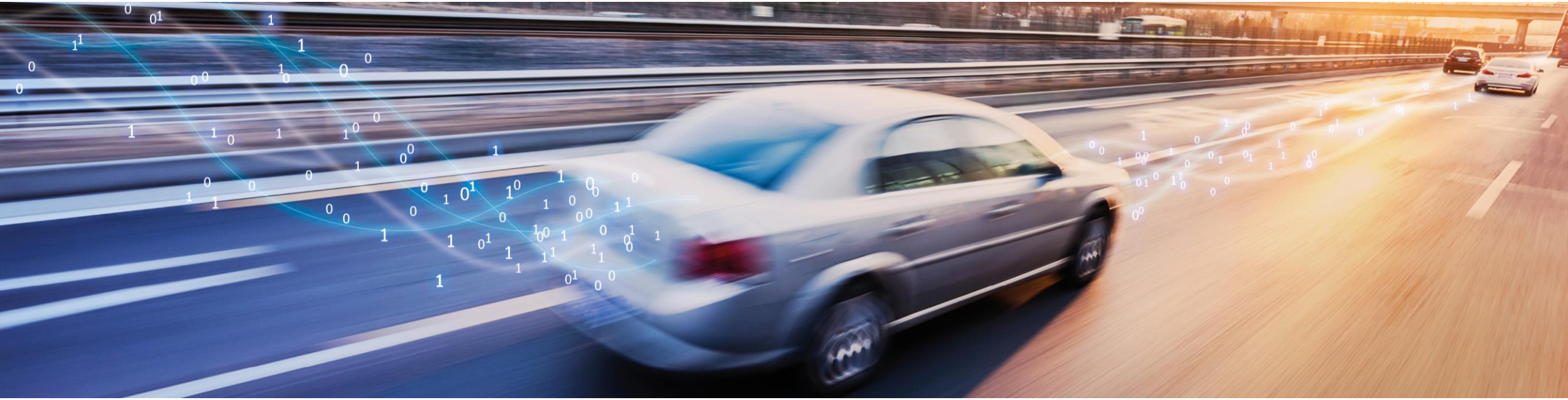
Software Debugging over XCP

Effectively Debugging ECUs in the Field

Dr.-Ing. Ralf König
Vector Informatik GmbH

October 7th, 2021



ASAM Association for Standardization of Automation and Measuring Systems

Agenda

1	Motivation
2	Key Features of the Standard
3	Use Cases
4	Supported Targets
5	ASAM Project & Workgroup

The Roots of XCP

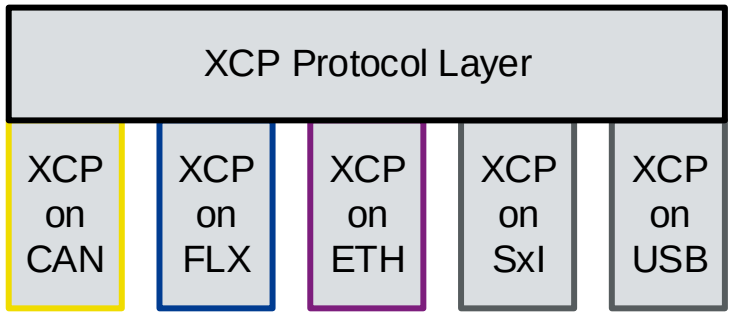
Motivation

As successor of the CAN Calibration Protocol (CCP) the Universal Measurement and Calibration Protocol (XCP) is primarily used for

- Measurement: acquisition of values of internal variables of an ECU
- Calibration: adjustment of internal variables

XCP is designed as a two-layer protocol

- Unique protocol layer
- Transport layer: support for different transport media/busses



