HARDWARE IMPLEMENTATION OF
A TENT MAP-BASED CHAOTIC GENERATOR

Cristian-Iulian RÎNCU[1] Vasile-Gabriel IANA[2] Gheorghe ŞERBAN[2] Ion TUTĂNESCU[2]
e-mail: r_iulian@mta.ro gabi@upit.ro serban@upit.ro tutanescu@upit.ro
[1]Communication and Military Electronic Systems Department, Military
Technical Academy, 81-83 George Coşbuc Blvd., 75275, Bucharest, Romania
[2]Department of Electronics, Communications and Computers, University of
Piteşti, Târgul din Vale, 0300, Piteşti, Romania

Key words: chaotic map, FPGA, implementation

ABSTRACT

In the last ten years the interest for chaos has increased due to the great resources represented by the dynamical chaotic systems which can be used to assure secure and wide band communications. Also, chaotic systems are widely used to design analog and digital blocks for communication systems. In this paper we present some aspects regarding the hardware implementation of one well-known digital chaotic map that is used to achieve chaotic generators.

I. INTRODUCTION

The seemingly random behaviour of chaotic phenomena would appear to have little to do with the ordered discipline required to send a sequence of 0s and 1s in a way that can be accurately and reliably received. In the last period of time there were tested communications systems using chaos techniques in order to send digital messages at Gbps speeds over one hundred kilometers using commercial optical fibre [1]. The tested system proved that chaotic generator, even it seems to be disorganized, can be controlled and used in practical applications. An important feature of chaotic systems is that their long-term behaviour is often impossible to predict but their near-term behaviour is quite easy to anticipate, so their immediate evolution can be controlled.

A transmission method based on chaotic generator can be a somewhat similar to a FM radio transmission. A message is embedded on a carrier wave, but the wave in this case is rather chaotic than sinusoidal.

Retrieving the message is simply a question of subtracting the carrier wave from the transmitted signal. One of the advantages to use chaos-based communication systems is that it is often easier to generate robust, high-power chaotic signals than conventional ones. In one demonstration, the researchers used light from commercially available laser diodes, and the transmissions proved remarkably robust. The amount of lost bits was 10^{-7} (at Gbps speeds).

It is also harder for an eavesdropper to identify a chaotic signal because it is difficult to distinguish from background noise. Of course, if the chaotic technique is not enough to assure secure communication for higher levels of privacy, the message can be easily encrypted using classical methods of cryptography, or even ones based on chaos. As it happens, there is growing evidence that nature may also employ chaos to send information [1].

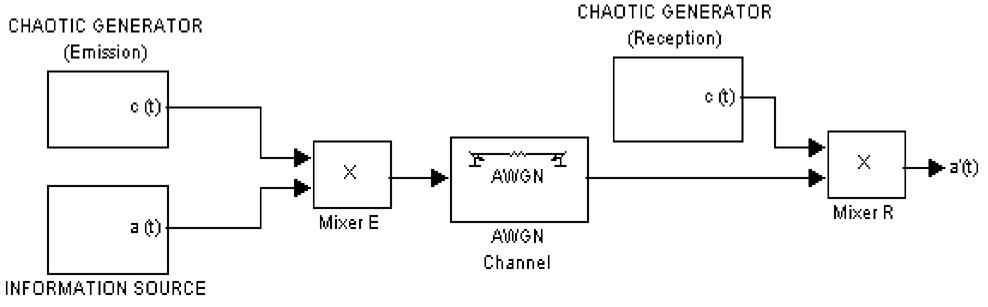

Figure 1. Chaos-based communication system

A general block diagram for a secure communication system based on chaotic techniques is presented in figure 1. An information signal is mixed with the output of a chaotic generator. The result signal contains the information, but it is invisible to one intruder that has no knowledge about the parameters and initial conditions of the transmitter's generator. The right receiver, who possesses this knowledge, reproduces the chaotic signal (sequence) or, at least, an approximation of it and is able to recover the information that was transmitted.

Over past ten years, new chaos-based spreading techniques have been developed including chaotic masking, chaotic modulation, chaos shift keying (CSK) and predictive control (or symbolic) modulation. The methods mentioned above represent techniques that are used to implement communication networks in order to protect the information and also to use efficiently the available spectrum.

