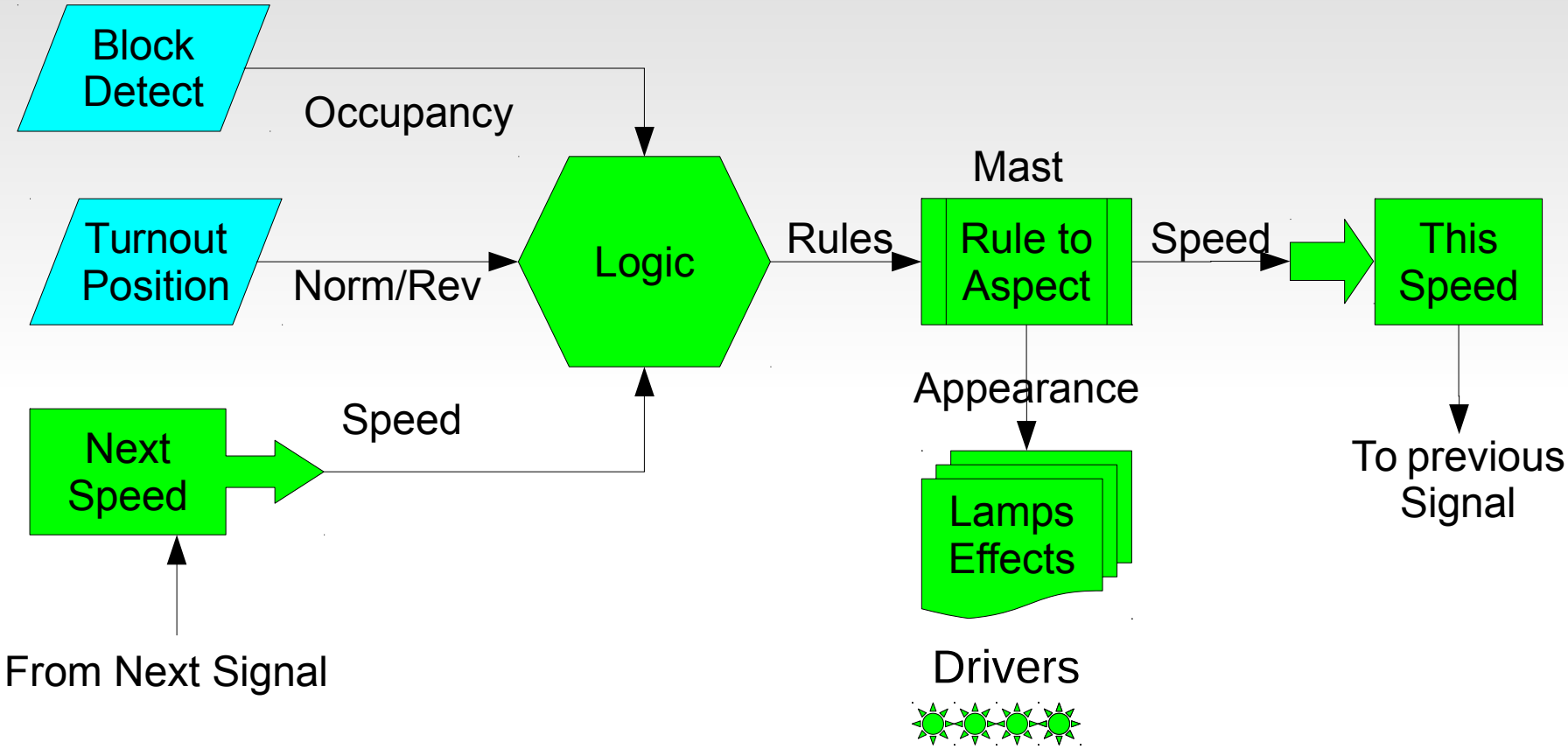
CPL Example

Signal Logic Example



With the Signal LCC all of the control functions required for signaling exist in a single node. Light blue items are taken care of with a daughter card.

If you want to off load (or monitor) any function with a computer you may do so by intercepting the LCC EventIDs that link sections with each other.

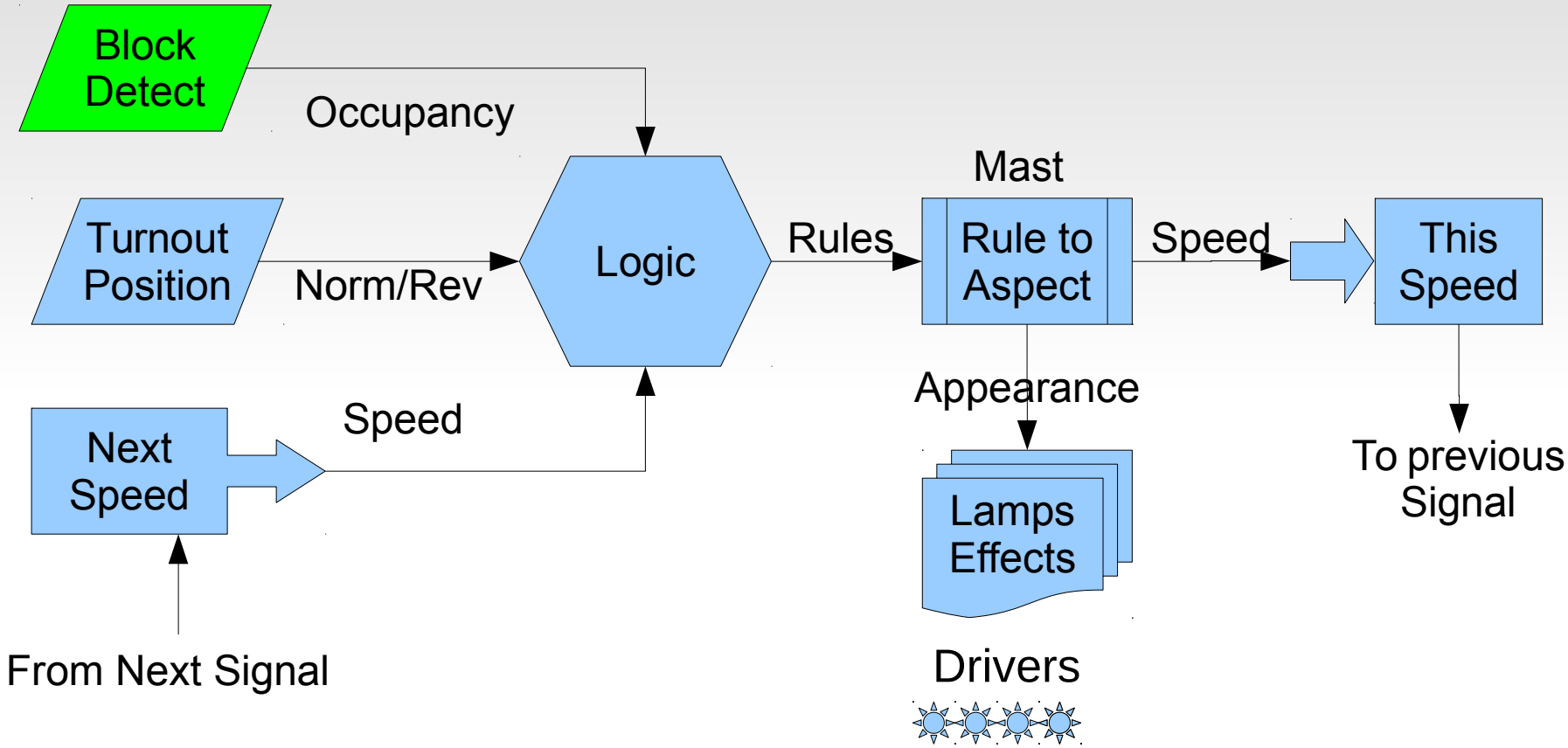
Signal Logic

- Signal Logic

In order to build a signal controller that watches all related status Events from the railroad and CTC panel, and makes independent decisions about the proper signal states and appearances, it must contain internal logic. This logic must either be user controlled or else it must understand all known signaling rules.

Triggering the evaluation of a conditional is done when any monitored event is seen. There are two trigger options. In the first option evaluation of a conditional is only done if the monitored event actually changes the state of the variable. In the second case the evaluation is done when ever the event is seen, even if there is no resulting change to a variable. This allows repeated single events to trigger a conditional multiple times.

Block Detect Example

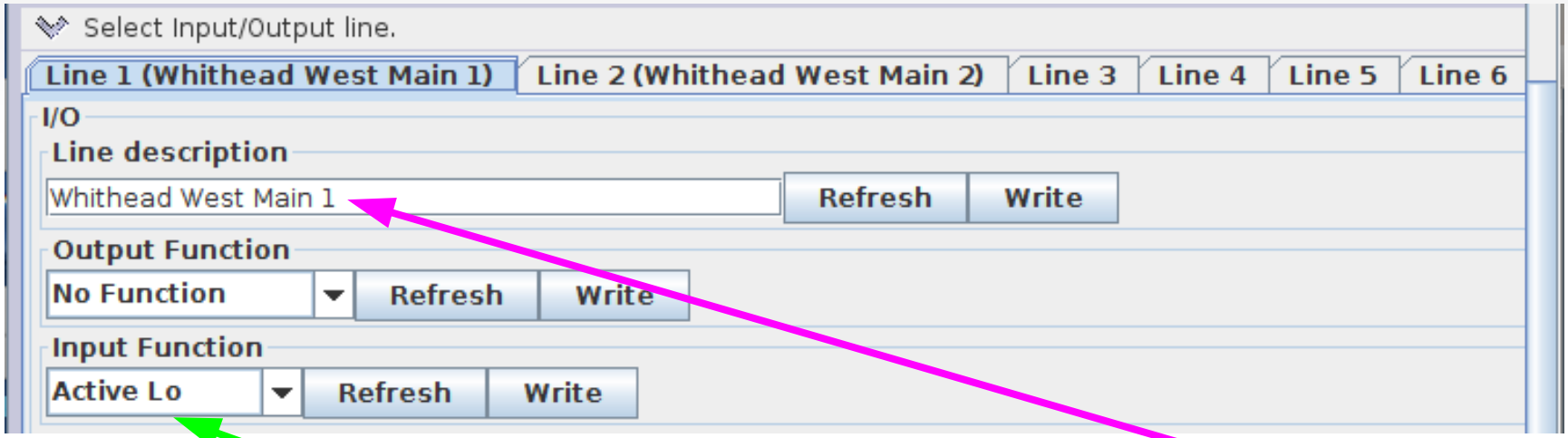


The BOD4 and BOD4-CP cards each include 4 block detector circuits for easy connection to the Signal LCC board. These boards use CT coils to prevent track voltage drop and provide 100% isolation.

Block Detector Variables

- Variables

Variables are used to follow the state of objects of interest such as block detectors, turnout positions, etc. Normally two events will allow the variable to follow the state of some object, true/false, normal/diverging, clear/occupied, etc.

