

Real-Time FPGA-Based SiPM Detector Emulation Using a Temporally Quantized Model

Stefano Carsi, Edoardo Proserpio, Andrea Abba, Francesco Caponio, and Valentina Arosio

Abstract—We present a real-time hardware implementation of a versatile detector emulator capable of reproducing realistic silicon photomultiplier signals. Our approach builds upon the open-source SimSiPM framework, originally developed to simulate the microscopic response of silicon photomultipliers, including photon detection efficiency, optical crosstalk, afterpulsing, and dark counts. SimSiPM provides idealized photon-level data with arbitrary temporal and amplitude resolution. In contrast, our emulator, built on a field-programmable gate array, translates this fine-grained simulation into physically realizable analog signals, maintaining real-time operation and finite hardware resolution. The system receives simulated photon events either via a 10-gigabit Ethernet stream or directly from the processing system of a system-on-chip, and performs on-chip temporal quantization, dividing time into bins equal to one clock cycle. All photon hits within a bin are accumulated, and their contribution is combined through a weighted temporal averaging scheme that preserves sub-bin precision. Signal shaping is executed entirely in hardware, using parallel one-pole recursive filters that synthesize the rise and the two decay components of the response. The resulting waveform is converted to analog through dual 16-bit digital-to-analog converters operating at 2.5 gigasamples per second. This architecture generates physically accurate detector signals in real time, rather than replaying precomputed waveforms. It also generalizes beyond silicon photomultipliers, providing a flexible framework for hardware-in-the-loop testing of front-end electronics. The proposed implementation demonstrates high throughput, low latency, and minimal processor overhead.

Index Terms—Detector emulator, field-programmable gate array (FPGA), hardware-in-the-loop, infinite impulse response (IIR) filter, real-time signal processing, silicon photomultiplier (SiPM).

I. INTRODUCTION

A DETECTOR emulator is an instrument that generates analog signals indistinguishable from those produced by a real particle detector [10]. It allows the characterization, debugging, and validation of front-end electronics—preamplifiers, shapers, analog-to-digital converters, and trigger logic—without a physical detector or a radiation source. Because the energy, timing, and multiplicity of every generated event are known exactly, the emulator also provides perfect ground truth for the validation of reconstruction algorithms, and it enables hardware-in-the-loop testing of firmware and

Manuscript submitted June 29, 2026. This work was carried out at Nuclear Instruments SRL.

S. Carsi and E. Proserpio are with Nuclear Instruments SRL, 22045 Lambrugo (CO), Italy, and also with the Department of Science and High Technology, Università degli Studi dell’Insubria, 22100 Como, Italy (e-mail: s.carsi@nuclearinstruments.eu; e.proserpio@nuclearinstruments.eu).

A. Abba, F. Caponio, and V. Arosio are with Nuclear Instruments SRL, 22045 Lambrugo (CO), Italy.

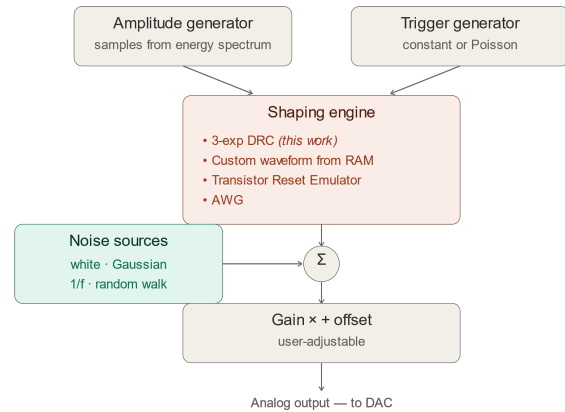


Fig. 1. Functional architecture of the detector emulator. An amplitude generator and a trigger generator drive a configurable shaping engine; noise, gain, and offset are applied before the digital-to-analog converter. The three-exponential shaping core described in this work is one of the available shaping models.

data-acquisition chains under fully reproducible conditions, including controlled rate, pile-up, and background.

The key difference between a detector emulator and a conventional arbitrary waveform generator (AWG) is that the emulator reproduces the *physics* of the detector rather than replaying a fixed, precomputed waveform. As sketched in Fig. 1, an amplitude generator (constant or sampled from an energy spectrum) and a trigger generator (periodic or Poisson distributed) feed a shaping engine that synthesizes the detector response; noise, gain, and offset are then applied before the signal is sent to the output. The shape produced by the shaping engine, and the parameters that control it, can be changed at runtime, so that the response of many different detectors can be reproduced by a single instrument.

