

# Enhanced ARIA-based counter mode deterministic random bit generator random number generator implemented in verilog

Eugene Rhee<sup>1</sup>, Jihoon Lee<sup>2</sup>

<sup>1</sup>Department of Electronic Engineering, College of Engineering, Sangmyung University, Cheonan, Republic of Korea

<sup>2</sup>Department of Smart Information and Telecommunication Engineering, College of Engineering, Sangmyung University, Cheonan, Republic of Korea

## Article Info

### Article history:

Received Nov 21, 2023

Revised Jan 4, 2024

Accepted Jan 11, 2024

### Keywords:

Algorithm

Cipher

CTR

DRBG

Random number

## ABSTRACT

This paper presents a study aimed at effectively implementing a deterministic random bit generator (DRBG) IP in verilog language, based on the standard encryption algorithm. By controlling the existing round generation and key generation blocks, the internal modules of the counter mode deterministic random bit generator (CTR-DRBG) were successfully implemented and operated, ensuring the secure and efficient generation of random bit sequences. The research focused on parallel operation of modules and optimized module placement to achieve improved clock frequencies. By concurrently operating two modules in the derivation and internal update modules of CTR-DRBG, the processing speed was enhanced compared to the conventional algorithm. Additionally, integrating the reseeding and initialization modules of CTR-DRBG into a single module successfully reduced size. Furthermore, this IP supports the special function register (SFR) interface. The safety of the CTR-DRBG was validated through known answer test (KAT) verification utilizing test vectors from certification. Future research should explore additional studies on CTR-DRBG operating on real FPGA or ASIC, not only using normal algorithm but also employing other block cipher algorithms.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



## Corresponding Author:

Jihoon Lee

Department of Smart Information and Telecommunication Engineering, College of Engineering

Sangmyung University

Cheonan, Republic of Korea

Email: vincent@smu.ac.kr

## 1. INTRODUCTION

The creation of secret keys and encryption protocols in cryptographic algorithms necessitates unpredictable and secure random numbers. These random values are instrumental in various information security systems and cryptographic products, bolstering the safety and reliability of domestic communication networks. The generation of secure random numbers plays a pivotal role as a fundamental element in cryptographic applications [1]–[3], crucially contributing to the generation of cryptographic keys, initialization vectors, nonces, signature keys, certificates, and more. counter mode deterministic random bit generator (CTR-DRBG) is designed to meet these requirements [4]–[6]. It operates as a random number generator based on the counter mode encryption technique, generating secure random numbers. 'Counter Mode' is a block encryption method that encrypts consecutive counter values to create a block sequence similar to randomness [7]–[9]. Leveraging this characteristic, CTR-DRBG combines a secure initial vector and counter to produce distinct, unpredictable random numbers on each occasion.

The use of CTR-DRBG provides reliability in generating unpredictable random numbers, enabling secure generation of encryption keys and safeguarding communication data. Furthermore, CTR-DRBG contributes to enhancing communication confidentiality and integrity by providing reliable random numbers for various security protocols. CTR-DRBG is designed in accordance with the standards recommended by National Institute of Standards and Technology (NIST), ensuring its safety through these standard recommendations, which guarantee the safety of information protection systems on an international level. Moreover, CTR-DRBG finds application across various environments. In scenarios requiring secure random numbers, such as secure communication [10]–[12], authentication processes [13]–[15], digital signature generation [16]–[18], virtual private network (VPN) connections [19]–[21], and security protocols [22]–[24], CTR-DRBG ensures high reliability and safety. Consequently, communication networks can attain higher levels of safety and trustworthiness. In this manner, CTR-DRBG plays a crucial role in enhancing the safety and reliability of domestic communication networks through secure random number generation.

In today's information security landscape, hardware-based random number generators play a crucial role [25]–[27]. These generators are recognized as essential elements for producing secure random numbers in cryptographic applications like integrity verification, digital signatures, data encryption, and various security systems. Therefore, secure and efficient random number generation is considered a central requirement in security technology. This research aims to efficiently design a CTR-DRBG internet protocol (IP) based on the algorithm using verilog language to generate secure and integrity-assured random numbers in hardware environments. Additionally, it seeks to increase clock frequencies through parallel processing of cryptographic modules and appropriate arrangement, while reducing memory usage by designing reusable modules for each function of the existing CTR-DRBG. This approach focuses on enhancing the performance of hardware-based random number generators while simultaneously optimizing resource utilization to minimize costs and memory footprint. Through this research, it is anticipated to enhance the safety and reliability of communication networks while contributing to improving the performance of security systems and cryptographic products.

