*Sig-naTrak®*

Model railway electronics by GFB Designs

# *SIGM20*

## Colour Light Signal Controller

# *User Manual*

**Drives up to 8 or 10 colour-light LED Signals from DCC or LocoNet®.**

***Sig-naTrak*® by GFB Designs**

***www.signatrak.co.uk***

67 Boddens Hill Road, Stockport, SK4 2DG, United Kingdom

T: +44(0)161 883 2022 | F: +44(0)161 883 2077 | E: info@signatrak.co.uk

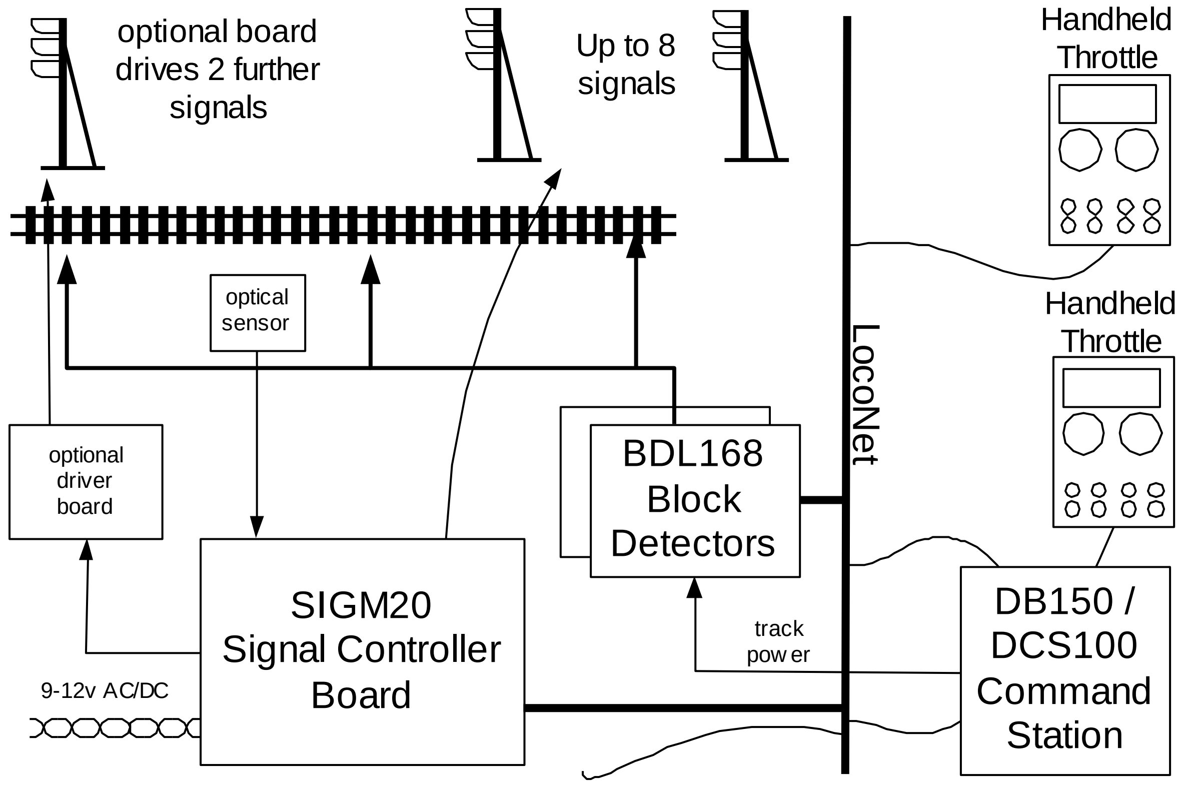
# 1 Introduction

The SIGM20 is an automatic controller for up to 10 colour light signals. The signals are controlled automatically in response to the settings of the track and the passage of trains. This allows complex, realistic operation of a signalling system in response to all of the normal events that happen on a railway:

- Block operation as trains pass block sensors;
- Signals being held at red automatically if the track ahead isn't set correctly;
- Correct operation at junctions;
- Automatic or manual control of "running direction" for single tracks;
- Optional power isolation to halt a train at a red signal.

LocoNet® makes all of this is possible. LocoNet® connects together all of the control and sensor functions within the model railway and allows everything to have access to the information it needs. For example:

- BDL168 and other sensors can send messages reporting whether each track section is occupied;
- Handheld throttles can send commands to change points (switches);
- DTM30 Tower Master boards controlling Centralised Traffic Control and similar 'mimic' panels can send commands to change points (switches) and can display the block sensor states;
- The signal controller can listen to all of the messages about point state and sensor occupancy and determine the correct settings for the signals;
- Signal state can be reported by signal control messages.



# 2 SIGM20 Overview

This section presents a quick overview of what is on the board. The next section presents a longer description of what it will do. For those familiar with it, this product is very similar to its predecessor the SIGM10.

## 2.1 Signal Control

- The board supports 10 signal cells. Each cell works out the state for one “colour light” signal with up to four LEDs.

- Signals are designed to be controlled automatically, after user programming.
  - Each signal can be programmed to drive several different kinds of model including searchlight, three and four aspect signals.
  - Each signal can cascade through track blocks from the next signal, so that for example if the next signal is red, this signal will go amber.
  - Each signal can be programmed to go red if the block ahead of it is occupied. It determines this by detecting sensor messages (e.g. from a BDL-162) on LocoNet.
  - Each signal can be programmed to go red in response to a number of user programmed conditions, for example points being set to a particular state. This allows signals to be held at red if the track work would not allow a train to proceed safely. This can be used to program operation at junctions.
  - Each signal (except where programmed for four aspect signals) may have an “auxiliary” output. This can be programmed to light up according to user defined conditions: for example to illuminate the “feathers” for a diverging junction when the route is set and the train is in the block in front of the signal.

  The SIGM20 is fully compatible with the SIGM10 controller, and uses the same messages. The two can be mixed on a layout without problem.

## 2.2 Signal Outputs

The SIGM20 has 24 output wires to drive LEDs. It can drive up to 8 signals directly.

Some users will want either 4 aspect signals, or some signals with an additional “auxiliary” LED output. This can be used to light the Route Indication “feather” display at a junction. In these cases, the SIGM20 will drive up to six 4 aspect signals.

The outputs are configured in two banks of 12wires. Each bank can be set to drive 3 aspect or 4 aspect signals allowing both types to be mixed on the board for maximum usefulness.

A new feature is an ability to simulate the thermal inertia of a signal lamp, making it brighten or dim over a period of approximately half a second. Thus the signals appear to be "going out" or "coming on" rather than just switching instantly on or off.

## 2.3 Optional Expansion Boards

An optional add-on board can drive two further signals, or up to 8 power isolating relays. This provides a simple way to expand the output from the board.

- When used to drive additional LEDs, the output board will drive two 4 aspect signals.
- When used for power isolating relays, the output board will drive up to 8 relays. These will remove power from a track section immediately after the signal mast to halt a train that has inadvertently been driven through a red signal.

The add-on boards can be obtained from GFB Designs (details on the website). Alternatively we will publish the required design, for home constructors.

## 2.4 Sensor Inputs

The board has 8 inputs. These can be programmed to act as sensor inputs and generate the appropriate LocoNet sensor messages. Also they can be used as local control inputs, to drive a signal to red. This may be useful for manual signalling actions.

When used for sensors. These inputs can be connected to optical detectors (e.g. "IRDOT" from Heathcote Electronics) or to DCC occupancy detectors. The Digitrax BD4 works effectively with the SIGM20; up to two BD4s can be connected.

The SIGM20 will also listen for sensor messages from LocoNet®. These sensor messages are generated by the DAC20, DTM30 and Digitrax BDL168.

# 3 Principles of Signal Operation

This section describes the main signal control functions of the board in more detail, including examples of the programming that could be used to achieve certain effects. This has been written to describe the simpler functions first, with the more complex features being left to later sections. If this appears to be a little daunting, then please break off from it at whatever stage you are comfortable with and come back to it later. There are application note documents available on our website www.signatrak.co.uk which give worked examples of how to achieve control of signals in a variety of circumstances.

A PC based program to capture the board's programming is available free of charge: this is the recommended method of programming the board. Chapter 5 describes how to use this program to specify the programming needed.

## 3.1 Signal Numbering

Signals are numbered in the same way as points: they each have a unique number between 1 and (realistically) 999. Signals need to have unique numbers and not have the same numbers as other points or accessory devices on the layout.

The SIGM20 numbers all its signals consecutively. The number of the first signal can be set to any value in the range 1-2041 but the others follow one after the other.

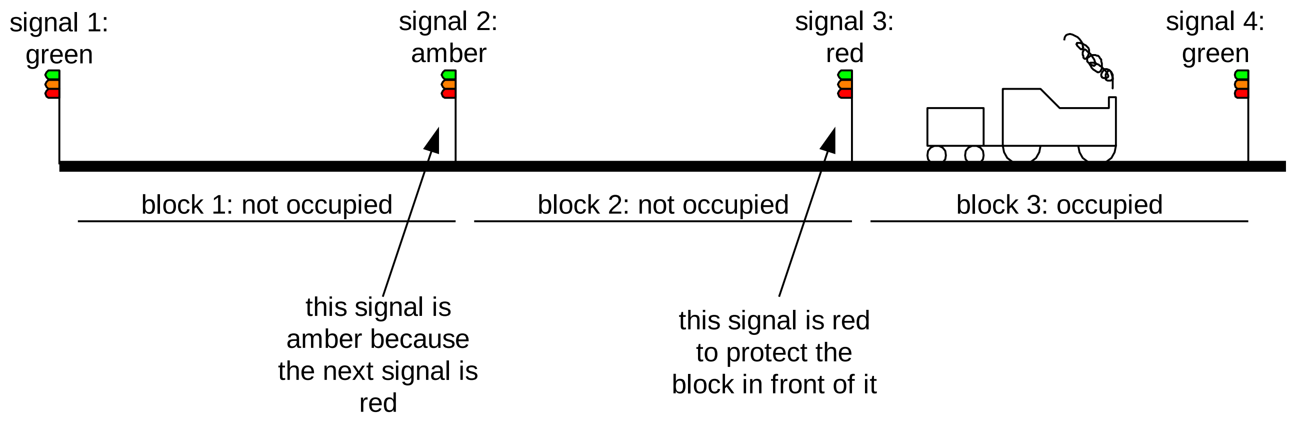
## 3.2 Block Control

### 3.2.1 Description

The basic operation of each signal follows simple block control. This block control approach is used on most lines to keep trains apart from each other. The various other control mechanisms provided act as extensions to block control, but this behaviour is always operating in the background.

The SIGM20 can implement 3 or 4 aspect block control. For 3 aspect signals the following behaviour is implemented:

- Each signal protects a block of track;
- If the block ahead is occupied, the signal is set to red;
- If the block is clear but the next signal is red, the signal is set to amber;
- If the block is clear and the next signal is amber or green, the signal is set to "green".

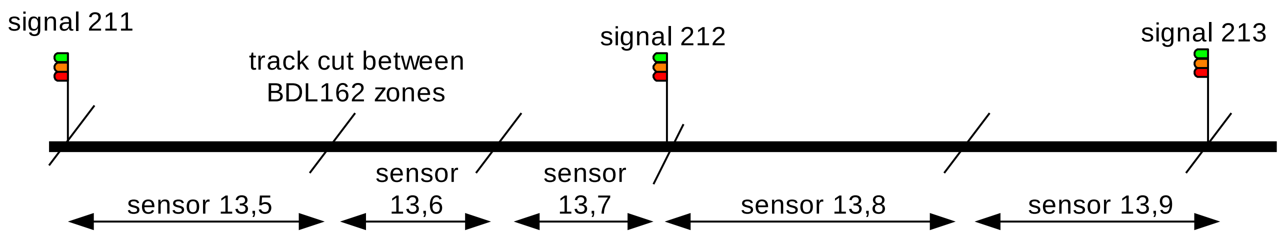


For 4 aspect signals, the same basic approach is used. However a “double amber” aspect is inserted between the “amber” and “green” states. A “double amber” state tells the driver that the next signal is amber.

To implement block behaviour, each signal needs to have two pieces of information programmed:

- The number of the “next signal” at the far end of the block;
- The sensor number(s) of the block occupancy detector for the block ahead of the signal;
- (Because the user may have wired more than one BDL16 zone for the block ahead of the signal, the SIGM20 accepts three occupancy sensor numbers per block; if any one of these is occupied then the block is considered occupied. The extra two sensor numbers can be ignored if not needed.)

### 3.2.2 Programming Example



Signal 211 is programmed with its “next signal” set to 212. There are 3 BDL162 zones that represent the occupancy of the next block. It is programmed “block is occupied when: sensor 13,5 is occupied, sensor 13,6 is occupied, sensor 13,7 is occupied”.

Signal 212 is programmed with its “next signal” set to 213. There are 2 BDL162 zones that represent the occupancy of the next block. It is programmed “block is occupied when: sensor 13,8 is occupied, sensor 13,9 is occupied”.

### 3.2.3 Notes

The SIGM20 implements this behaviour by "listening" to block occupancy sensor messages on LocoNet. These messages can be provided by using BDL162 boards, or by using BD1 or similar sensors connected to an accessory decoder. It is also possible to connect some sensors to the SIGM20 board itself. If the SIGM20 receives a sensor message, then all of the signals will be looked at to see if their blocks have changed state to occupied or not occupied and the signals are updated. When a signal is updated, a signal state message is generated onto LocoNet. By monitoring these messages, each SIGM20 board can determine the state of the other signals so it can find the state of the "next" signal. It does not matter whether the "next signal" is on this board or a different board.

