DVCon 2012

Design & Verification Conference & Exhibition

A SystemC Library for Advanced TLM Verification – Towards an *Enhanced* OVM / UVM for SystemC

Marcio F. S. Oliveira, Christoph Kuznik, Wolfgang Mueller

University of Paderborn / C-LAB
Germany

marcio@c-lab.de kuznik@c-lab.de
wolfgang@acm.org

UNIVERSITÄT PADERBORN c-lab

Wolfgang Ecker, Volkan Esen

Infineon Technologies
Munich, Germany

Wolfgang.Ecker@infineon.com
Volkan.Esen@infineon.com

Infineon

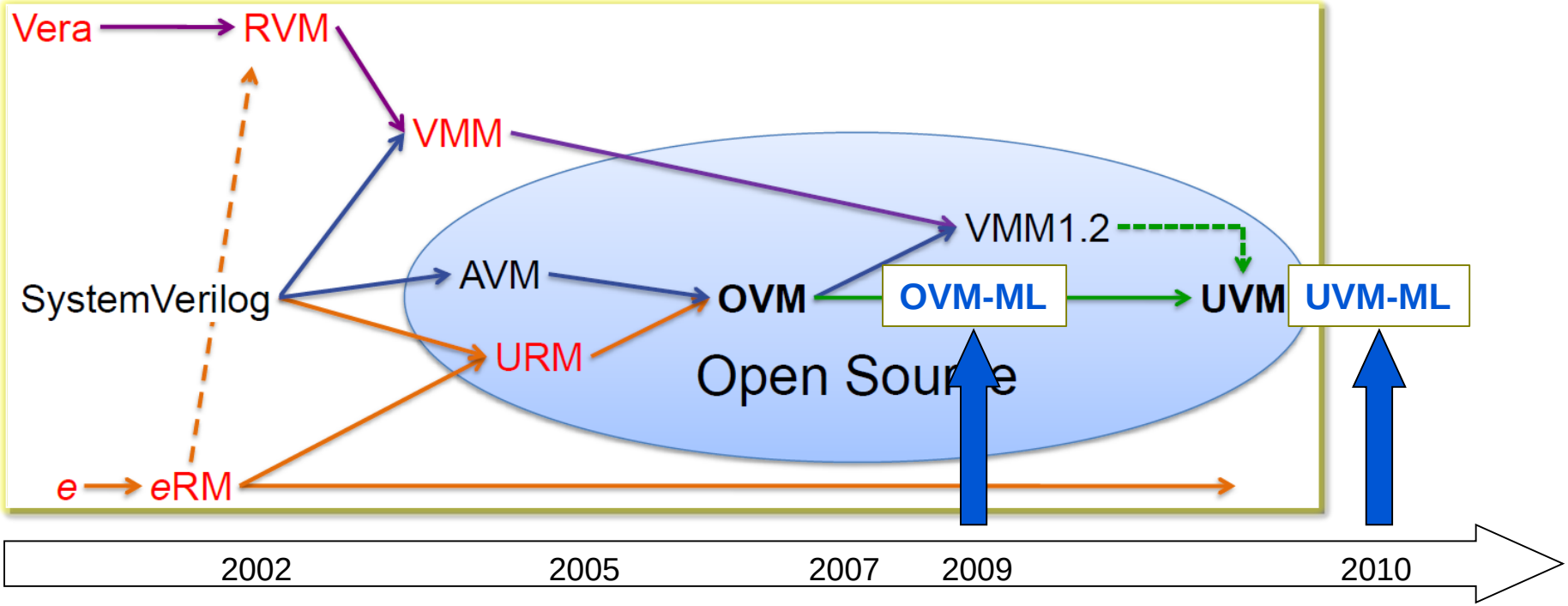
Februar 28, 2012

Agenda

- Introduction
  - Verification Methodology Libraries: Quick History Review
  - OVM and UVM for SystemC
- System Verification Methodology Library
  - Library Overview
  - Available Features
- Library Examples
- Final Remarks

Introduction

Verification Methodology Libraries: Quick History Review



Introduction

OVM and UVM for SystemC

- Partial implementation of OVM/UVM base package:
  - Phasing
  - ovm_component
  - Configuration
  - Factory
  - Packing

Provides interoperation with SystemVerilog Environment

No verification methodological expertise transfer from SystemVerilog to SystemC version

