CTC LCC Node

Central Traffic Control Layout Command Control Node

www.lccsignals.com

For Version D Series Hardware; Software Version .2 and newer

Table of Contents

Introduction

The CTC Node was created with the intention of enabling modelers and operators to create working, physical reproductions of the prototype control machines quickly and easily. These control machines were found on the prototype from the late 1920s until the 1980s. In the past, creating these machines for model use demanded a significant investment in time, research and spending, which put them out of the reach of many modelers. Virtual CTC panels, such as those that are present in JMRI (Java Model Railroad Interface; jmri.org) are common in the hobby, allowing folks to replicate the physical functionality on a screen using a personal computer.

As a layout matures, being able to model a “real” control machine that can be touched by a dispatcher will put your layout’s uniqueness higher.

Thank you for purchasing the CTC Node. We hope that you will appreciate our hard work in bringing this solution to you!

For support and updates, please see our website at www.lccsignals.com. Collaboration with other users is on the Layout Command Control group on groups.io.

Quick Start

If you have background knowledge on the general theory of operation of CTC panels, this will be enough to get you started. If you are just starting out, or would like a refresher, please read the entire manual to be well prepared as you start this project.

The CTC LCC Node must be connected to your LCC network via the RJ45 style connection jacks at the bottom of the board. An external network terminator is required; there is not one built into the node.

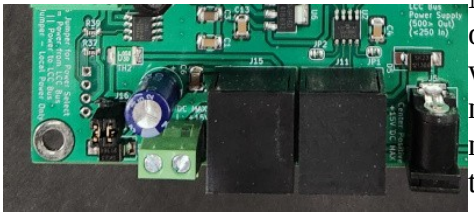


Figure 1: Power Selection Jumpers

Note the power selection jumpers at the bottom left of the board, on the back. If you are using the barrel connector (2.1mm size, with center positive) or terminal block to provide DC power to the node, you can set the jumpers to provide power to the LCC network, or remove the jumpers entirely to only provide power to this individual node.

There is a green LED towards the bottom of the node that will illuminate when the node has the correct power. The Node ID is labeled on the node in this same area. See Figure 1 at right.

Figure 2: Power LED and Node ID Detail

Once connected to the network and powered on, you can use JMRI or other LCC configuration tool to access the CDI and start setting up the events that you would like to use.

On the back of the node at either the left or right edges are micro pushbutton switches that are for the maintainer call, call on and code buttons. These are useful for testing and validation during initial setup. We recommend that you wire up and install normally open pushbutton switches that face the user. If you have purchased one of our ready made front panels, the lower row of holes is intended for a pushbutton to be installed.

Note that flipping either lever or pushing a button will generate an event on to the LCC network. We generally do not recommend a “direct drive” style setup where the same event ID for the lever is used for a switch machine. Our recommended style is that the event IDs generated from lever movements or button pushes are utilized in logic that checks for safety attributes, and then sends the appropriate events to the network for use by the signals and switches. The onboard Cue logic can be used for this, or other LCC nodes, or a PC running JMRI with LogixNG configured that is on the LCC network.

The logic of the panel itself does not include a reference to a turnout being a left or right handed. The logic focuses on “normal”, which is generally closed or straight through and “reversed”, which is generally thrown or diverging. Similar to the switch lever, the panel does not reference a cardinal or map direction, such as north and south, or east and west for the signal (or traffic) levers, but considers traffic moving from left to right OR right to left across the railroad as represented by the portion under control of the machine, and represented on the model board.

For more information on Cue, please refer to that section of this manual.

Hardware Overview

The CTC Node replicates two complete columns of a Union Switch and Signal Centralized Traffic Control machine. Each column has a switch lever at top, with the attendant switch indicator LEDs above; generally green on the left for switch normal, and yellow on right for switch reversed.

Below that are the three traffic indicator LEDs (center red and green for left and right) and the traffic lever.

