Platforms and Algorithms for Autonomous Driving (platforms and safety design)

Nacho Sañudo
University of Modena and Reggio Emilia
Ignacio.sanudoolmedo@unimore.it

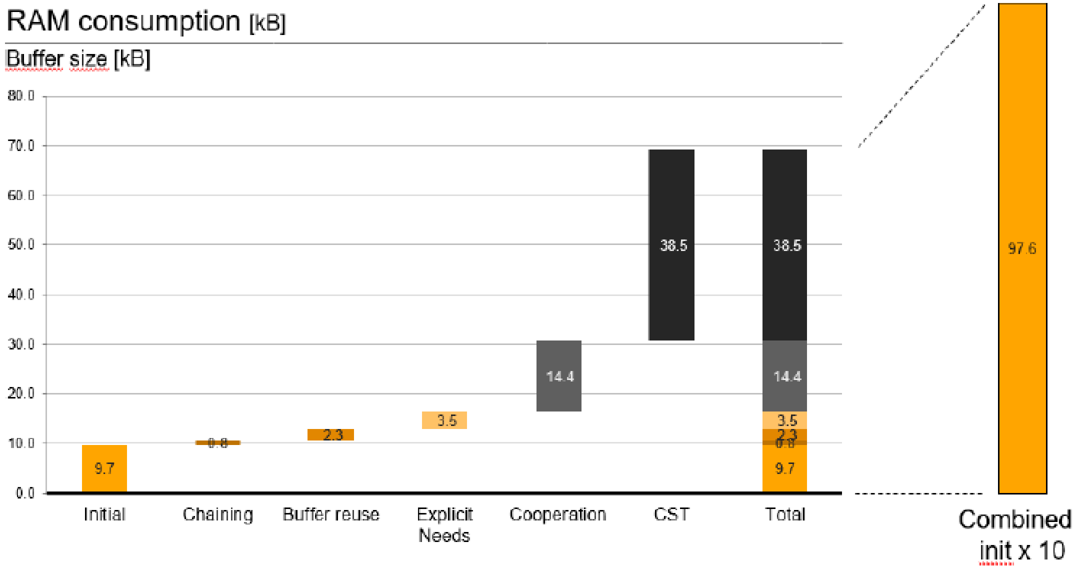
Engine control application (i) – Memory footprint

- Typical project (RAM consumption)
- ***“Safety critical applications are safe and simple by design”***

Typical Project

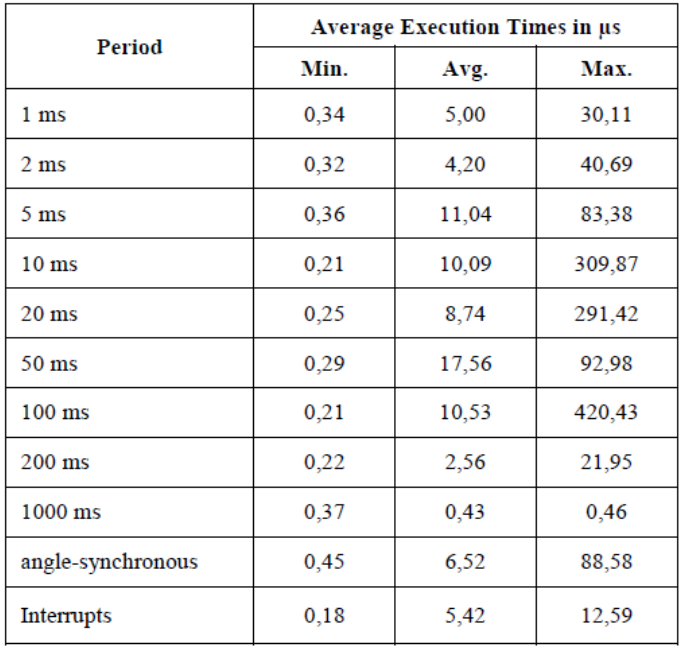
3.500 modules
30.000 data
16.000 c-functions

120 Tasks
1500 Runnables
100.000 access points

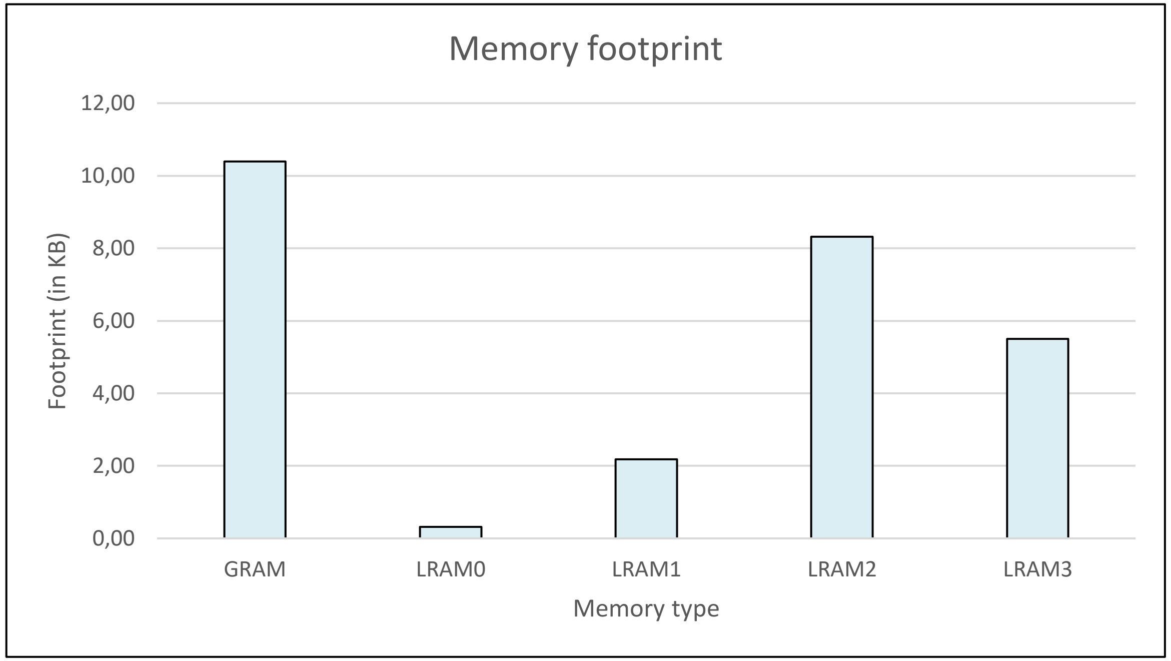


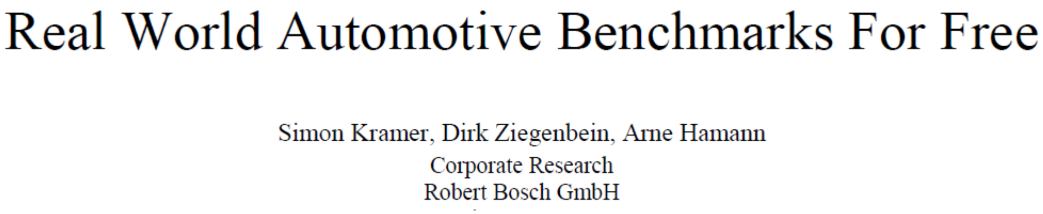
CRITICAL: EFFICIENCY OF BUFFERING ALGORITHM

Engine control application (ii) – Memory footprint



Period	Average Execution Times in µs		
	Min.	Avg.	Max.
1 ms	0,34	5,00	30,11
2 ms	0,32	4,20	40,69
5 ms	0,36	11,04	83,38
10 ms	0,21	10,09	309,87
20 ms	0,25	8,74	291,42
50 ms	0,29	17,56	92,98
100 ms	0,21	10,53	420,43
200 ms	0,22	2,56	21,95
1000 ms	0,37	0,43	0,46
angle-synchronous	0,45	6,52	88,58
Interrupts	0,18	5,42	12,59



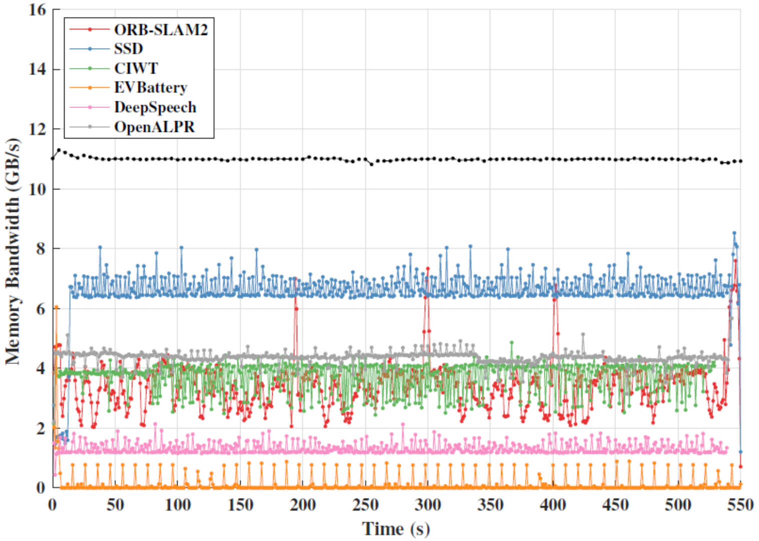


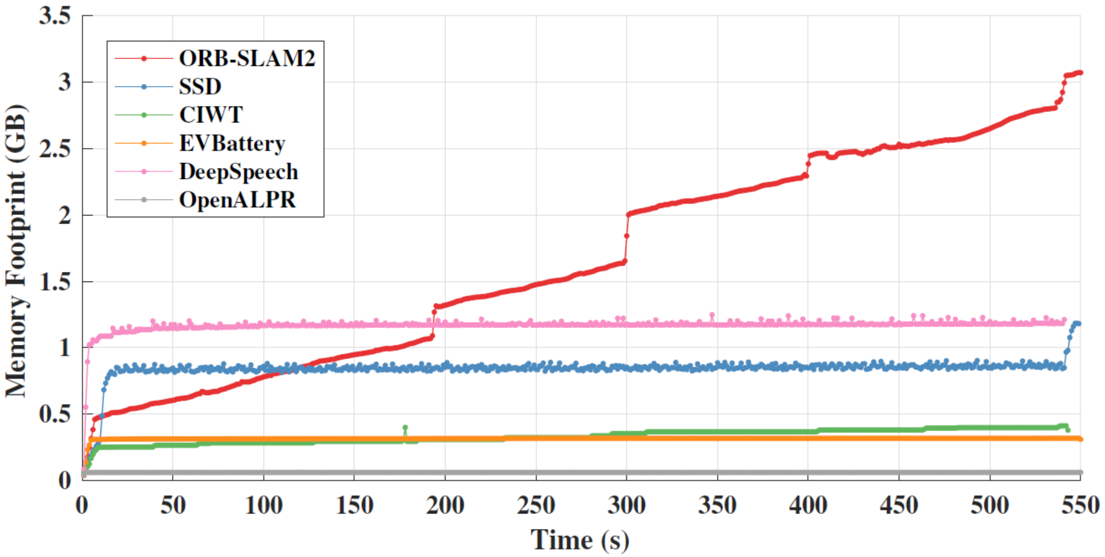
Real World Automotive Benchmarks For Free

Simon Kramer, Dirk Ziegenbein, Arne Hamann
Corporate Research
Robert Bosch GmbH

Autonomous driving complexity – Memory footprint

- Many applications used for autonomous driving are memory intensive
- *Memory is the main **predictability bottleneck** for heterogeneous SoCs*

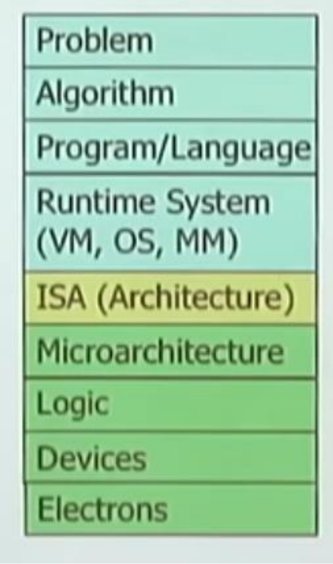




CAVBench: A Benchmark Suite for Connected and Autonomous Vehicles
Yifan Wang et al.

Computer architecture

- How do we ensure that problems are solved by electrons?
  - Model of multiple layers
  - **Which architectures have been used to integrate SW-C in the automotive domain?**



Problem

Algorithm

Program/Language

Runtime System
(VM, OS, MM)

ISA (Architecture)

Microarchitecture

Logic

Devices

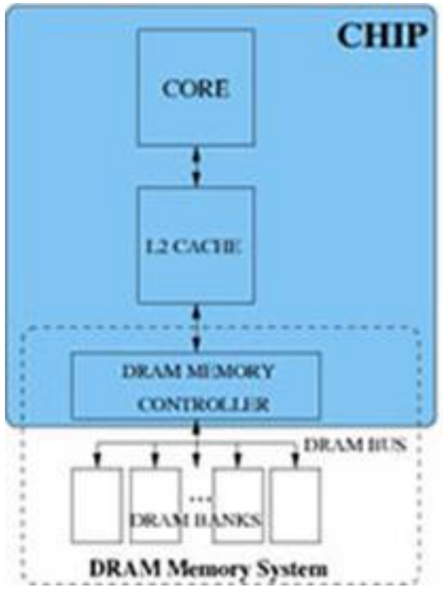
Electrons

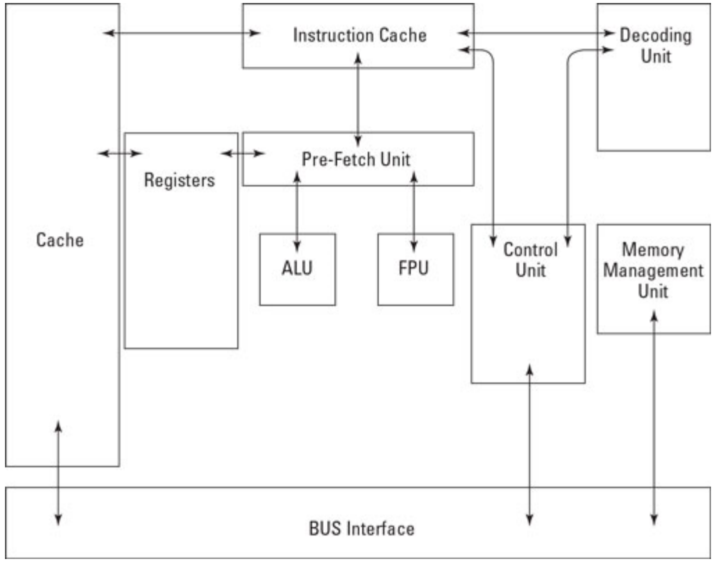
Seminar in computer Architecture (Introduction and basics). Onur Mutlu

CPU Single-core

- The CPU (Central Processing Unit) is the main component of a computer
- A single-core processor is a microprocessor with a single core on a chip, running a single thread at any one time
  - At the hardware level, a computer executes sequences of individual instructions
  - Each instruction tells the computer to add, subtract, multiply….

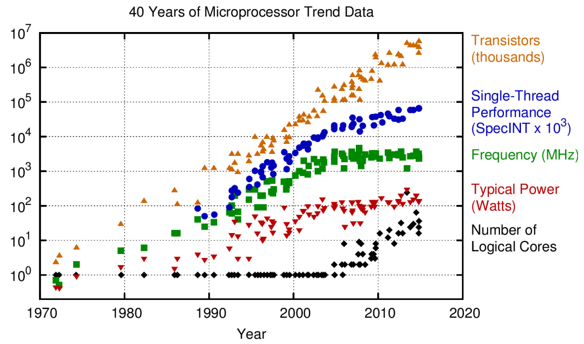
```
int A=0;
float B=0.0;
A=2+2;
B=A*2.0;
```





Single-core to multi-core

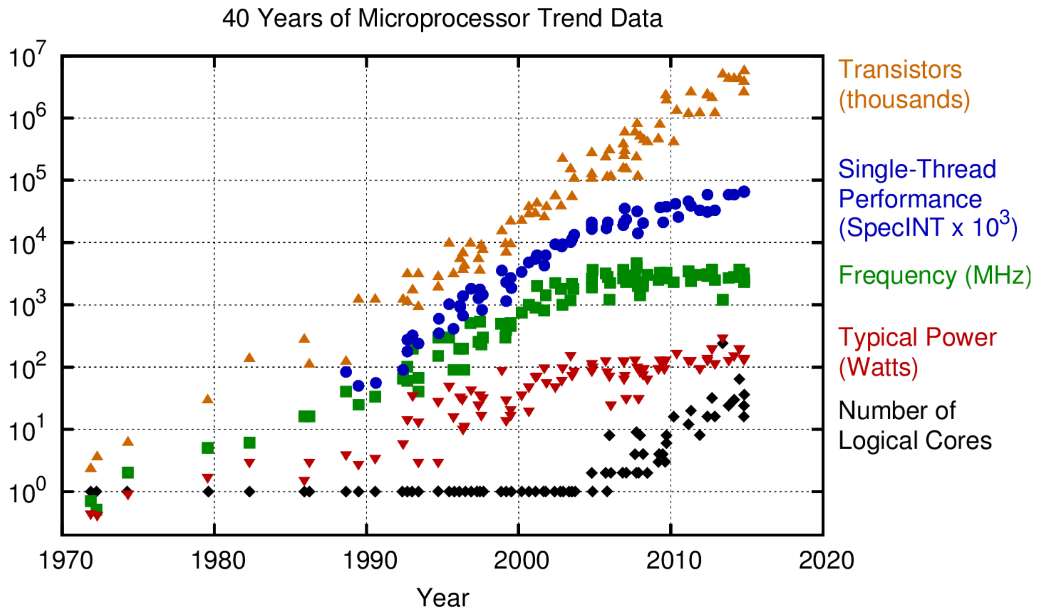
- The shift from *single- core* to *multi-core* architectures allowed to integrate more complex SW-C
  - The design goal for the late 1990’s and early 2000’s was to drive the clock rate up. This was done by adding more transistors to a smaller chip
    - “The number of cores will double every 18 months, while power, clock frequency and costs will remain constant” [Patterson & Hennessy]
    - Unfortunately, this increased the power dissipation of the CPU chip beyond the capacity of inexpensive cooling techniques (power wall)
  - **Solution**: put multiple cores on the same die
    - Currently, 4/8 cores are mainline, and counting…
      - AMD: 16-core, 32-thread 3.4-GHz Threadripper 1950X
      - Intel: 18-core Core i9-7980X
      - GPU, FPGA, TPU



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Single-core to multi-core

- Similar trend for automotive embedded real-time systems:
  - Nvidia Parker, Xavier, Pegasus
  - Xilinx Zynq Ultrascale+, Versal
  - Intel GO, EyeQ4/5
  - Renesas R-Car
  - Qualcomm S820A, NXP BlueBox



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Computer architecture

- **Designing a smartphone is different from an automotive ECU**
- Computer architecture: Science and art of designing computing platforms (HW, SW etc...)
- Computers are designed to achieve **different design goals**
  - High performance
  - Power efficiency
  - Cost
  - Performance/cost
  - **Safety**

Automotive

- › Automotive components have additional constraints:
  - Safety
  - Scalability
  - Portability
  - Energy efficiency
  - Cost
  - .....

