System Conformance Testing

8

Test Suite Supplement H

H

KNXnet/IP Testing

Summary

This document contains test specifications for conformity testing of KNXnet/IP compliant devices.

Version 01.03.01 is a KNX Approved Standard.

This document is part of the KNX Specifications v2.1.

Document Updates

Version	Date	Modifications
0.1	31.05.2005	Document creation
0.2	24.04.2007	Update of document following commenting from KNX Certification Department and reaction of ise
0.3	9.10.2007	Update document with additional test cases
0.4	27.11.2008	Total review of the document as regards test conditions and test results
0.5	19.02.2009	Resolution of comments from KNX IP TF - Preparation for Release for Voting
0.6	29.04.2009	Resolution of comments from RfV - Preparation for Final Voting
1.0	June 2009	Resolution of comments from FV – Readying document for publication in V2.0 of the KNX specifications
1.1WD	August 2011	Adding clause 7 remote diagnostics and configuration
1.1WD2	September 2012	Integration of comments from KonCert meeting
1.2WD	16.11.12	Added clause 4.3 cEMI Transport Layer – added 5.3.4 to 5.3.6 – not yet included routing busy and knx/ip performance (still under discussion in TF IP/implementation in Validation Tool).
1.2DP	01.02.13	Last minute changes in KonCert Telco 0113 – preparation for RfV
1.2DV	04.04.13	Resolution of comments from release for voting – reading for FV
1.3AS	2013.09	No comments in Final voting – publication as Approved Standard
01.03.01	2013.10.24	Editorial updates for the publication of KNX Specifications 2.1.

Filename: 08_TSSH System Conformace Testing - KNXnet-IP Tests v01.03.01 AS.docx
Version: 01.03.01
Status: Approved Standard
Savedate: 2013.10.24
Number of pages: 153

Contents

1 Test Set-up

1.1 Requirements

The following hard- and software is required for the automated KNX/IP device tests.

1. KNX/IP Test device, BDUT
2. PC with as required installed components: KNXnet/IP validation tool[1], .NET Framework 1.1, Falcon Runtime/Developer Version 1.23 (File version 1.0.2177.0, latest Falcon release) or higher.
3. 10/100 MBit HUB/Switch
4. KNX Installation:
   - KNX DIN rail (in case of DIN rail mounting of BDUT)
   - RS232 , EIBlib/IP, KNXnet/IP tunnelling server or USB access
   - Power supply
   - KNX load switch
5. Ethernet TP and KNX TP1 cables
6. DHCP-Server for automatic assignment of IP addresses.
7. Optional: PC with Wireshark or equivalent with KNX/IP extensions. This PC can be used to monitor KNXnet/IP traffic, but is not required for the tests.

1.2 Layout

1.2.1 Overview

The BDUT shall be connected to the bus and - if required - to an external power supply. It shall also be connected to the Ethernet installation. A KNX load switch can be used to interrupt bus connection of the BDUT.

The Ethernet installation shall not be connected to other network segments to avoid interference with Ethernet traffic created by other devices than those required for the tests. It is therefore recommended to use a non-switching HUB, if available (especially, if PC running Wireshark or equivalent is connected to the Ethernet). This is however not a requirement for the tests to succeed.

See Figure 1: KNX/IP Validation - Test Layout for a topological view of the test layout.

1.2.2 Load switch

The KNX load switch shall have two channels allowing the complete disconnection of the BDUT from the KNX installation, this means disconnecting both wires of the TP1 cable in use. Both channels shall use the same Group Address(1 bit) for switching on (value = 1) and off (value = 0): 1/1/50. The Individual Address of the load switch shall be set to 1.1.50.

If the BDUT supports KNXnet/IP tunnelling, the load switch will also be used during one of the tests (point to point communication over a tunnelling connection). It is therefore important that the Individual Address of the load switch is set to the above-mentioned value.

1 Available free of charge from KNX Association for every EITT licensee

1.3 KNX access of test application

The KNXnet/IP validation tool can be configured to access KNX in the following ways:

1. via serial connection using the PEI16/PEI10 protocol and an RS232
2. via USB connection using the USB protocol and a USB-interface
3. via an EIBlib/IP connection using the EIBlib/IP protocol and an arbitrary (this includes a PC running EIBlib/IP server software) EIBlib/IP server.
4. via KNXnet/IP Tunneling .

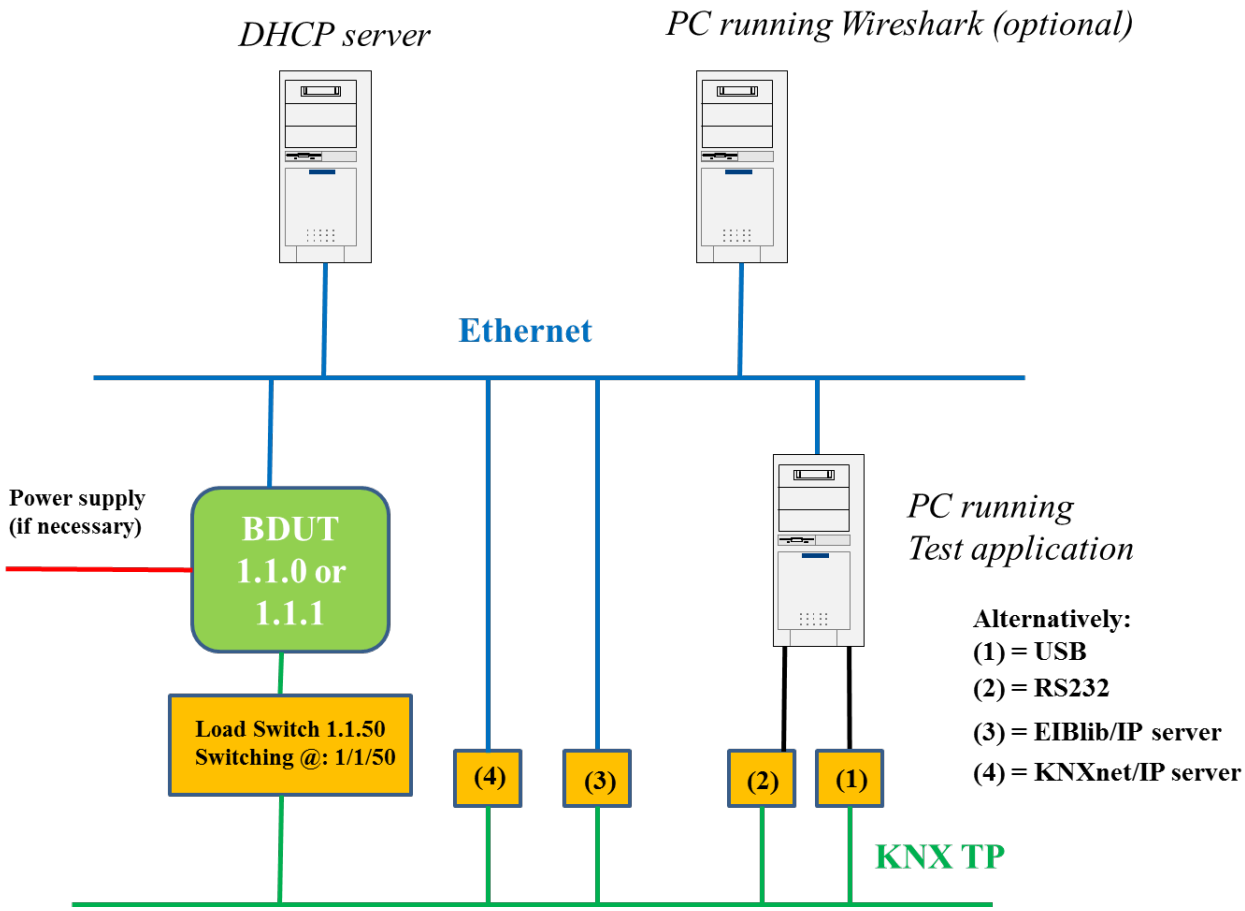

Figure 1: KNX/IP Validation - Test Layout

1.4 Test prerequisites

1. The BDUT shall have the following Individual Address:
   - 1.1.0 if it does support routing,
   - 1.1.1 otherwise
2. The KNX load switch shall have the Individual Address 1.1.50
3. The KNX load switch shall be in the “on” position, that is, the BDUT shall be connected to the KNX installation

The bus interface shall have the Individual Address 1.1.2.

1.5 Important remarks to test case “Hard Reset”

Unlike all other test cases, test case “7.5.9 Hard Reset” is special, and should be executed separately from all other tests.

After a Hard Reset (which forces the BDUT into factory configuration), the test client cannot satisfactorily automatically restore the old BDUT configuration:

- It can happen that after Hard Reset, the Test Client has no longer access to the BDUT due to IP address conflicts (particularly if there is no DHCP server connected to the BDUT). In this case, the Test Client reports necessarily “Fatal Error”, although paradoxically the Hard Reset has succeeded.
- Even when the Test Client regains access to the BDUT after Hard Reset, it cannot completely restore its old configuration. In particular, it cannot restore IP-router filtering options (determining which telegrams are routed or not), so that certain Routing tests which previously succeeded may fail after executing the Hard Reset test:

In order to restore the old BDUT configuration after the Hard Reset test, the BDUT must be manually configured, typically by ETS download via KNX bus.

2 Definitions

2.1 Variables

X: Don't care

2.1.1 Depending on BDUT

IP_BDUT: IP-address of BDUT

CPORT_BDUT: control port of BDUT

DPORT_BDUT: data port of BDUT

KNXADDR_BDUT: Individual Address of BDUT

MAC_BDUT: MAC address of BDUT

IP_ROUTING: IP routing multicast address

2.1.2 Depending on Test client

IP_TC: IP-address of Test Client

CPORT_TC: control port of Test Client

DPORT_TC: data port of Test Client

KNXADDR_TC: Individual Address of Test Client

MAC_TC: MAC address of Test Client

2.2 Frames

2.2.1 Standard UDP-Frame

Destination MAC (6)			Source MAC (6)	
... Source MAC (6)		Ethertype (2)	V/IHL(1)	TOS(1)
Total IP Length (2)	Identification (2)	Flags / Fragment Offset (2)	TTL (1)	Protocol (1)
IP-Header checksum (2)	Source IP Address (4)		Destination IP Address (4)	
… Destination IP Address (4)	Source UDP Port (2)	Destination UDP Port (2)	UDP Length (2)	
Correct UDP checksum (2)				

The size of a field in bytes is written in brackets.

2.2.1.1 Representation in Test Sequences:

UDP-frame (Source IP Address, Source UDP Port, Destination IP Address, Destination UDP Port)
Data transported over the UDP-frame

Example:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06	0x10	0x207	0x0010	CID	0x00
0x08	0x01	IP_TC		CPORT_TC	

2.2.1.2 Fields defined in IEEE 802.3

Destination MAC: Must be the Ethernet MAC address of the receiver (compare RFC 826) or a Multicast MAC (RFC 1112 – 6.4) (no criterion for acceptance)

Source MAC: Must be the Ethernet MAC address of the sender (compare RFC 826) (no criterion for acceptance)

Ethertype: is always 0x800 (Internet Protocol, Version 4)

2.2.1.3 Fields defined in RFC 791

V/IHL: is always 0x4L (0x4 = Version, L = IP Header Length / 4)

TOS: should be set to 0 (no criterion for acceptance)

Total IP Length: see RFC

Identification: see RFC (no criterion for acceptance)

Flags / Fragment Offset: should be set to 0x4000, as devices are not required to support fragmentation (no criterion for acceptance)

TTL: see RFC (no criterion for acceptance)

Protocol: should be 0x11 (UDP)

IP-Header checksum: see RFC.

Source IP Address: The IP Address of the sender or a multicast IP Address. It is specified as parameter of the UDP frame.

Destination IP Address: The IP Address of the receiver or a multicast IP Address. It is specified as parameter of the UDP frame.

Optional options and padding are ignored and no criteria for acceptance.

2.2.1.4 Fields defined in RFC 768

Source UDP Port: The UDP Port of the sender. It is specified as parameter of the UDP frame.

Destination UDP Port: The UDP Port of the receiver. It is specified as parameter of the UDP frame.

UDP Length: see RFC

Correct UDP checksum: see RFC

2.2.2 Standard ICMP-Frame

<table>
<tr><td colspan="4">Destination MAC (6)</td><td colspan="2">Source MAC (6)</td></tr>
<tr><td colspan="3">... Source MAC (6)</td><td>Ethertype (2)</td><td>V/IHL(1)</td><td>TOS(1)</td></tr>
<tr><td>Total IP Length (2)</td><td colspan="2">Identification (2)</td><td>Flags / Fragment Offset (2)</td><td>TTL (1)</td><td>Protocol (1)</td></tr>
<tr><td>IP-Header checksum (2)</td><td colspan="3">Source IP Address (4)</td><td colspan="2">Destination IP Address (4)</td></tr>
<tr><td>… Destination IP Address (4)</td><td>ICMP Type (1)</td><td>ICMP Code (1)</td><td>ICMP Checksum (2)</td><td colspan="2">ICMP type depended header (4)</td></tr>
<tr><td>… ICMP type depended header (4)</td><td colspan="5"></td></tr>
</table>

Size of a field in bytes is written in brackets.

2.2.2.1 Representation in Test Sequences:

`ICMP-frame (Source IP Address, Destination IP Address, ICMP Type, ICMP Code [,ICMP type depended header])`
Data transported over the ICMP-frame

Example:

`ICMP-frame (Source IP Address, Destination IP Address, 3 (Destination Unreachable), 3 (Port unreachable))`
Internet Header + 64 bits of Original Data Datagram (don't care)

2.2.2.2 Fields defined in IEEE 802.3

Destination MAC: Must be the Ethernet MAC address of the receiver (compare RFC 826) or a Multicast MAC (RFC 1112 – 6.4) (no criterion for acceptance)

Source MAC: Must be the Ethernet MAC address of the sender (compare RFC 826) (no criterion for acceptance)

Ethertype: is always 0x800 (Internet Protocol, Version 4)

2.2.2.3 Fields define in RFC 791

V/IHL: is always 0x4L (0x4 = Version, L = IP Header Length / 4)

TOS: should be set to 0 (no criteria for acceptance)

Total IP Length: see RFC

Identification: see RFC (no criterion for acceptance)

Flags / Fragment Offset: should be set to 0x4000, as devices are not required to support fragmentation (no criteria for acceptance)

TTL: see RFC (no criterion for acceptance)

Protocol: should be 0x01 (ICMP)

IP-Header checksum: see RFC.

Source IP Address: The IP Address of the sender or a multicast IP Address. It is specified as parameter of the UDP frame.

Destination IP Address: The IP Address of the receiver or a multicast IP Address. It is specified as parameter of the UDP frame.

Optional options and padding are ignored and no criteria for acceptance.

2.2.2.4 Fields defined in RFC 792

ICMP Type: The type of the ICMP Message. It is specified as parameter of the ICMP frame.

ICMP Code: The Code of the ICMP Message. It is specified as parameter of the ICMP frame.

ICMP Checksum: The correct checksum. See RFC.

ICMP type depended header: Should be set to 0x0. It is specified if needed.

3 Core

3.1 Unspecific

3.1.1 Undefined Discovery Code

Function ID: 10101

Description: This test repeatedly sends random undefined service codes to the discovery endpoint of the test device.

Expectation: The test device shall ignore these services and not send any answers.

Parameters: Repeats (Type = Integer, Default = 3)

Preparation: None

Test sequence:

TC sends undefined service:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	Service code (2)	0x000E (Total Length)	0x08 (Struct Length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

`Service code` shall be set to some undefined values, e.g. 0x4910, 0xb5f1, 0x22d3.

Cleanup: None

3.1.2 Undefined Control Code

Function ID: 10204

[RNA: Added illegal messages to data endpoint, fixed and modified implementation.]

Description: This test opens a device management connection and then repeatedly sends random undefined service codes to the control endpoint and data endpoint of the test device. Between these illegal messages, it sends legal device management requests to the test device.

Expectation: The test device should ignore all illegal messages and not send any answers, whereas it should correctly respond to the interspersed legal request messages by sending back corresponding response messages.

Parameters: Repeats (Type = Integer, Default = 3), Service Code= 0xabab

Preparation:

Open connection (see 3.4.1)

Used results: Channel ID (CID)

Test sequence:

Send several undefined services amongst several Device Configuration Request / Device Configuration Response services.

TC sends undefined service:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	Service code (2)	0x0010 (Total Length)	CID	0x00 (res.)
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC	

Service code shall be set to some undefined values, e.g. 0xabab.

