Journals Pub

International Journal of
Telecommunications &
Emerging Technologies

ISSN: 2455-0345
Volume 11, Issue 2, 2025
July–December
DOI (Journal): 10.37628/IJTET

https://journalspub.com/journal/IJTET/

Review IJTET

Building a Secure IoT Ecosystem with TRNGs and VHDL

Kazi Kutubuddin Sayyad Liyakat*

Abstract

Internet of Things (IoT), which links billions of devices and produces enormous volume of data, is growing quickly. Although this interconnection offers previously unheard-of potential, it also poses a serious security risk. Robust cryptographic solutions are necessary to protect sensitive IoT data from malevolent actors, and True Random Number Generator (TRNG) is a dependable source of randomness at the core of these solutions. Despite their efficiency, traditional pseudo-random number generators (PRNGs) are susceptible to assaults because they are deterministic and predictable. Conversely, TRNGs use physical processes to produce genuinely unexpected sequences. The implementation of TRNGs using Very High-Speed Integrated Circuit Hardware Description Language (VHDL) is examined in this article, emphasizing how it might improve security in IoT devices with limited resources. Security is still a top priority as the Internet of Things continues to enter our daily lives. VHDL offers a strong framework for putting TRNGs into practice, which can greatly improve IoT device security. Developers may produce reliable and effective TRNG designs that address the difficulties of the IoT environment by utilizing the flexibility and hardware optimization capabilities of VHDL. Adopting VHDL-based TRNGs is essential for creating a future for the Internet of Things that is more reliable and secure. Future studies should focus on creating more effective post-processing methods, investigating new noise sources, and creating TRNGs that are naturally immune to certain types of attacks.

Keywords: IoT, security, VHDL, true random number generator, malicious

INTRODUCTION

Connecting a huge number of devices to send and receive data, the Internet of Things is a quickly expanding sector. Strong security measures are becoming more necessary as the number of devices is rising. The creation of real random numbers is a crucial component of IoT security since it may be utilized to generate encryption keys, stop replay attacks, and guarantee the system's overall integrity. The development of secure true random number generators (TRNGs) for Internet of Things applications will be examined in this article using VHDL (VHSIC Hardware Description Language) programming [1–3].

***Author for Correspondence**
Kazi Kutubuddin Sayyad Liyakat
E-mail: drkkazi@gmail.com

Professor and Head, Department of Electronics and Telecommunication Engineering, Brahmdevdada Mane Institute of Technology, Solapur, Maharashtra, India

Received Date: October 26, 2025
Accepted Date: October 28, 2025
Published Date: December 31, 2025

Citation: Kazi Kutubuddin Sayyad Liyakat. Building a Secure IoT Ecosystem with TRNGs and VHDL. Journal of Telecommunication and Emerging Technologies. 2025; 11(2): 9–16p.

Electronic system modelling, design, and documentation are the main uses for VHDL, a hardware description language. VHDL's high degree of abstraction makes it possible to create intricate digital systems, which makes it the perfect choice for TRNG design. An entropy source and a randomness extractor are the two primary parts of a TRNG in VHDL [4–6].

The raw random data is produced by the entropy source. It could be based on software-based algorithms or physical phenomena like thermal or

shot noise. Ring oscillators, shift register cascades, and linear feedback shift registers (LFSRs) are common entropy sources for VHDL TRNGs [7–9].

Ring oscillators, for example, take advantage of the inherent variability in the delay times of combinational logic gates. By comparing the frequencies of two or more ring oscillators, it's possible to generate random bit streams.

The randomness extractor refines the raw random data produced by the entropy source. Its role is to eliminate bias, correlations, and other patterns that might compromise the security of the generated random numbers. Extractors often employ various mathematical algorithms like Von Neumann debiasing, Toeplitz matrices, or shrinking generators [10–13].

In VHDL, extractors can be implemented using a combination of shift registers, XOR gates, and counters, depending on the chosen algorithm.

When implementing TRNGs in IoT systems, security must be the top priority. To ensure the highest level of security, several considerations should be considered.

