the initialization, the LFSR can receive a 31-bits word as input via that Multiplexer (selecting input 1 of the MUX). To generate this output is through an addition mod (2^{31}-1) between the outputting signal of a feedback logic and the 31-bits output of the nonlinear function F named W (removing the rightmost bit of the output W, W $\gg$ 1). Figure 9 below illustrates the implementation of the feedback logic circuit:

There is not any output but clocking the cipher in this procedure. Therefore, a 32-bits output register is at the position of the output end of the cipher that can hold those produced data. Additionally, in the work mode, the LFSR selects input2 of MUX and does not get any newly arrived input. It can discard the output W if finishing the cipher once. Then after this, the cipher generates a 32-bits keystream Z per clock cycle.

A bit-by-bit XOR between the W and X_3 word can generate this keystream Z as an output of the BR layer. During the stage of the operation, the 32-bits output register locks its input onto its corresponding output signal. Figure 10. shows the implementation of ENTITY and ARCHITECTURE of the XOR gate based on the FPGA of the CPLD as below:

E. THE IMPLEMENTATION OF THE ADDER MODE (2^{31}-1)

The feedback logic circuit is an Arithmetic Logic Unit (ALU) that combines cyclical shift registers and additions of the mod (2^{31}-1). The MUX regulates the configuration according to operation scenarios of the cipher (an initialization or a work step). In addition, the proposed system of the evolutionary ZUC needs another adder mod (2^{31}-1) and adopts this adder's results as the first input of the MUX. There are six adders mod (2^{31}-1) in this feedback logic. Figure 11 shows the structure for the two-inputs X, Y, adder mod (2^{31}-1) as below:

As shown in the figure above, three adders are in use, respectively: one adder for adding the values of S_0 and 2^8 S_0, the second one for adding 2^{20} S_4 and 2^{21} S_{10}, and the third for adding 2^{17} S_{13}, with 2^{15} S_{15}.

Figure 12. gives the implementation of ENTITY and ARCHITECTURE of the Adder based on the FPGA of the CPLD as below:

V. EXPERIMENTAL RESULTS, DISCUSSIONS AND COMPARISONS

Based on the theoretical research and 2FSK modulation mode above, the authors apply the simulation model of the FH communication system in figure 13 and simulate it in Simulink.

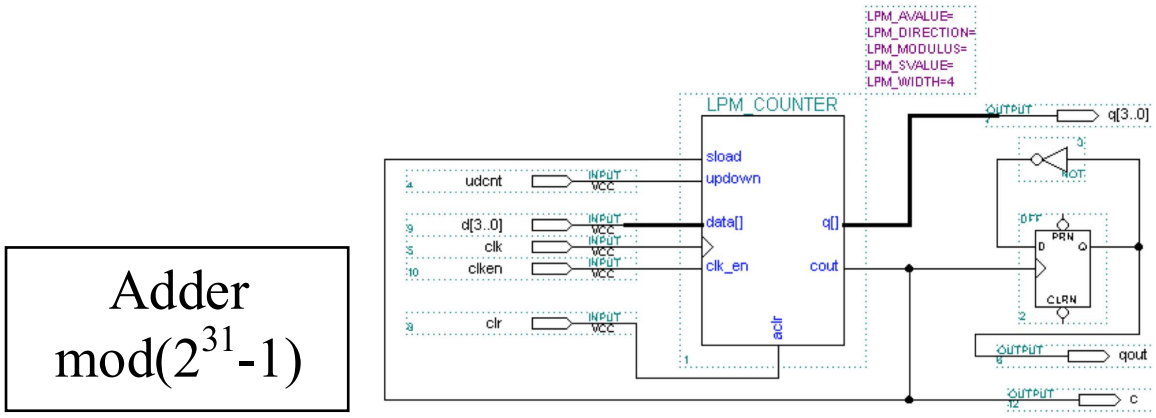
The authors selected several frequencies for the FH demonstration in this paper. Because frequencies will have a poor display effect on the picture and are difficult to identify while the frequency range is too high. Consequently, making these choices is just for the convenience of this demonstration.

