Checking the OpenLCB Event Transport Protocol Standard

The OpenLCB Group

February 10, 2024

1 Introduction

This note documents the procedure for checking an OpenLCB implementation against the Event Transport 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.

1.1 Required Equipment

See the separate “Installing the OpenLCB Checker Software” document for initial installation and set up of the checker program.

If a direct CAN connection will be used, a supported USB-CAN adapter [1] is required. Connect the adapter to the DBC using a single UTP cable and connect two CAN terminators.

Provide power to the DBC using its recommended method.

2 Set Up

The following steps need to be done once to configure the checker program:.

1. Start the checker configuration program.
2. Select “Set Up”.
3. Get the Node ID from the DBC[2]

[1] See “Installing the OpenLCB Checker Software”

[2] Where do we require this to be marked on a node?

4. Enter that Node ID into the program.
5. Configure the checker program for the USB-CAN adapter's device address or the TCP hostname and port.
6. Return from the setup section and reply "Y" to "Save configuration?" when prompted.
7. Quit from the program.

The following steps need to be done at the start of each checking session.

1. Check that the DBC is ready for operation.
2. Start the checker program.

3 Event Transport Procedure

Select "Event checking" in the check program, then select each section below in turn. Follow the prompts for when to reset/restart the node and when to check outputs against the node documentation.

A node which does not self-identify in PIP that it supports Event Transfer will be deemed to have passed these checks. [3]

A node which does self-identify in PIP that it supports Event Transfer is expected to consume or produce at least one event. The checks are structured to check for that.

Note: Proper handling of known events should be addressed.

Note: This does not address the proper use of Unique IDs for Event IDs.

3.1 Identify Events Addressed

This section checks the addressed interaction in Standard section 6.2 and the message formats in Standard section 4.3 through 4.8.

The check starts by sending an Identify Events message addressed to the DBC. It then checks

1. That one or more Producer Identified, Producer Range Identified, Consumer Identified and/or Consumer Range Identified messages are returned,
2. That those show the DBC node as their source.

[3]Using the -p option or setting the checkpip default value False will skip this check.

3.2 Identify Events Global

This section checks the unaddressed (global) interaction in Standard section 6.2 and the message formats in Standard section 4.3 through 4.8.

The check starts by sending an Identify Events unaddressed (global) message. It then checks

1. That one or more Producer Identified, Producer Range Identified, Consumer Identified and/or Consumer Range Identified messages are returned,
2. That those show the DBC node as their source,
3. That these identify the same events produced and consumed as the addressed form of the Identify Events message.

3.3 Identify Producer

This section checks the interaction in Standard section 6.3, and the message formats in Standard section 4.5 through 4.7.

The check proceeds by sending multiple Identify Producers messages for the zero or more individual event IDs returned by an Identify Events message addressed to the DBC. If there are none of these, this check passes. If there are one or more, it then checks:

1. That exactly one reply is received for each Identify Producers message sent.
2. That those show the DBC node as their source,
3. That these identify the same event ID as the corresponding Identify message.

3.4 Identify Consumer

This section checks the interaction in Standard section 6.4, and the message formats in Standard section 4.2 through 4.4.

The check proceeds by sending multiple Identify Consumers messages for the zero or more individual event IDs returned by an Identify Events message addressed to the DBC. If there are none of these, this check passes. If there are one or more, it then checks:

1. That exactly one reply is received for each Identify Consumers message sent.
2. That those show the DBC node as their source,
3. That these identify the same event ID as the corresponding Identify message.

3.5 Initial Advertisement

Follow the prompts when asked to reset or otherwise initialize the DBC.

This section's checks the interaction in the preamble to Standard section 6, and the messages in Standard sections 4.1, 4.3, 4.4, 4.6 and 4.7.

Note: There's no requirement that the Identify Producer messages be sent immediately, only that they be sent before the events are produced. Nodes typically send them immediately, and that's what this checks.

The checks starts by restarting the node, which causes a transition to Initialized state. That is then followed by the node identifying events that it will produce and consume by appropriate messages. It then checks:

1. That the Producer Identified, Producer Identified Range, Consumer Identified and Consumer Identified Range messages produced at node startup are the same as the ones emitted in response to an addressed Identify Events,
2. That those messages show the DBC as their source.
3. That no Producer Consumer Event Report messages are sent before the corresponding producer has been identified.
4. That any PCER messages that are sent show the DBC as their source.