C&S Conformance Test
CAN – Data Link Layer
Robustness Test Specification

Author(s) and Organization(s):	Andreas Meitrodt C&S group
Reviewed by:	FF C&S group
Revision:	2.1
Date:	2020-03-26
Status:	Released
Working document:	**public**
Document Scope: (Short description only)	Description of robustness tests performed by C&S

Disclaimer

This test specification as released by C&S group GmbH is intended for the purpose of information only. The use of material contained in this test specification requires written agreement with C&S group GmbH. C&S group GmbH will not be liable for any unauthorized use of this test specification. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

Revision History

Amendment		**Description**	Editor	Date
from revision	**to revision**			
1.4	2.0	Added of FD test cases	Meitrodt	Feb 2017
2.0	2.1	Disclaimer added	Wosnitza	Mar 2020

1 The Robustness Tests – Scope

According to the ISO 9646 standard, a test suite usually consists of short deterministic Test Cases (TCs) whereas each TC focuses only on a particular function of the protocol. This approach however neglects the complex interaction between various state machines of the implementation. When checking a state machine, it is not sufficient to check each of the various paths individually only. The behaviour of a state machine is dependent on the sequential order of the paths, too. Therefore, these dependencies ought to be checked for CAN devices in order to ensure their compliance to the CAN standard.

Therefore C&S has done many efforts to develop generic robustness tests verifying the correctness of the implementation under various bus loads and during a long period of time. As the number of combinations of sequential paths in CAN is very large these test sequences are generated randomly. These robustness tests cover implementation problems like clock skew due to critical timing, sporadic errors occurring only under very particular conditions and other protocol or processor interface problems not covered by the incomplete test suite of the deterministic TCs.

The test sequences typically run for 10 000 000 Frames. The Test duration depends on the complexity of the CAN device and the number of correspondingly required tests cases. There are various parameters modified while producing the sequential order of “standard”, “extended”, “classic frame” and “fd frame” test cases such as random generation of identifiers with standard or extended length and coding, random length and random content of data, various bit rates, very high and low bus load such as >= 50% <=100%, with/without errors, etc. All the tests are executed in real time.

The verification of the tests is done in real time at message level at lower tester (LT) side and upper tester (UT) side.

2 References

[1] ISO 11898-1:2015 Road vehicles - Interchange of digital information - Controller area network (CAN) for high-speed communication

[2] ISO 16845-1:2016 Road vehicles -- Controller area network (CAN) conformance test plan -- Part 1: Data link layer and physical signalling

[3] ISO 16845-2:2014 Road vehicles -- Controller area network (CAN) conformance test plan -- Part 2: High-speed medium access unit with selective wake-up functionality

3 Abbreviated Terms

For the purposes of this document the following abbreviations apply:

All abbreviations in this document are written in upper case letters.

ACK	Acknowledgement
BOR	Break of ROB Test
CAN	Controller Area Network
CRC	Cyclic Redundancy Check
DLC	Data Length Code
EOF	End Of Frame
EOR	End of ROB Test
IDEN	Identifier
ISO	International Standardisation Organisation
IUT	Implementation Under Test
LT	Lower Tester
OSI	Open System Interface
REC	Receive Error Counter
ROB	Robustness (Test)
RTR	Remote Transmission Request
SOF	Start Of Frame
SOR	Start of ROB Test
TEC	Transmit Error Counter
UT	Upper Tester

3.1 Glossary

All expression starting with capital letters are listed in the glossary.

Active Error Flag :
First field of an Active Error Frame.

Active Error Frame:
Error frame that starts with an Active (Dominant) Error Flag.

Active State:
: A node is in the Active State when it can transmit an Active Error Frame.

Arbitration Field:
: The field starting after the SOF bit and finished with the RTR bit.

Bit Error:
: Error condition encountered when the received bit does not correspond to the transmitted or to the expected bit.

Dominant:
: see Dominant State.

Dominant State:
: The CAN bus is in Dominant State when at least one CAN node drives a Dominant value on the line.

Error Active:
: see Active State.

Error Delimiter:
: Second field of an Error Frame.

Error Flag:
: First field of an Error Frame.

Error Frame:
: Formatted sequence of bits indicating an error condition.

Error Passive:
: see Passive State.

Idle State:
: The CAN bus is in Idle State when no frame is started after Intermission Field.

