Specification
eBUS Conformance Testing

Data Link Layer Testing

Asynchronous Mode
Version 1.0 / 22.10.1998
Ref eBUS Specification Layer 1 / 2, Vers 1.1.1

CONTENTS

1 Introduction

This Test Plan refers to the eBUS Specification layer1,2 ver 1.0.
This document does not consider the synchronous mode.

1.1 Architecture of the Test Plan

This Test Plan (TP) is used to validate the OSI layer 2 for the eBUS Specification layer 1,2 ver 1.0. The Test Plan will involve up to three participants; Lower Tester (LT), Upper Tester (UT) and Implementation Under Test (IUT).

An IUT is the same as a device and the UT is acting as a user of the IUT. The LT is sending, receiving and verifying messages on the bus. In hardware the UT is at the top of the IUTs micro controller and the LT resides in a PC, with an interface to the bus. More vividly we may say that the LT tells the UT actions to be performed and the UT provides the LT with information concerning the internal behaviour of the IUT.

1.2 Test Case Organization

Each Test Case consists of four states:

- Set Up State
- Test State
- Verification State
- Final State

1.2.1 Set Up State

State the IUT starts out of before entering the Tests State.

1.2.2 Test State

The state in which one or several test frames are executed.

1.2.3 Verification State

This is a state in which the LT verifies whether the test frames have been handled correctly or not according to the eBus Specification layer 1,2.

1.2.4 Final State

This is the state the IUT has to be in after each Elementary Test.

1.3 Hierarchical Structure of the Tests

The tests are divided into three levels of categories:

- Test Types
- Test Classes
- Test Cases

1.3.1 Test Types

- Receiver tests, includes all tests for data frames received by the IUT.
- Sender tests, includes all tests for data frames sent by the IUT.
- Bidirectional tests, includes all tests for both received and sent by the IUT.

1.3.2 Test Classes

- Error free message handling
- Error detection in message
- Error handling of message

Timing tests are separately headed after the bidirectional tests.

1.3.3 Test Cases

The Test Cases are a further partitioning of Test Classes to test particular details of the eBUS Specification. Sometimes there are several valid values of a parameter and may be subdivided even further into several Elementary Tests. A Test Case may also consist of several Sub Test Cases which in turn are composed of Elementary Tests.

1.4 Description of the Notation of a Test Case

Each Test Case is first explained in an abstract way according to OSI Conformance Testing in the Test Case organization table. The Test Case organization table consists of four different states, where every state has a description. A Test Case can, as mentioned before, consist of one or more Elementary Tests. The description of the different values that are used for each Elementary Test preceeds the Test Case organization table. In the description of the Set Up State only the Idle State is used, its meaning is described in the enclosure. In short, the Idle State is the state when the IUT is ready to receive but not willing to send. After every Elementary Test the Final State should be Idle State. The number of the Test Frames used for each Elementary Test are mentioned in the description of the Test State. In the Verification State the expected reactions of the IUT are explained but also how the verification of the Elementary Test is done.

In the second table an Elementary Test is explained in detail, with three columns; Lower Tester, Flow and Upper Tester. For each column the time starts on the top and ends at the bottom of the table. In the flow column there are two pairs of arrows that represent the data transfer between LT and UT via eBUS and indicate the steps to reach and execute the Set Up State and the Test State. The verification is done in the Lower Tester during and after the Test State. The Final State will be reached after the Test State and the verification. On the Lower Tester and Upper Tester side, functions with parameters are used to describe the behavior of each part.

This second table should be seen as a more detailed but still formal description.
A short explanation of the parameters used throughout this document can be found in the appendix.

2 Function Tests

This documentation mainly contains of function tests, i.e tests for layer 2. The function tests are tests concerning the functionality of the IUT.

The function tests are partitioned into two groups:

- Function tests for master message
- Function tests for slave message

2.1 Function Tests for Master Message

The function tests for master message includes two test types:

- Receiver tests
- Sender tests

2.1.1 Receiver Tests

The receiver tests includes two Test Classes:

- Error free message handling
- Error detection in message

For the receiver tests the third Test Class, error handling of a message, does not include any Test Cases, so this class is not needed.