The adopted simulation process of the frequency hopping communication system in this paper is as follows: to begin with, the signal generator generates the original sequence; then 2FSK frequency shift keying modulates it with a modulation law controlled by the frequency hopping sequence; Furthermore, the receiver mixes and amplifies it again and sends it to the demodulation module to recover the original information signal. As shown in figure 13, a series of modulation, mixing, demodulation, decision, and other processes can ensure the data recovered at the receiving end is mostly the same as the original sequence at the transmitting end.

The following parts are the explanation of all units in this system:

(1). Signal Generation Unit (SGU): A random integer signal generator can produce this part, which generates the binary random signal with a frequency of 1 Hz;

(2). Transmission Unit (TU): multiplying the FH signal generated through the subsystem module and the 2FSK signal



(a). the ENTITY of an Adder. (b). the ARCHITECTURE of an Adder.

FIGURE 12. The circuit of an adder.

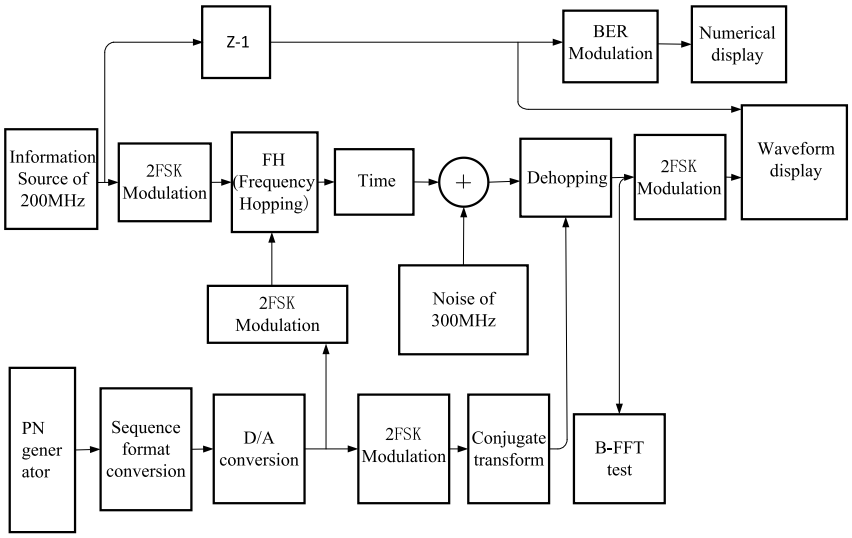


FIGURE 13. The simulation model of the FH communication system.

produced by the baseband to obtain a mixed-signal, and then sending this mixed-signal via the channel loaded with an additive Gauss white noise;

(3). Receiving Unit (RU): the whole process of this unit is equivalent to an inverse procedure of the TU, which means that for finally getting the required low-frequency baseband signal, this process needs to undergo de-hopping, coherent demodulation, and band-pass filtering, etc.;

(4). Decision Unit (DU): this part first receives the low-frequency signal from the previous unit, then compares and determines between the upper path low-frequency signal and the lower path one. The comparator compares the threshold with the threshold of the symbol. If it is higher than the threshold, this signal is a "1" code, or it is a "0" code;

(5). the Subsystem of the FH Unit (SFU): for the convenience of demonstration, supposing the transmission rate of ad hoc information is 1 bit/s and the frequency hopping speed is two hop/s in this scheme. In this subsystem, the sequence outputs one column of 01 signal into two columns of these 01 sequences through a cache, which will become an integer by converting a bit by bit. After passing through the anti-buffer and zero-order holder, the 01 sequences generated by a PN sequence generator can be considered a required FH sequence and sent to the frequency synthesizer;

(6). BER Computation Unit (BCU): a BER meter module can detect the BER. This unit compares the sequence code after a specific delay with the recovered sequence code, counts the different numbers, and divides the number of errors by the total number to obtain the bit error rate.

