2025
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
MUNICH, GERMANY
OCTOBER 14-15, 2025

Tutorial: Scalable Virtual Platforms for Automotive and Beyond

Introduction

Lukas Jünger, MachineWare GmbH

MachineWare arm tracetronic ICE Institute for Communication Technologies and Embedded Systems RWTH AACHEN UNIVERSITY accellera SYSTEMS INITIATIVE

Agenda

- Session #1
  - VCML: An Open-Source Framework for Building Scalable Virtual Platforms
  - Lukas Jünger, MachineWare GmbH

- Session #2
  - Trends in Software and Virtual Platforms
  - Daniel Owens, Arm

- Session #3
  - Integrating Virtual Platforms as Scalable Testbeds for Automotive Software
  - Matthias Berthold, tracetronic GmbH

- Session #4
  - Virtual Platforms for Embedded Fuzzing
  - Chiara Ghinami, RWTH Aachen

MachineWare

2025
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
MUNICH, GERMANY
OCTOBER 14-15, 2025

Tutorial: Scalable Virtual Platforms for Automotive and Beyond

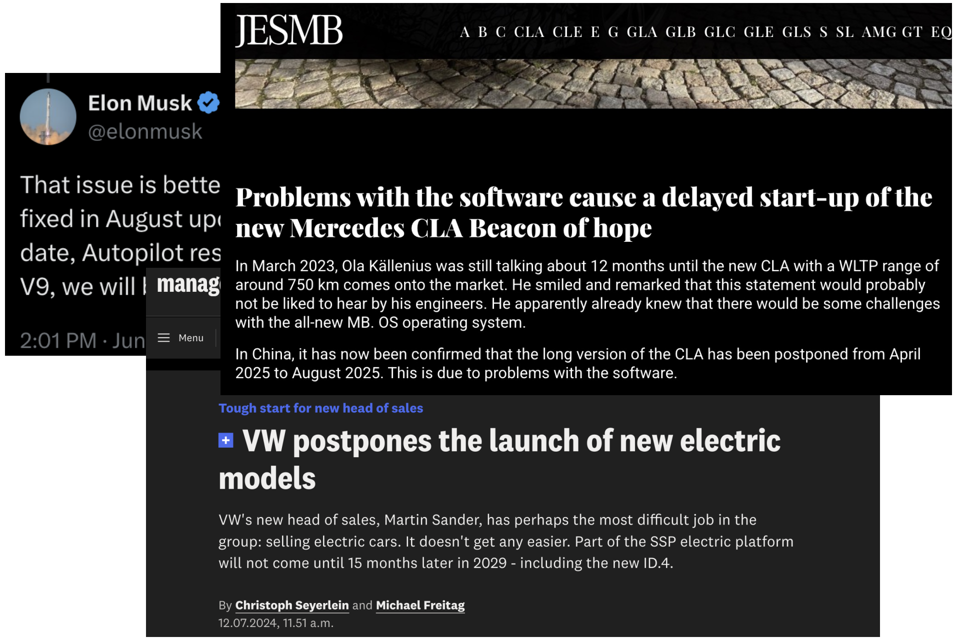
Session #1: VCML: An Open-Source Framework for Building Scalable Virtual Platforms

Lukas Jünger, MachineWare GmbH

accellera
SYSTEMS INITIATIVE

Motivation

- Software complexity ever increasing
  - Software Defined Vehicle
  - Integration points extremely complex
- Software problems lead to delays
- Software TCO high
  - Fixing problems late is expensive (10-100x)
- Need: **Better software, earlier!**



Elon Musk
@elonmusk

That issue is bette
fixed in August up
date, Autopilot res
V9, we will

2:01 PM · Jun

JESMB

A B C CLA CLE E G GLA GLB GLC GLE GLS S SL AMG GT EQ

Problems with the software cause a delayed start-up of the new Mercedes CLA Beacon of hope

In March 2023, Ola Källenius was still talking about 12 months until the new CLA with a WLTP range of around 750 km comes onto the market. He smiled and remarked that this statement would probably not be liked to hear by his engineers. He apparently already knew that there would be some challenges with the all-new MB. OS operating system.

In China, it has now been confirmed that the long version of the CLA has been postponed from April 2025 to August 2025. This is due to problems with the software.

Tough start for new head of sales

VW postpones the launch of new electric models

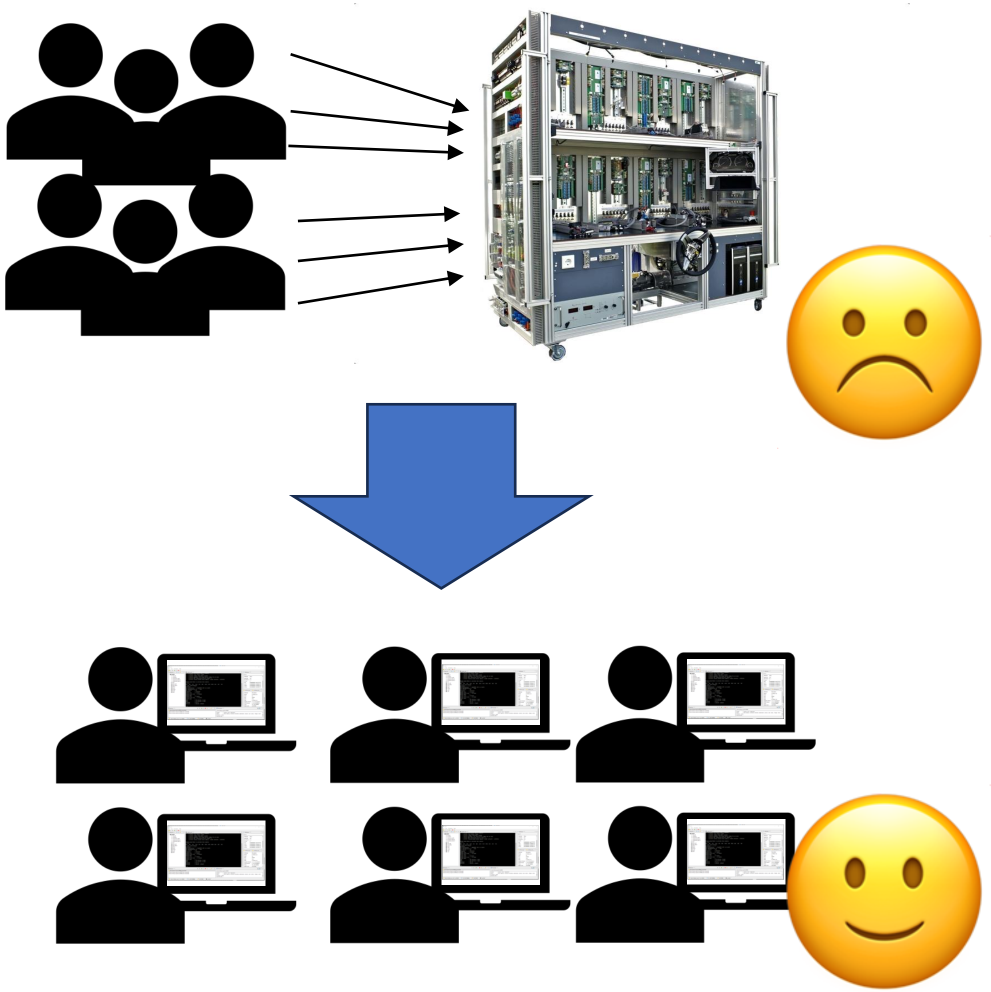
VW's new head of sales, Martin Sander, has perhaps the most difficult job in the group: selling electric cars. It doesn't get any easier. Part of the SSP electric platform will not come until 15 months later in 2029 - including the new ID.4.

By Christoph Seyerlein and Michael Freitag
12.07.2024, 11.51 a.m.

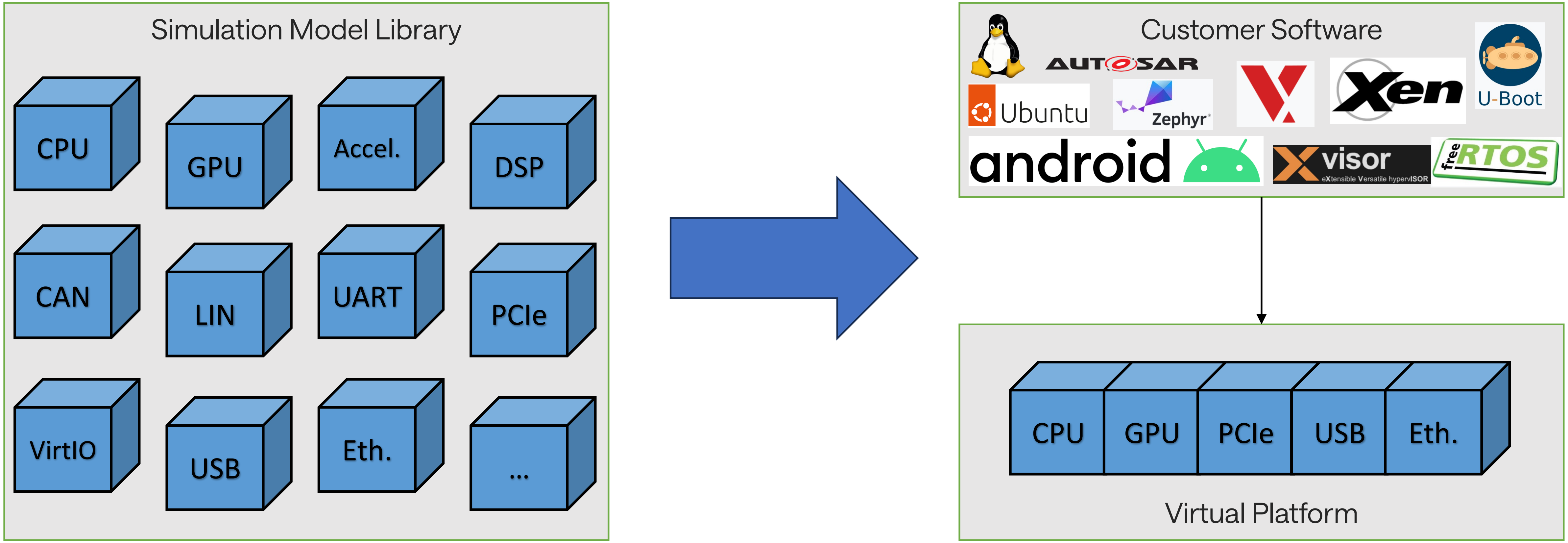
At CARIAD, revenue from contract licenses rose by around 30 percent to EUR 1.1 billion, as the software is being used in increasingly more Group vehicles, as planned. Due to the business model, this division recorded an operating loss of EUR 2.4 billion, as CARIAD makes significant advance payments for future software architectures, which are remunerated via license payments. In operational terms, the Group in the software area has focused on launching important products such as the Porsche Macan Electric and the Audi Q6 e-tron this year.

Virtual Prototyping

- Virtual Platform: **Full System Simulation**
  - Execute unmodified target software
- Indispensable in modern SW development
- Advantages over physical prototypes
  - Available early
  - Scalable deployment
  - Full flexibility, deep introspection

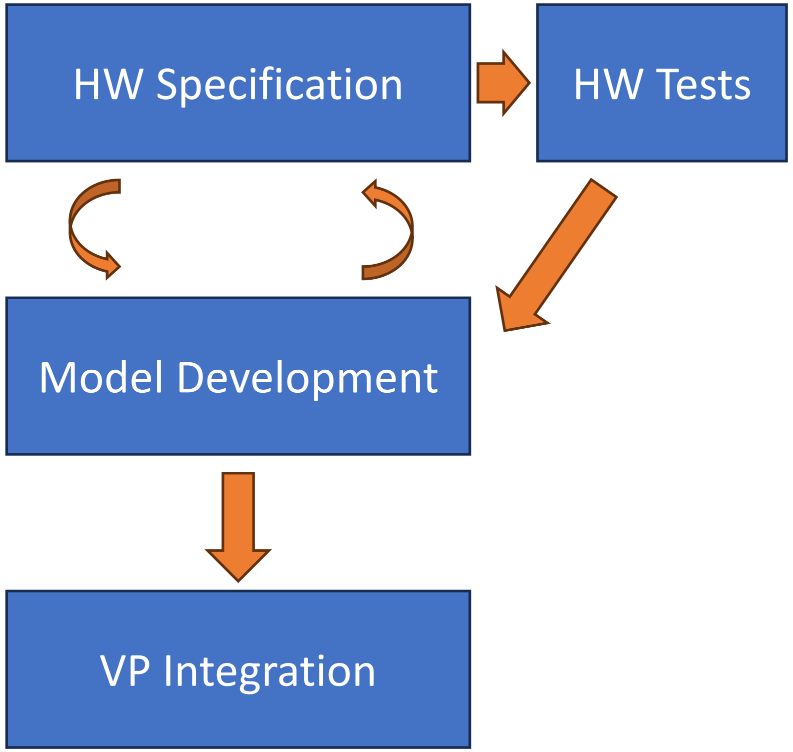


Building Virtual Platforms



Scalability Challenge: Early Availability

- Requires **model availability and integration**
- Building (fast) models
  - HW spec. availability and format
    - IP-XACT, SAIL vs PDF, Excel
  - Requires trained engineers (AI?)
- Integrating models from different sources
  - Build & execution environments
  - Model interfaces
    - Functional: model input/output
    - Non-functional: Configuration, Control, Inspection

