Systematic Application of UCIS to Improve the Automation on Verification Closure

Christoph Kuznik, Marcio F. S. Oliveira, Gilles Bertrand Defo, Wolfgang Mueller
University of Paderborn / C-LAB
Germany
{kuznik, marcio, defo, wolfgang}@c-lab.de
Motivation – Verification, a language feat.?

Despite SystemC language is one of the main drivers for virtual prototype development its verification features are rather limited:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Language</th>
<th>IEEE-1800 SystemVerilog</th>
<th>IEEE-1647 e</th>
<th>IEEE-1666 SystemC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage Facility</td>
<td>++</td>
<td>+++</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Assertions</td>
<td>+++</td>
<td>+++</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>RTPG</td>
<td>+++</td>
<td>+++</td>
<td>+ (SCV)</td>
<td></td>
</tr>
<tr>
<td>Verification Methodology</td>
<td>+++</td>
<td>+++</td>
<td>x(^1)</td>
<td></td>
</tr>
<tr>
<td>TLM</td>
<td>++</td>
<td>+</td>
<td>+++</td>
<td></td>
</tr>
<tr>
<td>AOP</td>
<td>x</td>
<td>++</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>C-Software Simulation</td>
<td>simulator dependent</td>
<td>simulator dependent</td>
<td>+++</td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) not with OSCI simulator
Motivation – Verification, a language feat.?

- Previous work:
  - Establish a functional coverage and verification methodology for SC

- With this foundation, let's improve the automation on verification closure in SC TBs

- Enhance the flow from coverage plan capture to functional coverage implementation within the testbench

  - Basis:
    - recently released Unified Coverage Interoperability Standard (UCIS)

- Target Language/Simulator: SystemC Reference Simulator
Outline

- Motivation
  - Verification Features of HLDVLs
  - Abstraction and Automation for Verification Closure Productivity

- UCIS to Improve the Automation on Verification Closure in SC
  - UCIS and Verification Process
  - Steps in Detail

- Case Study
  - Tooling for UCIS with OSCI SystemC

- Lessons Learned

- Final Remarks
Unified Coverage Interoperability Standard

Why rely on UCIS?

- A schema for creation, merge, export of coverage information
- Interoperability across simulators
- Use models that cover common verification activities

Approach

- Utilize UCIS to store, generate & accumulate coverage plan metrics in SystemC simulation with OSCI reference simulator!

Can answer *Does it work?* and *Are we done?* question

Coverage Producers

Coverage Consumers

Simulation

Static Checks

Formal

Emulation

SystemC Coverage Database

RTL Annotation

Testplan Update

Report generation
UCIS with OSCI SystemC

- Challenges
  - Accelera UCIS v1.0 Std.
    - API functions description
    - API header file
    - XML schema
  - Consequently, own API implementation needed!
  - API = Setter / Getter / Advanced

- In this work, we focus on the functional coverage scope UCIS_CVG_SCOPE
UCIS and Verification Process

- MDE enabled process

1. Verification Plan (in particular: Coverage Plan)
   Specification

2. Feature List
   Coverage Plan
   Checker Plan
   Schedule

3. Verification Environment
   (in particular: Func. Coverage Metrics)

4. Implementation

Insufficient automation support, error-prone manual tasks

Verification Plan with systematized Coverage Plan
Specification

Improved automation, soundness

Systematized assisted / automatized

Systematized assisted / automatized
UCIS and Verification Process cont’

- 1) Capture cover plan data in spreadsheet style data model (language independent)
- 2) Transform cover plan data to UCIS metric model (in fact a template of the metric to generate)
- 3) Generate func. coverage metric skeletons from the UCIS metric model
- 4) Assisted completion of the verification environment skeletons

- concrete tooling
Case Study: Adaptive Cruise Controller

- Architecture of ACC model

- DUV has various I/O, functional coverage is a valuable metric to determine end of testing

- Proceed the individual methodology steps
  - Build coverage plan
  - Use UCIS to automatize metric skeleton generation
Case Study: Verification Planning

Coverage Plan Spreadsheet structure

- Excel table vs. custom coverage plan format
- Usage of OMG Requirement Interchange Format (ReqIF)
  - Allows to define a custom „coverage plan spreadsheet“
  - MDE/Eclipse tooling compatible

