Hindawi
Security and Communication Networks
Volume 2021, Article ID 2439427, 16 pages
https://doi.org/10.1155/2021/2439427

Research Article

A Novel TRNG Based on Traditional ADC Nonlinear Effect and Chaotic Map for IoT Security and Anticollision

Gang Li,[1] **Haoyang Sun**,[1] **Peiqi Wu**,[1] **Yuedan Zhou**,[1] **Xiaochuan Fang**,[2] **Zhenbing Li**,[1] **Jian Li**,[1] **Yongjun Huang**,[1] **and Guangjun Wen**[1]

[1]*University of Electronic Science and Technology of China, Chengdu, China*
[2]*Queen Mary University of London, London, UK*

Correspondence should be addressed to Guangjun Wen; wgj@uestc.edu.cn

Received 10 August 2021; Revised 12 September 2021; Accepted 7 October 2021; Published 23 October 2021

Academic Editor: Zhili Zhou

In the rapidly developing Internet of Things (IoT) applications, how to achieve rapid identification of massive devices and secure the communication of wireless data based on low cost and low power consumption is the key problem to be solved urgently. This paper proposes a novel true random number generator (TRNG) based on ADC nonlinear effect and chaotic map, which can be implemented by traditional processors with built-in ADCs, such as MCU, DSP, ARM, and FPGA. The processor controls the ADC to sample the changing input signal to obtain the digital signal D_{ADC} and then extracts some bits of D_{ADC} to generate the true random number (TRN). At the same time, after a delay based on D_{ADC}, the next time ADC sampling is carried out, and the cycle continues until the processor stops generating the TRN. Due to the nonlinear effect of ADC, the D_{ADC} obtained from each sampling is stochastic, and the changing input signal will sharply change the delay time, thus changing the sampling interval (called random interval sampling). As the input signal changes, D_{ADC} with strong randomness is obtained. The whole operation of the TRNG resembles a chaotic map, and this method also eliminates the pseudorandom property of chaotic map by combining the variable input signal (including noise) with the nonlinear effect of ADC. The simulation and actual test data are verified by NIST, and the verification results show that the random numbers generated by the proposed method have strong randomness and can be used to implement TRNG. The proposed TRNG has the advantages of low cost, low power consumption, and strong compatibility, and the rate of generating true random number is more than 1.6 Mbps (determined by ADC sampling rate and processor frequency), which is very suitable for IoT sensor devices for security encryption algorithms and anticollision.

1. Introduction

In recent years, random numbers (RNs) have been widely used in the fields of encryption algorithm, wireless communication, statistical analysis, and radio frequency identification (RFID) [1–10]. RNs can be divided into pseudorandom numbers (PRNs) [11] and true random numbers (TRNs) [12]. PRNs are realized by deterministic algorithms, which have periodic behaviors and can be completely repeated. Their randomness is determined by the complexity and computational accuracy of the algorithm [13]. TRNs are often derived from physical phenomena (such as thermal noise and scintillation noise) and realized by combining certain algorithms and postprocessing, which have truly unpredictable characteristics [14].

With the rapid development of IoT technology, wireless sensor networks (WSNs) have been deeply studied and widely applied in various fields [15–21]. Due to the explosive growth of wireless communication equipment, the communication security has attracted more and more attention [22, 23]. It is known that the encryption algorithm can effectively improve the security performance of wireless communication system, and RNs play a very important role in the encryption algorithm [24]. Although PRNs do not require external circuits, as for deterministic algorithms, PRNs are very vulnerable to malicious attacks. Also, in WSNs, especially in wearable and implantable systems, the design of low-power and low-resource consuming structures is crucial, because it can extend the longevity of batteries or lengthen the distance of wireless communication between

passive sensing nodes. To improve the security performance of the wireless communication in an effective method, it is obligatory to design a special true random number generator (TRNG) applied to sensing nodes.

The noise of analog circuit is used as the entropy source, making TRNG extremely susceptible to noise. According to the manifestation form of noise, the structure of TRNG can be divided into three categories, as shown in Figure 1: (1) comparison structure based on thermal noise [25–27]; (2) beat frequency detection (BFD) structure based on clock jitter [28–32]; (3) ADC residual recycling structure [33–39], also known as chaotic map. In Figure 1, the red dotted part can be omitted.

The noise comparison structure harvests the random information of the entropy source (resistance thermal noise) with comparators (equivalent to 1-bit ADC) and converts the noise signal into a random sequence. The noise must be amplified to a certain level to meet the accuracy requirements of the comparator/ADC. In order to make the amplified noise output as white as possible, a high-gain and wide-bandwidth amplifier is required. This amplifier has high power and cost, and it is not suitable for low power or low cost equipment.

The BFD structure harvests the random information of an entropy source (clock jitter noise of an oscillator) with a register, which is a common method to realize TRNs. This method requires the use of custom chip or Field Programmable Gate Array (FPGA) [40–44]. BFD is a more reliable noise sampling technique compared to noise comparison. However, due to the oscillator jitter is insufficient, the generated data is not random enough. Moreover, two continuously oscillating clocks are energy engulfing, resulting in the increase of the power consumption of the system. Furthermore, the structure also requires special chips or FPGA. In other words, it not only increases the cost and power consumption of the system, but also has finite applications, as it cannot be used in MCU, DSP, ARM, and other traditional microprocessors.

The ADC based on residual recycling structure takes the quantization error of the ADC as its next input signal. After several iterations, the RNs of the ADC output show completely different characteristics. This is exactly the property of chaotic map—a small change of the input signal leads to an utterly distinctive output. Therefore, the structure based on ADC nonlinear chaotic map, which has been widely investigated, can be a desirous alternative for the generation of TRNs. Literature [1] proposed the use of SAR ADC and dynamic residual amplifiers to achieve TRNG. This method reduces the power consumption of the system through selective activation of the fine-SAR ADC by coarse-SAR ADC. Literature [26] proposed a method to realize TRNG by combining resistive thermal noise, oscillator sampling, and discrete time chaotic systems, and its performance is better than the TRNG realized by these three methods alone. In [33–36], multiple ADCs were proposed to realize TRNG using pipeline architecture. In each level, ADC used a resolution of 1.5 bits, and the residual signal output of the previous level becomes its new input. In literature [37], after the completion of the SAR ADC, the comparator was used to continue to compare the lowest bit (residual) of the ADC output once, and the comparison result was regarded as TRNs, so that the analog-to-digital conversion function and the TRNG can be completed at the same time. Pipeline ADC was also adopted in [2] in the realization of TRNG. Compared with [1, 33–35], a dynamic residual amplifier with a gain of less than 2 (1.9 for simulation) was used to avoid system oscillation. In addition, the bit shuffling technique was employed to replace the shift register so as to improve the statistical characteristics of the generated sequence.

The previously mentioned methods of realizing TRNG based on ADC chaotic map are relatively complex to implement and requires special circuit structures. Its application for dedicated chips complexes the design and increases the expense of the system. Literature [25] introduces an approach to generate TRNs by the voltage value of ADC sampling resistance divider circuit using traditional MCU. The approach relies unduly on the resistance and the thermal noise of the circuit. If the thermal noise is low, and the ADC precision is not high enough, the ADC sampling output data shows slight or no changes, which makes it difficult to generate TRNs. Fortunately, literatures [38, 39] present a method to realize TRNG using a structure of combined traditional microprocessor and pipeline ADC. This is a typical example of realizing TRNG based on ADC chaotic map with microprocessors (as shown in Figure 1(c)). However, its structure is too complex to be used in low-power devices.

To solve the above problems, this paper proposes a novel method for TRNG based on ADC nonlinear effect and chaotic map, which can be realized by either custom chips including programmable logic devices with ADC, such as FPGA, or traditional microprocessors such as MCU, DSP, and ARM. An RC circuit and the sensor circuit with RC function are used as the entropy source of TRNG. The processor controls the working state of the entropy source circuit (on or off), so that the entropy source circuit can output varying voltage signal V_{ADC}. The processor then controls the ADC to sample the V_{ADC} to obtain the digital signal D_{ADC}, which is used to generate TRNs. At the same time, the processor delays some time based on D_{ADC}, then continues to control ADC to obtain D_{ADC}, and generate TRNs afterwards. This cycle goes on until enough TRNs are generated. Because of the randomness of circuit noise, D_{ADC} also has a certain randomness, and the nonlinear effect of ADC can further increase the randomness of D_{ADC}. These two factors add great uncertainty to the proposed TRNG. Furthermore, changing input voltage (excluding noise) makes D_{ADC} change as well, and therefore, the sampling interval of ADC also has randomness based on the delay of D_{ADC}. In short, because of the changing input voltage V_{ADC}, random interval sampling further improves the randomness of D_{ADC}. The circuit noise and the nonlinear effect of ADC are used as the entropy source for the proposed TRNG, and the changing input voltage and the random interval sampling resemble a chaotic map, which further accelerates the changing process of the input signal, making D_{ADC} with extremely high randomness.

