IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 9, Issue 2, Ser. I (Mar. - Apr. 2019), PP 14-18
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Design of a Random Number Generator Using VHDL

1Aurodeep Mohanty, 2Dr. Amol Morankar, 3Dr. Mohit Kumar

1PG Scholar, Department of Electronics Engineering, G. H. Raisoni College of Engineering, Nagpur, Maharashtra, India 2Managing Director, Riva Labs (P) Limited, V.N.I.T, Nagpur, Maharashtra, India
3AssistantProfessor, Department of Electronics Engineering, G. H. Raisoni College of Engineering, Nagpur, Maharashtra, India.
Corresponding Author: Aurodeep Mohanty

Abstract: *An array of numbers or symbols whose values are not based on its preceding value is called Random numbers. The number is said to be random if it does not depends on any calculative algorithm or any seed value. This research paper presents a hardware implementation ofa random number generator using VHDL.The Proposed RNG will be a system where strings of unpredictable bits will be present.Field Programmable Gate Arrays (FPGAs) are an increasingly popular choice of platform for the implementation of cryptographic systems. The random numbers generated by our design are verified against the NIST test for statistical correctness.*

Keywords: *RNG (random number generator), NIST (National institute of standard and technology)*

Date of Submission: 02-04-2019 Date of acceptance: 17-04-2019

I. Introduction

A random number generator is an important module in encryption or cryptographic protocol. The cryptography of such systems depend on the assumption that the next values in the sequence cannot be predicted from the observed sequence. Hence RNG used for cryptographic process should be considered as an integral part of the encryption system. Encryption systems are those systems which needs to be highly secure in terms of transmission of data. Hence the random numbers should be unpredictable in nature and also should not depend on any previous data bits. There are many real time applications of random number generators like otp (one time password) generator, captcha generation for web portals, computer simulation, digital gaming and statistical sampling.There are two types of RNGs used for such applications which are TRNG (True random number generator) & PRNG (Pseudo random number generator).

The major difference between both the generators cannot be predicted by studying the number pattern. A number is simply a number, the difference is present in the statistical properties of a stream of numbers. Generally, A PRNG is deterministic and their outputs are periodic, but the period is so large that it is not easy to determine. A true random number generator uses entropy sources that already exist instead of producing them. Entropy here refers to the amount of uncertainty about an outcome. A true random number generator is truly uncertain and unpredictable. It is the source of entropy which makes it unpredictable. The entropy is converted into sequence of numeric bits, there is no internal state kept in the generator and the output is based only on the physical process and not on any previously produced bits.

Pseudo random number generators are algorithms, which produce a longest sequence, this sequence is initialized by a seed value. The numbers are generated from an algorithm which is initialized externally and proved to be random. The seed value determines the next state of random number.In spite of their characteristic properties it is difficult for the researchers to select which generator to use.A lots of phenomena comes in picture while implementing such an encrypted device when the target is to use it for the cryptography application.

With the Transformation of technologies leading to computers and soft wares, researchers have start searching for low cost, high power, large area, and possibly sequence of Random Number Generators (RNGs). This search was started with John von Neumann, who developed a technique based on some computer simulation test operations. He generated numbers by taking out the middle term from the square of the previously generated number and repeated the sequence many a times. This method is called as mid term-square and it terminates in a very short time.Hence, periodicity and deterministic outputs that use an algorithm functions are the main diff erence with the previous generators. They are called as pseudorandom number generators (PRNGs), while devices that use an entropy source to produce randomness are called “true” random number generators (TRNGs).

The implementation part consist of designing the module in software first and then jump towards hardware implementation. Hardware random number generator gives good throughput as well as are good in practical real time operations. One of the major criteria in RNG is to select an encryption algorithm which will make the post processing structure of the module. The post processing unit in RNG plays a major role in statistical improvement of the system as well as improving the randomness. A postprocessor should be designed in a highly complicated manner to get a secure encrypted data. The design algorithm can be a hash function, xor corrector, Von Neumann corrector, or any linear compression technique, depending on throughput and implementation platform.

