LAUTERBACH
DEVELOPMENT TOOLS

WHEN TRADITIONAL DEBUG APPROACHES ARE NOT FEASIBLE

Effective Debugging of Electronic Control Units in Vehicles via XCP

Measurement and calibration (MC) and software debugging (DBG) are essential techniques used during all stages of ECU (Electronic Control Unit) development. MC and DBG typically rely on the same target debug interface for ECU access. Switching between MC-hardware and debug probe is cumbersome, furthermore the mechanical setup might even prevent a debug probe access to permanent installed ECUs. Debugging over XCP overcomes these challenges.

Introduction in XCP

As the successor to the CAN Calibration Protocol (CCP), the Universal Measurement and Calibration Protocol (XCP) is primarily used for measurement – the acquisition of values of internal variables of an ECU – and calibration – the adjustment of internal variables. It has been standardized by an ASAM (Association for Standardization of Automation and Measuring Systems) working group. Lauterbach made a significant contribution to the debugging over XCP working group. ASAM is an organization of automotive OEMs, suppliers, and tool manufacturers. In addition to Lauterbach, key players such as dSPACE, ETAS, Robert Bosch, and Vector are represented in the responsible ASAM working group.

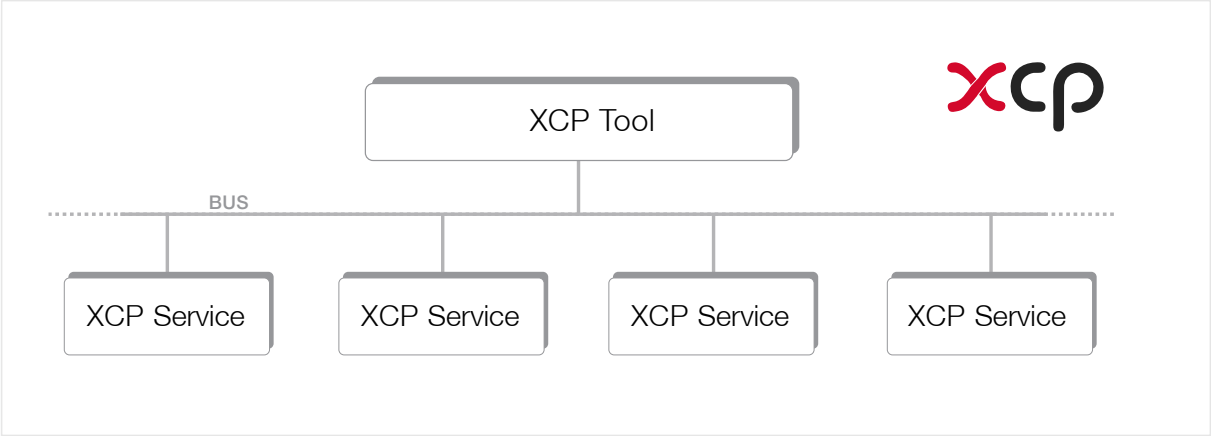

Figure 1. XCP as a Tool/Service solution.

XCP is designed as a two-layer protocol that implements a unique protocol layer and a transport layer that supports different transport media/ busses. The following transport layers are particularly relevant to the automotive industry XCP on CAN and CAN FD, XCP on SxI (SPI, SCI), XCP on Ethernet (TCP/IP and UDP/IP), and XCP on FlexRay.

An XCP solution consists of a tool (e.g. for measurement and calibration) that communicates with an XCP service. The XCP service is either implemented in the ECU or an POD connected to it. Even if the architecture allows a tool to communicate with several services (figure 1), in practice one tool often communicates with one service.

Various measurement and calibration tasks can be performed by different configurations of the XCP service without recompiling the ECU application code. The procedure works universally and is not limited to embedded ECUs.

CHALLENGE:
Limitations in Various ECU Debugging Approaches

While MC and DBG techniques have typically been used apart in the past, the demand of concurrent use is growing over time. Different approaches result in different disadvantages, including the impossibility of connecting a debugger to the ECU.

For example, switching of ECU debug signals (Figure 2) using a hardware-based arbitration mechanisms which lacks semantical information of the arbitration request. This limits interoperability, system performance and usability. In addition, switching of high-speed signals is impossible. Last but not least, if the POD is encapsulated within ECU housing, the debug probe is unable to access the POD.