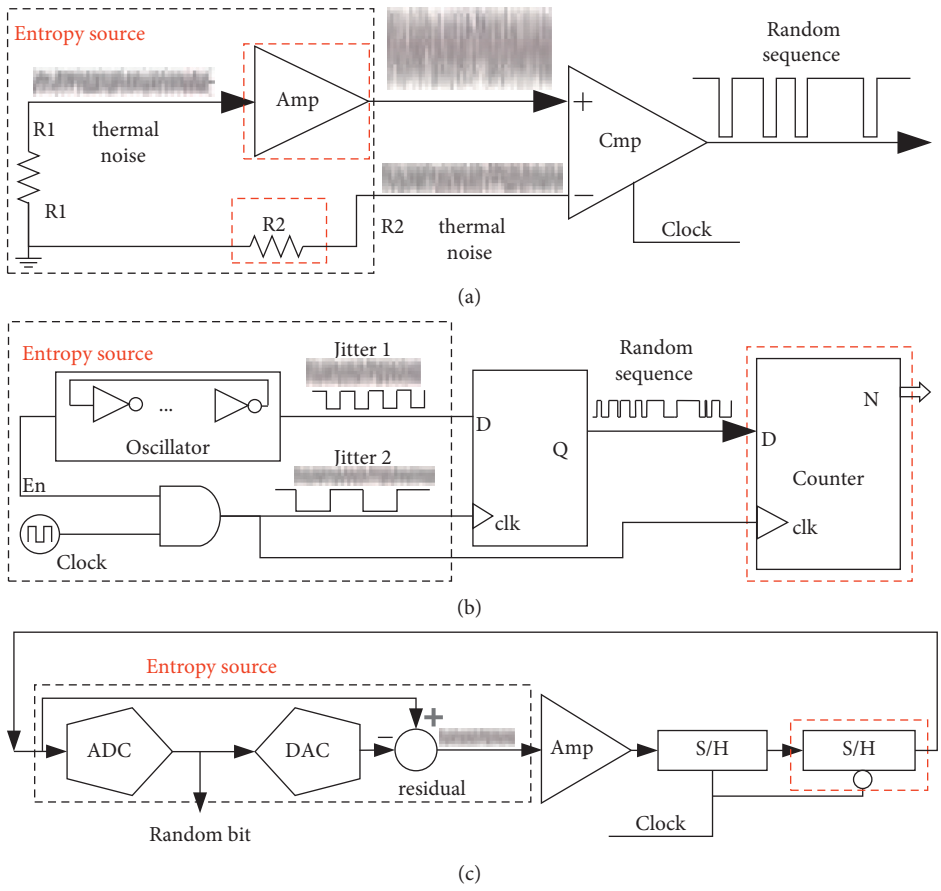

Figure 1: Architecture of TRNG. (a) Comparing the noise, (b) beat frequency detect, and (c) ADC chaotic map.

The main innovation of this paper lies in the proposition of a mechanism with the combination of variable analog input signals and ADC random interval sampling, which is very suitable for low-power application scenarios, especially with WSN nodes. Wireless sensor network nodes need ADC to collect data and directly use the sensor circuit as the entropy source circuit of TRNG, which do not need to add any additional circuit. This results in an enormous reduction on the system design and manufacturing costs, as well as the power consumption.

The rest of this paper is arranged as follows. Section 2 analyzes chaotic systems. Section 3 elaborates in detail the mathematical model and implementation method of TRNG based on ADC. Section 4 introduces the simulation and verification of the proposed mathematical model. Section 5 proposed TRNG implementation and verification. Section 6 discusses TRNG applications. Section 7 concludes the papers with the value and the prospect of the work.

2. Random Analysis of Chaotic Map

In the chaotic map system, even if the initial conditions are slightly different, the system can produce completely different output results after multiple iterations [45]. Among the methods to implement chaotic map, the most widely used is the chaotic map based on 1D linear piecewise affine Markov (PWAM) [34, 46, 47]. To achieve a 1D linear PWAM map, the conditions in equation (1) must be satisfied [34]:

$$X_{n+1} = f(X_n), \quad n = 0, 1, 2, 3, \ldots, \tag{1}$$

where n is the number of time steps (number of iterations), X_0 is the initial state of the system, and X_n is the state of the system after n steps of iteration. And the domain of $f(x)$ is the same as its range. A typical example to further explain how PWAM works can be illustrated as follows, where $f(x)$ is defined as in

$$f(x) = \begin{cases} 3x + 2.00001, & \text{if } -1 \le x < -\frac{1}{2}, \\ x + 1, & \text{if } -\frac{1}{2} \le x < 0, \\ -2x + 1.00001, & \text{if } 0 \le x \le 1. \end{cases} \tag{2}$$

The definition domain of $f(x)$ is within $[-1, 1]$, and 0.00001 in the map can avoid the situation, where the simulation data is always equal to the boundary value (similar to overflow of an actual circuit. It will lead to the degradation of the random quality of the system. Therefore, special treatment is needed to avoid the overflow). In the process of research, two slightly different initial values are used to observe how the output X_n of the PWAM map varies

with the number of iterations. The first operated initial value X_0 is 0.5, and the second $X_0 = 0.50001$. The difference between the two initial values is 0.00001, and the number of iterations of each run is $n = 100$. The results of the two runs are shown in Figure 2.

As can be seen, after 7 iterations, X_n starts to show obvious differences, and the differences become increasingly big as the number of iterations increases. This is sufficient to indicate that, even without noise, due to the limited precision of the system, long time of iterations entails the unpredictable characteristics of PWAM chaotic map, which is exactly what RNs featured by. Specifically, there are three essential criteria for RNs [48]:

(1) It looks random and can pass the random statistical tests.

(2) It is unrepeatable. Its next bit is an uncertainty between being 0 or 1, even if the algorithm that produces the sequence and any already produced number of sequences are known.

(3) It is unrepeatable. The obtained sequence is diverse, even with entirely identical input, under the circumstance of the same algorithm and hardware circuit.

Compared with PRNs, which only require the first of the above criteria to be satisfied, TRNs demand all the listed three criterions to be simultaneously satisfied. Briefly speaking, the next symbol of a random number must be independent of the previously generated symbol, which is similar to a Markov process. This further illustrates that random number generators can be implemented by 1D linear PWAM chaotic map. In terms of PWAM chaotic maps, Bernoulli shift map is one of the most extensively used ones. It can be represented by equation (3), whose definition domain ranges between [0, 2]. When $N_{\text{noise}} = 0$, its corresponding map is shown in Figure 3(a). When the initial value x_0 is different, varying output sequences will be generated. In addition, the system has only two states, S_0 and S_1, and the state jump probability is 0.5, resembling the fair coin toss, as demonstrated in Figure 3(b), which corresponds to a true random system [35].

$$f(x_n) = \begin{cases} 2x_n + N_{\text{noise}}, & \text{if } 0 \le x_n < 1, \\ 2x_n - 2 + N_{\text{noise}}, & \text{if } 1 \le x_n \le 2. \end{cases} \tag{3}$$

What must be noted is that TRNs cannot be achieved using PWAM alone, because PWAM is a deterministic system. In response, an unpredictable initial state for PWAM shall be provided. In real circuits, a common approach to provide the entropy source information for PWAM is to add some analog devices, such as diodes and transistors. The behavior of these analog devices might bring some minor changes under the influence of noise. It happens because PWAM is very sensitive to small inputs. Therefore, the combination of these analog devices and PWAM empowers the system to generate truly unpredictable behaviors, thus achieving TRNG. In equation (3), PWAM based on Bernoulli shift map can be employed to implement TRNG when N_{noise} is not zero, and the noise comes from the circuit.

In the 1D linear PWAM described above, its constraint range (domain) is [−1, 1] and [0, 2]. But in actual circuits, the output can easily exceed the constraint range, because of the influence of noises, which makes it difficult for the chaotic map to return to the normal map range, resulting in the degradation of the randomness quality of the system [49]. However, providing sufficient redundancy for the system state can effectively remove the constraint problem. A prevalent method is to implement Bernoulli shift map to eliminate the constraint problem by ADC [1, 2, 25, 26, 33, 35, 37–39]. However, previously described methods require special integrated circuits [1, 2, 26, 33, 35, 37], the addition of complex circuit structures to microprocessors [38, 39], or the input of noise with statistical characteristics [25]. All these methods would bring great limitations to the application of TRNG. Fortunately, to overcome the above challenges, this paper proposes a common circuit architecture of simple structure and low cost, which can be implemented either on a custom chip or on a traditional microprocessor (with embedded ADC).

3. The Structure of Proposed TRNG

The architecture of the proposed TRNG based on ADC nonlinear effect and chaotic map is shown in Figure 4. It consists of an entropy source circuit and a microprocessor, where the microprocessor includes an ADC, a Memory, a True Random Number Generator Control (TRNGC) module, and configurable pins.

The core of the entropy source circuit is an RC circuit (any other circuits with the same function of RC circuit can also be used), which is used to realize a changeable voltage signal. V_{power} is controlled by TRNGC, and when V_{power} is high (V_{CC}), the RC circuit realizes charging function; when V_{power} is low (Gnd), the RC circuit realizes discharging function. Therefore, by controlling V_{power}, TRNGC controls the RC circuit to produce a constantly changing output voltage. ADC is also controlled by TRNGC in order to sample the produced changing output voltage signal, and because the sampling interval is random, the digital data has strong randomness. The biggest advantage of this architecture is that it can be used as a TRNG and a sensor information collecting (the entropy source circuit seen as the sensor circuit). As a result, when this method is used in sensor equipment of IoT, it is not necessary to add any hardware circuits, because the sensor equipment normally has general-purpose devices such as ADCs and sensors. Therefore, the TRNG proposed in this paper has strong compatibility and can be used in traditional circuit structures.

3.1. TRNG Working Principle. For a single-stage ADC with N-bit rounding-down, when the input signal is within the ideal conversion range, the output is

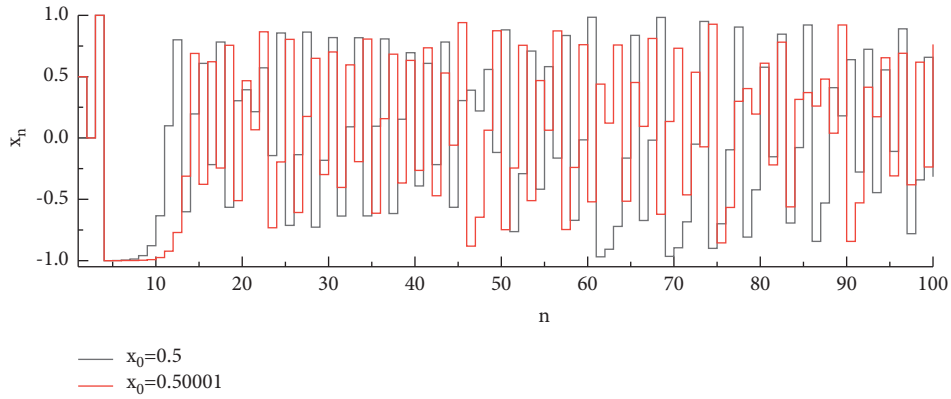

Figure 2: Sensitivity of chaotic maps to initial values.

