The How To's of Metric Driven Verification to Maximize Productivity

Author/Prensenter: Matt Graham

Author: John Brennan

Cadence Design Systems, Inc.

cadence®

accellera SYSTEMS INITIATIVE

2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

The How To's of Metric Driven Verification to Maximize Productivity

Cadence Design Systems, Inc.
DVCon Europe – Munich, Germany
October 14, 2014

cadence®

Agenda

Section 1: MDV Methodology IP to SoC Verification

Section 2: MDV Approaches Beyond RTL IP Level

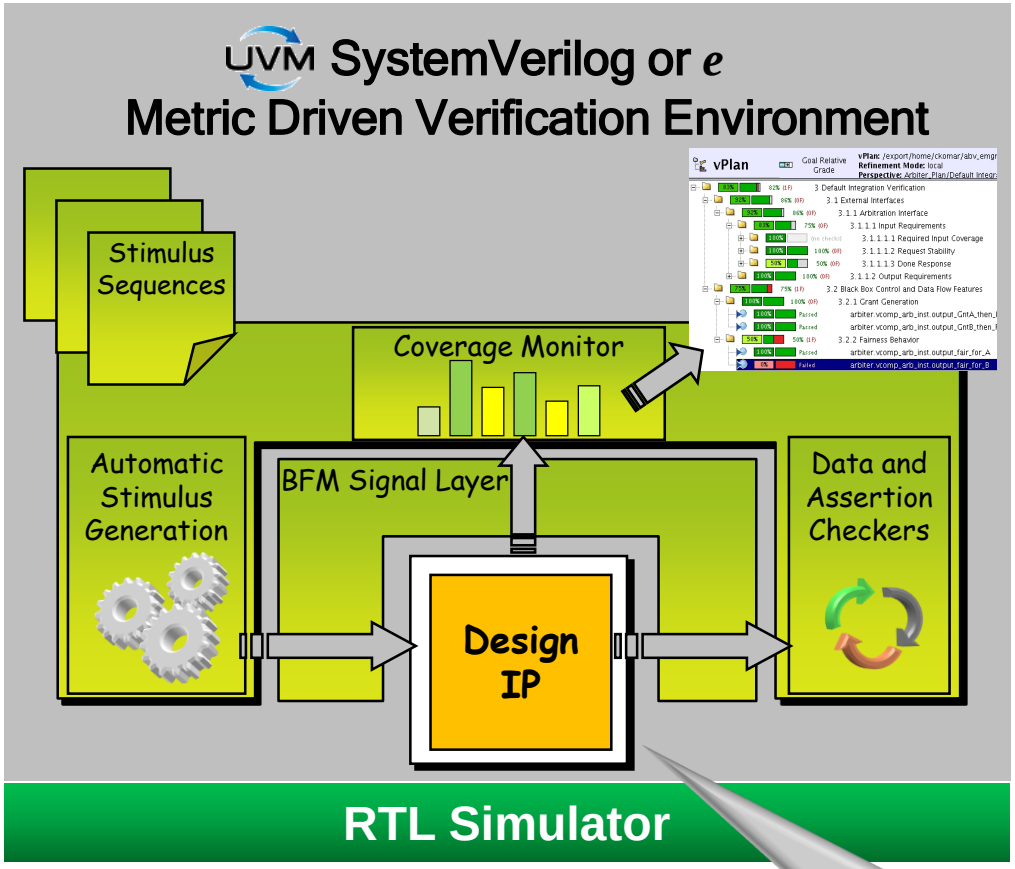
Section 3: Team Based Verification Management

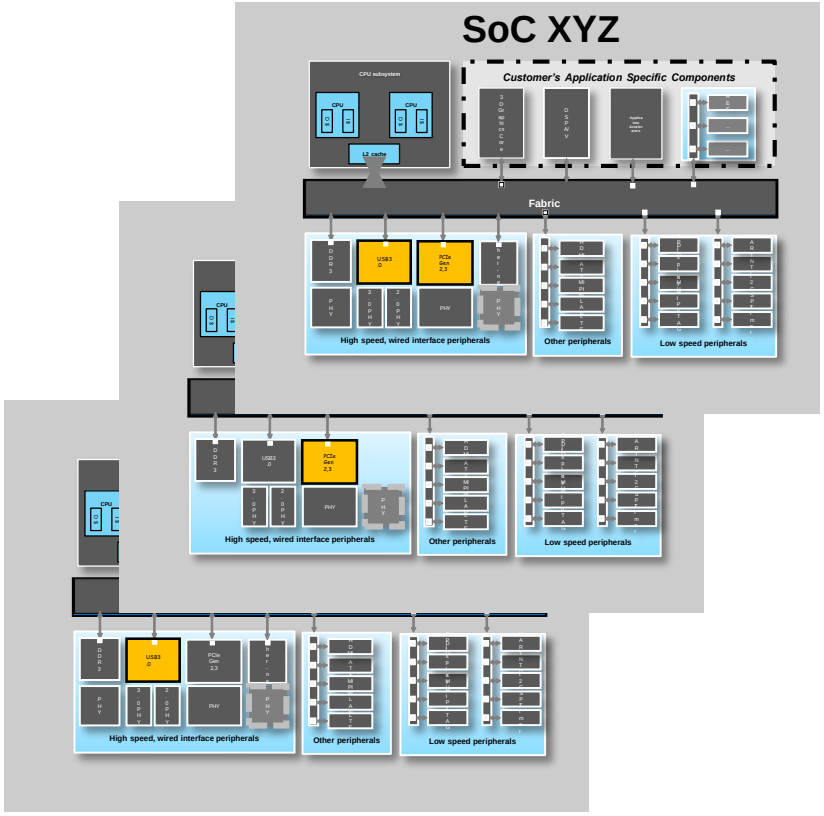
Section 4: MDV In Action

cadence®

IP/Subsystem UVM *e*/SV Metric Driven Verification

Main Verification Flow Being Adopted Past 15 years





Verify IP exhaustively should work in **ANY** SoC context

cadence®

SoC HW/SW Integration & Verification Challenges

Multi-core SW development and HW/SW verification

SoC **SW and IP integration** with 10's to 100's of IPs

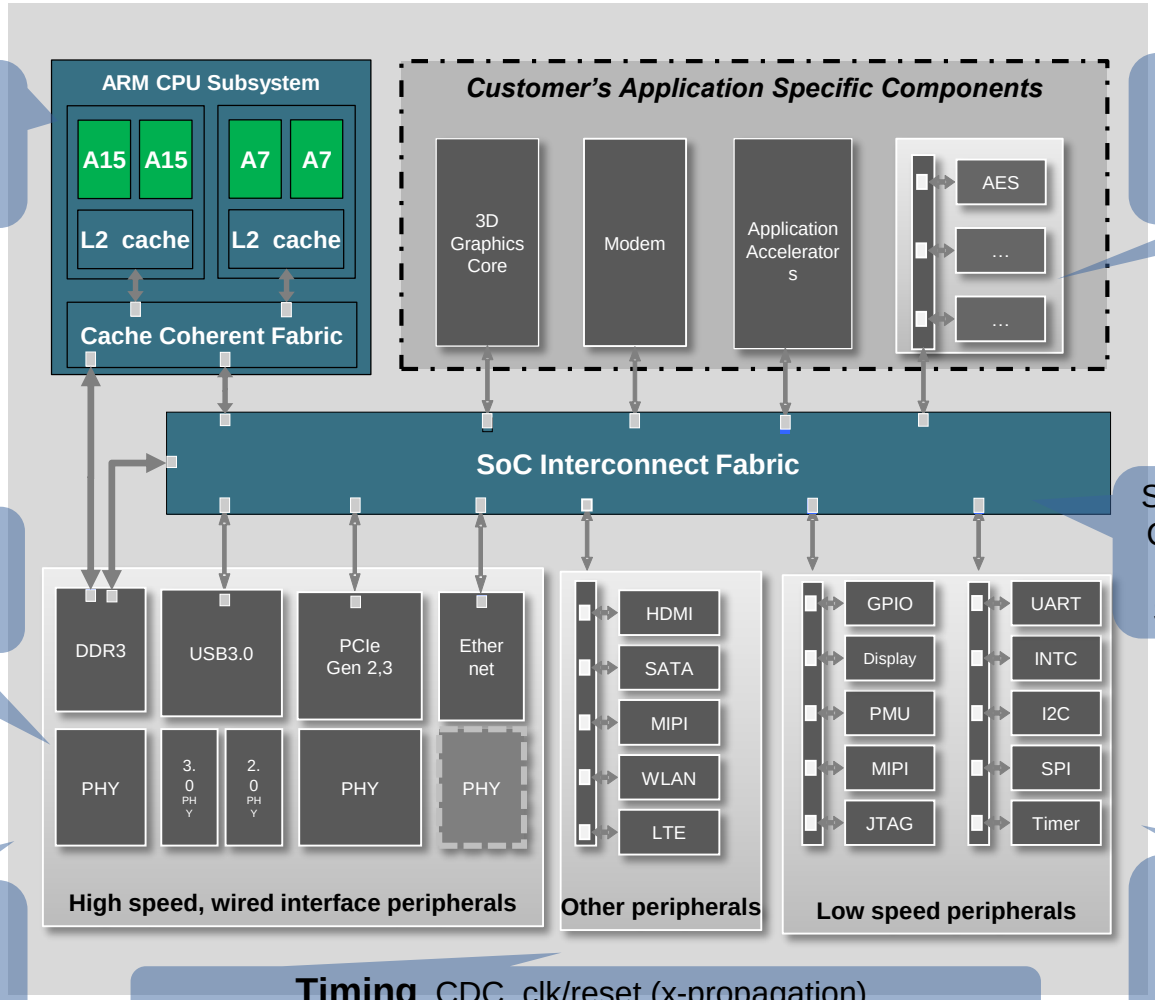
Integration of **Analog-mixed signal**

SoC Coherent and Non-Coherent **Interconnect Complexity** key to **System Performance**

Complex **low-power** design features need to be verified at SoC level spanning HW/SW

Requires **many development environments** on different platforms

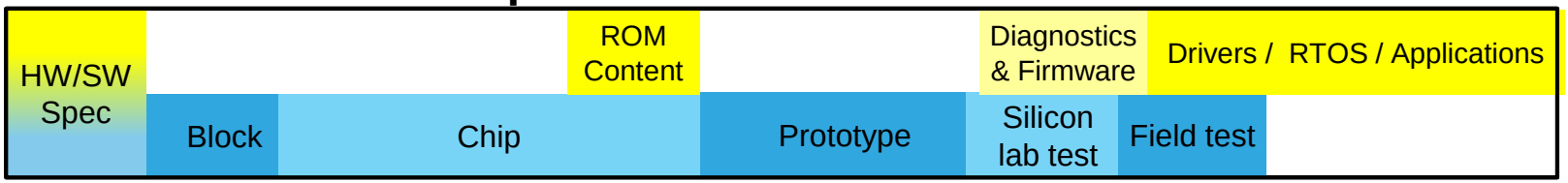
Timing, CDC, clk/reset (x-propagation), **gate-level simulation**



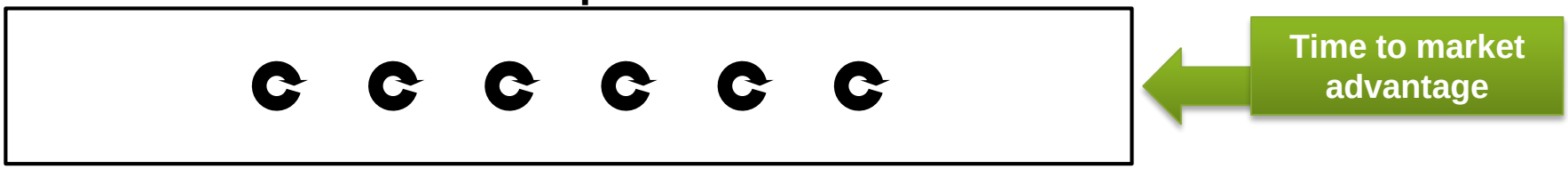
cadence®

Need for Concurrent HW/SW Development
Shift Left

Serial HW->SW Development



Concurrent HW->SW Development

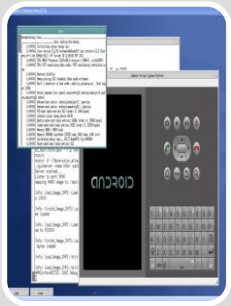
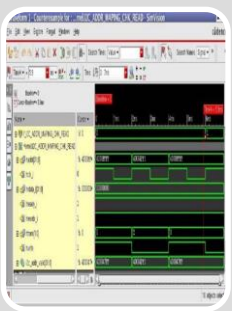
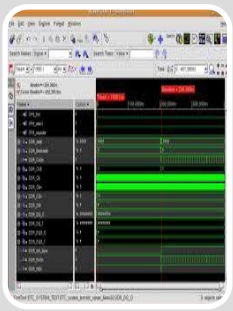


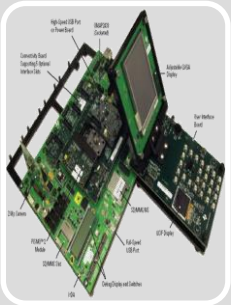
- Integrate HW/SW early and often
- HW designed and verified in SW context
- Software exposed early to HW spec changes
- Verify SoC can support required SW applications

cadence

Many Platforms for IP to SoC HW/SW Development

Verification and Software platforms need to interoperate

SDK OS Simulation

- Highest speed
- Earliest in the flow
- Ignore hardware

Virtual Platform

- Almost at speed
- Less accurate (or slower)
- Before RTL
- Great to debug (but less detail)
- Easy replication

Formal Analysis

- Non-scalable
- Exhaustive
- Early RTL
- Great for IP
- No SW execution

HDL Simulation

- KHz range
- Accurate
- Excellent HW debug
- Broadly available
- Mixed-abstractions
- Limited SW execution

Acceleration Emulation

- MHz Range
- RTL accurate
- After RTL is available
- Good to debug with full detail
- Expensive to replicate

FPGA Prototype

- 10's of MHz
- RTL accurate
- After stable RTL is available
- OK to debug
- More expensive than software to replicate

Prototyping Board

- Real time speed
- Fully accurate
- Post Silicon
- Difficult to debug
- Sometimes hard to replicate

cadence®

Challenges with Many Disconnected SoC Development Environments

Develop high speed abstract C/C++/SystemC Environment f

Develop high speed accurate FPGA Prototype

- ✗ *Many specialized engineering resources required*
- ✗ *Significant development effort for each environment*
- ✗ *Limited sharing of models/VIP between environments*
- ✗ *Difficult to reuse tests across environments*
- ✗ *A lot of effort to migrate between environments*