Mechanical Challenges

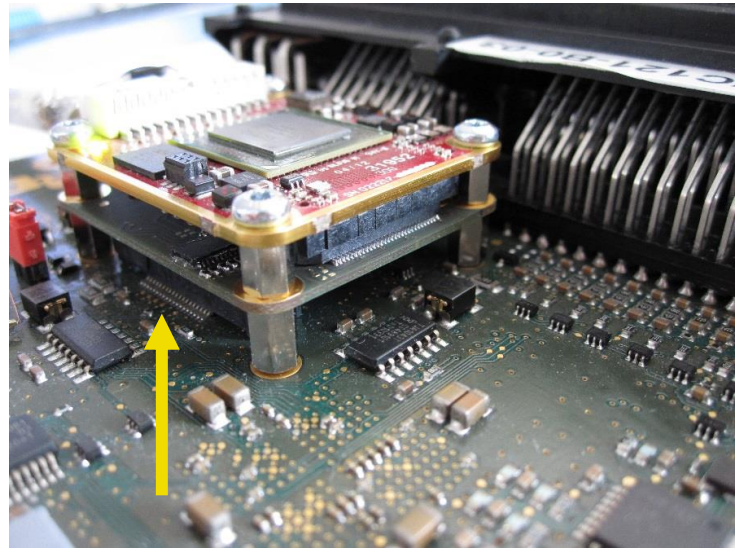
Motivation

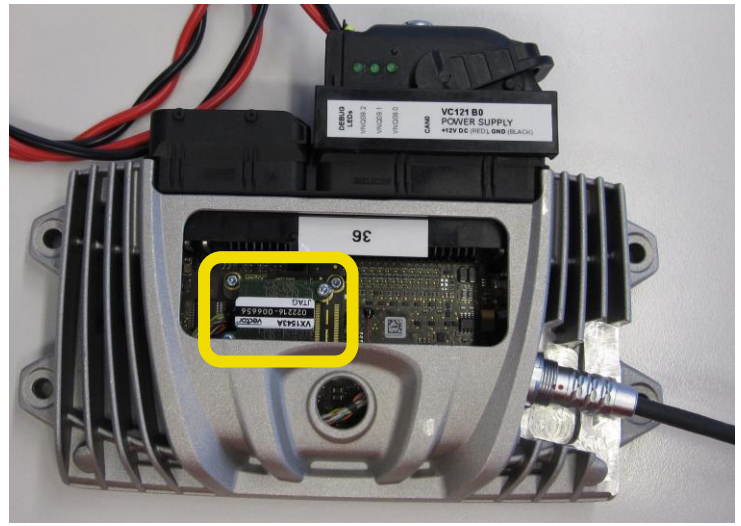
MC and DBG typically rely on the same target debug interface for ECU access

- Switching between MC-HW and Debug Probe is cumbersome
- Mechanical setup might even prevent Debug Probe access to ECU

Data acquisition and calibration (MC) and software debugging (DBG) are essential techniques used during all stages of ECU development

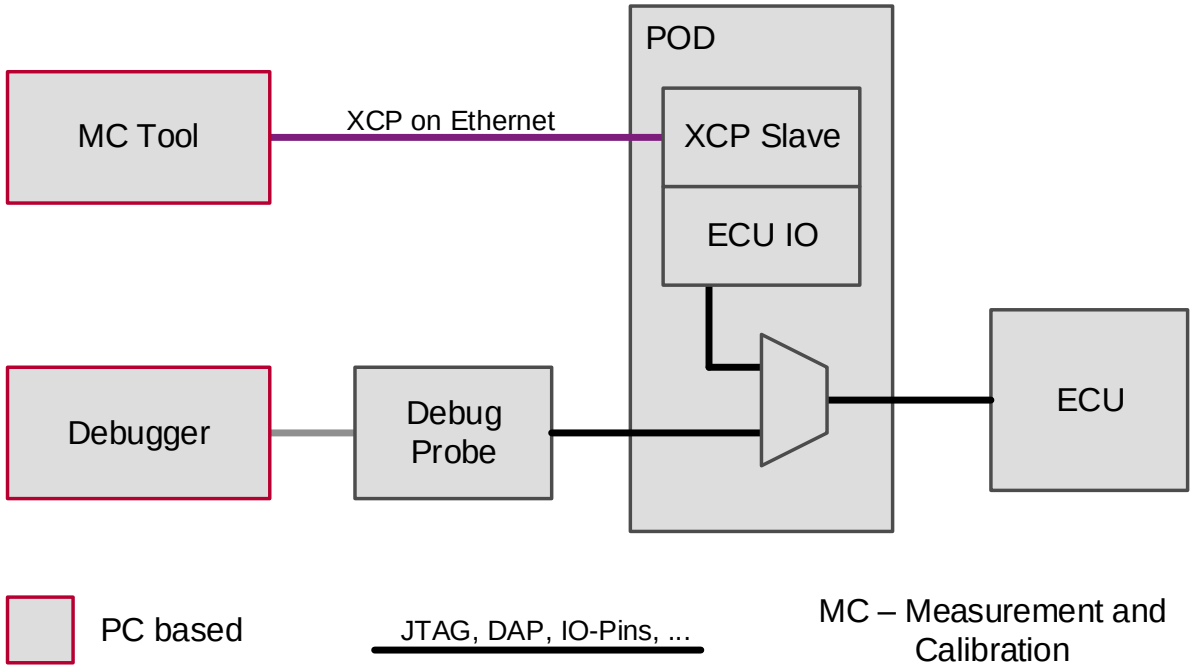
- Techniques have typically been used apart in the past
- Demand of concurrent use will even grow in future