## 3.3 Manual Control of Signals

### 3.3.1 Description

An operator can force a signal to red. This will have the effect of closing down operation on one piece of track. The signals further up the line will see the signal is set to red and modify their aspects to amber etc. following the "normal" block rules. This can be used to hold a train in a station, or to stop a train entering a piece of track (for example to protect a shunting operation).

In the SIGM20, the operator forcing of signal state is done using a DCC accessory command as would be used to control a point. Each signal has its own DCC accessory address (the same as the signal number) and setting that accessory to THROWN will make the signal go red and stay red. When that accessory is set to CLOSED the signal will resume normal operation following its "normal" rules.

A signal can also be forced to red by a manual input – e.g. connected to a pushbutton.

### 3.3.2 Example

To force signal 217 to red, pick up a handheld throttle and use it to set switch address 217 to THROWN. To allow the signal to resume normal block operation, use the throttle to set switch address 217 to CLOSED.

### 3.3.3 Notes

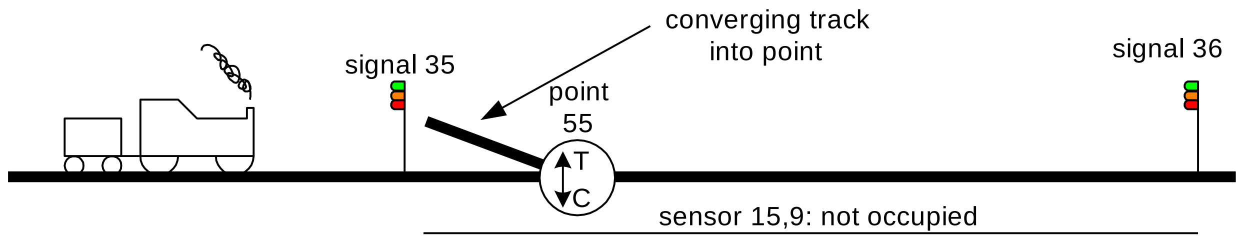
An operator can force a signal to red. Note that an operator ***cannot*** force a signal that is set to red by block control (or for other reasons) to be not red. This is a safety issue: you can force a signal to a safer state but not to a more dangerous state.

Manual control can be initiated from handheld throttles, from CTC panels such as the DTM30, or from PC applications.

## 3.4 More Complex Automatic Control

Additional programming can be used to force signals to red if it is not safe for a train to proceed. This allows the state of signals to be set according to the settings of points (switches) and other effects. This can allow control of junctions: e.g. not allowing a train to pass a signal unless the junction ahead of it is set correctly. This is controlled by adding conditions to make the signal go red if certain events happen. (Conditions are described more fully in section 5.2).

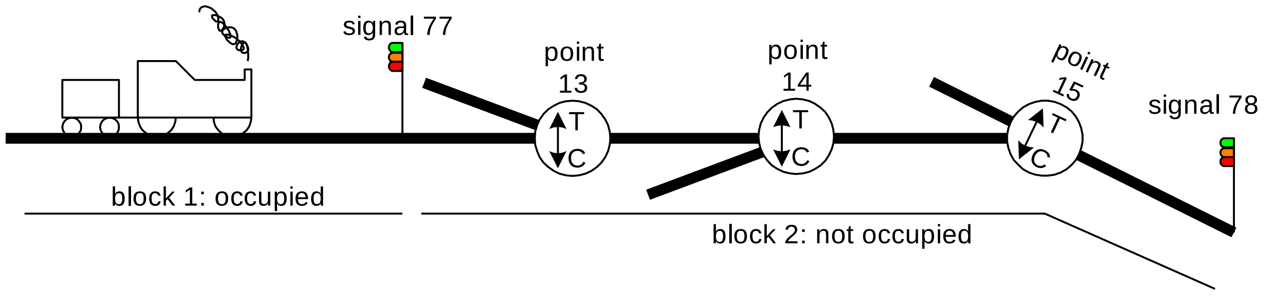
Consider the case of a train approaching a point (switch) on the track:



Signal 35's block ahead is not occupied (sensor 15,9) so according to block rules the signal will not be red. However, there is a point (switch) in the track ahead. If the point is set to "closed" it will be safe for the train to proceed. However if the point is set to "thrown" then the train must halt because the point is set against it. In this case the correct setting of the signal is now dependent on the setting of the point a well as the next signal and the block occupancy sensors.

The SIGM20 allows for this by allowing the user to define "conditions" that control forcing the signal to red. In this case the user would assign a condition for signal 2 "go to red when: point 55 is thrown".

In practice the SIGM20 allows several conditions for each point to be defined. This allows junctions that are more complex than just one point to be defined. Far more complex conditions to be defined if needed, without grossly overcomplicating the programming process.



In the diagram above the user would assign for signal 77:
"go to red when: point 13 is thrown, if point 14 is closed, if point 15 is thrown"

## 3.5 Signal Types & Junctions

### 3.5.1 Block Signals

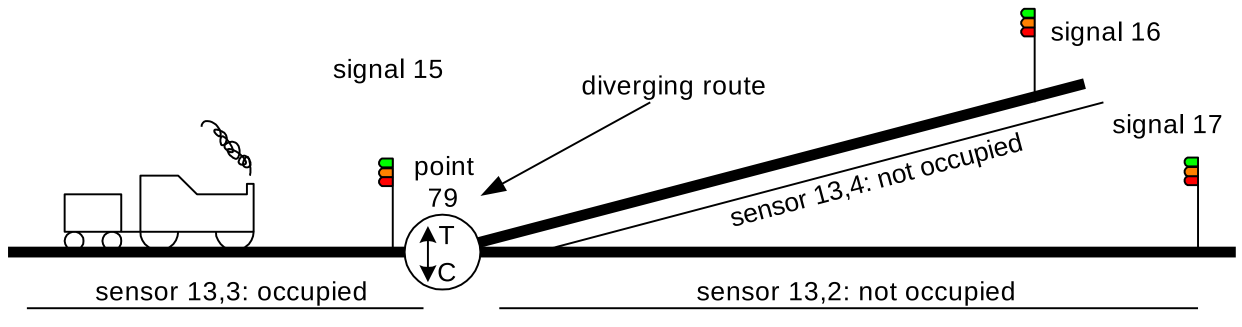
The section above has described the behaviour of block section signals. These can control simple block sections – with no intermediate track – up to automatic control of junctions to keep the signal red until it is safe for a train to proceed.

### 3.5.2 Diverging Junction Signals

However, a problem arises if the train reaches a point (switch) where the track diverges. In this case, there are two possible routes the train can take two possible block sections to consider and two “next signals” to look at. To cope with this, the SIGM20 recognises another type of signal that is programmed differently: the ***Diverging Junction Signal***.

A diverging junction signal recognises that there are two different “next signals” and two sets of track work ahead. The correct set is determined by knowing whether the point at the junction is closed or thrown.

The diverging junction signal is programmed with a point number. It looks at this point state to determine whether it should look at the block & next signal for its “closed” route or the block & next signal for its “thrown” route. It then applies the same behaviour as described for block signals with the user being able to program different behaviours for each of the two possible routes out from the signal.



In the example shown, signal 15 will be dependent on point 79. If 79 is “thrown” it will consider sensor 13,4 and signal 16 to determine its state. If X is “closed” it will consider sensor 13,2 and signal 17 to determine its state.

In the same way as for block signals, the user can assign additional conditions that will force the signal to red. This time there are separate sets of conditions for each of the cases where point X is closed, and point X is thrown. This allows the board to take account of track downstream of point 79 in both possible directions – for example to allow for further points along the line.

Because of restrictions within the microprocessor chip, there is less space available to program the conditions that would force the signal to red for a diverging signal. This is because twice as much information needs to fit into the same total space.

For junctions that are significantly more complicated, the SIGM20 incorporates another concept: the “virtual signal” described in section 3.9. This is a type of signal used to resolve the logic into smaller, simple “chunks”. Using these “virtual signals”, junctions of any complexity can be signalled.

## 3.6 Auxiliary LEDs

There are signals that exist in real life that don’t fit the block occupancy rules. The SIGM20 can drive 4 LEDs per signal cell. When the cell controls a 3 aspect or searchlight signal, a single “auxiliary” LED output is available to be used to construct other signal aspects.

The board supports two uses:

1. Conditions can be used to control automatically a junction indicator (“feathers”) aspect. This is used typically on a diverging junction to tell the driver that he is taking the “turning” route for which a speed restriction would normally apply. The auxiliary LED output will be turned on if the junction’s point has the appropriate state (closed or thrown), and the signal is not red.
2. The board can control the auxiliary LED from two user defined conditions. The LED can be used to construct things like starting signals. Typically an unused DCC accessory address would be assigned and set by the operator. The auxiliary LED would be set to come on if that DCC accessory address was set to THROWN.

For some signal types (for example the “European distant” signal) other effects can be programmed. The “European Distant” signal goes dark (no LEDs lit) if the Auxiliary condition group is true. In this case it allows the distant signal to be dark if the “main” signal is red.

## 3.7 Power Isolation

### 3.7.1 Description

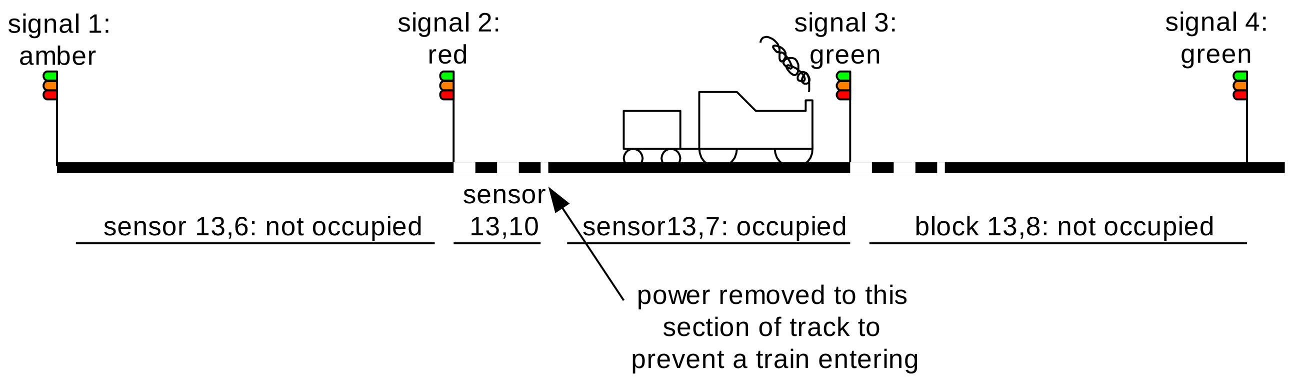
The board sets the signals to the correct aspect as described above to protect the track in front of the signal. In an ideal world the layout operators will observe those signals; they will slow down at an amber signal and halt at a red one. However, there is nothing to force the train to stop.

To provide an extra measure of safety, it is possible for the signal to control a relay that removes power to a section of track just after the signal when the signal is red. The relays are driven by an optional add-on board. This will ensure that a train does not proceed into a protected block if the signal was red. This can be useful in "automatically" controlled layouts to prevent damage occurring to models. Each layout is likely to have a different configuration of the power isolating relays. This will depend on how the sensors are configured.

The board has a facility to "hold off" the isolation under some conditions. The reason for this is to ensure that power is available to the locomotive to clear the signal before the isolation takes place – otherwise a train might never be able to get past the isolation section because the model is dependent on power pickup for its motor.

- There is a programmable delay applied before power is removed to allow time for sensors to be triggered, or for models to clear the zone.
- Additionally, "Isolation zone override" conditions are provided. These allow the user to specify for example "override when: sensor 13,10 is occupied".

### 3.7.2 Example



Two different configurations here are shown.

- Signal 3 has a power isolation zone just past its signal mast. The whole block is sensed be sensor 13,8. When sensor 13,8 first becomes occupied, the signal goes red. After the programmable delay (typically 5-10 seconds) the relay is activated and power is removed. Any train attempting to pass the signal mast will be halted. The relay will remain in this state until the train has exited from the block, and the signal released from red.
- Signal 2 has an additional sensor (13,10) on the power isolation zone. The signal's isolation zone override conditions is programmed as "override when: sensor 13,10 is occupied". Its occupancy is programmed as "block occupied when: sensor 13,7 occupied". As soon as 13,10 becomes occupied, the signal goes red. However the relay will not be activated while 13,10 is occupied: this will be for as long as it takes the train engine to exit from the isolation region. As soon as it does that, sensor 13,7 will be activated; the signal will stay red but the isolating relay will activate.

### 3.7.3 Notes

- As soon as the isolation zone override conditions become true, the signal is set to red. The override conditions don't need to be included in the block conditions.
- The zone is isolated when both the delay has expired, and the isolation conditions are not true. If either the delay is still being counted, or an override condition is true, then the power is maintained.
- Once a signal has been passed and its isolation relay has been applied, then the relay will not de-activate if the isolation zone override conditions become true. Otherwise, a train passing the red signal would cause the relay to deactivate – cancelling the protection mechanism.
- If the signal has gone red because of a manual control operation or a "go red if" condition, then the power is isolated immediately and stays isolated until the signal is no longer red.

## 3.8 Reverse Running Zones

(This section may be skipped at first reading)

### 3.8.1 General

The discussion so far has centred on track where the trains will run in a fixed direction. This is the case with a double track main line: the trains run one way on the left hand track, and the other on the right hand track. In a simple loop of track, there would never be a conflict.

However, in a real railway that almost never occurs. There are sections of track where the trains can go either way; most model railways will have them. Examples of such circumstances include:

- A platform at a terminus station;
- a piece of single track line in a rural area;
- A reception siding in a goods yard that is approached from a double track main line but in either direction.

Reverse Running provides a mechanism to decide under manual or automatic control which direction a piece of track is going to be operated. Each signal can be allocated to a zone with a specified running direction. This allows the signals to be correctly set depending on the running direction of the track.

