Low-bias high-speed quantum random number generator via shaped optical pulses

Michael A. Wayne* and Paul G. Kwiat

Department of Physics, University of Illinois at Urbana-Champaign,
1110 W. Green Street, Urbana IL, 61801, USA
*mwayne@illinois.edu

Abstract: We present an optical quantum random number generator (QRNG) based on the digitized time interval between random photon arrivals. By tailoring the photon flux of the laser diode, the statistics of the waiting-time distribution are altered to approximate the ideal, uniform case. This greatly reduces the need for post-processing, and enables fast, secure quantum random number generation at rates exceeding 110 Mbit/s.

©2010 Optical Society of America

References and links

1. A. Stefanov, N. Gisin, L. Guinnard, and H. Zbinden, "Optical quantum random number generator," J. Mod. Opt. **47**(4), 595–598 (2000).
2. T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger, "A fast and compact quantum random number generator," Rev. Sci. Instrum. **71**(4), 1675–1680 (2000).
3. W. Dultz, and E. Hildebrandt, "Optical random-check generator based on the individual photon statistics at the optical beam divider," PCT Patent WO/98/16008, April 1998.
4. J. G. Rarity, P. C. M. Owens, and P. R. Tapster, "Quantum random-number generation and key sharing," J. Mod. Opt. **41**(12), 2435–2444 (1994).
5. M. A. Wayne, E. R. Jeffrey, G. M. Akselrod, and P. G. Kwiat, "Photon arrival time quantum random number generation," J. Mod. Opt. **56**(4), 516–522 (2009).
6. N. Lutkenhaus, J. Cohen, and H. Lo, "Efficient use of detectors for random number generation," U.S. Patent 7197523, March 27, 2007.
7. P. Kwiat, E. Jeffrey, and J. Altepeter, "Quantum random number generator," U.S. Patent Application 20060010182, January 12, 2006.
8. D. L. Snyder, and M. I. Miller, *Random Point Processes in Time and Space,* (New York, NY: Springer-Verlag, 1991).
9. E. Ph. D. Jeffrey, Thesis, University of Illinois at Urbana-Champaign, Urbana, IL, 2007.
10. C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal,* vol. 27, pp. 379–423 and 623–656, (1948).
11. A. Rènyi, "On measures of information and entropy," in *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics, and Probability,* 1960, pp. 547–561.
12. National Institute of Standards and Technology, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," April 2009. [Online]. Available: *http://csrc.nist.gov/publications/nistpubs/800-22-rev1/SP800-22rev1.pdf*.
13. A. R. Dixon, J. F. Dynes, Z. L. Yuan, A. W. Sharpe, A. J. Bennett, and A. J. Shields, "Ultrashort dead time of photon-counting InGaAs avalanche photodiodes," Appl. Phys. Lett. **94**(23), 231113 (2009).
14. I. Reidler, Y. Aviad, M. Rosenbluh, and I. Kanter, "Ultrahigh-speed random number generation based on a chaotic semiconductor laser," Phys. Rev. Lett. **103**(2), 024102–024104 (2009).

1. Introduction

Random number generators are used in a variety of applications, including encryption, statistical analysis, and numerical simulations. Due to the large speed advantage, a pseudo-random number generator, a deterministic method which mimics the expected behavior of a truly random source, is often used. Select applications, however, such as quantum cryptography, require a *true* source of randomness. One such source is a quantum random number generator (QRNG). A QRNG exploits the inherent randomness present in quantum processes to create random numbers. Most previous implementations [1–4] relied on the behavior of an incoming photon at a beam-splitter to generate data. These approaches have the

significant drawback, however, that each photon can generate at most one bit of data. Additionally, these systems are heavily limited by the saturation rate of the single-photon detectors.

