Verification Environment for a Simple Pixel Chip Model

E. Conti[*], P. Placidi[*], S. Marconi[*, +], J. Christiansen[+]

* DIEI – University and INFN Perugia (Italy)
[+] CERN

elia.conti@diei.unipg.it

Goals

- Activity performed in **RD53** (workpackage WG3: "*Simulation Testbench*")

- Requirements

  – Definition and evaluation of requirements for next generation pixel chips

  – System performance evaluation for different pixel architectures

  – Pixel architecture optimization based on pixel grouping (validation of statistical study)
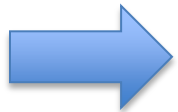
  Devoted simulation and verification framework

# Verification Environment – Current Version

SystemVerilog class-based verification environment

- – Automated hit generation (random and from file)

- – Conformity checks between pixel chip inputs and outputs

- – Monitoring of lost hits

- – Reporting on lost hits (why hits are lost)
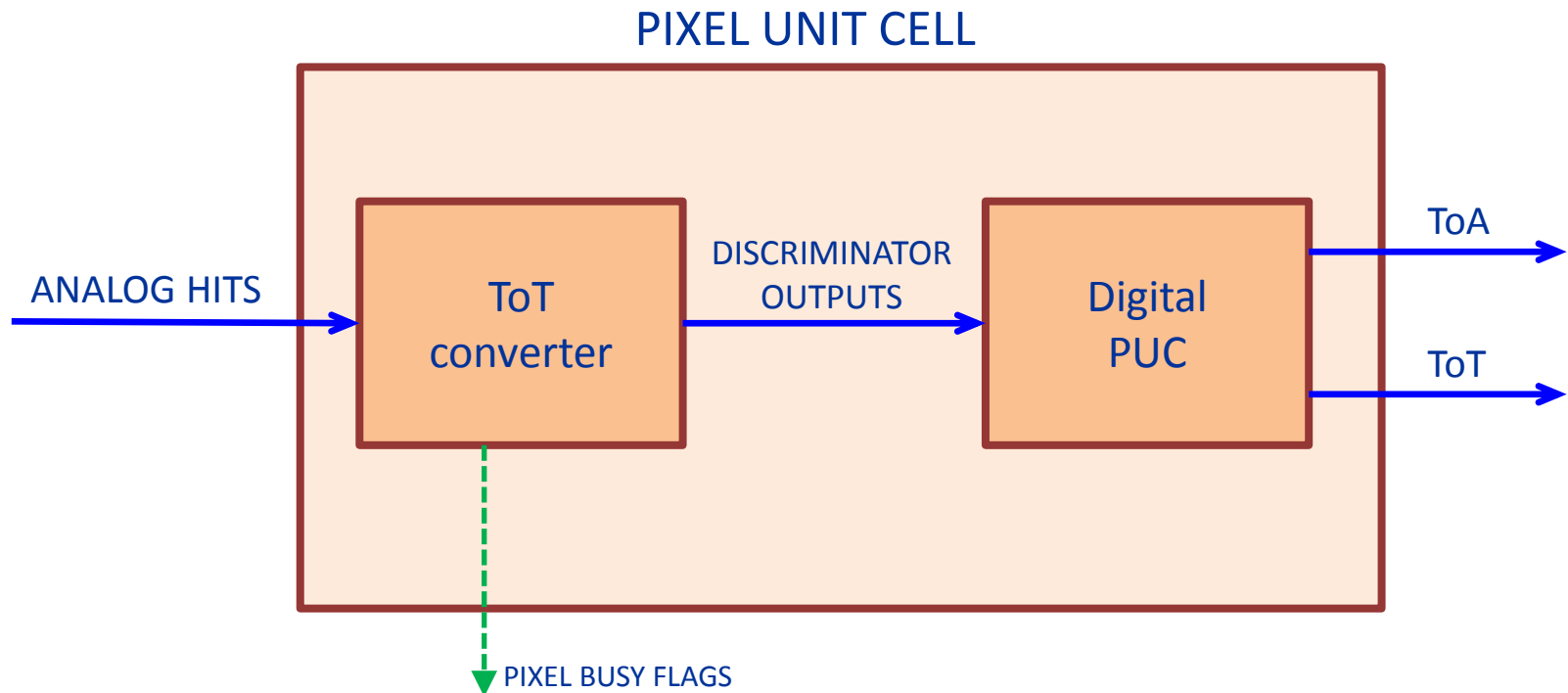
Verification Environment – DUT (1)

