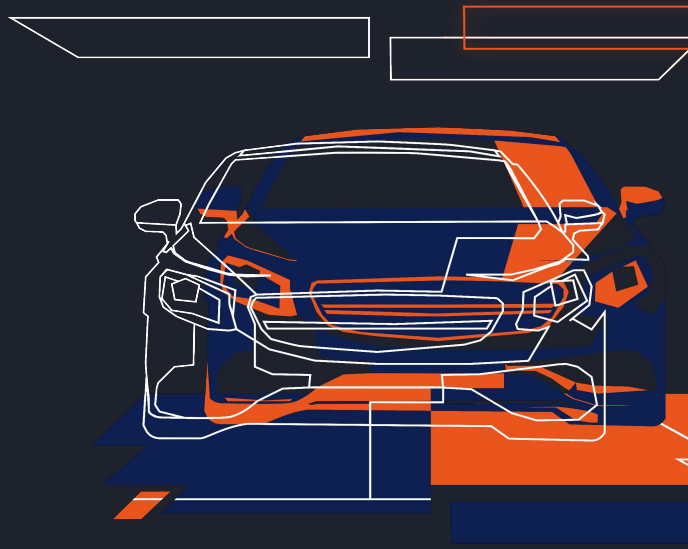
Presentation Copyright Permission

- A non-exclusive, irrevocable, royalty-free copyright permission is granted by **MachineWare GmbH** to use this material in developing all future revisions and editions of the resulting draft and approved Accellera Systems Initiative **SystemC** standard, and in derivative works based on the standard.

accellera SYSTEMS INITIATIVE



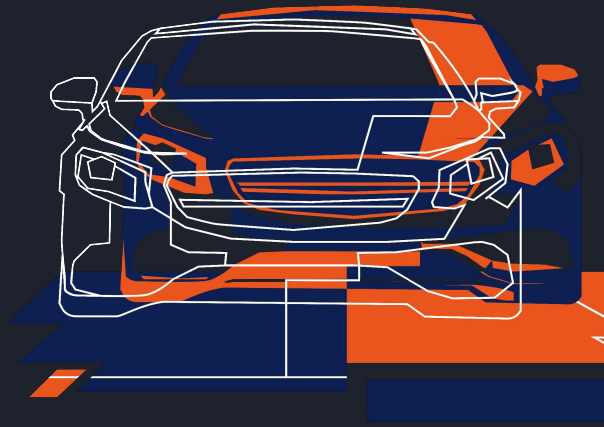
MACHINEWARE

Towards a Unified Solution for

SystemC Tracing and Analysis

Lukas Jünger, MachineWare GmbH

lukas@mwa.re



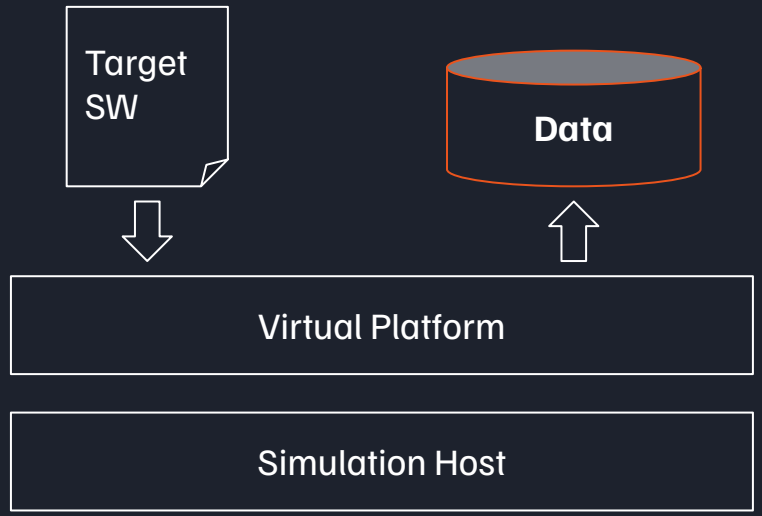
MACHINEWARE

Speaker Introduction

- Co-Founder and Managing Director at MachineWare GmbH
- Working on simulation topics since 2017 (GreenSocs, Synopsys, ...)
- Scientific background
  - PhD researcher at RWTH Aachen until 2023
  - Topics: Full System Simulation, Novel Acceleration Techniques
- Active at Accellera Systems Initiative
  - Chair of SystemC CCI Working Group
  - Member of SystemC Language Working Group and Common Practices Subgroup

