Bridging the Gap: Consistent SystemC Code Generation for Multi-Abstraction-Level State Charts

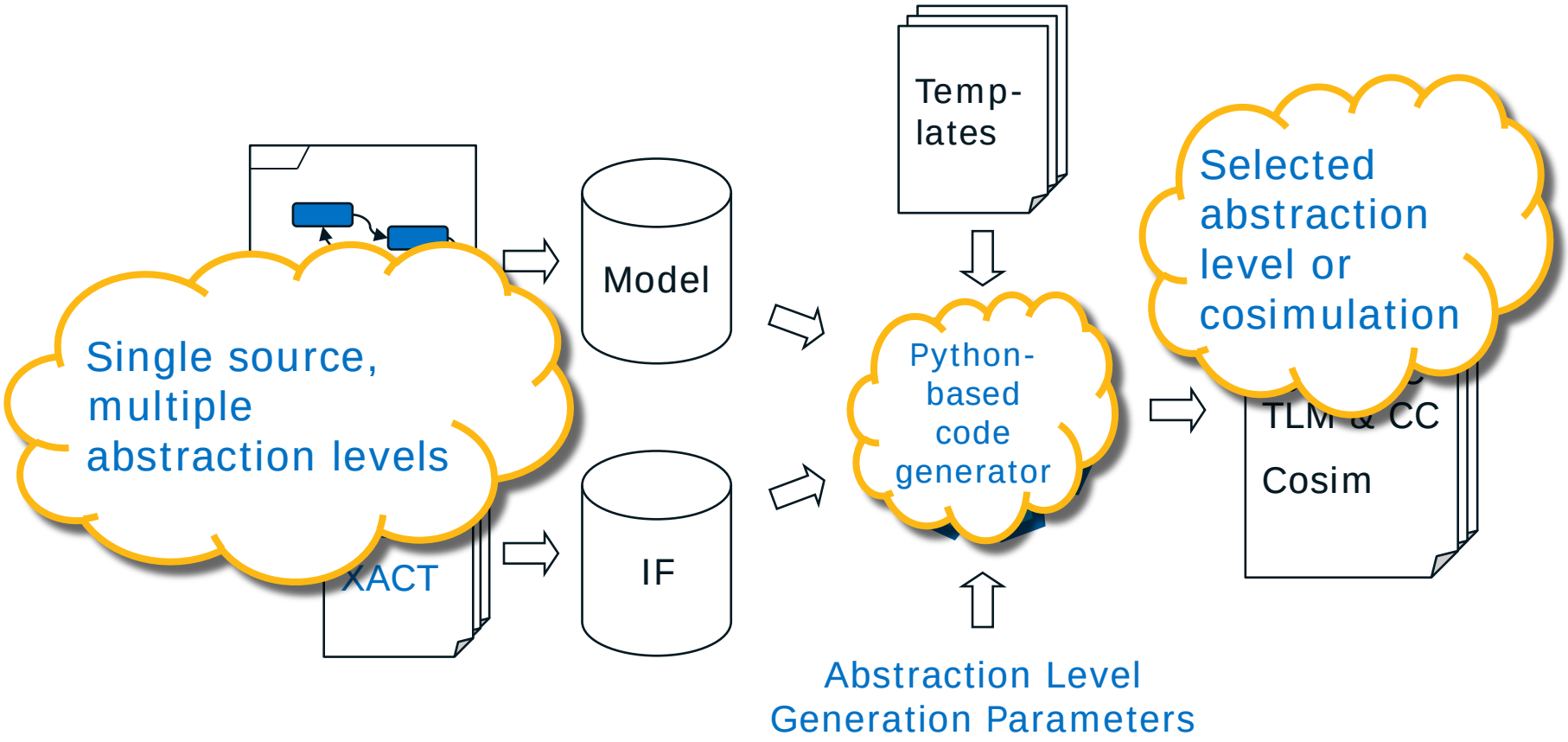
Rainer Findenig, Wolfgang Pauli,
Thomas Leitner, Wolfgang Ecker

wolfgang.pauli@intel.com

Agenda

1. **Single Source Modeling and Code Generation**
2. Ensuring Consistency
3. Results
4. Summary

