Specification

Z-Wave MAC Layer Test Specification

Version:	1.0
Description:	Test case descriptions for the Z-Wave MAC-Layer
Written By:	CSWG
Date:	2021.08.27
Reviewed By:	CSWG
Restrictions:	Public

Approved by:
Z-Wave Alliance Board of Directors

REVISION RECORD

Doc. Rev	Date	By	Pages affected	Brief description of changes
0.7	2020.09.11	CSWG	ALL	Full MAC test plan written and reviewed
0.8	2021.03.01	CSWG	None	Revision updated to 0.8 for TC review
0.9	2021.03.30	CSWG	Frontpage	Cleanup for IPR review • Added abbreviations
1.0	2021.08.27	ZWA Board		Approved for Publication

Table of Contents

1 ABBREVIATIONS

Abbreviation	Explanation
ACK	Acknowledgement
CRC	Cyclic Redundancy Check
FCS	Frame Check Sequence
FL	Frequently Listening
ID	Identification
ISM	Industrial, Scientific and Medical
LBT	Listen Before Talk
LLC	Logical Link Control
LSB	Least Significant Bit
MAC	Medium Access Control
MSDU	MAC Data
MFR	MAC Footer
MHR	MAC Header
MIB	Management Information Base
MLME	MAC Layer Management Entity
MLME-SAP	MAC Layer Management Entity – Service Access Point
MPDU	MAC Protocol Data Unit
MSB	Most Significant Bit
MSDU	MAC Service Data Unit
NPDU	Network layer Protocol Data Unit
PHY	Physical layer
RF	Radio Frequency
RSSI	Receive Signal Strength Indication
RX	Receive/Receiver
TX	Transmit/Transmitter

2 INTRODUCTION

2.1 Purpose

To provide a set of tests that help verify compliance with the MAC layer of the Z-Wave technology implemented outside of the Z-Wave development R&D group and Silicon Labs.

2.2 Audience and prerequisites

An RF sniffer hardware and analyzer software that can be tuned in on the valid frequencies for Z-Wave or a Z-Wave Zniffer module and PC Application. Z-Wave PC controller or equivalent to execute communication between the different nodes.

3 MAC-LAYER TEST CASE DESCRIPTIONS

3.1 General assumptions

The PHY layer is functional and follows the specification in ITU-T G.9959 (01/2015) and is verified by the set of tests on INS14660.

All components are defined in [1] and that document is the sole reference for the present Test Plan.

All **times** and **time-out periods** must be compliant with the values described in tables 8.18 & 8.19 from [1].

Inclusion in Z-Wave refers to Bootstrapping in [1] - 8.1.1.2, each device has the same HomeID and different NodeID. The controller needs to have each End node registered in its Node List with a unique NodeID.

3.2 Format of MPDU, Singlecast 2-channel frequencies

A device must be able to produce the 3 types of frames: Single Cast, Acknowledge and Multicast in 2-channel frequencies.

3.2.1 Prerequisites

1 x Z-Wave Zniffer.
1 x Z-Wave PC controller.
1 x End node.

3.2.2 Test Setup

1. Include End node to PC Controller network.
2. PC Controller sends Singlecast frame to End node with Data Payload (MSDU) = 0x00 (NOP).

3.2.3 Test Result

2. Verify on sniffer that End node responds with an Acknowledgement frame to the Controller (Header type: 0x03).

3.2.4 Pass Criteria

If the frames are displayed in the Z-Wave Zniffer, that means the PHY-layer header and EOF Delimiter are structured correctly (8.1.1.4.1.1).

1. The singlecast frame sent to the End node has the format from ITU figure 8.2 (8.1.1.4.1.1).
2. The singlecast frame sent to the End node has the frame type set to: 0x01 (8.1.2.1.3.1 – Table 8.4).
3. The singlecast frame sent to the End node has the ACK bit set to 0x01 (8.1.3.3)
4. The End node responds with an Acknowledgement frame (8.1.1.4.2.2).
5. This Acknowledgement frame matches the description (8.1.1.4.1.2).
6. This Acknowledgement singlecast responded has the frame type set to: 0x03 (8.1.2.1.3.1 – Table 8.4).
7. The ACK bit (byte 5, bit 6) in the Acknowledgement frame is set to 0 (8.1.3.3.2 – Figure 8.11).
8. This singlecast acknowledgement responded has the same HomeID as the sent singlecast (8.1.1.2).
9. This singlecast acknowledgement responded has the destination ID set to the node ID of the Controller (8.1.1.2).

3.2.5 Fail Criteria

1. The single cast does Not have the format described by figure 8.2 (8.1.1.4.1.1).
2. The singlecast frame sent to the End node doesn’t have the frame type set to: 0x01 (8.1.2.1.3.1 – Table 8.4).
3. The singlecast frame sent to the End node doesn’t have the ACK bit set to 0x01 (8.1.3.3).
4. The End node did not respond using an Acknowledgement frame (8.1.1.4.2.2).
5. The Acknowledgement singlecast frame does not match the description (8.1.1.4.1.2).

6. This Acknowledgement singlecast responded does not have the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
7. The ACK bit (byte 5, bit 6) in the Acknowledgement frame is NOT set to 0 (8.1.3.3.2 – Figure 8.11).
8. This singlecast acknowledgement responded has a Different HomeID than the singlecast (8.1.1.2).
9. This singlecast acknowledgement responded has a different destination ID than the node ID of the Controller (8.1.1.2).

3.3 Format of MPDU, Singlecast 3-channel frequencies

A device must be able to produce the 3 types of frames: Single Cast, Acknowledge and Multicast in 3-channel frequencies.

3.3.1 Prerequisites

1 x Z-Wave Zniffer.
1 x Z-Wave PC controller.
1 x End node.

3.3.2 Test Setup

1. Include End node to PC Controller network.
2. PC Controller sends Singlecast frame to End node with Data Payload (MSDU) = 0x00 (NOP).

3.3.3 Test Result

2. Verify on sniffer that End node responds with an Acknowledgement frame to the Controller (Header type: 0x03).

3.3.4 Pass Criteria

If the frames are displayed in the Z-Wave Zniffer, that means the PHY-layer header and EOF Delimiter are structured correctly (8.1.1.4.1.1).

1. The singlecast frame sent to the End node has the format from ITU figure 8.2 (8.1.1.4.1.1).
2. The singlecast frame sent to the End node has the frame type set to: 0x01 (8.1.2.1.3.1 – Table 8.4).
3. The singlecast frame sent to the End node has the ACK bit set to 0x01 (8.1.3.3)
4. The End node responds with an Acknowledgement frame (8.1.1.4.2.2).
5. This Acknowledgement frame matches the description (8.1.1.4.1.2).
6. This Acknowledgement singlecast responded has the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
7. The ACK bit (byte 5, bit 7) in the Acknowledgement frame is set to 0 (8.1.3.3.2– Figure 8.12).
8. This singlecast acknowledgement responded has the same HomeID as the sent singlecast (8.1.1.2).
9. This singlecast responded has the destination ID set to the node ID of the Controller (8.1.1.2).

3.3.5 Fail Criteria

1. The single cast does Not have the format described by figure 8.2 (8.1.1.4.1.1).
2. The singlecast frame sent to the End node doesn’t have the frame type set to: 0x01 (8.1.2.1.3.1 – Table 8.4).
3. The singlecast frame sent to the End node doesn’t have the ACK bit set (8.1.3.3).
4. The End node did not respond using an Acknowledgement frame (8.1.1.4.2.2).
5. The singlecast Acknowledgement frame does not match the description (8.1.1.4.1.2).

6. This singlecast Acknowledgement responded does not have the frame type set to: 0x03 (8.1.2.1.3.1 – Table 8.4).
7. The Request ACK bit (byte 5, bit 7) in the Acknowledgement frame is NOT set to 0 (8.1.3.3.2 – Figure 8.12).
8. This singlecast acknowledgement responded has a Different HomeID than the singlecast (8.1.1.2).
9. This singlecast acknowledgement responded has a different destination ID than the node ID of the Controller (8.1.1.2).

3.4 Format of MPDU, Multicast 2-channel frequencies

A device must be able to produce the 3 types of frames: Single Cast, Acknowledge and Multicast in 2-channel frequencies.

3.4.1 Prerequisites

1 x Z-Wave Zniffer.
1 x Z-Wave PC controller.
2 x End nodes

3.4.2 Test Setup

1. Include End node 1 and End node 2 to PC Controller network.
2. PC Controller sends Multicast frame to End node 1 and 2 with Data Payload (MSDU) = 0x00 (NOP).

3.4.3 Test Result

2. Controller sends 1 Multicast frame to both End nodes, automatically followed by one Singlecast to each of them with the same payload.
    a. End nodes respond with an Acknowledgement frame to the Singlecast frames (Header type: 0x03).

3.4.4 Pass Criteria

If the frames are displayed in the Z-Wave Zniffer, that means the PHY-layer header and EOF Delimiter are structured correctly (8.1.1.4.1.1).

1. The multicast frame has the format from ITU figure 8.2 (8.1.1.4.1.3).
2. The multicast frame sent to the End nodes has the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The multicast frame sent to the End nodes has the ACK bit set to 0x00 (8.1.3.6.1)
4. The multicast frame has 29 Mask Bytes and uses only the first one for the 2 End nodes nodes (8.1.3.6.1)
5. End nodes do not respond to the Multicast frame with an Acknowledgement frame (8.1.3.6.1).
6. Each Single cast frame complies with the all the Pass Criteria from TC 1.

3.4.5 Fail Criteria

1. The multicast frame does Not have the format from ITU figure 8.2 (8.1.1.4.1.3).
2. The multicast frame sent to the End nodes doesn't have the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The multicast frame sent to the End nodes Doesn't have the ACK bit set to 0x00 (8.1.3.6.1)
4. The multicast frame does not reserve the 29 Mask Bytes or uses more than the first byte for the 2 End nodes nodes (8.1.3.6.1)
5. At least one of the End nodes answers the Multicast frame with an Acknowledgement frame (8.1.3.6.1).
6. Any of the Fail criteria from TC1 is met.

3.5 Format of MPDU, Multicast 3-channel frequencies

A device must be able to produce the 3 types of frames: Single Cast, Acknowledge and Multicast in 3-channel frequencies.

3.5.1 Prerequisites

1 x Z-Wave Zniffer.
1 x Z-Wave PC controller.
2 x End nodes

3.5.2 Test Setup

1. Include End node 1 and End node 2 to PC Controller network.
2. PC Controller sends Multicast frame to End node 1 and 2 with Data Payload (MSDU) = 0x00 (NOP).

3.5.3 Test Result

2. Controller sends 1 Multicast frame to both End nodes, automatically followed by one Singlecast to each of them with the same payload.
    a. End nodes respond with an Acknowledgement frame to the Controller (Header type: 0x03) to the Singlecast frames.

3.5.4 Pass Criteria

If the frames are displayed in the Z-Wave Zniffer, that means the PHY-layer header and EOF Delimiter are structured correctly (8.1.1.4.1.1).

1. The multicast frame has the format from ITU figure 8.2 (8.1.1.4.1.3).
2. The multicast frame sent to the End nodes has the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The multicast frame sent to the End nodes has the ACK bit set to 0x00 (8.1.3.6.1)
4. The multicast frame has 29 Mask Bytes and uses only the first one for the 2 End nodes nodes (8.1.3.6.1)
5. End nodes do not respond to the Multicast frame with an Acknowledgement frame (8.1.3.6.1).
6. Each Single cast frame complies with the all the Pass Criteria from TC 2.

3.5.5 Fail Criteria

1. The multicast frame does Not have the format from ITU figure 8.2 (8.1.1.4.1.3).
2. The multicast frame sent to the End nodes doesn’t have the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The multicast frame sent to the End nodes Doesn’t have the ACK bit set to 0x00 (8.1.3.6.1)

4. The multicast frame does not reserve the 29 Mask Bytes or uses more than the first byte for the 2 End nodes nodes (8.1.3.6.1)
5. At least one of the End nodes answers the Multicast frame with an Acknowledgement frame (8.1.3.6.1).
6. Any of the Fail criteria from TC2 is met.

3.6 Network Robustness, Clear channel assessment: 2-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.6.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Noise Generator
1 x Spectrum Analyzer or RF Linx device for monitoring RF traffic over a specified bandwidth

3.6.2 Test Setup

1. Include End node to PC Controller network.
2. Configure Noise Generator to operate in the same frequency as the Z-Wave devices.
3. Use the Spectrum Analyzer or RF Linx device to monitor the RF traffic around the channels of the used frequency.
4. On the Zniffer observe that communication between Controller and End node is possible and which channels are used.
5. Start the noise Generator and observe on the RF traffic that there is higher volume of traffic along the channels of the chosen frequency.
6. Perform an ERTT test with 100 iterations observing how the transmissions come into a pause waiting for lower traffic in the channel.

3.6.3 Test Result

6. Observe on the Zniffer how the singlecast are not sent immediately as the PHY layer is waiting for an opening in traffic and that they are on the channel with less traffic.

3.6.4 Pass criteria

1. The singlecast frames don’t appear on the Zniffer right after being commanded, instead the node waits for an idle channel before transmitting (8.1.1.4.2.1)
2. The singlecast frames wait a bare minimum of 1100ms before being sent (8.1.4.1 – Table 8.18)
3. The singlecast uses the channel with higher communication rate (8.1.1.4.2.1)
4. The singlecast is responded with an acknowledgement frame from the End node.
5. The singlecast is responded with an acknowledgement frame from the End node in the same channel.

3.6.5 Fail criteria

1. The node does not wait for an idle channel before transmitting (8.1.1.4.2.1).
2. The singlecast fails being transmitted by trying to use a high traffic channel (8.1.1.4.2.1).
3. The singlecast is not responded with an acknowledgement frame at all.
4. The singlecast is not responded with an acknowledgement frame from the End node in the same channel.

3.7 Network Robustness, Clear channel assessment: 3-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.7.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Noise Generator
1 x Spectrum Analyzer or RF Linx device for monitoring RF traffic over a specified bandwidth

3.7.2 Test Setup

1. Include End node to PC Controller network.
2. Configure Noise Generator to operate in the same frequency as the Z-Wave devices.
3. Use the Spectrum Analyzer or RF Linx device to monitor the RF traffic around the channels of the used frequency.
4. On the Zniffer observe that communication between Controller and End node is possible and which channels are used.
5. Start the noise Generator and observe on the RF traffic that there is higher volume of traffic along the channels of the chosen frequency.
6. Perform an ERTT test with 100 iterations observing how the transmissions come into a pause waiting for lower traffic in the channel.

