ADVANCED QUANTUM TECHNOLOGIES
www.advquantumtech.com

Mem-Transistor-Based Gaussian Error–Generating Hardware for Post-Quantum Cryptography Applications

*Moon-Seok Kim, Shania Rehman, Muhammad Farooq Khan, and Sungho Kim**

Quantum computing can potentially hack the information encrypted by traditional cryptographic systems, leading to the development of post-quantum cryptography (PQC) to counteract this threat. The key principle behind PQC is the "learning with errors" problem, where intentional errors make encrypted information unpredictable. Intentional errors refer to Gaussian distributed data. However, implementing Gaussian distributed errors is challenging owing to computational and memory overhead. Therefore, this study proposes a Gaussian error sampler that employs the intrinsic Gaussian properties of nanometer-scale semiconductor devices. The proposed Gaussian error sampler significantly reduces computational and memory overhead. This work comprehensively evaluates the effectiveness of the proposed device by conducting statistical normality tests and generating quantile–quantile plots. The optimal programming voltage is identified to be −5.25 V, and the experimental results confirmed the Gaussian distribution of error data generated by the proposed module, aligning closely with software-generated Gaussian distributions and distinct from uniform random distributions.

1. Introduction

The advancement of quantum computing technology poses a significant threat to traditional cryptographic systems, which rely on mathematical problems for their security, such as Rivest Shamir Adleman (RSA) and elliptic curve cryptography (ECC) systems.[1,2,3] This is because quantum computing technology can potentially hack confidential information encrypted by the traditional RSA and ECC cryptographic systems.[4–6] Consequently, post-quantum cryptography (PQC) has emerged as a new cryptographic paradigm to counteract the threat posed by quantum computing technology.[7,8] PQC refers to cryptographic systems that a quantum computer cannot hack, thus protecting the confidentiality of the information generated by these systems.[9,7] In other words, a PQC system is a solution to address the advancements of quantum computing technology. To rapidly address threats posed by quantum computers, the National Institute of Standards and Technology has established international standardization in terms of PQC mathematical systems.[10] Lattice- and code-based mathematical problems are promising candidates for PQC algorithms owing to their stability with implementation and compatibility with the existing security protocols.[11,12,13,14] The primary principle of these mathematical algorithms is the "learning with errors" (LWE) problem, which means that an injecting error prevents adversaries from decoding confidential information, even in worst cases.[15,16] In other words, intentional error injections make a ciphered message appear random and unpredictable to adversaries. From an implementation perspective, the most significant difference between PQC and traditional cryptographic systems is the need for devices that consistently generate errors. In the context of LWE problems, these errors are Gaussian distributed signals, also known as the normal distribution.[17,18] Gaussian distribution is a type of probability distribution characterized by its mean and standard deviation, which are derived from the exponential function.[19] However, calculating the exponential function is challenging for hardware such as arithmetic logic units and field programmable gate array devices.[20,21] Thus, various techniques have been devised to generate Gaussian distributed errors, such as Box–Muller transformation, rejection sampling, and Ziggurat sampling.[20–22] These techniques use primitives such as random number generators and precomputed cumulative distribution tables.[20–22] A random number generator generates uniformly distributed

M.-S. Kim
Department of Semiconductor System Engineering
Hanbat National University
125 Dongseo-daero, Yuseong-gu, Daejeon 31538, Republic of Korea
S. Rehman
Department of Semiconductor Systems Engineering
Sejong University
Seoul 05006, Republic of Korea
M. F. Khan
Department of Electrical Engineering
Sejong University
Seoul 05006, Republic of Korea
S. Kim
Division of Electronic & Semiconductor Engineering
Ewha Womans University
Seoul 03760, Republic of Korea
E-mail: sunghok@ewha.ac.kr

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/qute.202400394

DOI: 10.1002/qute.202400394

random numbers, whereas a precomputed cumulative distribution table involves prestoring Gaussian distributed datasets in the system's memory. However, both random number generators and precomputed distribution tables entail significant overhead from an implementation perspective.

In summary, it is crucial to devise an error sampler with a Gaussian distribution for PQC system implementation. However, existing techniques experience significant memory and computation time overhead. Therefore, it is essential to develop a Gaussian error sampler that avoids these overhead factors. In this study, we devised a Gaussian error sampler hardware that leverages the intrinsic characteristics of nanometer-scale semiconductor devices. The proposed Gaussian error sampler exploits the varying electrical resistance characteristics via resistive switching of the semiconductor material. The proposed technology enables PQC implementation without the overhead associated with large computing and memory capacities. Figure S1, Supporting Information illustrates a schematic of the various methods used to implement a Gaussian sampler between the (a) existing and (b) proposed Gaussian samplers. The proposed technique mitigates the overhead against computing and memory capacities as Gaussian generating transistor is adopted.

2. Results and Discussion

In this study, we adopted a mem-transistor (a memristor with a transistor structure) to generate Gaussian distributed errors, also referred to as a Gaussian generating transistor. The term mem-transistor is a combination of "memristor" and "transistor." A memristor is a two-terminal electronic component with variable resistance, whereas a transistor acts as an electrical switch composed of three terminals. The memristor plays a pivotal role in generating Gaussian errors owing to its intrinsic device physics and exhibits Gaussian distribution. A transition-metal dichalcogenide material is employed as a channel of the mem-transistor.[23,24] Bulk traps located in the tin disulfide (SnS_2) nanosheet provide highly reliable nonvolatile resistive switching behavior.[25,26] Resistive switching is achievable through the electrical pulse applied to a gate electrode. The intrinsic resistive switching characteristics of the SnS_2 mem-transistor lead to the generation of Gaussian distributed errors by the proposed device. **Figure 1** shows the schematic and microscopic images of the fabricated SnS_2 mem-transistor. Figure 1a illustrates the circuit diagram of the Gaussian generating mem-transistor, while Figure 1b shows the schematic of the proposed Gaussian generating mem-transistor, displaying its gate, source, and drain terminals. Figure 1c shows the optical microscopic image of the fabricated SnS_2 mem-transistor with its gate, source, and drain terminals, while Figure 1d displays the cross-sectional transmission electron microscope image of the fabricated device. We experimentally demonstrated that the proposed mem-transistor functions as a Gaussian error sampler, which is vital for PQC implementation. Gaussian error data were used for addictive operations to generate ciphered data. **Figure 2a** shows a representative schematic illustration of the PQC encryption process using McEliece cryptography.[27,28] Error data were added to codeword data obtained from the matrix multiplication of message data and the generator matrix. The added ciphered data could not be subjected to cryptanalysis owing to unpredictable error