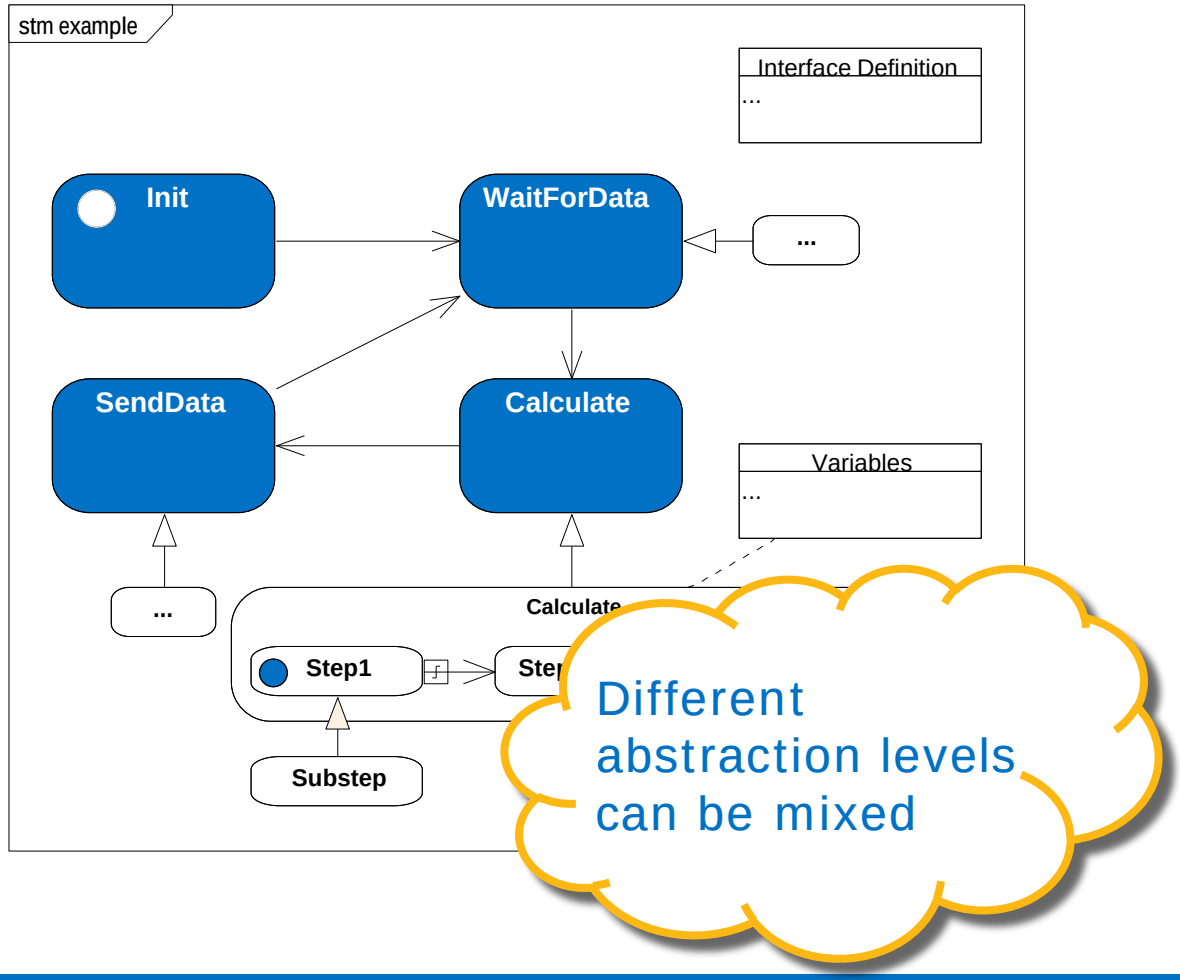
Code Generation



Single Source, Multiple Abstraction Levels

User View: UML profile

- Integrate with standard UML
- Support multiple abstraction levels for behavior and interface
- Allow consistent modeling style
- Reference interface definition