Among the detectors that a modern emulator must reproduce, the silicon photomultiplier (SiPM) is one of the most demanding, owing to its fast sub-nanosecond rise and its rich microphysics. A natural source of realistic SiPM events is SimSiPM [1], [2], an open-source framework that simulates the SiPM response cell by cell, modeling photon detection efficiency, optical crosstalk, afterpulsing, and dark counts at the microscopic level. SimSiPM produces a list of photon hits, each with a timestamp and an amplitude, at arbitrary temporal and amplitude resolution; from this list a waveform can be reconstructed once the detector time constants are fixed. SimSiPM, however, runs on a general-purpose processor: it is well suited to offline studies but is not designed to drive an

analog output in real time.

In this work we bridge that gap. We describe a real-time SiPM channel for a next-generation detector emulator in which all signal shaping is performed on a field-programmable gate array (FPGA). The FPGA receives a compact stream of high-level event parameters and synthesizes the analog waveform itself, with no precomputed waveforms, while preserving the full three-time-constant physical response of the SiPM. The remainder of the paper is organized as follows. Section II reviews the SiPM signal model and the form used in the hardware. Section III describes the decoupling of event generation from signal shaping and the temporal quantization scheme. Section IV details the hardware shaping core and the timing-closure strategy. Section V reports the implementation and resource usage, Section VI presents the validation, and Section VII discusses generalization and use cases. Conclusions are drawn in Section VIII.

II. SiPM SIGNAL MODEL

A SiPM is a matrix of single-photon avalanche diodes (microcells) operated in Geiger mode and connected in parallel, each in series with a quenching resistor R_q . When a photon is detected in a cell, it triggers an avalanche that is subsequently quenched; the cell delivers a charge $Q_{\text{cell}} = C_d \Delta V$, where C_d is the junction capacitance and ΔV the overvoltage. Solving the standard single-cell equivalent circuit [3], in which C_d in series with the avalanche switch is in parallel with the quenching parasitic capacitance C_q , biased through R_q and coupled to a readout impedance R_S , yields a current response with three characteristic time constants:

$$\tau_r \approx R_S (C_d + C_q), \quad (1)$$

$$\tau_{ff} \approx R_S C_{\text{grid}}, \quad (2)$$

$$\tau_{fs} \approx R_q (C_d + C_q), \quad (3)$$

namely a fast rise τ_r (sub-nanosecond to about 1 ns), a fast decay τ_{ff} (nanoseconds to tens of nanoseconds, the discharge of the fired cell through the readout), and a slow decay τ_{fs} (tens to hundreds of nanoseconds, the recharge through R_q). The single-cell response is commonly written in the product form

$$V(t) \propto (1 - e^{-t/\tau_r}) \left[\alpha e^{-t/\tau_{ff}} + (1 - \alpha) e^{-t/\tau_{fs}} \right], \quad (4)$$

where α is the fast-component fraction. On top of this deterministic shape, real SiPMs exhibit dark counts (thermally generated avalanches), optical crosstalk (a photon emitted during one avalanche firing a neighboring cell), and afterpulsing (a trapped carrier released after the avalanche, producing a delayed pulse) [9]. These effects are reproduced upstream by SimSiPM and reach the hardware as additional photon hits, so the shaping core needs to implement only the single-cell response (4).

For the hardware we adopt the equivalent linear-superposition form

$$H(t) = (1 - S_f) e^{-t/\tau_{ff}} + S_f e^{-t/\tau_{fs}} - e^{-t/\tau_r}, \quad (5)$$

where S_f is the slow-component fraction. Equation (5) is practically equivalent to (4) when $\tau_r \ll \tau_{ff}$, which holds for