A partitioned MC system (figure 3) using a proprietary protocol for communication between a base module and a plug-on device (POD) even prevents to relay debug probe signals.

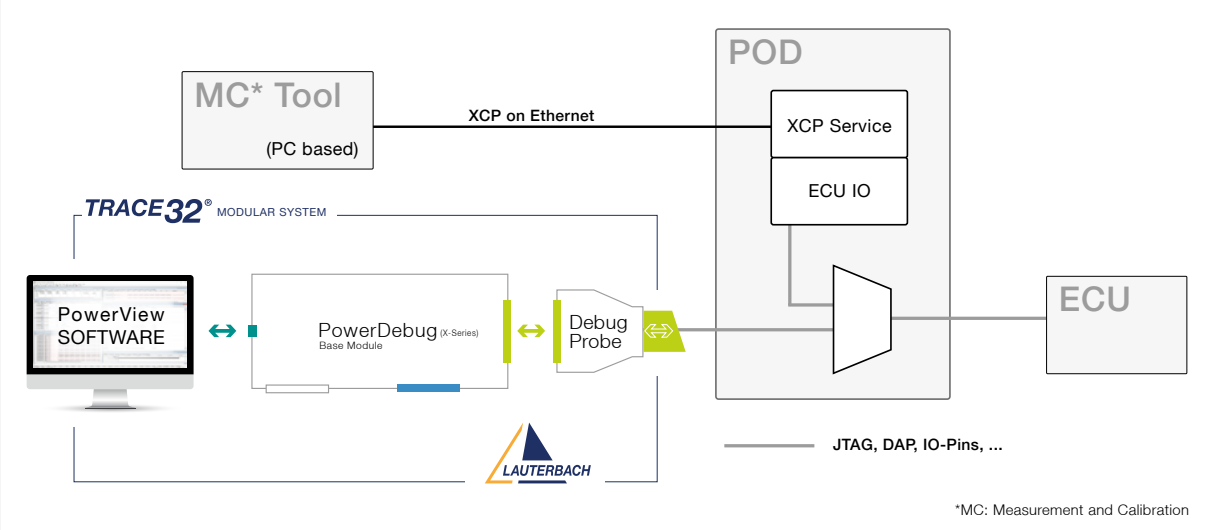

Figure 2: Switching of ECU debug signals suffers from several disadvantages.

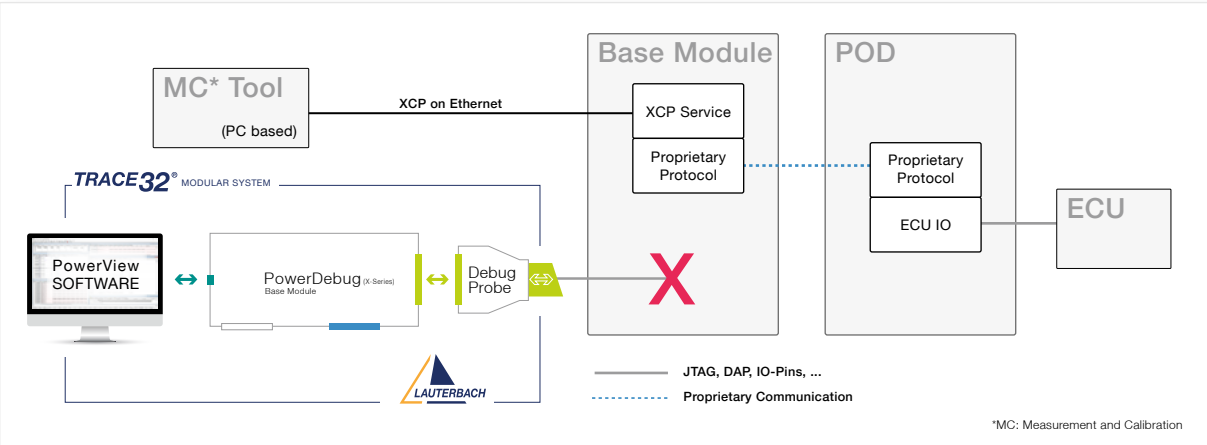

Figure 3. Partitioned MC system is unable to provide debug signals.

