AIP | Review of Scientific Instruments

# Digital configurable instrument for emulation of signals from radiation detectors

A. Abba, F. Caponio, and A. Geraci

Citation: Review of Scientific Instruments **85**, 013506 (2014); doi: 10.1063/1.4861920
View online: http://dx.doi.org/10.1063/1.4861920
View Table of Contents: http://scitation.aip.org/content/aip/journal/rsi/85/1?ver=pdfcov
Published by the AIP Publishing

# Digital configurable instrument for emulation of signals from radiation detectors

A. Abba, F. Caponio, and A. Geraci
*Politecnico di Milano, Department of Electronics, Information and Bioengineering-DEIB, Milan 20133, Italy*

(Received 9 October 2013; accepted 29 December 2013; published online 21 January 2014)

The paper presents a digital instrument characterized by a specially designed architecture that is able to emulate in real time signals from a generic radiation detection system. The instrument is not a pulse generator of recorded shapes but a synthesizer of random pulses compliant to programmable statistics for height and starting time of events. Completely programmable procedures for emulation of noise, disturbances, and reference level variation are implemented. © *2014 AIP Publishing LLC*. [http://dx.doi.org/10.1063/1.4861920]

## I. INTRODUCTION

The development of setups at the state-of-the-art for radiation measurement[1–3] requires the development of sophisticated techniques for emulating the entire experiment in which the instrumentation will operate. The advantages of such an approach are clear and important. The measurement setup can be tested in near to the operative conditions while other parts of the experiment are still in construction; the development phase can be performed in laboratories not subject to stringent radiation control procedures; it is possible to test the equipment with signals whose characteristics are completely known, repeatable, and can be changed for better understanding the behavior of the instrumentation. The quality of the experiment is also positively affected. In fact, the availability of a configurable signal source simplifies the testing of the processors, allows absolute and fair comparisons among different processing techniques, and permits to directly evaluate different algorithms and to adjust the processing flow. Lastly, if the signal stream to be analyzed is obtained from a limited set of data, it is easy to store and share it among geographically sparse laboratories. To better understand this point, consider that a 2-bytes data stream at the rate of $100 \times 10^6$ of samples per second (Msamples/s), which could be a typical ADC output of a preamplifier signal, consists of roughly 12 GByte of data per minute. Instead, with the proposed solution one can use the first-order statistics of relevant parameters and a single seed for a pseudorandom number generator to synthesize an arbitrarily long data stream with a very reduced amount of transmitted information.

Unfortunately, this approach has the drawback of the huge amount of calculations that has to be performed in real time. The simple solution of using a personal computer interfaced with high speed DACs is therefore ruled out, and a special hardware is necessary. We found convenient a software-hardware co-design approach employing a FPGA device as main processor. For this reason, from now on, we will use the term "architecture" to indicate the hardware and/or the software of the instrument.

We have inserted throughout the text some quantitative considerations regarding the implementation and examples. All of them are based on the assumption of using a FPGA device Xilinx Virtex-5 FX 100T and a digital-to-analog converter Analog Devices AD9726. The synthesis process of the signal is the cascade of several steps represented in Fig. 1.

On the basis of a reference shape and a statistical distribution of occurrence times and amplitudes of events, the system generates two independent pulses that can be selectively added together to emulate the pile-up phenomenon. The resulting signal is completed by adding noise and a deviation of the reference level ("baseline") to it, and can also be shaped to take into account the electronic conditioning stage. The generated signal can be made available at the output of the instrument in digital or analog form. The first choice gives also the possibility of introducing nonlinearity effects and quantization noise, whereas the second one requires a high-performance digital-to-analog conversion process.

In particular, of primary importance are the generation of pulse amplitudes compliant to an energy spectrum user-defined and the calculation of pulse occurrence times distributed on the basis of a programmable statistic. For this purpose, an algorithm was developed[4] that retrieves the properties of a statistic variable from its histogram. By initializing the algorithm with a real spectrum shape, a statistical amplitude distribution of events is generated. On the other hand, from the distribution probability in time of the events, we generate the statistically corresponding occurrence times of the pulses.[4]

In addition, the instrument implements the emulation of the baseline drift, noise, disturbances, output signal shaping, and analog or digital output interface.

The requested hardware has been built and is under testing in several different scientific environments.

## II. PULSE GENERATION

An algorithm for getting the first order statistical characteristics of a random process from the histogram of a set of realizations has been devised and implemented.[4] This method returns the statistics of distribution of amplitudes and occurrence times of the events. The statistical generator of occurrence times acts as the trigger for the output of the signal shape, whose amplitude is scaled according to the sorted amplitude value. The architecture of the pulse generator is shown in Fig. 2.

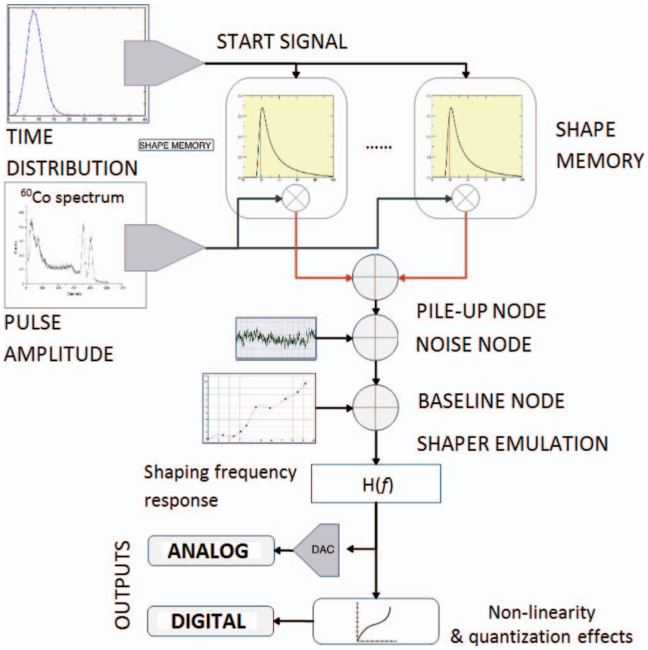

FIG. 1. Block diagram of the emulation process. The inputs by the user are: (i) the reference signal shape; (ii) the amplitude statistic; (iii) the occurrence time statistic; (iv) the noise spectral description; (v) the baseline profile; (vi) the optional shaper transfer function; and (vii) the description of non-linearity and quantization effects.

