2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
OCTOBER 14-15, 2025

Data-Driven Approach to Accelerate Coverage Closure on Highly Configurable ASIC Designs

Tulio Pereira Bitencourt, Rasadhi Attale, Samuel Man-Shun Wong, Po-Shao Cheng, Anton Tschank

Tessent Embedded Analytics – Siemens

SIEMENS

accellera SYSTEMS INITIATIVE

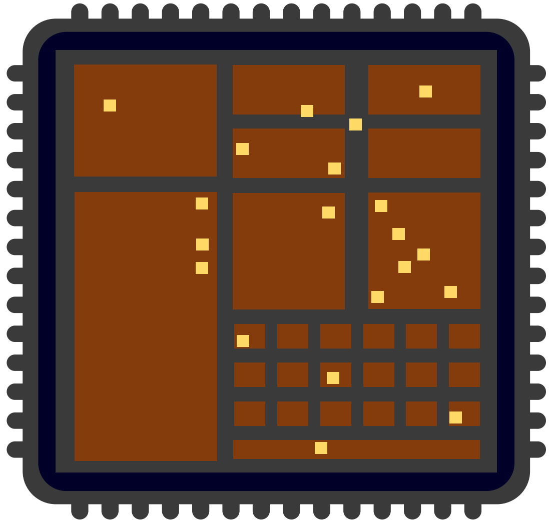
Agenda

Tessent Embedded Analytics functional monitoring

Observing non-intrusively if your SoC behaves as it was meant to

Full visibility into HW/SW interactions in deployed systems enabling optimizations and debugging throughout the entire system lifecycle from lab to deployment

- Real-time debug and trace environment
- Optimize software to achieve better performance and efficiency
- Use historical performance data to inform designs of next generation designs



Bus Monitor	Status Monitor	Trace Encoder	Static Instrumen-tation	CPU Debug Module	NoC Monitor	Direct Memory Access	Trace Receiver

AI   Data Centre   Automotive   5G/6G   Storage   Audio

Introduction – Tessent Embedded Analytics

1

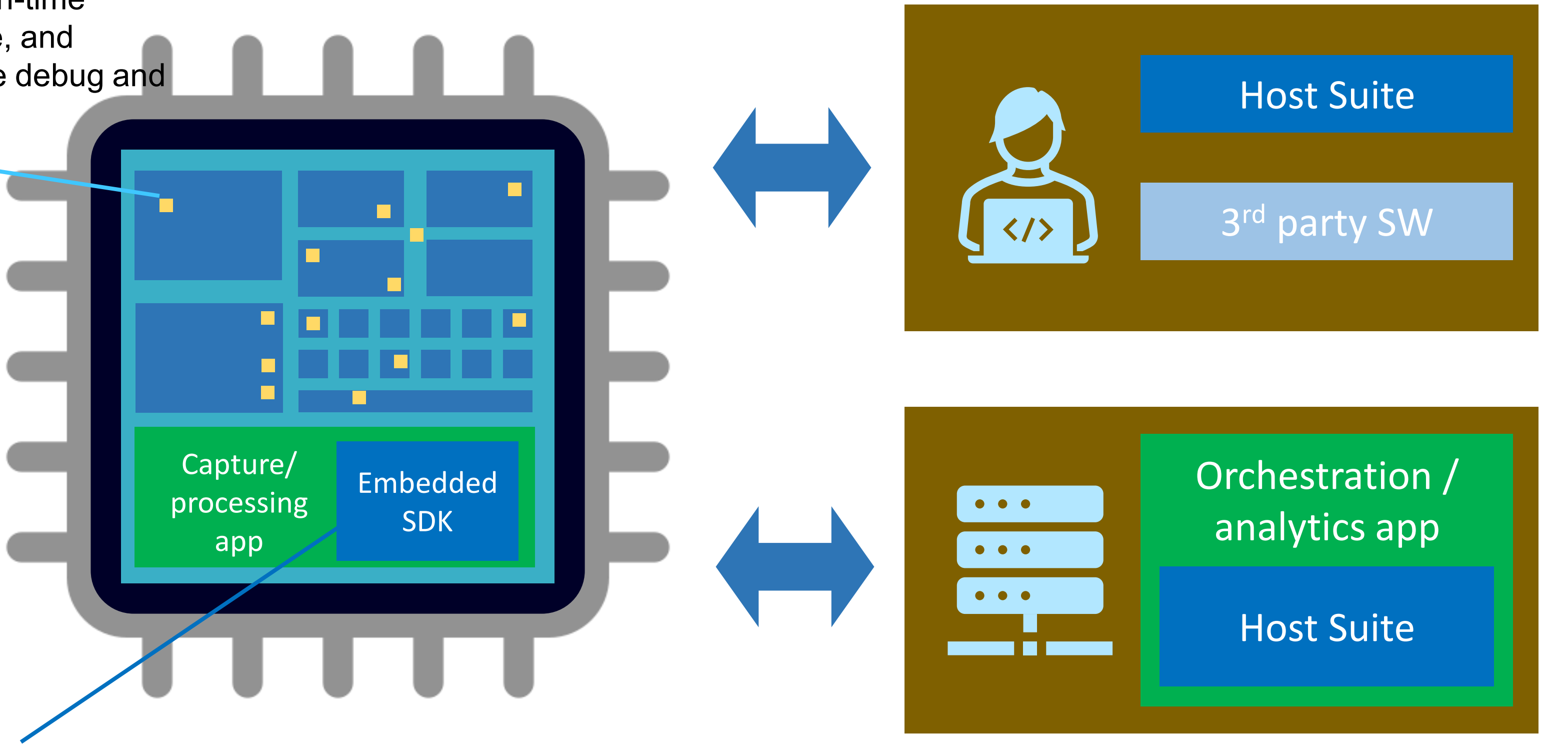
Smart monitors

Range of ~40 IP blocks including run-time configurable monitors, infrastructure, and interfaces that enables non-intrusive debug and performance monitoring

2

Software for interactive debug and optimization

Debug software running on a separate PC is used to interact with the EA smart monitors



3

Edge analytics enablement

Applications developed using the Embedded SDK interact with the monitors, capture, and process results

4

Fleet monitoring enablement

Applications developed using Host Suite can automate data orchestration and analytics from one or multiple devices

Tools & Requirements

• **Requirement tracking tool** (documentation)	(e.g., Polarion)
• **Verification management tool**	(e.g., VIQ)
• **Digital simulation tool**	(e.g., QuestaOne)
• **Regression running tool**	(e.g., VRM)
• **Coverage visualization & merging tool**	(e.g., Visualizer)
• **Regression scheduler & CI/CD**	(e.g., Jenkins)
• **Metrics, data visualization and dashboards**	(e.g., VIQ)

Vendor-agnostic flow!

Challenges of Working with Highly Configurable IPs

Highly-Configurable IPs

"Highly-Configurable"

- Designs with many RTL parameters
- Compile-time constant, large impact on synthesis results
- "CONFIG" = "set of parameters"

Who

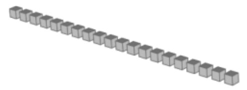
- IP vendors must provide configurable designs
  - Standard protocols, AXI, USB etc.
  - Optional features/optimisation
  - E.g. Tessent Embedded Analytics has extremely high configurability
- Scaling up means increasingly modular designs
  - Accumulates wider range of configurability
  - More inter-dependencies that result in bugs or invalid configurations

Parameter-space Scales Exponentially

`protocolA_channel_width`

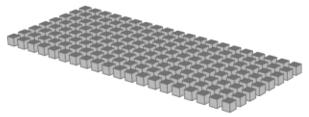
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
```



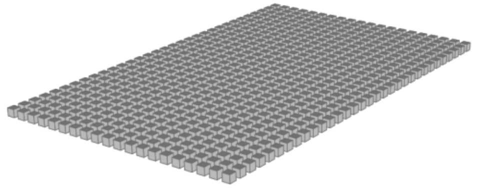
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
```



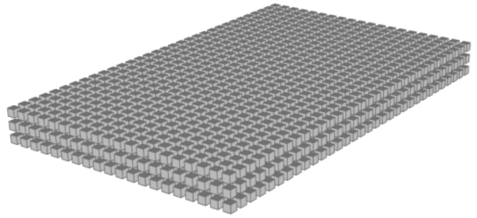
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
```



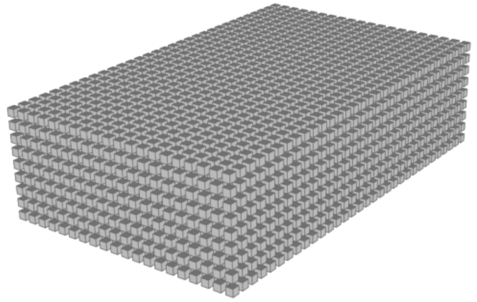
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
```



Parameter-space Scales Exponentially

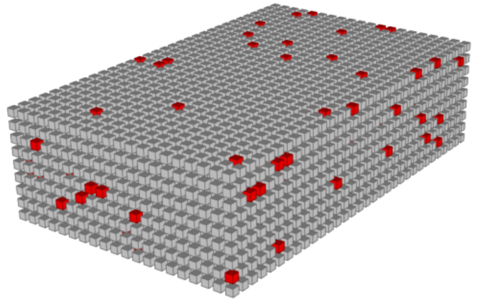
```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2
```



Parameter-space Scales Exponentially

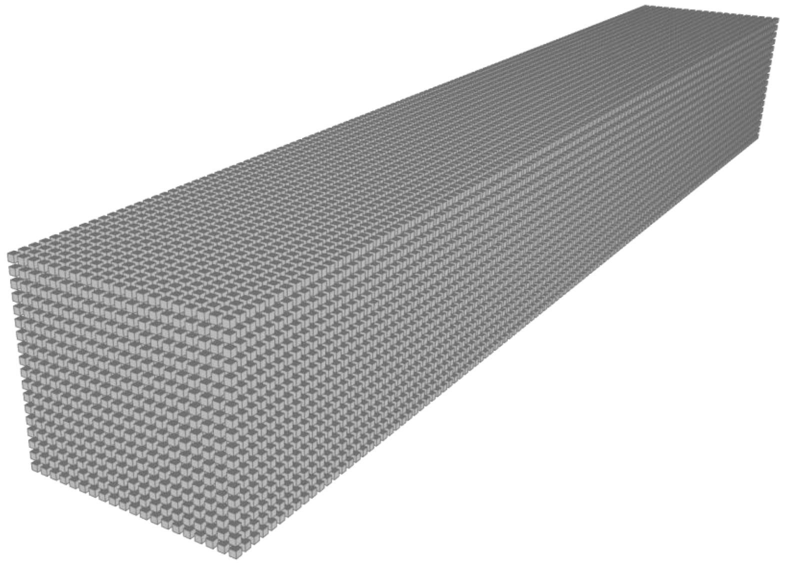
```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2
```

BUGS



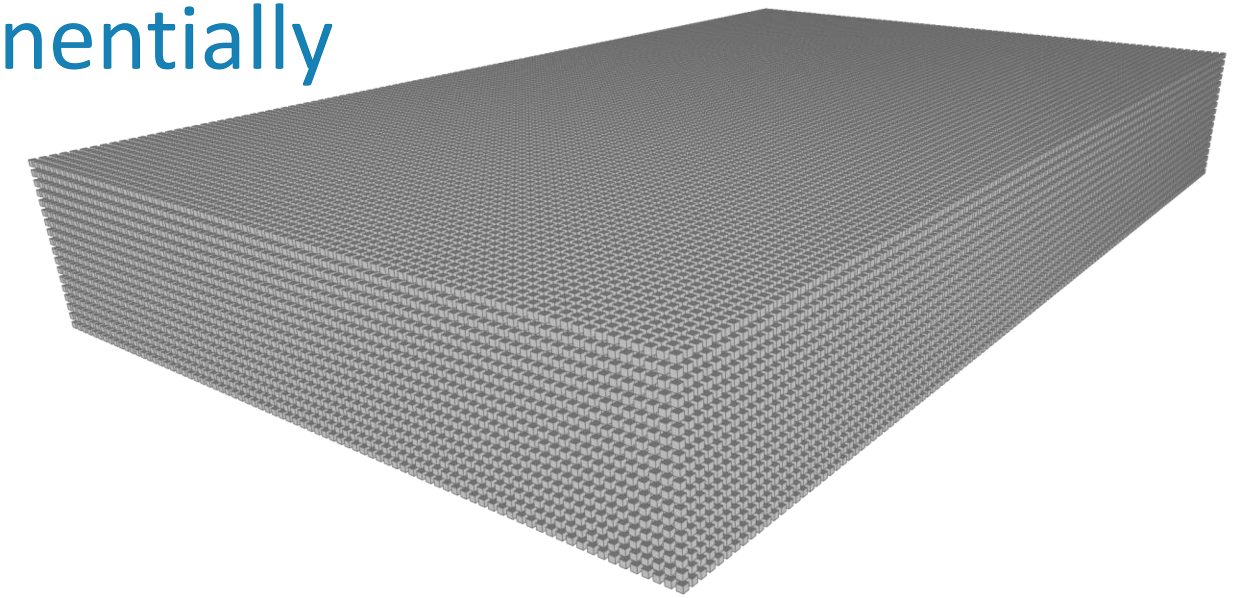
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2

buffer_size
num_buffers
```



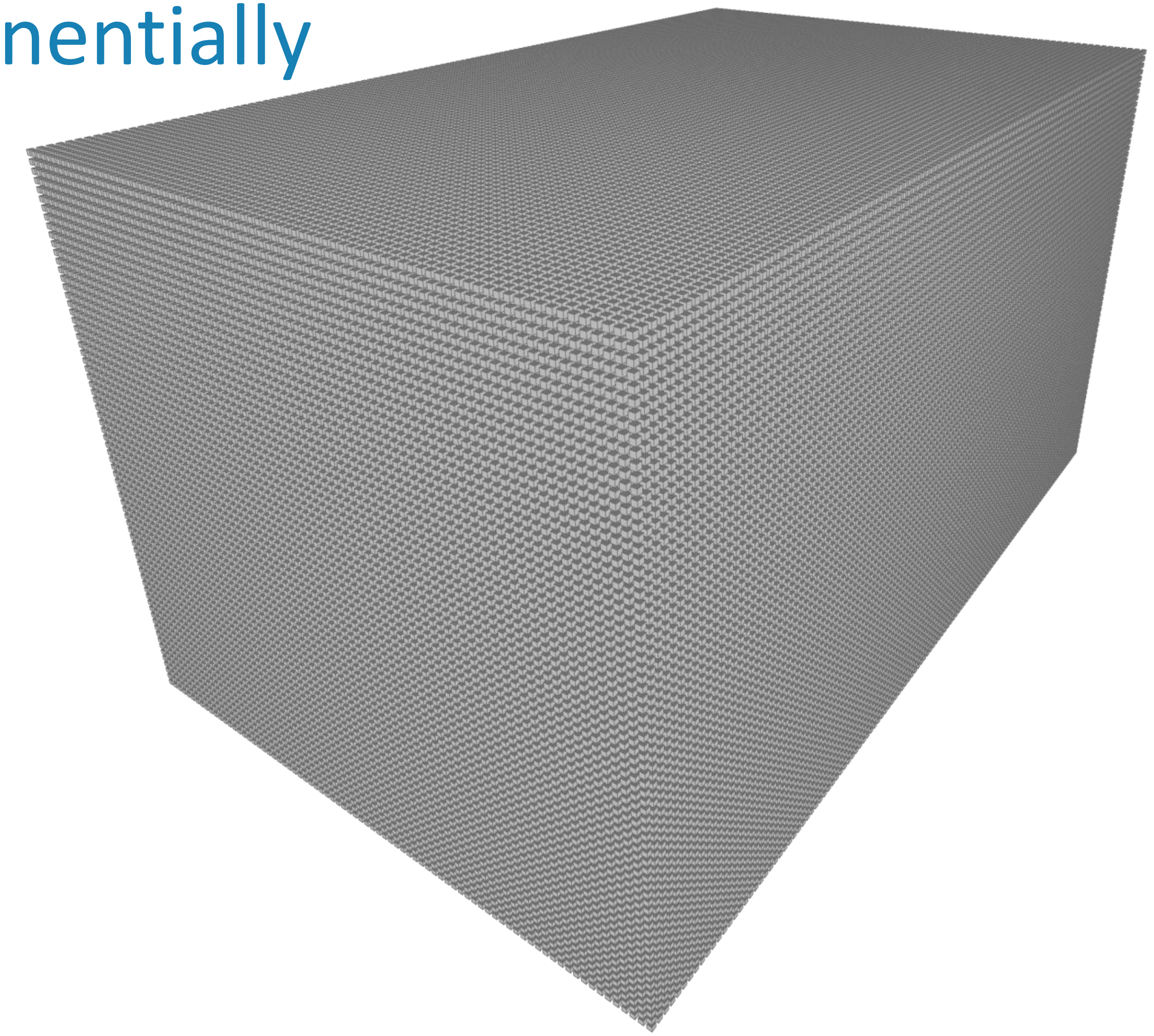
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2

buffer_size
num_buffers
retiming_options
reset_value
```



Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2

buffer_size
num_buffers
retiming_options
reset_value
optional_optimisation
more_feature_support
```

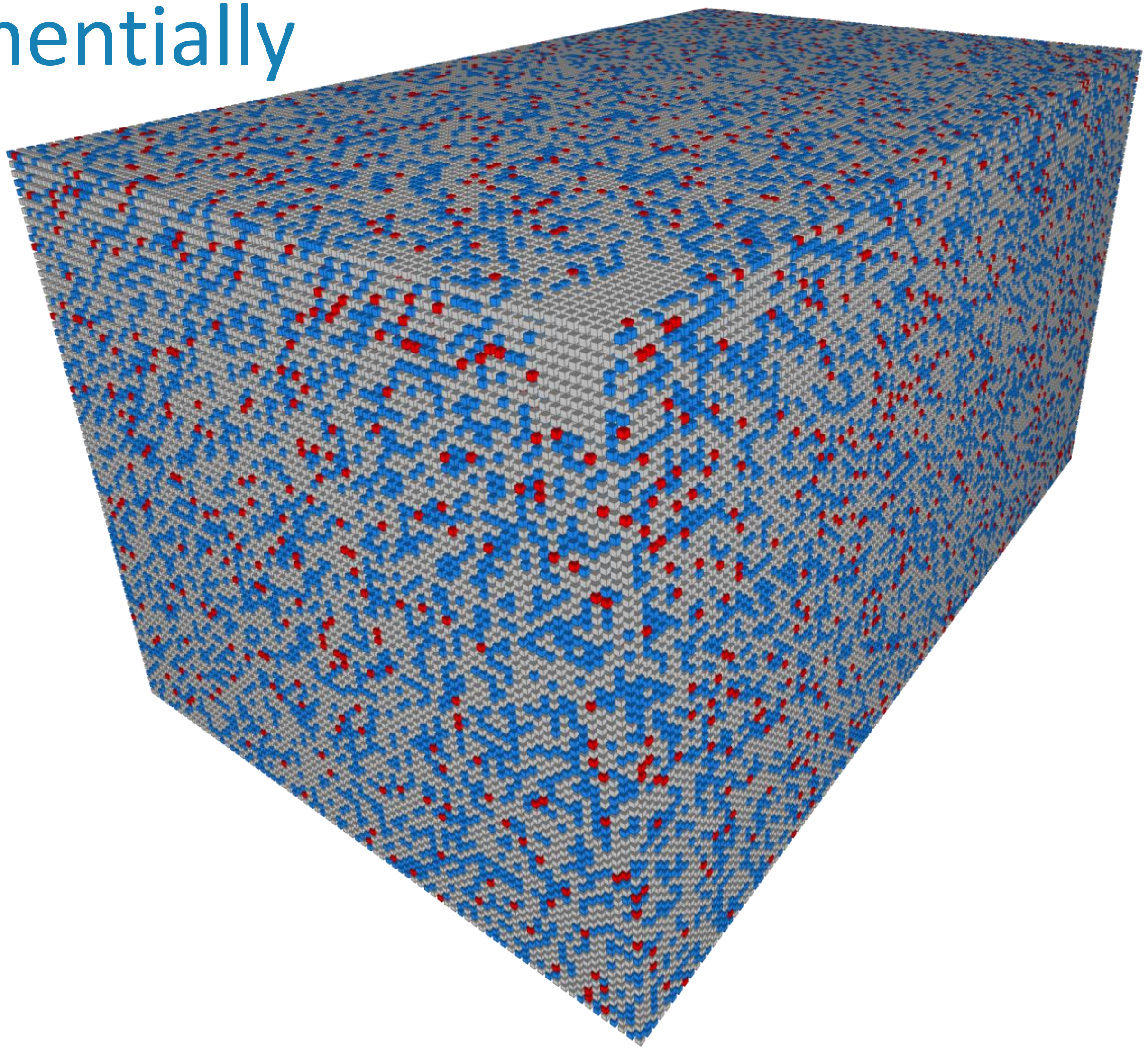


Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2

buffer_size
num_buffers
retiming_options
reset_value
optional_optimisation
more_feature_support
```

BUGS

COVERAGE



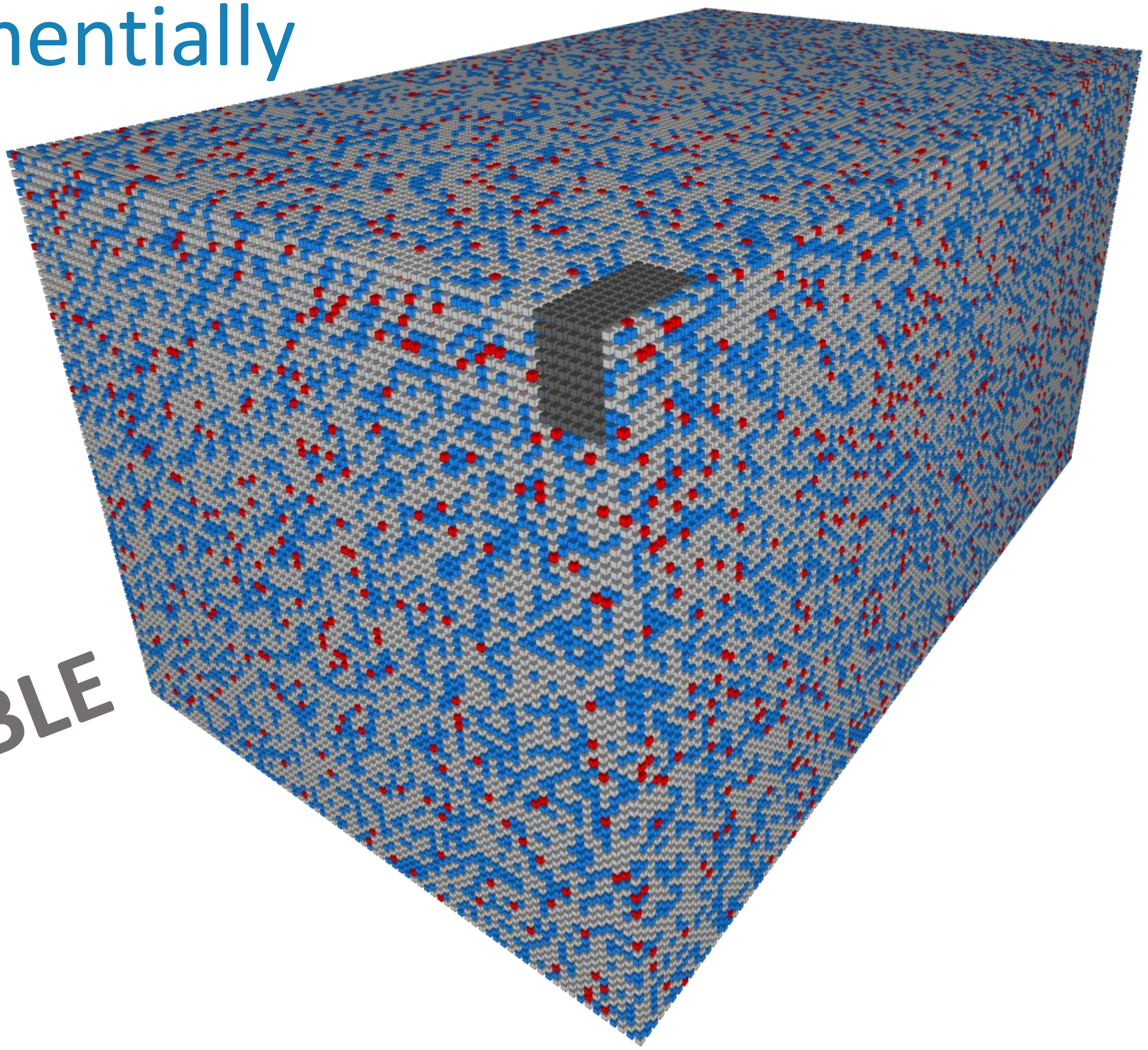
Parameter-space Scales Exponentially

```
protocolA_channel_width
protocolA_num_channels
protocolB_data_width1
protocolB_data_width2
protocolB_option1
protocolB_option2

buffer_size
num_buffers
retiming_options
reset_value
optional_optimisation
more_feature_support
```

BUGS

COVERAGE

UNREACHABLE



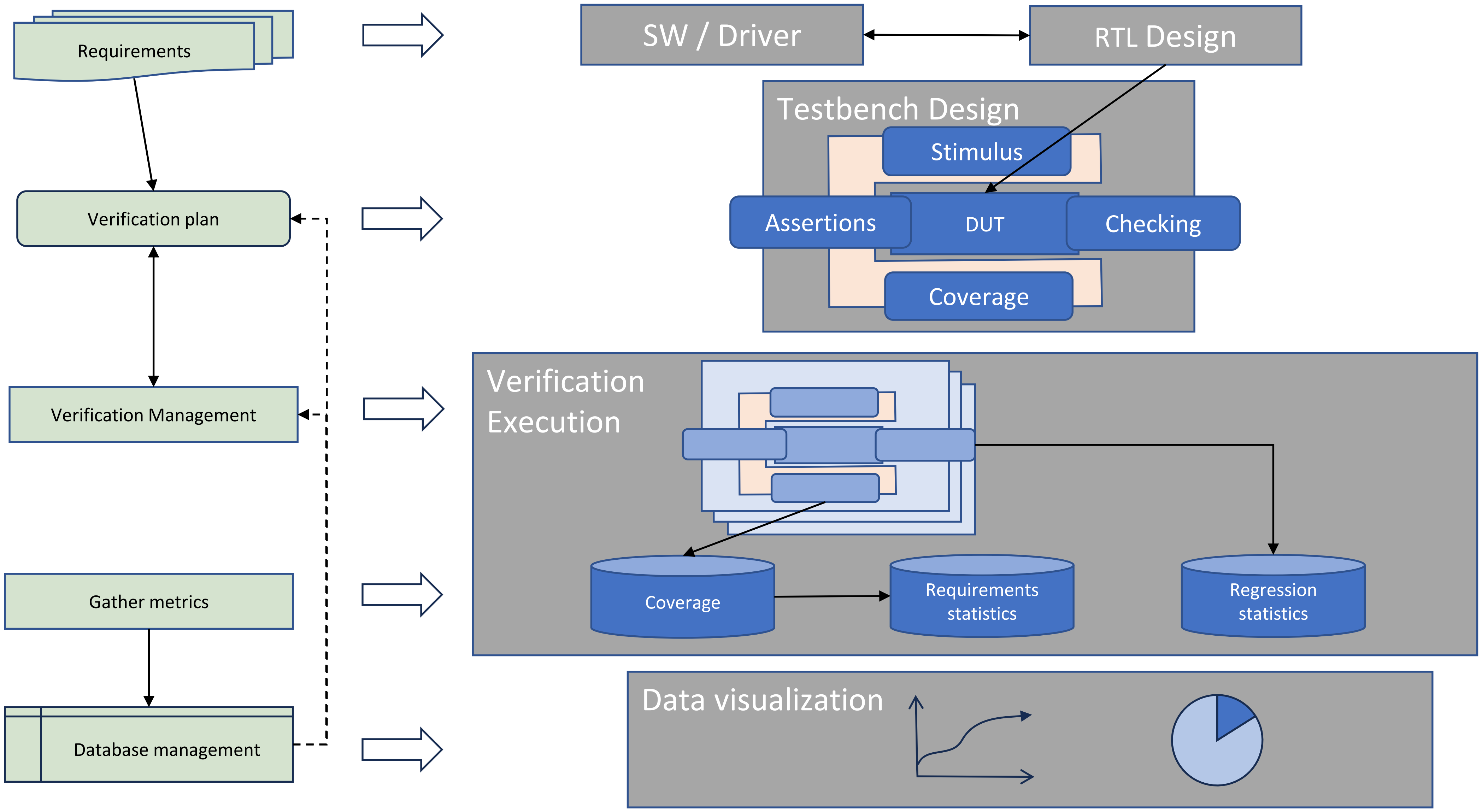
Verification Challenges

Verification complexity scales exponentially with design ...