A. THE SIMULATION TOOLS

Utilizing VHDL with structural description logic can capture the proposed implementation. The VHDL code has been simulated and verified using test vectors provided via the 3GPP standard.

An SYNPLICITY FPGA has advanced more useful features for this design than traditional LUTs and registers. A versatile DSP block is one of these advanced factors. This

TABLE 1. Comparison of the differential performance.

Number of Circles	Difference Analysis			differential uniformity
	Eight circles	Ten circles	Twelve circles	
DES S-box	6.60766	13.2861	15.7154	16
Evolutionary S-box	6.87073	14.0010	16.83109	16

TABLE 2. Comparison of linear performance.

Number of Circles	Linear Analysis			differential uniformity
	Four circles	Six circles	Eight circles	
DES S-box	3.3559	8.7122	10.7693	20
Evolutionary S-box	4.0011	9.5578	11.3618	16

TABLE 3. Comparison of the performance on hardware utilization.

Schemes	Freq(MHz)	Area(slices)	Throughout(Mbps)	Throughout/Area
1.SNOW 3G [10]	126	311	2016	6.5
2.chaotic system [11]	108	356	3456	9.7
3.Proposed ZUC	222.4	575	7111	12.3

DSP block can implement resource- and timing-critical components (i.e., arithmetic operations on Boolean expressions or Integers). However, that leads to being resource-demanding or slower than the implementation with the traditional logic elements. The DSP block can operate the device working at the maximum frequency, i.e., up to 550MHz. This core was in use both in function F and the feedback logic circuit, and its corresponding aim is respectively for the implementation of the adders mod 2^{32} and the adders mod $(2^{31}-1)$. Finally, the operation of that required S-boxes was through a ROM.

To implement the proposed evolutionary ZUC cipher method, the authors of this research use the SYNPLICITY tool and an ALTERA FPGA Device based on Xilinx Virtex-5 for further logic optimization.

As the FPGA tool, Xilinx Virtex-5 is famous for its extensive and powerful clock management functions, such as clock deskew, frequency synthesis, phase shifting, and dynamic reconfiguration. Consequently, this Xilinx Virtex-5 is suitable and feasible to build the FH system due to the clock management functions.

There are three main reasons to explain why the authors selected Xilinx Virtex-5 rather than Virtex-4 or Spartan-6 to implement this novel proposed stream cipher ZUC on an FPGA: firstly, compared with Virtex-4, Virtex-5 has a more extensive and powerful clock management functions, such as clock deskew, frequency synthesis, phase shifting, and dynamic reconfiguration; secondly, Virtex-5 can achieve higher density and performance with lower power consumption and cost than virtex4, including higher accuracy, larger storage capacity and lower power consumption with only 50% slice; in addition, compared with spartan-6, Virtex5 can support most common and emerging standards and has low power consumption. However, due to the military characteristics of spartan-6, it supports fewer standards and high-power consumption.

B. THE PERFORMANCE ANALYSIS OF EVOLUTIONARY S-BOX

Table 1 and Table 2 illustrate the performance comparison between the common S-box with DES and the evolutionary S-box.

Table 1 illustrates that the evolutionary S-box is better than the common one but almost alike in differential uniformity (all equal 16). Therefore, the proposed implementation of the ZUC algorithm can operate at lower clock rates but is virtually on par with prior methods.

It can be seen from Table 2 that the evolutionary S-box has much stronger resistance to the linear attack and better linear uniformity than the common S-box.

Furthermore, besides Tables 1 and 2, to evaluate the proposed architecture further, the authors compared the performance and area utilization with other stream ciphers, including Snow [12] and chaotic [13].

