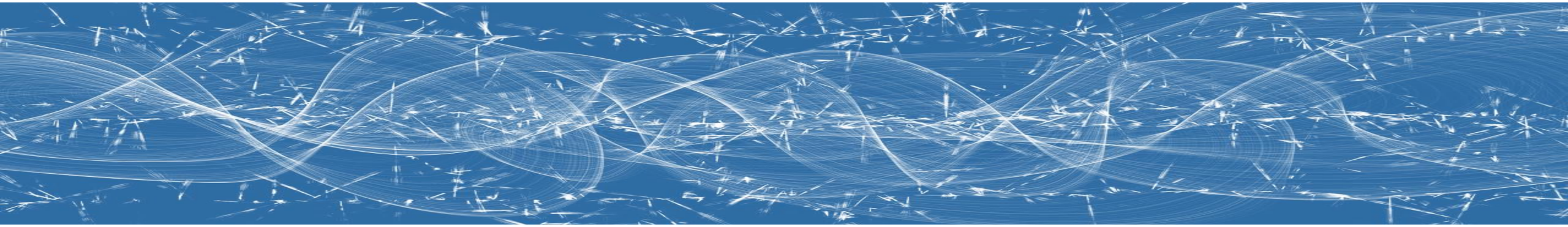
# Real-Time FPGA-Based SiPM Detector Emulation using Temporally Quantized Model

S. Carsi[1,2,*], E. Proserpio[1,2], A. Abba[1], F. Caponio[1], V. Arosio[1]

*Speaker: s. carsi@nuclearinstruments.eu

1. Nuclear Instruments SRL

2. Università degli Studi dell'Insubria



IEEE Realtime 2026 – La Biodola, Isola d'Elba (Italy)
May 27th, 2026

# What is a detector emulator?

**What it is**

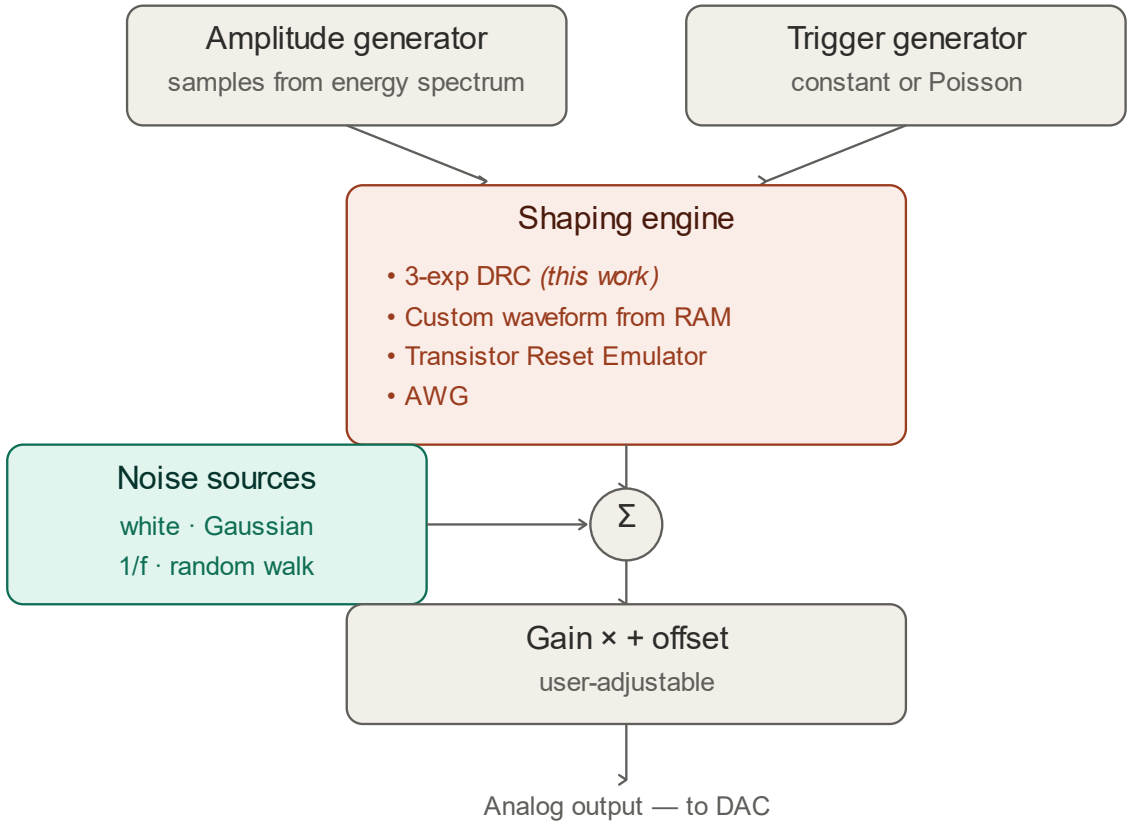
- An instrument that **generates** analog **signals indistinguishable** from a **real detector**

**Use cases:**

- Front-end characterization (preamp, shaper, ADC, trigger) without physical detector
- Hardware-in-the-loop validation of firmware/DAQ
- Stress testing under controlled rate, pile-up, background
- **Perfect ground truth** for reconstruction-algorithm validation