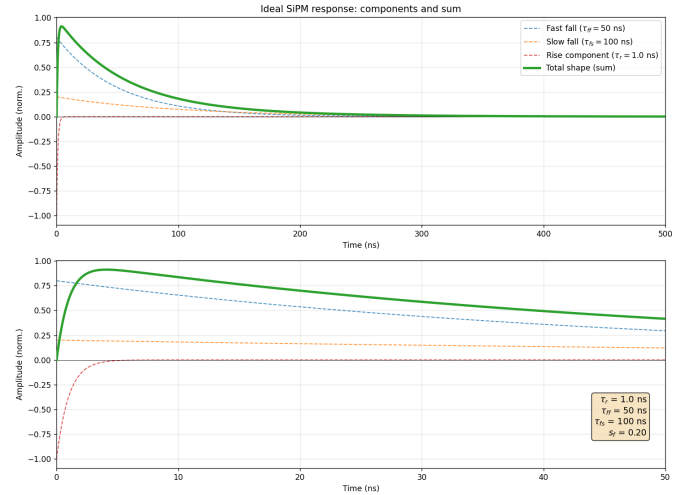


Fig. 2. Ideal SiPM response of (5), shown as the sum (solid) of its rise, fast-decay, and slow-decay components (dashed), for typical parameters $\tau_r = 1$ ns, $\tau_{ff} = 50$ ns, $\tau_{fs} = 100$ ns, and $S_f = 0.20$. Top: 0–500 ns. Bottom: zoom on the rising edge.

any realistic SiPM, and it has the decisive advantage of being a *sum* of three independent exponentials. As shown in Fig. 2, the total shape decomposes naturally into a rise term and two decay terms, each of which can be realized as a separate first-order recursive filter. The four parameters τ_r , τ_{ff} , τ_{fs} , and S_f are programmable at runtime.

III. SYSTEM ARCHITECTURE

The central design choice is to *decouple event generation from signal shaping*. The software side—the processing system, a Geant4 simulation [4], or SimSiPM—produces the list of events, that is, *what* happens and *when*. The FPGA side takes those events and shapes them into the analog waveform, that is, *how* the detector responds. Events are transmitted as pairs $(\Delta t, A)$, where Δt is the inter-arrival time with respect to the previous event and A is the summed amplitude. This wire format yields a drastic reduction in bandwidth compared with streaming raw waveforms, makes the detector-model parameters reconfigurable at runtime without re-synthesis, and lets a single shaping core serve many physics models.

A. Temporal Quantization

On a processor, photon timestamps have effectively arbitrary precision. In digital hardware, time is discrete: every operation advances on the clock edge. The shaping core runs on a fabric clock of 156.25 MHz, corresponding to a coarse cycle of 6.4 ns. To recover sub-cycle timing, each cycle is subdivided into eight sub-phases of 0.8 ns. The simulation events are therefore binned upstream into sub-phases: any hits falling in the same 0.8 ns bin are summed, and only the non-empty bins are transmitted, as (sub-phase index, summed amplitude) pairs. Figure 3 illustrates this process for a 20-photoelectron event: the continuous-time hits (top) are accumulated into the 0.8 ns sub-phase bins that constitute the hardware input (bottom). Pile-up is absorbed natively, because multiple hits in the same

cycle simply add. The effective trigger time resolution is thus 0.8 ns while the fabric runs at a modest 156.25 MHz.

B. Data Path

Figure 4 shows the system data path. Quantized events enter through one of two paths: from the processing system (PS) of the system-on-chip over an AXI interface, or streamed from an external host via 10-gigabit Ethernet (10 GbE) using the User Datagram Protocol (UDP). The 10 GbE link connects directly to the programmable logic (PL), bypassing the PS so that software does not become a bottleneck at high event rates. Both paths feed a common event first-in-first-out (FIFO) buffer. A sub-phase scheduler then groups all events belonging to a given clock cycle, maps each Δt onto a cycle and a sub-phase, and hands the result to the shaping core. The temporal quantization itself is performed upstream (in the PS or in the UDP sender), which keeps the programmable logic lean.

IV. HARDWARE SHAPING CORE

The shaping core, detailed in Fig. 5, exploits the superposition form (5). Each exponential is realized as a one-pole infinite-impulse-response (IIR) filter

$$y[n] = M y[n-1] + x[n], \quad M = e^{-T_S/\tau}, \quad (6)$$

where $T_S = 0.8$ ns is the sub-phase period and M is the pole corresponding to a given time constant τ . Three such banks run in parallel—one for the rise, one for the fast decay, and one for the slow decay—rather than in cascade, which is possible precisely because the adopted response is a linear superposition. Each bank is replicated across the eight sub-phases.