II. DIGITAL IMPLEMENTATION OF THE TENT MAP

The chaotic generators, that have a random and unpredictable behaviour, can be used in communication systems, which are implemented using both analogic and digital techniques. In the last years, many chaotic generators are tested and evaluated in order to be used in communication systems. The interest is motivated by the behaviour of that kind of dynamical systems that can assure spread spectrum and security in the same time. In digital form, chaotic maps can be used to implement PRNGs (Pseudo Random Number Generators) or the digital chaotic map can be used to design and implement the block ciphers. At the same time, chaotic pseudo-random coding techniques have also developed separately in other areas, such as electronics and communications.

In an analogic form, chaotic generators can be implemented using nonlinear circuits elements but the physical parameters of those analogic components need to be constant in order to assure a permanent synchronization between emitter and receiver. In a digital form, chaotic generators can be implemented based on digital chaotic maps models using microcontrollers, DSP, FPGA or other digital processors.

In order to study and implement a chaotic generator on a FPGA (Field-Programmable Gate Array), we have chosen a well-known one dimensional chaotic map. This digital function is one type of tent map that was chosen because its properties are good to prove the utility of chaotic maps in secure communication and, in the same time, it has a simple form that is not so expensive regarding the physical resources that are necessary for implementation [2].

The tent map is defined by a piece-wise linear characteristic, and is described by the following equation:

$$x(n+1) = f_T\left(x(n)\right)$$

$$f_T(x) = \begin{cases} \dfrac{x}{p}, & 0 < x \le p \\ \dfrac{1-x}{1-p}, & p < x < 1 \end{cases} \qquad (1)$$

where p is the parameter of tent map that determines the chaotic behavior, with

$$f_T : (0,1) \to (0,1)$$

and

$$p \in (0,\, 1)$$

for good chaotic properties.

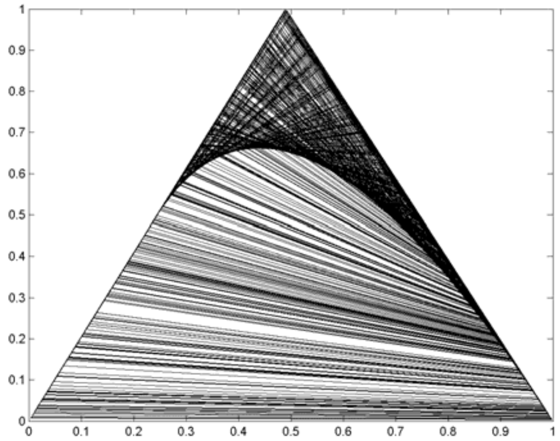

Figure 2. Tent map attractor.

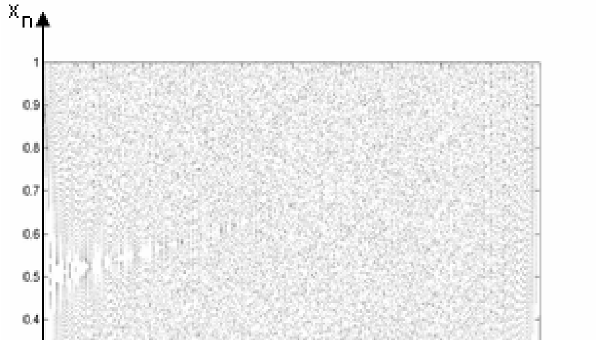

Figure 3. Tent map bifurcation diagram.

III. RESULTS AND DISCUSSION

The attractor and the bifurcation diagram for tent chaotic map are presented in figures 2 and 3 respectively.

The hardware description for the implementation of the chaotic generator is based on the tent-map equation (1). It was developed using the hardware language VHDL. The hardware algorithm was made in a close relationship with the theoretical chaotic equation to obtain the same results. The model entity for the chaotic generator is presented in figure 4. The input ports are *clk*, *load*, *p_init*, and *x_init*. The output port is only one, *out_data*.

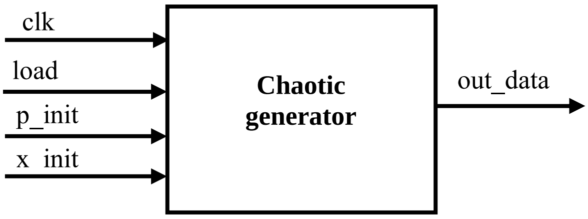

Figure 4. Entity of the chaotic generator.