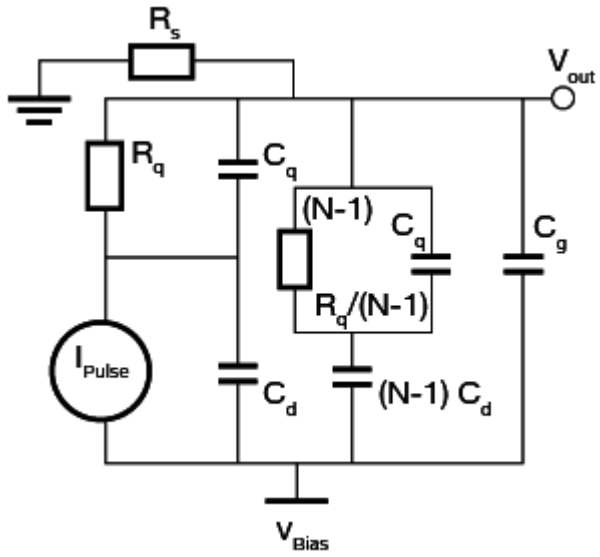
**Key differentiator vs AWGs**

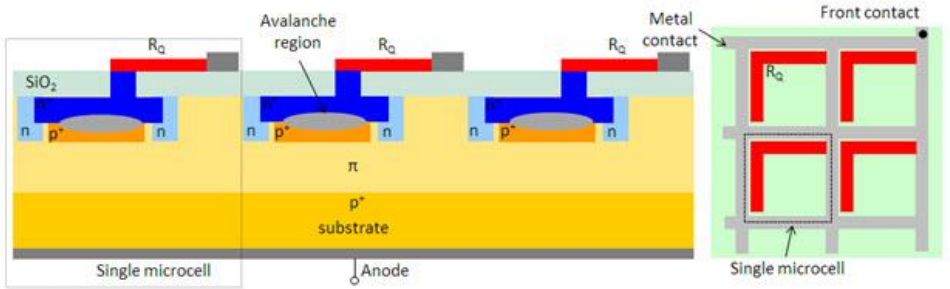
- **Reproduces** detector **physics**, not just a fixed **waveform** → runtime-controlled response



# Silicon Photomultipliers (SiPM) basics

- SiPM = **matrix** of SPADs (microcells) connected in **parallel**, each with a **quenching resistor** $R_q$
- Each cell undergoing **avalanche** produces a charge $Q_{cell} = C_d \cdot \Delta V$, where $C_d$ is the junction capacitance and $\Delta V$ the overvoltage
- **Equivalent circuit model** (per cell): $C_d$ in series with the avalanche switch, in parallel with $C_q$ (quenching parasitic capacitance), the whole thing biased through $R_q$ and coupled to the readout (typically 50 Ω)



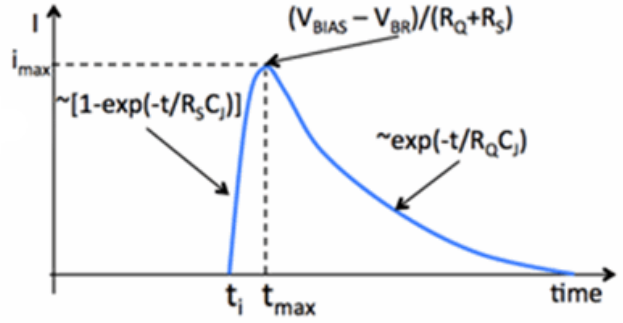



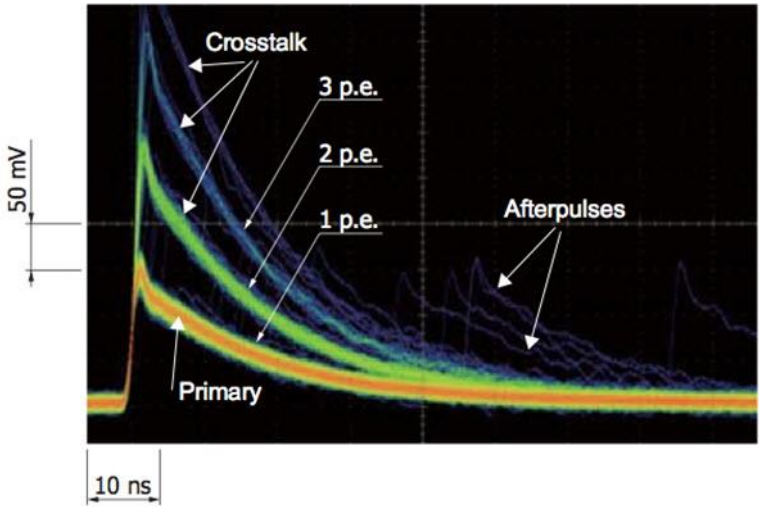
# SiPM Signal Shape

- Solving the circuit equations yields a **three-time-constant shape**:
  - **Rise** $\tau_r \approx R_S(C_d + C_q)$ — sub-ns to ~1 ns
  - **Fast decay** $\tau_{ff} \approx R_S \cdot C_{grid}$ (cell discharges through the readout) — ns to tens of ns
  - **Slow decay** $\tau_{fs} \approx R_q(C_d + C_q)$ — tens to hundreds of ns
- Common analytical form:

$$V(t) \propto (1 - e^{-\frac{t}{\tau_r}})[\alpha e^{-\frac{t}{\tau_{ff}}} + (1-\alpha)e^{-\frac{t}{\tau_{fs}}}]$$

- Plus microphysics → all add complexity
  - Dark counts
  - Optical crosstalk
  - Afterpulsing



# SimSIPM

- **Open-source** framework for cell-by-cell **simulation** of the **SiPM** response
- **Models** PDE, optical crosstalk, afterpulsing, and dark counts at the **microscopic level**

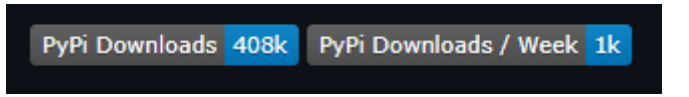
**Output:**

- **List** of "photon hits" with **timestamp** and **amplitude**, at arbitrary temporal and amplitude resolution (idealized)
- Fixing the **timing constants**, you can **build the waveform**

**Limitation**:

- Runs on **CPU**, not real-time, produces abstract high-resolution waveforms
- **Good** for **offline** studies
- **Not designed** to drive an **analog output** in **real time**

