Computing and Informatics, Vol. 23, 2004, 501–515

A SIMPLE PLL-BASED TRUE RANDOM NUMBER GENERATOR FOR EMBEDDED DIGITAL SYSTEMS

Miloš Drutarovský, Martin Šimka

Department of Electronics and Multimedia Communications
Technical University of Košice
Park Komenského 13, 041 20 Košice, Slovak Republic
e-mail: {Milos.Drutarovsky, Martin.Simka}@tuke.sk

Viktor Fischer, Frédéric Celle

Laboratoire Traitement du Signal et Instrumentation
Unité Mixte de Recherche CNRS 5516, Université Jean Monnet
10, rue Barrouin, 42000 Saint-Etienne, France
e-mail: {fischer, celle}@univ-st-etienne.fr

Manuscript received 22 December 2004

Abstract. The paper presents a simple True Random Number Generator (TRNG) which can be embedded in digital Application Specific Integrated Circuits (ASICs) and Field Programmable Logic Devices (FPLDs). As a source of randomness, it uses on-chip noise generated in the internal analog Phase-Locked Loop (PLL) circuitry. In contrast to traditionally used free-running oscillators, it uses a novel method of randomness extraction based on two rationally related synthesized clock signals. The generator has been developed for embedded cryptographic applications, where it significantly increases the system security, but it can be used in a wide range of other applications. The functionality of the proposed solution is demonstrated for the Altera Apex FPLD family, but the same principle can be used for all recent ASICs or FPLDs that include an on-chip reconfigurable analog PLL. The quality of the TRNG output is confirmed by applying special DIEHARD and NIST statistical tests, which pass even for high output bit-rates of several hundreds of Kbits/s.

Keywords: Cryptography, FPGA, PLL, clock jitter, TRNG, DIEHARD, NIST, statistical tests

1 INTRODUCTION

Producing unpredictable, i.e. irreproducible uniformly distributed random number sequences on hardware is one of the central issues of the design of cryptographic systems. Random numbers are needed, in particular, for the key generation, authentication protocols, zero-knowledge protocols, padding, in many digital signature schemes, and even in some encryption algorithms [1, 2]. In all these applications, security greatly depends on the quality of the source of random numbers.

True Random Number Generators (TRNGs) can be produced using some non-deterministic process. The most typical implementations, which cannot be embedded in digital devices, use quantum mechanics [3] producing a random sequence with a rate up to 1 Mbit/s. Radioactivity can also provide true random bits [4], but at a much lower rate of several hundreds of bits per second. Many types of hardware generators, which can be embedded in digital devices, have already been published. They usually exploit the thermal noise (resistance or shoot) in electronic devices and *free*-running oscillator(s) [5]. In the well-known Intel Random Number Generator (RNG) [6], a slow clock signal modulated by an analog thermal noise is amplified and sampled using fast *asynchronous* clock signal from a free-running oscillator. The drift between the two clocks provides a source of random bits with an average rate of 75 Kbits/s.

Our aim was to find a solution, which could be embedded in a digital circuit. Digital circuits are well suited for implementation of so called Pseudo-Random Number Generators (PRNGs). However, PRNGs themselves cannot provide sufficient security for applications in cryptography. Even the best cryptographically secure PRNGs (like BBS etc. [1]) require a truly random initialization sequence that is typically provided by a TRNG embedded in the target hardware. Typical digital circuits include only a limited number of sources of randomness, e.g. metastability, frequency of a free-running oscillator, clock jitter, etc. Usually, reliable generators based on the metastability and/or frequency instability are difficult to implement or they are not secure enough for cryptographic applications. In some cases, the entropy increase is not sufficiently high, so the output of the generator can be predicted [7]. Free-running oscillators are typically used in known TRNGs realized in Field Programmable Logic Devices (FPLDs) [8, 9]. The design [8] uses off-chip components that generally decrease the cryptographic security of the implementation. Implementation [9] requires very careful placement of ring oscillator pairs embedded in the Xilinx FPLD. It can provide random bits at speeds up to 0.5 Mbit/s.

In contrast to these methods, we have proposed a novel method of randomness extraction (see [10]) based on two *rationally* related periodic signals. It was shown that it is perfectly suited for modern FPLDs with the internal analog Phase-Locked Loop (PLL) circuitry (e.g. Apex [11] or Stratix [12] FPLDs from Altera). The generator provides random data with only small deviations from the ideal one.

In this paper, we present a modified version of our generator [10] having simplified structure and higher data rate. It is based on an extended knowledge of the jitter obtained by measurements on application cards in real conditions.

