AUTOSAR Layered Software Architecture

KPIT

Agenda

1 Introduction

2 Generic Layered Architecture

3 Overview of Software Layers

4 Overall Basic Software Architecture

Introduction

- Provides a conceptual layered software architecture
- Provides mapping of the identified BSW modules to software layers

- Automotive ECUs having the following properties can adapt to AUTOSAR
  - Strong interaction with hardware (sensors and actuators)
  - Connection to vehicle network via CAN, LIN or FlexRay
  - Microcontrollers from 16 to 32 bit
  - Real Time Operating System
  - Program execution from internal or external flash memory

- Extensibility
  - Modules can be added or the functionality of existing ones can be extended, but their configuration has to be considered in the Basic SW configuration process
  - Complex drivers can easily be added

Agenda

1 Introduction

2 Generic Layered Architecture

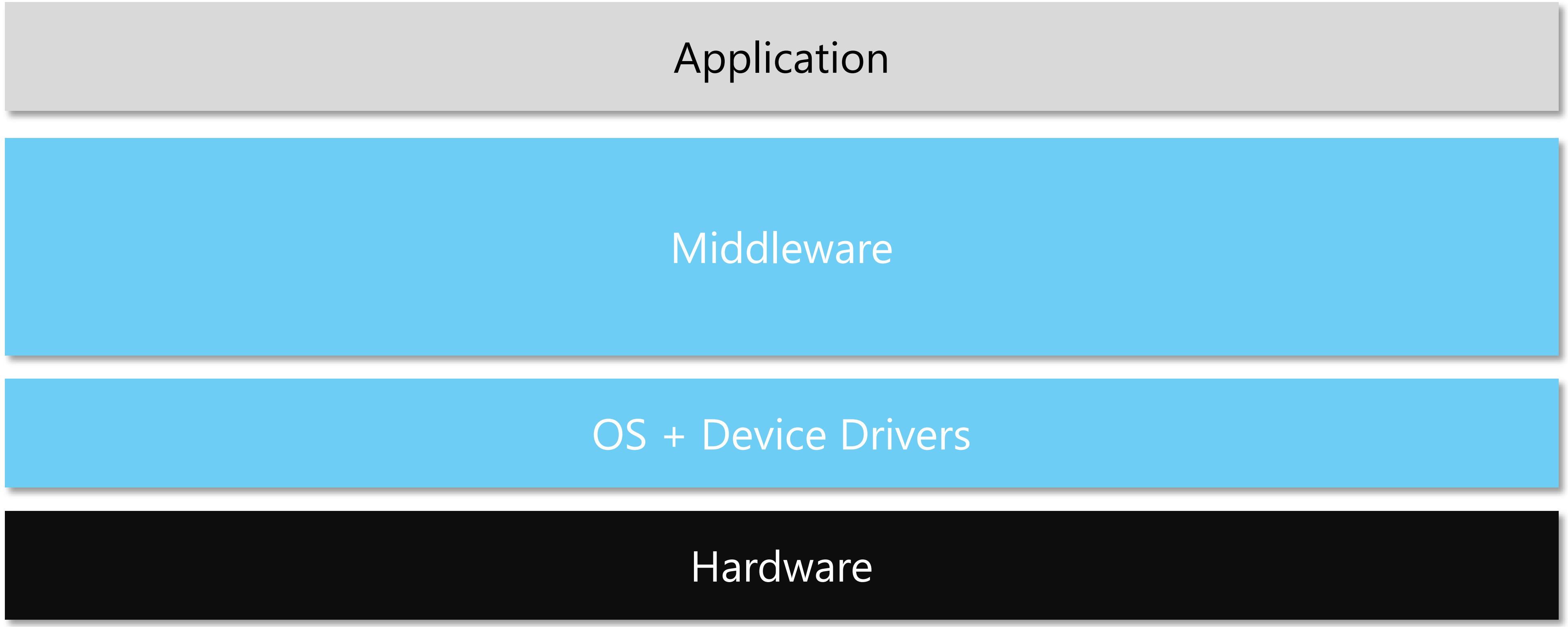
3 Overview of Software Layers

4 Overall Basic Software Architecture

Generic Layered Architecture

Generic layered architecture allows two different systems to communicate despite of their core architecture (hardware or software)

Layered architecture provides standard methodology to design flexible, robust and interoperable architecture



