INTERNATIONAL STANDARD

ISO 16845-1

First edition
2016-11-01

Road vehicles — Controller area network (CAN) conformance test plan —

Part 1: Data link layer and physical signalling

Véhicules routiers — Plan d'essai de conformité du gestionnaire de réseau de communication (CAN) —

Partie 1: Couche liaison de données et signalisation physique

Reference number
ISO 16845-1:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

The committee responsible for this document is ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

This first edition of ISO 16845-1 cancels and replaces ISO 16845:2004, which has been technically revised.

A list of all parts in the ISO 16845 series can be found on the ISO website.

Introduction

ISO 16845 was first published in 2004 to provide the methodology and abstract test suite necessary for checking the conformance of any CAN implementation of the CAN specified in ISO 11898-1.

Road vehicles — Controller area network (CAN) conformance test plan —

Part 1: Data link layer and physical signalling

1 Scope

This document specifies the conformance test plan for the CAN data link layer and the physical signalling as standardized in ISO 11898-1. This includes the Classical CAN protocols as well as the CAN FD protocols.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 11898-1:2015, *Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*

ISO/IEC 9646-1, *Information technology — Open Systems Interconnection — Conformance testing methodology and framework — Part 1: General concepts*

ISO/IEC 9646-2, *Information technology — Open Systems Interconnection — Conformance testing methodology and framework — Part 2: Abstract Test Suite specification*

ISO/IEC 9646-4, *Information technology — Open Systems Interconnection — Conformance testing methodology and framework — Part 4: Test realization*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 11898-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp/

3.1
bit rate prescaler
BRP
minimum time quantum used for a TQ in CAN Bit time configuration

3.2
conformance testing
applying the *test plan* (3.17) to an IUT

3.3
default state
state of the IUT

Note 1 to entry: The default state is characterized by the default value presented in 5.3.2.5.

3.4
dominant
represents the logical 0

3.5
dominant state
CAN bus is in dominant state when at least one CAN node drives a dominant value on the line

3.6
elementary test
repetitions of the test case for several values of the parameter to test

3.7
end of frame
last field of a data or remote frame before the intermission field

3.8
idle state
CAN bus is in idle state when no frame is started after intermission field

3.9
lower tester
supervises the *test suite* (3.18)

3.10
REC passive state
device is in the passive state because the value of the REC has reached the error passive limit

3.11
recessive
represents the logical 1

3.12
recessive state
CAN bus is in the recessive state when no CAN node drives a dominant value on the line

3.13
TEC passive state
device is in the passive state because the value of the TEC has reached the error passive limit

3.14
test case
each test case is defined by a specific number and a particular name in the *test suite* (3.18)

3.15
test class
each *test type* (3.19) is divided in eight test classes

3.16
test frame
CAN frames containing the test pattern specified in this document

3.17
test plan
specific application of the «OSI conformance testing general concepts» standard

3.18
test suite
checks the behaviour of the IUT for particular parameters of ISO 11898-1

3.19
test type
defines the direction of the *test frames* (3.16)

EXAMPLE Behaviour of the IUT if receiving and/or transmitting messages.

3.20
upper tester
acts as a user of the IUT

4 Abbreviated terms

All abbreviated terms in this document are written in upper case letters.

CTRL	Control field of CAN frame + SRR/RTR + IDE bit
CBFF: CTRL	= RTR, IDE, FDF, DLC merged together as 7 bit hexadecimal value
FBFF: CTRL	= RRS, IDE, FDF, res, BRS, ESI, DLC merged together as 10 bit hexadecimal value
CEFF: CTRL	= SRR, IDE, RTR, FDF, r0, DLC merged together as 9 bit hexadecimal value
FEFF: CTRL	= SRR, IDE, RRS, FDF, res, BRS, ESI, DLC merged together as 11 bit hexadecimal value
IPT	information processing time
LT	lower tester
NTQ(D)	number of time quantum in data bit rate
NTQ(N)	number of time quantum in nominal bit rate
Phase_Seg1(D)	Phase Segment 1 (Phase_Seg1) for data phase bit rate
Phase_Seg1(N)	Phase Segment 1 (Phase_Seg1) for nominal bit rate
Phase_Seg2(D)	Phase Segment 2 (Phase_Seg2) for data phase bit rate
Phase_Seg2(N)	Phase Segment 2 (Phase_Seg2) for nominal bit rate
Prop_Seg(D)	propagation segment (Prop_Seg) for data phase bit rate
Prop_Seg(N)	propagation segment (Prop_Seg) for nominal bit rate
Sampling_Point(D)	Sync_Seg(D) + Prop_Seg(D) + Phase_Seg1(D)
Sampling_Point(N)	Sync_Seg(N) + Prop_Seg(N) + Phase_Seg1(N)
SJW(D)	synchronization jump width (SJW) for data phase bit rate
SJW(N)	synchronization jump width (SJW) for nominal bit rate
Sync_Seg(D)	synchronization segment (Sync_Seg) for data phase bit rate
Sync_Seg(N)	synchronization segment (Sync_Seg) for nominal bit rate
TP	test plan

TQ(D)	time quantum in data bit rate
TQ(N)	time quantum in nominal bit rate
UT	upper tester

5 Global overview

5.1 Scope of test plan

ISO 9646-1, ISO 9646-2 and ISO 9646-4 define the methodology and the abstract test suite necessary to check the conformance of any CAN implementation to ISO 11898-1. The architecture of the TP is as shown in Figure 1.

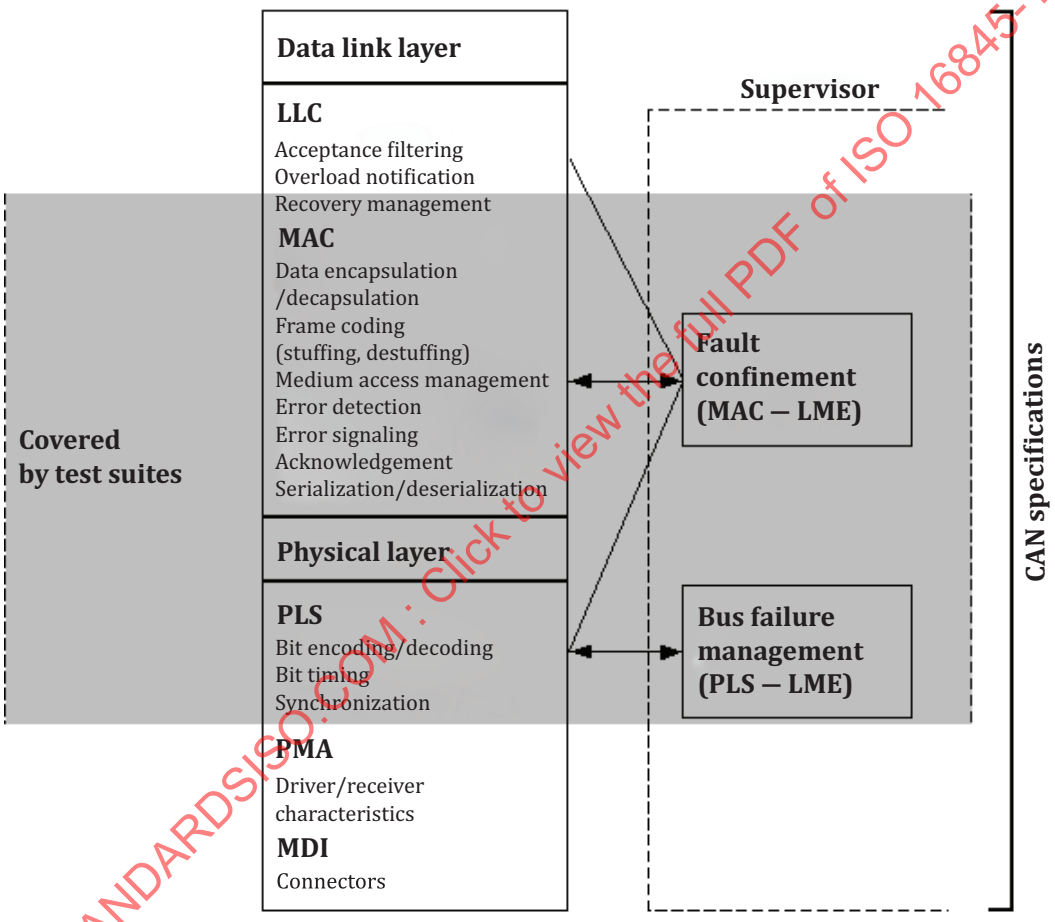

Figure 1 — Architecture of the test plan

5.2 Architecture of test plan

This methodology and the associated abstract test suites will be hereafter referred to as test plan (TP).

The TP is a specific application of the «OSI conformance testing general concepts» ISO 9646-1 and is restricted to the single party testing mode. Since the upper service boundary of a CAN implementation is not standardized and in some cases may not be observed and controlled [due to an application specific behaviour embedded in this implementation, for example, CAN SLIO (serial linked input/output)], the TP will rely either on the «coordinated test method» or the «remote test method».

Depending on the test method applied, the TP will involve up to the following three test functions:

— a lower tester (LT) operating in a way similar to the CAN implementation to be tested (IUT), running test suite and granting test verdict;

— an upper tester (UT) acting as user of the IUT (IUT dependant);

— a test management protocol between the IUT and the LT. The protocol consists in test coordination procedures.

The last two functions are only applicable to the coordinated test procedure.

During test execution, the LT can observe and control the standardized lower service boundary of the IUT (PCO) through the two service primitives provided by the CAN physical signalling sub-layer: PLS-Data.indicate and PLS-Data.request in most cases.

The environment that implements the TP is described in Figure 2.

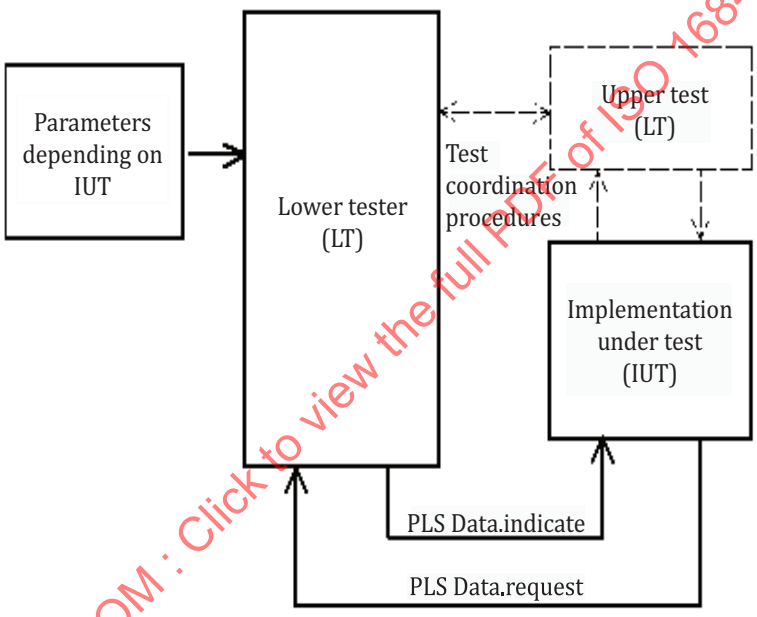

Figure 2 — CAN conformance TP environment

Using the network interface, the LT indicates to the UT the actions to be performed and the UT provides the LT with information concerning the internal behaviour of the IUT.

In order to allow the LT and the UT to communicate, it is necessary to define some test coordination procedures between them. These procedures use the network to the exclusion of any other physical link. They are used to set up the UT and to verify the test results.

5.3 Organization

5.3.1 General organization

The LT verifies if the IUT complies with the MAC, LLC and PLS sub-layers of ISO 11898-1. The LT points out differences between what is expected from the standard and the actual behaviour of the IUT.

The abstract test suites of the TP are independent to one another. Each abstract test suite checks the behaviour of the IUT for a particular parameter of ISO 11898-1. Each test case may be executed one after another in any order or alone.

Test cases requiring variations of individual parameters (identifier, number of data, etc.) should be repeated for each value of the parameter. Each repetition is named elementary test. A test case including different elementary tests is valid only if all tests pass.

5.3.2 Test case organization

5.3.2.1 General

Each elementary test is made of the following three states:

— set-up state;

— test state;

— verification state.

At the PCO, these states involve exchanges of valid sequences of PLS service primitives [CAN frame(s)] or invalid sequences of PLS primitives (invalid CAN frames or noise).

Before the first elementary test is started, the IUT has to be initialized into the default state.

5.3.2.2 Set-up state

The set-up state is the state in which the IUT has to be before entering the test state.

5.3.2.3 Test state

This is the part of the elementary test in which the parameter or protocol feature is actually checked. This state needs one or several exchanges or frames. These frames are named test frames.

5.3.2.4 Verification state

Verification state is made of the data reading frames which verify that the data have been handled in accordance with ISO 11898-1. These data should be checked.

For tests belonging to classes 1 to 6, the LT should be able to detect the correct value of the bit. For bit timing tests (classes 7 and 8), the LT should be able to detect a faulty synchronization of one time quantum.

For tests belonging to class 6, a readable error counter should be used for verification. In case the error counter value is not readable, the test should be applied by driving the IUT to the next error state by additional bus errors. The state change at expected position indicates the correct error counting up to this state change.

5.3.2.5 Default state

The default state is characterized by the following default value:

— both REC and TEC should be equal to 0;

— no pending transmission should be present;

— IUT should be in idle state;

— PLS-Data indicate and PLS-Data request should be recessive.

After the end of each elementary test, the default state should be applied.

5.3.3 Hierarchical structure of tests

5.3.3.1 Overview

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:

— test types;

— test classes;

— test cases.

5.3.3.2 Test types

The types define the direction of the frames. The three types are as follows.

— **Type 1: Received frame type**

It includes all the tests evaluating the behaviour 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 behaviour 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.

5.3.3.3 Test classes

Each of the three test types defined in 5.3.3.2 is divided in eight classes grouping tests by topic regarding to ISO 11898-1. These eight classes are as follows.

— **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 behaviour 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 Classical CAN frame format**

This class includes the tests verifying the correct management of bit timing by the IUT. This class should only be applied to components performing only recessive to dominant edge synchronization (if the recessive to dominant edge synchronization exists, it should be disabled).

— **Class 8: Bit timing CAN FD frame format**

This class includes the tests verifying the correct management of bit timing by the IUT. This class should only be applied to components performing only recessive to dominant edge synchronization (if the recessive to dominant edge synchronization exists, it should be disabled).

5.3.3.4 Test cases

Any basic entry of the test list is intended to check a particular parameter of ISO 11898-1 in the IUT.

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 test.

6 LT parameters

6.1 Overview

ISO 11898-1 allows several IUT implementations. Consequently, the user should provide the LT with parameters in order to indicate which kind of IUT is going to be tested.

These parameters can be classified in two categories.

— **Communication parameters**

This category defines which tests can be executed for the IUT, and which test method will be applied.

— **Application parameters**

This category defines the features of the frames used for each test case selected according to the previous parameters.

6.2 Description of parameters

6.2.1 Communication parameters

6.2.1.1 Categories of communication parameter

These parameters are subdivided in three categories:

— implementation parameters;

— timing parameters;

— other parameters.

In this document, they are always written in upper case letters.

6.2.1.2 Implementation classes

Some parameters depending on the IUT should be specified by the user in order to allow the LT to match the IUT. These parameters are as follows.

a) CAN_VERSION

This parameter indicates the version implemented in the IUT. It can take three values.

— Classical CAN

A Classical CAN node that cannot receive or transmit frames in CAN FD frame format and it disturbs them by error treatment.

— CAN FD tolerant

A Classical CAN compatible node that cannot receive or transmit frames in CAN FD frame format, but does not disturb them.

— CAN FD enabled

A CAN node that is able to receive or transmit in CAN FD frame format as well as in Classical CAN frame format.

b) OPEN/SPECIFIC

This parameter indicates if the IUT is open regarding to the application layers, or if it includes a specific application. It can take two values.

— OPEN

Open IUT allowing the test coordination procedure to be implemented in an UT. These IUT are tested with the «coordinated test method» of ISO/IEC 9646-1.

— SPECIFIC

IUT that can be tested only with the help of a specific configuration procedure. These IUT are tested with the «remote test method» of ISO/IEC 9646-1.

6.2.1.3 Timing parameters

The LT also needs some timing parameters to be in accordance with the IUT and the UT characteristics. These parameters are as follows.

a) TIMEOUT

This parameter indicates the minimum duration time for which the LT should wait in order to respect the three following conditions:

— the UT should have enough time to put the IUT into the set-up state;

— the IUT should have enough time to transmit a response frame after a remote frame;

— the LT should consider an optional additional waiting time after the end of the minimum bus-off recovery sequence before the IUT enters error-active state again.

b) Bit timing parameter as defined in ISO 11898-1.

6.2.2 Application parameters

Except the test cases for which a particular profile of data is defined by the TP, the ID, DLC and content of the data used during the test cases should be chosen by the user and is not relevant for test purpose.

CAN FD format frames are configured with BRS = 1 and ESI = 0 if not defined by the TP.

6.2.3 Bit rate configuration parameter variation for bit timing tests

The configuration range for the number of time quanta and the resultant sampling point is IUT-specific. A variation of bit rate configuration setups used for bit timing tests shall be done according the IUT's individual configuration range.

For Classical CAN configurations:

— at least one configuration for minimum programmable number of TQ(N)s (ISO 11898-1:2015, Table 8);

— at least one configuration for maximum programmable number of TQ(N)s;

— at least one configuration for a programmable number of TQ(N)s in the middle between MIN and MAX number of TQ(N)s.

For FD enabled configuration:

— test with shared prescaler (different number of TQs for nominal and data bit rate) and if supported, test with separate prescalers (constant number of TQs for nominal and data bit rate and different TQ time);

— at least one configuration for minimum possible number of TQ(D)s;

— at least one configuration for maximum possible number of TQ(D)s;

— at least one configuration for a possible number of TQ(D)s in the middle of MIN and MAX.

If enabled, the transmitter delay compensation could be implemented with the following two options for configuring the delay compensation time in TQ(D)s:

— the IUT may use a preconfigured fix value to describe the SSP position;

— the IUT can do an internal-edge to external-edge measurement to determine the real delay plus a preconfigured fix offset to describe the SSP position relative to the delay measurement.

There shall be one elementary test to perform for each possible way of configuration of delay compensation (fix programmed or automatically measured).

"Delay compensation" + "programmed offset" shall lead to a value similar like "applied delay" + Sync_Seg(D) + Prop_Seg(D) + Phase_Seg1(D), so that the SSP evaluate the delayed bit at the same position inside the delayed bit as the programmed sampling point describe it for a non-delayed bit.

For test cases using TDC, the bit timings need to be selected so that [Sync_Seg(N) + Prop_Seg(N) + Phase_Seg1(N)] > 2 data bit times.

7 Test type 1, received frame

7.1 Test class 1, valid frame format

7.1.1 Identifier and number of data test in base format

Table 1 specifies the test of the correct reception of frames in base frame format.

Table 1 — Identifier and number of data test in base format

Item	Description	
Purpose	This test verifies the behaviour of the IUT when receiving a correct data frame with different identifiers and different numbers of data bytes in base format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID DLC FDF = 0	ID DLC FDF = 1
Elementary test cases	The CAN ID shall be element of: $\in [000_h, 7FF_h]$ Different CAN IDs are used for test. #1 CAN ID = 555_h #2 CAN ID = $2AA_h$ #3 CAN ID = 000_h #4 CAN ID = $7FF_h$ #5 CAN ID = a random value Tested number of data bytes: $\in [0, 8]$ Number of tests: 45	The CAN ID will be element of: $\in [000_h, 7FF_h]$ Different CAN IDs are used for test. #1 CAN ID = 555_h #2 CAN ID = $2AA_h$ #3 CAN ID = 000_h #4 CAN ID = $7FF_h$ #5 CAN ID = a random value Tested number of data bytes: $\in [0, 8] \cup [12] \cup [16] \cup [20] \cup [24] \cup [32] \cup [48] \cup [64]$ Number of tests: 80
Set-up	The IUT is left in the default state.	
Execution	The test system sends a frame with ID and DLC as specified in elementary test cases definition.	
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data received by the IUT during the test state should match the data sent in the test frame.	
NOTE	An implementation with limited payload capabilities will be tested in range of their payload capabilities.	

