

Software-Defined Vehicle Realization with Model-Based Design and FEV's Gateway

Dr. Hamzeh Alzubi, FEV North America

- Intelligent mobility and software manager.
- Leads the design, development, and implementation of software programs and applications.
- Over 15 years of professional experience in software development for aerial and ground vehicles.
- Ph.D. from Oakland University. His Ph.D. research project was the first-place winner at the UAE Drones for Good competition in 2016.





Software-Defined Vehicle Realization with Model-Based Design and FEV's Gateway

Hamzeh Alzubi, FEV.io



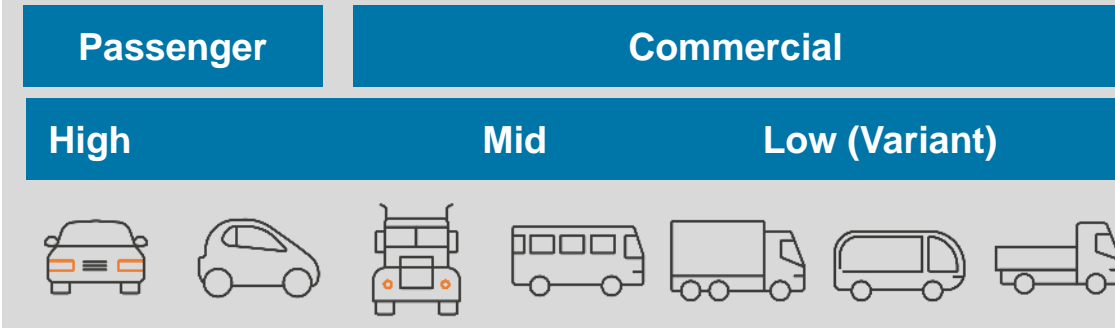
MathWorks
**AUTOMOTIVE
CONFERENCE 2024**

AGENDA

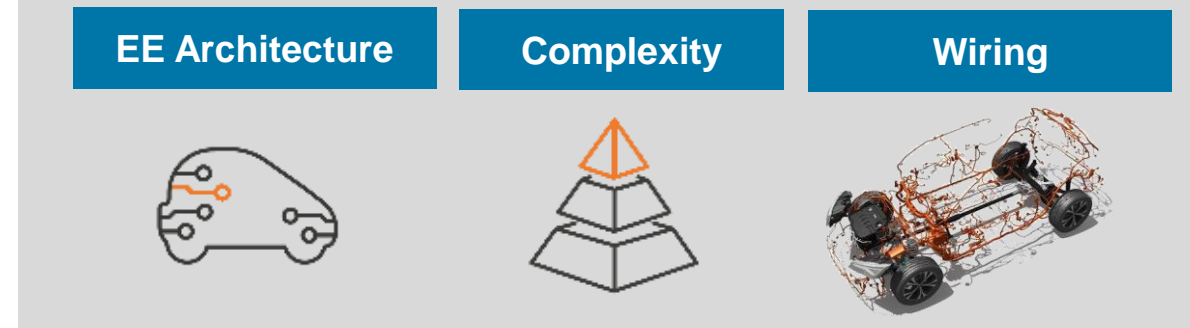
- **Introduction**
- Software Defined Vehicle (SDV) Motivation
- FEV.io's SDV HPC Software Development
- Applications development for SDV with MathWorks tools
- SDV HPC testing on an existing EE architecture using FEV.io's gateway
- FEV.io: full service SDV development

