Military Technical College
Kobry El-kobbah,
Cairo, Egypt

5th International Conference
on Electrical Engineering
ICEENG 2006

Efficient Uses of FPGAS for Hardware Implementation of Data Encryption Standard

Aly E. Salama* , Fawzy H. Aly** , M. Nabil.**

Abstract

In this work, a proposed pipeline implementation of Data Encryption Standard DES algorithm on field programmable gate arrays (FPGAs) is introduced with multiple design versions. All these versions are described in Electronic Code Book mode (ECB) using the hardware description language VHDL (Very high speed integrated circuit hardware description language). These versions have differences in the architecture and the techniques that substitution boxes (S_BOXes) can be implemented. All these design were implemented on devices from XILINX and we achieved speeds of up to **4.23 Gbits/s**.

Besides, a comparative study is conducted between the proposed designs for DES algorithm, another design for DES (Full Rolling) and the other previous implementations based on many aspects as architecture, cost and performance.

Keywords : Cryptography, data encryption standard, electronic code book, field programmable gate array, substitution boxes and very high speed integrated circuit hardware description language, symmetric block cipher.

* Professor, Faculty Of Engineering, Cairo University, Giza, Egypt.

** Egyption Armed Forces.

1.Introduction

The use of modern computers makes great advances in the Cryptography algorithms, permits the algorithms to become more complicated, and gives the ability to get faster processing. In the recent decades, extensive academic research has made a great development in modern Cryptography protocols and algorithms which leads to the invention of **Data Encryption Standard (DES**) by a group of IBM corporation.

The implementation of cryptographic algorithms on reconfigurable hardware devices based on Field Programmable Gate Arrays (FPGA) devices are highly attractive. They can run faster than software implementations while preserving the physical security of hardware solutions. In the same time, they maintain the flexibility obtained by using those software solutions [1].

The main task that is intended form the design is to get high-speed encryptor / decryptor. Consequently, to achieve this aim, an analysis study is provided for DES algorithm and multiple design versions of the DES are proposed. We analyze the effect of using the Pipelining technique with the DES algorithm. A design is introduced with Full Rolling DES (*i.e.* without pipelining) and another design with the Pipelining technique, after that, a comparative study is held on the resulting performance among the two methods and the previous designs for DES [1]. Moreover, Three techniques are described to implement the S_BOXs, the 1st technique describes the S_BOXs using logic equations [6], the 2nd and the 3rd using the dedicated ROM and the BLOCK RAM elements respectively of the targeted chip.

In the DES algorithm, the S_BOXs are the core of the DES algorithm, so good and firm design of those S_BOXs is the guaranteed technique to achieve a good design performance.

2. DES Algorithm

2.1 Basic DES

The Data Encryption Standard known as DES is one of the most well known algorithms in the past two decades. It has been used worldwide standard for twenty years is used for the cryptographic protection in versatile fields. For instance, sensitive information, computer data, and telecommunication media. It also has been held up remarkably well against cryptanalysis for many years and is still secure compared with most powerful cryptography algorithms.

DES is Symmetric key-Block cipher. That means the same algorithm is used for both encryption and decryption of the data. At which a block of 64-bit of data is encrypted using a 56-bit key. The resultant of this operation is a 64-bit chunk of encrypted data. Some documentation indicates that **DES** uses a 64-bit key ,eight bits of the 64-bit key value are simply parity bits that the encryption algorithm never uses.

After an initial permutation the block is broken into a right half and a left half. Each is 32 bits long. Then there are 16 rounds of identical operations called function f in which the data are combined with the key in a non-linear fashion[2,3,4,5,6]. After the sixteenth round, the right and left halves are joined, a final permutation (the inverse of the initial permutation) then the algorithm terminated, and the output cipher comes out. In each round, the key bits are shifted and then 48 bits are selected from the 56 bits of the key to enter the corresponding round.

As shown in Fig.1, this operation is repeated 16 times making 16 rounds of the **DES**.