Application

Middleware

OS + Device Drivers

Hardware

Agenda

1 Introduction

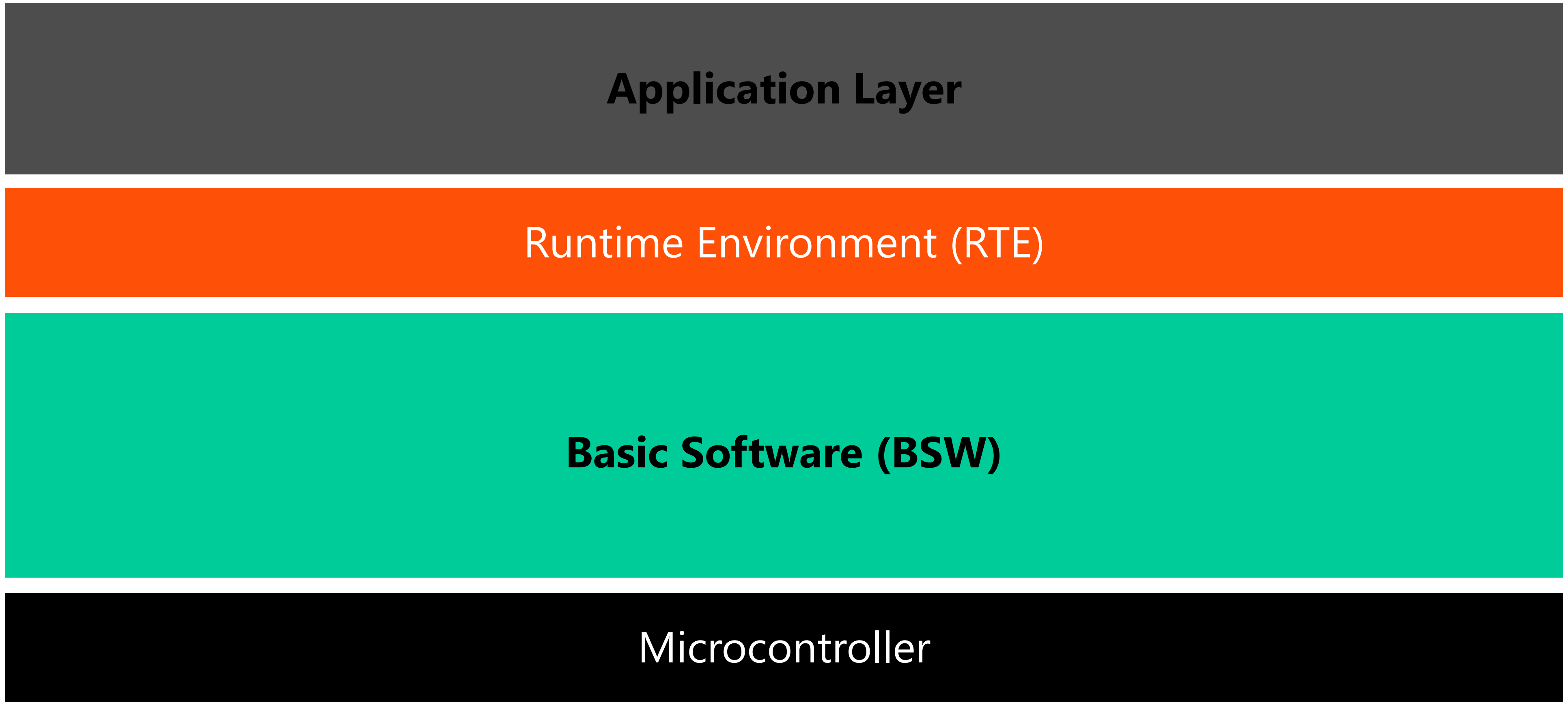
2 Generic Layered Architecture

3 Overview of Software Layers

4 Overall Basic Software Architecture

Overview of Software Layers : Top View

The AUTOSAR Architecture distinguishes on the highest abstraction level between three software layers : Application, Runtime Environment and Basic Software which run on a Microcontroller.



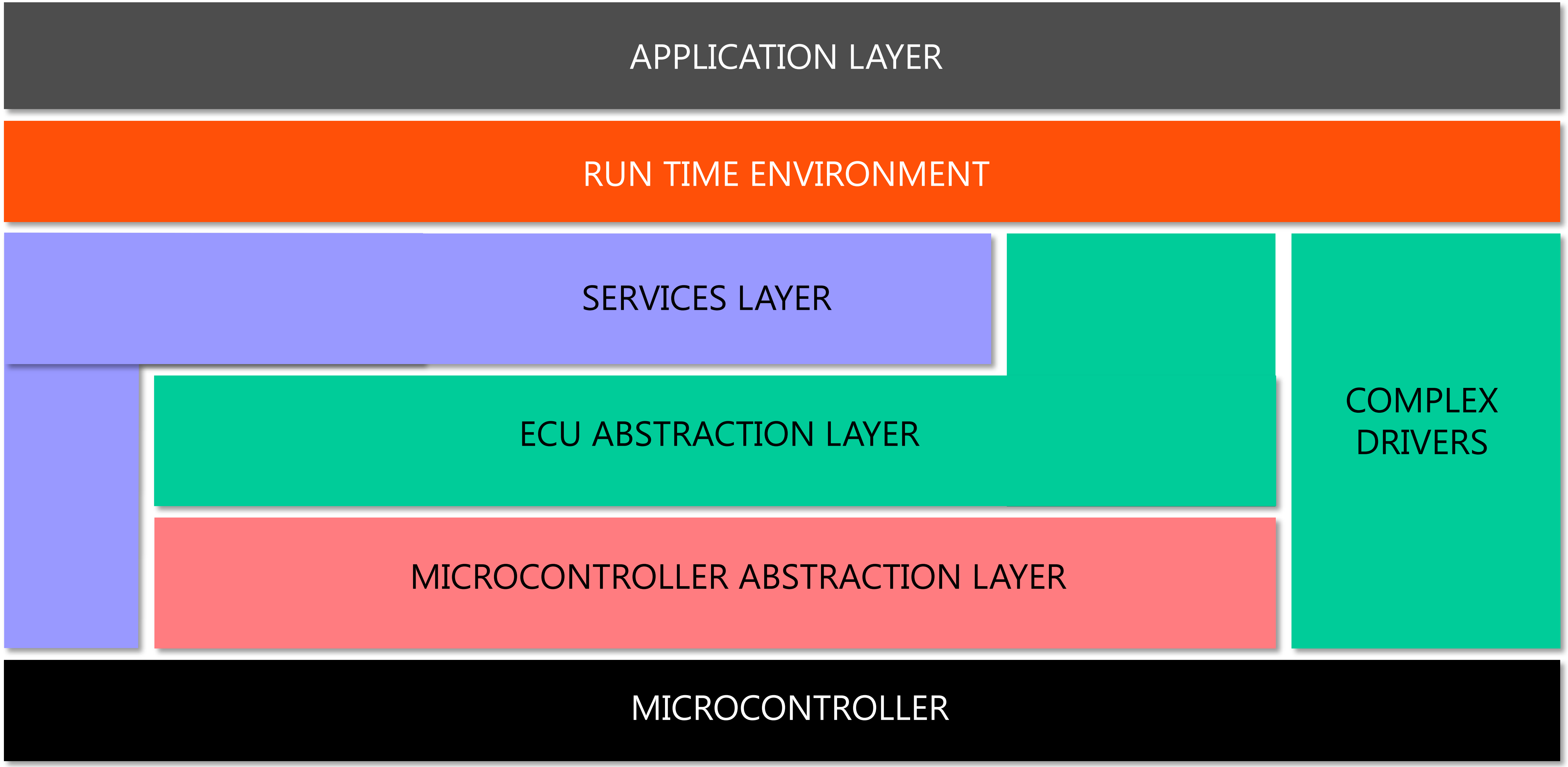
Application Layer

Runtime Environment (RTE)