7.1.2 Identifier and number of data test in extended format

Table 2 specifies the test of the correct reception of frames in extended frame format.

Table 2 — Identifier and number of data test in extended format

Item	Description	
Purpose	This test verifies the behaviour of the IUT when receiving a correct data frame with different identifiers and different numbers of data bytes in extended format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID DLC FDF = 0	ID DLC FDF = 1

Table 2 *(continued)*

Item	Description	
Elementary test cases	The CAN ID shall be element of: ∈ [00000000$_h$, 1FFFFFFF$_h$] Different CAN IDs are used for test. #1 CAN ID = 15555555$_h$ #2 CAN ID = 0AAAAAAA$_h$ #3 CAN ID = 00000000$_h$ #4 CAN ID = 1FFFFFFF$_h$ #5 CAN ID = random value Tested number of data bytes: ∈ [0, 8] Number of tests: 45	The CAN ID shall be element of: ∈ [00000000$_h$, 1FFFFFFF$_h$] Different CAN IDs are used for test. #1 CAN ID = 15555555$_h$ #2 CAN ID = 0AAAAAAA$_h$ #3 CAN ID = 00000000$_h$ #4 CAN ID = 1FFFFFFF$_h$ #5 CAN ID = random value Tested number of data bytes: ∈ [0, 8] ∪ [12] ∪ [16] ∪ [20] ∪ [24] ∪ [32] ∪ [48] ∪ [64] Number of tests: 80
Set-up	The IUT is left in the default state.	
Execution	The test system sends a frame with ID and DLC as specified in elementary test cases definition.	
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data received by the IUT during the test state shall match the data sent in the test frame.	
NOTE	An implementation with limited ID range may not be able to receive the frame. An implementation with limited payload capabilities will be tested in range of their payload capabilities.	

7.1.3 Reception after arbitration lost

Table 3 specifies the test of the reception of arbitration winning frame, while the IUT loses the arbitration.

Table 3 — Reception after arbitration lost

Item	Description	
Purpose	This test verifies the capability of the IUT to manage the reception of arbitration winning frame, while the IUT loses the arbitration.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID all bit = 1 IDE SRR (in case of IDE = 1) FDF = 0 DLC = 0 RTR = 1	ID all bit = 1 IDE SRR (in case of IDE = 1) FDF DLC = 0 RTR = 1

Table 3 *(continued)*

Item	Description							
Elementary test cases		LT frame format	IUT frame format	Bit for arb. lost		LT frame format	IUT frame format	Bit for arb. lost
	#1	CBFF	CBFF	RTR	#1	CBFF	FBFF	LSB base ID
	#2	CBFF	CEFF	SRR	#2	FBFF	CBFF	RTR
	#3	CBFF	CEFF	IDE	#3	CEFF	FEFF	LSB ID ext.
	#4	CEFF	CBFF	LSB base ID	#4	FEFF	CEFF	RTR
	#5	CEFF	CEFF	LSB ID ext.				
	#6	CEFF	CEFF	RTR				
Set-up	The IUT is left in the default state.							
Execution	The LT causes the IUT to transmit a frame according to "IUT frame format" in elementary test cases. Then, the LT forces the bit described at "bit for arbitration lost" in elementary test cases to dominant state and continues to send a valid frame according to elementary test cases.							
Response	The IUT shall become the receiver when sampling the dominant bit sent by the LT. The frame received by the IUT shall match the frame sent by the LT. As soon as the bus is idle again, the IUT shall restart the transmission of the frame. The IUT shall not generate any error flag during the test. The content of the frame shall match the LT request.							
NOTE	An implementation with limited ID range may not be able to transmit/receive the frame.							

7.1.4 Acceptance of non-nominal bit in base format frame

Table 4 specifies the test of an IUT that accepts the non-nominal bit value in a valid base format frame.

Table 4 — Acceptance of non-nominal in base format frame

Item	Description			
Purpose	The purpose of this test is to verify that the IUT accepts the non-nominal value of bit described in test variables in a valid base format frame.			
CAN_VERSION	Classical CAN		CAN FD enabled	
Test variables	FDF		RRS FDF = 1	
Elementary test cases	**Test**	**FDF**	**Test**	**RRS**
	#1	1	#1	1
Set-up	The IUT is left in the default state.			
Execution	A single test frame is used for the elementary test cases.			
Response	The IUT shall not generate any error flag in this test frame. The IUT shall acknowledge the test frame. The data received by the IUT during the test state shall match the data sent in the test frame.			

7.1.5 Acceptance of non-nominal bit in extended format frame

Table 5 specifies the test of an IUT that accepts the non-nominal bit value in a valid extended frame.

Table 5 — Acceptance of non-nominal in extended format frame

Item	Description									
Purpose	The purpose of this test is to verify that the IUT accepts the non-nominal value of bit described in test variables in a valid extended format frame.									
CAN_VERSION	Classical CAN				CAN FD Tolerant			CAN FD enabled		
Test variables	SRR FDF r0				SRR r0 FDF = 0			SRR RRS FDF = 1		
Elementary test cases	**TEST**	**SRR**	**r0**	**FDF**	**TEST**	**SRR**	**r0**	**TEST**	**SRR**	**RRS**
	#1	1	1	1	#1	1	1	#1	1	1
	#2	1	1	0	#2	0	1	#2	0	1
	#3	1	0	1	#3	0	0	#3	0	0
	#4	0	1	1						
	#5	0	1	0						
	#6	0	0	1						
	#7	0	0	0						
Set-up	The IUT is left in the default state.									
Execution	A single test frame is used for each of the elementary tests.									
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data received by the IUT during the test state shall match the data sent in the test frame.									
NOTE	An implementation with limited ID range may not be able to receive the frame.									

7.1.6 Protocol exception behaviour on non-nominal bit

Table 6 specifies the test of an IUT that switches to protocol exception on non-nominal bit value

Table 6 — Protocol exception behaviour

Item	Description									
Purpose	The purpose of this test is to verify that the IUT switches to protocol exception on non-nominal values of the bits described in test variables.									
CAN_VERSION	CAN FD Tolerant					CAN FD enabled				
Test variables	FDF = 1 DLC Data: All data byte with the same value Bit rate ratio between nominal and data bit rate					FDF = 1 “res” bit = 1 DLC Data: All data byte with the same value Bit rate ratio between nominal and data bit rate				
Elementary test cases	**Test**	**Format**	**DLC**	**data**	**bit rate ratio**	**Test**	**Format**	**DLC**	**data**	**bit rate ratio**
	#1	FBFF	A_h;	AA_h;	1:2	#1	FBFF	A_h;	AA_h;	1:2
	#2	FBFF	F_h;	FF_h;	1:8	#2	FBFF	F_h;	FF_h;	1:8
	#3	CBFF[a]	F_h;	FF_h;	—	Protocol exception handling shall be enabled.				

[a] Valid classical frame with non-nominal FDF bit.

Table 6 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	A single test frame is used for the elementary test, followed immediately by a valid Classical CAN frame.
Response	The IUT shall not generate any error flag in this test frame. The IUT shall not acknowledge the test frame. A following data frame in classical frame format received by the IUT during the test state shall match the data sent in the test frame.
[a] Valid classical frame with non-nominal FDF bit.	

7.1.7 Minimum time for bus idle after protocol exception handling

Table 7 specifies the test of the correct reception behaviour after protocol exception (IDLE detection).

Table 7 — Minimum time for bus idle after protocol exception

Item	Description	
Purpose	This test verifies the behaviour of an IUT in protocol exception state when receiving frames separated by different times for inter-frame space.	
CAN_VERSION	CAN FD tolerant	CAN FD enabled
Test variables	Intermission field length	Intermission field length Protocol exception handling shall be enabled
Elementary test cases	#1 The second frame starts after the third intermission bit + 1 bit time after the first frame. #2 The second frame starts after the third intermission bit of the first frame. #3 The second frame starts after the second intermission bit of the first frame followed by a third frame starts after the third intermission bit of the previous frame.	
Set-up	The IUT is left in the default state.	
Execution	The LT send a frame with non-nominal bit in control field causing protocol exception behaviour. The LT send a valid classical frame according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The IUT shall only acknowledge the last test frame in each test sequence.	

7.1.8 DLC greater than 8

Table 8 specifies the test of correct reception of classical frame with a DLC field greater than 8.

Table 8 — DLC greater than 8

Item	Description
Purpose	This test verifies the behaviour of the IUT when receiving a correct classical frame with a DLC greater than 8.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	DLC FDF = 0

Table 8 *(continued)*

Item	Description
Elementary test cases	There are seven elementary tests, for which DLC ∈ [9_h, F_h]. **DLC** #1 9_h #2 A_h #3 B_h #4 C_h #5 D_h #6 E_h #7 F_h
Set-up	The IUT is left in the default state.
Execution	A single test frame is used for each of the elementary tests.
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data and DLC received by the IUT during the test state shall match the data and DLC sent in the test frame.

7.1.9 Absent bus idle — Valid frame reception

Table 9 specifies the test of the correct reception of two consecutive frames which are not separated by a bus idle state.

Table 9 — Absent bus idle — Valid frame reception

Item	Description	
Purpose	This test verifies the behaviour of the IUT when receiving two consecutive frames not separated by a bus idle state.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission field length FDF = 0	Intermission field length FDF = 1
Elementary test cases	#1 The second frame starts after the second intermission bit of the first frame. #2 The second frame starts after the third intermission bit of the first frame.	
Set-up	The IUT is left in the default state.	
Execution	Two different test frames are used for each of the two elementary tests.	
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frames.	

7.1.10 Stuff acceptance test in base format frame

Table 10 specifies the test of the correct reception of frames in base frame format with particular data containing critical stuffing bit profiles in the different fields of the frame.

Table 10 — Stuff acceptance test in base format frame

Item	Description											
Purpose	This test verifies the behaviour of the IUT when receiving a correct base format frame with particular data containing critical stuffing bit profiles in the different fields of the frame according to test variables.											
CAN_VERSION	Classical CAN				CAN FD tolerant CAN FD enabled				CAN FD enabled			
Test variables	ID RTR FDF DLC DATA				ID RTR DLC DATA				ID RRS BRS ESI DLC DATA			
Elementary test cases			CBFF				CBFF				FBFF	
		ID	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**
	#1	078_h	08_h	01_h, all others $E1_h$	#1	078_h	08_h	01_h, all others $E1_h$	#1	078_h	$0AE_h$	$F8_h$ all others 78_h
	#2	$41F_h$	01_h	00_h	#2	$41F_h$	01_h	00_h	#2	$47C_h$	$0A8_h$	all bytes $3C_h$
	#3	707_h	$1F_h$	all bytes $0F_h$	#3	707_h	$0F_h$	all bytes $0F_h$	#3	$41E_h$	$0BE_h$	all bytes $1E_h$
	#4	360_h	10_h	—	#4	360_h	00_h	—	#4	$20F_h$	$09F_h$	all bytes $0F_h$
	#5	730_h	10_h	—	#5	730_h	00_h	—	#5	107_h	$28F_h$	all bytes 87_h
	#6	$47F_h$	01_h	$1F_h$	#6	$47F_h$	01_h	$1F_h$	#6	$7C3_h$	083_h	all bytes $C3_h$
	#7	758_h	00_h	—	#7	758_h	00_h	—	#7	$3E1_h$	$0A3_h$	all bytes $E1_h$
	#8	777_h	01_h	$1F_h$	#8	777_h	01_h	$1F_h$	#8	$1F0_h$	$0A1_h$	all bytes $F0_h$
	#9	$7EF_h$	42_h	—	#9	$7Ef_h$	42_h	—	#9	000_h	$0A0_h$	—
	#10	$3EA_h$	$5F_h$	—	#10	$3Ea_h$	$4F_h$	—	#10	$7FF_h$	$0B0_h$	—
Set-up	The IUT is left in the default state.											
Execution	A single test frame is used for each elementary test.											
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data received by the IUT during the test state shall match the data sent in the test frame.											

7.1.11 Stuff acceptance test in extended format frame

Table 11 specifies the test of the correct reception of frames in extended frame with particular data containing critical stuffing bit profiles in the different fields of the frame.

Table 11 — Stuff acceptance test in extended format frame

Item	Description											
Purpose	This test verifies the behaviour of the IUT when receiving a correct extended frame with particular data containing critical stuffing bit profiles in the different fields of the frame according to test variables.											
CAN_VERSION	Classical CAN				CAN FD tolerant CAN FD enabled				CAN FD enabled			
Test variables	ID SRR RTR FDF r0 DLC DATA				ID SRR RTR DLC DATA FDF = 0				ID SRR RRS BRS ESI DLC DATA FDF = 1			
Elementary test cases			CEFF				CEFF				FEFF	
		ID	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**
	#1	$07C30F0F_h$	188h	all bytes $3C_h$	#1	$07C30F0F_h$	188_h	all bytes $3C_h$	#1	$01E38787_h$	$6AE_h$	$F8_h$
	#2	$07C0F0F0_h$	181_h	00_h	#2	$07C0F0F0_h$	181_h	00_h				all others 78_h
	#3	$01E31717_h$	$19F_h$	all bytes $0F_h$	#3	$01E31717_h$	$19F_h$	all bytes $0F_h$	#2	$11F3C3C3_h$	$2A8_h$	all bytes $3C_h$
	#4	$01E00FF0_h$	$1BC_h$	$1F_h$, $0F_h$, $E0_h$, $F0_h$, $7F_h$, $E0_h$, FF_h, 20_h	#4	$01E00FF0_h$	$19C_h$	$1F_h$, $0F_h$, $E0_h$, $F0_h$, $7F_h$, $E0_h$, FF_h, 20_h	#3	$1079C1E1_h$	$6BE_h$	all bytes $1E_h$
									#4	$083DF0F0_h$	$69F_h$	all bytes $0F_h$
									#5	$041EF878_h$	$68F_h$	all bytes 87_h
									#6	$1F0C3C3C_h$	683_h	all bytes $C3_h$
	#5	$1FB80000_h$	181_h	$A0_h$	#5	$1FB80000_h$	181_h	$A0_h$	#7	$0F861E1E_h$	$6A3_h$	all bytes $E1_h$
	#6	$00BC540F_h$	$1E0_h$	—	#6	$00BC540F_h$	$1C0_h$	—	#8	$07C30F0F_h$	$6A1_h$	all bytes $F0_h$
	#7	$155D5557_h$	$1FF_h$	—	#7	$155D5557_h$	$1DF_h$	—	#9	$01E38787_h$	$3A0_h$	—
									#10	$11F3C3C3_h$	380_h	—
									#11	00000000_h	$6B0_h$	—
Set-up	The IUT is left in the default state.											
Execution	A single test frame is used for each of the elementary tests.											
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The data received by the IUT during the test state shall match the data sent in the test frame.											
NOTE	An ID limited implementation may not receive the frame.											

7.1.12 Message validation

Table 12 specifies the test of the point of time at which a message is taken to be valid.

Table 12 — Message validation

Item	Description	
Purpose	The purpose of this test is to verify the point of time at which a message is taken to be valid by the IUT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	EOF FDF = 0	EOF FDF = 1
Elementary test cases	Number of elementary tests: 1 #1 The last bit of the EOF is forced to dominant state.	
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for the elementary test.	
Response	The IUT shall not generate any error flag during the test. The IUT shall acknowledge the test frame. The IUT shall generate an overload frame. The data received by the IUT during the test state shall match the data sent in the test frame.	

7.2 Test class 2, error detection

7.2.1 Bit error in data frame

Table 13 specifies the test of an IUT that detects a bit error in data frame when the dominant ACK slot is forced to recessive state by LT.

Table 13 — Bit error in classical data frame

Item	Description	
Purpose	This test verifies that the IUT detects a bit error when the dominant ACK slot is forced to recessive state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ACK Slot FDF = 0	ACK Slot FDF = 1
Elementary test cases	Number of elementary test cases: 1 #1 The dominant acknowledgement bit sent by the IUT is forced to recessive state.	
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for the elementary test.	
Response	The IUT shall generate an active error frame starting at the bit position following the bit error.	
NOTE	The receiver will generate an ACK immediately after CRC delimiter.	

7.2.2 Stuff error for basic frame

Table 14 specifies the test of a stuff error whenever it receives 6 consecutive bits of the same value until the position of the CRC delimiter in a base format frame.

Table 14 — Stuff error for basic frame

Item	Description											
Purpose	This test verifies that the IUT detects a stuff error whenever it receives 6 consecutive bits of the same value until the position of the CRC delimiter in a base format frame.											
CAN_VERSION	Classical CAN				CAN FD tolerant CAN FD enabled				CAN FD enabled			
Test variables	ID RTR FDF DLC DATA				ID RTR DLC DATA				ID RTR BRS ESI DLC DATA byte 0 defined, all others = 55_h			
Elementary test cases	All 63 stuff bits within the defined frames will be tested.				All 59 stuff bits within the defined frames will be tested.				All 40 stuff bits within the defined frames will be tested.			
			CBFF				CBFF				FBFF	
		ID	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**		**ID**	**CTRL**	**DATA**
	#1	078_h	08_h	01_h, all others $E1_h$	#1	078_h	08_h	01_h, all others $E1_h$	#1	078_h	$0AE_h$	$F8_h$,
	#2	$41F_h$	01_h	00_h	#2	$41F_h$	01_h	00_h	#2	$47C_h$	$0A8_h$	$3C_h$
	#3	707_h	$1F_h$	all byte $0F_h$	#3	707_h	$0F_h$	all byte 87_h	#3	$41E_h$	$0BE_h$	$1E_h$
	#4	360_h	10_h	—	#4	360_h	00_h	—	#4	$20F_h$	$09F_h$	$0F_h$
	#5	730_h	10_h	—	#5	730_h	00_h	—	#5	107_h	$28F_h$	87_h
	#6	$47F_h$	01_h	$1F_h$	#6	$47F_h$	01_h	$1F_h$	#6	$7C3_h$	083_h	$C3_h$
	#7	758_h	00_h	—	#7	758_h	00_h	—	#7	$3E1_h$	$0A3_h$	$E1_h$
	#8	777_h	01_h	$1F_h$	#8	777_h	01_h	$1F_h$	#8	$1F0_h$	$0A1_h$	$F0_h$
	#9	$7EF_h$	42_h	—	#9	$7EF_h$	42_h	—	#9	000_h	$0A0_h$	—
	#10	$3EA_h$	$5F_h$	—	#10	$3EA_h$	$4F_h$	—	#10	$7FF_h$	$0B0_h$	—
Set-up	The IUT is left in the default state.											
Execution	A single test frame is used for each elementary test. The LT forces one of the stuff bits to its complement.											
Response	The IUT shall generate an active error frame starting at the bit position following the stuff error.											

7.2.3 Stuff error for extended frame

Table 15 specifies the test of a stuff error whenever it receives 6 consecutive bits of the same value until the position of the CRC delimiter in an extended frame.