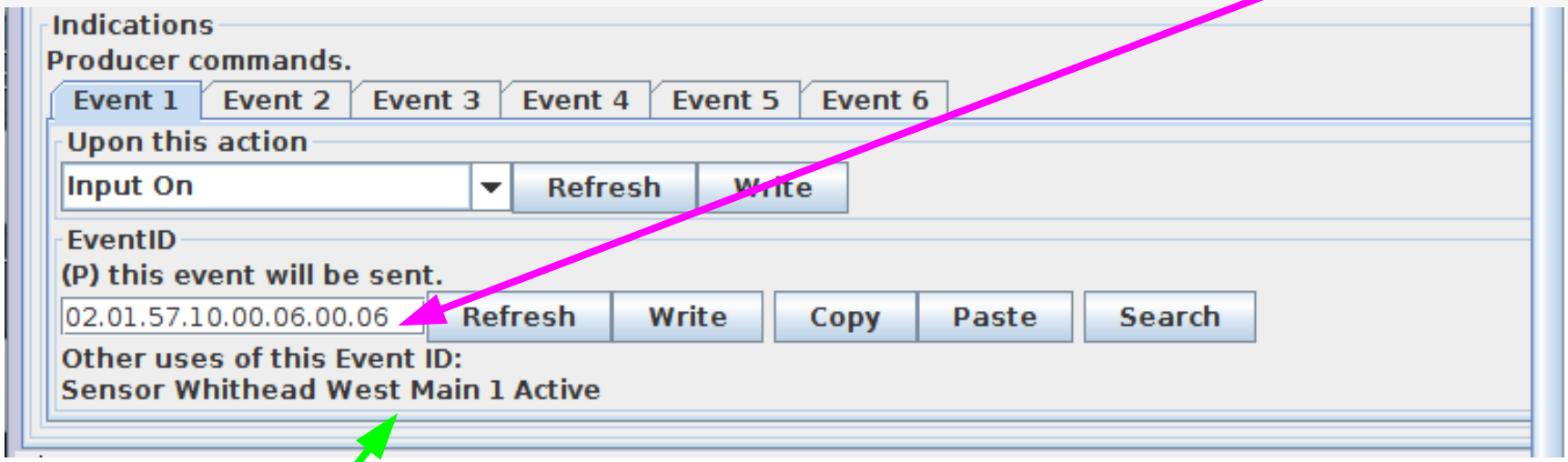


Lets start by connecting a block sensor to an input. Its description is 'Whithead West Main 1' so we enter it in the description block and 'Write' it to the node. Detectors are Input Functions with 'Active Lo' so we set that and write it. For a normal Input be sure that the Output Function is set to 'No Function'. Of course 'Line 1' is the one connected to our first block detector.

Block Detector Variables

- Input (Producer) Events

We now go to the Indications (Producers) for this line, and enable two events. The first (Event 1) will be sent when the Input is 'On' (goes low in our application) When we need to know if the block goes occupied, we will use this EventID.

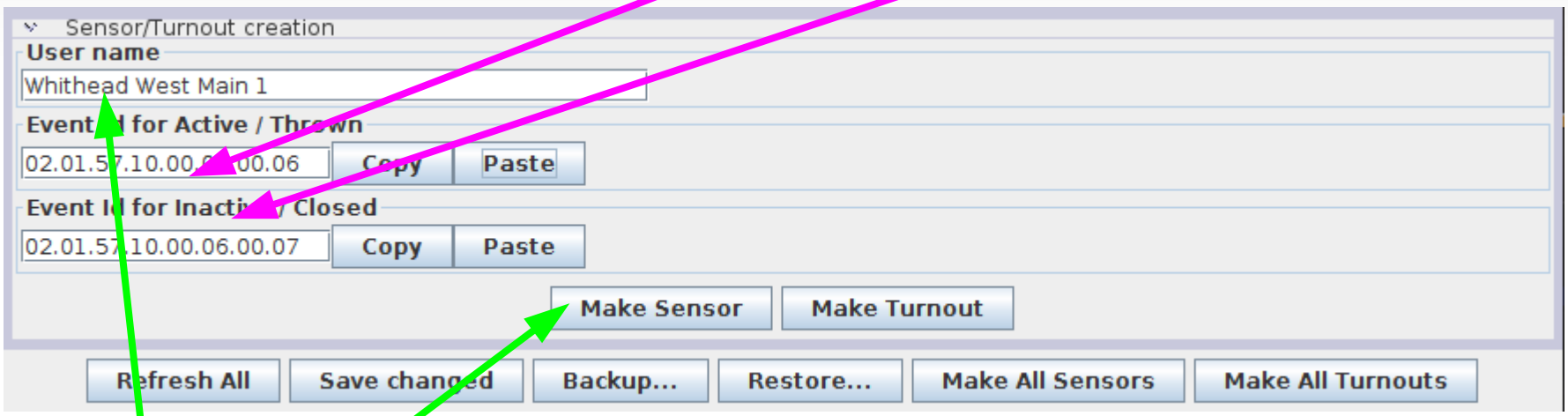


Event 2 will be set to 'Input Off'. Use 'Copy' and 'Paste' when you need to utilize the magic numbers (EventID) for these events. Its description 'Whithead West Main 1' Is noted here to remind you of its function. This information is known because I made a JMRI sensor that follows it. This is a JMRI feature available in the JMRI CDI tool.

Block Detector Variables

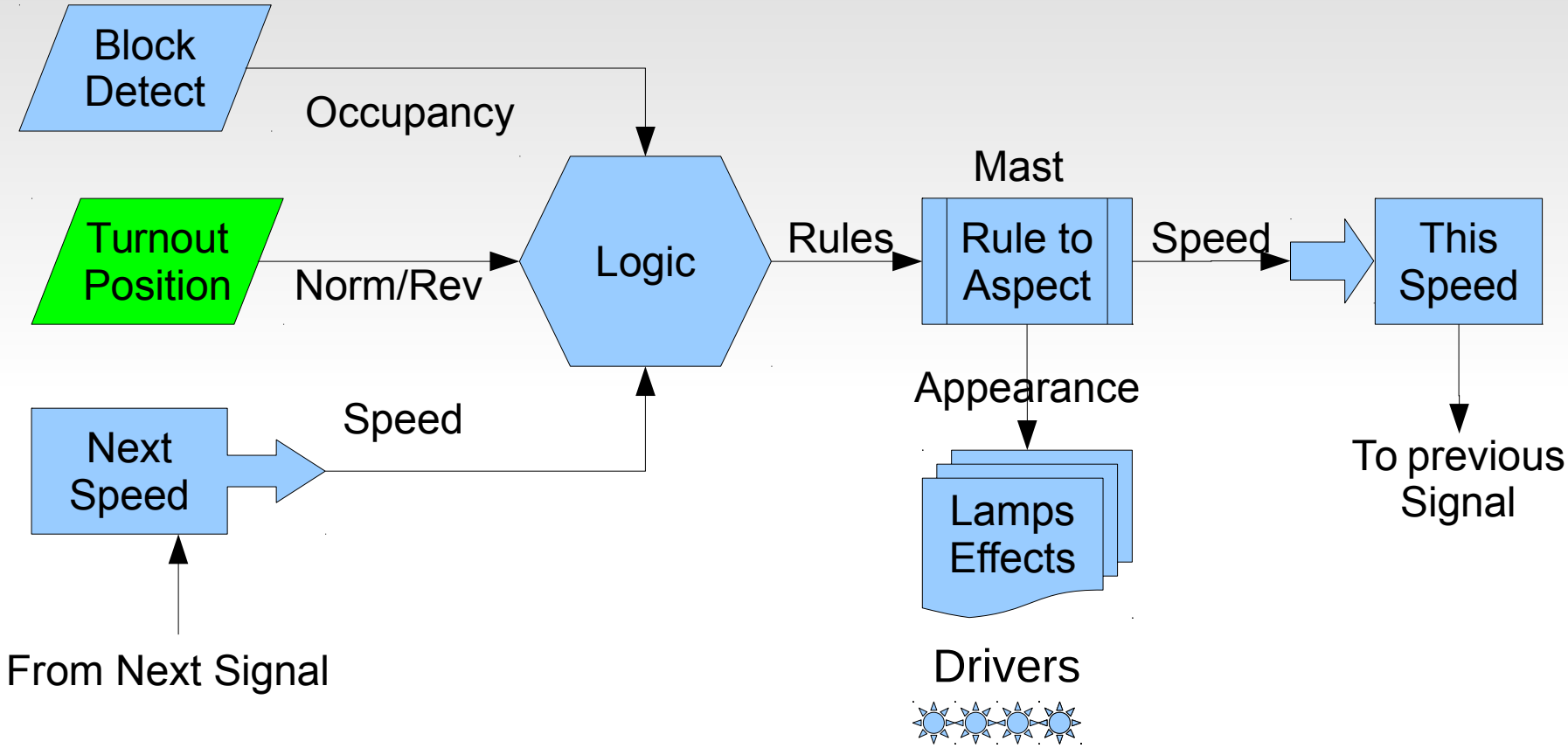
- JMRI Sensors

JMRI includes a handy tool at the bottom of the CDI window to make sensors or turnouts from events. LCC Nodes may use two (or more) EventIDs to control sensors and turnouts, so you must use cut/paste to choose the pair that you want for JMRI. For this sensor we use Event 1 and Event 2 that we just defined.



Enter the JMRI user name for this sensor, (or turnout) then click on the Make Sensor button. This item will automatically be added to your JMRI Sensor (or turnout) table. Be sure to save the table for future use. Normally this data will become part of a ‘Panels’ file, and be synchronized with the node when loaded.