An overview of growing challenges within the automotive industry

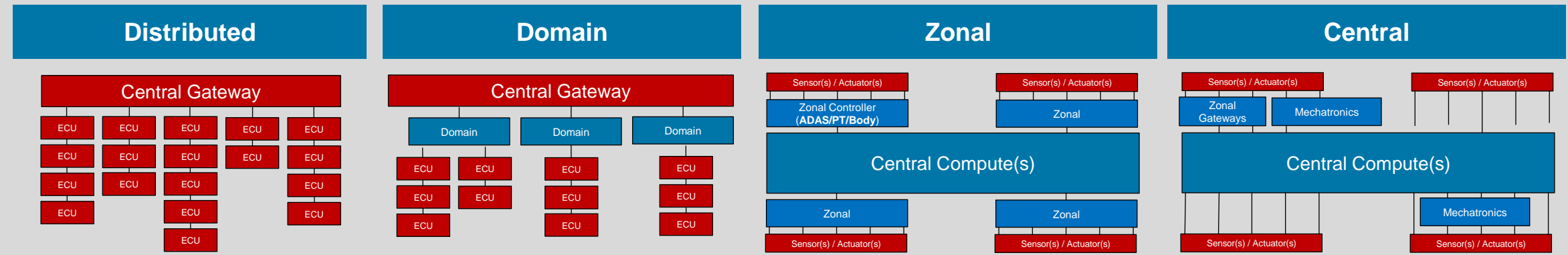
WIDE RANGE OF MOBILITY



GROWING INFLUENCERS



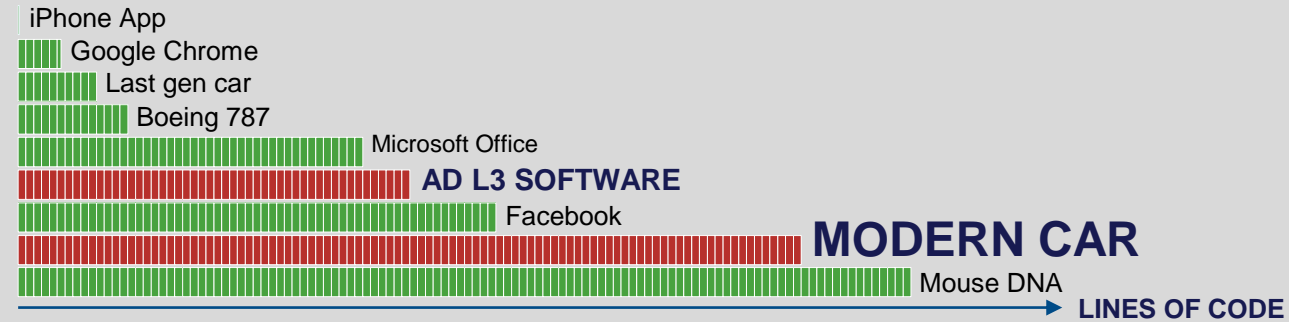
EVOLVING EE ARCHITECTURE



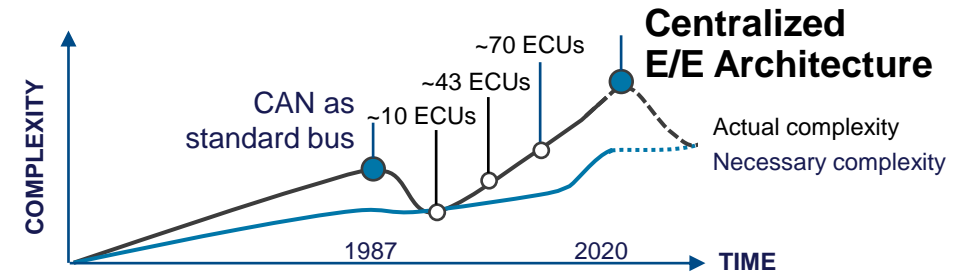
One SOLUTION for all multi-dimensional requirements **»» SOFTWARE DEFINED VEHICLE**

CAV related services are disrupting product development and are strongly pushing towards redefined E/E and SW architectures

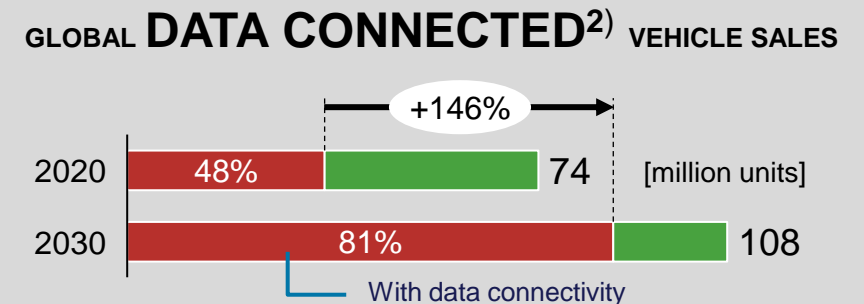
150 million LINES OF CODE
 TYPICALLY CONTAINED BY A MODERN CAR
 (50 MILLION LINES REQUIRED FOR AD L3 AD¹) SYSTEM)



100 ELECTRONIC CONTROL UNITS (ECU)
 IN CONVENTIONAL CARS ARE INCREASING COMPLEXITY
 OF CURRENT E/E ARCHITECTURES



>1,000 GB OF DATA
 PER VEHICLE AND DAY EXPECTED
 TO BE GENERATED BY 2030



1) L3 AD = SAE level 3 Automated Driving; 2) Direct connection to OEM backend via at least 2G/2G/4G connectivity; CAV: connected, automated vehicle

AGENDA

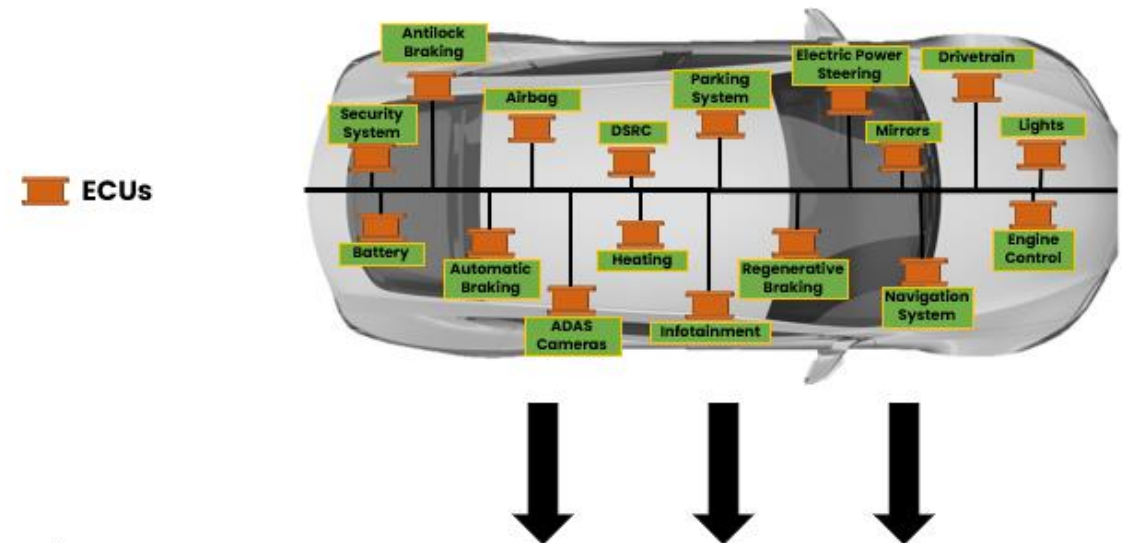
- Introduction
- **Software Defined Vehicle (SDV) Motivation**
- FEV.io's SDV HPC Software Development
- Applications development for SDV with MathWorks tools
- SDV HPC testing on an existing EE architecture using FEV.io's gateway
- FEV.io: full service SDV development

