From Software to Software Systems – New Megatrend: Domain Controller

Elektrobit

Jan Rüdiger
2015-08-21

Lauterbach Expert Forum – August 2015

Agenda

- Why?
  - Complexity as a Challenge
  - Path to Domain Controller
- How
  - Possible Set-up for a Domain Controller (Case Study)
  - Example for Domain Controller

Complexity as a Challenge

Complexity as the Challenge

- In the 1940‘ies the US was abuzz with energy to turn the post-war economy to civil use and „do it right“ from scratch. Example: chemical process industry.
- Economy of scale: *„bigger is better, because bigger is cheaper“*.

- However, the bigger the plant, the:
  - More places there were for failure to occur **(number-of-failures)**
  - Harder it was to pinpoint the cause of failure **(time-to-pinpoint)**
  - Harder it was to fix a fault without producing side effects **(time-to-fix)**
  - Greater the loss in productivity each time the plant shut down **(lost-opportunity-cost)**

- These factors multiplied together to raise the cost of scale beyond the economies of scale. **Construction of the largest plants was abandoned.**

- **All these factors are non-linear in nature! Complexity control is crucial.**

The Software is the Function!

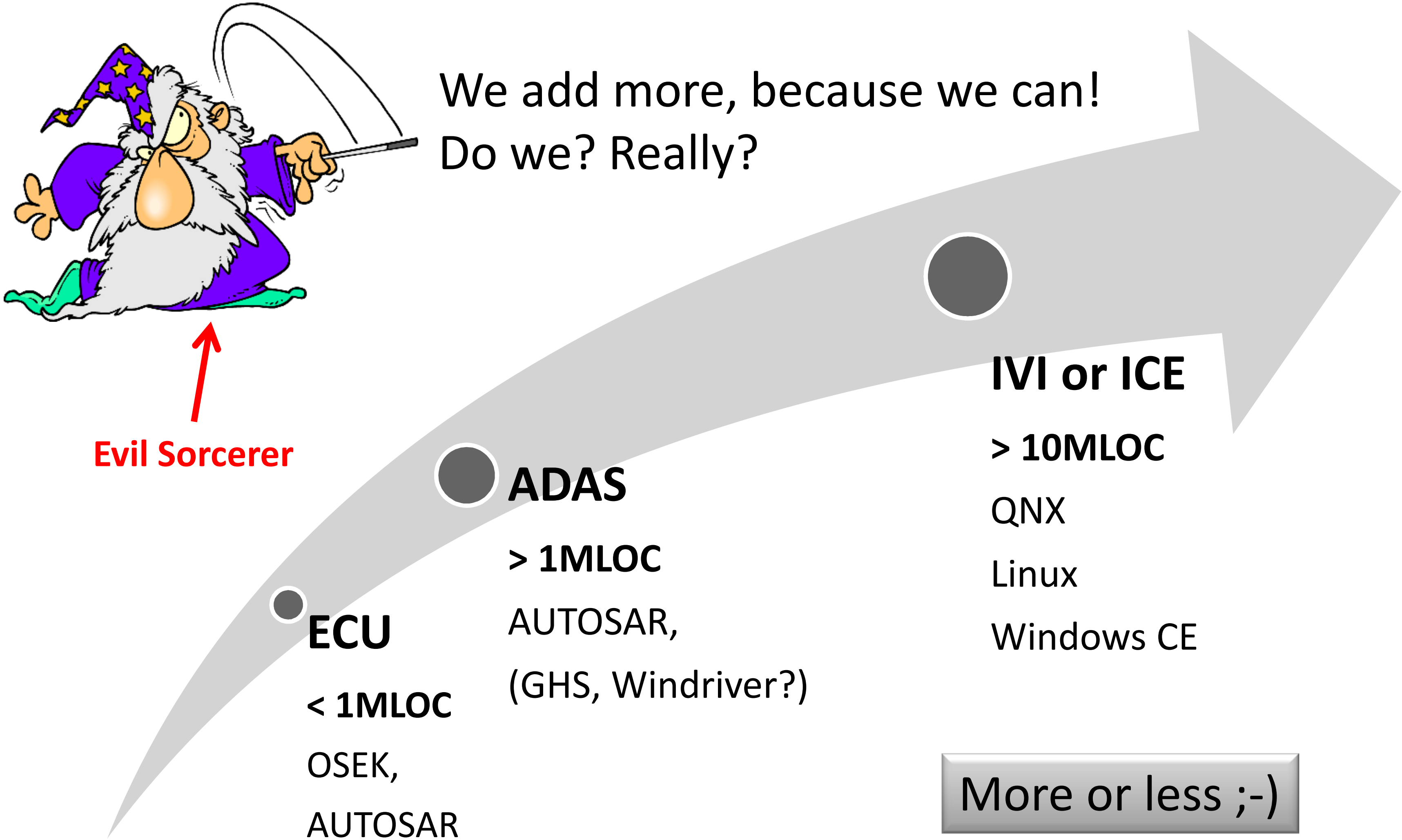
Software + General-Purpose Computer = Special-Purpose Machine

- Software is the ”*design of a machine abstracted from its physical realization*”.
- **The most complex systems ever built are all software systems.**

- OEMs don‘t want to build computers: **OEMs need to create functions for cars**.
- Functional modelling on car or system level is for OEMs.
- Application modelling is area of the OEM and Tier-1.

- **System responsibility (HW and SW) is area of? Tier-1?**

Increasing Complexity



We add more, because we can!
Do we? Really?

ECU

< 1MLOC

OSEK,

AUTOSAR

ADAS

> 1MLOC

AUTOSAR,

(GHS, Windriver?)