## 2. CTR-DRBG

### 2.1. Background

There are two main methods of generating random numbers. The first relies on physically unpredictable procedures to generate all the bits composing the random number, while the second method generates random numbers from a given input using deterministic algorithms. The first method is known as "non-deterministic random bit generator (NRBG) [28]," and the second is referred to as "deterministic random bit generator (DRBG)". Typically, the first method is utilized in creating actual random numbers, known as "true random number generators (TRNG) [29]–[31]." The second method is known as "pseudo-random number generator (PRNG) [32]–[34]". Figure 1 illustrates a block diagram of the deterministic random number generator. As depicted in Figure 1, a deterministic random number generator produces the same random number output for the same input due to a predetermined algorithm. This mechanism governed by the specified algorithm is termed the deterministic random number generation mechanism (DRBG mechanism), operating based on a deterministic algorithm that generates bit strings from an initial value called a seed. This seed is determined by values obtained from an entropy source. While the bit string generated by the DRBG mechanism can be predicted and reproduced if the input information, including the algorithm used and seed, is known, properly managed inputs and well-designed algorithms render the output bit string of the DRBG mechanism indistinguishable from actual random numbers. The CTR-DRBG, the primary focus of this study, operates as one type of such deterministic random number generators, functioning as a function based on block ciphers. CTR-DRBG is used in various information security systems and cryptographic products by combining with an entropy source to ensure secure random number generation [35]–[37].

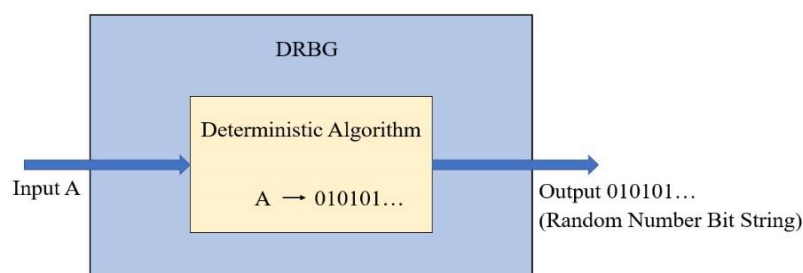


Figure 1. Deterministic random number generator block diagram

## 2.2. Problems

The CTR-DRBG algorithm, widely used for secure random number generation, faces several inherent issues [38]–[40]. Firstly, in the derivation function, generating secure random numbers using the counter value leads to increased computational time as the counter starts at 0 and increments, causing a significant rise in computation time with larger counter values [41]. Particularly, the necessity to reset upon each counter value change consumes time during operations. Secondly, although the reseeding and initialization functions in CTR-DRBG share nearly identical structures, separate implementations are required due to different input values. This redundancy induces unnecessary duplicate structures, potentially increasing chip size in hardware implementations [42]. Thirdly, while CTR-DRBG generates a key stream to provide randomness, the use of the counter value increases predictability in the key stream [43]. If specific Counter values become predictable, it's possible to predict the corresponding key stream values, ultimately compromising the security of the generated random numbers.

## 2.3. Algorithm

Figure 2 depicts the flow chart of the CTR-DRBG algorithm. CTR-DRBG operates as a deterministic random number generator aiming to safeguard and ensure the integrity of the inputs to the output-generation function, which embeds a deterministic algorithm to protect the entropy characteristics from external attacks. In this context, the input to the output-generation function is referred to as the 'seed,' composed of values known as key and value. These elements—seed, key, and value—are all considered the internal state of the random number generator. As shown in Figure 2, the random number generator initiates by determining the initial internal state, subsequently updating it, and then utilizing the updated internal state to generate output (random numbers). The function responsible for determining the initial value of the internal state is termed the 'instance creation (initialization) function,' while the function updating the internal state is denoted as the 'internal state update function.' The function creating the output is known as the 'output generation function.'

