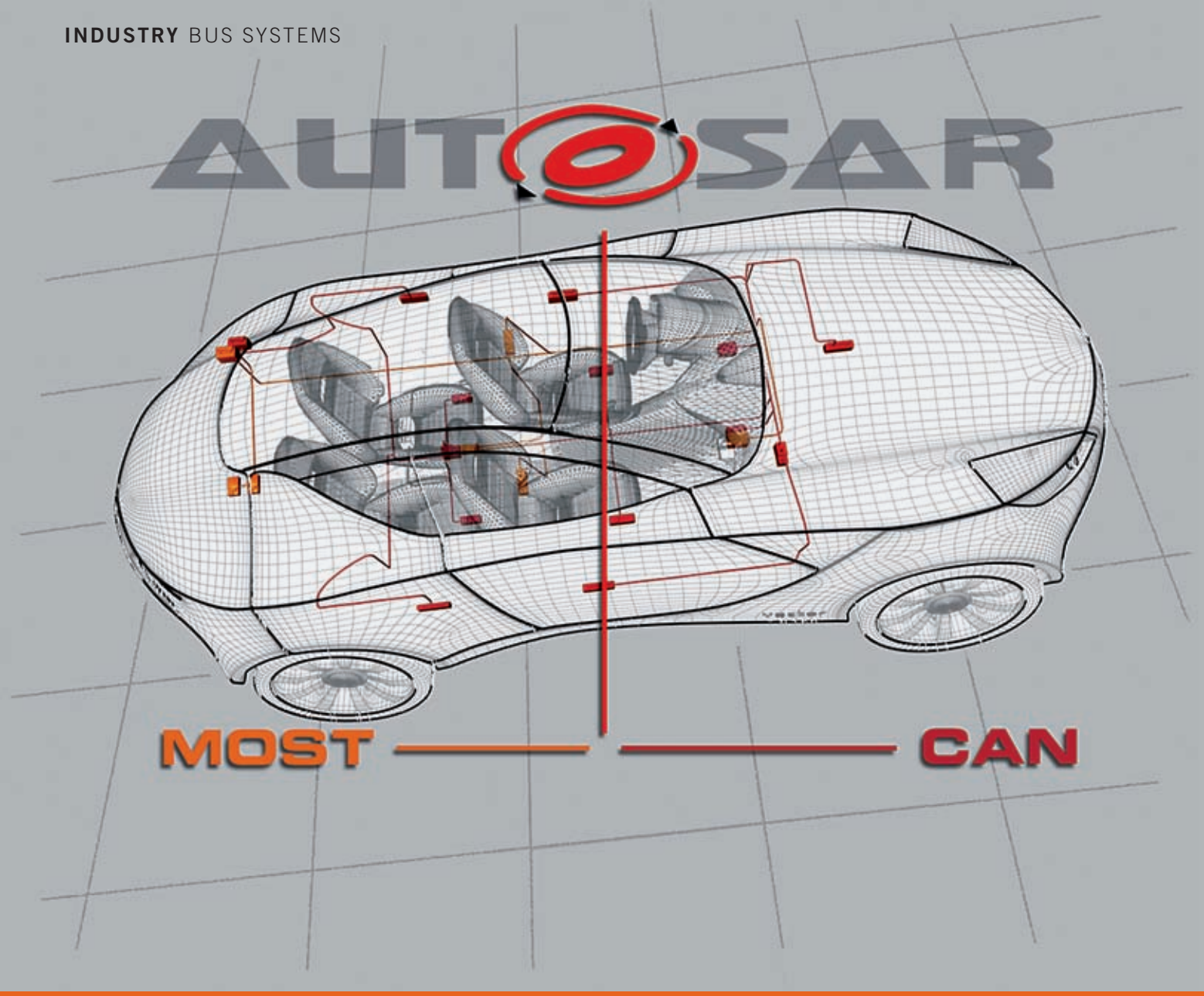



# IMPLEMENTING A CAN-MOST GATEWAY WITH AUTOSAR BASIC SOFTWARE

The gateway architecture defined in Autosar makes it possible to interconnect different bus systems. However, at this time Autosar only defines couplings of CAN, LIN, Flexray and Ethernet buses. Despite this limitation, there are various solutions for implementing a CAN-Most gateway based on Autosar. But which is the right one? Vector Informatik presents three approaches and describes their respective advantages and disadvantages.

AUTHOR

**DIPL.-ING. (FH) PATRICK MARKL**
is Senior Software Development Engineer responsible for concept development in the Embedded Software Product Line at Vector Informatik GmbH in Stuttgart (Germany).