IVI or ICE

> 10MLOC

QNX

Linux

Windows CE

More or less ;-)

Life-cycle management & Engineering

- LM is a typical system engineering approach which utilizes the holistic view of the life-cycle: definition, realization, deployment and use and product and service life management.
- Mobility as a Service (MaaS) will be the future see seminal paper “MaaS by E. Verhulst)
- High assurance levels require feedback loops during operation which will be monitored and regulated by an independent regulatory agency

- ➔ Life-cycle engineering is a requirement
- ➔ Resilience and anti-fragility are the over-arching ideas

Testing vs. Operating

- System complexity rises, but consequences are not fully understood or considered: many development and operational activities are highly non-linear in nature, e.g. system verification and assurance, field monitoring, etc.
- Growing gap between testing effort and operational hours
  - Aggravated by standardization of components, e.g. MQB, MLB, etc.
- Lesson from Toyota-case:
  - Pure existence of defect was sufficient
  - Operational hours will eventually trigger the defect with certainty (!)
- Quote from VW: “due to high operational hours of components every defect in software will be triggered”
- Consequence: we produce systems where we don’t know what they do
- Most important effect: unknown unknowns (UNK-UNK)

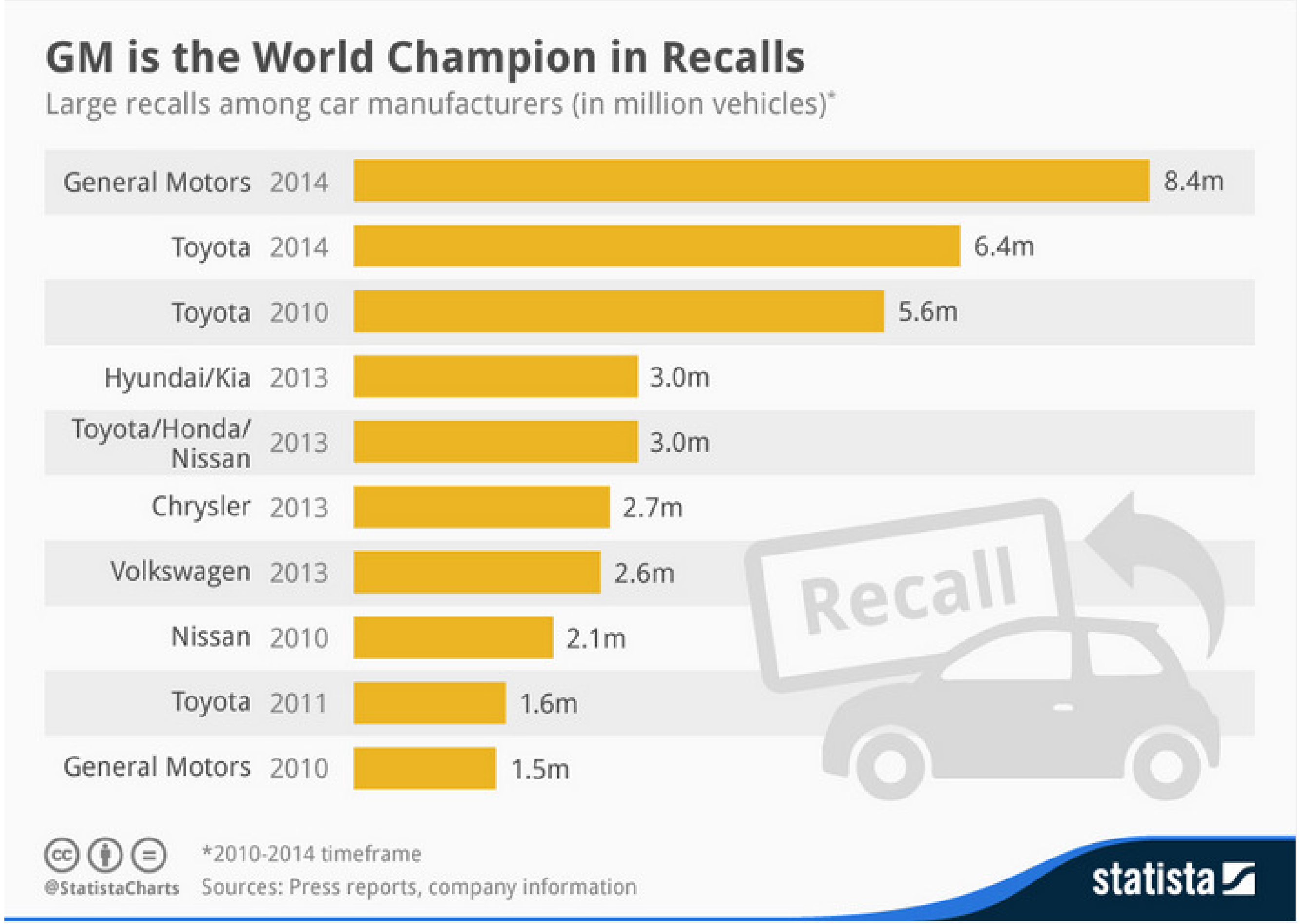
Different Systems

- Types of systems: simple != complicated != complex (!)
- Simple systems: (few) sensor(s) -> single point of control -> (few) actuators
  - Clearly defined system boundaries
  - Clearly defined relationships between in/out, mon/con
- Complicated systems: MIMO (multiple input, multiple output) with multiple points of control
  - “Muddy” system boundaries
  - Interaction between sub-systems is limited and fixed
  - Sub-systems and their interaction don’t adapt, e.g. to system heuristics
- Complex systems:
  - No system boundaries, the system is the domain
  - System heuristics steer, but don’t control directly
  - Adaptive sub-systems with adaptive interactions

➔ Feedback loop during operation, monitoring and permanent updates

Latest Recall Numbers

- Source:
  Statista, 2015



GM is the World Champion in Recalls

Large recalls among car manufacturers (in million vehicles)*

*2010-2014 timeframe

Sources: Press reports, company information

Path to Domain Controller

Elektrobit

From Software to Software Systems

- Complexity in software is controlled by:
  - **Keep the functionality simple**: *„Challenge the requirements“* (problem space).
  - **Keep the software simple**: software architecture (solution space).
  - Problem space: **highly non-linear effect** on the solution space!
- Real value of EB tresos™ Safety products:
  - Reduce and control complexity
  - **Enable co-existence of different software parts in the same system**
- **Complexity is controlled on a software system level.**
  We need to understand systems engineering on a software level!
  - **Software development tools need to adapt to this**
  - **Hardware analyzing (i.e. Debugger) tools need to adapt this**
- **Domain controllers require safety software architectures and safety products.**

The Path to Domain Controllers

- Growing computational demand + economy of scale (production).