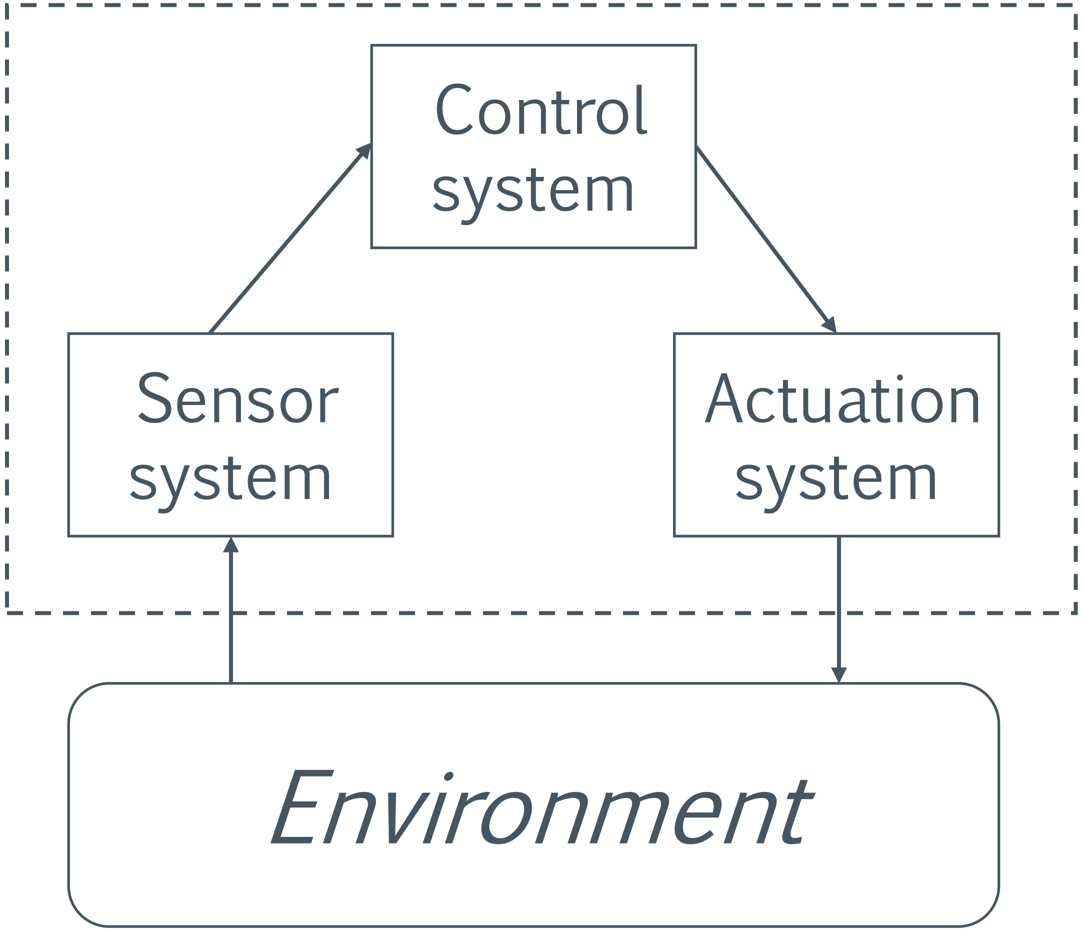
AUTOSAR

ISO 26262
Road Vehicles - Fuctional Safety

SIL

- › The increasing complexity does not help to provide systems that guarantee those constraints
  - − **Complexity = Unpredictability**

Real-Time Systems

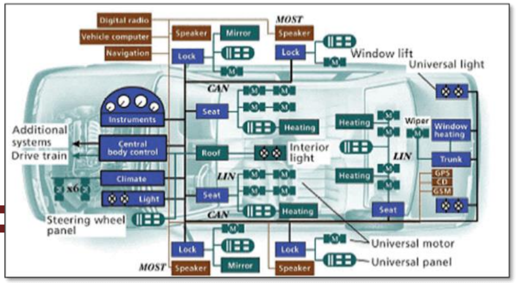


"Real-time systems are computing systems that must react within precise time constraints to events in the environment."

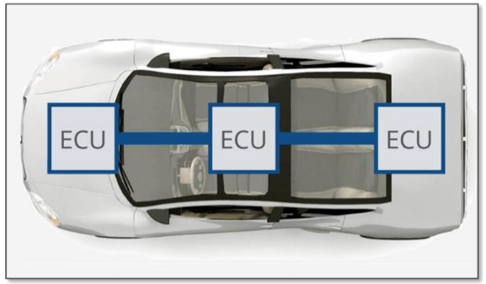
Giorgio Buttazzo.
Hard Real-Time Computing Systems

Paradigms

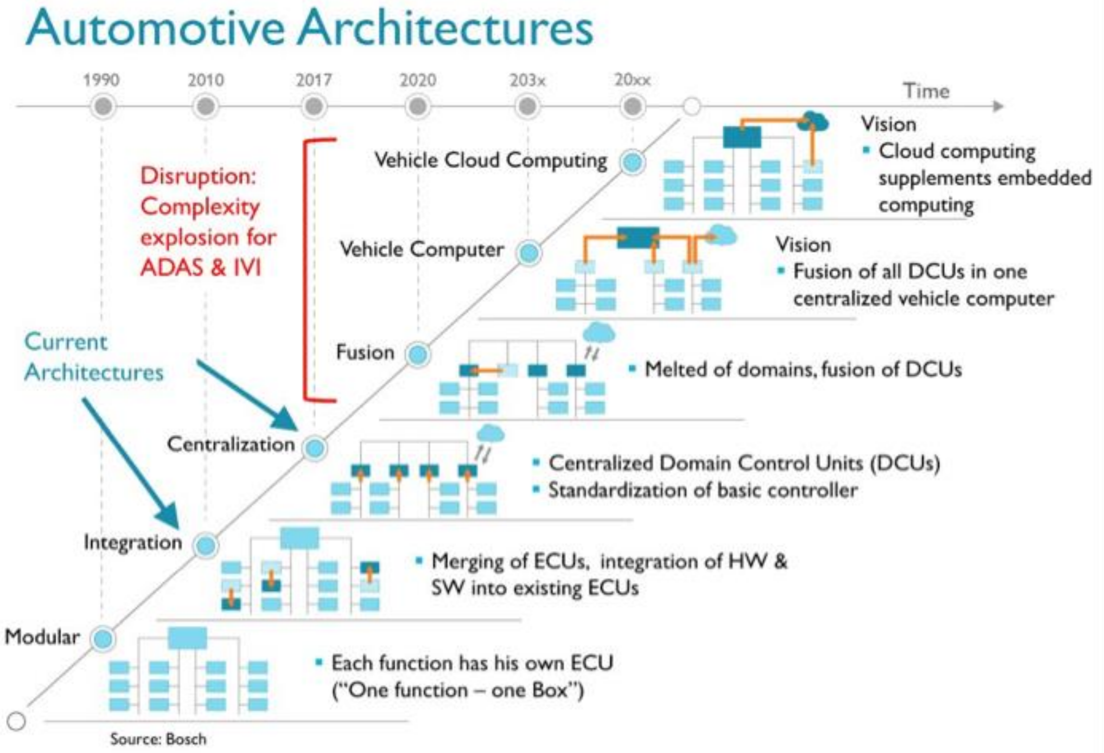
Conventional Architecture



Software Centric Approach



- From an OEM point of view, it is still unclear what would be the more convenient approach for the development of ADAS and self-driving applications
- Companies like Bosch, Ford or General Motors are evaluating different **paradigms** and architectural solutions to understand how domain controllers will be built
  - **Distributed E/E Architecture** (yesterday)
  - **Domain/Zonal Centralised E/E Architecture** (today)
  - **Vehicle Centralised E/E Architecture** – (tomorrow)



Automotive Architectures

1990 2010 2017 2020 203x 20xx Time

Disruption: Complexity explosion for ADAS & IVI

Current Architectures

Vehicle Cloud Computing
Vision
- Cloud computing supplements embedded computing

Vehicle Computer
Vision
- Fusion of all DCUs in one centralized vehicle computer

Fusion
- Melted of domains, fusion of DCUs

Centralization
- Centralized Domain Control Units (DCUs)
- Standardization of basic controller

Integration
- Merging of ECUs, integration of HW & SW into existing ECUs

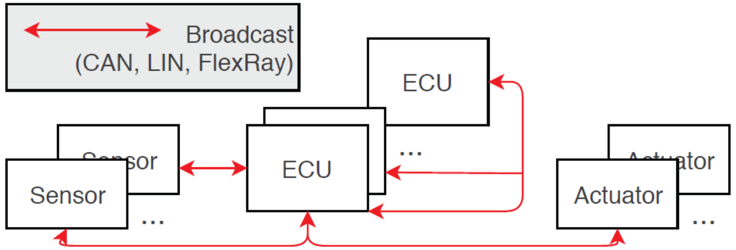
Modular
- Each function has his own ECU ("One function – one Box")

Source: Bosch

Source: Bosch

Distributed E/E architecture

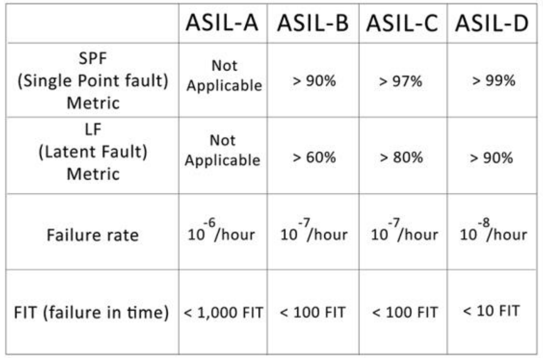
- Traditional automotive architectures
- **Each function is delivered using a specific ECU**
- Several ECUs are connected via a broadcast bus
- The distributed architecture is highly modular as each function is delivered using a **specific** ECU
  - ECUs are connected to the vehicle network and have an embedded microcontroller, which controls sensors, actuators, etc…
- OEM is **VERY** dependent on the TIER-1 for hardware and software



Examples:

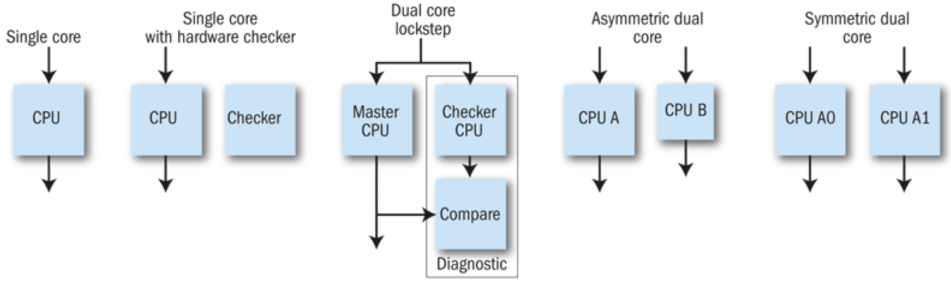
- Volvo P80 platform
- Ford EUCD platform (Ford, Jaguar, Land Rover, Volvo….)
- Volkswagen Group B platform (Volkswage, SEAT, Audi, Skoda…)

Distributed E/E architecture

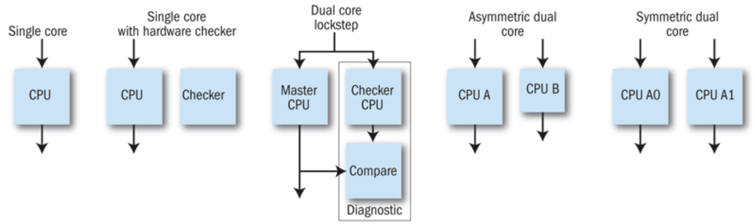


	ASIL-A	ASIL-B	ASIL-C	ASIL-D
SPF (Single Point fault) Metric	Not Applicable	> 90%	> 97%	> 99%
LF (Latent Fault) Metric	Not Applicable	> 60%	> 80%	> 90%
Failure rate	10^{-6}/hour	10^{-7}/hour	10^{-7}/hour	10^{-8}/hour
FIT (failure in time)	< 1,000 FIT	< 100 FIT	< 100 FIT	< 10 FIT

- **How to safely integrate SW-C within a CPU?**
- To support the real-time requirements of automotive functional safety applications, CPUs should provide safe designs
  - **detect** their own operational failures in real-time and react in a way to avoid harming the people
  - **redundancy** is the addition of extra components to increase the reliability and availability of the system
    - The redundancy can be added in the form of hardware or software
- CPU safety designs can be classified as
  - single core, single core with hardware checking, dual-core lockstep, asymmetric dual core, symmetric dual core

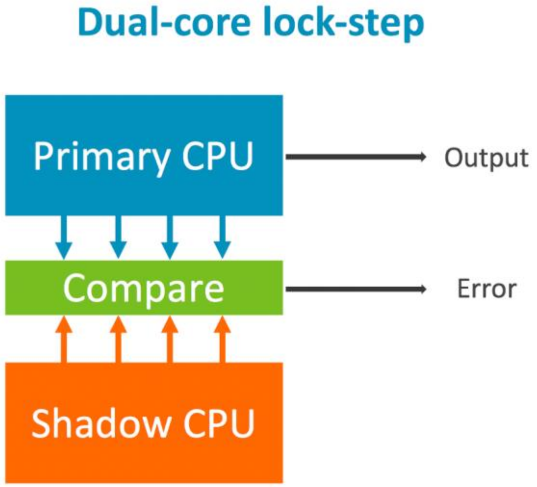


https://www.eetimes.com/matching-processor-safety-strategies-to-your-system-design/



Single core/Dual core

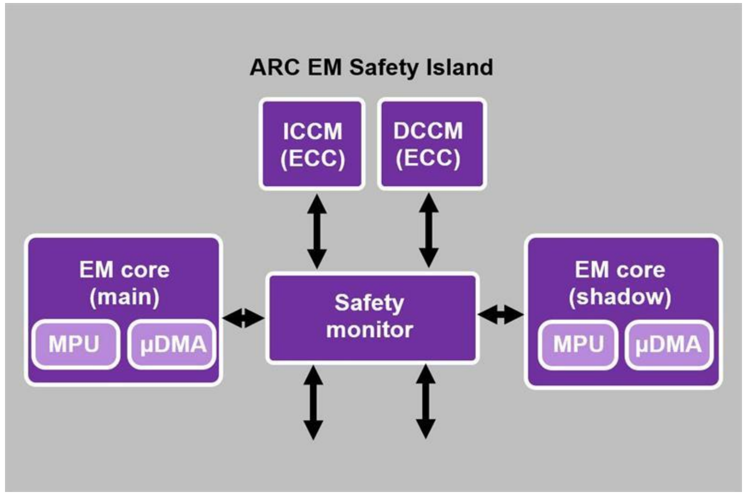
1. **A single core** processor is a microprocessor with a single core on a chip running a single thread or a single process at a time
   - Safety: **software-based diagnostics** (big overhead)
2. **Hardware checker**: device that periodically checks the correctness of the CPU



