

Advances in Parallel Discrete Event Simulation for Embedded Computer Systems

Rainer Dömer, Professor EECS

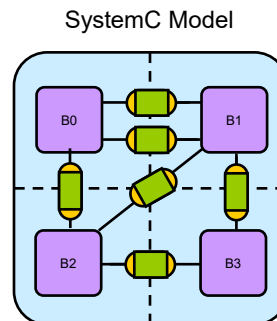
doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

1

Embedded System Design

- Model Based System Design
 - Abstract description of a complete system
 - Hardware + Software
- Key Concepts in System Modeling
 - Explicit Structure
 - Block diagram structure
 - Connectivity through ports
 - Explicit Hierarchy
 - System composed of components
 - Explicit Concurrency
 - Potential for parallel execution
 - Potential for pipelined execution
 - Explicit Communication and Computation
 - Modules
 - Channels and Interfaces



PDES for Embedded Systems, Chapman University, 4/25/23

(c) 2023 R. Doemer, UC Irvine 2

2

Embedded System Design

- **Model Validation through Simulation!**
 - Efficient system-level simulation is critical
 - Fast, and
 - Accurate!
 - Complexity of system models grows constantly
 - Need for speed!
- **Parallel Simulation!**
 - Parallelism explicitly specified in model
 - System-level Description Language (SLDL)
 - SystemC [Groetker et. al, 2002]: `SC_THREAD`, `SC_METHOD`
 - SpecC [Gajski et. al, 2000]: `par { }`, `pipe { }`
 - Parallel processing available in standard PCs
 - Multi-core host PCs readily available
 - Many-core technology is arriving

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 3

3

Related Work: Faster Simulation

Improved Modeling Techniques

- Transaction-level modeling (TLM).
- TLM temporal decoupling.
- Savoiu et al. [MEMOCODE'05]
- Razaghi et al.[ASPDAC'12]

Distributed Simulation

- Chandy et al. [TSE'79]
- Huang et al. [SIES'08]
- Chen et al. [CECS'11]

↑

Traditional System Simulation is slow

↓

Hardware-based Acceleration

- Sirowy et al. [DAC'10]
- Nanjundappa et al. [ASPDAC'10]
- Sinha et al. [ASPDAC'12]

SMP Parallel Simulation

- Fujimoto [CACM'90]
- Chopard et al. [ICCS'06]
- Ezudheen et al. [PADS'09]
- Mello et al. [DATE'10]
- Schumacher et al. [CODES'11]
- Chen et al. [TCAD'14]
- Yun et al. [TCAD'12]
- Schmidt et al. [DAC'17]
- and many others

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 4

4

Project with Intel: Key Points

- Advanced Parallel SystemC Simulation
 - Out-of-Order PDES on many-core host platforms
 - Maximum compliance with current execution semantics
 - Support for parallel execution of virtual platforms
- Introduction of a Dedicated SystemC Compiler
 - Recoding Infrastructure for SystemC (RISC)
 - Advanced static analysis for parallel execution
 - Model instrumentation and code generation
- Parallel SystemC Core Library
 - Out-of-order parallel scheduler, multi-thread safe primitives
 - Many-core target platform (e.g. Intel® Xeon Phi™)
- Open Source
 - Collaboration with Accellera SystemC Language WG

PDES for Embedded Systems, Chapman University, 4/25/23

(c) 2023 R. Doemer, UC Irvine 5

5

Discrete Event Simulation

- Traditional Discrete Event Simulation (DES)
 - Reference simulators run *sequentially*, only one thread at a time (cooperative multi-threading model)
 - Cannot utilize the capabilities of multi- or many-core hosts
- Parallel Discrete Event Simulation (PDES)
 - Threads run in *parallel* (if at the same delta cycle and time)
 - Simulation-cycles are absolute barriers!
- Out-of-order Parallel DE Simulation (OoO PDES)
 - Threads run in *parallel and out-of-order* [DATE'12, TCAD'14] even in different delta and time cycles if there are no conflicts!
 - Aggressive, runs maximum number of threads in parallel, but *fully preserves DES semantics and model accuracy!*