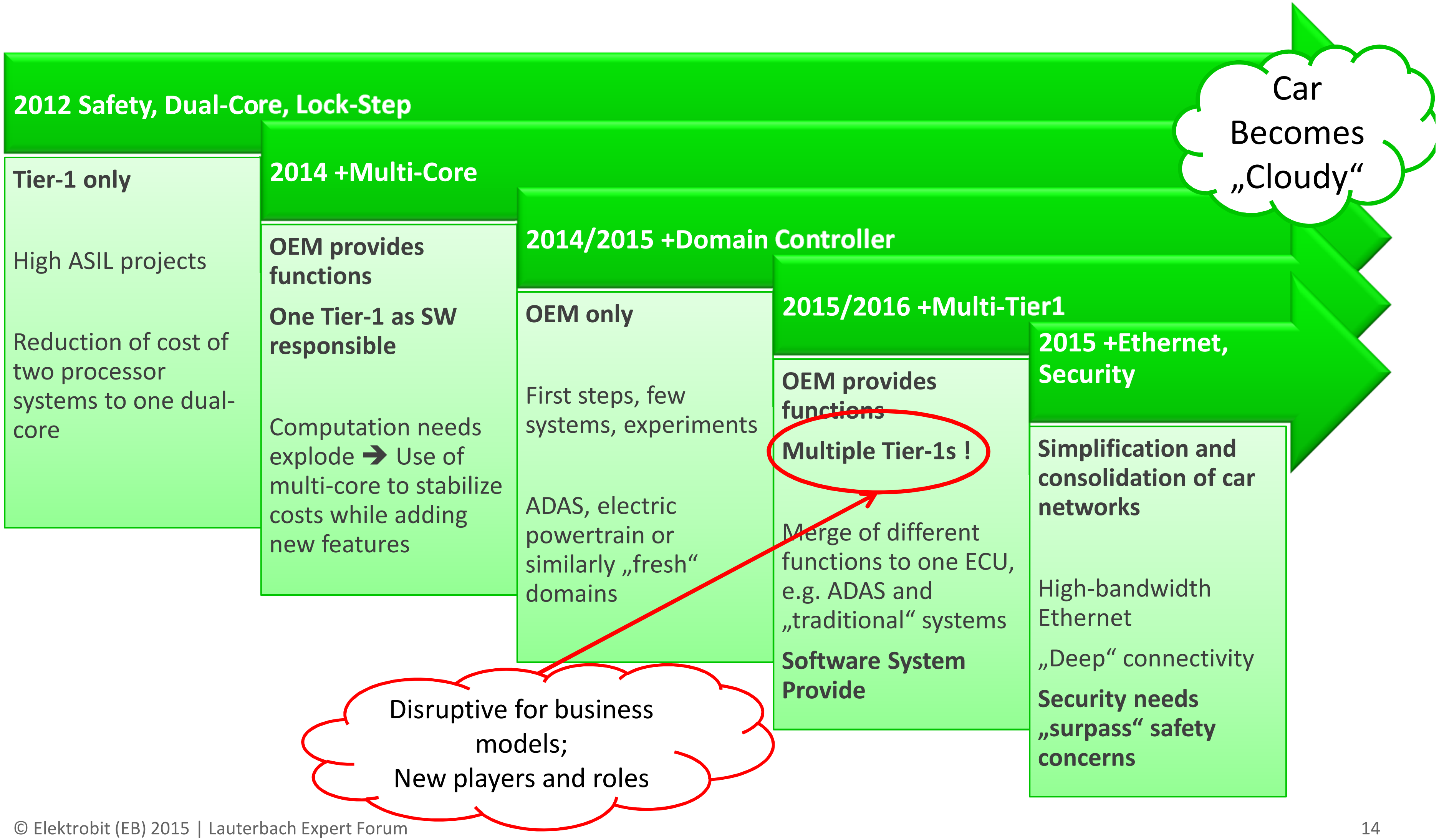
- Reduce „intelligence“ (and cost) of sensors and actuators to a minimum.
- Replace such „intelligence“ with **centralized software functions.**

- **Leverage multi-core** (application from OEM + **one Tier-1**).

- Introduce **domain controllers** (application from OEM + **multiple Tier-1s**).

- Different domain areas ECU, ADAS and part of IVI will merge as a result.

Time-Line



2012 Safety, Dual-Core, Lock-Step

Tier-1 only

High ASIL projects

Reduction of cost of two processor systems to one dual-core

2014 +Multi-Core

OEM provides functions

One Tier-1 as SW responsible

Computation needs explode ➔ Use of multi-core to stabilize costs while adding new features

2014/2015 +Domain Controller

OEM only

First steps, few systems, experiments

ADAS, electric powertrain or similarly „fresh“ domains

2015/2016 +Multi-Tier1

OEM provides functions

Multiple Tier-1s !

