NXP's AUTOSAR Solution for Ethernet Switches and Transceivers

Mihai Morarescu

Senior Software Engineer

October 2019 | Session #AMF-AUT-T3841

SECURE CONNECTIONS FOR A SMARTER WORLD

Company Public – NXP, the NXP logo, and NXP secure connections for a smarter world are trademarks of NXP B.V. All other product or service names are the property of their respective owners. © 2019 NXP B.V.

Agenda

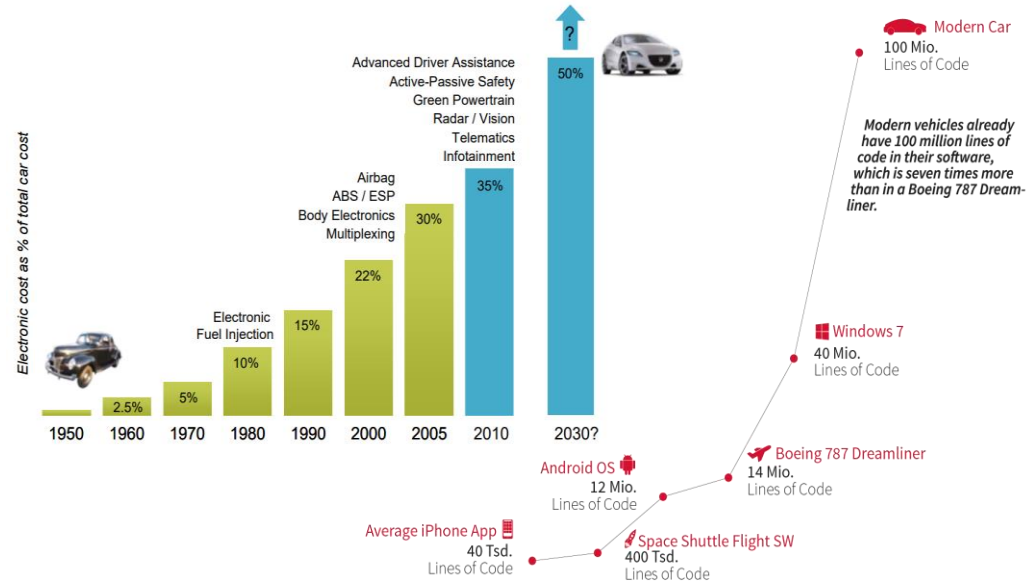
- Introduction
- Ethernet Switch Driver
- Ethernet Transceiver Driver
- Sample Application
- Conclusions
- Q&A

Introduction

AUTOSAR Overview

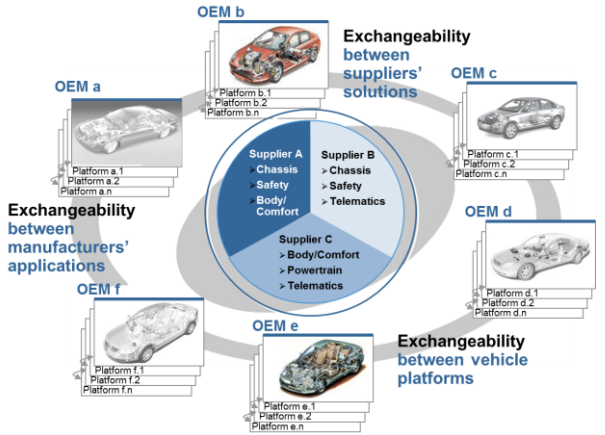
The challenges:

- Complexity of automotive E/E systems is growing
- Amount of software in cars grows exponentially
- Number of HW platforms and interdependencies increased
- Development processes and data formats are not harmonized

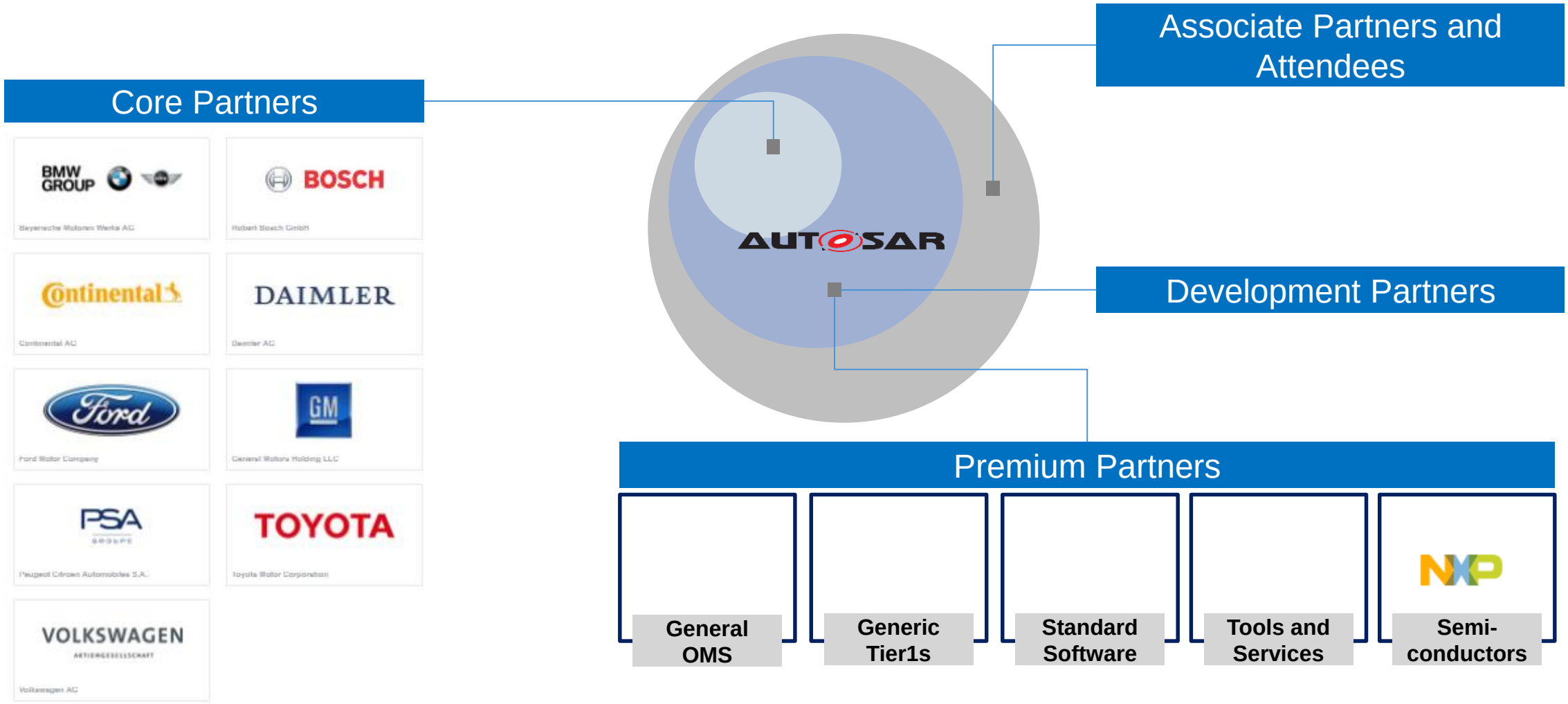


The main objective of AUTOSAR:

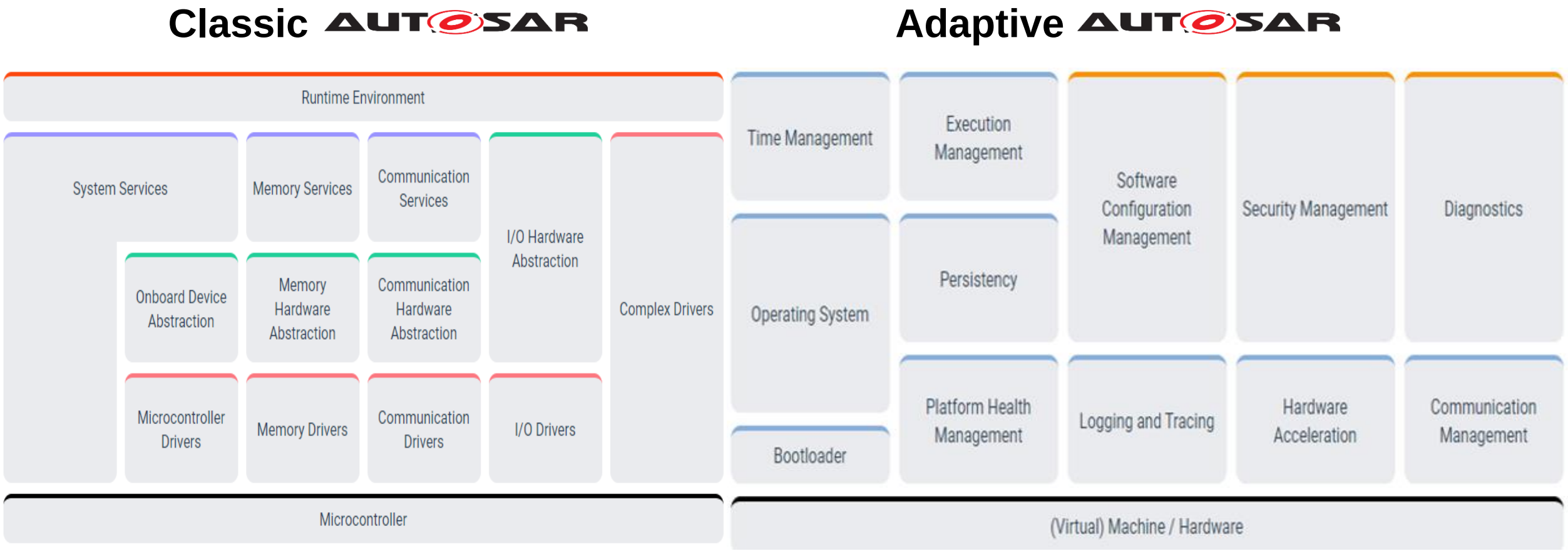
- **Master the ECUs growing complexity by enhancing SW quality and re-usability**
  - Functionalities are exchangeable and reusable between OEM and supplier
  - The development methods and tools are re-used
  - Common basic software, concentrating on functions with competitive value