Virtual Platform

FPGA Platform

Develop IP, Subsystem & SoC RTL Verification Environments

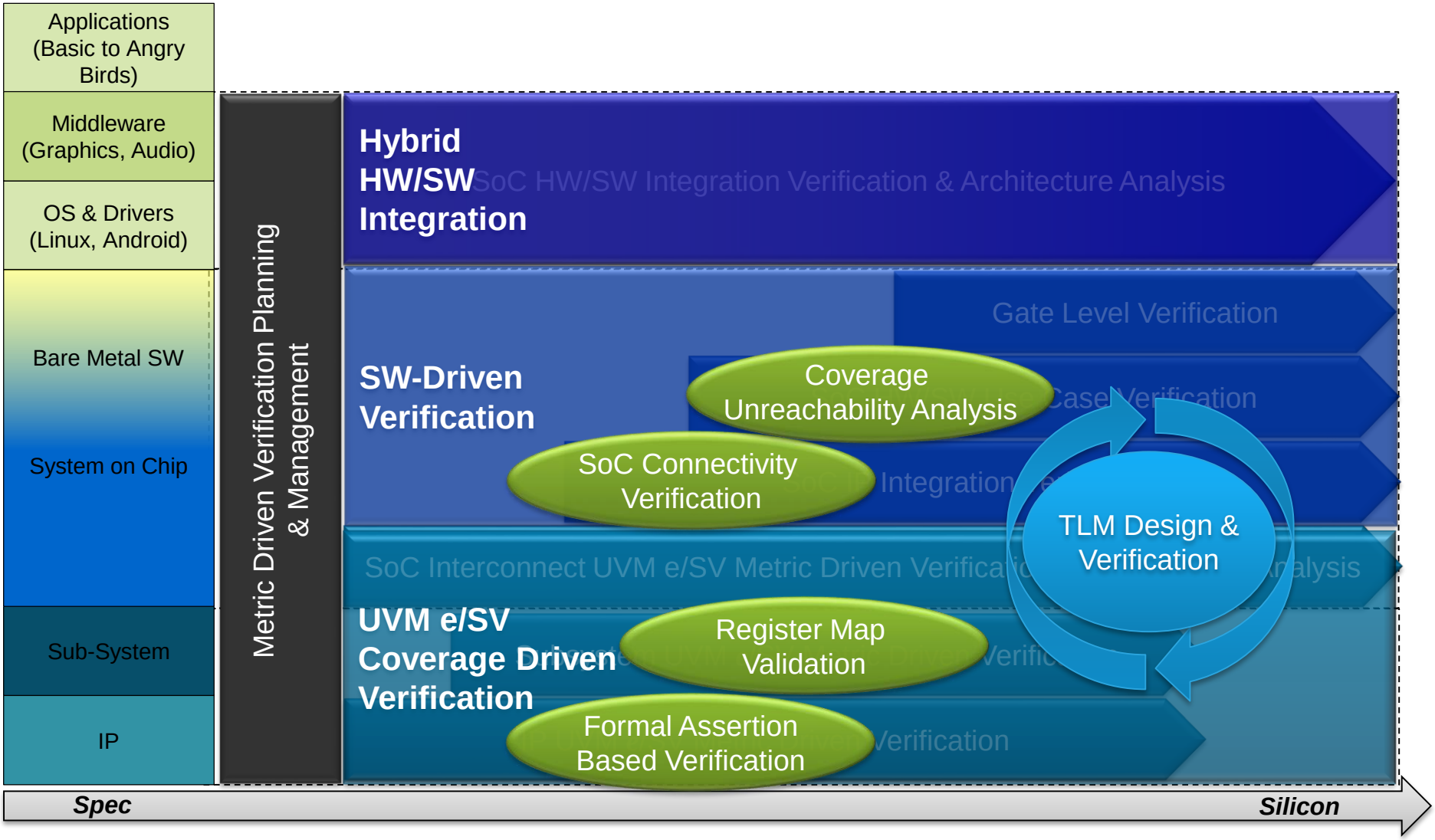
Develop SubSystem, SoC RTL & HW/SW Integration Verification Environments

Simulation Platform

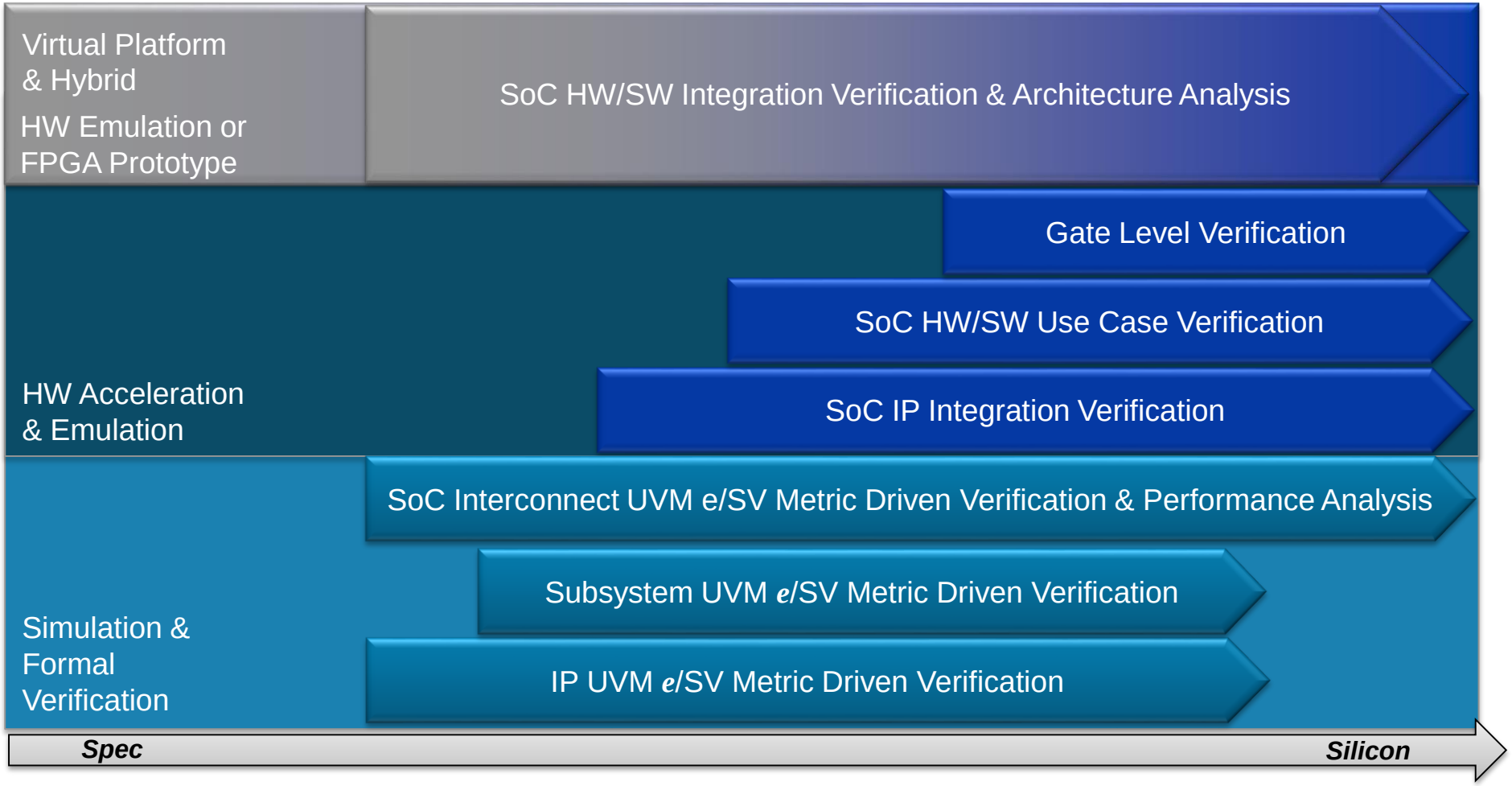
HW Accel Emulation Platform

cadence®

IP to SoC HW/SW Integration & Verification Flows



IP to SoC Pre-Silicon Verification Platforms



Expanding Requirements for Metric Driven Verification

- Consistent planning and management across different flows
  - CDV, Formal, Low Power, AMS, Use Case SW-Driven
- Need to support large-scale, multi-site SoC projects
  - Scalability of coverage merging and analysis
  - Scalability of aggregating & archiving data from different teams & sites
- Consistent metrics support across verification platforms
  - Simulation, Acceleration, Emulation, Virtual Platform
- Uniform metrics based project tracking from IP to SoC flows
  - Flexibility to “mine” verification database for customized reporting

cadence®

Section 1: Conclusions and Summary

- Key to optimized IP to SoC verification flow is choosing the best platform for the specific verification task with the right methodology
  - For efficient flow, requires highly integrated SoC development platforms
- Scalable metrics-based verification planning & management across multiple platforms and verification flows
- Early HW/SW Integration critical for fastest time to market
  - Must continually verify HW in SW context
- SW-Driven Verification best suited for SoC integration verification & use case verification
  - Horizontal reuse across virtual, simulation, emulation, & FPGA
- UVM SV/e MDV best suited for IP/Subsystem verification on RTL Simulator or HW Accelerator
  - Use TLM design & verification flow for more efficient development of new IP
  - Formal verification integrated for specific tasks to augment simulation-based verification

cadence®

Agenda

Section 1: MDV Methodology IP to SoC Verification

Section 2: MDV Approaches Beyond RTL IP Level

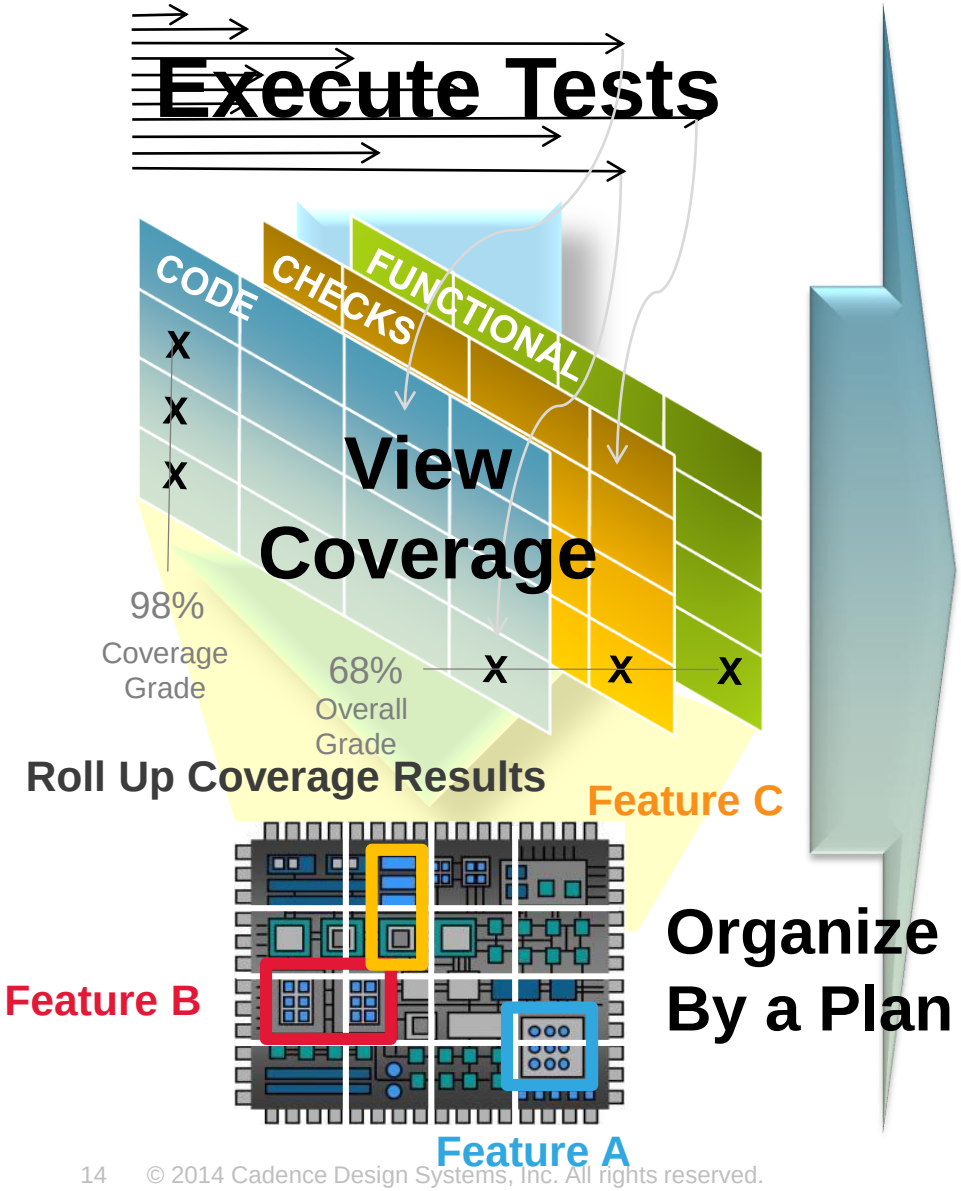
Section 3: Team Based Verification Management

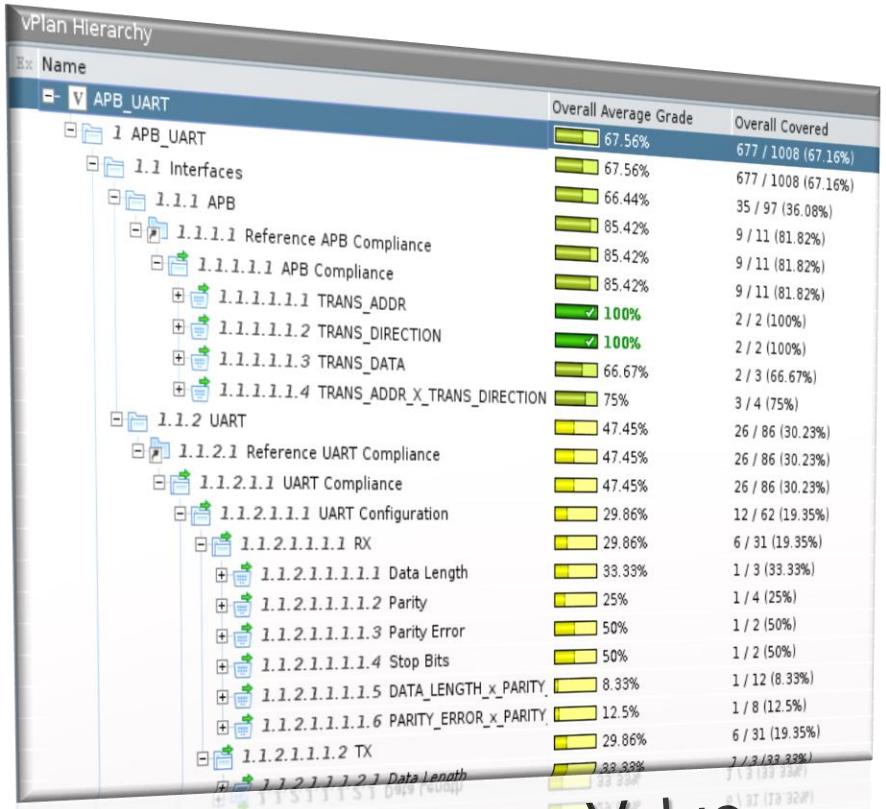
Section 4: MDV In Action

cadence®

MDV: Correlating Metrics with Verification Concerns

Data Driven Decisions and Objective Signoff Criteria





MDV Unique Value

cadence

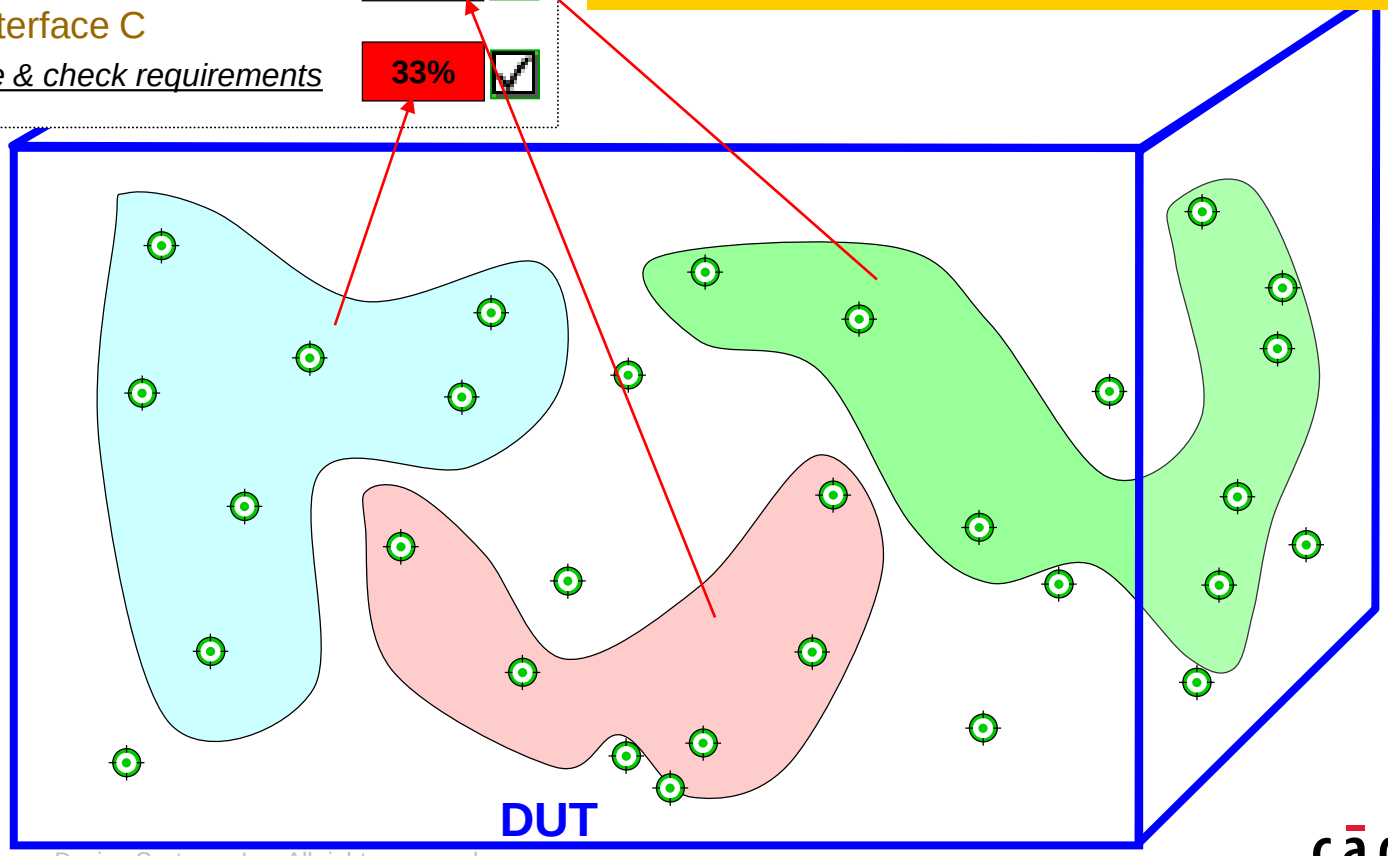
Planning is Essential

DUT Feature-Based Plan

- Input Interface A
  - *Coverage & check requirements* — 66%
- Core Function B
  - *Coverage & check requirements* — 100%
- Output Interface C
  - *Coverage & check requirements* — 33%