Turnout Control Example

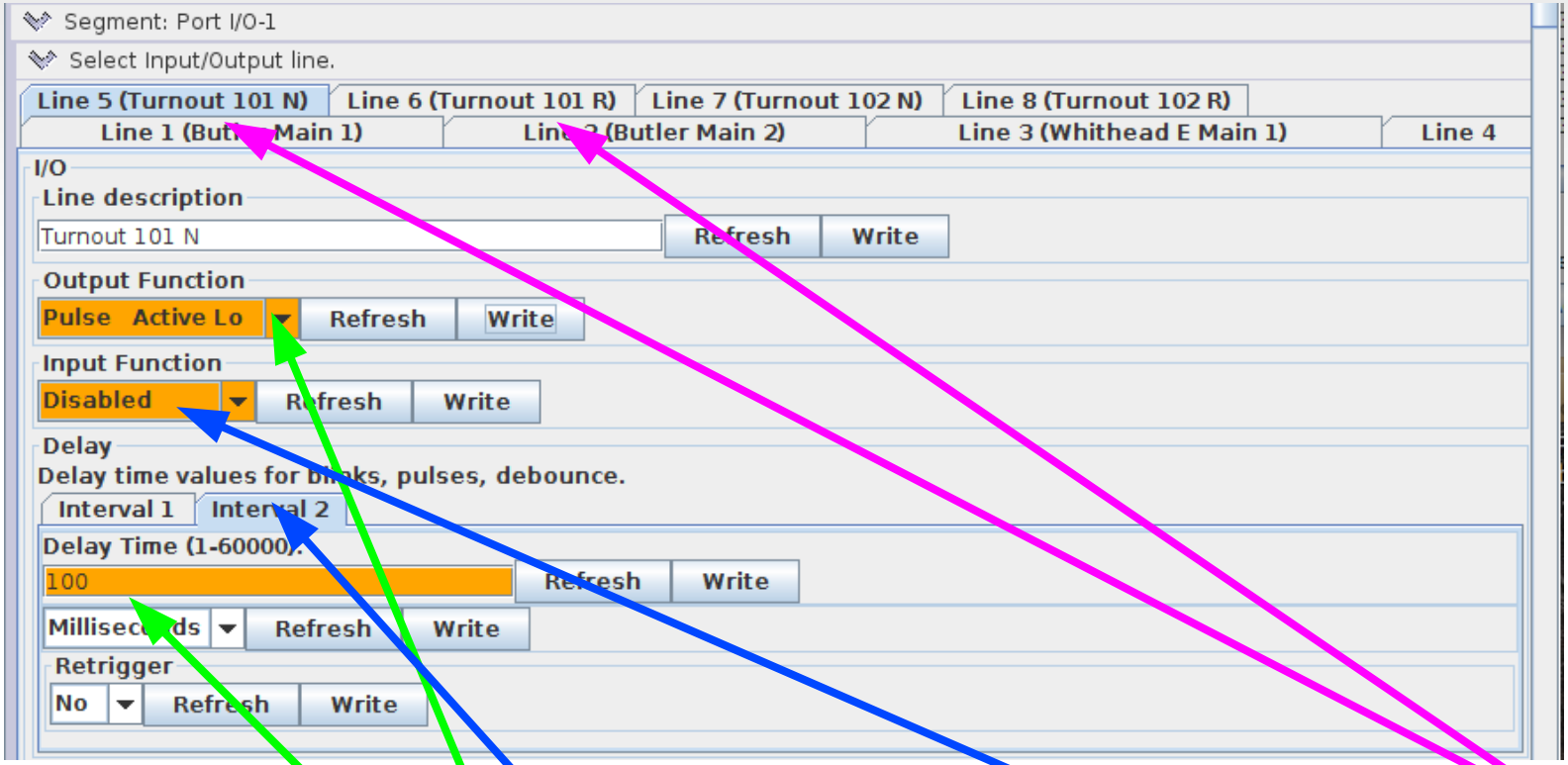


The BOD4-CP cards also include 2 ‘H’ Bridge drivers controlled by the Signal LCC board. These drivers are isolated from the LCC to prevent power supply issues.

Turnout Variables

- Output (Consumer) Events

We now go to the Indications (Producers) for a line. (on board ...07)

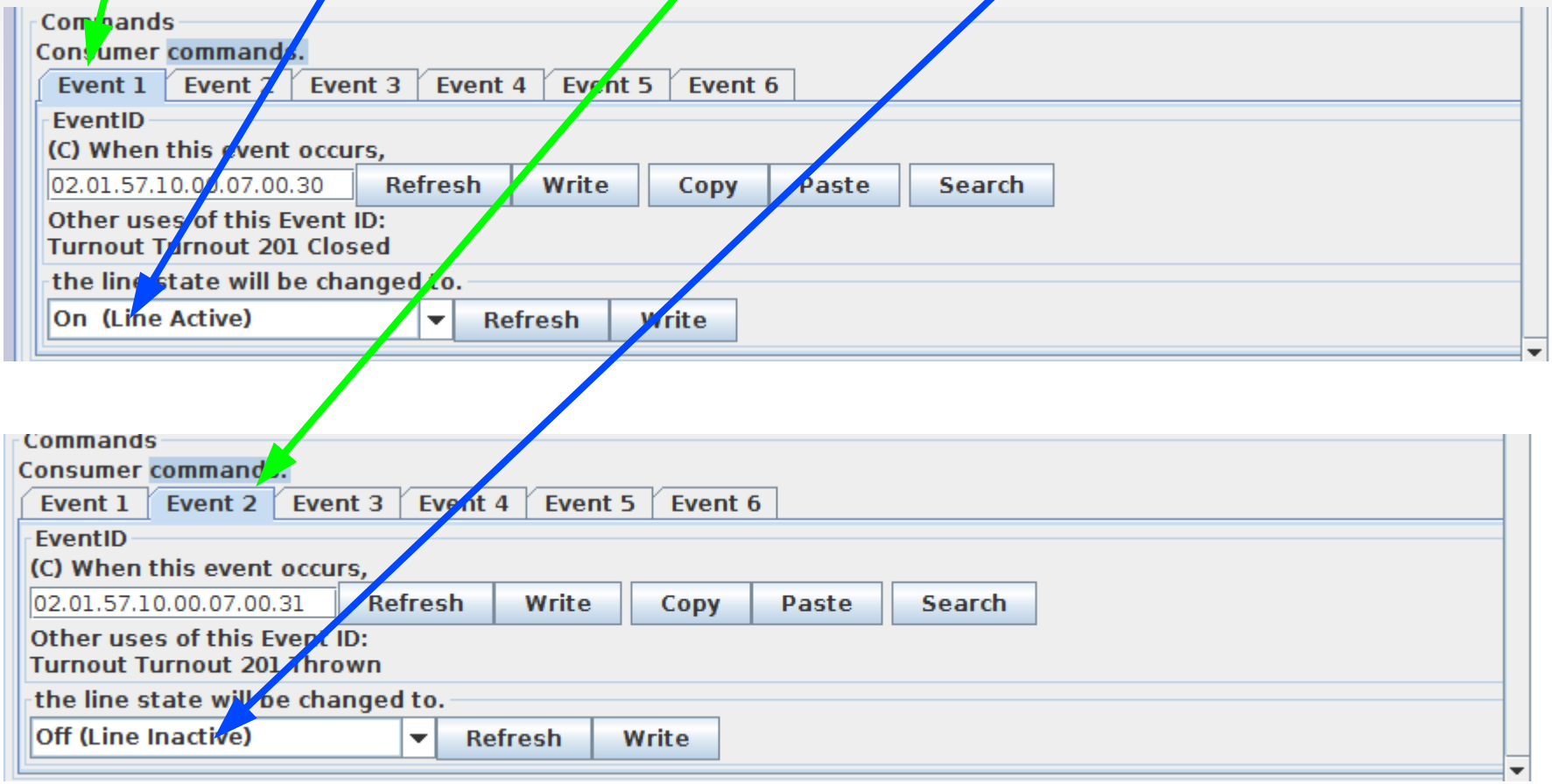


Our turnouts are controlled by Kato dual coil solenoids. This requires dual line drivers and 100mS pulse outputs. Normally inputs are disabled for Output Functions. Note: Use Interval 2 for pulse length. Interval 1 is the pulse delay.

Turnout Variables

- Output (Consumer) Events

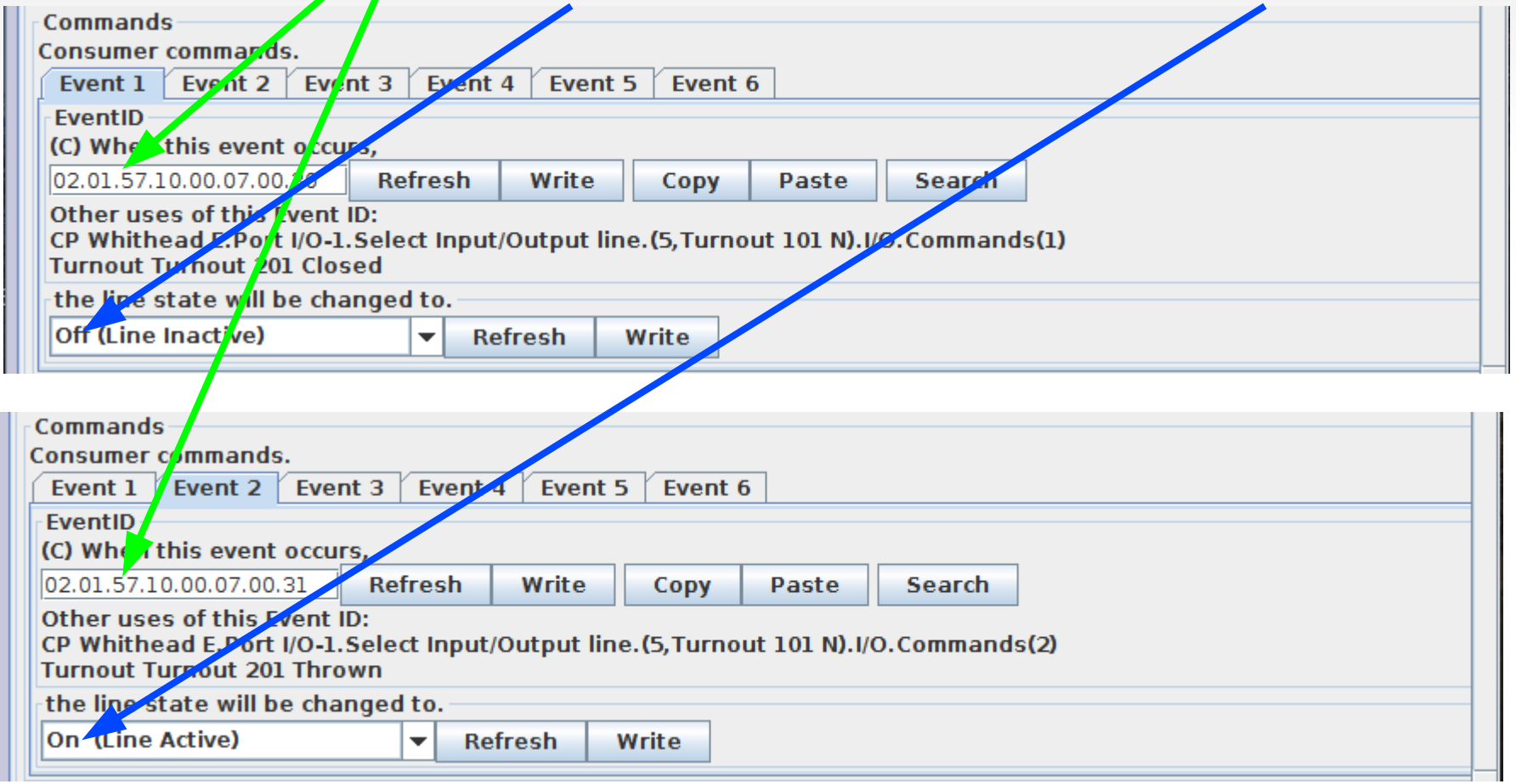
Event 1 will turn ‘On’ the line and event 2 will turn it ‘Off’. Remember we already specified that ‘On’ just sends a 100mS pulse, so our coils are safe.



Turnout Variables

- Output (Consumer) Events

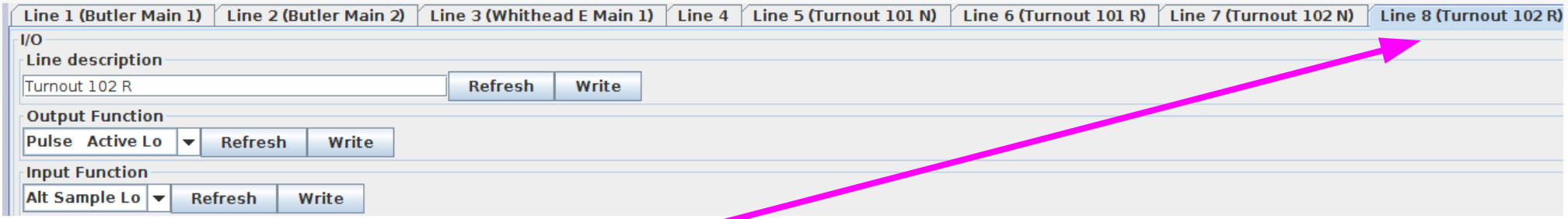
To configure the second coil we will do two tricks with events. First we copy and paste the two events from the first line to the second line. Next we reverse their actions. Event 1 will turn ‘Off’ the line and event 2 will turn it ‘On’. Done!



