A Unified Methodology for Designing Hardware Random Number Generators Based on Any Probability Distribution

Xiaoming Chen, *Member, IEEE,* Boxun Li, *Student Member, IEEE,* Yu Wang, *Senior Member, IEEE,* Yongpan Liu, *Senior Member, IEEE,* and Huazhong Yang, *Senior Member, IEEE*

***Abstract*—We propose a unified methodology for converting any probability distribution which satisfies a few simple conditions to a uniformly distributed random bit stream. The proposed methodology is based on a bit truncation scheme and can be trivially implemented by basic circuit modules. A sufficient condition is derived to determine the optimal truncation. The proposed methodology is verified by three different true random number generators. Experimental results reveal that the proposed methodology can cope with different types of probability distributions, and the generated random numbers are of high quality and good randomness.**

***Index Terms*—True random number generator, probability distribution, bit truncation, random telegraph noise**

I. Introduction

A TRUE random number generator (TRNG), or hardware random number generator (HRNG), is an apparatus that extracts randomness and generates random numbers from unpredictable physical processes. TRNG is a critical building block in lots of cryptographic/ciphering operations and encryption systems. The quality of random numbers generated from TRNGs greatly affects the security of encryptions.

Typically, a TRNG is composed of three modules: entropy source, harvester, and post-processing. The entropy source provides raw random signals which are extracted from unpredictable physical phenomena, such as device noises [1]–[3], clock jitter [4], [5], metastability [6], [7], chaos [8], [9], the time of device breakdown [10], stochastic behaviors of devices [11], [12], etc. The harvester converts the raw signals to a binary bit stream. Typical harvesters include comparators, analog-to-digital convertors (ADC), counters, etc. The post-processing is usually used to de-bias the bit stream.

Many entropy sources are modeled by normal distributions, such as the white noise [1], the jitter of a free-running oscillator [4], [5], the time of breakdown [10], etc. Actually sampling a normal distribution offers much entropy, but existing TRNGs usually generate only one bit from each sampling, leading to a low entropy utilization. In practice, there are also many non-normal distributions, such as bell-shaped non-normal distributions (e.g., skew normal distributions, log-normal distributions) and exponential distributions which are common in the real world. Except the two-point distribution, non-normal distributions are rarely studied to build TRNGs.

This paper aims to provide a simple and feasible solution to the above two issues. We will develop a unified methodology for designing TRNGs based on any probability distribution which satisfies a few simple conditions. Basically, we need to convert an arbitrary distribution to a uniform distribution. In mathematics, any cumulative distribution function (CDF) is uniformly distributed in $[0, 1]$. However, it is expensive to implement an arbitrary CDF by hardware. Histogram equalization [13] can achieve this goal, but it is a software technique. In addition, the two methods are both sensitive to the distribution details. Due to variations and uncertainties of physical phenomena, the distribution details can vary dramatically.

We make the following major contributions in this paper.

- We propose and prove a unified bit truncation-based methodology for converting any probability distribution to a uniformly distributed random bit stream. It can be trivially implemented by basic circuit modules. We also propose a theoretical sufficient condition to determine the optimal truncation for implementation.
- We propose a race condition-based TRNG and two random telegraph noise (RTN)-based TRNGs to verify the proposed methodology. Their randomness sources follow an irregular bell-shaped distribution, an approximated exponential distribution, and an irregular distribution, respectively.

The rest of this paper is organized as follows. Section II shows the TRNG model used in our methodology. Section III presents the proposed methodology in detail. We study three TRNG cases in Section IV. Section V concludes the paper.

II. TRNG Model

Our proposed methodology is based on a general TRNG model shown in Fig. 1. The entropy source provides a random fluctuation X which obeys a specific probability density function (PDF) $p(x)$. The theoretical domain of definition of $p(x)$ is $(-\infty, \infty)$. In practice, X should have a limited

This work was supported in part by 973 project 2013CB329000, in part by National Natural Science Foundation of China (No. 61373026, 61261160501), in part by Tsinghua University Initiative Scientific Research Program, and in part by Huawei Technologies Co. Ltd.