- *Seed Value:* The seed value used to initialize the TRNG should be truly random and unique for each device. This can be achieved by incorporating hardware-based unique identifiers, like physical unclonable functions (PUFs), or using external sources like atmospheric noise.
- *Testability:* Implementing testability features in the VHDL code, like built-in self-test (BIST) routines, can ensure the TRNG is functioning as intended and eliminate potential weaknesses.
- *Entropy Estimation:* Regularly estimating the entropy of the generated random numbers can help assess the strength of the TRNG and take corrective action, if necessary.
- *Tamper Resistance:* Protecting the TRNG against physical and logical attacks is crucial. This can be accomplished by incorporating security features like secure key storage, encryption, and authentication mechanisms.

VHDL programming is a powerful tool for creating secure TRNGs in IoT applications. By carefully designing and implementing entropy sources and randomness extractors, and addressing security considerations, developers can build robust and reliable TRNGs to strengthen IoT security [14].

VHDL FOR TRN'S IN IOT

VHDL offers several advantages for implementing TRNGs in the context of IoT security.

- *Hardware Optimization:* VHDL allows precise hardware control, enabling developers to tailor the TRNG implementation to the specific characteristics of the target chip. This is crucial for optimizing performance and minimizing resource consumption, essential for battery-powered IoT devices.
- *Design Flexibility:* VHDL provides the flexibility to explore different TRNG architectures, allowing for experimentation with various noise sources and post-processing techniques. This adaptability is important for addressing the diverse security requirements of different IoT applications.
- *Security Hardening:* By implementing the TRNG directly in hardware using VHDL, the design is inherently more resistant to software-based attacks. This is particularly important in IoT environments where software vulnerabilities are often exploited.
- *Verification and Testing:* VHDL facilitates rigorous verification and testing of the TRNG's statistical properties, ensuring the quality and reliability of the generated random numbers. This is critical for cryptographic applications where predictability can have severe consequences.

Designing a robust VHDL-based TRNG for IoT security requires careful consideration of several key factors.

- *Noise Source Selection:* Choosing a reliable and unpredictable noise source is paramount. Common noise sources include:

  - *Ring Oscillators*: Exploiting the jitter in ring oscillators to generate randomness.
  - *Metastability*: Utilizing the metastable state of flip-flops to introduce unpredictability.
  - *Thermal Noise*: Amplifying and digitizing thermal noise from resistors.
- *Post-Processing Techniques:* Raw random data generated by the noise source often exhibits biases. Post-processing techniques, like Von Neumann correctors or cryptographic hash functions, are necessary to improve the statistical properties and eliminate correlations.
- *Hardware Implementation:* The VHDL code must be carefully optimized for the target hardware platform. This includes minimizing the use of power-hungry components and maximizing the overall efficiency of the design.
- *Statistical Testing:* Thorough statistical testing is essential to ensure the quality and reliability of the generated random numbers. Standard test suites, like the NIST Statistical Test Suite, should be employed to evaluate the performance of the TRNG.

ROLE OF VHDL IN IOT SECURITY

In the rapidly evolving world of IoT, security has become a paramount concern. With billions of devices connected to the internet, the need for robust and reliable security measures has never been greater. One crucial aspect of IoT security is the use of true random number generators (TRNGs) to generate unpredictable and secure keys and codes. In this context, VHDL programming plays a vital role in implementing TRNGs in IoT devices. This article discusses the role of VHDL programming for TRNGs in IoT security [1, 8–10].

TRNGs are hardware-based devices that generate random numbers by measuring physical phenomena like thermal noise, radioactive decay, or atmospheric noise. TRNGs are essential for generating unpredictable keys and codes for encryption, decryption, and authentication in IoT devices. By using TRNGs, IoT devices can enhance their security and protect against various attacks like brute force, replay, and man-in-the-middle attacks [1–3].

VHDL is a hardware description language used to design and implement digital circuits. VHDL programming is particularly useful for TRNGs because it allows for the creation of custom digital circuits that can generate random numbers with high entropy. VHDL programming can also help in the design of TRNGs that are resistant to various attacks, like timing and power analysis attacks.

