



Using a SystemC Transactional Prototype for Design and Verification

Mark Milligan
VP of Marketing, CoWare Inc.



CoWare: The System-level Design Leader

● Largest investment in SLD technology

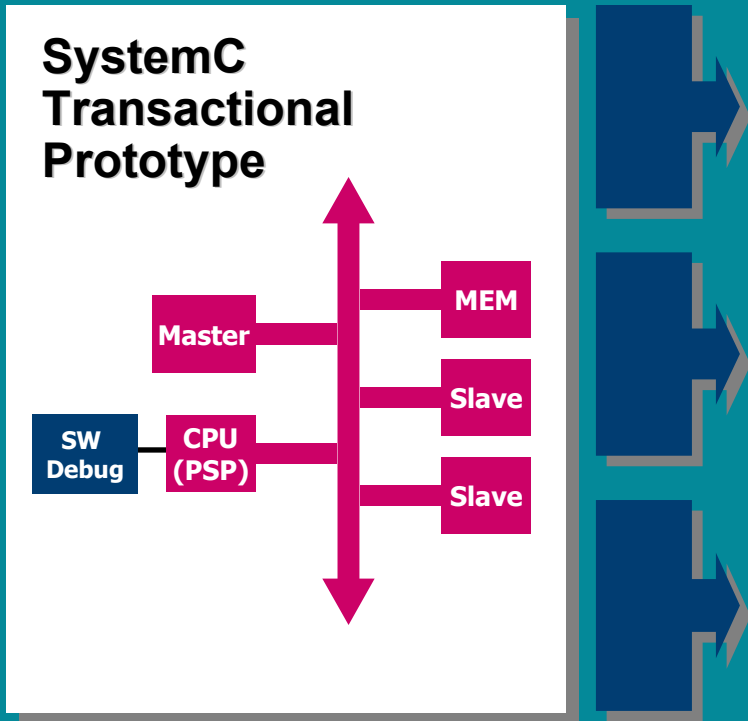
- Largest global R&D and support team, established in 1997
- Broadest set of solution technology: processor modeling and design, SoC platform design, and algorithm design
- Unique R&D partnerships with IMEC and RWTH Aachen, leading technology institutes in Europe
- Strong technical advisory board and world-wide university program

● Recognized experts in SLD

- Founded and launched the Open SystemC Initiative
- Cadence Design Systems' strategic partner for system-level design
- Greatest market share by revenue and installed base
- Selected by the world's largest consumer electronics company, world's largest microprocessor company, and Europe's largest semiconductor company

Top-down Verification

with a SystemC Transactional Prototype



1. Build the right thing

- *Rapidly create a system model for architecture validation with software*

2. Check as you build each block

- *Re-use as a high-speed, functionally complete system reference to verify RTL as it is successively refined*