X. Chen is with the Department of Electronic Engineering, Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084, China and the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA (e-mail: chenxm1986@gmail.com).

B. Li, Y. Wang, Y. Liu and H. Yang are with the Department of Electronic Engineering, Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: lbx13@mails.tsinghua.edu.cn; yu-wang@tsinghua.edu.cn; ypliu@tsinghua.edu.cn; yanghz@tsinghua.edu.cn).

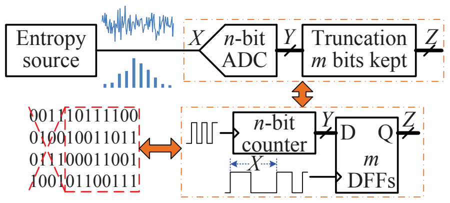

Fig. 1: TRNG model used in this paper.

range (A, B). Assume that $p(x)$ is continuous in (A, B). We assume that there is no correlation between sampled X values, otherwise a non-zero correlation will decrease the randomness of the generated numbers. The harvester converts X into n-bit digital words Y. The post-processing is a bit truncation scheme. After truncation, only the m ($m \leq n$) low-order bits are kept, and the $n - m$ high-order bits are truncated. Let Z be the output words after truncation. Please note that this is an ideal model used to illustrate our idea. The model can be different according to the entropy source. For example, Fig. 1 also shows another model in which the randomness source is the period of a slow and jittery clock. An n-bit counter is driven by a fast clock. The slow and jittery clock triggers m D-type flip-flops (DFF) which are connected to the m low-order bits of the counter. Obviously, the counter plays the role of an ADC, and the fluctuation X is converted to the counter output Y which is sampled by the slow and jittery clock. For the counter-based model, as long as $n \geq m$, n will not affect Z, so we can use $n = m$ in practice.

Let τ be the quantized interval when converting X to Y. For the counter-based model, τ is the clock period of the counter. For the ADC-based model, τ is the quantized level of the ADC. A unified model for converting X to Z is expressed as

$$Y = \left\lfloor \frac{X + R}{\tau} \right\rfloor, Z = Y \% 2^m, \tag{1}$$

where R is an offset. For the counter-based model, R is the value of the previous sampling, because the counter output is accumulated. For the ADC-based model, R is a fixed offset which depends on the design of the ADC.

III. Proposed Methodology

In this section, we will present the proposed methodology in detail. To guarantee high randomness, the probabilities of 0 and 1 should be both 0.5. This condition is equivalent to that all the output words are uniformly distributed, i.e., $P(Z = j) = \frac{1}{2^m}$ $(j = 0, 1, \cdots, 2^m - 1)$. The difficulty to achieve this goal is that $p(x)$ is an arbitrary PDF. However, we will prove that, for any $p(x)$ that satisfies three simple conditions, we can always find a proper m such that this goal is achieved, so the generated numbers are of good randomness. We will also give a theoretical sufficient condition to calculate m.

A. Piecewise Linear Approximation of $p(x)$

For deriving the methodology, $p(x)$ is required to be approximated. Please note that the approximation of $p(x)$ is not an actual step in the implementation. We will only use the approximation to prove the proposed methodology.

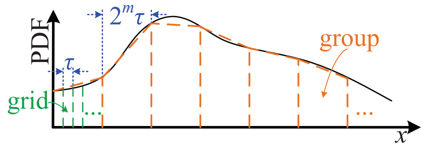

Fig. 2: Piecewise linear approximation of $p(x)$.