Switching of ECU Debug Signals

Motivation

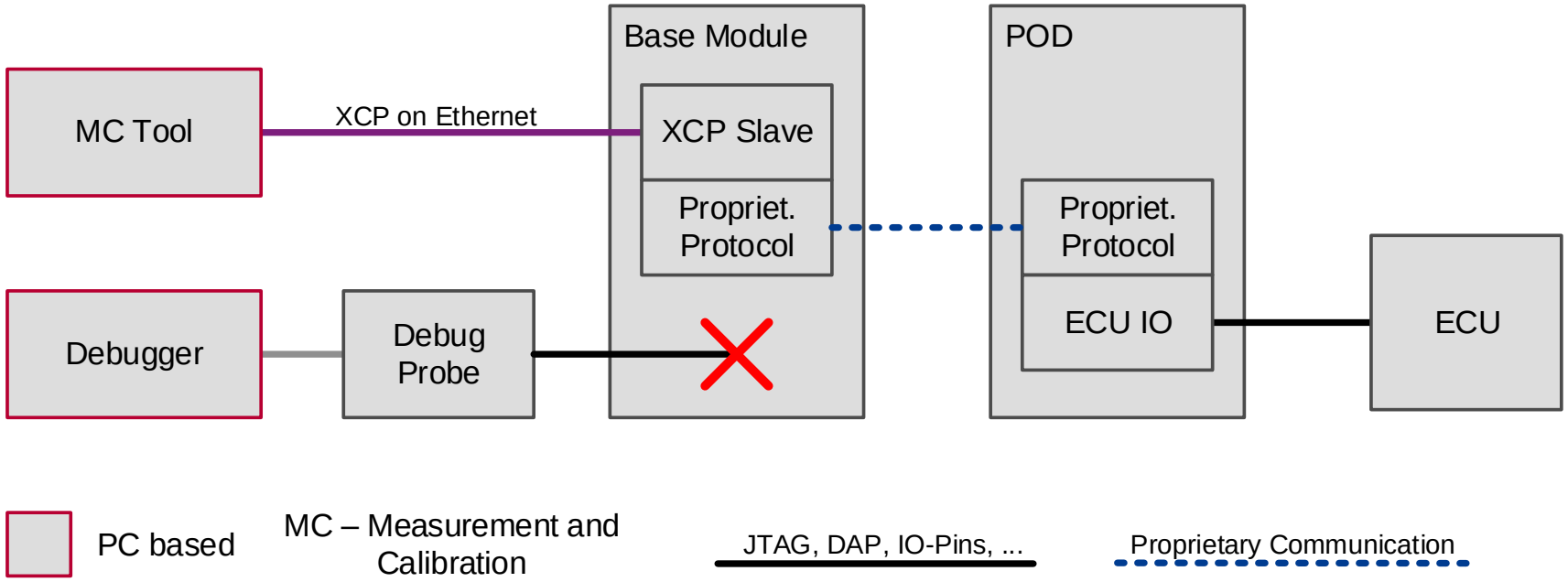


Limitations

- Hardware-based arbitration mechanisms lack semantical information of the arbitration request
  - Limits interoperability, system performance and usability
- Switching of high-speed signals impossible
- POD encapsulated within ECU housing
  - Debug Probe unable to access ECU

