Virtual prototyping of pixel detectors with PixESL framework in High Energy Physics

Jashandeep Dhaliwal, Francesco Brambilla, Davide Ceresa, Stefano Esposito

12th of July 2024

WP5 IC Tech
EP R&D

Introduction

1. Overview
2. ESL design
3. Electronic design flow

Overview

Context:

- EP R&D Programme on Technologies for Future Experiments: IC Technology Work-Package (WP5)
- Developed started in 2023 in the EP-ESE-ME section

Project:

- Name – PixESL (virtual Electronic System-Level prototyping framework for Pixel detectors)
- GitLab repository – https://gitlab.cern.ch/iod/pixesl

Publications:

- [2024] Poster: Virtual prototyping of pixel detectors with PixESL framework in High Energy Physics
- [2024] Paper: SystemC framework for architecture modelling of electronic systems in future particle detectors
- [2023] Poster: Pix-ESL: a SystemC framework for architectural modelling of readout systems in HEP

Electronic System Level – ESL design

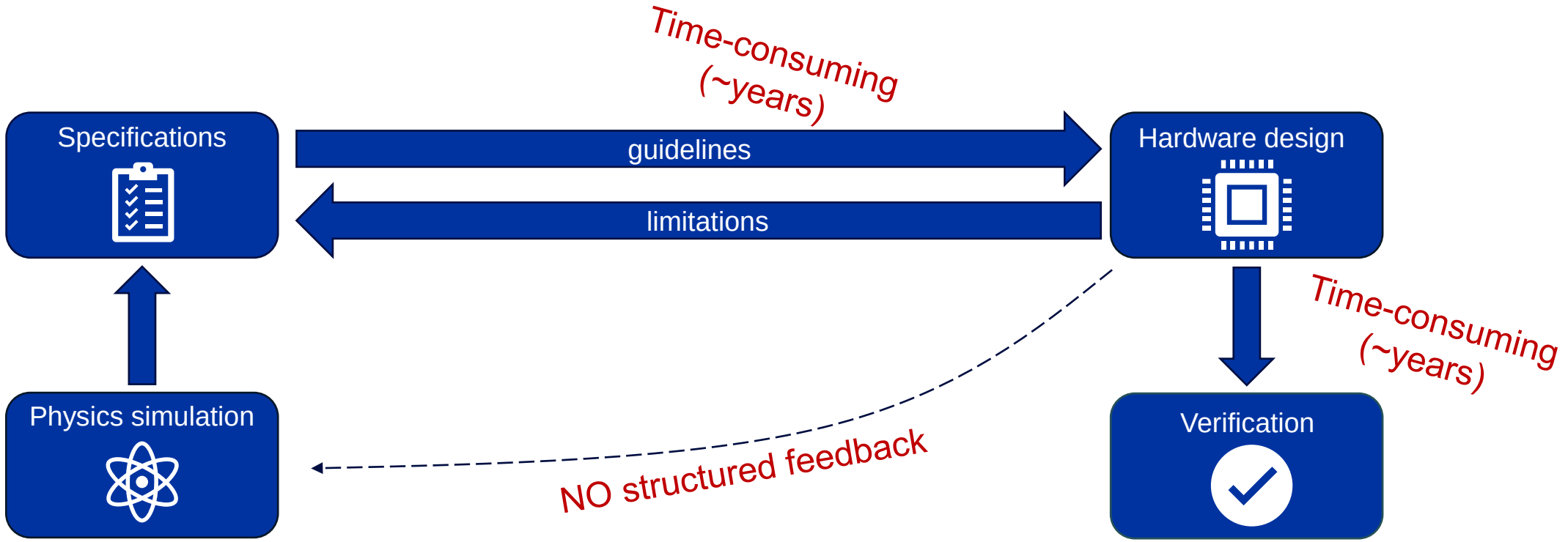
ESL is a methodology for **modelling** and **simulating complex electronic systems**.

Key points:

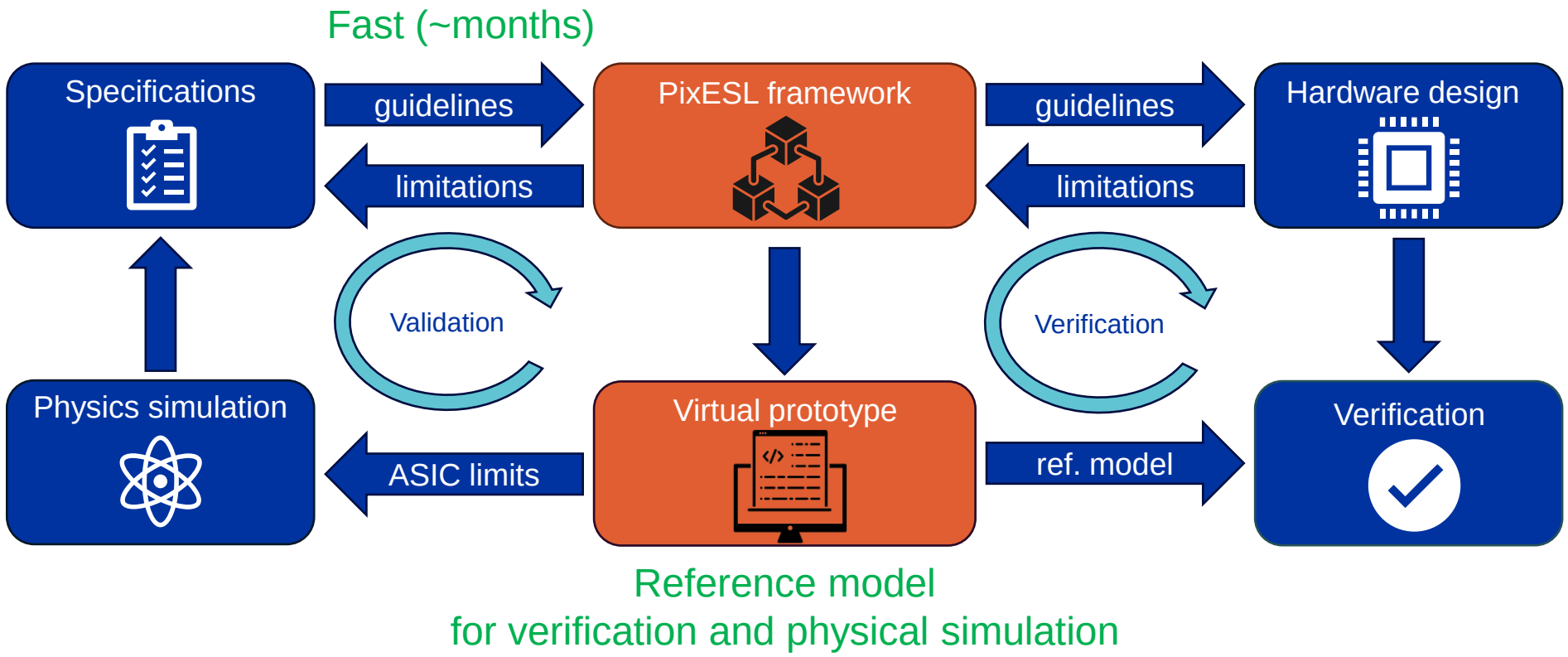
- **Abstraction**
- **System-level perspective**
- **Modelling**
- **Simulation and analysis**
- **Design space exploration**
- **Validation**

PixESL Framework

Electronics design flow



Electronics design flow with ESL prototype



PixESL framework

1. Approach
2. Methodology
3. Front-End model
4. Readout model
5. Co-simulation for chip verification

PixESL - Approach

PixESL applies the **Electronic System Level** concepts to **pixel detectors** design flow for a **cycle-accurate** simulation from **analog front-end** to **chip/system readout**.

Open-source

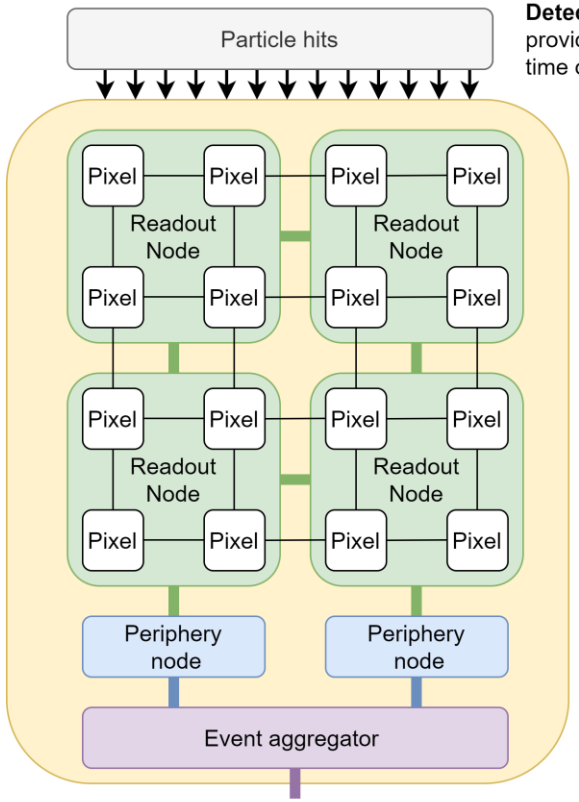
- The model is based on **C++** and SYSTEMC [link]
- Performance analysis is based on **Python**

User-friendly

- **Reconfigurable architectures** and **network**
- **User** and **developer** roles are separated

Reusable

- **Generalized** layers and **standardized** packet transport
- A **library** of useful layers and components
- **Co-simulation** with **SystemVerilog** / **UVM environment**



Detector Simulation e.g. AllPix2, Geant4,..., provides pixel hit input data with charge and time of injection

ROC Virtual Prototype in PixESL provides a C++-based high-level simulation platform to study the entire electronics system chain at an early stage of development.

ReadOut Chip output data with performance metrics

PixESL - Methodology

PixESL applies the **Electronic System Level** concepts to **pixel detectors** design flow for a **cycle-accurate** simulation from **analog front-end** to **chip/system readout**.