TC sends DEVICE_CONFIGURATION_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_ CONFIGURATION_ REQUEST)	0x0011 (Total Length)	0x04 (struct. length)	CID
Sequence counter	0x00 (res.)	0xFC (message code)	cEMI frame (6)		

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0311 (DEVICE_ CONFIGURATION_ ACK)	0x000A (Total Length)	0x04 (struct. length)	CID
Sequence counter	0x00 (Status)				

BDUT sends DEVICE_CONFIGURATION_REQUEST:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_ CONFIGURATION_ REQUEST)	0x0012 (Total Length)	0x04 (struct. length)	CID
Sequence counter	0x00 (res.)	0xFB	cEMI frame (7)		

TC sends DEVICE_CONFIGURATION_ACK:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0311 (DEVICE_ CONFIGURATION_ ACK)	0x000A (Total Length)	0x04 (struct. length)	CID
Sequence counter	0x00 (Status)				

Cleanup:

Close Connection

3.2 Search Request

3.2.1 Standard Case

Function ID: 10201

Description This test verifies the standard behaviour by sending valid search request message to the discovery endpoint.

Expectation: The test device must send a search response message to the requested answer address including its own control endpoint address information, a valid device description block and a list of supported service families.

Parameters: None

Out Parameters: None

Preparation: None

Test sequence:

TC sends SEARCH_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0201 (SEARCH_REQUEST)	0x000E (Total length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC		

BDUT sends SEARCH_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0202 (SEARCH_ RESPONSE)	Total Length (1)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x36 (Struct. length)	0x01 (descr type: DEVICE_ INFO)
KNX medium (1)	Device status (1)	Individual address (2)	Project ID (2)	Serial number (6)	
			Multicast address (4)		
MAC address (6)				Device friendly name (30d)	
...					

Struct. length (1)	0x02 (descr type: SUPP_SVC _FAMIL)	Service family ID (1)	Service family version (1)	...	...

Cleanup: None

3.2.2 Invalid Version

Function ID: 10202

Description: This test sends a syntactically correct search request, but the header contains a version field other than 1.0.

Expectation: The test device shall ignore this invalid message and not send a search response.

Parameters: Version (Type = Byte, Default = 0x11)

Out Parameters: None

Preparation: None

Test sequence:

TC sends SEARCH_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	Version (1)	0x0201 (SEARCH_REQUEST)	0x000E (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

Cleanup: None

3.2.3 Invalid Header Size

Function ID: 10203

Description: This test sends a syntactically correct search request, but the header size is other than 6 bytes.

Expectation: The test device should ignore this invalid message and not send a search response.

Parameters: Header Size (Type = Byte, Default = 0x01)

Out Parameters: None

Preparation: None

Resulting telegrams:

TC sends SEARCH_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
Header length (1)	0x10 (Ver.)	0x0201 (SEARCH_REQUEST)	0x000E (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

Cleanup: None

3.2.4 Incomplete Message

Function ID: 10205

Description: This test sends a syntactically correct search request message, but the total size field in the header indicates a longer message.

[RNA: Replaced “shorter” by “longer” in text above, and changed implementation accordingly, to conform to title and other texts]

Expectation: The test device should ignore this invalid message and not send a search response.

[RNA: Violation of this rule is reported as error.]

Parameters: Bytes Missing (Type = Integer, Default = 1)

Preparation: None

Resulting telegrams:

TC sends SEARCH_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0201 (SEARCH_REQUEST)	0x000E+bytes missing (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

Cleanup: None

3.2.5 Oversized Message

Function ID: 10206

Description: This test sends a syntactically correct search request message, but the total size field in the header indicates a shorter message.

[RNA: Replaced “larger” by “shorter” in text above, and changed implementation accordingly, to conform with title and other texts]

Expectation: The test device should ignore this oversized message and not send a search response.

[RNA: Violation of this rule is reported as error.]

Parameters: Padding Bytes (Type = Integer, Default = 1)

Preparation: None

Resulting telegrams:

TC sends SEARCH_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0201 (SEARCH_REQUEST)	0x000E-padding bytes (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

Cleanup: None

3.3 Description Request

3.3.1 Standard Case

Function ID: 10301

Description: This test verifies the standard behaviour by sending valid description request message to the control endpoint that the test device returned in response to a previously sent search request.

Expectation: The test device must send a description response message to the requested answer address including a valid device description block, the list of supported service families and an optional manufacturer specific data block.

Parameters: None.

Preparation: None

Resulting telegrams:

TC sends DESCRIPTION_REQUEST

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0203 (DESCRIPTION_REQUEST)	0x000E (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC		

BDUT sends DESCRIPTION_RESPONSE

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0204 (DESCRIPTION_ RESPONSE)	Total Length (1)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x36 (Struct. length)	0x01 (descr type: DEVICE_ INFO)
KNX medium (1)	Device status (1)	Individual address (2)	Project ID (2)	Serial number (6)	
			Multicast address (4)		
MAC address (6)				Device friendly name (30d)	
...					

Length (1)	0x02 (descr type: SUPP_SVC_ FAMILIES)	Service family ID (1)	Service family version (1)	...	...

Cleanup: None

3.4 Connect Request

3.4.1 Standard Case

Function ID: 10401

Description: This test verifies the standard behaviour by sending a valid connect request message for an arbitrary mandatory connection type.

Expectation: The test device must send a connect response message to the requested answer address indicating E_NO_ERROR and containing a currently unused communication channel ID.

Parameters: None.

Preparation: None

Resulting telegrams:

TC sends CONNECT_REQUEST

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x0018 (Total Length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			DPORT_TC	CRI (2)	

BDUT sends CONNECT_RESPONSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0012 (Total length)	CID (1)	0x00 Status= E_NO_ERROR
0x08 (Struct length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC	
CRD (2)					

Cleanup: Close connection (see 3.6.1)

3.4.2 Invalid Connection Type

Function ID: 10402

Description: This test sends a connect request message for an undefined connection type.

Expectation: The test device must send a connect response message to the requested answer address indicating E_CONNECTION_TYPE.

Parameters: Connection Type (Type = Byte, Default = 0x42)

Preparation: None

Resulting telegrams:

TC sends CONNECT_REQUEST

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			FPORT_TC	0x04 length	Connection Type
0xFF00 (data)					

BDUT sends CONNECT_RESPONSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total Length)	0x00 (CID)	0x22 E_CONNECTION_TYPE

Cleanup: None

3.5 Connectionstate Request

3.5.1 Standard Case

Function ID: 10501

Description: This test verifies the standard behaviour by sending a valid connection state request message for an active communication channel ID.

Expectation: The test device must send a connection state response message to the requested answer address indicating E_NO_ERROR and containing the communication channel ID from the request.

Parameters: None.

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID)

Test sequence:

TC sends CONNECTIONSTATE_REQUEST**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x207 (CONNECTIONSTATE_ REQUEST)	0x0010 (Total length)	CID	0x00 (reserv ed)
0x08 (Struct Length)	0x01 (IPV4_U DP)	IP_TC		CPORT_TC	

BDUT sends CONNECTIONSTATE_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x208 (CONNECTIONSTATE_ RESPONSE)	0x0008 (Total length)	CID	0x00 (E_NO_E RROR)

Cleanup:

Close Connection (see 3.6.1)

3.5.2 Invalid Channel

Function ID: 10502

Description: This test sends a connection state request message for a currently invalid communication channel ID.

Expectation: The test device must send a connection state response message to the requested answer address indicating E_CONNECTION_ID.

Parameters: ID Offset (Type = Byte, Default = 1)

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID)

Test sequence:

TC sends **CONNECTIONSTATE_REQUEST**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x207 (CONNECTIONSTATE _REQUEST)	0x0010 (Total length)	CID +ID Offset	0x00 (res.)
0x08 (Struct Length)	0x01 (IPV4_U DP)	IP_TC		CPORT_TC	

BDUT sends **CONNECTIONSTATE_RESPONSE**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x208 (CONNECTIONSTATE_RESPONSE)	0x0008 (Total length)	CID + ID Offset	0x21 (E_CONNECTION_ID)

Cleanup:

Close Connection (see 3.6.1)

3.5.3 Time Out

Function ID: 10802

Description: This test verifies the standard behaviour by letting a device management connection time out by not sending connection state requests.

Expectation: The test device must send a disconnect request message to the local communication endpoint after its internal time out period. This period is defined as 120 seconds; nevertheless, a range of 110...130 seconds is accepted as success.

Parameters: None

Out Parameters: Measured time out value in seconds

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID),

Test sequence:

Test Application begins measurement of time immediately after receiving the correct connect response.

Expected Answer 120 s after begin of measurement. (Tolerance ± 10 s)

TC sends DISCONNECT_REQUEST

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x209 (DISCONNECT_REQUEST)	0x00010 (Total length)	CID Offset	0x00 (reserved)
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_BDUT		CPORT_BDUT	

Cleanup: none

3.5.4 Bus connection interrupted

Function ID: 10503

Description: This test interrupts the bus connection by means of a load switch and checks the response of a subsequent connection state request.

Expectation: The returned status code should be E_KNX_CONNECTION.

Parameters: Group address for load switch (Type = String, Default = 1/1/50)

Out Parameters: None

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID),

Interrupt bus connection

Resulting telegrams:

TC sends CONNECTIONSTATE_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x207 (CONNECTIONSTATE_REQUEST)	0x0010 (Total length)	CID	0x00 (reserved)
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		CPORT_TC	

BDUT sends CONNECTIONSTATE_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x208 (CONNECTIONSTATE_RESPONSE)	0x0008 (Total length)	CID	0x27 (E_KNX_CONNECTION)

Cleanup:

Re-establish bus connection

3.6 Disconnect Request

3.6.1 Standard Case

Function ID: 10601

Description: This test verifies the standard behaviour by sending a valid disconnect request message for an active communication channel ID.

Expectation: The test device must send a disconnect response message to the requested answer address indicating E_NO_ERROR and containing the communication channel ID from the request.

Parameters: None.

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID),

Resulting telegrams:

TC sends **DISCONNECT_REQUEST**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x209 (DISCONNECT_REQUEST)	0x0010 (Total length)	CID	0x00 (reserved)
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		CPORT_TC	

BDUT sends **DISCONNECT_RESPONSE**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x20A (DISCONNECT_RESPONSE)	0x0008 (Total length)	CID	0x00

Cleanup: none

3.6.2 Invalid Channel

Function ID: 10602

Description: This test sends a disconnect request message for a currently invalid communication channel ID.

Expectation: The test device must send a disconnect response message to the requested answer address indicating E_CONNECTION_ID.

Parameters: ID Offset (Type = Byte, Default = 1)

Preparation:

Open connection (see 3.4.1):

Used results: Channel ID (CID),

Resulting telegrams:

TC sends **DISCONNECT_REQUEST**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x209 (DISCONNECT_REQUEST)	0x0010 (Total length)	CID+ ID offset	0x00 (reserved)
0x08 (Struct Length)	0x01 (IPV4_UDP)	IP_TC		CPORT_TC	

BDUT sends **DISCONNECT_RESPONSE**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x20A (DISCONNECT_RESPONSE)	0x0008 (Total length)	CID+ID offset	0x21 E_CONNECTION_ID

Cleanup: Close connection (see 3.6.1)

4 Device Management

4.1 Connection Handling

4.1.1 Multiplicity

Description: This test verifies the standard behaviour by sending valid connect request message for a device management connection until the device cannot take any more connections.

Expectation: When the maximum number of connections has been reached, the test device must send a connect response to the requested answer address indicating E_NO_MORE_CONNECTIONS.

Parameters: None

Preparation:

Open device management connection:

TC sends **Connection Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x205 (CONNECTION_REQUEST)	0x0018 (Total Length)	0x08 (Struct Length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC	0x08 (Struct Length)	0x01 (IPV4_UDP)
IP_TC			DPORT_TC	0x02(Struct Length)	0x03 (DEVICE_MGMT_CONNECTION)

BDUT sends **Connection Response**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x208 (CONNECTION_RESPONSE)	0x0008 (Total Length)	**CID** (Channel ID)	0x00 (E_NO_ERROR)
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		DPORT_TC	
0x02(Struct Length)	0x03 (DEVICE_MGMT_CONNECTION)				

Result: CID

Test sequence:

TC sends **Connection Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x205 (CONNECTION_REQUEST)	0x0018 (Total Length)	0x08 (Struct Length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC	0x08 (Struct Length)	0x01 (IPV4_UDP)
IP_TC			DPORT_TC	0x02(Struct Length)	0x03 (DEVICE_MGMT_CONNECTION)

BDUT sends a **Connection Response**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x208 (CONNECTION_RESPONSE)	0x0008 (Total Length)	X (Channel ID)	0x24 (E_NO_MORE_CONNECTIONS)

Cleanup:

Close opened connection (see 3.6.1)

4.2 Device Configuration Request

4.2.1 Standard Case

Description: This test verifies the standard behaviour by sending a device configuration request over a valid device management connection. The request contains a property read request for the KNX_INDIVIDUAL_ADDRESS property.

Expectation: The test device must send a device configuration ACK followed by a device configuration request containing the requested data.

Parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

TC sends **Device Configuration Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	CID
0x00 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x000b (Interface Object Type = KNXnet/IP Parameter)	0x01 (Object Instance)	0x34 (PID = KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE=0x1, Six=0x001)
…						

BDUT sends a **Device Configuration Ack**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0x00 (Sequence Counter)	0x00 (Status)				

BDUT sends **Device Configuration Request:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0013 (Total Length)	0x04(Struct Length)	CID
0x00 (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	0x000b (Interface Object Type = KNXnet/IP Parameter)	0x01 (Object Instance)	0x34 (PID = KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE=0x1, Six=0x001)
…	**KNXADDR_BDUT**					

TC sends **Device Configuration Ack**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0x00 (Sequence Counter)	0x00 (Status)				

Cleanup:

Close opened connection (see 3.6.1)

4.2.2 Read mandatory device properties

Description: This test reads all mandatory properties from the KNXnet/IP parameter object. The results for KNX_INDIVIDUAL_ADDRESS, ROUTING_MULTICAST_ADDRESS, MAC_ADDRESS, and FRIENDLY_NAME are compared with corresponding info from DIB. The test reads also the mandatory property SERIAL_NUMBER from the device interface object and compares it with info from DIB.

Expectation: The test device must implement all properties and return values consistent with info from DIB.

Parameters: None

Out parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

This sequence should run with following values:

Seq	IOT	PID	Property Name	Resultlen	resultdata	NoESix
0	0x0000 (Device Object)	0x0b	PID_SERIAL_NUMBER	0x0017	XX XX XX XX XX XX (6 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x33	PID_PROJECT_INSTALLATION_ID	0x0013	XX XX (2 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x34	PID_KNX_INDIVIDUAL_ADDRESS	0x0013	KNXADDR_BDUT (2 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x35	PID_ADDITIONAL_INDIVIDUAL_ADDRESSES	0x0013	CNT_ADD_ADDR (2 bytes)	0x1000
prev +1	0x000b (KNXnet/IP Parameter Object)	0x35	PID_ADDITIONAL_INDIVIDUAL_ADDRESSES	0x0013	XX XX (2 bytes)	0x1 0x001
…	…	…	…	…	…	…
prev +1	0x000b (KNXnet/IP Parameter Object)	0x35	PID_ADDITIONAL_INDIVIDUAL_ADDRESSES	0x0013	XX XX (2bytes)	0x1000 + CNT_ADD_ADDR
prev +1	0x000b (KNXnet/IP Parameter Object)	0x37	PID_IP_ASSIGNMENT_METHOD	0x0012	XX (1 byte)	0x1 0x001

prev +1	0x000b (KNXnet/IP Parameter Object)	0x39	PID_CURRENT_IP_ADDRESS	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x3A	PID_CURRENT_SUBNET_MASK	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x3B	PID_CURRENT_DEFAULT_GATEWAY	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x3C	PID_IP_ADDRESS	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x3D	PID_SUBNET_MASK	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x3E	PID_DEFAULT_GATEWAY	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x40	PID_MAC_ADDRESS	0x0017	MAC of BDUT (6 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x41	PID_SYSTEM_SETUP_MULTICAST_ADDRESS	0x0015	E0 00 17 0C (4 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x42	PID_ROUTING_MULTICAST_ADDRESS	0x0015	XX XX XX XX (4bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x43	PID_TTL	0x0012	XX (1 byte)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x44	PID_KNXNETIP_DEVICE_CAPABILITIES	0x0013	XX XX (2 bytes)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x45	PID_KNXNETIP_DEVICE_STATE	0x0012	XX (1 byte)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x4C	PID_FRIENDLY_NAME	0x0012	XX (1 byte)	0x1 0x001
prev +1	0x000b (KNXnet/IP Parameter Object)	0x4C	PID_FRIENDLY_NAME	0x0012	XX (1 byte)	0x1 0x002
…	…	…	…	…	…	…
prev +1	0x000b (KNXnet/IP Parameter Object)	0x4C	PID_FRIENDLY_NAME	0x0012	XX (1 byte)	0x1 0x01E

TC sends Device Configuration Request**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	CID
seq (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	**IOT** (Interface Object Type)	0x01 (Object Instance)	**PID** (PID)	NoESix (NoE, Six)
…						

BDUT sends a Device Configuration Ack**:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
seq (Sequence Counter)	0x00 (Status)				

BDUT sends Device Configuration Request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		**resultlen** (Total Length)	0x04(Struct Length)	CID
Seq (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	**IOT** (Interface Object Type)	0x01 (Object Instance)	**PID** (PID)	NoESix (NoE, Six)
…	**resultdata**					

TC sends Device Configuration Ack**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
seq (Sequence Counter)	0x00 (Status)				

Cleanup:

Close opened connection (see 3.6.1)

4.2.3 Write to read-only device property

Function ID: 20205

Description: This test attempts to write to the read-only property PID_CURRENT_IP_ADDRESS

Expectation: Confirmation with error code “Read-only” or "Unspecified Error".