Source: ARM

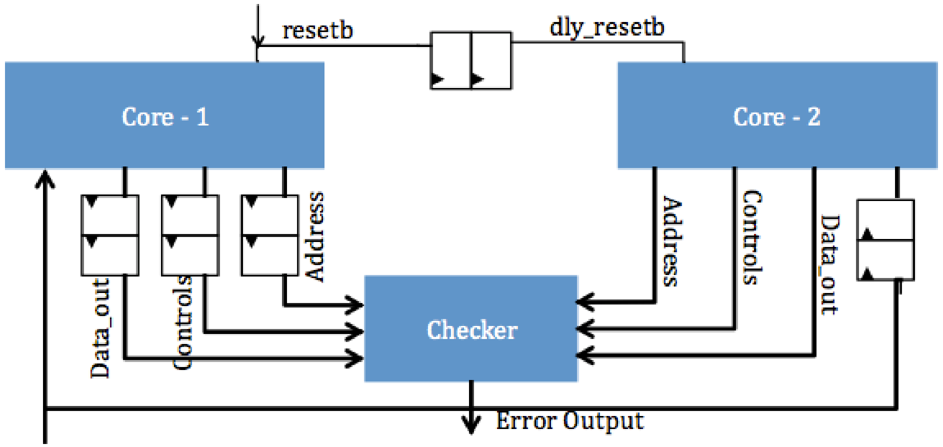
Lockstep

- **Dual-core lockstep** systems are fault-tolerant computers that run the same set of operations at the same time in parallel
  - Low-complexity
  - Widely used
- Software runs synchronized in the two processors
  - Partially or fully duplicated processor core
    - perform the same calculations in the same clock cycle
  - Hardware checker to compare the results
    - If the identical result is not obtained, then it is assumed that a fault has occurred
  - No software overhead in verifying the results

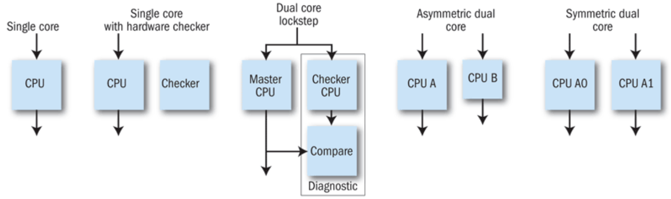


Delayed Lockstep

- Software runs synchronized in the two processors
- *Inputs* of one core are delayed by N clock cycles
  - time diversity is achieved

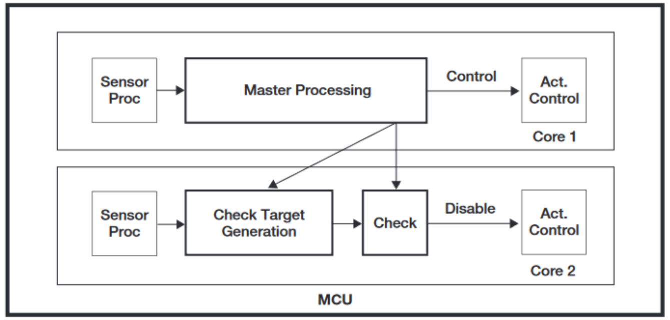


Source: https://www.edn.com/redundancy-for-safety-compliant-automotive-other-devices/

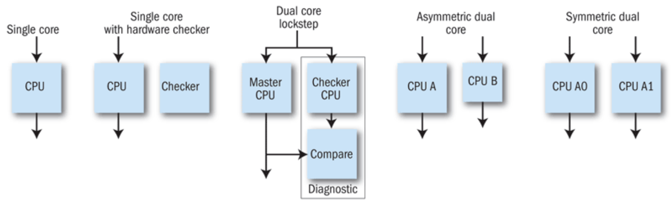


Asymmetric design

- **Asymmetric** two processors of different hardware architectures
  - Separate checking software is executed that controls if the software is correctly working
    - Easier to certify
  - Makes easy the detection of issues in both CPU and software through diverse hardware and software implementations (because the architectures are different)
  - The performance delta between the two cores can difficult the safety assessment

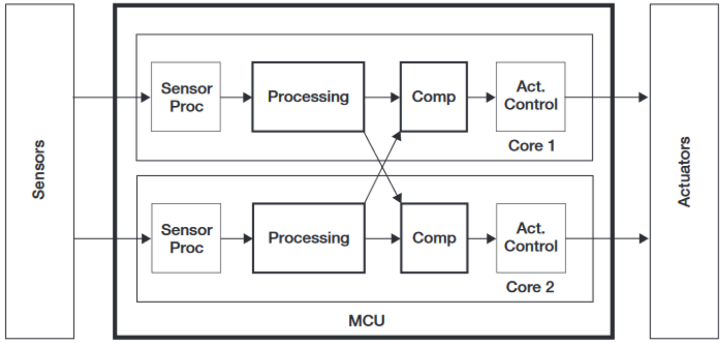


Source: NXP



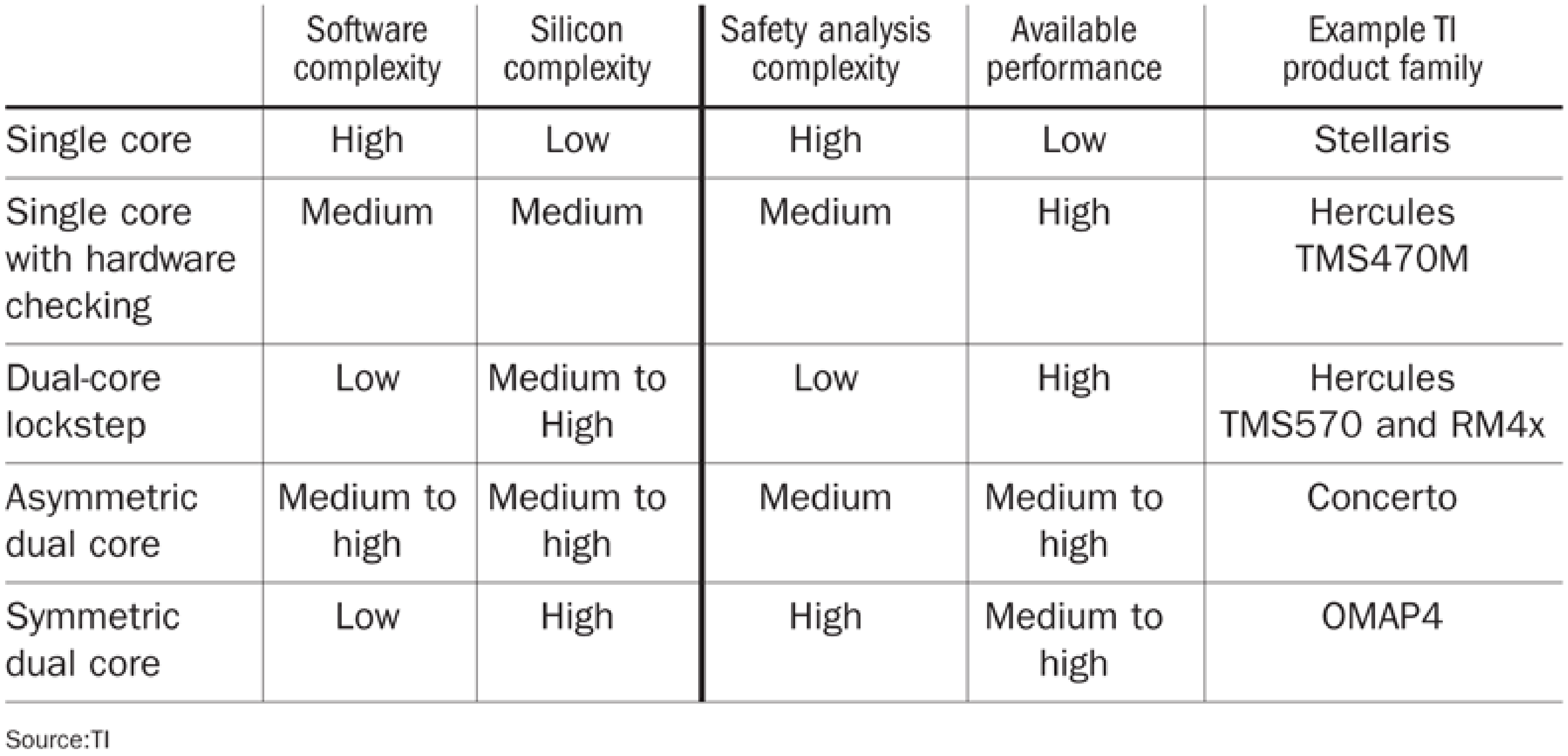
Symmetric designs

- **Symmetric** two instances of the software are executed on the cores using the same inputs of the same hardware architecture
  - Two instances of the same software are executed
    - diverse algorithms can also be used (+safety)
  - **Results are compared using software**
  - **Cores are not synchronized as lockstep**
  - **In lockstep software sees only one core**



Source: NXP

Summary safety design

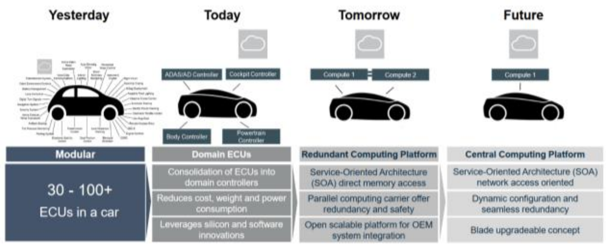


	Software complexity	Silicon complexity	Safety analysis complexity	Available performance	Example TI product family
Single core	High	Low	High	Low	Stellaris
Single core with hardware checking	Medium	Medium	Medium	High	Hercules TMS470M
Dual-core lockstep	Low	Medium to High	Low	High	Hercules TMS570 and RM4x
Asymmetric dual core	Medium to high	Medium to high	Medium	Medium to high	Concerto
Symmetric dual core	Low	High	High	Medium to high	OMAP4

Source:TI

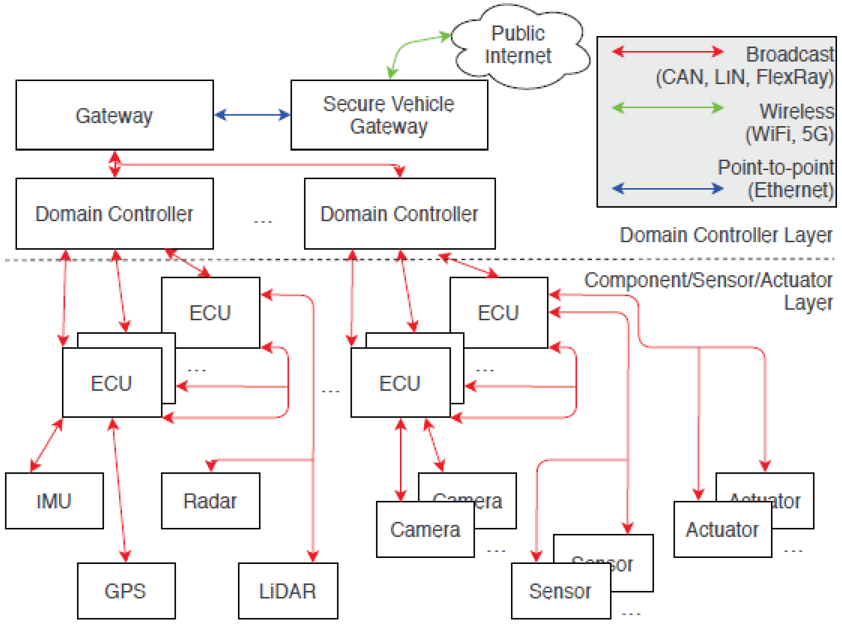
Source: https://www.eetimes.com/matching-processor-safety-strategies-to-your-system-design/

Domain/Zonal architecture



Yesterday	Today	Tomorrow	Future
Modular	Domain ECUs	Redundant Computing Platform	Central Computing Platform
30 - 100+ ECUs in a car	Consolidation of ECUs into domain controllers	Service-Oriented Architecture (SOA) direct memory access	Service-Oriented Architecture (SOA) network access oriented
	Reduces cost, weight and power consumption	Parallel computing carrier offer redundancy and safety	Dynamic configuration and seamless redundancy
	Leverages silicon and software innovations	Open scalable platform for OEM system integration	Blade upgradeable concept

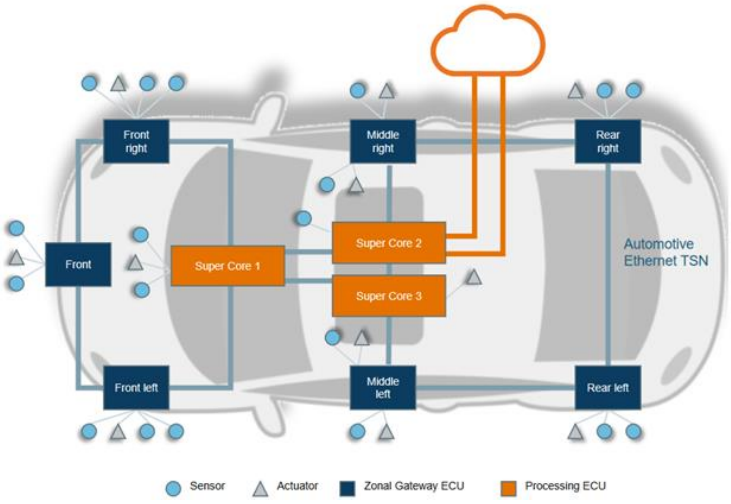
- The current distribute E/E automotive architecture has reached its computational limits
- **Zonal architecture**: for each subsystem there is a dedicated Electronic Control Units (ECU)
  - Layered architectural style for grouping ECUs
    - Chassis, Power Train, ADAS, infotainment…
  - Each domain has:
    - **Vehicle Server** - ("high" performance ECU)
    - **Zonal Gateway** - local connectivity hub that manages data through a single high-speed Ethernet link to the backbone
  - Focus on software qualities as scalability, robustness, and maintainability

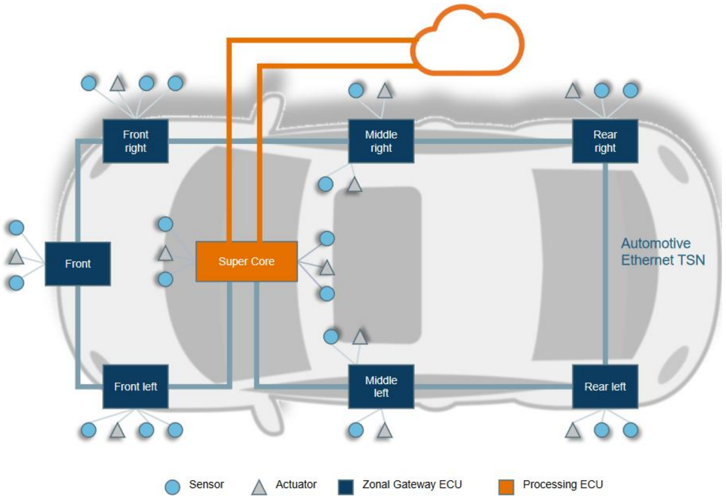


<u>Example</u>: Volvo XC90

Domain/Zonal architecture

- Zonal architectures bear savings of 50% and more in length of the wiring harness
- Power efficiency
- Increase of savings with larger number of sensors/actuators





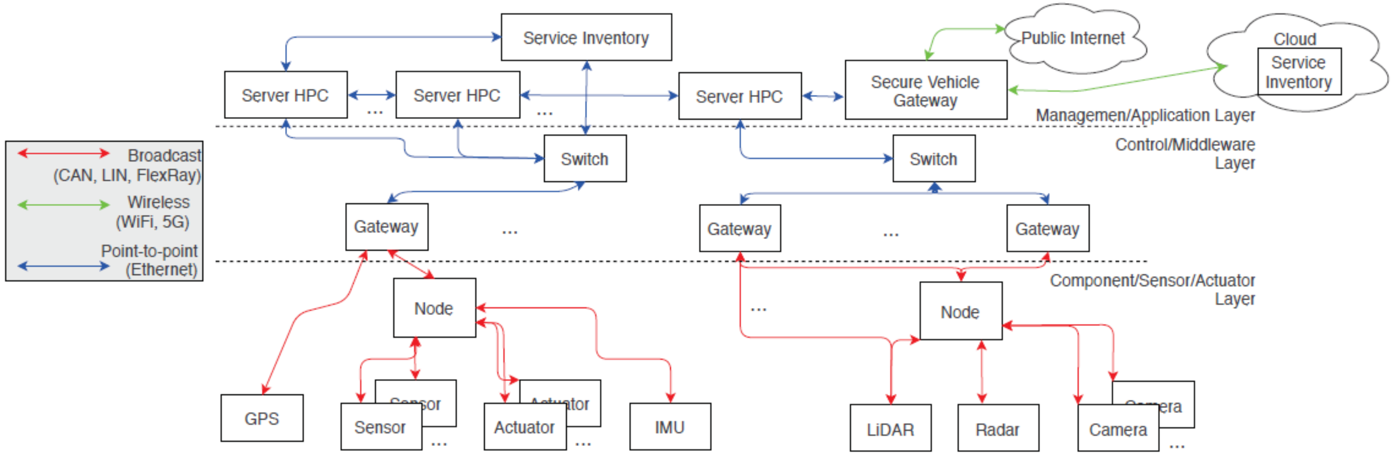
Source: visteon & continental

Vehicle centralised architecture

› High-performance server acts as the brain of the vehicle

› Instead of using domains, this architecture employs a layered architecture (Component/Sensor/ Actuator Layer, Control/Middleware Layer, and Management/ Application Layer)