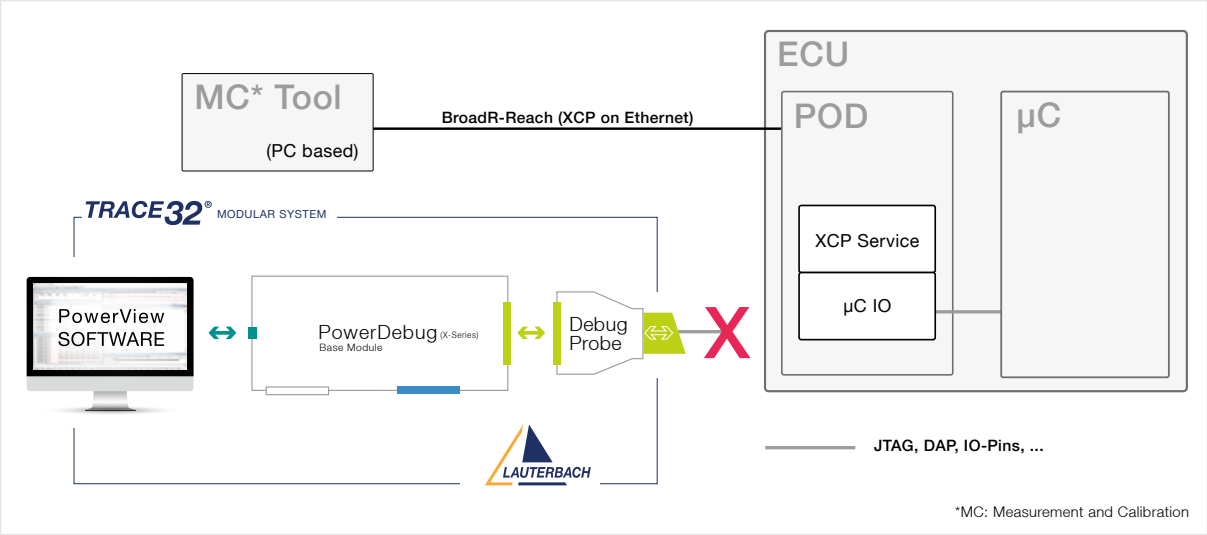

Figure 4. Tool accesses via e.g. BroadR-Reach lack an interface to connect the debug probe to a POD inside an ECU.

Last but not least, there are applications where the POD is integrated directly into the ECU, e.g. in gearboxes or in the engine compartment. Here, the MC is connected via the BroadR-Reach Ethernet physical layer standard from Broadcom [1], for example. In this case, the necessary tool access fails, too. As figure 4 shows, there is no interface option to connect the debug probe to a POD inside an ECU.

SOLUTION:
Utilization of XCP for efficient ECU debugging

To overcome the challenges described before, XCP debugging provides a manufacturer-independent mechanisms addressing todays and future needs of ECU debugging. The idea was to create a standard enabling the interoperability of a debugger like Lauterbach's TRACE32® PowerView with different PODs and different MC tools. Figure 5 shows the general setup.

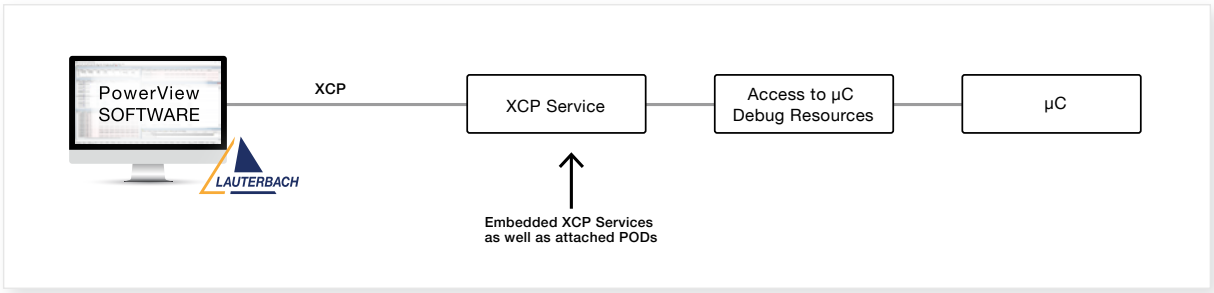

Figure 5. XCP Debugging at a glance.

