International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

PERFORMANCE EVALUATION OF BEAT FREQUENCY-BASED TRUE RANDOM NUMBER GENERATION

[#1]**PARLAPALLI PRABHAKAR**, *Associate Professor,*
[#2]**MEKALA SWAPNA**, *B.Tech Student,*
[#3]**GUNDA HARINI**, *B.Tech Student,*
[#4]**CHOPPARI VIGNESH**, *B.Tech Student,*
[#5]**GANDALA CHARAN PATEL**, *B.Tech Student,*
Department of Electronics and Communication Engineering,
TRINITY COLLEGE OF ENGINEERING AND TECHNOLOGY, PEDDAPALLY, TG.

Abstract: A comprehensive performance analysis of a TRNG system that is based on beat frequency is provided in this paper for secure, high-entropy applications. The proposed method generates unpredictable bit streams by leveraging the intrinsic unpredictability caused by the frequency difference between two distinct oscillators. The system employs phase noise and variations in the surrounding environment to increase entropy without the need for laborious post-processing procedures. The study evaluates statistical variance using well-known test suites such as the NIST and Diehard tests to ensure high-quality output. The entropy rate, stability, power consumption, and throughput are among the critical performance metrics that are pertinent. The experimental data firmly refutes exogenous influences and deterministic patterns.

***Keywords*:** *Beat Frequency, True Random Number Generator (TRNG), Entropy Analysis, Oscillator Phase Noise, Statistical Randomness Testing.*

1. INTRODUCTION

Cryptography uses random numbers. Random number generators offer many ways to generate really random data. These are used for network and PC security. Security fails if random numbers are not completely random. A new strategy is needed. Fair distribution and access to self-care resources are crucial. Therefore, no matter how difficult its design, a flawless RNG must meet all of these conditions. You can build random number generators with FPGAs. Space applications may benefit from the FPGA designs. Many things must be considered before sending something into orbit. To improve reliability, consider equipment lifespan and damage when designing a redundant system.

Problem Description

In addition to cryptographic quality, an enhanced random number generator should function well in extreme conditions like space. Cryptography-grade random numbers are used in cryptography. An FPGA-based design is necessary. We will use an FPGA and another hardware device to produce the noise source, the key generator's main randomness source. During plan execution, the most relevant statistical freedom must be considered. A real RNG is one type. Another category is deterministic random number generators. The physical True Random Number Generator ensures statistical privacy and

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

fair bits. Generators forget their settings. Instead, its power source will add unpredictability. This arrangement will give high throughput in the defined area. A steady data rate is also produced. Quality is assessed by statistical analysis following random series creation.

A fully digital TRNG circuit is presented in this paper. Standard digital logic monitors the small frequency difference between two operationally free DCMs.

2. RELATED WORK

True Random Number Generator Circuits Based on Single- and Multi-Phase Beat Frequency Detection

A completely digital TRNG finds the beat frequency, which is the frequency differential between two ring oscillators. This is done in order to achieve jitter at random frequencies. By merging less important bits that satisfy the NIST randomness standards, a continuous stream of high-entropy random bits can be generated. Using a multiphase system to generate electricity is even more efficient.

A Novel Attack on a FPGA based True Random Number Generator

Numerous cryptographic methods and algorithms rely on TRNGs, or true random number generators. They describe a method for targeting an FPGA-based TRNG while minimizing the output bitstream's randomness. Finally, they demonstrate how to construct and evaluate a "in-field" Hardware Trojan Horse (HTH) insertion method after deployment using the Dynamic Partial Reconfiguration (DPR) element of modern FPGAs. The attack can be started by anyone having a device connected to the FPGA through a network.

A Self-timed Ring Based True Random Number Generator

Since self-timed rings incorporate analog effects into their stage layout, they function in a manner analog to synthesizers. Various events occur in a continuous flow. One distinguishing feature is its ability to swiftly transmit precise multiphase signals. In order to construct a true random number generator, the study demonstrates how events move chaotically over an entropy-rich self-timing ring.

