Verification of Interconnects

Yoav Lurie VIP RnD Architect

November 2013

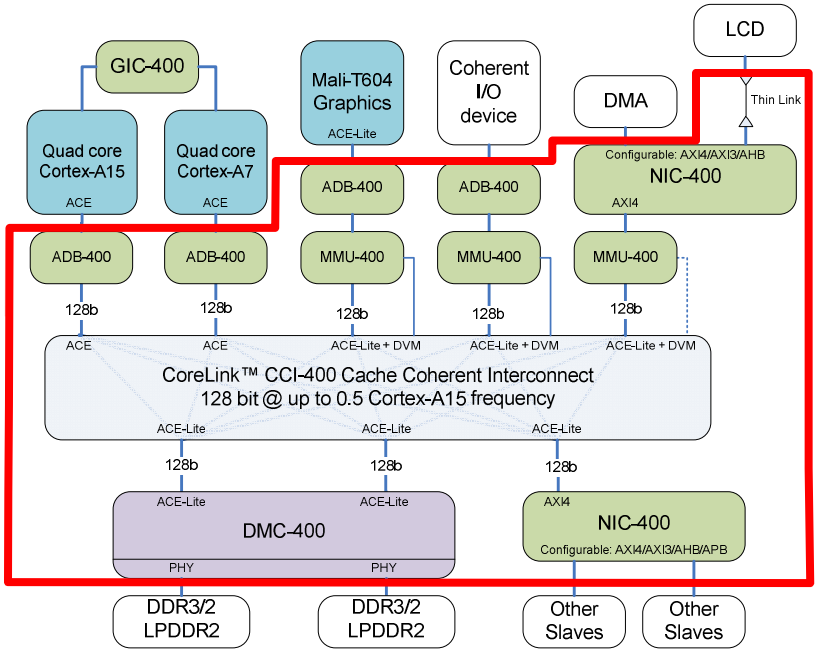
cadence®

SoC evolution

- ~1995 – today
- 1st gen – one CPU, several connected blocks, proprietary connections
- 2nd gen – shared bus architecture (AHB)
- 3rd generation – Interconnect architecture (AXI)
- 4th generation – Coherent interconnect (ACE)
- 5th generation – Coherent interconnect with internal memory (CHI)

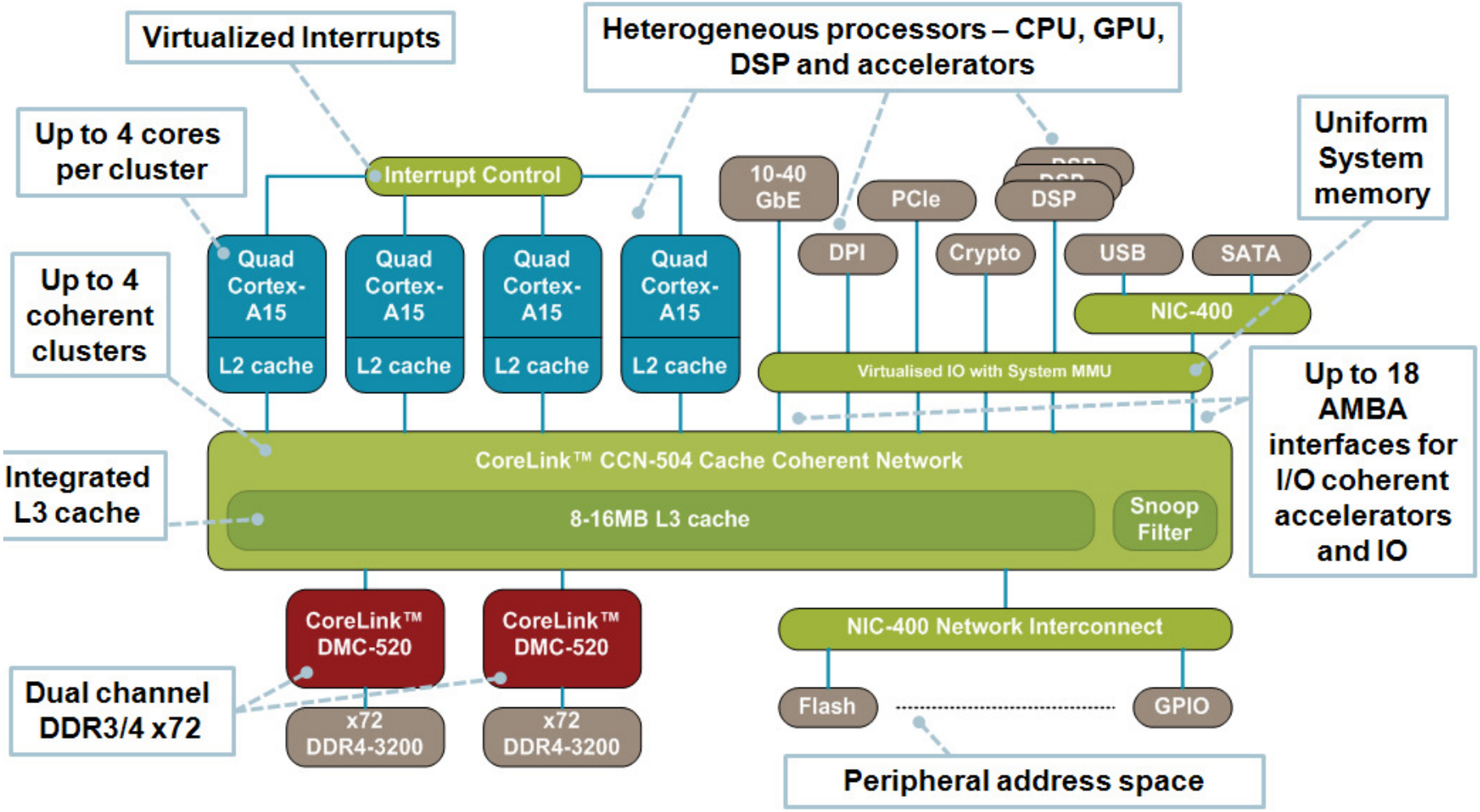
What is an InterConnect anyway?

- Main routing block. Directs traffic to and from various design blocks inside the system.
- Address-Data based.
- Can control Coherency as well as Data flows
- Can contain internal memory