3. Check you built the thing right

- *Link to partner verification flows to verify complete RTL in the system*

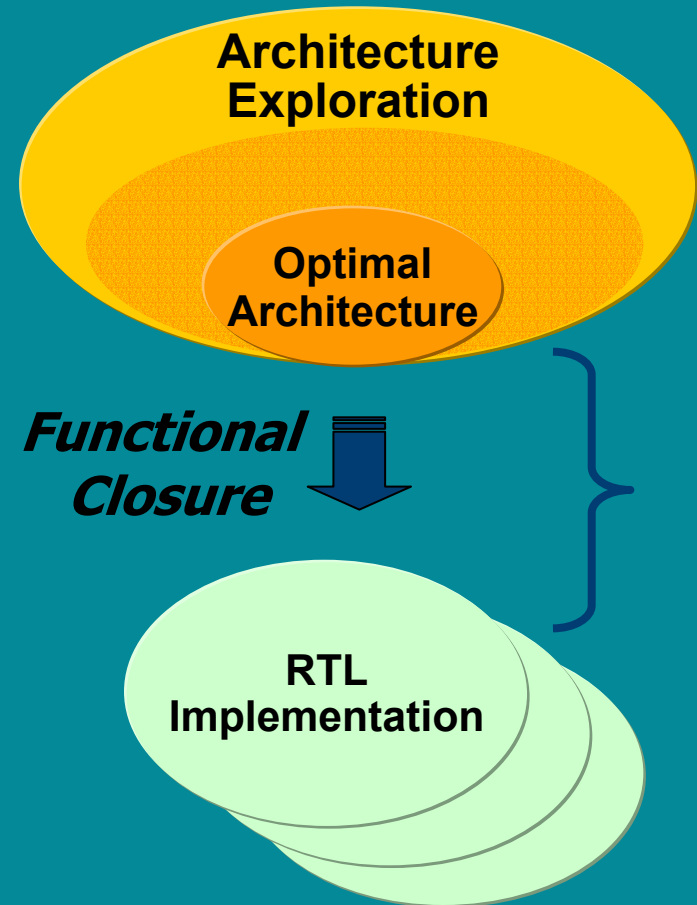
***More Verification,
Done Earlier***

CoWare

VERIFY
2003

Historical Barriers to Top-down Verification

- **No standard way of modeling hardware in C/C++**
 - Ad hoc modeling methods
 - Lack of model availability
 - Difficult to share and re-use
 - Project-based ROI
- **No Functional Closure between system model and RTL**
 - Remodeling effort
 - Loss of design intent
 - Re-creation of test cases
 - Divergence inhibits architecture reuse
- **Result: Re-spins!**



