This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2019.2926555, IEEE Transactions on Circuits and Systems II: Express Briefs

An Analysis of DCM-based True Random Number Generator

Naoki Fujieda, *Member, IEEE,* Masaaki Takeda, and Shuichi Ichikawa, *Senior Member, IEEE*

***Abstract*—A TRNG (True Random Number Generator) using DCM blocks of a Xilinx FPGA has advantages of minimal usage of logic elements and tunability of clock frequencies to search for better randomness. Its operating principle is Beat Frequency Detection, which uses the LSBs of a counter that counts the number of successive logic-ones captured by a D flip-flop. In this paper, we evaluate the DCM-based TRNG with 100 frequency pairs. The findings of this work include that (1) only 12 frequency pairs passed the diehard statistical tests when three LSBs were extracted as directed in the previous work, (2) the value of the counter exhibits biphasic histograms, (3) small counter values show less uncertainty, and (4) the number of frequency pairs that passed the tests was doubled to 24 by extracting only one LSB from small counter values, with a 16.9% reduction in the generation rate.**

***Index Terms*—True Random Number Generator, FPGA, Digital Clock Manager, Statistical Tests, Beat Frequency Detection.**

I. Introduction

In cryptographic systems, a true random number generator (TRNG) is an essential component used in key generation and authentication protocols. It extracts physical uncertainty as unpredictable random numbers. Since it is preferred that the amount of additional hardware for a TRNG is as small as possible, many TRNGs that utilize existing hardware elements have been proposed. For example, in FPGA systems, metastability of latches [1] or flip-flops [3] can be used as a source of entropy. Uncertainty in an external chip, such as the read noise of NAND flash [5], is also utilized to produce random numbers.

For Xilinx FPGAs, Johnson et al. [2] proposed a TRNG using two DCM (Digital Clock Manager) blocks. Its operating principle is called Beat Frequency Detection (BFD) [7]. There, two DCMs generate clock signals that have slightly different clock frequencies. They are connected to the data and clock inputs of a D flip-flop (DFF). The output of the DFF becomes a repetition of consecutive logic-zeros and logic-ones. Since the least significant bits (LSBs) of the number of consecutive logic-ones have randomness because of jitter and metastability, they are collected as a source of entropy. An advantage of this DCM-based TRNG is that the number of required logic elements is small. Although two additional DCMs are required, there are very few applications that use up all of the DCMs on an FPGA. Also, since a pair of frequencies can be dynamically reconfigured, the TRNG can be tuned by looking for a pair that shows good randomness. However, the previous work [2] did not show the systematic information to find a good frequency pair, which prevents this type of TRNG from practical applications.

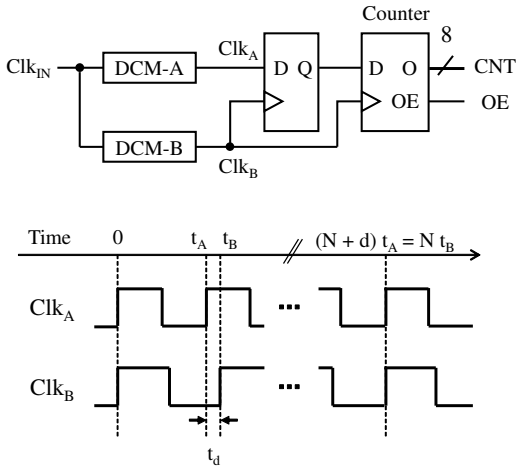

Fig. 1. Block diagram and timing diagram of the DCM-based TRNG [2].

In this paper, we provide a quick way to find a good frequency pair for the DCM-based TRNG. To this end, we evaluate a DCM-based TRNG with 100 pairs of frequencies. The previous work [2] presented 23 pairs and the results of statistical tests were shown only for three pairs of them. This paper quantitatively shows the effect of selection of a pair of frequencies on the quality of random numbers and the generation rate of the TRNG by passing random bit strings for all the pairs to the diehard statistical tests [4]. This gives designers a guideline on which pair of frequencies should be tried at first. We also propose a method to make a finer trade-off between the quality of random numbers and the generation rate, using the fact that the distribution of the number of consecutive logic-ones in the DFF exhibits biphasic histograms. Through the evaluation, we show that the proposed method can greatly increase the number of pairs that give good random numbers with a small penalty on the generation rate.

