Design and Hardware Implementation of a Chaotic Encryption Scheme for Real-time Embedded Systems

Amit Pande and Joseph Zambreno
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011
Email: {amit, zambreno}@iastate.edu

***Abstract*—Chaotic encryption schemes are believed to provide a greater level of security than conventional ciphers. In this paper, a chaotic stream cipher is first constructed and then its hardware implementation details using FPGA technology are provided. Logistic map is the simplest chaotic system and has a high potential to be used to design a stream cipher for real-time embedded systems. The cipher uses a pseudo-random sequence generator based on modified logistic map (MLM) and a random feedback scheme. MLM has better chaotic properties than the logistic map in terms of uniformity of bifurcation diagram and also avoids the stable orbits of logistic map, giving a more chaotic behavior to the system. The proposed cipher gives 16 bits of encrypted data per clock cycle. The hardware implementation results over Xilinx Virtex-6 FPGA give a synthesis clock frequency of 93 MHz and a throughput of 1.5 Gbps while using 16 hardware multipliers. This makes the cipher suitable for embedded devices which have tight constraints on power consumption, hardware resources and real-time parameters.**

Keywords: chaos, encryption, stream cipher, FPGA implementation

I. Introduction

Chaotic systems are characterized by sensitive dependence on initial conditions, similarity to random behavior, and continuous broad-band power spectrum. The possibility for self-synchronization of chaotic oscillations [1] has sparked an avalanche of works on the application of chaos in cryptography. The random behavior and sensitivity to initial conditions and parameter settings allows chaotic systems to fulfill the classic Shannon requirements of confusion and diffusion [2]. A tiny difference in the starting state and parameter setting of these systems can lead to enormous differences in the final state of the system over a few iterations. Thus, sensitivity to initial conditions manifests itself as an exponential growth of error and the behavior of system appears chaotic.

Several schemes have been developed to exploit this property of chaotic systems for secure communications. Some of them are chaotic masking, chaos shift keying, and chaotic modulation. Continuous time chaotic systems have been researched but they require repeated resynchronization to match the phase at the encoding and decoding ends.

Discrete-time chaotic systems behave like private-key encryption algorithms [3] and are amenable to implementation in fixed point hardware. They can be broadly divided into two types: chaotic block ciphers and chaotic stream ciphers. The work by Baptista [4] was one of the earliest attempts to build a block cipher based on chaotic encryption. Each character of the message is encrypted as the integer number of iterations performed in the logistic equation, in order to transfer the trajectory from an initial condition towards a predefined interval inside the logistic chaotic attractor. However, its encryption speed is very slow since at least 250 iterations of the chaotic map are required for encrypting an 8-bit symbol. The number of iterations may vary up to 65532.

Chaotic encryption has also been used to design image and signal encryption schemes [5], [6]. However, many schemes have proven to be weak against cryptanalysis using known-plaintext attacks and others [7], [8].

A stream cipher based on chaotic map was presented in early 1991 by [9] and its cryptanalysis was presented by [7]. Chen et al. [10], [11] constructed a block cipher based on three-dimensional maps while [12] proposed a cipher by direct discretization of two dimensional Baker map. A good survey and introductory tutorial on these schemes is found in [13], [14]. [15] present a crypto-system based on a discretization of the skew tent map. [16] presents chaotic Feistel and chaotic uniform operations for block ciphers. Although various schemes/ maps have been proposed in research literature, the logistic map remains one of the simplest maps and is used in many schemes.

Some of the factors influencing the design of a good chaotic stream cipher for real-time applications are as follows:

1) The scheme must be resistant to known cryptographic attacks. Many chaotic ciphers have been found to be insecure against plain-text attacks.
2) The range of the control parameter must be large enough to withstand any brute-force cryptographic attacks.
3) A hardware implementation requiring lesser hardware resources and achieving high throughput is desired.
4) The dynamics of discrete chaotic systems is different than those for the continuous-time chaotic systems. Discretization leads to severe degradations such as short cycle-length, non-ideal distribution and correlation, etc. These issues need to be properly addressed in the design of a chaotic crypto-system.

