CAN FD Conformance Testing: Minimum requirement to ensure interoperability.

Patrick Isensee, C&S group GmbH

At the beginning of 2012, Bosch released the first version of the CAN FD specification to fulfill the increasing demands for bandwidth and cost efficient communication protocols. CAN FD enhances the CAN protocol to support bit rates higher than 1 Mbit/s and payloads longer than 8 bytes per frame. Many OEMs worldwide are very interested and some are heavily committed to this approach.

The use of different implementations will lead to the question how interoperable they are. Experiences in established automotive bus systems like CAN, LIN and FlexRay have shown that it is not a matter of course that devices of different manufacturers work together in diverse environmental conditions. Conformance testing is the solution for this problem to ensure a level of interoperability of devices from different suppliers in a system environment.

This presentation points out how conformance tests are drafted and specified, which techniques are used as well as how traceability, reproducibility and dependability are achieved. Last but not least, a detailed overview of the NWIP ISO16845 in conjunction with the respective test system implementation will be presented.

The common goal: Interoperability

Automotive standards are created to improve quality, facilitate innovation and increase speed-to-market, but the major motivation is the costs reduction achieved through multiple supplier solutions and cost sharing among car makers, Tier-1s and suppliers.

One of the main problems that often arises is that these standardized specifications are ambiguous or not spelled out clear enough and designers disagree on what is meant by their requirements.

The goal of all network and application designers is certainly interoperability and correct application and system behavior. To achieve this, they need to rely on some basic assumptions, e.g. that the standardized components behave "as expected" (as standardized).

If this is not assured, one can never know what the actual source of error in a complex system is.

If there is a system of more than one node, even if it is a small vehicle network, the purpose of that system is not only to exchange information but to provide certain system functionality. It is important to have reliable components that are approved, that actually do what they are supposed to do.

The conformance test (CT) can certainly not totally guarantee interoperability, but it can safeguard and drastically increase the chance of interoperability in a system with appropriate test coverage. If a specified standard defines interoperable components (i.e. does not prevent interoperability by specification bugs), the conformance test assures that all implementations to that standard passing the test are most possibly interoperable, even in corner conditions – as such situations can also be tested in the conformance test.

The basic idea of conformance tests

The basic requirement for an application of a conformance test is that a specified standard exists. This could be each kind of standard, also minor company-specific standards. Even if the conformance test can be applied as well if only a single implementation according to a standard exists, a CT is normally only considered in case that there are more than one implementation, typically also from different implementers.

Only this fact creates a situation where different implementations could be combined into a single system or network. Only that combination bears the risk of

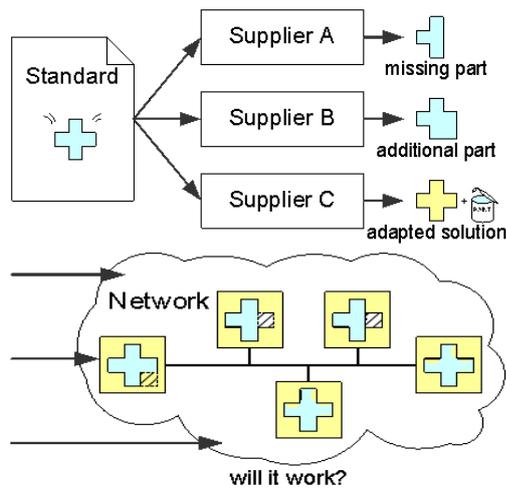troubles in case of deviations between the different implementations.

Figure 1: Multi-Supplier-Solutions

It can be assumed that a solution of a single supplier is basically interoperable with other implementations of the same kind. If all share the same non-standardized behavior, they have a good chance to "interact" apparently correct. If another implementation is introduced, a non-standardized behavior of an implementation might prevent the expected (specified) behavior in certain situations that are difficult to find in system-level tests and by try-outs. Therefore, the conformance test needs to be considered in case of systems with multi-supplier solutions.

How conformance tests are drafted and specified - Tagging and test case definition

Tagging (placing tags) is a method to make sure that no testable items are forgotten during test design. At first, all testable items, as given in the specified standard, need to be marked. Testable items are statements about the specified behavior, specified states, decisions, state information, requirements, etc.