1. Plan Specifies Metrics Required for DUT Features:
   Verification Goals based on:
   - Analysis of specifications
   - Experience of the team

2. Plan Provides Feature Based Tracking of Progress
   - Implemented metrics to concretely measure Goals
   - Regression results annotated back to Plan Features



Benefits of an Executable Feature-based Plan

Without a vPlan

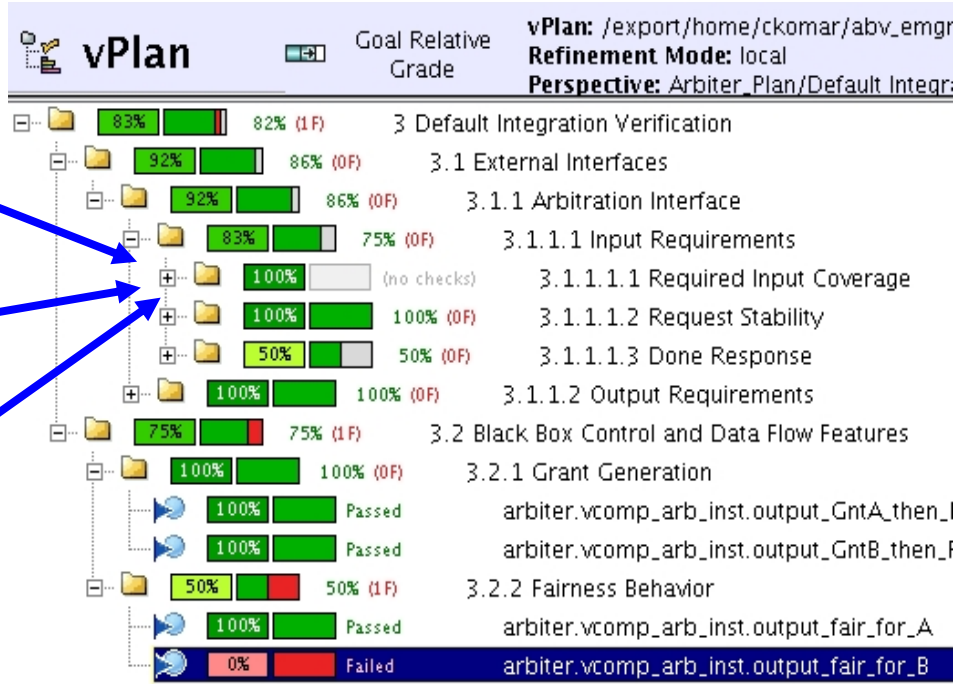
(**Coverage Driven Verification**)

Name	Count
Control-oriented Coverage	61 / 70
uart_tb_top.apbi0.apb_master_if0.assertPSelUnknown	960000
uart_tb_top.apbi0.apb_master_if0.assertPEnableUnknown	960000
uart_tb_top.apbi0.apb_master_if0.assertPWdataUnknown	960000
uart_tb_top.apbi0.apb_master_if0.assertPRwdUnknown	960000
uart_tb_top.apbi0.apb_master_if0.assertPAddrUnknown	960000
uart_tb_top.uif0.assertRxdUnknown	960000
uart_tb_top.uart_dut.ua_rcvr1.output_rx_data_ready	27
uart_tb_top.uart_dut.ua_rcvr1.output_rx_active	219712
uart_tb_top.uart_dut.ua_rcvr1.core_rx_fsm_d_stop2_to_d_idle	21
uart_tb_top.uart_dut.ua_rcvr1.core_rx_fsm_d_stop1_to_d_idle	6
uart_tb_top.uart_dut.ua_rcvr1.core_rx_fsm_d_stop1_to_d_stop2	4
uart_tb_top.uart_dut.ua_rcvr1.core_rx_fsm_d_stop1_to_d_stop2	17
Data-oriented Coverage	113 / 187
uart_tb_top.ovm_test_top.ve.apb0.bus_monitor.apb_transfer_cg	8 / 11
uart_tb_top.ovm_test_top.ve.uart0.Rx.monitor.rx_traffic_cg	6 / 8
uart_tb_top.ovm_test_top.ve.uart0.Rx.monitor.rx_protocol_cg	2 / 4
uart_tb_top.ovm_test_top.ve.uart0.Rx.monitor.uart_trans_frame_	17 / 37
uart_tb_top.ovm_test_top.ve.uart0.Tx.monitor.tx_traffic_cg	6 / 8
uart_tb_top.ovm_test_top.ve.uart0.Tx.monitor.tx_protocol_cg	2 / 4

- Without a vPlan, all coverage appears flat
- Difficult to correlate to verification plan
- Difficult to differentiate between high priority and lower priority coverage

With a vPlan

(**Plan based Metric Driven Verification**)



- With a vPlan, sections can be created to organize by feature areas of interest
- Various types of coverage/check metrics can be mapped to each section
- Very easy to measure progress relative to your plan and priorities

cadence

IP/Subsystem Verification Flow Concerns

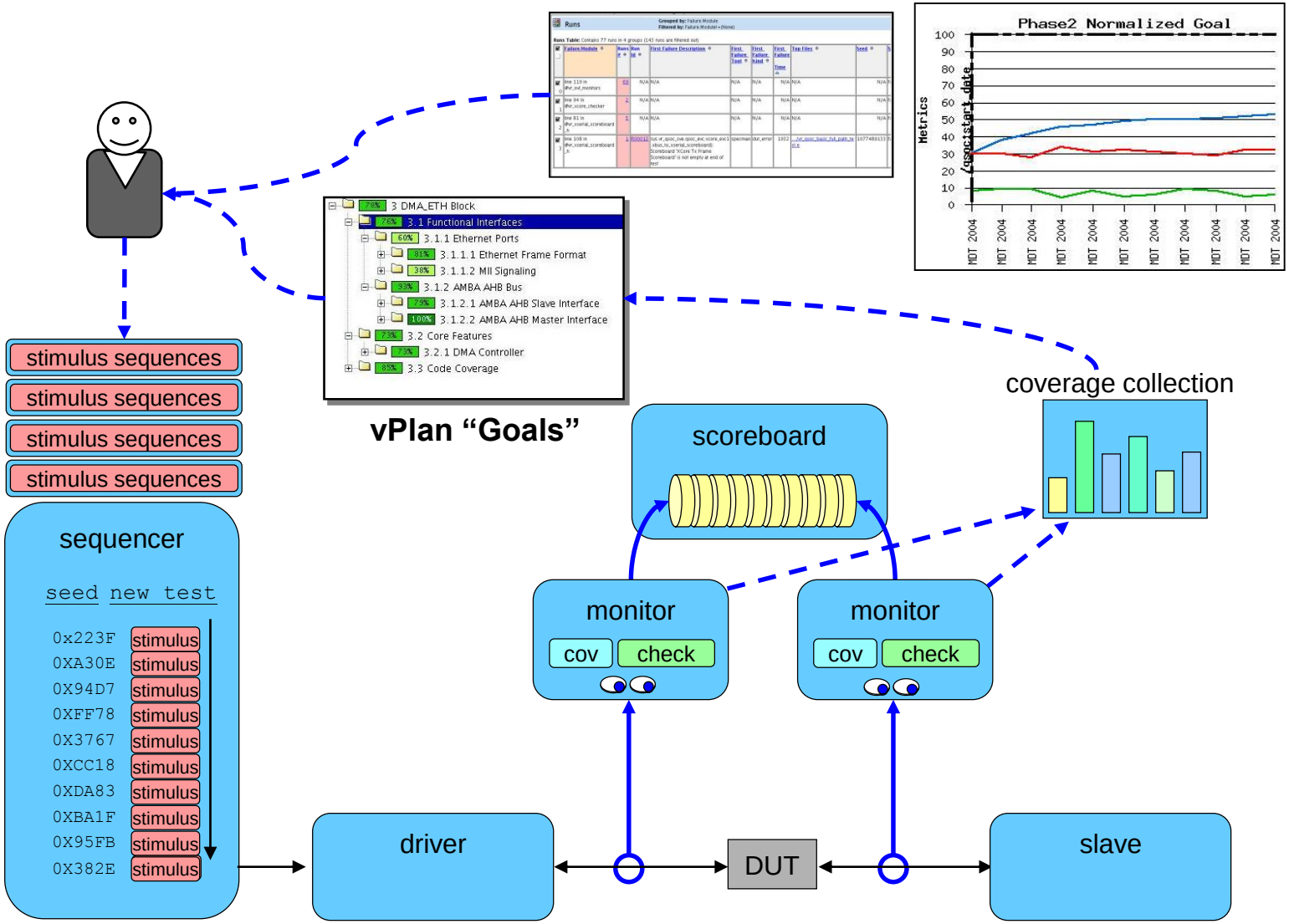
Must be very thorough for efficient SoC verification

- Verification Concerns
  - Interface protocol compliance
  - IP/Subsystem configuration, operations, and data paths
  - Low power modeling
  - Micro-architecture design features
  - Stress testing of complex traffic scenarios
- Create UVM e/SV IP/Subsystem Verification Environment
  - Augment with formal for block level and RTL linting
  - Commercial interface VIP for standard protocols
  - Reuse interface UVCs for proprietary protocols
  - Constrained-random stimulus sequences
  - Reference model, register modeling, and scoreboard for data checking
  - Assertions for protocol checking
  - Functional coverage for measuring features exercised
  - Code coverage for measuring HDL implementation exercised
    - Formal unreachability analysis of code coverage to reach 100%
  - Reuse IP Verification Environments to create Subsystem Testbench

cadence®

Traditional MDV Methodology

IP and Subsystem Verification



SoC Interconnect Verification & Performance Concerns

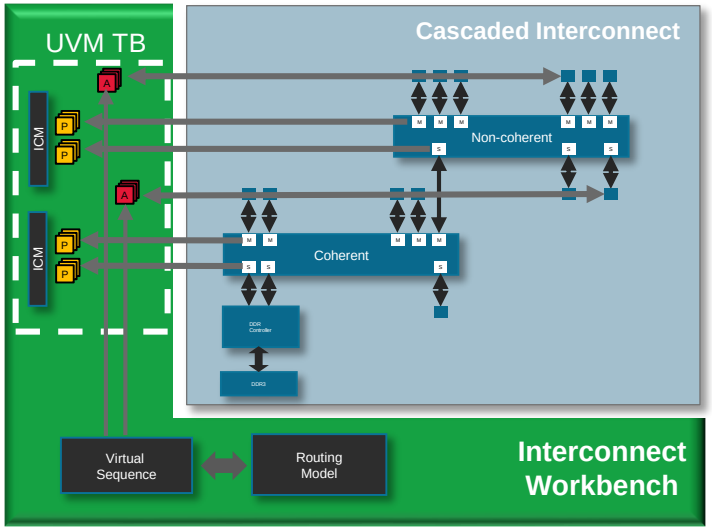
- SoC Interconnect includes hierarchy of connectivity across IPs and memories
- Interconnect Functional Verification
  - Address map and decoding
  - Configuration and address remapping
  - All Initiator to target paths
  - All target from initiator paths
  - Multi-protocol transaction transformations
  - Cache behavior for cache coherent interconnect
- Interconnect (and Memory subsystem) Performance Verification and analysis
  - Latency for critical data paths
  - Bandwidth and throughput for heavy traffic stress scenarios
  - QoS/QVN requirements
  - Cache performance for critical use cases

cadence®

SoC Interconnect Verification vPlan

Design Feature	Coverage Metric	Platform
Address map and decoding	Functional	Sim
Configuration and address remapping	Functional	Sim
All Initiator to all target paths	Functional	Sim
All target from all initiator paths	Functional	Sim
Multi-protocol transaction transformations across interconnect	Functional, Assertion	Sim
Cache behavior for cache coherent interconnect	Functional, Assertion	Sim

- Automatic generation of interconnect TB
- Built on UVM-based VIP
- Same Metrics as IP Verification



SoC IP Integration Verification Concerns

- Signal Connectivity in SoC
  - IP connectivity in SoC
  - Clock, interrupt, & reset connectivity
  - IO Pad connectivity
- IP Configuration, Primary Operations, & Data Path Connectivity in SoC context
  - SoC clocking & reset modes
  - IP access to Memory
  - IP I/O access and data path transaction flow
  - IP programmer's view and primary operations from SW Driver API
  - IP Interrupt scenarios
- IP Low power integration
  - Hierarchical low power control and power modes – power shut-off and voltage configurations
  - Low power interconnect and interface – isolation behavior

cadence®

SoC IP Integration Verification vPlan

Design Feature	Coverage Metric	Platform
IP Connectivity in SoC	Formal Assertion, Toggle	Formal Sim
Clock, interrupt, & reset connectivity	Formal Assertion	Formal
IO Pad connectivity	Formal Assertion	Formal
IP access to Memory	Functional, Toggle	Sim
IP I/O access and data path transaction flow	Functional, Toggle	Sim
IP programmer's view and primary operations from SW Driver API	Functional	Sim
IP Interrupt scenarios	Functional, Assertion	Sim
SoC boot/initialization scenarios	Functional, Assertion	Sim/Accel
Hierarchical low power control and power modes – power shut-off & voltage configs	Functional, Assertion	Sim/Accel
Low power interconnect & interface – isolation behavior	Functional, Assertion	Sim/Accel

cadence®

SoC Use Case Verification Concerns

- SoC level features
  - SoC FW boot up and initialization
  - Primary IO Pad configurations
  - Scan chain connectivity and test mode operations
- End application use case scenarios
  - Verified on firmware or lower layers of SW stack
  - Adherence to power and performance requirements
  - Cache and IO Coherency
  - End to end data path scenarios
    - E.g., CPU programs camera -> camera sends image data -> CPU processes image -> image sent to display
  - Stress tests on resource contention and multi-master scenarios
  - Cross use case scenarios with low power configurations, modes & sequencing

cadence®

SoC Use Case Verification vPlan

Design Feature	Coverage Metric	Platform
SoC FW boot up and initialization	Functional, Assertion	Sim/Accel
Primary IO Pad configurations	Functional, Toggle	Sim/Accel
Scan chain connectivity and test mode operations	Functional, Assertion	Sim/Accel
Cache and IO Coherency	Functional, Assertion	Sim/Accel
End to end data path scenarios – functional, power, & performance	Functional, Assertion	Sim/Accel
Stress tests on resource contention and multi-master scenarios	Functional, Assertion	Emulation
Cross use case scenarios with low power configurations, modes & sequencing	Functional, Assertion	Emulation