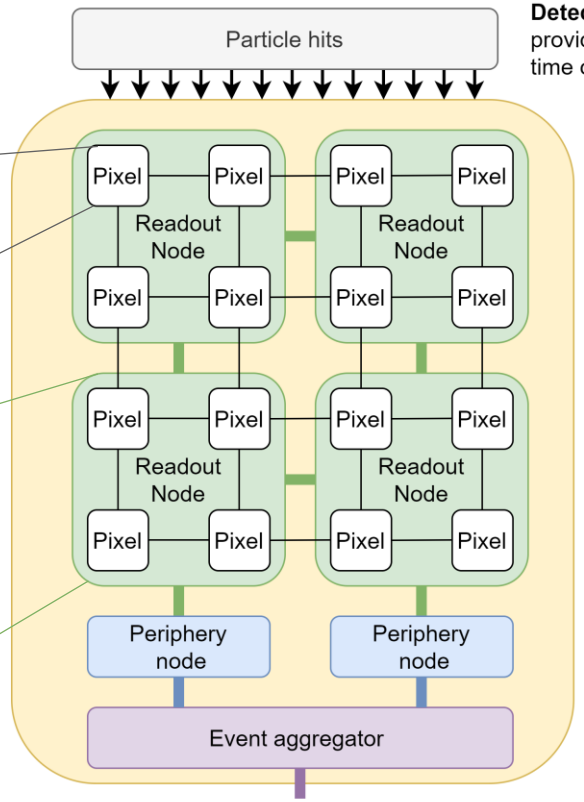
PixESL models the **entire ROC** with 2 components:

Pixel [C++/SystemC]

- Analog Front-End
- Digital Front-End

Readout & Periphery [SystemC TLM 2.0]

- Readout network
- Data processing
- Output link



Detector Simulation e.g. AllPix2, Geant4,..., provides pixel hit input data with charge and time of injection

ROC Virtual Prototype in PixESL provides a C++-based high-level simulation platform to study the entire electronics system chain at an early stage of development.

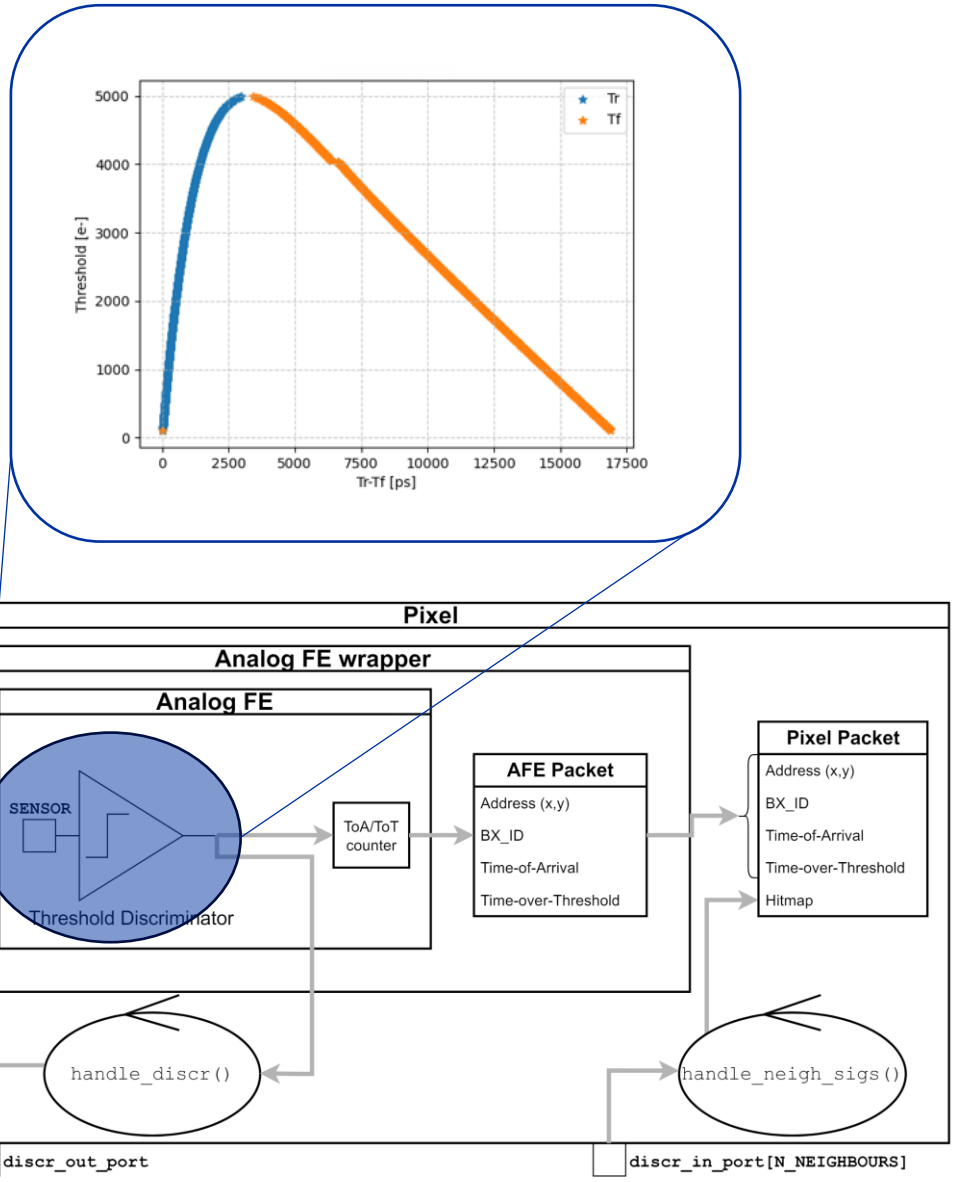
ReadOut Chip output data with performance metrics

