

Lecture at Voronezh State Univ.

Introduction to Automotive Embedded Systems

8-June-2012

Hiroaki TAKADA

**Executive Director and Professor
Center for Embedded Computing Systems, Nagoya Univ.
Chairman, TOPPERS Project**

Email: hiro@ertl.jp URL: <http://www.ertl.jp/~hiro/>

Introduction of Nagoya and Nagoya Univ.

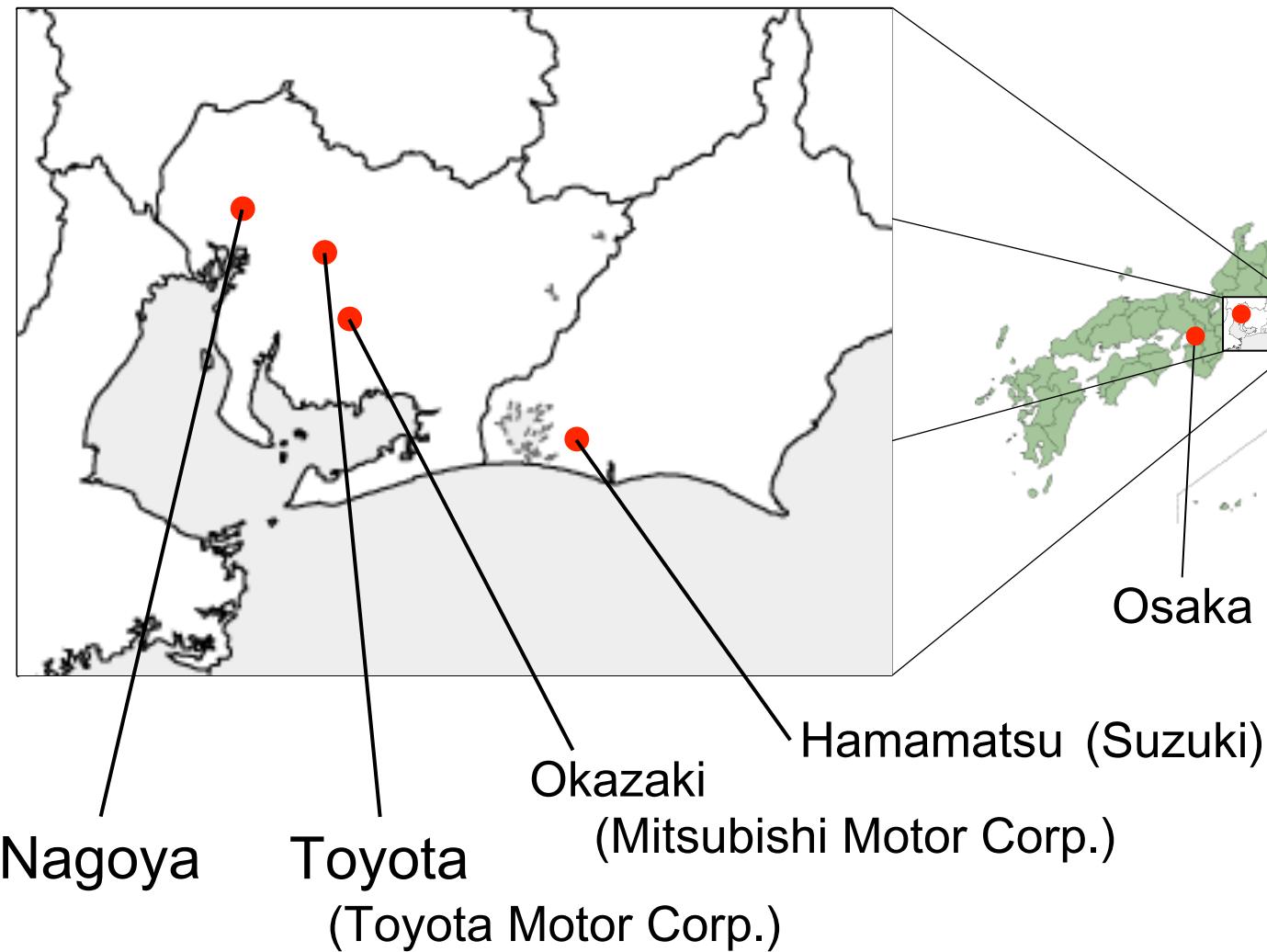
Nagoya

- ▶ Center city of third largest metropolitan area in Japan
 - ▶ Tokyo (incl. Yokohama), Osaka, Nagoya, ...
- ▶ Located around the center of Japanese Main Island (between Tokyo and Osaka)
- ▶ Manufacturing industry center of Japan
- ▶ Automotive industries are concentrated, especially
 - ▶ The headquarters of Toyota Motor Corp. (located in Toyota City) is near to Nagoya.

Nagoya University

- ▶ National University located in Nagoya City
- ▶ Within top 10 (I hope top 5!) universities of Japan
- ▶ 4 Nobel Prize Winners

Location of Nagoya



Self Introduction – Hiroaki Takada

Current Positions

- ▶ Professor, Nagoya University
- ▶ Executive Director, Center for Embedded Computing Systems (NCES), Nagoya University
- ▶ Chairman, TOPPERS Project
and several others

Major Research Topics

- ▶ Real-time operating systems for embedded systems
 - ▶ Real-time scheduling and analysis
 - ▶ Electronic system-level design
 - ▶ Automotive embedded systems
- !** *several joint projects with Toyota Motor Corp. and other Japanese automotive industries*

Table of Contents

Introduction to Automotive Embedded Systems

- ▶ Automotive Embedded Systems and their Features
- ▶ Classification of Automotive Embedded Systems
- ▶ Example Systems – Engine Management, ...
- ▶ Evolution Steps of Automotive Control Systems
- ▶ Current Problems of Automotive Embedded Systems
- ▶ Platform-base Development, AUTOSAR, JASPAR
- ▶ ISO 26262 – Functional Safety Standard

Brief Introduction to Our Activities – NCES and TOPPERS

- ▶ Introduction to NCES
- ▶ Introduction to TOPPERS Project

INTRODUCTION TO AUTOMOTIVE EMBEDDED SYSTEMS

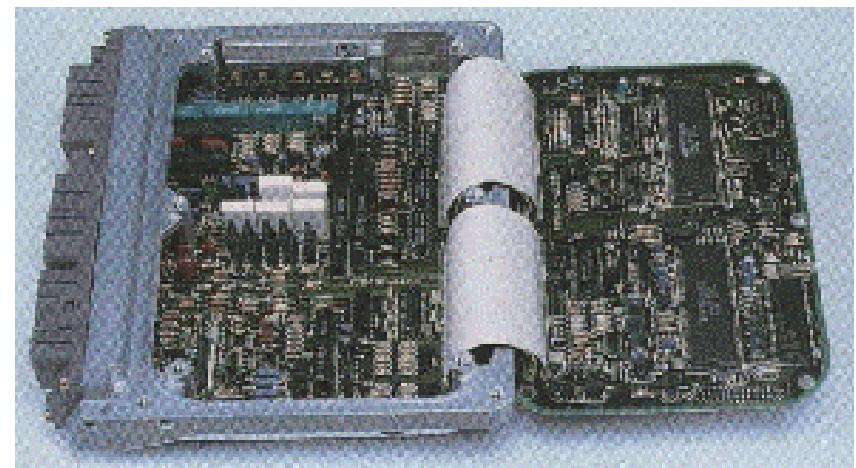
Automotive Embedded (Computing) Systems

Embedded (Computing) Systems

- ▶ A computer system that is embedded into a piece of equipment or a machine to control it.
- ▶ Embedded systems are applied to most electric/electronic equipment, recently.