Advantages

1 Model Reuse

- Don‘t throw away high-level models
- Reuse and refine

2 Consistent Changes

- Minimize need to modify several models
- Make corresponding parts visible

3 Mix and Match

- Different applications have different requirements
- Allow mixing different abstraction levels in a single model

4 Verification Support

- Ease verification of refinement
- Localize errors easier and earlier

Agenda

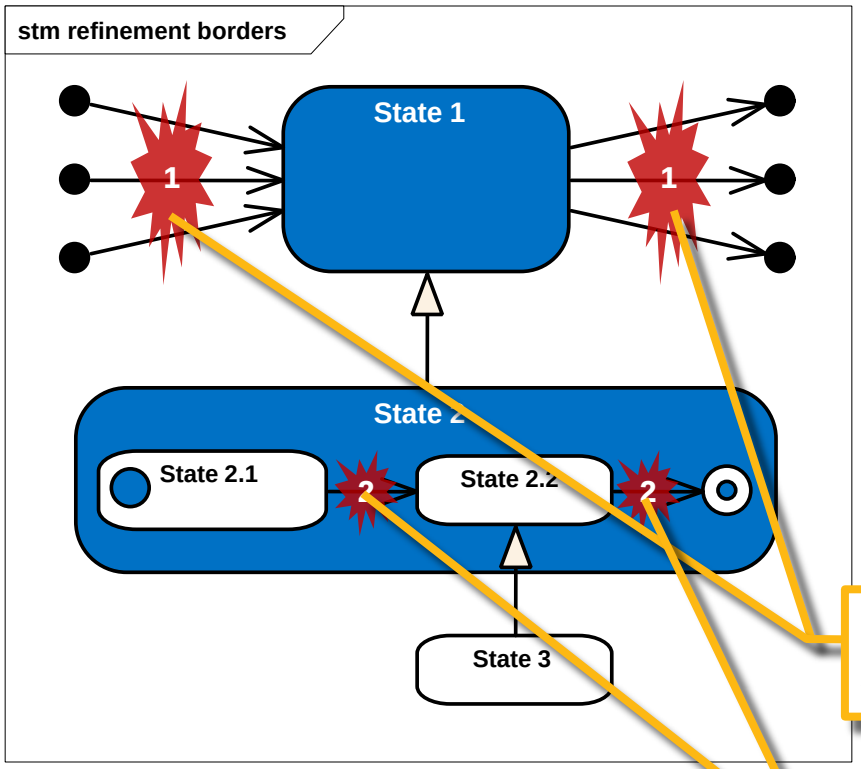
1. Single Source Modeling and Code Generation
2. **Ensuring Consistency**
3. Results
4. Summary

Consistency under Refinement

- Approach allows arbitrary code in the State Chart
  - Rules out existing static consistency approaches

- Cosimulation to check two abstraction levels:
  - Compare refinement to original
  - Use scoreboards to keep track of consistency
  - Currently no automation of testbench

Consistency under Refinement: States



State Refinement

- Liskov Substitution Principle
- Undecidable in general
- Cosimulation based on refinement borders
- Automatic scoreboard generation allows tests during cosimulation

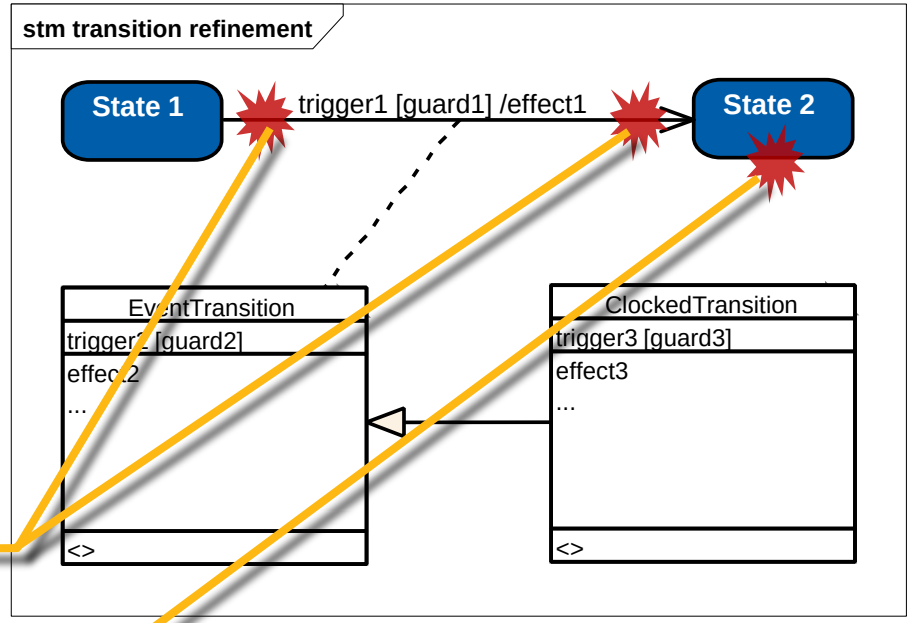
Scoreboard for refining state 1 by state 2