2 JITTER OF THE PLL-SYNTHESIZED CLOCK SIGNALS

2.1 Analog PLLs Embedded in Digital Circuits

New digital VLSI circuits use advanced clock generation and distribution circuitry based on embedded analog PLLs [11–14]. A simplified block diagram of one analog PLL block typically available in advanced digital circuits is shown in Figure 1. Each PLL block can provide at least one synthesized clock signal with the frequency F_{OUT}:

$$F_{OUT} = F_{IN}\frac{m}{n \times k} = F_{IN}\frac{K_M}{K_D}, \tag{1}$$

where F_{IN} is the frequency of the external input clock source. Reference-, feedback- and post-divider values n, m, and k can vary from one to several hundreds in FPLDs [11, 12], or to several thousands in ASICs [14].

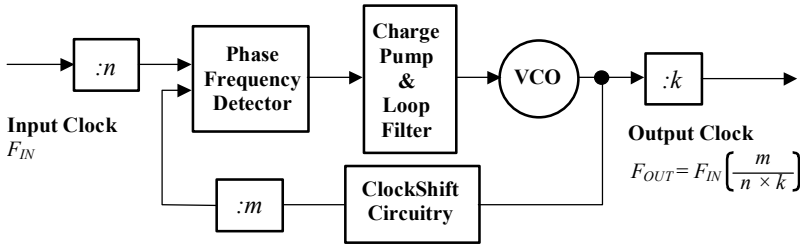


Fig. 1. Simplified block diagram of a typical analog embedded PLL circuitry

2.2 Jitter of the PLL Synthesized Clock Signals

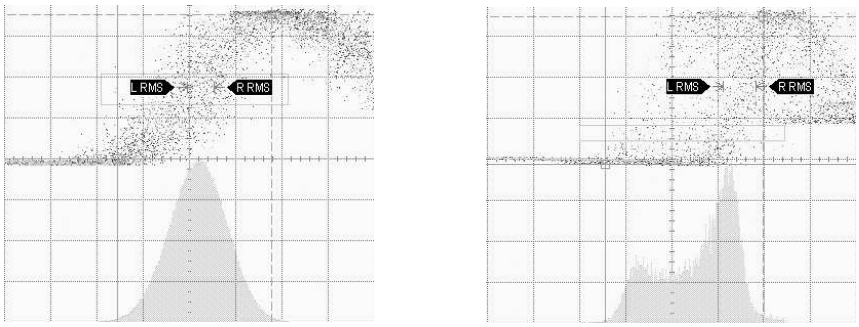
In analog PLLs, a noise causes the Voltage Controlled Oscillator (VCO) to fluctuate in frequency. Other frequency fluctuations are caused by variations of supply voltage, temperature, and by a noisy environment. The internal PLL control circuitry adjusts the VCO back to the specified frequency, but a certain part of the fluctuations caused by the non-deterministic noise cannot be compensated for and is seen as a clock *jitter*.

The size of the intrinsic jitter depends on the quality factor Q of the VCO, on the bandwidth of the Loop Filter, and on the so-called pattern jitter introduced by the Phase Frequency Detector. It is often given in peak-to-peak value or 1-sigma (or RMS) value. 1-sigma value of the jitter (σ_{jit}) depends on the technology and the configuration of the PLL and it can range from 3.5 ps to 10 ps for ASICs [14], or up to 50 ps for FPLDs [11, 12]. Since the technology of the PLL and the quality of the VCO is usually defined, the user can change the output jitter by a modification of the divider values and the filter bandwidth.

For example, the analog PLL jitter in an Apex FPLD has 1-sigma value of $\sigma_{jit} \approx 15.9$ ps for a $F_{OUT} = 66.6$ MHz synthesized clock signal and multiplication factor 2 [15]. These results were acquired under "ideal conditions", with only a minimal amount of FPLD resources occupied by the application and minimal input/output activities. Our last measurements (cf. in Subsection 2.3) show that the clock jitter in the Apex FPLD is significantly higher (about 140 ps) when higher frequency multiplication factors are used and when internal flip-flops are switching on different clock frequencies. Altera application engineers have confirmed these facts, too.

2.3 Jitter Size Measurement

Since the knowledge of the jitter and its statistical features is crucial for a correct settings of the TRNG parameters, several measurements have been made for different configurations of the PLL (different values K_M and K_D). We used Agilent Infiniium DCA 86100B wide-bandwidth oscilloscope and the Altera Nios development board [16] with Apex EP20K200EFC484-2X device for the reference measurements. The 1-sigma value of the jitter measured at the output of the PLL has achieved values $\sigma_{jit} \approx 32$ ps for $F_{IN} = 33.3$ MHz, and $K_M/K_D = 2/1$. For ratios $K_M/K_D = 3/2, 4/3, 5/4, 6/5, \ldots$, the 1-sigma value was in the range $\sigma_{jit} \approx 47$–64 ps and the jitter has approximately Gaussian distribution as it is illustrated in Figure 2(a) for the configuration $K_M/K_D = 6/5$.



