

# Integrating Parallel SystemC Simulation into Simics® Virtual Platform

**Daniel Mendoza**, UC Irvine and Intel Corporation

Ajit Dingankar, Intel Corporation

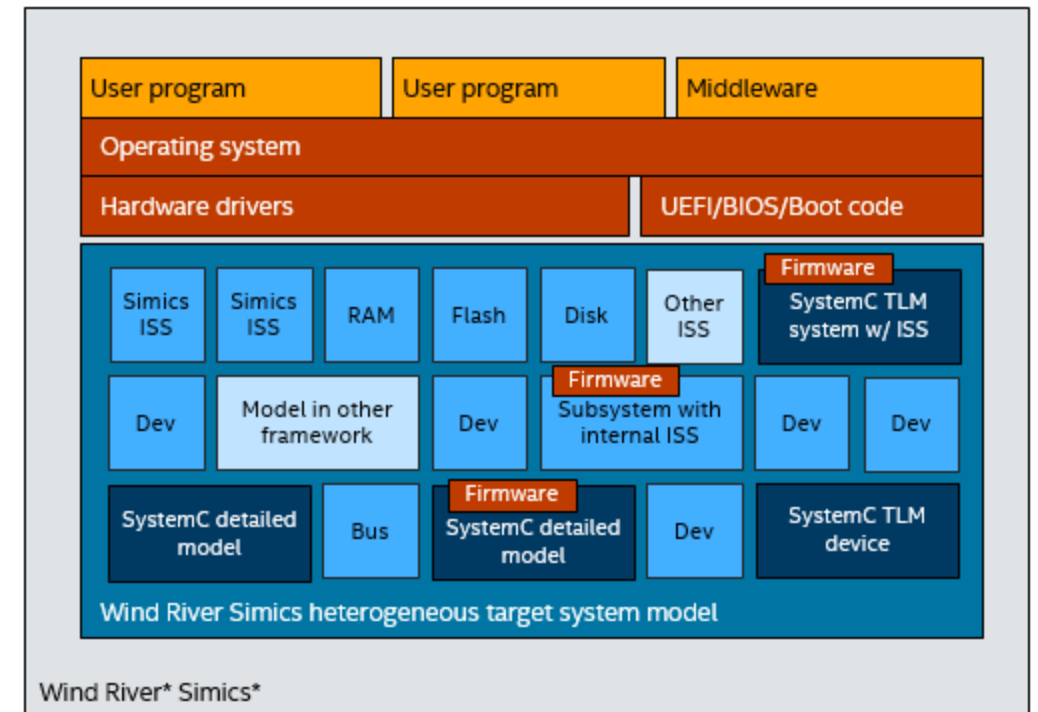
Zhongqi Cheng and Rainer Doemer, UC Irvine

# Overview

- What is Simics?
- Standard SystemC Simulation vs. Out-of-Order Parallel SystemC Simulation
- Parallel SystemC Integration into Simics
- Experimental Results
- Conclusions

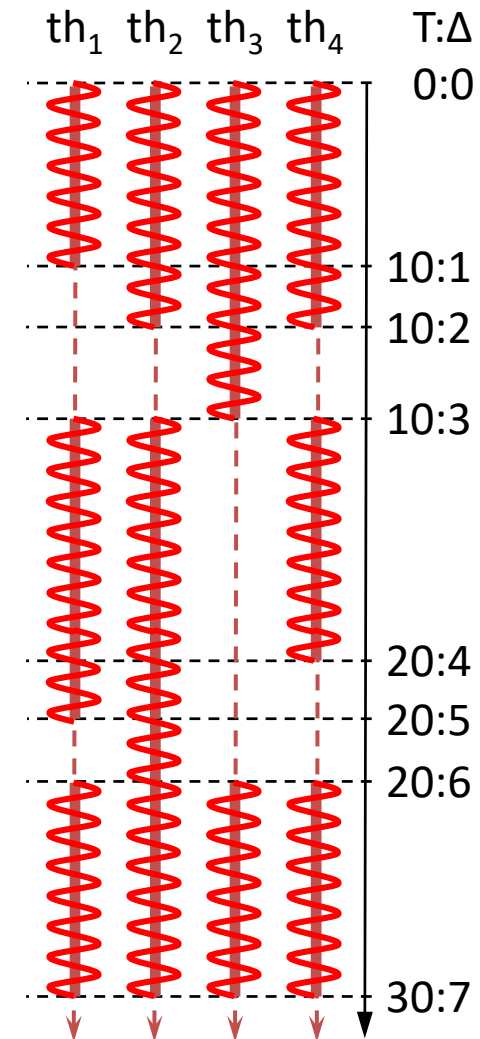
# Simics®

- Virtual Platform for modeling applications
  - Pre-silicon software development
  - Hardware validation
  - BIOS regression testing
- Supports system modeling languages:
  - C, C++
  - Python
  - DML
  - **SystemC**