PDES for Embedded Systems, Chapman University, 4/25/23

(c) 2023 R. Doemer, UC Irvine 6

6

Discrete Event Simulation (DES)

- Traditional Sequential Simulation
 - Concurrent threads of execution
 - Managed by a central scheduler
 - Driven by events and time advances
 - Delta cycle
 - Time cycle
 - Partial temporal order with barriers
- IEEE 1666 Standard Simulator
 - SystemC reference simulator uses cooperative multi-threading
 - A single thread is active at any time!
 - Cannot exploit parallelism
 - Cannot utilize multiple cores
 - Sequential simulation is slow

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 7

7

Parallel Discrete Event Simulation (PDES)

- Parallel DES [Fujimoto1990]
 - Threads execute in parallel *iff*
 - in the same delta cycle, *and*
 - in the same time cycle
 - Significant speed up!
 - *Synchronous* PDES: Cycle boundaries are *absolute barriers!*
- Aggressive Parallel DES
 - Conservative Approaches
 - Careful static analysis prevents conflicts
 - Optimistic Approaches
 - Conflicts are detected and addressed (*roll back*)

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 8

8

Parallel Discrete Event Simulation (PDES)

- Out-of-Order PDES
 - Threads execute in parallel *iff*
 - in the same delta cycle, *and*
 - in the same time cycle,
 - **OR if there are no conflicts!**
 - Breaks synchronization barrier!
 - Threads run as soon as possible, even ahead of time
 - Significantly higher speedup!
 - Results at [DATE'12], [IEEE TCAD'14]
 - Advanced compiler fully preserves...
 - DES execution semantics
 - Accuracy in results and timing

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 9

9

Recoding Infrastructure for SystemC (RISC)

- Advanced Parallel SystemC Simulation
 - Aggressive PDES on many-core host platforms
 - Maximum compliance with IEEE SystemC semantics
- Introduction of a Dedicated SystemC Compiler
 - Advanced conflict analysis for safe parallel execution
 - Automatic model instrumentation and code generation
- Parallel SystemC Simulator
 - Out-of-order parallel scheduler, multi-thread safe primitives
 - Multi- and many-core host platforms (e.g. Intel® Xeon Phi™)
- Open Source
 - Freely available for evaluation and collaboration
 - Thanks to Intel Corporation!

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 10

10

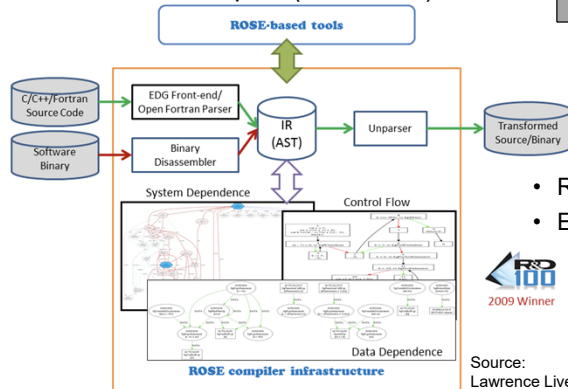
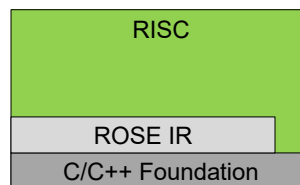
Recoding Infrastructure for SystemC (RISC)

- Out-of-Order PDES Key Ideas
 1. Dedicated *SystemC compiler* with advanced model analysis
 - Static conflict analysis based on Segment Graphs
 2. *Parallel simulator* with out-of-order scheduling on many cores
 - Fast decision making at run-time, optimized mapping
- Fundamental Data Structure: *Segment Graph*
 - Key to semantics-compliant out-of-order execution [DATE'12]
 - Key to prediction of future thread state [DATE'13]
 - “Optimized Out-of-Order Parallel DE Simulation Using Predictions”
 - Key to May-Happen-in-Parallel Analysis [DATE'14]
 - “May-Happen-in-Parallel Analysis based on Segment Graphs for Safe ESL Models” (**Best Paper Award**)
 - Journal article: “OoO PDES for TLM” [IEEE TCAD'14]
 - Comprehensive summary with HybridThreads extension

