Checking the OpenLCB Message Network Standard

January 5, 2025

1 Introduction

This note documents the procedure for checking an OpenLCB implementation against the OpenLCB Message Network Standard.

The checks are traceable to specific sections of the Standard.

The checking assumes that the Device Being Checked (DBC) is being exercised by other nodes on the message network, e.g. is responding to enquiries from other parts of the message network.

2 Message Network Procedure

2.1 Node Initialized checking

This section checks the interaction in Standard section 3.4.1 Node Initialization.

It does this by having the operator reset/restart the DBC and then checking that a Node Initialized message is received and

1. The Node Initialized message is the first message received. [1]

2. The message indicates its source is the DBC.

3. The message uses the appropriate MTI [2] for whether PIP indicates the node uses the Simple Protocol or not.

4. The data part of the message contains the source node ID.

---

[1] See Section 3.2 of the Standard
[2] See Section 3.3.1 of the Standard

2.2 Verify Node checking

This section checks the interaction in Standard section 3.4.2 Node ID Detection.

It does this by

1. Sending a global Verify Node ID message and checking the reply.

   (a) The reply message indicates its source is the DBC.

   (b) The reply message contains the Node ID of the DBC in its data field.

   (c) The reply message uses the appropriate MTI [3] for whether PIP indicates the node uses the Simple Protocol or not.

2. Sending a Verify Node ID message addressed to the DBC and checking the reply.

   (a) The reply message indicates its source is the DBC.

   (b) The reply message contains the Node ID of the DBC in its data field.

   (c) The reply messages uses the appropriate MTI for whether PIP indicates the node uses the Simple Protocol or not.

3. Sending a Verify Node ID message addressed to a different address from the DBC and checking for the absence of a reply.

2.3 Protocol Support Inquiry checking

This section checks the interaction in Standard section 3.4.3 Protocol Support Inquiry and Response.

It does this by

1. Sending a Protocol Support Inquiry message to the node, and checking for the resulting Protocol Support Response message.

   (a) The reply message is addressed to the checking node.

   (b) The reply message indicates its source is the DBC.

   The check displays the resulting list of supported protocols for checking against the DBC's documentation.

2. Sending a Protocol Support Inquiry message addressed to a different address from the DBC and checking for the absence of a reply.

---

[3]See Section 3.3.3 fo the Standard

3. Sending three Protocol Support Inquiry messages from three separate node IDs as fast as possible, i.e. at line rate allowing for interleaved replies, followed by checking for three replies. These can be either Protocol Support Response message(s), Terminate Due to Error message(s) or Optional Interaction Rejected[4] message(s). In either case, the mandatory fields in the messages are checked for correctness.

2.4 Optional Interaction Rejected checking

This section checks the interaction in Standard section 3.5.1 Reject Addressed Optional Interaction.

First it sends a global unallocated MTI [5] and checks for no reply.

It then sends an unallocated MTI [6] addressed to a checker node (i.e. not the DBC) and checks for no reply.

Finally, it sends an unallocated MTI addressed to the DBC and checking for the Optional Interaction Rejected message in reply.

1. The reply message is addressed to the checking node.

2. The reply message indicates its source is the DBC.

3. The reply message carries at least four bytes of content, with the third and fourth bytes carrying the original MTI.

2.5 Duplicate Node ID Discovery checking

This section checks the interaction in Standard section 3.5.4 Duplicate Node ID Discovery.

It does this by sending a global Verified Node ID message with the identified node ID set to the DBC's ID. This message should have only been emitted by the DBC.

The checker then looks for the well-known global event. If it finds that, the check passes. If not, it prompts the operator to confirm that indication has been made "using whatever indication technology is available", e.g. via LEDs or other form.

---

[4]Protocol Support Inquiry is not an optional message per se, so this check may be excessively accepting. See section 3.5.3 and section 3.5.1 of the Message Network Standard for background.

[5]Initially 0x030. Others may be added in the future for an eventual check of all possible unallocated MTIs

[6]Initially 0x048. Others may be added in the future for an eventual check of all possible unallocated MTIs

3   Frame Level Procedure

This section is only applicable to implementations what use a CAN-format link layer implementation, e.g. via a USB-CAN adapter or by using Grid Connect coding over a TCP/IP link. See Standard section 7.

All checks in the prior section are assumed to have passed before these checks are run, as they depend on message-layer behaviors.

In general, proper operation of the frame level is checked by exercising it through the message level in the checks above and in subsequent documents. We limit ourselves here to checking atypical behaviors.

3.1   Reserved Field Handling

This section discusses the reserved bit fields described in Standard section 3.1.1.

The CAN implementation does not carry bits 15-14 nor bit 12 of the MTI. The full 12 bits of the rest of the MTI are checked in other sections of this check plan.

However, in a CAN implementation, the 0x1_0000_0000 bit is reserved, send as 1, don't check on receipt.

To check this, the checker sends a Verify Node ID Global with zero in the 0x1_0000_0000 bit. It then checks:

1. That a Verified Node ID message frame is received from the DBC,

2. That the 1_0000_0000 bit in that frame is a one.