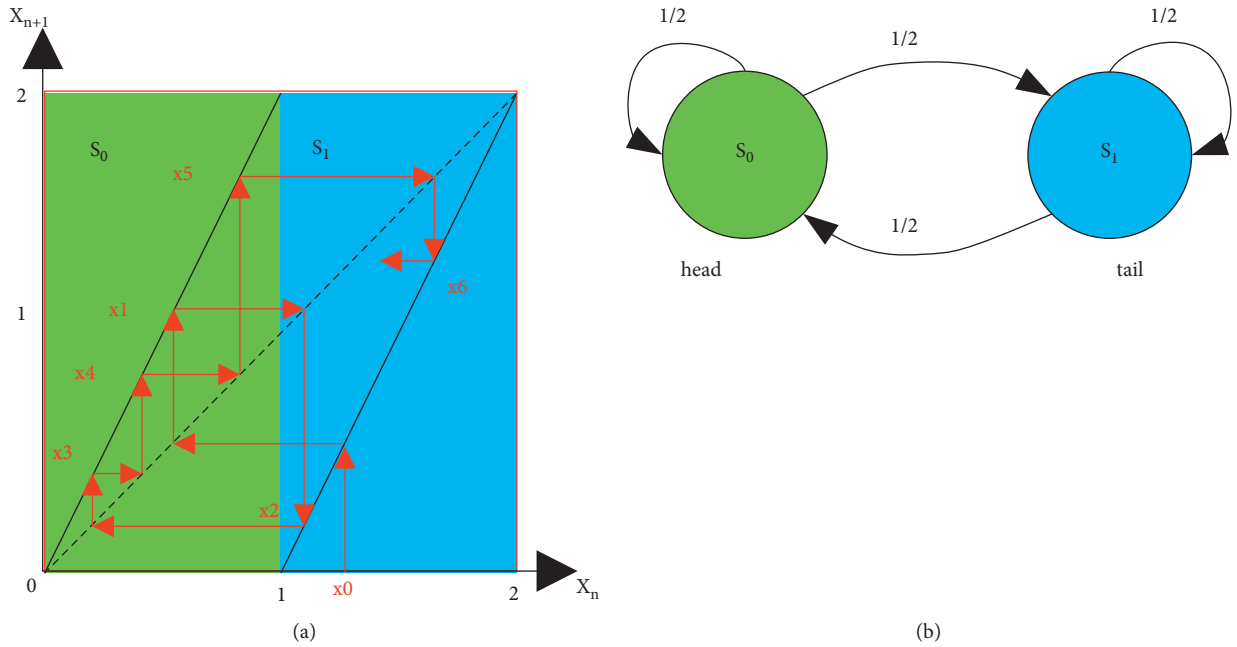

Figure 3: Linear Markov map. (a) Corresponding linear Markov map; (b) Markov chain of the toss of an unbiased coin.

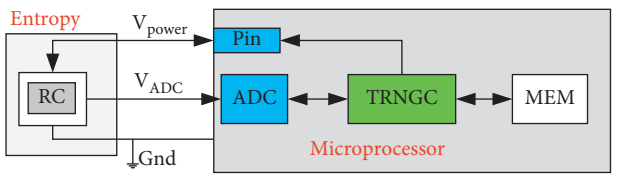

Figure 4: The proposed true number generator based on ADC.

$$D_{ADC} = ADC(V) = \left\lfloor \frac{2^N}{V_{CC}} V_{ADC} \right\rfloor, \quad (4)$$

where D_{ADC} represents the digital output signal after ADC conversion, $ADC(\cdot)$ represents ADC conversion function, $\lfloor \cdot \rfloor$ represents rounding-down operation, V_{CC} represents the reference voltage of ADC, and V_{ADC} represents the input voltage of ADC. Equation (4) is the working principle of traditional ADC, and in this paper, the proposed TRNG implemented is based on traditional ADC, RC circuit, and random interval sampling mechanism. The following equation is used to express the map between the input and output of the entire system:

$$x_{n+1} = M(x_n), \quad (5)$$

where $M(\cdot)$ represents the map function of the proposed system, x_n represents the input signal, and x_{n+1} represents the output signal of the map. Next, the expression of the map function of the proposed TRNG will be derived and discussed.

In Figure 4, assuming that when V_{power} is high or low, the entropy source circuit realizes a simple RC charging or discharging function, and then the output voltage can be calculated as follows:

$$V_{\mathrm{ADC}} = \begin{cases} V_0 + \left(V_{\mathrm{power}} - V_0\right) * \left(1 - e^{-t/\mathrm{RC}}\right) + V_{\mathrm{noise}}, & \text{(Charge)}, \\ V_0 * e^{-t/\mathrm{RC}} + V_{\mathrm{noise}}, & \text{(Discharge)}, \end{cases} \tag{6}$$

where V_{ADC} represents the output voltage of the entropy source circuit, which is also the input voltage of ADC, V_{power} represents the output voltage of MCU pin, RC represents the product of the equivalent resistance and equivalent capacitance of the entropy source circuit, and V_{noise} represents the noise of the circuit. When $t = 0$, $V_{\mathrm{ADC}} = V_{\mathrm{power}} + V_{\mathrm{noise}}$, thus V_{ADC} only changes with noise. When $t \geq 5\mathrm{RC}$, V_{ADC} output is stable, and if it is a charging process, $V_{\mathrm{ADC}} = V_{\mathrm{power}} + V_{\mathrm{noise}}$; if it is a discharging process, $V_{\mathrm{ADC}} = V_{\mathrm{noise}}$. The entropy source circuit can be treated as a resistor divider circuit when $t = 0$ or $t \geq 5\mathrm{RC}$ (resistance is infinitely large and infinitely small). However, the entropy source information of the system input is only decided by noise [25], causing weak randomness (the traditional low precision ADC is hard to identify noises). Therefore, in order to improve the randomness, it is necessary to ensure that, during the ADC sampling process, the charge and discharge states must be switched for every 5RC duration. Another problem is that the value of RC changes in the actual circuit; thus, the charging and discharging time cannot be accurately controlled. Therefore, in order to prevent V_{ADC} from remaining a stable state, two thresholds are set: a high threshold (D_{HT}) and a low threshold (D_{LT}). When V_{ADC} surpasses D_{HT}, the RC circuit begins to discharge, and when V_{ADC} reaches a value below D_{LT}, the RC circuit begins to charge. The following equation can be used to express the state of the system after a long-time operation:

$$V_{\mathrm{ADC}}^{k+1} = \begin{cases} V_{\mathrm{ADC}}^{k} + \left(V_{\mathrm{ADC}}^{\mathrm{power}} - V_{\mathrm{ADC}}^{k}\right) * \left(1 - e^{-t/\mathrm{RC}}\right) + V_{\mathrm{ADC}}^{\mathrm{noise}}, & \text{(Charge)}, \\ V_{\mathrm{ADC}}^{k} * e^{-t/\mathrm{RC}} + V_{\mathrm{ADC}}^{\mathrm{noise}}, & \text{(Discharge)}, \end{cases} \tag{7}$$

where k represents the number of sampling times. From (7), it can be seen that V_{ADC}^{k+1} is affected by noise, as well as k and t. In other words, after the RC circuit starts charging or discharging, even if the initial V_{ADC} is unchanged (noise is ignored), the randomness of V_{ADC}^{k+1} can be improved through TRNG by controlling the value of V_{power} and ADC sampling interval (time t). Additionally, with the increase of the number of iteration times k, a completely different data set V_{ADC}^{k+1}, $k > 0$ can be obtained, making the system have certain chaotic map characteristics. The microprocessor then converts V_{ADC}^{k+1} into a digital signal D_{ADC}^{k+1} through ADC and generates TRNs with high randomness using each converted D_{ADC}^{k+1}. Furthermore, in (7), the randomness of the system can be further improved by controlling the charge and discharge conditions (charge and discharge threshold).

3.2. TRNG Implementation. In order to improve the performance of the proposed TRNG to generate TRNs, random numbers are used to generate threshold voltages D_{HT} and D_{LT}. Moreover, a cyclic shift is performed on D_{ADC}^{k+1}, and the lower 4 bits of the shifted data are used to generate TRNGs, which effectively improves the production efficiency of TRNG. The steps of implementation of the proposed TRNG to generate TRNs are as follows:

(1) Firstly, RNs, representing the number of digits of the random numbers to be generated, are determined, V_{power} is set to high, and RN_sum is cleared. Then, TRN_0, representing the last stored true random number, is extracted from a specific address to generate an initial random delay t_0 based on (9).

(2) After the initial random delay, D_{ADC}^{k} (firstly 0) is compared with D_{HT} and D_{LT}. When it is greater than D_{HT}, set V_{power} to 0, and the entropy source circuit starts to discharge; when it is less than D_{LT}, set V_{power} to 1, and the entropy source circuit starts to charge.

(3) Next, the true random number $\mathrm{TRN}_{\mathrm{DADC}}$ is extracted from memory using the lower 8 bits of D_{ADC}^{k} as the relative address. Then, using $\mathrm{TRN}_{\mathrm{DADC}}$, a random interval delay t_r is generated based on (10).

