International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:2, 2015

A Very Efficient Pseudo-Random Number Generator Based On Chaotic Maps and S-Box Tables

M. Hamdi, R. Rhouma, S. Belghith

***Abstract*—**Generating random numbers are mainly used to create secret keys or random sequences. It can be carried out by various techniques. In this paper we present a very simple and efficient pseudo random number generator (PRNG) based on chaotic maps and S-Box tables. This technique adopted two main operations one to generate chaotic values using two logistic maps and the second to transform them into binary words using random S-Box tables.

The simulation analysis indicates that our PRNG possessing excellent statistical and cryptographic properties.

***Keywords*—**Chaotic map, Cryptography, Random Numbers, Statistical tests, S-box.

I. Introduction

IN recent years, a variety of pseudo random number generators have been proposed. They are based on different methods [1] such as those based on nonlinear principle [2].

The theory of nonlinear systems has been applied to cryptography in order to increase the level of security. For this reason many schemes of chaotic systems [3], [4] are proposed to cure the weaknesses of classical encryption algorithms such as DES, AES and RSA. They have become less efficient for securing multimedia data in real time.

The methods of PRNG using chaotic systems can produce high-quality random numbers and good cryptographic characteristics [5]-[7].

Cryptography chaotic function is much more powerful than all existing ciphers [8]. Due to the non-deterministic nature of the Random Number Generator, it is virtually impossible to crack a cipher chaos [9].

We propose, for the present, a new pseudo-random number generator based on two chaotic functions and S-Box tables.

Therefore, we analyze the performance of our proposal and we compare it with performance of AES algorithm.

II. Concept of Chaotic and Iterative System

A. Chaotic System

A system is called chaotic when its evolution in time is so sensitive to initial conditions we can't predict exactly what condition it will be if we wait too long.

A chaotic map is defined as a discrete-time autonomous system:

$$f:]0,1[\longrightarrow]0,1[$$
$$x_i \longmapsto x_{i+1}$$

Mimoun Hamdi, Rhouma Rhouma, and Safya Belghith are with the National Engineering School of Tunis (NEST), Tunisia (e-mail: my.hamdi@gmail.com, rhoouma@gmail.com, safyabelghith@yahoo.fr).

Starting from an arbitrary initial condition $x_0 \in]0,1[$, it generates the sequence $x_1, x_2 \ldots x_{i+1}$.

Chaotic systems are dynamic systems that evolve in a bounded region, which have infinite non-periodic trajectories. They are very sensitive to initial conditions [9]: two very similar initial conditions lead to two paths rapidly moving away from each other. Indeed, the high sensitivity of chaotic systems to changes in initial conditions and parameters leads to the generation of many different chaotic sequences when necessary. However, the selection of a chaotic application able to satisfy these requirements remains a great challenge. For a chaotic application is applicable as PRNG, it must satisfy the random signals criteria.

III. Chaotic Map

The chaotic map chosen for our proposal is one of the most known chaotic maps. It is the logistic map defined by:

$$x_{i+1} = \lambda \cdot x_i(1 - x_i), x_0 \in]0,1[\,, 0 < \lambda < 4 \qquad (1)$$

where λ is the control parameter and x_0 is the initial condition. It is assigned to an internal state variable before the first iteration. The dynamic system changes is in]0,1[, and it is depending on the behavior of λ. From a certain value of λ, the iterations of the above equation lead to a chaotic regime. According to any studies on the theory of logistic map a good value λ between 3.6 and 4 [10].

Note that in most cases the distribution of values of the chaotic function is not completely uniform; indeed the intensity values are high near zero and near one.

This remark is confirmed by Fig. 1 which shows that the generated values in the intervals [0, 0.1] and [0.9, 1] are more occurrences. Fig. 1 depicts the relative frequency of each generated chaotic value. A total of 10000 chaotic values are shown. 3500 of them are between 0.3 and 0.7 about all the control parameter values of λ between 3.6 and 4.

In addition, the sizes of the two subintervals $[0, 0.3[\cup [0.7, 1[$ and $[0.3, 0.7[$ are respectively $\frac{6}{10}$ and $\frac{4}{10}$ of the entire interval $[0.3, 0.7[$, instead of getting to a uniform distribution, the following probabilities:

- $prob([0.3, 0.7[) = 0.4$,
- $prob([0, 0.3[\cup [0.7, 1[) = 0.6$,

We had:

- $prob([0.3, 0.7[) = 0.35$,
- $prob([0, 0.3[\cup [0.7, 1[) = 0.65$.