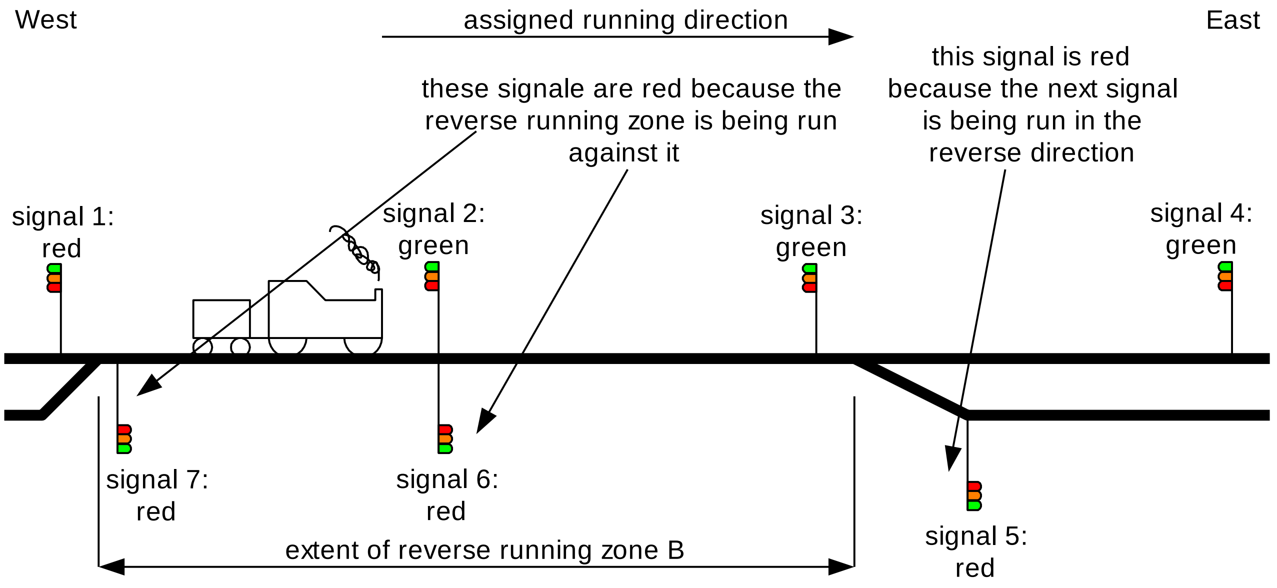
### 3.8.2 Conventions

As a convention that is used in this document, a piece of single track line is assumed to be aligned West-to-East. A train may be running West-To-East or East-To-West. This is an entirely arbitrary distinction and, of course, pleases no constraints onto the orientation of track that can be signalled!

Each signal mast displays an aspect in one direction only along the track: the train driver does not respond to signals controlling trains going the other way on the track. If the model requires two signal masts – one pointing in each direction – then these are treated as two independent signals. Each signal has a specified running direction (West-to-East or East-to-West) if it is assigned to a reverse running zone.

### 3.8.3 Signal Behaviour

The behaviour required where the track can be run in both directions is a little different from that expected on a single track line. If the signal controls the track in the direction it is being run – i.e. the train driver will see and respond to the signal mast - then the signal behaves normally. However if the train approaches the signal from behind, then the signal should be held at red [Because it would be dangerous for a train to proceed in the direction the signal controls] and ***the next signal up the line should be held at red.*** This is because the protected block in front of that signal is about to be entered from the other end.


West
assigned running direction
East
these signale are red because the reverse running zone is being run against it
this signal is red because the next signal is being run in the reverse direction
signal 1: red
signal 2: green
signal 3: green
signal 4: green
signal 7: red
signal 6: red
signal 5: red
extent of reverse running zone B

On the SIGM20, there are three reverse running zones defined (A, B and C). Each signal may belong to one of these, or alternatively to none of them (as would be the case for "normal" double track main line). The zone number and the zone direction is specified for each signal, e.g. for the example above signals 6 & 7 would be assigned to zone B in the East-to-West direction & signals 2 & 3 would be assigned to zone B in the West-to-East direction.

- If zone B is set to run East-To-West, signals 6 and 7 are running in their "normal" direction and signals 2 and 3 will be reverse running. Signals 2&3 are forced to red and the next signal up the line from 2 (signal 1 in this case) will also be set to red.

- If zone B is set to run West-To-East, signals 2 and 3 are running in their "normal" direction and signals 6 and 7 will be reverse running. Signals 6&7 are forced to red and the next signal up the line from 6 (signal 5 in this case) will also be set to red.

### 3.8.4 Zone Direction Modes

Each zone normally operates in "West to East" or "East to West" directions. These directions are used to control signal aspect depending on whether a signal is assigned. When set to one of these directions, the track is expected to be operated in one direction.

The zones can also be put into a SHUNTING mode. In this mode all signals in the zone behave as if they are reverse running, i.e. they display a red aspect and the next signals up the line from them will also be red. This is used where a track section is being operated in both directions (typical for shunting). In this case it is inappropriate for a train to enter from either end, and the signals at all entries to the zone would be held at red.

### 3.8.5 Zone Direction Control

Reverse running zone direction can be set either manually or automatically. This allows trains to set the direction themselves where it is practical to do so, but allows the user to force the setting (as might be required for timetabled operation). The particular behaviour that is possible is highly layout dependent. However broadly speaking it is probably possible for a train to set the appropriate signals for a run through a reversing loop. For a pair of trains on a branch line and sharing a single track it may not be possible depending on the operating practices.

Each reverse running zone has four sets of conditions that control automatic operation. These allow the direction to be set automatically according to point and block sensor states. They also allow the SHUNTING mode to be entered. Full details are provided in section 5.2.3.

Each reverse running zone also has manual controls using two DCC accessory addresses. These allow the zone direction to be set using a handheld throttle. The zone can be set to be fixed (so it cannot be changed automatically) or automatic control can still be enabled. These options allow a useful range of manual, timetabled or automatic control methods to be implemented.

### 3.8.6 Power Isolation when Reverse Running

When a signal is set to red because it is reverse running, its power isolating relay is ***not*** energised and track power is maintained. This is to allow shunting operations etc. to progress normally and for trains to be able to drive backwards through the red signal.

## 3.9 Virtual Signals

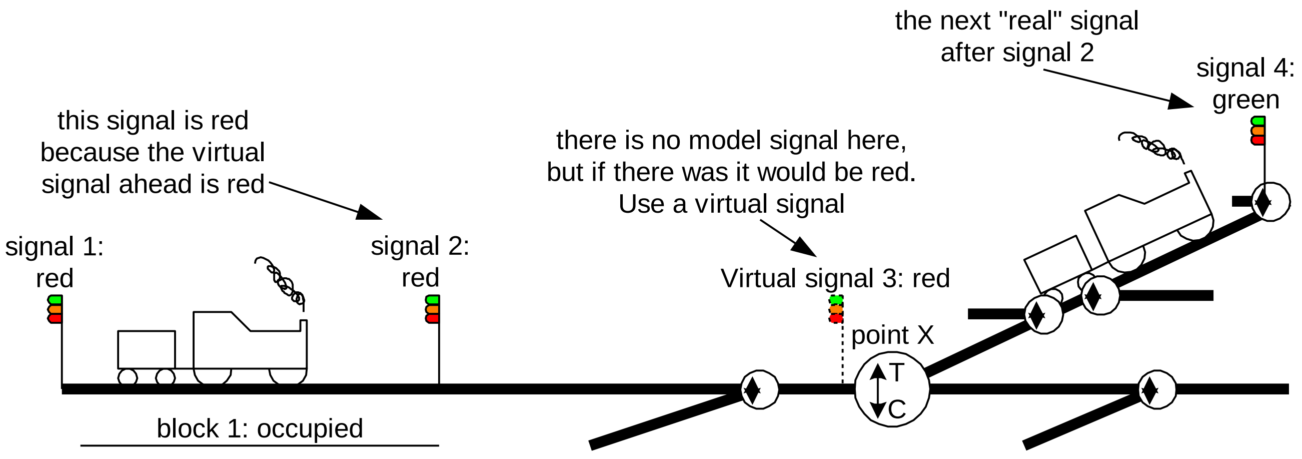
(This section can be skipped at first reading and revisited later)

The SIGM20 signals described so far are intended to be used where a "real" signal would exist on a railway. In most model railways, the complexity of track per scale mile of track is more complicated than in the prototype railway world. This can lead to junctions being grouped together – so that they fit into a room in a house, for example - in a manner more complex that would happen in the real world. Sometimes, it becomes too difficult to program the logic required into one signal for complex junctions and is most likely to arise where there is a diverging junction involved.

The SIGM20 allows the user to allocate one or more SIGM20 signal cells to pieces of track where a "real" signal head will not be placed. These extra signals are used to break up the complexity of the logic: they determine whether a train can proceed past that location based on its visibility of the track ahead of it. These signals are known as virtual signals. There is nothing special about them: any SIGM20 signal cell can be used as a ***Virtual Signal*** and they require no special programming. The only "special" thing is that no signal head is attached and lit up in the model. Essentially Virtual Signals provide a way of breaking up very complex logic into more manageable chunks.

The signals up the line need to know about the virtual signals, however, or they will display the wrong aspect. Consider a signal with a virtual signal ahead of it. Because the track ahead of the virtual signal is blocked, the virtual is set to red. However there is no signal *head* that is set to red. The signal up the line would ordinarily set itself to amber, because its next signal is red. However if the next signal is a virtual signal, it should set itself to the *same state* as the virtual one.

In the example below, signal 1 is red because its block ahead is occupied. The track ahead of signal 2 is too complex for a single signal at resolve all of the required logic. The layout designer has added a "virtual" signal at point 3 (by the diverging junction) to see whether it is safe for a train to proceed past here. However, because of the prototype modelled, there isn't a real signal mast at that location. In this case it isn't safe for a train to pass signal 3 if X is thrown, because another train is blocking the track. The virtual signal is red because a real signal at that point would be red. The next real signal up the line is signal 2: this must go red (i.e. to the same aspect as the virtual signal) because that is the last real signal able to tell the train driver to stop.


the next "real" signal after signal 2
signal 4: green
this signal is red because the virtual signal ahead is red
there is no model signal here, but if there was it would be red. Use a virtual signal
signal 1: red
signal 2: red
Virtual signal 3: red
point X
T
C
block 1: occupied

Lastly, note that if there is no "real" signal at that point, there is no need for a train to halt if the virtual signal is red. (It should have been halted at the signal before it). Therefore no power isolating relay is necessary. The SIGM20 only drives 8 power isolating relays and signals 9 & 10 don't have them. Signals 9 and 10 can conveniently be used for the virtual signals.

## 3.10 <u>How the SIGM20 Works</u>

The SIGM20 is able to work out the correct signal aspect given all of the information needed – sensor occupancy, point states, other signal states. None of this information is permanently stored: after power up, the board knows nothing about the states of those devices. However, LocoNet ensures that all the information is provided. After track power is turned on, the command station sends a set of "interrogation" messages: see Appendix A.

- Sensor states are determined by listening to general sensor messages on LocoNet. These are generated by DTM30, (or the older DTM16) boards and accessory decoders with add-on sensors (e.g. BD1 or optical sensors).
- Point states are determined initially by listening to Output State or Turnout feedback messages on LocoNet. Points that generate a turnout feedback response will have a positive position reporting micro switch attached to the tie bar, and these responses will then always be used. If only output state messages were seen, then the points are not physically sensed and the board uses the initial information provided in the output state message. Thereafter the board detects the DCC accessory packets to the accessory decoder to detect changes in state. [Note that DS54 boards driving solenoid point motors do not accurately report position after power up. In this case either add positive feedback micro switches, or initialise all points to an initial state after power up. DAC12 boards respond correctly.]
- Signal states are reported onto LocoNet using the new <SE> message. Each SIGM20 board monitors these messages to detect the states of other signals. <SE> messages are reported by the SIGM20 at the end of the interrogation sequence.