On the back of the node, there are terminal screw blocks or pin headers that provide a connection point for pushbuttons and LEDs. Each connection is labeled with functionality on the board itself.

There are 6 micro push button switches on each edge of the node for the Code, Call On and Maintainer Call (MC) inputs. These buttons are useful for testing and validation.

The node has 14 outputs to be use for LED indicators, which can be used to show track (block) occupancy on the model board, traffic direction, maintainer call activated, and any other visual indication that you would like you panel to have.

The node also has two outputs to integrate sound modules, such as the CTC approach bell, from Iowa Scaled Engineering. Any sound module that can use a 5V signal to turn on the sound should be compatible. Please reference the support portion of the lccsignals.com website to view the list of known working options.

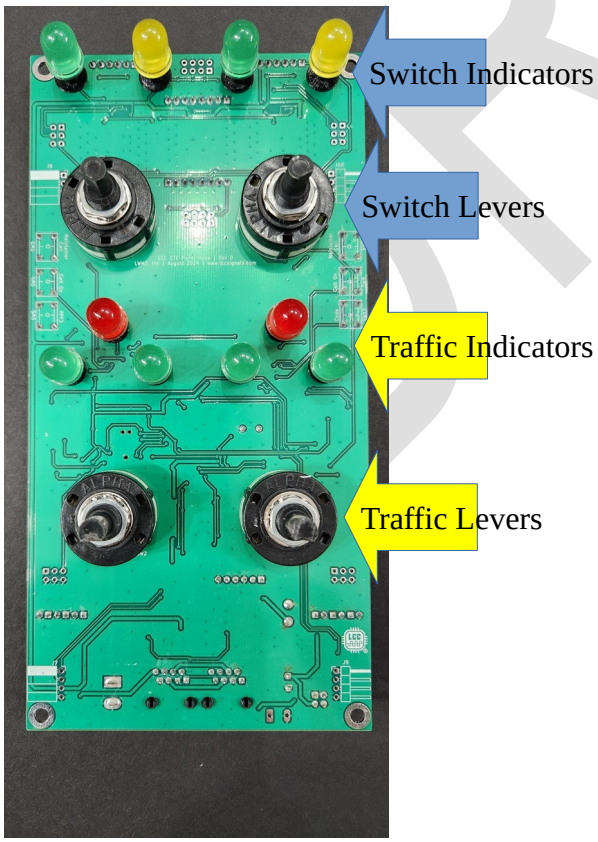


Figure 4: CTC Node Front

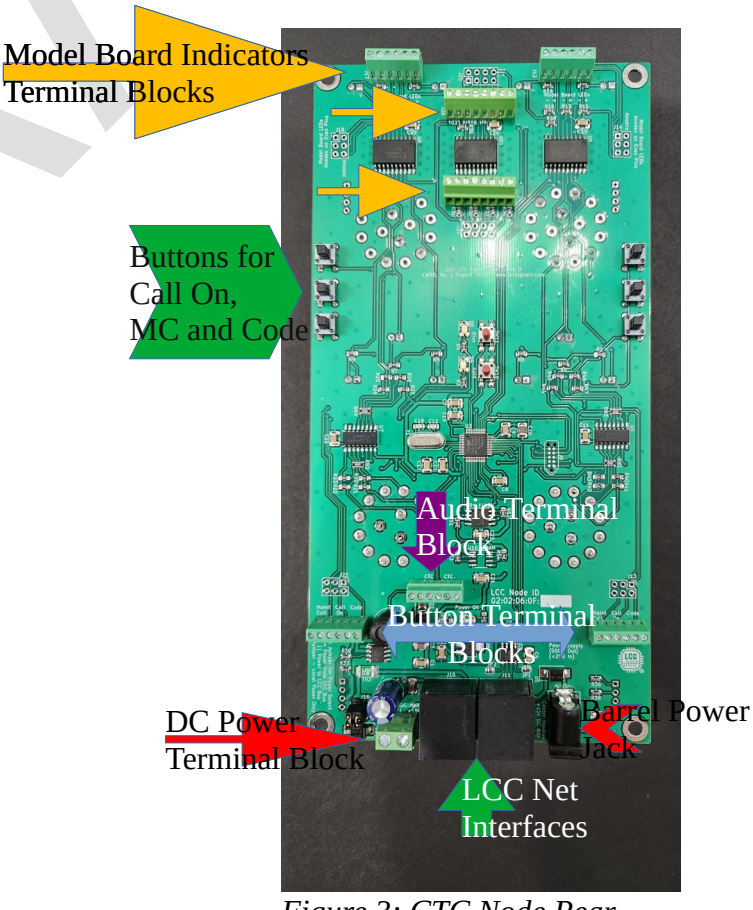


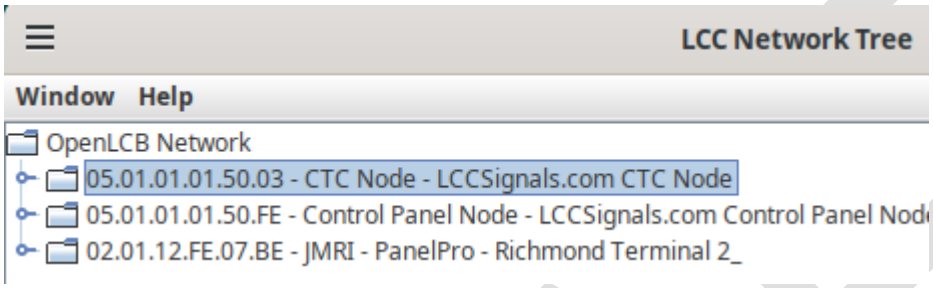
Figure 3: CTC Node Rear

User Configuration Interface (CDI)

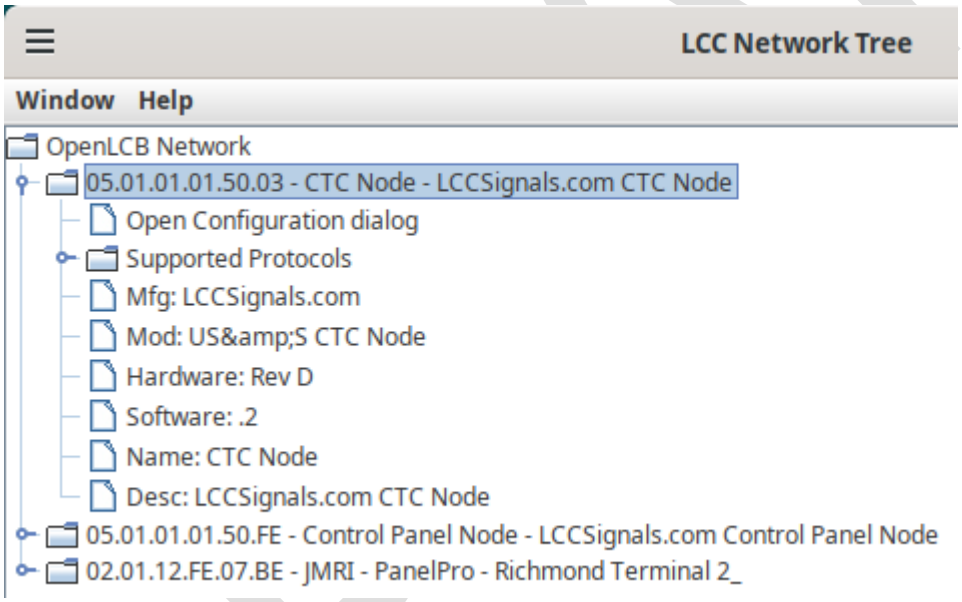
Similar to other Layout Command Control products, the CTC Node offers a user interface to interact with the configuration aspects.