SiPM++




https://github.com/edopro98/simsipm

# Motivation: from "PC simulator" to real-time emulator

**Goal**:

- **Real-time SiPM** channel for our next-gen **detector emulator**

**Targets**:

- No precomputed waveforms — **fully in-hardware shaping**
- Physical accuracy: **3-exp shape** + microphysics
- 16-bit @ **2.5 GS/s** analog output (DAC)
- Low CPU overhead, low and deterministic latency

**Challenge**

- **Stream** photon-hit sequences
- **Shape** in **real time** on FPGA

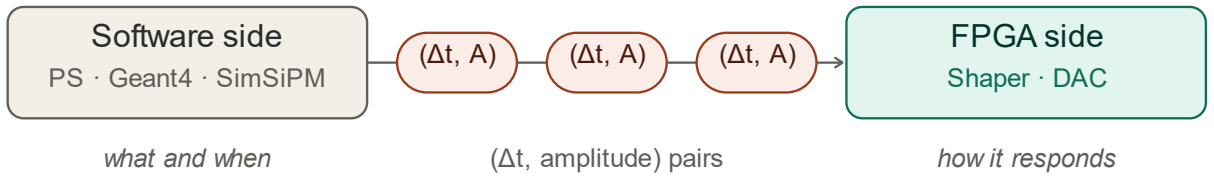
# Core idea: decouple *event generation* from *signal shaping*

- **Software side** (PS / Geant4 / **SimSiPM**): produces **event lists** — *what* and *when*
- **FPGA side**: **shapes events** into **analog waveforms** — *how the detector responds*

Wire format: lists of (Δt, amplitude) pairs — Δt = inter-arrival time

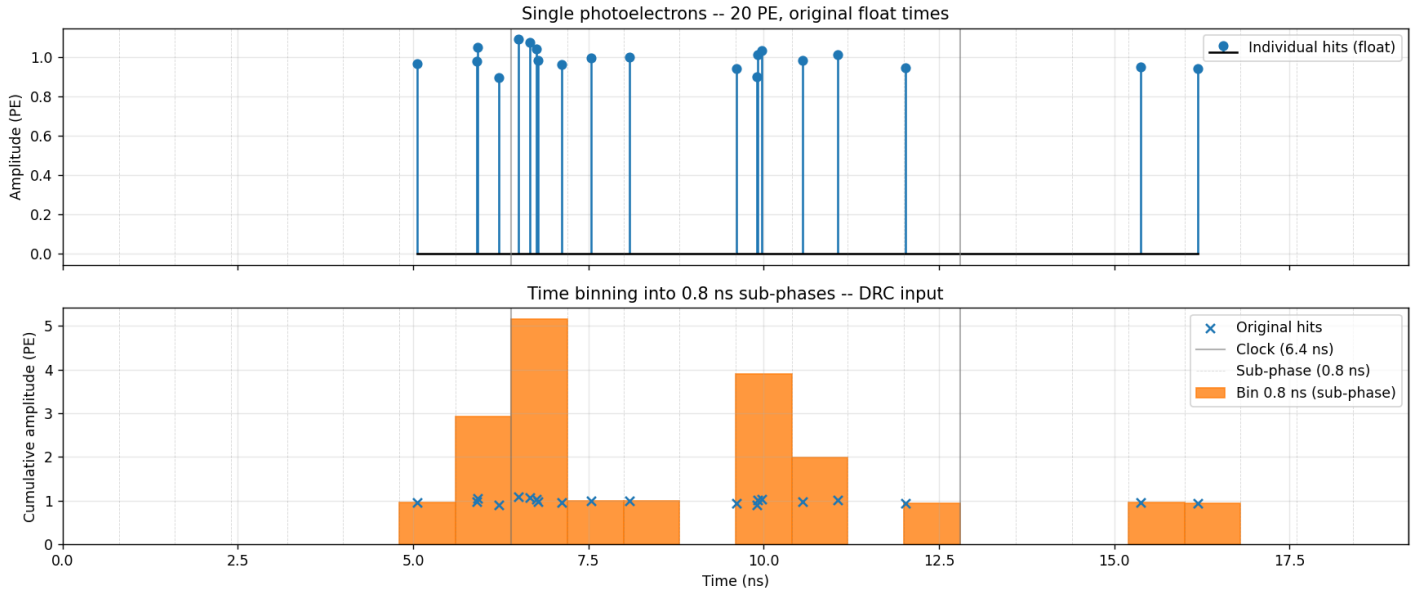
**Implications**:

- Drastic **bandwidth reduction** vs streaming raw waveforms
- Detector model **parameters reconfigurable** at runtime — no re-synthesis
- Single shaping core ↔ many physics models