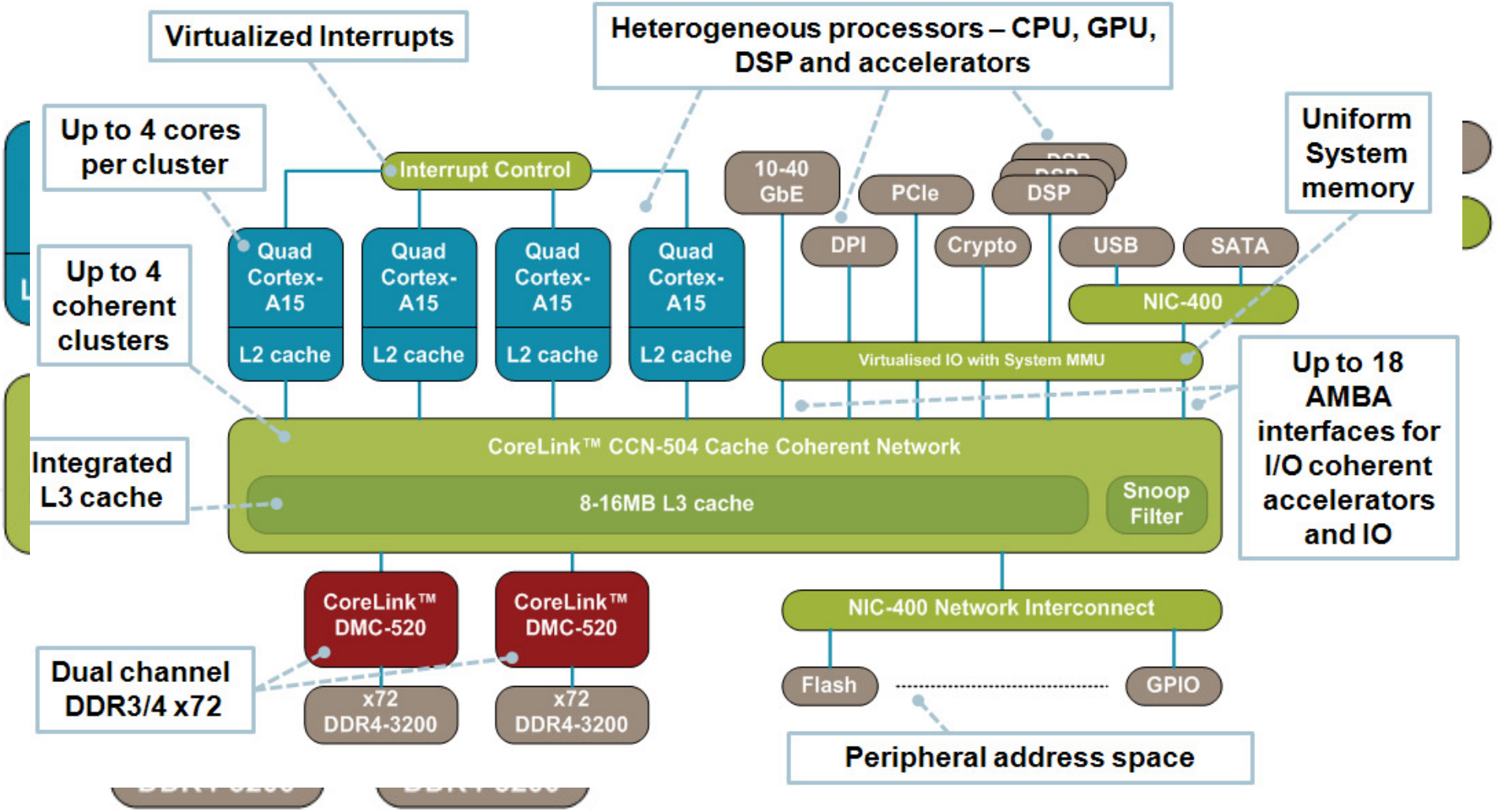
cadence

An example of today's SOC design
ARM's CoreLink™ CCN-504 and DMC-520

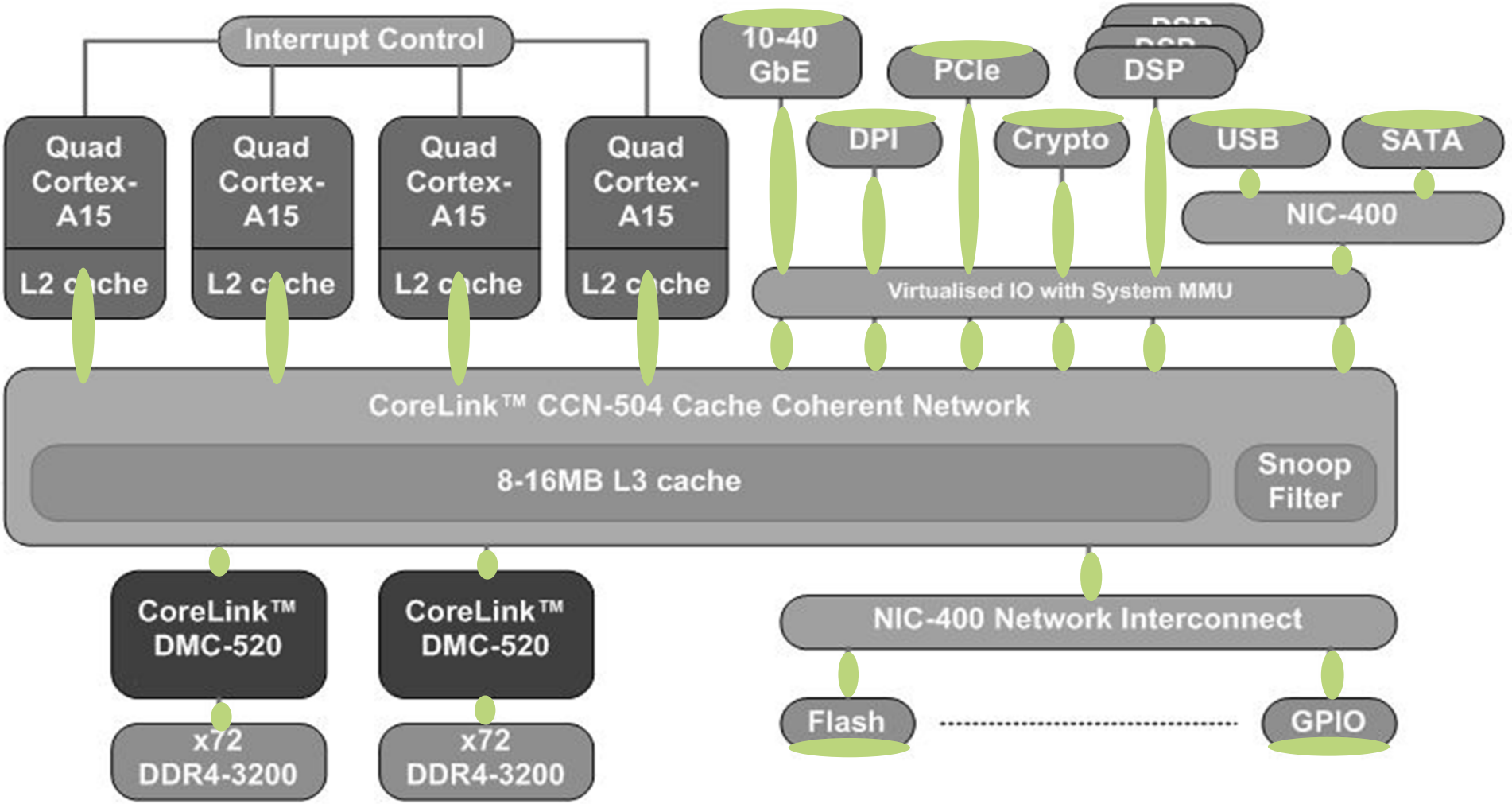


cadence

Current VIP checking scope

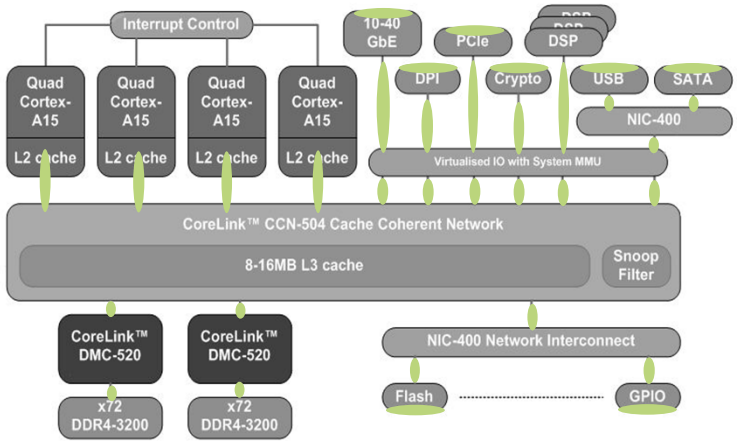


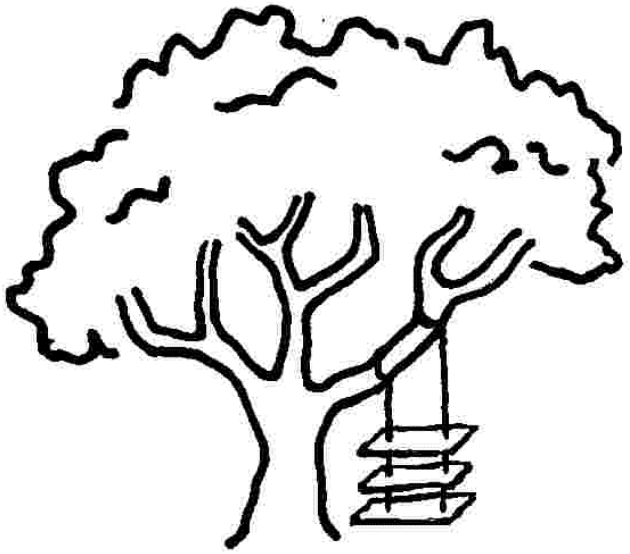
Current VIP checking scope



What the user wanted – system verification challenges

- Understand system scenarios
- Check System behavior
- Cover functionality
- Validate Performance

