Product Information
MICROSAR
Table of Contents

1  MICROSA R - The Vector Solution for AUTOSAR ECU Software ................................................................. 3
2  MICROSA R.OS – The Real-Time Operating System for the AUTOSAR Standard ........................................... 9
3  MICROSA R.COM – AUTOSAR Basic Software Modules for Communication .................................................. 12
4  MICROSA R.CAN – AUTOSAR Basic Software Modules for CAN Communication ........................................... 15
5  MICROSA R.FR – AUTOSAR Basic Software Modules for FlexRay Communication ....................................... 18
6  MICROSA R.LIN – AUTOSAR Basic Software Modules for LIN Communication ............................................. 21
7  MICROSA R.ETH – AUTOSAR Basic Software Modules for Ethernet-Based Communication .......................... 24
8  MICROSA R.V2G – Basic Software Modules for Communicating with External Infrastructure .......................... 28
9  MICROSA R.AVB – Basic Software Modules for Audio/Video Communication via Ethernet ............................... 31
10 MICROSA R.MEM – AUTOSAR Basic Software Modules for Memory Management ......................................... 34
11 MICROSA R.SYS – System-related Basic Software Modules for AUTOSAR .................................................... 37
12 MICROSA R.DIAG – AUTOSAR-compatible implementation of the UDS protocol ......................................... 41
13 MICROSA R.MCAL – AUTOSAR Driver for Control of Microcontroller Peripherals ......................................... 46
14 MICROSA R.EXT – AUTOSAR drivers for control of external devices ............................................................. 50
15 MICROSA R.IO – AUTOSAR Input-Output Hardware Abstraction ................................................................. 52
16 MICROSA R.RTE - The optimized Run-time Environment for Software Components per the AUTOSAR Standard ...... 55
17 MICROSA R.AMD – AUTOSAR monitoring and debugging ............................................................................. 58
18 MICROSA R.SAFE – Functional Safety according to ISO 26262 up to ASIL D for ECU software ...................... 62
19 MICROSA R.Security – Access Security for AUTOSAR ECUs ........................................................................ 66
20 MICROSA R.Multi-Core – The AUTOSAR Solution for Multi-Core Processors ............................................... 69
21 MICROSA R.Variant Handling – Solutions for flexible configurations in AUTOSAR .................................... 72
22 MICROSA R.J1939 – AUTOSAR Basic Software Modules Specially Designed for Heavy-Duty Vehicles .......... 75
23 MICROSA R.VIT – Virtual Integration with vVIRTUALtarget Basic ............................................................... 77
24 MICROSA R.SIP and MICROSA R.EIP – A Quick Start to Your AUTOSAR Project ........................................... 81
25 AUTOSAR Evaluation Bundle – The complete package for the evaluation of AUTOSAR basic software and tools .................................................. 85
26 Additional Information .................................................................................................................................. 89

Please consider the environment before printing this document.

V1.6 December 2015
MICROSAR – The Vector Solution for AUTOSAR ECU Software

MICROSAR is the AUTOSAR solution from Vector for your ECU software. MICROSAR consists of the MICROSAR.RTE and MICROSAR basic software modules (BSW), which cover all aspects of the AUTOSAR standard and many extensions. Each AUTOSAR BSW module is assigned to a MICROSAR package. For detailed descriptions of individual packages and the MICROSAR.RTE, please refer to the following chapters. Vector combines and releases the BSW modules needed in individual “software integration packages” (SIP).

1.1 List of available MICROSAR Packages

Figure 1: The MICROSAR packages contain all modules of the AUTOSAR 4.2 standard

- **MICROSAR.AMD** – Monitoring and debugging of application and MICROSAR BSW
- **MICROSAR.AVB** – Basic software modules for Audio/Video communication via Ethernet
- **MICROSAR.MCAL** – Driver for driving the microcontroller periphery
- **MICROSAR.CAN** – Basic software modules for CAN communication
- **MICROSAR.COM** – Basic software modules for network-independent communication and gateways
- **MICROSAR.EXT** – Driver for driving external chips
- **MICROSAR.FR** – Basic software modules for FlexRay communication
- **MICROSAR.DIAG** – Basic software modules for diagnostics
- **MICROSAR.IO** – Interface between the microcontroller periphery and the application
- **MICROSAR.ETH** – Basic software modules for Ethernet-based communication
- **MICROSAR.LIBS** – AUTOSAR Libraries
- **MICROSAR.LIN** – Basic software modules for LIN communication
**MICROSAR.MEM** – Basic software modules for managing nonvolatile memory

**MICROSAR.OS** – Real-time operating system according to the AUTOSAR standard

**MICROSAR.RTE** – Optimized run-time environment for software components according to the AUTOSAR standard

**MICROSAR Safe** – Safety per ISO 26262 up to ASIL D for ECU software

**MICROSAR.SYS** – System-related basic software modules for AUTOSAR ECUs

**MICROSAR V2G** – Basic software modules for communication with external infrastructure

**MICROSAR XCP** – Measurement and calibration of an AUTOSAR-ECU using XCP incl. transport layer for Ethernet, FlexRay und CAN

### 1.2 Application Areas

The BSW modules of the MICROSAR packages assure basic functionality of the ECU. They contain the implementations of AUTOSAR standard services you will need for your functional software. You are able to develop your functional software platform independently, because the AUTOSAR architecture follows a consistent strategy of hardware abstraction.

Modules from the MICROSAR.OS and MICROSAR.MCAL packages are hardware-dependent. Vector offers these modules for a large number of different hardware platforms and compilers, e.g. to enable quick change-out of the controller device. The operating system MICROSAR.OS is available for single core and multi core-processors. Based on its ongoing contacts with OEMs, Vector is able to offer a number of OEM-specific BSW modules and extensions such as the diagnostic modules. You can configure all necessary MICROSAR BSW modules based on your project’s requirements, and after generation you can integrate them with the functional software. This produces a complete set of ECU software. If the functional software consists of AUTOSAR-conformant SWCs, you will need a run-time environment (RTE). The MICROSAR.RTE implements communication between the SWCs and their access to data and services from the BSW modules. Along with managing the entire flow of events and information, the MICROSAR.RTE also assures consistency in the exchange of information and coordinates accesses across core or memory protection boundaries.

ECU projects without SWC architecture (and therefore also without RTE) are optionally supported by the Vector BRE (Basic Runtime Environment). BRE simplifies the BSW integration by providing configurable BSW scheduling, critical section management and creation of type definitions for service layer BSW modules which normally would be created by the RTE. Hereby the BRE speeds up and simplifies building up AUTOSAR 4 based projects which do not have a RTE.

### 1.3 Properties

Development of the MICROSAR basic software modules is founded on the SPICE-based Vector development process for standard modules, which is based on SPICE. All MICROSAR packages offer the following features:

- Efficient memory utilization and short execution times
- Available for production use
- Available for AUTOSAR 4.x and 3.x
- Assistants and timely checks support you in achieving consistent configuration of your basic software
- Highly scalable, adaptable to your specific application
- Optimally integrated into your development process
- AUTOSAR Monitoring for testing and analysis of ECUs
- User-selectable configuration point (pre-compile, link-time or post-build)
- Supports multiple ECUs
- Optional delivery as source code
- Together with MICROSAR Safe: well-suited for safety-relevant functions (ISO 26262)
1.4 Production Use

The MICROSAR BSW modules are already being used in production projects. MICROSAR lets you benefit from Vector’s many years of experience in implementing embedded standard software. Before delivery, all MICROSAR Software modules undergo systematic integration tests for the specific conditions of your application (hardware platform, compiler, processor device, OEM, with/without RTE, etc.). Upon request, these tests may be extended to cover software modules from third-party producers (e.g. MCAL drivers).

1.5 Support of AUTOSAR 4.x and 3.x

Regardless of whether you use AUTOSAR 4.x or 3.x, from Vector you get the entire basic software from a single source. In migration of your projects, you benefit from the uniform development workflow for both AUTOSAR 4.x and 3.x:

- The configuration tools DaVinci Developer and DaVinci Configurator Pro are designed for both releases. This lets you avoid a change in tools.
- MCAL drivers from different AUTOSAR releases can be combined with MICROSAR.

In the case of a migration from AUTOSAR 3 to 4, we can support you in adapting your application software to the interfaces that were changed in the AUTOSAR 4.x standard.

Another advantage of MICROSAR lies in its many extensions in the BSW modules for AUTOSAR 3.x, which are specified in AUTOSAR 4.x. Some examples are the multi-core operating system, as well as support for J1939, XCP and Ethernet/IP, which are already available for AUTOSAR 3.x. The MICROSAR.MOST bundle is a Vector supplement to the standard and is compatible with AUTOSAR 4.x and AUTOSAR 3.x.

1.6 Consistent and Simple Configuration

With AUTOSAR, the work of manually developing or adapting the basic functionality of ECU software is replaced by configuration of the BSW modules. The intuitive, user-friendly and well-coordinated AUTOSAR tools from Vector (DaVinci) support the user in this process. Multi User Support of the DaVinci tools enables simultaneous work on a project by multiple users. The DaVinci tools require an “ECU Extract of System Description” file as input. It is also possible to produce a configuration based on commonly used network description files (DBC, FIBEX, LDF, etc.).

Early in the configuration process, all DaVinci tools check the validity of individual parameters, complex parameter groups and their interrelationships. In case of invalid configurations, the tools make recommendations for corrections, if possible. This extension of the AUTOSAR method simplifies integration of the basic software in your ECU and reduces integration time.

The DaVinci tools optimally assist you in configuring the RTE and the BSW modules. In a bottom-up process, for example, the SWC service ports (including runnables) are automatically generated to match the BSW configuration. This automation relieves you of tasks that recur frequently and are prone to errors when performed manually. This saves you time and costs.
Figure 2: You use DaVinci Configurator Pro to configure the BSW modules and the RTE.

Figure 3: You use DaVinci Developer to define the functional software (SWCs).

For more details on DaVinci tools from Vector, please refer to the relevant datasheets.

1.7 Scalability

In addition to fulfilling AUTOSAR requirements, the MICROSAR BSW modules also provide a number of functional extensions. The extended configuration options let you deactivate unnecessary functions to optimize the MICROSAR code for your application. This scalability makes the MICROSAR modules the optimal solution for both small and challenging applications. MICROSAR is already being implemented in a wide range of ECUs, such as steering angle sensors, door ECUs, engine ECUs, central gateways, etc. MICROSAR may also be used with other operating systems such as Linux or QNX.
1.8 AUTOSAR Monitoring and Debugging (AMD)

The AMD package simplifies testing and analysis of BSW and application functions by transferring important status information and results to tools such as CANape or CANoe.AMD. In addition, AMD offers the option of executing run-time measurements at the BSW and application functions. For more information, please refer to the section on MICROSAR AMD.

1.9 User-Selectable Time Point for BSW Configuration

The configuration point of all MICROSAR basic software modules is user-selectable. You can select the configuration point for each BSW module; choices are: pre-compile, link-time or post-build.

1.10 ECU Variants andReloading the BSW Configuration

To save on logistical costs in AUTOSAR ECUs, the MICROSAR modules are available with the Identity Manager option. This option can be used to store multiple configurations (e.g. left or right door) in one ECU. This lets you install an identical ECU for multiple functionalities within a model series or in different model series.

The BSW option "Post-Build Loadable" lets users modify many parameters of the BSW configuration at a later time without having to recompile the ECU software. For example, routing tables and send types can be modified and extended without the ECU’s build environment and without requiring that the supplier be contracted for a new ECU variant.

For more information, please refer to the section "MICROSAR Variant Handling".

1.11 Functional Safety according to ISO 26262: MICROSAR Safe

For use of the MICROSAR BSW in safety-related functions, Vector – together with TTTech Automotive – offers a complete solution for your AUTOSAR ECU. For more information, please refer to the chapter on MICROSAR Safe.

1.12 Scope of Delivery

The standard scope of delivery includes the following components:

- Software modules
- Command line based generator (for Windows XP/Windows 7)
- BSW Module Description
- Documentation

Additional or alternative components are listed below - separately for each module. For a comfortable configuration we recommend our DaVinci Configurator Pro. For details please see the separate Product Information.

1.13 Optional Delivery of Source Code

The MICROSAR modules are also available as source code, upon request. The source code lets you make pre-compile optimizations and simplifies testing.

1.14 License and Maintenance

Vector offers you flexible licensing – individually tailored to your requirements. In the framework of a maintenance contract, you get software updates that keep you current to the latest level of development.

1.15 Additional Services

- Consulting on system design
- Extension of MICROSAR BSW modules per customer wishes
- Development of customer-specific software components (SWC)
- Support in adapting existing functional software
Complete software integration in your ECU – including third-party software
Migration of existing software into an AUTOSAR-based concept
Hotline, special workshops and training courses on the topic of embedded software and AUTOSAR

1.16 The complete AUTOSAR solution from Vector
The Vector AUTOSAR solution consists of the DaVinci tools, the MICROSAR BSW and the MICROSAR.RTE. You will find information on general properties of the BSW modules from the MICROSAR packages in the following chapters. Details on functional features of the individual DaVinci Tools are found in the particular product information.

1.17 Contact and Availability
MICROSAR BSW modules are available for a large number of commonly used microcontrollers and in OEM-specific variants. You can obtain more information at www.microsar.com/availability/ or upon request.
Please find your contact person at www.vector.com/contact.

1.18 Training
In the framework of our training program, we can offer various training events and workshops for MICROSAR in our classrooms as well as at your business site.
For more information on individual training events and dates please visit: www.vector-academy.com
2 MICROSAR.OS – The Real-Time Operating System for the AUTOSAR Standard

MICROSAR.OS is a pre-emptive real-time multitasking operating system with optimized properties for use on microcontrollers. Vector’s many years of experience in developing operating systems and drivers for microcontrollers are bundled into this small, robust operating system core.

Figure 4: MICROSAR.OS module according to AUTOSAR 4.2

2.1 Overview of Advantages

- Small, quick, resource-economizing operating system and with short boot times
- MICROSR.OS is available for AUTOSAR 4.x and 3.x
- Optional: Available as multi core operating system
- Optional: Safe context switching per ISO26262 / ASIL-D
- Quality process per SPICE Level 3
- Graphic configuration tool for easy configuration
- Available for many 8, 16, 32 and 64 bit microcontrollers

2.2 Properties

MICROSAR.OS is based on the AUTOSAR OS specification, an extension of the practice-proven OSEK/VDX-OS operating system standard. This standard was extended to include functions for time monitoring and memory protection.

MICROSAR.OS is fully conformant to the AUTOSAR OS specification and supports all scalability classes:

- **SC1**: Real-time operating system implemented per the OSEK/VDX-OS standard and extended to include schedule tables
- **SC2**: Real-time operating system with time synchronization and monitoring of the time behavior of individual tasks and interrupt service routines
SC3: Real-time operating system with memory protection mechanisms on microcontrollers with suitable hardware support

SC4: Combines scalability classes SC2 and SC3

2.3 Optional Extensions

2.3.1 Synchronization with the global system time
Schedule tables can be synchronized with the global system time, e.g. the time transmitted over the FlexRay bus. This makes synchronized and simultaneous execution of tasks possible in a distributed system.

2.3.2 Memory protection (SC3, SC4)
Memory protection assures that application components do not mutually destroy data. This makes the integration of applications easier and more reliable.

2.3.3 Timing Protection (SC2, SC4)
Timing protection ensures that assumptions made in the early design phase related to execution times are preserved during run-time as well. As a result, a defective application section cannot impair the execution times of other running processes.

2.3.4 Execution time measurements (SC2, SC4)
You can use functions of scalability classes 2 and 4 to measure the execution times and interrupt disable times of applications. These measured data can later be used as practice-based values in designing and integrating future applications.

2.3.5 High Resolution Timer
The high resolution timer mechanism offers time resolutions of less than 1ms without increasing the interrupt load. Depending on the controller, this may enable resolutions into the microseconds range.

2.4 Operating system for applications according to ISO 26262
For safety-relevant applications per ISO 26262, you can obtain Vector’s operating system variant MICROSAR.OS SafeContext that was developed according to ASIL D. It is based on the AUTOSAR scalability classes SC3 and SC4 and is responsible for memory protection and safe context switching. To preserve “freedom from interference” with regard to memory protection, you need a suitable processor, e.g. with a Memory Protection Unit (MPU).

You can use MICROSAR.OS SafeContext to use safety-relevant application components together with standard components on the same CPU.

Scope of delivery for MICROSAR.OS SafeContext:

- Operating system core as source code
- Graphic configuration tool for Windows 7 / Windows XP
- Command line based generator
- BSW Module Description
- Description files for DaVinci Configurator Pro
- Documentation
- Read-back tool
- Safety Manual

For more details on the Vector solution for safety, please refer to the separate chapter about MICROSAR Safe.
2.5 Multi-core Operating System

MICROSAR.OS Multi-core is an advanced development of the proven real-time operating system MICROSAR.OS from Vector. You can use it wherever a multi-core system is to be developed according to the AUTOSAR specification. MICROSAR.OS Multi-core is based on AUTOSAR specification 4.x, but it can also be used in AUTOSAR 3.x projects.

2.5.1 Functions of the Multi-core Operating System

The multi-core operating system enables parallel and independent operation of multiple processor cores, each with a separate instance of the AUTOSAR operating system. Thereby, its configuration and system services match those of the single-core operating system. The extensions SC2 to SC4 and High Resolution Timing are available. In addition, MICROSAR.OS Multi-core offers mechanisms for coordinating and synchronizing tasks that run on different cores.