Historical Barriers to Top-down Verification

- No standard way of modeling hardware in C/C++

- Ad hoc modeling
- Architecture reuse
- Increased ROI

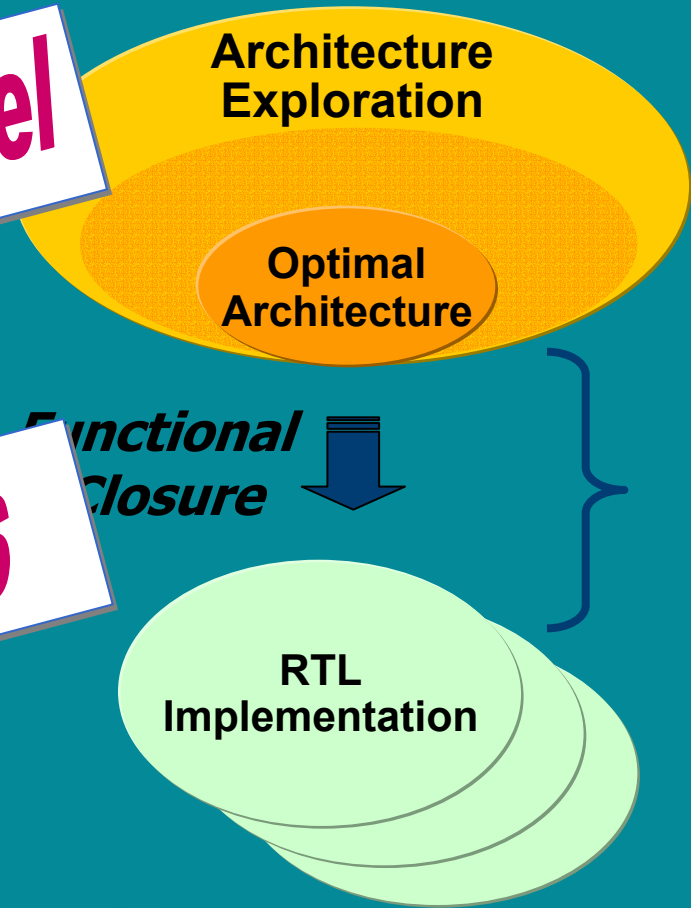
SystemC at transaction level

- No Functional Closure between system model and implementation

- Divergence in test cases
- Divergence inhibits architecture reuse

New methods and tools

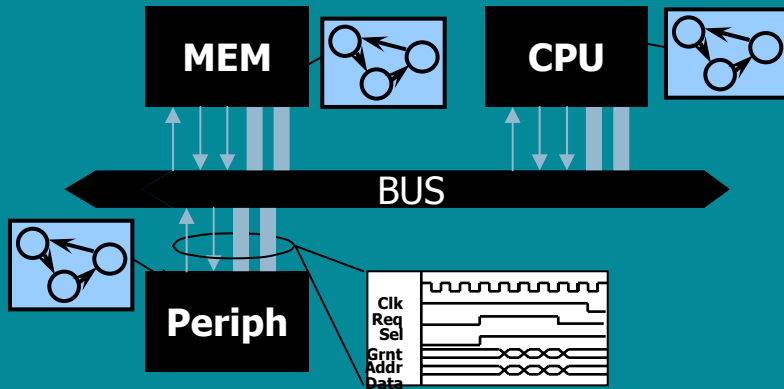
- Result: Re-spins!



CoWare

VERIFY
2003

What is Transaction Level Modeling?

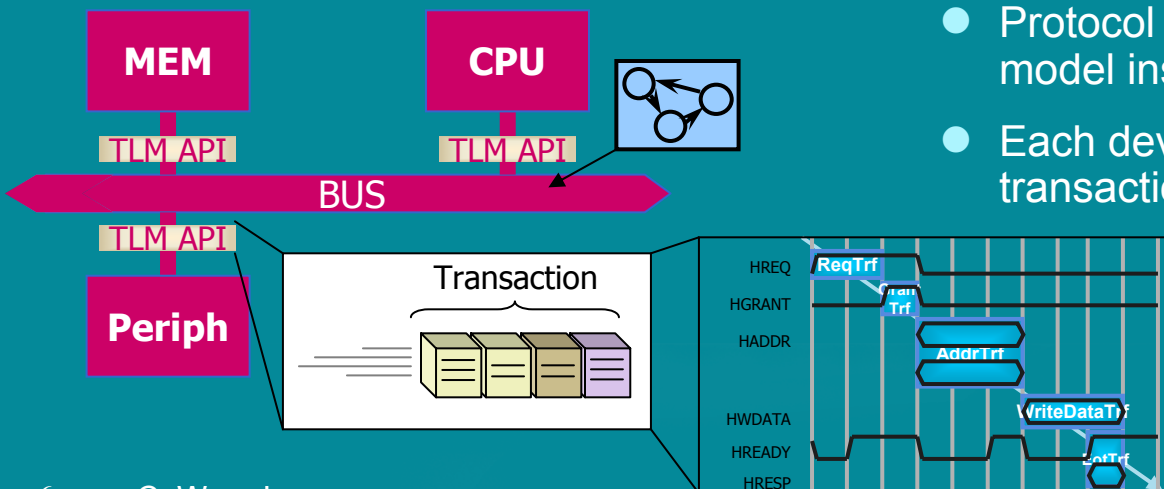


- **RTL bus: redundant complexity results in slow simulation**

- Each device interface must implement the bus protocol
- Each device on the bus has a pin-accurate interface

- **TLM bus: less code, fewer pins and events, yield faster simulation**

- Protocol is modeled as a single bus model instead of in each device
- Each device communicates via transaction level API

