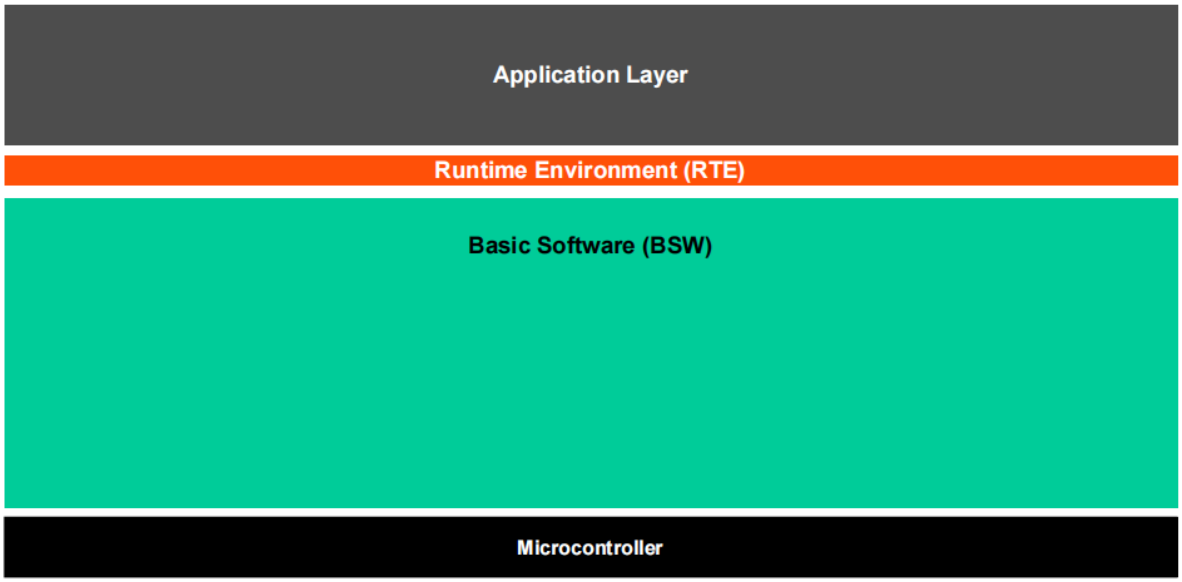
OVERVIEW OF SOFTWARE LAYERS OF AUTOSAR ARCHITECTURE

OVERVIEW OF SOFTWARE LAYERS OF AUTOSAR ARCHITECTURE
Top view
BSW
- Coarse view of BSW
- Detailed view of BSW
- Microcontroler Abstraction Layer（MCAL)
- ECU Abstraction Layer
- Complex Drivers
- Services Layer

RTE
Libraries
Types of Basic Software Module(BSW)
- Driver
  - Internal Driver
  - External Driver
- Interface
- Handler
- Manager