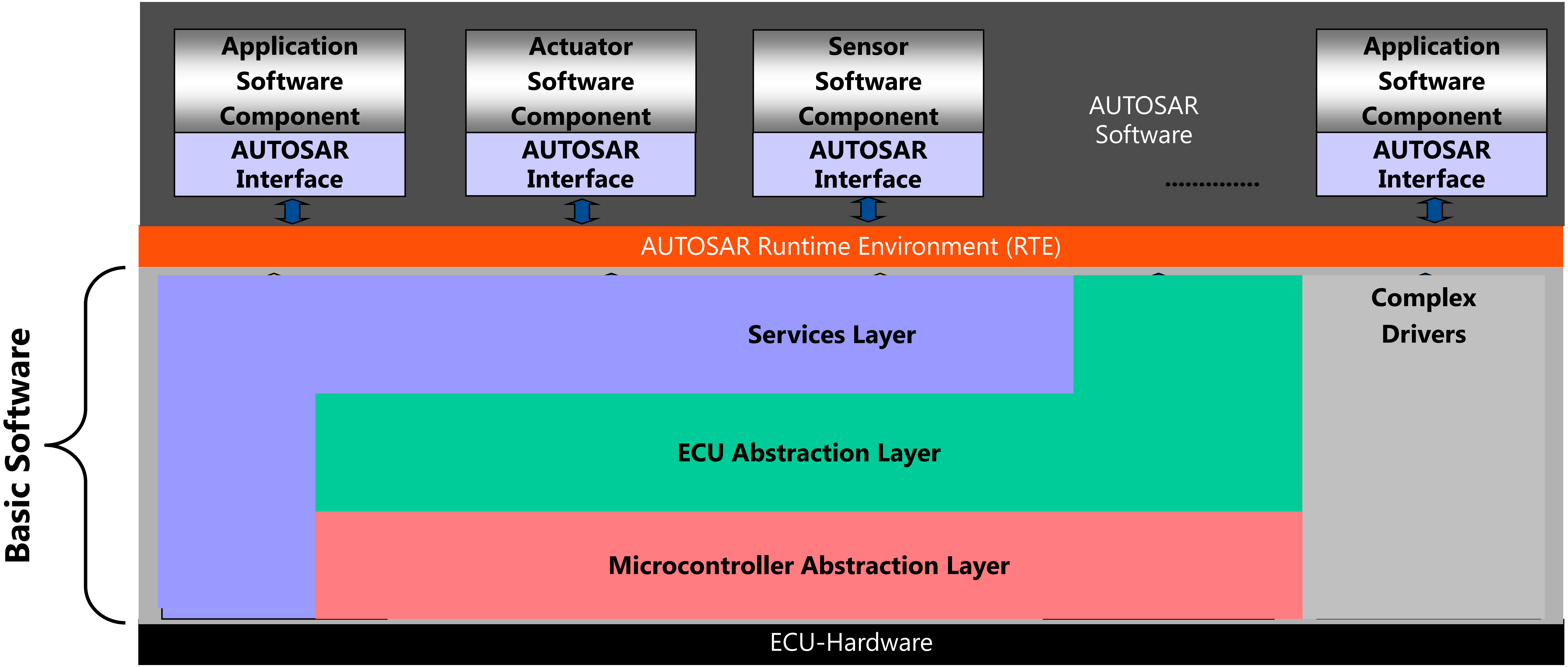
Basic Software (BSW)

Microcontroller

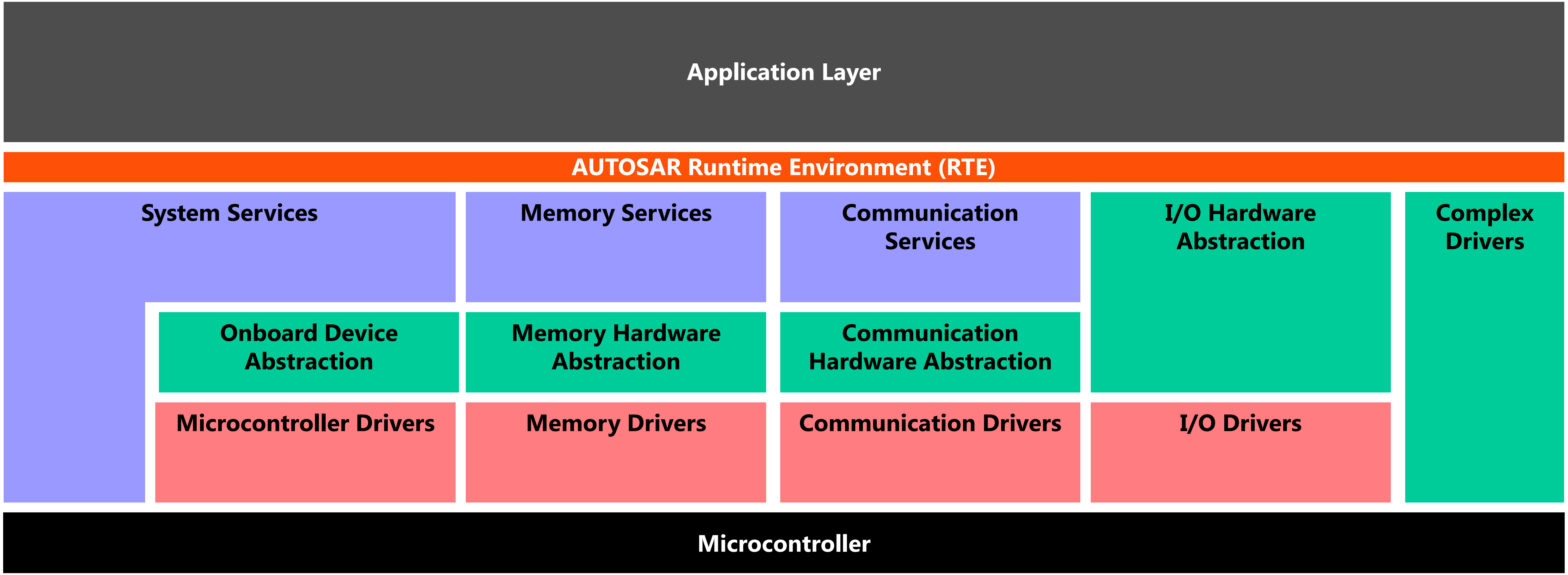
Overview of Software Layers : Coarse View