2.1.1.1 Error Free Message Handling

The error free message handling includes all Test Cases, where the master is receiving error free messages. The Test Cases use a single test frame for each Elementary Test, which means that Lower Tester sends just one test frame for each Elementary Test.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

In Test Case 2.1.1.1.4a the verification is done by adding the first two databytes together to avoid sending the same reserved bytes, which would have been the case if the same verification function would have been used. If the same message is send back the verification is not reliable, because the case could be that the IUT could be able to receive reserved bytes but not able to send.

2.1.1.1.1 Receiving Messages with Different Addresses (QQ)

Master receiving messages with different source addresses.(QQ)

QQ ∈ {all master addresses}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	25 test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.1b		
Start loop Change_Sourceaddr(QQ) LT_Send(MSG)	→	
		Start loop UT_Receives(MSG) UT_Send_Ack(MSG) End loop
LT_Wait_For_Ack(MSG,TIME)		
LT_Check(SMSG,RMSG) End loop	←	End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.1.1.2 Receiving a Broadcast Message

Master receiving a broadcast message.

ZZ ∈ {broadcast address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must not send an ACK on a broadcast address. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.2		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.1.1.3 Receiving Messages with Different Data Length

Sub Test Case a
Master receiving messages with different data length from master.

NN ∈ {0,1,9,10}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.3a		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master receiving messages with different data length from master.

NN $\in \{NN_{max} - 1, NN_{max}, NN_{max} + 1\}$

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.3b		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.1.1.4 Receiving Messages Containing Reserved Bytes

Sub Test Case a
Master receiving messages with data or NN containing the reserved bytes.

Data ∈ {169,170}
NN ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.4.a		
LT_Send(MSG)	→	
LT_Wait_For_Ack(MSG,TIME)		
		UT_Receives(MSG) UT_Add(1.DB,2.DB) UT_Send_Ack(SUM)
LT_Check(SMSG,RMSG)	←	
		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master receiving messages with CRC consisting of a reserved byte.

CRC ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.1.4.b		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(SUM,TIME)		
LT_Check(SMSG,RMSG)	←	
		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.1.2 Error Detection in Message

The error detection includes all Test Cases, where the master is receiving an error in messages from a master. The Test Cases mainly use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

An error is detected when the CRC is not correct, which means that the receiver (IUT) has to send a NACK. When a sender receives the first NACK it has to resend but on the second it is not allowed to resend. These Test Cases check whether the IUT as master is able to detect a bad CRC and after that sending correct combinations of ACK-NACK.

The verification of an Elementary Test is made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

2.1.1.2.1 Master Receiving Error in CRC Byte Once

Master receiving a message with bad CRC from master.
Master sending the combination NACK-ACK.

CRC = {wrong CRC, right CRC}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send a NACK on a message containing bad CRC. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.2.1		
LT_Send(MSG,BAD CRC)		
LT_Wait_For_Nack(TIME)	→	
LT_Resend(MSG, CRC)		
		UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.1.2.2 Master Receiving Error in CRC Byte Twice

Master receiving a message with bad CRC from master twice.
Master sending the combination NACK-NACK and then have to recover.

CRC = {wrong, wrong, right]

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Three test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send NACK on message containing bad CRC. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.1.2.2		
LT_Send(MSG,BAD CRC) LT_Wait_For_Nack(TIME)		
	→	
LT_Resend(MSG,BAD CRC) LT_Wait_For_Nack(TIME)		
		UT_Receives(MSG) UT_Send_Ack(MSG)
LT_Send(MSG)		
LT_Wait_For_Ack(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.2 Sender Tests

The sender tests includes two Test Classes:

- Error free message handling
- Error handling of a message

For the sending tests the third Test Class, error detection in message, does not include any Test Cases, so this class is not needed.

Arbitration is also a sort of sending tests, therefore it is handled as an extra Test Class under sending tests.

2.1.2.1 Error Free Message Handling

The error free message handling includes all Test Cases, where the master is sending error free messages. The Test Cases use a single test frame for each Elementary Test, which means that Lower Tester sends just one test frame for each Elementary Test.

The verification of an Elementary Test is made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back.