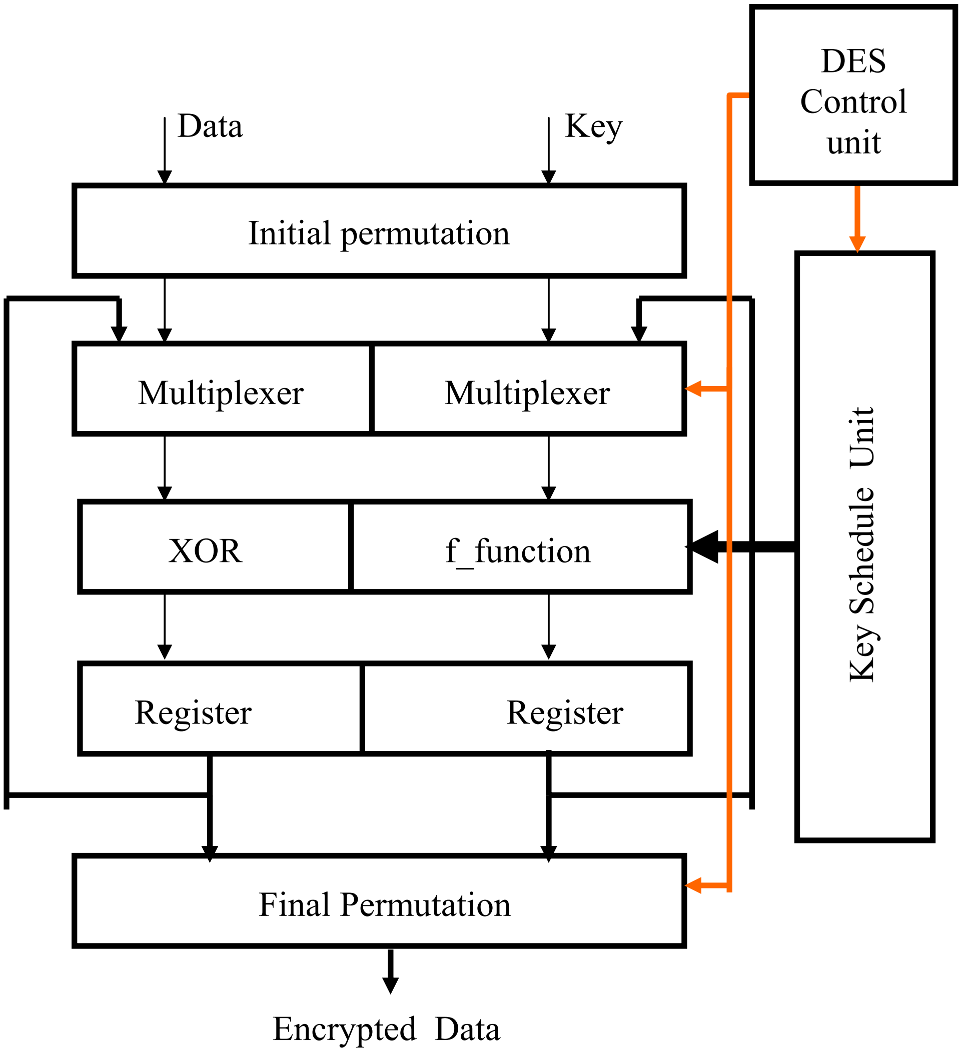

Fig 1.Basic DES Algorithm

2.2 Pipelining

In order to gain speed and accelerate the DES algorithm, pipelining technique is used. Here, 16 copies of the single round is built together to loop the data through 16 stage .Data is processed one block of data at a time, i.e. With every rising edge clock, 64 bit of data is introduced to first stage of the sixteen stage used. For pipelining, this increases the encryption (decryption) of data rate by factor of sixteen but at the cost of approximately sixteen times logic resources needed to achieve this target [1].

