Block Signals - Richard Maxted

Version 1 – August 2014

This is a guide to setting up and using my block signals. They are based on the default KUJU assets for coloured lights but involve quite a bit of re-scripting.

The Problem

There are two problems for narrow gauge in particular:

- The current set of signals are not really able to handle single line working with passing loops. For many narrow gauge routes the platforms are single sided on one line with either a passing / shunting loop around them or located beyond the platform.
- For most narrow gauge routes a token system is in operation and visible signals are few if any.

It is possible to fake up a solution to these but it requires covering the route in signals which causes a further problem:

In general all signals show up on the 2D (Press 9) Map regardless of their purpose.

Some Basics

Token system routes are divided into blocks. The permission to move between blocks in normal signals is obtained from the signal position / colour but in a block system it generally has to be sought by the driver either from block houses (e.g. Talyllyn) or by telephones connected to a master signal box (e.g. Leek & Manifold).

There is sometimes a token system used in a building en route but, however the system is organised, knowledge of the state of the next block is not visible to the driver on route but has to be obtained by stopping and finding out.

In Railworks knowledge of the state of the block ahead is conveyed via the signals or/and the 2D map. This matches normal practice. A block which is occupied by a train records a message on the signals in that block. To get this to work signals in Railworks pass messages about the occupation of the blocks up and down the line.

In general the signal system in Railworks is uni-directional. It assumes that all trains on a track are moving along it in one direction. There are exceptions but this is the basic premise. Signals pass messages back down the direction of travel to free up blocks behind and pass messages forwards to warn of on-coming trains so that distant signals (Yellow) can be set.

In addition, the signal system carries out a second function which is to ensure the route to the destination is set. A signal will prevent a train running onto a wrongly set point for instance.

Railworks signals do not really control AI traffic. That is largely controlled by block occupation and route set. However, because signals are the way that Railworks records block occupation it means that signals do often cause problems with AI traffic if the block occupation gets in a muddle.

Railworks Signals

It cannot be emphasised just how important it is to understand that, despite appearances to the contrary, Railworks signalling is uni-directional. Every signal has a direction. It faces either up or down. ***Signals only communicate with other signals facing the same way.***

There is probably a way of getting messages passed between the up and down signal systems but so far I have not been able to do this successfully. There are ways (Reverse Junctions) of apparently achieving this effect but in practise it is not a direct communication so much as a means of obtaining the effect by some complex scripting.

There are also signals such as ground signals that do not pass messages but just show if the points are set against the train. Finally there are special signals (Yard Entry) which effectively remove a train that goes past them from the whole block system.

Railworks achieves all this by each signal counting the number of trains in its block a storing this in its Block Occupation Table. As a train enters the block this increments by one and as a train leaves it decrements by one.