# Three-stage temporal hierarchy

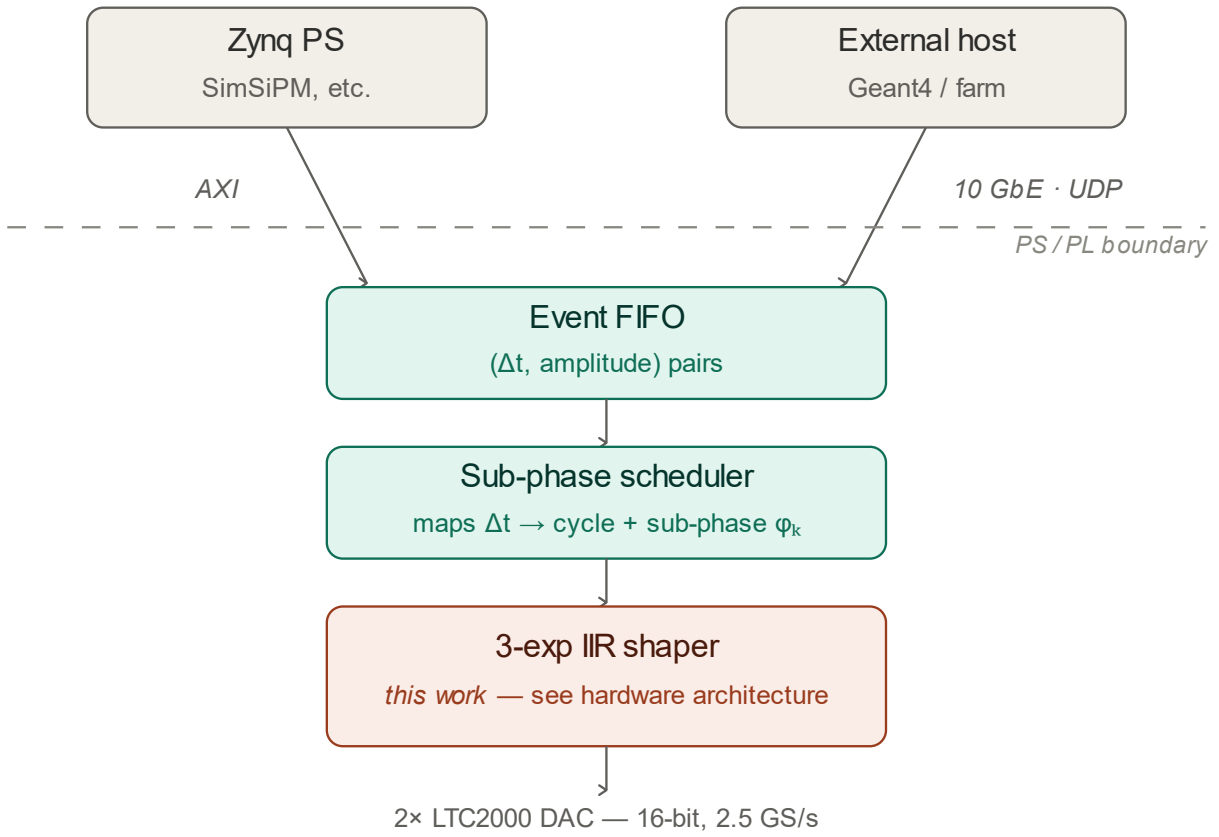
- **Coarse** — fabric clock @ **156.25 MHz** → 6.4 ns cycle
- **Fine — 8 sub-phases per cycle** → 0.8 ns spacing
- 8-bit trigger vector + 8 parallel amplitudes
- Pile-up natively absorbed (multiple sub-phase hits in same cycle = superposition)
- **Output** — 2× linear interpolation → **16 DAC samples/cycle** @ 0.4 ns = **2.5 GS/s**
- Effective trigger time resolution: 0.8 ns at 156 MHz fabric clock
- Transmit only **non-empty bins**



# System architecture + data path

Event format: **(Δt, amplitude)** pairs
Δt = inter-arrival time vs previous event

- **Temporal quantization done upstream** (PS or UDP sender) — keeps the PL lean
- 10 GbE connects **directly to the PL** — bypasses the PS for high-throughput streaming
- Two input paths **share** the same **downstream pipeline**



# Emulated signal equation + shaping implementation

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

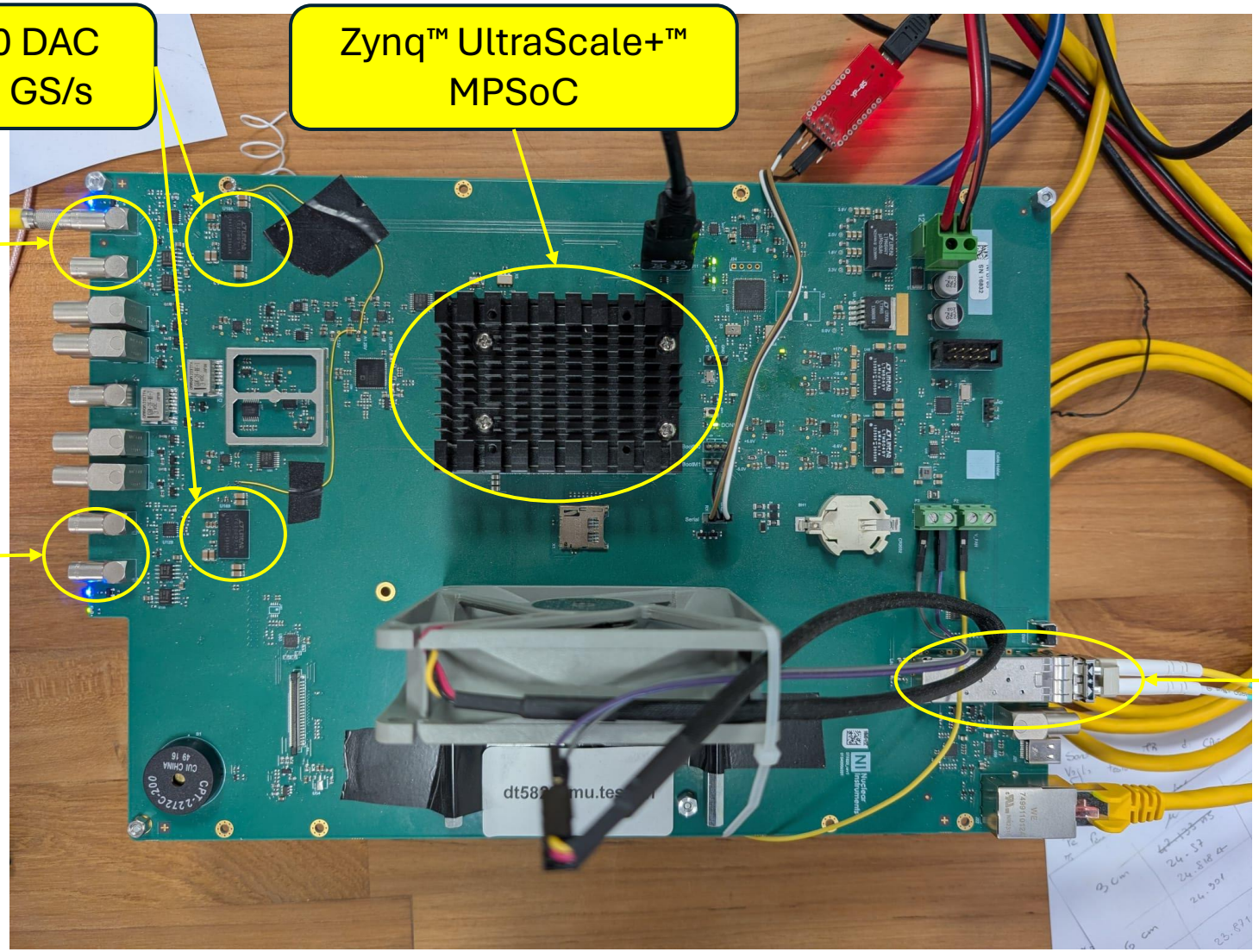
Linear-superposition form of the SiPM response — *practically equivalent* to the textbook product form when $\tau_r \ll \tau_{ff}$, and **decomposes naturally into 3 independent IIR banks**

