BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXI No. 3/2009	173 - 180	Seria Tehnică

Encryption Algorithms Design using Programmable Logic Circuits

Otilia Cangea, Cristina Popescu

Petroleum – Gas University of Ploieşti, 39 Bucureşti Blvd., Ploieşti, ROMÂNIA
e-mail: ocangea@upg-ploiesti.ro, cristinap@upg-ploiesti.ro

Abstract

The development of cryptographic methods for data transmission systems is a field of great interest, considering the importance of the need of digital data protection and security. Along with the elaboration of powerful encryption algorithms, software implemented, today one develops hardware encryption techniques, frequently preferred. In this respect, powerful security solutions with low consumptions use FPGA (field programmable gate array) structures. The paper presents the implementation of a well-known encryption algorithm, D.E.S., the simulation of the structure being obtained using the specialized Xilinx® Web PACK™ program.

Key words: *encryption algorithms, field programmable gate array, Xilinx® Web PACK™ program.*

Introduction

Information encryption solutions are *software* and *hardware*. Any encryption algorithm may be software implemented, having as disadvantages the speed, the cost and the easiness in handling and modification. The advantages are flexibility, portability, and the easiness in using and upgrading possibilities.

Even though the software encryption is increasingly dominant, the hardware encryption is preferred in major importance military or commercial applications, due to several factors. The first and most important is the *speed*. Encryption consists of a multitude of complicated operations performed upon a clear text (in the form of bit rows), that have to be computer simulated. In this respect, specialized hardware gains by a better speed.

A second argument is the *security*. An usual encryption algorithm implemented on a computer does not have a proper protection. The hardware encryption devices are embedded and the protection may be easily assured.

Currently, it is important to implement powerful security solutions with low consumptions. In this respect, the use of FPGA (Field Programmable Gate Array) is recommendatory [2] for encryption structures, considering also the fact that they are reprogramable when errors occur.

Programmable logic circuits

In the *reconfigurable systems* a set of gates is designed to implement a certain behavior. The configuration is defined by the configuration bit that describe the interconnection mode and the

behavior of the gates. The reconfigurable systems have proven to be an economic method for signal processing, emulation of different computing systems, and encryption systems.

The circuits used for implementing reconfigurable systems are the *FPGA*. The majority of FPGA structures implementations use *lookup tables* (LUT) as combinational logic elements. The most simple CLB (Configurable Logic Block) consists of a LUT, a latch, one or more multiplexors, and an output buffer, as seen in figure 1 [3].

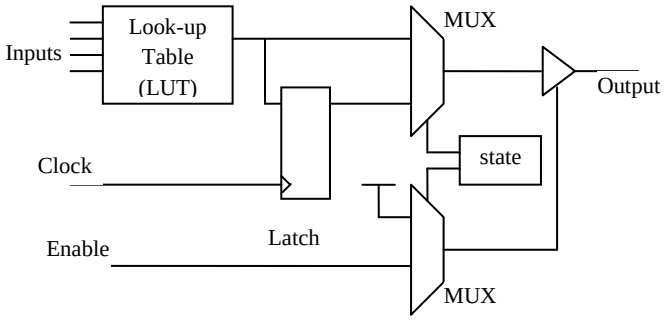

Fig. 1. Simple configurable logic block.

FPGA has been introduced in the 1990's by Xilinx Inc. and it offers the possibility of establishing interconnections between all the elements. FPGA, that are defined [3] as types of programmable logic units, may also be described as a row of cells containing configurable logic and memory elements. These cells may be interconnected using a large number of programmable switches, in a large variety of virtual structures for each digital system.