After tagging, the test cases are designed to cover specific tags. It is important to specify tests in a way that the respective tag is visited and that the corresponding functional behavior is verified.
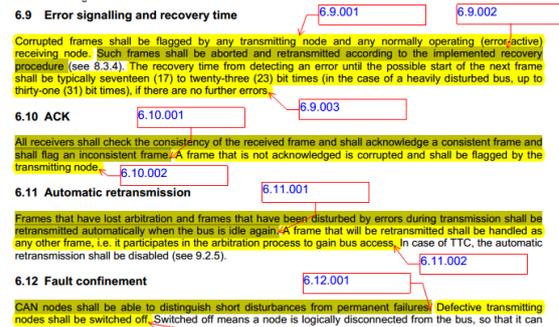

Figure 2: Tagging method applied to ISO 11898-1

The SOVS – for System Operational Vector Space - method is also used to derive appropriate test cases for a specified standard. In contrast to the tagging method, in this method all configuration parameters, operating modes, environmental conditions, outputs, failures, all external inputs and outputs are of interest.
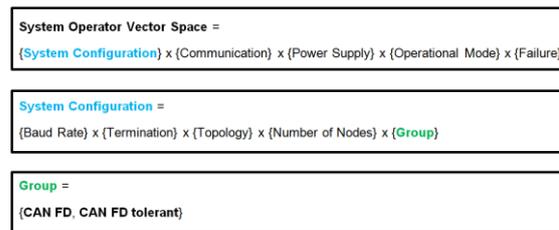

Figure 3: SOVS application example

The underlying idea is to describe all inputs that affect the operation of the implementation under test (IUT) in a virtual vector room. The whole space that is given by such vectors is the operating space in which the IUT can operate. States outside that space are illegal states that shall never happen. The IUT is then tested by applying different values for each vector, typically minimum, average, and maximum, to apply spot checks distributed over the complete operating space. If the IUT is able to work correctly in all checked spots of the SOVS, it can be assumed that the behavior "within" the limits of the operational vector space is as expected and that the IUT is implemented conform to the specified standard.

This method can be applied if a lot of configuration parameters or operating modes exist and a limited amount of test cases needs to be defined.

For the CAN FD conformance test definition, both methods of Tagging and SOVS have been combined. The full ISO 11898-1 has been tagged, taking into account every single requirement. The action domain of the test cases has been given by applying the SOVS method. By implementing the combination of this both methods, it is possible to generate a conformance test specification, covering every single requirement from the standard and ensuring that no significant test scenario is left aside.

The testing dilemma

The testing dilemma states that it is never possible to test "all" potential combinations. A reasonable big set of test cases, test configurations and test conditions need to be selected, defining some properties as constant (e.g. temperature = ambient, moisture = ambient, supply voltage = nominal), to select only some of all potential configuration sets and to limit the coverage of tests to visited states, paths, tags or another metric that can be handled in terms of complexity and efforts.

It is never useful to intend to cover all permutations that are "imaginable". It is sufficient to cover a limited amount of spot checks, touching all states or paths with minimum/average/maximum parameterization, for example, as it is very improbable that a system shows expected behavior in all "extreme" situations at its limits while it still "hides" some faults in between. If e.g. a device operates correctly at lowest clock rate and also at highest clock rate, it is probable that it also works well in between, at average clock rate, too. The same theory is used for all input parameters like strict/weak timing, minimum/maximum supply voltage, minimum/maximum payload, etc.

Test coverage metrics

To measure the current test coverage, different metrics can be applied. They differ in efforts and theoretical background, while none of them can be considered as especially "weak" or "strong" coverage.

The state coverage intends to visit all specified states at least once. If all states can be visited, it implies that all states are implemented, that certain transitions between them exist and that the transitions can be triggered by the expected means – as the stimuli of the test scenario have obviously caused the expected transitions into the expected states.

The path coverage is used to consider explicitly all paths (transitions) that can be used to get from one state to another. On such transitions, respective actions are taken like setting status flags, triggering actions, etc. Such "signatures" can be verified to make sure that the expected path has actually been taken. The path coverage is best applicable if the specified standard already provides detailed descriptions of those paths, e.g. by system description language (SDL) diagrams.