The user loads in the memory of the system, as sequence of samples, the shape of the signal to be emulated. This stored shape is normalized to unit maximum value and is called reference pulse. By extracting a value of starting time and a value of amplitude on the basis of the statistical characteristics that are initialized by the user, the reference shape is recovered from memory and scaled. If requested, the significance of the pulse amplitude with respect to a fixed threshold can be checked. In particular, a linear interpolation process that significantly reduces the required memory is implemented.[5] For instance, at the rate of 350 Msamples/s, a signal lasting 10 ms needs 3.5 × $10^6$ of words (Mwords) to be stored. This is too much for the on-chip memory resources of the FPGA that we used (about 8 Mbits). Each shape generator has a dedicated memory of 16 thousand words, that is, enough to store a 52 $\mu$s long shape. Instead, using a 1:1000 interpolation, it is possible to generate a 52 ms signal. Since the complexity of the interpolator should be as low as possible, a linear interpolator has been implemented.

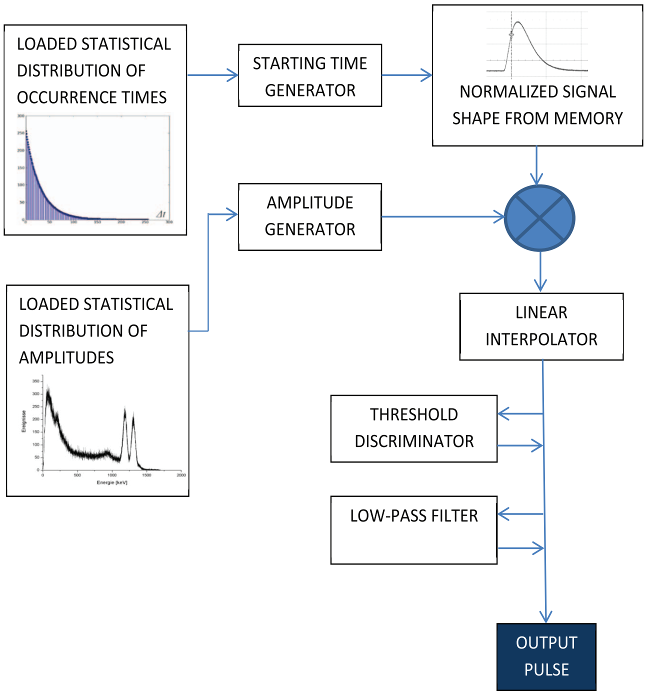
LOADED STATISTICAL DISTRIBUTION OF OCCURRENCE TIMES
Δt
STARTING TIME GENERATOR
NORMALIZED SIGNAL SHAPE FROM MEMORY
AMPLITUDE GENERATOR
LOADED STATISTICAL DISTRIBUTION OF AMPLITUDES
LINEAR INTERPOLATOR
THRESHOLD DISCRIMINATOR
LOW-PASS FILTER
OUTPUT PULSE

FIG. 2. Structural diagram of the pulse generator module.

Each generator is triggered independently by a generated starting time. The shapes at output of the two generators in parallel are added together (Fig. 1) in order to emulate the pile-up. There is also the possibility of inhibiting pile-up of pulses that occur too close. This is important in the development and test of algorithms for pile-up rejection.

Optionally, the pulse can be low-pass filtered to emulate, for instance, the preamplifier integration effect.

## III. BASELINE DRIFT EMULATION

Even in absence of events, the baseline of the electrical signal is not stable for several issues, such as the thermal effects in detectors and electronics or the 1/f current noise. The drift of the baseline value is not statistical but is introduced by the user as a deterministic shape. The reason of this choice is to allow the emulation of deterministic variations, like periodic interferences or couplings from nearby electronics. Figure 3 shows the block diagram of the baseline emulator.

A small memory that contains the key-points of the baseline shape is read and the output is interpolated by a cubic spline function.[5] The cubic splines are often used when high degrees of smoothness are required in the interpolation process. The reason why a spline interpolator is used here, while a linear interpolator is used for the pulses, is the best usage of the FPGA resources. In fact, the pulse signals are short and fast, so a modest amount of memory is necessary to store them but data have to be output at very fast rate. This requirement rules out the use of a cubic spline, since there is no time for the calculations involved with it. The opposite happens for the baseline. The approach of a deterministic baseline implies the necessity of defining it for long time ranges. An aggressive approach to the reduction of stored data is therefore necessary. On the other hand, the interest of emulating the baseline drift is confined to the low frequencies: the interpolator has therefore a longer time for the calculations. The implementation of spline functions for the interpolation is computationally intensive. First, an inverse operation calculates the coefficients of the basis functions. Second, a forward calculation generates the interpolating spline.[6] On the contrary, the linear interpolation has the advantages of not requiring the solution of the inverse problem (the data points are themselves the coefficients of the triangular basis functions) and therefore of an extremely efficient generation of the output samples. The linear interpolating function has only $C^0$ continuity (the function is continuous but its derivatives are not) and therefore lacks the smoothness necessary for the baseline drift in the low