As shown in Table 3, as a stream cipher method, ZUC can meet various demands for the consumed area of hardware resource and adjustments of throughput in implementation on FPGA in terms of different scenarios, which satisfies the design needs of the modern cryptographic algorithms. Based on those simulation results displayed in Table 4, the third scheme (i.e., the proposed algorithm in this paper), a pipelined structure of the ZUC implementation, can obtain maximum speed by consuming a little more area of 575 slices. Moreover, Scheme 3 achieves the best optimal performance over average throughput per area, 12.3 compared to 9.7 of Scheme 2 and 6.5 of Scheme 1.

The numerical simulation illustrates that Scheme 3 has the shortest critical path because its work frequency is inversely proportional to this critical path. Though this path of Schemes 1 and 2 is also that of the LFSR process, their actual values are different because of various optimized ways. However, the critical pathing of the proposed Scheme 3 is that of the nonlinear function F process and is much shorter than it of both Schemes 1 and. Consequently, the presented

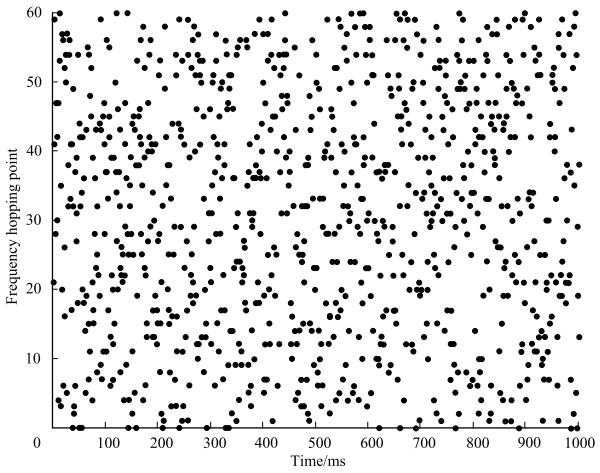


FIGURE 14. Frequency hopping pattern.

Scheme 3 can obtain maximum throughput of 7.1 Gbps only with 575 slices of consuming area. Furthermore, it also achieves the best average throughput per area.

Therefore, the security performance of the evolutionary S-box is much better.

C. RANDOMNESS TEST

This part first shows a result of the presented FH mode and then illustrates the results of a NIST (National Institute of Standards and Technology) Randomness Test. To test the security of the proposed algorithm. The article adopts a randomness/passing the NIST pseudorandom test suite, but this does not directly equate to the security concern. It is just an acceptable and more reasonable one of those most feasible methods.

The TOD is 64 bits, and the period of the generated 64-bits cipher-text is 264 in the proposed scheme. From Figure 14 below, the distribution of these 500 hops is relatively uniform, almost has no period, and presents an asymmetric performance, which illustrates that the FH sequence has a better aperiodic law and much better randomness.

Besides the test of the HF mode above, this research also utilized the NIST (National Institute of Standards and Technology) test components [17] to detect the HF sequence better. The test methods mainly include 16 test methods, such as frequency test, inner block frequency test, and run-length test. According to different test methods, the authors tested the FH sequence with a length of 200000 and achieved the p-value (p-value, probability, PR) values of various test methods. Table 4 shows the final results as below:

These six testing items include the inner block frequency test, cumulative sum test, binary matrix rank test, non-overlapping template matching test, overlapping template matching test, and linear-complexity test. Consequently, we can consider that the FH sequence generated through the proposed FH system is rightly random.

TABLE 4. NIST test results of the frequency hopping sequence.

Test items	*P*-value
Frequency test	0.316932
Inner block frequency test (m=128)	0.559048
Accumulation sum test -Reverse	0.632409
Accumulation sum test -Forward	0.748845
Run-length detection	0.085505
Longest continuous "1" test in block	0.213746
Rank test of binary matrix	0.591424
Discrete Fourier transform (DFT) test	0.137726
Matching test for non-overlapping modules (m=9, B= 000000 001)	0.514595
Matching test for overlapping modules (m=9)	0.244663
Global general statistical test	0.321932
Approximate entropy detection (m=10)	0.095473
Random offset test (x=+1)	0.136937
Random offset variable test (x=-1)	0.428143
Serial test (m= 16)	0.155857
Linear complexity detection (M=500)	0.825982

D. EXPERIMENTAL RESULTS AND COMPARISONS

Table 5 below shows the synthesis implementation results. Also, there are comparisons among the presented algorithm with other similar bitstream ciphers and those previous versions of ZUC encryption methods.