Automotive Embedded (Computing) System

- ▶ A computer system that is embedded into a car to control it.
- ▶ An embedded computer unit is called an ECU (Electronic Control Unit).



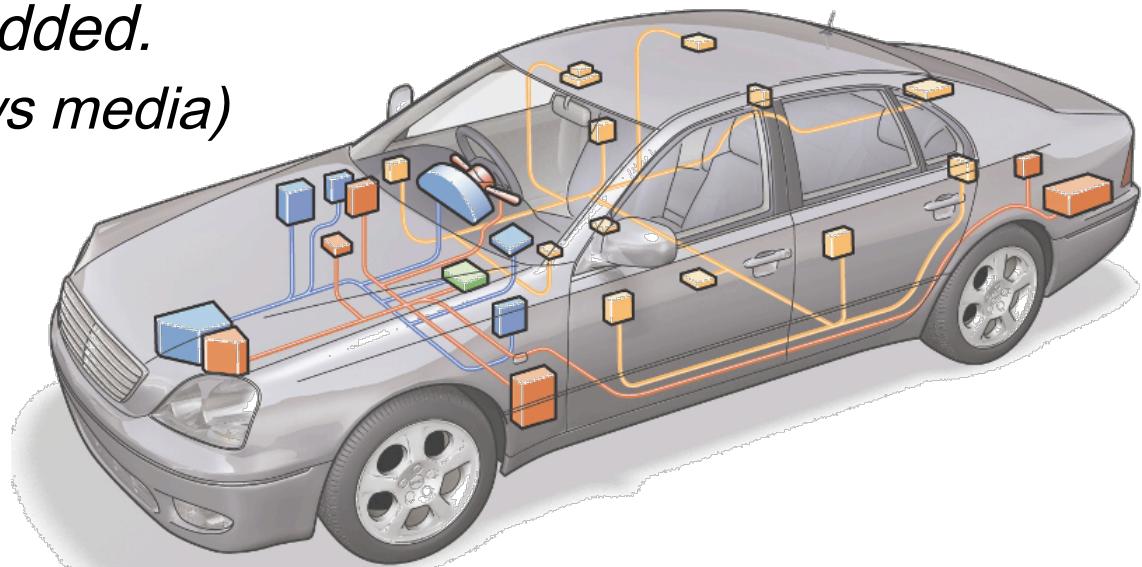
Engine Management ECU

Example: LEXUS LS-460

- ▶ released in Sep., 2006.
- ▶ more than *100 ECUs*
embedded when all optional equipments are installed.
- ▶ *about 7,000,000 lines of software embedded.*
(from different news media)



<http://www.lexus.jp/>



Automotive Embedded Systems and Networks

General Features of Automotive Embedded Systems

- ▶ Many (as many as 100) ECUs are used for the following purposes:
 - ▶ energy saving & low emission
 - ▶ safety (active & passive)
 - ▶ comfortableness, convenience, entertainment
 - ▶ cost & weight reduction
- ▶ ECUs are connected with several in-vehicle networks.
- ▶ *High reliability and safety requirements*
- ▶ *Strict real-time property required*
- ▶ Severe environmental conditions (temperature, EMC)
- ▶ *Severe production cost restriction*
- ! ECUs for different systems/services have different requirements and require different technologies.

Classification of Automotive Embedded Systems

Powertrain and Chassis Control

- ▶ engine, automatic transmission, hybrid control, ...
- ▶ steering, brake, suspension, ...

Body Electronics

- ▶ instrument panel, key, door, window, lighting, ...
- ▶ air bag, seat belt, ...

Multimedia (Infortainment) Applications

- ▶ car audio, car navigation, traffic information, ...
- ▶ electronic toll collection (ETC), backguide monitor, ...

Integrated Systems/Services

- ▶ electronic stability control, pre-crash safety, ...
- ▶ parking assistance, lane keeping assistance, ...

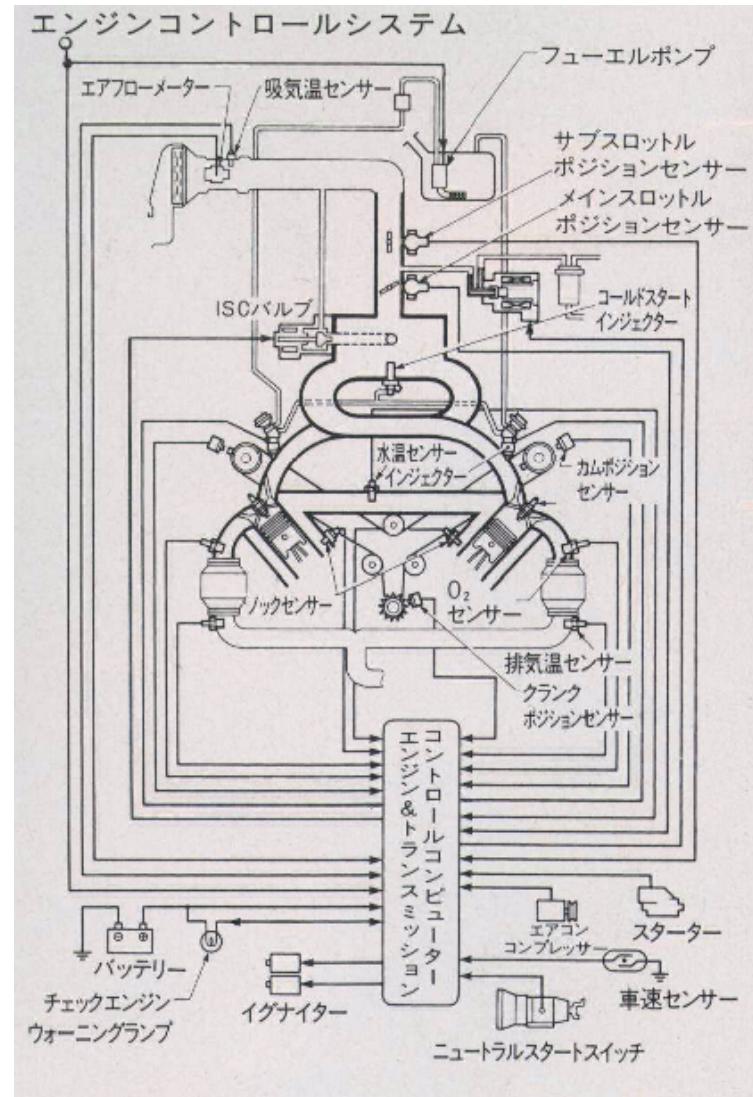
Example (1) – Engine Management System

System Components

- ▶ control computer (ECU)
- ▶ many sensors
 - ▶ crank position sensor
 - ▶ air flow meter
 - ▶ intake temperature sensor
 - ▶ throttle sensor
- ▶ some actuators