A. Sub-Phase Precision via Pre-Convolution

A trigger that arrives at sub-phase k does not affect only that sub-phase: it propagates to the later sub-phases of the same cycle and into the following cycle, weighted by the pole. This sub-phase precision is obtained with an input pre-convolution stage that injects each impulse into the eight sub-phase lanes weighted by the pre-loaded coefficients $1, M, M^2, \dots, M^7$, a finite-impulse-response kernel of length eight per bank. The outputs of the three banks are combined by a weighted combiner that forms

$$y_{\text{out}} = (1 - S_f) y_{ff} + S_f y_{fs} - y_r, \quad (7)$$

reproducing (5). The result is saturated, quantized to 16 bits, and up-sampled by a factor of two through linear interpolation, yielding 16 samples per cycle at 0.4 ns spacing, that is, 2.5 gigasamples per second (GS/s).

B. Timing Closure: The Look-Ahead Transformation

The naive recursion (6) has a feedback distance of one: within a single fabric cycle, the result must be multiplied, accumulated, and fed back. The internal pipeline registers of the digital signal processing (DSP) slices cannot be used, because the output is needed in the same cycle in which it is produced, and the design does not close timing at 156.25 MHz. We

therefore apply a look-ahead transformation [5] that rewrites the recursion as

$$y[n] = M^2 y[n-2] + M x[n-1] + x[n], \quad (8)$$

which has a feedback distance of two while preserving the same impulse response and the same pole. As illustrated in Fig. 6, two cycles are now available to close the feedback loop, which is exactly what is needed to enable the MREG pipeline register inside the DSP48E2 slice. Two further measures support timing closure: M^2 is precomputed and latched (read before write), so that no cascaded multiplications occur in the same cycle; and the inter-phase coefficients are truncated to 27 bits so that every multiplication fits in a single DSP48E2 (a 27×18 multiplier). With these techniques the core reaches an initiation interval of one ($\text{II} = 1$) with three banks \times eight phases at 156.25 MHz, meaning that it accepts eight sub-phase triggers and eight amplitudes on every fabric cycle.

V. IMPLEMENTATION AND RESOURCES

The prototype is built around a Zynq UltraScale+ MP-SoC [6], which provides both an ARM processing system and FPGA fabric on a single device. Figure 7 shows the board: at its center the Zynq UltraScale+ device; on one side the 10 GbE small-form-factor-pluggable (SFP) transceiver for the UDP path; and on the other side the two LTC2000 digital-to-analog converters [7] (16 bit, 2.5 GS/s) driving the two analog channels.

The shaping core is described in Vitis HLS [8], that is, in C++ compiled into register-transfer-level code. High-level synthesis is well suited here because the algorithm is arithmetic-intensive: the recursion and the pre-convolution are expressed at the algorithmic level, and the timing techniques of Section IV reduce to a small set of synthesis directives. The synthesized core achieves an initiation interval $\text{II} = 1$ with an end-to-end shaping latency of 13 cycles (83.2 ns). It uses 235 DSP48E2 slices (approximately 65 for the IIR banks and 170 for the pre-convolution), about 10% of the DSP resources available on the device, together with 13684 flip-flops and 24387 lookup tables, and no block or ultra RAM. The modest footprint leaves ample room for replicating the core across several channels.

VI. RESULTS AND VALIDATION

The input photon-hit sequences used in the following tests were generated with SimSiPM and then temporally quantized, so that the stimulus reflects the full SiPM microphysics. The hardware output was validated by comparing it against two references: a software simulation of the same filter operating on the binned input (which tests the fixed-point implementation), and the ideal analytical response of (5) evaluated without binning (which tests the effect of temporal quantization). All measurements use typical SiPM parameters: $\tau_r = 1$ ns, $\tau_{ff} = 50$ ns, $\tau_{fs} = 100$ ns, and $S_f = 0.20$.

