Circuits and Systems, 2016, 7, 1250-1258
Published Online June 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.78109

FPGA Implementation of Non-Linear Cryptography

Thammampatti Natarajan Prabakar[1], Balasubramanian Lakshmi[2], Gopalakrishnan Seetharaman[1]

[1]Oxford Engineering College, Trichy, India
[2]Researcher, Trichy, India
Email: tnprabakar@gmail.com, lakshminmc@gmail.com, jgsraman@gmail.com

Received 13 March 2016; accepted 10 April 2016; published 7 June 2016

Abstract

The paper focuses on the design and Field Programmable Gate Array (FPGA) implementation of embedded system for time based dual encryption scheme with Delay Compulsion Function (DCF) and also illustrates the application of DCF in time based cryptography. Further, the strength of the time based FPGA encryption algorithm with and without using DCF is analyzed using a Nios II processor. This proposed scheme enhances the security of vital data against Brute force attack by incorporating a temporal key distribution where two different keys encrypt the data simultaneously, one being the regular key and the other being the time. The time is included using a dynamically varying number of shifts thereby allowing the system to wait for the duration and this forms the second dimension of the key. Presently, available encryption systems suffer from Brute Force attack in which all the key combinations are tried in order to find the correct key. In such a case, the time taken for breaking the key depends on the speed of the system used for cryptanalysis. The proposed system adds complexity by using dynamically varying sequence of operations, by including the time as a second dimension of the key besides minimizing the possibility of Brute Force attack and increasing the time required for cryptanalysis irrespective of the system capability. As the proposed system needs concurrent execution and real time processing, the system is implemented using Altera Stratix II FPGA and the results are presented.

Keywords

FPGA, Encryption, Delay Compulsion Function, Cryptanalysis, Brute Force Attack

1. Introduction to Cryptography

Cryptography has emerged as a fundamental building block of computer security and is the science of keeping

How to cite this paper: Prabakar, T.N., Lakshmi, B. and Seetharaman, G. (2016) FPGA Implementation of Non-Linear Cryptography. *Circuits and Systems*, **7**, 1250-1258. http://dx.doi.org/10.4236/cs.2016.78109

data secure which is used both in system level and in network level using various encryption algorithms. It protects information from unauthorized or accidental disclosure while the information is in transit or in storage, using two processes called encryption (data locking with the key) and decryption (data unlocking with the key).

Information security demands increase in the strength of existing encryption algorithms due to rapid improvements in processing capabilities. Also the encryption algorithms are effectively implemented both on software and hardware platforms. Even though the software implementation of encryption algorithms has the advantages of portability, flexibility, and ease of use, it provides a limited physical security and agility compared to hardware implementations. The advantages of hardware implementation include less power consumption, small circuit size, hardware reconfiguration, cost effectiveness, high operating speed and security. That is, the implementation of software algorithms as hardware using FPGAs or Application Specific Integrated Circuits (ASICs) further reduces the time taken for cryptanalysis [1].

1.1. Cryptanalysis and Cryptanalytic Techniques

Cryptanalysis is the science of analyzing and breaking the strength of an algorithm. This is achieved by attacking the encryption system to recover the key in use rather than simply to recover the plaintext of a cipher text. Cryptanalysts are also called as attackers. The cryptanalysis of encryption algorithms consumes a significant amount of computing resources such as CPU time, memory, and battery power [2]. The security of encrypted data is entirely dependent on two things, the strength of the cryptographic algorithm and the secrecy of the key. The basic and common cryptanalysis technique an intruder tries is brute force attack [3]. It involves traversing the search space of all possible keys until the correct key is found. For example, if there are 2^{64} possible keys, a brute force attack is on average, expected to find a key after 2^{63} trials [4]. The strength of an encryption algorithm is evaluated based on the key size and is given by the formula $2^{\text{key-size}}$, so a 56-bit key requires 65,536 (2^{56}) combinations [5].