Algorithm 1: Finding the maximum group size for PWL approximation of $p(x)$ that has no analytical form.

```
1 $M = H$;
2 for $i = 0$ to $H - 1$ do
3     for $j = i + 1$ to $H - 1$ do
4         for $k = i + 1$ to $j - 1$ do
5             if $\left|\frac{(h[j]-h[i])(c[k]-c[i])}{c[j]-c[i]} + h[i] - h[k]\right| \geq tol \cdot \max_{i \leq l \leq j} h[l]$
              then
6                 $M = \min(M, j - i - 1)$;
7                 $j = H$; //used to break two levels of for-loops
8                 break;
9 $2^m\tau = \frac{M(B-A)}{H}$; $m = \left\lfloor \log_2 \frac{M(B-A)}{H\tau} \right\rfloor$;
```

The entire X-axis is partitioned into infinite intervals with the identical length $2^m\tau$. Each interval is a region $[i2^m\tau, (i+1)2^m\tau]$, where i is the interval number. We call the shape constructed by the PDF curve and the X-axis in each interval as a *group*, as shown in Fig. 2. When $2^m\tau$ is small enough, $p(x)$ can be approximated by a piecewise linear (PWL) function $\phi(x)$ at the granularity of group, so each group can be approximated by a trapezoid. Each group is further partitioned into 2^m *grids* with the identical interval length τ on the X-axis, so each grid is also approximated by a trapezoid. Each group is numbered the same as its corresponding interval's number. All the grids in one group are locally numbered as $0, 1, \cdots, 2^m - 1$ from left to right. Let $S(i, j)$ be the area of grid j in group i. According to the PWL approximation, $S(i, j)$ is approximated by the area of the trapezoid, i.e.,

$$S(i, j) \approx \tau \left[p(i2^m\tau) + \frac{j\Delta_i}{2^m} + \frac{\Delta_i}{2^{m+1}} \right], \\ \Delta_i = p((i+1)2^m\tau) - p(i2^m\tau). \tag{2}$$

Now we consider how to select m to make the PWL approximation accurate. If $p(x)$ is twice differentiable, the maximum difference between $p(x)$ and the approximated PWL function $\phi(x)$ in each interval has a closed form, i.e.,

$$|p(x) - \phi(x)| \leq \frac{1}{8}(2^m\tau)^2 \max |p''(x)|, \\ x \in [i2^m\tau, (i+1)2^m\tau], \forall i. \tag{3}$$

Eq. (3) implies that, if $\max |p''(x)| \ll \frac{1}{\tau^2}$, a proper m can be found such that the PWL approximation has a negligible error. For a PDF which has no analytical form or is not twice differentiable, we develop a heuristic algorithm shown in Algorithm 1 to calculate m such that the PWL approximation error is small. $p(x)$ is expressed by an H-bin histogram. Let $c[i]$ and $h[i]$ be the central value and normalized count of each bin ($0 \leq i < H$), respectively. The total area of all the bins is normalized to 1, so $h[i]$ can be considered as $p(c[i])$, and hence, $p(x)$ is approximated by all the $(c[i], h[i])$ pairs. The goal of Algorithm 1 is to find the largest M, such that for any $0 \leq i < j < H$ and $j - i \leq M$, if successive bins $i, i+1, \cdots, j$ are approximated by a trapezoid constructed by the four vertexes $(c[i], 0)$, $(c[i], h[i])$, $(c[j], h[j])$ and $(c[j], 0)$, the maximum error of $h[i+1], h[i+2], \cdots, h[j-1]$ is less than $tol \cdot \max_{i \leq l \leq j} h[l]$, where tol is a given tolerance (lines 2 to 8). In other words, M found by Algorithm 1 ensures that for any x in any interval $I \in (A, B)$ of length $\frac{M(B-A)}{H}$, $|p(x) - \phi(x)| < tol \cdot \max_{x \in I} p(x)$ always holds. This also means that M is the largest number of bins that can be put into one group to guarantee the accuracy of the PWL approximation. Hence, $m = \lfloor \log_2 (M(B-A)/(H\tau)) \rfloor$ (line 9).