Overview of Software Layers : Simplified Component View


Application Software Component
AUTOSAR Interface
Actuator Software Component
AUTOSAR Interface
Sensor Software Component
AUTOSAR Interface
AUTOSAR Software
....
Application Software Component
AUTOSAR Interface
AUTOSAR Runtime Environment (RTE)
Basic Software
Services Layer
ECU Abstraction Layer
Microcontroller Abstraction Layer
Complex Drivers
ECU-Hardware

Overview of Software Layers : Detailed View



Application Layer

AUTOSAR Runtime Environment (RTE)

System Services

Memory Services

Communication Services

I/O Hardware Abstraction

Complex Drivers

Onboard Device Abstraction

Memory Hardware Abstraction

Communication Hardware Abstraction

Microcontroller Drivers

Memory Drivers

Communication Drivers

I/O Drivers

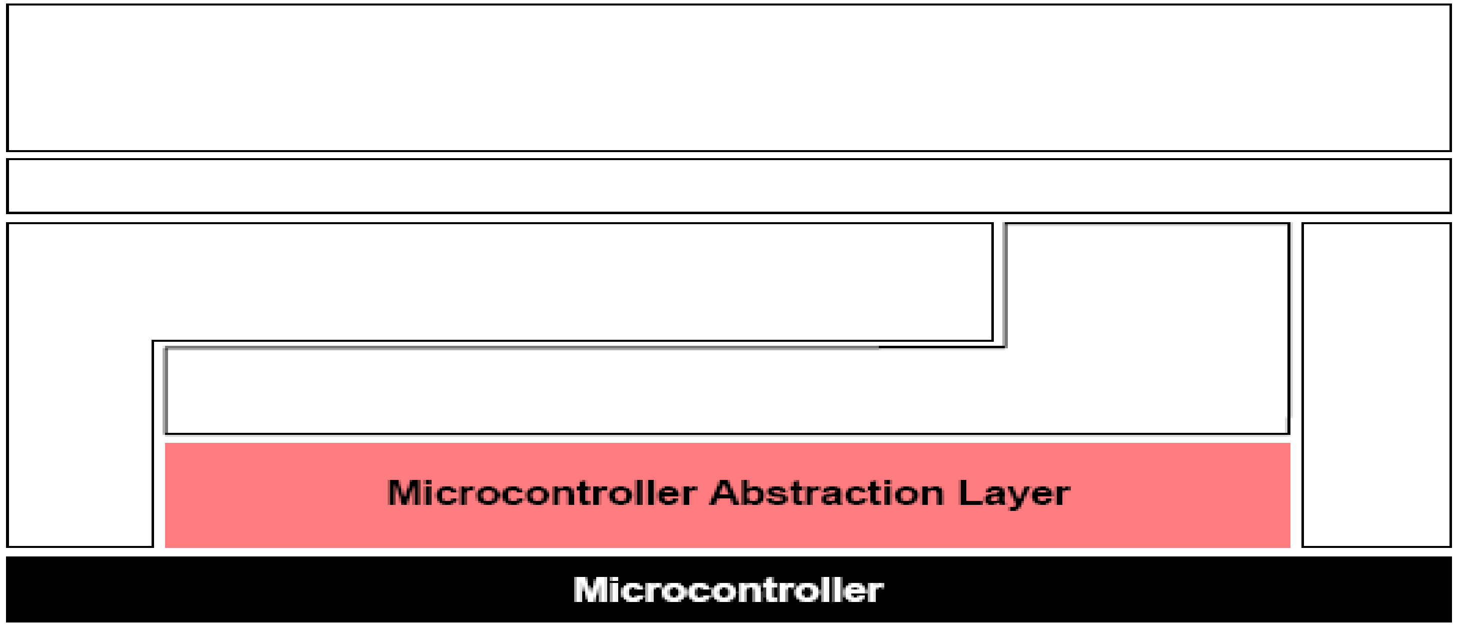
Microcontroller

Microcontroller Abstraction Layer

The **Microcontroller Abstraction Layer is the lowest software layer of the Basic Software.**

It contains internal drivers, which are software modules with direct access to the microcontroller and internal peripherals.