Software Defined Vehicle (SDV) Motivation

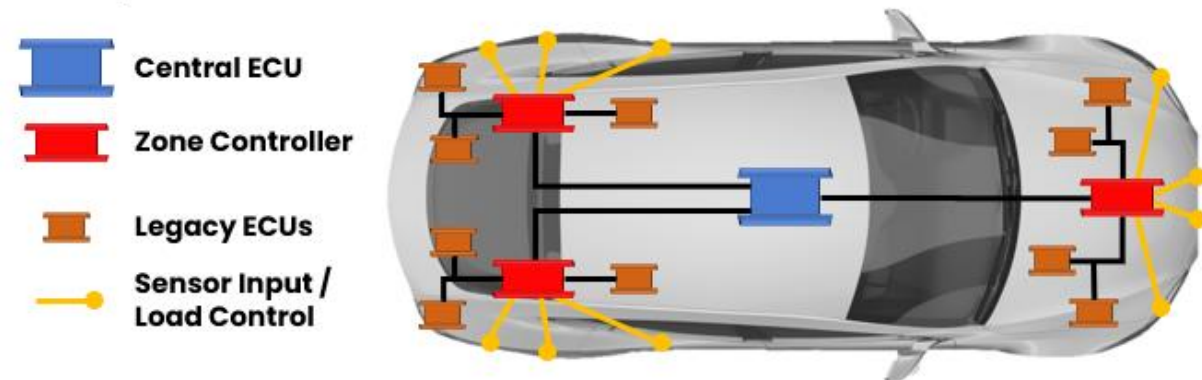
Example highlights of SDV motivation:

- Increasing software complexity
- New trends in the automotive market such as electrification, autonomous driving and connectivity to meet customers' expectations
- Reduced number of ECUs
- Decoupling of hardware from software
- Improved vehicle life cycle management
- Over-the-Air (OTA) software updates
- Enabling new ways to interact with vehicle users
- SDV provides the following benefits:
 - Lower costs
 - Less weight
 - Faster time to market
 - Increased security

Conventional Vehicle E/E Architecture



SDV E/E Architecture



AGENDA

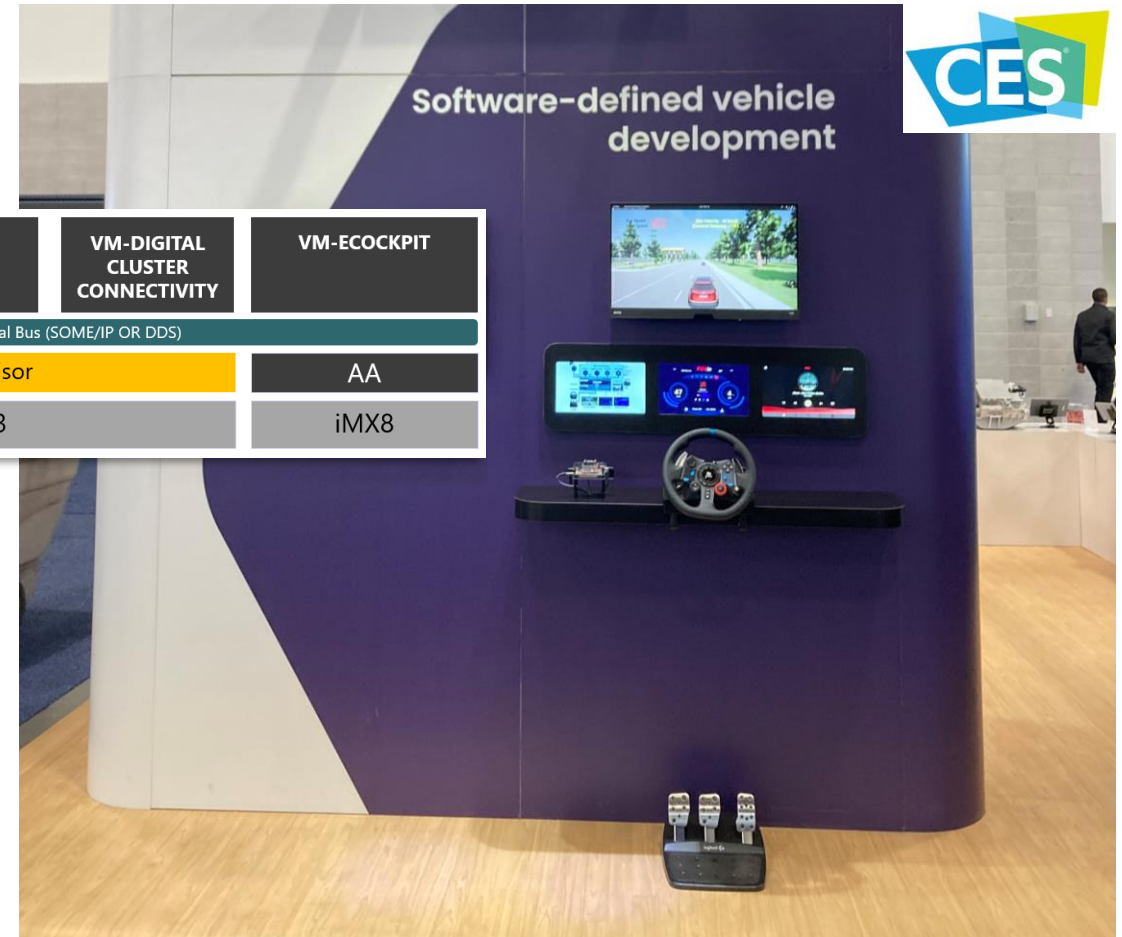
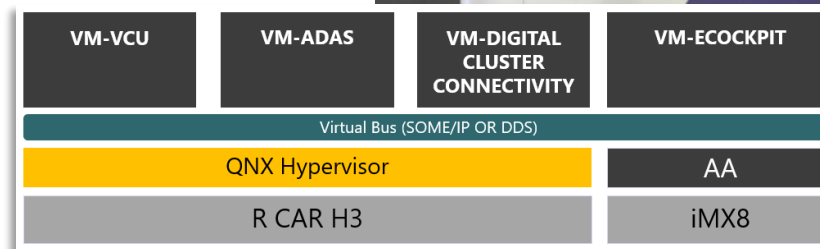
- Introduction
- Software Defined Vehicle (SDV) Motivation
- **FEV.io's SDV HPC Software Development**
- Applications development for SDV with MathWorks tools
- SDV HPC testing on an existing EE architecture using FEV.io's gateway
- FEV.io: full service SDV development