3.7.3 Test Result

6. Observe on the Zniffer how the singlecast are not sent immediately as the PHY layer is waiting for an opening in traffic and that they are on the channel with less traffic.

3.7.4 Pass criteria

1. The singlecast frames don’t appear on the Zniffer right after being commanded, instead the node waits for an idle channel before transmitting (8.1.1.4.2.1)
2. The singlecast frames wait a bare minimum of 1100ms before being sent (8.1.4.1 – Table 8.18)
3. The singlecast uses the channel with the least traffic or higher communication rate or lower channel number if the channels have the same communication rate and injection side (8.1.1.4.2.1)
4. The singlecast is responded with an acknowledgement frame from the End node.

5. The singlecast is responded with an acknowledgement frame from the End node in the same channel.

3.7.5 Fail criteria

1. The node does not wait for an idle channel before transmitting (8.1.1.4.2.1).
2. The singlecast fails being transmitted by trying to use a high traffic channel (8.1.1.4.2.1).
3. The singlecast is not responded with an acknowledgement frame at all.
4. The singlecast is not responded with an acknowledgement frame from the End node in the same channel.

3.8 Network Robustness, Acknowledgement 2-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.8.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller
1 x End node

3.8.2 Test Setup

1. Include End node to PC Controller network.
2. Include Secondary Controller to the primary one.
3. On the Zniffer observe that communication between both Controllers and End node is possible.
4. Perform an ERTT with 50 ms delay from one of the controllers to the End node and from the other Controller, send singlecast with MSDU = 0x00 (NOP) to the End node at the same time.

3.8.3 Test Result

4. Observe on the Zniffer how the End node replies to both controllers with one acknowledgement frame using the same sequence number as each singlecast. If the End node does not answer, the controller retransmits the frame with the same sequence number and the End node answer with the frame with it as described.

3.8.4 Pass criteria

1. The singlecast frames are answered with an acknowledgement frame (8.1.1.4.2.2).
2. This Acknowledgement frame matches the description (8.1.1.4.1.2).
3. This Acknowledgement singlecast responded has the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
4. The ACK bit (byte 5, bit 7) in the Acknowledgement frame is set to 0 (8.1.3.3.2– Figure 8.12).
5. This singlecast acknowledgement responded has the same HomeID as the sent singlecast (8.1.1.2).
6. This singlecast acknowledgement responded has the destination ID set to the node ID of the Controller that sent it (8.1.1.2).
7. When the Controllers send commands very close, the End node does not answer because both overlap for the End node to read correctly (8.1.1.4.2.2).
8. The controller retransmits the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)
9. When the Controller retransmit the frames, the End node answers with an acknowledgement using the same Sequence Number as the received frame (8.1.3.3.7).

3.8.5 Fail criteria

1. The End node did not respond using an Acknowledgement frame (8.1.1.4.2.2).
2. The Acknowledgement singlecast frame does not match the description (8.1.1.4.1.2).
3. This Acknowledgement singlecast responded does not have the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
4. The ACK bit (byte 5, bit 6) in the Acknowledgement frame is NOT set to 0 (8.1.3.3.2 – Figure 8.11).
5. This singlecast acknowledgement responded has a Different HomeID than the singlecast (8.1.1.2).
6. This singlecast acknowledgement responded has a different destination ID than the node ID of the Controller (8.1.1.2).
7. The End node answers the singlecasts with Acknowledgement frame even if it can’t read them correctly (8.1.1.4.2.2).
8. The controller does not retransmit the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)
9. When the Controllers retransmit the frames, the End node does not answer or answers with an Acknowledgement using a different Sequence Number as the received frame (8.1.3.3.7).

3.9 Network Robustness, Acknowledgement 3-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.9.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller
1 x End node

3.9.2 Test Setup

1. Include End node to PC Controller network.
2. Include Secondary Controller to the primary one.
3. On the Zniffer observe that communication between both Controllers and End node is possible.
4. Perform an ERTT with 50 ms delay from one of the controllers to the End node and from the other Controller, send singlecast with MSDU = 0x00 (NOP) to the End node at the same time.

3.9.3 Test Result

4. Observe on the Zniffer how the End node replies to both controllers with one acknowledgement frame using the same sequence number as each singlecast. If the End node does not answer, the controller retransmits the frame with the same sequence number and the End node answer with the frame with it as described.

3.9.4 Pass criteria

1. The singlecast frames are answered with an acknowledgement frame (8.1.1.4.2.2).
2. This Acknowledgement frame matches the description (8.1.1.4.1.2).
3. This Acknowledgement singlecast responded has the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
4. The ACK bit (byte 5, bit 7) in the Acknowledgement frame is set to 0 (8.1.3.3.2– Figure 8.12).
5. This singlecast acknowledgement responded has the same HomeID as the sent singlecast (8.1.1.2).
6. This singlecast ackowledgement responded has the destination ID set to the node ID of the Controller that sent it (8.1.1.2).
7. When the Controllers send commands very close, the End node does not answer because both overlap for the End node to read correctly (8.1.1.4.2.2).
8. The controller retransmits the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)
9. When the Controller retransmit the frames, the End node answers with an acknowledgement using the same Sequence Number as the received frame (8.1.3.5).

3.9.5 Fail criteria

1. The End node did not respond using an Acknowledgement frame (8.1.1.4.2.2)
2. The Acknowledgement singlecast frame does not match the description (8.1.1.4.1.2).
3. This Acknowledgement singlecast responded does not have the frame type set to: 0x03 (8.1.2.1.3.1– Table 8.4).
4. The ACK bit (byte 5, bit 7) in the Acknowledgement frame is NOT set to 0 (8.1.3.3.2 – Figure 8.11).
5. This singlecast acknowledgement responded has a Different HomeID than the singlecast (8.1.1.2).
6. This singlecast acknowledgement responded has a different destination ID than the node ID of the Controller (8.1.1.2).
7. The End node answers the singlecasts with Acknowledgement frame even if it can't read them correctly (8.1.1.4.2.2).
8. 8.The controller doesn't retransmit the frames that weren't answered with an Acknowledgement (8.1.1.4.2.3).
9. When the Controllers retransmit the frames, the End node does not answer or answers with an Acknowledgement using a different Sequence Number as the received frame (8.1.3.5).

3.10 Network Robustness, Acknowledgement OFF 2-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.10.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Frame Generator*

3.10.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections of the frame modified individually in order to test the behavior of the receiver.

1. Include End node to PC Controller network.
2. On the Zniffer observe that communication between both Controllers and End node is possible.
3. Generate a frame that sends to the End node a singlecast with MSDU = 0x00 (NOP) making sure ACK bit (byte 5, bit 6) is set to 0x00.
4. Send this frame as singlecast to the End node.

3.10.3 Test Result

4. Observe on the Zniffer how the End node ignores the singlecast and the Controller re-transmits the frame.

3.10.4 Pass criteria

1. The ACK bit (byte 5, bit 6) in the singlecast frame is set to 0 (8.1.3.3.2– Figure 8.11).
2. The singlecast frames are not answered with an acknowledgement frame (8.1.1.4.2.2).

3.10.5 Fail criteria

1. The End node did respond using an Acknowledgement frame (8.1.1.4.2.2).

3.11 Network Robustness, Acknowledgement OFF 3-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.11.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Frame Generator*

3.11.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End node to PC Controller network.
2. On the Zniffer observe that communication between both Controllers and End node is possible.
3. Generate a frame that sends to the End node a singlecast with MSDU = 0x00 (NOP) making sure ACK bit (byte 5, bit 7) is set to 0x00.
4. Send this frame as singlecast to the End node.

3.11.3 Test Result

4. Observe on the Zniffer how the End node ignores the singlecast and the Controller re-transmits the frame.

3.11.4 Pass criteria

1. The ACK bit (byte 5, bit 7) in the singlecast frame is set to 0 (8.1.3.3.2– Figure 8.12).
2. The singlecast frame is not answered with an acknowledgement frame (8.1.1.4.2.2).

3.11.5 Fail criteria

1. The End node did respond using an Acknowledgement frame (8.1.1.4.2.2).

3.12 Network Robustness, Multicast Acknowledgement ON 2-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.12.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End nodes
1 x Frame Generator*

3.12.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections of the frame modified individually in order to test the behavior of the receiver.

1. Include End nodes to PC Controller network.
2. On the Zniffer observe that communication between both Controller and End nodes is possible.
3. Generate a frame that sends to the End nodes a Multicast with MSDU = 0x00 (NOP) making sure ACK bit (byte 5, bit 6) is set to 0x01.
4. Send this frame as multicast to the End nodes.

3.12.3 Test Result

4. Observe on the Zniffer how the End nodes ignore the multicast and the Controller transmits the singlecast follow-up frames.

3.12.4 Pass criteria

1. The ACK bit (byte 5, bit 6) in the multicast frame is set to 1 (8.1.3.3.2– Figure 8.11).
2. The multicast frame sent to the End nodes has the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The Multicast frames are not answered with an acknowledgement frame (8.1.1.4.2.2).
4. The follow-up single cast frames are answered by the End nodes with an Acknowledgement frame (8.1.1.4.2.2).

3.12.5 Fail criteria

1. The End nodes did respond using an Acknowledgement frames to the Multicast frame (8.1.1.4.2.2).
2. The follow-up singlecast frames were not answered by the End nodes with an Acknowledgement frame (8.1.1.4.2.2).

3.13 Network Robustness, Multicast Acknowledgement ON 3-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.13.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End nodes
1 x Frame Generator*

3.13.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections of the frame modified individually in order to test the behavior of the receiver.

1. Include End nodes to PC Controller network.
2. On the Zniffer observe that communication between both Controller and End nodes is possible.
3. Generate a frame that sends to the End nodes a Multicast with MSDU = 0x00 (NOP) making sure ACK bit (byte 5, bit 6) is set to 0x01.
4. Send this frame as multicast to the End nodes.

3.13.3 Test Result

4. Observe on the Zniffer how the End nodes ignore the multicast and the Controller transmits the singlecast follow-up frames.

3.13.4 Pass criteria

1. The ACK bit (byte 5, bit 7) in the multicast frame is set to 1 (8.1.3.3.2– Figure 8.12).
2. The multicast frame sent to the End nodes has the frame type set to: 0x02 (8.1.2.1.3.1 – Table 8.4).
3. The Multicast frames are not answered with an acknowledgement frame (8.1.1.4.2.2).
4. The follow-up single cast frames are answered by the End nodes with an Acknowledgement frame (8.1.1.4.2.2).

3.13.5 Fail criteria

1. The End nodes did respond using an Acknowledgement frames to the Multicast frame (8.1.1.4.2.2).
2. The follow-up singlecast frames were not answered by the End nodes with an Acknowledgement frame (8.1.1.4.2.2).

3.14 Network Robustness, Retransmission

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.14.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node

3.14.2 Test Setup

1. Include End node to PC Controller network.
2. Disable the End node device by removing power or removing the antenna from it.
3. Send a singlecast from the controller to the End node. With MSDU = 0x00 (NOP).
4. On the Zniffer observe that communication between both Devices is not possible and the Controller sending the frame re-tries sending it.

3.14.3 Test Result

4. Observe on the Zniffer the transmission is attempted up to 2 times more (the maximum number of frame transmission retries “aMacMaxFrameRetries”) before increasing the Sequence Number and each re-transmission waits a random period to prevent collisions with other frames that may be being sent at the same time.

3.14.4 Pass criteria

1. The Controller sends only 2 retransmissions (“aMacMaxFrameRetries”) with the same Sequence Number waiting a random period of time after each attempt (8.1.1.4.2.3).
2. The Controller issues a new frame with the same contents but with its Sequence Number value increased by one and sent also only up to the value of “aMacMaxFrameRetries” waiting a random period of time after each attempt (8.1.1.4.2.3).

3.14.5 Fail criteria

1. When the Controllers retransmit the frames, the sequence number changes each time and does it a different amount of times than the one defined by “aMacMaxFrameRetries” (8.1.1.4.2.3).
2. The Controller does not issue any new frame, or it issues them with a Sequence Number entirely unrelated to the previously used one (8.1.1.4.2.3).

3.15 Network Robustness, Multi-hop routing

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.15.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller
1 x End node

3.15.2 Test Setup

1. Include End node and secondary Controller to PC Controller network.
2. Disable the End node device by removing power or removing the antenna from it.
3. Send a singlecast from the controller to the End node. With MSDU = 0x00 (NOP).

3.15.3 Test Result

3. On the Zniffer observe that communication between both Devices is not possible and the Controller sending the frame re-tries sending it.
   a. Observe on the Zniffer the transmission is attempted up to 2 times more (the maximum number of frame transmission retries “aMacMaxFrameRetries”)
   b. Then it increases the Sequence Number and tries again.
   c. Re-transmissions wait a random period to prevent collisions with other frames that may be being sent at the same time.
   d. On the Zniffer observe that after retransmitting, the Controller sends routed frames through the secondary Controller up to the “aMacMaxFrameRetries”.
   e. After the second retransmission is attempted, the Controller sends a singlecast to the secondary Controller in order to reach the End node.
   f. The secondary Controller attempts to reach the End node up to “aMacMaxFrameRetries”.

3.15.4 Pass criteria

1. The Controller sends only 2 retransmissions ("aMacMaxFrameRetries") with the same Sequence Number waiting a random period of time after each attempt (8.1.1.4.2.3).
2. The Controller issues a new frame with the same contents but with its Sequence Number value increased by one and sent also only up to the value of "aMacMaxFrameRetries" waiting a random period of time after each attempt (8.1.1.4.2.3).
3. The singlecast from the Controller to the secondary Controller has Header Type: 0x08
4. The singlecast from the Controller to the secondary Controller has Repeaters: 1.
5. The singlecast from the Controller to the secondary Controller has ACK bit set to 0x00.
6. The secondary Controller sends a singlecast to the End node with the same MSDU.
7. The singlecast the secondary Controller sends has Header Type: 0x08
8. The singlecast the secondary Controller sends has Repeaters: 1.
9. The singlecast the secondary Controller sends has Hops: 0x01.
10. The singecast the secondary Controller sends has ACK bit set to 0x00.
11. The secondary Controller retransmits the singlecast to the End node up to the value of "aMacMaxFrameRetries" waiting a random period of time after each attempt (8.1.1.4.2.3).

