Checking the OpenLCB Datagram Transport Protocol

The OpenLCB Group

February 10, 2024

1 Introduction

This note documents the procedure for checking an OpenLCB implementation against the Datagram 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 Datagram Transport Procedure

Select "Datagram checking" in the checkere program, then select each section below in turn. Follow the prompts for when to check outputs against the node documentation.

Note that this process is unable to check datagrams from the DBC to the checker. There is no standard way to elicit those. They will be checked as part of the Memory Configuration Protocol checking, if applicable.

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

3.1 Datagram Reception

This checks the messages in Standard sections 4.1 and 4.2 or 4.3, including the error code constraints in Standard section 4.3.1 [4] and the interactions in sections 6.1 or 6.2.

The checker will send a datagram to the DBC. The DBC in turn will either accept the datagram (sections 4.2 and 6.1) or reject it (sections 4.3 and 6.2). Either response is acceptable.

This section's checks cover:

1. If the datagram is accepted,
   (a) That the Datagram OK message is received,
   (b) That the Datagram OK message has proper source and destination node IDs,
   (c) that the Datagram OK message contains exactly 1 byte of flags,

[3] Using the -p option or setting the checkpip default value False will skip this check.
[4] This section references the Message Transport Standard, section 3.5.5

(d) that the 0x70 bits of the flags are zeros.

2. if the datagram is rejected,

   (a) That the Datagram Rejected message is received,

   (b) That the Datagram Rejected message has proper source and destination node IDs,

   (c) that the Datagram Rejected message contains at least two bytes of error code,

   (d) that the 0xF000 bits of the error code are either 0x1000 or 0x2000,

   (e) that the 0x0F00 bits of the error code are zeros.

This process is repeated for datagrams of length 1, 10 and 72 bytes to exercise the full range of datagram framing.