2.5.1.1 Synchronized Start-Up

The operating system assures that all cores are started up and initialized before the associated application is started.

2.5.1.2 Inter-Core Coordination

Processes on different cores are synchronized by task activations, setting of events, starting and stopping of alarms or ScheduleTables.

2.5.1.3 Shared Resource Access

If different cores should access commonly used resources, the operating system provides a coordination mechanism known as Spinlocks.

2.5.1.4 Inter-Core Communication

For consistent data exchange between two cores, the operating system provides an efficient interface in the form of the Inter-OS Application Communicator (IOC).

2.6 Graphical configuration and generation tool

For a comfortable configuration we recommend the DaVinci Configurator Pro. It contains consistency checks and the call of the generator. The generator is implemented as a command-line tool to enable its integration in an automated development environment.

2.7 Scope of Delivery

The MICROSAR.OS product is supplied with the following components:

- Operating system core as source code
- Graphic configuration and generation tool for Windows XP/Windows 7
- DaVinci Configurator Base as basic editor
- Command line based generator
- BSW modules description
- Documentation
3 MICROSAR.COM – AUTOSAR Basic Software Modules for Communication

The basic software modules (BSW) of MICROSAR.COM include AUTOSAR services for ECU communication. These services can support any number of communication channels. They are bus-independent and are needed in every communication stack. Per AUTOSAR architecture, they handle control and full integration in the ECU software of bus-specific communication modules for CAN, CAN-FD, J1939, FR, LIN and ETH.

Figure 5: MICROSAR.COM modules according to AUTOSAR 4.2

3.1 Overview of Advantages

> Code and execution time optimized by application-specific configuration
> Available for AUTOSAR 4.x and 3.x
> Contains many useful extensions beyond the AUTOSAR standard, see chapter “functions”
> Extended support for NM coordinators, see chapter “functions”
> NM module: OSEK NM compatibility is configurable
> Supports simultaneous operation of AUTOSAR NM and OSEK NM in NM migration projects
> Very efficient signal access via function macros (for AUTOSAR 3)

3.2 Application Areas

MICROSAR.COM lets users develop their functional software by a fully bus-independent approach. All necessary tasks for transmitting messages and for cross-bus network management activities are handled by the configurable BSW modules COM, NM, PDUR and IPDUM from MICROSAR.COM.

For a gateway ECU, you do not need any additional software. The COM and PDUR BSW modules from MICROSAR.COM enable routing of signals and TP or application messages.
3.3 Functions

The BSW modules from MICROSAR.COM contain functions defined in AUTOSAR 4.x:

- Services of the COM module organize transmission of messages according to their send type (cyclic, event-triggered, etc.). A key task is to implement bus-independent signals of the functional software in PDUs.

- The PDU Router (PDUR) provides the modules COM, DCM and the complex drivers with an interface to the communication modules (interface, transport protocol and network management) of the different bus systems. This interface serves to transmit and receive data via PDUs. The PDUR also implements a gateway between the communication modules of the various bus systems. The MICROSAR module CDD allows TP- and IF-PDUs to be integrated into the COM stack:
  - above or below the PDU router
  - above the communication interfaces

- The Network Management Interface (NM) bundles inter-bus network management activities of all of the ECU’s communication channels. As NM coordinator, it synchronizes wake-up and sleep of the communication channels.

- The optional I-PDU Multiplexer (IPDUM) module supports multiple usage of frames with different data contents, via a static configuration for the classic bus systems or alternatively via dynamic data content mapping for CAN-FD.

- Transformer: Allows efficient transmission of complex data structures and big PDUs over the network.
  - COMXF: Allows efficient signal groups with many group signals. The placement is derived from the system extract.
  - SOMEIPXF: Provides a serialization strategy for various data types. LDCOM can be used here for a highly efficient transmission.
  - E2EXF: Enables end-to-end protection for network communication that is managed using the AUTOSAR transformer concept (i.e. serialization through COMXF or SOMEIPXF).

- SecOC: For details to the Secure OnBoard Communication (SecOC), please refer to the chapter "MICROSAR Security".

Beyond the AUTOSAR standard, MICROSAR.COM also contains the following important services:

- COM: Invalidity declaration of TX signals in case of RX signal timeout
- COM, PDUR & IPDUM: Pre-compile and post-build loadable optimizations such as finding and removing redundant data
- NM: Synchronous sleep and wake-up of multiple networks via different NM coordinators
- NM: Backup coordinator

The following functions are optionally available:

- COM: PDU Replication (for AUTOSAR 4.x)
- COM: The COM module can be supplied with gateway functionality. Routing is possible for signals and signal groups. Routing in COM is possible via a configuration description without existence of a real signal or signal group.
- NM: Support of OSEK NM (configurable)
- NM: Mixed operation of OSEK and AUTOSAR-NM on one channel
- PDUR: Support for
  - TP and message routing
  - routing via meta data in case of range routing
  - routing of variable addresses ("dynamic gateway")
  - Routing of dynamic PDU lengths
- Post-build loadable and Post-build selectable: For details, please refer to the section "MICROSAR Variant Handling".
3.4 Configuration

For convenient configuration we recommend our DaVinci Configurator Pro. For more details, see the separate datasheet.

Using DaVinci Configurator Pro, you can easily link the MICROsAR.COM module PDUR and the modules CANIF, LINIF, FRIF and ETHIF (SOAD) to your complex drivers by configuration.

Figure 6: Configuration of the communication modules with DaVinci Configurator Pro
4 MICROSAR.CAN – AUTOSAR Basic Software Modules for CAN Communication

The MICROSAR.CAN package contains the BSW modules defined in the AUTOSAR architecture for CAN communication: CANIF, CANNM, CANTP, CANSM and optional modules for J1939 and XCP.

4.1 Overview of Advantages

> Available for AUTOSAR 4.x and 3.x
> Contains numerous useful extensions
> Code and execution time are optimized based on need-specific configuration
> Inter-module configuration of all communication-specific software modules
> Fast wakeup handling at ECU startup
> CANTP: ISO 15765-2 compatibility is configurable
> OSEK NM is available as a configurable module, compatible with CANNM
> CANNM, CANSM: Control of the communication stack (on and off) depending on the partial network state
> CAN-FD: Support of up to 64 Byte data with enhanced bandwidth. Available for many CAN-FD controller.

4.2 Application Areas

MICROSAR.CAN is used to handle communication in CAN networks. It is also well-suited as a foundation for calibration with XCP, gateways or re-programming. You can also extend MICROSAR.CAN with the optional J1939TP package to enable operation of an AUTOSAR ECU in a J1939 net-work. The BAM and CMDT transport protocols are available for this.
4.3 Functions

The BSW modules in MICROSAR.CAN contain functions defined in AUTOSAR 4.x. Beyond the standard itself, MICROSAR.CAN offers the following important services:

- **CANIF**: Double hash search algorithm for efficient filtering the Rx messages
- **CANNM, CANTP**: Pre-compile optimizations, e.g. for single-channel systems
- **CANSM**: Support of ECU Passive Mode
- **CANTP**: Supports mixed addressing (11 bit CAN ID); typically for CAN/LIN gateway applications
- **CANTP**: Optimized routing (e.g. with Burst Transmission) together with the PDUR from MICORSAR.COM
- **CANTP**: ISO 15765-2 compatibility is configurable
- **CANTSYN**: Time Synchronization over CAN (CANTSYN) implements the Generalized Precision Time Protocol (gPTP) according to IEEE 802.1AS. This permits clock synchronization between CAN ECUs. The Synchronized Time Base Manager (STBM) BSW module from MICROSAR.SYS is available as a higher-level time coordinator.

The following functionalities are available as options:

- **J1939**: Network Manager J1939NM, Request Manager J1939RM, diagnostic module J1939DCM and in J1939TP the transport protocols BAM and CMDT for J1939 networks
- **CAN driver**: Increase the number of Full-CAN objects by combining multiple CAN controllers on one physical CAN bus (Common CAN)
- **For CANIF**: support of an external CAN controller
- **Post-build loadable and Post-build selectable**: For details, please refer to the section "MICROSAR Variant Handling".

4.4 Configuration

For convenient configuration, we recommend to use DaVinci Configurator Pro. For more details, please refer to the separate product information.

4.5 Other MICROSAR Products for a Complete CAN Communication Stack

Based on the AUTOSAR architecture, a complete communication stack for CAN can be formed using MICROSAR.CAN together with the BSW modules from the separately available MICROSAR.COM, MICROSAR.MCAL and MICROSAR.EXT packages. To interface MICROSAR.CAN to the application and the hardware, you will still need the following BSW modules:

- **Hardware-specific CAN driver (CANDRV)** from MICROSAR.MCAL
- **Hardware-specific transceiver control (CANTRCV)** from MICROSAR.EXT, also for partial networking
- **General communication modules (COM, NM, PDUR, IPDUM)** from MICROSAR.COM

Modules in MICROSAR.MCAL and MICROSAR.EXT are available for many microcontrollers and transceivers.

4.6 Other relevant MICROSAR Products for CAN

- **DCM and DEM** from MICROSAR.DIAG
- **DET, ECUM and COMM** from MICROSAR.SYS
- **MICROSAR XCP** lets you measure and calibrate per ASAM XCP. Here, the module was specifically optimized for use together with CANoe.XCP and CANoe.AMD as well as CANape. For CAN ECUs, MICROSAR XCP contains the suitable CANXCP transport layer.

Beyond the AUTOSAR standard, MICROSAR XCP supports generic readout of measuring objects. As a result, no addresses must be defined and updated in the a2l file. Data from any version or variant can be extracted with an a2l file independent from the MCU build. The generic readout feature requires usage of CANoe.AMD or CANape as XCP tool.
For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000If allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000If.

You will find more information on J1939 ECUs in commercial vehicles in the section "MICROSAR J1939"

4.7 The Vector toolchain for developing CAN ECUs

Figure 8: Vector offers a comprehensive range of products and services for your CAN projects
Vector offers you MICROSAR.FR, an AUTOSAR-conformant package for FlexRay communication. It contains the following BSW modules defined in the AUTOSAR architecture: FRIF, FRNM, FRSM and a choice between FRTP and FRISOTP. MICROSAR.FR can be extended with XCP as an option.

Figure 9: MICROSAR FlexRay modules according to AUTOSAR 4.2

5.1 Overview of Advantages

> Available for AUTOSAR 4.x and 3.x
> Activates/deactivates partial networks and provides data, depending on the partial network state
> Small code size and short execution times due to optimized administration of job lists of the FlexRay interfaces
> Either the FRTP (AUTOSAR) or FRISOTP (ISO 10681) transport protocol may be used
> Support of ECU passive mode in the FlexRay State Manager
> Early detection of synchronization losses

5.2 Application Areas

MICROSAR.FR is used to handle communication in FlexRay networks including partial networking. Furthermore, it is ideal as a foundation for calibration with XCP, gateways or flashing.

5.3 Functions

The BSW modules in MICROSAR.FR contain functions defined in AUTOSAR 4.x, where FRISOTP is a supplement to AUTOSAR 3.x. MICROSAR.FR contains the following important services that go beyond the standard:
> FRDRV: Optimized wakeup during operation (WUDOP)
> FRDRV & FRIF: Support of the following APIs: CancelTransmit and L-PdU reconfiguration
> FRIF: Dual channel redundancy for redundant transmission of frames and PDU-specific voting function for the SWCs
> FRDRV, FRIF, FRNM, FRTP: Pre-compile optimizations, e.g. for single-channel systems
> FRSM: Support of ECU Passive Mode, immediate startup after passive wakeup, extended error handling by State Change Notification, configurable time delay for FlexRay startup at passive wakeup as well as configurable number of wakeup patterns.
> FRTSYN: Time Synchronization over FlexRay (FRTSYN) implements the Generalized Precision Time Protocol (gPTP) according to IEEE 802.1AS. This permits clock synchronization between FR ECUs. The Synchronized Time Base Manager (STBM) BSW module from MICROSAR.SYS is available as a higher-level time coordinator.

The following functionalities are available as options:
> Post-build loadable and Post-build selectable: For details, please refer to the section "MICROSAR Variant Handling".

5.4 Operating System

FlexRay basic software modules can be used entirely without an operating system. However, it makes sense to use an AUTOSAR OS or a conventional OSEK-OS (e.g. Vector osCAN). Ideally suited for FlexRay applications is MICROSAR.OS from Vector.

5.5 Configuration

For convenient configuration we recommend the DaVinci Configurator Pro. For more details, please refer to the separate product information.

5.6 Other MICROSAR Products for a complete FlexRay Communication Stack

Based on the AUTOSAR architecture, a complete communication stack for FlexRay can be made with MICROSAR.FR together with the BSW modules from the separately available MICROSAR.COM, MICROSAR.MCAL, MICROSAR.SYS and MICROSAR.EXT packages. You will also need the following BSW modules to interface MICROSAR.FR to the application and hardware:
> Hardware-specific FlexRay driver (FRDRV) from MICROSAR.MCAL
> Hardware-specific transceiver control (FRTRCV) from MICROSAR.EXT
> General communication modules (COM, NM, PDUR, IPDUM) from MICROSAR.COM

The modules in MICROSAR.MCAL and MICROSAR.EXT are available for many different microcontrollers and transceivers.

5.7 Other MICROSAR Products for FlexRay

> DCM and DEM from MICROSAR.DIAG
> DET, ECUM and COMM from MICROSAR.SYS
> MICROSAR XCP lets you measure and calibrate according to ASAM XCP. MICROSAR XCP was specially optimized for use together with CANoe.XCP and CANoe.AMD as well as CANape. For FlexRay ECUs, MICROSAR XCP contains the related FRXCP transport layer.

Beyond the AUTOSAR standard, MICROSAR XCP supports generic readout of measuring objects. As a result, no addresses must be defined and updated in the a2l file. Data from any version or variant can be extracted with an a2l file independent from the MCU build. The generic readout feature requires usage of CANoe.AMD or CANape as XCP tool.

> For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000If allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000If.
5.8 The Vector toolchain for developing FlexRay ECUs

Figure 10: Vector offers a comprehensive portfolio of products and services for your FlexRay projects
6 MICROSAR.LIN – AUTOSAR Basic Software Modules for LIN Communication

MICROSAR.LIN contains the BSW modules defined in the AUTOSAR architecture for LIN communication: LINIF, LINSM and LINNM. According to AUTOSAR, LINTP is a part of LINIF. The LIN transport protocol is offered as an option, because not every LIN communication stack requires a transport protocol. XCP for the MICROSAR.LIN Master is also available as an ASAM extension.

Figure 11: MICROSAR.LIN modules according to AUTOSAR 4.2

6.1 Overview of Advantages

> Available for AUTOSAR 4.x and 3.x
> Contains numerous useful extensions
> Minimized scheduling jitter for multi-channel master
> Optimized routing of diagnostic requests to LIN Slaves
> Quick start of the LIN channel
> Reliable switching of schedule tables possible
> Based on Vector’s many years of experience with production software for LIN

6.2 Application Areas

MICROSAR.LIN handles communication tasks for a LIN Master in a LIN network. In addition, it may be used as a foundation for gateways or re-programming.

6.3 Functions

The BSW modules from MICROSAR.LIN contain functions defined in AUTOSAR 4.x.
Beyond the standard, MICROSAR.LIN includes the following important services:

- LINIF: Configurable wakeup delay
- LINIF and LINTP: Separately configurable memory mapping of configuration data for LINIF and LINTP. This is especially appealing for controllers with segmented memory. The MICROSAR.LIN product includes two separate generators for LINIF and LINTP for this purpose.
- LINIF/LINSM: Notification of Schedule Table End
- LINIF: configurable schedule tables to reduce maximum task run-time in multi-channel systems
- LINIF: wakeup through LIN tranceiver. After an external wakeup, this function allows to omit a second (unwanted) wakeup pulse by the master.
- LINSM: Extended polling of LINSM sub-modes for controlled switchover of LIN Schedule Tables
- LINSM: Optimized startup behavior by automatic selection of a schedule table (configurable)

The following optional functionalities are also available:

- LINIF: LINTP implementation for segmented transmission, e.g. of UDS diagnostic communication
- LINTP: Extension of gateway functionality for precise switchover of schedule tables in diagnostic requests and responses
- Post-build loadable and Post-build selectable: For details, please refer to the section "MICROSAR Variant Handling".

6.4 Configuration

For convenient configuration, we recommend the DaVinci Configurator Pro. For more details, please refer to the separate product information.

6.5 Other MICROSAR Products for a Complete LIN Communication Stack

Based on the AUTOSAR architecture, a complete communication stack for LIN is formed by MICROSAR.LIN together with the BSW modules from the separately available MICROSAR.COM, MICROSAR.MCAL and MICROSAR.EXT packages. To interface MICROSAR.LIN to the application and hardware, you will still need the following BSW modules:

- Hardware-specific LIN driver (LINDRV) from MICROSAR.MCAL
- Hardware-specific transceiver control (LINTRCV) from MICROSAR.EXT
- General communication modules and gateway functions (COM, PDUR) from MICROSAR.COM

The modules in MICROSAR.MCAL and MICROSAR.EXT are available for many different microcontrollers and transceivers.

6.6 Other Relevant MICROSAR Products for LIN

- DET, ECUM and COMM from MICROSAR.SYS
- MICROSAR XCP enables measurement and calibration according to ASAM XCP. This module was specially optimized to be used together with CANoe.XCP and CANoe.AMD as well as CANape. For LIN ECUs, MICROSAR XCP contains the related transport layer LINXCP. Since XCP-on-LIN has not been officially defined, this XCP-on-LIN implementation is an extension of the ASAM standard.