3.15.5 Fail criteria

1. When the Controllers retransmit the frames, the sequence number changes each time and does it a different amount of times than the one defined by “aMacMaxFrameRetries” (8.1.1.4.2.3).
2. The Controller does not issue any new frame, or it issues them with a Sequence Number entirely unrelated to the previously used one (8.1.1.4.2.3).
3. The singlecast from the Controller to the secondary Controller has a different eadet Type than: 0x08.
4. The singlecat from the Controller to the secondary Controller has a different number of repeaters than: 1.
5. The singlecast from the Controller to the secondary Controller as the ACK bit set to 0x01.
6. The secondary Controller sends a singlecast to the End node with a different MSDU.
7. The singlecast the secondary Controller sends has Heater Type different than: 0x08.
8. The singlecast the secondary Controller sends has Repeaters other than: 1.
9. The singlecast the secondary Controller sends has set Hops other than: 0x01.
10. The singlecast the secondary Controller sends has ACK bit set to 0x01.
11. The secondary Controller does not retransmit the singlecast to the End node up to the same value of “aMacMaxFrameRetries” and waits a consistent period of time after each attempt or does not retransmit at all (8.1.1.4.2.3).

3.16 Network Robustness, Data Validation Corrupt FCS, 2-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.16.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Frame Generator*

3.16.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End node to PC Controller network.
2. On the Zniffer observe that communication between both Controllers and End node is possible.
3. Generate a frame that sends to the End node a singlecast with MSDU = 0x00 (NOP) making sure the 8 bits for FCS are random and not generated automatically.
4. Send this frame as singlecast to the End node.

3.16.3 Test Result

4. Observe on the Zniffer how the End node ignores the singlecast and the Controller re-transmits the frame.

3.16.4 Pass criteria

1. The singlecast frame is not answered with an acknowledgement frame (8.1.1.4.2.5).
2. The controller retransmits the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)
3. When the Controller retransmit the frames, the End node remains without answering with an acknowledgement (8.1.1.4.2.2).

3.16.5 Fail criteria

1. The End node does respond using an Acknowledgement frame (8.1.1.4.2.2).
2. The End node answers the singlecasts with Acknowledgement frame even if it has bit errors as per FCS manipulation (8.1.1.4.2.2).
3. The controller does not retransmit the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)

3.17 Network Robustness, Acknowledgement Corrupt CRC 3-channel frequencies

A device must ensure robustness in data transmission. This is achieved by the mechanisms: Backoff Algorithm, Frame Acknowledgement, Data Verification and Frame Retransmission.

3.17.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node
1 x Frame Generator*

3.17.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End node to PC Controller network.
2. On the Zniffer observe that communication between both Controllers and End node is possible.
3. Generate a frame that sends to the End node a singlecast with MSDU = 0x00 (NOP) making sure the 16 bits of CRC are random and not generated automatically.
4. Send this frame as singlecast to the End node.

3.17.3 Test Result

4. Observe on the Zniffer how the End node ignores the singlecast and the Controller re-transmits the frame.

3.17.4 Pass criteria

1. The singlecast frames is not answered with an acknowledgement frame (8.1.1.4.2.2).
2. The controller retransmits the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)
3. When the Controller retransmit the frames, the End node remains without answering with an acknowledgement (8.1.1.4.2.2).

3.17.5 Fail criteria

1. The End node does respond using an Acknowledgement frame (8.1.1.4.2.2).
2. The End node answers the singlecasts with Acknowledgement frame even if it has bit errors as per CRC manipulation (8.1.1.4.2.2).
3. The controller does not retransmit the frames that weren’t answered with an Acknowledgement (8.1.1.4.2.3)

3.18 Power Consumption considerations, Sleeping nodes

Power consumption considerations are handled by the PHY layer. (8.1.1.4.3)

Beaming frames are covered un the MPDU Format for section (8.1.3)

3.19 MPDU Format, Components 2-channels

The MAC Protocol Data Unit (MPDU), consists of three basic components: A MAC Header (MHR), a MAC data payload (MAC Service Data Unit (MSDU)) and a MAC Footer (MFR).

3.19.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.19.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver

1. Include the End nodes to the Controller’s Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to one End node.
3. Observe the structure of the singlecast sent.
4. Select both End nodes and Send a multicast to both devices.
5. Observe the structure of the multicast sent.

3.19.3 Test Result

3. The singlecast is displalyed correctly on the Zniffer.
5. The multicast is displayed correctly on the Zniffer.

3.19.4 Pass Criteria

1. The singlecast shows:
   a. MHR: (8.1.3 – Figure 8.5)
      i. Home ID: 4 bytes (8.1.3.1)
      ii. Source Node ID: 1 byte (8.1.3.2)
      iii. Frame Control (2 bytes): (8.1.3.3 – Figure 8.11)
         1. 1^{st} byte (properties 1):
            a. Header type: 4 bits (8.1.3.3.5)
            b. Speed Modifier: 1 bit (8.1.3.3.4)
            c. Low Power: 1 bit (8.1.3.3.3)
            d. Ack Req: 1 bit (8.1.3.3.2)
            e. Routed: 1 bit (8.1.3.3.1)
         2. 2^{nd} byte:
            a. Sequence number: 4 bits (8.1.3.3.7)
            b. Reserved: 1 bit
            c. Beaming info: 2 bits (8.1.3.3.6)
            d. Reserved (SUC Present): 1 bit
      iv. Length: 1 byte (8.1.3.4)
      v. Destination Node ID: 1byte (8.1.3.6)
   b. MSDU Payload: (8.1.3.7)
      i. 1 Byte = 0x00 (NOP)

c. MFR (not described in the structure in the Zniffer): (8.1.3.8)
    i. FCS: 2 bytes

2. The multicast shows:
    a. MHR: (8.1.3 – Figure 8.7)
        i. Home ID: 4 bytes (8.1.3.1)
        ii. Source Node ID: 1 byte (8.1.3.2)
        iii. Frame Control (2 bytes): (8.1.3.3 – Figure 8.11)
            1. 1st byte (properties 1):
                a. Header type: 4 bits (8.1.3.3.5)
                b. Speed Modifier: 1 bit (8.1.3.3.4)
                c. Low Power: 1 bit (8.1.3.3.3)
                d. Ack Req: 1 bit (8.1.3.3.2)
                e. Routed: 1 bit (8.1.3.3.1)
            2. 2nd byte:
                a. Sequence number: 4 bits (8.1.3.3.7)
                b. Reserved: 1 bit
                c. Beaming info: 2 bits (8.1.3.3.6)
                d. Reserved (SUC Present): 1 bit
        iv. Length: 1 byte (8.1.3.4)
        v. Destination Node ID (Properties 3): (8.1.3.6.1)
            1. Number of Mask bytes: 5 bits (set to value 29)
            2. Offset addres: 3 bits (set to value 0)
        vi. Mask Byte: A list of the nodes addressed (2 bytes & 27 bytes set to 0)
    b. MSDU Payload: (8.1.3.7)
        i. 1 Byte = 0x00 (NOP)
    c. MFR (not described in the structure in the Zniffer): (8.1.3.8)
        i. FCS: 2 bytes

3.19.5 Fail Criteria

1. At Least one of the components of the format of the MPDU for the different frame types has a different length.

3.20 MPDU Format, Components 3-channels

The MAC Protocol Data Unit (MPDU), consists of three basic components: A MAC Header (MHR), a MAC data payload (MAC Service Data Unit (MSDU)) and a MAC Footer (MFR).

3.20.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.20.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver

1. Include the End nodes to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to one End node.
3. Observe the structure of the singlecast sent.
4. Select both End nodes and Send a multicast to both devices.
5. Observe the structure of the multicast sent.

3.20.3 Test Result

3. The singlecast is displalyed correctly on the Zniffer.
5. The multicast is displayed correctly on the Zniffer.

3.20.4 Pass Criteria

1. The singlecast shows:
    a. MHR: (8.1.3 – Figure 8.6)
        i. Home ID: 4 bytes (8.1.3.1)
        ii. Source Node ID: 1 byte (8.1.3.2)
        iii. Frame Control (2 bytes): (8.1.3.3 – Figure 8.12)
            1. 1st byte (properties 1):
                a. Header type: 4 bits (8.1.3.3.5)
                b. Reserved (Speed Modified & SUC Present): 2 bits (8.1.3.3.7)
                c. Low Power: 1 bit (8.1.3.3.3)
                d. Ack Req: 1 bit (8.1.3.3.1)
            2. 2nd byte:
                a. Reserved: 4 bits
                b. Beaming info: 3 bits (8.1.3.3.6)
                c. Reserved (Extended): 1 bit
        iv. Length: 1 byte (8.1.3.4)
        v. Sequence Number: 1 byte (8.1.3.5)
        vi. Destination Node ID: 1byte (8.1.3.6)
    b. MSDU Payload: (8.1.3.7)
        i. 1 Byte = 0x00 (NOP)
    c. MFR (not described in the structure in the Zniffer): (8.1.3.9)

            i. CRC: 2 bytes
2. The multicast shows:
    a. MHR: (8.1.3 – Figure 8.8)
        i. Home ID: 4 bytes (8.1.3.1)
        ii. Source Node ID: 1 byte (8.1.3.2)
        iii. Frame Control (2 bytes): (8.1.3.3 – Figure 8.12)
            1. 1st byte (properties 1):
                a. Header type: 4 bits (8.1.3.3.5)
                b. Reserved (Speed Modified & SUC Present): 2 bits (8.1.3.3.7)
                c. Low Power: 1 bit (8.1.3.3.3)
                d. Ack Req: 1 bit (8.1.3.3.2)
            2. 2nd byte:
                a. Reserved: 4 bits
                b. Beaming info: 3 bits (8.1.3.3.6)
                c. Reserved (Extended): 1 bit
        iv. Length: 1 byte (8.1.3.4)
        v. Sequence Number: 1 byte (8.1.3.5)
        vi. Destination Node ID (Properties 3): (8.1.3.6.1)
            1. Number of Mask bytes: 5 bits (set to value 29)
            2. Offset addres: 3 bits (set to value 0)
        vii. Mask Byte: A list of the nodes addressed (2 bytes & 27 bytes set to 0)
    b. MSDU Payload: (8.1.3.7)
        i. 1 Byte = 0x00 (NOP)
    c. MFR (not described in the structure in the Zniffer): (8.1.3.9)
        i. CRC: 2 bytes

3.20.5 Fail Criteria

1. At Least one of the components of the format of the MPDU for the different frame types has a different length.

3.21 MPDU Format, Home ID

The MAC Protocol Data Unit (MPDU), consists of three basic components: A MAC Header (MHR), a MAC data payload (MAC Service Data Unit (MSDU)) and a MAC Footer (MFR). Home ID are 4 bytes that identify all nodes in the same domain.

3.21.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.21.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End node devices to the Network of the Controller.
2. Verify the Controller can communicate with both End nodes by sending Singlecast to each of them and Multicast to both.
3. Send a singlecast to one of the End nodes, modifying the Home ID to be different from the original value.
4. Send a multicast to both End nodes with modified Home ID.
5. Send a singlecast to one of the End nodes extending the length of the Home ID value*.
6. Send a singlecast to one of the End nodes shortening the length of the Home ID value*.
   *only possible if the Frame generator allows to modify the length of the different fields.

3.21.3 Test Result

2. Both End nodes answer with an Acknowledgement frame as expected to the singlecast and the singlecast follow-up after the Multicast.
3. The End node does not answer, since the Home ID is not the Home ID it has been included to.
4. Neither End node answers since the Home ID is different from the one, they have been included to.
5. After the Home Id component, the rest of the frame will not match against the expected values, therefore the frame should be rejected.
6. After the Home Id component, the rest of the frame will not match against the expected values, therefore the frame should be rejected.

3.21.4 Pass Criteria

1. On the singlecast the Home ID occupies only 4 bytes (8.1.3.1)
2. On the Multicast, the Home Id occupies only 4 bytes (8.1.3.1)
3. No node responds to any frame that holds a modified Home Id in any way, because of mis-matching Home ID value or because the frame is affected because of re-sizing the HomeID (8.1.3.1)

3.21.5 Fail Criteria

1. Any of the methods for altering the Home ID Component is accepted by the receiving node and answered with an acknowledgement frame.

3.22 MPDU Format, Source NodeID

The MAC Protocol Data Unit (MPDU), consists of three basic components: A MAC Header (MHR), a MAC data payload (MAC Service Data Unit (MSDU)) and a MAC Footer (MFR). Source Node ID is 1 byte that identifies the node within one domain that have transmitted the frame.

3.22.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.22.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver

1. Include End nodes to Controller's network.
2. Verify the Controller can communicate with both End nodes by sending Singlecast to each of them and Multicast to both.
3. Send a singlecast to one of the End nodes, modifying the Source Node ID to be different from the original value.
4. Send a Multicast to both End nodes, modifying the Source Node Id to be different from the original value.
5. Send a singlecast to one of the End nodes, modifying the Source Node ID to be 0x00.
6. Send a singlecast to one of the End nodes, modifying the Source Node ID to be a value between 0xE9 & 0xFF (reserved values).
7. Send a singlecast to one of the End nodes, modifying the Source node ID to be 2 bytes long * *only possible if the Frame generator allows to modify the length of the different fields.

3.22.3 Test Result

3. End node received the singlecast and responds to the Source Node ID modified with an Acknowledgement frame.
4. The End nodes do not answer the multicast frame but answer to the singlecast follow-up frames originated after the Multicast
5. The End node receiving the singlecast with Source Node ID set to 0x00, will answer to it with an Acknowledgement frame.
6. The End node receiving the singlecast with Source Node ID set to a reserved value will not answer, since a network is limited to that number of nodes.
7. As the structure of the frame will be affected by an oversized Source Node ID component, it won't be received correctly by the End node and won't answer with an Acknowledgement frame. The Controller should try to retransmit the modified frame.

3.22.4 Pass Criteria

1. When the Controller does not receive the Acknowledgement frame, the Controller re-tries sending the same frame up to 3 times and then routing through the other End node because Each time the receiving End node answers to the modified Source Node ID. This happens for single cast, singlecast follow-up, frames addressed to Node 0 or frames with a structure affected by a longer Source Node ID field. (8.1.3.6) (8.1.1.4.2.3)
2. The End nodes won’t answer to a frame with Source Node Id set to a reserved value, causing the controller to retransmit the frame. (8.1.3.6)

3.22.5 Fail Criteria

1. The End node answers to the controller with an Acknowledgement frame directly, ignoring the field Source Node ID (8.1.3.6)
2. The Controller does not re-transmit when the Acknowledgement frames are not addressed to it. (8.1.1.4.2.3)

3.23 MPDU Format, Frame Control, Routed

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Routed subfield is 1 bit in length. It should be set to 1 when routing and 0 otherwise. It is used only by 2-channel frequencies.

