Digital Verification

Budi Murty

Aug 1, 2023

cadence®

Introduction

Budi has been working with Cadence since 2006 started as an Application Engineer. He has wide experience on verification technologies along with the Cadence platforms like – Xcelium, vManager and Safety. He has worked in closed collaboration with different customers around the globe to optimize the verification cycle in terms of regression throughput, automation, coverage closure, etc. Currently, he is working as Sr. AE Manager managing AE team working on different Cadence verification tools.

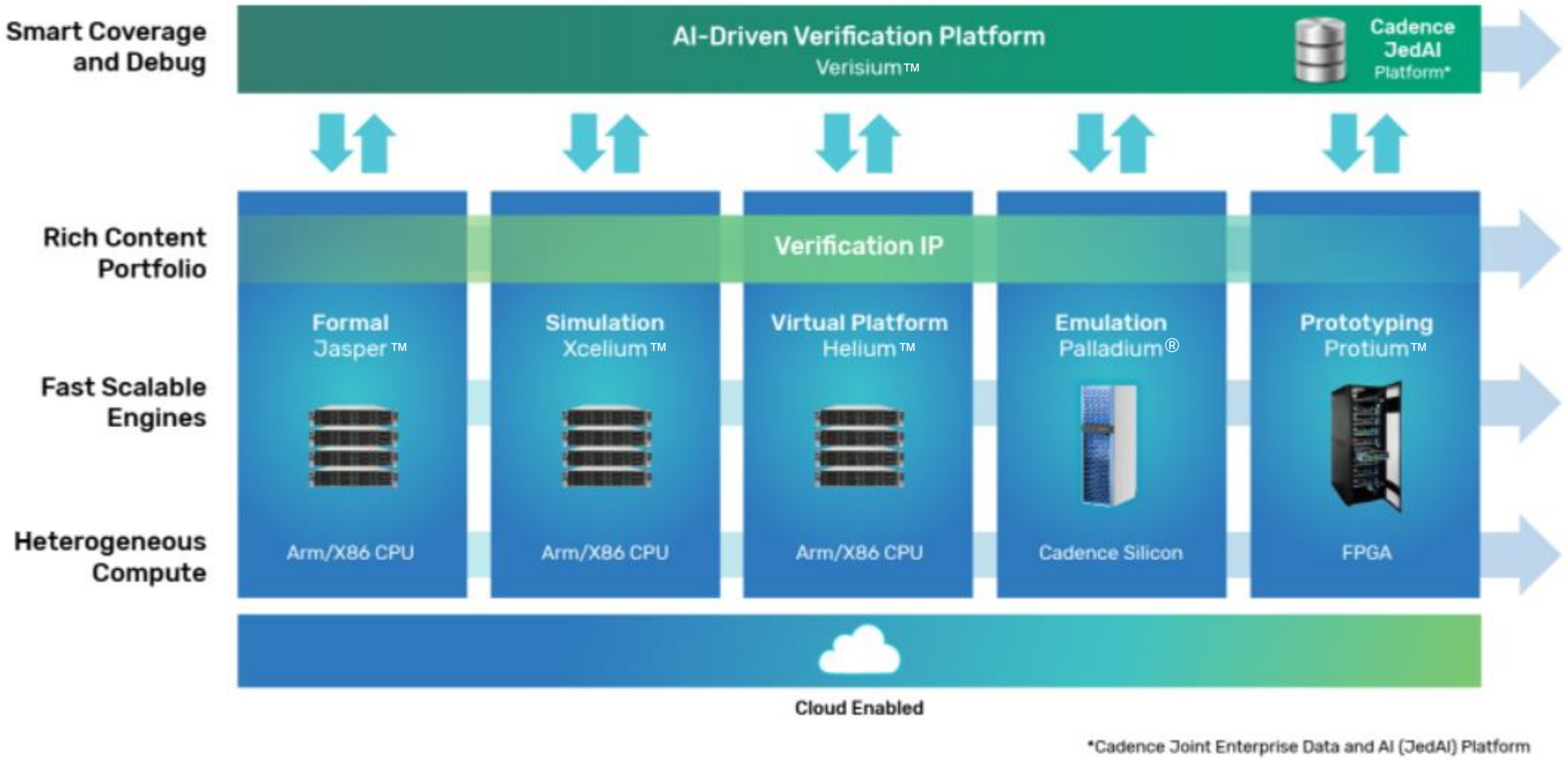
AICTE | arm Education | life.augmented

Supported by cadence

Agenda

- Verification Platform - Overview
- Functional Verification Methodologies
- Verification Environment
- Metrics Driven Verification
- Gate Level Simulation
- Debugging Methods
- Formal Verification
- Regression Environment