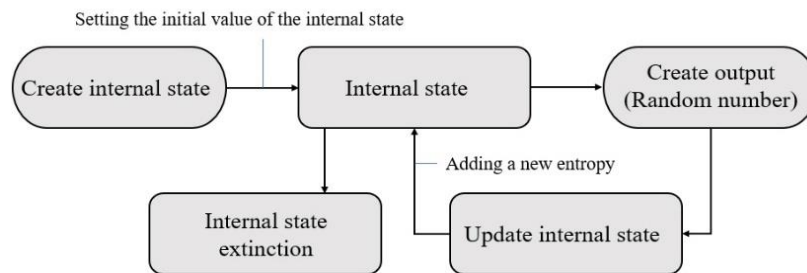


Figure 2. CTR-DRBG algorithm flow chart

## 3. IMPLEMENTATION

### 3.1. Internal state update module

Figure 3 illustrates the block diagram of the internal update module. The internal update function (IUF) receives input data along with key and value and is responsible for updating the key and value. The IUF module, serving as the internal state update function, consists of the CTR operating system and a CTR result that processes input data through XOR operations with the key and value [44], [45]. The encrypted block, cipher<sub>i</sub>, is generated by encrypting the value obtained by adding counter to key and value. As the counter increments, these cipher blocks concatenate sequentially to produce the CTR result block. This CTR result block undergoes updating by XOR operations with the input<sub>data</sub>. Consequently, the left block represents the key, while the right block signifies the value after the update.

As shown in Table 1, the internal update module operates through the following input/output ports: it receives the values to update, value and key, via *i\_value* and *i\_key*, respectively, each in blocks of 128 bits. Subsequently, the updated output data is delivered through *o\_iuf\_data* in segments of 256 bits. In this context, the updated output data, *o\_iuf\_data*, presents the concatenated form of the updated key and value. The finite state machine (FSM) of the internal update module is as shown in Figure 4. It consists of a total of four states, each of which is described in detail in Table 2.

When *i\_iuf\_en* is activated, the transition occurs from the idle state, IUF\_IDLE, to the key expansion stage, represented by IUF\_KS. During the key expansion phase, the KEY\_SCHED module becomes active, generating 12 round keys as part of the key expansion process. Once the key expansion process concludes and *w\_key\_expand* is set, the transition proceeds to the next state, IUF\_ENC.

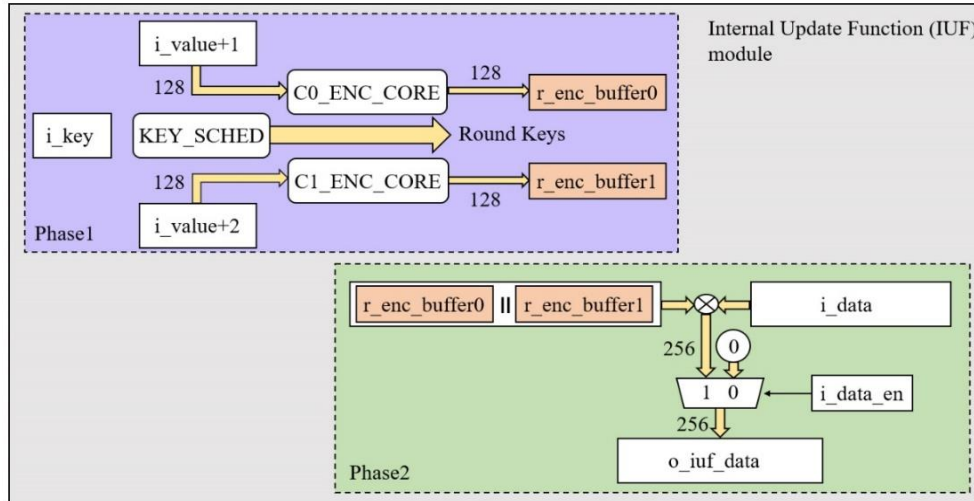


Figure 3. IUF block diagram

Table 1. IUF input/output port

Port	Width (bit)	Description
i_clk		Input clock signal function
i_rstn		Input reset signal (active low)
i_value	128	Value to update
i_key	128	Key to update
i_iuf_en	1	Signal to activate internal update
o_iuf_data	256	Updated output data
o_iuf_done		Signal for completion of internal update function

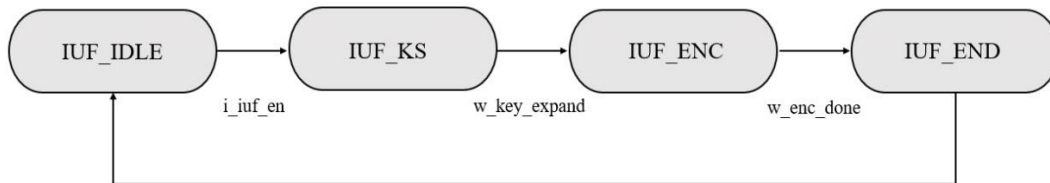


Figure 4. IUF FSM

Table 2. States of IUF FSM

State	Description	Conditions for transitioning to the next state
IUF_IDLE	Idle state	i_iuf_en == 1
IUF_KS	Key expansion phase	Upon completion of key expansion
IUF_ENC	Encryption process	Upon completion of encryption
IUF_END	Termination phase and Updated data output phase	Automatic transition