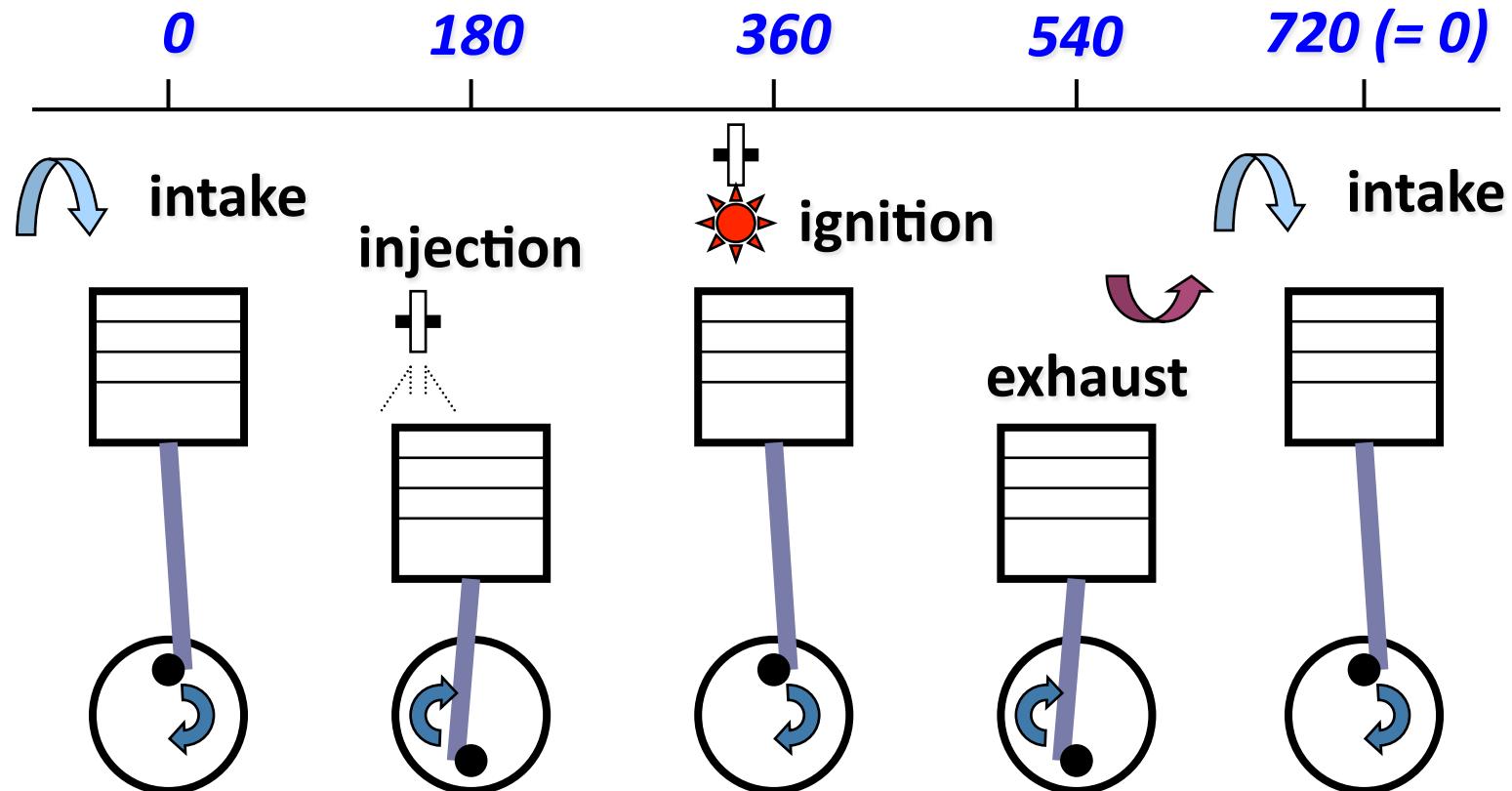
Basic Functions of the Control System

- ▶ to calculate fuel injection volume and ignition timing, and to control the actuators in every rotation cycle



Timing Behavior of Engine Management System

- ▶ When rotation speed is 6000rpm , one cycle is 20msec.
- ▶ Timing precision of the ignition is $10\mu\text{sec.}$ order.



Required Real-Time Property (Example)

- ▶ The calculation of the fuel injection volume *must be finished* before the injection timing.
- ▶ The calculation of the ignition timing *must be finished* before the ignition timing.
- ▶ Calculating too early has no additional value.

Safety Requirement (Example)

- ▶ Missing an ignition *must not happen*, because inflammable gas is emitted outside of the engine and can lead to a fire (because catalyst burns).
- ▶ If the ignition plug of a cylinder is broken, fuel *must not be injected* to the cylinder.



- ▶ *The engine management system monitors the ignition plug and stops the injection if the plug is broken.*

Example (2) – ABS

Function of ABS

ABS = Anti-lock Breaking System

- ▶ The speed of the car and the rotational speed of the wheel are monitored, and a skid is detected.
- ▶ When a skid is detected, hydraulic pressure to the brake is reduced to stop the skid.
- ▶ The system is relatively simple, but is becoming more complex, recently.

Safety Requirement (Example) and Fail-Safe Design

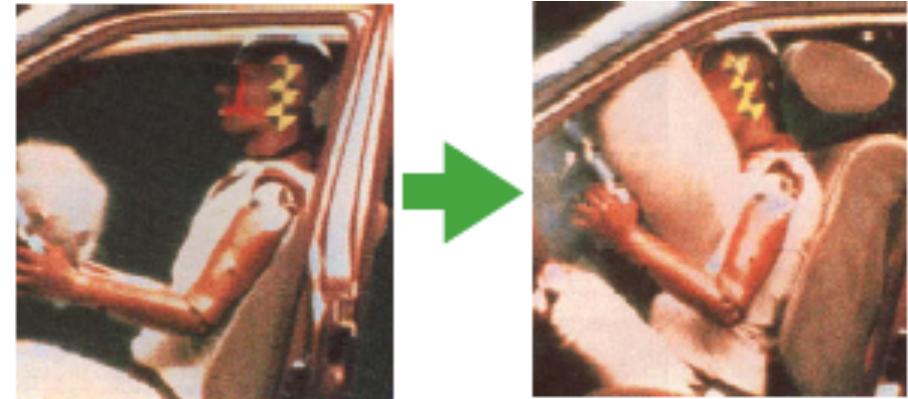
- ▶ Continuous reduction of hydraulic pressure causes non-braking.
- ▶ If some fault is detected, ABS stops functioning. Then, the brake works though a skid cannot be avoided.

→ *fail-safe design*

Example (3) – Airbag Control

Function of Airbag Control

- ▶ Airbag control system monitors various sensors including accelerometers and detects a collision.
- ▶ If a collision is detected, the ignition of a gas generator propellant is triggered to inflate a bag.