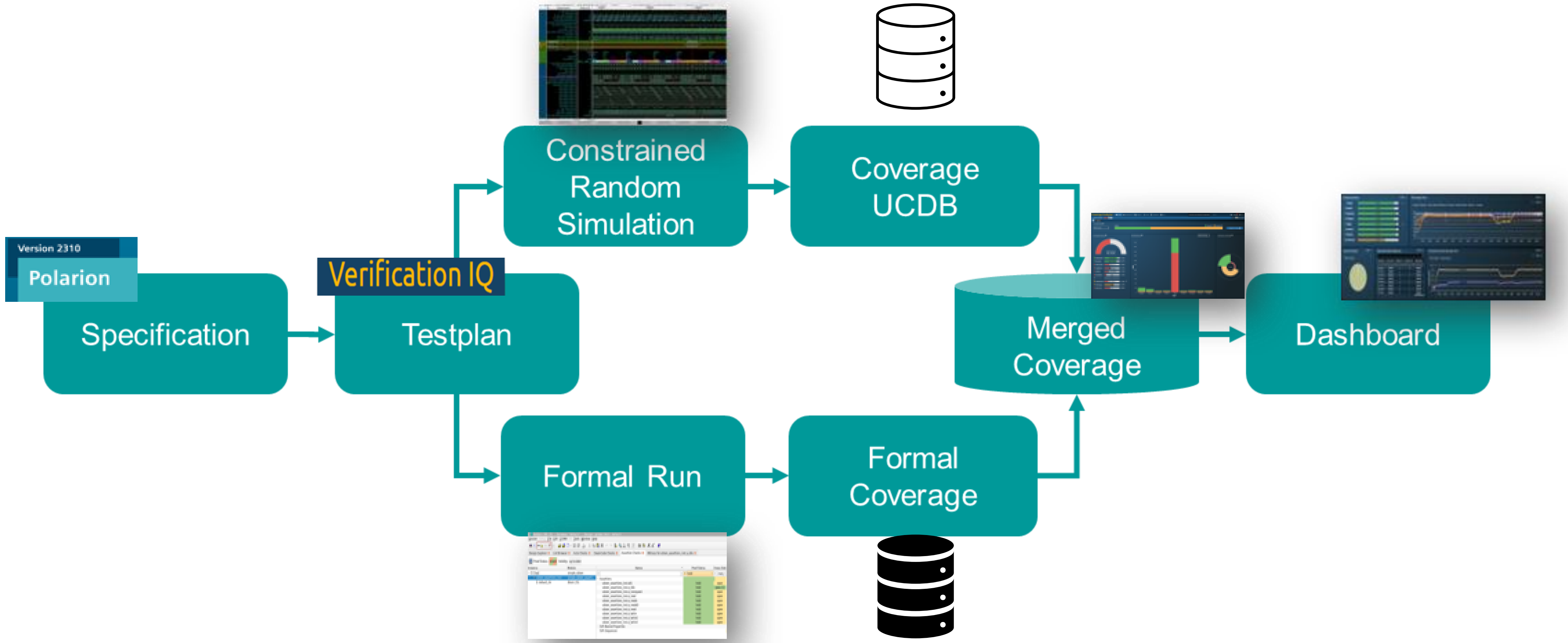
- Many more test scenarios to cover
- Many more bugs to find
- Incomplete insight into verification progress
- Unclear where best to invest resources
- Time-to-market scales drastically

... So a data-driven methodology is needed.

- Trace and track progress of many configurable requirements in documentation
- Managing complex verification plans for both shared and unique features
- Manage compatibility of verification execution with all parameter sets
- Gather metrics to give confidence of high verification quality across all configurable functionality
- Visualizing verification metrics across all configurations for quick feedback


Requirements
Verification plan
Verification Management
Gather metrics
Database management
SW / Driver
RTL Design
Testbench Design
Stimulus
Assertions
DUT
Checking
Coverage
Verification Execution
Coverage
Requirements statistics
Regression statistics
Data visualization

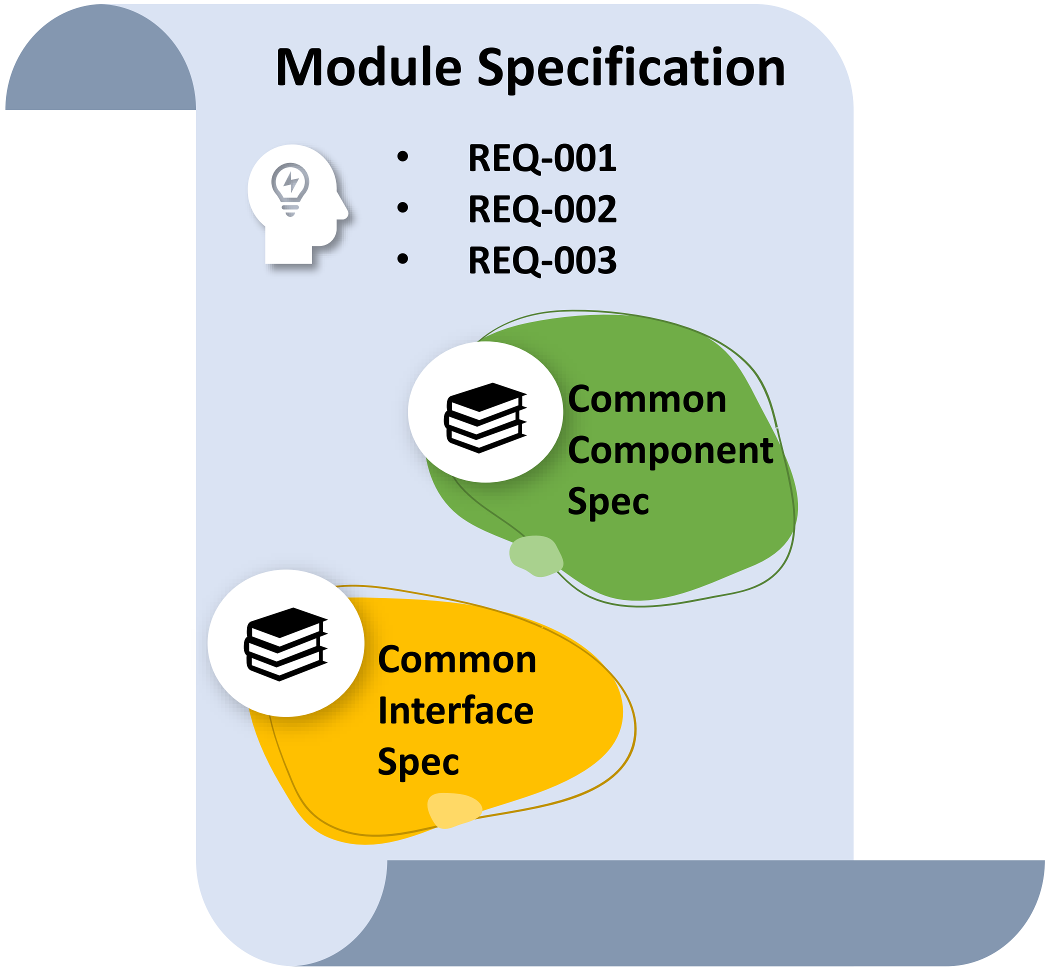
Summary of the proposed flow



Requirement Management and Verification Planning

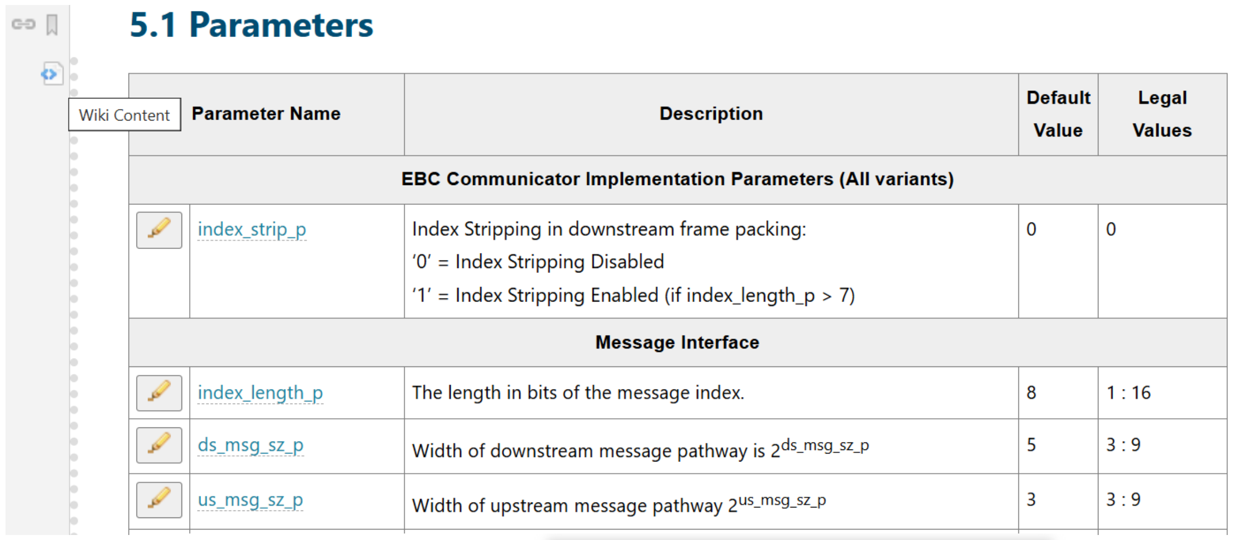
Specification Management

- Itemised requirements for each IP module
- Referenced specification from common components/interface/module
  - Maximise reusability
  - Easier to maintain
  - Only need to be formally reviewed once
- Customised approval flow/work items/scripts, baseline, branching, and much more …



Requirement Traceability

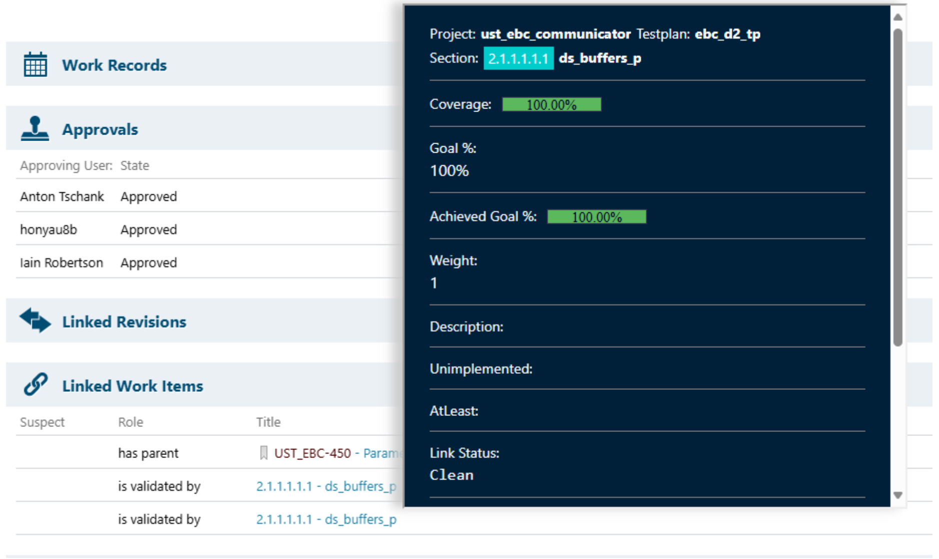
- Ensure every requirement is linked to the test plan
- Traceability is straight forward and easy to read
- Window pop-out showing details of the test plan item
- Crucial aspect of adhering to industry standards e.g. ISO 26262 and DO-254



5.1 Parameters

Wiki Content

Parameter Name	Description	Default Value	Legal Values
EBC Communicator Implementation Parameters (All variants)			
index_strip_p	Index Stripping in downstream frame packing: '0' = Index Stripping Disabled '1' = Index Stripping Enabled (if index_length_p > 7)	0	0
Message Interface			
index_length_p	The length in bits of the message index.	8	1 : 16
ds_msg_sz_p	Width of downstream message pathway is $2^{ds\_msg\_sz\_p}$	5	3 : 9
us_msg_sz_p	Width of upstream message pathway $2^{us\_msg\_sz\_p}$	3	3 : 9



Work Records

Approvals

Approving User:	State
Anton Tschank	Approved
honyau8b	Approved
Iain Robertson	Approved

Linked Revisions

Linked Work Items

Suspect	Role	Title
	has parent	UST_EBC-450 - Param
	is validated by	2.1.1.1.1.1 - ds_buffers_p
	is validated by	2.1.1.1.1.1 - ds_buffers_p

Project: **ust_ebc_communicator** Testplan: **ebc_d2_tp**
Section: 2.1.1.1.1.1 **ds_buffers_p**

Coverage: 100.00%

Goal %:
100%

Achieved Goal %: 100.00%

Weight:
1

Description:

Unimplemented:

AtLeast:

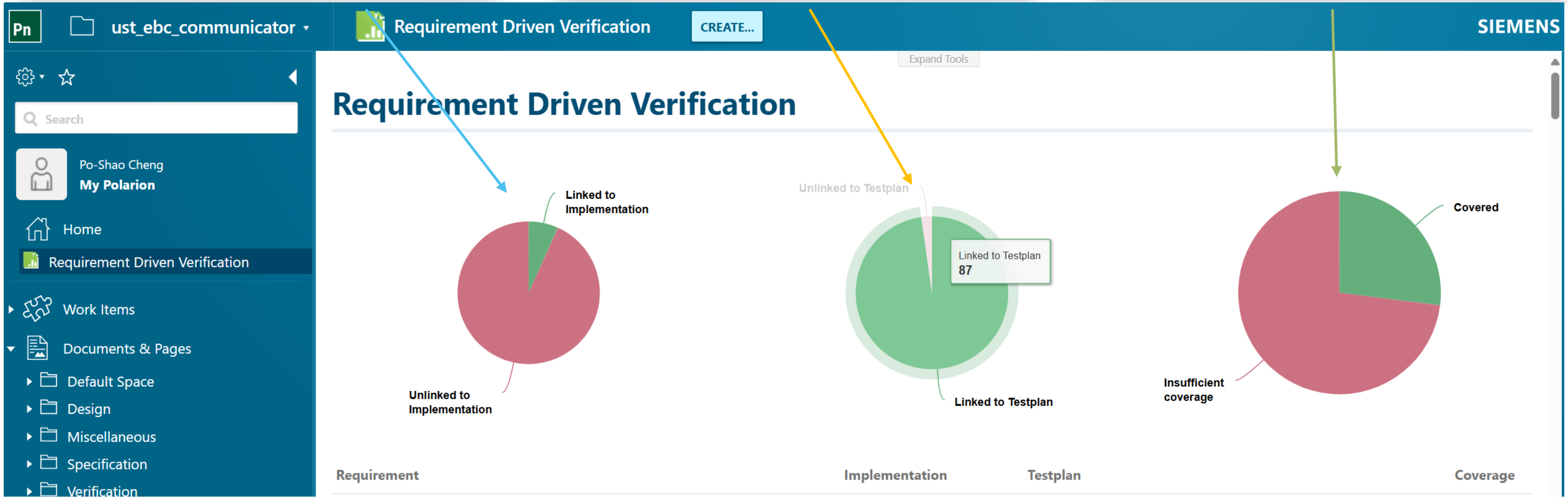
Link Status:
Clean

Requirement Traceability Dashboards

01 RTL Implementation
Requirements linked to the actual RTL implementation e.g. VHDL, Verilog using pragmas.

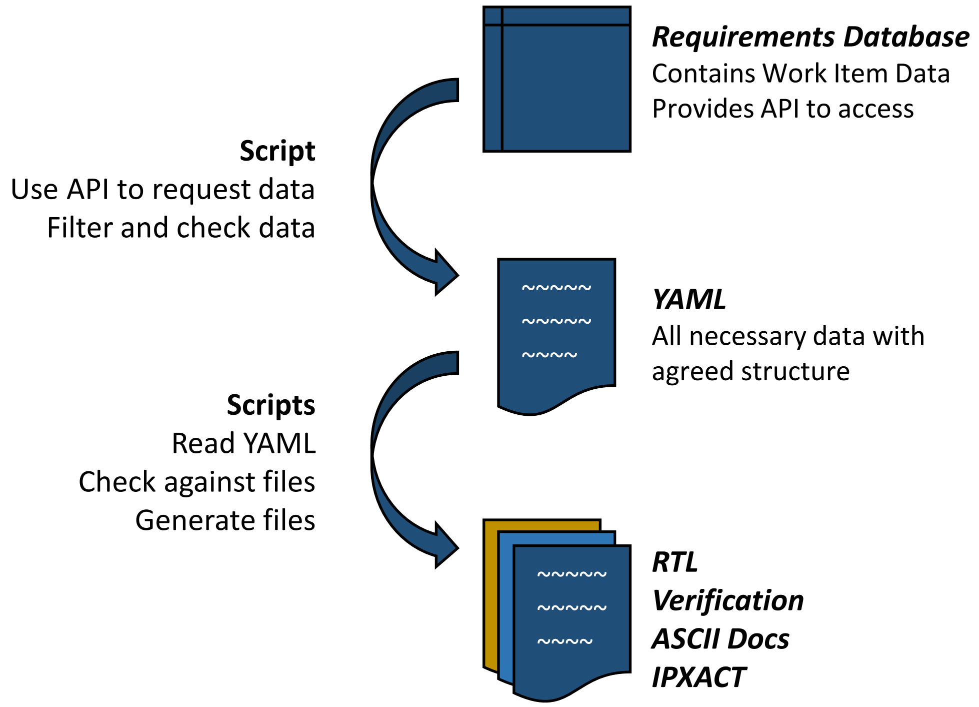
02 Test Plan
Requirements linked to the Test Plan.

03 Coverage Results
Shows coverage number for each test plan items that the requirements are linked to.

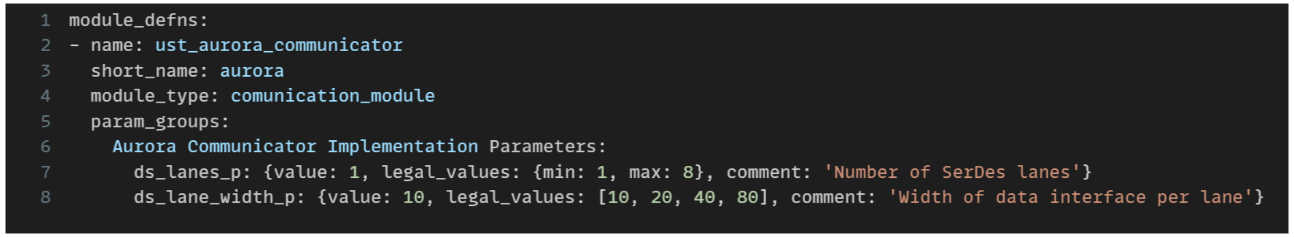


Exporting Data from Specification

- Single source of truth
- API to access its database
- Auto-generated YAML files
- Further automated process
  - Auto-generated RTL sub-components
  - Auto-constraining parameters of the designs



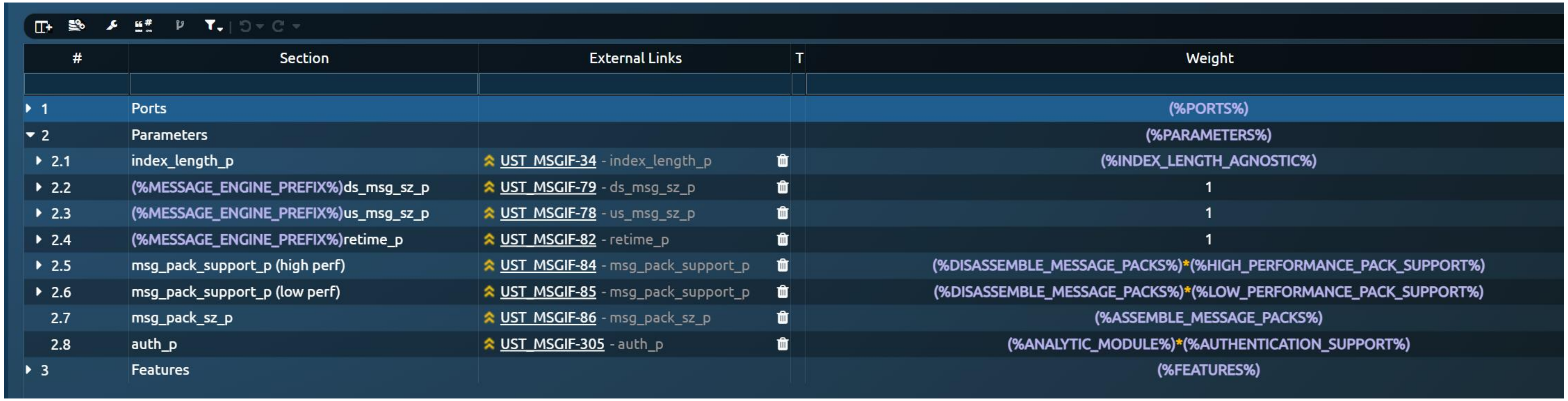
Requirements Database
Contains Work Item Data
Provides API to access
Script
Use API to request data
Filter and check data
~~~~~~
~~~~~~
~~~~
***YAML***
All necessary data with agreed structure
**Scripts**
Read YAML
Check against files
Generate files
***RTL***
***Verification***
***ASCII Docs***
***IPXACT***



```yaml
module_defns:
- name: ust_aurora_communicator
  short_name: aurora
  module_type: comunication_module
  param_groups:
    Aurora Communicator Implementation Parameters:
      ds_lanes_p: {value: 1, legal_values: {min: 1, max: 8}, comment: 'Number of SerDes lanes'}
      ds_lane_width_p: {value: 10, legal_values: [10, 20, 40, 80], comment: 'Width of data interface per lane'}
```
~~~~

VIQ - Testplan Author

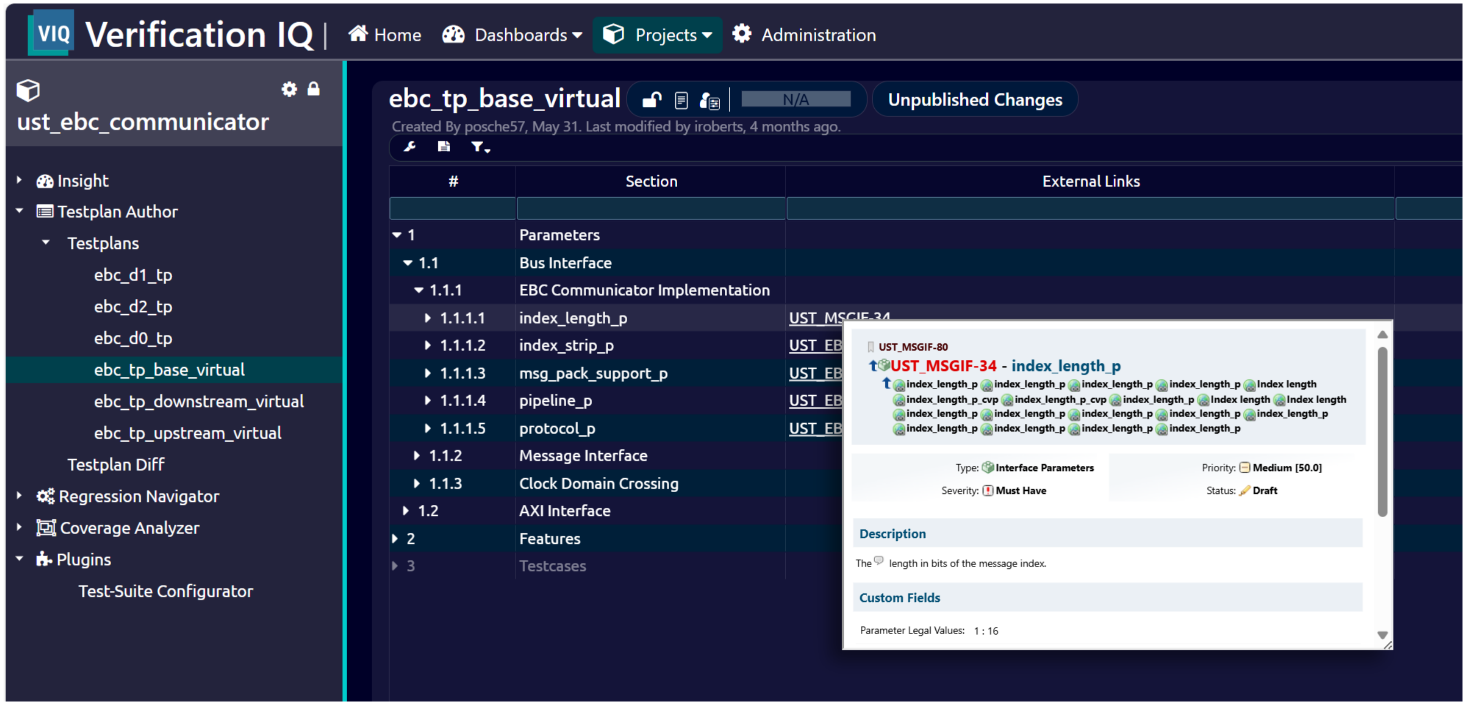
- Parameters used in the test plan – highly configurable reuseable
- Maximise reusability across multiple IP modules for common components, interfaces, protocols
- Strong emphasis on collaboration and effective communication
  - Comment, reuse of other testplans, coverage analyzer looking at UCDBs



#	Section	External Links	T	Weight
1	Ports			(%PORTS%)
2	Parameters			(%PARAMETERS%)
2.1	index_length_p	UST_MSGIF-34 - index_length_p		(%INDEX_LENGTH_AGNOSTIC%)
2.2	(%MESSAGE_ENGINE_PREFIX%)ds_msg_sz_p	UST_MSGIF-79 - ds_msg_sz_p		1
2.3	(%MESSAGE_ENGINE_PREFIX%)us_msg_sz_p	UST_MSGIF-78 - us_msg_sz_p		1
2.4	(%MESSAGE_ENGINE_PREFIX%)retime_p	UST_MSGIF-82 - retime_p		1
2.5	msg_pack_support_p (high perf)	UST_MSGIF-84 - msg_pack_support_p		(%DISASSEMBLE_MESSAGE_PACKS%)*(%HIGH_PERFORMANCE_PACK_SUPPORT%)
2.6	msg_pack_support_p (low perf)	UST_MSGIF-85 - msg_pack_support_p		(%DISASSEMBLE_MESSAGE_PACKS%)*(%LOW_PERFORMANCE_PACK_SUPPORT%)
2.7	msg_pack_sz_p	UST_MSGIF-86 - msg_pack_sz_p		(%ASSEMBLE_MESSAGE_PACKS%)
2.8	auth_p	UST_MSGIF-305 - auth_p		(%ANALYTIC_MODULE%)*(%AUTHENTICATION_SUPPORT%)
3	Features			(%FEATURES%)

Test Plan Traceability

- Shows the requirement linked to the test plan item
- Window pop-out showing details of the requirement



Created By posche57, May 31. Last modified by iroberts, 4 months ago.