The rest of this paper is organized as follows. The operating principle of the DCM-based TRNG is explained in Section II. Section III describes our evaluation environment and Section IV presents a preliminary experiment. The proposed method based on the results of the experiment is described and

N. Fujieda is with the Department of Electrical and Electronics Engineering, Aichi Institute of Technology, Toyota, Aichi, 470-0392 JAPAN. M. Takeda and S. Ichikawa are with the Department of Electrical and Electronic Information Engineering, Toyohashi University of Technology, Toyohashi, Aichi, 441-8580 JAPAN (e-mail: nfujieda@aitech.ac.jp; m.takeda@ccs.ee.tut.ac.jp; ichikawa@ieee.org).

A part of this work was supported by JSPS Grants-in-Aid for Scientific Research (KAKENHI) Grant Number 16K00072.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2019.2926555, IEEE Transactions on Circuits and Systems II: Express Briefs

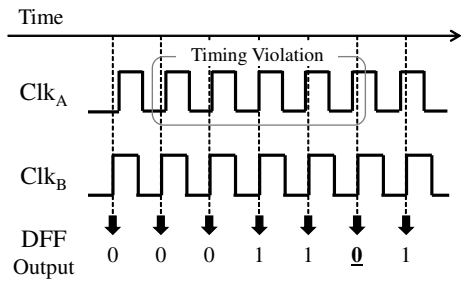

Fig. 2. Effect of timing violation in the DCM-based TRNG.

evaluated in Section V. We conclude the paper in Section VI.

II. Operating Principle of DCM-based TRNG

A. Beat Frequency Detection

Figure 1 depicts the block and timing diagrams of the DCM-based TRNG [2]. It consists of two DCMs, a D flip-flop (DFF), and a counter. The counter counts and outputs the number of consecutive logic-ones. A DCM in the target FPGA family, Virtex-5, have a frequency synthesizer that provides an output clock with a frequency of M/D times the frequency of the input clock. The integer parameters M and D must meet $2 \leq M \leq 33$ and $1 \leq D \leq 32$. Let $\mathrm{Clk_A}$ and $\mathrm{Clk_B}$ be the output clocks, whose respective parameters are (M_A, D_A) and (M_B, D_B). We define the integer ratio of two frequencies as $N + d : N$, where $d > 0$. We also define the period of the clocks as t_A, t_B and the difference of them as $\Delta t = t_B - t_A$.

First, we analyze the circuit operation without considering the effect of jitter and metastability. Suppose that both clocks rise at time zero. The next rising edge comes at t_A and t_B, respectively. $\mathrm{Clk_B}$ thus captures logic-one after Δt of the rising edge of $\mathrm{Clk_A}$. It keeps capturing logic-one for a while and starts capturing logic-zero when the delay reaches $t_A/2$. Eventually, both clocks again rise at the same time of $(N+d)t_A = Nt_B$. At this time, $\mathrm{Clk_A}$ goes d cycles in advance of $\mathrm{Clk_B}$. The bit string captured by $\mathrm{Clk_B}$ becomes d times of the repetition of consecutive logic-ones and logic-zeros. Therefore, the average number of the consecutive logic-ones n becomes $n = (t_A/2)/\Delta t = N/2d$. When there is no effect of jitter and metastability, especially when n is an integer, the value of the counter will always be n. The period of generation of counter values becomes $2nt_B$.

We then model the effect of jitter differently from the previous work [2]. We focus on the delay of $\mathrm{Clk_B}$ from $\mathrm{Clk_A}$, rather than the time, and consider that jitter can shift the time for the sum of delay to reach half of the period of $\mathrm{Clk_A}$. We suppose that the maximum value of the shift is equivalent to the peak-to-peak jitter of the DCM t_j. In this case, the number of successive logic-ones varies within the range of $W = t_j/\Delta t$. For the sets of parameters used in this study, t_j ranges approximately from 400 to 600 ps and Δt ranges from 10 to 40 ps. Therefore, the width of the variation of the number of successive logic-ones, W, is estimated to be 10–60, which especially depends on the value of Δt.

