Advances and Applications in Mathematical Sciences
Volume 17, Issue 1, November 2017, Pages 271-292

FPGA IMPLEMENTATION OF HYBRID SYSTEM USING TIMING ATTACK RESISTANT CRYPTOGRAPHIC TECHNIQUE

NARENDRA BABU T and FAZAL NOORBASHA

Department of Electronics and Communication Engineering
K L University, Guntur
Andhra Pradesh, India
E-mail: tatininarendra@kluniversity.in
fazalnoorbasha@kluniversity.in

Abstract

In this article, a hybrid system is presented for security and reliability of data while it transmits over communication channel. In this proposed system Brackets, Order, Division, Multiplication, Addition and Subtraction (BODMAS) sequence of operations are used as an encryption algorithm. The data reliability is obtained by using orthogonal codes. This coding technique improves the percentage of error detection and correction rates up to 99.9999%. The segmented decoder technique is used to reduce clock cycles. The clock cycles required to decode the received data is equal to half of the orthogonal code length. The results show that our proposed system resistant to timing attacks for variable data inputs.

1. Introduction

Security is a significant factor to transmit the reliable data on the internet and is a key to success of all applications [1]. Various cryptography techniques are proposed to secure the data being exchanged by electronic media. Cryptographic algorithms play a crucial role to achieve the security services such as authentication, confidentiality, and integrity. Symmetric and asymmetric cryptography are the two types of encryption techniques used in cryptosystems. Data Encryption Standard (DES) [2] and Advanced Encryption Standard (AES) [3] are the symmetric cryptography, also called private-key cryptography uses only one key for encryption and decryption.

2010 Mathematics Subject Classification: 68N13, 97N50.
Keywords: BODMAS, Encryption, Orthogonal code.
Received March 10, 2017; Accepted July 20, 2017

RSA [4] and Elliptic curve cryptography [5] are asymmetric key cryptography, also called public-key cryptography requires special keys to encode and decode messages. The security of all these cryptosystems are completely relies on the complexity of solving the mathematical problems used in the respective cryptographic technique. The efficiency of cryptosystem is fully influenced by the mathematical operations. These mathematical expressions are also constructed by using Boolean functions. Most often the execution time taken by the cryptosystems is varied for different inputs. This leads to a timing attacks. Timing attacks are side channel attacks in which the third party might extract secret data and intellectual property by analysing the execution time of the cryptographic algorithms. In this paper we focus on that to do not change the timing for different data inputs and reliability of data during transmission.

The organization of this paper is as follows:

Brief overviews of related work of some existing cryptographic algorithms and error detection and corrections codes are presented in Section 2. The design methodology of proposed hybrid encryption algorithm is introduced in Section 3. Section 4 presents the implementation of proposed algorithm. Section 5 presents the numerical results and the simulation results of proposed algorithm. Finally, conclusion is presented in Section 6.

2. Related Work

C.W. Chiou et al., proposed a semi systolic limited field multiplier over GF (2^m) has detriments of the global and the feedback lines. Their demonstrated DB semi systolic Hankel multiplier disposes of feedback lines. Already proposed DB systolic Hankel multiplier with CEC capacity requires only about 3% of additional space unpredictability and will be around 15% of time [5].

Shalini Sharma et al., demonstrated a method with less number of computations. It is simple method for finding errors by removing candidate bits. It helps in reducing complexity. Their method can be used to detect, correct errors from data bits and parity bits without any more calculation. Other correction codes can be used with this method to increasing errors for correcting in single line [6].

N. Julai et al., suggested a solitary occasion disturb tolerant lock that incorporates error location and rectification abilities. The proposed hook identifies errors by changing over a solitary rail information flag to double rail information. The second proposed hook utilized a razor flip tumble to correct errors by utilizing a shadow lock controlled by deferred clock to re-establish and amend values [7].

Bernhard Fechner et al., proposed the novel checksum plot for the pipelined processors called String Mark Math, supporting a discretionary count for stages and strings. They presented 2 sub-plots, the pressure based and pressure free checksum analytics. Plans can be utilized inside a multiprocessing situation with shared memory applications, either on by applying processor level or between the duplexed processors, incorporating checksum instrument in control and information ways [8].

Debiao He et. al, demonstrated a top to bottom review ECC-based RFID verification plans. They recognized a portion of the security prerequisites that a RFID validation plan ought to fulfill and proposed ECC-based RFID plans. They planned ECC-based RFID confirmation plans, which are provably secure in the security display [9].

Newton G. Bretas et al., suggested that power framework stage estimation require two stages. In the initial stage their reproductions has indicated numerous circumstances where traditional state of estimation bombs of gross error recognition and also in the recognizable proof test at the same time, utilizing their proposition past distinguishing and present distinguishing the estimations with gross errors effectively [10].

S. K. Hafizul Islam et al., proposed 3PAKE convention which has high calculation cost because of the association of the extra elliptic bend scalar point duplications and symmetric cryptosystem. The security investigation on 3PAKE convention affirmed that the proposed convention revamps Tan's 3PAKE convention, as well as secure against other known assaults [11].

Mustafa Demirci et al., demonstrated on a few executions of two existing decoders. The usage have been assessed and another execution is proposed for streamlining of mapping to FPGA model [12].

Chi-Yuan Chen et al. proposed quantum cryptography, quantum key appropriation and quantum secure direct interchanges pass the mystery

message after queue bit was encoded. There is another intriguing theme called dazzle quantum calculation, which encodes calculation asset to organizations that can help us figure enormous information. This will be helpful in distributed computing [13].

Milos Krstic et al., demonstrated two models for delicate error location and remedy in combinational and successive circuits. The utilization of particular indicator hardware for combinational rationale, constraining the equipment overhead that is required to detect error. This design coordinates with current error tolerant systems in consecutive rationale [14].

