Efficient random number generation techniques for CMOS SPAD array based devices

Andrea Stanco, Davide G. Marangon, Giuseppe Vallone, Samuel Burri, Edoardo Charbon, and Paolo Villoresi

1Dipartimento di Ingegneria dell’Informazione, Università degli Studi di Padova, via Gradenigo 6B, 35131 Padova, Italia
2Dipartimento di Fisica e Astronomia, Università degli Studi di Padova, via Marzolo 8, 35131 Padova, Italy
3EPFL, Route Cantionale, 1015 Lausanne, Switzerland
(Dated: October 14, 2019)

This work presents new techniques to produce true random bits by exploiting single photon time of arrival. Two FPGA-based QRNG devices are presented: Randy which uses one discrete SPAD and LinoSPAD which uses a CMOS SPAD array, along with a time-to-digital converter (TDC). Post-processing procedures are explained in order to extract randomness taking care of SPAD and TDC non-idealities. These procedures are based on the application of Peres [1] and Zhou-Bruk [2] extraction algorithms. Achieved generation rates are 1.8 Mbit/s for Randy device and 310 Mbit/s for LinoSPAD device. Randy QRNG also features a real time procedure which was used for the realization of fundamental tests of physics.

I. INTRODUCTION

In very recent years, there has been a widespread interest in random number generators based on physical processes of quantum nature. In fact, these devices, the so-called quantum random number generators (QRNGs), represent the ultimate way to obtain reliable randomness, free from the typical non-random issue that affects algorithmic random number generators. Typically, QRNGs exploit the quantum properties of the optical radiation field in many “recipes”. The different architectures exploit different properties; setups using entangled systems and violating Bell inequalities feature the highest unpredictability in the so-called device-independent (DI) framework [3–7]. Systems that trust the measurement apparatus but not the states of the quantum system or vice versa, the so-called semi-device-independent generators, work under less strict assumptions [8]: for this reason, they feature, in principle, less unpredictability than DI-QRNG but are more feasible to implement and reach even larger generation rates. The last category includes those generators that work in a framework of complete trust in both the quantum system and the measurement apparatus. This means that the absence of side information, exploitable by an adversary, is assumed. The most famous example of this kind of generator is the welcher weg QRNG that produces randomness according to which path a photon takes after interacting with a beam-splitter [9, 10]. In this work we consider the sub-category of trusted QRNG that were developed in order to limit the number of single photon detectors. Random number generation with just one detector is indeed possible by exploiting time as an additional degree of freedom and by leveraging on the statistical features of the photon detection distribution. This could be done with the following procedure: time sampling of the photon detector with a sampling rate almost equal to the photon rate; application of dedicated generation protocols; application of dedicated unbiasing algorithms. In this work we will show that it is possible to generate true random numbers without the application of dedicated generation protocols. Using a higher sampling rate we applied unbiasing algorithms directly to the samples stream achieving higher generation rate of true random numbers. We used this technique on two different systems: the first one, Randy, uses a standard clock to sample the signal of one single photon detector; the second one, LinoSPAD, uses both a standard clock and a time-to-digital converter (TDC) to sample the signals from a matrix of 256 single photon detectors.

The paper is structured as follows: in Sec. II the most common ways to generate random numbers from time will be reviewed. In Sec. III we will introduce our method and we will compare it with the previous ones. The results achieved by applying it to the numbers obtained with a single detector connected to an FPGA will be presented. In Sec. V we will extend this method to a matrix of 256 detectors with time tagging capabilities. In Sec. VI the conclusions will be presented.

II. EXISTING PARADIGMS OF SINGLE DETECTOR QRNGS

We now introduce two elaborated generation protocols, reported in literature, that can extract randomness from a photons distribution.

As a first example, let us consider the generation protocol introduced by Stipčević et al. in [11], denominated here “Diff-QRNG”. The Diff-QRNG comprises: a light source attenuated to single photon level illuminating a single photon detector; a clock that counts the time between the detection events. A binary random variable $X_j = \{0, 1\}$ is obtained by comparing the length of time intervals between three consecutive detections. It is therefore convenient to define the discrete random
variable \( T^j \), associated to the detection instants, such that \( T^j = \{ t^j_0, t^j_1, t^j_2 \} \). Hence, \( X_j \) takes its value \( x_j \) according to the following rule: given \( \Delta T_1 = t_1 - t_0 \) and \( \Delta T_2 = t_2 - t_1 \),

- if \( \Delta T_1 > \Delta T_2 \) then \( x_j = 0 \)
- if \( \Delta T_1 < \Delta T_2 \) then \( x_j = 1 \)
- if \( \Delta T_1 = \Delta T_2 \) then \( x_j = \emptyset \), i.e. no bit is generated.