Top view

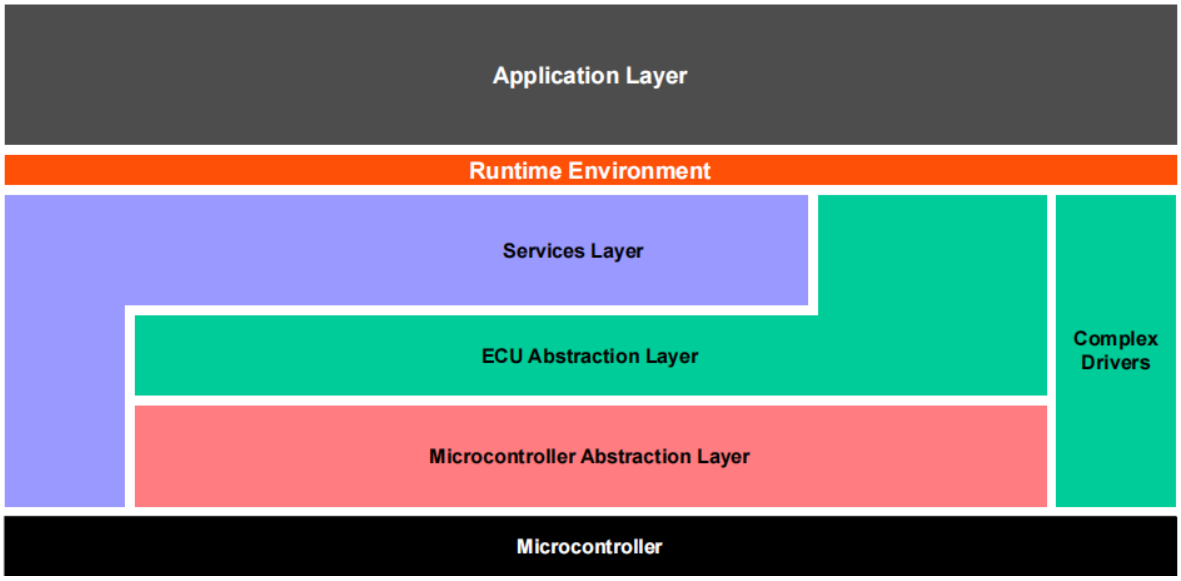
在最高的抽象级别上，AUTOSAR分为三个软件层（从上到下）：应用层（Application Layer）、运行时环境（Runtime Environment(RTE)）、基础软件层（Basic Software(BSW)）。在BSW之下，是微控制器（Microcontroller/MCU），不属于AUTOSAR Architecture。



BSW

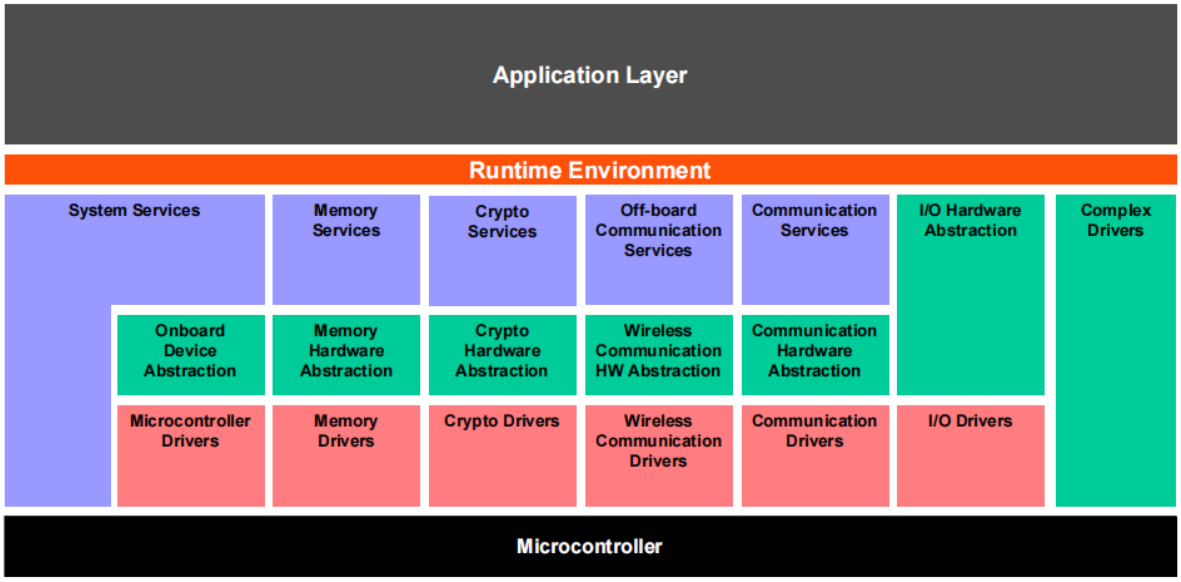
Coarse view of BSW

对于BSW，可以分成四层/部分：服务层（Services Layer）、ECU抽象层（ECU Abstraction Layer）、微控制器抽象层（Microcontroller Abstraction Layer(MCAL)）、复杂驱动层（Complex Drivers）。