#	Section	External Links
1	Parameters	
1.1	Bus Interface	
1.1.1	EBC Communicator Implementation	
1.1.1.1	index_length_p	UST_MSGIF-34
1.1.1.2	index_strip_p	UST_EB
1.1.1.3	msg_pack_support_p	UST_EB
1.1.1.4	pipeline_p	UST_EB
1.1.1.5	protocol_p	UST_EB
1.1.2	Message Interface	
1.1.3	Clock Domain Crossing	
1.2	AXI Interface	
2	Features	
3	Testcases	

UST_MSGIF-80

UST_MSGIF-34 - index_length_p

index_length_p index_length_p index_length_p index_length_p Index length index_length_p_cvp index_length_p_cvp index_length_p Index length Index length index_length_p index_length_p index_length_p index_length_p index_length_p index_length_p index_length_p index_length_p index_length_p

Type: Interface Parameters
Priority: Medium [50.0]
Severity: Must Have
Status: Draft

Description

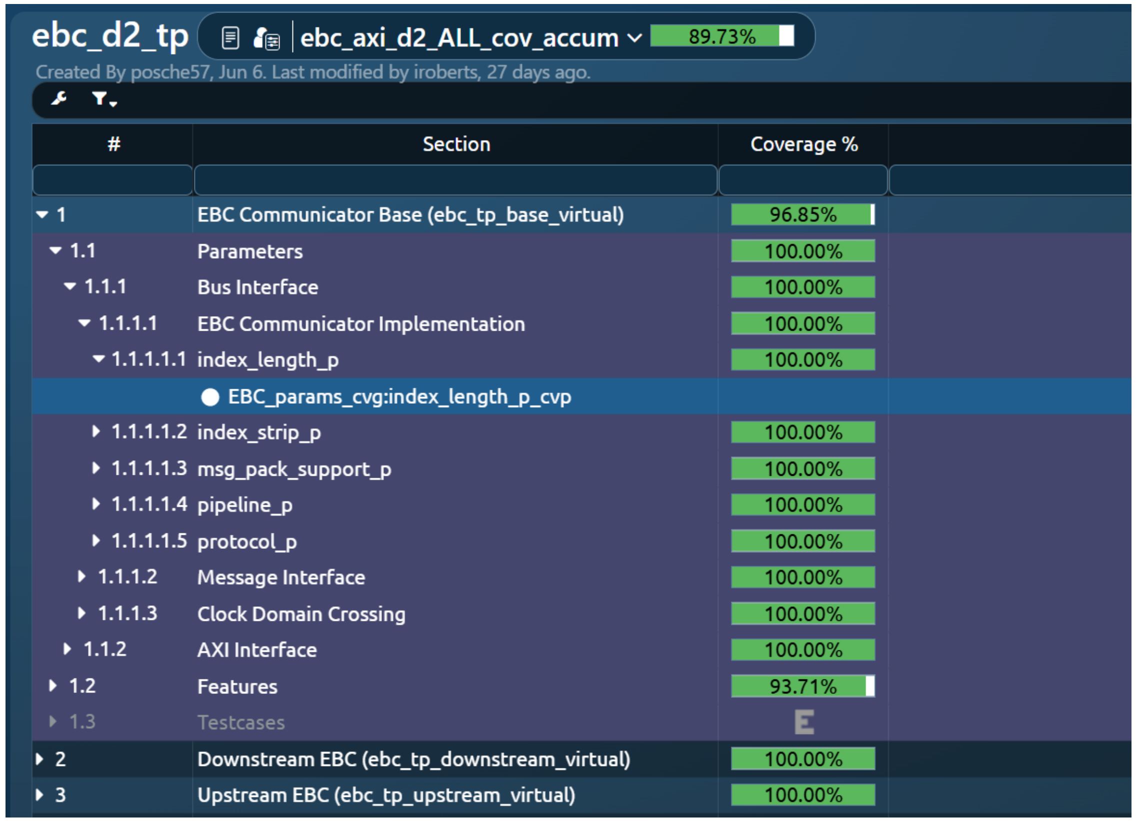
The length in bits of the message index.

Custom Fields

Parameter Legal Values: 1 : 16

Coverage Tracking in Test Plan

- Coverage Database linked to the test plan
- Functional coverage
  - Test plan is linked to covergroup and/or coverpoints
- Code coverage
  - Ports in test plan are linked to toggle coverage
- Test plan coverage
  - Shows overall progress of the test plan
  - How well the test plan is executed
- Exclusion option available in test plan



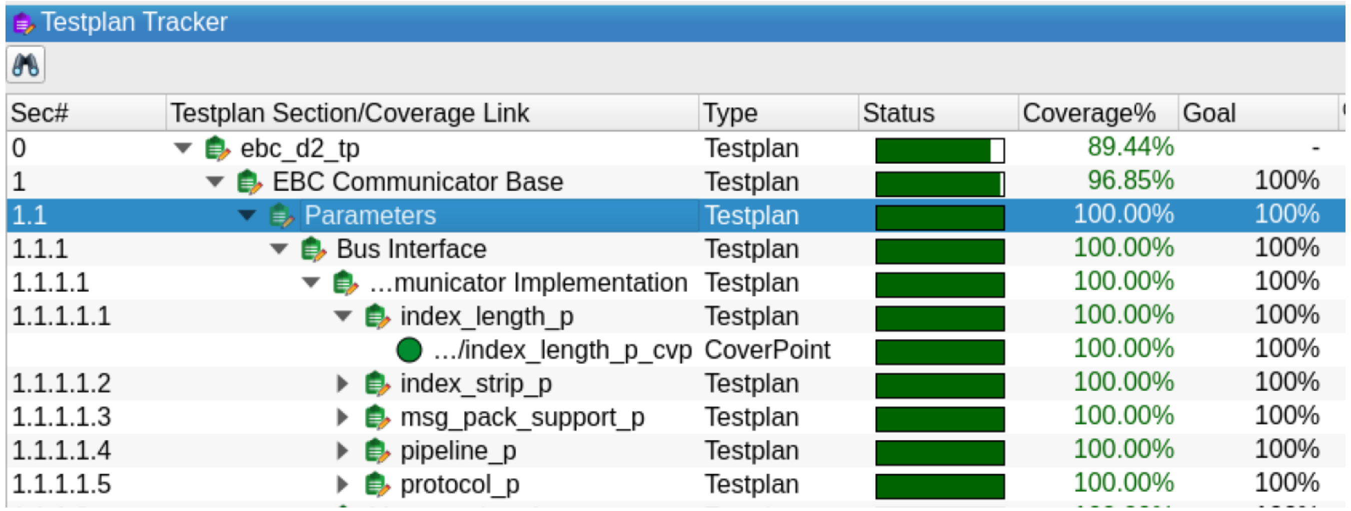
ebc_d2_tp | ebc_axi_d2_ALL_cov_accum 89.73%

Created By posche57, Jun 6. Last modified by iroberts, 27 days ago.

#	Section	Coverage %
1	EBC Communicator Base (ebc_tp_base_virtual)	96.85%
1.1	Parameters	100.00%
1.1.1	Bus Interface	100.00%
1.1.1.1	EBC Communicator Implementation	100.00%
1.1.1.1.1	index_length_p	100.00%
	● EBC_params_cvg:index_length_p_cvp	
1.1.1.1.2	index_strip_p	100.00%
1.1.1.1.3	msg_pack_support_p	100.00%
1.1.1.1.4	pipeline_p	100.00%
1.1.1.1.5	protocol_p	100.00%
1.1.1.2	Message Interface	100.00%
1.1.1.3	Clock Domain Crossing	100.00%
1.1.2	AXI Interface	100.00%
1.2	Features	93.71%
1.3	Testcases	E
2	Downstream EBC (ebc_tp_downstream_virtual)	100.00%
3	Upstream EBC (ebc_tp_upstream_virtual)	100.00%

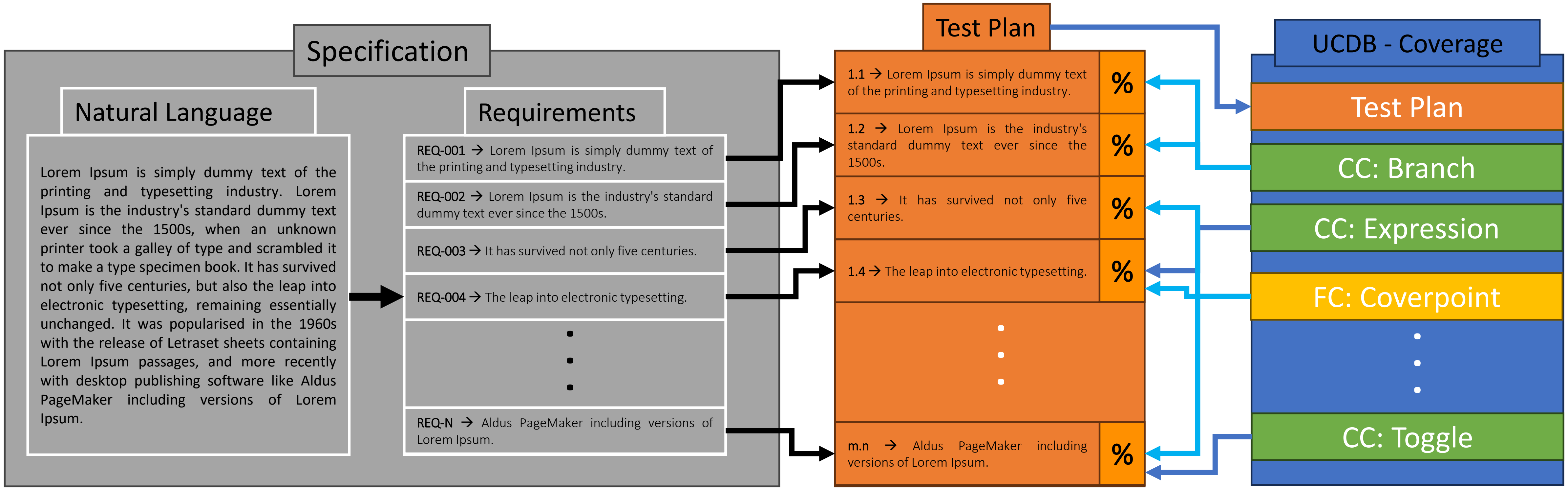
Test Plan Coverage Tracking in UCDB

- Test plan merged to the UCDB
  - UCDB = Unified Coverage Database
- Automatically merged at the end of each nightly regression run
- Accumulated UCDB reflected in test plan
- UCDB flow and VIQ dashboards covered in later topics


Testplan Tracker

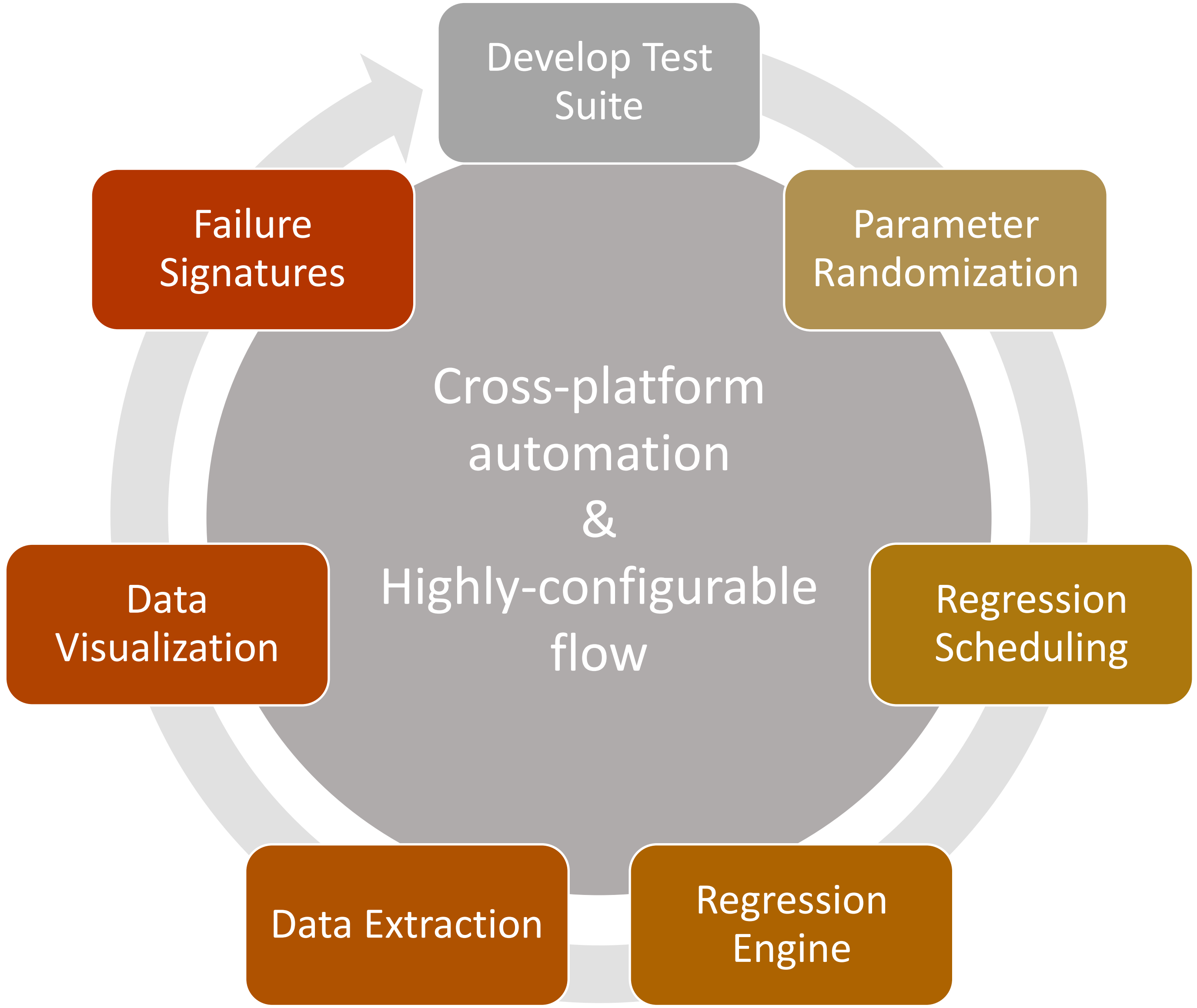
Sec#	Testplan Section/Coverage Link	Type	Status	Coverage%	Goal
0	ebc_d2_tp	Testplan	🟩	89.44%	-
1	EBC Communicator Base	Testplan	🟩	96.85%	100%
1.1	Parameters	Testplan	🟩	100.00%	100%
1.1.1	Bus Interface	Testplan	🟩	100.00%	100%
1.1.1.1	...municator Implementation	Testplan	🟩	100.00%	100%
1.1.1.1.1	index_length_p	Testplan	🟩	100.00%	100%
	.../index_length_p_cvp	CoverPoint	🟩	100.00%	100%
1.1.1.1.2	index_strip_p	Testplan	🟩	100.00%	100%
1.1.1.1.3	msg_pack_support_p	Testplan	🟩	100.00%	100%
1.1.1.1.4	pipeline_p	Testplan	🟩	100.00%	100%
1.1.1.1.5	protocol_p	Testplan	🟩	100.00%	100%

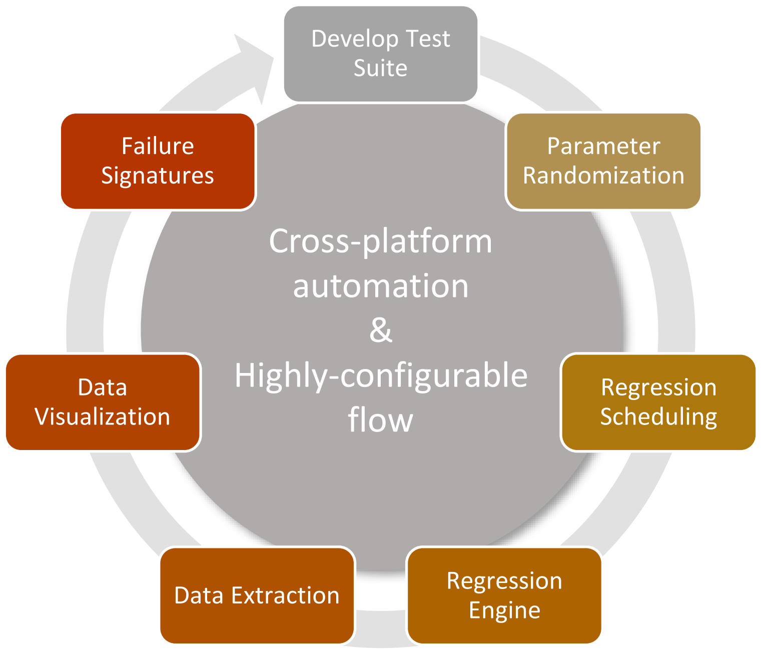
Specification vs. Test Plan vs. Coverage

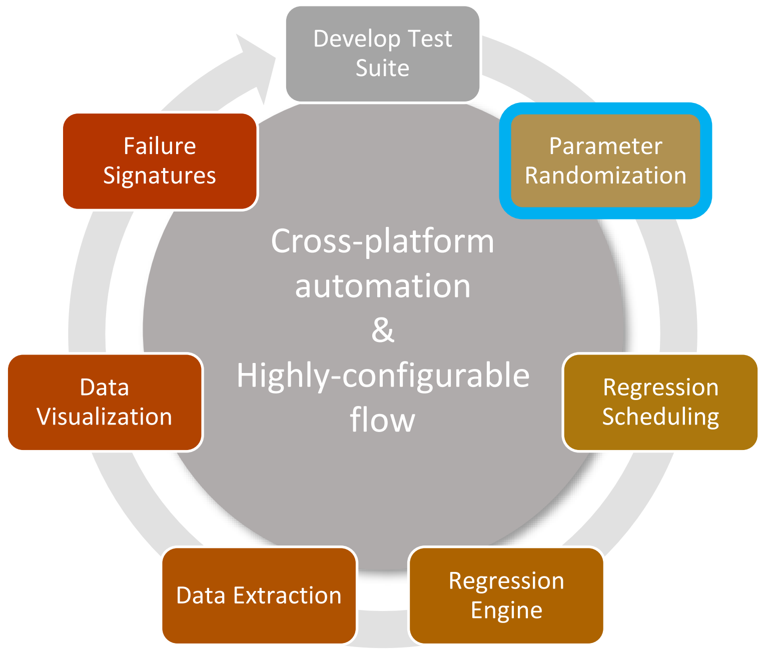


Specification
Natural Language
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum is the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.
Requirements
REQ-001 → Lorem Ipsum is simply dummy text of the printing and typesetting industry.
REQ-002 → Lorem Ipsum is the industry's standard dummy text ever since the 1500s.
REQ-003 → It has survived not only five centuries.
REQ-004 → The leap into electronic typesetting.
REQ-N → Aldus PageMaker including versions of Lorem Ipsum.
Test Plan
1.1 → Lorem Ipsum is simply dummy text of the printing and typesetting industry.
%
1.2 → Lorem Ipsum is the industry's standard dummy text ever since the 1500s.
%
1.3 → It has survived not only five centuries.
%
1.4 → The leap into electronic typesetting.
%
m.n → Aldus PageMaker including versions of Lorem Ipsum.
%
UCDB - Coverage
Test Plan
CC: Branch
CC: Expression
FC: Coverpoint
CC: Toggle

Structure for Effective Regressions

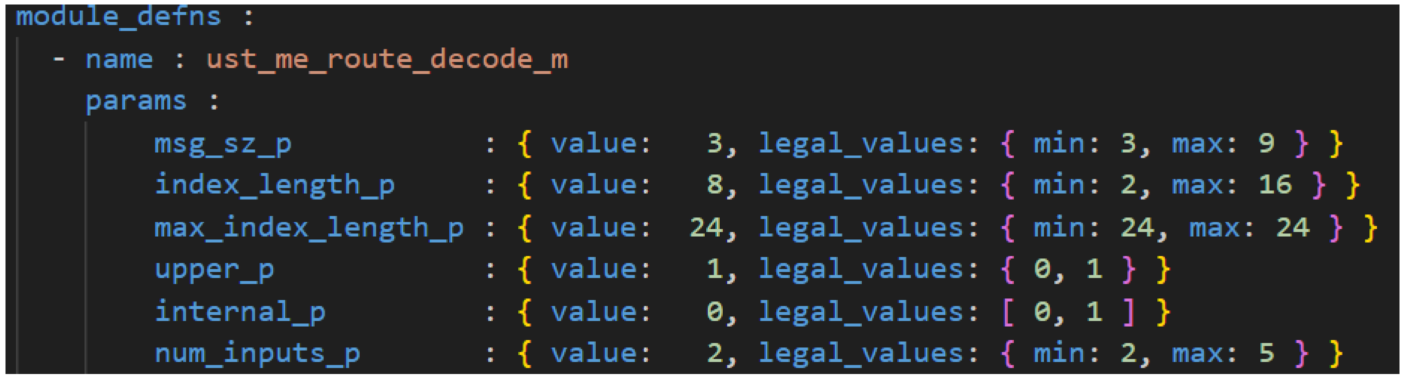

Develop Test Suite
Parameter Randomization
Regression Scheduling
Regression Engine
Data Extraction
Data Visualization
Failure Signatures
Cross-platform automation & Highly-configurable flow


Develop Test Suite
Parameter Randomization
Regression Scheduling
Regression Engine
Data Extraction
Data Visualization
Failure Signatures
Cross-platform automation & Highly-configurable flow

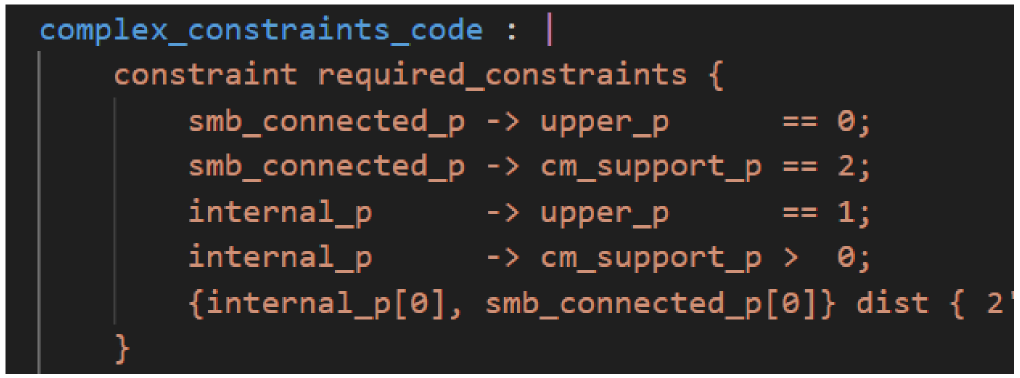


Parameter Randomization

- Essential for covering param-space
- Seed-based randomization for reproducibility
- Constraints for inter-dependencies (improves efficacy of batch-runs)

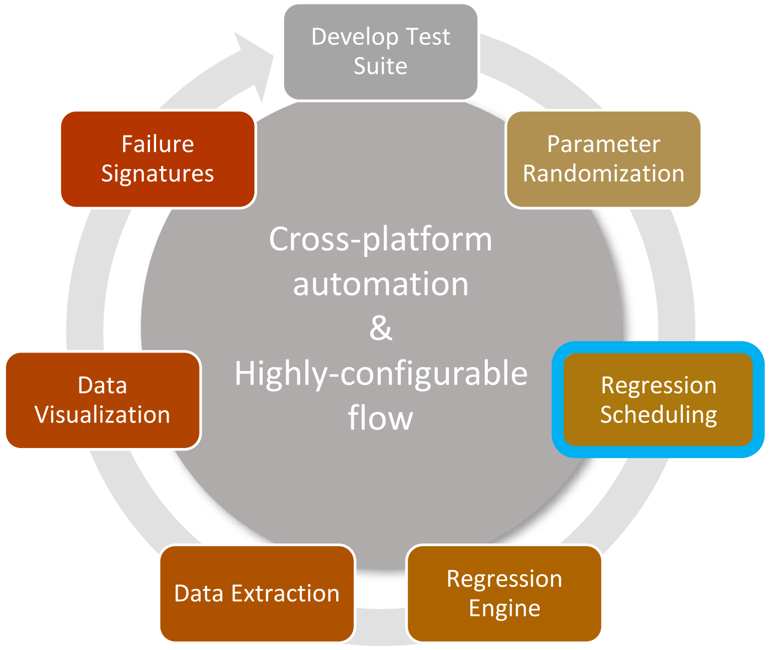


```
module_defns :
  - name : ust_me_route_decode_m
    params :
        msg_sz_p           : { value:   3, legal_values: { min: 3, max: 9 } }
        index_length_p     : { value:   8, legal_values: { min: 2, max: 16 } }
        max_index_length_p : { value:  24, legal_values: { min: 24, max: 24 } }
        upper_p            : { value:   1, legal_values: { 0, 1 } }
        internal_p         : { value:   0, legal_values: [ 0, 1 ] }
        num_inputs_p       : { value:   2, legal_values: { min: 2, max: 5 } }
```

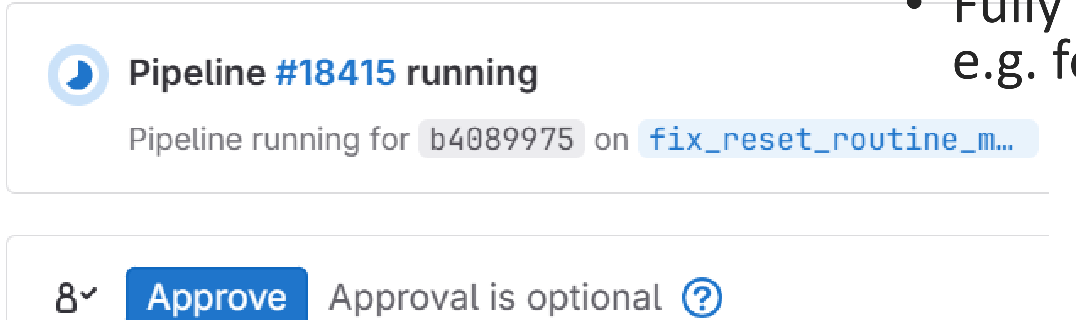


```
complex_constraints_code : |
    constraint required_constraints {
        smb_connected_p -> upper_p      == 0;
        smb_connected_p -> cm_support_p == 2;
        internal_p      -> upper_p      == 1;
        internal_p      -> cm_support_p >  0;
        {internal_p[0], smb_connected_p[0]} dist { 2
    }
```

Regression Scheduling



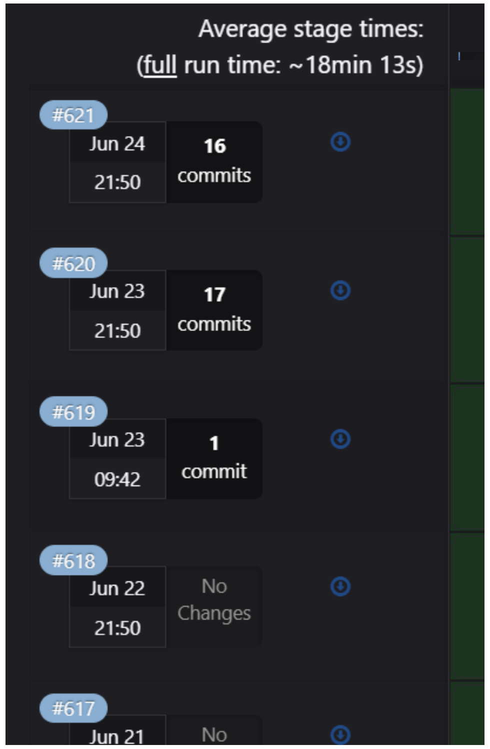
- Frequency: continuous integration, nightly & weekends
- Different regression lists for different purposes:
  - Mini (5mins), nightly, long (weekend), formal, PSS
  - Fully flexible for custom test lists e.g. for specific configurations



Pipeline #18415 running

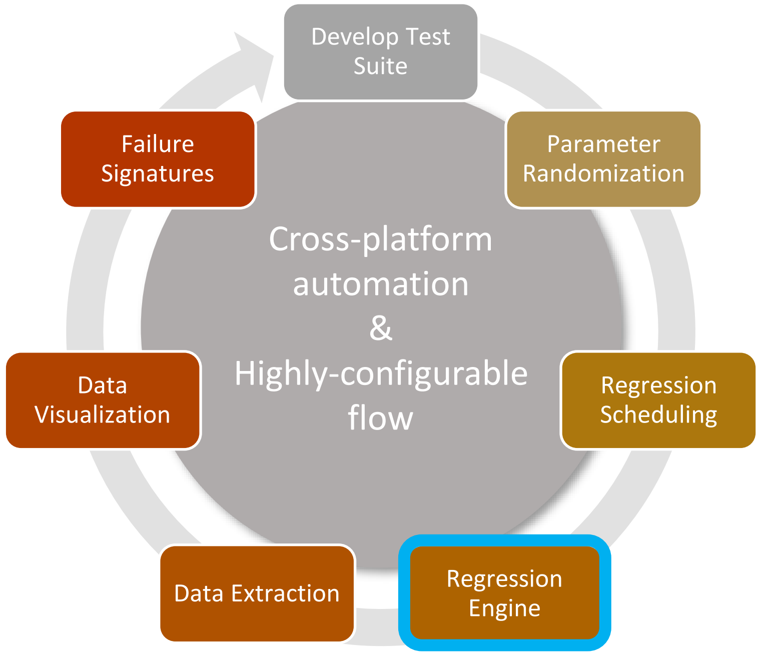
Pipeline running for `b4089975` on `fix_reset_routine_m…`

8 Approve Approval is optional



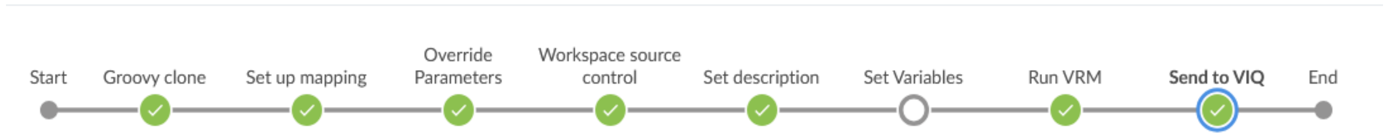
Average stage times:
(full run time: ~18min 13s)