3.23.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.23.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to the Controller's network.
2. Disable the antenna in the Controller and one of the End nodes and try to send a frame with MPDU = 0x00 (NOP) to this End node.
3. Generate a frame with the Routed flag (Bye 6, bit 7 in the frame) disabled and try to reach out to the same End node again.

3.23.3 Test Result

2. The frame sent to the End node does not reach it and the controller Retransmits the frame without receiving an Acknowledgement frame and then it tries by routing the frame through the other End node. The End node answers with an Acknowledgement frame routed through the other End node.
3. The Controller tries to reach directly but fails, its routed frame is displayed as a regular singlecast in the Zniffer and the repeater does not send this frame to the destination node, but it sends an Acknowledgement frame to the Controller.

3.23.4 Pass Criteria

1. The routed frame has its Routed flag (Byte 7, bit 7) set to 1 (8.1.3.3.1)
2. The routed frame has its Destination node Id Set to the selected End node value (8.1.3.6)
3. The Routed frame also sets its repeaters count to 1 and sets the Repeater field to the Node ID of the repeater Node.
4. The Routed Acknowledgement frame has the Routed flag set to 1 (8.1.3.3.1)
5. The routed Acknowledgement frame has its Destination node Id Set to the Controller node value (8.1.3.6)
6. The Routed Acknowledgement frame also sets its repeaters count to 1 and sets the Repeater field to the Node ID of the Controller Node.
7. The Routed Acknowledgement frame from the End node has its Ack bit set to 0x00 (8.1.3.3.2)
8. The Routed Acknowledgement frame from the Routing End node has its Ack bit set to 0x01
9. The Controller sends an Acknowledgement frame to the Repeater. (8.1.3.3.2)

10. When The frame has the routed flag disabled (set to 0), the Ack Request field is still enabled, and the Repeater responds with an Acknowledgement frame to the Controller. (8.1.3.3.1)
11. Since the Controller received an Acknowledgement frame, It doesn’t pursue sending the frame any further.

3.23.5 Fail Criteria

1. The Routed flag is not set to 1 in the standard routed frames.
2. The Routed frame is not Acknowledged with a routed Acknowledgement frame.
3. The Routed Acknowledgement frame from the Destination Node has its Ack flag set to 1.
4. The Controller does not respond to the Repeater with a single Acknowledgement Frame at the end of the transmission.
5. The Repeater ignores the Routed flag set to 0 and Repeats the frame to the Destination Node.
6. The Destination Node responds with an Acknowledgement Frame to either the Controller or the Repeating Node.

3.24 MPDU Format, Frame Control, Low Power 2-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Low Power subfield is 1 bit that informs a destination node that the actual transmission was using low power. A receiving node shall return an acknowledgement Frame in low power in response to a frame with this bit enabled.

3.24.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.24.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controller's Network.
2. Send a regular singlecast to one End node with MDPU = 0x00 (NOP)
3. Generate a Frame that has Low Power bit (Bye 5, bit 5), set to 1 and send it to End node.
4. Send a multicast to both End nodes with the Low Power bit set to 1.

3.24.3 Test Results

2. End node answers with an Acknowledgement Frame to the Controller in regular Power.
3. The End node answers to the Controller with an Acknowledgement frame in Low Power.
4. The Multicast is sent in Low Power and so are the Singlecast Follow-up frames.

3.24.4 Pass Criteria

1. The singlecast is sent by default with the Low Power bit set to 0. (8.1.3.3.3)
2. The End node answers with an Acknowledgement frame in regular power and its Low Power bit set to 0 (8.1.3.3.3).
3. The modified singlecast is responded with an Acknowledgement frame in low Power with its Low Power bit set to 1 (8.1.3.3.3).
4. The Multicast is not answered by any of the End nodes (8.1.3.6.1).
5. The Singlecast Follow Up frames have their Low Power set to 1. The End nodes answer to the singlecast follow-up frames with Acknowledgement frames in low power and with the Low Power bit set to 1 (8.1.3.3.3).

3.24.5 Fail Criteria

1. The default singlecast is sent in Low Power with its Low Power bit set to 1 (8.1.3.3.3).
2. The End nodes answer to a regular singlecast with a Low Power Acknowledgement frame with the Low Power bit enabled (8.1.3.3.3).
3. The End nodes do not answer to the singlecast with its Low Power bit set to 1 (8.1.3.3.3).

4. The End nodes answer to the singlecast with Low power set to 1 in regular power and with their Low Power bit set to 0 (8.1.3.3.3).
5. The End nodes answer to the modified Multicast frame directly with an Acknowledgement frame (8.1.3.6.1).
6. The End nodes do as steps Fail criteria 3 & 4 for the Singlecast Follow-up frames (8.1.3.3.3).

3.25 MPDU Format, Frame Control, Low Power 3-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Low Power subfield is 1 bit that informs a destination node that the actual transmission was using low power. A receiving node shall return an acknowledgement Frame in low power in response to a frame with this bit enabled.

3.25.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.25.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controller's Network.
2. Send a regular singlecast to one End node with MDPU = 0x00 (NOP)
3. Generate a Frame that has Low Power bit (Bye 5, bit 6), set to 1 and send it to End node.
4. Send a multicast to both End nodes with the Low Power bit set to 1.

3.25.3 Test Results

2. End node answers with an Acknowledgement Frame to the Controller in regular Power.
3. The End node answers to the Controller with an Acknowledgement frame in Low Power.
4. The Multicast is sent in Low Power and so are the Singlecast Follow-up frames.

3.25.4 Pass Criteria

1. The singlecast is sent by default with the Low Power bit set to 0. (8.1.3.3.3)
2. The End node answers with an Acknowledgement frame in regular power and its Low Power bit set to 0. (8.1.3.3.3)
3. The modified singlecast is responded with an Acknowledgement frame in low Power with its Low Power bit set to 1. (8.1.3.3.3)
4. The Multicast is not answered by any of the End nodes. (8.1.3.6.1)
5. The Singlecast Follow Up frames have their Low Power set to 1. The End nodes answer to the singlecast follow-up frames with Acknowledgement frames in low power and with the Low Power bit set to 1. (8.1.3.3.3)

3.25.5 Fail Criteria

1. The default singlecast is sent in Low Power with its Low Power bit set to 1. (8.1.3.3.3)
2. The End nodes answer to a regular singlecast with a Low Power Acknowledgement frame with the Low Power bit enabled. (8.1.3.3.3)
3. The End nodes do not answer to the singlecast with its Low Power bit set to 1. (8.1.3.3.3)
4. The End nodes answer to the singlecast with Low power set to 1 in regular power and with their Low Power bit set to 0. (8.1.3.3.3)
5. The End nodes answer to the modified Multicast frame directly with an Acknowledgement frame. (8.1.3.6.1)
6. The End nodes do as steps Fail criteria 3 & 4 for the Singlecast Follow-up frames. (8.1.3.3.3)

3.26 MPDU Format, Frame Control, Speed Modified subfield (2-channel Only)

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Speed modified subfield is one bit used to show that the frame is sent at a lower speed than supported by the source and destination. It should not be used for routing nor multicast frame. It should be reset to 0 if the Frame is sent at the highest supported speed.

3.26.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.26.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes to Controller's Network
2. Send a singlecast to one of them with MPDU = 0x00 (NOP)
3. Send a multicast to both nodes with MPDU = 0x00 (NOP)
4. Generate a frame modifying the Speed Modified subfield to 1 and send it as a singlecast to one of the nodes
5. Generate a frame modifying the Speed Modified subfield to 1 and send it to both End nodes as a multicast
6. Disable one of the End nodes and send a singlecast to it from the Controller
7. Generate a frame with Speed Modified subfield set to 1 and try to send it to the disabled End node
8. Exclude both nodes from the Controller's Network

3.26.3 Test Result

2. Singlecast has the Speed Modified subfield set to 0
3. The Multicast as well as the singlecast follow-up frames have the Speed Modified subfield set to 0x00
4. The Singlecast is sent in Lower speed and it's received correctly by the End node
5. The multicast is sent in nominal speed with its Speed Modified subfield set to 1 and the singlecast are sent in lower speed to each End node with their respective Speed Modified subfield set to 1. *
6. The routed singlecast has its Speed Modificed subfield set to 0.
7. The direct singlecast has its Speed Modified subfield set to 1 but when re-trasmitting, the routed frames have their Speed Modified subfield set to 0. *
8. During exclusion Observe that the last exclusion NOP frame has enabled Speed Modified field.

*needs practical verification.

3.26.4 Pass Criteria

1. Other than for the mentioned exclusion frames, no frame has its Speed Modified subfield set to 0x01. (8.1.3.3.4)

3.26.5 Fail Criteria

1. Speed Modified subfield is set to 1 in any other case than in the last exclusion NOP frame. (8.1.3.3.4)

3.27 MPDU Format, Frame Control, Header Type, singlecast 2-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The header type defines the frame Header type. A broadcast MPDU is a singlecast MPDU (type 0x01) carrying destination Node ID = 0xFF.

3.27.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node

3.27.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes to Controller's Network.
2. Send one single cast to the End node with MPDU 0x00 (NOP).
3. Generate a frame with Header going from 0x02 to 0x0F according to table 8-14 and send it to the End node.
4. Generate a frame with Header going from 0x02 to 0x0F according to table 8-14 and send it to Node ID 0xFF (255).

3.27.3 Test Results

2. Singlecast is sent correctly and it's answered with an Acknowledgement frame.
3. Each frame sent to the node is displayed as the corresponding type on the Zniffer, it's ignored by the End node and no Acknowledgement frame is responded. (0x08 is shown as a singlecast)
4. Each frame sent to Node ID 255 is displayed as the corresponding type on the Zniffer*

* needs practical verification.

3.27.4 Pass Criteria

1. Each frame sent by the Controller in 3. & 4. is displayed as the corresponding type on the Zniffer, making each frame correctly defined. (8.1.3.3.5)
2. None of the frames sent in 3. Are answered by definition.
3. None of the frames sent in 4. Are answered by definition nor by being addressed to a reserved Destination Node ID.

3.27.5 Fail Criteria

1. Any frame is displayed as singlecast on the Zniffer regardless of the different Header. (8.1.3.3.5)
2. Any frame sent in 3. received an Acknowledgement frame.
3. Any frame sent in 4. received an Acknowledgement frame.

3.28 MPDU Format, Frame Control, Header Type, singlecast 3-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The header type defines the frame Header type. A broadcast MPDU is a singlecast MPDU (type 0x01) carrying destination Node ID = 0xFF.

3.28.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node

3.28.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes to Controller's Network.
2. Send one single cast to the End node with MPDU 0x00 (NOP).
3. Generate a frame with Header going from 0x02 to 0x0F according to table 8-14 and send it to the End node.
4. Generate a frame with Header going from 0x02 to 0x0F according to table 8-14 and send it to Node ID 0xFF (255).

3.28.3 Test Results

2. Singlecast is answered correctly with an Acknowledgement frame.
3. Each frame sent to the node is displayed as the corresponding type on the Zniffer, it's ignored by the End node and no Acknowledgement frame is responded. (0x08 is shown as a routed frame).
4. Each frame sent to Node ID 255 is displayed as the corresponding type on the Zniffer*

* needs practical verification.

3.28.4 Pass Criteria

1. Each frame sent by the Controller in 3. & 4. is displayed as the corresponding type on the Zniffer, making each frame correctly defined. (8.1.3.3.5)
2. None of the frames sent in 3. Are answered by definition.
3. None of the frames sent in 4. Are answered by definition nor by being addressed to a reserved Destination Node ID.

3.28.5 Fail Criteria

1. Any frame is displayed as singlecast on the Zniffer regardless of the different Header. (8.1.3.3.5)
2. Any frame sent in 3. received an Acknowledgement frame.
3. Any frame sent in 4. received an Acknowledgement frame.

3.29 MPDU Format, Frame Control, Header Type, Multicast

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The header type defines the frame Header type. A broadcast MPDU is a singlecast MPDU (type 0x01) carrying destination Node ID = 0xFF.

3.29.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.29.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes to the Controller’s network.
2. Send one multicast to both End nodes with MPDU 0x00 (NOP).
3. Generate a frame with Header going from 0x01 to 0x0F (except 0x02) according to table 8-14 and send it to both End nodes.
4. Generate a frame with Header going from 0x01 to 0x0F (except 0x02) according to table 8-14 and send it to Node ID 0xFF (255).

3.29.3 Test Result

2. The Multicast is sent correctly, followed by its corresponding singlecast Follow-up frames, which are answered with Acknowledgement frames from the End nodes.
3. Each frame sent is constructed as a multicast frame, but with the corresponding header from table 8-14. Therefore, it’s displayed on the zniffer as the expected type with a longer structure.
4. Each frame is constructed as a multicast frame, but with the corresponding header from table 8-14. Therefore, it’s displayed on the zniffer as the expected type with a longer structure. *

* needs practical verification.

3.29.4 Pass Criteria

1. Each frame sent by the Controller in 3. & 4. is displayed as the corresponding type on the Zniffer, making each frame correctly defined. (8.1.3.3.5)
2. None of the frames sent in 3. Are answered by definition.
3. None of the frames sent in 4. Are answered by definition nor by being addressed to a reserved Destination Node ID.

3.29.5 Fail Criteria

1. Any frame is displayed as multicast on the Zniffer regardless of the different Header. (8.1.3.3.5)
2. Any frame sent in 3. received an Acknowledgement frame.
3. Any frame sent in 4. received an Acknowledgement frame.

3.30 MPDU Format, Frame Control, Header Type, Acknowledgement

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The header type defines the frame Header type. A broadcast MPDU is a singlecast MPDU (type 0x01) carrying destination Node ID = 0xFF.

3.30.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller

3.30.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include the End node Controller to Primary Controller’s network.
2. Send one singlecast to the End node with MPDU 0x00 (NOP).
3. Generate an Acknowledgement frame with Header going from 0x01 to 0x0F (except 0x03) according to table 8-14 to be answered by the End node to the Primary Controller and send a singlecast to the End node for each header type.
4. Generate an Acknowledgement frame with Header going from 0x01 to 0x0F (except 0x03) according to table 8-14 to be answered by the End node to Node ID 0xff (255) and send a singlecast to the End node for each header type.

