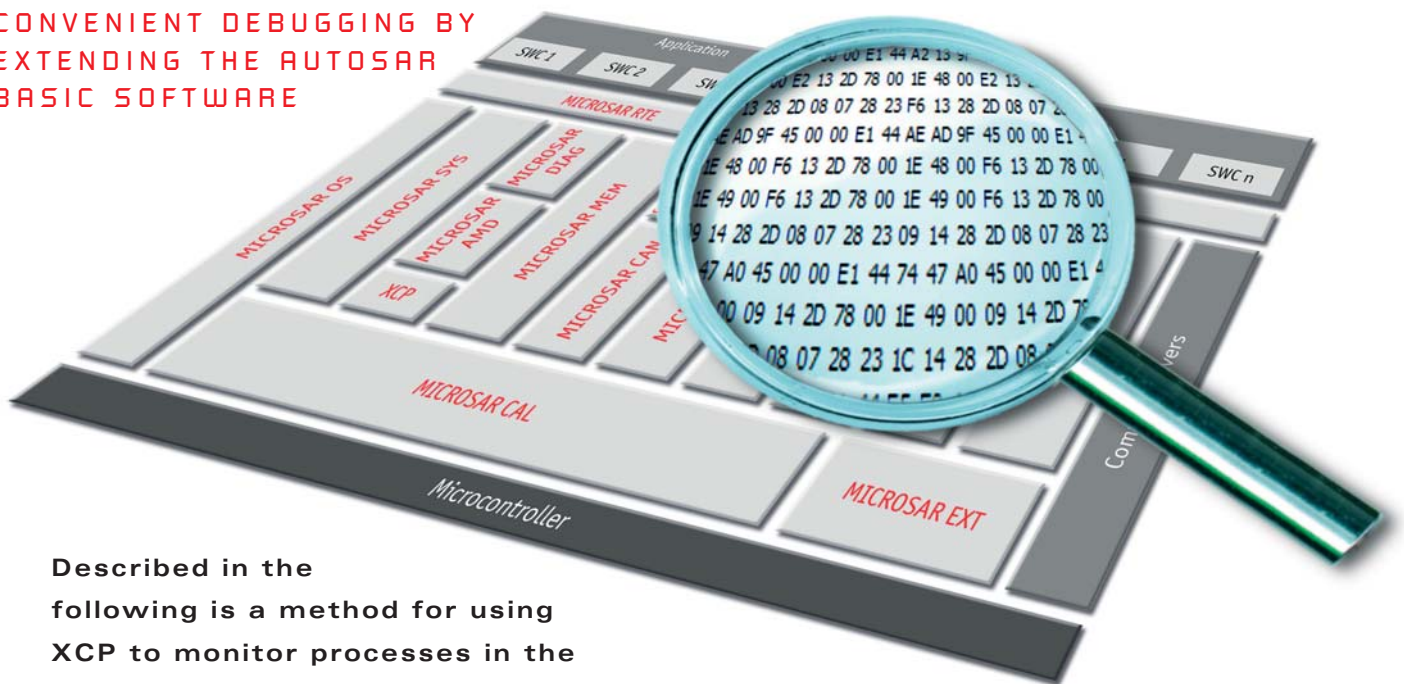


Analyzing AUTOSAR ECU software with XCP

CONVENIENT DEBUGGING BY
EXTENDING THE AUTOSAR
BASIC SOFTWARE



Described in the following is a method for using XCP to monitor processes in the AUTOSAR basic software (BSW) and in the software components (SWC). Certain features must be added to the basic software for these measurement purposes. Specialized extensions for the analysis tool allow for efficient debugging and easy evaluation.

When debugging in an environment of networked ECUs, debuggers are often of limited use, especially if errors only occur sporadically or in the test vehicle. In such cases, the proven measurement and calibration protocol XCP offers useful services. Many test tools available today support the approach of testing individual ECUs using a remaining-bus simulation. They let users verify required functions in an early stage and independent of the ECU network. This results in very high quality at an early stage of software development. However, testing options are reduced considerably when an ECU is installed in a test bench or in the test vehicle, or when a function is distributed over several ECUs via a network. It is no longer so easy to localize the errors that occur. The causes of this are:

- An ECU is installed in such a way that it is difficult to access.
- No more connection terminals are available for debugging.
- The debuggers for the different ECUs do not condition the data uniformly.
- The test tools cannot be housed in the vehicle.