2.1.2.1.1 Sending Messages with Different Addresses (ZZ)

Master sending messages to different target addresses.(ZZ)

ZZ ∈ {all master addresses}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	25 test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.1b		
Start loop Change_Sourceaddr(QQ) LT_Send(MSG)	→	
		Start loop UT_Receives(MSG) UT_Send_Ack(MSG) End loop
LT_Wait_For_Ack(MSG,TIME)		
LT_Check(SMSG,RMSG) End loop	←	End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.2.1.2 Sending a Broadcast Message

Master sending a broadcast message

ZZ ∈ {broadcast address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.2		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.2.1.3 Sending Messages with Different Data Length

Sub Test Case a
Master sending a message with different data length to a master.

NN ∈ {0,1,9,10}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.3.a		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master sending a message with different datalength to a master.

NN $\in \{NN_{max} - 1, NN_{max}, NN_{max} + 1\}$

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.3.b		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)		
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)	←	
Wait_For_End_Of_Test(OK)		

2.1.2.1.4 Handling of Reserved Bytes

Sub Test Case a
Master sending a message with data containing the reserved bytes.

Data ∈ {169,170}
NN ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.4.a		
LT_Send(MSG)		
	→	
		UT_Receives(MSG) UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master sending a message with CRC consisting of a reserved byte.

CRC ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.1.4b		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)		
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)	←	
Wait_For_End_Of_Test(OK)		

2.1.2.2 Error Handling of a Repeated Message

The error handling of a message includes all Test Cases, where the master is handling somehow after detecting error in a message. The Test Cases mainly use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

The verification of an Elementary Test is made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

Lower Tester sending NACK is defined as a test frame because it is sending a NACK without receiving a bad CRC. The reason for this is that it is hard for the Upper Tester to produce a bad CRC.

When a sender (IUT) receives the first NACK it has to resend but on the second it is not allowed to resend.

2.1.2.2.1 Master Resending after one NACK

Master receive a the combination NACK-ACK.
Master resending after one NACK and then receives an ACK.

ANSWER = {NACK, ACK}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT has to resend after receiving a NACK. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.1.1.3.1		
LT_Send(MSG)		
	→	
		UT_Receives(MSG)
LT_Wait_For_Msg(MSG,TIME)		
		UT_Send(MSG)
	←	
LT_Send_NACK()		
LT_Wait_For_Msg(MSG,TIME)		
	→	
		UT_Resend(MSG)
		End_Of_Test(OK)
	←	
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.2.2.2 Master not Resending after two NACK

Master receive the combination NACK-NACK
Master resending after first NACK but not after the second → has to recover.

Answer = {NACK, NACK}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Three test frames are used for this Elementary Test.
Verification	The IUT shall not generate any error flag during the test. The IUT must send an ACK on a correctly received frame. The IUT has to resend after receiving a NACK but not after the second. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR)		
		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.1.2.2.2		
LT_Send(MSG)		
	→	
LT_Wait_For_Msg(MSG,TIME)		UT_Receives(MSG)
		UT_Send(MSG)
	←	
LT_Send_NACK()		
	→	
LT_Wait_For_Msg(MSG,TIME)		
		UT_Resend(MSG)
	←	
LT_Send_NACK() LT_Send(MSG)		
	→	
LT_Wait_For_Msg(MSG,TIME)		
		UT_Send(MSG)
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.1.2.3 Arbitration

Arbitration is not really a sender test, it is a check for who is allowed to send its message on the bus. The arbitration is done by comparing the sent source addresses (QQ), that are existing on the bus under a specified time.

There are three different kinds of arbitration:

- 1 – cycle arbitration, IUT winning
- 1 – cycle arbitration, IUT loosing
- 2 – cycle arbitration, IUT loosing

The second cycle of the 2 – cycle arbitration, IUT winning is a repetition of the first cycle, so it is not necessary to have a separateTest Case.

The arbitration includes all Test Cases, where the master arbitrate for the right to send on the bus. Several test frames are used for each Test Case. Some test frames, which are just a complement to the Test Case, are not defined as test frames in the test organization table.

2.1.2.3.1 1 – Cycle Arbitration, IUT Winning

IUT as master winning arbitration in first cycle.

IUT QQ= {same priority, lower address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for the UT for this Elementary Test. Several test frames are used to arbitrate with UT.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.3.1		
LT_Send(MSG) // To UT		
LT_Send(MSG,REPEAT) // Sending on bus	→	
LT_Wait_For_Msg(MSG,TIME)		UT_Receives(MSG)
LT_Stop_Sending(MSG)		UT_Send_Msg(MSG) // Arbitration
	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.2.3.2 1 – Cycle Arbitration, IUT Loosing

IUT as master loosing arbitration in first cycle.