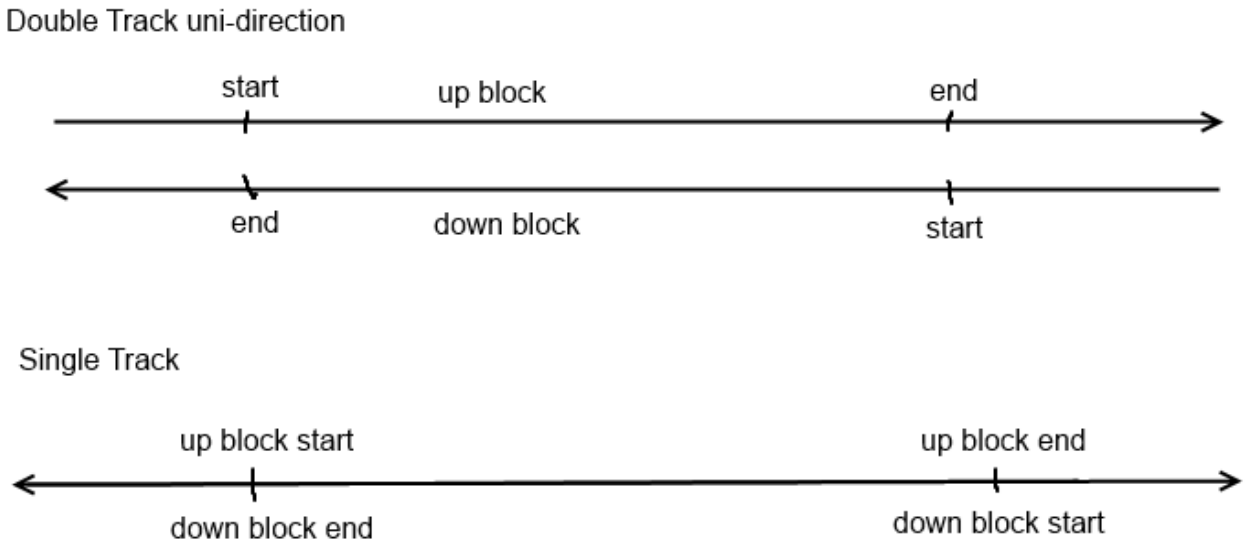
A light bulb moment

Thinking about the difference between token and signal working it became clear that the solution lay in the fact that in a signalled system messages are passed between blocks but in a token system there is no need to pass messages.

If this is the case then there is no need to have signals at all. So long as the 2D map shows the state of each block clearly than the equivalent of obtaining a token can be achieved by looking at the 2D map for the state of the next block. Prototypical practise would suggest this is done at block huts or stations with telephones !

This means there is no need for a signal to pass messages between blocks. All it needs to know is “is my block occupied or unoccupied ?” and show this on the map.

Single line working can be thought of as two quite separate signalling directions, one for up and one for down, which happen to work on the same track. Therefore each block will have to have an indicator to show the occupation of the up block and down block - even though this is on the same track.



If the start of the block is a signal which shows RED for occupied and GREEN for unoccupied then each block will be marked by a signal at each end. One for the up block and one for the down block.

And on the 2D map this will have the effect of showing:

[block A][block B][block C]

Which is exactly what is required. At which point it is pretty much job done except for getting it to actually work !

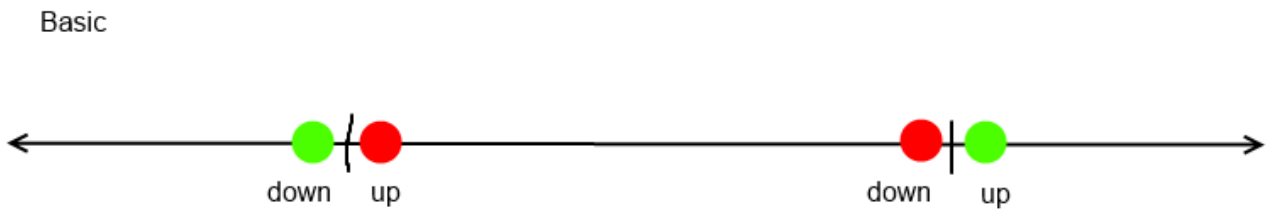
The Block Signals

I have supplied a set of signals:

Name	Purpose
RMax_BlockSig_Basic	This is a simple unit which cuts the line into blocks
RMax_BlockSig_1T	This is the same as above but is used where the entry to the block needs has a point and the block should not show clear unless the point is set. It uses the 1T convention of other signals.
RMax_BlockSig_2T	The same as above but where the line divides into two tracks both of which need to be unoccupied at the same time for the block to show as unoccupied. If either line is occupied the block will show occupied.
RMax_BlockSig_2T1E	A special version that has one of the two tracks that the route divides into as a YARD. This means that if the train enters the Yard the block will reset to being unoccupied even though there is a train in the yard.
RMax_BlockSig_3T2E	As above but has the track dividing into three two of which are Yards where the block is counted as unoccupied. Very useful if there are yards both sides of a through line.
RMax_BlockSig_YardEntry	Many narrow gauge routes terminate in a yard, either a head shunt for a loop or even a direct run into a yard. This simple marker says anything beyond this is a yard and clears the occupation of the block before it. Effectively beyond this the train disappears from the block system.
RMax_BlockSig_PointStop	This as a simple holding point. It does not affect the block occupation at all but is used to ensure that (within a block) AI trains will wait at a particular location for a point to be set for them. It stops trains creeping forward to be close to the point ahead and helps ensure they wait at platforms
RMax_BlockSig_Sem_PointStart	A special I needed for my Leek and Manifold which has two semaphore signals. Rather than have these interfere with the block system They are visual indicators tied to the point stop type signal. They are based on the default ground signals in that they only indicate the point work they cover is set for the route.