The standard covers two possible scenarios: The POD-based solution (figure 6) and an embedded XCP service, where the XCP service is part of the ECU itself (figure 7).

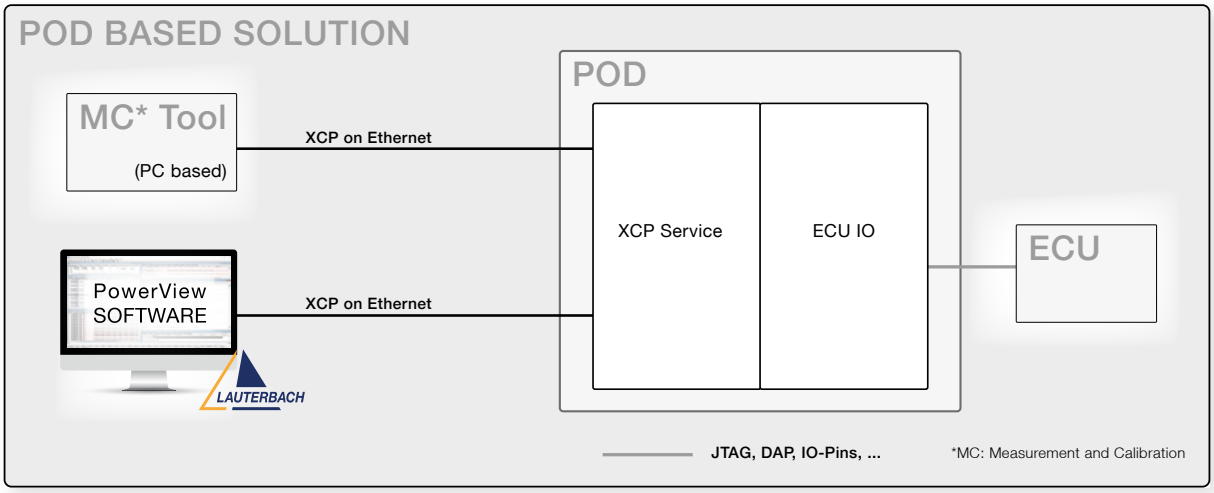

Figure 6. The POD based solution provides the largest range of debug functionality.

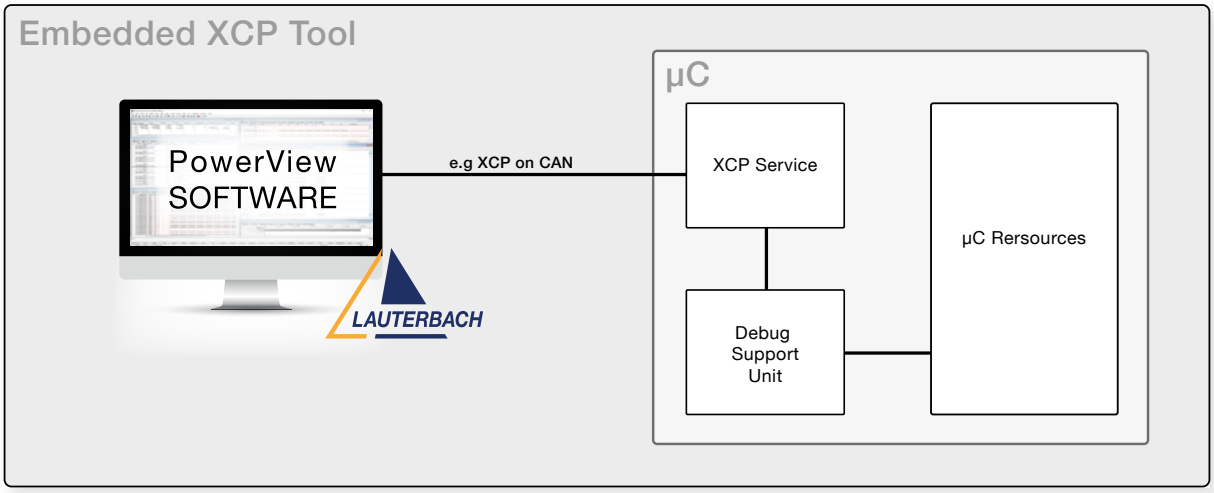

Figure 7. XCP service is embedded in ECU.