PAN and LIU proposed a so-called Old ZUC algorithm, i.e., a pipeline implementation of the previous version of ZUC [10]. This implementation achieves better time performance than the proposed method in this paper but requires much more hardware resources. In this new version, the output W of that F function combines with the feedback logic circuit by an adder mod (231-1) instead of an XOR gate, which can cause a clear increment in the critical path delay. Another reason is the feedback logic circuit utilizes pipeline registers in the Old-ZUC implementation to decrease this delay of the critical path.

However, it also increases the operation of the algorithm latency. And the last reason is the initialization step was in the software, which triggers a big decrement in the delay in that critical path and the hardware resource. The implementation in this research supports all stages of the execution. The implementation used in this paper executes that initialization stage using the OR-gates among LFSR cells, and the addition (231-1) combines the feedback logic with the MUX and the output W of function F. The implementation of the SNOW 3G ASIC in [11] can obtain a throughput of 7.9 Gbps under a 249 MHz clock frequency, and the maximum cover is 24.4 K GEs. One part of the ISO/IEC 18033-4: 2005 is a

TABLE 5. Results and comparisons with other methods.

Architecture	Technology	# FFs	# Slice	Freq (MHz)	Bit rate (Gbps)
Proposed ZUC	XC5VLX50T3FF324	641	386	66	2.09
Old ZUC [10]	XC5VLX110T	-	575	232	7.1
SNOW 3G [11]	0.13μm	25016GEs*		248	7.8
MUGI [12]	2V500FG456	2437	1965	111	9

*: **Note please**, this GEs (Gate Equivalent) is equal to the specific area of a two-input NAND gate, which denotes the area metric for ASIC designs.

MUGI generator [12]. That paper proposed an FPGA implementation of the MUGI stream cipher, which can achieve a throughput of 7 Gbps under an 11 MHz clock frequency.

VI. CONCLUSION

As a well-known communication method, the security performance of the FH communication is an essential part of it. The ZUC is a new bit-stream cipher adopted in the 5G wireless system. This paper describes the high-speed structure of a novel ZUC and also implements this architecture using an FPGA design. According to the traditional ZUC algorithm, this thesis innovatively applied the permutation polynomial algorithm and an evolutionary DES cipher algorithm to the ZUC algorithm, achieved a novel improved ZUC method, and then utilized it to construct the new frequency hopping communication system. The introduction of that new DES and the permutation polynomial can efficiently increase the security performance of the ZUC system but also avoid some shortcomings of the UHF communication.

The experimental results gained in this paper denote that this ZUC implementation is a resilient, flexible and robust solution for the 4G of LTE usage, even for 5G to some extent. This article proposed the implementation of the novel bit-stream cipher ZUC on an FPGA. In the improved scheme, the permutation polynomial in the first layer (i.e., the top one is the LFSR layer) increases the linear complexity. Furthermore, the evolutionary DES algorithm in the bottom layer also efficiently obtains security performance. This FPGA implementation presented in our experiment can achieve a 2.08 Gbps throughput under a clock frequency of 65 MHz. In addition, results highly recommend that the users choose an appropriate optimized scheme according to their specific requirements and should thoughtfully consider other relevant factors, including the throughput of the system and the consumed area of the algorithm. Finally, although this research is a successful attempt at the FH communication, the authors hope to utilize the evolutionary S-box and the algorithm with better security performance to build a better and more efficient FH system to improve the performance of the communication system in future research.

The authors will supplement much more results for the practice environments to perfect our paper in future work.