In this paper we address these issues and present the design and implementation of a chaotic stream cipher that uses less

hardware, has promising security and has high throughput to serve the requirements of real-time embedded systems. The main contributions of this paper can be summarized as under:

1) We present a Modified Logistic Map which has better properties than the Logistic Map - in terms of higher confusion (larger Lyapunov exponent) and a flatter distribution for various parameter values in the bifurcation diagram.
2) To the best knowledge of the authors, this is the first hardware implementation of a chaotic stream cipher.
3) We present an optimized implementation of 64 bit multiplication in FPGA leading to savings in hardware resource requirements.
4) A throughput of 1.5 Gbps was obtained for Virtex-6 XCVLX75TL FPGA. The design was synthesized and implemented using the Xilinx ISE 11.0 tool.

II. Algorithmic Description

The logistic map is a polynomial mapping of degree 2. It demonstrates chaotic behavior although using a simple non-linear dynamical equation. Mathematically, the logistic map is written as

$$x_{n+1} = \lambda_{LM} \times x_n(1 - x_n)$$

where λ_{LM} is a positive number.

The behavior of logistic map is dependent on the value of λ_{LM}. At $\lambda_{LM} \approx 3.57$ is the onset of chaos. Most values beyond 3.57 exhibit a chaotic behavior, but certain isolated values of λ_{LM} appear to show non-chaotic behavior and are referred to as islands of stability. Beyond $\lambda_{LM} = 4$, the values eventually leave the interval $[0, 1]$ and diverge for almost all initial values.

A rough description of chaos is that chaotic systems exhibit a great sensitivity to initial conditions – a property of the logistic map for most values of λ between about 3.57 and 4. This stretching-and-folding does not just produce a gradual divergence of the sequences of iterates, but an exponential divergence, evidenced also by the complexity and unpredictability of the chaotic logistic map.

The Modified Logistic Map (MLM): Our initial experimentation involved generation of pseudo-random number sequences by varying the parameter λ_{LM} in the range $[3.57, 4]$. It led to several observations:

1) The histogram obtained for different λ_{LM} values (with 50000 samples) is skewed and not uniform or flat. This is illustrated for the $\lambda_{LM} = 3.61$ and $\lambda_{LM} = 3.91$ values in Figure 1(a-b).
2) For $\lambda_{LM} = 4$, the logistic map equation $x_{n+1} = \lambda_{LM} \times x_n(1 - x_n)$ has the same domain and range intervals $(0, 1)$. For $\lambda_{LM} < 4$ and input x_n in range $(0, 1)$, the range of x_{n+1} in the expression is $(0, \lambda_{LM}/4]$ and the distribution of random numbers is biased towards 0 or 1 (as seen in the distributions in Figure 1(a-b)). It is desirable to have a distribution of random numbers symmetric around 0.5.

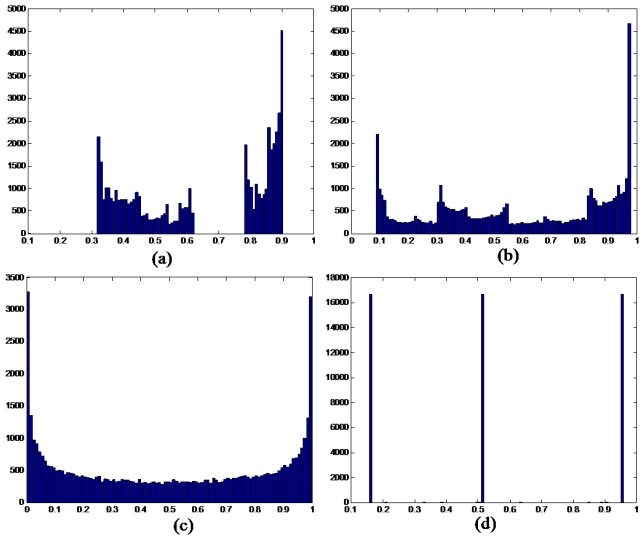

Fig. 1. Histogram for 50000 samples obtained using Logistic map with initial seed 0.100010 and (a) $\lambda_{LM} = 3.61$ and (b) $\lambda_{LM} = 3.91$ (c) $\lambda_{LM} = 4$ and (d) $\lambda_{LM} = 3.83$