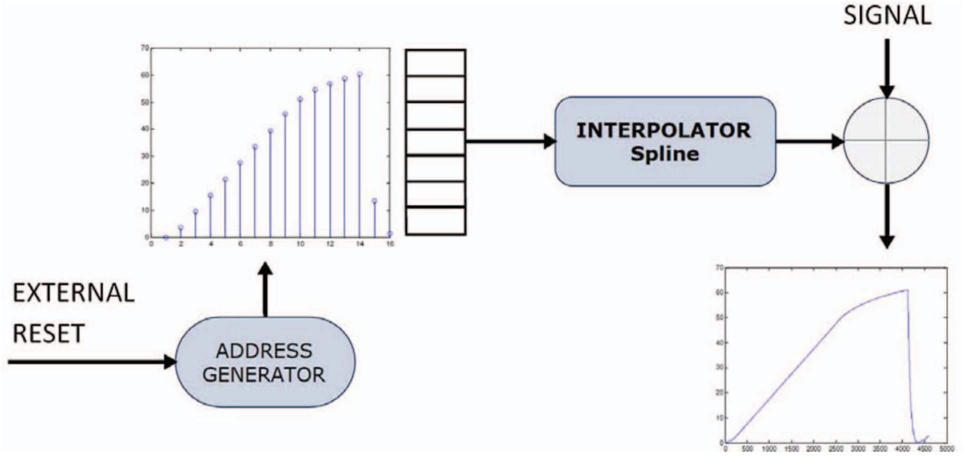

FIG. 3. Block diagram of the baseline emulator. The address generator scans the key-points memory and a spline interpolator generates the up-sampled shape. In the plot, the "signal" is equal to zero in order to highlight the interpolated shape. The subplots have only qualitative value and therefore the numerical quotations of the axes have no meaning.

frequency range. Instead, the cubic spline interpolation provides $C^2$ continuity (i.e., the function and the first two derivatives are continuous) and consequently the baseline spectrum can be maintained at low frequencies.

The critical point of the interpolation process is the matrix inversion. In order to overcome this computational burden, we have implemented a technique based on a particular class of spline basis functions that solves the problem using only simple arithmetic shifts and fixed-point additions both for the forward operation and for the inverse one.[6]

The proposed system operates with interpolation factors selectable between 2 and 219 samples with 4096 key points. At the rate of 300 MHz, the system is able to emulate a drift profile up to 7 s long. A pipelined implementation of the interpolation process has been performed. In order to minimize the pipeline depth and to maximize the speed, all computational blocks operate in fixed-point arithmetic. The precision of 64-bit on all computational blocks is necessary to ensure the stability of the system. The output is an interpolated signal at the rate of 160 MHz.

## IV. NOISE EMULATION

Three main physical sources of stationary noise have been taken into account, i.e., thermal noise, low-frequency voltage noise, and shot noise. Random disturbances, which are mainly related to the experimental environment, can be also emulated.

In principle, the first operation is the analysis of the spectrum of the noise to be emulated. Then the emulator is fed with the noise spectral density and generates a statistic process having the required characteristics. The conceptually trivial way of shaping a white noise spectrum, according to the measured noise characteristics and then calculating the inverse Fourier transform, is not feasible at the requested rate (i.e., up to 300 Msamples/s).

Instead, we adopted the strategy of separating the noise spectrum in three regions: low, medium, and high frequency. The architecture of the noise generator is shown in Fig. 4 and operates like a multi-channel graphic equalizer. Three independent linear-feedback-shift registers (LFSR)[7] generate

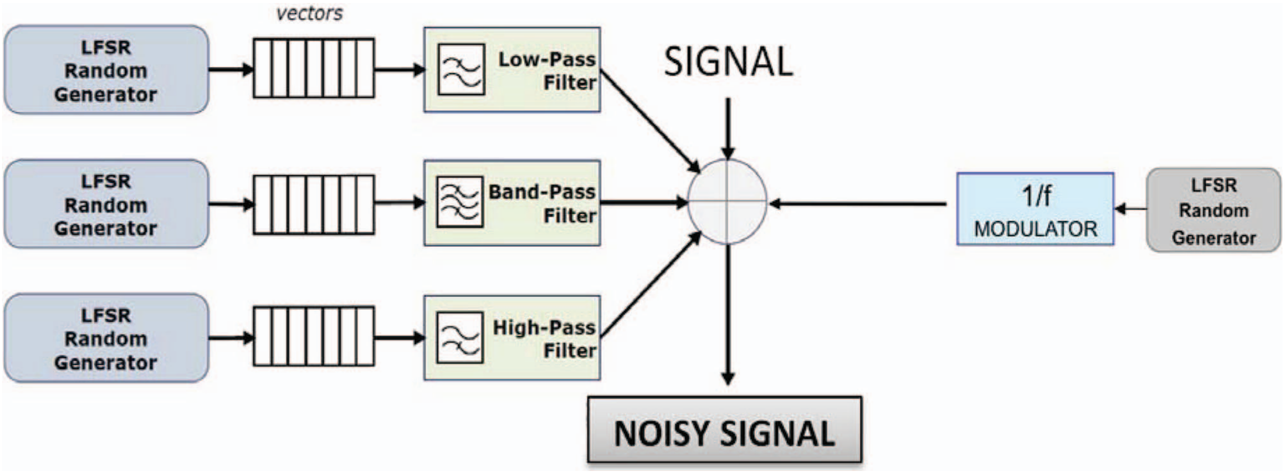

FIG. 4. The architecture of the emulator of the noise is constituted by three parallel stages. In each stage a generator of random numbers feeds a FIFO, whose output is treated by a configurable digital filter. The outputs of the digital filters sum in an adder block, which generates the final noisy signal. A separate path is devoted to the emulation of the 1/f noise.

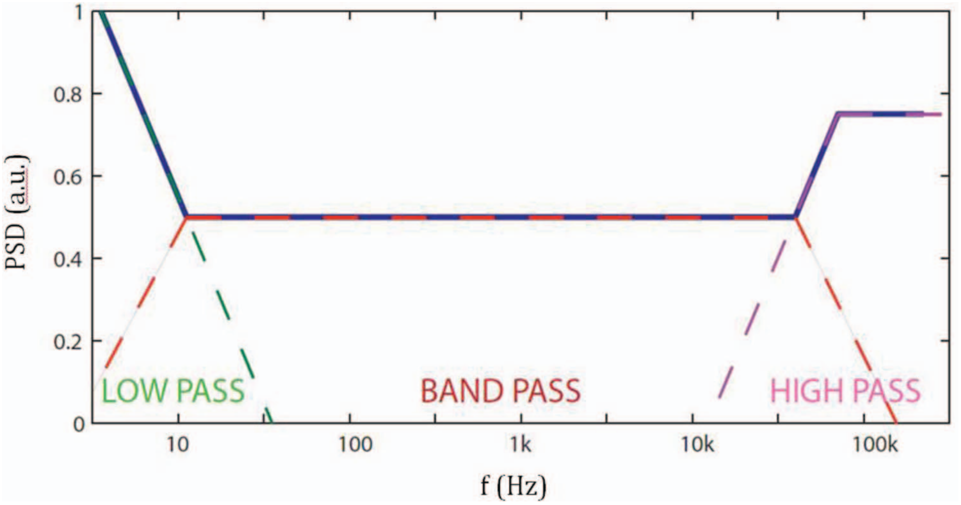

FIG. 5. Example of a spectrum of the emulated noise.

sequences of random numbers, which are filtered by first-order infinite impulse response (IIR) filters (i.e., low-, band-, high-pass filter). Figure 5 shows an example of emulated noise spectrum.