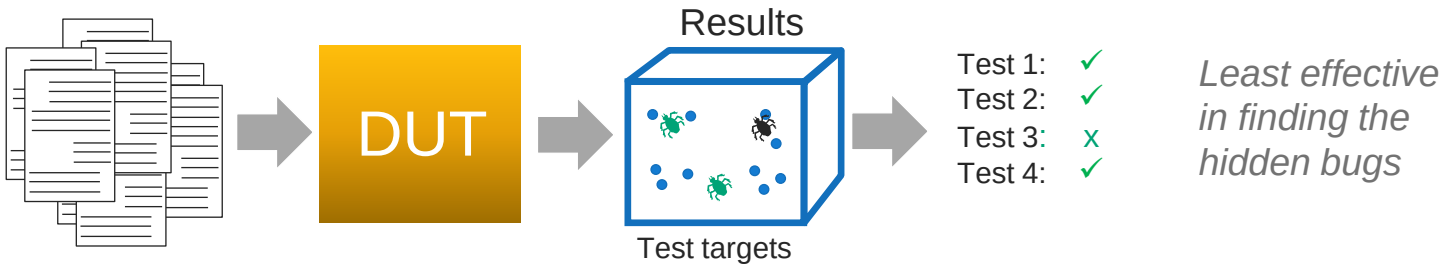
Verification Platform - Overview



*Cadence Joint Enterprise Data and AI (JedAI) Platform

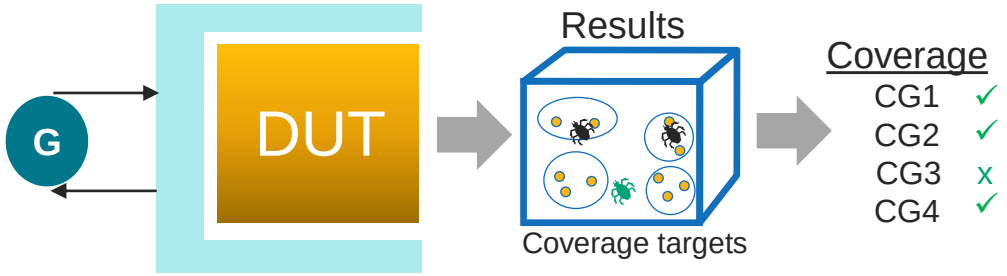
Functional Verification Methodologies

Directed Tests Driven



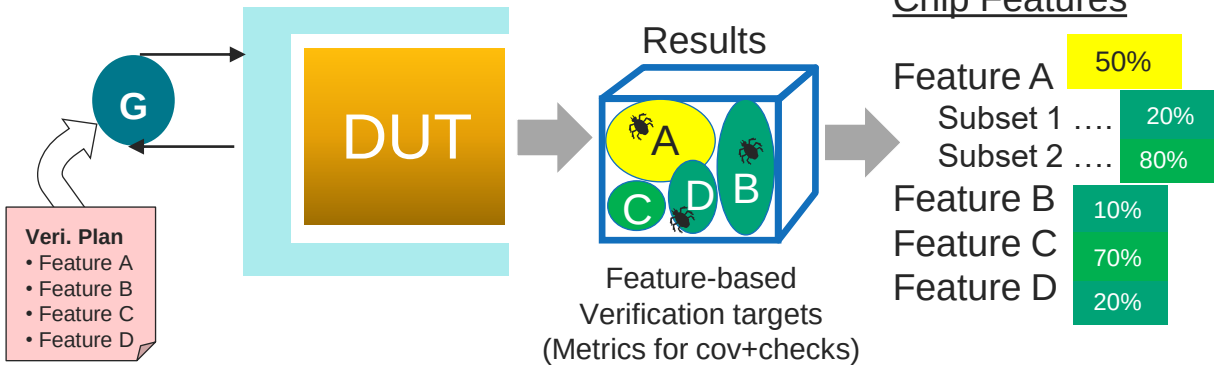
Least effective in finding the hidden bugs

Coverage Driven



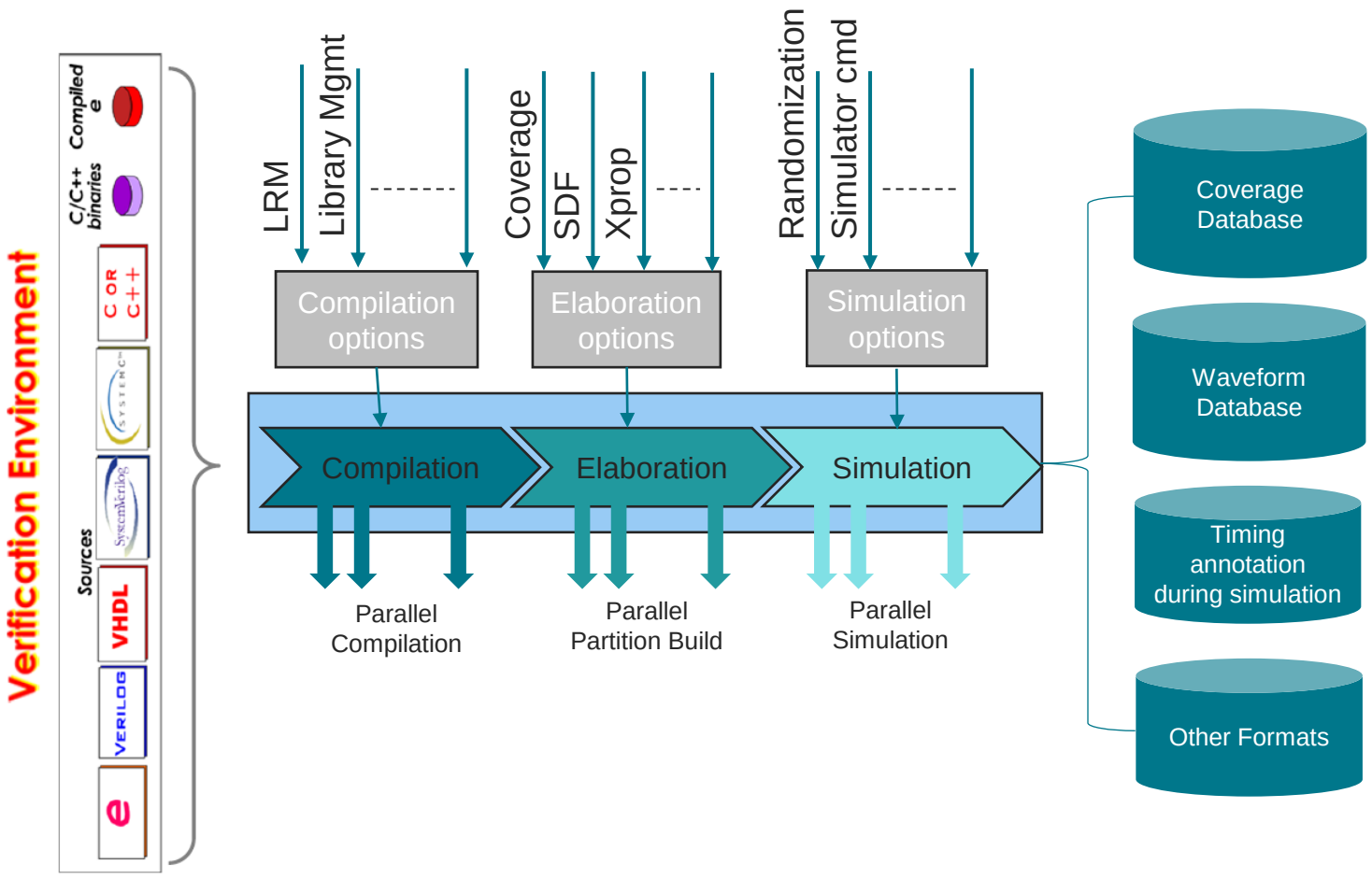
Adds quality and productivity, but difficult to estimate completion