3) There are certain isolated values of λ_{LM} that appear to show non-chaotic behavior and are referred to as islands of stability. For example: $\lambda_{LM} = 1 + \sqrt{(8)} \approx 3.83$ show oscillation between three values.
4) The distribution for $\lambda_{LM} = 4$ is most flat and symmetric (see figure 1(c)). It is desirable to have a flatter distribution of samples drawn from the logistic map in order to increase its randomness.

We address these issues by developing a MLM, defined by the following equation:

$$x_{n+1} = \lambda \times x_n(1 - x_n) + \mu$$

where the x_n values are restricted to the interval $[\alpha, 1 - \alpha]$, $\alpha < 0.5$. The maxima of this function occurs at $x_n = 0.5$ and the maximum value is $\lambda/4 + \mu$ while the minimum (in specified domain) occurs at $x_n = \alpha$ or $x_n = 1 - \alpha$ and the minimum value is $\lambda \times \alpha(1 - \alpha) + \mu$. Equating the maximum and minimum values to the range $[\alpha, (1 - \alpha)]$ leads to the following equations:

$$\alpha = \lambda\alpha(1 - \alpha) + \mu$$

$$1 - \alpha = \frac{\lambda}{4} + \mu$$

On solving these equations, we get $\lambda = \frac{4}{1-2\alpha}$ and $\mu = \frac{\alpha(2\alpha-3)}{1-2\alpha}$. Substituting these values, we get a flatter histogram for the new logistic map as evident in Figure 2. This modified logistic map addresses the requirements of flatter and symmetric distribution and also avoids islands of stability by generating a flat distribution for all values of α.

Quantization

The output of the modified logistic map (x_n) is quantized to get a 16 bit value y_n. The value x_n, $(0 < x_n < 1)$ is

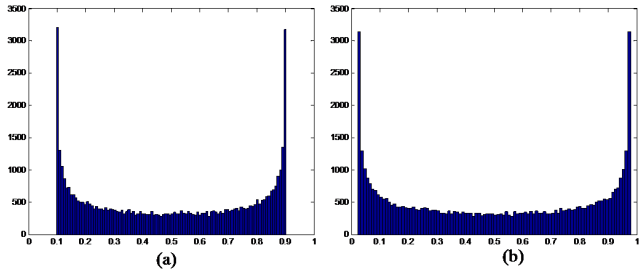

Fig. 2. Histogram for 50000 samples obtained using Modified Logistic map with α values corresponding to (a)$\lambda_{LM} = 3.61$ and (b) $\lambda_{LM} = 3.91$

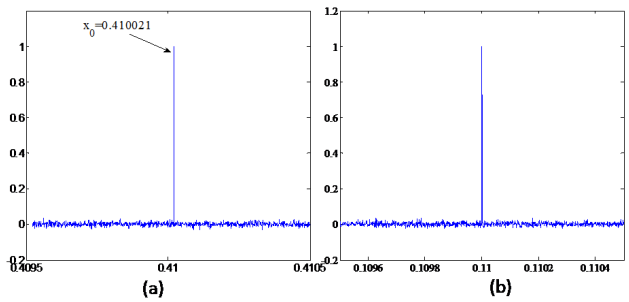

Fig. 3. Correlation test of the pseudo-random sequence. (a) Generated using different initial values x_0 and (b) different initial parameter α. The plots are measured against initial value $\alpha = 0.110000$ and $x_0 = 0.410021$

represented in fixed point as follows:

$$x_n = \sum_{j=0}^{N-1} \{a_j\} \times 2^{j-N}$$

where a_j are individual bit values.