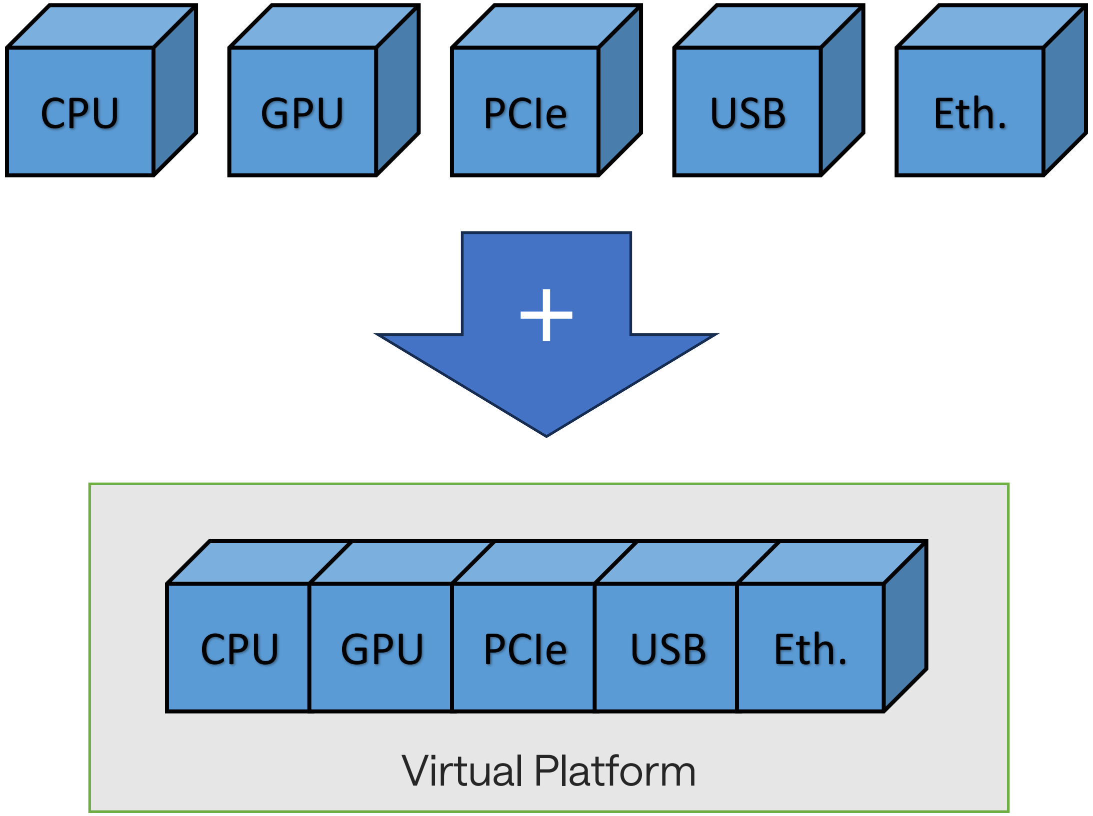


Scalability Challenge: Performance

- Usability of VP is determined by performance
  - SW execution in MIPS (Million Simulated Instructions per Wall-Clock second)
- **Performance / Accuracy trade-off**
  - Higher accuracy -> simulate more -> lower performance
  - Accuracy requirement depends strongly on use case
    - Difficult to define use case: ”simulate everything”
- Introspection/profiling for all VP components

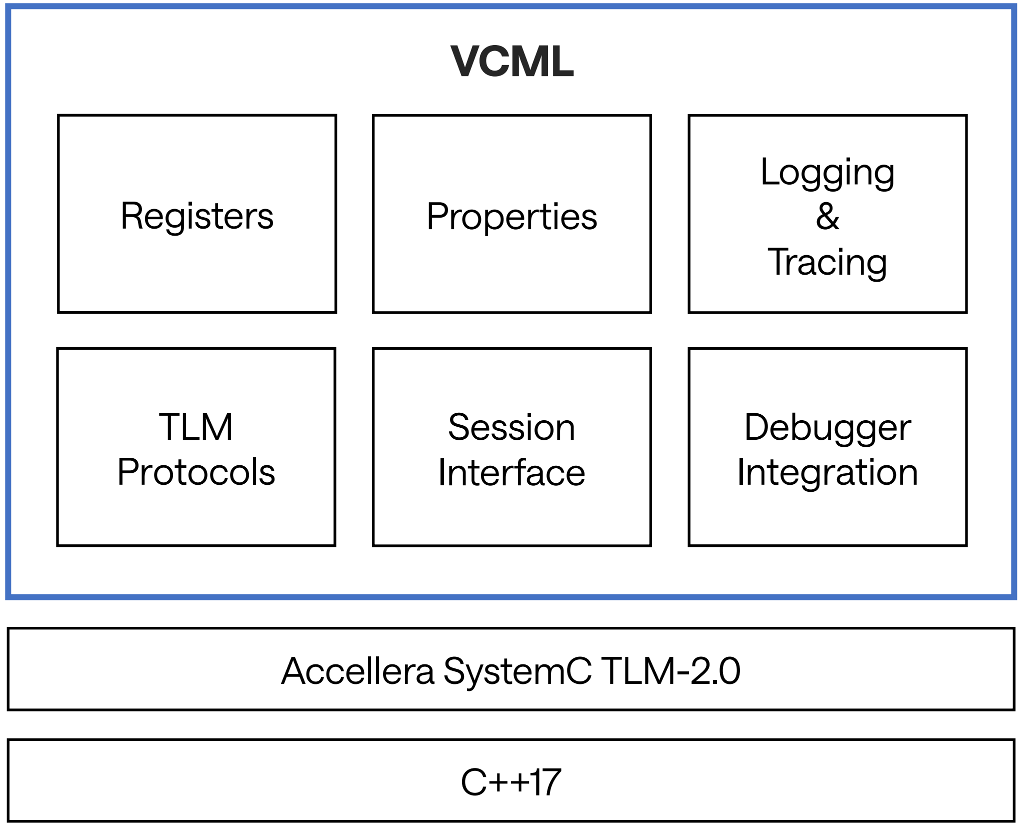
SystemC TLM-2.0

- IEEE standardized simulation framework (1666-2023) in C++
  - TLM extension for fast simulation
- Focus on model interfaces, virtual time keeping
- Missing
  - Model internals (e.g. registers)
  - Configuration, Control, Inspection
  - Interconnects (CAN, SPI, …)
  - …



VCML

- SystemC-based VP toolkit
  - Everything that’s missing in SystemC
  - Free and open-source (Apache 2)
    - github.com/machineware-gmbh/vcml
- Goals
  - Training resource for newcomers
    - machineware.de/vcml-community
  - Open, scalable platform for building fast VPs and models
  - “Batteries included” (examples, documentation, support)



VCML Ecosystem

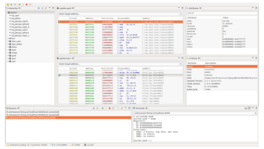
Standards/Interfaces

- FMI – Functional Mock-up Interface
- sil-kit
- SystemC™
- VCML Session

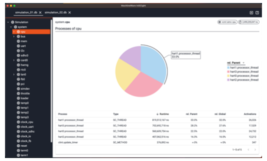
Virtual Hardware

- arm
- RISC-V®
- Renesas
- QEMU
- BOSCH
- VirtIO
- Codasip
- Andes Technology

MachineWare Tools



ViPER



InSCight

3rd Party Tools

- Lauterbach Development Tools
- TASKING
- SEGGER
- iar
- Rapita Systems – a Danlaw company
- dSPACE
- Vector
- Ashling – Experts in Embedded Development Tools & Services – www.ashling.com
- tracetronic
- Codasip

MachineWare VCML

Accellera SystemC TLM-2.0

VCML is a powerful open-source foundation for building fast, scalable Virtual Platforms

2025
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
MUNICH, GERMANY
OCTOBER 14-15, 2025

Trends in Software and Virtual Platforms

Daniel Owens, Product Director, Arm

arm

accellera
SYSTEMS INITIATIVE

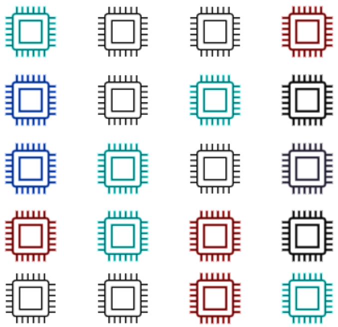
Trends in Software

SYSTEMS INITIATIVE

The AI-Defined Vehicle

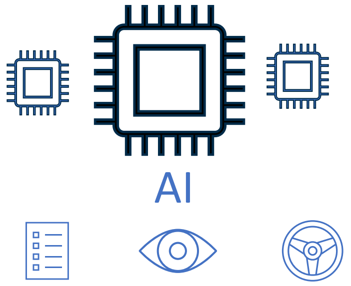
PAST

- Distributed ECUs, rule-based, deterministic systems
- Not software-defined, but software-controlled



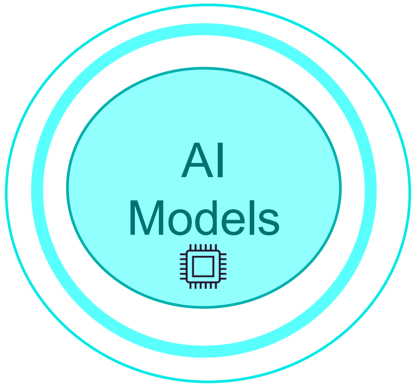
PRESENT

- Centralized/Zonal compute vs distributed
- Modular system with LLMs emerging
- Software defines the system, orchestrates, OTA updates



FUTURE

- Sophisticated AI models pervasive
- AI central to vehicle innovation
- Software-Defined infrastructure as foundation



Software Controlled | Software-Defined Vehicle Era | AI-Defined Vehicle Era

Enabling Safer, Smarter Driver Experiences

New AI Models Underpin Driver Experiences of the Future

ADAS

Example: Intuitive Driving

Adjusts to environment and driver behavior in real-time to enhance safety. V2X coordination.

"Switching to 'City Mode' for enhanced awareness."

IVI

Example: Voice & App Interaction

Delivers personalized, voice-driven control for a more intuitive in-vehicle experience.

"Play my playlist and order my usual at Starbucks."

Vehicle System Control

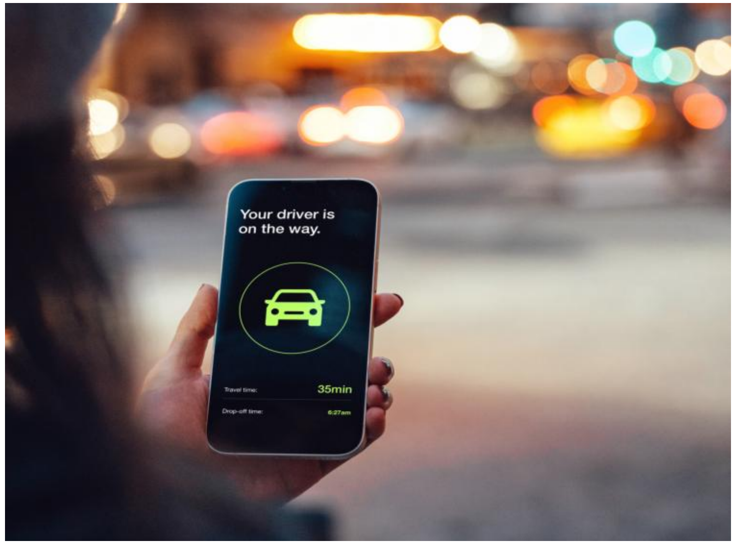
Example: Real-time Optimizations

Continuously improves performance and efficiency based on conditions.

"No passengers – Eco Drive activated."

Industry Needs to Deliver the AI-Defined Vehicle Reality

More Complexity, More Demands on Vehicle Hardware and Software



Flexibility to innovate

on top of proven safety-capable, secure compute foundations

Scale AI

scalable platforms to power AI across diverse vehicle types and use cases

Faster time to market

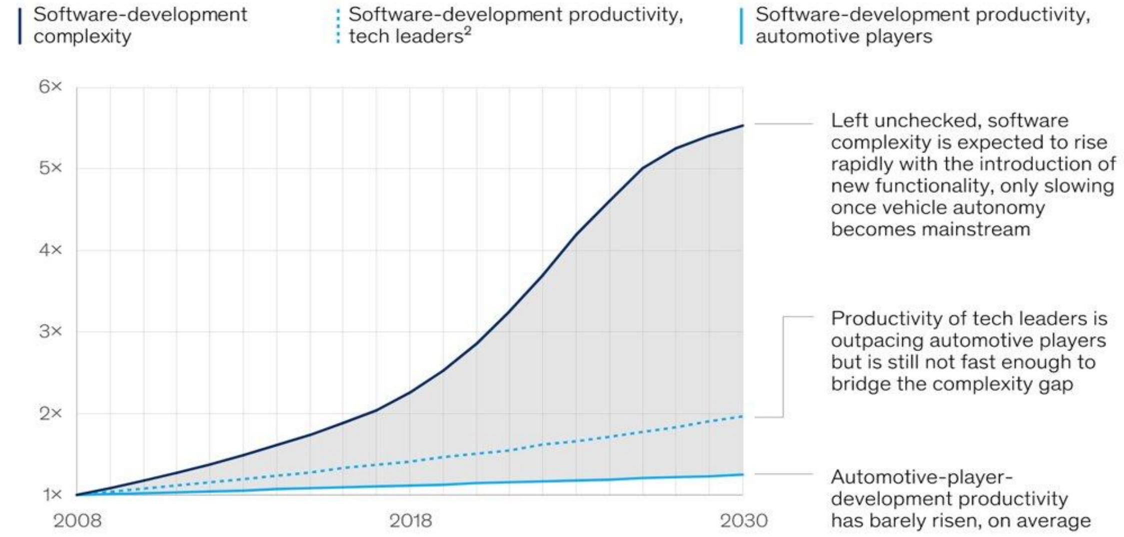
for tech powering new driver experiences

Significant Software Challenges for Automotive

Complexity is ballooning, while development productivity is lagging[1]

The automotive industry is confronting a widening and unsustainable gap between software complexity and productivity levels.