Metric Driven

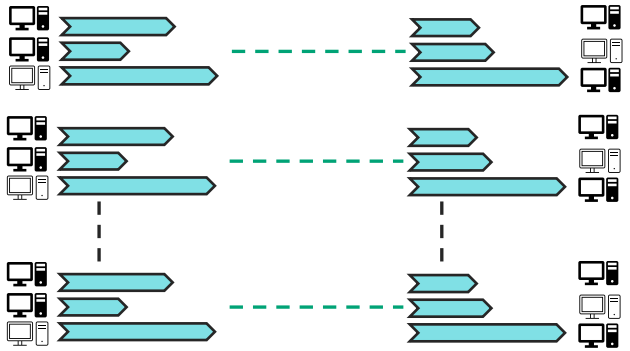


Feature-based plan with extended metrics enables efficient and accurate project closure

Verification Environment

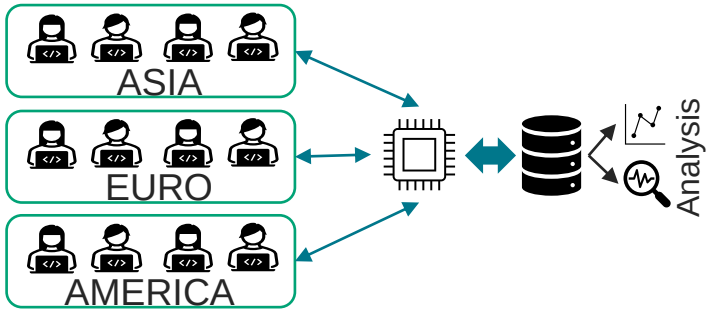


Execution Environment



DRM – Farms are used to execute such heavy computational tasks

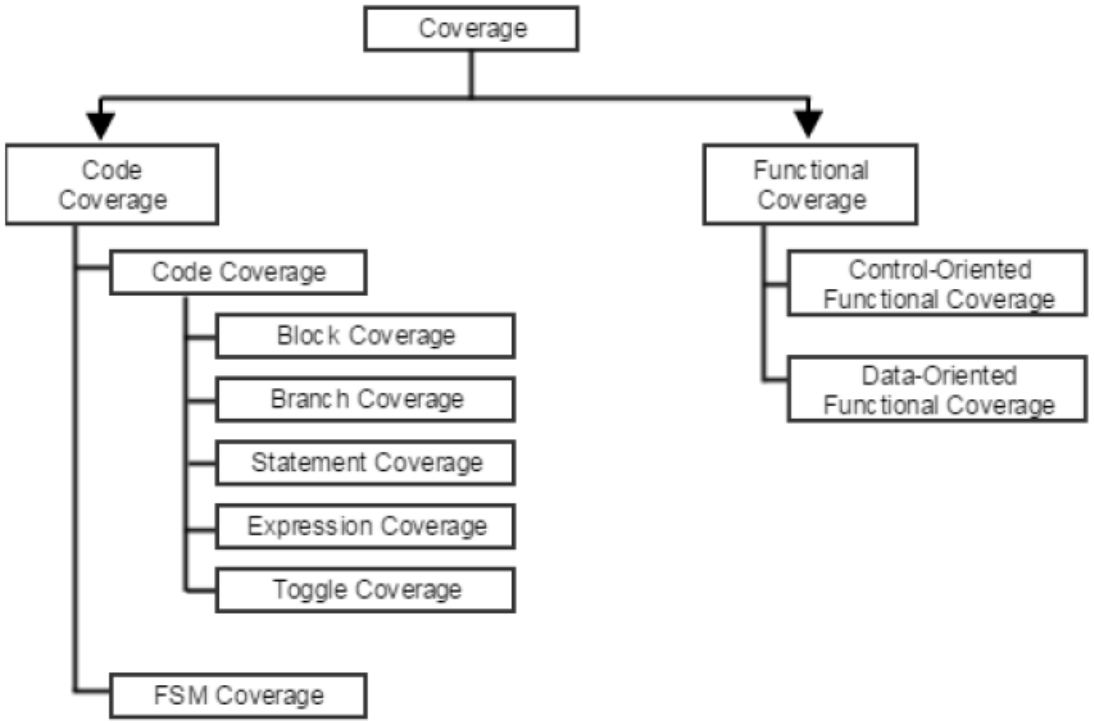
Global Connect



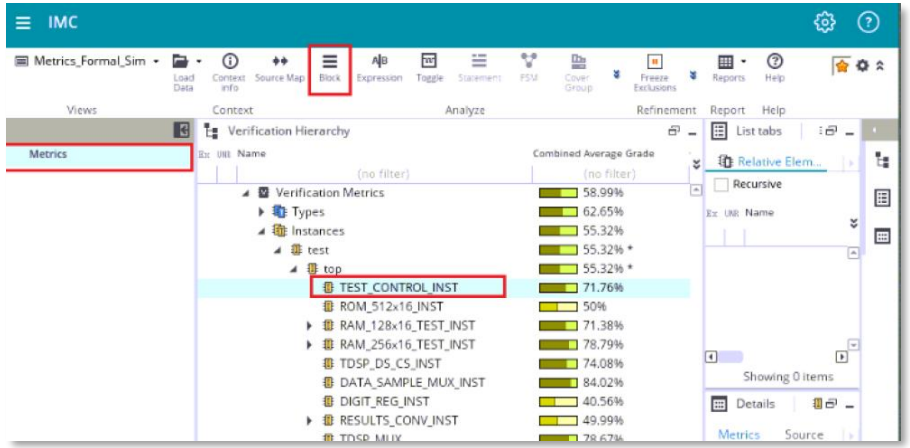
A powerful mechanism is needed to connect and collect data from global users.