Parameters: None

Out parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

TC sends **Device Configuration Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0015 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xf6 (M_PropWrite.req)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x39 (PID = PID_CURRENT_IP_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)
…	0x12345678 (Data)					

BDUT sends a **Device Configuration Ack**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

BDUT sends **Device Configuration Request:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0012 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xf5 (M_PropWrite.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x39 (PID = PID_CURRENT_IP_ADDRESS)	**0x0001** (NoE = **0x0**, Six = 0x001)
…	**0x00 (unspecified Error) or 0x05 (Read only)**					

TC sends **Device Configuration Ack**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

Cleanup:

Close opened connection (see 3.6.1)

4.2.4 Read non-existing device property

Function ID: 20206

Description: This test attempts to read a non-existing property of the test device.

Expectation: Response with NoE = 0 and error code “Void DP” or "Unspecified Error".

Parameters: PID = ID of non-existing property (Type = Byte, Default = 0xF0)

Out parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

TC sends **Device Configuration Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	**PID** (PID)	0x1001 (NoE = 0x1, Six = 0x001)
…						

BDUT sends a **Device Configuration Ack**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

BDUT sends **Device Configuration Request:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0012 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	**PID** (PID)	**0x0001** (NoE = **0x0**, Six = 0x001)
…	**0x00 (unspecified Error) or 0x07 (Viod DP)**					

TC sends **Device Configuration Ack**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

Cleanup:

Close opened connection (see 3.6.1)

4.2.5 Get/set programming mode of device

Function ID: 20207

Description: This test accesses the device's programming mode by device property, and checks the written property value against info from freshly requested DIB.

Expectation: Device's programming LED should react correspondingly.

Parameters: New programming mode state, 0=LED on, 1=LED off, 2=toggle (Type = Byte, Default = 0x02)

Out parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

TC sends **Device Configuration Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)	0x0011 (Total Length)		0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x0000 (Interface Object Type = Device Object)	0x01 (Object Instance)	0x36 (PID = PID_PROGMODE)	0x1001 (NoE = 0x1, Six = 0x001)
…						

BDUT sends a **Device Configuration Ack**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

BDUT sends **Device Configuration Request:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)	0x0012 (Total Length)		0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x36 (PID = PID_PROGMODE)	0x1001 (NoE = 0x1, Six = 0x001)
…	**PRG_MOD**					

TC sends **Device Configuration Ack**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

TC sends **Device Configuration Request**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)	0x0012 (Total Length)		0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xf6 (M_PropWrite.req)	0x0000 (Interface Object Type = Device Object)	0x01 (Object Instance)	0x36 (PID = PID_PROGMODE)	0x1001 (NoE = 0x1, Six = 0x001)
…	**(PRG_MOD ^0x01)**					

BDUT sends a **Device Configuration Ack**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

BDUT sends **Device Configuration Request:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0012 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xf5 (M_PropWrite.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x36 (PID = PID_PROGMODE)	0x1001 (NoE = 0x1, Six = 0x001)
…						

TC sends **Device Configuration Ack**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

Read Device Description (see 3.3.1): Field Device status must be PRG_MOD ^0x01**.**

Cleanup:

Close opened connection (see 3.6.1)

4.2.6 Get/set programming mode of device by memory access

Function ID: 20208

Description: This test accesses the device's programming mode by device memory cell 0x60, and checks the written value against info from freshly requested DIB.

Expectation: Device's programming LED should react correspondingly.

Parameters: New programming mode state, 0=LED on, 1=LED off, 2=toggle (Type = Byte, Default = 0x02)

Out parameters: None

Preparation: none

Test sequence:

```
KNX: TC -> BDUT: B0 11 FD 11 00 60 80 :52           T_Connect
KNX: TC -> BDUT: B0 11 FD 11 00 61 43 00 :90        MaskVersionRead
KNX: BDUT -> TC: B0 11 00 11 FD 60 C2 :10           T_Ack
KNX: BDUT -> TC: B0 11 00 11 FD 63 43 40 09 1A :C1  MaskVersionResponse
KNX: TC -> BDUT: B0 11 FD 11 00 60 C2 :10           T_Ack
KNX: TC -> BDUT: B0 11 FD 11 00 63 46 01 00 60 :F6  MemoryRead
KNX: BDUT -> TC: B0 11 00 11 FD 60 C6 :14           T_Ack
KNX: BDUT -> TC: B0 11 00 11 FD 64 46 41 00 60 81 :30     MemoryResponse
KNX: TC -> BDUT: B0 11 FD 11 00 60 C6 :14           T_Ack
KNX: TC -> BDUT: B0 11 FD 11 00 64 4A 81 00 60 00 :7DMemoryWrite
KNX: BDUT -> TC: B0 11 00 11 FD 60 CA :18           T_Ack
KNX: TC -> BDUT: B0 11 FD 11 00 63 4E 01 00 60 :FE  MemoryRead
KNX: BDUT -> TC: B0 11 00 11 FD 60 CE :1C           T_Ack
KNX: BDUT -> TC: B0 11 00 11 FD 64 4A 41 00 60 00 :BDMemoryWrite
KNX: TC -> BDUT: B0 11 FD 11 00 60 CA :18           T_Ack
KNX: TC -> BDUT: B0 11 FD 11 00 60 81 :53           Disconnect
```

Read Device Description (see 1.2.2): Field Device status must be PRG_MOD ^0x01.

Cleanup: None

4.2.7 Change Individual Address

Function ID: 20209

Description: This test changes the device's Individual Address and checks the corresponding device and KNXnet/IP properties.

Expectation: Both properties must reflect the change, and must agree with info from DIB.

Parameters: None

Out parameters: None

Preparation:

Set Progmode on BDUT (DMP_ProgModeSwitch_RCo)

Set Individual address of BDUT to 0x1200 (NM_IndividualAddress_Write)

Test sequence:

Read Device Description (see 3.3.1): Field **Individual address** must be **0x1200**.

Read the following properties (see 4.2.2)

Seq	IOT	PID	Property Name	Resultlen	resultdata	NoESix
0	0x000b (KNXnet/IP Parameter Object)	0x34	PID_KNX_INDIVIDUAL_ADDRESS	0x0013	0x1200 (2 bytes)	0x1 0x001
prev +1	0x0000 (Device Object)	0x39	PID_SUBNET_ADDR	0x0012	0x12 (1 byte)	0x1 0x001
prev	0x0000 (Device Object)	0x3A	PID_DEVICE_ADDR	0x0012	0x00 (1 byte)	0x1 0x001

+1						

Cleanup:

Write KNX IA back to 0x1100 and restart device

4.2.8 Change Individual Address by IP property

Function ID: 20210

Description: This test changes the device's Individual Address by writing to KNXnet/IP property KNX_INDIVIDUAL_ADDRESS and checks the corresponding device and KNXnet/IP properties.

Expectation: Both properties must reflect the change, and must agree with info from DIB.

Parameters: None

Out parameters: None

Preparation:

IP DevMan Write KNX IA (KNXnet/IP) to 0x1200, Restart

Test sequence:

Read Device Description (see 3.3.1): Field **Individual address** must be **0x1200**.

Read the following properties (see 4.2.2)

Seq	IOT	PID	Property Name	Resultlen	resultdata	NoESix
0	0x000b (KNXnet/IP Parameter Object)	0x34	PID_KNX_INDIVIDUAL_ADDRESS	0x0013	0x1200 (2 bytes)	0x1 0x001
prev +1	0x0000 (Device Object)	0x39	PID_SUBNET_ADDR	0x0012	0x12 (1 byte)	0x1 0x001
prev +1	0x0000 (Device Object)	0x3A	PID_DEVICE_ADDR	0x0012	0x00 (1 byte)	0x1 0x001

Cleanup:

Write KNX IA back to 0x1100 and restart device

4.2.9 Invalid Endpoint

Function ID: 20201

Description: This test sends a device configuration request to an invalid endpoint (UDP/IP port) of the test device.

Expectation: The test device should ignore this request and not send any answer

Parameters: IEP = Port number (Type = Integer, Default = 1)

Out Parameters: None

Preparation: None

Test sequence:

TC sends KNXnet/IP Packet to wrong UDP Port:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, **IEP**)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	0x0 (CID)
0 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x34 (PID = PID_KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)
…						

BDUT sends ICMP Packet:

ICMP-frame (IP_BDUT, IP_TC, **3 (Destination Unreachable), 3 (Port unreachable)**)
Internet Header + 64 bits of Original Data Datagram (don't care)

Cleanup: None

4.2.10 Unconnected Endpoint

Function ID: 20202

Description: This test sends a device configuration request to the control endpoint of the test device, without connecting to it.

Expectation: The test device should ignore this request and not send any answer

Parameters: None

Out Parameters: None

Preparation: None

Test sequence:

TC sends KNXnet/IP Packet:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_DTU)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	0x0 (CID)
0 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x34 (PID = PID_KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)
…						

No answer from BDUT is allowed for 10 seconds.

Cleanup: None

4.2.11 Repeat and timeout after missing ACK

Function ID: 20211

Description: This test sends a property read request to the test device without sending ACK after receiving the response and checks how often and how long the server repeats its response before disconnecting.

Expectation: The test device should repeat its message 3 times every 10 seconds and then disconnect.

Parameters: None

Out Parameters: Observed number of repetitions and total duration of repetitions

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_DTU)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xfc (M_PropRead.req)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x34 (PID = PID_KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)
…						

BDUT sends a Device Configuration Ack**:**

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

BDUT sends Device Configuration Request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0013 (Total Length)	0x04(Struct Length)	CID
0x2 (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x34 (PID = PID_KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)

…	KNXADDR_BDUT

No Device Configuration Ack form test client. BDUT must resend the Device Configuration Request 10 second+-10% after its first Device Configuration Request. (3 times)

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0013 (Total Length)	0x04(Struct Length)	CID
0x2 (Sequence Counter)	0x00 (reserved)	0xfb (M_PropRead.con)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x34 (PID = PID_KNX_INDIVIDUAL_ADDRESS)	0x1001 (NoE = 0x1, Six = 0x001)
…	KNXADDR_BDUT					

No Device Configuration Ack form test client. BDUT must not resend the Device Configuration **Request.** Wait 20 seconds.

Cleanup:

Close opened connection (see 3.6.1)

4.2.12 Bus connection interrupted

Function ID: 20212

Description: This test interrupts and then re-establishes the bus connection by means of a load switch and checks for corresponding KNXNETIP_DEVICE_STATE property info indications received on a device management connection.

Expectation: These property info indications should occur after disconnecting from bus and after re-connecting to bus.

Parameters: Group address for load switch (Type = String, Default = 1/1/50)

Out Parameters: None

Preparation:

Open a device management connection (see 4.1.1)

Used results: Channel ID (CID),

Test sequence:

Interrupt bus connection

BDUT sends Device Configuration Request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0012 (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (reserved)	0xf7 (M_PropInfo.ind)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x45 (PID = PID_KNXNETIP_DEVICE_STATE)	0x1001 (NoE = 0x1, Six = 0x001)
…	**0x01 (KNX Fault)**					

TC sends Device Configuration Ack**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
0 (Sequence Counter)	0x00 (Status)				

Reconnect bus connection

BDUT sends Device Configuration Request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x310 (DEVICE_CONFIGURATION_REQUEST)		0x0012 (Total Length)	0x04(Struct Length)	CID
1 (Sequence Counter)	0x00 (reserved)	0xf7 (M_PropInfo.ind)	0x000b (Interface Object Type = KNXnet/IP Parameter Object)	0x01 (Object Instance)	0x45 (PID = PID_KNXNETIP_DEVICE_STATE)	0x1001 (NoE = 0x1, Six = 0x001)
…	**0x00 (No Fault)**					

TC sends Device Configuration Ack**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x311 (DEVICE_CONFIGURATION_ACK)	0x000a (Total Length)	0x04(Struct Length)	CID
1 (Sequence Counter)	0x00 (Status)				

Cleanup:

Close opened connection (see 3.6.1)

4.3 cEMI Transport Layer

These tests check KNXnet/IP Device Management via DEVICE_CONFIGURATION_REQUEST using cEMI Transport Layer services (T_Data_Individual, T_Data_Connected) according to AN118.

4.3.1 T_Data_Individual.req to unconnected BDUT

Function ID: 20301

Description: This test sends KNXnet/IP DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read to the Control Endpoint of the BDUT without KNXnet/IP connection.

Expectation:

The BDUT ignores this request in unconnected context, and sends no DEVICE_CONFIGURATION_ACK, and no responding DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response.

Parameters: None.

Preparation: None.

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read to the Control Endpoint of the BDUT:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0019 (Total Length)
0x04 (Struct Length)	0x00 (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x4A 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Individual.req PropValueRead OX=0 PID=1 N=1 X=1)				

BDUT does not respond.

Cleanup: None.

4.3.2 T_Data_Individual.ind to unconnected BDUT

Function ID: 20302

Description: This test sends KNXnet/IP DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response to the Control Endpoint of the BDUT without KNXnet/IP connection.

Expectation:

The BDUT ignores this request in unconnected context, and sends no DEVICE_CONFIGURATION_ACK.

Parameters: None.

Preparation: None.

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response to the Control Endpoint of the BDUT:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0x00 (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Individual.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

BDUT does not respond.

Cleanup: None.

4.3.3 T_Data_Connected.req to unconnected BDUT

Function ID: 20303

Description: This test sends KNXnet/IP DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read to the Control Endpoint of the BDUT without KNXnet/IP connection.

Expectation:

The BDUT ignores this request in unconnected context, and sends no DEVICE_CONFIGURATION_ACK, and no responding DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response.

Parameters: None.

Preparation: None.

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read to the Control Endpoint of the BDUT:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0019 (Total Length)
0x04 (Struct Length)	0x00 (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x41 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Connected.req PropValueRead OX=0 PID=1 N=1 X=1)				

BDUT does not respond.

Cleanup: None.

4.3.4 T_Data_Connected.ind to unconnected BDUT

Function ID: 20304

Description: This test sends KNXnet/IP DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response to the Control Endpoint of the BDUT without KNXnet/IP connection.

Expectation:

The BDUT ignores this request in unconnected context, and sends no DEVICE_CONFIGURATION_ACK.

Parameters: None.

Preparation: None.

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response to the Control Endpoint of the BDUT:

<table>
<tr><td colspan="5">UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)</td></tr>
<tr><td>0x06
(Header Length)</td><td>0x10
(Ver.)</td><td colspan="2">0x0310
(CONFIG_REQUEST)</td><td>0x001B
(Total Length)</td></tr>
<tr><td>0x04
(Struct Length)</td><td>0x00
(Channel ID)</td><td>0x00
(Sequence Counter)</td><td>0x00
(Reserved)</td><td></td></tr>
<tr><td colspan="5">0x89 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07
0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00
(cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)</td></tr>
</table>

BDUT does not respond.

Cleanup: None.