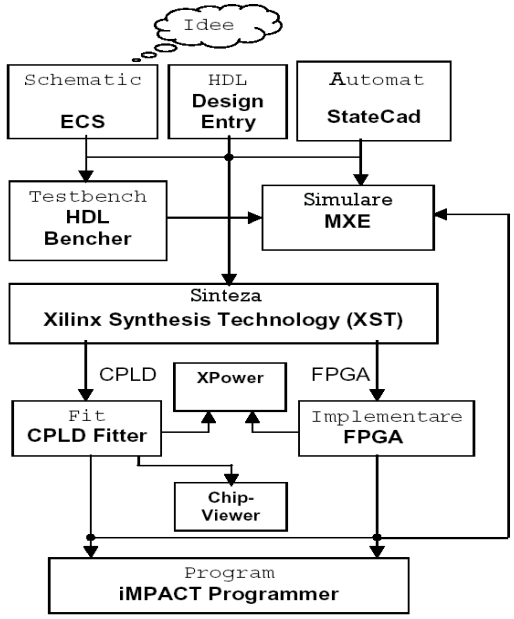

Fig. 2. Xilinx® Web PACK™ program.

In oder to achieve the implementation and the simulation, the authors used the *ISE® Xilinx Web Pack*™ program, whose structure is presented in figure 2, that is an integrated medium for digital systems design using FPGA circuits.

ISE® Xilinx WebPACK™ is [6] the ideal downloadable solution for FPGA and CPLD design, offering HDL synthesis and simulation, implementation, device fitting, and JTAG programming. It delivers a complete, front-to-back design flow providing instant access to the ISE features and functionality at no cost. Xilinx has created a solution that allows convenient productivity by providing a design solution that is always up to date with error-free downloading and single file installation.

The designer may use description languages, such as VHDL or Verilog. In order to simulate these descriptions, ISE® Xilinx Web PACK™ contains a simulator for a functional simulation of these descriptions and a graphical interface, being one of the most complex systems in the field of programmable circuits.

The implementation stages for the FPGA circuits [5] are the following:

1. *Translate*: Interpreting the structure and verifying the correctness („design rule check"-DRC);
2. *Map*: Computing and resources assignment in the selected device;
3. *Place & Route*: Placing the logic blocks in the respective logic positions and using the resources for trajectories placement;
4. *Configure*: Creation of a binary row to be discharged in the device.

Data Encryption Standard implementation

D.E.S. (Data Encryption Standard) is one of the most known examples of block ciphers, approved in 1977 in the USA by the National Bureau of Standards as the federal standard of information processing for the encryption of non-classified information.

D.E.S. is a block cipher having the length of 64 bit processed in conjunction with a key, composed of 56 bit pseudo-randomly generated and 8 bit used for transmission errors detection. The key is expanded to the length of the block and is kept by all the members of a group of users.

The fundamental construction of a D.E.S. block is [1] a combination of two elementary encryption techniques, that are substitution followed by permutation, performed upon the clear text, using a key. This construction is known as a *round*, D.E.S. being composed of 16 rounds. The algorithm is based on a set of permutations, substitutions and modulo 2 sums, 16 times iteratively applied upon a 64 bit block, using every time a different 48 bit key, extracted from a 56 bit key.

D.E.S. encryption consists of the following processing categories performed upon the block that contains the clear text, as presented in figure 3 [1]:

- an initial permutation "IP" performed upon the 64 bit block;
- a complex calculus, depending on the key, performed upon the obtained block, consisting of 16 functional identical iterations, parameterized by different keys;
- an interchange of the two half blocks, each having 32 bit;
- a final permutation (transposition), inverse of the initial one.

Processing at each "*i*" iteration consists of the following operations [4]:

- one notes *L(i-1)* and *R(i-1)* the 32 bit of each half, left and right, that compose the block submitted to the respective iteration;
- considering *k(i)* the key and a block of 48 bit selected from the 64 bit of the key, one may consider the following relations for computing the outputs *L(i)* and *R(i)* :

$$L(i) = R(i\text{-}1);$$

$$R(i) = L(i\text{-}1) \oplus f(R(i\text{-}1), K(i)). \quad (1)$$

The last iteration is different, being defined by the following relations:

$$L(16) = R(15);$$

$$R(16) = L(15) \oplus f(R(15), K(16)) \quad (2)$$

The used *f* encryption function performs a nonlinear substitution, so that an expanding function *E* is applied on the initial 32 bit block, which generates 48 bit at the output. Consequently,

E(R(i-1)) is modulo 2 summed up with the 48 bit of the *K(I)* key. The result is partitioned in eight 6 bit-blocks that represent the inputs of 8 boxes *S(j), j=1,..,8,* that perform a nonlinear substitution with 6 inputs and 4 outputs.