Beyond the AUTOSAR standard, MICROSAR XCP supports generic readout of measuring objects. As a result, no addresses must be defined and updated in the a2l file. Data from any version or variant can be extracted with an a2l file independent from the MCU build. The generic readout feature requires usage of CANoe.AMD or CANape as XCP tool.

- For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000If allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000If.
6.7 The Vector toolchain for developing LIN ECUs

Figure 12: Vector offers you a comprehensive range of products and services for your LIN projects
7 MICROSAR.ETH – AUTOSAR Basic Software Modules for Ethernet-Based Communication

The Internet Protocol and the higher level Transport Protocols UDP and TCP are very widely used standards for high speed data exchange over Ethernet.

The MICROSAR.ETH (Ethernet) package contains the AUTOSAR BSW modules and a TCP/IP stack developed to an automotive standard for Ethernet-based communication between ECUs. AUTOSAR 4.0 is the first version to specify Ethernet as a network technology. In AUTOSAR 4.1, the specifications were substantially modified and extended. Further extensions, e.g. Ethernet switch configuration and time synchronization between ECUs are specified in AUTOSAR 4.2. The BSW modules of MICROSAR.ETH are available according to AUTOSAR 4.x and as a supplement to AUTOSAR 3.x.

Figure 13: The MICROSAR.ETH BSW modules according to AUTOSAR 4.2

7.1 Overview of Advantages of MICROSAR.ETH

> BSW modules are available for AUTOSAR 4.1, 4.0 and as a supplement to AUTOSAR 3.x.
> TCP/IP stack developed to an automotive standard and tested by the Fraunhofer ESK organization
> No Open Source Software
> Seamless integration, e.g. of Vehicle-to-Grid communication (MICROSAR V2G) and Audio/Video Bridging (MICROSAR AVB), in the AUTOSAR Ethernet- and TCP/IP stack
> Simple integration of customer-specific functions/modules on all levels

7.2 Application Areas

With MICROSAR.ETH in the ECU (Server) and a conventional PC or diagnostic tester as Client, you can

> diagnose the vehicle in accordance with ISO 13400-2 (DoIP) and
> reprogram ECUs quickly and in parallel.
The larger data throughput of Ethernet shortens total software download and diagnostic times considerably. An existing gateway in the vehicle can be used to route diagnostic requests to internal vehicle networks. For instance, this gives you the ability to reprogram multiple CAN ECUs in parallel over DoIP. In combination with other MICROSAR packages, MICROSAR.ETH implements the required gateway functionality. If MICROSAR.ETH is being used in the flash bootloader (FBL), an ECU that is connected to the Ethernet network (e.g. the gateway itself) can be reprogrammed directly over DoIP.

MICROSAR XCP on Ethernet is available to you for measuring and calibrating Ethernet ECUs, which gives you the benefit of larger bandwidth. XCP routing extends a gateway by adding the ability to calibrate also CAN and FlexRay ECUs over XCP via the Ethernet (vehicle) port.

Along with the application areas of diagnostics, measurement and calibration, in which Ethernet-based communication occurs between the external infrastructure and the vehicle, MICROSAR.ETH also offers the ability to efficiently use vehicle-internal Ethernet networks. You could transmit data service-oriented, for instance. This might involve use of the Service Discovery BSW module which was introduced in AUTOSAR 4.1.1. In conjunction with service-oriented communication, the SOME/IP serialization protocol is also used. An implementation of SOME/IP is available as an RTE transformer. You will find more information on the SOME/IP transformer in the chapters on MICROSAR.RTE and MICROSAR.COM.

Of course, you can also transmit data on Ethernet in a signal- and PDU-based way.

Parts of MICROSAR.ETH also serve as the basis for Vehicle-to-Grid communication and audio/video bridging. You will find more details on these application areas in the sections on MICROSAR V2G and MICROSAR AVB.

### 7.3 Functions

The following BSW modules from MICROSAR.ETH contain the functions defined in AUTOSAR 4.1. For use in an AUTOSAR 4.0 or AUTOSAR 3.x software stack, they have suitable compatible interfaces:

- **ETHIF**: The Ethernet Interface enables hardware-independent access to the Ethernet driver (ETHDRV) and Ethernet transceiver driver (ETHTRCV). Starting with AUTOSAR 4.1, this module is also responsible for VLAN handling. Hardware-independent control of Ethernet switch drivers (ETHSWTDRV and ETHSWTEXT) has been part of ETHIF since AUTOSAR 4.2.

- **ETHSM**: To start up or shut down communication in Ethernet clusters, the Ethernet State Manager (ETHSM) provides an abstract interface to the Communication Manager (COMM). The ETHSM accesses the Ethernet hardware over the ETHIF.

- **ETHTSYN**: Time Synchronization over Ethernet (ETHTSYN) implements the Generalized Precision Time Protocol (gPTP) according to IEEE 802.1AS. This permits clock synchronization between Ethernet ECUs. The Synchronized Time Base Manager (STBM) BSW module from MICROSAR.SYS is available as a higher-level time coordinator.

- **ETM**: The Ethernet Testability Module (ETM) implements a standardized counterpart for protocol conformity tests. The module enables an externally connected test environment to trigger defined actions, e.g. sending UDP packages or creating a TCP-connection. The ETM module is specified in AUTOSAR 4.3 and currently available as an extension.

- **TCPIP**: This module contains all protocols for UDP- and TCP-based communication. It supports IPv4 and IPv6 and parallel operation of IPv4 and IPv6 in one ECU. It contains the following protocols:
  - IPv4, ICMPv4 and ARP
  - IPv6, ICMPv6 and NDP
  - UDP, TCP, DHCPv4 (Client) and DHCPv6 (Client)

  In conjunction with Ethernet switch support in AUTOSAR 4.2, the TCPIP module was supplemented with a DHCP server which assigns IP addresses based on the switch port.

- **SOAD**: The Socket Adapter (SOAD) converts the communication over PDUs defined in AUTOSAR into socket-oriented communication. In AUTOSAR 4.0, the SOAD also contains the diagnostic functionality (DoIP) defined in ISO 13400-2. In AUTOSAR 4.1, this plug-in is made separate and is specified as an independent module (DOIP). Moreover, extensions for XCP routing are implemented in the SOAD. The option "SOAD (BSD)" enables the SOAD and the overlying modules to be used in a non-AUTOSAR environment, e.g. LINUX.
DOIP: Effective with AUTOSAR 4.1.1, the Diagnostics over IP (DOIP) module contains the diagnostic functionality by the same name according to ISO 13400-2. Up to and including AUTOSAR 4.0.3, this functionality is part of the Socket Adapter (SOAD).

SD: Service Discovery (SD) is specified for the first time in AUTOSAR 4.1.1. An ECU informs its communication partners about the availability of its services with the protocol implemented in this module. In addition, ECUs can register to receive automatic notifications, e.g. in the event of a signal update.

UDPNM: You can implement synchronous going to sleep of Ethernet ECUs by network management over UDP (UDPNM). Beyond the standard itself, MICROSAR.ETH offers the following important extensions:

Circumstantially for some use cases, e.g. for communication with external infrastructure outside the car, additional functions for a TCP/IP stack may be necessary. In MICROSAR.ETH two extensions are available which cover the following IETF RFCs:

- IPv6 Extensions
  - RFC3810 “Multicast Listener Discovery Version 2 (MLDv2) for IPv6” (RFCs 2710, 2711, 3590)
  - RFC4941 “Privacy Extensions for Stateless Address Autoconfiguration in IPv6”

- TCP Extensions
  - RFC1323 “TCP Extensions for High Performance” (RTTM – Round-Trip Time Measurement)
  - RFC2018 “TCP Selective Acknowledgment Options”
  - RFC5482 “TCP User Timeout Option”
  - RFC5681 “TCP Congestion Control (RFCs 6298, 6582)

Options:

- Post-build loadable: This feature is available for the modules SOAD and SD.

### 7.4 Configuration

We recommend our DaVinci Configurator Pro for a convenient configuration. You will find more details in the separate product information.

Ethernet- and TCP/IP-specific configuration parameters are saved as an extension in the "ECU Configuration Description (ECUC)" for AUTOSAR 3.x. This also applies to non-specified configuration parameters within AUTOSAR 4.x.

### 7.5 Other Relevant MICROSAR Products

Based on the AUTOSAR architecture, MICROSAR.ETH, together with the BSW modules from the MICROSAR.MCAL and MICROSAR.EXT that are available separately, forms a complete communications stack for Ethernet and TCP/IP.

- You will still need the following BSW modules to interface MICROSAR.ETH to the hardware:
  - Hardware-specific Ethernet driver (ETHDRV from MICROSAR.MCAL)
  - Hardware-specific transceiver driver (ETHTRCV from MICROSAR.EXT)
  - Optional: Hardware-specific Ethernet-Switch driver (ETHSWTDRV from MICROSAR.MCAL and ETHSWTEXT from MICROSAR.EXT)

- The modules in MICROSAR.MCAL and MICROSAR.EXT are available for many microcontrollers and transceivers.

- If PDUs should be passed to other software modules of the AUTOSAR stack, you will also require the PDU Router (PDUR) module from the MICROSAR.COM package.

- The modules of the MICROSAR.SYS package can be used to control the Ethernet stack and TCP/IP stack:
  - COMM: Central coordination point for starting up and shutting down the communication stack
  - NM: Central coordination point for network management
  - DET: Detects and evaluates errors during the development period
> DEM: You can use the DEM module from the MICROSAR.DIAG package to manage the detected system events (errors and environmental data).

> MICROSAR XCP enables measuring and calibrating according to ASAM XCP. The module was especially optimized for use together with CANoe.XCP and CANoe.AMD as well as CANape. For Ethernet ECU's, MICROSAR XCP contains the related transport layer ETHXCP.

> For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000If allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000If.

> The Vehicle-to-Grid application cases are covered by the modules of the MICROSAR V2G package. Also building upon MICROSAR.ETH are the modules from MICROSAR AVB that are available for audio/video bridging.

7.6 Other Relevant Products for Ethernet

With the CANoe option "Ethernet" you can conveniently extend your existing CANoe installation to include the ability to analyze and simulate Ethernet-based communication.

The VN5610 network interface is recommended as the hardware interface, especially when BroadR-Reach® is used as the physical layer. Along with two Ethernet channels (individually configurable for BroadR-Reach® or 100BASE-TX), it also offers two High-Speed CAN channels.

7.7 The Vector Toolchain for Developing Ethernet ECUs

Figure 14: Vector offers you a comprehensive range of products and services for your Ethernet projects.
8 MICROFAR V2G – Basic Software Modules for Communicating with External Infrastructure

Ethernet and the higher-level TCP/IP stack provide the basic technology needed to communicate with vehicle-external infrastructure.

The MICROFAR V2G (Vehicle-to-Grid) package contains BSW modules for intelligent charging of electric and hybrid vehicles and for communication with the infrastructure over Internet technologies such as HTTP. All of the modules from this package are not specified in AUTOSAR. However, they are integrated in the Vector AUTOSAR solution. The extensions are offered for both AUTOSAR 4.x and AUTOSAR 3.x. Required as a basis for MICROFAR V2G are modules from the MICROSAR.ETH package. You will find details on this in section 8.5.

![Figure 15: The MICROFAR V2G BSW modules according to AUTOSAR 4.2](image)

8.1 Overview of Advantages of MICROFAR V2G

- Implements all protocols needed for Smart-Charge-Communication (SCC)
- Supports communication over Internet mechanisms and protocols
- Seamlessly integration of all BSW modules into an AUTOSAR environment
- Easy to incorporate customer-specific functions by generic interfaces

8.2 Application Areas

MICROFAR V2G lets you perform intelligent charging of electric and hybrid vehicles at a suitable charging station. Supported are the standards

- ISO 15118 and
- DIN 70121

with their options of charging by AC or DC current.
You can use the modules of the MICRO SAR V2G package to also have your ECU communicate with a server via commonly used Internet protocols.

If necessary, the communication may also be encrypted.

### 8.3 Functions

MICROSAR V2G contains the following BSW modules:

- **DNS**: The DNS module contains a DNS Resolver. It is responsible for resolving a domain, e.g., Vector.com, into a valid IP address.
- **TLS**: This module contains a Transport Layer Security Client. TCP-based communication is encrypted with TLS. You can select the encryption algorithm to be used.
- **HTTP**: One application of the Hypertext Transfer Protocol is to transmit browser requests to a server. The module contains an HTTP Client.
- **XML Security**: This module is used to generate and validate XML signatures to EXI-encoded data based on the W3C XML Security Standard.
- **EXI**: The EXI module is used to interpret XML documents and convert them to binary format. This makes processing and transmission of the files more efficient, in order to economize on communication bandwidth.
- **SCC**: This module is responsible for Smart Charge Communication according to ISO 15118 or DIN 70121, and it contains the V2GTP transport protocol used for this purpose. It supports DC and AC charging as well as the related Plug-and-Charge (PnC) and External Identification Means (EIM) profiles.

In addition, the following modules are available within an AUTOSAR 3.x environment:

- **XML Engine**: The XML Engine module contains a parser for processing and a generator for creating valid XML 1.0 documents. It is used in the Vehicle2Grid field.
- **JSON**: This module contains a JSON parser. JSON is a JavaScript-based data exchange format and can be used instead of XML.

### 8.4 Configuration

The modules of the MICRO SAR V2G package are configured with DaVinci Configurator. You will find more details in the separate product information.

The specific configuration parameters are saved as an extension in the “ECU Configuration Description”.

### 8.5 Other Relevant MICRO SAR Products

MICROSAR V2G builds upon the MICRO SAR.ETH package and requires the Ethernet stack and TCP/IP stack as a basis for communication. It comprises the following modules:

- **Ethernet Interface (ETHIF)** for abstracting the underlying hardware
- **Ethernet State Manager (ETHSM)** for switching Ethernet-based communication on and off
- **The TCP/IP stack (TCPIP)** with the related IP Version (IPv4 and/or IPv6)

Also needed are an Ethernet driver (ETHDRV) from MICRO SAR.MCAL and an Ethernet transceiver driver (ETHTRCV) from MICRO SAR.EXT. Special drivers and transceiver drivers for Powerline Communication (PLC) are available for Smart Charge Communication here.

If the control of Smart Charge Communication is implemented via an AUTOSAR software component, we recommend the use of MICRO SAR.RTE.

The module DET from MICRO SAR.SYS is available for detecting and evaluating errors during the development period.

The module DEM from the MICRO SAR.DIAG package can be used to manage detected system events (errors and environment data).
8.6 Other Relevant Products for Ethernet

You can conveniently extend your existing CANoe installation to include the ability to analyze and simulate Ethernet-based communication with the related CANoe Option for Ethernet. The Smart-Charge-Communication Add-On, which is available free-of-charge, also lets you analyze SCC data traffic in CANoe. This lets you set up complex vehicle and charging station simulations based on the DIN standard.

Vector offers a plug-in card for Powerline Communication for the VT system.

8.7 The Vector Toolchain for Developing Ethernet/V2G ECUs

![Diagram of the Vector Toolchain for Developing Ethernet/V2G ECUs]

Figure 16: Vector offers you a comprehensive range of products and services for your Ethernet/V2G projects.
9 MICRO SAR AVB – Basic Software Modules for Audio/Video Communication via Ethernet

MICRO SAR AVB (Audio/Video Bridging) over Ethernet enables quick and reliable transport of audio/video data. The MICRO SAR AVB package contains various BSW modules which are overlaid on the Ethernet interface, e.g. from MICRO SAR.ETH. The solution based on AUTOSAR 4.x supports AVTP (Audio/Video Transport Protocol), PTP (Precision Time Protocol) and on request also BMCA (Best Master Clock Algorithm). This makes it possible to implement AVB end points as well as simple bridge functionality.

Figure 17: The MICRO SAR AVB BSW modules according to AUTOSAR 4.2

9.1 Overview of Advantages of MICRO SAR AVB

- The BSW modules are optimized for AUTOSAR, but they may be integrated in other environments as well.
- Trouble-free integration into the Ethernet stack MICRO SAR.ETH. This enables parallel use of AVB, DoIP and TCP/IP, for instance.
- Easy to incorporate customer-specific functions and modules
- Supports various Ethernet controllers
- Supports VLANs for isolating and prioritizing data, e.g. for audio, video and diagnostics.

9.2 Application Areas

9.2.1 A/V Streaming

The MICRO SAR AVTP enables the exchange of audio/video data, including their time stamps, between different end points. It is implemented according to the specification IEEE 1722/1722a.
9.2.2 Choosing the Most Precise Clock

Before a precise clock can be displayed system-wide, the device with the most precise clock – typically a bridge – must be defined. It can be set statically or dynamically through BMCA. This involves use of the specification IEEE 802.1AS.

9.2.3 Displaying a Synchronous System Time

The ECU with the most precise time distributes it in the network. This means that all end points and bridges are working with the same time prescribed by the Grand-Master. This makes it possible to transmit an A/V data stream and play it back time-synchronously. The protocol for distributing the time stamp is implemented in the PTP module according to the specification IEEE 802.1AS.

To enable an exact time measurement, it may be necessary to use extended hardware support which is typically implemented in the "High-End Feature" of the Ethernet driver.

9.3 Functions

The following BSW modules from MICROSAR AVB contain the functions defined in the above mentioned IEEE specifications. Also necessary are software properties which permit seamless integration in an AUTOSAR environment.