The rule is iterated so that for the next bit \( b_{j+1} \) the new time interval starts with the end of the previous one, i.e., \( t^j_2 = t^0_{j+1} \). The physical principle that guarantees the identical (\( \Pr[ x_j = 0 ] = \Pr[ x_j = 1 ] \)) and independent (\( \Pr[ x_j | x_j-1 ] = \Pr[ x_j | x_j-2, \cdots, x_1 ] = 1/2 \)) distribution of the bits follows from the memoryless property that characterizes the exponential distribution of the interarrival times \( \Delta T \), namely \( \Pr[ \Delta T_1 > \Delta T_2 ] = \Pr[ \Delta T_2 > \Delta T_1 ] = 1/2 \). As a consequence, a random string \( X = \{ X_1, X_2, \ldots, X_n \} \) with \( n \to \infty \), is characterized by full Shannon entropy, i.e. \( H(X) = 1 \) bit. This implies that the average number of i.i.d. bits that are generated per unit time, is equal to \( r_{\text{i.i.d.}} = r_{\text{phot}}/2 \), where \( r_{\text{phot}} \) is the number of photo-detections per unit time [12].

As a second example, let us consider the generation protocol introduced by Fürst et al. in [13], denominated here “OdEven-QRNG”. As in the previous example, in the OdEven-QRNG a light source attenuated to single photon level illuminates a photomultiplier but in this case a counter enumerates the number of photons detected within a fixed time interval, \( \tau \), corresponding to the period of a sampling signal. Defining \( n^j \) as the number of detections within the interval \( \tau_j \), a random binary variable \( X_j \) assumes its value \( x_j \), with the following rule:

- if \( n^j \mod 2 = 0 \) then \( x_j = 0 \)
- if \( n^j \mod 2 = 1 \) then \( x_j = 1 \),

in other words, the bit value is determined according whether an even or odd number of detections is registered in the time interval \( \tau \). For a Poisson distribution we can write the probability of having an even or odd number of detections:

- \( \Pr[2n] = \sum_{n=0}^{\infty} \frac{(\lambda \tau)^{2n}}{(2n)!} e^{-\lambda \tau} = \frac{1 + e^{-2\lambda \tau}}{2} = \Pr[ x_j = 0 ] \)
- \( \Pr[2n + 1] = \sum_{n=0}^{\infty} \frac{(\lambda \tau)^{2n+1}}{(2n+1)!} e^{-\lambda \tau} = \frac{1 - e^{-2\lambda \tau}}{2} = \Pr[ x_j = 1 ] \)

where \( \lambda \) is the mean number of photons per second and \( \lambda \tau = \langle n_\tau \rangle \) is the mean number of photons per time interval \( \tau \). To avoid a bias in the output, \( \langle n_\tau \rangle \) has to be sufficiently large so that \( \Pr[ x_j = 0 ] \simeq \Pr[ x_j = 1 ] \). Therefore, since \( H(X) \) is a function of \( \langle n_\tau \rangle \), the generation rate is given by \( R = H(X)/\tau \).

\[ x_j = \begin{cases} 1 & \text{if } S_{\text{det}}(j \tau_{\text{clk}}) = U \neq S_{\text{det}}((j-1) \tau_{\text{clk}}) \\ 0 & \text{otherwise} \end{cases} \]

with the index \( j \in \{0,1,2,\ldots\} \). The above rule is equivalent to having \( x_j = 1 \) whenever the clock detects a rising edge of \( S_{\text{det}}(t) \) and \( x_j = 0 \) otherwise. It is worth to be noticed that all existing paradigms of QNRG based on photon time of arrival can be seen as a (non-optimal) post-processing algorithm of the sequence \( X = (x_1, x_2, \cdots, x_n, \cdots) \). We note that the maximum content of randomness that can be extracted by the above

![FIG. 1. A schematic view of Randy system. The SPD links the optical domain to the digital domain by detecting photons from a light source attenuated to single photon level and by sending electrical signals to the FPGA. Through its internal clock, the FPGA samples the SPD events by its own time domain. Clock and SPD rates are not on scale.](image)