AUTHOR CONTRIBUTIONS

Wang Jinpeng designed the measurement scheme, carried out the simulations, and wrote the article; Zhao Xin supervised the work, arranged the architecture, and contributed to the writing of the article; and Zhang Teng, Zhang Bo, and Jeremy-Gillbanks performed the field test and analyzed the data.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] *Internationalization of Domestic Cryptographic Algorithm Standards Creates New Achievements ZUC Algorithm Offi-Cially Becomes the ISO/IEC International Standard*. [Online]. Available: http://www.oscca.gov.cn/sca/xwdt/202005/11/content_1060747.shtml.2020

[2] N. Drucker and S. Gueron, "Fast constant time implementations of ZUC-256 on x86 CPUs," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2019, pp. 1–7, doi: 10.1109/CCNC.2019.8651851.

[3] J. Wang, Z. Ye, T. M. Sanders, B. Li, and N. Zou, "A novel linear antenna synthesis for linear dispersion codes based on an innovative HYBRID genetic algorithm," *Symmetry*, vol. 11, no. 9, p. 1176, Sep. 2019.

[4] D. Zhang, A. Zhao, X. Yang, Y. Sun, and J. Xiao, "Generalized synchronization between Chen system and Rucklidge system," *IEEE Access*, vol. 7, pp. 8519–8526, 2019.

[5] X. Liu, X. Bi, H. Yan, and J. Mou, "A chaotic oscillator based on meminductor, memcapacitor, and memristor," *Complexity*, vol. 2021, pp. 1–16, Dec. 2021.

[6] T. Liu, H. Yan, S. Banerjee, and J. Mou, "A fractional-order chaotic system with hidden attractor and self-excited attractor and its DSP implementation," *Chaos, Solitons Fractals*, vol. 145, Apr. 2021, Art. no. 110791.

[7] T. Liu, S. Banerjee, H. Yan, and J. Mou, "Dynamical analysis of the improper fractional-order 2D-SCLMM and its DSP implementation," *Eur. Phys. J. Plus*, vol. 136, no. 5, p. 506, May 2021.

[8] C. Ma, J. Mou, P. Li, and T. Liu, "Dynamic analysis of a new two-dimensional map in three forms: Integer-order, fractional-order and improper fractional-order," *Eur. Phys. J. Special Topics*, vol. 230, nos. 7–8, pp. 1945–1957, Aug. 2021.

[9] X. Li, J. Mou, Y. Cao, and S. Banerjee, "An optical image encryption algorithm based on a fractional-order laser hyperchaotic system," *Int. J. Bifurcation Chaos*, vol. 32, no. 3, Mar. 2022, Art. no. 2250035.

[10] X. Gao, J. Mou, L. Xiong, Y. Sha, H. Yan, and Y. Cao, "A fast and efficient multiple images encryption based on single-channel encryption and chaotic system," *Nonlinear Dyn.*, vol. 108, no. 1, pp. 613–636, Mar. 2022, doi: 10.1007/s11071-021-07192-7.

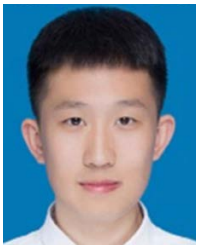
[11] X. Gao, J. Mou, S. Banerjee, Y. Cao, L. Xiong, and X. Chen, "An effective multiple-image encryption algorithm based on 3D cube and hyperchaotic map," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1535–1551, Apr. 2022, doi: 10.1016/j.jksuci.2022.01.017.

[12] X. Ma, J. Mou, J. Liu, C. Ma, F. Yang, and X. Zhao, "A novel simple chaotic circuit based on memristor-memcapacitor," *Nonlinear Dyn.*, vol. 100, no. 3, pp. 2859–2876, 2020.

[13] R. Tang, J. Duan, and H. Deng, "Image encryption algorithm based on logistic chaotic sequence and DES," *J. Comput. Appl.*, vol. 1, pp. 89–92, Jan. 2017.