B. Derivation of the Methodology

Considering a specific sampling (so R is fixed in both the ADC- and counter-based models), based on Eq. (1), the probability of observing $Y = i$ is

$$\begin{aligned} P(Y=i) &= \int_{i\tau - R}^{(i+1)\tau - R} p(x)\mathrm{d}x \\ &= S\left(\left\lfloor \frac{i\tau - R}{2^m \tau} \right\rfloor, \left\lfloor i - \frac{R}{\tau} \right\rfloor \% 2^m \right), \end{aligned} \tag{4}$$

where $\%$ is the modulo operator and it always returns a nonnegative integer, e.g., $(-11)\%8 = 5$ but not -3. The probability of observing $Z = j$ $(j = 0, 1, \cdots, 2^m - 1)$ is

$$\begin{aligned} P(Z=j) &= \sum_{k=-\infty}^{\infty} P(Y = j + k2^m) \\ &= \sum_{k=-\infty}^{\infty} S\left(k, \left\lfloor j - \frac{R}{\tau} \right\rfloor \% 2^m \right). \end{aligned} \tag{5}$$

Based on Eq. (2) (i.e., if $p(x)$ can be accurately approximated by a PWL function) and Eq. (5), for any $0 \leq i, j \leq 2^m - 1$, the relative difference between the probabilities of observing i and j in the output words is expressed as (considering that all $P(Z=j)$'s should be close to $\frac{1}{2^m}$)

$$\begin{aligned} 2^m |P(Z=i) - P(Z=j)| &\approx \tau \left| \sum_{k=-\infty}^{\infty} \Delta_k [(i-j)\%2^m] \right| \\ &< 2^m \tau \left| \sum_{k=-\infty}^{\infty} \Delta_k \right| = 2^m \tau |p(B) - p(A)|. \end{aligned} \tag{6}$$

Note that Eqs. (4), (5) and (6) are independent with the specific PDF so they always hold for both cases with or without an analytical PDF. Eq. (6) indicates that, if $|p(B) - p(A)| \ll \frac{1}{\tau}$, we can select a proper m such that the output words are nearly uniformly distributed. Actually, $\frac{1}{\tau}$ is usually large in practice. For bell-shaped PDFs, due to the approximate symmetry, $|p(B) - p(A)|$ is generally close to 0. An exception is when $p(x)$ is monotonous, e.g., the exponential distribution. However, when $\frac{1}{\tau}$ is large enough, the relative difference of Eq. (6) can still be close to 0. Here the PWL approximation error is ignored in Eq. (6). In the next subsection, we will quantize the overall theoretical error and provide a sufficient condition to determine the optimal truncation.

C. Sufficient Condition for Determining the Truncation

Considering the error of the PWL approximation, Eq. (2) can be rewritten by an exact expression:

$$S(i,j) = \tau \left[p(i2^m\tau) + \frac{j\Delta_i}{2^m} + \frac{\Delta_i}{2^{m+1}} + E_{i,j} \right], \tag{7}$$

where $E_{i,j}$ is the error of the PWL approximation in each group. If $p(x)$ is twice differentiable, $E_{i,j}$ is constrained by Eq. (3). Then the theoretical upper bound of Eq. (6) can be recalculated as

$$\begin{aligned} &2^m |P(Z=i) - P(Z=j)| < 2^m \tau |p(B) - p(A)| + \\ &\frac{(2^m\tau)^3}{4} \sum_{k=\lfloor A/(2^m\tau) \rfloor}^{\lceil B/(2^m\tau) \rceil} \max_{k2^m\tau \leq x \leq (k+1)2^m\tau} |p''(x)|. \end{aligned} \tag{8}$$

For a PDF which has no analytical form or is not twice differentiable, we have given an upper bound of the PWL approximation error in the analysis of Algorithm 1. Then, we can get a similar theoretical upper bound of Eq. (6):

$$\begin{aligned} &2^m |P(Z=i) - P(Z=j)| < 2^m \tau |p(B) - p(A)| + \\ &2^{m+1}\tau \cdot tol \cdot \sum_{k=0}^{\lceil \frac{B-A}{2^m\tau} \rceil - 1} \max_{\lceil \frac{k2^m\tau H}{B-A} \rceil \leq l \leq \lfloor \frac{(k+1)2^m\tau H}{B-A} \rfloor} h[l], \end{aligned} \tag{9}$$

where $p(A)$ and $p(B)$ can be obtained from the first bin and last bin of the histogram.