PixESL – Front-End model

Modeled with C++ and SystemC specialized classes:

- **Analog FE [C++]:** analytical model of the analog circuits
- **AFE Wrapper [C++ → SystemC]:** translates the AFE outputs into SystemC events
- **Pixel [SystemC]:** formats the pixel packet and routes it into the readout

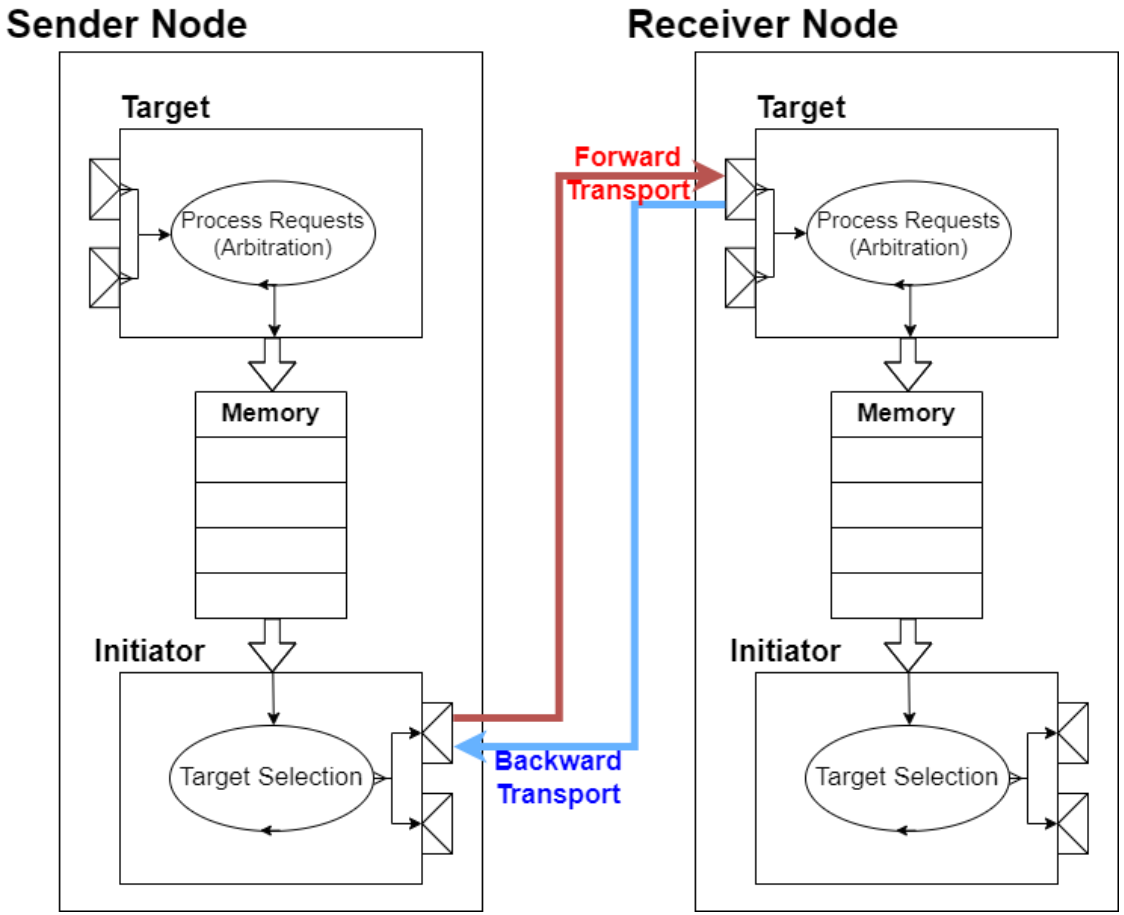
Modular design approach: each module can be easily changed to model different functionality.



PixESL – Readout model

Modeled with **SystemC** Transaction Level Modeling 2.0 (**TLM 2.0**):

- Each node contains**:**
  - **Initiator:** Selects a target and transmits packets to other nodes
  - **Target:** arbitrates incoming packets
  - **Memory**: buffers packets between init. and target
- **Packet transfer protocol** implemented with **TLM 2.0 library**
- Nodes can be **connected arbitrarily** and their **functionality customized**

