Scientific Research Publishing

Circuits and Systems, 2017, 8, 237-246
http://www.scirp.org/journal/cs
ISSN Online: 2153-1293
ISSN Print: 2153-1285

A Fast FPGA Implementation for Triple DES Encryption Scheme

Edni Del Rosal, Sanjeev Kumar

IEEE Network Security Research Lab, Department of Electrical/Computer Engineering, The University of Texas Rio Grande Valley, Edinburg, USA
Email: sj.kumar@utrgv.edu

How to cite this paper: Rosal, E.D. and Kumar, S. (2017) A Fast FPGA Implementation for Triple DES Encryption Scheme. *Circuits and Systems*, **8**, 237-246.
https://doi.org/10.4236/cs.2017.89016

Received: June 18, 2017
Accepted: September 22, 2017
Published: September 25, 2017

Abstract

In cryptography, the Triple DES (3DES, TDES or officially TDEA) is a symmetric-key block cipher which applies the Data Encryption Standard (DES) cipher algorithm three times to each data block. Electronic payment systems are known to use the TDES scheme for the encryption/decryption of data, and hence faster implementations are of great significance. Field Programmable Gate Arrays (FPGAs) offer a new solution for optimizing the performance of applications meanwhile the Triple Data Encryption Standard (TDES) offers a mean to secure information. In this paper we present a pipelined implementation in VHDL, in Electronic Code Book (EBC) mode, of this commonly used cryptography scheme with aim to improve performance. We achieve a 48-stage pipeline depth by implementing a TDES key buffer and right rotations in the DES decryption key scheduler. Using the Altera Cyclone II FPGA as our platform, we design and verify the implementation with the EDA tools provided by Altera. We gather cost and throughput information from the synthesis and timing results and compare the performance of our design to common implementations presented in other literatures. Our design achieves a throughput of 3.2 Gbps with a 50 MHz clock; a performance increase of up to 16 times.

Keywords

Data Encryption Standard, Triple DES, DES, TDES, 3DES, Non-Pipelined, Pipelined, Cyclone II, FPGA, VHDL

1. Introduction

In cryptography, the Triple DES (3DES, TDES or officially TDEA) is a symmetric-key block cipher [1] which applies the Data Encryption Standard (DES) ci-

pher algorithm [2] three times to each data block. Electronic payment systems are known to use the TDES scheme for the encryption/decryption of data, and hence faster implementations are of great significance [3] [4]. Mail applications, such as Microsoft Outlook, make use of this scheme as well [5].

This paper focuses on increasing the performance of TDES, in Electronic Codebook (ECB) mode [6], by implementing a 48-stage pipelined depth design. In [7], a common design to increase the computational power (performance) of TDES is evaluated by implementing a 3-stage pipelined design. The pipeline stages are placed after each DES process and each DES process consists of one Feistel Function round. The input string must loop the one-round 16 cycles before the next input string can be fed. This implementation is common where cost constrain requirements are present.

Our approach to increase the performance consists on implementing a 48-stage pipeline TDES design. To do so, 3 different DES components, consisting of 16 Feistel Function rounds, are required. Each DES process must be pipelined at every round to make a 16-depth pipeline. Pipelining each DES component allows us to increase the depth to 48 stages and yield a higher throughput. An input string can be fed at every cycle and, as a consequence, a processed string will output at every cycle. To achieve the coherency between the 3 input keys and the data, as it traverses the stages, we design a key bank. This key bank properly buffers the keys to match each DES stage. The last design modification, for coherency, is incurred in the DES decryption key scheduler: the key scheduler performs right rotations instead of left rotations.

The structure of this paper is as follows: In Section 2, we detail the modifications made, to the TDES scheme presented in the NIST SP 800-67, which coherently pipelines TDES in ECB mode. Section 3 contains the performance and cost results as portrayed by the EDA tools and calculations based on the Cyclone II technology. We include a comparison subsection of the performance yield by the pipelined method implemented in [7] and the pipelined method implemented here. Lastly, Section 4 contains our conclusion.

2. TDES Pipelined Design

To pipeline our TDES design we take advantage of the 16 Feistel function rounds in DES. We pipeline after every Feistel function round. The pipeline is also applied to the key schedulers. A key bank buffers the 3 input keys so that, as the data traverses the stages, the proper keys and sub keys are fed. The pipeline depth of our DES design is 16 stages and the depth of our TDES design is 48 stages. The TDES scheme is designed as presented in [1]. Our modification to the scheme is the addition of registers after every Feistel Function round in DES, the right rotations in the DES decryption scheduler and the TDES Key Bank.