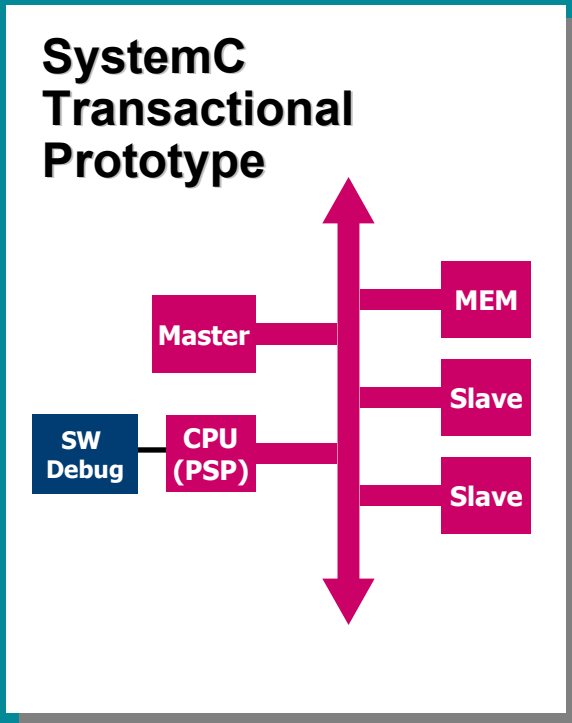


CoWare

VERIFY
2003

Top-down Verification

with a SystemC Transactional Prototype



1. Rapidly create a system model for architecture validation with software
 - *“Build the right thing”*
2. Re-use as a high-speed, functionally complete system reference to verify RTL as it is successively refined
 - *“Check as you build each block”*
3. Link to partner verification flows to verify complete RTL in the system
 - *“Check you built the thing right”*

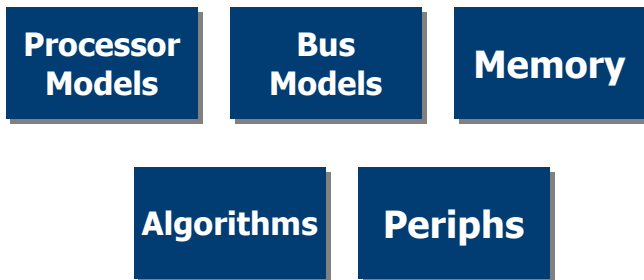
***More Verification,
Done Earlier***

CoWare

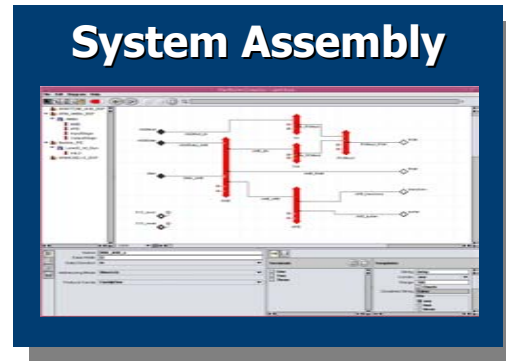
VERIFY
2003

Rapid Creation of System Models

- Open standard SystemC language
- Greater model availability
- Easier to share and re-use
- Automate platform creation with ConvergenSC