Case Study Example

<table>
<thead>
<tr>
<th>Name</th>
<th>Range</th>
<th>Type</th>
<th>Weight</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>desired_speed</td>
<td>[10:100]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>current_speed</td>
<td>[0:100]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>desired_distance</td>
<td>[10:30]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>current_distance</td>
<td>[0:150]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>enable_acc</td>
<td>[0:1]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>enable_dist</td>
<td>[0:1]</td>
<td>BIN</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>
Case Study: Model Mapping

- Multi Pane Editor View

  - Mapped model contain references to both UCIS metric template and design model

- A model of UCIS can be generated using MDE tooling, moreover API Setter/Getter functions can be generated
Case Study: Generated Metrics

```cpp
// Init the factory
svm_pFac = svm_Factory::init();

// Set UCIS data model format
svm_pFac->setCoverageDb("acc_sim_cov.xml");

// Specify metric, covergroups
cv_pCG = svm_pFac->getCoverGroup("ACC_DUV");

// Specify metric, bin types
cv_pBa = svm_pFac->newBins(cov_pCP, "CURRENT_SPEED", AUTOBINS);

// Connects the bin types to the metric
(cv_pBa << range(20, 49) << range(50, 69) << range(70, 100));
(cv_pBa->connect(current_speed));

(cv_pBb = svm_pFac->newBins(cov_pCP, "DESIRED_SPEED", AUTOBINS);
(cv_pBb << range(10, 49) << range(50, 69) << range(70, 100));
(cv_pBb->connect(desired_speed));
(...)
```
Case Study: Simulation

- Stimuli = Computation Tree Method + Random
  - Tree-oriented decomposition of test scenarios into individual variable ranges
  - RTPG for the selected ranges

- Coverage
  - Usage of our functional coverage prototype for SystemC
  - Accumulation of hits w.r.t. metric

```
1  BIN: ACC_SPEED_CTRL:desired_speed::: 2014 Hits
2  BIN: ACC_SPEED_CTRL:current_speed::: 2189 Hits
3  BIN: ACC_SPEED_CTRL:desired_distance::: 2077 Hits
4  BIN: ACC_SPEED_CTRL:current_distance::: 2338 Hits
5  BIN: ACC_SPEED_CTRL:enable_ac::: 41 Hits
6  ...
```
Case Study: Simulation cont’

- Output to UCIS format
  - Usage of generated Setter/Getter functions (w.r.t. schema)

Example will be inserted asap
Lessons learned: UCIS

- "Missing" reference API Writer
  - Effort to build Setter/Getter API for C/C#/… w.r.t. XML schema is low
  - Advanced API commands implementation requires in-depth expertise in UCIS and its use models

- Slight inconsistencies of API and XML
  - Donation and glue residue

- A common standard format for *Does it work?* and *Are we done?*
  - A great thing ;-)
Lessons learned: MDE Tooling Effort

- Eclipse Tooling Construction
  - the devil is in the details…
  - ReqIF, EMF, EMF4CPP, specific Eclipse version requirements and dependencies
  - MDE expertise definitely necessary

- Eclipse Tools Usage
  - Editors for coverage plan, UCIS, model weaving work
  - BUT: domain-specific (EDA) view preferable
Conclusion

- We see potential for usage of UCIS in earlier phases of verification, in particular verification/coverage plan creation.

- Building Setter/Getter API for UCIS is easy.

- General: FC metric generation can avoid error-prone manual coding.
References

- Accellera Organization Inc.: *Unified Coverage Interoperability Standard (UCIS)*
  Link: http://www.accellera.org/downloads/standards/ucis

  Link: http://dl.acm.org/citation.cfm?doid=2380445.2380497


- Christoph Kuznik, Wolfgang Mueller: "A SystemC Based Library for Functional Coverage," DVCON 2011, San Jose, February 2011,
This work was partly funded by the DFG Collaborative Research Centre 614 and by the German Ministry of Education and Research through the project SANITAS (16M3088I).
Thank you for your attention.

Christoph Kuznik, Marcio F. S. Oliveira, Gilles Bertrand Defo, Wolfgang Mueller

University of Paderborn / C-LAB Germany

{kuznik, marcio, defo, wolfgang}@c-lab.de