Metrics-Driven Verification

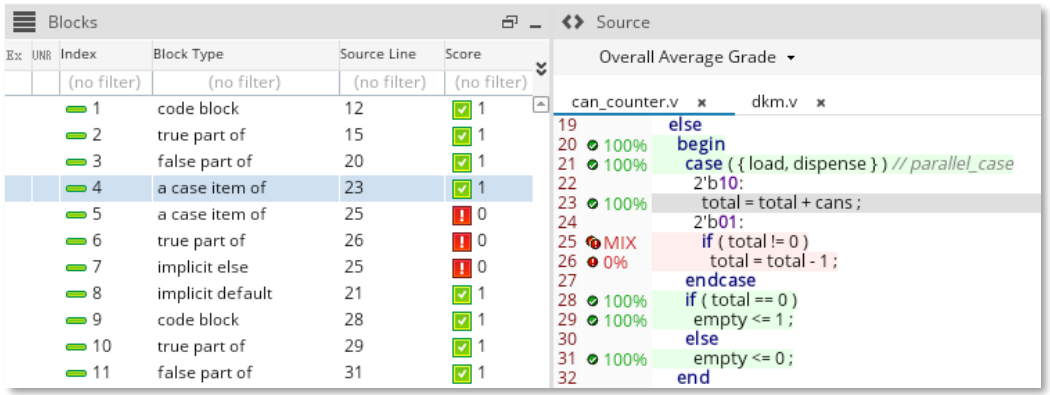
- Code coverage
  - This method assesses how well the test cases test the intended behavior
  - The extent to which they execute the design