When implementing the POD-based solution, the PODs are connected to the control unit via serial interfaces such as JTAG, SWD and DAP. This interface is used both for debugging the software and for measurement and calibration via XCP. By including the debugging use case in the XCP standard, developers can use the debugger and the measurement and calibration tool over the same robust physical interface. This makes it possible to debug ECUs that are already permanently installed in the vehicle. The debugger software is connected to an XCP service via XCP on Ethernet.

Debug commands entered via the TRACE32® PowerView software are encoded into XCP commands instead of sending them directly to the target CPU via a low-level debug protocol. These XCP commands can either be read or write memory, perform low-level communication or access I/Os. This works like classical XCP memory access mechanisms but without address translation. The XCP commands are then sent over the host computer's TCP stack and a network cable to the 3rd party XCP service. The 3rd party XCP service translates the XCP commands back into low-level debug commands.

Low-level communication commands are a fallback solution if resources are not memory mapped or a POD is not aware of accessing arbitrary memory locations. For more complex, atomic accesses exclusive access can be requested. Concerning I/O control the debugger may control target reset, watchdog disable and other functions a POD might not be aware of.

The POD can optimize the scheduling of XCP commands from different XCP tools (MC, debugger) to improve the system performance.

The XCP debug standard covers nearly all every-day debug uses cases from setting breakpoints and watchpoints to single stepping and investigations of structures, objects and so on. The standard is not designed to support high end debug use cases like continuous off-chip instruction and/or data trace or use cases that require extremely low communication delay.

TRACE32® PowerView supports debugging for Infineon AURIX™, MPC5xxx/SPC5xxx PowerPC devices, Renesas RH850 devices and Arm® Neoverse as well as Cortex-A/R/M devices.

Partnering with Automotive Key Players

For XCP debugging, Lauterbach collaborates with market leaders in measurement and calibration tools in the automotive industry.

ETAS offers a broad range of solutions for measurement, calibration and vehicle testing. They consist of the software INCA, the ES8xx interface modules and FETK/XETK ECU interfaces [5]. They support also software debugging via the XCP standard extension and perfectly match TRACE32®.

Vector's VX1000 Measurement and Calibration hardware [6] provides direct access to ECU-internal resources such as RAM or Flash. Several tools can simultaneously access the ECU via XCP. An exemplary scenario is debugging ECU software with TRACE32® while simultaneously measuring ECU data or calibrating ECU parameters with CANape.

Conclusion

In today's automotive ECU development, software debugging and tracing as well as measurement and calibration are essential techniques. In this industry, concurrent operation of multiple tools has been a necessity for many years, and the demand is ever increasing.

Using Lauterbach's TRACE32® PowerView debug software with XCP enables developers to use many of TRACE32® world class debugging features such as multicore debugging, OS awareness and on-chip trace. Furthermore high-level language (HLL) debugging, flash programming, full peripheral register access using a descriptive menu tree and performance benchmark counters are supported.

As a member of the Software Debugging over XCP Working Group, Lauterbach is cooperating with major vendors of MC tools like ETAS and Vector. In close collaboration with these partners Lauterbach has merged two previously separate processes, ECU debugging and engine data management, to overcome the previous ECU debug challenges and support all major automotive CPU architectures.

REFERENCES:

[1] BroadReach Technology: https://en.wikipedia.org/wiki/BroadR-Reach

[2] Lauterbach TRACE32® XCP Debugging: https://www.lauterbach.com/products/software/debugging-via-xcp

[3] XCP debugging standard documentation: https://www.asam.net/standards/detail/mcd-1-xcp

[4] Video Tutorial explaining TRACE32®XCP debugging: https://support.lauterbach.com/kb/articles/trace32-tutorials-debugging-over-xcp

[5] ETAS tools for measurement and calibration: https://www.etas.com/de/portfolio/etk_fetk_xetk_ecu_interfaces.php

[6] Vector tools for measurement and calibration: https://www.vector.com/int/en/products/products-a-z/hardware/vx1000/