(4) After the random interval delay, V_{ADC} signal is sampled using ADC to obtain digital data D_{ADC}^{k+1}. Based on D_{ADC}^{k+1}, three generated TRNs are extracted from the memory, whose LSBs are then used to form 3-bit data, represented as SBS. Then, an SBS-bit cyclic shift is performed on D_{ADC}^{k+1} to obtain $D_{\mathrm{ADC}}^{\mathrm{SBS}}$, and the last 4 bits are extracted to generate the TRNs

(5) Steps (3) and (4) 4 are repeated four times to obtain a 16-bit TRN before it is written to memory. The address of TRN is automatically added by 1 and copied to a specific address.

(6) Finally, whether to continue generating a new map is determined. If yes, skip to step 2; otherwise, the generated TRN is sent to the application module.

The detailed workflow of the proposed TRNG is shown in Figure 5, where RN_M is the number of cyclic sampling (here set to 4), and RN is the TRN from memory, used to generate D_{HT} and D_{LT}, which enables a changeable threshold function, resulting in effectively improved randomness of TRN.

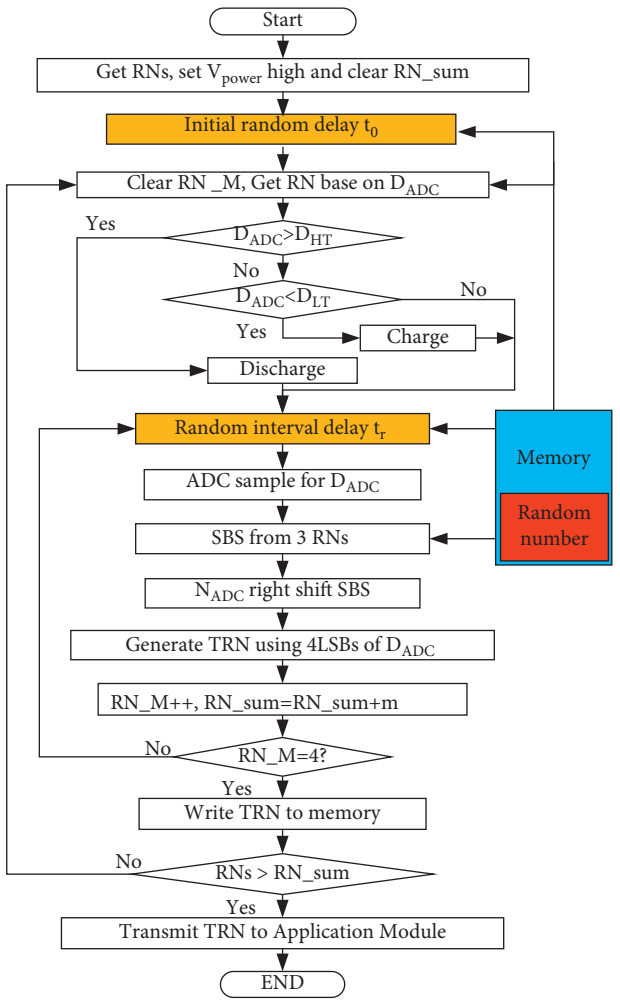

Figure 5: Flow chart for generating map.

3.3. Proposed TRNG Performance Analysis. In Figure 5, the unit time of the initial random delay and random interval delay is the clock cycle of the microprocessor, and the initial random delay t_0 is determined by the last generated TRN. The initial random delay follows the equation

$$t_0 = \frac{1}{f_P}\mathrm{TRN}_0 \,\&\, \mathrm{const1}, \tag{8}$$

where f_P represents the frequency of the microprocessor, & represents the bitwise AND, and const1 represents a constant number. For example, const1 = 15 (indicating F in hexadecimal notation). Therefore, TRN_0 & const1 represents the extraction of the last four digits of TRN_0.

Similarly, the random interval delay follows the equation

$$t_r = \frac{1}{f_P}\mathrm{TRN}_{D_{\mathrm{ADC}}} \,\&\, \mathrm{const2}, \tag{9}$$

where D_{ADC} represents the output signal after ADC sampling, whose lowest 8 bits are used as the relative address to extract the TRN of the corresponding address in the memory (indicated by $\mathrm{TRN}_{\mathrm{DADC}}$), and const2 represents a constant, similar to const1.

Before and after ADC sampling, the microprocessor needs a certain amount of time t_p to process data (determined by the working frequency and the number of clock cycles). Assuming that the microprocessor takes cnt clock cycles in total to process data, then

$$t_P = \frac{\mathrm{cnt}}{f_P}. \tag{10}$$

Therefore, according to (5)~(11), and taking $k = 0$ into consideration, the equation of V_{ADC}^{k+1} with ADC sampling times can be obtained as follows:

$$V_{\mathrm{ADC}}^{k+1} = M\left(V_{\mathrm{ADC}}^{k}\right) = \begin{cases} V_{\mathrm{ADC}}^{\mathrm{power}}\left(1 - e^{-\left(t_0+t_r+t_p\right)/RC}\right) + V_{\mathrm{ADC}}^{\mathrm{noise}}, & \text{if } (*1), \\ V_{\mathrm{ADC}}^{kl} + \left(V_{\mathrm{ADC}}^{\mathrm{power}} - V_{\mathrm{ADC}}^{kl}\right)\left(1 - e^{-\sum_{i=k1+1}^{k}\left(\left(t_r^i+t_s+t_p\right)/(RC)\right)}\right) + V_{\mathrm{ADC}}^{\mathrm{noise}}, & \text{if } (*2), \\ V_{\mathrm{ADC}}^{kh} * e^{-\sum_{i=kh+1}^{k}\left(\left(t_r^i+t_s+t_p\right)/RC\right)} + V_{\mathrm{ADC}}^{\mathrm{noise}}, & \text{if } (*3), \end{cases} \tag{11}$$

where t_s represents the time consumed by ADC to achieve digital-to-analog conversion, $(*1)$ represents $k = 0$, $(*2)$ represents $D_{\mathrm{ADC}}^{kl} < D_{\mathrm{LT}}$ & $D_{\mathrm{ADC}}^{k} < D_{\mathrm{HT}}$, and $(*3)$ represents $D_{\mathrm{ADC}}^{kh} > D_{\mathrm{HT}} \& D_{\mathrm{ADC}}^{k} > D_{\mathrm{LT}} \cdot V_{\mathrm{ADC}}^{kl}$ and V_{ADC}^{kh} represent the input voltage values of ADC when $D_{\mathrm{ADC}}^{kl} < D_{\mathrm{LT}}$ and $D_{\mathrm{ADC}}^{kh} > D_{\mathrm{HT}}$, respectively. Equation (11) shows the map relationship between V_{ADC}^{k+1} and V_{ADC}^{k} (D_{ADC}^{k+1} and D_{ADC}^{k} in the microprocessor), which is similar to 1D linear piecewise affine Markov. Noise V_{noise} directly affects D_{ADC}^{k+1}, and the randomness of D_{ADC}^{k+1} is further improved by V_{noise} through the parameters t_0, t_r, V_{LT} and V_{HT}. Furthermore, from the character of ADC and (4), it can be derived that DADC, the output signal of ADC, has certain nonlinear characteristics and quantization errors, which will also increase the randomness of D_{ADC}^{k+1}. Therefore, using the map in (11) to implement TRNG has more randomness than using the periodic sampling level fixed in ADC [25] (only noise changes).

According to (11), the time required for ADC to sample k times is

$$t = t_0 + k * \left(t_r + t_p + t_s\right). \quad (12)$$

Taking TMS320F2803x, a microprocessor on the market, for example, its sampling frequency can reach 3 MHz, and its main frequency can reach $f_p = 60$ MHz (other microprocessors, such as DSP and ARM, have higher sampling frequencies and main frequencies that can further improve the efficiency of TRNG to generate map). The processing consumes about 60 clock cycles, and after testing, when const1 = const2 = 63, the randomness basically meets the requirements. When the lowest 6 bits of TRN_0 and $\mathrm{TRN}_{\mathrm{DADC}}$ are 1, the time consumed is the longest, which takes 16 clock cycles. Then, the time can be calculated as

$$t = \frac{63}{fp} + k * \left(\frac{63}{fp} + \frac{60}{fp} + \frac{1}{fs}\right) = \frac{63}{60} + k * \left(\frac{63}{60} + 1 + \frac{1}{3}\right). \quad (13)$$

When $k = 1$, the result is 3.43 us, which is 0.29 Mbps. That is, the slowest rate of the generating map is 0.29 Mbps. Similarly, the average rate of generating map is calculated to be 0.42 Mbps (TRN_0 and $\mathrm{TRN}_{\mathrm{DADC}}$ both take half of their maximum value). Every time a 4-bit true random number is generated for each sampling (Section 5 will verify its feasibility), it can be obtained that the proposed TRNG generates a true random number at a rate of about 1.68 Mbps.

4. Simulation and Verification

The sources of randomness of the proposed TRNG in this paper mainly include (1) circuit noise, (2) ADC nonlinearity, (3) random interval sampling, and (4) varying input voltage (noise not included). Among them, circuit noise and ADC nonlinearity add uncertainty to the system, while random interval sampling and varying input voltage provide the system with the characteristics of chaotic map. The combination of the two can achieve high-performance TRNG. Since the ADC nonlinearity is an inherent characteristic of the chip (during simulation, only the quantization error of ADC is considered), we mainly simulate the performance of the proposed chaotic map and the performance of TRNG based on the map.

4.1. Performance Analysis of the Varied Input and Random Intervals. To simplify the analysis, a linear input signal with the slope of 1 is used to replace the RC circuit, and four different situations are simulated in order to analyze the performance of the proposed chaotic map. On the one hand, the four situations are divided into two by the input signal of ADC:

(1) A 1 V constant voltage superimposed with a 1 mV average noise,

(2) A linearly rising voltage with slope is 1 superimposed with a 1 mV average noise.

On the other hand, the four situations are divided into two by the sampling frequency:

(1) ADC performs periodic sampling,

(2) ADC performs sampling at random intervals.