Merge of different functions to one ECU, e.g. ADAS and „traditional“ systems

Software System Provide

2015 +Ethernet, Security

Simplification and consolidation of car networks

High-bandwidth Ethernet

„Deep“ connectivity

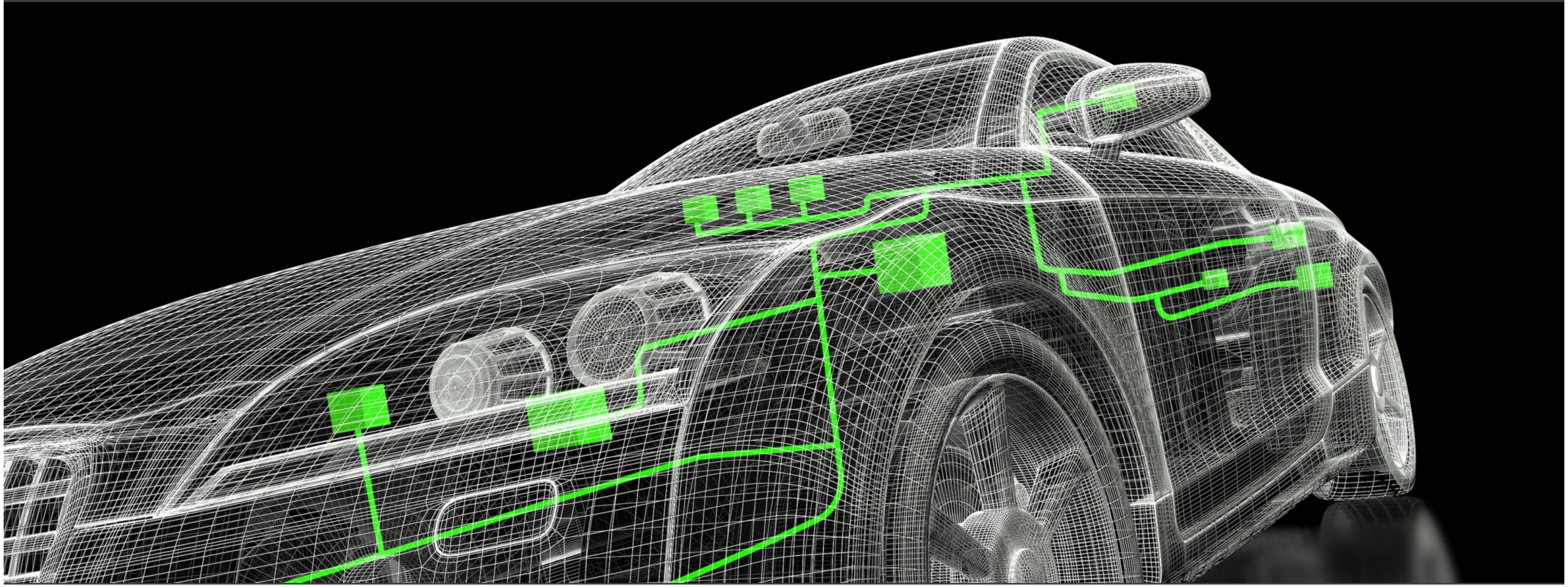
Security needs „surpass“ safety concerns