As shown in Fig. 2. The first block of data is loaded with the start of encryption (decryption) process, and then it goes through the initial permutation. The output of this stage is then loaded to the first round with associated sub-key. The result is then loaded into two registers of 32-bit length. The two registers are located between every round to hold the result and load them to the next round with the next rising edge clock. Again, the second round is loaded with the stored data from the previous round along with its associated sub-key of the second round. Then the entered data are held in the second register pair waiting for the next rising edge clock.

In the same time, 2nd 64 bit block of data are loaded to the initial permutation stage and continue the way as the previous block of data and so on, until the pipeline stages will be full at the sixteen clock cycle.

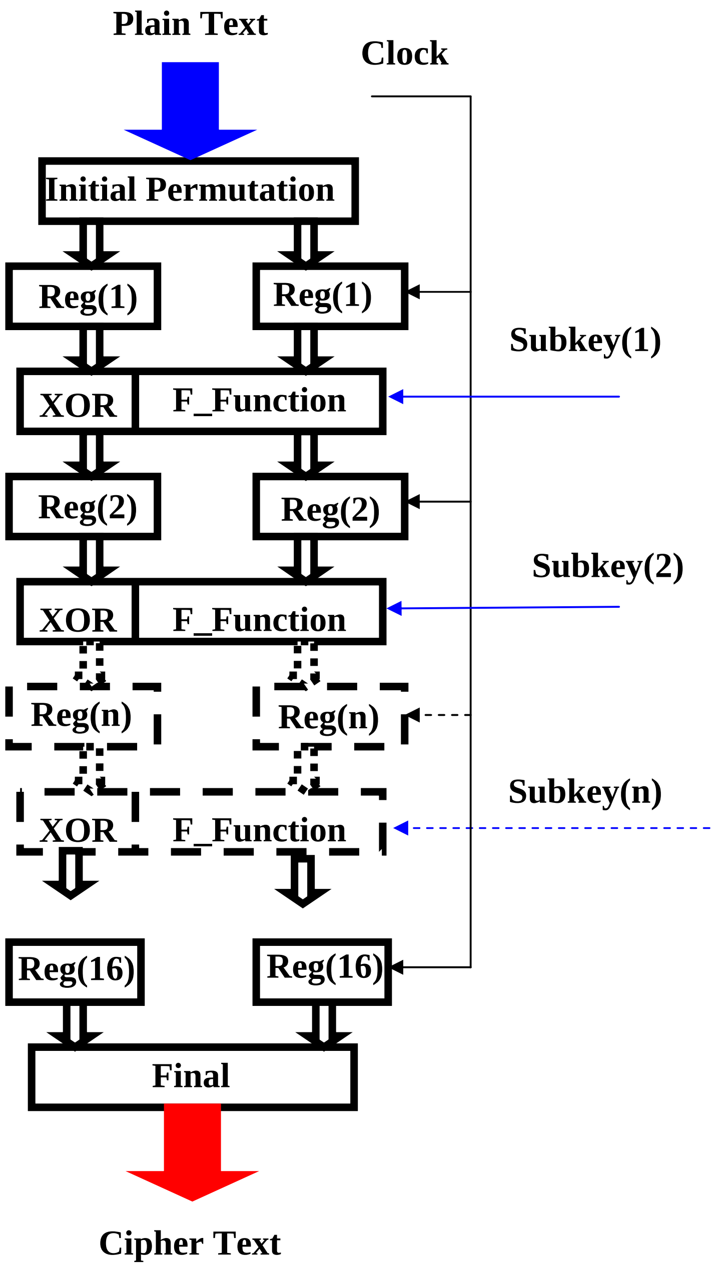

Fig. 2. the Pipeline Architecture.

The first block of data continues the trip through the sixteen rounds. Until the first encrypted block of data is appeared at the output of the final permutation stage.

From this moment, with each rising edge clock cycle, 64-bit block of data entered the pipeline, another iteration of data of this block and sub-key are computed, and then a block of encrypted data appears at the output. The same algorithm is used for decryption process, but in the reverse order for the sub-keys. The data, which entered first, will also leave first. Therefore, the performance for the algorithm increased significantly by the factor of sixteen.