Scoreboard for refining state 2.2 by state 3

Consistency under Refinement: Transitions

Transition Refinement

- Behavior: Liskov Substitution Principle
- State trajectory: additional scoreboard required



Scoreboard for effect

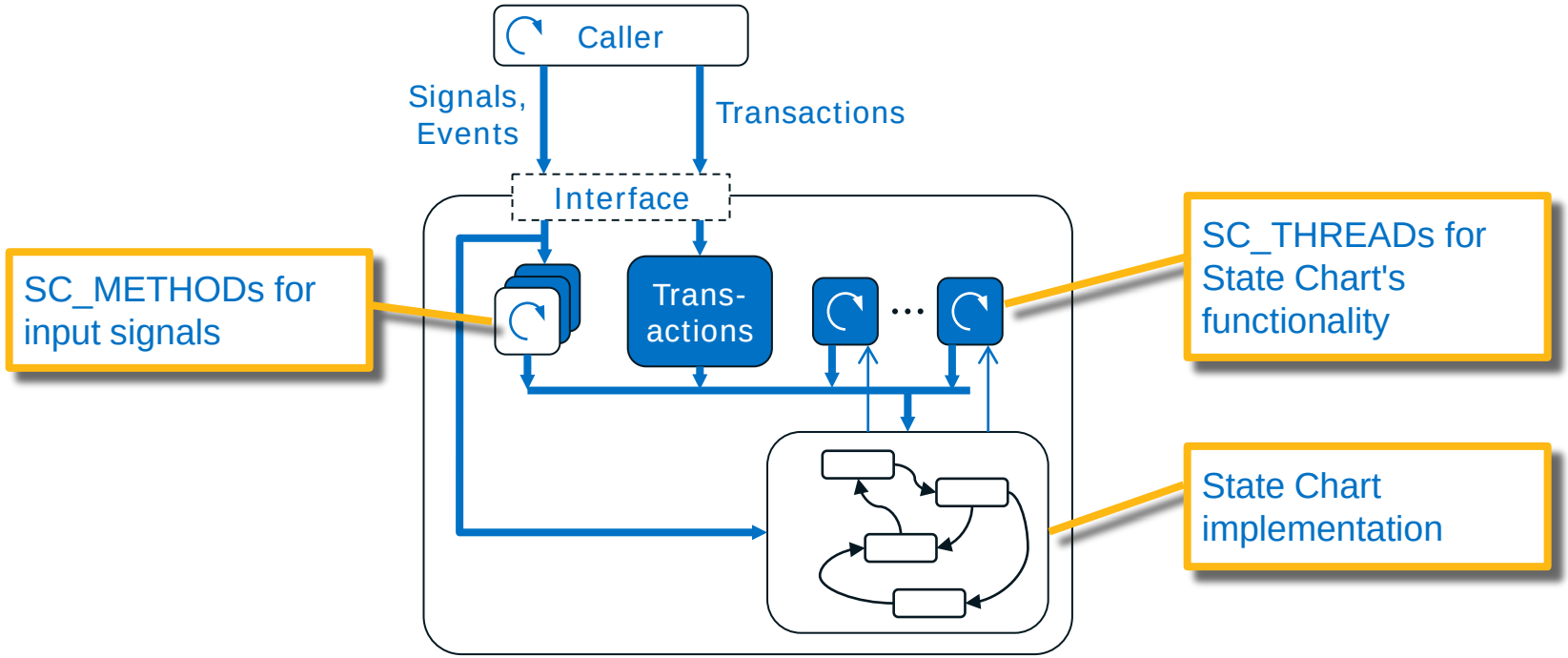
Scoreboard for state trajectory

Agenda

1. Single Source Modeling and Code Generation
2. Ensuring Consistency
3. **Results**
4. Summary

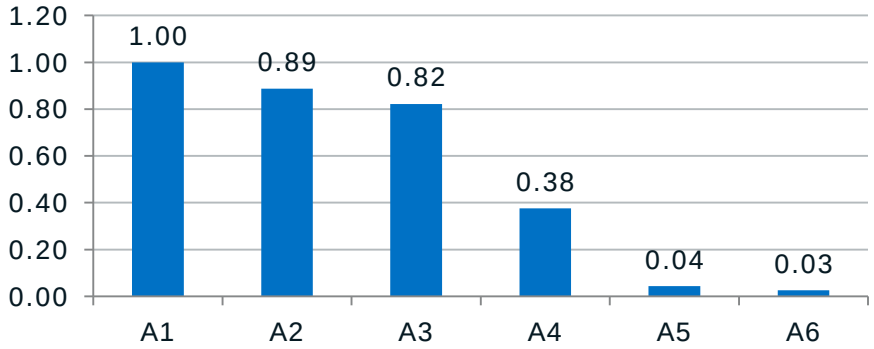
Code Generation: Module Structure

- Module is generated automatically
- Only transactions are defined externally (by deriving)
- Blue elements: Not used in cycle callable model