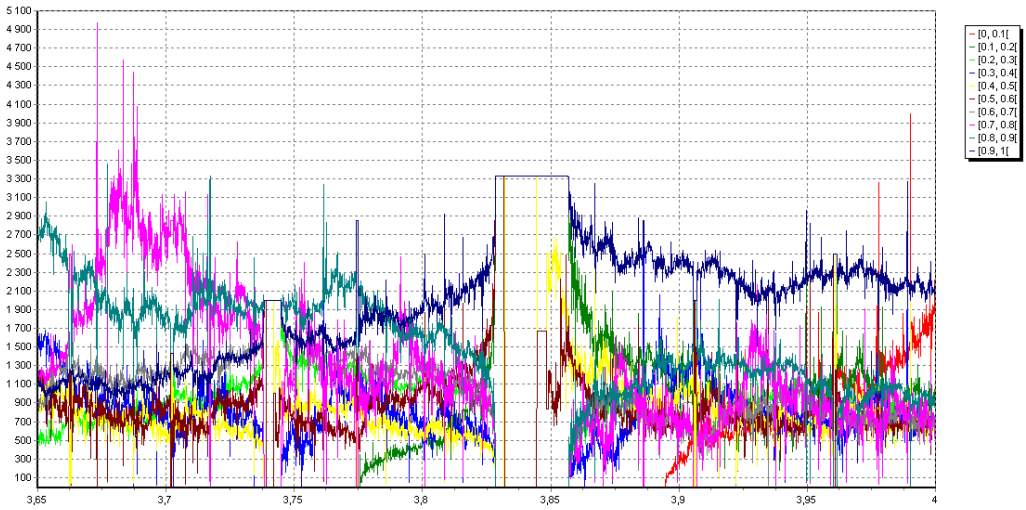

Fig. 1 Frequencies of generated chaotic values

We would like to generate x_1, x_2, ... such that x_i must be independent and follow a U [0,1] law.

x_i should uniformly distributed in the interval [0,1]. Similarly the points (x_i, x_{i+1}) should be approximately uniformly distributed in [0,1] $*$ [0,1].

In this paper, we use the values of x_{i+1} (1) between 0.3 and 0.7 to obtain a uniform distribution (Fig. 2).

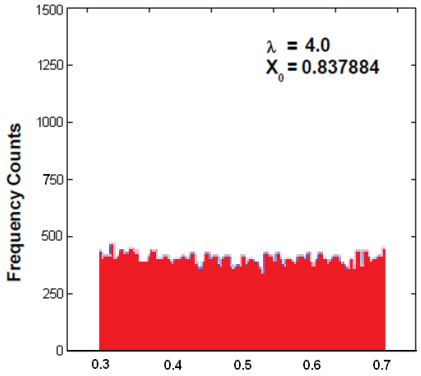

Fig. 2 Uniform distribution in [0.3, 0.7[

In our proposal the distribution is very close to being uniform, the probabilities of each part of the interval [0.3, 0.7[are equally likely [11].

This idea also minimizes the autocorrelation of consecutive chaotic values (x_i and x_{i+1}) and eliminates it for the values near 0 and 1. In [11] we have proved the usefulness of several iterations between taking chaotic values (x_i and x_{i+k}, $k > 0$) to increase the randomness of the generated sequences, especially in the case of encryption of signals (audio, image, video) having multiple redundancies.

IV. Description of the Proposed PRNG

A. Presentation of the PRNG Principle

The principle of our PRNG is based largely on the mix of two logistic maps and two transformation blocs (Fig. 3). For the first function we use a variable control parameter λ_p which is calculated by the second logistic map in which μ_q is his variable control parameter. The initial conditions are x_0 and y_0 respectively, IS_F and IS_G are the Initial States used in transformations **F** and **G** respectively. All those values form together a secret key.