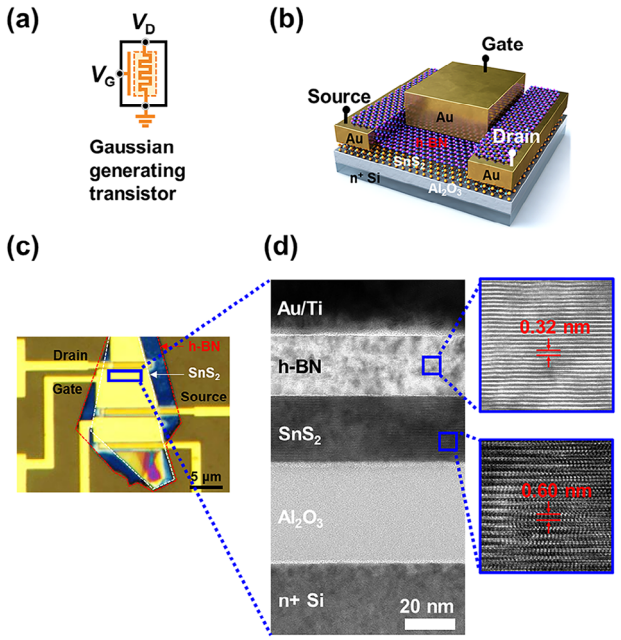

Figure 1. a) Circuit diagram of the proposed Gaussian generating transistor, exhibiting the characteristics of a mem-transistor. b) Schematic illustration of a single Gaussian generating transistor with gate, source, and drain terminals. c) Optical microscopic image showing the top view of the fabricated Gaussian generating transistor. d) Transmission electron microscope images showing the cross-sectional view of the fabricated Gaussian generating transistor.

data. This property ensures resistance to cryptanalysis, including attacks by quantum computers with respect to mathematical cryptography.[29,30] Figure S2, Supporting Information illustrates the entire process of LWE-based PQC: (a) public key generation, (b) encryption, and (c) decryption processes. Error data were employed during the public key generation process.

The Gaussian noise sampler is a fundamental primitive for PQC applications. Figure 2b,c shows a comparison of cryptanalyzed images between the systems (b) without addictive Gaussian errors and (c) with addictive Gaussian errors. Without errors, the ciphered image was perfectly decrypted because quantum computers can exactly compute the inverse matrix of the generator matrix.[17] By contrast, the added errors prevented quantum computers from decrypting the images, as shown in Figure 2c. Figure 2d shows the bit error rate (BER) according to the standard deviation of addictive Gaussian errors. A low BER indicates that quantum computers can successfully decrypt the data, whereas a high BER indicates that quantum computers fail to decrypt the data.

The Gaussian distribution was demonstrated as resistance switched in a Gaussian generating transistor. The resistive switching phenomenon was induced by the Gaussian generating procedures in the Gaussian generating transistor. Gaussian generating procedures are categorized by programming and erasing operations. The programming operation involved the application of a negative voltage pulse (V_P) to the gate terminal with a magnitude of −5.25 V and a duration of 10 ms, whereas the erasing operation involved the application of a positive voltage pulse (V_E) to the gate terminal with a magnitude of 2.5 V and a

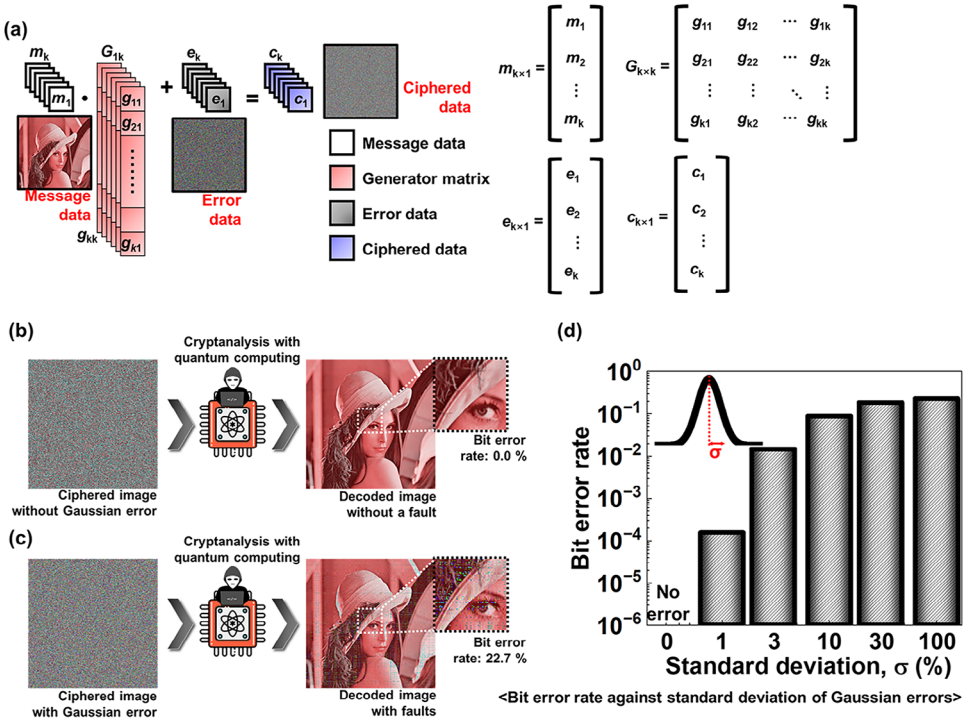

Figure 2. a) Representative schematic illustration of the McEliece cryptography encrypting process, a type of post-quantum cryptography (PQC). Cryptanalyzed images between b) without Gaussian errors and c) with addictive Gaussian errors. Gaussian errors prevent quantum computing from decoding original images. d) Bit error rate according to the standard deviation of Gaussian errors when quantum computing decodes ciphered images. Gaussian errors with a high standard deviation make quantum computing decode images with high error rates.