**Each exponential → 1-pole IIR** (Infinite Impulse Response)
with $M = e^{-T_S/\tau}$, $T_S = 0.8\ ns$

All four parameters runtime-programmable

| Symbol | Meaning |
|---|---|
| $\tau_r$ | Rise time constant |
| $\tau_{ff}$ | Fast decay |
| $\tau_{fs}$ | Slow decay |
| $S_f$ | Slow-component fraction |

# Hardware platform



# Hardware architecture

**Three parallel 1-pole IIR banks**

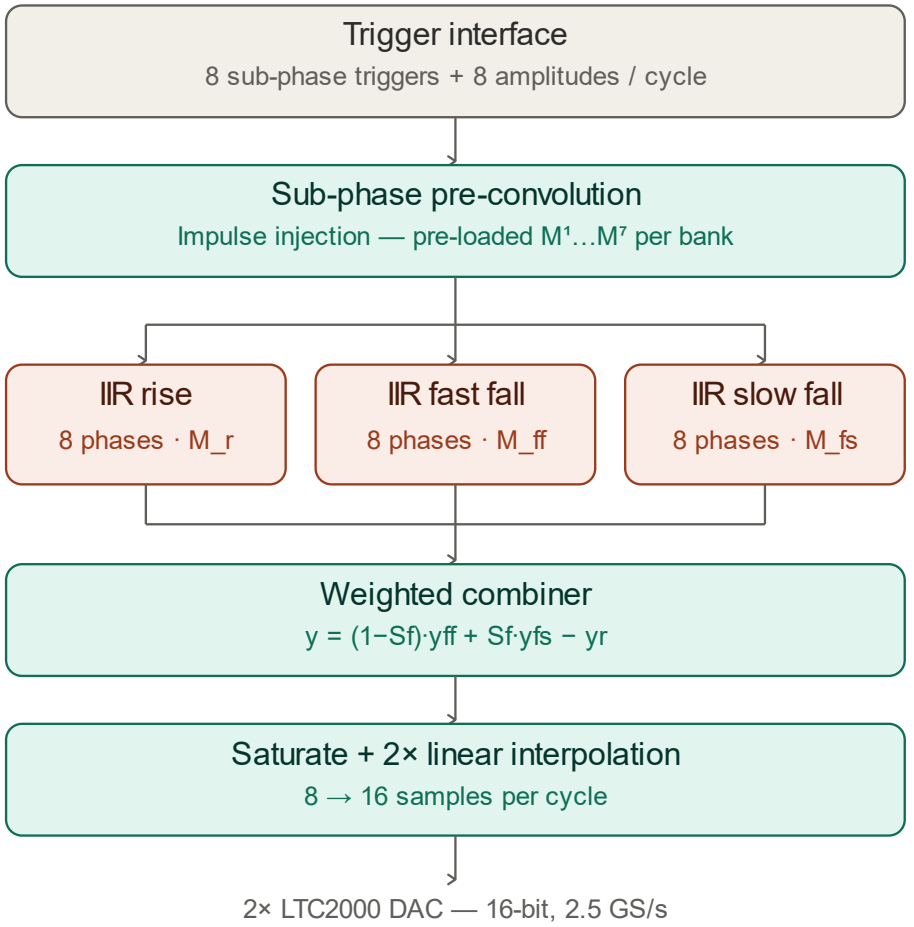
- one per exponential — each replicated across the 8 sub-phases

**Sub-phase precision via input pre-convolution**

- A **trigger** at sub-phase k **propagates** to **later sub-phases** of the same cycle and into the next cycle
- Weighted by **pre-loaded coefficients** $1, M, M^2 \ldots M^7$ (FIR Kernel of length 8)

**Output stage**

- $y_{out} = (1 - S_f)y_{ff} + S_f y_{fs} - y_r$ → saturate → 16-bit → 2× LERP → DAC