The tag coverage is based on the tagging method. During the test design it needs to be taken care that all tags are touched by at least one test case. Redundant coverage of the same tag is typically given, as each test case obviously uses many protocol mechanisms for a complete test scenario, but there should be a test case that is actually intended to cover one/some specific tag/tags.

All described metrics can be used to make test coverage "measureable", but there is no "one-and-only" way to design tests and measure their quality. The benefit of all methods is that they can be processed in a systematic way. It does not only depend on the skills of a test designer or the considered use cases but on formal, measureable facts.

General test case structure

The test purpose is essential to define a good test case. Only with a clear and unambiguous test purpose, it is even possible to judge if the following test sequence is appropriate to test the desired feature. A typical fault is that test purposes are defined so weakly that the reader cannot determine the exact scenario that shall be covered by a test case. With an unclear test purpose, the given test sequence might not cover the initially intended scenario.

Almost all specified standards define options (optional features, optional functional classes, different

implementation options, etc.). In addition, many configuration options allow adapting a communication protocol to the user's needs. Each test case shall contain an explicit statement about the required and forbidden options to apply this specific test case and also the configuration set that needs to be applied. If such information is not considered, the behavior of the IUT might not be predictable in detail. And, if a test case is intended to verify the correct operation of a certain option, the test will obviously fail if the respective option is not implemented. It is also clear that e.g. a configurable operating mode cannot be tested if it is not enabled by the test configuration. Such statements on applicability and configuration are important, as a conformance test shall only test specified behavior and only supported options according to the implementation conformance statement (ICS) of the implementer.

The setup state and the way how to get there is important, as obviously the test sequence is only applicable when starting in the expected state. It is like a route description where the starting point shall be defined. Otherwise, the expected final state will most probably not be reached by the IUT. Of course, the IUT cannot be treated as faulty if the initial conditions were incorrect. That is why the path to enter the setup state should be defined, too – as it affects the "history" how one has entered the setup state.

The test sequence describes all actions that shall be taken by the test environment (the upper tester and the lower tester) in a predefined order and timely manner. In addition, it describes what signals of the IUT need to be observed in what time periods (permanently during the test, in specific time ranges, at a certain point in time). The test sequence is the route description with instructions what to do and things to check in between. At the end, the IUT shall arrive in the predicted final state.

The criteria that are applied to judge between PASS and FAIL need to be based on information that can be acquired during the test sequence. Status data, signals or other information that is not accessible from observing the IUT interfaces shall not be considered.

Conformance tests are generally so-called "black box" tests. Such criteria can check if the acquired status data is as expected, the transmitted data was as expected, the expected operating modes have been entered, and so on. The basic idea is: If the IUT has shown the predicted behavior as specified, it operates correctly. If there would be a fault in the implementation, this would be shown in some unexpected behavior in at least some test cases. The coverage can be increased by checking more aspects of the IUT, by observing more signals that can be predicted by verifying them by more spot checks, for example.

Achieving traceability, reproducibility and dependability - Fundamental characteristics of conformance tests

As conformance tests shall be applied to different implementations of a specified standard, conformance test specifications as well as implemented conformance test cases shall be independent of any IUT or specific test environment. Tests shall be specified in a way that only requirements to a test environment are included but that no specific test environment is required. The test purposes, sequences, conditions, configurations, pass/fail criteria etc. shall be specified, but in an abstract way – so that tests can be implemented on different test environments in general. A test specification shall not contain commands or sequences that can only be applied to a specific test environment, e.g. of a pre-defined manufacturer.

Obviously, all specified test sequences shall be repeatable and shall lead to the same test result each time. It is clear that the test result shall not depend on chance, the respective test engineer, and so on. All test environments executing a specific conformance test shall be comparable and they all need to fulfill the (hopefully) specified minimum requirements as defined in the test specification. It shall be safeguarded that all test houses that perform tests according to the same test specification would have the same test results for the same IUT. That implies that all tests, the way how to test, the test environment, etc. have a comparable level. International standards and

accreditations according to them provide a certain level of dependability.

From the initial setup state on, all actions, taken by the test environment, shall be defined and logged accordingly. In addition, all test-relevant reactions of the IUT shall be acquired and logged to. Such logged data is the basis to determine the test result and to reproduce it later. The appropriate observation is given by appropriate measurement equipment for digital and analog signals, software functions that record IUT states and data, etc.