duration of 10 ms. After completing Gaussian generating procedures, reading operations were performed to characterize the resistance of target devices. The reading operation involved applying a small voltage (V_R) to the drain terminal for 10 ms. **Figure 3a** illustrates the Gaussian generating procedures and reading operations, whose sequences are composed of programming, erasing, and reading operations. Gaussian generating procedures resulted in the resistive switching phenomenon. Thus, resistance variations in the form of Gaussian distributions were attributed to programming and erasing operations. Finally, a reading operation was conducted to characterize the resistance of Gaussian generating transistors, which was extracted using the reading current (I_R) value according to Ohm's law.

Figure 3b shows the energy band diagram of the fabricated SnS_2 mem-transistor (Gaussian generating transistor) in the equilibrium state. The previously reported electron affinity and energy bandgap of SnS_2 are 4.22 and 2.11 eV, respectively, whereas those of n+ polysilicon are 4.1 and 1.12 eV, respectively.[25,26,31] Furthermore, the difference between the conduction band edge and the Fermi level ($E_C - E_F$) of the fabricated SnS_2 is 0.04 eV.[25,26,31] Thus, the band alignment at the equilibrium state was described as shown in Figure 3b after establishing contact among n+ polysilicon, Al_2O_3, and SnS_2 materials.

Figure 3c shows the schematic illustrations of the resistive switching process in the SnS_2 mem-transistor (Gaussian generating transistor) during the Gaussian generating operation. The resistive switching phenomenon was generated by the charge trapping/de-trapping process, as shown in Figure 3c. First, the high negative voltage pulse ($V_P = -5.25$ V) completely depleted the number of traps in the SnS_2 channel. In other words, during the negative voltage pulse, the number of electrons with energy levels higher than the E_F was depleted in both interface and bulk traps. Next, the positive voltage pulse ($V_E = +2.5$ V) facilitated electron trapping and diffusion processes. Electron trapping refers to the rapid trapping of electrons from the inverted channel in the interface traps, and the diffusion process involves the diffusion of these trapped electrons toward the bulk trap inside the SnS_2 layer over time. The resistive switching process in the SnS_2 mem-transistor possesses two main characteristics. First, the trapped electrons at bulk traps are responsible for the nonvolatile conductance at the SnS_2 layer.[25,26] Next, the resistance modulation exhibits intrinsic stochastic properties, which indicate that the resistance value varies as programming and erasing voltage pulses are applied.[25,31] In summary, Gaussian distribution is formed by the resistive switching phenomenon of SnS_2 mem-transistors (Gaussian generating transistors). In other words, the resistance resulting from the resistive switching of the SnS_2 mem-transistor (Gaussian generating transistor) exhibits nonvolatile characteristics that form Gaussian normal distribution. Although previous studies have revealed that restive switching occurs owing to the charge trapping and detrapping processes,[23,24] the present study experimentally demonstrated that resistive statistical distribution is formed due to Gaussian normal distribution.

Figure 3d depicts the transient curve tracking the resistance of the Gaussian generating transistor as Gaussian generating procedures and reading operations are repeated 200 times. To visually confirm that this resistance distribution followed a

