So you think you have good Stimulus
System-level distributed metrics analysis and results

Driving Forces
Large SoC Integrations
Complex Component Interactions
Current Coverage Metrics
Constrained Random Stimulus
Visibility across SoC required

First Results
New visibility into simulation activity
- Examine activity within a simulation that is orthogonal
to traditional coverage and correctness checks
- Found errors in constraints that were trivial once seen,
but passed all checks
- Immediate simulation efficiency improvement

Visibility across SoC required
Current Coverage Metrics
Complex Component Interactions
Large SoC Integrations
Visualization helps understanding
Plot results
Find system-level holes in operations
Examine interactions between groups of transactions
Find patterns of operations
Analyze interactions
Understand system-level interaction
Group transactions for analysis
Track operations moving between IPs
Correlate independent transactions
- GPU / PCIe accesses
- Memory accesses
- Fabric operations
- Cache operations
Capture transaction-level information across system
- Identify and report interactions across data
- Correlations may exist beyond a single simulation
- Between simulations in a regression
- Across engines or abstractions
- Find and report on relationships that may be independent of
test pass/fail criteria
- Fairness
- Arbitration
- Priorities

Statistical Coverage
Finding relationships between disconnected sets of data
- Could be separated in time
- Could be from different parts of the design
- Identify and report interactions across data
- Correlations may exist beyond a single simulation
- Between simulations in a regression
- Across engines or abstractions
- Find and report on relationships that may be independent of
test pass/fail criteria
- Fairness
- Arbitration
- Priorities

Cache Operation Distribution
Provide a measurement of processor stimulus
- What is stimulus actually doing in the system
- How is the system reacting to stimulus
- Does the distribution match expectations
- Can distribution be changed
- Functional coverage looks fine, tests all pass
- Once distribution is seen, it is possible to change constraints

Initial stimulus distribution was not what was desired
- Functional coverage looks fine, tests all pass
- Once distribution is seen, it is possible to change constraints

Project Overview
Using a current dual-cluster ARM SoC project
Modern System/SoC verification flow
High-quality constrained-random stimulus
Achieved code and functional coverage closure
Statistical coverage installed in existing project
Measure bus, cache operations
Traffic and interactions between IP blocks
Running existing stimulus suite
Obtain results on entire regression suite
- Statistical - addresses, time, tests abstracted away
- Cumulative - can see if events every happened
Obtain results on entire regression suite
Results are based on an entire regression suite

Representative System
CoreLink™ CCN-504 and DMC-620

Method
- Correlate interdependent transactions
- Track operations moving between IPs
- Group transactions for analysis
- Understand system-level interaction
- Analyze interactions
- Find patterns of operations
- Examine interactions between groups of transactions
- Find system-level holes in operations
- Plot results
- Visualization helps understanding
- Outliers and behaviors are identified

Distributions
In a complex SoC, the distribution of operations in one component
is significantly driven by surrounding components.
Achieving a desired distribution may require modification of
constraints and configuration of several other blocks within
the design.

Examples below required a number of constraint and simulation
configuration changes to achieve better distributions

Operations per time slice before and after
Read latency distribution before and after
Mean and max queue fill

Abstraction & Statistical Coverage
Examine patterns across large data sets
Trends in time
- Complex correlations across system
- Abstraction allows easier grasp of activity
- Many aspects of operations are dropped
Abstract cache sharing plot
Abstraction can show high-level trends
Abstraction can also hide critical issues
- Patterns can look good at high level
- but hide low-level issues
Correct abstraction level is critical

More detailed cache sharing plot