FPGA Implementation of A Cryptography Technology Using Pseudo Random Number Generator

Hariprasad[1] **NagaDeepa. Ch.**[2]
[1](M.Tech, Department of Electronics and Communication Engineering,VNR-VJIET, Hyderabad, India.)

[2](Assistant Professor, Department of Electronics and Communication Engineering,VNR-VJIET, Hyderabad, India)

ABSTRACT:
In this paper, we discuss some aspects of* linear *and non linear pseudorandom number generators (PRNG's). The outputs of such generators may be used in many cryptographic applications, such as the generation of key material. Generators suitable for use in cryptographic applications may need to meet stronger requirements than for other applications. In particular, their outputs must be unpredictable in the absence of knowledge of the inputs. The implementation of RM-PRNG suitable for cryptography application like encryption and decryption. The simulation results are obtained by using Modelsim and Synthesis is observed by Xilinx ISE 10.1 and also see in FPGA

Key words: linear prng non linear prng, generator, RM-PRNG, cryptography.

I.INTRODUCTION

The choice of the RNG for a specific application depends on the requirements specific to the given application. If the ability to regenerate the random sequence is of crucial significance such as debugging simulations, or the randomness requirements are not very stringent (flying through space on your screen saver), or the hardware generation costs are unjustified, then one should resort to pseudo-random number generators (PRNGs). PRNGs are algorithms implemented on finite-state machines and are capable of generating sequences of numbers which appear random-like from many aspects. Though they are necessarily periodic ("Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin", John von Neumann), their periods are very long, they pass many statistical tests and can be easily implemented with simple and fast software routines. However, when ultimate security is necessary one must turn to the only cipher whichis theoretically unbreakable—one-time pad requires a truly random sequence, and PRNGs are inappropriate for such a purpose. It is also an absolute necessity that cryptographic keys and initialization variables in cryptographic protocols are generated by RNGs. Otherwise, if a PRNG is employed, then security of the cryptographic algorithm and protocol can be no higher than the security of the PRNG.

Pseudo Random Numbers

Obtaining random numbers from a physical source can often be impractical in many applications such as portable web applications. This led to the development of a mathematical method to create a sequence of numbers that could mimic true random numbers. Because a mathematical source is not a true source of random numbers, it is called a pseudo random number. This is due to the mathematical function being completely deterministic and hence, non-random. In the 1940's von Neuman developed the first mathematical algorithm to create random numbers. This was known as the middle-square method, and while it could produce seemingly random number sequences, it quickly proved to be a very poor source of pseudo random numbers. These methods of producing pseudo random numbers are known as pseudo random number generators or PRNG for short.

PRNG must be able to alter its secret state by repeatedly processing input values (seed). PRNG.Prngs are produce long period random number sequence linear prngs are very useful, some of the linear prngs are linear feedback shift registers (lfsrs), linear congruential generators (lcgs), and multiple recursive generators (mrgs).These linear prngs are good in hardware cost and throughput rate. But due to their linear structure output random numbers of these generators are easily predictable. To overcome the predictability problem nonlinear chaos-based prngs (CB-prngs) were proposed, it is efficient in hardware cost, but due to quantization error there exists short periods in such nonlinear prngs. They produce only one bit per iteration hence throughput rate is low. And then to produce long periods and high throughput rate reseeding-mixing PRNG (RM-PRNG) were proposed. The RM-PRNG consists of a CB-PRNG and MRG. The reseeding method removes the

short periods in the CB-PRNG and by mixing MRG with CB-PRNG the overall system period length increases.

In this paper, we propose a new cryptography technology; in this technology has an encryption scheme the message or information is encrypted by using an encryption method, the plain text is converted into unreadable cipher text. An adversary that can see the cipher text should not able to determined anything about the original message. However the cipher text is converted to plain text (original message) by using decryption method. In this encryption and decryption methods are implementing an operation of the plain text and RM-PRNG key with "**Xoring"** operation.

II. RM-PRNG

Reseeding Technique

Reseeding technique is widely used in LFSR for test pattern generation and in CB-PRNG for period extension. Cernák presented a reseeding method either to perturb the state value or the system parameter of digitized logistic map (LGM) for removing the short periods of a CB-PRNG. In 1998, Sang *et al.* Applied a different reseeding method in perturbing a CB-PRNG to extend its period length up to $3.362x10_{29}$ and the lower bound of the reseeded system can be calculated. Li *et al.* discovered that the reseeding technique not only removes the short periods but also improves the statistical properties of CB-PRNG. Recently, these merits of reseeding were confirmed by exhaustive simulation in a 32-b implementation of a CB-PRNG.