CoWare and Partner TLM models are available in the ConvergenSC Model Library

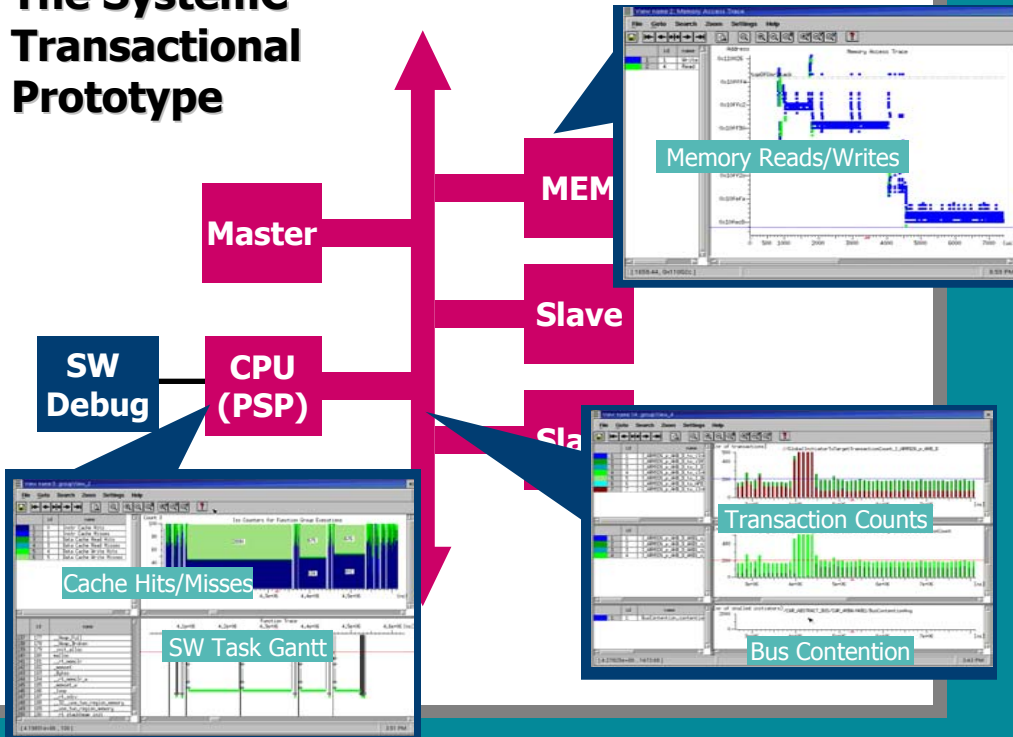


Rapidly configure bus topology and generate SystemC code in ConvergenSC

Architecture Validation with Software

“Build the right thing”

The SystemC Transactional Prototype



ConvergenSC
SystemC
Simulation
and Analysis

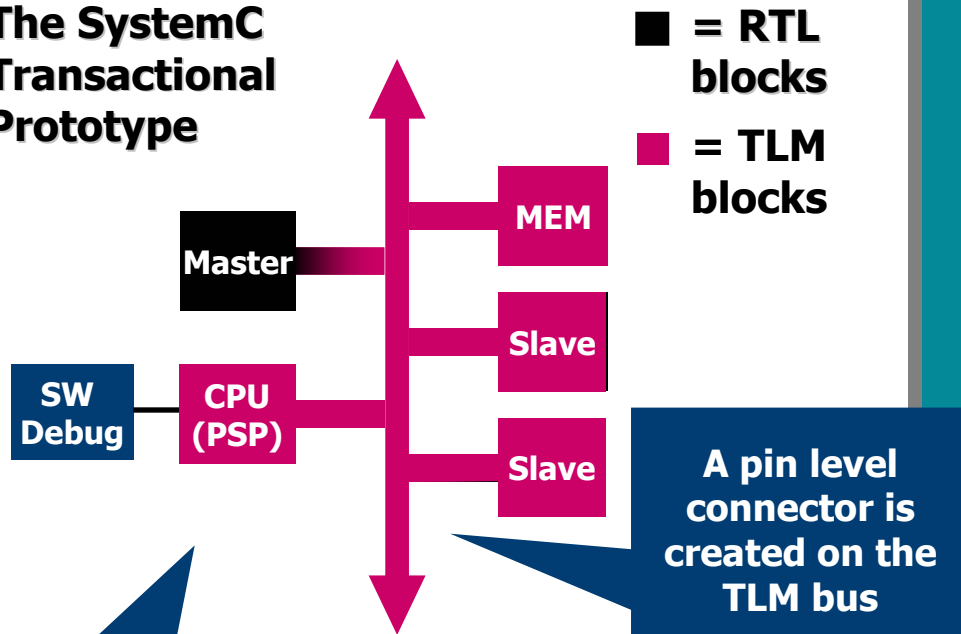
*Easily Deliver a Fast, Early
Model of the System for
Software Developers*

- **Fast and accurate enough for firmware**
 - Boots Linux in 5 minutes
 - Runs 2 to 5 times faster than OSCI Reference
 - Commercially supported
- **Powerful SystemC debug and analysis**
 - Fully SystemC-aware debug
 - Superior software, memory, and bus analysis

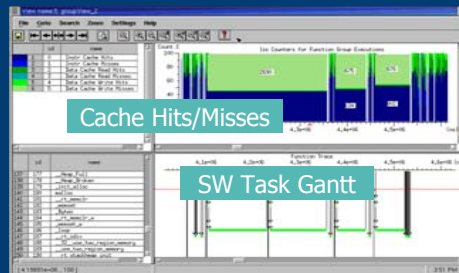
Introduce RTL into the TLM System

“Check as you build each block”

The SystemC Transactional Prototype



Each engineer can work in parallel, using the same system model to isolate and verify their RTL subsystem