In other words, the 4 simulation situations are as follows: (1) $V_{\mathrm{ADC}} = 1 + \mathrm{noise}$ (without random intervals), (2) $V_{\mathrm{ADC}} = 1 + \mathrm{noise}$ (with random intervals), (3) $V_{\mathrm{ADC}} = t + \mathrm{noise}$ (without random intervals), and (4) $V_{\mathrm{ADC}} = t + \mathrm{noise}$ (with random intervals). The lowest bit of D_{ADC} is used to generate random numbers, and $4 * (10^6)$ bits of data are produced for each simulation situation. Finally, the randomness of the generated random numbers is verified using the U.S. National Institute of Standards and Technology (NIST) [50] test suite, and the results are shown in Table 1 ($P \geq 0.01$ indicates that the test is passed, "Pass" indicates that all subcases pass the test, and "Fail" is the opposite of pass). It can be seen from the table that when $V_{\mathrm{ADC}} = 1 + \mathrm{noise}$, the NIST verification result is very poor regardless of whether random intervals are added between ADC samples. When $V_{\mathrm{ADC}} = t + \mathrm{noise}$, and no random interval is added between ADC samples, the NIST verification result is also very poor, but the results are much better than those of the former situation. Furthermore, when $V_{\mathrm{ADC}} = t + \mathrm{noise}$, by adding random intervals between ADC samples, the results verified by NIST indicate the effectiveness of our proposed chaotic map in improving randomness.

In (1) and (2), because the input signal is fixed, only the circuit noise changes. However, the noise in the circuit is so small that the accuracy of the 12-bit ADC is not enough to sample the noise directly. As a result, the D_{ADC} sampled by ADC is almost fixed. In (3), because the input is changing, the nonlinear effect of ADC can result in a certain degree of randomness in D_{ADC}. From the results, it can be seen that the randomness in (3) is better than that in (1) and (2). However, because the input voltage is linearly changing in (3), it is difficult to drastically change the time interval of ADC sampling just relying on noise and nonlinear characteristics of ADC. Consequently, the interval of each ADC sampling does not change much, resulting in low randomness of D_{ADC}. In (4), a random interval is added, which can further influence the sampling interval based on D_{ADC}, so that the data of each sampled D_{ADC} is completely different, and true random numbers can be generated.

4.2. Performance Analysis of the Proposed TRNG. In the previous section, we have verified that the proposed chaotic map can effectively improve the system's performance in generating random numbers. However, only the LSB output from ADC is used to generate random numbers (1 bit of random number is extracted after each ADC sampling), which is less efficient. In this section, the entropy source circuit of the proposed TRNG can use an RC circuit to obtain more than a simple linear function, and the randomness of the RC output signal is also improved by controlling it to constantly charge and discharge. In addition, this paper proposes to use cyclic shift to process D_{ADC} during postprocessing, and the lowest 4 bits of the processed data are used to generate the true random number, which can improve the efficiency of the TRNG greatly.

Table 1: True random number simulation verification results.

NIST- sts-2.1.2, randomness test	$V_{ADC} = 1 + noise$ (without random interval)		$V_{ADC} = 1 + noise$ (with random interval)		$V_{ADC} = t + noise$ (without random interval)		$V_{ADC} = t + noise$ (with random interval)		Proposed TRNG	
	P value	Prop.	P value	Prop.	P value	P value	Prop.	P value	Prop.	P value
Frequency	<0.01	0/10	<0.01	0/10	<0.01	6/10	0.21331	10/10	0.534146	10/10
Block frequency	<0.01	5/10	<0.01	3/10	0.350485	10/10	0.739918	10/10	0.122325	10/10
Cumulative sums 0	<0.01	0/10	<0.01	0/10	<0.01	6/10	0.739918	10/10	0.122325	10/10
Cumulative sums 1	<0.01	0/10	<0.01	0/10	<0.01	6/10	0.911413	10/10	0.534146	9/10
Runs	<0.01	0/10	<0.01	0/10	0.213309	10/10	0.534146	10/10	0.911413	10/10
Longest run	<0.01	0/10	<0.01	0/10	0.739918	10/10	0.350485	10/10	0.534146	10/10
Rank	0.350485	10/10	0.911413	10/10	0.534146	10/10	0.739918	10/10	0.739918	9/10
FFT	0.350485	10/10	0.122325	10/10	0.739918	10/10	0.911413	10/10	0.213309	10/10
Nonoverlapping template	Fail	Fail	Fail	Fail	Fail	Pass	Pass	Pass	Pass	Pass
Overlapping template	<0.01	3/10	<0.01	4/10	0.739918	10/10	0.739918	10/10	0.534146	10/10
Universal	0.350485	10/10	0.739918	10/10	0.122325	10/10	0.350485	10/10	0.017912	10/10
Approximate entropy	<0.01	0/10	<0.01	0/10	0.350485	10/10	0.739918	10/10	0.534146	10/10
Random excursions	Fail	Fail	Fail	Fail	—	Pass	—	Pass	—	Pass
Random excursions variant	Fail	Fail	Fail	Fail	—	Pass	—	Pass	—	Pass
Serial 0	0.122325	8/10	0.213309	8/10	0.534146	10/10	0.911413	10/10	0.534146	10/10
Serial 1	0.122325	10/10	0.350485	10/10	0.350485	9/10	0.066882	10/10	0.911413	10/10
Linear complexity	0.350485	10/10	0.017912	10/10	0.213309	10/10	0.213309	10/10	0.017912	10/10

In traditional data interaction, most of the methods use integer multiples of bytes for data interaction. In order to achieve generality, integers multiples of 16 bits are generated each time when true random numbers are generated. Since the lowest 4 bits of D_{ADC} of each ADC sampling data are used to generate true random numbers, four times of ADC sampling is required to obtain a true random number of $4 * 4 = 16$ bits. The TRNG represented by (11) is simulated here, and its flow is shown in Figure 5, where m represents the number of ADC cycles, and RNs mean that at least bits true random number is generated each time. In Figure 5, sets $m = 4$ and RNs = 16, and the simulation algorithm is shown in Algorithm 1.

Under the same initial conditions, which means that the input V_{ADC} of ADC is 0, and the $D^0_{ADC}[3:0]$ obtained from the first sampling of ADC are 0 and 3, respectively. Two simulations are conducted on Algorithm 1, each iterated 100 times. Figure 6 shows the lowest 4 bits of D_{ADC}. It can be seen from the figure that the output data of the two simulations are different, which implies that the proposed TRNG architecture has nonrepeatable characteristics. That is, even if the initial conditions are the same, D_{ADC} will be completely different due to circuit noise and ADC nonlinear characteristics. Furthermore, the simulation generated a $4 * (10^6)$ bits random number, and the random numbers are verified using NIST. The verification results are shown in the "proposed TRNG" column in Table 1, which suggests that the proposed TRNG has good performance on the randomness of its output.

It can be seen from the simulation results of Section 4.1 and Section 4.2 that the changing input voltage and random interval sampling have the characteristics of chaotic map. Moreover, combining them with circuit noise and ADC nonlinearity can achieve high-performance and high-efficiency TRNG.

5. Implementation and Validation

The structure of the proposed TRNG in this paper is very simple and has strong compatibility. It is especially suitable for sensing equipment, in which the sensor circuit can be directly used as the entropy source circuit without adding any additional circuit. In order to demonstrate its compatibility and advantages in the field of WSN for the IoT, we implemented two proposed TRNG based on RFID tags of separated components:

(1) The entropy source circuit adopts a pure RC circuit, which is a general structure for the proposed TRNG. RC can be adjusted freely to improve the performance of the proposed, and its structure is shown in Figure 7(a).

(2) The entropy source circuit adopts sensor circuit, which is a special structure by the proposed TRNG and is mainly used in sensing devices of the IoT. No additional circuit is needed, which greatly reduces the cost of TRNG, and its structure is shown in Figure 7(b).

The TRNG, whose entropy source is based on RC circuit, is specially used to generate true random numbers (no other functions). The value of RC can be arbitrarily adjusted in order to get TRNG with good performance. The TRNG based on sensing circuit as entropy source is generally used in scenarios compatible with sensor functions. In such case, the sensor circuit is mainly used for sensing functions, while the TRNG is an incidental function, which can generate true random numbers without occupying any hardware resources, and with lower cost and simpler design.

In this paper, the microprocessor MSP430 (embedded 12-bits ADC) [51] is used to implement the RFID Protocol and realize the software control of the TRNG. LDO provides

Input: m, RNs, const1, const2, SNR = 10
Output: TRNs
(1) Set V_{power} high, for charging input voltage V_{ADC}
(2) Get TRN_0 and calculate t_0 based on (8)
(3) Dealy_function (t_0) for initial delay
(4) Get RN base on D_{ADC}
(5) Calculate V_{HT} and V_{LT} based on RN, which is compared with D_{ADC} for charge or discharge V_{ADC}
(6) Get TRN_{DADC} based on D_{ADC}, and calculate t_r based on (9)
(7) Dealy_function (t_r) for random delay
(8) ADC samples, $D_{ADC} = \lfloor 2^N V_{in}/V_{cc} \rfloor$
(9) Get three RNs for SBS, and N_{ADC} right shift SBS bits
(10) Extract 4 LSBs from shifted N_{ADC} for generating TRN
(11) **If** RN_M = m **Then**
Write TRN to memory and Jump to step 12
Else
Jump to step 6
End If
(12) **If** RNs > RN_sum **Then**
Exit
Else
Jump to step 4
End If

Algorithm 1: TRNG simulation algorithm based on ADC nonlinear effect and chaotic map.