The digital filters of the noise generation module are IIR structures. The advantage of the IIR filters with respect to the finite-impulse-response (FIR) filters is that IIR filters usually require fewer coefficients for executing the same operations, are faster, and need less memory. The flow graph of the Direct Form II of the first-order IIR filter is shown in Fig. 6.[8] This structure is canonic, i.e., needs the minimum number of memory elements to be implemented, and allows the design of both low-pass and high-pass filters. The band-pass filter requires the cascade of a couple of these cells.

The transfer function[8] of the implemented IIR filter in the z-domain[9] is

$$H(z) = H_0 \cdot \frac{b_0 + b_1 z^{-1}}{a_0 + a_1 z^{-1}}, \qquad (1)$$

where coefficients $a_i$, $b_i$ define the transfer function of the filter.

We have adopted a first order Butterworth filter[9] as elementary cell, whose equivalent digital low-pass (LP) and high-pass (HP) filters are[9]

$$H(z)|_{LP} = \frac{1 + z^{-1}}{(1 + c_{LP}) + (1 - c_{LP})z^{-1}}, \qquad (2)$$

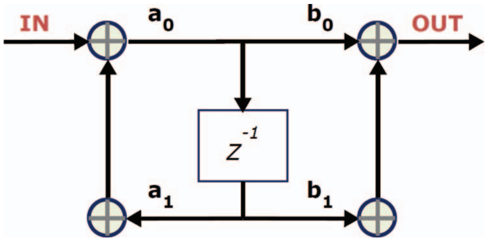

FIG. 6. First-order Direct Form II implementation of an IIR filter.

$$H(z)|_{HP} = \frac{1 - z^{-1}}{(1 + c_{HP}) + (1 - c_{HP})z^{-1}}, \qquad (3)$$

where

$$c_{LP/HP} = \text{cotan}(0.5 \cdot F_{c,LP/HP}/F_s) \qquad (4)$$

and $F_{c,LP/HP}$ is the cut-off frequency of the filter, *LP* or *HP*, and $F_s$ is the clock frequency.

The transfer function of the band-pass stage is

$$H(z)|_{BP} = \frac{1 + z^{-1}}{[1 + c_{LP}] + [1 + c_{LP}]\,z^{-1}} \cdot \frac{1 - z^{-1}}{[1 + c_{HP}] + [1 + c_{HP}]\,z^{-1}}. \qquad (5)$$

## A. Emulation of 1/f noise

The implementation of the 1/*f* noise emulator should not significantly increase the computational burden to avoid impairing or limiting the performance of the system.

The power spectral density $S_{1/f}(f)$ of the 1/*f* noise can be obtained by properly shaping the power spectral density $S_{white}(f)$ of a white noise generated by a LSFR (see Fig. 4) with a transfer function *T*(*f*), i.e.,

$$S_{1/f}(f) = S_{white}(f) \cdot |T(f)|^2. \qquad (6)$$

The transfer function *T*(*f*) is squared because *S*(*f*) represents the intensity of a power spectral density. One simple way to obtain the transfer function *T*(*f*) of the shaping filter consists in considering the 1/*f* trend as the superposition of *n* first-order low-pass filters (LPFs) with poles positioned at equal distance on a logarithmic scale in the range of frequencies between $f_{min}$ and $f_{max}$, where the 1/*f* shaping is desired,

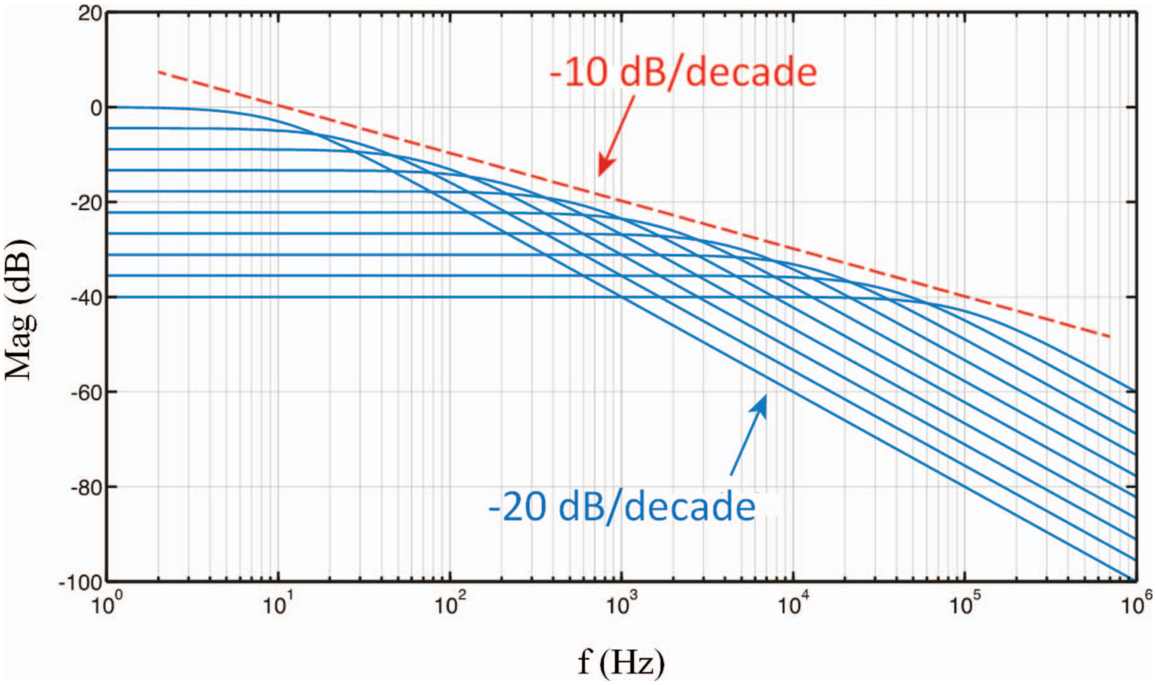

FIG. 7. The profile of the 1/*f* spectrum is the envelope of the frequency responses of first-order low-pass filters.