Relative growth over time, for automotive features,[1] indexed, 1 = 2008

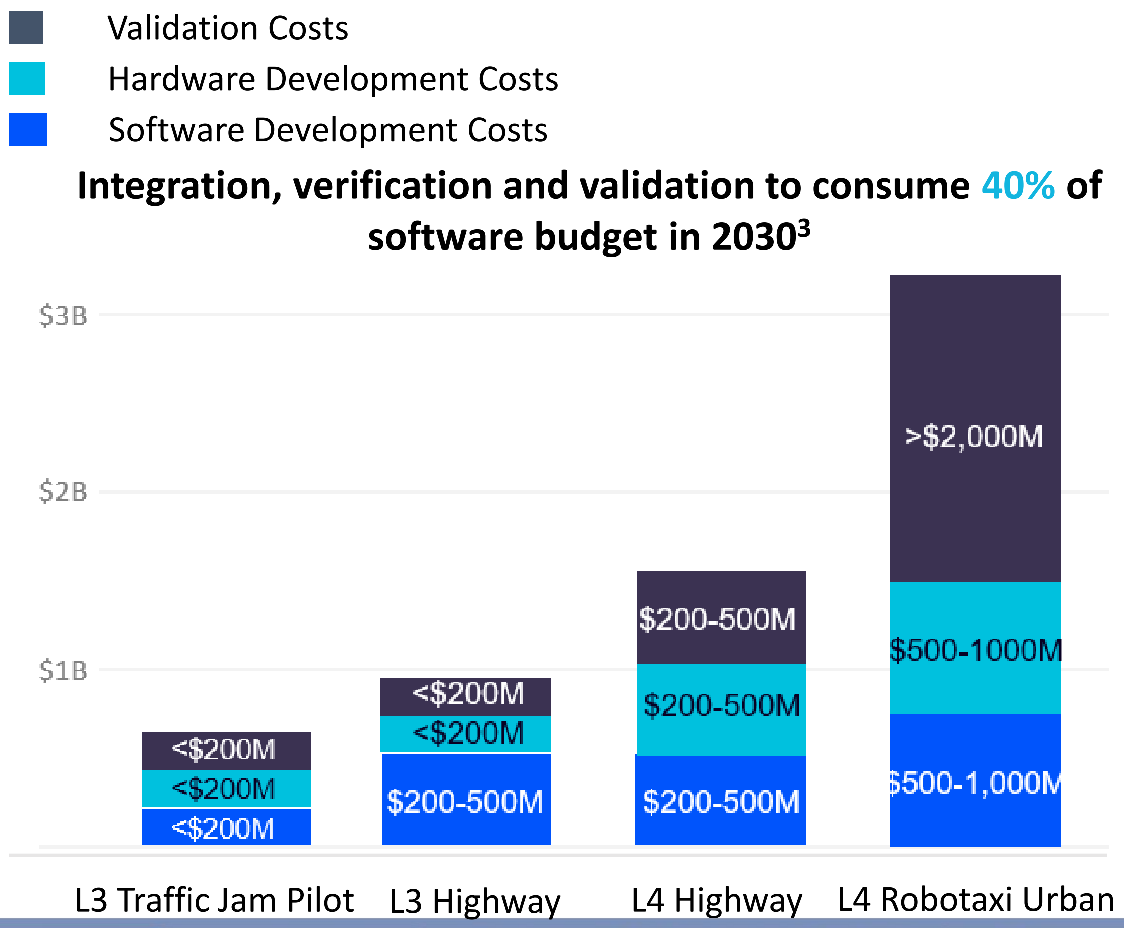


[1]Analysis of >200 software-development projects from OEMs and from tier-1 and tier-2 suppliers.
[2]Top-performing quartile of technology companies.
Source: Numetrics by McKinsey

McKinsey & Company

1 McKinsey, The case for an end-to-end automotive-software platform, January 16, 2020

Autonomous Vehicle Development Costs (USD)[2]



2 McKinsey, What's Next for Autonomous Vehicles
3 McKinsey, Automotive software and electronics 2030

accellera SYSTEMS INITIATIVE

2025 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

The Arm Automotive Technology Stack

A Complex Landscape That Requires Collaboration

AUTOMOTIVE APPLICATIONS

- AUTONOMOUS / ADAS
- IN-VEHICLE INFOTAINMENT / DIGITAL COCKPIT
- VEHICLE CONTROL

FOUNDATIONAL AUTOMOTIVE SOFTWARE

(OPERATING SYSTEMS & MIDDLEWARE)

VIRTUAL PROTOTYPING SOLUTIONS

(VIRTUAL PLATFORMS & CLOUD INFRASTRUCTURE)

SCALABLE AUTOMOTIVE COMPUTE HARDWARE

(HETEROGENEOUS IP & COMPUTE SUB-SYSTEMS)

The Arm Automotive Technology Stack

A Complex Landscape That Requires Collaboration

AUTOMOTIVE APPLICATIONS

- AUTONOMOUS / ADAS
- IN-VEHICLE INFOTAINMENT / DIGITAL COCKPIT
- VEHICLE CONTROL

FOUNDATIONAL AUTOMOTIVE SOFTWARE
(OPERATING SYSTEMS & MIDDLEWARE)

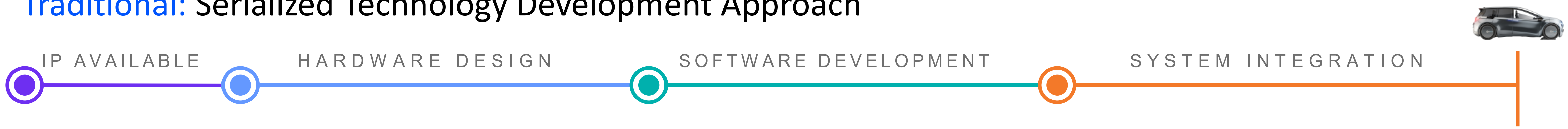
VIRTUAL PROTOTYPING SOLUTIONS
(VIRTUAL PLATFORMS & CLOUD INFRASTRUCTURE)

SCALABLE AUTOMOTIVE COMPUTE HARDWARE
(HETEROGENEOUS IP & COMPUTE SUB-SYSTEMS)

Compressing Time to Market | Arm Innovation

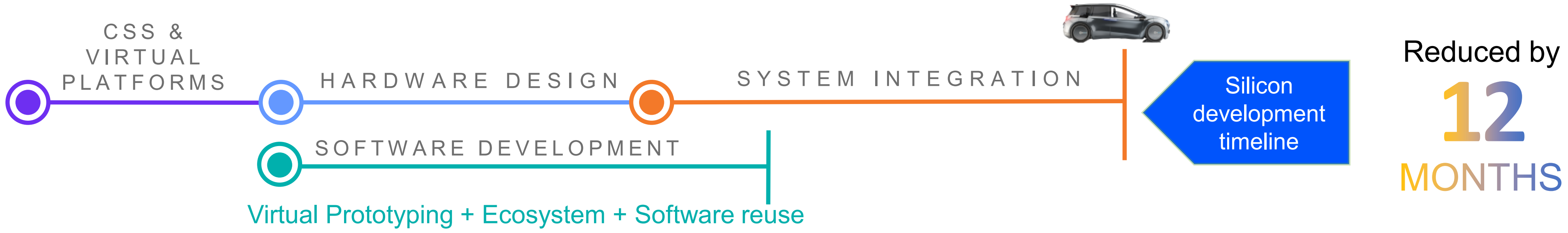
Reducing Silicon and Software Development Timelines

Traditional: Serialized Technology Development Approach



Past Years: Arm Individual IP Development Approach with Virtual Prototyping

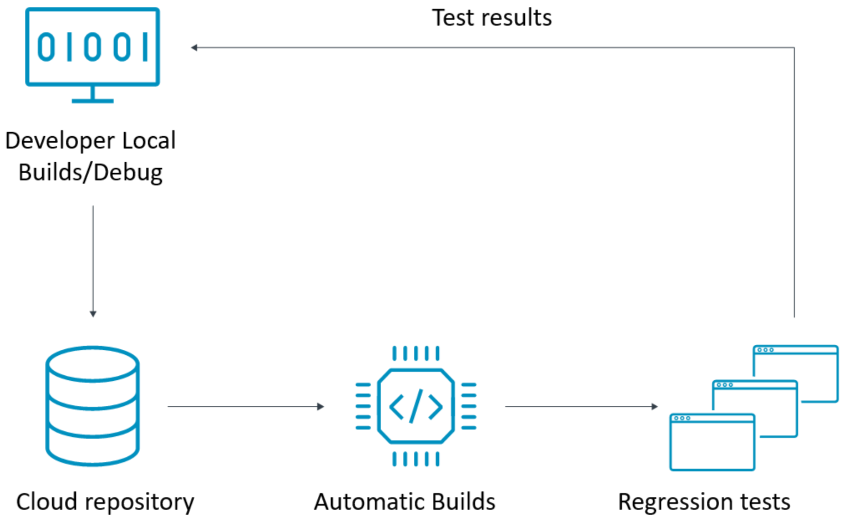
Today: Reduced Time-to-silicon and Accelerated Software Development



Trends in Arm-based Simulation

Evolving Arm Hardware Landscape

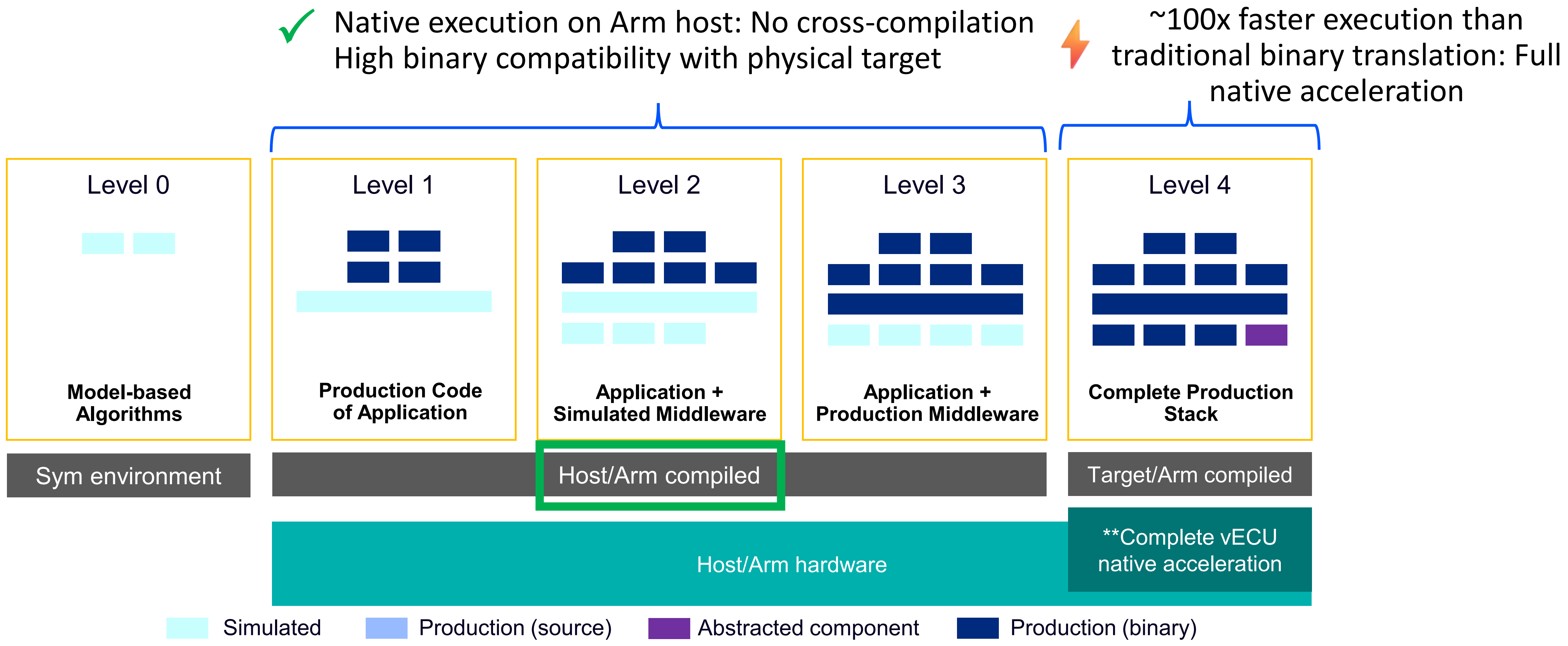
- Trends in cloud hardware
  - Google Cloud, Microsoft Azure, AWS hosting Arm servers
  - Arm v9 now in the cloud
  - Cost effective development and automation
- Arm-native execution a reality
  - ISA parity
  - Expanding ecosystem