- **AVTP:**
  - Interface to the ETHIF module for receiving and sending AVTP frames
  - Distinction made between Stream and Control channels
  - Display and validation of the time stamp
  - Detection of the transported data stream

- **PTP:**
  - Initialization of all necessary hardware interfaces
  - Ethernet frames with Ethertype 0x88F7 are routed from the ETHIF to the PTP
  - Support of General and Event messages
  - Support of calculation of propagation time delays: Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
  - Support of transport of the synchronous time stamp: Sync, Follow_Up

- **BMCA:**
  - Support of PortAnnounceReceive, PortAnnounceInformation, PortRoleSelection, PortAnnounceTransmit

- **SRP:**
  - Data stream registration with admission control

9.4 Configuration

We recommend that you use our DaVinci Configurator Pro for a convenient configuration. You will find more details in the separate product information.

AVB-specific configuration parameters are saved as an extension in the "ECU Configuration Description".

9.5 Interfaces to Related MICROSAR Products

Based on the MICROSAR architecture, MICROSAR AVB – together with the basic software modules from the separately available packages MICROSAR.MCAL, MICROSAR.EXT and MICROSAR.ETH – forms a complete communication stack for AVB in the automobile. To interface MICROSAR AVB to the hardware, you will still need the following BSW modules:

- Hardware-specific Ethernet driver (ETHDRV from MICROSAR.MCAL)
- Hardware-specific transceiver driver (ETHTRCV from MICROSAR.EXT)
- Ethernet driver abstraction and control layer (ETHIF, ETHSM from MICROSAR.ETH)

The modules in MICROSAR.MCAL and MICROSAR.EXT are already available for many microcontrollers and transceivers, and they will be also developed for new microcontrollers and transceivers as necessary.

9.6 Additional Relevant MICROSAR Modules for Ethernet

- TCPIP, SOAD, DOIP, SOME/IP from MICROSAR.ETH
- DCM and DEM from MICROSAR.DIAG
9.7 Other Relevant Products for Ethernet

The CANoe option "Ethernet" lets you conveniently extend your existing CANoe installation to include the ability to analyze and simulate Ethernet- and AVB-based communication.

The VN5610 network interface is recommended, especially when BroadR-Reach® is used as the physical layer. Along with two Ethernet channels (individually configurable for BroadR-Reach® or 100BASE-TX), it also offers two High-Speed CAN channels.

9.8 The Vector Toolchain for Developing Ethernet/AVB ECUs

Figure 18: Vector offers you a comprehensive range of products and services for your Ethernet/AVB projects
10 MICROSAR.MEM – AUTOSAR Basic Software Modules for Memory Management

The MICROSAR.MEM package contains all of the AUTOSAR modules for memory management: NVM, MEMIF, EA and FEE. They support management, checking and restoring of data from nonvolatile memories (Flash or EEPROM). The Basic Software Modules (BSW) from MICROSAR.MEM are fast, reliable and robust.

![Figure 19: MICROSAR.MEM modules according to AUTOSAR 4.2](image)

### 10.1 Overview of Advantages

- Available for AUTOSAR 4.x and 3.x
- Exceptionally secure data transactions
- Efficient data accesses
- Efficient and robust management of nonvolatile memories
- Redundant storage of management data increases reliability of data access
- Inter-module configuration of the entire memory stack
- Platform-optimized memory stack solution from a single source

### 10.2 Application Areas

MICROSAR.MEM contains AUTOSAR services for reading, writing and erasing persistent application data in flash and/or EEPROM memories. This gives the functional software hardware-independent access to memory. The application does not need to know the specific type of memory that exists on the platform or whether this memory is internal to the controller or externally connected to it.
10.3 Functions

The BSW modules in MICROSAR.MEM contain functions defined in AUTOSAR 4.x. In each memory stack, you will need the BSW modules NVM and MEMIF from MICROSAR.MEM. They handle block-oriented and technology-independent access to the memory areas – without requiring prior knowledge of memory attributes. Your memory stack will need additional BSW modules, depending on the use case:

> When a flash memory is used: Flash EEPROM Emulation (FEE) e.g. from MICROSAR.MEM and a flash driver (FLSDRV) that is suitable for your hardware included in the service of our MCAL Integration Package, or for external memory the module DRVEXT from MICROSAR.EXT. To manage the data, the FEE module needs at least two physical flash sectors.

> When using an EEPROM: EEPROM Abstraction (EA) from MICROSAR.MEM and an EEPROM driver (EEPDRV), for example the module DRVEXT from MICROSAR.EXT for external memory.

It is possible to mix multiple flash and EEPROM chips in one ECU.

For special requirements, Vector offers platform-optimized solutions, e.g. for using the BSW module EA in data flashing or optimizing the FEE module for specific hardware.

Beyond the standard, MICROSAR.MEM contains the following important services:

> Fixed, predefined maximum execution times for all MICROSAR.MEM functions. This enables system optimizations to shorten access times.

> NVM: Allocates RAM for CRC memory storage

> NVM: Dedicated interface for the DCM diagnostic module for direct read-out and modification of data blocks

> NVM and EA: Additional configurable transaction security, which is a standard feature of the FEE module

> FEE: High-performance administration of stored memory data

> FEE: Common usage of FEE module by Flash Bootloader (FBL) and application possible - also with common memory blocks. An update of the ECU software can be done without adjustment of the FBL.

> FEE/EA: Redundant storage of management data for increased reliability of data access

> FEE: Flexible placement of the FEE sectors in the DataFlash

> FEE: Services for handling of undervoltage situations

> FEE: Frequently used data is isolated from extremely important data by introducing partitions. This further increases data availability in fault situations (e.g. reset while writing or erasing data).

The following function is optional:

> NVM: Block type “DATASET_ROM” with multiple ROM blocks

> FEE: Update support for adjustment of non-volatile memory after ECU re-programming. This is done with a new configuration table (content and size).

10.4 Configuration

We recommend DaVinci Configurator Pro for a convenient configuration. This contains certain functions that make work simpler, such as optimization assistance, visual representation of flash utilization, etc. See the separate product information for more details.
10.5 Other MICROSAR Products relevant for a Memory Stack

Based on the AUTOSAR architecture, a complete memory stack can be formed with memory services from MICROSAR.MEM together with other platform-specific BSW modules from the separately available MICROSAR.MCAL and MICROSAR.EXT packages.

- FLSDRV and/or EEPDRV from the MCAL Integration Package
- DRVEXT from MICROSAR.EXT for external memory chips

In addition, it is easy to integrate MCAL drivers from semiconductor manufacturers in the MICROSAR memory stack. Depending on the desired safety level, you can verify your memory data with the checksum module (CRC) from the MICROSAR.LIBS bundle.
11 MICROSAR.SYS – System-related Basic Software Modules for AUTOSAR

The system services in the MICROSAR.SYS basic software modules (BSW) cover an important part of your AUTOSAR ECU’s basic functionality. They are called by the functional software (via the RTE) and the remaining BSW modules. The modules of MICROSAR.SYS offer all key functions for state handling of the ECU.

Figure 21: MICROSAR.SYS modules according to AUTOSAR 4.2

11.1 Overview of Advantages

> Available for AUTOSAR 4.x and 3.x
> The ECUM module is included as either a resource-saving Pre-Compile variant or as a flexible Post-Build solution. You will find more information on this in the "Identity Manager for AUTOSAR" section.
> Easy configuration of the initial BSWM and ECUM by assistants in DaVinci Configurator Pro
  > Create initialization sequences
  > Configure an ECU state machine (startup, shutdown, etc.)
> Managing of the communication modes

11.2 Application Areas

System services include power and mode management, control of all communication channels and partial networks, monitoring of individual software components (SWC) of the functional software and within AUTOSAR 3.x scheduling of all BSW modules.
11.3 Functions

The BSW modules in MICROSAR.SYS contain functions defined in AUTOSAR 4.x:

- **BSWM**: The Basic Software Mode Manager manages mode change requests from the BSW modules and SWCs and executes them according to the standardized action lists. For example, the BSWM module is responsible for activating and deactivating PDU groups and NM-PDUs for diagnostics.

- **COMM**: The “Communication Manager” monitors the state changes of the communication channels connected to the ECU and of the sub-networks configured in the ECU. It can keep the ECU awake and ready for communication as necessary. Furthermore, it coordinates access of all SWCs to the communication channels and sub-networks. Optionally, COMM supports the bus type “Internal”.

- **BSWM and COMM**: Support of partial networking

- **DET**: The Development Error Tracer collects the development errors of the SWCs and BSW modules. Optionally, DET supports the Service Ports.

- **ECUM**: The ECU State Manager is responsible for Startup, Shutdown and WakeUp. In AUTOSAR 3.x, there are other fixed defined operating states that are managed by the ECUM. In AUTOSAR 4.x, these operating states are flexibly defined by the user in the BSWM. This makes it possible to implement individual energy-saving states or different power-up behaviors.

- **SCHM**: The Schedule Manager/BSW Scheduler coordinates execution of the BSW modules. In configuring you define tasks and the cycle times of the BSW modules. You also define the exclusive areas settings for each module centrally. For AUTOSAR 3.x, SCHM is part of MICROsAR.SYS. In AUTOSAR 4.x, the MICROSAR.RTE assumes the functionality of the SCHM.

- **WDGM and WDGIF**: The Watchdog Manager monitors the correct operation of the functional software with the modules WDGF and WDGDV (MICROSAR.MCAL).

MICROSAR.SYS contains the following important services that extend beyond those of the AUTOSAR standard:

- **COMM**: Compatibility with OSEK NM (for AUTOSAR 3.x)

- **WDGM, WDGIF**: Precise supervision of defined time windows for the watchdog (even for high resolution window watchdogs)

- **ECUM**: The ECUM module is implemented according to the AUTOSAR ECUM Flex specification, so it offers a high level of configuration options that can also support complex state transitions. When ECUs are being developed that have reduced requirements for state management, the ECUM module can optionally also behave compatibly with the AUTOSAR ECUM Fixed Specification. In this case, the ECUM module offers the following functionalities:
  - **ECUM Run Request Protocol**
  - **ECUM State Management**
  - **ECUM Fixed Compatible Service SWC interfaces**

The following BSW modules are available as options:

- **CSM**: The Crypto Service Manager gives the SWCs and the BSW modules uniform access to cryptographic functions such as AES 128 encryption (Advanced Encryption Standard).

- **STBM**: The Synchronized Time-Base Manager enables precise time synchronization between different parts of the ECU software. It does this by making one or more common time bases available to the BSW modules and the SWCs. The relevant Tsyn modules for the CAN, FR and ETH bus systems are available to serve as a communication service for vehicle-wide time synchronization of ECUs.

- **WDGM**: Program flow and deadline monitoring to observe the SWCs.

- **Post-build loadable and Post-build selectable**: For details, please refer to the section "MICROSAR Variant Handling”.

11.4 The Basis Software Manager (BSWM)

The BSWM is a central part of the mode management and is implemented according to the AUTOSAR 4 standard. However, numerous helpful features beyond the standard offer extended comfort while configuring your ECU software.
To react on mode changes in other BSW modules or to request such mode changes, the BSWM module in AUTOSAR 4 allows free configuration of arbitrage rules, logical expressions and actions.

Due to the given AUTOSAR configuration structure, this may become very complex within short time because of the fact that even simple configurations request numerous intertwining steps:

- Actions must be combined to action lists.
- These lists are linked to the true or false result of a predefined logical expression by a rule.
- The logical expression itself consists of one or more conditions which are based on the incoming modes (request ports).

Also standard tasks such as the configuration of a state machine in the style of ECUM Fix in AUTOSAR 3, the call of the corresponding initialization functions of the BSW modules with the matching parameters or switching on and off PDU groups (I-PDU) rises to a challenge even for experienced developers.

DaVinci Configurator Pro supports you by providing intelligent and powerful assistants which are able to solve many of the above mentioned tasks automatically and also lets you configure them by a simple click afterwards. This especially applies to:

- the ECU state machine (ECU State Handling, see figure below)
- the initialization of the basic software modules (Module Initialization) and
- switching of PDU groups (Communication Control).

![Figure 22: pre-configured state machine / auto-configuration of the BSWM with DaVinci Configurator Pro. Also visible: the auto-configuration for module initialization and communication control and the area for your project related configuration of the BSWM (Custom Configuration).](image)

Though this is not a static configuration, rather all necessary parameters are considered. If parameters are changed, e.g. if a new PDU group is created, the tool will recognize this immediately and will inform the user about a necessary re-configuration.

The built-in assistants of DaVinci Configurator Pro also support free configuration tasks such as creating rules or action lists. They guide you through the configuration, offer know-how on possible and necessary parameters and they identify mistakes and offers possible corrections. However, they also accept settings which you intentionally want to set differently.

The MICROSAR BSWM provides a lot more than specified by AUTOSAR. As an example, it is also possible to switch on and off the analysis of rules at runtime. You can realize a timer whose expiration can be analyzed by the BSWM. The timer can be started and stopped by actions. If not all SWCs which you have to connect are available yet, the BSWM itself can create all necessary mode declarations and hereby allow you to perform a bottom-up configuration.
11.5 Watchdog for ISO 26262 applications

All watchdog modules are also available as SEoOs (Safety Element out of Context) for safety-relevant functions up to ISO 26262 / ASIL D. They are well suited for validation of run-time monitoring of tasks as well as flow control for the SWCs. For more details, please refer to the chapter about MICROSAR Safe.

11.6 Configuration

For convenient configuration, we recommend the DaVinci Configurator Pro. For more details, please refer to the separate Product Information.

11.7 Other Relevant MICROSAR Products

> DIAG: The system service for diagnostics is available separately in the MICROSAR.DIAG package.
> OS: MICROSAR.OS, which is available separately, may be used as the operating system.
> LIBS: The LIBS package contains the Cyclic Redundancy Check Library (CRC), Crypto Abstraction Library (CAL) and the E2ELIB. It is available as a separate package for AUTOSAR 4.x (LIBS is part of MICROSAR.SYS for AUTOSAR 3.x.)
12 MICROSAR.DIAG – AUTOSAR-compatible implementation of the UDS protocol

MICROSAR.DIAG contains the BSW modules for implementing the UDS protocol ISO 14229-1:2006 (resp. ISO 14229-1:2013) according to AUTOSAR, making it the diagnostic software for your vehicle project. MICROSAR.DIAG handles a number of tasks:

> OEM-specific implementation of fault memory and its management
> OEM-specific implementation of the diagnostic protocol for communication between the diagnostic tester and the ECU
> Deactivation of certain functionalities due to active error entries

Combined with CANdelaStudio, the widely used specification tool for creating diagnostic data, you get a complete diagnostic solution from a single source.

Figure 24: MICROSAR.DIAG modules according to AUTOSAR 4.2

**12.1 Overview of advantages**

> OEM-independent solution for AUTOSAR 4.x and 3.x
> Customized solutions available for many automotive OEMs
> Vector’s numerous years of experience in the diagnostics field
> Supports OBDII and WWH-OBD (Euro VI)
> Variants handling already included for diagnostic configuration
> Configurable with the CANdela-and ODX format
> Generation of optimized application code templates

**12.2 Application Areas**

Beyond the AUTOSAR standard, every OEM has its own requirements for diagnostics. That is why Vector offers MICROSAR.DIAG with OEM-specific extensions. It is well-suited for production use and is already available to many OEMs. For ECUs without a special diagnostic specification, an OEM-independent bundle of MICROSAR.DIAG is available.
MICORSAR DIAG can be used for legal requirements of today and tomorrow, such as EURO VI. Support of OBDII (ISO 15031/ SAE J1979) and WWH-OBD (ISO27145) is available as an option.

If your ECU requires variants in the diagnostic configuration, MICROSAR.DIAG offers a high-performance solution for this. You can define up to 31 different parameterizations and store them in the ECU, in a resource-optimized way. This avoids redundancies in the ECU software, because identical interfaces to the same data, services and DTCs are combined in the generated diagnostic code.

12.3 Functions

The BSW modules in MICROSAR.DIAG contain the functions for the three BSW modules DCM, DEM and FIM that are defined in AUTOSAR 4.x and 3.x:

12.3.1 Diagnostic Event Manager (DEM) Functions

The OEM-specific DEM module contains the implementation of the respective OEM requirements concerning the fault memory of an ECU. The OEM-independent variant is available for AUTOSAR 4.x as well as 3.x. It supports the following functions as standard features:

- Management of all DTC status bits according to the UDS standard
- Definition of individual snapshots and extended records
- Predefined extended records (e.g. OccurrenceCounter)
- Counter and time-based error de-bounce algorithms
- Suppression of low-priority errors when memory is full
- Flexible unlearning (aging) of errors
- Variants handling for diagnostic configuration
- Link time configuration
- Compressed Configuration Data to optimize code size
- Support of combined errors
- Suitable for "mixed AUTOSAR" projects

The following functions are available as options:

- OBDII (ISO 15031 / SAE J1979)
- WWH-OBD (ISO27145)
- Post-build loadable and Post-build selectable: For details, please refer to the section "MICROSAR Variant Handling".

The DEM base functionality differs only a little between AUTOSAR 4.x and 3.x, except for the interface definitions. That is why Vector offers a migration solution that you can use to easily migrate your AUTOSAR 3.x compatible SWCs to an AUTOSAR 4.x project.

12.3.2 Diagnostic Communication Manager (DCM) Functions

The DCM implements UDS and OBDII services in the ECU.

The OEM-independent variant of the DCM is available for both AUTOSAR 4.x and 3.x. For a complete list of supported services, please refer to the table at the end of this chapter about MICROSAR.DIAG.