Real-Time Constraint

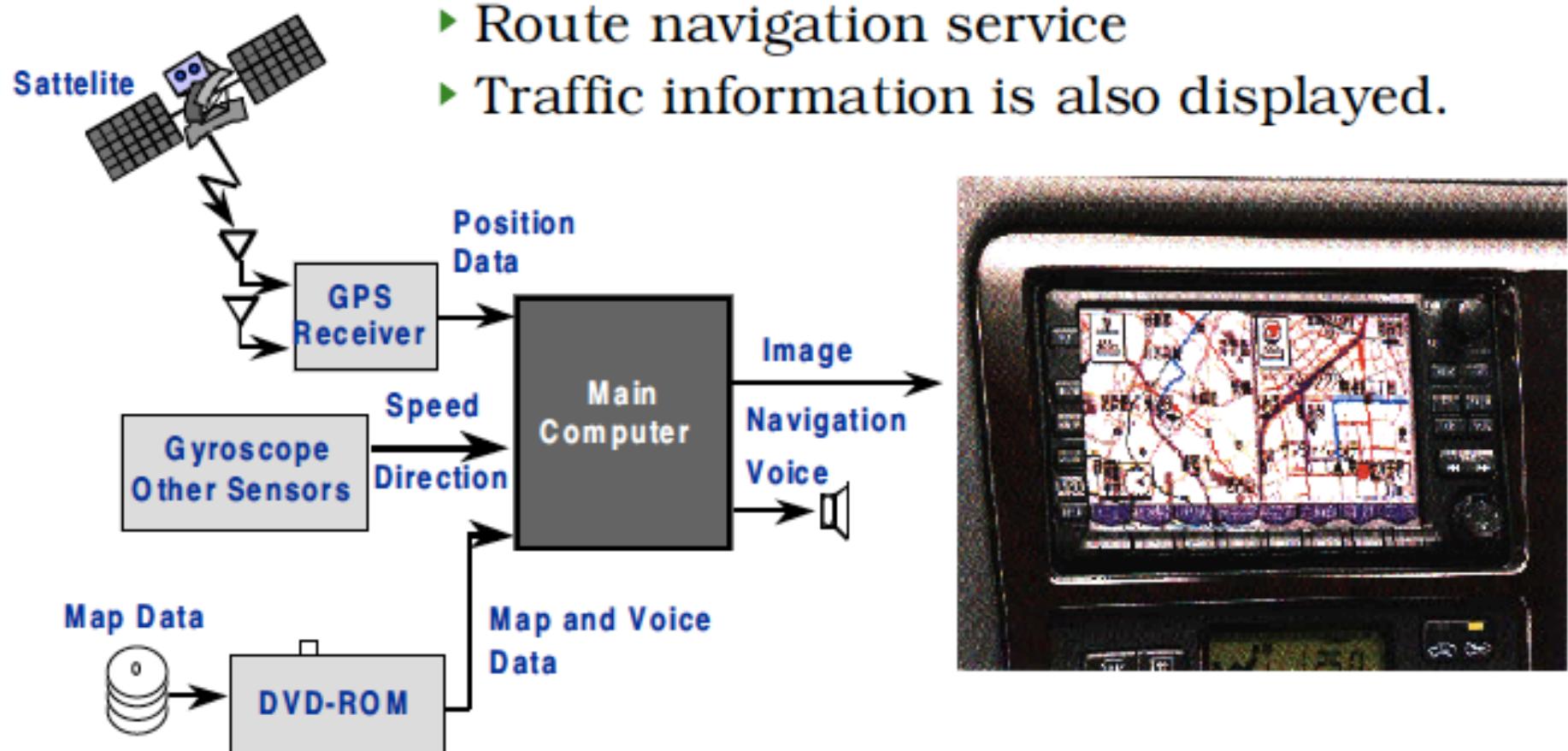
- ▶ The trigger must be within 10-20 msec. after the collision.

Safety Requirements

- ▶ Fail-safe design cannot be applied.
! even harder than ABS

Example (4) – Car Navigation System

- ▶ The current position of the car obtained from GPS, gyroscope, and others is displayed with the map.



Specific Requirements on Example Systems

Engine Management

- ▶ very short response time (10 or $100\mu\text{sec}$. order)
- ▶ large software and high computing power required
- ▶ high reliability

Air Bag

- ▶ a kind of signal processing application
- ▶ short response time (10msec . order)
- ▶ very high reliability

Car Navigation System

- ▶ largest and most complicated software in a car
- ▶ large computing power required
- ▶ moderate reliability, real-time property still required

Requirements on In-Vehicle Networks

Chassis Network → *high-speed CAN, FlexRay*

- ▶ short and guaranteed response time
- ▶ small data size
- ▶ high reliability

Body Electronics Network → *low-speed CAN, LIN*

- ▶ a large number of network nodes and data
- ▶ moderate reliability, low power consumption

Multimedia Network → *MOST, IDB-1394*

- ▶ high bandwidth for multimedia data
- ▶ moderate reliability

(True) By-Wire Network ... in future → *FlexRay?, TTP?*

- ▶ **very high reliability**

Evolution Steps of Automotive Control Systems

! Evolution of automotive control systems and networks is well understood with the following 4 stages.

Stage 1

- ▶ Computer control (ECU) is applied to various component (engine, brake, steering, and so on), independently.
- ▶ In-vehicle network is not used.

Stage 2

- ▶ Each control system (ECU) exchanges useful data for improving the quality of the control system.
- ▶ Each system operates almost independently, and timing constraints on networks are loose.

Stage 3 (Current) *integrated systems/services*

- ▶ Each system still operates autonomously, and some services are provided with multiple ECUs connected with in-vehicle networks.
- ▶ Mechanical backup system still exists, thus the basic functions of a car are preserved even if an electronic system fails.

Stage 4.A (Future)

- ▶ Networks with outside of the car (communication with another car and the road) are intensively used.

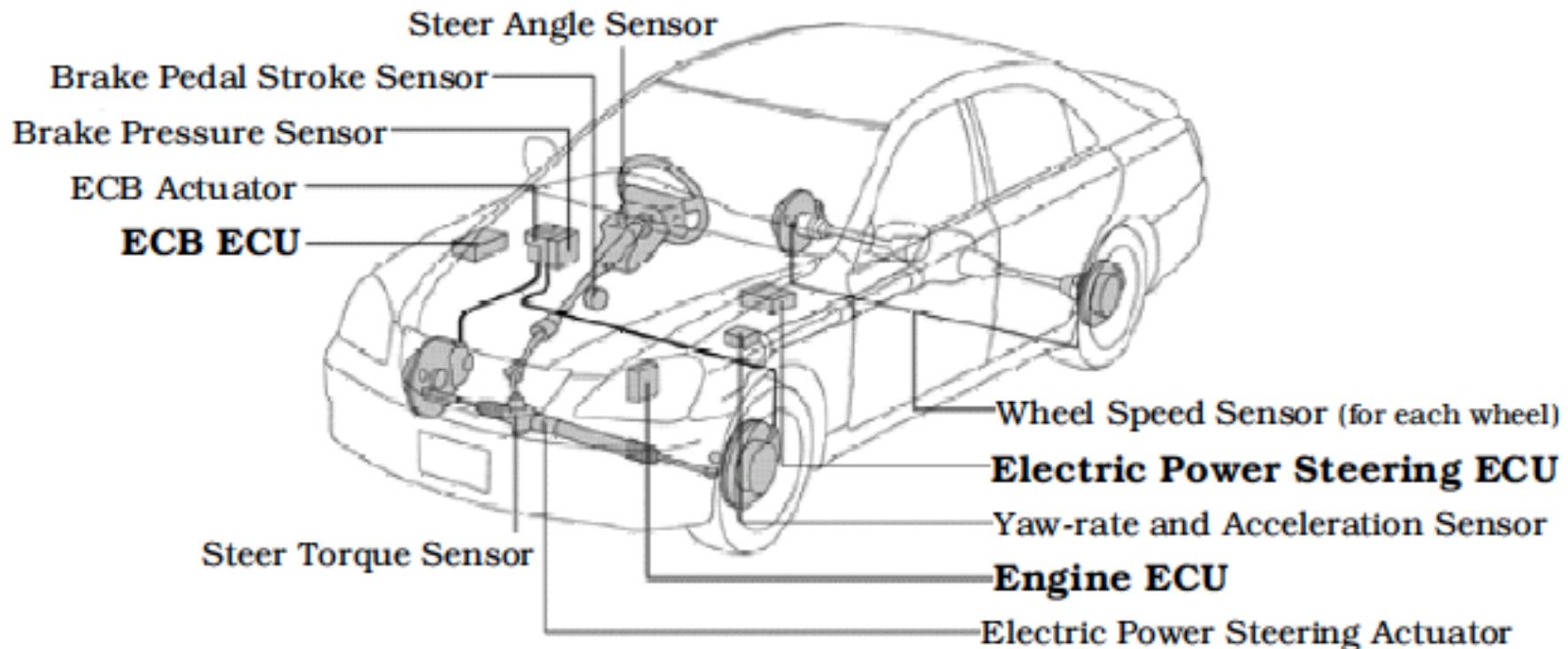
Stage 4.B (Future) *(true) by-wire systems*

- ▶ Mechanical systems (incl. backups) are replaced with ECUs and networks.
- ▶ A failure of electronic systems is life-critical.