2.1. DES Algorithm

A coherent DES pipelined design is necessary for implementing the pipelined

TDES. The full description of the DES algorithm is presented in [2]. In this section, we show the pipeline at every stage in the DES algorithm.

The DES scheme is conformed of two permutations and 16 rounds of Feistel Functions. To pipeline DES, we add registers after every round and one last register following the final permutation. The DES component contains 16 stages in the pipeline. Seen in **Figure 1** are the 30 32-bit registers after every Feistel Function round (L1 through L15 and R1 through R15). The final register, following the final permutation, is the 64-bit buffer (cypher buffer). Both, the encryption and decryption components for DES are identical. The main difference between the encryption and decryption DES schemes are the order in which the 16 sub-keys, generated in the key schedulers, are inserted in the Feistel Function rounds.

2.2. DES Decryption Key Scheduler

The coherency requirement for the pipelined TDES involves applying buffers in the key scheduler. As the input data string traverses the rounds, the buffers ensures each round encrypts with the proper sub key. 16 sub keys are generated in the key scheduler. We apply 15 buffers in the schedulers. These 15 56-bit registers can be seen in **Figure 2** (Reg1, Reg2, Reg3… Reg15). The registers contain the left (cn) and the right (dn) halves. The key scheduler shown in **Figure 2** is employed in the DES decryption component. The main difference between the DES encryption scheduler and DES decryption scheduler is that the encryption

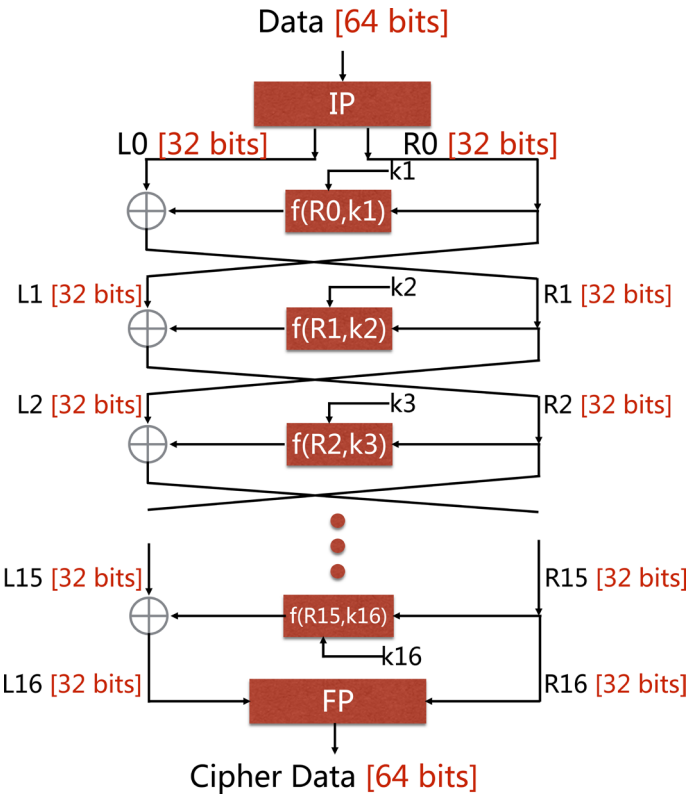

Figure 1. DES Encryption Pipeline.