11

RISC: Dedicated SystemC Compiler

- RISC Software Stack
 - *Recoding Infrastructure for SystemC*
 - C/C++ foundation
 - ROSE compiler (from LLNL)



- ROSE Internal Representation
- Explicit support for
 - Source code analysis
 - Source-to-source transformations

Source: Lawrence Livermore National Laboratory (LLNL)

12

RISC: Dedicated SystemC Compiler

- RISC Software Stack
 - *Recoding Infrastructure for SystemC*
 - SystemC Internal Representation
- Class hierarchy to represent SystemC objects

```

class Definition {
public:
    +type_pointer_ : SType
    +ast_pointer_ : ASTNode
    +get_name() : std::string
    +get_type_name() : std::string
    +get_ast_node() : SNode
    +get_type() : SType
    +get_file_name() : std::string
    +get_line_numbers() : int
    +get_position_in_line() : int
    +has_source_location() : bool
};
                
```

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
13

13

RISC: Dedicated SystemC Compiler

- RISC Software Stack
 - *Recoding Infrastructure for SystemC*
 - 1) Segment Graph
 - 2) Parallel access conflict analysis

Step 1: Build a Segment Graph

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
14

14

RISC: Dedicated SystemC Compiler

- Segment Graph
 - Segment Graph is a directed graph
 - Nodes: Segments
 - Code statements executed between two scheduling steps
 - Expression statements
 - Control flow statements (if, while, ...)
 - Function calls
 - Edges: Segment boundaries
 - Primitives that trigger scheduler entry
 - wait(event)
 - wait(time)
 - Segment Graph is built automatically by the compiler [TCAD'14]
 - From the model source code
 - Via Abstract Syntax Tree and Control Flow Graph

Segment Graph

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
15

15

RISC: Dedicated SystemC Compiler

- RISC Software Stack
 - Recoding Infrastructure for SystemC
 - 1) Segment Graph construction
 - 2) Parallel access conflict analysis
 - 3) Model instrumentation

Segment Graph

Conflict	Seg 1	Seg 2	Seg 3
Seg 1	True	False	False
Seg 2	False	True	True
Seg 3	False	True	True

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
16

16

RISC: Compiler and Simulator

- Compiler and Simulator work hand in hand
 - Compiler performs conservative static analysis
 - Analysis results are passed to the simulator
 - Simulator can make safe scheduling decisions quickly
- Automatic Model Instrumentation
 - Static analysis results are inserted into the source code

Model Instrumentation:
 Segment and Instance IDs
 Segment Conflict Tables
 Time Advance Tables

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 17

17

RISC: Parallel SystemC Simulator

- Simulator kernel with Out-of-Order Parallel Scheduler
 - Conceptual OoO PDES execution

Issue threads...

- truly in *parallel* and *out-of-order*
- whenever they are *ready*
- and have *no conflicts!*
 - Fast conflict table lookup
 - Optimized thread-to-core mapping

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 18

18

RISC: Experiments and Results

- DVD Player Example
 - Parallel video and audio decoding with different frame rates

```

1: SC_MODULE(VideoCodec)
2: { sc_port<_receiver> p1;
3:   sc_port<_sender> p2;
4:   ...
5:   while(1) {
6:     p1->receive(&inFrm);
7:     outFrm = decode(inFrm);
8:     wait(33330, SC_US);
9:     p2->send(outFrm);
10:  }
11: };
                
```

```

1: SC_MODULE(AudioCodec)
2: { sc_port<_receiver> p1;
3:   sc_port<_sender> p2;
4:   ...
5:   while(1) {
6:     p1->receive(&inFrm);
7:     outFrm = decode(inFrm);
8:     wait(26120, SC_US);
9:     p2->send(outFrm);
10:  }
11: };
                
```

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
19

19

RISC: Experiments and Results

- DVD Player Example
 - Parallel video and audio decoding with different frame rates