(a) $K_M/K_D = 6/5$ (b) $K_M/K_D = 157/48$

Fig. 2. Probability distribution of the jitter for two different PLL configurations: (a) the Gaussian distribution for $K_M/K_D = 6/5$ with $\sigma_{jit} \approx 47$ ps, (b) two-peak jitter distribution for $K_M/K_D = 157/48$ with $\sigma_{jit} \approx 140$ ps and 600 ps peak-to-peak value

However, for $K_M/K_D = 157/48$ used in [10], the jitter was $\sigma_{jit} \approx 140$ ps and it exhibited two peaks with a total size of 600 ps (peak-to-peak). The obtained jitter distribution is visualized in Figure 2(b) and it is compatible with a reference measurement made by Xilinx on Altera FPLDs [15], where a two-peak jitter

distribution has been documented. Since the jitter included in the clock signal in real conditions was significantly higher than that documented by Altera (note that Altera has used the same kind of development board [16], but with different parameters K_M and K_D), we could significantly reduce the TRNG complexity. As far as K_M and K_D parameters are chosen properly, the proposed method of randomness extraction is insensitive to the jitter distribution (see Section 3 and [10]).

3 ROBUST RANDOMNESS EXTRACTION FROM THE CLOCK JITTER GENERATED BY THE PLL CIRCUITRY

The basic principle behind our method is an extraction of the randomness from the jitter of the clock signal synthesized in the embedded analog PLL. The jitter is detected by the sampling of a reference (clock) signal using a rationally related (clock) signal synthesized in the on-chip analog PLL. The fundamental problem lies in the fact that the reference signal has to be sampled near the edges influenced by the jitter. The structure of a simplified true random number generator is depicted in Figure 3.

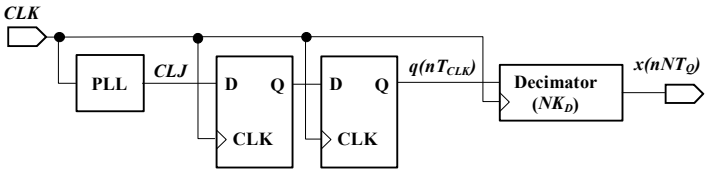


Fig. 3. Structure of a simplified true random number generator using PLL-synthesized low-jitter clock signal

Because there is a probability that the first flip-flop could become metastable, the second flip-flop is cascaded. In the case when the first flip-flop produces a metastable output, it can resolve until its output is clocked by the second flip-flop. This flip-flops connection does not assure that only the stable signal is clocked, but the probability that the output $q(nT_{CLK})$ will get a valid logical value is much higher [17].

Let CLJ be an on-chip PLL-synthesized rectangular clock waveform with the frequency

$$F_{CLJ} = F_{CLK}\frac{K_M}{K_D}, \tag{2}$$

where CLK is a reference clock signal and parameters K_M and K_D defined in (1) are related to the PLL structure. A signal CLJ is sampled into the D flip-flops using a clock signal with frequency F_{CLK}. There are K_D rising edges of CLK signal and $2K_M$ (rising and falling) edges of a CLJ waveform during the time period

$$T_Q = K_D T_{CLK} = K_M T_{CLJ}. \tag{3}$$

It has been shown in [10] that if K_M and K_D are relatively prime, the set of samples creates an equidistant set of values with a distance step

$$d = \frac{T_{CLK}}{2K_M}\mathrm{GCD}(2K_M, K_D) = \frac{T_{CLJ}}{2K_D}\mathrm{GCD}(2K_M, K_D), \tag{4}$$

where GCD means Greatest Common Divisor. It has been shown that the worst-case distance between the two closest edges of CLK and CLJ during the period T_Q is given as

$$\mathrm{MAX}(\Delta T_{min}) = d/2. \tag{5}$$

If K_M, K_D and F_{CLJ} are chosen so that

$$\sigma_{jit} > \mathrm{MAX}(\Delta T_{min}), \tag{6}$$

we can guarantee that during T_Q the sampling edge of CLK will fall at least once into the edge zone of CLJ (the edge zone means the time interval around the edge with a width smaller than σ_{jit}). Therefore, during the period T_Q, K_D values of CLJ will be sampled into the first D flip-flop and at least one sampled value will statistically depend on the random jitter, so the output value $q(nT_{CLK})$ of the second flip-flop will be nondeterministic. In [10] we used delay elements illustrated in Figure 4 to increase the probability of overlapping of CLK and CLJ edge zones.

