Verifying Multiple DUV Representations with a Single UVM-e Testbench

Matt Graham – Sr. Staff Product Engineer
DVCon 2014
Doubletree, San Jose
March 3-6, 2014

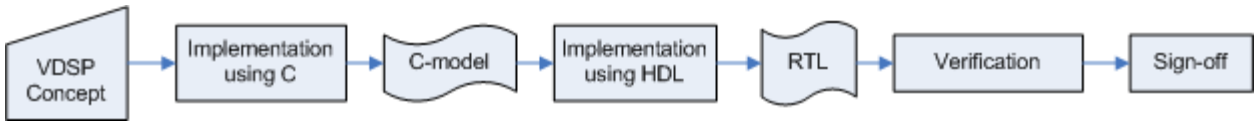
cadence®

Background
Typical IP Development Scenario

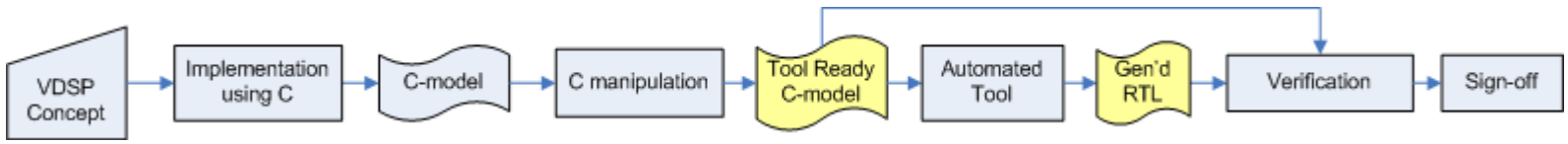
- SoC Development strives for Reuse and application of Advanced Methodology for both Design and Verification
  - Efficient application of these terms is non-trival
- Case Study:
  - Advanced processing IP development team within large semi-conductor manufacturer
  - Experienced DV engineers applying advanced methodologies including MDV and UVM
  - Constant pressure for improvement
    - Greater levels of complexity and integration
    - Shrinking headcount, budget, time (or all three!)
- Development teams are more and more looking for creative and innovative solutions to meet requirements within tightened constraints.

IP Development Flow

- IP development cycle ~6 months
- Processing algorithm development primarily in C/C++
- Digital design team implements algo in HW/RTL



  - New development flow to automate C ➔ RTL development
  - "Tool Ready" C model is not SystemC per conversion tool requirement