4.3.5 T_Data_Individual.req to BDUT

Function ID: 20305

Description: This test opens a KNXnet/IP Device Management connection, and sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0019 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x4A 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Individual.req PropValueRead OX=0 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.6 T_Data_Individual.ind to BDUT

Function ID: 20306

Description: This test opens a KNXnet/IP Device Management connection, and sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response.

Expectation:

The BDUT sends only DEVICE_CONFIGURATION_ACK.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Individual.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.7 T_Data_Connected.req to BDUT

Function ID: 20307

Description: This test opens a KNXnet/IP Device Management connection, and sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0019 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x41 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Connected.req PropValueRead OX=0 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x89 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.8 T_Data_Connected.ind to BDUT

Function ID: 20308

Description: This test opens a KNXnet/IP Device Management connection, and sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response.

Expectation:

The BDUT sends only DEVICE_CONFIGURATION_ACK.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.9 M_PropRead.req after T_Data_Individual.req

Function ID: 20309

Description: This test opens a KNXnet/IP Device Management connection, then sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read, then reads and acknowledges the response, and then sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.req A_PropertyValue_Read:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)			
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)	0x0019 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)
0x4A 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Individual.req PropValueRead OX=0 PID=1 N=1 X=1)			

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)			
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)	0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0011 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFC 0x00 0x00 0x01 0x01 0x10 0x01 (cEMI: M_PropRead.req OT=0 OI=1 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0013 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFB 0x00 0x00 0x01 0x01 0x10 0x01 0x00 0x00 (cEMI: M_PropRead.con OT=0 OI=1 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.10 M_PropRead.req after T_Data_Individual.ind

Function ID: 20310

Description: This test opens a KNXnet/IP Device Management connection, then sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response, and then sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Individual.ind A_PropertyValue_Response:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Individual.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0011 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFC 0x00 0x00 0x01 0x01 0x10 0x01 (cEMI: M_PropRead.req OT=0 OI=1 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0013 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFB 0x00 0x00 0x01 0x01 0x10 0x01 0x00 0x00 (cEMI: M_PropRead.con OT=0 OI=1 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.11 M_PropRead.req after T_Data_Connected.req

Function ID: 20311

Description: This test opens a KNXnet/IP Device Management connection, then sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read, then reads and acknowledges the response, and then sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.req A_PropertyValue_Read:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0019 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x41 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x05 0x03 0xD5 0x00 0x01 0x10 0x01 (cEMI: T_Data_Connected.req PropValueRead OX=0 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0011 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFC 0x00 0x00 0x01 0x01 0x10 0x01 (cEMI: M_PropRead.req OT=0 OI=1 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0013 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFB 0x00 0x00 0x01 0x01 0x10 0x01 0x00 0x00 (cEMI: M_PropRead.con OT=0 OI=1 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

4.3.12 M_PropRead.req after T_Data_Connected.ind

Function ID: 20312

Description: This test opens a KNXnet/IP Device Management connection, then sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response, and then sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req.

Expectation:

The BDUT sends DEVICE_CONFIGURATION_ACK, and then a responding DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf.

Parameters: None.

Preparation:

TC opens Device Management connection (see 4.1.1).

Test sequence:

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI T_Data_Connected.ind A_PropertyValue_Response:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x001B (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	
0x94 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x03 0xD6 0x00 0x01 0x10 0x01 0x00 0x00 (cEMI: T_Data_Connected.ind PropValueResponse OX=0 PID=1 N=1 X=1 $0000)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x00 (Sequence Counter)	0x00 (Reserved)	

TC sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.req:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0011 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFC 0x00 0x00 0x01 0x01 0x10 0x01 (cEMI: M_PropRead.req OT=0 OI=1 PID=1 N=1 X=1)				

BDUT sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

BDUT sends DEVICE_CONFIGURATION_REQUEST with cEMI M_PropRead.conf:

UDP frame (IP_BDUT : CPORT_BDUT -> IP_TC : CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0310 (CONFIG_REQUEST)		0x0013 (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	
0xFB 0x00 0x00 0x01 0x01 0x10 0x01 0x00 0x00 (cEMI: M_PropRead.con OT=0 OI=1 PID=1 N=1 X=1 $0000)				

TC sends DEVICE_CONFIGURATION_ACK:

UDP frame (IP_TC : CPORT_TC -> IP_BDUT : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0311 (CONFIG_ACK)		0x000A (Total Length)
0x04 (Struct Length)	0xXX (Channel ID)	0x01 (Sequence Counter)	0x00 (Reserved)	

Cleanup:

TC closes Device Management connection (see 3.6.1).

5 Tunnelling

5.1 Connection Handling

5.1.1 Standard Case

Function ID: 30101

Description: This test verifies the standard behaviour by sending a connection request message (KNX Link Layer) for a tunnelling connection to the test device.

Expectation: If the device supports tunnelling, the connection should succeed (test result: true), otherwise the connection should fail (test result is also: true).

Parameters: None

Out Parameters: None

Preparation:

Read the possible KNX tunnel addresses from the Property "PID_ADDITIONAL_INDIVIDUAL_ADDRESSES" (object type 0x000B, instance: 0x01, property id 0x35).

Used results: KNX tunnel address table (KTAT)

Test sequence:

Send CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			DPORT_TC	0x04 length	0x04 TUNNEL_CONNECTION
0x02 TUNNEL_LINKLAYER	0x00				

BDUT sends CONNECT_RESPONSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total length)	CID	0x00 Status= E_NO_ERROR
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		DPORT_TC	
0x04 (Struct length)	0x04	KNX Individual Address			

The returned KNX additional Individual Address shall be an element of the KNX tunnel address table (KTAT).

Used Results: CID

Send DISCONNECT_REQUEST

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x209 (DISCONNECT_REQUEST)	0x0010 (Total length)	CID	0x00 (reserved)
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		CPORT_TC	

BDUT sends DISCONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x20A (DISCONNECT_RESPONSE)	0x0008 (Total length)	CID	0x00

5.1.2 Multiplicity

Function ID: 30102

Description: This test verifies the standard behaviour by sending a valid connect request message for a tunnelling connection (KNX Link Layer) until the device cannot take any more connections.

Expectation: When the maximum number of connections has been reached, the test device must send a connect response to the requested answer address indicating E_NO_MORE_CONNECTIONS.

Parameters: Minimum number of connections (Type = Integer, Default = 1), Maximum number of connections (Type = Integer, Default = 10).

Out Parameters: Measured maximum number of concurrent device tunnel connections

Preparation:

Read the possible KNX tunnel addresses from the Property "PID_ADDITIONAL_INDIVIDUAL_ADDRESSES" (object type 0x000B, instance: 0x01, property id 0x35).

Used results: Length of KNX tunnel address table = N

Test sequence:

Send **CONNECT_REQUEST**:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			DPORT_TC	0x04 length	0x04 TUNNEL_CONNECTION
0x02 TUNNEL_LINKLAYER	0x00				

BDUT sends CONNECT_RESPONSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total length)	CID	0x00 Status= E_NO_ERROR
0x08 (Struct length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC	
0x04 (Struct length)	0x04	KNX Individual Address			

The returned KNX additional Individual Address shall be an element of the KNX additional Individual Address table (KTAT).

Save CID

Repeat this sequence until the response of the BDUT is:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total length)	X (CID)	0x24 Status= E_NO_MORE_ CONNECTIONS

The repeat count shall be higher or equal than minimum number of connections and lower or equal than maximum number of connections.

Clean up:

Close all connections.

5.1.3 cEMI Raw Mode

Function ID: 30105

Description: This test verifies the standard behaviour by sending a valid connect request message for a tunnelling connection (cEMI Raw Mode) to the test device.

Expectation: If the device supports cEMI Raw Mode Tunnelling, the connection should succeed; otherwise the device should response to the requested answer address indicating E_CONNECTION_OPTION.

Parameters: cEMI Raw Mode Supported (Type = Boolean, Default = true)

Out Parameters: None

Reaction to valid connect request message for a tunnelling connection when supporting cEMI Raw Mode tunnelling still to be added

Preparation: None

Test sequence:

Send **CONNECT_REQUEST:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_UDP)
IP_TC			DPORT_TC	0x04 length	0x04 TUNNEL_CONNECTION
0x04 TUNNEL_RAW	0x00				

BDUT sends **CONNECT_RESPONSE** if "cEMI Raw Mode Supported" = FALSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total length)	X (CID)	0x23 E_CONNECTION_OPTION

Sequence ends

BDUT sends **CONNECT_RESPONSE** if "cEMI Raw Mode Supported" = TRUE

(Constructed in analogy to 4.1.1, FuncID 30101) / TBD

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total length)	CID	0x00 Status= E_NO_ERROR
0x08 (Struct length)	0x01 (IPV4_UDP)	IP_TC		CPORT_TC	
0x04 (Struct length)	0x04	KNX Individual Address			

Cleanup:

Close the connection

5.1.4 KNX Busmonitor Mode

Function ID: 30107

Description: This test verifies the standard behaviour by sending a valid connect request message for a tunnelling connection (KNX Busmonitor Mode) to the test device.

Expectation: If the device supports KNX Busmonitor Mode Tunnelling, the connection should succeed; otherwise the device should response to the requested answer address indicating E_CONNECTION_OPTION.

Parameters: KNX Busmonitor Mode Supported (Type = Boolean, Default = true)

Out Parameters: None

Preparation: None

Test sequence:

Send CONNECT_REQUEST**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			DPORT_TC	0x04 length	0x04 TUNNEL_ CONNECTION
0x80 TUNNEL_ BUS-MONITOR	0x00				

BDUT sends CONNECT_RESPONSE if “KNX Busmonitor Mode Supported” = FALSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total length)	X (CID)	0x23 E_ CONNECTION _OPTION

Sequence ends

BDUT sends CONNECT_RESPONSE if "KNX Busmonitor Mode Supported" = TRUE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total length)	CID	0x00 Status= E_NO_ERROR
0x08 (Struct length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC	
0x04 (Struct length)	0x04	KNX Individual Address			

Cleanup:

Close the connection

5.1.5 Invalid KNX layer code

Function ID: 30108

Description: This test verifies the standard behaviour by sending a tunnelling connect request containing a random invalid KNX layer code to the test device.

Expectation: The test device should respond indicating a “connection option not supported” error (E_CONNECTION_OPTION).

Parameters: Number of retries with random layer codes (Type = Integer, Default = 3)

Out Parameters: None

Preparation: None

Test sequence:

Repeat this sequence for 3 different not defined tunnel layer codes (TLC != 0x02, 0x04, 0x80)

Send CONNECT_REQUEST**:**

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			DPORT_TC	0x04 length	0x04 TUNNEL_ CONNECTION
TLC	0x00				

BDUT sends CONNECT_RESPONSE

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total length)	X (CID)	0x23 E_ CONNECTION _OPTION

5.2 Tunnelling Request

5.2.1 Standard Case Tunnelling to KNX

Function ID: 30204

Description: This test verifies the standard behaviour by sending a tunnelling request message on a KNX Link Layer Tunnelling Connection. The tunnelling message contains an L_Data.req service with fixed data.

Expectation: The telegram must be received over an open Falcon KNX connection.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open a tunnelling connection

Used Results: CID, KNX Individual Address

Test sequence:

TC sends TUNNELLING_REQUEST with a valid L_DATA_REQUEST cEMI message (CEMI_TEST)

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_DATA_REQUEST cEMI message (CEMI_TEST) e.g.: 11 00 bc c0 00 00 12 34 04 00 80 56 78 9A			
…					

Sequence mark 5.2.1/1

BDUT sends TUNNELLING_ACK:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends TUNNELLING_REQUEST with valid L_DATA_CONFIRM for the :CEMI_TEST telegram and inserted KNX Individual Address

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 2E 00 bc c0 KNX Individual Address (2) 12 34 04 00 80 56 78 9A			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends on KNX line (received over Falcon) telegram according to send CEMI_TEST telegram. Telegram may be received from sequence mark 5.2.1/1 onward. e.g.:

BC (Control Byte)	KNX Individual Address (Source address)		1234 (Destination address)	C4 (APDU-Length =4)	00	80
56	78	9A	6C (Check-sum)			

Cleanup

Close connection

5.2.2 Standard Case Tunnelling from KNX

Function ID: 30205

Description: This test verifies the standard behaviour by sending a group data message over a Falcon KNX connection.

Expectation: The telegram must be received over an open Tunnelling connection.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open tunnelling connection

Used Results: CID

Test sequence:

TC sends telegram on KNX line. This telegram shall be a valid KNX group telegram (GRP_TEST). E.g.:

BC (Control Byte)	11 FD (Source address)		1234 (Destination address)	E4 (APDU-Length =4)	00	80
56	78	9A	Check-sum			

BDUT sends TUNNELLING_REQUEST with valid cEMI L_DATA_INDICATION for the GRP_TEST telegram (the cEMI message may have additional information's).

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 29 00 bc e0 11 fd 12 34 04 00 80 56 78 9a			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 Status= E_NO_ERROR				

Cleanup

Close connection

5.2.3 Not increased Sequence Counter

Function ID: 30206

Description: This test verifies the error case behaviour by sending a number of tunnelling requests to the test device; after that, a tunnelling request with the same sequence counter as before (not increased) is transmitted.

Expectation: The last tunnelling request must be responded with a tunnelling ACK, but not transmitted on the KNX installation.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open connection

Used Results: CID, KNX Individual Address

Repeat following sequence for 10 times. SC is the sequence counter from 0 to 9 incremented every cycle:

TC sends TUNNELLING_REQUEST with a valid L_DATA_REQUEST cEMI message (CEMI_TEST)

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)

0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 (reserved)	Valid L_DATA_REQUEST cEMI message (CEMI_TEST) e.g.: 11 00 bc c0 00 00 12 34 05 00 80 00 00 00 (SC+1)			
…					

Sequence mark 5.2.3/1

BDUT sends TUNNELLING_ACK:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends TUNNELLING_REQUEST with valid L_DATA_CONFIRM for the :CEMI_TEST telegram and inserted KNX Individual Address

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 2E 00 bc c0 KNX **Individual Address (2)** 12 34 05 00 80 00 00 00 (SC+1)			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends on KNX line (received over Falcon) telegram according to send CEMI_TEST telegram. Telegram may be received from sequence mark 5.2.3/1 onward. e.g.:

BC (Control Byte)	KNX Individual Address (Source address)		1234 (Destination address)		C5 (APDU-Length =4)	00	80
00	00	00	(SC+1)	Valid check sum			

End of repeated sequence

Test sequence:

TC sends TUNNELLING_REQUEST with a valid L_DATA_REQUEST cEMI message (CEMI_TEST) and wrong sequence number (sequence number of last transaction).

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
09 (Sequence counter)	0x00 (reserved)	Valid L_DATA_REQUEST cEMI message (CEMI_TEST) e.g.: 11 00 bc c0 00 00 12 34 05 00 80 00 00 00 0b			
…					

BDUT sends TUNNELLING_ACK:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
09 (Sequence counter)	0x00 Status= E_NO_ERROR				

Wait for more than 10 seconds (The wait time depends on the cEMI server maximal reaction time of the BDUT)

The BDUT shall not send a TUNNELLING_REQUEST with a cEMI L_DATA_CONFIRM message.

The BDUT shall not send a telegram on the KNX line.

Cleanup

Close the connection

5.2.4 Sequence Counter increased by two

Function ID: 30207

Description: This test verifies the error case behaviour by sending a number of tunnelling requests to the test device; after that, a tunnelling request with a sequence counter of one too large is transmitted.

Expectation: The last tunnelling request must not be responded with a tunnelling ACK, and not be transmitted on the KNX installation.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open tunnelling connection

Used Results: CID, KNX Individual Address

Repeat following sequence for 10 times. SC is the sequence counter from 0 to 9 incremented every cycle:

TC sends TUNNELLING_REQUEST with a valid L_DATA_REQUEST cEMI message (CEMI_TEST)

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 (reserved)	Valid L_DATA_REQUEST cEMI message (CEMI_TEST) e.g.: 11 00 bc c0 00 00 12 34 05 00 80 00 00 00 (SC+1)			
…					

Sequence mark 5.2.4/1

BDUT sends TUNNELLING_ACK:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ACK)	0x000A (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends TUNNELLING_REQUEST with valid L_DATA_CONFIRM for the: CEMI_TEST telegram and inserted KNX Individual Address

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 2E 00 bc c0 KNX **Individual Address (2)** 12 34 05 00 80 00 00 00 (SC+1)			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ACK)	0x000A (Total length)	0x04 (Struct length)	CID
SC (Sequence counter)	0x00 Status= E_NO_ERROR				

BDUT sends on KNX line (received over Falcon) telegram according to send CEMI_TEST telegram. Telegram may be received from sequence mark 5.2.4/1 onward. e.g.:

BC (Control Byte)	KNX Individual Address (Source address)		1234 (Destination address)		C5 (APDU-Length =4)	00	80
00	00	00	(SC+1)	Valid check sum		

End of repeated sequence

Test sequence:

TC sends TUNNELLING_REQUEST with a valid L_DATA_REQUEST cEMI message (CEMI_TEST) and wrong sequence number (sequence number of last transaction plus two).

