FPGA Implementation of Toeplitz Hashing Extractor for Real Time Post-processing of Raw Random Numbers

Xiaoguang Zhang, You-Qi Nie, Hao Liang, Jun Zhang

***Abstract*—Random numbers are widely used in many fields. However, most existing random number generators cannot directly generate ideal random bits without post-processing. With the development of generation techniques, the generation rate of raw random data has reached Gbps magnitude and the speed of existing post-processing cannot satisfy the growth of demand. To solve this issue, we propose a concurrent pipeline algorithm based on Toeplitz matrix hashing and implement it in a resource limited field-programmable gate array (FPGA). By taking advantage of the concurrent computation features of FPGA instead of common computer serial computation, the post-processing speed is improved by three orders of magnitudes to above 3.36 Gbps, which is suited for Gbps real time post-processing of raw random numbers. After post-processing, the final extracted random bits can well pass the standard randomness tests. To implement the scheme, a printed circuit board (PCB) is designed for raw data acquisition, real time post-processing and final extracted random data transmission. On the PCB, the random signal is sampled and digitalized as raw random data and then the data are fed into a FPGA for real time post-processing. At the same time, three different transmission interfaces including a small form-factor pluggable (SFP) fiber transceiver, a universal serial bus (USB) 2.0 port and a Gigabit Ethernet port are designed for different scenarios. An optional DDR3 memory module is also provided for testing purpose.**

I. Introduction

RANDOM numbers are widely used in many fields such as statistical analysis, numerical simulation and cryptography. Conventional pseudo-random numbers generated by a deterministic algorithm with a random seed as an input appear to be random but in fact are fully determined and repeated on a long sequence of scales. True random numbers are unpredictable, irreproducible, unbiased and generated from a physical system. Up to present, various schemes have been proposed to generate true random numbers[1–9]. However, most existing random number generators cannot directly generate ideal random bits without post-processing (or randomness extraction), where complicated mathematical operation is applied. There are several kinds of post-processing, simple ones widely employed in the past such as making bitwise exclusive-OR operation between two raw biased sequences to obtain an unbiased random sequence[2] or applying non-Universal hashing functions on raw sequence to distill randomness[5] are not informational-theoretically provable. Recently new approaches using Universal hashing functions such as Toeplitz hashing are proposed and verified[8–10], which is informational-theoretically provable. The Toeplitz hashing randomness extractor requires a large amount of computations, when implemented in computer serial computations will consume a lot of time and therefore can only get a relatively low processing speed of 1.6 Mbps[9] or 441 kbps[8]. While, with the development of device and data acquisition techniques, the speed of raw random number generation has been increased up to Gbps magnitude[3, 8, 9] and existing randomness extraction speed cannot satisfy the growth of demand.

To solve this issue, in this paper, by carefully investigating the calculation law of the Toeplitz hashing extractor, we propose a concurrent pipeline post-processing algorithm that successfully transplant the Toeplitz hashing extractor from computer to a resource limited field-programmable gate array (FPGA). By taking advantage of the concurrent computation features of FPGA instead of common computer serial computation, the post-processing speed is greatly improved by three orders of magnitudes to above 3.36 Gbps, which is suited for Gbps real time post-processing of raw random numbers. After post-processing, the final extracted random bits can well pass the standard randomness tests. To implement the Toeplitz hashing randomness extractor, a printed circuit board (PCB) is designed for raw data acquisition, real time post-processing and final extracted random data transmission. On the PCB, the random signal is sampled and digitalized as raw random data by an 8-bit analog to digital converter (ADC) and the raw data are then fed into a high performance FPGA for real time post-processing. At the same time, a small form-factor pluggable (SFP) fiber transceiver is employed to output the final random bits at a real time speed of 3.2 Gbps, optional interfaces including a universal serial bus (USB) 2.0 port and a Gigabit Ethernet port are also provided for different scenarios as well as corresponding demo software. An optional DDR3 memory module that can store large amounts of continuous

Manuscript received May 29, 2016. This work was supported in part by the National Basic Research Program of China Grant No. 2013CB336800, the National Natural Science Foundation of China Grant No. 61275121, and the Chinese Academy of Sciences.