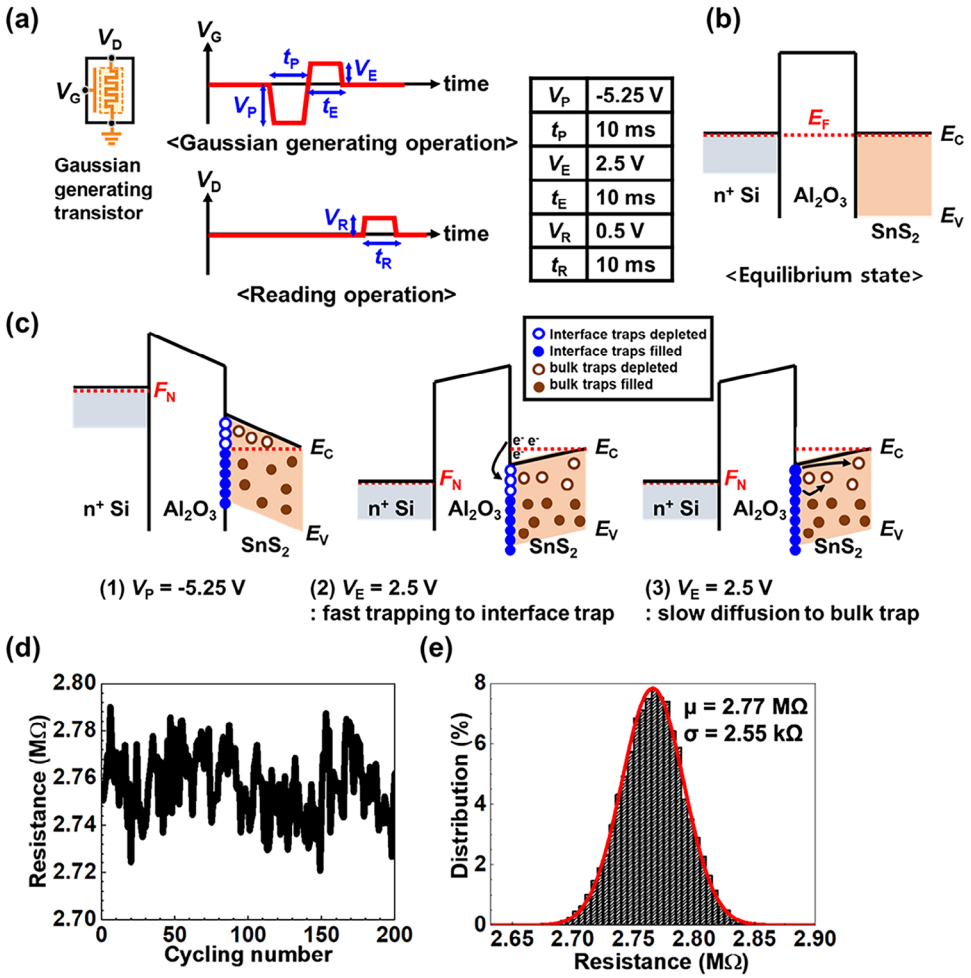

Figure 3. a) Schematic illustration of the procedures comprising Gaussian generating and reading operations. b) Energy band diagram consisting of n+ poly Si, Al_2O_3, and SnS_2 stacks at the equilibrium state. The conduction band (E_C) between n+ Si and SnS_2 is almost flat because the electron affinities of n+ Si and SnS_2 are 4.1 and 4.22 eV, respectively. c) Energy band diagrams showing the programming ($V_P = -5.25$ V) and erasing voltage ($V_E = 2.5$ V) stages. The diagrams show the procedure to generate the resistive switching phenomenon. The SnS_2 channel–depleted region is generated during the programming voltage stage. Next, electrons undergo fast trapping to form interface traps during the erasing voltage stage. Finally, the trapped interface electrons diffuse the bulk traps. d) Transient curve of resistance values when Gaussian generating and reading operations are repeated 200 times. e) Histogram of resistance values when Gaussian generating and reading operations are repeated 10000 times.

Gaussian distribution, the Gaussian generating procedures of a single Gaussian generating transistor were repeated 10 000 times. Figure 3e shows a histogram used to extract the resistance probability distribution during 10 000 resistive switching operations, wherein the probability distributions are indicated by black bars. The red line represents an ideal Gaussian normal distribution with a mean of 2.77 MΩ and a standard deviation of 2.55 kΩ. Figure 3e shows the presence of a visual similarity between the theoretical Gaussian and experimental resistance distributions of the proposed Gaussian generating transistor.

We found that Gaussian distributed error data were generated by a Gaussian generating module. The Gaussian generating module was composed of Gaussian generating transistors and peripheral components such as a microcontroller and a transimpedance amplifier (TIA). With the aid of peripheral integrated circuits (ICs), the Gaussian generating module system was demonstrated using the Gaussian generating transistor. Consequently, the Gaussian generating module was deemed suitable for supplying Gaussian error data for PQC applications. **Figure** 4a illustrates the procedure to sequentially generate Gaussian error data from the Gaussian generating module. The generated error data ranged from -(q-1)/2 to +(q-1)/2. Typically, the value of integer q is determined to be within the range of 128 to 256, whose decisions are designed according to the mathematical cryptographic security.[17,32,33] Figure 4b illustrates the generation of Gaussian error data by the Gaussian generating module upon request. First, the PQC system issues a command to generate Gaussian noise. The ATmega32U4 microcontroller acts as an intermediary between the PQC system and the Gaussian generating primitives. Next, the ATmega32U4 delivers voltage pulse signals to the Gaussian generating transistor for performing the Gaussian generating procedures and reading operations. The ICL7660 IC acts as a voltage converter to generate the negative voltage pulse, V_P (−5,.25 V). Subsequently, when reading operations are

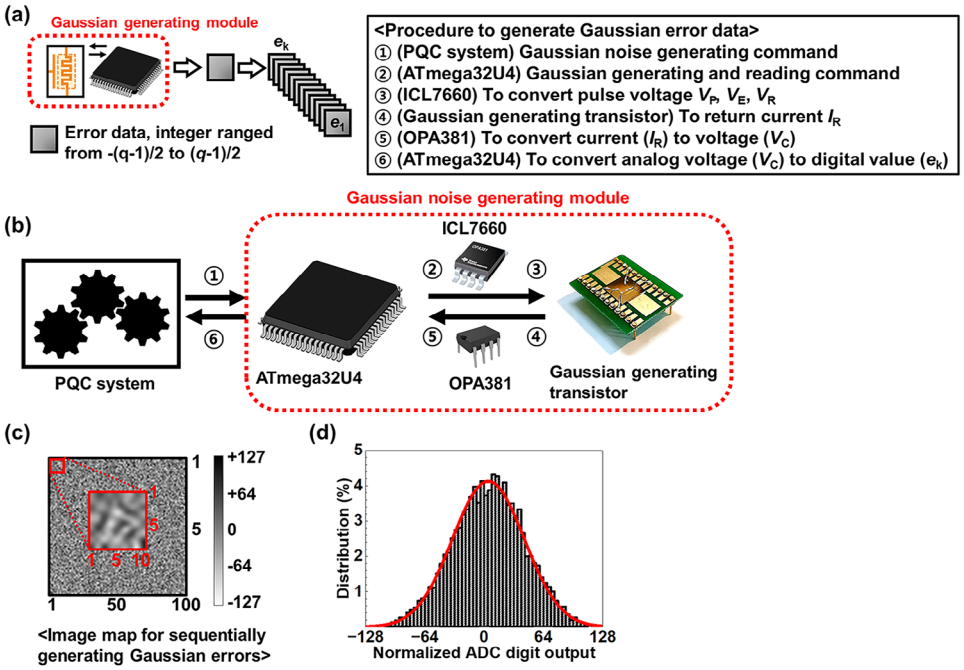

Figure 4. a) Procedure to generate Gaussian error data using the Gaussian noise–generating module. b) Procedure of generating Gaussian error data using the Gaussian noise–generating module according to submodules, such as ATmega32U4, ICL7660, OPA381, and Gaussian generating transistor. c) Square-typed image maps for sequentially generating Gaussian errors: 100 × 100 integers (black: + 127; white: −127). d) Histogram of digital data obtained from Figure 3c for 10000 integer samples.