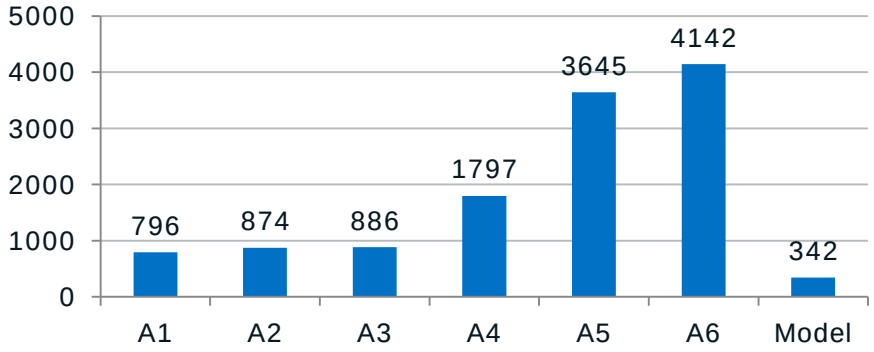


Results: AES Encryption Core

- Performance, normalized to highest



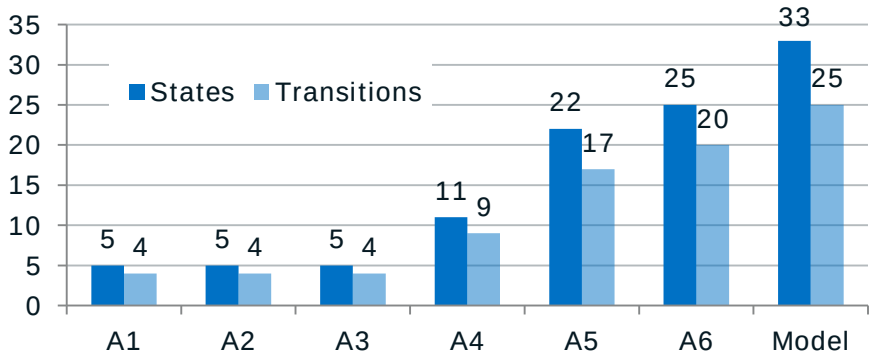
- Generated lines of code for model



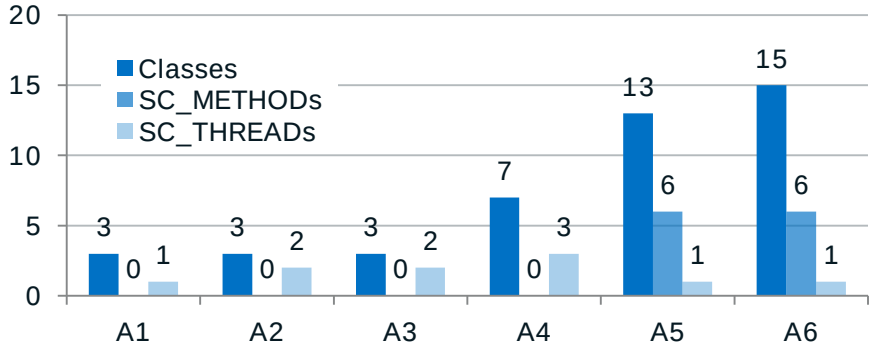
Abstraction Levels
A1 OpenSSL (3rd party library)
A2 A1 with timing
A3 Software model, timed
A4 A3, AT
A5 partly CC
A6 CC
A1-4 Transaction IF
A5-6 Pin-true IF

Results: AES Encryption Core

- Elements on the different abstraction levels



- Structure of the generated code



Abstraction Levels
A1 OpenSSL (3rd party library)
A2 A1 with timing
A3 Software model, timed
A4 A3, AT
A5 partly CC
A6 CC
A1-4 Transaction IF
