Volume 4, Issue 9, September 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Data Encryption Standard (DES) Using Xilinx Virtex-6 FPGA Technology

Janreddy
M.TECH (VLSID)
Ashoka College of Eng & Tech
Nalgonda, India

Srinivas. T
Proffesor Hod Dept (ECE)
Ashoka College of Eng & Tech
Nalgonda, India

Abstract*—*Data encryption process can easily be quite complicated and usually requires significant computation time and power despite significant simplifications. This paper discusses about pipelined and non-pipelined implementation of one of the most commonly used symmetric encryption algorithm, Data Encryption Standard (DES). The platform used for this matter is, Xilinx new high performance silicon foundation, Virtex-6 Field Programmable Gate Array technology. Finite state machine is used only in non-pipelined implementation, and it is not implemented for the pipelined approach. The testing of the implemented design shows that it is possible to generate data in 16 clock cycles when non-pipelined approach is employed. When pipelined approach is employed on the other hand, 17 clock signals are required for the initial phase only, and one clock signal is sufficient afterwards for each data generation cycle. The Very High Speed Integrated Circuit Hardware Description Language (VHDL) is used to program the design.

Keywords*—*Data Encryption Standard, Field Programmable Gate Arrays, Very High Speed Integrated Circuit Hardware Description Language, Finite State Machine.

I. INTRODUCTION

The most commonly used technique for producing confidentiality in data transmission is symmetric encryption [3], [11]. Symmetric encryption scheme, also referred to as single key encryption method, has five main modules [11]. The term plaintext is used to denote the original incoming data. The key is an essential part of the encryption process and it provides the secure data traffic among the sender and the recipient. Encryption algorithm performs various mathematical and logical functions on the plaintext by using the key. Cipher data is the encrypted message produced by encryption algorithm by using the key and the plain text.

Decryption algorithm is employed on cipher data and performs reverse action to generate the plain text. The symmetric key encryption algorithm shares the same key between sender and receiver and a strong algorithm for encryption and decryption processes is essential to provide an efficient security mechanism [11]. The block cipher algorithms are commonly used symmetric encryption techniques which analyse the input data as fixed size blocks and produce the scrambled data as equal size as the original data [11]. One of the most widely used

Various approaches are presented in above mentioned studies to improve the performance of DES algorithm. Some approaches improve the theoretical part of DES such as [3] and others improve the DES based on the different implementation approaches such as [1], [10]. The last well- known, fastest and fully pipelined implementation of DES was announced by Xilinx Company in [10] where the modification was on the implementation part and not on the theoretical algorithm of DES. The data rate in the fore mentioned design is 15.1 Gbps using VirtexII FPGA platform [10]. In this study we use the first definition of DES in [2] and in addition to comparison of non-pipelined and pipelined approaches, we propose a pipelined design with data rate up to 18.82 Gbps by using Virtex 6 FPGA technology

The paper is organised as follows: The next section presents the DES algorithm under study. The section on Xilinx Virtex- 6 FPGA provides the details of the hardware platform employed. The sections on non-pipelined and pipelined implementation give the details of two different approaches respectively. Results and relevant conclusions are drawn in the final section.

II. DES ALGORITHM

DES algorithm uses complicated logical functions such as various types of permutations, XOR and SHIFT functions. Since the key employed is transformed to mentioned function, by following the algorithm provided, the only way to decrypt the plaintext is to apply the same key in decryption algorithm as well. DES takes 64 bits plaintext and 56 bits key as input and generates 64 bits cipher data as output [2]. The block diagram of algorithm is shown in Fig. 1. Sometimes the key is considered as 64 bits where 8 bits is used for parity check. The DES structure is first described by Horst Feistel in 1973 [2], as shown in Fig. 2.

In this method, after initial permutation (IP) of the plaintext, it is divided into two halves L(0), R(0). The two halves pass through 16 rounds. Then after the final permutation (FP), the cipher data is produced. IP and FP work exactly in opposite ways to each other [2].

Each Round i has three inputs: L (i-1), R (i-1) and K (i) where K (i) is generated from Key Scheduler program which is described in the following section. All rounds have the same structure. In Round i, L (i) is equal to R (i-1). But R (i) is derived from [L (i-1) XOR F(R (i-1), K (i))a]. Where XOR is an exclusive-OR operator and F is a round function. The function diagram of F is shown in Fig. 3.