Rawya Rizk et al., demonstrated a powerful cross breed security calculation for WSN. It is intended to take care of a few issues as useful execution, short reaction time, productive calculation and the quality of cryptosystem. It offers better security for a shorter encryption and decoding time and littlest figure content size [15].

Sheng Yuan, Jianbin Yao et al. demonstrated the precept for CGI based type of encryption technique and found that the method is far susceptible for chosen plaint text attack. Three techniques were proposed and discussed their effectiveness proven by numerical simulation. Computational outcomes display that the following approach is simple but it is effective to avoid these assaults [16].

Claudio Urrean, et al., proposed technique which is particularly valuable in interchanging transport sections that are inclined to transmission errors brought on by electromagnetic obstruction. In these cases, the proposed strategy may not be perfect with every one of the devices connected to the particular transport. To guarantee similarity with all devices, a constrained measure of equality information must be presented in each casing [17].

Claudio Urrea, et al., demonstrated the likelihood of enhancing quality in transmission of MODBUS-RTU. Modern correspondences utilizing re-transmitter devices with their capacity for recognising and revising errors. This type of strategy regards parameters of MODBUSRTU convention, so transport improved re-transmitter devices can be utilized alongside normal devices. Future usage of utilizing VHDL may defeat this confinement by method for expanding the baud rate [18].

Stjepan Picek, et al., led a broad examination on the execution of EA's while developing Boolean capacities for cryptography. They consider just the Boolean capacity measurements that are of useful importance. Two properties that are significant (in particular, AI and FAI) are computationally excessively unpredictable, making it impossible to be parts of the work. In the event that the nature of the AI and FAI properties is basic [19].

Hossein Moradian et al., has proposed a new fault localization, identification, error rectification strategies for self-checking BSDN method. A better, less complicated quality way demonstrates that its capacity is achievable utilizing the FA property. The proposed strategies allows to accomplish both the error correction, fault localization in single step [20].

Eun-Jun Yoon et al. audits and crypt analysis proposed a protocol for area construct administrations with respect to uneven key cryptography and shows various security confinements for protocol and upgraded vigorous validation protocol for versatile administrations in light of elliptic curve cryptography. The casual and formal security investigations exhibits the resistance of proposed protocol against a wide range of security assaults [21].

Donald G. MacKay et al. demonstrated a model for error remedy and discovery for verbatim transcript of H. M. Ten reviews showed that with respect to deliberately coordinated memory-typical controls, H. M. At long last, investigations of H.M's. Uncorrected errors showed two reliable elements (exclusion and oddity) that Embroiled encoding challenges for errors, ED shortages, error checking deficiencies, And EC shortfalls [22].

Praveen Singh et al., proposed the design for limited field multiplier under subthreshold condition is executed. The edge voltage for NMOS and PMOS are 0.2V and - 0.22V individually. Subsequently, multiplier circuit execution will be null under subthreshold condition. This impediment of postpone corruption can overcome by utilizing pipelined design of multiplier [23].

Yibin Li et al., demonstrated on the issue of the cloud information stockpiling and intended to give an approach that could stay away from the cloud administrators achieving client' delicate information. Their assessments had demonstrated that their proposed plan could adequately shield significant dangers from cloud-side. Their future work would address

securing information duplications keeping in mind the end goal to expand the level of information accessibility [24].

Debiao He et al. proposed another versatile client validation convention for the multi-server structures without the need for on-line enlistment jog. Security examination demonstrates that proposed convention is provably secured in arbitrary prophet display and fulfils security necessities in versatile framework with multi-server designs [25].

Satya Narayana Vollala et al., suggested two calculations BFW1 and BFW2 which are enhanced forms of the calculations utilized for open key cryptographic changes. This will impressively enhance the execution of security calculations on the current server workloads. Equipment acknowledged for the proposed calculations have additionally been illustrated [26].

Quinn DuPont et al., proposed a model for Edge Cryptography and Advancement of PC Systems and Cyber security for transforming information innovation to give cryptographic difficulties. One of the outcomes of the structural choice to place cryptography at the edge (and later end) that it implied security on the Arpanet and the ensuring Web included scrambled bundle payloads and decoded parcel metadata, which set the phase for protection faces [27].

Narendra Babu T et al., demonstrated that error detection and the error correction rate is grown up to 100% of 8-bit input length. The increase in input data increases the encryption and decryption time delays [28].

Narendra Babu T et al., proposed a technique to implement Error detection and correction that has reached 100% efficiency. If there are any errors present in the data then their technique can be capable to detect and correct in the receiver and the original data can be recovered. And they compared the delay for the 4, 5, 6, 7, 8 bits of data. Therefore the delay is increased when they increase the number of bits and it gets corrected up to (n/4-1) bits for n-bit orthogonal codes [29].

T. Narendra Babu et al., proposed that time taken to detect the input bits increases drastically with respect to the size of LFSR (m-bit). More time it takes to detect the input security is higher [30].

3. Design Methodology

Our approach is to improve the performance of the system and to increase the security level by using a multiple keys to encrypt and decrypt the data. These keys are generated by using the Pseudo random number generator. There are two major blocks involved in the proposed method, i.e., Transmitter and Receiver. The proposed diagram for transmitter and receiver is shown in figure 1 and figure 2 respectively. These blocks are described as follows:

3.1. Transmitter:

Transmitter is fed up with input a-bit data and generates serial data as output. It consists of three blocks namely an Encoder, Encryptor and a Shift register. These are described are as follows:

a. Encoder:

Encoder maps the given input to the unique equivalent orthogonal code. The encoder takes a-bit data as input and produces the $b = 2^{a-1}$ bits as an output data.

b. Encryptor:

The encoded data is sent for encryption, where each block of encryptor performed with Sequence of BODMAS operations with random keys generated by the Pseudo random number generator. The encryption steps are as follows:

Step 1. In this step the b-bit data is divided with key 1.