Finally, we consider the case with uncertainty of the output of the DFF due to metastability. Figure 2 shows a timing diagram of the two clocks when $\mathrm{Clk_A}$ is overtaking $\mathrm{Clk_B}$. When $\mathrm{Clk_A}$ rises or falls almost at the same time as a rising edge of $\mathrm{Clk_B}$, it might violate the timing constraints of the DFF. When the violation occurs, the final output value of the DFF becomes indeterminate due to metastability. This may occur when the output of the DFF is going to transit between logic-one and logic-zero. As a result, quite a small counter value will be produced by a temporary appearance of a logic-zero (one) inside the successive logic-ones (zeros).

By cutting out LSBs of the number of consecutive logic-ones, a random number with small bias can be obtained. In addition, a von Neumann Corrector (VNC) [8] is adopted as a postprocessing, which eliminates bias if there is no correlation between successive bits. It reduces the generation rate of the random numbers to at most 1/4. The previous work recommended to cut out three LSBs and pass them through the VNC [2].

B. Reconfiguration of Clock Frequencies

A DCM in Xilinx FPGAs has a dynamic reconfiguration port (DRP), which enables modification of its output frequency without reconfiguration of the entire circuit. This functionality makes the DCM-based TRNG tunable: although the use of a single pair of frequencies might result in generation of random numbers of poor quality, the TRNG can find a desirable pair of frequencies that shows good randomness by trying multiple pairs of frequencies while testing the generated random numbers. In the previous work [2], the following conditions for the parameters of the DCMs were shown explicitly or implicitly:

- $M_A < D_A$ and $M_B < D_B$ (i.e. the output frequency is lower than the input frequency),
- $d = 1$ (the integer ratio of frequencies is $N+1 : N$), and
- $400 \leq N \leq 1000$ (the average counter value n is 200–500).

As sets of parameters that meet the conditions, 23 pairs of frequencies were presented.

However, it is unclear how they selected the pairs of frequencies. In fact, there are 48 sets of parameters that meet the above conditions. For example, D_A and M_B becomes commutative if $M_A < D_A, M_B < D_B$, while only one of such a pair of sets were included. It is also unclear how many pairs of frequencies give good random numbers. The previous work showed that the DCM-based TRNG passed statistical tests for three pairs [2], while the results for the other pairs were not shown in the paper. The proportion of pairs that give good random numbers is inversely proportional to the expected value of the number of pairs of frequencies to be tried to find a desirable one. If it is too small, the practicality of the DCM-based TRNG will be reduced by the time taken for parameter search.

III. Evaluation System

Since this research requires massive sequences of random numbers with multiple postprocessing methods, our evaluation system first collects sequences of counter values on an FPGA evaluation board. Generation of random numbers through cutting out the LSBs and passing them to the VNC is done

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2019.2926555, IEEE Transactions on Circuits and Systems II: Express Briefs

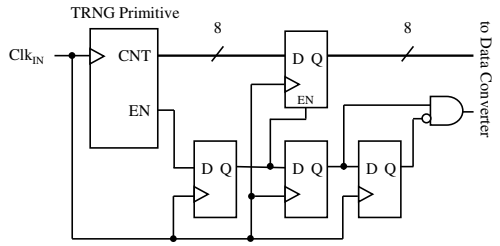

Fig. 3. Block diagram of synchronization and edge detection circuit.

TABLE I
A PART OF FREQUENCY PAIRS SELECTED IN THIS PAPER.

ID	M_A	D_A	M_B	D_B	t_j [ps]	Δt [ps]	n
1	31	32	30	31	609	10.8	480.0
2	30	31	29	30	592	11.5	449.5
3	29	30	28	29	576	12.3	420.0
68	31	32	28	29	593	34.6	149.3
98	17	23	14	19	437	42.0	161.0
99	9	10	26	29	412	42.7	130.0
100	18	20	26	29	412	42.7	130.0