Logic Cell	241,152
Slices	37,680
Max Distributed RAM(Kb)	3,650
User I/O	600
Size (mm)	35 x 35

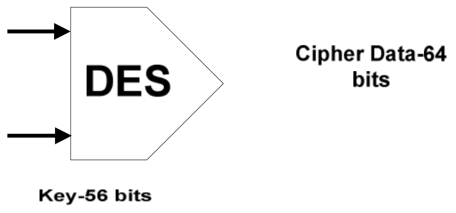

Fig. 1. DES Block View

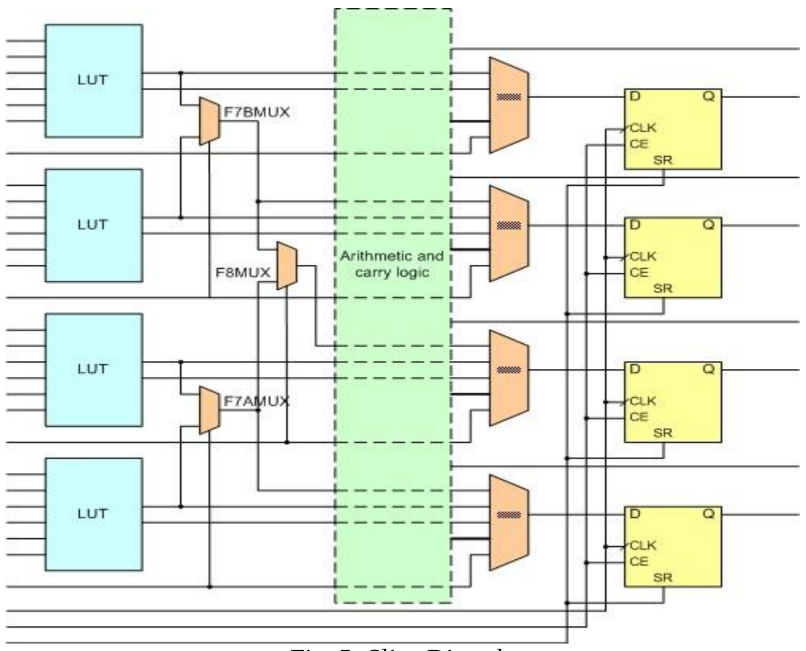

Fig. 5. Slice Diagrthe

In the non-pipelined mode implementation, the cipher code will be generated in 16 clocks. The decryption process will generate the original data in 16 clocks as well. The simulation waveform created by Modelsim [6] is shown in Fig. 7. The waveform depicts the following information:
Encryption Process, input data (64 bits): x”AAAAAAAAAAAAAAAA” key (64 bits)
x”BBBBBBBBBBBBBBBB”
ciphered Data (64 bits) x”AC972FC04E884797” Decryption Process, input data (64 bits):
x”AC972FC04E884797”key (64 bits) x”BBBBBBBBBBBBBBBB” deciphered Data (64 bits)
x”AAAAAAAAAAAAAAAA”.

III. PIPELINED IMPLEMENTATION

Pipeline is an important technique to increase the performance of a system. The basic idea is to overlap the processing of several tasks so that more tasks can be completed in the same amount of time. If a combinational digital circuit can be divided into stages, we can insert buffers (registers) at proper places and convert the circuit into a pipelined design [8]. Two criteria are used to examine the performance of a system [8]:

- Delay: is the time required to complete one task.
- Throughput: is the number of tasks that can be completed per unit time.Adding pipeline into a combinational design can only increase a system’s throughput. Such an approach does not reduce the delay

in an individual task. Actually, because of stage division, the delay will be worse than that of the non-pipelined design [8].