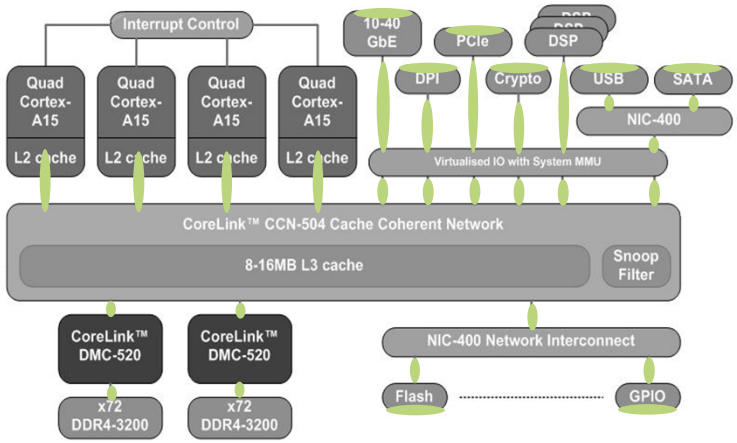




cadence

Possible solutions

- Ad-hoc VE
  - Time consuming
  - Very limited functionality
  - Not scalable
- Generic solution



cadence

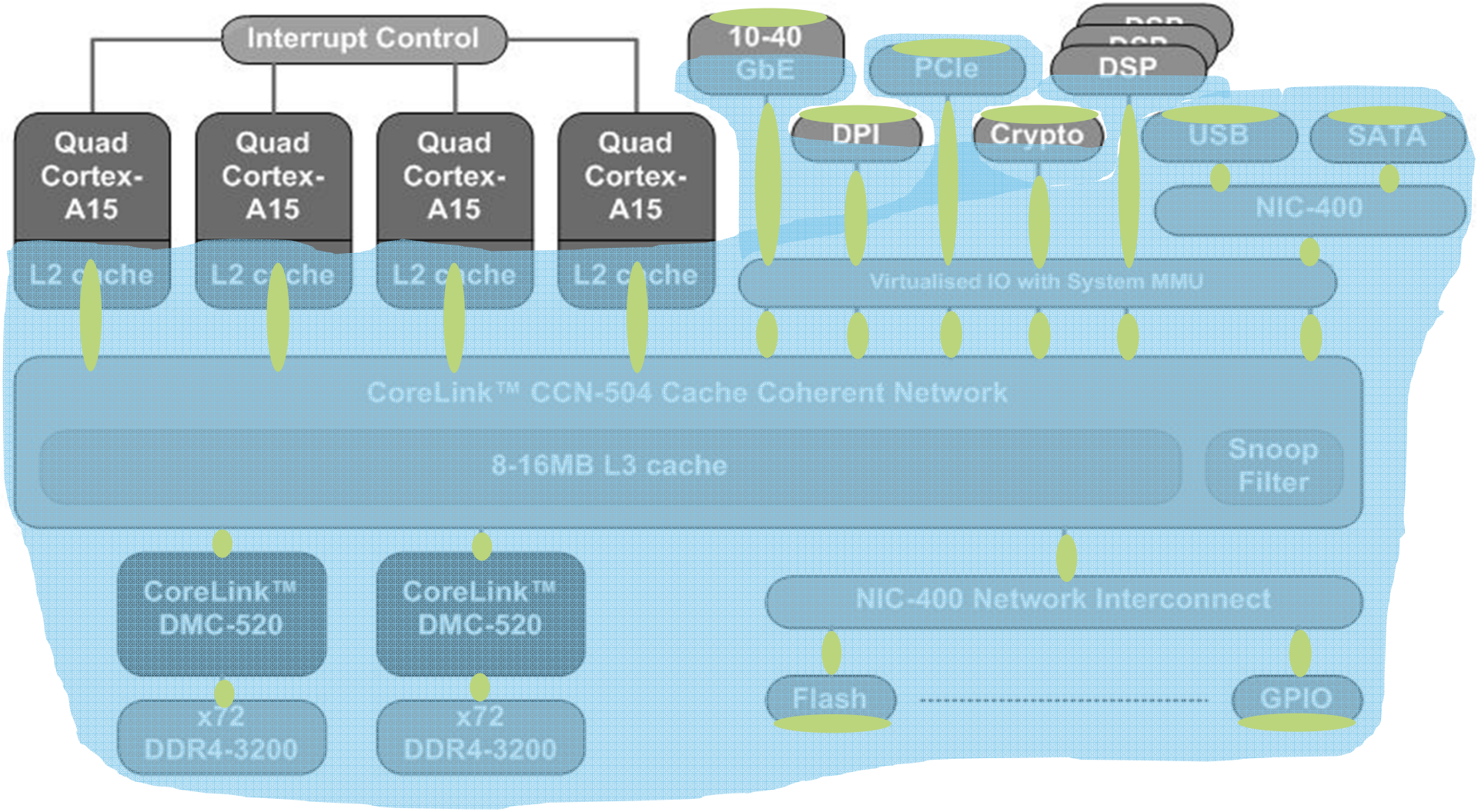
Generic solution requirements:

- Black-Box verification methodology
- Maximum flexibility
- Reuse between projects
- Reuse during projects stages of integration
- Reuse (Yes, it is that important) for different verification tasks:
  - Validation
  - Coverage
  - Profiling and Performance

Generic interconnect assumptions:

- Invariant definition
  - Most of the components in ANY SoC uses the pair of Address - Data
- Tracking Address – Data invariants across the system is feasible
  - This enables system verification
- Feasible ≠ Easy