AUTOSAR Overview



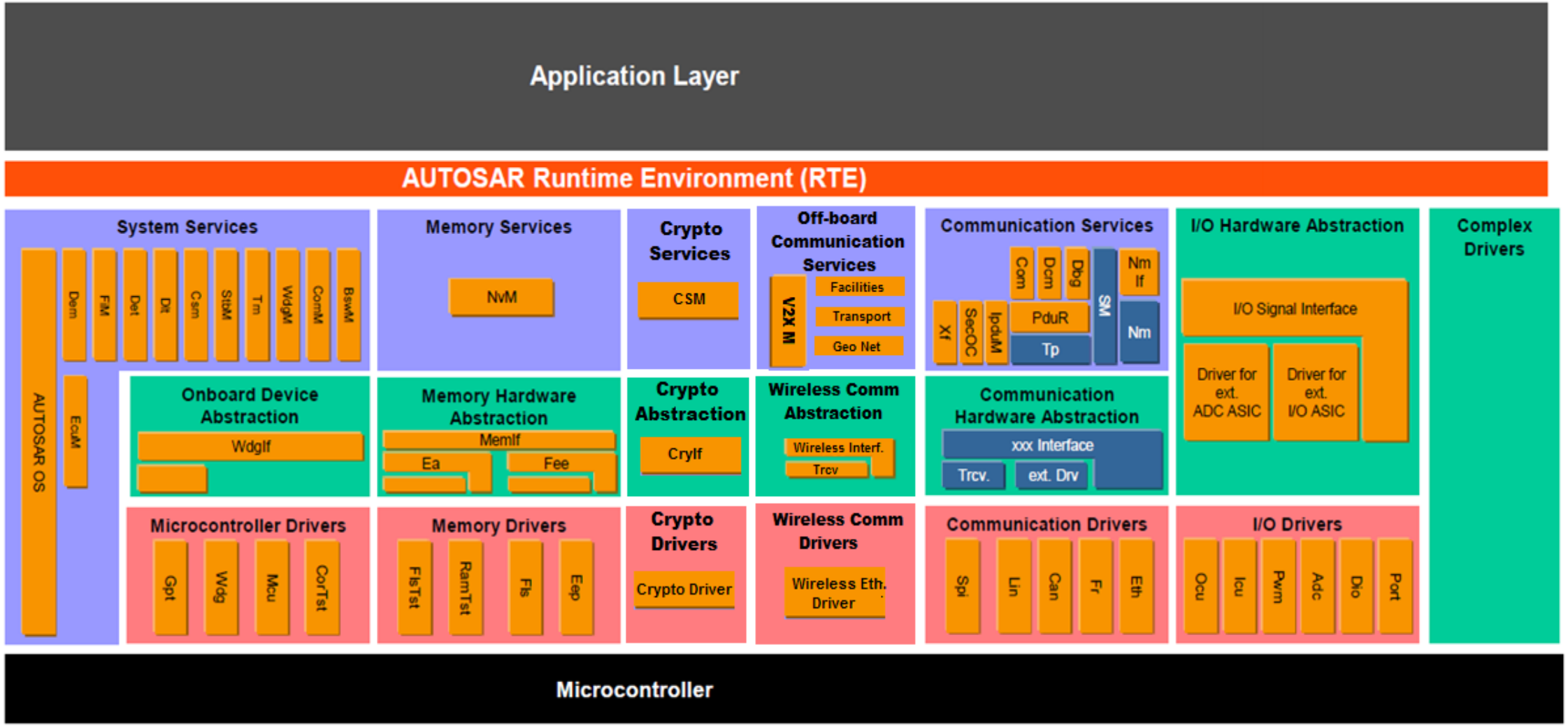
AUTOSAR Overview – Classic vs Adaptive



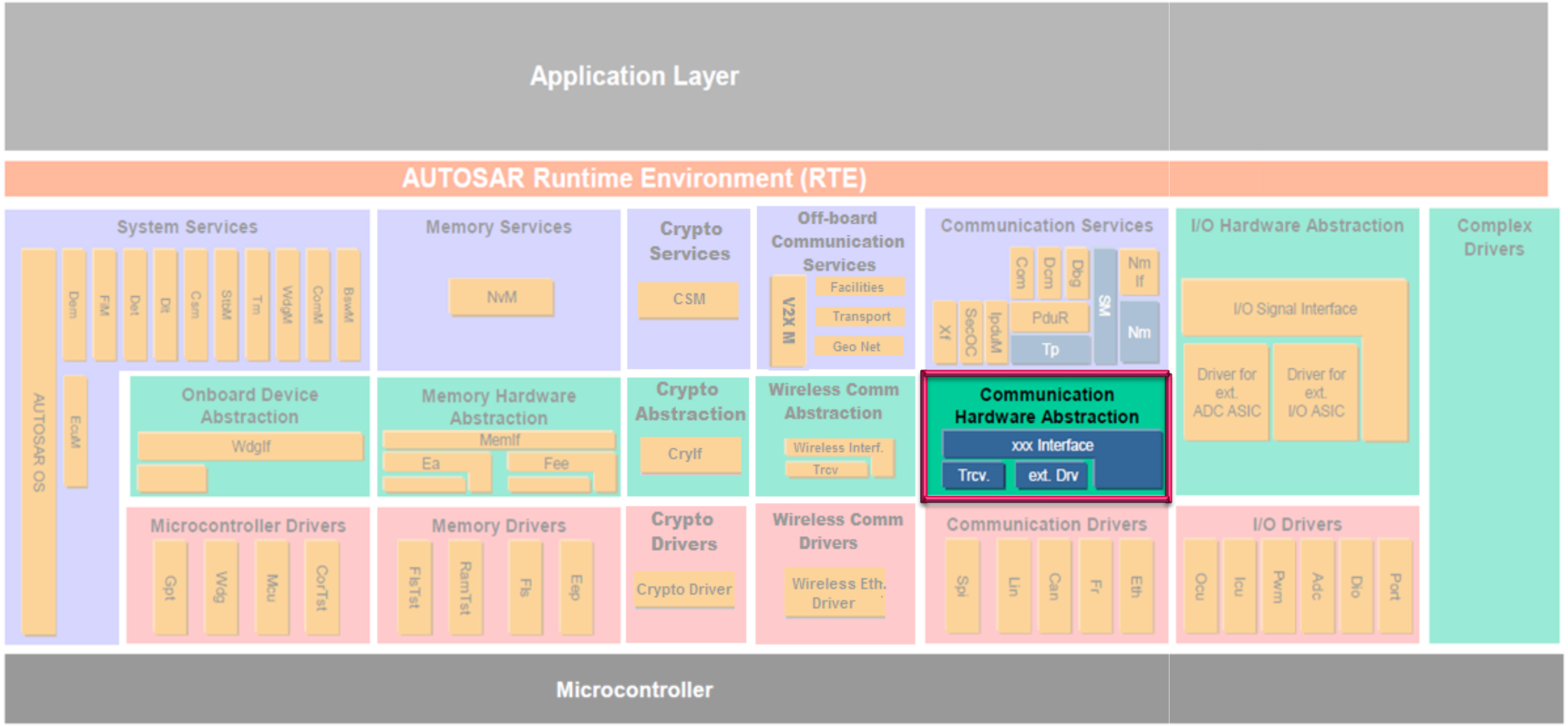
AUTOSAR Overview – Classic vs Adaptive

	AUTOSAR Classic Platform	AUTOSAR Adaptive Platforms
Operating System	OSEK OS	POSIX specification
Communication Protocols	Signal-based Communication (CAN, FlexRay, Most)	Service Oriented Communication (SOME/IP)
Scheduling Mechanisms	Fixed task configuration	Dynamic scheduling strategies
Memory Management	Same address space for applications (MPU)	Virtual address space for each app (MMU)