# Timing closure: the look-ahead trick

**Naïve IIR** won't close timing at 156.25 MHz

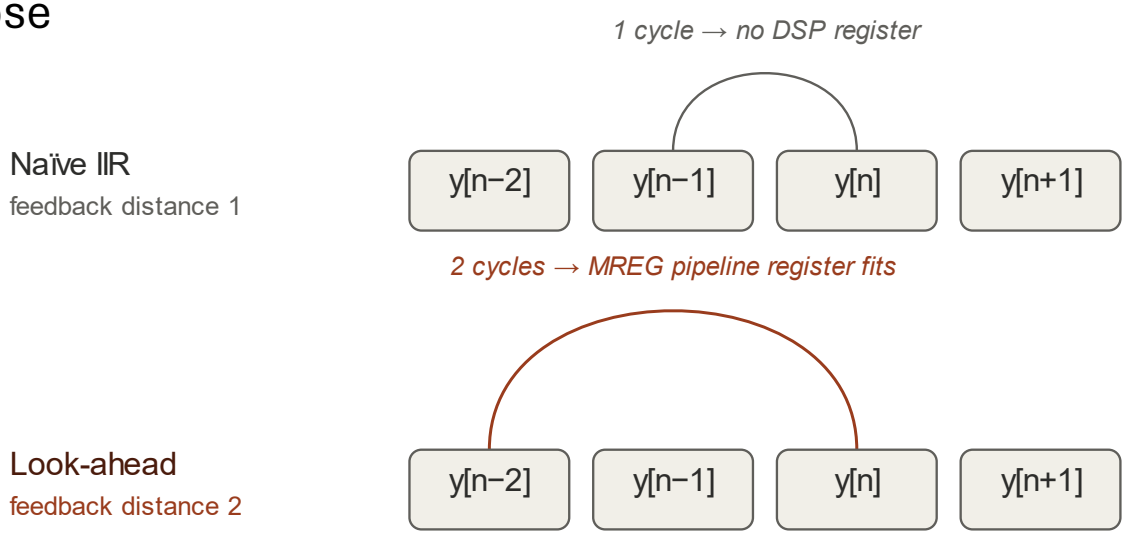
$$y[n] = M \cdot y[n-1] + x[n]$$

- **feedback distance 1**: multiply + accumulate must close in one fabric cycle, **no** DSP **pipeline registers** usable

**Look-ahead transformation**

$$y[n] = M^2 \cdot y[n-2] + M \cdot x[n-1] + x[n]$$

- **feedback distance 2** → DSP48E2 **MREG enabled**
- **same impulse response** and **same pole**




**Supporting tricks**

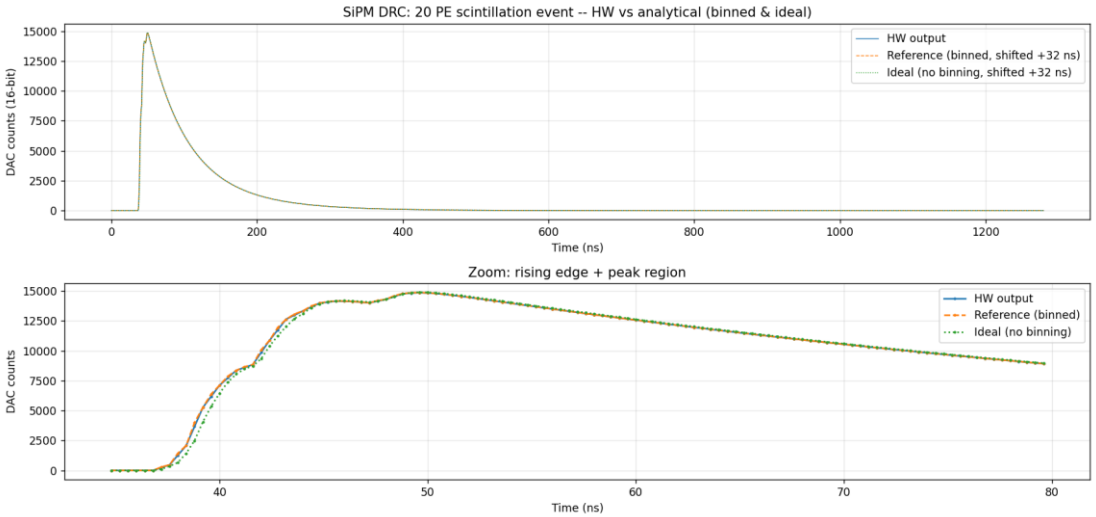
- $M^2$ **precomputed** and latched (read-before-write) → no cascaded multiplies in the same cycle
- Inter-phase **coefficients truncated** to 27 bits → each multiply fits in **one** DSP48E2 (27×18)

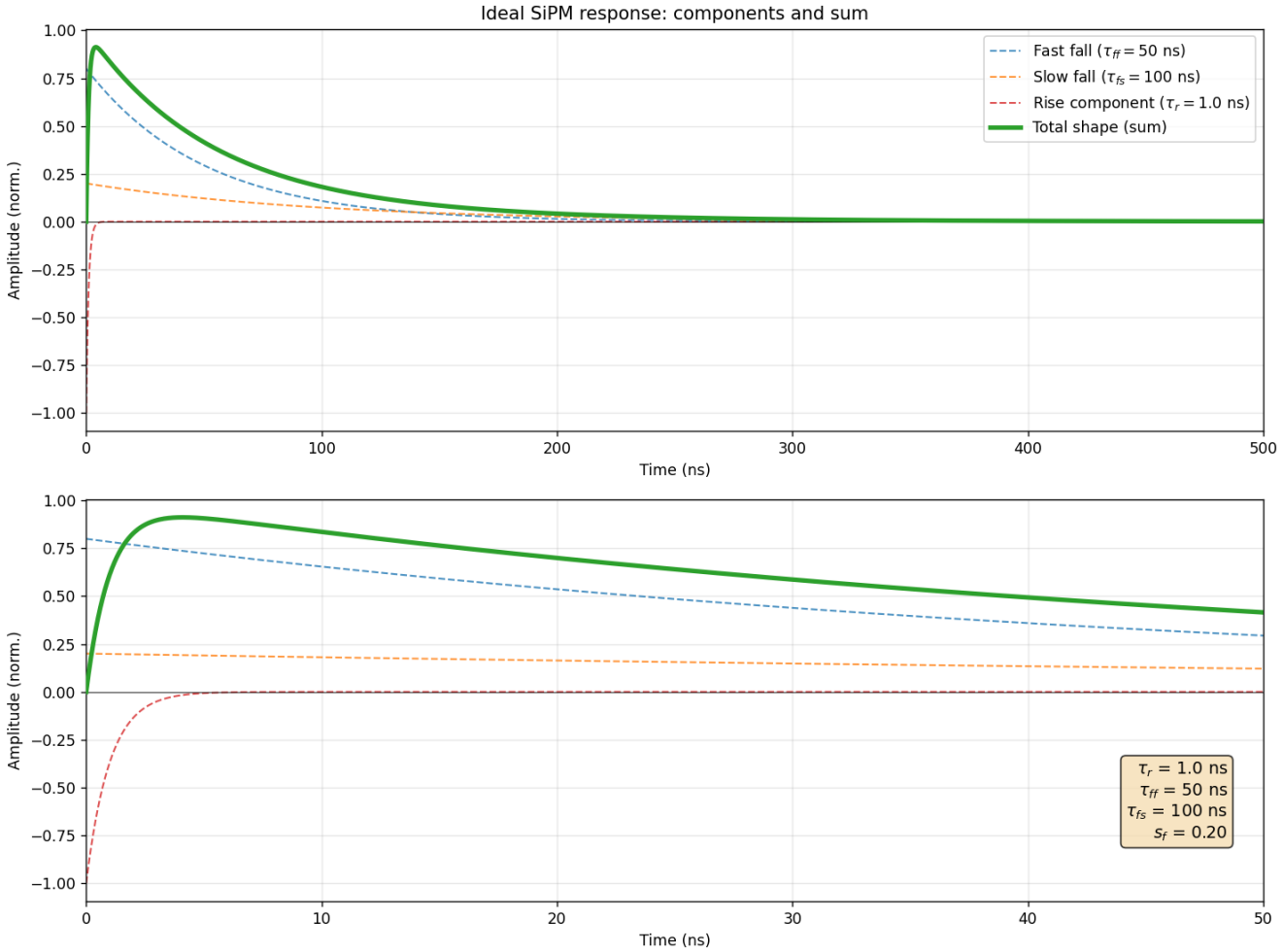
**Result**: Initialization Interval **II=1** closure with 3 banks × 8 phases at 156.25 MHz