Table 15 — Stuff error for extended frame

Item	Description		
Purpose	This test verifies that the IUT detects a stuff error whenever it receives 6 consecutive bits of the same value until the position of the CRC delimiter in an extended frame.		
CAN_VERSION	Classical CAN	CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	basic ID SRR ID extension RTR FDF r0 DLC DATA	basic ID SRR ID extension RTR FDF = 0 r0 DLC DATA	basic ID SRR ID extension RRS FDF = 1 res BRS ESI DLC DATA byte 0 defined, all others = 55_h
Elementary test cases	There are 90 elementary tests to perform.	There are 85 elementary tests to perform.	There are 88 elementary tests to perform.

CEFF

	ID	CTRL	DATA
#1	$07C30F0F_h$	188_h	all 3Ch
#2	$07C0F0F0_h$	181_h	00h
#3	$01E31717_h$	$19F_h$	all $0F_h$,
#4	$01E00FF0_h$	$1BC_h$	$1F_h$, $0F_h$, $E0_h$, $F0_h$, $7F_h$, $E0_h$, FF_h, 20_h
#5	$1FB80000_h$	181_h	$A0_h$
#6	$00BC540F_h$	$1E0_h$	—
#7	$155D5557_h$	$1FF_h$	—
#8	00000000_h	181_h	00_h

CEFF

	ID	CTRL	DATA
#1	$07C30F0F_h$	188_h	all $3C_h$
#2	$07C0F0F0_h$	181_h	00_h
#3	$01E31717_h$	$19F_h$	all $0F_h$,
#4	$01E00FF0_h$	$19C_h$	$1F_h$, $0F_h$, $E0_h$, $F0_h$, $7F_h$, $E0_h$, FF_h, 20_h
#5	$1FB80000_h$	181_h	$A0_h$
#6	$00BC540F_h$	$1C0_h$	—
#7	$155D5557_h$	$1DF_h$	—
#8	00000000_h	181_h	00_h

FEFF

	ID	CTRL	DATA
#1	$01E38787_h$	$6AE_h$	$F8_h$
#2	$11F3C3C3_h$	$2A8_h$	$3C_h$
#3	$1079C1E1_h$	$6BE_h$	1E
#4	$083DF0F0_h$	$69F_h$	$0F_h$
#5	$041EF878_h$	$68F_h$	87_h
#6	$1F0C3C3C_h$	683_h	$C3_h$
#7	$0F861E1E_h$	$6A3_h$	$E1_h$
#8	$07C30F0F_h$	$6A1_h$	$F0_h$
#9	$01E38787_h$	$3A0_h$	—
#10	$11F3C3C3_h$	380_h	—
#11	00000000_h	$6B0_h$	—

Set-up	The IUT is left in the default state.
Execution	A single test frame is used for each elementary test. The LT forces one of the stuff bits to its complement.
Response	The IUT shall generate an active error frame starting at the bit position following the stuff error.

7.2.4 Stuff error for FD frame payload bytes

Table 16 specifies the test of a stuff error whenever the IUT receives 6 consecutive bits of the same value until the position of the CRC in an FD format frame with maximum payload DLC = 15.

Table 16 — Stuff error for CAN FD enabled implementation

Item	Description
Purpose	This test verifies that the IUT detects a stuff error whenever it receives 6 consecutive bits of the same value until the position of the CRC delimiter in a base format frame.
CAN_VERSION	CAN FD enabled
Test variables	DATA byte 0–63 ID = 555_h IDE = 0 DLC = 15 FDF = 1
Elementary test cases	All 1 008 stuff bits within the defined data bytes 1 to 63 will be tested. Data Byte 0 / Data bytes 1 – 63 #1 10_h / 78_h #2 78_h / $3C_h$ #3 34_h / $1E_h$ #4 12_h / $0F_h$ #5 $0F_h$ / 87_h #6 17_h / $C3_h$ #7 43_h / $E1_h$ #8 21_h / $F0_h$
Set-up	The IUT is left in the default state.
Execution	A single test frame is used for each elementary test. In each elementary test, the LT forces one of the stuff bits to its complement.
Response	The IUT shall generate an active error frame starting at the bit position following the stuff error.

7.2.5 CRC error

Table 17 specifies the test of an IUT that uses the specific CRC mechanism and detecting a CRC error, generates an error frame at the correct position and does not detect an error when monitoring a dominant bit at the ACK slot while sending a recessive one.

Table 17 — CRC error

Item	Description	
Purpose	The purpose of this test is to verify — that the IUT uses the specific CRC mechanism according to frame format, — that the IUT detecting a CRC error and generates an error frame at the correct position, and — that the IUT does not detect an error when monitoring a dominant bit at the ACK slot while sending a recessive one.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	CRC FDF = 0 SOF	CRC DLC - to cause different CRC types FDF = 1 SOF

Table 17 *(continued)*

Item	Description	
Elementary test cases	Number of elementary tests: 3 #1 A dominant bit in the CRC field is changed in a recessive bit. #2 A recessive bit in the CRC field is changed in a dominant bit. #3 The dominant SOF bit in the frame is changed in a recessive one followed by an ID 001_h.	Number of elementary tests: 6 #1 and #2 A dominant bit in the CRC field is changed in a recessive bit for CRC-17 with DLC ≤ 10 (#1) and CRC-21 with DLC > 10 (#2) (test for CRC value). #3 and #4 A recessive bit in the CRC field is changed in a dominant bit for CRC-17 with DLC ≤ 10 (#3) and CRC-21 with DLC > 10 (#4) (test for CRC value). #5 The test system sends a frame where two times a recessive stuff bit becomes a normal bit by losing one of the previous bits by synchronization issues while the CRC register is equal zero (test for stuff-counter). #6 The parity bit of the stuff count and the following fixed stuff bit changed into their opposite values (test for stuff-counter parity bit value).
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for each elementary test. The LT modifies the frame according to elementary test cases.	
Response	The IUT shall not acknowledge the test frame. The IUT shall generate an active error frame starting at the first bit position following the ACK delimiter.	The IUT shall not acknowledge the test frame. The IUT shall generate an active error frame starting at the fourth bit position following the CRC delimiter.

7.2.6 Combination of CRC error and form error

Table 18 specifies the test of an IUT that detects a CRC error and a form error on the CRC delimiter in the same frame.

Table 18 — Combination of CRC error and form error

Item	Description	
Purpose	The purpose of this test is to verify that an IUT detecting a CRC error and a form error on the CRC delimiter in the same frame generates only one single 6 bits long error flag starting on the bit following the CRC delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	CRC FDF = 0	CRC DLC - to cause different CRC types FDF = 1
Elementary test cases	Number of elementary tests: 1 #1 CRC (15)	Number of elementary tests: 2 #1 DLC ≤ 10 – > CRC (17) #2 DLC > 10 – > CRC (21)

Table 18 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	A single test frame is used for the elementary test. The LT generates a CAN frame with CRC error and form error at CRC delimiter according to elementary test cases.
Response	The IUT shall generate one active error frame starting at the bit position following the CRC delimiter.

7.2.7 Form error in data frame at "CRC delimiter" bit position

Table 19 specifies the test of an IUT that detects a form error when the recessive bit of CRC delimiter is forced to dominant state.

Table 19 — Form error in data frame at "CRC delimiter" bit position

Item	Description	
Purpose	This test verifies that the IUT detects a form error when the recessive bit of CRC delimiter is forced to dominant state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	CRC delimiter FDF = 0	CRC delimiter FDF = 1
Elementary test cases	There is only one elementary test to perform. #1 CRC delimiter = 0	
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for the elementary test. The LT generates a CAN frame with form error at CRC delimiter according to elementary test cases.	
Response	The IUT shall generate an active error frame at the bit position following the CRC delimiter.	

7.2.8 Form error at fixed stuff bit in FD frames

Table 20 specifies the test of an IUT that detects a form error when a fixed stuff bit did not match to the previous bit.

Table 20 — Form error at fixed stuff bit in FD frames

Item	Description
Purpose	This test verifies that the IUT detects a form error when a fixed stuff bit did not match to the previous bit.
CAN_VERSION	CAN FD enabled
Test variables	DLC - to cause different CRC types FDF = 1

Table 20 *(continued)*

Item	Description
Elementary test cases	There are 22 elementary tests to perform. Tests to perform on recessive stuff bits: #1 DLC ≤ 10 – > CRC (17) field – (6 bits) #2 DLC > 10 – > CRC (21) field – (7 bits) Tests to perform on dominant stuff bits: #3 DLC ≤ 10 – > CRC (17) field – (6 bits) #4 DLC > 10 – > CRC (21) field – (7 bits)
Set-up	The IUT is left in the default state.
Execution	The LT corrupts a fixed stuff bit according to elementary test cases.
Response	The IUT shall generate an error frame at the bit position following the stuff bit.

7.2.9 Form error in data frame at "ACK delimiter" bit position

Table 21 specifies the test of an IUT that detects a form error when the recessive ACK delimiter is forced to dominant state.

Table 21 — Form error in data frame at "ACK delimiter" bit position

Item	Description	
Purpose	This test verifies that the IUT detects a form error when the recessive ACK delimiter is forced to dominant state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ACK delimiter ACK = 0 FDF = 0	ACK delimiter ACK1 = 0 ACK2 = 0 FDF = 1
Elementary test cases	There is only one elementary test to perform. #1 ACK delimiter = 0	
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for the elementary test. The LT generates a CAN frame with form error at ACK delimiter according to elementary test cases.	
Response	The IUT shall generate an active error frame at the bit position following the ACK delimiter.	

7.2.10 Form error in data frame at "EOF"

Table 22 specifies the test of an IUT that detects a form error when one of 6 first recessive bits of EOF is forced to dominant state.

Table 22 — Form error in data frame at "EOF"

Item	Description	
Purpose	This test verifies that the IUT detects a form error when one of 6 first recessive bits of EOF is forced to dominant state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	EOF FDF = 0	EOF FDF = 1
Elementary test cases	There are five elementary tests to perform: #1 to #5 corrupting the first until the fifth bit position.	
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for the elementary test. The LT generates a CAN frame with form error at EOF according to elementary test cases.	
Response	The IUT shall generate an active error frame at the bit position following the corrupted bit.	

7.2.11 Message non-validation

Table 23 specifies the test of the point of time at which a message is still considered as non-valid by the IUT.

Table 23 — Message non-validation

Item	Description	
Purpose	The purpose of this test is to verify the point of time at which a message is still considered as non-valid by the IUT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	EOF FDF = 0	EOF FDF = 1
Elementary test cases	There is only one elementary test to perform. #1 The sixth bit of the EOF is forced to dominant.	
Set-up	The IUT has to be initialized with data different from those used in the test frame.	
Execution	A single test frame is used for the elementary test. The LT generates a CAN frame with form error at EOF according to elementary test cases.	
Response	The IUT shall generate an active error frame. The data initialized during the set-up state shall remain unchanged. No frame reception shall be indicated to the upper layers of the IUT.	

7.3 Test class 3, error frame management

7.3.1 Error flag longer than 6 bits

Table 24 specifies the test that the IUT tolerates up to 7 consecutive dominant bits after sending an active error flag.

Table 24 — Error flag longer than 6 bits

Item	Description	
Purpose	This test verifies that the IUT tolerates up to 7 consecutive dominant bits after sending an active error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 lengthening the error flag by 1 dominant bit; #2 lengthening the error flag by 4 dominant bits; #3 lengthening the error flag by 7 dominant bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field. The LT lengthens the error flag generated by the IUT according to elementary test cases.	
Response	After sending the active error flag, the IUT sends recessive bits.	

7.3.2 Data frame starting on the third bit of intermission field

Table 25 specifies the test of an IUT that accepts a frame starting after the second bit of the intermission following the error frame it has transmitted.

Table 25 — Data frame starting on the third bit of intermission field

Item	Description	
Purpose	The purpose of this test is to verify that an IUT accepts a frame starting after the second bit of the intermission following the error frame it has transmitted.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Frame is started 2 bits after the end of the error delimiter.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field. The LT sends a valid frame according to elementary test cases.	
Response	The IUT shall acknowledge the test frame in data field. The data received by the IUT during the test state shall match the data sent in the test frame.	

7.3.3 Bit error in error flag

Table 26 specifies the test of a bit error in error flag when one of the 6 dominant bits of the error flag the IUT transmits is forced to recessive state.

Table 26 — Bit error in error flag

Item	Description	
Purpose	This test verifies that the IUT detects a bit error when one of the 6 dominant bits of the error flag it transmits is forced to recessive state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There are three elementary tests to perform: #1 corrupting the first bit of the error flag; #2 corrupting the third bit of the error flag; #3 corrupting the sixth bit of the error flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field. The LT forces one of the bits of the error frame generated by the IUT to recessive state according to elementary test cases.	
Response	The IUT shall restart with an active error frame at the bit position following the corrupted bit.	

7.3.4 Form error in error delimiter

Table 27 specifies the test of a form error when receiving an invalid error delimiter.

Table 27 — Form error in error delimiter

Item	Description	
Purpose	This test verifies that the IUT detects a form error when receiving an invalid error delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	The LT replaces one of the 8 recessive bits of the error delimiter by a dominant bit. Elementary tests to perform: #1 corrupting the second bit of the error delimiter; #2 corrupting the fourth bit of the error delimiter; #3 corrupting the seventh bit of the error delimiter.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field. The LT forces one of the bits of the error delimiter generated by the IUT to dominant state according to elementary test cases.	
Response	The IUT shall restart with an active error frame at the bit position following the replaced bit.	

7.4 Test class 4, overload frame management

7.4.1 MAC overload generation during intermission field

Table 28 specifies the test of a MAC overload generation frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field.

Table 28 — MAC overload generation during intermission field

Item	Description	
Purpose	This test verifies that the IUT generates an overload frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There are two elementary tests to perform: #1 first bit of intermission; #2 second bit of intermission.	
Set-up	The IUT is left in the default state.	
Execution	One test frame is used for each of the two elementary tests. The LT forces one of the 2 first bits of the intermission field of the test frame to dominant state according to elementary test cases.	
Response	The IUT generates an overload frame at the bit position following the dominant bit.	

7.4.2 Last bit of EOF

Table 29 specifies the test of a correct generated overload frame when detecting a dominant state on the last bit of EOF.

Table 29 — Last bit of EOF

Item	Description	
Purpose	This test verifies that the IUT generates an overload frame when detecting a dominant state on the last bit of EOF.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	EOF FDF = 0	EOF FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Last bit of the EOF.	
Set-up	The IUT is left in the default state.	
Execution	The LT forces 1 bit of the EOF to a dominant state according to elementary test cases.	
Response	The IUT generates an overload frame at the bit position following the dominant bit.	

7.4.3 Eighth bit of an error and overload delimiter

Table 30 specifies the test of a correct generated overload frame when detecting a dominant bit on the eighth bit of an error and overload delimiter it is transmitting.

Table 30 — Eighth bit of an error and overload delimiter

Item	Description	
Purpose	This test verifies that the IUT generates an overload frame when detecting a dominant bit on the eighth bit of an error and overload delimiter it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Error delimiter Overload delimiter FDF = 0	Error delimiter Overload delimiter FDF = 1
Elementary test cases	There are two elementary tests to perform: #1 apply error at the eighth bit of the error delimiter; #2 apply error at the eighth bit of the overload delimiter.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field or an overload frame after a data frame. The LT forces 1 bit to dominant state according to elementary test cases.	
Response	The IUT generates an overload frame starting at the bit position following the dominant bit forced by LT.	

7.4.4 Bit error in overload flag

Table 31 specifies the test of a bit error when one of the 6 dominant bits of the overload flag the IUT transmits is forced to recessive state.

Table 31 — Bit error in overload flag

Item	Description	
Purpose	This test verifies that the IUT detects a bit error when one of the 6 dominant bits of the overload flag it transmits is forced to recessive state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Overload flag FDF = 0	Overload flag FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the overload flag; #2 corrupting the third bit of the overload flag; #3 corrupting the sixth bit of the overload flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. The LT forces 1 bit of the overload flag to the recessive state according to elementary test cases.	
Response	The IUT shall generate an error frame at the bit position following the corrupted bit.	

7.4.5 Form error in overload delimiter

Table 32 specifies the test of an IUT that detects a form error when receiving an invalid overload delimiter.

Table 32 — Form error in overload delimiter

Item	Description	
Purpose	This test verifies that the IUT detects a form error when receiving an invalid overload delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Overload delimiter FDF = 0	Overload delimiter FDF = 1
Elementary test cases	The LT replaces one of the 8 recessive bits of the overload delimiter by a dominant bit. Elementary tests to perform: #1 corrupting the second bit of the overload delimiter; #2 corrupting the fourth bit of the overload delimiter; #3 corrupting the seventh bit of the overload delimiter.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. The LT forces 1 bit of the overload delimiter to the dominant state according to elementary test cases.	
Response	The IUT generates an error frame starting at the bit position following the replaced bit.	

7.4.6 MAC overload generation during intermission field following an error frame

Table 33 specifies the test of a correct generation of a MAC overload frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field.

Table 33 — MAC overload generation during intermission field following an error frame

Item	Description	
Purpose	This test verifies that the IUT generates an overload frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	intermission field FDF = 0	intermission field FDF = 1
Elementary test cases	There are two elementary tests to perform” #1 intermission field bit 1 dominant; #2 intermission field bit 2 dominant.	
Set-up	The IUT is left in the default state.	
Execution	One test frame is used for each of the two elementary tests. The LT causes the IUT to generate an error frame in data field. The LT forces one of the 2 first bits of the intermission field after the previous error delimiter of the test frame to a dominant value according to elementary test cases.	
Response	The IUT generates an overload frame at the bit position following the dominant bit.	

7.4.7 MAC overload generation during intermission field following an overload frame

Table 34 specifies the test of a correct generation of a MAC overload frame when detection a dominant bit on one of the 2 first recessive bits of the intermission field.

Table 34 — MAC overload generation during intermission field following an overload frame

Item	Description	
Purpose	This test verifies that the IUT generates an overload frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission field of overload frame FDF = 0	Intermission field of overload frame FDF = 1
Elementary test cases	There are two elementary tests to perform: #1 intermission field bit 1 dominant; #2 intermission field bit 2 dominant.	
Set-up	The IUT is left in the default state.	
Execution	One test frame is used for each of the two elementary tests. The LT causes the IUT to generate an overload frame after a data frame. The LT forces one of the 2 first bits of the intermission field after the overload delimiter of the test frame to a dominant value.	
Response	The IUT generates an overload frame at the bit position following the dominant bit.	

7.5 Test class 5, passive error state class

7.5.1 Passive error flag completion test 1

Table 35 specifies the test of an error passive IUT that considers the passive error flag as completed after the detection of 6 consecutive bits of the same value.

Table 35 — Passive error flag completion test 1

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT considers the passive error flag as completed after the detection of 6 consecutive bits of the same value.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Passive error flag FDF = 0	Passive error flag FDF = 1
Elementary test cases	Elementary tests to perform: #1 superimposing the passive error flag by an active error flag starting at the first bit; #2 superimposing the passive error flag by an active error flag starting at the third bit; #3 superimposing the passive error flag by an active error flag starting at the sixth bit.	
Set-up	The IUT is set in passive state.	
Execution	The LT causes the IUT to generate a passive error frame in data field. During the passive error flag sent by the IUT, the LT sends an active error flag according to elementary test cases. At the end of the active error flag, the LT waits for (8 + 2) bit time before sending a valid test frame.	
Response	The IUT shall acknowledge the test frame.	

7.5.2 Data frame acceptance after passive error frame transmission

Table 36 specifies the test of an error passive IUT that accepts a frame starting after the second bit of the intermission following the error frame it has transmitted.