cadence

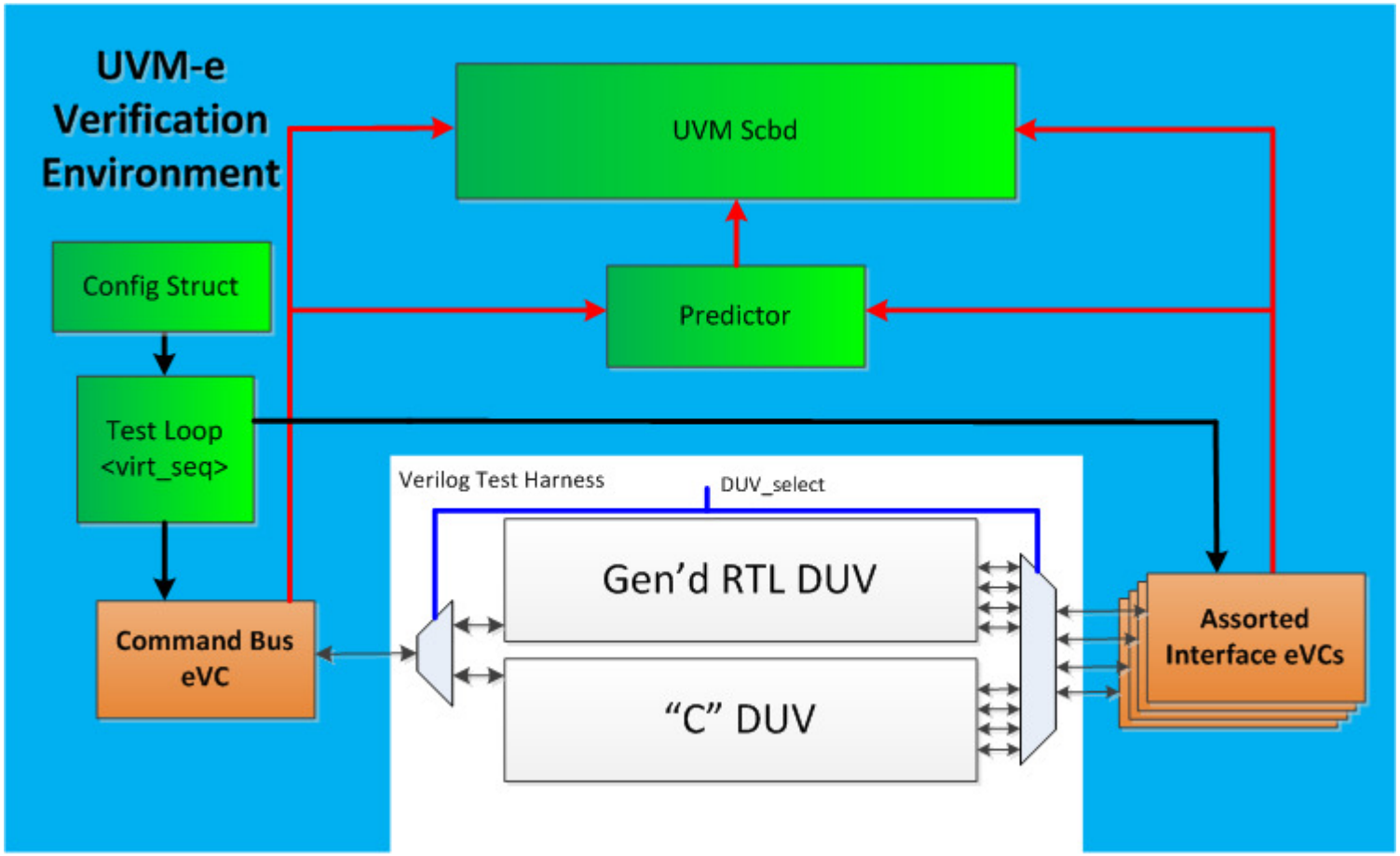
Problem Statement

- Algo developed in C, converted with tool to RTL
  - Provided code is C not SystemC
  - RTL availability will lag C delivery by about 5 weeks
- Sign-off process must take place on final RTL
  - Func Cov, all sims
  - Timing Dependent Interfaces must be verified
  - No formal method of proving C and RTL equivalent
- Require verification environment(s) for both DUV representations