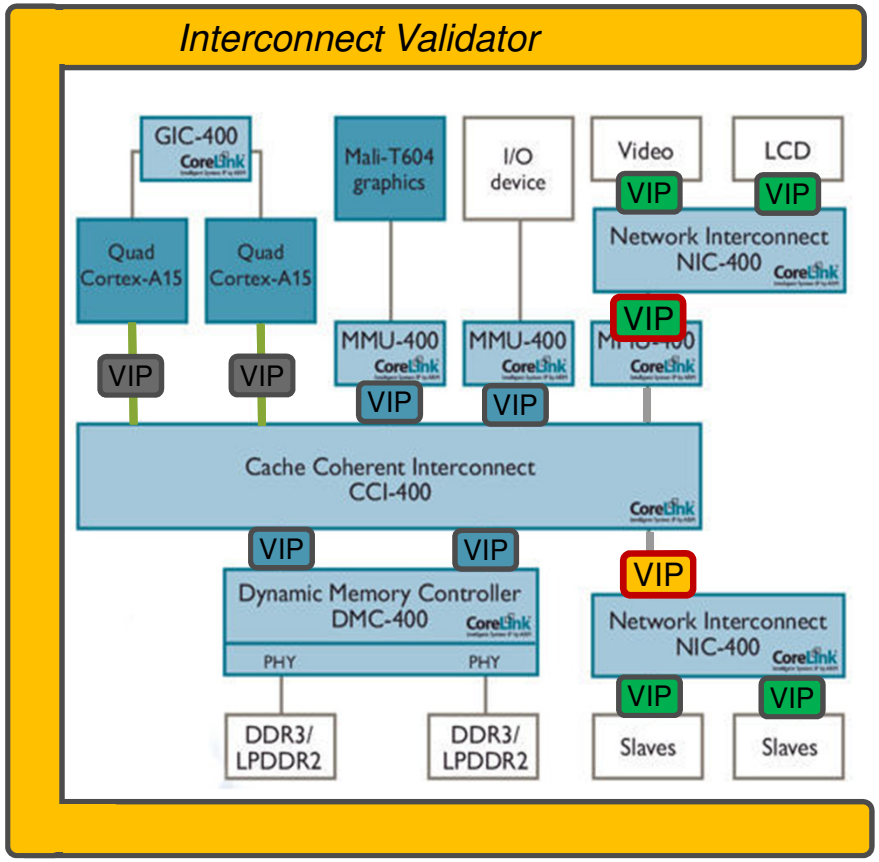
System invariant scope



cadence®

Fabric Verification Requires VIP AND *Interconnect Validator*

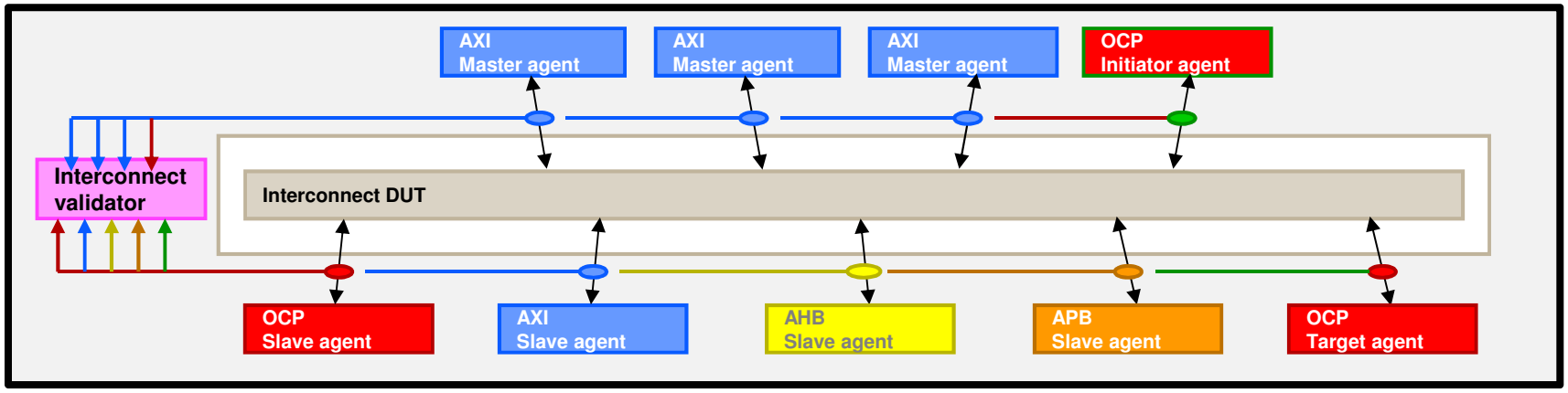
- VIP verifies each individual port (master or slave) complies with the protocol specification
- *Interconnect Validator* verifies interconnect behavior.



cadence

Interconnect Validator - A System VIP

- *Interconnect Validator* is a meta-monitor that tracks and understands system level scenarios
- *Interconnect Validator* has a built in mechanism to check data integrity
- *Interconnect Validator* operates at a higher abstraction level, and supports multiple ports with multiple protocols.
- *Interconnect Validator enable performance analysis*



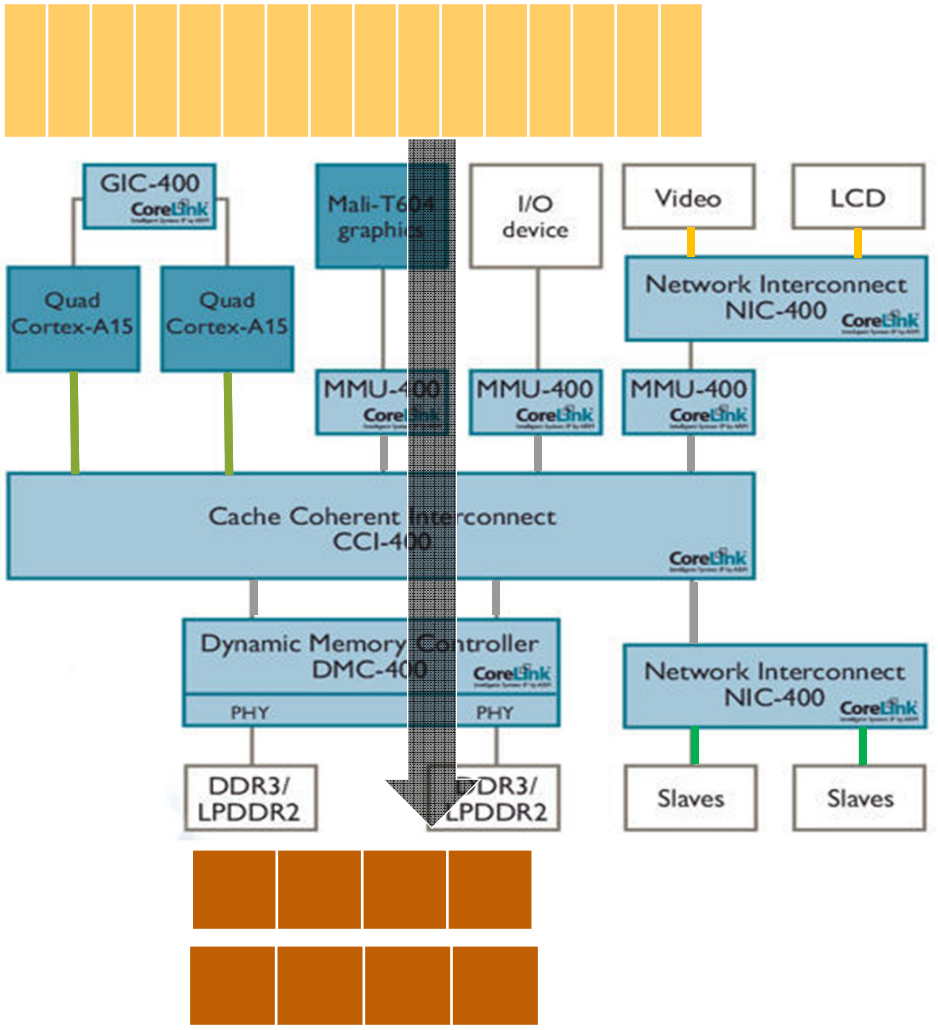
cadence

Checking System Behavior – Data transformation support

Example:

- Master send a burst with 16 transfers of 8 byte.
- The slave gets 2 burst, each with 4 transfers of 16 bytes.

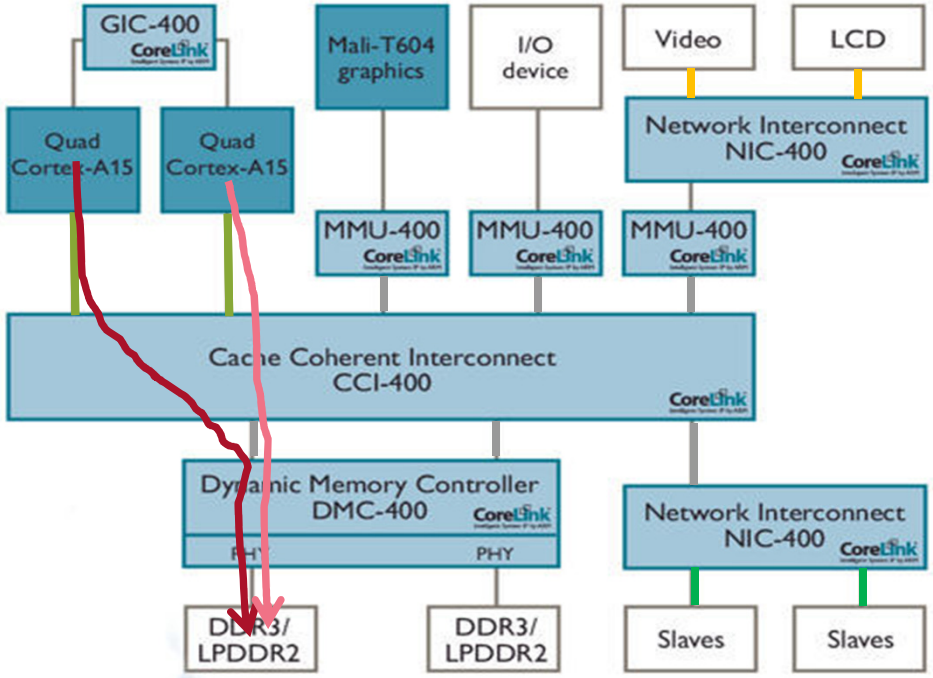
=> Split in the burst level while merging in the transfer level.