High-Speed True Random Number Generation with Logic Gates Only

We demonstrate that measuring the level of genuine randomness produced by the newly suggested Galois and Fibonacci ring oscillators is achievable by beginning the oscillators from the same point and tracking the rise of the oscillating signals' standard deviations with time. The innovative oscillators are able to attain entropy rates that are orders of magnitude higher than standard ring oscillators, even though they both use the repeating technique.

Dynamic partial reconfiguration in FPGAs

Since they only need to load a subset of the bits in a larger bitstream, certain reconfigurable FPGAs can be configured faster and with less memory than full reconfigurable FPGAs. A fundamental reconfigurable system is shown and the advantages of the most recent dynamic partial reconfiguration design flow are described in this paper.

A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications

This post will teach you how to pick and

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

assess a pseudo-random number generator or random number generator. The output of these generators can be used for a variety of cryptographic procedures, including key production. Additional restrictions on the kinds of generators that can be employed for cryptography are possible. For example, even when fed known inputs, their outputs should surprise. In this paper, we will go over the things you should think about while selecting a generator. Further discussion follows on the topic of statistics and their use to cryptanalysis, followed by a few proposed tests. A generator's suitability for a given security task can be determined by these tests.

A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks

This paper advances the idea of employing oscillator ring phase jitter sampling to create real random number generators. By avoiding some pitfalls and solving other apparently impossible challenges, they arrive at a generic model that, when seriously applied, can generate random bit throughputs in the megabits per second range, making them difficult for bad actors to manipulate. An integral aspect of this is the fill rate, which measures the apparent randomness of the temporal domain analog output signal. We discovered that for a given increase in the number of oscillators, the fill rate is significantly higher. To get around this, they employ a powerful function during the post-processing phase. They enable the author to generate random samples from a signal, but with a little slow fill rate. Consequently, a far smaller set of oscillators is required than in previous systems. Last but not least, they build and use fault-tolerant models to protect their workloads.

Evaluation of Random Number Generators on FPGAs

The current state of security is dependent on real-world random bit generators. Our objective was to determine whether it was possible to generate really random numbers using reconfigurable circuit jitter. According to our findings, the nondeterministic aspects of jitter have the potential to produce really random bits under the right conditions. Two proposals were made to capitalize on this uncertainty by precisely extracting data from uncertain events.

A Design of Reliable True Random Number Generator for Cryptographic Applications

It takes a certain kind of system to reliably produce random binary sequences. There are various alternatives to pseudorandom transformations that can achieve the desired statistical features while avoiding bias and correlation. The proposed method can be better understood with the use of an analytical model that depicts the system's behavior in both its operational and non-operational states. Even when the circuit's properties change, the model demonstrates that the system remains robust. Without relying on output statistical analysis, it is feasible to devise a method to verify that the generator is functioning properly.

3. PROPOSED METHOD

Digitally switched electrical connections jitter. In this manner, crucial components of a digital signal deviate from their correct temporal positions. Noise from the power source, discharge, or even ambient changes can induce jitter. One way to

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

generate random numbers is to employ a jitter source that is sufficiently unpredictable. The two most significant types of noise that might cause unpredictability, in fact, are local Gaussian sources and global predictable sources. Local sources of Gaussian noise generate transistor-level random noise that is unaffected by external disturbances. For example, in ring oscillators and inverters, these factors often lead to widely spaced periods, such as the oscillation period and propagation delay. The central limit theorem states that when independent random variables are added together, their distribution functions resemble normal density functions.

You can see this in Figure 1. When looking to generate randomness, flat-band white noise is the way to go because it is neutral and uncorrelated. Using frequency analysis, we may classify the local random noise into many categories. A common culprit here is thermal noise, which occurs when charge carriers flow erratically across a PN junction or a transistor's collector or drain. Among the many types of noise, one can find flicker noise, sometimes called 1 F noise. Furthermore, it is connected due to its frequency-dependent utility. Due to its far stronger frequency dependence, the 1/F2 Brownian noise is not suitable for use in producing random numbers.