1.2. Weak Keys of IDEA/Attacks on DES/IDEA

The very simple key schedule makes IDEA subject to a class of weak keys [6]. To avoid keys containing a large number of 0 bits, a simple fix was proposed by XORing each sub key with a 16-bit constant, such as 0 × 0DAE. The problem with the runs of ones' will require complete redesigning of IDEA key schedule [7]. It is proved that the problem of weak keys can be eliminated by slightly modifying the key schedule of IDEA [8]. A chosen-key differential attack on 3-round IDEA recovers 32 bits of the key using six chosen plaintexts two under first key, four under second key [9] [10]. The only confirmed DES cracker was the COPACOBANA [11] [12] machine, which consists of 120 reconfigurable commercially available FPGAs. COPACOBANA finds the right key in approximately 8.7 days using brute force attack.

1.3. Field Programmable Gate Arrays (FPGA)

FPGAs are an array of programmable logic cells interconnected by a matrix of wires and programmable switches. Each cell performs a simple logic function defined as per the user instructions. An FPGA has a large number (64 to over 20,000) of these cells that are used as building blocks in complex digital circuits. FPGA is a hardware, which is (Re) configurable [13]. The main advantage of FPGAs is concurrent processing, where all the hardware modules work concurrently. This enables any number of modules implemented on FPGA to run simultaneously [14].

1.4. Issues in Conventional Symmetric Algorithms

The conventional symmetric algorithms suffer from the following issues.

1) The algorithmic strength purely depends on the key size used for encryption.
2) The algorithms are vulnerable to brute force attack and a proper cryptanalysis can easily bring out the message content easily. This is true because, in most cases, in a random code space of 'n' sets, the key can be extracted using "n/2" sets of data. That is, the expected number of trials before finding the correct key is equal to half of the size of the key space. Also, the cryptanalysis is purely dependent on the speed of the system used.
3) The linear cryptanalysis is feasible in most of the cases because almost all algorithms possess a linear relationship between inputs and outputs except S-boxes in DES.

4) The algorithms include a set of weak keys (a sequence of consecutive 1's or 0's) that further weakens the algorithm [15].
5) The same process of encryption is repeated for a predetermined number of rounds irrespective of the significance of data.
6) The continuous increase in the key size makes the cryptanalysis difficult and leads to computational and resources overhead.
7) The invention of dedicated embedded devices like FPGA or ASIC further reduces the time taken for cryptanalysis as they provide a hardware implementation of the software.

To overcome the above issues, a scheme is initially proposed in which the cryptanalysis using brute force attack is independent of the speed of the system used and the time taken to break the algorithm is increased [16]. To achieve this, a temporal key validation is introduced, where the time is taken as the second dimension of the key and both data and time validation are performed for successful decryption. This scheme requires a real time system to interpret the manual time based key inputs. Since the manual key entry is not a realistic solution in system security, the new proposed scheme uses a FPGA that replaces manual key entry and the need for a real time system.

The rest of this paper is organized as follows. In Section 2, a short background about the time based cryptography is presented. The proposed DCF is described in Section 3. Application of DCF in FPGA based encryption and decryption is discussed in Section 4. The cryptanalysis of the system with and without DCF is presented. In Section 5 the paper is concluded.

2. Time Based Dual Encryption

In this scheme, the encryption is similar to conventional schemes except that two random numbers are used one for selecting the number of rounds each data set to be operated and the other for selecting the order of execution of functions. The key is composed of the text and the time interval. The decryption system expects the right key to be entered at the right time. The process in this case mimics a real time system. When this scheme is implemented on a desktop computer, it requires real time operating system to keep track of key entry times.

Since the manual key entry is not a realistic solution in system security, the proposed scheme uses FPGA that replaces manual key entry and the need for a real time system.

3. Delay Implementation - Delay Compulsion Function (DCF)

3.1. Need for DCF

The proposed scheme uses FPGA to perform encryption (or decryption) and key management. The concurrency in the FPGA processing is exploited to emulate real time systems that can perform both key processing and encryption (or decryption) simultaneously [17]. The second dimension of the key, the time is incorporated as a delay using a dynamic delay compulsion function. For validation of time interval, a Delay Compulsion Function (DCF) is proposed.