A5-6 Pin-true IF

Results: AES Encryption Core

- Number of scoreboards generated for checking consistency in cosimulation

Refined Model	Original Model: A1 S	A1 T	A1 R	A1 Σ	A2 S	A2 T	A2 R	A2 Σ	A3 S	A3 T	A3 R	A3 Σ	A4 S	A4 T	A4 R	A4 Σ
A2	1	0	0	*1*	-	-	-	-	-	-	-	-	-	-	-	-
A3	1	0	0	*1*	1	0	0	*1*	-	-	-	-	-	-	-	-
A4	1	2	1	*4*	1	2	1	*4*	3	3	2	*8*	-	-	-	-
A5	1	2	1	*4*	1	2	1	*4*	3	3	2	*8*	1	0	0	*1*

S, T, R, Σ: # of Scoreboards for States, Transitions, Regions, and total

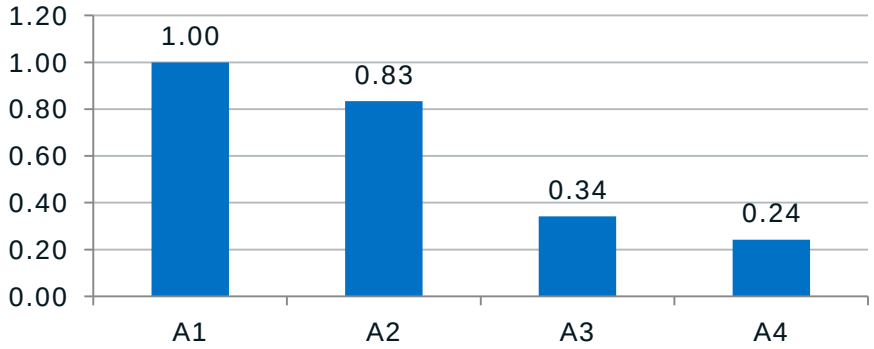
Abstraction Levels

A1 OpenSSL (3rd party library)
A2 A1 with timing
A3 Software model, timed
A4 A3, AT
A5 partly CC
A6 CC