Eq. (8) or (9) give the theoretical upper bound of the relative difference between the probabilities of output words. For an analytical and twice differentiable PDF, m is selected to be the maximum such that the right side of Eq. (8) is less than a given threshold. For other cases, m is selected to be the maximum value in $1, 2, \cdots, \lfloor \log_2 (M(B-A)/(H\tau)) \rfloor$ such that the right side of Eq. (9) is less than a given threshold. This is a sufficient but unnecessary condition, because the upper bound given by Eq. (8) or (9) is the worst case and may not be tight in practice.

D. Summary

We have proved that, for any distribution which satisfies three simple conditions: 1) continuous, 2) can be accurately approximated by a PWL function, and 3) $|p(B) - p(A)| \ll \frac{1}{\tau}$, we can always truncate a few high-order bits from the generated random words such that the remaining words are nearly uniformly distributed, so the randomness is guaranteed. A sufficient condition (i.e., Eq. (8) or Eq. (9)) is provided to calculate the maximum m such that the overall error is less than a threshold. From Eqs. (8) and (9), the maximum m depends on τ. If τ is smaller, m can be larger. Actually, τ can be regarded as the resolution of the system. It in turn means that m depends on the resolution of the system. Due to the physical meaning of $\frac{1}{\tau}$, it is usually large in practice, especially for the counter-based model (e.g., $\frac{1}{\tau} = 10^9$Hz), so it should be easy to satisfy these three conditions in practice.

The circuit may suffer from variations. To reduce the impact of variations, we should select an entropy source with big fluctuations, such as RTN which can cause tens of millivolts in the V_{th} variation [14], the memristor resistance which can vary by more than 100× [11], etc. In addition, we can calculate m

```
void *RaceCondition(void *arg) {
    for (int i=10000; i!=0; --i) ++shared;
    return NULL;
}
```

Fig. 3: Thread processing function of the RACE TRNG.

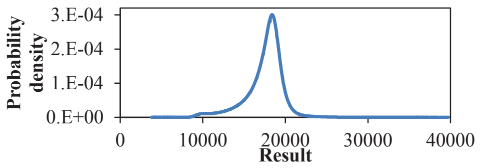

Fig. 4: PDF of the value of the global variable `shared`.

under the worst case such that the circuit can handle runtime variations (e.g., $p(x)$ may vary at runtime). Since raw random signals are converted to digital words in our model, variations have negligible impacts on the digital processing part.

IV. Case Study

We propose three TRNG cases to verify the proposed methodology: a race condition-based TRNG named as RACE and two RTN-based TRNGs named as RTN1 and RTN2. Their randomness sources follow an irregular bell-shaped distribution without an analytical PDF, an approximated exponential distribution, and an irregular distribution with an analytical PDF, respectively. In this section, we will present the design principles and results of the three TRNGs.

A. Proposed TRNGs

1) Race Condition-Based TRNG: The idea of using race conditions to build TRNGs is proposed in [15]. It is actually a software entropy source, but we adopt it to verify the proposed methodology. We run a race condition-based TRNG on two Intel Xeon E5-2690 processors using 16 threads. Each thread executes the identical C code shown in Fig. 3. The global variable `shared` is simultaneously updated by 16 concurrent threads so its final value is unpredictable due to race conditions. We generate 10^8 random results and then apply the proposed methodology. The PDF of the 10^8 values is shown in Fig. 4. It is a bell-shaped non-normal distribution. Algorithm 1 gives $M = 8$ when using $H = 1000$ and $tol = 0.05$. Eq. (9) gives $m = 8$ when the threshold is 0.1.