VHDL programming for TRNGs involves the creation of entities, architectures, and components that can generate random numbers based on physical phenomena. These entities and architectures can be implemented using Field Programmable Gate Arrays (FPGAs) or Application-Specific Integrated Circuits (ASICs).

In IoT security, VHDL programming for TRNGs can help in several ways, such as

- *Generating Secure Keys and Codes:* By using TRNGs programmed in VHDL, IoT devices can generate secure keys and codes for encryption, decryption, and authentication. These keys and codes can be used to protect against various attacks like brute force and replay attacks.
- *Designing Secure Hardware:* VHDL programming can help in designing secure hardware that is resistant to various attacks like timing and power analysis attacks. By using VHDL programming, designers can create custom digital circuits that implement TRNGs with high entropy and low bias.
- *Implementing TRNGs in FPGAs and ASICs:* VHDL programming is particularly useful for implementing TRNGs in FPGAs and ASICs. By using VHDL programming, designers can create custom digital circuits that are optimized for TRNGs and can be easily integrated into IoT devices.
- *Ensuring Compliance with Standards:* VHDL programming can help ensure compliance with various standards like the National Institute of Standards and Technology (NIST) guidelines for TRNGs. By using VHDL programming, designers can create TRNGs that meet these guidelines and are suitable for use in IoT devices.

VHDL programming plays a vital role in the implementation of true random number generators (TRNGs) in IoT security. By using VHDL programming, designers can create custom digital circuits that generate random numbers with high entropy and low bias. These TRNGs can be used to generate secure keys and codes for encryption, decryption, and authentication, ensuring the security and privacy of IoT devices. Furthermore, VHDL programming can help in designing secure hardware and ensuring compliance with various standards, making it an essential tool for IoT security [2, 6].

SCHEMING VHDL STEPS FOR TRNG

With billions of devices connected and massive volumes of data being generated, IoT is growing quickly. This interconnectedness, however, opens new avenues for security vulnerabilities. Cryptographic algorithms are essential for securing IoT devices and data, but their effectiveness hinges on the quality of the random numbers used for key generation and other security functions. Enter TRNGs, critical components for generating unpredictable and truly random sequences.

This article delves into the design considerations for implementing TRNGs in hardware using VHDL, specifically focusing on the requirements for securing IoT devices.

Unlike Pseudo-Random Number Generators (PRNGs), which rely on deterministic algorithms and an initial seed, TRNGs leverage physical phenomena to generate randomness. This makes them significantly harder to predict and, therefore, more suitable for security applications.

In the context of IoT, TRNGs are vital for,

- *Key Generation:* Securely generating cryptographic keys for encrypting sensitive data transmitted between devices and cloud servers.
- *Secure Communication:* Enabling secure communication protocols, like TLS/SSL, by providing unpredictable challenges and nonces.
- *Authentication:* Generating strong authentication tokens to prevent unauthorized access to devices and networks.
- *Protection Against Replay Attacks:* Preventing attackers from replaying previously valid data to gain unauthorized access.

The foundation of a TRNG is its noise source–the physical phenomena that generates the randomness. Common choices include,

- *Thermal Noise:* Generated by the random motion of electrons in resistors and other electronic components.
- *Jitter:* Variations in the timing of digital signals due to various factors like process variations, temperature fluctuations, and voltage noise.
- *Metastability:* A state in asynchronous circuits where a flip-flop remains in an unstable condition for an unpredictable amount of time.
- *Radioactive Decay:* The random emission of particles from radioactive materials (less common in typical IoT devices due to cost and regulatory constraints).

Once the noise source is chosen, the next step is to design the TRNG using VHDL. Here are critical considerations for robust and secure implementation.

Amplification and Conditioning

- The raw noise signal from the source is often weak and needs amplification. VHDL can be used to design amplifiers with low noise figures to minimize the injection of artificial biases.
- Conditioning circuits are essential to remove any DC offset and filter unwanted frequencies. VHDL allows for the precise design of such filtering stages.

Sampling and Digitization

- The amplified and conditioned noise signal is sampled and converted into a digital signal using Analog-to-Digital Converter (ADC).