**III. THE NOVEL APPROACH**

Our contribution takes into consideration the simplest and most efficient way to generate random numbers by using the temporal degree of freedom without the need to devise complex “rules”. In its essence, the process of random number generation with a single-photon detector, for instance a single-photon avalanche diode (SPAD), can be considered as a process with two signals: a square-wave signal \( S_{\text{det}}(t) \) generated by the single-photon detector that is sampled by a periodic signal \( S_{\text{clk}}(t) \), which is generated by a clock with a period \( \tau_{\text{clk}} \). Without impinging photons \( S_{\text{det}}(t) \) has a typical value \( U \). Given a photo-detection at the time instant \( t_i \), the \( S_{\text{det}}(t) \) toggles its state from \( S_{\text{det}}(t)(t < t_i) = L \) to \( S_{\text{det}}(t)(t_i < t < \tau_{\text{_dead}}) = U \). \( \tau_{\text{_dead}} \) is the specific fixed time interval in which a SPAD keeps the state \( U \) before returning to \( L \) and typically \( \tau_{\text{_dead}} < \tau_{\text{clk}} \), being \( \tau_{\text{_dead}} \) the SPAD dead-time. This means that \( S_{\text{det}}(t) \) toggles back to the \( L \) state before the SPAD being able to detect another photon. The binary random variable \( X_j \) takes its value \( x_j \) at the instant \( j \tau_{\text{clk}} \) according to the following rule:

\[ x_j = \begin{cases} 1 & \text{if } S_{\text{det}}(j \tau_{\text{clk}}) = U \neq S_{\text{det}}((j-1) \tau_{\text{clk}}) \\ 0 & \text{otherwise} \end{cases} \]
Generation rates, Entropy, Probabilities vs. Sampling rate

where \( \rho_0 \) and \( \rho_1 \) are the probability of obtaining \( X = 0 \) and \( X = 1 \) respectively.

It is clear that the maximum generation rate is given by 
\[
r_{\text{max}} = \tau_{\text{clk}}^{-1}
\]
where \( \tau_{\text{clk}} \) is the clock period. Given a mean photon number per second \( \lambda = r_{\text{phot}} \), for a Poisson distribution the probability of no detections within the \( \tau_{\text{clk}} \) interval is equal to \( \Pr[0] = e^{-r_{\text{phot}} \tau_{\text{clk}}} \). To achieve the rate \( r_{\text{max}} \) it is necessary to avoid a bias in the output string \( X \), by tuning the photon detection rate in order to obtain at least one detection within a sampling period, namely \( 1 - \Pr[0] = 1/2 \) from which we obtain \( r_{\text{phot}} = \ln 2 / \tau_{\text{clk}} \). It is therefore clear that the faster the clock rate is, the larger the detection rate should be to keep low the bias in 0. However, \( r_{\text{phot}} \) cannot be set arbitrarily large as it is strongly limited by the detector dead time. Given a fixed \( r_{\text{phot}} \), our idea is to increase the generation rate \( r \) by deliberately increasing the sampling rate and producing a highly biased string with plenty of 0’s and very few 1’s. Then, we re-balance the bias through an optimal post-processing algorithm, introduced by Peres [1]. The Peres algorithm is a revision of the famous von Neumann algorithm [14] and allows to make a biased string uniform while distilling the maximum entropy from that. We give a brief description of it. Given a sequence of samples \( s_w \), the Peres procedure is defined as

\[
\Psi(s_w) = \Psi_N(s_w) || \Psi(\Psi_U(s_w)) || \Psi(\Psi_V(s_w))
\]

where \( \Psi_N(s) \) is the von Neumann algorithm applied to the string, \( \Psi(\Psi_U(s)) \) is the Peres algorithm applied to a new sub-string \( \Psi_U(s) \) and \( \Psi(\Psi_V(s)) \) is the Peres algorithm applied to another new sub-string \( \Psi_V(s) \). \( \Psi_U(s) \) is created by an XOR operation on samples pair while \( \Psi_V(s) \) is created by taking the latest samples of every 00 and 11 pairs. For a full description of the procedure refer to [1]. Clearly, the i.i.d. property comes from the Poisson distribution itself.

The extraction rate after Peres algorithm, under asymptotic assumptions, is given by

\[
r_{\text{r.i.d.}} = \tau_{\text{clk}}^{-1} \times H(X)
\]

As shown in Fig. 2, by fixing \( r_{\text{phot}} \), the extraction rate increases with the sampling rate despite the value of \( H(X) \) (the maximum entropy is reached with a sampling rate equal to \( r_{\text{phot}} / \ln 2 = 288.539 \text{ kHz} \)). These plots confirm the following: with a fixed \( r_{\text{phot}} \), the generation rate is more influenced by the actual input length than by the bias value. Therefore, the optimal choice is to have \( \tau_{\text{clk}} >> r_{\text{phot}} \).