Task
Make higher software layers independent of microcontroller



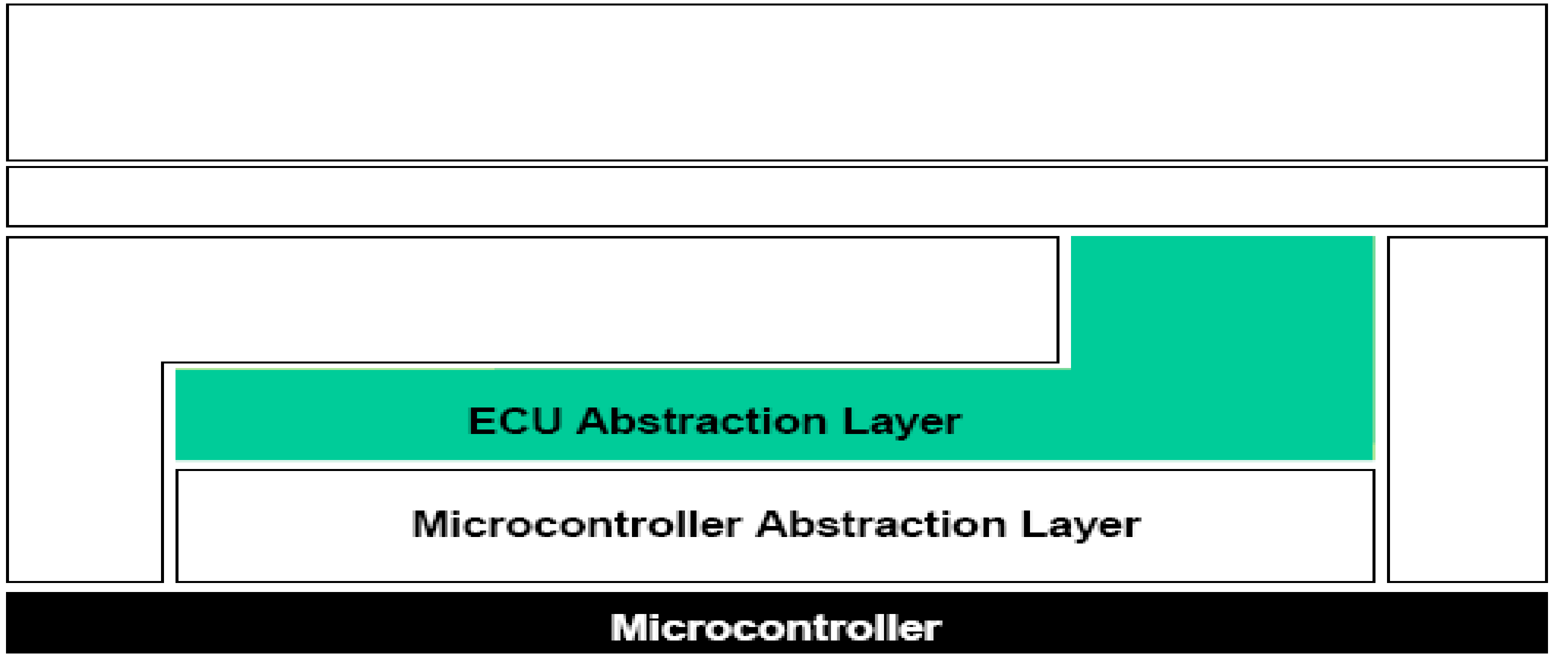
ECU Abstraction Layer

The ECU Abstraction Layer interfaces the drivers of the Microcontroller Abstraction Layer. It also contains drivers for external devices.

It offers an API for access to peripherals and devices regardless of their location (µC internal/external) and their connection to the µC (port pins, type of interface)

Task

Make higher software layers independent of ECU hardware layout



Complex Drivers

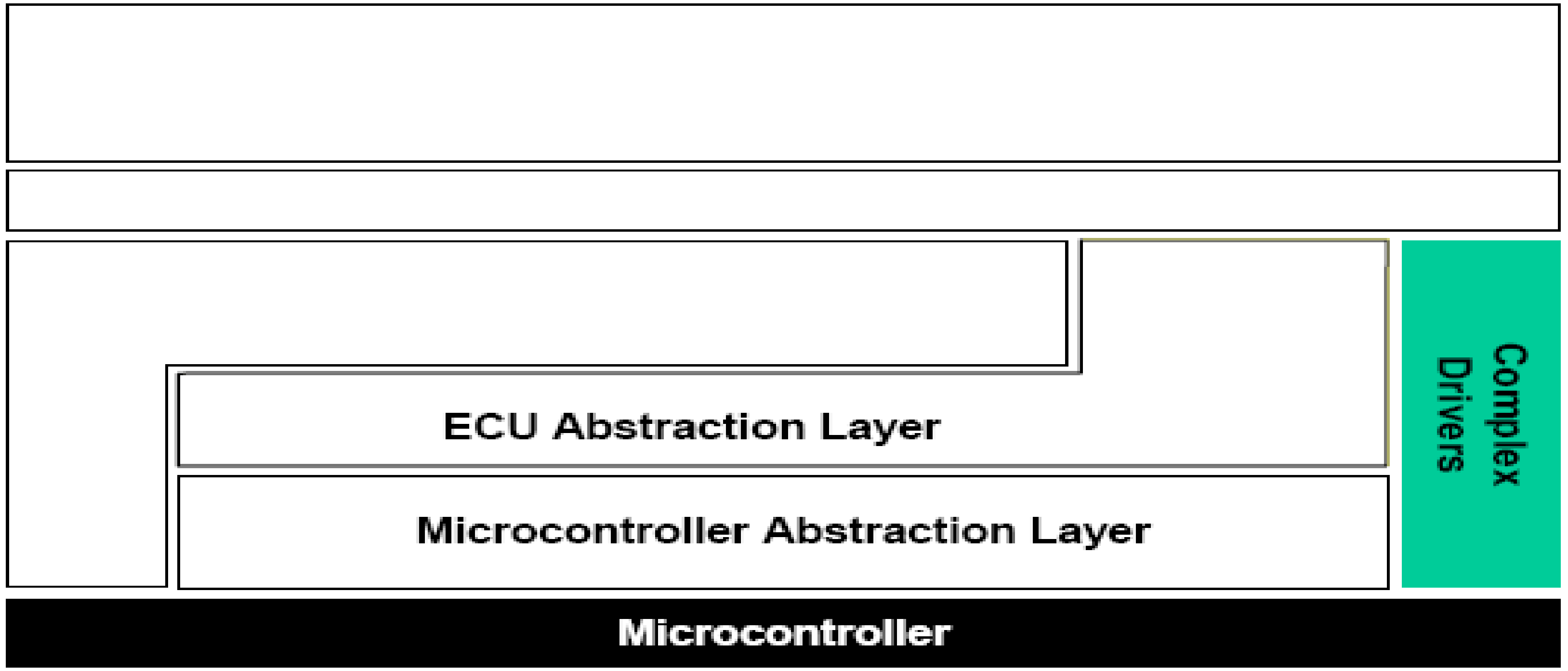
The **Complex Drivers Layer spans from the hardware to the RTE.**