Step 2. In this step the output obtained from Step-1 is taken as input and multiplied with key 2.

Step 3. In step 3 the output obtained from step-2 is taken as input and Added with key 3.

Step 4. In step 4 the output obtained from step-3 is taken as input and subtracted with key 4.

The data obtained from step 4 is a cypher data which is given to shift register. The shift register will convert the parallel data into serial data and will transmitted bit by bit.

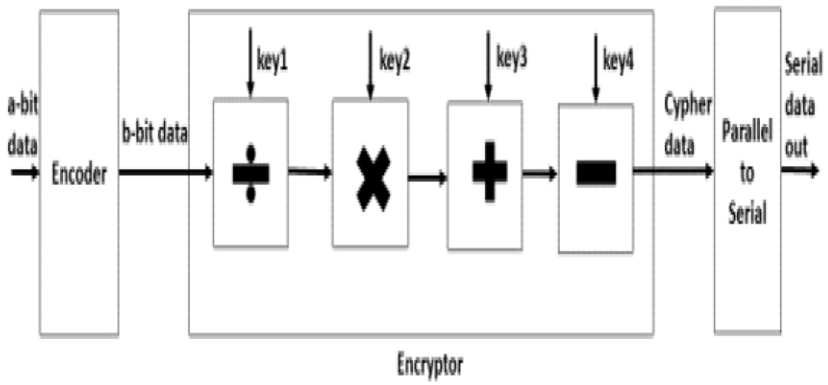

Figure 1. Block diagram of transmitter.

3.2. Receiver:

The reverse arrangement make transmitter acts as a receiver which accepts serial data as input and produces a-bit data as output. The receiver consists of a Shift register, Decryptor and Decoder. These are described as follows.

a. Decryptor:

The output of the shift register is fed to the decryptor. The decryptor takes the cypher text as input and performs the reverse sequence of BODMAS operations and produces b-bit data as an output. The decryption steps are as follows:

Step 1. In this step the cypher data is taken as input and added with key 4.

Step 2. In this step the output obtained from Step-1 is taken as input and subtracted with key 3.

Step 3. In this step the output obtained from step-2 is taken as input and divided with key 2.

Step 4. In this step the output obtained from step-3 is taken as input and multiplied with key 1.

The output obtained from step-4 is taken as b-bit data and sent to decoder.

b. Decoder:

The decoder receives b-bit data as input and produces a-bit data as output. It offers a decision making process, where in which the received b-bit orthogonal code is verified for correct matching with neighbouring codes. The valid code must have good autocorrelation value after comparison. This can be verified by correlation process, where in which a pair of b-bit codes m_1, m_2, m_b and $n_1, n_2, \ldots, n_b$ is compared to yield,

$$R(m, n) = \sum_{i=1}^{b} m_i n_i \leq \frac{b}{4} - 1.$$

Where $R(m, n)$ is matching function for code length of 'b'. Where average count of errors that could be corrected from the following equation

$$t = b - R(m, n) = \frac{b}{4} - 1.$$

Where 't' is number of errors that can be corrected for an orthogonal code. The below table-1 shows some orthogonal codes and their corresponding error correcting capabilities.

Table 1. Orthogonal codes with their corresponding error correcting capabilities.

b	t
8	1
16	3
32	7
4	15

Decoder consists of three parts namely orthodecoder, anti orthodecoder and comparator. These are described as follows:

C. Ortho Decoder:

The b-bit data will be taken to the orthodecoder. The input is XOR'ed with all the combinations of orthogonal codes stored in the pre-defined table for error detection. This will be verified by calculating the count of one's from the signal resulting from b-bit data xor'ed with every sequence of orthogonal

codes in the pre-defined table. Here counter is defined to count one's from output data and inspects for minimum count. Orthogonal codes in the pre-defined table will be combined with minimum count will be the nearest match for corrupted code. A similar orthogonal code from the pre-defined table will be determined as error free code, which is then decoded to count1 and output 1.

d. Anti Ortho Decoder:

The b-bit data will be taken to the anti-orthodecoder. The input is XOR'ed with all the combinations of antipodal codes stored in the pre-defined table for error detection. This will be verified by calculating the count of one's from the signal resulting from b-bit data xor'ed with every sequence of orthogonal codes in the pre-defined table. Here counter is defined to count one's from output data and inspects for minimum count. Antipodal codes in the pre-defined table will be combined with minimum count will be the nearest match for corrupted code. A similar antipodal code from the pre-defined table will be determined as error free code, which is then decoded to count 2 and output 2.

e. Comparator:

Comparator will compare the outputs of the both orthodecoder and anti-orthodecoder and generates the correct matching output with comparing data of count 1 and count 2. The receiver will correct up to (b/4)-1 bits from the received impaired code. If count 1>count 2 output 1 is transmitted as a-bit data. If count 1<count 2, output 2 is transmitted as a-bit data. However, the minimum count which associated with more than one combination of orthogonal code with orthodecoder and anti-orthodecoder a signal' REQ' goes high i.e., Invalid data.

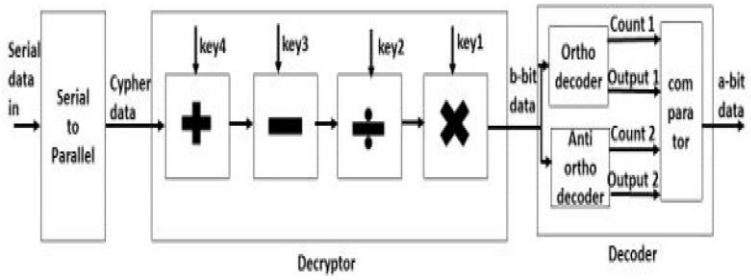

Figure 2. Block diagram of receiver.