Recently we presented a QRNG which used the photon arrival *time* as the quantum random variable [5]. As originally proposed [6,7], the time between successive photons is divided into time-bins, which are created by a high-resolution counter operating in parallel with the detector. A given detection time interval can therefore provide *multiple* random bits per detection event. Using 5-ns time-bins, a detection rate of 11 Mc/s, and Xilinx Spartan-3 FPGA technology, we were able to achieve a final random number generation rate of 40 Mbit/s, as shown in Fig. 1.
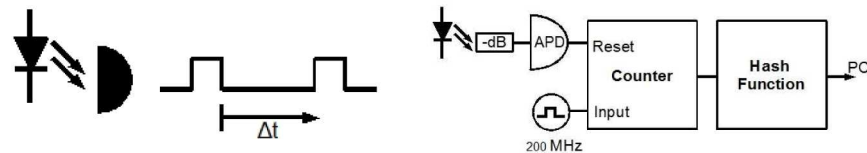

Fig. 1. (a) Time intervals between successive registered photons are translated into time-bin values. (b) Data-flow diagram of our previous constant-current implementation [5]. Photons emitted from the laser diode cause the APD to output pulses which are registered by the counter. These counts are then accumulated until enough entropy is present to 'whiten' using the SHA-256 hash function. The final data is then output through the PCI bus.

Although this implementation has a rate substantially higher than all previous QRNGs, it has a significant potential drawback in that the "random" data contains an unwanted bias: Because the photon statistics for a laser diode operating above threshold are Poissonian, the waiting-time distribution takes the shape of a decaying exponential. Specifically, for a system with a known rate parameter λ, the resulting waiting-time probability distribution is of the form $\lambda e^{-\lambda t}$. For most applications, however, a *uniform* probability distribution is desired. The bias must therefore be washed out by "whitening" the data, typically with a hash function. This then has the caveat of placing a large amount of trust in a possibly insecure hash function, in addition to obtaining lower random bit rates.

The focus of this article is largely on our efforts to reduce the amount of post-processing required to create quality random numbers. By temporally controlling the rate parameter λ, we are able to alter the incoming photon statistics to approximate a uniform waiting-time distribution. This greatly reduces the amount of hashing required, and places more trust into the better understood physics of the laser diode, which increases the system security and allows for better characterization of the amount of output entropy. Additional improvements have also been implemented in the supporting electronics, which enable our system to output quality, secure data at an average rate of 112 Mbit/s. In Section 2 we describe the theory of operation of our scheme, and derive the optimal pulse shape. The metric of min-entropy is discussed in Section 3, while in Section 4 we present the details of our experimental setup; Section 5 describes the various randomness tests used to characterize the final data. Finally, in Section 6 we conclude with a discussion of methods to further improve our implementation.

2. Theory of operation

The randomness in our QRNG system is extracted from the arrival time of successive photons emitted from a laser operating above threshold. This process is believed to be one for which events occur continuously and independently, i.e., a Poissonian process. Our previous constant-current implementation [5] operated by measuring the waiting-time distribution of photons from a source with a *constant* average emission rate. A Poissonian process for which the average rate of events λ does not vary is a *homogenous* Poissonian process, and the number of events k in the time interval [t, t + τ] is characterized by the equation $P[N(t+\tau)-N(t)=k] = e^{-\lambda t}(\lambda t)^k / k!$ Furthermore, since all events are independent,

the time between arrivals is the same as the time until the *first* arrival, and the associated probability distribution is given by the equation $P[N(t)-N(0)=0]=e^{-\lambda t}(\lambda t)^0/0!=e^{-\lambda t}$.

As shown in Fig. 2(a), the waiting-time distribution is a decaying exponential with average value λ. As such, the entropy associated with each detection will be less than that if the distribution were simply uniform, since the earlier time-bins are obviously more likely. In order to compensate for this, a data hashing technique must be used to whiten the random number string, thereby preparing a shorter but more random string, with entropy approaching the ideal case of one random bit per bit, as shown in Fig. 2(b).
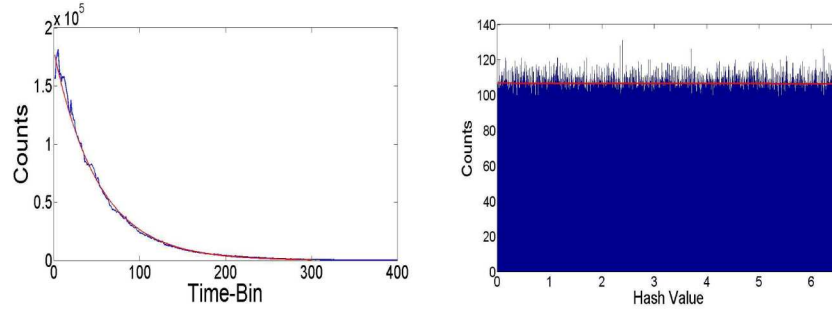
Fig. 2. Pre – (a) and Post – (b) processed random data for our previous constant-current QRNG implementation [5]. Red curves are the theoretical expectations for infinite-size data sets. Bias due to the Poissonian nature of our source was compensated for by whitening via the SHA-256 hash function.