All of which sounds complex but isn't. The Yard Entry, PointStop and Sem_PointStart signals are designed not to show up on the 2D map. They control what the traffic does but do not affect the block occupation at all.

How to block out a route - A straight track with no points.



Here is a single line track with three blocks. Thinking of it as two tracks superimposed over each with the middle block having a train in it. So far as the up track (left to right) is concerned the block is occupied so its signal at the start of the block shows red. The next block along the line is green as it is unoccupied.

Ditto for the down line (right to left) it sees the train as occupying its block **as well** and so the signal at the start of the down line block also shows red but the next block shows an unoccupied green.

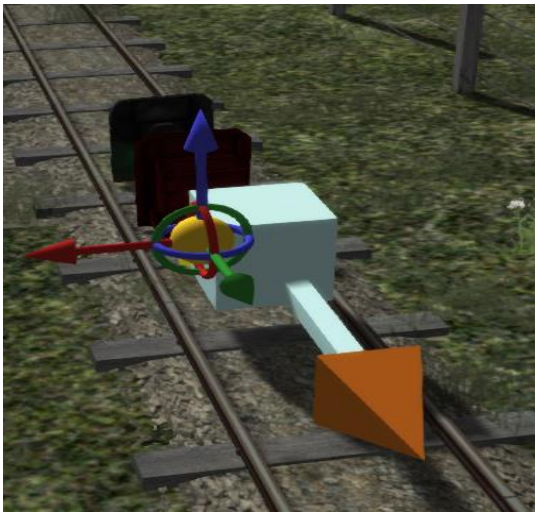
The effect is, of course, the occupied block is bracketed by two red signals and the unoccupied blocks each side start with a green signal. This is the effect that block signalling is trying to achieve where only the start and end block signals show up on the 2D map.

How to lay a block signal - RMax_BlockSig_Basic

Select signals, having first enabled my assets using the blue cube thingy. You will see the basic signal. Place it on the track between the rails facing the direction of travel. The location of the coloured light determines where the dot will show on the 2D map. Between the rails means the dot occurs on the 2D map line.

You will then be given an arrow box which needs to be clicked just ahead. For testing I tend to leave signals above ground to start with but the final part of the process is to bury all the coloured lights below ground so nothing shows in the route.

So now you have one block boundary set in one direction.