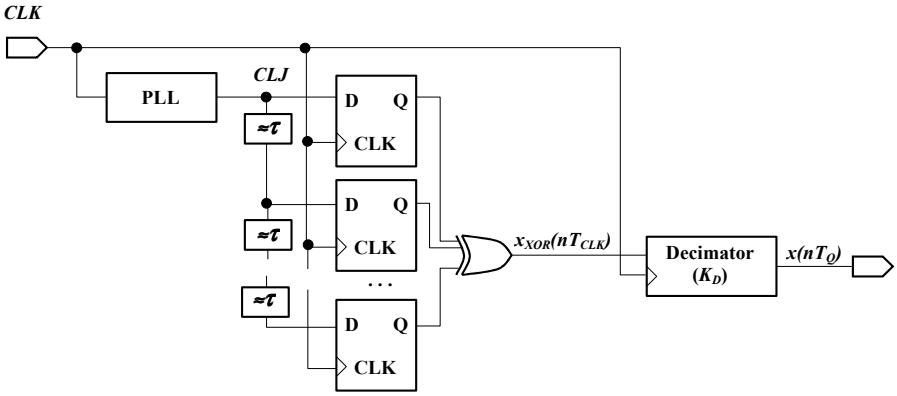


Fig. 4. Structure of the originally proposed PLL-based true random number generator with delay line elements and XOR corrector [10]

Thanks to the measurements, we obtained the information that the jitter value, and also the probability of edges overlapping are higher as it was expected before. Therefore, the delay line is not needed anymore for the condition

$$\sigma_{jit} \gg \mathrm{MAX}(\Delta T_{min}). \tag{7}$$

The decimated output signal of the simplified TRNG

$$x(nT_Q) = q(nT_Q) \oplus q(nT_Q - T_{CLK}) \oplus \ldots \oplus q(nT_Q - (NK_D - 1)T_{CLK}), \quad (8)$$

which is generated at the output of an Exclusive-OR (XOR)-based decimator [18] as a bit-wise addition modulo 2 ($\oplus$) of NK_D samples $q(.)$ sampled with the frequency F_{CLK} will be nondeterministic, too. Note that the delay line can still be a useful building block for $\sigma_{jit} \approx \max(\Delta T_{\min})$ or $\sigma_{jit} < \max(\Delta T_{\min})$, as it was shown in [19].

It can be seen that comparing to [10] we have changed the basic structure of the TRNG in several ways:

- because of the higher jitter value, which has been approved by precise jitter measurements, we could replace the delay line and the bank of flip-flops by a single flip-flop,
- the metastability behavior of the signal $q(nT_{CLK})$, and thus of the generator in general was improved by the addition of the second flip-flop,
- we have validated that if condition (7) is fulfilled, the speed of the generator can be increased by reducing the decimation factor to NK_D, where $N = 1$ is a number of T_Q periods.

4 TRNG IMPLEMENTATION

We have validated our simplified structure of the TRNG using Altera analog PLLs embedded in Apex E FPLD family. We have used a custom evaluation board with a PC Card interface and Apex EP20K160 E device. The Apex EP20K160ETC144-2X device has an included TRNG, 16 × 128-bit FIFO, PC Card interface, and a custom logic. As the best option, a 2-PLL configuration (shown in Figure 5) with only one common input clock signal has been chosen. Note that such clock configuration was not possible for original Altera NIOS board [16].

Synthesized clock signals CLK and CLJ are not fed out from the FPLD (in the design presented in [10], one synthesized signal has been fed out of the device and reused in FPLD, what is the definitely less secure solution). Therefore, they cannot be manipulated separately without a circuit reconfiguration. This fact is very important for cryptographic applications, because it significantly improves overall system security.

Multiplication and division factors for individual output signals were selected as follows (the first number represents the PLL index and the second number the PLL output index):

$$\begin{aligned} clk20 &= 40 \times 53/22 = 96.36\,\text{MHz}, \\ clk40 &= 40 \times 19/38 = 20\,\text{MHz}, \\ clk41 &= 40 \times 19/8 = 95\,\text{MHz}. \end{aligned} \quad (9)$$