Cloud Native Software Development

Accelerating vECU Development | Arm Native

Faster | Binary-Compatible | No Cross-Compilation | Near-Silicon Performance

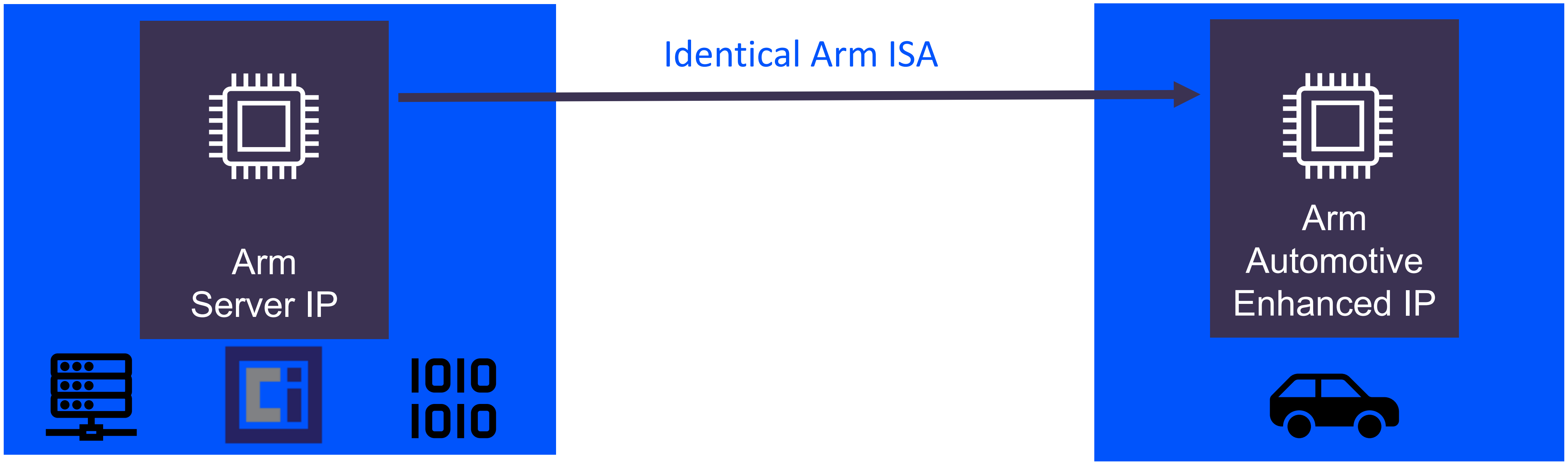


ISA Parity enables Seamless Cloud-to-Car Portability

Build Once, Run Everywhere

Cloud/OnPrem Platform

Car Platform



virtual prototypes containers algorithms/libraries

Accelerated Validation Pipeline

Use containers and virtual prototypes to validate functional software continuously – earlier, faster and cheaper.

Build once, Test everywhere

The same artifacts run natively across the cloud and vehicle edge.

True Native Execution, No cross-compilation

Arm Fast Models

Fast, accurate, flexible models of Arm CPUs, Media and System IP

Complete programmer's view, including the most complex Arm IP

Suitable for driver, firmware, RTOS and embedded application development

SystemC / TLM 2.0 compliant for platform expansion

Fast Models – Important Enabler

- IP and integration validation
  - Architecture Validation Suite/Device Validation Suite, ISS Compare, RAVEN
- Early software development (internal)
  - Reference firmware (EDK2, SCP, Trusted Firmware, etc.)
- Platform for reference software
  - Free of charge Fixed Virtual Platforms (FVPs)
- Integrated with Arm products (external)
  - Arm Development Studio
  - Keil MDK
- Integrated with ecosystem products
  - Hybrid solutions
- Augments modelling ecosystem
  - Fast Models, QEMU, Partner Models

Summary

- The trend toward AI in vehicles is driving the need for earlier and higher quality software.
- Virtual prototypes have distinct advantages over traditional hardware-based approaches in terms of earlier availability, better scalability and greater accessibility throughout the supply chain.
- Native execution is redefining simulation with near-silicon speed for both pre- and post-hardware development.
- Fast Models remain a viable technology for detailed, early access models of the Arm architecture.

END

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
OCTOBER 14-15, 2025

Tutorial: Scalable Virtual Platforms for Automotive and Beyond

Session #2: Integrating Virtual Platforms as Scalable Testbeds for Automotive Software

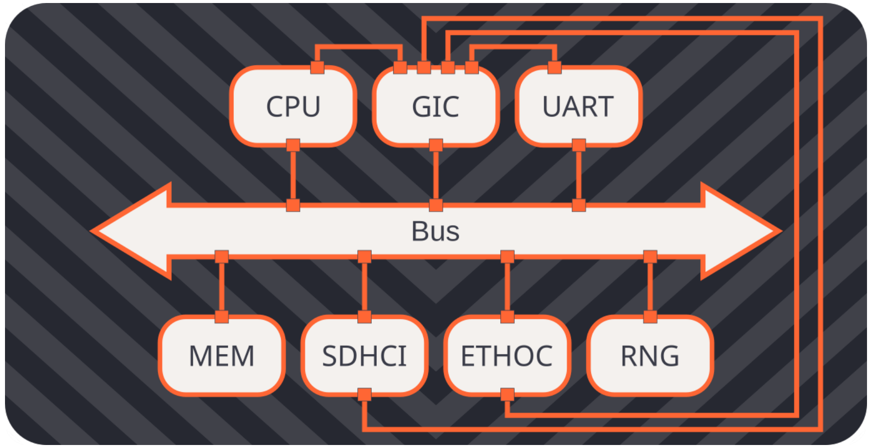
tracetronic

Matthias Berthold, tracetronic GmbH

accellera SYSTEMS INITIATIVE

Motivation

- What we see so far:
  - What is a VP?
  - How do I use it?
  - Why use it all?



- But: All for a single test, only once, **manually**.

- Before scaling: Automation!

Motivation - Why test automation?

- Increasing SW-complexity
- Tons of requirements
- Legal require-ments, e.g. ISO-26262 or UNECE
- Plenty of variants
- Iterative development
- …

Growing need of tests and test executions → ensure quality

Test automation!

Motivation – Automotive POV

- Testing and automation in Automotive:
  - Hardware-in-the-loop and lab cars
  - In-vehicle
  - Software-in-the-loop (ADAS/AD, Infotainment…)

- Current challenges/trends:
  - "Shift left" → More testing at the beginning of development
  - "Modern" software development → OTA updates…
  - More standards and open formats (e.g. ASAM XIL, OpenSCENARIO, FMI…)

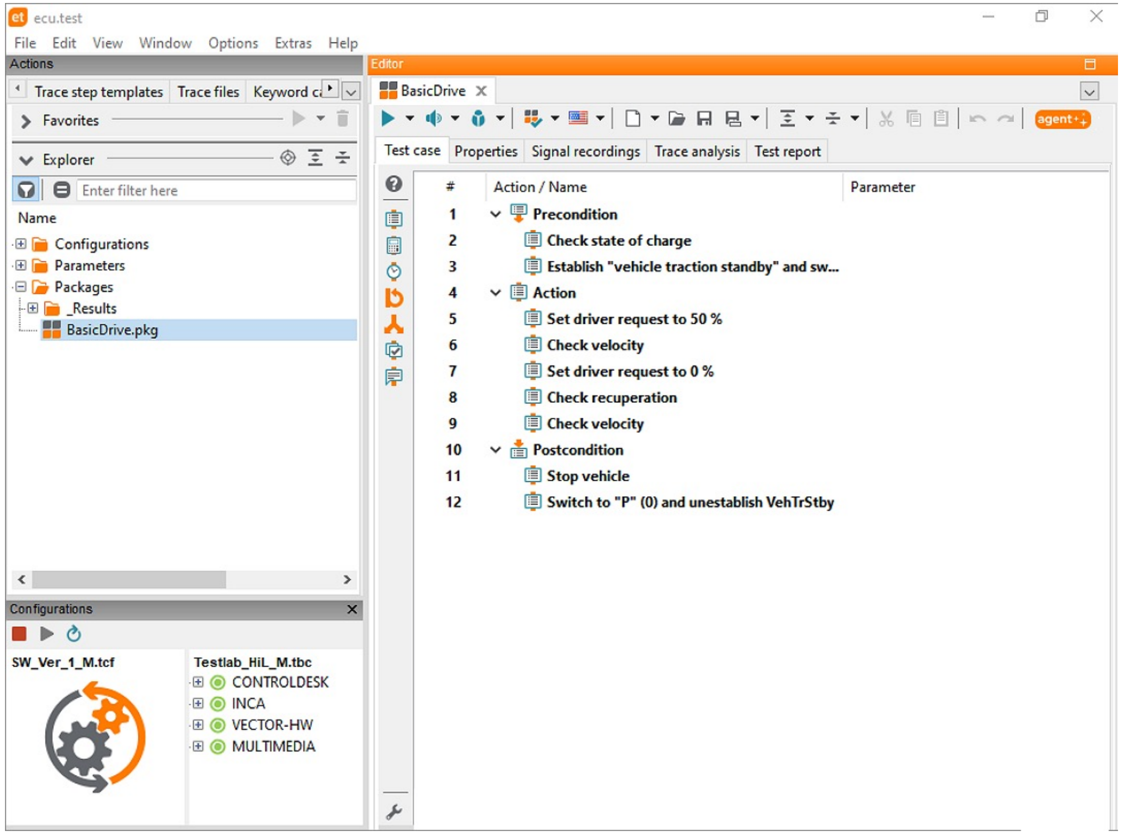
Demo – Overview

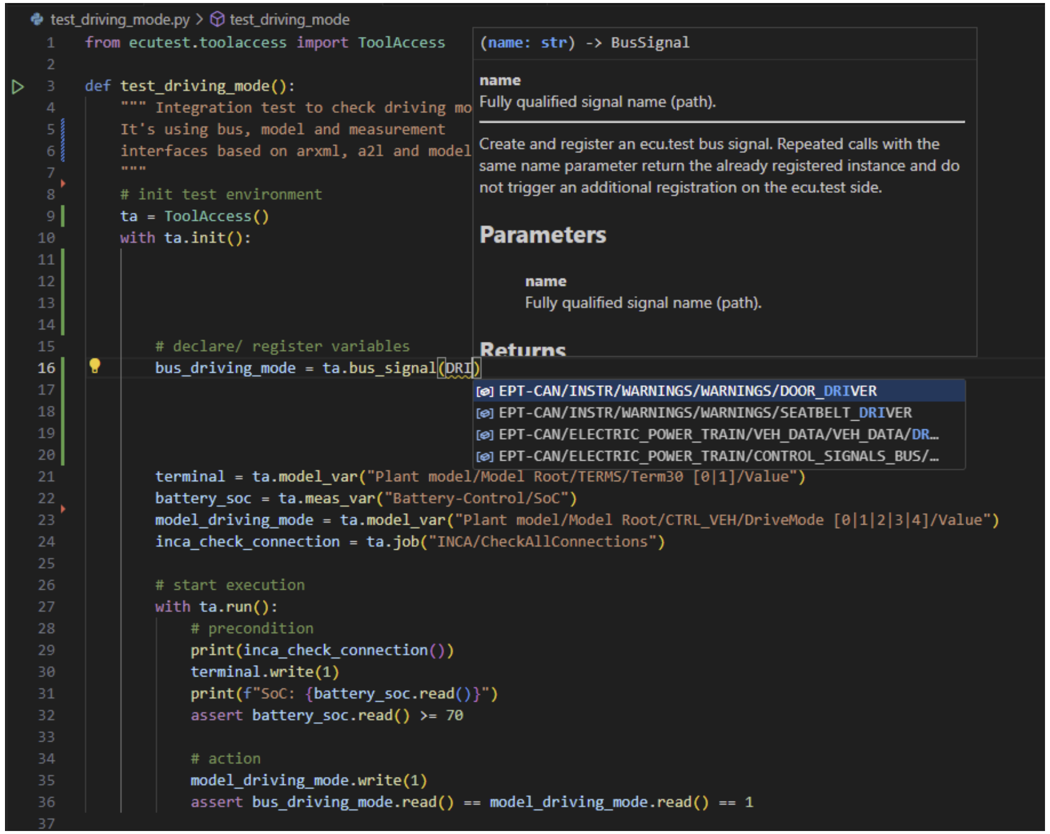
- Create configs to use the VP
- Create a test case and execute it
- Add a trace analysis to check behavior over time
- Stimulation (open loop tests)
- Reporting incl. code coverage
- Debugging

Demo – ecu.test

- Test automation: unit tests → integration tests → system tests → in vehicle tests
- Used by most automotive OEMs and Tier-1s
  - But also agricultural, Construction machines, new energy…
- GUI or Python code for test case implementation
- Over 60 tool connections and possibility for user extensions
- Connections to ALMs (sync requirements and reports)
- Q-Kit available (ISO-26262)