4. Implementation

The proposed method has been tested using Spartan-3E hardware board and Xilinx ISE 14.2 software. The code has been synthesized and simulated using ModelSimXE. The simulation results were verified for the combinations of 8-bit input data, 128-bit orthogonal code and 128 bit key size. The simulation results show all the values in hexadecimal number system. The signal 'reset' is used in the simulation results to clear all the values to "0". In the simulation results all the output values are affected with the every raising edge of the clock. The simulation results of transmitter and receiver are described as follows:

4.1 Transmitter:

a. Encoder:

Figure 3 shows the simulation result of an encoder corresponding to the 8-bit input data value "81H" labelled as 'in'. The encoder maps the input data to the unique 128-bit orthogonal code "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaH" labelled as 'out'.

b. Encryptor:

Figure 4 shows the simulation result of an encryptor corresponding 128-bit input data value "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaH" is labeled as 'PLAIN_TEXT', 128-bit keys "4e415245 4e44524120424142552e2054H", labeled as 'key1', "46415a414c204e4f4f522042415348 41H", labeled as 'key2', "52454e554b45535741524152414f2047H", labeled as 'key3', and "52414d414e2020204152494d494c4c49H", labeled as 'key4'. The encryptor performs the following sequence of operations:

Step 1. The input data value "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaH" is divided with the key1 value "4e4152454e44524120424142552e2054H" and produces an output value "0e2806200e2206286a262826004e6a02000000 00000000000000000000000002H" labeled as 'DIV'.

Step 2. The data value is "0e2806200e2206286a262826004e6a020000 00000000000000000000000002H" is multiplied with the key2 value "46415a414c204e4f4f52204241534841H" and produces an output value "03e28ed40087629506e59304f7cd43831ba51c9bf84bc0f6cd7c7897025f7a828c8 2b48298409c9e9ea4408482a69082H" labeled as 'MUL'.

Step 3. The data value is "03e28ed40087629506e59304f7cd43831ba 51c9bf84bc0f6cd7c78970 25f7a828c82b48298409c9e9ea4408482a69082H" is added with the key 3 value "52454e554b 45535741524152414f2047H" and produces an output value "3e28ed40087629506e59304f7cd43831ba51c9b f84bc0f6cd7c7897025f7a82bec802d7e385eff5dff681d6c3f5b0c9H" labeled as 'ADD'.

Step 4. The data value is "3e28ed40087629506e59304f7cd43831ba5 1c9bf84bc0f6cd7c7897025f7a82bec802d7e385eff5dff681d6c3f5b0c9H" is subtracted with the key4 value "52414d414e 2020204152494d494c4c49H" and produces an output value "3e28ed40087629506e59304f7cd43831ba51c9bf84bc 0f6cd7c7897025f7a828c86b5969565cfd59ea438897aa96480H" labeled as 'CIPHER_TEXT'.

4.2. Receiver:

a. Decryptor:

Figure 5 shows the simulation result of decryptor corresponding received data value "3e28ed40087629506e59304f7cd43831ba51c9bf84bc0f6cd7c78 9702 5f7a828c86b5969565cfd59ea438897aa96480H" is labeled as 'CIPHER_TEXT', 128-bit keys "4e4152454e 44524120424142552e2054H", labeled as 'key1', "46415a414c204e4f4f52204241534841H", labeled as 'key2', "52454e554b45535 741524152414f2047H", labeled as 'key3' and "52414d414e2020204152494d49 4c4c49H", labeled as 'key4'. The decryptor performs the following sequence of operations:

Step 1. The received data value is "3e28ed40087629506e59304f7cd 43831ba51c9bf84bc0f6cd7c897025f7a828c86b5969565cfd59ea438897aa96480 H" is added with the key4 value "52414d414e2020204152494d494c4c49H" and produces an output value "32283d40087629506e59304f7cd43831ba 51c9bf84bc0f6cd7c7897025f7a82dec802d7e385eff5dff681d6c3f5b0c9H" labeled as 'ADD'.

Step 2. The data value is "2283d40087629506e59304f7cd43831ba51c9b f84bc0f6cd7c7897025f7a82dec802d7e385eff5dff681d6c3f5b0c9H" is subtracted with the key3 value "52454e554b455 35741524152414f2047H" and produces an output value "3e28ed40087629506e59304f7cd43831ba51c9bf84bc0f6cd 7c7897025f7ea828c82b48298 409c9e9ea4408482a69082H" labeled as 'SUB'.

Step 3. The data value is "3e28ed40087629506e59304f7cd43831ba51c 9bf84bc0f 6cd7c7897025f7ea828c82b48298409c9e9ea4408482a69082H" is divided with the key2 value "46415a414c204e4f4f52204241534841H" and produces an output value "0e2806200e2206286a262826004e6a02000000000 0000000000000000000002H" labeled as 'DIV'.