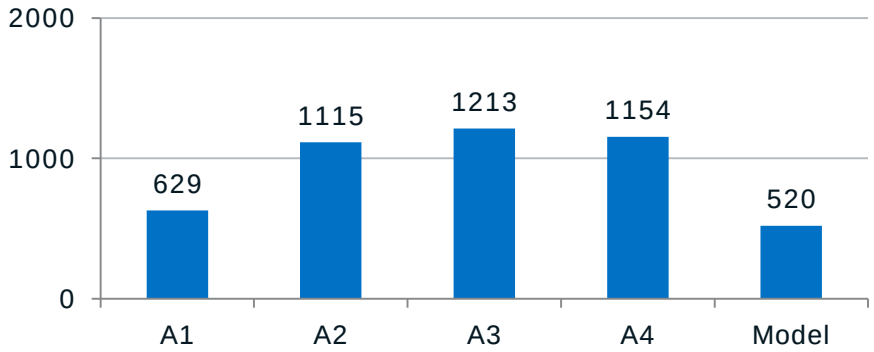
A1-4 Transaction IF
A5-6 Pin-true IF

Results: MIPS Core

- Performance, normalized to highest



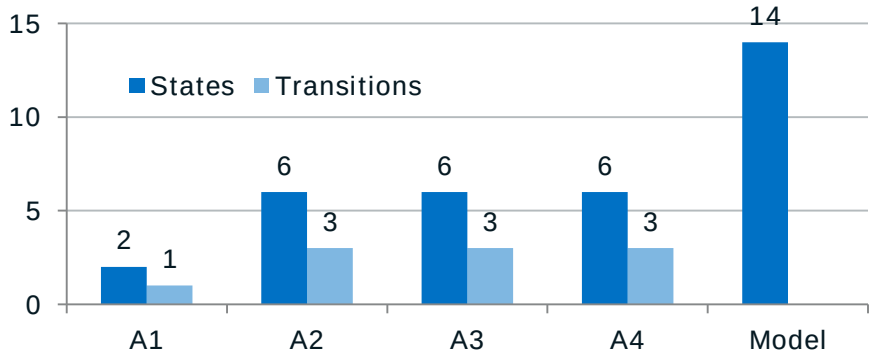
- Generated lines of code for model



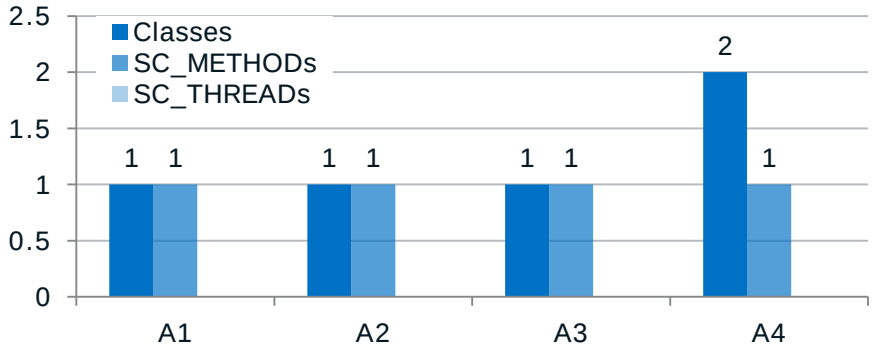
Abstraction Levels
A1 Untimed
A2 Untimed with hierarchy
A3 AT
A4 CC
A1-3 Transaction IF
A4 Pin-true IF

Results: MIPS Core

- Elements on the different abstraction levels



- Structure of the generated code



Abstraction Levels
A1 Untimed
A2 Untimed with hierarchy
A3 AT
A4 CC
A1-3 Transaction IF
A4 Pin-true IF

Results: MIPS Core

- Number of scoreboards generated for checking consistency in cosimulation

Refined Model	Original Model: A1 S	A1 T	A1 R	A1 Σ	A2 S	A2 T	A2 R	A2 Σ	A3 S	A3 T	A3 R	A3 Σ
A2	1	0	0	*1*	-	-	-	-	-	-	-	-
A3	1	0	0	*1*	0	4	1	*5*	-	-	-	-
A4	1	0	0	*1*	0	4	1	*5*	0	4	1	*5*

S, T, R, Σ: # of Scoreboards for States, Transitions, Regions, and total

Abstraction Levels

A1 Untimed

A2 Untimed with hierarchy

A3 AT

A4 CC

A1-3 Transaction IF

A4 Pin-true IF

Agenda

1. Single Source Modeling and Code Generation
2. Ensuring Consistency
3. Results
4. **Summary**

Summary

- We presented
  - a single-source modeling approach
  - based on UML for behavior specification
  - and a code generation framework that allows
  - generating a required abstraction level or
  - generating a cosimulation framework for
  - consistency checking.

- Further work:
  - More static code checking
  - Model optimizations to further improve performance


DMCE
Danube Mobile Communications Engineering
majority owned by Intel Mobile Communications

Thank you!