by software later. The counter of the TRNG primitive shown in Fig. 1 is designed so that only its MSB can be saturated (i.e. the next value to `0xff` can be `0x80`). The number of successive logic-ones may exceed 2^8, or even 2^9, yet its 2^7 or higher bits can be easily recovered by calculation in this design. While the DFF and the counter in the TRNG primitive are synchronized with $\mathrm{Clk_B}$, the other circuits are synchronized with the 100-MHz input clock $\mathrm{Clk_{IN}}$. To connect the output of the TRNG primitive to the subsequent circuits, synchronizer and edge detection circuits are required to prevent malfunction of the subsequent circuits. The block diagram of the additional circuits is shown in Fig. 3. In the evaluation shown in Section V, a data converter circuit that compresses an 8-bit counter value to 4 bits is also added, which will be described in Section V.A. Obtained counter values are sent to a PC via a UART connection of 460800 bps using a controller with a 4-KB buffer. When the output of the counter is written to the end of the buffer, subsequent counter values will be discarded until all data in the buffer is sent out. The DCMs are not dynamically reconfigured. An individual bit stream to the whole FPGA is generated for each set of parameters of the DCMs.

On the selection of pairs of frequencies, we adopted only the first kind of conditions shown in Section II.B: $M_A < D_A$ and $M_B < D_B$. We then selected 100 pairs of frequencies in ascending order of Δt. Table I enumerates some of the selected pairs of frequencies. The clock jitter t_j is calculated from the average of the peak-to-peak jitter of the DCMs, which is available in the IP core generator [9]. As a consequence, we had to exclude one pair (out of 23) listed in the previous work [2], whose parameters were $(M_A, D_A), (M_B, D_B) = (15, 31), (14, 29)$, because its Δt (47.6 ps) was too large. There were two pairs that met $d \neq 1$, including the 68th pair. Their integer ratio of frequencies was $899 : 896$.

The evaluation system is configured for a Xilinx Virtex-5 FPGA in the same way as the previous work [2]. ISE 14.7 and an ML501 board are used as a CAD tool and an evaluation board, respectively.

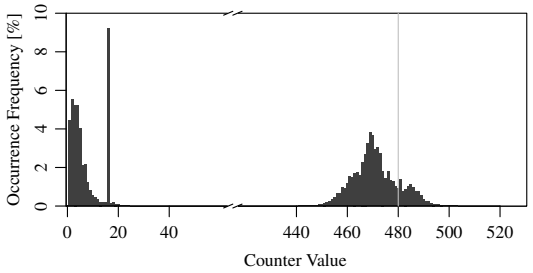

Fig. 4. Distribution of counter value (1st pair).

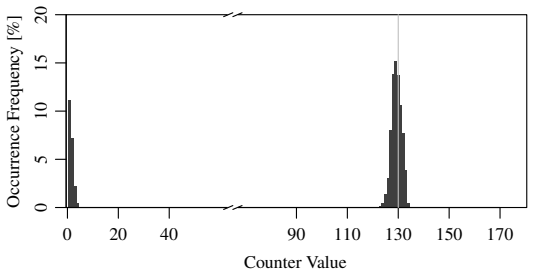

Fig. 5. Distribution of counter value (100th pair).

IV. DISTRIBUTION OF COUNTER VALUES

This section describes a preliminary evaluation, where the 8-bit counter values are obtained from the evaluation system and distribution of them is plotted. We obtained 10 million counter values for each pair of frequencies.

Figures 4 and 5 plot the distribution of counter values obtained from the 1st and 100th frequency pairs, respectively. The X-axis represents the counter value and the y-axis represents the occurrence frequency for each value. A gray vertical line stands for n. For all the pairs of frequencies, the obtained distributions exhibited biphasic histograms, whose peaks appeared around zero and n. The variation of counter values around n was due to jitter, whose width agreed with our model described in Section II.A. The counter values that appeared around zero came from metastability. They were no greater than 47. In Fig. 4, the peak around n was slightly shifted to the left because counter values around zero were frequently observed.

As it can also be understood by comparing Fig. 4 with Fig. 5, the probability of having counter values around zero and the standard deviation of counter values around n had strong negative correlation with Δt. Their correlation coefficients were -0.643 and -0.871, respectively. The standard deviation of counter values around zero was much smaller than around n. When Δt is quite small, as shown in the counter value of 16 in Fig. 4, another peak was observed at a particular value.