System Verification Methodology Library - SVM

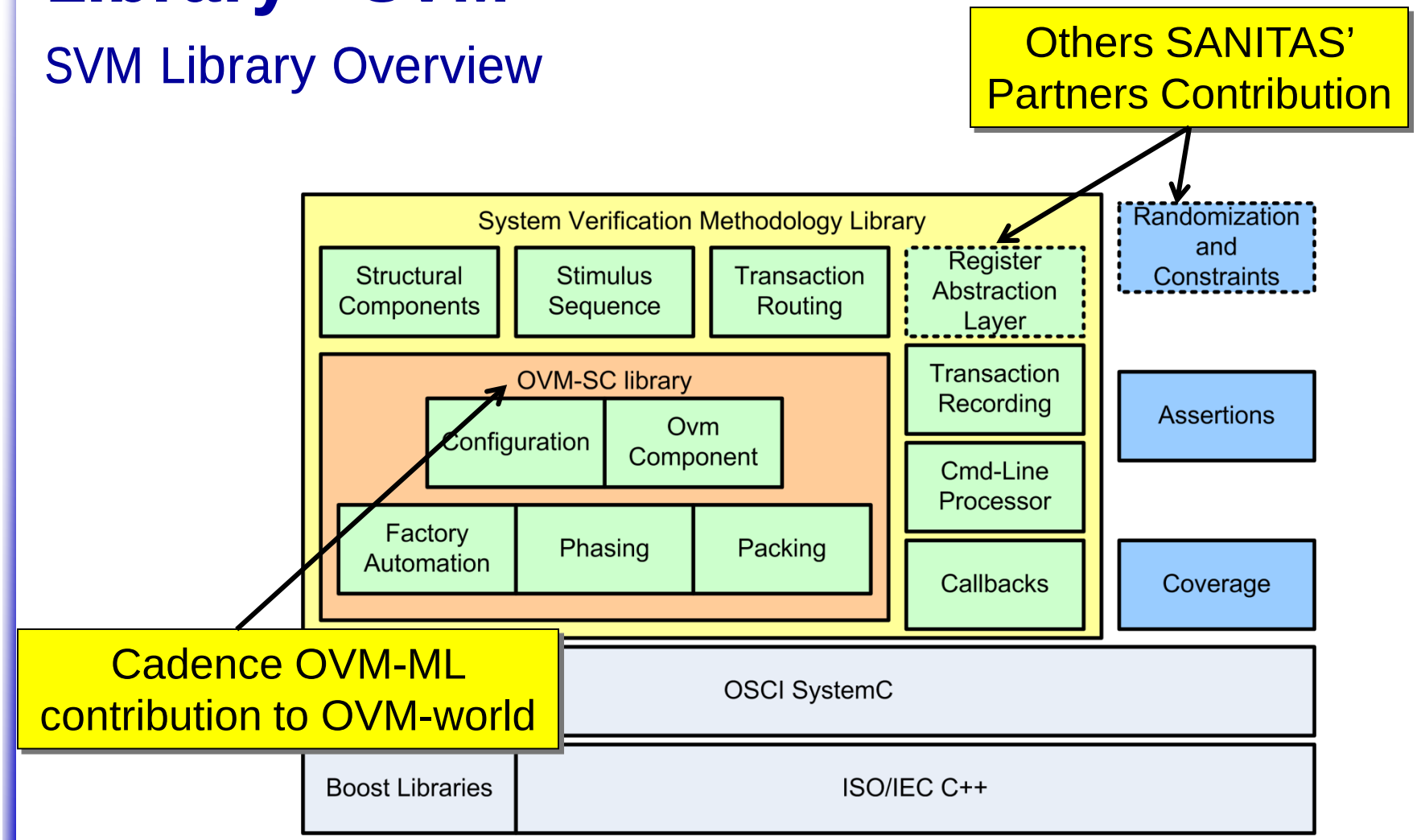
- SANITAS project: Improvements for verification at TLM
  - Industrial automation
  - Automotive
- OVM and SystemC as basic technologies

- Vendor neutral implementation

- Promotes the convergence to unified verification methodology library for SystemVerilog and SystemC

System Verification Methodology Library - SVM

SVM Library Overview



System Verification Methodology Library - SVM

Base Package

- Add features:
  - Callbacks
  - Transaction routing
  - Transaction recording
  - Command-line processor
- Alignment of OVM and SystemC Simulation Phases
- Emulation of Connection Phase
- Limitation: no full hierarchical path name for binding

System Verification Methodology Library - SVM

Component package

- Provides structural components for systematic testbench development
- Includes Testbench, Environment, Agent, Sequencer, Driver, Monitor, Scoreboard, Subscriber
- Adds transaction API to components
- Use of TLM to TLM and TLM to RTL interfaces