- Real time schedule: fully parallel

- Reference simulator schedule (DES)

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
20

20

RISC: Experiments and Results

- DVD Player Example
 - Parallel video and audio decoding with different frame rates
 - 1. Real time schedule: fully parallel

Time [ms]: 0, 26.12, 52.25, 78.38, 100

- 3. Synchronous parallel schedule (PDES)

Time [ms]: 0, 26.12, 52.25, 78.38, 100

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
21

21

RISC: Experiments and Results

- DVD Player Example
 - Parallel video and audio decoding with different frame rates
 - 1. Real time schedule: fully parallel

Time [ms]: 0, 26.12, 52.25, 78.38, 100

- 4. Out-of-order parallel schedule (OoO PDES)

Time [ms]: 0, 26.12, 52.25, 78.38, 100

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
22

22

RISC: Experiments and Results

- DVD Player Example
 - Parallel video and audio decoding with different frame rates
- Simulator Run Times
 - 4-core Intel® Xeon® CPU at 3.4 GHz
 - RISC v0.2.1, Posix-threads

		DES	PDES	OoO PDES
10 sec stream	Run Time	6.98 s	4.67 s	2.94 s
	CPU Load	97%	145%	238%
	Speedup	1 x	1.49 x	2.37 x
100 sec stream	Run Time	68.21 s	45.91 s	28.13 s
	CPU Load	100%	149%	251%
	Speedup	1 x	1.49 x	2.42 x

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
23

23

RISC: Experiments and Results

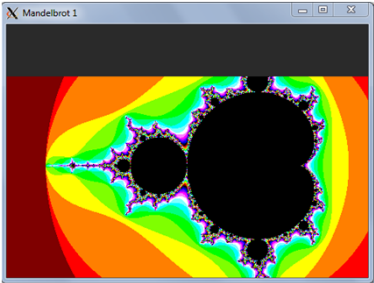
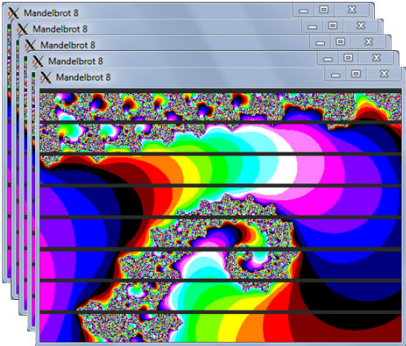
- Mandelbrot Renderer (Graphics Pipeline Application)
 - Mandelbrot Set
 - Mathematical set of points in complex plane
 - Two-dimensional fractal shape
 - High computation load
 - Recursive/iterative function
 - Embarassingly parallel
 - Parallelism at pixel level
 - SystemC Model
 - TLM abstraction
 - Horizontal image slices
 - Highly configurable
 - Parallelism parameter from 1 to 256 slices

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
24

24

RISC: Experiments and Results

- Mandelbrot Renderer (Graphics Pipeline Application)
 - *Simulated Graphics Demonstration*
(when network delays prevent actual graphical demo)

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
25

25

RISC: Experiments and Results

- Mandelbrot Renderer (Graphics Pipeline Application)
 - Simulator run times on 16-core Intel® Xeon® multi-core host
 - 2 CPUs at 2.7 GHz, 8 cores each, 2-way hyper-threaded
 - RISC V0.2.1, Posix-threads

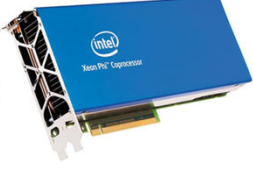
Parallel Slices	DES		PDES			OOO PDES		
	Run Time	CPU Load	Run Time	CPU Load	Speedup	Run Time	CPU Load	Speedup
1	162.13 s	99%	162.06 s	100%	1.00 x	161.90 s	100%	1.00 x
2	162.19 s	99%	96.50 s	168%	1.68 x	96.48 s	168%	1.68 x
4	162.56 s	99%	54.00 s	305%	3.01 x	53.85 s	304%	3.02 x
8	163.10 s	99%	29.89 s	592%	5.46 x	30.05 s	589%	5.43 x
16	164.01 s	99%	19.03 s	1050%	8.62 x	20.08 s	997%	8.17 x
32	165.89 s	99%	11.78 s	2082%	14.08 x	11.99 s	2023%	13.84 x
64	170.32 s	99%	9.79 s	2607%	17.40 x	9.85 s	2608%	17.29 x
128	174.55 s	99%	9.34 s	2793%	18.69 x	9.39 s	2787%	18.59 x
256	185.47 s	100%	8.91 s	2958%	20.82 x	8.90 s	2964%	20.84 x