Pixel chip contains a single Pixel Region (PR) with a parametrised number of Pixel Unit Cells (PUC)
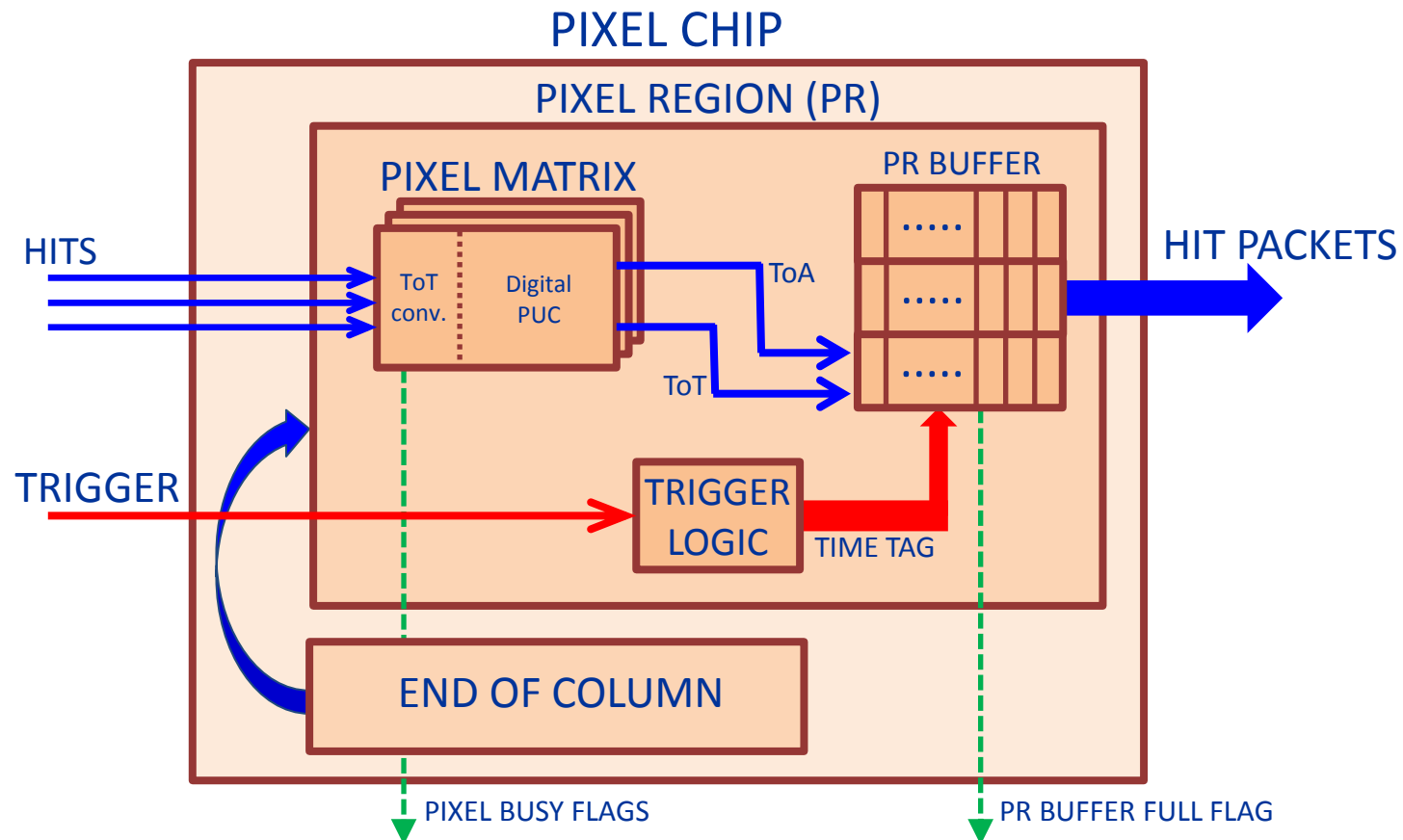