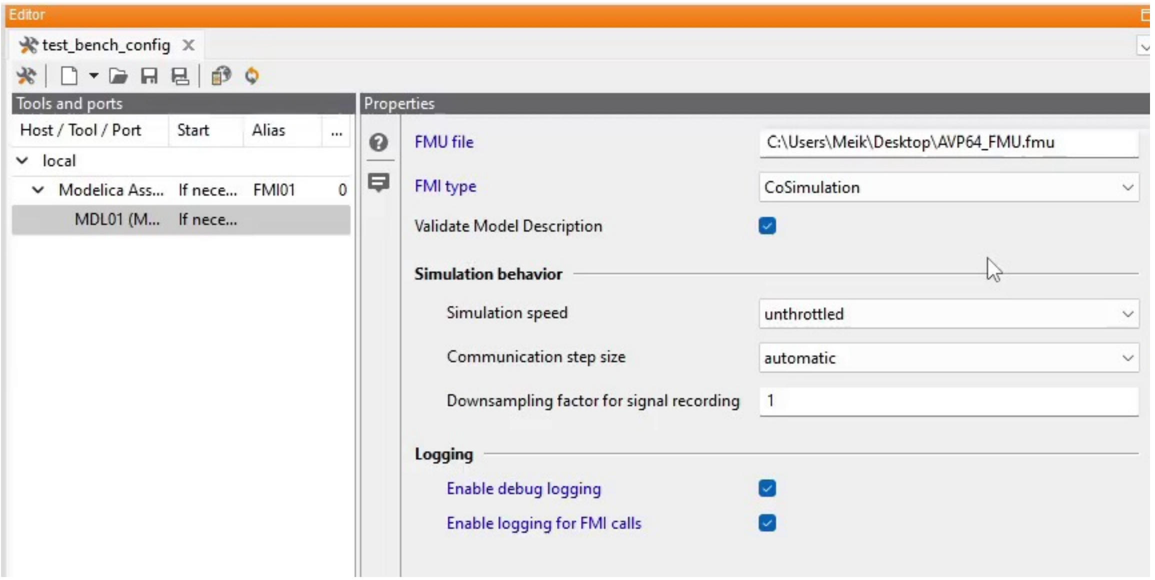
Demo – ecu.test



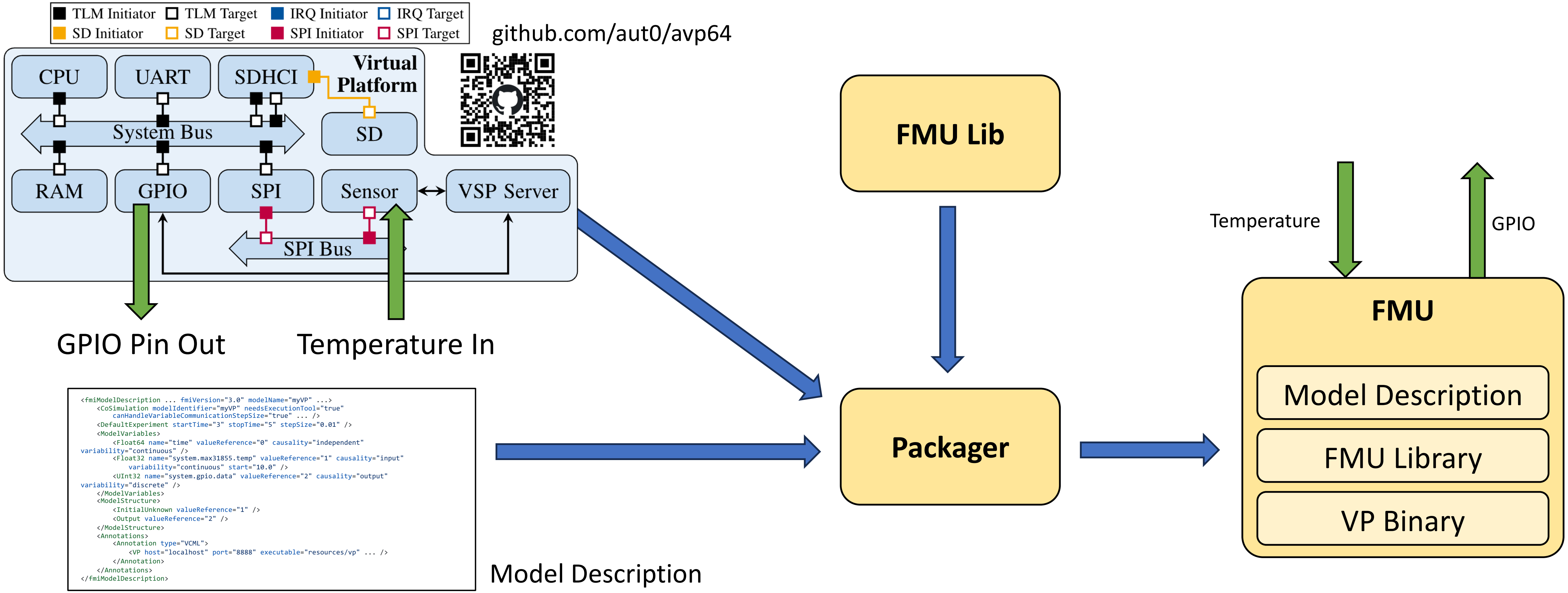


Demo – Configuration

- Connection ecu.test - VP: **FMI**
- FMI = Functional Mock-up Interface
- Free and open standard for dynamic simulation models (FMU)
- Support with > 250 tools
- Layered Standards for BUS & XCP



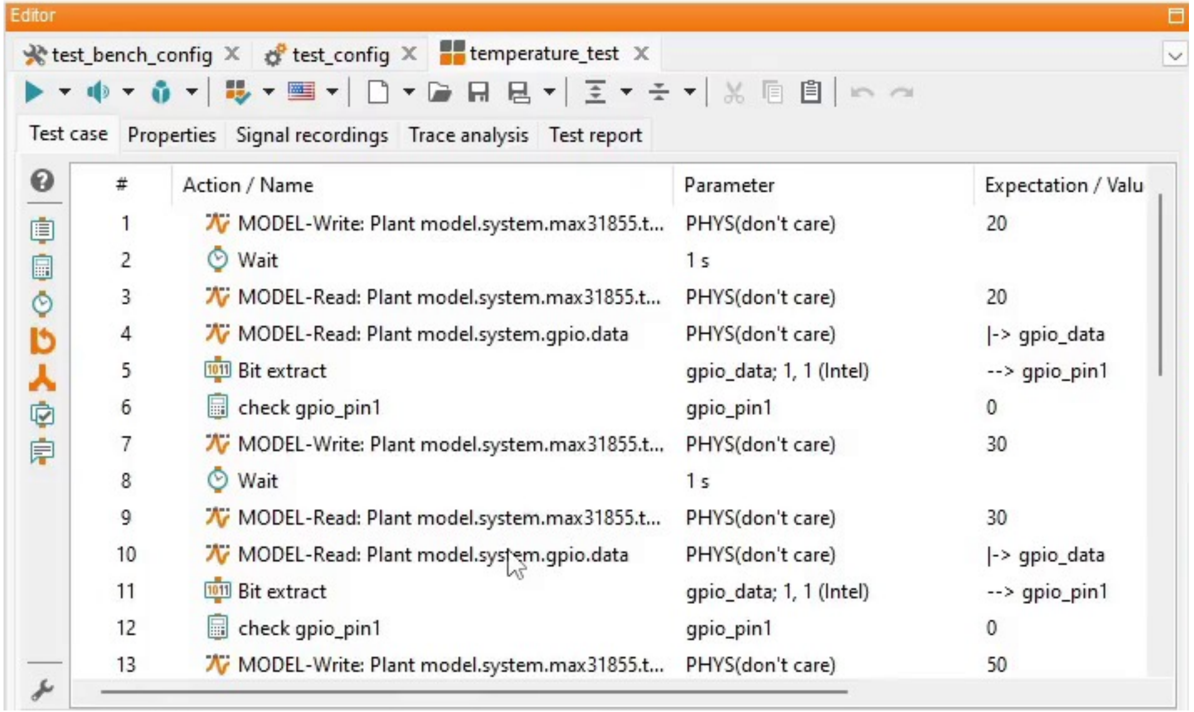
Demo – Configuration




```xml
<fmiModelDescription ... fmiVersion="3.0" modelName="myVP" ...>
    <CoSimulation modelIdentifier="myVP" needsExecutionTool="true"
        canHandleVariableCommunicationStepSize="true" ... />
    <DefaultExperiment startTime="3" stopTime="5" stepSize="0.01" />
    <ModelVariables>
        <Float64 name="time" valueReference="0" causality="independent"
variability="continuous" />
        <Float32 name="system.max31855.temp" valueReference="1" causality="input"
            variability="continuous" start="10.0" />
        <UInt32 name="system.gpio.data" valueReference="2" causality="output"
variability="discrete" />
    </ModelVariables>
    <ModelStructure>
        <InitialUnknown valueReference="1" />
        <Output valueReference="2" />
    </ModelStructure>
    <Annotations>
        <Annotation type="VCML">
            <VP host="localhost" port="8888" executable="resources/vp" ... />
        </Annotation>
    </Annotations>
</fmiModelDescription>
```

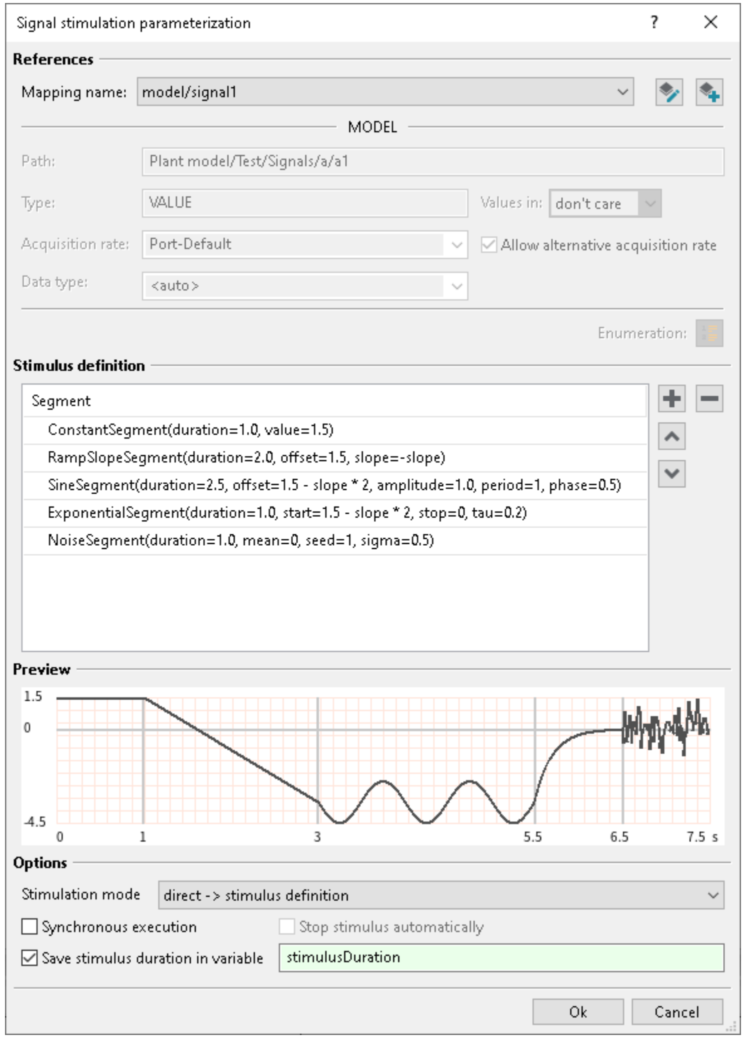
Demo – Test case creation

- Access variables (from FMU)
- Read, write, checks…
- „wait“ == simulation time



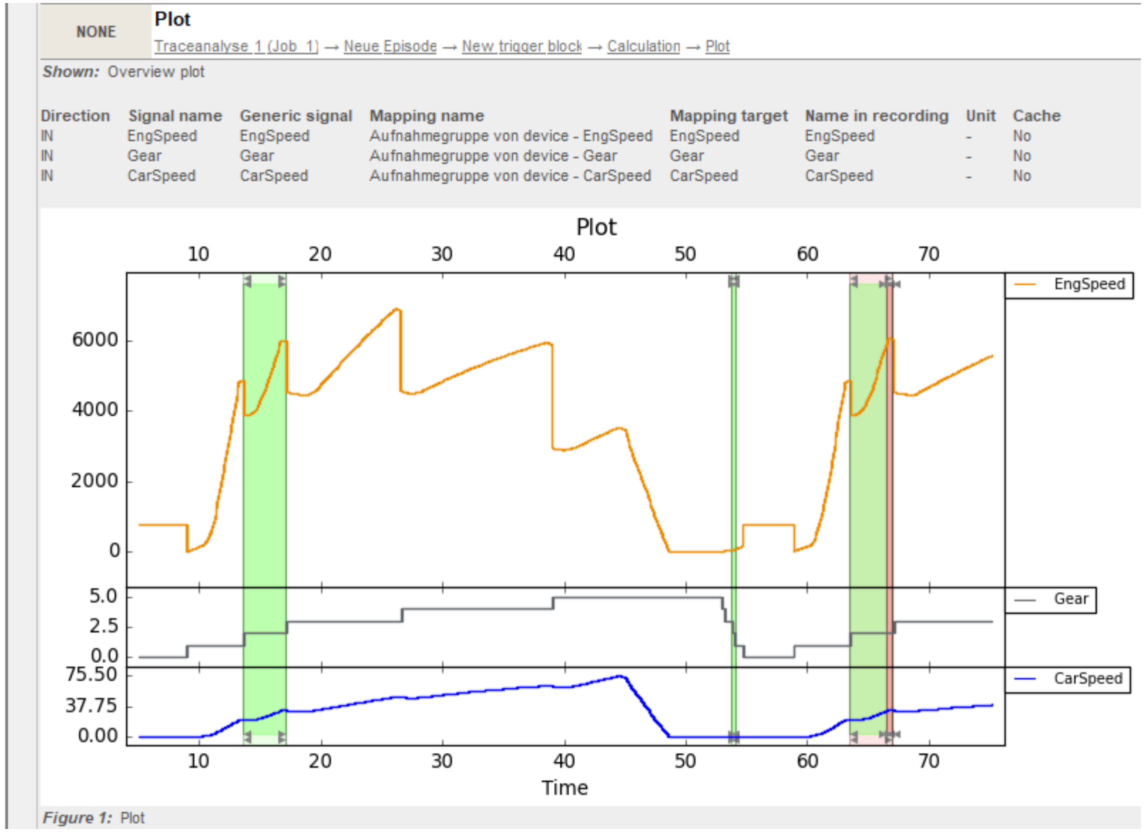
Demo – Stimulation

- Define complex signal inputs
  - e.g. Ramps, sine, noise…
- Used for “open loop testing”