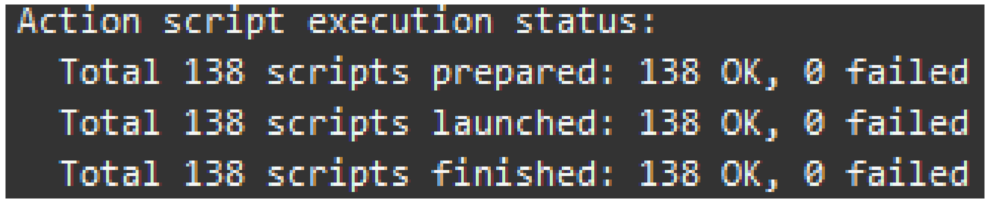
#621	Jun 24 21:50	16 commits
#620	Jun 23 21:50	17 commits
#619	Jun 23 09:42	1 commit
#618	Jun 22 21:50	No Changes
#617	Jun 21	No



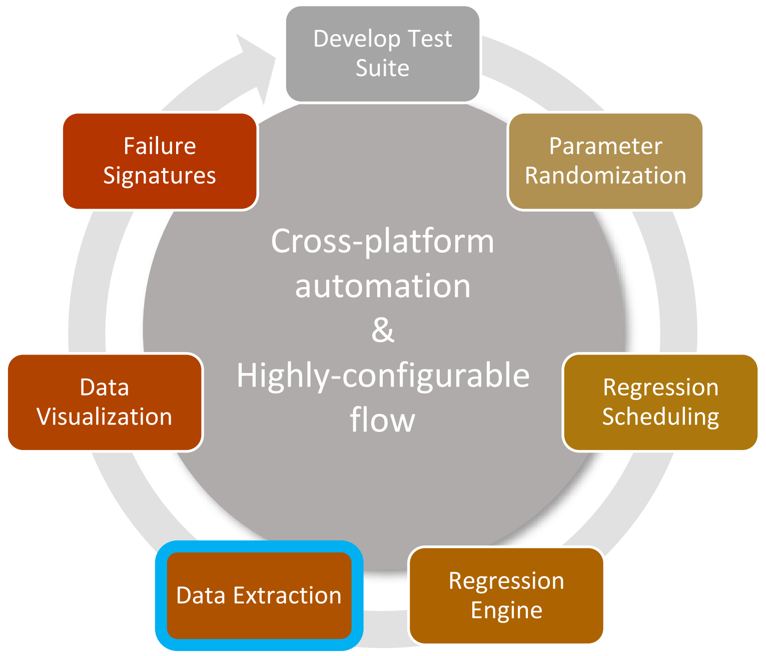
Regression Engine

- Many stages of tests & scripts
- Executes config-randomization, PSS, simulation, coverage, data extraction

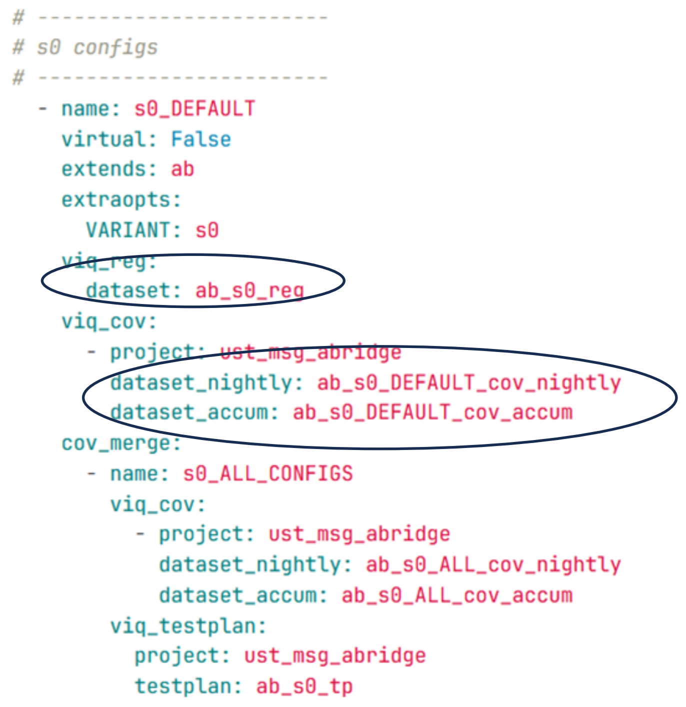




```
Action script execution status:
   Total 138 scripts prepared: 138 OK, 0 failed
   Total 138 scripts launched: 138 OK, 0 failed
   Total 138 scripts finished: 138 OK, 0 failed
```

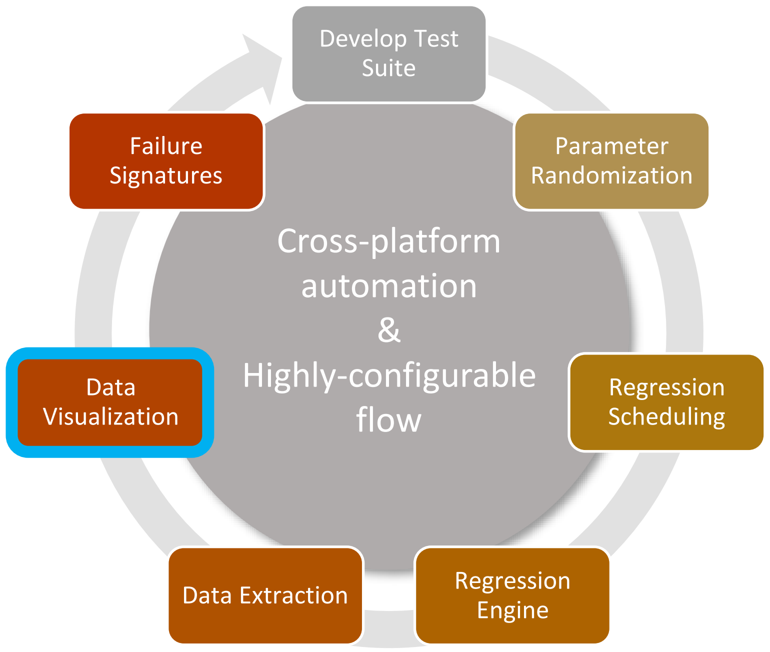


Data Extraction



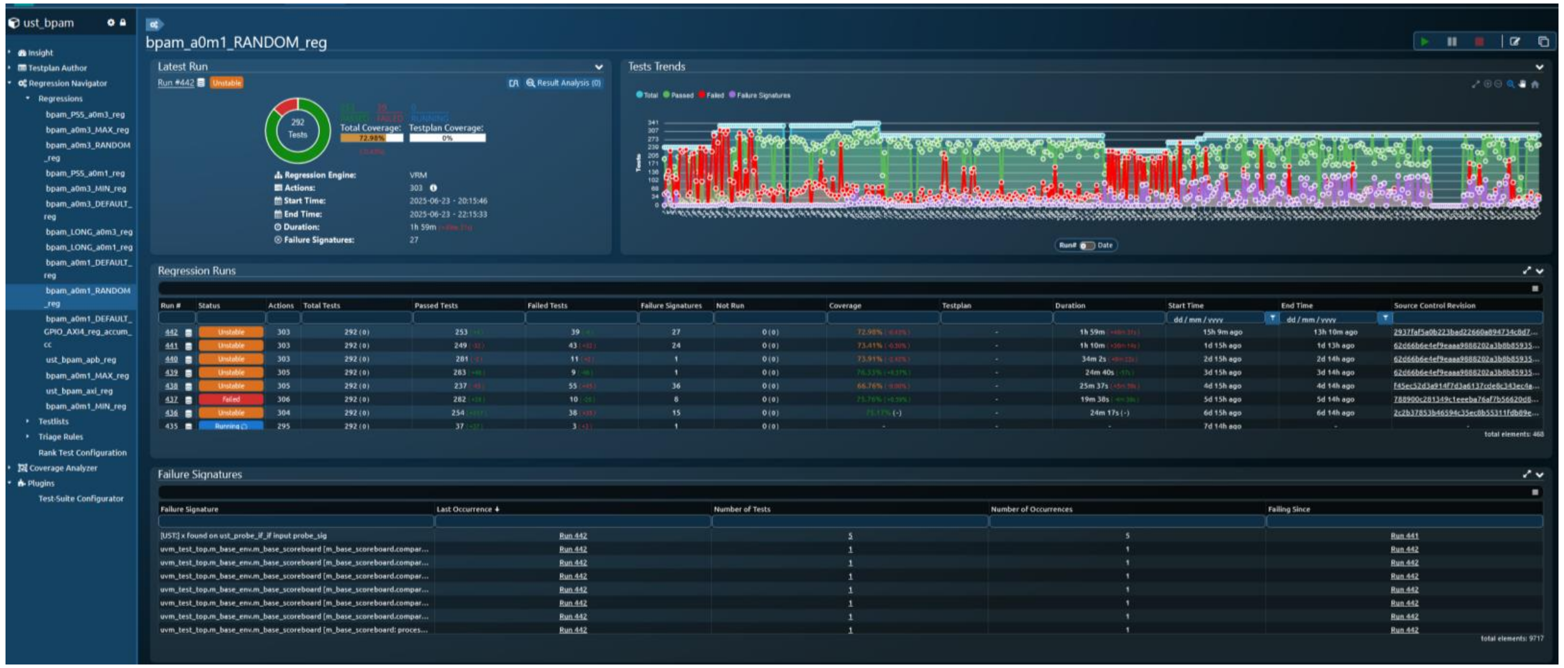
```
# -------------------------
# s0 configs
# -------------------------
  - name: s0_DEFAULT
    virtual: False
    extends: ab
    extraopts:
      VARIANT: s0
    viq_reg:
      dataset: ab_s0_reg
    viq_cov:
      - project: ust_msg_abridge
        dataset_nightly: ab_s0_DEFAULT_cov_nightly
        dataset_accum: ab_s0_DEFAULT_cov_accum
    cov_merge:
      - name: s0_ALL_CONFIGS
        viq_cov:
          - project: ust_msg_abridge
            dataset_nightly: ab_s0_ALL_cov_nightly
            dataset_accum: ab_s0_ALL_cov_accum
        viq_testplan:
          project: ust_msg_abridge
          testplan: ab_s0_tp
```

- Extractor to port all regression results and UCDBs to one location
- In-house flow (Yaml, Make, Python, Groovy)
- Automated test plan coverage annotation, coverage merging, unreachability-based waivers & repository data statistics



Data Visualization

- Separate datasets can be visualized
- Trigger for regression engine



ust_bpam

- Insight
- Testplan Author
- Regression Navigator
  - Regressions
    - bpam_P55_a0m3_reg
    - bpam_a0m3_MAX_reg
    - bpam_a0m3_RANDOM_reg
    - bpam_P55_a0m1_reg
    - bpam_a0m3_MIN_reg
    - bpam_a0m3_DEFAULT_reg
    - bpam_LONG_a0m3_reg
    - bpam_LONG_a0m1_reg
    - bpam_a0m1_DEFAULT_reg
    - bpam_a0m1_RANDOM_reg
    - bpam_a0m1_DEFAULT_GPIO_AXI4_reg_accum_cc
    - ust_bpam_apb_reg
    - bpam_a0m1_MAX_reg
    - ust_bpam_axi_reg
    - bpam_a0m1_MIN_reg
  - Testlists
  - Triage Rules
  - Rank Test Configuration
- Coverage Analyzer
- Plugins
  - Test-Suite Configurator

bpam_a0m1_RANDOM_reg

Latest Run

Run #442 Unstable — Result Analysis (0)

292 Tests

Total Coverage: 72.98% — Testplan Coverage: 0%

Regression Engine:	VRM
Actions:	303
Start Time:	2025-06-23 - 20:15:46
End Time:	2025-06-23 - 22:15:33
Duration:	1h 59m
Failure Signatures:	27

Tests Trends

● Total ● Passed ● Failed ● Failure Signatures

Run# Date

Regression Runs

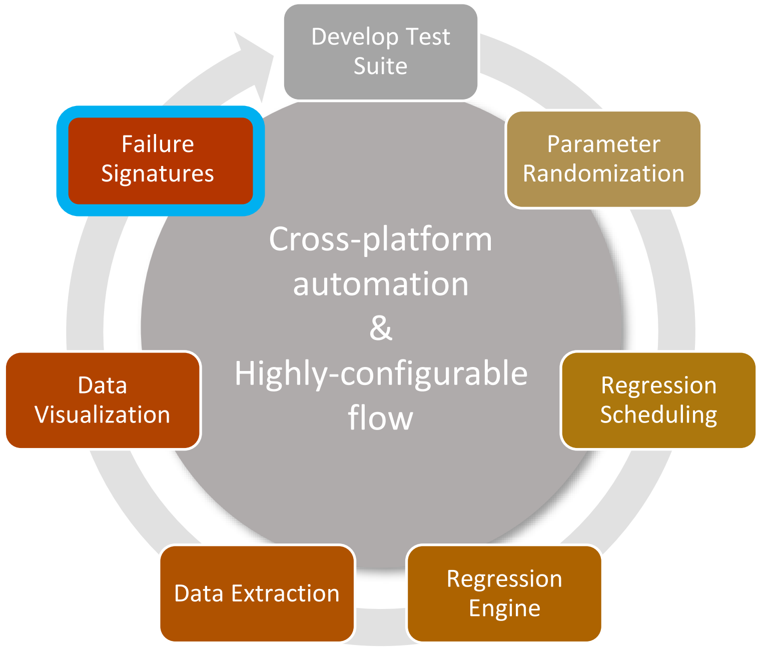
Run #	Status	Actions	Total Tests	Passed Tests	Failed Tests	Failure Signatures	Not Run	Coverage	Testplan	Duration	Start Time	End Time	Source Control Revision
											dd / mm / yyyy	dd / mm / yyyy	
442	Unstable	303	292 (0)	253	39	27	0 (0)	72.98%	-	1h 59m	15h 9m ago	13h 10m ago	2937faf5a0b223bad22660a894734c8d7...
441	Unstable	303	292 (0)	249	43	24	0 (0)	73.41%	-	1h 10m	1d 15h ago	1d 13h ago	62d66b6e4ef9eaaa9888202a3b0b85935...
440	Unstable	303	292 (0)	281	11	1	0 (0)	73.91%	-	34m 2s	2d 15h ago	2d 14h ago	62d66b6e4ef9eaaa9888202a3b0b85935...
439	Unstable	305	292 (0)	283	9	1	0 (0)	76.33%	-	24m 40s	3d 15h ago	3d 14h ago	62d66b6e4ef9eaaa9888202a3b0b85935...
438	Unstable	305	292 (0)	237	55	36	0 (0)	66.76%	-	25m 37s	4d 15h ago	4d 14h ago	f45ec52d3a914f7d3a6137cde8c343ec4a...
437	Failed	306	292 (0)	282	10	8	0 (0)	75.76%	-	19m 38s	5d 15h ago	5d 14h ago	788900c281349c1eeeba76af7b56620d8...
436	Unstable	304	292 (0)	254	38	15	0 (0)	75.17% (-)	-	24m 17s (-)	6d 15h ago	6d 14h ago	2c2b37853b46594c35ec0b55311fdb89e...
435	Running	295	292 (0)	37	3	1	0 (0)	-	-	-	7d 14h ago	-	-

total elements: 468

Failure Signatures

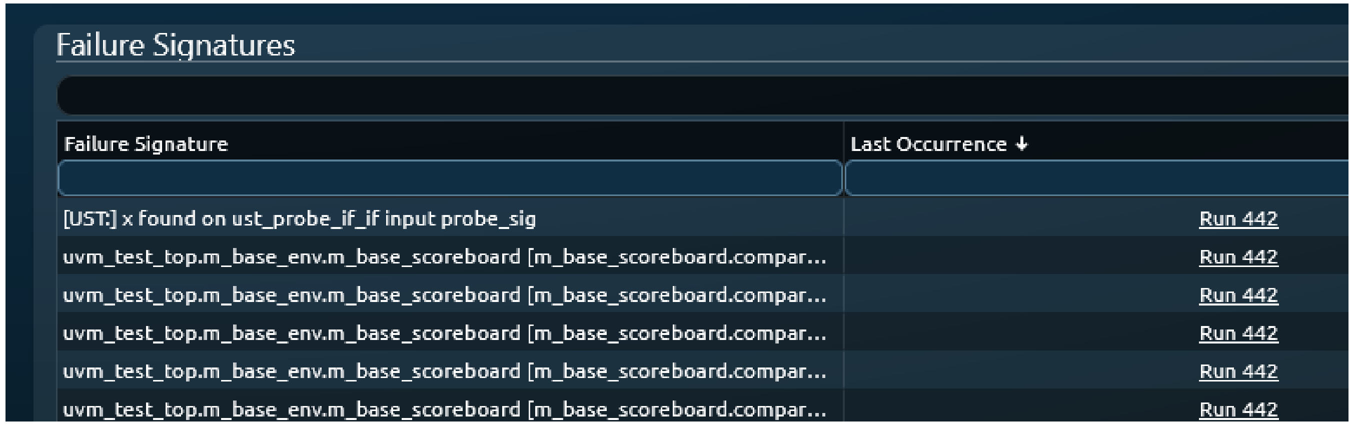
Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
[UST:] x found on ust_probe_if_if input probe_sig	Run 442	5	5	Run 441
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442	1	1	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard: proces...	Run 442	1	1	Run 442

total elements: 9717



Failure Signatures

- Automated data scraping
- View statistics on each failure message



Failure Signatures

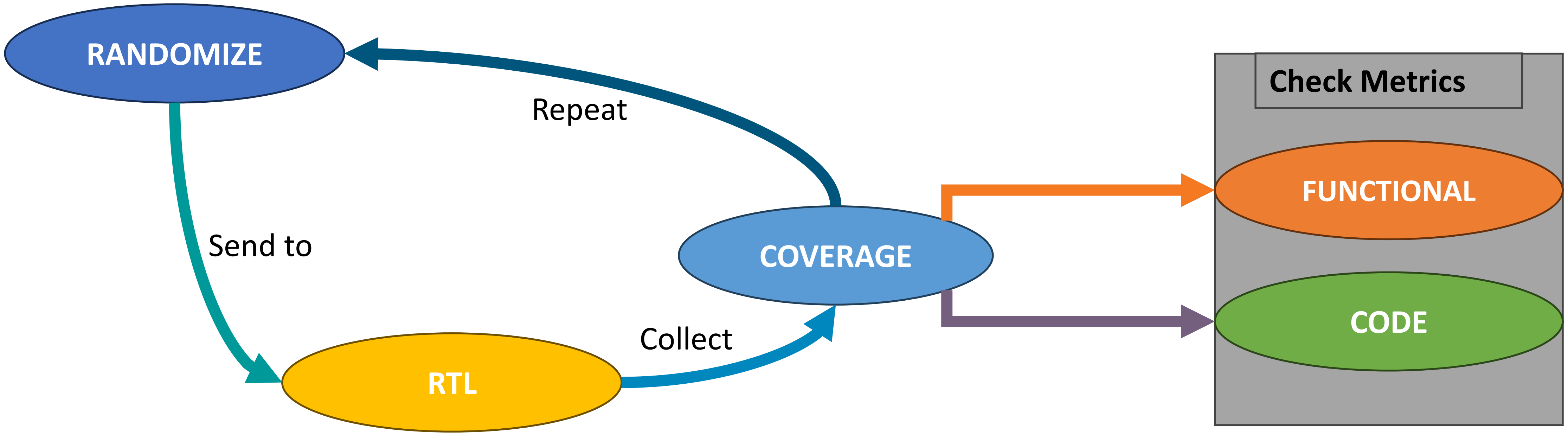
Failure Signature	Last Occurrence ↓
[UST:] x found on ust_probe_if_if input probe_sig	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442
uvm_test_top.m_base_env.m_base_scoreboard [m_base_scoreboard.compar...	Run 442

- View failure log and run command with rand-seeds to rerun the same param config and test stimulus

Accumulated Coverage Structure

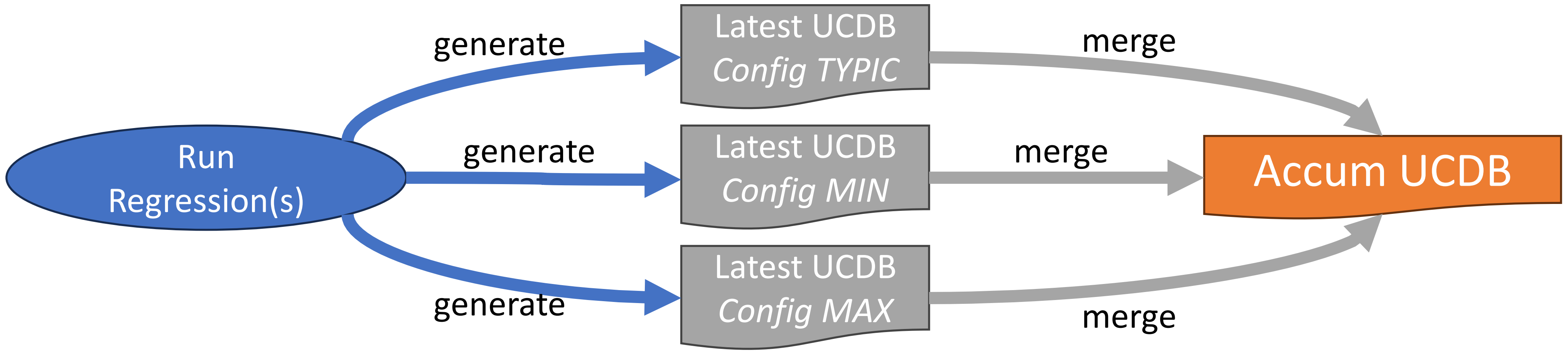
Metric-Driven Verification (MDV)

- When should we use it?
  - Sometimes, a system has **too many possible inputs**.
  - Brute-force **is not**, therefore, **an option**.
  - Metric-Driven Verification is based on **RANDOMIZATION.**

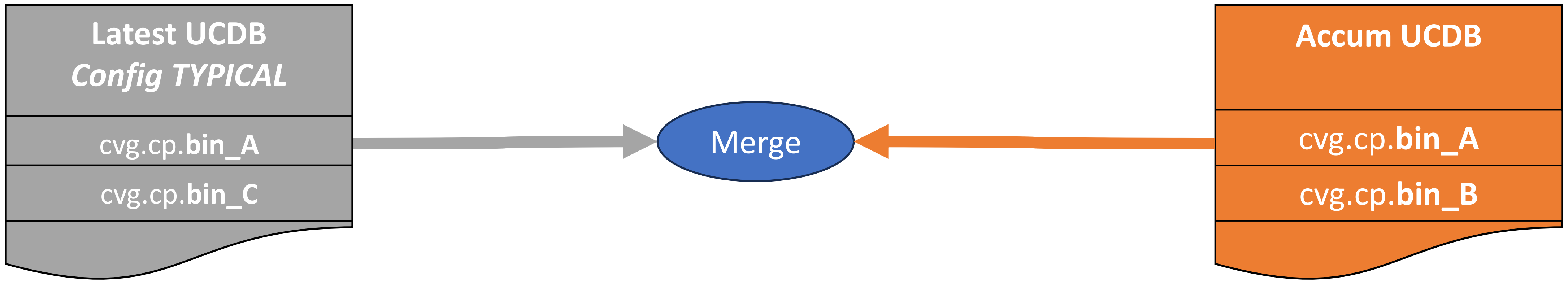


<u>**M**</u>etric-<u>**D**</u>riven <u>**V**</u>erification (<u>**MDV**</u>)

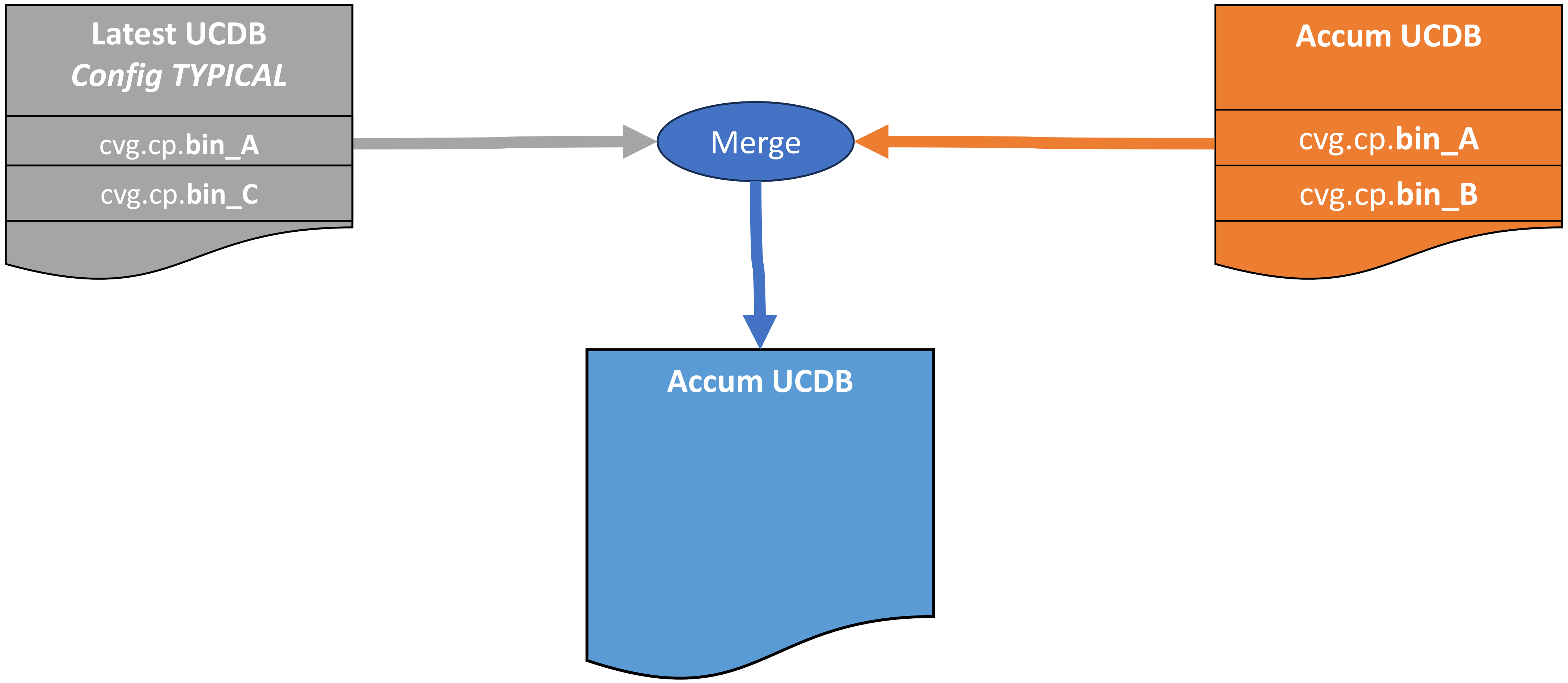
- Collecting coverage for highly-configurable IPs require **robust strategy**.
- Using **automation tools** (e.g., Jenkins) is critical to success.
- Two different accumulation approaches will be explored.
- Accumulation is based on <u>**merging**</u> UCDB files.