While the whitened data has an average entropy of 0.999996 random bits per bit, the post-processing required is computationally expensive. Therefore, instead of relying on a hash function to whiten the data *after* it is produced, we have developed a technique to efficiently do so *before*. In particular, it has been shown [6,7] that by temporally shaping the photon flux incident on the detector, the counting statistics can be altered, and the waiting-time distribution can be tailored to approximate the ideal, uniform case.

To determine how we need to modify the photon flux, we must now consider an *inhomogeneous* Poissonian process [8]. In this case, our rate parameter is dependent on time, and the expected number of events between times a and b is $\lambda_{a,b}=\int_a^b \lambda(t)dt$. Consequently, the waiting-time distribution is now given by $\lambda(t)e^{-\int_a^b \lambda(t')dt'}$. Given a waiting-time distribution with R possible time-bins, the ideal case is one for which the probability of every bin is time-independent and exactly 1/R. Therefore, ideally λ(t) must be a solution to the equation $\lambda(t)e^{-\int_0^t \lambda(t')dt'}=1/R$. A rate parameter of the form 1/(T-t) is a suitable solution [9], where T corresponds to the user-defined reset period, as discussed further in Section 3.

Since λ(t) represents the photon arrival probability, it is proportional to the photon flux out of the laser diode, which in turn has a linear relationship with the input current. Therefore, if the current is proportional to I(t) = 1/(T-t), then the photon flux will approximate the ideal case. It is important to note that this pulse shape is asymptotic to infinity, so obviously can never be precisely achieved. However, we are able to approximate the shape well enough to see a significant advantage in our random output. By reducing the amount of bias present in the photon statistics, the amount of post-processing can thus be substantially reduced.

3. Min-entropy

As discussed in Section 2, although the underlying process that drives our QRNG may be random, the resulting statistics are not necessarily ideal. When this is the case, it becomes

increasingly important to properly quantify the amount of randomness. Typically, this is done by measuring the amount of information (in bits) contained in the message, or the Shannon entropy [10]. For a message with R possible outcomes, the resulting Shannon entropy is given by the equation $S = -\sum_{i=0}^{R} P_i \log_2 P_i$, where P_i is the probability of the ith outcome. An ideal distribution would have equal probability for each outcome, giving its Shannon entropy the maximum value of exactly $\log_2(R)$ bits. For the Poissonian waiting-time distribution from a constant-intensity light source, however, the most likely time-bin value is always the first. From a security standpoint, this introduces undesirable vulnerability. For example, if an attacker were to have access to the hash function used, they could infer information about its input, and therefore potentially about its output.

Of interest in RNGs, particularly in security applications, is a measure known as min-entropy. Given by the equation $H_\infty = -\log\{\max(P_i)\}$, where $\max(P_i)$ is the probability of the most likely event, the min-entropy is a measure of the worst-case scenario, or the maximum amount of information that can be gained from a single attack [11]. For example, if one time-bin occurred 50% of the time, an adversary consistently guessing that particular value would gain 50% of the total information. Therefore, if we consider the case where our QRNG input is insecure, the min-entropy can be of more importance than the Shannon entropy, and should be used instead when calculating the amount of entropy output. For example, the distribution shown in Fig. 2(a) would give on average 7.08 bits of Shannon entropy per detection, but only 5.65 bits of *min*-entropy per detection. This corresponds to 0.79 random bits per bit (defined as the entropy of the measured distribution divided by the entropy of the ideal distribution) of Shannon entropy, but only 0.63 random bits per bit of min-entropy. Therefore, if we were to quantify our output with Shannon entropy, we could be overestimating the randomness of our data. In the ideal shaped-pulse case, the min-entropy *equals* the Shannon entropy, as all R possible time-bin values have the same probability: 1/R. Therefore, by driving our QRNG with the previously discussed 1/(T-t) pulse shape, the waiting-time distribution will approach the ideal case, and the amount of min-entropy generated per detected photon will approach the maximum value of $\log_2(R)$ bits per detection.