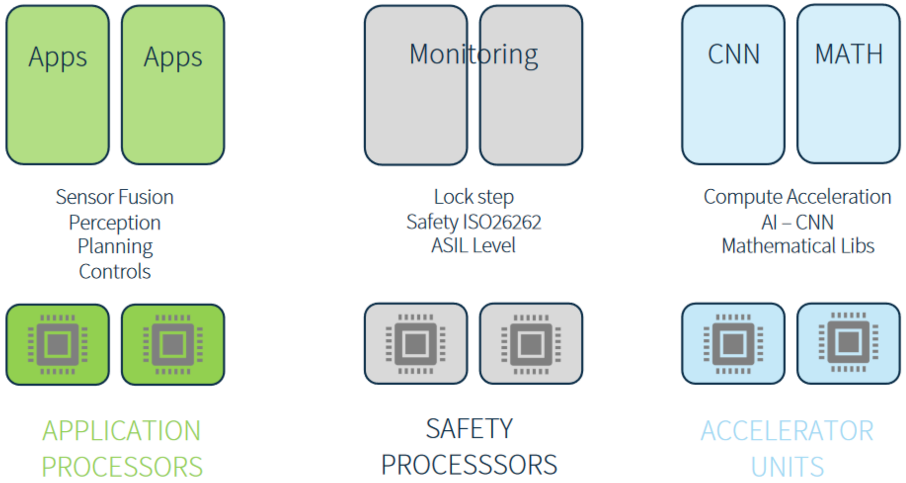
  – Exploits artificial intelligence, neural networks, cloud, and software over-the air (SOTA) updates



- BMW
- Volvo cars (SPA2)

ECU types

- Different Electronic Control Units (ECU) can include different hardware
- The functionality to support determines the hardware used



Apps | Apps

Sensor Fusion
Perception
Planning
Controls

APPLICATION PROCESSORS

Monitoring

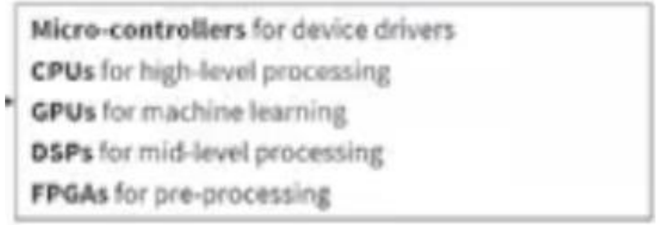
Lock step
Safety ISO26262
ASIL Level

SAFETY PROCESSSORS

CNN | MATH

Compute Acceleration
AI – CNN
Mathematical Libs

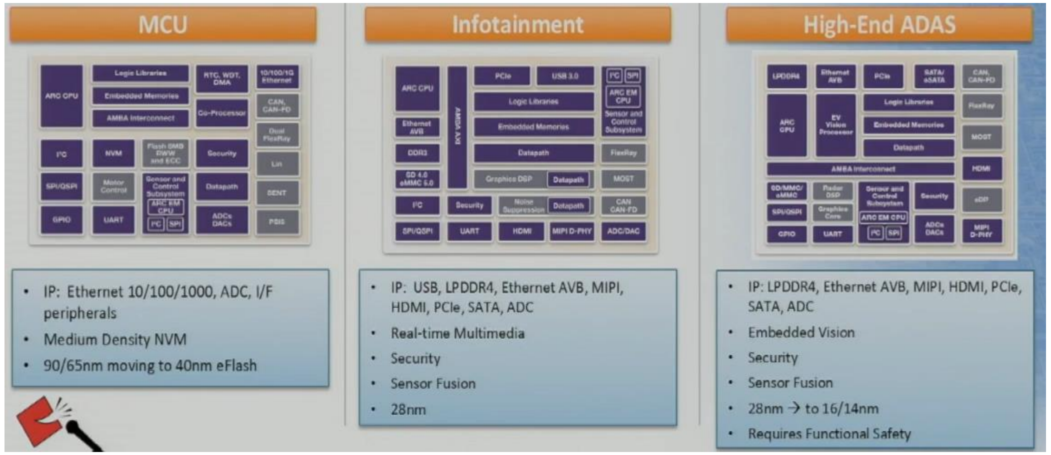
ACCELERATOR UNITS



Micro-controllers for device drivers
CPUs for high-level processing
GPUs for machine learning
DSPs for mid-level processing
FPGAs for pre-processing

ECU types

- **MCU (Microcontroller Unit)**
  - Is a small computer on a single metal-oxide-semiconductor (MOS) integrated circuit (IC) chip
    - Single-Core/Multi-Core
- **Infotainment (IVI In-vehicle infotainment)**
  - Combination of vehicle systems which are used to deliver entertainment and information to the driver and passengers using audio/video interfaces
    - FPGA, GPU, ASIC, Multi-Core...
- **High-End ADAS**
  - High performance architectures designed to implement compute intensive ADAS applications (perception, localization etc...)
    - FPGA, GPU, ASIC, Multi-Core...



MCU

- IP: Ethernet 10/100/1000, ADC, I/F peripherals
- Medium Density NVM
- 90/65nm moving to 40nm eFlash

Infotainment

- IP: USB, LPDDR4, Ethernet AVB, MIPI, HDMI, PCIe, SATA, ADC
- Real-time Multimedia
- Security
- Sensor Fusion
- 28nm

High-End ADAS

- IP: LPDDR4, Ethernet AVB, MIPI, HDMI, PCIe, SATA, ADC
- Embedded Vision
- Security
- Sensor Fusion
- 28nm → to 16/14nm
- Requires Functional Safety

Source: ChipEstimate

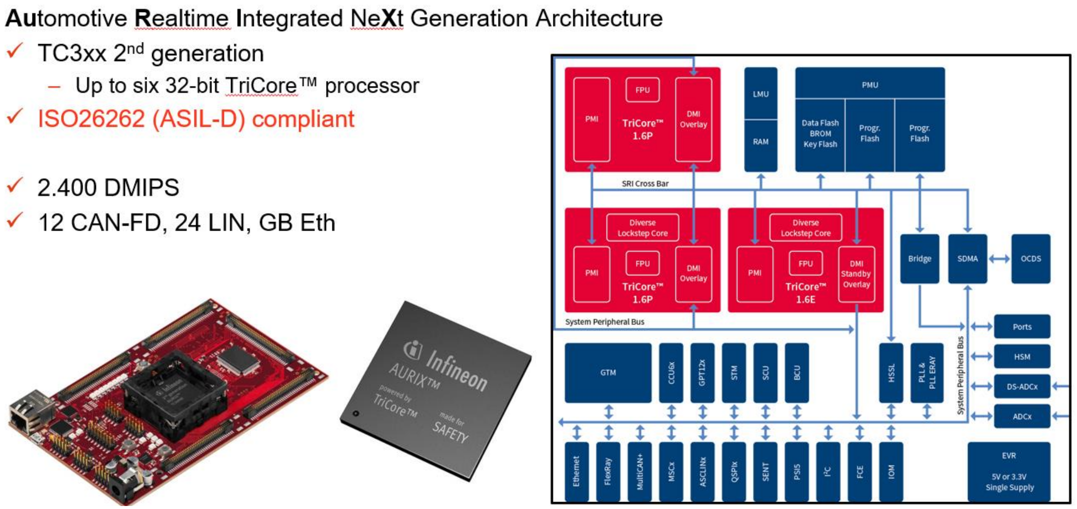
Microcontroller/MCU

- A microcontroller (MCU or µC) is essentially a single-chip which incorporates a microprocessor
- A microcontroller contains one or more **CPUs** (processor cores) along with memory and programmable input/output peripherals
  - Some microcontrollers execute code directly from the flash memory to execute code fast after powering up
- Microcontrollers are generally used to execute small purpose-built programs in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls….

Commercial reference product: **Aurix Tricore**

MCU (Infineon Aurix)

- Contains up to three cores
- Dual-lockstep processors implemented with additional architectural diversity
  - The two cores run the same code
  - The cores have different hardware design to reduce the probability of common cause error

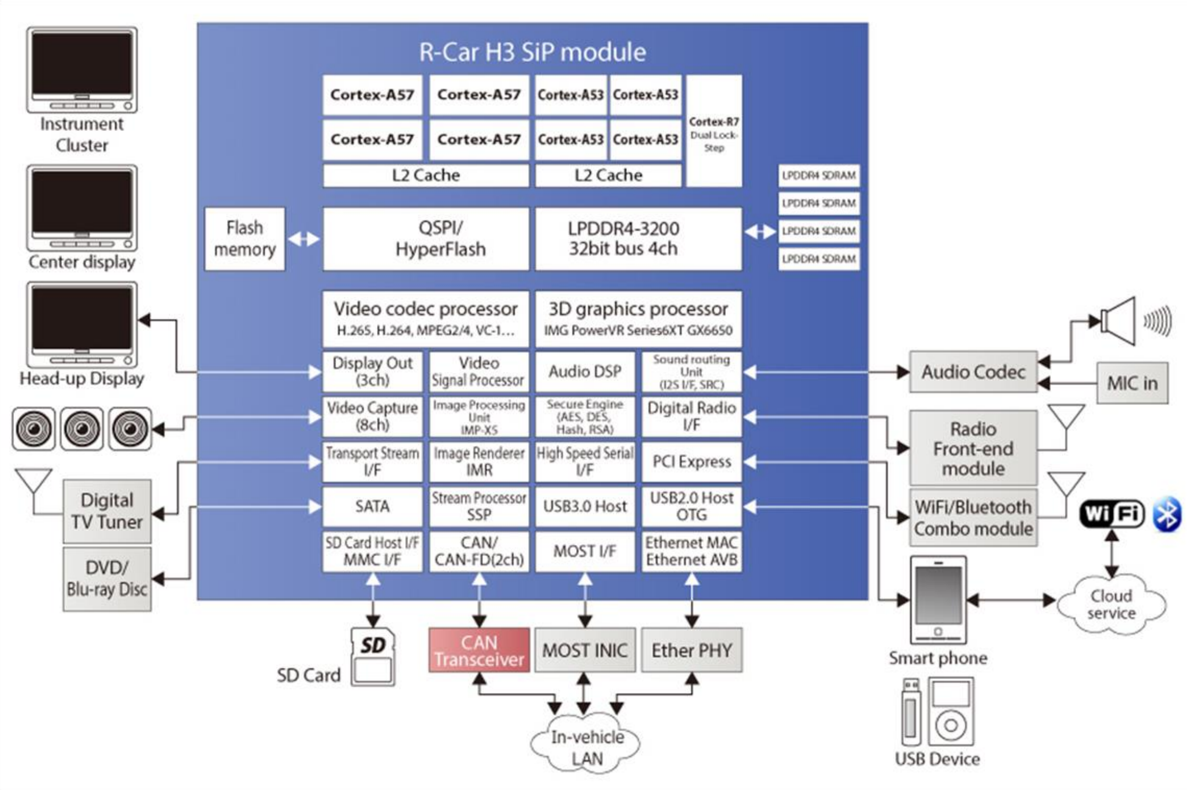


Automotive **R**ealtime **I**ntegrated Ne**X**t Generation Architecture

- ✓ TC3xx 2nd generation
  - Up to six 32-bit TriCore™ processor
- ✓ ISO26262 (ASIL-D) compliant
- ✓ 2.400 DMIPS
- ✓ 12 CAN-FD, 24 LIN, GB Eth

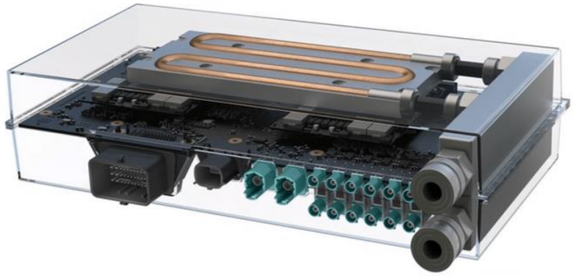
MCU (Renesas R-Car3)

- ✓ R-Car H3/M3
  - ASIL B
  - Quad A57 + Quad A53 + Dual R7
  - Imagination Tech GPU (PowerVR)
- ✓ RH850/P1X Series
  - ASIL-D MCUs
- ✓ R-Car V3M SoC
  - ASIL C for computer vision
  - Dual A53 + Dual R7 + CV engines
  - Samples available
  - Mass production mid 2019
- ✓ Adopted by Denso, Toyota



GPU

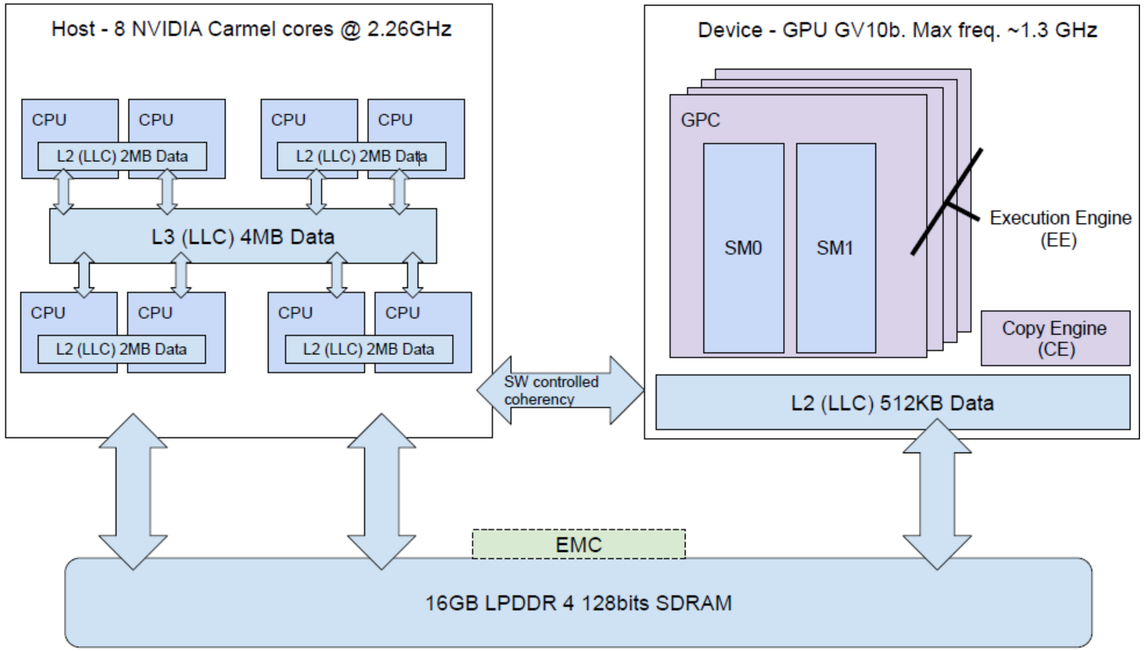
- A Graphics Processing Unit (**GPU**) is a programmable logic chip specifically designed for rendering graphics and displaying on electronic devices
- It is characterized by the host-to-device paradigm
- For many critical applications such as pedestrian detection, line following, and path planning the Graphic Processing Unit (**GPU**) is the most popular choice for obtaining orders of magnitude increases in performance at modest power consumption
- The programming language used is **CUDA**



Commercial reference product: **DRIVE PX Parker AutoChauffeur (ASIL-C)**

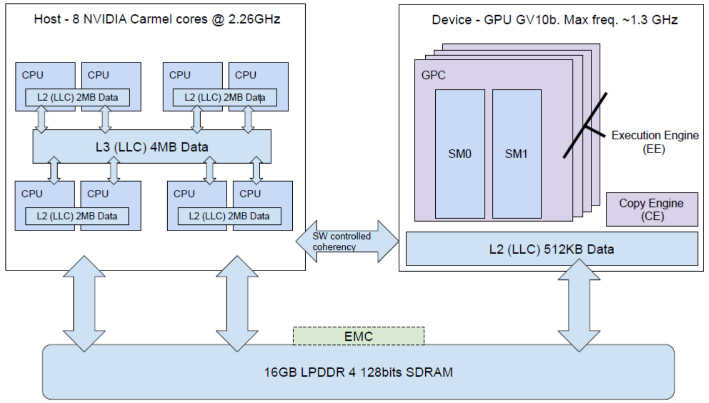
GPU hardware model

- The typical execution pattern of a heterogeneous CPU-GPU system:
  - 1) move data from the host CPU to the GPU device
  - 2) execute the kernel on the GPU device
  - 3) move data from the device back to the host