We target the hardware architectures to use N bit fixed point arithmetic (N¿16) but restrict y_n to the least significant 16 bits only. The larger the value of N, the more secure our system becomes. Thus, y_n is given by:

$$y_n = \sum_{j=0}^{15} \{a_j\} \times 2^{j-N}$$

The quantization step or truncation of more significant bits is non-linear in nature (it is a many-one mathematical function) thereby increasing the complexity of any attacks that try to recover the logistic map information from the cipher text using any cryptanalysis. We extract another single bit from the logistic map output which is used later for the random feedback scheme. For example, the single bit output sequence b_n can be obtained from the bits of x_n as follows:

$$b_n = \{a_{N-1}\}$$

i.e. the MSB of x_n is used to get b_n.

Pseudo-Random Sequence -2

We generate another pseudo-random sequence z_n from the given sequence y_n by the following operation:

$$z_n = y_n \oplus y_{n-1} \oplus y_{n-2}$$

There is no linear correlation between the two sequences y_n and z_n. Statistical de-correlation makes it difficult to back-track y_n from z_n.

Masking Operation and Random Feedback

The ciphertext C_n is obtained from the plaintext P_n by the following operation:

$$C_n = P_n \oplus z_n \oplus Fb_n$$

where z_n is the pseudorandom sequence and Fb_n is the random feedback input from the past ciphertext output. The value Fb_n is obtained as follows:

$$Fb_n = \left\{ \begin{array}{l} C_{n-1} \text{ when } b_n = 0 \\ C_{n-2} \text{ when } b_n = 1 \end{array} \right\}$$

The performance and accuracy of discrete chaotic ciphers is a translation of properties of the underlying dynamical system (or chaotic map). The chaotic properties of logistic maps and hence MLM have been established in the past decades by several researchers [17]. Some of the necessary conditions for a secure stream cipher are long period, large linear complexity, randomness and proper order of correlation immunity [3]. A long period is assured by taking a large value of N (say 64). Figure 3 (a) and (b) show the low correlation between sequences obtained using slightly different (a) initial value x_0 and (b) parameter λ. It can be seen that a very poor correlation is obtained amongst sequences generated using slightly different initial condition or parameter.

III. Security

A serious drawback of chaotic crypto systems is that they are weak against known-plaintext attacks. If the plain-text and the cipher-text are known, it is easy to XOR both the values and obtain the key value that was XORed to the original plaintext. Our proposed scheme lays many practical difficulties against such reverse engineering:

- The random feedback scheme makes it difficult to predict the key value XORed to the original plaintext.
- The sequences z_n and y_n are linearly uncorrelated from each other making it difficult to reverse engineer the values of y_n from z_n.
- The sequence y_n is obtained by sampling of x_n which is used to iterate the chaotic map. In the hardware implementation (presented in the next section), we sample the Least Significant 16 bits (out of 64) of x_n to get y_n. Because the chaotic map is more sensitive to the MSB than to the LSB (and we have 48 unknown MSB bits), it is extremely difficult to trace back the x_n value.
- We allowed 100 iterations of MLM in the beginning to allow the diffusion of initial key bits and parameter values. It was found that within approximately 20 iterations of Logistic Map the initial parameter values are fully diffused: the two logistic maps with a slight difference in initial conditions will appear completely de-correlated in their outputs after at most 20 iterations. Allowing 100

iterations, help us to be on a safer side to allow full diffusion of the initial key parameters.

Thus, the presented scheme is secure against known-plaintext attacks. In the next section we present a hardware implementation of the scheme that uses a 128 bit encryption key (64 bits each for initial condition and parameterλ settings).