Considering the boxes $S_1, S_2,...,S_8$, the permutation function *P* and the expanding function *E*, in order to define the function *f(R(i-1), K(i))* one performs the 6 bit-blocks $B_1,...,B_8$

$$B_1,...,B_8 = K(i) \oplus E(R(i-1)) \tag{3}$$

In this case, the *f(R(i-1), K(i))* block may be defined as

$$f(R(i-1), K(i)) = P(S_1(B_1)\, S_2(B_2) \ldots S_8(B_8)) \tag{4}$$

After the 16 iterations calculus, the 32 bit-block is submitted to an inverse permutation IP^{-1}, an inverse of the IP.

Decryption consists in using the same algorithm, but with the keys *K(i)* applied in reversed order, from K_{16} to K_1. The first step in decryption is applying the IP permutation, that solves the last step IP^{-1} from the encryption operation. Then, one generates in reversed order :

$$R(i-1) = L(i)$$

$$L(i-1) = R(i) \oplus f(L(i), K(i)) \tag{5}$$

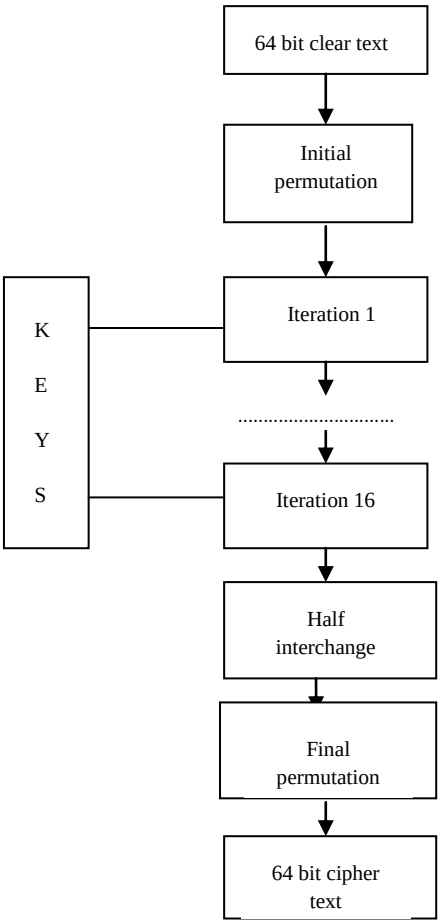

Fig. 3. General diagram of D.E.S.

Relations (5) are to be iteratively applied beginning with *R(16)* and *L(16)*, in order to finally generate *R(0)* and *L(0*). Finally, the 64 bit obtained block is submitted to an inverse permutation IP^{-1}, that leads to the message clear-text type.

As soon as one performed the structure according to the block diagram for the implementation of the D.E.S. algorithm, one obtained the simulation of this structure. Figure 4 presents the application before performing the simulation.