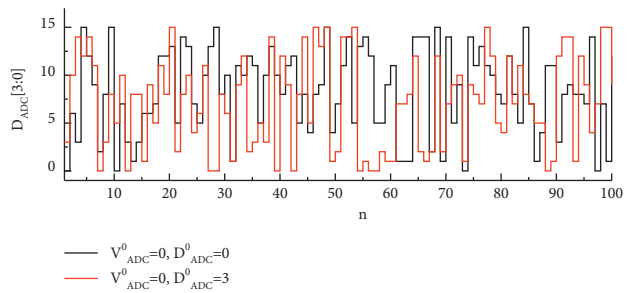

Figure 6: Comparison of D_{ADC} [3 : 0] between two simulations.

a stable voltage to the sensor circuit, AMP is used to amplify the output signal of the sensor, TRNGC is the control module, and MEM is the built-in memory of the MSP430.

The hardware object of the proposed TRNG is shown in Figure 8, Figures 8(a) and 8(b) show the general RC structure and the sensor circuit for entropy source circuit, respectively. Here, Figure 8(b) is a special structure of the proposed TRNG, which can not only realize TRNG, but also realize the function of information sensing. Moreover, the true random number can be used to increase the reliability of the encryption algorithm, so as to further improve the communication security of wireless sensor network.

For TRNG based on sensor circuit, because the adjustable range of sensor circuit is small in order to realize sensing function, we use 1 bit of each ADC sample to realize the TRN, such as the 1st LSB, 2nd LSB, 3rd LSB, and 4th LSB of ADC output. In each case, random numbers of $4^*(10^6)$ bits are generated. The NIST verification results are shown in the column of the proposed TRNG (sensor circuit) in Table 2. It can be seen from the test results that the random numbers generated by the four situations have strong randomness. The verification results of the Approximate Entropy of the random number generated by 4th LSB are not good, which can also indicate that the randomness of the generated random number begins to weaken from the fourth bit of the ADC output. At the same time, we also implemented TRNG based on RC structure and adopted the proposed TRNGC process in III-B. The ADC is sampled once to generate 4 bits of true random numbers (which is more efficient than the sensor structure), and a total of $3 * (10^7)$ bits of random numbers were generated. The results of NIST are shown in the column of the proposed TRNG (RC) in Table 2. It can be seen from the results that the proposed TRNG meets all the requirements of NIST test, indicating that the proposed method can be used to realize TRNG.

Table 3 lists the performance comparison of a variety of TRNGs, as well as their compatibility in mainstream microprocessors such as MCU, DSP, ARM, and FPGA. It can be seen from the table that the proposed TRNG in this paper occupies the least resources, has low power consumption, and is very compatible. That is, it can be implemented in various processors or through simple dedicated chips. The proposed TRNG has great advantages in low power consumption, low cost, miniaturization, and strictly time-required application scenarios. Moreover, it is particularly suitable to be used in wireless sensor network sensor equipment without occupying additional circuit resources.

From the above equations, simulations, test results, and performance comparison, it can be seen that, based on the ADC nonlinear chaotic map method proposed in this paper, it can realize a TRNG, and compared with other existing

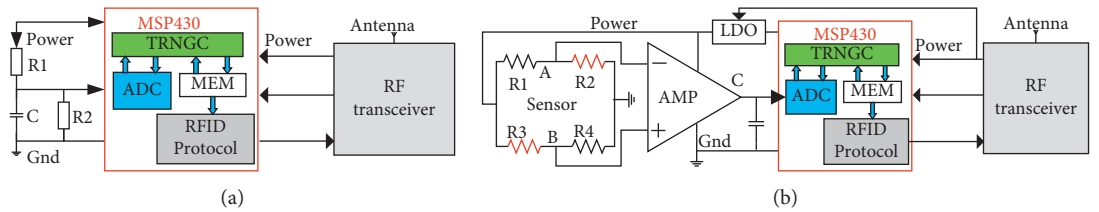

Figure 7: The structure of the proposed TRNG. (a) The general structure of proposed TRNG. (b) A special general structure of proposed TRNG.

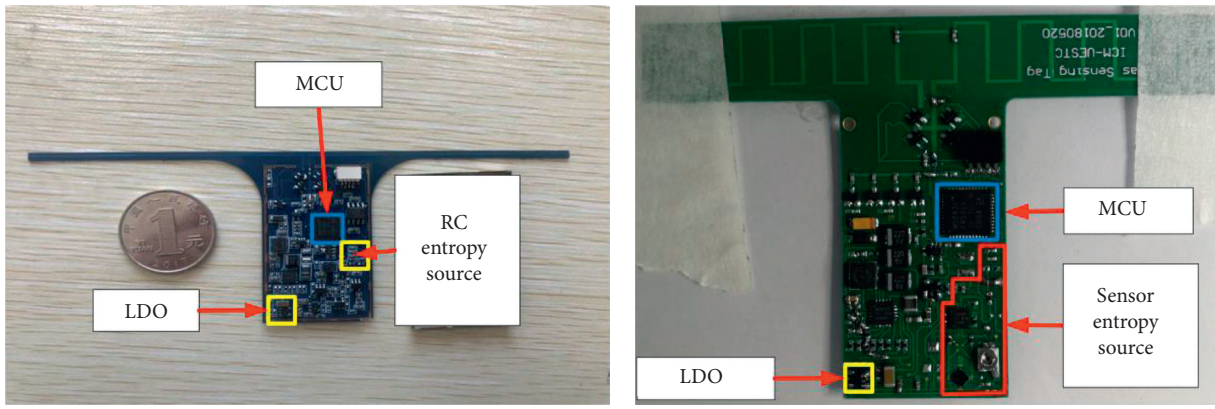

Figure 8: The TRNG is based on RFID sensor tag. (a) Entropy source is RC circuit. (b) Entropy source is sensor circuit.

Table 2: True random number test results.

NIST- sts-2.1.2, randomness test	Proposed TRNG (sensor circuit)								Proposed TRNG (RC)	
	1st LSB		2nd LSB		3rd LSB		4th LSB		1st~4th-LSBs	1st LSB
	P value	Prop.	*P* value	Prop.	*P* value	*P* value	Prop.	*P* value	Prop.	*P* value
Frequency	0.739918	1	0.964295	1	0.637119	1	0.534146	1	0.637119	1
Block frequency	0.122325	0.8	0.739918	1	0.534146	1	0.911413	1	0.162606	1
Cumulative sums 0	0.122325	1	0.739918	1	0.437274	1	0.534146	1	0.911413	1
Cumulative sums 1	0.534146	0.9	0.437274	1	0.534146	1	0.739918	1	0.213309	1
Runs	0.534146	0.9	0.275709	1	0.437274	1	0.213309	1	0.017912	0.95
Longest run	0.350485	1	0.637119	0.95	0.964295	1	0.350485	1	0.834308	1
Rank	0.534146	1	0.834308	1	0.911413	0.9	0.122325	1	0.964295	1
FFT	0.534146	1	0.739918	1	0.534146	1	0.739918	1	0.025193	1
Nonoverlapping template	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
Overlapping template	0.350485	1	0.834308	1	0.350485	0.9	0.739918	1	0.534146	1
Universal	0.911413	0.9	0.213309	1	0.834308	1	0.350485	1	0.035174	0.95
Approximate entropy	0.122325	1	0.122325	1	0.437274	1	<0.01	0.9	0.035174	0.9
Random excursions	—	Pass	—	Pass	—	Pass	—	Pass	Pass	Pass
Random excursions variant	—	Pass	—	Pass	—	Pass	—	Pass	Pass	Pass
Serial 0	0.534146	1	0.637119	1	0.162606	1	0.739918	1	0.048716	1
Serial 1	0.534146	1	0.090936	0.95	0.739918	1	0.350485	1	0.275709	1
Linear complexity	0.066882	1	0.437274	1	0.213309	1	0.350485	1	0.739918	1

TRNG, it has great advantages in terms of low power consumption, low cost, and strong compatibility.

6. Discussion

The proposed TRNG in this paper has the characteristics of low power consumption, low design complexity, and strong compatibility, which can be very convenient to be used in the field of security encryption and anticollision, especially in the passive sensor tags of the Internet of Things, which has the lowest power consumption.

The proposed TRNG is used in secure encryption to improve secure communication performance of IoT. Taking an encryption algorithm for example, an initial F operation

Table 3: Performance comparison.

Parameter	This work	[25]	[34]	[12]	[43]	[40]	[41]
Resources	RC	R	Multi ADCs	836 μm^2	141~2387 LUT-FF	300 ROs	5 CLBs
Power level	mA	mA	μA	μA	A	A	A
MCU	Yes	Yes	No	No	No	No	No
DSP	Yes	Yes	No	No	No	No	No
ARM	Yes	Yes	No	No	No	No	No
FPGA	Yes	Yes	No	No	Yes	Yes	Yes
Chip	Yes	Yes	Yes	Yes	Yes	Yes	Yes
NIST test	Pass	N/A	Pass	Pass	Pass	Pass	Pass

*RC is resistance and capacitance circuits; R is resistance. LUT-FF is Look Up, RO is ring oscillator, and CLB is configurable logical block.