In Figure 3, there is a block diagram corresponding to phase 1 within the IUF module. During the encryption phase, i\_value+1 and i\_value+2 were each fed into different encryption modules, C0\_ENC\_CORE and C1\_ENC\_CORE. These inputs underwent encryption in parallel using previously expanded keys. Additionally, the resulting encryption from both modules is stored in separate buffers, r\_enc\_buffer0 and r\_enc\_buffer0, at the next clock cycle. Once encryption concludes in both encryption modules and the termination signal, w\_arai\_c0\_done, is set, the transition to the next state, IUF\_END, occurs.

There is the block diagram for phase 2 within the IUF module also. In IUF\_END, phase 2 is executed, where the two encryption results obtained in phase 1 are stored in r\_enc\_buffer0 and r\_enc\_buffer1. The values from these two buffers are sequentially retrieved, then undergo XOR operations with the input data, i\_data, of the IUF. As a result, the updated internal state, o\_iuf\_data, is produced. Simultaneously, upon output

generation, `o_iuf_done` is set to conclude the internal update. Consequently, the state automatically transitions to the idle state, `IUF_IDLE`.

### 3.2. Initialization and re-initialization module

This paper proposes a method to integrate the initialization and re-initialization functions in the CTR-DRBG algorithm, identifying their structural similarities. These similarities stem from the considerable resemblance in functionality and operations performed by both functions. Firstly, the initialization and re-initialization functions exhibit a similar structure in how they generate random bits and update the internal state [46]–[48]. Both employ counter mode, repeating block ciphers multiple times, and combine them to generate random bits while updating the internal state through XOR operations [49], [50]. Secondly, they share similarities in their input and output formats. Both functions receive initialization data as input and return generated random bits or key streams as output. Such analogous input-output formats make them ideal candidates for integration into a single module. Leveraging this structural similarity allows for the integration of initialization and re-initialization functions while maintaining code simplicity and consistency without compromising security. Moreover, the integrated module enables smooth transitions between initialization and re-initialization stages, facilitating more efficient system operations.

The IF module serves as both an initialization module and a re-initialization function. The IF module encompasses a derivative function (DF) and an IUF internally, with its operational mode dependent on the use of the DF. When utilizing the DF, the IF module receives inputs such as entropy, nonce, and a personalization string (PS). Nonce and PS are optional components. The input for the initialization function is composed of 'Entropy || Nonce || PS,' and for the ARIA-128 algorithm, this input's length must exceed the output length of the DF, i.e., at least 256 bits. Irrespective of the use of nonce and PS, the updated data output from the DF serves as the input data (`i_data`) for the IUF function. In cases where the DF is not employed, the input for the initialization function consists of 'Entropy || PS.' The PS is an optional element, and the length of Entropy matches that of the output of the IUF function. For the ARIA-128 algorithm, the length is 256 bits. Regardless of the utilization of PS, its length aligns with the output length of the IUF function, either equivalent or zero bits. When using PS for ARIA-128, it is 256 bits, and in the absence of PS, it's 0 bits. The PS value after the XOR operation with Entropy is utilized as the input data for the IUF function". `i_init_en` (initialization enable signal) acts as an initial activation trigger for the system to commence a process. When activated, it sends a signal to initiate the system's initialization procedure. `i_df_en` (derivative function enable signal) determines whether the DF should be activated. When activated, it indicates that DF operations are enabled and should be utilized as part of data processing. `i_data` (32-bit input data) accommodates a 32-bit input data stream, serving as raw data input for the system. The data provided through this port is processed as part of the system's operations. `i_data_en` (input data validity signal) verifies the integrity of the input data stream. Activation signifies that the data present at the `i_data` port is valid and ready for processing. `i_Elen` (31-bit entropy length) and `i_PSlen` (31-bit PS length) specify the lengths of entropy and the personalization string in bits. Entropy signifies the unpredictability of the data, while the personalization string adds user-defined customization to the process. These lengths provide essential information for the system to handle data accurately. `i_N` (32-bit output data length in bytes) determines the byte-based length of the output data generated by the system. This specifies the output size of the data created by the system, ensuring consistency for external components or interfaces. These input ports facilitate the system in processing input data, customizing processing using entropy and personalization strings, activating specific functionalities like initialization and derivation, and generating output data of specified lengths, thus facilitating intended system operations and functionalities.