Power supply noise and environmental changes (e.g., temperature, electromagnetic radiation, etc.) are instances of noise that does not occur at random and effects each component of a circuit equally. You shouldn't utilize these noise sources in a TRNG system since they compromise security. As a backdoor for cryptographic assaults, they can be utilized due to their unexpectedness and ease of management. They are difficult to quantify since they might also outstrip neighboring random sources.

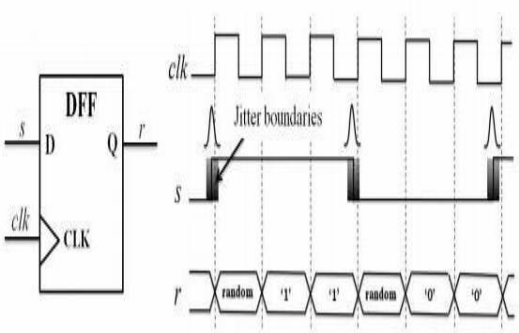

Fig. 2. A strategy for creating a reference oscillator and determining beat frequencies.

An all-digital TRNG, the DCM-BFD-TRNG detects jitter using the Beat Frequency Detection (BFD) technique. It takes little to no additional hardware to include multiphase DCM-BFD-TRNG into an existing single-phase design. The strategy will have better luck if carried out in stages. Using Fig. 1 as a reference, this section provides an overview of the (single phase) DCM-BFD-TRNG's architecture and operation. 3. The circuit contains two DCM units that are identical to one another. Their acronym is DCMA, while their middle name is DCMB. Because of the effects of process variation, one oscillator (DCMA) moves faster than the other (DCMB). The DCM primitives allow for the generation of a wide range of slightly varying frequencies via the manipulation of the values of the Multiplication Factor (M) and the Division Factor (D).

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

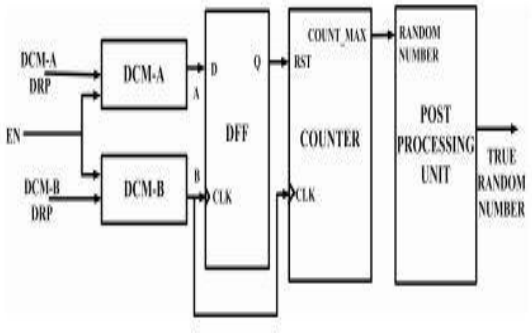

Fig. 3: It is recommended to utilize a programmable digital clock manager based on DCM-BFD-TRNG.

D flip-flops (DFFs) allow for the comparison of the outputs of two DCMs. Assume, for the purpose of argument, that the DFF's clock input and D-input are linked to the outputs of ROSCB and ROSCA, respectively, without limiting ourselves to what is practically possible. Because of jitter in the DCM circuitry, the proposed design is not predictable. Because DCM modules do not require initial calibration, designers have more leeway in selecting clock waveforms [3].

By comparing the two DCMs' frequencies, we can see that the faster oscillator signal will inevitably phase with the slower one at some point. The DPR ports and capabilities dynamically change the DCM parameters. This is essential because otherwise the DFF would generate a logic-1 at odd intervals due to random jitter, leading to collecting events that occur at irregular intervals known as "Beat Frequency Intervals." The designer has more leeway with this function than with the DCM-BFD-TRNG using a ring oscillator.

The DFF moves and resets a counter it monitors at regular intervals that match the beat frequency through its logic-1 output. Every count-up interval ends with the output of a free-running counter reaching a new peak. Because of the unpredictable jitter, this peak point shifts at each interval. When the DFF is powered on, it sends out a clock signal that clears the counter it is driving. At its core, the counter is responsible for increasing the output of the numeric generators. Just before the counter's output hits its maximum value, a sample clock reads it. Among the last three LSBs of the counter, there is a huge discrepancy between the maximum possible count values. To further remove bias from the bits that were created at random, we employ a basic post-processing tool known as the Von Neumann Corrector (VNC). If you want to remove bias from a random bitstream, a popular and cheap method is VNC. We exclude users whose input bit patterns begin with the numerals "00" or "11". Only the first bit of bit patterns that begin with "01" or "10" are utilized; however, the rest are kept for users. uploading the final three least significant bits (LSBs) of the generated random number to the virtual network controller (VNC). Using the VNC results in a little lower output, but better statistical quality.