We recommend creating a configuration backup of the node before you begin making changes.

The following screenshots are taken with JMRI's PanelPro.
Find the node in the network list.



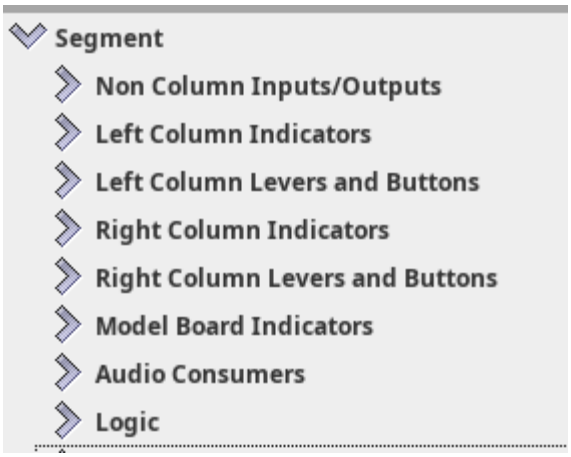
When you expand, you will see details about the node, such as the hardware and software version.



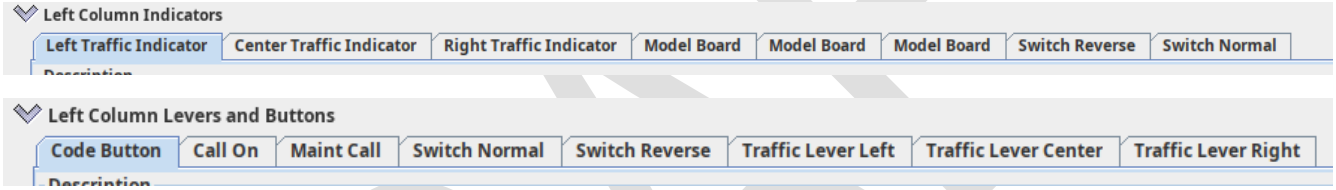
Open the configuration by clicking the "Open Configuration dialog" line. A small dialog will open that the details are being retrieved. This will take anywhere from half a minute to a few minutes, depending on your computer performance, network availability, and if you have added lots of logic blocks that will need to be collected.

Once opened, we recommend going to full screen view, by using your window manager's maximize option on that window.

This interface is presented to the user with the various portions grouped together by column. The dedicated model board and audio outputs have their own sections.