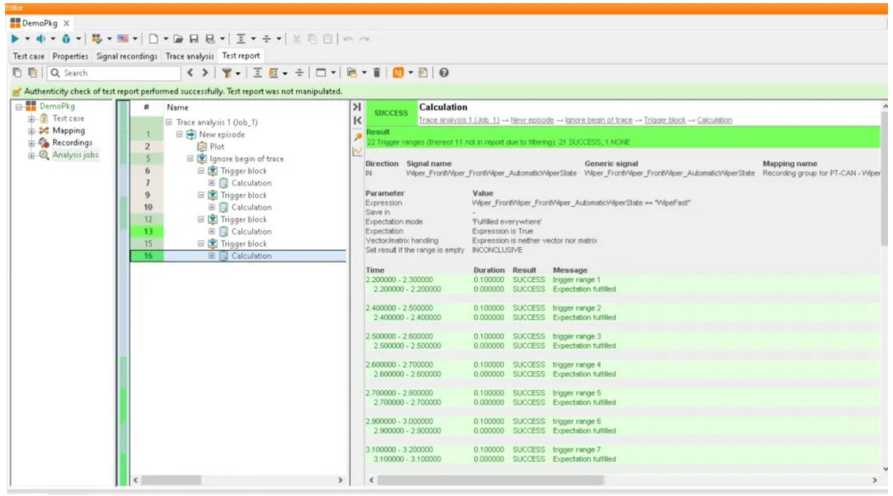
Demo – Trace Analysis

- Verify signals and their behaviour over time
- Define (combined) conditions, triggers…
- Reuse templates

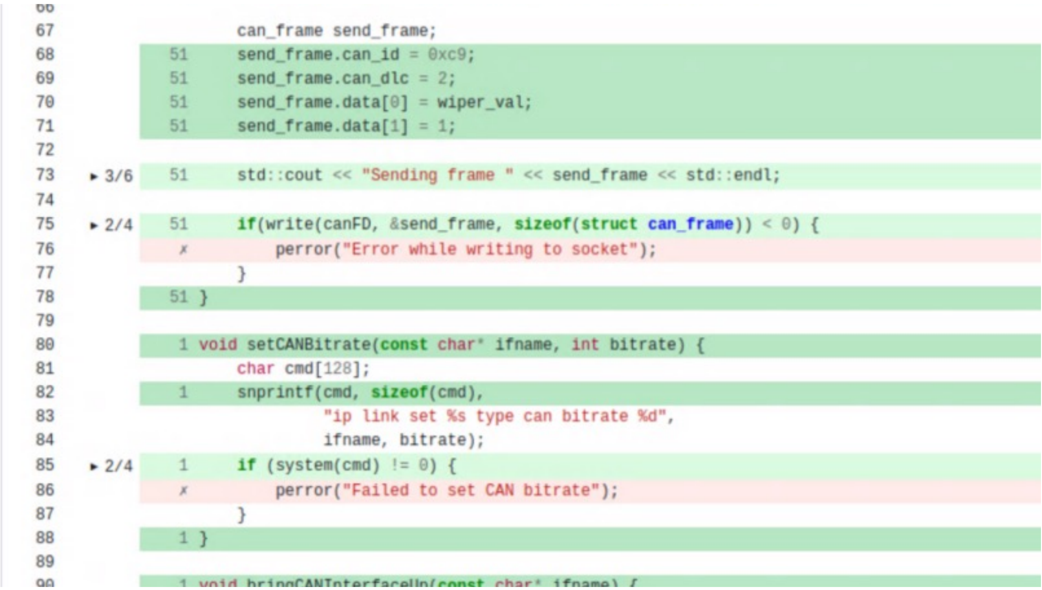


Demo – Reporting and Code Coverage

- Results of execution
- Basis for synchronization with ALMs

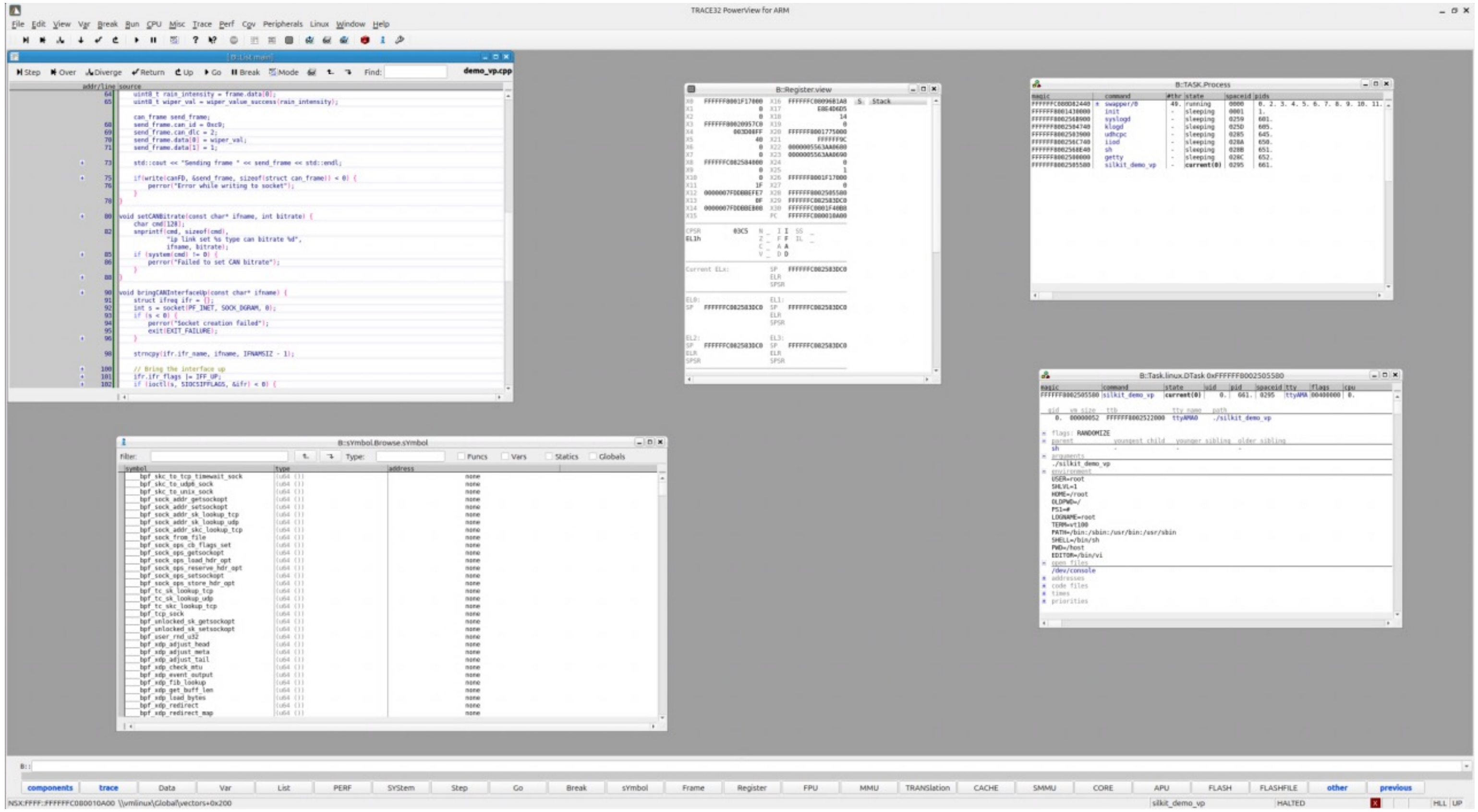


- Create more test cases by analyzing code coverage



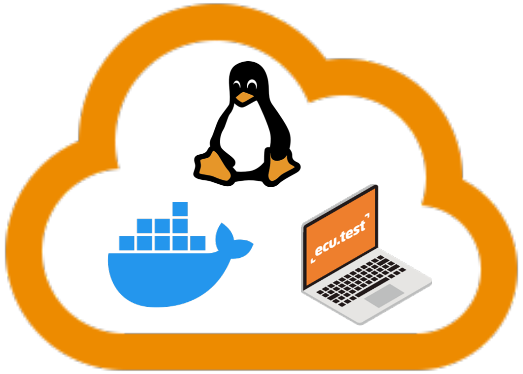
Demo – Debugging

- Attach debugger to test case to find problems in software from within testcase
- Lauterbacher, VS Code…



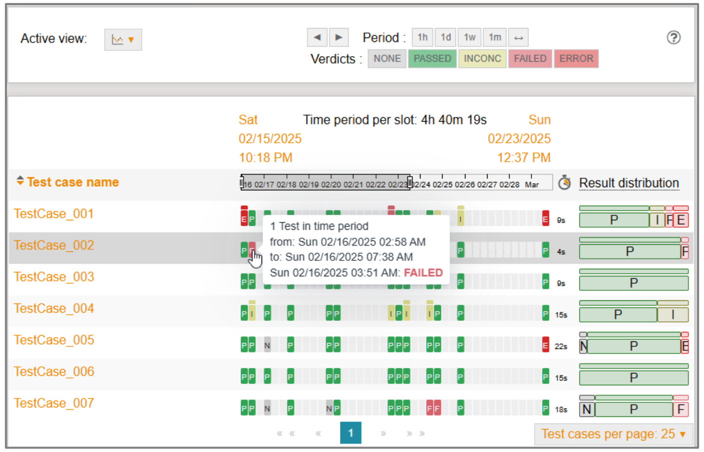
Scaling

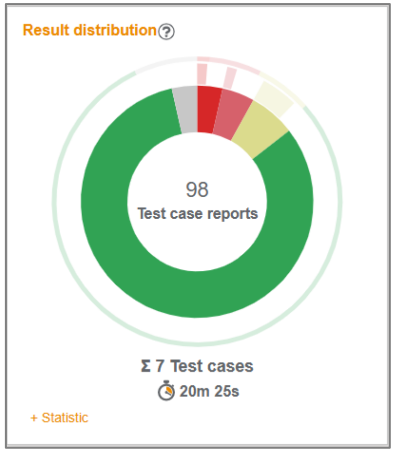
- After automation: Scaling!
- Goal: Scale as much as you can!
  - That's the benefit of virtual platforms → No hardware or other dependencies
  - Needed for all the requirements, variants, code changes…
- APIs and technologies:
  - REST API or command line interface
  - Windows, Linux, Docker…
  - Jenkins, Github Actions, custom code/platform, test.guide…

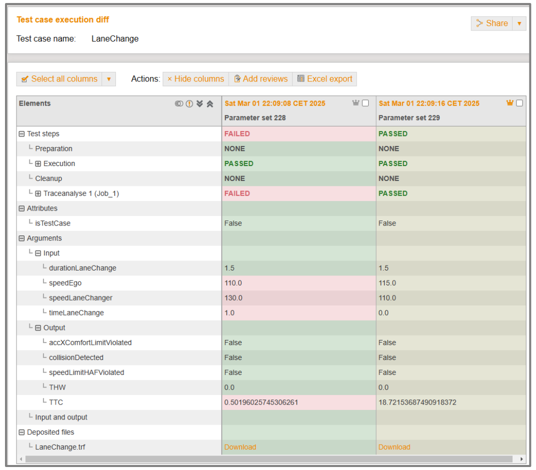


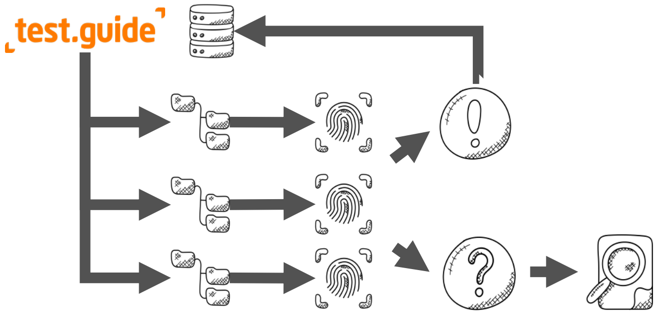
Reviews

- What to do with tons of reports?!
- Overviews (e.g. test.guide):
  - Dashboards
  - Filters
  - Comparisons
- (Automatic) reevaluations
  - E.g. by finger printing
- Address devs AND management!