Fig. 2 shows the schematic diagram of the RM-PRNG, which is composed of three modules: Nonlinear Module, Reseeding Module, and Vector Mixing Module. In a 32-b implementation, the Nonlinear Module has a controlled 32-b state register and a Next-State construction circuitry. The state register stores the state value (Xt) which is set to seed1 by using the start command. The next state construction is used to produce the next state value (X_{t+1}) by using recursive formula $X_{t+1}=F(X_t)$ [1].

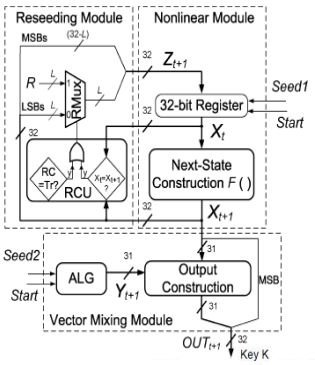

Figure 2. Structure of RM-PRNG

For each generated state value, the reseeding control unit (RCU) in the Reseeding Module compares the values for checking the fixed point condition,and increases the reseeding counter (RC) at the same time. The RC will be reset and the reseeding operation will be activated when either the fixed point condition is detected or the RC reaches the reseeding period. When RC reaches the reseeding period T_r or the fixed point condition is detected then RC will be reset and the reseeding operation will be activated. The sate register will be loaded through the rmux, when reseeding is activated [1]. The value of X_{t+1} is directly loaded into the state register if the reseeding is not activated. Vector Mixing Module is implemented by an auxiliary linear generator (ALG) and output construction. By mixing X_{t+1} with the output Y_{t+1} from ALG in Vector Mixing Module, we obtain the output of the RM-PRNG (32-bit implementation).

A. Nonlinear Module

We use the LGM as the next-state construction function in the Nonlinear Module so that

$$X_{t+1} = F(X_t) = \gamma x_t (1- X_t),\ t \geq 0 \text{..... (1)}$$

Choosing a value 4 for not only makes the LGM Chaotic but also simplifies the implementation of

(1) to merely left-shifting the product by 2 b. However, the state size decreases from 32 to 31 b, because the dynamics (1) are the same. This is equivalent to a degradation of resolution by 1 b. In addition, fixed as well as short periods exist when the LGM is digitized. From exhaustive runs for all of the seeds, we obtain all other periods for the 32-b LGM without reseeding. They are given in Table I with the longest period (18 675) and the set of short listed separately along with their total occurrences. Clearly, the performance of a CB-PRNG using only the Nonlinear Module is unsatisfactory. To solve the fixed points and short-period problem, a Reseeding Module is in order.

B. Reseeding Module

The removal of the fixed points by the reseeding mechanism is obvious. When the fixed point condition is detected or the reseeding period is reached, the value loaded to the state register will be perturbed away from in the RCU by the fixed pattern according to the formula

$$Z_{t+1} = \begin{cases} X_{t+1}[j], & 1 \leq j \leq 32 - L; \\ R[i], & 33 - L \leq j \leq 32,\ i = j + L - 32 \end{cases}$$

Where subscripts ,i ,j are the bit-index, L is integer. In order to minimize the degradation of the statistical properties of chaos dynamics, the magnitude of the perturbation of the fixed pattern should be small compared Here, we set L=5 so that the maximum relative perturbation is only $(2^5-1)/232$ and the degradation can be ignored [15]. Clearly, the effectiveness of removing short-periods depends on the reseeding period as well as the reseeding pattern. However, choosing the optimal reseeding period and the reseeding pattern is nontrivial. Nevertheless, several guidelines to choose a suitable combination had been proposed and discussed in our previous work . First, the reseeding period should avoid being the values or the multiples of the short periods of the unperturbed digitized LGM. Otherwise, if the 5 lsbs of equal to when the reseeding procedure is activated. Then no effective

reseeding will be realized and the system will be trapped in the short-period cycle. Hence, prime numbers should be used as the reseeding period candidates. Although the average period of the reseeded PRNG has increased more than 100 times relative to that of the non reseeded counterpart, the period can in fact be extended tremendously in the Vector Mixing Module described below.