Step 4. The data value "00000000000000000000000000000000H" is multiplied with the key1 value "4e4152454e44524120424142552e2054H" and added with "0e2806200e2206286a262826004e6a02000000000000000000000 00000000002H" and produces an output value "aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaH" labeled as 'PLAIN_TEXT'.

b. Decoder:

The decrypted PLAIN_TEXT is XOR'ed with all the combinations orthogonal codes in orthodecoder and the same PLAIN_TEXT is XOR'ed with all the combinations of antipodal codes in anti-orthodecoder. There are three different cases of simulation results. The comparator gives the minimum count of orthodecoder and anti-orthodecoder. The decoded output is the final data which is associated with the minimum count. The simulation results are the examples of data without error, with 2 error bits, 4 error bits, 8 error bits, 16 error bits, 31 error bits and 32 error bits. Each case is explained as follows:

a. Case 1: Data without error:

Figure 6 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaa aaaaaaH" is 'XORed' with orthogonal and antipodal codes *s* present in the pre-defined table and gives the minimum count value "00H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

b. Case 2: Data With 1-bit error:

Figure 7 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaa aaaaabH" is 'XORed' with orthogonal and antipodal codes *s* present in the predefined table and gives the minimum count value "01H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

c. Case 3: Data With 2-bit error:

Figure 8 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaaaaaa aaaacH" is 'XORed' with orthogonal and antipodal codes *s* present in the predefined table and gives the minimum count value "02H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

d. Case 4: Data With 4-bit error:

Figure 9 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaaaaaa aaaaccH" is 'XORed' with orthogonal and antipodal codes s present in the predefined table and gives the minimum count value "04H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

e. Case 5: Data With 8-bit error:

Figure 10 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaa aaaccaaccH" is 'XORed' with orthogonal and antipodal codes *s* present in the predefined table and gives the minimum count value "08H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

f. Case 6: Data With 16-bit error:

Figure 11 shows the simulation result of an orthodecoder and anti ortho decoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaaaaccccccH" is 'XORed' with orthogonal and antipodal codes *s* present in the predefined table and gives the minimum count value "10H" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

g. Case 7: Data With 31-bit error:

Figure 12 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaa b5555555H" is 'XORed' with orthogonal and antipodal codes *s* present in the

predefined table and gives the minimum count value "1fH" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "0" labelled as signal 'req'.

h. Case 8: Data With 32-bit error:

Figure 13 shows the simulation result of an orthodecoder and anti orthodecoder for corresponding data value "aaaaaaaaaaaaaaaaaaaaaaaa55 555555H" is 'XORed' with orthogonal and antipodal codes *s* present in the predefined table and gives the minimum count value "ZZH" labeled as 'count', the associated final output value "81H" labelled as 'out' and the value "1" labelled as signal 'req'. The data value "1" at signal 'req' indicates that the data is corrupted and requesting for retransmit the data.

5. Screenshots

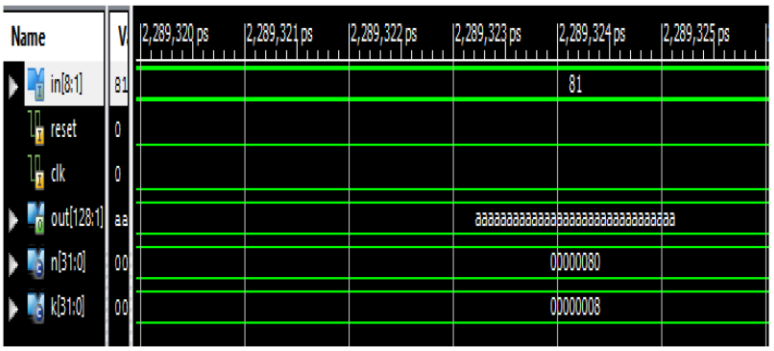

Figure 3. Simulation output of encoder.

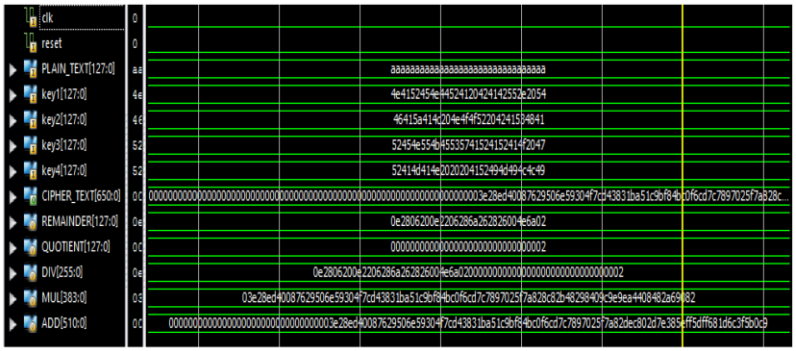

Figure 4. Simulation output of encryptor.

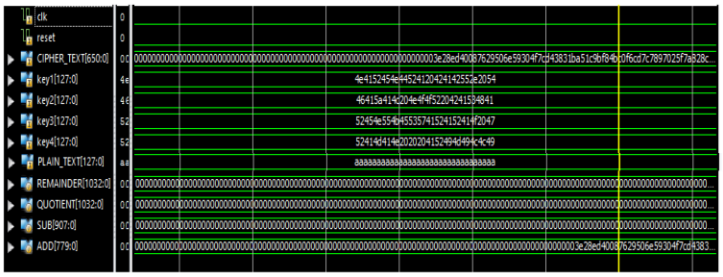

Figure 5. Simulation output of decryptor.

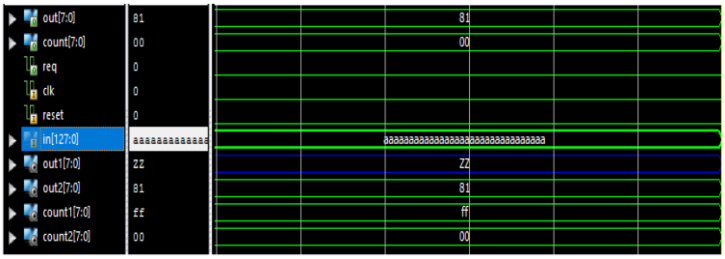

Figure 6. Simulation output of decoder without error.

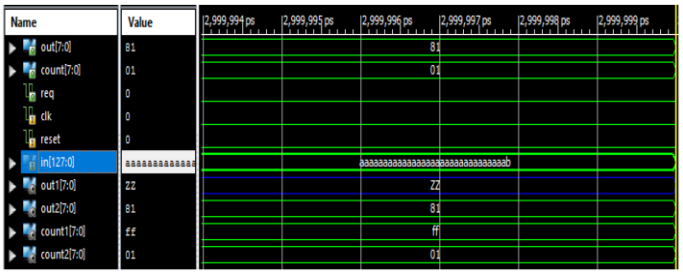

Figure 7. Simulation output of decoder with 1-bit error.