V. EVALUATION WITH VARIOUS WIDTH OF COUNTER LSBS

A. Biphasic Postprocessing

In this study, we propose to make the bit width of LSBs of the small counter values to be extracted smaller than that of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2019.2926555, IEEE Transactions on Circuits and Systems II: Express Briefs

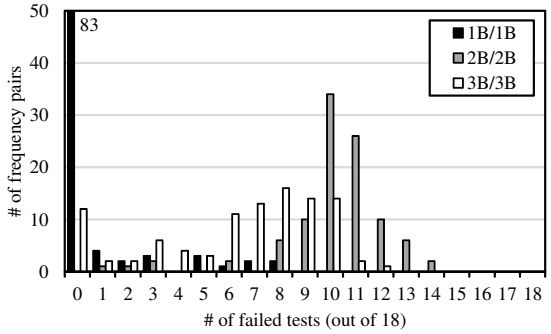

Fig. 6. Distribution of the number of failed tests (same width of LSBs).

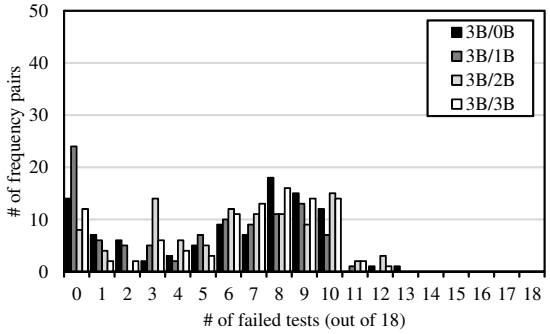

Fig. 7. Distribution of the number of failed tests (different widths of LSBs).

large counter values. From the preliminary experiment shown in Section IV, counter values can be clearly divided into large ones and small ones. A comparison of their standard deviation implies that small counter values have less uncertainty than large ones. Therefore, it is expected that our proposal can improve the quality of generated random numbers with a minimal reduction of the generation rate of the TRNG.

In this evaluation, an 8-bit counter value is compressed to 4 bits before being sent out from the evaluation system. The MSB of the compressed value is set to one when the counter value is 48 or greater. This means the MSB is a flag to determine if the counter value is classified as large. The threshold is based on the results shown in Section IV. For the other three bits, the three LSBs of the counter value are used without modification. An 8-bit value is generated by concatenating two compressed values, which is sent to the buffer. We obtained 500 million (i.e. 250 MB of) compressed counter values for each pair of frequencies.

A 100-Mbit random bit string, generated from each series of counter values by bit extraction and postprocessing, is tested by the diehard test [4]. Although the diehard test is simple, it is useful for filtering apparently poor random numbers. It consists of 18 types of tests that give 1–100 p-values. In this paper, we define a failed p-value as not meeting the condition of $10^{-6} \leq p \leq 1-10^{-6}$. If a test gives less than five p-values, it is considered as failed when it gives at least one failed p-value. Otherwise, the result of the Kolmogorov–Smirnov test of given p-values is used as the criterion, which corresponds to the flatness of the p-values.

In the evaluation, the bit widths of LSBs of large and small counter values to be cut out are defined as w_l and w_s, respectively. A TRNG with these parameters is expressed as w_lB/w_sB. Nine combinations of parameters that meet $1 \leq w_l \leq 3$ and $0 \leq w_s \leq w_l$ are evaluated. The recommendation in the previous work [2] corresponds to 3B/3B.

The generation rate of the TRNG is estimated from the fact that the period of generating a large counter value is $2nt_B$ (as shown in Section II.A). When we suppose that the proportion of small counter values is α and the generation rate is reduced to $1/4$ by the VNC, the generation rate can be calculated by $(1/8nt_B) \times \{w_l + w_s \times \alpha/(1-\alpha)\}$.

TABLE II
CORRELATION COEFFICIENTS BETWEEN THE NUMBER OF FAILED TESTS AND PARAMETERS OF DCMS.