# 4 Installation & Connections

## 4.1 General

The SIGM20 unit comprises a single circuit board with dimensions 4” x 3.5” as shown in Figure 4-1. It may be mounted onto spacers or pillars using four screws into the four corner holes: 6BA or M2.5mm screws will be ideal. All connections to the SIGM20 are made through the following connectors:

| | |
|---|---|
| Power connections | SK1 |
| LocoNet® Ports | SK2, SK3 |
| Optional Add-on board connections | SK4 |
| Signal LED Outputs | SK5, SK6 |
| Sensor Input Connections | SK7 |

Each connector, and the connections which may be made to it, are described below. (Connections shown for 3 aspect signals; see section 4.4 for 4 aspect wiring)

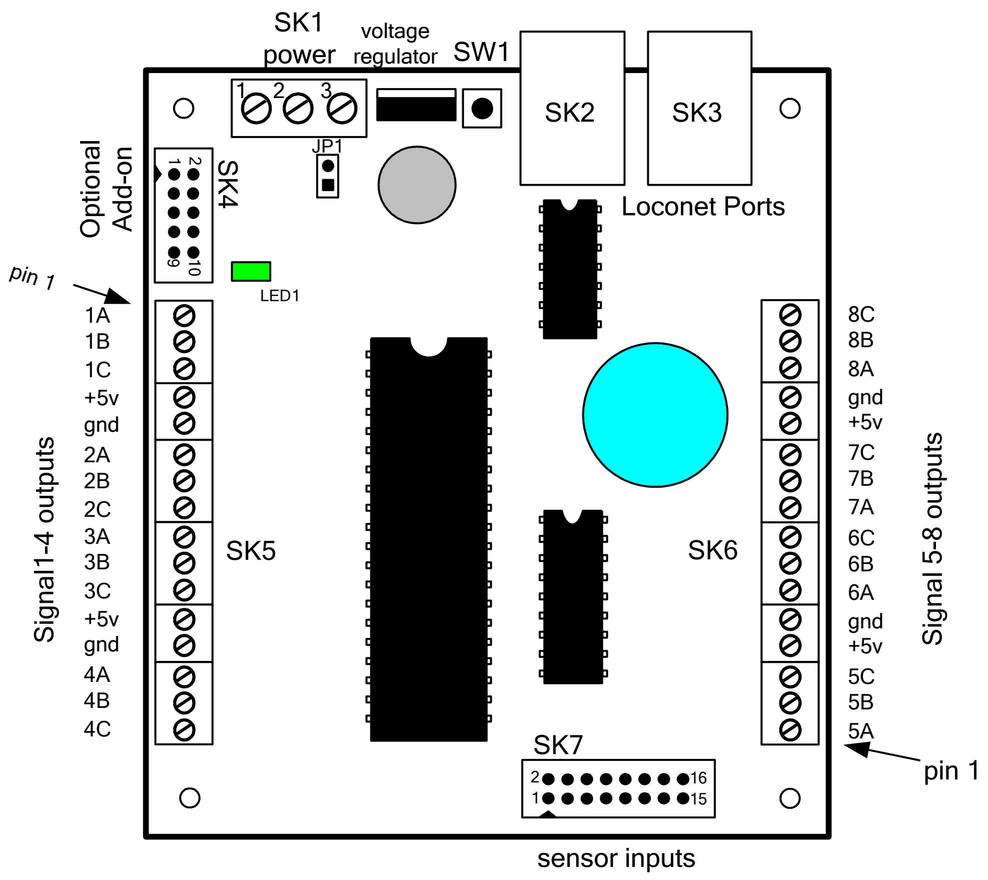


**Figure 4-1: SIGM20 Interconnections**

JP1 should be REMOVED for normal operation.

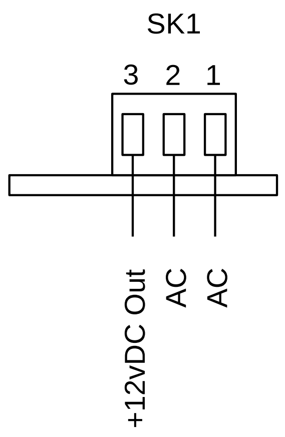
## 4.2 Safety First!

Before beginning to use the SIGM20, there are a few safety points to remember:

- **Hold the board by its edges**. By the nature of its construction, some of the pins on the reverse side of the board are sharp and could cause skin abrasions etc if handled incorrectly.
- **Allow airflow around the board**. Some of its components may run warm in use. Do not obstruct free circulation of air, or allow cloths etc to cover the board.
- **Do not exceed rated operating voltage**. The board could be damaged if an excessive input voltage is applied. **The input voltage must not exceed 12v AC or 16v DC.**
- **Do not handle the board when in use**. The voltages present on the board (<25v DC) are not considered hazardous to health. However if they should come into contact with sensitive parts of the body (e.g. the mouth) a nasty shock might result. The same is true of the voltage on the rails of a DCC (or other model railway) system, so take care!
- Make sure the power feed is isolated from any other power feeds to your DCC system – for example that it is isolated from the booster supply.
- **Don't rest the board when operating on the black bag**: it is conductive!

## 4.3 SK1: Power Connections

The power connector is a 3 terminal screw connector at the top of the board. These terminals accept input wires of a variety of sizes.

|  <br>View looking into connector from board edge | SK1: 3 pin screw terminal | | |
|---|---|---|---|
| | **Pin** | **Function** | **Signal Level** |
| | pin 1 | AC power | Connects to 9-12v AC or DC input.<br><br>Connect to programming track for configuration |
| | pin 2 | AC power | |
| | pin 3 | +12v out | 12v output if required (for connecting push switches to SK7)<br><br>(not normally required for connection) |

In normal use, SK1 is connected to 9v-12 AC or DC in. If DC is used it may be connected either way round. In general, the lower the supply voltage the better. The board needs no more than 9v DC for normal operation. It is important that this power feed comes from a transformer winding that is isolated from any other power feeds to your DCC system. **In particular it must be isolated from any power feeds to the booster(s) and DS54/DAC10 accessory decoders.**

In normal use the SIGM20 derives its operating power from the AC/DC input. In normal operation it consumes up to approximately 130mA from the input if all LEDs are lit. This will be higher if power isolating relays are activated. The power feed can be obtained from many sources and should be readily available within the model railway world. Suitable power supplies from Argos are:

- 982-7253 (300mA: will supply 1-2 SIGM20 boards);
- 982-7538 (1200mA: will supply 4-6 SIGM20 boards).

The voltage regulator (A device with a metal tab near the power input connector) can get very hot depending on the input supply voltage and the number of LEDs lit. It may be necessary to use a heatsink on the regulator device. A piece of aluminium approx. 2” x 3” should be sufficient. Alternatively a suitable heatsink can be purchased from GFB Designs. The temperature is minimised if the input supply voltage is minimised as described above.

If it is desired to reprogram SVs manually using a programming track, the DCC programming track should be connected to the 2 pins of SK1. The DCC programming track input is not polarity sensitive: either rail input may be connected to either pin. Both jumpers need to be inserted to allow programming operations, and removed for normal operation.

## 4.4 SK5, SK6: Signal LED Connectors

These connectors allow connection of external Light Emitting Diode (LED) multiple aspect signals to display the correct trackside aspect. The SIGM20 includes built in current limiting resistors for these LED outputs. These resistors limit the current to approximately 6mA.

- **It is important that the LEDs are connected to these connectors only and not to any other power source e.g. +12v.**
- **Check carefully how to wire the LEDs from the following sections.**

These connectors are identical, and allow 3 or 4 signals to be connected for each “bank” of signals. Connector SK5 is for bank A (signals 1-4); Connector SK6 is for bank B (signals 5-8). Note the wiring is different depending on whether each bank is configured to have signals with 4 LEDs, or signals with 3 LEDs

| Connector | SK5 | 16 pin screw connectors |
|---|---|---|
| **Pin** | **If bank A signals have 3 LEDs** | **If bank A signals have 4 LEDs** |
| 1 | Signal 1 A | Signal 1 A |
| 2 | Signal 1 B | Signal 1 B |
| 3 | Signal 1 C | Signal 1 C |
| 4 | +5v | +5v |
| 5 | GND | GND |
| 6 | Signal 2 A | Signal 1 D |
| 7 | Signal 2 B | Signal 2 A |
| 8 | Signal 2 C | Signal 2 B |
| 9 | Signal 3 A | Signal 2 C |
| 10 | Signal 3 B | Signal 2 D |
| 11 | Signal 3 C | Signal 3 A |
| 12 | +5v | +5v |
| 13 | GND | GND |
| 14 | Signal 4 A | Signal 3 B |
| 15 | Signal 4 B | Signal 3 C |
| 16 | Signal 4 C | Signal 3 D |

| Connector | SK6 | 16 pin screw connectors |
|---|---|---|
| **Pin** | **If bank B signals have 3 LEDs** | **If bank B signals have 4 LEDs** |
| 1 | Signal 5 A | Signal 5 A |
| 2 | Signal 5 B | Signal 5 B |
| 3 | Signal 5 C | Signal 5 C |
| 4 | +5v | +5v |
| 5 | GND | GND |
| 6 | Signal 6 A | Signal 5 D |
| 7 | Signal 6 B | Signal 6 A |
| 8 | Signal 6 C | Signal 6 B |
| 9 | Signal 7 A | Signal 6 C |
| 10 | Signal 7 B | Signal 6 D |
| 11 | Signal 7 C | Signal 7 A |
| 12 | +5v | +5v |
| 13 | GND | GND |
| 14 | Signal 8 A | Signal 7 B |
| 15 | Signal 8 B | Signal 7 C |
| 16 | Signal 8 C | Signal 7 D |

Signal outputs A,B,C and D are connected differently depending on the signal type. This is described in the next sections.

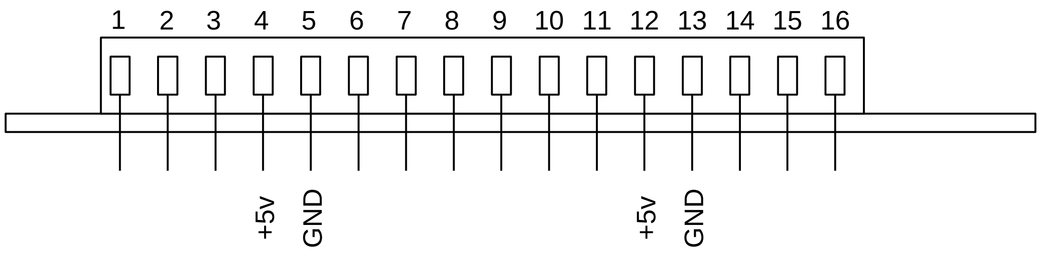


**Figure 4-2: SK5 & 6, view looking into connectors from board edge**

### 4.4.1 Signal LED Polarity

Signal masts are usually wired with one wire for each LED, and a common "return". Depending on the wiring, it may be that the "return" wire needs to go to ground or to 5v It is important to establish which: this varies from model to model and both methods may exist on the same layout.

- For a "common anode" signal, the "common" wire will always be positive and the individual aspect wires will need to be grounded to light the signal. For these signals, the common return wire must be connected to +5v. This is available at pins 4&12 of SK5 & SK6.

- For a "common cathode" signal, the opposite is true: the "common" wire will always be grounded and the individual aspect wires will be positive to light the signal. . For these signals, the common return wire must be connected to GND. This is available at pins 5&13 of SK5 & SK6.

    - This can be determined using a multimeter or <u>momentary</u> connection to a small 9V battery. The "common" wire will often include a resistor. For signals driven by this board, the resistor may be removed to make the signal brighter.

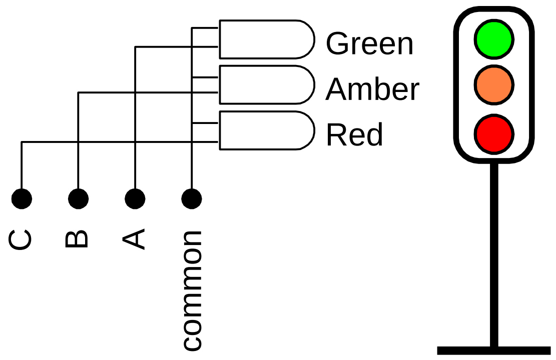
### 4.4.2 Signal Head Types

The board can be programmed to drive 2 aspect, 3 aspect or 4 aspect signals, or searchlight signals. This section describes how to connect each type.

### 4.4.3 3 Aspect Signals

These signals display using separate LEDs for red/amber/green aspects. A 3 aspect signal will have 4 wires including the "return" wire.

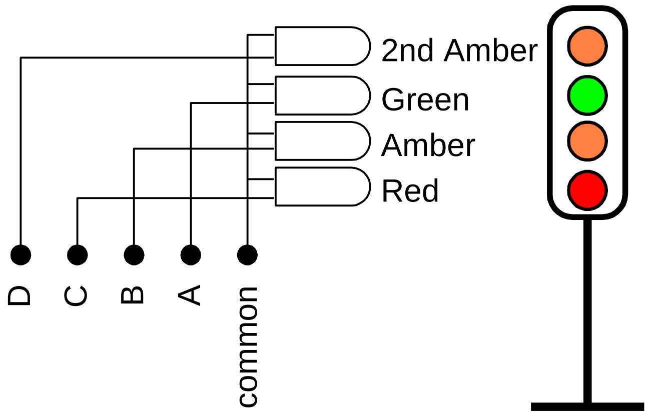
3 aspect signals are normally used with the board programmed to drive four signals for than bank of outputs. If the signal is programmed to be a "US style" signal, it will also display a "flashing amber" aspect but still only need three wires.

| | | |
|---|---|---|
| Common wire | If common anode: +5v<br>If common cathode: GND |  |
| A | Green LED | |
| B | Amber LED | |
| C | Red LED | |

Exceptionally, some "3 aspect" signals also have an extra LED (e.g. to show a "feathers" indication at a junction). If a special signal is needed like this, then a 4th control wire can be used connected to the "D" pin of SK5 or SK6. In these circumstances the bank of signals will need to be selected to be three signals, each with 4 LED outputs.

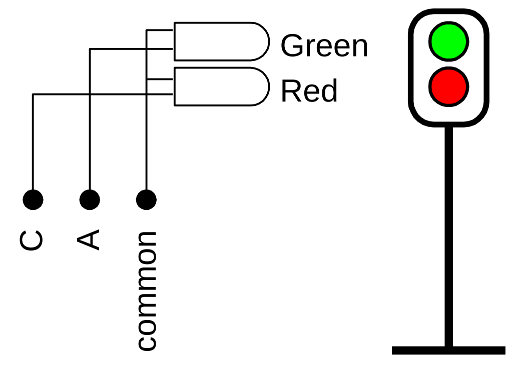
### 4.4.4 4 Aspect Signals

These signals display using separate LEDs for red/amber/green and a second amber aspects. They need to be connected to a bank programmed to drive three signals, each with 4 wires.

| | | |
|---|---|---|
| Common wire | If common anode: +5v<br>If common cathode: GND |  2nd Amber<br>Green<br>Amber<br>Red<br>D<br>C<br>B<br>A<br>common |
| A | Green LED | |
| B | Amber LED | |
| C | Red LED | |
| D | 2nd amber LED | |

### 4.4.5 2 Aspect Signals

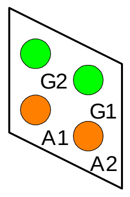
These signals display using separate LEDs for red & green aspects. They need to be connected to a bank programmed to drive three signals, each with 4 wires.

| | | |
|---|---|---|
| Common wire | If common anode: +5v<br>If common cathode: GND |  |
| A | Green LED | |
| B | Always +5v | |
| C | Red LED | |

Pin B is driven to +5v for these signals. This allows a constantly lit LED to be driven if required. Connect between pin 3 (anode) and GND. This may be desired for some forms of 2 aspect signal that indicate by means other than a “red or green” indication: for example some forms of ground “shunting” signals.