Car Becomes „Cloudy“

Disruptive for business models;
New players and roles

Possible Set-up for a Domain Controller

Elektrobit

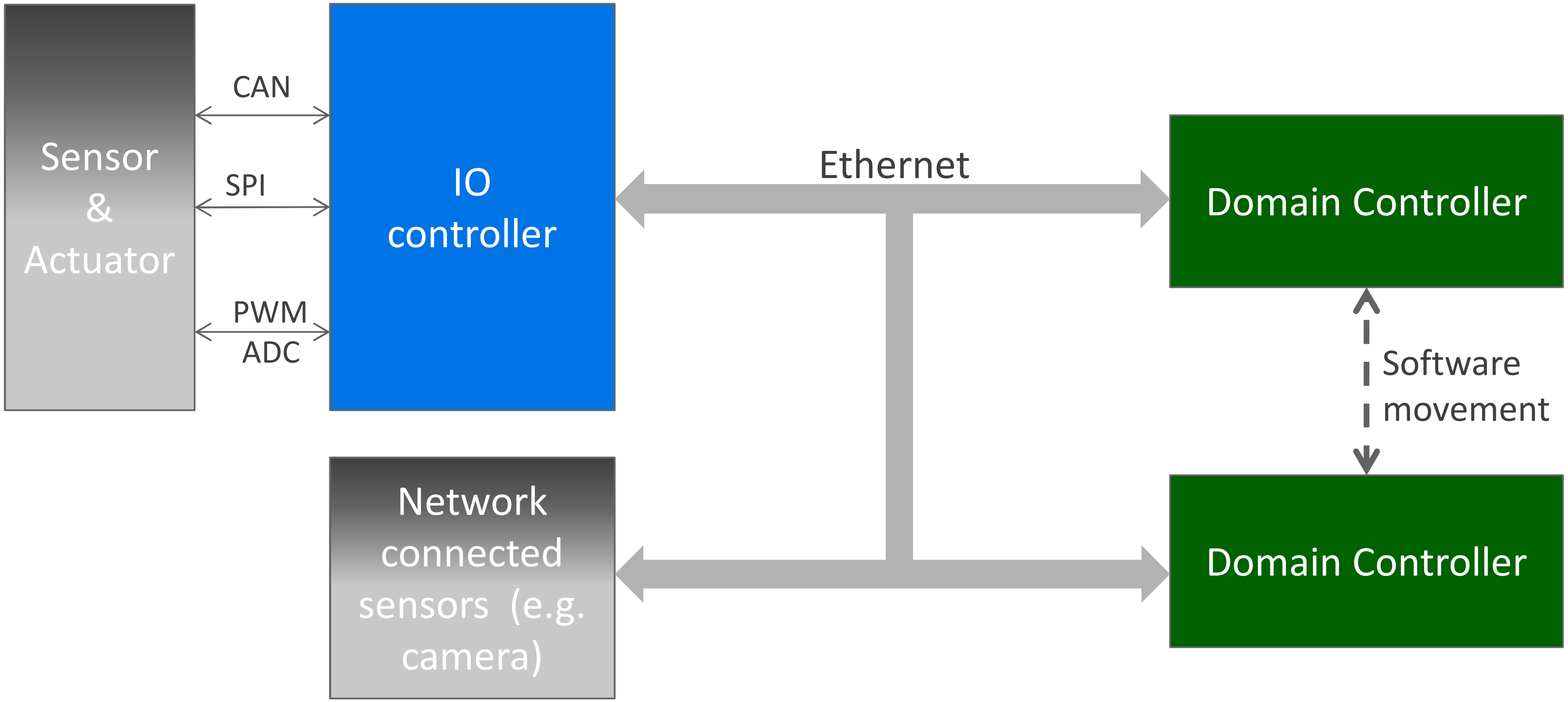


Vision

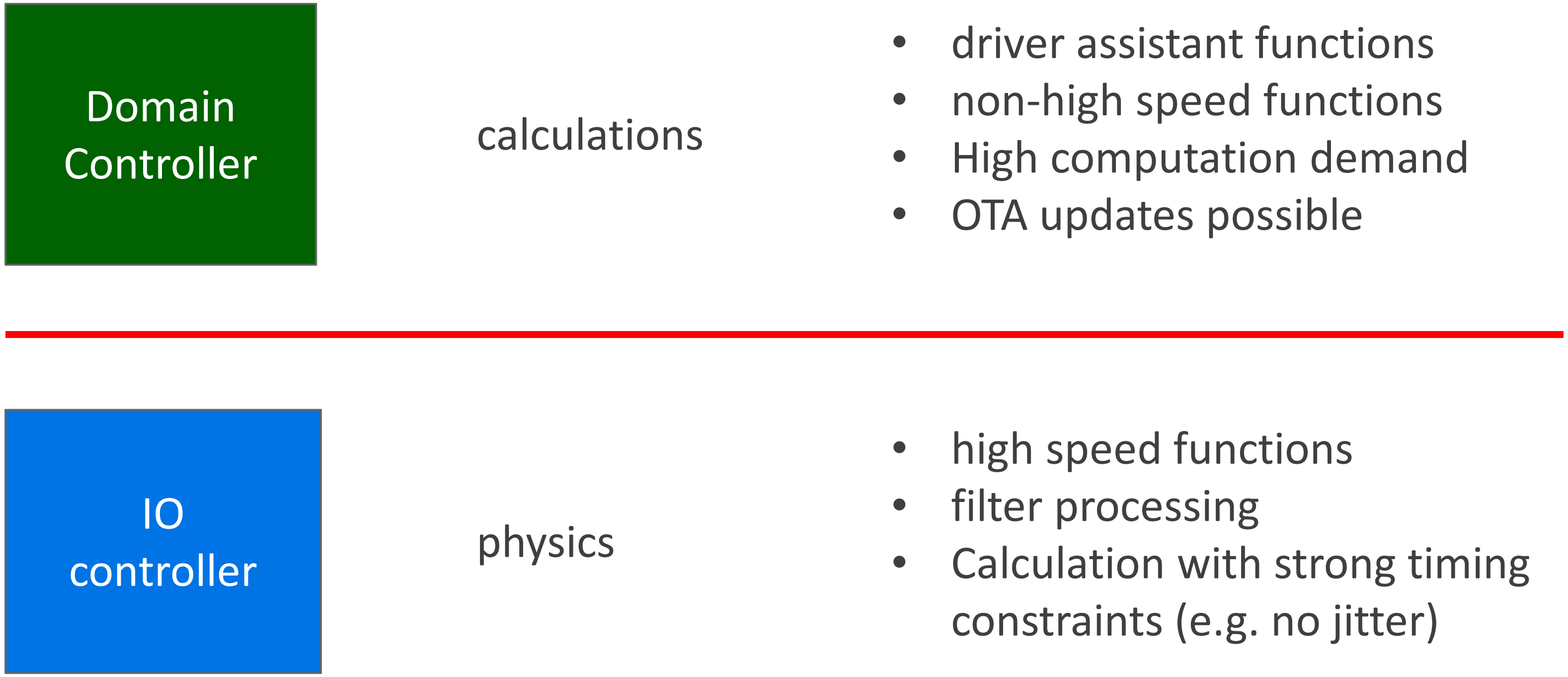
- Dependable System
  - functions can be switched between Central Controllers (cold or hot standby)

- Reloadable functions
  - Updatable, reconfigurable functions
  - “functions in the AppStore” – e.g. race-line in head up display for Brands Hatch circuit

Remote I/O Architecture



How to divide the functionalities?



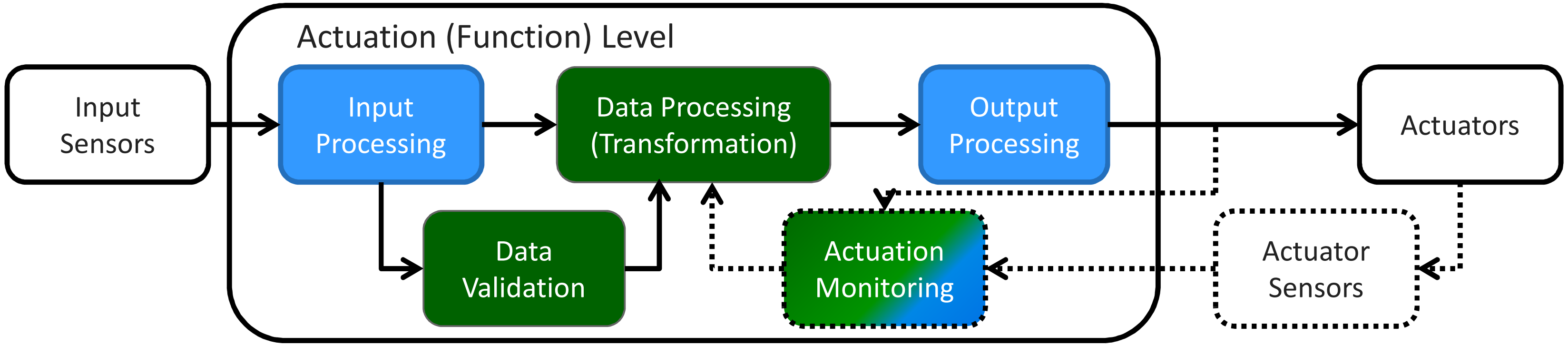
- driver assistant functions
- non-high speed functions
- High computation demand
- OTA updates possible

- high speed functions
- filter processing
- Calculation with strong timing constraints (e.g. no jitter)