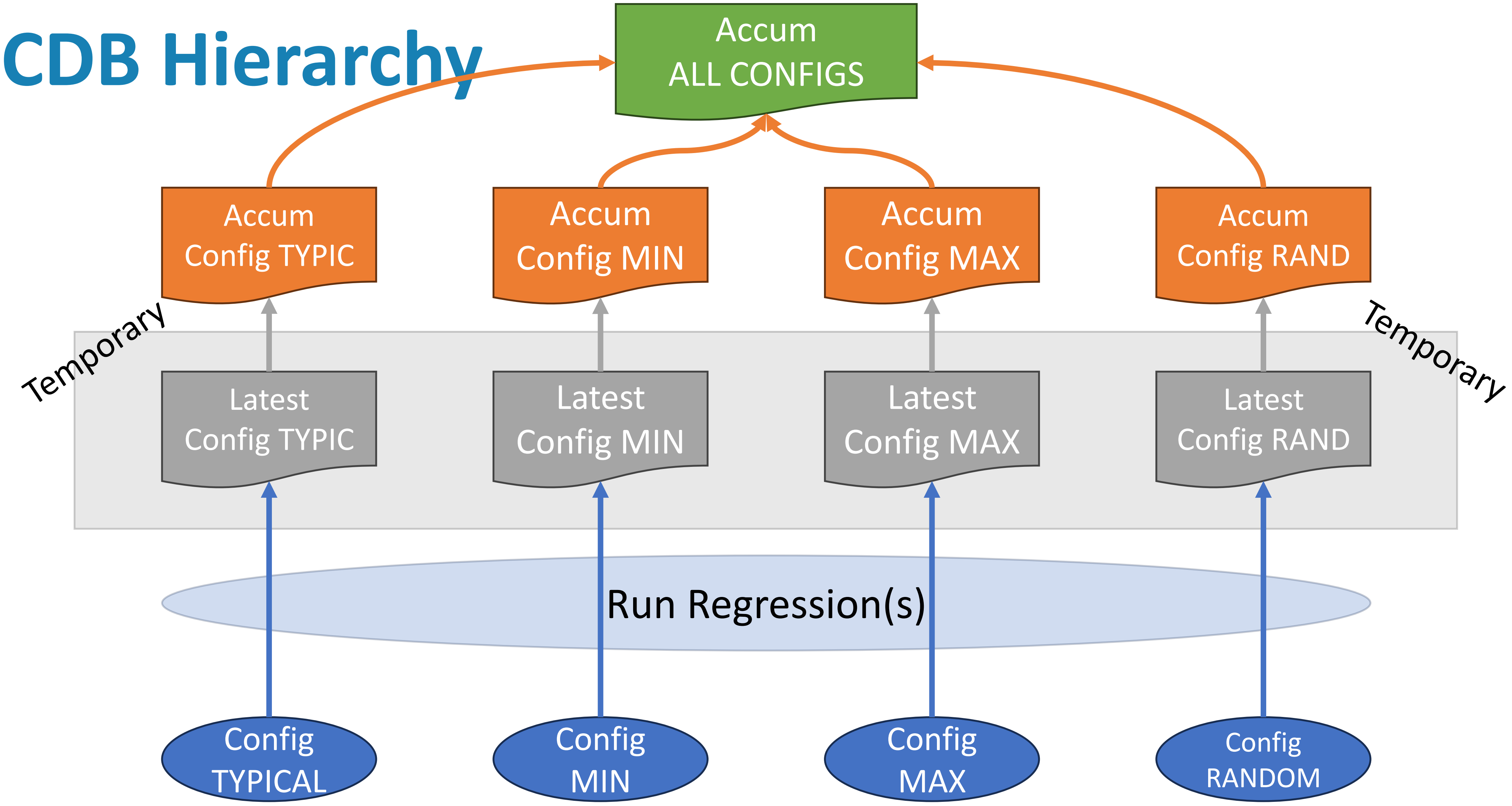
Merging Strategy



Merging Strategy

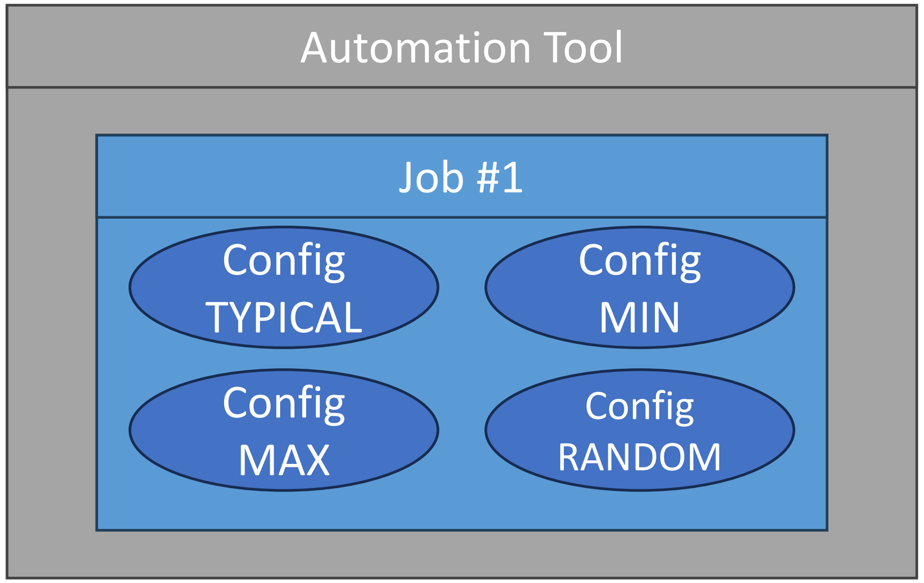


UCDB Hierarchy



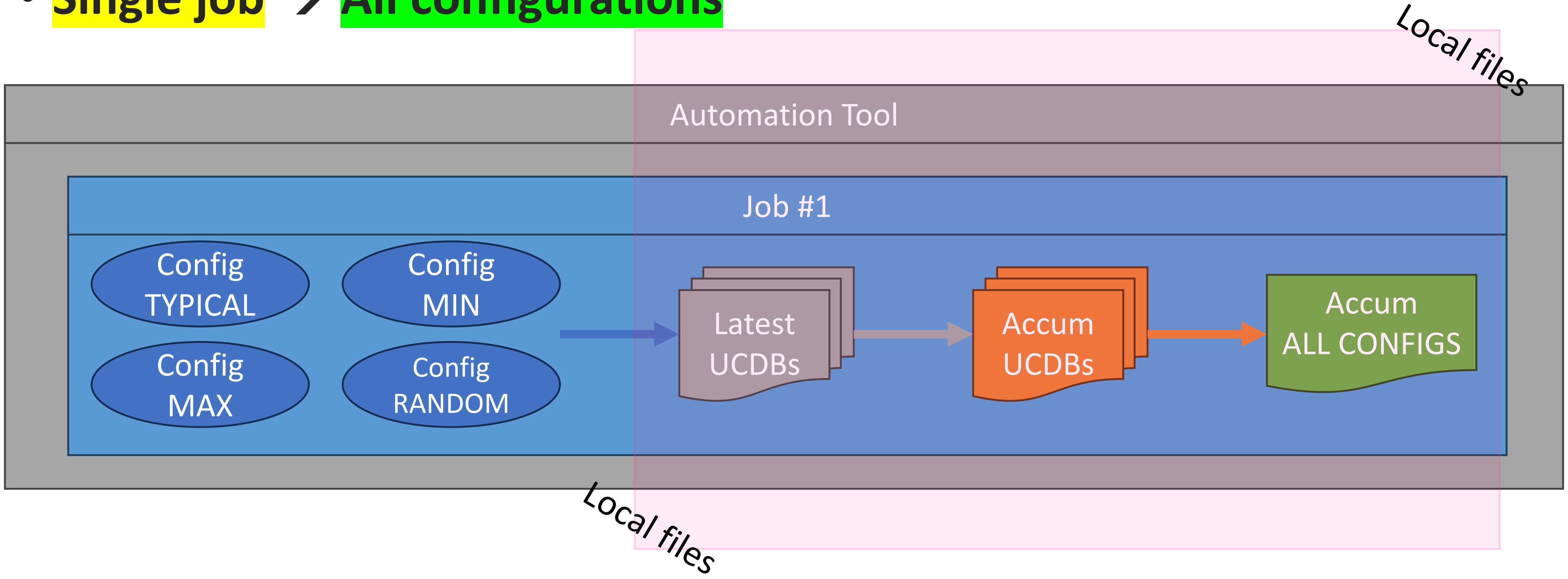
Automation: **Scenario 1**

- **Single job → All configurations**



Automation: **Scenario 1**

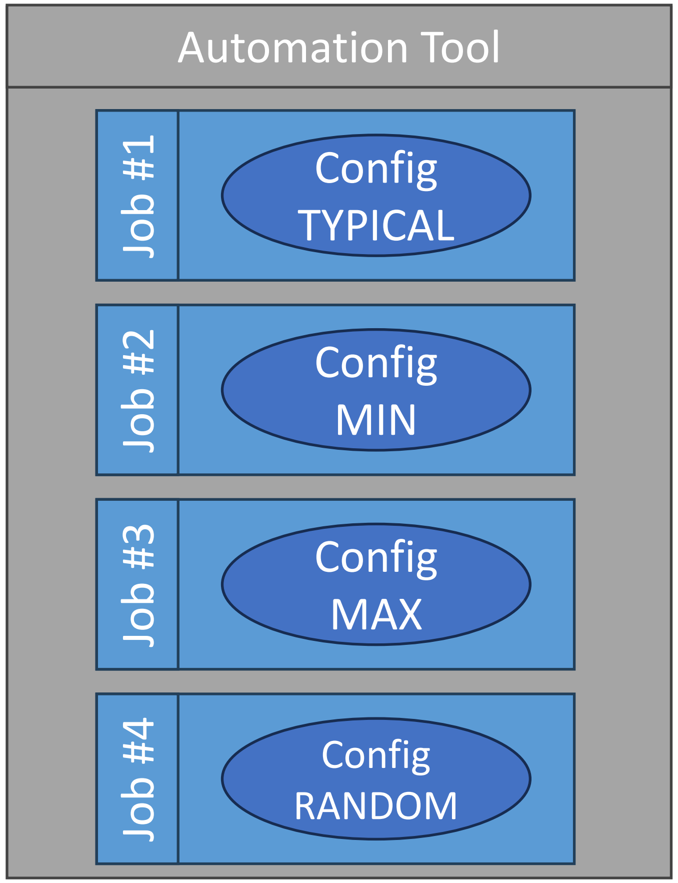
- **Single job → All configurations**



Local files *→ within a given job's workspace*

Automation: **Scenario 2**

- **Single job → Single configuration**

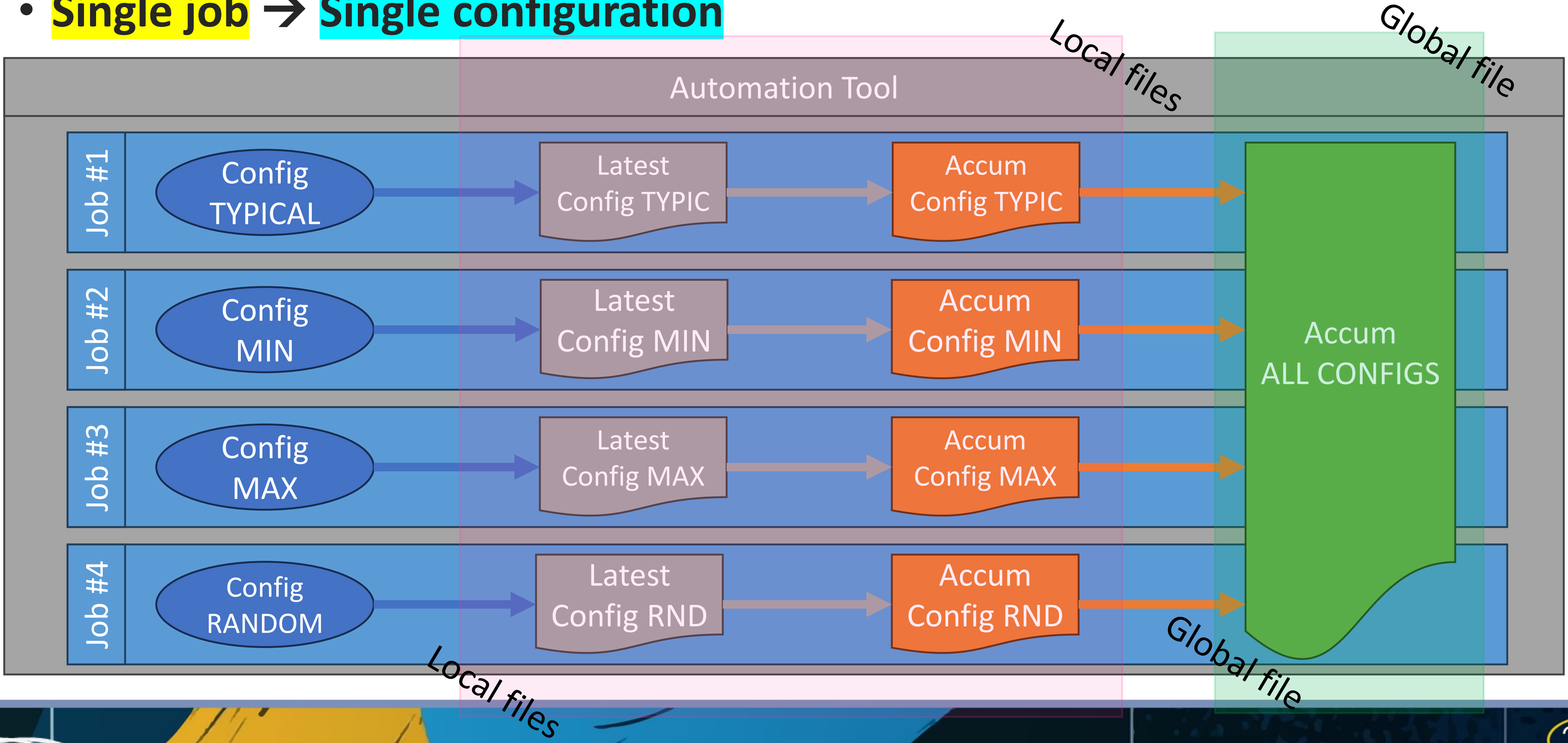


Automation: **Scenario 2**

Local files *→ within a given job's workspace*

Global files *→ shared by multiple jobs*

- **Single job** → **Single configuration**



Considerations

- Choose the right accumulation path based on the structure.

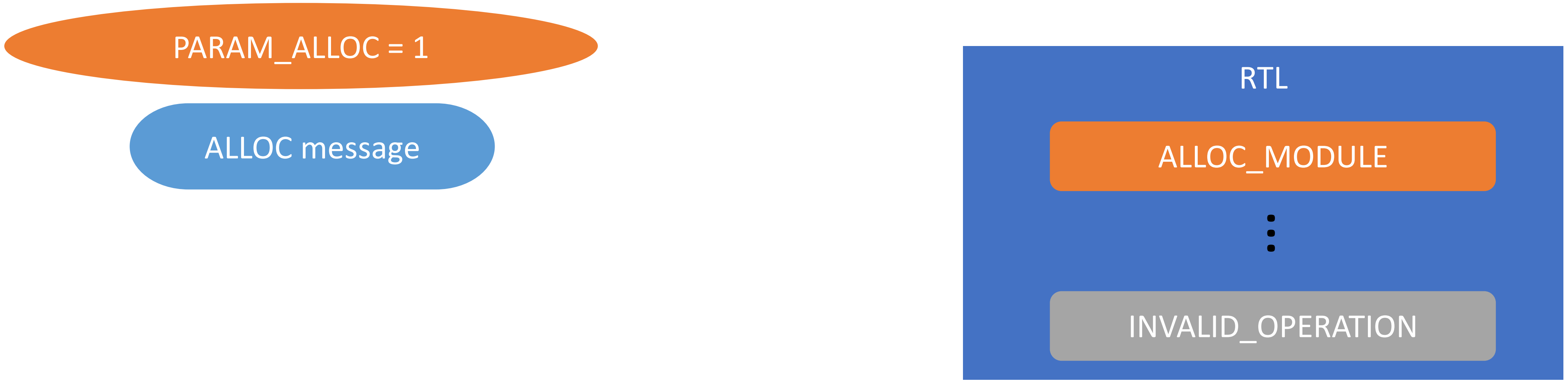
- Analyze intermediary UCDBs as needed.

- Easily check how thoroughly the IP was verified using the ALL_CONFIGS.

Traffic Light System & Unreachability

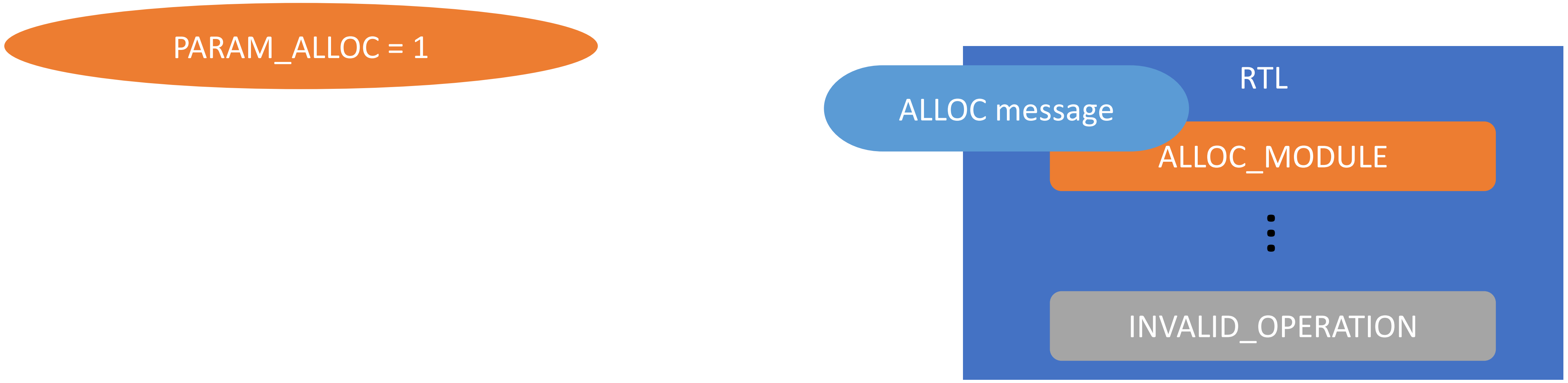
Introduction

- In a world with **highly configurable IPs**, different parts of the RTL might be unreachable depending on the configuration.



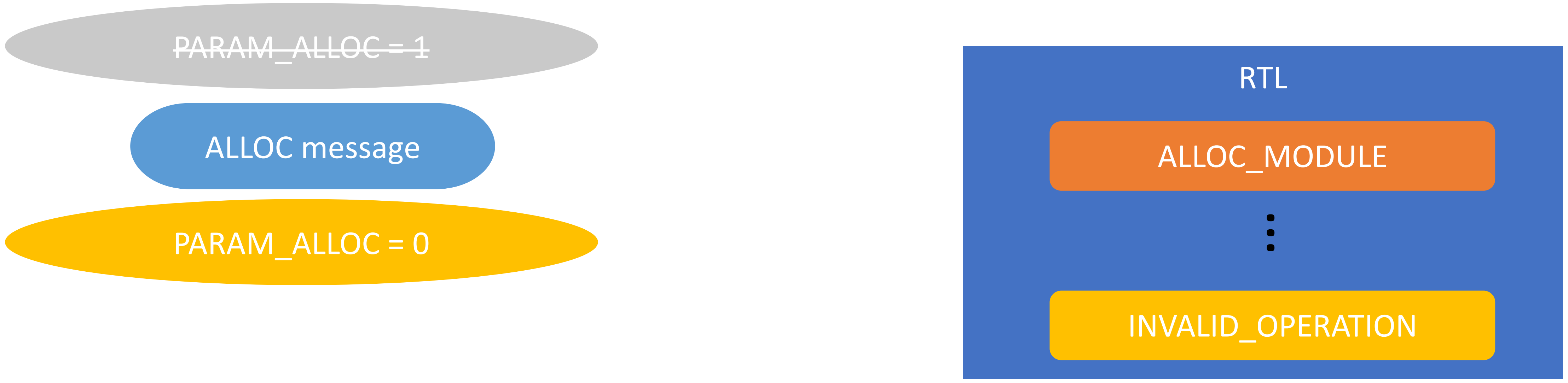
Introduction

- In a world with **highly configurable IPs**, different parts of the RTL might be unreachable depending on the configuration.



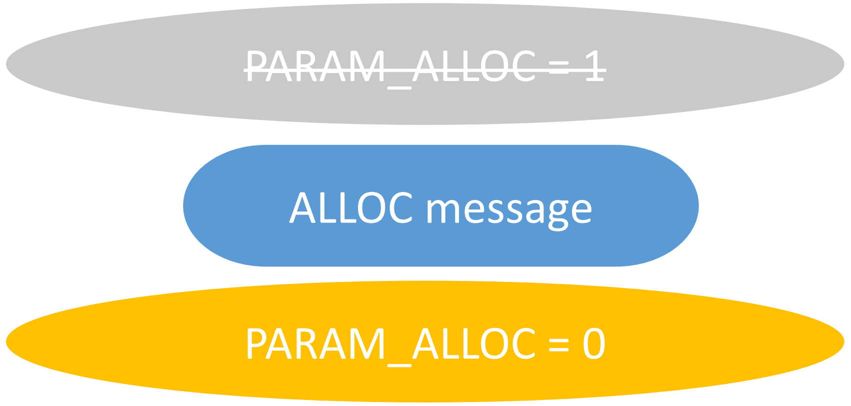
Introduction

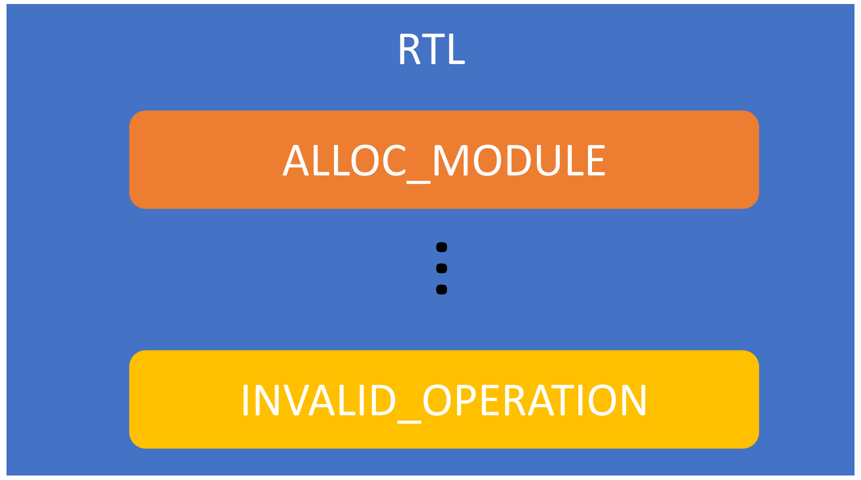
- In a world with **highly configurable IPs**, different parts of the RTL might be unreachable depending on the configuration.



Introduction

- In a world with **highly configurable IPs**, different parts of the RTL might be unreachable depending on the configuration.





- With **PARAM_ALLOC disabled**, ALLOC_MODULE can't be covered.
- **How to deal with that?**

Traffic Light System: *Coverage Goals*

- Coverage collected both for **function coverage** and **code coverage**.

- Coverage target goals are:
  - ➢ **100% Function Coverage**
    - ❑ Upgraded methodology to be param aware.
  - ➢ **100% "explained" Code coverage**
    - ❑ Explained means there are waivers for unreachable code.

- **Why the Traffic Light System?**
  - ➢ Allows for initial thorough analysis.
  - ➢ Forces, on Red Waivers, to go back for more implementation/analysis.

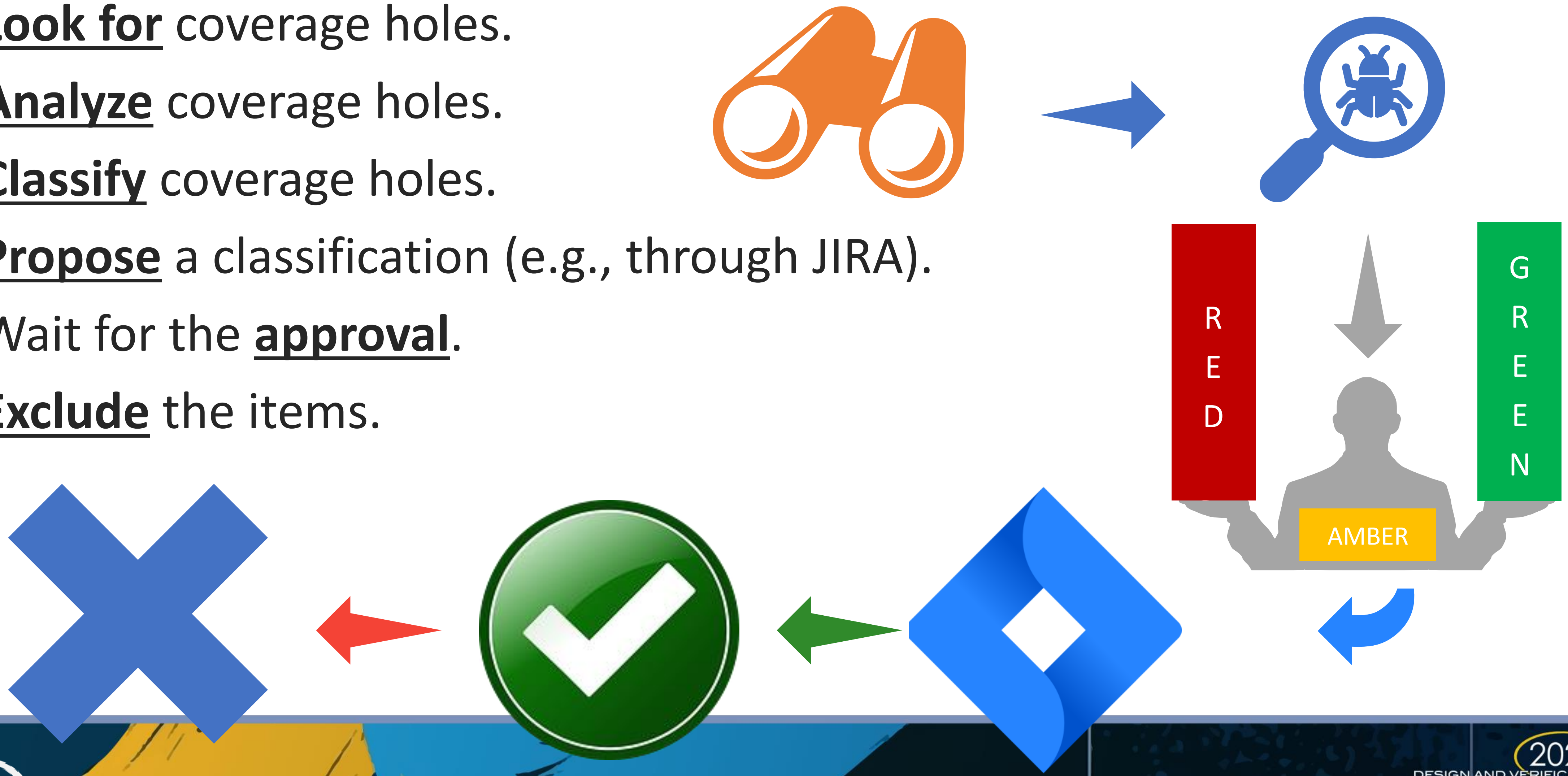
Traffic Light System

Waiver classifications:

- Red
  - ➢ Reachable, but **untested**
  - ➢ Requires further attention.
- Amber
  - ➢ Reachable, but only **partly tested**.
  - ➢ High confidence it will be covered.
  - ➢ **Might** require further attention.
  - ➢ e.g., Cover Crosses (FC).
- Green
  - **Unreachable**.
  - Requires clear justification and reviews.
  - Once accepted, no need for further attention.
  - e.g., different configuration

Traffic Light System: *Proposed methodology*

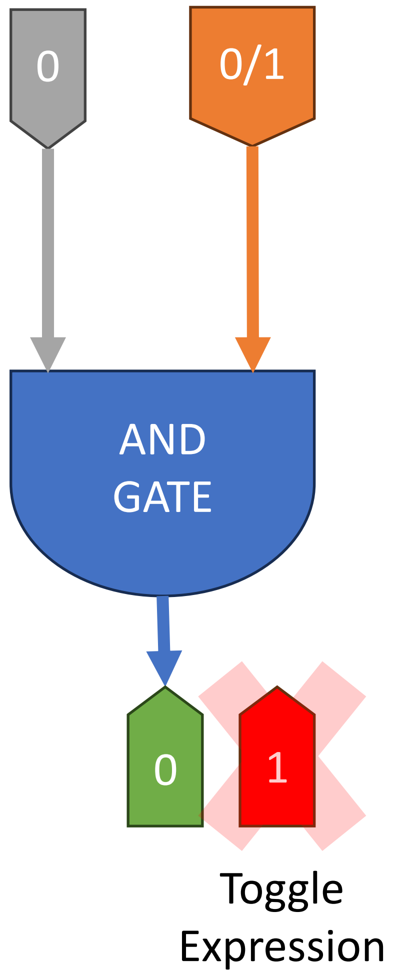
1. **Look for** coverage holes.
2. **Analyze** coverage holes.
3. **Classify** coverage holes.
4. **Propose** a classification (e.g., through JIRA).
5. Wait for the **approval**.
6. **Exclude** the items.