Table 36 — Data frame acceptance after passive error frame transmission

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT accepts a frame starting after the second bit of the intermission following the error frame it has transmitted.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission field of passive error frame FDF = 0	Intermission field of passive error frame FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Error delimiter + intermission − 1 (8 + 2 bit time)	
Set-up	The IUT is set in passive state.	
Execution	The LT causes the IUT to generate a passive error frame in data field. At the end of the passive error flag, the LT waits according to elementary test cases before sending a valid test frame.	
Response	The IUT shall acknowledge the test frame.	

7.5.3 Acceptance of 7 consecutive dominant bits after passive error flag

Table 37 specifies the test of an error passive IUT that does not detect any error when detecting up to 7 consecutive dominant bits starting at the bit position following the last bit of the passive error flag.

Table 37 — Acceptance of 7 consecutive dominant bits after passive error flag

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT does not detect any error when detecting up to 7 consecutive dominant bits starting at the bit position following the last bit of the passive error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Error delimiter of passive error frame FDF = 0	Error delimiter of passive error frame FDF = 1
Elementary test cases	Elementary test to perform: #1 transmitting 1 consecutive dominant bit; #2 transmitting 4 consecutive dominant bits; #3 transmitting 7 consecutive dominant bits.	
Set-up	The IUT is set in passive state.	
Execution	The LT causes the IUT to generate a passive error frame in data field. After the passive error flag, the LT starts transmitting dominant bits according to elementary test cases. After the dominant bit sequence, the LT waits for error delimiter + intermission − 1 (8 + 2) bit time before sending a valid test frame.	
Response	The IUT shall acknowledge the test frame.	

7.5.4 Passive state unchanged on further errors

Table 38 specifies the test of an error passive IUT that does not become error active on any error detection.

Table 38 — Passive state unchanged on further errors

Item	Description	
Purpose	This test verifies that an error passive IUT does not become error active on any error detection.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Passive error frame FDF = 0	Passive error frame FDF = 1
Elementary test cases	There is one elementary test to perform. #1 LT send at least nine frames.	
Set-up	The IUT is set to passive state.	
Execution	The LT sends test frames with error condition in data field according to elementary test cases.	
Response	The IUT shall not generate any active error frame.	

7.5.5 Passive error flag completion — Test case 2

Table 39 specifies the test of an error passive IUT that restarts the passive error flag when detecting up to 5 consecutive dominant bits during its own passive error flag.

Table 39 — Passive error flag completion — Test case 2

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT restarts the passive error flag when detecting up to 5 consecutive dominant bits during its own passive error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Passive error flag FDF = 0	Passive error flag FDF = 1
Elementary test cases	Elementary tests to perform superimposing the passive error flag by the sequence of 5 dominant bits starting at #1 the first bit of the passive error flag, #2 the third bit of the passive error flag, and #3 the sixth bit of the passive error flag.	

Table 39 *(continued)*

Item	Description
Set-up	The IUT is set in passive state.
Execution	The LT causes the IUT to generate a passive error frame in data field. During the passive error flag sent by the IUT, the LT sends a sequence of 5 dominant bits according to elementary test cases. After this sequence, the LT waits for (6 + 7) bit time before sending a dominant bit, corrupting the last bit of the error delimiter.
Response	The IUT shall generate an overload frame starting at the bit position following the last dominant bit sent by the LT.

7.5.6 Form error in passive error delimiter

Table 40 specifies the test of an error passive IUT that detects a form error when receiving an invalid error delimiter.

Table 40 — Form error in passive error delimiter

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT detects a form error when receiving an invalid error delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Error delimiter of passive error frame FDF = 0	Error delimiter of passive error frame FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the second bit of the error delimiter; #2 corrupting the fourth bit of the error delimiter; #3 corrupting the seventh bit of the error delimiter.	
Set-up	The IUT is set in passive state.	
Execution	The LT causes the IUT to generate a passive error frame in data field. During the error delimiter, the LT creates a form error according to elementary test cases. After the form error, the LT waits for (6 + 7) bit time before sending a dominant bit, corrupting the last bit of the error delimiter.	
Response	The IUT shall generate an overload frame starting at the bit position following the last dominant bit sent by the LT.	

7.5.7 Transition from active to passive ERROR FLAG

Table 41 specifies the test of an IUT that changes its state from active to passive.

Table 41 — Transition from active to passive ERROR FLAG

Item	Description	
Purpose	The purpose of this test is to verify that an IUT changes its state from active to passive.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Error at error frame FDF = 0	Error at error frame FDF = 1
Elementary test cases	There is one test to perform. #1 Bit error up to REC passive limit by sending 17 recessive bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error frame in data field. The LT corrupts the following active error flag according to elementary test cases. After this sequence, the IUT shall be error passive and sending a passive error flag. The LT send a valid frame 6 + 8 + 3 bit after dominant part of previous error sequence.	
Response	The IUT shall generate a passive error flag starting at the bit position following the last recessive bit sent by the LT. The IUT shall acknowledge the following test frame.	

7.6 Test class 6, error counter management

7.6.1 REC increment on bit error in active error flag

Table 42 specifies the test of an IUT that increments its REC by 8 when detecting a bit error during the transmission of an active error flag.

Table 42 — REC increment on bit error in active error flag

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 8 when detecting a bit error during the transmission of an active error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the active error flag; #2 corrupting the third bit of the active error flag; #3 corrupting the sixth bit of the active error flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error frame in data field. The LT corrupts one of the dominant bits of the error flag according to elementary test cases.	
Response	The IUT's REC value shall be increased by 8 on the corrupted bit.	

7.6.2 REC increment on bit error in overload flag

Table 43 specifies the test of an IUT that increments its REC by 8 when detecting a bit error during the transmission of an overload flag.

Table 43 — REC increment on bit error in overload flag

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 8 when detecting a bit error during the transmission of an overload flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the overload flag; #2 corrupting the fourth bit of the overload flag; #3 corrupting the sixth bit of the overload flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. The LT corrupts one of the dominant bits of the overload flag according to elementary test cases.	
Response	The IUT's REC value shall be increased by 8 on the corrupted bit.	

7.6.3 REC increment when active error flag is longer than 13 bits

Table 44 specifies the test of an IUT that increments its REC by 8 when detecting the eighth consecutive dominant bit following the transmission of its active error flag and after each sequence of additional 8 consecutive dominant bits.

Table 44 — REC increment when active error flag is longer than 13 bit

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 8 when detecting the eighth consecutive dominant bit following the transmission of its active error flag and after each sequence of additional 8 consecutive dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 16 bit dominant	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error frame in data field. After the error flag sent by the IUT, the LT sends a sequence of dominant bits according to elementary test cases.	
Response	The IUT's REC value shall be increased by 8 on each eighth dominant bit after the error flag.	

7.6.4 REC increment when overload flag is longer than 13 bits

Table 45 specifies the test of an IUT that increments its REC by 8 when detecting the eighth consecutive dominant bit following the transmission of its overload flag and after each sequence of additional 8 consecutive dominant bits.

Table 45 — REC increment when overload flag is longer than 13 bits

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 8 when detecting the eighth consecutive dominant bit following the transmission of its overload flag and after each sequence of additional 8 consecutive dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 16 bit dominant	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. After the overload flag sent by the IUT, the LT sends a sequence of dominant bits according to elementary test cases.	
Response	The IUT's REC value shall be increased by 8 on each eighth dominant bit after the overload flag.	

7.6.5 REC increment on bit error in the ACK field

Table 46 specifies the test of an IUT that increments its REC by 1 when detecting a bit error on the ACK slot it transmits.

Table 46 — REC increment on bit error in the ACK field

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a bit error on the ACK slot it transmits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 The ACK slot is corrupted to recessive value.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to send a dominant acknowledgement and apply an error according to elementary test cases.	
Response	The IUT's REC value shall be increased by 1 on the corrupted bit.	

7.6.6 REC increment on form error in CRC delimiter

Table 47 specifies the test of an IUT that increments its REC by 1 when detecting a form error at CRC delimiter.

Table 47 — REC increment on form error in CRC delimiter

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a form error at CRC delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 CRC delimiter changed to a dominant value.	
Set-up	The IUT is left in the default state.	
Execution	The LT sends a frame with the CRC delimiter modified according to elementary test cases.	
Response	The IUT's REC value shall be increased by 1 on the dominant CRC delimiter.	

7.6.7 REC increment on form error in ACK delimiter

Table 48 specifies the test of an IUT that increments its REC by 1 when detecting a form error on ACK delimiter.

Table 48 — REC increment on form error in ACK delimiter

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a form error on ACK delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Classical CAN frame (ACK = 0, ACK delimiter = 0)	There is one elementary test to perform. In FD format, 2 ACK bits are used to check the second possible ACK bit position (delayed ACK). #1 FD frame (ACK bit 1 = 0; ACK bit 2 = 0, ACK delimiter = 0)
Set-up	The IUT is left in the default state. The LT sends a frame with a stuff error in it and force 1 bit of error flag to recessive. This initializes the REC counter to 1 + 8 REC = 9.	
Execution	The LT sends a frame according to elementary test cases.	
Response	The IUT's REC value shall be increased by 1 on the dominant ACK delimiter. The REC value shall be decreased by 1 because the frame is error free until ACK. The REC value shall be unchanged as previous initialized while set up.	

7.6.8 REC increment on form error in EOF field

Table 49 specifies the test of an IUT that increments its REC by 1 when detecting a form error on the EOF field during reception of a data frame.

Table 49 — REC increment on form error in EOF field

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a form error on the EOF field during reception of a data frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the second bit of the EOF; #2 corrupting the third bit of the EOF; #3 corrupting the fifth bit of the EOF.	
Set-up	The IUT is left in the default state. The LT sends a frame with a stuff error in it and force 1 bit of error flag to recessive. This initializes the REC counter to 1 + 8 REC = 9.	
Execution	The LT sends a frame with the EOF modified according to elementary test cases.	
Response	The REC value shall be decreased by 1 because the frame is error free until ACK. The REC value shall be increased by 1 on the replaced bit of the EOF. The REC value shall be unchanged as previous initialized while set up.	

7.6.9 REC increment on stuff error

Table 50 specifies the test of an IUT that increments its REC by 1 when detecting a stuff error.

Table 50 — REC increment on stuff error

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a stuff error.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform on recessive stuff bits: #1 arbitration field; #2 control field; #3 data field; #4 CRC field. Elementary tests to perform on dominant stuff bits: #5 arbitration field; #6 control field; #7 data field; #8 CRC field.	Elementary tests to perform on recessive stuff bits: #1 arbitration field; #2 control field; #3 data field. Elementary tests to perform on dominant stuff bits: #4 arbitration field; #5 control field; #6 data field.

Table 50 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a sequence of 6 consecutive bits according to elementary test cases.
Response	The IUT's REC value shall be increased by 1 on the sixth consecutive bit.

7.6.10 REC increment on CRC error

Table 51 specifies the test of an IUT that increments its REC by 1 when detecting a CRC error during reception of a frame.

Table 51 — REC increment on CRC error

Item	Description	
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a CRC error during reception of a frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC ACK = 1 Bit recessive FDF = 0	REC DLC to cause different CRC types ACK = 2 Bit recessive FDF = 1
Elementary test cases	There is one elementary test to perform. #1 CRC (15) error	Elementary tests to perform: #1 DLC ≤ 10 – > CRC (17) error #2 DLC > 10 – > CRC (21) error
Set-up	The IUT is left in the default state.	
Execution	The LT sends a frame containing an error according to elementary test cases.	
Response	The IUT sends a recessive acknowledge. The IUT starts the transmission of an active error frame at the first bit position following the ACK delimiter. The IUT's REC value shall be increased by 1 by starting the error frame.	The IUT sends a recessive acknowledge. The IUT starts the transmission of an active error frame at the fourth bit position following the CRC delimiter. The IUT's REC value shall be increased by 1 by starting the error frame.

7.6.11 REC increment on dominant bit after end of error flag

Table 52 specifies the test of an IUT that increments its REC by 8 when detecting a dominant bit as the first bit after sending an error flag.

Table 52 — REC increment on dominant bit after end of error flag

Item	Description	
Purpose	This test verifies that an error active IUT increases its REC by 8 when detecting a dominant bit as the first bit after sending an error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Dominant bit at the bit position following the end of the error flag sent by the IUT.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error flag in data field. The LT sends a dominant bit according to elementary test cases.	
Response	The IUT's REC value shall be increased by 8 after reception of the dominant bit sent by the LT.	

7.6.12 REC increment on form error in error delimiter

Table 53 specifies the test of an IUT that increments its REC by 1 when detecting a form error on a bit of the error delimiter it is transmitting.

Table 53 — REC increment on form error in error delimiter

Item	Description	
Purpose	This test verifies that a receiver increases its REC by 1 when detecting a form error on a bit of the error delimiter it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform: #1 the second bit of the error delimiter is corrupted; #2 the seventh bit of the error delimiter is corrupted.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error frame in data field. The LT corrupts 1 bit of the error delimiter according to elementary test cases.	
Response	The IUT's REC value shall be increased by 1 after reception of the dominant bit sent by the LT.	

7.6.13 REC increment on form error in overload delimiter

Table 54 specifies the test of an IUT that increments its REC by 1 when detecting a form error on a bit of the overload delimiter it is transmitting.

Table 54 — REC increment on form error in overload delimiter

Item	Description	
Purpose	This test verifies that a receiver increases its REC by 1 when detecting a form error on a bit of the overload delimiter it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	Elementary tests to perform: #1 the second bit of the overload delimiter is corrupted; #2 the seventh bit of the overload delimiter is corrupted.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. The LT corrupts 1 bit of the overload delimiter according to elementary test cases.	
Response	The IUT's REC value shall be increased by 1 after reception of the dominant bit sent by the LT.	

7.6.14 REC decrement on valid frame reception

Table 55 specifies the test of an IUT that decrements its REC by 1 when receiving a valid frame.

Table 55 — REC decrement on valid frame reception

Item	Description	
Purpose	This test verifies that the IUT decreases its REC by 1 when receiving a valid frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 One valid test frame.	
Set-up	The IUT is left in the default state.	
Execution	The LT sends a frame with a stuff error in it and force 1 bit of error flag to recessive. The LT sends test frame according to elementary test cases.	
Response	The IUT's REC value shall be decreased by 1 after the successful transmission of the ACK slot.	

7.6.15 REC decrement on valid frame reception during passive state

Table 56 specifies the test of an IUT that decrement its REC to a value between 119 and 127 when receiving a valid frame while being error passive.

Table 56 — REC decrement on valid frame reception during passive state

Item	Description	
Purpose	This test verifies that the IUT sets its REC to a value between 119 and 127 when receiving a valid frame while being error passive.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 One valid test frame.	
Set-up	The LT causes the IUT's REC value to be at error passive level.	
Execution	The LT sends valid test frame according to elementary test cases.	
Response	The IUT's REC value shall be decremented to a value between 119 and 127 after the successful transmission of the ACK slot.	

7.6.16 REC non-increment on last bit of EOF field

Table 57 specifies the test of the IUT that does not change the value of its REC when detecting a dominant bit at the last bit of the EOF it is receiving.

Table 57 — REC non-increment on last bit of EOF field

Item	Description	
Purpose	This test verifies that the IUT does not change the value of its REC when detecting a dominant bit at the last bit of the EOF it is receiving.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Frame with a dominant bit at the last bit of EOF.	
Set-up	The IUT is left in the default state.	
Execution	The test system causes a receive error to initialize the REC value to 9. The LT sends one valid test frame according to elementary test cases.	
Response	The IUT's REC value shall be 8.	

7.6.17 REC non-increment on 13-bit length overload flag

Table 58 specifies the test of the IUT that does not change the value of its REC when receiving a 13-bit length overload flag.

Table 58 — REC non-increment on 13-bit length overload flag

Item	Description	
Purpose	This test verifies that the IUT does not change the value of its REC when receiving a 13-bit length overload flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 7 dominant bits.	
Set-up	The IUT is left in the default state.	
Execution	a) The test system causes a receive error to initialize the REC value to 9. b) The LT causes the IUT to generate an overload frame after a valid frame reception (REC-1). After the overload flag sent by the IUT, the LT sends a sequence according to elementary test cases.	
Response	The correct frame up to the EOF will decrement REC and the overload enlargement will not increase REC. The IUT's REC value shall be 8.	

7.6.18 REC non-increment on 13-bit length error flag

Table 59 specifies the test of the IUT that does not increase its REC after the seventh bit of the received error flag.

Table 59 — REC non-increment on 13-bit length error flag

Item	Description	
Purpose	This test verifies that the IUT does not increase its REC after the seventh bit of the received error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 7 dominant bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to generate an active error frame in data field. After the error flag sent by the IUT, the LT sends a sequence according to elementary test cases.	
Response	The IUT's REC value shall be not further incremented after the increment due to the dominant bit which followed the error flag sent by the IUT.	

7.6.19 REC non-increment on last bit of error delimiter

Table 60 specifies the test of the IUT that does not change the value of its REC when detecting a dominant bit at the last bit of an error delimiter it is transmitting.

Table 60 — REC non-increment on last bit of error delimiter

Item	Description	
Purpose	This test verifies that the IUT does not change the value of its REC when detecting a dominant bit at the last bit of an error delimiter it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 It corrupts the last bit of the error delimiter.	
Set-up	No action required, the IUT is left in the default state.	
Execution	The LT causes the IUT to generate an error frame in data field. The LT applied an error according to elementary test cases.	
Response	The IUT's REC value shall be one.	

7.6.20 REC non-increment on last bit of overload delimiter

Table 61 specifies the test of the IUT that does not change the value of its REC when detecting a dominant bit at the last bit of an overload delimiter it is transmitting.

Table 61 — REC non-increment on last bit of overload delimiter

Item	Description	
Purpose	This test verifies that the IUT does not change the value of its REC when detecting a dominant bit at the last bit of an overload delimiter it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 It corrupts the last bit of the overload delimiter.	
Set-up	No action required, the IUT is left in the default state.	
Execution	The LT causes the IUT to generate an overload frame after a data frame. The LT applies an error according to elementary test cases.	
Response	The IUT's REC value shall be zero.	

7.6.21 REC non-decrement on transmission

Table 62 specifies the test of the IUT that does not change the value of its REC when transmitting a frame successfully.

Table 62 — REC non-decrement on transmission

Item	Description	
Purpose	This test verifies that the IUT does not change the value of its REC when transmitting a frame successfully.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	REC FDF = 0	REC FDF = 1
Elementary test cases	There is one elementary test to perform. #1 The higher prior frame is disturbed by an error to increase REC.	
Set-up	No action required, the IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. The LT sends a frame with higher ID priority to cause the IUT to lose arbitration according to elementary test cases. The IUT will repeat its transmission after error treatment.	
Response	The IUT's REC value shall be incremented and not decremented after transmission.	

7.6.22 REC increment on form error at fixed stuff bit in FD frames

Table 63 specifies the test of an IUT that increments its REC by 1 when detecting a form error.

Table 63 — REC increment on form error at fixed stuff bit in FD frames

Item	Description
Purpose	This test verifies that the IUT increases its REC by 1 when detecting a form error.
CAN_VERSION	CAN FD enabled
Test variables	REC DLC - to cause different CRC types FDF = 1
Elementary test cases	Elementary tests to perform on recessive stuff bits: #1 DLC ≤ 10 – > CRC (17) field; #2 DLC > 10 – > CRC (21) field. Elementary tests to perform on dominant stuff bits: #3 DLC ≤ 10 – > CRC (17) field #4 DLC > 10 – > CRC (21) field
Set-up	The IUT is left in the default state.
Execution	The LT corrupts a fixed stuff bit according to elementary test cases.
Response	The IUT's REC value shall be increased by 1 on the corrupted fixed stuff bit.

7.6.23 REC non-increment on protocol exception in FD frames

Table 64 specifies the test of an IUT that switches to protocol exception on non-nominal values and did not change the REC.