IUT QQ= {same priority, higher address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for the UT for this Elementary Test. Several test frames are used to block the bus for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.3.2		
LT_Send(MSG) // To UT		
LT_Send(MSG,REPEAT) // Blocking bus	→	
LT_Stop_Sending(MSG) LT_Send(MSG) // To UT		UT_Receives(MSG) UT_Send_Msg(MSG)
		UT_Send_Msg(MSG)
LT_Wait_For_Msg(MSG,TIME)	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.2.3.3 2 – Cycle Arbitration, IUT Loosing

IUT as master winning arbitration in second cycle.

IUT QQ = {Lower Priority, lower address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for the UT for this Elementary Test. Several test frames are used to block the bus for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.3.3		
LT_Send(MSG) // To UT		
LT_Send(MSG,REPEAT) // Blocking bus	→	UT_Receives(MSG)
		UT_Tries_To_Send_Msg(MSG)
LT_Stop_Sending(MSG) LT_Wait_For_Msg(MSG,TIME)		UT_Send_Msg(MSG)
	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.1.2.4 SYN Byte

2.1.2.4.1 Sending a SYN Byte

Sub Test Case a
Testing whether a SYN byte is sent at the end of a master->master message

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.4.1a		
LT_Send(MSG)		
	→	UT_Receives(MSG)
LT_Wait _For Msg(MSG, TIME)		
		End_Of_Test(OK)
LT_Check(SYN)	←	
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Testing whether a SYN byte is sent at the end of a master->slave message

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.1.2.4.1b		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Answer_On_Request(MSG,CRC)		
LT_Wait_For_Msg(MSG,TIME)	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2 Function Tests for Slave Message

The function tests for slave message include one test type, bi-directional tests, that has two sub test types:

- Bi-directional tests, IUT as slave
- Bi-directional tests, IUT as master

In a bi-directional Elementary Test there is always one receiving part and one sending part to be tested. Often this is done in same Elementary Test, but sometimes a Test Case must be split into several Test Cases to realize the test. This is done in following Test Cases:

- 2.2.1.1.4
- 2.2.2.1.3

These Test Cases consern the sending and receiving reserved bytes, and especially sending and receiving messages with CRC consisting of a reserved byte.

2.2.1 Bidirectional Tests – IUT as Slave

The bi-directional tests – IUT as slave includes three Test Classes:

- Error free message handling
- Error detection in message
- Error handling of a message

The repetition of the request, when receiving answer with (ACK+no data), is not defined as a test frame. In the second table the repetition of the request is used, but this only to be aware that the slave may be too slow to answer on the first request. The slave should answer directly, and if not, then the master has to resend the request within a undefined time.

2.2.1.1 Error Free Message Handling

The error free message handling includes all Test Cases, where the slave is receiving and answering error free messages. The Test Cases use a single test frame for each Elementary Test, which means that Lower Tester sends just one test frame for each Elementary Test.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

Both Test Case 2.2.1.1.4b and Test Case 2.2.1.1.4c, have the same test parameters. The difference is that in the first mentioned the test parameters are for the Lower Tester and in the second the test parameters are for the Upper Tester. In Test Case 2.2.1.1.4c the Lower Tester test parameters depend on the addresses of LT and UT. Therefore are the sent test parameters from LT not the mentioned in Test Case 2.2.1.1.4.

2.2.1.1.1 Slave Answering Messages with Different Addresses (QQ)

Slave receiving messages with different source addresses.(QQ)
Slave answering messages with different source addresses.

QQ = {all slave addresses}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	228 test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Request_For_Ready(FUNCNR)		
Wait_For_Ready_For_Test(FUNCNR,TIME)		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.1.1b		
Start loop Change_Sourceaddr(QQ) LT_Send_Request(MSG)	→	
LT_Send_Request_For_Ack(MSG)		Start loop UT_Receives(MSG) UT_Answer_Ack(MSG) End loop
LT_Wait_For_Ack(MSG,TIME)		
LT_Check(SMSG,RMSG) End loop	←	End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.2.1.1.2 Slave Receiving a Broadcast Message

Slave receiving a broadcast message.