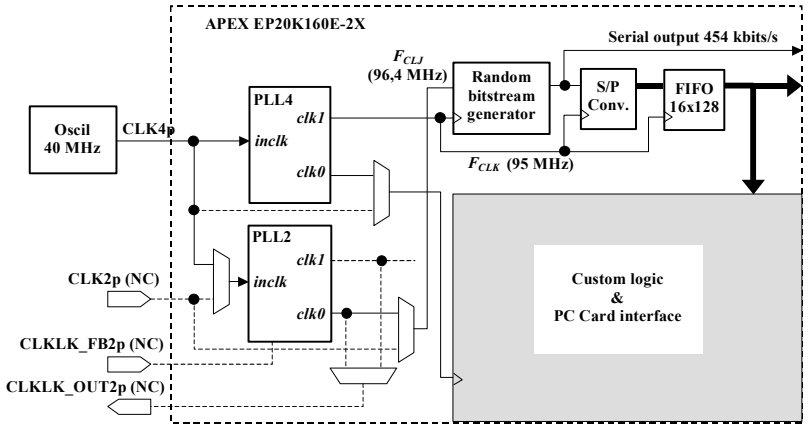


Fig. 5. Block diagram of the experimental PC card with Apex EP20K160ETC144-2X

Therefore, according to (4) and (5) $\mathrm{MAX}(\Delta T_{min}) \approx 12\,\mathrm{ps}$. This value is much smaller than the measured jitter value σ_{jit} (see Section 2.3) and the condition (7) is fulfilled. The TRNG was implemented for $N = 1$, and so $R_{TRNG} \approx 454.000\,\mathrm{bit/s}$ (about six times more than in [10]). This was done intentionally in order to check a quality of the generated TRNG output bitstream under more stringent conditions.

The FPLD resource requirements of the proposed TRNG block as well as supporting logic (FIFO and control logic) are shown in Table 1. The first four columns show resource requirements (Logic Cells (LC), Embedded System Blocks (ESB), PLL outputs and Global Clocks (GCLK)) for the generator. The other four columns give requirements of the complete circuit including FIFO buffer and data bus controller. Presented results have been obtained using Altera Quartus II v. 2.2 SP2.

TRNG only				TRNG+FIFO and logic			
LC	ESB	PLL	GCLK	LC	ESB	PLL	GCLK
19	0	2	2	129	2	2	3

Table 1. Resource requirements of TRNG and supporting logic

While 19 logic cells represent less than 1 % of available FPLD cells, the project uses 100 % of available PLLs (two PLLs). Since each PLL contains two clock outputs, two additional clock outputs remain available (one at each PLL) and, with some restrictions, they can be used inside the application area (e.g. in our case for the PC Card interface). An internal usage of the PLL clock signals does not represent a security problem as these signals cannot be externally manipulated.

5 STATISTICAL EVALUATION OF THE SIMPLIFIED TRNG

There are some well-documented general statistical tests that can be used to look for deviations from an ideal TRNG [1, 20–23]. A good TRNG should pass all kinds of these tests. The following subsections present testing results of the proposed TRNG using the standard DIEHARD and NIST statistical test suits. The DIEHARD and NIST tests are the most frequently used statistical packages for evaluation of PRNGs and TRNGs. These tests are applied to the 80-Megabit sequence and a set of 1-Megabit ones for DIEHARD and NIST, respectively. Such record lengths are typically used for evaluation of TRNGs [6, 20, 21]. To prove even further the quality of the generator, we have also applied very strict Frequency (Monobit) tests for significantly longer records.

5.1 Results of the DIEHARD Statistical Test

The DIEHARD test is a series of statistical tests developed by George Marsaglia for testing mainly PRNGs [20]. In spite of that, it is frequently used also for testing of TRNGs.

Our DIEHARD statistical tests were performed on standard continuous 80-Megabit TRNG output records (for $N = 1$). The DIEHARD is a collection of 15 tests, most of which give several results. There are 234 p-values generated by the test. For ideal random numbers, the p-values are uniform over the range $\langle 0, 1)$. Figure 6 shows the distribution of p-values for an ideal RNG (dashed straight line of the uniform distribution) and for the tested TRNG (solid line). It is clearly visible that the distribution of p-values for the tested TRNG can be closely approximated by the uniform distribution. Moreover, the absence of p-values equal to 1.00000 shows that no individual DIEHARD test has failed [20].

5.2 Results of the NIST Statistical Test

It seems that the NIST statistical test suite [21] is currently the most comprehensive publicly available test tool. It is developed for testing of PRNGs as well as TRNGs. In comparison to the DIEHARD test, it provides comprehensive support for the interpretation of test results.

Our NIST statistical tests were performed on continuous 1-Gigabit TRNG output records (for $N = 1$) and followed the testing strategy, general recommendations, and result interpretation described in [21]. We have used a set of 1000 1-Megabit[1] sequences produced by the TRNG and we have evaluated the set of P-values[2] at a significance level $\alpha = 0.01$. We have got similar results as in [10] (with the exception of FFT test, see below) so we can conclude that there are no detectable

[1] Modified Lempel-Ziv test [24] was used with 1 000 000 bit sequences.

[2] Note that P-values generated by NIST test are not compatible with p-values generated by DIEHARD test.

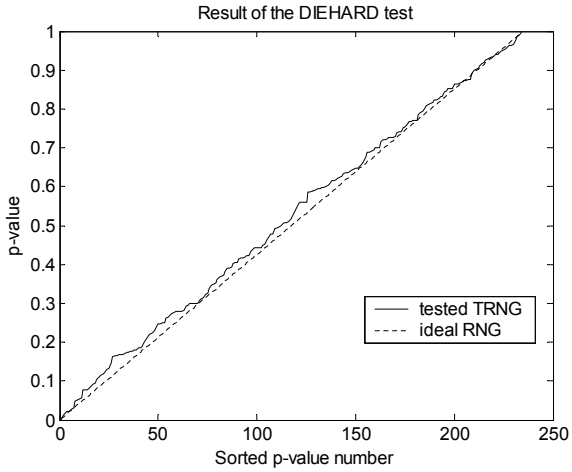


Fig. 6. DIEHARD test results of the tested TRNG (solid line) vs. ideal RNG

differences for ensemble of 1000 1-Megabit records. Results of these tests are included in Table 2.