Table 64 — REC non-increment on protocol exception in FD frames

Item	Description									
Purpose	The purpose of this test is to verify that the IUT switches to protocol exception on non-nominal values of the bits described in test variables and did not change the CAN error counter.									
CAN_VERSION	CAN FD tolerant					CAN FD enabled				
Test variables	FDF = 1 DLC Data: All data byte with the same value Bit rate ratio between nominal and data bit rate					FDF = 1 "res" bit = 1 DLC Data: All data byte with the same value Bit rate ratio between nominal and data bit rate				
Elementary test cases	**Test**	**Format**	**DLC**	**data**	**bit rate ratio**	**Test**	**Format**	**DLC**	**data**	**bit rate ratio**
	#1	FBFF	A_h;	AA_h;	1:2	#1	FBFF	A_h;	AA_h;	1:2
	#2	FBFF	F_h;	FF_h;	1:8	#2	FBFF	F_h;	FF_h;	1:8
	#3	CBFF[a]	F_h;	FF_h;	—	Protocol exception handling shall be enabled.				
Set-up	The IUT is left in the default state.									
Execution	a) The test system causes a receive error to initialize the REC value to 9. b) A single test frame is used for the elementary test, followed immediately by a valid Classical CAN frame.									
Response	The IUT shall not generate any error flag in this test frame. The IUT shall not acknowledge the test frame. A following data frame in classical frame format received by the IUT during the test state shall match the data sent in the test frame. The IUT's REC value shall be 8 after reception of the valid Classical CAN frame.									

[a] Valid classical frame with non-nominal FDF bit.

7.7 Test class 7, bit timing Classical CAN frame format

7.7.1 Sample point test

Table 65 specifies the test of the correct position of the sample point of an IUT.

Table 65 — Sample point test

Item	Description
Purpose	The purpose of this test is to verify the position of the sample point of an IUT.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform. Refer to 6.2.3.

Table 65 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT shortens a dominant stuff bit in arbitration field by an amount of Phase_Seg2(N) and then later shortens another dominant stuff bit by an amount of [Phase_Seg2(N) + 1] according to elementary test cases.
Response	The IUT shall generate an error frame on the bit position following the second shortened stuff bit.

7.7.2 Hard synchronization on SOF reception

Table 66 specifies the test of the IUT that makes a hard synchronization when receiving an early SOF delayed by ***e***, ***e*** ∈ [1, NTQ(N)].

Table 66 — Hard synchronization on SOF reception

Item	Description
Purpose	The purpose of this test is to verify that the IUT makes a hard synchronization when receiving an early SOF delayed by *e*, *e* ∈ [1, NTQ(N)].
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform for each possible value of ***e***. #1 Length of third bit of intermission field is ***e*** ∈ [1, NTQ(N)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a first test frame disturbed by an error frame and after the second bit of the intermission field, it sends an SOF delayed by ***e*** time quanta depending on the elementary test cases. The SOF is followed by a sequence of 5 dominant bits.
Response	The IUT shall respond with an error frame 6 bit times –1TQ(N) (Sync_Segment) or up to 6 bit times after the recessive to dominant edge at the beginning of the SOF.

7.7.3 Synchronization when *e* > 0 and *e* ≤ SJW(N)

Table 67 specifies the test of the behaviour of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(N).

Table 67 — Synchronization when *e* > 0 and *e* ≤ SJW

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(N).
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. FDF = 0

Table 67 *(continued)*

Item	Description
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with ***e*** ∈ [1, SJW(N)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT delays a dominant stuff bit in arbitration field by an amount of ***e*** time quanta and shortens the same bit by an amount of [Phase_Seg2(N) + 1TQ] according to elementary test cases.
Response	The IUT shall generate an error frame 1 bit time after the recessive to dominant edge of the delayed stuff bit.

7.7.4 Synchronization when *e* > 0 and *e* > SJW(N)

Table 68 specifies the test of the correct behaviour of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(N).

Table 68 — Synchronization when *e* > 0 and *e* > SJW

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(N).
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $e \in [SJW(N) + 1], [NTQ(N) - Phase\_Seg2(N) - 1]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT delays a dominant stuff bit in arbitration field by an amount of ***e*** time quanta and shortens the same bit by an amount of [Phase_Seg2(N) + 1TQ + ***e*** − SJW(N)] according to elementary test cases.
Response	The IUT shall generate an error frame 1 bit time − [***e*** − SJW(N)] time quanta after the recessive to dominant edge of the delayed stuff bit.

7.7.5 Synchronization when *e* < 0 and |*e*| ≤ SJW(N)

Table 69 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with |***e***| ≤ SJW(N).

Table 69 — Synchronization when $e < 0$ and $|e| \leq$ SJW

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW(N).
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with $|e| \in [1, \text{SJW(N)}]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT shortens the last recessive bit before an expected dominant stuff bit in arbitration field by an amount of $|e|$ time quanta and then sends a dominant value for one time quantum followed by a recessive state according to elementary test cases.
Response	The IUT shall generate an error frame 1 bit time after the last recessive to dominant edge.

7.7.6 Synchronization when e < 0 and |e| > SJW(N)

Table 70 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| >$ SJW(N).

Table 70 — Synchronization when e < 0 and |e| > SJW

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| >$ SJW(N).
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta, where: $|e| \in \{[(\text{SJW(N)} + 1], \text{Phase\_Seg2(N)}\}$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT shortens the last recessive bit before an expected dominant stuff bit in arbitration field by an amount of $|e|$ time quanta and then sends a dominant value for one time quantum followed by a recessive state according to elementary test cases.
Response	The IUT shall generate an error frame 1 bit time + $[|e| - \text{SJW(N)}]$ time quanta after the last recessive to dominant edge.

7.7.7 Glitch filtering test on positive phase error

Table 71 specifies the test to verify that there is only one synchronization within 1 bit time if there are two recessive to dominant edges between synchronization segment and sample point.

Table 71 — Glitch filtering test on positive phase error

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there are two recessive to dominant edges between synchronization segment and sample point. The test also verifies that an IUT is able to synchronize on a minimum duration pulse obeying to the synchronization rules.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Glitch pulse length = 1 TQ(N) FDF = 0
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 Recessive glitch at third TQ(N). Refer to 6.2.3.
Set-up	No action required, the IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in arbitration field. After the first two time quanta of dominant value, it changes one time quantum to recessive value according to elementary test cases. This dominant stuff bit is followed by 6 recessive bits.
Response	The IUT shall respond with an error frame exactly 7 bit times after the first recessive to dominant edge of the stuff bit.

7.7.8 Glitch filtering test on negative phase error

Table 72 specifies the test to verify that there is only one synchronization within 1 bit time if there are two recessive to dominant edges between two sample points where the first edge comes before the synchronization segment.

Table 72 — Glitch filtering test on negative phase error

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there are two recessive to dominant edges between two sample points where the first edge comes before the synchronization segment. The test also verifies that an IUT is able to synchronize on a minimum duration pulse obeying to the synchronization rules.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Glitch pulse length = 1 TQ(N) FDF = 0
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 Recessive glitch at third TQ(N) Refer to 6.2.3.

Table 72 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in arbitration field. The recessive bit before the stuff bit is shortened by one time quantum. After the first two time quanta of dominant value, it changes one time quantum to recessive value according to elementary test cases. This dominant stuff bit is followed by 6 recessive bits.
Response	The IUT shall respond with an error frame exactly 7 bit times after the first recessive to dominant edge of the stuff bit.

7.7.9 Glitch filtering test in idle state

7.7.9.1 Glitch filtering test in idle state 1 (single pulse)

Table 73 specifies the test of an IUT that will not detect an SOF when detected dominant level ≤ (Prop_Seg+ Phase_Seg1 − 1 TQ).

Table 73 — Glitch filtering test in idle state 1 (single pulse)

Item	Description
Purpose	The purpose of this test is to verify that an IUT will not detect an SOF when: detected dominant level ≤ [Prop_Seg(N) + Phase_Seg1(N) − 1 TQ(N)].
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. Glitch Pulse length = Prop_Seg(N) + Phase_Seg1(N) − 1 TQ(N) FDF = 0
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 Dominant pulse on IDLE bus [Prop_Seg(N) + Phase_Seg1(N) − 1 TQ(N)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a dominant glitch according to elementary test cases for this test case. Then the LT waits for 8 bit times.
Response	The IUT shall remain in the idle state.

7.7.9.2 Glitch filtering test in idle state 2 (multiple pulses)

Table 74 specifies the test of an IUT that will not use any edge for resynchronization after detection of a recessive to dominant edge in idle state (after hard synchronization).

Table 74 — Glitch filtering test in idle state 2 (multiple pulses)

Item	Description
Purpose	The purpose of this test is to verify that an IUT will not use any edge for resynchronization after detection of a recessive to dominant edge in idle state (after hard synchronization).
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. Dominant pulses on IDLE bus. Pulse group: a) First glitch = (Prop_Seg(N) + Phase_Seg1(N) − 2)/2 b) Recessive time = 2 TQ(N) c) Second glitch = {[Prop_Seg(N) + Phase_Seg1(N) - 2]/2} − 1 minimum time quantum d) Recessive time = 1 TQ(N) + 2 minimum time quantum e) Third glitch = Prop_Seg(N) + Phase_Seg1(N) − 2 FDF = 0
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 Three dominant glitches separated by recessive TQ(N) times. The first glitch activates the edge detection of IUT. The next two glitches cover the TQ(N) position of the configured Sampling_Point(N) regarding to the first glitch. Refer to 6.2.3.
Set-up	No action required, the IUT is left in the default state.
Execution	The LT sends a dominant glitch group according to elementary test cases for this test case. Then, the LT waits for 8 bit times to check that no error frame will start after that.
Response	The IUT shall remain in the idle state.

7.7.10 Non-Synchronization after a dominant sampled bit

Table 75 specifies the test to verify that no edge shall be used for resynchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge.

Table 75 — Non-Synchronization after a dominant sampled bit

Item	Description
Purpose	The purpose of this test is to verify that no edge shall be used for resynchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Glitch between 2 dominant sampled bits FDF = 0
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 One TQ recessive glitch in Phase_Seg2(N). Refer to 6.2.3.

Table 75 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in arbitration field. At the position [NTQ(N) - Phase_Seg2(N) + 1] time quanta after the falling edge at the beginning of the stuff bit, the LT changes the value to recessive for one time quantum according to elementary test cases. The stuff bit is followed by 5 additional dominant bits.
Response	The IUT shall respond with an error frame exactly 6 bit times after the recessive to dominant edge at the beginning of the stuff bit.

7.7.11 Synchronization when $e < 0$ and $|e| \leq$ SJW(N) at "ACK" bit position

Table 76 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW(N) on bit position ACK.

Table 76 — Synchronization when $e < 0$ and $|e| \leq$ SJW(N) at "ACK" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW(N) on bit position ACK.
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. FDF = 0
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with $|e| \in [1, \text{SJW(N)}]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame. The LT forces an amount of ***e*** TQ from end of CRC delimiter bit to dominant. Additionally, the ACK bit shall be forced to recessive from end of bit toward Sampling_Point(N) for Phase_Seg2(N) + ***e*** according to elementary test cases. The bit shall be sampled as dominant.
Response	The frame is valid, no error flag shall occur.

7.8 Test class 8, bit timing CAN FD frame format

7.8.1 Sample point test

7.8.1.1 Sample point test at "BRS" bit position

Table 77 specifies the test of the correct position of the sample point of an IUT on bit position BRS.

Table 77 — Sample point test at "BRS" bit position

Item	Description
Purpose	The purpose of this test is to verify the position of the sample point of an IUT on bit position BRS.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. BRS FDF = 1
Elementary test cases	There are two elementary tests to perform for at least 1 bit rate configuration: #1 test for early sampling point: bit level change to recessive before sampling point; #2 test for late sampling point: bit level change to recessive after sampling point. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Test BRS #1: The LT forces the BRS bit to dominant from beginning up to one TQ(N) before Sampling_Point(N). Test BRS #2: The LT forces the BRS bit to dominant from beginning up to Sampling_Point(N).
Response	Test BRS #1: The modified BRS bit shall be sampled as recessive. The frame is valid. No error flag shall occur. Test BRS #2: The modified BRS bit shall be sampled as dominant. The frame is valid. No error flag shall occur. The bit rate will not switch for data phase.

7.8.1.2 Sample point test at "DATA" field

Table 78 specifies the test of the correct position of the sample point of an IUT on bit position DATA.

Table 78 — Sample point test at "DATA" field

Item	Description
Purpose	The purpose of this test is to verify the position of the sample point of an IUT on bit position DATA field.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) configuration as available by IUT. DATA field FDF = 1
Elementary test cases	There are two elementary tests to perform for at least 1 bit rate configuration: #1 test for early sampling point: bit level change to recessive before sampling point; #2 test for late sampling point: bit level change to recessive after sampling point. Refer to 6.2.3.

Table 78 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Test DATA #1: The LT forces a recessive bit to dominant from beginning up to one TQ(D) before the sampling point. Test DATA #2: The LT forces a dominant bit to recessive for Phase_Seg2(D).
Response	Test DATA #1: The modified data bit shall be sampled as recessive. The frame is valid. No error flag shall occur. Test DATA #2: The modified data bit shall be sampled as dominant. The frame is valid. No error flag shall occur.

7.8.1.3 Sample point test at "CRC delimiter" bit position

Table 79 specifies the test of the correct position of the sample point of an IUT on bit position CRC delimiter.

Table 79 — Sample point test at "CRC delimiter" bit position

Item	Description
Purpose	The purpose of this test is to verify the position of the sample point of an IUT on bit position CRC delimiter.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) configuration as available by IUT. CRC delimiter EDF = 1
Elementary test cases	There are two elementary tests to perform for at least 1 bit rate configuration: #1 test for early sampling point: bit level change to recessive before sampling point; #2 test for late sampling point: bit level change to recessive after sampling point. Refer to 6.2.3.

Table 79 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Test CRC delimiter #1: The LT forces a recessive CRC delimiter bit to dominant from beginning up to one TQ(D) before the Sampling point. Test CRC delimiter #2: The LT forces a recessive CRC delimiter bit to dominant from beginning up to the sampling point.
Response	Test CRC delimiter #1: The modified CRC delimiter bit shall be sampled as recessive. The frame is valid. No error flag shall occur. Test CRC delimiter #2: The modified CRC delimiter bit shall be sampled as dominant. The frame is invalid. An error frame shall follow.

7.8.2 Hard synchronization on “res” bit

7.8.2.1 Hard synchronization on “res” bit (delayed end of FDF)

Table 80 specifies the test of an IUT that makes a hard synchronization when receiving a recessive to dominant edge delayed by ***e***, where ***e*** ∈ [SJW(N) + 1, NTQ(N) − Phase_Seg2(N) − 1].

Table 80 — Hard synchronization on “res” bit reception (delayed end of FDF)

Item	Description
Purpose	The purpose of this test is to verify that the IUT makes a hard synchronization when receiving a recessive to dominant edge delayed by ***e***, where: ***e*** ∈ [SJW(N) + 1, NTQ(N) − Phase_Seg2(N) − 1]
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. “res” bit FDF = 1 BRS = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The LT generates a valid frame with prolonged FDF bit by an amount of ***e*** ∈ [SJW(N) + 1, NTQ(N) − Phase_Seg2(N) − 1]. Refer to 6.2.3.

Table 80 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. The LT sets the first [Prop_Seg(N) + Phase_Seg1(N)] TQ's of the recessive BRS bit to dominant.
Response	The modified BRS bit shall be sampled as recessive. The hard synchronization shall correct the maximum phase error as defined in ISO 11898-1. The frame is valid. No error flag shall occur.

7.8.2.2 Hard synchronization on "res" bit (early end of FDF)

Table 81 specifies the test of an IUT that makes a hard synchronization when receiving a recessive to dominant edge ahead by a negative phase error ***e***, where ***e*** ∈ {[SJW(N)+1], Phase_Seg2(N)}.

Table 81 — Hard synchronization on res bit reception (early end of FDF)

Item	Description
Purpose	The purpose of this test is to verify that the IUT makes a hard synchronization when receiving an early recessive to dominant edge between FDF and "res" bit by ***e***, where: ***e*** = Phase_Seg2(N)
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) configuration as available by IUT. SJW(N) = 1 res FDF = 1 BRS = 0 ESI = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The LT generates a valid frame with shortened FDF bit by an amount of ***e*** = Phase_Seg2(N). Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. The LT sets the last Phase_Seg2(D) TQ of the dominant BRS bit to recessive.
Response	The modified BRS bit shall be sampled as dominant. The hard synchronization shall correct the maximum phase error as defined in ISO 11898-1. The frame is valid. No error flag shall occur. The bit rate will not switch for data phase.

7.8.3 Synchronization when *e* > 0 and *e* ≤ SJW(D)

7.8.3.1 Synchronization when *e* > 0 and *e* ≤ SJW(D) at ESI bit position

Table 82 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position ESI.

Table 82 — Synchronization when *e* > 0 and *e* ≤ SJW(D) at ESI bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position ESI.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. ESI = 1 FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with ***e*** ∈ [1, SJW(D)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The first ***e*** TQ(D) (***e*** according to elementary test cases) of the ESI bit are set to recessive, then the following {Prop_Seg(D) + Phase_Seg1(D)] TQ(D)'s are set to dominant. The rest of the ESI bit, Phase_Seg2(D)+1 is set to recessive. At all, ESI is lengthened by e TQ(D).
Response	The modified ESI bit shall be sampled as recessive. The frame is valid. No error flag shall occur.

7.8.3.2 Synchronization when *e* > 0 and *e* ≤ SJW(D) at DATA field

Table 83 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position DATA.

Table 83 — Synchronization when *e* > 0 and *e* ≤ SJW(D) at DATA field

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with ***e*** ∈ [1, SJW(D)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in DATA field. Then, the recessive to dominant edge before this dominant stuff bit shall be delayed by additional ***e*** TQ(D)'s of recessive value at the beginning of this stuff bit according to elementary test cases. The LT forces a part of Phase_Seg2(D) of the delayed stuff bit to recessive. This recessive part of Phase_seg2 start at ***e*** − 1 TQ(D) after sampling point.
Response	The modified data bit shall be sampled as recessive. The wrong value of stuff bit shall cause an error frame.

7.8.3.3 Synchronization when *e* > 0 and *e* ≤ SJW(D) at CRC delimiter bit position

Table 84 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position CRC delimiter.

Table 84 — Synchronization when *e* > 0 and *e* ≤ SJW(D) at CRC delimiter bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** ≤ SJW(D) on bit position CRC delimiter
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. CRC: LSB = 1 CRC delimiter FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with ***e*** ∈ [1, SJW(D)]. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a test frame with a recessive bit value at last bit of CRC. The LT forces the CRC delimiter to dominant bit value. Then, the recessive to dominant edge between LSB of CRC and CRC delimiter shall be delayed by additional ***e*** TQ(D)'s of recessive value at the beginning of CRC delimiter bit according to elementary test cases. The LT forces a part of Phase_Seg2(D) of the delayed CRC delimiter bit to recessive. This recessive part of Phase_seg2 start at ***e*** − 1 TQ(D) after sampling point.
Response	The modified CRC delimiter bit shall be sampled as recessive. The frame is valid, no error flag shall occur.

7.8.4 Synchronization when *e* > 0 and *e* > SJW(D)

7.8.4.1 Synchronization when *e* > 0 and *e* > SJW(D) at ESI bit position

Table 85 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position ESI.