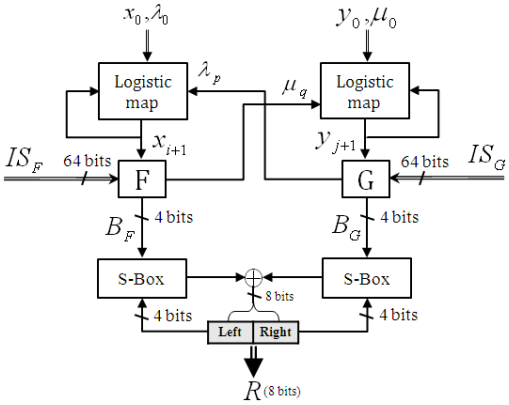

Fig. 3 Block diagram of the PRNG

Since the output of a logistic map is the input of the other, so to execute the steps of the encryption algorithm, we need to select a logistic map which controls the other. We choose as example, the first logistic map (x_0 , λ_0) to control the second (y_0 , μ_0).

After one iteration, our PRNG outputs a binary 8-bits number which is denoted by R. This number it can be represented by two hexadecimal values $Left$and $Right$.

B. The Steps of the Algorithm

Step 1. Initialization Key Values and S-Box Tables

This step is the initialization of our PRNG scheme where we assign random values into two pairs of real values (x_0 , λ_0) and (y_0 , μ_0).

The initial states IS_F and IS_G are used as two binary sequences having uniform distributions. Take for example two binary sequences of IS_F and IS_G such as:

$$\boldsymbol{IS_F} = 1001\ 0100\ 1101\ 0000\ 1010\ 0011\ 0111\ 1011\ 0101\ 0010\ 1100\ 1111\ 0110\ 0001\ 1110\ 1000$$

$$\boldsymbol{IS_G} = 0101\ 1000\ 0011\ 1110\ 0010\ 1010\ 0000\ 0111\ 1111\ 0001\ 0110\ 0100\ 1101\ 1011\ 1001\ 1100$$

Each IS_F and IS_G initial states must have $4 \times 16 = 64$ bits size. Table I contains partial and total sizes of the PRNG key.

TABLE I
PARTIAL AND TOTAL SIZES OF THE PRNG KEY (BITS)

x_0	λ_0	y_0	μ_0	IS_F	IS_G	Total size
48	48	48	48	64	64	320

Step 2. Generation of a Real Value x_{i+1}

To generate the value of x_{i+1}, we use the value of λ_p previously calculated by the transformation G. But in the first time we use λ_0.

The transformation F considers only the generated values x_{i+1} which belongs to the interval [0.3,0.7[. Other values which are not in [0.3, 0.7[will be rejected and the chaotic map will be run for several iterations while the generate value x_{i+1} is not in [0.3, 0.7[.

Step 3. Transformation of the Values x_{i+1}

The function F transforms the values x_{i+1} generated in step 2 into a 4-bit binary words B_F.

Definition: $F : I \longrightarrow U^4, U = \{0,1\}, I \subset [0.3, 0.7[$
$x_{i+1} \longmapsto B_F$

According to the binary sequences of IS_F and IS_G we determine an encoding table which contains binary codes. To perform transformations F and G a binary 4-bits code is associated to each subinterval of $[0.3, 0.7[$. We should subdivide the interval $[0.3, 0.7[$ to 16 parts with the same size and then assigning each subinterval a unique code of 4 bits (Table II).

In this step the algorithm computes the control parameter μ_q used in the second logistic map, i.e.:

$$\mu_q = 3.3 + x_{i+1}, if\ x_{i+1} \in [0.3, 0.7[\quad (2)$$

Giving $3.6 \leq \mu_q < 4$ and the system of the second logistic map (3) follows a behavior chaotic.

$$y_{j+1} = \mu_q \cdot y_j(1 - y_j), y_0 \in]0,1[\ , 3.6 \leq \mu_q < 4 \quad (3)$$

Once an x_{i+1} value is found in the interval $[0.3, 0.7[$ and the control parameter μ_q is calculated (2) by the transformation F, the process of the transformation G is activated to find first an y_{j+1} value in the interval $[0.3, 0.7[$ (3) and then calculate λ_p (4) and B_G on the same manner as B_F, according to a coding table that is determined from the values of the initial state IS_G.