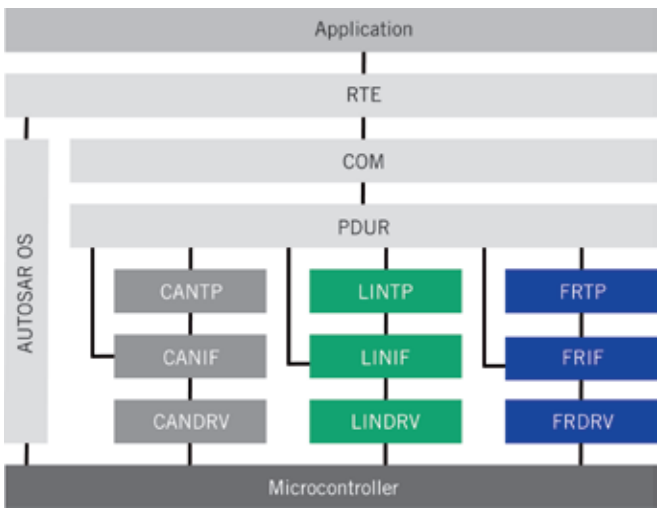
## FUNDAMENTALS

Networking in modern vehicles is based on several different bus systems. CAN, LIN and Flexray are used in the convenience and powertrain areas. Multimedia and infotainment ECUs are networked primarily over the Most bus. The exchange of information between networks is the task of gateway ECUs. They exist in various forms – ranging from relatively simple gateways that only exchange minimal information between buses of the same kind to central gateways that interface to several buses of different types.

Autosar offers ECU developers a solid software basis for interfacing their ECUs to CAN, LIN, Flexray and Ethernet buses. Gateway tasks are handled by various software modules of the Autosar architecture depending on the specific functionality that is required.

## PDU GATEWAY FOR ROUTING ENTIRE PDUS

The PDU gateway is a part of the PDU Router (PDUR). This gateway routes entire data packets, Protocol Data Units (PDUs), from one network to another, ❶. This type of routing assumes that the PDUs are defined identically on the source network and target network. This means that the PDUs must match in length and content. However, it must be noted that PDU routing is a spontaneous routing method: The PDU is immediately routed as soon as it is received. This means that it is not possible to convert the cycle time or send type. In some cases, however, such a conversion is necessary. Then routing must be performed by means of a signal gateway.




❶ Rough architecture of an Autosar ECU

## SIGNAL GATEWAY

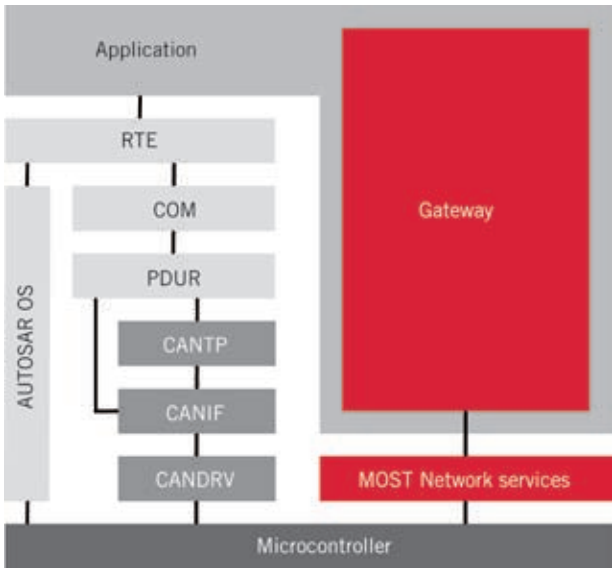
Often only individual signals of a PDU must be transmitted to another network, because only these signals are needed on the target bus. A second reason is that the source and target PDUs may be laid out differently, and a signal may be located at different positions. In signal routing, the received PDUs are first disassembled into their individual signals, so that they can then be copied to one or more target PDUs. Along with changing the composition of the PDUs, it is also possible to change the send type and cycle time. Routing parameters can also be modified flexibly. Signal routing is performed by the Communication Module (COM). This module is located in the Autosar Stack above the PDUR, ①.