FEV.io provides cutting edge EE architecture and software solutions to address the challenges of transitioning to SDV

FEV.io demonstrator of cross-domain Central Controller for Software Defined Vehicles

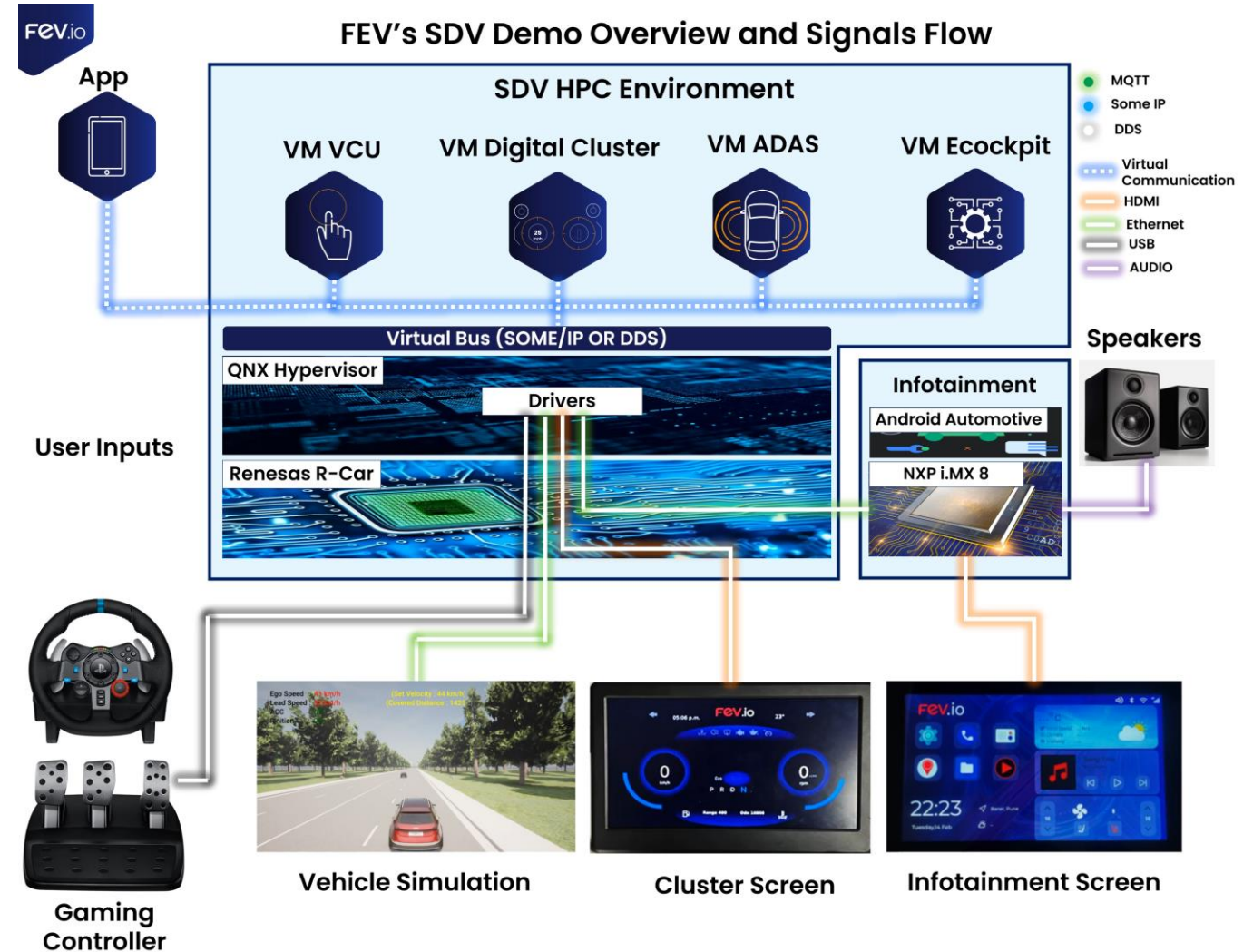
Scalable, modular demonstrator to test, improve and showcase the project objectives

- Demonstrate mix criticality load on HPC
- Redesign legacy functions into SOA Architecture
- Create virtual bus for maximal abstraction of SW



