A TLM-driven Design and Verification Methodology

Brian Bailey Consulting
Work while consulting for Cadence Design Systems

Email: brian_bailey@acm.org
Tel:  503 632 7448
Cell: 503 753 6040
Web: brianbailey.us
Existing Methodology

System
Sub-System
Major Blocks
Smaller Blocks

Refine & Partition
Integration & Verification
Validation
Methodology Objective

- Validate Specification
  - System
    - Sub-System
      - Major Blocks
        - Smaller Blocks
      - Verify Micro-Architecture
    - Verify Architecture
  - Verify Implementation
Methodology Advantages

- Increases efficiency
- Early validation
- Incremental verification
- Increases testbench modularity and re-use
- Eliminates integration verification
- Verify important stuff first
  - Implementation details verified later
- More predictable schedules
Separating Concerns

• Separating Computation and Communication
  - Enables greater reuse
  - Enables functional virtual prototypes
  - Feeds high-level synthesis process
  - Independent refinement

• Separating Function and Architecture
  - One function, multiple implementation
  - Verify once
Multi-level metric driven

Functional specification

Plan

OVM VIP portfolio

Construct

Measure and analyze

Execute

Signoff?

Failure and metric analysis

Yes

No

Done

Testbench simulation, formal, hybrid, HW/SW co-sim, equivalence checking, CDC, constraint checking, LPV, MSV, sim acceleration, emulation

Source: Cadence Design systems

March 10

Copyright © 2005-2010 Brian Bailey Consulting
Three Stage Flow

1. Verification
   - C/C++/SC Simulation
   - High-Level Model Algorithm DUT

2. Verification
   - SystemC TLM Simulation

3. Verification
   - RTL Simulation
   - HW Accel

Source: Cadence Design Systems
**Design and Verification Flows**

**Testbench**

- **GP-OVC env**
  - sequencer
  - monitor
  - BFM

- **AXI OVC master**
  - sequencer
  - monitor
  - BFM

- sequence library

**Design**

- **Computation IP (C/C++)**
  - Registers
  - TLM interface (SC, TLM+GP)
  - Adapter
  - TLM interface (SC, TLM+GP)

- **Communications Unit**

**Source:** Cadence Design systems

*March 10* Copyright © 2005-2010 Brian Bailey Consulting
TLM Interface

- Start with basic transport mechanisms from TLM 1.0
- Must be synthesizable
  - Removes all simulation features
  - Removes all dynamic allocation capabilities
- Needs extra constructs
  - Reset capabilities
- Extend with TLM 2.0 capabilities
  - GP capabilities added
- Synthesizable TLM+GP interface to be offered for standardization