Checking System Behavior – Ambiguity

Example:

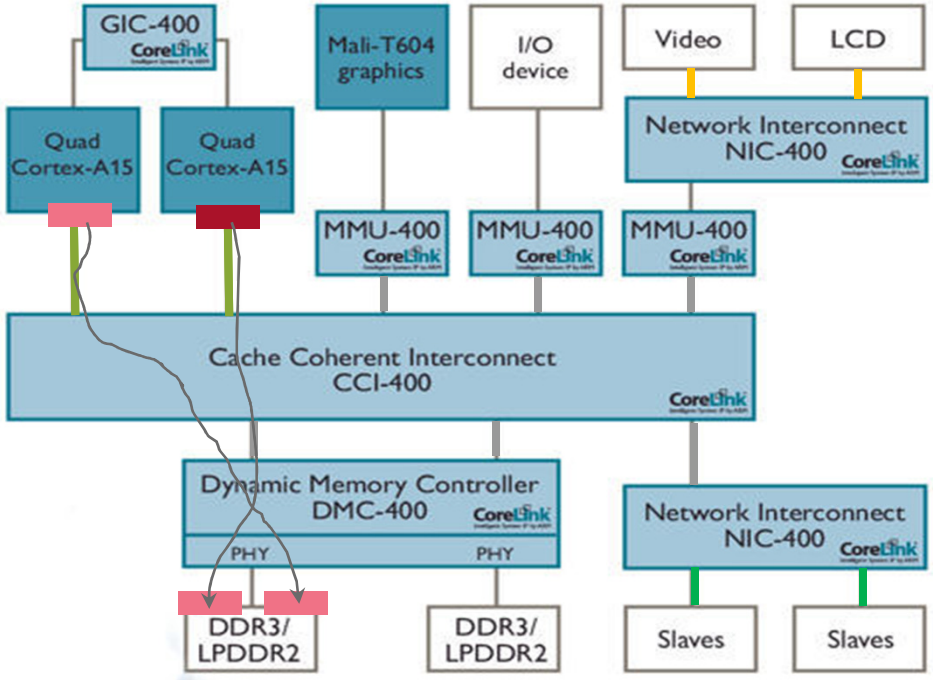
- Two masters Access the same address simultaneously.
- This is not the case from verification point of view



Checking System Behavior – Ambiguity from verification point of view

Example:

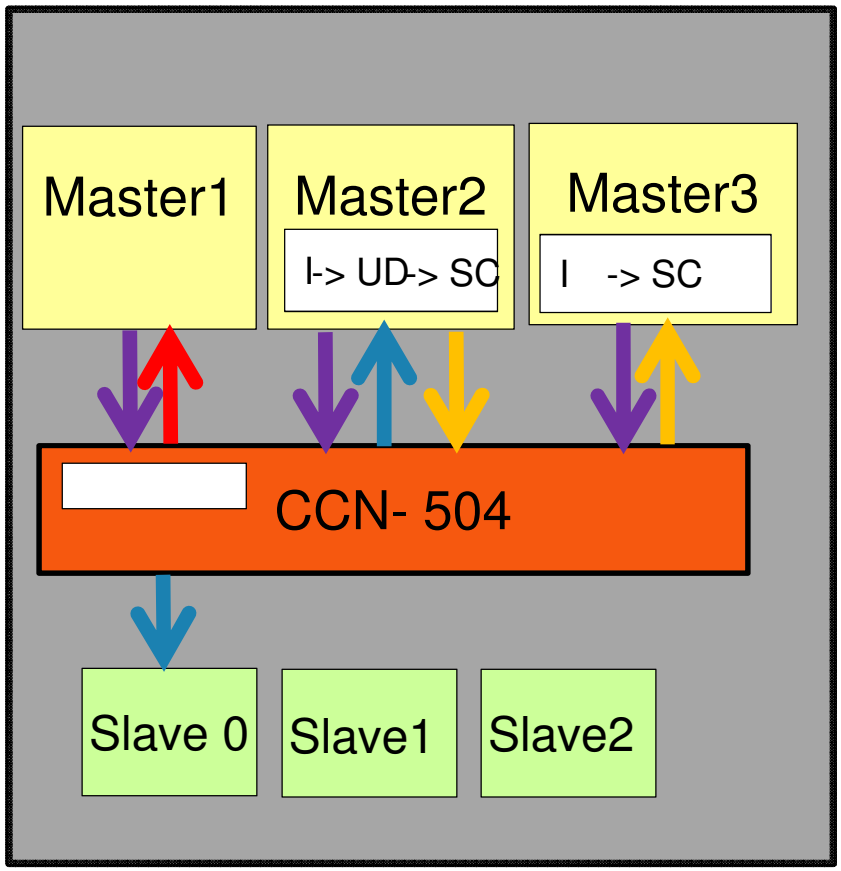
- Two masters Access the same address simultaneously.
- You have to draw the lines



Is data integrity all we can do???

- Well, at least we can do it…
- Seriously now…
- Data integrity checks enables “connecting the dots” => enables system level visualization
- System level visualization enables many checks in hindsight which are very hard to predict ***A priori***:
  - Arbitration
  - Ordering
- It is always easier to get forgiveness than permission

Checking System Behavior - Internal memory



Scenario description:

- M2 – Perform a ReadUnique
  - Get the data from main memory
  - Move to a unique state
- M3- Perform a ReadShared
- to the same address
  - Data is return from M2
  - Both M3 and M2 move to SC
  - L3 updated with the data as well
- M1 – Perform ReadOnce
  - M1 get the date from the interconnect and not from the other masters

Interconnect Validation Challenges – much more than a SCBD

Non Coherent Protocol

- Support any number of masters/slaves
- Mix and match protocols
- Splitting, Upsizing, Downsizing
- Addressing modes (incr, wrap, fixed)
- Supports independent address forwarding per master
- Response calculation
- Supports internal address ranges, unmapped access handling
- Handling error during memory access
- Transaction ordering
- Match before add
- Dynamic remapping

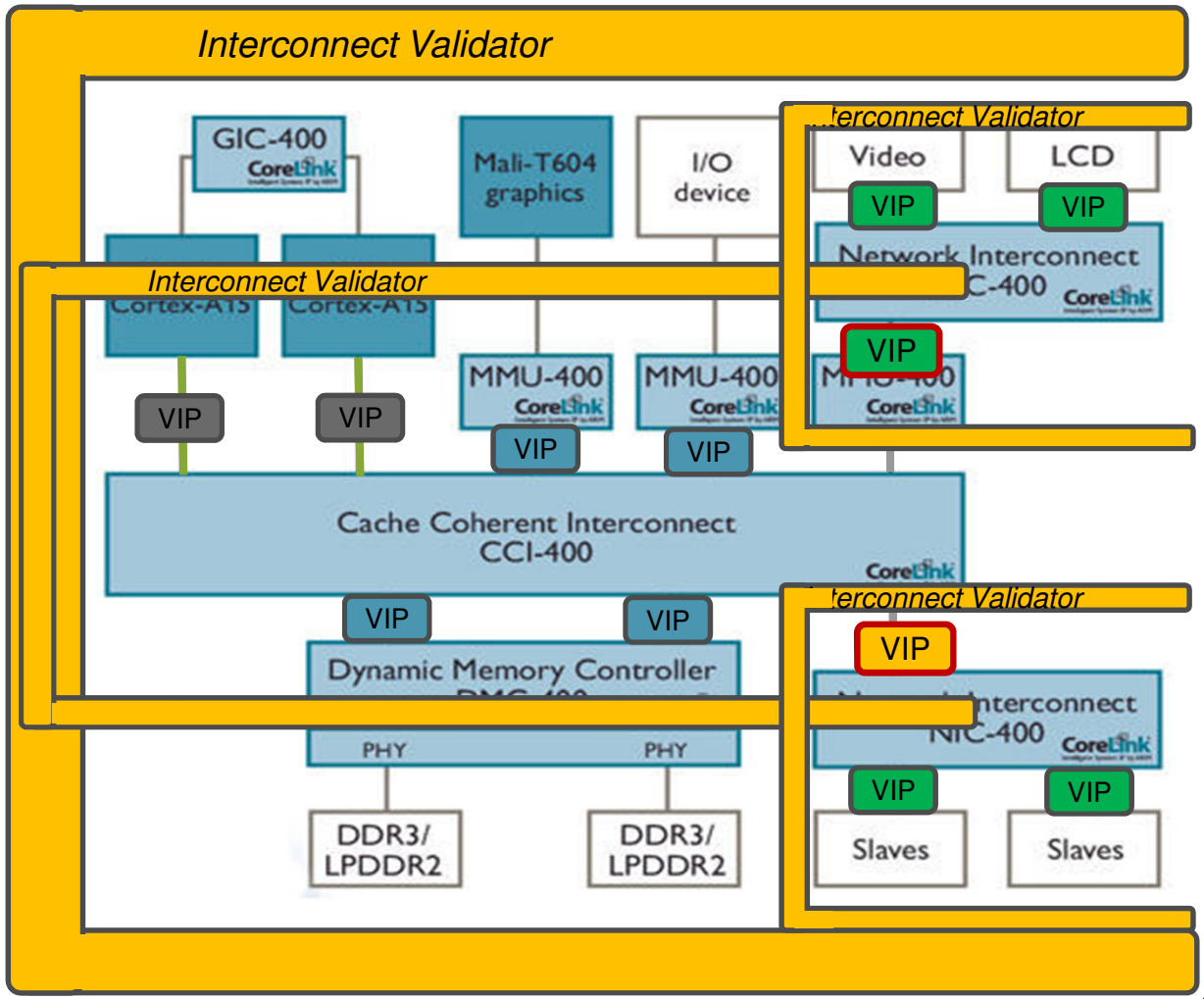
Support L3 cache inside interconnect

- Match reads with cache data
- Update cache with transaction data
- Support Gray states (user define)