## 4. RESULT

Figure 5 shows the block diagram of the galois field (GF) module, functioning as the output bit sequence generator. It incorporates the internal update module, IUF, and the ARIA-CTR module while receiving additional input alongside key and value parameters. The IUF module ensures protection by updating the internal state once more before the final output, guaranteeing resilience even if the internal state is compromised due to an attack. Simultaneously, within the CTR module, leveraging the internal state, it proceeds with ARIA-CTR mode to generate 256 bits of random output. To expedite processing, this research employs two CTR modules in parallel, subsequently denoted as the `TWIN_CTR` configuration.

The FSM consists of four primary states. The initial state, `IDLE`, signifies the system being in an idle state. When the `i_gf_en` input is activated to trigger the output generation function, the system transitions to the `PRE_CTR` stage. During `PRE_CTR`, preparatory tasks for output creation in CTR mode are performed. If additional input exists and a prior re-initialization has not been executed, the `i_pre_ctr_en` input is activated, transitioning from `IDLE` to `PRE_CTR`. In the `PRE_CTR` stage, Key and V undergo an additional update via the IUF invocation. Upon completion of this update, the system moves to the `CTR_RUN` stage upon receiving

the activated `w_iuf_done` signal from the IUF module. Within the `CTR_RUN` stage, utilizing `Key`, `V`, and output length (`len_output`) as inputs, the system generates an output bit sequence using CTR mode. Throughout this process, the `TWIN_CTR` module drives the CTR progress, producing the generated random bit sequence as output. Furthermore, `V` gets updated to `V+len_output`. Upon completion of the `TWIN_CTR` module, the system transitions to the `IUF_RUN` stage upon receiving the activated `w_ctr_done` signal. Finally, within the `IUF_RUN` stage, by receiving additional input data (`AD`), `Key`, and the updated `V`, the system recalls the IUF to perform another key and `V` update.

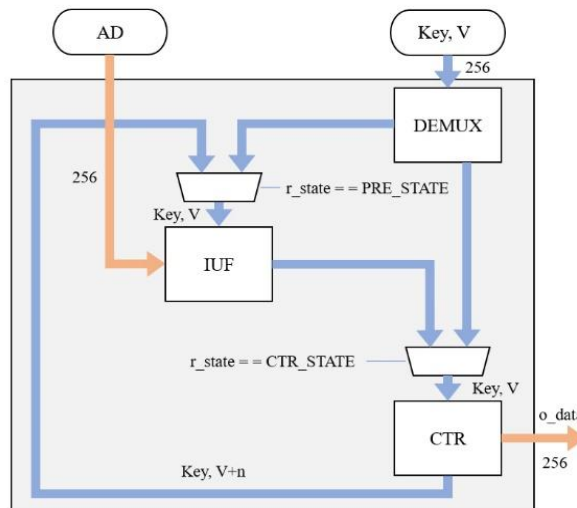


Figure 5. Output generation function GF module block diagram

## 5. CONCLUSION

An IP that implements CTR-DRBG in verilog, providing a secure and efficient random number generation capability, is successfully developed. The validation results, conducted using test vectors, were all confirmed as expected. The overall clock cycle observed in the experiment was measured at 3.85 ms. In this study, implementing CTR-DRBG in verilog has allowed the creation of a secure and integrity-assured random number generation system. This system provides secure randomness for cryptographic applications and information security systems.

## ACKNOWLEDGEMENTS

This research was funded by a 2023 research Grant from Sangmyung University (2023-A000-0082).