PDES for Embedded Systems, Chapman University, 4/25/23
(c) 2023 R. Doemer, UC Irvine
26

26

RISC: Experiments and Results

- Many-Core Target Platform: Intel® Xeon Phi™
 - Many Integrated Core (MIC) architecture
 - 1 Coprocessor 5110P CPU at 1.052 GHz
 - 60 physical cores with 4-way hyper-threading
 - Appears as regular Linux host with 240 cores
 - Up to 8 lanes available for vector processing



- RISC extended for exploiting 2 types of parallelism
 - Out-of-Order PDES: *thread-level* parallelism
 - Intel® compiler SIMD: *data-level* parallelism
- RISC SIMD Advisor identifies functions with data-level parallelism suitable for SIMD vectorization
- DAC '17 paper:
 - "Exploiting Thread and Data Level Parallelism for Ultimate Parallel SystemC Simulation"*

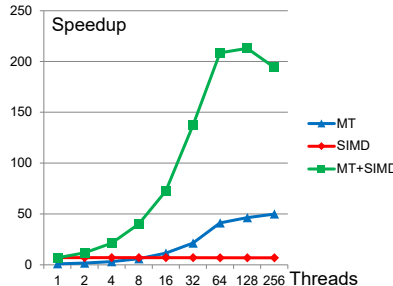
PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 27

27

RISC: Experiments and Results

- Many-Core Target Platform: Intel® Xeon Phi™
 - Exploiting thread- and data-level parallelism [DAC'17]
 - Mandelbrot renderer (graphics pipeline application)
- Experimental Results:

PAR	MT	SIMD	MT+SIMD
1	1.00	6.92	6.94
2	1.68	6.92	11.77
4	3.04	6.92	21.19
8	5.84	6.92	40.10
16	11.37	6.92	72.52
32	21.32	6.91	137.21
64	41.07	6.90	208.41
128	46.29	6.89	212.96
256	49.90	6.87	194.19


 - Increasing degree of parallelism (PAR = number of threads) reaches a combined multi-threading (MT) and data-level (SIMD) speedup of **up to 212x!**

PDES for Embedded Systems, Chapman University, 4/25/23 (c) 2023 R. Doemer, UC Irvine 28

28

RISC: Open Source Software

- RISC Compiler and Simulator are freely available
 - <http://www.cecs.uci.edu/~doemer/risc.html#RISC062>
 - Installation notes and script: **INSTALL, Makefile**
 - Open source tar ball: **risc_v0.6.3.tar.gz**
 - Docker script and container: **Dockerfile**
 - Doxygen documentation: **RISC API, OOPSC API**
 - Tool manual pages: **risc, simd, visual, ...**
 - BSD license terms: **LICENSE**
 - Companion Technical Report
 - CECS Technical Report 19-04: **CECS_TR_19_04.pdf**

```
bash# docker pull ucirvinelecs/risc063
bash# docker run -it ucirvinelecs/risc063
[dockeruser]# cd demodir
[dockeruser]# make test
```

- Docker container:

- <https://hub.docker.com/r/ucirvinelecs/risc063/>

29

Conclusion

- Recoding Infrastructure for SystemC (RISC)
 - Out-of-Order Parallel SystemC Simulation
 - Aggressive PDES on many-core host platforms
 - Maximum compliance with IEEE SystemC semantics
 - Introduction of a Dedicated SystemC Compiler
 - Advanced conflict analysis for safe parallel execution
 - Automatic model instrumentation and code generation
 - Parallel SystemC Simulator
 - Out-of-order parallel scheduler, multi-thread safe primitives
 - Multi- and many-core host platforms (e.g. Intel® Xeon Phi™)
 - Open Source
 - Freely available for use and collaboration (BSD license)
 - Thanks to Intel Corporation!

