2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Portable Stimulus vs Formal vs UVM

A Comparative Analysis of Verification Methodologies Throughout the Life of an IP Block

Gaurav Bhatnagar

12/12/2017

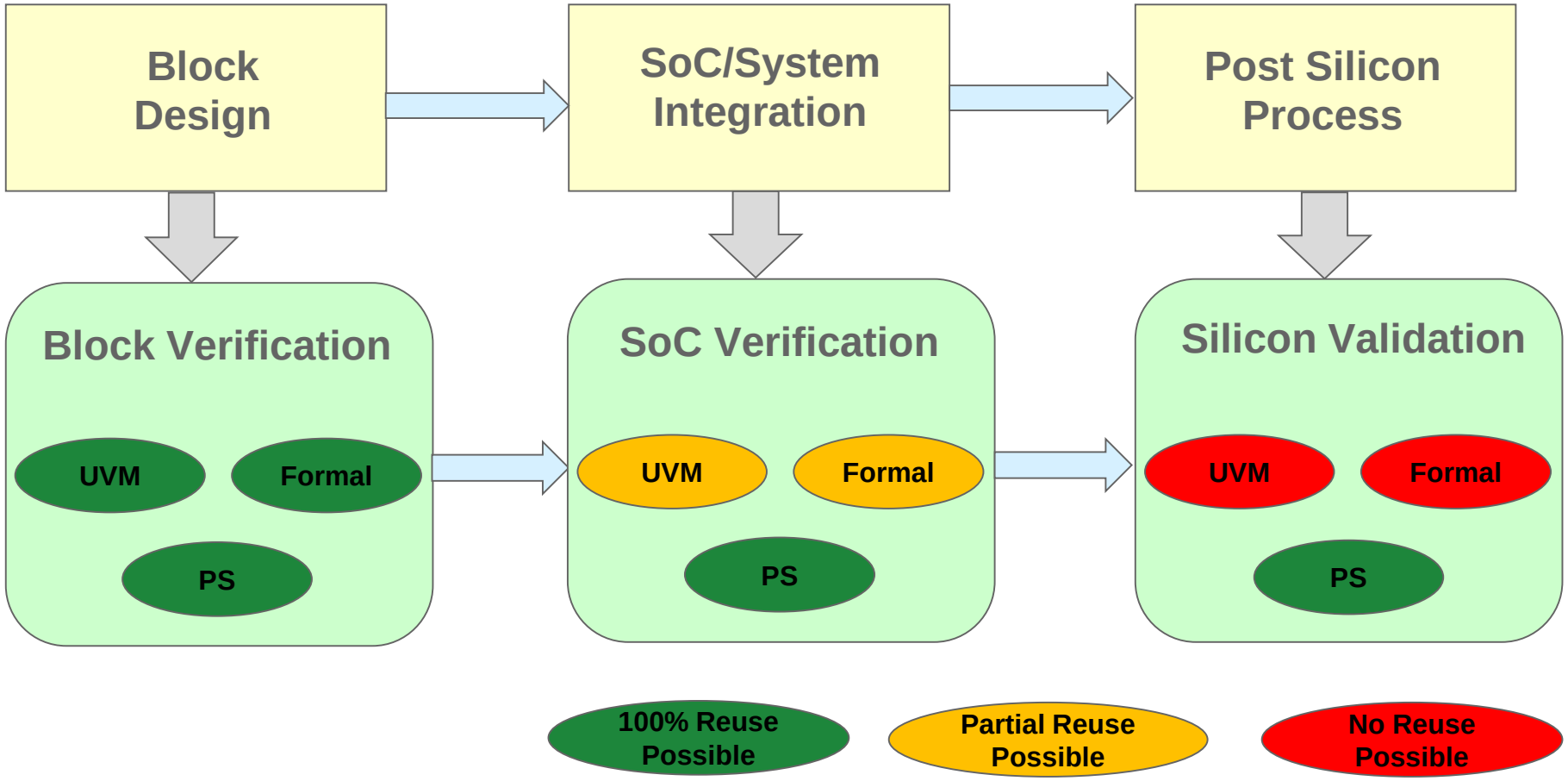
Introduction

- DV Challenges
  - Creating all possible scenarios to test a design
  - Achieving a 100% Coverage
  - Shrinking time to market
  - Under Pressure to Do More, Faster, and with Less!
- Proven Methodologies UVM based and Formal Verification
- Upcoming PS based Verification Methodology
- Choice of Verification Methodology based on requirements
- Comparative Analysis

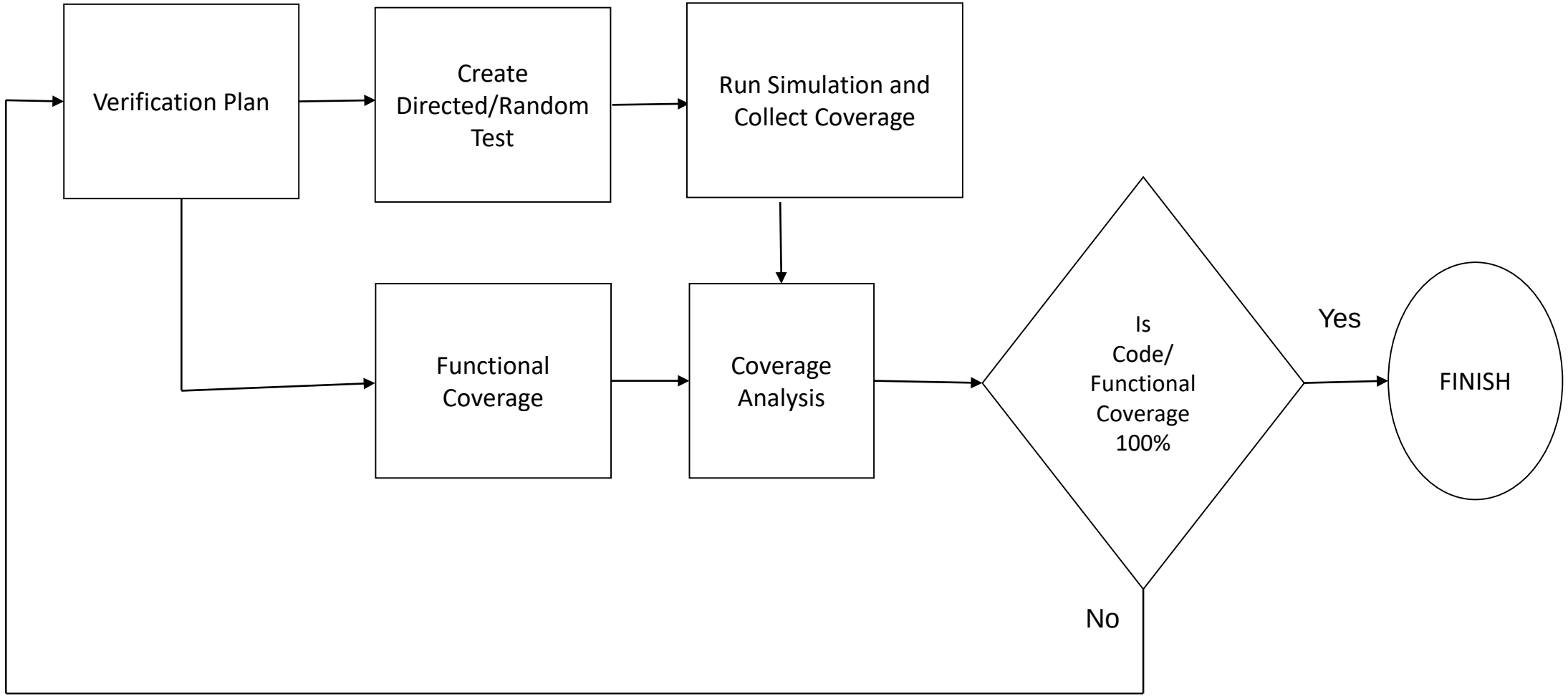
Life cycle of an IP Block

- Block Level Verification
  - Focus on micro-architecture
  - Exhaustive test conditions using UVM or Formal techniques
- SoC Level Verification
  - Focus on connectivity checks
  - Integration and Directed testing using C and SV based tests
- Silicon Validation
  - Focus on Customer Scenario
  - Use Case testing using C/C++ based platforms