To summarize, any QRNG based on the photon time of arrival can be modelled by a binary signal \( S_{\text{det}}(t) \) sampled by a clock \( S_{\text{clk}}(t) \) that gives an output sequence \( X \). Standard generation protocols, such as Diff-QRNG or OdEven-QRNG, can be applied to \( X \). Nevertheless, these protocols are far from be efficient. We here propose an optimal post-processing able to extract the maximum available entropy from the string \( X \) based on the Peres algorithm. In the next section we will show how to apply our method to physical generators, while taking care of the non-idealities of the detectors.

IV. SINGLE DETECTOR IMPLEMENTATION

We first implemented our method by using a single SPAD shined by an attenuated laser light. We designed a system that is capable to produce true random numbers using different generation protocols as well as Peres algorithm. The system was developed from an FPGA/CPU device [15]. The FPGA allows us a full control over the generation process and a great flexibility in order to easily switch from one protocol to another. We called this design RANDY. A schematic view of the setup is shown in Fig. 1. We set the light source intensity to keep the photon count rate around 200 kcounts/s, due to the SPAD non-linear behaviour on higher rates. As a first implementation we used the default 100 MHz system clock to sample the SPAD signals. The advantages are quite clear since an FPGA allows a full description of the time evolu-
tion of the system: every operation can be described as a multiple of the fundamental time unit $\tau_{\text{clk}}$ defined by the system clock. Therefore, the behaviour of the system is fully deterministic apart from the non-deterministic side due to the true randomness of the photon time of arrival.

The FPGA saves a logical 0 for every clock cycle whenever the SPAD signal is low, a logical 1 otherwise. The raw string is then post-processed on a dedicated CPU. Due to the huge difference between the clock rate and the photon rate (100 MHz over 200 kcounts/s), the raw strings are heavily biased in zero with a percentage of 99.8%. Ideally, the role of Peres algorithm is to reduce the bias to zero. Nevertheless, Peres algorithm cannot work around any correlation. On the contrary, it could emphasize it. Therefore, any correlation has to be removed before the application of Peres algorithm. In our case, correlation comes from afterpulses and dead time of the detectors and we describe how to remove them in the following.

We evaluated the distribution of the time interval between consecutive events. For ideal Poissonian events, this distribution is expected to be a decreasing exponential. As shown in Fig. 3, the experimental distribution differs from the ideal distribution by two effects.

The second effect is the afterpulse. Afterpulses are spurious events that occur randomly within a fixed time interval from a real detection. As a result, they produce a peak which undermines the exponential trend of the events difference distribution. We evaluated a 18 clock cycles (180 ns) cross-point between the only-true-events region and the afterpulses-region.

Dead-time and afterpulses introduce correlations in the output string. Clearly, after a value $x_j = 1$, some of the few following bits in $X$ are not independent. Since we estimated 18 clock cycles as the required time to be in the true-event region, to remove the correlation we may simply remove 18 values of $X$ following any value $x_j = 1$.

As a result, this procedure eliminated a portion of valid random events which we evaluate to be around 14%. However, the resulting distribution of the difference between consecutive 1’s follows the expected decreasing exponential (see Fig. 3 bottom). Moreover, the process eliminates the correlation initially present in the string $X$, as demonstrated by Fig. 4.

Once dead time and afterpulse are compensated, we apply Peres algorithm to the bit string. With an actual photon count rate of 172 kcounts/s and an equivalent sampling frequency of 97 MHz (due to afterpulses and dead time removal), a real time implementation of the Peres algorithm would have yielded a final true random bit rate of 1.8 Mbit/s according to the rates shown in Fig. 2. As shown in Fig. 4 the correlation over the output string is within the limits of statistical acceptance.

Moreover, the two generation protocols Diff-QRNG and OdEven-QRNG described in [11, 13] were also implemented on the same FPGA-system in order to compare their performances with our method. The high time resolution ($r_{\text{phot}} \ll \tau_{\text{clk}}^{-1}$) allows enough precision to identify time differences and to have a good estimation on the detected photons within a fixed intervals. Indeed, the system was successfully used to produce timed random numbers in the work of Vedovato et al. [17]. Of course, the only uncertainty came from the photons time of arrival but it was handled by setting the photon counting rate to a proper value [18]. On the other hand, these protocols have low rate performances: a photon count rate of 200 kcounts/s produce a rate of true random bits of 100 kbit/s for the Diff-QRNG protocol and of 20 kbit/s for the OdEven-QRNG [19].