Xiaoguang Zhang and Hao Liang are with the Department of Modern Physics, University of Science and Technology of China and the State Key Laboratory of Technologies of Particle Detection & Electronics, Hefei, Anhui 230026, China (e-mail: zxg@mail.ustc.edu.cn; simonlh@ustc.edu.cn).

You-Qi Nie and Jun Zhang are with the Hefei National Laboratory for Physical Sciences at the Microscale, University of Science and Technology of China and the CAS Center for Excellence and Synergetic Innovation Center in Quantum Information and Quantum Physics, Hefei, Anhui 230026, China (e-mail: nyqyh@mail.ustc.edu.cn; zhangjun@ustc.edu.cn).

978-1-5090-2014-0/16/$31.00

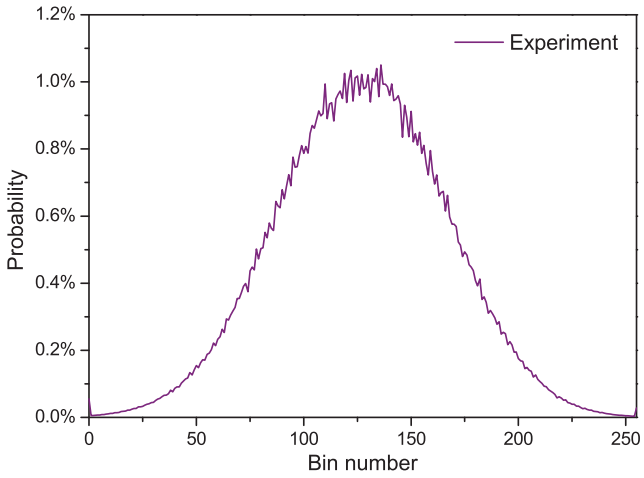

Fig. 1. Probability distribution in 2^8 (256) bins for 10^8 raw samples.

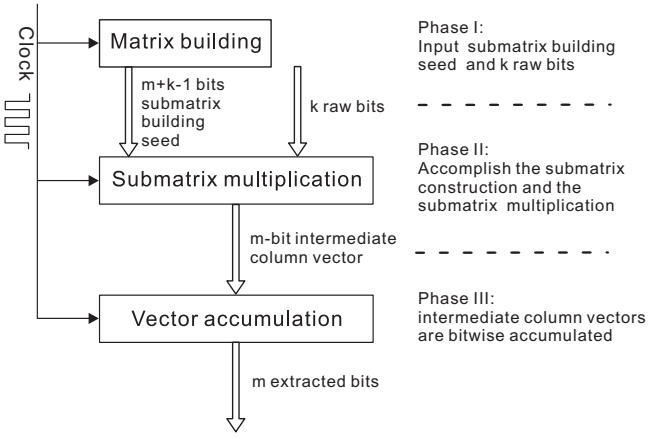

Fig. 3. FPGA implementation of Toeplitz hashing randomness extractor.

data for testing purpose is also provided.

II. Algorithm of Real Time Post-processing

A. Min-entropy Evaluation

To determine the extraction ratio that one can extract from the original raw random data to the final unbiased true random bits, min-entropy (or randomness) evaluation of raw samples is employed[8–10]. The Min-entropy is defined as

$$H_\infty(X) = -\log_2(\max_{x\in\{0,1\}^N} P_r[X=x]), \qquad (1)$$

which quantifies the amount of randomness of a given probability distribution X on $\{0,1\}^N$. In terms of our system, N equals 8 because the ADC resolution is 8-bit, and all the raw samples will be mapped in 2^8 (256) bins, thus form a probability distribution which is shown in Fig. 1. According to Eq. 1, the min-entropy is determined by the sample point x with maximal probability. By means of the variance of raw samples and the assumption that raw samples follow Gaussian distribution, min-entropy can be easily evaluated. The min-entropy measured by our system is 0.812 bits per bit, it means that we can extract as much as 81.2% randomness from raw random numbers.