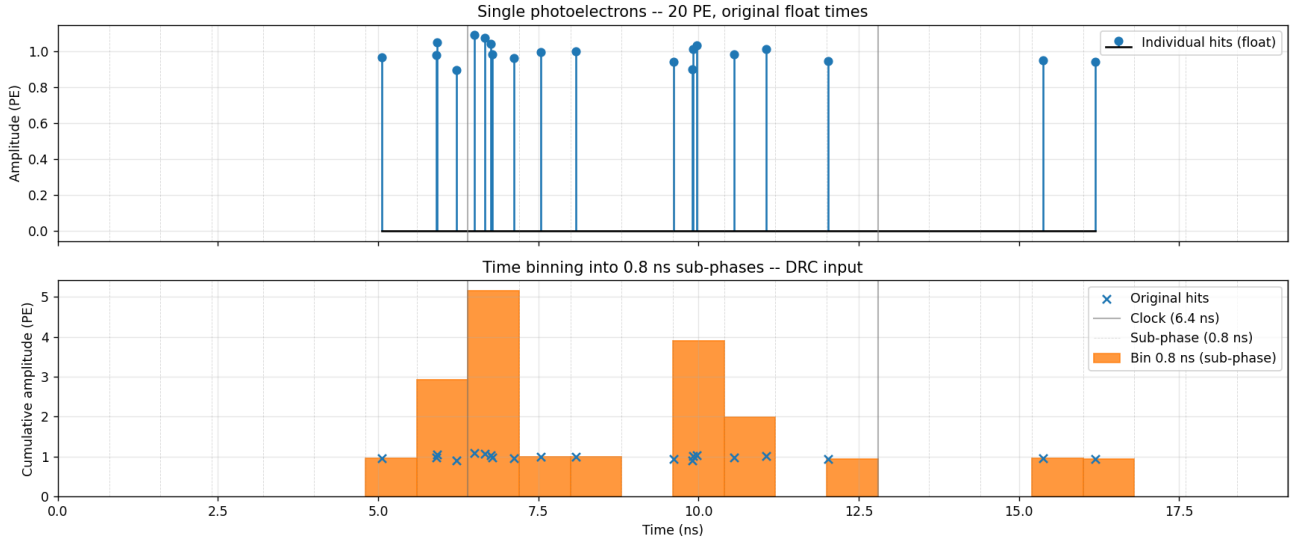


Fig. 3. Temporal quantization of a 20-photoelectron event. Top: individual photon hits with continuous-time timestamps as produced by the simulation. Bottom: the same hits accumulated into 0.8 ns sub-phase bins, which form the input to the hardware shaping core. Vertical lines mark the 6.4 ns coarse clock and the 0.8 ns sub-phase grid.

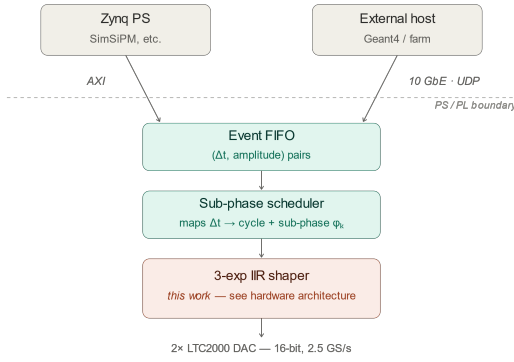


Fig. 4. System architecture and data path. Quantized $(\Delta t, A)$ events arrive either from the processing system over AXI or from an external host over 10-gigabit Ethernet directly into the programmable logic. Both paths share a common event FIFO, a sub-phase scheduler, and the three-exponential shaping core, which drives two digital-to-analog converters.

A. Single Event

Figure 8 shows a 20-photoelectron event—the same event used in Fig. 3. The hardware output, the binned reference, and the ideal curve overlap to within the digital-to-analog converter quantization noise. The agreement with the binned reference confirms that the fixed-point filter behaves as designed; the agreement with the ideal, unbinned curve confirms that the 0.8 ns quantization does not degrade the pulse shape. The relative error (bottom panel) is below the percent level over the whole pulse except for a transient at the very beginning of the rising edge, where the signal amplitude is small and the relative error is therefore dominated by quantization; the residual saw-tooth pattern visible on the far tail is the linear-interpolation ripple at the level of a few least-significant bits.