**V. MUXIPLEXING THE GENERATION RATE**

The main limitation concerning the use of QRNG based on SPADs, i.e. discrete variable QRNG, is the limited generation rate achievable. Typically, SPADs feature maximum count rates of a few Mcps (counts per second). QRNGs based on continuous variable protocols are therefore preferred when it comes to obtain rates in the order of Gbps [20, 21]. However, the recent advancements in miniaturization techniques, and especially the creation of deep-submicron CMOS SPADs, have led to
In this work we consider a sensor of recent introduction, LinoSPAD [27], from the AQUA lab at Delft University & EPFL, which features 256 pixels arranged in four linear arrays of 64 pixels each. The peculiarity of LinoSPAD is a time tagging functionality, which associates a temporal coordinate to every detection, thus potentially enabling a further increase in the generation rate. In the following, we will apply the techniques described in the previous Sections for the single detector case to LinoSPAD.

LinoSPAD features 64 FPGA-based time-to-digital converters (TDCs) [28, 29] that tag the detections of each SPAD in a given bank. Each TDC is implemented by a delay line with 35 carry elements of 4 bits and it is sampled with a frequency $f_{\text{clock}} = 400$ MHz. Every time interval $\tau_{\text{clock}} = 2.5$ ns the TDC emits an output code $b \in \{0, 139\}$. The TDC has therefore a sub-resolution of $\tau_{\text{sub}} = \tau_{\text{clock}}/140 \approx 17.86$ ps and this represents the fundamental time resolution of the system. The following considerations take into account that the actual number of valid pixels is 64 and not 256 due to the limitations to 64 TDCs. The measurements of the photon time of arrival are taken with respect to a “reference” time signal, whose period determines the integration time of a frame. The buffer of the system can add output codes to a maximum of $2^{28}$ bins. Hence, the longest measurable time interval between the reference clock signal and a photon detection cannot be larger than $\tau_{\text{sub}} \cdot 2^{28} \approx 4.8$ ms.

Since the device registers a maximum of 512 tags per pixel during the integration time, every frame is composed at most by 64 x 512 tags. Similarly to the paradigm adopted in the previous Sections, random bits can be extracted directly from the bare physics of the process: a string $\tau_{\text{frame}}/\tau_{\text{sub}}$ bits long is associated to each frame and the 1’s, equal in number to the number of tags, are located according to the tag values. Again, the limit of this approach is that the strings are consistently biased towards zero. This bias is the result of two concurring causes: the first one is the dead time of the SPADs, which being of $\tau_{\text{dead}} \approx 40$ ns, implies that every bit 1 is necessarily followed by $\tau_{\text{dead}}/\tau_{\text{sub}} \approx 2240$ 0’s. However, the 0’s due to the dead time can be removed, as it was previously done. The second cause is the limited buffer size: each string produced in a frame will always feature at most 512 1’s. Indeed, an extraction approach by means of the Peres debiasing procedure could be suitable even for this framework. As in Randy, we studied the interarrival detections time, whose distribution is reported in Fig. 5, in order to detect the artefacts induced by the physical limits of the device.

The histogram starts approximately at 40 ns due to the dead time and a peak starting at 40 ns and extending up to 200 ns indicates the presence of afterpulses. A noticeable feature is an unusual peaks pattern. This pattern was due to a non-linear behaviour of the TDC (further details in subsection V.B). Hence, to manage these non-idealities we decided to separate the tag resolution between coarse resolution (Clock sampling) and fine resolution (TDC) implementing two different post-processing procedures. This distinction allows an easy post-processing procedure since it separates the treatment of non-idealities, i.e. coarse for detectors ones and
The coarse resolution is defined by the 400 MHz system clock. The tag information is related to the number of clock cycles in which an event is detected. Therefore, it describes the temporal distance between an event and the zero reference with a resolution of 2.5 ns. We remove dead time and afterpulses of the SPADs with the same technique described in section III. However, the SPAD arrays suffer from pixel cross-correlation which causes a pixel to output an event when its neighbour receives a photon, i.e. it produces a fake event as in the afterpulse. In order to remove it and to be sure that no cross correlation exists, we discard approximately one third of the pixels of a bank losing all the events from those pixels. This procedure reduces the actual number of pixel from 64 to 22. Once the cross correlation is removed as well as the dead time and the afterpulse, we apply Peres algorithm to every selected pixel independently. As discussed in section III and according to Fig. 2, higher rates can be achieved by preferring longer heavily biased bit strings over shorter slightly biased bit strings. Therefore, we treat every selected pixel as an autonomous QRNG. As in the case of single detector implementation, we evaluated the serial correlation on a sampled bit string and it re-enters within the limit of acceptance after removing dead time, afterpulse and pixel cross-correlation. By summing the bit rate of the selected pixels we obtain a total bit rate of $R_{\text{coarse}} \simeq 87 \text{ Mbit/s}$. The value is averaged since the event rate varies from pixel to pixel as shown in Fig. 6. The mean extraction rate per pixel is equal to $R_{\text{coarse}} = R_{\text{coarse},p} \simeq 1.36 \text{ Mbit/s}$. Considering an ideal SPAD array with no correlation, the hypothetical extraction rate per pixel would be equal to $R_{\text{coarse}}/22 = R_{\text{coarse},p}^* \simeq 3.95 \text{ Mbit/s}$.