- VHDL models can be used to simulate the behavior of ADCs, allowing for the optimization of sampling rates and resolution for optimal randomness extraction.
- Careful design of the sampling clock is critical to prevent any deterministic patterns from influencing the randomness.

Randomness Extraction

- The digitized noise data is often biased and correlated. Random extractors, like XOR operations, Von Neumann correctors, or complex hash functions, are used to mitigate these issues.
- VHDL provides the flexibility to implement a variety of randomness extraction algorithms. The choice depends on the characteristics of the noise source and the desired statistical properties of the output.

Statistical Testing

- Extensive statistical testing is crucial to verify the quality of the random numbers generated. The NIST has developed a statistical test suite (SP 800-22) specifically for random number generators.
- VHDL simulation can be used to generate large datasets of TRNG outputs, which can then be analyzed using the NIST statistical tests. This allows for thorough validation and optimization before deployment.

Security Hardening

- Protecting the TRNG from physical attacks is a major concern. Techniques, like masking, hiding, and shielding, can be implemented at the hardware level to make it more difficult for attackers to probe or manipulate the internal signals.
- VHDL can be used to implement countermeasures against Differential Power Analysis (DPA) and other side-channel attacks.

Integration with IoT System

- The VHDL design should facilitate seamless integration with the overall IoT system. This includes defining appropriate interfaces for accessing the generated random numbers and managing the TRNG's power consumption.

Example: VHDL Snippet for a Simple XOR-Based Randomness Extractor

```
entity xor_extractor is
 Port ( clk : in STD_LOGIC;
 data_in : in STD_LOGIC_VECTOR(7 downto 0);
 data_out : out STD_LOGIC);
end xor_extractor;
architecture Behavioral of xor_extractor is
begin
process (clk)
begin
 if rising_edge(clk) then
 data_out <= data_in(0) xor data_in(1) xor data_in(2) xor data_in(3) xor
 data_in(4) xor data_in(5) xor data_in(6) xor data_in(7);
 end if;
end process;

end Behavioral;
```

Note: This is a highly simplified example and does not represent a fully functional SHA-256 implementation. A real implementation would require significantly more complex logic and would need to adhere to the SHA-256 standard.

This simple example demonstrates a basic XOR-based randomness extractor. The data-in signal represents the digitized noise data, and the data-out signal is the random bit extracted. More sophisticated extractors can be implemented using more complex VHDL code.

The field of TRNG design is constantly evolving. Future trends include:

- *PUF-Based TRNGs:* Leveraging Physical Unclonable Functions (PUFs), which exploit the random variations in manufacturing processes, to create highly secure and tamper-resistant TRNGs.
- *Low-Power TRNGs:* Designing TRNGs with ultra-low power consumption to cater to the constraints of battery-powered IoT devices.
- *Standardization:* Development of standardized interfaces and protocols for TRNGs in IoT devices to improve interoperability and security.

True Random Number Generators are essential components for securing IoT devices. By carefully designing TRNGs using VHDL, leveraging appropriate noise sources, and implementing robust randomness extraction and security hardening techniques, a more secure and trustworthy IoT ecosystem can be built. As the complexity of IoT devices and the sophistication of attacks increase, the importance of high-quality, unpredictable random numbers will only continue to grow. Therefore, a strong understanding of VHDL-based TRNG design is crucial for developers and engineers working in the field of IoT security.

DISCUSSION

IoT has exploded, connecting billions of devices ranging from smart home appliances to critical industrial sensors. However, this interconnectedness comes with a significant challenge: security. Securing IoT devices is paramount, and at the heart of any robust security system lies a dependable source of randomness: TRNG.

While software-based PRNGs are often used, they are deterministic and predictable, making them vulnerable to attack in security-critical applications. TRNGs, on the other hand, leverage physical phenomena, like thermal noise or metastability, to generate truly unpredictable random numbers. This makes them essential for tasks like key generation, cryptographic protocols, and secure communication within the IoT ecosystem.

This article explores the role of VHDL in designing and implementing TRNGs for enhanced IoT security.