3.30.3 Test Result

2. The singlecast is sent correctly and it’s answered with an Acknowledgement frame from the End node.
3. Each frame answered is constructed as an Acknowledgement frame, but with the corresponding header from table 8-14.
    a. Therefore, it’s displayed on the zniffer as the expected type.
    b. Since the frames are not identified as Acknowledgement Header Type, the Primary Controller tries to retransmit each time. *
4. Each frame answered is constructed as an Acknowledgement frame, but with the corresponding header from table 8-14 and addressed to Node ID 0xFF.
    a. Therefore, it’s displayed on the zniffer as the expected type except for type 0x01, showing as a Broadcast.
    b. Since the frames are not identified as Acknowledgement Header Type, the Primary Controller tries to retransmit each time. *

* needs practical verification.

3.30.4 Pass Criteria

1. Each frame sent by the Controller in 3. is answered with the corresponding type on the Zniffer, making each frame correctly defined. (8.1.3.3.5)
2. None of the frames sent in 3. Are answered by the Primary Controller by definition.
3. None of the frames sent in 4. Are answered by the Primary Controller by definition.

3.30.5 Fail Criteria

1. Any frame is displayed as Acknowledgement on the Zniffer regardless of the different Header. (8.1.3.3.5)
2. Any frame sent in 3. received an Acknowledgement frame.
3. Any frame sent in 4. received an Acknowledgement frame.

3.31 MPDU Format, Frame Control, Header Type, Routed MPDU (3-channel only)

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The header type defines the frame Header type. A broadcast MPDU is a singlecast MPDU (type 0x01) carrying destination Node ID = 0xFF.

3.31.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.31.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes to the Controller's network
2. Send a singlecast to each End node
3. Disable one of the End nodes Antenna and try to reach to it from the Controller
4. Generate a Routed frame with its Header type going from 0x01 to 0x0F (except 0x08) according to table 8-14 and send them to the End node with Antenna disabled.
5. Generate a Routed frame with its Header type going from 0x01 to 0x0F (except 0x08) according to table 8-14 and address it to Node ID 0xFF (255).

3.31.3 Test Result

2. Each End node answers the singlecast with an Acknowledgement frame
3. The Controller tries to reach the disabled End node through the other one sending a routing frame (Header type 0x08 – Routing).
    a. The End node responds with an Acknowledgement frame routed through the repeater.
4. The Controller tries to reach the End node with disabled antenna through the repeater with each frame constructed as a Routed frame but with header types from table 8-14.
    a. The repeater does not recognize the routing frames except for type 0x01 (singlecast) and only responds with Acknowledgement to this one, the rest do not reach the End node with disabled antenna.
    b. The Controller tries retransmitting each of those frames.
    c. Each frame is displayed on the Zniffer as the corresponding type. *
5. The Controller sends each frame constructed as a Routed frame but with header types from table 8-14 and addressed it to Node ID 0xFF (255). Each frame is displayed on the Zniffer as the corresponding type. No frame is answered with Acknowledgement frame. *

* needs practical verification.

3.31.4 Pass Criteria

1. The routed frame has Header Type 0x08 (Routed). (8.1.3.3.5)
2. None of the frames sent in 4. Other than 0x01, received an Acknowledgement frame.
3. None of the frames sent in 5. received an Acknowledgement frame.

3.31.5 Fail Criteria

1. Any frame is displayed as Routing on the Zniffer regardless of the different Header. (8.1.3.3.5)
2. Any frame sent in 4. Except for 0x01 received an Acknowledgement frame.
3. Any frame sent in 5. received an Acknowledgement frame.

3.32 MPDU Format, Frame Control, Beaming Information 2-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Beaming information sub-field are 2 bits that shall advertise the capability of a sending FL node to be awakened by a beam. For 2-channel frequencies, refer to Table 8.14a.

3.32.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End node

3.32.2 Test Setup

1. Include FliRS End node to Controller's network.
2. Send a singlecast callback command to FLiRS device.

3.32.3 Test Result

2. Verify communication is possible with FLiRS device.

3.32.4 Pass Criteria

1. Observe in Singlecast and broadcast frames sent from the FLiRS End node that they are configured as 0x10: "Source Wake Up Beam 1000ms: true", corresponding to "Long continuos beam" or 0x01: "Source Wake Up Beam 250ms: true", corresponding to "Short continuos beam" in Table 8.14a (8.1.3.3.6)

3.32.5 Fail Criteria

1. No frame sent from the FLiRS End node has any configuration

3.33 MPDU Format, Frame Control, Beaming Information 3-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The Beaming information sub-field are 3 bits that shall advertise the capability of a sending FL node to be awakened by a beam. For 3-channel frequencies, refer to Table 8.14b.

3.33.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End node

3.33.2 Test Setup

1. Include FliRS End node to Controller's network.
2. Send a singlecast callback command to FLiRS device.

3.33.3 Test Result

2. Verify communication is possible with FLiRS device.

3.33.4 Pass Criteria

1. Observe in singlecast and broadcast frames sent from the FLiRS End node that they are configured as "Source Wake Up: 0x04", corresponding to "Fragmented beam" in Table 8.14b (8.1.3.3.6)

3.33.5 Fail Criteria

1. No frame sent from the FLiRS End node has any configuration

3.34 MPDU Format, Frame Control, Sequence number 2-channel

The Frame Control field is 16 bits (2 bytes) in length. It defines the frame type and other control flags. The sequence number is a 4-bit sub-field provided by higher layers when transmitting. The same Sequence Number shall be used for al retransmissions of a given MPDU that first fails being delivered. A receiving node shall return the same value in an Acknowledgement frame if the Ack bit is present in the received frame. For Backwards compatibility, an Acknowledgement frame received can hold Sequence number = 0.

3.34.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller
2 x End node

3.34.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes and secondary Controller to the Primary Controller's network.
2. Disable one of the End nodes' antenna and send a singlecast with MPDU = 0x00 (NOP) to it.
3. Enable the End node again and try sending a singlecast to it again.
4. Select both End nodes and send a multicast from the controller.
5. Configure an Acknowledgement frame so that it has a random non-zero static value on the Secondary Controller when the Primary Controller transmits a singlecast to it and proceed to send a singlecast to the Secondary Controller.
6. Configure an Acknowledgement frame so that it has a static value of zero on the Secondary Controller when the Primary Controller transmits a singlecast to it and proceed to send a singlecast to the Secondary Controller.
7. Configure a singlecast frame with static sequence number and send it twice or more to one of the End nodes from the Primary Controller.
8. Configure a singlecast frame with static sequence number set to 0 and send it twice or more to one of the End nodes from the Primary Controller.
9. Configure a singlecast frame with a longer Sequence Number value than 4 bits and send it to one of the End nodes.

3.34.3 Test Result

2. Observe that the Controller re-tries sending the command to the disabled End node and all frames have the same sequence number.
    a. The Controller tries reaching the destination node routing through the other End node and/or the Secondary Controller. The routed frames have the same sequence number.
3. The Controller routes directly through a repeater and the frame reaches correctly the destination Node.

    a. The End node Answers with an Acknowledgement frame using either the same sequence number or sequence number = 0x0.
4. The multicast frame and its respective single cast follow-up frames have their own sequence numbers
5. When receiving the Singlecast, the secondary Controller responds with the generated Acknowledgement frame with the static Sequence Number value.
    a. When the Controller received this Acknowledgement frame with a Sequence Number value different from the singlecast it transmitted, it tries to retransmit the frame again. Since this is equivalent to not having received the proper Ackowledgement frame*
6. When receiving the Singlecast, the secondary Controller responds with the generated Acknowledgement frame with the static Number value set to 0.
    a. The Primary Controller accepts this Acknowledgement frame correctly.
7. The End node receiving the Singlecast answers correctly to the first one with an Acknowledgement frame using the same Sequence Number.
    a. In the following frames, it ignores the frame, as it holds the same Sequence Number value as previous frames.
    b. Since the frames go unanswered, the Primary Controller tries re-transmitting the frames again.
8. The End node receiving the Singlecast with Sequence Number set to 0 Does not answer to it.
    a. The Primary Controller tries re-transmitting the frame with the same Sequence Number set to 0.
    b. The receiver it ignores the frames.
9. Since the sequence number has a larger size than it's supposed to have by definition, the Receiving node finds a miscalculation in the Frame Check Sequence (FCS).
   *Needs practical verification.

3.34.4 Pass Criteria

1. The re-transitted frames when the Controller doesn't reach the End node have the same Sequence Number. (8.1.3.3.7)
2. The routed frames that are repeated by the secondary Controller and the second End node have the same Sequence Number. (8.1.3.3.7)
3. The Ack frames, both routed and the final one directed from the Controller to the repeater have the same Sequence number as the original singlecast frame sent from the Controller. (8.1.3.3.7)
4. The Ack frames with Sequence Number set to zero are accepted by the Primary Controller. (8.1.3.3.7)
5. The Multicast and its follow-up singlecast have successive Sequence numbers. (8.1.3.3.7)
6. Acknowledgement frames with value that do not match the one of the singlecast that originated them are rejected by the Controller and re-transmitted by the receiving node. (8.1.3.3.7)
7. Sequence Number only has 4 bits going from 0x1 to 0xF. (8.1.3.3.7)

3.34.5 Fail Criteria

1. The retransmitted frames have their own Sequence Number value. (8.1.3.3.7)
2. The routed frames repeated by either of the repeaters have their own Sequence Number value. (8.1.3.3.7)
3. The Ack frames routed and the final one from the Controller to the repeater have different Sequence Number value. (8.1.3.3.7)

4. The Ack frames with Sequence number set to 0 are rejected by the Primary Controller. (8.1.3.3.7)
5. The Multicast and its successive Follow-up singelcast frames have the same Sequence Number value. (8.1.3.3.7)
6. All Acknowledgement frames with non-zero value different from the singlecast that originated them are accepted. (8.1.3.3.7)
7. Sequence Number can have more than 4 bits of length and the singlecast can hold value 0x0. (8.1.3.3.7)

3.35 MPDU Format, Sequence number 3-channel

The Sequence Number in 3-channel is an 8-bit field of the MPDU Header (MHR). The same Sequence Number shall be used for al retransmissions of a given MPDU that first fails being delivered. A receiving node shall return the same value in an Acknowledgement frame if the Ack bit is present in the received frame. Sequence Number can be in the range from 0x00 to 0xFF, cycling through with 0x00 used after 0xFF.

3.35.1 Prerequisites

1 x Z-Wave Zniffer
2 x Z-Wave PC Controller
2 x End node

3.35.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include both End nodes and secondary Controller to the Primary Controller's network.
2. Disable one of the End nodes' antenna and send a singlecast with MPDU = 0x00 (NOP) to it.
3. Enable the End node again and try sending a singlecast to it again.
4. Select both End nodes and send a multicast from the controller.
5. Configure an Acknowledgement frame so that it has a random static value on the Secondary Controller when the Primary Controller transmits a singlecast to it and proceed to send a singlecast to the Secondary Controller.
6. Configure a singlecast frame with static sequence number and send it twice or more to one of the End nodes from the Primary Controller.
7. Configure a singlecast frame with a longer Sequence Number value than 8 bits and send it to one of the End nodes.

3.35.3 Test Result

2. Observe that the Controller re-tries sending the command to the disabled End node and all frames have the same sequence number.
    a. The Controller tries reaching the destination node routing through the other End node and/or the Secondary Controller. The routed frames have the same sequence number.
3. The Controller routes directly through a repeater and the frame reaches correctly the destination Node.
    a. The End node Answers with an Acknowledgement frame using either the same sequence number.
4. The multicast frame and its respective single cast follow-up frames have their own sequence numbers
5. When receiving the Singlecast, the secondary Controller responds with the generated Acknowledgement frame with the static Sequence Number value.
    a. When the Controller received this Acknowledgement frame with a Sequence Number value different from the singlecast it transmitted, it tries to retransmit the frame again. Since this is equivalent to not having received the proper Ackowledgement frame*

6. The End node receiving the Singlecast answers correctly to the first one with an Acknowledgement frame using the same Sequence Number.
    a. In the following frames, it ignores the frame, as it holds the same Sequence Number value as previous frames.
    b. Since the frames go unanswered, the Primary Controller tries re-transmitting the frames again.
7. Since the sequence number has a larger size than it's supposed to have by definition, the Receiving node finds a miscalculation in the Frame Check Sequence (FCS).
   *Needs practical verification.

3.35.4 Pass Criteria

1. The re-transitted frames when the Controller doesn't reach the End node have the same Sequence Number. (8.1.3.5)
2. The routed frames that are repeated by the secondary Controller and the second End node have the same Sequence Number. (8.1.3.5)
3. The Ack frames, both routed and the final one directed from the Controller to the repeater have the same Sequence number as the original singlecast frame sent from the Controller. (8.1.3.5)
4. The Multicast and its follow-up singlecast have successive Sequence numbers. (8.1.3.5)
5. Acknowledgement frames with value that do not match the one of the singlecast that originated them are rejected by the Controller and re-transmitted by the receiving node. (8.1.3.5)
6. Sequence Number only has 8 bits going from 0x00 to 0xFF. (8.1.3.5s)

3.35.5 Fail Criteria

1. The retransmitted frames have their own Sequence Number value. (8.1.3.5)
2. The routed frames repeated by either of the repeaters have their own Sequence Number value. (8.1.3.5)
3. The Ack frames routed and the final one from the Controller to the repeater have different Sequence Number value. (8.1.3.5)
4. The Ack frames with Sequence number set to 0x00 are rejected by the Primary Controller. (8.1.3.5)
5. The Multicast and its successive Follow-up singelcast frames have the same Sequence Number value. (8.1.3.5)
6. All Acknowledgement frames with value different from the singlecast that originated them are accepted. (8.1.3.5)
7. Sequence Number can have more than 8 bits of length (8.1.3.5)

3.36 MPDU Format, Length, 2-channel singlecast

The length field is 1 byte that indicates the length of the MPDU in bytes. It's limited by "aMacMaxMSDUSizeX" defined on table 8.18. A receiving node shall not accept a frame larger than the maximum length allowed for the actual data rate. For Singlecast in 2 channels, it's "aMacMaxMSDUSizeR1"/"aMacMaxMSDUSizeR2".

3.36.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node

3.36.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include the End node to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to the End node.
3. Look for the Length field.
4. Populate the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeR1"/"aMacMaxMSDUSizeR2" and send it to the End node.
5. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 11 and less than 54. Send it to the End node.
6. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 54. Send it to the End node.
7. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 1 byte. Send it to the End node.
8. Generate a singlecast populating the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeR1"/"aMacMaxMSDUSizeR2" and modify the value of the Length field to be 11. Send it to the End node.

