Vehicle diagnostics is a very useful yet very complex topic over a vehicle’s development and service life. Development and service department times can only be minimized if reliable, correct and precise data is available at all times. Suitable test tools are needed to reach this high level of quality while developing ECUs and their diagnostic functionality. This article describes both the test tools and the methods used to quickly and reliably avoid the causes of errors.

The importance of good diagnostic functions for successful vehicles is illustrated by countless reports of vehicle defects that required multiple repair attempts until they were finally corrected. This causes unnecessary aggravation and costs, and it is damaging to the manufacturer’s image. Therefore, the motto must be: “If there is a defect, it must be corrected quickly and accurately.”

The fault memory logs irregularities before they finally lead to component failure. With regular vehicle maintenance in a service department, defects can be detected and corrected before undesirable effects make their appearance. Diagnostic functions are also important during product development, because error states naturally occur more frequently during testing and trial phases. Relevant data from the vehicle makes it easier to perform diagnostics, possibly without even requiring any additional test instruments. Therefore, it is important to correctly log error states in fault memory.

**Functional Principle of Fault Memory**

Based on the widely used Unified Diagnostic Service (UDS) standard ISO 14229 [1], this article explains which data is provided for fault diagnostics. A central element of the UDS standard is 24-bit trouble codes. Most of the trouble codes identify OEM-specific error causes such as “Battery voltage – voltage too low.” In addition, the standard defines certain trouble code groups such as “Drivetrain.”

For each trouble code, an 8-bit status mask indicates whether the error occurred and when, such as “Test not passed since current power up.” Even more supplemental details can be stored as optional environmental data. This data is OEM-specific and it records other helpful measurement data, such as the number of error events and at which odometer reading the error was observed for the first time and most recently.

During vehicle operation, each ECU collects the error states that have occurred in its fault memory. With suitable tools, you can use diagnostic services to access this memory. Typical requests are:
for trouble code XYZ, check whether it is active and whether the voltage was between 11.5 and 12.5 Volt when the error occurred!" The test code for this function would be extremely compact and clear.

Efficient Access to Fault Memory
The described abstraction level might, for example, be provided by the CANoe test tool in combination with the Test Automation Editor test design tool. Both are software products from Vector, and they support both the UDS and KWP 2000 protocols.

A number of test functions are intended for fault memory tests, and they can:

- Read out individual trouble codes with their environmental data, check them and document them in a test report. If necessary, allowable trouble code combinations or alternatives can also be shown.
- Read out all trouble codes that have a specific status mask
- Clear the fault memory
- Read out the supported trouble codes
- Read out the status of the fault memory

Test functions in the Test Automation Editor can be used intuitively, and CANoe automatically selects the correct OEM-specific diagnostic services. Automatic selection makes the test sequences easier to reuse for other ECUs, and they are easier to create overall. This lets users work very effectively, since they can focus on the actual goals of testing.

Testing Fault Memory Functions
One strength of fault memory is that the vehicle provides the algorithms for detecting error states. However, this assumes that the vehicle’s diagnostic functionality itself is operating correctly. Consequently, this functionality must be intensively tested during the vehicle’s development. Tests of diagnostic functionality can be very complicated due to the very large number of existing trouble codes, and the extensive environmental data and error states. This is especially complex, since the parameters of the diagnostic requests must be set, and the response parameters must be checked for each test. However, this situation can be considerably simplified by having the tester use an abstract perspective. This fully hides routine activities for setting, checking and finding error entries from the user.

It would be ideal, if – after stimulation (Figure 1) of a situation logged in the fault memory – a test function were available that could hide all UDS-specific aspects and automatically check the diagnostic responses. Such a function might do the following: “Read the environmental data for trouble code XYZ, check whether it is active and whether the voltage was between 11.5 and 12.5 Volt when the error occurred!” The test code for this function would be extremely compact and clear.

Figure 2 illustrates the method for accessing the fault memory. During execution, the ECU’s fault memory is

![Figure 1: Left: Testing fault memory in the development phase – Right: Fault memory being used for vehicle diagnostics during everyday vehicle operation.](image1)

![Figure 2: Testing fault memory with the Test Automation Editor](image2)
polled by entering the desired “DTC status mask”. In the response, which might contain many different trouble codes in any sequence, a check is made of whether the trouble code P000004 was reported. If this trouble code is found, the status bit “Failed since last clear” must be set. Each trouble code has associated environmental data, as illustrated by the odometer reading in Figure 2, and this is very easy to check.

Although this test function controls a complex sequence, it is easy to set its parameters. In a conventional programming language without dedicated fault memory support, however, it would take considerable effort to realize the same functionality.

Fault Memory – Not a Riddle Wrapped in an Enigma
The primary goal has been, and continues to be, a fault-free and robust vehicle. Vehicle problems can be detected early and corrected using the fault memory. Therefore, correct self-diagnostics with fault memory should be checked with extensive tests starting early in development. This checking can be formulated in an uncomplicated and very efficient way on an intuitive abstraction level using test design and test execution tools. In this way, fault memories can contribute towards early detection of vehicle problems before they lead to vehicle failure in everyday driving.

Translation of a German publication in Elektronik automotive, issue 2/2013

Image rights:
Vector Informatik GmbH

Link:
Homepage Vector: www.vector.com

Martin Huck
studies electrical engineering at the Ruhr-University in Bochum, Germany. He was then employed as a development engineer in the communications industry for a number of years. At Vector, he worked on software development for CANoe’s Test Feature Set until 2012, and he is now a technical trainer.

Siegfried Beeh
studies electrical engineering at the University of Stuttgart, Germany. Since 2002, he has been actively working in software development at Vector, and since 2006 he has been responsible for the “Test Tools & Diagnostic Features” group. As Product Manager for Testing Tools, he is responsible for such product components as the CANoe Test Feature Set and the Test Automation Editor.