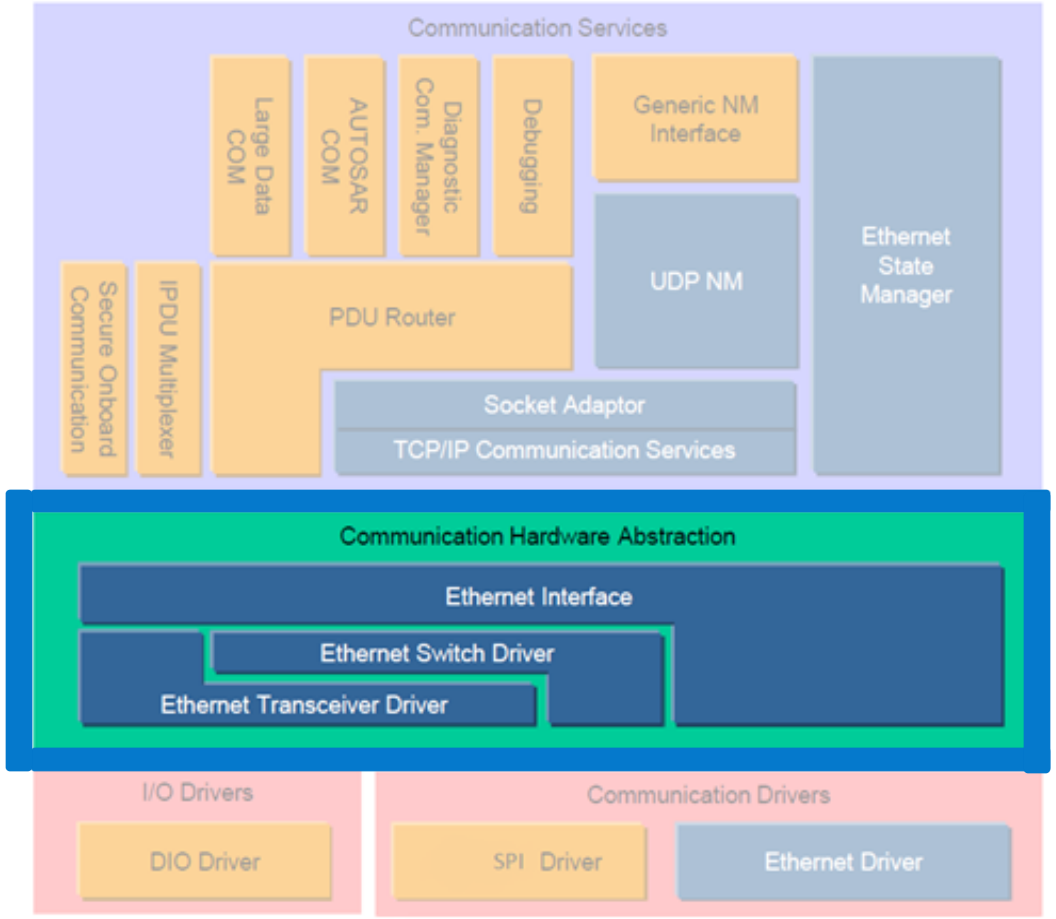
AUTOSAR Classic – Layered Architecture



AUTOSAR Classic – Layered Architecture



AUTOSAR Classic – Ethernet Switch and Transceiver



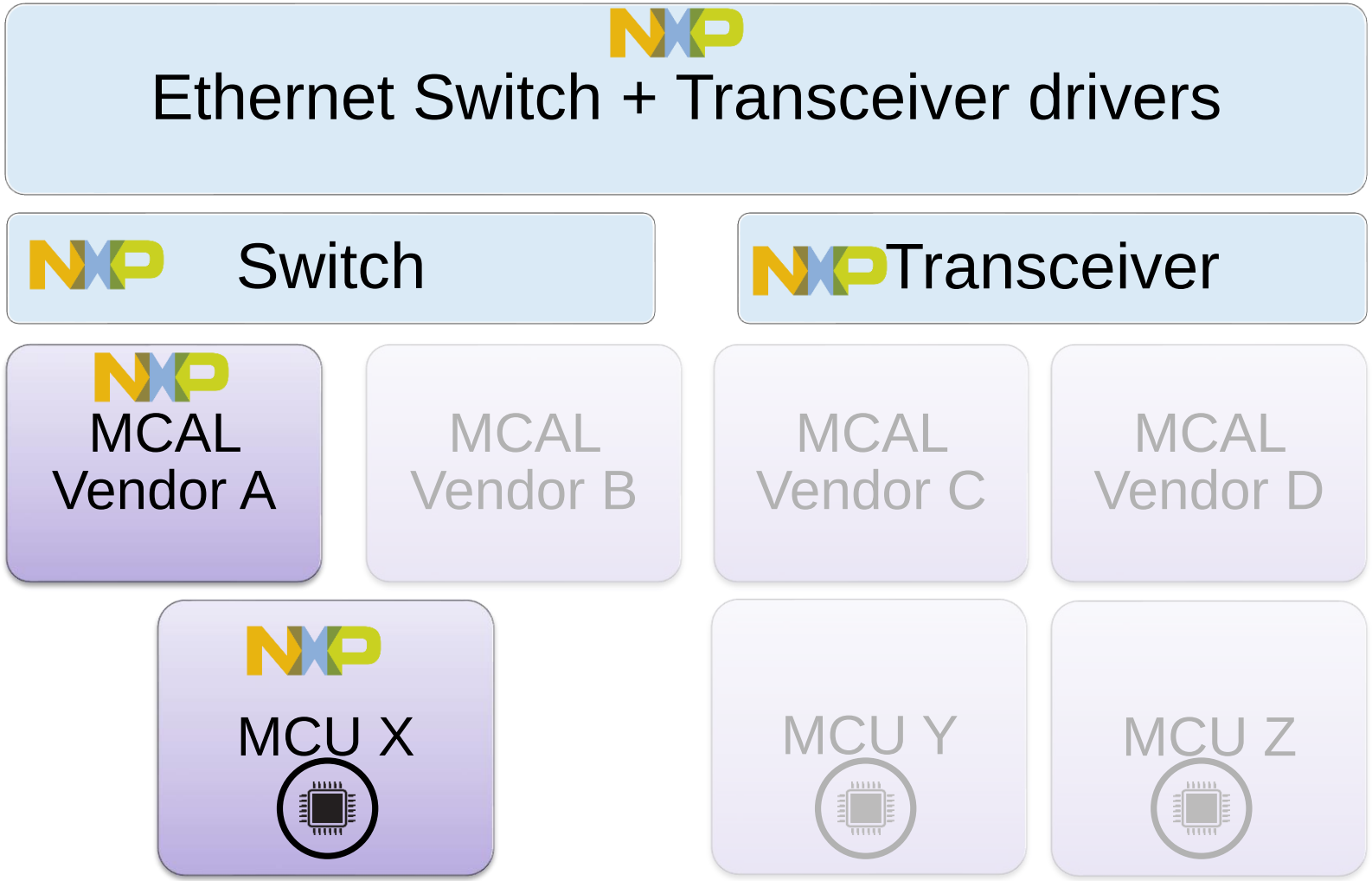
2019
Release 4.4.0
2017
Release 4.3.0
2014
Release 4.2.1
2009
Release 4.0
Release 3.0
Release 2.0
Release 1.0
2002
Start of AUTOSAR

1st NXP Autosar drivers for Switch/Transceiver products

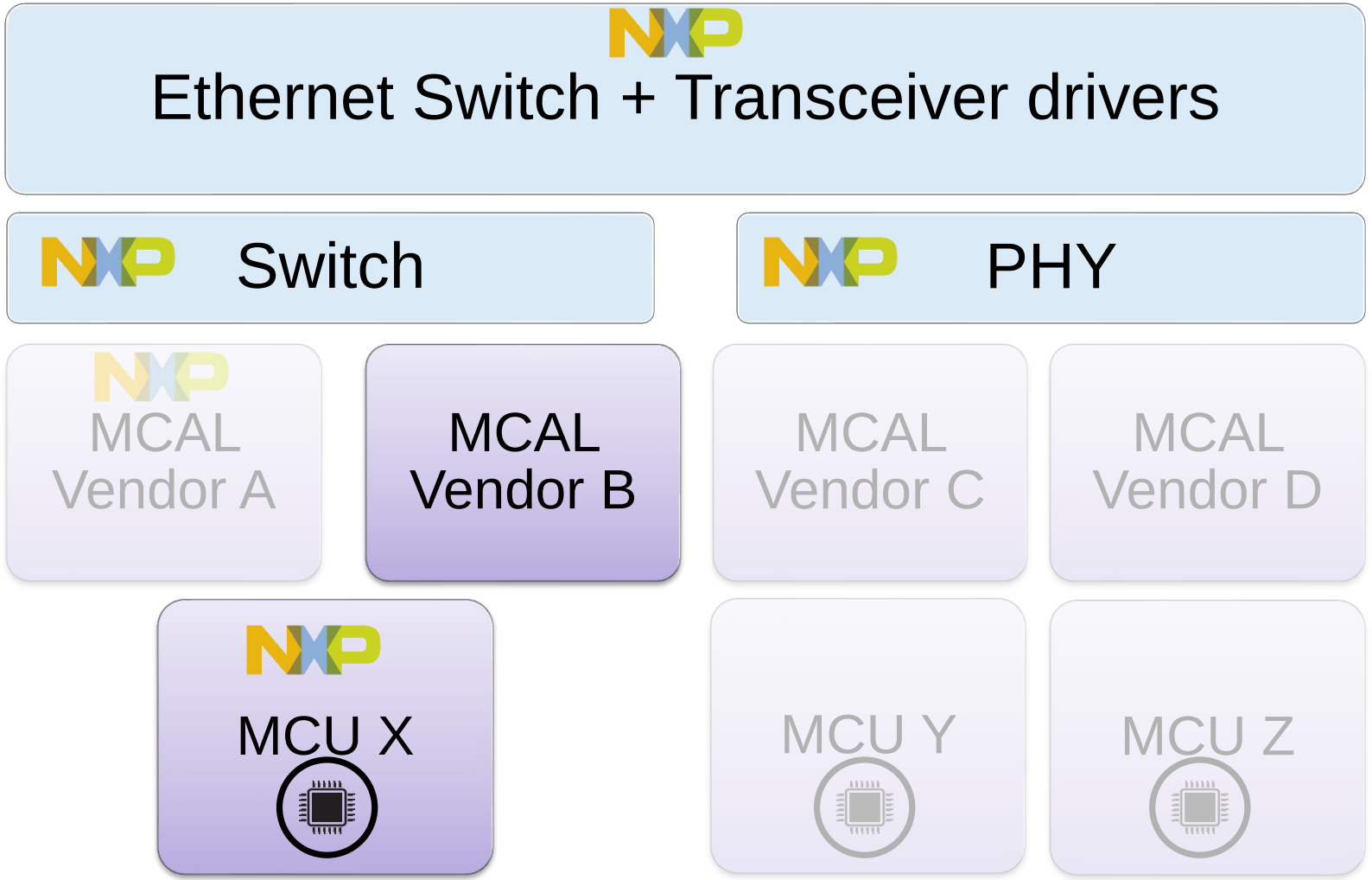
1st Ethernet Switch software specification

1st Ethernet Transceiver software specification

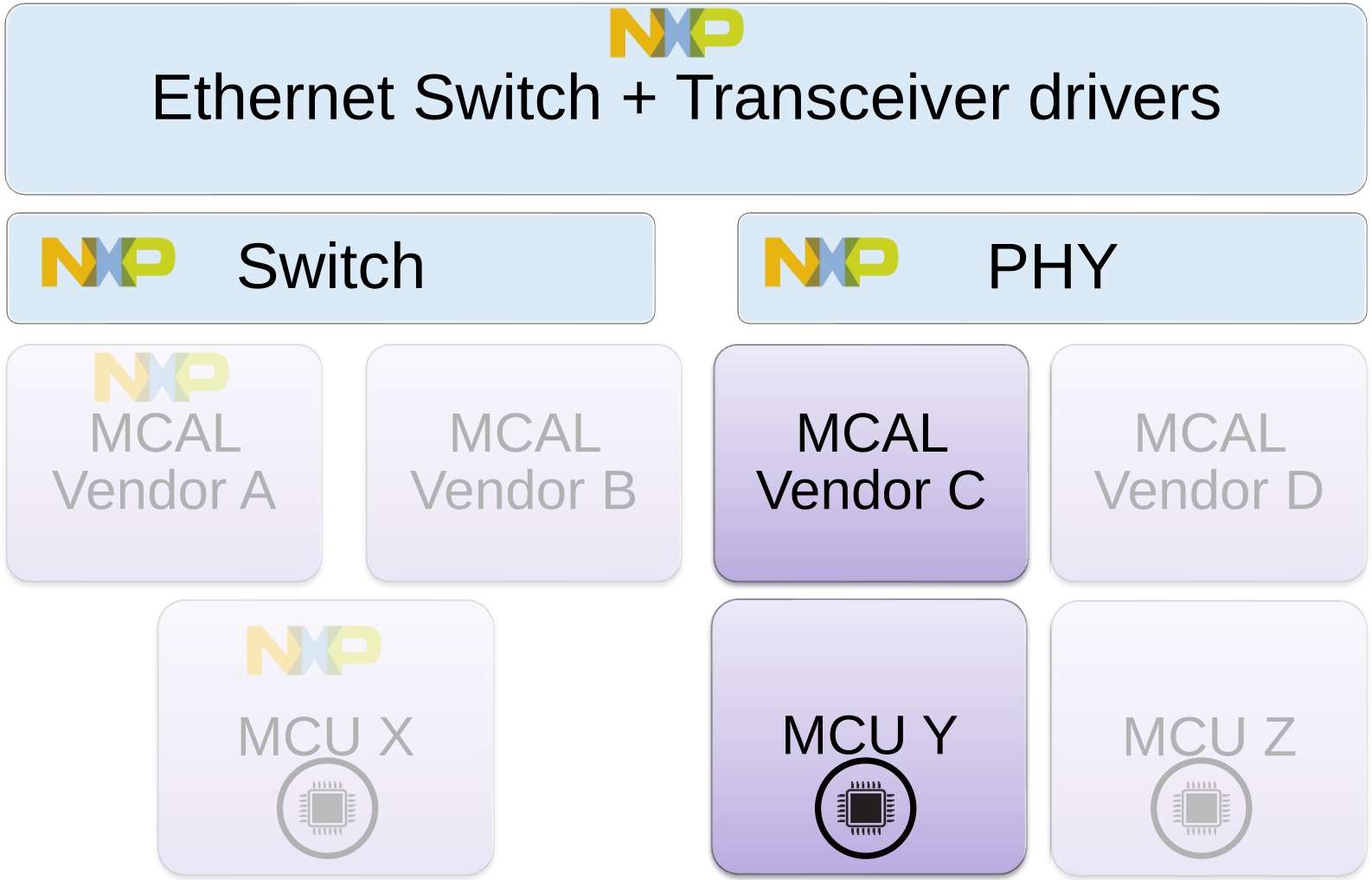
AUTOSAR Ethernet Drivers – Integration Examples