Detailed view of BSW

更详细地，BSW的四个层次可以继续细分。



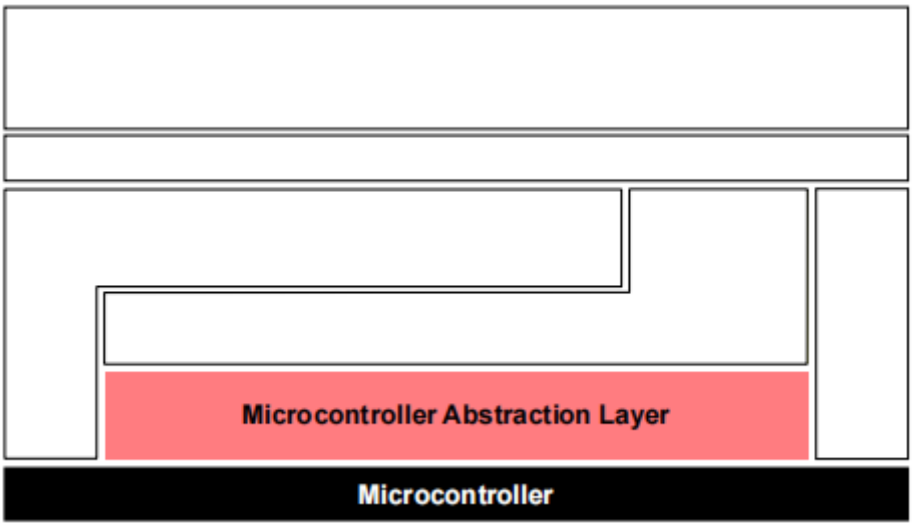
Microcontroler Abstraction Layer（MCAL）

MCAL是BSW的最底层。MCAL主要包含内部驱动，这些驱动是可以直接访问MCU和MCU内部外设的软件模块或驱动。

主要作用：向上层软件层提供标准化接口，使上层软件层独立于MCU。

实现依赖：具体的MCU。

MCAL中的内部驱动：微控制器驱动（Microcontroler Drivers）、内存驱动（Memory Drivers）、加密驱动（Crypto drivers）、无线通信驱动（Wireless Communication Drivers）、通信驱动（Communication Drivers）、I/O驱动（I/O Drivers）。（见“Detailed view of BSW”章节）



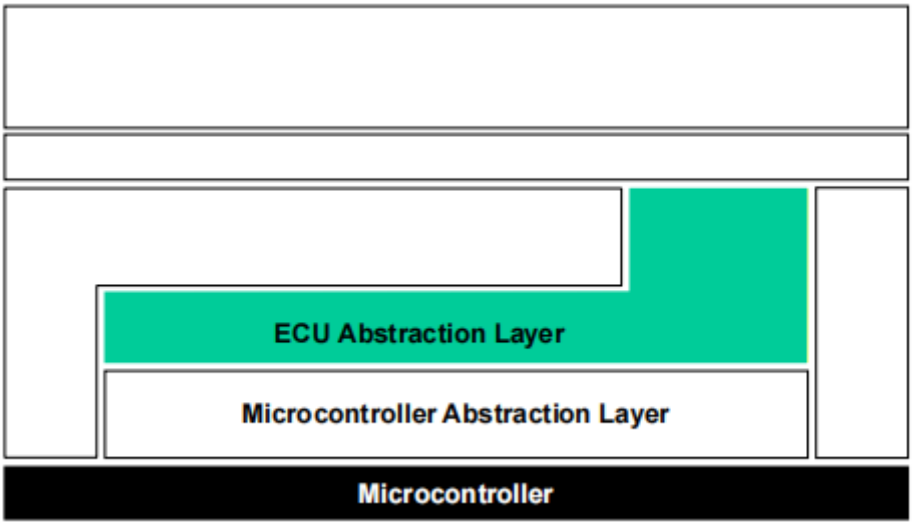
ECU Abstraction Layer

相对于MCAL，ECU抽象层包含了外部（MCU之外）设备驱动，并于MCAL对接，从而向上层提供进一步的抽象：对ECU的抽象。ECU抽象层向上层提供一组API访问来外设和设备。上层访问外设和设备时，不必关心这些外设和设备是位于MCU内还是之外，也不必关心它们与MCU的连接方式（port pins, type of interface）。PS. 外部驱动通过MCAL层的驱动（Internal Driver）访问外部设备。