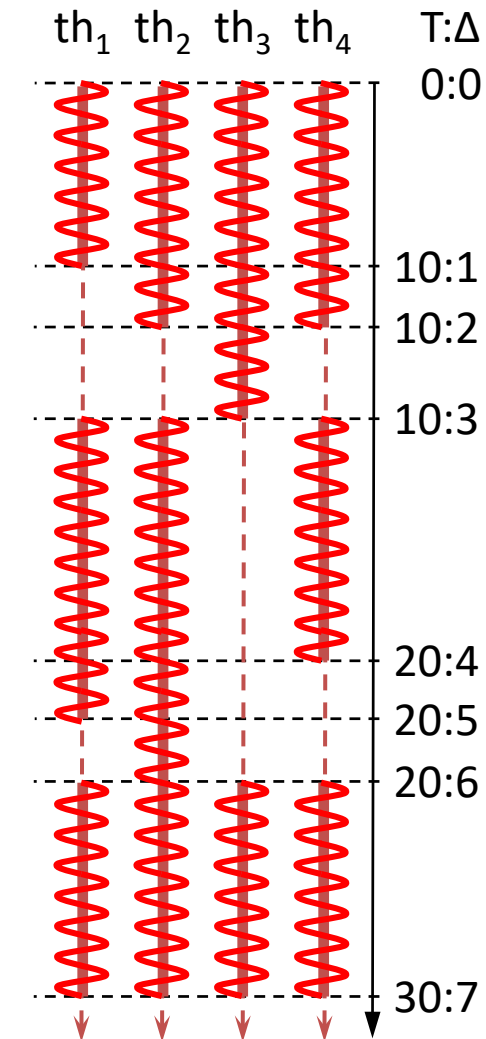
# Standard Sequential SystemC Simulation

- Reference: Accellera SystemC
  - IEEE 1666-2011 standard
  - Discrete Event Simulation (DES)



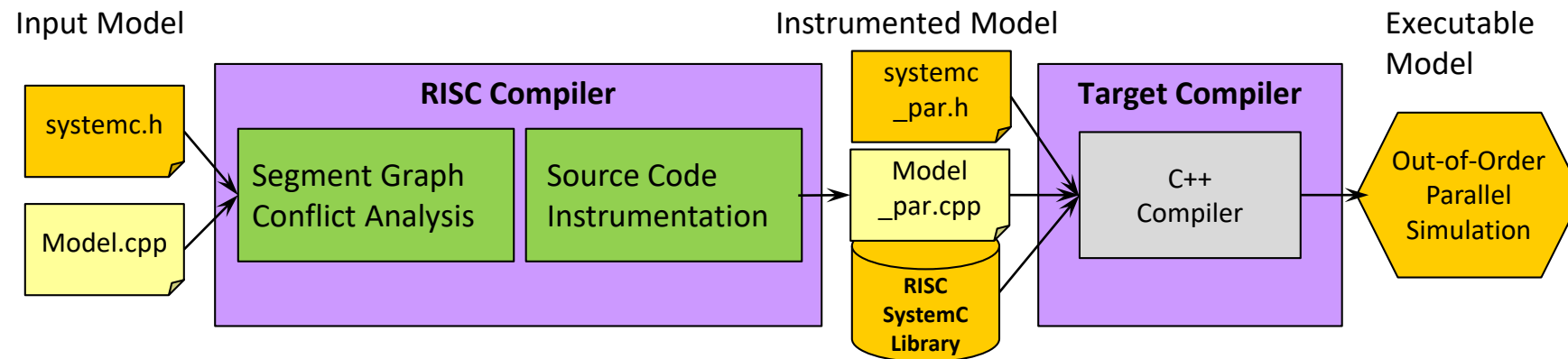
# Out-of-Order Parallel SystemC Simulation