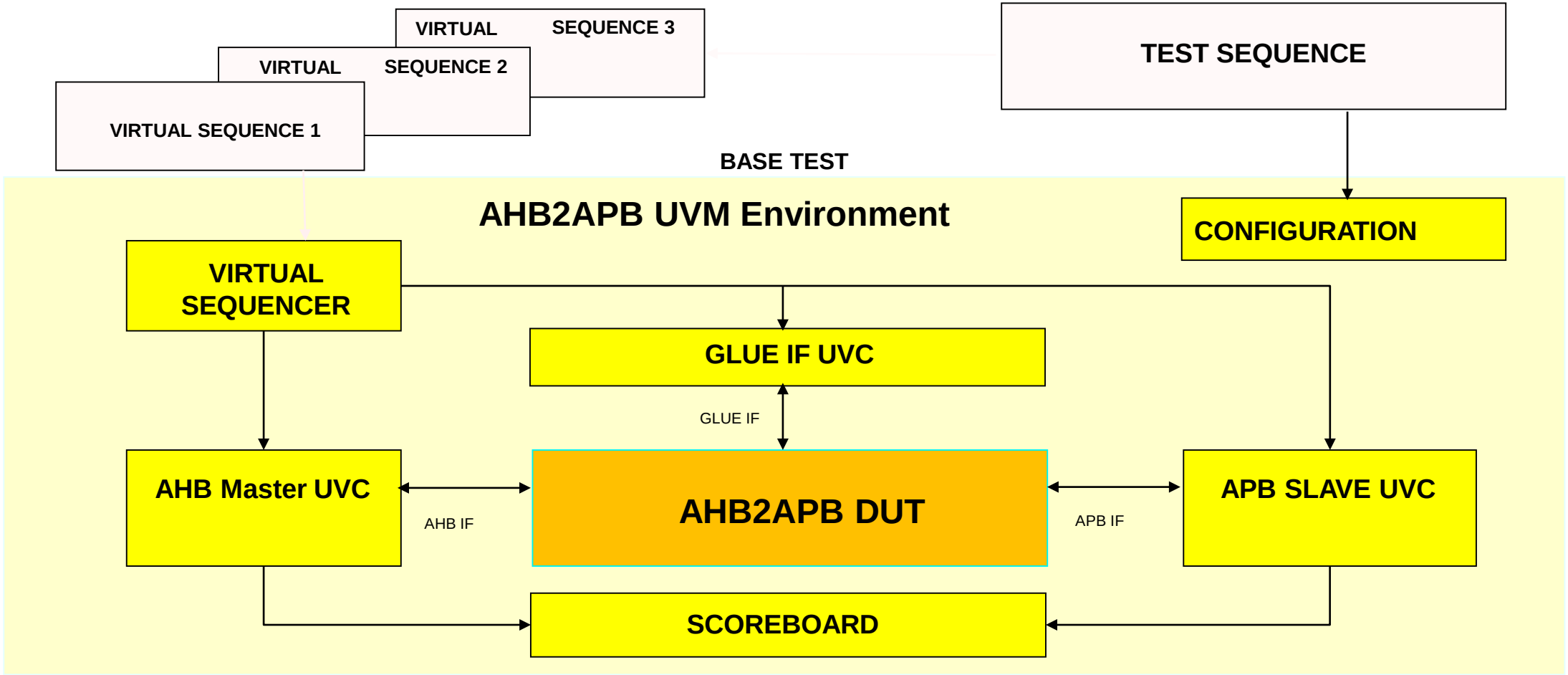
Life cycle of an IP Block



UVM Based Verification Flow



AHB2APB UVM based Verification



2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

AHB2APB UVM Verification

- Pre-verified IP block upgraded with a glue logic
- AHB to APB conversion with additional control and debug logic
- AHB Master UVC, APB Slave UVC and Glue Interface UVC
- Directed and Random tests
- Regressions are run and reports are generated
- Functional and Code Coverage
- No Bugs Found

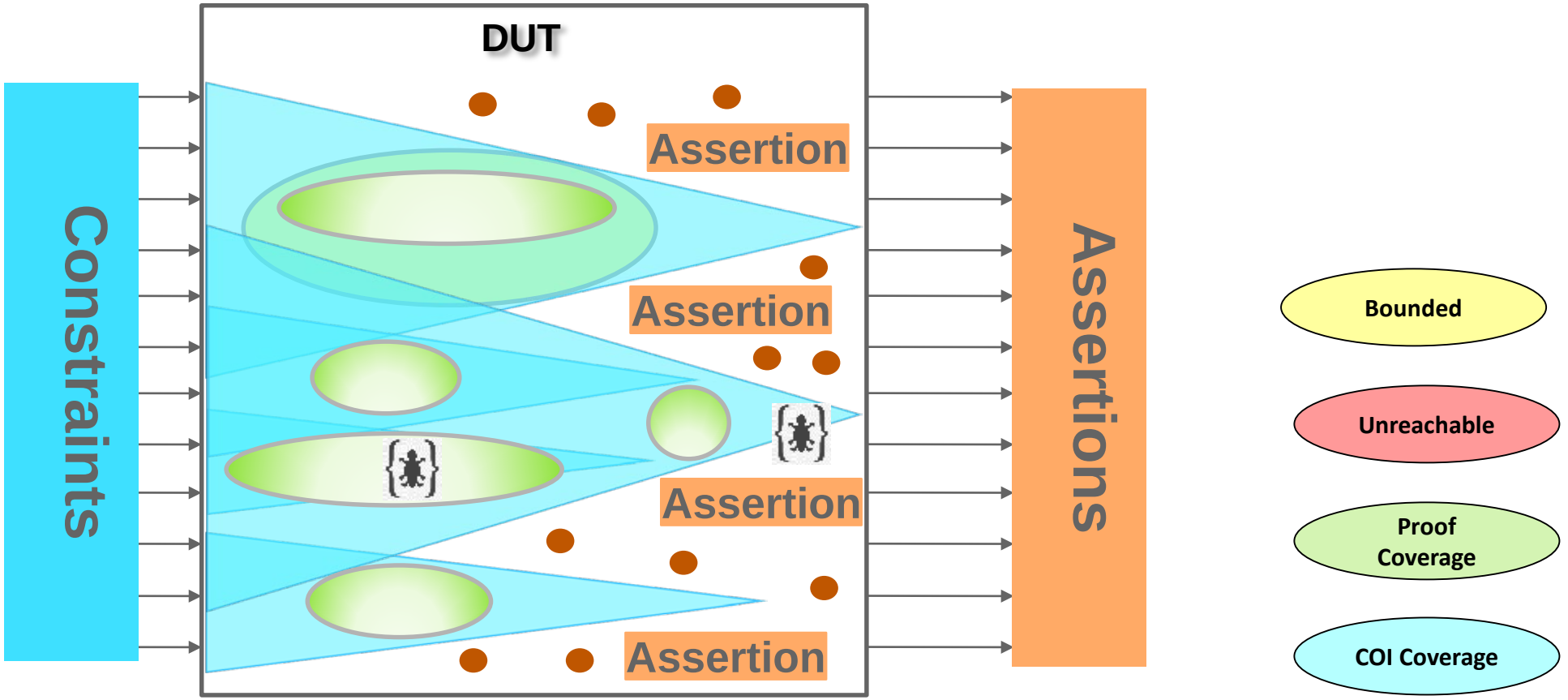
2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