B. General Ideas to Realize the Toeplitz Hashing Extractor in FPGA

After the min-entropy evaluation, the next step is applying a Toeplitz hashing randomness extractor to distill the raw random numbers. Here we introduce the procedure in brief, while the rigorous discussion about Teoplitz hashing extractor can be found in reference [10]. As is shown in Fig. 2(a), m extracted random bits are extracted by multiplying n raw bits by a binary Toeplitz matrix of a size of $m \times n$. The matrix is constructed by a sequence of only $m+n-1$ random bits (matrix building seed) for its characteristic that all the elements on the same diagonal descending from left to right are the same. In our scheme, we choose $m = 1024$ and $n = 1520$, so the extraction ratio is 67.4% and is smaller than 81.2% prescribed by the evaluated min-entropy.

In order to improve the post-processing speed, the Toeplitz hashing extractor is implemented in FPGA rather than in computer, taking advantage of the hardware concurrent computation capability in FPGA. But it is impossible to directly transplant such a large matrix extractor to FPGA due to the limitation of resources in FPGA, so we propose a concurrent pipeline algorithm to achieve it. Fig. 2 shows the general idea. The entire large Toeplitz matrix extractor is evenly decomposed into n/k (n/k is an integer) small sub-matrix multiplication steps. For each step, the multiplication between $m\times k$ binary matrix and k raw bits is accomplished within one clock period with every row calculations of the multiplication concurrently carried out. Different steps are accomplished in a sequential order and therefore all steps totally take n/k clock periods. The n/k created intermediate m-bit column vectors of each step are accumulated to generate m final extracted random bits. The whole procedure is called an extraction-period. When one extraction-period finishes, another period starts. It is noted that all steps have the same computing structure so they can be sequentially accomplished in a shared time-division multiplexing computing unit. By employing such time-division multiplexing computing unit and concurrent pipeline mode calculation structure, required FPGA resources are substantially reduced and the large matrix randomness extractor can be successfully realized.

C. Concrete Realization of the Algorithm in FPGA

According to the above ideas, in terms of specific implementation, we choose $k = 80$ and design three computing units in FPGA working in a pipeline mode as is shown in Fig. 3. These units are working with a synchronized clock of 62.5 MHz. The $m+n-1$ (2543) bits seed required to construct the entire Toeplitz-matrix are stored in the matrix building unit. In every clock period the matrix building unit will select $m+k-1$ (1103) bits sub-seed and send them to sub-matrix multiplication unit to construct the corresponding sub-matrix along with the process of the pipeline calculation. The sub-matrix multiplication unit accomplishes the sub-matrix construction and the multiplication between the sub-matrix and input k raw bits and then output m-bit intermediate column vector. Note that the sub-matrix is constructed by the

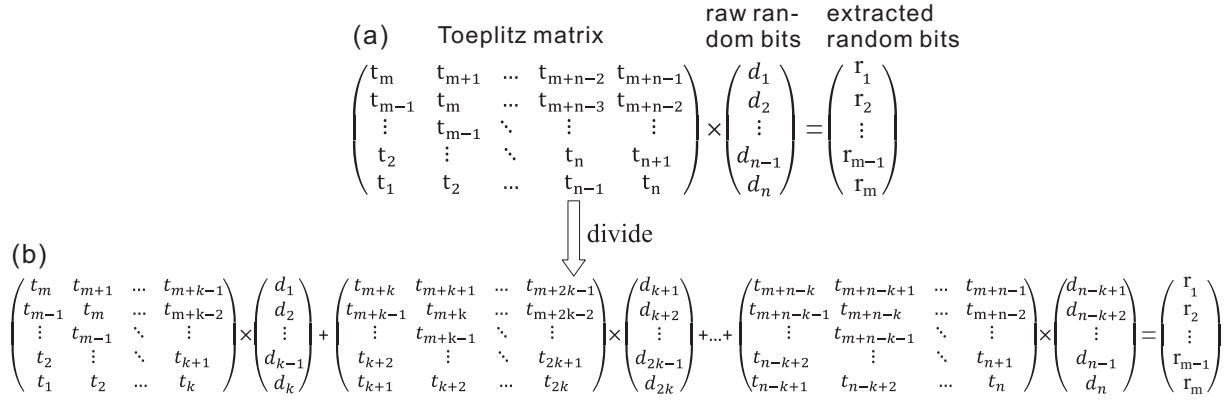

Fig. 2. General idea of the way to transplant Teoplitz hashing extractor from computer to a resource limited FPGA.