TABLE II
EXAMPLE OF ENCODING TABLES

Intervals	B_F		B_G	
	Bin	Hexa	Bin	Hexa
[0.3, 0.325[	1001	9	0101	5
[0.325, 0.35[	0100	4	1000	8
[0.35, 0.375[	1101	D	0011	3
[0.375, 0.4[	0000	0	1110	E
[0.4, 0.425[	1010	A	0010	2
[0.425, 0.45[	0011	3	1010	A
[0.45, 0.475[	0111	7	0000	0
[0.475, 0.5[	1011	B	0111	7
[0.5, 0.525[	0101	5	1111	F
[0.525, 0.55[	0010	2	0001	1
[0.55, 0.575[	1100	C	0110	6
[0.575, 0.6[	1111	F	0100	4
[0.6, 0.625[	0110	6	1101	D
[0.625, 0.65[	0001	1	1011	B
[0.65, 0.675[	1110	E	1001	9
[0.675, 0.7[	1000	8	1100	C

TABLE III
S-BOX TABLE

$B_H, H = F\ or\ G$	***Left*** if B_F, else ***Right***															
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	F0	4B	B0	09	81	F1	33	E2	6E	0F	C1	49	DB	01	EE	25
1	10	D2	3A	8B	21	DC	50	F2	87	E8	73	ED	55	8A	7C	9B
3	40	94	61	C0	5D	72	D1	47	26	A0	A6	3B	AC	08	4A	EA
3	59	C7	00	DA	7A	BA	07	7D	D6	16	B1	2D	D0	68	BE	36
4	A8	69	32	4E	E0	2F	EB	56	C5	22	63	89	41	F3	17	C6
5	0A	F6	51	70	9A	38	6D	FB	29	B5	34	79	CA	5E	DF	9^E
6	B3	6F	11	BC	20	A5	E9	86	E4	3C	A1	03	F7	37	54	B4
7	23	7B	E1	58	8F	FA	0C	42	99	CD	67	C9	46	8C	AB	18
8	AE	2B	88	BF	3F	62	A7	D7	5C	1B	2A	AA	83	EF	D8	24
9	C4	4F	B9	05	FC	78	02	2C	95	9C	74	E5	B7	0D	5A	FF
A	0B	CB	31	FE	12	C3	6B	4D	B2	FD	35	77	3E	F9	6C	A2
B	E6	52	D3	43	82	DD	90	1C	04	CF	66	93	AF	CC	84	1E
C	96	13	64	98	1D	27	3D	F4	7E	18	44	DE	06	5B	C2	0E
D	9F	45	CE	14	A4	8E	A9	57	1F	91	E3	4C	9D	19	8D	E7
E	7F	D4	5F	71	C8	65	B6	75	19	F5	60	97	BD	48	A3	B8
F	53	F8	6A	EC	39	D9	80	92	A9	15	D5	85	28	76	1A	30

Step 4. Calculation of λ_p

The calculation of λ_p is performed by the transformation G. The input of this function is the value y_{j+1}, generated by the second logistic map which μ_q is its control parameter. Transform G makes a test if the value y_{j+1} is belong to the interval $[0.3, 0.7[$ then calculates λ_p by

$$\lambda_p = 3.3 + y_{j+1},\ y_{j+1} \in [0.3, 0.7[\quad (4)$$

This leads to check that the control parameter λ_p of the first logistic map belongs to the interval $[3.6, 4[$ [12]. Hence the system of (5) follows a chaotic behavior.

$$x_{i+1} = \lambda_p \cdot x_i(1 - x_i), x_0 \in]0,1[\ , 3.6 \leq \lambda_p < 4 \quad (5)$$

Step 5. The S-Box Transformation

The S-Box transformation is a non linear exchange that operates independently on each binary words B_F and B_G. This S-Box (Table III) is presented in hexadecimal form. For example, if $B_F = 7$ and the left side of R is $Left = D$ the result

('8C') is determined by the intersection of the row with index '7' and the column with index 'D'.

V. Simulations and Results

In this section, several experiments are carried out to test the performance and specification of the proposed secure.

A. Performance and Security Analysis

In order to measure the degree of security and analyze the performance of our proposal, some cryptographic tests must be applied such as randomness test and test of independence based on the correlation coefficient.

1. Histogram Analysis

In this step we have generated 43000 numbers (size 8 bits) using our PRNG algorithm. Fig. 4 shows the corresponding histogram.