### 4.4.6 European “Distant” Signal

This signal type has 4 lamps to indicate that the signal ahead is red, amber or green. This can be controlled by the SIGM20. The signal must be connected to a signal bank for three signals, programmed to have 4 LEDs each.



When signal ahead is green: G1, G2 lit
When signal ahead is red: A1, A2 lit
When signal ahead is amber: G1, A1 lit
When “main” signal on the mast is green, all LEDs extinguished

For this signal to work correctly, the “next signal” must be set to “Virtual”. This means that this signal copies the aspect shown by the next signal. The signal will go dark is the auxiliary LED condition group is true.

The signal head should be wired as follows:

| **Pin** | **Function** |
|---|---|
| A | “Amber 1” LED |
| B | “Amber 2” LED |
| C | “Green 1” LED |
| D | “Green 2” LED |

### 4.4.7 2 Wire Searchlight Signals

“2 Wire Searchlight” signals have a single indicator that can illuminate to be either red, amber or green driven by just 2 wires. These are modelled using an LED that can be either red or green depending on which way round the 2 wires are powered. To indicate amber, the LED is alternated between red & green at high speed and *appears* amber to the eye. The signal will have 2 wires; there may also be a resistor that may be removed.

Using a multimeter or small 9v battery connected momentarily, identify which wire needs to be positive to make the signal red.

| **2 wire searchlight signals** | |
|---|---|
| **Pin** | **Function** |
| A | Wire which when negative makes signal red |
| B | Don’t connect |
| C | Wire which when positive makes signal red |

Exceptionally, some “searchlight” signals also have an extra LED (e.g. to show a “feathers” indication at a junction). If a special signal is needed like this, then a 4th control wire can be used connected to the “D” pin of SK5 or SK6. In these circumstances the bank of signals will need to be selected to be three signals, each with 4 LED outputs.

### 4.4.8 3 Wire Searchlight Signals

“3 Wire Searchlight” signals have a single indicator that can illuminate to be either red, amber or green. These are modelled using an LED that has both red and green in a single package. To indicate amber, both LEDs are lit and it *appears* amber to the eye. The signal should have 3 wires; there may also be a resistor that may be removed.

The signal will have 3 wires. Two of these will need to be connected to a positive supply (test with a multimeter or momentarily with a small 9v battery) to light the read and green internal LEDs. The third (common) wire will connect to the negative supply.

| **3 wire searchlight signals** | |
|---|---|
| **Pin** | **Function** |
| A | Wire which when positive makes signal GREEN |
| B | Don’t connect (*) |
| C | Wire which when positive makes signal RED |

- Pin B is driven to +5v for these signals. This allows a constantly lit LED to be driven if required. Connect between pin 3 (anode) and GND. This may be desired for some forms of 2 aspect signal that indicate by means other than a “red or green” indication: for example some forms of ground “shunting” signals.

Exceptionally, some “searchlight” signals also have an extra LED (e.g. to show a “feathers” indication at a junction). If a special signal is needed like this, then a 4th control wire can be used connected to the “D” pin of SK5 or SK6. In these circumstances the bank of signals will need to be selected to be three signals, each with 4 LED outputs.

### 4.4.9 Auxiliary Aspect Outputs

Exceptionally, some "3 aspect" signals also have an extra LED (e.g. to show a "feathers" indication at a junction). If a special signal is needed like this, then a 4th control wire can be used connected to the "D" pin of SK5 or SK6. In these circumstances the bank of signals will need to be selected to be three signals, each with 4 LED outputs.

## 4.5 SK4: Optional Expansion Connector

This connector can be used to connect an optional expansion board to drive 8 power isolating relays, or alternatively 2 additional four aspect signals.

It can also connect an optional sensor input expansion board, to give an additional 8 sensor inputs.

**Do not make any other connections to this connector**!

## 4.6 SK7: Sensor Input Connector

This connector is provided to allow up to 8 track sensors to be connected to the board. These sensors generate the same LocoNet messages as BDL16 sensors or BD-1 sensors connected to DS-54 or DAC10 accessory decoders. This works well with opto sensors such as the IRDOT, and with BD4 detectors.

- For "IRDOT" type sensors, the sensor relay output should connect between the sensor input pin and +12v from SK1.
- For BD4 sensors, the BD4 pins connect via a ribbon cable either to the first 8 pins or the second 8 pins. A cable is available from GFB Designs if required.

- For pushbutton or toggle switch inputs (used for forcing the signal to red from a local panel), the pushbutton or toggle switch should connect between the sensor input pin and +12v from SK1.

| Connector type | 16 pin header | | | |
|---|---|---|---|---|
| **Pin** | **Function** | **Pin** | **Function** | **Sensor level** |
| 1 | Input 1 | 2 | GND | Active if +12v<br>Inactive if 0v |
| 3 | Input 2 | 4 | GND | |
| 5 | Input 3 | 6 | GND | |
| 7 | Input 4 | 8 | GND | |
| 9 | Input 5 | 10 | GND | |
| 11 | Input 6 | 12 | GND | |
| 13 | Input 7 | 14 | GND | |
| 15 | Input 8 | 16 | GND | |

## 4.7 SK2, SK3: LocoNet Connectors

These two identical connectors allow for connection to a LocoNet® network using conventional 6 pin RJ12 (US style telephone) connectors. The two connectors are wired in parallel: the LocoNet® wiring may be connected to either port, or may be daisy-chained through the SIGM20.

**Do not connect via the "front" connectors on a Digitrax utility panel, e.g. UP3: use only the real connectors. The board will not function correctly because the RAIL_SYNC signals are propagated differently on those connectors.**

| **SK11, SK12** | 6 pin RJ12 |
|---|---|
| **Pin** | **Function** |
| 1 | RAIL_SYNC- |
| 2 | LocoNet® Ground |
| 3 | LocoNet® - |
| 4 | LocoNet® + |
| 5 | LocoNet® Ground |
| 6 | RAIL_SYNC+ |

These signals are defined in the LocoNet® Specification which is available from Digitrax. The board decodes the DCC accessory packets from the RAIL_SYNC signals on these connectors.

## 4.8 Configuring the board

The SIGM20 has over 500 configuration variables. These can be calculated manually and programmed through a handheld throttle. However, this is not recommended because of the large number of variables involved.

Instead, a PC based configuration program has been created which is described in section 6 of this manual. This captures the required settings for the board in a "user friendly" form and works out the required settings for each variable. It can also program these into the board through LocoNet in only a few seconds.

For users that wish to know more about the variables themselves, a separate document is available to download from our website at www.signatrak.co.uk listing the variables and their meanings. It describes how to program the board manually using a programming track if desired.

# 5 Configuration & Control Rules

This section provides more detail and specific rules about board programming. When using the PC configuration program "LocoAnalyse" these rules are usually enforced by the PC software.

## 5.1 Numbering and Addressing

### 5.1.1 Identifying Each Signal

Each signal on the layout must have a unique DCC accessory number. This is used to identify it both for the user (e.g. to be able to manually set a signal to red) and for the software in the SIGM20 boards (to be able to identify other signal states). These numbers must not conflict with any points or other DCC accessories.

The board addresses are defined from a starting address known as the ***Base Address***. The board occupies 16 DCC accessory addresses starting from that address. This is equivalent to 16 points. This means that other accessory devices – e.g. points – must not overlap with it.

For example: if the base address is set to 241, then the board will respond to accessory numbers 241 to 256.

### 5.1.2 Sensor Inputs

The 8 sensor inputs occupy one sensor board address. The same sensor board numbering system is used as for the BDL-162 product from Digitrax: the "sensor board number" is between 1 and 256. This board address must not conflict with any other sensor addresses, e.g. those provided by BDL16, DS54 or DAC10.

The Digitrax BDL16x series of products have 16 sensors per board number. The 8 sensors on the board can occupy sensor numbers 1-8, or numbers 9-16.

It is also possible to connect an option add-on sensor input board to provide an additional 8 sensor inputs. These may also be chosen to be numbers 1-8 or numbers 9-16.

(Note that the sensor board number and the base address for the signal addresses are set entirely independently. They are not interrelated in the same way as DS54 or DAC10 accessory decoder addresses).

## 5.2 Condition Groups

### 5.2.1 Introduction to Conditions

The logic that controls how the signals are affected by the passage of trains etc is controlled by ***conditions***. These are used in the areas listed in the sections below to define the rules by which the signals are set. This section describes what is meant by conditions and what can be programmed into them.

We could define the signal logic with a statement such as:
“if X happens, make the signal go to red”

Actually we want to be able to control the signals according to a list of several things so we might have:
“make the signal go red if A happens, or if B happens, or if C happens”

This is what conditions are. They are simply a list of events where if any of them happen, the specified action for the signal will occur.

Conditions can be declared for the following events:

- A point being CLOSED or THROWN;
- A signal being RED or NOT RED;
- A sensor being OCCUPIED or NOT OCCUPIED;
- A reverse running zone being set to EAST-TO-WEST or WEST-TO-EAST.

By creating combinations of these conditions, very complex signal logic can be constructed; for example:

“make this signal go red if point 45 is thrown,
or if sensor 5 on board 3 is occupied,
or if reverse running zone B is set to west-to-east,
or if signal 29 is red”

It is worth noting that accessory numbers (e.g. point numbers) are not restricted to those assigned to real points and accessories. If a user wants to make something happen under user control, it can be achieved by setting unused accessory numbers to thrown or closed. For example, if point 503 does not exist on the layout, then effects in a SIGM20 can be invoked by setting 503 to closed or thrown using a throttle: this does not affect the operating track because there is no point numbered 503.

### 5.2.2 Signal Condition Group

Condition Groups can be defined for each signal to control the following:

| | |
|---|---|
| Block Occupancy: | For each signal, 3 conditions are available to define when the block ahead of the signal is considered occupied. These would normally be a list of sensor numbers. Depending on layout complexity, one or several sensors might be needed to tell if the block ahead contains a train. |
| Make the Signal “Red”: | For each signal, a number of conditions are available to define if the signal should go red. This section would typically list any points which if set would make the signal go red.<br><br>For a block signal, up to 14 “go red if” conditions can de defined. For a diverging junction up to 5 conditions can be defined for each of the two possible routes out of the junction.<br>(This is in addition to the block occupancy conditions above: the block sensor do not need to be re-listed) |
| Isolation Zone: | For each signal, 3 conditions are available to define if the train is in the section of track immediately ahead of the signal and therefore power should not be isolated. This would typically list any sensor numbers that detect the train position just after the signal. These conditions make the signal go red, but keep track power applied. |
| Auxiliary LED: | Two conditions per signal define whether the auxiliary signal aspect LED should be lit. These might be used to light a starting signal if a DCC accessory address is set to THROWN, or to light a junction signal’s “feathers” automatically. |

### 5.2.3 Conditions for Reverse Running Zones

Conditions can be used with reverse running zones to set the track running direction automatically according to the passage of the trains. For each reverse running zone the following groups of conditions are defined:

| | |
|---|---|
| Force track to “West-to-East” | If one or more of these conditions is valid, they will cause the running direction to be set to “West-to-East”. Typically these would be used to detect the presence of a train entering from the west. |
| Force track to “East-to-West” | If one or more of these conditions is valid, they will cause the running direction to be set to “East-to-West”. Typically these would be used to detect the presence of a train entering from the east. |

| | |
|---|---|
| Don't allow direction change | If one or more of these conditions is valid, they will cause the running direction not to change automatically. Typically these would be used to detect the presence of a train already on the shared track: while it is occupied you would not expect the running direction to be changed. |
| Enter SHUNTING mode | When one or more of these conditions is valid, the zone enters SHUNTING mode and all signals in the zone behave as if they are reverse running. This would normally be invoked by using an unassigned accessory number, manually set by a handheld throttle. |

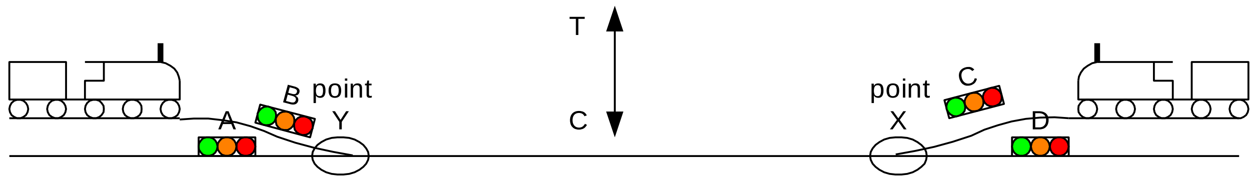
Note that the reverse running zones can be forced to a particular direction by the operators using DCC accessory commands. When forced, the "automatic" settings here will have no effect.

Note also that the reverse running zones will only change state when an unambiguous decision is possible. This is to make sure that the running zone direction doesn't change when a train is ready to enter at one end but a new train appears at the other end. Under these conditions it would be a matter of timetabling – and not automatic decision making – which train was allowed to proceed first.

- To allow a change to "west-to-east", at least one of the "west-to-east" conditions must be true, none of the "east-to-west" conditions must be true, and none of the "Don't allow direction change" conditions may be true.

- To allow a change to "east-to-west", at least one of the "east-to-west" conditions must be true, none of the "west-to-east" conditions must be true, and none of the "Don't allow direction change" conditions may be true.