From a practical application point, it is necessary that RNGs be made using a common cheap silicon process. Moreover, it is highly desirable to design a RNG using purely digital implementation technique because analog circuitry gives an easy hand to the attackers to disturb the encryption and also easy prediction of numbers become possible. Hence, digital circuitry allows for easier integration with digital processors and also makes it easy to design a RNG on popular platforms like FPGAs and CPLDs.FPGAs can be reprogrammed after manufacturing, due to their programmable nature they are an ideal fit for many different markets. Xilinx is the industry leader of such models and provide comprehensive solutions regarding FPGA devices. After implementation of a cryptographic module one should also go through the statistical property check to determine the randomness in the output. There are few tests for such statistical property check like NIST and Die hard test.

The module should pass all the NIST test to undergo implementationin FPGA development kits. In this paper we have presented an implementation of a completely digital random number generator based on FPGA platform.The proposed module consist of two counters as synchronous and asynchronous counter, one D flip flop , Post processing unit and a key generation unit connected parallel to the encryption block. The system is designed for testing purposes and produces an encrypted output of 32 bit.

II. Implementation

The above mentioned blocks are implemented in VHDL codes and the simulation is studied in Xilinx ISE platform. The First step is to design a 8 bit asynchronous counter. The asynchronous counter is responsible for the counting the binary bits and then selecting the higher and lower frequency. These frequency are connected to the D flip flop inputs. The 32 bit synchronous counter is connected to the key generation unit of the system which is used here for producing key logics. The key structure is of 32 bits. The operation undergoes by sub division of the 32 bits in 4 bit each stream into 8 blocks. These eight blocks are xor-ed with each other and make a string of data. Again the string of data undergoes the same xor operation and the cycle continues for four times. More the number of cycles more will be the complexity level of the system. Our main objective is to make a highly complex algorithm which won't be easy to deceive.

Once we get the key logics, now the task is to connected the block with the encryption unit. The encryption unit is 32 bit system made by cryptographic s-box algorithms. The two inputs of 32 bit each are connected to key unit output and synchronous counter output. Hence the post processing block to produce encrypted bits of random numbers.

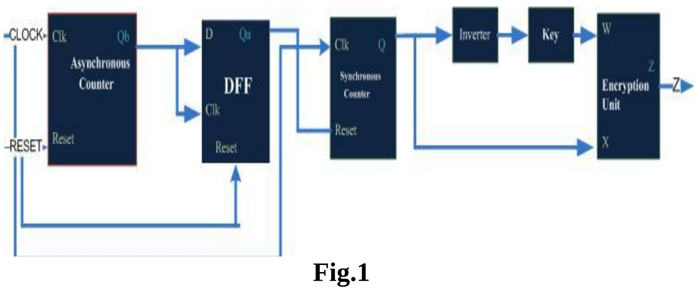

Fig.1

The above figure 1 is the proposed architectural design of the desired random number generator. The above blocks are connected in the same manner as it is shown. The s-box block is the encryption unit. This paper is a contribution to the theory of random number generator based on testing purposes for any kind of digital circuits and digital sampling. After discussing several misconceptions arises regarding the encryption unit. To be more specific, the old traditional method of designing a post processing unit is through by a xor corrector and a Von –Neumann corrector. A designer must keep in mind about the bit date standards and frequency concepts while designing a post processing unit. Other related standards are resilient function, extractor function, cryptographic hash functions. Any data encryption standard or stream cipher can be used to design a key algorithm, provided they should not decrease the bit rate of the system. Here we have introduced a

hummingbird cryptographic algorithm, this algorithm is also based on xoring of the bits but it maintains the bit rate. The counter output is connected to an inverter and the output is directly given to the key generation unit.
A 32 bit key generation unit is hence derived by splitting and xoring the bits with one another. The output analysed shows that the bits are random in nature with a good throughput. This key is then carried away by the s-box algorithm and is dependent on the output after integration of both the blocks. The s-box is here is the encryption part of the system. The input is of 32 bit from synchronous counter, this counter input is further xored with the key output bits, which are also 32 bits. The operation inside the encryption block is carried away again similar like the key generation unit.