## REFERENCES




- [1] Y. Oohama, "Performance analysis of the interval algorithm for random number generation based on number systems," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1177–1185, Mar. 2011, doi: 10.1109/TIT.2010.2103730.
- [2] P.-H. Tseng, M.-H. Lee, Y.-H. Lin, H.-L. Lung, K.-C. Wang, and C.-Y. Lu, "ReRAM-based pseudo-true random number generator with high throughput and unpredictability characteristics," *IEEE Transactions on Electron Devices*, vol. 68, no. 4, pp. 1593–1597, Apr. 2021, doi: 10.1109/TED.2021.3057028.
- [3] E. Bach, "Efficient prediction of Marsaglia-Zaman random number generators," *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 1253–1257, May 1998, doi: 10.1109/18.669305.
- [4] N. Hayati, K. Ramli, S. Windarta, and M. Suryanegara, "A novel secure root key updating scheme for LoRaWANs based on CTR\_AES DRBG 128," *IEEE Access*, vol. 10, pp. 18807–18819, 2022, doi: 10.1109/ACCESS.2022.3150281.
- [5] S. Cohny *et al.*, "Pseudorandom black swans: Cache attacks on CTR\_DRBG," in *2020 IEEE Symposium on Security and Privacy (SP)*, May 2020, pp. 1241–1258, doi: 10.1109/SP40000.2020.00046.
- [6] L. Guan, J. Lin, Z. Ma, B. Luo, L. Xia, and J. Jing, "Copker: A cryptographic engine against cold-boot attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 742–754, Sep. 2018, doi: 10.1109/TDSC.2016.2631548.
- [7] A. Perez-Resca, M. Garcia-Bosque, C. Sanchez-Azqueta, and S. Celma, "A new method for format preserving encryption in high-data rate communications," *IEEE Access*, vol. 8, pp. 21003–21016, 2020, doi: 10.1109/ACCESS.2020.2968816.
- [8] H. M. Heys and L. Zhang, "Pipelined statistical cipher feedback: A new mode for high-speed self-synchronizing stream encryption," *IEEE Transactions on Computers*, vol. 60, no. 11, pp. 1581–1595, Nov. 2011, doi: 10.1109/TC.2010.167.
- [9] D. Chakraborty and P. Sarkar, "HCH: A new tweakable enciphering scheme using the hash-counter-hash approach," *IEEE Transactions on Information Theory*, vol. 54, no. 4, pp. 1683–1699, Apr. 2008, doi: 10.1109/TIT.2008.917623.