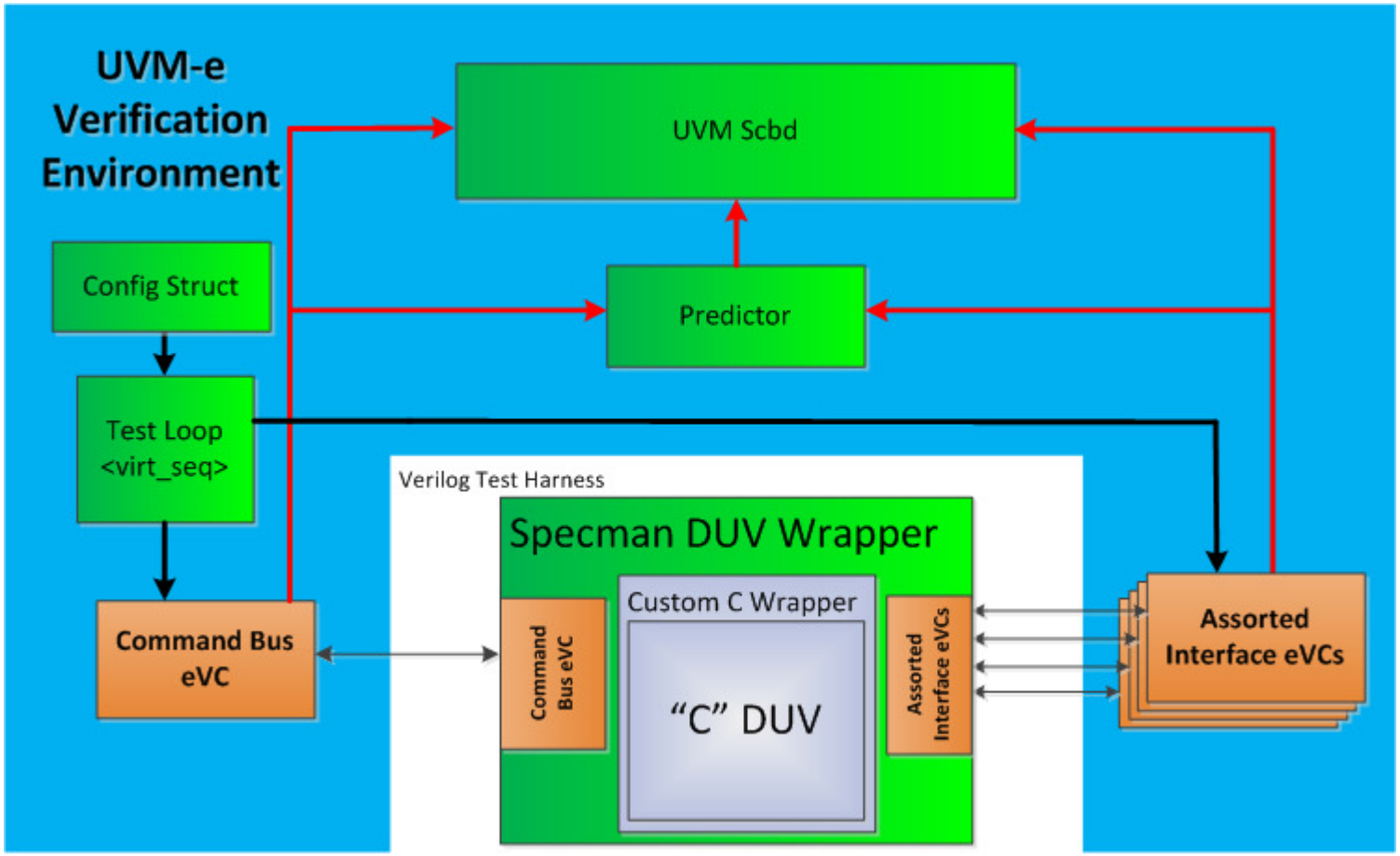
Goals

- Improve efficiency for debug
  - Utilize higher abstraction levels
  - Earlier availability of C code
- Utilize existing RTL verification IP (eVCs, coverage definition and test flow)
- Architect single UVM-e environment that can verify both RTL and “C” DUV implementations