Partitioned MC System

Motivation

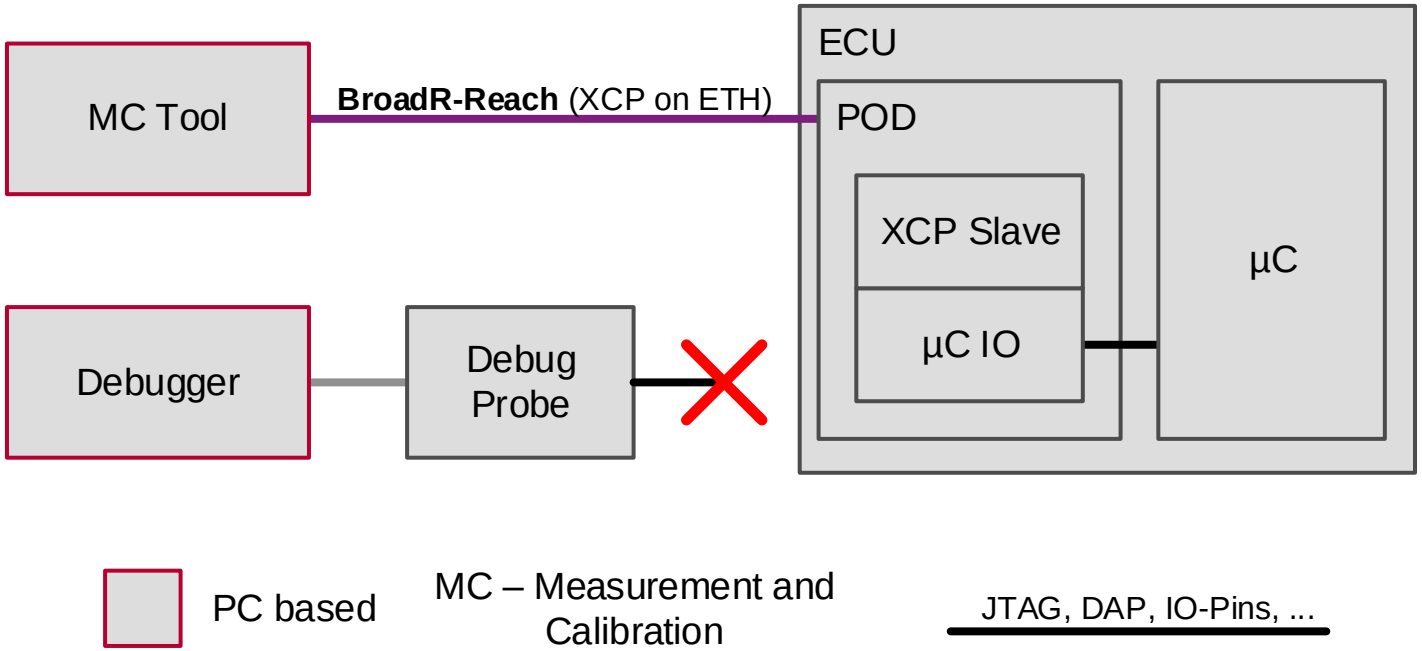


Limitations

- Proprietary protocol used for communication between Base Module and POD prevents relay of Debug Probe signals

BroadR-Reach Tool Access

Motivation



Limitations

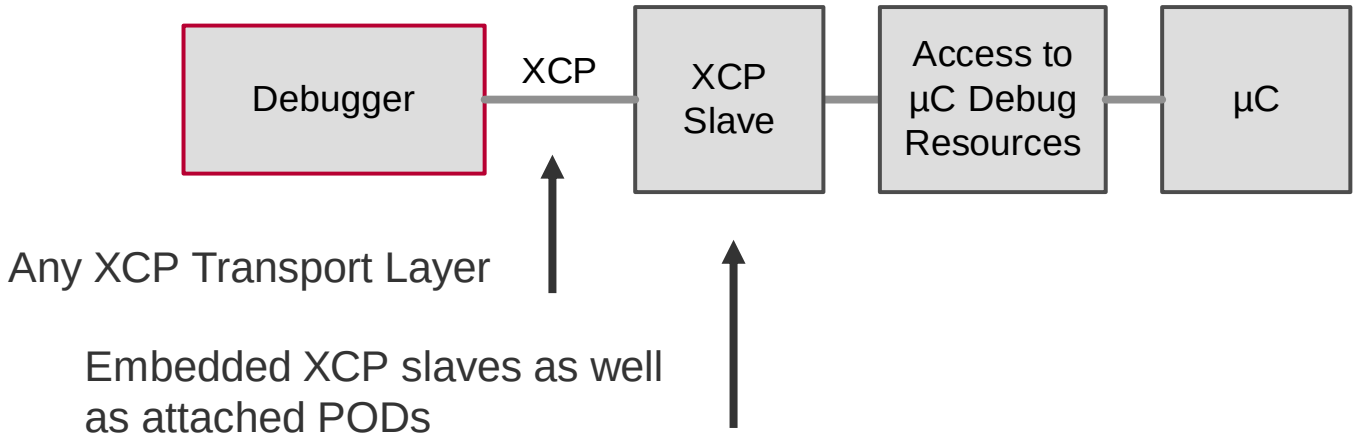
- No interface option to connect Debug Probe to POD inside ECU