method of signal fanout. All the n/k (19) intermediate column vectors generated in the same extraction-period are bitwise accumulated in the vector accumulation unit to generate the m final bits, which are the extracted bits from n raw bits. We should point out that binary multiplications are realized by bitwise-AND and binary additions by bitwise exclusive-OR (XOR). In order to fulfill the timing request in FPGA, we also take a number of measures such as register-copy to reduce excessive fanouts and some timing constraints. With such a configuration, high speed Toeplitz hashing extractor is successfully achieved in a FPGA, thus real time generation rate of unbiased true random bits reaches 3.36 Gbps (5 Gbps $\times 1024/1520 \sim$ 3.36 Gbps).

III. Hardware Design of Data Acquisition & Post-processing System

To implement the scheme, a printed circuit board (PCB) is designed for raw data acquisition, real time post-processing and final extracted random data transmission. The schematic block diagram of the board is illustrated in Fig. 4. The core of the board is a high performance FPGA (Xilinx, Virtex-6) which acts as system controller and data processor. Input signal is digitized at sampling rates of 1 GSPS by an 8-bit ADC (TI, ADC083000), which is clocked by a high performance frequency synthesizer system with integrated VCO. Three different data interfaces are available for various application scenarios including an integrated 10/100/1000 Mbps Gigabit Ethernet transceiver configured as the Gigabit Media Independent Interface (GMII) mode, two 4.0 Gbps bi-directional small form-factor pluggable (SFP) fiber transceivers and a universal serial bus (USB) 2.0 port. An optional 2GB DDR3 memory module that can store large amounts of continuous data for testing purpose is also provided. Besides, we also develop a dedicate demo software to flexibly control the system by computer.

The signal flow is also shown in Fig. 4. Input random signal is amplified by an operational amplifier to fit the input range of the ADC. After sampling and digitization ADC parallel output 4 samples (32 bits) in double-data-rate (DDR) mode as well as the 125 MHz data synchronous clock. When entering FPGA, the 32 raw random bits are restored to 64 parallel bits of single-data-rate (SDR) mode by exploiting the IDDR primitive of FPGA. For every sample, the highest two bits whose randomness is not good and the lowest bit that is overwhelmed in the electronics noises are discarded by the data select & deserializer module. The remained 40 bits are further deserialized as 80 bits in this module to lower speed. Meanwhile the 125 MHz data synchronous clock is frequency divided to 62.5 MHz by an integrated PLL in FPGA to keep data synchronization. Then the 80 raw random bits out of 16 samples are sent to Toeplitz hashing extractor module to generate extracted unbiased random bits. After extracted, data rate reaches about 3.36 Gbps. The extracted data are then sent to series of FIFOs for clock domain switch between ADC clock domain and transceivers domains. For each transmission ports, a corresponding interface control module is designed as well as a command decoder module which is used to identify external commands.

TABLE I
Key parameters of the ADC in the data acquisition & post-processing system.

SNR	SINAD	THD	SFDR	ENOB
43.8 dB	42.0 dB	-46.7 dB	47.7 dB	6.68 bits

IV. Test Results

A. Electronics Tests

Effective Number of Bits (ENOB) of ADC is the most important parameter of a data acquisition system, which decides the effective resolution for a concerned signal. To measure the ENOB as well as other important parameters of ADC including Signal to Noise Ratio (SNR), Signal to Noise Plus Distortion Ratio (SINAD), Total Harmonic Distortion (THD) and Spurious-Free dynamic Range (SFDR), a function/arbitrary waveform generator Tektronix AFG 3252C is employed. The output sine waveform of the generator was sampled and digitalized by ADC, and then the sampled data are continuously stored in the DDR3 module until it is full. Then the stored data are transmitted to a computer for performance testing. The FFT spectrum of the sampled data with fin=20 MHz and fsample=1 GSPS is shown in Fig. 4, from which the key parameters are calculated and shown in Table I.