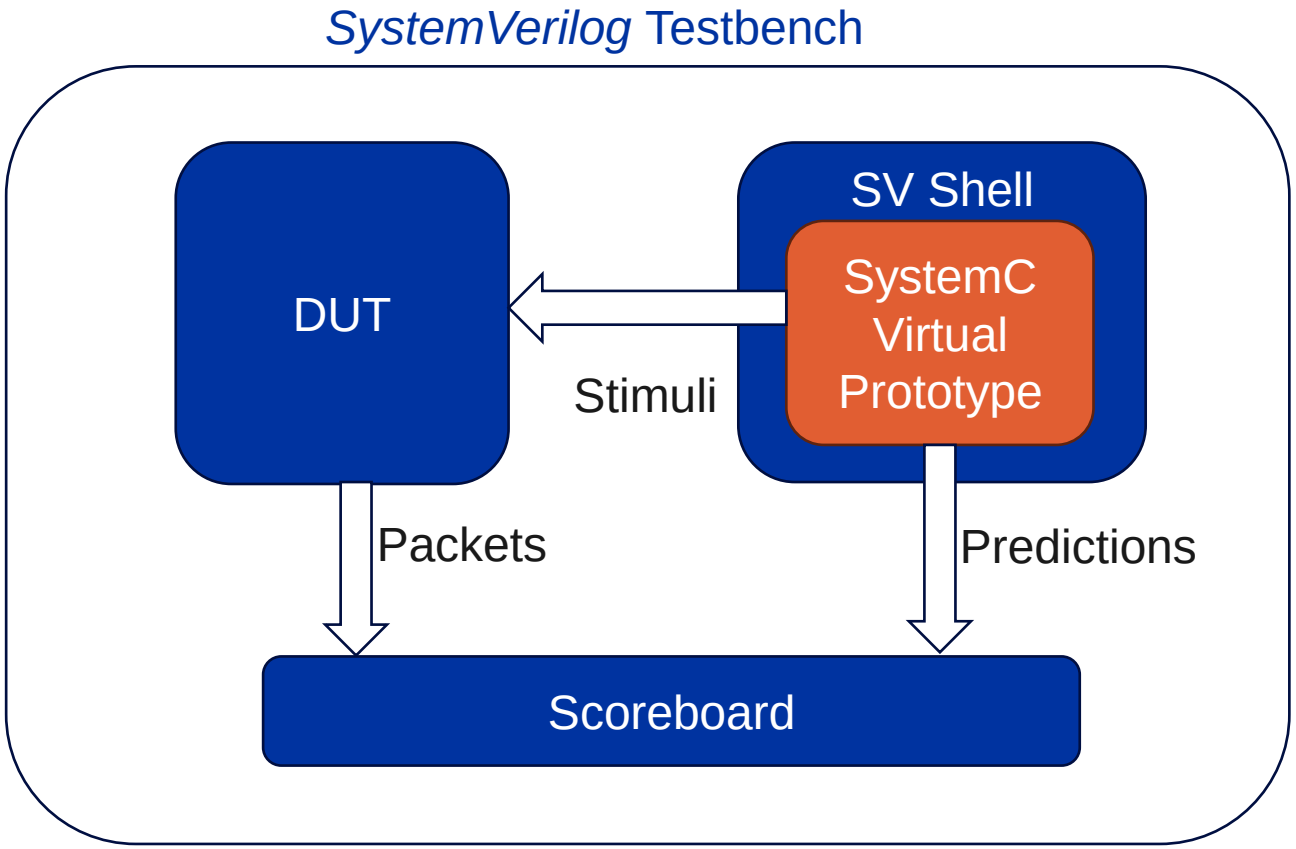


Co-simulation for chip verification

PixESL usage in **testbench** to:

- **Model** the **Analog FE** to **drive** the **DUT**
- **Predict** the **output packets** from pixels

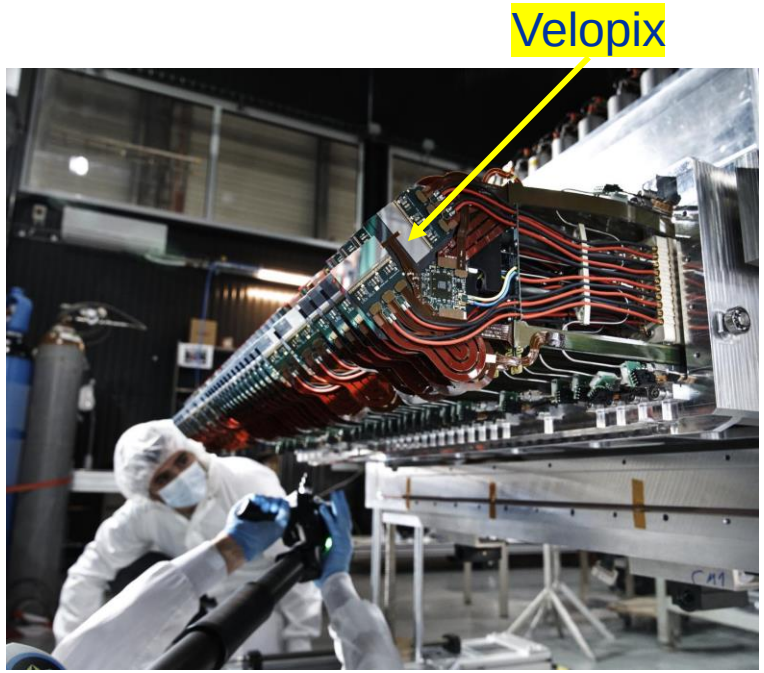
Reuse of AFE and **Pixel models** from Validation phase.