3.2. Definition for DCF

A delay compulsion function is one whose output should progress through the number of clock cycles without getting the results before the designated number of clock cycles. Linear Feedback Shift Register (LFSR) based Pseudo random number generator output XORed with key with the seed dictated by the key itself can be used as a delay compulsion function.

In case of FPGA based delay generation for temporal validation, a simple key rotation is used. That is for generating 10 units of delay, the key is rotated by 10 bits, one bit per clock so that after 10 clock cycles time consumed is 40 seconds and the original key is rotated by 10 times either clockwise or anticlockwise. However, in actual implementation, the ten-bit rotation can be done in a single clock cycle so that the time dependency is not compelled. Hence, a delay compulsion function is introduced. **Figure 1** is the Verilog code for DCF that is used for this implementation.

The implementation details of Delay compulsion function on Cyclone II device is explained below.

1) The process begins by providing an actual key and the required delay to the DCF.
2) It has an in built LFSR pseudo random number generator and the key is initialized as the seed.

3) For every clock, LFSR generates new pseudo random number (PNR). A counter is also incremented.
4) When the counter value reaches the required delay, the LFSR contents are XORed with key and thus new key is given out.

4. Application of DCF in FPGA Encryption

4.1. Scrambler Function

Weak keys constitute sequences of consecutive 1's, 0's or both, which lead to weaken the strength of an algorithm. The weak keys minimize the time taken for cryptanalysis. Therefore, to eliminate the weak keys, the scheme uses a scrambler function [18].

A **s**crambler is a function that eliminates the repetitive pattern in the input data stream and is mainly used to eliminate long sequences consisting of consecutive 1's and 0's. The scrambler can be implemented as either serial or parallel data processing. A serial scrambler with polynomial as $1 + x^{14} + x^{15}$ is shown in **Figure 2** [19].

4.2. Description of FPGA Based Dual Encryption

The algorithm accepts a plain text of 64 bits and key of 64 bits as the inputs. The size of plain text block and key can be changed with ease as FPGA is reprogrammable. The original key is initially given to a scrambler function to get rid of weak keys.

The algorithm picks out four-bit positions (can be standardized) in the key and concatenates the bits. This concatenated number and the original key is given to delay compulsion function. For example, if the concatenated value equals 10 (say 1010), the delay compulsion function gives out modified key after 10 clock cycles. It also ensures the same result cannot be obtained in less than 10 clock cycles so that time is consumed based on the time bits selected. This value is taken as the delay and the system needs to wait for that time. A counter is used to check whether the system's delay reaches the timer value. Further 8 bit positions are selected in which the concatenation of any two bits correspond to the encryption (or decryption) function that is to be executed in each round. **Figure 3** shows the execution flow of the proposed algorithm.

```
module dcf(clk, start, keyin, keyout, delay);
      input clk, start; input [7:0] keyin; input [7:0] delay;
      output reg [7:0] keyout; reg [7:0] lfsr;
      reg [7:0] count; reg xorout;
      always@ (posedge clk)
      if(!start) begin lfsr = keyin; count = 8'd0; end
      eles
        begin
        count = count + 1'b1;
        xorout = lfsr[0]^lfsr[1];
        lfsr = {xorout, lfsr [7:1]};
        if(count == delay) keyout = keyin^lfsr
        end
endmodule
```

Figure 1. Verilog code for DCF.