C. Vector Mixing Module

The Vector Mixing Module is constructed by using ALG and output construction. In this module an efficient MRG which is called as DX generator [1], [7] acts as the ALG. By using the following recurrence formula

$$Y_{t+1} = Y_t + B_{DX} . Y_{t-1} \ mod \ M, t \geq 7$$

In output construction unit, to obtain the lsbs of the output The lsbs of Y_{t+1} and that of X_{t+1} are mixed by using XOR operation according to the following equation

$$OUT_{t+1}[1:31] = X_{t+1}[1:31] \oplus Y_{t+1}[1:31]$$

To form the full 32-b output vector out_{t+1} the MSB of X_{t+1} is added to $out_{t+1}[1:31]$.

The DX generatator is implemented described below the implementation [2] of the DX generator is (the ALG) done by using 8-word registers, circular-left-shift (CLS), circular 3-2 counter and End Around Carry- carry look ahead adder (EAC-CLA). By using flip-flops the eight-word register was implemented. For generating two partial products signal Y_{t-7} is circular-left-shifted 28 and 8 b [1], using the modules CLS-28 and CLS-8 respectively. To combine these three 31-b operands into two 31-b operands a circular 3-2 counter is used, which

Consumes 247 gates. To evaluate Y_{t+1} 31-b EAC-CLA is used with 348 gates. The schematic design of the 31-b EAC-CLA [4], [9] is shown in Figure 2(b). The schematic design of the 31-b EAC-CLA includes four modules they are propagation and generation (PG) generators, end-around-carry (EAC) generator, internal carry (IC) generator, and clas [5]. When EAC is generated by group of pgs, EAC is then fed to the IC generator and then to least-significant 8-b CLA. On clas, the final addition was performed.

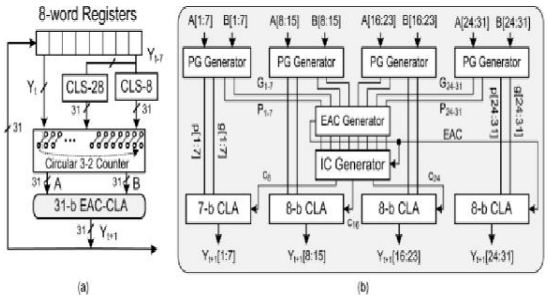

Figure 3. (a) Structure of the DX generator.
(b) Structure of the 31-b EAC-CLA.

III. PROPOSED CRYPTOGRAPHY

Cryptography technlogy is where security engineering meets mathematics. It provides us with the tools that underlie most modern security protocols. It is probably the key enabling technology for protecting distributed systems, yet it is surprisingly hard to do right. Encryption and decryption technlogy is the practice and study of techniques for secure INFORMATION SHARING in the presence of ADVERSARYIES.In cryptography technology; encryption is the process of encoding messages or information in such a way that hackers cannot read it. Encryption and decryption technology is designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary. Encryption is the process of converting plain text or information into unintelligible cipher text. Any adversary that can see the cipher text should not know anything about the original message. Decryption is the reverse, in other words, moving from the unintelligible cipher text back to plaintext.

A.ENCRYPTION

Encryption is the process of translating plain text data (*plaintext*) into something that appears to be random and meaningless (*ciphertext*). To encrypt more than a small amount of data, *symmetric encryption* is used. A *symmetric key* is used during both the encryption and decryption processes. To decrypt a particular piece of ciphertext, the key that was used to encrypt the data must be used. The goal of every encryption algorithm is to make it as difficult as possible to decrypt the generated ciphertext without using the key. If a really good encryption algorithm is used, there is no technique significantly better than methodically trying every possible key. For such an algorithm, the longer the key, the more difficult it is to decrypt a piece of ciphertext without possessing the key.

It is difficult to determine the quality of an encryption algorithm. Algorithms that look promising sometimes turn out to be very easy to break, given the proper attack. When selecting an encryption algorithm, it is a good idea to choose one that has been in use for several years and has successfully resisted all attacks.

B. DECRYPTION

Decryption is the process of taking encoded or encrypted text or other data and converting it back into text that you or the computer are able to read and understand. This term could be used to describe a method of un-encrypting the data manually or with un-encrypting the data using the proper codes or keys.