The 32 bits are sub-divided and stored in blocks of 4 bit each. And then the 4 bit data undergo xor operation with one another. This process continues for four rounds and then a linear transform algorithm is applied. This transform algorithm is a mathematical expression where the bits are shifted and then added by the original bit. The designer should always keep in mind that whatever the input is the output bit rate should not decrease. This always effect the throughput and delay in random outputs. The final step left here is the integration of all the blocks in ISE platform. Mentioning signals and port mapping the inputs and outputs similar to the block diagram shown in figure 1. The below mentioned RTL view is from 14.7 ISE design suit.

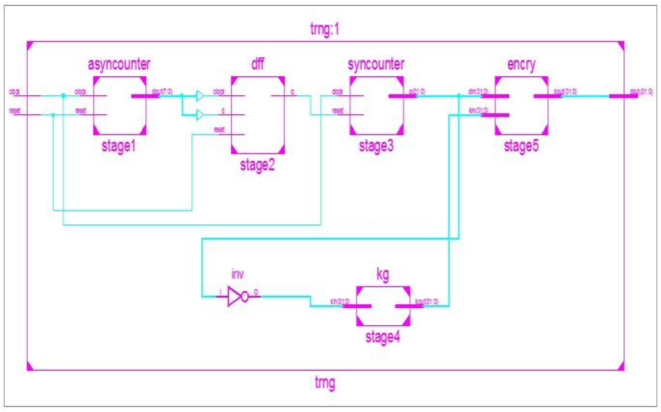

Fig.2

The above figure 2 is the RTL view of the proposed design. The design parameters have been set according to the FPGA development board we are using. Once the system is ready to work, the major step comes in the picture which is the functioning of post processing unit. The post processing unit in a random number generator is the major unit and plays the most important role in the system as is gives a proof of letting the designer know about whether the numbers generated are random in nature or not. And if the numbers are random then it should pass few tests. NIST(national institute of standard and technology)or Die hard test are the tests used to check the randomness of a sequence of numbers. NIST test provides a statistical test suite for random number generator and pseudo random number generator used in cryptographic applications. The generator output must be unpredictable in the absence knowledge of the input. NIST tests are useful in determining whether or not a generator is suitable for a particular cryptographic application. Each test has a separate and unique mathematical algorithm. These procedures are useful in detecting deviations of a binary sequence of randomness. Yet it is obvious that certain failures will definitely occur in random numbers test. It totally depends on the tester to perform the test perfectly and also do the correct interpretation of the test results. This paper will also focus on all the test results and the p-value obtained in each tests. Provided there is a uniqueness in each tests and the way to perform it.

III. Simulations

The below figure 3 and figure 4 are the simulation results of the designed random number generator in Xilinx ISE design suit. Here we can see the variations in signals from all the 32 pins. The next simulation figure shows the unsigned decimal numbers, where it is clear that the following numbers generated are random and unpredictable.