For example consider an "end to end" layout with two simple termini:



- If Y is set to Thrown, and X is set to thrown, then no train can proceed because they would collide.
- In practice it would not be wise to use the "arrival platform" point setting in the direction logic, because prototypically that point may not be set correctly until the train nears the terminus and is within the shared track zone. It is unrealistic for that point to have to be set before the train can depart.
- On that basis, there is no way of knowing which train will leave first and the direction in which the central track will be run can't be worked out.

The user can set one of the signals A,B,C or D to red manually. Assuming the signal states have been taken account of in the logic, that may be sufficient to allow the direction to be established.

Note that when in SHUNTING mode, the zone direction continues to be set by manual or automatic commands. When the SHUNTING condition is released, the zone will return to the direction that has been set using the "normal" controls. This allows a train to be driven away under signal control after a shunting operation.

| **Zone direction** | **Signals assigned to West-to-East** | **Signals assigned to East-to-West** |
|---|---|---|
| Zone direction = W-E | Aspect displayed normally. | Signal shows red; track power not isolated. |
| Zone direction = E-W | Signal shows red; track power not isolated. | Aspect displayed normally. |
| Zone set to SHUNTING | Signal shows red; track power not isolated. | Signal shows red; track power not isolated. |

### 5.2.4 Complex Conditions

So far, conditions groups have been described that are dependent on one or more individual events happening, e.g. "go red if: point 75 is closed, or point 67 is thrown".

It is also possible to define conditions where two or more events have to happen at the same time, e.g. "go red if "point 75 is closed AND point 67 is thrown". In this case the signal will only go red when both points are set as indicated.

Conditions such as this can be incorporated into the condition groups freely and can be intermingled with other conditions to create complex logic.

## 5.3 Manual Control

The signals and reverse running zones can be controlled manually using DCC accessory controls such as are used to set points to Thrown or Closed. Each signal has its own accessory address; each reverse running zone has two addresses as shown below.

### 5.3.1 Manually Controlling Signals

The first 10 addresses starting at the board base address N control signals 1-10:

- If Address N is set to THROWN, signal 1 is forced to RED;
- If Address N+1 is set to THROWN, signal 2 is forced to RED;
- If Address N+2 is set to THROWN, signal 3 is forced to RED;
- If Address N+3 is set to THROWN, signal 4 is forced to RED;
- If Address N+4 is set to THROWN, signal 5 is forced to RED;

- If Address N+5 is set to THROWN, signal 6 is forced to RED;
- If Address N+6 is set to THROWN, signal 7 is forced to RED;
- If Address N+7 is set to THROWN, signal 8 is forced to RED;
- If Address N+8 is set to THROWN, signal 9 is forced to RED;
- If Address N+9 is set to THROWN, signal 10 is forced to RED.
- If those addresses are set to CLOSED the corresponding signal is set automatically.

Example: the board base address is 201. If address 204 is set to THROWN, signal 4 will be forced to red.

(Note that signals 9 & 10 are those connected to an optional expansion board, if used)

### 5.3.2 Manually Controlling Reverse Running Zones

The next 6 addresses are used to force the state of the reverse running zones.

- If (N + 10) is set to CLOSED, set reverse running zone A to (E-to-W)
- If (N + 10) is set to THROWN, set reverse running zone A to (W-to-E)
  - The running direction is unlocked & can change using the assigned conditions
- If (N + 11) is set to CLOSED, lock reverse running zone A to (E-to-W)
- If (N + 11) is set to THROWN, lock reverse running zone A to (W-to-E)
  - The running direction is locked until a new DCC accessory command to address N+10 or N+11 is used.

- If (N + 12) is set to CLOSED, set reverse running zone B to (E-to-W)
- If (N + 12) is set to THROWN, set reverse running zone B to (W-to-E)
  - The running direction is unlocked & can change using the assigned conditions
- If (N + 13) is set to CLOSED, lock reverse running zone B to (E-to-W)
- If (N + 13) is set to THROWN, lock reverse running zone B to (W-to-E)
  - The running direction is locked until a new DCC accessory command to address N+12 or N+13 is used.

- If (N + 14) is set to CLOSED, set reverse running zone C to (E-to-W)
- If (N + 14) is set to THROWN, set reverse running zone C to (W-to-E)
  - The running direction is unlocked & can change using the assigned conditions
- If (N + 15) is set to CLOSED, lock reverse running zone C to (E-to-W)
- If (N + 15) is set to THROWN, lock reverse running zone C to (W-to-E)
  - The running direction is locked until a new DCC accessory command to address N+14 or N+15 is used.

(Note that to “unlock” a zone direction after it has been locked, a DCC command to address N+10, N+12 or N+14 is used to set the zone direction to an unlocked state.)

For timetabled operation under computer control is it probably appropriate to use the “force” modes.

## 5.4 Detect Signal Passed when Red

The board can detect signals being passed when at red (“danger”) state in some circumstances. This can be done if a sensor is available detecting train presence just after the signal mast. Typically this would be the same sensor programmed as the “isolation zone override” condition.

After the signal has gone red and the “isolation zone override” condition is not true, the SIGM20 knows that the train that passed the signal mast has passed completely. If the “isolation zone override” conditions becomes true again, then the SIGM20 thinks the signal has just been passed again. The board can be programmed to “beep” if this is detected.

This works well with optical sensors. It can be used with a BDL162 type sensor, but not if power isolation is used (because the power to that sensor would have been removed).

## 5.5 External Device Table

The board is programmed and operates through having a table of all the points, sensors and signals that are involved in the signal setting process. This table has limited size and it is possible in extreme cases, that the table will be too large and the programming desired cannot be achieved. Cases where this happens have not yet been found even on complex layouts but it is a possibility. [This restriction is caused by limitation on internal memory in the processor; the approach that has been used maximised the functionality available given the memory available].

Currently the external table allows a total of 48 points & sensors (e.g. 34 points and 14 sensors), and up to 12 signals. Entries are needed in the table for:

- Each “next signal” when that signal is not driven from the same board;
- Each point used to control a diverging junction signal;
- All points, sensor and signals used in conditions for signals or reverse running zones.
- (Note that only one entry is needed for each point, sensor or signal. For example, if three signals have a condition dependent on point 13, only one entry in the table is needed for point 13).

If this limit is reached the PC program will be unable to program the signal logic into the board and will generate an error message. If this should happen, consider the following methods to resolve the problem:

- Simplify the logic used for signalling;

- Make sure the signals around a junction are all driven from the same board: typically the signals around a junction will be dependent on the same sent of points and sensor addresses;
- If necessary, split the signals onto two or more boards.

# 6 PC Configuration Program

A PC configuration program has been created to define, store & program SIGM20 board settings. This can be downloaded from our website at www.signatrak.co.uk.

## 6.1 General

The configuration program captures user settings for these boards and to configure the boards automatically through LocoNet. This implemented as a Windows program. This should run on all common versions of Microsoft Windows from windows 95 onwards. The program provides facilities to:

- Define signal board programming requirements;
- Calculate the required SVs;
- Download the SVs into the target boards through LocoNet;
- Read back boards from LocoNet to create definitions;
- Renumber board serial numbers;
- Read and write arbitrary LocoNet messages for debug purposes.

The SIGM20 uses a LocoNet message to allow the PC to configure the board. It allows the PC to read and write SVs through LocoNet; the board can even be reinitialised to make it adopt the new settings. This mechanism means a board can be fully programmed in around 10 seconds. Manually programming the SVs using the programming track is not necessary (but can be done if wanted).

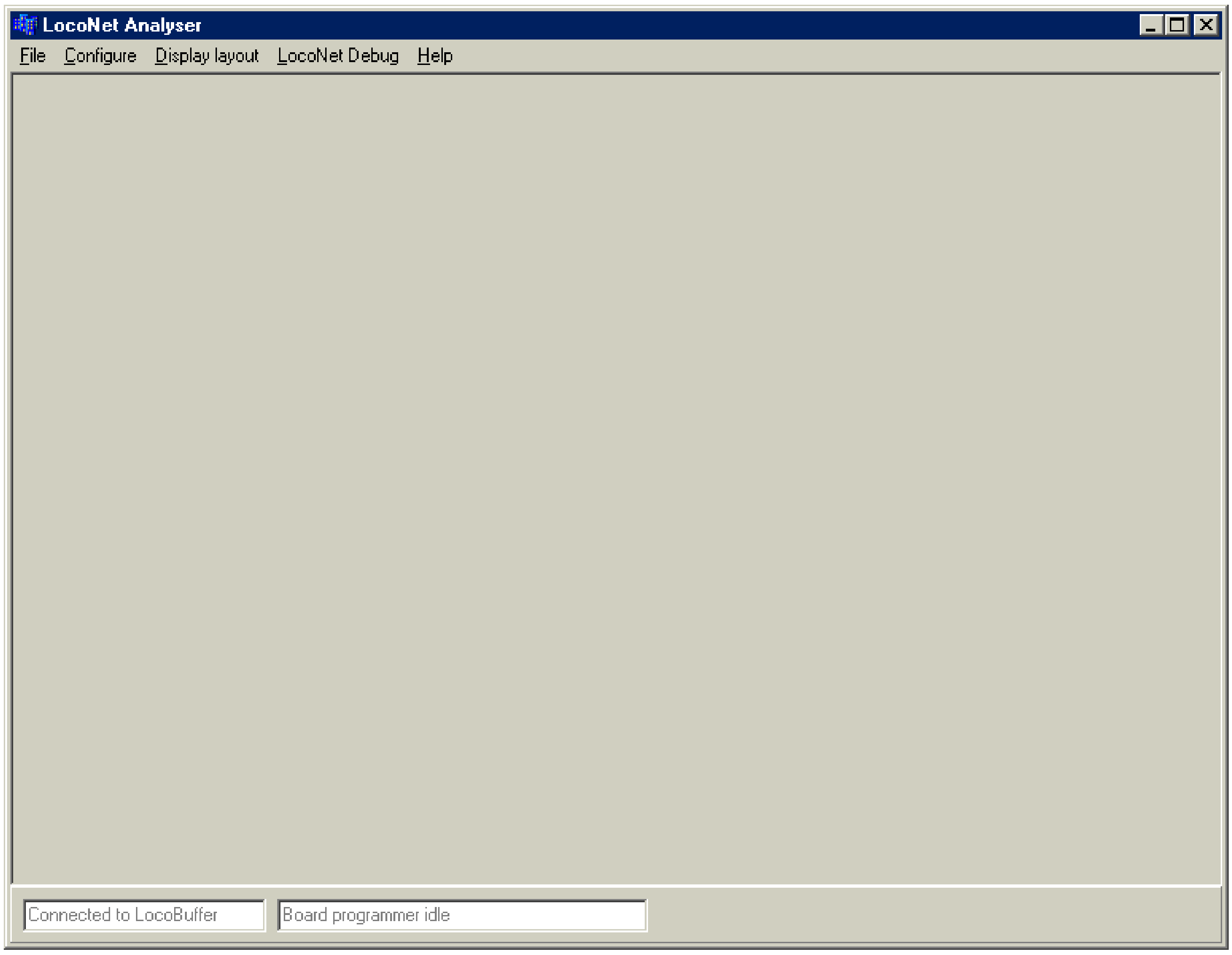
This mechanism relies on the board serial number to identify the board to be configured. This number is factory programmed into each board that supports this mechanism and should be clearly marked on the board. (On the SIGM20, it is marked on the processor label). This number must be unique on the layout: if two board numbers are the same, the programming mechanism will not work. If by chance the serial numbers of two boards are the same, there is a mechanism to change the number of one of the boards. See section 6.9 for details of this process.

After programming SIGM20 boards, they are reset by the PC program and are then ready to execute using their new programming. However they need to re-learn all of the point & sensor states: it is necessary to perform a track power off then a track power on so that all devices re-report their current states.

A LocoBuffer or MS100 interface unit is required. LocoBuffer must have software version 1.5 or greater. This software was first made available around September 2003; older LocoBuffer units can be reprogrammed using a PC. The MS100 is not recommended for complex layouts with lots of message traffic.

## 6.2 Main form

When the program is opened it offers a main form:



The menu allows access to a number of facilities. Note that these menus are augmented as other forms are opened. The initial menu content is:

**File Menu**

| | |
|---|---|
| Connect… | Opens the LocoNet connection form. This allows the user to choose how the program connects from the PC to LocoNet. |
| Exit | Exits the program. |

**Configure**

| | |
|---|---|
| Renumber Boards | Invokes the “Renumber Boards” form to change the factory programmed serial number if required. |
| Configure Boards | Invokes the “Configure Boards” form for definition & programming operations. |

| | |
|---|---|
| **Display Layout** | No facilities yet provided. |
| **LocoNet Debug** | This provides facilities for debugging through viewing and generating LocoNet messages. These facilities will not be needed by most users. |

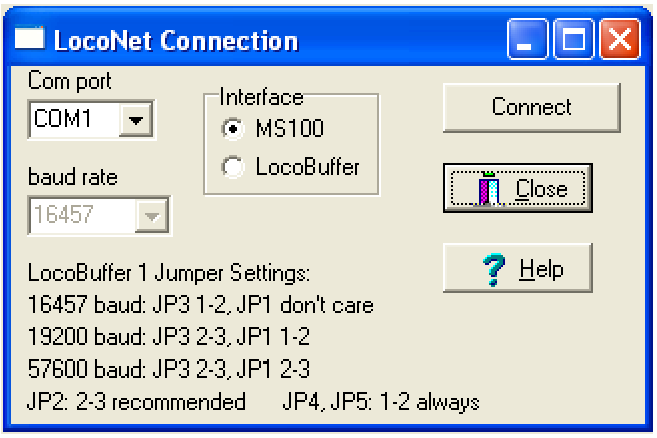
| | |
|---|---|
| Message Display | Opens the LocoNet message display form. This allows the user to capture and display LocoNet traffic for analysis. |
| Generate SWRQ | Opens the "Generate Switch Request" form. |
| Generate Arbitrary Message | Opens the "Generate arbitrary LocoNet message" form. |
| Track Power On | Issues a LocoNet GPON message to turn on track power. |
| Track Power Off | Issues a LocoNet GPOFF message to turn off track power. |

**Help**

| | |
|---|---|
| Help | Invokes the "Help" program |
| About | Displays program version information |

There is also a status bar at the bottom of the form with sections to indicate if the program has connected to LocoNet, and to indicate status of remote board programming.