IV. Hardware Implementation

For hardware implementation, we chose a fixed-point implementation over a floating point implementation because fixed- point operations can be implemented more efficiently in hardware. The bitwidth of the Plaintext and the Ciphertext are 16 bits or 2 bytes. However, for the implementation of MLM, we chose a bit width of 64 bits. Thus, the iterating value of MLM ($x(i)$ and the parameters λ and μ, are all implemented with 64 bits fixed point precision.

The permissible range of parameter α was chosen to be $(0, 0.375)$ which is represented in fixed point with 0 integer bits and 64 fractional bits. This is represented shortly as 0.64 in Q I.F format. The range for parameter λ is then calculated to be $(4, 16)$ which is implemented with 5.59 Q I.F format. The range for μ is $(-3, -15.0975)$ which is represented using 5.59 Q I.F format. Thus, the multiplication $\lambda \times x(i) \times (1-x(i))$ is truncated to 5.59 I.F format and then added to μ to obtain the new value for $x(i)$.

The parameter α can take 3×2^{61} values while the parameter x_0 can take approximately 2^{63} values. Thus, we get an effective keyspace with 3×2^{124} or approximately 2^{125} key values to choose from.

We synthesized the design over a Xilinx Virtex-6 XCVLX75TL FPGA using Xilinx ISE 11.0. The new XtremeDSP DSP48E1 slice in Virtex-6 SXT series facilitates faster and optimized DSP functions (including multiplications). They can deliver over 1 TeraMACs at 550 MHz with up to 2016 user-configurable XtremeDSP DSP48E1 slices and cuts the power consumption by 65% using innovative, efficient power management. A direct implementation of the design gave a clock frequency of 35 MHz. By adding two pipelining stages to the multiplier (DSP48E1 slices), we obtain a clock frequency of 70 MHz for the design.

A single DSP48E1 slice can perform a maximum of 25×18 bits multiplication and hence 12 slices are required for a 64×64 bits multiplication. Two multiplication require 24 DSP48E1 slices. However, since we truncate the 128 bit output of 64×64 bits multiplication to only 64 bits, some optimization is possible. Xilinx XST (synthesis tool) reduces one DSP48E1 slice by optimization thus requiring 23 slices for implementation.

We present an optimization of usage of DSP multipliers based on above observations for the multiplication of two 64 bit numbers X and Y. X is sign extended to 72 bits (X_{SE}) and represented by $X_a X_b X_c$ where X_a,X_b and X_c are each 24 bit long sequences.

$$\{X_{SE}\}_0^{71} = \{X_a\}_{48}^{71}\{X_b\}_{24}^{47}\{X_c\}_0^{23}$$

Similarly, we can represent Y as combination of four 16 bit numbers $Y_w Y_x Y_y Y_z$.

$$\{Y\}_0^{71} = \{Y_w\}_{48}^{63}\{Y_x\}_{32}^{47}\{Y_y\}_{17}^{31}\{Y_z\}_0^{15}$$

Numerically,

$$X = X_{SE} = X_a \times 2^{48} + X_b \times 2^{24} + X_c$$

and

$$Y = Y_w \times 2^{48} + Y_x \times 2^{32} + Y_y \times 2^{16} + Y_z$$

The product $X \times Y$ can then be represented as:

$$\begin{aligned} X \times Y = (X_a \times 2^{48} + X_b \times 2^{24} + X_c) \times (Y_w \times 2^{48} \\ + Y_x \times 2^{32} + Y_y \times 2^{16} + Y_z) \\ \Rightarrow X \times Y = 2^{96} \times X_a Y_w + 2^{72} \times X_b Y_w + 2^{48} \times X_c Y_w \\ + 2^{80} \times X_a Y_x + 2^{56} \times X_b Y_x + 2^{32} \times X_c Y_x \\ + 2^{64} \times X_a Y_y + 2^{40} \times X_b Y_y + 2^{16} \times X_c Y_y \\ + 2^{48} \times X_a Y_z + 2^{24} \times X_b Y_z + 2^{0} \times X_c Y_z \end{aligned}$$