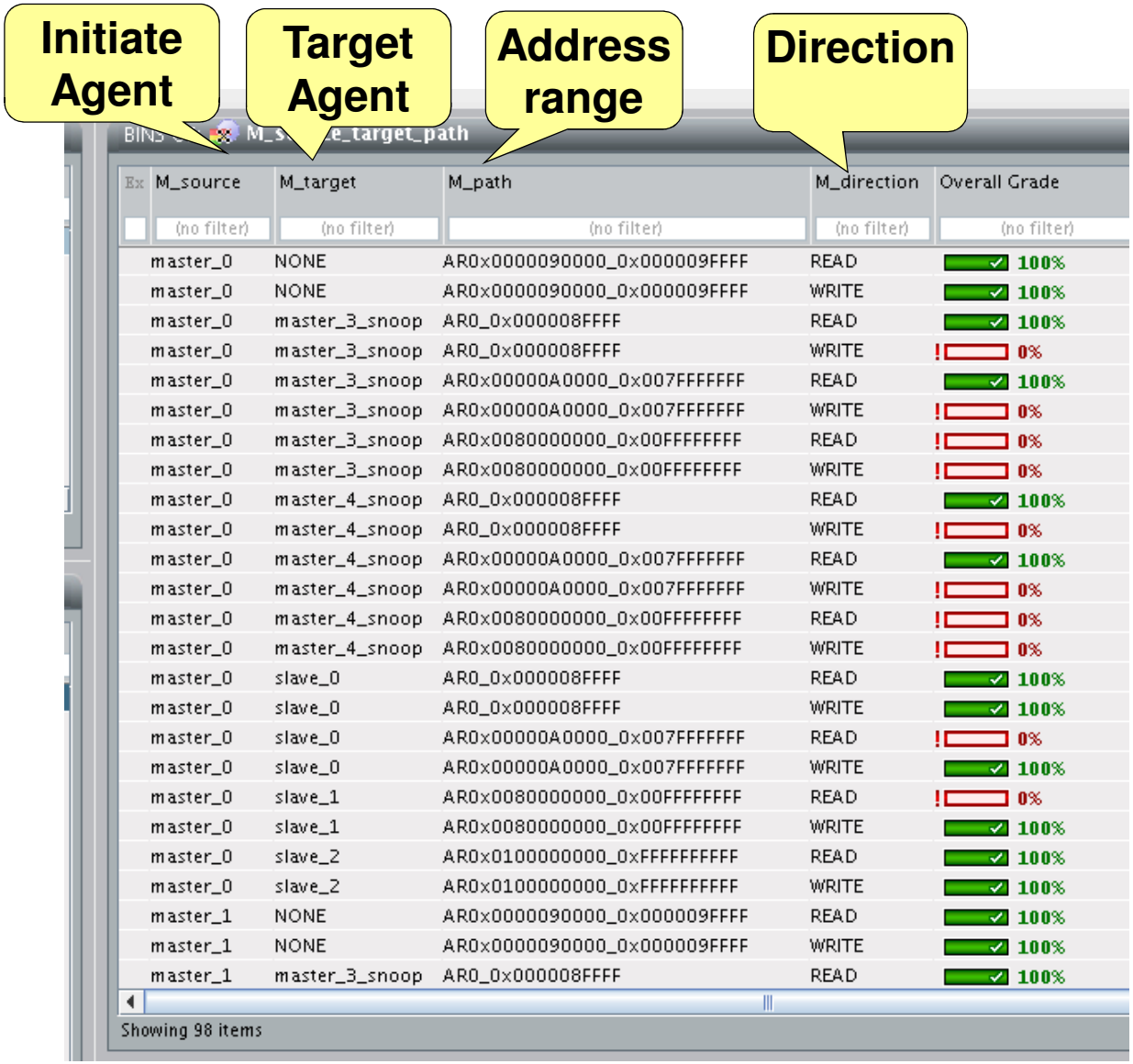
Additions for Coherency:

- Verifies all snoop conversions
- Verifies all snoop propagation possibilities
- verifies cross cache line operations
- DVM transactions
- Verifies barrier transactions
- Supports and verifies **snoop filter** behavior
- Interconnect initiated operations
- Support any number of outer and inner domains