Figure 5 shows the distribution of the time differences between successive detections on LinoSPAD device.

**A. COARSE resolution**

The coarse resolution is defined by the 400 MHz system clock. The tag information is related to the number of clock cycles in which an event is detected. Therefore, it describes the temporal distance between an event and the zero reference with a resolution of 2.5 ns. We remove dead time and afterpulses of the SPADs with the same technique described in section III. However, the SPAD arrays suffer from pixel cross-correlation which causes a pixel to output an event when its neighbour receives a photon, i.e. it produces a fake event as in the afterpulse. In order to remove it and to be sure that no cross correlation exists, we discard approximately one third of the pixels of a bank losing all the events from those pixels. This procedure reduces the actual number of pixel from 64 to 22. Once the cross correlation is removed as well as the dead time and the afterpulse, we apply Peres algorithm to every selected pixel independently. As discussed in section III and according to Fig. 2, higher rates can be achieved by preferring longer heavily biased bit strings over shorter slightly biased bit strings. Therefore, we treat every selected pixel as an autonomous QRNG. As in the case of single detector implementation, we evaluated the serial correlation on a sampled bit string and it re-enters within the limit of acceptance after removing dead time, afterpulse and pixel cross-correlation. By summing the bit rate of the selected pixels we obtain a total bit rate of $R_{\text{coarse}} \simeq 87 \text{ Mbit/s}$. The value is averaged since the event rate varies from pixel to pixel as shown in Fig. 6. The mean extraction rate per pixel is equal to $R_{\text{coarse}} = R_{\text{coarse},p} \simeq 1.36 \text{ Mbit/s}$. Considering an ideal SPAD array with no correlation, the hypothetical extraction rate per pixel would be equal to $R_{\text{coarse}}/22 = R_{\text{coarse},p}^* \simeq 3.95 \text{ Mbit/s}$.

**B. FINE resolution**

The fine resolution is defined by the TDC and has a time resolution of approximately 17.86 ps. The TDC outputs a number between 0 and 139 which identifies a precise moment within a clock cycle in which there was an event. Hence, the access to the TDC value is done every clock cycle. As in the coarse resolution, we decide to treat every pixel independently. The peaks of Fig. 5 are separated exactly by 2.5 ns, which can be explained by the presence of a dead time in accessing the TDC. This behaviour brings to the tag distribution shown in Fig. 7 which represents an entire 2.5 ns clock cycle. The figure shows that there are no events in the lower bins and also a significant bias on several values. This behaviour is due to the TDC implementation on an FPGA technology which introduces non-linearity caused by different propagation delays over hardware blocks [30]. It is worth to notice that, while the distribution is not uniform, its entropy is equal to

$$H_{\exp} = - \sum_{k=0}^{139} p_k \log_2 p_k \simeq 6.8 \text{ bits}$$

very close to the maximum entropy $H_{\text{th}} = \log_2 140 \simeq 7.12$ achievable with 140 uniformly distributed decimal values. We also point out that, if such 140 integers are transformed into their binary description, the biased distribution introduces a correlation of the bits. Numbers in decimal basis are just biased and not correlated; by transforming the integers into a binary basis, the resulting bits are correlated due to the non uniform original distribution. Moreover, in order to describe 140 different values, 8 bits are required,
implying that there are no events from 140 to 255, which worsens the situation. Clearly, Peres algorithm cannot be applied to this string.