ZZ ∈ {broadcast address}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must not send an ACK on a broadcast message. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Request_For_Ready(FUNCNR)		
Wait_For_Ready_For_Test(FUNCNR,TIME)		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.1.2		
LT_Send_Request(MSG)	→	
		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Send_Request_For_Ack(MSG)		
LT_Wait_For_Ack(MSG,TIME)		End_Of_Test(OK)
	←	
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.1.1.3 Slave Answering a Master Request with Different Data Length

Sub Test Case a
Slave receiving messages with different datalength from master.
Slave sending a message with different datalength to a master.

NN ∈ { 0,1,9,10}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Request_For_Ready(FUNCNR)		
Wait_For_Ready_For_Test(FUNCNR,TIME)		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.2.1.1.3.a		
LT_Send_Request(MSG)		
	→	
LT_Send_Request_For_Ack(MSG)		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Slave receiving messages with different datalength from master.
Slave sending a message with different datalength to a master.

NN $\in \{NN_{max} - 1, NN_{max}, NN_{max} + 1\}$

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
Request_For_Ready(FUNCNR) Wait_For_Ready_For_Test(FUNCNR,TIME)		Prepare_Test(FUNCNR)
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.1.3.b		
LT_Send_Request(MSG)		
	→	
LT_Send_Request_For Ack(MSG)		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Msg(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.1.1.4 Slave Answering a Master Request with Messages Containing Reserved Bytes

Sub Test Case a
Slave receiving messages containing reserved bytes from master.
Slave sending a message with data containing the reserved bytes to a master.

Data ∈ {169,170}
NN ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)	→	
		Prepare_Test(FUNCNR)
Request_For_Ready(FUNCNR)		
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.2.1.1.4.a		
LT_Send_Request(MSG)	→	
		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Send_Request_For_Ack(MSG)		
LT_Wait_For_Ack(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Slave receiving messages with CRC consisting of a reserved byte.

CRC ∈ { 169,170 }

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
Request_For_Ready(FUNCNR)		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.2.1.1.4.b		
LT_Send_Request(MSG)		
	→	
LT_Send_Request_For_Ack(MSG)		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

Sub Test Case c
Slave sending a message with CRC consisting of a reserved byte.

CRC ∈ { 169,170 }

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
Request_For_Ready(FUNCNR)		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
	←	
LT_Check(SFUNCNR,RFUNCNR)		
Test 2.2.1.1.4.c		
LT_Send_Request(MSG)		
	→	
LT_Send_Request_For_Ack(MSG)		
		UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.1.2 Error Detection in Slave Message

Error detection in slave message includes all Test Cases, where the slave is receiving and answering messages, after detecting errors in them. The Test Cases use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

As mentioned before, the repeating of a request is not defined as a test frame, while Lower Tester sending NACK is defined as a test frame.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

2.2.1.2.1 Slave Receives one NACK

Slave resending once.
Slave receives the combination NACK–ACK from master.

ANSWER = {NACK,ACK}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must resend after receiving a NACK. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
Request_For_Ready(FUNCNR)	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.2.1		
LT_Send_Request(MSG)		
LT_Send_Request_Again(MSG)	→	
LT_Send_NACK()		UT_Receives(MSG) Prepare_Answer(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.1.2.2 Slave Receives two NACK

Slave not resending twice.
Slave receives the combination NACK-NACK→recover(ACK) from master.

ANSWER = {NACK,NACK}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Four test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must resend after receiving a NACK. The IUT must send an ACK on a correctly received frame. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
Request_For_Ready(FUNCNR)		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.2.2		
LT_Send_Request(MSG) LT_Send_Request_Again(MSG) LT_Wait_For_Ack(MSG) LT_Send_NACK() LT_Wait_For_Ack(MSG) LT_Send_NACK()	→	UT_Receives(MSG) UT_Prepare(MSG) UT_Answer_Ack(MSG)
LT_Send_Request(MSG)		UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG)	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.1.3 Error Handling of a Resent Slave Message

Error handling of a resent slave message includes all Test Cases, where the slave is receiving and answering messages, after receiving a message including a bad CRC. The Test Cases use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

If same message is send first with a bad CRC and then with a correct CRC, then the test frames are defined to two, because it is two different messages. Otherwise, when the same message is resent, then it is not defined as a test frame.