Integrated System/Service Examples

Vehicle Dynamics Integrated Management (VDIM)

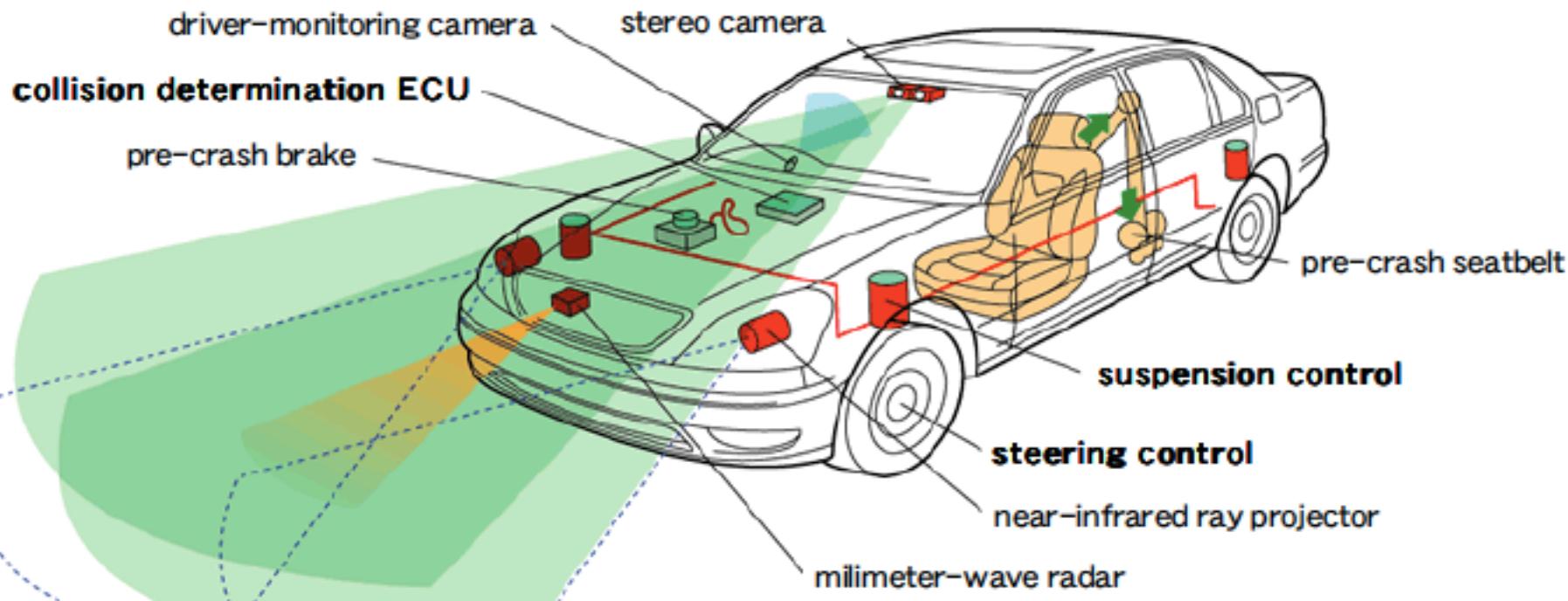
- ▶ Control brake, steering, and engine for avoiding slip and spin.



*ECB= Electronically Controlled Brake system

Pre-crash Safety System (PCS)

- ▶ When an obstacle is detected with stereo camera and millimeter-wave radar, the system retracts the seatbelts, warns the driver, and applies the brake.
- ▶ Driver's condition (eg. face direction) is monitored.



Current Problems of Automotive Embedded Systems

Complicated system design

- ▶ Increasing development cost and time
- ▶ How to achieve high reliability and safety?

Large-scale and complicated software

- ▶ How to achieve high reliability and safety?
- ▶ How to effectively reuse existing software?

Too large number of ECUs

- ▶ Increasing cost
- ▶ Insufficient space (in a car) for ECUs

Complicated network architecture

- ▶ Increasing design complexity

Platform-base Development

Conventional Component-base Development

- ▶ Each ECU (component) is developed (usually independently) at first.
 - ▶ An automotive component supplier develops both of the hardware and software of ECU.
- ▶ Car (system) is designed by integrating the ECUs developed by different suppliers.

Platform-base Development

- ▶ Platform (PF) should be developed at first.
 - ▶ PF = Hardware PF + Software PF + Network
 - ▶ Software PF = OS + middleware
- ▶ Application software should be developed on the PF.

AUTOSAR (Automotive Open System Architecture)

- ▶ A global partnership of carmakers, car component, electronics, semiconductor, and software industries founded in 2003.
 - ▶ defines a methodology that supports a distributed, function-driven development process.
 - ▶ standardizes the software-architecture for ECU.

Core partners:

- | | | |
|---------------|-----------|-----------------------|
| ▶ BMW | ▶ Daimler | ▶ PSA Peugeot Citroen |
| ▶ Bosch | ▶ Ford | ▶ Toyota Motor |
| ▶ Continental | ▶ GM | ▶ Volkswagen |

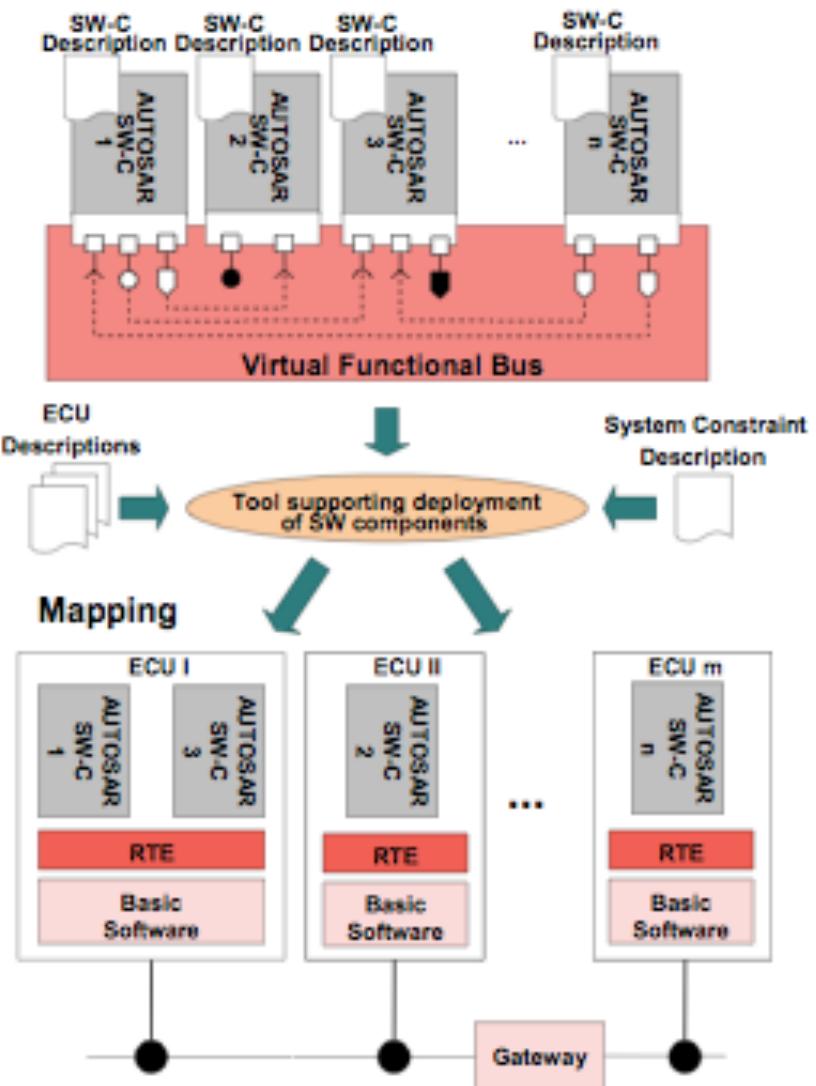
Results and Current Status

- ▶ Recent version (Release 4.0-REV 3) consists of about 100 specifications and 80 related documents.
- ▶ Phase III activity was started in 2010.