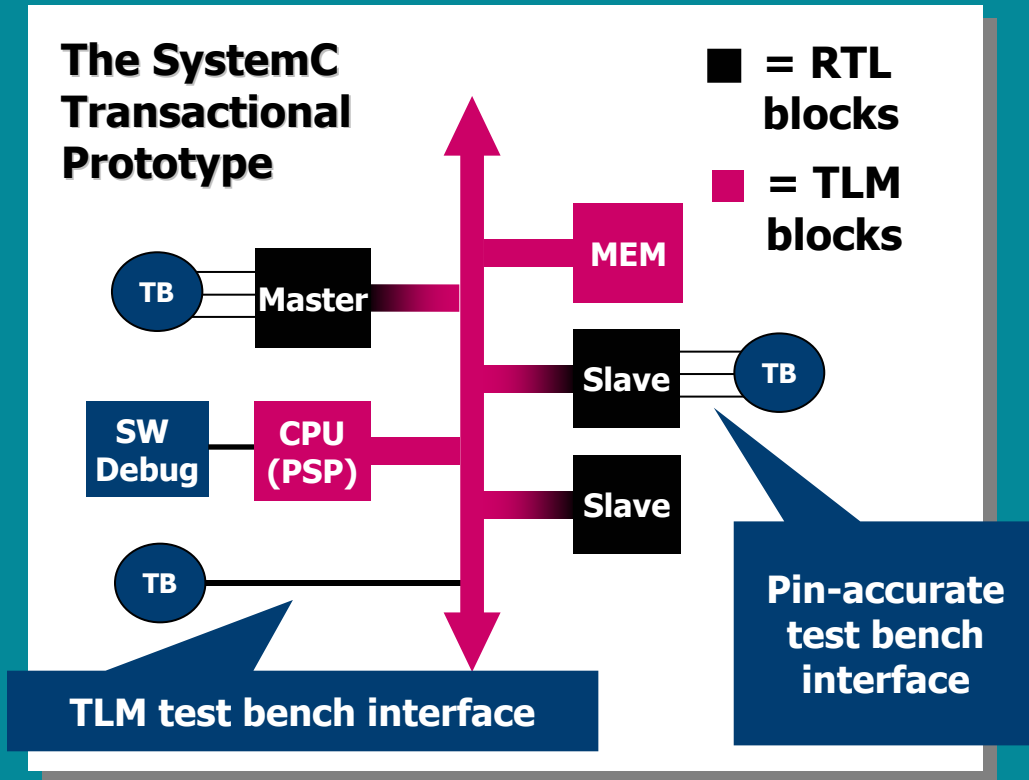
- Successively refine the sub-system RTL
 - ConvergenSC automates TLM/RTL bus connections
 - Simulates the whole system at the speed of a single block in HDL
- Re-run system-level analysis
 - Confirming the RTL performs properly in the system

CoWare

VERIFY
2003

Verifying RTL in the TLM System

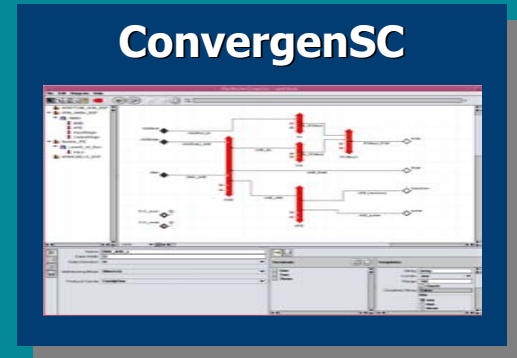
“Check you built the thing right”



- Verify RTL using test benches in SystemC or *e*
 - Pin-accurate and TLM
- Using Partner flows:
 - Randomize system data to improve coverage
 - Add assertions to enhance verification
 - Simulate exhaustively to verify RTL