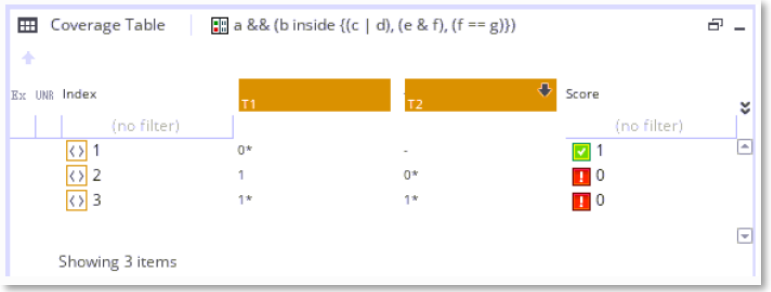
Coverage Hierarchy



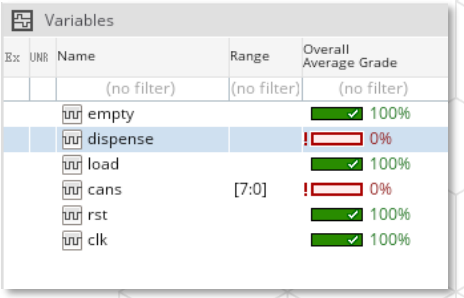
Block Coverage



Expression Coverage

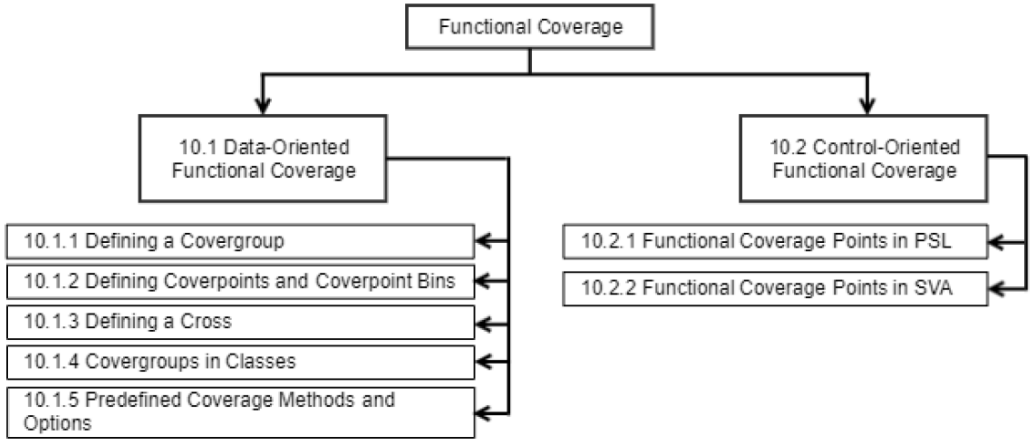


Toggle Coverage

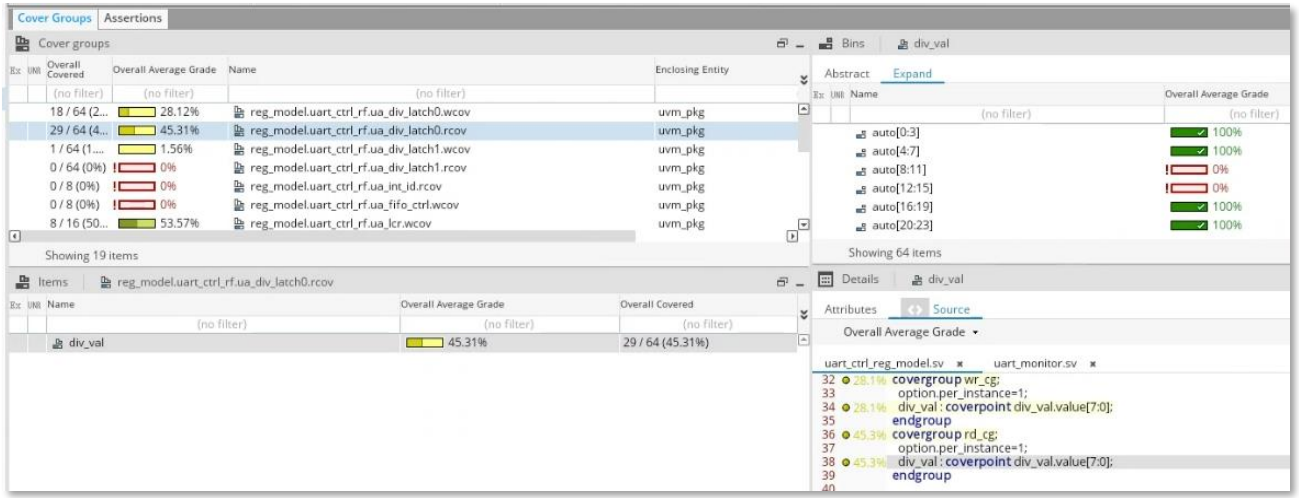


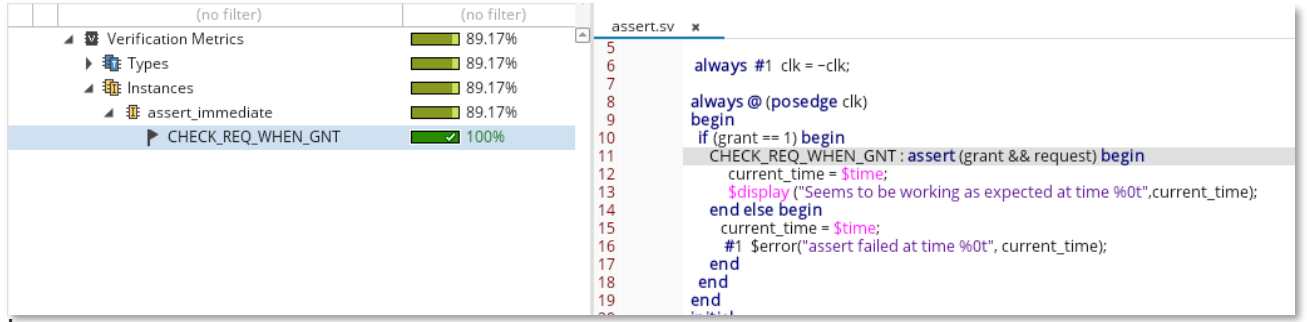
Metrics-Driven Verification

- Functional coverage
  - Focuses on the functional aspects of a design
  - Helps in verification goals set by a test plan