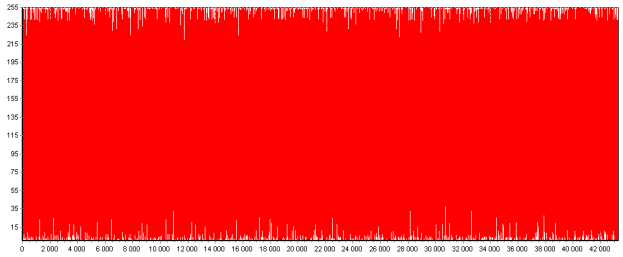

Fig. 4 Histogram of 43 000 numbers

According to the histogram we can consider that the numbers generated by our PRNG algorithm have uniform distribution.

2. Correlation Analysis

We make between generated elements a test of independence based on the calculation of the correlation coefficient:

$$C_r = \frac{COV(X,Y)}{\sqrt{VAR(X).VAR(Y)}} \quad (6)$$

where in X and Y are random variables.

The covariance is determined by the difference in E(XY) and E(X).E(Y). If X and Y were statistically independent then E(XY) would equal E(X).E(Y) and the covariance would be zero.

The covariance of a random variable with itself is equal to its variance.

$$COV(X,X) = E([X - E(X)]^2) = VAR(X) \quad (7)$$

The coefficient C_r varies between -1 and +1. If it is close to 0, there is no linear relationship between X and Y.

On sequences of size N, the covariance is defined as follows:

$$Cov(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}) \quad (8)$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i = E(x)$ is the estimation of mathematical expectations of x.

The estimation covariance between x and y can also be written as follows:

$$Cov(x,y) = E(xy) - E(x)E(y) \quad (9)$$

The correlation coefficient becomes:

$$C_r = \frac{N\sum_{i=1}^{N} x_i y_i - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i}{\left[N\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2\right]^{\frac{1}{2}}\left[N\sum_{i=1}^{N} y_i^2 - (\sum_{i=1}^{N} y_i)^2\right]^{\frac{1}{2}}} \quad (10)$$

The statistical analysis has been performed on any binary sequences generated by the PRNG algorithm. This is shown by a test of the correlation between two successive values in each sequence. We select two sequences of N=20000 bits and we calculate the correlation coefficients, respectively by using (10) replacing y_i by x_j (where x_i and x_j are two values taken successively in the interval $[0.3, 0.7[$.

The correlation coefficients are presented by Table IV:

TABLE IV
Correlation Coefficients of Two Binary Sequences

	First sequence	Second sequence
C_r	0.0225806613	0.0132336147

3. Randomness Tests

To ensure the security of an encryption system the cipher must have some properties such as good distribution, long period and high complexity. The NIST (National Institute of Standards and Technology) designed a variety of different statistical tests based on mathematical theory to test randomness of binary sequences produced by random generator system. So, we used the NIST test suite in order to test the randomness of sequence generated by our pseudo-random number generator.

The results of statistical tests are shown by Table V. Since the computed P-value of each test is > 0.01, then conclude that the output sequence of our PRNG is random. In most tests our PRNG gives high P-values compared with the computed P-value of randomness sequence generated by AES algorithm [13].

TABLE V
Randomness Test Results

Statistical test	AES Algorithm	Proposed Scheme
Random Excursions Variant	0.86318	0.61469
Random Excursions	0.70442	0.70843
Cumulative Sums	0.51903	0.53490
Approximate Entropy	0.48572	0.74287
Serial	0.36901	0.60975
Linear Complexity	0.25783	0.58446
Maurer	0. 13094	0.51048
Overlapping Template Matching	0.70496	0.83369
Non Overlapping Template Matching	0.66205	0.72506
Discrete Fourier Transform	0.31027	0.49780
Binary Matrix Rank	0.30881	0.71588
Longest Run of Ones in a Block	0.73943	0.69402
Runs	0.12778	0.37009
Frequency within a Block	0.50468	0.85619
Frequency	0.75133	0.62805

B. Speed Analysis

We have analyzed the speed of the proposed algorithm on an Intel Pentium Dual-Core @ 2.0 GHz with 2 GB Running on Microsoft Window XP, using Borland Builder C++ compiler.

In this analysis we compare our PRNG algorithm speed with AES algorithm speed, the result is shown by TABLE VI.

TABLE VI
GENERATE SPEED

	Our PRNG	AES (8 bits)
Speed (Mbps)	158,52	110,28

C. Differential Analysis

In this section, in order to study the security of the proposed algorithm against differential attack, we applied very small changes in initial conditions x_0 and y_0.