The test outcome needs to be classified in clearly separated states. A test case shall only result in PASS or FAIL, depending on the fulfillment of the expected behavior. If that setup state deviates for any reason from the expected conditions, the test result for a specific test sequence cannot be FAIL. This is because the test sequence has been applied to an IUT that was not set up accordingly. If the initial setup state was not as expected, it cannot be determined anymore, if the IUT would behave correctly out of that initial state. Therefore, the test result needs to be INCONCLUSIVE in such case – as it is not known if the IUT would have behaved correctly, if the initial state would have been as intended. If the test environment encounters an error during the execution of a test, the result can also be ERROR. Both, INCONCLUSIVE and ERROR mean that the respective test case can neither be voted as PASS or FAIL. Such tests might be repeated after finding the reason for the unexpected setup state or error during testing.

NWIP ISO16845-1 - Overview in conjunction with the respective test system implementation

All the tests defined in the test plan are grouped into categories in order to aid planning, development, understanding or execution of each Test Case. There are three levels of categories:

- The Test Types,
- The Test Classes,
- The Test Cases.

The Test Types define the direction of the frames. There are three types:

- Type 1 - Received frame type:
It includes all the tests evaluating the behavior of the IUT for Data frames and Remote frames received by the IUT.
- Type 2 - Transmitted frame type:
It includes all the tests evaluating the behavior of the IUT for Data frames and Remote frames transmitted by the IUT.
- Type 3 - Bi-directional frame type:
It includes all the tests with Data frames or Remote frames both received and transmitted by the IUT.

Each of the 3 Test Types previously defined is divided into 8 classes, grouping tests by topic regarding to the harmonized CAN specification. These 8 classes are:

- Class 1 - Valid frame format class:
This class includes the tests involving only error free data or remote frames.
- Class 2 - Error detection class:
This class includes the tests corrupting data or remote frames. These tests check the correct error detection by the IUT.
- Class 3 - Active error frame management class:
This class includes the tests verifying the IUT correct management of error-free and of corrupted Active Error Frames.
- Class 4 - Overload frame management class:
This class includes the tests verifying the IUT correct management of error free and of corrupted Overload Frames.
- Class 5 - Passive error state and bus-off class:
This class includes the tests verifying the IUT behavior during Passive Error State and bus-off state.
- Class 6 - Error counters management class:
This class includes the tests verifying the correct management of the TEC and REC by the IUT in both Active and Passive Error State.
- Class 7 - Bit timing class:
This class includes the tests verifying the correct management of bit timing by the IUT. This class of test must only be applied to components, performing only Recessive to Dominant edge

synchronization (if the Dominant to Recessive edge synchronization exists, it must be disabled). CAN FD format is not considered within this class since it has its own class assigned.

- Class 8 - Bit timing class CAN FD:

The concept behind this class is the same as the Bit timing class for Tolerant CAN FD. Due to the possibility of combining two different speeds within a CAN FD frame, there are new factors to be tested and analyzed. This class takes care of those special points and is specific for CAN FD compatible devices.

For tests belonging to Classes 1 to 6, the lower tester (LT) must be able to detect the correct value of the bit. For bit timing tests (Class 7 and 8), the LT must be able to detect a faulty synchronization of one Time Quantum.

Each test case is defined by a specific number and a particular name in order to differentiate the test cases and to easily summarize the goal of the test case. Some test cases may be subdivided into elementary tests which are repetitions of the test case for several values of the parameter to be tested.

Each test case defined in the conformance test specification clearly defines the CAN version coverage. Two version groups have been defined:

- CAN FD
- CAN FD tolerant

Depending on the case, the same test case could apply to both CAN version groups. In these cases, the test case must be carefully specified in order to provide the necessary information to carry out the test case for both CAN versions. This is accomplished by accordingly establishing the action domain for each version.

Test Automation

Testing typically implies a lot of rather simple but many times repeated tasks in different order. To form test scenarios by applying stimuli to the IUT and acquiring the IUT, the reaction is based on simple actions that can be automated. Also the sequence itself can be automated, e.g. by

scripting, by making use of pre-defined simple instructions that create a stimulus, acquire something, issue commands to the test environment, etc.