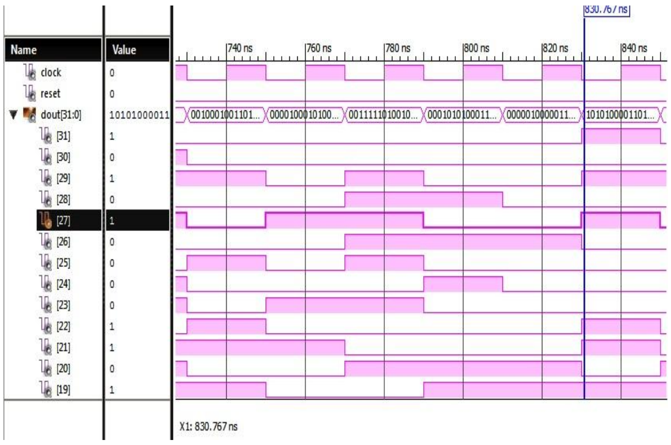

Fig. 3

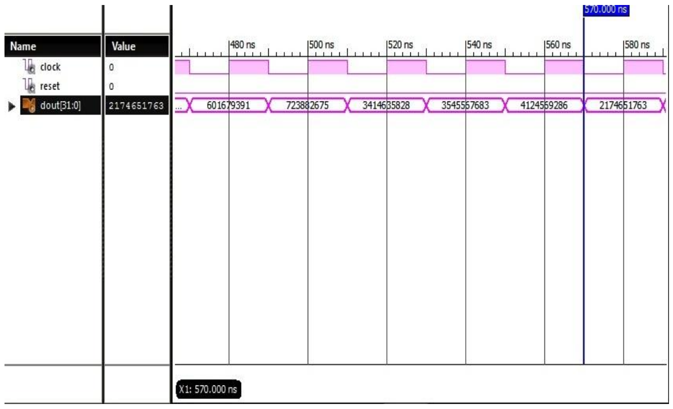

Fig.4

IV. Result Analysis

The above description about the design aspects of random number generator in VHDL is properly mentioned with all the block descriptions and algorithms used. The below mentioned simulation results are from Xilinx ISE design suit. Hence a random number generator of 32 bit working in 404.08 MHz frequency is designed and implemented in Spartan 3e FPGA development board.

V. Conclusion

We have presented a fully digital 32 bit random number generator for testing and cryptographic applications working on FPGA platform. This device has passed all NIST statistical tests and the results have been mentioned in the following table.

Table No.1

NIST TESTS	P-VALUES
Frequency monobit test	0.307
Frequency test within a block	0.0757
The run test	0.543
Test for the longest run of ones in a block	0.3052
The binary matrix rank test	0.9162
The DFT test	0.9484
Non overlapping template matching test	0.8744
Overlapping template matching test	0.419
Universal statistical test	0.3422
Approximate entropy test	0.4472
The cummulative sum test	1
Random excursion test	0.9987

References

[1]. A. Johnson, R. Chakraborty and D. Mukhopadhyay, "A PUF-Enable Secure Architecture for FPGA-Based IOT Applications", *IEEE Transactions on Multi-Scale Computing System*, vol. 1, no. 2, pp. 110-122, 2015.

[2]. A. Rukhin, J. Soto, J. Nechvatal, M. Smid and E. Barker, "A Statistical test suite for random and pseudorandom number generator for cryptographic applications", *Nat. Inst. Standards Technol. (NIST),Gaitherburg, MD,USA,DTIC Document, Tech. Rep*,, 2001.

[3]. N. Tadic, B. Goll and H. Zimmermann, "Laser Diode Current Driver With (1-t/T)-1 Time Dependence in 0.35-μm BiCMOS technology for Quantum Random Number Generators", *IEEE Transactions on Circuits And System-II: Express Briefs*, vol. 64, no. 5, pp. 510-514, 2017.

[4]. M. Yalcin, J. Suykens and J. Vandewalle, "True random Bit Generation From a Double-Scroll Attractor", *IEEE Transactions On Circuits And System_I : Regular Papers*, vol. 51, no. 7, 2004.

[5]. B. Sunar, W. Martin and D. Stinson, "A Provably Secure True random Number Generator with Built-In Tolerance to Active Attacks", *IEEE Transactions On Computers*, vol. 56, no. 1, pp. 109-119, 2007.

[6]. A. Johnson, R. Chakraborty and D. Mukhopadhyay, "An Improved DCM-Based Tunable True Random Number Generator For Xilinx FPGA", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 4, pp. 452-456, 2017.

[7]. D. Liu, Z. Liu, L. Li and X. Zou, "A Low-Cost Low-Power Ring Oscillator-Based Truly Random Number Generator for Encryption on Smart Cards", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 6, pp. 608-612, 2016.

Aurodeep Mohanty. "Design of a Random Number Generator Using VHDL." IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) , vol. 9, no. 2, 2019, pp. 14-18.