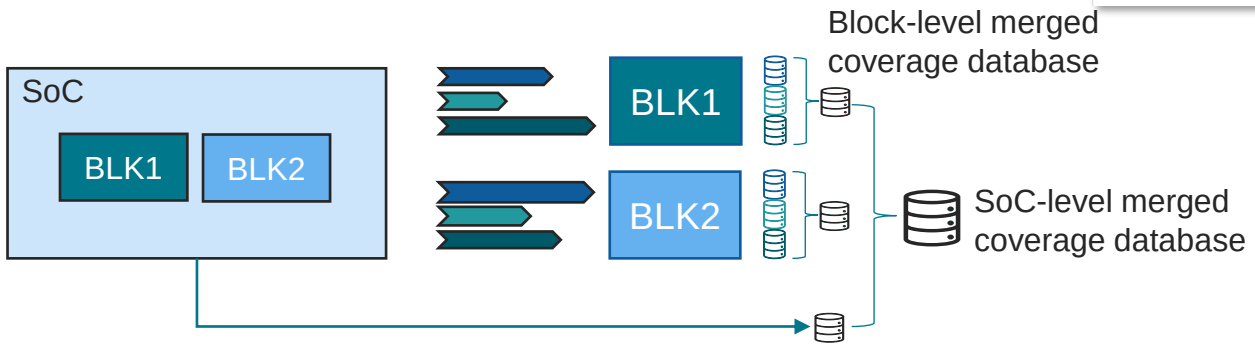
Covergroup Coverage





Assertion Coverage

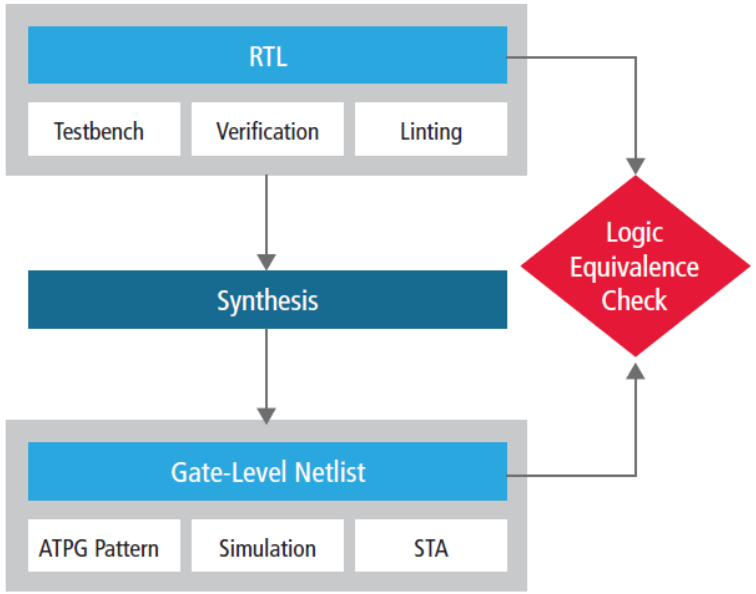
Merging of Coverage Database



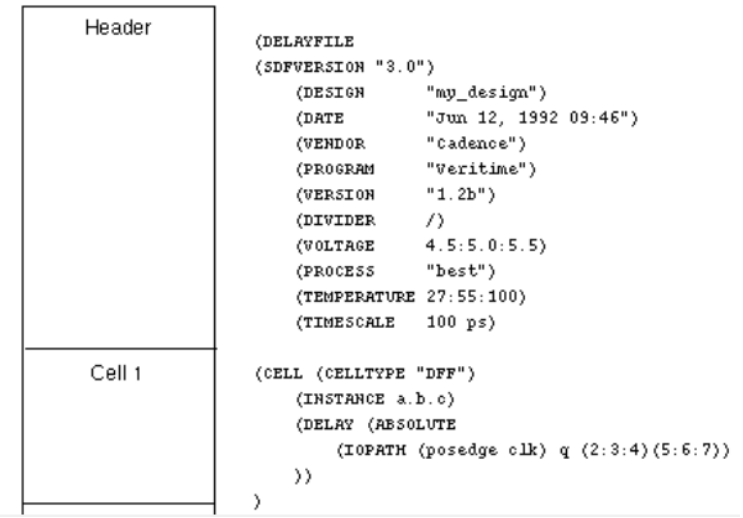
Gate-Level Simulation

- Simulation of a netlist, with or without gate delays
- To ensure that the desired functionality is not lost in the translation from high-level RTL to low-level gate implementation
- GLS is done after the synthesis of the RTL code or post-P&R
- Timing Annotation
  - Specific Block
  - SDF (Standard Delay Format)
- SDF Content
  - IO PATH
  - TIMING CHECK