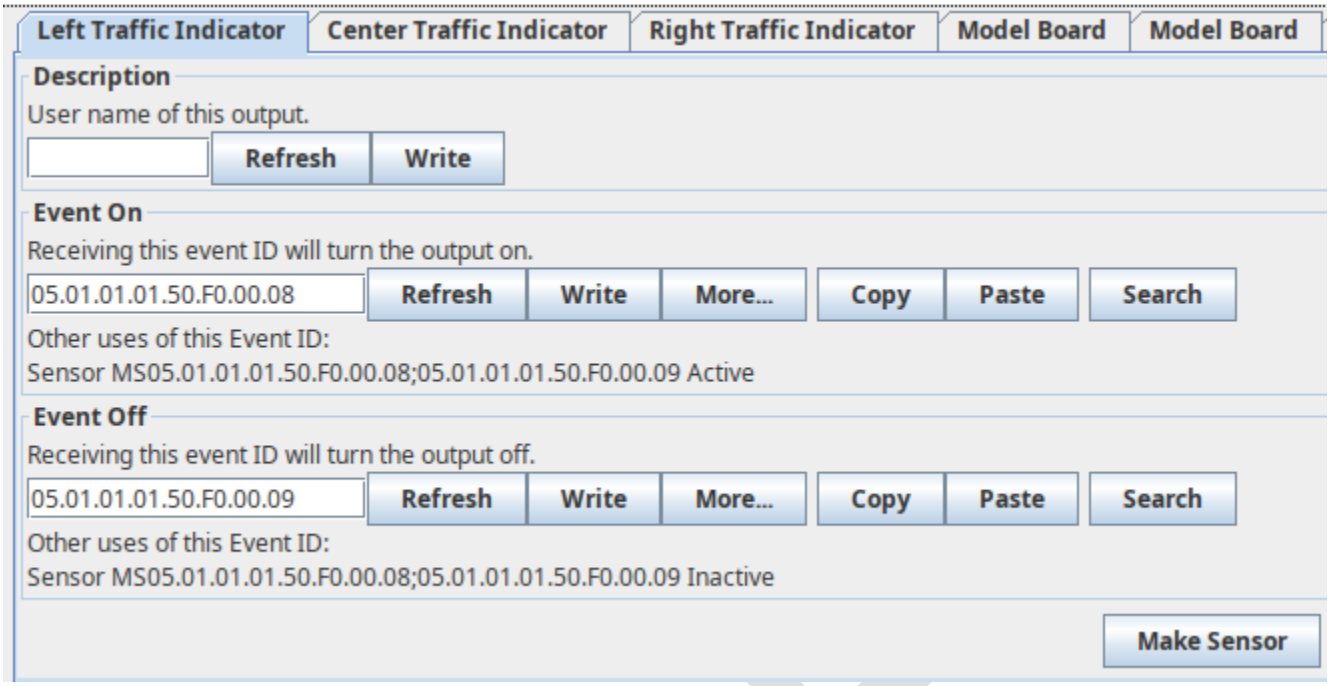


If you expand out any of these groups, the display with change to show you the individual components that are linked to that column.



Note that the order of Indicators or Levers is consistent on both columns. Also, remember that the labeling of Left and Right with respect to the columns is based upon looking at the FRONT of the node, where the levers and indicators (LEDs) are placed and where the user will normally be interacting with the node from. If you are working on your machine from the back, remember that you are looking at a mirrored situation.

You can use the “Make Sensor” quick button in JMRI for the Indicators to help you build out your logic.



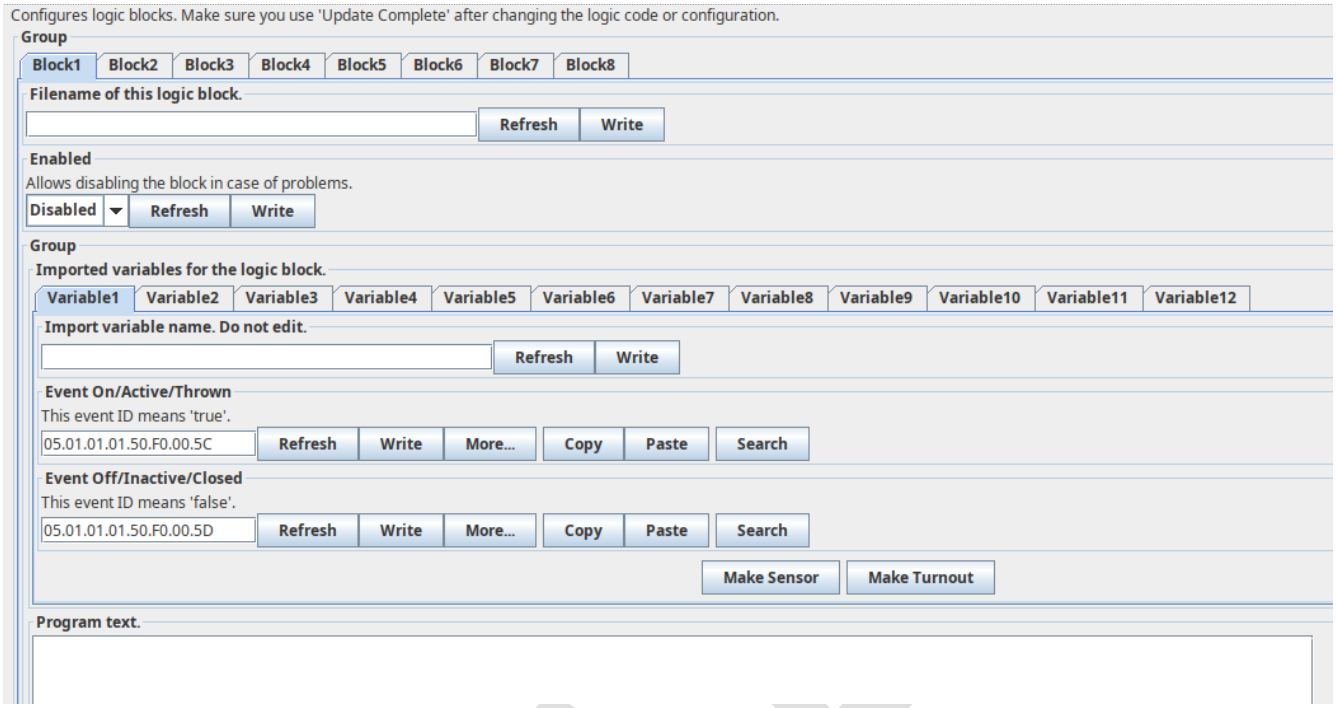
For the input buttons and levers, the Make Sensor button works, but JMRI's Active/Inactive tracking is reversed from the actual state of the Node. This is will be fixed in a future software release.