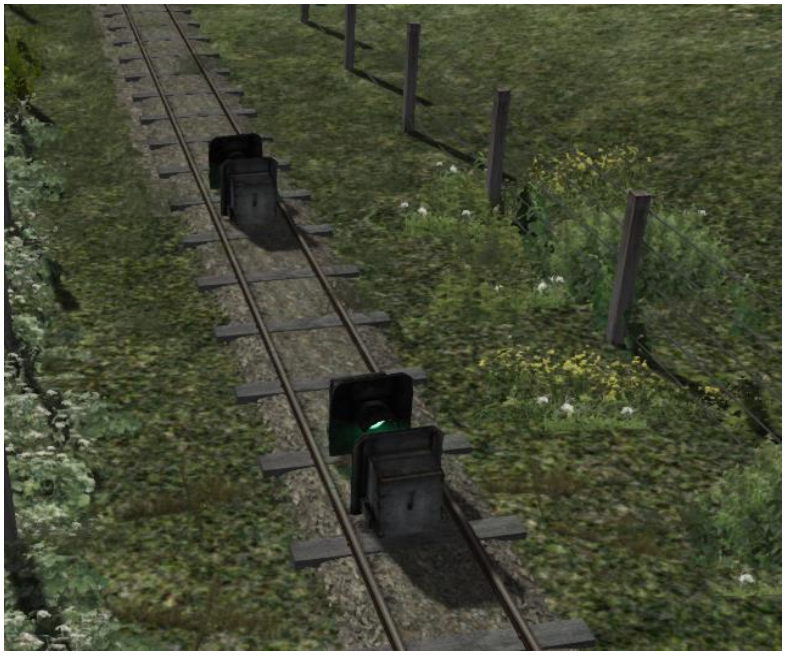
Now repeat the process but in the opposite direction.

Set the thing up so the signal lights face each other and the blue cube thingy pointers point away from each other.

You have now set up the up and down indicators at one block boundary

Repeat the process at the next block boundary and you will have set up the diagram at the start of the page.

This is what it would look like - although the block in this case is rather short being only about 6 sleepers long.



The trick with this is to get the distance between facing pairs of lights just right so that on the map they show as linked but separate blobs of colour. I find about three to four feet is about right to show separate blobs without having to zoom in too much.

At the end you will sink the lights into the ground but leave them for the moment.

How to block out a route - The Loop.

The whole point of this exercise is to enable single track loops which are multi-directional to be handled properly and flexibly.

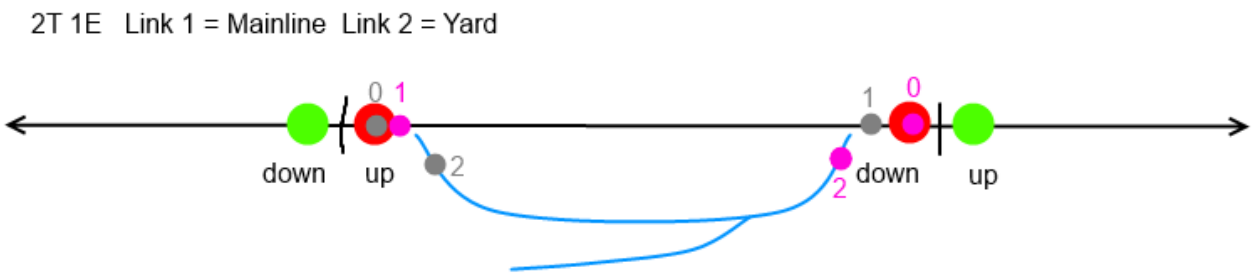
Here is a typical narrow gauge station with a loop, a platform and a siding.

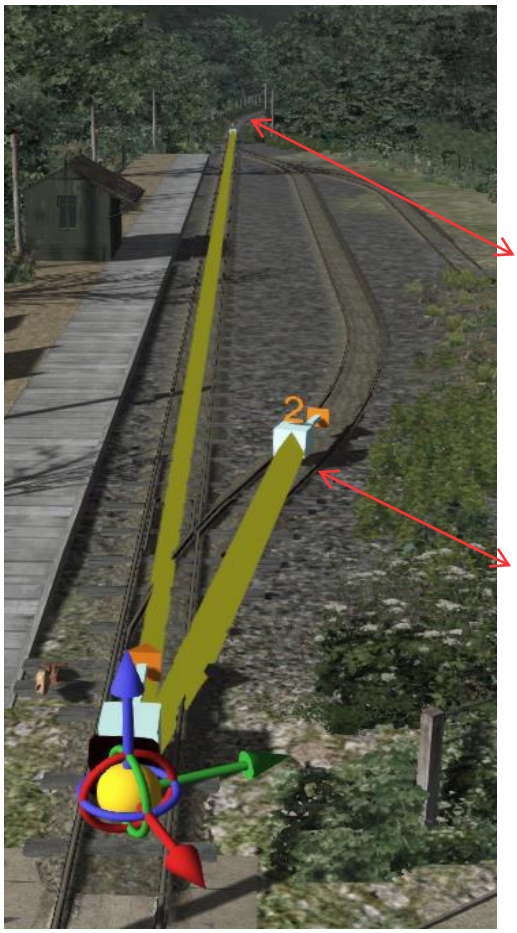
The platform will be used in both directions as will the loop for both passing trains and run round into the siding.