PR buffer is an array of SystemVerilog queues
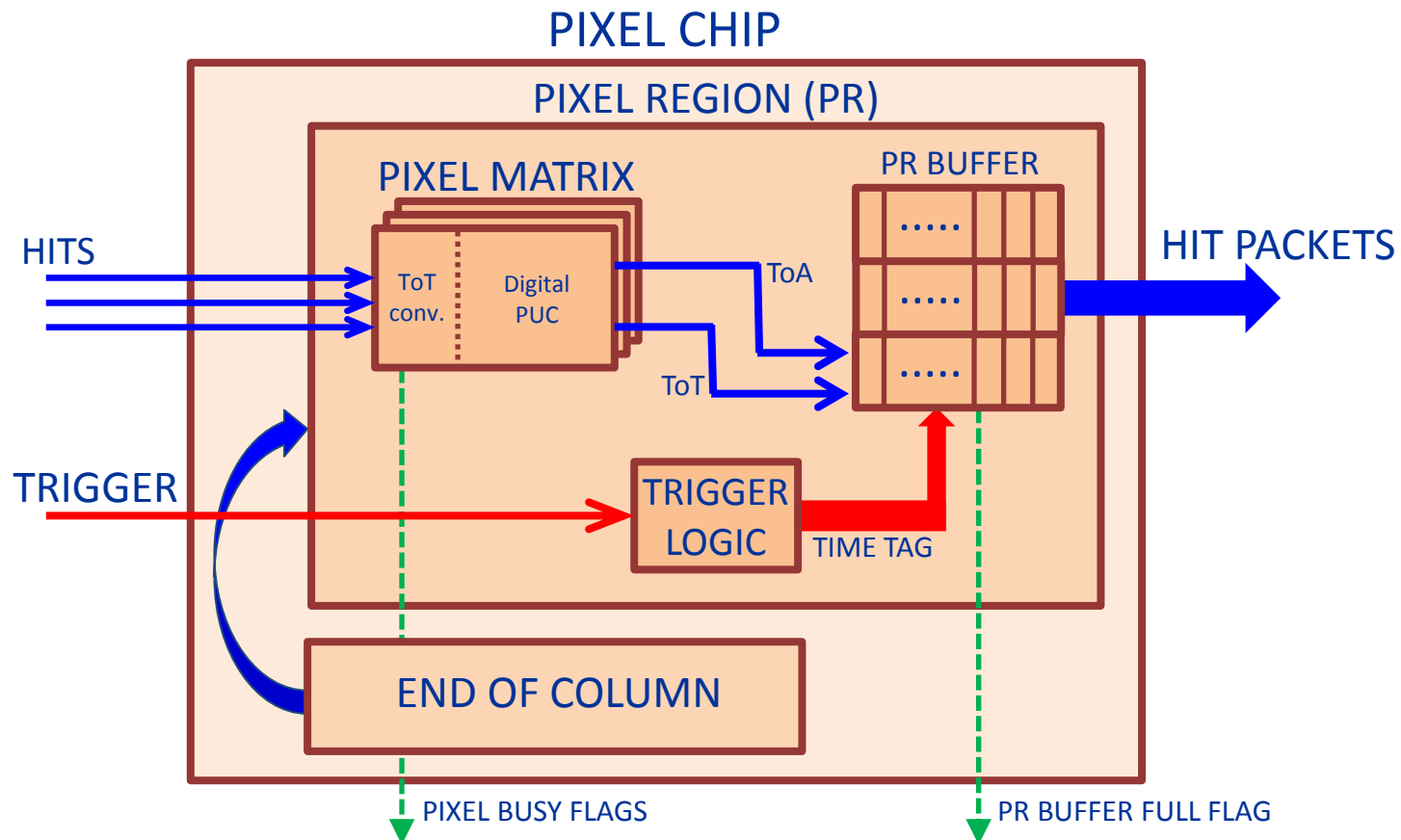
Verification Environment – DUT (2)