# Resulting waveform – Comparison with SimSIPM

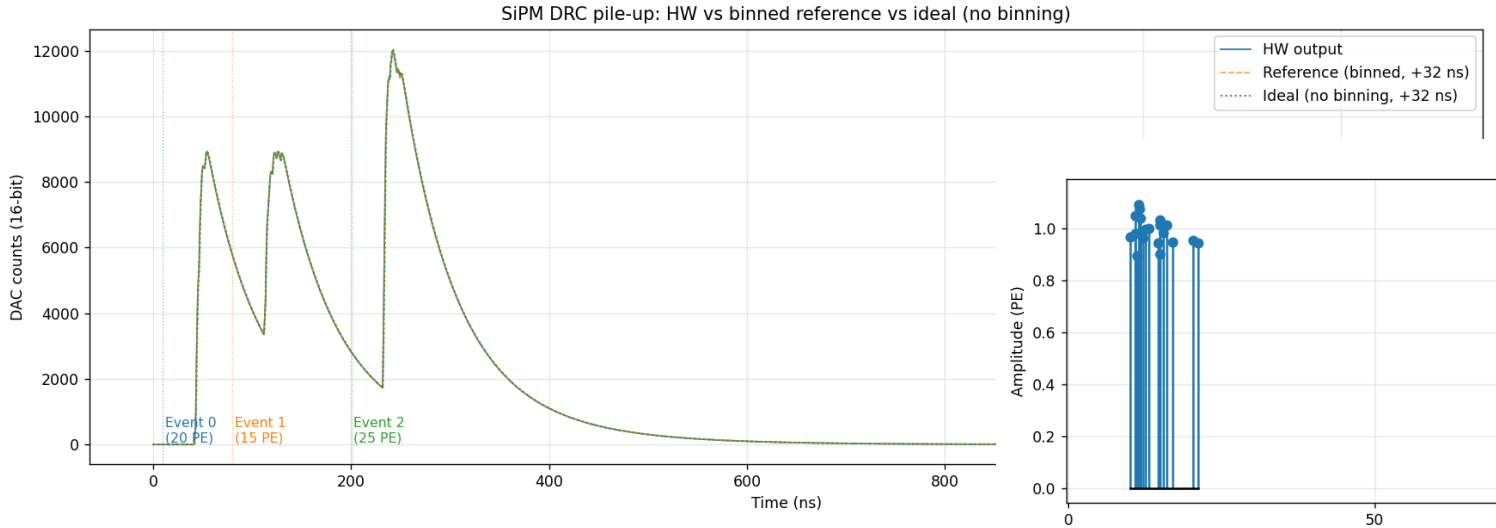
- Validation: HW output vs analytical reference (20 photo-electron event)

- Blue dots: HW output of the core
- Orange: **simulation** of the **filter response**
  Good agreement, the **filter** is **doing the job**
- Green: ideal **3 exponential response**
  The 0.8 ns **quantization does not degrade** the shape