FEV.io SDV demo architecture with DDS, SOME/IP, and MQTT

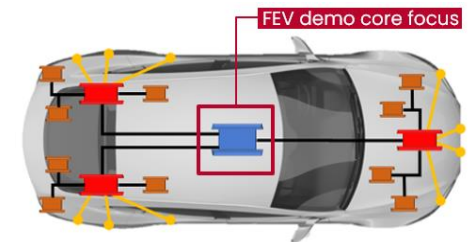
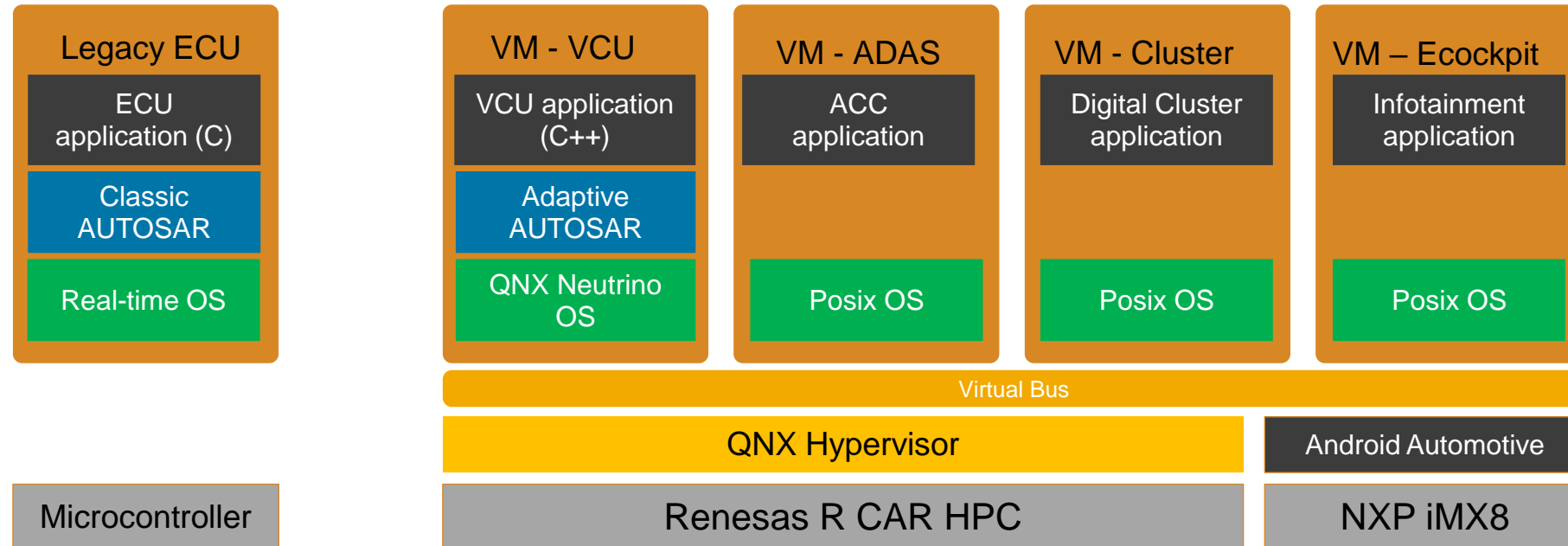
SDV demo video



AGENDA

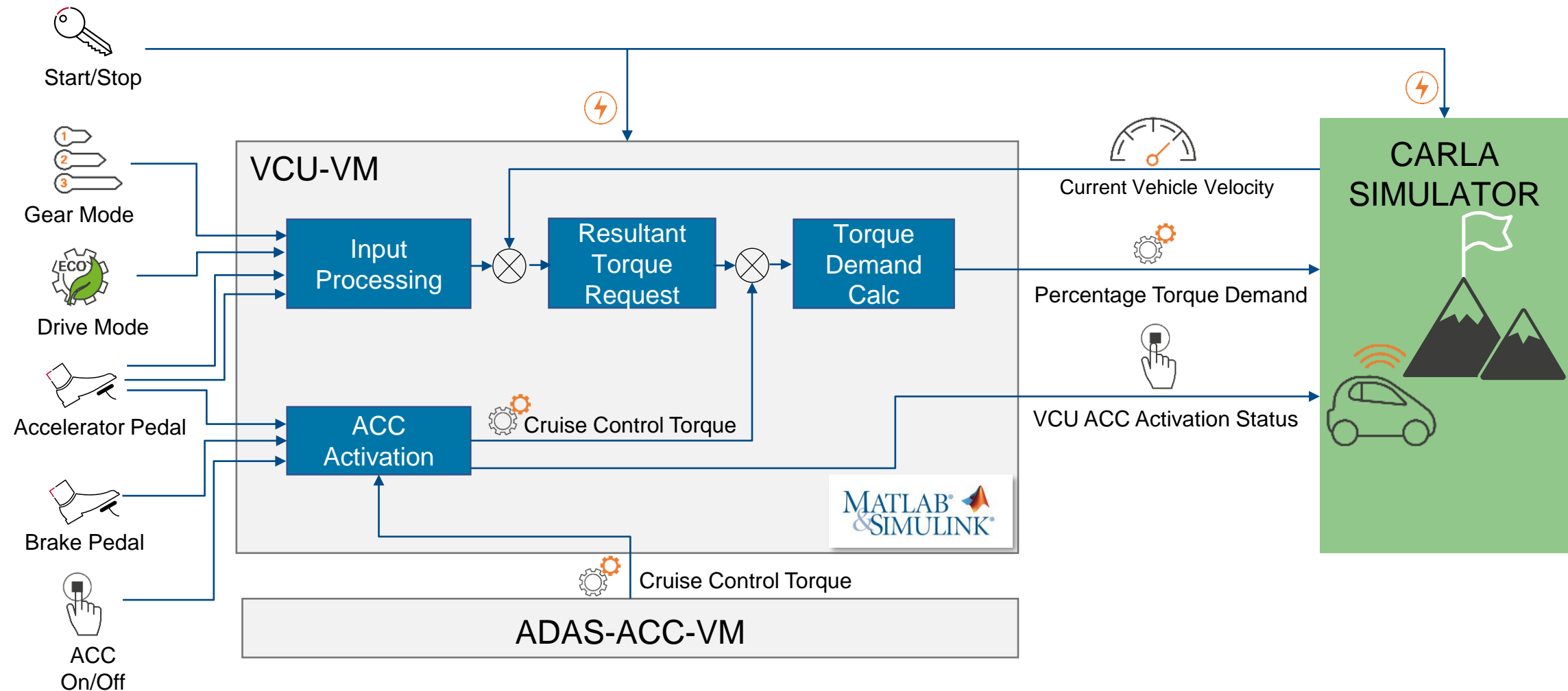
- Introduction
- Software Defined Vehicle (SDV) Motivation
- FEV.io's SDV HPC Software Development
- **Applications development for SDV with MathWorks tools**
- SDV HPC testing on an existing EE architecture using FEV.io's gateway
- FEV.io: full service SDV development