In order to keep hit generation as general as possible a simple ToT converter module has been defined which abstracts the behavior of the analog front-end
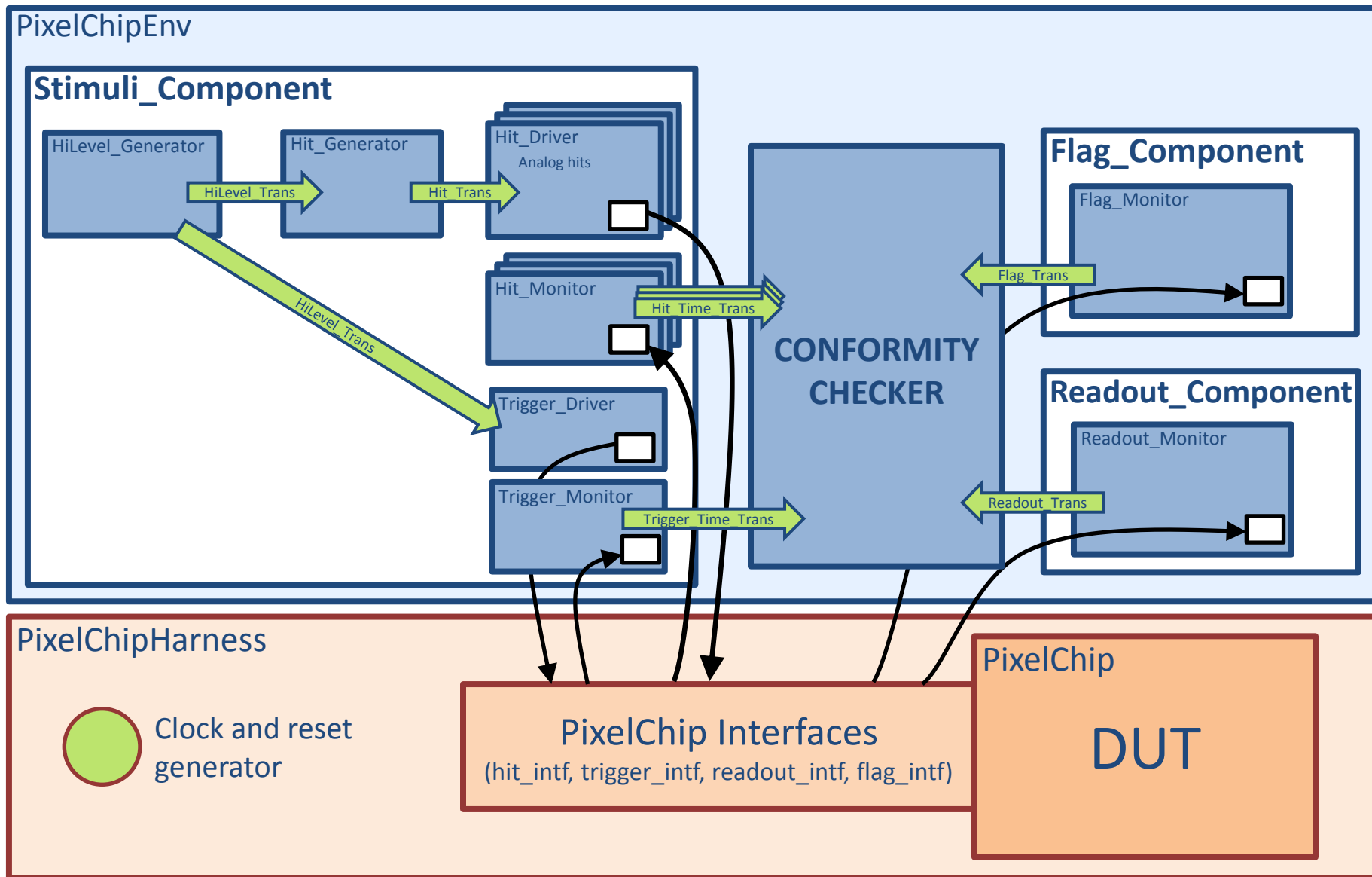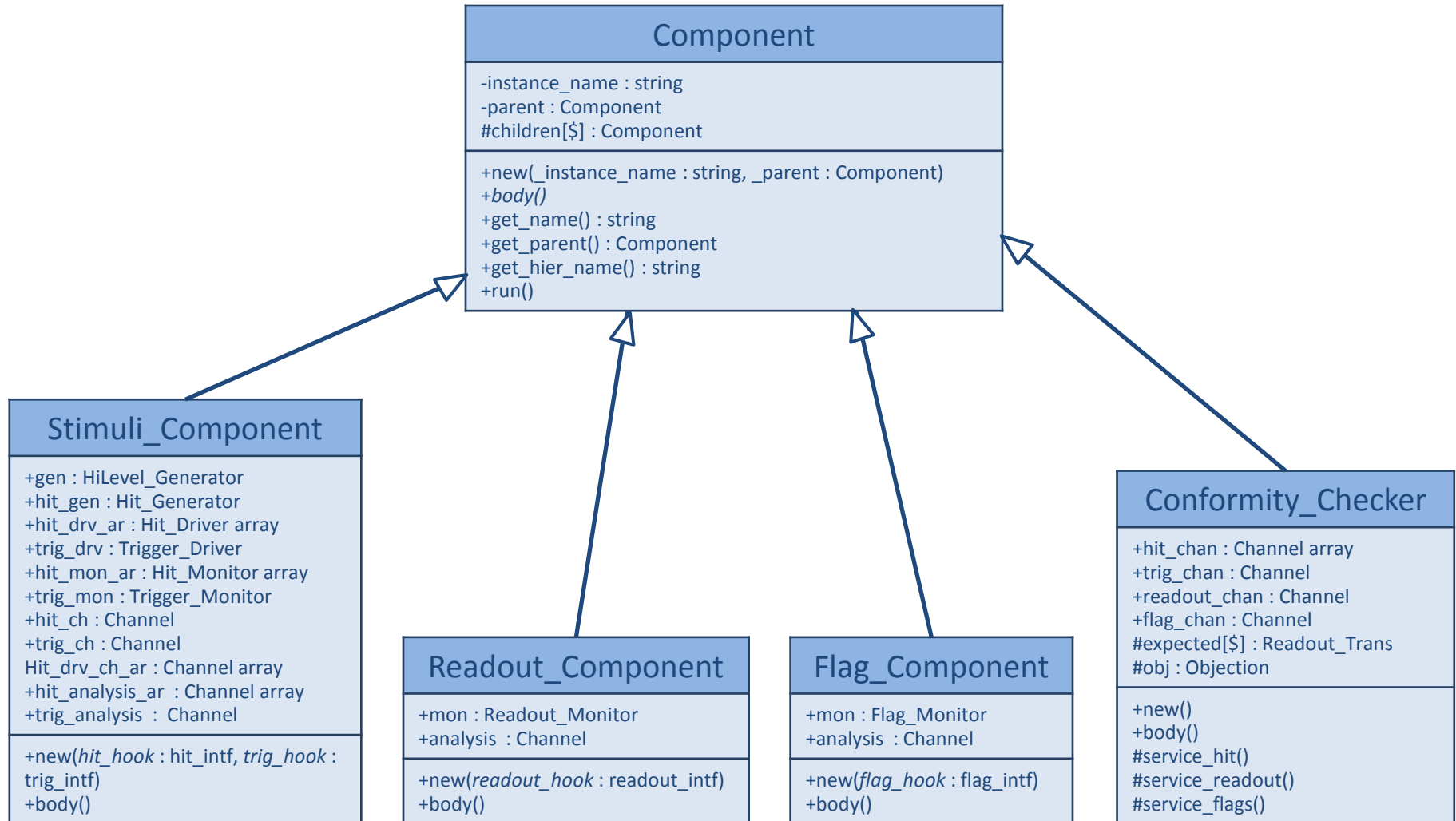
PIXEL UNIT CELL