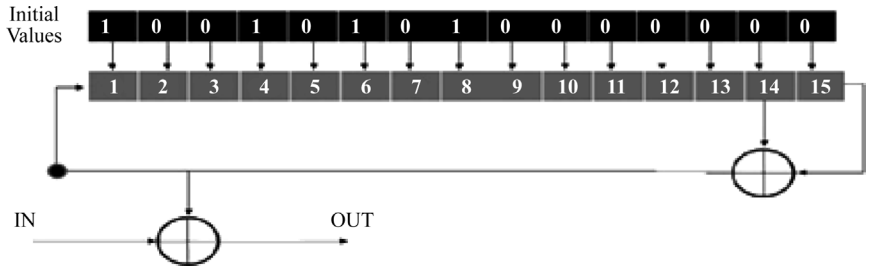

Figure 2. Scrambler function.

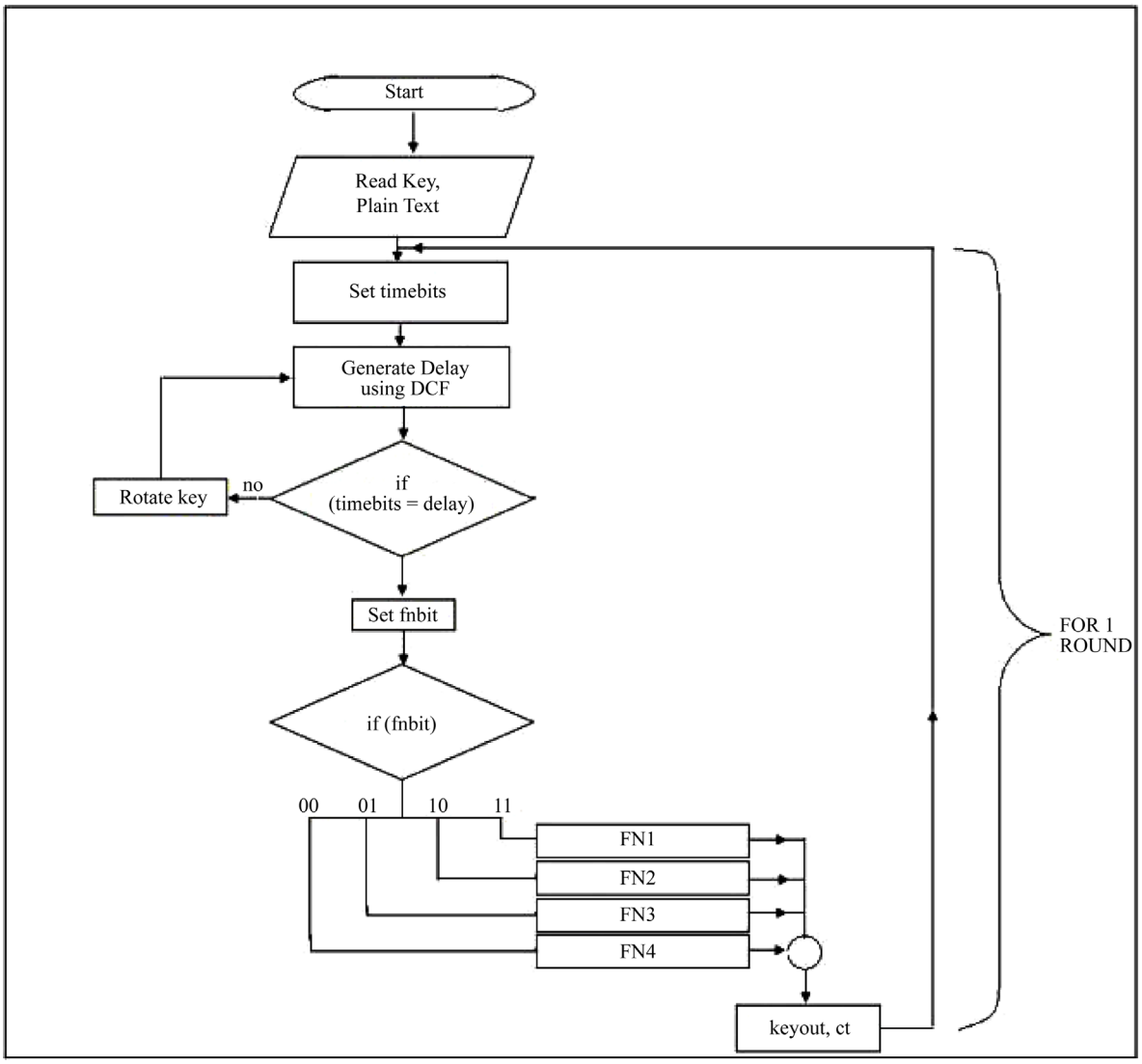

Figure 3. Execution flow of proposed system.