Basic considerations

- IO Controller shall put every qualified signal on the network
  - Pro: enables easy integration of new/changed functionality on Central Controller
  - Pro: reduced complexity, due to low dependency between ECUs
  - Con: high busload
- IO Controller shall work time sliced
  - Pro: ease verification and validation
  - Pro: synchronization to time triggered bus
  - Pro: reduced scheduling overhead cause by preemptions
  - Con: latencies
- Domain Controller
  - Large processing capabilities
  - Multiple operating systems (e.g. Embedded Linux, AUTOSAR) with hypervisor

Pattern: Protected Single Channel

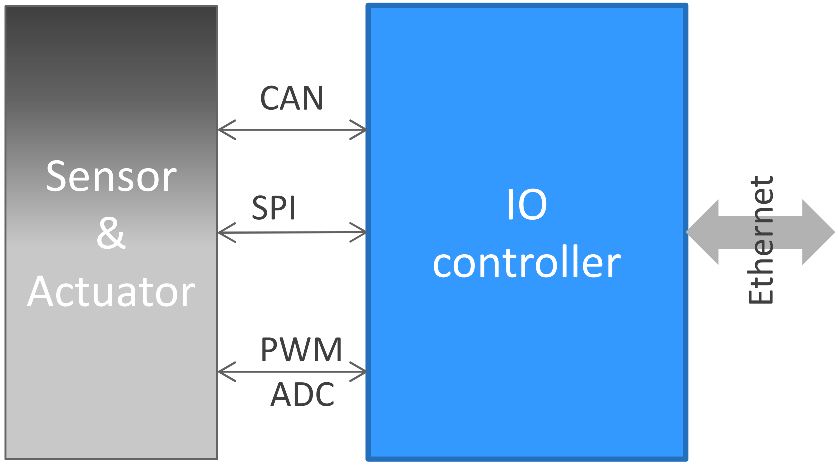


- Data validation: HW check, plausibility check, “smoothing”, etc.
- Data integrity
  - local: redundant storage or checksum
  - Communication: Alive counter & checksum or security algorithm (e.g. CMAC)
- Dotted lines: measuring the final result – Goal: a closed-loop system

🟩 Domain Controller

🟦 IO controller

IO controller



- AUTOSAR as base architecture
  - Reuse of existing SW-C
  - Safety concepts available
  - Full support of diagnostics
  - IO driver available
  - Complex driver non standard IO connections

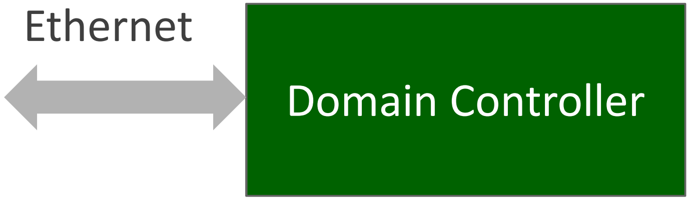
Assumptions:

- Sensor cycle times between 100 us and 100ms
- Actuator response times is critical
- Support of different safety requirements (QM – ASIL D)

Domain Controller

Multi Core CPU with focus on performance

- Support of “dynamics”
  - Start/stop applications
  - Memory management
- Support of high-level requirements libraries e.g.
  - OpenGL
  - Qt
- Support of automotive standard functions
  - Integration of AUTOSAR software components
  - Reuse of diagnostic software modules e.g. UDS, OBD



Proposal: Hypervisor controlling multiple operating systems

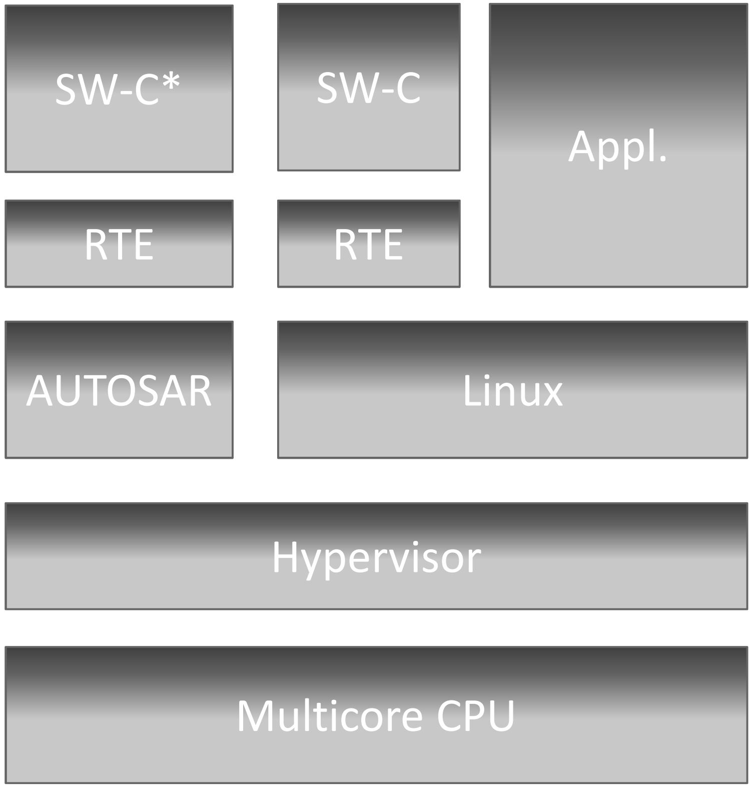
Domain Controller

Hypervisor concept shall

- Ensure spatial and temporal separation to show that the real-time application is not impacted by the non-real-time applications
- Enable migration /reuse of existing software

AUTOSAR SW-C* can be integrated on

- Standard AUTOSAR environment
- RTE running on a Linux



*) AUTOSAR software component

Communication

Requirements

- High and guaranteed bandwidth for sensor data streams
- Low latency
- Global Time Sync for global time and synchronous task execution
- Timestamps for application data elements
- Quality of Service
- Fault tolerance