Applications

1. LHCb VeLo upgrade II (data-driven architecture)
2. RD53c for ATLAS and CMS (triggered architecture)

PixESL for LHCb VELO Upgrade II

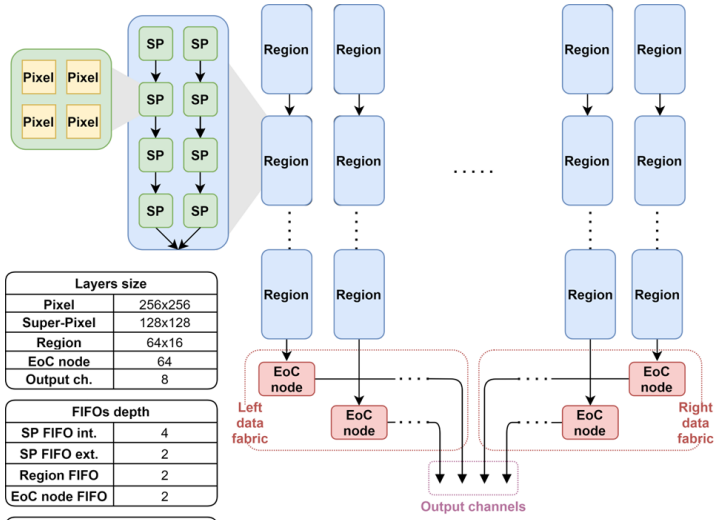


The Readout Chip for the **LHCb tracker upgrade II** is the prototype of **Velopix2**:

- **Data-driven** readout architecture
- Pixel hit rate **3.5 GHz/cm²**
- **28 nm** CMOS technology
- Design derived from **Velopix** and **Timepix4**
- **Time resolution 25 ps**, 50 µm pixel pitch

PixESL:

- The **pixel** emulates the **Analog Front-End model**
- **Super-Pixel**, **region** and **EoC node** are digital readout stages
- The simulation is based on **physical events**
- The **geometry** of the architecture is **user-configurable**



Layers size	
Pixel	256x256
Super-Pixel	128x128
Region	64x16
EoC node	64
Output ch.	8

FIFOs depth	
SP FIFO int.	4
SP FIFO ext.	2
Region FIFO	2
EoC node FIFO	2

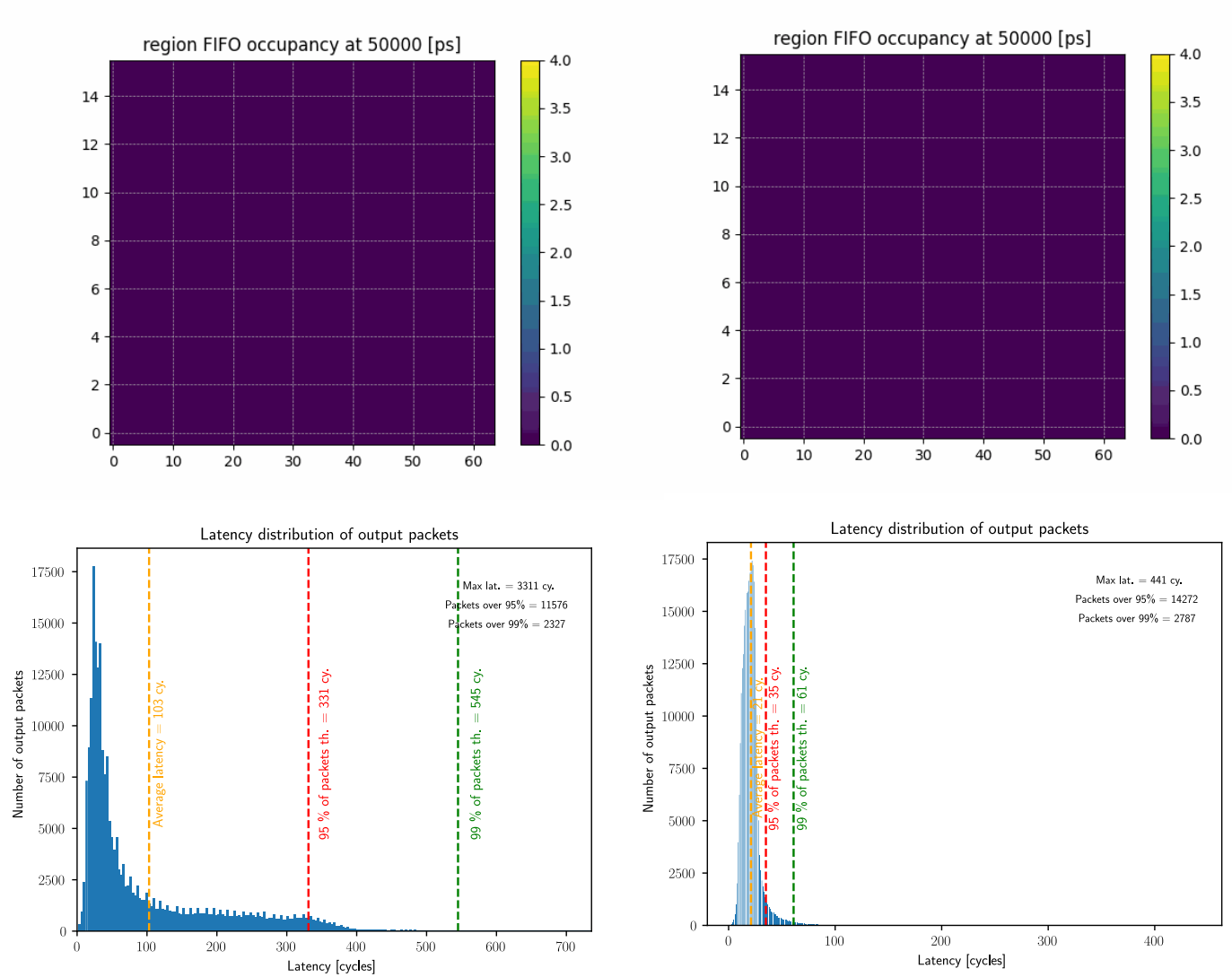
*Clocks	
Matrix clock	x1
EoC node clock	x8

