More security and autonomy for users: encryption system with evolutive key

E. Franti[1] & M. Dascalu[2]
[1]*IMT Bucharest, Romania*
[2]*Research Institute for Artificial Intelligence, Romania*

Abstract

This paper presents a complex project that aims towards the development and design of an encryption system with an evolutive key based on the cellular automata model. The novelty and the main interest in our project is the orientation towards the hardware implementation. Most of the research effort was concentrated on the algorithm for generating the evolutive key. The final solution is based on cellular automata and gives a special constructive simplicity, simultaneously with a very high complexity of the evolution of the keys. Special software has been developed for the analysis of different encryption/decryption strategies with cellular automata. Between the various strategies and algorithms studied, the best were selected for hardware implementation. The implementation of the encryption system was realized on the Spartan 3 XC3S400 platform produced by Xilinx. The result is a fast, cheap, efficient and versatile cryptographic system implemented on a FPGA Spartan III platform.
Keywords: cryptography, parallel computing, cellular automata, hardware/software co-design.

1 Introduction

In this age of information, communications and electronic connectivity, security is a topic of general interest that should never be underestimated. The security of databases, of data communications, of Internet connections, of scientific research and of personal e-mail and phone calls are examples for cases in which the encryption of data/information plays a major role. Therefore, cryptography has become an important field of theoretical research and applications development, not only in military communications as it was at its origins.

WIT Transactions on Information and Communication Technologies, Vol 38, © 2007 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/DATA070331

Because of its importance, cryptography is nowadays a science by itself, strongly related to other modern research fields as complexity theory, chaos, dynamical systems, computing theory etc. The state-of-the art for the field of cryptography is probably classified as it has military applications, but for the public domain a good reference can be found in [1] and [2]. The *encryption* of a message/data file/other information is a process (algorithm) that modifies this message/data file/information making it completely unintelligible, except for the person who knows the encryption *key*. The *key* refers to the encryption algorithm that has been used – in fact, to the reverse algorithm that should be used for decryption – and the particular parameters that have been used during the encryption. The *decryption* algorithm should render the original message/file/information complete and unaltered. Encryption can be achieved by constructing two different types of ciphers – stream ciphers and block ciphers. In the case of the block cipher, the message is split into successive blocks that are encryption using a single key or multiple keys. In a stream cipher the message is broken into successive bits or characters and then the string of characters is encryption using a key stream. The *cryptographic scheme* refers to the assembly of encryption and decryption algorithms. An ideal cryptographic scheme or algorithm has not been developed yet, as an ideal cryptographic scheme implies: no data expansion during the encoding process; fast encoding algorithm; small dimension key; fast decoding algorithm; correct and complete rendering of message after encryption/decryption; invulnerability to attacks. The last point is a major issue in cryptography; complex mathematical studies and research have to be done in order to establish the vulnerability of each cryptographic scheme. In simple words, this answers the basic question: how difficult is to break the code? This "difficulty" has to be established in terms of complexity, cost and computing time. Therefore, depending of the particular applications, sometimes it is enough to have a code and cryptographic scheme that requires a long search for the key, although the process is very simple. This is the situation for the briefcases with cipher, where the breaking process is quite simple: one has to try *all* possible numbers in order to find the right one. Cellular automata are applied with success in cryptography mainly because their vast phenomenology and apparently big complexity require a very long computing time to break well-chosen cryptographic schemes. There are indeed a lot of parameters and factors that can drastically affect the encryption message and therefore the complexity of the attack is considerably increased. Cellular automata offer an ideal mathematical model for massive parallel computation, but most research and applications in cellular automata domain are done through simulation. However, it is obvious that only the hardware implementations of this model fully exploit its computing and high-speed possibilities. In particular, cellular automata applications in cryptography are efficient because of the massive parallelism of the model. When implemented by means of other computing systems (simulated in software or emulated with microcontrollers etc.) the parallel processes are in fact executed sequentially. Special cellular automata hardware is the only means to benefit of all the advantages of the model.