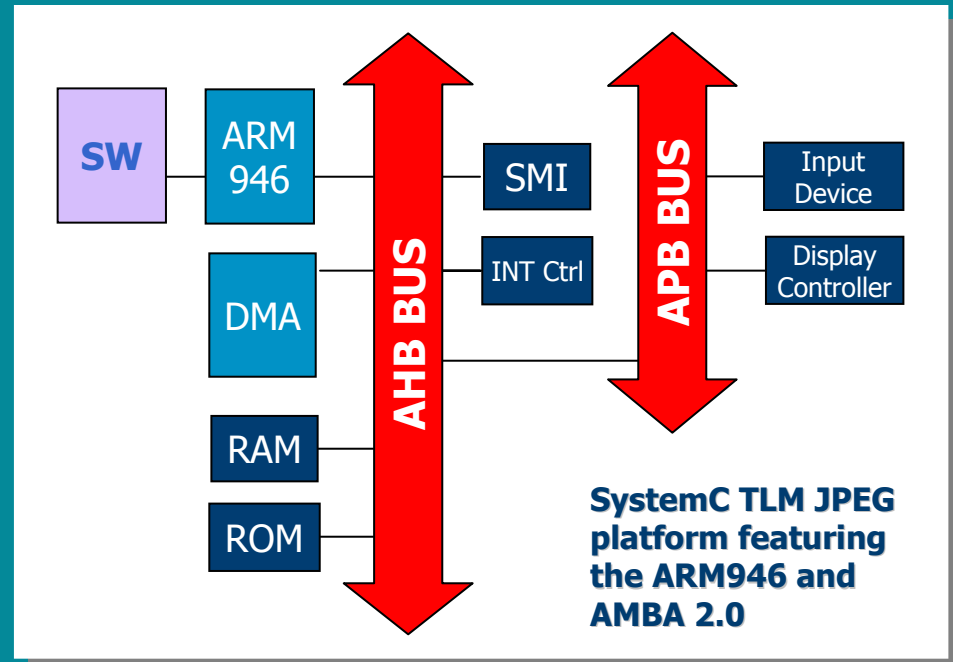
ConvergenSC Industry Partnerships

- Commitment to open standards
- Delivering commercially supported SystemC integrations with partner IP and verification flows



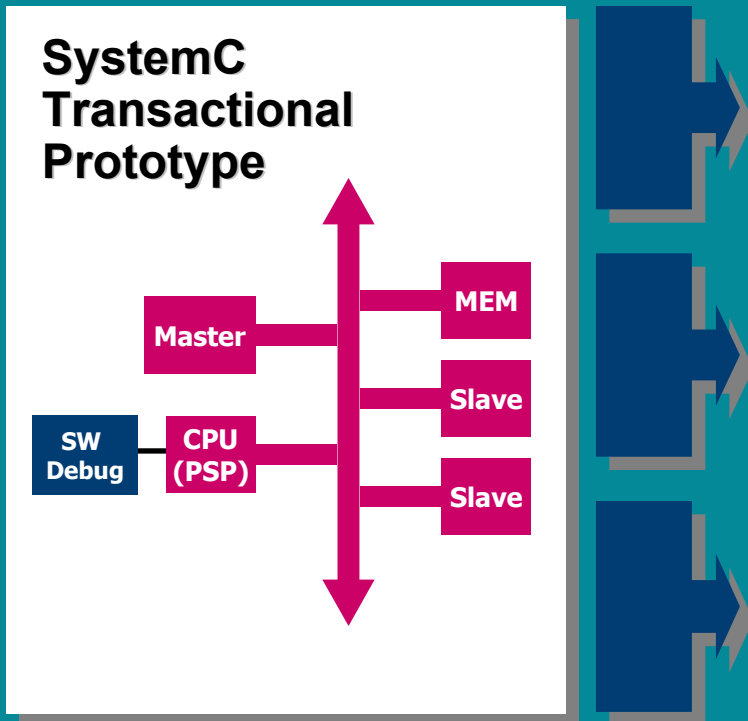
ConvergenSC Demonstration

- Rapid assembly of a SystemC Transactional Prototype
- Refinement of a TLM sub-system to RTL
- Using system-level analysis for verification



For more information on SystemC, ConvergenSC, and CoWare, visit: www.CoWare.com

Summary: Top-down Verification with a SystemC Transactional Prototype



1. Build the right thing

- *Rapidly create a system model for architecture validation with software*

2. Check as you build each block

- *Re-use as a high-speed, functionally complete system reference to verify RTL as it is successively refined*

3. Check you built the thing right

- *Link to partner verification flows to verify complete RTL in the system*

***Result: 1st Sample Up
and Running in Hours***