i.e.,

$$\begin{aligned}\log(f_{pole,i+1}) - \log(f_{pole,i}) &= \frac{\log(f_{\max}) - \log(f_{\min})}{n-1} \\ \frac{f_{pole,i+1}}{f_{pole,i}} &= \left(\frac{f_{\max}}{f_{\min}}\right)^{\frac{1}{n-1}}.\end{aligned} \tag{7}$$

The gain at low frequency of the *i*th filter has to be

$$G_{0,i} = \sqrt{\frac{f_{pole,1}}{f_{pole,i}}}. \tag{8}$$

Consequently, the global transfer function of the filter that shapes the LSFR output is

$$T(s) = \sum_{i=1}^{n} \frac{G_{0,i}}{1 + s \cdot \tau_{pole}}, \tag{9}$$

where $s = j \cdot 2\pi \cdot f$ ($j$ imaginary unit) and $\tau_{pole}$ is the time constant of the pole positioned at the frequency $f_{pole}$ $= 1/(2\pi \cdot \tau_{pole})$. The generation process is represented in Fig. 7.

In order to state the number *n* of low-pass filters that are necessary for an acceptable approximation, an inductive investigation has been performed, whose evidence is shown in Fig. 8.

The plot shows the global transfer functions *T*(*f*) for different numbers *n* of superimposed low-pass filters. Above $n = 8$, a very good fitting is achieved with no significant improvement as *n* increases. That is why, according to the strategy of minimum system overhead, the superposition of $n = 10$ filters was performed in the system.

The low-pass filters should closely approximate the steepness −20 dB/decade; therefore, IIR filters were used.[9] It should be noticed that sampling frequencies in the order of tens of MHz and a range of frequencies of 1/*f* noise from few Hz to about 100 kHz correspond to singularities far from the Nyquist frequency limit.

As validation of the approach, consider a white noise that feeds the 10 IIR low-pass filters whose singularities and gains are defined by Eqs. (7) and (8), and whose output signals are summed. The Fast Fourier Transform (FFT) of the output signal is shown in Fig. 9, which highlights that a very good approximation has been achieved. The deviation between the 1/*f* shape obtained and the ideal 1/*f* trend is below 0.1% up well above the upper limit of 100 kHz of the frequency range of interest.

## V. OUTPUT SIGNAL SHAPING

In general, the signal at the output of the detector is conditioned by a preamplifier and, in many cases, also by shaping electronics.

While the presence of a preamplifier can be simply emulated by taking into account its time constant $\tau_p$ in the reference shape, the emulation of the shaper requires a particular digital filter that implements its frequency response. Since this signal shaping should affect also baseline and noise, this stage is placed at the end of the emulation chain (Fig. 1).

The semi-Gaussian transfer function is at the base of many structures of shapers[10] and is the cascade of one differentiator and *n* integrators, i.e.,

$$H(s)|_{semi-G} = \left(\frac{s\tau_d}{1 + s\tau_d}\right) \cdot \left(\frac{A}{1 + s\tau_i}\right)^n, \tag{10}$$

where $\tau_d$ and $\tau_i$ are the time constants of differentiation and integration, respectively, and *A* is the integration dc gain. The number *n* of the integrators determines the shaper order. The kernel for the implementation of the shaper transfer function is the Direct Form II IIR filter (Fig. 6). For instance, a second

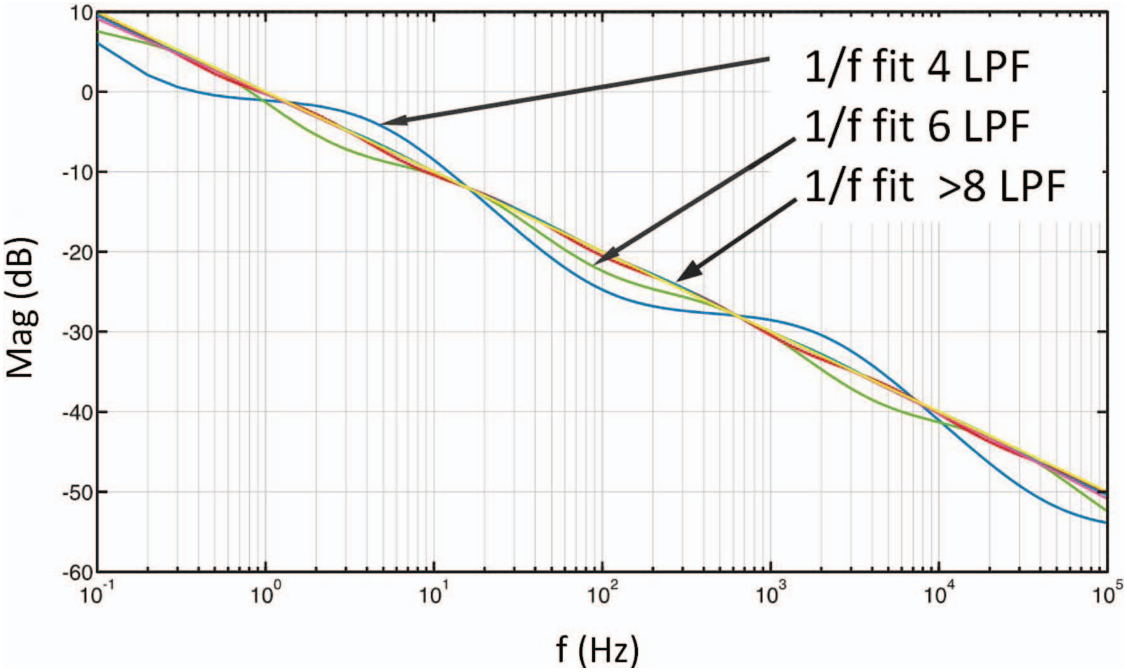

FIG. 8. Comparison of frequency responses of the superposition of $n$ low-pass filters (squared modules) for different values of $n$. No lines can practically be distinguished with respect to the $1/f$ ideal trend ($-10$ dB/decade) for $n > 8$. In particular, for $n = 8$, the maximum deviation from the ideal trend is below 0.1%.