3.36.3 Test Result

2. Communication is possible and End node answers with an Acknowledgement frame. Check on the Singlecast the Length field.
3. The Length field should be in Byte 7 of the frame and be show value 11 (0x0B) for a NOP MPDU.
4. The singlecast should show the corresponding size in length.
5. When the End node receives it, the stated size and the actual size as well as the FCS values do not correspond, and the End node ignores the singlecast.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.
6. When the End node receives it, the stated size is larger than "aMacMaxMSDUSizeR1"/"aMacMaxMSDUSizeR2" and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.

7. When the End node receives it, the structure of the frame is outside specifications and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.
8. When the End node receives it, the stated size of the frame is smaller than it actually is and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.

3.36.4 Pass Criteria

1. The Length field is only one byte in length. (8.1.3.4)
2. The Length field is in byte 7 of the Frame (8.1.3 – Figure 8.5)
3. The value of the Length field is always less or equal than “aMacMaxMSDUSizeR1”/“aMacMaxMSDUSizeR2”. (8.1.3.4)
4. The receiving node ignores all instances where the Length field does not match the actual length of the frame. (8.1.3.4)

3.36.5 Fail Criteria

1. The Length field is different from one byte in length. (8.1.3.4)
2. The Length field is located outside byte 7 of the Frame (8.1.3 – Figure 8.5)
3. The value of the length field can be more than “aMacMaxMSDUSizeR1”/“aMacMaxMSDUSizeR2”. (8.1.3.4)
4. The receiving node accepts and answers with an Acknowledgement frame any frame regardless of the size and value of the Length field. (8.1.3.4)

3.37 MPDU Format, Length, 2-channel multicast

The length field is 1 byte that indicates the length of the MPDU in bytes. It's limited by "aMacMaxMSDUSizeX" defined on table 8.18. A receiving node shall not accept a frame larger than the maximum length allowed for the actual data rate. For multicast in 2 channels, it's "aMacMaxMSDUSizeMultiR1"/"aMacMaxMSDUSizeMultiR2" , the full size comprises both the MPDU and the 29 bytes for the multicast mask.

3.37.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.37.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include the End nodes to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to each End node.
3. Send a multicast with MPDU = 0x00 (NOP) to both End nodes.
4. Populate the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeMultiR1"/"aMacMaxMSDUSizeMultiR2" and send it to both End nodes.
5. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 42 and less than 54. Send it to the End nodes.
6. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 54. Send it to the End nodes.
7. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 1 byte. Send it to the End nodes.
8. Generate a multicast populating the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeMultiR1"/"aMacMaxMSDUSizeMultiR2" and modify the value of the Length field to be 11. Send it to the End node.

3.37.3 Test Result

2. Communication is possible and each End node answers with an Acknowledgement frame. Check on the Singlecast the Length field.
    a. The Length field should be in Byte 7 of each frame and be show value 11 (0x0B) for a NOP MPDU.
3. The multicast is sent followed by a singlecast follow-up for each End node.
    a. The Multicast frame has its Length byte in Byte 7, with a value of 42 (0x2A) for a NOP MPDU for 2 End nodes.
4. The Multicast and follow-up singlecast should show the corresponding size in length.
5. The multicast is not responded, and the End nodes ignore the singlecast follow-up frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame from either End node.*

6. The multicast is not responded, when the End nodes receive the singlecast follow-up frames, the stated size is larger than “aMacMaxMSDUSizeR1”/“aMacMaxMSDUSizeR2” and the End nodes ignore the frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*
7. The multicast is not responded, when the End nodes receive the singlecast follow-up frames, the structure of the frame is outside specifications and the End nodes ignore the frame. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*
8. The multicast is not responded, when the End nodes receive the follow-up singlecast frames, the stated size of the frames is smaller than it actually is and the End nodes ignore the frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*

*Needs practical verification.

3.37.4 Pass Criteria

1. The Length field is only one byte in length. (8.1.3.4)
2. The Length field is in byte 7 of the Frame (8.1.3 – Figure 8.6)
3. The value of the Length field is always less or equal than “aMacMaxMSDUSizeMultiR1”/“aMacMaxMSDUSizeMultiR2”. (8.1.3.4)
4. The receiving node ignores all instances where the Length field does not match the actual length of the frame. (8.1.3.4)

3.37.5 Fail Criteria

1. The Length field is different from one byte in length. (8.1.3.4)
2. The Length field is located outside byte 7 of the Frame (8.1.3 – Figure 8.6)
3. The value of the length field can be more than “aMacMaxMSDUSizeMultiR1”/“aMacMaxMSDUSizeMultiR2”. (8.1.3.4)
4. The receiving node accepts and answers with an Acknowledgement frame any frame regardless of the size and value of the Length field. (8.1.3.4)

3.38 MPDU Format, Length, 3-channel singlecast

The length field is 1 byte that indicates the length of the MPDU in bytes. It's limited by "aMacMaxMSDUSizeX" defined on table 8.18. A receiving node shall not accept a frame larger than the maximum length allowed for the actual data rate. For Singlecast in 3 channels, it's "aMacMaxMSDUSizeR3".

3.38.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x End node

3.38.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include the End node to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to the End node.
3. Look for the Length field.
4. Populate the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeR3" and send it to the End node.
5. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 12 and less than 158. Send it to the End node.
6. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 158. Send it to the End node.
7. Generate a singlecast with MPDU = 0x00 (NOP) and modify the Length field to be more than 1 byte. Send it to the End node.
8. Generate a singlecast populating the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeR3" and modify the value of the Length field to be 11. Send it to the End node.

3.38.3 Test Result

2. Communication is possible and End node answers with an Acknowledgement frame. Check on the Singlecast the Length field.
3. The Length field should be in Byte 7 of the frame and be show value 12 (0x0C) for a NOP MPDU.
4. The singlecast should show the corresponding size in length.
5. When the End node receives it, the stated size and the actual size as well as the FCS values do not correspond, and the End node ignores the singlecast.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.
6. When the End node receives it, the stated size is larger than "aMacMaxMSDUSizeR3" and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.

7. When the End node receives it, the structure of the frame is outside specifications and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.
8. When the End node receives it, the stated size of the frame is smaller than it actually is and the End node ignores the frame.
    a. The Controller tries re-transmitting the same singlecast because of not receiving an Acknowledgement frame.

3.38.4 Pass Criteria

1. The Length field is only one byte in length. (8.1.3.4)
2. The Length field is in byte 7 of the Frame (8.1.3 – Figure 8.7)
3. The value of the Length field is always less or equal than “aMacMaxMSDUSizeR3”. (8.1.3.4)
4. The receiving node ignores all instances where the Length field does not match the actual length of the frame. (8.1.3.4)

3.38.5 Fail Criteria

1. The Length field is different from one byte in length. (8.1.3.4)
2. The Length field is located outside byte 7 of the Frame (8.1.3 – Figure 8.7)
3. The value of the length field can be more than “aMacMaxMSDUSizeR3”. (8.1.3.4)
4. The receiving node accepts and answers with an Acknowledgement frame any frame regardless of the size and value of the Length field. (8.1.3.4)

3.39 MPDU Format, Length, 3-channel multicast

The length field is 1 byte that indicates the length of the MPDU in bytes. It's limited by "aMacMaxMSDUSizeX" defined on table 8.18. A receiving node shall not accept a frame larger than the maximum length allowed for the actual data rate. For multicast in 3 channels, it's "aMacMaxMSDUSizeMultiR3", the full size comprises both the MPDU and the 29 bytes for the multicast mask.

3.39.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.39.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include the End nodes to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to each End node.
3. Send a multicast with MPDU = 0x00 (NOP) to both End nodes.
4. Populate the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeMultiR3" and send it to both End nodes.
5. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 42 and less than 158. Send it to the End nodes.
6. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 158. Send it to the End nodes.
7. Generate a multicast with MPDU = 0x00 (NOP) and modify the Length field to be more than 1 byte. Send it to the End nodes.
8. Generate a multicast populating the MPDU with a long amount of random data, less than "aMacMaxMSDUSizeMultiR3" and modify the value of the Length field to be 42. Send it to the End node.

3.39.3 Test Result

2. Communication is possible and each End node answers with an Acknowledgement frame. Check on the Singlecast the Length field.
    a. The Length field should be in Byte 7 of each frame and be show value 12 (0x0C) for a NOP MPDU.
3. The multicast is sent followed by a singlecast follow-up for each End node.
    a. The Multicast frame has its Length byte in Byte 7, with a value of 42 (0x2A) for a NOP MPDU for 2 End nodes.
4. The Multicast and follow-up singlecast should show the corresponding size in length.

5. The multicast is not responded, and the End nodes ignore the singlecast follow-up frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame from either End node.*
6. The multicast is not responded, when the End nodes receive the singlecast follow-up frames, the stated size is larger than “aMacMaxMSDUSizeR3” and the End nodes ignore the frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*
7. The multicast is not responded, when the End nodes receive the singlecast follow-up frames, the structure of the frame is outside specifications and the End nodes ignore the frame. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*
8. The multicast is not responded, when the End nodes receive the follow-up singlecast frames, the stated size of the frames is smaller than it actually is and the End nodes ignore the frames. *
    a. The Controller tries re-transmitting the same singlecasts because of not receiving an Acknowledgement frame.*

*Needs practical verification.

3.39.4 Pass Criteria

1. The Length field is only one byte in length. (8.1.3.4)
2. The Length field is in byte 7 of the Frame (8.1.3 – Figure 8.8)
3. The value of the Length field is always less or equal than “aMacMaxMSDUSizeMultiR3”. (8.1.3.4)
4. The receiving node ignores all instances where the Length field does not match the actual length of the frame. (8.1.3.4)

3.39.5 Fail Criteria

1. The Length field is different from one byte in length. (8.1.3.4)
2. The Length field is located outside byte 7 of the Frame (8.1.3 – Figure 8.8)
3. The value of the length field can be more than “aMacMaxMSDUSizeMultiR3”. (8.1.3.4)
4. The receiving node accepts and answers with an Acknowledgement frame any frame regardless of the size and value of the Length field. (8.1.3.4)

3.40 MPDU Format, Destination ID, singlecast – 2-channel

The destination Node ID specifies a destination node in the same domain identified by the HomeID. It shall comply with table 8.15.

3.40.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.40.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controler's network.
2. Send a singlegast with MPDU = 0x00 (NOP) to each End node.
3. Look for the Destination NodeID field in the frames on the Zniffer.
4. Generate a singlecast frame with Destination ID value different to either End node, send it.
5. Generate a singlecast frame with Destination ID value higher than 0xE8 (232), send it.
6. Generate a singlecast frame with Destination ID field longer than 1 byte, send it.
7. Generate a singlecast frame with Destination ID value of 0xFF (255), send it.

3.40.3 Test Result

2. Communication is correct. End nodes answer with an Ackowledgement frame.
3. Destination NodeID is byte 8 in the frames. It holds the value of the End node's NodeID.
4. The Controller tries to reach this End node but can't reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node but can't reach it.
5. The Controller tries to reach this End node but can't reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node but can't reach it.
6. The Controller tries to reach this End node but can't reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node, but they can only recognize the Most Significative Byte of this larger Destination ID (because the frame structure is distorted) and route to it.
7. The Controller sends this frame as a Broadcast.
    a. The End nodes don't respond to this Broadcast frame.

3.40.4 Pass Criteria

1. The Destination Node ID is one byte in length. (8.1.3.6)
2. The Destination Node ID can be any value up to 0xE8 (232) (8.1.3.6)

3. The Destination Node ID is in Byte 8 of the frame (8.1.3 – Figure 8.5)
4. Routing devices repeat frames respecting the NodeID it is addressed to (regardless of its value), except for 0xFF (broadcast) (8.1.3.6)

3.40.5 Fail Criteria

1. The Destination Node ID can be more than one byte in length (8.1.3.6).
2. The Destination Node ID can be any value beyond 0xE8 (232) (8.1.3.6).
3. Routing devices don't repeat all frames regardless of the Destination Node ID all the same (8.1.3.6).
4. Listening devices respond or try to route singlecast addressed to 0xFF (Broadcast) (8.1.3.6).

3.41 MPDU Format, Destination ID, singlecast – 3-channel

The destination Node ID specifies a destination node in the same domain identified by the HomeID. It shall comply with table 8.15.

3.41.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.41.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controler’s network.
2. Send a singlegast with MPDU = 0x00 (NOP) to each End node.
3. Look for the Destination NodeID field in the frames on the Zniffer.
4. Generate a singlecast frame with Destination ID value different to either End node, send it.
5. Generate a singlecast frame with Destination ID value higher than 0xE8 (232), send it.
6. Generate a singlecast frame with Destination ID field longer than 1 byte, send it.
7. Generate a singlecast frame with Destination ID value of 0xFF (255), send it.

3.41.3 Test Result

2. Communication is correct. End nodes answer with an Ackowledgement frame.
3. Destination NodeID is byte 9 in the frames. It holds the value of the End node’s NodeID.
4. The Controller tries to reach this End node but can’t reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node but can’t reach it.
5. The Controller tries to reach this End node but can’t reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node but can’t reach it.
6. The Controller tries to reach this End node but can’t reach it.
    a. The Controller retransmits and routes through the existing End nodes.
    b. The End nodes try to route the frame to this node, but they can only recognize the Most Significative Byte of this larger Destination ID (because the frame structure is distorted) and route to it.
7. The Controller sends this frame as a Broadcast.
    a. The End nodes don’t respond to this Broadcast frame.

3.41.4 Pass Criteria

1. The Destination Node ID is one byte in length. (8.1.3.6)
2. The destination Node ID can be any value up to 0xE8 (232) (8.1.3.6)
3. The Destination Node ID is in Byte 9 of the frame (8.1.3 – Figure 8.6)

4. Routing devices repeat frames respecting the NodeID it is addressed to (regardless of its value), except for 0xFF (broadcast) (8.1.3.6)

3.41.5 Fail Criteria

1. The Destination Node ID can be more than one byte in length (8.1.3.6).
2. The Destination Node ID can be any value beyond 0xE8 (232) (8.1.3.6).
3. Routing devices don't repeat all frames regardless of the Destination Node ID all the same (8.1.3.6).
4. Listening devices respond or try to route singlecast addressed to 0xFF (Broadcast) (8.1.3.6).

3.42 MPDU Format, Destination ID, multicast – 2-channel

The destination Node ID specifies a destination node in the same domain identified by the HomeID. It shall comply with table 8.15. A multicast shall carry a Multicast Control field and a multi-byte Multicast Bit Mask complying with Figure 8.14.