30

Acknowledgments

- For solid work, fruitful discussions, and honest feedback, I would like to thank:
 - My team at UCI
 - Emad Arasteh, Aditya Harit, Vivek Govindasamy
 - Zhongqi Cheng, Daniel Mendoza
 - Tim Schmidt, Guantao Liu
 - Farah Arabi, Spencer Kam
 - Our collaborators at Intel
 - Ajit Dingankar
 - Desmond Kirkpatrick
 - Abhijit Davare
 - Philipp Hartmann
 - And many others...
- This work has been supported in part by substantial funding from Intel Corporation. Thank you!

31

References (1)

- [Springer'20b] R. Dömer, Z. Cheng, D. Mendoza, E. Arasteh: "*Pushing the Limits of Parallel Discrete Event Simulation for SystemC*", in "*A Journey of Embedded and Cyber-Physical Systems*" by Jian-Jia Chen, Springer Nature, Switzerland, August 2020.
- [IJPP'20] Z. Cheng, T. Schmidt, R. Dömer: "*Scaled Static Analysis and IP Reuse for Out-of-Order Parallel SystemC Simulation*", International Journal of Parallel Programming (IJPP), Springer, June 2020.
- [DATE'20] D. Mendoza, Z. Cheng, E. Arasteh, R. Dömer: "*Lazy Event Prediction using Defining Trees and Schedule Bypass for Out-of-Order PDES*", Proceedings of DATE, Grenoble, France, March 2020.
- [ASPDAC'20] Z. Cheng, A. Arasteh, R. Dömer: "*Event Delivery using Prediction for Faster Parallel SystemC Simulation*", Proceedings of ASPDAC, Beijing, China, Jan. 2020.
- [Springer'20a] Z. Cheng, T. Schmidt, R. Dömer: "*SystemC Coding Guideline for Faster Out-of-Order Parallel Discrete Event Simulation*", chapter 6 in "*Languages, Design Methods, and Tools for Electronic System Design*" by T. Kazmierski, S. Steinhorst and D. Grosse, reprint of best papers at FDL 2018, Springer Nature, Switzerland, January 2020.
- [DVCon'19] D. Mendoza, A. Dingankar, Z. Cheng, R. Dömer: "*Integrating Parallel SystemC Simulation into Simics® Virtual Platform*", Proceedings of DVCon Europe, Munich, Germany, Oct. 2019.
- [TECS'19] Z. Cheng, R. Dömer: "*Analyzing Variable Entanglement for Parallel Simulation of SystemC TLM-2.0 Models*", ACM Transactions on Embedded Computing Systems (TECS), vol. 18, no. 5s, article 79, 20 pages, October 2019.

32

References (2)

- [CECS'19] G. Liu, T. Schmidt, Z. Cheng, D. Mendoza, R. Dömer: *"RISC Compiler and Simulator, Release V0.6.0: Out-of-Order Parallel Simulatable SystemC Subset"*, CECS TR 19-04, Sep. 2019.
- [IESS'19b] E. Arasteh, R. Dömer: *"An Untimed SystemC Model of GoogLeNet"*, Proceedings of IESS, Springer, Friedrichshafen, Germany, Sep. 2019.
- [IESS'19a] Z. Cheng, T. Schmidt, R. Dömer: *"Enabling IP Reuse and Protection in Out-of-Order Parallel SystemC Simulation"*, Proceedings of IESS, Springer, Friedrichshafen, Germany, Sep. 2019.
- [FDL'18] Z. Cheng, T. Schmidt, R. Dömer: *"SystemC Coding Guideline for Faster Out-of-Order Parallel Discrete Event Simulation"*, Proceedings of FDL, Munich, Germany, Sep. 2018.
- [DATE'18] T. Schmidt, Z. Cheng, R. Dömer: *"Port Call Path Sensitive Conflict Analysis for Instance-Aware Parallel SystemC Simulation"*, Proceedings of DATE, Dresden, Germany, March 2018.
- [CECS'17] D. Mendoza, R. Dömer: *"A Tool for Visualization of SystemC Models"*, CECS Technical Report 17-06, Nov. 2017.
- [HLDVT'17] Z. Cheng, T. Schmidt, G. Liu, R. Dömer: *"Thread- and Data-Level Parallel Simulation in SystemC, a Bitcoin Miner Case Study"*, Proceedings of HLDVT, Santa Cruz, California, Oct. 2017.
- [DAC'17] T. Schmidt, G. Liu, R. Dömer: *"Towards Ultimate Parallel SystemC Simulation through Thread and Data Level Parallelism"*, Proceedings DAC, Austin, TX, June 2017.