## 6.3 "Connect to LocoNet" form

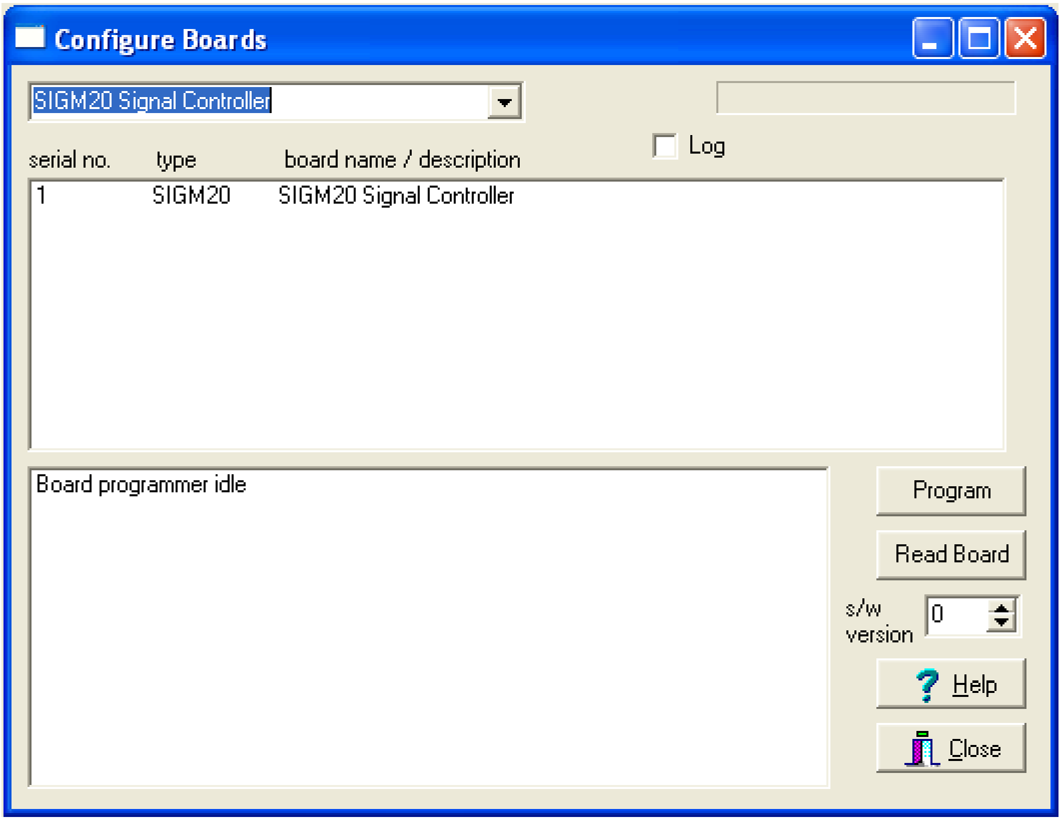


LocoBuffer 1 Jumper Settings:
16457 baud: JP3 1-2, JP1 don't care
19200 baud: JP3 2-3, JP1 1-2
57600 baud: JP3 2-3, JP1 2-3
JP2: 2-3 recommended JP4, JP5: 1-2 always

This form captures the settings needed for connection to the LocoBuffer to allow the program to communicate with LocoNet. The connection settings are stored into an initialisation file and the connection is restored using these settings each time the program is run subsequently.

## 6.4 "Configure Boards" Form

This form is used to store and edit board settings, and to program & read boards through LocoNet. This form is the key form for defining the settings for multiple boards, saving & loading those settings, and programming boards.



| serial no. | type | board name / description |
|---|---|---|
| 1 | SIGM20 | SIGM20 Signal Controller |

Board programmer idle

s/w version 0

| | |
|---|---|
| Board Type Selector | This is used to choose the type of board to be edited. Only board of that type are shown in the list. Currently SIGM10, SIGM20 and DTM30 boards are shown but others will be added. |
| Board List Window | This is the list of boards defined by the user and available to be configure. Most operations require a board in this list to be selected. |
| Message Window | This window is used to display the progress of programming operations. If “log” is checked, additional information is recorded that may be useful in identifying problems. |

When this form is active, new menu items appear on the main form’s menu. The new items are as follows.

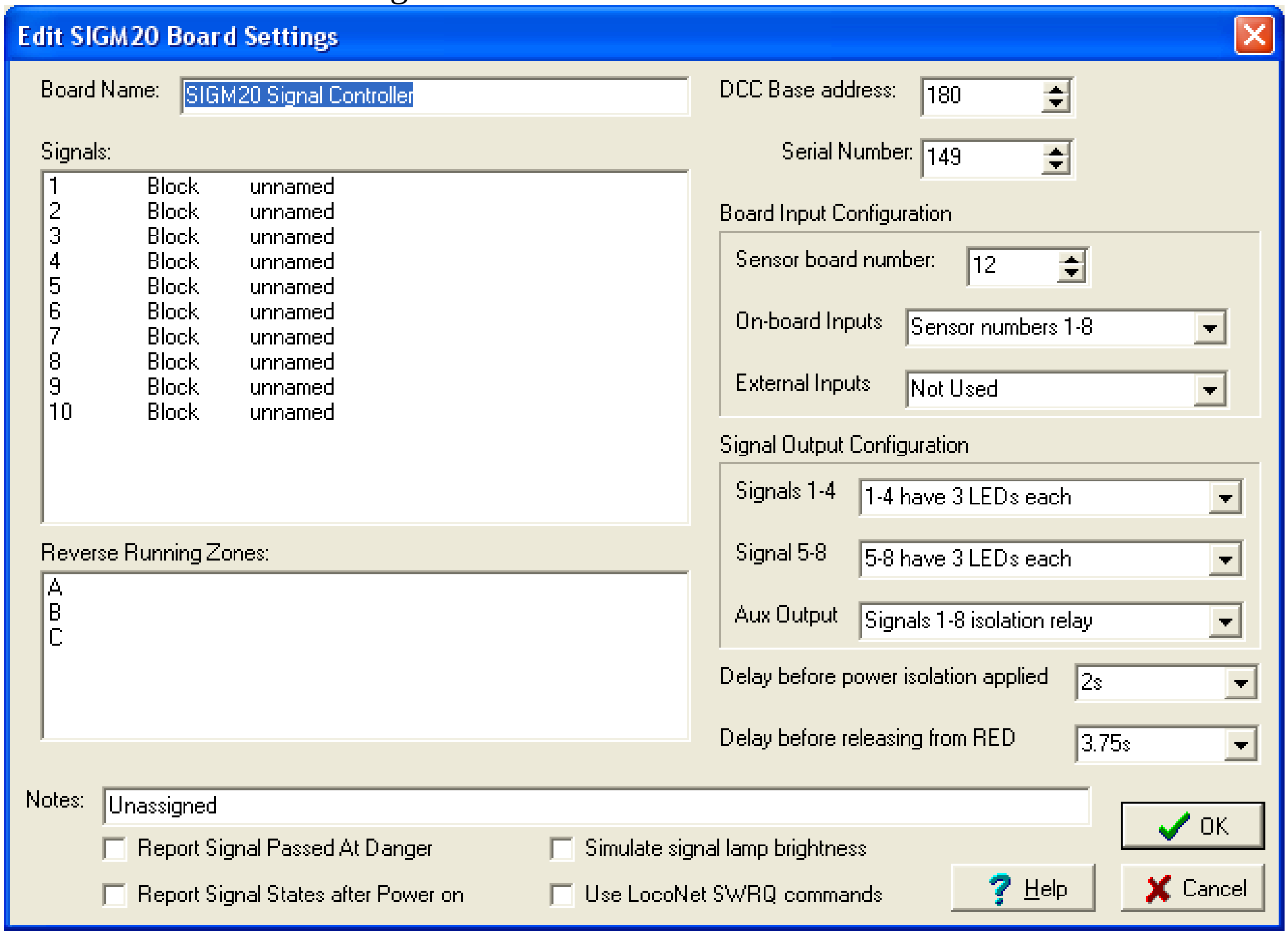
| | |
|---|---|
| File Menu | Additional items are added to open and save files with board configuration information. |
| Open… | Open a file to load board settings. The file will normally have a “.CSV” file extension. A dialog box to select the required file is opened. |
| Save | Save the settings for all of the boards of the selected type to the current file. (Note if you have |

| | |
|---|---|
| | several board types, then separate files are used for each) |
| Save As… | Save the settings to a new file. A dialog box to select the required file is opened. |
| Write SV to file…. | Write the SVs for the selected board to a text file. A dialog box to select the required file is opened. This operation is not commonly required. The board software version must be set into the edit box on the form, because it is not determined automatically. Contact your board vendor for details. |
| Boards Menu | This menu is used to add, edit and delete boards from the list. |
| Add new board | Causes a new board of the selected type to be added to the list. This is necessary before a board can be edited, programmed or read back. |
| Delete board | Delete the selected board. All stored information is deleted. |
| Insert board | Insert a new board at the selected position in the list. |
| Edit Board | Edit the selected board. Opens a form to edit the board’s configuration. (Double clicking on the board list box has this same effect). |
| Display Board | Display the settings for the selected into a new window. These settings can be stored & printed for record keeping purposes, or for more relaxed checking later. |
| Programming Menu | This menu includes items for programming and reading board details. |
| Program Board | Causes the currently defined information to be programmed into the selected board. The board type is checked and the software version is determined from the board, and automatically taken into account. |
| Read Board | Reads back all of the settings from the selected board number. The PC will then have all the settings from the board on the layout. This is useful if the file of board settings is lost. (Note that the “documentation” settings e.g. signal description are not recovered). |

| | |
|---|---|
| Program Button | Has the same effect as the "Program Board" menu item. |
| Read Board button | Has the same effect as the "Read Board" menu item. |
| S/W version | sets the software version to be used when saving the SV values to a text file. This is not commonly needed. |

## 6.5 "Edit Board" Form

This form allows the configuration of a board to be edited:



Board Name: SIGM20 Signal Controller

Signals:

| | | |
|---|---|---|
| 1 | Block | unnamed |
| 2 | Block | unnamed |
| 3 | Block | unnamed |
| 4 | Block | unnamed |
| 5 | Block | unnamed |
| 6 | Block | unnamed |
| 7 | Block | unnamed |
| 8 | Block | unnamed |
| 9 | Block | unnamed |
| 10 | Block | unnamed |

Reverse Running Zones:
A
B
C

DCC Base address: 180
Serial Number: 149

Board Input Configuration
Sensor board number: 12
On-board Inputs: Sensor numbers 1-8
External Inputs: Not Used

Signal Output Configuration
Signals 1-4: 1-4 have 3 LEDs each
Signal 5-8: 5-8 have 3 LEDs each
Aux Output: Signals 1-8 isolation relay

Delay before power isolation applied: 2s
Delay before releasing from RED: 3.75s

Notes: Unassigned

☐ Report Signal Passed At Danger
☐ Simulate signal lamp brightness
☐ Report Signal States after Power on
☐ Use LocoNet SWRQ commands