Unlike our constant-current implementation where the counter is reset with every incoming detection, this system also requires a user-defined reset period T. Each successive 1/(T-t) pulse is reset either after time T, or after a successful detection, whichever occurs first. The choice of the reset period depends on several factors. Since the entropy-per-detection increases on a logarithmic scale (i.e., to go from 6 to 7 bits requires 64 extra bins, while increasing from 7 to 8 bits requires 128), an optimal reset period may not necessarily be the longest one. As seen in Fig. 4, the optimal period is strongly dependent on the deadtime of the detector. As deadtime decreases, the optimal reset period becomes smaller. Unfortunately, at small reset periods, achieving a reasonable approximation to the ideal optical pulse shape of 1/(T-t) becomes increasingly difficult. The expected entropy per second is determined by multiplying the average entropy per detection and the number of detections per second.
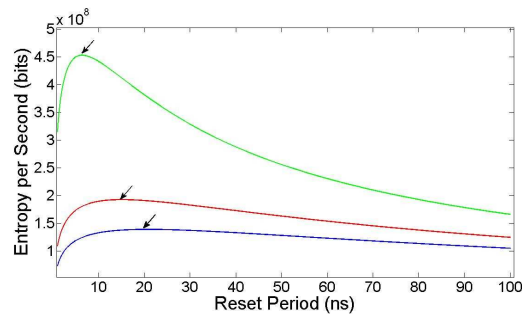

Fig. 3. Peak min-entropy generation rate vs. reset period T for detector deadtimes of 45 ns (dotted blue), 30 ns (dashed red) and 10 ns (solid green), and time-bin resolution of 100 ps. Optimal trigger periods (denoted by arrows) decrease with deadtime.

4. Performance of device

The performance of our QRNG system can be evaluated in one of two ways: the quality of the 'raw' pre-hashed data, i.e., the data corresponding to the waiting-time between detections, and the whitened hashed data (below we discuss why any hashing is needed at all). Because the hash is assumed to be secure, is *designed* to pass random number tests, and has already been evaluated in our previous article [5], the majority of our analysis is focused on the quality of the random numbers output *before* the hash.

The 1/(T-t) pulse shape was approximated with a circuit containing three major components: a sawtooth generator, logarithmic converter, and differentiator. The sawtooth generator provides the (T-t) shape, the logarithmic converter transforms this into ln(T-t), and the differentiator creates the final ~1/(T-t) waveform, as indicated in Fig. 4. Besides generic small 0402 package components, the majority of the circuitry was implemented using OPA847 wideband high-speed operational amplifiers and 1N4148W small-signal diodes. The sawtooth generator was constructed from a triangle-wave generator circuit, with various capacitors to control the speed of the rising and falling edge. The logarithmic converter employed a 1N4148W diode as the feedback element around an OPA847 operational amplifier, and operated in the logarithmic region of the diode. The differentiator was the elementary RC op-amp implementation, but was actually the most problematic component, as the abrupt changes in the waveform resulted in spikes in the derivative. Additional secondary circuitry such as gain stages, output buffers, and voltage followers were also required. Finally, the current from the circuit is used to drive the light source, a 650-nm laser diode operating well above threshold. This serves to eliminate any spontaneous emission noise, as well as creating a near-linear output photon-flux to input current relationship. The laser diode output was attenuated to the single-photon level using a combination of neutral density filters and spatial filtering. Finally, the signal was detected by a fast avalanche photodiode (id Quantique 100-MMF50-ULN). The detector output was then processed by a high speed FPGA (Xilinx ML555 Virtex 5 Evaluation Board).
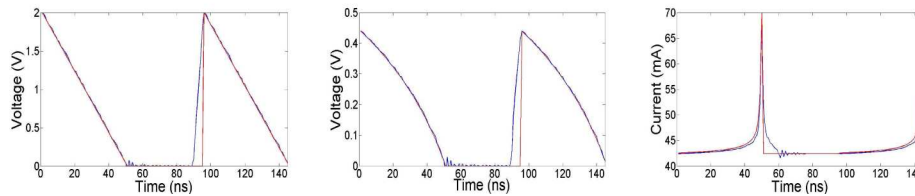

Fig. 4. Ideal (red) and simulated SPICE (blue) waveforms of the (a) sawtooth, (b) logarithmic converter, and (c) differentiator stages of the pulse-shaping circuit assuming a 50-ns reset period. The delay between pulses is to accommodate the 45-ns deadtime of the APD, and the offset on the final waveform is to keep the laser diode operating above threshold.