AHB2APB UVM Verification

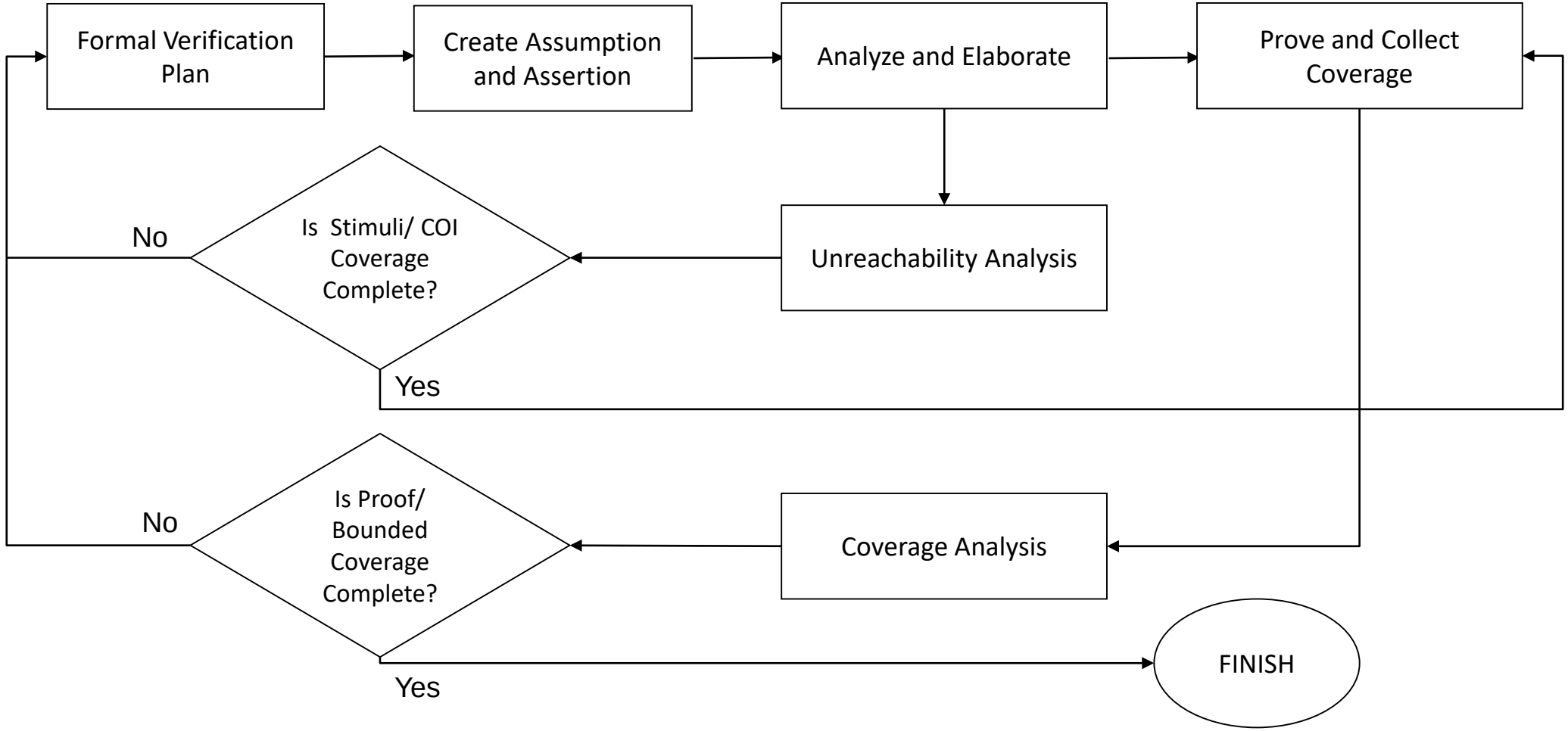
Initial Setup	Directed Tests	Random Test	Coverage Closure	Overall Development Time
1 week	2 weeks	1 weeks	2 weeks	6 weeks