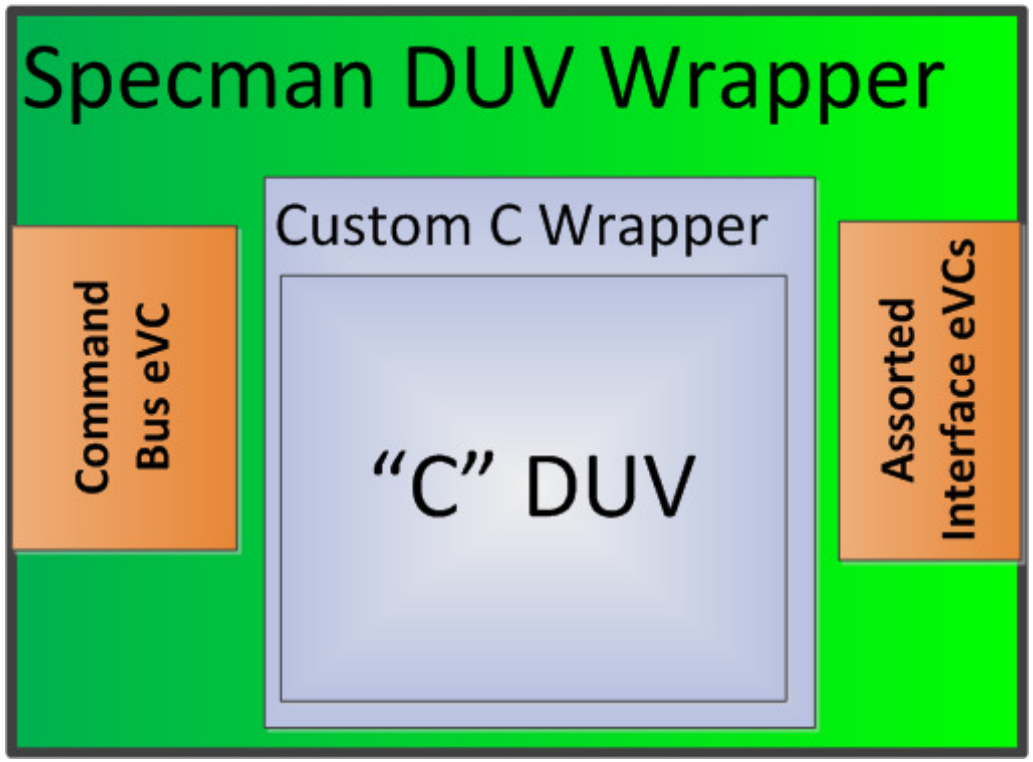
Verification Environment – Dual DUV



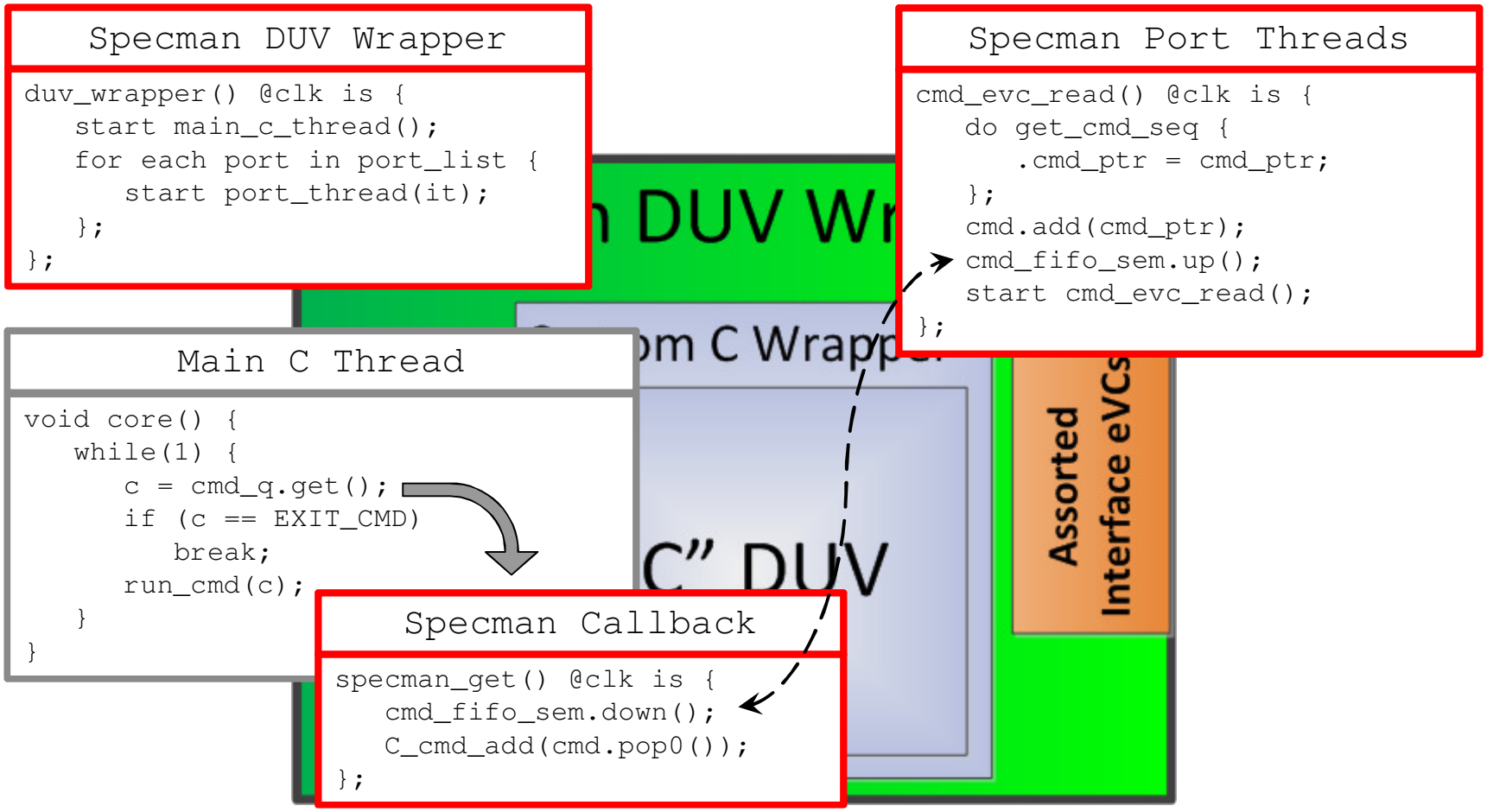
Verification Environment – “C” DUV



Verification Environment – UVM-e DUV Wrapper



“C” DUV – Timing Context



Results

- 10% total feature coverage achieved before any RTL was ready
- 14 of 59 design bugs found and fixed in this phase
- Single environment used for coverage capture and stimulus generation

Results - Challenges

- Live debugging of C code can be troublesome in mixed representation environment
  - Simple C based testbench driven by stimulus files from UVM-e Environment required for gdb debug

- Associated cost with developing e-to-C wrapper of model and interfaces
  - High Initial Investment: 4 person weeks
  - Low investment (1 person week) for subsequent projects (basic e-to-C interface layers now exist)

Results – Lessons Learned

- SystemC based wrapper warrants further investigation
  - Enables timing information embedded in C DUV
  - Simplified debug for SystemC through native tool support
  - Easier connection to VE via standard TLM
  - Coverage collection built in

- Ability to dynamically select DUV representation is a major benefit