Checking the OpenLCB Train Control Protocol Standard

January 2, 2025

1 Introduction

This note documents the procedure for checking an OpenLCB implementation against the Train Control Protocol 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 Train Control Protocol Procedure

This procedure checks the operation of a Train Node.

A node which does not self-identify in PIP that it supports the Train Control Protocol should be considered to have passed these checks.

Note that this commands the speed and functions of a locomotive node. Although the commanded speed is quite low and only for a short time, the checks should be run in a way that the associated physical locomotive does not run away.

2.1 Defined Event ID checking

This section checks that the node supports the isTrain event defined in section 4.1 and 6.4 of the Train Control Protocol Standard.

It does this by issuing an Identify Events to the node, and then checking for a Producer Identified reply carrying the isTrain Event ID.

2.2 Check set and query speeds

This checks the ability to set and query the speed and direction of the train. This interaction is defined in sections 4.3 and 4.4 of the Standard. Note that it checks that forward/reverse is independent of the speed setting, particularly at zero.

1. Set the speed and direction to 0.75 reverse.
2. Query the speed and direction. Check for 0.75 reverse.
3. Set the speed and direction to 0 reverse.
4. Query the speed and direction. Check for 0 reverse.
5. Set the speed and direction to 0.75 forward.
6. Query the speed and direction. Check for 0.75 forward.
7. Set the speed and direction to 0 forward.
8. Query the speed and direction. Check for 0 forward.

In any case, even if the earlier checks failed, end by setting the speed and direction to 0 forward.

2.3 Check set and query of functions

This section assumes that F0 is available on the train. The FDI is not used to confirm that.

The message and reply are defined in sections 4.3 and 4.4 of the Standard; the interaction is defined in section 6.3.

1. Set F0 to on.
2. Query F0 and check for an “on” response.
3. Set F0 to off.
4. Query F0 and check for an “off” response.

In any case, even if the earlier checks failed, end by setting F0 to “off”.

2.4 Check Emergency Stop

This checks the behavior defined in section 6.2 of the Standard. It does not check that the train stops.

1. Set the speed and direction to 0.1 scale m/s reverse.

2. Query the speed and direction. Check for 0.1 scale m/s reverse.
3. Send an emergency stop command to the train.
4. Query the speed and direction. Check for 0 scale m/s reverse.
5. Set the speed and direction to 0.1 scale m/s forward.
6. Query the speed and direction. Check for 0.1 scale m/s forward.
7. Set the speed and direction to 0 scale m/s forward.

In any case, even if the earlier checks failed, end by setting the speed and direction to 0 scale m/s forward.

2.5 Check Global Emergency Stop

This checks the behavior defined in section 6.2 of the Standard. It does not check that the train stops.

1. Set the speed and direction to 0.1 scale m/s reverse.
2. Query the speed and direction. Check for 0.1 scale m/s reverse.
3. Produce an Emergency Stop All event.
4. Query the speed and direction. Check for 0.1 scale m/s reverse.
5. Set the speed and direction to 0.1 scale m/s forward.
6. Query the speed and direction. Check for 0.1 scale m/s forward.
7. Produce a Clear Emergency Stop event.
8. Set the speed and direction to 0 scale m/s forward.

In any case, even if the earlier checks failed, end by setting the speed and direction to 0 scale m/s forward.

2.6 Check Global Emergency Off

This checks the behavior defined in section 6.2 of the Standard. It does not check that the train stops and that the train's outputs power off.

1. Set the speed and direction to 0.1 scale m/s reverse.
2. Query the speed and direction. Check for 0.1 scale m/s reverse.
3. Produce an Emergency Off All event.
4. Query the speed and direction. Check for 0.1 scale m/s reverse.

5. Set the speed and direction to 0.1 scale m/s forward.
6. Query the speed and direction. Check for 0.1 scale m/s forward.
7. Produce a Clear Emergency Off event.
8. Set the speed and direction to 0 scale m/s forward.

In any case, even if the earlier checks failed, end by setting the speed and direction to 0 scale m/s forward.

2.7 Check memory spaces

This section checks the existence and properties of the memory spaces defined in section 7 of the Train Control Protocol Standard.

Note that the 0xF9 space is optional. An info message is issued if the 0xF9 space is not present, but the check still passes. The 0xFA space is required for the check to pass.

This does not check the content of the 0xFA space defined by the Function Definition Information Standard.

1. Use a Get Address Space Information Command datagram from the Memory Configuration protocol to enquire about the status of space 0xF9.
2. Check that the response is complete and indicates the memory space exists.
3. Use a Get Address Space Information Command datagram from the Memory Configuration protocol to enquire about the status of space 0xFA.
4. Check that the response is complete and indicates the memory space exists.

2.8 Checking function to/from memory space connection

This section assumes that the F0 function is available in the train node. The FDI is not used to confirm that.

Note that the 0xF9 space is optional. An info message is issued if the 0xF9 space is not present, but the check still passes.

This uses the message definition from section 4.3 of the Standard and the interaction defined in section 6.3.

1. Set function 0 “off” with a Set Function command.
2. Write byte 0 in the 0xF9 memory space to 1.
3. Check that function 0 is “on” with a Query Function command.

Note the language in the Standard that says "A Train Node representing a DCC locomotive ... may, but is not required to provide the last written data upon a read command." This prevents checking that a Set Function command changes the corresponding value in the F9 memory space.

2.9 Check Controller Configuration command and response

1. Send a Set Speed 0 to the node being checked.
2. Send a Train Control Assign Controller command with the checker's Node ID.
3. Check for a Train Control Assign Controller reply with OK in the flags.
4. Send a Train Control Query Controller command
5. Check for a Train Control Query Controller response with the checker's Node ID in the Active Controller field.
6. Send a Train Control Release Controller command
7. Check for a Train Control Query Controller response with a zero Node ID in the Active Controller field.
8. Send a Train Control Query Controller command
9. Check for a Train Control Query Controller response with zeros in the Active Controller field.

In any case, including if some check fails, end by sending a Train Control Release Controller command.

2.10 Check Train Control Management command and response

This assumes that the node being checked has not been reserved by another node.

1. Send a Train Control Management Reserve command
2. Check for a Train Control Management Reserve response with an OK code in the flags.
3. Send a Train Control Management Release command. No response is expected.
4. Send a Train Control Management Reserve command
5. Check for a Train Control Management Reserve response with an OK code in the flags.
6. Send a Train Control Management Reserve command

7. Check for a Train Control Management Reserve response with a fail code in the flags.
8. Send a Train Control Management Release command. No response is expected.

2.11 Check Listener Configuration command and response

This section is still to be written.