Tests Run	Passed	Failed	Not Run	Overall Code Coverage
106	106	0	0	2634/2864

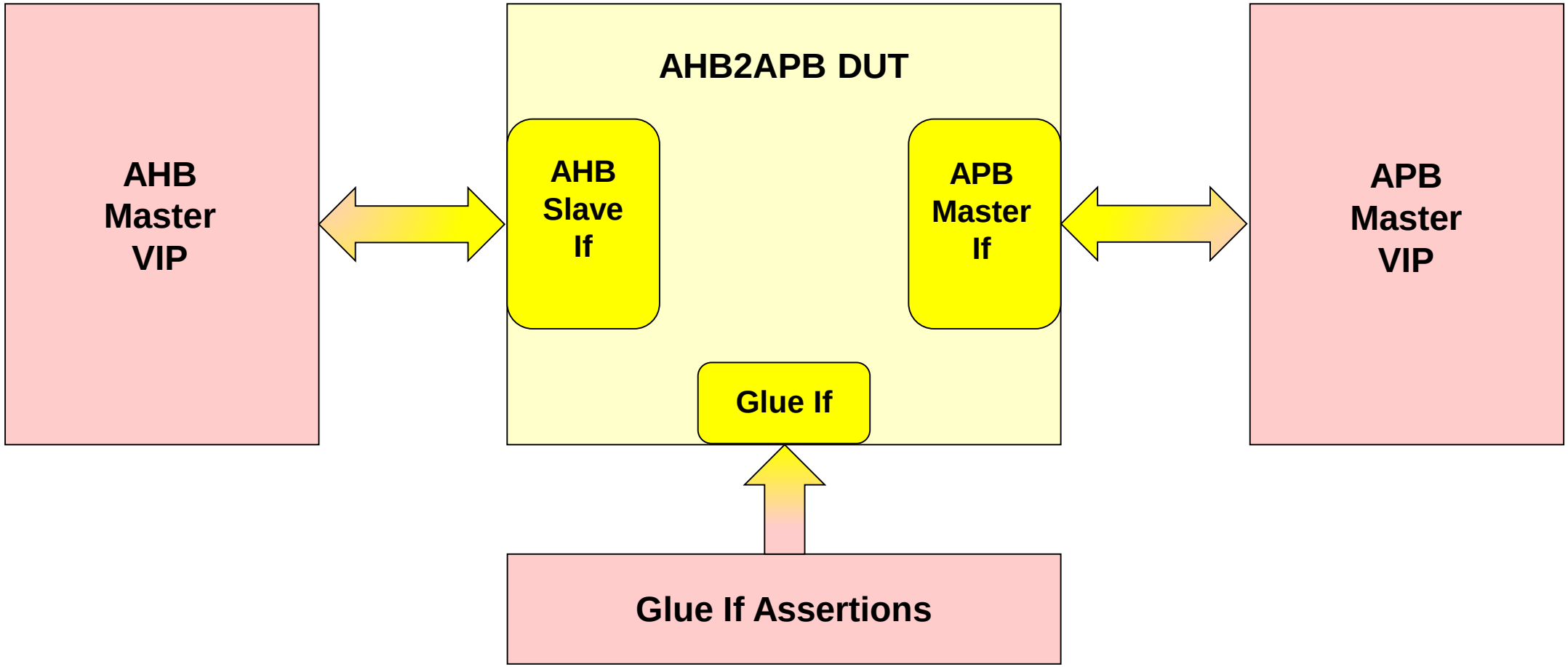
Formal Verification: Coverage Types



Formal Verification Flow



AHB2APB Formal Verification Setup



2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

AHB2APB Formal Verification Results

- AHB and APB ABVIP for interface assertions
- Increased Verification Quality due to standard components
- User defined assertions specific to Glue Logic
- **BUG!!** found related to Glue Logic
- Unwanted switching of logic found
- Glue logic driver limitation in UVM environment found
- **The UVM based environment also finds the error after the fix**
- Random test and code coverage analysis in UVM environment

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

AHB2APB Formal Verification Results

Initial Setup	Unreachability Analysis, ABVIP Assertion and Debug	Manual Assertion Coding and Debug	Analysis and Coverage Closure	Overall Development Time
1 week	1 weeks	1 weeks	2 weeks	5 weeks

Total number of COI Items	Unreachable Waived Items	Undetermined	Bounded and Waived	Proof Coverage
106	106	0	0	2634/2864

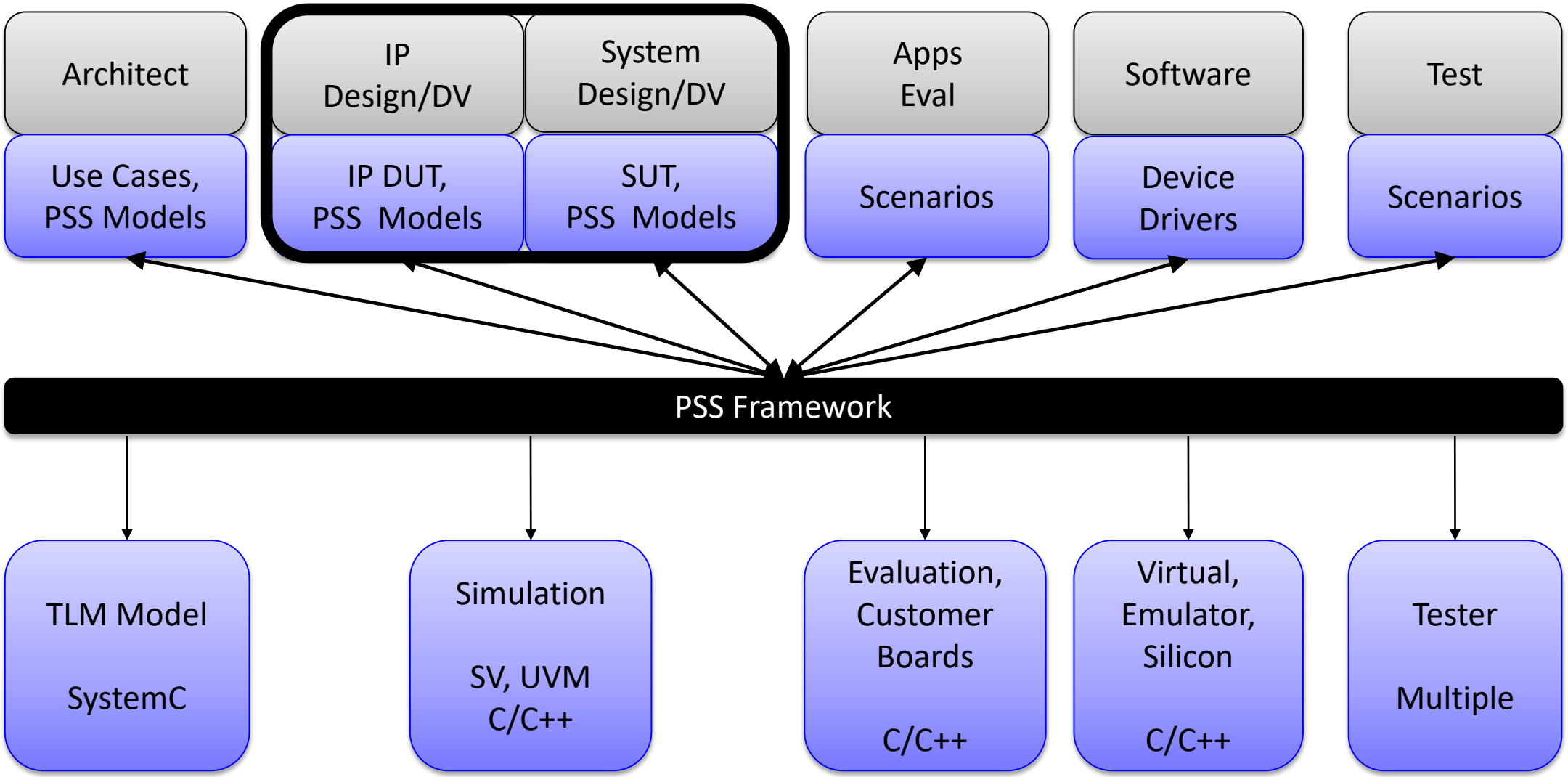
Proven	Unprocessed	Processing	Failed	Passed
483	0	0	0	483

What is Portable Stimulus?