In general I find it easier to treat stations like this as a contained block with the track before and after being separate single track blocks. The reason for this is othewise some of the set up gets too fiddly.

We want to achieve the effect above. Keeping the loops and signals working but still having only block end indicators without any showing signals. But at the same time we need to stop the system from forgetting about a train.

To do this we use the RMax_BlockSig_2T1E signal. If we place the signal link 2 on the loop this means anything entering the loop is forgotten about setting back the block to unoccupied. This what we want to happen. Although in real life the block is occupied it doesn't matter because the next train will go past on an unoccupied part of the line (the platform).





This is how you set it up. Every RMax_BlockSig_2T1E has three blue cube pointy things (which we will call links).

The first link you get to place is Link 0 and that should be placed near the signal as in the basic block signal.

The next link you will be offered is link 1. This is the “mainline” link for the platform and covers the part of the route in which stock will affect the block occupation status.

For a loop this generally needs to be placed at the far end of the loop as close as possible to the throat of the point. This is so that a loco which enters the end of the loop furthest from the signal will reset the occupation.

Link 2 is the one that when passed sends the train into a yard and resets the block to unoccupied. Just place it after the entry point work.

Don't forget that you will need to place an opposite facing BASIC signal to set up the section break just as you did for the straight section above.

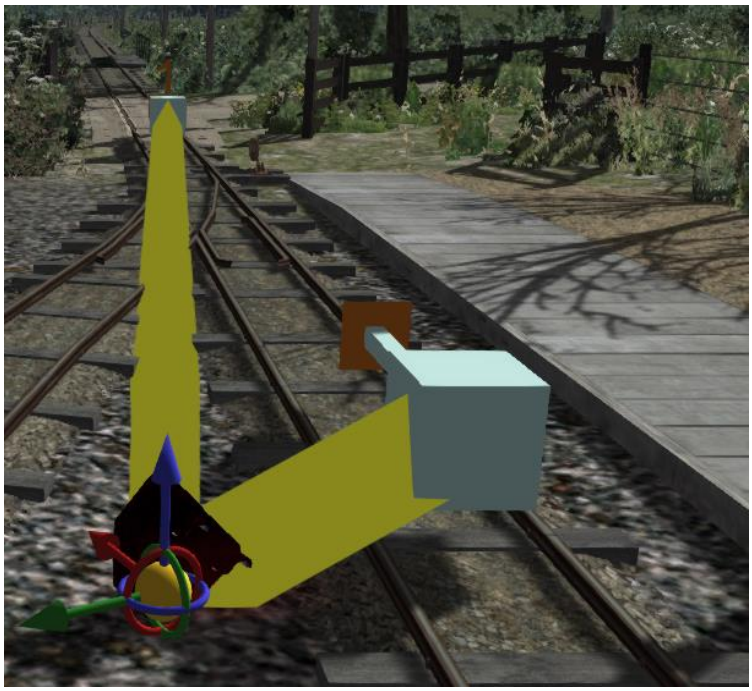
Repeat this at the other end of the station facing in the opposite direction (the purple on the fig above)

What is perhaps surprising is that this arrangement ensures that the loop and anything off it like the siding does not cause the block to be occupied but as soon as you exit the loop the block occupies correctly.

It immediately prevents the dreaded AI Scenario Editor Block Occupied errors.