Table 85 — Synchronization when *e* > 0 and *e* > SJW(D) at ESI bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position ESI.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. ESI = 1 FDF = 1

Table 85 *(continued)*

Item	Description
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $e \in \{[SJW(D) + 1], [NTQ(D) - Phase\_Seg2(D) - 1]\{$ Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame with recessive ESI bit. The LT invert the value of ESI bit to dominant value. Then, the recessive to dominant edge between BRS and ESI shall be delayed by additional ***e*** TQ(D)'s of recessive value at the beginning of ESI bit according to elementary test cases. The LT forces a part of Phase_Seg2(D) of the delayed ESI bit to recessive. This recessive part of Phase_seg2 start at SJW(D) − 1 TQ(D) after sampling point.
Response	The modified ESI bit shall be sampled as recessive. The frame is valid, no error flag shall occur.

7.8.4.2 Synchronization when *e* > 0 and *e* > SJW(D) at DATA field

Table 86 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position DATA.

Table 86 — Synchronization when *e* > 0 and *e* > SJW(D) at DATA field

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $e \in \{[SJW(D) + 1], [NTQ(D) - Phase\_Seg2(D) - 1]\}$ Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in DATA field. Then, the recessive to dominant edge before this dominant stuff bit shall be delayed by additional ***e*** TQ(D)'s of recessive value at the beginning of this stuff bit according to elementary test cases. The LT forces a part of Phase_Seg2(D) of the delayed stuff bit to recessive. This recessive part of Phase_seg2 start at SJW(D) − 1 TQ(D) after sampling point.
Response	The modified data bit shall be sampled as recessive. The wrong value of stuff bit shall cause an error flag.

7.8.4.3 Synchronization when $e > 0$ and $e >$ SJW(D) at "CRC delimiter" bit position

Table 87 specifies the test of an IUT that detects a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position CRC delimiter.

Table 87 — Synchronization when $e > 0$ and $e >$ SJW(D) at "CRC delimiter" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a positive phase error ***e*** on a recessive to dominant edge with ***e*** > SJW(D) on bit position CRC delimiter.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. CRC: LSB = 1 CRC delimiter FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $e \in \{[SJW(D) + 1], [NTQ(D) - Phase\_Seg2(D) - 1]\}$ Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a test frame with a recessive bit value at last bit of CRC. The LT forces the CRC delimiter to dominant bit value. Then, the recessive to dominant edge between LSB of CRC and CRC delimiter shall be delayed by additional ***e*** TQ(D)'s of recessive value at the beginning of CRC delimiter bit according to elementary test cases. The LT forces a part of Phase_Seg2(D) of the delayed CRC delimiter bit to recessive. This recessive part of Phase_seg2 start at SJW(D) − 1 TQ(D) after sampling point.
Response	The modified CRC delimiter bit shall be sampled as recessive. The frame is valid, no error flag shall occur.

7.8.5 Synchronization when $e < 0$ and $|e| \leq$ SJW

7.8.5.1 Synchronization when $e < 0$ and $|e| \leq$ SJW(D) at "ESI" bit position

Table 88 specifies the test of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW(D) on bit position ESI.

Table 88 — Synchronization when $e < 0$ and $|e| \leq$ SJW(D) at "ESI" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW(D) on bit position ESI.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. ESI = 0 FDF = 1

Table 88 *(continued)*

Item	Description
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with $\|e\| \in [1, SJW(D)]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame with dominant ESI bit. The LT shortened the BRS bit by an amount of $\|e\|$ TQ according to elementary test cases. Additionally, the ESI bit shall be forced to recessive value from [Sync_Seg(D) + Prop_Seg(D) + Phase_Seg1(D) − ***e***] up to end of bit.
Response	The modified ESI bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.5.2 Synchronization when $e < 0$ and $|e| \leq SJW(D)$ at "DATA" field

Table 89 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq SJW(D)$ on bit position DATA.

Table 89 — Synchronization when $e < 0$ and $|e| \leq SJW(D)$ at "DATA" field

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $\|e\| \leq SJW(D)$ on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with $\|e\| \in [1, SJW(D)]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in DATA field. The LT shortened the DATA bit before the dominant stuff bit by an amount of $\|e\|$ TQ according to elementary test cases. Additionally, the Phase_Seg2(D) of the dominant stuff bit shall be forced to recessive.
Response	The modified stuff bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.5.3 Synchronization when $e < 0$ and $|e| \leq SJW(N)$ at "ACK" bit position

Table 90 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq SJW(N)$ on bit position ACK.

Table 90 — Synchronization when $e < 0$ and $|e| \leq$ SJW(N) at "ACK" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| \leq$ SJW on bit position ACK.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Phase error ***e*** ACK FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta with $|e| \in [1, \text{SJW(N)}]$. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame. The LT shortened the CRC delimiter by an amount of $|e|$ TQ according to elementary test cases. Additionally, the Phase_Seg2(N) of this dominant ACK bit shall be forced to recessive.
Response	The modified ACK bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.6 Synchronization when $e < 0$ and $|e| >$ SJW

7.8.6.1 Synchronization when $e < 0$ and $|e| >$ SJW(D) at "ESI" bit position

Table 91 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| >$ SJW(D) on bit position ESI.

Table 91 — Synchronization when $e < 0$ and $|e| >$ SJW at "ESI" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| >$ SJW(D) on bit position ESI.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Phase error ***e*** ESI = 0 FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $|e| \in \{[\text{SJW(D)} + 1], \text{Phase\_Seg2(D)}\}$ Refer to 6.2.3.

Table 91 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame. The LT forces an amount of $\|e\|$ TQ from end of Phase_Seg2(D) of BRS bit to dominant according to elementary test cases. By this, the BRS bit of the IUT is shortened by an amount of SJW(D). Additionally, the Phase_Seg2(D) of ESI bit shall be forced to recessive.
Response	The modified ESI bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.6.2 Synchronization when $e < 0$ and $|e| > SJW(D)$ at "DATA" field

Table 92 specifies the test of the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| > SJW(D)$ on bit position DATA.

Table 92 — Synchronization when $e < 0$ and $|e| > SJW(D)$ at "DATA" field

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $\|e\| > SJW(D)$ on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Phase error ***e*** DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $\|e\| \in \{[SJW(D) + 1], Phase\_Seg2(D)\}$ Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a dominant stuff bit in DATA field. The LT forces an amount of $\|e\|$ TQ from end of Phase_Seg2(D) of the DATA bit before the dominant stuff bit to dominant according to elementary test cases. By this, the DATA bit of the IUT is shortened by an amount of SJW(D). Additionally, the Phase_Seg2(D) of the dominant stuff bit shall be forced to recessive.
Response	The modified stuff bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.6.3 Synchronization when $e < 0$ and $|e| > SJW(N)$ at "ACK" bit position

Table 93 specifies the test if the correct behaviour of an IUT that detects a negative phase error ***e*** on a recessive to dominant edge with $|e| > SJW(N)$ on bit position ACK.

Table 93 — Synchronization when $e < 0$ and $|e| > SJW(N)$ at "ACK" bit position

Item	Description
Purpose	The purpose of this test is to verify the behaviour of an IUT detecting a negative phase error ***e*** on a recessive to dominant edge with $|e| > SJW$ on bit position ACK.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Phase error ***e*** ACK FDF = 1
Elementary test cases	There is one elementary test to perform for each possible value of ***e*** for at least 1 bit rate configuration. #1 The values tested for ***e*** are measured in time quanta where: $|e| \in \{[SJW(N) + 1], Phase\_Seg2(N)\}$ Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame. The LT forces an amount of $|e|$ TQ from end of Phase_Seg2(N) of CRC delimiter bit to dominant according to elementary test cases. By this, the CRC delimiter bit of the IUT is shortened by an amount of SJW(N). Additionally, the Phase_Seg2(N) of ACK bit shall be forced to recessive.
Response	The modified ACK bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.7 Glitch filtering test on positive phase error

7.8.7.1 Glitch filtering test on positive phase error at "res" bit position

Table 94 specifies the test to verify that there is only one synchronization within 1 bit time if there are additional recessive to dominant edges between synchronization segment and sample point on bit position "res" bit.

Table 94 — Glitch filtering test on positive phase error at "res" bit position

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there are additional recessive to dominant edges between synchronization segment and sample point on bit position "res" bit.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Glitch between synchronization segment and sample point. FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the second TQ of "res" bit to recessive. Refer to 6.2.3.

Table 94 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(N) of "res" bit shall be forced to recessive.
Response	The modified "res" bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.7.2 Glitch filtering test on positive phase error at "DATA" field

Table 95 specifies the test to verify that there is only one synchronization within 1 bit time if there are additional recessive to dominant edges between synchronization segment and sample point on bit position DATA.

Table 95 — Glitch filtering test on positive phase error at "DATA" field

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there are additional recessive to dominant edges between synchronization segment and sample point on bit position DATA. The test also verifies that an IUT is able to synchronize on a minimum duration pulse obeying to the synchronization rules.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Glitch between synchronization segment and sample point. FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the second TQ of a dominant stuff bit to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(D) of this dominant stuff bit shall be forced to recessive.
Response	The modified stuff bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.7.3 Glitch filtering test on positive phase error at "ACK" bit position

Table 96 specifies the test to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between synchronization segment and sample point on bit position ACK.

Table 96 — Glitch filtering test on positive phase error at "ACK" bit position

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between synchronization segment and sample point on bit position ACK.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Glitch between synchronization segment and sample point. ACK FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the second TQ of ACK bit to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(N) of this ACK bit shall be forced to recessive.
Response	The modified ACK bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.8 Glitch filtering test on negative phase error

7.8.8.1 Glitch filtering test on negative phase error at "ESI" bit position

Table 97 specifies the test to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position ESI.

Table 97 — Glitch filtering test on negative phase error at "ESI" bit position

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position ESI.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Bit start with negative offset and glitch between synchronization segment and sample point. ESI = 0 FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT reduce the length of BRS bit by one TQ(D) and the LT force the second TQ of ESI to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(D) of ESI bit shall be forced to recessive.
Response	The modified ESI bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.8.2 Glitch filtering test on negative phase error at “DATA” field

Table 98 specifies the test to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position DATA.

Table 98 — Glitch filtering test on negative phase error at “DATA” field

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Bit start with negative offset and glitch between synchronization segment and sample point. DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT reduce the length of a DATA bit by one TQ(D) and the LT force the second TQ of this dominant stuff bit to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(D) of this dominant stuff bit shall be forced to recessive. The bit shall be sampled as dominant.
Response	The modified stuff bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.8.3 Glitch filtering test on negative phase error at “ACK” bit position

Table 99 specifies the test to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position ACK.

Table 99 — Glitch filtering test on negative phase error at “ACK” bit position

Item	Description
Purpose	The purpose of this test is to verify that there is only one synchronization within 1 bit time if there is an additional recessive to dominant edge between two sample points where the first edge comes before the synchronization segment on bit position ACK. The test also verifies that an IUT is able to synchronize on a minimum duration pulse obeying to the synchronization rules.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Bit start with negative offset and glitch between synchronization segment and sample point. ACK FDF = 1

Table 99 *(continued)*

Item	Description
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT reduce the length of CRC delimiter bit by one TQ(D) and the LT force the second TQ of ACK bit to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame according to elementary test cases. Additionally, the Phase_Seg2(N) of this dominant ACK bit shall be forced to recessive.
Response	The modified ACK bit shall be sampled as dominant. The frame is valid, no error flag shall occur.

7.8.9 No synchronization after a dominant sampled bit

7.8.9.1 No synchronization after a dominant sampled bit at "BRS" bit position

Table 100 specifies the test to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position BRS.

Table 100 — No synchronization after a dominant sampled bit at "BRS" bit position

Item	Description
Purpose	The purpose of this test is to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position BRS.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(N) and SJW(N) configuration as available by IUT. Recessive to dominant edge between 2 dominant bits. BRS = 0 FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the first two TQ(N) and the complete Phase_Seg2(N) of BRS bit to recessive. Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame with dominant BRS bit. The LT inverts parts of BRS bit according to elementary test cases.
Response	The modified BRS bit shall be sampled as dominant. The frame is valid. No error flag shall occur. The bit rate will not switch for data phase.

7.8.9.2 No synchronization after a dominant sampled bit at "DATA" field

Table 101 specifies the test to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position DATA.

Table 101 — No synchronization after a dominant sampled bit at "DATA" field

Item	Description
Purpose	The purpose of this test is to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position DATA.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Recessive to dominant edge between 2 dominant bits. DATA field FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the stuff bit to dominant from the second TQ(D) until the beginning of Phase_Seg2(D). Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT sends a frame containing a recessive stuff bit in data phase. The LT forces a recessive stuff bit inside DATA field to dominant according to elementary test cases.
Response	The modified stuff bit shall be sampled as dominant. The dominant sampled stuff bit shall be detected as a stuff error and shall be followed by an error frame.

7.8.9.3 No synchronization after a dominant sampled bit at "CRC delimiter" bit position

Table 102 specifies the test to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position CRC delimiter.

Table 102 — No synchronization after a dominant sampled bit at "CRC delimiter" bit position

Item	Description
Purpose	The purpose of this test is to verify that no edge shall be used for synchronization if the value detected at the previous sample point is the same as the bus value immediately after the edge on bit position CRC delimiter.
CAN_VERSION	CAN FD enabled
Test variables	Sampling_Point(D) and SJW(D) configuration as available by IUT. Recessive to dominant edge between 2 dominant bits. CRC delimiter FDF = 1
Elementary test cases	There is one elementary test to perform for at least 1 bit rate configuration. #1 The LT forces the CRC delimiter bit to dominant from the second TQ until the beginning of Phase_Seg2(N). Refer to 6.2.3.
Set-up	The IUT is left in the default state.
Execution	The LT generates a frame with last CRC bit dominant. The LT forces the CRC delimiter bit to dominant according to elementary test cases.
Response	The modified CRC delimiter bit shall be sampled as dominant. The frame is invalid. The CRC delimiter shall be followed by an error frame.

8 Test type 2, transmitted frame

8.1 Test class 1, valid frame format

8.1.1 Identifier and number of data bytes test in base format

Table 103 specifies the test of the correct transmission of frames in base frame format with different identifiers and data field lengths.

Table 103 — Identifier and number of data bytes test in base format

Item	Description	
Purpose	This test verifies the capacity of the IUT to transmit a frame with different identifiers and different numbers of data in a base format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID DLC FDF = 0	ID DLC FDF = 1, res = 0, BRS = 1, ESI = 0 A device with limited payload bytes will be tested with the CC_h padding payload for the not supported bytes of payload.
Elementary test cases	The CAN ID shall be element of: ∈ [000_h, $7FF_h$] Different CAN IDs are used for test. #1 CAN ID = 555_h #2 CAN ID = $2AA_h$ #3 CAN ID = 000_h #4 CAN ID = $7FF_h$ #5 CAN ID = a random value Tested number of data bytes:∈ [0, 8]. Number of tests: 9 × selected ID	The CAN ID shall be element of: ∈ [000_h, $7FF_h$] Different CAN IDs are used for test. #1 CAN ID = 555_h #2 CAN ID = $2AA_h$ #3 CAN ID = 000_h #4 CAN ID = $7FF_h$ #5 CAN ID = a random value Tested number of data bytes: ∈ [0, 8] ∪[12] ∪ [16] ∪ [20] ∪ [24] ∪ [32] ∪ [48] ∪ [64] Number of tests: 16 × selected ID
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for each elementary test. The LT causes the IUT to transmit a data frame with the parameters according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The content of the frame shall match the LT request.	

8.1.2 Identifier and number of data bytes test in extended format

Table 104 specifies the test of the correct transmission of frames in extended format with different identifiers and data field lengths.

Table 104 — Identifier and number of data bytes test in extended format

Item	Description	
Purpose	This test verifies the capacity of the IUT to transmit a data frame with different identifiers and different numbers of data in an extended format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID DLC FDF = 0 r0 = 0	ID DLC FDF = 1, res = 0, BRS = 1, ESI = 0 A device with limited payload bytes will be tested with the CC_h padding payload for the not supported bytes of payload.
Elementary test cases	The CAN ID shall be element of: ∈ [00000000_h, $1FFFFFFF_h$] Different CAN IDs are used for test. #1 CAN ID = 15555555_h #2 CAN ID = $0AAAAAAA_h$ #3 CAN ID = 00000000_h #4 CAN ID = $1FFFFFFF_h$ #5 CAN ID = random value Tested number of data bytes: ∈ [0, 8] Number of tests: 9 × selected ID	The CAN ID shall be element of: ∈ [00000000_h, $1FFFFFFF_h$] Different CAN IDs are used for test. #1 CAN ID = 15555555_h #2 CAN ID = $0AAAAAAA_h$ #3 CAN ID = 00000000_h #4 CAN ID = $1FFFFFFF_h$ #5 CAN ID = random value Tested number of data bytes: ∈ [0, 8] ∪[12] ∪ [16] ∪ [20] ∪ [24] ∪ [32] ∪ [48] ∪ [64] Number of tests: 16 × selected ID
Set-up	The IUT is left in the default state.	
Execution	A single test frame is used for each elementary test. The LT causes the IUT to transmit a data frame with the parameters according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The content of the frame shall match the LT request.	
NOTE	An implementation with limited ID range may not be able to transmit the frame.	

8.1.3 Arbitration in base format frame

Table 105 specifies the test of the capability to manage the arbitration mechanism on every bit position in a base format frame of the IUT.

Table 105 — Arbitration in base format frame

Item	Description	
Purpose	This test verifies the capability of the IUT to manage the arbitration mechanism on every bit position in a base format frame it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID RTR FDF = 0	ID RRS = 0 FDF = 1

Table 105 *(continued)*

Item	Description
Elementary test cases	For an OPEN device, there are, at most, 11 elementary tests to perform. **Transmitted frame** / **ID** / **RTR/RRS** / **DATA field** / **Description of the concerned arbitration bit(s)** / **Number of elementary test** $7EF_h$ / 0 / No data / Collision on all bits equal to 1 / 10 010_h / 0 / No data / Collision on all bits equal to 1 / 1 For a SPECIFIC device, all possible possibilities of transmitting a recessive arbitration bit shall be considered. For CAN FD enabled test, the RTR is represented by RRS and transmitted as 0.
Set-up	The IUT is left in the default state.
Execution	The LT causes the IUT to transmit a frame. Then, the LT forces a recessive bit in the arbitration field to the dominant state according to the table in elementary test cases and continues to send a valid frame.
Response	The IUT shall become receiver when sampling the dominant bit sent by the LT. As soon as the bus is idle, the IUT shall restart the transmission of the frame. The IUT shall not generate any error flag during the test. The content of the frame shall match the LT request.

8.1.4 Arbitration in extended format frame test

Table 106 specifies the test of the capacity that manages the arbitration mechanism on every bit position in an extended format frame of the IUT.

Table 106 — Arbitration in extended format frame test

Item	Description	
Purpose	This test verifies the capacity of the IUT to manage the arbitration mechanism on every bit position in an extended format frame it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID RTR FDF = 0	ID RRS = 0 FDF = 1
Elementary test cases	For an OPEN device, there are at most 31 elementary tests to perform. **Transmitted frame** / **ID** / **RTR/RRS** / **DATA field** / **Description of the concerned bit(s)** / **Number of elementary test** $1FBFFFFF_h$ / 0 / No data / Collision on all bits equal to 1 / 28 00400000_h / 0 / No data / Collision on all bits equal to 1 / 1 00400000_h / 0 / No data / Collision on SRR and IDE bit / 2 For a SPECIFIC device, all possible cases of transmitting a recessive arbitration bit shall be considered. For CAN FD enabled test, the RTR is represented by RRS and transmitted as 0.	
Set-up	The IUT is left in the default state.	