performed, the Gaussian generating transistor generates a corresponding current (I_R) that flows toward the microcontroller ATmega32U4. OPA381 IC serves as a TIA, converting the current signal to a corresponding voltage signal. The resistance of transistors is extracted as V_R and is divided by I_R according to Ohm's law. The extracted resistance is converted into integer error data with the aid of the analog-to-digital converter (ADC), which is the internal component of ATmega32U4. The integer error data ranges from -(q-1)/2 to +(q-1)/2. Finally, the generated integer error data enable the PQC system to perform secure encryption and decryption against quantum computing. Figure 4c visually demonstrates the Gaussian error data generated by the Gaussian noise–generating module. The procedure to generate Gaussian error data is shown in Figure 4b. Parameter q was set to 128 and was verified to provide mathematical cryptographic security.[17,32,33] Figure 4d depicts the histogram used to statistically verify that the generated error data follow a Gaussian normal distribution from digital data shown in Figure 4c. We confirmed that the experimental results of the Gaussian noise–generating module visually resemble the theoretical Gaussian normal distribution, similar to that shown in Figure 3e.

The proposed Gaussian generating module produced error data that possessed a Gaussian normal distribution. **Figure 5** shows the normality test conducted to quantitatively verify whether the proposed Gaussian generating module can produce data with Gaussian distribution characteristics. For comparison, three groups were established: experimental group (group A), consisting of error data generated by the proposed Gaussian error–generating module; control group B, consisting of Gaussian error distributions generated using software; and control group C, comprising uniformly distributed random numbers generated using software. The comparison between groups A and B was performed to quantitatively evaluate how closely the hardware-based Gaussian error–generating module mimics the ideal Gaussian normal distribution achieved by the existing software-based Gaussian generating technology. Conversely, the comparison results between groups A and C highlighted the distinct differences in quantitative normality metrics between the implemented Gaussian normal distribution and uniform random numbers. Figure 5a displays the histogram of noise data distributions of groups A, B, and C. Although group A included experimental data obtained from the proposed Gaussian noise–generating module, groups B and C were the control groups, designed via software, and were compared with group A in the quantitative normality test.

Figure 5b shows the quantile–quantile (Q–Q) plots of all groups. The Q–Q plot is a graphical tool used to compare sample datasets with a theoretical Gaussian normal distribution.[34,35] The straight line marked on the Q–Q plot indicates that the sample datasets possess a perfect Gaussian normal distribution.[34,35] As shown in Figure 5b, both groups A and B exhibited straight lines, whereas group C showed a scattered pattern. The Q–Q plots indicate that the proposed Gaussian noise–generating module is suitable for PQC applications in terms of mathematical normality. Figure 5c,d,e shows the mathematical metrics adopted to provide quantitative information in terms of mathematical normality. Figure 5c displays the Kullback–Leibler (KL) divergence values of all groups. KL divergence is a metric used to quantify the mathematical difference between one probability distribution and a reference probability distribution.[36,37] Thus, a

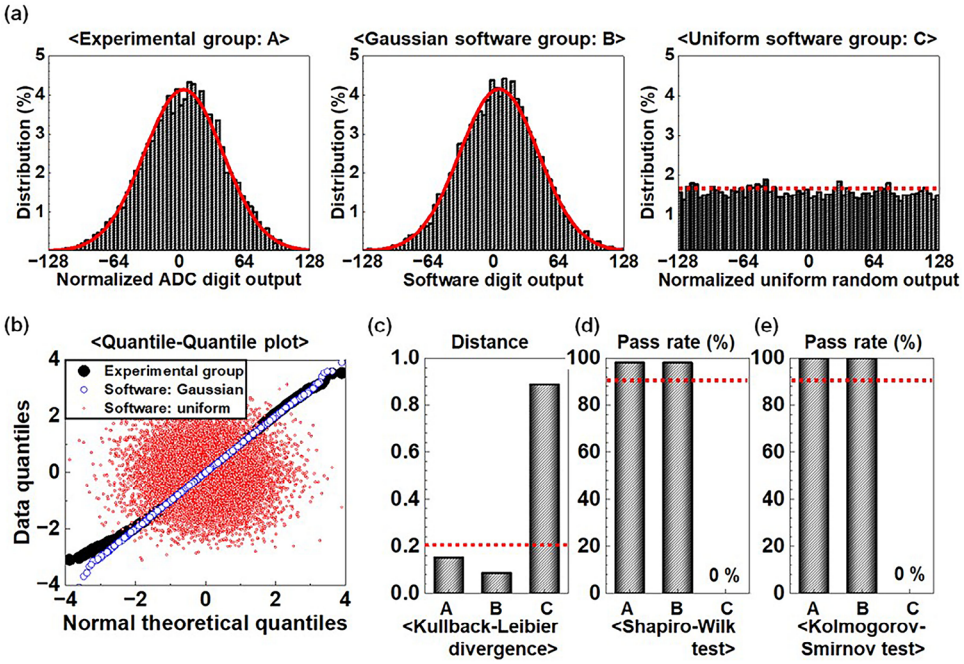

Figure 5. Normality test results of Gaussian errors generated by the proposed Gaussian noise–generating module and control groups. a) Histograms of groups A, B, and C. Group A indicates experimental data obtained from the proposed Gaussian noise–generating module and is called the experimental group. Groups B and C are the control groups, designed using software. Group B simulates an ideal Gaussian normal distribution, whereas group C simulates ideal uniform random numbers. b) Quantile–Quantile plot to compare the datasets of a theoretical Gaussian normal distribution between groups A, B, and C. c) Kullback–Leibler divergence value used to quantifiably compare the mathematical difference between sample datasets and ideal Gaussian normal distribution for groups A, B, and C. d) Shapiro–Wilk normality pass rate (%) to verify that sample datasets possess Gaussian normal distribution for groups A, B, and C. e) Kolmogorov–Smirnov normality pass rate (%) to determine whether sample datasets possess Gaussian normal distribution for groups A, B, and C.