It offers an API for access to peripherals and devices regardless of their location (µC internal/external) and their connection to the µC (port pins, type of interface)

Task

Provide the possibility to integrate special purpose functionality, e.g. drivers for devices:

- which are not specified within AUTOSAR,
- with very high timing constrains or
- for migration purposes etc.

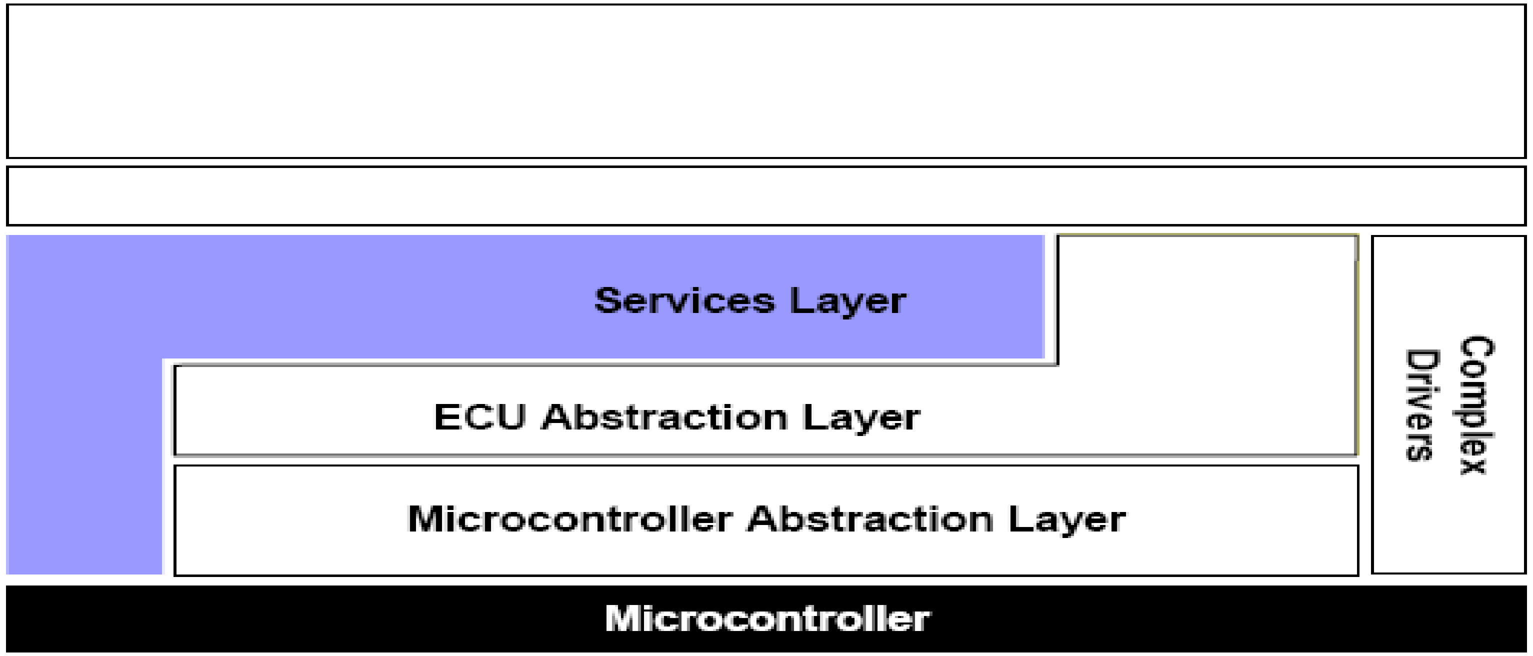


Services Layer

The Services Layer is the highest layer of the Basic Software which also applies for its relevance for the application software: while access to I/O signals is covered by the ECU Abstraction Layer, the Services Layer offers

- Operating system functionality
- Vehicle network communication and management services
- Memory services (NVRAM management)
- Diagnostic Services (including UDS communication, error memory and fault treatment)
- ECU state management, mode management
- Logical and temporal program flow monitoring (Wdg manager)

Task
Provide basic services for applications and basic software modules.



Runtime Environment (RTE)