cadence®

SoC Gate Level Verification Concerns

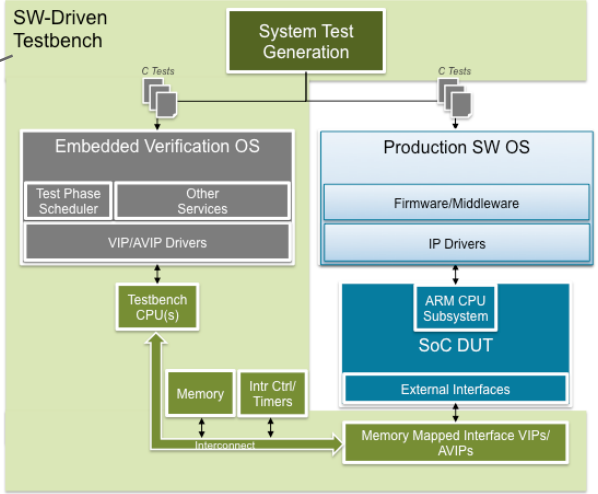
- Gate Level Focuses on a critical sub-set of concerns
  - Tests to be run in zero delay mode
    - Reset verification, Initialization, & verification of clocking
    - Basic heart beat test to detect functional issues or issues related to X mismatches
    - Verify unexpected synthesis transformations
    - Validate functional effects after DFT and Low Power insertion
  - Tests to be run with timing
    - Tests to cover/verify STA timing constraints like multi-cycle paths, false paths
    - Test to cover asynchronous paths
    - Verify DFT with timing
    - CDC verification because automatic CDC failing too much at SoC level
    - Validation of physical netlist low power implementation
    - Safety standards on reliability testing via Fault insertion
- Uses same environment as for SoC Use Case Verification
  - Except for scan chain verification and other physical netlist artifacts
  - Same metrics and engines used as well
    - Metrics: Black box Functional, Assertion, Toggle
    - Engines: Sim/Accel

cadence®

SoC HW/SW Integration Verification Concerns

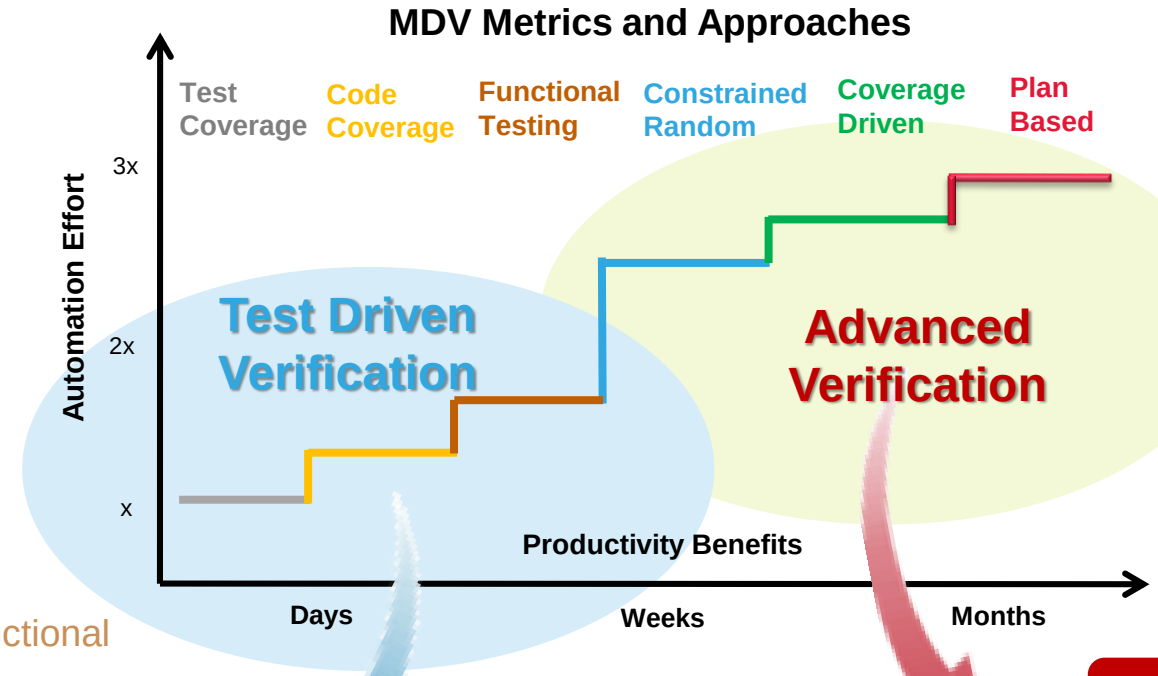
- Key concerns
  - Integration & bring-up of OS & higher SW layers on RTL SoC
    - Debug integration issues on pre-silicon emulated HW platform
    - Validate OS boot up
    - Validate middleware and real applications on SoC platform
    - Validate performance requirements
  - Validate dynamic power usage for critical applications
    - Based on real running real SW application snippets
  - Graphics GPU OpenGL SW API compliance

- Effective Approaches
  - Use-cases, scenarios, and functional metrics
  - Using SW-Driven testbench approaches
  - Leverage Emulation & FPGA Prototypes



MDV Metric Options

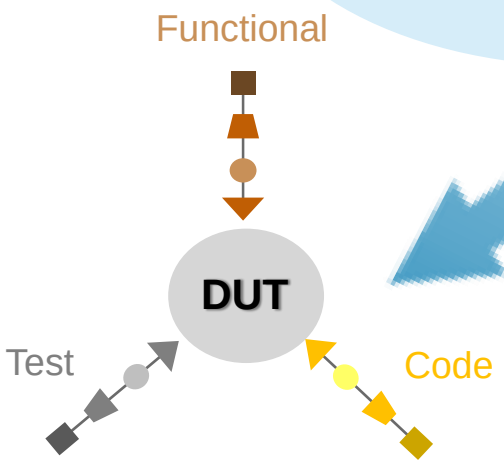
Measuring the right metrics for the task at hand



More effort, but more effective if resources permit

Leverages advanced planning technology

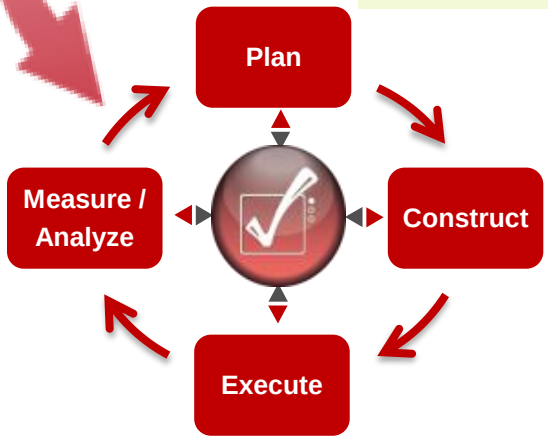
Greater ROI



Quickly establish test area

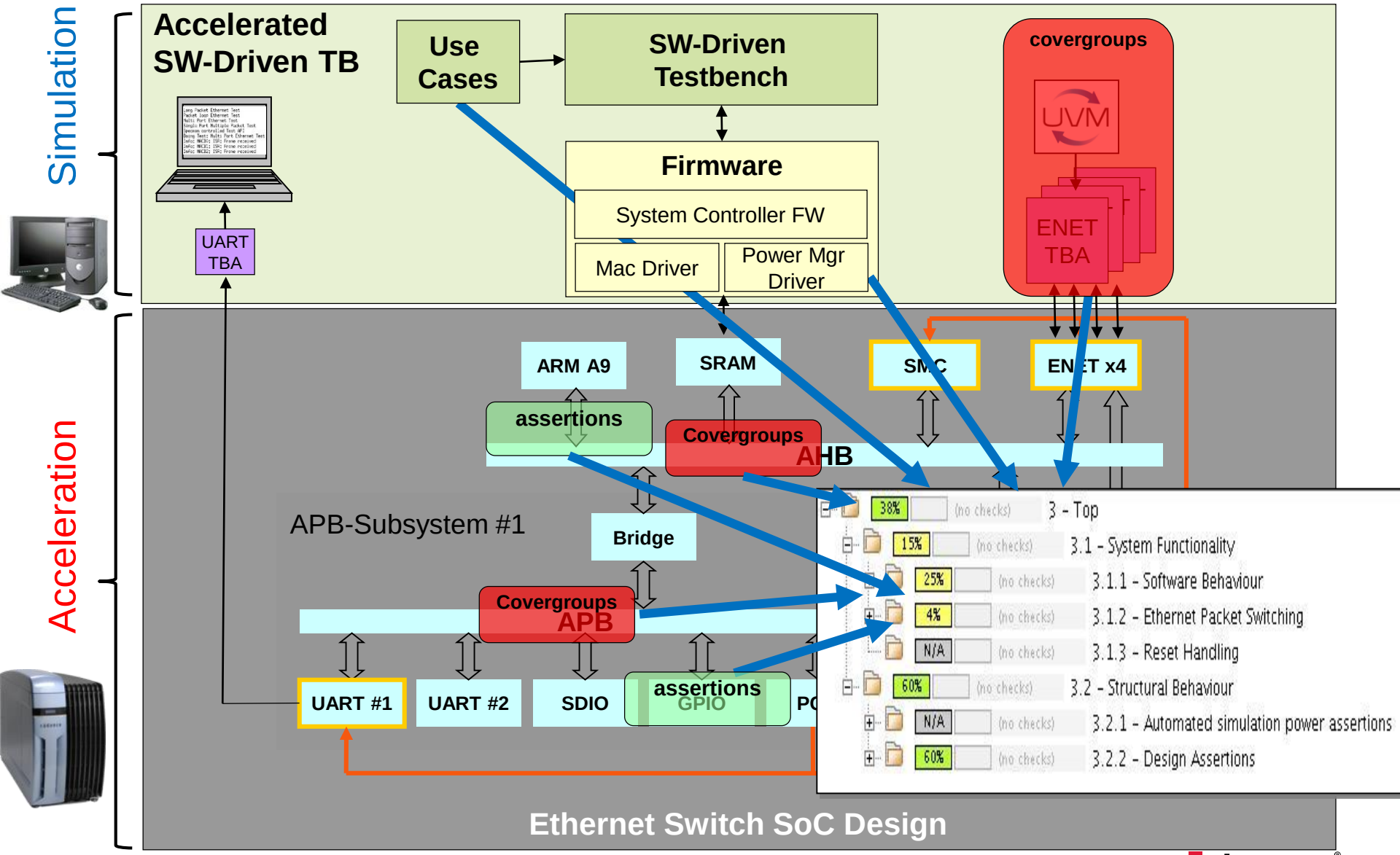
Give credit / track designers work

Effective test driven verification environment



cadence®

SoC Verification Metrics Mapped to the Plan



cadence

SoC MDV Enabler – Multi Engine, Multi Metric Plan

Executable verification plan that can link to all necessary engines and metrics

vPlan editor

Switch Verification Plan

- 1. SoC
  - 1.1. HW/SW Integration [VIRTUAL PROTOTYPE]
  - 1.2. Use Case [SW DRIVEN]
- 2. Subsystem
  - 2.1. ENet [SIM-ACCEL]
  - 2.2. IO Subsystem
    - 2.2.1. Interconnect Verification [SIMULATION]
    - 2.2.2. IP Connectivity [FORMAL]
    - 2.2.3. Power Intent [SIMULATION]
- 3. IP
  - 3.1. UART [SIMULATION]
  - 3.2. SMC [FORMAL]
  - 3.3. PLL [MIXED SIGNAL SIM]
  - 3.4. GPIO [SIMULATION]
  - 3.5. SPI [SIMULATION]

Testcase Metrics

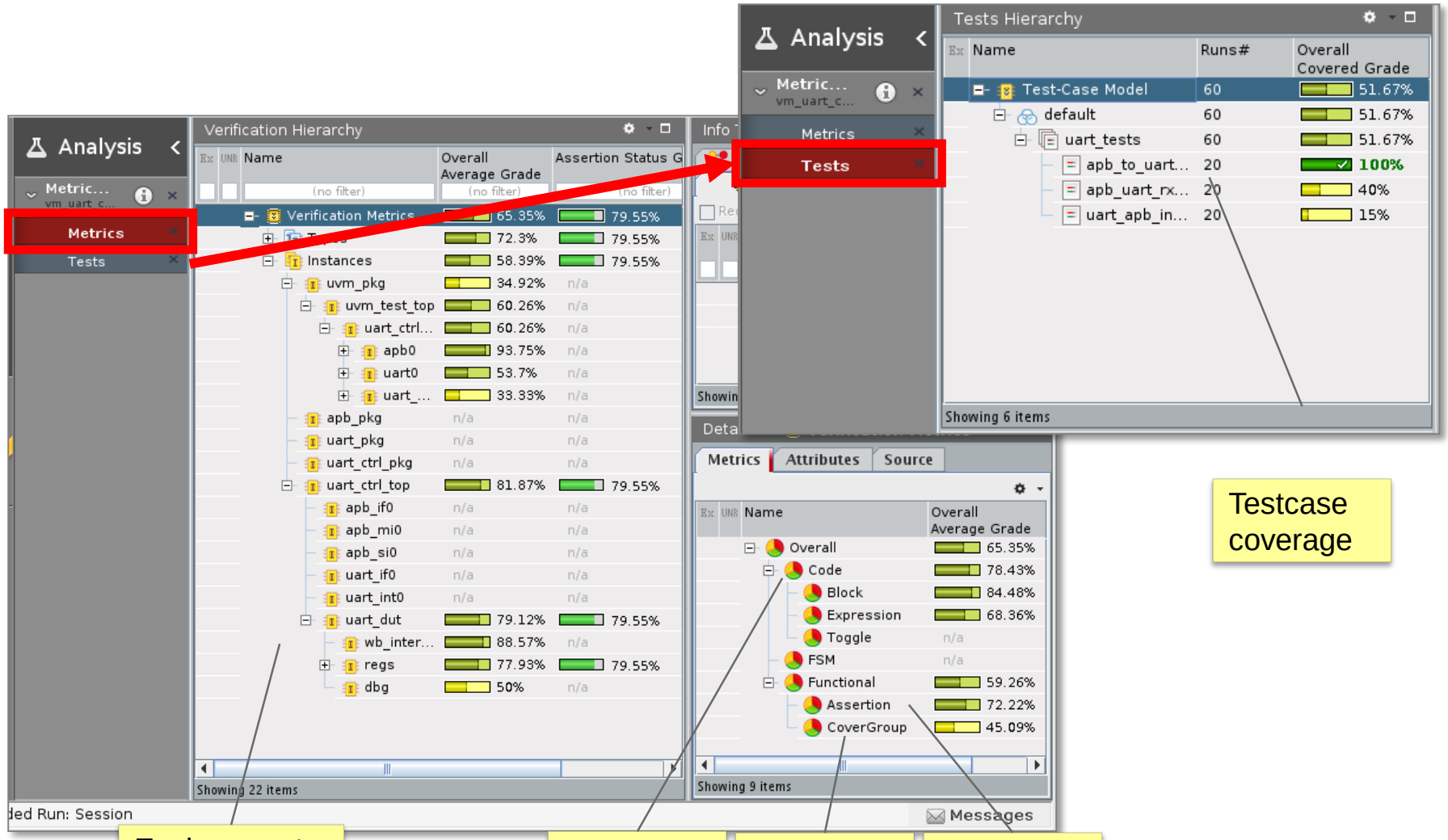
Functional Metrics

Assertion Metrics

cadence

SoC MDV Enabler - Manage All Metrics in One Spot

Multi Engine, Multi Metric results collection in unified environment



cadence

Agenda

Section 1: MDV Methodology IP to SoC Verification

Section 2: MDV Approaches Beyond RTL IP Level

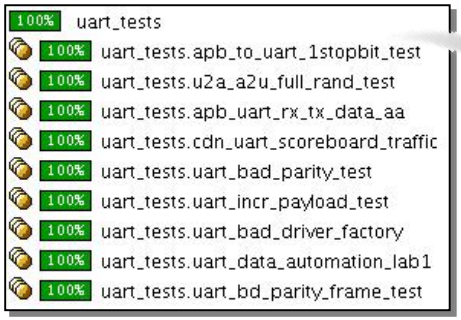
Section 3: Team Based Verification Management

Section 4: MDV In Action

cadence®

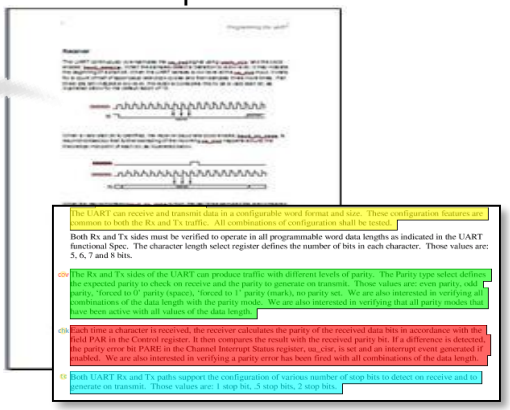
Team MDV: *It Still Starts with a Plan!*