KL divergence value of 0 indicates that the sample dataset perfectly matches the theoretical Gaussian normal distribution. Both groups A and B exhibited KL divergence values of less than 0.2, indicating a close match with the Gaussian normal distribution. However, group C showed a KL divergence value greater than 0.8, indicating a lesser correlation with the Gaussian normal distribution. Figure 5d,e shows the pass rate in terms of Shapiro–Wilk (SW) normality and Kolmogorov–Smirnov (KS) tests for groups A, B, and C. SW normality test is used to verify whether sample datasets possess a normal distribution, and the KS test is used to determine whether two samples are derived from the same probability distribution.[38,39] Both test results are denoted by pass or fail according to the significance level, which was set to 0.05. Figure 5d,e demonstrates that both groups A and B possess pass rates of more than 95% for normality.

Figure 6 shows the trends of means and standard deviation of the Gaussian distribution when the programming voltage of the proposed Gaussian generating transistor is adjusted. The purpose of experiments controlling the programming voltage was to experimentally verify optimal points in terms of the programming voltage. Figure 6a illustrates the changes in V_p during programming operations. Figure 6b shows that both the statistical mean and standard deviation are adjusted as V_p is adjusted. Figure 6c,d,e displays the results of KL divergence, SW, and KS tests, respectively. The results showed that a V_p of −5.25 V is the optimal point to generate Gaussian distributed error samples. Furthermore, ambient temperature experiments were conducted to verify the environmental properties. The ambient temperature was set to −20, 20, 60, and 100 °C. **Figure** 7a,b,c shows the results of KL divergence, SW, and KS tests, respectively. The findings indicated that the proposed Gaussian error samplers consistently passed the statistical normality test.

3. Conclusions

In this study, we developed a Gaussian noise–generating module using mem-transistors (Gaussian generating transistors) to generate Gaussian distributed errors. Gaussian distributed errors are essential to ensure that encrypted information is unpredictable against quantum computing technology. The intrinsic device physics of Gaussian generating transistors, which exhibit Gaussian distribution, are crucial for generating Gaussian errors. A transition-metal dichalcogenide material was employed as the channel of Gaussian generating transistors, with bulk traps in the SnS_2 thin film demonstrating reliable nonvolatile resistive switching behavior. Resistive switching was achieved through electrical pulses applied to the gate electrode, enabling the generation of Gaussian distributed errors. Experimental results demonstrated the effectiveness of the proposed Gaussian–noise generating module as a Gaussian error sampler, which is vital for PQC implementation. The generated Gaussian error data were used in additive operations to produce ciphered data, providing resistance to cryptanalysis, including quantum computing attacks. Statistical normality tests confirmed the

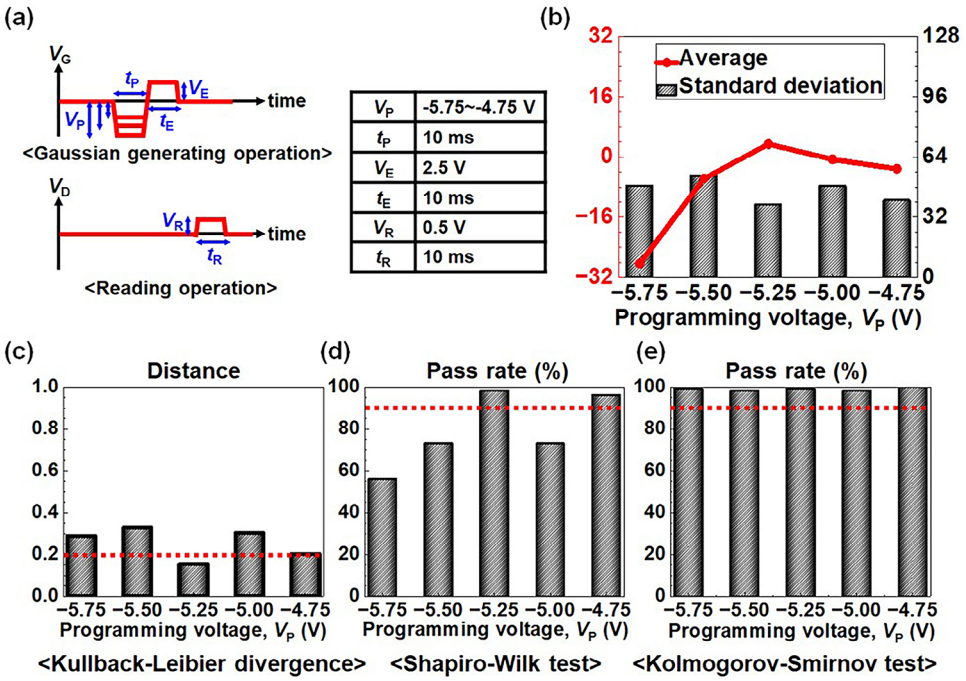

Figure 6. Normality test results when the programming voltage (V_P) is controlled to range from −5.75 to −4.75 V. a) Schematic illustration of procedures when V_P is adjusted. (b) Statistical mean and standard deviation values when V_P ranged from −5.75 to −4.75 V. c) Kullback–Leibler divergence value used to quantifiably compare the mathematical difference between experimental datasets and ideal Gaussian normal distribution at a V_P of −5.75, −5.5, −5.25, 5.0, and −4.75 V. d) Shapiro–Wilk normality pass rate (%) to verify that experimental datasets possess Gaussian normal distribution at a V_P of −5.75, −5.5, −5.25, 5.0, and −4.75 V. e) Kolmogorov–Smirnov normality pass rate (%) to determine whether experimental datasets possess Gaussian normal distribution at a V_P of −5.75, −5.5, −5.25, 5.0, and −4.75 V.