However, looking at the picture above what can happen is that an AI train leaving the station towards the viewer will pull forwards until it is just short of the points. In the case of the Yard this can actually cause interference if the point exit lines are close. It also looks rubbish for a train to pull a short distance from the station before stopping.

How to block out a route - The Stopper Unit.



The point stopper is a useful gadget that you set out like this.

The light is there for setting out only - you will sink that eventually.

The link 0 is where the train will stop if the route isn't set to link 1 just beyond the point.

Other than that it does nothing and doesn't interfere with anything. It has no impact on the block status and doesn't even show up on the map.

But it means you can control where AI stock holds for a point to clear.

The RMax_BlockSig_Sem_PointStart is just the same signal but has a wooden post signal attached as a visual so you can have a sort of station starter. I also thought it worth including as it shows how to add other visual indication if needed to a point stopper unit.

How to block out a route - The Yard Entry Unit.

Consider Waterhouses on the Leek and Manifold.

Everything after the platform beyond the bridge is frankly yard. I really don't want to bothered by block management in a yard. The Yard entry RMax_BlockSig_YardEntry is simply used to say “Everything after this point is a Yard”. It is a bit clunky to place. Just place link 0 and 1 in a line.

It is also useful if you have a single siding coming off the main route inside a longer block. Just place it after the siding point in the yard the block will reset to unoccupied once the train enters the siding.

How to block out a route - Some things to think about.

In real life a block encompasses a station and the track away from it to the next station. This can be achieved using the block system but if the stations include loops what I was finding is that there was often several miles between the link 0 of a RMax_BlockSig_2T1E signal and links 1 and 2. This made laying the route very difficult and prone to errors.

Looking at what I think happens on the Talyllyn it looks as if the stations and loops are almost outside of the token system being under local control so trains can pass and collect tokens for the line ahead.

Therefore I think for several reasons it is easier to lay out, error trap, and use a route of the loops and stations are a full block in their own right.

What happens at the end of the track ? There is now no problem as the terminus will have an entry block indicator of one kind or another that will go red as a train passes it going into the terminus and green as a train passes it going back out again. However, the terminus often ends in a head shunt and run round loop.

The Loop will need to be identified as a yard with a YardEntry unit but I find it is better to actually include the head shunt as well. This places the Yard entry Link 0 before the loop point on the platform line and link 1 in the head shunt. This has the effect of loosing the loco from the block system once it is in the head shunt.

AI Traffic and the Block System

As I hinted at the start signalling is only part of the mix in how the AI dispatcher deals with scenario traffic. The dispatcher seems to use signal occupation tables to find out the block occupation status but also to use the relative priority of the train type as well. The Dispatcher then tries to set up the point work to route the highest priority train through the block.

However, the dispatcher also looks at the type of train and routes passengers via platforms and goods via loops - sometimes !

Using my Block Signals seems to mean that the dispatcher is generally happy about the block status of loops and will hold a train in one and route the other past it quite happily. He (assuming the dispatcher is a chap) does get exercised if the priorities are wrong and can end up holding a train at the start of the block.

In general the traffic type is only important on single lines in terms of making sure passenger trains are identified as such and that the train arriving in the loop first is a lower priority than the train arriving second. My preference is to make the first train local passenger and the later train express passenger.

Taking my own Leek & Manifold as an example I have been able to route two AI trains past each other in opposite directions. The first arrives in the loop ahead of Ecton platform and holds. The second arrives at Ecton Platform a moment (1 min) later, discharges passengers, and routes into the opposite loop. Both trains then exit the loops at the same time in opposite directions, with the first train stopping at the platform on its way up the line.

Conclusion

This is probably still a work in progress. I would like to be able to develop a unit that detects Yard entry that sends a clear block message back down the line from a trailing yard entry so that link 1 of a 2T1E signal would not need to stretch over both the loop points. But this needs quite a bit more Lua code experience.

I am sure this is an improvement - if only that routes no longer need to be covered in coloured dots or signals but I am also sure that there will be problems.