4.3. Encryption Algorithm

The algorithm consists of the below steps.

1) Read 64-bit plain text and 64-bit key as inputs.
2) The key is passed through a scrambler function [9] to avoid weak keys.
3) Select four bits from the key. Assume that the bit positions 12, 24, 36, 48 are chosen but it can also be standardized. The concatenation of these four bits form 'time bits'.
4) For every clock, the key is rotated in the anti-clockwise and a counter is incremented by one.
5) When the counter value is equal to the 'time bits' the key is ready for the next round. As the key rotates continuously, the four bits drawn from the key varies and hence the number of rotations varies. Hence, the time taken for processing the key also varies. This forms the time based key concept. Here, number of key rotations is determined using the delay compulsion function.
6) Now, the rotated key of 64 bits and plain text of 64 bits enter in to the round one. Eight bits namely fnbit1, fnbit2, fnbit3 and fnbit4 each of size two bits are extracted from the key and correspond to the order of execution of the four functions.
7) In conventional algorithms, the number of functions and the order of execution of functions remain fixed but in the proposed algorithm, based on the selected bits, the order of execution of functions is varied.

8) In the proposed algorithm, four functions are used. Function 1 is "XOR"ing key and plain text bits, Function 2 performs modulo 2 addition, Function 3 is "XOR"ing the plain text and the key followed by gray code conversion and Function 4 is shuffling the plain text bits.
9) If the fnbit1 is "00" Function 1 is executed first and if the fnbit1 is "01" Function 2 is executed first and so on. This is step-1 in the round one.
10) After this, the new "keyin" and the output of round one is given to round two. In round two, again the four bits from locations 12, 24, 36, 48 are taken out from the new rotated key and key is rotated by that amount. For an intruder, since the total key is unknown, the number of key rotations and selection of functions at each step is unknown. This forms a clause of continuous 'S' box architecture where it is difficult to establish a relation between the input and the output.
11) This procedure is repeated for eight rounds.

Since the original key is not known, the intruder doesn't know the number of times the bits are to be shifted in each round, the function used in first round and also the key used for second round. Therefore the possible number of combinations for brute force attack automatically increases. In case of fixed number of rotations, the full rotation can be done in a single clock. But in this scheme since the number of shifts is dynamically varied, a single clock shift is not possible that leads to a complex cryptanalysis.

4.4. Decryption Algorithm

The decryption process is merely the inverse of the encryption process. The receiver has two parameters to extract the original plain text. One is the final cipher text and the second is the original key. Since the intended receiver is aware of the key as well as the standardized bit positions say 12, 24, 36, 48 the key is initially passed through a scrambler function, rotated for the concatenated value times and the key for the next round is generated in the same way. In addition the receiver also picks out the function bits to know the functions executed in round one. The procedure is repeated for all rounds for key generation and execution of functions. To perform decryption the last round key and the cipher text comes out of the last round is used. Since all the functions are reversible, it is possible to go back and generate the plain text.

4.5. Implementation of Proposed Scheme

The proposed scheme is implemented on an Altera Stratix II (EP2S130F1020C4) device as shown in the compilation summary in **Figure 4**.

The system consumes 5038 combinational ALUTs and 1609 dedicated logic registers. **Figure 5** shows the FPGA chip editor view of proposed scheme.

4.6. Advantages

1) Unlike the conventional encryption algorithms where the flow of operational sequence is fixed, the sequence of operations to be carried out for each round remains changed.
2) The two processes such as encryption (or decryption) and key generation are performed concurrently without using a real time system.