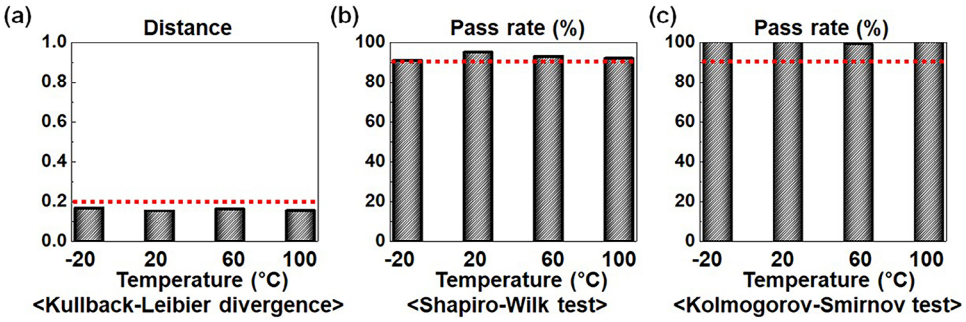

Figure 7. Normality test results against the temperature ranging from −20 to 100 °C. a) Kullback–Leibler divergence value between ideal Gaussian normal distribution and experimental datasets at −20, 20, 60, and +100 °C. b) Shapiro–Wilk normality pass rate (%) to verify that experimental datasets possess Gaussian normal distribution at −20, 20, 60, and +100 °C. c) Kolmogorov–Smirnov normality pass rate (%) to determine whether experimental datasets possess Gaussian normal distribution at −20, 20, 60, and +100 °C.

Gaussian distribution of error data generated by the proposed module, aligning closely with software-generated Gaussian distributions and distinct from uniform random distributions. The study identified the optimal programming voltage (−5.25 V) and verified the robustness of the Gaussian error sampler across various temperatures. In conclusion, the results confirmed that the proposed Gaussian noise–generating module is suitable for PQC applications. Ultimately, the our goal is to develop a system that integrates electronic components such as Gaussian generating transistors and ADCs into an SoC. The system encompasses all PQC functions, such as public key generation, encryption, and decryption, while achieving low power consumption and cost-effectiveness within a single chip.

4. Experimental Section

Fabrication of the Gaussian Error Sampler Device: To implement the proposed Gaussian error–generating transistor, SnS_2–channel mem-transistor was fabricated.[25,26] Figure 1a illustrates the circuit diagram of the proposed Gaussian generating transistor, while Figure 1b shows a schematic of the fabricated Gaussian generating transistor, displaying its

gate, source, and drain terminals. The Gaussian generating transistor is the principle component in constructing a Gaussian generating module. Notably, SnS_2 channel exhibits resistive switching characteristics, which attribute intrinsic stochastic fluctuation and an unpredictable Gaussian distribution. A heavily n-type doped Si wafer (resistivity: <0.005 Ω•cm) was employed to fabricate the SnS_2-channel mem-transistor. The heavily doped Si wafer served as the back-gate electrode. First, a 40-nm-thick Al_2O_3 film was deposited on the Si wafer through atomic layer deposition (Nano-ALD2000, IPS) at 350 °C. The N_2 carrier gas pressure was set to 1.0 Torr, and $Al(CH_3)_3$ and H_2O were introduced into the N_2 carrier flow. The growth rate of the Al_2O_3 layer was ≈1 Å per cycle. Next, an SnS_2 thin film was synthesized, which acted as the current semiconductor channel. The SnS_2 thin film was prepared by exfoliating SnS_2 nanosheets from the bulk SnS_2 (HQ Grpahene) using sticky scotch tape. A 25-nm-thick SnS_2 thin film was then transferred onto the Si/Al_2O_3 substrate. Subsequently, source and drain electrodes composed of Ti/Au (10 nm/50 nm) were fabricated using thermal evaporation and electron-beam lithography. Following this, hexagonal boron nitride (h-BN) and Ti/Au thin films were synthesized to create the top gate. The h-BN thin film was employed as the gate dielectric layer for the top gate, whereas the Ti/Au thin film acted as the metal electrode for the top gate. While the heavily n-typed doped Si bottom gate performs the Gaussian generating operation for all transistors, the Ti/Au top gate is account for the Gaussian generating operation for an individual transistor. A 25-nm-thick h-BN thin film was transferred onto the SnS_2 channel area, and the Ti/Au (10 nm/100 nm) thin film was deposited using a thermal evaporator. These h-BN and Ti/Au thin films acted as top gate electrodes patterned using electron-beam lithography. Figure 1c shows the top view of the fabricated Gaussian generating transistor, and Figure 1d displays the cross-sectional transmission electron microscope image of the fabricated device.

Microcontroller and Peripheral Circuits: The ATmega32U4 microcontroller was integrated into the Gaussian noise–generating module. First, the microcontroller converts analog resistance data into a digit output using the internal ADC. Next, the microcontroller transmits input and output data from the PQC system. A TIA (OPA381, Texas Instruments) was employed to measure the channel resistance (conductance) of the Gaussian generating transistor (mem-transistor), converting the current signal to a voltage signal. Finally, a voltage converter (ICL7660, Renesas) was equipped to generate the negative polarity of the voltage pulse.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2024-00449412), the research fund of Hanbat National University in 2023.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflict of Interest

The authors declare no conflict of interest.