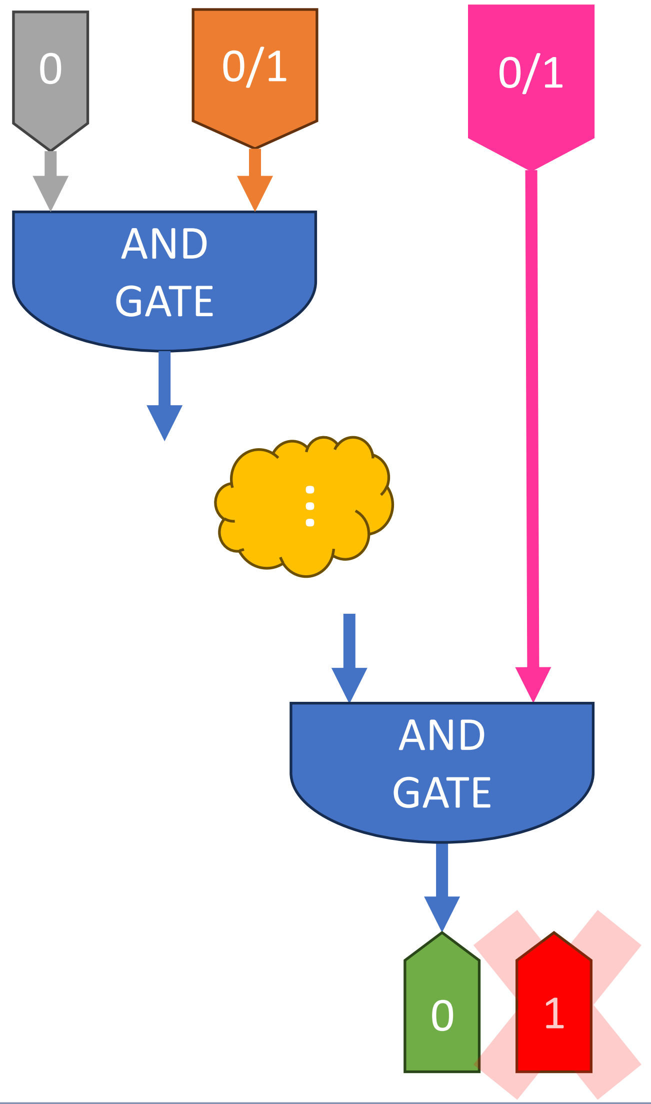


Unreachability

- Formal Unreachability
  - Mathematically unreachable.
  - Formal Verification tools.

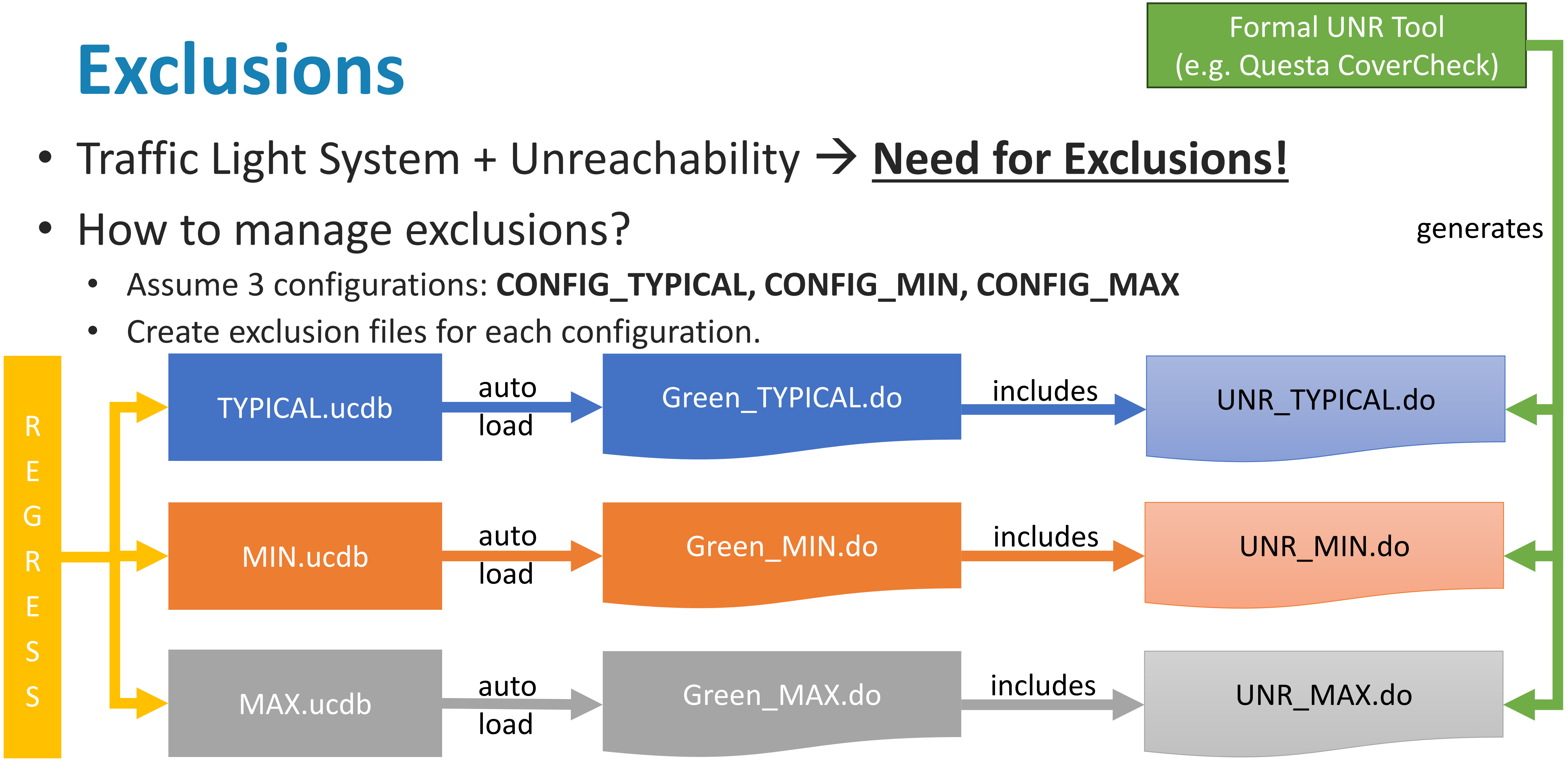
- Proposed Methodology:
  1. Run unreachability analysis tool.
  2. Analyze proposed exclusions.
  3. Apply exclusions.





Exclusions

- Traffic Light System + Unreachability → <u>**Need for Exclusions!**</u>
- How to manage exclusions?
  - Assume 3 configurations: **CONFIG_TYPICAL, CONFIG_MIN, CONFIG_MAX**
  - Create exclusion files for each configuration.



Parameter-aware Functional Coverage

- Code Coverage can be excluded; **Functional Coverage shall not be for Green Waiver.**
- Introducing **Parameter-aware Functional Coverage**:
  - ➢ Create **auto-excluding** Covergroups, Coverpoints and Bins based on parameters.
- Covergroups → `if (<COND>) cvg = new("name");`
- Coverpoints:

```
option.weight          = (<COND>);
option.at_least        = (<COND>);
type_option.weight     = (<COND>);
```

What if `option.per_instance = 1`?

- Bins → `bins b_A = {'h0} with (<COND>);`

RANDOM Configuration

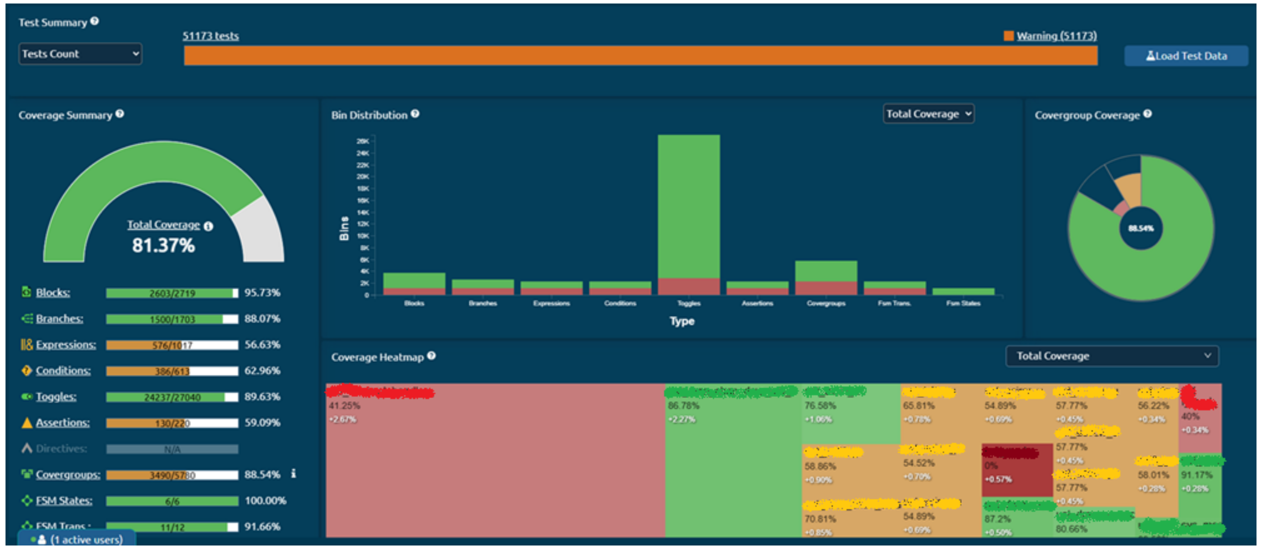
- Randomize all parameters → Allows for covering all possible configurations over time.

- Addition of **COV_ALL** flag.
  - ➢ **No parameter-aware in RANDOM** → We need to see coverage holes.

➢ Add **covergroup** for parameters.

Collaborative Work – Coverage Analyzer (VIQ)

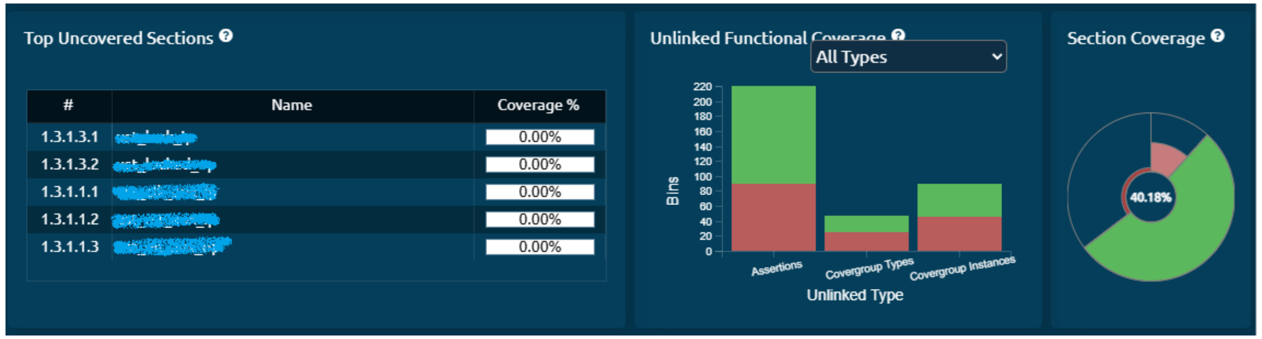
- VIQ offers a collaborative work environment to analyze and manage coverage.
- It also keeps track of testplans merged into coverage files



Coverage Summary

Total Coverage 81.37%

Blocks:	2603/2719	95.73%
Branches:	1500/1703	88.07%
Expressions:	576/1017	56.63%
Conditions:	386/613	62.96%
Toggles:	24237/27040	89.63%
Assertions:	130/220	59.09%
Directives:	N/A	
Covergroups:	3490/5780	88.54%
FSM States:	6/6	100.00%
FSM Trans.:	11/12	91.66%



Top Uncovered Sections

#	Name	Coverage %
1.3.1.3.1	[redacted]	0.00%
1.3.1.3.2	[redacted]	0.00%
1.3.1.1.1	[redacted]	0.00%
1.3.1.1.2	[redacted]	0.00%
1.3.1.1.3	[redacted]	0.00%

Considerations

- For **Code Coverage**, well-maintained Exclusions are the key to success.

- For **Functional Coverage**, well-designed parameter-aware covergroups shall expedite verification.

Real-time and Interactive Dashboards

Dashboards

ASIC Verification Complexity

- Highly configurable IPs require verification across **multiple configurations**
- Leads to **exponential growth** in verification effort (almost infinite permutations
- Managing diverse configurations and variants creates significant data challenges
- Need for **data-driven decision making** thus super important to present this data in a concise and meaningful way.
- Crucial that data is accurate and Real-Time

Type of Metrics

- Regression Pass / Fail Status
- Code and Function Coverage
- Bug tracking system statistics
- Specifications / requirements items
- Verification plan items
- Repository statistics
- Compute resources metrics (CPU, Memory etc)
- Test/regression resource statistics (e.g. run time)

Challenges Without Real-Time Dashboards

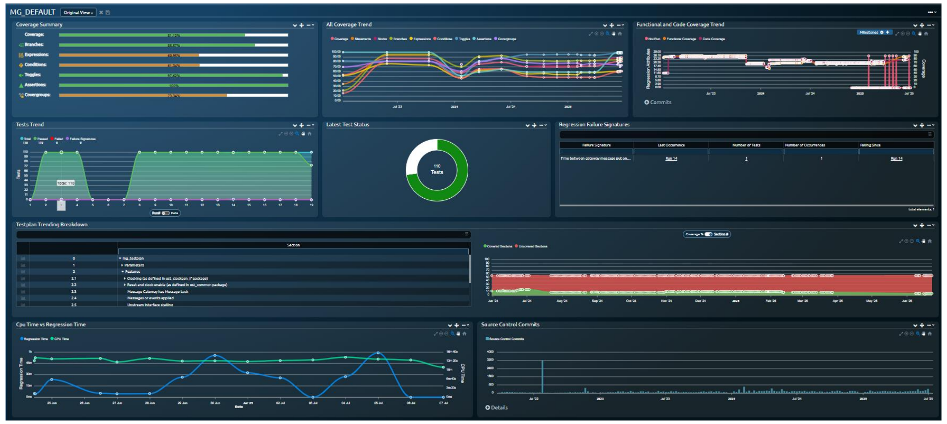
- **Delayed insights** into verification progress and issues
- **Manual collection and correlation** of metrics across configurations can lead into misinterpretation
- **Reactive rather than proactive** issue detection
- **Siloed information** between teams and verification stages
- Can lead to **stale data** very quickly
- **Time-consuming** report generation and analysis

Benefits of Real-Time & Interactive Dashboards

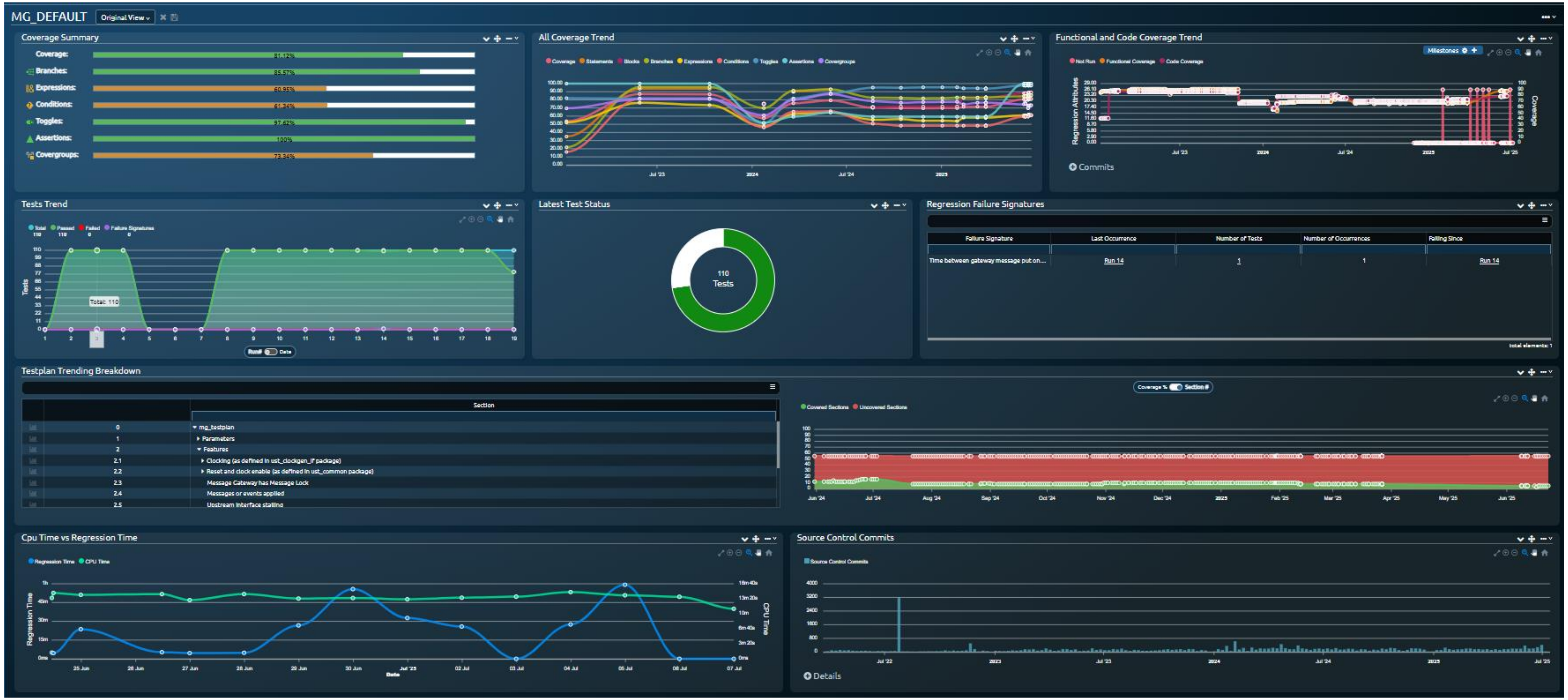
- Immediate visibility into verification progress
- Early bug detection and trend analysis
- Resource optimization and management
- Comprehensive coverage tracking across configurations
- Data-driven decision making
- Efficient generation of verification health reports.

Benefit 1: Immediate Visibility

- **At-a-glance** “health status” overview of design and verification
- **Real-time status** across all regression runs and configurations
- **Aggregated view** of pass/fail metrics across variants and also coverage
- **Immediate feedback** on new test additions or code changes
- Better understanding of **verification bottlenecks**


MG_DEFAULT
Coverage Summary
All Coverage Trend
Functional and Code Coverage Trend
Tests Trend
Latest Test Status
Regression Failure Signatures
Testplan Trending Breakdown
Cpu Time vs Regression Time
Source Control Commits

Benefit 1: Immediate Visibility



Coverage Summary

Coverage:	81.12%
Branches:	85.57%
Expressions:	60.95%
Conditions:	61.34%
Toggles:	97.62%
Assertions:	100%
Covergroups:	73.34%

Regression Failure Signatures

Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
Time between gateway message put on...	Run 14	1	1	Run 14

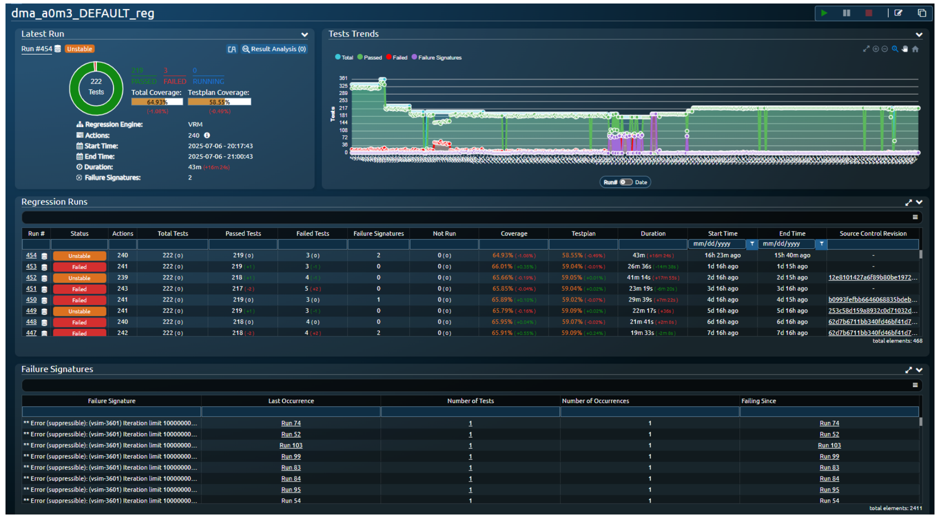
total elements: 1

Testplan Trending Breakdown

	Section
0	▾ mg_testplan
1	▸ Parameters
2	▾ Features
2.1	▸ Clocking (as defined in ust_clockgen_if package)
2.2	▸ Reset and clock enable (as defined in ust_common package)
2.3	Message Gateway has Message Lock
2.4	Messages or events applied
2.5	Upstream interface stalling

Benefit 2: Early Bug Detection

- **Live bug convergence trends** highlight potential issues early
- **Interactive bug curves** allow drill-down into problematic areas
- **Cross-configuration bug correlation** identifies systemic issues
- **Predictive analytics** for forecasting time to verification closure



dma_a0m3_DEFAULT_reg

Latest Run

Run #454 Unstable — Result Analysis (0)

222 Tests — 219 PASSED, 3 FAILED, 0 RUNNING

Total Coverage: 64.93% (-1.08%) Testplan Coverage: 58.55% (-0.49%)

Regression Engine:	VRM
Actions:	240
Start Time:	2025-07-06 - 20:17:43
End Time:	2025-07-06 - 21:00:43
Duration:	43m (+16m 24s)
Failure Signatures:	2

Regression Runs

Run #	Status	Actions	Total Tests	Passed Tests	Failed Tests	Failure Signatures	Not Run	Coverage	Testplan	Duration	Start Time	End Time	Source Control Revision
											mm/dd/yyyy	mm/dd/yyyy	
454	Unstable	240	222 (0)	219 (0)	3 (0)	2	0 (0)	64.93% (-1.08%)	58.55% (-0.49%)	43m (+16m 24s)	16h 23m ago	15h 40m ago	-
453	Failed	241	222 (0)	219 (+1)	3 (-1)	0	0 (0)	66.01% (+0.35%)	59.04% (-0.01%)	26m 36s (-14m 38s)	1d 16h ago	1d 15h ago	-
452	Unstable	239	222 (0)	218 (+1)	4 (-1)	0	0 (0)	65.66% (-0.19%)	59.05% (+0.01%)	41m 14s (+17m 55s)	2d 16h ago	2d 15h ago	12e8101427a6f89b80be1972...
451	Failed	243	222 (0)	217 (-2)	5 (+2)	0	0 (0)	65.85% (-0.04%)	59.04% (+0.02%)	23m 19s (-6m 20s)	3d 16h ago	3d 16h ago	-
450	Failed	241	222 (0)	219 (0)	3 (0)	1	0 (0)	65.89% (+0.10%)	59.02% (-0.07%)	29m 39s (+7m 22s)	4d 16h ago	4d 15h ago	b0993fefbb6646068835bdeb...
449	Unstable	241	222 (0)	219 (+1)	3 (-1)	0	0 (0)	65.79% (-0.16%)	59.09% (+0.02%)	22m 17s (+36s)	5d 16h ago	5d 16h ago	253c58d159a8932c0d71032d...
448	Failed	240	222 (0)	218 (0)	4 (0)	0	0 (0)	65.95% (+0.04%)	59.07% (-0.02%)	21m 41s (+2m 8s)	6d 16h ago	6d 16h ago	62d7b6711bb340fd46bf41d7...
447	Failed	242	222 (0)	218 (-2)	4 (+2)	2	0 (0)	65.91% (+0.55%)	59.09% (+0.24%)	19m 33s (-2m 8s)	7d 16h ago	7d 16h ago	62d7b6711bb340fd46bf41d7...

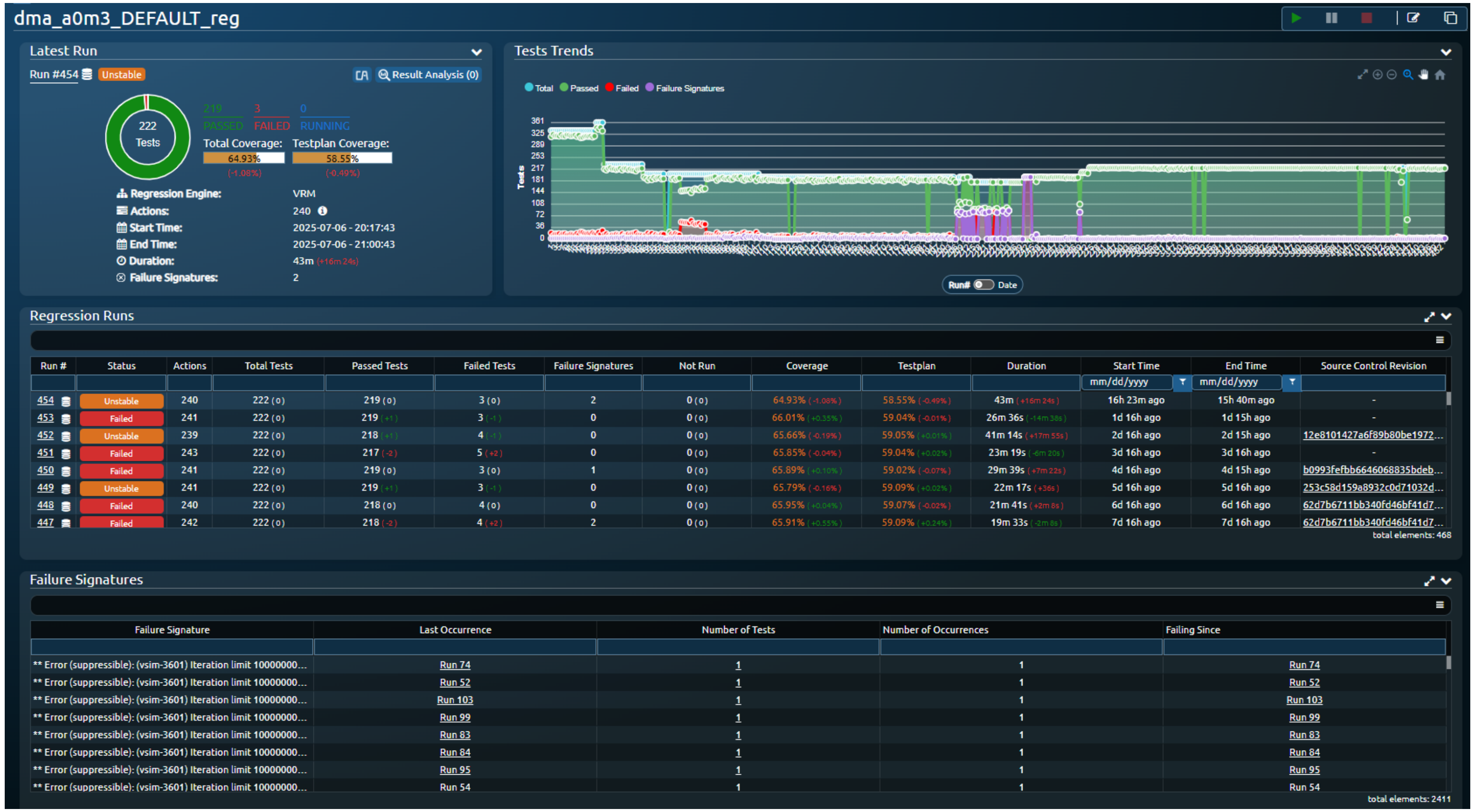
total elements: 468

Failure Signatures

Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 74	1	1	Run 74
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 52	1	1	Run 52
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 103	1	1	Run 103
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 99	1	1	Run 99
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 83	1	1	Run 83
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 84	1	1	Run 84
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 95	1	1	Run 95
** Error (suppressible): (vsim-3601) Iteration limit 10000000...	Run 54	1	1	Run 54

total elements: 2411

Benefit 2: Early Bug Detection



dma_a0m3_DEFAULT_reg

Latest Run

Run #454 Unstable — Result Analysis (0)

222 Tests — 219 PASSED, 3 FAILED, 0 RUNNING

Total Coverage: 64.93% (-1.08%) Testplan Coverage: 58.55% (-0.49%)

Regression Engine:	VRM
Actions:	240
Start Time:	2025-07-06 - 20:17:43
End Time:	2025-07-06 - 21:00:43
Duration:	43m (+16m 24s)
Failure Signatures:	2

Tests Trends

Regression Runs

Run #	Status	Actions	Total Tests	Passed Tests	Failed Tests	Failure Signatures	Not Run	Coverage	Testplan	Duration	Start Time	End Time	Source Control Revision
											mm/dd/yyyy	mm/dd/yyyy	
454	Unstable	240	222 (0)	219 (0)	3 (0)	2	0 (0)	64.93% (-1.08%)	58.55% (-0.49%)	43m (+16m 24s)	16h 23m ago	15h 40m ago	-
453	Failed	241	222 (0)	219 (+1)	3 (-1)	0	0 (0)	66.01% (+0.35%)	59.04% (-0.01%)	26m 36s (-14m 38s)	1d 16h ago	1d 15h ago	-
452	Unstable	239	222 (0)	218 (+1)	4 (-1)	0	0 (0)	65.66% (-0.19%)	59.05% (+0.01%)	41m 14s (+17m 55s)	2d 16h ago	2d 15h ago	12e8101427a6f89b80be1972…
451	Failed	243	222 (0)	217 (-2)	5 (+2)	0	0 (0)	65.85% (-0.04%)	59.04% (+0.02%)	23m 19s (-6m 20s)	3d 16h ago	3d 16h ago	-
450	Failed	241	222 (0)	219 (0)	3 (0)	1	0 (0)	65.89% (+0.10%)	59.02% (-0.07%)	29m 39s (+7m 22s)	4d 16h ago	4d 15h ago	b0993fefbb6646068835bdeb…
449	Unstable	241	222 (0)	219 (+1)	3 (-1)	0	0 (0)	65.79% (-0.16%)	59.09% (+0.02%)	22m 17s (+36s)	5d 16h ago	5d 16h ago	253c58d159a8932c0d71032d…
448	Failed	240	222 (0)	218 (0)	4 (0)	0	0 (0)	65.95% (+0.04%)	59.07% (-0.02%)	21m 41s (+2m 8s)	6d 16h ago	6d 16h ago	62d7b6711bb340fd46bf41d7…
447	Failed	242	222 (0)	218 (-2)	4 (+2)	2	0 (0)	65.91% (+0.55%)	59.09% (+0.24%)	19m 33s (-2m 8s)	7d 16h ago	7d 16h ago	62d7b6711bb340fd46bf41d7…

total elements: 468

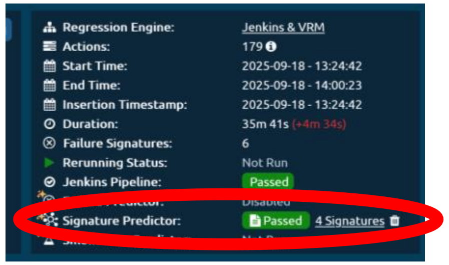
Failure Signatures

Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 74	1	1	Run 74
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 52	1	1	Run 52
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 103	1	1	Run 103
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 99	1	1	Run 99
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 83	1	1	Run 83
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 84	1	1	Run 84
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 95	1	1	Run 95
** Error (suppressible): (vsim-3601) Iteration limit 10000000…	Run 54	1	1	Run 54

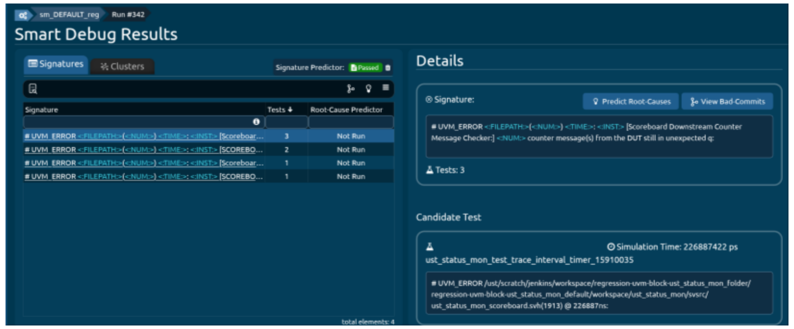
total elements: 2411

Benefit 2: Early Bug Detection (2)

- Smart debug - > Using ML to triage and identify candidate tests to debug.
- Speeds up debug cycles helping in earlier bug detection.