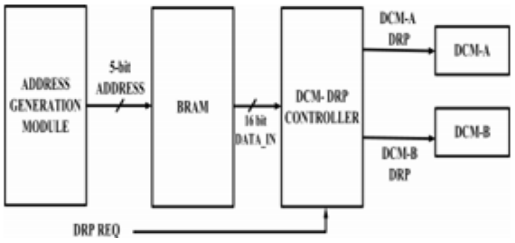

Fig. 4: Putting control circuits to work.

The random bitstream can be obtained by serializing the sampled response. In most cases, up to three LSBs have the right amount of randomness even without correction factors.

Figure 1 depicts the whole configuration of the tuning hardware. 4. The actual

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

values entered for the parameters define the desired clock frequency. A counter's volatility and random numbers are unaffected by M and S to the same extent. This makes it possible to modify the suggested TRNG using the supplied M and D values. Evidence suggests that the circuit is vulnerable to unregulated DPR. Thus, during the design phase, the FPGA stores the safe working value combinations of the D and M parameters in an on-chip Block RAM (BRAM) memory block for each DCM.

4. SIMULTION RESULTS

Table 1. comparison b/w TRNGs

method	area	power	dealy
existing	62%	32.48mw	10.56ns
proposed	28&	14.14mw	4.55ns

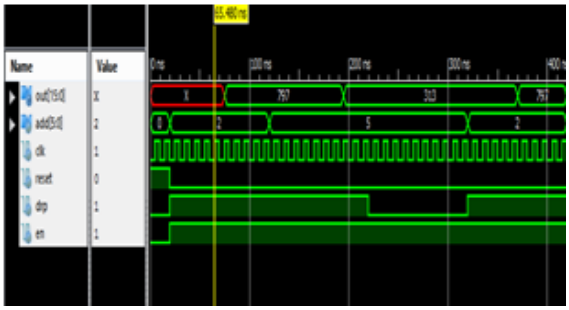

Fig. 5: simulation waveform

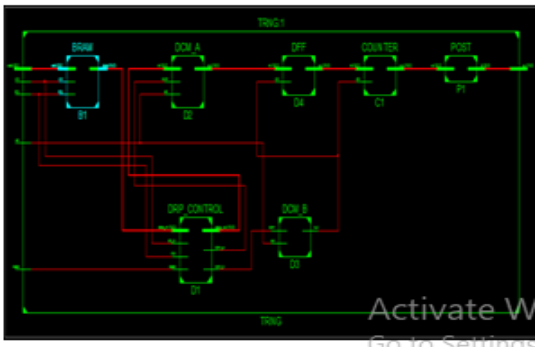

Fig. 6: RTL schematic

Fig. 7: power report

```
--------------------------------------------------------------------------------
|                        On-Chip Power Summary                                 |
--------------------------------------------------------------------------------
|      On-Chip       | Power (mW) |  Used  | Available | Utilization |
--------------------------------------------------------------------------------
| Clocks             |       0.45 |      4 |       --- |         --- |
| Logic              |       0.00 |     27 |      2400 |             |
| Signals            |       0.00 |     56 |       --- |         --- |
| IOs                |       0.00 |     23 |       102 |             |
| Static Power       |      13.69 |        |           |             |
| Total              |      14.14 |        |           |             |
--------------------------------------------------------------------------------
```