The chaotic generator is made in three versions. These differ through bit number whereon operate. In this paper are chosen the data busses on 8, 16 or 32 bits. This module of chaotic map becomes synchronized after rising edge of the *clk* signal. The *load* signal is used to initialize the *p* variable and the first value of the *x[n]*. This act is made all the time when appear the falling edge of the *load* signal.

The input data have fixed point representation with 1:31 format and to other cases 1:15, respectively 1:7 formats. Only fractional numbers can be represented with the most significant bit used for sign representation. For data acquisition at maximum frequency, the program has instruction blocks implemented in parallel hardware structures. This structures work independently through separate blocks. The data is sampled and processed only when on specific input ports are detected changes. These also have one extra advantage, consisting in the fact that the structures are synchronous and give the possibility for data to be processed right after it is available. The clock signal is used for synchronization with external systems to the FPGA structures.

The schematic block of the implementation model in FPGA hardware for the chaotic generator is presented in Figure 5.

The *U1* block is used to retain the value charged on *p_init signal*. This value is taken on rising edge of the *load* signal. The binary number *p* and *x* is compared through *U2* block.

The *U3* and *U4* blocks give the binary two complement for the subtraction operation *(*$1-x[n]$*)* and (1- *p*) so that, finally, the *U5* block performs the division between *x* and *p* or (1-*x*) and *(1-p)*. These five blocks are implemented asynchronously. The *U6* is used for obtaining a single clock delay, synchronously with the *clk* signal. These blocks are concurrent processes. They are implemented structurally in hardware structures.

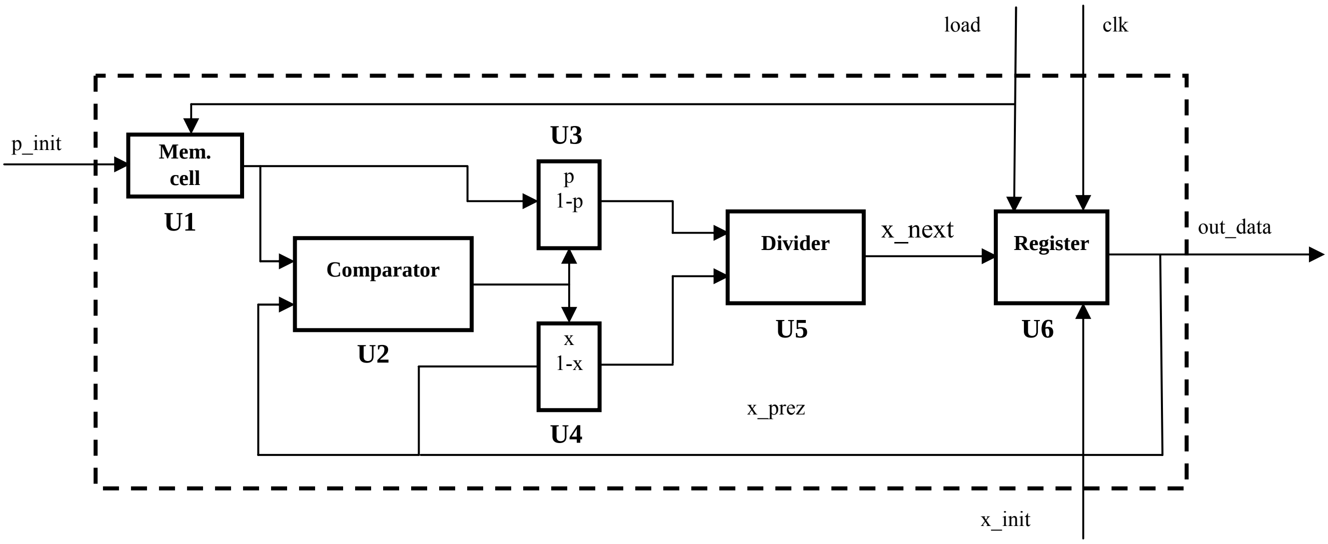

Figure 5. Schematic block of the chaotic map generator on 32 bits.

The testing model was designed for functional circuit verification. This consists in an integrated environment, where the projects (also called Unit Used for Testing - UUT) are verified by applying stimulus signals and monitoring the systems responses.

In other words, the simulation model substitutes the environment where the design generator will function so that we can observe and analyse its behaviour. Project testing is performing in two steps:

In the first step, the project initialization is verified by applying stimulus using the test simulation module. As long as *load* signal is ‘1’, the output *data_out* is initialized with *x_init* value. The first data from the output signal (a numeric sample) is obtained of the first rising edge of the *clk* signal (figure 6).