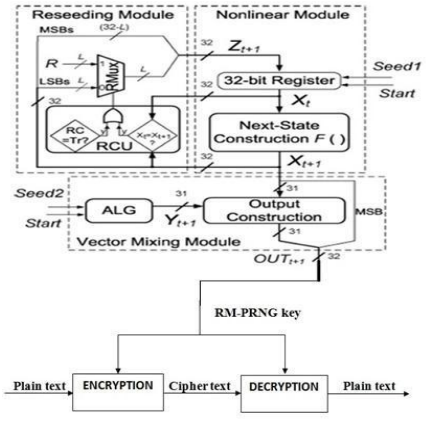

Figure 4. Cryptography technology

IV. RESULTS AND SIMULATION

Pseudo Random Number Generator, Encryption and Decryption were designed using Verilog language in modelsim 6.3. All the simulations are performed using modelsim 6.3 simulator. The simulated output of Pseudo Random Number Generator, Encryption and Decryption are shown in Figure 5&6. Fpga results are shown in fig 7

Figure 5. Simulation results for RM-PRNG key

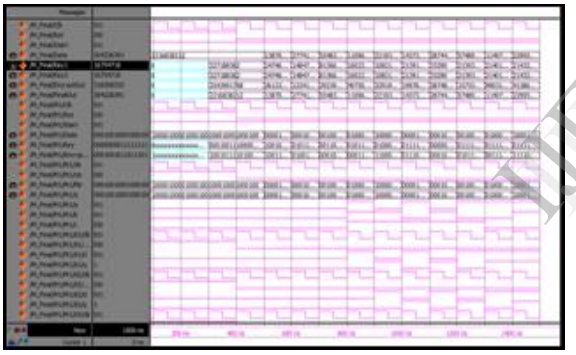

Figure 6. Simulation results for cryptographic method

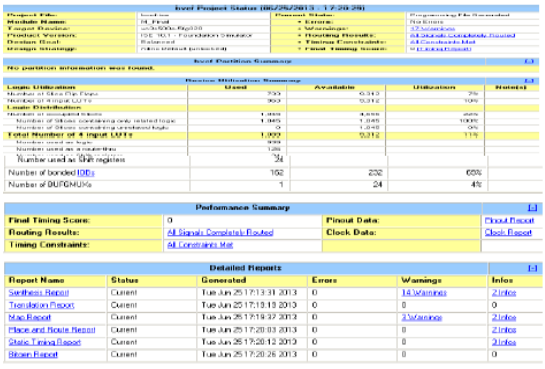

Figure 6. FPGA results for cryptographic method

The design is tested by dumping into Spartan 3E kit and output is analyzed using chipscope pro analyzer. This is used to monitor the output that is continuously changing where we cannot directly see on the kit. The output of chipscope analyzer after dumping on to the kit is shown in figure 7. The RTL schematic is shown in figure 7