2.2.1.3.1 Slave Answering with Combination NACK-(ACK+DATA)

Slave receiving a message containing a bad CRC from master.
Slave sending the combination NACK-(ACK+DATA) to a master.

CRC = {wrong,right}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send a NACK on a frame with bad CRC. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
Request_For_Ready(FUNCNR) Wait_For_Ready_For_Test(FUNCNR,TIME)		Prepare_Test(FUNCNR)
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.3.1		
LT_Send_Request(MSG, BAD CRC)		
	→	
LT_Wait_For_Nack(TIME) LT_Resend_Request(MSG)		
		UT_Receives(MSG,BAD CRC) UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

2.2.1.3.2 Slave Answering with Combination NACK-NACK

Slave receiving a message containing a bad CRC from master.
Slave sending the combination NACK-NACK→recover(ACK) to a master.

CRC = {wrong, wrong, right}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send a NACK on a frame with bad CRC. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.1.3.2		
LT_Send_Request(MSG,BAD CRC) LT_Wait_For_Nack(TIME) LT_Resend_Request(MSG,BAD CRC) LT_Wait_For_Nack(TIME) LT_Send_Request(MSG)	→	UT_Receives(MSG,BAD CRC) UT_Receives(MSG,BAD CRC) UT_Receives(MSG) UT_Answer_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)	←	
Wait_For_End_Of_Test(OK)		

2.2.2 Bidirectional Tests – IUT as Master

The bi-directional tests – IUT as master includes three Test Classes:

- Error free message handling
- Error detection in message
- Error handling of a message

2.2.2.1 Error Free Message Handling

The error free message handling includes all Test Cases, where the slave is receiving and answering error free messages. The Test Cases use two test frames for each Elementary Test, which means that Lower Tester sends one test frame as master and one test frame as a slaveanswer for each Elementary Test.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

2.2.2.1.1 Master Sending Messages with Different Addresses

Sub Test Case a
Master sending messages to different slave addresses.
Master receiving answers from different slave addresses.

ZZ ∈ {all slave addresses}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	228 test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.1a		
Start loop LT_Send(MSG) Change_Sourceaddr()		
	→	
		Start loop Change_Targetaddr() UT_Receives(MSG) UT_Send_Ack(MSG) End loop
LT_Answer_On_Request(MSG)		
LT_Wait_For_Ack(MSG,TIME)		
		End_Of_Test(OK)
	←	
LT_Check(SMSG,RMSG) End loop Wait_For_End_Of_Test(OK)		

2.2.2.1.2 Master Sending Slave Messages with Different Data Length

Sub Test Case a
Master sending messages with different data length to a slave.
Master receiving messages with differrent data length from slave.

NN ∈ {0,1,9,10}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.2.a		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Answer_On_Request(MSG) LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master sending a message with different data length to a slave.
Master receiving a message with different data length from slave.

NN $\in \{NN_{max} - 1, NN_{max}, NN_{max} + 1\}$

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.2.b		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Answer_On_Request(MSG,CRC)		
LT_Wait_For_Msg(MSG,TIME)	←	
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.2.1.3 Handling of Reserved Bytes in Slave Message

Sub Test Case a
Master sending a message with data containing the reserved bytes to slave.
Master receiving messages containing reserved bytes from slave.

Data ∈ {169,170}
NN ∈ {169,170}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.3.a		
LT_Send(MSG)	→	
		UT_Receives(MSG) UT_Send_To_Slave(MSG)
LT_Answer_On_Request(MSG)		UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

Sub Test Case b
Master sending a message with CRC consisting of a reserved byte to a slave.

CRC ∈ { 169,170 }

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.3b		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Answer_On_Request(MSG) LT_Wait_For_Ack(MSG,TIME)		
	←	
LT_Check(SMSG,RMSG)		End_Of_Test(OK)
Wait_For_End_Of_Test(OK)		

Sub Test Case c
Master receiving messages containing reserved bytes from slave.