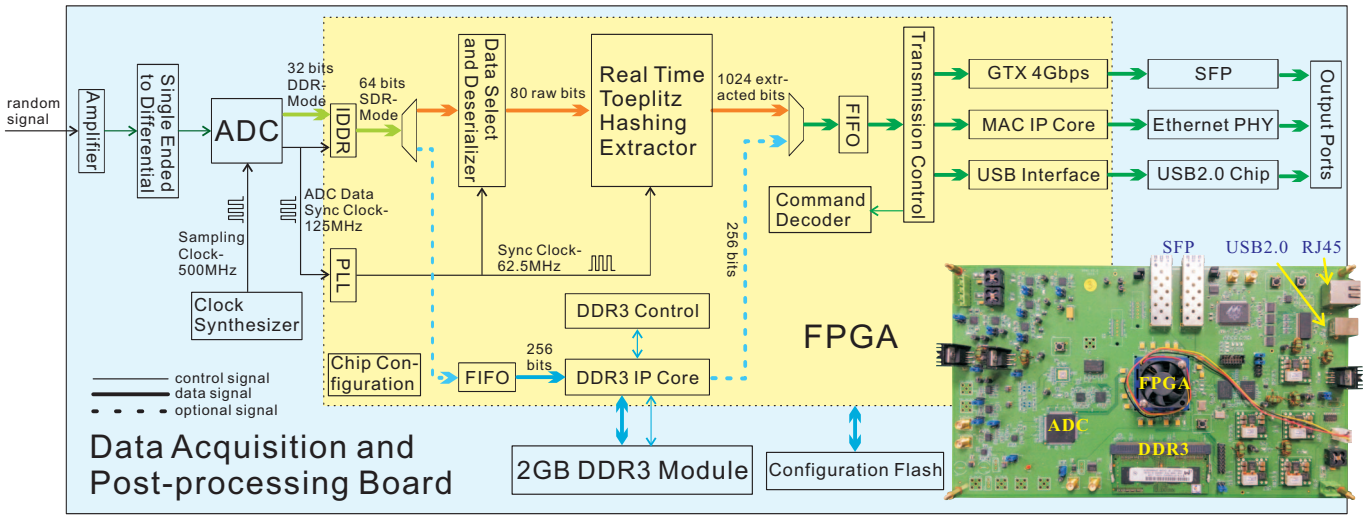

Fig. 4. Schematic block diagram of the data acquisition and post-processing board.

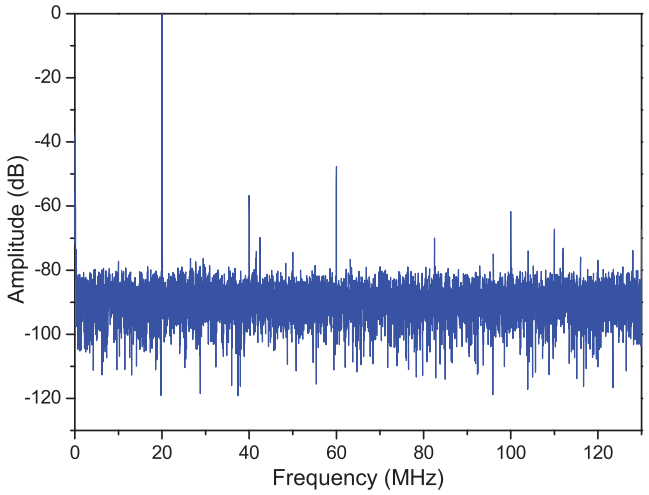

Fig. 5. FFT spectrum of sampled sine wave with fin=20 MHz and fsample=1 GSPS.

In order to test the transmission correctness of Gigabit Ethernet transceiver and USB 2.0, we stored 2 GB data of linear values in DDR3 module and transfer them to computer via Ethernet and USB 2.0 ports with limit speed. A parity check shows no error exists in the transmission. For SFP, we employ Xilinx dedicate software, IBERT, to test the bit error rate (BER) of a Pseudo-random Binary Sequence (PRBS) after propagated in a multi-mode fiber of 10 meters. For the continuously transmitted data of 3.6×10^{13} bits, no error is detected, and the BER reaches an extremely low level of 2.8×10^{-14}.