- RISC Compiler and Simulator
  - Recoding Infrastructure for SystemC
  - Out-of-Order Parallel DES
  - Orders of magnitude speedup (200x)
  - Maximum compliance with IEEE std.
  - Open Source (sponsored by Intel Corp.)  
<http://www.cecs.uci.edu/~doemer/risc.html>



# RISC Tool Flow

- Input model automatically transformed into parallel model
  - RISC compiler analyzes data and event conflicts
  - Parallel model linked to RISC out-of-order parallel simulator



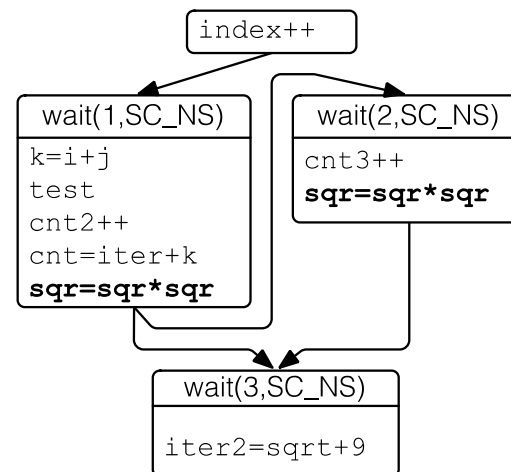
# Segment Graph

- Parallel model based upon Segment Graph data structure
  - RISC creates Segment Graph of input model
  - Conflict Tables are inferred from Segment Graph
  - Tables are passed to parallel simulator for fast scheduling decisions

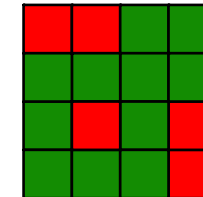
Source Code

```
0 void foo() {  
1   index++;  
2   wait(1, SC_NS);  
3   k=i+j;  
4   if(test) {  
5     cnt2++;  
6     wait(2, SC_NS);  
7     cnt3++;  
8   } else {  
9     cnt=iter+k;  
10  }  
11  sqr=sqr*sqr;  
12  wait(3, SC_NS);  
13  iter2=sqr+9; }
```

Segment Graph



Conflict Table



(Schmidt et al. 2018)

# RISC Experimental Results

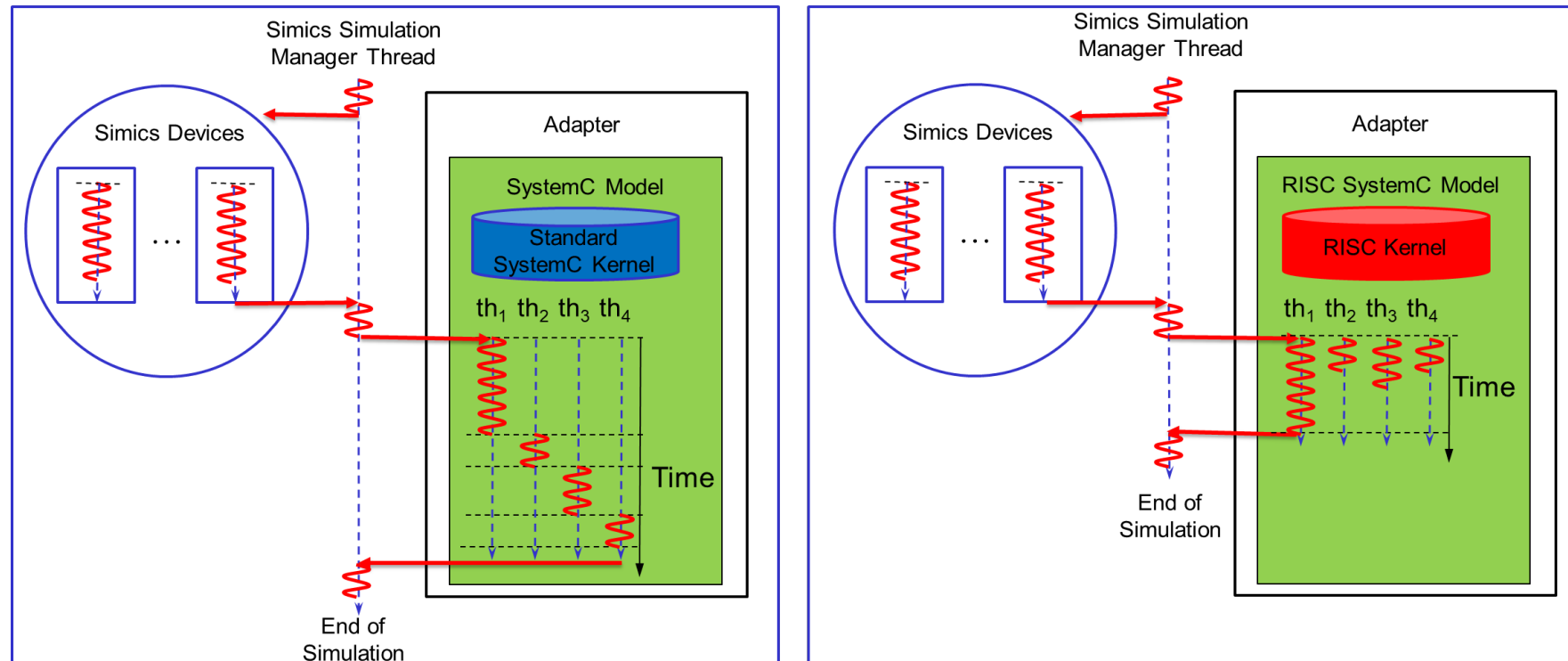
- Mandelbrot Renderer Simulation
  - Highly parallel model that generates Mandelbrot frames
  - 60 core Intel® Xeon Phi™ host
  - Thread and data level parallelism
  - Peak speedup with RISC is 212x [DAC'17]