The DCM modules for specific OEMs implement the specifications of the particular OEM. Therefore, the lists of supported services vary, e.g. in supplemental support of ResponseOnEvent or LinkControl. We would be glad to provide you with detailed information on this. In addition, the DCM contains the following extensions as standard features:
> Variant handling for diagnostic configurations
> Easy integration of the Vector Flash Bootloader
> Generation of an application code template for the ECU software (AUTOSAR 3.x)
> J1939DCM: DCM module specially designed for heavy-duty vehicles

The following functions are available as options:
> Support of OBDII (ISO15031-5)
> WWH-OBD (ISO27145)

12.3.3 Function Inhibition Manager (FIM) Functions
MICORSAR FIM contains the functional features of AUTOSAR 4.x and 3.x as standard.

12.4 Configuration and Parameterization
You adapt the BSW modules from MICROSAR.DIAG to your application’s needs comfortably by configuration with DaVinci Configurator Pro. This may be done either with the help of a CANdela or ODX file or by an “ECU Configuration Description”.
For AUTOSAR 3.x, diagnostic-specific parameterization of the DCM is performed exclusively via a CANdela file. You can create it quickly and simply or import it from most commonly used ODX dialects with the proven “diagnostic authoring tool” CANdelaStudio.

Figure 25: Parameterization of the MICROSAR.DIAG modules is done with CANdela Studio.
12.5 Scope of Delivery
In addition to the standard components, a converter for CANdela Diagnostic Descriptions is provided as well.

12.6 Services for Diagnostic Applications
- Customer-specific extensions of MICORSAR DIAG
- Creating customer-specific diagnostic applications
- Integrating diagnostics in your ECU software

12.7 Other Relevant Vector Products
To fulfill specific ISO standards, you can combine MICROsAR.DIAG with the following MICROsAR products:
- MICROsAR.CAN (ISO 15765-3 oder ISO/DIS 14229-3)
- MICROsAR.FR (ISO/DIS 14229-4)
- MICROsAR.ETH (ISO/DIS 14229-5)

You can use CANdelaStudio to parameterize the CANdela or ODX file for configuration of MICROsAR.DIAG. For more information, please refer to the separate CANdelaStudio product information.

For heavy-duty vehicle diagnostics, you will need the J1939-specific modules from MICROsAR.CAN.

12.8 Supported Diagnostic Services
The module DCM from MICROsAR.DIAG supports by default the following UDS diagnostic services:

<table>
<thead>
<tr>
<th>Diagnostic Service Name (ISO 14229-1)</th>
<th>Service ID (hex)</th>
<th>AUTOSAR 4.x: The SWC has to ...</th>
<th>AUTOSAR 3.x: The SWC has to ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiagnosticSessionControl</td>
<td>10</td>
<td>(handled in DCM internally)</td>
<td>(handled in DCM/BSWM)</td>
</tr>
<tr>
<td>ECUReset</td>
<td>11</td>
<td>(handled in DCM/BSWM)</td>
<td>+ (handled in DCM/BSWM)</td>
</tr>
<tr>
<td>SecurityAccess</td>
<td>27</td>
<td>... calculate seed/key for each security level</td>
<td>... calculate seed/key for each security level</td>
</tr>
<tr>
<td>CommunicationControl</td>
<td>28</td>
<td>+ (handled in DCM/BSWM)</td>
<td>+ (handled in DCM/BSWM)</td>
</tr>
</tbody>
</table>
The module DCM from MICROSAR.DIAG optionally supports the following OBD II Diagnose Services:

<table>
<thead>
<tr>
<th>Diagnostic Service Name (ISO 15031-5)</th>
<th>Service ID (hex)</th>
<th>The SWC has to …</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diagnostic Service Definition for CAN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request Current Powertrain Diagnostic Data</td>
<td>01</td>
<td>… handle data acquisition for each PID other than the &quot;supported ID&quot; and DEM ones</td>
</tr>
<tr>
<td>Request Powertrain Freeze Frame Data</td>
<td>02</td>
<td>… (handled in DEM module)</td>
</tr>
<tr>
<td>Request Emission-Related Diagnostic Trouble Codes</td>
<td>03</td>
<td>… (handled in DEM module)</td>
</tr>
<tr>
<td>Clear/Reset Emission-Related Diagnostic Information</td>
<td>04</td>
<td>… (handled in DEM module)</td>
</tr>
<tr>
<td>Request On-Board Monitoring Test Results for Specific Monitored Systems</td>
<td>06</td>
<td>… handle data acquisition for each TestId of a MonitorId</td>
</tr>
<tr>
<td>Request Emission-Related Diagnostic Trouble Codes Detected During Current or Last Completed Driving Cycle</td>
<td>07</td>
<td>… (handled in DEM module)</td>
</tr>
<tr>
<td>Request Control of On-Board System, Test or Component</td>
<td>08</td>
<td>… process each TestId</td>
</tr>
<tr>
<td>Request Vehicle Information</td>
<td>09</td>
<td>… handle data acquisition for each InfoType ID other than the &quot;supported ID&quot; and DEM ones</td>
</tr>
<tr>
<td>Request Emission-Related Diagnostic Trouble Codes with Permanent Status</td>
<td>0A</td>
<td>… (handled in DEM module)</td>
</tr>
</tbody>
</table>
13 MICROSAR.MCAL – AUTOSAR Driver for Control of Microcontroller Peripherals

The MICROSAR.MCAL package contains the drivers for controlling the microcontroller’s peripherals. Its drivers are fully compatible with the AUTOSAR specifications of the Microcontroller Abstraction Layer (MCAL). Each MICROSAR.MCAL driver is optimized controller-specifically.

![MICROSAR.MCAL modules according to AUTOSAR 4.2](image)

### 13.1 Overview of advantages

- Available for AUTOSAR 4.x and 3.x
- Perfect support of microcontroller peripherals
- Simplified configuration by considering cross-module parameter dependencies in the configuration tool
- Development accelerated by plausibility tests and completeness tests in the configuration tool
- Resource-saving, as functionalities can be deactivated
- Reduced hardware requirements due to optimized utilization of hardware buffers
- Gateway developments are supported by efficient auxiliary functions

### 13.2 Application Areas

MICROSAR.MCAL is a turn-key solution for driving your microcontroller peripherals. When switching to different hardware, it is not necessary to make changes to the functional software. You will only need to replace MICROSAR.MCAL to integrate the appropriate new drivers.

The MICROSAR.MCAL drivers perfectly fit to the overall MICROSAR bundle. Depending on the requirements of your application, you may use additional bundles (e.g. MICROSAR.CAN, MICROSAR.MEM, etc.) and thereby obtain a complete communications stack or memory management as an example.
13.3 Functions

The MICROSAR.MCAL package contains the driver modules CAN, ETH, FR, LIN, and IIC, and in the version for AUTOSAR 4.x the RamTst test module. The modules conform to AUTOSAR 4.x and are available for many different available microcontrollers. With AUTOSAR 4.2, the ethernet switch driver is included in the delivery.

In addition, MICROSAR.MCAL contains the following extensions:

- **CANDRV**: Notification (Callback) on message reception and after successfully sending a message. This allows to automatically execute application-specific code.
- **FRDRV**: Supports self-diagnostics. When the FlexRay controller detects an error, it notifies the application so that it can call up the error status.

The following functionalities or modules are available as options:

- **CANDRV**: Option “HighEnd” offers extended filter options for Multiple Basic CAN objects, RX queue to shorten the interrupt time during reception, individual polling of mailboxes to assure data consistency and reduction of interrupt load.
- **IICDRV**: The IICDRV contains drivers for interfacing to external peripheral chips via the Inter-Integrated Circuit Bus (I2C) (AUTOSAR extension).
- **Post-build loadable / Post-build selectable**: these features are available as options for several determined MCALs.

13.4 Configuration

For convenient configuration, we recommend the DaVinci Configurator Pro. For more details please refer to the separate product information.

![Configuration with the DaVinci Configurator: Clock settings based on example of Freescale MPC560x8 (Bolero)](image)

13.5 MICROSAR MCAL Integration Package

The MICROSAR MCAL Integration Package covers several work packages which realize an unobstructed integration of a 3rd party MCAL into the embedded software and tool environment of Vector. In conjunction with the customer and the semiconductor
vendor, Vector performs a set-up of the MCAL. We check its conformity -respectively its possibility to integrate it- and create supplemental packages which allow to embed the 3rd party Software into the Vector embedded software and the Vector tool chain.

![Figure 29: Workflow within the MCAL Integration Package](image)

Before delivery of the stack, the MCAL by default is removed again so that at customer side the desired packages have to be supplemented. A script is provided with the delivery which to a large extend automates the merge of all packages and applies indispensable changes to the packages of the semiconductor vendor. This allows an independent update of the MCAL by the customer at any point in time.

In case of an update of the used MCAL you can of course request the service of our MCAL integration package again.

This service portfolio for the provided MCAL makes on-site setup of the MCAL easy to a large extend, so that your team can focus on the development of your application.

The MCAL Integration Package contains the following services:

![Figure 30: service profile of the MCAL Integration Package](image)
13.5.1 Coordination with the customer

- Coaching in relation to the timeline of the MCAL vendor
- Check on the basic parameters as compiler version, AUTOSAR Version, etc.
- Early clarification of technical issues, e.g., in case of compatibility issues
- Lead contact in relation to the interaction of BSW and MCAL

13.5.2 Embedded-Integration

- Incoming goods inspection and proof of integratability: start-up of the MCAL on a suitable evaluation board and performance of integration tests with the MICROSAR basic software.
- Compile- / Link-Test in relation to the higher software layers
- Test of the basic functionalities of the MCAL which are relevant for the BSW (e.g., CAN communication, NV data storage, etc.)
- Create/serve embedded-interfaces between BSW and MCAL (MemMap, Compiler Config, Make-Files)
- Development of wrappers in case of "mixed AUTOSAR" projects

13.5.3 Tool-Integration

Depending on the characteristics of the chosen MCAL there are several possibilities to integrate it into the Vector tool chain. Our emphasis is to achieve the best and most comfortable solution for the customer. Basic conditions for embedding the MCAL components into the BSW configuration hereby are the following:

- existence of AUTOSAR conformant description files and
- the ability of the MCAL validators / configurators to work on AUTOSAR conformant configuration files.

If the MCAL supports these pre-conditions, it will be integrated into the configuration tool DaVinci Configurator Pro, which allows generating the module afterwards.

If proprietary formats are being used or if the MCAL configuration tool contains complex abstractions for an easy creation of the configuration and validation, Vector will provide adequate means to facilitate the parallel usage of different configuration files and tools.
14 MICROSTAR.EXT – AUTOSAR drivers for control of external devices

MICROSTAR.EXT contains communication-related AUTOSAR transceiver drivers for CAN (CANTRCV), FlexRay (FRTRCV), LIN (LINTRCV), Ethernet (ETHTRCV) and drivers for other external devices such as EEPROM, flash memory and watchdog (DRVEXT). Functions contained in the drivers were specified by AUTOSAR in the “ECU Abstraction Layer”. They conform to AUTOSAR 4.x and have each been optimized for a specific device and are available for many commonly used devices.

Figure 31: MICROSTAR.EXT modules according to AUTOSAR 4.2

14.1 Overview of Advantages
>
> Optimal control of your external transceiver and memory devices
> Available for AUTOSAR 4.x and 3.x
> Support of transceivers for partial networking
> Additional support for LIN and Ethernet transceivers
> Simplified configuration by considering parameter dependencies with other modules
> Development accelerated by plausibility and completeness checks in the Configurator

14.2 Application Areas

MICROSTAR.EXT gives you a turn-key solution for driving your external peripheral devices. So, it is not necessary to modify the functional software when switching out external hardware. All you need to do is switch out the relevant drivers from MICROSTAR.EXT.

Depending on the requirements of your application, you can add other packages (e.g. MICROSTAR.CAN, MICROSTAR.MEM, etc.) for a complete communication stack or memory management per AUTOSAR specification.

Partial networking in CAN networks requires special transceivers. For many of these transceivers a suitable driver (CANTRCV) is already available with MICROSTAR.EXT.
14.3 Functions

The BSW modules in MICROSAR.EXT contain functions defined in AUTOSAR 4.x plus the following extended functions:

- Bus error detection on FlexRay
- LIN transceiver drivers (also for AUTOSAR 3.x)
- Ethernet transceiver drivers (also for AUTOSAR 3.x)
- Ethernet Switch driver

The MICROSAR SBC driver is available as an option. It supports different peripheral devices such as transceivers and watchdogs depending on the specific hardware used. It makes it easy to link external peripherals to the MICROSAR software. Please inquire about the exact functional scope of the SBC driver in reference to the hardware you are using. Optionally, the SBC driver can also be offered within the solution SafeBSW and is available from ASIL A to ASIL D.

A generic SBC driver is also available which gives you access to the SBC registers to implement control functions for the SBC hardware.

Configuration

For convenient configuration we recommend our DaVinci Configurator Pro. For more details, see the separate product information.

Figure 32: Configuration of MICROSAR.EXT with DaVinci Configurator Pro

14.4 Other Relevant MICROSAR Products

The external devices are physically driven via SPI, DIO or a port. You will need the relevant drivers for this (SPIDRV, DIODRV or PORTDRV) from MICROSAR.MCAL.

14.5 Additional Services

Vector can offer services for integrating the configuration of your drivers or drivers by third-party producers, such as semiconductor manufacturers, in DaVinci Configurator Pro. This lets you configure the entire ECU software with a single tool, seamlessly and quickly.
15 MICROefined Autosar Input-Output Hardware Abstraction

The cluster IO establishes a connection between the application (e.g. the SWCs) and the MCAL modules. This gives the application or SWC access to I/O ports, for example to read sensor data or steer actors.

To fulfill this, the IO cluster contains specialized BSW modules. Additionally, Vector offers the possibility to establish a perfectly fitting IOHWAB for the ECU by using DaVinci Developer.

Figure 33: The MICROefined Autosar.IO cluster

15.1 Overview of Advantages

- Quick implementation of user-specific code for acquiring and providing sensor and actor signals
- Generation of code examples: the user can select read/write digital signal templates with expandable C-code
- Provision of SWC descriptions with all necessary interface definitions

15.2 Application Areas and Functions

The Digital Input Output Hardware Abstraction (DIOHWAB) establishes a connection between the application and DIO signals from the MCAL. The MICROefined Autosar DIOHWAB module therefore creates a SWC interface as well as DIO specific code templates. It offers a quick and easy access to the DIO module. This way, the DIOHWAB module covers a part of the IOHWAB and can be extended by further IOHWAB modules which for example allow access to ADC or ICU channels.

The optional SENT driver from within MICROefined Autosar.IO is an extension to the AUTO ISAR standard and allows access to sensors which are implemented according to the SENT protocol based on J2716. The driver is hardware independent through the interface connection with the ICU module. It offers the following functionalities:

- Realization of SENT master node functionality
- Multiple parallel SENT interfaces/channels
- Slow- and fast channel
> Enhanced serial message format on slow channel
> Timeout monitoring of periodic slow channel data (MICROSAR Extension)
> Debug mode for protocol debugging (MICROSAR Extension)
> Provision of sample SWC templates for the tool DaVinci Developer
  > Sender/Receiver-Port for Fast Channel data
  > Client/Server-Port for Slow Channel data

15.3 Configuration

For convenient configuration we recommend the DaVinci Configurator Pro. For further details, please refer to the separate Product Information.

15.3.1 DIOHWAB

DaVinci Configurator Pro checks for plausibility of the configuration parameters for MICROSAR.MCAL and MICROSAR.IO. The following “bottom up” approach is recommended:

> Configure the MCAL driver
> Configure MICROSAR.IO
> Generate the SWC description belonging to MICROSAR.IO

In configuring MICROSAR.IO, you define each individual signal. Any number of port prototypes may be derived from a port interface here. 1:n allocations of port prototypes to runnable entities is possible for sender/receiver interfaces. Runnable entities in the RTE and schedulable entities in SchM are user-configurable. MICROSAR.IO provides all of the necessary Client/Server interfaces and code templates needed to give the functional software access to the I/O signals via the RTE. They let the user condition and filter the signals.

![Figure 34: Configuration of the DIOHWAB module with DaVinci Configurator Pro](image)

15.3.2 IOHWAB

Using DaVinci Developer, you can create an IOHWAB SWC description and implement it into the SWC design afterwards with only little effort. Beyond this, the MICROSAR.RTE or the DaVinci Developer Option.CPG both offer the possibility to directly create a code template for your individual IOHWAB implementation from this IOHWAB SWC description. The code template is enhanced by ECU specific algorithms (debouncer, signal filters, etc) by the development engineer and connected to the MCAL API’s.
15.4 Other Relevant MICROSAR Products

If I/O signals are accessed via an external device, your I/O Stack will need an additional suitable driver from MICROSAR.EXT.

15.5 Additional Services

Vector would be glad to support you in the development of a complete and ECU specific I/O/HWAB layer for your ECU in the framework of project work. You benefit from Vector’s detailed knowledge of the AUTOSAR specification and methodology as well as its extensive experience in integration of ECU software.
16 MICROSAR.RTE - The optimized Run-time Environment for Software Components per the AUTOSAR Standard

MICROSAR.RTE (Run Time Environment) is the scalable and highly optimized AUTOSAR run-time environment from Vector. The RTE is a module introduced by AUTOSAR that manages communication between the software components (SWCs). It assures consistency of the overall information flow and represents the interface between the functional software, basic software (BSW) and complex drivers (CDD).