## TRANSPORT PROTOCOL GATEWAY

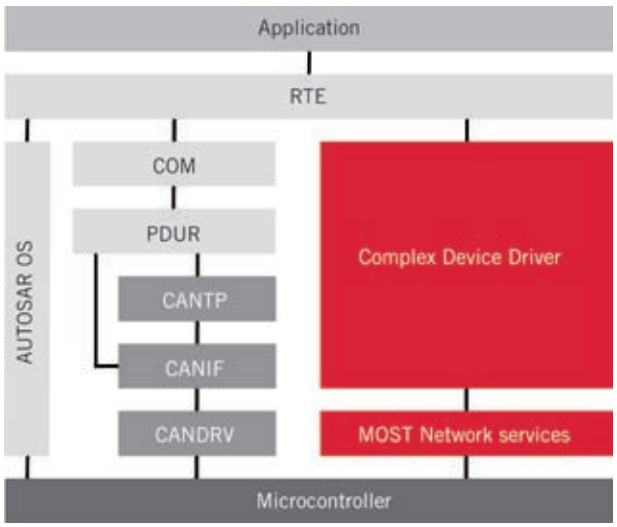
Another task of the PDUR is to route transport protocol data. This routing mechanism might be used, for example, to transmit an ECU's diagnostic data from one network to another. This involves receiving and transmitting the data over the transport protocol (TP). So, this type of routing is performed above Layer 4 of the ISO/OSI reference model, and it allows conversion to different addressing methods and various bus systems. TP data is routed by two mechanisms: One method uses the "store and forward" principle, the TP gateway receives the source data in its entirety before sending it to the target network. However, this mechanism is very memory-intensive and leads to longer latency times in routing. To keep delays and RAM requirements as low as possible in the gateway, the TP gateway also supports what is known as "on the fly routing." In this case, the gateway does not wait to receive all of the TP data first, but instead begins to send out the data to the target network at an early time point. So, it receives and transmits simultaneously.

## PROPERTIES OF THE MOST BUS

The Most bus has a ring topology. The ring consists of a Master ECU – also




❷ CAN-Most gateway as a hybrid solution




❸ CAN-Most gateway as a complex device driver

known as the timing master – that provides the system clock, and other ECUs known as timing slaves. Usually the timing master in the vehicle is contained in the Head Unit of the infotainment system. Currently, three baudrates are possible on the Most bus: 25, 50 and 150 Mbit/s. Most uses an optical physical layer. In addition, an electrical physical layer was specified for Most50. An ECU's software is interfaced to Most over the so-called Network Services. Network Services is a protocol stack standardized by the Most specification.

On the Most bus, data is transmitted over the synchronous and asynchronous channels, as well as the control channel. The synchronous channel is used to transmit video and audio data. Larger packet data whose transmission is not periodic, e.g. Ethernet tunneling or navigation data, is transmitted over the asynchronous channel. The control channel, whose bandwidth is vastly reduced, compared to the two other channels generally transports commands and status data.

The choice of the mechanism used to convert data from other networks to Most depends on the data itself. Functional data (sensor data, states, etc.) is usually just a few bytes long; on the Most bus, it is generally sent on the control channel. In the case of diagnostic data, which is also transmitted in larger blocks on CAN, it is advisable to send it on Most using the Most High Protocol (MHP). The MHP operates on the asynchronous channel, which has greater bandwidth than the control channel, which enables faster provision of the data packets to a specific receiver.

### CAN-MOST GATEWAY AS HYBRID SOLUTION

A CAN-Most gateway can be realized as a hybrid solution. In this context, a hybrid solution is a gateway architecture that uses a dedicated communication stack for each bus and implements the actual gateway by additional software layers. In this case, the specific gateway code must be contained in the application, ❷. Often, existing gateway software modules cannot be used in this approach, since interfacing to the other communication stacks is not supported.

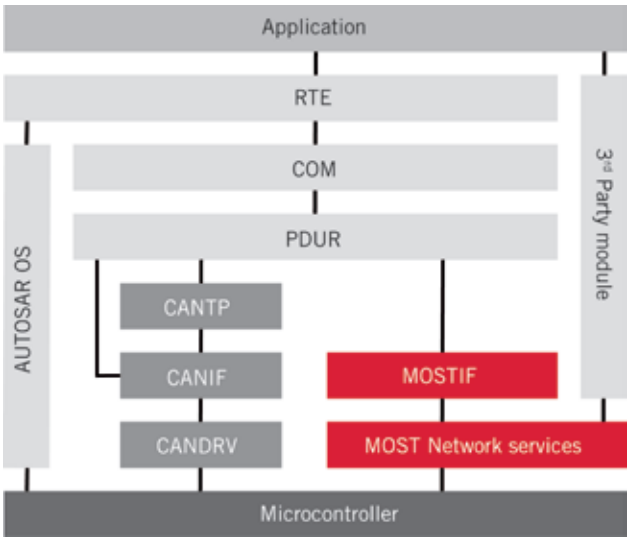
To interface the different communications stacks, hybrid solutions frequently also require wrapper software, which leads to greater propagation time. Another disadvantage is the different configuration tools that need to be used for the communication stacks. Usually, the stacks can only be configured separately, so routing rules cannot be considered. This results in additional work effort whenever the communication descriptions are changed.