Intermission Field:
: The field after EOF, Error Delimiter or Overload Delimiter.

Lower Tester:
: The Lower Tester supervises the Test Suite.

Passive Error Flag:
: First part of a Passive Error Frame.

Passive State:
: The device is in the Passive state because the value of the REC or the TEC has reached the Error Passive limit.

Recessive:
: see Recessive State.

Recessive State:
: The CAN bus is in the Recessive State when no CAN node drives a Dominant value on the line.

Suspend Transmission Field:
: Waiting time added after Intermission Field for Error Passive transmitters before it can start another transmission.

Test Case:
: Each Test Case is defined by a specific number and a name in the Test Suite.

Test Frame:

Test Frames are CAN frames containing the test pattern specified in this document.

Test Plan:

Test Plan is a specific application of the « OSI Conformance Testing General Concepts » standard.

Test Suite:

Test Suite check the behaviour of the IUT for particular parameters of the Harmonised CAN Specification.

Test Type:

Test Types define the direction of the test frames (e.g. behaviour of the IUT if receiving and/or transmitting messages).

Time Quantum:

Elementary time unit of the CAN bit time derived from the oscillator clock and the prescaler.

Upper Tester:

The Upper Tester acts as an user of the IUT. User code on host CPU of IUT

4 Architecture and Implementation of the Test Environment

The hardware environment - see fig.1 - for the “System Under Test“ consists usually of a micro-controller evaluation system = **Upper Tester (UT)** which is connected to the standalone CAN implementation = **Implementation Under Test (IUT)**. In case the CAN implementation is integrated on a micro-controller, the micro-controller will be used as UT. The **Underlying Service Provider** consists of the RX and TX CAN signals.

The UT is remote controlled - via CAN bus - from the **Lower Tester LT**, a specially equipped PC. The test coordination is performed by a program executed on this PC (LT).

The Test Cases (TC) are defined as ASCII script files. The LT also provides a test protocol for each TC.

The test system is equipped with a CAN interface and additionally with a special coupler box, allowing the programmable configuration of the CAN Bus emulation.

- IUT RX & TX connected with CCT CAN interface

- IUT RX connected with Pattern generator

The coupler box is controlled via an interface to the supervisor. In addition logic analyzer and bit pattern generators are used and remote controlled for exact analysis. The implementation of the tester architecture is shown in fig.2:

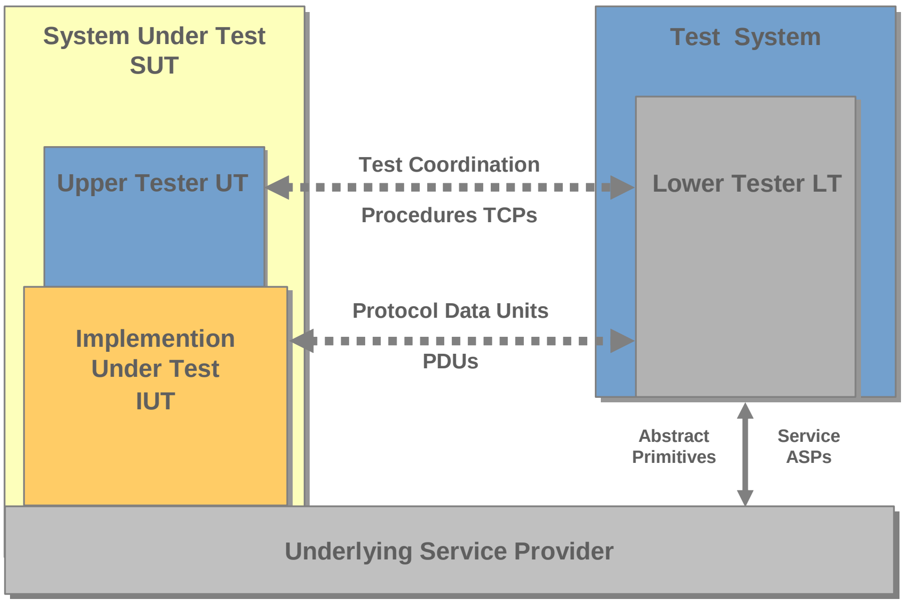

Fig 1: General Test Method

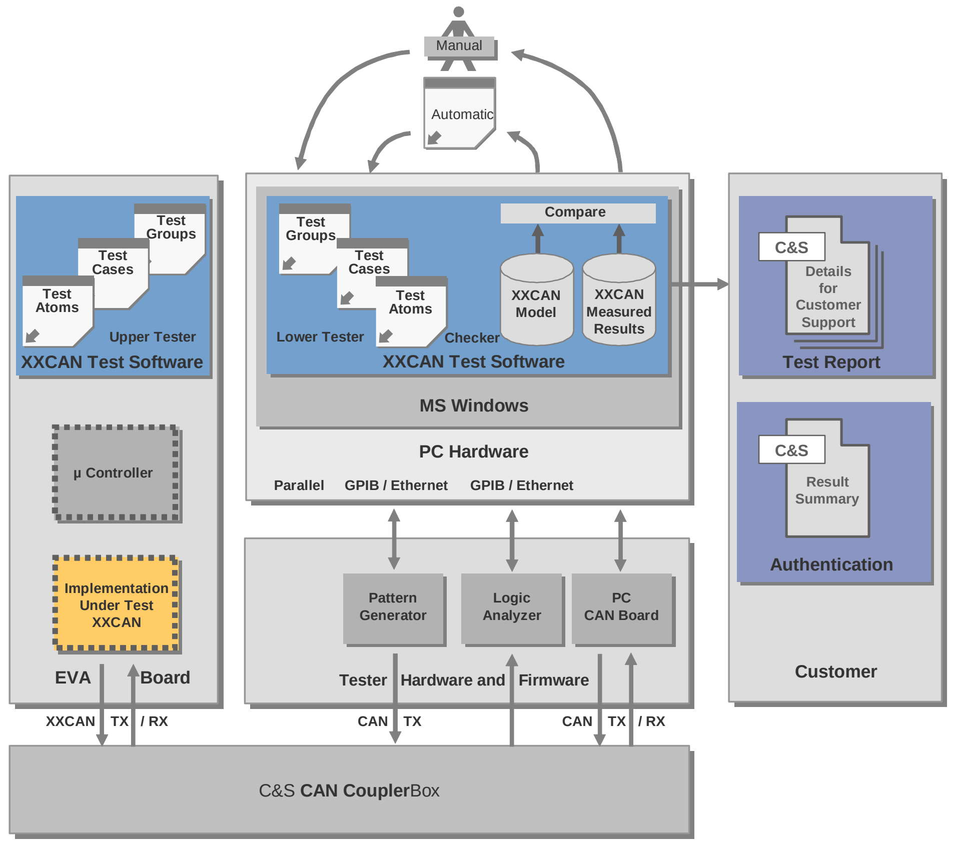

Fig. 2: C&S CAN Conformance Tester Architecture

5 Test Execution

5.1 Concept of the Robustness Test

The robustness tests are based on the exchange of CAN messages between the Lower Tester and the Upper Tester. The tests are running for several hours under different conditions of:

- CAN bus load
- error condition
- message (identifier and data) configuration
- bit timing configuration (Bit rate)

The test purpose is to check the behaviour of the IUT during asynchronous action on the physical CAN interface and on the register interface to the host controller.

To cover a maximum amount of combinations/accesses between the various state machines of the IUT (CAN core and host controller,), CAN bus events and CPU events must be occurred asynchronous. To generate asynchrony events the following conditions are used:

- random CAN messages (for transmission and reception)
- interrupt driven transmission and reception and sequential message generation
- Bus independent error bit generation

Bus load

Each robustness test is executed at <= 100% bus load. The maximal bit rate used to run the tests at 100% bus load depends on the computing power of the UT. The UT must be able to calculate the random values and generate the random messages to be transmitted guaranteeing a bus load of min 50%. In parallel the UT must generate the random messages expected to be received from the LT and to check them before the next reception.

As both, LT and UT try to send at 100% bus load, an arbitration occurs at each transmission. At lower bus load the transmitter part of the UT are delayed by hardware limitations and LT are delayed by test configuration.

Error generation

The robustness tests can be classified in two main groups:

- robustness tests without error
- robustness tests with errors

The errors are generated by the test environment (Pattern Generator) as follows. The test environment forces a bit sequence of the frame transmitted by the IUT, to dominant. Depending on the error sequence, the IUT will stay in Active Error State, reach the Passive Error State or will even go BUS OFF. Whether Active, Passive or even BUS OFF will be reached, depends on the Error sequence frequently generated, possibly disabling the IUT to recover its error counters between subsequent error bursts.