System Verification Methodology Library - SVM

Component package

```
class ActorAgent : public svm_agent {

  tlm::tlm_analysis_port<tlm::tlm_generic_payload > aport;

  ActorDriver *pDriver;
  ActorMonitor *pMonitor;
  ActorSequencer *pSequencer;

  SVM_COMPONENT_UTILS(ActorAgent)

void ActorAgent::build()
{
    svm_agent::build();
    get_config_int("debug", debug);

    pSequencer = DCAST<ActorSequencer*>(
    svm_factory::create_component(
        "ActorSequencer", "","pSequencer") );
…
```

System Verification Methodology Library - SVM

Component package

```
void ActorDriver::run()
{
  while(true)
  {
    IfxCommandItem *item = new IfxCommandItem();

    seq_item_port->get(*item);

    //Converts the sequence item to generic_payload
    …
    // Drives the TLM DUT's ports
    _iSktOfSif->b_transport(trans,myT);

    //Syncronize
    myT += sc_time(1000,SC_NS);
    wait(myT);

  }
}
```

System Verification Methodology Library - SVM

Sequence Package

- Decouple stimuli from the component hierarchy
- Provides advanced stimuli management
- Ordered stream of high level behavior defined as a set of transactions, with routing of responses to request
- Built-in Sequences: random, simple, all_sequences

System Verification Methodology Library - SVM

Sequence Package

```
class IfxCommandItem : public svm_sequence_item, public rand_obj
{
  randv<IfxCommandValT> command;
  randv<unsigned int> degree;
  randv<unsigned int> percent;

  IfxCommandItem(const std::string& name)
    : svm_sequence_item(name)
  {
       constraint(0 <= degree() && degree() <= 255);
       constraint(0 <= percent() && percent() <= 99);
  }
…
SVM_OBJECT_UTILS(IfxCommandItem);
};
```

System Verification Methodology Library - SVM

Sequence Package

```
void ActorTest::run()
{
  SequenceOfCommands *pSequenceOfCommands;
  pSequenceOfCommands = DCAST<SequenceOfCommands*>(
    svm_factory::create_object(
    "SequenceOfCommands","","pSequenceOfCommands",false) );

  pSequenceOfCommands->start(
      pActorAgent->pSequencer, 0, 100, true );

  svm_manager::get_instance()->stop_request();
}
```