	3B/3B	3B/1B
Multiplier of $\mathrm{Clk_A}$ (M_A)	-0.317	-0.292
Divisor of $\mathrm{Clk_A}$ (D_A)	-0.220	-0.245
Multiplier of $\mathrm{Clk_B}$ (M_B)	**-0.434**	-0.365
Divisor of $\mathrm{Clk_B}$ (D_B)	-0.349	-0.336
Difference of clock periods (Δt)	0.183	**0.471**
Clock jitter (t_j)	-0.207	**-0.447**
Expected counter value (n)	-0.127	**-0.405**
Width of variation of counter (W)	-0.224	**-0.449**

B. Results

Figure 6 plots the results of the diehard test in the cases where the same width of LSBs are cut out from all the counter values (i.e. $w_l = w_s$). The x-axis represents the number of types of tests judged to be failed (out of 18). The y-axis stands for the number of such pairs of frequencies. Only 12 pairs passed all the tests in 3B/3B, which corresponded to the previous work [2], while 83 pairs passed in 1B/1B. All frequency pairs in 2B/2B failed at least one test. In fact, 2B/2B showed a similar result to 1B/1B without the VNC. The VNC did not help improve this case because it merely cut out the cases where the two LSBs of the counter were "01" or "10."

Figure 7 shows the summary of the results where w_l is fixed to 3 bits. In 3B/1B, or the case where only one LSB was cut out from each of small counter values, the number of frequency pairs that passed all the tests was doubled to 24. This means the time to find a proper frequency pair becomes almost halved. Although uncertainty of small counter values was low, one LSB of them was still available for generating random numbers. The quality of random numbers was also slightly improved in 3B/0B, where small counter values were simply discarded; however, it was not as good as 3B/1B. Successive large counter values might be somewhat correlated.

Table II enumerates the correlation coefficients between the number of failures in the diehard test and DCM parameters. Randomness basically comes from the variation of counter values, quantified by $W = t_j/\Delta t$. It was actually well correlated with the result of 3B/1B. Since W strongly depends on Δt, a simple strategy to try a parameter with a small Δt at first is effective to find a good frequency pair for 3B/1B.

However, the result of 3B/3B was not much correlated with W. The correlation was relatively weaken by less uncertainty

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2019.2926555, IEEE Transactions on Circuits and Systems II: Express Briefs

TABLE III
THE NUMBER OF FREQUENCY PAIRS PASSED ALL THE TESTS AND THE GENERATION RATE OF TRNG.

w_l	w_s	Pass	Rate (All) [kb/s]	Rate (Passed) [kb/s]
1	0	77	55.8	54.1
1	1	83	70.0	72.3
2	0	2	111.5	123.1
2	1	**46**	**125.7**	**118.9**
2	2	0	139.9	n/a
3	0	14	167.3	117.3
3	1	**24**	**181.5**	**173.6**
3	2	8	195.7	126.9
3	3	12	209.9	209.0

TABLE IV
RESULTS OF THE NIST SP 800-22 TEST SUITE (1ST PAIR).

	3B/3B		3B/1B	
name	p-value	prop.	p-value	prop.
Frequency	0.37967	98.9%	0.01068	99.6%
BlockFrequency	**0.00000**	98.4%	**0.00000**	98.4%
CumulativeSumsUp	0.48334	99.2%	0.28684	99.2%
CumulativeSumsDown	0.65400	99.2%	0.53818	100.0%
Runs	**0.00000**	**5.6%**	**0.00000**	**7.5%**
LongestRun	0.00378	97.9%	0.11539	98.8%
Rank	0.67645	98.9%	0.40828	98.8%
FFT	0.05789	98.7%	0.03292	99.2%
NonOverlappingTemplate	**0.00000**	**80.4%**	**0.00000**	**93.2%**
OverlappingTemplate	**0.00000**	**85.6%**	0.00001	**95.3%**
Universal	**0.00000**	**88.0%**	0.00052	**96.4%**
ApproximateEntropy	**0.00000**	**13.1%**	**0.00000**	**31.2%**
RandomExcursions	0.08362	98.7%	0.26551	98.4%
RandomExcursionsVariant	0.80802	99.6%	0.11763	99.3%
Serial1	**0.00000**	**50.7%**	**0.00000**	**94.1%**
Serial2	0.23714	98.4%	0.24424	98.8%
LinearComplexity	0.53143	98.9%	0.75187	99.6%

in small counter values, which degraded the quality of random numbers. Instead, a certain level of correlation was observed with M_B. Although we do not have an internal mechanism that account for this relation so far, we can suggest that a parameter with a large M_B should be tried first.