Legacy tests

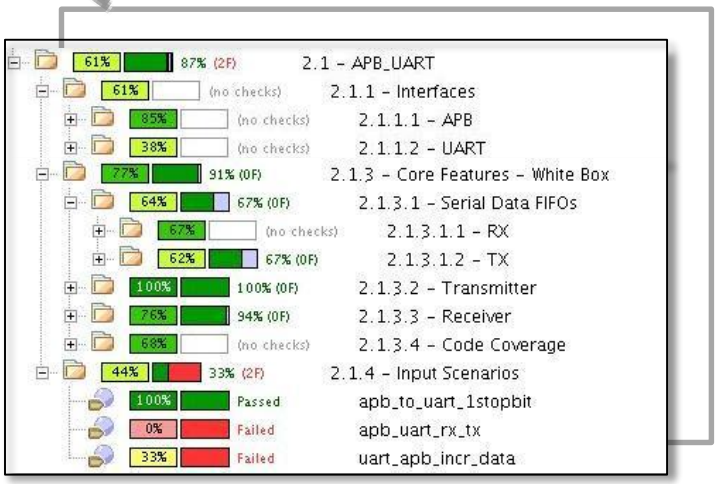


Requirements Management System

Outline from a Functional Spec



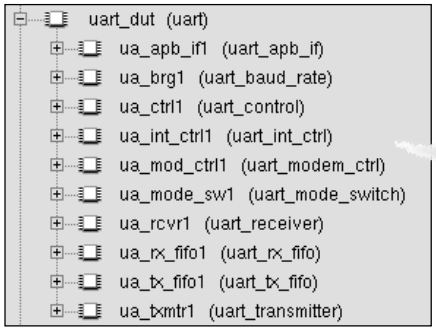
Heterogeneous Verification Tools (ie Formal,Simulation)



Distributed / Hierarchical Plans

VIP Compliance vPlan and module level vPlans

Code coverage and other metrics

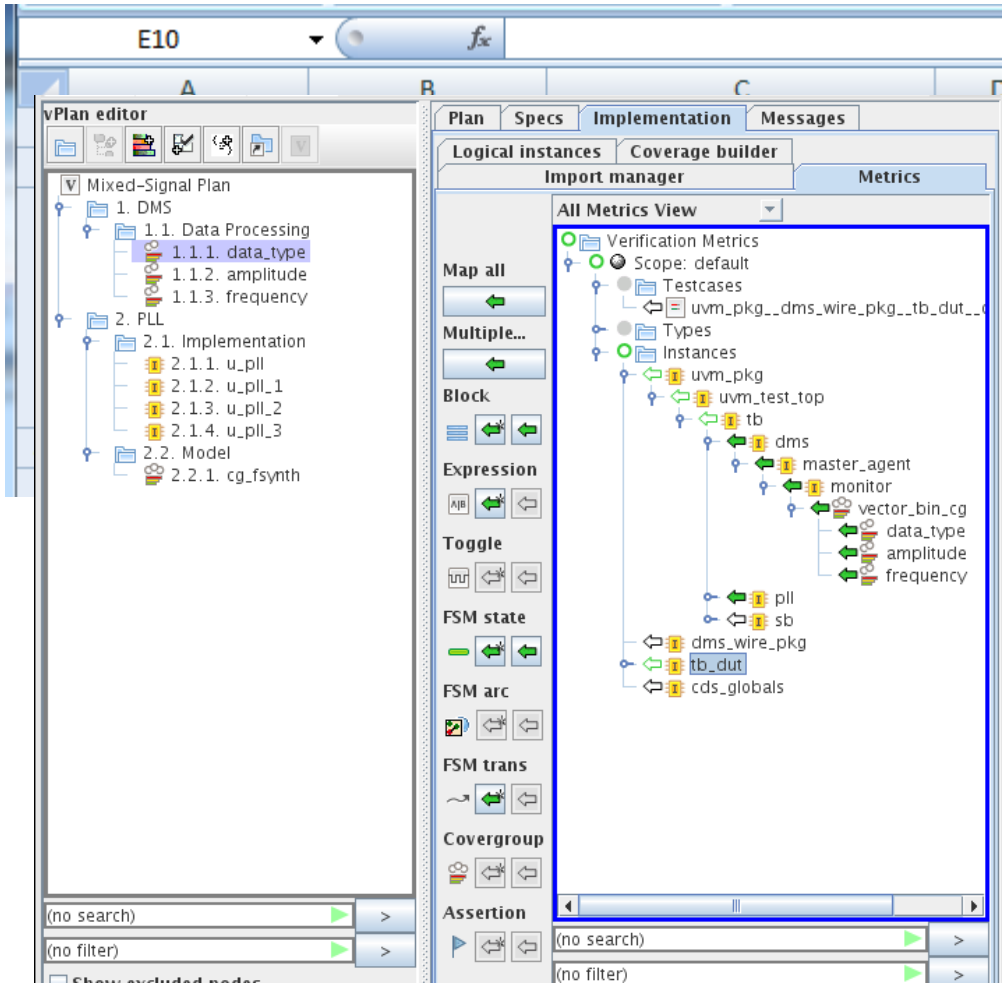


Brainstorming

The verification plan becomes the anchor to connect teams and technologies together

cadence®

Plan Composure and Creation: Scalability!

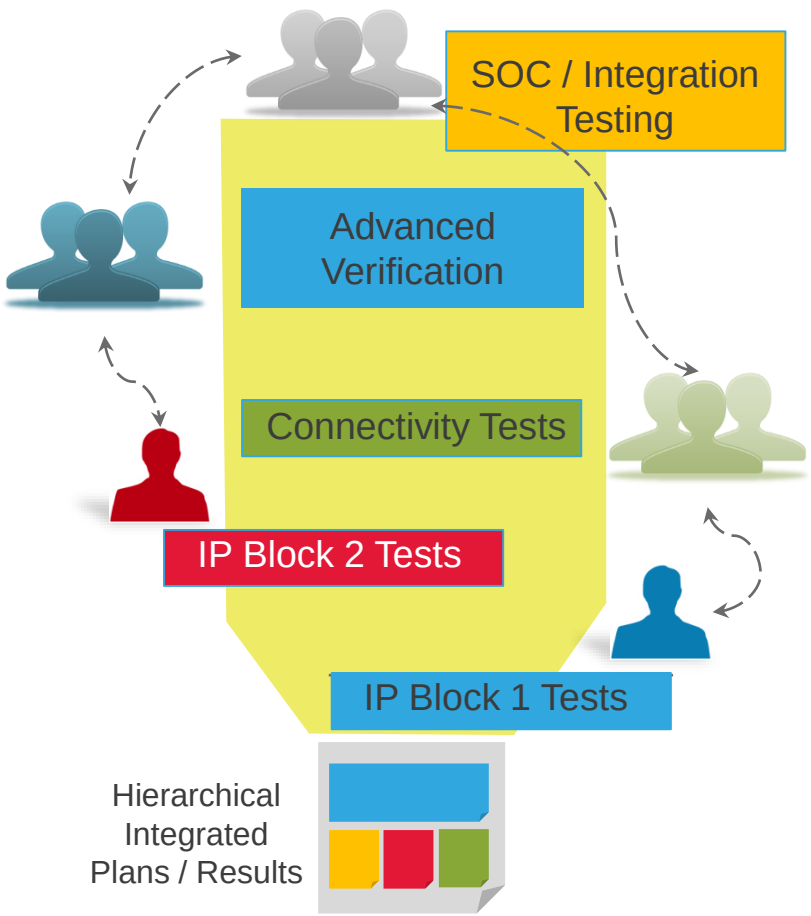


- Long paths mapping metrics to plan
- Issue compounded across engines
- Further worsens at great levels of integration

- Connection to data during plan composure enables efficiency
- Export/Import to/from popular formats (XML, CSV, HTML) enables scripting, publishing, etc
- Resultant plan is mapped "Correct by construction."

cadence

MDV for the SoC Team

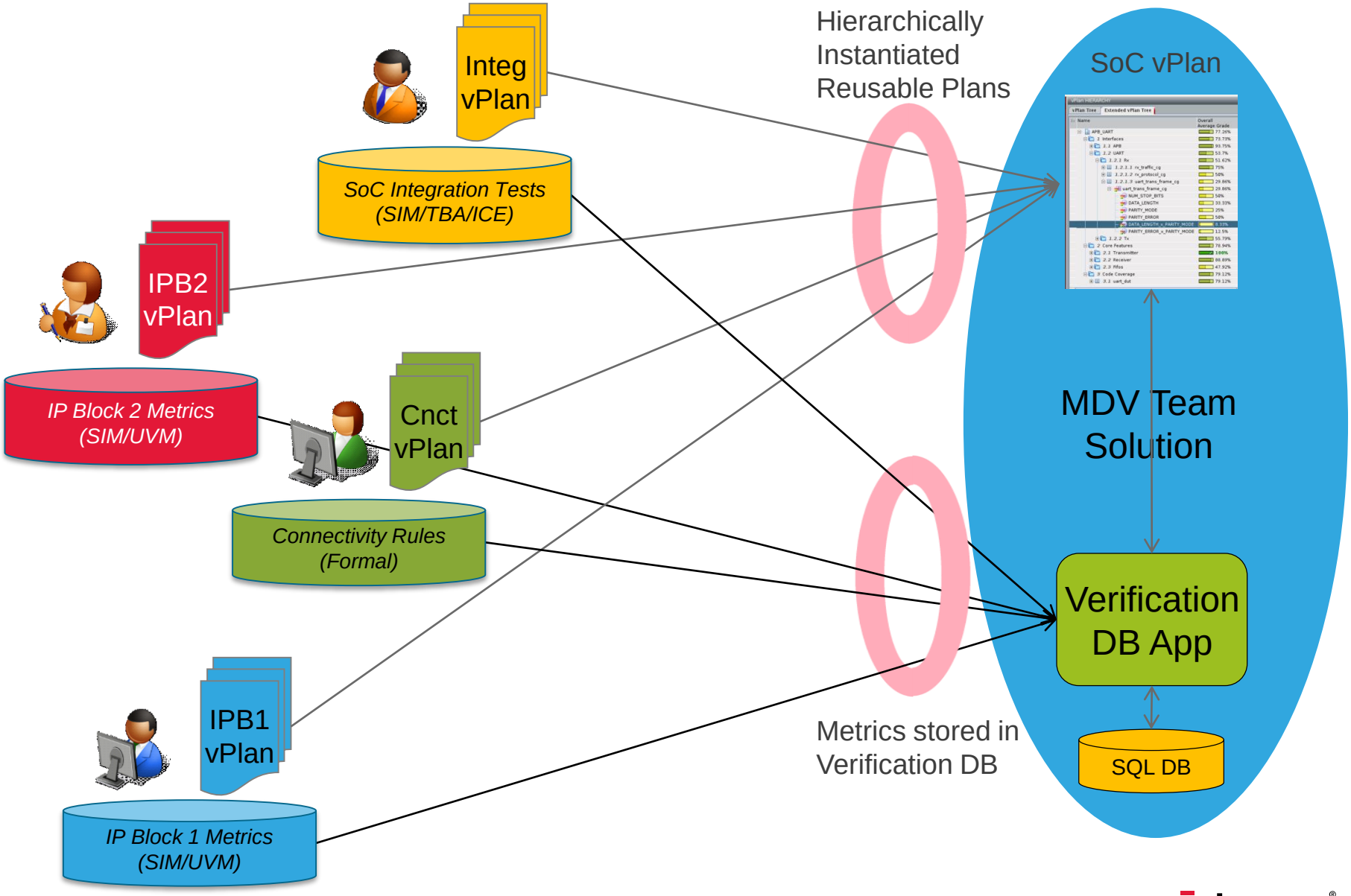


Simulation Formal Acceleration Emulation

- Disparate islands of information
- Inconsistent and incompatible verification approaches
  - Verification methodologies
  - Different levels of integration
  - Design technologies
- Everyone contributes, but no single coordinated view of who is doing what and how
- Goal: provide an independent yet integrated [multi-user] metric management and Plan to Closure methodology

cadence

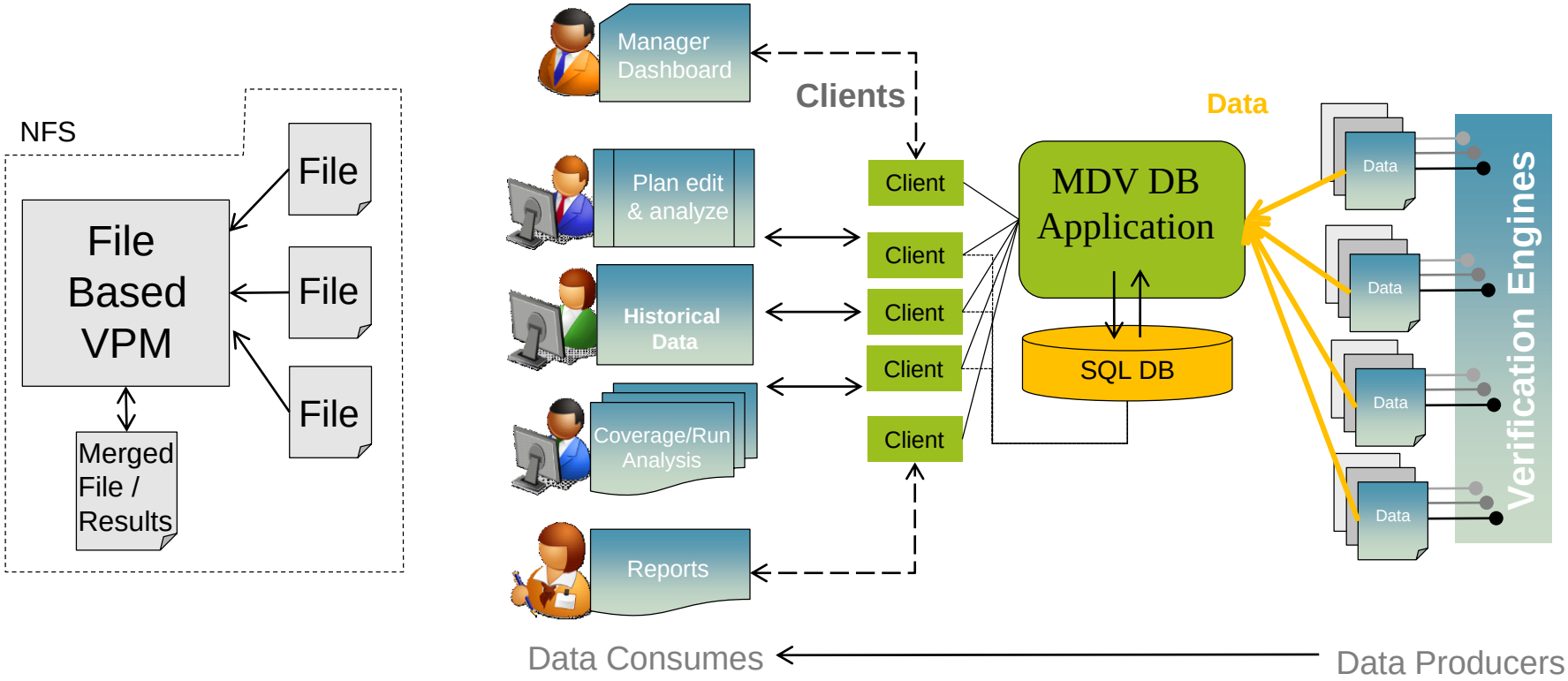
Team MDV – Multi-user, Multi-engine, Multi-analysis



cadence

Enabling the SoC Verification Team with MDV

Next generation MDV Architecture



- File based data mgmt does not scale
- Data does not inherently stay synchronous
- Single User Environment – Difficult to Share
- Static data – reporting is manual / intensive
- Batch coverage merge not suited to 24/7 runs

- DB gives orders of magnitude greater scaling
- Data synchronicity throughout life of a project
- Multi User Environment – Easy to Share
- Dynamic – fresh data, built-in real time reports
- Continuous operations mode / “always on”

cadence

Database Driven Architecture

Complete project data

Sessions:

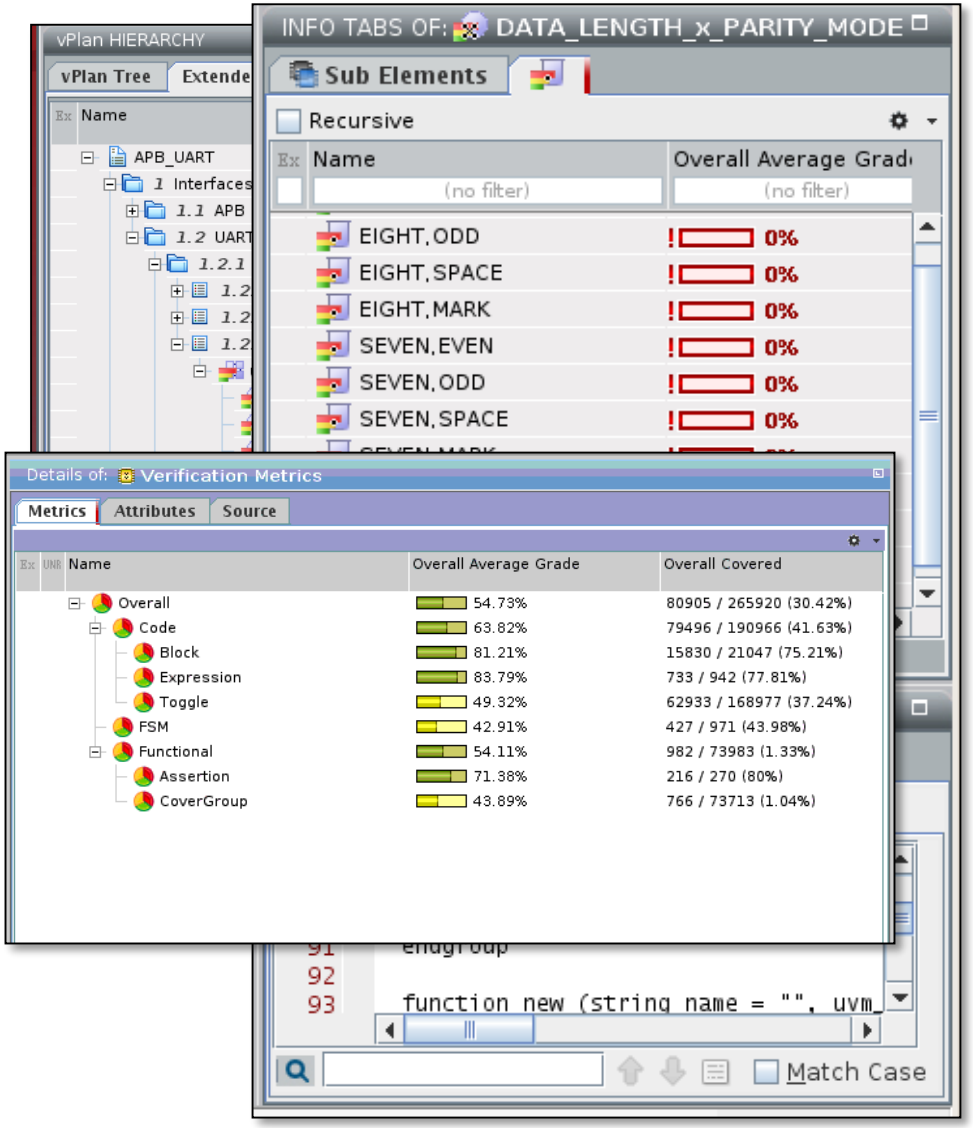
Session Status	Name	Total Runs	#Passed	#Failed	#Running	#Waiting	#Other	Start Time	Owner
(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)
completed	UART_Block.14_02_07_13_...	12	6	6	0	0	0	2/7/14 1:48 PM	johnn
completed	SMC_Block_Formal	12	11	1	0	0	0	2/3/14 11:54 AM	pcarzola
completed	APB_Subsytem.14_02_03_...	8	8	0	0	0	0	2/3/14 11:29 AM	johnn
completed	UART_compact_2014-02-01	5	3	2	0	0	0	1/31/14 11:46 AM	nsegal
completed	UART_compact_2014-02-02	5	3	2	0	0	0	1/31/14 11:44 AM	nsegal
completed	UART_Block.14_01_31_10_...	12	7	5	0	0	0	1/31/14 10:57 AM	johnn
completed	UART_Block.14_01_31_10_...	12	5	7	0	0	0	1/31/14 10:47 AM	johnn
completed	DMS_sanity_1	1	1	0	0	0	0	1/29/14 1:30 PM	magraham
completed	DMS_smoke_2014_1_29	2	2	0	0	0	0	1/29/14 1:30 PM	magraham
completed	Formal_Interconnect_Verifi...	1	0	1	0	0	0	1/27/14 11:54 AM	pcarzola
completed	LowPower	1	1	0	0	0	0	1/24/14 2:18 PM	johnn
completed	Unreachability	3	3	0	0	0	0	1/24/14 2:04 PM	nsegal

Direct access to regression data for deeper analysis.

cadence

Requirement - Unified Analysis Environment

- Analysis, exclusion and reporting
- Top level verification plan down to low level bin/line/toggle level analysis
  - Historically split between multiple tools (spreadsheet, scripts, single run coverage analysis tools)
- Single environment for ALL metric analysis
  - The right data at the right time
  - Low latency access (seconds, single click)



cadence

Requirement - Unified Analysis Environment
Includes Failure Triage

- Failure analysis complements metric roll up in MDV Cockpit
- Integration and automation with debug is a natural fit
- Push button automated rerun with dumping of debug data
- Tight integration with advanced debug platforms
  - e.g. Cadence Incisive Debug Analyzer

cadence®

Unified Analysis Environment

Failure Triage Included

Groups of Errors Pre-Grouping Filter: ((Severity IN [error, critical]) AND (Category IN [first

Name	Description	Context	Severity	Category	Number Of Entities
UVM_ERROR	[SCRBD] ####### FAIL : APB...	/grid/avs/inst...	error	first	27
UVM_ERROR	[SCRBD] ####### FAIL : uart...	/grid/avs/inst...	error	first	2

Showing 2 items

Failure Binning

Errors UVM_ERROR

Name	Description	Context	Time	Severity
UVM_ERROR	[SCRBD] ####### FAIL : uart0 R...	/grid/avs/inst...	400500	error
UVM_ERROR	[SCRBD] ####### FAIL : uart0 R...	/grid/avs/inst...	408500	error

Showing 2 items

Failures in Bin

Runs of: UVM_ERROR

Index	Name	Status	Duration (sec.)
6	/uart_tests/apb_uart_rx_tx	failed	3

Showing 1 items

Run Producing The Failure

Details of: UVM_ERROR

Attributes | Logs

local_log.log

```
322 UVM_INFO /grid/avs/install/incisiv/13.1/lat
323 UVM_ERROR /grid/avs/install/incisiv/13.1/la
324 UVM_INFO /grid/avs/install/incisiv/13.1/lat
325 UVM_INFO /grid/avs/install/incisiv/13.1/lat
326 UVM_INFO /grid/avs/install/incisiv/13.1/lat
327 UVM_INFO /grid/avs/install/incisiv/13.1/lat
328 UVM_INFO /grid/avs/i...isiv/13.1/lat
329 UVM_INFO /grid...13.1/lat
330 UVM_INFO /gri...3.1/lat
331 UVM_INFO /gri...13.1/lat
332
333 --- UVM Report catcher summary ---
```

Log file with Error in context

Match Case

un: Session | Busy... | Messages

cadence

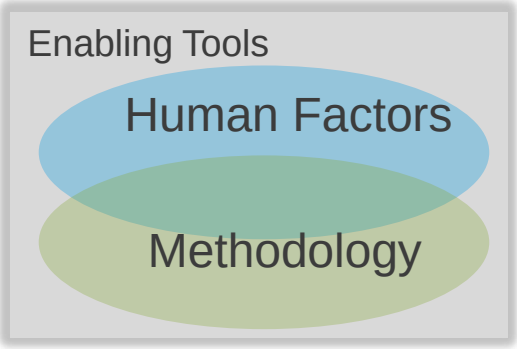
React Real-time to Trends

- Utilize “One Touch” real time access to up-to-date results
- Track critical verification indicators over time for visibility and predictability

- Project Definition
  - Set of data
  - Metrics to track
  - Criteria for sample
- Project Tracking and Analysis
  - Graphical and textual presentation of the metrics results over time
  - Persistent storage of trend data in the DB enables team access

cadence

Practical guidelines



- Consider the intangibles upfront
  - Human factors and verification methodology
- Plan
  - Leverage plans built at all levels of integration, with metrics from all available engines
  - Expedite plan composure with access to metric definition information
  - Instantiate IP level plans for SoC plan creation efficiency
- Collect
  - Take credit for work already done ➔ aggregate results across users, engines, time
  - Metrics must be easily accessible (view, report, query) ➔ utilize common database architecture
- Analyze
  - Snapshot results at regular intervals
  - Find trends, filter blips (charts, reports)
- React
  - Objective Data ➔ Exploit connection of metrics to plan, spec
  - Instant appreciation of project wide effect of decisions based on real time data

cadence®

Agenda

Section 1: MDV Methodology IP to SoC Verification

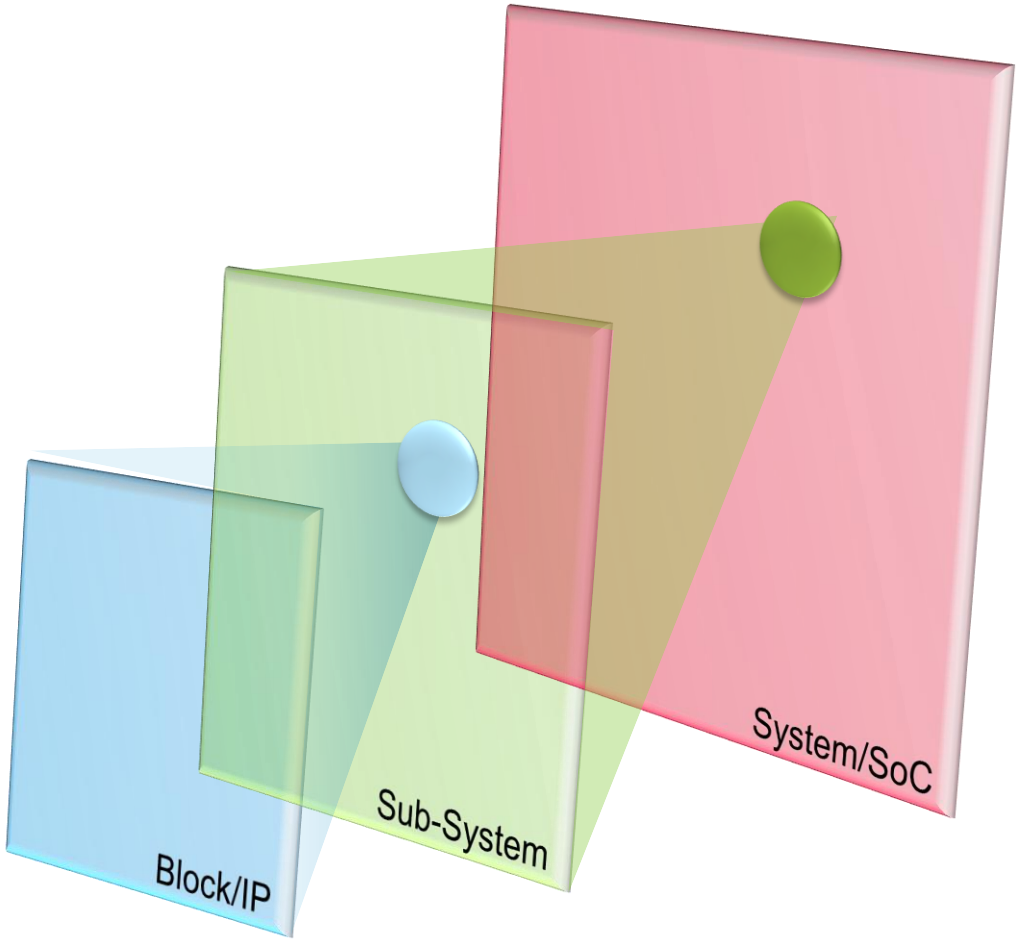
Section 2: MDV Approaches Beyond RTL IP Level

Section 3: Team Based Verification Management

Section 4: MDV In Action

cadence®

Apply coverage at several stages of development cycle



System/SoC
- Expand coverage analysis with live interfaces and real software/firmware execution
- Use coverage techniques to optimize designs and software tests
- Continue to use assertions to ensure correctness and localize problems

Sub-System
- Create realistic scenarios and transactions to exercise interfaces
- Continue to use assertions to ensure correctness and localize problems
- Use code and functional coverage to monitor interfaces and testbench effectiveness

Block/IP
- Create synthetic scenarios to hit paths
- Use assertions to ensure correctness
- Use code and functional coverage to monitor interfaces and testbench effectiveness

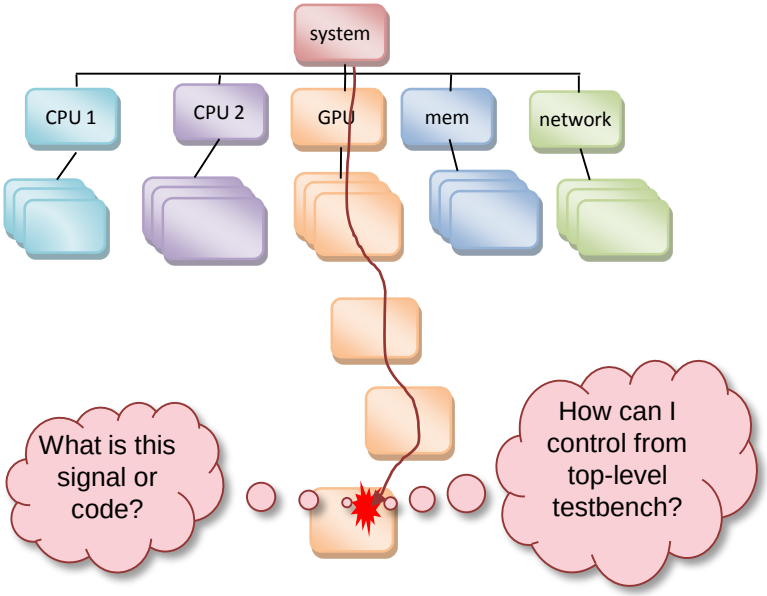
cadence

Coverage
Use Cases

Use Cases	User Explorations (examples)	Applicable Coverage	
		Code	Functional
SoC Integration Verification	• What is the activity between sub-blocks? • What is the top level activity—perhaps 1 or 2 levels?	✓	
Localized and Full-design focus	• How can I run detailed coverage analysis into specific area of interest? • How do I achieve 100% coverage?	✓	
Verify Modes of Operation	• Are two processing units simultaneously active? Were interfaces active simultaneously? Was interrupt issued when CPU transfers data to GPU? • How do I correlate coverage to design features that I'm testing and measure progress against my overall verification plan?		✓
Design Optimization	• How is this buffer being used? Undersized? Oversized? • What is the latency on this operation? Average? Max?	✓	✓
Improving Hardware Coverage of Software Tests	• How much of hardware is being exercised by software tests? • Should I improve my software tests to achieve higher coverage?	✓	✓

cadence®

Code coverage problem statement

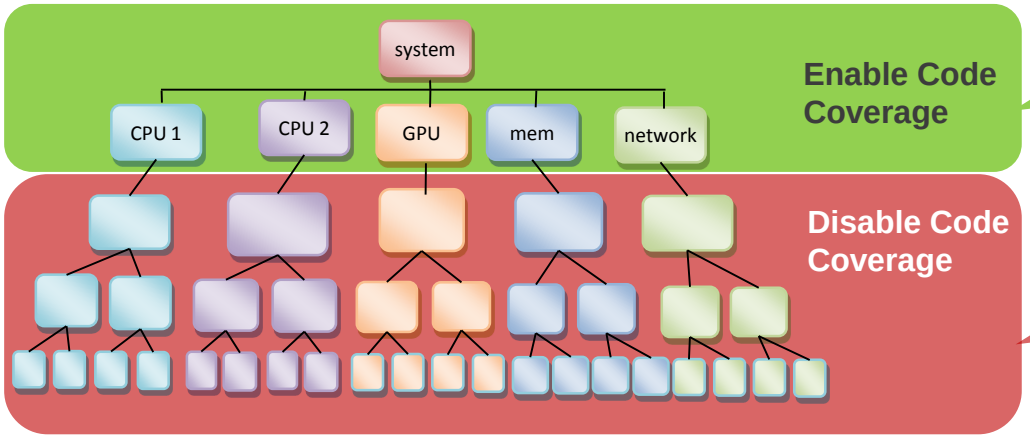


- Traditional code coverage use model is difficult
  - Add an option and get overwhelmed with data
  - System verification engineers aren't going to understand coverage data at low levels of the design
  - Even if they did, very difficult to influence low level logic from system level tests
- Solution?
  - Focus on actionable data