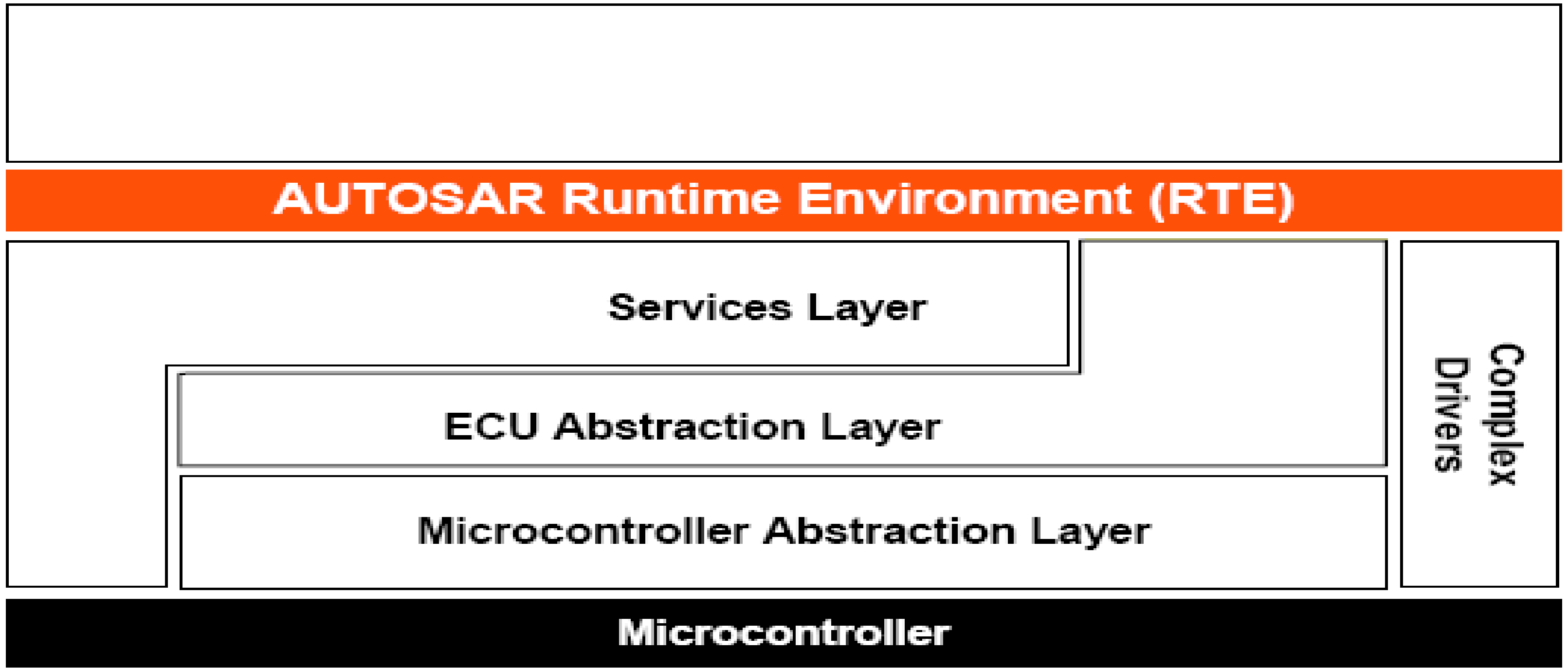
The **RTE is a layer providing communication services to the application software (AUTOSAR Software Components and/or AUTOSAR Sensor/Actuator components).**

Above the RTE the software architecture style changes from “layered“ to “component style“.

The AUTOSAR Software Components communicate with other components (inter and/or intra ECU) and/or services via the RTE.

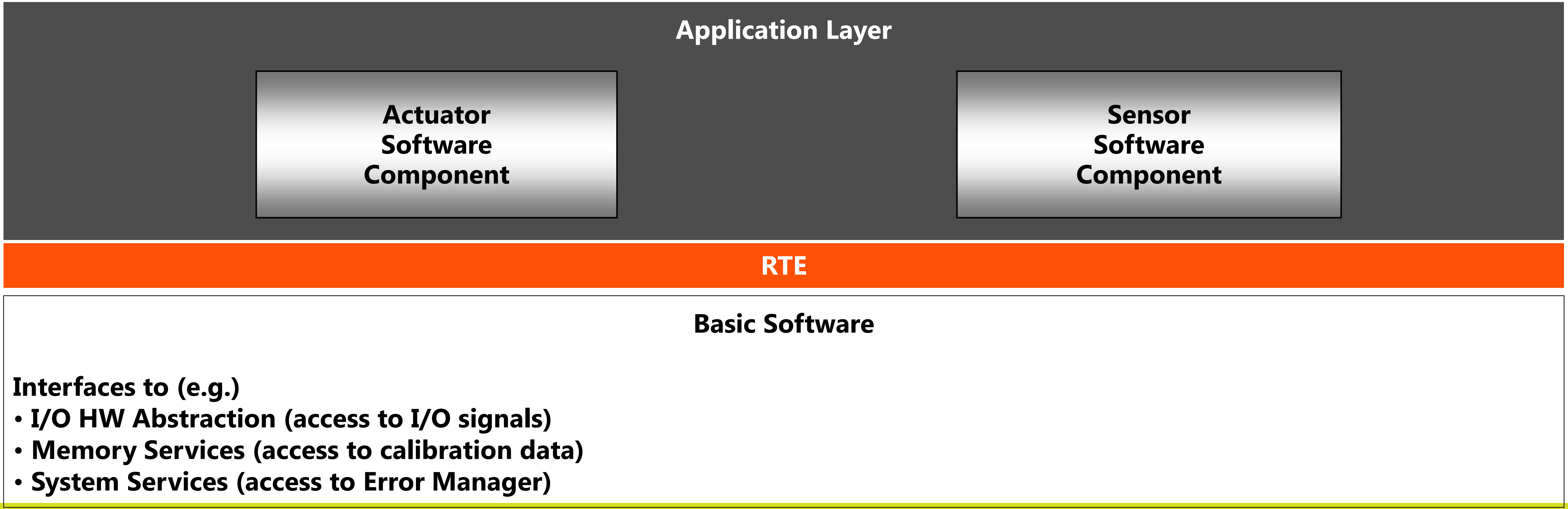
Task

Make AUTOSAR Software Components independent from the mapping to a specific ECU.