- [10] L. Du, C. Huang, W. Guo, J. Ma, X. Ma, and Y. Tang, "Reconfigurable intelligent surfaces assisted secure multicast communications," *IEEE Wireless Communications Letters*, vol. 9, no. 10, pp. 1673–1676, Oct. 2020, doi: 10.1109/LWC.2020.3001119.
- [11] S. Wei *et al.*, "Physical layer secure key distribution based on artificial amplitude noise in QAM/QNSC optical communication systems," *IEEE Communications Letters*, vol. 27, no. 9, pp. 2288–2292, Sep. 2023, doi: 10.1109/LCOMM.2023.3295809.
- [12] M. Cheng, J.-B. Wang, and J. Cheng, "A new lower bound based secure beamforming in MISO communication networks," *IEEE Communications Letters*, vol. 23, no. 9, pp. 1474–1478, Sep. 2019, doi: 10.1109/LCOMM.2019.2923212.
- [13] J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on Edge computing for vehicular Ad Hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1621–1632, May 2019, doi: 10.1109/TITS.2018.2827460.
- [14] H. Fang, X. Wang, and L. Hanzo, "Learning-aided physical layer authentication as an intelligent process," *IEEE Transactions on Communications*, vol. 67, no. 3, pp. 2260–2273, Mar. 2019, doi: 10.1109/TCOMM.2018.2881117.
- [15] K. Xue, X. Luo, Y. Ma, J. Li, J. Liu, and D. S. L. Wei, "A distributed authentication scheme based on smart contract for roaming service in mobile vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 5284–5297, May 2022, doi: 10.1109/TVT.2022.3148303.
- [16] F. Shahid, I. Ahmad, M. Imran, and M. Shoaib, "Novel one time signatures (NOTS): A compact post-quantum digital signature scheme," *IEEE Access*, vol. 8, pp. 15895–15906, 2020, doi: 10.1109/ACCESS.2020.2966259.
- [17] A. Sengupta, E. R. Kumar, and N. P. Chandra, "Embedding digital signature using encrypted-hashing for protection of DSP cores in CE," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 3, pp. 398–407, Aug. 2019, doi: 10.1109/TCE.2019.2924049.
- [18] I. Z. Berta, L. Buttyan, and I. Vajda, "A framework for the revocation of unintended digital signatures initiated by Malicious Terminals," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 3, pp. 268–272, Mar. 2005, doi: 10.1109/TDSC.2005.28.
- [19] P. D. Ojha and R. C. Hansdah, "A heuristic approach to detect MPLS L3 VPN misconfiguration in multi-homed multi-VRF site-redundant CE environments," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2294–2307, Jun. 2021, doi: 10.1109/TNSM.2020.3009301.
- [20] S. Budiayanto and D. Gunawan, "Comparative analysis of VPN protocols at layer 2 focusing on voice over internet protocol," *IEEE Access*, vol. 11, pp. 60853–60865, 2023, doi: 10.1109/ACCESS.2023.3286032.
- [21] H. Qian, S. Dispensa, and D. Medhi, "Balancing request denial probability and latency in an agent-based VPN architecture," *IEEE Transactions on Network and Service Management*, vol. 7, no. 4, pp. 282–295, Dec. 2010, doi: 10.1109/TNSM.2010.1012.100103.
- [22] X. He, J. Liu, C.-T. Huang, D. Wang, and B. Meng, "A security analysis method of security protocol implementation based on unpurified security protocol trace and security protocol implementation ontology," *IEEE Access*, vol. 7, pp. 131050–131067, 2019, doi: 10.1109/ACCESS.2019.2940512.
- [23] M. A. Saleem, S. Shamshad, S. Ahmed, Z. Ghaffar, and K. Mahmood, "Security analysis on 'a secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems,'" *IEEE Systems Journal*, vol. 15, no. 4, pp. 5557–5559, Dec. 2021, doi: 10.1109/JSYST.2021.3073537.
- [24] J. Guo, Y. Du, X. Wu, and M. Li, "An anti-quantum authentication protocol for space information networks based on ring learning with errors," *Journal of Communications and Information Networks*, vol. 6, no. 3, pp. 301–311, Sep. 2021, doi: 10.23919/JCIN.2021.9549124.
- [25] C.-Y. Huang, W. C. Shen, Y.-H. Tseng, Y.-C. King, and C.-J. Lin, "A contact-resistive random-access-memory-based true random number generator," *IEEE Electron Device Letters*, vol. 33, no. 8, pp. 1108–1110, Aug. 2012, doi: 10.1109/LED.2012.2199734.
- [26] P.-H. Tseng, M.-H. Lee, Y.-H. Lin, H.-L. Lung, K.-C. Wang, and C.-Y. Lu, "ReRAM-based pseudo-true random number generator with high throughput and unpredictability characteristics," *IEEE Transactions on Electron Devices*, vol. 68, no. 4, pp. 1593–1597, Apr. 2021, doi: 10.1109/TED.2021.3057028.
- [27] C.-B. Chen, T. Chen, Y.-H. Huang, and Y.-H. Huang, "35.56-Gbits/sec chaos random number generator supporting 1.2-Gsamples/sec noise generation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 10, pp. 3802–3806, Oct. 2023, doi: 10.1109/TCSII.2023.3293786.
- [28] R. J. Parker, "Entropy justification for metastability based nondeterministic random bit generator," in *2017 IEEE 2nd International Verification and Security Workshop (IVSW)*, Jul. 2017, pp. 25–30, doi: 10.1109/IVSW.2017.8031540.
- [29] R. Della Sala and G. Scotti, "Exploiting the DD-cell as an ultra-compact entropy source for an FPGA-based re-configurable PUF-TRNG architecture," *IEEE Access*, vol. 11, pp. 86178–86195, 2023, doi: 10.1109/ACCESS.2023.3304901.
- [30] S. Fu *et al.*, "RHS-TRNG: A resilient high-speed true random number generator based on STT-MTJ device," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 10, pp. 1578–1591, Oct. 2023, doi: 10.1109/TVLSI.2023.3298327.
- [31] X. Li *et al.*, "A 144-fJ/Bit reliable and compact TRNG based on the diffusive resistance of 3-D resistive random access memory," *IEEE Transactions on Electron Devices*, vol. 70, no. 8, pp. 4139–4144, Aug. 2023, doi: 10.1109/TED.2023.3288839.
- [32] C.-Y. Li, Y.-H. Chen, T.-Y. Chang, L.-Y. Deng, and K. To, "Period extension and randomness enhancement using high-throughput reseeding-mixing PRNG," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 2, pp. 385–389, Feb. 2012, doi: 10.1109/TVLSI.2010.2103332.
- [33] J. M. Mcginthy and A. J. Michaels, "Further analysis of PRNG-based key derivation functions," *IEEE Access*, vol. 7, pp. 95978–95986, 2019, doi: 10.1109/ACCESS.2019.2928768.
- [34] S. S. da Silva, M. Cardoso, L. Nardo, E. Nepomuceno, M. Hubner, and J. Arias-Garcia, "A new chaos-based PRNG hardware architecture using the HUB fixed-point format," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–8, 2023, doi: 10.1109/TIM.2023.3235457.
- [35] S. Chen and B. Li, "A dynamic reseeding DRBG based on SRAM PUFs," in *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, Oct. 2016, pp. 50–53, doi: 10.1109/CyberC.2016.18.
- [36] M. Amin and M. Afzal, "On the vulnerability of EC DRBG," in *2015 12th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Jan. 2015, pp. 318–322, doi: 10.1109/IBCAST.2015.7058523.
- [37] G. Revadigar, C. Javali, Y. Li, and S. Viswanathan, "On the effectiveness of electric network frequency (ENF) as a source of randomness," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Aug. 2019, pp. 849–854, doi: 10.1109/TrustCom/BigDataSE.2019.00123.
- [38] F. Wen, M. Qin, P. V. Gratz, and A. L. N. Reddy, "Hardware memory management for future mobile hybrid memory systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 11, pp. 3627–3637, Nov. 2020, doi: 10.1109/TCAD.2020.3012213.