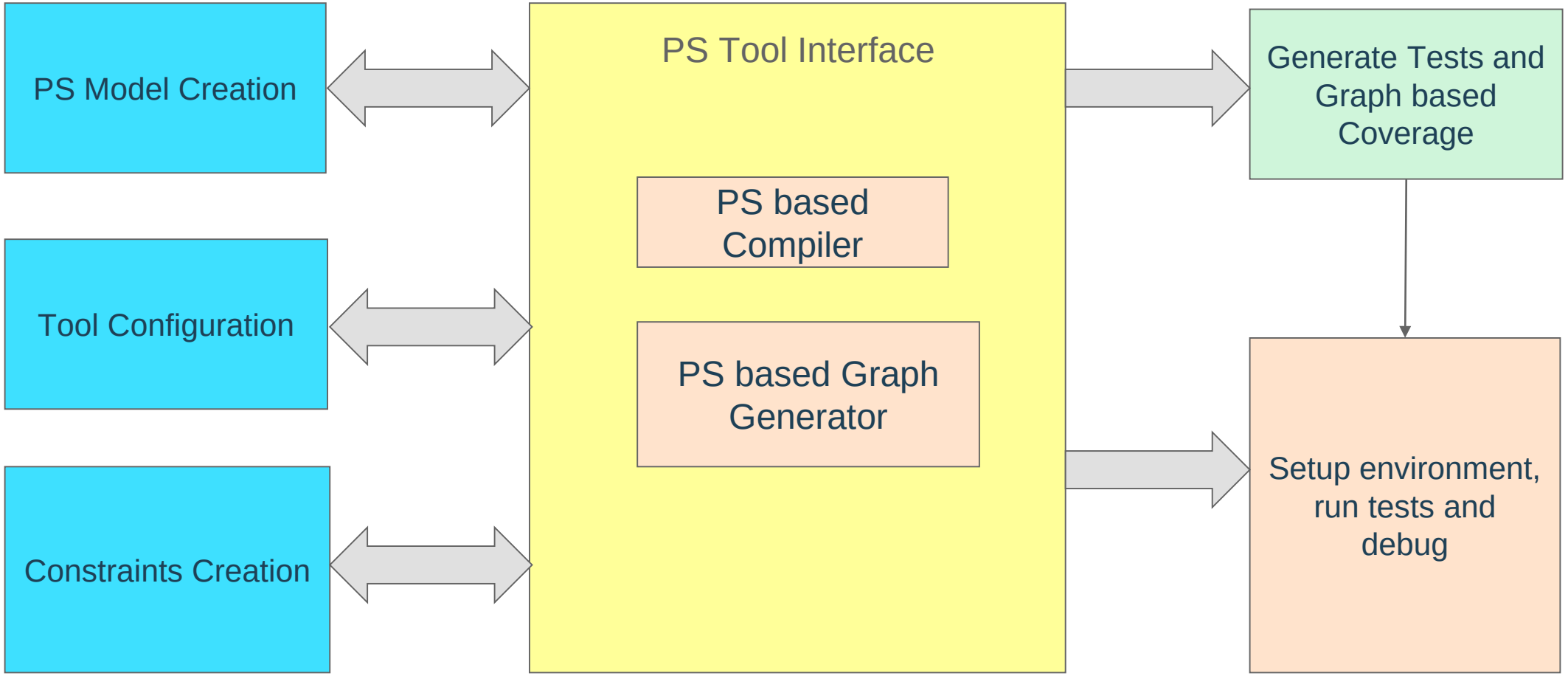
- New standard defining new test writing language for Portable Tests
- Vertical Reuse
  - Tests developed at the IP level easily integrated/reused at SOC level
- Horizontal Reuse
  - Simulation, Emulation, board level, tester etc.
- Bi-Directional Re-Use
  - Evaluation Board Failure to IP Test
  - Reuse S/W drivers in simulation