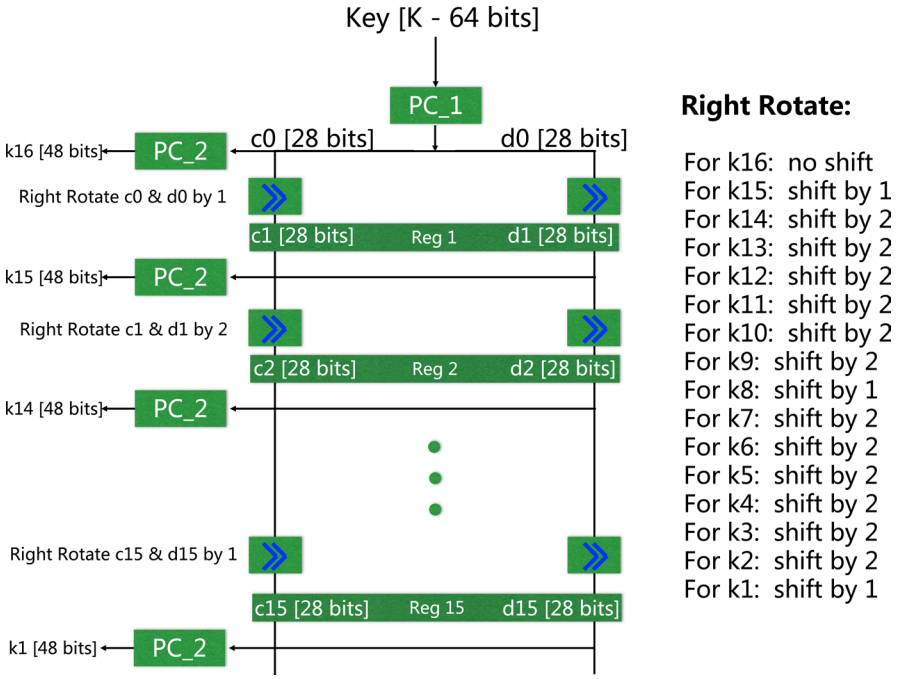

Figure 2. Key Scheduler Pipelined Design for Decryption.

Table 1. Key Scheduler Rotations.

DES Encryption Sub Key	Left Rotation	DES Decryption Sub Key	Right Rotation
K1	1	K16	No Shift
K2	1	K15	1
K3	2	K14	2
K4	2	K13	2
K5	2	K12	2
K6	2	K11	2
K7	2	K10	2
K8	2	K9	2
K9	1	K8	1
K10	2	K7	2
K11	2	K6	2
K12	2	K5	2
K13	2	K4	2
K14	2	K3	2
K15	2	K2	2
K16	1	K1	1

scheduler performs left rotations while the decryption scheduler performs right rotations. These rotations are executed in the cn and dn halves.

In Table 1 we show the positions by which the encryption and decryption schedulers perform the left and right rotations. Generating the 16 sub keys, by

performing right rotations in the decryption scheduler and feeding them in order, is equivalent to generating the sub keys by performing left rotations and feeding the keys to the decryption rounds in reverse order as specified in [2]. For pipelining, it is convenient to maintain the data and key coherency by inserting the sub keys in top to bottom order instead of bottom to top order.

2.3. Key Bank

TDES consists of 3 different DES components: DES encryption (DES1 e), DES decryption (DES2 d), DES encryption (DES3 e) for TDES encryption and DES decryption (DES1 d), DES encryption (DES2 e), DES decryption (DES3 d) for TDES decryption. TDES encryption is performed as follows: DES1 e (Key 1), DES2 d (Key 2) and DES3 e (Key 3). TDES decryption is performed as follows: DES1 d (Key 3), DES e (Key 2) and DES d (Key 1).

One difficulty faced with linking three pipelined DES components is that the 3 input keys (Key 1, Key 2, Key 3) don't map to the data as it traverses the DES components. The keys need to be properly buffered before they are inserted into their respective DES component. Otherwise DES2 d and DES3 e components will begin processing the incorrect data as soon as they are fed.

The concept behind our key bank is that the keys be buffered the proper cycles count until the output of the previous DES component reaches the input of the DES component for which the key was meant. See **Figure 3**.

For the TDES encryption we have Key 1, Key 2 and Key 3. Key 1 is inserted in to the encryption key scheduler and begin processing. There is no need to buffer Key 1 because the data enters the DES1 e component right away. However, Key 2 and Key 3 cannot begin processing right away. Key 2 waits until the

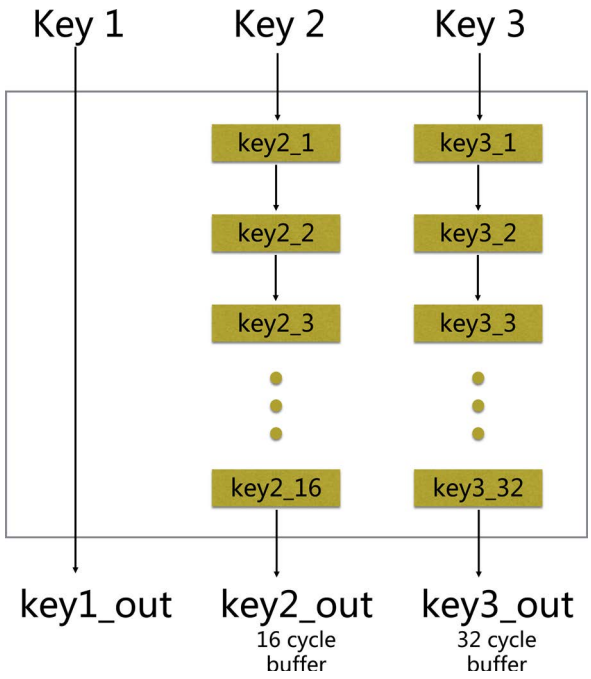

Figure 3. Key Bank.