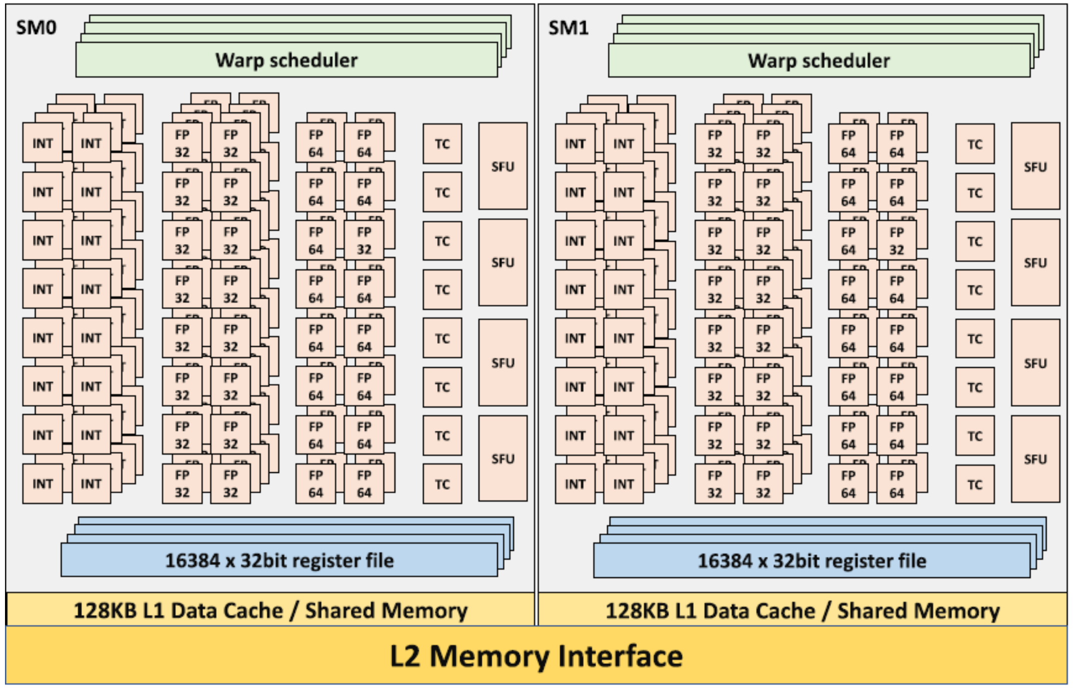


Dissecting the CUDA scheduling hierarchy: a Performance and Predictability Perspective. Ignacio Sañudo, Nicola Capodieci, Jorge Martinez, Andrea Marongiu, Marko Bertogna

GPU hardware model



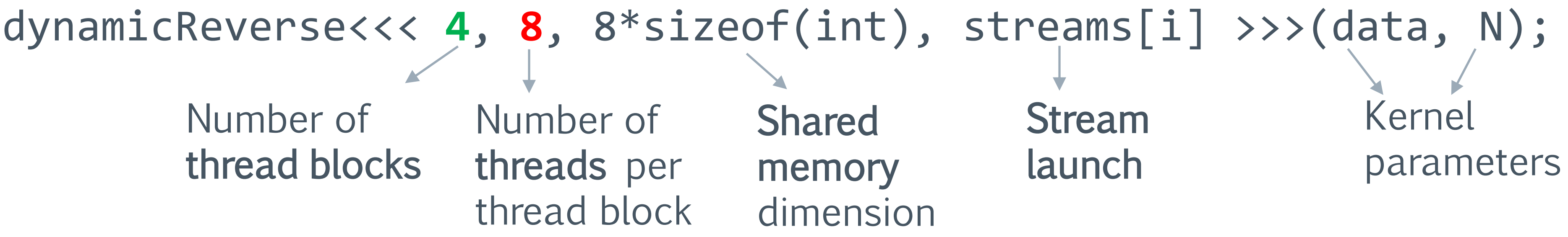
- Xavier – Volta microarchitecture
- Copy engine
- Execution Engine
  - 8 Streaming multiprocessors (SMs)
  - 64 CUDA cores per SM
  - 2048 maximum schedulable warps
  - 4 warp schedulers
- SMs are grouped into graphic processing clusters (GPCs)



SM0
SM1
Warp scheduler
INT
FP 32
FP 64
TC
SFU
16384 x 32bit register file
128KB L1 Data Cache / Shared Memory
L2 Memory Interface

CUDA software model

- CUDA is the parallel computing API provided by NVIDIA that gives access to the GPU's instruction set



```
dynamicReverse<<< 4, 8, 8*sizeof(int), streams[i] >>>(data, N);
```

Number of **thread blocks**

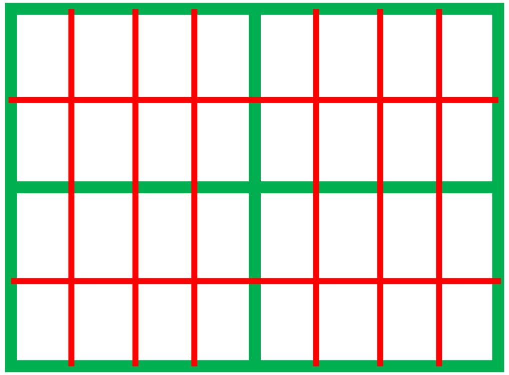
Number of **threads** per thread block

Shared memory dimension

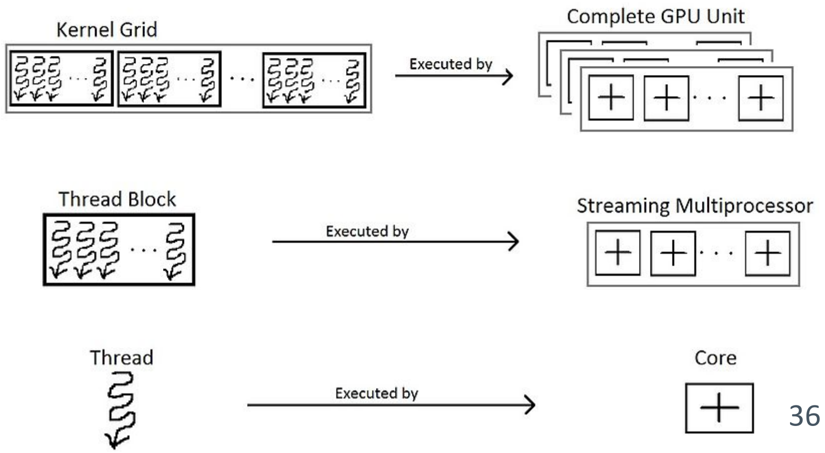
Stream launch

Kernel parameters

```
dynamicReverse<<<
```

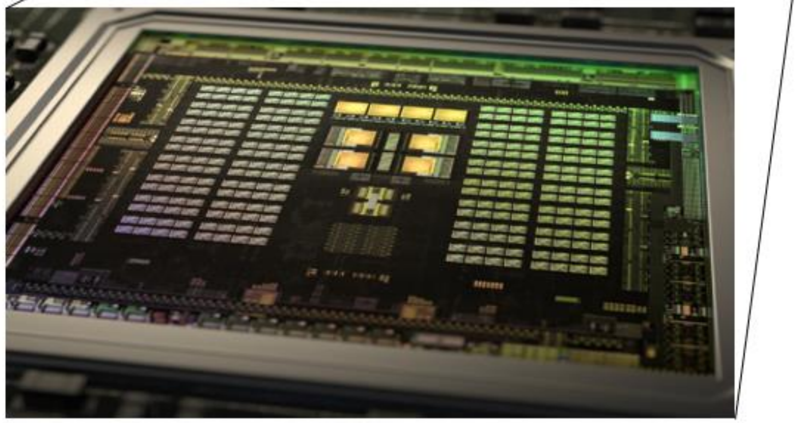


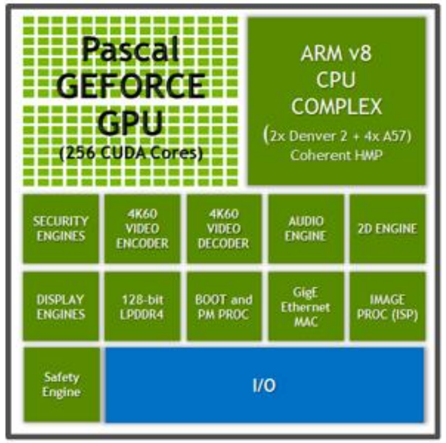
```
>>>
```



GPU (NVIDIA Drive PX2)

- ✓ Base SoC for Nvidia Drive PX2
  - – “Autocruise” version for highway automated driving
- ✓ Pascal GPU
  - – 256 cores
- ✓ 64bit ARM v8
  - – 2 Denver + 4 A57
- ✓ 3 x 4k Video @ 60fps
- ✓ 12 x camera input
  - – 1.4 GP/s
- ✓ CAN, eAVB, USB3, …
- ✓ ASIL B safety processor
  - – R5 lockstep





FPGA

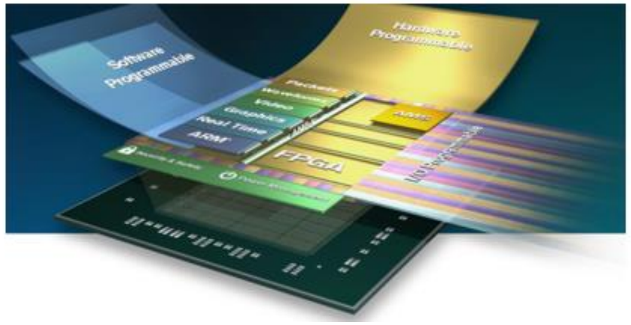
- A field-programmable gate array (**FPGA**) is an integrated circuit designed to be configured by a customer or a designer after manufacturing
  - FPGAs contain an array of programmable logic blocks, and a hierarchy of "reconfigurable interconnects" that allow the blocks to be "wired together", like many logic gates that can be inter-wired in different configurations
- FPGAs are becoming increasingly important in many domains; they are used to perform ‘number cruncher’ operations (for example the execution of deep neural networks)
- Traditionally HDL (Verilog, VHDL); newer systems include C/C++ via openCL or OpenMP
  - You can also use HLS

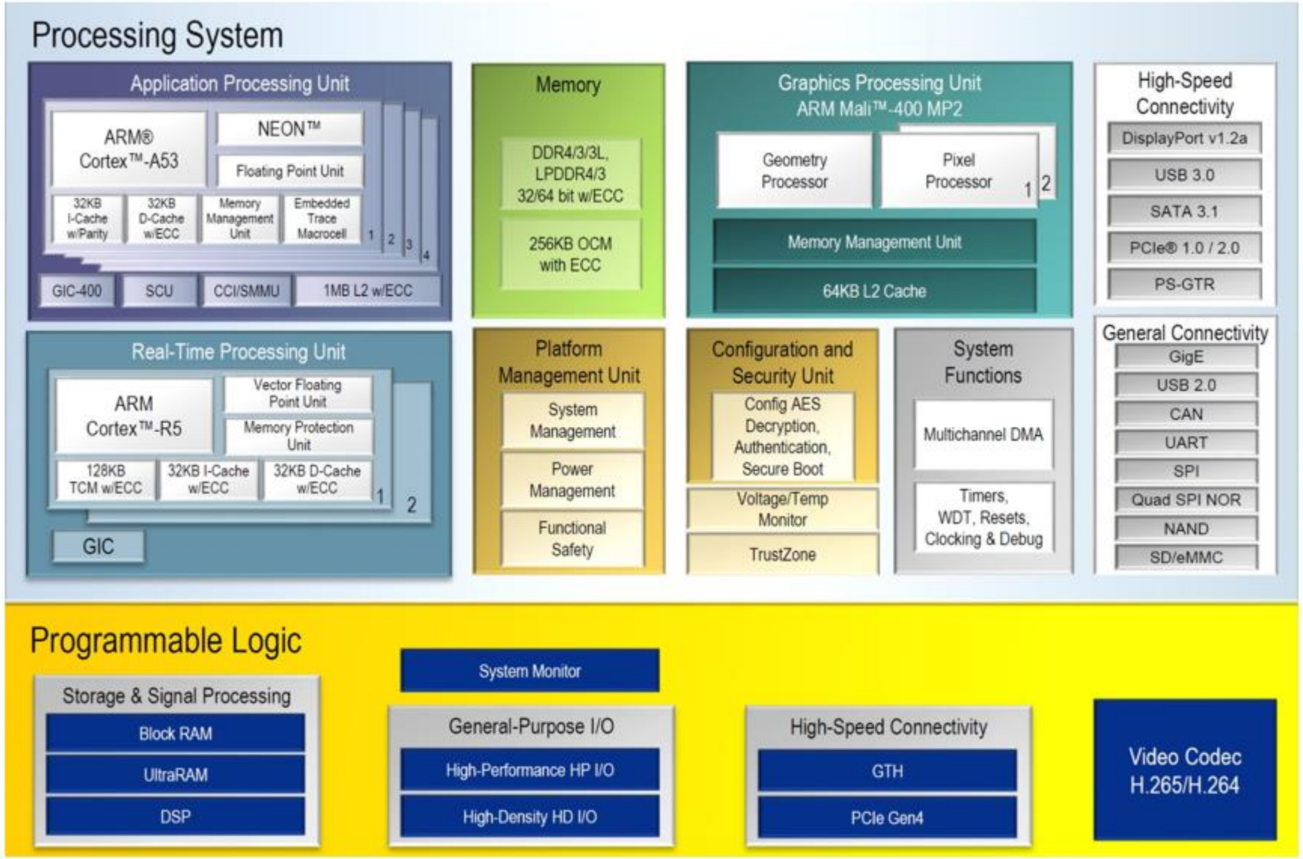
Commercial reference product: **Xilinx Zynq Ultrascale+ (ASIL-C)**

FPGA (Zync Ultrascale +)

- ✓ Big.LITTLE ARM Complex
  - – 4 x A53 + 2 x R5 (lockstep)
- ✓ State-of-the-art FPGA
- ✓ ARM Mali GPU
- ✓ Rich connectivity
- ✓ CAN, GigE

- ✓ Adopted by Autoliv, Mercedes, etc.





Processing System

Application Processing Unit
ARM® Cortex™-A53
NEON™
Floating Point Unit
32KB I-Cache w/Parity
32KB D-Cache w/ECC
Memory Management Unit
Embedded Trace Macrocell
1 2 3 4
GIC-400
SCU
CCI/SMMU
1MB L2 w/ECC

Memory
DDR4/3/3L, LPDDR4/3 32/64 bit w/ECC
256KB OCM with ECC

Graphics Processing Unit ARM Mali™-400 MP2
Geometry Processor
Pixel Processor 1 2
Memory Management Unit
64KB L2 Cache

High-Speed Connectivity
DisplayPort v1.2a
USB 3.0
SATA 3.1
PCIe® 1.0 / 2.0
PS-GTR

Real-Time Processing Unit
ARM Cortex™-R5
Vector Floating Point Unit
Memory Protection Unit
128KB TCM w/ECC
32KB I-Cache w/ECC
32KB D-Cache w/ECC
1 2
GIC

Platform Management Unit
System Management
Power Management
Functional Safety

Configuration and Security Unit
Config AES Decryption, Authentication, Secure Boot
Voltage/Temp Monitor
TrustZone

System Functions
Multichannel DMA
Timers, WDT, Resets, Clocking & Debug

General Connectivity
GigE
USB 2.0
CAN
UART
SPI
Quad SPI NOR
NAND
SD/eMMC

Programmable Logic

Storage & Signal Processing
Block RAM
UltraRAM
DSP

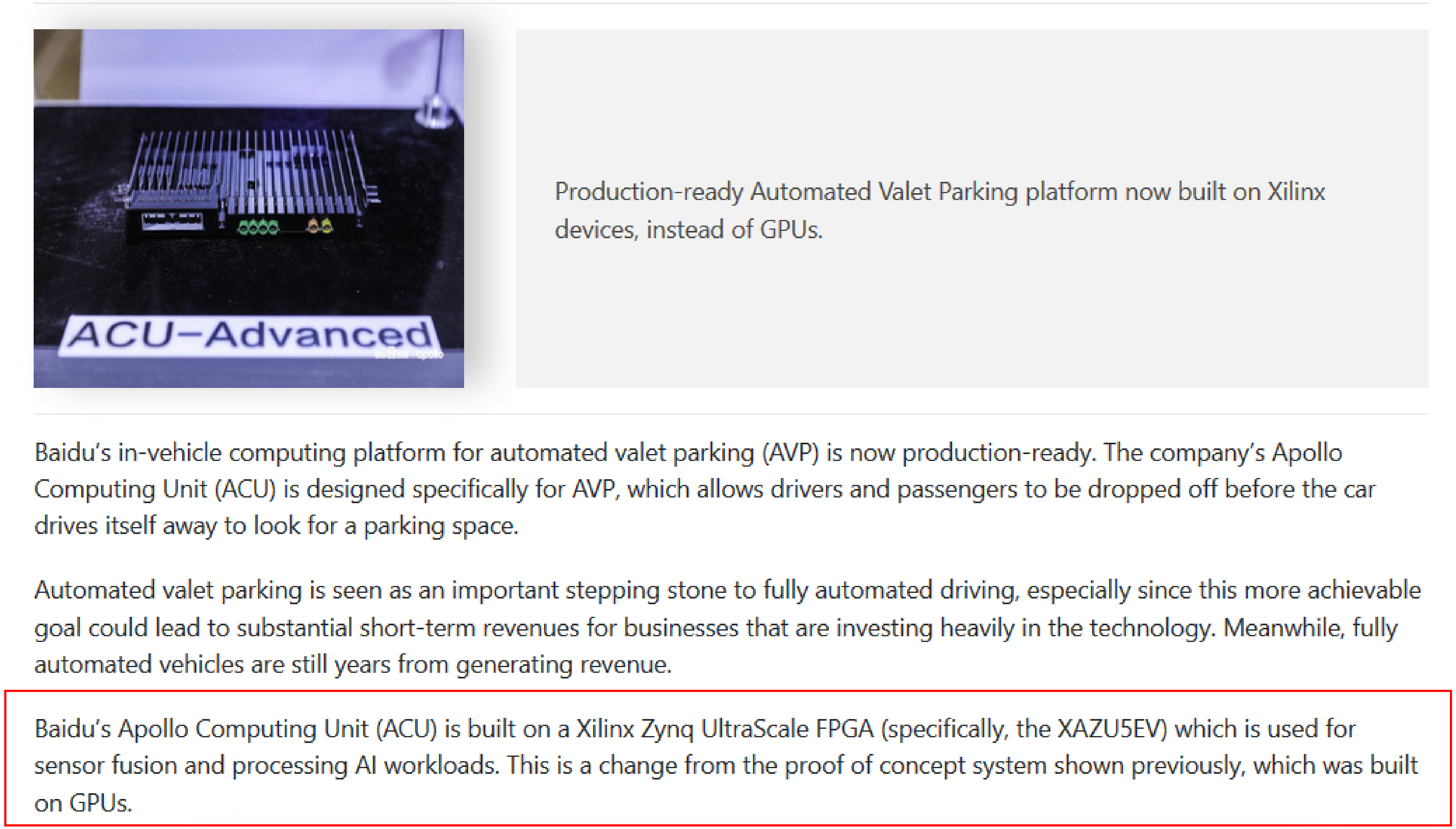
System Monitor

General-Purpose I/O
High-Performance HP I/O
High-Density HD I/O

High-Speed Connectivity
GTH
PCIe Gen4

Video Codec H.265/H.264

FPGA (Zync Ultrascale +)



Production-ready Automated Valet Parking platform now built on Xilinx devices, instead of GPUs.