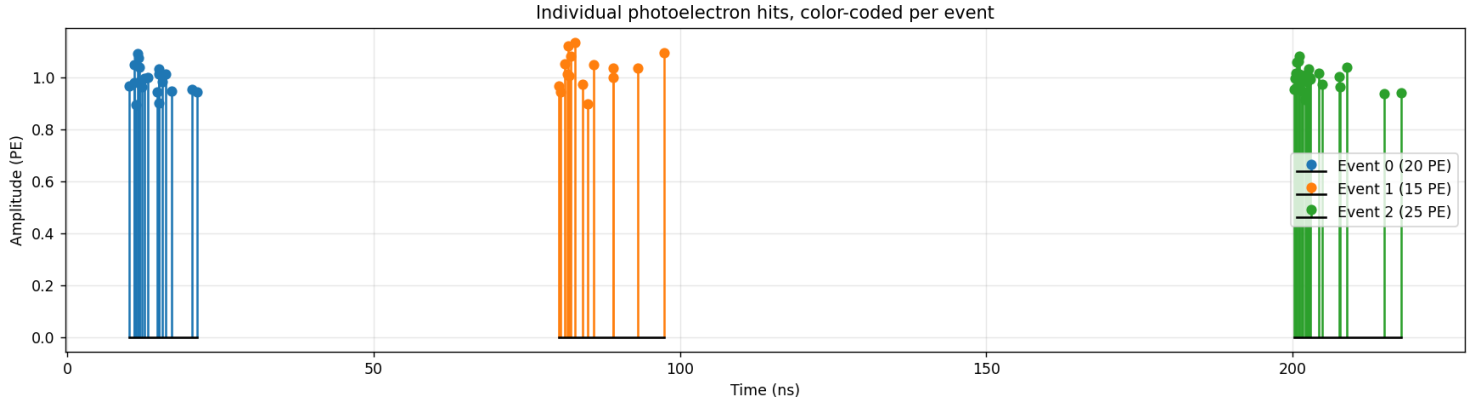



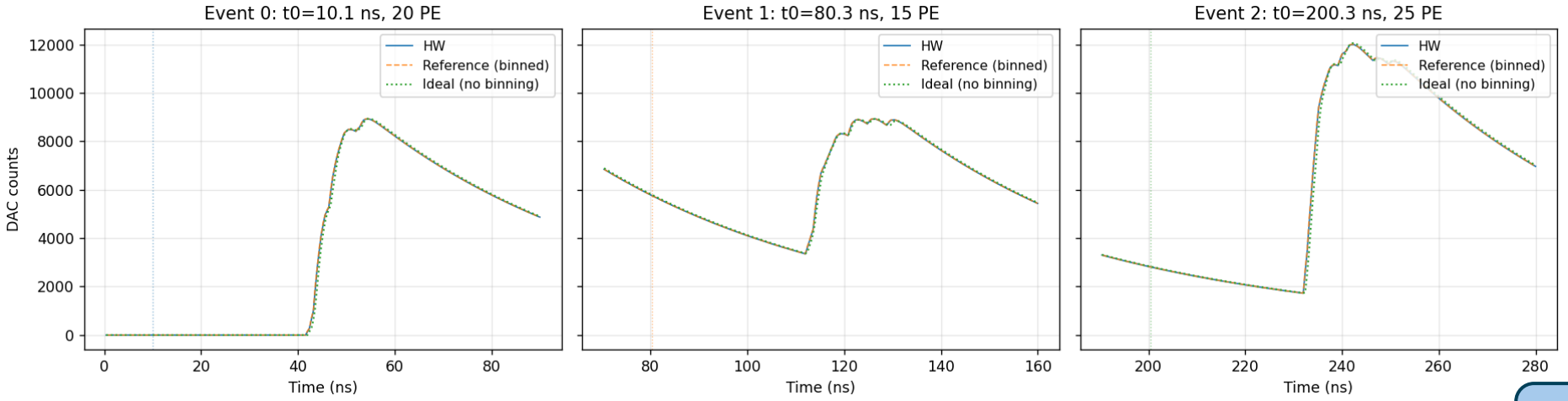
# Pile-up validation: three overlapping scintillation events




**3 events** at t = 10, 80, 200 ns

- HW output matches binned reference and ideal sum within DAC noise



# Implementation status and FPGA resources

**Status**

- working prototype
- validation ongoing

**Implementation**

- written in **Vitis HLS** (High Level Synthesis): C++ code compiled into RTL
- **math-heavy DSP** described at the **algorithmic level** recursion and pre-convolution

DSP48E2: 235 used (≈ 65 IIR + ≈ 170 pre-convolution)
~10% of available DSP resources

| Latency(cycles) | Latency(ns) | Iteration Latency | Interval | Trip Count | Pipelined | BRAM | DSP | FF | LUT | URAM |
|---|---|---|---|---|---|---|---|---|---|---|
| 13 | 83.200 | - | 1 | - | yes | 0 | 235 | 13684 | 24387 | 0 |

Initializazion Interval **II=1**

# Generalization and use cases

Primary use case:

- **Front-end testbench** Drives any analog-input front-end with realistic signals — no physical detector required

Beyond SiPMs:

- **Coupled with Geant4** Particle-level interactions → analog detector output, in real time
- **Pulse Shape Discrimination** Tuning Sf + decay constants reproduces **gamma vs neutron pulses** in organic scintillators — same firmware, software-only switch
- The shaping core is a **primitive** — any detector with a **3-time-constant response** fits, no re-synthesis

# Conclusions

**Demonstrated**

- Real-time, physically-accurate **SiPM emulation on FPGA**
- 2.5 GS/s @ 16-bit output, 83 ns end-to-end shaping latency, **II=1**
- **Sub-phase trigger precision** (0.8 ns) at modest 156.25 MHz fabric clock
- **Validated** against analytical reference and SimSiPM, both single-event and pile-up

**Architecture is general**

- **Reusable** shaping core for arbitrary 3-time-constant detectors

**Next steps**

- End-to-end **validation with real front-end** electronics
- **PSD** demonstration
- Direct **Geant4** → DAC pipeline

# Thanks for your attention!

Stefano Carsi

Nuclear Instruments SRL

s.carsi@nuclearinstruments.eu

# Backup slides

# What is a detector emulator?

- Detector emulator = an instrument that generates **realistic analog signals** "indistinguishable" from those of a real detector
- What it's used for:
  - Testing, debugging, and characterization of front-end electronics (preamp, shaper, ADC, trigger) **without** a physical detector
  - Hardware-in-the-loop: validation of firmware/DAQ with reproducible, controlled scenarios
  - Stress testing: rate, pile-up, backgrounds, anomalous conditions generated on demand
  - Perfect ground truth: energy, timing, and multiplicity are known *exactly* → useful for validating reconstruction algorithms
- Key advantage over arbitrary waveform generators: it reproduces the **physics** of the detector, not just a waveform


Caen DT5800


Caen DT5810