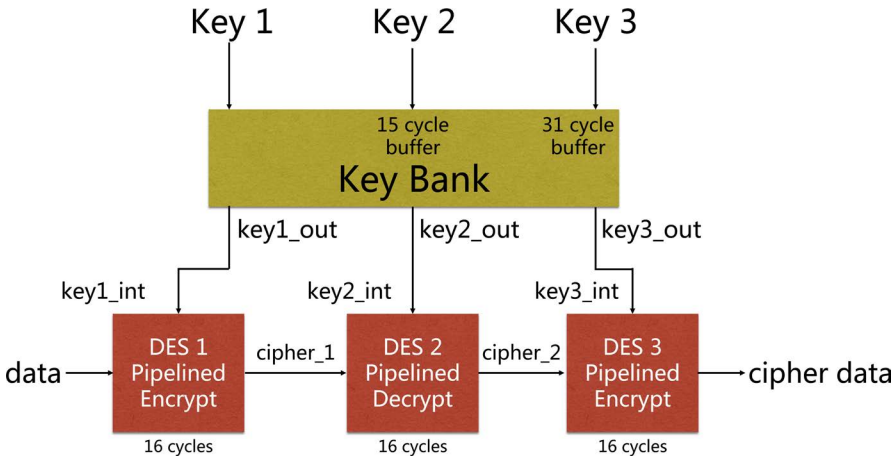

Figure 4. TDES Pipelined Design.

DES1 e component is done processing. As otherwise stated in **Figure 4**, Key 2 is buffered, 15 cycles, until data reaches cypher 1. This is done by implementing 15 registers (key2 1 ... key2 15) in the Key Bank. In the 16th cycle, Key 2 enters the decryption key scheduler just as cypher 1 enters DES2 d. Key 3 must wait 15 more cycles after that to begin processing. Key 3 is buffered, 15 cycles, from cypher 1 to cypher 2: a total of 31 cycles from data to cypher 2. This is done by implementing 31 registers (key3 1 ... key3 31) in the Key Bank. In the 32nd cycle, Key3 enters the encryption key scheduler just as the processed data enters DES3 e. A 64-bit encrypted string is output in the 48th cycle.

3. TDES Design Evaluation

We make use of the EDA tools provided in the Altera's website to evaluate our design. These tools, Quartus II Web Service Pack 1 edition and the Altera University Program Simulator [8], allow the code to be built, compiled, synthesized, simulated and finally programed into the DE2 hardware.

In this work we use the Altera's Cyclone II DE2 Board EP2C35F672C6 platform. The technology in Cyclone II was released in 2005 [9]. The density, of model EP2C35F672C6, is 33,216 LEs and the technology is 90 nm. It contains an internal 50 MHz clock [10]. This development board is available in Terasic's website [11].

3.1. Performance

The performance results are retrieved from Altera's U.P. Simulator. The simulations were performed using the 50 MHz internal clock. The throughput calculations are based on this internal clock signal. In **Table 2** we compare the propagation times and throughputs of the non-pipelined and pipelined designs.

The non-pipelined design reflects a high propagation delay. TDES's propagation delay is 245 ns. Clocking an input string every 260 ns should process the string free of violations.

$$\text{Non-Pipelined Throughput} = 64 \text{ bits}/(260 \text{ ns}) = 237 \text{ Mbps}$$

Table 2. Non-Pipelined VS Pipelined Performance Comparison.

	TDES Encrypt/Decrypt (Non-Pipelined)	TDES Pipelined Encrypt/Decrypt
Propagation Delay	245 ns	960 ns*
Clock Period	260 ns	20 ns
Throughput	237 Mbps	≈3.2 Gbps

*There's an 8 ns delay after the clock event. The 960 ns propagation is the initial delay.