Turnout Control

- Input (Producer) Events.

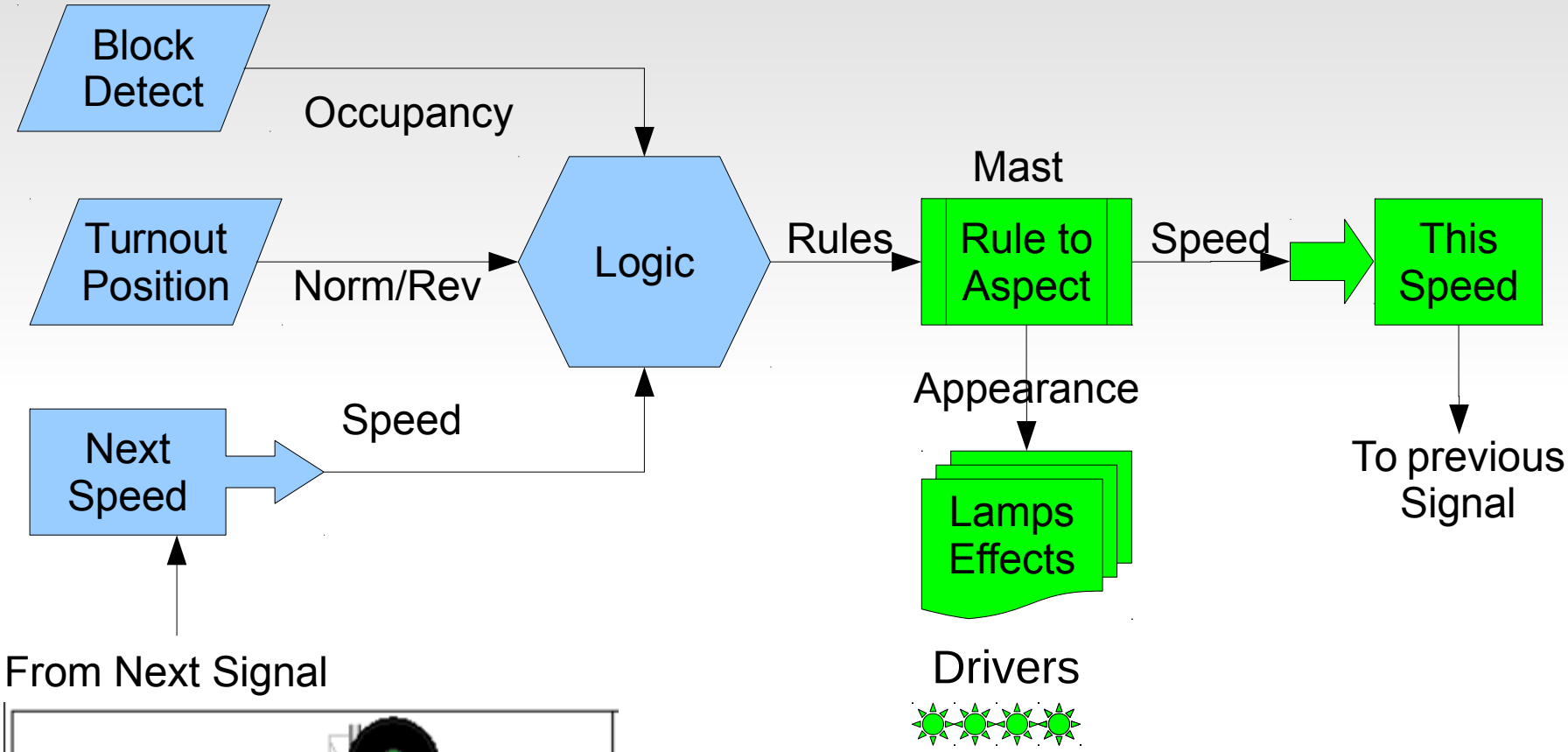
We now get really fancy. To be compatible with the Berrett Hill Touch Triggers we added a 'Sample' option to our I/O lines. We take advantage of that on the BOD4-CP. Each output driver has a corresponding input line.

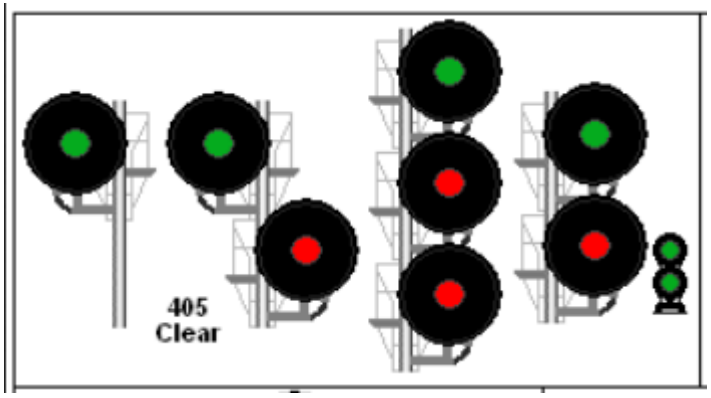


We use the input for Line 8 and connect it to a push button. We set the 'Input Function' to be 'Alt Sample Lo'. This means that each time the input goes low it will alternate the function state. The line still sends its output to drive the turnout as before. We can also use the same line (physical wire) as an input by sampling it.

- For simplicity just have the line send the turnout control events directly. For realism, combine the control events with occupancy and/or panel information that prevents any turnout movement when occupied, or locked.

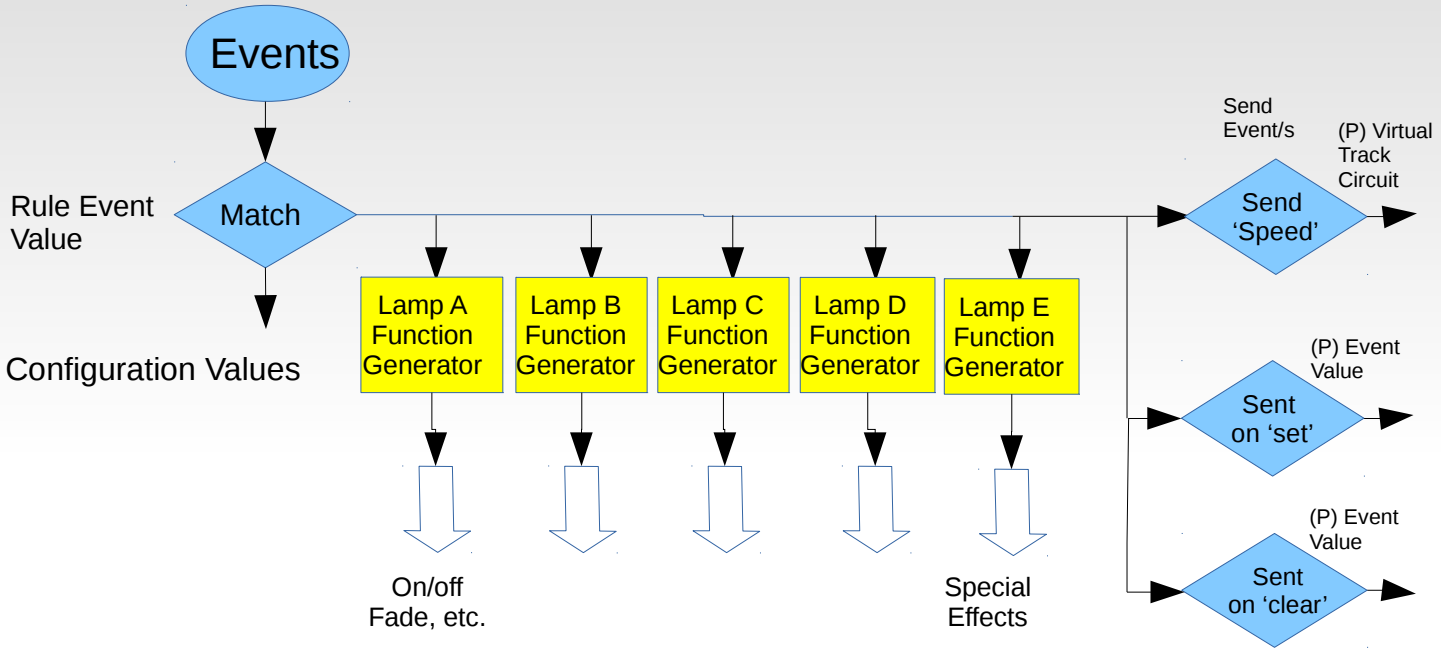
Rule to Aspect





A quick look at any railroad rule book will reveal that the same rule may be displayed in many ways. This means that we need a 'Rule' to 'Aspect' conversion process.

Rule to Aspect



- Signal Masts

This flow chart shows Mast functionality. Any signal rule that is seen (matched) can send up to 4 lamp control messages, an optional special effect, what speed is sent by the track circuit, and send optional events. These optional events may be used to send indications to a CTC panel.

Signal Masts

- The EventIDs sent and responded to by each rule are also controlled by the MASTS segment. Because this Rule to Aspect conversion is actually the links between the 'Rule' events and the actual hardware we call it all 'MASTS' and treat it as one segment in the CDI.
- The Signal LCC supports 8 Masts, each of which supports 8 indications. If a single mast requires more than 8 aspects, then any mast may be logically linked with a previous one.
- A 'Mast' definition makes two assumptions.

    1) Only one aspect may be shown at a time. Setting any aspect automatically cancels any previous aspect.

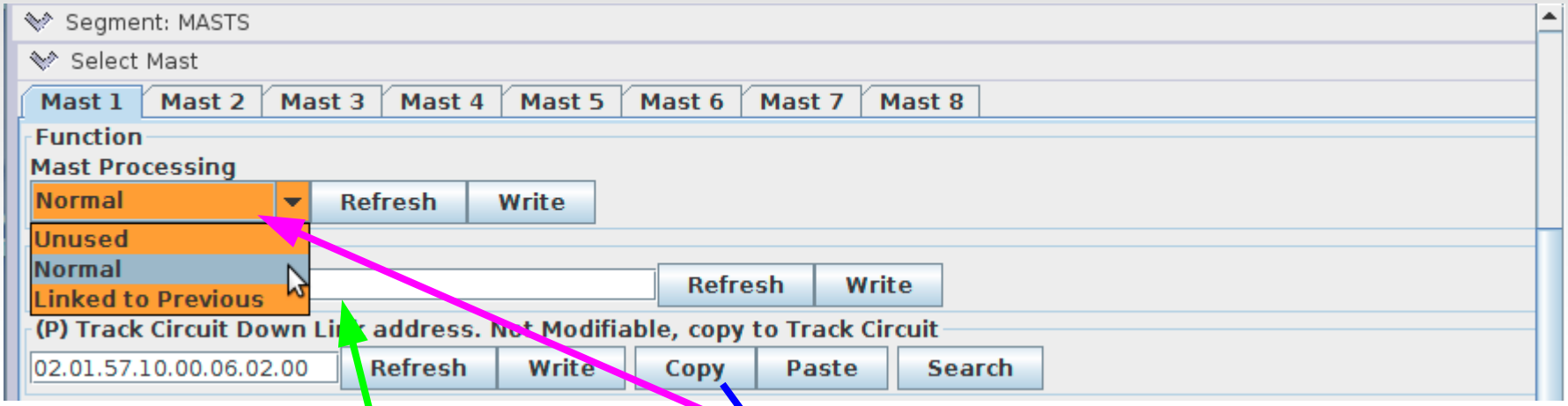
    2) A mast may only have a single speed limit at a time. This 'Speed' is the present allowed speed going past the mast.