[14] J. S. Sousa and J. P. Vilela, "A characterization of uncoordinated frequency hopping for wireless secrecy," in *Proc. 7th IFIP Wireless Mobile Netw. Conf. (WMNC)*, May 2014, pp. 1–4.
[15] A. Boonkajay and F. Adachi, "PAPR reduction for STBC transmit diversity with transmit FDE using blind selected mapping," in *Proc. IEEE VTS Asia Pacific Wireless Commun. Symp. (AP-WCS)*, Incheon, South Korea, Aug. 2017, pp. 1–5.
[16] D. Sun and Q. Zhang, "A secure constellation design for polarized modulation in wireless communications," *IEEE Access*, vol. 8, pp. 130589–130597, 2020.
[17] W. Chao, L. Pengcheng, and L. Rui, "Security frequency hopping communication system based on an improved ZUC algorithm," *Tsinghua Univ. Sci. Technol.*, vol. 59, no. 2, pp. 154–161, 2019.
[18] X. T. Feng, "The ZUC stream cipher algorithm," *J. Inf. Secur. Res.*, vol. 2, no. 11, pp. 1028–1041, 2016.
[19] J. K. M. S. U. Zaman and R. Ghosh, "Review on fifteen statistical tests proposed by NIST," *J. Theor. Phys. Cryptogr.*, vol. 1, pp. 18–31, Nov. 2012.
[20] X. Wang and H.-L. Zhang, "A novel image encryption algorithm based on genetic recombination and hyper-chaotic systems," *Nonlinear Dyn.*, vol. 83, nos. 1–2, pp. 333–346, 2016.
[21] A. N. Bikos and N. Sklavos, "Architecture design of an area efficient high speed crypto processor for 4G LTE," *IEEE Trans. Depend. Secure Comput.*, vol. 15, no. 5, pp. 729–741, Sep. 2018, doi: 10.1109/TDSC.2016.2620437.
[22] X. Jin, Y. Chen, S. Ge, K. Zhang, X. Li, Y. Li, Y. Liu, K. Guo, Y. Tian, G. Zhao, and X. Zhang, "Color image encryption in CIE L *a*b*: Space," in *Proc. ATIS 6th Int. Conf. Appl. Techn. Inf. Secur.*, Berlin, Germany: Springer, 2015, pp. 74–85.
[23] L. Xu, Z. Li, J. Li, and W. Hua, "A novel bit-level image encryption algorithm based on chaotic maps," *Opt. Lasers Eng.*, vol. 78, no. 21, pp. 17–25, 2016.
[24] Design Team, "ZUC-256 stream cipher," *J. Cryptol. Res.*, vol. 5, no. 2, pp. 167–179, 2018, doi: 10.13868/j.cnki.jcr.000228.
[25] W. Jinpeng, C. Fan, and Z. Nianyu, "Multi carrier system joint receiving method based on MAI and ICI," *J. Jilindaxue*, vol. 41, no. 6, pp. 1793–1797, 2018.
[26] J. Wang, Y. Zhengpeng, J. Gillbanks, T. M. Sanders, and N. Zou, "A power control algorithm based on chicken game theory in multi-hop networks," *Symmetry*, vol. 11, no. 5, p. 718, May 2019.
[27] J. G. Andrews, S. Buzzi, C. Wan, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.
[28] Y. Niu, Y. Li, D. Jin, L. Su, and A. V. Vasilakos, "A survey of millimeter wave communications (mmWave) for 5G: Opportunities and challenges," *Wireless Netw.*, vol. 21, no. 8, pp. 2657–2676, Nov. 2015.
[29] E. Turgut and M. C. Gursoy, "Average error probability analysis in mmWave cellular networks," in *Proc. IEEE 82nd Veh. Technol. Conf. (VTC-Fall)*, Sep. 2015, pp. 1–5.
[30] *Studyon Supporting256-Bitalgorithms for 5G[R/OL]*, Standard 3GPPSA3.TR33.841, 2018.
[31] Design Team, "ZUC-256 stream cipher," *J. Cryptolog. Res.*, vol. 5, no. 2, pp. 167–179, 2018, doi: 10.13868/J.cnki.jcr.0002281.
[32] P. Ekdahl, T. Johansson, A. Maximov, and J. Yang, "A new SNOW stream cipher called SNOW-V," *IAcR Cryptol. ePrint Arch.*, vol. 2018, Jan. 2018, Art. no. 1143. [Online]. Available: http://eprint.iacr.org//2018/1143.pdf
[33] G. Han and J. Song, "Extensions of the I-MMSE relationship to Gaussian channels with feedback and memory," *IEEE Trans. Inf. Theory*, vol. 62, no. 10, pp. 5422–5445, Oct. 2016.
[34] J. Zhang, J. Wang, and S. Zhang, "Pseudorange measurement method based on AIS signals," *Sensors*, vol. 17, p. 1183, May 2017.
[35] K. Zheng, Q. Hu, and J. Zhang, "Positioning error analysis of ranging-mode using AIS signals in China," *J. Sensors*, vol. 2016, pp. 1–11, Aug. 2016.
[36] M. Hu, Y. Li, X. Lu, and H. Zhang, "Tone reservation to minimize nonlinearity impact on OFDM signals," *IEEE Trans. Veh. Technol.*, vol. 64, no. 9, pp. 4310–4314, Sep. 2015.
[37] B. Mondal, T. Thomas, E. Visotsky, F. Vook, A. Ghosh, Y.-h. Nam, Y. Li, J. Zhang, M. Zhang, Q. Luo, Y. Kakishima, and K. Kitao, "3D channel model in 3GPP," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 16–23, Mar. 2015.
[38] R. Guido and D. Conforti, "A hybrid genetic approach for solving an integrated multi-objective operating room planning and scheduling problem," *Comput. Oper. Res.*, vol. 87, pp. 270–282, Nov. 2017.
[39] W. Jinpeng, C. Fan, and Z. Nianyu, "Cooperative distributed antenna transmission method based on co-channel interference in 5G mobile communication system," *J. Jilindaxue*, vol. 48, no. 1, pp. 333–341, 2020.
[40] J. Barrueco, J. Montalban, E. Iradier, and P. Angueira, "Constellation design for future communication systems: A comprehensive survey," *IEEE Access*, vol. 9, pp. 89778–89797, 2021.
[41] Z. Babar, Z. B. K. Egilmez, L. Xiang, D. Chandra, R. G. Maunder, S. X. Ng, and L. Hanzo, "Polar codes and their quantum-domain counterparts," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 123–155, 1st Quart., 2020.