- Supported by Breker Trek, Cadence Perspec and Questa Infact

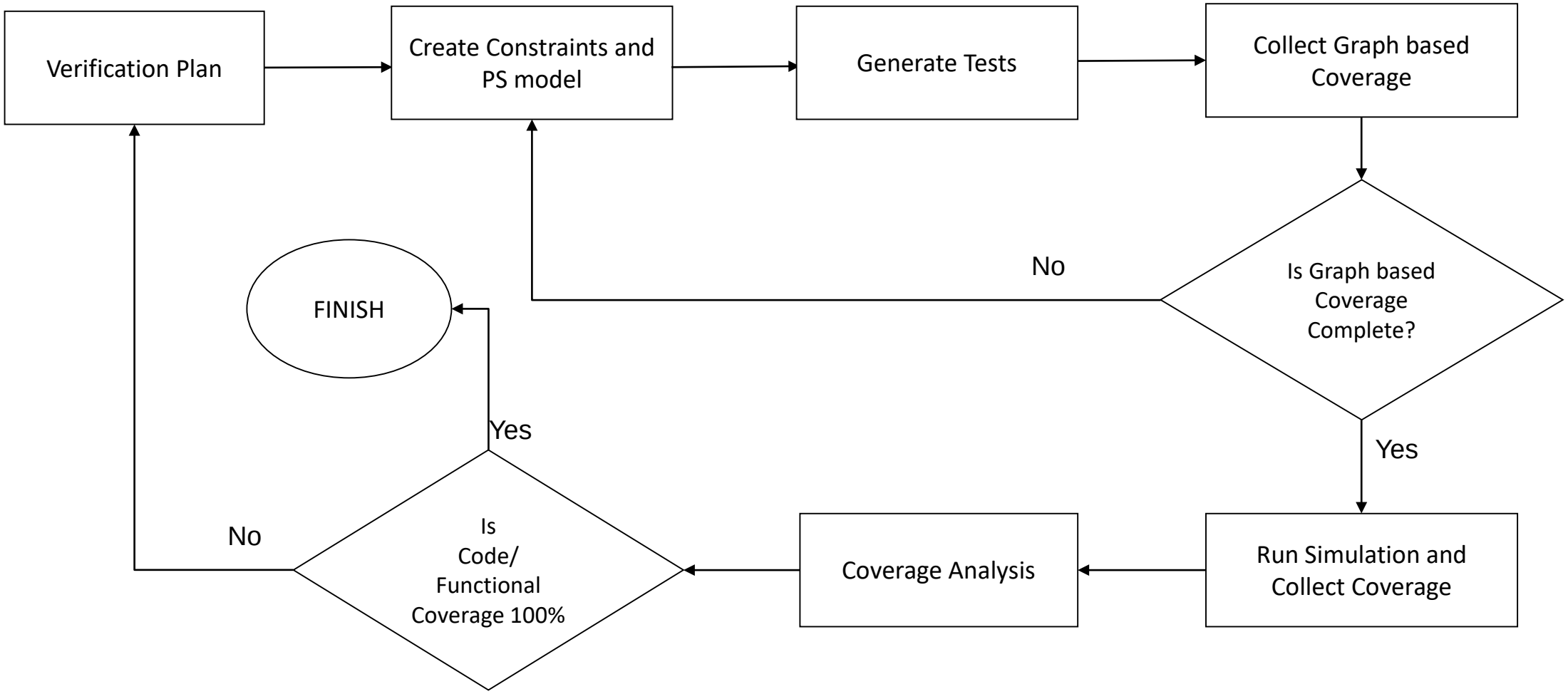
PS based Development Process



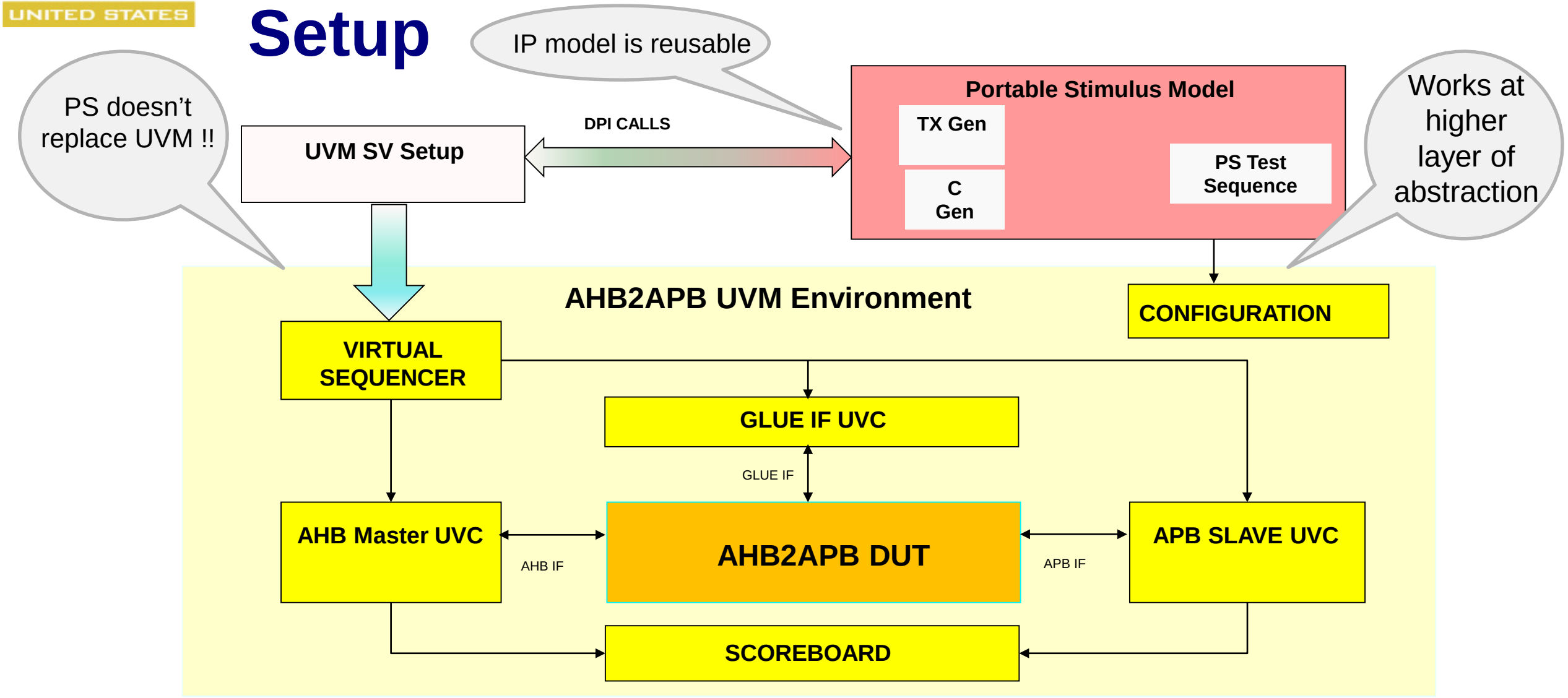
Generating tests from PS Flow



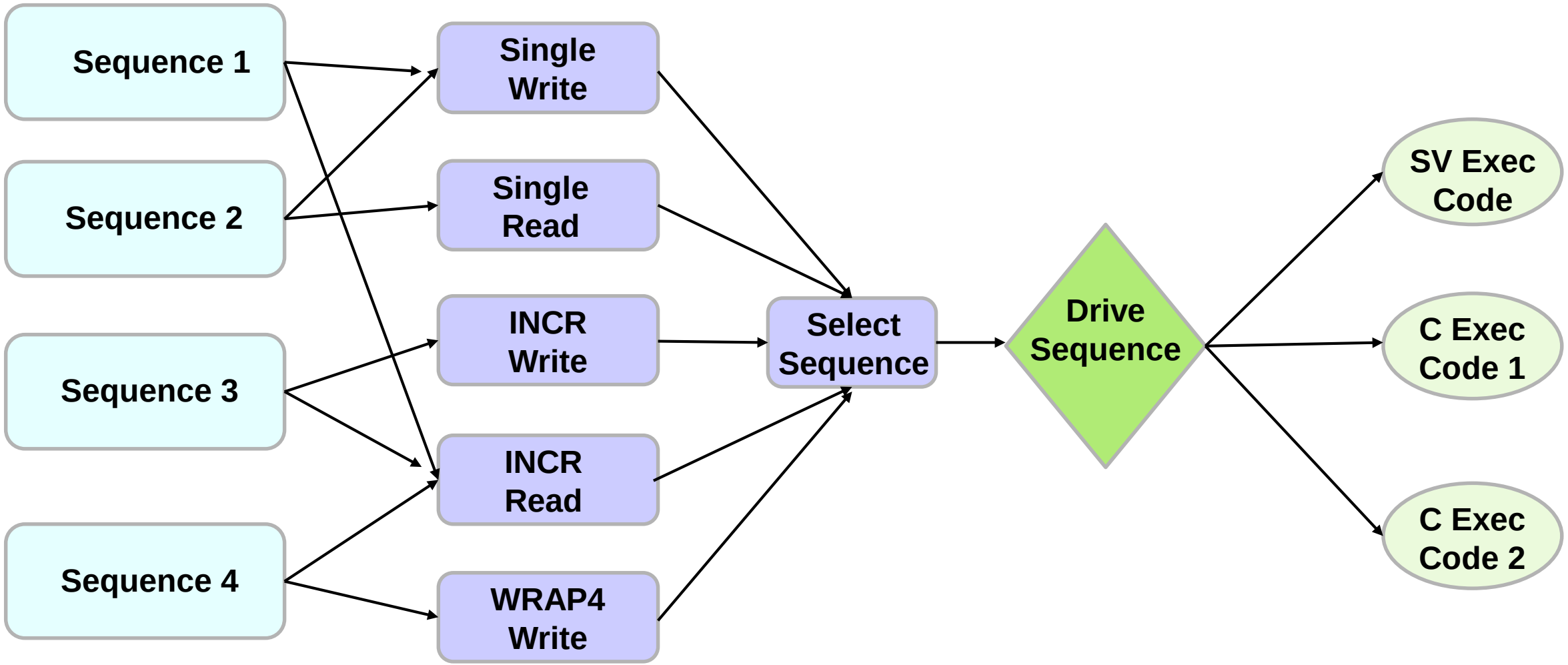
PS based Verification Flow



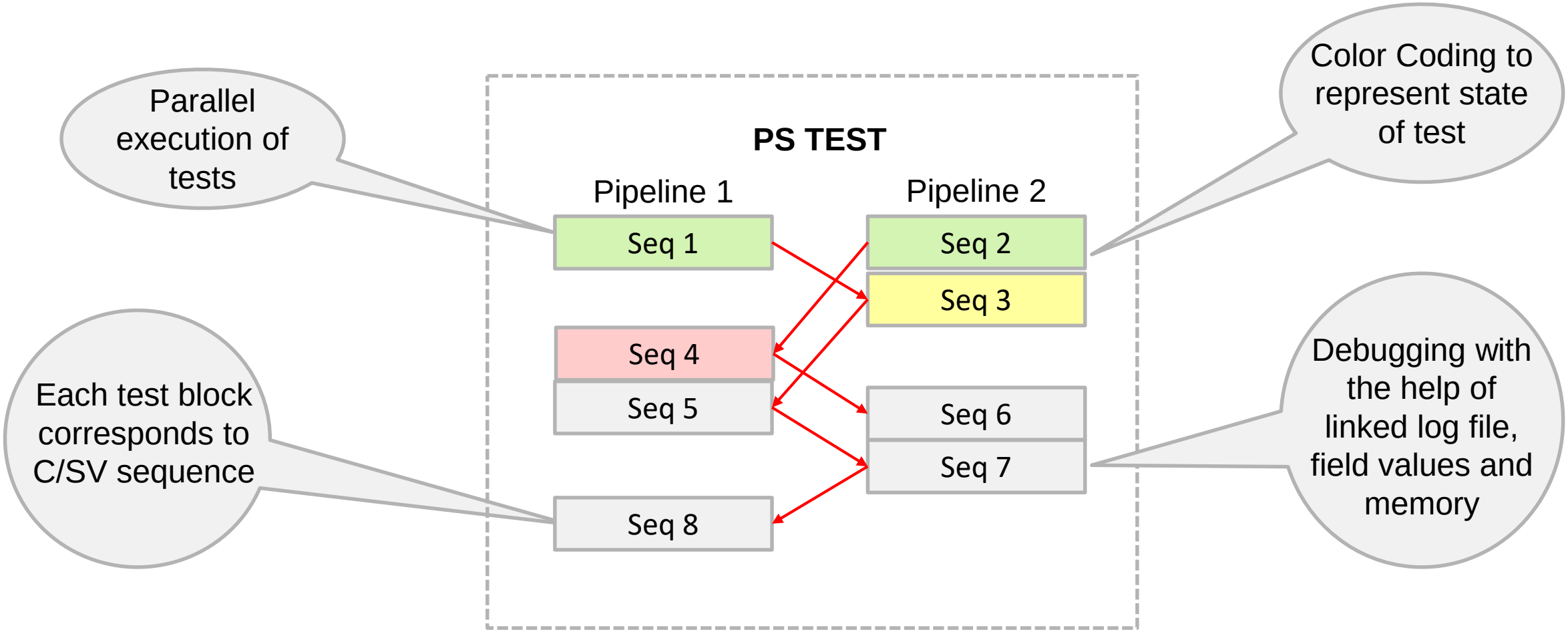
AHB2APB PS based Verification Setup