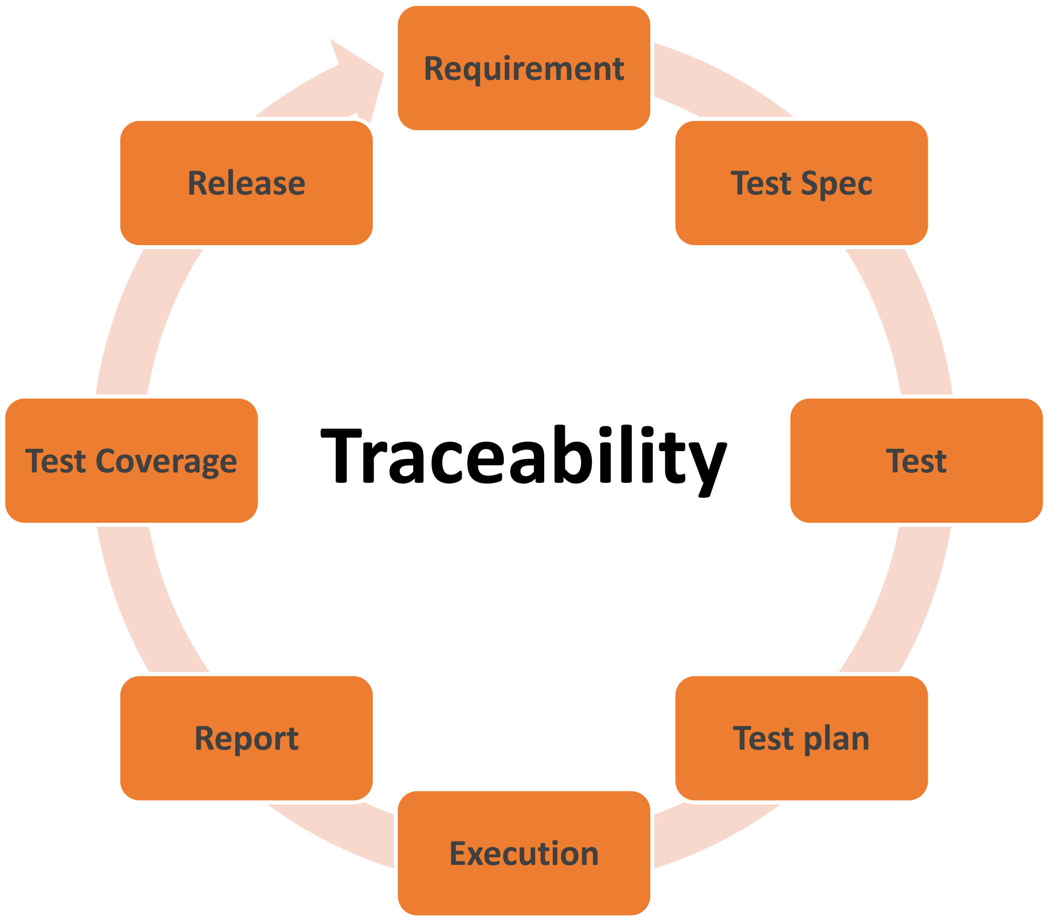






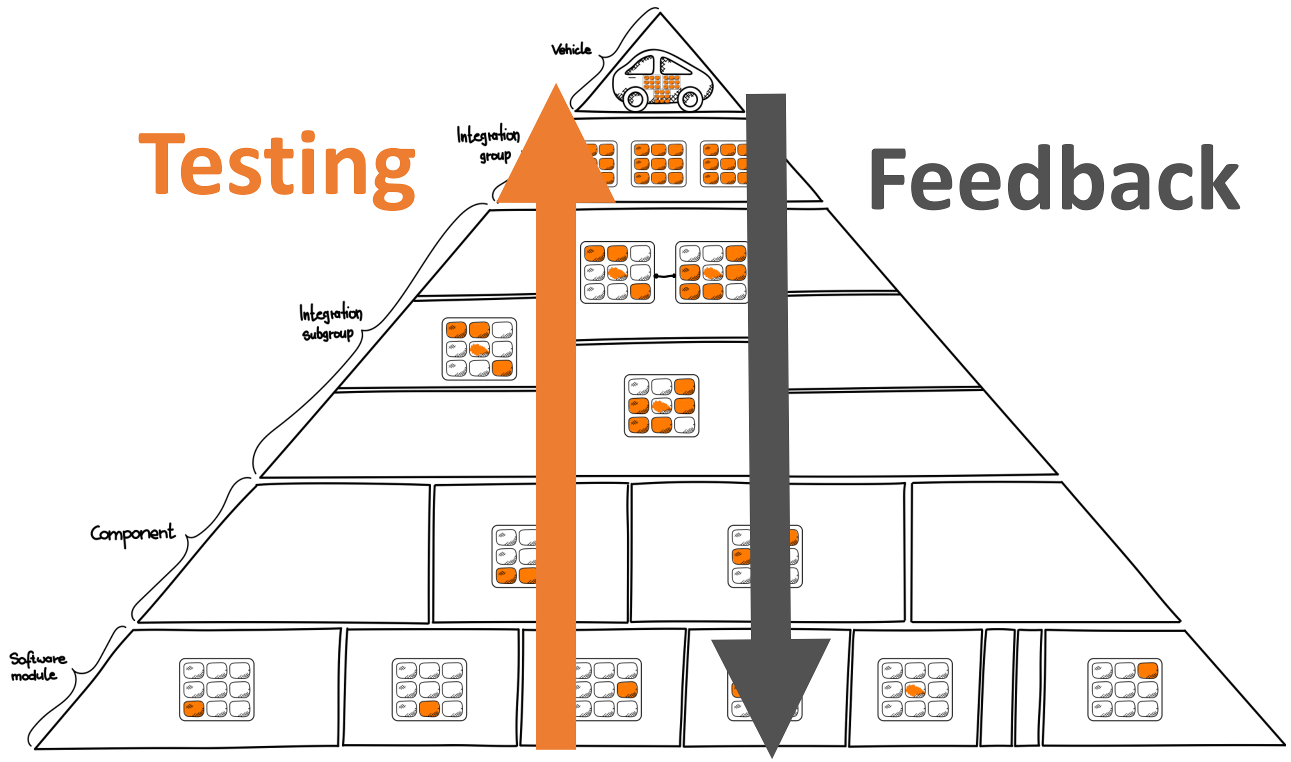
Traceability

- Overview:
  - How good is my requirement tested?
  - Can I release it?
- Requirements:
  - Links between all artifacts
  - Central organized system for reporting with sync to ALMs



Collaboration

- Smart usage of test results
- “Instant” feedback for developers (especially between integration levels)
- Use of stages, branches, custom test sets…
- Github PRs, Jira ticket creation…



Summary

- Complete automation workflow:
  - a single test case on a single VP Linux based test bench
- Scale executions for more variants, platforms, systems...
  - “Do shift left” (bugfixing at the beginning), avoid Big Bang at the end!
- Handling of tons of results with reviews and sync to requirements

→ From (manual) testing to collaborative platforms!

→ From requirement to synced test results **with target compiled code**!

→ Thanks to Virtual Platforms without any hardware!

Outlook

- Native test execution on ARM systems → Performance
- AI based test case creation
- AI based test result reviews and reevaluations
- Reuse test cases with HiL systems or in-vehicle
  - System test with final hardware
  - Using mapping abstraction layer

Questions

Institute for Communication Technologies and Embedded Systems

RWTH AACHEN UNIVERSITY

MACHINEWARE

CISPA

2025
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE

MUNICH, GERMANY
OCTOBER 14-15, 2025

Leveraging SystemC-TLM-based Virtual Prototypes for Embedded Fuzzing

C. Ghinami[1], J. Winzer[1], N. Bosbach[1], L. Jünger[2], S. Wörner[3], and R. Leupers[1]

[1]RWTH Aachen University, Aachen, Germany

[2]MachineWare Gmbh, Aachen, Germany

[3]CISPA Helmotz Center for Information Security, Saarbrücken, Germany

accellera SYSTEMS INITIATIVE

About Us

Chiara Ghinami

- PhD student at Chair for Systems on Silicon (SSS) RWTH Aachen University
- Embedded Software Testing using VPs

Rainer Leupers

- Head of the chair

~ 10 researchers

R&D Topics:

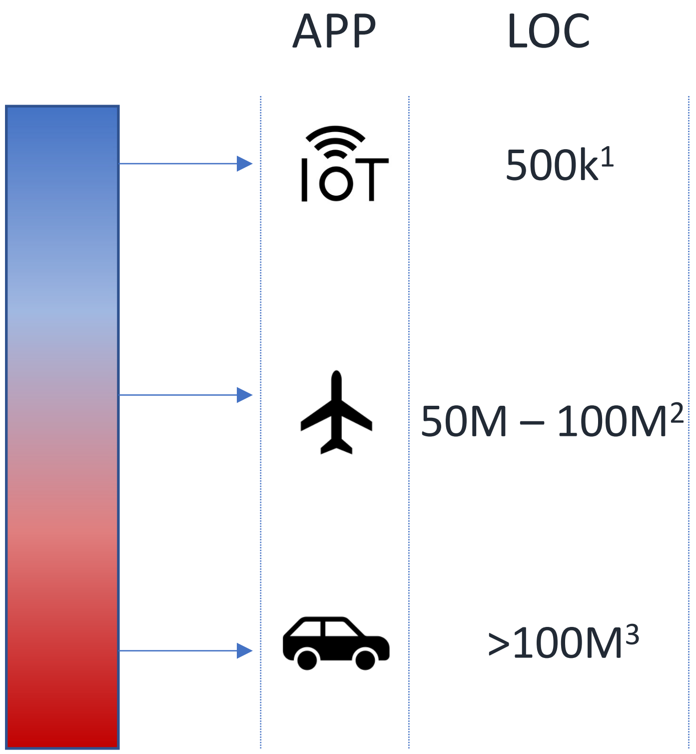
- Virtual Prototypes
- AI for Edge Computing
- Neuromorphic Chips

Agenda

- Automatic Software Testing
- Hardware vs VP Fuzzing
- CAN: Case Study
- Preliminary Results
- Final Remarks

Embedded Systems

When Complexity Gets Out of Hand



[1]The State of IoT Software Development, Memfault Report, 2024
[2]Is Digital Thread/Digital Twin affortable?, Complex Adaptive Systems Conference, 2017
[3]Why car companies are hiring computer security experts, The NY Times, 2017

VW burns €2 billion as software losses mount, group profits plunge

All the operating profits of Audi, Bentley, Ducati, and Lamborghini combined burned on software losses.

Ford scraps key software program after $10 billion in losses

CAN Injection: keyless car theft

Ford recalls nearly 1.1 million vehicles over rearview camera software issue

This is a detective story about how a car was stolen - and how it uncovered an epidemic of high-tech car theft.

'Full of bugs': how the world's biggest carmakers fell behind in software

Software-defined vehicles are forecast to become dominant in all regions

Mitsubishi has recalled nearly 200,000 vehicles due to a software issue that causes the rearview camera to freeze, increasing the risk of a crash

Exclusive: My Volvo Lost Brakes And Crashed After Software Update

Terrifying moments caught on camera as this XC90 lost braking while it descended a mountain road

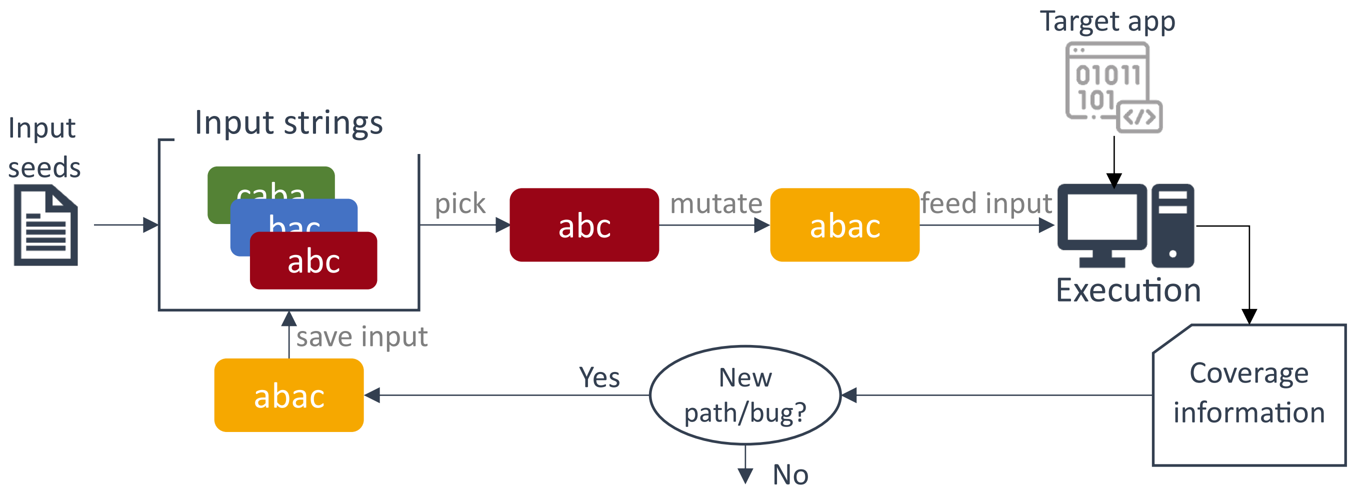
Automatic Software Testing

Wish List:

1. Automation ✓
2. Speed ✓
3. Reliability ✓
4. Usability ✓
5. Clear Reports ✓

Fuzzer:

1. Only seeds needed
2. Lightweight
3. Deterministic
4. Plug-and-play
5. GUI with coverage, crashes



Already used for Desktop APPs

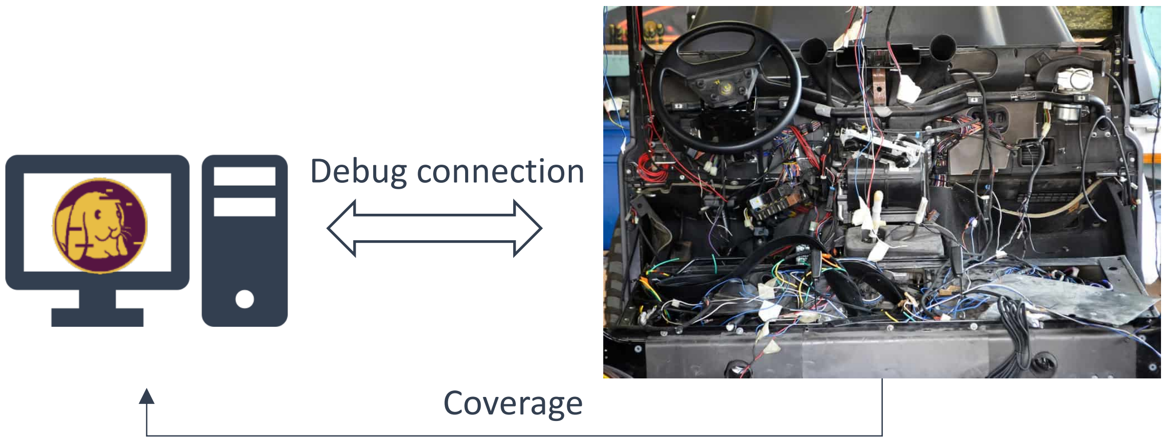
Google: AFL++, OSS-Fuzz
Microsoft: OneFuzz
Apple: CrossFire, KextFuzz, Hyperpom
Amazon: SnapChange

But what about Embedded Systems…

Hardware vs VP Fuzzing

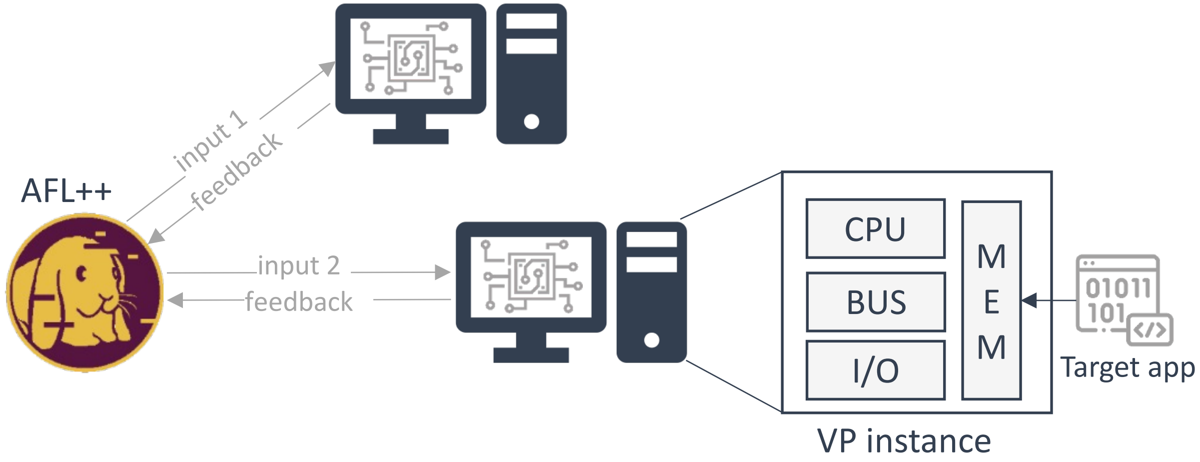
On-board testing

- High execution **fidelity**
- Difficult to **debug**
- Difficult to **scale**



VP-based testing

- **Faster execution** → faster bug detection
- Easy **introspection** & **tracing**
- **Scalable**
- Early Testing



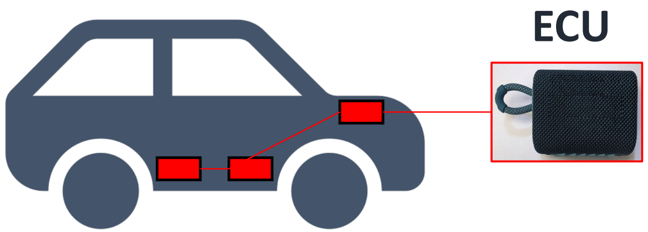
Case of Study: CAN

What is CAN?

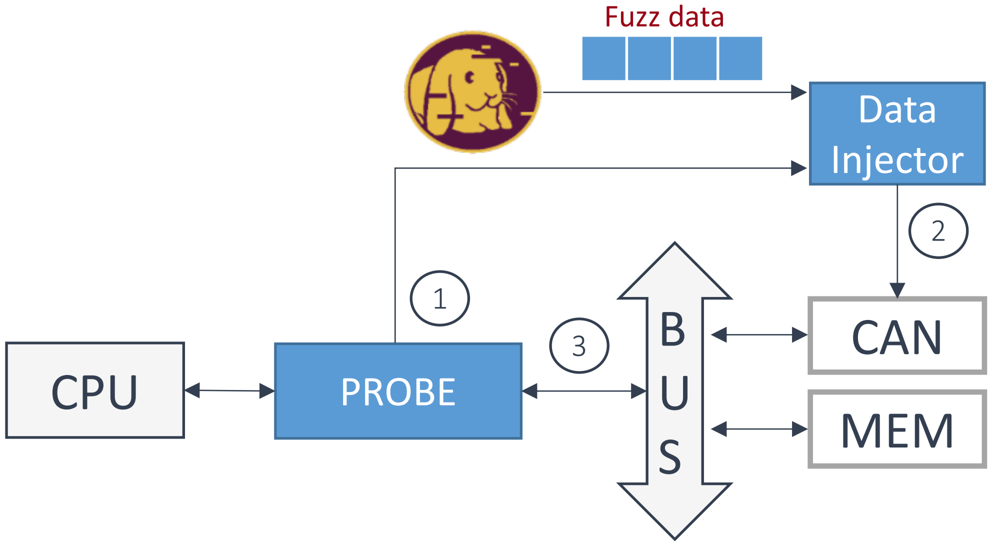
“The CAN is a communication protocol designed specifically for the automotive. It enables the communication between ECUs.”

Why CAN?

How to Get Away With Car Theft: Unveiling the Dark Side of the CAN Bus



Reproduced Set-up:



1 Data read access intercepted

2 Fuzzer data injection

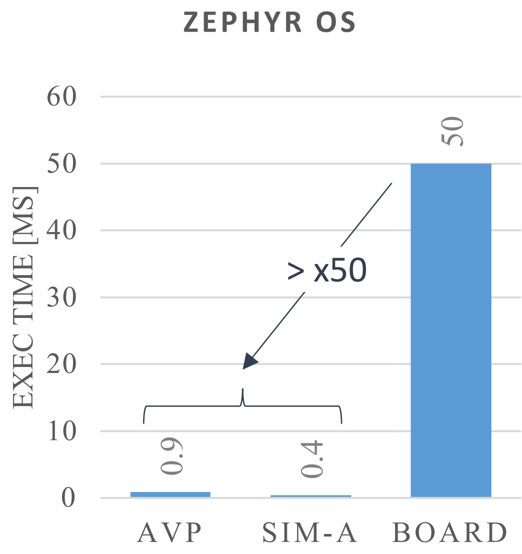
3 Read transaction takes place

Preliminary Results

Can we execute real-word workloads?

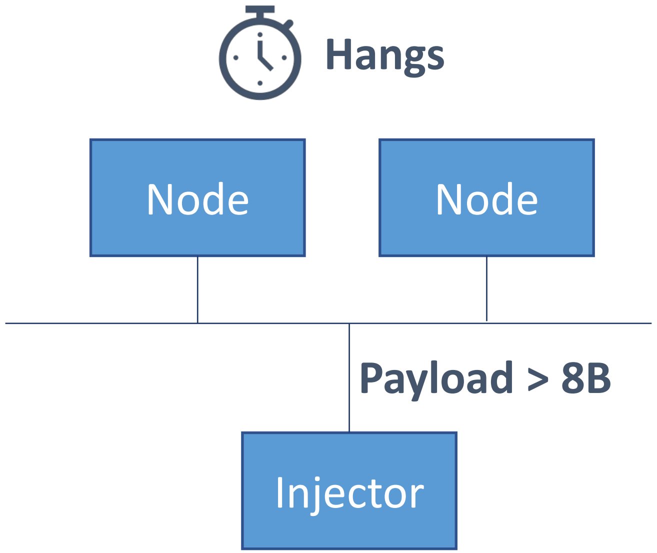
- Buildroot (Linux)
- Drone firmware
- Robot (on its way)

Are we faster than physical hardware?



Lower is better!

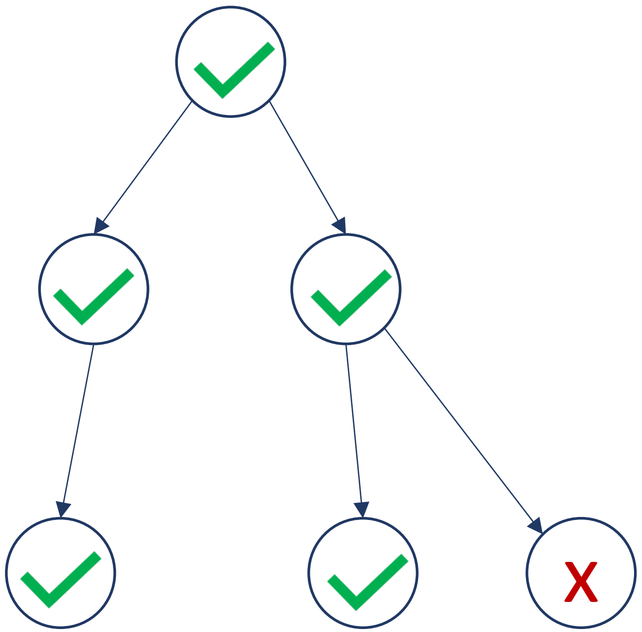
Can we unveil bugs?



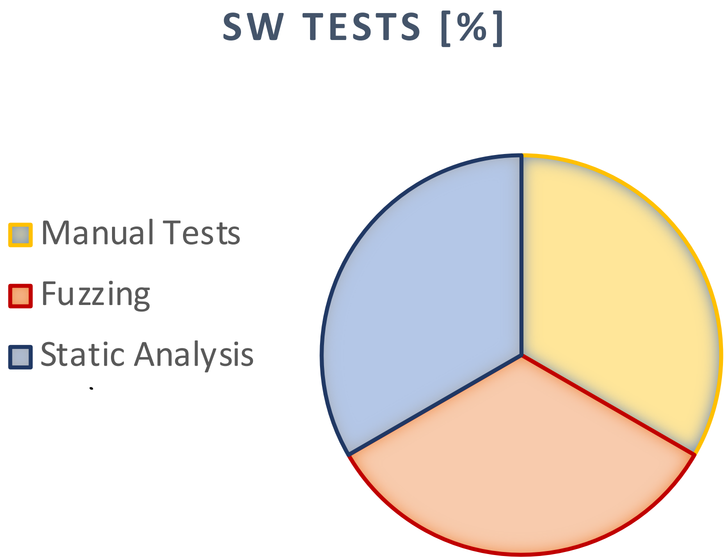
Final Remarks

Fuzzer is not bulletproof…

Code Coverage ≠ 100%

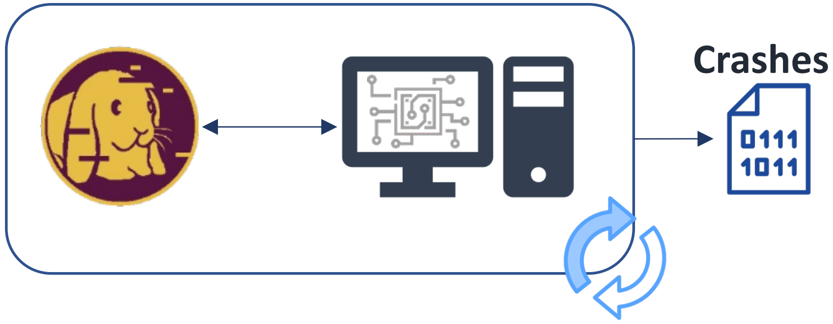


…But it can still help

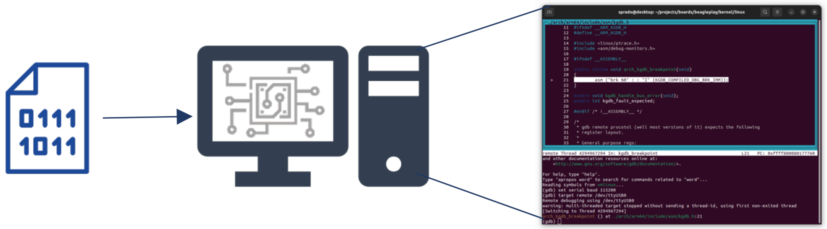


Post-fuzzing analysis is important

1. Fuzzing Loop



2. Source Code Inspection



Conclusion

There is now a fast, flexible and easy-to-use embedded fuzzing option for AFL++

Performance:
- Faster than physical board

Early Testing:
- Software testing ahead of hardware availability

Flexibility:
- Extensive peripheral support

Usability:
- Integrated debugging option
- Easier testing and tracing

Thank you for your attention!