CRC ∈ { 169,170 }

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.1.3c		
LT_Send(MSG)		
	→	UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Answer_On_Request(MSG) LT_Wait_For_Ack(MSG,TIME)		
		End_Of_Test(OK)
	←	
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.2.2 Error Handling in Message

Error handling in message includes all Test Cases, where the master (IUT) is receiving NACK answers from a slave. The Test Cases use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

2.2.2.2.1 Master Receiving the Combination NACK-(ACK+DATA)

Master receiving NACK–(ACK+DATA) from slave.
Master resend message once.

ANSWER = {NACK,ACK+DATA}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Two test frames are used in this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must resend after receiving a NACK. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.2.1		
LT_Send(MSG)		
	→	
LT_Send_NACK()		UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.2.2.2 Master Receiving the Combination NACK-NACK→ Recover

Master receiving NACK-NACK→recover.
Master not resending message twice.

ANSWER = {NACK,NACK}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Four test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must resend after receiving a NACK. The IUT must send an ACK on a correctly received frame. The IUT must verify a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.2.2		
LT_Send(MSG)		
LT_Send_NACK() LT_Send_NACK()	→	
		UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Send(MSG)		
LT_Wait_For_Ack(MSG,TIME)		
		End_Of_Test(OK)
LT_Check(SMSG,RMSG)	←	
Wait_For_End_Of_Test(OK)		

2.2.2.3 Error Detection of a Repeated Message

Error detection in message includes all Test Cases, where the master (IUT) is receiving bad CRCs in a answer from a slave. The following Test Cases checks whether the IUT handles the different combinations of ACK-NACK correctly. The Test Cases use two or more test frames for each Elementary Test, which means that Lower Tester sends several test frames for each Elementary Test.

If same message is send first with a bad CRC and then with a correct CRC, then the test frames are defined to two, because it is two different messages. Otherwise, when the same message is resent, then it is not defined as a test frame.

The verification of an Elementary Test is nearly always made with the mirroring function. The mirroring function receives a message, whereafter it sends the same message back as a verification.

2.2.2.3.1 Master Sending one NACK to Slave

Master receiving a message containing a bad CRC from a slave.
Master sending the combination NACK-ACK to a slave

CRC = {wrong CRC, right CRC}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Three test frames are used for each Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send a NACK on a frame with bad CRC. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.3.1		
LT_Send(MSG)		
	→	
LT_Answer_On_Request(MSG,BAD CRC) LT_Answer_On_Request(MSG)		UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)		
	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

2.2.2.3.2 Master Sending two NACK to Slave

Master receiving two bad CRC bytes from slave.
Master sending the combination NACK-NACK to a slave →recover (ACK+DATA)

CRC ∈ {wrong CRC, wrong CRC}

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	Three test frames are used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The IUT must send a NACK on a frame with bad CRC. The IUT must send back a received frame. The data received from IUT must match the data sent in the test frame.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 2.2.2.3.2		
LT_Send(MSG)		
	→	
LT_Answer_On_Request(MSG,BAD CRC) LT_Answer_On_Request(MSG,BAD CRC) LT_Send(MSG)		UT_Receives(MSG) UT_Send_To_Slave(MSG) UT_Send_Ack(MSG)
LT_Wait_For_Ack(MSG,TIME)	←	End_Of_Test(OK)
LT_Check(SMSG,RMSG)		
Wait_For_End_Of_Test(OK)		

3 Timing Tests

3.1 Timing Tests for the Master Message

3.1.1 Time Constraints for the ACK Byte

The time between the CRC byte from master to the ACK byte from another master.

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The time between CRC from LT and ACK from IUT has to be less than T_ASYN_{min}.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	
		Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 3.1.1		
LT_Send(MSG)		
LT_Start_Timer(CRC)	→	
LT_Stop_Timer(ACK)		
		UT_Receives(MSG)
LT_Wait(TIME)		
	←	
		End_Of_Test(OK)
LT_Check(TCRC,TACK)		
Wait_For_End_Of_Test(OK)		

3.2 Timing Tests for the Slave Message

3.2.1 Time Constraints for the ACK-Byte

The time between the CRC byte from master to the ACK byte from a slave.

Test Case Organization:

State	Description
Set Up State	Idle State.
Test State	A single test frame is used for this Elementary Test.
Verification	The IUT shall not generate any errors during the test. The IUT must send an ACK on a correctly received frame. The time between CRC from LT and ACK from IUT has to be less than T_ASYN_{min}.
Final State	Idle State.