2 The encryption algorithm

For the encryption process was chosen the RC4 algorithm. This algorithm starts with the fact that the input data are represented by a raw of 32 multiple number of bits, in case of encryption and 64 in the case of decryption. The encryption/decryption key is a raw of 64 multiple number of bits. The output data are represented by a raw of bits of a number which is equal with the double of the number of bits of the input raw at encryption, respectively with the half of the number of bits of the input raw at decryption. The main idea of the RC4 encryption/decryption algorithm is as follows:

The initial text is divided in text words of 32 bits at encryption, respectively 64 bits at decryption $t_0, ..., t_{n-1}$. The encryption key is divided in m keys of 64 bits each: $k_0, ..., k_{m-1}$. For each i between 0 and n − 1, the text word t_i will be encryption/decrypted with the k_{imodm} key.

The implementation of the function for the evolutive key initialization was done with the KSA algorithm. The purpose of this operation is to initialize the key of the system based on a key of a variable length, owned by the user. The algorithm has as input data an initial k_{init} key, of 64 bits (that is eight octets). k_{init} will be regarded as a vector of eight numbers between 0 and 255, marked as $k_{init}[0]$, $k_{init}[1]$,...$k_{init}[7]$.

After going through this algorithm, an initial key will be generated, which will consist in the input data for generating the evolutive key.

3 Cellular automata generator of evolutive key

The strong point of the encryption-decryption process is the algorithm for generating the evolutive key starting from an initial key. Even a weak encryption algorithm may become extremely strong if we use a fluid key with a high degree of invulnerability. For this reason, a big part of the research effort was concentrated on the algorithm for generating the evolutive key. The final solution is based on using the cellular automata and gives a special constructive simplicity, simultaneously with a very high complexity of the evolution of the keys.

3.1 Advantages of cellular automata for VLSI design

The design style under VLSI technology prefers the simple, modular and local connected logic circuit structure. Cellular automata are ideal from the hardware designer's point of view. The local connectivity, regularity and the simple basic components make CA very appropriate to implement low-cost and robust massive parallel machines or application-oriented circuits. The reason why there are not so many cellular-automata inspired electronic circuits is mainly the difficulty of synthesis for this computing model; however, dedicated circuits for specific applications (signal generators, associative memories, image pre-processing blocks, various simulators for natural phenomena) have been successfully designed and produced [6].

3.2 Method for generating the evolutive encryption keys, based on cellular automata

The method of the evolutive key generation (implemented and tested by the authors) is based on using the cellular automata for changing the key according to a preset algorithm, during the process of message encryption. The first effect is the decreasing of the vulnerability to attacks due to the huge space (10^{10000}) of possible states. This method implies the following:

- binary, linear cellular automata are used;
- the topology of the cellular automata's neighbourhood is defined;
- constant ***P*** is chosen which signifies the selection step of the cellular automata states;
- selection (with the KSA algorithm) of an initial state for the cellular automata and start the execution;
- in each ***m x P*** step of the evolution of the cellular automata, the ti text words of the message are bitwise combined with the corresponding cellular automata configuration, applying the algorithm described above.

For the decryption, we need to know the dimension of the cellular automata, the neighbourhood topology, the initial state, the step ***P*** for the state selection and the evolution rule of the cellular automata used during the encryption process. Thereby:

- the encryption message will be spilt in words of the given length;
- the evolution of the cellular automata will be started from the same initial state;
- in each ***m x P*** step will be bitwise combined – with the correspondent algorithm – the words of the encrypted message with the decryption keys (configurations of the cellular automata);
- the decrypted message is obtained from the bitwise combined sequence of the new words resulted from the previous operation.

The cryptographic system described in this section has the following characteristics:

- *the key* is a state of the system (the cellular automata);
- *the key has a pseudo random evolution in the encryption process;* if also the process of generating the selection step becomes pseudo random, then the identification of the key becomes practically impossible for a hacker. Even if the hacker has the initial message and the encryption message (situation which helps a lot in finding the key), it would be impossible for him to find out the key, because, in a message, the key evolved and changed many times;
- the total number of keys is 2^{2^N}, where N is the number of neighbours.