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-value	Proportion	Statistical Test
104	89	104	98	100	107	109	101	90	98	0.916599	0.9860	Frequency
92	111	123	85	101	107	84	101	109	87	0.092041	0.9940	Block-Freq.
105	98	97	107	98	95	92	105	110	93	0.939005	0.9870	Cusum
107	117	75	99	105	99	101	89	95	113	0.159910	0.9890	Runs
100	86	103	99	92	123	87	92	109	109	0.216713	0.9920	Long-run
84	108	99	101	102	86	104	101	114	101	0.599693	0.9900	Rank
113	92	97	114	82	101	105	95	101	100	0.520102	0.9910	FFT
92	118	124	83	102	103	111	85	86	96	0.032489	0.9870	Periodic-Templ.
119	89	85	93	108	108	103	107	101	87	0.267573	0.9790	Universal
96	110	88	113	91	96	110	97	105	94	0.641284	0.9910	Apen
98	101	100	114	92	92	103	98	99	103	0.940080	0.9860	Serial
92	90	116	86	136	76	78	118	103	105	0.307671	0.9880	Lempel-Ziv
112	110	89	97	89	99	93	97	97	117	0.482707	0.9890	Linear-Compl.

Table 2. NIST test results of the simplified TRNG (uniformity of P-values and proportion of passing sequences) for the 1-Gigabit record that passed all tests

It was claimed in [10] that some 1-Gigabit TRNG records did not pass NIST FFT tests. During the evaluation of the proposed new TRNG implementation we have found errors in the NIST FFT test formulation (confirmed also in [24]). After correction of these errors in the NIST package, all FFT tests passed without any problems for all tested (1-Gigabit) records.

5.3 Frequency (Monobit) Test of Very Long Records

The most common statistical test of the TRNGs is the Frequency (Monobit) test [21]. Good TRNGs should provide independent binary (Bernoulli) random variables 0 and 1 with the same probability. For a sequence of independent identically distributed Bernoulli random variables $x(nNK_D)$ we can define the variable

$$S_n = X_1 + \ldots + X_n, \tag{10}$$

where $X_n = 2x(nNK_D) - 1$ are antipodally encoded values $\{-1, 1\}$. By the classic De Moivre-Laplace theorem [21], for a sufficiently large number of trials, the distribution of the normalized binomial sum

$$s_n = \frac{S_n}{\sqrt{n}} = \frac{X_1 + \ldots + X_n}{\sqrt{n}} \tag{11}$$

is closely approximated by a standard normal distribution $N(0, 1)$, and, roughly said, $|S_n| < 3\sqrt{n}$ for almost[3] all n. Note that we used extremely long TRNG record in order to detect also very small deviations. Results for 74-Gigabit record with decimation factor $N = 1$ (bottom curve a)) and 37-Gigabit record with decimation factor $N = 2$ (upper curve b)) are shown in Figure 7.

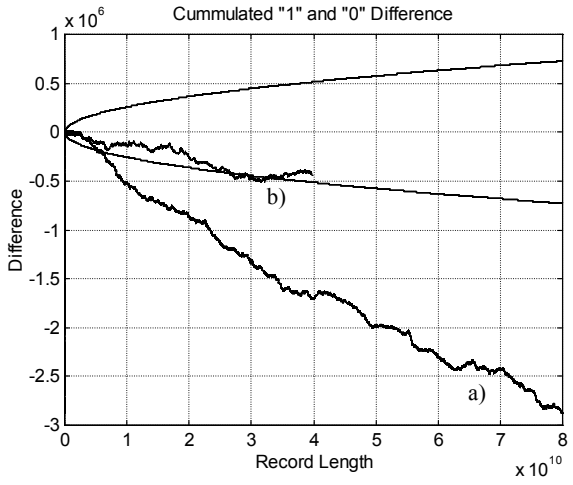


Fig. 7. Results of the Frequency (Monobit) test of very long TRNG outputs: curve a) is for a 74-Gigabit record with the decimation factor $N = 1$; curve b) is for a 37-Gigabit record with the decimation factor $N = 2$

[3] The well-known rule for $N(0, 1)$ distribution states that at 3 standard deviations ($3\sigma = 3$ for $N(0, 1)$) to the right and left, 99.73 % of all data should fall within that range [1].