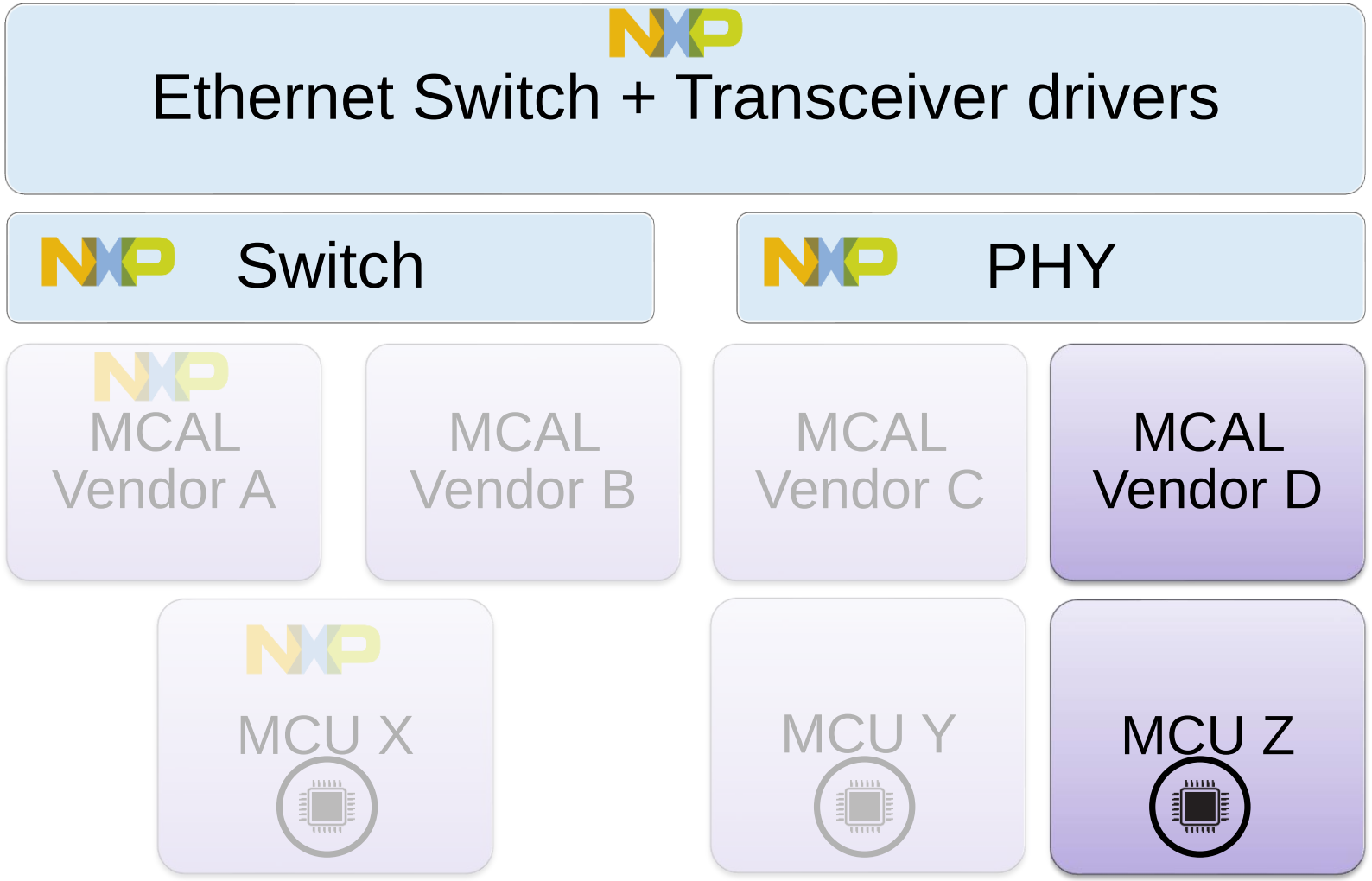
AUTOSAR Ethernet Drivers – Integration Examples



AUTOSAR Ethernet Drivers – Integration Examples



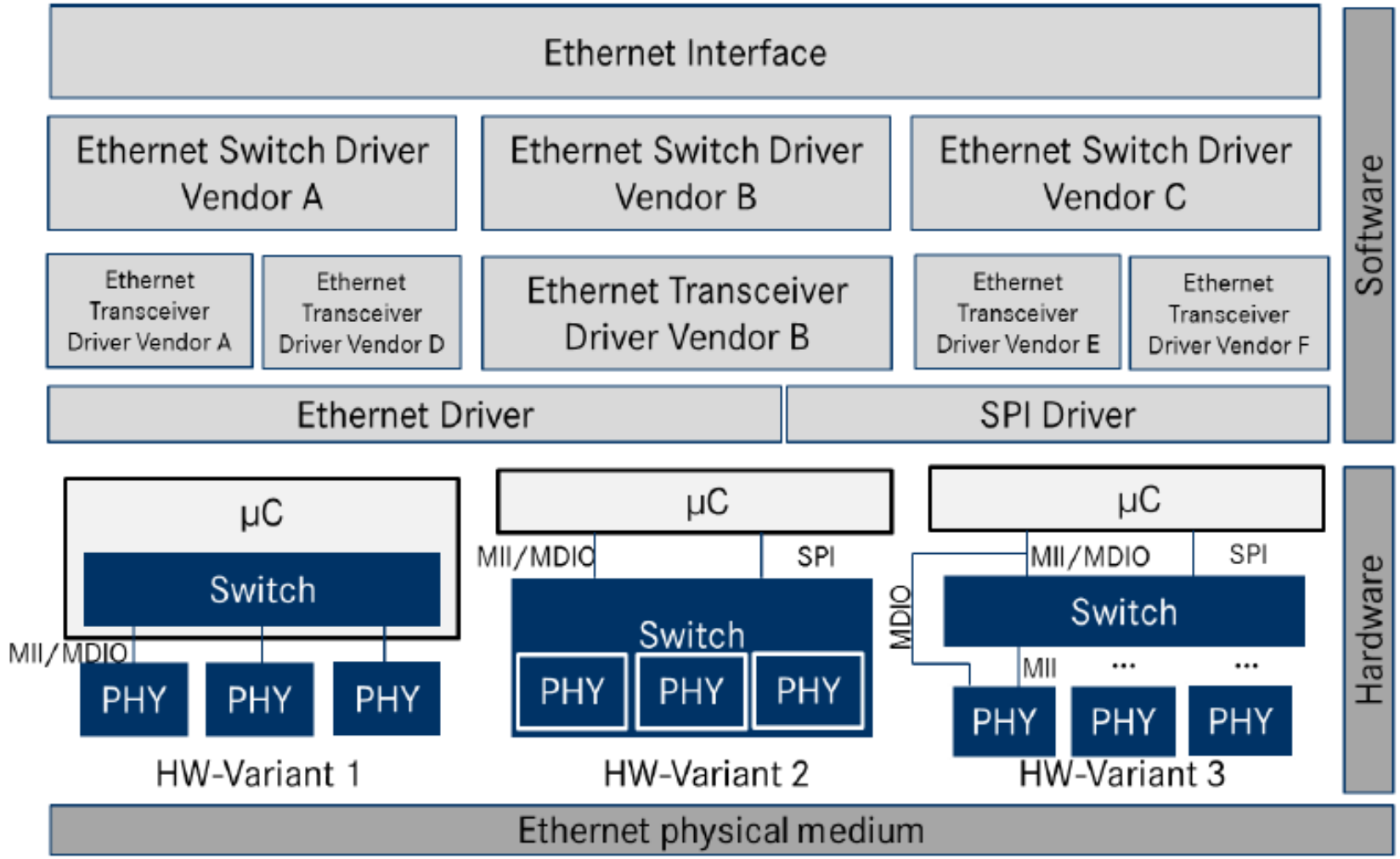
AUTOSAR Ethernet Drivers – Integration Examples