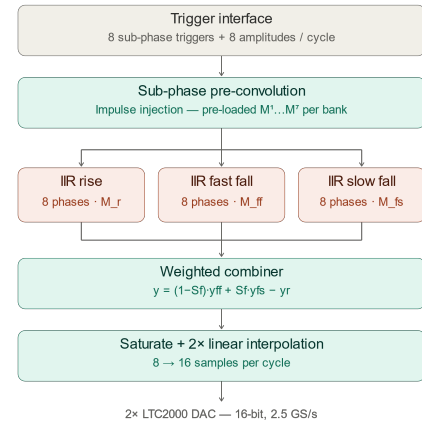


Fig. 5. Hardware shaping architecture. The trigger interface delivers eight sub-phase triggers and eight amplitudes per cycle. A pre-convolution stage provides sub-phase precision; three parallel one-pole IIR banks (rise, fast fall, slow fall), each replicated over the eight sub-phases, are combined and up-sampled before the digital-to-analog converters.

B. Pile-Up

Because the IIR is a linear operator, overlapping events should simply add. To verify this, three events of 20, 15, and 25 photoelectrons were generated at $t = 10, 80,$ and 200 ns, respectively, so that each event lands on the decay tail of the previous one. Figure 9 shows that the hardware output matches both the binned reference and the ideal sum within the converter noise: the shape remains clean even when an event sits on top of a long decay tail, confirming that pile-up is handled correctly by superposition.

C. Uncertainty Discussion

The dominant sources of deviation are well understood and bounded. The temporal quantization introduces a timing uncer-

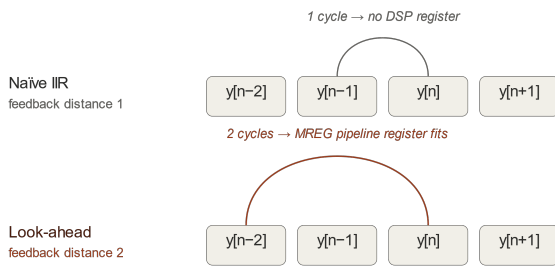


Fig. 6. Look-ahead transformation. The naive IIR (top) has feedback distance one and leaves no room for a pipeline register. The look-ahead form (8) (bottom) has feedback distance two, providing the two cycles needed to enable the DSP48E2 MREG register while preserving the same pole and impulse response.



Fig. 7. Prototype board. The central heat sink covers the Zynq UltraScale+ MPSoC; the two LTC2000 digital-to-analog converters and their analog outputs are at the top left, and the 10-gigabit SFP transceiver is at the bottom left.

tainty of at most one sub-phase (0.8 ns); its effect on the pulse shape is below the converter noise, as the comparison with the unbinned ideal curve in Fig. 8 demonstrates. The fixed-point arithmetic contributes a truncation error consistent with the 27-bit coefficient and 16-bit output word lengths, visible only as a few-least-significant-bit ripple on the tail. The $2\times$ linear interpolation introduces a small, deterministic interpolation error between true samples. None of these exceeds the percent level over the body of the pulse, so the emulated waveform is faithful to the analytical model to within the resolution of the output stage.

VII. GENERALIZATION AND USE CASES

The primary use case is a front-end testbench: the emulator output is connected directly to the input of the front-end under development, which then sees signals indistinguishable from those of a real SiPM, with perfect ground truth and

full reproducibility, and without any detector or radioactive source. Beyond SiPMs, the same hardware generalizes in several directions. Coupled with Geant4 [4], particle-level interactions can be transformed directly into analog detector outputs in real time. By tuning S_f and the decay constants, the core reproduces the differing pulse shapes of gamma rays and neutrons in organic scintillators, enabling pulse-shape-discrimination studies with the same firmware and a software-only switch. More generally, the shaping core is a primitive: any detector whose response is described by three time constants can be emulated without re-synthesis, simply by reprogramming the four parameters.

VIII. CONCLUSION

We have demonstrated a real-time, physically accurate emulation of silicon photomultiplier signals on an FPGA. The system synthesizes the full three-time-constant SiPM response in hardware from a compact stream of high-level events, producing a 16-bit analog output at 2.5 GS/s with an end-to-end shaping latency of 83 ns and an initiation interval of one. A temporal-quantization scheme combined with a sub-phase pre-convolution delivers an effective trigger time resolution of 0.8 ns from a modest 156.25 MHz fabric clock, and a look-ahead transformation of the recursive filter makes timing closure possible. The output was validated against both an analytical reference and the SimSiPM framework, for single events and for pile-up, with agreement at the level of the output quantization noise. The architecture is general: the same shaping core serves any detector with a three-time-constant response, which makes the emulator a flexible instrument for hardware-in-the-loop testing of front-end electronics. Future work will pursue end-to-end validation with real front-end electronics, a pulse-shape-discrimination demonstration, and a direct Geant4-to-converter pipeline.