WANG JINPENG was born in Wulanhaote, Xingan, China, in 1979. He received the B.S. and M.S. degrees in electronic engineering from the Dalian University of Technology, Dalian. His current research interest includes numerical optimization.



ZHANG TENG received the B.S. and M.S. degrees in electronic engineering from the University of Zhengzhou, Henan, in 2019. He is currently pursuing the master's degree with the School of Information Science & Engineering, Dalian Polytechnic University, Dalian, Liaoning, China.

ZHANG BO received the B.S. degree in electronic engineering from the University of Haerbin Technology, Heilongjiang, China, in 2001. He is currently pursuing the master's degree with the School of Information Science & Engineering, Dalian Polytechnic University, Dalian, Liaoning, China.

JEREMY-GILLBANKS was born in Perth, Western Australia, Australia, in 1977. He received the B.S. and M.S. degrees in electronic engineering from The University of Western Australia (M350), Australia, in 2016, where he is currently pursuing the Ph.D. degree with the School of Electronic, Electrical and Computer Engineering. He has authored or coauthored several research articles indexed in either Scopus or web of science.

ZHAO XIN (Member, IEEE) was born in Jinzhou, Liaoning, China, in 1968. He received the B.S. and M.S. degrees from Jilin University, Changchun, in 1996. He is currently a Professor with Dalian Polytechnic University, Dalian, Liaoning. His current research interest includes numerical optimization.