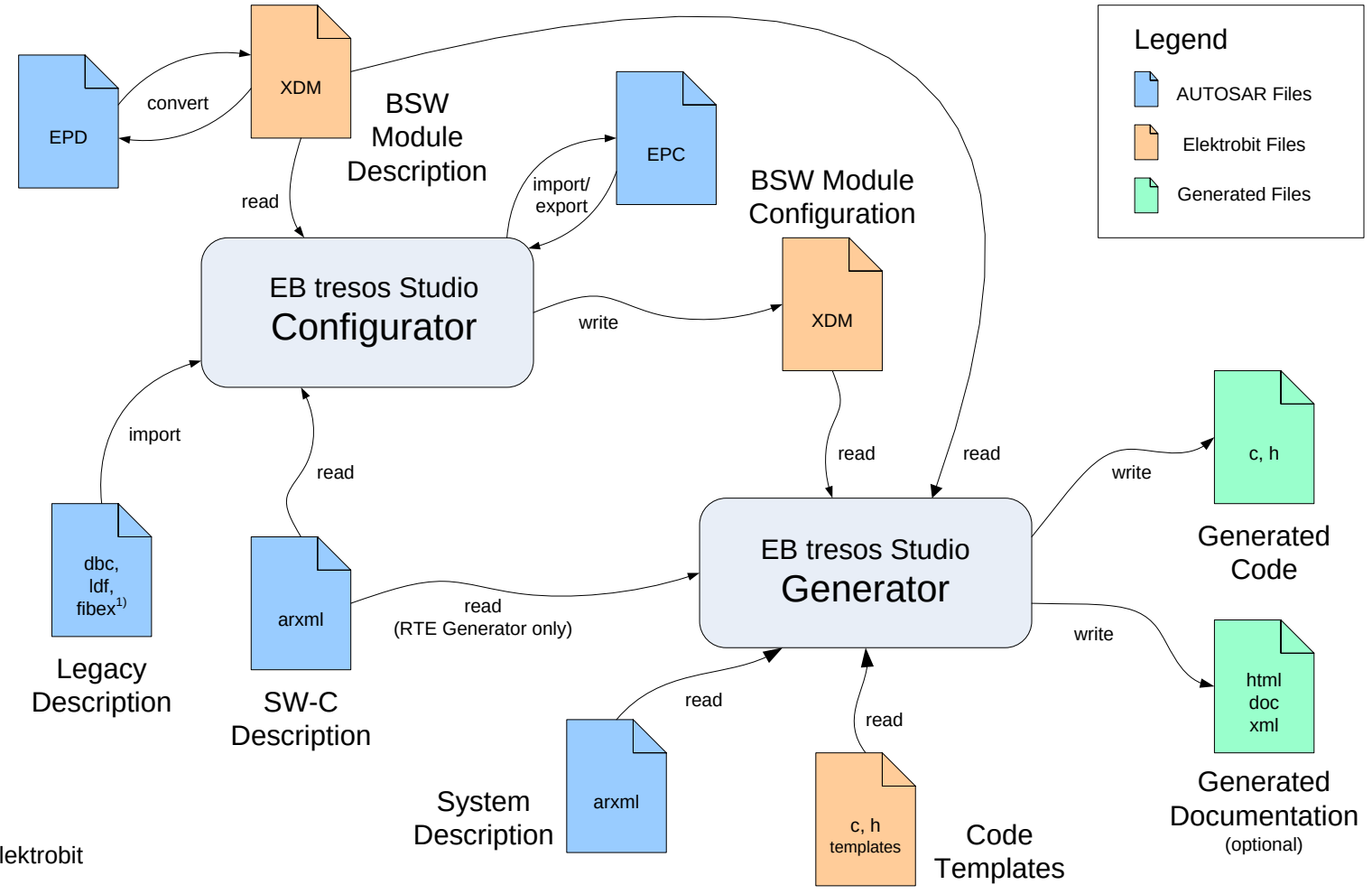
AUTOSAR Ethernet Drivers – Topologies

Various vendors

Different Topologies & Access types

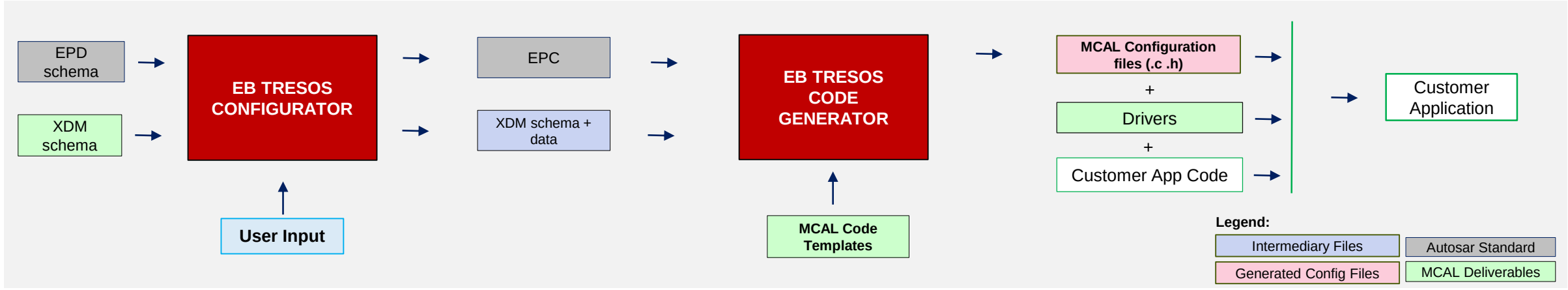


NXP MCAL Product – Configurator and generator



Source: Elektrobit

NXP AUTOSAR Solution



Autosar Configurator

- **EB Tresos**
- Vector DaVinci
- Mentor

Autosar Code Generator

- **EB Tresos**
- Customer specific generators for Autosar components

Customer specific makefiles or IDE projects

NXP AUTOSAR Release Content

Software Package

- Eclipse plugins
  - Driver source code
  - Configuration templates and scripts
  - Driver User Manual
  - Driver Integration Manual
- Sample Application

Quality Package*

- Test Specification
- Test Summary Report
- MISRA Summary Report
- Code Coverage Summary Report
- Traceability Matrix
- VSMD Report
- Code size, Stack size, RAM size Reports
- Static analysis Report