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0019 (Total length)	0x04 (Struct length)	CID
0B (Sequence counter)	0x00 (reserved)	Valid L_DATA_REQUEST cEMI message (CEMI_TEST) e.g.: 11 00 bc c0 00 00 12 34 05 00 80 00 00 00 0b			
…					

Wait for more than 10 seconds (The wait time depends on the cEMI server maximal reaction time of the BDUT)

The BDUT shall not send a TUNNELLING_ACK

The BDUT shall not send a TUNNELLING_REQUEST with a cEMI L_DATA_CONFIRM message.

The BDUT shall not send a telegram on the KNX line.

Cleanup

Close the connection

5.2.5 Standard Case Busmonitor Tunneled from KNX

Function ID: 30208

Description: This test verifies the standard behaviour by sending a busmonitor indication message from KNX by busmonitor tunnelling connection.

Expectation: The telegram must be received over an open busmonitor tunnelling connection.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open tunnelling busmonitor connection (if supported)

Used Result: CID

Test sequence:

TC sends telegram on KNX line. This telegram shall be a valid KNX group telegram (GRP_TEST). E.g.:

BC (Control Byte)	11 FD (Source address)		1234 (Destination address)	E4 (APDU-Length =4)	00	80
56	78	9A	Check-sum		

BDUT sends TUNNELLING_REQUEST with valid cEMI L_Busmon.ind for the GRP_TEST telegram (the cEMI message may have additional information).

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_Busmon.ind cEMI message for CEMI_TEST e.g.: 2b 00 bc e0 11 fd 12 34 04 00 80 56 78 9a			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 Status= E_NO_ERROR				

Cleanup: Close connection

5.2.6 Standard Case Raw Mode Tunneled from KNX

Function ID: 30209

Description: This test verifies the standard behaviour by sending a raw data indication message from KNX by raw tunnelling connection.

Expectation: The telegram must be received over an open raw tunnelling connection.

Parameters: Falcon connection parameters

Out parameters: None

Preparation: Open tunnelling raw connection (if supported) - Used Result: CID

Test sequence:

TC sends telegram on KNX line. This telegram shall be a valid KNX group telegram (GRP_TEST), e.g.:

BC (Control Byte)	11 FD (Source address)		1234 (Destination address)	E4 (APDU-Length =4)	00	80
56	78	9A	Check-sum		

BDUT sends TUNNELLING_REQUEST with valid cEMI L_Raw.ind for the GRP_TEST telegram (the cEMI message may have additional information).

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_Raw.ind cEMI message for CEMI_TEST e.g.: 2d 00 bc e0 11 fd 12 34 04 00 80 56 78 9a			
…					

TC sends TUNNELLING_ACK:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0421 (TUNNELLING_ ACK)	0x000A (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 Status= E_NO_ERROR				

Cleanup: Close connection

5.2.7 Repeat and timeout after missing ACK

Function ID: 30210

Description: This test sends a KNX read request to the test device by tunnelling, without sending ACK after receiving the telegram on the tunnelling connection and checks how often and how long the server repeats it before giving up.

Expectation: The test device should repeat the telegram once after 1 second.

Parameters: None

Out Parameters: Observed number of repetitions and total duration of repetitions

Preparation: Open tunnelling Data Link Layer connection

Used Result: CID

Test Sequence

Send telegram on KNX line. This telegram shall be a valid KNX group telegram (GRP_TEST). E.g.:

BC (Control Byte)	11 FD (Source address)		1234 (Destination address)	E4 (APDU-Length =4)	00	80
56	78	9A	Check-sum			

BDUT sends TUNNELLING_REQUEST with valid cEMI L_DATA_INDICATION for the GRP_TEST telegram (the cEMI message may have additional information's).

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 29 00 bc e0 11 fd 12 34 04 00 80 56 78 9a			
…					

BDUT sends TUNNELLING_REQUEST with valid cEMI L_DATA_INDICATION for the GRP_TEST telegram (the cEMI message may have additional information's). The Telegram shall be transmitted one second ±10% after the first TUNNELLING_REQUEST

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_ REQUEST)	0x0018 (Total length)	0x04 (Struct length)	CID
0x00 (Sequence counter)	0x00 (reserved)	Valid L_DATA_CONFIRM cEMI message for CEMI_TEST e.g.: 29 00 bc e0 11 fd 12 34 04 00 80 56 78 9a			
…					

Cleanup: Close connection

5.2.8 Broadcast telegram tunnelled to KNX

Function ID: 30211

Description: This test checks tunnelling of a broadcast telegram to KNX subnet.

Expectation: The test telegram should appear on the KNX connection, and the test device should send a confirmation telegram back to the tunnelling connection.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: Test application sends a valid CONNECT_REQUEST (tunnelling, Data Link Layer, see 2.4.1)

BDUT sends a valid CONNECT_RESPONSE

Used results: Channel ID (CID), Individual Address (IA)

Test sequence:

Test application sends:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)							
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0015 (Total Length)		0x04 (Struct-Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x11 (cEMI Service Code)	0x00 (additional info length)	0xA0 (Ctrl1)	0xE0 (Ctrl2)	0x0000 (Source Address)	
0x0000 (Destination Address)		0x01	0x01	0x00			

BDUT sends a valid TUNNELLING_ACK

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0015 (Total Length)		0x04 (Struct-Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x2E (cEMI Service Code)	0x00 (additional info length)	0xB0 (Ctrl1)	0xE0 (Ctrl2)	IA (Source Address)	
0x0000 (Destination Address)		0x01	0x01	0x00			

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the telegram is received via KNX bus connection:

`B0 [ I A ] 00 00 E1 01 00`

Cleanup: Close connection

5.2.9 Broadcast telegram tunnelled from KNX

Function ID: 30212

Description: This test checks tunnelling of a broadcast telegram from KNX subnet by calling Falcon IndividualAddressRead.

Expectation: A corresponding data indication telegram should arrive on the tunnelling connection.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: Test application sends a valid CONNECT_REQUEST (tunnelling, Data Link Layer, see 2.4.1)

BDUT sends a valid CONNECT_RESPONSE

Used results: Channel ID (CID), Individual Address (IA)

Test sequence:

Test application calls Falcon function 'IndividualAddressRead'

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0015 (Total Length)		0x04 (Struct - Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x29 (cEMI Service Code)	0x00 (additional info length)	0xB0 (Ctrl1)	0xE0 (Ctrl2)	IA (Source Address)	
0x0000 (Destination Address)		0x01	0x01	0x00			

Test Application sends a valid TUNNELLING_ACK

Cleanup: Close connection

5.2.10 Point-to-point telegram tunnelled to KNX and back

Function ID: 30213

Description: This test checks tunnelling of a point-to-point telegram to KNX subnet and back by sending a tunnelled MaskVersionRead request to the load switch and reading the tunnelled response of this device..

Expectation: The response telegram must contain the load switch address as source address.

Parameters: Device address of load switch (Type = String, Default = 1.1.50)

Out Parameters: None

Preparation: Test application sends a valid CONNECT_REQUEST (tunnelling, Data Link Layer, see 2.4.1)

BDUT sends a valid CONNECT_RESPONSE

Used results: Channel ID (CID), Individual Address (IA)

Test sequence:

Test application sends:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0014 (Total Length)		0x04 (Struct - Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x11 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	0x0000 (Source Address)	
0x1132 (Destination Address)		0x00	0x80				

BDUT sends a valid TUNNELLING_ACK

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0014 (Total Length)		0x04 (Struct - Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x2E (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	IA (Source Address)	
0x1132 (Destination Address)		0x00	0x80				

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the telegram is received via KNX bus connection:

`BC [ I A ] 11 32 50 80`

Test application sends:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0015 (Total Length)		0x04 (Struct - Length)	CID
0x01 (Sequence counter)	0x00 (reserved)	0x11 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	0x0000 (Source Address)	
0x1132 (Destination Address)		0x01	0x43	0x00			

BDUT sends a valid TUNNELLING_ACK

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0015 (Total Length)		0x04 (Struct - Length)	CID
0x01 (Sequence counter)	0x00 (reserved)	0x2E (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	IA (Source Address)	
0x1132 (Destination Address)		0x01	0x43	0x00			

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the following telegrams are received via KNX bus connection:

`BC [ I A ] 11 32 51 43 00`

B0 11 32 [I A] 60 C2

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)						
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0014 (Total Length)		0x04 (Struct - Length)	CID
0x02 (Sequence counter)	0x00 (reserved)	0x29 (cEMI Service Code)	0x00 (additional info length)	0xB0 (Ctrl1)	0x60 (Ctrl2)	0x1132 (Source Address)	
IA (Destination Address)		0x00	0xC2				

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the following telegram is received via KNX bus connection:

BC 11 32 [I A] 63 43 40 00 12

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)						
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0017 (Total Length)		0x04 (Struct - Length)	CID
0x03 (Sequence counter)	0x00 (reserved)	0x29 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x60 (Ctrl2)	0x1132 (Source Address)	
IA (Destination Address)		0x03	0x43	0x40	0x00	0x12	

Test Application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)						
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0014 (Total Length)		0x04 (Struct - Length)	CID
0x02 (Sequence counter)	0x00 (reserved)	0x11 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	0x0000 (Source Address)	
0x1132 (Destination Address)		0x00	0x81				

BDUT sends a valid TUNNELLING_ACK

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0014 (Total Length)		0x04 (Struct-Length)	CID
0x04 (Sequence counter)	0x00 (reserved)	0x2E (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0x50 (Ctrl2)	IA (Source Address)	
0x1132 (Destination Address)		0x00	0x81				

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the following telegram is received via KNX bus connection:

`BC [ I A ] 11 32 50 81`

Cleanup: Close connection

5.2.11 Group address telegram tunnelled to KNX

Function ID: 30215

Description: This test checks tunnelling of a Group Address telegram to KNX subnet.

Expectation: The test telegram should appear on the KNX connection, and the test device should send a confirmation telegram back to the tunnelling connection.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: Test application sends a valid CONNECT_REQUEST (tunnelling, Data Link Layer, see 2.4.1)

BDUT sends a valid CONNECT_RESPONSE

Used results: Channel ID (CID), Individual Address (IA)

Test sequence:

Test application sends:

UDP-frame (IP_TC, DPORT_TC, IP_BDUT, DPORT_BDUT)							
0x06 (Header-Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0018 (Total Length)		0x04 (Struct-Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x11 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0xC0 (Ctrl2)	0x0000 (Source Address)	
0x1234 (Destination Address)		0x04	0x00	0x80	0x56	0x78	0x9A

BDUT sends a valid TUNNELLING_ACK

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0018 (Total Length)		0x04 (Struct - Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x2E (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0xC0 (Ctrl2)	IA (Source Address)	
0x1234 (Destination Address)		0x04	0x00	0x80	0x56	0x78	0x9A

Test Application sends a valid TUNNELLING_ACK

Test Application checks if the telegram is received via KNX bus connection:

`BC [ I A ] 12 34 C4 00 80 56 78 9A`

Cleanup: Close connection

5.2.12 Group address telegram tunnelled from KNX

Function ID: 30216

Description: This test checks tunnelling of a Group Address telegram from KNX subnet.

Expectation: The test telegram should appear on the tunnelling connection.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: Test application sends a valid CONNECT_REQUEST (tunnelling, Data Link Layer, see 2.4.1)

BDUT sends a valid CONNECT_RESPONSE

Used results: Channel ID (CID)

Test sequence:

Test application sends a telegram (with SA = Source Address) via KNX bus connection:

`BC [ S A ] 12 34 C4 00 80 56 78 9A`

BDUT sends:

UDP-frame (IP_BDUT, DPORT_BDUT, IP_TC, DPORT_TC)							
0x06 (Header - Length)	0x10 (Ver.)	0x0420 (TUNNELLING_REQUEST)		0x0018 (Total Length)		0x04 (Struct - Length)	CID
0x00 (Sequence counter)	0x00 (reserved)	0x29 (cEMI Service Code)	0x00 (additional info length)	0xBC (Ctrl1)	0xC0 (Ctrl2)	SA (Source Address)	
0x1234 (Destination Address)		0x04	0x00	0x80	0x56	0x78	0x9A

Test Application sends a valid TUNNELLING_ACK

Cleanup: Close connection

5.3 Tunnel Addresses

5.3.1 Tunnel Addresses Standard Case

Function ID: 30110

Description: This test sets device property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES (list of tunnel addresses) to (FA, FA+1,..., FA+NA-1), where the first address FA is a parameter. Then as many as possible simultaneous tunnel connections are opened, and the returned tunnel addresses (KNX Individual Addresses in the tunnel connect response message) are checked.

Expectation: The returned tunnel addresses should be from the list defined by properties PID_KNX_INDIVIDUAL_ADDRESS and PID_ADDITIONAL_INDIVIDUAL_ADDRESSES, where PID_KNX_INDIVIDUAL_ADDRESS should not be used for tunnelling if the device is a router. Also, returned tunnel addresses must not be of the form x.y.0, even if specified so by device properties.

Parameters: First Tunnel Address (Type = String, Default = 1.1.111)

Out Parameters: None

Preparation:

Open Device Management Connection

Read number of elements of property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES = NA

Set NA additional Individual Addresses to FA, FA+1,..., FA+NA-1

Close Device Management Connection

Test sequence:

Open NA + 1 simultaneous tunnel connections (Data Link Layer) (see 2.4.1)

Used results: Channel ID (CID), Individual Address (IA)

BDUT sends NA responses for NA connect requests:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct Length)	0x01 (IPV4_UDP)	IP_BDUT		DPORT_BDUT	
0x04 (Struct Length)	0x04 (TUNNELLING_CONNECTION)	**IA**			

Test Application checks each returned IA in structure CRD in CONNECT_RESPONSE if it is contained in the list defined by properties PID_KNX_INDIVIDUAL_ADDRESS and PID_ADDITIONAL_INDIVIDUAL_ADDRESSES.

After the NA + 1 connect request BDUT sends:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total Length)	0x00	0x24 (NO_MORE_CONNECTIONS)

Cleanup:

Close all opened tunnel connections.

5.3.2 Tunnel Addresses Uniqueness

Function ID: 30111

Description: This test sets device property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES (list of tunnel addresses) to (TA, TA,..., TA), where the tunnel address TA is a parameter. Then as many as possible simultaneous tunnel connections are opened, and the returned tunnel addresses (KNX Individual Addresses in the tunnel connect response message) are checked.

Expectation: No more than one tunnel connection with address TA should open, even when specified multiply in PID_ADDITIONAL_INDIVIDUAL_ADDRESSES. If TA is of the form x.y.0, then the device should refuse to open a tunnel connection with address TA.

Parameters: Tunnel Address (Type = String, Default = 1.1.111)

Out Parameters: None

Preparation:

Open Device Management Connection

Read number of elements of property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES = NA

Set NA additional Individual Addresses to TA

Close Device Management Connection

Test sequence:

Open more than one simultaneous tunnel connections (Data Link Layer) (see 2.4.1)

Used results: Channel ID (CID)

BDUT sends response for the first connect request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct Length)	0x01 (IPV4_UDP)	IP_BDUT		DPORT_BDUT	
0x04 (Struct Length)	0x04 (TUNNELLING_CONNECTION)	**TA**			

BDUT sends response for the second connect request:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0008 (Total Length)	0x00	0x24 (NO_MORE_CONNECTIONS)

Cleanup:

Close all opened tunnel connections.

5.3.3 Tunnel Addresses Assignment Method

Function ID: 30112

Description: This test opens a tunnel connection and checks the returned tunnel address (RA1). Then it opens a second tunnel connection and checks the returned address (RA2) if possible. Then it closes the first tunnel connection. Then it opens a further tunnel connection and checks the returned address (RA3) if possible.

Expectation: RA3 should always equal RA1, as the device should always return the first free tunnel address value from its internal list.