Besides physical conditions that restrict debugging in the vehicle, there is another factor as well: some errors only occur sporadically. Determining the causes of sporadic errors is not always easy in the laboratory either.

Requirements for debugging

To find the cause of an error, the values of various software variables are typically checked. It is also important to simultaneously check the variables of multiple software mod-

ules in reference to a trigger point. This lets users monitor inter-module consistency conditions. Key examples of this are the „manager modules“ whose state machines are interdependent, e.g. the AUTOSAR „Bus State Managers“ and „Communication Manager“ (ComM). Similarly, the test engineer must determine which application requests caused the ECU to enter the measured states. It is also necessary to observe causative chains beyond module boundaries, e.g. to trace a signal path from bus level to application level. Moreover, time stamps are necessary to reconstruct the error history in the analysis.

XCP in AUTOSAR 3.x projects

The AUTOSAR 3.x standard does not specify any mechanisms for remote debugging. For years now, however, the „Universal Measurement and Calibration Protocol“ (XCP) has been successfully used to measure and calibrate ECUs in vehicle development, and it covers the requirements cited above very well. XCP provides mechanisms for reading and writing variables via bus systems. Based on DAQ (Data Acquisition) lists, multiple variables can be measured consistently and with a common time stamp. Dynamic DAQ lists make it possible to reconfigure the data records to be read out during the measurement. They offer a way to optimally utilize the available bandwidth of the communication bus.

Typically one or more A2L files are used in conjunction with XCP measurements. An A2L file contains information on the relevant measurement objects in the ECU. Required for each of these objects is information such as memory address, data type, symbolic interpretation and conversion rules. Along with the description of parameters for communication between the XCP Master (e.g. analysis tool) and

the XCP Slave (ECU), an A2L file also contains a hierarchically structured representation of the measurement objects. XCP is a powerful protocol for debugging with its many different ways to access internal ECU variables and convenient configuration via an A2L file.

The XCP protocol is implemented in a dedicated BSW module, which however is not defined in the AUTOSAR 3.x standard. It is linked to the relevant AUTOSAR communication drivers for the bus system being used (CAN, FlexRay, Ethernet, etc.) (Figure 1). The XCP driver is seamlessly integrated in the existing BSW configuration chain for convenient configuration of XCP and the remaining modules.

Creating an A2L file

It is a challenge to create the A2L file using the supplemental information noted above. To accomplish this task, the test engineer needs a defined process and suitable tools. Due to the many different configuration options of the AUTOSAR BSW, it is advantageous to generate the A2L file and therefore any changes made to the BSW configuration are accounted for in the A2L file.

The A2L file needed to measure the data flows between software components (SWCs) can be generated with the RTE generator from Vector (Figure 2). This enables monitoring of Intra- and Inter-ECU communication and measurement and stimulation of the connected sender/receiver ports. The A2L file is generated based on the RTE configuration which is created with DaVinci Developer. In addition, the RTE generator inserts changes needed for the measurement in the RTE code.

To track the data flow in the modules beneath the RTE as well (Figure 2), Vector also offers an A2L generator for measuring parameters in the basic software. The A2L