33

References (3)

- [Springer'17] R. Dömer, G. Liu, T. Schmidt: *"Parallel Simulation"*, chapter 17 in *"Handbook of Hardware/Software Codesign"* by S. Ha and J. Teich, Springer Netherlands, June 2016.
- [ASPDAC'17] T. Schmidt, G. Liu, R. Dömer: *"Hybrid Analysis of SystemC Models for Fast and Accurate Parallel Simulation"*, Proceedings ASPDAC, Tokyo, Japan, January 2017.
- [IEEE ESL'16] R. Dömer: *"Seven Obstacles in the Way of Standard-Compliant Parallel SystemC Simulation"*, IEEE Embedded Systems Letters, vol. 8, no. 4, pp. 81-84, Dec. 2016.
- [DAC'15] R. Dömer: *"Towards Parallel Simulation of Multi-Domain System Models"*, Keynote, DAC workshop on System-to-Silicon Performance Modeling and Analysis, June 2015.
- [IEEE TCAD'14] W. Chen, X. Han, C. Chang, G. Liu, R. Dömer: *"Out-of-Order Parallel Discrete Event Simulation for Transaction Level Models"*, IEEE Transactions on CAD, vol. 33, no. 12, pp. 1859-1872, December 2014.
- [DATE'14] W. Chen, X. Han, R. Dömer: *"May-Happen-in-Parallel Analysis based on Segment Graphs for Safe ESL Models"*, Proceedings of DATE, Dresden, Germany, March 2014.
- [DATE'13] W. Chen, R. Dömer: *"Optimized Out-of-Order Parallel Discrete Event Simulation Using Predictions"*, Proceedings of DATE, Grenoble, France, March 2013.
- [DATE'12] W. Chen, X. Han, R. Dömer: *"Out-of-Order Parallel Simulation for ESL Design"*, Proceedings of DATE, Dresden, Germany, March 2012.

34

RISC Open Source Releases

- [RISCv063] R. Dömer, V. Govindasamy, Y. Wang.
RISC Compiler and Simulator, Release V0.6.3. August 2021.
– <http://cecs.uci.edu/~doemer/risc.html#RISC063>
- [RISCv062] Z. Cheng, R. Dömer, A. Harit.
RISC Compiler and Simulator, Release V0.6.2. September 2020.
- [RISCv060] Z. Cheng, R. Dömer, S. Kam, and D. Mendoza.
RISC Compiler and Simulator, Release V0.6.0. September 2019.
- [RISCv050] Z. Cheng, R. Dömer, D. Mendoza, and T. Schmidt.
RISC Compiler and Simulator, Release V0.5.0. September 2018.
- [RISCv040] R. Dömer, G. Liu, and T. Schmidt.
RISC Compiler and Simulator, Release V0.4.0. July 2017.
- [RISCv030] R. Dömer, G. Liu, and T. Schmidt.
RISC Compiler and Simulator, Beta Release V0.3.0. September 2016.
- [RISCv020] R. Dömer, G. Liu, and T. Schmidt.
RISC Compiler and Simulator, Alpha Release V0.2.0. September 2015.
- [RISCv010] R. Dömer, G. Liu, and T. Schmidt.
RISC API, Alpha Release V0.1.0. June 2014.