Overview of AUTOSAR Framework

AUTOSAR Approach

- ▶ Describe a system as a set of software components (SW-C) connected with virtual function bus, logically.
- ▶ Map the system to ECUs connected with in-vehicle network by a tool.
- ▶ ECU and system constraint descriptions are inputs to the tool.
- ▶ The software platform consists of RTE and BSW.



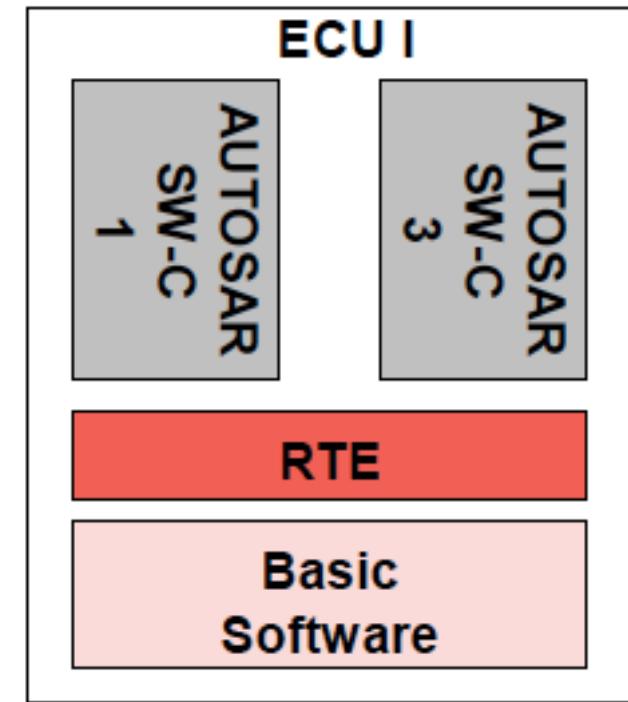
Structure of AUTOSAR Software Platform

Run-Time Environment (RTE)

- ▶ Provides interfaces between SW-Cs and between SW-C and BSW.
- ▶ Provides the BSW services to SW-C (API abstraction)
- ▶ Source code of RTE is generated by a tool from the description of communication interfaces
 - ! the most distinctive feature of AUTOSAR platform*

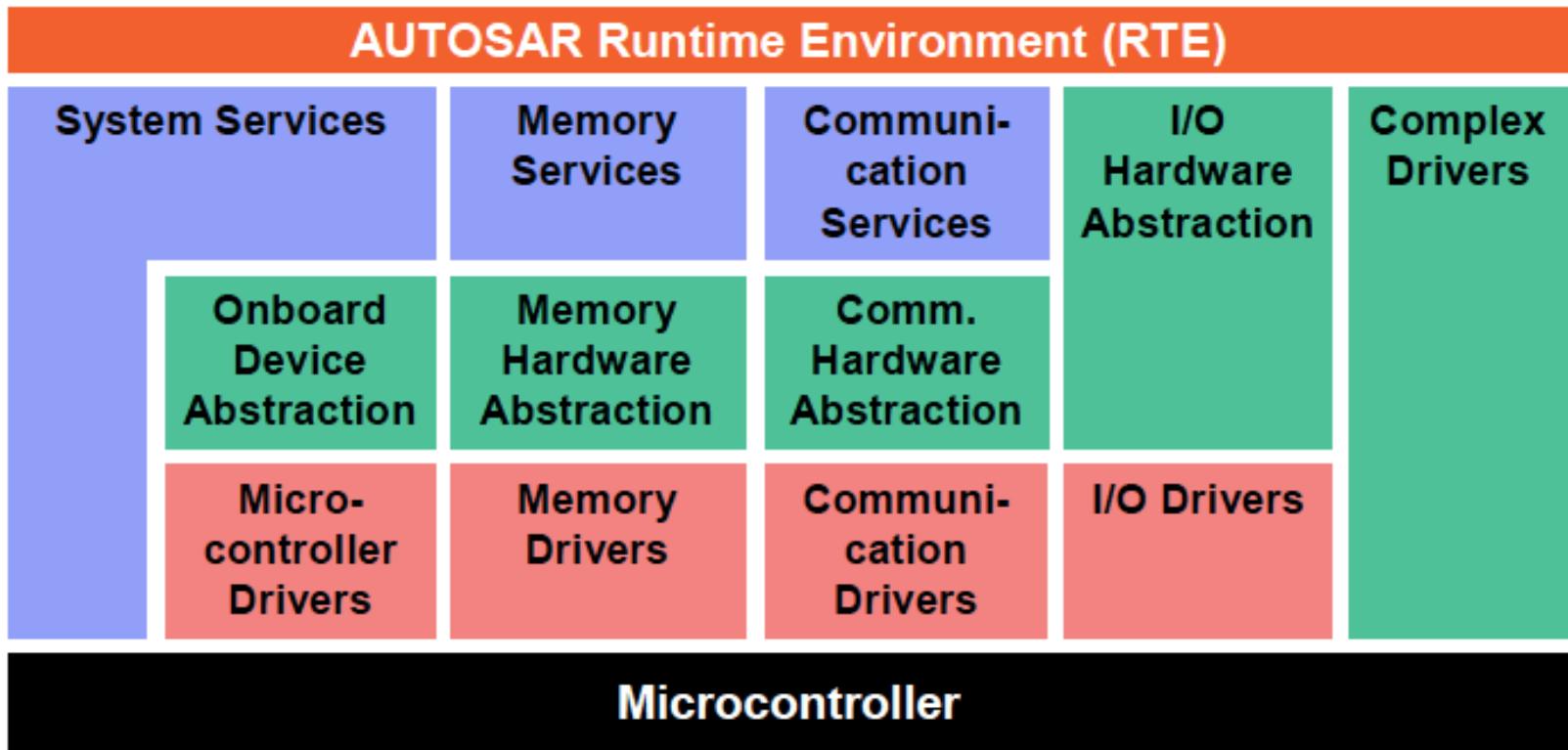
Basic Software (BSW)

- ▶ operating system (OS)
- ▶ device drivers
- ▶ middleware



Structure of Basic Software (BSW)

- ▶ 4 function groups (system, memory, communication, I/O) are organized in 3 layers (service, ECU abstraction, microcontroller abstraction)
- ▶ complex drivers (to shortcut the layers)



JasPar (Japan Automotive Software Platform Architecture)

- ▶ Car makers and other companies jointly develop network technology, middleware, software platform for automotive control systems (founded in 2004).

Board Members

- ▶ Toyota Motor
- ▶ Honda
- ▶ DENSO
- ▶ Nissan
- ▶ Toyota Tsusho Electronics

Members

- ▶ about 100 carmakers, car component suppliers, semiconductor companies, and software companies

Major Activities and Results

- ▶ Standardization related to FlexRay (wiring rule, ...)
- ▶ Development of software platform based on AUTOSAR Standard.
- ▶ Development of design guidelines for ISO 26262.

ISO 26262 – Functional Safety Standard

What is Safety?