- Making a mast 'Linked to Previous' carries the above assumptions over from any previous mast/masts. Speed is always taken from the first mast.

Signal Masts

- LED Drivers
- Different colors of LEDs have different voltage drops. This drop is subtracted from the drive voltage when calculating the series resistance. A typical red LED operates at 1.9V and a green operates at 3.3V. This means that at 5V the red resistor drops 3.1V and the green resistor drops 1.7V. With the same resistor values, the red LED will draw nearly twice the current as the green. Using a 12V source, the resistor voltage drops are 10.1V and 8.7V respectively, or just a 15% difference in current.
- Sometimes it is easiest to wire 2 LEDs in series for Position Light or Color Position Light signals. The voltage drops of green and yellow LEDs make it difficult or impossible to drive these with 5V supplies. As a result all of our RR-CirKits signal driver boards have always supplied 10V or more to the drive circuits.
- Brightness settings help you match the intensity of LEDs in the same mast.

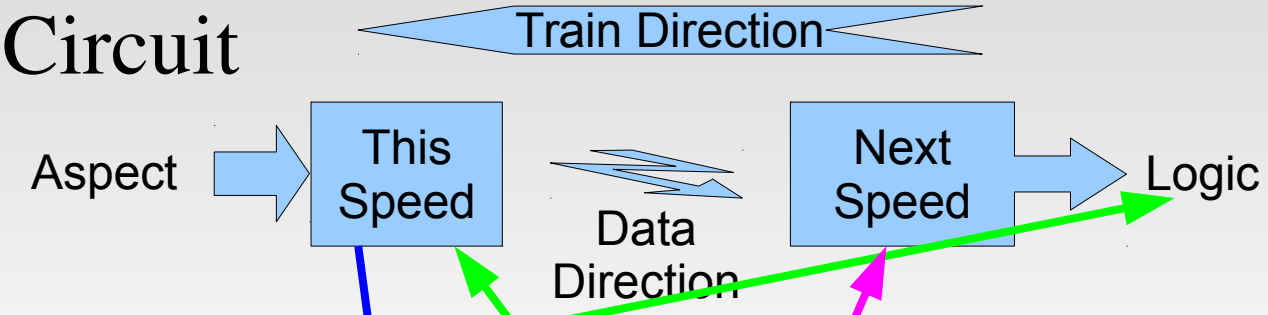
Signal Mast Setup



- Function
- To use a mast you must first change it to 'Normal' or 'Linked to Previous'.
- Next give it a Mast ID so you can easily find it again later. This could be a CTC panel number, a mile marker, a control point name, etc.
- Track Circuit Down Link Address. This fixed EventID is used as a pointer to the current track speed setting for this mast. Copy this number to any track circuit receive (RX) table to make it easy for logic to follow speed.

Signal Masts

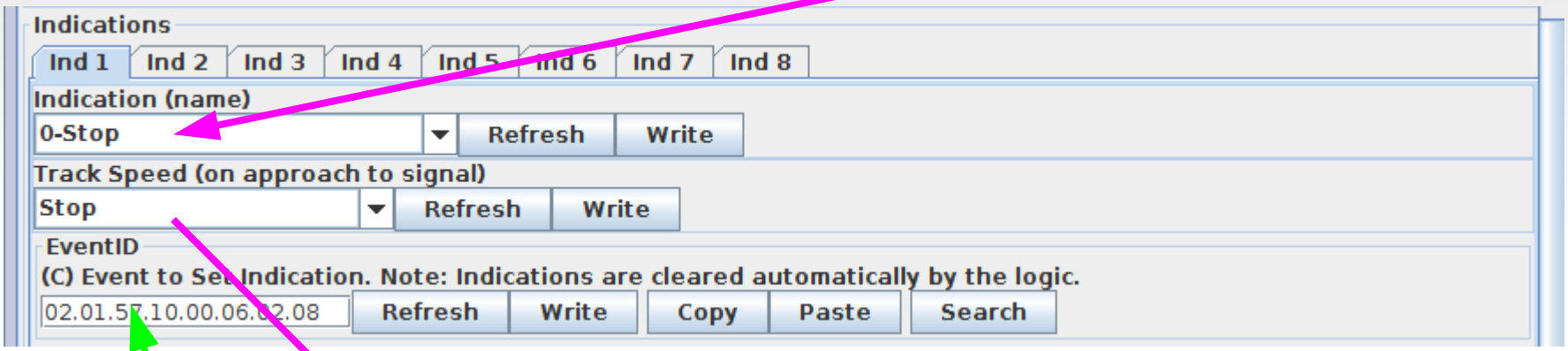
- Track Circuit



- When calculating signal rules, the most important information from the next signal is the required track speed on approach to that signal. In many cases this information is actually a part of the rule name.
- In modular layout setup, getting this information easily from module to module is the single biggest roadblock to installing authentic signaling. Our Virtual Track Circuit concept is designed to simplify this.
- To link the speeds selected on a mast to the logic of another mast, simply copy the 'Track Circuit Down Link address' from one mast, and paste it into the 'Remote Mast Up Link address' of another. This automatically makes the speed information from one mast available to the logic of another mast without requiring the entry of specific EventID information for each speed change into the appropriate logic conditionals.

Indications (Name)

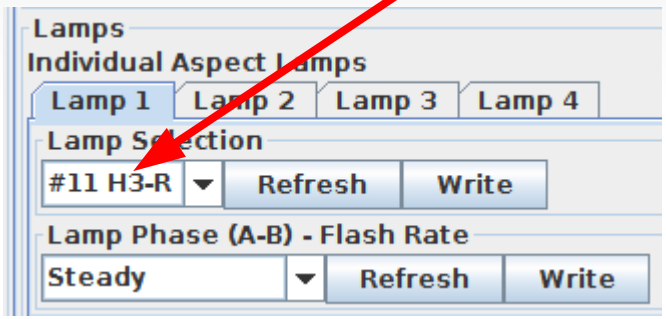
- Indications tell the crew what to do at a signal. The 'Rule' or 'Name' is the shorthand for the Indication.

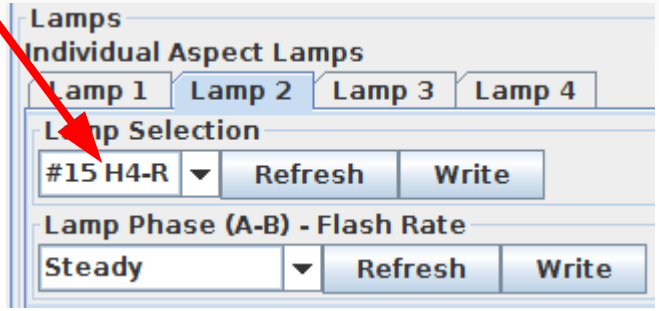


- The selected 'Track Speed' (one of eight possible) is the value that will be sent back to the previous signal over the Virtual Track Circuit. If the names don't match your rules, simply pick something similar. Its just a number to the track circuit.
- EventID to Set Indication. This is the EventID used by the signal logic to set this signal rule.

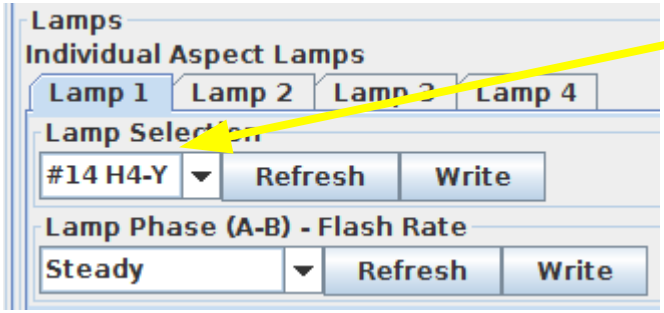
Lamps

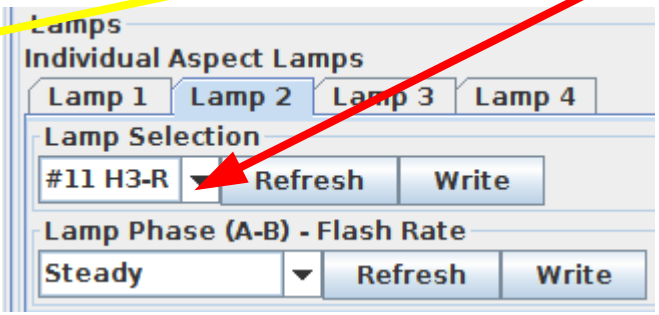
- The bottom line in displaying an aspect is to choose what lamps are lit. After all, that is what the crew (and the visitors) actually see.
- The mast on the signal bridge is dual head searchlights. This means 'Stop' will display Red over Red. Pick the appropriate lamps to show this.





- To show Indication 2 - 'Approach' display Yellow over Red.





- Continue in like manner until you have entered each possible aspect.

Lamps

- Each Indication (Aspect) can display with as many as four lamps. I you have a rare signal aspect that can not be shown with just 4 lighted lamps you can make a duplicate mast to light any additional lamps. Remember dual lamps that light together only count as a single lamp. (e.g. in Position Lights and Color Position lights) Only lighted lamps count. Only controlled lamps count. A marker that is always lighted can simply be powered full time.
- Lamp Phase – Flash Rate may be used to flash signals automatically. One common example is 'Advance Approach' which is commonly displayed with a flashing yellow lamp. Setting an appropriate Flash Rate means that the signal logic doesn't need to worry about flashing the signal or overloading the bus with unecessary traffic. Providing both A and B phase options is handy for grade crossing flashers or other alternating lamp situations.