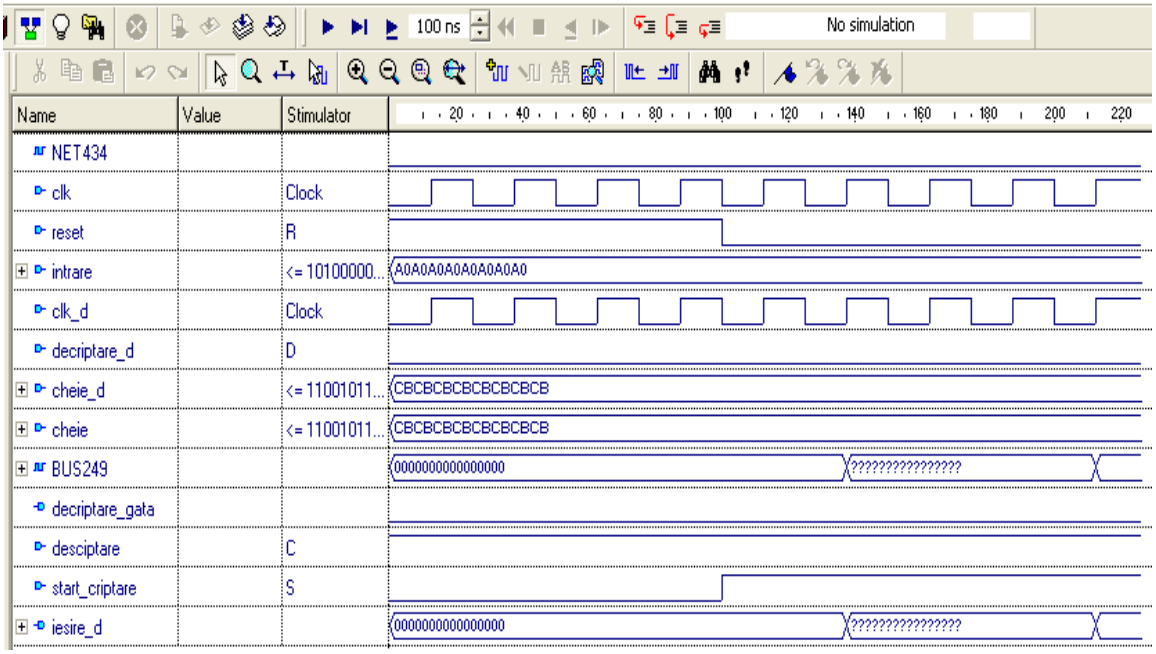

Fig. 4. Display capture before performing the cryptographic simulation.

There are several phases to be fulfilled in order to perform the simulation, among which the most important are:

- the field ”start_criptare =1” (“start encryption=1”) thus initialized, indicates the fact that the *encryption is beginning*;
- the *„Run”* command will be activated until the field *„NET 434”* will receive the value 1, meaning that the *encryption has been performed;*
- the *„Run”* command will be active until the field „decriptare_gata” (“decryption ready”) will receive the value 1, meaning that the d*ecryption has also been performed.*

The results of the simulation are presented in figure 5.

An important variant of D.E.S. is *Triple D.E.S.*, known as *Triple Data Encryption Algorithm* (T.D.E.A.) [5], widely used in configuring banking cards (from 2002, MasterCard and, partially, Visa), as well as Zapp mobile phone system. The following results are regarding *FPGA implementation and optimization on T.D.E.A. structures.*

Due to the fact that the designed encryption and decryption structures are characterized by large surfaces of FPGA circuits, one has to find methods to reduce the dimensions of these structures. An attempt to implement the encryption and decryption circuits structure on a FPGA Spartan 3 has determined an occupation degree of 120%, that imposed an optimization. There are several methods of decreasing the surface of the implemented structure. The authors used the automated and manual optimization technique of routing in FPGA structures by using *Live Design*, which relies on the same tools as Xilinx, but it is dedicated to the specific plate, that is *Live Design Evaluation Board*, presented in figure 6, in order to minimize the costs.