Parameters: None

Out Parameters: None

Preparation: None

Test sequence:

Open first tunnel connection (Data Link Layer) (see 2.4.1)

Used results: RA1 (returned tunnel address)

Open second tunnel connection (Data Link Layer) (see 2.4.1)

Close first tunnel connection

Open tunnel connection (Data Link Layer) (see 2.4.1)

Used results: RA3 (returned tunnel address)

Test o.k. if RA1 == RA3

Cleanup:

Close all opened tunnel connections

5.3.4 Tunnel E_NO_MORE_CONNECTIONS

Function ID: 30113

Description: This test sets device property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES (list of tunnel addresses) to (FA, FA+1,..., FA+NA-1), where the first address FA is a parameter. Then as many as possible simultaneous tunnel connections are opened, and the returned tunnel addresses (KNX Individual Addresses in the Tunnel CONNECT_RESPONSE messages) and error code (in the last, negative Tunnel CONNECT_RESPONSE message) are checked.

Expectation: The Tunnel CONNECT_RESPONSE error code should be 0x24 = E_NO_MORE_CONNECTIONS, after having exhausted all specified tunnel addresses.

Parameters: First Tunnel Address (Type = String, Default = FA)

Out Parameters: None

Preparation:

Open Device Management Connection.

Read NA = number of elements of property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES.

Set the NA additional Individual Addresses to FA, FA+1,... , FA+NA-1.

Close Device Management Connection.

Test sequence:

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0014 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_BDUT		DPORT_BDUT
0x08 (Struct Length)	0x04 (Tunnel)	FA		

…

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends negative Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)			
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0008 (Total Length)
0x00	0x24 (E_NO_MORE_CONNECTIONS)		

Cleanup:

Close all opened tunnel connections.

5.3.5 Tunnel E_NO_MORE_UNIQUE_CONNECTIONS Case 1

Function ID: 30114

Description: This test sets via device property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES all tunnel addresses to the same value TA, which is a parameter. Then as many as possible simultaneous tunnel connections are opened, and the returned tunnel addresses (KNX Individual Addresses in the Tunnel CONNECT_RESPONSE messages) and error code (in the last, negative Tunnel CONNECT_RESPONSE message) are checked.

Expectation: The Tunnel CONNECT_RESPONSE error code should be

- 0x24 = E_NO_MORE_CONNECTIONS, if NA <= 1 and TA <> DA,
- 0x25 = E_NO_MORE_UNIQUE_CONNECTIONS, if NA > 1, or NA = 1 and TA = DA and DA can be tunnel address,

where NA = number of Additional Individual Addresses, and DA = KNX address of the BDUT.

Note that DA cannot be tunnel address if the BDUT supports KNXnet/IP Routing.

Parameters: Tunnel Address (Type = String, Default = TA)

Out Parameters: None

Preparation:

Open Device Management Connection.

Read NA = number of elements of property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES.

Set the NA additional Individual Addresses to TA, TA,...

Close Device Management Connection.

Test sequence:

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0014 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_BDUT		DPORT_BDUT
0x08 (Struct Length)	0x04 (Tunnel)	TA		

…

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends negative Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)			
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0008 (Total Length)
0x00	0x25 (E_NO_MORE_UNIQUE_CONNECTIONS)		

Cleanup:

Close all opened tunnel connections.

5.3.6 Tunnel E_NO_MORE_UNIQUE_CONNECTIONS Case 2

Function ID: 30115

Description: This test sets via device property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES all tunnel addresses to the same value DA, which is the regular KNX Individual Address of the BDUT. Then as many as possible simultaneous tunnel connections are opened, and the returned tunnel addresses (KNX Individual Addresses in the Tunnel CONNECT_RESPONSE messages) and error code (in the last, negative Tunnel CONNECT_RESPONSE message) are checked.

Expectation: The Tunnel CONNECT_RESPONSE error code should be

- 0x24 = E_NO_MORE_CONNECTIONS, if NA = 0 and DA can be tunnel address,
- 0x25 = E_NO_MORE_UNIQUE_CONNECTIONS, otherwise,

where NA = number of Additional Individual Addresses, and DA = KNX address of the BDUT.

Note that DA cannot be tunnel address if the BDUT supports KNXnet/IP Routing.

Parameters: None.

Out Parameters: None.

Preparation:

Open Device Management Connection.

Read NA = number of elements of property PID_ADDITIONAL_INDIVIDUAL_ADDRESSES.

Set the NA additional Individual Addresses to DA, DA,...

Close Device Management Connection.

Test sequence:

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0014 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_BDUT		DPORT_BDUT
0x08 (Struct Length)	0x04 (Tunnel)	TA		

…

TC sends Tunnel CONNECT_REQUEST:

UDP-frame (IP_TC, CPORT_TC -> IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Version)	0x0205 (CONNECT_ REQUEST)	0x001A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		DPORT_TC
0x08 (Struct Length)	0x04 (Tunnel)	0x02 (Link Layer)	0x00	

BDUT sends negative Tunnel CONNECT_RESPONSE:

UDP-frame (IP_BDUT, CPORT_BDUT -> IP_TC, CPORT_TC)			
0x06 (Header Length)	0x10 (Version)	0x0206 (CONNECT_ RESPONSE)	0x0008 (Total Length)
0x00	0x25 (E_NO_MORE_UNIQUE_CONNECTIONS)		

Cleanup:

Close all opened tunnel connections.

5.4 NAT Compatibility

5.4.1 Standard Case NAT Compatible Tunnelling to KNX

Function ID: 30301

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the standard behaviour by sending a NAT compatible tunnelling request message on the tunnel connection, which should produce a corresponding KNX bus telegram. NAT compatibility means that all HPAI fields in KNXnet/IP packets are set to zero, where the receiver of such packets implicitly takes the IP address information normally contained in the HPAI fields from the preceding IP telegram header.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the tunnelling request, the expected KNX bus telegram must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of control endpoint)			0x0000 (port number of control endpoint)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of data endpoint)			0x0000 (port number of data endpoint)	0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends a valid TUNNELLING_REQUEST with CID

BDUT sends a valid TUNNELLING_ACK

Test application checks if the telegram is received via KNX bus connection

Test application checks if the L_Data.con is received via tunnel connection

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of control endpoint)		0x0000 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

5.4.2 NAT compatible Tunneling to KNX with IP address set

Function ID: 30302

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the correct behavior by sending a NAT compatible tunneling request message on the tunnel connection, which should produce a corresponding KNX bus telegram. In this case, the HPAI fields containing the port number of control endpoint and data endpoint are both set to zero, while the HPAI fields containing the IP address of control endpoint and data endpoint are set to the real IP address.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the tunneling request, the expected KNX bus telegram must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x ac 18 f0 37 (ip address of control endpoint)			0x0000 (port number of control endpoint)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x ac 18 f0 37 (ip address of data endpoint)			0x0000 (port number of data endpoint)	0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends a valid TUNNELLING_REQUEST with CID

BDUT sends a valid TUNNELLING_ACK

Test application checks if the telegram is received via KNX bus connection

Test application checks if the L_Data.con is received via tunnel connection

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x ac 18 f0 37 (ip address of control endpoint)		0x0000 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

5.4.3 NAT compatible Tunneling to KNX with port number set

Function ID: 30303

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the correct behavior by sending a NAT compatible tunneling request message on the tunnel connection, which should produce a corresponding KNX bus telegram. In this case, the HPAI fields containing the IP address of control endpoint and data endpoint are both set to zero, while the HPAI fields containing the port number of control endpoint and data endpoint are set to the real port numbers.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the tunneling request, the expected KNX bus telegram must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of control endpoint)			0x05 f1 (port number of control endpoint)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of data endpoint)			0x05 f2 (port number of data endpoint)	0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends a valid TUNNELLING_REQUEST with CID

BDUT sends a valid TUNNELLING_ACK

Test application checks if the telegram is received via KNX bus connection

Test application checks if the L_Data.con is received via tunnel connection

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of control endpoint)		0x05 f1 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

5.4.4 Standard Case NAT Compatible Tunnelling from KNX

Function ID: 30304

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the standard behaviour by sending a group-addressed KNX bus telegram, which should produce a corresponding NAT compatible tunnelling request from the test device. NAT compatibility means that all HPAI fields in KNXnet/IP packets are set to zero, where the receiver of such packets implicitly takes the IP address information normally contained in the HPAI fields from the preceding IP telegram header.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the KNX bus telegram, the expected tunnelling request must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct-Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of control endpoint)			0x0000 (port number of control endpoint)	0x08 (Struct-Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of data endpoint)			0x0000 (port number of data endpoint)	0x04 (Struct-Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct-Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct-Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends telegram via KNX bus connection

Test application checks if BDUT sends a valid TUNNELLING_REQUEST

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of control endpoint)		0x0000 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

5.4.5 NAT compatible Tunneling from KNX with IP address set

Function ID: 30305

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the correct behavior by sending a group-addressed KNX bus telegram, which should produce a corresponding NAT compatible tunneling request from the test device. In the request message, the HPAI fields containing the IP address of control endpoint and data endpoint are both set to zero, while the HPAI fields containing the port number of control endpoint and data endpoint are set to the real port number.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the KNX bus telegram, the expected tunnelling request must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x ac 18 f0 37 (ip address of control endpoint)			0x0000 (port number of control endpoint)	0x08 (Struct - Length)	0x01 (IPV4_UDP)
0x ac 18 f0 37 (ip address of data endpoint)			0x0000 (port number of data endpoint)	0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header - Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct - Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends telegram via KNX bus connection

Test application checks if BDUT sends a valid TUNNELLING_REQUEST

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct - Length)	0x01 (IPV4_UDP)	0x ac 18 f0 37 (ip address of control endpoint)		0x0000 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

5.4.6 NAT compatible Tunneling from KNX with port number set

Function ID: 30306

Description: This test opens a NAT compatible tunnel connection and a KNX bus connection, and verifies the correct behavior by sending a group-addressed KNX bus telegram, which should produce a corresponding NAT compatible tunneling request from the test device. In this case, NAT compatibility means that in KNXnet/IP packets, the HPAI field containing the IP address is set to zero, while the HPAI field containing the port number is set to the real port number.

Expectation: The test device should respond with NAT compatible telegrams, and as a result of the KNX bus telegram, the expected tunnelling request must appear.

Parameters: Falcon connection parameters

Out Parameters: None

Preparation: None

Test sequence:

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0205 (CONNECT_REQUEST)	0x001A (Total Length)	0x08 (Struct-Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of control endpoint)			0x0621 (port number of control endpoint)	0x08 (Struct-Length)	0x01 (IPV4_UDP)
0x00000000 (ip address of data endpoint)			0x0622 (port number of data endpoint)	0x04 (Struct-Length)	0x04 (TUNNEL_CONNECTION)
0x02 (TUNNEL_LINKLAYER)	0x00 (reserved)				

BDUT sends response:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)					
0x06 (Header-Length)	0x10 (Ver.)	0x0206 (CONNECT_RESPONSE)	0x0014 (Total Length)	CID	0x00 (NO_ERROR)
0x08 (Struct-Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of data endpoint)		0x0000 (port number of data endpoint)	
0x04 (Struct-Length)	0x04 (TUNNEL_CONNECTION)	XXXX (Individual Address)			

Used results: Channel ID (CID)

Test application sends telegram via KNX bus connection

Test application checks if BDUT sends a valid TUNNELLING_REQUEST

Test application sends a valid TUNNELLING_ACK

Test application sends:

UDP-frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0209 (DISCONNECT_REQUEST)	0x0010 (Total Length)	CID	0x00 (reserved)
0x08 (Struct-Length)	0x01 (IPV4_UDP)	0x00000000 (ip address of control endpoint)		0x0621 (port number of control endpoint)	

BDUT sends a valid DISCONNECT_RESPONSE

Cleanup: None

6 Routing

6.1 Routing Indication

6.1.1 Standard Case 1

Function ID: 40201

Description: This test verifies the standard behaviour by sending a valid KNXnet/IP routing indication on the standard multicast address

Expectation: The telegram must be received over an open Falcon KNX connection.

Parameters: Falcon connection parameters.

Preparation: none

Test sequence:

Send KNX**net/IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 78 9a					

Expected Answer must be received not later than x s after sending the KNX/IP Routing

KNX telegram:

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

Cleanup: none

6.1.2 Standard Case 2

Function ID: 40202

Description: This test verifies the standard behaviour by sending a group data message over a Falcon KNX connection.

Expectation: The telegram must be received over KNXnet/IP routing.

Parameters: Falcon connection parameters.

Preparation: none

Test sequence:

Send group data message over a Falcon KNX connection:

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

Expected KNXnet/IP routing:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc b0 11 fd 12 34 04 00 80 56 78 9a					

Cleanup: none

6.1.3 Changed multicast address, Case 1

Function ID: 40203

Description: This test changes the routing multicast address and verifies the standard behaviour by sending a valid KNXnet/IP routing indication on the changed multicast address.

Expectation: The telegram must be received over an open Falcon KNX connection.

Parameters: Falcon connection parameters.

Preparation:

Send Connection Request

Send Configuration Request:

M_PropWrite.req, IOT=11, OI=1, PID=66, NoE=1, SIx=1

Data: $ EF C0 27 ED

Send Device Configuration Request:

M_Reset.req

Close Connection

Test sequence:

Send **KNXnet/IP routing**:

UDP-frame (IP_BDUT, CPORT_BDUT, **IP_ ROUTING (EF C0 27 ED)**, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 78 9a					

Expected Answer must be received not later than x s after sending the KNX/IP Routing:

KNX telegram:

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

Cleanup:

Send Connection Request

Send Configuration Request:

M_PropWrite.req, IOT=11, OI=1, PID=66, NoE=1, SIx=1

Data: $ E0 00 17 0C

Send Device Configuration Request:

M_Reset.req

Close Connection

6.1.4 Changed multicast address, Case 2

Function ID: 40204

Description: This test changes the routing multicast address and verifies the standard behaviour by sending a group data message over a Falcon KNX connection.

Expectation: The telegram must be received over KNXnet/IP routing.

Parameters: Falcon connection parameters.

Preparation:

Send Connection Request

Send Configuration Request:

M_PropWrite.req, IOT=11, OI=1, PID=66, NoE=1, SIx=1

Data: $ EF C0 27 ED

Send Device Configuration Request:

M_Reset.req

Close Connection

Test sequence:

Send group data message over a Falcon KNX connection:

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

Expected KNXnet/IP routing:

UDP-frame (IP_BDUT, CPORT_BDUT, **IP_ ROUTING (EF C0 27 ED)**, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data.ind	0x00 Additional Info
bc b0 11 fd 12 34 04 00 80 56 78 9a					

Cleanup:

Send Connection Request

Send Configuration Request:

M_PropWrite.req, IOT=11, OI=1, PID=66, NoE=1, SIx=1

Data: $ E0 00 17 0C

Send Device Configuration Request:

M_Reset.req

Close Connection

6.1.5 Property PID_MSG_TRANSMIT_TO_KNX

Optional test, only needed if PID_MSG_TRANSMIT_TO_KNX is implemented

Function ID: 40205

Description: This test checks the counter PID_MSG_TRANSMIT_TO_KNX before and after routing 5 routed and 3 non-routed group data messages to KNX subnet.

Expectation: The counter should be 0 before and increased by 5 after sending the test telegrams.

Parameters: Max number of retrials to reopen connection after resetting device, Waiting time in milliseconds after resetting device, Falcon connection parameters

Preparation:

Send DeviceConnection Request

Send Device Configuration Request:

M_Reset.req

Close Connection

Test sequence:

Send DeviceConnection Request

Send Device Configuration Request: M_PropRead.req

UDP-frame (IP_TC, CPORT_TC, IP_ BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFC M_PropRead.req	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1						

Expected Answer: Device Configuration Response

Expected Answer: Device Configuration Request: M_PropRead.conf

UDP-frame (IP_ BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0015 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFB M_PropRead.conf	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1	**0x00000000 Data**					

Close Connection

Send 5 **KNXnet/IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 78 9a					

Expected Answer:

Received KNX telegram:

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

Send 3 **KNXnet/IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc 80 00 00 12 34 04 00 80 56 78 9a					

Send DeviceConnection Request

Send Device Configuration Request: M_PropRead.req

UDP-frame (IP_TC, CPORT_TC, IP_ BDUT, CPORT_BDUT)						
0x06 (Header - Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFC M_PropRead.req	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1						

Expected Answer: Device Configuration Response

Expected Answer: Device Configuration Request: M_PropRead.conf

UDP-frame (IP_ BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0015 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFB M_PropRead.conf	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1	**0x00000005 Data**					

Close Connection

Cleanup: none

6.1.6 Property PID_MSG_TRANSMIT_TO_IP

Optional test, only needed if PID_MSG_TRANSMIT_TO_IP is implemented

Function ID: 40206

Description: This test checks the counter PID_MSG_TRANSMIT_TO_IP before and after routing 5 routed and 3 non-routed group data messages from a Falcon KNX connection.