2) RTN-Based TRNGs: RTN is explained by the trapping/detrapping mechanism caused by oxide traps [16]. RTN causes random fluctuations of V_{th}, I_{ds}, etc. The time-domain of an RTN signal shows a binary fluctuation in response to the capture/emission process of a single trap. The capture time τ_c and emission time τ_e follow exponential distributions [16]:

$$p(\tau_c) = \frac{1}{\bar{\tau}_c}\exp\left(-\frac{\tau_c}{\bar{\tau}_c}\right), p(\tau_e) = \frac{1}{\bar{\tau}_e}\exp\left(-\frac{\tau_e}{\bar{\tau}_e}\right), \quad (10)$$

where the means $\bar{\tau}_c$ and $\bar{\tau}_e$ are called the capture and emission time constants. The RTN-induced fluctuation is compared with a reference voltage to generate a random binary signal which clocks m DFFs, as shown in Fig. 5. We propose two schemes RTN1 and RTN2. In RTN1, both the rising and falling edges of the binary signal can clock the DFFs, so these DFFs are required to be double-edge-triggered. In RTN2, we only use the rising edges to clock conventional rising-edge-triggered DFFs. For RTN2, the fluctuant period $t = \tau_c + \tau_e$ obeys an irregular distribution with an analytical PDF:

$$p(t) = \frac{1}{\bar{\tau}_c - \bar{\tau}_e}\left[\exp\left(-\frac{t}{\bar{\tau}_c}\right) - \exp\left(-\frac{t}{\bar{\tau}_e}\right)\right]. \quad (11)$$

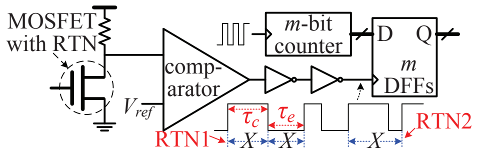

Fig. 5: RTN-based TRNG schemes.

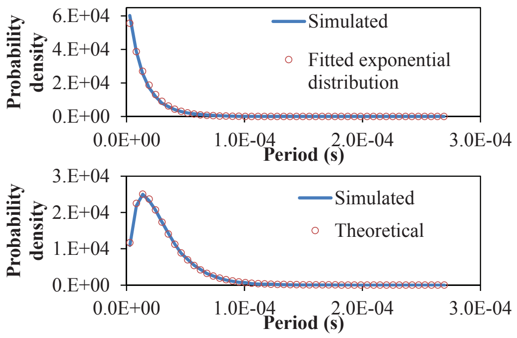

Fig. 6: PDFs of the clock period of the RTN-based TRNGs (Upper figure: for RTN1; Lower figure: for RTN2).

The RTN-based TRNGs are simulated by the method proposed in [17] using a modified Simulation Program with Integrated Circuit Emphasis (SPICE)-based simulator. The transistor shown in Fig. 5 is affected by a single trap whose time constants are $\bar{\tau}_c = 10\mu s$ and $\bar{\tau}_e = 20\mu s$ (in practice, an eligible trap with expected time constants can be selected from a large transistor array [2]). The clock frequency of the counter is 1GHz, i.e., $\tau = 1 \times 10^{-9}$. Fig. 6 shows the PDFs of the fluctuant clock period which is affected by RTN. For RTN1, the simulated PDF can be fitted by an exponential distribution whose mean is $\frac{\bar{\tau}_c+\bar{\tau}_e}{2} = 15\mu s$. For RTN2, the theoretical PDF is given by Eq. (11). By applying Eq. (8) with a threshold of 0.01, we get $m = 7$ and 11 for RTN1 and RTN2, respectively. We generate random numbers of 2 seconds for test.

B. Randomness Test

The test suite provided by the National Institute of Standards and Technology (NIST) [18] is adopted to evaluate the randomness for the three TRNGs, using the m values calculated above. Test results are shown in Table I. All the NIST tests are passed, indicating that the generated random numbers are of good randomness. The biases (i.e., $P(1) - 0.5$, where $P(1)$ is the probability of observing 1 in the bit stream) of the three TRNGs are 2.27×10^{-6}, -3.74×10^{-4} and 9.55×10^{-6}, respectively, which are all very near 0.

C. Comparison on RTN-based TRNGs

Our proposed RTN-based TRNGs are different from two published RTN-based TRNGs [2], [3], which periodically

TABLE I: P-values reported from the NIST test suite. A p-value larger than 0.01 indicates that the test is passed.