REFERENCES

- [1] E. Proserpio and R. Santoro, "SimSiPM: a library for SiPM simulation," Como, Italy, 2021. [Online]. Available: <https://github.com/EdoPro98/SimSiPM>. Accessed on: June 28, 2026.
- [2] E. Proserpio and R. Santoro, "A simulation framework for SiPMs," 2024, arXiv:2411.16710.
- [3] F. Corsi *et al.*, "Modelling a silicon photomultiplier (SiPM) as a signal source for optimum front-end design," *Nucl. Instrum. Methods Phys. Res. A*, vol. 572, no. 1, pp. 416–418, Mar. 2007.
- [4] S. Agostinelli *et al.*, "Geant4—a simulation toolkit," *Nucl. Instrum. Methods Phys. Res. A*, vol. 506, no. 3, pp. 250–303, Jul. 2003.
- [5] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. New York, NY, USA: Wiley, 1999, ch. 10.
- [6] *Zynq UltraScale+ MPSoC Data Sheet: Overview*, DS891, Advanced Micro Devices (Xilinx), San Jose, CA, USA, 2022.
- [7] *LTC2000 16-Bit, 2.5 GSPS Digital-to-Analog Converter Data Sheet*, Analog Devices, Wilmington, MA, USA, 2014.
- [8] *Vitis High-Level Synthesis User Guide*, UG1399, Advanced Micro Devices (Xilinx), San Jose, CA, USA, 2023.
- [9] F. Acerbi and S. Gola, "Understanding and simulating SiPMs," *Nucl. Instrum. Methods Phys. Res. A*, vol. 926, pp. 16–35, May 2019.
- [10] *DT5800D Digital Detector Emulator*, CAEN S.p.A., Viareggio, Italy. [Online]. Available: <https://www.caen.it>. Accessed on: June 28, 2026.