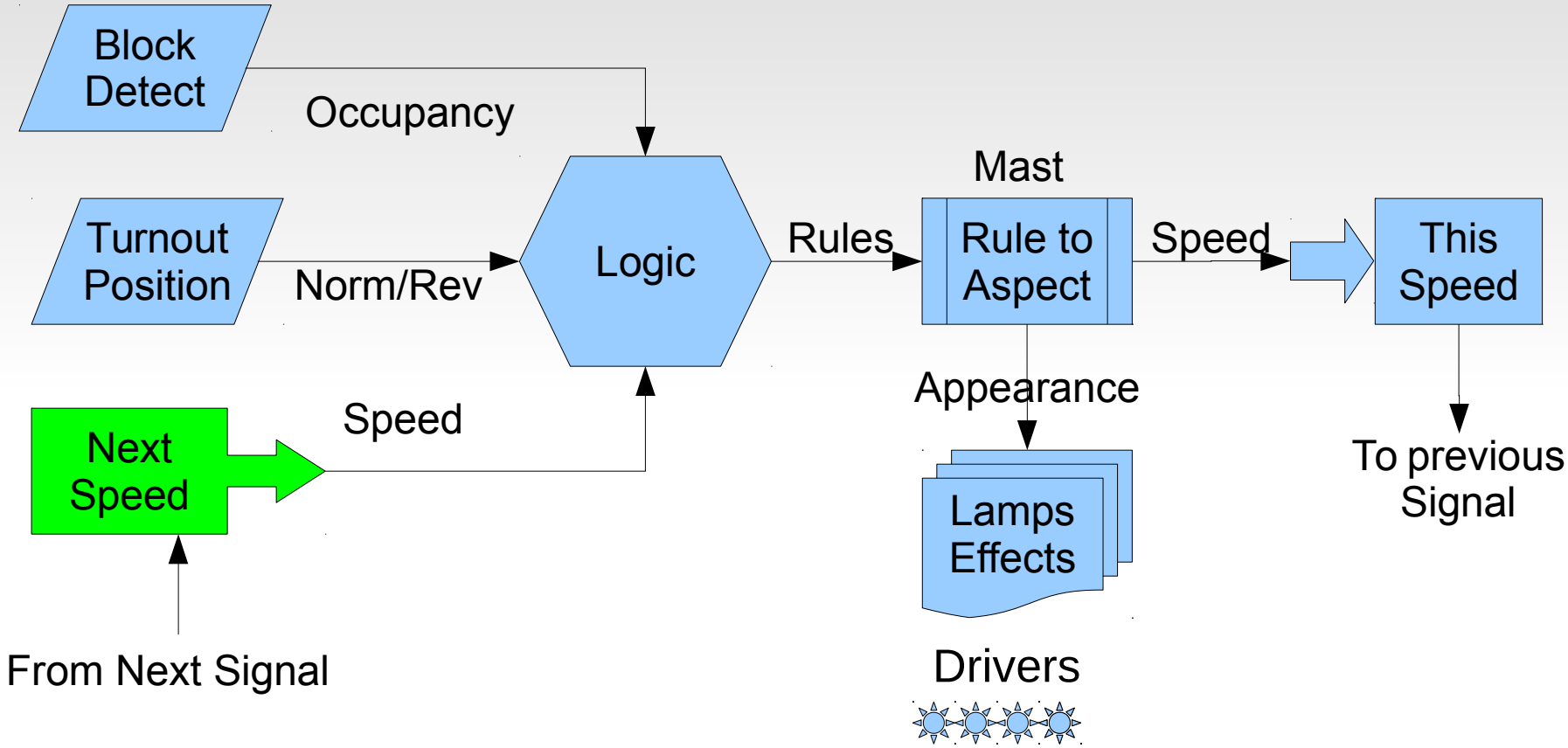
Lamps – Special Effects

- Incandecent fade. Signal lamps are wired differently than standard household lamps. They include a ballast resistor in series with the lamp. This ballast serves two purposes. One is simply to set the brightness of the lamp. More importantly when the cold lamp is first powered up it prevents the normal inrush current by dropping most of the voltage across the ballast until the lamp warms up. The visual result of this is that a signal does not blink on rapidly. In fact signals fade on slowly enough to be noted. Of course even houshold incandecents fade off slowly as the lamps cool down again.
- Transition effects. The B&O signal clip we saw earlier shows an interesting transition between Clear and Stop. Not only does it show the fade up and down, but it interjects a brief 'Approach' into the change. I can not tell you why this is done, but selecting 'Transition Down' as a special effect on 'Stop' will allow you to do this. (and wow your rivet counting crew)

Lamps – Special Effects

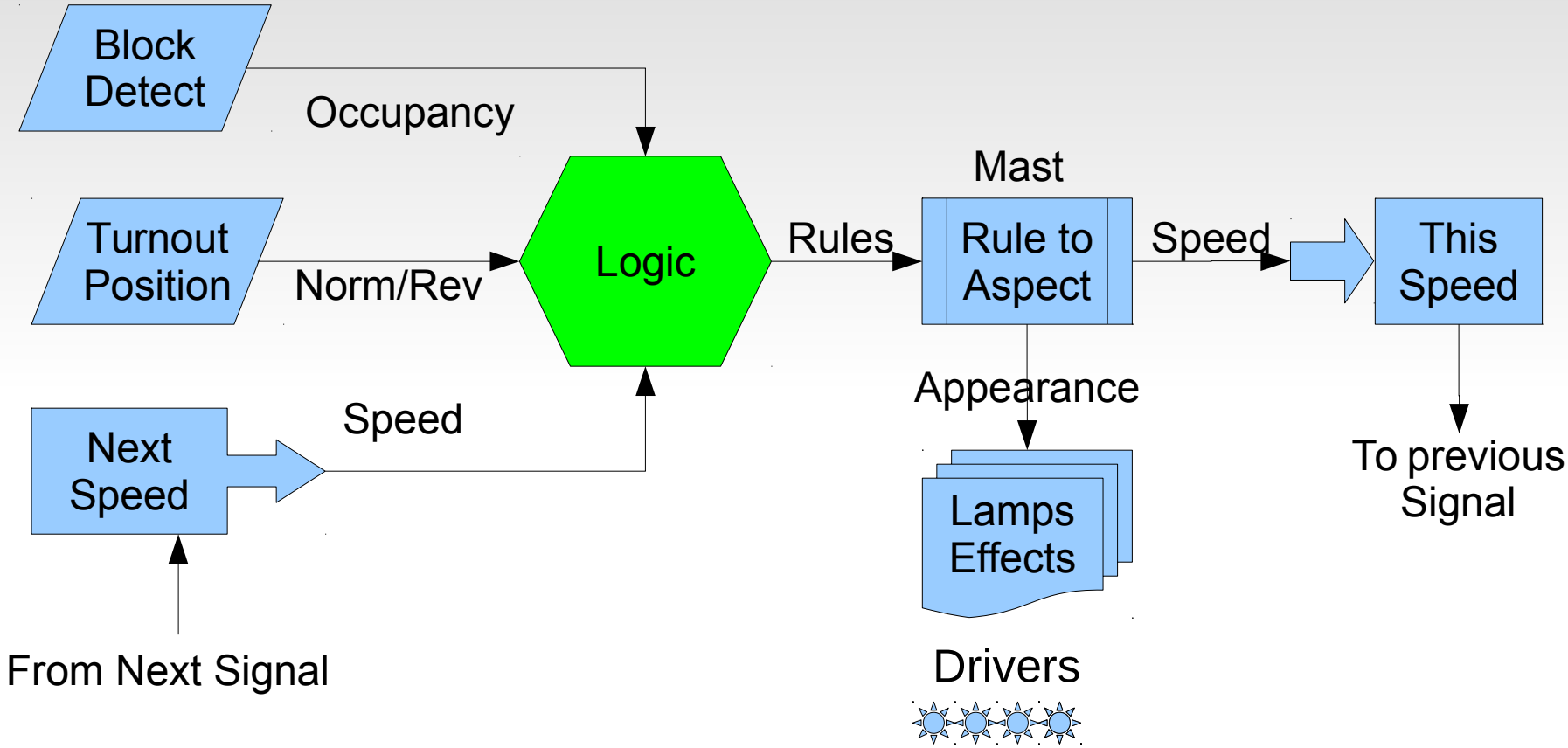
- H2 Red Flicker. Many of you know that real searchlight signals (not just the H2) have an internal arm that swings back and forth in front of the lamp. It hangs by gravity with a red roundel in center position. Either green or yellow requires swinging the arm out of its center position with electromagnets. Not quite as obvious is the fact that you can not change between yellow and green without passing the red between them. This causes the red flash. The other part of the effect is that the arm is free swinging and during a change it will overshoot its position as it settles down. This swings the color roundels past their normal positions which causes the signal to appear to flicker.
- Strobe lights can be found around the layout. Sometimes it is nice to be able to utilize unused signal outputs for other purposes.

Track Circuits



Paste the next masts ‘Up Link EventID’ from any mast to a track circuit. This creates a virtual link directly into the logic variables by virtual name rather than by using actual event numbers. The logic for a mast can be setup, or mass produced, without knowing any actual mast IDs ahead of time.

Signal Logic



We have covered all the edges. Now we can talk about the central subject, Signal Logic itself.

Signal Logic

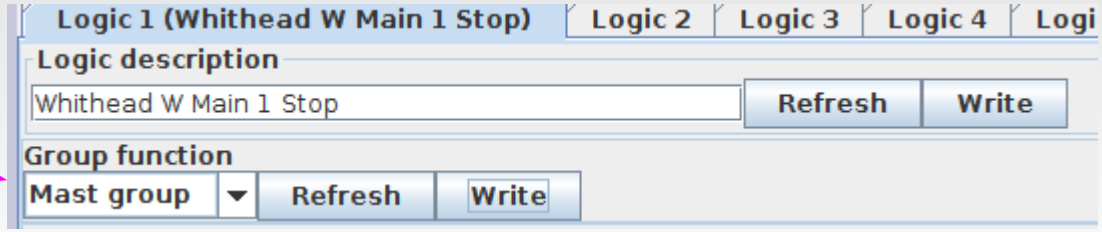
- Signal Logic is just a series of conditions (called conditionals) that are checked to see what signal rule should currently be in effect.
- Logic conditionals should be easy to cascade with the calculations for the most restrictive rules having priority over less restrictive rules. We do this by checking each conditionals in order from top down. Any rule that is found to be true first checks for any more restrictive rule still in effect. (which exits processing if found) Then it sends its appropriate events, and finally skips over any less restrictive rules for the mast.
- Any conditional may directly send up to 4 events representing signal rules (or anything) when it is found to be true. (or false) A cascade option allows even more events to be sent in special situations. Note: this logic may be used for many other purposes than just calculating signal aspects.

Signal Logic

- Logic Functions consist of the usual AND, OR, XOR operators. In addition there are two ’change’ operators. These change the true/false sense of a conditional based on the AND and OR of the variables.
- Additionally we have added a non-standard logic operator called ’AND Then’. This makes it very easy to keep track of train direction. You can simply watch two block detectors and determine train direction by the order in which they are activated.
- A recent addition is the ability to control the action associated with both true and false evaluations of a conditional. These options are to ’Send then Exit Group’, ‘Send then Evaluate Next’, ’Send then Send Next’, ‘Exit Group’, and ’Evaluate Next’. The ‘Send then Send Next’ automatically goes to the next conditional and always treats it as if it were true. This makes it easy to send more than 4 events from a single conditional.