In order to remove the bias and the correlation on the modulo 140 distribution, it is possible to use another post-processing method, the algorithm proposed by Zhou and Bruck [2]. The latter is the generalization of the Peres algorithm for biased distributions over a finite number of integers. Now we briefly describe the Zhou-Bruck algorithm (for a full description of the method refer to [2]). Let's consider a random variable $X$ with $n$ possible outcomes with a biased probability distribution. Let's define $b = \lceil \log_2 n \rceil$ as the number of bits required for a binary description of the outcomes (in our case with $n = 140$ we have $b = 8$). If the outcomes are labelled as $0, 1, \cdots, n-1$ and converted to binary, a string of $b$ bits corresponds to each outcome. For a given sequence $(x_1, \cdots, x_M)$, we can convert any $x_k$ to the corresponding binary string and consider only the first bit of each string: the resulting sequence is biased but with no correlation. Let's now consider the second bit of each string. We may create two sequences corresponding to the two possible values of the first bit: the first sequence collects the second bits related to a 0 first bit and vice versa for the second one. Again, these two sequences have bias but no correlation. The idea can be iterated: the $i^{th}$ bits can be grouped into $2^k - 1$ sequences according to the values of the $i-1$ bits. After this manipulation, one gets $N$ different sequences where $N = 2^k$. Since these $N$ sequences have bias but no correlation, the Peres algorithm can be applied separately to each of them (if the sequence is not empty). We implemented the Zhou-Bruck method on the tags modulo 140 of each pixel. As a matter of fact, the Zhou-Bruck method is quite efficient: for example, for the pixel 256, from each tag we get an average of 6.3 unbiased bits: this value is close to the maximum $H_{\exp} \simeq 6.8$ that can be extracted with a perfect efficient algorithm. The obtained bits were evaluated in term of bias (Fig. 8) and binary entropy extraction efficiency (Fig. 9) as well as serial correlation which re-enters within the limit of acceptance.

The final rate of the fine resolution is equal to $R_{\text{fine}} = 223$ Mbit/s for the whole pixels array. In order to evaluate the rate for a single pixel it must be taken in account that there are only 64 TDCs which switch among the four pixel banks. Therefore, the actual number of pixels (in terms of rate) truly is 64 and the final rate per pixel is $R_{\text{fine}}/64 = R_{\text{fine},p} \simeq 3.48$ Mbit/s per pixel. Thus, by considering the fine and coarse resolution, the average generation rate per pixel is given by $R_{\text{tot},p} = R_{\text{fine},p} + R_{\text{coarse},p} \simeq 4.84$ Mbit/s while the total generation rate for LinoSPAD device is given by $R_{\text{tot}} = R_{\text{fine}} + R_{\text{coarse}} \simeq 310$ Mbit/s.

VI. CONCLUSION

In this paper we showed improved techniques to produce true random numbers by processing single photon
Starting from a single detector system (Randy), we defined an efficient procedure to fully exploit the temporal degree of freedom. With respect to existing paradigms, it integrates a temporal tagging system with a detector matrix and thus it allows to use the temporal degree of freedom in addition to the much more common spatial one. Starting from a single detector system (Randy), we defined an efficient procedure to fully exploit the temporal degree of freedom. With respect to existing paradigms which are based on complex rules and are far from being information efficient, our procedure extracts the most of the system entropy, achieving the maximum bit rate allowed by the system. This procedure is based on the use of a high frequency sampling clock (with respect to the photon rate) and on the use of the Peres unbiasing algorithm. Therefore, the generation rate is only limited by the physical device performances and not by the technique itself. Applying this technique to LinoSPAD required a further step to deal with detectors matrix non-idealities (pixel cross correlation) and TDC ones. Hence, a dedicated post-processing procedure, which included the usage of the Zhou-Brük algorithm, was developed in order to work around such non-idealities. The summary comparison bar diagram of Fig. 10 clearly shows the remarkable differences among different generation procedures. Our Peres-based procedure clearly reaches a higher bit rate with respect to a protocol-based one. Furthermore, moving from the discrete SPADs framework (Randy) to the CMOS SPAD array one and increasing the sampling frequency as well as adding a time-to-digital converter (LinoSPAD) improves the generation rate even more. Final results show a bitrate per SPAD/pixel equal to:

\[ R_{\text{Randy,spad}}(100 \text{ MHz}) = 1.8 \text{ Mbit/s} \]
\[ R_{\text{LinoSPAD,spad}}(400 \text{ MHz + TDC}) = 4.84 \text{ Mbit/s} \]

and for LinoSPAD a total bitrate of

\[ R_{\text{LinoSPAD}} = R_{\text{LinoSPAD,spad}} \times 64 = 310 \text{ Mbit/s}. \]

Moreover, applying these techniques to other physical devices with better performances will further increase the generation rate. Future steps will also consider a real-time implementation of both procedures since the pre-processing, the Peres and Zhou-Brük algorithms could be effectively implemented via FPGA.

![QRNGs bitrate over different protocols](image)

**FIG. 10.** Bar diagram with different protocols rates comparison. The diagram shows the qualitative bit rates differences between different protocols. It is divided in two areas which represent the use of discrete SPADs (as in Randy) and SPAD array (LinoSPAD).