4.7. Disadvantages

In this scheme FPGA performs key generation as well as encryption (or decryption). Hence the overhead increases for bulk data transfer between FPGA and computer. Since FPGA can handle only limited amount of data, the scheme is modified in such a way that the key scheduling is carried out by FPGA to generate right key at right time intervals which is then connected to a Nios II processor that communicates the keys to the personal computer through JTAG communication. Hence, the FPGA implemented with the key scheduler algorithm is used as a hardware key (dongle) for software encryption [17].

5. Nios II Based Cryptanalysis for Dual Encryption with DCF

To evaluate the strength of the time based encryption algorithm, the algorithm is implemented on FPGA and Nios II processor is used to start and stop the process. The processor also computes the time consumed for performing a complete brute force attack of time based dual encryption (Nios II is a virtual Microcontroller, that

can be brought out of any ALTERA brand FPGA) [20]-[22].

The Nios II processor is custom made to have one bit output and one bit input. The input is connected through Interrupt line. The scheme is designed for a 4 bit key and hence, the 16 combinations are stored in the memory. Cryptosystems have the program for the encryption (or decryption). The address generator generates the address. Based on the address, the key will be given out. The key is passed to the delay compulsion function. The delay is again determined from the key internally. Then the key is given to cryptosystem, and plain text is provided from an external pin outs. Nios II processor works at a lower speed around 150 MHz while the cryptosystem works around 390 MHz. Hence, a Phase Locked Loop is provided for frequency multiplication. When all the data is over, the address bus will have 16 at the output, which is connected to Nios II processor as an end signal. **Figure 6** shows the Nios II based Analyzer.

The following code is executed in the Nios II processor. The program flow is as follows.

1) Start pulse is given to the delay compulsion function unit and to the cryptosystem.
2) The first key is fetched from the memory. The value is "0000".
3) This key is passed through the delay compulsion function and is given to the cryptosystem. The system consumes only one clock cycle at around 390 MHz. The address generator operates at around 150 MHz. So, no collision of data occurs.

Flow Status	Successful - Fri Nov 20 22:16:25 2009
Quartus II Version	8.1 Build 163 10/28/2008 SJ Full Version
Revision Name	gujarat
Top-level Entity Name	gujarat
Family	Stratix II
Device	EP2S130F1020C4
Timing Models	Final
Met timing requirements	Yes
Logic utilization	1%
Combinational ALUTs	5.038/106.032 (5%)
Dedicated logic registers	1.609/106.032 (2%)
Total registers	1609
Total pins	258/743 (35%)
Total virtual pins	0
Total block memory bits	0/6.747.840 (0%)
DSP block 9-bit elements	0/504 (0%)
Total PLLs	0/12 (0%)
Total DLLs	0/2 (0%)

Figure 4. Compilation report of proposed scheme.

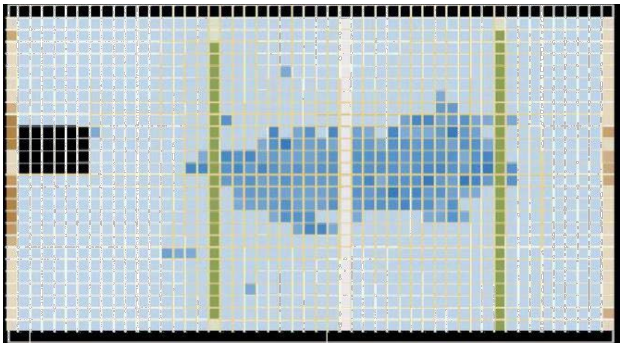

Figure 5. FPGA chip editor view of proposed scheme.