主要作用：使上层独立于ECU硬件（含MCU）。

实现依赖：MCAL、ECU Hardware。

ECU 抽象层提供的抽象类型：板上设备抽象（Onboard Device Abstraction）、内存硬件抽象（Memory Hardware Abstraction）、加密硬件抽象（Crypto Hardware Abstraction）、无线通信硬件抽象（Wireless Communication HW Abstraction）、通信硬件抽象（Communication Hardware Abstraction）、I/O硬件抽象（I/O Hardware Abstraction）。（见“Detailed view of BSW”章节）

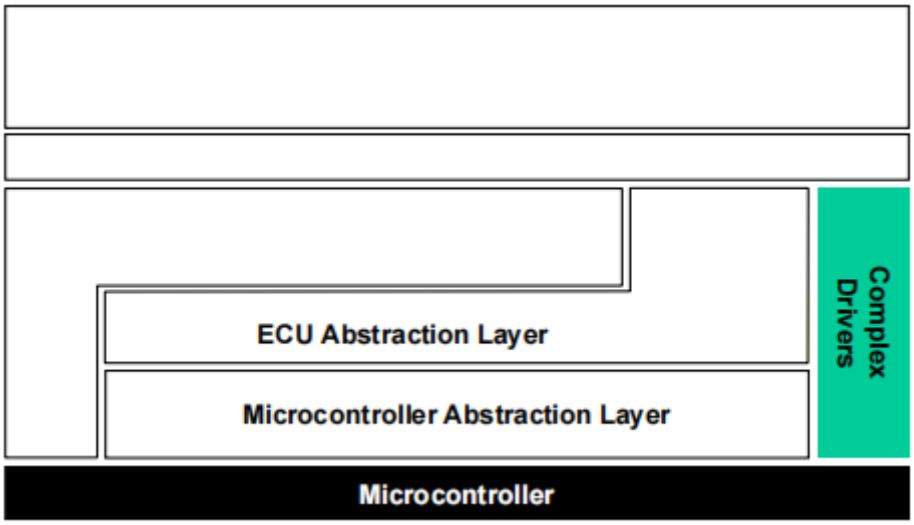


Complex Drivers

复杂驱动横跨硬件到RTE层。

主要作用：为特殊目的的功能提供集成进AUTOSAR的可能性，比如设备驱动：这些驱动未在AUTOSAR中进行规范说明、具有非常高的时间约束条件、或者用于迁移目的等待。

实现依赖：Complex Drivers可能实现为应用软件，依赖于MCU和ECU硬件。



Services Layer

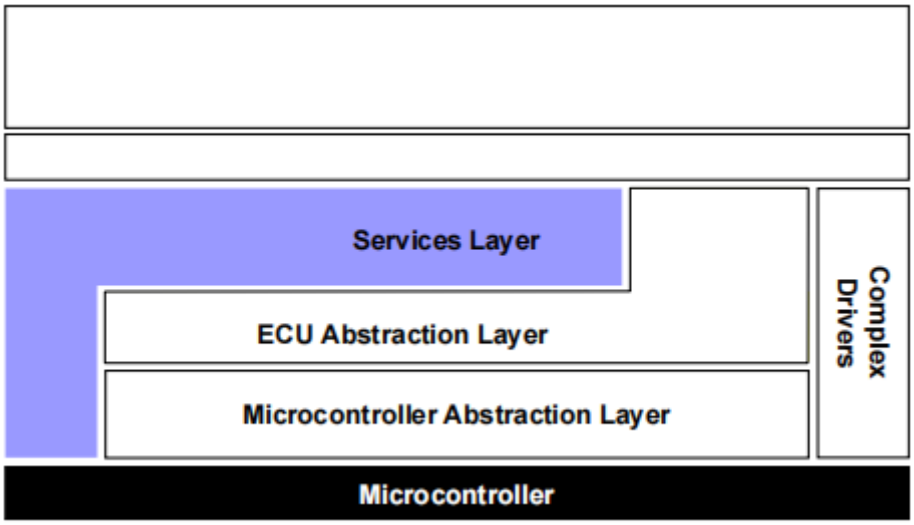
服务层是BSW的最高层，且与应用软件有联系：当应用软件访问被ECU抽象层覆盖/抽象的I/O信号等时，服务层向应用软件提供相应接口。