order semi-Gaussian transfer function with 3-poles and 2-zeros, which compensate the preamplifier time constant $\tau_p$, i.e.,

$$H(s)|_{semi-G} = A\left(\frac{s\tau_d}{1+s\tau_d}\right)\cdot\left(\frac{1+s\tau_p}{(1+s\tau_{i1})(1+s\tau_{i2})}\right) \tag{11}$$

is implemented by the cascade of one Direct-Form II second-order IIR stage and two Direct-Form-II first-order IIR filters, as shown in Fig. 10. The transfer function of the corresponding digital filter in the z-domain is

$$H(z) = H_0\cdot\frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}}. \tag{12}$$

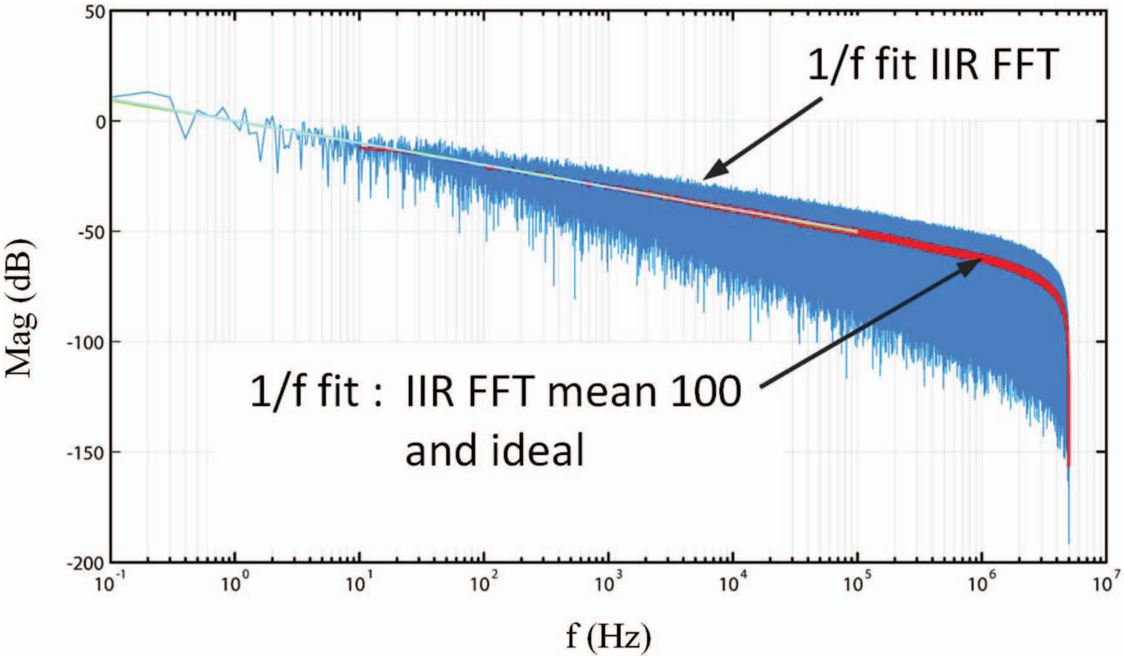

FIG. 9. Comparison between ideal $1/f$ noise spectrum (compact line) and spectrum generated by the 10 IIR digital filters (spread line). The compact line is the superposition of the ideal $1/f$ trend and the digitally synthesized spectrum averaged on 100 samples in order to reduce artifacts introduced by the FFT. The two tracks are indistinguishable.

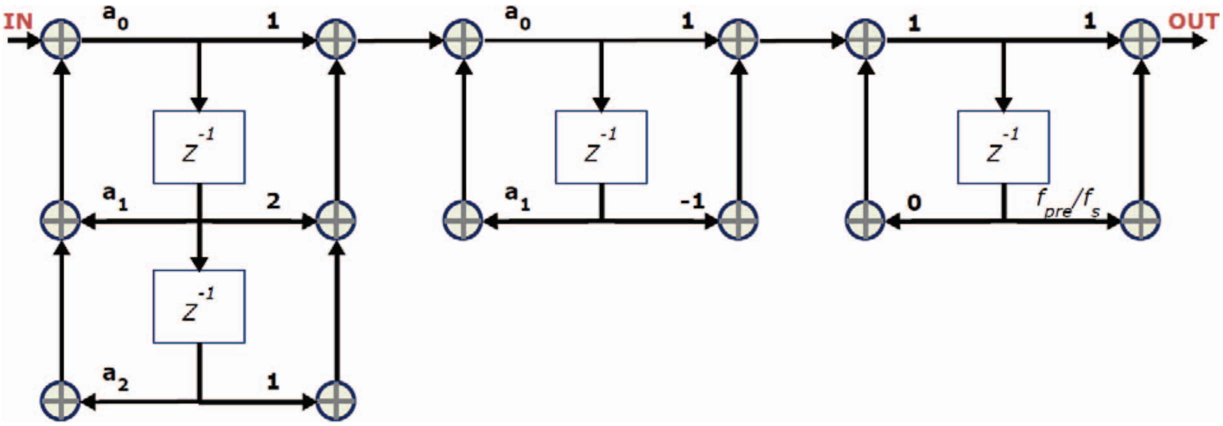

FIG. 10. Second order semi-Gaussian shaping architecture.