System Verification Methodology Library - SVM

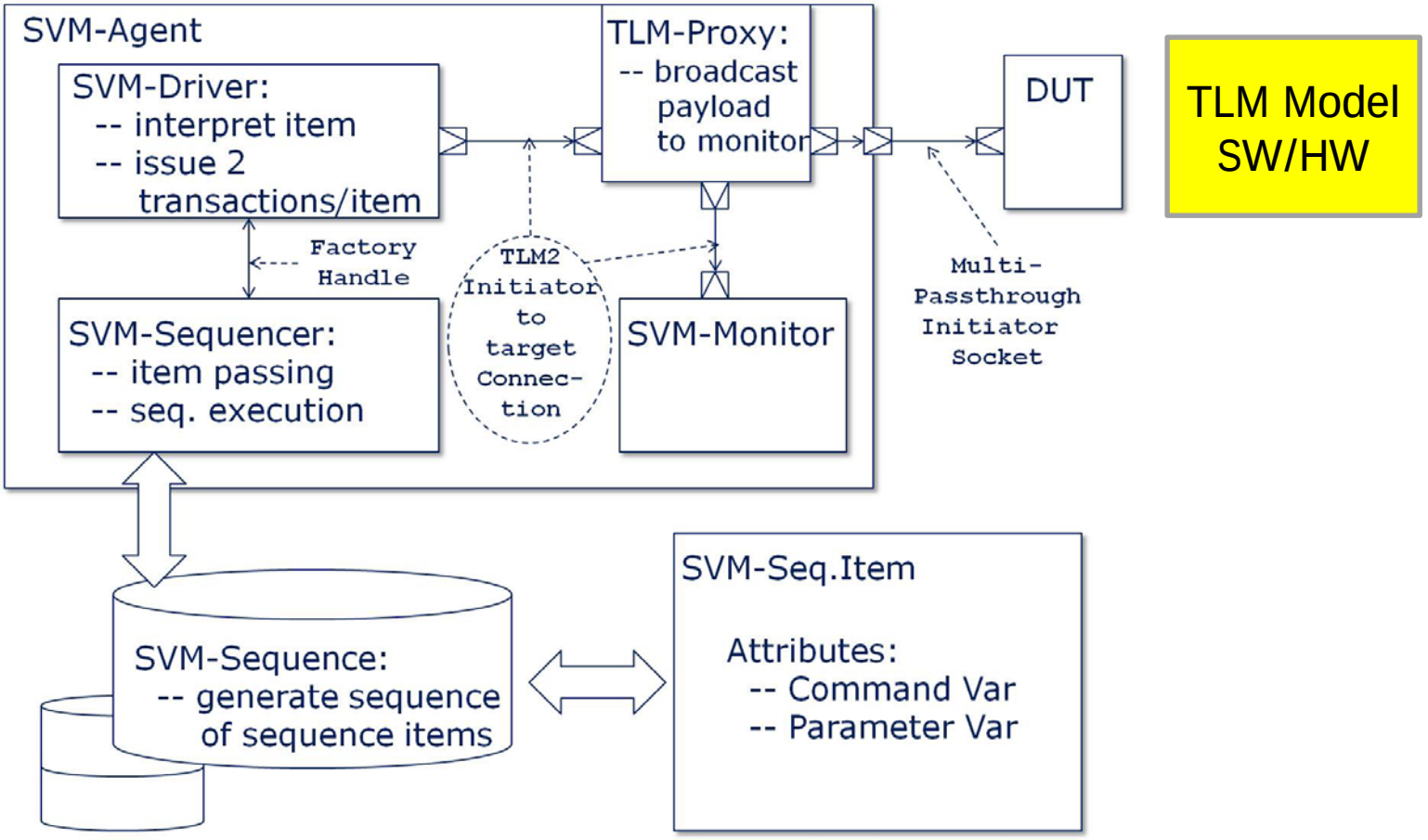
Sequence Package

```
void SequenceOfCommands::body()
{
  static Generator<Context> inline_constraints;
  static rand_vec<IfxCommandValT> commands;
  inline_constraints(commands().size() <= 20);
  inline_constraints(commands().size() >= 10);
    …
  SequenceOneCommand *pSequenceOneCommand;
  for (int i = 0; i < commands.size(); ++i)
  {
    pSequenceOneCommand = DCAST<SequenceOneCommand*>(
       svm::svm_factory::create_object("SequenceOneCommand", "",
       "pSequenceOneCommand", false) );

    start_item(pSequenceOneCommand);
    pSequenceOneCommand->fixedCmd = commands[i];
    finish_item(pSequenceOneCommand);
  }
}
```

Feature	OVM	UVM	OVM-SC	SVM
Call-backs	Yes	Yes	No	Yes
Comparison	Yes	Yes	No	No
Methodology Components	Yes	Yes	No	Yes
Phasing	Yes	I	Yes	Yes
Polices	Yes	Yes	No	No
Recording	Yes	Yes	SCV	I
Register Abstraction Layer	Yes*	Yes	No	Yes**
Reporting	Yes	Yes	SC	SC
Transaction Routing	Yes	Yes	No	Yes
Sequencing / Stimulus	Yes	Yes	No	Yes
Assertion	SV	SV	No	I
Coverage	SV	SV	SCV	I
Randomization / Cosntraints	SV	SV	SCV	I

Library Example



Library Example

- TLM-Proxy broadcasts transactions to DUT and Monitor
- Regular TLM2 interfaces connects DUT and VC
  - Delta-free connection
  - Better performance by applying Quantum Keeping
- Plain SystemC provides several advantages
  - Avoid delta-cycles between DUT and testbench
  - Enables the verification of timing-abstraction such as TLM+ and quantum keeping

Final remarks

- A (fully) implemented Verification Methodoly Library for SystemC based on OVM
- Provides advanced features for TLM verification
- Improved the verification methodology for SystemC by adding new features as Components, Sequences, etc
- Integrated Coverage, Assertion and Randomization

Outlook

- Alignment with other outcomes of SANITAS project
  - Integration of Register Abstraction Layer
- Improvements in the API
  - Conformance to UVM
  - New features
- SystemC revision P1666-2011
  - Exploit the new features

Acknowledgement

SPONSORED BY THE

Federal Ministry of Education and Research

This work was partly funded by the DFG Collaborative Research Centre 614 and by the German Ministry of Education and Research (BMBF) through the BMBF project SANITAS (01M3088).

SANITAS

Thank you for your attention!

Are there any questions?

Library Example