3.42.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.42.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controler's network.
2. Send a Multicast to both End nodes with MPDU 0x00 (NOP).
3. Look for the Multicast Control byte in the multicast frame.
4. Look for the Multicast Bit Mask.
5. Look at the singlecast follow-up frames sent to each of the End nodes.
6. Generate a Multicast frame with the Acknowledgement Request bit set to 1, send it to both End nodes.
7. Generate a Multicast frame with the Address Offset sub-field set to 1, send it to both End nodes.
8. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 20, send it to both End nodes.
9. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 30, send it to both End nodes.
10. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 29, include a list of 30 bytes set to 0x00 in the Mask Bytes list, send it to both End nodes.

3.42.3 Test Result

2. Multicast is sent to both End nodes correctly with a singlecast follow-up for each of them.
3. The Multicast Control Byte is byte 8 in the frame.
    a. The Number of Mask Bytes sub-field is 5 bits in length, and it's set to 0x1D (length of mask bytes is 29).
    b. The Address Offset sub-field is 3 bits in length, and it's set to 0x00.
4. The Multicast Bit Mask section starts in byte 9 and is 29 bytes in length.
    a. It holds the mask for the selected End nodes (for End nodes 2 & 3, the mask should be 0x06 = 0110 => For bits 2 & 3 in the mask).
5. Each of the singlecast follow-up frames complies with the results step 3 for TC 38.
6. Both End nodes ignore the Multicast frame and don't respond to it because the Multicast frame does not have header type 0x01 (singlecast).*
    a. The follow-up singlecast have the Acknowledgement Request bit set to 1 and they are answered with an Acknowledgement frame.
7. The multicast frame has Address Offset sub-field set to 1.

   a. The Mask Byte fields show: 10 & 11.*
   b. The destination Nodes in the Zniffer are 10 & 11.*
   c. The singlecast follow-up are addressed to Nodes 10 & 11.*
   d. These singlecast follow-up frames are not received by any End node and the Controller tries to re-transmit. *
8. The multicast frame has is addressed correctly to Node ID’s 2 & 3.
   a. The multicast frame has a total of only 20 Mask Bytes.
   b. The frame is not answered by the End nodes.
   c. The follow-up singlecast frames are addressed correctly to Node ID’s 2 & 3.
9. The Multicast is transmitted correctly
   a. The Mask Bytes Subfield is set to 30.
   b. The amount of Mask Bytes is 30.*
   c. The singlecast follow-up are sent, addressed to End nodes 2 & 3.*
   d. The singlecast follow-up are NOT responded with Ack frames.*
   e. The Controller tries re-transmitting the singlecast follow-up frames.*
10. The Multicast is transmitted correctly
   a. The Mask Bytes sibfield is set to 29.
   b. The amount of Mask Bytes is 30. *
   c. The singlecast follow-up are sent, addressed to End nodes 2 & 3.*
   d. The singlecast follow-up are NOT responded with Ack frames.*
   e. The Controller tries re-transmitting the singlecast follow-up frames.*

* Needs practical verification.

3.42.4 Pass Criteria

1. The multicast frame has the Multicast Control Byte in position 8 in the frame. (8.1.3, Figure 8.7).
2. The multicast frame has its Acknowledgement Request bit set to 0 (8.1.3.6.1)
3. The Address Offset sub-field is 3 bits long (8.1.3.6.1 – Figure 8.14)
4. The Number of Mask Bytes sub-field is 5 bits long and be set to 0x1D (29) (8.1.3.6.1 – Figure 8.14)
5. The amount of Mask Bytes is exactly 29 (8.1.3.6.1 – Figure 8.14)

3.42.5 Fail Criteria

1. The Multicast Control Byte is NOT in position 8 in the frame. (8.1.3, Figure 8.7).
2. The multicast frame has its Acknowledgement Request bit set to 1 (8.1.3.6.1)
3. The Address Offset sub-field is NOT 3 bits long (8.1.3.6.1 – Figure 8.14)
4. The Number of Mask Bytes sub-field is NOT 5 bits long (8.1.3.6.1 – Figure 8.14)
5. There is a different amount of Mask Bytes than 29 (8.1.3.6.1 – Figure 8.14)

3.43 MPDU Format, Destination ID, multicast – 3-channel

The destination Node ID specifies a destination node in the same domain identified by the HomeID. It shall comply with table 8.15. A multicast shall carry a Multicast Control field and a multi-byte Multicast Bit Mask complying with Figure 8.14.

3.43.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.43.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to Controler's network.
2. Send a Multicast to both End nodes with MPDU 0x00 (NOP).
3. Look for the Multicast Control byte in the multicast frame.
4. Look for the Multicast Bit Mask.
5. Look at the singlecast follow-up frames sent to each of the End nodes.
6. Generate a Multicast frame with the Acknowledgement Request bit set to 1, send it to both End nodes.
7. Generate a Multicast frame with the Address Offset sub-field set to 1, send it to both End nodes.
8. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 20, send it to both End nodes.
9. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 30, send it to both End nodes.
10. Generate a Multicast frame with Address Offset set to 0 and Number of Mask Bytes set to 29, include a list of 30 bytes set to 0x00 in the Mask Bytes list, send it to both End nodes.

3.43.3 Test Result

2. Multicast is sent to both End nodes correctly with a singlecast follow-up for each of them.
3. The Multicast Control Byte is byte 9 in the frame.
    a. The Number of Mask Bytes sub-field is 5 bits in length, and it's set to 0x1D (length of mask bytes is 29).
    b. The Address Offset sub-field is 3 bits in length, and it's set to 0x00.
4. The Multicast Bit Mask section starts in byte 10 and is 29 bytes in length.
    a. It holds the mask for the selected End nodes (for End nodes 2 & 3, the mask should be 0x06 = 0110 => For bits 2 & 3 in the mask).
5. Each of the singlecast follow-up frames complies with the results step 3 for TC 39.
6. Both End nodes ignore the Multicast frame and don't respond to it because the Multicast frame does not have header type 0x01 (singlecast).*
    a. The follow-up singlecast have the Acknowledgement Request bit set to 1 and they are answered with an Acknowledgement frame.
7. The multicast frame has Address Offset sub-field set to 1.

   a. The Mask Byte fields show: 10 & 11.*
   b. The destination Nodes in the Zniffer are 10 & 11.*
   c. The singlecast follow-up are addressed to Nodes 10 & 11.*
   d. These singlecast follow-up frames are not received by any End node and the Controller tries to re-transmit. *
8. The multicast frame has is addressed correctly to Node ID's 2 & 3.
   a. The multicast frame has a total of only 20 Mask Bytes.
   b. The frame is not answered by the End nodes.
   c. The follow-up singlecast frames are addressed correctly to Node ID's 2 & 3.
9. The Multicast is transmitted correctly
   a. The Mask Bytes Subfield is set to 30.
   b. The amount of Mask Bytes is 30.*
   c. The singlecast follow-up are sent, addressed to End nodes 2 & 3.*
   d. The singlecast follow-up are NOT responded with Ack frames.*
   e. The Controller tries re-transmitting the singlecast follow-up frames.*
10. The Multicast is transmitted correctly
   a. The Mask Bytes sibfield is set to 29.
   b. The amount of Mask Bytes is 30. *
   c. The singlecast follow-up are sent, addressed to End nodes 2 & 3.*
   d. The singlecast follow-up are NOT responded with Ack frames.*
   e. The Controller tries re-transmitting the singlecast follow-up frames.*

* Needs practical verification.

3.43.4 Pass Criteria

1. The multicast frame has the Multicast Control Byte in position 8 in the frame. (8.1.3, Figure 8.7).
2. The multicast frame has its Acknowledgement Request bit set to 0 (8.1.3.6.1)
3. The Address Offset sub-field is 3 bits long (8.1.3.6.1 – Figure 8.14)
4. The Number of Mask Bytes sub-field is 5 bits long and be set to 0x1D (29) (8.1.3.6.1 – Figure 8.14)
5. The amount of Mask Bytes is exactly 29 (8.1.3.6.1 – Figure 8.14)

3.43.5 Fail Criteria

1. The Multicast Control Byte is NOT in position 8 in the frame. (8.1.3, Figure 8.7).
2. The multicast frame has its Acknowledgement Request bit set to 1 (8.1.3.6.1)
3. The Address Offset sub-field is NOT 3 bits long (8.1.3.6.1 – Figure 8.14)
4. The Number of Mask Bytes sub-field is NOT 5 bits long (8.1.3.6.1 – Figure 8.14)
5. There is a different amount of Mask Bytes than 29 (8.1.3.6.1 – Figure 8.14)

3.44 MPDU Format, Mac Footer (MFR): FCS – 2-channel

FCS is the 8-bit non-correcting Frame Check Sequence (FCS) used for validating the integrity of a frame, used in 2-channel frequencies. It shall be calculated from the HomeID field to the Data Payload, both included.

3.44.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node Test Setup

3.44.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to each End node.
3. Look for the FCS field in each frame.
4. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its FCS value to a random hardcoded value, send it to each End node separately.
5. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its FCS value to be more than 8 bits, send it to each End node separately.
6. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its FCS value to be less than 8 bits, send it to each End node separately.
7. Send an unmodified multicast frame with MPDU = 0x00 (NOP) to both End nodes. Look for the FCS in the multicast.
8. Look for the FCS field in each singlecast follow-up frame.
9. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its FCS value to a random hardcoded value, send it to both End nodes. Look for the FCS in the multicast.
10. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its FCS value to be more than 8 bits, send it to both End nodes. Look for the FCS in the multicast.
11. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its FCS value to be less than 8 bits, send it to both End nodes. Look for the FCS in the multicast.

3.44.3 Test Result

2. Communication is correct with both End nodes and they answer with an Acknowledgement frame each.
3. Make sure there are 8 MFR bits corresponding to the FCS section in each frame.
4. Neither End node answers to the frame, since the FCS does not allow to verify its integrity.
    a. Each time the frame is displayed on the Zniffer as a CRC error.
    b. The controller tries retransmitting the frame since it doesn't receive an Acknowledgement frame and tries routing it through the other End node.
    c. Neither End node routes the retransmitted frame, since it still holds the same modified FCS and they are unable to distinguish the contents of the frame.

5. Neither End node answers to the frame, since the modified FCS does not allow to verify its integrity.
    a. Each time the frame is displayed on the Zniffer as a CRC error.
    b. The controller tries retransmitting the frame since it doesn’t receive an Acknowledgement frame and tries routing it through the other End node.
    c. Neither End node routes the retransmitted frame, since it still holds the same modified FCS and they are unable to distinguish the contents of the frame.
6. Neither End node answers to the frame, since the modified FCS does not allow to verify its integrity.
    a. Each time the frame is displayed on the Zniffer as a CRC error.
    b. The controller tries retransmitting the frame since it doesn’t receive an Acknowledgement frame and tries routing it through the other End node.
    c. Neither End node routes the retransmitted frame, since it still holds the same shortened FCS and they are unable to distinguish the contents of the frame.
7. The multicast is not responded with an Acknowledgement frame by either End node.
    a. The FCS in the multicast frame is 8 bits long.
8. Same as in step 3.
9. The multicast is not responded with an Acknowledgement frame by either End node.
    a. The multicast frame is displayed on the Zniffer as a CRC error.
    b. The FCS in the multicast frame is 8 bits long.
    c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*
10. The multicast is not responded with an Acknowledgement frame by either End node.
    a. The multicast frame is displayed on the Zniffer as a CRC error.
    b. The FCS in the multicast frame is more than 8 bits long and makes the frame end in a series of zeroes.*
    c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*
11. The multicast is not responded with an Acknowledgement frame by either End node.
    a. The multicast frame is displayed on the Zniffer as a CRC error.
    b. The FCS in the multicast frame is less than 8 bits long and makes the frame end in a series of zeroes.*
    c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*

* Needs practical verification.

3.44.4 Pass Criteria

1. The FCS can only be 8 bits long at the end of the frame (8.1.3.8)
2. It signalizes the integrity of the frame and when it is modified, the receiver and Zniffer are uncapable of identifying it (8.1.3.8)

3.44.5 Fail Criteria

1. The FCS can be different from 8 bits long at the end of the frame (8.1.3.8)
2. When it is modified, the receiver and Zniffer are capable of identifying it and responding to the frame (8.1.3.8)

3.45 MPDU Format, Mac Footer (MFR): CRC – 3-channel

CRC is the 16-bit non-correcting Cyclic Redundancy Code (CRC) used for validating the integrity of a frame, used in 3-channel frequencies. It shall be calculated from the HomeID field to the Data Payload, both included.

3.45.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
2 x End node

3.45.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include End nodes to the Controller's Network.
2. Send a singlecast with MPDU = 0x00 (NOP) to each End node.
3. Look for the CRC field in each frame.
4. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its CRC value to a random hardcoded value, send it to each End node separately.
5. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its CRC value to be more than 16 bits, send it to each End node separately.
6. Generate a singlecast frame with MPDU = 0x00 (NOP) and modify its CRC value to be less than 16 bits, send it to each End node separately.
7. Send an unmodified multicast frame with MPDU = 0x00 (NOP) to both End nodes. Look for the CRC in the multicast.
8. Look for the CRC field in each singlecast follow-up frame.
9. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its CRC value to a random hardcoded value, send it to both End nodes. Look for the CRC in the multicast.
10. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its CRC value to be more than 16 bits, send it to both End nodes. Look for the CRC in the multicast.
11. Generate a multicast frame with MPDU = 0x00 (NOP) and modify its CRC value to be less than 8 bits, send it to both End nodes. Look for the CRC in the multicast.