Expectation: The counter should be 0 before and increased by 5 after sending the test telegrams

Parameters: Max number of retrials to reopen connection after resetting device, Waiting time in milliseconds after resetting device, Falcon connection parameters

Preparation:

Send DeviceConnection Request

Send Device Configuration Request:

M_Reset.req

Close Connection

Test sequence:

Send DeviceConnection Request

Send Device Configuration Request: M_PropRead.req

UDP-frame (IP_TC, CPORT_TC, IP_ BDUT, CPORT_BDUT)						
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFC M_PropRead.req	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1						

Expected Answer: Device Configuration Response

Expected Answer: Device Configuration Request: M_PropRead.conf

UDP-frame (IP_ BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header-Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0015 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFB M_PropRead.conf	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1	**0x00000000 Data**					

Close Connection

Send 5 KNX telegrams:

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

Send 3 KNX telegrams:

BC 11 FD 12 34 84 00 80 56 78 9A :39

CC

BC 11 FD 12 34 84 00 80 56 78 9A :39

CC

BC 11 FD 12 34 84 00 80 56 78 9A :39

CC

Expected Answer:

Receive 5 KNX**net/IP routing**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header-Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data.ind	0x00 Additional Info
bc b0 11 fd 12 34 04 00 80 56 78 9a					

Send DeviceConnection Request

Send Device Configuration Request: M_PropRead.req

UDP-frame (IP_TC, CPORT_TC, IP_ BDUT, CPORT_BDUT)						
0x06 (Header - Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0011 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFC M_PropRead.req	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1						

Expected Answer: Device Configuration Response

Expected Answer: Device Configuration Request: M_PropRead.conf

UDP-frame (IP_ BDUT, CPORT_BDUT, IP_TC, CPORT_TC)						
0x06 (Header - Length)	0x10 (Ver.)	0x0310 (DEVICE_CONFIGURATION_REQUEST)		0x0015 (Total Length)	0x04 structure length	x Channel ID
0x00 Sequence counter	0x00 reserved	0xFB M_PropRead.conf	0x000B ObjectType	0x01 Instance	0x4B PID	0x10 NoE=1
0x01 Six=1	**0x00000005 Data**					

Close Connection

Cleanup: none

6.1.7 Mixed Case 1

Function ID: 30201

Description: This test verifies the standard behaviour by sending a valid tunnelling request message on a KNX Link Layer Tunnelling Connection. The tunnelling message contains an L_Data.req service with fixed data.

Expectation: The telegram must be received both over KNXnet/IP routing and a local Falcon KNX connection.

Parameters: Falcon connection parameters

Preparation: none

Test sequence:

Send Tunnelling Connection Request

Send a Tunnelling Request: L_DATA.req 11 00 bc c0 00 00 12 34 04 00 80 56 78 9a

Expected Answer:

Receive **KNXnet/IP routing**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data.ind	0x00 Additional Info
bc b0 11 e8 12 34 04 00 80 56 78 9a					

Receive KNX telegram:

BC 11 E8 12 34 C4 00 80 56 78 9A :6C

CC

Receive Tunnelling Request: L_DATA.conf 2e 00 bc c0 11 e8 12 34 04 00 80 56 78 9a

Close Connection

Cleanup: none

6.1.8 Mixed Case 2

Function ID: 30202

Description: This test verifies the standard behaviour by sending a valid tunnelling request message over a local Falcon KNX connection, while a KNX Link Layer Tunnelling Connection is open. The Falcon KNX message contains an L_Data.req service with fixed data.

Expectation: The telegram must be received both over KNXnet/IP routing and the KNX Tunnelling connection.

Parameters: Falcon connection parameters

Preparation: none

Test sequence:

Send Tunnelling Connection Request

Send KNX telegram

BC 11 FD 12 34 C4 00 80 56 78 9A :79

CC

Expected Answer:

Receive Tunnelling Request: L_DATA.ind 29 00 bc c0 11 fd 12 34 04 00 80 56 78 9a

Receive KNX**net/IP routing**:

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data.ind	0x00 Additional Info
bc b0 11 fd 12 34 04 00 80 56 78 9a					

Close Connection

Cleanup: none

6.1.9 Mixed Case 3

Function ID: 30203

Description: This test verifies the standard behaviour by sending a valid tunnelling request message over KNXnet/IP routing, while a KNX Link Layer Tunnelling Connection and a Falcon KNX connection is open. The KNXnet/IP routing message contains an L_Data.ind service with fixed data.

Expectation: The telegram must be received both over the local Falcon KNX connection and the KNX Tunnelling connection.

Parameters: Falcon connection parameters

Preparation: none

Test sequence:

Send Tunnelling Connection Request

Send KNX**net/IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data.ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 78 9a					

Expected Answer:

Receive Tunnelling Request: L_DATA.ind 29 00 bc b0 00 00 12 34 04 00 80 56 78 9a

Receive KNX telegram

BC 00 00 12 34 B4 00 80 56 78 9A :E5

CC

Close Connection

Cleanup: none

6.2 Routing Lost Message

6.2.1 Standard Case

Function ID: 40101

Description: This test verifies the standard behaviour by sending a large number of KNXnet/IP routing indications in a short period of time, stopping if a routing lost message indication has been received.

Expectation: After a certain number of routing indications, the test device will lose one or more telegrams and transmit a Routing Lost Message Indication.

Parameters: Max. number of Routing Indications to transmit (Type = Integer, Default = 1000), Wait time in milliseconds if no lost message indication has been received (Type = Integer, Default = 10), Falcon connection parameters

Preparation: none

Test sequence:

Send in short period of time n KNXnet/**IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 ss ss (ss ss = a sequencecounter start with 0, increment per 1)					

Expected Answer:

Receive n-1 KNX telegrams

BC 00 00 12 34 B4 00 80 56 00 00

CC

BC 00 00 12 34 B4 00 80 56 00 01

CC

…

BC 00 00 12 34 B4 00 80 56 n-1

CC

Receive Routing Lost Message

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0531 (ROUTING_LOST_MESSAGE)	0x000A (Total Length)	0x04 structure length	0x00 device state
0x0001 Number of lost messages					

Cleanup: none

6.2.2 Continuous overflow

Function ID: 40102

Description: This test verifies the standard behaviour by sending a large number of KNXnet/IP routing indications in a short period of time. Unlike the previous test, the tool does not stop sending after receiving the routing lost message.

Expectation: After a certain number of routing indications, the test device might lose one or more telegrams and transmit Routing Lost Message Indications. The added number of lost and routed messages should be the total number of messages transmitted. The time between subsequent lost message indications should be about one second.

Parameters: Max. number of Routing Indications to transmit (Type = Integer, Default = 1000), Wait time in milliseconds if no lost message indication has been received (Type = Integer, Default = 10), Falcon connection parameters

Preparation: none

Test sequence:

Send in a short period of time m * n KNXnet/**IP routing**:

UDP-frame (IP_TC, CPORT_TC, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0530 (ROUTING_INDICATION)	0x0014 (Total Length)	0x29 L_data. ind	0x00 Additional Info
bc c0 00 00 12 34 04 00 80 56 ss ss (ss ss = a sequencecounter (start with 0, increment per 1)					

Expected Answer:

Receive m*(n-1) KNX telegrams, m = 1

BC 00 00 12 34 B4 00 80 56 00 00

CC

BC 00 00 12 34 B4 00 80 56 00 01

CC

…

BC 00 00 12 34 B4 00 80 56 m*(n-1)

CC

Receive Routing Lost Message

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0531 (ROUTING_LOST_MESSAGE)	0x000A (Total Length)	0x04 structure length	0x00 device state
0x0001 Number of lost messages					

Receive m*(n-1) KNX telegrams, m = 2

BC 00 00 12 34 B4 00 80 56 m*n + 0

CC

BC 00 00 12 34 B4 00 80 56 m*n + 1

CC

…

BC 00 00 12 34 B4 00 80 56 m*n - nL

CC

Receive Routing Lost Message

UDP-frame (IP_BDUT, CPORT_BDUT, IP_ ROUTING, CPORT_BDUT)					
0x06 (Header - Length)	0x10 (Ver.)	0x0531 (ROUTING_LOST_MESSAGE)	0x000A (Total Length)	0x04 structure length	0x00 device state
nL Number of lost messages					

Cleanup: none

7 Remote Diagnosis and Configuration

7.1 General

This test subclass is concerned with KNX/IP Service Family “Remote Diagnosis And Configuration” in general.

7.1.1 Supported Service Family

Function ID: 50101

Description: This test checks whether “Remote Diagnosis and Configuration” is contained in “Supported Service Families” reported by the target device via Discovery SEARCH_RESPONSE.

Expectation: “Remote Diagnosis and Configuration” (Service Family 7, Version 1) is contained in “Supported Service Families” reported by the target device via Discovery SEARCH_RESPONSE.

Parameters: None.

Preparation: None.

Test sequence:

TC sends SEARCH_REQUEST:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)					
0x06 (Header Length)	0x10 (Ver.)	0x0201 (SEARCH_REQUEST)	0x000E (Total length)	0x08 (Struct length)	0x01 (IPV4_ UDP)
IP_TC			CPORT_TC		

BDUT sends SEARCH_RESPONSE:

UDP frame (IP_BDUT : CPORT_BDUT, IP_TC, CPORT_TC)							
0x06 (Header Length)	0x10 (Ver.)	0x0202 (SEARCH_RESPONSE)		0x0052 (Total Length)			
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC				CPORT_TC	
0x36 (Struct Length)	0x01 (Device Info DIB)	...					
0x0C (Struct Length)	0x02 (Supp Svc Fams DIB)	X	X	X	X	X	X
X	X	**0x07 (Remote Diag And Config)**	0x01 (V1)				

Cleanup: None.

Remark: for this test, it is only relevant that the BDUT sends a search response at least indicating support of remote diagnostics and configuration (all other supported service families are irrelevant).

7.1.2 Illegal Service Code

Function ID: 50102

Description: This test sends a RemoteDiagAndConfig datagram with an illegal Service Code, i.e. a KNX/IP datagram with undefined Service Code in the range 0740h...07FFh (these are reserved for RemoteDiagAndConfig; 0744h…07FFh are undefined). The test datagram is constructed by modifying the KNX/IP Service Code of a REMOTE_DIAGNOSTIC_REQUEST datagram selecting the target device by MAC address.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: Illegal Service Code (Type = String, Default = 0x0744).

Preparation: None.

Test sequence:

TC sends RemoteDiagAndConfig datagram with illegal ServiceCode:

UDP frame (IP_TC : CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	**0x0744 (illegal Service Code)**	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.1.3 Illegal Header Length

Function ID: 50103

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with illegal KNX/IP Header Length (not 0x06). The test datagram is constructed by modifying the Header Length value in the KNX/IP header of a REMOTE_DIAGNOSTIC_REQUEST datagram selecting the target device by MAC address.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: Illegal Header Length (Type = Byte, Default = 0x05).

Preparation: None.

Test sequence:

TC sends RemoteDiagAndConfig with illegal HeaderLength:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x05 (illegal Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.1.4 Illegal Protocol Version

Function ID: 50104

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with illegal KNX/IP Protocol Version (not 0x10). The test datagram is constructed by modifying the Protocol Version value in the KNX/IP header of a REMOTE_DIAGNOSTIC_REQUEST datagram selecting the target device by MAC address.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: Illegal Protocol Version (Type = Byte, Default = 0x11).

Preparation: None.

Test sequence:

TC sends RemoteDiagAndConfig with illegal HeaderLength:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	**0x11 (illegal Version)**	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.2 REMOTE_DIAGNOSTIC_REQUEST

This test subclass sends REMOTE_DIAGNOSTIC_REQUEST datagrams.

Unless specified otherwise, the TC sends these test datagrams with MAC SELECTOR to the BDUT.

7.2.1 Illegal Total Length

Function ID: 50201

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with illegal KNX/IP Total Length. The test datagram is constructed by modifying the Total Length value in the KNX/IP header of a REMOTE_DIAGNOSTIC_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: Illegal Total Length (Type = String, Default = 0x0018).

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST with illegal TotalLength:

UDP frame (IP_, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	**0x0018 (illegal Total Length)**	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.2.2 Missing HPAI

Function ID: 50202

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST without Discovery Endpoint HPAI. The test datagram is constructed by omitting the HPAI field in a REMOTE_DIAGNOSTIC_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST without HPAI:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x000E (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.2.3 Missing Selector

Function ID: 50203

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST without SELECTOR. The test datagram is constructed by omitting the SELECTOR field in a REMOTE_DIAGNOSTIC_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST without SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x000E (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC

BDUT sends no response.

Cleanup: None.

7.2.4 Illegal Selector

Function ID: 50204

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with illegal SELECTOR. The test datagram is constructed by modifying the Selection Code (2nd byte) value in a PrgMode SELECTOR (without MAC address field) or MAC SELECTOR (with MAC address field) in a REMOTE_DIAGNOSTIC_REQUEST datagram.

Expectation: Target device ignores these invalid datagrams, and sends no response.

Parameters: Illegal Selection Code (Type = Byte, Default = 0x00).

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST with illegal PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	**0x00 (ill. Sel.)**			

BDUT sends no response.

TC sends REMOTE_DIAGNOSTIC_REQUEST with illegal MAC SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	**0x00 (ill. Sel.)**	MAC_BDUT		

BDUT sends no response.

Cleanup: None.

7.2.5 Selection by Programming Mode

Function ID: 50205

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR. First, BDUT is set to PrgMode=1 via KNX/IP Device Management, then TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR, then BDUT is set to PrgMode=0 via KNX/IP Device Management, then TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE if and only if it is in Programming Mode.

Parameters: None.

Preparation: None.

Test sequence:

TC sets BDUT to PrgMode=1 via KNX/IP Device Management.

TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	0x01 (PrgMode Sel.)			

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with PrgMode SELECTOR and all of its supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)			
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)
0x08 (Struct Length)	0x01 (PrgMode Sel.)		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...	
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...	
...			

TC sets BDUT to PrgMode=0 via KNX/IP Device Management.

TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	0x01 (PrgMode Sel.)			

BDUT sends no response.

Cleanup: None.

7.2.6 Selection by MAC Address

Function ID: 50206

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR. First, TC sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR with correct MAC Address, and then TC sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR with wrong MAC Address (obtained by inverting all bits).

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE if and only if its MAC address matches.

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR with correct MAC Address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with MAC SELECTOR and all of its supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

TC sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR with wrong MAC Address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	(wrong MAC Address)		

BDUT sends no response.

Cleanup: None.

7.2.7 Request via Broadcast

Function ID: 50207

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST via Broadcast (destination address 255.255.255.255 instead of multicast address 224.0.23.12).

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE.

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST via Broadcast:

UDP frame (IP_TC, CPORT_TC -> **255.255.255.255, CPORT_BDUT**)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE and all of its supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

Cleanup: None.

7.3 REMOTE_DIAGNOSTIC_RESPONSE

This test subclass is concerned with REMOTE_DIAGNOSTIC_RESPONSE datagrams.

7.3.1 Spontaneous REMOTE_DIAGNOSTIC_RESPONSE

Function ID: 50301

Description: This test sends REMOTE_DIAGNOSTIC_RESPONSE to the Control Endpoint of the target device.

Expectation: Target device ignores this, and sends no response.

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_RESPONSE to BDUT:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0x0C (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

BDUT sends no response.

Cleanup: None.

7.3.2 Supported DIBs

Function ID: 50302

Description: This test sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR.

Expectation: Target device responds with a REMOTE_DIAGNOSTIC_RESPONSE datagram containing all supported DIBs (mandatory: IpConfig, IpCurConfig, KnxAddresses[2]).

Parameters: None.

Preparation: None.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST with MAC SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0740 (REMOTE_ DIAG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with MAC SELECTOR and all of its supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	**0x02 (MAC Sel.)**	MAC_BDUT		
0x10 (Struct Length)	**0x03 (IpConfig DIB)**	...		
0x14 (Struct Length)	**0x04 (IpCurConfig DIB)**	...		
0xXX (Struct Length)	**0x05 (KnxAddresses DIB)**	...		
xxx	**xxx**	…		

Cleanup: None.

[2] Mandatory DIBs: subject to final approval by TF IP (currently 0413 pending).