Various implementations are presented in various platforms for DES algorithm in [1], [3], [4], [9], [10]. However the design provided in this study is implemented in Virtex6 FPGA. In the current pipelined design, we put buffers in input and output stages of each round. The key scheduler part does quite a novel task in current implementation. It floods the sub keys to appropriate round. The key scheduler implementation specifications are as follow:

- In output part of sub key 1 K(1), there is one register which means in every one clock signal, K(1) will be transferred to core function (f) of program.
- In output part of sub key 2 K(2), there are two registers which means in every two clock signal, K(2) will be transferred to core function (f) of the program. This sequence will continue until 16th sub key K(16) which will be generated and transferred to control part of the program in every 16 clock signal. Fig. 9 shows the diagram of generating the sub keys and core function.
- As a result, for a specific plaintext and key, the appropriate K(1) to K(16) will be generated exactly at the needed clock cycle. For proper result, key scheduler must be synchronized properly to core function otherwise the result will be incorrect. Fig. 8 illustrates the simulation wave form generated by ISim [14], for the following test bench (encryption):

Wait for 10 ns; DATA<=X"0000000000000000"; Key<= X"0000000000000000"; Wait for 10 ns; Key<=X"3b3898371520f75e"; Wait for 10 ns; DATA<=X"1111111111111111";
The results clearly show that, in pipelined implementation, the ciphered or deciphered data will be generated in one clock signal.

KEY-[1:64] Data-[1:64]
Database file (NGC) which describes logical design reduced to Xilinx primitives. The MAP process fits the design to available resources in the targeted device and optionally place the design. The results are output to a Native Circuit Description file (NCD) which is used in PAR section. PAR takes the NCD file, places and routes the design based on timing constraint file. The results are forwarded to another NCD file which will be used by following section.

Generate programming file, generates bit stream file based on previously created NCD file which can be downloaded to the correspond device.

IV. IMPLEMENTATION RESULTS

The timing report of both designs is briefly shown in Table II. It is obvious that there is a trade-off between clock or delay and throughput. While we increase the throughput, we face with some delays in the design. In the current design, while we increase the throughput from 4.8 Gbps to 18.82 Gbps, we lose clock frequency from 1201.923 MHz to 294.031 MHz. However, the throughput of current pipelined design is more than the Xilinx one [10] .

Table II Design results

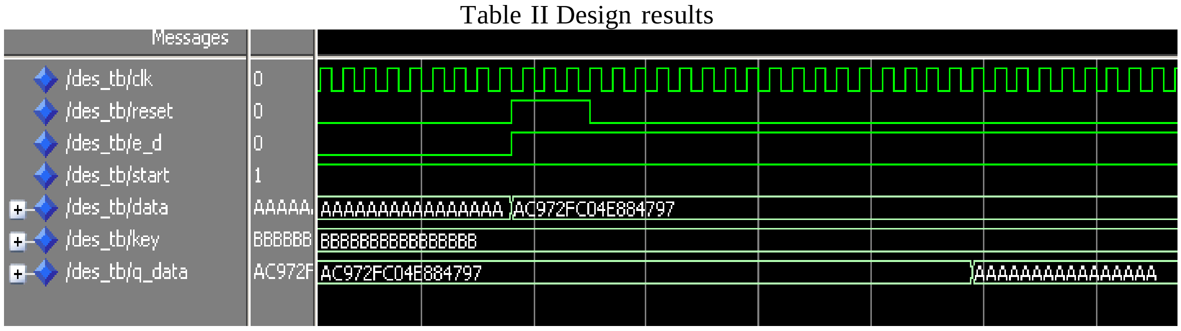

Fig. 7. Waveform of DES simulation (Non-pipelined Mode)

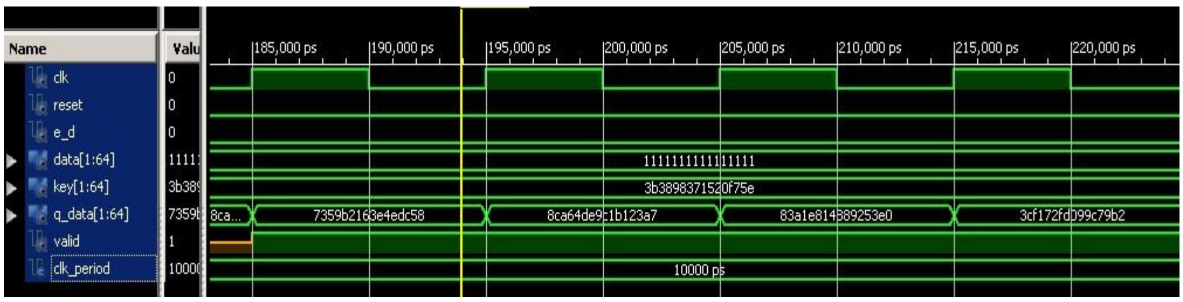

Fig. 8. Waveform of DES simulation (Pipelined Mode)