Regression Engine:	Jenkins & VRM
Actions:	179
Start Time:	2025-09-18 - 13:24:42
End Time:	2025-09-18 - 14:00:23
Insertion Timestamp:	2025-09-18 - 13:24:42
Duration:	35m 41s (+4m 34s)
Failure Signatures:	6
Rerunning Status:	Not Run
Jenkins Pipeline:	Passed
Signature Predictor:	Passed 4 Signatures



sm_DEFAULT_reg Run #342

Smart Debug Results

Signatures Clusters Signature Predictor: Passed

Signature	Tests	Root-Cause Predictor
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [Scoreboar...	3	Not Run
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [SCOREBO...	2	Not Run
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [Scoreboar...	1	Not Run
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [SCOREBO...	1	Not Run

total elements: 4

Details

Signature: Predict Root-Causes View Bad-Commits

```
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [Scoreboard Downstream Counter Message Checker:] <:NUM:> counter message(s) from the DUT still in unexpected q:
```

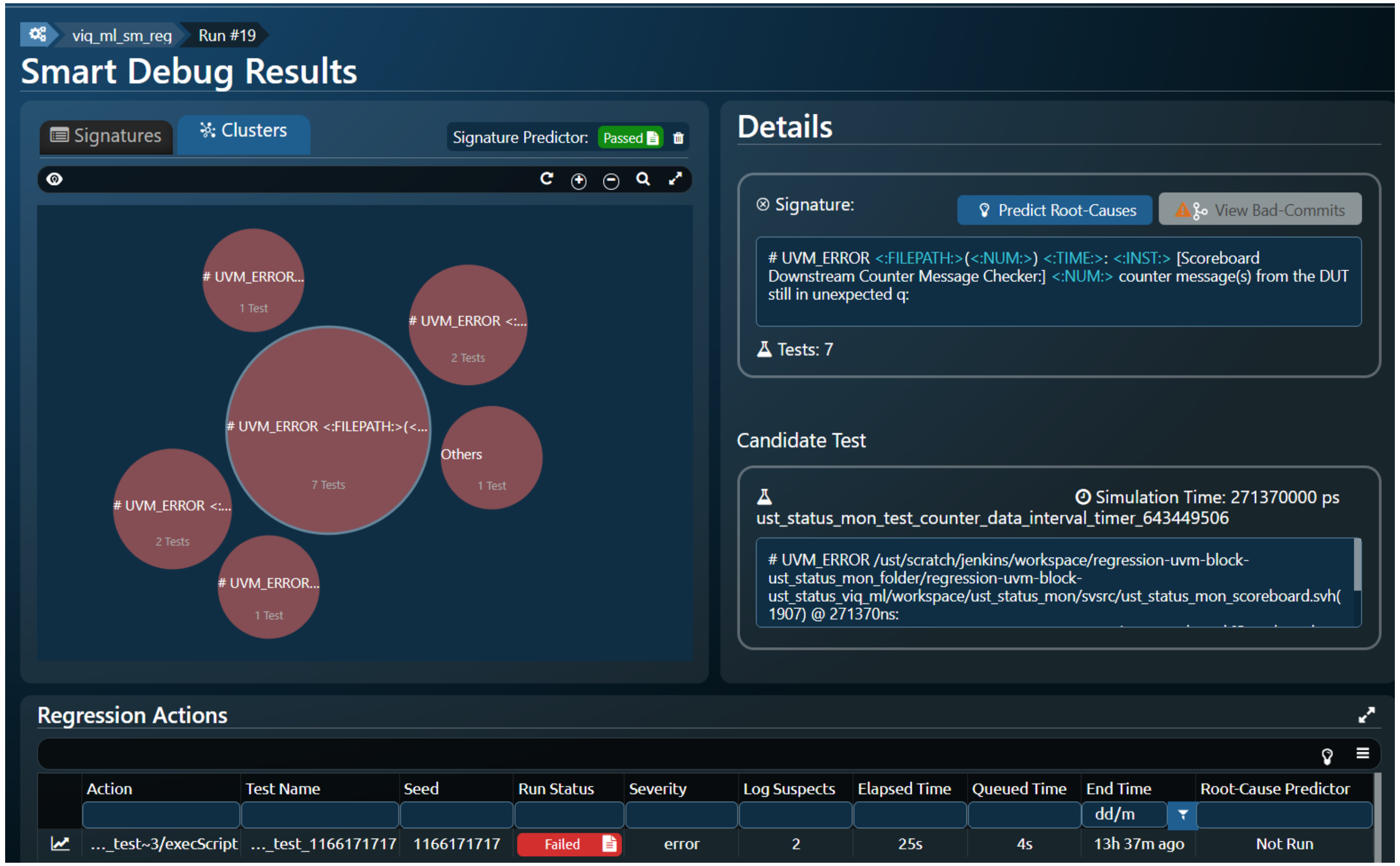
Tests: 3

Candidate Test

Simulation Time: 226887422 ps

ust_status_mon_test_trace_interval_timer_15910035

```
# UVM_ERROR /ust/scratch/jenkins/workspace/regression-uvm-block-ust_status_mon_folder/regression-uvm-block-ust_status_mon_default/workspace/ust_status_mon/svsrc/ust_status_mon_scoreboard.svh(1913) @ 226887ns:
```



Smart Debug Results

Signature Predictor: Passed

Details

Signature: Predict Root-Causes View Bad-Commits

```
# UVM_ERROR <:FILEPATH:>(<:NUM:>) <:TIME:>: <:INST:> [Scoreboard Downstream Counter Message Checker:] <:NUM:> counter message(s) from the DUT still in unexpected q:
```

Tests: 7

Candidate Test

Simulation Time: 271370000 ps

ust_status_mon_test_counter_data_interval_timer_643449506

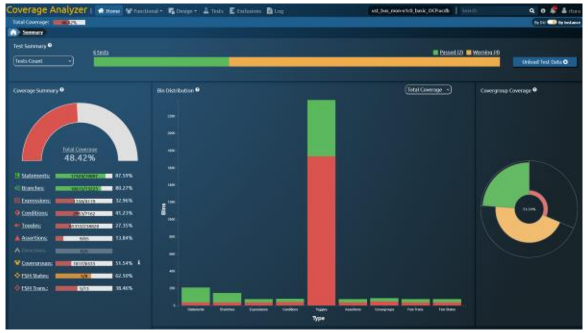
```
# UVM_ERROR /ust/scratch/jenkins/workspace/regression-uvm-block-ust_status_mon_folder/regression-uvm-block-ust_status_viq_ml/workspace/ust_status_mon/svsrc/ust_status_mon_scoreboard.svh(1907) @ 271370ns:
```

Regression Actions

Action	Test Name	Seed	Run Status	Severity	Log Suspects	Elapsed Time	Queued Time	End Time	Root-Cause Predictor
								dd/m	
..._test~3/execScript	..._test_1166171717	1166171717	Failed	error	2	25s	4s	13h 37m ago	Not Run

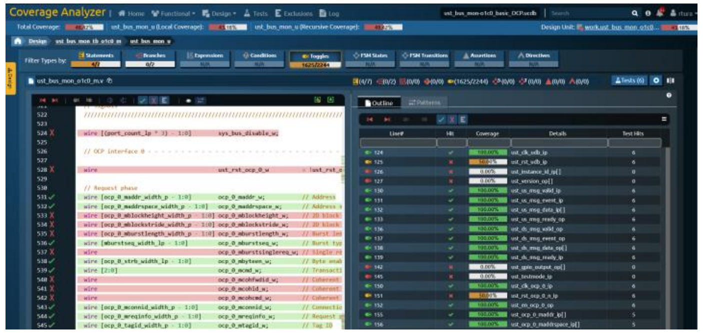
Benefit 3: Comprehensive Coverage Tracking

- **Real-time coverage metrics** across all configurations and all coverage types
- **Interactive coverage** linked and annotated directly to testplan items
- **Gap analysis** between configurations highlights untested scenarios
- **Coverage progression tracking** shows verification maturity over time



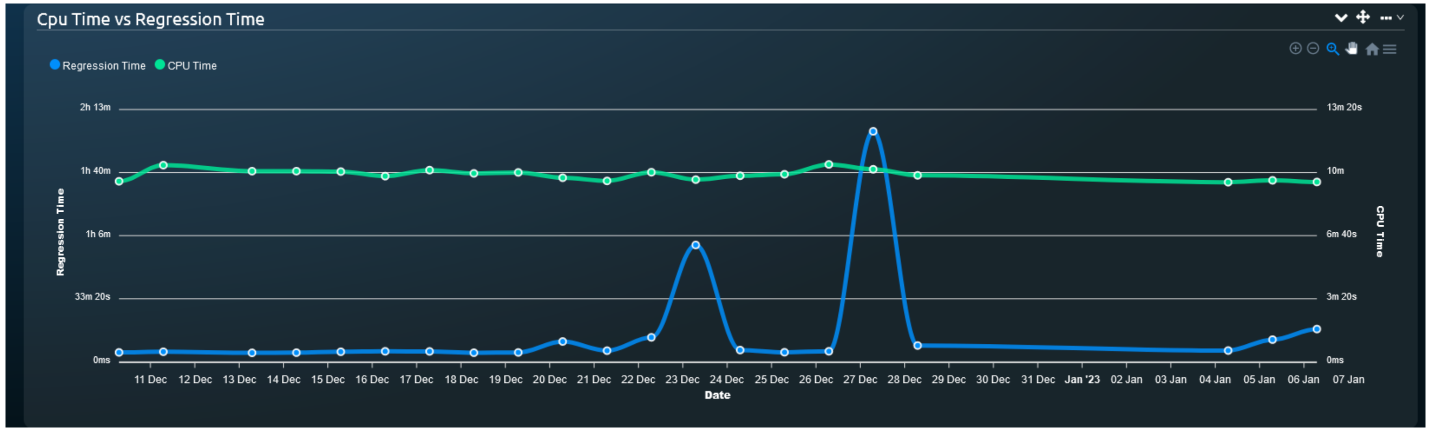
Total Coverage
48.42%

Statements:	87.59%
Branches:	80.27%
Expressions:	32.96%
Conditions:	41.23%
Toggles:	27.35%
Assertions:	13.84%
Covergroups:	51.54%
FSM States:	62.50%
FSM Trans.:	38.46%



Benefit 4: Resource Optimization

- **CPU / Memory utilization tracking** across all tests to identify performance anomalies
- **Test runtime analysis** to identify inefficient simulations
- **Regression execution patterns** reveal infrastructure bottlenecks

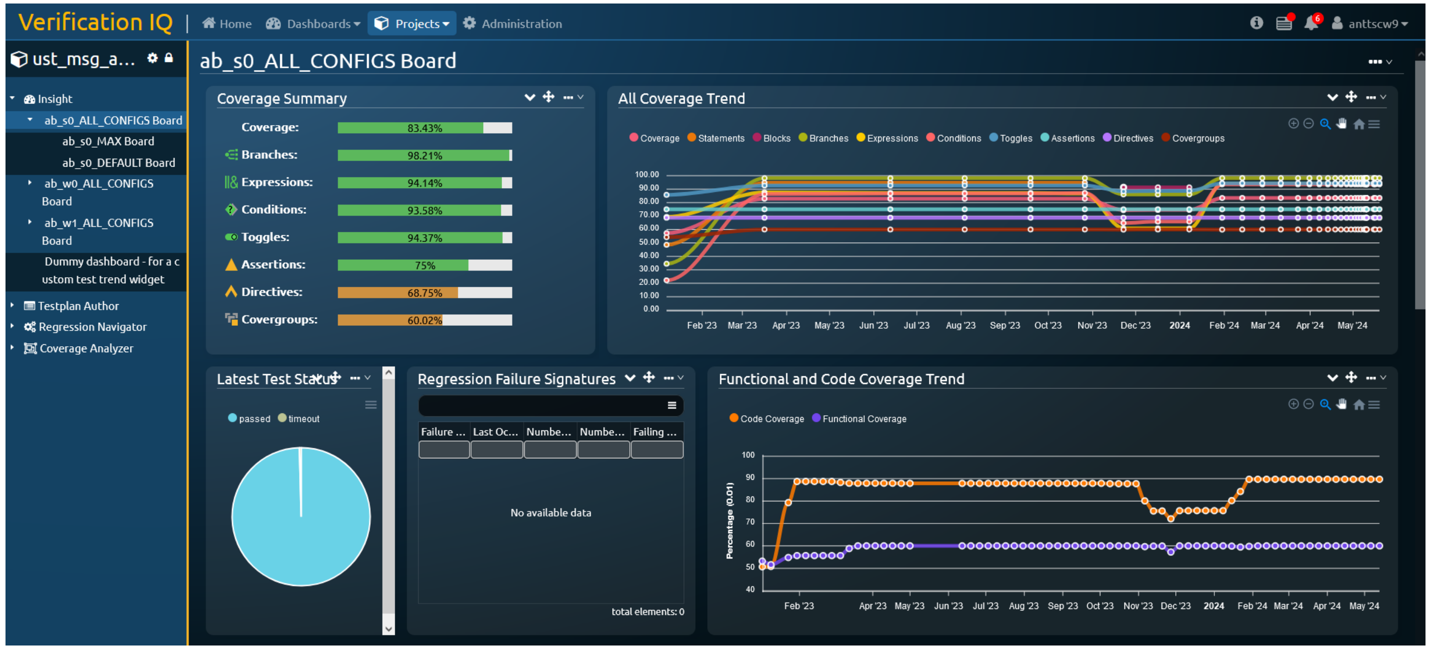


Benefit 5: Data-Driven Decision Making

- **Actionable insights** rather than raw data
- **Interactive filtering** to isolate specific issues or configurations
- **Custom views** for different stakeholders (managers, engineers, leads)
- **Historical trend analysis** for continuous process improvement

Tessent Embedded Analytics Dashboard (1)

- Use Questa Verification IQ (VIQ) extensively for verification management and dashboarding
  - Various metrics: regression status, coverage scores, database stability, CPU runtimes, etc.
  - Each IP has separate Hierarchical dashboard per variant and configuration.

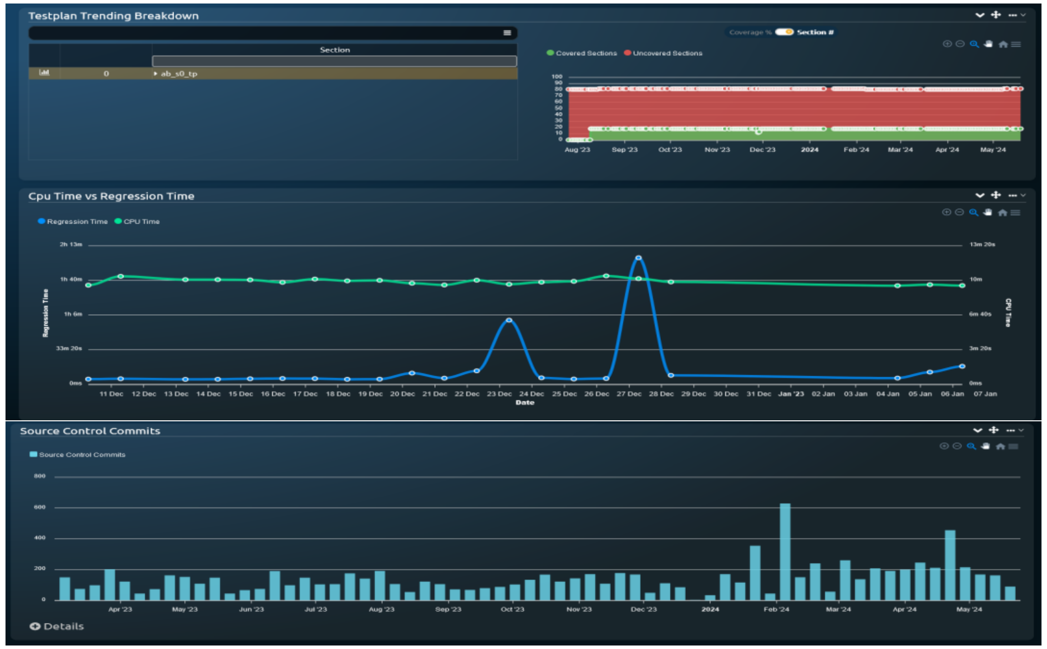


Coverage:	83.43%
Branches:	98.21%
Expressions:	94.14%
Conditions:	93.58%
Toggles:	94.37%
Assertions:	75%
Directives:	68.75%
Covergroups:	60.02%

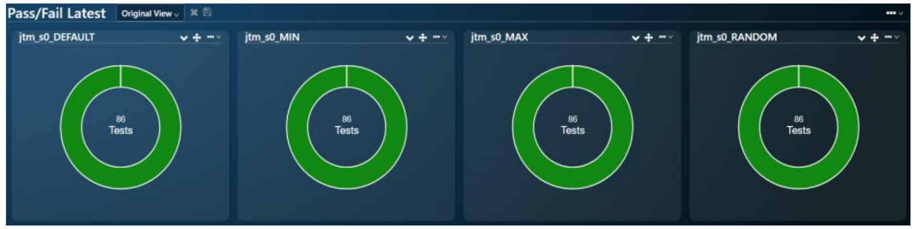
Failure ...	Last Oc...	Numbe...	Numbe...	Failing ...

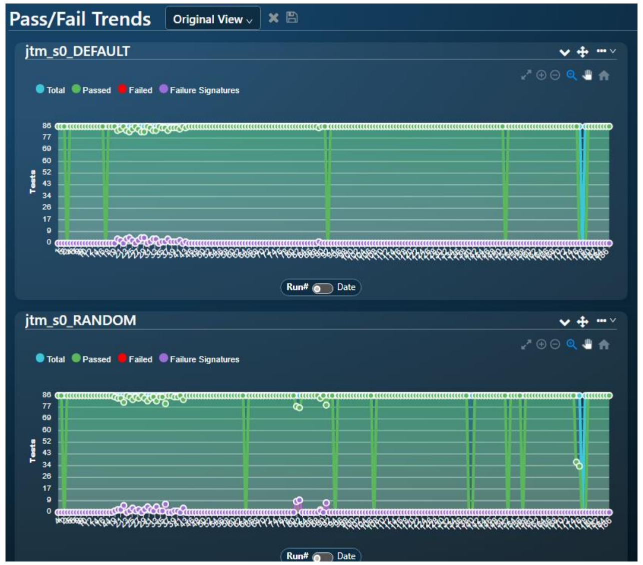
No available data

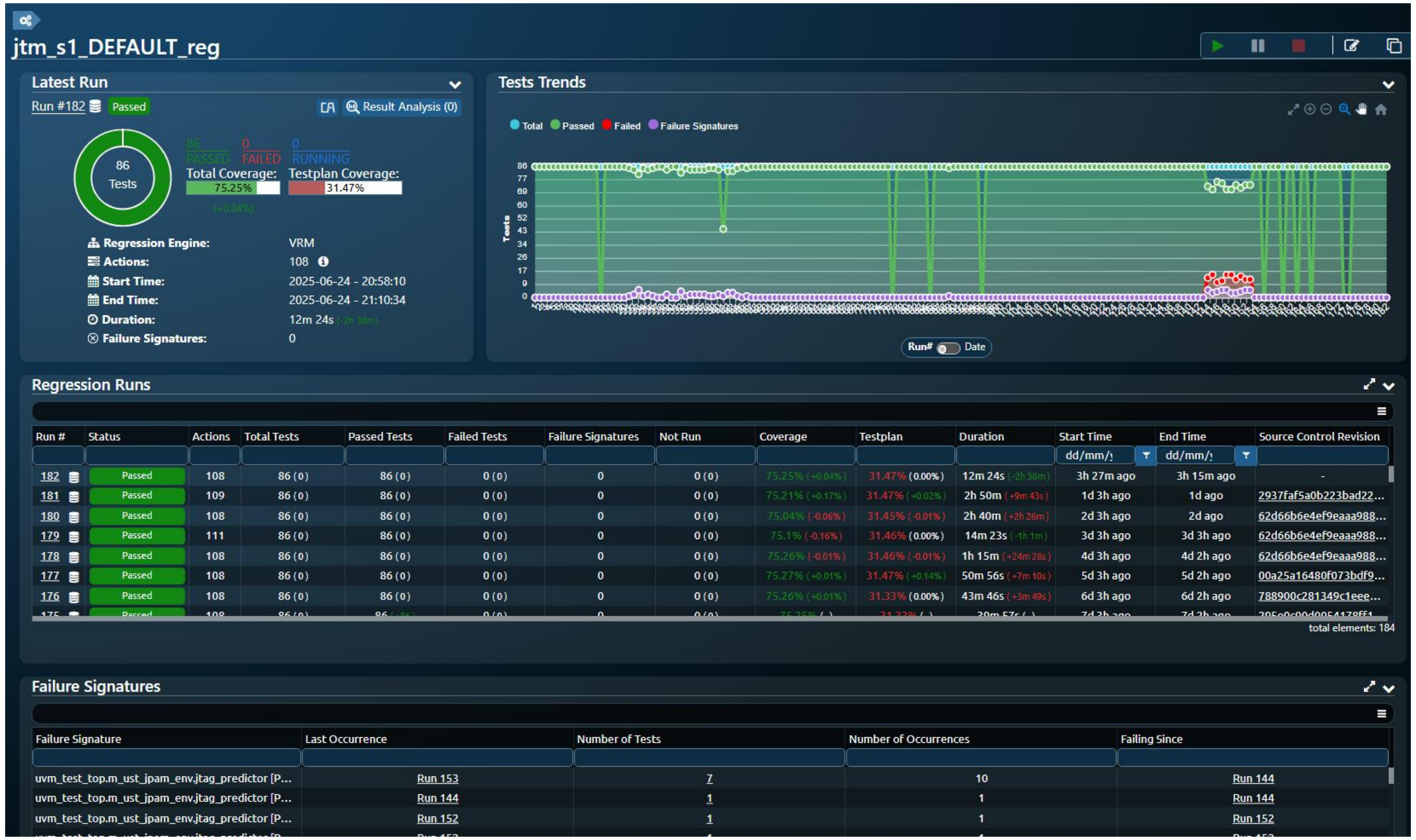
total elements: 0



Tessent Embedded Analytics Dashboard (2)







jtm_s1_DEFAULT_reg

Latest Run

Run #182 Passed — Result Analysis (0)

86 Tests — 86 PASSED, 0 FAILED, 0 RUNNING

Total Coverage: 75.25% (+0.04%)

Testplan Coverage: 31.47%

Regression Engine:	VRM
Actions:	108
Start Time:	2025-06-24 - 20:58:10
End Time:	2025-06-24 - 21:10:34
Duration:	12m 24s (-2h 38m)
Failure Signatures:	0

Regression Runs

Run #	Status	Actions	Total Tests	Passed Tests	Failed Tests	Failure Signatures	Not Run	Coverage	Testplan	Duration	Start Time	End Time	Source Control Revision
182	Passed	108	86 (0)	86 (0)	0 (0)	0	0 (0)	75.25% (+0.04%)	31.47% (0.00%)	12m 24s (-2h 38m)	3h 27m ago	3h 15m ago	-
181	Passed	109	86 (0)	86 (0)	0 (0)	0	0 (0)	75.21% (+0.17%)	31.47% (+0.02%)	2h 50m (+9m 43s)	1d 3h ago	1d ago	2937faf5a0b223bad22...
180	Passed	108	86 (0)	86 (0)	0 (0)	0	0 (0)	75.04% (-0.06%)	31.45% (-0.01%)	2h 40m (+2h 26m)	2d 3h ago	2d ago	62d66b6e4ef9eaaa988...
179	Passed	111	86 (0)	86 (0)	0 (0)	0	0 (0)	75.1% (-0.16%)	31.46% (0.00%)	14m 23s (-1h 1m)	3d 3h ago	3d 3h ago	62d66b6e4ef9eaaa988...
178	Passed	108	86 (0)	86 (0)	0 (0)	0	0 (0)	75.26% (-0.01%)	31.46% (-0.01%)	1h 15m (+24m 28s)	4d 3h ago	4d 2h ago	62d66b6e4ef9eaaa988...
177	Passed	108	86 (0)	86 (0)	0 (0)	0	0 (0)	75.27% (+0.01%)	31.47% (+0.14%)	50m 56s (+7m 10s)	5d 3h ago	5d 2h ago	00a25a16480f073bdf9...
176	Passed	108	86 (0)	86 (0)	0 (0)	0	0 (0)	75.26% (+0.01%)	31.33% (0.00%)	43m 46s (+3m 49s)	6d 3h ago	6d 2h ago	788900c281349c1eee...