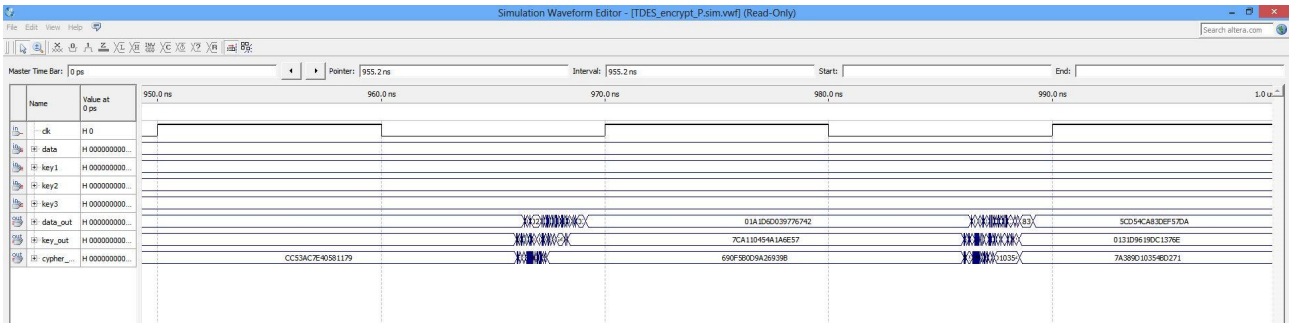

Figure 5. TDES Pipelined Timing Diagram.

Our TDES pipelined design has an initial 48-cycle propagation delay: 960 ns. However, passed the initial propagation delay, TDES outputs a processed string of 64 bits every 20 ns. The throughput achieved is approximately 64 bits × 20 ns = 3.2 Gbps. See **Figure 5**.

3.2. Performance Comparison

As mentioned earlier, a common TDES pipelined design is presented in [4]. In this sub section, we compare the performance of our design against this common design and other designs presented in [12] [13] [14] [15] & [16]. We use the 50 MHz clock (20 ns period) to normalize the calculations for all designs.

Each DES component in the designs, mentioned in the literature above, achieved an increase in performance by implementing a 16-stage pipeline. Common ways to implement TDES are by either feeding 3 keys to 1 DES component, or by inserting 3 keys to 3 DES components. When 3 keys are processed via 1 DES component, a 64-bit string output is processed every 48 cycles. When 3 keys are processed via 3 DES components, a 64-bit string output is processed every 16 cycles.

Using a 50 MHz clock, when TDES outputs a processed string of bits every 48 cycles, the performance achieved is 66.67 Mbps.

$$\text{Throughput (1 DES component)} = 64 \text{ bits}/(20 \text{ ns} \times 48) = 66.67 \text{ Mbps}$$

Using a 50 MHz clock, when TDES outputs a processed string of bits every 16 cycles, the performance achieved is 200 Mbps.

$$\text{Throughput (3 DES components)} = 64 \text{ bits}/(20 \text{ ns} \times 16) = 200 \text{ Mbps}$$

In [17] the author's achieved performance is 860.66 Mbps with a maximum clock frequency of 215.165 MHz using Xilinx Virtex4 series technology.

The throughput yield of the design presented in this work is as follows:

$$\text{Throughput} = 64 \text{ bits}/20\text{ns} = 3.2 \text{ Gbps}$$

The performance of our TDES pipelined design is 48 and 16 times greater than the common TDES implementations, 3.6 times greater than the performance shown in [16] and 13.5 times greater than our TDES Non-Pipelined design.

3.3. Cost

The parameter of interest for discussion, from the Quartus II software, is the number of Total Logic Elements (LEs). The Analysis and Synthesis results from Quartus II yield the values seen in Table 3. The total hardware space available in the Cyclone II EP2C35F672C6 platform is 33,216 LEs.

The table contains the number of logic elements for the non-pipelined and pipelined TDES designs. Our non-pipelined TDES implementation requires 12,285 LEs while our TDES pipelined design requires 13,915 LEs. The increase in the cost is due to the additional registers we added in the key schedulers, the Feistel Function rounds, and the Key Bank.

4. Conclusions

In this paper, a design to increase the performance of TDES ECB mode in VHDL using Alteras Cyclone II technology was evaluated. With a clock speed of 50 MHz, the throughput achieved is 3.2 Gbps for our TDES design. The cost of implementing our TDES pipelined design is 13,915 LEs. We achieved this by making three modifications to the TDES scheme. Piplining each DES component and Key Schedulers was the first modification. The second modification involved implementing right rotations to the decryption key scheduler. This helps maintain coherency between the sub keys and the data as it traverses the Feistel Function rounds. The third modification was the Key Bank that buffers the keys for 15 and 31 cycles.

We observe that to increase the performance, more stages must be implemented. However, more stages yield a higher cost. A higher clock speed also yields a higher throughput and does not affect the cost. However, as the number of logic elements, a string of bits must traverse, increases, the propagation delay

Table 3. Non-Pipelined VS Pipelined Cost Comparison.