Cue Logic Blocks

The CTC Node includes a powerful logic feature, called Cue, that allows the user to replicate the functionality of the "relay" logic from the prototype control machines as well as enabling other functionality.

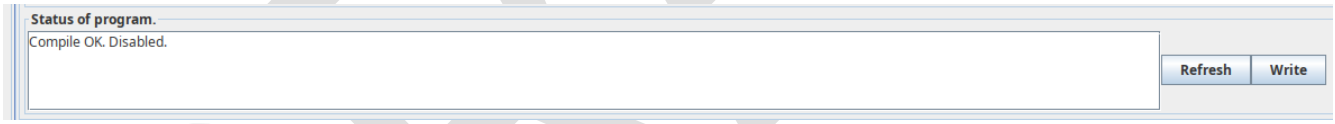
The node will track the state of a linked event ID as either Active Or Inactive. This simple yes or no check enables some sophisticated scenarios to be created.

Before we dive into creating our first logic block, let's review a logic block. You can find the logic blocks at the bottom of the groups in the CDI.



From the top, you can name this logic block, enable or disable the state, view the variable name, view or set the associated event ID, and then the text area for the Cue program that we are going to utilize.

At the bottom of this tab that contains a logic configuration, there is a status box for the state of the program text that we have input.



This Status box can be queried by using the “Refresh” button on the right.

The Status state can tell you if the operation of this block is nominal, or if there is some sort of error, either in the program text, or if something else isn’t quite correct.

Compile OK == Ready to go.

Compile OK. Disabled == Check the status drop down to make sure it is enabled.

Compile Failed == Something isn’t correct inside the program text.

Common issues are “syntax errors” meaning that a line terminating ‘;’ isn’t present, or that you have a mismatch in the number of curly brackets ‘{‘ or ‘}’ used to enclose a statement.

Simple typos, such as using “Of” instead of “Off” can also cause syntax errors to appear. As the node is a computer, it will only do what you tell it, and not what you mean.

Sample Cue Example

The program text should be formatted in this general flow. Start with declaring the variables. The variables are what provide the link to the event IDs. Following the variable, you can write the logic sections.