Deviation for $N = 1$ is clearly visible. After a decimation by the factor $N = 2$, the deviation is within expected values (marked as a parabola with a peak point in 0), but it still remains visible. This is the only observable deviation from the ideal RNG currently known to us. On the other hand, decreasing of the bias by the use of the XOR decimator is applicable only for independent bits. The fact that the decimation operation has reduced the bias, gives us an indirect evidence that the output bits are almost independent and very close to perfectly random bits.

6 CONCLUSIONS

In this paper we have described and evaluated a simplified method of the true random bitstream generation inside modern digital VLSI circuits. The design of the TRNG and the method of randomness extraction guarantee that the output depends on a physical undeterministic internal random process. The randomness of the sequence of numbers has been extensively tested and only extremely small differences from the ideal RNG have been detected.

The proposed solution is very cheap, it uses very small number of logic resources and it is faster than other comparable methods. Although the functionality of the proposed solution has been demonstrated for Altera Apex FPLD family, the same principle can be used for all recent high-performance ASICs or FPLDs that include an on-chip reconfigurable analog PLL(s) for internal clock synthesis.

Since the quality of the random output depends mainly on the parameters of the jitter, a good knowledge of the jitter is very valuable. For this reason, we will concentrate our effort to the real-time jitter measurement and on-line testing in our future research activities.

Acknowledgments

This work has been done in the frame of the project CryptArchi included in the French national program ACI Cryptologie (project number CR/02 2 0041) and the Slovak scientific project VEGA 1/1057/04.

REFERENCES

[1] Menezes, J. A.—Oorschot, P. C.—Vanstone, S. A.: Handbook of Applied Cryptography. New York: CRC Press, October 1996. Available on: http://www.cacr.math.uwaterloo.ca/hac/.

[2] Eastlake, D.—Crocker, S. D.—Schiller, J.: Randomness Recommendations for Security, Internet Engineering Task Force, RFC 1750. December 15, 1994. Available on: http://www.rfc-editor.org/rfc/rfc1750.txt.

[3] Jennewein, T.—Achleitner, U.—Weihs, G.—Weinfurter, H.—Zeilinger, A.: A Fast and Compact Quantum Random Number Generator. Rev. Sci. Inst. 71, 2000, pp. 1675–1680.

[4] Walker, J.: Hotbits: Genuine Random Numbers, Generated by Radioactive Decay. 2002. Available on: http://www.fourmilab.ch/hotbits/.

[5] Faifield, R. C.—Mortenson, R. L.—Coulthart, K. B.: An LSI Random Number Generator (RNG). Lecture Notes in Computer Science, Vol. 0196. Berlin, Germany: Springer-Verlag, 1984, pp. 203–230.

[6] Jun, B.—Kocher, P.: The INTEL Random Number Generator. Cryptography Research, Inc., White Paper prepared for Intel Corporation, April 1999, pp. 1–8. Available on: http://www.cryptography.com/resources/whitepapers/IntelRNG.pdf.

[7] Dichtl, M.: How to Predict the Output of a Hardware Random Number Generator. In: C. D. Walter, C. K. Koc, and C. Paar (Eds.): Workshop on Cryptographic Hardware and Embedded Systems – CHES 2003, Lecture Notes in Computer Science, Vol. 2779. Berlin, Germany: Springer-Verlag, September 2003, pp. 181–188.

[8] Tsoi, K.—Leung, K.—Leong, P.: Compact FPGA-Based True and Pseudo Random Number Generators. In: Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM), California, USA, 2003, pp. 51–61.

[9] Kohlbrenner, P.—Gaj, K.: An Embedded True Random Number Generator for FPGAs. In: Proceeding of the ACM/SIGDA 12th international symposium on Field Programmable Gate Arrays. ACM Press, 2004, pp. 71–78.

[10] Fischer, V.—Drutarovský, M.: True Random Number Generator Embedded in Reconfigurable Hardware. In: B. S. Kaliski, Jr., C. K. Koc, and C. Paar (Eds.): Workshop on Cryptographic Hardware and Embedded Systems – CHES 2002, Lecture Notes in Computer Science, Vol. 2523, Berlin, Germany: Springer-Verlag, August 2002, pp. 415–430.

[11] Using the ClockLock & ClockBoost PLL Features in Apex Devices. Altera Application Note 115, v. 2.3, May 2002, pp. 1–55. Available on: http://www.altera.com.

[12] Using PLLs in Stratix Devices. Altera Application Note 200, v. 1.0, February 2002, pp. 1–70. Available on: http://www.altera.com.

[13] Maneatis, J. G.: Selecting PLLs for ASIC Applications Requires Tradeoffs. Planet Analog Magazine 9/2003. Available on: http://www.planetanalog.com.