Test name	RACE	RTN1	RTN2
ApproximateEntropy	0.924676	0.839692	0.546299
BlockFrequency	0.070922	0.200913	0.597122
CumulativeSums[a]	2/2	2/2	2/2
FFT	0.510140	0.739458	0.509726
Frequency	0.069611	0.469832	0.986952
LinearComplexity	0.579504	0.808437	0.648132
LongestRun	0.599659	0.998850	0.906442
NonOverlappingTemplate[a]	148/148	148/148	148/148
OverlappingTemplate	0.267610	0.214956	0.800599
RandomExcursions[a]	8/8	8/8	8/8
RandomExcursionsVariant[a]	18/18	18/18	18/18
Rank	0.276454	0.716125	0.598541
Runs	0.500683	0.022085	0.602368
Serial[a]	2/2	2/2	2/2
Universal	0.219075	0.608469	0.313213

[a] These tests have multiple p-values, so the reported numbers are the number of passed tests/the total number of tests.

sample the RTN-induced binary fluctuation as the entropy source. The autocorrelation coefficient of an RTN signal is $\rho(f_s) = \exp(-1/(\tau_0 f_s))$, where $\frac{1}{\tau_0} = \frac{1}{\bar{\tau}_c} + \frac{1}{\bar{\tau}_e}$, and f_s is the sampling frequency. f_s must be small enough (e.g., $\frac{1}{2.3\tau_0}$) to guarantee a low autocorrelation (e.g., 10%). Actually, if f_s is too high, successive samplings get the same state so it will generate many all-1 or all-0 long subsequences, which significantly reduces the randomness. In addition, the two existing methods always require a post-processing for de-biasing due to the unequal $\bar{\tau}_c$ and $\bar{\tau}_e$, which further reduces the bit rate. Take the von Neumann corrector [19] adopted by [2] as an example. It can be derived that the theoretical bit rate of the von Neumann corrector is

$$\text{BR}_{\text{vN}} = \frac{\tau_0 f_s}{\bar{\tau}_c + \bar{\tau}_e}(1-\rho(f_s)) < \frac{1}{\bar{\tau}_c + \bar{\tau}_e}. \quad (12)$$

The upper bound is achieved when $f_s = \infty$. If we want to make $\rho(f_s) <10\%$ (i.e., $f_s < \frac{1}{2.3\tau_0}$), $\text{BR}_{\text{vN}} < \frac{1}{2.3(\bar{\tau}_c+\bar{\tau}_e)}$.

Our schemes take the duration time of the binary state as the entropy source, so multiple bits can be generated once the trap changes its state. Since the RTN signal obeys a Poisson process [16], durations of different states are independent. The theoretical bit rates of RTN1 and RTN2 are

$$\text{BR}_{\text{RTN1}} = \frac{2m}{\bar{\tau}_c + \bar{\tau}_e}; \text{BR}_{\text{RTN2}} = \frac{m}{\bar{\tau}_c + \bar{\tau}_e}. \quad (13)$$

Hence, it is guaranteed in theory that RTN1 and RTN2 both have higher bit rates than [2], [3]. In our cases, the simulated bit rates of RTN1 and RTN2 are 466.3kbit/s and 366.4kbit/s, respectively. If we also sample the RTN-induced fluctuation like [2], [3] and use the von Neumann corrector, the simulated bit rate is only 13.0kbit/s when $f_s = \frac{1}{2.3\tau_0} = 65.2$kHz.

V. Conclusions

We have proposed a unified methodology for designing TRNGs based on any probability distribution which satisfies three simple conditions. The proposed methodology is verified by three cases with three representative distributions: a bell-shaped distribution without an analytical PDF, a monotonous exponential distribution, and an irregular distribution with an analytical PDF. The proposed methodology can not only cope with different distributions, but also offer higher entropy utilization/bit rate than some existing designs. By applying the proposed methodology, other common distributions, like normal and log-normal distributions, can also be easily utilized to build TRNGs.

References

[1] N. Göv, M. Mihcak, and S. Ergun, "True Random Number Generation Via Sampling From Flat Band-Limited Gaussian Processes," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 58, no. 5, pp. 1044–1051, May 2011.