# slices	seq.simd (M)	par (N)	par.simd (NxM)
1	6.9153	1.001	6.94462
2	6.9176	1.682	11.7748
4	6.9183	3.042	21.1943
8	6.9176	5.845	40.0967
16	6.9167	11.37	72.5175
32	6.9124	21.32	137.213
64	6.896	41.07	208.413
128	6.8948	46.29	212.957
256	6.8736	49.9	194.187



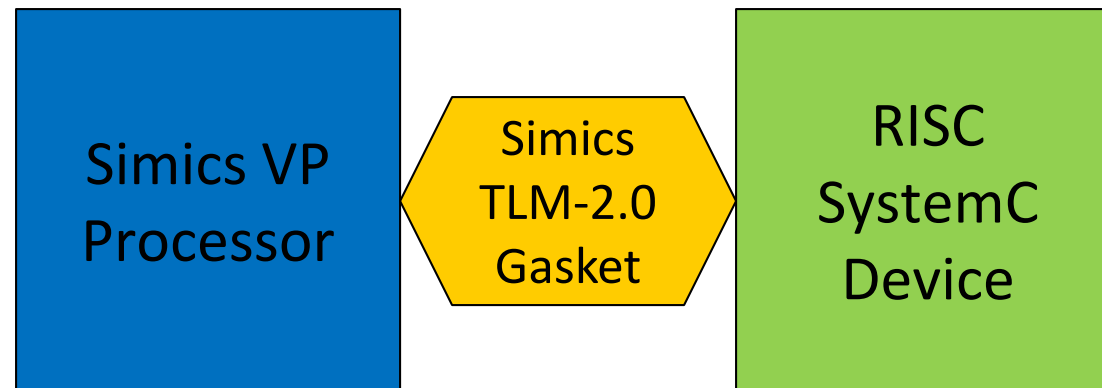
# Parallel SystemC in Simics

- Replace Standard SystemC Kernel with RISC Kernel



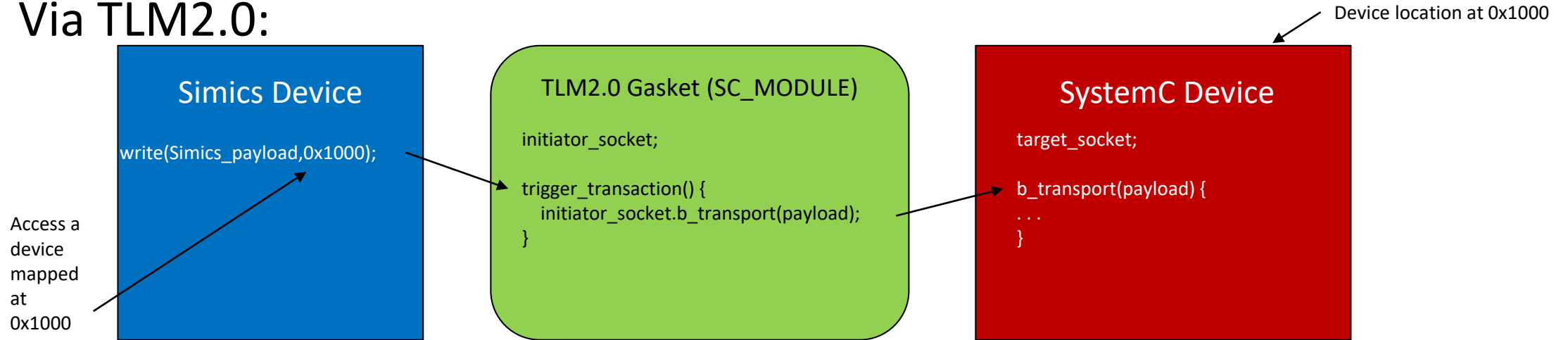
# Communication between Simics and SystemC