Baidu's in-vehicle computing platform for automated valet parking (AVP) is now production-ready. The company's Apollo Computing Unit (ACU) is designed specifically for AVP, which allows drivers and passengers to be dropped off before the car drives itself away to look for a parking space.

Automated valet parking is seen as an important stepping stone to fully automated driving, especially since this more achievable goal could lead to substantial short-term revenues for businesses that are investing heavily in the technology. Meanwhile, fully automated vehicles are still years from generating revenue.

Baidu's Apollo Computing Unit (ACU) is built on a Xilinx Zynq UltraScale FPGA (specifically, the XAZU5EV) which is used for sensor fusion and processing AI workloads. This is a change from the proof of concept system shown previously, which was built on GPUs.

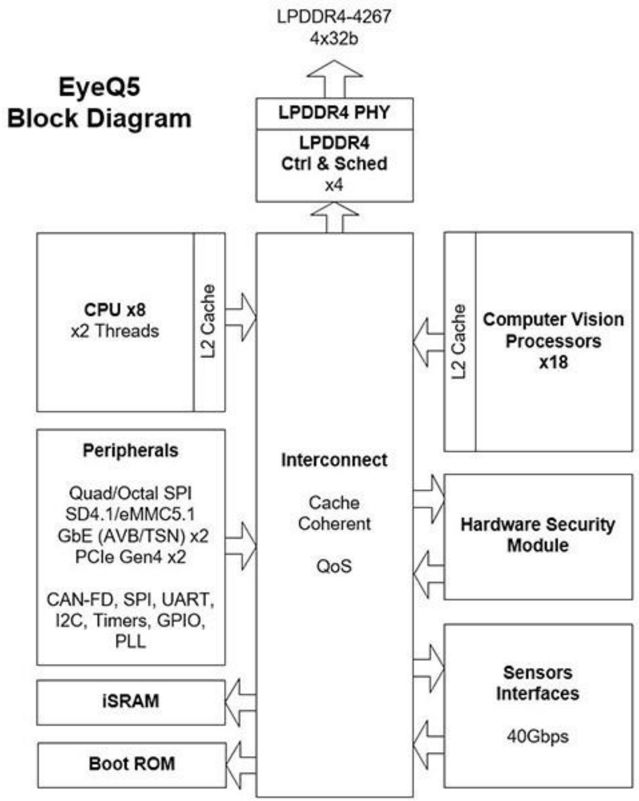
ASIC

- An Application-Specific Integrated Circuit (**ASIC**), is an integrated circuit designed to execute software with a **specific purpose**
  - Simple tasks like encoders to complex tasks like pedestrian detection etc…
- They can contain over two billion transistors (e.g, TPU architecture from Google)
- ASICs are mainly designed for the execution of applications with **high performance and low power consumption constraints**. However, the most important drawback is that **they are not reconfigurable**, therefore, **if changes want to be applied the entire chip must be redesigned**
- Designers of digital ASICs often use a hardware description language (HDL), such as Verilog or VHDL, to describe the functionality of ASICs

Commercial reference product: **Mobileye EyeQ3 (ASIL-D)/Tesla new chip**

ASIC (Mobileye EyeQ5)

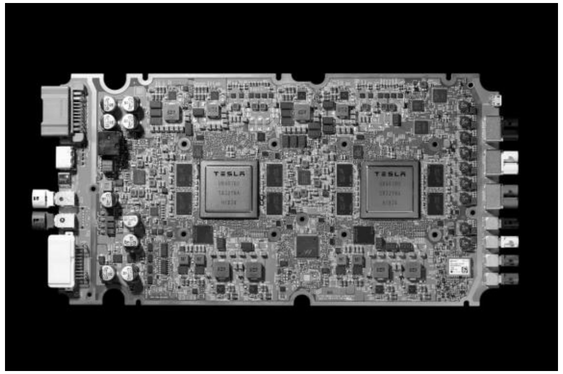
- "Mobileye's system-on-chip (SoC) – the EyeQ® family – provides the processing power to support a comprehensive suite of ADAS functions based on a single camera sensor
  - It provides different ADAS functions, for instance: – *Auto Emergency Braking (AEB), Lane Departure Warning (LDW), Forward Collision Warning (FCW), Traffic Sign Recognition (TSR), or Intelligent High-beam Control (IHC) among many other ADAS functions*
- Mobileye's approach is based on the use of a single camera that must be mounted on the windshield. The camera feeds a sequence of images into the Mobileye SoC, the SoC will process the images giving as output a set of labels that identify the objects within the road
- In 2018, Intel acquired Mobileye. The new version of the EyeQ family, the EyeQ4, will be launched at the end of 2018. Intel-Mobileye are currently developing the EyeQ5, to act as the vision central computer and targeting to SAE level 5, it will be released in 2020



https://www.youtube.com/watch?v=kJD5R_yQ9aw

ASIC (Tesla)

- Full Self-Driving Chip (FSD Chip, previously Autopilot Hardware 3.0) is an autonomous driving chip designed by Tesla and introduced in early 2019 for their own cars
  - Fabricated on Samsung's 14 nm process technology
  - 3 quad-core Cortex-A72 clusters for a total of 12 CPUs operating at 2.2 GHz
  - a GPU operating 1 GHz
  - **2 neural processing units operating at 2 GHz, and various** other hardware accelerators



Source: https://en.wikichip.org

Pros and cons (automotive)

	MCU	GPU	FPGA	ASIC
Computing power	Moderate	Good/Very good	Good/Very good	Good/Very good
Programmability	Very good	Good	Not bad	Poor
Power efficiency	Not bad	Poor	Good	Very good
Flexibility	Good	Very good	Not bad	Poor

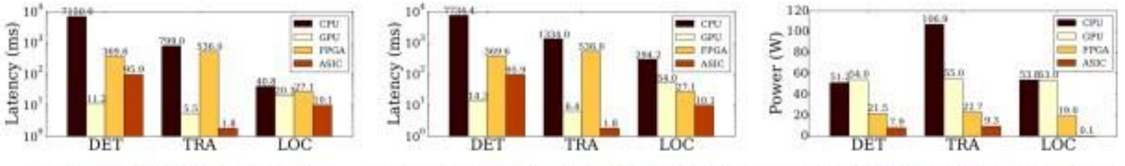
Pros and cons (general purpose)

Applications	CPU	FPGA	GPU	ASIC	Comments
Vision & image processing		✓	✓	✓	FPGA may give way to ASIC in high-volume applications
AI training			✓		GPU parallelism well-suited for processing terabyte data sets in reasonable time
AI inference	✓	✓	✓	✓	Everyone wants in! FPGAs perhaps leading; high-end CPUs (e.g., Intel's Xeon) and GPUs (e.g., Nvidia's T4) address this market
High-speed Search	✓	✓	✓	✓	Microsoft's Bing uses FPGAs; Google uses TPU ASIC; CPU needed for coordination & control
Industrial motor control	(✓)	✓		✓	Many motor-control MCUs and ASICs available; FPGAs offer a quick-turn ASIC alternative
Supercomputer HPC	✓		✓		Majority of TOP500 supercomputers uses some combination of CPUs and GPUs
General-purpose computing	✓		(✓)		CPU most versatile, flexible option; GPUs beginning to perform some tasks
Embedded control	✓	✓		✓	CPUs (-> MCU) dominant in low-cost, space-constrained, low-power, mobile applications
Prototyping, low-volume		✓			FPGAs best choice for low-volume, high-end applications; also pre-silicon validation, post-silicon validation and firmware development

Source: https://www.arrow.com

Silicon war

- **Different metrics** can be contemplated when considering the computing platform
  - **Latency** is how long it takes the software from the input of a signal to the decision
  - **Power** is how much electrical energy it takes to make that decision
- **In mission critical systems** the most important property is **determinism**
  - **Low-latency and low jitter** (i.e., the difference between the best and worst-case latency) are needed
  - But also the power consumption is important



(a) Mean Latency Across Platforms (b) 99.99th-Percentile Latency Across Platforms (c) Power Consumption Across Platforms

Figure 10. Acceleration results across various accelerator platforms. The latency of running DET or TRA alone on CPUs or FPGAs has already exceeded the end-to-end latency constraints, which suggests they are not viable candidates for running the complex DNN-based DET and TRA algorithms that demand large amount of computational resources.

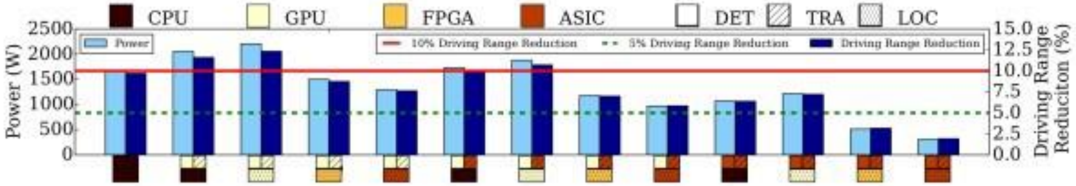


Figure 12. The power consumption and the corresponding driving range reduction of running different algorithmic components across different configurations. Configurations equipped with GPUs consume significant amount of power and reduce the driving range up to 12% while ASICs approaches can achieve efficiency which only reduce the driving range by 2%.

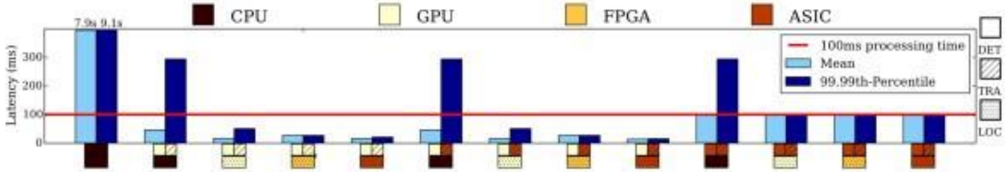


Figure 11. The mean and 99.99th-percentile latency of running different algorithmic components across different configurations denoted on x-axis. For each configuration, the color of each grid represents the computing platform each algorithm is running on (e.g., a red dotted grid represents running LOC on ASICs). There are several configurations that meet the performance constraints at tail latency of 100 ms, which means accelerator-based designs are viable for autonomous driving systems.

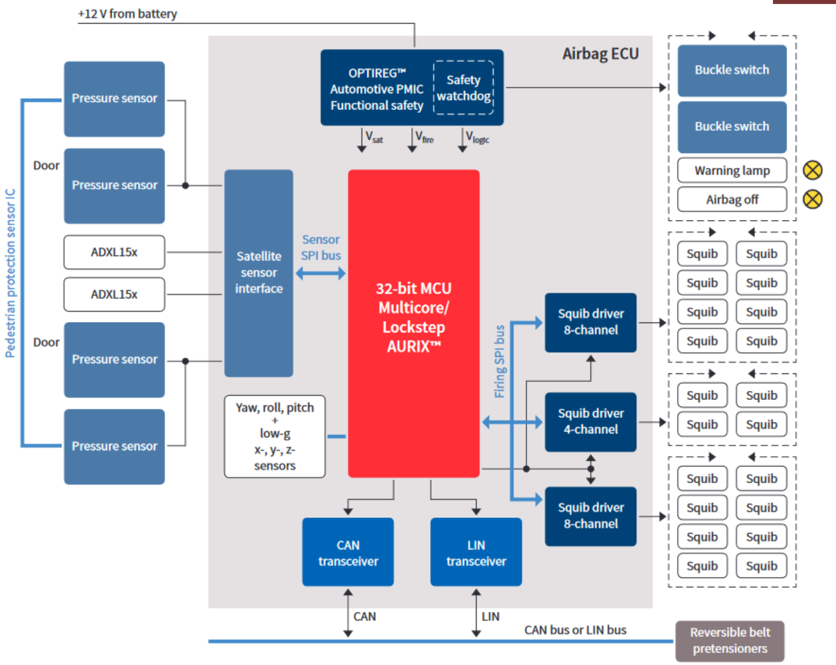
Source: The Architectural Implications of Autonomous Driving: Constraints and Acceleration. Shih-Chieh Lin et al

Use cases

- Now let’s discuss about different use case and the platforms that we can use for each of them
  - **MCU (Microcontroller Unit)**
  - **Infotainment (IVI In-vehicle infotainment)**
  - **High-End ADAS (Tesla Model 3)**

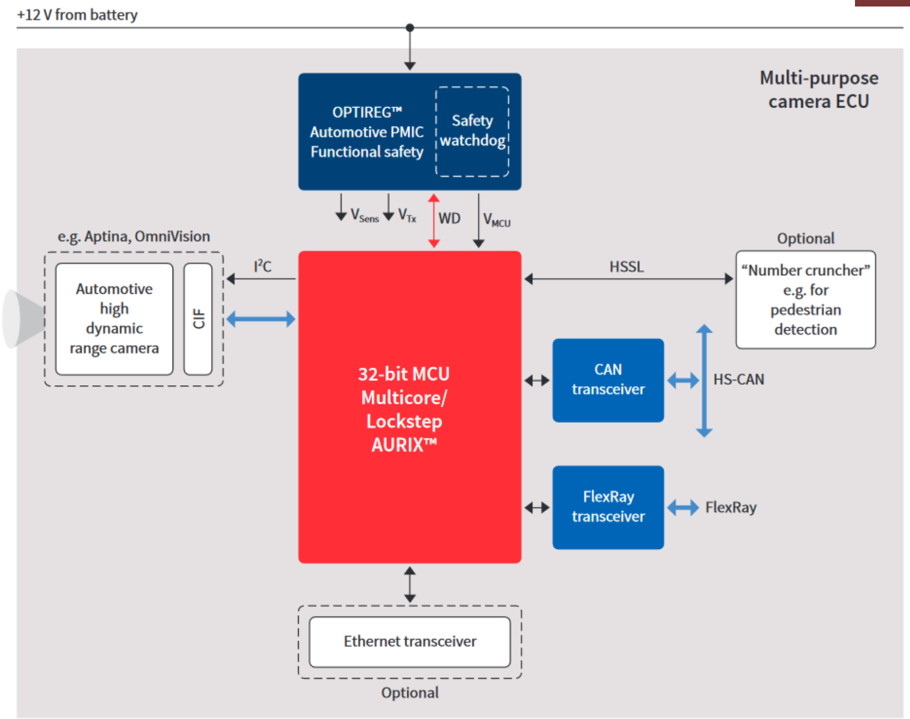
MCU - Airbag

- An airbag is a vehicle occupant-restraint system using a bag designed to inflate extremely quickly, then quickly deflate during a collision
- It consists of the airbag cushion, a flexible fabric bag, an inflation module, and an impact sensor
- The **squib** is a detonator wire used for example to ignite the explosive charge that inflates the **airbag** in the automobile



MCU – Camera system

- The camera system enables advanced driver assistance functions such as lane departure warning, forward collision warning, along with traffic sign and pedestrian recognition
  - To do so it is needed a “**number cruncher**” system



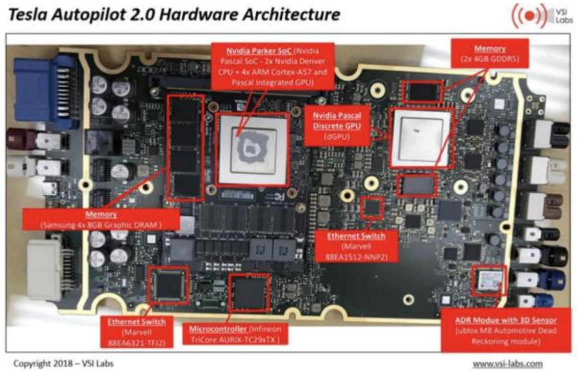
Infotainment – HUD/IHU

- **Integrated Head-Unit:** In-vehicle infotainment head unit is a touch screen based, tablet-like device, mounted on the vehicle's dashboard
  - You can see the output of the cameras, use it to reproduce media etc…
- **Heads-Up Display:** Automotive heads-up display is an integral part of high-end infotainment systems, which displays the vehicle's real-time information on the transparent screen integrated with the vehicle's windshield.
  - Heads-up display helps in reducing the driver's distraction while driving and assists him with key details like speed, navigation maps, etc...