3.45.3 Test Result

2. Communication is correct with both End nodes and they answer with an Acknowledgement frame each.
3. Make sure there are 16 MFR bits corresponding to the CRC section in each frame.
4. Neither End node answers to the frame, since the CRC does not allow to verify its integrity.
    a. Each time the frame is displayed on the Zniffer as a CRC error.

   b. The controller tries retransmitting the frame since it doesn't receive an Acknowledgement frame and tries routing it through the other End node.
   c. Neither End node routes the retransmitted frame, since it still holds the same modified CRC and they are unable to distinguish the contents of the frame.
5. Neither End node answers to the frame, since the modified CRC does not allow to verify its integrity.
   a. Each time the frame is displayed on the Zniffer as a CRC error.
   b. The controller tries retransmitting the frame since it doesn't receive an Acknowledgement frame and tries routing it through the other End node.
   c. Neither End node routes the retransmitted frame, since it still holds the same modified CRC and they are unable to distinguish the contents of the frame.
6. Neither End node answers to the frame, since the modified CRC does not allow to verify its integrity.
   a. Each time the frame is displayed on the Zniffer as a CRC error.
   b. The controller tries retransmitting the frame since it doesn't receive an Acknowledgement frame and tries routing it through the other End node.
   c. Neither End node routes the retransmitted frame, since it still holds the same shortened CRC and they are unable to distinguish the contents of the frame.
7. The multicast is not responded with an Acknowledgement frame by either End node.
   a. The CRC in the multicast frame is 16 bits long.
8. Same as in step 3.
9. The multicast is not responded with an Acknowledgement frame by either End node.
   a. The multicast frame is displayed on the Zniffer as a CRC error.
   b. The FCS in the multicast frame is 16 bits long.
   c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*
10. The multicast is not responded with an Acknowledgement frame by either End node.
   a. The multicast frame is displayed on the Zniffer as a CRC error.
   b. The CRC in the multicast frame is more than 8 bits long and makes the frame end in a series of zeroes.*
   c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*
11. The multicast is not responded with an Acknowledgement frame by either End node.
   a. The multicast frame is displayed on the Zniffer as a CRC error.
   b. The CRC in the multicast frame is less than 12 bits long and makes the frame end in a series of zeroes.*
   c. The singlecast follow-up frames are displayed correctly and responded by each End node with Acknowledgemnt frames.*

* Needs practical verification.

3.45.4 Pass Criteria

1. The CRC can only be 16 bits long at the end of the frame (8.1.3.9)
2. It signalizes the integrity of the frame and when it is modified, the receiver and Zniffer are uncapable of identifying it (8.1.3.9)

3.45.5 Fail Criteria

1. The CRC can be different from 16 bits long at the end of the frame (8.1.3.9)
2. When it is modified, the receiver and Zniffer are capable of identifying it and responding to the frame (8.1.3.9)

3.46 MPDU Format, Beam Frame Format, 2-channel

Beam frames are used to awake Frequently Listening (FL) nodes. They are transmitted back to back to ensure an FL node can detect a beam within a short time window. Each beam frame shall carry the Beam Tag and NodeID fields. The NodeID field should be followed by the optional HomeID Hash field. HomeID Hash values 0x0A, 0z4A, 0x55 shall be accepted as potential match for the actual HomeID. An FL node shall stay awake to receive the MPDU that follows if there is a match with the Hash or NodeID, else it may return to sleep.

3.46.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End nodes

3.46.2 Test Setup

1. Include FLiRS End node to the Controller's Network.
2. Send right away a singlecast with MPDU 0x00 (NOP) to the End node.
3. Wait 10 seconds to make sure FLiRS is asleep. Send the frame again.
4. Observe the Beam frame is sent to the End node.
5. Obsere the Beam Stop frame.
6. Once the beaming reaches the FLiRS in a waking up state, the FLiRs stays awake so that the Controlle tries with a transmission of the original singlecast again.

3.46.3 Test Result

2. Communication is correct and FLiRS End node answers with an Acknowledgement frame.
3. Observe that Controller can't deliver the frame and retransmits it.
    a. As soon as it fails, it starts sending Beam frames to the FLiRs End node.
4. The Beam is shown correctly in the Zniffer.
5. The Beam Stop frame shows the beam count of 230.
6. After the beaming, the Controller tries with the original singlecast sent and the FLiRS End node answers with an Acknowledgement frame in response.

3.46.4 Pass Criteria

1. The Beam frame consists of: (8.1.3.10 – Figure 8-18)
    a. A Beam tag: 0x55 (1 byte). (8.1.3.10 – Table 8-17)
    b. Destination NodeId: 0x01 .. 0xE8, 0xFF (1 byte). (8.1.3.10 – Table 8-17)

   c. Field “HomeID Hash Included”: 0x01 (1 byte). (8.1.3.10 – Table 8-17)
   d. Home ID Hash: the actual Hash or 0x0A, 0x4A or 0x55. (8.1.3.10 – Table 8-17)

3.46.5 Fail Criteria

1. Any of the elements of the Beam frame deviates from the description (8.1.3.10 – Table 8-17)

3.47 MPDU Format, Beam Frame Format, 3-channel

Beam frames are used to awake Frequently Listening (FL) nodes. They are transmitted back to back to ensure an FL node can detect a beam within a short time window. Each beam frame shall carry the Beam Tag and NodeID fields. The NodeID field should be followed by the optional HomeID Hash field. HomeID Hash values 0x0A, 0z4A, 0x55 shall be accepted as potential match for the actual HomeID. An FL node shall stay awake to receive the MPDU that follows if there is a match with the Hash or NodeID, else it may return to sleep.

3.47.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End nodes

3.47.2 Test Setup

1. Include FLiRS End node to the Controller’s Network.
2. Send right away a singlecast with MPDU 0x00 (NOP) to the End node.
3. Wait 10 seconds to make sure FLiRS is asleep. Send the frame again.
4. Observe the Beam frames are sent to the End node.
5. Obsere the Beam Stop frames.
6. Once the beaming reaches the FLiRS in a waking up state, the FLiRs stays awake so that the Controlle tries with a transmission of the original singlecast again.

3.47.3 Test Result

2. Communication is correct and FLiRS End node answers with an Acknowledgement frame.
3. Observe that Controller can’t deliver the frame and retransmits it.
   a. As soon as it fails, it starts sending Beam frames to the FLiRs End node.
4. The Beam is shown correctly in the Zniffer.
5. The Beam Stop frames show the beam count of 121.
6. After the beaming, the Controller tries with the original singlecast sent and the FLiRS End node answers with an Acknowledgement frame in response.

3.47.4 Pass Criteria

1. The Beam frame consists of: (8.1.3.10 – Figure 8-18)
   a. A Beam tag: 0x55 (1 byte). (8.1.3.10 – Table 8-17)

b. Destination NodeId: 0x01 .. 0xE8, 0xFF (1 byte). (8.1.3.10 – Table 8-17)
c. Field “HomeID Hash Included”: 0x01 (1 byte). (8.1.3.10 – Table 8-17)
d. Home ID Hash: the actual Hash or 0x0A, 0x4A or 0x55. (8.1.3.10 – Table 8-17)

3.47.5 Fail Criteria

1. Any of the elements of the Beam frame deviates from the description (8.1.3.10 – Table 8-17)

3.48 MPDU Format, Fragmented Frame Format, 3-channel frequency

A Beam Fragment is used for 3-channel frequencies. It comprises a number of beam frames. The Beam Fragment duration is in the range 110-115 ms. Beam frames shall be sent back to back to ensure the FL node can detect it upon waking up. The next Beam Fragment shall begin 190 – 200 ms after the beginning of the previous one. They shall be sent in different channels. When recognizing a Beam the receiving node shall answer with an Acknowledgement frame, upon receiving it, the Controller shall send the original singlecast but only if the Acknowledgement frame matches the originating HomeId and the NodeID of the destination NodeID on the original singlecast. A Beam Fragment can be addressed to 0xFF (255) turning it into a broadcast Wake Up Beam, but it can’t be answered directly by the End node.

3.48.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End nodes

3.48.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include FLiRS End node to the Controller’s Network.
2. Wait 10 seconds for FLiRs to sleep. Send a singlecast with MPDU 0x00 (NOP) to the End node.
3. Generate a Wake up Beam Frame with Beam Tag different from 0x55, send it to the End node.
4. Generate a Wake Up Beam Frame with a Destination ID different from the End node, send it to the End node.
5. Generate a Wake Up Beam Frame with a Destination ID 0xFF “Broadcast”, send it to the End node.
6. Generate a Wake Up Beam Frame with a random HomeID Hash hardcoded, send it to the End node.
7. Generate a Wake Up Beam Frame with a HomeID Hash hardcoded with values: 0x0A, 0x4A or 0x55, send each of them to the End node.
8. Generate a Wake Up Beam Frame and set it to be delayed more 300ms, send it to the End node.

3.48.3 Test Result

2. Observe there are 3 wake up Beam frames sent to the End node.
    a. Each Beam Fragment is sent between 80 and 90ms after the Beam Stop from the previous Beam Fragment (as to begin between 190 – 200 ms from the beginning of the previous Beam Fragment).
    b. Each Wake Up Beam Start is sent in a different channel than the corresponding Wake Up Beam Stop.
    c. When the End node recognizes the Beam, it answers with an Acknowledgement frame.
    d. Controller repeats the original singlecast.
3. Observe there are 3 wake up Beam frames sent to the End node.
    a. The End node never recognizes the beam and the Controller continues sending the Wake Up Fragment until it times out.
4. Observe there are 3 wake up Beam frames sent to the End node.
    a. The End node never recognizes the beam and the Controller continues sending the Wake Up Fragment until it times out.
5. Observe there are 3 wake up Beam frames sent to the End node.
    a. The End node recognizes the beam but does not respond to the “Broadcast” Wake Up Fragment with an Acknowledgemnt Frame, the Controller continues sending the Wake Up Fragment until it times out.
6. Observe there are 3 wake up Beam frames sent to the End node.
    a. The End node never recognizes the beam and the Controller continues sending the Wake Up Fragment until it times out.
7. Same Result as in step 2. The End node actually accepts the frame and recognizes it.
8. Observe there are 3 wake up Beam frames sent to the End node with 300ms between transmissions.
    a. The End node never manages to catch the beam when waking up and the Controller continues sending the Wake Up Fragment until it times out.*

   * Needs practical verification.

3.48.4 Pass Criteria

1. There are more than one wake up Beam frames sent to the End node (8.1.3.11).
2. The fragment lasts between 110-115ms (8.1.3.11).
3. There are between 190 – 200ms between two WakeUp Beam Start in the Fragment (8.1.3.11).
4. The Fragments are sent in different Channels (8.1.3.11).
5. The Beams can be addressed to any node from 1 to 232 and 255 (8.1.3.11).
6. The Receiving Node can validate a Hash of the HomeID or values 0x0A, 0x4A or 0x55 (8.1.3.10).
7. The End node answers the Beam with an Acknowledgement frame and the Controller repeats the original singlecast (8.1.3.11).
8. It’s used only in 3-channel frequencies (8.1.3.15)

3.48.5 Fail Criteria

1. Any of the Pass Criteria is not met.

3.49 MPDU Format, Continuous Beam Format, 2-channel frequency

A continuous beam is used for 2-channel frequencies. It's a series of Beam Frames sent over a period of time. They shall be sent back to back to prevent interruptions. It can address any node from 1 to 232 and 255. It may be a short continuous beam up to 300ms or a long continuous beam up to 1160ms. IT shall always be followed by a singlecast frame.

3.49.1 Prerequisites

1 x Z-Wave Zniffer
1 x Z-Wave PC Controller
1 x FLiRS End nodes

3.49.2 Test Setup

We assume a Frame Generator is available in order to generate frames with individual bits, bytes or sections modified individually in order to test the behavior of the receiver.

1. Include FLiRS End node to the Controller's Network.
2. Wait 10 seconds for FLiRs to sleep. Send a singlecast with MPDU 0x00 (NOP) to the End node.
3. Generate a Wake up Beam Frame with Beam Tag different from 0x55, send it to the End node.
4. Generate a Wake Up Beam Frame with a Destination ID different from the End node, send it to the End node.
5. Generate a Wake Up Beam Frame with a Destination ID 0xFF "Broadcast", send it to the End node.
6. Generate a Wake Up Beam Frame with a random HomeID Hash hardcoded, send it to the End node.
7. Generate a Wake Up Beam Frame with a HomeID Hash hardcoded with values: 0x0A, 0x4A or 0x55, send each of them to the End node.
8. Generate a Wake Up Beam Frame and set it to be sent during less than 300ms, send it to the End node.

3.49.3 Test Result

2. Observe there is 1 wake up Beam frame sent to the End node.
    a. The Continuous Beam sends the Start and the Stop with approximately 1100ms of difference.
    b. When the Controller finishes the Continuous Beam, it repeats the original singlecast.
    c. Since the End node recognizes the Beam, it answers with an Acknowledgement frame.
3. Observe there is 1 wake up Beam frame sent to the End node.
    a. The End node never recognizes the beam and never wakes up.
    b. The Controller continues sending the Wake Up Continuous Beam.
    c. The Controller tries sending the singlecast again unsuccessfully.
    d. The Controller sends Continuous Beam followed by the singlecast until it times out.
4. Observe there is 1 wake up Beam frame sent to the End node.
    a. The End node never recognizes the beam and never wakes up.
    b. The Controller continues sending the Wake Up Continuous Beam.
    c. The Controller tries sending the singlecast again unsuccessfully.
    d. The Controller sends Continuous Beam followed by the singlecast until it times out.
5. Observe there is 1 wake up Beam frame sent to the End node.
    a. The End node recognizes the beam but does not respond to the retransmission of the sinlecast.
    b. “Broadcast” Wake Up Continuous Beam with an Acknowledgemnt Frame.
    c. The Controller continues sending the Wake Up Fragment until it times out.
6. Observe there is 1 wake up Beam frame sent to the End node.
    a. The End node never recognizes the beam and never wakes up.
    b. The Controller continues sending the Wake Up Continuous Beam.
    c. The Controller tries sending the singlecast again unsuccessfully.
    d. The Controller sends Continuous Beam followed by the singlecast until it times out.
7. Same Result as in step 2. The End node actually accepts the frame and recognizes it.
8. Observe there is 1 wake up Beam frame sent to the End node.
    a. The Controller sends Continuous Beam Start and the Stop with approximately less than 300ms of difference.
    b. The End node never manages to catch the beam when waking up and the Controller continues sending the Continuous Beam until it times out.*

    * Needs practical verification.

3.49.4 Pass Criteria

1. There is 1 wake up Continuous Beam frame sent to the End node (8.1.3.13).
2. The Continuous Beam lasts between 300-1160ms (8.1.3.13).
3. The Continuous Beams are sent in different Channels.
4. The Beams can be addressed to any node from 1 to 232 and 255 (8.1.3.13).
5. The Receiving Node can validate a Hash of the HomeID or values 0x0A, 0x4A or 0x55 (8.1.3.10).
6. After the Continuous Beam, the Controller repeats the original singlecast and The End node answers with an Acknowledgement frame if it validated the Beam correctly (8.1.3.13).
7. It’s used only in 2-channel frequencies (8.1.3.14).

3.49.5 Fail Criteria

1. Any of the Pass Criteria is not met.

REFERENCES

[1] ITU-T G.9959, Short range narrowband digital radiocommunication transceivers – PHY & MAC layer specifications