Reuse in the product's life cycle



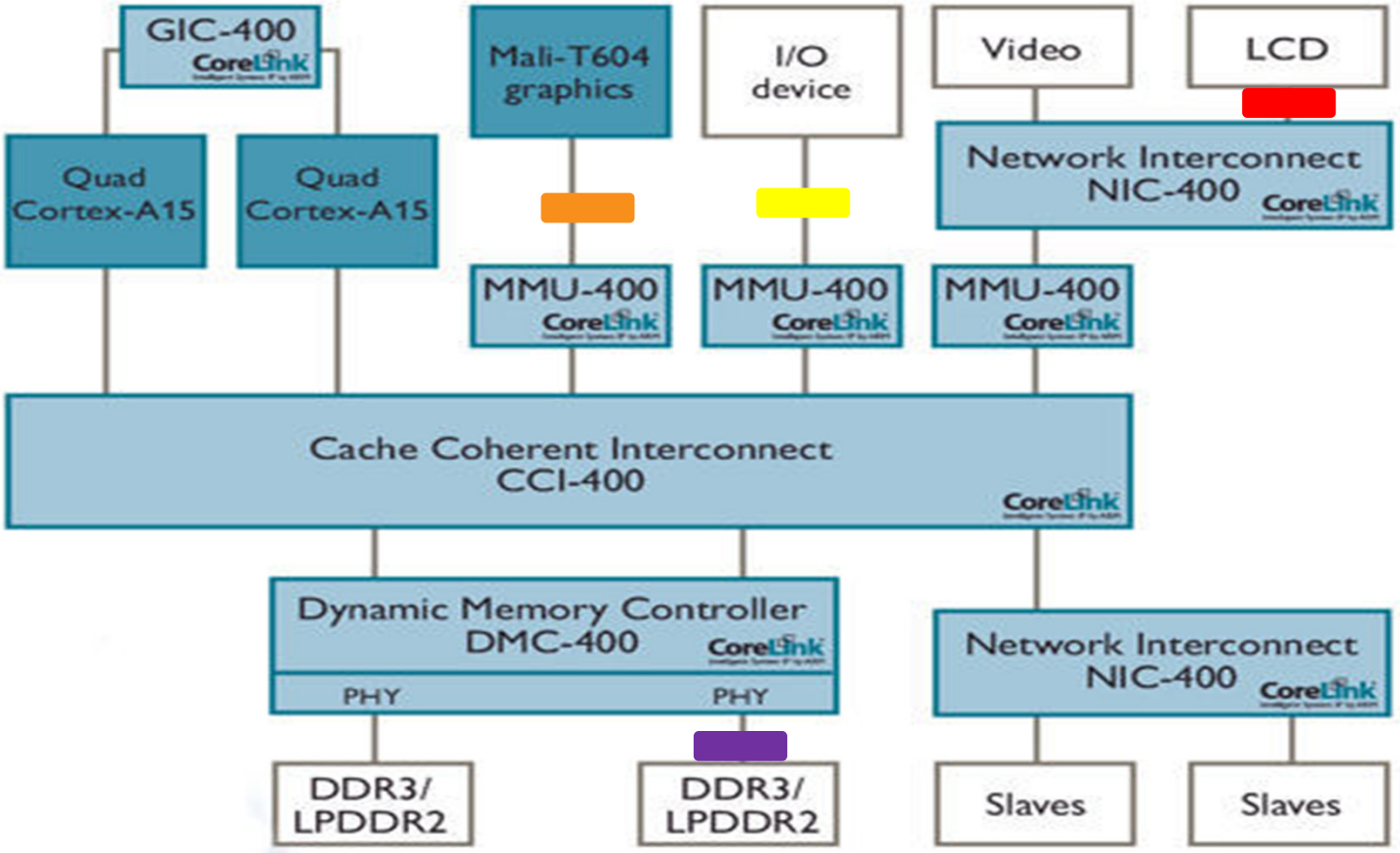
Providing System Coverage



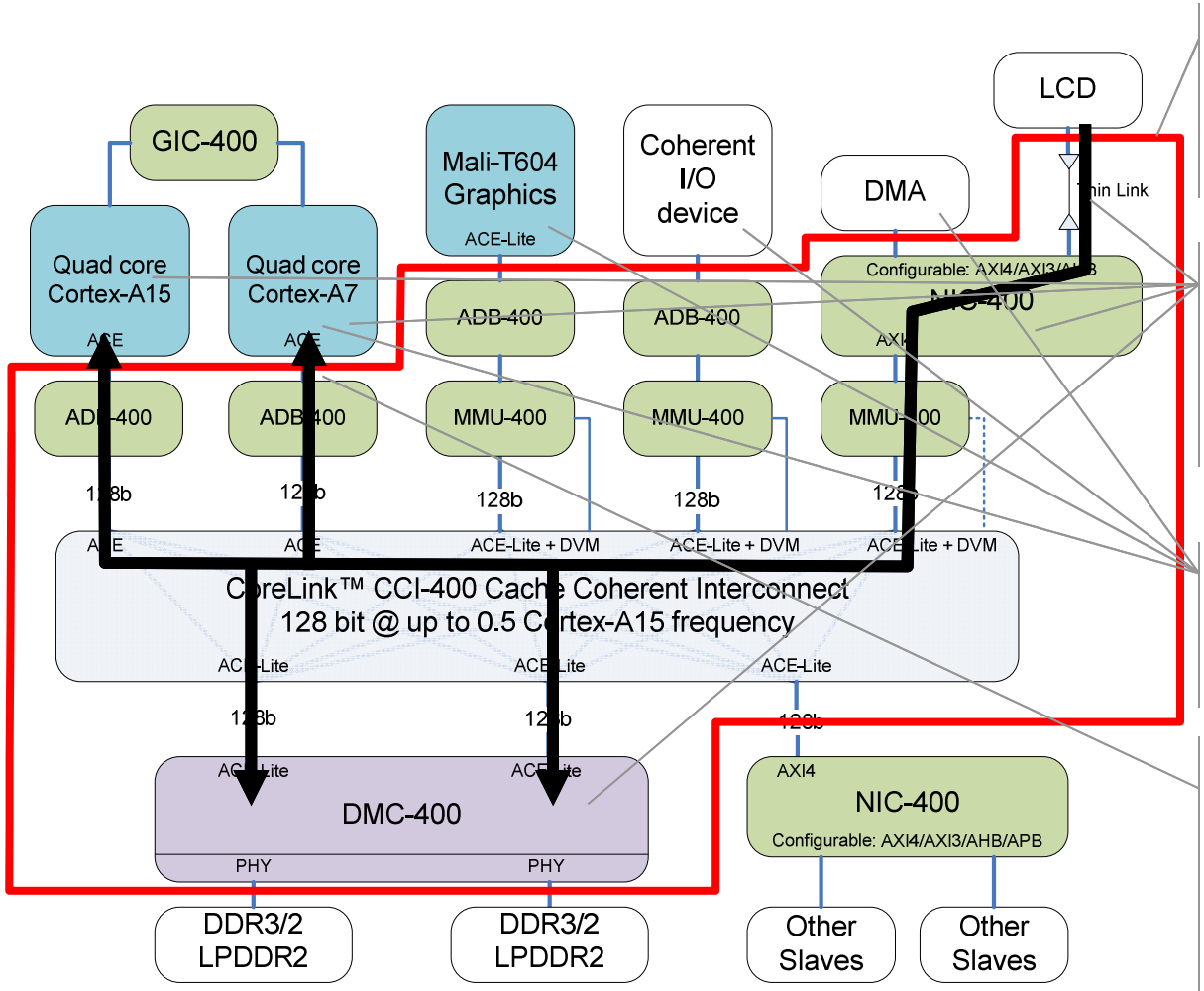
Coverage is tailor - generated for each interconnect specification

- Cross of source, destination, address range and direction ,transaction type

Understanding System Scenarios
with Debug Aids



Validating and Analyzing System Performance



Focus for performance of a path requires us to consider other masters that may influence the delay

Hardware influences : Thin links, NIC-400 configuration, QoS, L2 Cache Speed, DDR Controller speed.

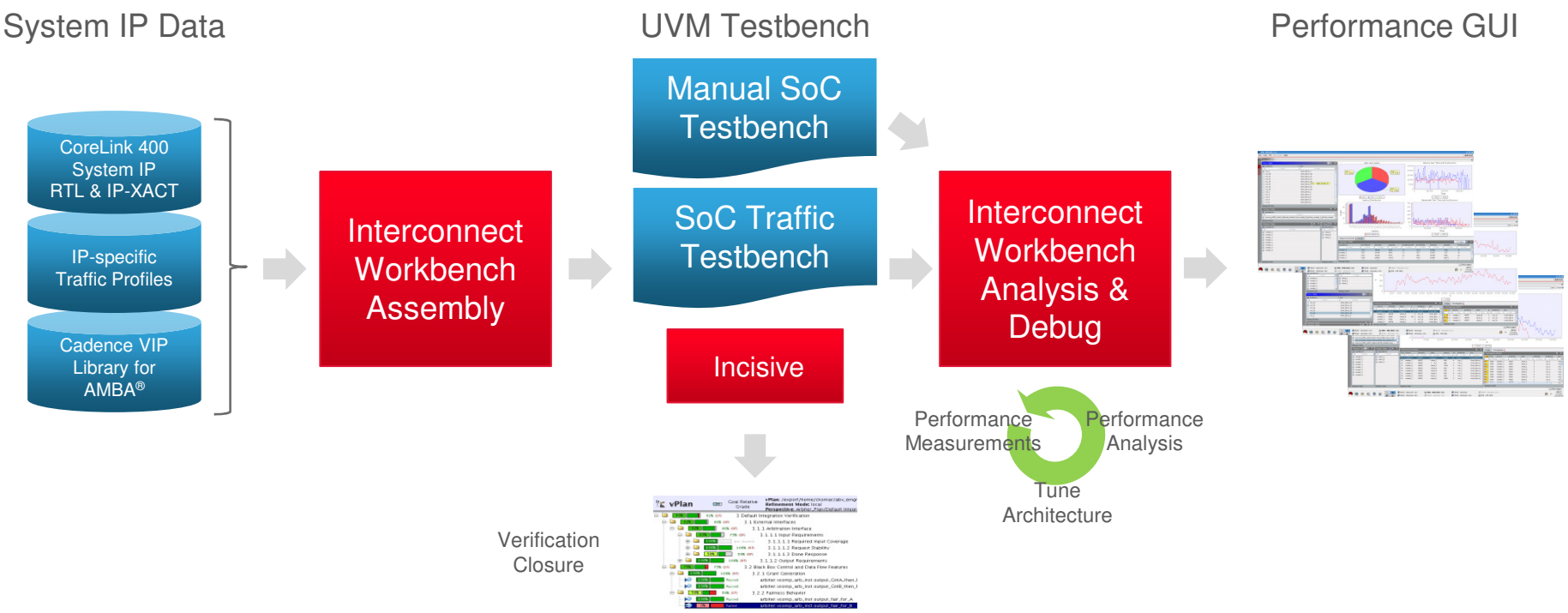
Scenario influences : Local traffic conflict, ACE-Lite Traffic, Processor Activity