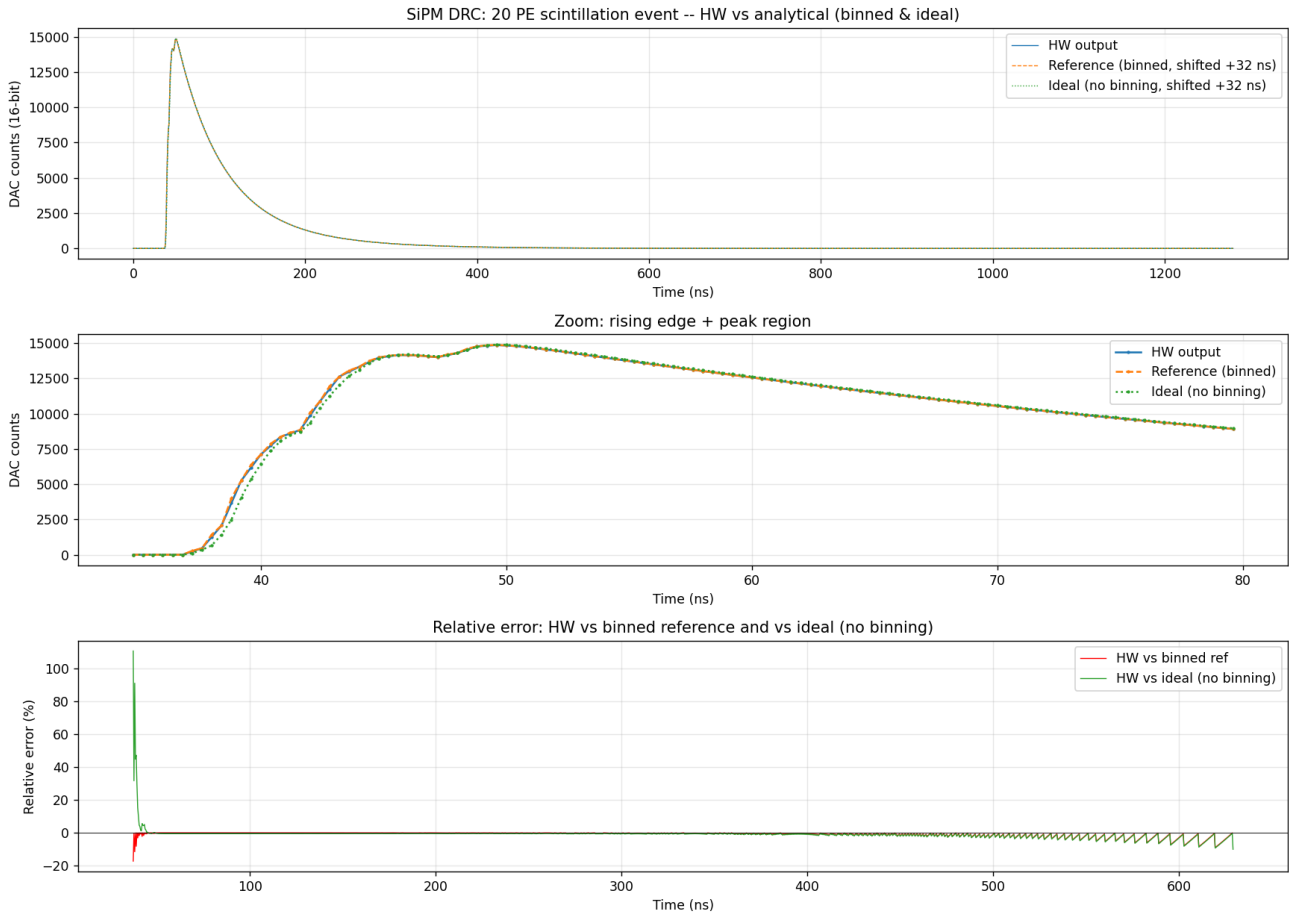


Fig. 8. Single-event validation for a 20-photoelectron event. Top: full waveform. Middle: zoom on the rising edge and peak. Bottom: relative error of the hardware output with respect to the binned reference and to the ideal (unbinned) analytical curve. The three traces overlap to within the digital-to-analog converter noise.

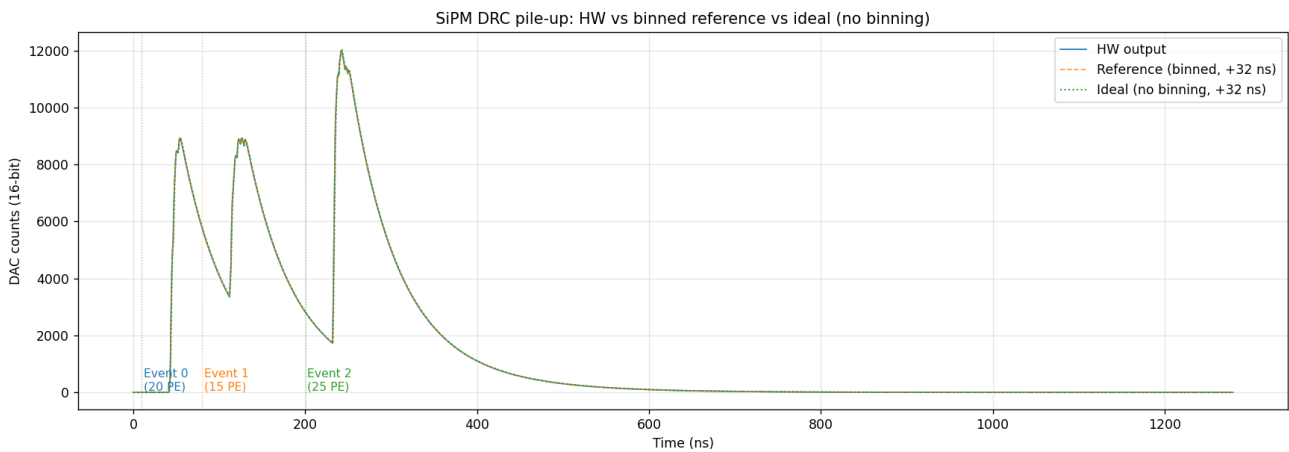


Fig. 9. Pile-up validation. Three scintillation-like events of 20, 15, and 25 photoelectrons at $t = 10, 80,$ and 200 ns overlap on each other's decay tails. The hardware output matches the binned reference and the ideal sum within the digital-to-analog converter noise.