*wrt BX rate

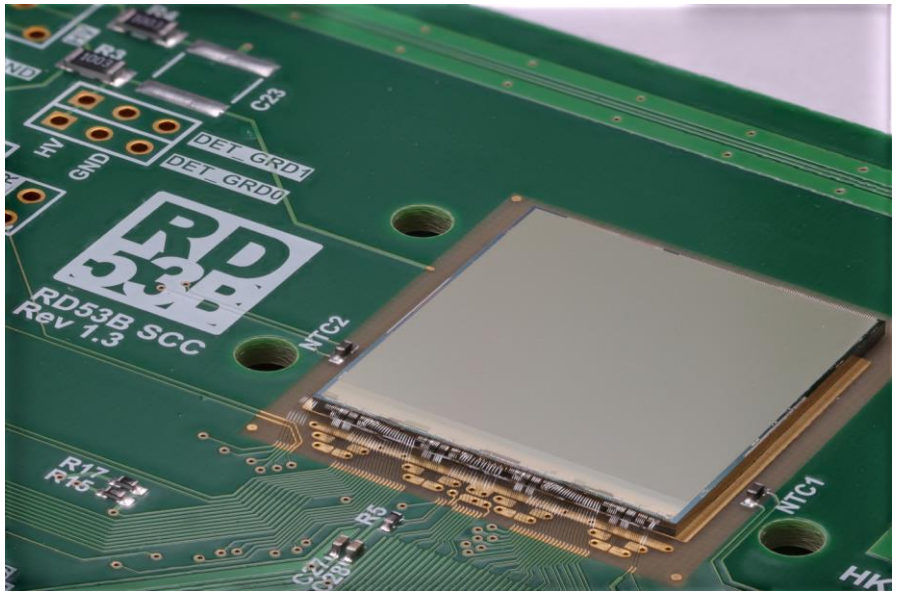
PixESL for LHCb VELO Upgrade II - Results

	Velopix	Proposed
Pixel	256x256	256x256
SP	128x128	128x128
Regions	64x8	128x8 *
EoC	64 (8 ch.)	128 (16 ch.)
Readout eff.	90 %	99 %
Avg. latency	~100 cy.	~20 cy.

*By doubling the transactions per cycle



PixESL for RD53c

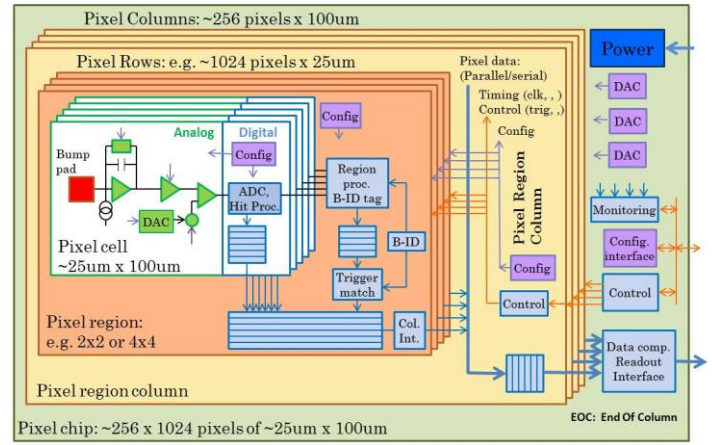


RD53c is the **inner tracker pixel chip** for **CMS** and **ATLAS**:

- **Triggered** readout architecture
- Pixel hit rate **3.5 GHz/cm^2**
- **65 nm** CMOS technology
- **Trigger rate 750 kHz / 1 MHz**, 50 µm pixel pitch

PixESL:

- The **Analog Front-End** model is not included
- **Pixel-regions** and **pixel core columns** are digital readout stages
- The simulation is based on **physical events** (ROOT files)
- The **geometry** of the architecture is **user-configurable**