Modeling all these HW artifacts in TLM is impractical. Accurate performance analysis must therefore use cycle-accurate RTL models

cadence

Cadence® Interconnect Workbench

Pre-integration Cycle-accurate Performance Analysis and Verification

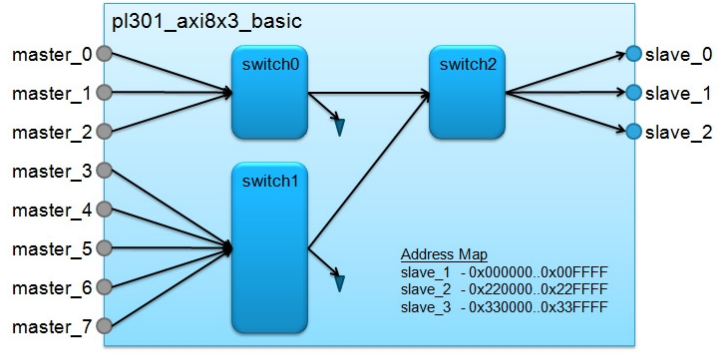


Benefits

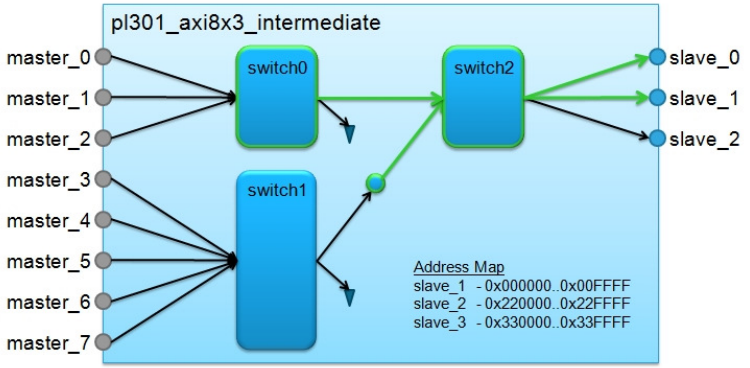
- Shorten performance tuning and analysis iteration loop from **days** to **hours**
- Reduce testbench development time from **weeks** to **hours**

Performance Analysis - example

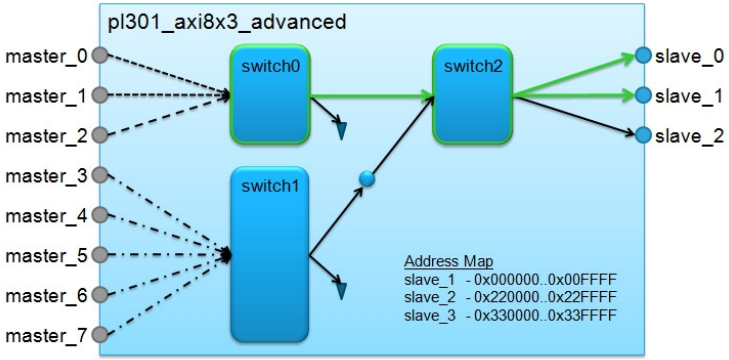
- For simplicity we have chosen and AXI 8x3 matrix
  - With 4 implementations, Basic, Intermediate, Advanced and Elite



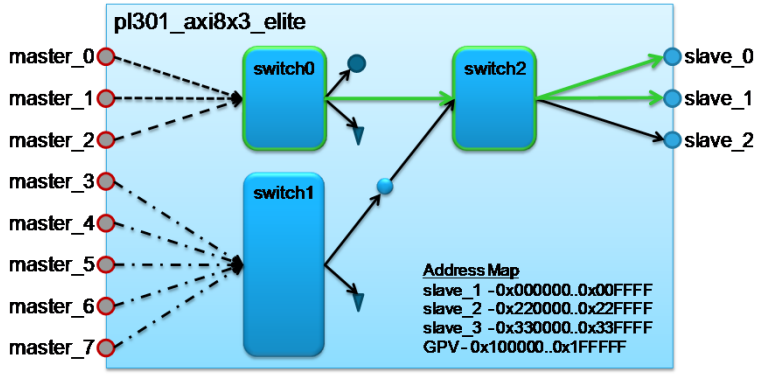
- 32bit databuses
- Read / Write issuing / acceptance capability = 1
- Default slave



- 128bit 'high-speed' datapath



- Read Issuing=16, Write Issuing=16, Total Issuing=32
- Read Issuing=8, Write Issuing=8, Total Issuing=16
- Read Issuing=4, Write Issuing=4, Total Issuing=8



- QoS Enabled master
- Global Programmers View (GPV) - accessible through master_0

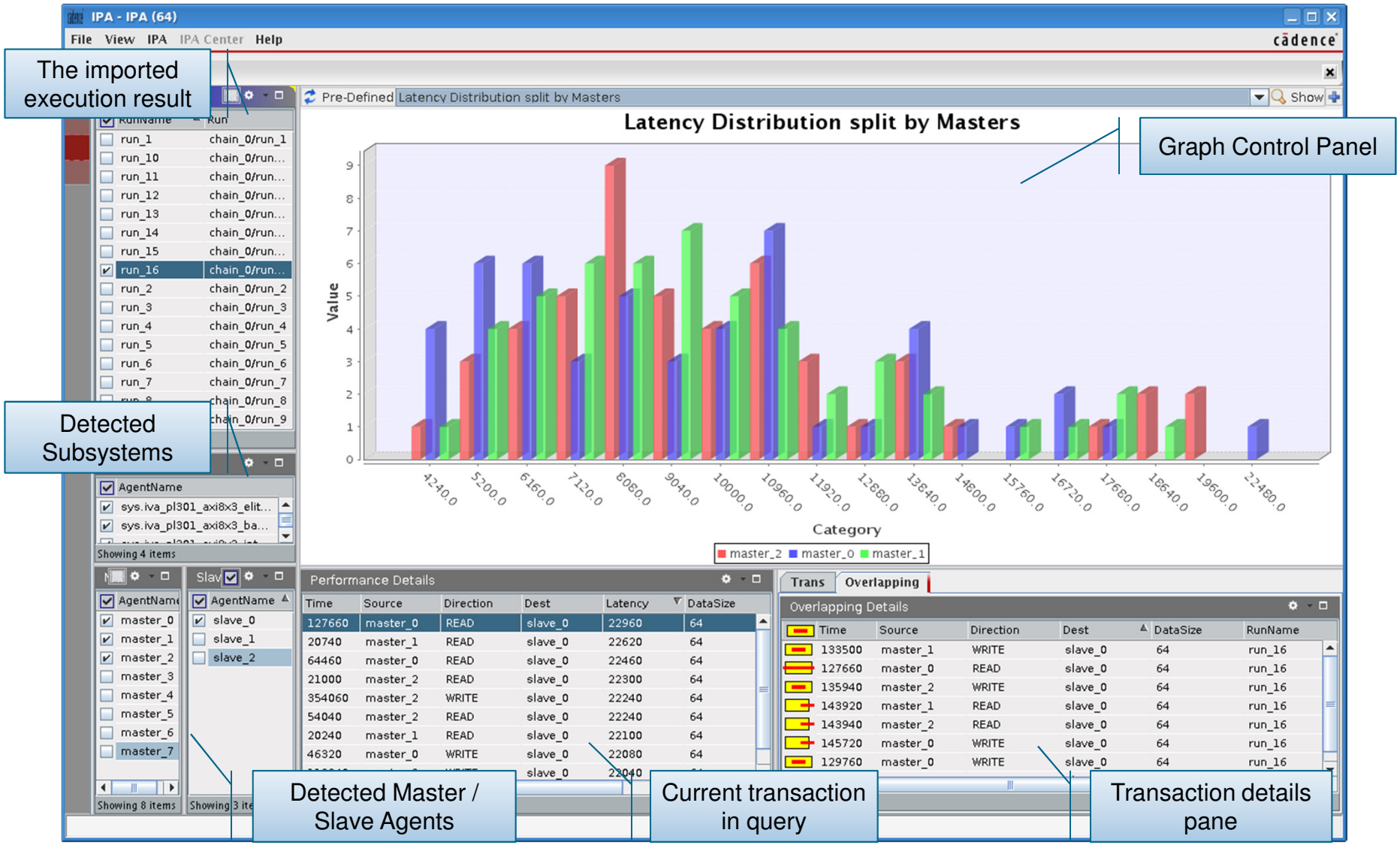
Performance Analysis Regression - DUT vs Scenario Matrix

- Performance Analysis gets really interesting when we can analyze data from related designs and use cases
- We have created a Performance Analysis Regression consisting of the family of 4 interconnects 'crossed' with the set of 4 Performance Use Cases as shown in the table below

	Scenario (Traffic Intensity)			
DUT Type	Low	Medium	High	Extreme
Basic	Run_1	Run_2	Run_3	Run_4
Intermediate	Run_5	Run_6	Run_7	Run_8
Advanced	Run_9	Run_10	Run_11	Run_12
Elite	Run_13	Run_14	Run_15	Run_16

Performance Analysis GUI

Latency Distribution split by masters



cadence

Interconnect solution Enables You to Verify Your Bus Based Systems

- Checking system behavior
  - Data consistency across the system
  - Checking interconnect and design specific features
- Providing system coverage
  - Ensure full coverage of system scenarios
- Understanding system scenarios
  - System debug aids
  - System Log information
- Validating system performance
  - Performance analysis and visualization


cadence®