- Qualcomm® Snapdragon™ 600 processor (Eragon 600 SoM)
- TI- Jacinto™ DRAx infotainment SoCs
- NXP – MX255 applications processor
- Samsung Exynos Auto V9

https://www.einfochips.com

Tesla Model 3

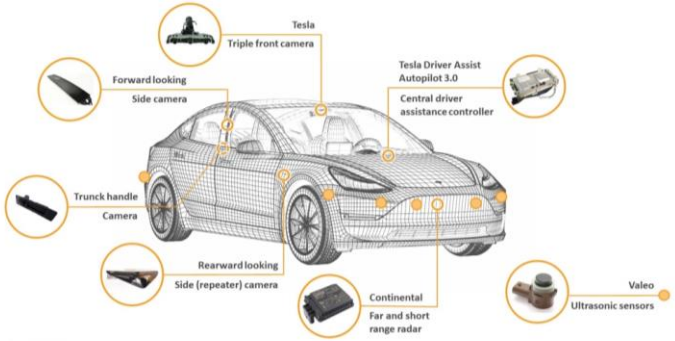


Tesla Autopilot 2.0 Hardware Architecture
VSI Labs
Nvidia Parker SoC (Nvidia Pascal SoC - 2x Nvidia Denver CPU + 4x ARM Cortex-A57 and Pascal Integrated GPU)
Memory (2x 4GB GDDR5)
Nvidia Pascal Discrete GPU (dGPU)
Memory (Samsung 4x 8GB Graphic DRAM)
Ethernet Switch (Marvell 88EA1512-NNP2)
Ethernet Switch (Marvell 88EA6321-TFJ2)
Microcontroller (Infineon TriCore AURIX-TC29xTX)
ADR Modue with 3D Sensor (ublox M8 Automotive Dead Reckoning module)
Copyright 2018 – VSI Labs
www.vsi-labs.com

- **All Tesla Motors vehicles manufactured from mid-October 2016 include a Drive PX 2, which will be used for neural net processing to enable Enhanced Autopilot and full self-driving functionality**

- **Autopilot**
  - **Traffic-Aware Cruise Control**: Matches the speed of your car to that of the surrounding traffic
  - **Autosteer**: Assists in steering within a clearly marked lane, and uses traffic-aware cruise control

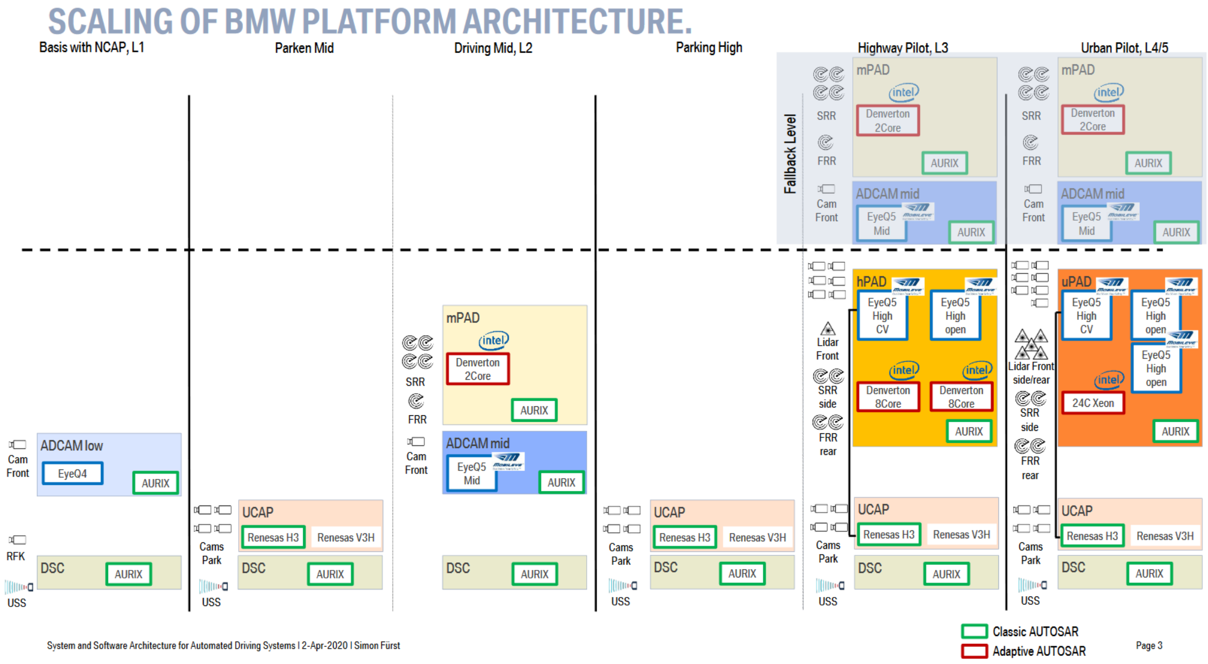
- **Full Self-Driving Capability**
  - **Navigate on Autopilot (Beta)**: Actively guides your car from a highway's on-ramp to off-ramp, including suggesting lane changes, navigating interchanges, automatically engaging the turn signal and taking the correct exit
  - **Auto Lane Change**: Assists in moving to an adjacent lane on the highway when Autosteer is engaged
  - **Autopark**: Helps automatically parallel or perpendicular park your car, with a single touch
  - **Summon**: Moves your car in and out of a tight space using the mobile app or key
  - **Smart Summon**: Your car will navigate more complex environments and parking spaces, maneuvering around objects as necessary to come find you in a parking lot.
  - **Traffic and Stop Sign Control (Beta)**: Identifies stop signs and traffic lights and automatically slows your car to a stop on approach, with your active supervision



https://www.tesla.com/support/autopilot

What is coming? BMW ADAS architecture

- Uncertain complexity of future AD solutions
  - **Design goals**: safety, performance, scalability and reusability of software and hardware
  - Dual processing architecture
    - Main and a fallback computer



SCALING OF BMW PLATFORM ARCHITECTURE.

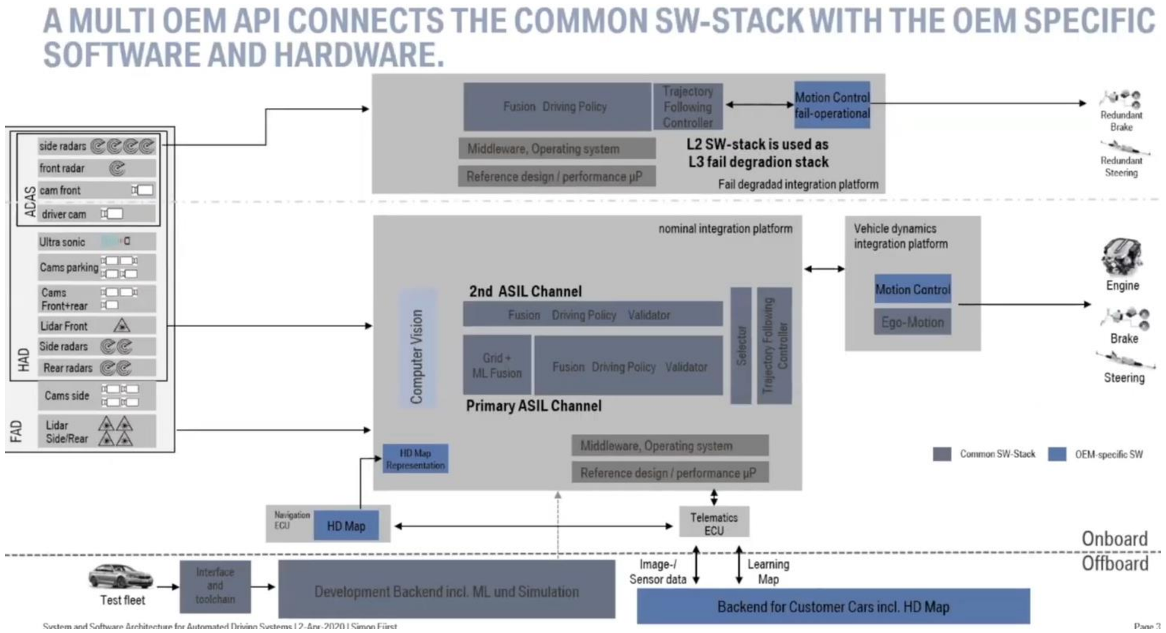
System and Software Architecture for Automated Driving Systems I 2-Apr-2020 I Simon Fürst

Page 3

Source: The Autonomous Safety & Architecture. Simon Furst. 2020

What is coming? BMW ADAS architecture

- **Dual processing architecture**
  - the primary channel calculates the **main trajectory** for the vehicle. The secondary channel is there to **supervise** the primary channel
  - When the channels find themselves in disagreement on a trajectory, the system goes into a **degraded** mode
    - This is handled by another processing block, driven by an independent power supply
    - It computes the minimum risk manouver



A MULTI OEM API CONNECTS THE COMMON SW-STACK WITH THE OEM SPECIFIC SOFTWARE AND HARDWARE.

ADAS
side radars
front radar
cam front
driver cam
HAD
Ultra sonic
Cams parking
Cams Front+rear
Lidar Front
Side radars
Rear radars
FAD
Cams side
Lidar Side/Rear
Fusion Driving Policy
Trajectory Following Controller
Motion Control fail-operational
Middleware, Operating system
Reference design / performance µP
L2 SW-stack is used as L3 fail degradion stack
Fail degradad integration platform
Redundant Brake
Redundant Steering
nominal integration platform
Computer Vision
2nd ASIL Channel
Fusion Driving Policy Validator
Grid + ML Fusion
Fusion Driving Policy Validator
Primary ASIL Channel
Selector
Trajectory Following Controller
Middleware, Operating system
Reference design / performance µP
HD Map Representation
Vehicle dynamics integration platform
Motion Control
Ego-Motion
Engine
Brake
Steering
Common SW-Stack
OEM-specific SW
Navigation ECU
HD Map
Telematics ECU
Onboard
Offboard
Test fleet
Interface and toolchain
Development Backend incl. ML und Simulation
Image-/ Sensor data
Learning Map
Backend for Customer Cars incl. HD Map

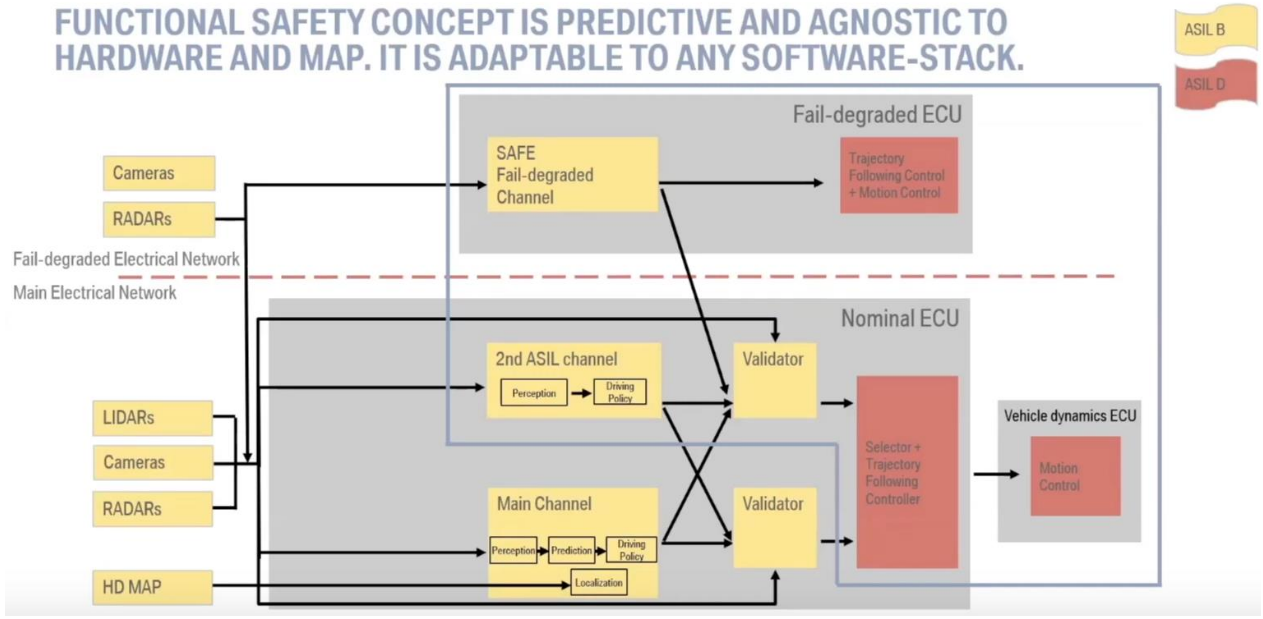
System and Software Architecture for Automated Driving Systems | 2-Apr-2020 | Simon Fürst

Page 3

Source: The Autonomous Safety & Architecture. Simon Furst. 2020

What is coming? BMW ADAS architecture

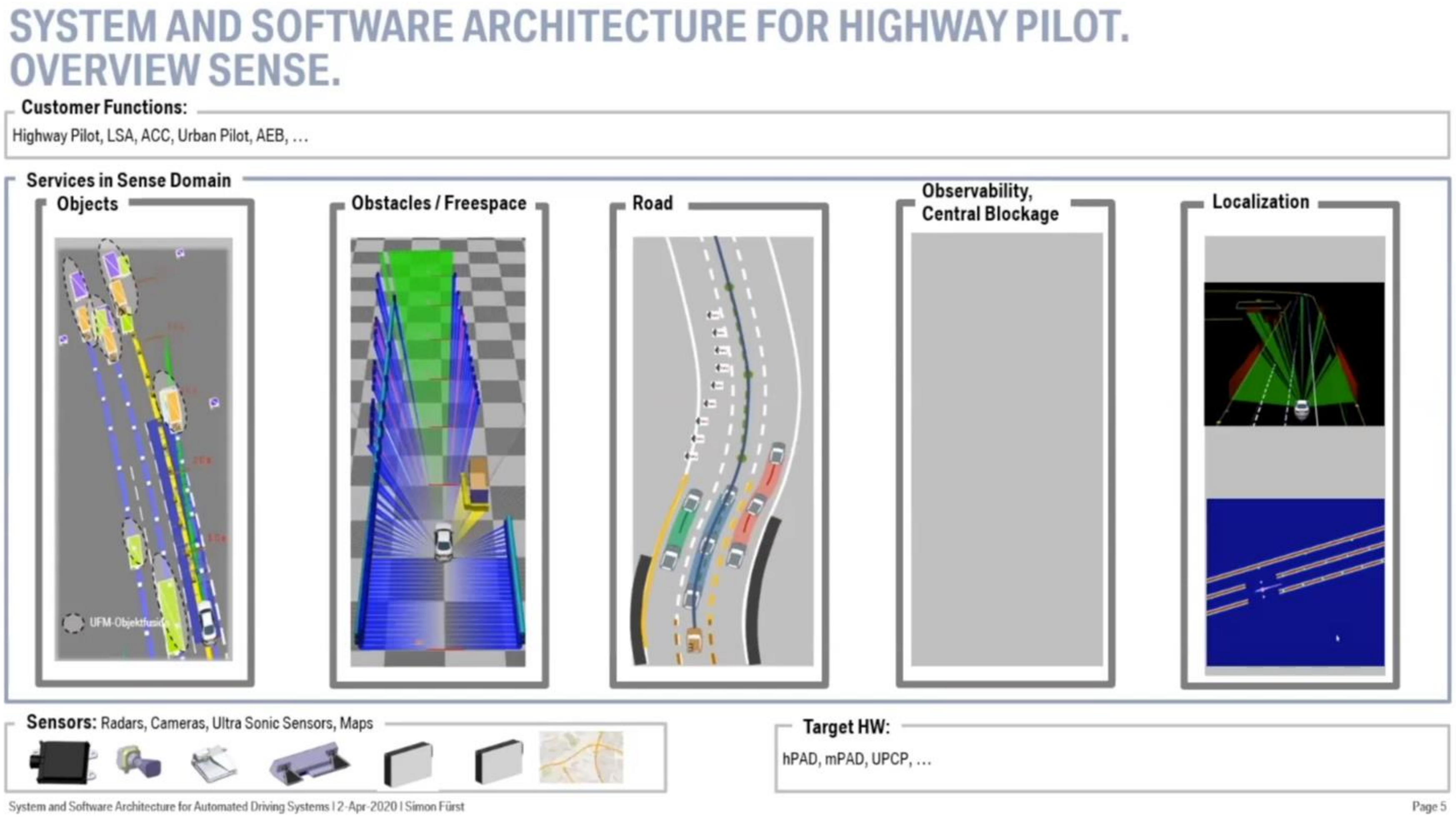
- Major parts of the system are ASIL B. Small components are required to be ASIL D



FUNCTIONAL SAFETY CONCEPT IS PREDICTIVE AND AGNOSTIC TO HARDWARE AND MAP. IT IS ADAPTABLE TO ANY SOFTWARE-STACK.

Source: The Autonomous Safety & Architecture. Simon Furst. 2020

What is coming? BMW ADAS architecture



SYSTEM AND SOFTWARE ARCHITECTURE FOR HIGHWAY PILOT. OVERVIEW SENSE.