- ▶ Safety : “freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property, or damage to the environment”
- ▶ Reliability : “the ability of a system or component to perform its required functions under stated conditions for a specified period of time”
- ▶ A safe system is not necessarily reliable and vice versa.

What is Functional Safety?

- ▶ Original meaning: safety achieved by functions
- ▶ “absence of unreasonable risk due to hazards caused by malfunctioning behavior of E/E systems” (ISO 26262-1)

What is ISO 26262?

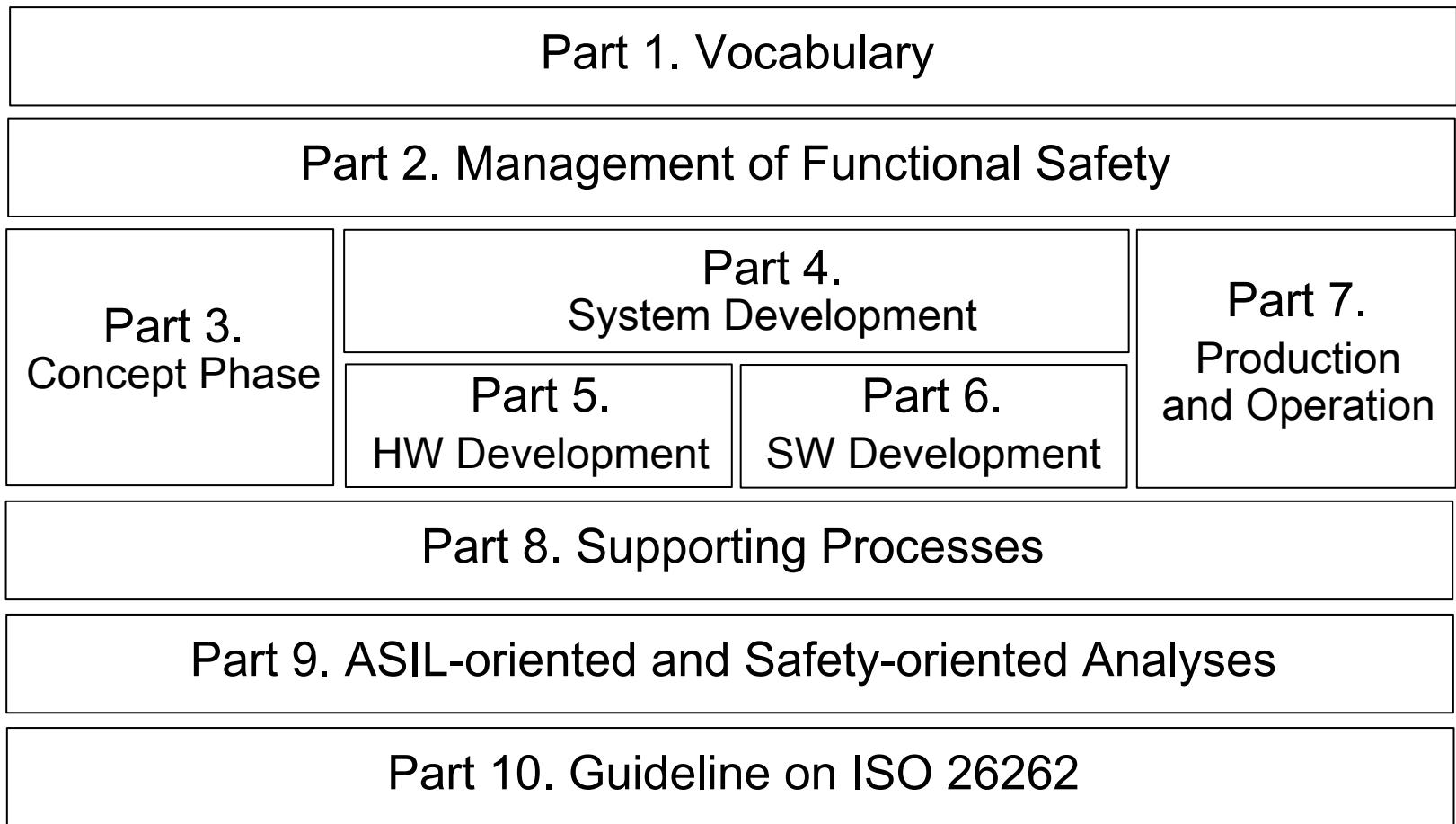
- ▶ “Road vehicles – Functional safety –”
- ▶ published in Nov. 2011.

Characteristics of ISO 26262

- ▶ A compilation of best practice for developing a safety-related system.
 - ▶ In the standard, many development techniques of safe systems are listed and required depending on the safety level (ASIL).
- ▶ Reliability of hardware is calculated from the failure rate of each component consisting the system.
- ▶ Reliability of software is achieved with software development process.
 - ▶ *A software that is developed with a process satisfying the requirements of the standard is considered to be enough reliable.*

Overall Structure of ISO 26262

- ▶ ISO 26262 consists of 10 parts (Part 10 has not been published yet).



ASIL (Automotive Safety Integrity Level)

- ▶ requirement level of safety measures to be applied for avoiding an unreasonable residual risk
- ▶ four (+ one) levels
 - ▶ ASIL D ... most stringent level
 - ▶ ASIL C
 - ▶ ASIL B
 - ▶ ASIL A ... least stringent level
 - ▶ QM (quality management) ... non-safety-related
- ▶ ASIL is determined for each hazardous event based on its severity, exposure, and controllability.

Future Trends of Automotive Embedded Systems

Advancement of Integrated Systems/Services

- ▶ communication with another car and the road
- ▶ using various information (surrounding situation and map information) for controlling a car
- ▶ emergence of true by-wire systems ??

Changes in System Architecture

- ▶ ECU integration (or reduction)
- ▶ platform-base development
- ▶ separation into two types of ECUs

Changes in Software Development

- ▶ model-base design
- ▶ component-base software development
- ▶ virtual platform for software development

BRIEF INTRODUCTION TO OUR ACTIVITIES – NCES AND TOPPERS –

Overview of Our Activities

ERTL (Embedded and Real-Time Systems Laboratory)

- ▶ Takada Laboratory
- ▶ several joint projects with car makers, semi-conductor makers, and software companies

NCES (Center for Embedded Computing Systems)

- ▶ several (relatively) large-scale joint projects with car makers and car component suppliers
- ▶ projects for educating embedded system engineers

TOPPERS ... independent non-profit organization

- ▶ development of open-source real-time operating system (RTOS) and middleware for embedded systems
- ▶ cooperation of academia, industry, public research institutes, and individual engineers

Introduction to NCES

NCES (Center for Embedded Computing Systems)

- ▶ a research center focused on embedded computing systems
 - ▶ automotive application is the main focus
 - ▶ several (relatively) large-scale joint projects with car makers and car component suppliers
- ▶ established in 2006 to form a hub for collaborative research and education of embedded systems technologies

Scope of NCES

- ▶ research aiming at practical use in industry
- ▶ development of prototype system/software
- ▶ education and human resource development