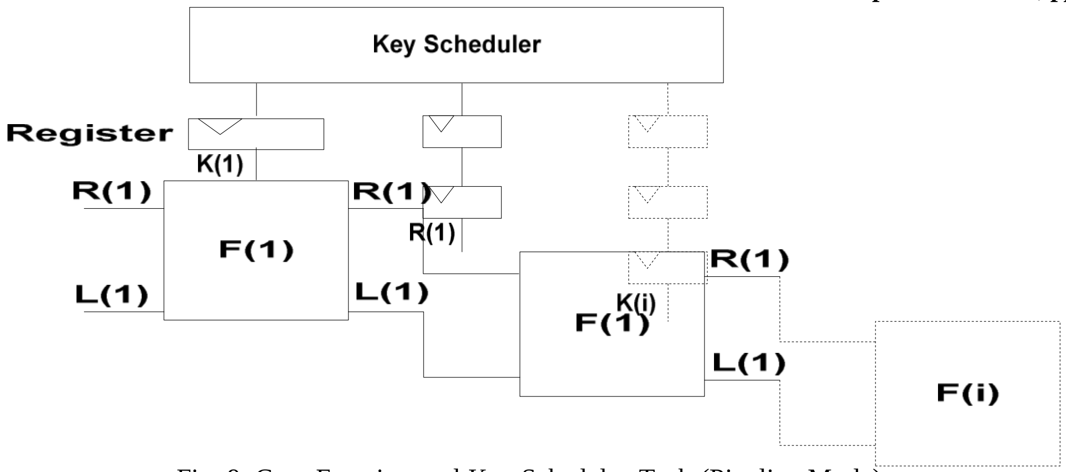

Fig. 9. Core Function and Key Scheduler Task (Pipeline Mode)

V. CONCLUSION

Non-pipelined and fully pipelined DES algorithm implementations are presented in this paper. The results show that is possible to implement a design by using FSM in order to operate at high system clock frequency (1.2 GHz).

Similar to the studies presented in [10], [1], [12], [7], the original implementation of the DES algorithm is considered and the theoretical part is not modified. The implementation presented by using Virtex-6 family FPGAs have better performance than the existing ones since it is possible to have throughput going up to18.82 Gbps by implementing a fully pipelined design including pipelined key schedulers. It is desirable to implement the improved DES algorithm presented in [3] by using a similar approach in order to test the performance improvements for the future studies.

REFERENCES

[1] C. Patterson, “High performance DES encryption in Virtex FPGA’s using JBits,” Field-Programmable Custom Computing Machines, 2000 IEEE Symposium on, pp.113-121.
[2] Data Encryption Standard, FIPS PUB 46, 1977 Jan 15, available from NTIS; Springfield, VA 22151 USA.
[3] Fu Li, Pan Ming, “A simplified FPGA implementation based on an Improved DES algorithm,” IEEE Gene and Evolutionary Computing, 2009. WGEC '09. 3rd International Conference on, pp.227-230.
[4] Ke Wang, “An encrypt and decrypt algorithm implementation on FPGA’s,” IEEE Semantics, Knowledge and Grid, 2009. SKG 2009. Fifth International Conference on, p
[5] L. Floyd, “Digital Fundamental with VHDL,” pp.362-368, ISBN: 0-13- 099527-4, Pearson Education, 2003
[6] Mentor Graphics, “Modelsim Data sheet,” 2008,
[7] V. Pasham, S. Trimberger, “High-Speed DES and Triple DES Encryptor/Decryptor,” Aug. 2001, Available:http://www.xilinx.com/support/documentation/application_no tes/xapp270.pdf.
[8] W. Stallings, L. Brown “Computer Security Principle and Practice pp.593-600, ISBN: 978-0-13-600424-0,
[9] Wong, K., Wark, M., Dawson, E.: A Single-Chip FPGA Implementation of the Data Encryption Standard (des) Algorithm. In: IEEE Globecom Communication Conf., Sydney, Australia (1998) 827–832
[10] Xilinx, “ISE In-Depth Tutorial,” pp.95-120, Jun.2009, Available: http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/ise_11tut.pdf.
[11] Xilinx, “ISE Simulator,” [Online] Available http://www.xilinx.com/tools/isim.htm .