We have generated two pairs of two 8-bits binary sequences (R, R') and (R'', R''') with different sizes, using respectively the following values : $(x_0 = 0.4170258228, x'_0 = 0.4170258229)$ and $(x''_0 = 0.4170258230, x'''_0 = 0.4170258231)$, keeping $y_0 = y'_0 = y''_0 = y'''_0 = 0.3928502674$.

The sum of absolute difference between a pair of sequences is calculated using the following equation:

$$SAD(R, R') = \frac{1}{N}\sum_{k=1}^{N}\frac{|R - R'|}{2^8}$$

The result of SAD is listed in Table VII. Since the sum of absolute difference is very close to ideal value of $\frac{1}{3}$ then the test result demonstrates sensitivity of the PRNG algorithm to the keys hence it can be concluded that it is highly resistive against differential attack.

TABLE VII
SUM OF ABSOLUTE DIFFERENCE SAD

	SAD	Number of values (8 bits)
(R, R')	0.3287	50 000
(R'', R''')	0.3306	100 000

VI. CONCLUSION

We have proposed a design of a pseudo random number generator (PRNG) based on two chaotic logistic maps starting from independent initial conditions. The pseudo random number is obtained by coding first the outputs of both the chaotic logistic maps and transforming them into 8-bits value using an S-Box table.

The algorithm of proposed PRNG was found to be very fast and secured. This method is very simple and provides random sequences which can be used to encryption data such as message, speech signal, image or video.

The evaluation of performance and security analysis is encouraging and show that the proposed PRNG has perfect cryptographic properties.

REFERENCES

[1] M. François, T. Grosges, D. Barchiesi, R. Erra, "A Novel Pseudo Random Number Generator Based on Two Plasmonic Maps", Applied Mathematics, Vol. 3, pp. 1664-1673, 2012.
[2] A. Lasota and M. C. Mackey, "Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics". Series: Applied Mathematical Sciences, Vol. 97, 1994.
[3] G. Jakimoski and L. Kocarev, "Chaos and Cryptography: Block Encryption Ciphers Based on Chaotic Maps", IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, 48(2), pp. 163-169, 2001.
[4] F. Özkaynak, S. Yavuz, "Security problems for a pseudorandom sequence generator based on the Chen chaotic system". Computer Physics Communications 184:9, pp. 2178-2181, 2013.
[5] L. Feng, G. Xiaoxing, "A New Construction of Pseudorandom Number Generator", journal of Networks, Vol. 9, No 8 , pp. 2176-2182, Aug 2014.
[6] A. Masmoudi, W. Puech, M. S. Bouhlel, "An Efficient PRBG Based on Chaotic Map and Engel Continued Fractions", J. Software Engineering & Applications, Vol. 3, pp. 1141-1147, 2010.
[7] X. Wang, X. Qin, "A new pseudo-random number generator based on CML and chaotic iteration", Nonlinear Dynamics, Vol.70, Issue 2, 1589–1592, 2012.
[8] M. S. Baptista, "Cryptography with chaos", Physics Letters, January 1998.
[9] K. T. Alligood, T. D. Sauer and A. J. Yorke, "Chaos : An introduction to dynamical systems", Springer verlag, New York, 1996.
[10] Ravindra K. Purwar, Priyanka, "An Improved Image Encryption Scheme Using Chaotic Logistic Maps", International Journal of Latest Trends in Engineering and Technology (IJLTET), Vol. 2 Issue 3 May 2013.
[11] M. Hamdi, H. Hermassi, R. Rhouma, S. Belghith, " A new secure and efficient scheme of ADPCM encoder based on chaotic encryption", IEEE conference, 1st International Conference on Advanced Technologies for Signal and Image Processing 978-1, pp. 4799-4889, March 2014.
[12] M. Francois, T. Grosges, D. Barchiesi, R. Erra, "A New Pseudo-Random Number Generator Based on Two Chaotic Maps", INFORMATICA, 2013, Vol. 24, No. 2, pp. 181–197.
[13] A. Jolfaei, A. Mirghadri, , "Image Encryption Using Chaos and Block Cipher", Computer and Information Science, Vol. 4, No. 1, pp 172-185, January 2011.