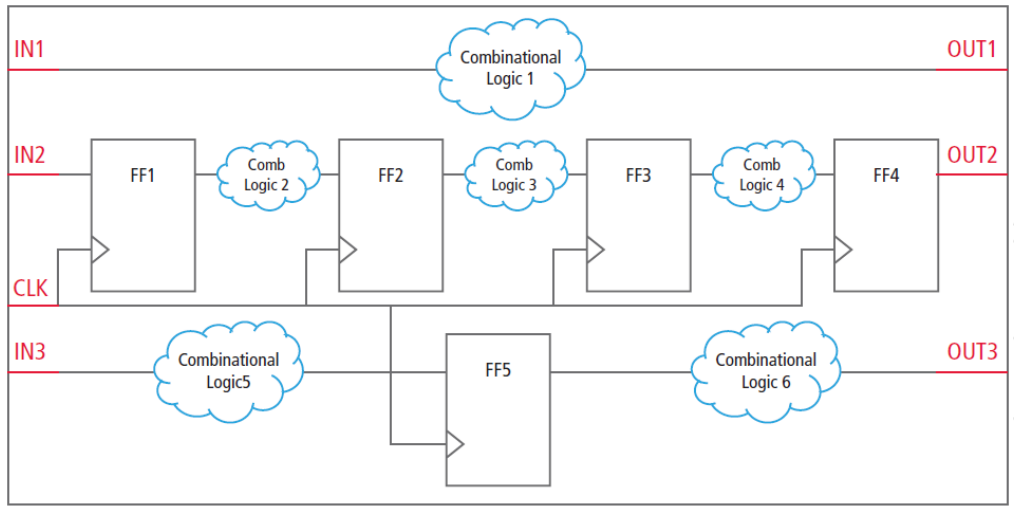
GLS Flow



SDF Sample

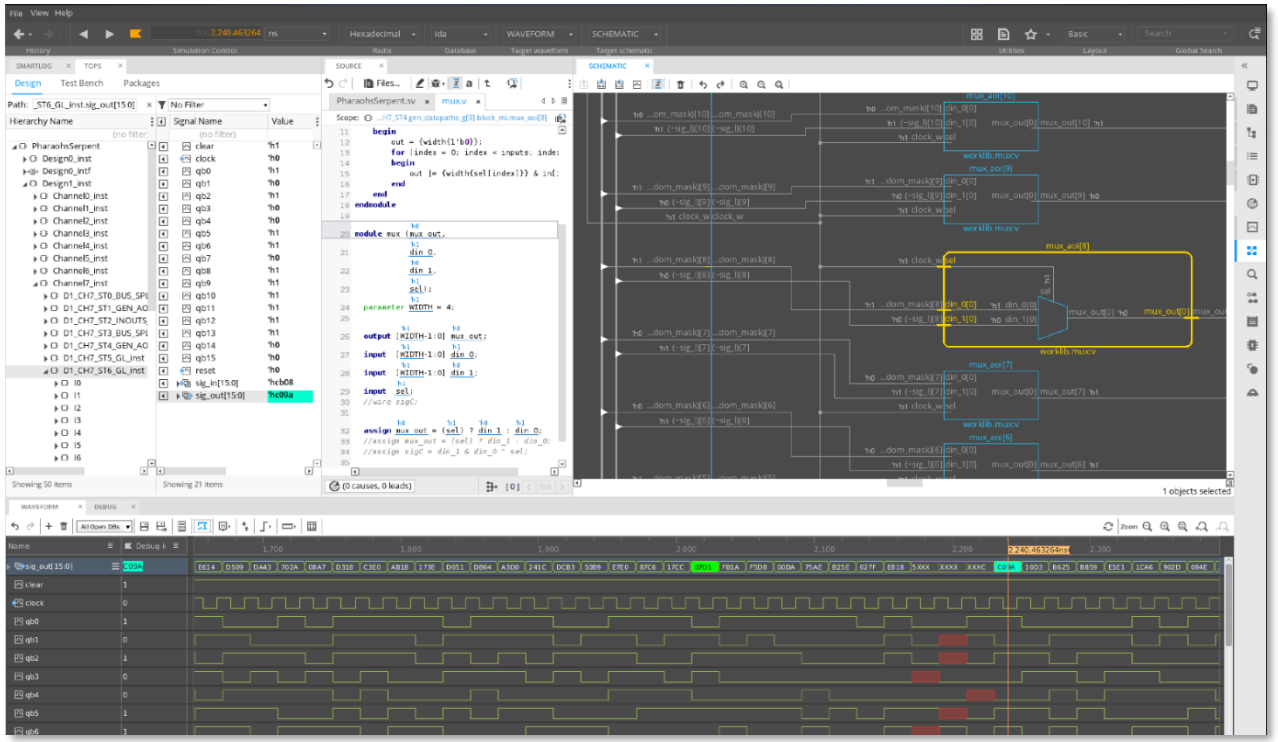


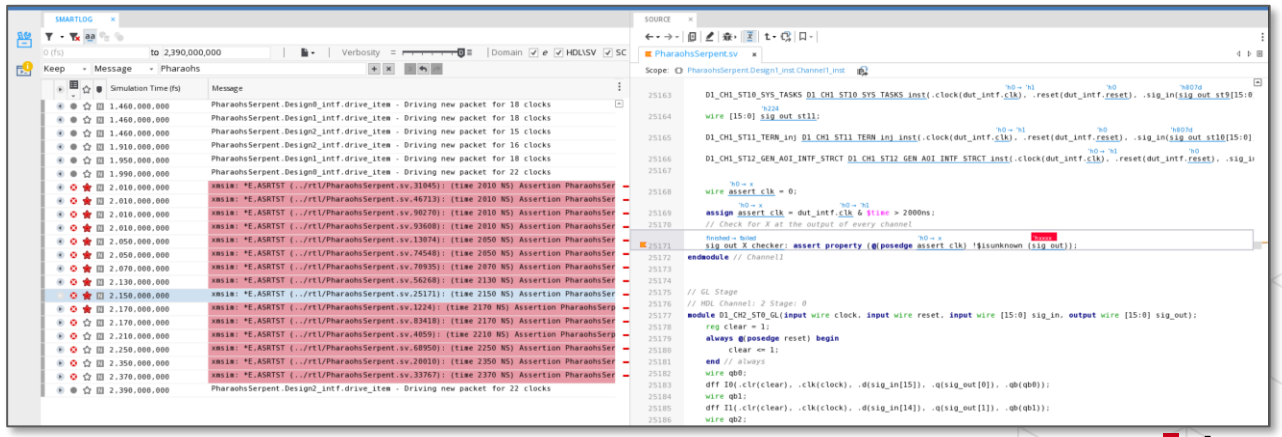
Design with sequential and combo blocks



Debugging Methods

- Code traversing
  - Code debug at each simulation time
- Waveform Debug
  - Analyzing dependent signals
  - Transaction, Mnemonic
- Schematic Tracing
  - Drivers and Loads
  - Connectivity verification (GLS)
- Analyzing Simulation log
  - Without running the simulation
  - Time traversing