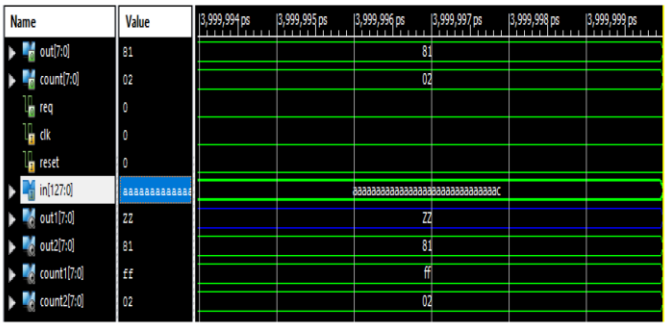

Figure 8. Simulation output of decoder with 2-bit error.

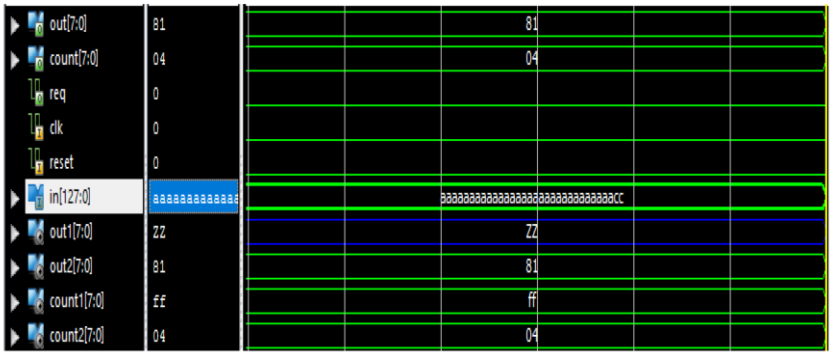

Figure 9. Simulation output of decoder with 4-bit error.

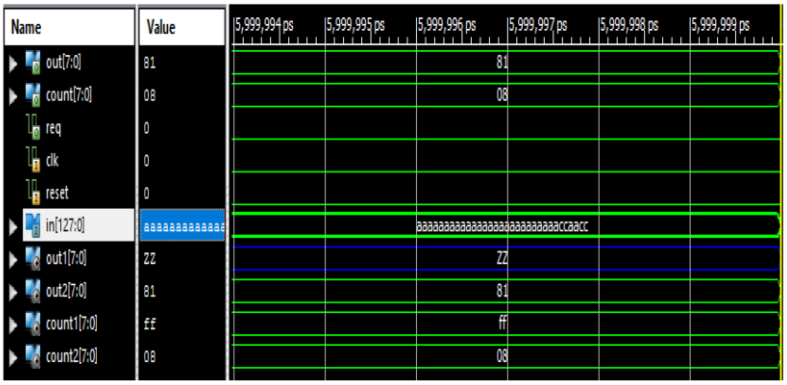

Figure 10. Simulation output of decoder with 8-bit error.

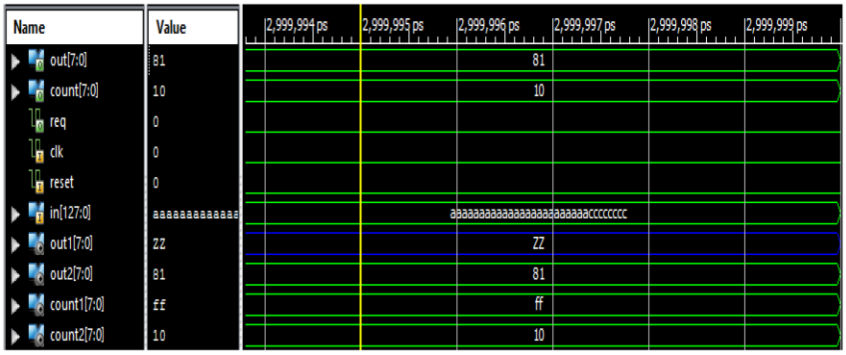

Figure 11. Simulation output of decoder with 16-bit error.

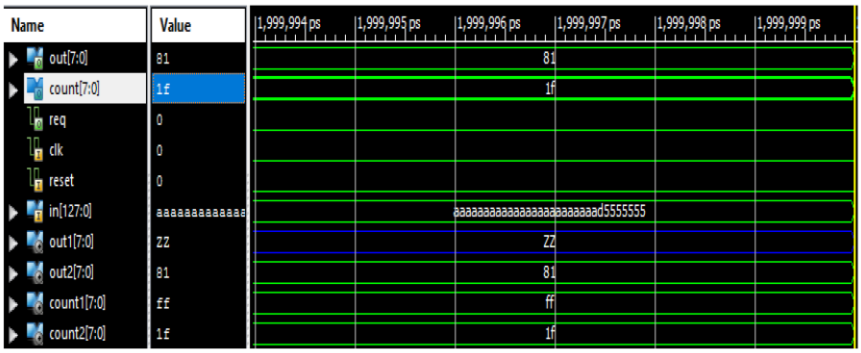

Figure 12. Simulation output of decoder with 31-bit error.