		TDES Encrypt./Decrypt (Non-Pipelined)	TDES Pipelined Encrypt./Decrypt.	Total Number Of Items Available
ALTERA DE2 BOARD (EP2C35F672C6) HARDWARE COST	Total Logic Elements	12,285	13,915	33,216

increases, and the clock frequency required for proper operation decreases. Pipelining increases the throughput by decreasing the output time of a processed

string.

Acknowledgements

The support for this research is provided in part by the US National Science Foundation under Grant No. 0421585 and Houston Endowment Chair in Science, Math and Technology.

References

[1] Barker (2012) Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. NIST Special Publication, National Institute of Standards and Technology, Gaithersburg, 800-867. https://doi.org/10.6028/NIST.SP.800-67r1

[2] NIST (1999) Data Encryption Standard (DES). National Institute of Standards and Technology, Gaithersburg. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[3] Pcisecurity Standards (2017) Official PCI Security Standards Council Site—Verify PCI Compliance, Download Data Security and Credit Card Security Standards. https://www.pcisecuritystandards.org/documents/PCI_DSS_Glossary_v3-2.pdf?agreement=true&time=1496432377875

[4] Security Standards Council (2016) PCI DSS and PA-DSS Glossary of Terms, Abbreviations, and Acronyms v3.2. https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2.pdf

[5] Support Office (2017) Digital Signing and Encryption Settings—Outlook for Mac. https://support.office.com/en-us/article/Digital-signing-and-encryption-settings-8a6eb21d-0beb-4e66-a63a-2d362966cf77

[6] Keller, S. (2000) Modes of Operation Validation System for the Triple Data Encryption Algorithm (TMOVS): Requirements and Procedures. National Institute of Standards and Technology, Gaithersburg.

[7] Yao, J. and Kang, H. (2011) FPGA Implementation of Dynamic Key Management for DES Encryption Algorithm. 2011 *International Conference on Electronic and Mechanical Engineering and Information Technology* (*EMEIT*), Harbin, 12-14 August 2011, 4795-4798. https://doi.org/10.1109/EMEIT.2011.6024111

[8] Altera (2016) Quartus Prime Design Software. https://dl.altera.com/13.0sp1/?edition=web

[9] John, L. (2016) Altera Parts History. University Of California, Berkeley. http://www-inst.eecs.berkeley.edu/~cs294-59/fa10/resources/Altera-history/Altera-history.html

[10] Altera (2007) Cyclone II Device Handbook, Volume 1. https://www.altera.com/en_US/pdfs/literature/hb/cyc2/cyc2_cii51002.pdf

[11] Terasic, N.P. (2016) Altera DE2 Board. https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=53&No=30

[12] Fu, L. and Pan, M. (2009) A Simplified FPGA Implementation Based on an Improved DES Algorithm. 3*rd International Conference on Genetic and Evolutionary Computing*, Guilin, 227-230. https://doi.org/10.1109/WGEC.2009.11

[13] Arich, T. and Eleuldj, M. (2002) Hardware Implementations of the Data Encryption Standard. *The* 14*th International Conference on Microelectronics*, 100-103.

[14] Wang, K. (2009) An Encrypt and Decrypt Algorithm Implementation on FPGAs.

5*th International Conference on Semantics, Knowledge and Grid*, Zhuhai, 298-301. https://doi.org/10.1109/SKG.2009.74

[15] Taherkhani, S., Ever, E. and Gemikonakli, O. (2010) Implementation of Non-Pipelined and Pipelined Data Encryption Standard (DES) Using Xilinx Virtex-6 FPGA Technology. IEEE 10*th International Conference on Computer and Information Technology*, Bradford, 1257-1262.

[16] Yao, J. and Kang, H. (2011) FPGA Implementation of Dynamic Key Management for DES Encryption Algorithm. *International Conference on Electronic and Mechanical Engineering and Information Technology*, Harbin, 4795-4798. https://doi.org/10.1109/EMEIT.2011.6024111

[17] Ren, F., Chen, L. and Zhang, T. (2011) 3 DES Implementation Based on FPGA. In: Zhiguo, G., Luo, X., Chen, J., Wang, F.L. and Lei, J. Eds., *Emerging Research in Web Information Systems and Mining*, Springer, Berlin Heidelberg, Vol. 238, 218224. https://doi.org/10.1007/978-3-642-24273-1_29