VHDL is a powerful hardware description language that allows engineers to model and simulate digital circuits at various levels of abstraction. It offers several advantages for TRNG development.

- *Hardware-Specific Control:* VHDL allows precise control over the hardware implementation of the TRNG, enabling optimization for specific physical characteristics and minimizing vulnerabilities.
- *Accurate Modeling and Simulation:* VHDL facilitates accurate modeling of the chosen physical phenomena and surrounding circuitry. Simulation allows designers to analyze the TRNG's performance, identify potential weaknesses, and optimize its entropy source before committing to hardware fabrication.
- *Portability and Reusability:* VHDL code can be synthesized for different target platforms, including FPGAs and ASICs. This allows for greater design flexibility and reusability across various IoT devices.
- *Security Certifiability:* Properly designed and documented VHDL code is crucial for security certification processes, ensuring that the TRNG meets the stringent requirements for cryptographic applications.

Several architectures for TRNGs can be implemented using VHDL, each leveraging different sources of entropy.

- *Ring Oscillator TRNGs:* These TRNGs utilize the inherent jitter in ring oscillators (circuits composed of an odd number of inverters) to generate random numbers. The frequency variations caused by thermal noise and other physical factors are sampled to create random bits. In VHDL, the ring oscillators can be modeled and simulated, and the sampling logic implemented for efficient entropy extraction. Careful consideration must be given to the number of oscillators, the frequency of operation, and the sampling method to achieve good statistical properties.
- *Metastability-Based TRNGs:* These rely on the unpredictable behavior of metastable states in flip-flops or latches. When forced into a metastable state, the settling time to a stable "0" or "1" is inherently random. VHDL allows the precise modeling of latching circuits and the implementation of logic to detect and process the metastable states to generate random bits. Factors, like power supply noise and temperature variations, need to be considered during the VHDL design and simulation phases.
- *Noise-Based TRNGs:* These exploit thermal noise generated in resistors or transistors. The amplified noise signal is sampled and digitized using ADC. While the analog portion is not directly described in VHDL, the digital post-processing of the ADC output can be implemented efficiently. VHDL can be used to model the ADC, implement bias correction, and perform statistical randomness tests on the generated bit stream.

Designing a secure TRNG in VHDL requires careful attention to several factors:

- *Entropy Source Evaluation:* Thoroughly analyzing and characterizing the chosen entropy source is crucial. VHDL simulation can help understand the characteristics of the physical phenomena and its susceptibility to external influences.
- *Bias Mitigation:* TRNGs can exhibit bias, meaning the generated bits are not equally likely to be "0" or "1". VHDL allows implementing debiasing techniques, like Von Neumann correctors or Exclusive OR (XOR) operations, to improve the statistical quality of the random numbers.
- *Statistical Randomness Testing:* Standard statistical tests, like the NIST SP 800-22 suite, are used to evaluate the randomness of the generated bit streams. VHDL simulation can incorporate these tests to verify the TRNG's performance before hardware implementation.
- *Tamper Resistance:* IoT devices are often deployed in physically insecure environments. Consider designing the TRNG to be tamper-resistant, making it difficult for attackers to manipulate the entropy source or extract the generated random numbers.
- *Side-Channel Resistance:* Protect the TRNG from side-channel attacks like power analysis and timing attacks. Techniques, like masking and hiding, can be implemented in VHDL to obscure the correlation between the generated random numbers and the device's power consumption or timing behavior.

CONCLUSIONS

VHDL-implemented TRNGs – True Random Number Generators – are crucial parts of IoT device security. Developers may establish strong and dependable security solutions by using VHDL's sophisticated platform for creating, simulating, and validating TRNGs. VHDL can be used to create TRNGs that are resistant to a variety of attacks and offer the robust randomness required for secure communication, key generation, and other crucial security operations in the constantly growing IoT landscape by carefully evaluating the entropy source, putting bias mitigation strategies into practice, and adding security measures. VHDL is an essential tool for creating a more secure IoT future since the need for safe and dependable TRNGs will only grow as the number of connected devices rises.