AUTOSAR Runtime Environment (RTE)
Services Layer
ECU Abstraction Layer
Microcontroller Abstraction Layer
Complex Drivers
Microcontroller

Application Layer and RTE

- Microcontroller and ECU HW independent and sensor and actuator dependent
- Abstract from the specific physical properties of sensors and actuators
- RTE is a layer providing communication services to the application software
- AUTOSAR Software Components communicate with other components (inter
- and/or intra ECU) and/or services via the RTE



Application Layer

Actuator Software Component

Sensor Software Component

RTE

Basic Software

Interfaces to (e.g.)
- **I/O HW Abstraction (access to I/O signals)**
- **Memory Services (access to calibration data)**
- **System Services (access to Error Manager)**

Agenda

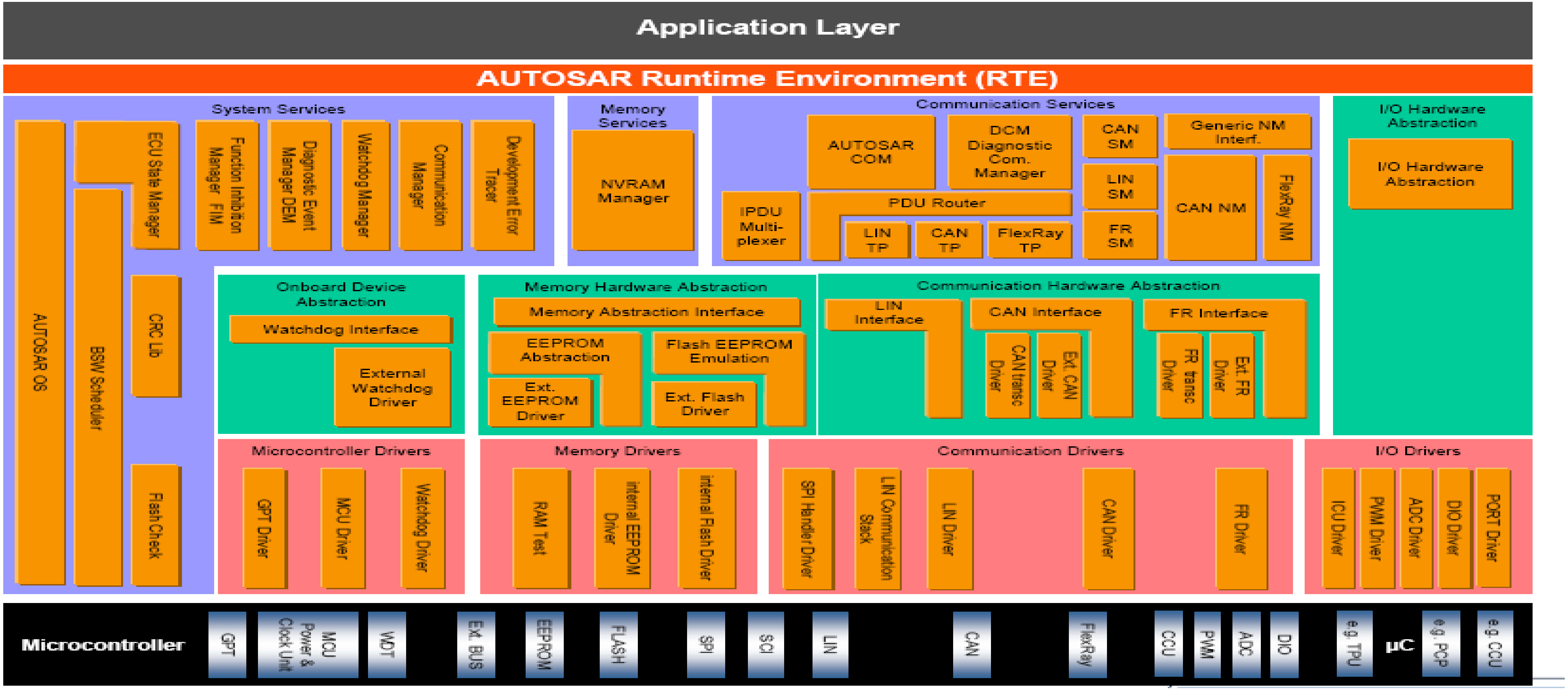
1 Introduction

2 Generic Layered Architecture

3 Overview of Software Layers

4 Overall Basic Software Architecture

Overall Basic Software Architecture



Application Layer

AUTOSAR Runtime Environment (RTE)

System Services

AUTOSAR OS

BSW Scheduler

ECU State Manager

Function Inhibition Manager FIM

Diagnostic Event Manager DEM

Watchdog Manager

Communication Manager

Development Error Tracer

CRC Lib

Flash Check

Memory Services

NVRAM Manager

Communication Services

AUTOSAR COM

DCM Diagnostic Com. Manager

CAN SM

LIN SM

FR SM

Generic NM Interf.

CAN NM

FlexRay NM

IPDU Multi-plexer

PDU Router

LIN TP

CAN TP

FlexRay TP

I/O Hardware Abstraction

I/O Hardware Abstraction

Onboard Device Abstraction

Watchdog Interface

External Watchdog Driver

Memory Hardware Abstraction

Memory Abstraction Interface

EEPROM Abstraction

Flash EEPROM Emulation

Ext. EEPROM Driver

Ext. Flash Driver

Communication Hardware Abstraction

LIN Interface

CAN Interface

CAN transc Driver

Ext. CAN Driver

FR Interface

FR transc Driver

Ext. FR Driver

Microcontroller Drivers

GPT Driver

MCU Driver

Watchdog Driver

Memory Drivers

RAM Test

internal EEPROM Driver

internal Flash Driver

Communication Drivers

SPI Handler Driver

LIN Communication Stack

LIN Driver

CAN Driver

FR Driver

I/O Drivers

ICU Driver

PWM Driver

ADC Driver

DIO Driver

PORT Driver

Microcontroller

GPT

MCU Power & Clock Unit

WDT

Ext. BUS

EEPROM

FLASH

SPI

SCI

LIN

CAN

FlexRay

CCU

PWM

ADC

DIO

e.g. TPU

µC

e.g. PCP

e.g. CCU

Questions

Thank You

www.kpit.com