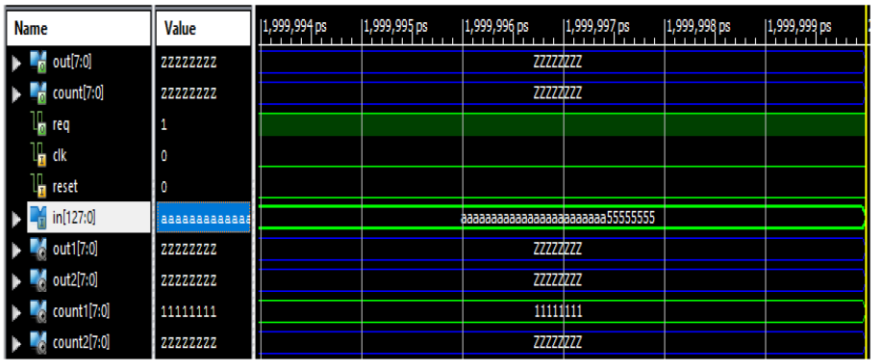

Figure 13. Simulation output of decoder with 32-bit error.

Table 2. Summary of error detection and error correction capabilities of orthogonal codes.

Input bits (a)	Orthogonal code length (b)	No of errors that can be detected (t)	%of errors detected and corrected	No of clk cycles required for decoding (n)
4	8	1	93.75	5
5	16	3	99.95	9
6	32	7	99.99	17
7	64	15	99.999	33
8	128	31	99.9999	65
a	$b = 2^{a-1}$	$t = \frac{b}{4} - 1$	$\% = \frac{2^b - 2b}{2^b}$	$n = \frac{b}{2} + 1$

Table 3. Summary of transmitter and receiver delay.

Input data bits	Encoder delay (ns)	Encryptor delay (ns)	Transmitter delay (ns)	Decryptor delay (ns)	Decoder delay (ns)	Receiver delay (ns)
4	4.010	7.354	7.354	7.354	8.834	7.504
5	4.010	7.354	7.354	7.354	14.327	13.960
6	4.010	7.354	7.354	7.354	24.385	26.158
7	4.010	7.354	7.354	7.354	56.147	59.381
8	4.010	7.354	7.354	7.354	62.970	66.993

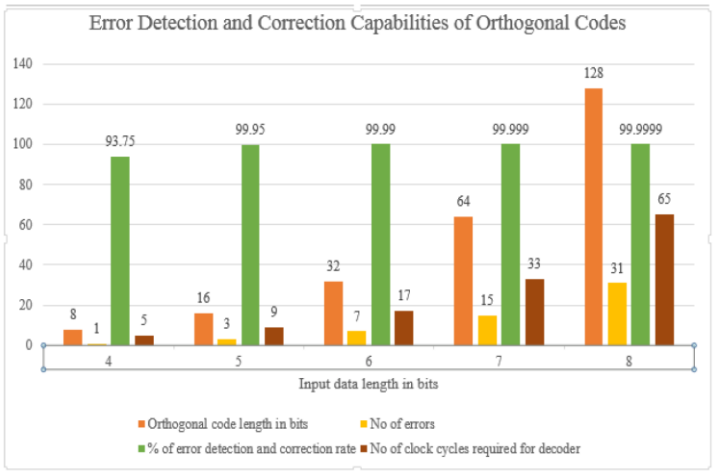

Figure 14. Graphical representation of error detection and correction capabilities of orthogonal codes.

6. Results

The table-2 shows the simulation results of encoder and decoder for input a-bit data, corresponding b-bit orthogonal code. The b-bit orthogonal code is able to detect the erroneous combination other than the combinations of original orthogonal code combinations in lookup table. Hence the % of error detection and correction is given by $\frac{2^b - 2b}{2^b}\%$ and the number of errors that can be detected is upto $\frac{b}{4} - 1$. For example if the received orthogonal code

length is 16-bit then the % of error detection and correction is $\frac{2^{16} - 2 * 16}{2^{16}} = 99.95\%$, and it can be able to correct upto $\frac{16}{4} - 1 = 7$. Similarly the % of error detection and correction capabilities are improved to 99.9999% and the number of errors that can be detected is increases as the orthogonal code length as shown in figure 14.

The table-3 shows the time delays of the encoder, encryptor, transmitter, decryptor, decoder and receiver in nano seconds. The simulation results are obtained for 4-bit, 5-bit, 6-bit, 7-bit and 8-bit input data. The delay timings of all these blocks are obtained during synthesis of code. All these blocks are implemented on Spartan 3EFPGA hardware board. The table-3 shows that time required to encrypt and decrypt the data for different data inputs is invariable. This shows that the proposed system is free from the timing attacks.

7. Conclusion

The results of the proposed technique are obtained for 4-bit, 5-bit, 6-bit, 7-bit and 8-bit input data. The length of the orthogonal code has been increases as the length of the input data increases. Hence it can be able to detect the errors up to $\frac{b}{4} - 1$ bits and the percentage of detection rate is improved up to 99.9999%. The proposed technique shows that the execution time of encryption and decryption do not vary even though there is change in data inputs.

References

[1] M. T. Siponen and H. Oinas-Kukkonen, A review of information security issues and respective research contributions, ACM Sigmis Database 38 (1) (2007), 60-80.

[2] G. Singh and Supriya, A study of encryption algorithms (RSA, DES, 3DES and AES) for information security, Int. J. Comput. Appl. 67(19) (2013), 33-38.

[3] W. Burr, Selecting the advanced encryption standard, IEEE Secur. Priv. 1(2) (2003), 43-52.

[4] R. L. Rivest, A. Shamir and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 21(2) (1978), 120-126.

[5] C. W. Chiou, C.-Y. Lee, J.-M. Lin, T.-W. Hou and C.-C. Chang, Concurrent error detection and correction in dual basis multiplier over GF (2^m), IEEE IET Circuit, Devices and Systems 3(1) (2009), 22-40.

[6] Shalini Sharmal and P. Vijayakumar, An HVD Based Error Detection and Correction of Soft Errors in Semiconductor Memories Used for Space Applications, IEEE 2012 International Conference on Devices, Circuits and Systems (2002), 563-567.