![Figure 35: MICROSAR.RTE module according to AUTOSAR 4.2](image)

**16.1 Overview of Advantages**

- Easy to configure and scalable
- Available for AUTOSAR 4.x and 3.x
- In-depth consistency check of the configuration
- Highly optimized code with intelligent synchronization mechanisms
- Quick entry into working with AUTOSAR, e.g. based on generated code templates for the software components (SWCs)
- Well-suited for migration projects
- Simplified testing of the application by XCP access to S/R ports, InterRunnable variables and Per-Instance memory

**16.2 Application Areas**

When the functional software of an ECU is implemented by AUTOSAR-conformant SWCs, the user needs the RTE as a run-time environment. This modular layout of the ECU software offers the user maximum flexibility: SWCs that have been manually developed or designed by model-based tools can be re-used in multiple ECU projects. The RTE only needs to be reconfigured and regenerated for the specific ECU and, if applicable, the BSW modules. It is also possible to use a SWC in multiple instances on one ECU.
In generating the MICROSAR.RTE, the user can choose between two modes:

- Contract Phase Generation for developing individual SWCs in an early phase. In this case, the generator just creates a header file for each SWC instead of the entire RTE. This makes it possible to compile the SWCs individually, e.g. to transfer them to a development partner as object code.
- RTE generation for the entire ECU software. The code generated in this mode is highly efficient and requires only little memory space. It is optimized for the entire ECU configuration and makes few demands on system resources due to short execution times and minimal interrupt disable times. One way this is achieved is by using intelligent synchronization mechanisms that are tuned to the properties of the hardware used.

### 16.3 Functions

MICROSAR.RTE is compatible with AUTOSAR 4.x and 3.x. Specifically, MICROSAR.RTE contains the following functionality:

- Sender/Receiver and Client/Server communication
- Mode management
- InterRunnable variables as well as Exclusive Areas.
- Access to Nv block software components via sender/receiver ports
- Trigger for runnables
- Online and offline calibration of SWCs are supported as well as measurement of S/R ports, InterRunnable variables and Per-Instance memory using the XCP protocol.
- Multiple instancing of SWCs and per-instance memory.
- Schedule Manager/BSW Scheduler (SCHM): Since AUTOSAR 4.0, the RTE has assumed the functionality of the SCHM module. This was previously contained in MICROSAR.SYS. For more information on the SCHM module, please refer to the chapter about MICROSAR.SYS.
- Support of the transformer interfaces for COMXF, SOMEIPXF and E2EXF. For details please refer to the chapter MICROSAR.COM.

External Client/Server communication (Inter-ECU) through the optional SOMEIPXF. Beyond the standard, MICROSAR.RTE for AUTOSAR 3 includes the following important services:

- Generation of code templates for SWCs based on the “SWC Description”. These templates contain all APIs of the RTE.
- Use of memory protection mechanisms as specified in the AUTOSAR operating system. This support is especially optimized when MICROSAR.OS is used.
- Configuration of initialization runnables for the AUTOSAR concept of “Mode-Dependent Runnables”.
- Generation of an HTML report showing RTE properties. It contains information such as the calculated RTE resource load (RAM + constants).
- Generation of an A2L file for simple linkage to existing calibration and diagnostic standards.

The following functionalities are available as options:

- MPU support for memory protection
- Multi-core support
- Post-build selectable: within the MICROSAR product family, this feature is available in the module ”Identity Manager”. For details please refer to the section ”MICROSAR Variant Handling”.

Optional features available upon request:

- Support of internal, external and background runnable triggers (for AUTOSAR 4.x)
- Scaling of ports and signals (for AUTOSAR 4.x)
- Range checks (for AUTOSAR 4.x)
Figure 36: MICROSAR.RTE enables generation of code templates for the SWCs

16.4 Basic Runtime Environment (BRE)

The Basic Runtime Environment (BRE) allows the integration of a non-AUTOSAR based application on the proven MICROSAR basic software. Using BRE a formal Software Component (SWC) Design as it is defined by AUTOSAR is not required. As the BRE allows using the existing application architecture usage of the BRE is the first step when migrating ECU projects towards an AUTOSAR architecture.

Using BRE the application accesses the BSW functionality directly without the need to define AUTOSAR software components (SWCs). The RTE as middleware between SWCs and BSW is not used and replaced by application code.

To simplify the integration of your application with the MICROSAR BSW, the BRE provides basic functionality usually implemented by the RTE:

- Generation of type definitions for service layer BSW modules
- Call of BSW main functions based on a configurable task mapping
- Managing of exclusive areas for BSW modules

As the BSW uses AUTOSAR conformant interfaces, the usage of the BRE requires that the application takes over the interface function normally provided by the RTE towards the BSW.

If the BRE provides too little functionality the MICROSAR RTE is the right choice. Experienced MICROSAR coaches can support you with your migration efforts towards an AUTOSAR compatible application and BSW architecture.

16.5 Configuration

For configuration of the RTE, you need either DaVinci Configurator Pro with Option .RTE or DaVinci Developer. For more details, please refer to separate product information.

16.6 Scope of Delivery

In addition to the standard product components, you also get sample programs with make files.

16.7 Other Relevant MICROSAR Products

The RTE requires the presence of an operating system such as MICROSAR.OS or osCAN.
17 MICRO SAR AMD – AUTOSAR monitoring and debugging

MICROSAR AMD contains many useful functions that simplify the development and testing of ECUs significantly. Core functions of AMD are reporting of errors and events from the application and the MICROSAR BSW as well as providing information on momentary CPU load and software execution times.

Figure 37: MICROSAR AMD modules according to AUTOSAR 4.2

17.1 Overview of advantages

- Simplified testing of ECUs by acquiring internal ECU data – already implemented for AUTOSAR 3
- Easy measurement of startup and go-to-sleep behavior
- Determining CPU load and execution times of the application and the basic software
- Automatic generation of required A2L file
- Reporting of BSW error states and program flow trace messages
- Use of the widely-used XCP protocol standardized by ASAM

17.2 Application areas

The MICROSAR AMD package efficiently supports you in testing your AUTOSAR ECU software. The basic software modules from MICROSAR AMD have access to all important internal variables, states and error messages of your MICROSAR basic software.

The XCP protocol (Universal Calibration Protocol) – familiar from the measurement and calibration field – is best suited for transferring ECU-internal parameters. Therefore, Vector decided to develop MICROSAR AMD based on XCP.
17.3 Functions

The functionality of the DBG (Debugging) and DLT (Diagnostic Log and Trace) modules is specified in AUTOSAR 4.x. Nonetheless, it is already available from Vector for AUTOSAR 3. In addition to the AUTOSAR standard, MICROSAR AMD offers the ability to determine CPU load and any execution times.

For display and analysis, the values read out with MICROSAR AMD are transmitted to an XCP Master via the XCP Slave.

You can use one of the following Vector tools as the XCP Master on the PC side:

- CANape – together with MICROSAR modules DLT und DBG
- CANoe.AMD from Version 8.1 – together with MICROSAR modules DLT, DBG and RTM.

With these tools and a description of the measurement objects in the form of an ASAM A2L file, you can select internal ECU data and analyze its flows. CANape was extended for AMD with the “Digital Window” and the “Status” signal display. CANoe.AMD was extended with the special “State Tracker” window. This lets you display separate states and binary signals in a well-organized layout in one window.

17.3.1 Acquiring internal variables from AUTOSAR BSW modules

In the case of ECU tests, you can use the DBG BSW module from MICROSAR AMD to determine which external and internal actions caused a state change of the BSW modules. The DBG module sends the needed internal variables of the MICROSAR BSW modules to the Master via XCP.
17.3.2 Acquiring internal variables from the MICROSAR.RTE

When using a MICROSAR.RTE as a run-time environment in your ECU software, it is possible to monitor the data flow between SWCs over XCP. After activating the “Measurement Support” option in the RTE configuration, the following RTE-internal objects are acquired:

- Inter-runnable variables
- Sender/Receiver Ports
- Mode Ports
- Per-Instance Memory

This is how you monitor the intra- and inter-ECU communication and measure or stimulate not connected sender/receiver ports.

17.3.3 Acquiring internal error messages and warnings

During run-time, the DLT BSW module acquires and transmits, via XCP, all occurring error messages and warnings that the BSW reports to the DET module. In addition, the DLT module can actively forward to the XCP Master notifications that have been made to the error memory (DEM).

To monitor the program flow and status of the application, the DLT module can report on any desired trace messages in plain text. This is done by providing a "WriteLine"-like API which can transmit any desired text at run-time. As an alternative, predefined text blocks optimized for execution time can be transmitted.

17.3.4 Determining CPU load and execution time

MICROSAR AMD and CANoe.AMD let you determine the execution time of any selected code sections from the application software or from the BSW modules. The measurement results are presented in HTML and CSV reports, which are generated with the CANoe Test Feature Set.

17.3.5 Generating the A2L file for the XCP Master

Based on the configuration data from the ECU configuration file (ECUC), you generate the A2L description file for the XCP Master with the DaVinci tools. If symbolic names exist for values of measurable objects, they are also stored for later visualization. It is also possible to incorporate other application-specific A2L fragments into the Master A2L file. Before starting the measurement, the ASAP2 Updater is used to update the A2L file with the actual addresses of the variables. The ASAP2 Updater is supplied as part of CANape and CANoe.AMD.
17.4 Configuration

For user-friendly configuration, we recommend DaVinci Configurator Pro. You will find more details on this in the separate product information.

17.5 Scope of Delivery

MICROSAR AMD consists of the following components:

- Software modules as source code
- A2L Generator (for Windows XP/Windows 7)
- BSW Module Description files
- Documentation

17.6 Other relevant products

Prerequisites for using MICROSAR AMD are the following Vector software modules:

- the simultaneous use of MICROSAR XCP for CAN, FlexRay or Ethernet, or
- VX1000 systems from Vector. These are advisable for maximum data throughput with minimal effects on execution time. You will find details on this at our product website: [www.vector.com/vx](http://www.vector.com/vx). For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000if allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000if.
18 MICROSAR Safe – Functional Safety according to ISO 26262 up to ASIL D for ECU software

The ISO 26262 safety standard defines criteria by which safety-related ECUs shall be developed in the automotive field. MICROSAR Safe from Vector gives you a solution for implementing safety-related functionality up to the highest safety level (ASIL D) in an AUTOSAR project.

AUTOSAR basic software developed according to ISO 26262 can help to reduce the number of partitions in the system and hereby can contribute to a higher performance. Many of our MICROSAR BSW modules are developed with the methods of ISO 262162/ASIL D and subsequently are able to co-exist with safety relevant SWCs without partitioning. Beyond this, further safety requirements can be implemented into the BSW if necessary.

18.1 Overview of Advantages

- Certified solution for all Automotive Safety Integrity Levels (ASIL A to ASIL D)
- ASIL and QM software sections are executed on a single microcontroller (Mixed ASIL)
- Reduces qualification costs
- Runtime optimized implementation of context management
- Monitoring of flow control and deadlines
- Validation of external communications
- Less partition switches and subsequently less runtime through safe basic software
- Suitable for Multi-core projects
18.2 Application areas

The MICROSAR Safe modules are "Safety Elements out of Context" (SEooCs), which are developed according to ISO 26262 / ASIL D.

MICROSAR Safe enables "freedom from interference" in executing safe software parts with different ASIL and non-safe software parts (QM software) on the same ECU (Mixed-ASIL systems). MICROSAR Safe is the result of many years of experience in the field of functional safety.

18.3 Functions

For projects according to the AUTOSAR 4 standard, MICROSAR Safe contains safe basic software with its core products SafeContext, SafeCOM and SafeWatchdog. They comply with the AUTOSAR specification and are compatible with all of the remaining MICROSAR modules which optionally can be provided in the context of SafeBSW as well. Beyond this, MICROSAR Safe also provides a SafeRTE.

The combination of SafeContext, SafeCOM and SafeWDG is the minimum expansion stage which is necessary for mixed-ASIL systems (ECUs which contain both ASIL functions and functions without qualification). An extension of this mandatory package is possible according to your specific requirements - up to a full BSW stack according to ASIL D.

The core products of MICROSAR Safe contain the following functionalities:

18.3.1 SafeContext – the safe AUTOSAR operating system

The MICROSAR operating system’s option SafeContext assures memory protection for ECUs with safety-relevant applications on suitable microprocessors, e.g. with a Memory Protection Unit (MPU). MICROSAR.OS (SC3/SC4) is used to separate the various software partitions from one another. This prevents unauthorized writing by SWCs to the memory of other SWCs that could corrupt the data. In addition, it is assured that the context is switched correctly during a task switch or interrupt.

By offering scheduling with related timing protection and application termination functionality developed according to ISO 26262, SafeContext improves the support of safe systems with high availability requirements.

SafeContext contains additional functions that support partitioning by the MPU e.g. protected access to privileged registers, MPU test functions and protected data exchange across partition boundaries.

18.3.2 SafeCOM - Safe intra-ECU and inter-ECU communication

Data that is exchanged between safety-relevant applications on different ECUs must be checked for correct transmission. A checksum protects the data contents and the correct sequence of data is monitored by a message counter. If one of these
checks fail, the application is informed, and it can react accordingly. This detects errors such as masquerading, failure, reversed data, etc.

The transformers COMXF, SOMEIPXF and E2EXF according to AUTOSAR 4.2 are available on QM level also within SafeCOM to support a safe communication. As an alternative Vector offers a protection wrapper within SafeCom which provides an AUTOSAR-conformant and signal-based interface. This enables convenient use of the E2E verification in the application above the RTE.

### 18.3.3 SafeWDG - Flow control of safety-relevant software components

The WDGM module from SafeWDG monitors the correct timing and execution behavior of safety-related functions. This includes the “Program Flow Monitoring”, which is specifically defined in AUTOSAR 4.x for safety-relevant applications.

The SafeWDG package implements the interface to an internal or external hardware watchdog with the help of the modules WDGIF, WDGDRV resp. WDGEXT and -on demand- can support external system basis chips (SBC) as well.

### 18.3.4 SafeRTE – safe ECU-internal communication

MICROSAR.RTE supports partitioning of memory sections via the operating system’s option "SafeContext". According to ISO 26262, a qualification of the RTE is necessary to establish a safe communication between applications within an ECU. The static analysis tool “RTE-Verify” is available as option for this purpose.

### 18.3.5 Validation of the hardware

The modules of MICROsAR Safe assume a safe hardware within the customer project. If this requirement is not fulfilled by the hardware, you may also accomplish this by suitable software test functions. This requires that you consider the project specific safety goals.

SafeContext already offers auxiliary functions in reference to the MPU. Other functions for validating the RAM, Flash, MPU, IO, etc. can be provided by Vector in the context of project work.

### 18.4 Configuration

We recommend that you use the DaVinci Configurator Pro tool to configure the BSW modules of MICROsAR Safe.

### 18.5 Scope of Delivery

MICROSAR Safe is developed according to the AUTOSAR 4 standard. In addition to the BSW modules every delivery of MICROsAR Safe also comes with the safety case documents and safety manuals for the individual Safety Elements out of Context (SEooC) which are mandatory for all safety related ECUs. The delivered software package altogether is one single SEooC.

MICROSAR Safe is also available for AUTOSAR 3 projects, enabling you to realize Mixed-ASIL projects according to this standard. The aspect “freedom from interference” in this case is supported by the functionalities MPU support including safe context switch, runtime measurement and safe external communication. MICROsAR Safe for AUTOSAR 3 contains SafeContext, SafeCOM and SafeWDG, plus SafeRTE and SafeCRC.

### 18.6 Implementation

Safety-relevant functions are supported by the above mentioned core products from MICROsAR Safe. They conform to the AUTOSAR specification, are developed according to ISO 26262/ ASIL D and in detail contain the following features:

SafeContext:

> Memory protection and safe context switching (option for MICROsAR.OS)
SafeCOM:
  > Protection Wrapper: End-to-end protection Wrapper
  > COMXF, SOMEIPXF, E2EXF
  > E2ELIB (End-to-end library)

CRC LIB SafeWatchdog
  > WDGM: Watchdog Manager
  > WDGIF: Watchdog Interface
  > WDGDRV: Driver for internal watchdog devices
  > WDGEXT: Driver for external watchdog devices
19  MICROSAR Security – Access Security for AUTOSAR ECUs

The growth in the amount of safety-related information and personal data in the automobile is making protection against intentional data manipulation and data theft increasingly more important. Security mechanisms are being used to protect the integrity, authenticity and confidentiality of information. In this area, Vector can offer you components specified in AUTOSAR 4.2.

Figure 43: Security modules according to AUTOSAR 4.2: Via CSM it is possible to use a hardware that conforms to the Standard Secure Hardware Extension (SHE).

19.1 Overview of Advantages of MICROSAR Security

> Standard-conformant implementation of security functions from a single source
> Established cryptographic algorithms
> Protection against unauthorized modification of critical data
> Authentication of communication end points
> Protection against unauthorized reading of data

19.2 Functions

The security functionality is distributed into two modules. The Crypto Abstraction Library (CAL) provides an AUTOSAR library, while the Crypto Service Manager (CSM) can be used in the System Service Layer.

Both modules offer nearly the same services, but they have different technical properties and are used differently. Both modules are designed as wrapper modules, they offer only abstract interfaces and thereby a logical service, e.g. symmetrical encryption. The actual algorithm that is used (e.g. AES-128) is set by the ECU configuration.

Both CAL and CSM offer the following functionalities:

> Access to cryptographic services.
Up to 32 different service configurations can be created for each cryptographic service. This defines the algorithm by which the service is performed.

Synchronous execution of the service.

Large data streams can be processed incrementally ("streaming").