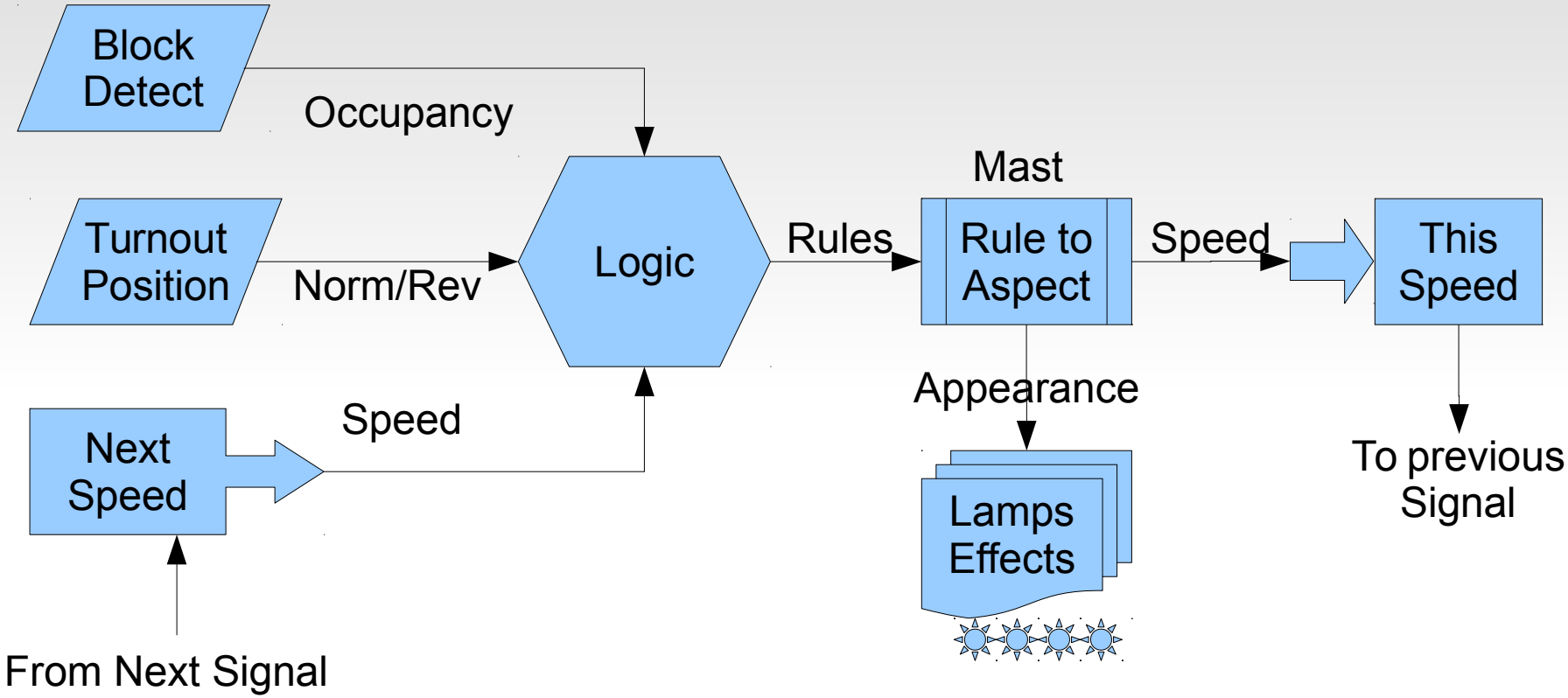
Logic Conditionals

- Normally Signal Logic Conditionals will have a Group function of 'Mast Group' or else 'Last'.



- The function of a conditional 'Group' is to pick the most restrictive rule for a mast and send it to the mast table for conversion to the proper aspect.

Signal Logic Example



The basic signal logic overview.

- Rule logic is calculated using layout status information and next signal speed.
- The resulting ‘Rules’ are converted to lighted lamps, effects, and speeds.

Trailing Signal Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	**Not** CTC-Right	OR	OS BOD	Stop	Stop
Siding selected / Main occupied	Turnout Reverse	OR	Main BOD	Stop	Stop
Next Main Stop	Main Mast Stop	null		Approach	Medium
Next Main Medium	Main Mast Medium	null		Approach Medium / Advance Approach	Clear
Next Main Clear	Main Mast Clear	null		Clear	Clear

- To create ’**Not** CTC-Right’ simply reverse the events controlling ’Variable 1’ for that conditional. This data is from the direction lever on a CTC panel.
- We check wrong CTC direction, the turnout against us, the OS occupied, and the track past the turnout occupied. Any of these will set the signal to Stop.
- If the signal has not been set to Stop, then we check to see if the next signal’s speed is ’Stop’. (Main Mast Stop) If so we set this signal to ’Approach’ with a speed of ’Medium’. (or ’Approach’) Sometimes it is helpful to realize that ’Approach’ when used by itself is short hand for ’Approach Stop’.
- If not Stop, then we check for next signal’s speed of ’Medium’ (or ’Approach’) and set our aspect appropriately.
- Finally, finding nothing more restrictive, we can set it to ’Clear’.

Facing Signal Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	Not CTC-Right	OR	OS BOD	Stop	Stop
Siding occupied	Turnout Reverse	AND	Siding BOD	Stop	Stop
Main occupied	Turnout Normal	AND	Main BOD	Stop	Stop
Next Siding Stop	Turnout Reverse	AND	Siding Mast Stop	Medium Approach	Medium
Next Main Stop	Turnout Normal	AND	Main Mast Stop	Approach	Medium
Next Siding Medium	Turnout Reverse	AND	Siding Mast Medium	Medium Approach Medium	Medium
Next Main Clear	Turnout Normal	AND	Main Mast Clear	Clear	Clear

- We now expand our logic to consider two possible routes.
- It should be clear from the above that calculating aspects for the signal prior to this interlocking is simplified by knowing the signal speeds, because there are five different aspects to check, but there are only three different speeds to check. The three different possible medium speed aspects do not cause any change in the signal prior to this one, so it only needs to show Clear, Approach Medium, (or Advance Approach) Approach, and Stop.

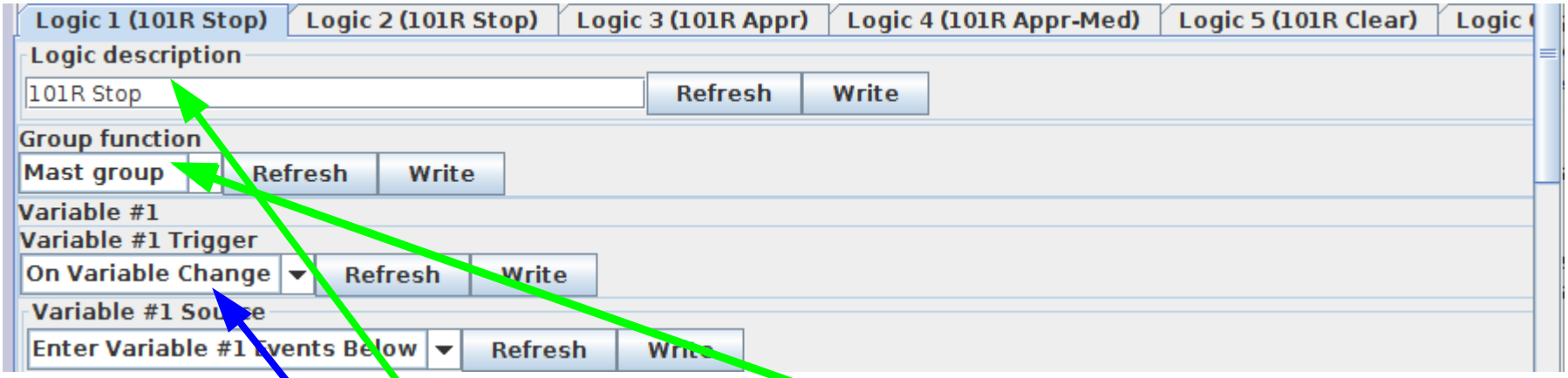
Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	**Not** CTC-Right	OR	OS BOD	Stop	Stop

System Name: MS02.01.57.10.00.06.01.01;02.01.57.10.00.06.01.00
User Name: CTC Lever Whithead RM1

System Name: MS02.01.57.10.00.06.00.1E;02.01.57.10.00.06.00.1F
User Name: Whithead OS Main 1

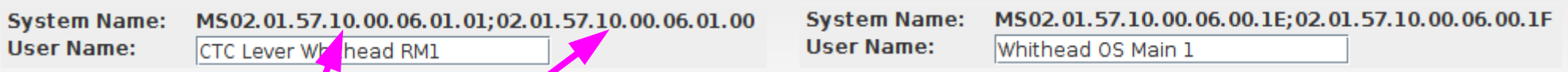
- These two variables as seen in JMRI. I used the Sensor/Turnout creation tool to enter them.



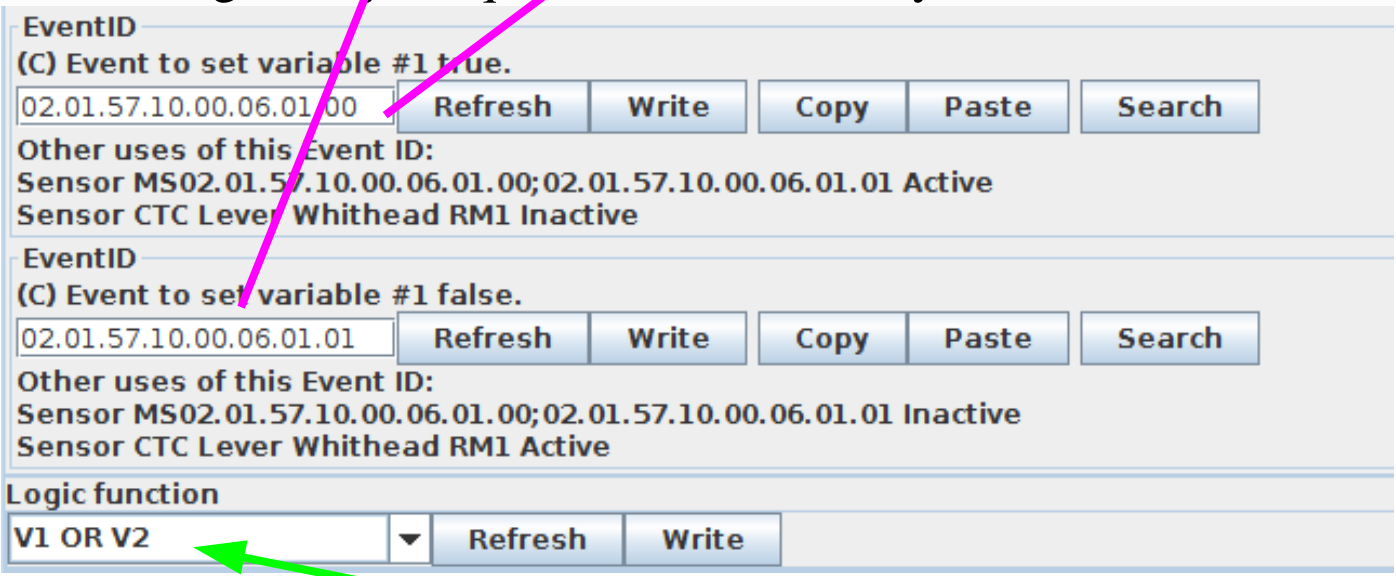
- Enter the logic description and set the function to Mast Group. Logic defaults to watching variable changes.

Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	**Not** CTC-Right	OR	OS BOD	Stop	Stop



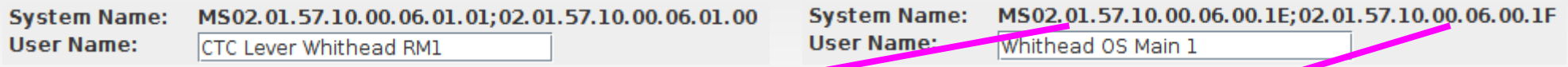
- I actually used the default EventIDs found in variable #1 to create my lever. EventIDs are globally unique so I had no worry about conflicts in meanings.