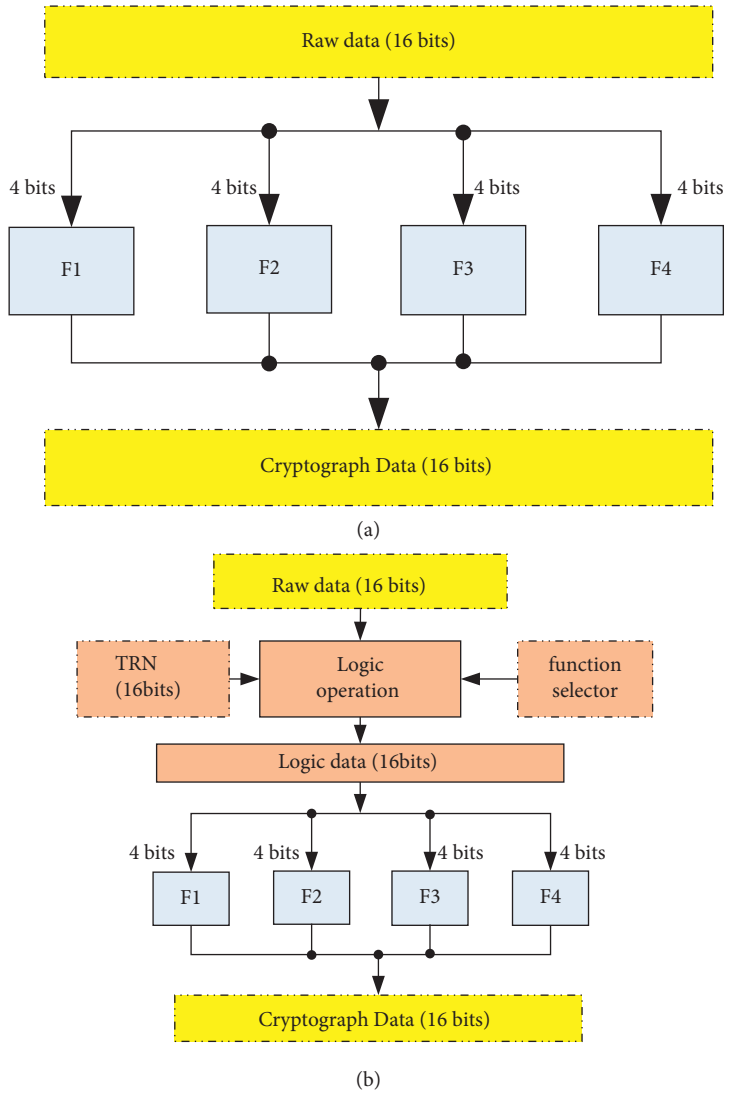

Figure 9: F operation structure: (a) original F operation; (b) improved F operation.

is shown in Figure 9(a), where the F operation divides the input 16-bit data into four 4 bits and then performing four sub-F operations.

In order to improve the performance of the encryption algorithm, the input data and 16-bit TRNs are combined to perform a simple logical operation before F operation, as shown in Figure 9(b), where 16-bit TRNs are generated by the proposed TRNG. TRN performs logical operation with 16-bit raw data to generate logic data, and function selection is used to select different logical operation operations. For example, if the logical operation is XOR operation, when TRN $\neq 0$, logic data is different from raw data, which will result in cryptograph data being completely different from the original cryptograph data. However, when TRNs = 0, the

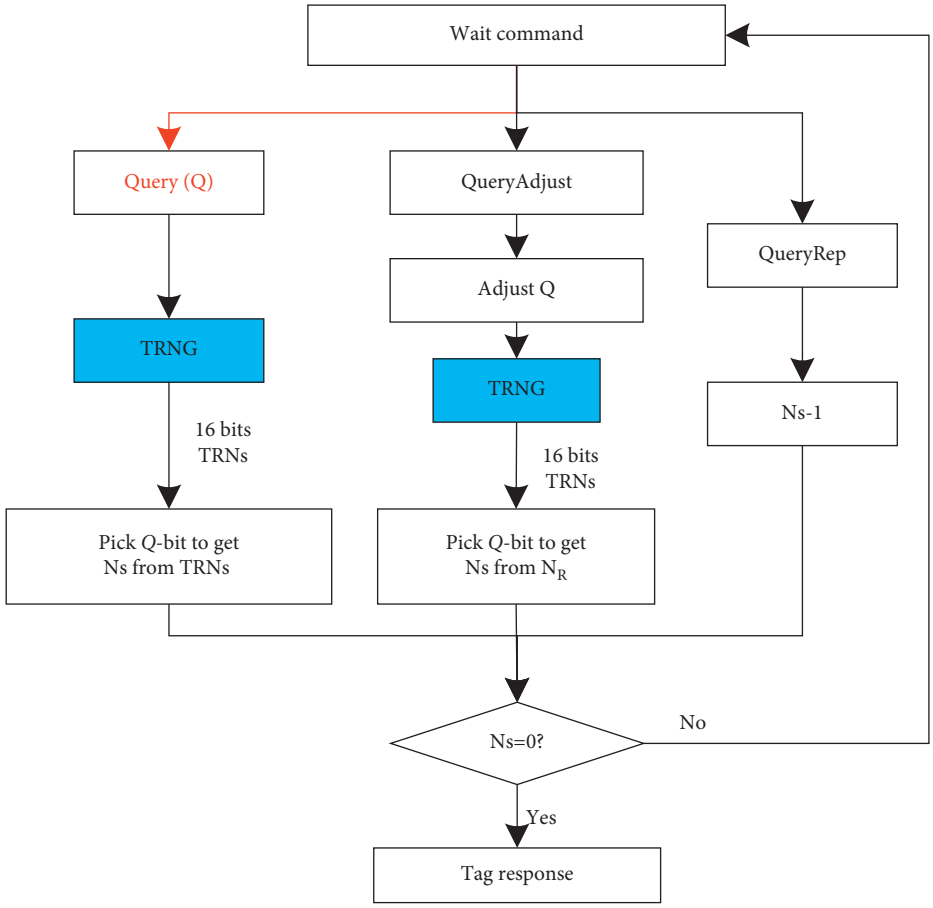

Figure 10: Structure of anticollision algorithm based on the proposed TRNG.

original data is the same as logic data (raw data XOR 0). Therefore, the improved encryption algorithm in Figure 9(b) can not only improve the encryption performance, but also be compatible with the original encryption algorithm, with huge flexibility.

The proposed TRNG can also be used in the anticollision field of RFID to improve the efficiency of multitag identification. For example, in ISO/IEC 18000-6 Type C Standard [52], the tag needs to implement pseudorandom number/true random number to implement the anticollision algorithm based on Q value, as shown in Figure 10.

The commands related to the anticollision algorithm include Query, QueryAdjust, and QueryRep. At the beginning of each inventory, the reader needs to send Query command to determine an initial Q value, and the tag uses the proposed TRNG in this paper to generate a 16-bit random number and intercept the Q-bits generating Ns. Finally, whether to return data can be determined according to whether the intercepted Q-bits data is 0, and when the intercepted Q-bits data is not zero, the reader needs to send QueryAdjust and QueryRep commands to control the tag to return the response data. Meanwhile, in the RFID protocol, QueryRep command has the least bits, so its time is the shortest but greater than 25 μs. Since the proposed TRNG can generate true random numbers at a rate greater than 1.68 Mbps, the proposed TRNG can generate at least 42 bits of true random number in 25 μs. Also, in RFID protocol, there is a requirement to delay between T1 and T2, which can also be used to generate more TRNs with TRNG. As a result, the process meets the requirement to produce multiple RNGs needed for RFID communication.

7. Conclusion

This paper introduces the feasibility of using ADC to realize TRNG and analyzes the shortcomings of existing TRNG based on ADC. A novel TRNG based on ADC nonlinear effect and chaotic map is proposed, which can be realized by using traditional processors with ADC. When the ADC sampling frequency in the processor is 3 MHz, and the main frequency is 60 MHz, the proposed TRNG can generate TRNs at a rate of about 1.68 Mbps. The proposed TRNG for sensor tag does not need any additional circuit, which greatly reduces the cost and power consumption of the system. The simulation results show that the proposed structure can effectively improve the randomness of the system. From the test results of the two proposed TRNG, it can be seen that the proposed TRNG not only improves the versatility of the ADC-based TRNG, but also reduces the complexity of the system design, and therefore, it has a very high practical value. In future work, the proposed TRNG in this paper can

be integrated into the RFID technology-based sensor tag (chip), which can speed up the construction of communication security in the IoT.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors of this paper declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Key R&D Program (2018YFB1802102 and 2018AAA0103203), in part by the Ministry of Education-China Mobile Fund Program (MCM20180104), in part by National Natural Science Foundation of China (61971113 and 61901095), in part by the Guangdong Provincial Research and Development Plan in Key Areas (2019B010141001 and 2019B010142001), in part by the Sichuan Provincial Science and Technology Planning Program (2020YFG0039, 2021YFG0013, and 2021YFH0133), in part by the Yibin Science and Technology Program—Key Projects (2018ZSF001 and 2019GY001), in part by the Grant SCITLAB-0010 and SCITLAB-100021 of Intelligent Terminal Key Laboratory of Sichuan Province, and in part by the Fundamental Research Funds for the Central Universities (YGX2019Z022).

References

[1] M. Kim, U. Ha, K. J. Lee, Y. Lee, and H.-J. Yoo, "A 82-nW chaotic map true random number generator based on a subranging SAR ADC," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 7, pp. 1953–1965, Jul. 2017.

[2] A. Gerosa, R. Bernardini, and S. Pietri, "A fully integrated chaotic system for the generation of truly random numbers," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 7, pp. 993–1000, Jul. 2002.

[3] A. Alabdulkarim, M. Al-Rodhaan, Y. Tian and Abdullah Al-Dhelaan, and A. Al-Dhelaan, "A privacy-preserving algorithm for clinical decision-support systems using random forest," *Computers, Materials & Continua*, vol. 58, no. 3, pp. 585–601, 2019.

[4] Z. Zhou, Q. M. J. Wu, Y. Yang, and X. Sun, "Region-level visual consistency verification for large-scale partial-duplicate image search," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 2, pp. 1–25, 2020.

[5] I. V. Chugunkov, M. A. Ivanov, E. A. Gridneva, and N. Y. Shestakova, "Classification of pseudo-random number generators applied to information security," in *Proceedings of the 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 370–373, St. Petersburg and Moscow, Russia, February 2017.

[6] S. Yasuda, K. Uchida, T. Tanamoto, R. Ohba, and S. Fujita, "Ultra-small physical random number generators based on Si nanodevices for security systems and comparison to other large physical random number generators," in *Proceedings of the 2003 Third IEEE Conference on Nanotechnology, 2003. IEEE-NANO 2003.*, vol. 2, pp. 531–534, San Francisco, CA, USA, August 2003.