If only very few tests are to be performed, the manual test execution takes more time per test but saves the initial efforts for the automation. If more tests are performed, the faster and more reliable test execution on an automated system returns the investment soon.

The automated execution will not only minimize drastically the chance of human faults, but it will also reduce drastically the execution time of the test. With automated tests execution, the repetibility of test results is also an increased factor when compared to manual executions.

In order to carry out the conformance tests, an automated test system as shown in *Figure 4* has been implemented at C&S. The system has been developed with enough flexibility to allow new protocols or protocol extensions to be tested, such as CAN FD.
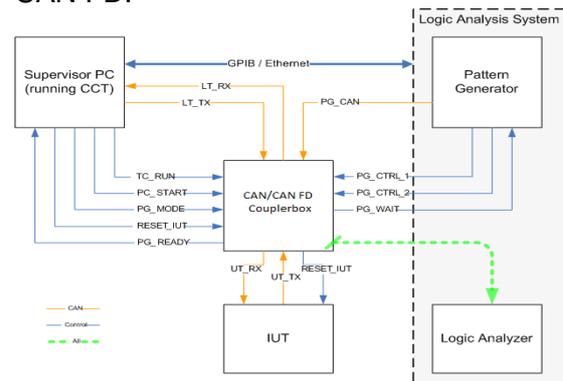

Figure 4: Concept of the Automated Test System implemented at C&S for CAN / CAN FD conformance testing.

The tests are programmed in a script language. Specific commands have been developed so that it is possible to test the unique characteristics of each protocol and their variants.

The test execution is based on phases – a preparation, real-time test execution and post processing phase. In this method, the complete test sequence shall be predefined, i.e. the IUT behavior needs to be completely deterministic – what is given in appropriate test scenarios. The interpretations of the logged data are typically done after the test runtime phase. This method is the only one that is applicable even to hard real-time systems,

as sufficiently fast test equipment can be selected, configured and prepared for the real-time test run without any time pressure, while the test execution runs autonomously on that real-time capable hardware after being triggered. The test control software coordinates the appropriate test preparation, test run and post processing steps like collecting logged data from data recorders, determination of the test result and report generation.

Figure 5: C&S CAN / CAN FD Automated Test System.

Conclusion

It is a fact that standardized specifications can be interpreted in different ways. This could lead to components with similar but not exactly equals behaviors. As these components are supposed to be capable to interact between each other, serious communication problems could arise due to a different perception of the standard specification at the moment of implementation. Based on this principle, an obvious question arises: how can it be guaranteed then that different solutions available on the market can work together? How could the designers and customers ensure an interoperable behavior between different implementation solutions? Conformance testing is the answer and the minimum requirement to guarantee interoperability. Applying the mentioned methods and concepts, C&S Group has drafted and elaborated a data link layer conformance test specification to verify the compliance to the CAN FD requirements. The first release of this specification is intended to become an international standard within ISO. The technical details are being discussed and aligned between the experts of CAN task force.

The implementation of the respective conformance test system is in the validation phase and testing services will be available from late summer of this year.

Patrick Isensee
C&S group GmbH
Am Exer 19b
D-38302 Wolfenbuettel
Tel.: +49 5331 90 555 0
info@cs-group.de
www.cs-group.de

David Bollati
C&S group GmbH
Am Exer 19b
D-38302 Wolfenbuettel
Tel.: +49 5331 90 555 0
info@cs-group.de
www.cs-group.de

References

[1] ISO/IEC 9646-1: 1994 Information technology - Open Systems Interconnection - Conformance testing methodology and framework.

[2] ISO/DIS 11898-1 Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signalling

[3] ISO/CD 11898-2 Road vehicles - Controller area network (CAN) - Part 2: High-speed medium access unit

[4] M. Scheurer: Entwicklung einer Methode zur systematischen Ableitung von Testfällen für Konformitätsprüfungen in der Informationstechnologie; C&S group, Germany, July 2000

[5] W. Lawrenz: Interoperability of Networked Components Requires Conformance Test of Communication Interfaces. SAE02 2002-01-0443.

[6] CAN: Controller Area Network: Grundlagen, Design, Anwendungen, Testtechnik. W. Lawrenz (Autor), N. Obermöller (Autor) ISBN 978-3-8007-3332-3