total elements: 184

Failure Signatures

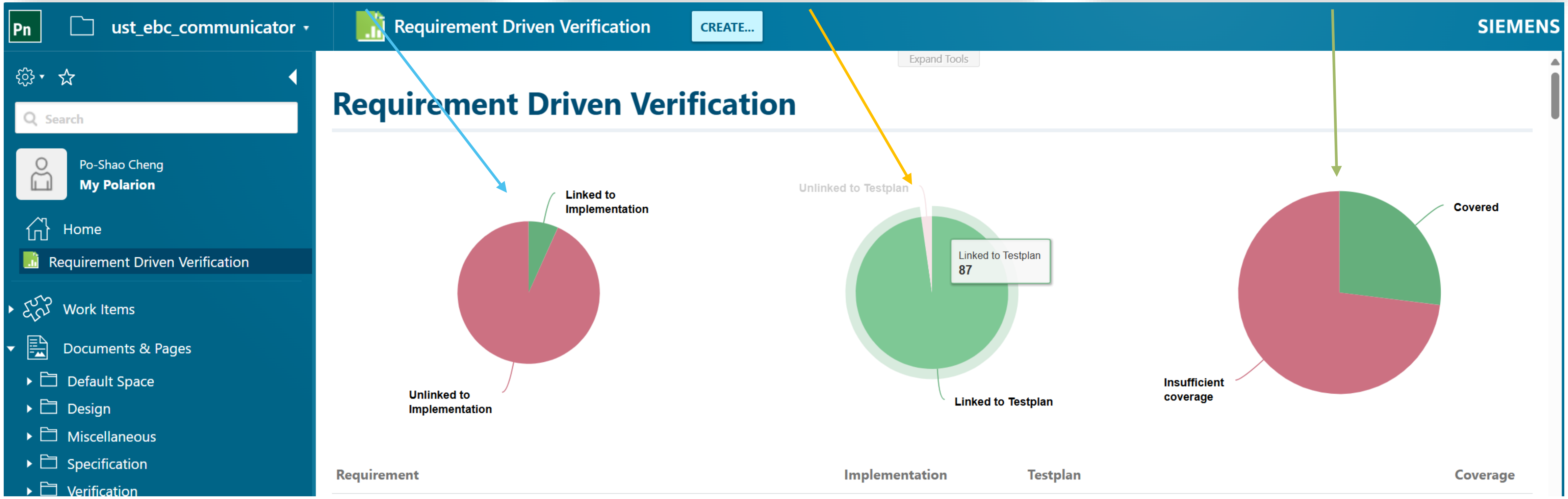
Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
uvm_test_top.m_ust_jpam_env.jtag_predictor [P...	Run 153	7	10	Run 144
uvm_test_top.m_ust_jpam_env.jtag_predictor [P...	Run 144	1	1	Run 144
uvm_test_top.m_ust_jpam_env.jtag_predictor [P...	Run 152	1	1	Run 152

Requirement Traceability Dashboards

01 RTL Implementation
Requirements linked to the actual RTL implementation e.g. VHDL, Verilog using pragmas.

02 Test Plan
Requirements linked to the Test Plan.

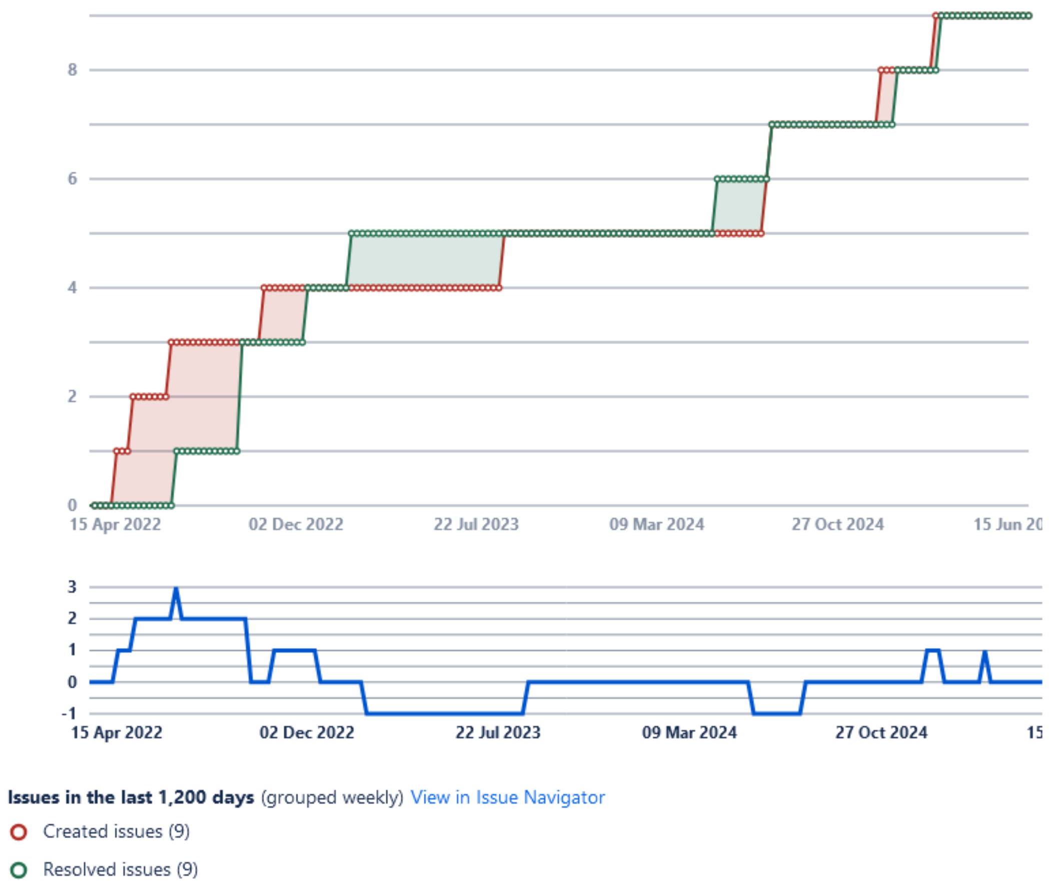
03 Coverage Results
Shows coverage number for each test plan items that the requirements are linked to.



Requirement Driven Verification

Bug / Issue Tracking

- Using JIRA for issue/bug tracking
- Will integrate with Verification Management tool



Issues in the last 1,200 days (grouped weekly) View in Issue Navigator

- Created issues (9)
- Resolved issues (9)

Early Bug Detection

How our coverage and dashboards help

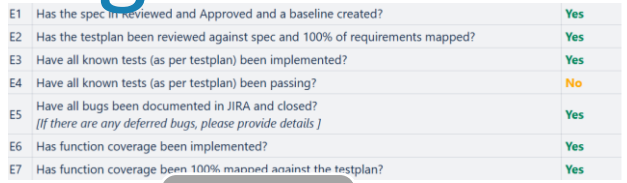
TO FIND BUGS EARLY – SPECIALLY DURING THE PRE-SILICON STAGE

CREATE ROBUST DESIGNS – HIGH QUALITY DESIGN

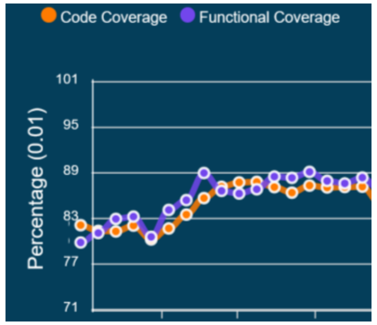
DEVELOPMENT TIME IS FASTER

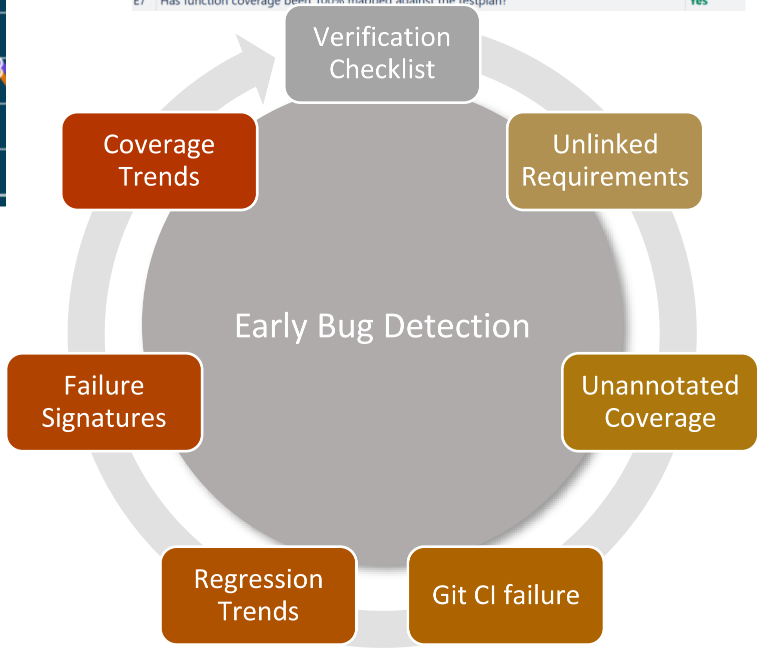
BETTER RELATIONSHIP WITH CUSTOMERS

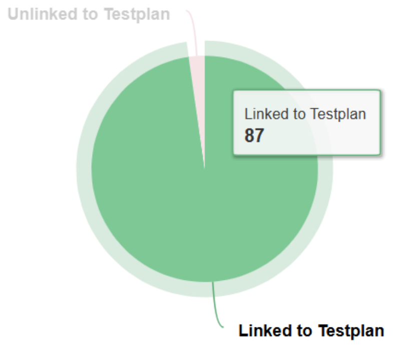
Early Indicators of Bugs

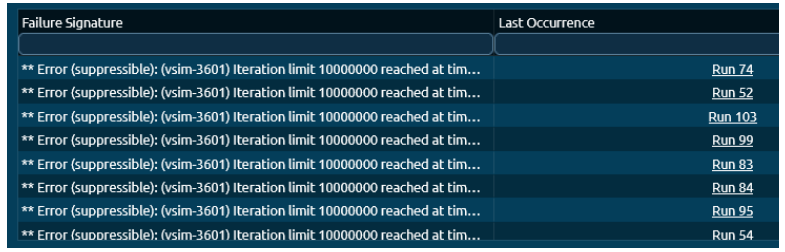


E1	Has the spec in Reviewed and Approved and a baseline created?	Yes
E2	Has the testplan been reviewed against spec and 100% of requirements mapped?	Yes
E3	Have all known tests (as per testplan) been implemented?	Yes
E4	Have all known tests (as per testplan) been passing?	No
E5	Have all bugs been documented in JIRA and closed? *[If there are any deferred bugs, please provide details]*	Yes
E6	Has function coverage been implemented?	Yes
E7	Has function coverage been 100% mapped against the testplan?	Yes

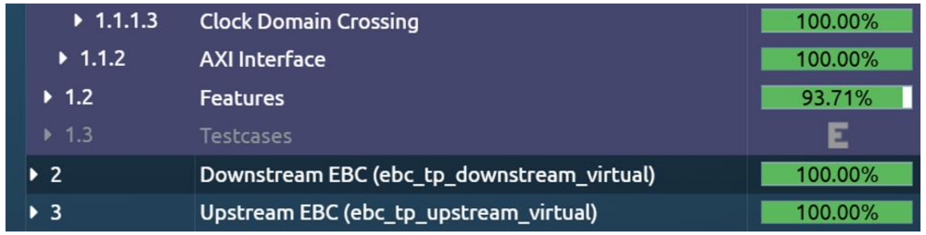




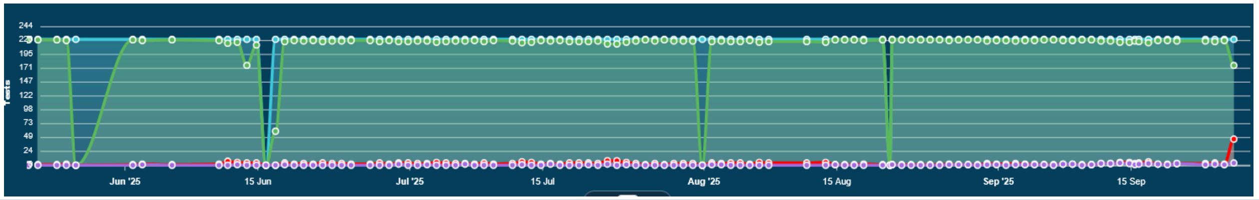


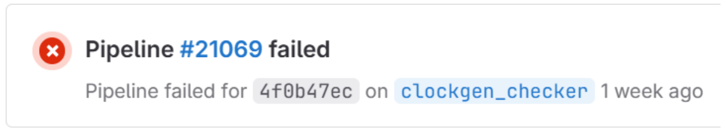


Failure Signature	Last Occurrence
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 74
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 52
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 103
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 99
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 83
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 84
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 95
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 54



1.1.1.3	Clock Domain Crossing	100.00%
1.1.2	AXI Interface	100.00%
1.2	Features	93.71%
1.3	Testcases	E
2	Downstream EBC (ebc_tp_downstream_virtual)	100.00%
3	Upstream EBC (ebc_tp_upstream_virtual)	100.00%



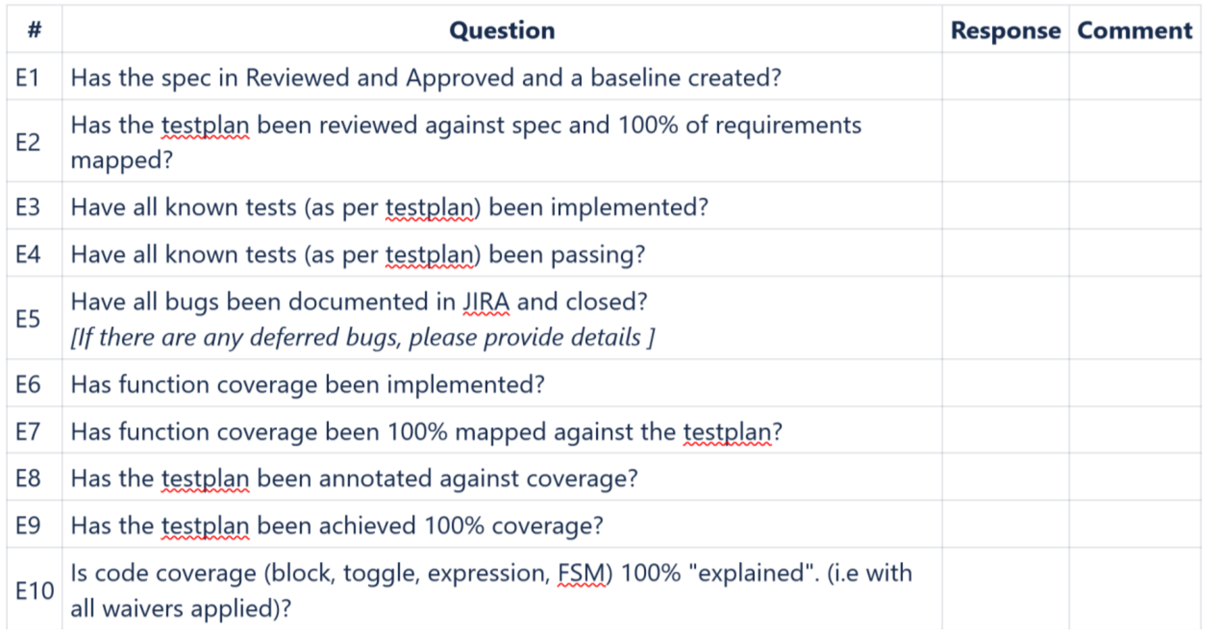


Pipeline #21069 failed

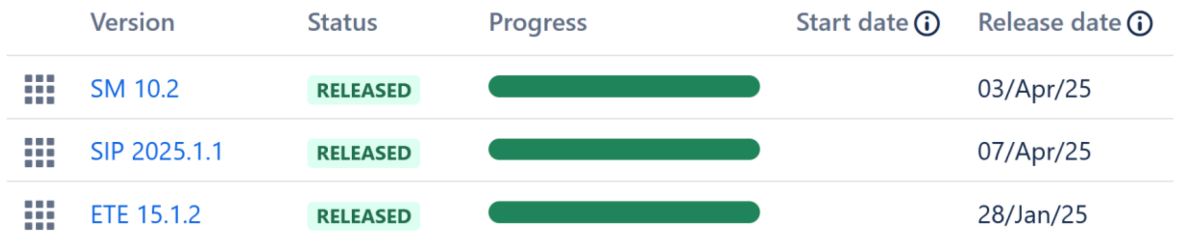
Pipeline failed for `4f0b47ec` on `clockgen_checker` 1 week ago

Verification checklists

- Final link in the methodology
- Created early in the development process of the release
- Completed prior to code freeze
- Reviewed and approved prior to release
- Captures all aspects of the methodology
- Progress bar in Jira is used to track checklist



#	Question	Response	Comment
E1	Has the spec in Reviewed and Approved and a baseline created?		
E2	Has the testplan been reviewed against spec and 100% of requirements mapped?		
E3	Have all known tests (as per testplan) been implemented?		
E4	Have all known tests (as per testplan) been passing?		
E5	Have all bugs been documented in JIRA and closed? *[If there are any deferred bugs, please provide details]*		
E6	Has function coverage been implemented?		
E7	Has function coverage been 100% mapped against the testplan?		
E8	Has the testplan been annotated against coverage?		
E9	Has the testplan been achieved 100% coverage?		
E10	Is code coverage (block, toggle, expression, FSM) 100% "explained". (i.e with all waivers applied)?		

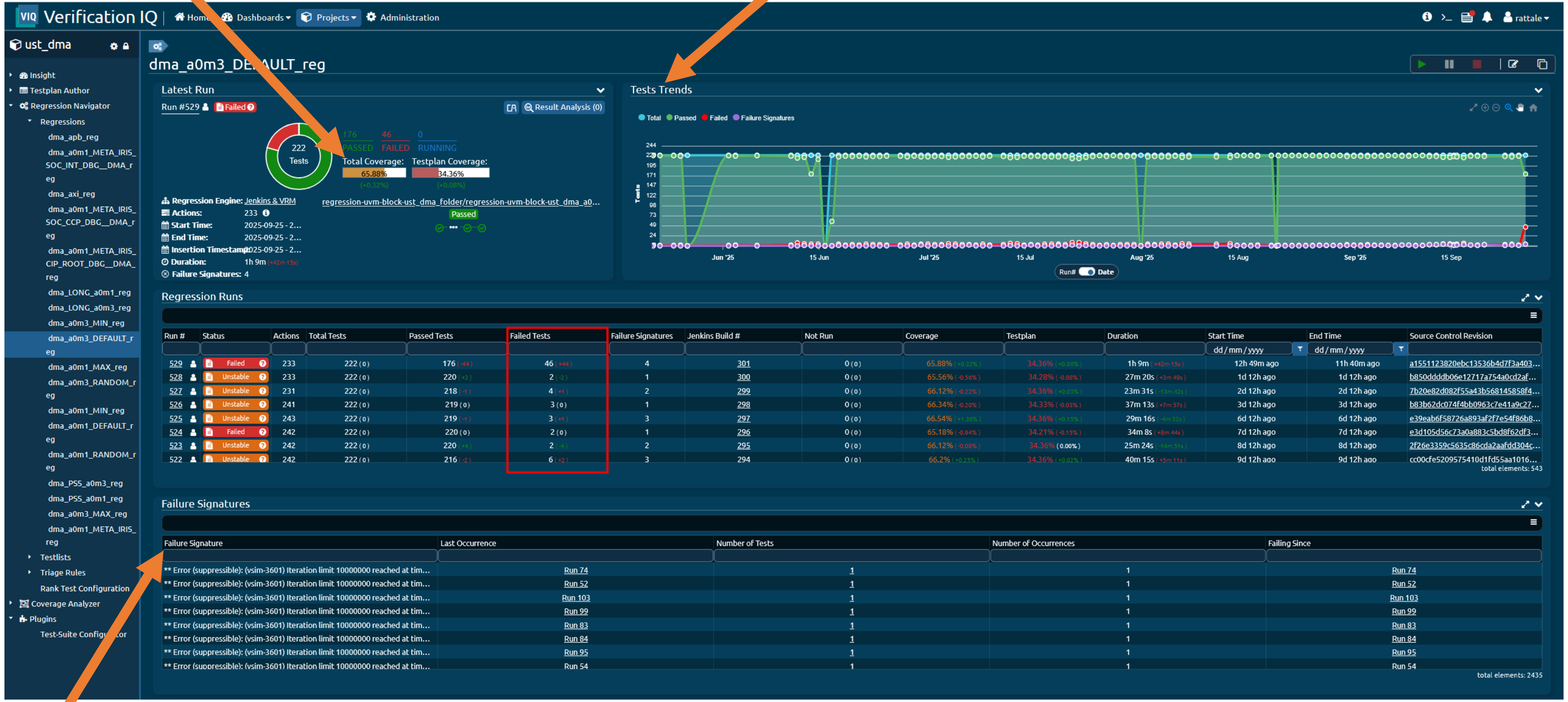


Version	Status	Progress	Start date	Release date
SM 10.2	RELEASED	🟩		03/Apr/25
SIP 2025.1.1	RELEASED	🟩		07/Apr/25
ETE 15.1.2	RELEASED	🟩		28/Jan/25

Regression Trends & Failing Signatures

Coverage rate for the regression

Pass/fail trends of the regressions



dma_a0m3_DEFAULT_reg

Latest Run

Run #529 Failed

222 Tests

176 PASSED | 46 FAILED | 0 RUNNING

Total Coverage: 65.88% (+0.32%)

Testplan Coverage: 34.36% (+0.08%)

Regression Engine: Jenkins & VRM
Actions: 233
Start Time: 2025-09-25 - 2...
End Time: 2025-09-25 - 2...
Insertion Timestamp: 2025-09-25 - 2...
Duration: 1h 9m (+42m 15s)
Failure Signatures: 4

regression-uvm-block-ust_dma_folder/regression-uvm-block-ust_dma_a0...
Passed

Regression Runs

Run #	Status	Actions	Total Tests	Passed Tests	Failed Tests	Failure Signatures	Jenkins Build #	Not Run	Coverage	Testplan	Duration	Start Time	End Time	Source Control Revision
												dd / mm / yyyy	dd / mm / yyyy	
529	Failed	233	222 (0)	176 (-44)	46 (+44)	4	301	0 (0)	65.88% (+0.32%)	34.36% (+0.08%)	1h 9m (+42m 15s)	12h 49m ago	11h 40m ago	a1551123820ebc13536b4d7f3a403...
528	Unstable	233	222 (0)	220 (+2)	2 (-2)	1	300	0 (0)	65.56% (-0.56%)	34.28% (-0.08%)	27m 20s (+3m 49s)	1d 12h ago	1d 12h ago	b850ddddb06e12717a754a0cd2af...
527	Unstable	231	222 (0)	218 (-1)	4 (+1)	2	299	0 (0)	66.12% (-0.22%)	34.36% (+0.03%)	23m 31s (-13m 42s)	2d 12h ago	2d 12h ago	7b20e82d082f55a43b568145858f4...
526	Unstable	241	222 (0)	219 (0)	3 (0)	1	298	0 (0)	66.34% (-0.20%)	34.33% (-0.03%)	37m 13s (+7m 57s)	3d 12h ago	3d 12h ago	b83b62dc074f4bb0963c7e41a9c27...
525	Unstable	243	222 (0)	219 (-1)	3 (+1)	3	297	0 (0)	66.54% (+1.36%)	34.36% (+0.15%)	29m 16s (-4m 52s)	6d 12h ago	6d 12h ago	e39eab6f58726a893af2f7e54f86b8...
524	Failed	242	222 (0)	220 (0)	2 (0)	1	296	0 (0)	65.18% (-0.94%)	34.21% (-0.15%)	34m 8s (+8m 44s)	7d 12h ago	7d 12h ago	e3d105d56c73a0a883c5bd8f62df3...
523	Unstable	242	222 (0)	220 (+4)	2 (-4)	2	295	0 (0)	66.12% (-0.08%)	34.36% (0.00%)	25m 24s (-14m 51s)	8d 12h ago	8d 12h ago	2f26e3359c5635c86cda2aafdd304c...
522	Unstable	242	222 (0)	216 (-2)	6 (+2)	3	294	0 (0)	66.2% (+0.25%)	34.36% (+0.02%)	40m 15s (+5m 11s)	9d 12h ago	9d 12h ago	cc00cfe5209575410d1fd55aa1016...

total elements: 543

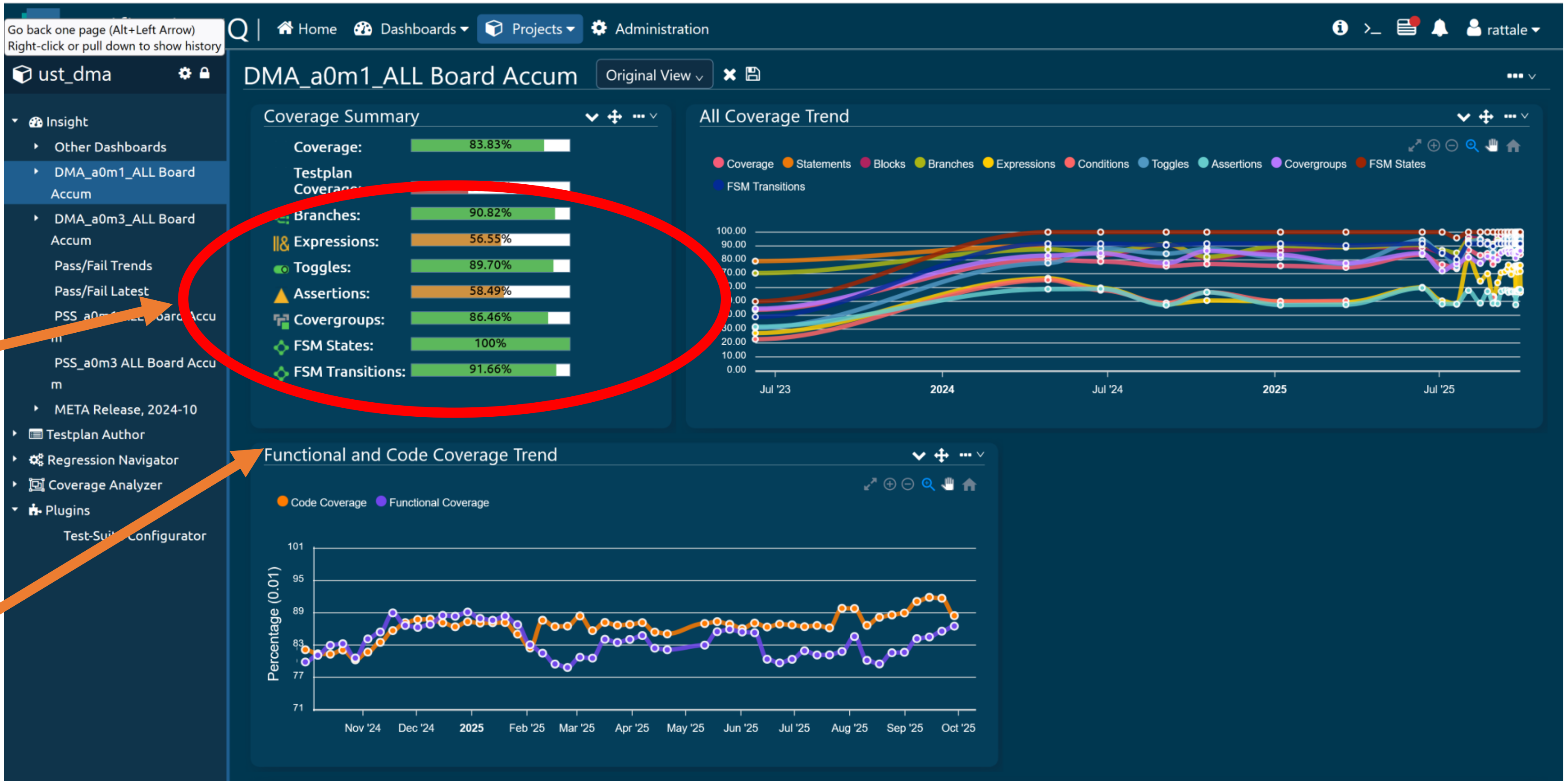
Failure Signatures

Failure Signature	Last Occurrence	Number of Tests	Number of Occurrences	Failing Since
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 74	1	1	Run 74
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 52	1	1	Run 52
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 103	1	1	Run 103
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 99	1	1	Run 99
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 83	1	1	Run 83
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 84	1	1	Run 84
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 95	1	1	Run 95
** Error (suppressible): (vsim-3601) Iteration limit 10000000 reached at tim...	Run 54	1	1	Run 54

total elements: 2435

Error messages seen in the regression

Coverage Trends



Coverage Summary

Coverage:	83.83%
Testplan Coverage:	
Branches:	90.82%
Expressions:	56.55%
Toggles:	89.70%
Assertions:	58.49%
Covergroups:	86.46%
FSM States:	100%
FSM Transitions:	91.66%

Helpful to identify which areas of the code is untested

Fluctuations in coverage can help identify bugs

Summary

Automated & Configurable Flow: Manages complex IP verification with cross-platform automation.

Real-time Visibility: Interactive dashboards enable early bug detection and data-driven decisions.

Enhanced Coverage: "Traffic Light System" for parameter-aware functional coverage.

Integrated Traceability: Seamless collaboration via JIRA and requirement management integration.

Optimized Resources: Monitors usage (CPU/memory, runtime) for efficiency.

Efficient Techniques: Leverages randomization, coverage merging, and automated data extraction.

Questions?