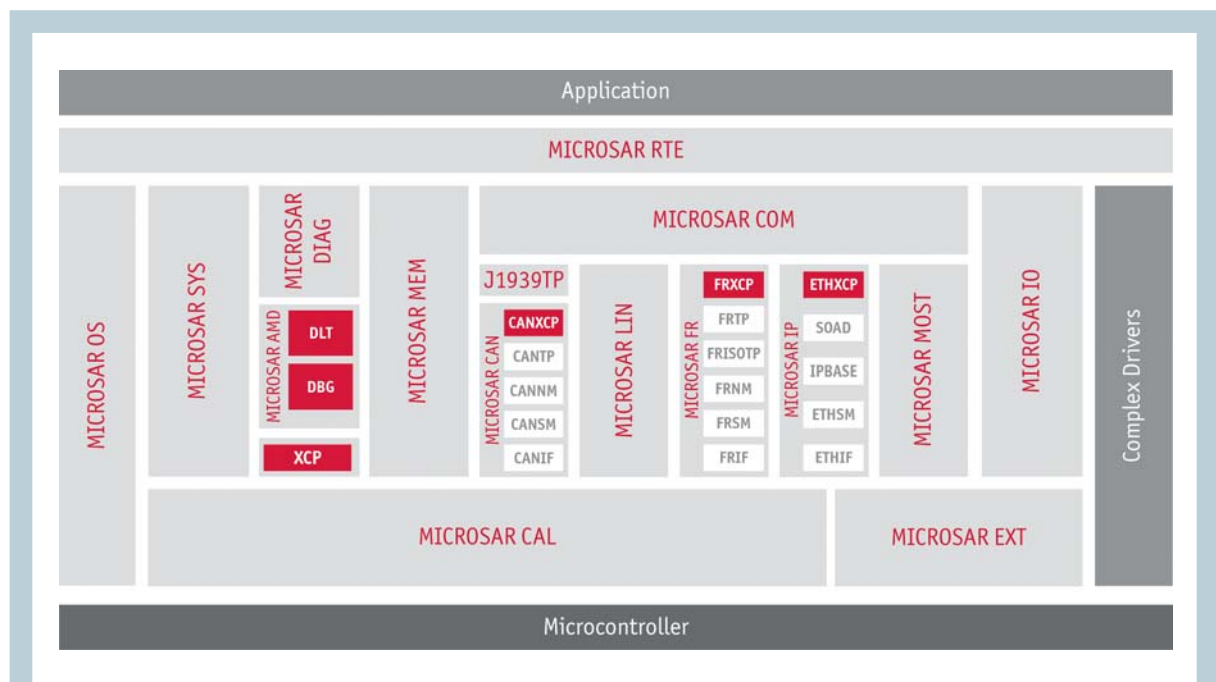


Figure 1: MICROSAR – the AUTOSAR basic software from Vector – contains modules for measuring internal ECU data via XCP.

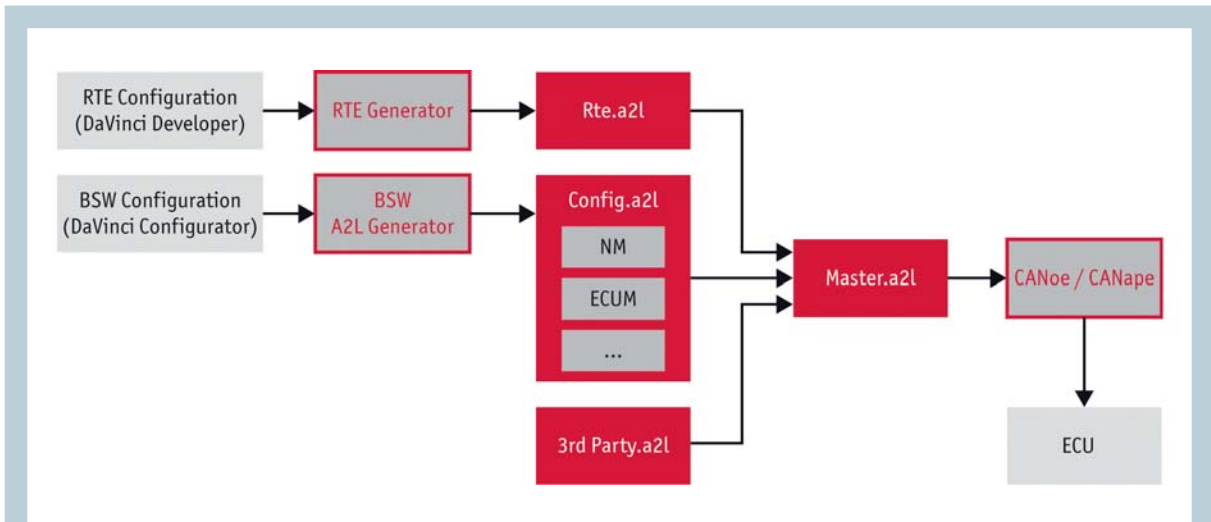


Figure 2: Generating A2L files based on the ECU configuration.

© automotive

file is also generated based on the configuration. To make it easier to find the measurement objects, they are stored hierarchically in a directory structure within the A2L file.

Acquiring measurement data

Often, the test engineer can already localize the modules responsible for the error and define the data for analysis based on the error pattern. First, the XCP Master establishes a connection to the XCP Slave. After the connection has been established, the first data can be read out from the ECU. This is generally information such as the version numbers, version identifiers, etc. Based on this information, the test engineer selects the A2L file matching the

ECU’s version level and starts the data measurement.