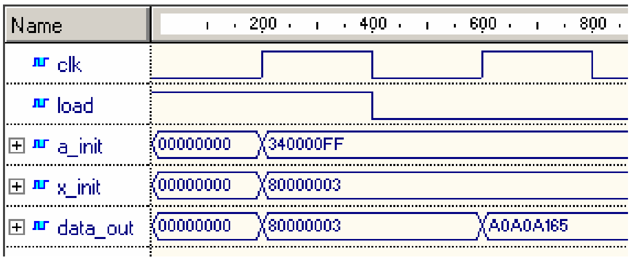

Figure 6. Initialization of the tent. map

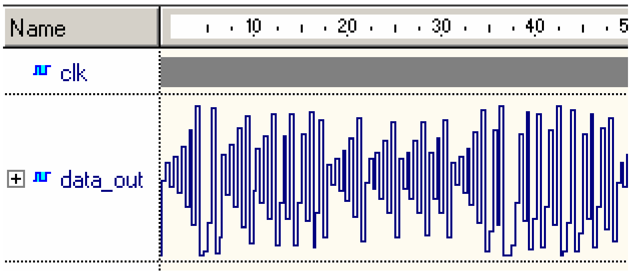

Figure 7. Simulation for module with data on 32 bits.

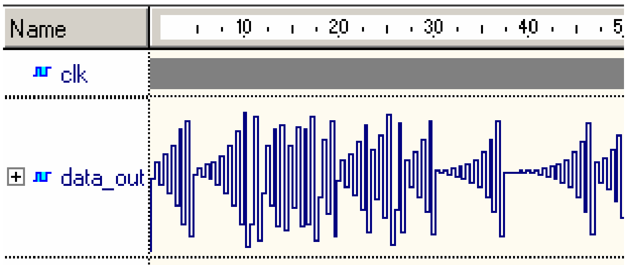

Figure 8. Simulation for data on 16 bits.

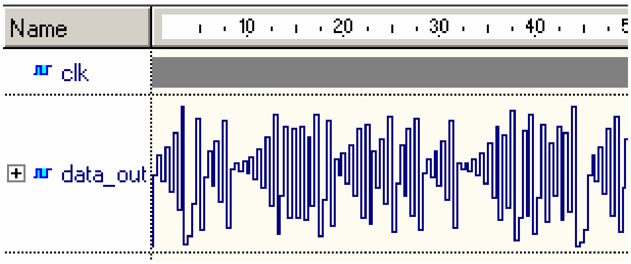

Figure 9. Simulation for data on 8 bits.

In the second step, the functional verification of the whole project is performed. Figures 7, 8 and 9 present the graphics behaviour of the tent map for the three modules with data path on 32, 16 and 8 bits respectively

The area usage and performances from Table 1 were obtained for chaotic generators implemented using *FPGA xc2vp30-5ff896* .

Table 1. Synthesis report.

Data path	Area usage	Speed
32 bits	2672 slices (19%)	6.247MHz
16 bits	666 slices (4%)	15.049MHz
8 bits	173 slices (1%)	31.888MHz

In order to evaluate the possibility to use this chaotic generator in secure applications we have evaluated the randomness of the output sequences. In all the computing representation forms, using the Discrete Fourier Transform (Spectral) Test applied to the output sequences, obtained for the designed generator, we have got a behaviour that is random, as is presented in Figures 10, 11 and 12, but that not assure best performances. Of course, that is not difficult to add in the digital structure a LFSR that can perturb the trajectory of the chaotic generator [2],[3].

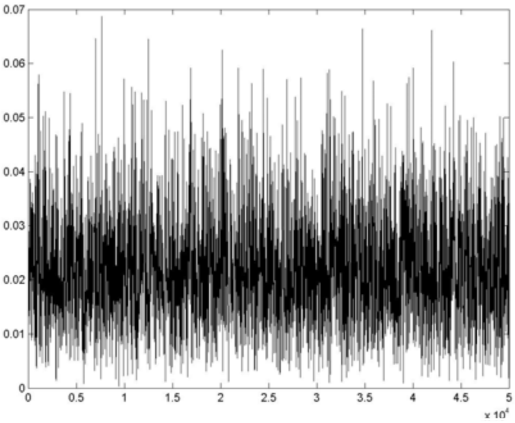

Figure 10. FFT representation for the tent generator on 32 bits.