Safety Package**

- Driver FMEA
- Safety Manual

** RTM releases*
*** Safety RTM releases*

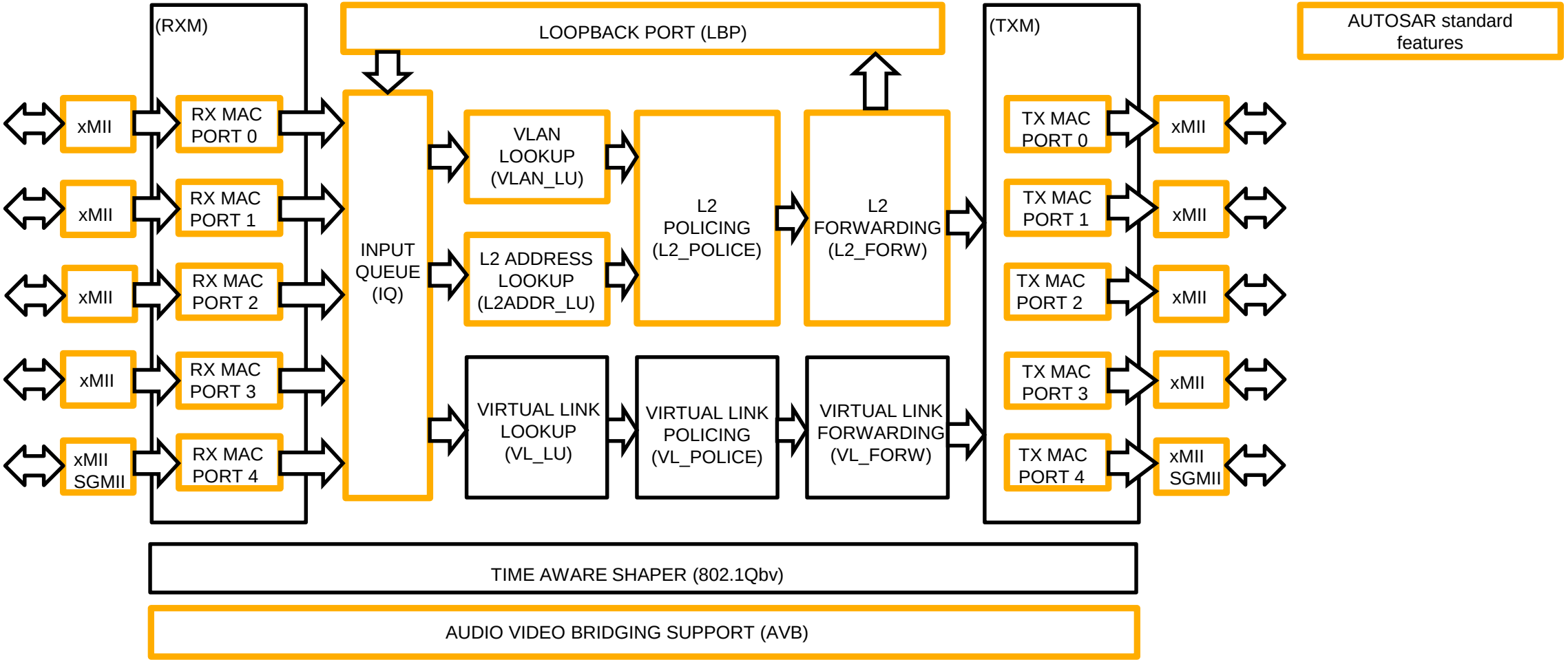
NXP AUTOSAR Ethernet Switch

NXP AUTOSAR Ethernet Switch Drivers

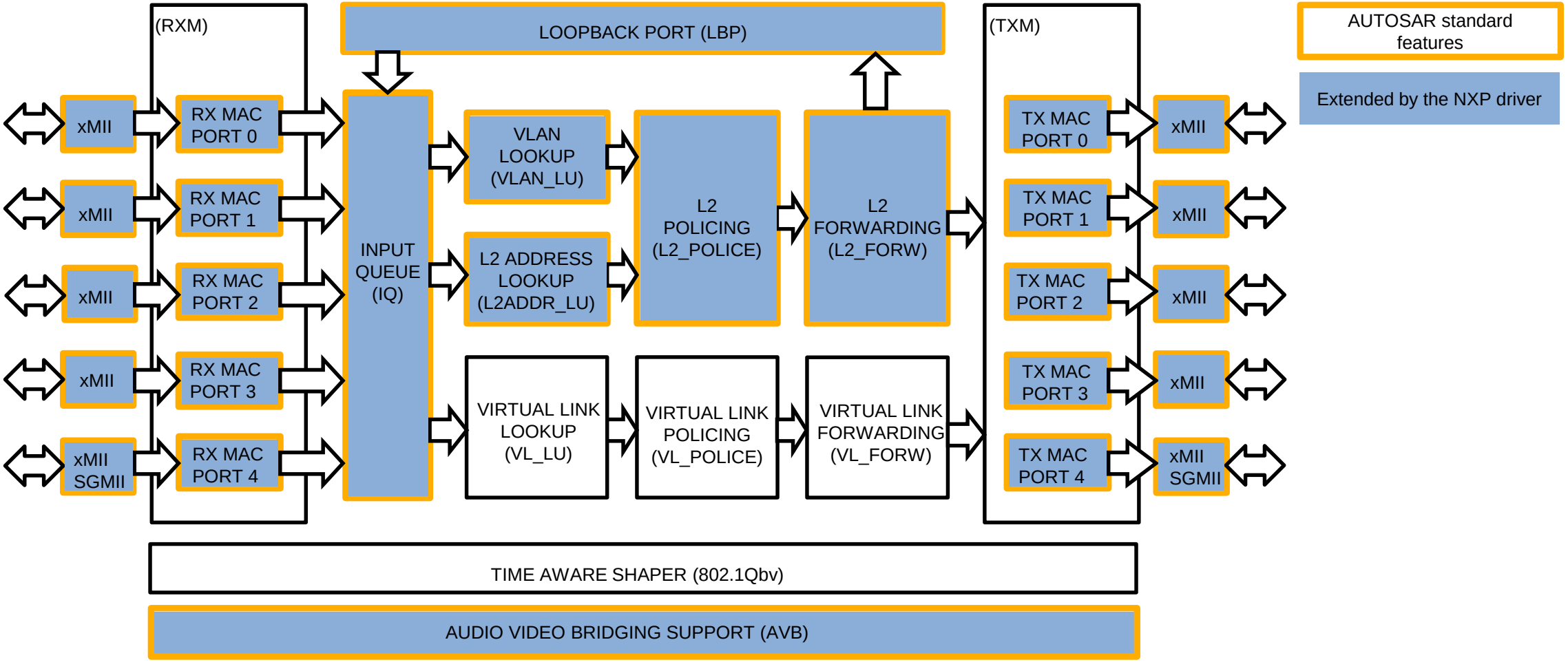
- SJA1105P/Q/R/S
  - Ready To Market release available
  - Autosar 4.3.1
  - Developed and tested using the MPC5748XG NXP MCU and NXP AUTOSAR MCAL drivers
  - Production quality
  - Safety package also available
  - Integrated by our partners also with competitor MCU and MCAL

- Next generation switch
  - Early Access Release available
  - Autosar 4.4.0
  - Developed and tested using a next generation 16ffc NXP MCU and NXP AUTOSAR MCAL drivers