Applications development for SDV with MathWorks tools



- Central ECU
- Zone Controller
- Legacy ECUs
- Sensor Input / Load Control

Migration of Legacy Functions to Service Oriented Architecture (SOA): E-Drive Torque Path Arbitration



Integration of vehicle function in VCU Cluster on HPC with Autosar Adaptive for SOA architectural design



Legacy to SOA architecture approach:

SYSTEM DESIGN

- Use Cases elaboration
- Features definition
- Requirements derivation
- System design

SOFTWARE DESIGN

- Service Architecture Definition
- Application Design in MATLAB/Simulink
- Signal Modeling including timing req. analysis
- Deployment configuration of Autosar Adaptive stack

INTEGRATION:

- Application implementation
- Integration with Vector Adaptive Stack
- Deployment of QNX OS and Hypervisor
- Set up of HIL Demo

Realizing the Legacy Functions as Adaptive SWC in Simulink

1

2

The image displays two screenshots of the Simulink environment. The first screenshot (labeled '1') shows the Simulink interface with the TorqPathApp model open. The 'Model Browser' on the left lists 'Simulink Test' and 'Code Inspector'. The 'Code Generation' section shows 'AUTOSAR Component Designer - Create AUTOSAR component'. The second screenshot (labeled '2') shows the 'AUTOSAR Dictionary: TorqPathApp' tool. The left pane shows the package structure, including 'AdaptiveApplications', 'Service Interfaces', and various SI packages like 'SI_AccActivation', 'SI_AccelPedal', 'SI_BrakePedal', 'SI_CompVehData', 'SI_CruiseCtrlTq', 'SI_CurrentVehData', 'SI_DriveMode', 'SI_GearMode', 'SI_KeyIgnSts', 'SI_NewVehData', and 'Persistence Key Value Interfaces'. The right pane shows 'View and Edit XML Options' with various settings like 'XML Options Source' (Inlined), 'Exported XML File Packaging' (Modular), 'Datatype Package' (/DataTypes/ImplementationDataTypes), and 'Interface Package' (/ServiceInterfaces). The 'Additional Options' section shows 'ImplementationDataType References' set to 'Allowed' and 'Identify Service Instance Using' set to 'InstanceSpecifier'.

Adaptive AUTOSAR compliant C++ Code Generation with Embedded Coder

3

PathApp - Simulink

Code for: TorqPathApp

Generate Code

SELECT OUTPUT

AUTOSAR Adaptive

AUTOSAR Adaptive - AUTOSAR-compliant C++ code (autosar_adaptive.tlc)

SIMULATION

Simulation Only

Model intended for simulation only.

Select System Target File...

Code Mappings - Component Interface

4

PathApp - Simulink

Code for: TorqPathApp

Generate Code

Configuration Parameters: TorqPathApp/QuickStart_50023_1_31_20_48_37302 (Active)

Search

Solver

Data Import/Export

Math and Data Types

Diagnosics

Hardware Implementation

Model Referencing

Simulation Target

Code Generation

Optimization

Report

Comments

Identifiers

Custom Code

Interface

Code Style

Verification

Templates

Code Placement

Data Type Replacement

AUTOSAR Code Generat...

Coverage

Target selection

System target file: autosar_adaptive.tlc

Description: AUTOSAR Adaptive

Shared coder dictionary: <empty>

Language: C++

Language standard: C++11 (ISO)