As discussed in Section 3, the choice of an optimal reset period depends heavily on detector deadtime and time-bin resolution. For our system, with 45-ns deadtime (enough for the APD to return to a steady-state and reduce the probability of after-pulsing) and 160-ps time-bins, the theoretical optimal reset period is 23 ns, which corresponds to a maximum possible rate of 126 Mbit/s. Any deviation from the 1/(T-t) pulse shape will cause the resulting waiting-time distribution to move away from the ideal case, and the min-entropy will decrease accordingly. Achieving a reasonable pulse shape at a 23-ns period proved problematic due to the performance of the designed circuit. Therefore, we tuned the system to have a reset period of 50 ns, which corresponds to a maximum min-entropy of 119 Mbit/s. While slightly slower, the resulting pulse better matched the desired shape and the min-entropy per detection actually increased.

The designed circuit produced a reasonable approximation to the desired pulse shape; however, due to the limited bandwidth of the electronics and the non-linearity of the diode, there were slight variations in the final pulse shape, as shown in Fig. 5(a). Because of this

discrepancy, the resulting waiting-time distribution did not exactly match the ideal [Fig. 5(b)]. The non-ideal regions detract from the min-entropy, but because the behavior is consistent across all runs of the device, we can simply ignore the effected bins, and only count detections arriving during the acceptable part of the distribution. Where we perform this cutoff is determined by characterizing the performance beforehand, and calculating the value for which the min-entropy is maximized. Before truncation, the random data has a min-entropy of 0.9624 bits per bit, versus 0.9984 afterwards. Furthermore, since the bits per detection scales exponentially (i.e., to obtain one extra bit per detection, one would have to *double* the number of available bins), this subtraction does not have a large effect on the final entropy generation rate.
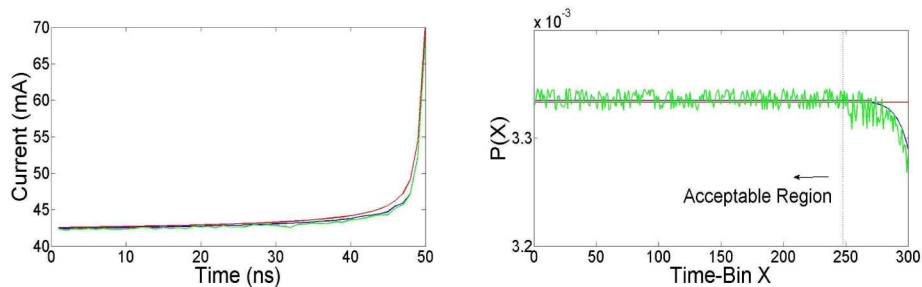

Fig. 5. (a) Theoretical (red), simulated (blue), and actual current pulse shape. The resulting waiting-time distribution (b) from the final QRNG system has a min-entropy of approximately 0.90 bits per bit; however, when counts outside the dotted line are discarded, resulting in a smaller but more random distribution, the min-entropy increases to 0.9984 bits per bit.

Because the truncated raw data still contains a small amount of bias, we whiten it as before, using the SHA-256 hash function. In this instance, however, each detection contains more entropy so it takes fewer detections to fill the hash input buffer to the required amount. Although this means that we must whiten at a faster rate, we rely on the hash less to create high quality random numbers. The raw data's min-entropy of 0.9984 random bits per bit is already suitable for some applications (for comparison, the pre-hashed min-entropy for the constant current implementation, i.e., with no pulse-shaping, was only 0.89 random bits per bit), but the hash function increases this to an average of 0.999996 random bits per bit, as well as washing out any additional effects that may occur within the FPGA. Taking all of this into account, we determine our final entropy generation rate to be 112 Mbit/s.