The filter can be synthesized using, for instance, second order Butterworth cells,[11] i.e.,

$$H(s)|_{LP} = \frac{1}{s^2 + \sqrt{2}s + 1}, \tag{13}$$

that corresponds to the digital filter

$$H(z, F)|_{LP} = \frac{1 + 2z^{-1} + z^{-2}}{\left(c_{LP}^2 + \sqrt{2}c_{LP} + 1\right) + \left(2 - 2c_{LP}^2\right)z^{-1} + \left(c_{LP}^2 + \sqrt{2}\right)z^{-2}}. \tag{14}$$

The high-pass filter has been implemented as a first order Butterworth cell. Referring to Eq. (1), the corresponding coefficients are $b_0 = 1$, $b_1 = 1/(2\pi\ \tau_p F_s)$, $a_0 = 1$, $a_1 = 0$.

From the implementation side, the best performance is achieved through a pipelined processing architecture. While a FIR structure can be easily pipelined, the IIR filter has a critical path depending on the value of the pole. Since in one clock cycle, one multiplication of $y(n\text{-}1)$ by a factor $a_0$ and one accumulation have to be performed, the maximum pipeline depth is equal to 1. Such a short pipeline limits the maximum speed of the system due to the propagation delay in the logic chain of multiplier and adder. For a non-recursive digital filter (FIR), parallel processing can be easily realized by replicating the filter many times depending on the throughput requirements. In case of recursive filters (IIR), the simple hardware replication cannot be used because the filter response of one unit depends on the response output of the preceding ones. This means that particular techniques need to be considered.

Very effective approaches for pipelining of recursive filters are the clustered and the scattered look-ahead techniques.[12] The basic idea of clustered look-ahead pipeline is to add poles and zeros of compensation to the filter transfer function such that the coefficients of $z^{-1}\ldots\ z^{-(M\text{-}1)}$ in the denominator of the transfer function are equal to zero. As consequence, the output sample $y(n)$ can be expressed in terms of the cluster of $N$ past outputs $y(n\text{-}M)$, $y(n\text{-}M\text{-}1)$, …, $y(n\text{-}M\text{-}N{+}1)$. The critical loop of this implementation, which contains $M$ delay elements and a single multiplication, can be pipelined by $M$ stages and the sample rate is increased by the factor $M$. Since no control is forced over the added poles, the stability of the pipelined filter is not guaranteed. Instead in the approach scattered look-ahead, the denominator of the transfer function is transformed to contain the $N$ terms $z^{-M}$, $z^{-2M}$, …, $z^{-NM}$, and the state $y(n)$ is computed in terms on $N$ past scattered states $y(n\text{-}M)$, $y(n\text{-}2M)$, …, $y(n\text{-}NM)$. In this case, for each pole in the original filter, ($M$-1) poles and zeros of compensation are introduced with the same angular spacing at a distance from the origin of the z-plane equal to that of the original pole. In this way, since the distance of the additional poles from the origin is the same of the original filter, the pipelined filter is stable if the original filter is stable. The block diagram of the second order cell

$$H(z) = \frac{1 - b_1 z^{-1}}{1 - a_1 z^{-1} - a_2 z^{-2}} \tag{15}$$

that has been implemented as scattered look-ahead pipelined filter

$$H(z) = \frac{\left(1 - b_1 z^{-1}\right) \cdot \left[1 + a_1 z^{-1} + \left(a_1^2 + a_2\right) z^{-2} - a_1 a_2 z^{-3} + a_2^2 z^{-4}\right]}{1 - \left(a_1^3 + 3a_1 a_2\right) z^{-3} - a_2^3 z^{-6}} \tag{16}$$

is shown in Fig. 11.

While the FIR section of the filter operates with 24-bit coefficients and a 32-bit adder, the IIR stage requires both 48-bit multiplier and adder. In fact, the feedback loop causes the accumulation of the error and the filter becomes instable if the poles move in proximity to the unit circle. With 24-bit coefficients and accumulation register, the filter is able to emulate time constants between 200 ms and 20 ns with a throughput of