Proposal: Time-Sensitive Networking (TSN)* on Ethernet

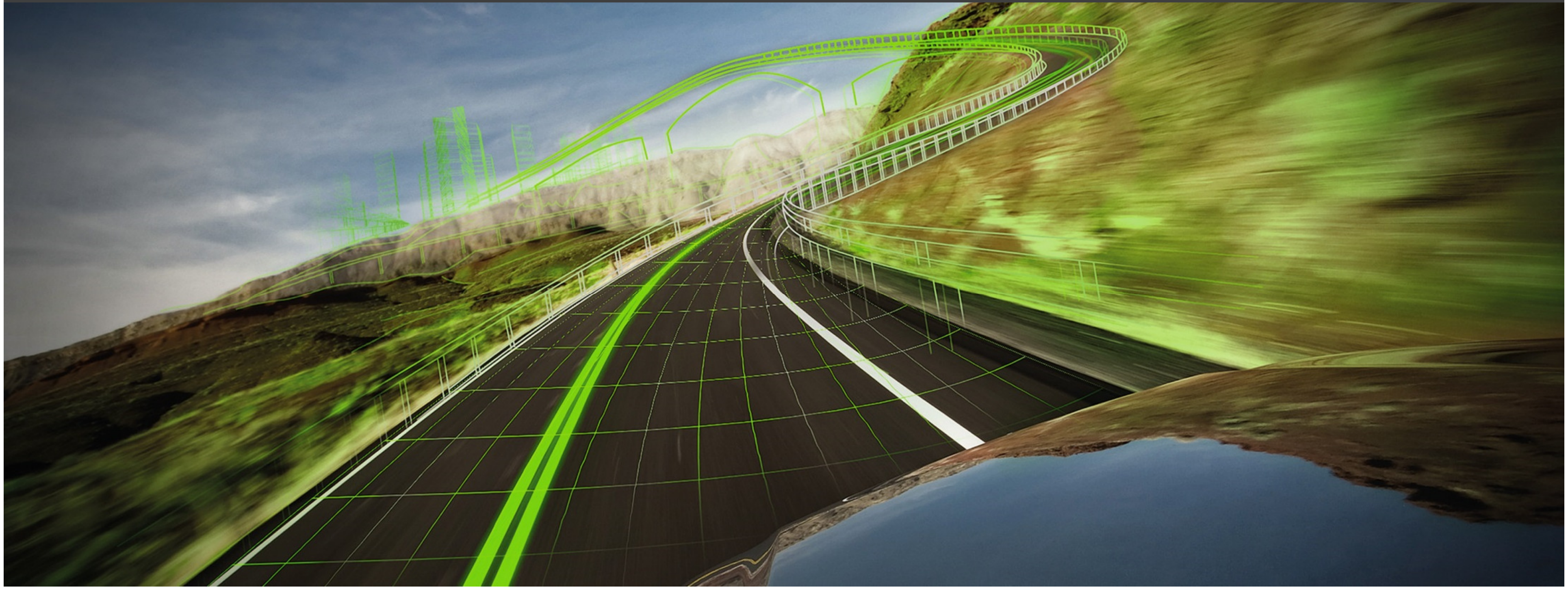
- 802.1AS: Timing and Synchronization for Time-Sensitive Applications
- 802.1Qav: Forwarding and Queuing for Time-Sensitive Streams (Guaranteed bandwidth & latency)

Experience: Concept validated with ASR 4.2.1 +RfCs in a demonstrator project

* Previously known as Audio Video Bridging (AVB)