Debugging over XCP Standard

Key Features of the Standard

Standardization of manufacturer-independent mechanisms addressing todays and future needs of ECU debugging

- Standard enables the interoperability of different debuggers with different PODs and different MC Tools
  - Great interest and adaptation in the market since version 1.0
- Extension of the widely used Universal Measurement and Calibration Protocol
  - By means of the ASAM Standard Debugging over XCP, associated to XCP
- Definition of generic mechanisms
  - Shall be applicable to embedded XCP Slaves and PODs

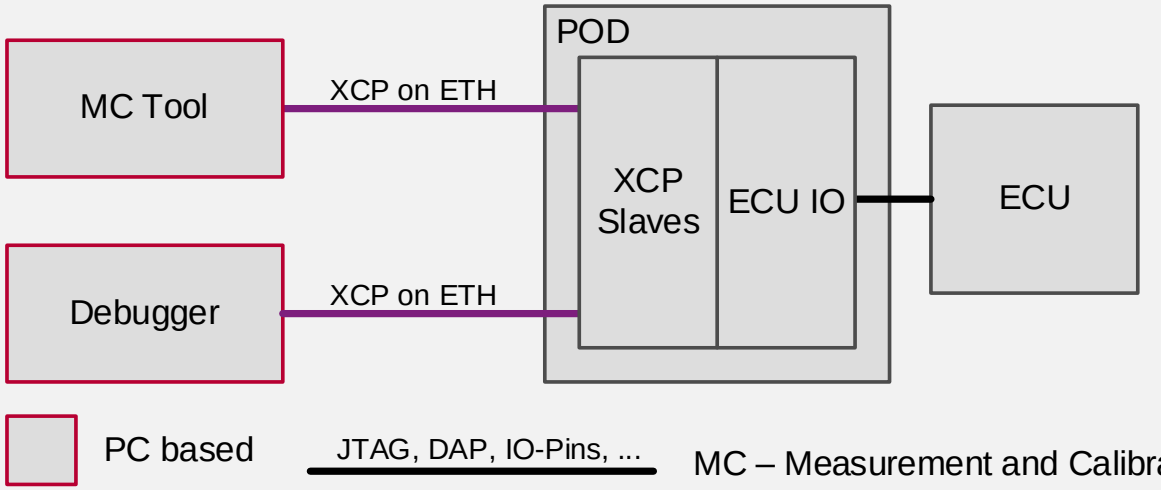


Range of Covered Technologies

Key Features of the Standard

POD based solution

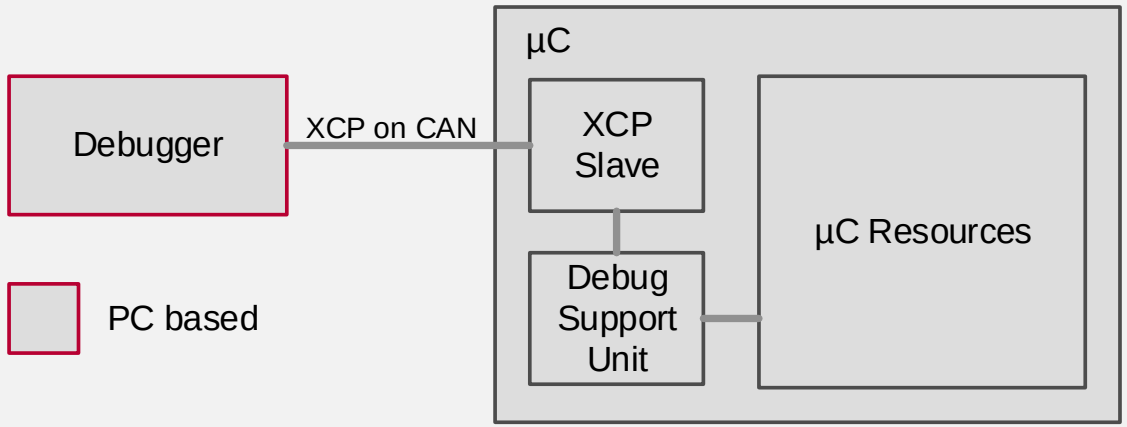
largest range of debug functionality



Embedded XCP Slave

reduced range of debug functionality

In depth comparison is given in the standard document



ASAM

Essential Features Enabling ECU Debugging

Key Features of the Standard

High level commands: reading and writing of arbitrary memory locations

- Most efficient method for interaction of debugger and target
- POD translates high level command in possibly several low-level target accesses
- Like classical XCP memory access mechanisms but without address translation

Low level target access

- Method for sending JTAG, DAP, SWD and SCI commands
- Fallback solution if
  - resources are not memory mapped
  - POD is not aware of accessing arbitrary memory locations
- For more complex, atomic accesses exclusive bus access can be requested

I/O control

- Enables debugger to control target reset, watchdog disable and other functions a POD might not be aware of

Methods Improving Parallel Use of MC and Debugging (1)

Key Features of the Standard

The XCP slave determines the service level

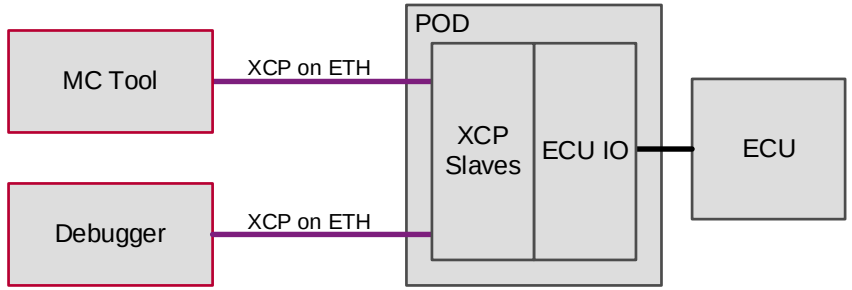
- The service level might change during run time
  - An event is sent to the debugger upon a service level change
- The debugger
  - shall adapt the feature set offered to the user according to the service level