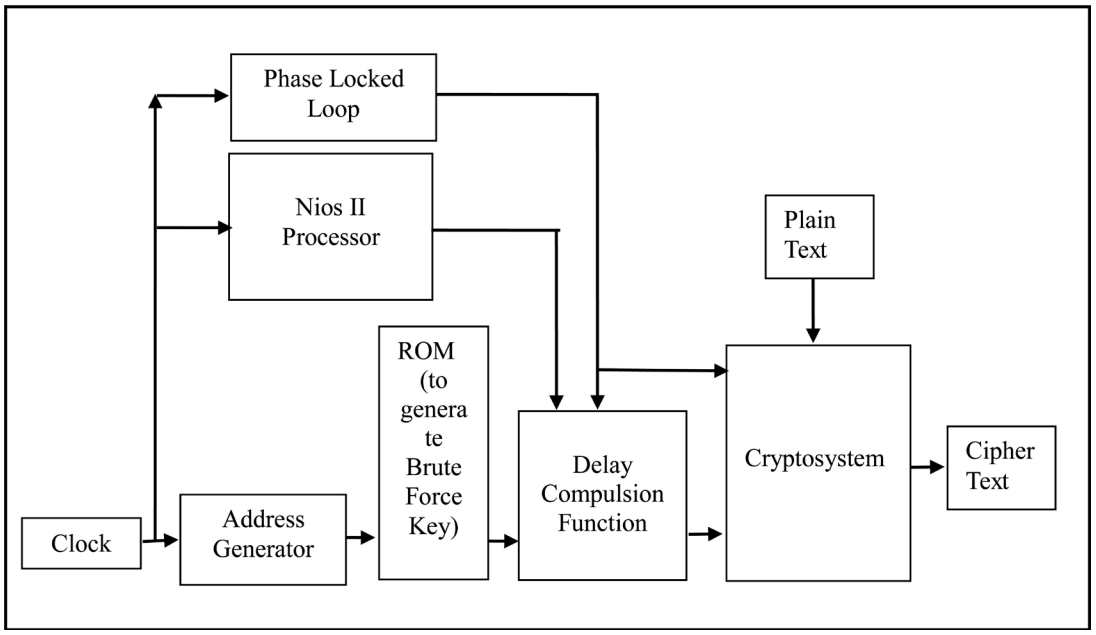

Figure 6. Nios II based analyzer.

Dual encryption without DCF	Dual encryption with DCF
timer function ok stared running timer over head 115 Number of clocks 5973	timer function ok stared running timer over head 114 Number of clocks 4294955561

Figure 7. Results of Nios II based analyzer.

4) The continuous clocks are bringing out the contents of ROM and all the possible combinations from "0000" to "1111" are given to the cryptosystem. Once the address value reaches "1111", the address unit generates a stop pulse to the Nios II processor. The Nios II processor in the meanwhile counts the clock signals and displays the delay consumed.

Nios II processor displays the time consumed when the FPGA based dual encryption is implemented with or without DCF. It is clear that when the second dimension time is incorporated, the time taken for cryptanalysis is increased. Also, the delay is made compulsory using the DCF. The displayed results of Nios II based Analyzer are shown below in **Figure 7**.

6. Results and Conclusion

In this paper, a scheme is proposed to strengthen conventional cryptographic algorithms against brute force attack and is achieved using time as a second dimension of the key. In this paper, the strength of the time based FPGA encryption algorithm with and without DCF is analyzed and it is seen that the dual encryption algorithm with DCF takes additional time for cryptanalysis and is independent of the speed of the system used due to dynamic selection of functions, dynamic number of rotations and DCF. The DCF also ensures the second dimension of the key, and the time is incorporated in terms of dynamic number of shifts. It is also proved from the results that the inclusion of time using DCF leads to complex cryptanalysis.

References

[1] Güneysu, T. (2011) Utilizing Hard Cores of Modern FPGA Devices for High-Performance Cryptography. *Journal of Cryptographic Engineering*, **1**, 37-55. http://dx.doi.org/10.1007/s13389-011-0002-2

[2] Salama, D., *et al*. (2008) Performance Evaluation of Symmetric Encryption Algorithms. *International Journal of Computer Science and Network Security*, **8**, 280-286.