Integration verification

Code	Functional
✓	

GPU Application (26 MGate)



# hier levels	# toggle signals
1	3,000
2	101,000

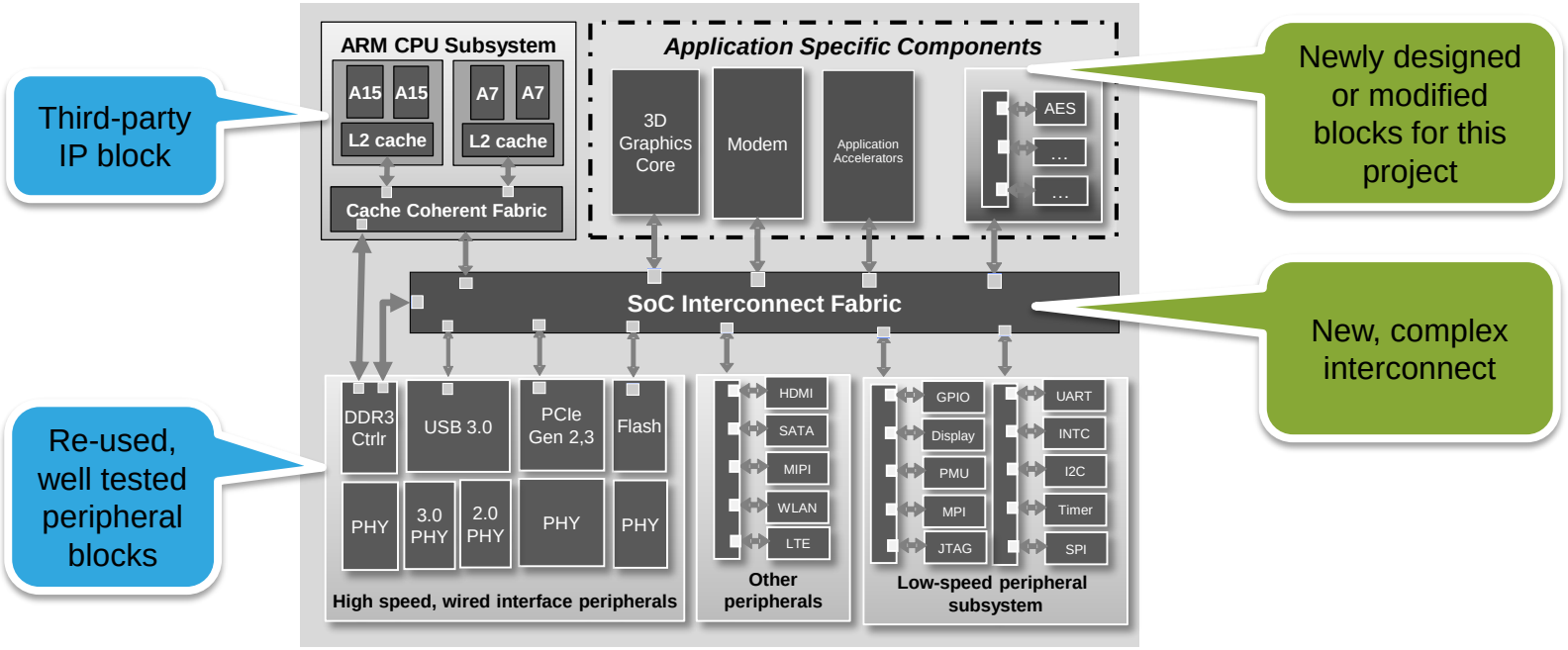
# hier levels	# toggle signals
3	717,000
5	14,400,000
all	27,400,000

- Cover connectivity between top-level modules
  - That's what's new and untested
  - Lower level blocks have been verified at the block level
  - Understandable and actionable by system verification engineers
  - Typically would use toggle coverage on ports of top-level blocks
  - Block coverage not as interesting at higher levels → limited RTL
    - Might have small pieces of new system-level controller logic

cadence

Localized focus—go deep

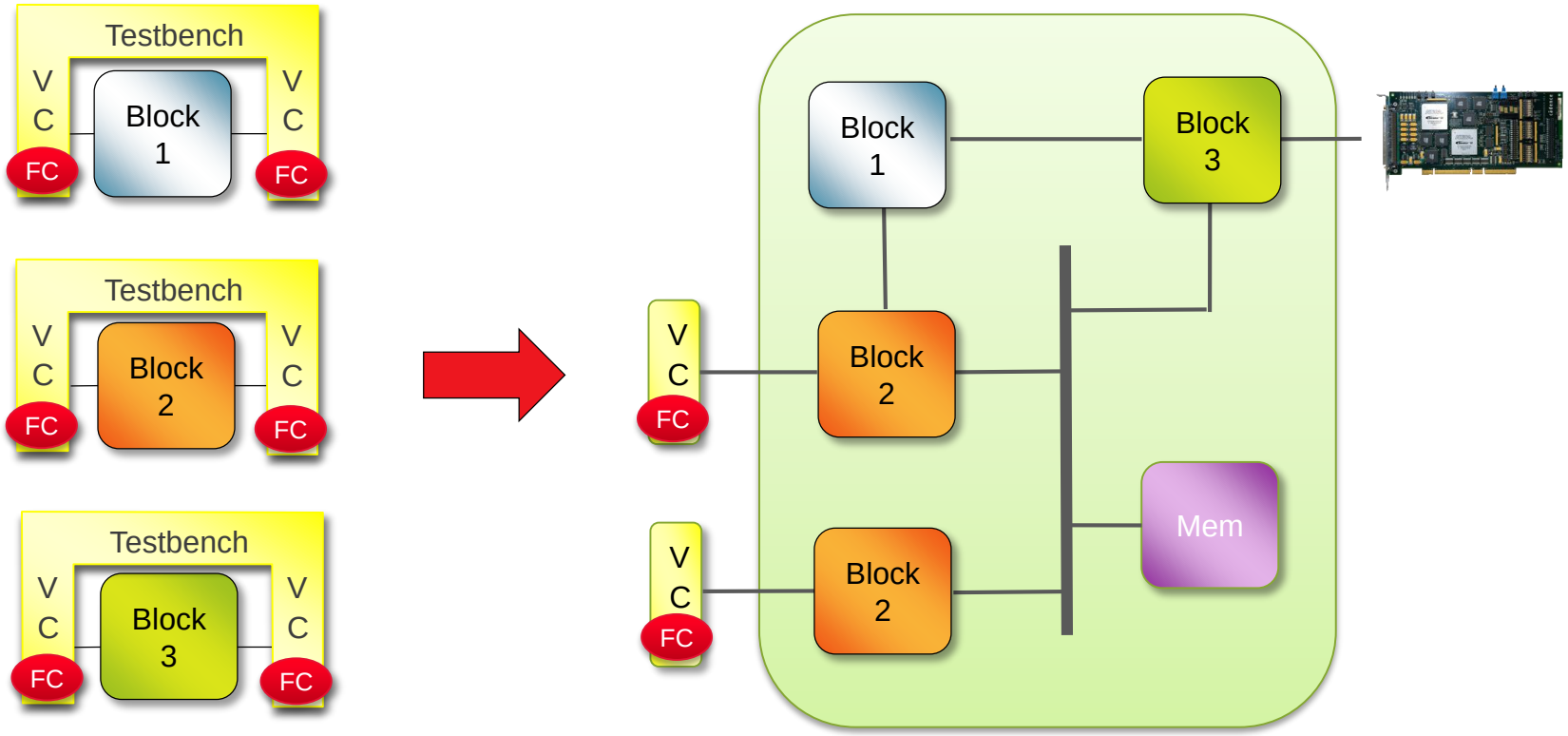
Code	Functional
✓	



- Focus on a particular region of the design
  - Manage “amount of coverage data”
  - New or lesser tested area
  - Specific concerns with coverage in an area
  - Access to designers
  - Can merge multiple regional coverage databases into a complete view

cadence®

System-level functional coverage example



Block-level verification focus

≠

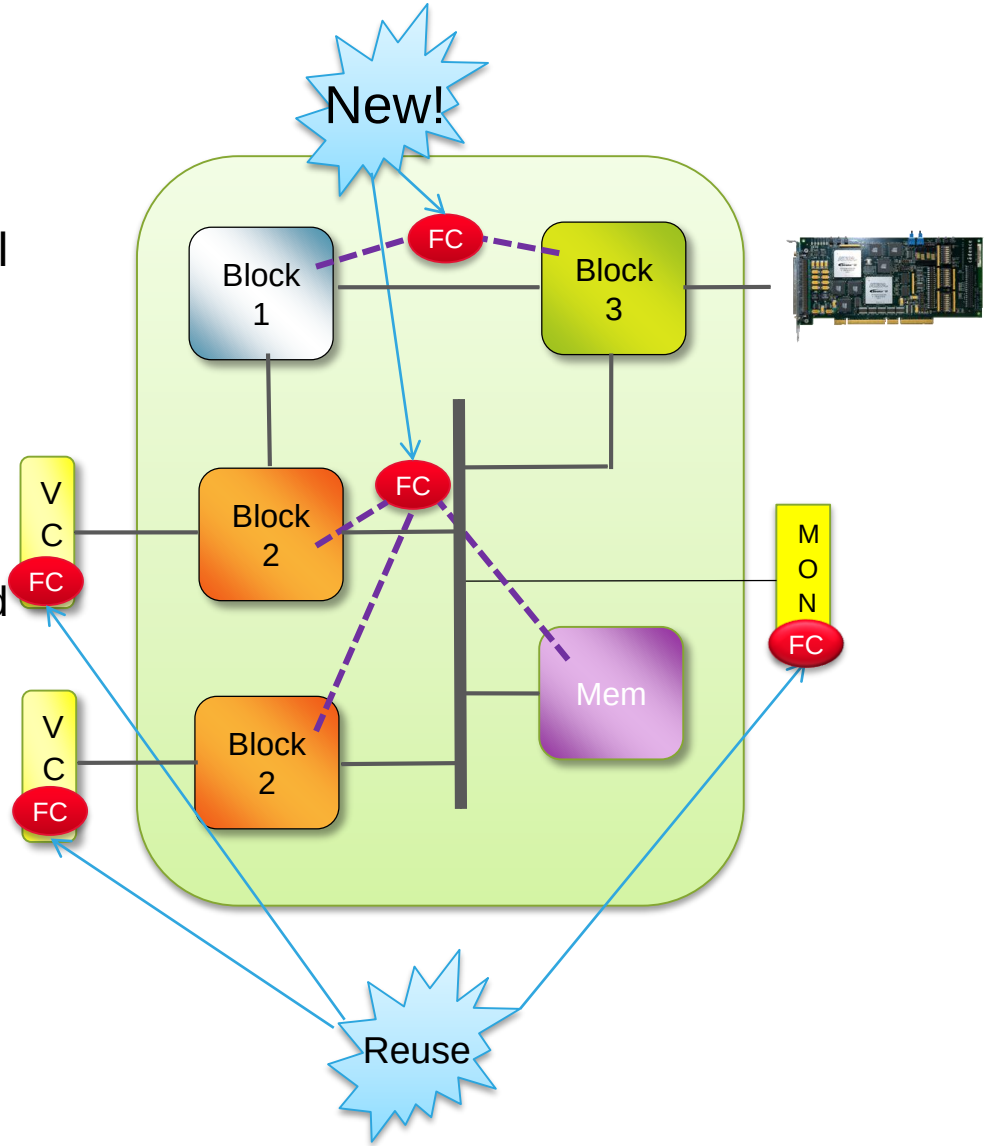
Sub-system, system-level verification focus

cadence®

Verify modes of operation

Code	Functional
	✓

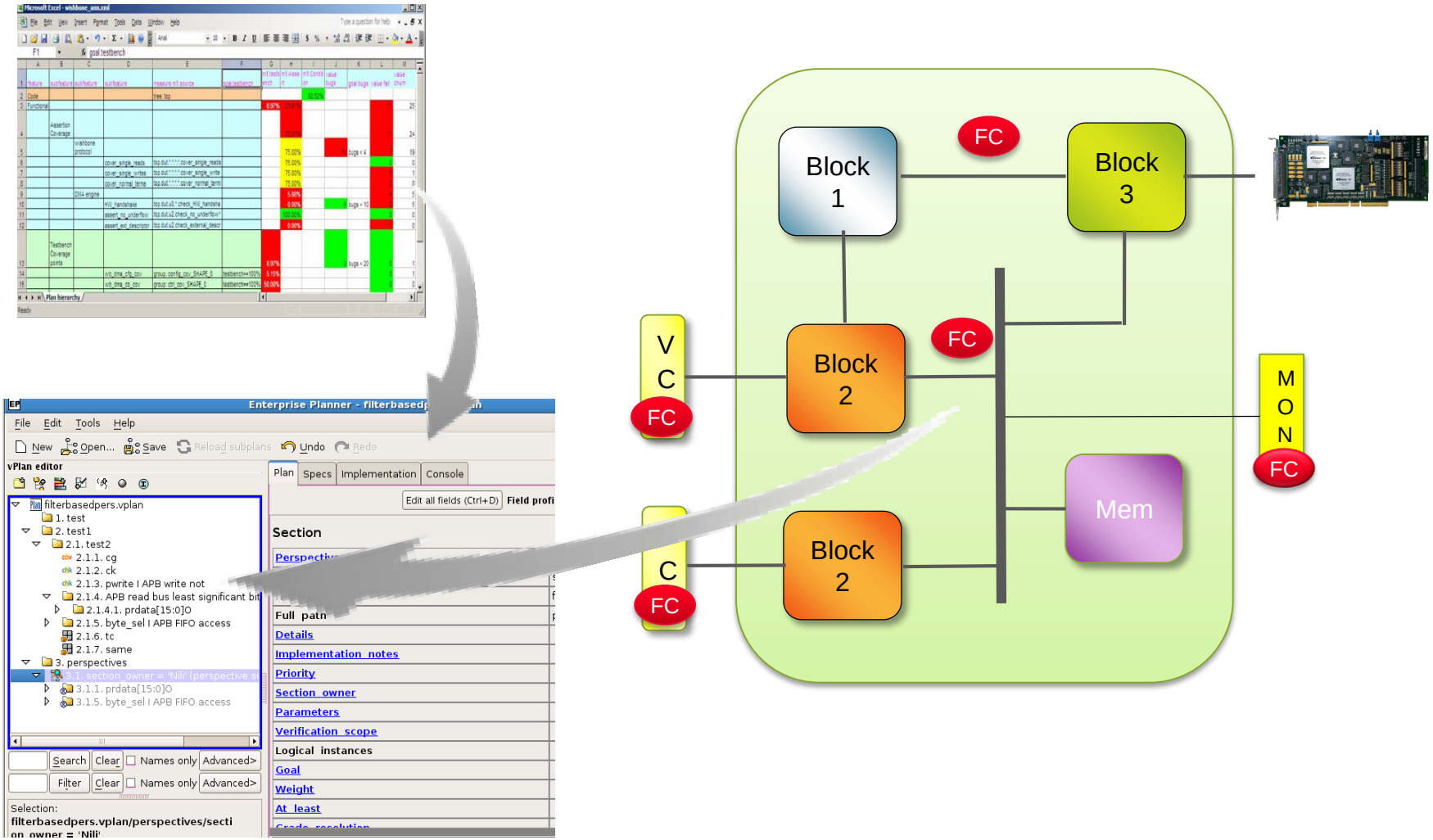
- Can maintain some monitors for coverage from the subsystem level
- Fundamentally, asking different questions at the system level
  - Concerned with interactions between subsystems
  - Implies a system level test plan tied to design spec
- Verifying modes of operation
  - Were these two processing units active simultaneously?
  - Were these interfaces active simultaneously?
  - Have I received an interrupt when the CPU is transferring data to the GPU?