There are three approaches to export the extracted random bits. Their limit real time transmission speeds are tested respectively. For SFP, the IBERT shows that the transmission speed is stable at 4 Gbps, which meets expectations because its driver GTX of FPGA is designed to be working at this speed. Note that the data is 8B/10B encoded in by GTX, thus the effective transmission speed of extracted random bits is 3.2 Gbps after decoded by the receiver. When testing the

TABLE II
REAL TIME EXPERIMENTAL TRANSMISSION SPEEDS OF DIFFERENT INTERFACES.

Interface	SFP	Gigabit Ethernet transceiver	USB 2.0
Speed	3.2 Gbps	968.7 Mbps	259.5 Mbps

Gigabit Ethernet port, in order to avoid the impact of the relatively low write speed of mechanical hard disk, a standard computer with a Solid State Drives (SSD) is used as the testing terminal. We continuously upload random numbers by 97.6 GB (104,857,600,000 Bytes) and it consumes 866 seconds, therefore, the real time transmission speed of true random numbers via Gigabit Ethernet transceiver is about 968.7 Mbps. The experimental speed of USB 2.0 port is 259.5 Mbps with similar testing method. Table II summarizes the experimental speeds of the three output interfaces.

B. Post-processing Tests

For ideal unbiased random data, the distribution is completely uniform. Compared with the probability distribution of raw samples shown in Fig. 1, in our test, the output extracted random data from Toeplitz hashing extractor closely follow a uniform distribution, as is shown in Fig. 6. It suggests that the Toeplitz hashing extractor in FPGA is effective. Note that every data is merged by eight extracted random bits so all of the data fall into 2^8 (256) bins. To further examine the effectiveness of the post-processing Toeplitz hashing extractor in FPGA, we make an autocorrelation analysis for 10^7 raw samples and 10^7 extracted random bits, as is shown in Fig. 7. For raw samples, the autocorrelation coefficients are relatively large. While for extracted random bits, the autocorrelation coefficients are significantly reduced, indicating that the autocorrelation existing in the raw data is eliminated by the Toeplitz hashing extractor. Finally, to more strictly examine and verify the randomness of final extracted random bits, we apply the standard NIST statistical tests[11] with 1 Gb data and the results show that the final extracted random bits can well pass the NIST tests.

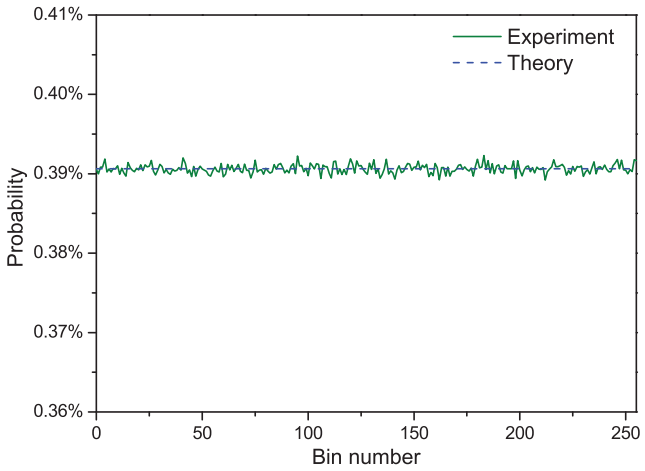

Fig. 6. Experimental (solid line) vs. theoretical (dashed line) probability distribution in 2^8 (256) bins for 800 Mb extracted random data.