19.2.1 The Crypto Abstraction Library (CAL):

As an AUTOSAR library the CAL is stateless which means that the context of a calculation is managed by the caller. The service is processed in the task of the caller, where any desired number of callers can place requests in parallel.

All services are realized in software.

The algorithm implementation is located in the auxiliary module Cryptographic Primitive Library (CPL). In selecting the CAL, the auxiliary module CPL from the MICROSAR.LIBS cluster is mandatory for symmetrical or asymmetrical execution.

19.2.2 The Crypto Service Manager (CSM):

The module is part of the Services Layer.

The manager offers access to cryptographic services via the RTE Port mechanism. As an alternative, BSW modules or CDDS may access the services directly.

It can be configured for synchronous or asynchronous execution.

The algorithm implementation is located in the auxiliary module Cryptographic Library Module (CRY). In selecting the CSM, the auxiliary module CRY from the MICROSAR.SYS cluster is mandatory for symmetrical or asymmetrical execution.

The services can be implemented directly in software in a CRY module, or they can be passed to a driver that is used to control the security hardware.

Upon request, we would be glad to offer you services that have not yet been defined as a package.

19.2.3 Secured OnBoard Communication (SecOC)

This module is used for secure communication between two ECUs. It prevents a third party from intervening or claiming to be the correct communication partner. This prevents manipulative interventions. The SecOC interacts with the PDU router, and it can be controlled by the application. The module offers the following functionalities:

Transmission of authenticated and integrity-protected I-PDUs. Either the CSM or CAL module calculates the authenticity validation according to the variant selected.

Prevention of replay attacks. A counter is used here, which checks - via the MAC address or a signature – to verify the currency of the signals in order to prevent attacks.

19.3 Supported Algorithms

<table>
<thead>
<tr>
<th>Service</th>
<th>API &lt;service&gt; name</th>
<th>Data Stream</th>
<th>CSM</th>
<th>CAL</th>
<th>Algorithms</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hash functions</td>
<td>Hash</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Basic</td>
</tr>
<tr>
<td>Checksum</td>
<td>Checksum</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>CRC-16, CRC-32</td>
<td></td>
</tr>
<tr>
<td>Random numbers</td>
<td>RandomGenerate</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>FIPS-186</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RandomSeed</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Authentication Code</td>
<td>MacGenerate</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>HMAC-RIPMD-160, HMAC-SHA-1, HMAC-SHA-256</td>
<td>Symmetrical Algorithms</td>
</tr>
<tr>
<td></td>
<td>MacVerify</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECB Mode</td>
<td>SymBlockEncrypt</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>AES, DES, Triple DES, RC2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SymBlockDecrypt</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBC Mode</td>
<td>SymEncrypt</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>AES, DES, Triple DES, RC2</td>
<td></td>
</tr>
<tr>
<td>Key generation</td>
<td>KeyDerive</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>PKCS #5 V2.0 X9.63</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SymKeyGenerate</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>KeyDeriveSymKey</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key management</td>
<td>SymKeyUpdate</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SymKeyExtract</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SymKeyWrapSym</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AsymPrivateKeyWrapSym</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asymmetrical encryption</td>
<td>AsymEncrypt</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AsymDecrypt</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Signature</td>
<td>SignatureGenerate</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SignatureVerify</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key exchange</td>
<td>KeyExchangeCalcPubVal</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>KeyExchangeCalcSecret</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrapping symmetric Key using public Key</td>
<td>KeyExchangeCalcSymKey</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SymKeyWrapAsym</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key Update</td>
<td>AsymPrivateKeyUpdate</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AsymPublicKeyUpdate</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key Extraction</td>
<td>AsymPublicKeyExtract</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AsymPrivateKeyExtract</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrapping asymmetric private Key using public Key</td>
<td>AsymPrivateKeyWrapAsym</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compression</td>
<td>Compress</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Decompress</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 19.4 Configuration

We recommend that you use the DaVinci Configurator Pro configuration tool for easier configuration of the modules of MICROSAR Security. For more details, please see the separate product information.

### 19.5 Implementation

- Either the Crypto Abstraction Library or the Crypto Service Manager is offered according to specific needs
- The necessary security algorithms are available as a software implementation in both CPL and CRY
- Upon request, internal or external security hardware, e.g. per HIS-SHE, can be supported.
20 MICROSAR Multi-Core – The AUTOSAR Solution for Multi-Core Processors

The introduction of multi-core processors has also changed the design of the downstream AUTOSAR software. Individual AUTOSAR applications can now be distributed to different processor cores and can then be operated simultaneously. Crucial to success here is a good distribution with minimal synchronization losses, e.g. due to wait times. MICROSAR Multi-Core assists you comprehensively in this effort.

20.1 Overview of MICROSAR Multi-Core Advantages

>- It is fully integrated into the overall MICROSAR package and is an efficient solution between MICROSAR.OS and MICROSAR.RTE, because it utilizes:
  > A single OS configuration for all processor cores
  > Shared memory and efficient spinlocks for the RTE to shorten execution time and memory usage
>- It is closely intermeshed with the Vector MICROSAR Safe solution: All multi-core functions fit seamlessly into the MICROSAR safety concept
>- It implements BSW satellites on slave processor cores to assure time-optimized handling of service functions; available for individual modules
>- Very low configuration effort

20.2 Application Areas

MICROSAR Multi-Core is used for the operation of existing processor cores in applications in which:

>- the BSW runs on one of the existing cores and
>- any desired distribution of the application software to any of the available cores is possible.

The goal of this approach is to distribute the application to multiple processor cores with the greatest degree of CPU utilization. MICROSAR Multi-Core offers the following functionalities to achieve this:

>- It offers efficient mechanisms for all processor cores to utilize BSW services
>- It implements BSW satellites on slave processor cores as necessary
>- It uses standard RTE mechanisms to share the services of the basic software, which optimizes execution time and minimizes memory usage

This multi-core approach also works with the MICROSAR Safe solution.

20.3 Functions

20.3.1 RTE

In the multi-core field, shared memory is advantageous for the MICROSAR.RTE, because it minimizes execution time and memory usage. Spinlocks are used as necessary to assure data consistency. Wait times for the processor cores then only occur if they access the same memory range at the same time.

20.3.2 Operating System

Only a single OS configuration is required for all participating processor cores. One aspect that goes beyond the AUTOSAR standard is that MICROSAR Multi-Core offers efficient spinlocks, which allows to keep the data in generally accessible memory areas.

20.3.3 ECU Manager

The ECU Manager runs on the main processor core. Coordination between the cores is handled by a satellite on each of the other cores. This means that just a single configuration is required, even for the ECUM. It is valid for all processor cores.
20.3.4 Watchdog Manager

The Watchdog Manager also runs on the main processor core. On all other cores, there is a satellite which processes the application’s "checkpoint reached" function and synchronizes its status with the master. This assures that service functions of the WDGM are processed to optimize execution time.

20.4 Configuration

20.4.1 Configuration of the OS

In the configuration, the operating system is statically assigned to one processor core. This establishes which processor core acts as master and which ones act as slaves.

![Assigning processor cores in DaVinci Configurator Pro](image)

20.4.2 Configuration of the RTE

The RTE must know which software is running on which processor core. Only then is the RTE able to:

- assure consistency in the data exchange
- activate runnables across processor cores if necessary

In the configuration process, this involves mapping the individual runnables to OS tasks and subsequently allocating these tasks to an OS application. Each OS application is then mapped back to a specific processor core.

![Mapping SWCs, tasks and OS instances to the processor cores - schematic diagram](image)

We recommend the use of DaVinci Configurator Pro to make it easy to configure your ECU. You will find more details on this process in the separate product information.
20.5 Other Relevant Products for MICROSAR Multi-Core

- MICROSAR.RTE: The RTE controls communication between the processor cores
- WDGF and WDGM from MICROSAR Safe: there is a separate instance of the two watchdog modules on each processor core; this assures efficient API access.
- ECUM from MICROSAR.SYS: The ECU Manager assures that the system is started up properly and synchronously on all processor cores.
- You will also need hardware-specific modules such as the MICROSAR.OS operating system and a WDG driver.

20.6 Scope of Delivery

You get all of the individual modules of the MICROSAR package that are pertinent to multi-core projects with this product.
21 MICROSAR Variant Handling – Solutions for flexible configurations in AUTOSAR

In traditional task distribution for ECU development, the automotive OEM defines the communication and diagnostic interfaces. The supplier implements and builds the ECU according to these requirements. If its parameters need to be changed after delivery of the ECU, the supplier must develop a new version of the ECU. In the case of minor changes, this development chain costs unnecessary time and money. The development of ECUs with a relatively volatile BSW configuration in particular can benefit from the added flexibility of Post-Build Loadable.

Additionally, in the automobile there is an increase in the number of ECUs that can be installed in different variants; they are known as multiple ECUs. Your advantage is the use of a single hardware for the different application areas. This reduces hardware costs and effort in inventory management. On the other hand, it results in increased effort in software development and the management of software variants for production and service. For this use case a suitable Post-build selectable solution is available with the optional MICROSAR Identity Manager.

21.1 Post-Build Loadable – Post-Build Modification of BSW Properties

21.1.1 Overview of Advantages

- Many BSW parameters can be modified directly by the automotive OEM
- Flexible and spontaneous modification of the BSW in the framework of development and testing
- Post-Build Updates without the use of compilers (compiler licenses)
- Central resource management makes it possible to flexibly extend the BSW configuration by adding other elements (e.g. messages)
- Special extensions in DaVinci Configurator Pro ensure trouble-free Post-Build configuration
- Maximum of safety by consistency checks in the MICROSAR BSW and DaVinci Configurator Pro
- No change needed to the application architecture: Use of the extensively proven MICROSAR BSW according to AUTOSAR
- Post-Build Loadable is available as option for numerous BSW modules and functions

21.1.2 Application Areas

Post-Build Loadable permits modification of certain BSW properties from the areas of diagnostics and communication after the ECU has been built. Along with modifying parameters such as the CAN ID, send type and default values, new objects can also be introduced into the ECU at post-build time. For example, gateway routing tables can be extended by adding new messages or signals.

For the adaptation of BSW parameters at a post-build time point, only the MICROSAR delivery is needed. Neither the application nor compiler / compiler licenses are needed for a post-build update. Also there is no need for an adaptation of the application layer. Modifications to the BSW can be made by the automotive OEM – directly and in an uncomplicated way.

21.1.3 Familiar Configuration Process

The configuration process at the post-build time point is performed with the standard configuration tool Da Vinci Configurator Pro. This makes a powerful tool available for smooth synchronization with many different databases (DBC, System Description, ODX, CDD, etc.).

The MICROSAR Post-Build Loadable tool chain generates a HEX file of the updated BSW configuration in a single step, which can then be loaded into the ECU using standard flash tools.
21.1.4 Scope of Delivery "Post-Build Loadable"

- License for enabling the option in the configuration tool
- Tools for creating a Build Loadable Update
- Documentation

21.2 MICROSAR Identity Manager - The Post-Build Selectable Solution for Multiple ECUs

21.2.1 Overview of Advantages

- Efficient handling of ECU variants
- Less administration effort
- Reduced inventory costs
- Resource optimized implementation of the BSW configuration

21.2.2 Application Areas

The Identity Manager alternatively can take over two functions:

Example "Door ECU": **ECUs that perform nearly identical tasks**, differing only in their Rx and Tx PDUs or in their diagnostics and address in the network, can be implemented in a vehicle as one part with just one part number. In this case, a single ECU is developed and produced, which during start-up knows which one it is and where it is installed, and therefore also which functions it needs to fulfill. The buffer for the Rx and Tx PDUs can be completely overlaid if their layout is identical. The application accesses signals and data elements independent of its identity. No differentiation is necessary in the code.

Example "carry-over ECU": **Functionally similar ECUs in multiple model series** can be developed and built as one part. These ECUs contain the software for all of the car lines in which they are used and support each model series with a communication description that may be completely different. In this case, the ECU Extracts upon which the configuration of the AUTOSAR ECU
software is based, may differ very significantly for the different variants – for example they may exhibit a completely different signal layout or support a different number of buses.

![Carline Diagram](image)

Figure 48: One ECU performs different functions with use of the Identity Manager

### 21.2.3 Familiar Configuration Process

The configuration process of an ECU project that supports multiple variants is largely identical to a classic project without variants. Nonetheless, a separate set of system descriptions (Extract of System Description or legacy formats such as DBC, diagnostics descriptions (CDD, ODX)) is required for each variant.

### 21.2.4 Variant Selection

If an ECU supports multiple configuration variants, the variant to be used is selected in ECU initialization. The source of the variant information is user-definable, and it is implemented by the application. Such information might be connector coding or memory coding in flash.

### 21.2.5 Saving Resources

An ECU that contains multiple variants reduces the number of physical ECUs that need to be developed, produced and kept in stock. As there are multiple variants stored in the ECU now, the need for resources in the ECU will increase.

In code generation for the MICROSAR BSW, a number of optimization options especially focus on the most efficient memory use possible in RAM and ROM.

### 21.2.6 Scope of Delivery "Post-Build Selectable"

- License for enabling in the configuration tool
- BSW modules with Identity Manager functionality
- Documentation
22  MICROSAR J1939 – AUTOSAR Basic Software Modules Specially Designed for Heavy-Duty Vehicles

In this section, we introduce the BSW modules defined in the AUTOSAR architecture for communication in J1939 networks: The network manager J1939NM, the request manager J1939RM, the transport protocol J1939TP and the diagnostic module J1939DCM. These modules are part of MICROSAR.CAN and MICROSAR.DIAG.

Figure 49: The MICROSAR J1939 modules according to AUTOSAR 4.2

22.1  Overview of Advantages of J1939-Specific MICROSAR Modules

> Available for AUTOSAR 4.x.
> J1939NM: Address arbitration and communication control according to SAE J1939-81. Available as option is the support for fully-dynamic addressing and address monitoring for self-configuring ECUs.
> J1939RM can be used together with CANNM to combine address arbitration and Sleep/Wakeup.
> J1939RM: Direct interface to the J1939NM, J1939DCM, COM and RTE modules as well as CDDs. Full access to request with timeout monitoring and acknowledgment (SAE J1939-21).
> J1939TP:
> Full implementation of the J1939 transport protocol according to SAE J1939-81 with the variants BAM and CMDT (RTS/CTS), extended by support for the extended transport protocol according to ISO 11783-3 (ETP) and the FastPacket transport protocol according to NMEA2000.
> Automatic selection of the right transport protocol (direct, BAM, CMDT, ETP) based on message size and destination address, reception of messages via any of the transport protocols.
> J1939TP (BAM and CMDT) also for AUTOSAR 3.x.
> J1939DCM: Diagnostic module for heavy-duty vehicle diagnostics according to SAE J1939-73. Fully integrated solution with joint use of the saved diagnostic data of the DEM over J1939 and UDS diagnostics.
> Routing of messages based on their PGN, independent of source and destination addresses, including for consecutive messages.
Access to message addresses in CDDs.

> Code- and run-time-optimized by need-specific configuration.
> Inter-module configuration of all communication-specific software modules.

### 22.2 Application Areas

The application area of the J1939 modules is in handling communication in heavy-duty vehicles over CAN networks with the special features defined in the SAE J1939 standard. They are implemented in J1939-specific BSW modules and are supported by extensions in neighboring modules. In addition, MICROSAR.CAN can also be used to implement ISOBUS ECUs (according to ISO 11783) in agricultural vehicles and implements. For that purpose, J1939NM and CANIF were extended by functionalities for fully dynamic address arbitration and address tracking, and the ETP and FastPacket transport protocols were also implemented in the J1939TP. Also maritime use cases according to NMEA2000 can be supported by FastPacket and fully dynamic address arbitration.

### 22.3 Functions

The BSW modules for J1939 contain the functions defined in AUTOSAR 4.1, specifically:

> J1939NM: Address arbitration per SAE J1939-81 for J1939 networks with unchangeable addresses
> J1939RM: Supports request/acknowledgment protocol per SAE J1939-21
> J1939TP: J1939 transport layer with support for broadcast (BAM) and directed communication (CMRD or RTS/CTS) according to SAE J1939-21
> J1939DCM: Support of the most important diagnostic messages of SAE J1939-73, parallel operation with DCM

The following functionalities extend beyond the AUTOSAR standard and are available as options:

> J1939NM/CANIF: Extension that adds fully-dynamic addressing, automatic change of the own address in case of conflict and automatic adaptation of all of the addresses used to the current changes
> J1939TP: Extension that adds the ETP (per ISO11783-3 and -7) and FastPacket (per NMEA200) transport protocols
> Post-build loadable and post-build selectable: For details, please refer to the section "MICROSAR Variant Handling".

### 22.4 Configuration

We recommend our DaVinci Configurator Pro for convenient configuration. You will find more details in the separate product information.

### 22.5 Other Relevant MICROSAR Products for J1939

> DEM from MICROSAR.DIAG
> DET, BSWM, ECUM and COMM from MICROSAR.SYS
> MICROSAR XCP enables measuring and calibrating according to ASAM XCP. The module was optimized, especially with regard to its use in conjunction with CANoe.XCP, CANoe.AMD and CANape. MICROSAR XCP contains the related transport layer CANXCP for CAN ECUs.

Beyond the AUTOSAR standard, MICROSAR XCP supports generic readout of measuring objects. As a result, no addresses must be defined and updated in the a2l file. Data from any version or variant can be extracted with an a2l file independent from the MCU build. The generic readout feature requires usage of CANoe.AMD or CANape as XCP tool.