cadence

Verify modes of operation

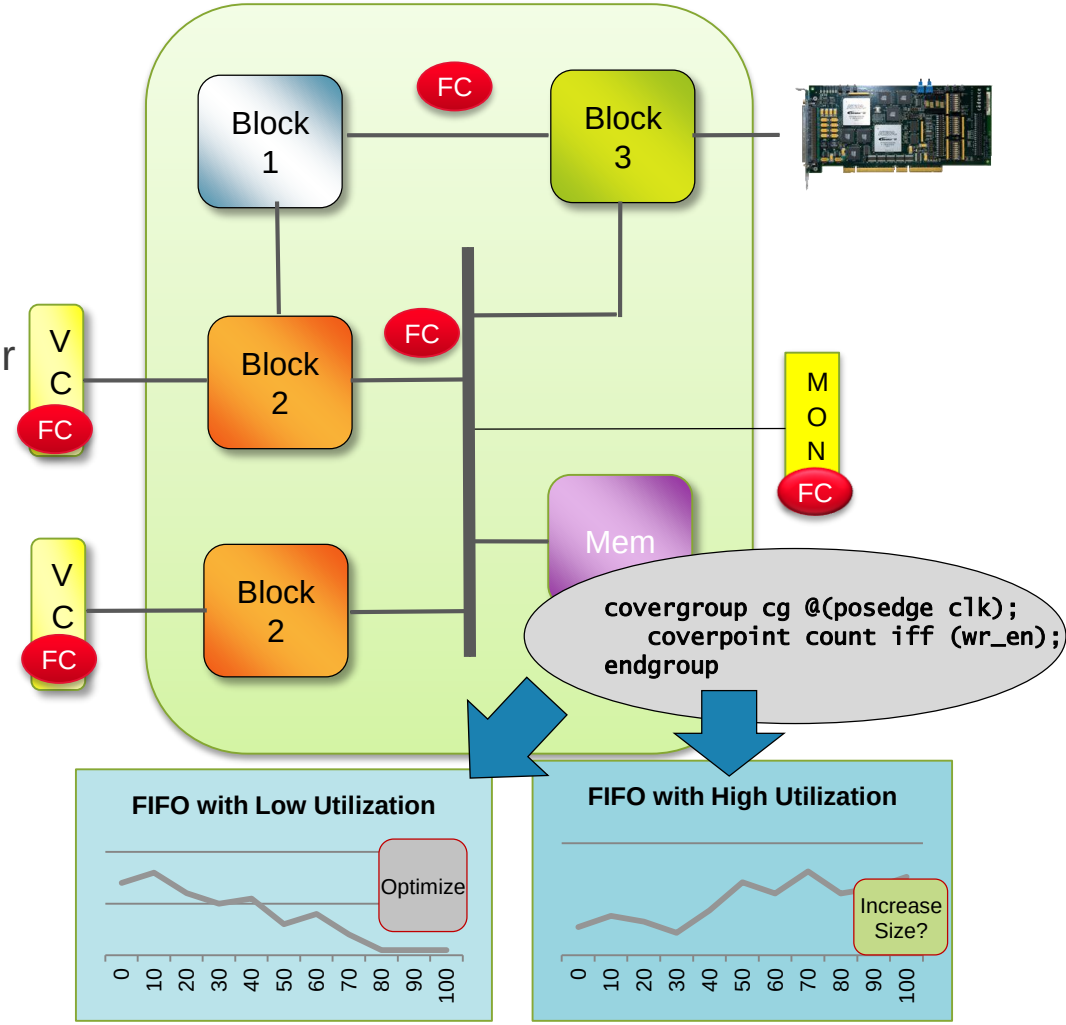
Code	Functional
	✓



Design optimization

Code	Functional
✓	✓

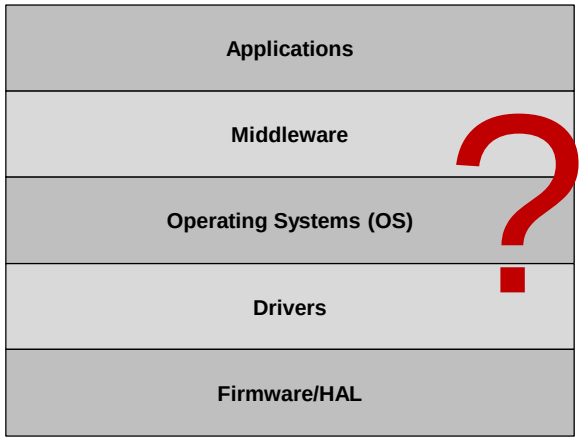
- Investigate performance in real-world scenarios
  - What is the average utilization of the FIFO?
  - If low, can we reduce the FIFO size?
  - If high, can we expand the FIFO or can we optimize the application software?
- You may have seen cases where designers put in special counters and instrumentation
  - Covergroups and cover properties are a very easy way to instrument, plus there are standard tools for merging, reporting and analyzing results

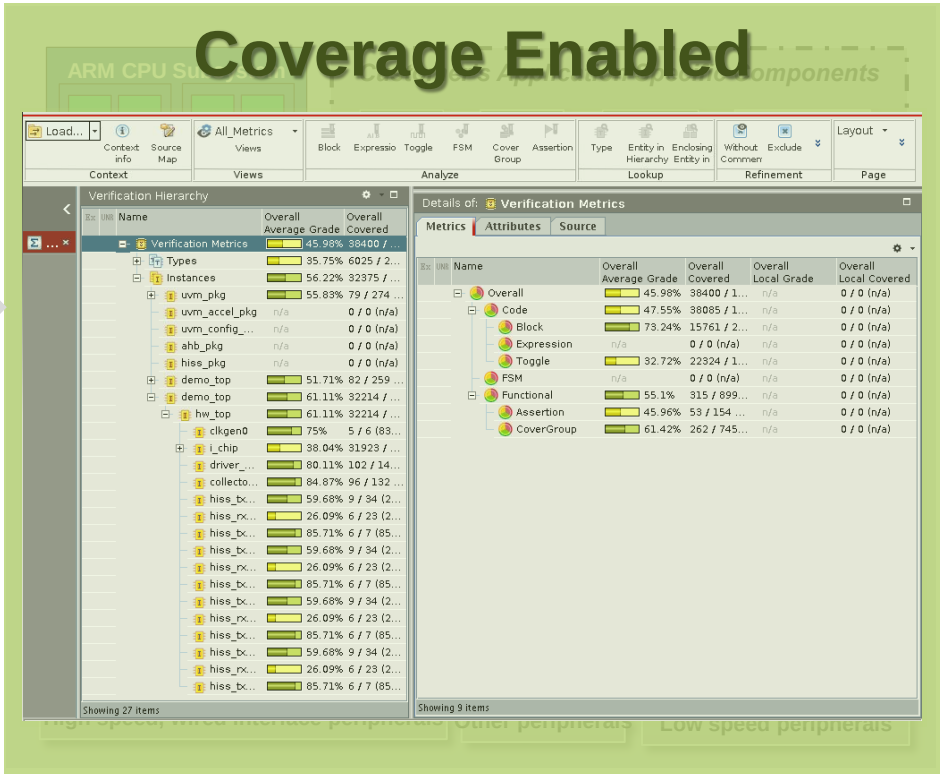


cadence

Improving hardware coverage of software tests

Code	Functional
✓	✓



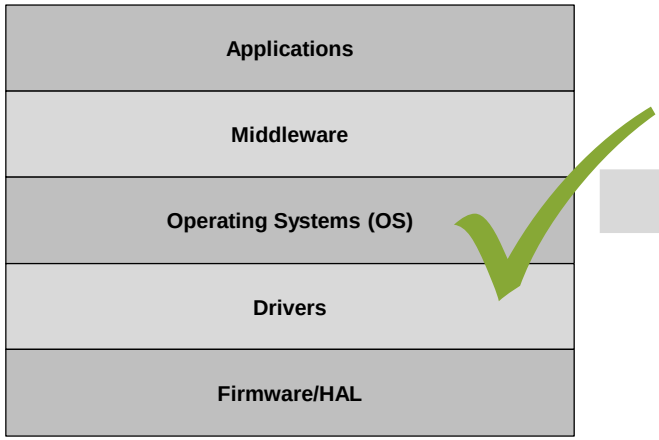


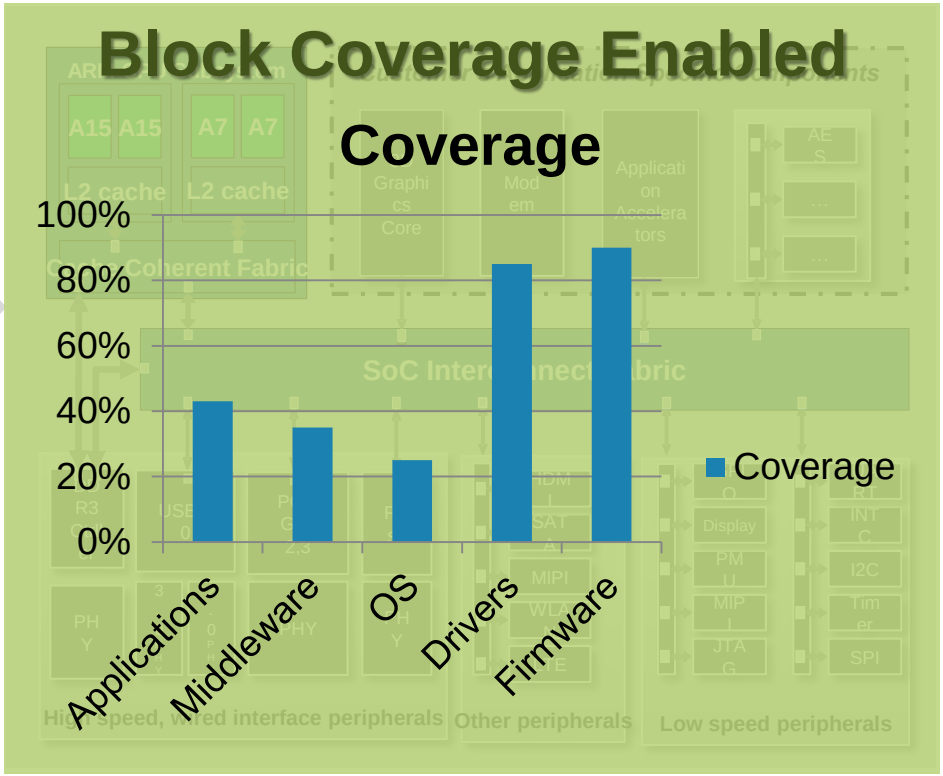
- Software-validation process often independent of hardware-verification process
- How well is the software exercising the hardware?
- Get a sense of “coverage” of the software through enabling hardware coverage during the running of software tests

cadence®

Improving hardware coverage of software tests

Code	Functional
✓	✓





- Software-validation process often independent of hardware-verification process
- How well is the software exercising the hardware?
- Get a sense of “coverage” of the software through enabling hardware coverage during the running of software tests

cadence®

DEMONSTRATION

MDV Tutorial Summary

SoC MDV – Multi User, Multi Engine, Multi Metric

Environment pulling together contributions from all users, engines, and metrics

Sessions:

Session Status	Name	Total Runs	#Passed	#Failed	Start Time	Owner
(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)	(no filter)
completed	Chip_ENET.14_02_26_11_35_38_...	3	3	0	2/28/14 ...	melancon
completed	ams_smoke	2	2	0	1/29/14 ...	magraham
completed	hw_sw_nightly	17	17	0	6/9/10 1...	magraham
completed	IO_SS_Connectivity	1	0	1	3/6/14 2:...	joseb
completed	SMC_Block_formal.14_02_27_13_...	11	10	1	2/27/14 ...	joseb
completed	SMC_Block_sim.14_02_27_13_37...	1	1	0	2/27/14 ...	joseb
completed	UART_Block_UNR	1	1	0	2/26/14 ...	joseb
completed	IO_Subsystem.14_02_28_16_07_...	12	12	0	2/28/14 ...	johnn
completed	UART_Block.14_02_26_11_57_41...	12	5	7	2/26/14 ...	johnn
completed	LowPower_Verification	1	1	0	1/24/14 ...	johnn

cadence®

SoC MDV – Multi Engine, Multi Metric Plan

Executable verification plan that can link to all necessary engines and metrics

vPlan Hierarchy

Name	Overall Average Grade	Assertion Status Grade	Assertion Failed
Switch Verification Plan	56.47% *	62.73%	1
1 SoC	58.39% *	n/a	0
1.1 HW_slash_SW Integration [VIRTUAL	58.39% *	n/a	0
1.2 Use Case [SW DRIVEN]	n/a	n/a	0
2 Subsystem	51.71%	60%	0
2.1 ENet [SIM-ACCEL]	69.62%	n/a	0
2.2 IO Subsystem	33.8%	60%	0
2.2.1 Interconnect Verification [SIMU	46.46%	n/a	0
2.2.2 IP Connectivity [FORMAL]	! 0%	n/a	0
2.2.3 Power Intent [SIMULATION]	54.95%	60%	0
3 IP	59.32%	63.75%	1
3.1 UART [SIMULATION]	73.98%	88.89%	0
3.2 SMC [FORMAL]	80.38%	43.18%	1
3.3 PLL [MIXED SIGNAL SIM]	53.27%	n/a	0
3.4 GPIO [SIMULATION]	58.25%	n/a	0
3.5 SPI [SIMULATION]	30.73%	n/a	0

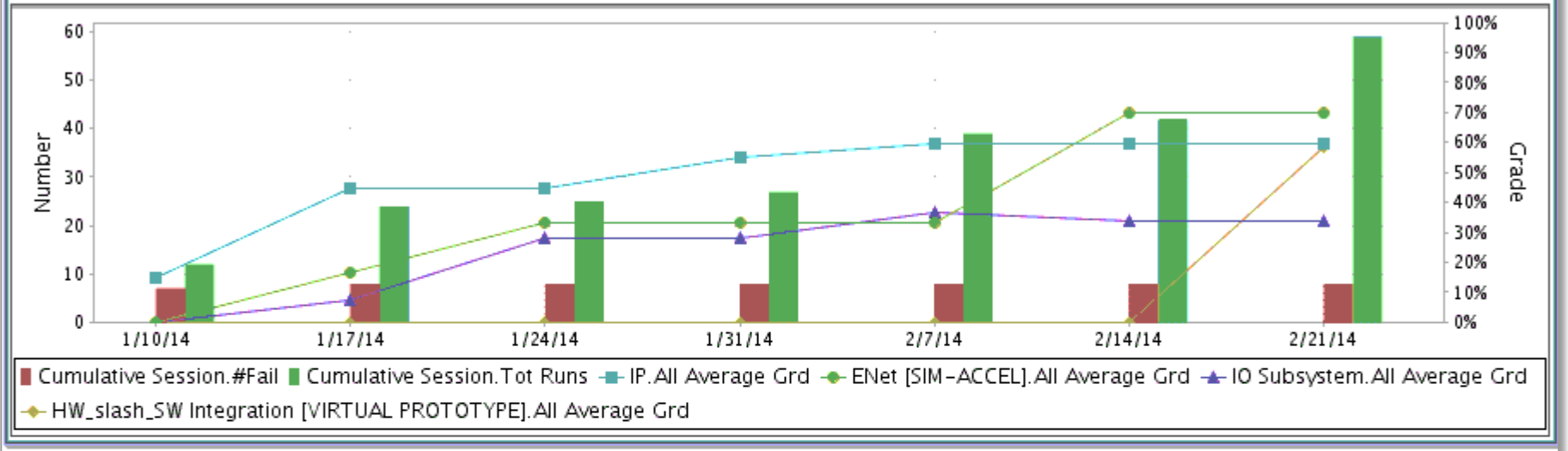
cadence

SoC MDV – Multi Engine, Multi Metric Tracking

Tracking progress of contributions from all users, engines, and metrics

Tracking

Snapshots | Switch SoC



Details | Source

Data Source of: **SoC_Snapshot_7**

Sessions | Metric Hierarchy | vPlan

Perspectives: Switch Verification Plan

Name	Overall Average Grade	Overall Covered
Switch Verification Plan	56.47% *	1146 / 2664 (43.02%)
SoC	58.39% *	90 / 383 (23.5%)
Subsystem	51.71%	273 / 890 (30.67%)
IP	59.32%	783 / 1391 (56.29%)

Showing 4 items

cadence

The How To's of Metric Driven Verification to Maximize Productivity

- MDV has been proved to improve predictability and productivity at IP to Sub-System Levels
- Today you have learned how MDV can be expanded using vManager to operate across specialized verification engines
- Additionally you have learned how MDV can be used thru to SOC level verification.
- MDV at SOC is new and emerging, and Cadence is committed to codify and optimize this for the industry, just like we did with UVM from eRM at IP levels
- Thank you for your participation today. You can learn more about the vManager Solution and MDV on the Cadence website – www.cadence.com

cadence®

cādence®

Questions ?