Table 106 *(continued)*

Item	Description
Execution	The LT causes the IUT to transmit a frame. The LT forces a recessive bit in the arbitration field to the dominant state according to the table in elementary test cases and continues to send a valid frame.
Response	The IUT shall become receiver when sampling the dominant bit sent by the LT. As soon as the bus is idle, the IUT shall restart the transmission of the frame. The IUT shall not generate any error flag during the test. The content of the frame shall match the LT request.
NOTE	An implementation with limited ID range may not be able to transmit the frame.

8.1.5 Message validation

Table 107 specifies the test to the correct point of time at which a message transmitted by the IUT is taken to be valid.

Table 107 — Message validation

Item	Description	
Purpose	The purpose of this test is to verify the point of time at which a message transmitted by the IUT is taken to be valid.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform: #1 On the first bit of the intermission field of the frame sent by the IUT, the LT forces the bit value to dominant.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a data frame. The LT causes the IUT to generate an overload frame according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The IUT shall not restart any frame after the overload frame.	

8.1.6 Stuff bit generation capability in base format frame

Table 108 specifies the test of the correct generation of stuff bits in a base format frame.

Table 108 — Stuff bit generation capability in base format frame

Item	Description	
Purpose	The purpose of this test is to verify that an IUT correctly generates the stuff bits in a base format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID RTR DLC DATA FDF = 0	ID RRS BRS ESI DLC DATA FDF = 1
Elementary test cases	For an OPEN device, there are six elementary tests to perform. CBFF **ID** / **CTRL** / **DATA** #1 078_h / 08_h / 01_h, all others bytes $E1_h$ #2 $41F_h$ / 01_h / 00_h #3 $47F_h$ / 01_h / $1F_h$ #4 758_h / 00_h / — #5 777_h / 01_h / $1F_h$ #6 $7EF_h$ / 42_h / — For a SPECIFIC device, all possible dominant and recessive stuff bits inside and at the end of each stuffed field shall be considered.	The following cases are tested. FBFF **ID** / **CTRL** / **DATA** #1 078_h / 0AEh / $F8_h$, all others bytes 78_h #2 $47C_h$ / $0A8_h$ / all bytes $3C_h$ #3 $41E_h$ / $0BE_h$ / all bytes $1E_h$ #4 $20F_h$ / $09F_h$ / all bytes $0F_h$ #5 107_h / $08F_h$ / all bytes 87_h #6 $7C3_h$ / 083_h / all bytes $C3_h$ #7 $3E1_h$ / $0A3_h$ / all bytes $E1_h$ #8 $1F0_h$ / $0A1_h$ / all bytes $F0_h$ #9 000_h / $0A0_h$ / — #10 $7FF_h$ / $0B0_h$ There are 10 elementary tests to perform.
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The IUT shall correctly generate all stuff bits.	

8.1.7 Stuff bit generation capability in extended frame

Table 109 specifies the test of the correct generation of stuff bits in an extended frame.

Table 109 — Stuff bit generation capability in extended frame

Item	Description	
Purpose	The purpose of this test is to verify that an IUT correctly generates the stuff bits in an extended frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID SRR RTR DLC DATA FDF = 0	ID SRR RRS BRS ESI DLC DATA FDF = 1
Elementary test cases	The following are the cases for an OPEN device. CEFF **ID** **CTRL** **DATA** #1 $07C30F0F_h$ 188_h all bytes $3C_h$ #2 $07C0F0F0_h$ 181_h 00_h #3 $1FB80000_h$ 181_h $A0_h$ There are three elementary tests to perform. For a SPECIFIC device, all possible dominant and recessive stuff bits inside and at the end of each stuffed field shall be considered.	FEFF **ID** **CTRL** **DATA** #1 $01E38787_h$ $6AE_h$ $F8_h$ all others bytes 78_h #2 $11F3C3C3_h$ $6A8_h$ all bytes $3C_h$ #3 $1079C1E1_h$ $6BE_h$ all bytes $1E_h$ #4 $083dF0F0_h$ $69F_h$ all bytes $0F_h$ #5 $041EF878_h$ $68F_h$ all bytes 87_h #6 $1F0C3C3C_h$ 683_h all bytes $C3_h$ #7 $0F861E1E_h$ $6A3_h$ all bytes $E1_h$ #8 $07C30F0F_h$ $6A1_h$ all bytes $F0_h$ #9 $1FFC0000_h$ $6A0_h$ — #10 $0003FFFF_h$ $6B0_h$ —
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame according to elementary test cases.	
Response	The IUT shall not generate any error flag during the test. The IUT shall correctly generate all stuff bits.	

8.1.8 Transmission on the third bit of intermission field after arbitration lost

Table 110 specifies the test of an IUT that is able to transmit a frame on reception of an SOF starting at the third bit of the intermission field following the arbitration winning frame.

Table 110 — Transmission on the third bit of intermission field after arbitration lost

Item	Description	
Purpose	The purpose of this test is to verify that an IUT is able to transmit a frame on reception of an SOF starting at the third bit of the intermission field following the arbitration winning frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission field = 2 bit FDF = 0	Intermission field = 2 bit FDF = 1
Elementary test cases	For OPEN devices, the identifier shall start with 4 dominant bits. For a SPECIFIC device which cannot send such an identifier, any other value may be used. There are two elementary tests to perform: #1 the identifier shall start with 4 dominant bits; #2 the identifier shall start with 5 recessive bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame according to elementary test cases. The LT sends a frame with higher priority at the same time, to force an arbitration lost for the frame sent by the IUT. At start of intermission, the LT waits for 2 bit times before sending an SOF.	
Response	The IUT shall repeat the frame starting with the identifier without transmitting any SOF.	

8.2 Test class 2, error detection

8.2.1 Bit error test in base format frame

Table 111 specifies the test of the ability of the IUT to detect a bit error when the bit it transmits in a base format frame is different from the bit it receives.

Table 111 — Bit error test in base format frame

Item	Description	
Purpose	This test verifies that the IUT detects a bit error when the bit it is transmitting in a base format frame is different with the bit it receives.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Each frame field with exception of the arbitration field where only dominant bits shall be modified and the ACK slot that will not be tested. FDF = 0	Each frame field with exception of the arbitration field where only dominant bits shall be modified and the ACK slot that will not be tested. DLC - to cause different CRC types FDF = 1

Table 111 *(continued)*

Item	Description	
Elementary test cases	The test shall, at minimum, modify at least 1 dominant and 1 recessive bit in each field of the frame except for the arbitration field for which only dominant bits shall be modified. The ACK slot is not tested. There are 13 elementary tests to perform.	The test shall, at minimum, modify at least 1 dominant and 1 recessive bit in each field of the frame except of the arbitration field for where only dominant bits shall be modified. The ACK slot is not tested. Bit error in a fix stuff bit for CRC (17) and CRC (21) + bit error in CRC (17) and CRC (21) There are 21 elementary tests to perform.
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit the frames and creates a bit error according to elementary test cases.	
Response	The IUT shall generate an active error frame starting at the bit position following the corrupted bit. The IUT shall restart the transmission of the data frame as soon as the bus is idle.	

8.2.2 Bit error in extended format frame

Table 112 specifies the test of the ability of the IUT to detect a bit error when the bit it transmits in an extended format frame is different from the bit it receives.

Table 112 — Bit error in extended frame test

Item	Description	
Purpose	This test verifies that the IUT detects a bit error when the bit it is transmitting in an extended frame is different from the bit it receives.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Each frame field with exception of the arbitration field where only dominant bits shall be modified and the ACK slot that will not be tested. FDF = 0	Each frame field with exception of the arbitration field where only dominant bits shall be modified and the ACK slot that will not be tested. DLC - to cause different CRC types FDF = 1
Elementary test cases	The test shall modify at least 1 dominant extended identifier bit and the “FDF”, “r0” bits. There are 14 elementary tests to perform.	The test shall modify at least 1 dominant extended identifier bit, bit error in fix stuff bit for CRC (17) and CRC (21) + bit error in CRC (17) and CRC (21) There are 21 elementary tests to perform.
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit the frames and creates a bit error according to elementary test cases.	
Response	The IUT shall generate an active error frame starting at the bit position following the corrupted bit. The IUT shall restart the transmission of the data frame as soon as the bus is idle.	

8.2.3 Stuff error test in base format frame

Table 113 specifies the test of the detection of an error in a base frame when after the transmission of 5 identical bits, the IUT receives a sixth bit identical to the five precedents.

Table 113 — Stuff error test in base format frame

Item	Description	
Purpose	This test verifies that the IUT detects an error when after the transmission of 5 identical bits, it receives a sixth bit identical to the five precedents. This test is executed with a base format frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ID RTR DLC DATA FDF = 0	ID DLC DATA byte 0 defined in test case, all other DATA byte = 55_h FDF = 1
Elementary test cases	All stuff bits within the defined frames will be tested. There are 35 elementary tests to perform. CEFF **ID / CTRL / DATA** #1 078_h 08_h 01_h, all others bytes $E1_h$ #2 $41F_h$ 01_h 00_h #3 $47F_h$ 01_h $1F_h$ #4 758_h 00_h — #5 777_h 01_h $1F_h$ #6 $7EF_h$ 42_h — For an OPEN device, at least one stuff error shall be generated at each stuffed field. For a SPECIFIC device, at least one stuff error shall be generated at each stuffed field, where a stuff bit can occur.	All stuff bits up to the second payload byte within the defined frames will be tested. There are 39 elementary tests to perform. FBFF **ID / CTRL / DATA** #1 078_h $0AE_h$ $F8_h$ #2 $47C_h$ $0A8_h$ $3C_h$ #3 $41E_h$ $0BE_h$ $1E_h$ #4 $20F_h$ $09F_h$ $0F_h$ #5 107_h $08F_h$ 87_h #6 $7C3_h$ 083_h $C3_h$ #7 $3E1_h$ $0A3_h$ $E1_h$ #8 $1F0_h$ $0A1_h$ $F0_h$ #9 000_h $0A0_h$ — #10 $7FF_h$ $0B0_h$ —
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit the frames and creates a stuff error according to elementary test cases.	
Response	The IUT shall generate an error frame at the bit position following the corrupted stuff bit. The IUT shall restart the transmission of the data frame as soon as the bus is idle.	

8.2.4 Stuff error test in extended frame format

Table 114 specifies the test of the detection of an error in an extended frame when after the transmission of 5 identical bits, it receives a sixth bit identical to the five precedents.

Table 114 — Stuff error test in extended frame format

<table>
<tr><th>Item</th><th colspan="2">Description</th></tr>
<tr><td>Purpose</td><td colspan="2">This test verifies that the IUT detects an error when after the transmission of 5 identical bits, it receives a sixth bit identical to the five precedents. This test is executed with an extended frame.</td></tr>
<tr><td>CAN_VERSION</td><td>Classical CAN
CAN FD tolerant
CAN FD enabled</td><td>CAN FD enabled</td></tr>
<tr><td>Test variables</td><td>ID
RTR
DLC
DATA
FDF = 0</td><td>ID
SRR
RRS
BRS
ESI
DLC
DATA byte 0 defined in test case, all other DATA byte = 55_h.
FDF = 1</td></tr>
<tr><td>Elementary test cases</td><td>All stuff bits within the defined frames will be tested.
There are 35 elementary tests to perform.
CEFF
ID / CTRL / DATA
#1 $07C30F0F_h$ / 188_h / All byte $3C_h$
#2 $07C0F0F0_h$ / 181_h / 00_h
#3 $1FB80000_h$ / 181_h / $A0_h$
#4 00000000_h / 181_h / 00_h
For an OPEN device, at least one stuff error shall be generated at each stuffed field.
For a SPECIFIC device, at least one stuff error shall be generated at each stuffed field, where a stuff bit can occur.</td><td>All stuff bits up to the second payload byte within the defined frames will be tested.
There are 79 elementary tests to perform.
FEFF
ID / CTRL / DATA
#1 $01E38787_h$ / $6AE_h$ / $F8_h$
#2 $11F38787_h$ / $6A8_h$ / $3C_h$
#3 $1079C1E1_h$ / $6BE_h$ / $1E_h$
#4 $083dF0F0_h$ / $69F_h$ / $0F_h$
#5 $041EF878_h$ / $68F_h$ / 87_h
#6 $1F0C3C3C_h$ / 683_h / $C3_h$
#7 $0F861E1E_h$ / $6A3_h$ / $E1_h$
#8 $07C30F0F_h$ / $6A1_h$ / $F0_h$
#9 $1C3FC3C3_h$ / $6A0_h$ / —
#10 $020FE1FF_h$ / $6B0_h$ / —</td></tr>
<tr><td>Set-up</td><td colspan="2">The IUT is left in the default state.</td></tr>
<tr><td>Execution</td><td colspan="2">The LT causes the IUT to transmit the frames and creates a stuff error according to elementary test cases.</td></tr>
<tr><td>Response</td><td colspan="2">The IUT shall generate an error frame at the bit position following the corrupted stuff bit.
The IUT shall restart the transmission of the data frame as soon as the bus is idle.</td></tr>
</table>

8.2.5 Form error test

Table 115 specifies the test of the detection of a form error when the transmitted fixed-form bit field is different from the received bit.

Table 115 — Form error test

Item	Description	
Purpose	This test verifies that the IUT detects a form error when the transmitted fixed-form bit field is different from the bit it receives.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	CRC delimiter ACK delimiter EOF (the first, the fourth and the last one) FDF = 0	CRC delimiter ACK delimiter, (2 ACK bits are used) EOF (the first, the fourth and the last one) Fixed stuff bit at CRC (17) Fixed stuff bit at CRC (21) FDF = 1
Elementary test cases	There are five elementary tests to perform. #1 CRC delimiter #2 ACK delimiter #3 EOF bit 1 #4 EOF bit 4 #5 EOF bit 7	There are 16 elementary tests to perform. #1 CRC delimiter #2 ACK delimiter #3 EOF bit 1 #4 EOF bit 4 #5 EOF bit 7 #6 Fix stuff bit at CRC (17) – (6 bits) #7 Fix stuff bit at CRC (21) – (7 bits)
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT creates a form error on the fields listed in elementary test cases.	
Response	The IUT shall generate an error frame at the bit position following the corrupted bit. The IUT shall restart the transmission of the frame as soon as the bus is idle.	

8.2.6 Acknowledgement error

Table 116 specifies the test of the detection of an acknowledgement error when the received ACK slot is recessive.

Table 116 — Acknowledgement error

Item	Description	
Purpose	This test verifies that the IUT detects an acknowledgement error when the received ACK slot is recessive.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	ACK slot: 1 bit FDF = 0	ACK slot: 2 bits FDF = 1
Elementary test cases	There is one elementary test to perform. #1 ACK slot = recessive	
Set-up	The IUT is left in the default state.	

Table 116 *(continued)*

Item	Description
Execution	The LT causes the IUT to transmit a base format frame. Then, the LT does not send the ACK slot according to elementary test cases.
Response	The IUT shall generate an error frame starting at the bit position following the ACK slot. The IUT shall restart the transmission of the frame as soon as the bus is idle.
NOTE	For classical format frame usage, the IUT shall generate an error frame starting at the bit position following the 1-bit wide ACK slot. For FD format frame, the IUT shall generate an error frame starting at the bit position following the 2-bit wide ACK slot.

8.2.7 Form field tolerance test for FD frame format

Table 117 specifies the test to check the behaviour in the CRC delimiter and acknowledge field when these fields are extended to 2 bits.

Table 117 — Form field tolerance test for FD frame format

Item	Description
Purpose	This test verifies the behaviour in the CRC delimiter and acknowledge field when these fields are extended to 2 bits.
CAN_VERSION	CAN FD enabled
Test variables	CRC delimiter ACK slot FDF = 1
Elementary test cases	There are two elementary tests to perform: #1 CRC delimiter up to 2-bit long (late ACK bit – long distance); #2 ACK up to 2-bit long (superposing ACK bits – near and long distance).
Set-up	The IUT is left in the default state.
Execution	The LT causes the IUT to transmit a frame. Then, the LT creates a CRC delimiter and an ACK bit as defined in elementary test cases.
Response	The frame is valid. The IUT shall not generate an error frame.

8.2.8 Bit error at stuff bit position for FD frame payload bytes

Table 118 specifies the test that the IUT detects an error when after the transmission of 5 identical bits, it receives a sixth bit identical to the five precedents until the position of the CRC in FD format frame with maximum payload DLC = 15.

Table 118 — Bit error at stuff bit position for FD frame payload bytes

Item	Description
Purpose	This test verifies that the IUT detects an error when after the transmission of 5 identical bits, it receives a sixth bit identical to the five precedents.
CAN_VERSION	CAN FD enabled
Test variables	DATA byte 0 to 63 ID = 555_h IDE = 0 DLC = 15 FDF = 1

Table 118 *(continued)*

Item	Description
Elementary test cases	All 1 008 stuff bit positions within the defined data bytes will be tested. Data byte 0 Data bytes 1 to 63 #1 to #126 10_h 78_h #127 to #252 78_h $3C_h$ #253 to #378 34_h $1E_h$ #379 to #504 12_h $0F_h$ #505 to #630 $0F_h$ 87_h #631 to #756 17_h $C3_h$ #757 to #882 43_h $E1_h$ #883 to #1 008 21_h $F0_h$
Set-up	The IUT is left in the default state.
Execution	A single test frame is used for each elementary test. In each elementary test, the LT forces another one of the stuff bits to its complement.
Response	The IUT shall generate an active error frame starting at the bit position following the bit error at stuff bit position.

8.3 Test class 3, error frame management

8.3.1 Error flag longer than 6 bit

Table 119 specifies the test of an IUT acting as a transmitter tolerates up to 7 dominant bits after sending its own error flag.

Table 119 — Error flag longer than 6 bits

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter tolerates up to 7 dominant bits after sending its own error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 the LT extends the error flag by 1 dominant bit; #2 the LT extends the error flag by 4 dominant bits; #3 the LT extends the error flag by 7 dominant bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. The LT corrupts this frame in data field causing the IUT to send an active error frame. The LT prolongs the error flag sent by IUT according to elementary test cases.	
Response	The IUT shall generate only one error frame. The IUT shall restart the transmission after the intermission field following the error frame.	

8.3.2 Transmission on the third bit of intermission field after error frame

Table 120 specifies the test of an IUT that is able to transmit a frame on reception of an SOF starting at the third bit of the intermission field following the error frame it has transmitted.

Table 120 — Transmission on the third bit of intermission field

Item	Description	
Purpose	The purpose of this test is to verify that an IUT is able to transmit a frame on reception of an SOF starting at the third bit of the intermission field following the error frame it has transmitted.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission Field = 2 Bit FDF = 0	Intermission Field = 2 Bit FDF = 1
Elementary test cases	There are two elementary tests to perform: #1 the identifier shall start with 4 dominant bits; #2 the identifier shall start with 5 recessive bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame according to elementary test cases. The LT corrupts this frame in data field causing the IUT to send an active error frame. At the end of the error flag sent by the IUT, the LT waits for (8 + 2) bit times before sending an SOF.	
Response	The IUT shall repeat the frame starting with the identifier without transmitting any SOF.	

8.3.3 Bit error in error flag

Table 121 specifies the test of an IUT acting as a transmitter that detects a bit error when one of the 6 dominant bits of the error flag it transmits is forced to recessive state by LT.

Table 121 — Bit error in error flag

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter detects a bit error when one of the 6 dominant bits of the error flag it transmits is forced to recessive state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the error flag; #2 corrupting the fourth bit of the error flag; #3 corrupting the sixth bit of the error flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. The LT corrupts this frame in data field causing the IUT to send an active error frame. Then the LT forces one of the 6 bits of the active error flag sent by the IUT to recessive state according to elementary test cases.	
Response	The IUT shall restart its active error flag at the bit position following the corrupted bit.	