REFERENCES

1. Ravale P. FPGA based finger vein recognition system for personal verification. Int J Eng Res Gen Sci. 2015;3(4).
2. Ravale PM, et al. Retinal image decomposition using variational mode decomposition. Int Res J Eng Technol. 2018;5(6).
3. Khadake S, Kawade S, Moholkar S, Pawar M. A review of 6G technologies and its advantages over 5G technology. In: Pawar PM, et al., editors. Techno-societal 2022. ICATSA 2022. Cham: Springer; 2024. doi: 10.1007/978-3-031-34644-6_107.

4. Patil VJ, Khadake SB, Tamboli DA, Mallad HM, Takpere SM, Sawant VA. Review of AI in power electronics and drive systems. In: 2024 3rd Int Conf Power Electron IoT Appl Renewable Energy Control (PARC). Mathura, India; 2024. p.94-99. doi: 10.1109/PARC59193.2024.10486488.
5. Patil VJ, Khadake SB, Tamboli DA, Mallad HM, Takpere SM, Sawant VA. A comprehensive analysis of artificial intelligence integration in electrical engineering. In: 2024 5th Int Conf Mobile Comput Sustain Informatics (ICMCSI). Lalitpur, Nepal; 2024. p. 484–491. doi: 10.1109/ICMCSI61536.2024.00076.
6. Magar SS, Sugandhi AS, Pawar SH, Khadake SB, Mallad HM. Harnessing wind vibration, a novel approach towards electric energy generation: Review. Int J Adv Res Sci Commun Technol. 2024;4(2):73–82. doi: 10.48175/IJARSCT-19811.
7. Khadake SB, Chounde A, Gopnarayan BB, Patil KB, Kamble SS. Human health care system: A new approach towards life. In: 15th Int Conf Adv Comput Control Telecommun Technol (ACT 2024). 2024;2:5487–5494.
8. Chounde AB, Suryagan AA, Mallad HM, Khadatare MR. AI-driven IoT (AI IoT) based decision-making system for high-blood pressure patient healthcare monitoring. In: 2024 Int Conf Sustain Commun Netw Appl (ICSCNA). Theni, India; 2024. p. 96–102. doi: 10.1109/ICSCNA63714.2024.10863954.
9. Sayyad. AI-powered IoT (AI IoT)-based decision-making system for BP patient's healthcare monitoring: KSK approach for BP patient healthcare monitoring. In: Aouadni S, Aouadni I, editors. Recent Theories and Applications for Multi-Criteria Decision-Making. IGI Global; 2025. p. 205–238. doi: 10.4018/979-8-3693-6502-1.ch008.
10. Sayyad. AI-powered IoT (AI IoT) for decision-making in smart agriculture: KSK approach for smart agriculture. In: Hai-Jew S, editor. Enhancing Automated Decision-Making Through AI. IGI Global; 2025. p. 67–96. doi: 10.4018/979-8-3693-6230-3.ch003.
11. Sayyad. KK approach to increase resilience in Internet of Things: A T-cell security concept. In: Darwish D, Charan K, editors. Analyzing Privacy and Security Difficulties in Social Media: New Challenges and Solutions. IGI Global; 2025. p. 87–120. doi: 10.4018/979-8-3693-9491-5.ch005.
12. Sayyad. KK approach for IoT security: T-cell concept. In: Kumar R, Peng S-L, Elngar A, editors. Deep Learning Innovations for Securing Critical Infrastructures. IGI Global; 2025.
13. Sayyad. Healthcare monitoring system driven by machine learning and Internet of Medical Things (MLIoMT). In: Kumar V, Katina P, Zhao J, editors. Convergence of Internet of Medical Things (IoMT) and Generative AI. IGI Global; 2025. p. 385–416. doi: 10.4018/979-8-3693-6180-1.ch016.
14. Shinde SS, Nerkar PM, Kazi SS, Kazi VS. Machine learning for brand protection: A review of a proactive defense mechanism. In: Khan M, Amin Ul Haq M, editors. Avoiding Ad Fraud and Supporting Brand Safety: Programmatic Advertising Solutions. IGI Global; 2025. p. 175–220. doi: 10.4018/979-8-3693-7041-4.ch007.