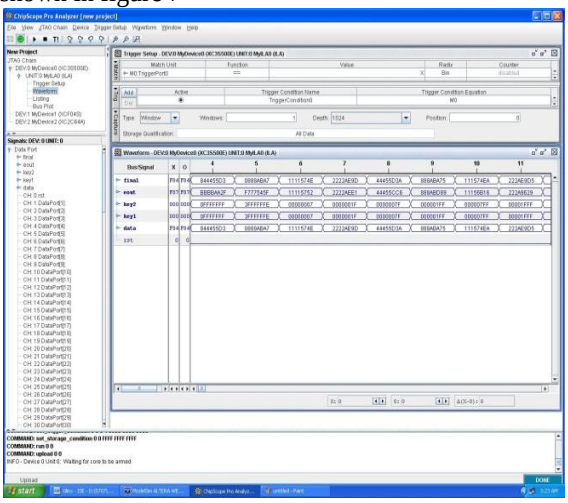

Figure 7. Chip scope analyzer output

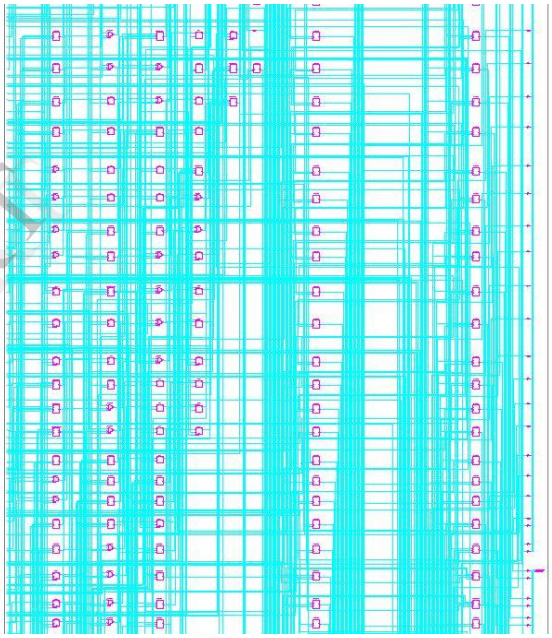

Figure 8: RTL schematic

V. CONCLUSION

We proposed a cryptographic algorithm using RM-PRNG to ensure secure communication. A FPGA implementation of RM-PRNG is to offer long periods and high throughput rate to established statistical standards for PRNGs.

The reseeding mechanism solves the short-period problem originated from the digitization of the chaotic map, while mixing a CB-PRNG with a long-period DX generator extends the period length to the theoretically calculated value greater than . Replacing a hardware-demanding CB-PRNG with a hardware-efficient MRG,.Hence,Simulation and Synthesis is observed by ModelSim 6.4b and Xilinx ISE 10.1 and also fpga implementation.

REFERENCES

[1] Chung-Yi Li, Yuan-Ho Chen, Tsin-Yuan Chang, Lih-Yuan Deng and Kiwing To, "Period Extension and Randomness Enhancement Using High-Throughput Reseeding-Mixing PRNG" transactions on (vlsi) systems, vol. 20, no. 2, february 2

[2] J. Cermak, "Digital generators of chaos," Phys Lett. A, vol. 214, no.3–4, pp. 151–160, 1996.

[3] D. Knuth, *The Art of Computer Programming, 2nd ed. Reading, MA:* Addison-Wesley, 1981.

[4] A. Klapper and M. Goresky, "Feedback shift registers, 2-adic span, and combiners with memory," J. Cryptology, vol. 10, pp. 111–147, 1997.

[5] D. H. Lehmer, "Mathematical methods in large-scale computing units," in Proc. 2nd Symp. Large Scale Digital Comput. Machinery, Cambridge, MA, 1951, pp. 141–146, Harvard Univ. Press.

[6] S. Li, X. Mou, and Y. Cai, "Pseudo-random bit generator based on couple chaotic systems and its application in stream-ciphers cryptography," in Progr. Cryptol.-INDOCRYPT, 2001, vol. 2247, pp. 316–329, Lecture Notes Comput. Sci.

[7] L. Y. Deng and H. Xu, "A system of high-dimensional, efficient, long cycle and portable uniform random number generators," ACM Trans. Model Comput. Simul., vol. 13, no. 4, pp. 299–309, Oct. 1, 2003.

[8] T. Sang, R.Wang, and Y.Yan, "Perturbance-based algorithm to expand cycle length of chaotic key stream," Electron. Lett., vol. 34, no. 9, pp. 873–874, Apr. 1998.

[9] T. Sang, R. Wang, and Y. Yan, "Clock-controlled chaotic keystream generators," Electron. Lett., vol. 34, no. 20, pp. 1932–1934, Oct. 1998

[10] D. Mukhopadhyay,D. R. Chowdhury, and C. Rebeiro, "Theory of composing non-linear machines with predictable cyclic structures," in Proc. 8th Int. Conf. Cellular Autom. Res. Ind., 2008, pp. 210–219, Springer.

[11] J. E. Gentle, *Random Number Generation and Monte Carlo Methods*,2nd ed. New York: Springer-Verlag, 2003.

[12] D. Mukhopadhyay, "Group properties of non-linear cellular automata," J. Cellular Autom., vol. 5, no. 1, pp. 139–155, Oct. 2009.

NagaDeepa.Ch received the B. Tech Degree in electronics &communications in 2008and the M.tech degree in Embedded systems in 2010 from the Jawaharlal Nehru Technological University of Hyderabad, India, where she is currently pursuing the Ph.D. degree. Her main research interests include signal processing and pattern recognition techniques applied to electronic mobile devices. She is currently working in the use of embedded systems as a platform for advanced signal processing for mobile devices..

A.Hari Prasad received the B. Tech Degree in electronics &communications in 2010.Presently studying M.Tech in VNR VJIET eith specialization in VLSI System Design.