The system of evolutive keys implies the continuous change of the key by a predefined law (through the algorithm) in a huge space. Such an algorithm with evolutive keys may be efficiently implemented for real time comunication systems (of data / voice) only on parallel computing systems like cellular automata. The evolution of the encryption keys is done in a huge space of values, based on an pseudo random generating algorithm. So, if we use configuration of

1000 bits, there will be 2^{1000} binary words and $2^{2^{1000}} \cong 2^{10^{300}}$ functions to go through these binary words. Such a system is very difficult to break through stochastic methods.

3.3 Linear cellular automata with memory

Generally, the complexity of a numeral sequence generated by a cellular automata, no matter whether it is about one bit (the state of a certain cell), more bits chosen from the network, the evolution of the global configuration increases with the dimensions of the celullar automata, idle with the total number of cells. In order to obtain the same effect of diversification with a reduced cost, we took a new model: the linear celullar automata with a additional memory layer. Such a cellular automata functions by a rule, with a neighbourhood dimension 3 , is of this form: $a(t+1)=f(a_{i-1}(t), a_i(t), a_{i+1}(t), a_i(t-1))$.

The structure of this CA is presented in the figure below:

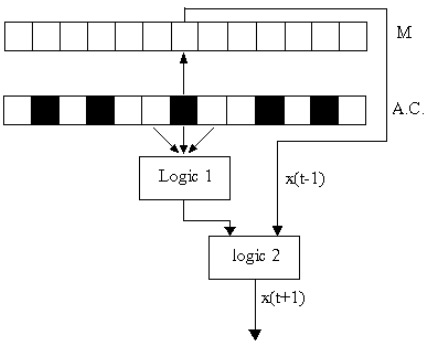

Figure 1: The structure of a cellular automata with memory layer.

In this case, the total dimension of the local rules of the space, for a homogeneous CA, with neighbourhood dimension 3, is 216. A somehow simplified roundabout line, which has a flutter on the fact that only a fraction from the total local rules are useful, implies the usage of two rules, as follows:
$a_i(t+1) = g(f(a_{i-1}(t), a_i(t), a_{i+1}(t); a_i(t-1))$.

So, first we calculate the dependence of neighbourhood and only afterwards is taken into account the previous state of the cell; this structuring reduces the space of the states to the $2^8 \times 2^4 = 212$ local possible rules and simplifies the implementation. In a way similar to the way in which the combination of two rules in hybrid structures (non homogeneous) leads to the admission of a behaviour of a different nature, and the introduction of the additional memory modifies the evolution of the cellular automata.

4 Dedicated simulation tool

The software realized in this phase of our project uses cellular automata with additional memory layer defined with specific facilities for the generators of the

evolutive keys in cryptographic applications. As a simulator of binary cellular automata (CA), the program can run:

- two topologies of cellular automata: linear and bi-dimensional;
- maximum length of the linear automata: 1000 cells;
- maximum length of the bi-dimensional automata: 1000x1000;
- neighbourhood dimension of the linear automata: 3-15 neighbours;
- maximum neighbourhood dimension of the bi dimensional automata: 9 neighbours;

The simulator displays the evolution of the cellular automata as a two-dimensional picture or an upside- down evolution chart (for linear CA). Several menus are available for parameters and topology definitions such as initial state, rules, etc. The purpose of this simulator is to acquire and analyze a huge quantity of data resulted from a large number of experiments. This can allow the selection of some strategies and best parameters for generating evolutive keys for the encryption applications. Therefore the graphic facilities were not as important as the databases facilities

5 The analysis and selection of the best strategies for encryption

The purpose of the simulator is to study and analyse different strategies and algorithms for encryption. In this direction special features were impemented for data bases and visualization of the results.

5.1 Example of encryption using a linear cellular automata with 64 bits

In the next figures is ilustrated an example of a encryption experiment realized with the program which simulates the parallel account with linear cellular automata. In this version of the simulator we work for simplicity with text messages.

Figure 2: The main 'action' window.

The messages which are going to be encrypted can be introduced from the keyboard, clipboard or from an external file. The button "Encrypt" opens another window that displays the encrypted message, obtained through the RC4 algorithm with evolutive keys generated with cellular automata. Figure 3 reproduces the window dedicated to the introduction and set up of various parameters of the keys. In the lower case one can see the keys (of 64 bits, displayed in HEXA format) which have been evolved during the encryption of the current message.