[7] N. Julai and A. Bystrov, Error Detection and Correction of Single Event Upset (SEU) Tolerant Latch, IEEE Proceedings of the 18th Symposium on On-line Testing (2012), 1-6.

[8] Bernhard Fechner, Fast online error detection and correction with thread signature calculate, Science Project - Micro Processors and Micro Systems 36 (6) (2012), 462-470.

[9] Debiao He and Sherali Zeadally, An Analysis of RFID Authentication Schemes for Internet of Things in Healthcare Environment Using Elliptic Curve Cryptography, IEEE Internet of Things Journal 2(1) (2015), 72-83.

[10] Newton G. Bretas and Arturo S. Bretas, A two steps procedure in state estimation gross error detection, identification, and correction, International Journal of Electrical Power and Energy Systems 73 (2015), 484-490.

[11] S. K. Hafizul Islam, Ruhul Amin, G.P. Biswas, Mohammad Sabzinejad Farash, Xiong Li and Saru Kumari, An improved three party authenticated key exchange protocol using hash function and elliptic curve cryptography for mobile-commerce environments, Journal of King Saud University-Computer and Information Science, 2015.

[12] Mustafa Demirci, Pedro Reviriego and Juan Antonio Maestro, Implementing Double Error Correction Orthogonal Latin Squares Codes in SRAM-based FPGAs, Micro Electronics Reliability 56 (2016), 221-227.

[13] Chi-Yuan Chen, Guo-Jyun Zeng, Fang-Jhu Lin, Yao-Hsin Chou and Han-Chieh Chao, Quantum Cryptography and Its Applications over the Internet, IEEE Network 29(5) (2015), 64-69.

[14] Miloš Krstić, StefanWeidling, Vladimir Petrović and Egor S. Sogomonyan, Enhanced architectures for soft error detection and correction in combinational and sequential circuits, Micro Electronics Reliability 56 (2016), 212-220.

[15] Rawya Rizk and Yasmin Alkady, Two-phase hybrid cryptography algorithm for wireless sensor networks, Journal of Electrical Systems and Information Technology 2(3) (2015), 296-313.

[16] Sheng Yuan, Jianbin Yao, Xuemei Liu, Xin Zhou and Zhongyang Li, Cryptanalysis and security enhancement of optical cryptography based on computational ghost imaging, Optics Communications 365 (2016), 180-185.

[17] Claudio Urrean, Claudio Morales and John Kern, Implementation of error detection and correction in the Modbus-RTU serial protocol, International Journal of Critical Infrastructure Protection 15 (2016), 27-37.

[18] Claudio Urrea, Claudio Morales and Rodrigo Muñoz, Design and implementation of an error detection and correction method compatible with MODBUS-RTU by means of systematic codes, Measurement 91 (2016), 266-275.

[19] Stjepan Picek, Claude Carlet, Sylvain Guilley, Julian F. Miller and Domagoj Jakobovic, Evolutionary Algorithms for Boolean Functions in Diverse Domains of Cryptography, Evolutionary Computation 24 (2016), 667-694.

[20] Hossein Moradian, Jeong-A Lee and Adnan Hashmi, Self-repairing radix-2 signed-digit adder with multiple error detection, correction, and fault localization, Microelectronics Reliability 54 (2014), 1-7.

[21] Alavalapati Goutham Reddy, Ashok Kumar Das, Eun-Jun Yoon and Kee-Young Yoo, A Secure Anonymous Authentication Protocol for Mobile Services on Elliptic Curve Cryptography, IEEE Access 4 (2016), 4394-4407.

[22] Donald G. MacKay and Laura W. Johnson, Errors, Error Detection, Error Correction and Hippocampalregion Damage: Data and Theories, Neuropsychologia 8 (2013), 1-9.

[23] Praveen Singh, Vaibhav Neema, Shreeniwas Daulatabad and Ambika Prasad Shah, Subthreshold Circuit Designing and Implementation of Finite Field Multiplier for Cryptography Application, Procedia Computer Science 79 (2016), 597-602.

[24] Yibin Li, Keke Gai, Longfei Qiu, Meikang Qiu and Hui Zhao, Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences 387 (2016), 103-115.

[25] Debiao He, Sherali Zeadally, Neeraj Kumar and Wei Wu, Efficient and Anonymous Mobile User Authentication Protocol Using Self-Certified Public Key Cryptography for Multi-Server Architectures, IEEE Transactions on Information Forensics and Security 11(9) (2016), 2052-2064.

[26] Satyanarayana Vollala and N. Ramasubramanian, Energy efficient modular exponentiation for public-key cryptography based on bitforwarding techniques, Information Processing Letters 119 (2016), 25-38.

[27] Quinn Du Pont and Bradley Fidler, Edge Cryptography and the Co-development of Computer Networks and Cybersecurity, IEEE Annals of the History of Computing 38(4) (2016), 55-73.

[28] T. Narendra Babu, Fazal Noorbasha and Leenendra Chowdary Gunnam, Implementation of High Security Cryptographic System with Improved Error Correction and Detection Rate using FPGA, International Journal of Electrical and Computer Engineering 6(2) (2016), 602-610.

[29] T. Narendra Babu, M. Fazal Noorbasha, N. Harita, Tejashree and K. Vamsi Krishna, FPGA Implementation of High Speed Error Detection and Correction of Orthogonal Codes using Segmentation Method, Indian Journal of Science and Technology 9(30) 2016.

[30] T. Narendra Babu, Fazal Noorbasha, Ch. Sai Krishna, K. Sai Charan and R. S. V. S. Sai Kalyan, FPGA Implementation Of Cryptographic System Using Bodmas Sequence of Operations, ARPN Journal of Engineering and Applied Sciences 11(19) (2016), 11475-11479.