In case of errors that are difficult to reproduce, a quick analysis of the data might not be possible while the measurement is running. In these cases, data loggers with integrated XCP functionality or software tools with an XCP Master store measurement values in the vehicle for longer time periods. In subsequent analysis of the data, XCP Master tools are used such as CANoe – the widely used simulation, analysis and test tool from Vector.

Analyzing measurement data

To test ECUs independent of the communication bus that is used, the XCP Master must support various bus systems such as CAN, FlexRay or Ethernet. Furthermore, in debug-

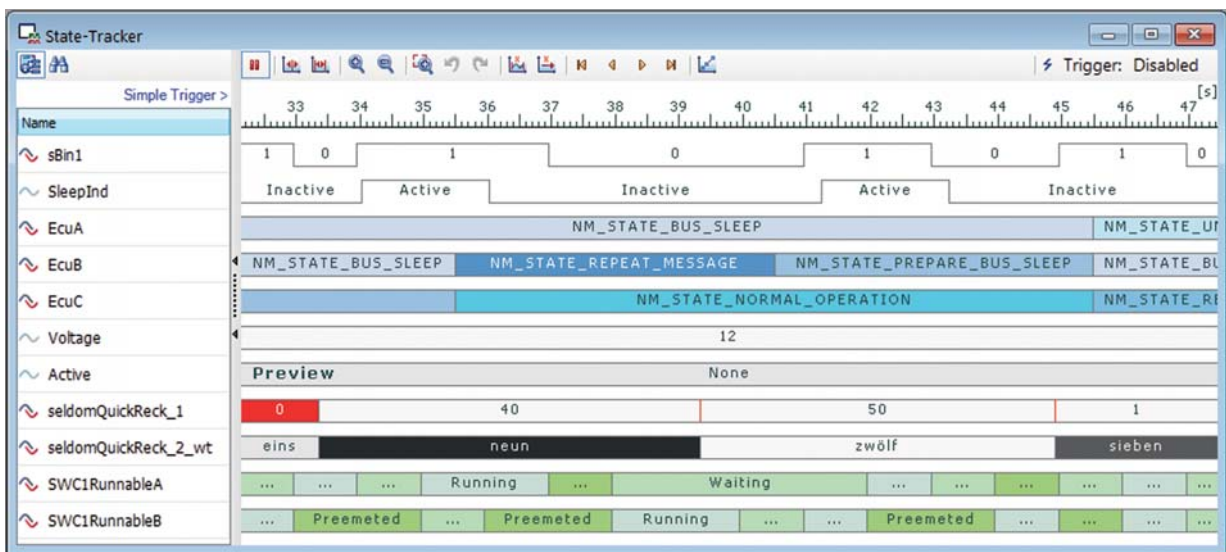


Figure 3: The CANoe State Tracker window offers users convenient visualization of internal ECU data acquired with XCP.

© automotive

ging it may be necessary to measure and evaluate the data of multiple ECUs in parallel. Therefore, it is advantageous to have an analysis tool that is also multi-master capable. The Option AMD (AUTOSAR Monitoring and Debugging) extends the functionality of CANoe with an XCP Master, in order to dynamically address multiple Slaves. Users can then change the XCP configuration during a running measurement. With Option AMD, the State Tracker (**Figure 3**), Graphic and Data analysis windows are also available for variables measured over XCP. Similarly, the XCP variables can be accessed from the integrated CAPL programming language or from test automation, enabling automated analysis.

The measurement values acquired from the ECU can be analyzed directly during the measurement. The XCP Master must offer suitable mechanisms for this purpose. Therefore CANoe includes definable trigger conditions in the State Tracker window, e.g. freezing the window contents for quick analysis when a certain measurement value is reached. In addition, CANoe enables complex evaluations of measurement values via a CAPL program.

If saved measurement values are to be evaluated in detail,

it is easier for the developer to work with symbolic representation of values rather than with raw values. To allow this, the A2L file must contain assignments of raw values to symbolic names – similar to Enums familiar from the C programming language. CANoe handles the display of symbolic values. This type of representation is excellently suited for observing state machines, whose states are often represented in code using Enums. This method can be applied to any desired variable, e.g. to symbolically indicate whether it is loaded with a substitute value or initialization value.