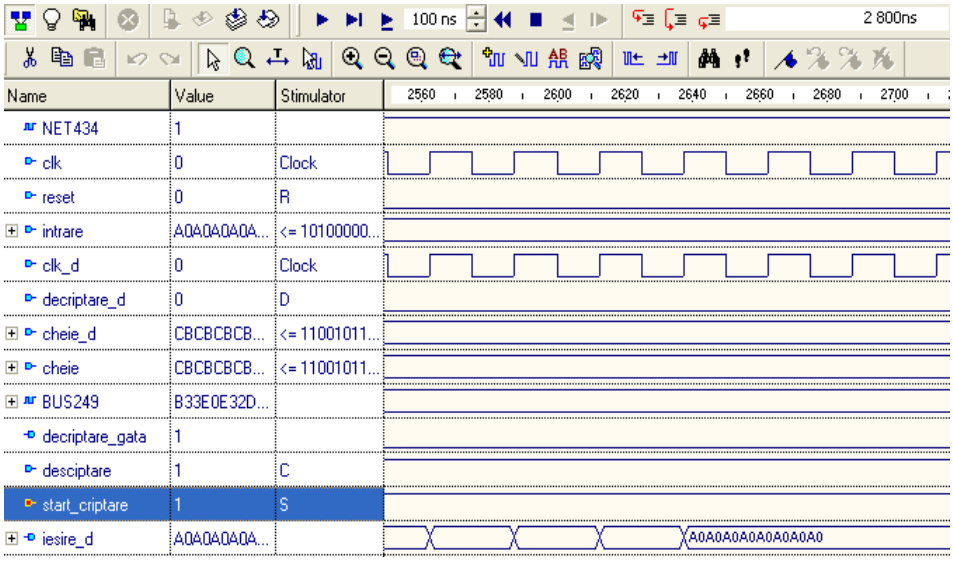

Fig. 5.Display capture after performing the cryptographic simulation.

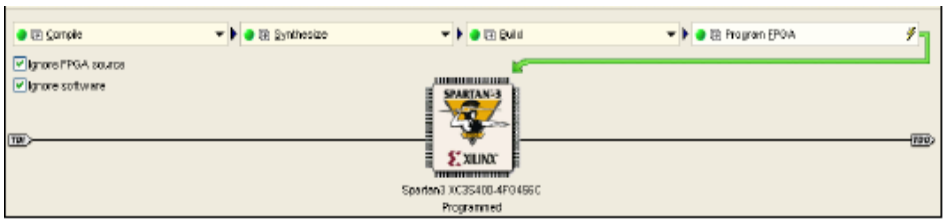

Fig. 6. Live Design Evaluation Board graphical interface.

The results of implementing the cryptographic structure in FPGA are shown in figures 7 and 8.

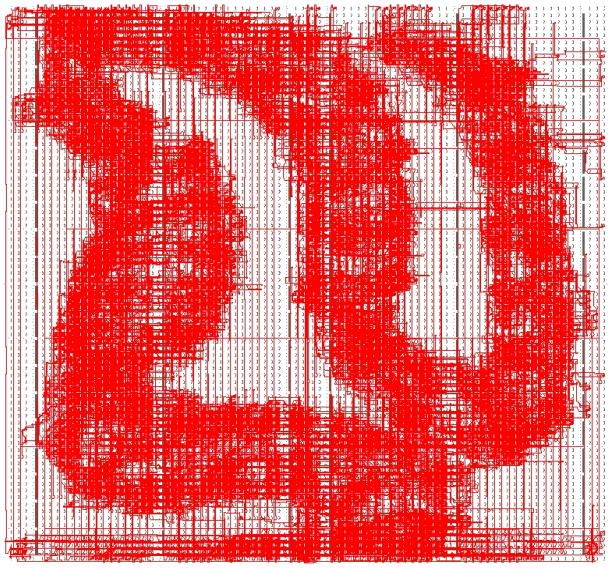

Fig.7. Implementation of the designed cryptographic structure in FPGA Virtex2.