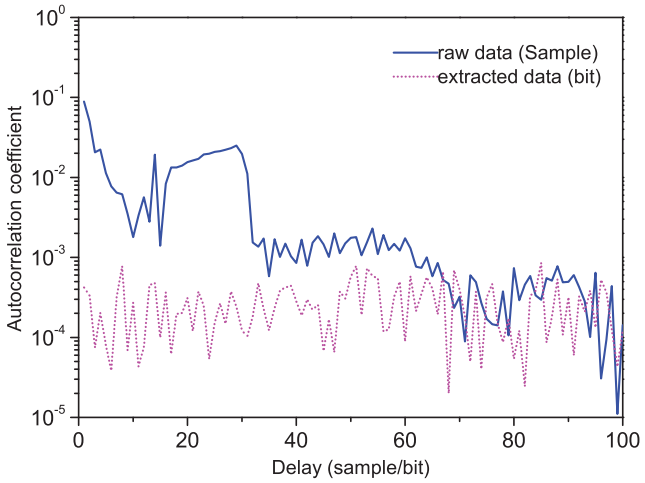

Fig. 7. Autocorrelation analysis for 10^7 raw samples (solid line) and 10^7 extracted random bits (dashed line).

V. Conclusion

In summary, we develop a concurrent pipeline algorithm based on Toeplitz hashing post-processing and implement it in a resource limited FPGA. By taking advantage of the concurrent computation features of FPGA, the post-processing speed is greatly improved by three orders of magnitudes to above 3.36 Gbps. The general idea is decomposing the entire large Toeplitz matrix extractor into several sub-matrix multiplications steps and accomplishing the steps by a shared time-division multiplexing unit in a pipeline mode. Such time-division multiplexing concurrent pipeline calculation structure substantially reduces the required resources of FPGA, thus the large matrix randomness extractor can be realized.

To implement the scheme, a PCB is designed for raw data acquisition, real time post-processing and final extracted random data transmission. On the PCB, the random signal is sampled and digitalized as raw random data by a high speed 8-bit ADC and the raw data are then fed into a high performance FPGA for real time post-processing. At the same time, a SFP fiber transceiver is employed to output the final extracted random bits at a real time speed of 3.2 Gbps, optional interfaces including a universal serial bus (USB) 2.0 port and a Gigabit Ethernet port can also real time output data at 259.5 Mbps and 968.7 Mbps, respectively. Besides, an optional DDR3 memory module is also provided for testing purpose. After real time post-processing, the final extracted random bits can well pass the standard randomness tests.

References

[1] C. S. Petrie, J. A. Connelly, *A noise-based IC random number generator for applications in cryptography*. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 47(5), 615-621, 2000.

[2] M. Epstein, L. Hars, et al., *Design and Implementation of a True Random Number Generator Based on Digital Circuit Artifacts*. Cryptographic Hardware and Embedded Systems-CHES 2003, 152-165, 2003.

[3] I. Reidler, Y. Aviad, et al., *Ultrahigh-Speed Random Number Generation Based on a Chaotic Semiconductor Laser*. Physical review letters, 103(2), 024102, 2009.

[4] A. Stefanov, N. Gisin, et al., *Optical quantum random number generator*. J. Mod.Opt., 47(4), 595-598, 2000.

[5] M. A. Wayne and P. G. Kwiat, *Low-bias high-speed quantum random number generator via shaped optical pulses*. Opt. Express, 18(9), 9351-9357, 2010.

[6] Y. Nie, H. Zhang, et al., *Practical and fast quantum random number generation based on photon arrival time relative to external reference*. Appl. Phys. Lett., 104(5), 051110, 2014.

[7] C. Gabriel, C. Wittmann, et al., *A generator for unique quantum random numbers based on vacuum states*. Nat. Photonics, 4(10), 711-715, 2010.

[8] F. Xu, B. Qi, et al., *Ultrafast quantum random number generation based on quantum phase fluctuations*. Opt. Express, 20(11), 12366-12377, 2012.

[9] Y. Nie, L. Huang, et al., *The generation of 68 Gbps quantum random number by measuring laser phase fluctuations*. Rev.Sci. Instrum., 86(6), 063105, 2015.

[10] X. Ma, F. Xu, et al., *Postprocessing for quantum random number generators: entropy evaluation and randomness extraction*. Phys. Rev. A, 87(6), 062327, 2013.

[11] 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, NIST, Special Publication 14, Revision 1a, 2010.