Figure 3: The window dedicated to the evolutive keys.

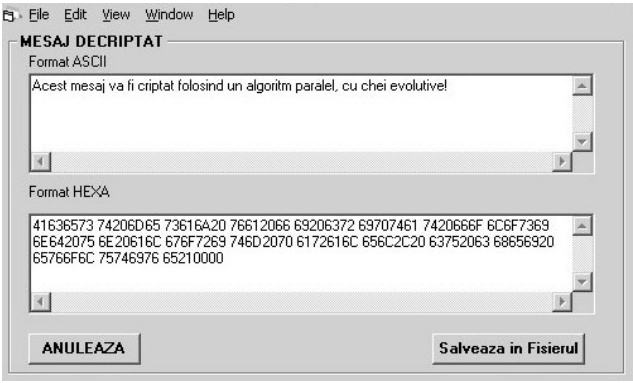

Figure 4: The decrypted message in ASCII and HEXA format.

The key can be introduced by the user from the keyboard or clipboard, which will be, in this case, a key for the user, which will initialize the key of the system with the KSA algorithm. Also, the key can be extracted from an external file or from an evolutive key library. After processing the initial message transformed in binary format, based on the evolutive keys generated by the cellular automata simulator and the RC4 algorithm, the encryption message will be generated and displayed in a new window. In this window, presented in the figure 4 the

encryption message appears automatically, after the encryption process, or may be introduced from the clipboard or an external file. It can be also saved, in order to be used in subsequent decryption or for sending it to the consignee of the message.

The encrypted message presented in one of the ways shown above, in converted ASCII format will be automatically displayed in HEXA format also. The button "decryption" will automatically open the dialog window for the key used for the decryption (in the case in which it was not previously introduced). In this window one can introduce the key using the same methods as for the encryption key and can be study the evolution of the decryption keys. Finally, the decrypted message will be displayed in a window, in both formats. As it can be seen, the experiment is successful; the decrypted message is identical with initial one "This message will be encryption using a parallel algorithm, with evolutive keys!"

6 Hardware implementation

The hardware implementation is the final stage of the project. The next figure presents the block scheme for the key stream encryption algorithms. This block scheme include a central unit (CU) (that generates the configuration signals for the word dimension, the synchronization signals and the rules for the cellular automata), the cellular automata (CA) lattice that occupies most of the chip area, the block generator (BG) that divides the original data stream in blocks or words of given dimension and a RC4 circuit.

The cellular automata lattice is a two-dimensional network of finite state machines. Even for linear topology, the cells are geometrically arranged in two-dimension and connected to obtain the linear network. The control and coordination of such a cellular automata lattice is presented in [6]. Note that the neighbourhood dimension is a fundamental parameter for the hardware implementation of the local logic circuits, as the dimension (number of logical variables) of this circuitry depends exponentially on the number of inputs. Starting from this general configuration, many simple particular cryptography tasks may be implemented for different specific products, each of them having a specific set of rules to be transposed in the PLC (programmable logic circuit).

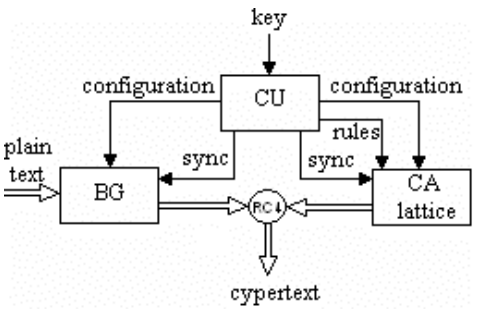

Figure 5: The block scheme for the key stream encryption.

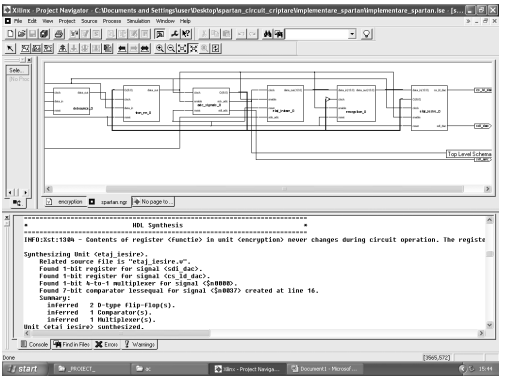

Figure 6: The cryptographic block scheme (Xilinx ISE webPack 8.1i).