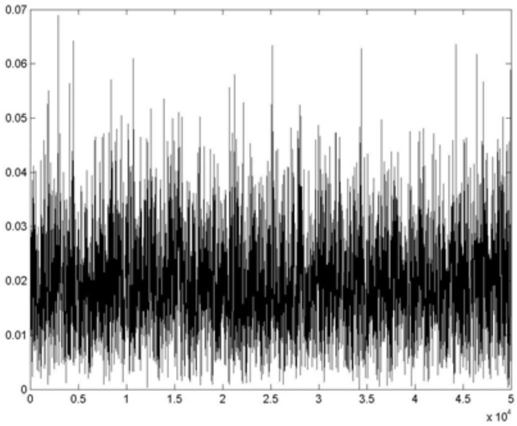

Figure 11. FFT results for 16 bits.

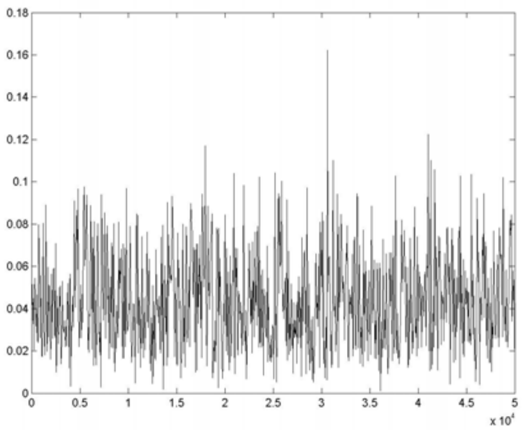

Figure 12. FFT results for 8 bits.

IV. CONCLUSIONS

Communication systems based on chaos theory have exceeded the stage of the laboratory simulations using Matlab or other designing tool.

This paper has pointed out the moment of the implementation using reprogrammable hardware structures [4]. We have presented some aspects on the digital implementation of one well-known digital chaotic map that can be used in communication systems to develop chaotic generators.

We have done some tests with the generator output values and we conclude that are no differences between the values obtained using the FPGA simulator and the others obtained using Matlab like in the logistic map case [5].

Using the FFT randomness test we have proved that tent map chaotic generator has a random behaviour and can be used in some real applications. The results obtained for the tent map are better then the others obtained for logistic map [5], [6], even we use only 8 bits for representation.

A secure data transfer can be achieved using a tent map generator instead a logistic one although for the first one we need more hardware resources and is a little bit more slowly. A higher throughput of the chaotic generator can be obtained by using dedicated circuits (*ASICs*), field programmable logic arrays (*FPGAs*) or faster general purpose processors.

REFERENCES

1. *IEEE Spectrum*, January 2006

2. S. Li, Q. Li, W. Li, X. Mou, Y. Cai, "*Statistical properties of digital piecewise linear chaotic maps and their roles in cryptography and pseudo-random coding*", Cryptography and Coding–8th IMA Int. Conf. Proc., Lecture Notes in Computer Science 2260, Springer-Verlag, 2001, pp. 205–221

3. Shujun Li, Xuanqin Mou, Yuanlong Cai, Zhen Ji and Jihong Zhang, "*On the security of a chaotic encryption scheme: problems with computerized chaos in finite computing precision*", Computer Physics Communications, vol. 153, no.1, pp. 52-58, 2003

4. K.H. Tsoi, K.H. Leung and P.H.W. Leong, "*Compact FPGA-based True and Pseudo Random Number Generators*", Proceedings of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, p.51, April 09-11, 2003

5. C.I. Rîncu, V.G. Iana, *Aspects of Digital Chaotic Map Implementations*, Communications 2006, Bucureşti, 2006

6. G. Iana, A. Şerbănescu, *Physical implementation of the chaos-based communication systems,* The 31th internationally attended conference *Modern Technologies in the XXI Century* of Military Technical Academy, Bucharest, 03-05 November 2005

7. I. Tutănescu, "*Some Methods and Results for increasing Communications Security using Chaotic Carrier Systems*", NATO RCMCIS 2003 International Conference, 5th edition, October 8-10, 2003, Zegrze, Poland, published in Conference Proceedings' CD-ROM

8. I.Tutănescu et al., *"Secure communications using nonlinear dynamic systems*", International Conference SSIA 2003, September 18-20, Iasi, published in Conference Proceedings' CD-ROM