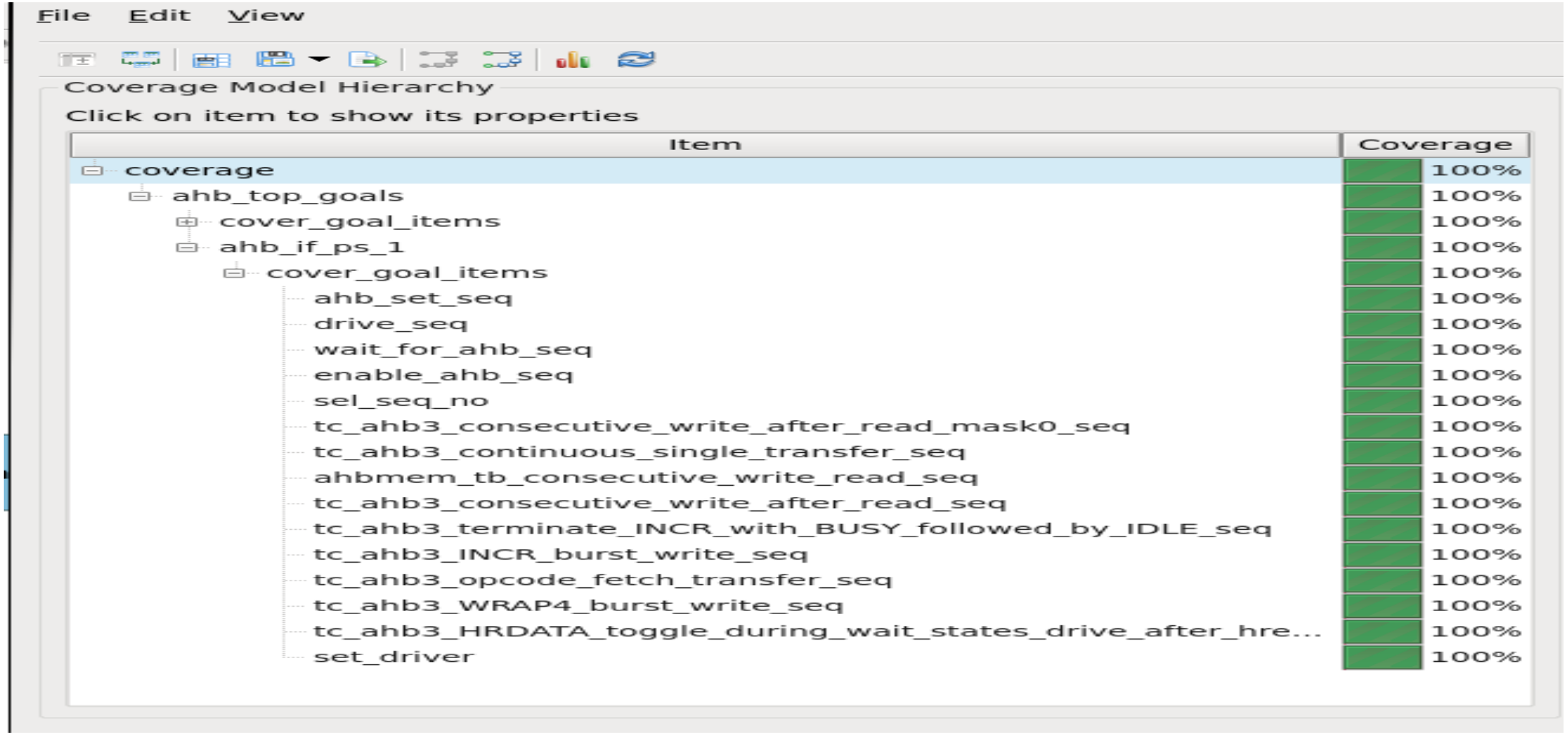
MODEL CREATION AND CONSTRAINTS



TEST GENERATION AND DEBUG



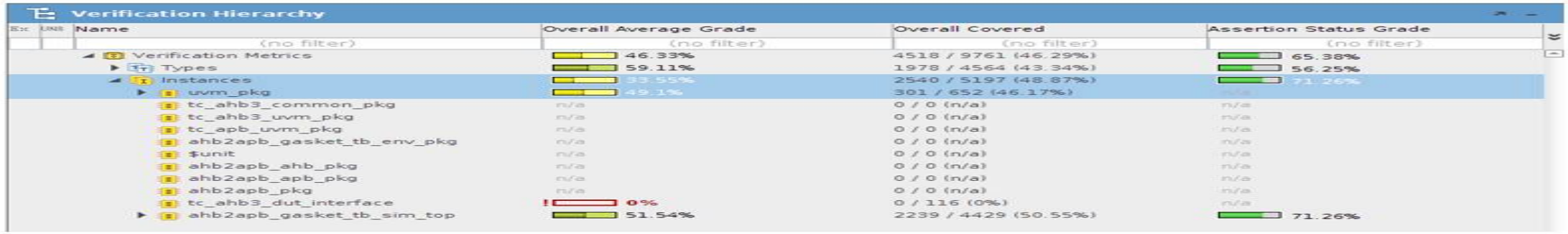
GRAPH COVERAGE



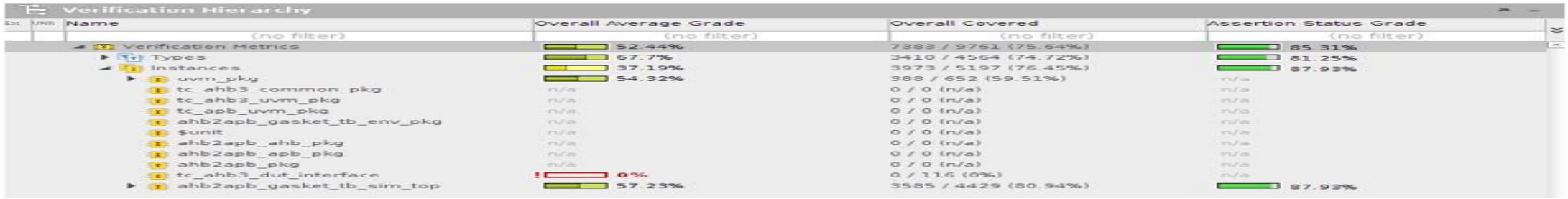
REGRESSION ANALYSIS

- SV Random vs PS Ransom comparison
- AHB2APB Gasket Regression with 10 seeds
- Exact same number of sequences run