[1] Y. Peres, Ann. Statist. 20, 590 (1992).
[2] H. Zhou and J. Bruck, arXiv:1209.0726 (2012).
[3] S. Pironio, A. Acín, S. Massar, A. B. de la Giroday, D. N. Matsukevich, P. Maunz, S. Olmschenk, D. Hayes, L. Luo, T. A. Manning, and C. Monroe, Nature 464, 1021 EP (2010).
[4] B. G. Christensen, K. T. McCusker, J. B. Altepeter, B. Calkins, T. Gerrits, A. E. Lita, A. Miller, L. K. Shalm, Y. Zhang, S. W. Nam, N. Brunner, C. C. W. Lim, N. Gisin, and P. G. Kwiat, Phys. Rev. Lett. 111, 130406 (2013).
[5] P. Bierhorst, E. Knill, S. Glancy, Y. Zhang, A. Mink, S. Jordan, A. Rommal, Y.-K. Liu, B. Christensen, S. W. Nam, M. J. Stevens, and L. K. Shalm, Nature 556, 223 (2018).
[6] Y. Liu, X. Yuan, M.-H. Li, W. Zhang, Q. Zhao, J. Zhong, Y. Cao, Y.-H. Li, L.-K. Chen, H. Li, T. Peng, Y.-A. Chen, C.-Z. Peng, S.-C. Shi, Z. Wang, L. You, X. Ma, J. Fan, Q. Zhang, and J.-W. Pan, Phys. Rev. Lett. 120, 010503 (2018).
[7] S. Gómez, A. Mattar, E. S. Gómez, D. Cavalcanti, O. J. Farías, A. Acín, and G. Lima, Phys. Rev. A 97, 040102 (2018).
[8] X. Ma, X. Yuan, Z. Cao, B. Qi, and Z. Zhang, npj Quantum Inf. 2, 16021 (2016).
[9] J. Rarity, P. Owens, and P. Tapster, J. Mod. Opt. 41, 2435 (1994).
[10] T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger, Rev. Sci. Instrum. 71, 1675 (2000).
[11] M. Stipčević and B. M. Rogina, Rev. Sci. Instrum. 78, 045104 (2007).
[12] With the exception of bit \( x_1 \) and \( x_n \), two photo-detection events are necessary to generate an i.i.d. bit.
[13] H. Fürst, H. Weier, S. Nauerth, D. G. Marango, C. Kurtsiefer, and H. Weinfurter, Opt. Express 18, 13029 (2010).
[14] J. von Neumann, in Monte Carlo Method, National Bureau of Standards Applied Mathematics Series, Vol. 12, edited by A. S. Householder, G. E. Forsythe, and H. H. Germond (US Government Printing Office, Washington, DC, 1951) Chap. 13, pp. 36–38.
[15] We used the ZedBoard produced by Avnet.
[16] We used the SPCM-ARQH by Excelitas.
[17] F. Vedovato, C. Agnesi, M. Schiavon, D. Dequal, L. Calderaro, M. Tomasin, D. G. Marango, A. Stano, V. Luceri, G. Blanco, G. Vallone, and P. Villoresi, Sci.
The time required by the FPGA in order to implement the protocols was totally negligible.

The time interval was set in order to have $\langle n_r \rangle \simeq 10$ according to what stated in section II.

D. G. Marangon, G. Vallone, and P. Villoresi, Phys. Rev. Lett. 118, 060503 (2017).

M. Avesani, D. G. Marangon, G. Vallone, and P. Villoresi, Nat. Commun. 9, 5365 (2018).

C. Niclass, M. Sergio, and E. Charbon, in Proceedings of the Design Automation Test in Europe Conference, Vol. 1 (2006) pp. 1–6.

E. Charbon, Philos. Trans. R. Soc. A 372, 20130100 (2014).

D. Stucki, S. Burri, E. Charbon, C. Chunnilall, A. Meneghetti, and F. Regazzoni, in Proc. SPIE, Vol. 8899 (2013).

S. Burri, D. Stucky, Y. Maruyama, C. Bruschini, E. Charbon, and F. Regazzoni, in International Image Sensor Workshop, EPFL-CONF-191217 (2013).

D. G. Marangon, G. Vallone, U. Zanforlin, and P. Villoresi, Quantum Sci. Technol. 1, 015005 (2016).

S. Burri, H. Homulle, C. Bruschini, and E. Charbon, in Optical Sensing and Detection IV, Vol. 9899, International Society for Optics and Photonics (SPIE, 2016) pp. 57–66.

C. Favi and E. Charbon, in Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, FPGA '09 (ACM, New York, NY, USA, 2009) pp. 113–120.

M. Fishburn, L. H. Menninga, C. Favi, and E. Charbon, IEEE Trans. Nuc. Sci. 60, 2203 (2013).

J. Song, Q. An, and S. Liu, IEEE Trans. Nuc. Sci. 53, 236 (2006).