Build process

Generate code only

Package code and artifacts

Toolchain: AUTOSAR Adaptive | CMake

Build configuration: Faster Builds

Code generation objectives

Prioritized objectives: Execution efficiency, RAM efficiency

Check model before generating code: Off

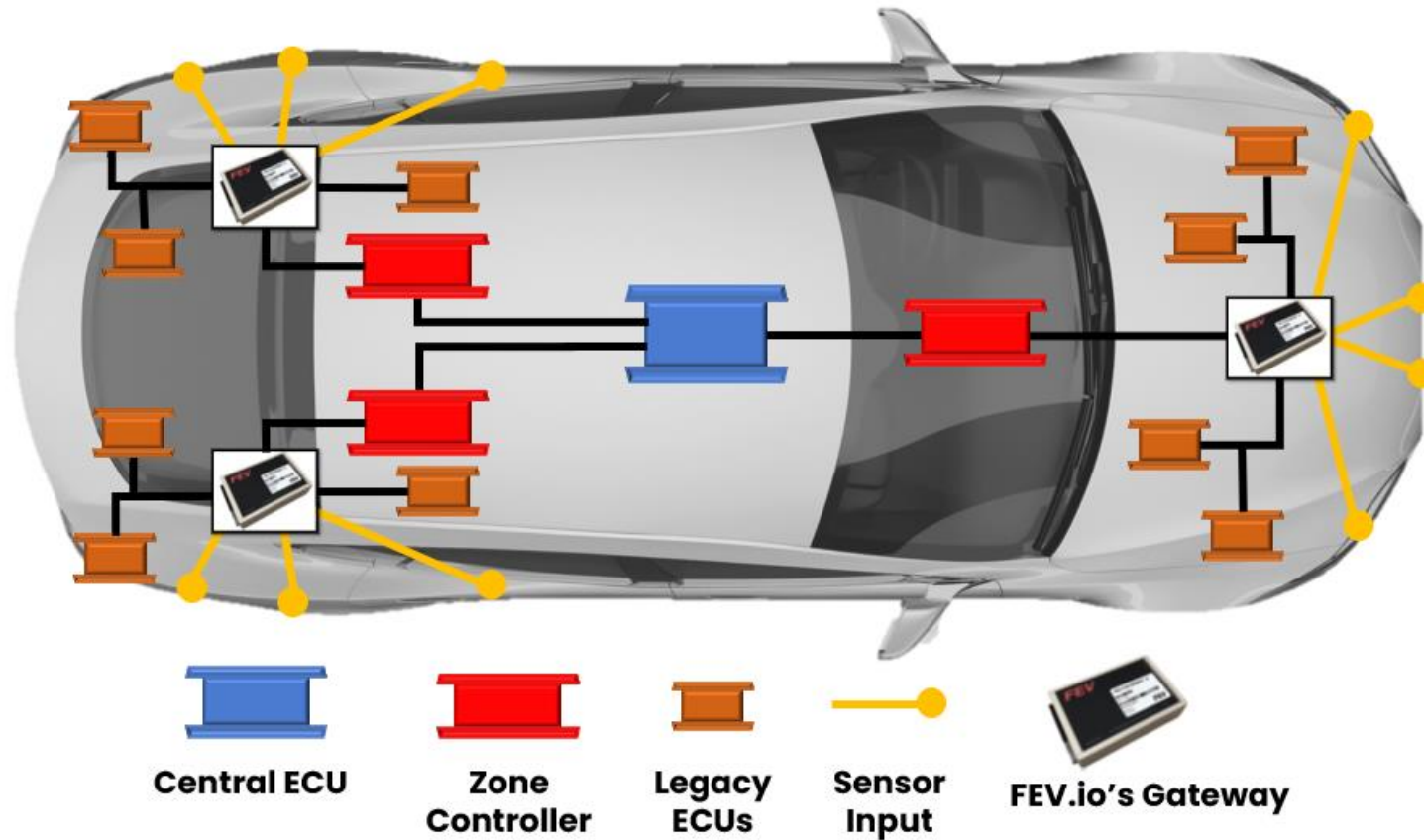
Advanced parameters

Built-in FFTW library callback

AGENDA

- Introduction
- Software Defined Vehicle (SDV) Motivation
- FEV.io's SDV HPC Software Development
- Applications development for SDV with MathWorks tools
- **SDV HPC testing on an existing EE architecture using FEV.io's gateway**
- FEV.io: full service SDV development

FEV.io's gateway enables SDV HPC and Zonal controller integration and testing on an existing OEM EE architecture



AGENDA

- Introduction
- Software Defined Vehicle (SDV) Motivation
- FEV.io's SDV HPC Software Development
- Applications development for SDV with MathWorks tools
- SDV HPC testing on an existing EE architecture using FEV.io's gateway
- **FEV.io: full service SDV development**

FEV.io helps customers to transform traditional signal-based ECUs to Service Oriented implementation



FEV VALUE PROPOSITION

- ✓ Function IP & Experience
- ✓ Domain Expertise
- ✓ SOA Experience
- ✓ FUSA & Cysec Experience