[2] R. Brederlow, R. Prakash, C. Paulus, and R. Thewes, "A low-power true random number generator using random telegraph noise of single oxide-traps," in *International Solid-State Circuits Conference (ISSCC)*, Feb 2006, pp. 1666–1675.

[3] C.-Y. Huang, W. C. Shen, Y.-H. Tseng, Y.-C. King, and C.-J. Lin, "A Contact-Resistive Random-Access-Memory-Based True Random Number Generator," *Electron Device Letters, IEEE*, vol. 33, no. 8, pp. 1108–1110, Aug 2012.

[4] V. Rozic, B. Yang, W. Dehaene, and I. Verbauwhede, "Highly efficient entropy extraction for true random number generators on FPGAs," in *Design Automation Conference (DAC)*, June 2015, pp. 1–6.

[5] S. Robson, B. Leung, and G. Gong, "Truly Random Number Generator Based on a Ring Oscillator Utilizing Last Passage Time," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 61, no. 12, pp. 937–941, Dec 2014.

[6] P. Wieczorek and K. Golofit, "Dual-Metastability Time-Competitive True Random Number Generator," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 61, no. 1, pp. 134–145, Jan 2014.

[7] V. Suresh and W. Burleson, "Entropy and Energy Bounds for Metastability Based TRNG with Lightweight Post-Processing," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 62, no. 7, pp. 1785–1793, July 2015.

[8] A. Beirami and H. Nejati, "A Framework for Investigating the Performance of Chaotic-Map Truly Random Number Generators," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 60, no. 7, pp. 446–450, July 2013.

[9] S. Dhanuskodi, A. Vijayakumar, and S. Kundu, "A Chaotic Ring Oscillator based Random Number Generator," in *International Symposium on Hardware-Oriented Security and Trust (HOST)*, May 2014, pp. 160–165.

[10] N. Liu, N. Pinckney, S. Hanson, D. Sylvester, and D. Blaauw, "A true random number generator using time-dependent dielectric breakdown," in *Symposium on VLSI Circuits (VLSIC)*, June 2011, pp. 216–217.

[11] Y. Wang, W. Wen, H. Li, and M. Hu, "A Novel True Random Number Generator Design Leveraging Emerging Memristor Technology," in *Great Lakes Symposium on VLSI (GLSVLSI)*, 2015, pp. 271–276.

[12] S. Balatti, S. Ambrogio, Z. Wang, and D. Ielmini, "True Random Number Generation by Variability of Resistive Switching in Oxide-Based Devices," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 5, no. 2, pp. 214–221, June 2015.

[13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[14] N. Tega, H. Miki, F. Pagette, D. Frank, A. Ray, M. Rooks, W. Haensch, and K. Torii, "Increasing threshold voltage variation due to random telegraph noise in FETs as gate lengths scale to 20 nm," in *Symposium on VLSI Technology (VLSIT)*, June 2009, pp. 50–51.

[15] Y. Yeom, "Generating Random Numbers for Cryptographic Modules Using Race Conditions in GPU," in *Computer Applications for Graphics, Grid Computing, and Industrial Environment*, 2012, pp. 96–102.

[16] M. Kirton and M. Uren, "Noise in solid-state microstructures: A new perspective on individual defects, interface states and low-frequency (1/f) noise," *Advances in Physics*, vol. 38, no. 4, pp. 367–468, 1989.

[17] M. Tanizawa, S. Ohbayashi, T. Okagaki, K. Sonoda, K. Eikyu, Y. Hirano, K. Ishikawa, O. Tsuchiya, and Y. Inoue, "Application of a statistical compact model for Random Telegraph Noise to scaled-SRAM Vmin analysis," in *Symposium on VLSI Technology (VLSIT)*, June 2010, pp. 95–96.

[18] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," DTIC Document, NIST Special Publication 800-22, Tech. Rep., 2001.

[19] J. Von Neumann, "Various Techniques Used in Connection With Random Digits," *National Bureau of Standards Applied Mathematics Series*, vol. 12, pp. 36–38, 1951.