[14] XpressArray High Density 0.18 um Structured ASIC. Web site of the AMI Semiconductors Company.
Available on: http://www.amis.com/pdf/xpressarray_hd_datasheet.pdf.

[15] Superior Jitter Management with DLLs. Virtech Tech Topic VTT013, v. 1.2, January 21, 2003, pp. 1–6. Available on: http://www.xilinx.com.

[16] Nios Embedded Processor Development Board. Altera Data Sheet, v. 2.1, April 2002, pp. 1–22. Available on: http://www.altera.com/nios.

[17] Metastability in Altera Devices. Altera Application Note 42, v. 4.0, May 1999, pp. 1–10. Available on: http://www.altera.com.

[18] Davies, R. B.: Exclusive OR (XOR) and Hardware Random Number Generators. February 28, 2002, pp. 1–11.
Available on: http://www.robertnz.net/pdf/xor2.pdf.

[19] Fischer, V.—Drutarovský, M.—Šimka, M.—Bochard, N.: High Performance True Random Number Generator in Altera Stratix FPLDs. In: J. Becker,

M. Platzner, S. Vernalde (Eds.): Field-Programmable Logic and Applications – FPL 2004, Lecture Notes in Computer Science, Vol. 3203. Berlin, Germany: Springer-Verlag, September 2004, pp. 555–564.

[20] Marsaglia, G.: A Battery of Test for Randomness.
Available on: `http://stat.fsu.edu/~geo/diehard.html`.

[21] Rukhin, A.—Soto, J.—Nechvatal, J.—Smid, M.—Barker, E.—Leigh, S.—Levenson, M.—Vangel, M.—Banks, D.—Heckert, A.—Dray, J.—Vo, S.: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22 (revised May 15, 2002). Available on: `http://csrc.nist.gov/rng`.

[22] Killmann, W.—Schindler, W.: A Proposal for: Fuctionality Classes and Evaluation Methodology for True (Physical) Random Number Generators. pp. 1–38, September 2001.
Available on: `http://www.bsi.bund.de/zertifiz/zert/interpr/trngk31e.pdf`.

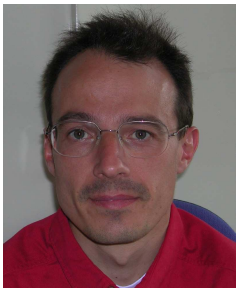
[23] NIST FIPS PUB 140-2, Security Requirements for Cryptographic Modules. Federal Information Processing Standards, National Institute of Standards and Technology, U.S. Department of Commerce, Tech. Rep., May 25, 2001. Available on: `http://csrc.nist.gov/publications/fips`.

[24] Kim, S.—Umeno, K.—Hasegawa, A.: Corrections of the NIST Statistical Test Suite for Randomness. Cryptology ePrint Archive, Report 2004/018, 2004. Available on: `http://eprint.iacr.org`.

Miloš Drutarovský received the M.Sc. degree in radioelectronics and Ph.D. degree in electronics from Technical University of Košice, Slovak Republic, in 1988 and 1995, respectively. He defended his habilitation work – Digital Signal Processors in Digital Signal Processing – in 2000. He is currently working as an Associated Professor at the Department of Electronics and Multimedia Communications, Technical University of Košice. His current research interests include applied cryptography, digital signal processing, and algorithms for embedded cryptographic architectures.

Viktor Fischer received the M.Sc. and Ph.D. degrees in electronics from Technical University of Košice, Slovak Republic, in 1981 and 1991, respectively. From 1982 to 1991 he was an Assistant Professor at the Department of Electronics, Technical University of Košice. Since 1991, he has been working at the Jean Monnet University of Saint-Etienne, France, as an Invited Professor in electronics and computer science. In the Laboratory Traitement du Signal et Instrumentation (TSI), UMR 5516 CNRS/University of Saint-Etienne, he works on signal and image processing, information security and embedded cryptographic systems. He is also currently working with Micronic in Košice, Slovak Republic, a company oriented toward the development and production of data security hardware and software.

Martin Šimka received the MSc degree in electronics and communications from Technical University of Košice, Slovak Republic, in 2002 after defending his Master's Thesis – Conception of connection of embedded processor to arithmetic coprocessor in SOPC Altera. Currently he is a Ph.D. student at the Department of Electronics and Multimedia Communications, Technical University of Košice. His current research interest includes implementation of cryptographic blocks on FPGAs.



Frédéric Celle received the National University Institute of Technology Graduate diploma in electrical engineering and industrial computing and B.Sc. degree in telecommunications from University of Saint-Etienne, France, in 1989 and 1997, respectively. Since 1990, he has been a design engineer at the TSI (Traitement du Signal et Instrumentation) laboratory of the UMR CNRS 5516 in France. He is associated with embedded electronic projects, using FPGA, PCB prototyping, and VHDL programming for real time processing projects.