主要作用：为应用软件、RTE、BSW提供基础服务。

实现依赖：Services Layer的实现不依赖于MCU和ECU硬件，也使得上层独立于MCU和ECU硬件。

服务类型：（见“Detailed view of BSW”章节）

- **Input/Output(I/O)：**标准化访问sensors、actuators和ECU板上外设；
- **Memory：**标准化访问内部或外部存储器（非易失性存储器）；
- **Crypto：**标准化访问密码原语，包括内部或外部硬件加速器；
- **Communication：**标准化访问：车载网络系统、ECU板上通信系统、ECU内部软件；
- **Off-board Communication：**标准化访问：车辆到X的通信、车内无线网络系统、ECU非车载通信系统；
- **System：**提供可标准化的（**操作系统、定时器、错误存储**）和特定的ECU（**ECU状态管理、看门狗管理**）服务和库功能。

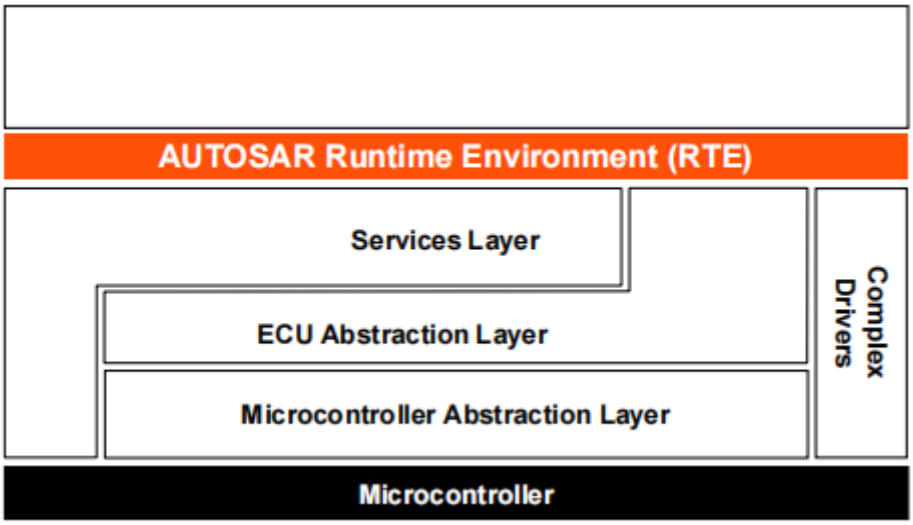


RTE

运行时环境为应用软件（AUTOSAR Software Components and/or AUTOSAR Sensor/Actuator components. I.e. SWC）提供通信服务。在RTE之上，软件的架构风格从“分层”转变为“组件风格”。AUTOSAR软件组件（在同一ECU上或不同ECU之间的软件组件）通过RTE进行相互通信。

主要作用：使上层（AUTOSAR软件组件，即SWC）完全独立于ECU，即不依赖于ECU。无论它们是映射到同一还是不同ECU上，SWC之间都可以通过RTE的相关接口进行通信。