6.1 Experimental implementation of the cellular automata encryption system on FPGA platform

The implementation of the encryption system was realized on Spartan 3 XC3S400 platform produced by Xilinx. This platform is specially designed for testing of the digital circuits, it is very flexible, capable to be adapted to an extremely large number of applications. The main characteristic of this platform is the circuit FPGA PQ208C which is remarkable for its number of 400.000 available logical gates. XC3S400 has also incorporated two converters which permit implementations of digital processing of an analog signal. The experimental results showed that a good encryption is possible even in the case in which we use a very small number of cells inside the cellular automata, but in this situation it has to used an evolution rule for the cellular automata, which will push it into the attraction basins with very big cycles. In figure 7 is presented the duplex communication system with two platforms of FPGA (Spartan III) during the testing of the cryptographic system.

7 Conclusions

Cipher schemes with cellular automata have already been studied and proved to be efficient and having good invulnerability properties. The paper presents the development and design of specific algorithm and corresponding hardware, using a dedicated simulator to emulate various cryptographic schemes with different parameters. The hardware implementation on FPGA platform result in cheap, robust, efficient and versatile encryption and decryption circuits that may be adapted to different devices (like phones) in order to ensure the privacy of data communication. Using the FPGA platforms offer the main advantage to be able to test, optimize and adapt at the application type, the encryption systems with parallel computing architectures which will be developed in the future stages of this project. In this direction, it is well known that only 10-15% from the encryption algorithms theoreticaly developed can be used in practical

applications, and finally, just a quater of these can be implemeted. The utilization of the cellular automata for generating evolutive keys, offers a large field for investigation for the development of high efficiency algorithms and encryption keys.

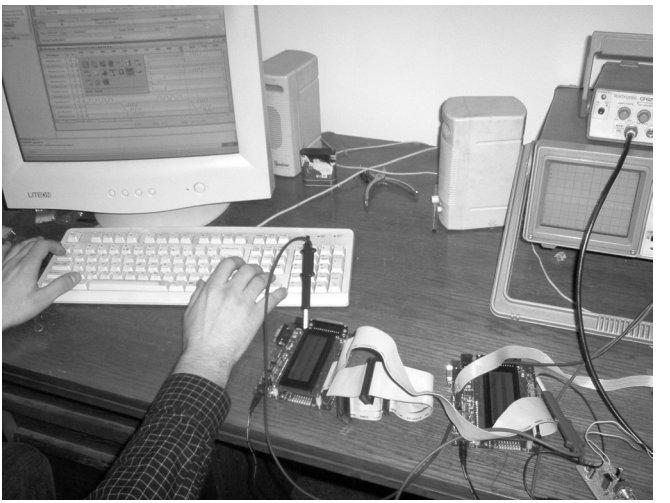

Figure 7: The implementation of the encryption system on the FPGA.

Acknowledgement

The work of this paper was done with support from SCRIPT 8/2005 project from the SECURITATE Romanian Research Program.

References

[1] William Stallings, *Cryptography and Network Security* Prentice-Hall, New Jersey, USA, 2003.
[2] Brian A. LaMacchia, *.NET Framework Security*, Addison Wesley, USA, 2002.
[3] S. Wolfram, *Cellular Automata and Complexity*, Addison-Wesley Publishing Company, 1994.
[4] J. Kari Reversibility of 2D Cellular Automata is Undecidable, *Physica D,* Vol. 45, pp. 379-385, 1999.
[5] H. Gutowitz, *Method and Apparatus for Encryption, Decryption and Authentication using Dynamical Systems* US Patent 5365589, 1994.
[6] P. Chaudhuri, D. Chowdhury, S. Nandi and S. Chattopadhyay, *Additive Cellular Automata: Theory and Applications*, IEEE Computer Society Press, 1999.