Major Research Projects of NCES

Projects funded by Industries

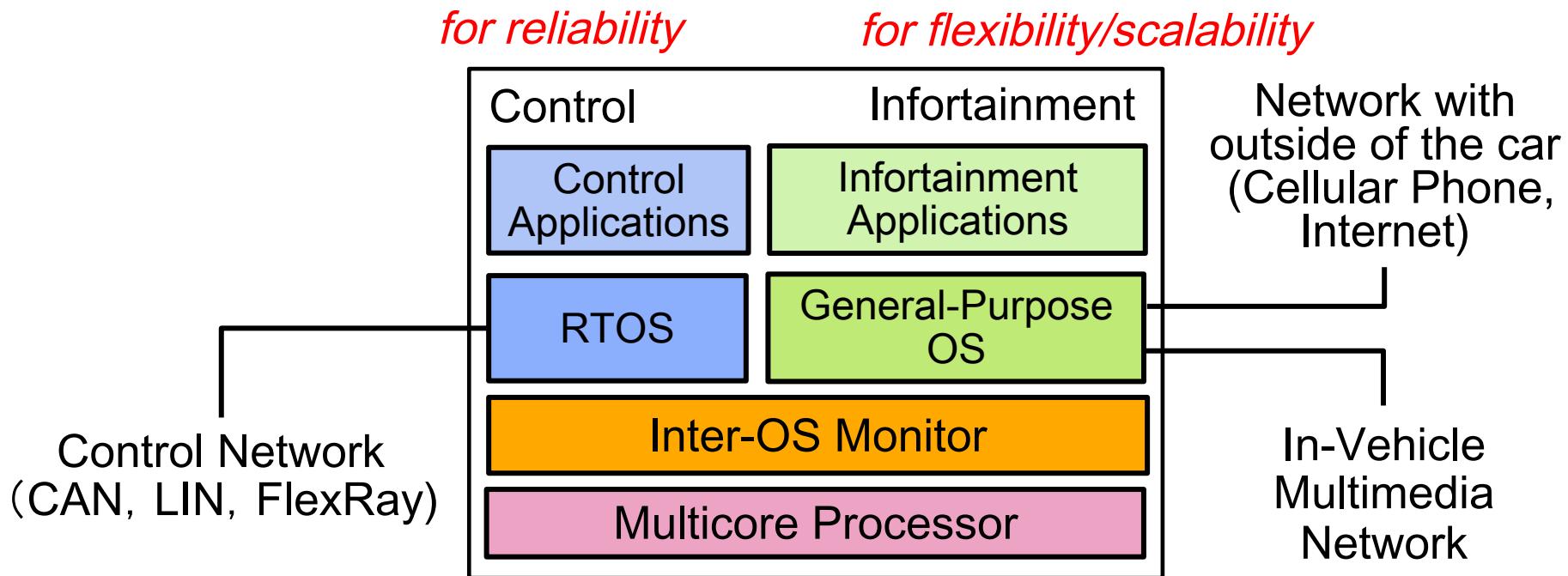
- ▶ OS for in-vehicle multimedia systems (TMC)
- ▶ Next-generation automotive network (AutoNetworks Technologies, Ltd.)
- ▶ Next-generation real-time operating system for automotive control systems (consortium type)
- ▶ Automotive data integration platform (consortium type)
- ▶ Studies on functional safety standard (JarPar)
- ▶ Software PF for Space Application (JAXA)

Projects funded by Government

- ▶ Energy Consumption optimization of embedded systems (JST CREST)
- ▶ Automotive software platform conforming to the functional safety standard (METI)

OS for In-vehicle Multimedia Systems

- ▶ joint project with Toyota Motor Corp. started in 2006
- ▶ a hybrid OS for in-vehicle multimedia systems to achieve both reliability and flexibility/scalability
- ▶ SafeG, the most important result, is distributed as an open-source software from TOPPERS Project.



Next-Generation Automotive Network

- ▶ joint project with AutoNetworks Technologies, Ltd. (a subsidiary of Sumitomo Electric Industries) started in April, 2006

Concept of Scalable CAN

- ▶ Scalable CAN is compatible with conventional CAN from software point of view and achieves higher bandwidth.
- ▶ Scalable CAN and conventional CAN can be connected with a gateway.

10Mbit/s CAN

- ▶ star topology, in which each node is connected to a central hub (gateway).

5Mbit/s CAN

- ▶ bus topology with limited number of nodes on a bus.

Next-Generation RTOS for Automotive Systems

- ▶ consortium-type project with NCES and 12 companies started in Apr. 2011
- ▶ 12 engineers from these companies are staying at NCES and engaged in the development.

What is to be developed

- ▶ specification of next-generation RTOS for automotive control systems based on AUTOSAR OS
- ▶ RTOS implementation based on the specification
- ▶ test suite for the RTOS

Release plan of the developed software

- ▶ Developed RTOS will be released as an open source software from TOPPERS Project.
- ▶ Test suite is shared only by the consortium members.

Introduction to TOPPERS Project

TOPPERS = Toyohashi Open Platform for
Embedded and Real-Time Systems



Objectives of the Project

- ▶ To develop various open-source software for embedded systems including RTOS and to promote their use.

Building a widely used open-source OS as Linux in the area of embedded systems!

Main Activities of the Project

- ▶ Building a definitive µITRON-conformant RTOS
- ▶ Developing a next generation RTOS technology
- ▶ Developing software development technology and tools for embedded systems
- ▶ Fostering Embedded System Engineers

Major Products (Software) of TOPPERS

! All SW listed below can be downloaded from the TOPPERS website at <http://www.toppers.jp/>.

TOPPERS/JSP Kernel (JSP = Just Standard Profile)

- ▶ RTOS conformant to the standard profile of μITRON4.0 specification

TOPPERS/ATK1 (ATK = Automotive Kernel)

- ▶ RTOS conformant to OSEK/VDX OS specification

TOPPER/ASP Kernel (ASP = Advanced Standard Profile)

- ▶ Improvement of JSP kernel
- ▶ Basis of TOPPERS new generation kernels

TOPPERS/FMP Kernel (FMP = Flexible Multiprocessing)

- ▶ Extension of ASP kernel to various types of multiprocessor systems

TECS (TOPPERS Embedded Component System)

- ▶ Specification and tools for component-based development of embedded software.

TINET

- ▶ Compact TCP/IP protocol stack conformant to ITRON TCP/IP API specification.
- ▶ Both IPv4 and IPv6 are supported.

TLV (TraceLogVisualizer)

- ▶ Customizable tool to visualize various trace logs, including the trace log of RTOS

Several Open Educational Materials

- ▶ Educational materials including presentation slides, software, and so on.

Application Example of TOPPERS OS

Consumer Applications



PM-A970 (EPSON)



DO!KARAOKE
(PANASONIC)



UA-101 (Roland)



IPSiO GX e3300 (Ricoh)



GT-541 (Brother)

Industrial and Other Applications



Kizashi (SUZUKI)



H-II B (JAXA)
under development

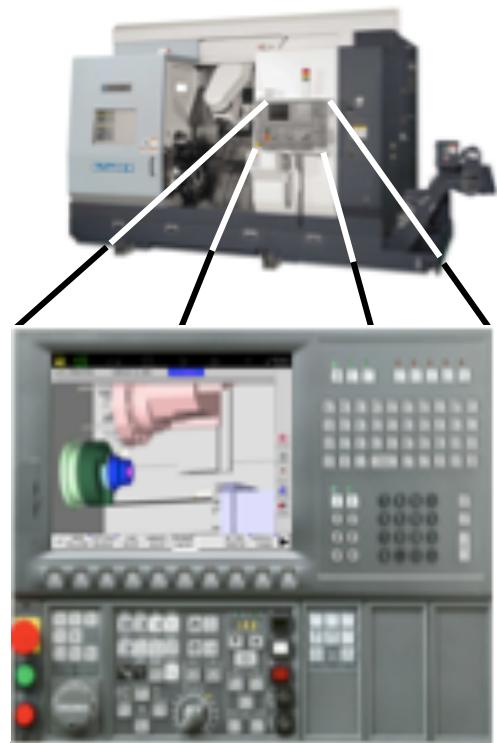


ASTRO-H (JAXA)
under development

DP-350
(Daihen)



AP-X (Kyowa
MEDIX)



OSP-P200 (Okuma)