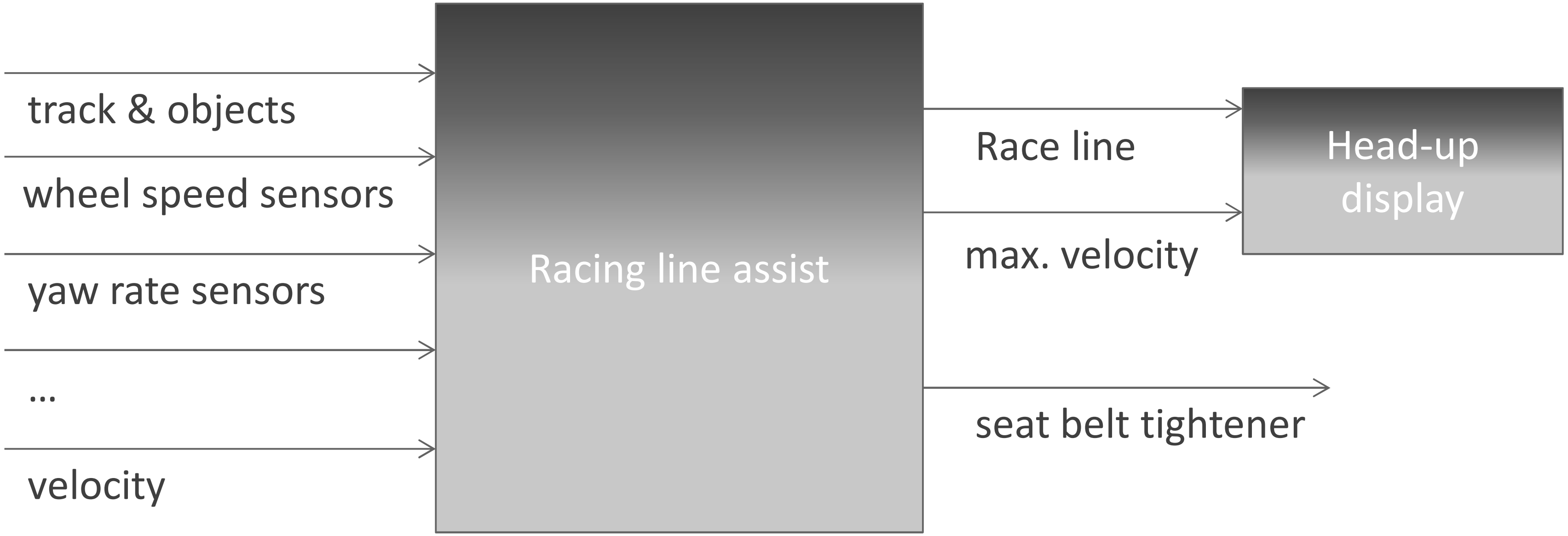
Example function – race track assistant

Elektrobit



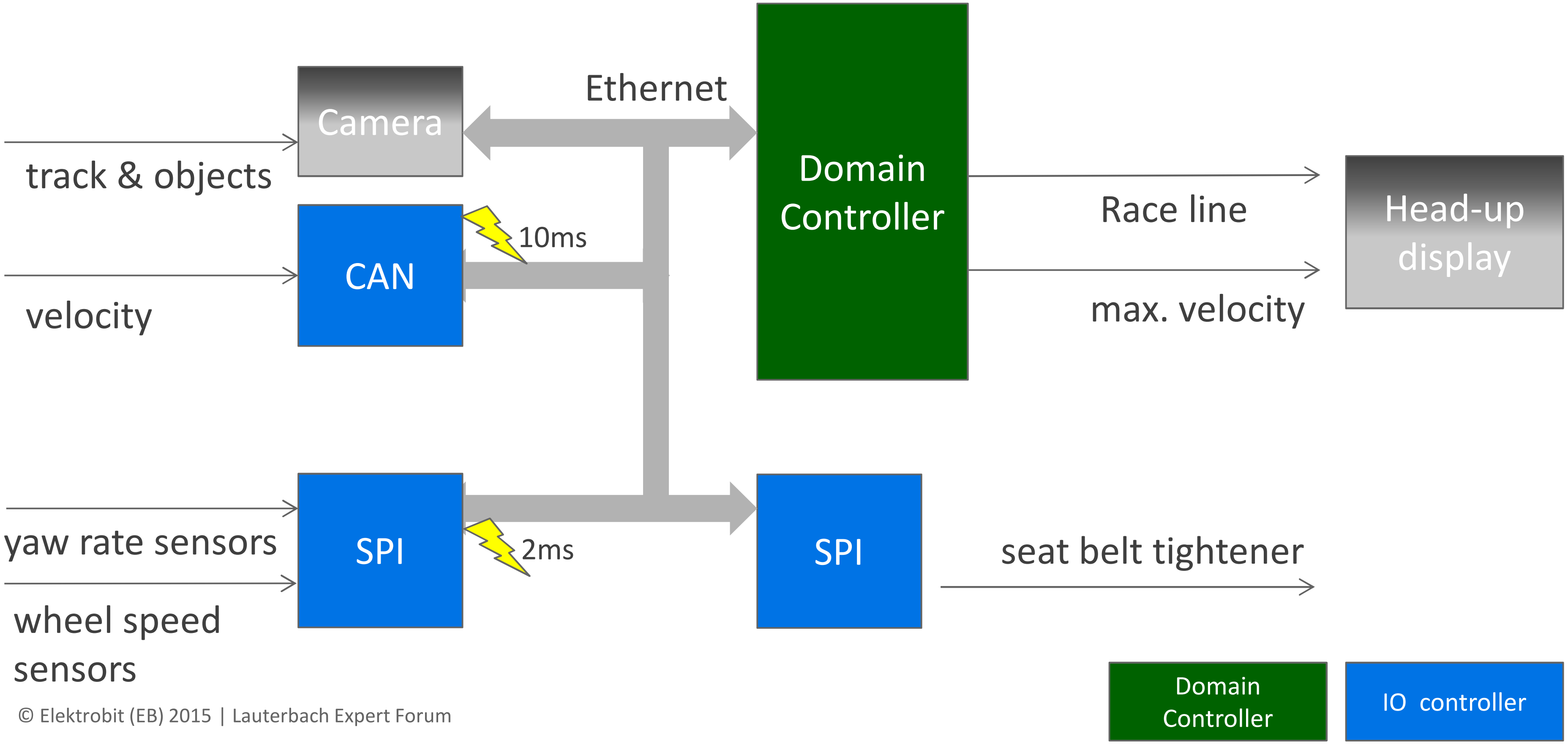
Top-level view

- Advises driver for best racing line
- controls seat belt tightener (pre-crash)



Data Flow

- Advises driver for best racing line
- In pre-crash situation the system activates the seat belt tightener



Tasks IO controller

Task 500us:

- Read yaw rate and
- Calculate filtered yaw rate
- Read wheel speed sensors
- Calculate filtered wheel speed values

Task 2 ms:

- Perform plausibility checks on wheel speed and yaw rates
- Send qualified measures on Ethernet

Task 10ms:

- Perform plausibility checks wheel speed and velocity
- Gateway CAN messages to Ethernet

Perform L3 checks (SystemDiagnosis):

- Temporal monitoring
- Voltage monitoring
- Build in self test

Task 1ms:

- Act on pre-crash calculations

Tasks Domain Controller

Common function

- Read data from communication bus
- Perform street and object recognition from camera picture

Comfort functions:

- Calculate driver information
- Forward driver information to head-up display

Safety function:

- Calculate possible object impact
- Initiate pre-crash actions (e.g seat belt tightener)

Outlook: reading the „crystal ball“

- **Multi-Core ECUs** are now in development for mass market:
  More functions per ECU, but usually the same supplier.

- **Domain-controllers** will follow:
  Even **more functions** per ECU, from **different suppliers** and the **OEM**.
  → This changes the software supplier structure
  → New **business and cooperation** models needed, e.g. software integrators?
  → **OTA** updated and Apps will be among the next big things with Domain controllers

- **Re-use** of standardized software architectures that support safety and multi-core is a **key success factor.**
- **Effective Tools** and using them in the right way will be basis for you success

- During this period: **reliability** is re-discovered as important quality aspect and more ECUs have **availability** requirements.

- After that **security** will be the next **hot topic**:
  → Ethernet, the connected car, autonomous driving, etc.

Questions?
Contact us!

Elektrobit

automotive.elektrobit.com
jan.ruediger@elektrobit.com