SV Random



PS Random



AHB2APB PS based Verification Results

- PS Models generated tests covering more conditions
- The number of tests in a regression required to achieve same coverage results is reduced by **50%**!!
- Time required to bring up the verification environment, code the models and generate tests is 6 weeks
- The bug found in case of Formal Verification took less number of regressions to hit
- Coverage Closure is easier due to high quality tests

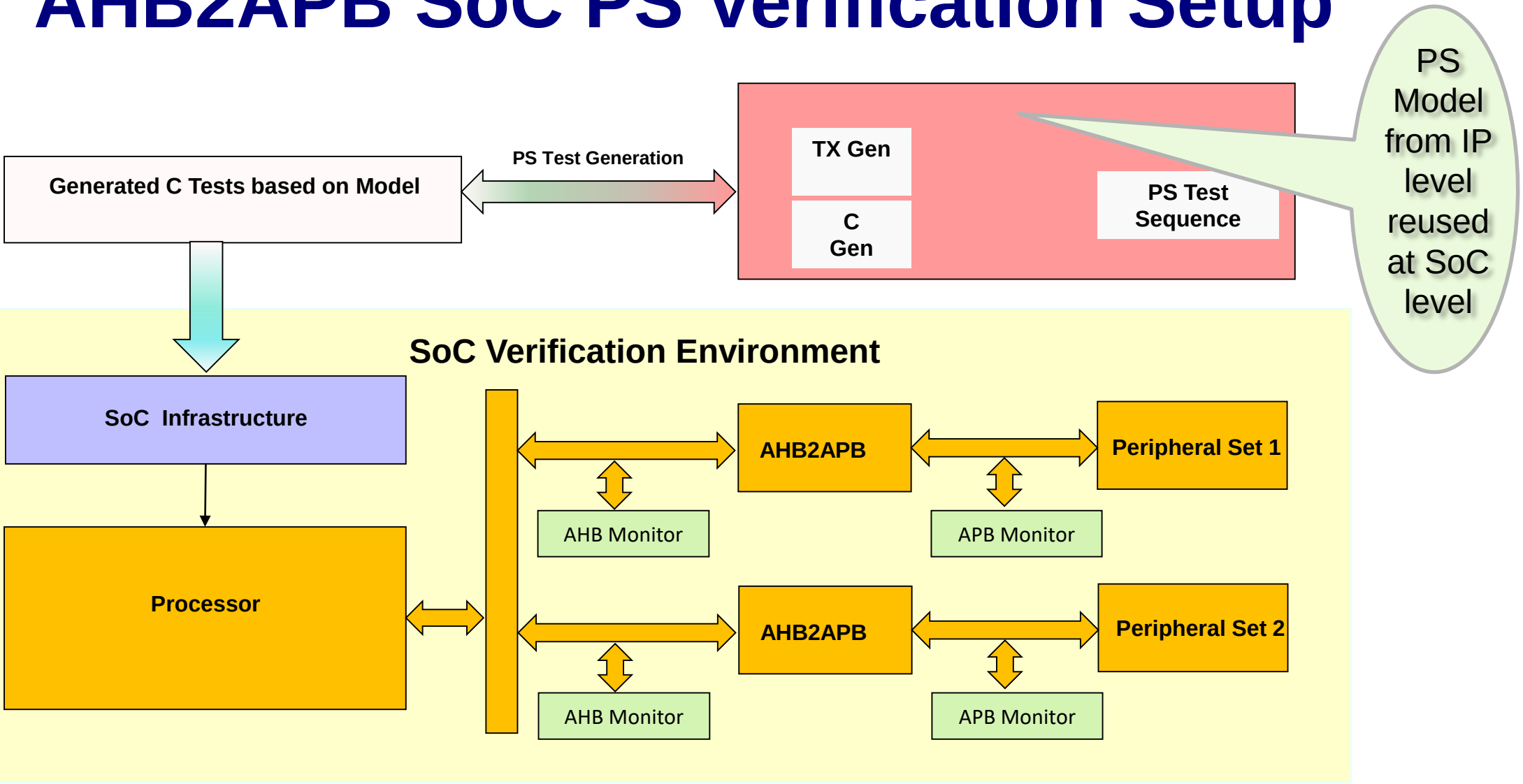
2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

AHB2APB PS based Verification Results

Initial Setup	Initial Model Development	Test Generation, Graph Coverage and Run	Coverage Closure	Overall Development Time
1 week	2 weeks	1 weeks	2 weeks	6 weeks

Tests Run	Passed	Failed	Not Run	Overall Code Coverage
50	50	0	0	2634/2864

AHB2APB SoC PS Verification Setup



AHB2APB SoC PS Verification Results

- PS Models can be used to generate C tests
- The C tests can be compiled and run on the processor for SoC simulation
- The tests can be integrated with Post Si Evaluation board setup to be run with different debuggers and tools
- The tests can be mapped to different instances of the AHB2APB VIP
- The tests inherit the same test quality as seen at the IP level
- Integration **BUG**! was uncovered using the IP based models at the SoC level

Post Si Evaluation with PS

- UVM and Formal can’t be used
- PS Models can be targeted for C test generation for Post Si Validation
- Integration with common debug platforms
- High quality, self checking constrained random tests
- Debug Interface for generated tests
- Failures on Post Si Evaluation Board can be ported back to simulation

Comparative Analysis

Point of Comparison	PS vs UVM	PS vs Formal
Verification Planning and Development Time	• Planning and Development time is very similar • Overhead of learning new language but it is very similar to C++	• More detailed verification plan required for Formal • Lesser development time for Formal • Availability of ABVIP considerably reduces the verification complexity
Infrastructure Development and Ease of integration	• Integration logic is required both for model and SV code • The integration logic is reusable across common interface	• Setup time will vary depending on the design complexity for Formal • Easier for Formal due to lesser complexity
Effective test generation	• Visual representation of tests • High Quality tests covering more conditions • Self Checking tests	• Visual representation of tests in PS • Tool generated stimulus in Formal covers most conditions easily

Comparative Analysis

Point of Comparison	PS vs UVM	PS vs Formal
Coverage Closure and Report Analysis	• Graph based Coverage • Easy to identify uncovered graph conditions • More coverage with lesser number of runs	• Code, Functional and Graph based Coverage in PS • COI, Stimuli, Proof and Bounded Coverage in Formal
Reuse and Portability	• Greater reuse possible especially when switching target platforms	Greater reuse possible especially when switching target platforms

Conclusion

- PS based Verification has an edge over UVM based verification
  - Better test quality
  - Reuse across different target platforms
- Formal Verification offers an effective verification approach and a clear winner at the IP level

- All the three verification methodologies are self sufficient and can provide a High Quality Verification

QUESTIONS ???

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

THANK YOU