ANALOG HITS → ToT converter → DISCRIMINATOR OUTPUTS → Digital PUC → ToA

→ ToT

PIXEL BUSY FLAGS

Verification Environment – DUT (3)

Pixel chip contains a single Pixel Region (PR) with a parametrised number of Pixel Unit Cells (PUC)

PR buffer is an array of SystemVerilog queues

Verification Environment – Block Diagram

Verification Environment – Components (1)

Component

-instance_name : string
-parent : Component
#children[$] : Component

+new(_instance_name : string, _parent : Component)
+*body()*
+get_name() : string
+get_parent() : Component
+get_hier_name() : string
+run()

Stimuli_Component

+gen : HiLevel_Generator
+hit_gen : Hit_Generator
+hit_drv_ar : Hit_Driver array
+trig_drv : Trigger_Driver
+hit_mon_ar : Hit_Monitor array
+trig_mon : Trigger_Monitor
+hit_ch : Channel
+trig_ch : Channel
Hit_drv_ch_ar : Channel array
+hit_analysis_ar : Channel array
+trig_analysis : Channel

+new(*hit_hook* : hit_intf, *trig_hook* : trig_intf)
+body()

Readout_Component

+mon : Readout_Monitor
+analysis : Channel

+new(*readout_hook* : readout_intf)
+body()

Flag_Component

+mon : Flag_Monitor
+analysis : Channel

+new(*flag_hook* : flag_intf)
+body()

Conformity_Checker

+hit_chan : Channel array
+trig_chan : Channel
+readout_chan : Channel
+flag_chan : Channel
#expected[$] : Readout_Trans
#obj : Objection

+new()
+body()
#service_hit()
#service_readout()
#service_flags()

Verification Environment – Components (2)

Stimuli_Component

- First stage: generation of high level transactions (randomized or from external file)
- High level transactions are fed to Trigger_Driver (generation of trigger signal) and to Hit_Generator and Hit_Driver's (generation of hits on each pixel of the matrix)

Readout_Component

- Generation of readout transactions using pixel chip outputs

Flag_Component

- Generation of flag transactions using pixel chip flags (pixel busy and PR buffer full)

Conformity_Checker

- Receives input hit information, trigger information, output data and diagnostic information (pixel chip flags)
- Reference model in the checker predicts DUT output from input transactions
- Predicted output and actual output are compared
- Pixel chip flags are always monitored
- Production of report messages

Verification Environment – Reporting (1)

- Conformity_Checker has successfully found conformity between input and output

- WARNING: Conformity_Checker has found discrepancy between input and output

- ERROR   : Conformity_Checker has found out that the PR buffer is full

- Conformity_Checker has found out that a pixel is busy and thus an input hit has been lost

Verification Environment – Reporting (2)

Example of console output

Further Developments

- On the DUT: replication of pixel regions

  - PR columns

  - Future support for L1 trigger and ROI readout

- On the verification environment:

  - Improve monitoring of lost hits

  - Add sources of uncertainty (delays, time walk, …)

  - Standardize into UVM for more solid base classes and highly customizable reporting features