### CAN-MOST GATEWAY BY COMPLEX DEVICE DRIVER

The Complex Device Driver (CDD) described in Autosar represents a software module that allows direct access to the Basic Software Modules (BSW) and the microcontroller hardware. It also makes it possible to integrate microcontroller interfaces that are not defined in Autosar. Implementation of a CAN-Most gateway as a CDD, ❸, simplifies interfacing between the gateway and the drivers for CAN and Most. As in the hybrid solution, any changes to routing rules must be handled afterwards at great effort. Another disadvantage of these two approaches is the redundant implementation of gateway functionality in the ECU and the additional memory required for this. Essentially, the CDD approach is comparable to the hybrid solution.

### CAN-MOST GATEWAY WITH AUTOSAR ARCHITECTURE

In its software architecture, Autosar has solved the described problems in creating




❹ Implementation of a CAN-Most gateway with the Autosar modules PDUR and COM

multibus gateways. So, it is natural to want to use this approach to develop a CAN-Most gateway in the same way as a CAN-LIN or CAN-Flexray gateway. To accomplish this, it is necessary to extend the Autosar stack by adding a Most bus interface.

According to the Autosar specification, the higher layers to communication buses are interfaced via two BSW modules. The first is the hardware-specific driver, which controls accesses to the specific communication controller, and the second is an interface module, whose task it is to route the data of the underlying driver, independent of hardware, up to the higher layers. For example, in the CAN bus, these are the two BSW modules CAN Driver (CANDRV) and CAN Interface (CANIF). A similar driver and interface architecture is also used on the LIN bus and Flexray bus. Above the interface layer, the bus systems are abstracted to the extent that the PDUR and COM modules can implement gateway functionalities hardware-independently.

Vector has chosen this architecture implementation as the best approach. Interfacing the Autosar stack to the Most bus requires a Most Interface (MostIF) and a Most Driver. Instead of implementing a Most driver, existing Most Network Services were accessed, similar to the way Autosar 4.x provides for the Ethernet interface. No MostTP module was used, since this functionality is already included in Most Network Services. In the framework of a project, Vector developed the MostIF module, which is part of the Microsar-Most package. The MostIF module is placed over the Most Network Services, ❹. The PDUR from Vector was extended to interface to the MostIF in addition to interfaces for CAN, LIN and Flexray. The COM module – responsible for signal routing – was adopted without changes.

The architecture presented here allows the application to access Most Network Services, so that existing Most software modules can continue to be used. The Vector Autosar stack has long supported the CAN, LIN and Flexray bus systems and more recently Ethernet as well. The gateway mechanisms available for this are re-used for the Most bus. It is very easy, for example, to implement sending of data by the Most notification mechanism in the Autosar environment via the trigger-transmit interfaces. TP data received by the CAN bus can be sent on Most over the MHP or over the control channel. The specific mechanism is selected at the time of configuration. Dynamic management of the TP buffers is handled by the PDUR.

## AUTOSAR-CONFORMANT CONFIGURATION

The Autosar BSW is configured with tool support, and the configuration data – e.g. the communication description of the ECU – is saved in an ECU-specific ECU Configuration (ECUC) description file. In the case of a gateway, the routing relationships are also included in this file. Additional Most-specific attributes are saved in extensions to the ECUC file. Describing the gateway configuration in a single file prevents errors to be introduced by any format conversions or even manual merging of parameters.

## ADVANTAGES OF IMPLEMENTATION BY AUTOSAR ARCHITECTURE

The Autosar architecture makes is possible to develop universal gateways to interconnect CAN, LIN, Flexray, Ethernet and Most buses. When a Most interface is integrated in the Autosar architecture, wrapper layers are unnecessary, and this results in better propagation time and lower memory usage. The presented architecture also has a central routing unit (PDUR and COM), which does not require any additional routing code in the application. This central routing unit enables consistent configuration of all routing relationships with one tool, and the configuration data is saved within a single description file.

Another advantage of the Autosar architecture is the post-build process. This offers a way to change the BSW configuration without re-compiling the application. In this process, the configuration data of the BSW is replaced (overwritten) in the ECU's flash memory. The ECU code is not changed by the post-build process. This provides a way to change the gateway's routing description during development or even in production use.

## OUTLOOK

The described implementation of a CAN-Most gateway with an Autosar architecture focuses on routing messages of constant length, which is the same on the CAN bus. It is conceivable that, in the future, the Most stack might be extended to include additional functionalities. This includes routing data of variable size – e. g. a list with different numbers of telephone book entries. Similarly, Most network management tasks could be handled by Autosar modules.

With the Ethernet interfacing available in Autosar, it will be possible to develop very powerful gateways with a Most connection and thereby utilize the advantages of the Autosar architecture.

**REFERENCE**

**[1]** Grzemba, A.: Most – Das Multimedia bus system for den Einsatz im Automobil ("Most – The Multimedia Bus System for Use in the Automobile")