[7] T. Zhang, L. Yi, X. Cui et al., "RFID based non-preemptive random sleep scheduling in wsn," *Computers, Materials & Continua*, vol. 65, no. 1, pp. 835–845, 2020.

[8] J. Su, Z. Sheng, A. X. Liu, Z. Fu, and C. Huang, "An efficient missing tag identification approach in RFID collisions," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[9] Z. Zhou, Y. Mu, and Q. M. J. Wu, "Coverless image steganography using partial-duplicate image retrieval," *Soft Computing*, vol. 23, no. 13, pp. 4927–4938, 2019.

[10] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Redundant rule detection for software-defined networking," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2735–2751, 2020.

[11] Pangratz and Weinrichter, "Pseudo-random number generator based on binary and quinary maximal-length sequences," *IEEE Transactions on Computers*, vol. C-28, no. 9, pp. 637–642, Sep. 1979.

[12] K. Yang, D. Blaauw, and D. Sylvester, "An all-digital edge racing true random number generator robust against PVT variations," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 4, pp. 1022–1031, April 2016.

[13] D. Li, Z. Lu, X. Zou, and Z. Liu, "PUFKEY: a high-security and high-throughput hardware true random number generator for sensor networks," *Sensors*, vol. 15, no. 10, pp. 26251–26266, Oct. 2015.

[14] Y. Ma, T. Chen, J. Lin, J. Yang, and J. Jing, "Entropy estimation for ADC sampling-based true random number generators," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 11, pp. 2887–2900, Nov. 2019.

[15] A. Hady, "Duty cycling centralized hierarchical routing protocol with content analysis duty cycling mechanism for wireless sensor networks," *Computer Systems Science and Engineering*, vol. 35, no. 5, pp. 347–355, 2020.

[16] H. Zhu, D. Gao, and S. Zhang, "A perceptron algorithm for forest fire prediction based on wireless sensor networks," *Journal on Internet of Things*, vol. 1, no. 1, pp. 25–31, 2019.

[17] S. Kaur and V. K. Joshi, "Hybrid soft computing technique based trust evaluation protocol for wireless sensor networks," *Intelligent Automation & Soft Computing*, vol. 26, no. 2, pp. 217–226, 2020.

[18] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Idle slots skipped mechanism based tag identification algorithm with enhanced collision detection," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 5, pp. 2294–2309, 2020.

[19] Z. Li, B. Chang, S. Wang, A. Liu, F. Zeng, and G. Luo, "Dynamic compressive wide-band spectrum sensing based on channel energy reconstruction in cognitive Internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2598–2607, June 2018.

[20] F. Xiao, W. Liu, Z. Li, L. Chen, and R. Wang, "Noise-tolerant wireless sensor networks localization via multinorms regularized matrix completion," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2409–2419, March 2018.

[21] Z. Li, F. Xiao, S. Wang, T. Pei, and J. Li, "Achievable rate maximization for cognitive hybrid satellite-terrestrial networks with AF-relays," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 2, pp. 304–313, Feb. 2018.

[22] A. de la Piedra, F. Benitez-Capistros, F. Dominguez, and A. Touhafi, "Wireless sensor networks for environmental

research: a survey on limitations and challenges," in *Proceedings of the Eurocon 2013*, pp. 267–274, Zagreb, Croatia, July 2013.

[23] L. Xu, C. Xu, Z. Liu, Y. Wang, and J. Wang, "Enabling comparable search over encrypted data for iot with privacy-preserving," *Computers, Materials & Continua*, vol. 60, no. 2, pp. 675–690, 2019.

[24] B. D. Reddy, V. V. Kumari, and K. Raju, "A new symmetric probabilistic encryption scheme based on random numbers," in *Proceedings of the 2014 First International Conference on Networks & Soft Computing (ICNSC2014)*, pp. 267–272, Guntur, India, Aug. 2014.

[25] L. Jinming, M. Jian, and L. Peiguo, "Design and implement of a MCU based random number generater," in *Proceedings of the 2016 11th International Conference on Computer Science & Education (ICCSE)*, pp. 945–948, Nagoya, Japan, Aug. 2016.

[26] C. S. Petrie and J. A. Connelly, "A noise-based IC random number generator for applications in cryptography," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 5, pp. 615–621, May 2000.

[27] W. T. Holman, J. A. Connelly, and A. B. Dowlatabadi, "An integrated analog/digital random noise source," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, no. 6, pp. 521–528, Jun. 1997.

[28] Q. Tang, B. Kim, Y. Lao, K. K. Parhi, and C. H. Kim, "True Random Number Generator circuits based on single- and multi-phase beat frequency detection," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, pp. 1–4, San Jose, CA, USA, Sep. 2014.

[29] N. Nalla Anandakumar, S. K. Sanadhya, and M. S. Hashmi, "FPGA-based true random number generation using programmable delays in oscillator-rings," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 3, pp. 570–574, Mar. 2020.

[30] N. Fujieda, M. Takeda, and S. Ichikawa, "An analysis of DCM-based true random number generator," *IEEE Trans. Circuits Syst. II*, 2020.

[31] Y. Liu, R. C. C. Cheung, and H. Wong, "A bias-bounded digital true random number generator architecture," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 1, pp. 133–144, Jan. 2017.

[32] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smartcard IC," *IEEE Transactions on Computers*, vol. 52, no. 4, pp. 403–409, Apr. 2003.

[33] S. Callegari, R. Rovatti, and G. Setti, "Reconfigurable ADC/True-RNG for secure sensor networks," in *Proceedings of the IEEE Sensors*, pp. 1072–1075, Irvine, CA, USA, November 2005.

[34] S. Callegari, R. Rovatti, and G. Setti, "Embeddable ADC-based true random number generator for cryptographic applications exploiting nonlinear signal processing and chaos," *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 793–805, Feb. 2005.

[35] F. Pareschi, G. Setti, and R. Rovatti, "Implementation and testing of high-speed CMOS true random number generators based on chaotic systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 12, pp. 3124–3137, Dec. 2010.

[36] S. Callegari and G. Setti, "ADCs, chaos and TRNGs: a generalized view exploiting Markov chain lumpability properties," in *Proceedings of the 2007 IEEE International Symposium on Circuits and Systems*, pp. 213–216, New Orleans, LA, May 2007.

[37] A. Jayaraj, N. N. Gujarathi, I. Venkatesh, and A. Sanyal, "0.6V-1.2V, 0.22pJ/bit true random number generator based on SAR ADC," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, p. 1, 2019.

[38] M. Fabbri and S. Callegari, "Very low cost entropy source based on chaotic dynamics retrofittable on networked devices to prevent RNG attacks," in *Proceedings of the 2014 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 175–178, Marseille, France, December 2014.

[39] S. Callegari, M. Fabbri, and A. Beirami, "Very low cost chaos-based entropy source for the retrofit or design augmentation of networked devices," *Analog Integrated Circuits and Signal Processing*, vol. 87, no. 2, pp. 155–167, May 2016.

[40] F. Kodytek and R. Lorencz, "A design of ring oscillator based PUF on FPGA," in *Proceedings of the 2015 IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits & Systems*, pp. 37–42, Belgrade, Serbia, Apr. 2015.

[41] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-based true random number generation using circuit metastability with adaptive feedback control," in *Cryptographic Hardware and Embedded Systems–CHES*, B. Preneel and T. Takagi, Eds., vol. 6917, pp. 17–32, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[42] X. Xu and Y. Wang, "High speed true random number generator based on FPGA," in *Proceedings of the 2016 International Conference on Information Systems Engineering (ICISE)*, pp. 18–21, Los Angeles, CA, USA, Apr. 2016.

[43] I. G. Tarsa, G.-D. Budariu, and C. Grozea, "Study on a true random number generator design for FPGA," in *Proceedings of the 2010 8th International Conference on Communications*, pp. 461–464, Bucharest, Romania, Jun. 2010.

[44] A. Marghescu, P. Svasta, and E. Simion, "Optimising ring oscillator-based true random number generators concept on FPGA," in *Proceedings of the 2016 39th International Spring Seminar on Electronics Technology (ISSE)*, pp. 149–153, Pilsen, Czech Republic, May 2016.

[45] S. Callegari, R. Rovatti, and G. Setti, "First direct implementation of a true random source on programmable hardware," *International Journal of Circuit Theory and Applications*, vol. 33, no. 1, pp. 1–16, Jan. 2005.

[46] T. Stojanovski and L. Kocarev, "Chaos-based random number generators-part I: analysis [cryptography]," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 3, pp. 281–288, Mar. 2001.

[47] G. Setti, G. Mazzini, R. Rovatti, and S. Callegari, "Statistical modeling of discrete-time chaotic processes—basic finite-dimensional tools and applications," *Proceedings of the IEEE*, vol. 90, no. 5, p. 29, 2002.

[48] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Wiley, New York, 1996.

[49] T. Addabbo, M. Alioto, A. Fort, S. Rocchi, and V. Vignoli, "A feedback strategy to improve the entropy of a chaos-based random bit generator," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 2, pp. 326–337, Feb. 2006.

[50] *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, National Institute for Standards and Technology, Gaithersburg, MD, US, Special publication 800-22, 2001.

[51] *Msp430Fr58xx, Msp430Fr59xx, MSP430FR68xx, and MSP430FR69xx Family User's Guide*, Texas Instruments, Dallas, Texas, United States, 2015.

[52] *Information Technology – Radio Frequency Identification for Item Management–Part 63: Parameters for Air Interface Communications at 860 MHz to 960 MHz Type C First Edition, Document*, International Standard, ISO (International Organization for Standardization) and IEC (the International Electrotechnical Commission), Geneva, Switzerland, 2013.