Here is a simple example that will help illustrate the format of the program text:

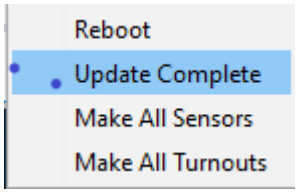
```
exported bool code_button;
exported bool to_lever;
exported bool turnoutled;

if ((code_button is Active) and (to_lever is InActive)) {
code_button = Off;
turnoutled = Off;
} else if ((code_button is Active) and (to_lever is Active) {
code_button = Off;
turnoutled = On;
}
```

From the top, the “exported bool *some_variable*” is where we are telling the node that we will be exporting a boolean variable, meaning tying to a pair of event IDs, with a name of some_variable. The name we choose here will be what appears on the variable tab when we refresh.

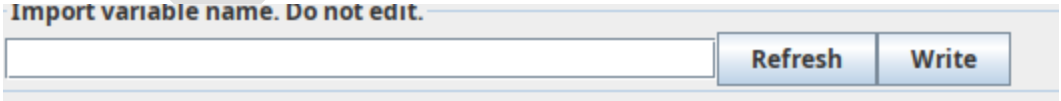
Our logic statements follow. We are checking if the code_button is active (meaning it has been pressed), and if the turnout lever is normal or thrown, by checking for Active or Inactive. You can also use “On” and “Off” as these keywords.

Once we have this saved, and have used the “Update Complete” option in JMRI, we can check that it compiles ok.

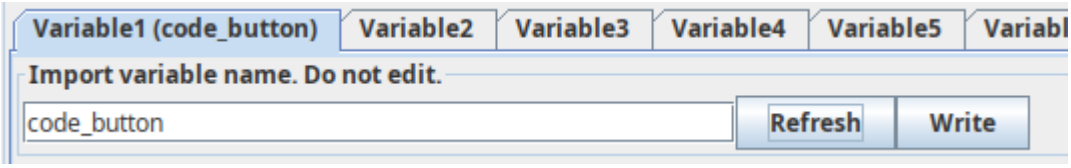


If it does, the next step is to update the event IDs per variable. If the compile failed, that should be fixed first.

So we know what variable is which, we will refresh our view. Before:



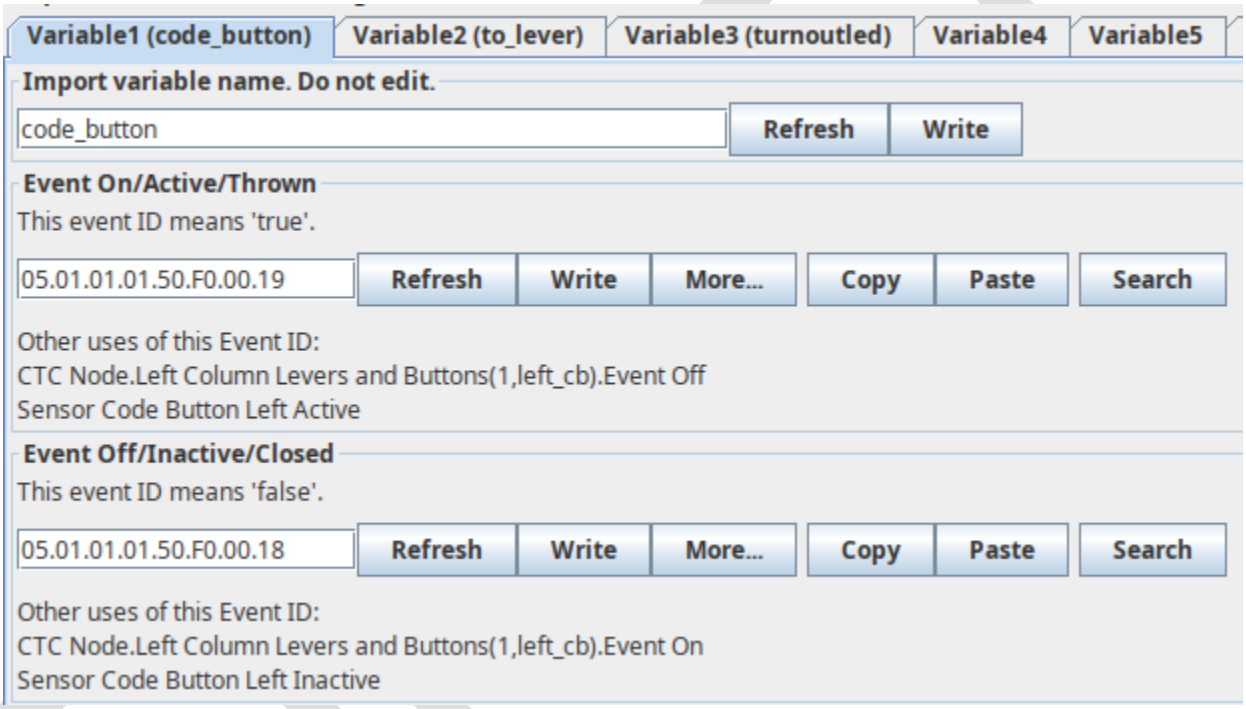
After:



After we have copied and pasted the event IDs into the correct variable's space, we can test.

We recommend that the events from the actual input or output be copied onto the variable. The reason for that, is if you find out you have the active (On) and inactive (Off) events reversed, you can quickly swap them by swapping the last digit (or possible the last 2 digits) of the events IDs already in place.

Here is a screenshot from JMRI showing this exact scenario.



Cue Hints

As the events are naturally paired, using the same two event IDs, just swapped, for the turnout switch and indicators, separately, will make writing the logic program text that much easier.

E.g. The event pair for the Switch Reverse and Switch Normal indicators will be

05.01.01.01.50.F0.00.14 On;05.01.01.01.50.F0.00.15 Off for Switch Reverse

and

05.01.01.01.50.F0.00.15 On;05.01.01.01.50.F0.00.14 Off for Switch Normal

Doing the same for the switch lever makes sense as well.

Support Information

LCCSignals provides support through email to assist you in resolving hardware and related configuration issues. This level of support does NOT include creating, troubleshooting or other work associated with logic integration.

Logic examples published on the lccsignals website or referenced in this manual are just that, examples. As every layout is a unique creation, these examples may or may not cover all attributes or situations that arise on your layout.

We can provide a paid engagement for integration and logic troubleshooting, but unfortunately, due to time constraints, we can not offer this service for free.

Contact Information

Please use our website at www.lccsignals.com to review the FAQ and other support information.

For email support, please email rick@lccsignals.com. We aim to respond to email requests for support within 48 hours to your message. Responses may be delayed around United States holidays, the NMRA National or Regional Conventions, RPM Meets and the Amherst (Springfield MA) annual train show.

Factory Reset

The node can be reset to factory defaults by the user. **This will totally, completely and irretrievably erase any user configurations on the node.** *This should be considered the "nuclear option" and should not be used as part of a normal workflow.* If you want to "restore" back to a previous point, we recommend utilizing the backup and restore functionality in JMRI's Panel Pro.

Only do this if you have a totally non-functional node due to an inadvertent corruption (attempting to write an event ID with non-acceptable characters seems to be the number 1 cause of corruption such to cause such a failure) or if support recommends this action.

Procedure:
1. Power Off Node.
2. Disconnect from LCC network.
3. Push and hold BOTH center buttons found on the back of the node. These buttons are labeled Blue and Gold.
4. Keeping both buttons pushed, plug node into power. Hold buttons for approximately 10 seconds, and then release.
5. The node will work to reformat storage. This **WILL** take up to 10 minutes. During this time, the node will **NOT** be reachable over the LCC network.
6. Once complete, the blue and gold LEDs on the back will light, and the node will be reachable over the LCC network.

Warranty Statement

LCCSignals provides a warranty on the hardware of the node against failure not caused by misuse, abuse, deliberate destruction or other non-acceptable use for a period of up to 1 year from the date of purchase.

This product is for hobby use only. This product is not intended to be used for 1:1 railroads. If you have a hobby 1:1 railroad, please contact us.