实现依赖：RTE的实现依赖于ECU和应用软件，即每个ECU独立生成自己的RTE。



Libraries

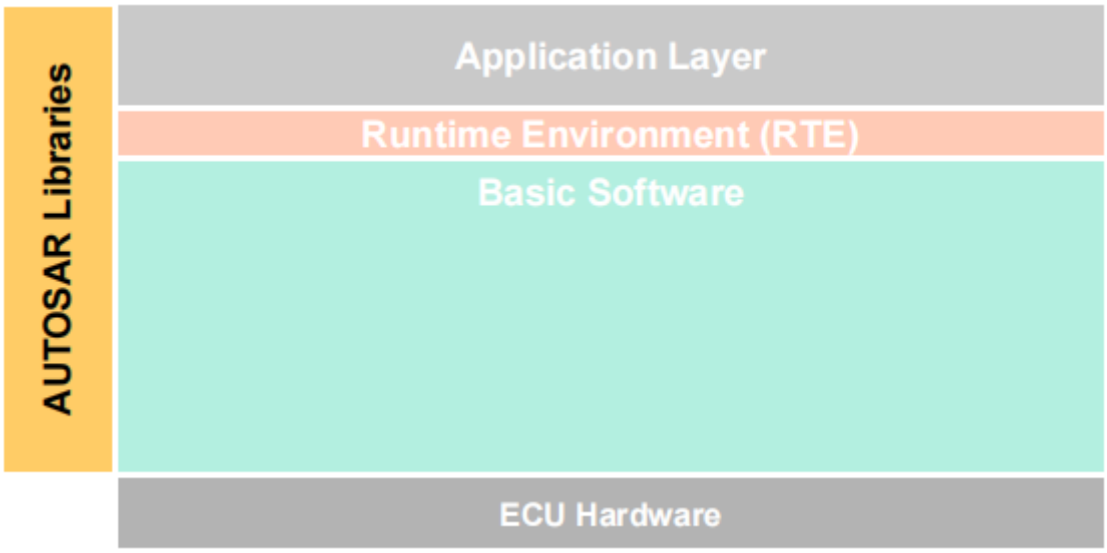
Libraries是用于相关目的的函数的集合。

特性：

- 可被BSW模块（包括RTE），SW-Cs、libraries或者集成代码调用；
- 在同一保护环境中的调用者的上下文中运行；
- 可重新进入；
- 没有内部状态；
- 不需要任何初始化；
- 是同步的，即它们没有等待点。

在AUTOSAR中描述的库：

- Fixed point mathematical,
- Floating point mathematical,
- Interpolation for fixed point data,
- Interpolation for floating point data,
- Extended functions (e.g. 64bits calculation, filtering, etc.),
- Bit handling,
- E2E communication,
- CRC calculation,
- Atomic multicore safe operations.



Types of Basic Software Module(BSW)

Driver

Driver可控制和访问内部或外部设备。

Internal Driver

内部设备指的是MCU内部的设备。比如：Internal EEPROM、Internal CAN controler、Internal ADC。

内部设备的驱动称为内部驱动（Internal Driver），位于MCAL层。

External Driver

外部设备指的是在ECU上，但在MCU外的设备。比如：External EEPROM、External watchdog、External flash。

外部设备的驱动称为外部驱动，位于ECU Abstraction Layer。外部驱动通过MCAL层的驱动（Internal Driver）访问外部设备。

例外：内存映射的外部设备（例如外部闪存）的驱动可以直接访问MCU。这些驱动位于MCAL层，因为它们依赖于MCU。

Interface

Interface包含了从在架构上位于它们之后的模块抽象而来的功能。比如：一个接口抽象于一个特定设备的硬件实现，它提供了一个通用的API来访问一个特定类型的设备，独立于该类型的现有设备的数量和不同设备的硬件实现。Interface不会改变数据的内容。

一般地，Interface位于ECU Abstraction Layer。

例子：一个CAN通信系统的接口提供了一个通用的API来访问CAN通信网络，独立于ECU上CAN控制器的数量和硬件实现（on chip, off chip）。

Handler

处理程序是一个特定的接口，它控制一个或多个客户端对一个或多个驱动程序的并发、多发、异步访问。即，它执行缓冲、排队、仲裁、多路复用功能。它不会改变数据的内容。

Handler通常包含在驱动和接口中（比如：SPIHandlerDriver、ADCDriver）。

Manager

Manager为多客户端提供特定的服务。在纯Handler的功能不足以从多个客户端进行抽象的所有情况下，都需要它。

一般地，Manager位于Service Layer。

例子：NVRAM manager管理对内部或外部存储器（比如Flash、EEPPROM）的并发访问。它还执行分布式的和可靠的数据存储、数据检查、默认值的提供等功能。