Now, considering the product $X_n(1 - X_n)$ in the logistic map, we multiply two 0.64 Q I.F values to get an output which is in 0.128 I.F format. We truncate the last 64 bits to get the 64 bit approximate value of X_{n+1}. Because X is represented in 72 bits, we can discard lower 72 bits of the product. Each of the product $X_\alpha Y_\beta$, such that $\alpha \in \{a, b, c\}$ and $\beta \in \{w, x, y, z\}$ is of size 40 bits and can be implemented in a single DSP48E1 slice.

Thus,

$$\begin{aligned} X \times Y = & 2^{96} \times X_a Y_w \quad + 2^{72} \times X_b Y_w + 2^{48} \times X_c Y_w \\ & + 2^{80} \times X_a Y_x \quad + 2^{56} \times X_b Y_x \\ & + 2^{64} \times X_a Y_y \quad + 2^{40} \times X_b Y_y \\ & + 2^{48} \times X_a Y_z \end{aligned}$$

The other multiplication operation can also be optimized in a similar manner. Thus, we can reduce the hardware requirements and critical path for the implementation. A direct implementation of our scheme using the above optimization achieved a clock frequency of 44 MHz on the above mentioned FPGA. By adding two pipelining stages to the 64×64 bits multiplier, we obtained a clock frequency of 93 MHz and required only 16 DSP48E1 slices in the design. The design summary are given in Table I. Further pipelining may lead to higher clock frequency but will also increase slice registers usage.

Figure 4 gives the block diagram of the hardware implementation of the encryption scheme. The input x_{n-1} is first multiplied with $(1 - x_{n-1})$ and the upper half bits (most significant bits) are then multiplied with λ. The output of this multiplication is then truncated and added with μ to get the value of x_n as shown in the figure. The output x_n is also used to extract the values y_n and b_n, both of which serve to generate the output cipher text. As shown in the figure, the multiplexer

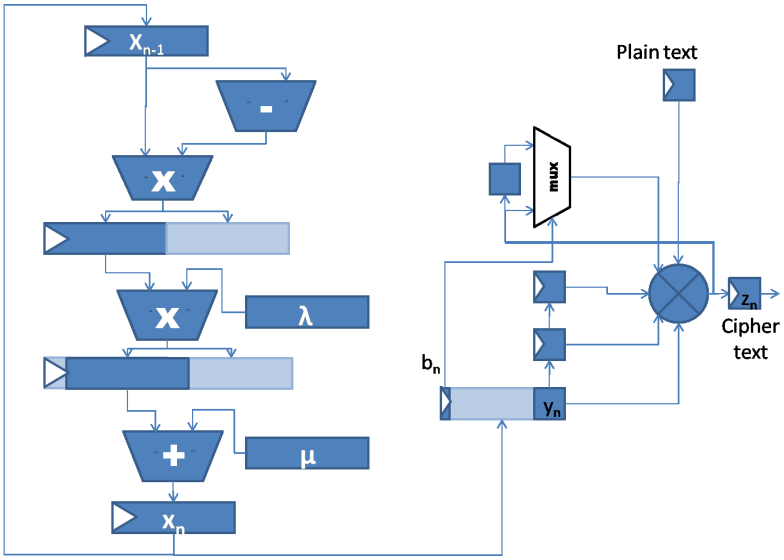

Fig. 4. Block Diagram showing the implementational details of the chaotic stream cipher

	Orig. Design	Opt. Design
Clock Frequency (MHz)	69	93
No. DSP48E1 slices	23	16
No. Slice Registers	228	160
No. Slice LUTs	354	643

TABLE I
RESOURCE UTILIZATION ON XILINX VIRTEX-6 FPGA

(mux) is used to provide the random feedback based on bit b_n.