2.3. Full Rolling DES

As shown in fig.1. the first block of data is initially permuted and then enters to the main round, which is XORed with its corresponding key from the key schedule module. The output data is then enrolled 16 times through the Feistel network.

In the same time, the corresponding 16 sub keys are generated in synchronization with the corresponding data enrolled from the previous round. The data from the pervious session and the associated sub key are entered to the F-function and the output is then XORed with the output form the multiplexer (MUX), which is used to loop the output back to the input of the Feistel network. Therefore, two multiplexers are used to switch data at the beginning of the encryption and decryption processes from the initial permutation process to the Feistel network. Then, in the second round of the 16th round from the output of the previous round to the input to the Feistel network.

Two registers is used at the output of the Feistel network to store the results of each loop and direct them to the two MUXs. Besides, the two registers control the data to be synchronized at the input of the final permutation stage.

The first block of data will be presented at the output of the network, after 16th round, i.e. each 64-bit block of encrypted or decrypted data will be ready at the output every 17th clock cycle. (One additional clock cycle for the start signal).

Worthy mentioning, pipelined architecture of DES can be realized only by ECB mode of the operations of the DES, for the reason that pipelined architecture can be fully exploited only in modes of operation which do not require feedback of the encrypted data. Meanwhile the Full Rolling one can be realized by the four modes of operation for DES.

For DES, there are four official modes of operation specified in Federal Information Processing Standards Publication (FIBS-74, 81) [4,7]. Consequently, cryptographic system designers must select one or more of the possible modes of operation for implementing the DES in a cryptographic system or security application.

The modes included in this standard are the Electronic Codebook (ECB) mode, the Cipher Block Chaining (CBC) mode, the Cipher Feedback (CFB) mode, and the Output Feedback (OFB) mode [6].

3. Design Methodology

Design Cycle

The design flow explains the procedures for the design cycle for the proposed DES algorithm. The design methodology for DES implementation started with software verification of DES. This step presents an important phase in realization of DES, wherever it supports the design validation during the hardware implementation phase. Then, in the hardware implementation phase the design flow started with dividing the algorithm into modules and sub modules to ease the verification and validation of each component, and then, VHDL code is written for describing of each component of the DES. Following, a functional simulation is performed for the written VHDL code to validate the correctness of operation of internal DES component.

Next, the process of synthesis and logic optimization is executed, followed by validation on the register transfer level. Then, the place and route process is performed for the design on the targeted technology. Next of this step, a timing simulation is executed. Finally we back annotated verification for the design.

In the case when the results could not be met, the design process loops back again to modify the VHDL code or modify the time constraints in the simulation test bench until the required results is obtained.

4. DES Design

DES has been broken down into small elementary computational units:

Permutations boxes and expansion boxes

They are just swapping, so no logic resources will be used to implement them. They consume only some routing resources anmd they cause no additional delays except some wiring delays. so there is no optimization at this stage is needed for multiple design versions (Pipeline And Full Rolling).

Registers

We decided to implement the registers by inferring them in VHDL language for multiple design versions (Pipeline And Full Rolling).

S_BOXes

The S_BOXes are crucial for the realization of a good design performance for the DES algorithm. The implementation of them is carried out using three different methods as mentioned before in the preceding chapter. The obtained results as following:

1- For **M.Kwan** equations method [8]

2- The second method, which implements S_BOXes on the ROM elements of the SPATAN II. Implementing the S_BOXes on the ROM elements reduces greatly the amount of gate count on the device as in the same time increases the performance of the DES. The S_BOXes tables are written to a file with extension (romx.coe) in hex format, where the letter 'x' stands for the S_BOX number and it ranges from 1 to 8. Then, it is converted to VHDL file by the synthesis tool that will implement the design on the device later in the place and route stage [9].

3-The 3rd method, implements S_BOXes on the Block RAM elements of the SPATAN II..