7.4 REMOTE_BASIC_CONFIGURATION_REQUEST

This test subclass sends REMOTE_BASIC_CONFIGURATION_REQUEST datagrams.

Unless specified otherwise, the TC sends these test datagrams with MAC SELECTOR to the BDUT.

Unless specified otherwise, these test datagrams contain an IpConfig DIB as found in the response to a previously sent REMOTE_DIAGNOSTIC_REQUEST.

In all tests here, the TC reads out the configuration via REMOTE_DIAGNOSTIC_REQUEST in the preparation phase, and restores the old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST in the cleanup phase.

7.4.1 Illegal Total Length

Function ID: 50401

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with illegal KNX/IP Total Length. The test datagram is constructed by modifying the Total Length value in the KNX/IP header of a REMOTE_BASIC_CONFIGURATION_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: Illegal Total Length (Type = String, Default = 0x0028).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with illegal TotalLength:

<table>
<tr><td colspan="5">UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)</td></tr>
<tr><td>0x06
(Header Length)</td><td>0x10
(Ver.)</td><td>0x0742
(REMOTE_ CONFIG_REQUEST)</td><td>0x0028
(illegal Total Length)</td><td></td></tr>
<tr><td>0x08
(Struct Length)</td><td>0x01
(IPV4_ UDP)</td><td colspan="2">IP_TC</td><td>CPORT_TC</td></tr>
<tr><td>0x08
(Struct Length)</td><td>0x02
(MAC Sel.)</td><td colspan="3">MAC_BDUT</td></tr>
<tr><td>0x10
(Struct Length)</td><td>0x03
(IpConfig DIB)</td><td colspan="3">...</td></tr>
</table>

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.2 Missing HPAI

Function ID: 50402

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST without Discovery Endpoint HPAI. The test datagram is constructed by omitting the HPAI field in a REMOTE_BASIC_CONFIGURATION_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_DIAGNOSTIC_REQUEST without HPAI:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x001E (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.3 Missing Selector

Function ID: 50403

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST without SELECTOR. The test datagram is constructed by omitting the SELECTOR field in a REMOTE_BASIC_CONFIGURATION_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and sends no response.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST without SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x001E (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.4 Illegal Selector

Function ID: 50404

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with illegal SELECTOR. The test datagram is constructed by modifying the Selection Code (2nd byte) value in a PrgMode SELECTOR (without MAC address field) or MAC SELECTOR (with MAC address field) in a REMOTE_BASIC_CONFIGURATION_REQUEST datagram.

Expectation: Target device ignores these invalid datagrams, and sends no response.

Parameters: Illegal Selection Code (Type = Byte, Default = 0x00).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with illegal PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0020 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	**0x00 (ill. Sel.)**			
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with illegal MAC SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	**0x00 (ill. Sel.)**	MAC_BDUT		
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.5 Selection by Programming Mode

Function ID: 50405

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with PrgMode SELECTOR. First, BDUT is set to PrgMode=1 via KNX/IP Device Management, then TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with PrgMode SELECTOR, then BDUT is set to PrgMode=0 via KNX/IP Device Management, then TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with PrgMode SELECTOR.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE if and only if it is in Programming Mode.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sets BDUT to PrgMode=1 via KNX/IP Device Management.

TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0020 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	0x01 (PrgMode Sel.)			
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with PrgMode SELECTOR and all supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)			
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)
0x08 (Struct Length)	0x01 (PrgMode Sel.)		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...	
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...	
...			

TC sets BDUT to PrgMode=0 via KNX/IP Device Management.

TC sends REMOTE_DIAGNOSTIC_REQUEST with PrgMode SELECTOR:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0020 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x02 (Struct Length)	0x01 (PrgMode Sel.)			
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.6 Selection by MAC Address

Function ID: 50406

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with MAC SELECTOR. First, TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with MAC SELECTOR with correct MAC Address, then TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with MAC SELECTOR with wrong MAC Address (obtained by inverting all bits).

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE if and only if its MAC address matches.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with MAC SELECTOR with correct MAC Address:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with MAC SELECTOR and all supported DIBs:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with MAC SELECTOR with wrong MAC Address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	(wrong MAC Address)		
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT sends no response.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.7 Missing DIB

Function ID: 50407

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST without DIBs.

Expectation: Target device may or may not respond with REMOTE_DIAGNOSTIC_RESPONSE.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST without DIBs:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0016 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT may respond by sending REMOTE_DIAGNOSTIC_RESPONSE:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.8 Unknown DIB

Function ID: 50408

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with a DIB of unknown type.

Expectation: Target device may or may not respond with REMOTE_DIAGNOSTIC_RESPONSE.

Parameters: Unknown Description Type Code (Type = Byte, Default = 0x00).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with unknown DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x10 (Struct Length)	**0x00 (unknown DIB)**	...		

BDUT may respond by sending REMOTE_DIAGNOSTIC_RESPONSE:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.9 Device Information DIB

Note: this test can only be performed if the Device Information DIB is supported by the BDUT

Function ID: 50409

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with DeviceInfo DIB specifying new (inverted) PrgMode.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE with DeviceInfo DIB specifying new PrgMode.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with new DeviceInfo DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x004C (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	**(new Programming Mode)**		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with new DeviceInfo DIB:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	**(new Programming Mode)**		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.10 Supported Service Families DIB

Note: this test can only be performed if the Supported Service Families DIB is supported by the BDUT

Function ID: 50410

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with new (empty) SupportedServiceFamilies DIB.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE with old (unchangeable) SupportedServiceFamilies DIB.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with new (empty) SupportedServiceFamilies DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0018 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x02 (Struct Length)	0x02 (SuppSvcFams DIB)			

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with old (unchanged) SupportedServiceFamilies DIB:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0x0C (Struct Length)	0x02 (SuppSvcFams DIB)	**(old Supported Service Families)**		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.11 IP Configuration DIB

Function ID: 50411

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with IpConfig DIB specifying new IP Address.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE with IpConfig DIB specifying new IP Address.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with new IpConfig DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x10 (Struct Length)	0x03 (IpConfig DIB)	**(new IP Address)**		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with new IpConfig DIB:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
...				
0x10 (Struct Length)	0x03 (IpConfig DIB)	**(new IP Address)**		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.12 IP Current Configuration DIB

Function ID: 50412

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with new (zero-valued) IpCurConfig DIB.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE with old (unchangeable) IpCurConfig DIB.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with new IpCurConfig DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x002A (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x14 (Struct Length)	0x04 (IpCurConfig DIB)	**(all values zero)**		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with old (unchanged) IpCurConfig DIB:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
...				
0x14 (Struct Length)	0x04 (IpCurConfig DIB)	**(old values)**		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.13 KNX Addresses DIB

Function ID: 50413

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with KnxAddresses DIB specifying new (main) KNX Address, obtained by incrementing Line Address.

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE with KnxAddresses DIB specifying new KNX Address.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST with new KnxAddresses DIB:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x00XX (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0xXX (Struct Length)	0x05 (KnxAddresses DIB)	**(new KNX Address)**		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE with new KnxAddresses DIB:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
...				
0xXX (Struct Length)	0x05 (KnxAddresses DIB)	**(new KNX Address)**		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.4.14 Request via Broadcast

Function ID: 50415

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST via Broadcast (destination address 255.255.255.255 instead of multicast address 224.0.23.12).

Expectation: Target device responds with REMOTE_DIAGNOSTIC_RESPONSE.

Parameters: None.

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_BASIC_CONFIGURATION_REQUEST via Broadcast:

UDP frame (IP_TC,CPORT_TC -> **255.255.255.255** : CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0742 (REMOTE_ CONFIG_REQUEST)	0x0026 (Total Length)	
0x08 (Struct Length)	0x01 (IPV4_ UDP)	IP_TC		CPORT_TC
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x10 (Struct Length)	0x03 (IpConfig DIB)	...		

BDUT responds by sending REMOTE_DIAGNOSTIC_RESPONSE:

UDP frame (IP_BDUT, CPORT_BDUT, IP_TC, CPORT_TC)				
0x06 (Header Length)	0x10 Ver.)	0x0741 (REMOTE_ DIAG_RESPONSE)	0x00XX (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x36 (Struct Length)	0x01 (DeviceInfo DIB)	...		
0xXX (Struct Length)	0x02 (SuppSvcFams DIB)	...		
...				

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5 REMOTE_RESET_REQUEST

This test subclass sends REMOTE_RESET_REQUEST datagrams.

Unless specified otherwise, the TC sends these test datagrams with MAC SELECTOR to the BDUT.

Unless specified otherwise, these test datagrams specify Restart (Reset Mode = Soft).

The BDUT should react here by performing the restart, without sending a response.

The TC tests this behavior by sending REMOTE_DIAGNOSTIC_REQUEST (with MAC SELECTOR) shortly after the REMOTE_RESET_REQUEST. Receiving REMOTE_DIAGNOSTIC_RESPONSE from the BDUT here means failure, as we expect that during a certain "dead time" (restarting phase) the BDUT will not respond.

In the cleanup phase, the TC repeatedly (every second) sends REMOTE_DIAGNOSTIC_REQUEST until the BDUT responds with REMOTE_DIAGNOSTIC_RESPONSE. If nothing happens after some time (configurable, default = 3 minutes), then the test will be aborted with fatal error message.

In all tests here, the TC reads out the configuration via REMOTE_DIAGNOSTIC_REQUEST in the preparation phase, and restores the old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST at the end of the cleanup phase.

7.5.1 Illegal Total Length

Function ID: 50501

Description: This test sends REMOTE_RESET_REQUEST with illegal KNX/IP Total Length. The test datagram is constructed by modifying the Total Length value in the KNX/IP header of a REMOTE_RESET_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and does not restart.

Parameters: Illegal Total Length (Type = String, Default = 0x0012); Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST with illegal TotalLength:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	**0x0012 (illegal Total Length)**	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.2 Missing Selector

Function ID: 50502

Description: This test sends REMOTE_RESET_REQUEST without SELECTOR. The test datagram is constructed by omitting the SELECTOR field in a REMOTE_RESET_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and does not restart.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST without SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)			
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0008 (Total Length)
0x01 (Restart)	0x00 (reserved)		

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.3 Illegal Selector

Function ID: 50503

Description: This test sends REMOTE_RESET_REQUEST with illegal SELECTOR. The test datagram is constructed by modifying the Selection Code (2nd byte) value in a PrgMode SELECTOR (without MAC address field) or MAC SELECTOR (with MAC address field) in a REMOTE_RESET_REQUEST datagram.

Expectation: Target device ignores this invalid datagram, and does not restart.

Parameters: Illegal Selection Code (Type = Byte, Default = 0x00); Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

TC sends REMOTE_RESET_REQUEST with illegal PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)			
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x000A (Total Length)
0x02 (Struct Length)	**0x00 (ill. Sel.)**		
0x01 (Restart)	0x00 (reserved)		

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

TC sends REMOTE_RESET_REQUEST with illegal MAC SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	**0x00 (ill. Sel.)**	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT does not restart.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.4 Selection by Programming Mode

Function ID: 50504

Description: This test sends REMOTE_RESET_REQUEST with PrgMode SELECTOR. First, BDUT is set to PrgMode=1 via KNX/IP Device Management, then TC sends REMOTE_RESET_REQUEST with PrgMode SELECTOR, then BDUT is set to PrgMode=0 via KNX/IP Device Management, then TC sends REMOTE_RESET_REQUEST with PrgMode SELECTOR.

Expectation: Target device restarts if and only if it is in Programming Mode.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sets BDUT to PrgMode=1 via KNX/IP Device Management.

TC sends REMOTE_RESET_REQUEST with PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)			
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x000A (Total Length)
0x02 (Struct Length)	0x01 (PrgMode Sel.)		
0x01 (Restart)	0x00 (reserved)		

BDUT restarts, and does not respond to subsequent REMOTE_DIAGNOSTIC_REQUEST.

TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds.

TC sets BDUT to PrgMode=0 via KNX/IP Device Management.

TC sends REMOTE_RESET_REQUEST with PrgMode SELECTOR:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)			
0x06 (Header Length)	0x10 Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x000A (Total Length)
0x02 (Struct Length)	0x01 (PrgMode Sel.)		
0x01 (Restart)	0x00 (reserved)		

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.5 Selection by MAC Address

Function ID: 50505

Description: This test sends REMOTE_RESET_REQUEST with MAC SELECTOR. First, TC sends REMOTE_RESET_REQUEST with MAC SELECTOR with correct MAC Address, then TC sends REMOTE_RESET_REQUEST with MAC SELECTOR with wrong MAC Address (obtained by inverting all bits).

Expectation: Target device restarts if and only if its MAC address matches.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST with MAC SELECTOR with correct MAC address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT restarts, and does not respond to subsequent REMOTE_DIAGNOSTIC_REQUEST.

TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds.

TC sends REMOTE_RESET_REQUEST with MAC SELECTOR with wrong MAC address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	(wrong MAC address)		
0x01 (Restart)	0x00 (reserved)			

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.6 Missing Reset Mode

Function ID: 50506

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST without Reset Mode (including reserved byte).

Expectation: Target device ignores this invalid datagram, and does not restart.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

TC sends REMOTE_RESET_REQUEST without ResetMode:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x000E (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.7 Unknown Reset Mode

Function ID: 50507

Description: This test sends REMOTE_BASIC_CONFIGURATION_REQUEST with unknown Reset Mode (value of first byte after SELECTOR).

Expectation: Target device ignores this invalid datagram, and does not restart.

Parameters: Unknown Reset Mode (Type = Byte, Default = 0x00); Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

TC sends REMOTE_RESET_REQUEST with unknown ResetMode:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x00 (unknown ResetMode)	0x00 (reserved)			

BDUT does not restart, and responds to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.8 Soft Reset

Function ID: 50508

Description: This test sends REMOTE_RESET_REQUEST with Reset Mode = Soft = Restart.

Expectation: Target device restarts, and afterwards has same IP configuration as before.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST with MAC SELECTOR with correct MAC address:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT restarts, and does not respond to subsequent REMOTE_DIAGNOSTIC_REQUEST.

TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds.

TC compares new and old IpConfig DIB from REMOTE_DIAGNOSTIC_RESPONSE.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.9 Hard Reset

CAUTION!

This test may abort the whole test run with fatal error.

Paradoxically this normally means success (i.e. Hard Rest was really executed), as the Hard Reset may produce an IP configuration making the BDUT inaccessible for the TC, so that the old configuration cannot be restored automatically. In this case, the old configuration must be restored manually (typically by ETS download via bus).

Automatically restoring the old configuration typically requires presence of a DHCP server in the IP subnet of the BDUT.

Function ID: 50509

Description: This test sends REMOTE_RESET_REQUEST with Reset Mode = Hard = MasterReset.

Expectation: Target device restarts, and afterwards has a changed IP configuration.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST with ResetMode = Hard:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x02 (Master Reset)	0x00 (reserved)			

BDUT restarts, and does not respond to subsequent REMOTE_DIAGNOSTIC_REQUEST.

TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds.

TC compares new and old IpConfig DIB from REMOTE_DIAGNOSTIC_RESPONSE.

TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

TC sends REMOTE_RESET_REQUEST with ResetMode = Soft:

UDP frame (IP_TC, CPORT_TC, IP_BDUT, CPORT_BDUT)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT restarts.

Cleanup: TC sends repeatedly REMOTE_DIAGNOSTIC_REQUEST until BDUT responds, then restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.

7.5.10 Request via Broadcast

Function ID: 50510

Description: This test sends REMOTE_RESET_REQUEST via Broadcast (destination address 255.255.255.255 instead of multicast address 224.0.23.12).

Expectation: Target device restarts.

Parameters: Max Dead Time / s (Type = Integer, Default = 180).

Preparation: TC reads configuration via REMOTE_DIAGNOSTIC_REQUEST.

Test sequence:

TC sends REMOTE_RESET_REQUEST via Broadcast:

UDP frame (IP_TC, CPORT_TC -> **255.255.255.255, CPORT_BDUT**)				
0x06 (Header Length)	0x10 (Ver.)	0x0743 (REMOTE_ RESET_REQUEST)	0x0010 (Total Length)	
0x08 (Struct Length)	0x02 (MAC Sel.)	MAC_BDUT		
0x01 (Restart)	0x00 (reserved)			

BDUT restarts, and does not respond to subsequent REMOTE_DIAGNOSTIC_REQUEST.

Cleanup: TC restores old configuration via REMOTE_BASIC_CONFIGURATION_REQUEST.