> For safety reasons, very often it is not allowed to keep the measurement and calibration interfaces active in serial production projects. The module VX1000If allows to keep the VX1000 measurement and calibration hardware driver in the BSW also in serial production, but in a deactivated mode. Through an API the VX1000 driver can be released again for check and development purposes. Delivery of the module must take place within a MICROSAR SIP to receive approval for this use case in serial production. The activation of the VX1000 driver within a serial production project at runtime however is also not allowed when using the module VX1000If.
23 MICRO SAR VTT – Virtual Integration with vVIRTUALtarget Basic

The process for developing an ECU typically is begun by creating the individual software modules (SWC). However, because of the continual trend towards shorter project times, there is generally insufficient time to sequentially create all of the necessary SWCs first and then test their interaction with one another and finally test their interplay with the basic software on the target hardware.

Numerous projects today are characterized by the need to start test runs in a very early project phase - sometimes in a phase in which the later target hardware has not yet been fully defined.

vVIRTUALtarget (VTT) offers the right solution for this situation. It is a runtime environment in which ECU software can be executed without requiring access to a real ECU. This environment enables decoupling of test execution from the real hardware and from the existence of the basic software. This, in turn, leads to substantial time advantages. The same configuration can be used for both the target hardware and the environment of vVIRTUALtarget:

After development of the SWCs and their system integration, the ECU software and its parameterization are at an advanced stage. Along with developing the application, another focus is on configuring the basic software. Here, vVIRTUALtarget enables testing that is independent of the target platform.
23.1 Overview of vVIRTUALtarget Advantages

> Early integration and testing of AUTOSAR 4 software modules.
> Development of the ECU software independent of the availability of the real target hardware.
> Dual target option enables parallel integration on virtual and real hardware.
> Greater depth of testing by additional testing in the virtual environment.
> Microsoft Visual Studio is used as a convenient development and debugging environment.
> Use of vVIRTUALtarget Basic does not change workflow in ECU development.
> Improved simulation of interruptions by interrupts and task preemption.

23.2 Application Areas

MICROSAR vVIRTUALtarget can be used wherever ECU integration needs to be accelerated. A CANoe license is needed to stimulate and measure communication and I/O interfaces.

23.3 Functions

vVIRTUALtarget Basic is intended for ECU integration and enables testing of the entire stack (application, SWCs, BSW) in a virtual environment. VTT offers the following functions:

> Early testing of the entire stack, even if the real ECU is still unavailable.
> Use of the same MICROSAAR configuration on the target hardware and in vVIRTUALtarget.
> More convenient testing of the ECU software in the virtual environment
> Support of AUTOSAR-conformant 3rd party modules
23.3.1 vVIRTUALtarget in conjunction with CANoe

Based on the current MICROSAR configuration, the vVIRTUALtarget tool generates a Visual Studio project in which a “System under Test” Windows DLL (SUT DLL) can be very conveniently generated from the BSW modules and the application. Debugging can be performed afterwards. Our solution vVIRTUALtarget is developed for Microsoft Visual Studio 2013 (Express & Professional).

Two options are available for debugging: One option is to debug directly in the runtime environment of vVIRTUALtarget. Another approach is to use CANoe as the runtime and debugging environment. This lets you stimulate and visualize the buses and I/O interfaces of the generated SUT DLL.

23.3.2 vVIRTUALtarget in conjunction with MICROSAR.MCAL and MICROSAR.OS

The BSW modules MICROSAR.MCAL and MICROSAR.OS were ported for the use of vVIRTUALtarget. The MCAL interfaces and behavior conform to AUTOSAR 4 specifications here. The MICROSAR.OS that is available for vVIRTUALtarget implements the SC1 “Scalability Class” without multi-core extensions.

The rest of the BSW modules do not require that any modifications be executed in the vVIRTUALtarget environment. The BSW modules are the same as those used in the real ECU.
23.4 Workflow

The workflow for configuring the BSW modules in the vVIRTUALtarget area does not differ from the approach taken with the real hardware platform. The only exception is that the MCAL modules specific to the vVIRTUALtarget may need special settings, if they cannot be automatically derived by the DaVinci Configurator Pro configuration tool.

![Workflow diagram]

Figure 54: MICROSAR workflow extended by the workflow of vVIRTUALtarget

23.5 Configuration

For convenient configuration, we recommend our DaVinci Configurator Pro. You will find more details in the separate product information. The vVIRTUALtarget runtime environment is configured via the Option.VTT, which is available for DaVinci Configurator Pro.

23.6 Product Components

The modules of vVIRTUALtarget are supplied as a Software Integration Package (SIP). The customer may choose to have the SIP supplied just for the vVIRTUALtarget platform (just vVIRTUALtarget MCAL), or a dual-target SIP can be supplied together with drivers for a customer-defined real hardware platform (real MCAL & vVIRTUALtarget MCAL).

23.7 System Requirements

Our product vVIRTUALtarget is a 64 bit application and therefore requires a 64 bit system.

Further relevant MICROSAR Products for VTT

The "AUTOSAR Evaluation Bundle VTT" is available to get to know virtual integration and for testing its functionalities. For further information, please see chapter "AUTOSAR Evaluation Bundle".
24 MICROSAR.SIP and MICROSAR.EIP – A Quick Start to Your AUTOSAR Project

The Software Integration Package (SIP) and the Extended Integration Package (EIP) from Vector give you a decisive advantage in developing your ECU software: we test your software package before delivery, and you can put the entire package into operation within just a few days.

24.1 Software Integration Package (SIP)

MICROSAR.SIP is a standard delivery item, and it focuses on the broadest possible range of use for your stack. It optimizes the usability of our delivery to you, even when constraints are altered slightly. Our portfolio in brief:

> Check the OEM-specific aspects of the BSW, including SWCs and the associated tool chain (e.g. support of OEM data formats for communication and diagnostics) and assure conformance to OEM specifications.

> Check the project-specific combination of BSW components and features for a defined ECU use case.

> Check the package under abstract constraints: both the configuration and the initial database can still be changed over the course of your project. This even occurs very frequently in practice. Our goal is to implement your MICROSAR package so that it covers a wide range, i.e. to enable implementation of as many additional variants to the initial configuration and database as possible.

> Check the package under project-specific constraints (microcontroller derivative, compiler/linker version and compiler/linker options). Here we look at the specific constraints of your project as closely as possible, so that the product can be integrated smoothly for you. For example, we select an evaluation board with a suitable device from the microcontroller product line you have selected, and test the software package on it. The goal here is to ensure that our delivery to you can be run on as many devices as possible from the preselected processor line.

> Offer management specific to delivery of the ordered MICROSAR modules in the context of our configuration management. The option for after-sale redeliveries is made sure for a time period of 10 years.

> Actively issue reporting at regular intervals.

24.1.1 Application Area of the SIP

MICROSAR.SIP is a fixed component of every MICROSAR delivery, regardless of whether it is a prototype, beta, update or production delivery. The performance range is adapted to the purpose of the SIP and the context in which it will be used.

Via a questionnaire, we catalog your requirements in as much detail as possible in advance of delivery. Afterwards we custom-build your Software Integration Package as individually as possible on this basis.

SIP Maintenance gives you active issue tracking beginning with the initial product delivery; it informs you of known errors and potential work-arounds. In addition, SIP Maintenance includes one SIP update delivery per calendar year, and Extended Maintenance even gives you two SIP update deliveries per year.

By offering the update SIP, we actively help you to introduce a new SIP and/or a new database into your project and to test it.
## 24.1.2 Optional extensions for the SIP

The following additional services are available with delivery of the basic SIP product:

**Automotive OEM-independent extensions** give you an easy way to start by having our integration team implement key tasks even before your MICROSAR Stack is delivered to you:

> "Start Application" extension:
  - Incorporates real project-specific communication descriptions
  - Adapts the database for communication and diagnostics as necessary
  - Executes a fundamental wakeup and shutdown of the ECU
  - Configures the OS with at least one task
  - Transmits periodic messages and receives on all existing communication channels
  - Illustratively executes a diagnostic service (Request/Response) and an fault memory operation
  - Configures an NVRAM block with read/write access to the internal NVRAM
  - Periodic watchdog trigger

Prerequisite for the execution of these tasks is that the required MICROSAR modules are part of the delivery.

> "Customer Hardware" extension:
  - Puts the ECU into an operationally ready state including CPU clock, PLL and internal watchdog
  - Configures the network transceiver
  - Executes the general delivery test at Vector on real customer hardware

**Automotive OEM-dependent extensions** require close interaction between you and our team; they handle extended ECU dependencies such as:

> **OEM application**: incorporates modules specially developed by the automotive OEM that supplement the AUTOSAR basic software

> **OEM test**: executes special tests that incorporate OEM modules and generates all test reports
To find out which options are available for your automotive OEM, please refer to the OEM-specific product information sheet that is automatically provided to you with our quotation.

24.2 Extended Integration Package (EIP)

Building upon the MICROSAR.SIP, in conjunction with extended SIP extensions (see above), MICROSAR.EIP assists in follow-up activities after an initial delivery. It offers crucial support in achieving a quick and comprehensive startup. The goal here is to pass the first bench test at the OEM.

Vector employees handle preparations that are normally conducted on-site at the customer. We perform this service at a fixed price and in the framework of task planning that is detailed in an agreement with you. We can:

- Place a start application in your ECU with a specification-conformant configuration and using the databases (communication and diagnostics) that are relevant to the project
- Perform other project management jobs such as coordination via regular consultation meetings and project reporting
- Create release planning tailored to your needs
- Perform on-site startup together with you at the end of the EIP service package
- Execute test cases related to the BSW that are required by the OEM

The results of these prepared activities are then also part of the delivery:

- Your basic software package including configured start application
- Release notes
- Write-up of related test reports

Figure 56: The Extended Integration Package and its interaction with the SIP options
24.2.1 Application Area of the EIP

MICROSAR.EIP represents an extended service with the goal of submitting a sample; it is made available for the first bench test at the OEM very soon after delivery. Therefore, the EIP is an option that may be of special interest when you are performing your first project in the AUTOSAR field or when there are special project conditions in the context of an initial delivery:

- You are working on your first project in the AUTOSAR field and want extended support
- You are unfamiliar with a certain OEM, and so you want to take advantage of Vector’s experience
- You need to add supplemental services (e.g. OEM-specific components or 3rd-party modules)
- If test verification is necessary under precisely your constraints (ECU, configuration)

MICROSAR.EIP is offered for select OEMs. If you are interested, our service team would be glad to explain this offered service and whether, and to what extent, it might support the OEM relevant to your project.

In the EIP service, we also determine your requirements in as much detail as possible by questionnaire. These requirements then serve as a foundation for all other activities:

First, as in the case of a SIP, we put together your basic software. Then an employee of our service team configures and tests the overall package under precisely your production conditions and starts up your package at your business site.

The service package is rounded out by documentation of the installation process, generation of test reports and subsequent support by Vector.

24.3 Configuration

We recommend the DaVinci Configurator Pro product for convenient configuration of your MICROSAR delivery. For more details please see the separate product information.

24.4 Other relevant MICROSAR Products for SIP and EIP

Our services are supplemented by a broad line-up of support options that assist you in successful project startup, project migration and project review. For details, please refer to the separate "Services" product information document that is available online: http://vector.com/portal/medien/cmc/info/EmbeddedServices_ProductInformation_EN.pdf.
25 AUTOSAR Evaluation Bundle – The complete package for the evaluation of AUTOSAR basic software and tools

The AUTOSAR Evaluation Bundle is a comprehensive package of OEM independent AUTOSAR basic software (MICROSAR) and the tools DaVinci Developer and DaVinci Configurator Pro. This package lets you develop your first ECU software with AUTOSAR-conformant software architecture. You get an in-depth look into the AUTOSAR world – from the design and configuration process to implementation of the actual basic software. You can obtain OEM-specific BSW modules, e.g. for diagnostics, in our MICROSAR Prototype SIP (see end of this chapter).

25.1.1 Overview of advantages

- Tools and basic software in production quality according to AUTOSAR 4.x or 3.x to evaluate the Vector solution for AUTOSAR
- Enables realistic evaluation of execution time and memory requirements for your ECU project
- Available for many different microcontrollers
- Quick way to train in AUTOSAR with a detailed sample project
- Support of both AUTOSAR-conformant files and conventional description files

25.1.2 Application Areas

The AUTOSAR Evaluation Bundle supports both the automotive OEM – in evaluating AUTOSAR processes and methods – and suppliers in creating a first AUTOSAR-conformant ECU software. Since the tools and basic software are at a level of production maturity, you can reliably use the Vector solution to evaluate AUTOSAR with regard to:

- Efficiency of the basic software
- Integration of the tools in your development environment
- Potential uses of AUTOSAR concepts in your application area

The AUTOSAR Evaluation Bundle also provides an optimal foundation for initial developments of AUTOSAR-conformant software components (SWCs) for service providers who focus on the application level.

25.1.3 Functions

The AUTOSAR Evaluation Bundle contains the tools and embedded software from Vector for creating a complete set of AUTOSAR ECU software, which consists of software components (SWCs), Run-time Environment (RTE) and basic software (BSW). The DaVinci tools are tailored to AUTOSAR and simplify your work in designing complex AUTOSAR applications. As input for configuring the MICROSAR software, you would use an “ECU Extract of System Description” (AUTOSAR XML) or as an alternative a conventional network description file (DBC, FIBEX, LDF).

- The **DaVinci Developer** tool gives you an easy way to generate AUTOSAR-conformant ECU applications. Using the graphic editor, you can describe your AUTOSAR software components quickly and clearly and define their interfaces. The SWCs serve as a basis for the RTE configuration process, which you also perform with DaVinci Developer.

- The **DaVinci Configurator Pro** tool is used to configure the basic software modules and the RTE. You can use the convenient and intuitive user interface to adapt parameter values for your ECU project.

- The **CANdelaStudio** tool from Vector is also available. It lets you define diagnostic data for your networks and ECUs. You can export this data via standard formats, and use the data in automatic configuration of the MICROSAR diagnostic basic software.
The AUTOSAR Evaluation Bundle is available for AUTOSAR 4.x and 3.x. The included MICROSAR basic software modules efficiently and flexibly implement all functions of the related AUTOSAR Releases. They also contain many extensions that go beyond the standard.

### 25.1.4 Included BSW Packages

The following table gives you an overview of the individual MICROSAR bundles contained in the AUTOSAR Evaluation bundle. For a complete description of the individual bundles, please refer to the separate chapters in this document.

<table>
<thead>
<tr>
<th>In EVAL bundle</th>
<th>Available options</th>
</tr>
</thead>
<tbody>
<tr>
<td>MICROSAR.OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt; Implementation of the &quot;Scalability Class&quot; SC1 is standard</td>
</tr>
<tr>
<td></td>
<td>&gt; SC2-SC4 are available as options, provided that they are supported by the processor</td>
</tr>
<tr>
<td></td>
<td>&gt; Multi-core is available as option</td>
</tr>
</tbody>
</table>

| MICROSAR.COM    | |
| MICROSAR.CAN    | |
| MICROSAR.LIN    | |
| MICROSAR.FR     | |
|                 | > Modules from the product segment as chosen by the customer |
In addition, the following options can be ordered with the evaluation bundle:

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MICROSAR.EXT</td>
<td>Modules for driving external chips</td>
</tr>
<tr>
<td>MICROSAR Safe</td>
<td>Complete solution for safety-related functional software according to ISO 26262</td>
</tr>
</tbody>
</table>

### 25.1.5 Special Functions

DaVinci Developer has an import/export interface for AUTOSAR XML files. This interface lets you exchange design and configuration data. For example, you might use it to integrate AUTOSAR software components in an ECU you have developed in a model-based approach using tools like MATLAB® Simulink®.

All MICROSAR products conform to

- “Implementation Conformance Class” ICC3 and
- “Configuration Conformance Class” CCC 2.

### 25.1.6 Additional Included Features

- Sample application in source code, and a detailed guide on its use
- AUTOSAR Training at Vector

### 25.1.7 Further Options

The AUTOSAR Evaluation Bundles CAN, LIN, IP and FlexRay may be used with one another in any combination.

Upon request, Vector can support you with extensive MICROSAR Coaching at initial startup and during integration of the MICROSAR basic software in your application. Vector can also provide coaching at your business site.
25.1.8 Available Hardware Platforms

The AUTOSAR Evaluation Bundle is available for the most commonly used 16-bit and 32-bit hardware platforms. Due to the hardware dependency of the MICROSAR.MCAL modules and the MICROSAR.OS, binding statements cannot be made without specific processor device numbers. The Vector Sales Team would be glad to provide this information.

Within an Evaluation you can save noteworthy resources by using the VC hardware from Vector. Additional modules within the MICROSAR basic software enable a quick and easy setup of your ECU project. The VC121 hardware is supported by the following contents in the MICROSAR BSW:

- VC121 SW LIB
- MCAL VC121
- Demo VCx
- SIP integration
- Flash Bootloader (included in the delivery of the VC121 hardware)
- Training-Voucher for one person

For details on the VC hardware please refer to the separate product information in our marketing portal at http://vector.com/vi_universal_controller_vc_en.html. Further variants of the VC hardware are in development.

25.1.9 MICROSAR Prototype SIP

If you need software for the prototype phase of a specific OEM project that goes beyond pure evaluation purposes, we recommend using our MICROSAR Prototype SIP (Software Integration Package). Please contact us for further information.
26 Additional Information

For further information on our products and our configuration tool DaVinci Configurator Pro, please see our internet site http://vector.com/vi_embedded_software_en.html.
Get more Information!

Visit our Website for:

> News
> Products
> Demo Software
> Support
> Training Classes
> Addresses

www.vector.com