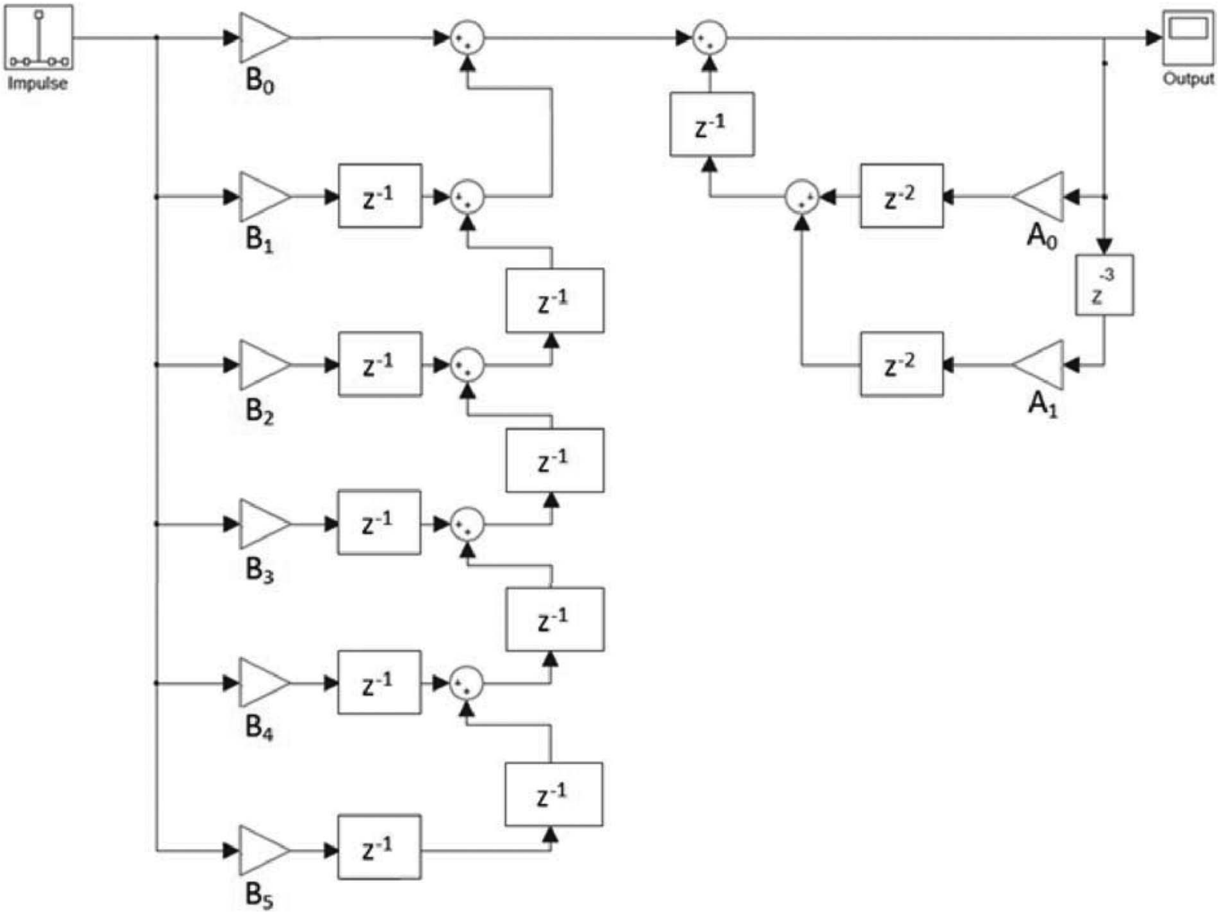

FIG. 11. Block diagram of the implemented IIR filter that is pipelined by means of the scattered look-ahead technique. Parameters $A_i$, $B_i$ are combinations of the coefficients $a_i$, $b_i$ according to Eq. (16) structure. The coefficients of the backward branches (denominator of Eq. (16)) are: $A_0 = a_1^3 + 3a_1a_2$, $A_1 = a_2^3$. The coefficients of the forward branches (numerator of Eq. (16)) are: $B_0 = 1$, $B_1 = a_1 - b_1$, $B_2 = a_1^2 - a_1b_1 + a_2$, $B_3 = -a_1\, a_2 - a_1^2b_1 - b_1a_2$, $B_4 = a_2^2 + a_1a_2\, b_1$, and $B_5 = -\, a_2^2b_1$.

312 Msample/s. The complete transfer function of the shaper is obtained from the cascade of two second-order transfer functions and a first-order one. The user is therefore able to set 5 poles and 3 zeros with time constants within the above specified range.

## VI. NONLINEARITY EMULATION

As Fig. 1 shows, the output of the emulator can be an analog or a digital signal. In this latter case, the nonlinearity of the analog-to-digital converter (ADC) and of ancillary electronics can be introduced. This allows, for instance, evaluating the impact of nonlinearity of different electronic front-end configurations and ADCs from simulations or from the characteristics on the data-sheets.

The dynamic characteristics of the electronic front-end circuit can be emulated by properly configuring the shaper transfer function.

The emulation of the DC transfer function is obtained using a very large (216 words) look-up table (LUT) that is addressed with the ideal code generated by the emulator. The LUT generates a value that is modified according to the stated nonlinear behavior.

## VII. CONCLUSIONS

A fully programmable system for the generation of signals that emulate the output of detection setups was presented.

The instrument is a synthesizer of true random pulses compliant to user-defined statistics. In this respect, an algorithm for getting statistic properties from a histogram has been implemented, by which amplitude modulation and temporal distribution of pulses are obtained.

The system is able to emulate the pile-up effect and the drift of the baseline value in time following a programmable generic profile. Sources of stationary white and 1/f noise, and generic disturbances can be emulated. Arbitrary conditioning of the signal at the output of the detector by a preamplifier and also by a shaper can be taken into account.

In alternative to the proposed instrument, the same tasks could be accomplished by the combined use of a traditional waveform generator and a computer. But, in this case, the shapes are generated on the computer and transferred to the generator, performing off-line all what the proposed instrument realizes in real-time, without overload of the host computer and with very higher performance.

[1] R. Peloso, P. Busca, C. Fiorini, A. Abba, A. Geraci, A. Manenti, A. Longoni, G. Padovini, C. Bianchi, G. L. Poli, K. Erlandsson, B. F. Hutton, P. Lechner, H. Soltau, L. Strüer, A. Pedretti, P. Van Mullekom, and L. Pallaro, in *Proceedings of the 2010 IEEE Nuclear Science Symposium*, Knoxville, TN, 30 October–6 November 2010 (IEEE, 2010), p. 1957.

[2] A. Abba, A. Manenti, A. Suardi, C. Fiorini, and A. Geraci, in *Proceedings of the 2009 IEEE Nuclear Science Symposium*, Orlando, FL, 25–31 October 2009 (IEEE, 2009), p. 559.

[3] A. Abba, A. Manenti, A. Suardi, C. Fiorini, and A. Geraci, in *Proceedings of the 2008 IEEE Nuclear Science Symposium*, Dresden, Germany, 19–25 October 2008 (IEEE, 2008), p. 1661.

[4] A. Abba, F. Caponio, F. Guerrieri, A. Geraci, and G. Ripamonti, in *Proceedings of the 2010 IEEE Nuclear Science Symposium*, Knoxville, TN, 30 October–6 November 2010 (IEEE, 2010), p. 1251.

[5] R. Feng, L. Jing, and X. Meihua, in *Proceedings of the International Conference on Electronic Packaging Technology and High Density Packaging ICEPT-HDP*, Shanghai, 28–31 July 2008 (IEEE, Piscataway, NJ, 2008).

[6] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (Cambridge University Press, Cambridge, 1992), Chap. 3.

[7] H. Beker and F. Piper, *Cipher Systems: The Protection of Communications* (Wiley-Interscience, New York, 1982), p. 212.

[8] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach* (McGraw-Hill, New York, 1998).

[9] A. Antoniou, *Digital Filters: Analysis, Design, and Applications* (McGraw-Hill, New York, 1993).

[10] C. Fiorini and M. Porro, IEEE Trans. Nucl. Sci. **51**(5), 1953 (2004).

[11] J. Millman and C. C. Halkias, *Integrated Electronics: Analog and Digital Circuits and Systems* (McGraw-Hill, New York, 1972).

[12] J. Living, M. Moniri, and S. B. Tennakoon, J. VLSI Signal Process **27**(3), 269 (2001).