Table III summarizes the number of frequency pairs that passed all the tests (Pass) and the average generation rate of the TRNG (Rate) for all the combinations of w_l and w_s. The column Rate (All) is the average of the generation rate of all 100 pairs, while the column Rate (Passed) is that of only the pairs that passed all the tests. When w_s is limited to one bit, the number of passing frequency pairs was greatly improved. A finer trade-off between the generation rate and the quality of random numbers (which is directly connected with the time to find a proper frequency pair) could be made by the use of 3B/1B and 2B/1B. Especially in 3B/1B, the number of frequency pairs that passed all the tests was doubled from 3B/3B, at a cost of only 16.9% of the reduction of the generation rate.

Finally, the frequency pairs that passed diehard tests in 3B/3B or 3B/1B are tested by the NIST SP 800–22 test suite [6], which is stricter than the diehard test. Although NIST recommends using 1000 bit strings of 1 Mbit, we used 253–411 bit strings, as many as we can generate from 500 million counter values. As a result, however, no frequency pairs passed all the kinds of tests, though the proportion of passing bit strings got better in 3B/1B than 3B/3B. The results of the tests in the 1st pair (in Tab. I) are summarized in Tab. IV for example. Possible reasons for this result include the individual difference of an FPGA or its elements, difference of measurement condition, and so on. A stronger postprocessing unit might be required. On the other hand, it is also confirmed through this experiment that the proposed 3B/1B TRNG improved the quality of random numbers on some level.

VI. CONCLUSION

In this paper, we presented a method that increased the practicality of the DCM-based TRNG, proposed by Johnson et al. [2], by making the tuning of frequency pairs easier. We comprehensively evaluated the TRNG and found that the value of the counter exhibits biphasic histograms. From this observation, we proposed to apply different bit widths between large and small counter values to cutting out their LSBs. Our evaluation showed that the quality of random numbers became better by extracting only one LSB from small counter values.

Our future work includes a further analysis and experiment for improving the quality of random numbers. In the NIST test suite, a low proportion of passing bit strings was observed especially in the Runs test, which implied that successive zeros and ones were generated slightly more frequently than expected. An analysis of the DCM-based TRNG on newer FPGAs is also important. A 7-series or newer Xilinx FPGA has more sophisticated DCMs called MMCMs (Mixed Mode Clock Managers). A finer adjustment can be expected by their frequency synthesizer, which supports 1/8 interval of the multiplier and the divisor.

REFERENCES

[1] N. Fujieda and S. Ichikawa, "A latch-latch composition of metastability-based true random number generator for Xilinx FPGAs," *IEICE Electronics Express*, vol. 15, no. 10, pp. 20 180 386:1–20 180 386:12, 2018.

[2] A. P. Johnson, R. S. Chakraborty, and D. Mukhopadyay, "An Improved DCM-Based Tunable True Random Number Generator for Xilinx FPGA," *IEEE Transaction on Circuits and Systems II: Express Briefs*, vol. 64, no. 4, pp. 452–456, 2017.

[3] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control," in *Proc. International Workshop on Cryptographic Hardware and Embedded Systems 2011*, pp. 17–32.

[4] G. Marsaglia. Diehard battery of tests of randomness (mirror). [Online]. Available: https://github.com/reubenhwk/diehard

[5] B. Ray and A. Milenković, "True Random Number Generation Using Read Noise of Flash Memory Cells," *IEEE Transactions on Electron Devices*, vol. 65, no. 3, pp. 963–969, 2018.

[6] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, *A statistical test suite for random and pseudorandom number generators for cryptographic applications*, NIST Special Publication 800–22, Rev. 1a, 2010.

[7] 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 *Proc. IEEE 2014 Custom Integrated Circuits Conference*, 2014, pp. 1–4.

[8] J. von Neumann, "Various techniques used in connection with random digits," *Monte Carlo Method, National Bureau of Standards Applied Mathematics Series 12*, pp. 36–38, 1951.

[9] Xilinx Inc., *Virtex-5 FPGA Data Sheet: DC and Switching Characterisics*, Product Specification DS202 (v5.5), 2016.