trace.check

Product data sheet

Version: 2025.2
as of May 2025

trace.check accelerates in-depth analyses

trace.check is a tool that automatically and thoroughly analyzes measurement data. It is specifically useful, wherever you need test results in the form of reports, which are easy to read and to interpret – no matter where the data comes from or into which tool chain you plan to integrate the tool.

Key features at a glance

- Easy analysis specification via
  - Triggered analyses
  - Timing diagrams
  - Python interface
- Support for all common recording formats
- High reusability of analyses
- Intuitive graphical user interface
- Clear presentation of results
  - Transition to the interactive trace.xplorer signal viewer
  - Plots enriched with result data

Means of specification

Various means of specification are available for the user to formalize requirements in the form of analysis components:
Any logical expressions from signal names, package variables and internal functions can be used in trigger blocks and calculation steps.

With timing diagrams both simple and complex signal relationships can be clearly and at the same time formally described.

The Python interface allows the users to implement the analysis specifications themselves. In doing so, they can focus on the actual requirements, as many tasks – from processing traces via the interpolation of different time axes to generating report entries – are carried out automatically by **trace.check**.

For the efficient processing of the signal data the user is provided with all the functionalities of the program libraries NumPy and SciPy.

In addition, **trace.check** provides various methods for synchronizing multiple recordings:

- AUTOSAR Time Synchronization/ PTP
- CrossCorrelation
- EqualnessMatching
- ExpectationMatching
- LeastSquares
- Offset
- UtcTimestamp

Interfaces

Automation interfaces (COM, REST) enable all **trace.check** work steps to be controlled, say for a seamless and fully automatic operation in an existing tool chain. On request we can also provide an adapted version of the **ecu.test** Jenkins plug-in.

System requirements

- OS: Windows 10 or 11, 64 bit
- CPU: at least 4 cores
- Free hard disk capacity: at least 8 GB
- RAM: at least 16 GB, recommended 32 GB
- Screen resolution: at least Full HD (1920 x 1080)

To use file paths longer than 256 characters on Windows, it is necessary to enable systemwide support for it (see: https://learn.microsoft.com/en-us/windows/win32/fileio/maximum-file-path-limitation)

Interfaces, formats, tools and standards

Supported formats

Bus description:
- ARXML (Classic Platform) 4.1.1 to R21-11
- ARXML (Adaptive Platform) to R20-11
- DBC
- FIBEX bis 4.1.1
- FIBEX für Ethernet 4.1.2
- FIBEX for AUTOSAR Diagnostic
- Log and Trace (DLT): Analyse non-verbose Mode
- LIN Description File (LDF)

ECU description:
- ASAP2 Database (A2L)
- Executable and Linkable Format (ELF)
- Intel HEX
- Motorola S19

Signal-based trace formats:
- ASTRACE, AS3TRACE (trace.xplorer)
- CSV
- MAT (MATLAB/Simulink, ControlDesk)
- MDF 3.0, 3.1, 3.2, 3.3, 4.0, 4.1, 4.2
- PARQUET (Apache)
- STI, STZ 2.0.1, 2.1, 2.2 ASAM XiL-API
- TDMS (National Instruments)

Bus logging:
- ASC (Vector)
- BLF (Vector)
- MDF 4.0, 4.1, 4.2
- TTL (TTTech)
- GIN (G.i.N.)

Ethernet:
- BLF (Vector)
- DLT
- PCAP, PCAPNG (tracetronic, Wireshark)
- MDF 4.0, 4.1, 4.2
- TTL (TTTech)
- GIN (G.i.N.)

Middleware/Cosimulation:
- AS3TRACE (FEP)
- eCAL 5.0, 5.1
- ROSBAG2 (ROS2)

ADAS:
- ERD (CarSim)
- ERG (CarMaker / RealTimeMaker)
- OSI/TXT (ASAM OSI) 3.5.0
- RDB (VTD)

Multimedia:
- Audio: WAV, FLAC, MP3, OGG
- Video: AVI, MP4, MKV, MTS, WMV

Other formats supported on request.