8.3.4 Form error in error delimiter

Table 122 specifies the test of an IUT acting as a transmitter that detects a form error when it receives an invalid error delimiter.

Table 122 — Form error in error delimiter

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter detects a form error when it receives an invalid error delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the second bit of the error delimiter; #2 corrupting the fourth bit of the error delimiter; #3 corrupting the seventh bit of the error delimiter.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. The LT corrupts this frame in data field causing the IUT to send an active error frame. Then, the LT forces 1 recessive bit of the error delimiter to the dominant state according to elementary test cases.	
Response	The IUT shall restart the error frame at the bit position following the corrupted bit.	

8.4 Test class 4, overload frame management

8.4.1 MAC overload generation in intermission field

Table 123 specifies the test of an IUT acting as transmitter that generates an overload frame when detecting a dominant bit on one of the 2 first recessive bits of the intermission field following a data frame it is transmitting.

Table 123 — MAC overload generation in intermission field

Item	Description	
Purpose	This test verifies that an IUT acting as transmitter generates an overload frame when it detects a dominant bit on one of the 2 first recessive bits of the intermission field following a data frame it is transmitting.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 dominant bit on first bit of the intermission field; #2 dominant bit on second bit of the intermission field.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT forces one of the bits of the intermission field to the dominant state according to elementary test cases.	
Response	The IUT shall generate an overload frame starting at the bit position following the dominant bit generated by the LT.	

8.4.2 Eighth bit of an error and overload delimiter

Table 124 specifies the test of an IUT that generates an overload frame when detecting a dominant bit on the eighth bit of an error or an overload delimiter.

Table 124 — Eighth bit of an error and overload delimiter

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter generates an overload frame when it detects a dominant bit on the eighth bit of an error or an overload delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 dominant bit on the eighth bit of an error delimiter, error applied in data field; #2 dominant bit on the eighth bit of an overload delimiter following a data frame.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to generate an error frame or overload frame according to elementary test cases. Then, the LT forces the eighth bit of the delimiter to a dominant state.	
Response	The IUT shall generate overload frames starting at the bit position following the dominant bit sent by the LT.	

8.4.3 Transmission on the third bit of intermission after overload frame

Table 125 specifies the test of an IUT that is able to transmit a data frame starting with the identifier and without transmitting SOF.

Table 125 — Transmission on the third bit of intermission field

Item	Description	
Purpose	The purpose of this test is to verify that an IUT is able to transmit a data frame starting with the identifier and without transmitting SOF, when detecting a dominant bit on the third bit of the intermission field following an overload frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	Intermission field = 2 bit FDF = 0	Intermission field = 2 bit FDF = 1
Elementary test cases	For OPEN devices, the identifier shall start with 4 dominant bits. For a SPECIFIC device which cannot send such an identifier, any other value may be used. There are two elementary tests to perform: #1 the identifier shall start with 4 dominant bits; #2 the identifier shall start with 5 recessive bits.	

Table 125 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT causes the IUT to transmit a frame according to elementary test cases. The LT disturbs the transmitted frame with an error frame, then the LT causes the IUT to generate an overload frame immediately after the error frame. Then, the LT forces the third bit of the intermission following the overload delimiter to dominant state.
Response	The IUT shall repeat the frame starting with the identifier without transmitting any SOF.

8.4.4 Bit error in overload flag

Table 126 specifies the test of an IUT that detects a bit error when one of the 6 dominant bits of the overload flag it transmits is forced to recessive state by LT.

Table 126 — Bit error in overload flag

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter detects a bit error when one of the 6 dominant bits of the overload flag it transmits is forced to recessive state by LT.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the overload flag; #2 corrupting the second bit of the overload flag; #3 corrupting the sixth bit of the overload flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to generate an overload frame. Then, the LT corrupts one of the 6 dominant bits of the overload flag to the recessive state according to elementary test cases.	
Response	The IUT shall generate an error frame starting at the bit position after the corrupted bit.	

8.4.5 Form error in overload delimiter

Table 127 specifies the test of an IUT that detects a form error when it receives an invalid overload delimiter.

Table 127 — Form error in overload delimiter

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter detects a form error when it receives an invalid overload delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1

Table 127 *(continued)*

Item	Description
Elementary test cases	Elementary tests to perform: #1 corrupting the second bit of the overload delimiter; #2 corrupting the fourth bit of the overload delimiter; #3 corrupting the seventh bit of the overload delimiter.
Set-up	The IUT is left in the default state.
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to generate an overload frame. The LT corrupts the overload delimiter according to elementary test cases.
Response	The IUT shall generate an error frame starting at the bit position following the corrupted bit.

8.5 Test class 5, passive error state and bus-off

8.5.1 Acceptance of active error flag overwriting passive error flag

Table 128 specifies the test of a passive state IUT that does not detect any error when detecting an active flag during its own passive error flag.

Table 128 — Acceptance of active error flag overwriting passive error flag

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter does not detect any error when detecting an active error flag during its own passive error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 superposing the passive error flag by an active error flag starting at the first bit; #2 superposing the passive error flag by an active error flag starting at the third bit; #3 superposing the passive error flag by an active error flag starting at the sixth bit.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to send a passive error flag in data field. During the passive error flag sent by the IUT, the LT sends an active error flag in date field according to elementary test cases. At the end of the error flag, the LT waits for (8 + 3) bit time before sending a frame.	
Response	The IUT shall acknowledge the last frame transmitted by the LT.	

8.5.2 Frame acceptance after passive error frame transmission

Table 129 specifies the test of a passive state IUT that accepts to receive a frame starting after passive error frame transmission.

Table 129 — Frame acceptance after passive error frame transmission

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter accepts to receive a frame starting after the second bit of the intermission following the error frame it has transmitted.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 LT waits for (8 + 2) bit time before sending a frame	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to send a passive error flag in data field. During the passive error flag sent by the IUT, the LT sends an active error flag. At the end of the error flag, the LT send a valid frame according to elementary test cases.	
Response	The IUT shall acknowledge the frame.	

8.5.3 Acceptance of 7 consecutive dominant bits after passive error flag

Table 130 specifies the test of a passive state IUT that does not detect any error when detecting dominant bits after passive error flag.

Table 130 — Acceptance of 7 consecutive dominant bits after passive error flag

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter does not detect any error when detecting dominant bits during the 7 first bit of the error delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 transmitting 1 consecutive dominant bits; #2 transmitting 4 consecutive dominant bits; #3 transmitting 7 consecutive dominant bits.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a data frame. Then, the LT causes the IUT to send a passive error flag in data field. At the end of error flag, the LT continues transmitting dominant bits according to elementary test cases. At this step, the LT waits for (8 + 3) bit time before sending a frame.	
Response	The IUT shall acknowledge the frame transmitted by the LT. The IUT shall restart the transmission of the corrupted frame (1 + 7 + 3) bit times after its ACK bit.	

8.5.4 Reception of a frame during suspend transmission

Table 131 specifies the test of a passive state IUT that is able to receive a frame during suspend transmission.

Table 131 — Reception of a frame during suspend transmission

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter is able to receive a frame during the suspend transmission.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 the received frame starts on the first bit of the suspend transmission; #2 the received frame starts on the fourth bit of the suspend transmission; #3 the received frame starts on the eighth bit of the suspend transmission.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. At the end of the EOF and intermission fields, the LT sends a frame according to elementary test cases.	
Response	The IUT shall acknowledge the last frame transmitted by the LT.	

8.5.5 Transmission of a frame after suspend transmission — Test case 1

Table 132 specifies the test of a passive state IUT that transmits a frame after suspend transmission.

Table 132 — Transmission of a frame after suspend transmission — Test 1

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter does not transmit any frame before the end of the suspend transmission following an error frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 After passive error flag, the LT forces the bus to recessive for error delimiter + intermission + suspend transmission time.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to send a passive error frame in data field according to elementary test cases.	
Response	After the intermission following the error frame, the LT verifies that the IUT wait 8 more bits before re-transmitting the frame.	

8.5.6 Transmission of a frame after suspend transmission — Test case 2

Table 133 specifies the test of a passive state IUT that transmits a data frame after suspend transmission following an overload frame.

Table 133 — Transmission of a frame after suspend transmission — Test 2

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT being transmitted does not transmit any data frame before the end of the suspend transmission following an overload frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 After overload flag, the LT forces the bus to recessive for overload delimiter + intermission + suspend transmission time.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit two frames. The LT lets the IUT transmit the first frame and causes the IUT to generate an overload frame according to elementary test cases.	
Response	After the intermission following the overload frame, the LT verifies that the IUT wait 8 more bits before transmitting the second frame.	

8.5.7 Transmission of a frame after suspend transmission — Test case 3

Table 134 specifies the test of a passive state IUT that transmits a frame after suspend transmission.

Table 134 — Transmission of a frame after suspend transmission — Test 3

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter does not transmit any frame before the end of the suspend transmission following a frame.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 After the dominant ACK, the LT forces the bus to recessive for ACK delimiter + EOF + intermission + suspend transmission time.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit two frames. The LT lets the IUT transmit the first frame according to elementary test cases. This frame shall end with EOF field followed by the intermission field.	
Response	After the intermission following the first frame, the LT verifies that the IUT wait 8 more bits before transmitting the second frame.	

8.5.8 Transmission of a frame without suspend transmission

Table 135 specifies the test of a passive state IUT, after losing arbitration that repeats the frame without inserting any suspend transmission.

Table 135 — Transmission of a frame without suspend transmission

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT, after losing arbitration, repeats the frame without inserting any suspend transmission.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 The LT causes the IUT to lose arbitration by sending a frame of higher priority.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame according to elementary test cases.	
Response	The LT verifies that the IUT re-transmits its frame (1 + 7 + 3) bit times after acknowledging the received frame.	

8.5.9 No transmission of a frame between the third bit of intermission field and eighth bit of suspend transmission

Table 136 specifies the test of a passive state IUT that does not transmit a frame when detecting a dominant bit at the third bit of the intermission frame.

Table 136 — No transmission of a frame between the third bit of intermission field and end of suspend transmission

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT does not transmit a frame starting with an identifier and without transmitting SOF when detecting a dominant bit on the third bit of the intermission field.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary test to perform: #1 dominant bit on the third bit of the intermission field; #2 dominant bit on the first bit of Suspend transmission; #3 dominant bit on the seventh bit of Suspend transmission.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. The LT corrupts this frame and corrupts the following error frame to set the IUT into passive state. The error frame shall end with error delimiter followed by the intermission field. At the bit, as specified in elementary test cases after the error delimiter, the LT starts sending a frame with lower priority	
Response	The IUT shall not start the re-transmission before the end of the frame sent by the LT.	
NOTE	The IUT shall acknowledge the frame sent by the LT.	

8.5.10 Bus-off state

Table 137 specifies that an IUT that switches to bus-off state no longer sends dominant bits.

Table 137 — Bus-off state

Item	Description	
Purpose	The purpose of this test is to verify that an IUT switching to bus-off state no longer sends dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 The LT forces the bus to recessive for bus-off recovery time (22 bits).	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame twice. The LT causes the IUT to generate an error frame. During the error flag transmitted by the IUT, the LT forces recessive state during 16 bit times. After the following passive error flag, the error delimiter is forced to dominant state for 112 bit times. Then, the IUT transmits its first frame. The LT acknowledges the frame and immediately causes the IUT to generate an overload frame. The LT forces the first bit of this overload flag to recessive state creating a bit error. (6 + 7) bit times later, the LT generates a dominant bit to cause the IUT to generate a new overload frame. The LT forces the first bit of this new overload flag to recessive state causing the IUT to increments its TEC to the bus-off limit. (6 + 8 + 3 + 8) bit times later, the LT sends a valid frame according to elementary test cases.	
Response	Only one frame shall be transmitted by the IUT. The IUT shall not acknowledge the frame sent by the LT. Error counter shall be reset after bus-off recovery.	
NOTE	Check error counter after bus-off, if applicable.	

8.5.11 Bus-off recovery

Table 138 specifies the test of an IUT which is bus-off that is not permitted to become error active.

Table 138 — Bus-off recovery

Item	Description	
Purpose	The purpose of this test is to verify that an IUT which is bus-off is not permitted to become error active (no longer bus-off) before 128 occurrences of 11 consecutive recessive bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: 1) the LT sends recessive bus level for at least 1 408 bit times until the IUT becomes active again; 2) the LT sends one group of 10 recessive bits, one group of 21 recessive bits followed by at least 127 groups of 11 recessive bits, each group separated by 1 dominant bit.	

Table 138 *(continued)*

Item	Description
Set-up	The IUT is left in the default state.
Execution	The LT ask the IUT to send a frame and sets it in the bus-off state. The LT sends the profiles defined in elementary test cases.
Response	The IUT shall not transmit the frame before the end of the profiles sent by the LT according to elementary test cases and shall send it before the end of the TIMEOUT. Error counter shall be reset after bus off recovery.
NOTE	Check error counter after bus-off, if applicable.

8.5.12 Completion condition for a passive error flag

Table 139 specifies the test of a passive state IUT that waits for 6 consecutive identical bit to complete its passive error flag.

Table 139 — Completion condition for a passive error flag

Item	Description	
Purpose	The purpose of this test is to verify that a passive state IUT acting as a transmitter waits for 6 consecutive identical bit to complete its passive error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 During the error flag, the LT sends 5 dominant bits, 5 recessive bits and then, 6 dominant bits.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT corrupts a bit in data field to cause the IUT to generate a passive error flag according to elementary test cases. After the 6 dominant bits, the LT waits for 8 bit time before sending a dominant bit.	
Response	The IUT shall generate an overload frame starting at the bit position following the last dominant bit generated by the LT.	

8.5.13 Form error in passive error delimiter

Table 140 specifies the test of an error passive IUT that detects a form error when monitoring a corruption in the error delimiter.

Table 140 — Form error in passive error delimiter

Item	Description	
Purpose	The purpose of this test is to verify that an error passive IUT acting as a transmitter detects a form error when monitoring a corruption in the error delimiter.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1

Table 140 *(continued)*

Item	Description
Elementary test cases	Elementary tests to perform: #1 corrupting the second bit of the error delimiter; #2 corrupting the fourth bit of the error delimiter; #3 corrupting the seventh bit of the error delimiter.
Set-up	The IUT is set to the TEC passive state.
Execution	The LT causes the IUT to transmit a data frame. Then, the LT corrupt a bit in data field to cause the IUT to generate a passive error frame. The LT creates a form error according to elementary test cases. After the form error, the LT waits for (6 + 7) bit time before sending a dominant bit.
Response	The IUT shall generate an overload frame starting at the bit position following the last dominant bit generated by the LT.

8.5.14 Maximum recovery time after a corrupted frame

Table 141 specifies the test of the maximum recovery time of an error passive IUT after a corrupted frame.

Table 141 — Maximum recovery time after a corrupted frame

Item	Description	
Purpose	The purpose of this test is to verify that the recovery time of an error passive IUT detecting an error is at most 31 bit times.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 At the bit position following the end of the passive error flag, the LT starts to send 6 dominant bits.	
Set-up	The IUT is set to the passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT corrupts a bit in data field of this frame causing the IUT to generate a passive error flag according to elementary test cases.	
Response	The IUT shall re-transmit the same frame 31 bit times after the detection of the corrupted bit.	

8.5.15 Transition from active to passive ERROR FLAG

Table 142 specifies the test of an active IUT that changes to an error passive IUT when detecting an error at most 17 bit times.

Table 142 — Transition from active to passive ERROR FLAG

Item	Description	
Purpose	The purpose of this test is to verify that an active IUT changes to an error passive IUT detecting an error is at most 17 bit times.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 The LT check that the repeated frame start 6 + 8 + 3 + 8 bit after the last dominant bit send by LT.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT corrupts a bit in data field of this frame and then, the LT corrupts following error flag to recessive for 16 bit times causing the IUT to generate a passive error flag. The LT receives the repeated frame according to elementary test cases.	
Response	The IUT shall generate a passive error flag and repeat the frame 6 + 8 + 3 + 8 bit after the last dominant bit sent by LT.	

8.6 Test class 6, error counter management

8.6.1 TEC increment on bit error during active error flag

Table 143 specifies the test of an IUT that increases its TEC when detecting a bit error during an active error flag.

Table 143 — TEC increment on bit error during active error flag

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter increases its TEC by 8 when detecting a bit error during the transmission of an active error flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the active error flag; #2 corrupting the third bit of the active error flag; #3 corrupting the sixth bit of the active error flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT corrupts a bit in data field to causes the IUT to generate an active error frame. The LT corrupts one of the dominant bits of the error flag according to elementary test cases.	
Response	The IUT's TEC value shall be increased by 8 on the corrupted bit.	

8.6.2 TEC increment on bit error during overload flag

Table 144 specifies the test of an IUT that increases its TEC when detecting a bit error during an overload flag.

Table 144 — TEC increment on bit error during overload flag

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter increases its TEC by 8 when detecting a bit error during the transmission of an overload flag.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	Elementary tests to perform: #1 corrupting the first bit of the overload flag; #2 corrupting the fourth bit of the overload flag; #3 corrupting the sixth bit of the overload flag.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to generate an overload frame after a data frame. The LT corrupts one of the dominant bits of the overload flag according to elementary test cases.	
Response	The IUT's TEC value shall be increased by 8 at the corrupted bit.	

8.6.3 TEC increment when active error flag is followed by dominant bits

Table 145 specifies the test of an IUT that increases its TEC when an active error flag is followed by dominant bits.

Table 145 — TEC increment when active error flag is followed by dominant bits

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter increases its TEC by 8 when detecting 8 consecutive dominant bits following the transmission of its active error flag and after each sequence of additional 8 consecutive dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 After the error flag sent by the IUT, the LT sends a sequence of 16 dominant bits.	
Set-up	The IUT is left in the default state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT causes the IUT to generate an active error frame in data field according to elementary test cases.	
Response	The IUT's TEC value shall be increased by 8 on each eighth dominant bit after the error flag.	

8.6.4 TEC increment when passive error flag is followed by dominant bits

Table 146 specifies the test of an IUT that increases its TEC when a passive error flag is followed by dominant bits.

Table 146 — TEC increment when passive error flag is followed by dominant bits

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter increases its TEC by 8 when detecting 8 consecutive dominant bits following the transmission of its passive error flag and after each sequence of additional 8 consecutive dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	After the error flag sent by the IUT, the LT sends a sequence of up to 16 dominant bits. There are five elementary tests to perform: #1 dominant bits after passive error flag: 1 bit; #2 dominant bits after passive error flag: 6 bits; #3 dominant bits after passive error flag: 8 bits; #4 dominant bits after passive error flag: 9 bits; #5 dominant bits after passive error flag: 16 bits.	
Set-up	The IUT is set to the TEC passive state.	
Execution	The LT causes the IUT to transmit a frame. Then, the LT corrupts a bit in data field to cause the IUT to generate a passive error frame. After the error flag sent by the IUT, the LT sends a sequence according to elementary test cases.	
Response	The IUT's TEC value shall be increased by 8 on each eighth dominant bit after the error flag.	

8.6.5 TEC increment when overload flag is followed by dominant bits

Table 147 specifies the test of an IUT that increases its TEC when an overload flag is followed by dominant bits.

Table 147 — TEC increment when overload flag is followed by dominant bits

Item	Description	
Purpose	This test verifies that an IUT acting as a transmitter increases its TEC by 8 when detecting 8 consecutive dominant bits following the transmission of its overload flag and after each sequence of additional 8 consecutive dominant bits.	
CAN_VERSION	Classical CAN CAN FD tolerant CAN FD enabled	CAN FD enabled
Test variables	FDF = 0	FDF = 1
Elementary test cases	There is one elementary test to perform. #1 Dominant bits after overload flag: 23 bits	