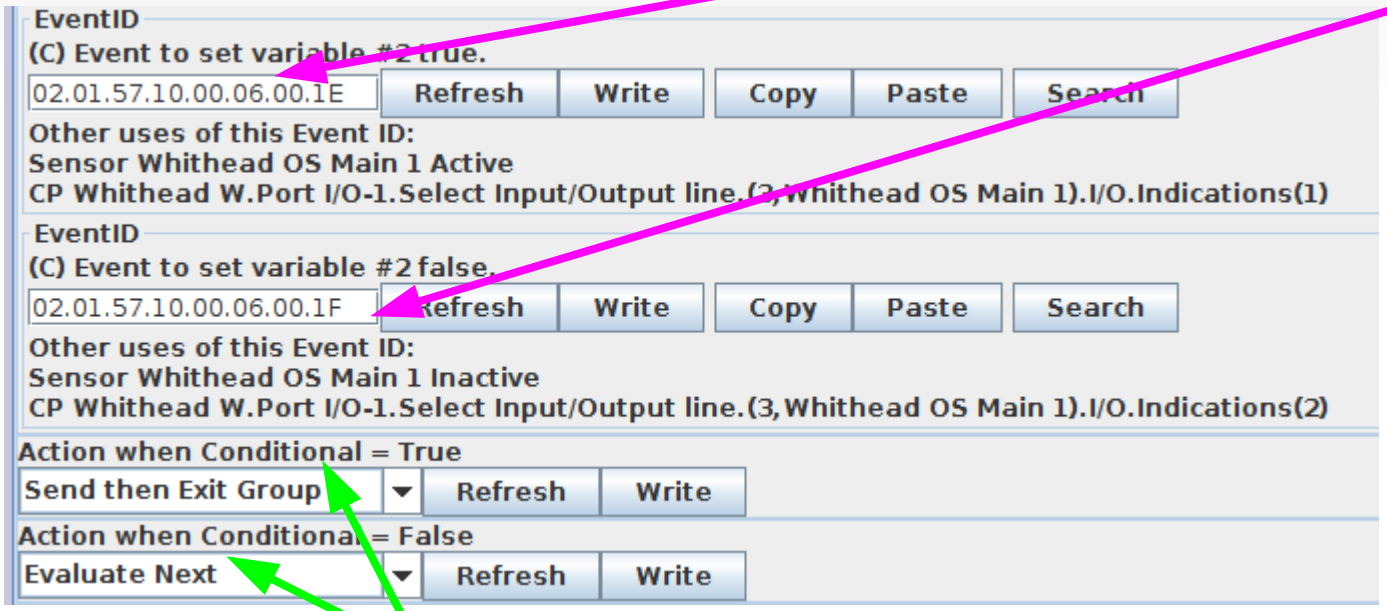
- Enter the logic function. In this case it is 'OR'.

Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	**Not** CTC-Right	OR	OS BOD	Stop	Stop



- For the block detector I copy/pasted from the I/O line into Variable #2.



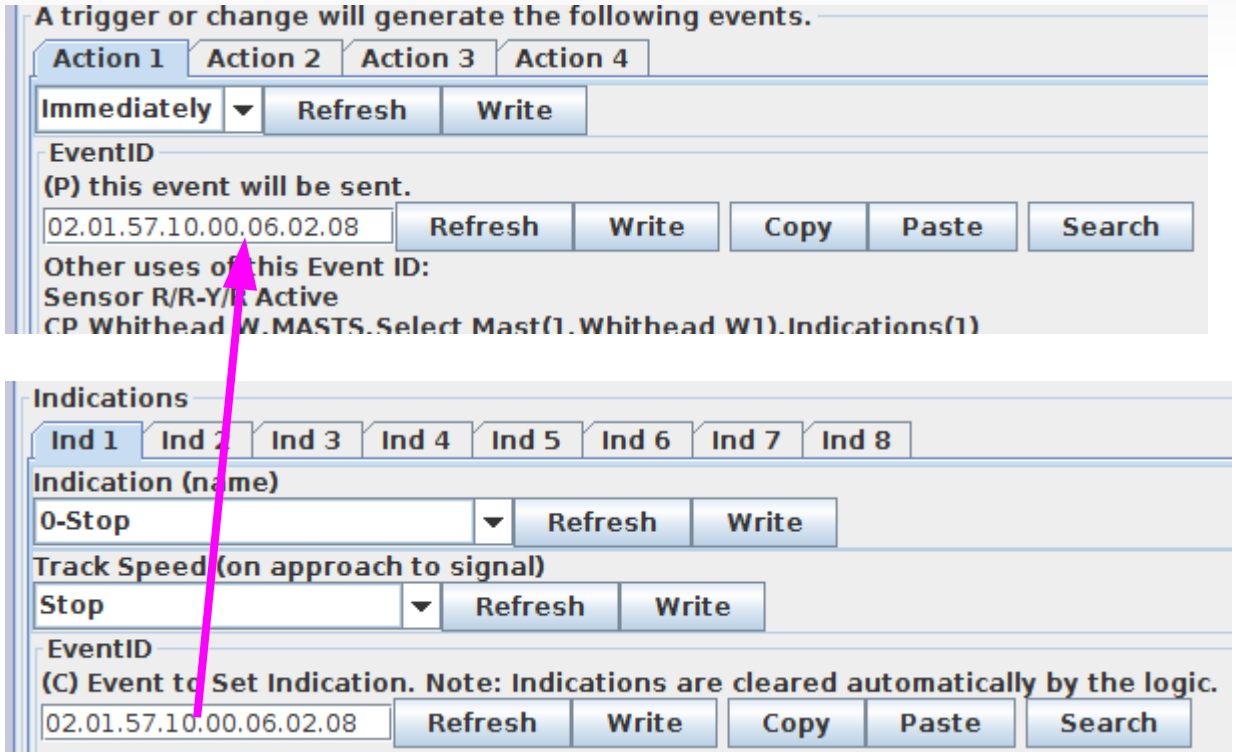
- These default actions are normal for mast logic conditionals. If the condition is true, then any actions are sent, and all less restrictive aspects are skipped.

Logic

Comments	Variable 1	Funct	Variable 2	This Aspect	This Speed
Not CTC-Right / OS occupied	**Not** CTC-Right	OR	OS BOD	Stop	Stop

System Name: MS02.01.57.10.00.06.01.01;02.01.57.10.00.06.01.00
User Name: CTC Lever Whithead RM1

System Name: MS02.01.57.10.00.06.00.1E;02.01.57.10.00.06.00.1F
User Name: Whithead OS Main 1



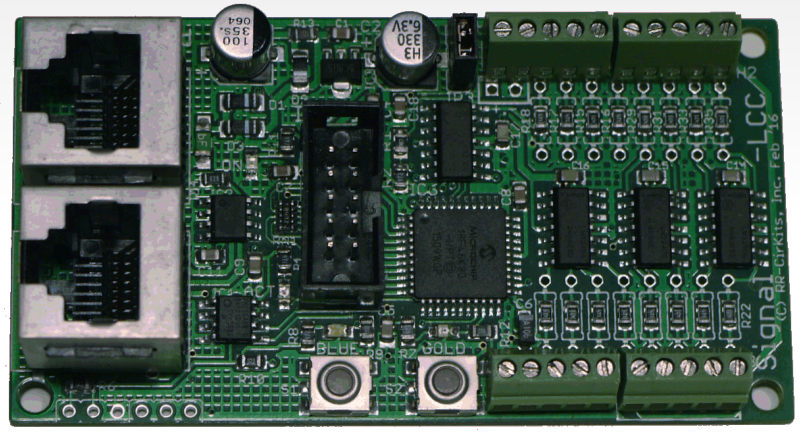
- I then copied the event that sets the Signal rule to 'Stop' into 'Action 1' of the logic. Therefore anytime the CTC direction lever is not 'Traffic Right' or if the OS section is occupied, then the signal will be set to 'Stop'.

Currently Available LCC Hardware

- With the recent addition of an option to place the DCC rail sync information on an otherwise unused pair, CAN can now support smart boosters.
- I have referred to the CAN version of LCC. Remember that the LCC protocol is also capable of being used over different systems, Ethernet, and Wi-Fi also being developed for use by other LCC developers.

Latest LCC Hardware

- The newest node we have designed is a Signal Driver.



Signal-LCC

- True aspect-based signaling
- Easy to configure logic
- Max. 8 signal masts, 16 LEDs
- Up to 32 aspects

Plus 8x I/O lines (like the Tower-LCC)

- # Other Layout Animation

  - Signaling is normally the most complex animation applied to a model railroad layout.

  - Crossing gates and flashers with or without sound is another closely related animation that is often attempted by modelers. Commercial gate animators have various levels of sophistication, from simple on – off, control to reasonably accurate operation. I have seen designers twist themselves into knots trying to figure out how to do it accurately in both directions. However if you think in terms of Events it is actually very simple. Define two blocks. The first covers the entire gate A*pproach* area. The second covers just the highway portion. We call it the I*sland*.

    The Logic:

    1. Approach clear AND Island clear = gates up (requires memory of the two events plus AND logic)
    2. Approach occupied event = gates down
    3. Island occupied event = gates down
    4. Island clear event = gates up

- Traffic signals. Simple flashers to full four or six cycle control.
- Building lighting and signage.
- Day – Night lighting.
- Street and parking lot lighting.
- Operating bridge spans.
- Warehouse doors.
- Mine skips.
- All of the above could be individual devices, or centrally controlled for even more realism. Building lights could follow room lighting, bright in the evening, off late at night, then on again early in the morning. Traffic signals go to flashing mode late at night. Warehouse doors open when trains arrive. Etc.

Acknowledgements

Key OpenLCB Contributors: Bob Jacobsen, Alex Shepherd, David Harris, Stuart Baker, Balazs Racz, Jim Kueneman, Don Goodman-Wilson, John Plocher

Developer Group

10 to 15 actively working on code at any time
25 to 50 regular contributors and supporters
Many of the same people as supporting JMRI

User Group

Started November 2009
July 2016 we had 226 addresses

NMRA liaison: Stephen Priest
NMRA w.g. chairman: Karl Kobel

Info

Yahoo Users Group

openlcb@yahoogroups.com

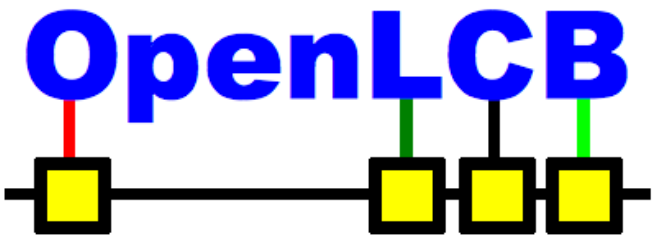
LayoutCommandControl@yahoogroups.com

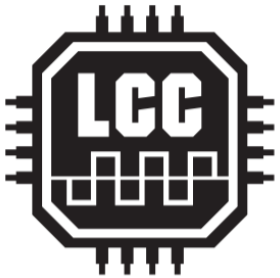
Useful Links

http://openlcb.org

http://openlcb.com

http://nmra.org, choose S&RP scroll to 9.7





Questions

- ?