- Simics expects data transfers between Simics and SystemC devices to happen via TLM2.0 *gaskets*
  - Gaskets are SystemC modules that contain TLM2.0 sockets
  - Distinct implementation for Simics-to-SystemC and SystemC-to-Simics gaskets
- TLM2.0 and TLM1.0 supported by RISC
- Simics-to-SystemC Communication
  - Gaskets interface to special SystemC target socket or port
- SystemC-to-Simics Communication
  - Initiator socket interfaces to gasket

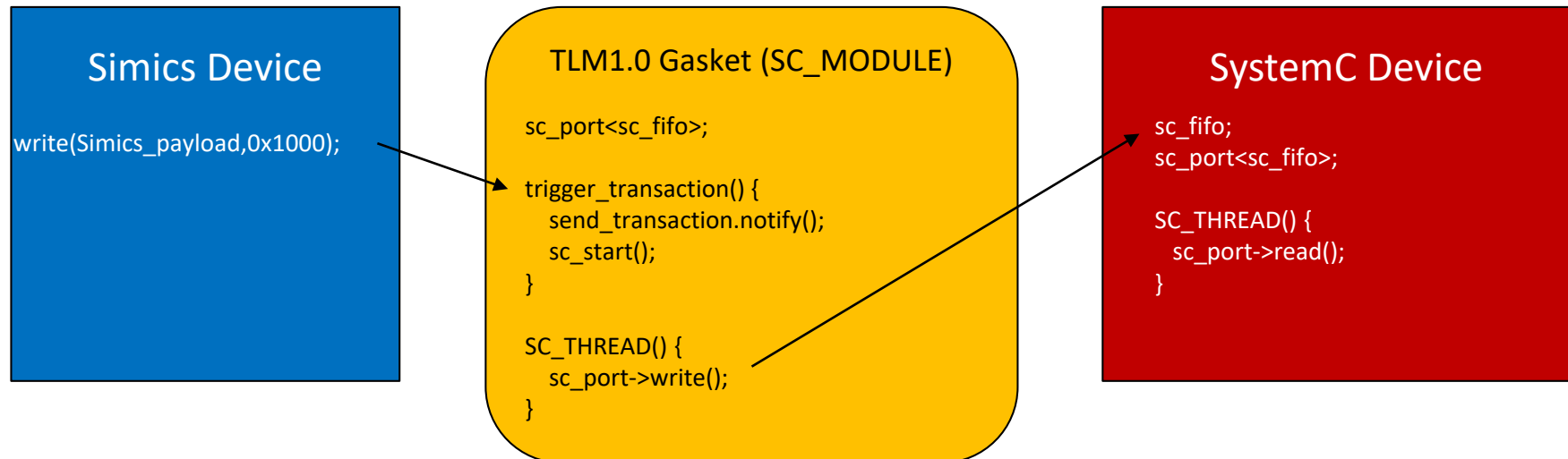


# Simics-to-SystemC Communication

Via TLM2.0:

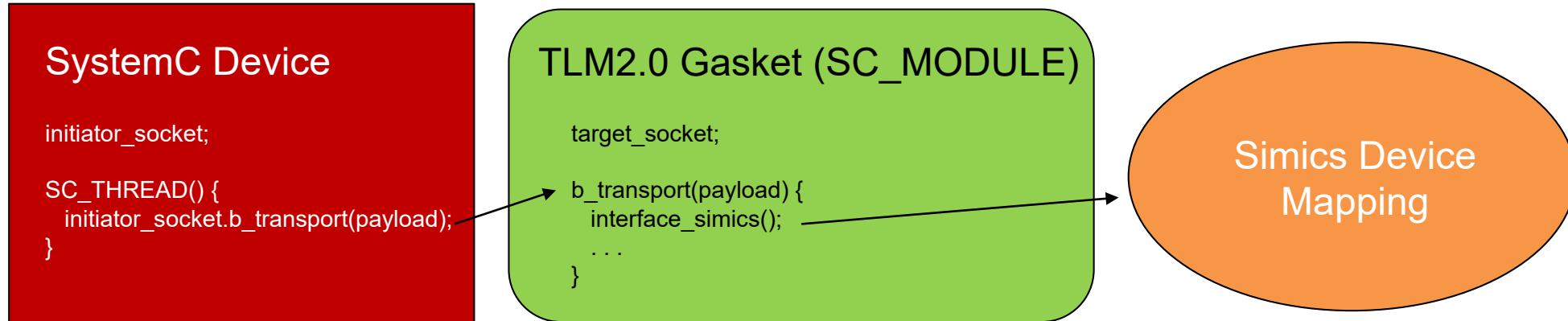


Via TLM1.0:

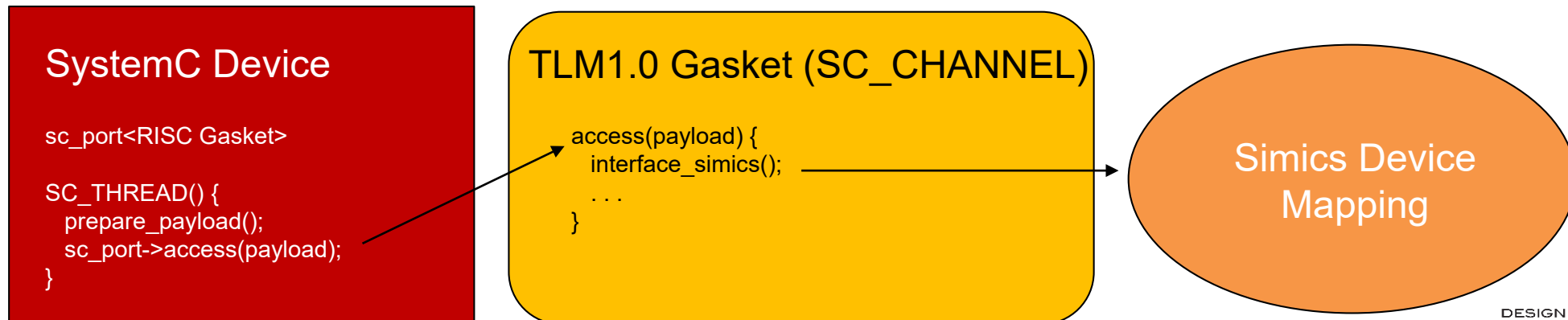


# SystemC-to-Simics Communication

Via TLM2.0:

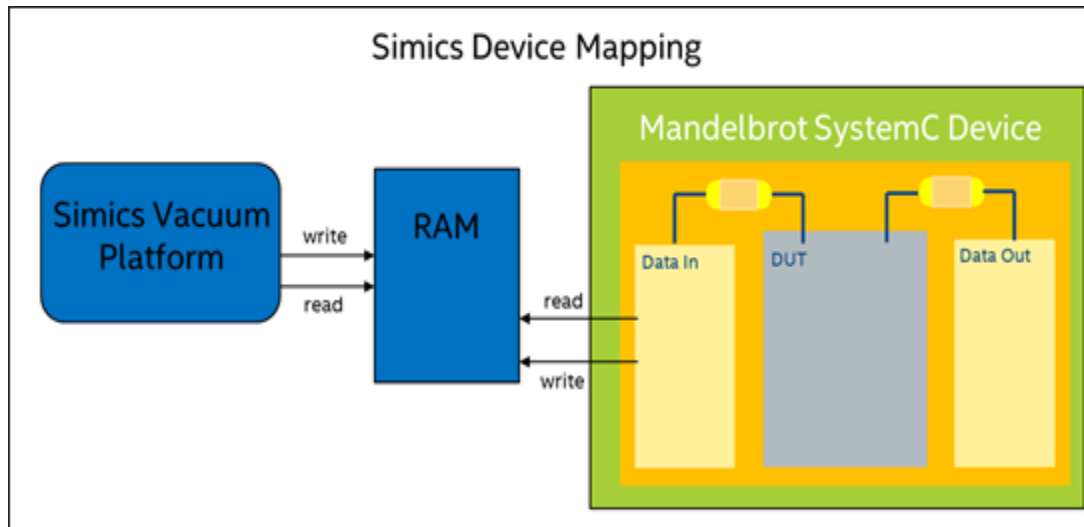


Via TLM1.0:



# Experiments of Parallel SystemC in Simics

- Handshaking Mandelbrot Renderer
  - SystemC Device reads coordinates from RAM device written by Simics Vacuum platform and renders an image corresponding to the input coordinates
  - 8 core Intel® Xeon® Processor E5-2670 (2.60GHz) and 60 core Intel® Xeon Phi™



	Standard SystemC Runtime	Standard SystemC CPU Utilization	RISC Runtime	RISC CPU Utilization	RISC Speedup	# of cores	Efficiency
Simics	59.93s	99%	9.37s	641%	6.40x	8	80.0%
Without Simics	394.2s	99%	7.9s	4902%	49.90x	60	83.2%

# Experiments of Parallel SystemC in Simics

- Panorama Filter Application
  - Inputs a number of images of the same panorama and attempts to remove the people within the image (Azumi et al. 2012). Example Input:





# Experiments of Parallel SystemC in Simics

- Panorama Filter Application
  - For each input Image, has a corresponding output image with people “disappearing”



# Experiments of Parallel SystemC in Simics

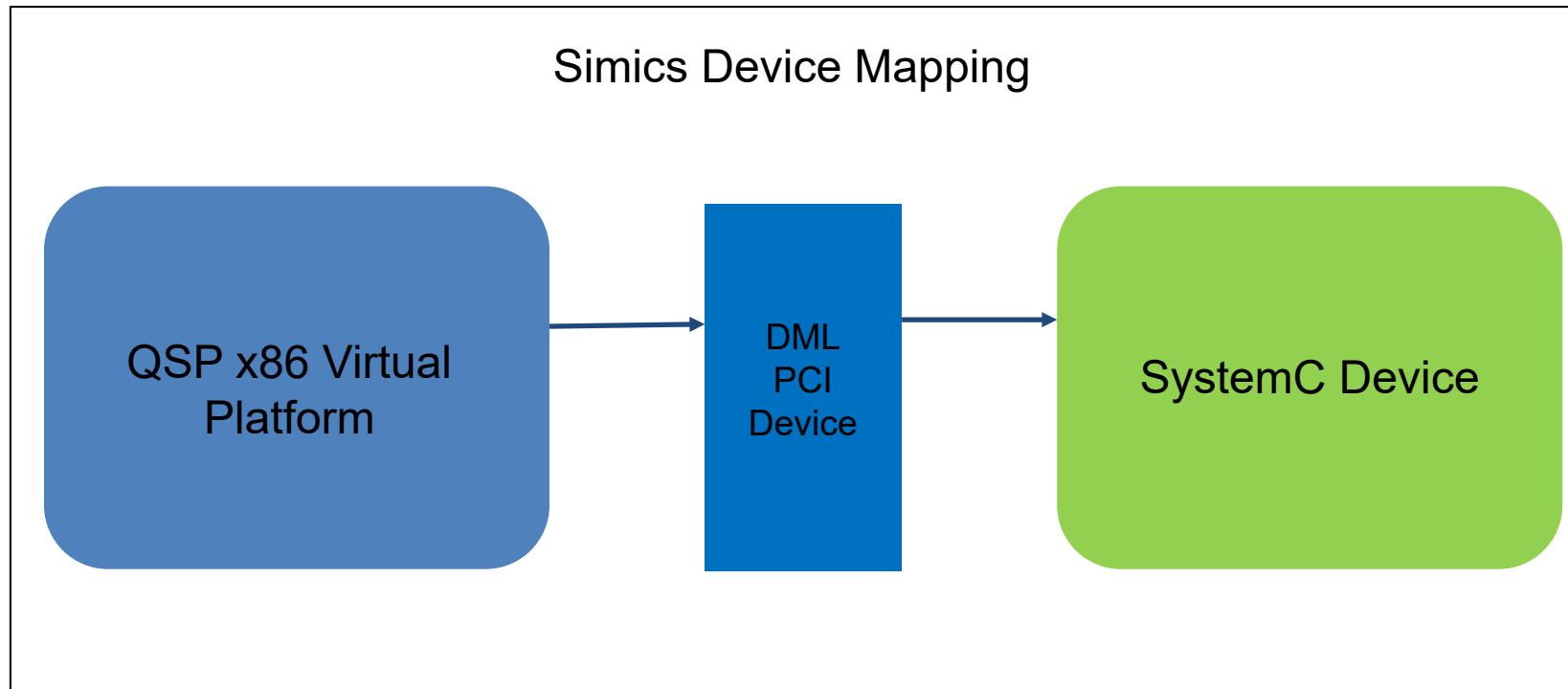
- Panorama Filter Application
  - Final output:





# Experiments of Parallel SystemC in Simics

- Panorama Filter Application features a Linux VP and PCI communication
  - Use PCI to allow communication between a Linux-based VP and SystemC Device



# Experiments of Parallel SystemC in Simics

- Panorama Filter Application Results
  - 8 core Intel® Xeon® Processor E5-2670 (2.60GHz)

	Standard SystemC Runtime	Standard SystemC CPU Utilization	RISC Runtime	RISC CPU Utilization	RISC Speedup	# of cores	Efficiency
Simics	50.11s	77%	27.20s	141%	1.84x	8	23.0%
Without Simics	73.42s	91%	49.47s	137%	1.48x	8	18.4%

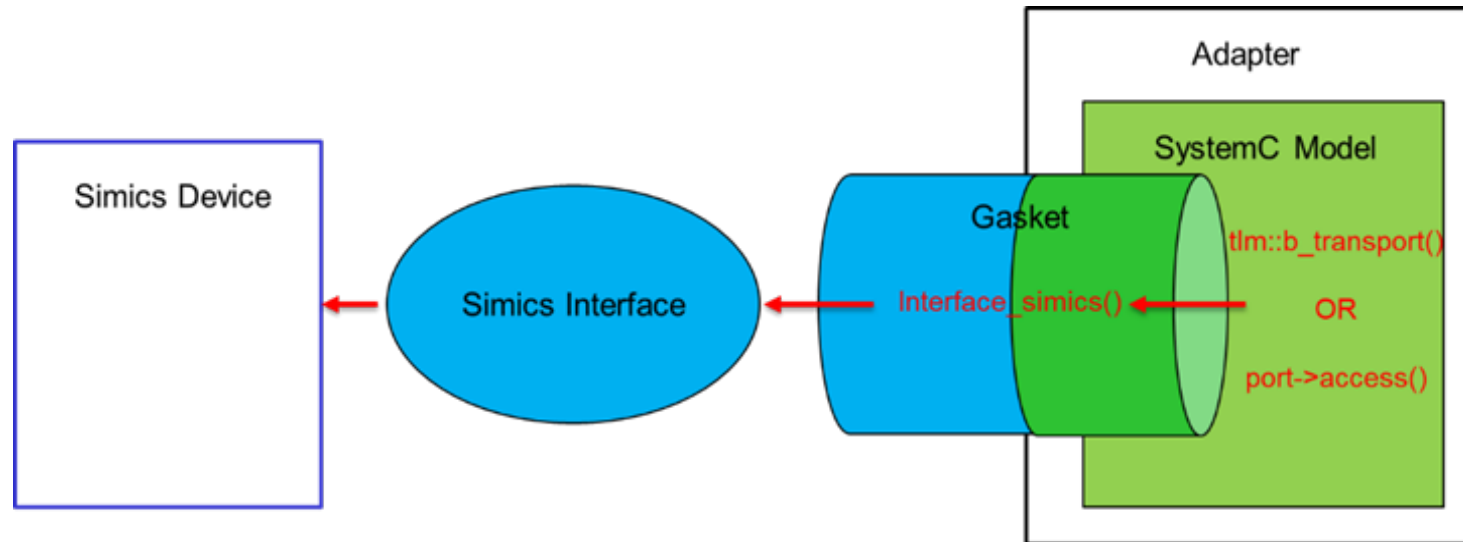
# Conclusions

- Two successful cases of a Simics simulation leveraging RISC
- Exhibits significant speedup
- Combination of RISC and Simics is feasible and valuable
- RISC can be used in a practical and realistic Simics simulation environment

# Questions

# SystemC-to-Simics Device Communication

- initiator socket `b_transport` call interfaces to gasket
  - SystemC-to-Simics gasket completes the TLM-2.0 transaction



# Simics-to-SystemC Device Communication

- TLM2.0 now available in RISC
  - Simics-to-SystemC Gaskets interface to special SystemC target socket or port