[3] Stallings, W. (2003) Cryptography and Network Security. 3rd Edition, Prentice Hall, Upper Saddle River.

[4] Schneier, B. (1996) Applied Cryptography. 2nd Edition, John Wiley & Sons Publications, Hoboken.

[5] Mousa, A. and Hamad, A. (2006) Evaluation of the RC4 Algorithm for Data Encryption. *International Journal of Computer Science &Applications*, **3**, 44-56.

[6] Elbaz, L. and Bar-EI, H. (2000) Strength Assessment of Encryption Algorithm. Whitepaper.

[7] Daemen, J., Govaerts, R. and Vandewalle, J. (1993) Weak Keys for IDEA. Advances in Cryptology, Crypto '93, Springer-Verlag, 224-231.

[8] Biryukov, A., *et al.* (2002) New Weak-Key Classes of IDEA. *ICICS Proceedings of the* 4*th International Conference*, Singapore, 9-12 December 2002, 315-326. http://dx.doi.org/10.1007/3-540-36159-6_27

[9] Kelsey, J., *et al.* (1996) Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. *Advances in Cryptology*, **1109**, 237-251. http://dx.doi.org/10.1007/3-540-68697-5_19

[10] Buell, D.A. and Pocek, K.L. (1995) Custom Computing Machines: An Introduction. *The Journal of Supercomputing*, **9**, 219-229. http://dx.doi.org/10.1007/BF01212869

[11] Güneysu, T., *et al.* (2008) Cryptanalysis with COPACOBANA. *IEEE Transactions on Computers*, **57**, 1498-1513.

[12] Chen, J., *et al.* (2008) An Analysis of International Data Encryption Algorithm (IDEA) Security against Differential Cryptanalysis. *Journal of Natural Sciences*, **13**, 697-701.

[13] Nawari, M., *et al.* (2015) FPGA Based Implementation of Elliptic Curve Cryptography. World Symposium on Computer Networks and Information Security (WSCNIS).

[14] Deshmukh, P. (2014) Modified AES Based Algorithm for MPEG Video Encryption. IEEE International Conference on Information Communication and Embedded Systems (ICICES), Chennai, 27-28 February 2014, 1-5. http://dx.doi.org/10.1109/icices.2014.7033928

[15] Kumar, P., Joseph, J. and Singh, K. (2016) Double Random Phase Encoding Based Optical Encryption Systems Using Some Linear Canonical Transforms: Weaknesses and Countermeasures. *Springer Series in Optical Sciences*, **198**, 367-396. http://dx.doi.org/10.1007/978-1-4939-3028-9_13

[16] Lakshmi, B., *et al.* (2008) Real Time Cryptography with Dual Key Encryption. *IEEE International Conference on Computing*, *Communication and Networking ICCCN*'08, St. Thomas, 18-20 December 2008, 1-4. http://dx.doi.org/10.1109/icccnet.2008.4787696

[17] Lakshmi, B., *et al.* (2010) FPGA Based Hardware Key for Temporal Encryption. *ICTACT Journal on Communication Technology*, **3**, 150-156.

[18] Daemen, J., *et al.* (1994) Weak Keys for IDEA. *Advances in Cryptology*, 13*th Annual International Cryptology Conference*, **773**, 224-231.

[19] PCI Express 16-Bit Parallel Scrambler/De-Scrambler, WhitePaper. http://www.actel.com/ipdocs/scrambler16_PB.pdf, PCI Express 16 Bit Parallel Scrambler/De-scrambler, Whitepaper
http://www.actel.com/ipdocs/scrambler16_PB.pdf, PCI Express 16 Bit Parallel Scrambler / De-scrambler, Whitepaper

[20] Altera Documentation Library (2003 and 2007) Altera Corporation, USA.

[21] Altera (2008) Application Note 507. Implementing PLL Reconfiguration in Cyclone III Devices, Ver 1.0.

[22] www.altera.com