5. Randomness tests

To test the quality of our random numbers we have run them through many tests (see [5] for details), including the NIST statistical test suite [12], for which they have passed all tests. A 'pass' or 'fail' status indicates that a string of given size has passed or failed *all* of the tests in the suite. Although these provide a good indication that our QRNG behaves randomly within the scope of these tests, it is important to note that there is no true set of tests to check for randomness. Therefore, we rely on the simple and well understood characteristics of our source, which allows for a better understanding and characterization of our entropy.

For testing, we have used a variety of block sizes, ranging from 1 kb to 1 Gb of data. By testing the data across varying size segments, we can detect possible errors over different time scales (e.g., a very short duration effect would not be detected when looking at 1 Gb blocks). The output of each test is a p-value, ranging from zero to one, corresponding to how well the sample data matches the expected behavior. Over a large sample size, the p-values should form a uniform distribution over the interval [0, 1]. One might expect that a good RNG will always have p-values close to 1 (matching the tests almost exactly), but a truly random source will necessarily sometimes appear very "nonrandom". Therefore, across a large number of samples, a *sorted* graph of all occurring p-values should appear as a straight line [12]. In some

sense, how well the measured data matches this line is an indication of how random the data appears.

As with our constant-current system, we tested our data both before and after the hash function. This allows us to not only detect problems with our random source, but also possible concerns with the hashing method. The whitened data passed all random number tests in the suite and gave a uniform distribution of p-values, as shown in Fig. 6. While this gives a good indication that the hash used is suitable for whitening, of much more interest here is how our pre-hashed raw data performs.
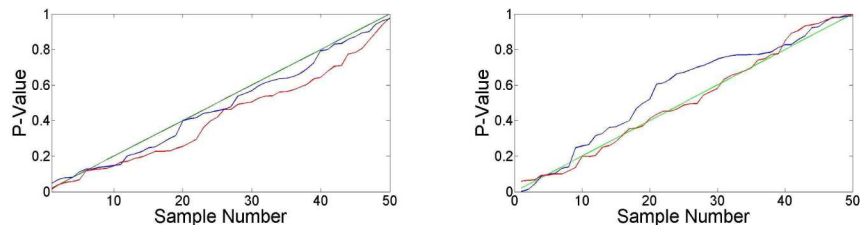

Fig. 6. Example of sorted p-values for (red) raw and (blue) whitened random data versus the expected distribution (green). Data from the hash performed better in the approximate entropy test (a) while in the FFT test (b) the unhashed data actually performed better.

As mentioned in Section 4, after characterizing the performance of our shaped pulse circuit, we truncate the region where the performance is not good enough to ensure high quality random numbers. We tested the raw data *without* truncating the affected area and, as expected, the random tests had a failure rate of over 90%. After truncating, the performance improved markedly, with the random data failing only 5% of the time. However, the uniformity of the sorted p-values was not as high as in the whitened case. Nevertheless, for the Serial, Binary Matrix Rank, and FFT tests, the uniformity was consistently *higher* than with the hashed data. This could suggest that, while not as uniform as the hashed data, our raw data contains fewer correlations than the hashed output, as the mentioned tests look for frequency effects. While of marginally lower quality than the hashed data, the 5% failure rate could be improved with greater precision on our shaped-pulse circuit, eventually leading to performance commensurate with the post-processed data, which passes over 99% of the time.

6. Future improvements and conclusions

The implementation detailed here can be upgraded in several ways – smaller time-bin resolution, higher precision electronics for the laser driving circuit, and shorter-deadtime detectors. At smaller time resolution, one must be careful to not exceed the timing jitter of the detectors, as the data could then be subject to electronic effects within the APD. As shown in Fig. 3, decreasing the detector deadtime greatly increases the maximum entropy generation rate, but would require the laser driver circuit to operate at higher rates. We are currently investigating improvements to our laser driver circuit, which could, in conjunction with faster detectors [13], allow random number generation at rates exceeding 10 Gbit/s.

We have presented a fast quantum random number generator, with reduced bias and the possibility for reduction or even elimination of post processing. Our final entropy generation rate of 112 Mbit/s is, to our knowledge, higher than any previously reported QRNG, and has an advantage over recently introduced chaotic RNGs [14] by relying on a simpler, better understood source of randomness. This allows for a more complete characterization of the random source, e.g., because it should be more robust against changes in the operating environment – which might then be more secure. Furthermore, with improvements to our detectors we should readily achieve random number generation rates comparable to those obtained with the chaotic sources.