Lower Tester	Flow	Upper Tester
Start_Of_Test(FUNCNR)		
	→	Prepare_Test(FUNCNR)
Wait_For_Ready_For_Test(FUNCNR,TIME)		
		Ready_For_Test(FUNCNR)
LT_Check(SFUNCNR,RFUNCNR)	←	
Test 3.2.1		
LT_Send(MSG)		
LT_Start_Timer(CRC)	→	
LT_Stop_Timer(ACK)		
		UT_Receives(MSG)
	←	End_Of_Test(OK)
LT_Check(TCRC,TACK)		
Wait_For_End_Of_Test(OK)		

4 Appendix, Enclosures and References

4.1 Appendix

4.1.1 Parameters (according to eBUS Specification rev 1.0)

NAME	VALUE	DESCRIPTION	SIDE
T_ASYN_{min}	30 ms	The minimum value of the time for the AUTO-SYN cycle	14
NN_{max}	240	The maximum number of data bytes	1

4.1.2 Set Up States

Name	Description
1. Idle State	• No traffic on the bus. • No data in receive and send buffer. • IUT ready to receive. • IUT not willing to send.

4.1.3 Abbreviations

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

ACK	Acknowledgement
ARB	Arbitration
CRC	Cyclic Redundancy Check
EOFSYN	End Of SYN
FUNCNR	Function Number
IUT	Implementation Under Test
LT	Lower Tester
MSG	Message
MSGNR	Message Number
NACK	Non Acknowledgement
OK	Okay
RFUNCNR	Received Function Number
RMSG	Received Message
SFUNCNR	Sent Function Number
SMSG	Sent Message
SOFQQ	Start Of QQ
SOFSYN	Start Of SYN
SYN	Syncronisation
UT	Upper Tester

4.1.4 Parameters Used in Second Table

CRC:
The last byte sent by LT.

FUNCNR:
The first data byte from LT, sent in Set Up State, that indicates to UT which function(s) will be used.

MSG:
The test frame, i.e. whole message sent from LT.

MSGNR:
In the Test State the first data byte in each message is reserved to contain the number of the Elementary Test in a Test Case. This byte is used for the verification in LT.

OK:
A data byte used by UT to indicate to LT that the test has passed on the Upper Tester side.

RCRC:
The LT receives the sent CRC from UT as a data byte. This is used for the verification.

RFUNCNR:
This is the FUNCNR received by LT from UT. It is used for the verification.

RMSG:
This is the MSG received by LT from UT. It is used for the verification.

SCRC:
This is the sent CRC byte from LT. It is used for the verification.

SFUNCNR:
This is the FUNCNR sent from LT to UT. It is used for the verification.

SMSG:
This is the MSG sent from LT to UT. It is used for the verification.

4.1.5 Glossary

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

Conformance Testing:
Applying the Test Plan to an IUT.

Elementary Test:
A single test from a Test Case.

Final State:
The state the IUT must be in after each Elementary Test.

Idle State:
A specified Set Up State for the IUT. Described in § 5.2.

Implementation Under Test:
The device that will be tested.

Lower Tester:
One participant of the Test Plan. Described in § 1.1.

Set Up State:
The state the IUT must be in before entering the Test State.

Sub Test Case:
Sometimes a Test Case have to be split into two or more Sub Test Cases.

Test Case:
Described in § 1.3.3 and 1.4.

Test Class:
Described in § 1.3.2.

Test Plan:
A documentation that includes the description and realization for a Conformance Test according to the OSI Standard.

Test Type:
Described in § 1.3.1.

Upper Tester:
One participant of the Test Plan. Described in § 1.1.

Verification State:
A state where the Lower Tester verifies the messages.

4.2 References

- eBUS Specification layer 1,2 ver 1.0
- OSI Conformance Testing Methodology and TTCN
  Bernd Baumgarten Alfred Giessler
  ISBN: 0 444 897127
- CAN CONFORMANCE TESTING 4.3.1998

5 Änderungsliste

Version	Datum	Bemerkungen	
1.1	1998	Initial Version	Frank Hoffmann
1.1.1	03.2007	Änderung der eBUS User Club Logos nach eBUS Interest Group	Frank Hoffmann