Domain Experts



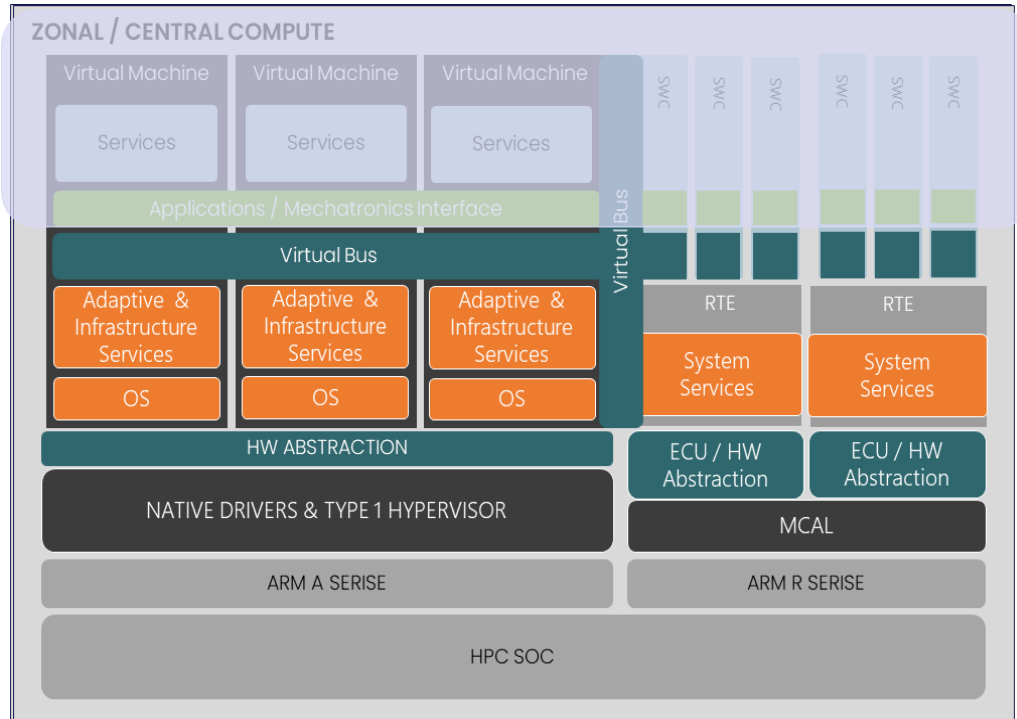
SOA Architects



System Architects

- ✓ Multiple HPC Experience
- ✓ FEV IP
- ✓ Classic & Adaptive AUTOSAR
- ✓ Zonal & Ethernet Experience

FEV.io is building a SDV platform utilizing Model-Based tools and supporting customers on SDV specific topics below



- HPC Board bring up
- HAL & BSP Porting
- Hypervisor porting
- FEV Virtual Bus IP (Porting services)
- Platform Partitioning for Zonal / Central compute
- Classic & Adaptive Platform bring-up
- SDV Platform Development & Maintenance
- Functional Safety & Cybersecurity Development



Qualcomm

RENESAS



FEV VALUE PROPOSITION

- ✓ Hypervisor Experience
- ✓ Linux & QNX Experience
- ✓ System Development



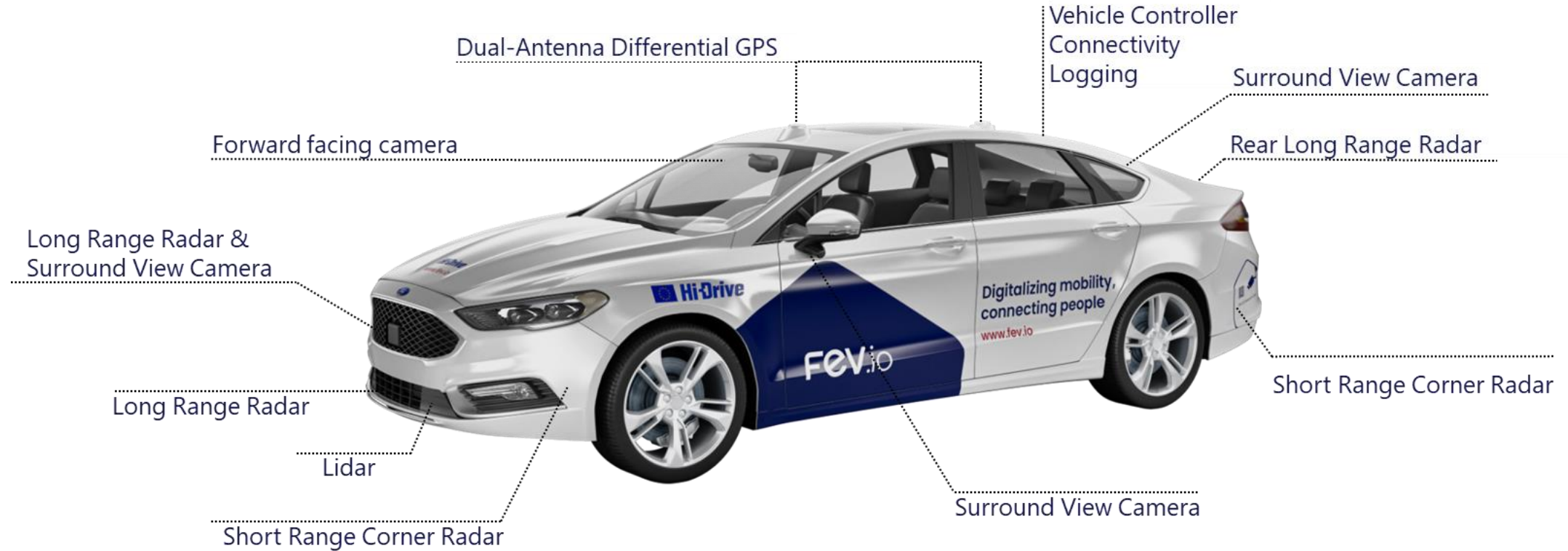
Platform Experts



System Architects

- ✓ FEV IP (Virtual Bus)
- ✓ Classic & Adaptive AUTOSAR
- ✓ Zonal & Ethernet Experience

FEV.io Smart Vehicle Demonstrator Becoming a SDV Development Platform



Additional Cameras (8x, for logging purposes only)
(Surround View, IC, Driver/Co-Driver, Steering Wheel, Driving Pedals)

Conclusion

- Automotive EE architecture is evolving to accommodate the continuous addition of complex features
- SDV supports the successful deployment of emerging technologies such as autonomous driving, connectivity, and electrification
- Transitioning to a SDV environment can be done efficiently with maximum use of legacy software
- FEV.io's gateway has proven to be a key enabler for SDV development and testing
- MathWorks provides the needed tools and support to migrate MATLAB/Simulink based legacy software to application layer of SDV environment
- FEV.io can be your development partner in emerging technologies like SDV

MathWorks
**AUTOMOTIVE
CONFERENCE 2024**
North America

Thank you



© 2024 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.