Debugging in AUTOSAR 4.x projects

The AUTOSAR 4.x standard satisfies the requirement for debugging options in the software by implementing the modules DBG (Debug) and DLT (Diagnostic Log and Trace). These two modules utilize mechanisms that are similar to those provided by XCP. The main task of the DBG module is to read ECU data, which is buffered inside the ECU and sent to a PC for further evaluation by the developer. Communication with the PC is done by a dedicated communication protocol. Similar to the A2L file for XCP, in the DBG

XCP

For years now, the Universal Measurement and Calibration Protocol (XCP) has been successfully used to measure and calibrate ECUs in vehicle development. XCP is a global standard that was standardized by the Association for Standardization of Automation and Measuring Systems (ASAM) in 2003. While the CAN Calibration Protocol (CCP) is limited to the CAN bus, XCP enables the exchange of the Transport Layer. It supports buses such as CAN, FlexRay, Ethernet, USB, etc. and is based on a Master/Slave principle. Using a Command Transfer Object (CTO), the XCP Master sends commands to the XCP Slave over the bus; then the XCP Slave responds with a Data Transfer Object (DTO). XCP can be used to read and write values in the ECU's memory.

The XCP Master periodically requests measurement data from the XCP Slave (polling) or has the data sent to it when specific events occur (XCP Events). Which mechanism is used depends on many factors. Measurements with XCP Events utilize bus bandwidth better than with polling. However, it is necessary to modify the ECU code to trigger the XCP Events; this is not necessary with polling. Similarly, XCP Events can already be provided at certain points in the application code during development. Aside from a slight execution time overhead, the XCP code behaves passively. When XCP Events are needed for the measurement, the XCP Master activates them during the measurement. Another advantage of XCP Events is their event-driven

output of multiple data with a common time stamp. This lets the developer precisely track certain processes in the software and check whether variables are consistent with one another.

With XCP Events, the test engineer can perform measurements independent of the state of the communication bus. In the case of polling, measurement commands must be received from the Master, and so bus communication is absolutely necessary. This requirement is omitted with an Event. However, the XCP Slave must provide a buffer that stores the measurement data until communication with the XCP Master has been restored. Due to the severely limited resources within the ECU, continuous buffer storage is understandably impossible. Two buffer strategies are conceivable here. First, there is the linear buffer, which only accepts measurement values until it is full. The other version is a ring buffer, in which the oldest entries are overwritten by new values.

Before a measurement can be executed via XCP Events, the XCP Master must configure the Slave driver suitably. This is done by a series of commands. First, the XCP Master establishes a connection with the XCP Slave, then it transfers the necessary configuration. However, if the measurement should be started immediately after ECU initialization, the measurement configuration must be stored in the ECU's non-volatile memory. This so-called Resume Mode enables measurements immediately after the ECU start.



Figure 4: The VX1000 measurement and calibration hardware from Vector enables acquisition of large quantities of data with minimal effects on ECU execution time.

© automotive

approach a description of the data to be measured is generated. This generation is based on the module descriptions of all BSW modules to be measured.

The DLT module has the task of routing the error messages and warnings generated at runtime to a PC. These messages are generated by the BSW modules DET (Development Error Trace), DEM (Development Event Manager) and by software components of the application. DLT also has its own communication protocol, which is not XCP-compatible. The functions of the two modules DBG and DLT can, however, also be implemented with XCP. Vector already offers this in its AUTOSAR 3.x solution (Figure 1).

High-performance access to measurement data

The described mechanisms provide a powerful debugging tool for any ECU – regardless of the AUTOSAR version. The developer can track and analyze processes in the ECU very conveniently using proven XCP tools such as CANoe and CANape from Vector. Use of the VX1000 measurement and calibration hardware is recommended for measuring large quantities of data at rates of up to 5 MByte/s with minimal effects on ECU execution time (Figure 4). In this case the ECU is accessed via the debug interfaces JTAG or NEXUS. (oe)



Dipl.-Ing. (FH) Patrick Markl is teamleader of the Concept Development team for the Embedded Software product line at Vector.



Dipl.-Ing. (FH) Stefan Albrecht is Senior Product Management Engineer for CANoe/CANalyzer Option AMD.