  - shall adapt the XCP command sequence, e.g. shorten time span of exclusive target access
- 4 service level are defined
  - Service level 1 – debugging not possible
  - Service level 2 – exclusive debugger access to target
  - Service level 3 – high bandwidth assigned to debugger
  - Service level 4 – low bandwidth assigned to debugger

Methods Improving Parallel Use of MC and Debugging (2)

Key Features of the Standard

Semantical awareness of debugger activities

- Debugger uses XCP commands rather than a primitive hardware arbitration mechanism
- POD can optimize scheduling of XCP commands from different XCP masters (MC, debugger) to improve system performance
  - When needed, the debugger can request exclusive target access



Covered Use Cases

The standard covers nearly all everyday debug uses cases

- Setting breakpoints, watchpoints
- Single stepping
- Investigation of structures, objects etc.

Debugging feature set depends on

- Technology used for ECU access
- Stand alone debugging vs. parallel use of MC and debugging

The standard is not designed to support high end debug use cases

- Use cases relying on continuous instruction and/or data trace
- Use cases that require extremely low communication delay
- Debug Probes are still required for handling these use cases

Supported Targets

- Infineon DAP devices
- MPC5xxx/SPC5xxx PowerPC devices
- Renesas RH850 devices
- Arm debug interface v5 based devices
- Arm debug interface v6 based devices

ASAM MCD-1 (XCP) Software Debugging over XCP

ASAM Workgroup

- dSPACE GmbH
- ETAS GmbH
- iSYSTEM AG für Informatiksysteme
- Lauterbach GmbH
- PLS Programmierbare Logik & Systeme GmbH
- Robert Bosch GmbH
- Vector Informatik GmbH

Latest Project

- *Project:* MCD-1_XCP_SW-Debug_AS_V1-1-0_Minor
- *Duration:* November 2020 – April 2021
- *Topics*
  - Support for ARM target device
  - Support for Renesas RH850 Serial Communication Interface (SCI)
  - Improved interoperability operability between tool and XCP slave

Thank you for your attention!

Dr.-Ing. Ralf König
Vector Informatik GmbH
Holderäckerstraße 36, 70499 Stuttgart
ralf.koenig@vector.com