Table I gives the resource utilization on a Xilinx Virtex-6 FPGA. It can be seen that the clock frequency achievable through the original design was 69 MHz and 23 DSP48E1 multiplier slices were utilized. The proposed optimization reduced the requirements of DSP48E1 slices from 23 to 16 leading to a subsequent increase in achievable clock frequency from 69 MHz to 93 MHz. By pipelining, we can further increase the clock frequency of the design.

V. CONCLUSION

This paper presents a novel chaotic stream cipher based on modified logistic map suitable for embedded real-time applications. A hardware implementation of proposed scheme was proposed and a clock frequency of 93 MHz was achieved.

A possible direction for future work is the study of more complex chaotic maps, study of behaviors of coupled-chaotic maps and their implementation over hardware platforms.

REFERENCES

[1] L. M. Pecora and T. Carroll, "Synchronization in chaotic systems," *Physical Review Letters*, vol. 64, no. 8, pp. 821–824, 1990.

[2] C. E. Shannon, "Communication theory of secrecy systems," *Bell Systems Technical Journal*, vol. 28, pp. 656–715, 1949.

[3] R. Rueppel, *Analysis and design of stream ciphers*. Springer, Berlin, 1986.

[4] M. S. Baptista, "Cryptography with chaos," *Physics Letters*, vol. 240, no. 1-2, pp. 50–54, 1998.

[5] M. Hamdi and N. Boudriga, "Four dimensional chaotic ciphers for secure image transmission," in *IEEE Intl. Conf. Multimedia and Expo*, 2008, pp. 437–440.

[6] R. Bose and S. Pathak, "A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system," *IEEE Trans. Circuits and Systems I*, vol. 53, no. 4, pp. 848–857, April 2006.

[7] E. Biham, "Cryptanalysis of the chaotic-map cryptosystem suggested at EUROCRYPT'91," in *Advances in Cryptology EUROCRYPT 91, LNCS*, 1991, pp. 532–534.

[8] G. lvarez, F. Montoya, M. Romera, and G. Pastor, "Cryptanalysis of dynamic look-up table based chaotic cryptosystems," *Physics Letters A*, vol. 326, no. 3-4, pp. 211–218, 2004.

[9] T. Habutsu, Y. Nishio, I. Sasase, and S. Mori, "A secret key cryptosystem by iterating a chaotic map," in *Advances in Cryptology EUROCRYPT 91, Lecture Notes in Computer Science*, 1991, pp. 127–140.

[10] Y. M. Guanrong Chen and C. K. Chui, "A symmetric image encryption scheme based on 3d chaotic cat maps," *Chaos, Solitons and Fractals*, vol. 21, no. 3, pp. 749–761, 2004.

[11] G. C. Yaobin Mao and S. Lian, "A symmetric image encryption scheme based on 3d chaotic baker maps," *Intl J Bifurcat Chaos*, vol. 14, no. 10, pp. 3613–3624, 2004.

[12] F. Pichler and J. Scharinger, "Finite dimensional generalized baker dynamical systems for cryptographic applications," in *EUROCAST '95: Select. Papers Fifth Intl. Work. Computer Aided Systems Theory*, 1996, pp. 465–476.

[13] T. Yang, "A survey of chaotic secure communication systems," *International Journal of Computational Cognition*, vol. 2, no. 2, 2004.

[14] L. Kocarev, "Chaos-based cryptography: a brief overview," *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6–21, 2001.

[15] N. Masuda and K. Aihara, "Cryptosystems with discretized chaotic maps," *IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 1, pp. 28–40, Jan 2002.

[16] N. Masuda, G. Jakimoski, K. Aihara, and L. Kocarev, "Chaotic block ciphers: from theory to practical algorithms," *IEEE Trans. Circuits and Systems I*, vol. 53, no. 6, pp. 1341–1352, June 2006.

[17] R. M. May, "Simple mathematical models with very complicated dynamics," *Nature*, vol. 261, pp. 459–467, June 10 1976.