Keywords

gaussian error sampler, learning with error, mem-transistor, post-quantum cryptography

Received: August 13, 2024
Revised: September 30, 2024
Published online: October 22, 2024

[1] A. Trabesinger, *Nature* **2017**, *543*, S1.
[2] J. P. Aumasson, *Comput. Fraud Secur.* **2017**, *6*, 8.
[3] Y. Dong, H. Liu, Y. Fu, C. Xuanxuan, *J. Appl. Phys.* **2023**, *134*, 024401.
[4] V. Bhatia, K. R. Ramkumar, In *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)* IEEE, Greater Noida, **2020**, 89–94.
[5] N. Shinohara, S. Moriai, *New Breeze* **2019**, *2019*, 9.
[6] G. Naqvi, M. B. Umar, S. Ali, *J. Nanoscope* **2023**, *4*, 83.
[7] D. Joseph, R. Misoczki, M. Manzano, J. Tricor, F. D. Pinuaga, O. Lacombe, S. Leichenauer, J. Hidary, P. Venables, R. Hansen, *Nature* **2022**, *605*, 237.
[8] D. Bernstein, T. Lange, *Nature* **2017**, *549*, 188.
[9] S. Sharma, K. R. Ramkumar, A. Kaur, T. Hasija, S. Mittal, B. Singh, In *Modern Electronics Devices and Communication Systems: Select Proceedings of MEDCOM 2021*, Springer, Berlin, **2023**.
[10] R. Shajahan, K. Jain, P. Krishnan, In *2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)* ICMCSI, Lalitpur, **2024**, 132–140.
[11] S. Kumari, M. Singh, R. Singh, H. Tewari, *Comput. Networks* **2022**, *217*, 109327.
[12] C. Zeng, D. He, Q. Feng, C. Peng, M. Luo, *J. Inf. Secur. Appl.* **2024**, *83*, 103782.
[13] D. Bucerzan, D. Vlad, T. K. Hervé, In *Innovative Security Solutions for Information Technology and Communications: 10th International Conference* SecITC, Bucharest, **2017**, 129–149.
[14] H. Nejatollahi, N. Dutt, S. Ray, F. Regazzoni, I. Banerjee, R. Cammarota, *ACM Comput Surv* **2019**, *51*, 1.
[15] R. Lindner, P. Chris, In *Topics in Cryptology–CT-RSA 2011: The Cryptographers Track at the RSA Conference* Springer, San Francisco, **2011**, 319–339.
[16] G. Alagic, D. Apon, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, Y. Liu, C. Miller, D. Mondy, R. Peralta, R. Perlner, A. Robinson, D. Smith-Tone, Status report on the third round of the NIST post-quantum cryptography standardization process. National Institute of Standards and Technology (NIST) Technical report NISTIR 8413 (**2022**). https://doi.org/10.6028/nist.ir.8413.
[17] R. Agrawal, L. Bu, M. A. Kinsy, In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* IEEE, San Jose, **2020**, 295–304.
[18] R. Kuang, M. Perepechaenko, M. Barbeau, *Quantum Inf. Process.* **2022**, *21*, 360.
[19] A. Wang, W. Tan, K. K. Parhi, Y. Lao, In *2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)* IEEE, Oxford, **2022**, 1–6.
[20] D. Bellizia, N. E. Mrabet, A. P. Fournaris, S. Pontie, F. Regazzoni, F. Standaert, E. Tasso, E. Valea, In *2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)* IEEE, Athens, Greece **2021**, 1–6.
[21] E. Karabulut, A. Erdem, A. Aydin, *IEEE Trans. Comput.* **2021**, *71*, 1810.
[22] Y. Fazea, F. Mohammed, M. Alsamman, In *2023 3rd International Conference on Emerging Smart Technologies and Applications (eSmarTA)* IEEE, Tiaz, Yemen, **2023**, 1–8.
[23] R. Ge, X. Wu, L. Liang, S. M. Hus, Y. Gu, E. Okogbue, H. Chou, J. Shi, Y. Zhang, S. K. Banerjee, Y. Jung, J. C. Lee, D. Akinwande, *Adv. Mater.* **2021**, *33*, 2007792.

[24] J. Jian, P. Dong, Z. Jian, T. Zhao, C. Miao, H. Chang, J. Chen, Y. Chen, Y. Chen, H. Feng, B. Sorli, *ACS Nano* **2022**, *16*, 20445.
[25] S. Rehman, M. S. Kim, M. F. Khan, S. Kim, *Nano Energy* **2024**, *127*, 109764.
[26] S. Rehman, M. F. Khan, H. D. Kim, S. Kim, *Nat. Commun.* **2022**, *13*, 2804.
[27] H. C. Ukwuoma, G. Arome, A. Thompson, B. K. Alese, *Open Comput. Sci.* **2022**, *12*, 142.
[28] S. R. Shrestha, Y. S. Kim, In *2014 14th International Symposium on Communications and Information Technologies (ISCIT)* ISCIT, Incheon, **2014**, 368–372.
[29] J. Wang, L. Liu, S. Lyu, Z. Wang, M. Zheng, F. Lin, Z. Chen, L. Yin, X. Wu, C. Ling, *Sci. China Inf. Sci.* **2022**, *65*, 111301.
[30] D. Stebila, M. Michele, In *International Conference on Selected Areas in Cryptography* SAC, St. John's, Newfoundland and Labrador, Canada **2016**, 14–37.
[31] S. Rehman, M. F. Khan, H. D. Kim, S. Kim, *Nano Energy* **2023**, *109*, 108333 .
[32] D. Kundi, A. Khalid, S. Bian, C. Wang, M. O'Neill, W. Liu, *IEEE Internet Things J* **2021**, *9*, 10492.
[33] L. Bahler, G. D. Crescenzo, Y. Polyakov, K. Rohloff, D. B. Cousins, In *2017 International Conference on High Performance Computing & Simulation (HPCS)* HPCS, Genoa, **2017**, 761–768.
[34] T. J. Cleophas, A. H. Zwinderman, In *Machine Learning in Medicine – A Complete Overview*, Springer, Berlin, **2020**.
[35] N. H. Augustin, E. A. Sauleau, S. N. Wood, *Comput. Stat. Data Anal.* **2012**, *56*, 2404.
[36] F. Nielsen, *J. Indian Inst. Sci.* **2022**, *102*, 1177.
[37] N. Bouhlel, A. Dziri, *IEEE Signal Process Lett.* **2019**, *26*, 1021.
[38] C. Gatidis, M. Schleiss, C. Unal, H. Russchenberg, *J. Atmos. Ocean Technol.* **2020**, *37*, 1765.
[39] F. Wang, W. Xiaodong, *IEEE Trans. Commun.* **2010**, *58*, 2324.