Fig. 8: On-Chip Power Summary

```
                                  Gate     Net
Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
 FD:C->Q               3   0.447   0.755  C1/out_7 (C1/out_7)
 LUT2:I0->O            1   0.203   0.579  P1/Mxor_b<8>_xo<0>1 (out_8_
 OBUF:I->O                 2.571          out_8_OBUF (out<8>)
----------------------------------------
Total                      4.555ns (3.221ns logic, 1.334ns route)
                                   (70.7% logic, 29.3% route)
```

Fig. 9: On-Chip delay Summary

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	40	4800	0%
Number of Slice LUTs	36	2400	1%
Number of fully used LUT-FF pairs	30	46	65%
Number of bonded IOBs	23	102	22%
Number of BUFG/BUFGCTRL/BUFHCEs	1	16	6%

Fig. 10: device utilization Summary

5. CONCLUSIONS

Based on its performance evaluation, beat frequency-based True Random Number Generation (TRNG) shows great promise for securely and reliably producing random numbers. To ensure low correlation and high entropy, the system uses independent oscillators to generate beat frequencies that are inherently unpredictable. Statistical research shows

International Journal of Technovation and Business Insights
Volume 02, Issue 01, January-March 2026
ISSN: 3107-7900, P.CODE: 26/2-1/10
DOI: https://doi.org/10.5281/zenodo.19132648

that the generated sequences have a uniform distribution and satisfy standard tests of randomness. This method is able to withstand external influences and changes in the environment. Because its hardware is so basic, it can also be easily integrated into embedded systems and cryptography. Both the power consumption and the implementation costs are low compared to complex TRNG designs. The technique also improves protections against attacks that use reverse engineering or prediction. Scalability and real-time generation further enhance its practical application. In conclusion, TRNG based on beat frequencies offers a reliable and efficient solution for modern security applications.

REFERENCES

1. Bhasini, S., Danger, J., Guilley, S., Ngo, X. T., and Saurvage, L. Hardware Trojan Horses in Cryptographic IP Cores. In Proc. of Workshop on FDTC (2013), pp. 15–29.
2. Cherkaoui, A., Fischer, V., Aubert, A., and Fesquet, L. A Self-Timed Ring Based True Random Number Generator. In Asynchronous Circuits and Systems (ASYNC), 2013 IEEE 19th International Symposium on (May 2013), pp. 99–106.
3. Dichtl, M., and Golic, J. ´ High-Speed True Random Number Generation with Logic Gates Only. In Cryptographic Hardware and Embedded Systems - CHES 2007, P. Paillier and I. Verbauwhede, Eds., vol. 4727 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 45–62.
4. Geary, R. The frequency distribution of the quotinet of two normal variables. J. Roy. Statist. Soc 93 (1930), 442–446. [5] Jin, Y., and Makris, Y. Hardware Trojans in wireless cryptographic ICs. IEEE Design & Test of Computers 27, 1 (2010), 26–35.
5. Johnson, A. P., Chakraborty, R. S., Mukhopadyay, D., and Goren, S. ¨ Fault Attack on AES via Hardware Trojan Insertion by Dynamic Partial Reconfiguration of FPGA over Ethernet. In 9th Workshop on Embedded Systems Security (WESS 2014 (part of ESWEEK 2014)) (October 2014).
6. Lie, W., and Feng-yan, W. Dynamic partial reconfiguration in FPGAs. In 2009 Third International Symposium on Intelligent Information Technology Application (2009), IEEE, pp. 445–448.
7. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., and Barker, E. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Tech. rep., DTIC Document, 2001.
8. Sunar, B., Martin, W., and Stinson, D. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. Computers, IEEE Transactions on 56, 1 (Jan 2007), 109–119.
9. Tang, Q., Kim, B., Lao, Y., Parhi, K., and Kim, C. True Random Number Generator circuits based on single- and multi-phase beat frequency detection. In Custom Integrated Circuits Conference (CICC), 2014 IEEE Proceedings of the (Sept2014), pp. 1–4.
10. Tehranipoor, M., and Koushanfar, F. A Survey of Hardware Trojan Taxonomy and Detection. IEEE Design and Test of Computers 27, 1 (2010), 10–25.