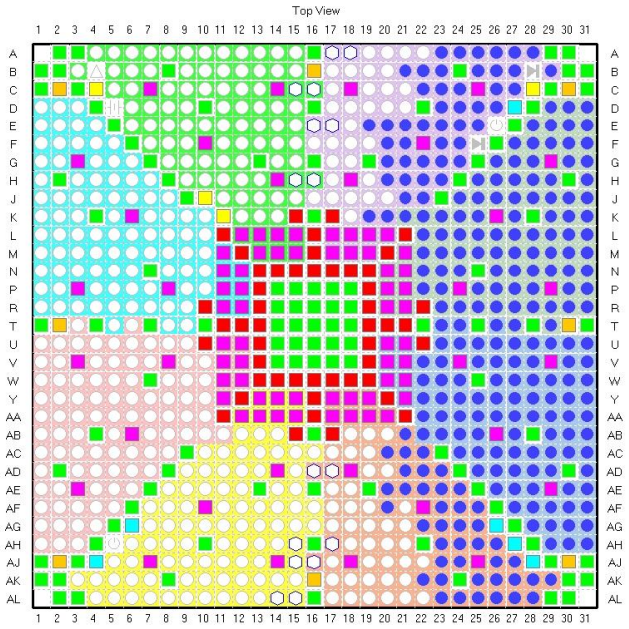

Fig. 8. FPGA Virtex 2 structure repartition for the complete diagram.

In order to design a FPGA circuit, the authors used a hardware descriptive language (HDL), one of the most used being VHDL and Verilog, based on an automatic drawing program that elaborated the cable of the FPGA circuits. The cable was then adapted to the available FPGA circuits architecture using a process named "position-establish-connections".

Conclusions

The systems provided by FPGA producers offer a high level description, using C or VHDL programming languages, or a systematic description using Verilog or VHDL .

In a first stage, the authors elaborated the D.E.S. algorithm structure, but one obtained a big size structure because the blocks were not properly organized. The optimization was achieved by means of re-use of the S-boxes and the use of FPGA memory in order to memorize the pre-computed transformations.

Consequently, it was performed the implementation of the permutations by means of interconnecting the trajectories, the information circulating 16 times through the multiplexer until the final result is obtained. The S-boxes use ROM memories simulated in FPGA records for an intermediate storage of the encryption.

Using FPGA circuits for encryption structures implementation is logic, considering the fact that these allow re-programming when errors are detected, are defined by low consumptions, and operate at high speed.

References

1. Biham, E. – *Differential Cryptanalysis of the Data Encryption Standard*. Springer Verlag Publishing House, 1997.
2. Chodowiec, P., Khuon, P., Gaj, K. – *Fast Implementations of Secret-Key Block Ciphers Using Mixed Inner- and Outer-Round Pipelining*. ACM/SIGDA Ninth International Symposium on Field Programmable Gate Arrays, Monterey, CA, February, 11-13, 2001.

3. Castano, S., Fugini, M., Martella, G., Samarati, P. – *Database Security*. Addison-Wesley, 1995.
4. Gaj, K., Chodowiec, P. – *Comparison of the hardware performance of the AES candidates using reconfigurable hardware*. Third Advanced Encryption Standard (AES) Candidate Conference, New York, 1999.
5. Chodowiec, P., Gaj, K., Bellows, P., Schott, B. – *Experimental Testing of the Gigabit IPSec-Compliant Implementations of Rijndael and Triple DES using SLAAC-1V FPGA Accelerator Board*, Proc. Information Security Conference, Malaga, Spain, October 1-3, Springer-Verlag, 2001.
6. http://www.xilinx.com/tools/webpack.htm.

Utilizarea circuitelor cu logică programabilă pentru implementarea algoritmilor de criptare

Rezumat

Dezvoltarea metodelor criptografice pentru sistemele de transmisii de date se bucură în prezent de un mare interes, luând în consideraţie importanţa necesităţii protecţiei şi securităţii datelor digitale. În plus faţă de elaborarea unor algoritmi de criptare cât mai puternici, cu implementare software, sunt dezvoltate astăzi tehnici de criptare hardware, deseori preferate. În acest sens, soluţii de securitate puternice, cu consumuri scăzute, folosesc structuri de tipul FPGA (field programmable gate array). Lucrarea prezintă implementarea unui important algoritm de criptare, D.E.S., simularea acestei structuri fiind obţinută prin utilizarea pachetului specializat de programe Xilinx Web PACK.