Customer Functions:

Highway Pilot, LSA, ACC, Urban Pilot, AEB, …

Services in Sense Domain

- Objects
- Obstacles / Freespace
- Road
- Observability, Central Blockage
- Localization

Sensors: Radars, Cameras, Ultra Sonic Sensors, Maps

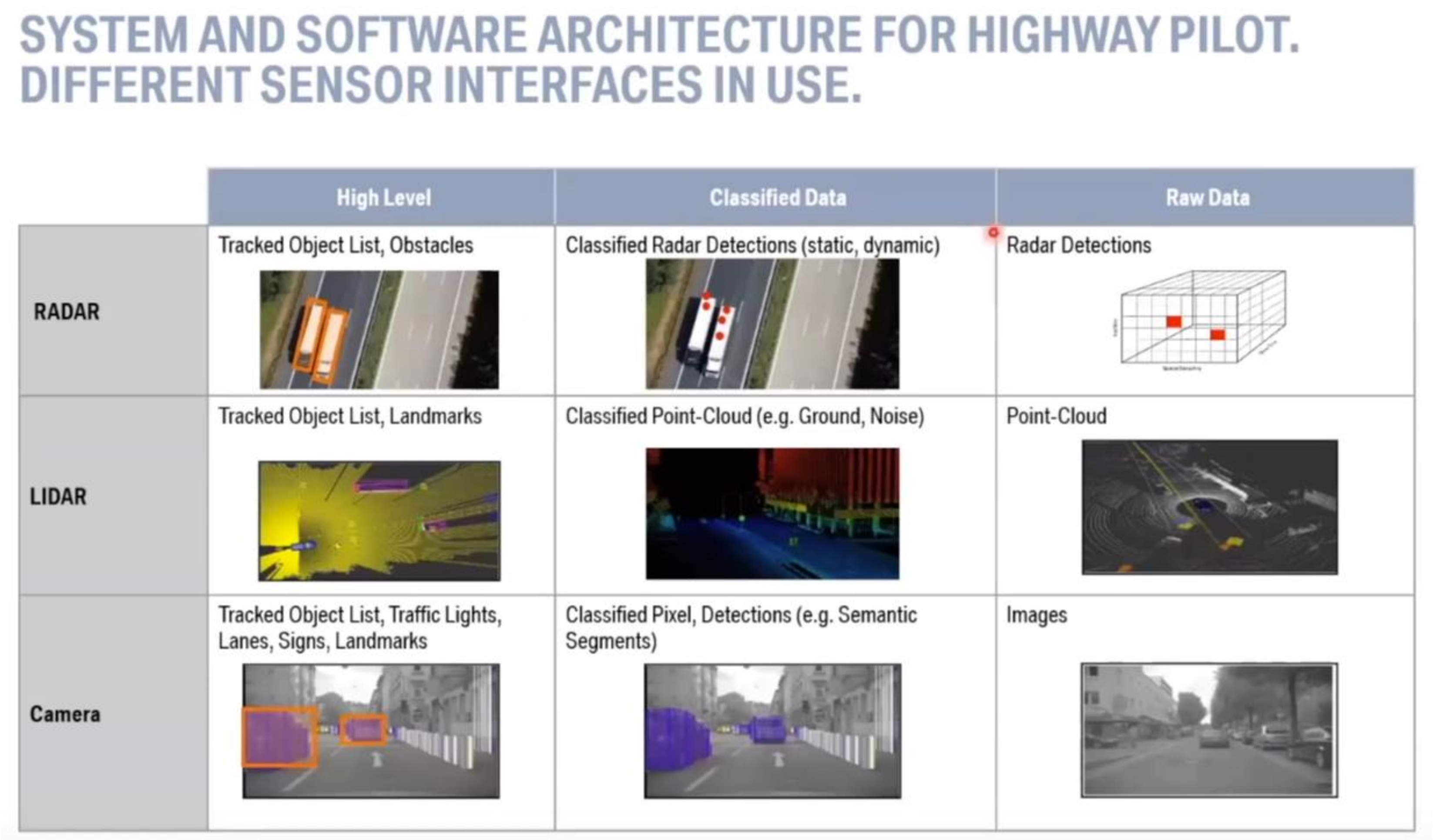
Target HW:

hPAD, mPAD, UPCP, …

System and Software Architecture for Automated Driving Systems | 2-Apr-2020 | Simon Fürst

Page 5

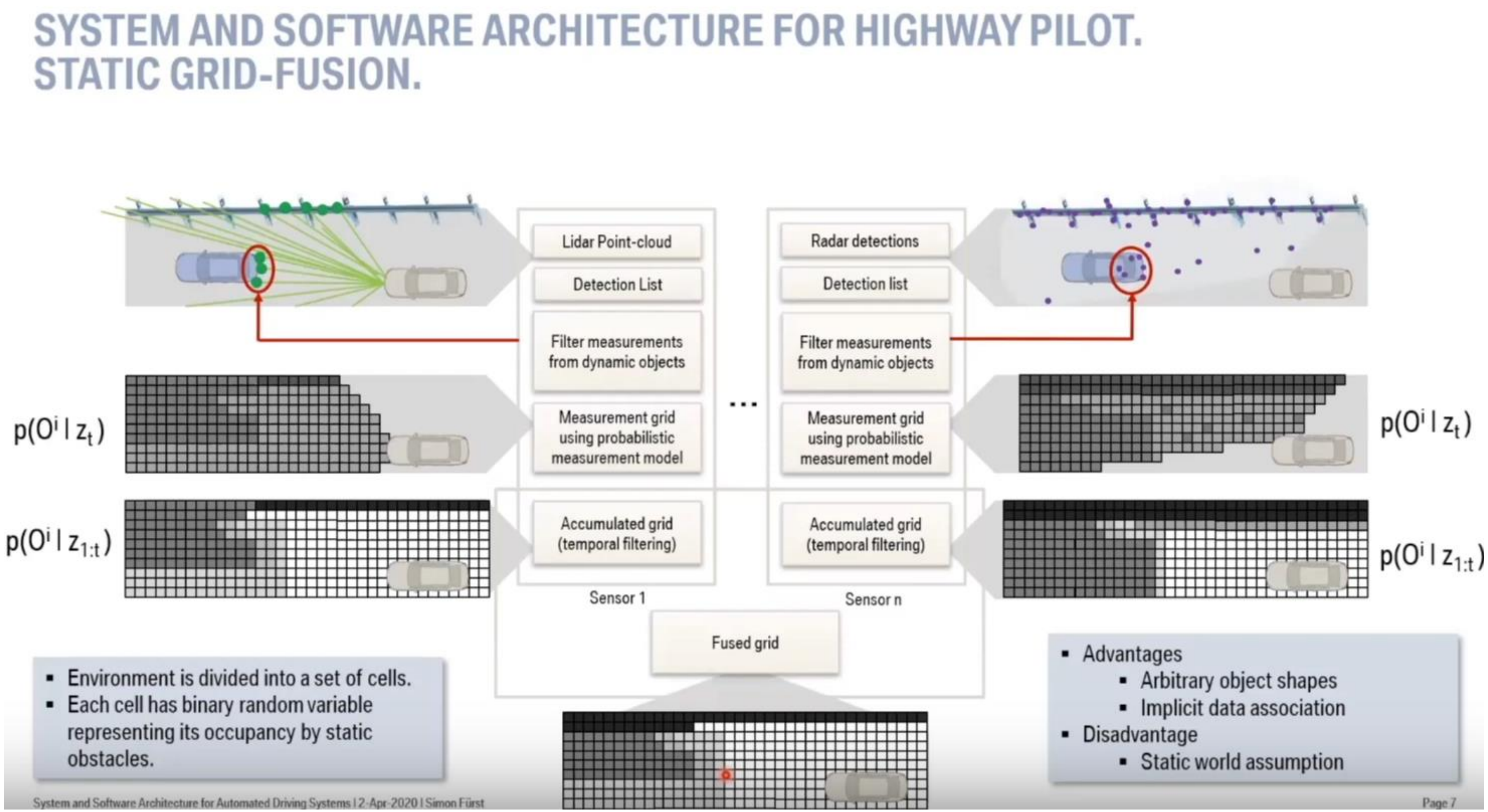
What is coming? BMW ADAS architecture



SYSTEM AND SOFTWARE ARCHITECTURE FOR HIGHWAY PILOT. DIFFERENT SENSOR INTERFACES IN USE.

	High Level	Classified Data	Raw Data
RADAR	Tracked Object List, Obstacles	Classified Radar Detections (static, dynamic)	Radar Detections
LIDAR	Tracked Object List, Landmarks	Classified Point-Cloud (e.g. Ground, Noise)	Point-Cloud
Camera	Tracked Object List, Traffic Lights, Lanes, Signs, Landmarks	Classified Pixel, Detections (e.g. Semantic Segments)	Images

What is coming? BMW ADAS architecture



SYSTEM AND SOFTWARE ARCHITECTURE FOR HIGHWAY PILOT. STATIC GRID-FUSION.

Lidar Point-cloud

Detection List

Filter measurements from dynamic objects

Measurement grid using probabilistic measurement model

Accumulated grid (temporal filtering)

Sensor 1

...

Radar detections

Detection list

Filter measurements from dynamic objects

Measurement grid using probabilistic measurement model

Accumulated grid (temporal filtering)

Sensor n

Fused grid

$p(O^i \mid z_t)$

$p(O^i \mid z_{1:t})$

$p(O^i \mid z_t)$

$p(O^i \mid z_{1:t})$

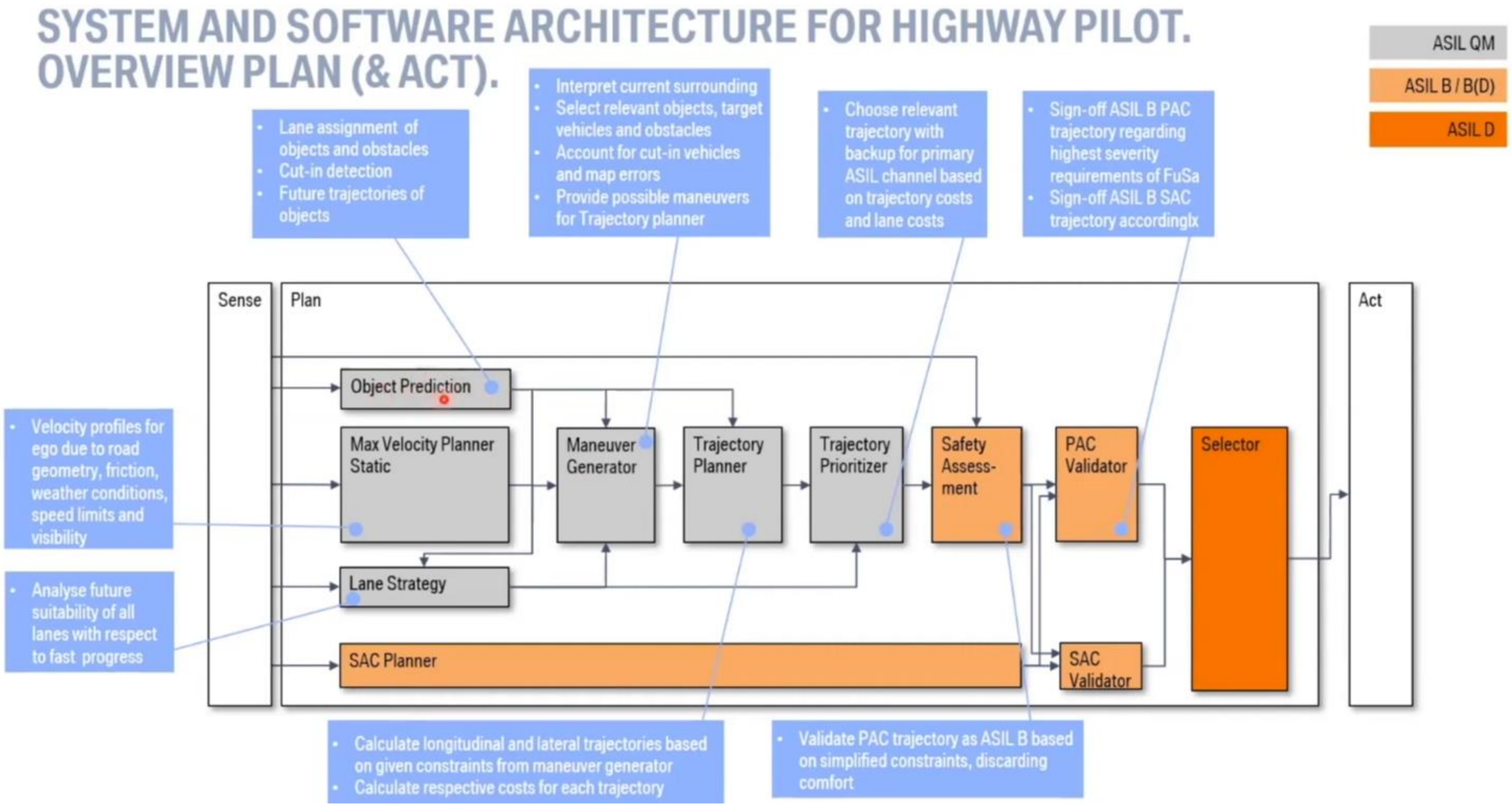
- Environment is divided into a set of cells.
- Each cell has binary random variable representing its occupancy by static obstacles.

- Advantages
  - Arbitrary object shapes
  - Implicit data association
- Disadvantage
  - Static world assumption

System and Software Architecture for Automated Driving Systems I 2-Apr-2020 I Simon Fürst

Page 7

What is coming? BMW ADAS architecture



SYSTEM AND SOFTWARE ARCHITECTURE FOR HIGHWAY PILOT. OVERVIEW PLAN (& ACT).

ASIL QM
ASIL B / B(D)
ASIL D

- Lane assignment of objects and obstacles
- Cut-in detection
- Future trajectories of objects

- Interpret current surrounding
- Select relevant objects, target vehicles and obstacles
- Account for cut-in vehicles and map errors
- Provide possible maneuvers for Trajectory planner

- Choose relevant trajectory with backup for primary ASIL channel based on trajectory costs and lane costs

- Sign-off ASIL B PAC trajectory regarding highest severity requirements of FuSa
- Sign-off ASIL B SAC trajectory accordinglx

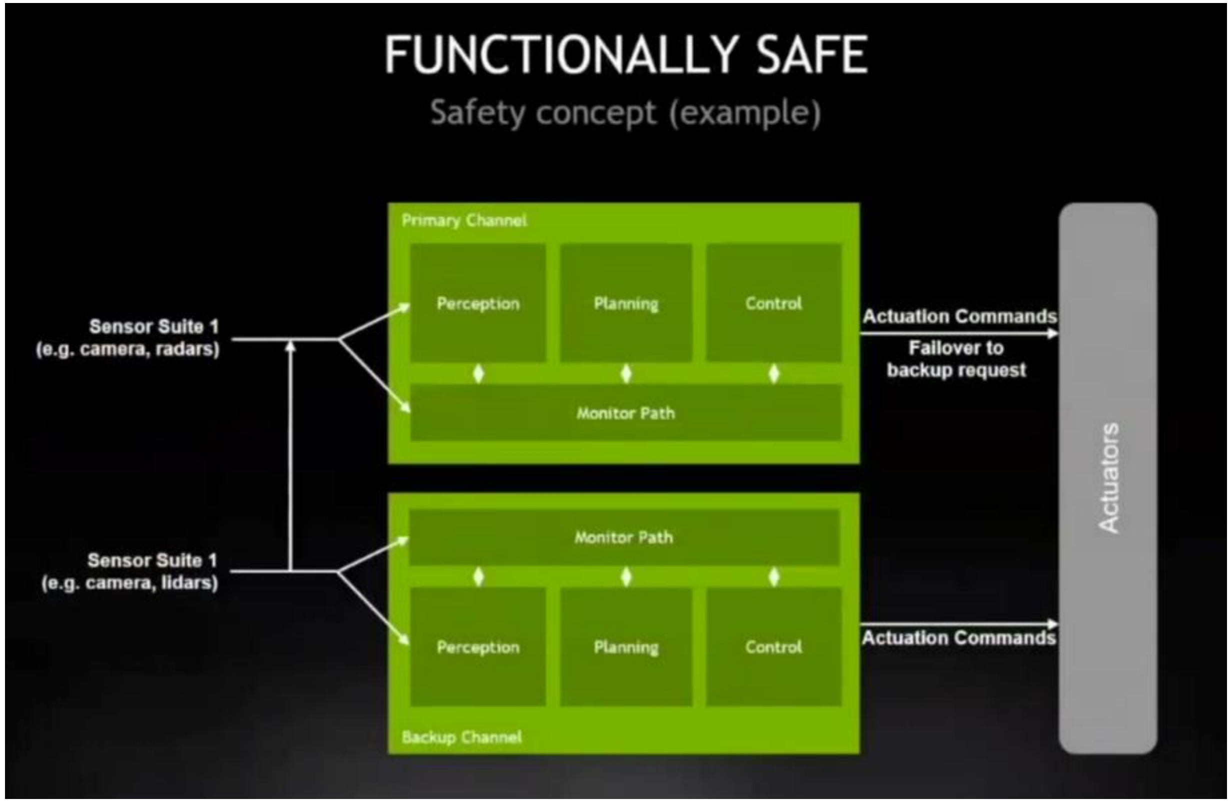
- Velocity profiles for ego due to road geometry, friction, weather conditions, speed limits and visibility

- Analyse future suitability of all lanes with respect to fast progress

- Calculate longitudinal and lateral trajectories based on given constraints from maneuver generator
- Calculate respective costs for each trajectory

- Validate PAC trajectory as ASIL B based on simplified constraints, discarding comfort

What is coming? NVIDIA ADAS architecture



Source: The Autonomous Safety & Architecture. Ricardo Mariani. 2020