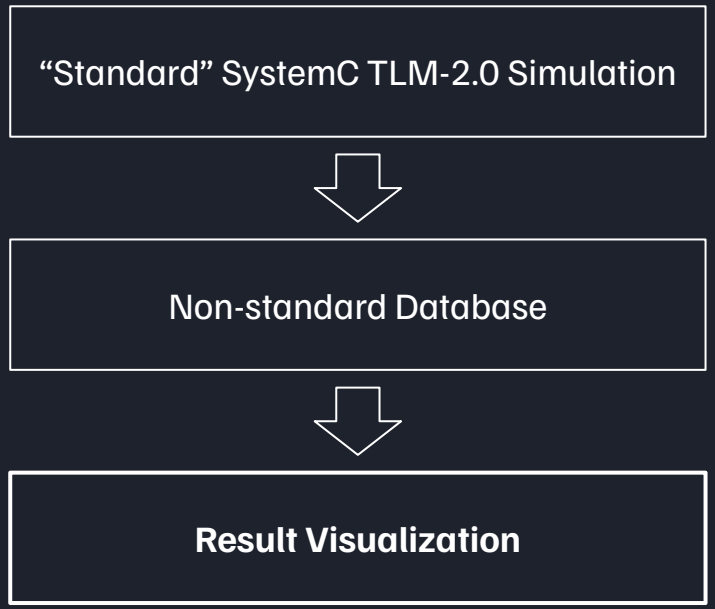
Virtual Platforms

- Virtual Platform: Full System Simulation
- Use cases:
  - SW verification & analysis
  - Architecture exploration
- Generates simulation data
  - SystemC data (events, processes, TLM, …)
  - Performance numbers
  - Target SW behavior
  - CPU cache usage
  - …
- User cares most about resulting data



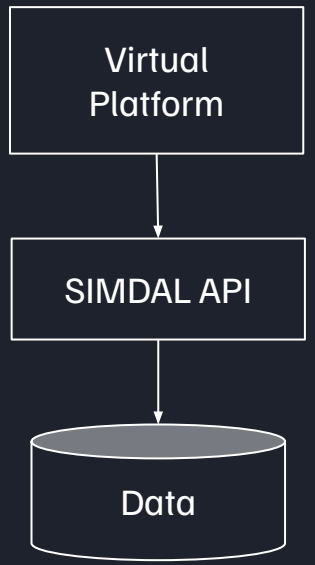
Problem Statement

- No standard for storing simulation data
  - Except VCD for signals
- No standardized format for data
  - Low hanging fruit: SystemC data
  - More complex: Model-specific data
- Even worse in federated simulation
- Need to evolve SystemC to cover this



Proposed Solution

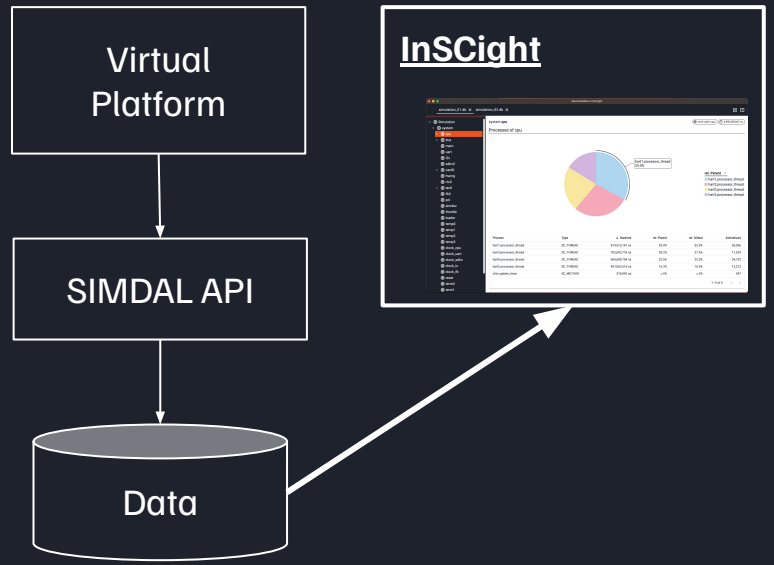
- Step 1: Align on storage format
  - Our proposal: SQLite database
- Divide and conquer
  - SystemC data first, more complex data later
- Evaluated at ICE, RWTH Aachen: SIMDAL[1]



[1] https://github.com/nbosb/simdal

Proposed Solution

- Step 1: Align on storage format
  - Our proposal: SQLite database
- Divide and conquer
  - SystemC data first, more complex data later
- Evaluated at ICE, RWTH Aachen: SIMDAL[1]
- Development at MachineWare: InSCight
- SystemC TLM-2.0 compatible



[1] https://github.com/nbosb/simdal

Performance Analysis

- But isn’t SQLite too slow? No.

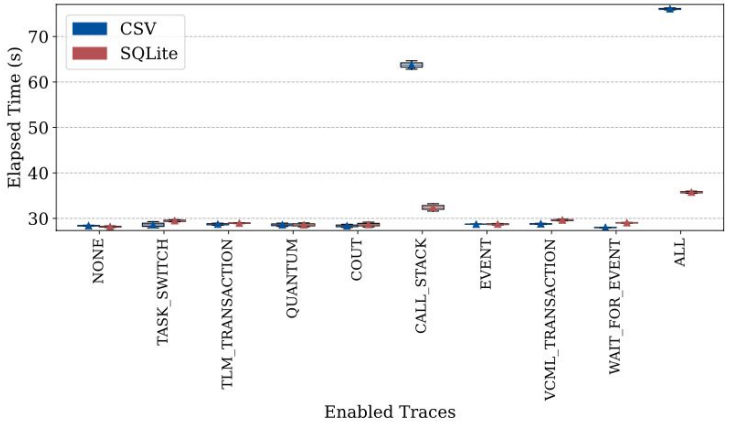


Figure 5.6: Time it takes to simulate 2 s of Linux boot on AVP64 with different enabled traces.

AVP64: https://github.com/aut0/avp64

Summary

- Take away: SQLite works fine
- Important: Align on trace format
- Want to learn more?
  - Follow us on LinkedIn mwa.re/li
  - Star us on GitHub mwa.re/gh
  - E-mail me at lukas@mwa.re
  - Join Accellera and get involved