- [39] B. K. Mohanty, P. K. Meher, S. Al-Maadeed, and A. Amira, "Memory footprint reduction for power-efficient realization of 2-d finite impulse response filters," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 1, pp. 120–133, Jan. 2014, doi: 10.1109/TCSI.2013.2265953.
- [40] P. Jokic, S. Emery, and L. Benini, "Improving memory utilization in convolutional neural network accelerators," *IEEE Embedded Systems Letters*, vol. 13, no. 3, pp. 77–80, Sep. 2021, doi: 10.1109/LES.2020.3009924.
- [41] H. M. Salih and R. S. Al Mahdawi, "The security of RC4 algorithm using keys generation depending on user's retina," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 1, pp. 452–463, Oct. 2021, doi: 10.11591/ijeecs.v24.i1.pp452-463.
- [42] D. H. Elkamchouchi, A. D. Algarni, R. M. Ghoniem, and H. G. Mohamed, "A deep learning signed medical image based on cryptographic techniques," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 1, pp. 481–495, Jan. 2022, doi: 10.11591/ijeecs.v29.i1.pp481-495.
- [43] N. A. Kako, H. T. Sadeeq, and A. R. Abraham, "New symmetric key cipher capable of digraph to single letter conversion utilizing binary system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, no. 2, pp. 1028–1034, May 2020, doi: 10.11591/ijeecs.v18.i2.pp1028-1034.
- [44] O. Kara and M. F. Esgin, "On analysis of lightweight stream ciphers with keyed update," *IEEE Transactions on Computers*, vol. 68, no. 1, pp. 99–110, Jan. 2019, doi: 10.1109/TC.2018.2851239.
- [45] T. Dierks and S. Jagannathan, "Online optimal control of affine nonlinear discrete-time systems with unknown internal dynamics by using time-based policy update," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 7, pp. 1118–1129, Jul. 2012, doi: 10.1109/TNNLS.2012.2196708.
- [46] M. A. Selver and C. Guzelis, "Semiautomatic transfer function initialization for abdominal visualization using self-generating hierarchical radial basis function networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 3, pp. 395–409, May 2009, doi: 10.1109/TVCG.2008.198.
- [47] S. Masood, M. N. Doja, and P. Chandra, "Analysis of weight initialization techniques for gradient descent algorithm," in *2015 Annual IEEE India Conference (INDICON)*, Dec. 2015, pp. 1–5, doi: 10.1109/INDICON.2015.7443734.
- [48] G. Thimm and E. Fiesler, "High-order and multilayer perceptron initialization," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 349–359, Mar. 1997, doi: 10.1109/72.557673.
- [49] Y. Qin and Y. Zhang, "Information encryption in ghost imaging with customized data container and XOR operation," *IEEE Photonics Journal*, vol. 9, no. 2, pp. 1–8, Apr. 2017, doi: 10.1109/JPHOT.2017.2690314.
- [50] X. Fu, S. Yang, and Z. Xiao, "Decoding and repair schemes for shift-XOR regenerating codes," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7371–7386, Dec. 2020, doi: 10.1109/TIT.2020.3019168.

## BIOGRAPHIES OF AUTHORS



**Eugene Rhee**    received the Ph.D. degree in electronics from Hanyang University, Korea, in 2010. He was a visiting professor at Chuo University, Japan from 2010 to 2011. Since 2012, he has been with Sangmyung University, Korea, where he is currently an associate professor in the Department of Electronic Engineering. His research area includes microwave, electromagnetic compatibility, electromagnetic interference, and reverberation chamber. He can be contacted at email: eugenerhee@smu.ac.kr.



**Jihoon Lee**    received B.S., M.S., and Ph.D. degrees in electronics engineering from Korea University in 1996, 1998, and 2001, respectively. From 2002 to 2011, he worked at Samsung Electronics as a senior research member. He is currently an associate professor in the Department of Smart Information and Telecommunication Engineering, Sangmyung University, Korea. His research interests include edge computing, secure M2M, and network security. He can be contacted at email: vincent@smu.ac.kr.