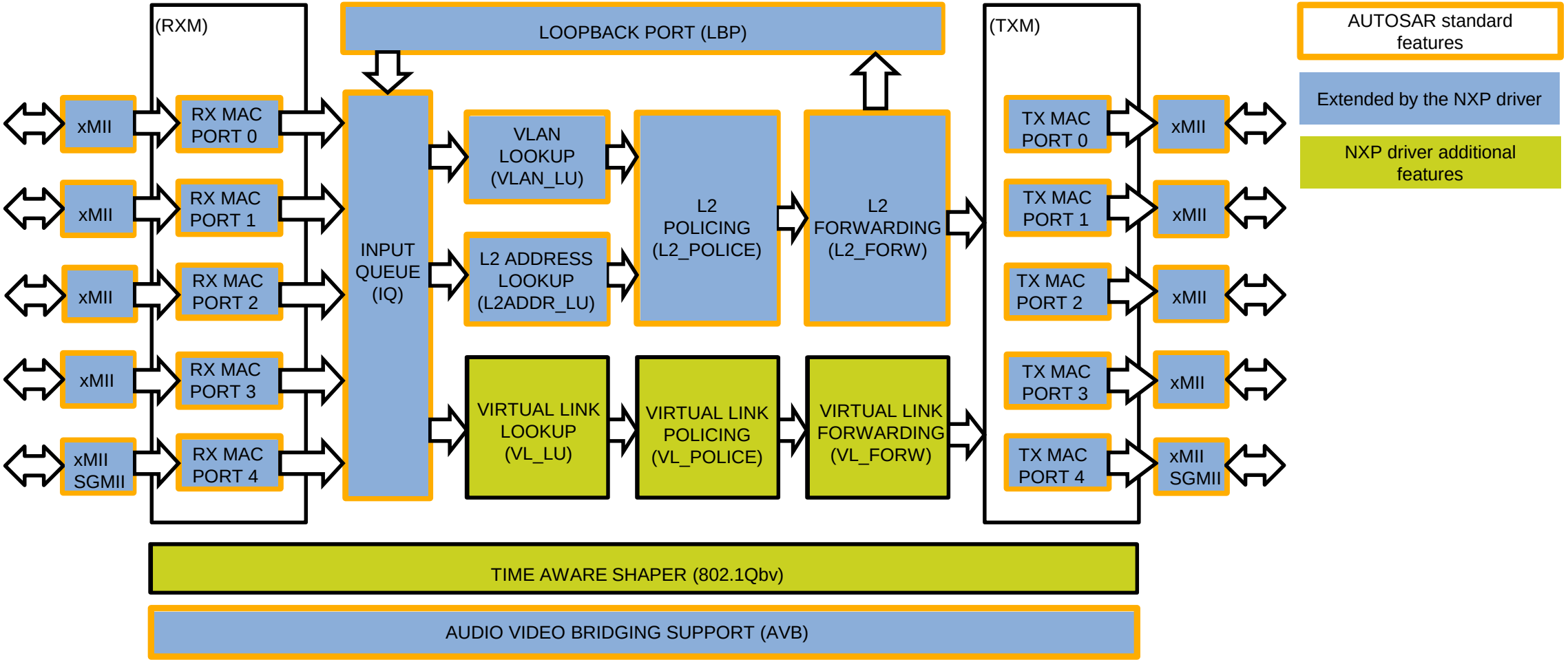
SJA1105P/Q/R/S Block Diagram



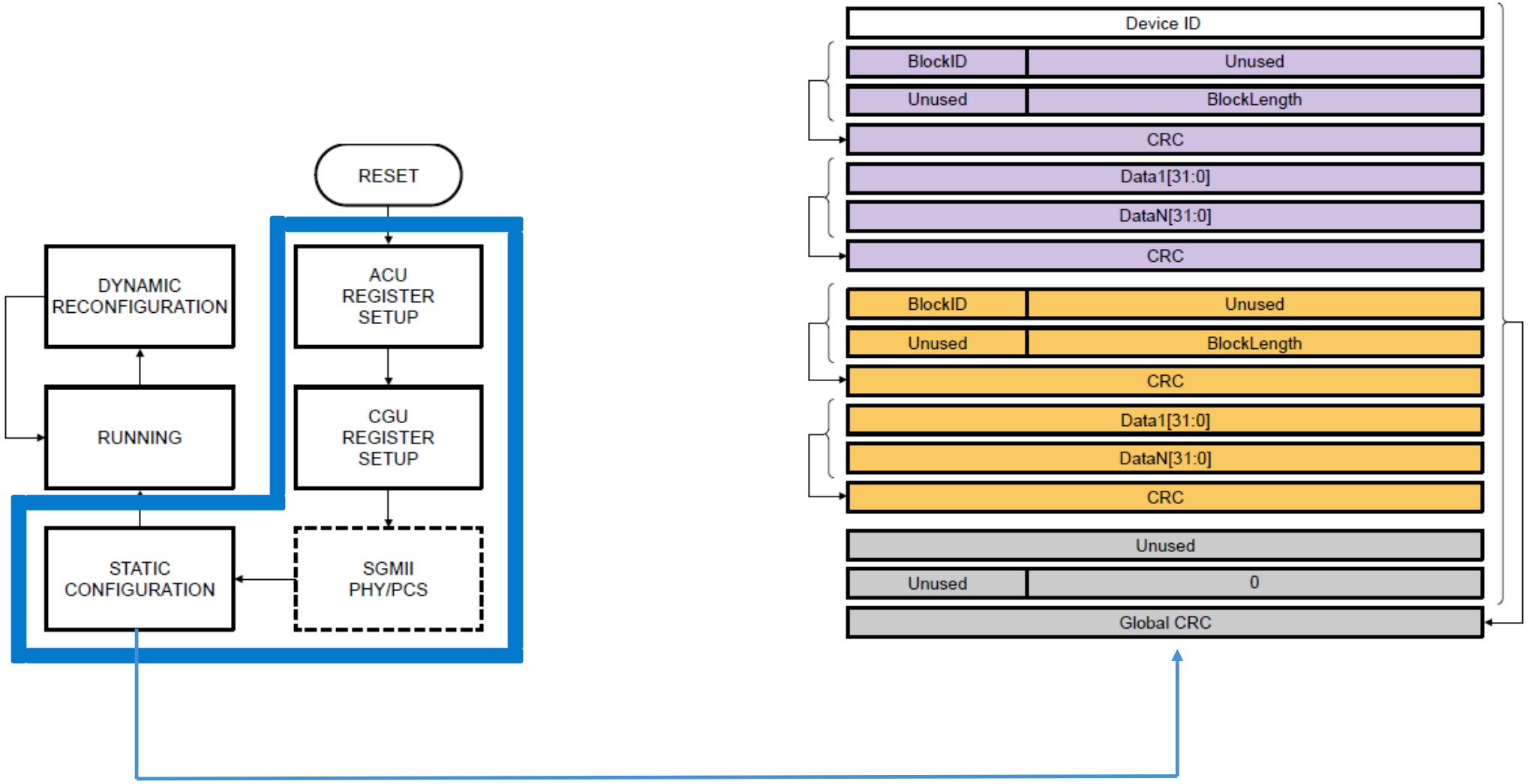
SJA1105P/Q/R/S Block Diagram



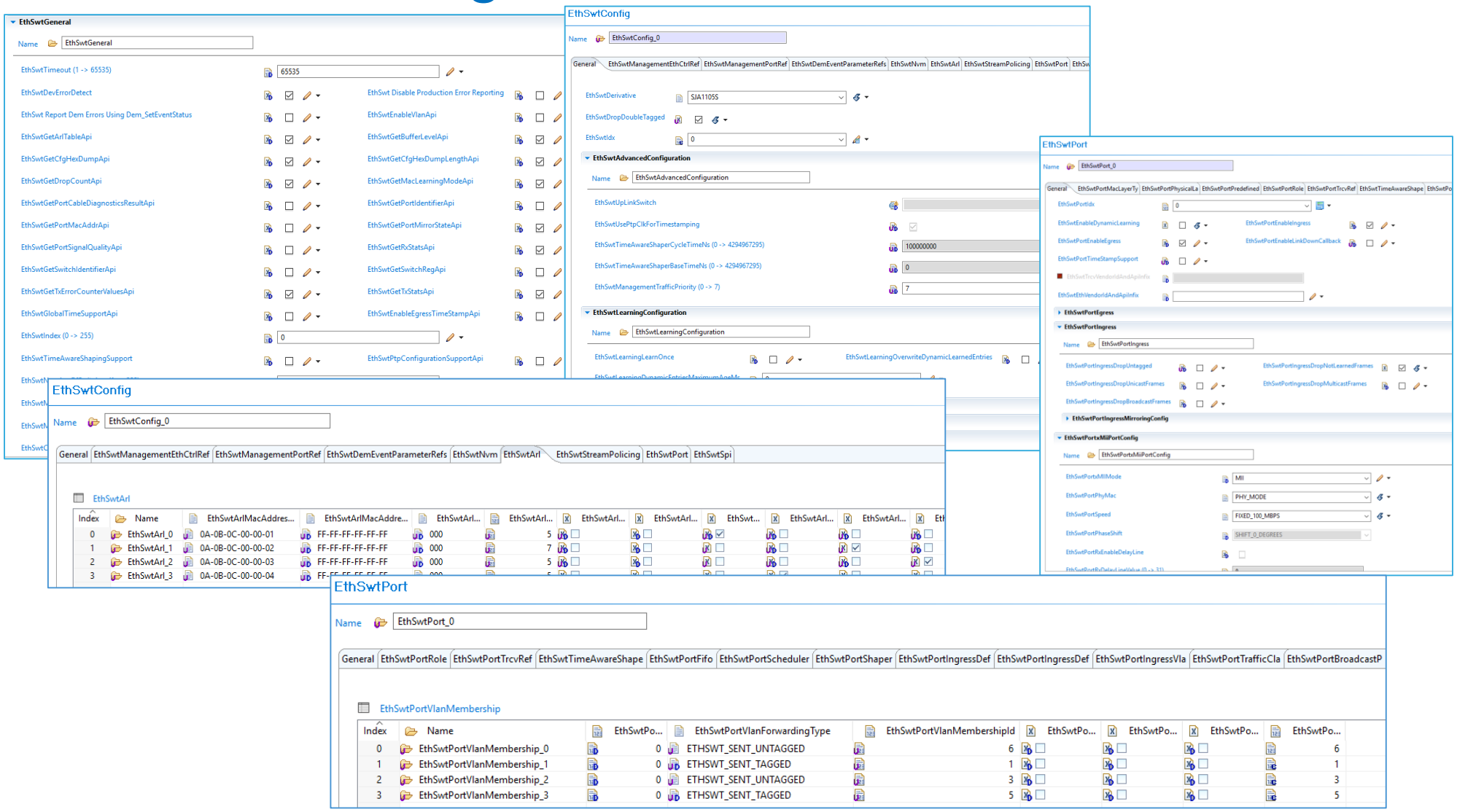
SJA1105P/Q/R/S Block Diagram



SJA1105P/Q/R/S Initialization

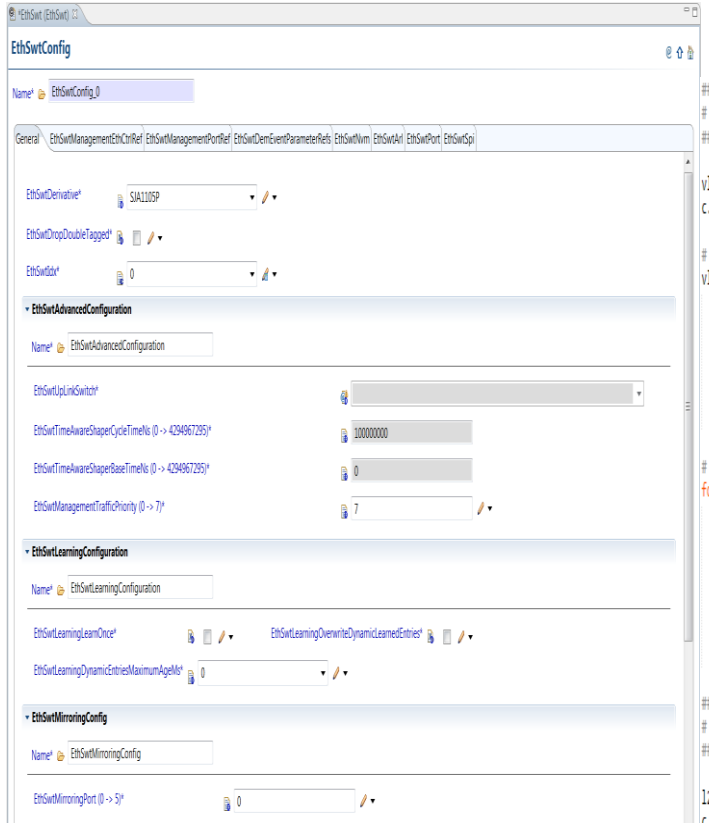


EthSwt Driver Configurator



Ethernet Switch Configuration Flow

Graphical UI/IDE → Scripting API → Config File → C code → Stream



Python package

```
##############################################################
# VLAN Lookup Table
##############################################################

vlan_lookup_table = make_table_by_layout(vlan_lookup_table_layout, layoutid_map)
c.append(vlan_lookup_table)

# Default VLAN
vlan_lookup_table.append({
    "VING_MIRR": 0,
    "VEGR_MIRR": 0,
    "VMEMB_PORT": 0x1F,  # All ports are member of the VLAN
    "VLAN_BC": 0x1F,  # Broadcast domain for the VLAN
    "TAG_PORT": 0x00,  # Egress frames are untagged
    "VLANID": default_vlan})

# Enable VLANs 0 to 15
for i in range(16):
    vlan_lookup_table.append({
        "VING_MIRR": 0,
        "VEGR_MIRR": 0,
        "VMEMB_PORT": 0x1F,  # all ports are member
        "VLAN_BC": 0x1F,  # Broadcast domain
        "TAG_PORT": 0x1F,  # Egress frames are tagged
        "VLANID": i})

##############################################################
# L2 Lookup Parameters Table
##############################################################

l2_lookup_parameters_table = make_table_by_layout(l2_lookup_parameters_table_layout, layoutid_
```

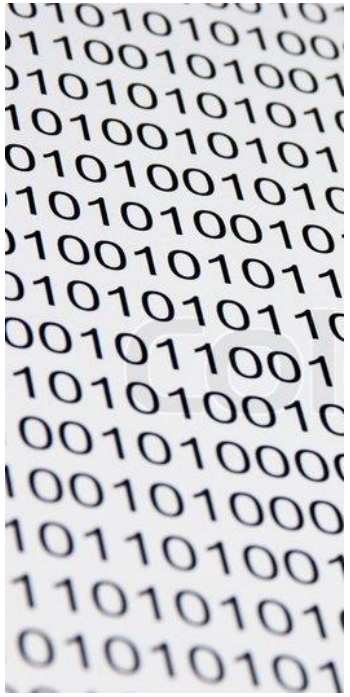
Intel HEX

```
:040000000E0300AE3D
:0400040000000006F2
:0400080050000000A4
:04000C006B256F21D0
:040010000000FBFEF3
:04001400FFFFFF03E8
:040018000000FBFEEB
:04001C00FFFFFF03E0
:040020000000FBFEE3
:04002400FFFFFF03D8
:040028000000FBFEDB
:04002C00FFFFFF03D0
:040030000000FBFED3
:04003400FFFFFF03C8
:040038000000FBFECB
:04003C00FFFFFF03C0
:040040000000FBFEC3
:04004400FFFFFF03B8
:040048000000FBFEBB
:04004C00FFFFFF03B0
```

Data array

```
uint32_t config_ptp_data[] =
{
    0xB700030EU, 0x06000000U,
    0x007FFF9FU, 0x4017D880U,
    0x4017D980U, 0x067FFF9FU,
    0x0A7FFF9FU, 0x4017DB00U,
    0x4017D800U, 0x107FFF9FU,
    0x147FFF9FU, 0x4017D980U,
    0x4017DA80U, 0x1A7FFF9FU,
    0x1E7FFF9FU, 0x4017D800U,
    0x4017D900U, 0x247FFF9FU,
    0x287FFF9FU, 0x4017DA80U,
    0x4017DB80U, 0x2E7FFF9FU,
    0x327FFF9FU, 0x4017D900U,
    0x4017DA00U, 0x387FFF9FU,
    0x3C7FFF9FU, 0x4017DB80U,
    0x4017D880U, 0x427FFF9FU,
    0x467FFF9FU, 0x4017DA00U,
    0x4017DB00U, 0x4C7FFF9FU,
    0x507FFF9FU, 0x4017D880U,
    0x4017D980U, 0x567FFF9FU,
    0x5A7FFF9FU, 0x4017DB00U,
    0x4017D800U, 0x607FFF9FU,
    0x647FFF9FU, 0x4017D980U,
    0x4017DA80U, 0x6A7FFF9FU,
    0x6E7FFF9FU, 0x4017D800U,
    0x4017D900U, 0x747FFF9FU,
```

SPI/AHB config stream



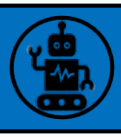
AUTOSAR Ethernet Switch Driver Features

- Enabling/disabling functionalities
- References to SPI/Ethernet
- Advanced learning settings
- Extended ARL entries
- Per stream filtering and policing
- Configuration of ports
  - Interface type, speed
  - Credit Based Shapers
  - Policers
  - VLAN membership
  - Predefined MAC addresses
  - Retagging options
  - Extended VLAN configuration
  - Extended forwarding configuration
  - Time Aware Shaping (802.1Qbv)

Configuration

- Configuration
- Direct register access
- Learning
- Mirroring
- Diagnostics and debugging
- PHY handling functions
- Management and time stamping
- Adding/subtracting offsets and setting rate corrections to the clock (gPTP)
- Dynamically read, update and add ARL entries
- Dynamically read and update forwarding options
- Start/Stop Time Aware Shaper (802.1Qbv)

Functionality



Standard AUTOSAR features

NXP extensions

NXP AUTOSAR Ethernet Transceiver

NXP AUTOSAR Ethernet Transceiver Driver

- TJA110X
  - Ready To Market release available
  - Autosar 4.3.1
  - Developed and tested using the MPC5748XG NXP MCU and NXP AUTOSAR MCAL drivers
  - Same driver for TJA1101, TJA1102, TJA1102S
  - Production quality
  - Safety package also available