Converter

An important issue must be stated, that is the input output problem. DES algorithm has input 64 bits plain text, 64 bits input key and 64 bits output cipher text. Besides, the control pins that controls the DES algorithm, which they are the clock, start, cipher, and the reset pins. Therefore, the sum of input output pins are 196 pins. On the other hand, **SPARTAN II XC2S200 – 6PQ208** has only available 144 input output pins. For that reason, a converter is used, which acts like a serial to parallel shift register. It consists of two parts:

1- Control state.

2- Converter.

The control state is a simple finite state machine that is responsible of producing an upload signal after three clock cycles and the upload signal is produced at the 4th clock cycle. This UPLOAD signal, its function, as its name indicates, to upload the 64 bits plaintext to the initial permutation unit of the pipelining algorithm.

The converter accepts 16 bits of the plaintext and key with every rising edge clock and accumulates them until the internal accumulator has 64 bits then an internal **Upload** signal is generated and the 64 bits will get out to the main Pipelining DES algorithm. Therefore, the design of DES has two unit converters one for the plaintext and another for input key.

Consequently, the output cipher-text will be ready at the output of the Pipelining DES after 20 cycle , i.e. we have a latency of 20 clock cycle, 4 cycles for the converter operation which needs 4 clock cycle to upload the plaintext and 16 clock cycle for the main Pipelining DES. The converter is activated by a start signal and the transition in its states is accomplished by the clock signal. By this technique, the number of input output has been reduced to 100 pins.

DES Controller

The DES controller is designed as simple counter with twenty stages.

The Control Unit has the burden of doing the following actions:

1- Guarantees that both the multiplexed 32 bits plaintext and the 48 bits sub-keys are brought together into the round to be processed.

2-It has the responsibility of producing the 16 states which in turn effecting in producing the sub-keys. The Control Unit is implemented as a simple 16-bit counter, which represents the required 16 states.

Multiplexer

Multiplexer is designed using combinational logic as ordinary synchronous multiplexer with 32 bits size controlled by the clock and the start signal. Two multiplexers are used to loop the data between every successive round.

Decryption Process

Since DES is a symmetric block cipher, then the same algorithm used for the encryption is the same one used for the decryption. The only difference in hardware implementation lies in operation of shift register in the key Schedule block. Accordingly, the same logic resources used for encryption are those used for decryption. The number and the direction of shifts per round for decryption process, differ.

5. Performance Analysis

- Pipeline DES Design results for implementing the s_boxes with 3 different methods:

a-M. Kwan Equations

Table.1.Device Utilization Summery For Pipeline design using M.Kwan eq^ns.

Attributes	Used	Percentage
Number of SLICEs	2924 out of 3072	95%
Number of 4 input LUTs	5257 out of 6144	85%

b- ROM Key Implementation

Table.2.Device Utilization Summery For Pipeline design using ROM Elements.

Attributes	Used	Percentage
Number of SLICEs	2063 out of 2352	87%
Number of 4 input LUTs	4022 out of 4704	85%

C- Block Ram Implementation

Table.3. Device Utilization Summery for Pipeline design using Block RAM Elements.

Attributes	Used	Percentage
Number of SLICEs	2049 out of 2352	87%
Number of 4_input LUTs	4027 out of 4704	85%
Number of BLOCK_RAMs	3 out of 14	21%

- Full rolling DES Design results for implementing the s_boxes with 2 different methods:

a- ROM Key Implementation

Table.5. Device Utilization Report For Full Rolling design using Rom elements

Attributes	Used	Percentage
Number of SLICEs	529 out of 2352	22%
Number of External IOBs	98 out of 140	70%

b- Block Ram Implementation

Table.5. Device Utilization Report For Full Rolling design using Block RAM elements

Attributes	Used	Percentage
Number of SLICEs	504 out of 2352	21%
Number of 4 input LUTs	704 out of 4704	14%

6- Conclusions

From the above discussion, we can conclude the following :

1- The first design pipe line DES with (**Mathew Kwan** equations)has the worst design speed and area results comparing with other two design results.