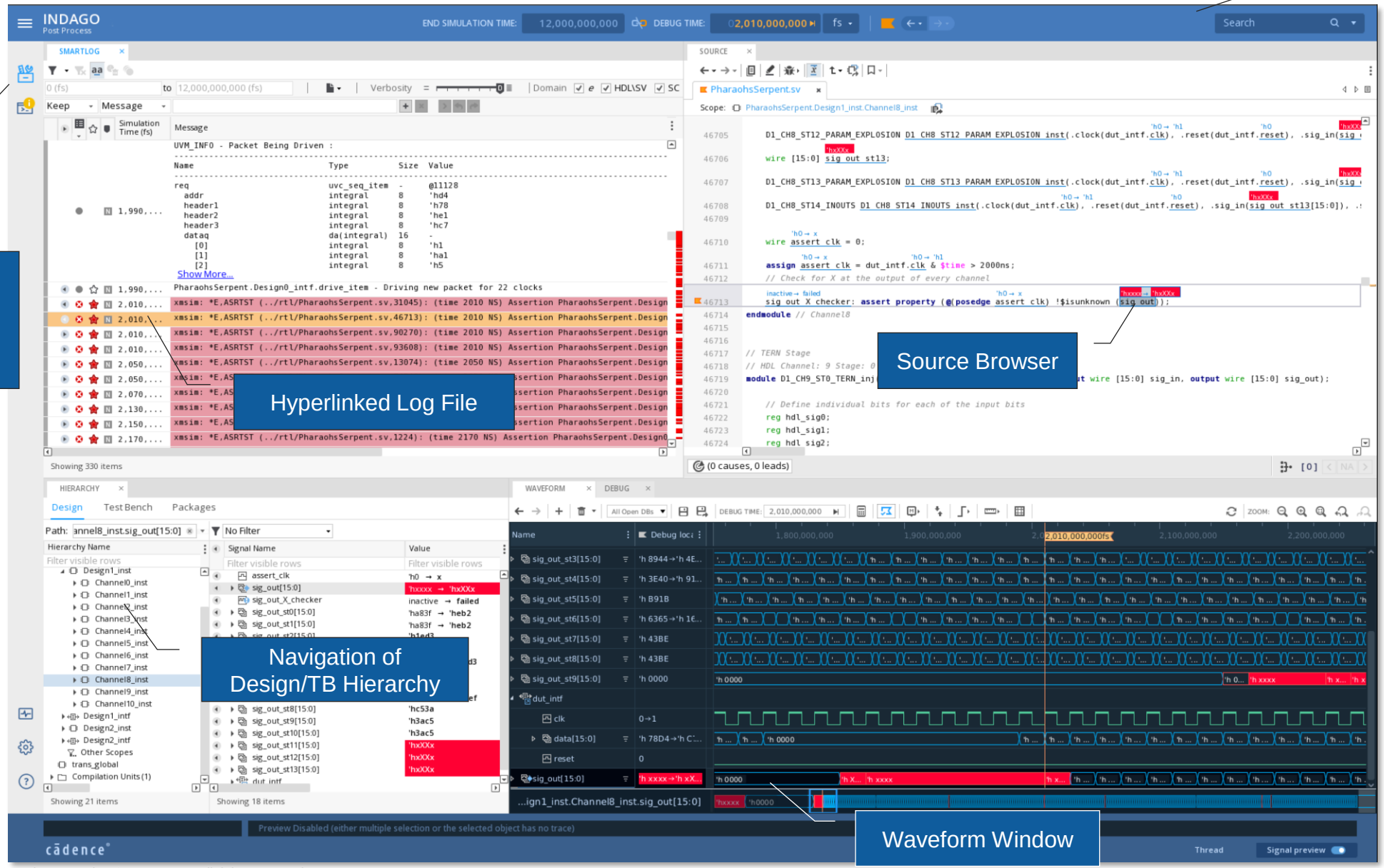




Debugging Methods

High-Performance GUI (Launching, Analysis/Search)

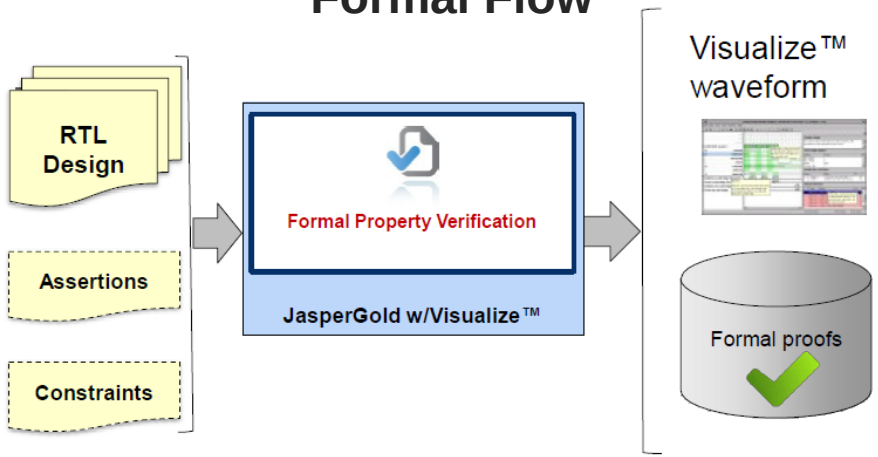
Quickly launch supporting debug tools

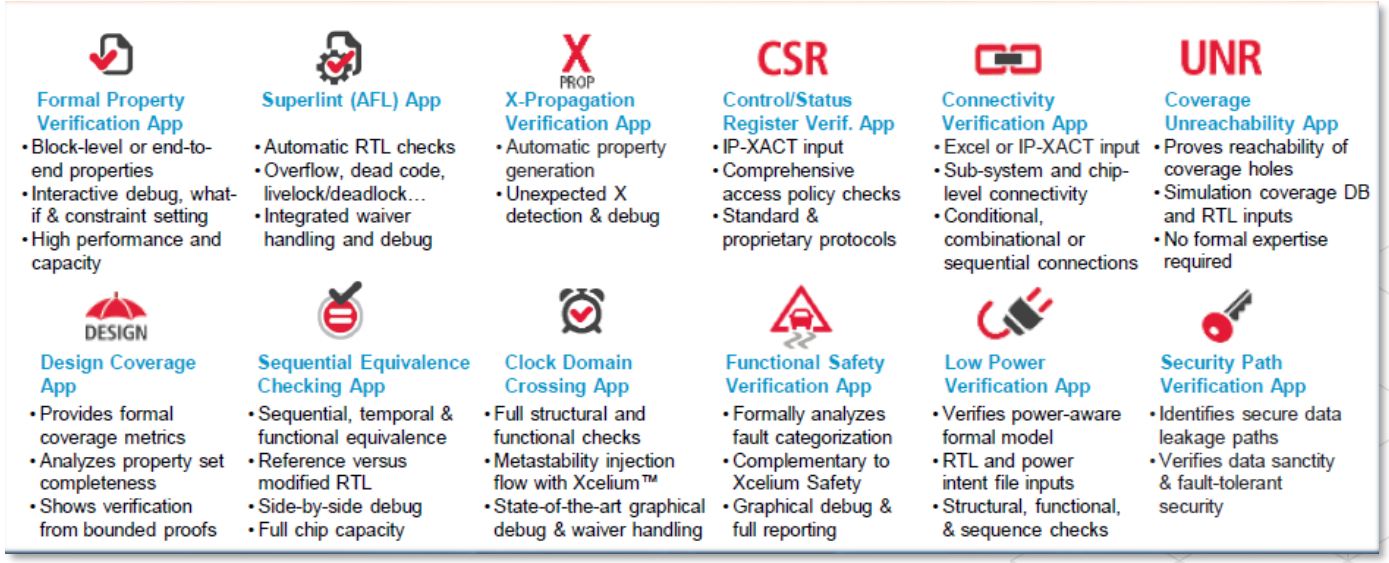


Formal Verification

- Checks exhaustive if a **model** meets a given **spec**
  - Model: Synthesizable RTL
  - Spec: Properties (Assertions and Covers)
- Formal Analysis
  - Formal tests all possible stimuli, one cycle at a time
- Properties
  - Formal checks properties as it is walking through stimuli
- Constraints
  - Not all input stimulus combinations are legal
  - Assume properties tell formal what is legal

Formal Flow





Regression Environment