- Next generation 100BASE-T1
  - Early Access Release available
  - Autosar 4.4.0
  - Developed and tested using a next generation 16ffc NXP MCU and NXP AUTOSAR MCAL drivers

AUTOSAR Ethernet Transceiver Driver Features

- Enabling/disabling functionalities
- References to Ethernet/Icu
- Configuration of transceiver hardware
  - Derivative selection
  - Interface type
  - Master/Slave selection
  - Speed
  - Address
  - Wakeup callout
  - Wakeup events mapping to EcuM
  - Advanced clock settings
  - Advanced pin settings
  - Advanced sleep/wakeup configuration and wake-up forwarding settings

Configuration

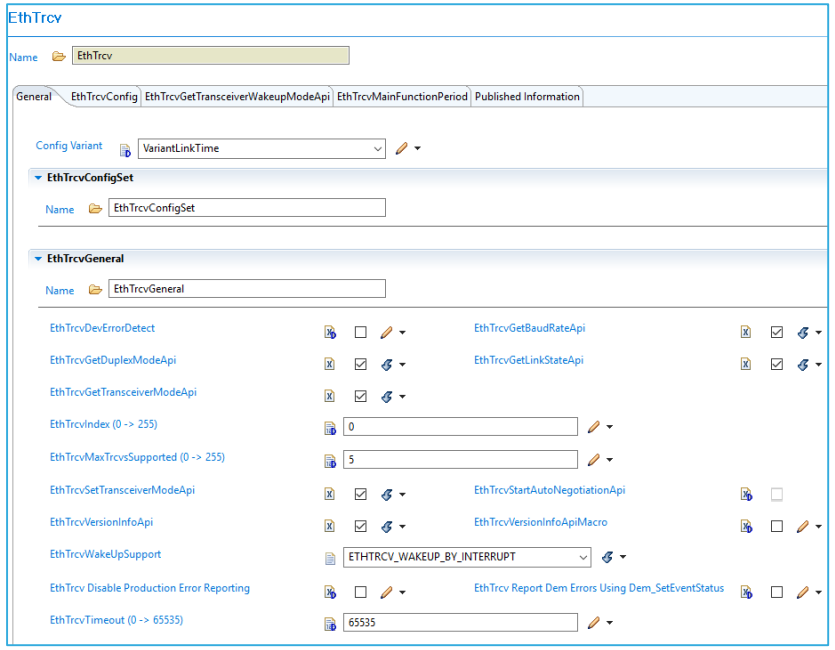
- Configuration
  - Loopback modes selection
  - Test modes selection
  - TX modes selection
- Status and configuration checks
  - Link state check
  - Baud rate/duplex mode check
- Diagnostics
  - Cable diagnostics
  - Signal quality
- Power management (with and without sleep/wakeup capability)
- Wake-up handling
- Multiple PHY support in the same driver

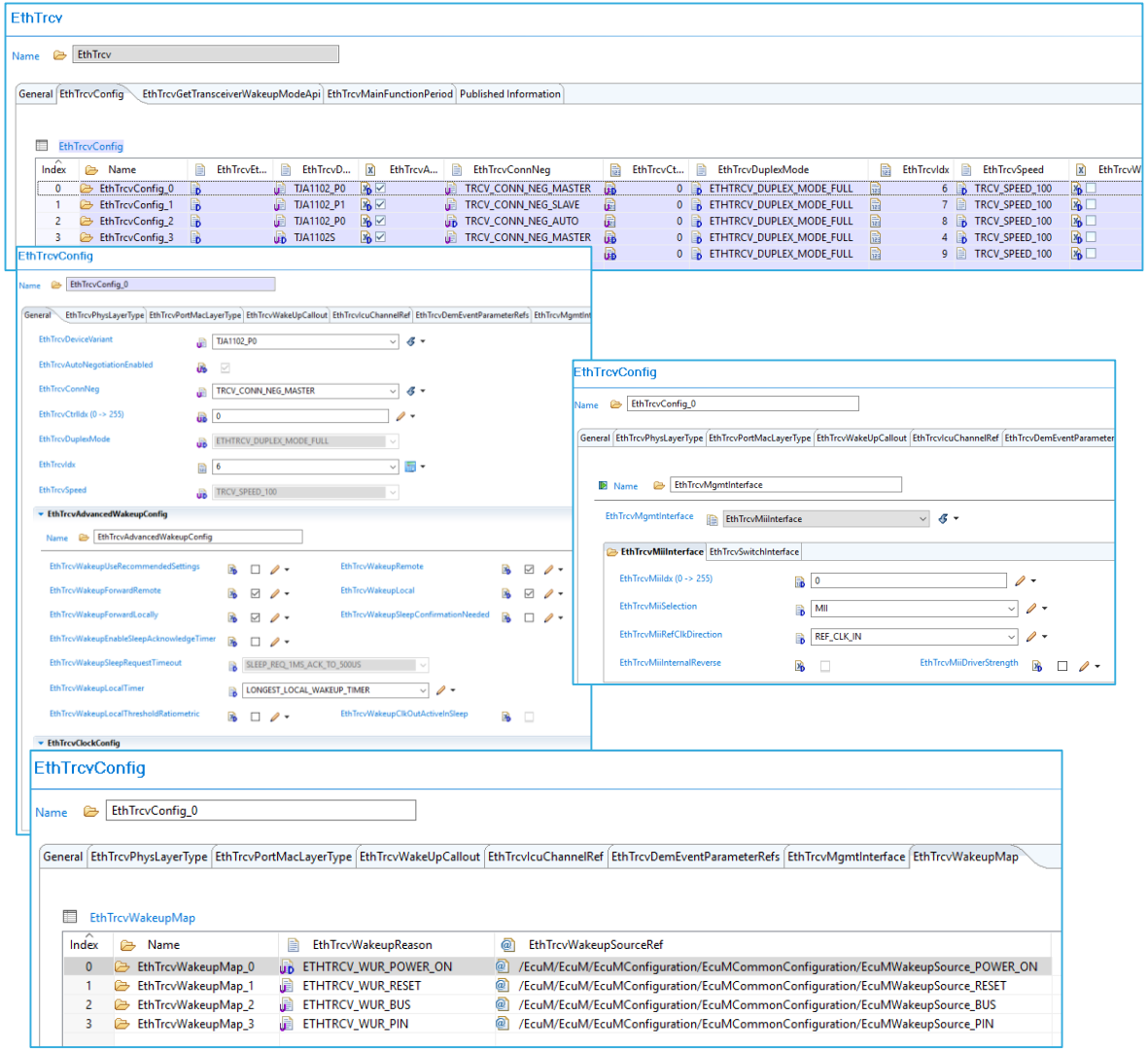
Functionality

Standard AUTOSAR features

NXP extensions

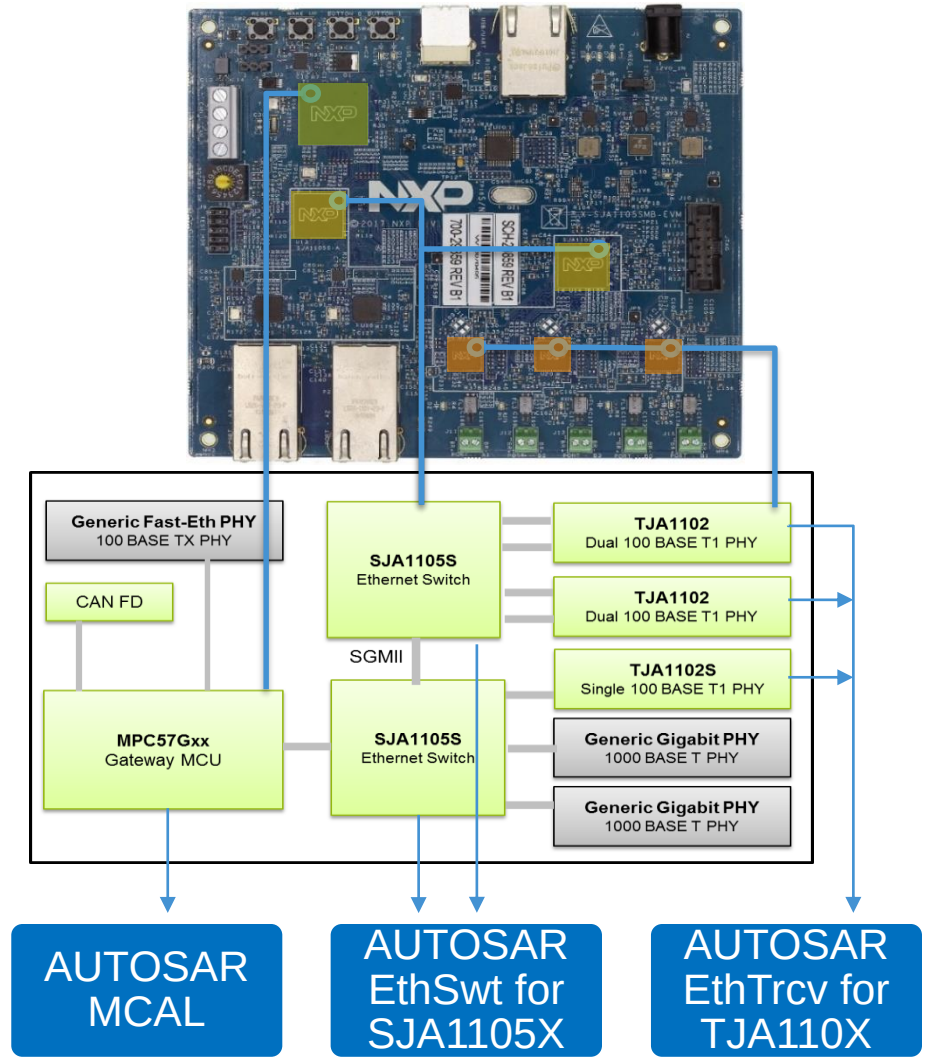
EthTrcv Driver Configurator





Sample Application

Sample Application on SJA1105SEVM



Conclusions

Conclusions

- Complete solution from NXP (MCU + Switch + PHYs + software)
- Extended support for hardware features
- Easy to integrate in an AUTOSAR stack
- Can be integrated with any MCU and/or MCAL
- Production quality software
- Can be integrated in safety products
- Sample application to ensure out of the box experience
- Support from NXP for both software and hardware

Questions & Answers

SECURE CONNECTIONS
FOR A SMARTER WORLD

NXP and the NXP logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners. © 2019 NXP B.V.