2- Using the ROM or Block RAM elements of the chip resources enhance significantly the performance of the design approximately double the speed comparing with the first design.

3- The chip routing resources are important for implementing the design. This effect appears when we change the device family.

4- The obtained results for designing pipelined DES using Block RAM resources gives data throughput that reaches 4.23 GBit/sec on SPARTAN II-200K chips which is more faster then other previous implementations , which reaches data throughput 1 GBit/sec on VERTIX 300K chips using the logic equation method for implementation the S_Boxes. On the other hand, the full rolling DES gives ٣٥٦,٢٥٢Mbit / sec with minimum area resources consuming .

5-The speed (throughput) is degraded dramatically by using the full rolling method ((٤٢٣١,١٠٤/٣٨٥,٨٠) = 11 times) . Mean while the area recourses has been saved (87% for the pipeline DES & 22% for the Full rolling DES).

The following chart in fig.3.shows the results obtained after the place and round process for both Pipeline and Full Rolling DES design.

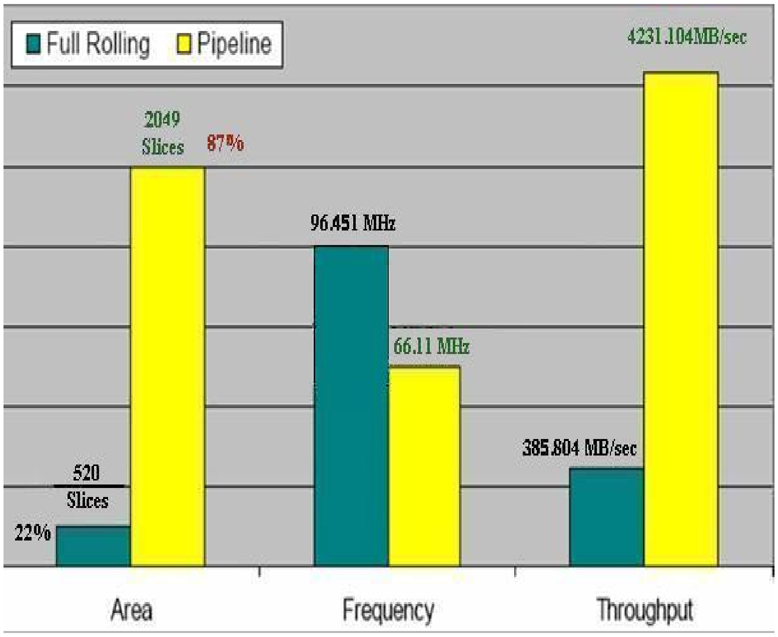

Fig.3. The Obtained Results for Pipeline and Full Rolling DES Implementation.

References:

[1]- Touria ARICH & Mohssine ELEULDJ. "Hardware Implementations of the Data Encryption Standard". December 2002 IEEE.

[2]- Bruce Schneier, "Applied Cryptography", 2nd Edition, Wiley, 1996.

[3]- A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. "Handbook of Applied Cryptography". CRC Press, 1996.

[4] FIPS PUB 81,DES Modes of Operation, December 2 1980.

http://www.itl.nist.gov/fipspubs/fip81.htm

[5] FIPS PUB 46-3, Data Encryption Standard (DES), January 15,1976. Reaffirmed 1999 October 25.

[6] NIST Special Publication 800-17, Modes of Operation Validation System (MOVES), Requirement and procedure S. Keller & M. Smid, February 1998.

[7] FIPS 74 - Guidelines for Implementing and Using the NBS Data encryption standard.

[8] M.Kwan. "Bit slices DES. May 1998, http://www.darkside.com.au/bitslice/.

[9]- Fast DES Implementations for FPGAs and its Application to a Universal Key Search Machine ". A paper introduced by Jens Peter Kaps and Christof Paar. Electrical and Computer Engineering Department Worcester Polytechnic Institute.