CAN frame generation

The messages transmitted by the Lower Tester (LT) and the Upper Tester (UT) during the robustness tests are randomly generated. For each message, identifier, data length code and data are generated with a 16 bit pseudo random generator.

The same random generator is implemented on the LT and on the UT, but different start values are used for the random generator leading to various random values hence various CAN messages.

To avoid errors caused by the simultaneous transmission of the same identifier but different data by the LT and the UT, during one test odd identifiers will be used by one component only (e.g. LT) and even identifiers by the other component. Both configurations will be used to ensure the IUT is able to handle the whole range of IDs the random generator could provide.

5.2 Verification of the Robustness Test

To verify the correct behaviour of the IUT during the robustness test, checks are performed at Processor Interface level.

At the processor interface level, the verification is performed by the LT and the UT. The LT checks at run time that every message sent by the IUT matches the expected random message. This verifies that the IUT sends all messages to the LT correctly and in the right order.

The UT checks at run time whether every message received by the IUT matches the expected random message. This verifies that the IUT receives all messages from the LT correctly. To ensure this verification process each component must calculate the random messages it has to transmit, and in addition it expects to receive the random messages in the right sequence.

The logic analyser stores the last messages (about 100 messages at 100% bus load) transmitted before an error has been detected by the LT or the UT. This high memory depth on the logic analyser is necessary to reconstitute and investigate the sequence which has led to the faulty behaviour.

6 Classical CAN Test Cases

Robustness Tests are executed applying test cases with the following characteristics:

- **identifier, randomly generated**
- **Control flag IDE randomly generated**
- **DLC, data, randomly generated**
- **bit rate**
  - customer defined bit rates
  - Typical bit rates (100K; 250K; 500K)
- **bus load**
  - >= 50% up to <= 100% (no separate varied, only to define a usable range)
- **error**
  - no
  - yes, asynchronous with configurable rate and width
- **run time**
  - >=10000000 Frames each Test Case

6.1 Classical CAN – ID, DLC and Data Random Test - LT: Odd Identifiers

6.1.1 Purpose and limits of this Test Case

CAN_VERSION $\in$ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

6.2 Classical CAN – ID, DLC and Data Random Test - LT: Even Identifiers

6.2.1 Purpose and limits of this Test Case

CAN_VERSION ∈ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

6.3 Classical CAN – ID, DLC and Data Random Test - LT: Odd Identifiers with Errors

6.3.1 Purpose and limits of this Test Case

CAN_VERSION ∈ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

6.4 Classical CAN – ID, DLC and Data Random Test - LT: Even Identifiers with Errors

6.4.1 Purpose and limits of this Test Case

CAN_VERSION ∈ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

7 CAN FD Test Cases

Robustness Tests are executed applying test cases with the following characteristics:

- **identifier, randomly generated**
- **Control flags IDE, FD, BRS randomly generated**
- **DLC, data, randomly generated**
- **bit rate:** - customer defined bit rates
  - Typical nominal bit rates (100K; 250K; 500K)
  - Typical data bit rates (500K; 1000K; 2000K)
- **bus load:** - >= 50% up to <= 100% (no separate varied, only to define a usable range)
- **error:** - no
  - yes, asynchronous with configurable rate and width
- **run time:** - >=10000000 Frames each Test Case

7.1 CAN FD – ID, FDF, BRS, DLC and Data Random Test - LT: Odd Identifiers

7.1.1 Purpose and limits of this Test Case

CAN_VERSION ∈ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

7.2 CAN FD – ID, FDF, BRS, DLC and Data Random Test - LT: Even Identifiers

7.2.1 Purpose and limits of this Test Case

CAN_VERSION ∈ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

7.3 CAN FD – ID, FDF, BRS, DLC and Data Random Test - LT: Odd Identifiers with Errors

7.3.1 Purpose and limits of this Test Case

CAN_VERSION $\in$ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K

7.4 CAN FD – ID, FDF, BRS, DLC and Data Random Test - LT: Even Identifiers with Errors

7.4.1 Purpose and limits of this Test Case

CAN_VERSION $\in$ {B}

The purpose of this test is to verify that an IUT is able to handle >=10 000 000 Frames in random order with random arbitration process and high busload.

There are up to three Elementary Tests to perform for the typical Bit rates 100; 250; 500K