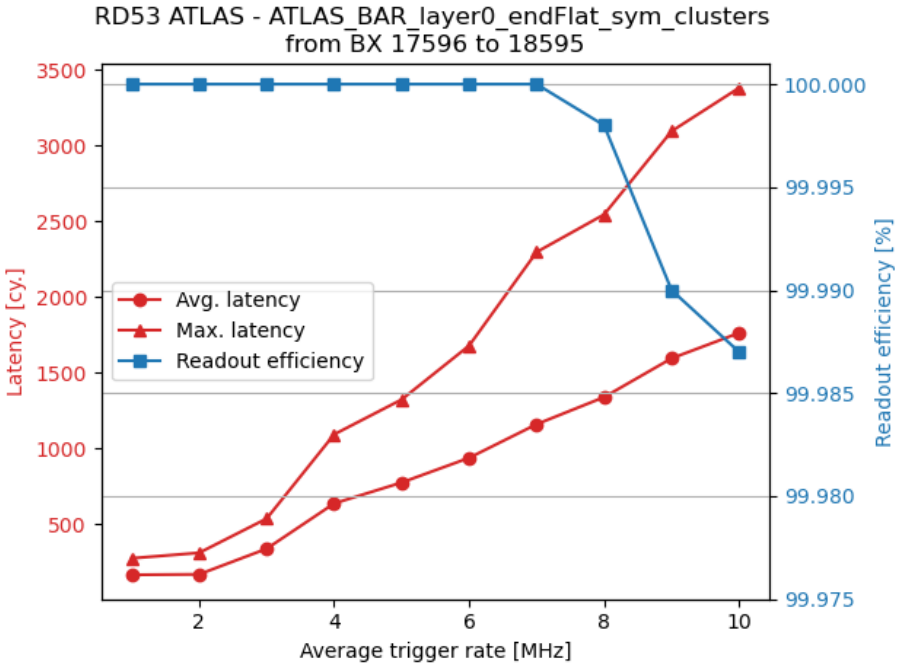
PixESL for RD53c - Results

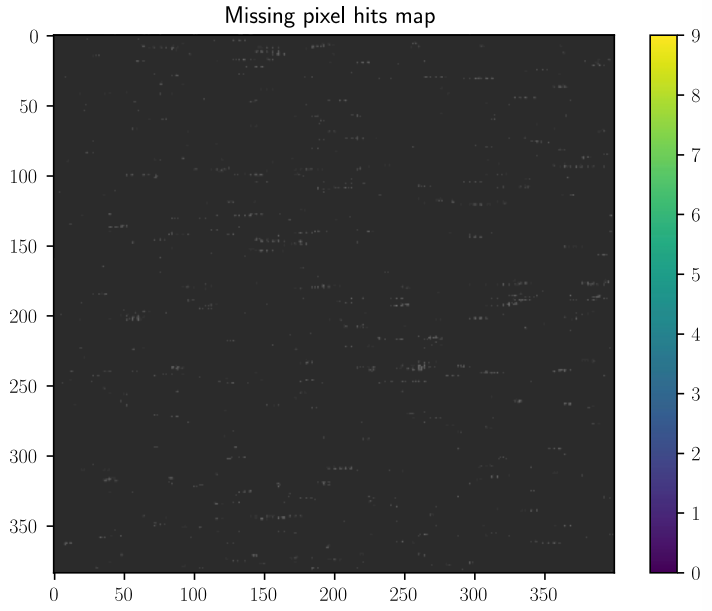
- Injection: 1000 cycles
- Sweep on the **trigger rate**
- Input data: **physics data*** (~92 packets/BX)

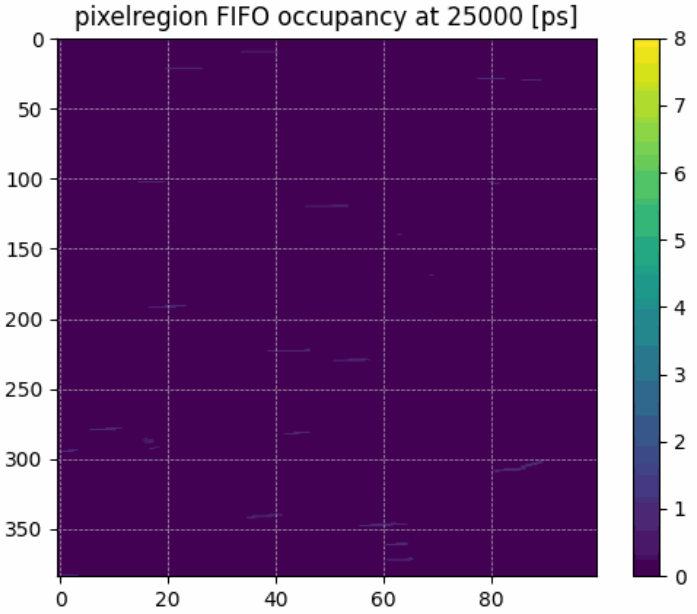
Layers size	
Pixel	400x384
Pixel regions	100x384
Columns	50

Parameters	
Latency buffer	8
Col. read latency	2 cycles
Trigger latency	100 cycles

*Monte Carlo simulation: ATLAS_BAR_layer0_endFlat_sym_clusters from cycle 17596 to 18595 presents the worst-case scenario







Applications summary

LHCb VELO Upgrade II

Data-driven readout architecture

Development of a novel design

Simulation build time: ~ seconds

Simulation runtime: ~ 70k transactions/s

Modelling of the readout: ~ 2 months with 2 FTEs

Modelling of the AFE and Pixel: ~ 1 month with 3 FTEs

UVC interface for SystemVerilog UVM co-simulation

RD53c

Triggered readout architecture

Exploration of an existing design

Simulation build time: ~ seconds

Simulation runtime: ~ 70k transactions/s

Modelling of the readout: ~ 1 month with 1 FTE

Conclusions

- **PixESL** is a valid tool for **high-level modelling** and **architectural analysis**
- The framework presents an **effective and quick prototyping approach**
- This project is **funded by** WP5 EP IC Tech R&D
- The **PixESL** project is part of the **ECFA DRD7 collaboration** ECFA European Committee for Future Accelerators

Team

Project manager – Davide Ceresa (davide.ceresa@cern.ch)
Developer – Jashandeep Dhaliwal (jashandeep.dhaliwal@cern.ch)
Developer – Francesco E. Brambilla (francesco.enrico.brambilla@cern.ch)

Project

Accessible upon request

PixESL GitLab Repository: https://gitlab.cern.ch/iod/pixesl

Thank you!

home.cern