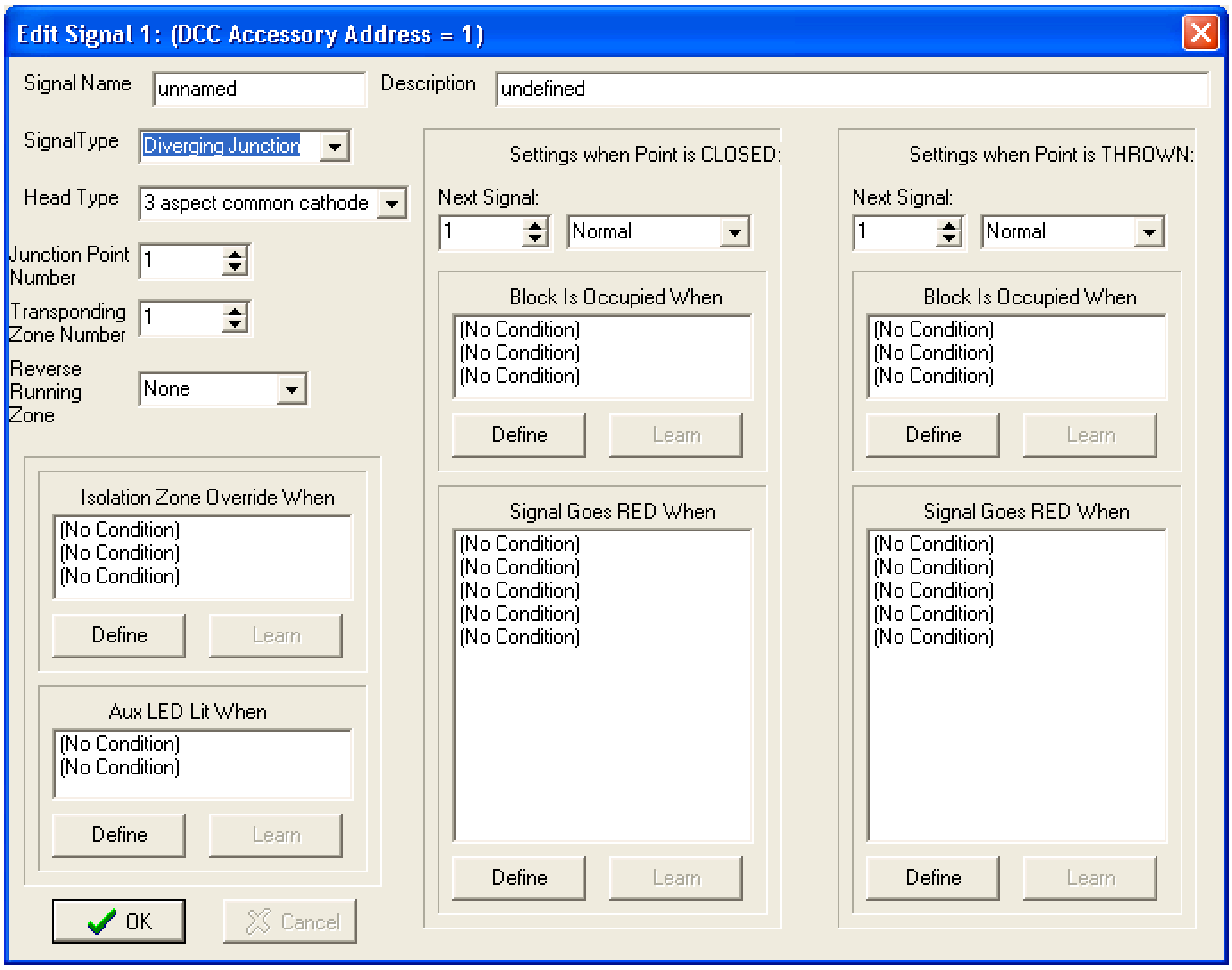
| | |
|---|---|
| **Board Name** | A user specified name for this board, e.g. "signals within Grotbury station". This is not used elsewhere within the database – purely for user reference. |
| **DCC Base Address** | Assigns the DCC address of the board. The board occupies 16 accessory addresses starting at this address. |
| **Serial Number** | The board serial that this information will be programmed into. This chooses the target board on LocoNet; this is the number marked on the processor. |

| | | |
|---|---|---|
| **Input configuration** | | This section of the form defines how the 8 sensor input are used: |
| | **Board Number** | This defines the sensor address used to report the sensor states to LocoNet. This must be different from any BDL16x, DS54 or DAC10 sensor addresses. |
| | **On-board inputs** | This box controls the function of the 8 inputs connected to SK7. The following configurations are available: |
| | Not used | These inputs have no function. |
| | Sensor numbers 1-8 | These inputs generate sensor messages 1-8 to LocoNet. Useful for connecting optical or DCC occupancy detectors (e.g. BD4) |
| | Sensor numbers 9-16 | These inputs generate sensor messages 9-16 to LocoNet. Useful for connecting optical or DCC occupancy detectors (e.g. BD4) |
| | Set Signal Red | When connected to 12v, these inputs make the corresponding signal go red. Useful for manual control. |
| | **External Inputs** | This box controls the function of 8 optional inputs connected to an expansion board. The same settings are available. |
| **Signal output Config** | | This section of the form chooses how the LED outputs from the board user used. The board's outputs are in two banks A & B; also an optional expansion board can be set up. |
| | Signals 1-4 | This chooses how bank A outputs are used. The board can drive signals 1-4 each having 3 LED outputs, or signals 1-3 each having 4 LED outputs. |
| | Signals 5-8 | This chooses how bank B outputs are used. The board can drive signals 5-8 each having 3 LED outputs, or signals 5-7 each having 4 LED outputs. |
| | Aux output | This chooses how an optional expansion board is used. If connected, this can be set to drive 8 LEDs (for signals 9&10) or power isolating relays (for signals 1-8). |
| **Delay before isolation** | | This sets the delay in seconds before a signal's power isolating relay is activated. This extra delay (in seconds) is provided to allow locomotives to clear the isolation zone before power is removed. |

| | |
|---|---|
| **Delay before releasing from RED** | This sets the delay in seconds before a is allowed to change from a red state. This extra delay (in seconds) is provided to allow points with slow motors to change state. |
| **Signals on Board** | A list box to select each of the 10 signals for editing. Double clicking on a signal row in this list opens the signal editor form. |
| **Reverse Running Zones** | A list box to select each of the 3 Reverse Running zones for editing. Double clicking on a zone's row in this list opens the reverse running zone editor form. |
| **Notes** | User notes about the board configuration. Not used by the database. |
| **Report SPAD** | If checked, the board "beeps" when it detects a signal being passed when red. |
| **Report signal states after power on** | If clicked, the board ALWAYS reports signal states to LocoNet, even if it hasn't yet been asked to do so. This option can be used on Non-Digitrax systems to allow normal operation in the absence of a Digitrax Command station. Leave unticked on Digitrax layouts. |
| **Simulate Lamp Brightness** | When ticked, the LEDs simulate "brightening" & "dimming" of the signal's lamps. This makes them turn on or off over a period of approximately half a second. This is a purely visual effect and doesn't affect normal operation. |
| **Use LocoNet SWRQ** | When ticked, the SIGM20 ignores DCC commands and takes "switch" (point) commands from LocoNet instead. Useful on "private" LocoNets with no command station. |

## 6.6 Signal Editor Form

This form allows the settings for each signal to be edited. Some settings are not used for "block" signals (diverging junctions require more information to be stored]. The diverging form is shown here. It contains a lot of information – don't panic!


Edit Signal 1: (DCC Accessory Address = 1)

Signal Name: unnamed
Description: undefined
SignalType: Diverging Junction
Head Type: 3 aspect common cathode
Junction Point Number: 1
Transponding Zone Number: 1
Reverse Running Zone: None

Isolation Zone Override When
(No Condition)
(No Condition)
(No Condition)

Aux LED Lit When
(No Condition)
(No Condition)

Settings when Point is CLOSED:
Next Signal: 1 Normal

Block Is Occupied When
(No Condition)
(No Condition)
(No Condition)

Signal Goes RED When
(No Condition)
(No Condition)
(No Condition)
(No Condition)
(No Condition)

Settings when Point is THROWN:
Next Signal: 1 Normal

Block Is Occupied When
(No Condition)
(No Condition)
(No Condition)

Signal Goes RED When
(No Condition)
(No Condition)
(No Condition)
(No Condition)
(No Condition)

**Signal Name** User text describing the signal e.g. "signal at Goffbury station throat". Not used by the database.

**Description** User text to store any other information. Not used by the database.

**Signal Type** Specifies the type of signal and controls how the signal is set according to the state of the track & input sensors etc. The two types allowed are:

Diverging junction — This signal controls entry into a junction with a point and where there are two possible routes ahead depending on the setting of the point.

Block Signal — This signal controls a track section where there is only one possible route. This may be a block section or a converging junction with only one route – for example an exit from a yard into a main line.

| | |
|---|---|
| **Signal Head Type** | This specifies the type of model used for the signal and controls how the board drives its LEDs. The types recognised are listed. Most can be "common cathode" or "common anode" depending on how the model is wired. |

| | |
|---|---|
| 4 aspect | The model has red, 2 amber & green lights. Typical of UK main line operation. |
| 3 aspect | The model has red, amber & green lights. In use worldwide. |
| 2 wire Searchlight | The model has a red/green bi-colour LED which can be driven to red or green, or to simulate amber by alternating between red & green. |
| 3 wire Searchlight | The model has an integrated red/green bi-colour LED which can be driven to red or green, or to simulate amber by lighting both. |
| European distant Signal | The model represents a "distant" signal used in some European signalling practice. |
| US style 3 aspect | The model has red, amber & green LEDs. It can display Red, Amber, flashing amber & green aspects. |
| US 3 aspect, approach lit | The model has red, amber & green LEDs. It can display Red, Amber, flashing amber & green aspects. Green lamp off unless the AUX condition is true. Set the AUX condition to be a sensor just in front of the signal. |
| 2 Aspect | The model has red & green LEDs, and can display Red & Green aspects only. |

(Other signal types may be added)

| | |
|---|---|
| **Junction Point number:** | Specifies the DCC accessory number of a point associated with the signal. This is only important for diverging junctions. |
| **Reverse Running Zone:** | Specifies the reverse running zone, if any, to which the signal belongs. |
| **Transponding Zone:** | Specifies the transponding zone number to which the signal belongs. Not currently used. |
| **"Settings When Closed"** | This section specifies the behaviour of the signal when the associated point is "closed". (For block signals, there is no associated point and only this section is enabled on the form). |

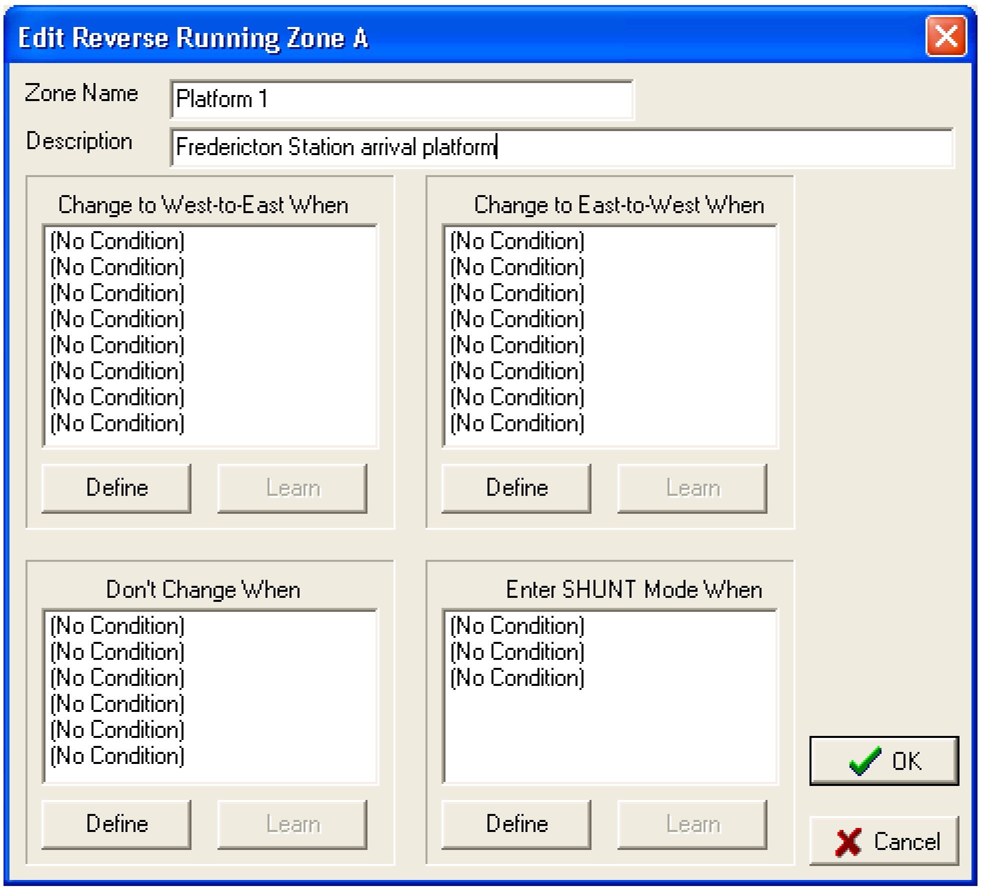
| | |
|---|---|
| Next Signal | Specifies the DCC accessory number of the next signal after the current one if the point is closed. The combo box allows the next signal type to be specified: |
| Normal | Specifies that this signal should behave normally: if the next signal is red this one will display amber etc. |
| Virtual | If the next signal is virtual, this signal will be set to the same aspect as the next signal unless the local conditions force it to red. Used to construct complex logic. |
| Always Green | There is no next signal: this one can only go to red or green. |
| Always Red | There is no next signal: this one can only go to red or amber. Used for the last signal before buffers in a terminus station for example. |
| Block occupied when: | A set of conditions that determine if the block ahead of the signal is occupied. If one or more of these conditions is met, the block is considered occupied and signal is forced to red. (Up to 3 conditions are allowed to cope with the case where the intervening track has been broken into several BDL16 zones, or the case where optical sensors are also used). Use the “define” button to edit these. |
| Signal goes red when: | A set of conditions that force this signal to a red, “danger” state. If one or more of these conditions is met, the signal is forced to red. Use the “define” button to edit these. |

| | |
|---|---|
| **“Settings When Thrown”** | This section specifies the behaviour of the signal when the associated point is “thrown”. The same settings as the other section are available and are not repeated here. For block signals this part of the form is disabled. |
| **Isolation zone override** | A set of conditions that override the power isolation relay. If one or more of these conditions is met, the relay is disabled and power is always applied to the track after the signal. (This is provided to allow power to be kept on if an optical sensor or BDL zone indicates that a train is still passing the signal. It can also be used to disabled the relay completely by manual point operation). Use the “define” button to edit these. |

| | |
|---|---|
| **Aux LED lit when** | A set of conditions that control an auxiliary LED for this signal. These auxiliary LEDs can be used to control route indicators or starting signals. Typically the entry here would be a DCC accessory address to light the signal if set to thrown. Use the "define" button to edit these. |

## 6.7 Reverse Run Zone Editor Form

This form allows the settings for each reverse running zone to be edited. These settings control the running direction of the track according to track state and the passage of trains. The settings can also be forced using DCC accessory packets (for example for timetabled operation).


Edit Reverse Running Zone A

Zone Name: Platform 1

Description: Fredericton Station arrival platform

Change to West-to-East When: (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition)

Change to East-to-West When: (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition)

Don't Change When: (No Condition) (No Condition) (No Condition) (No Condition) (No Condition) (No Condition)

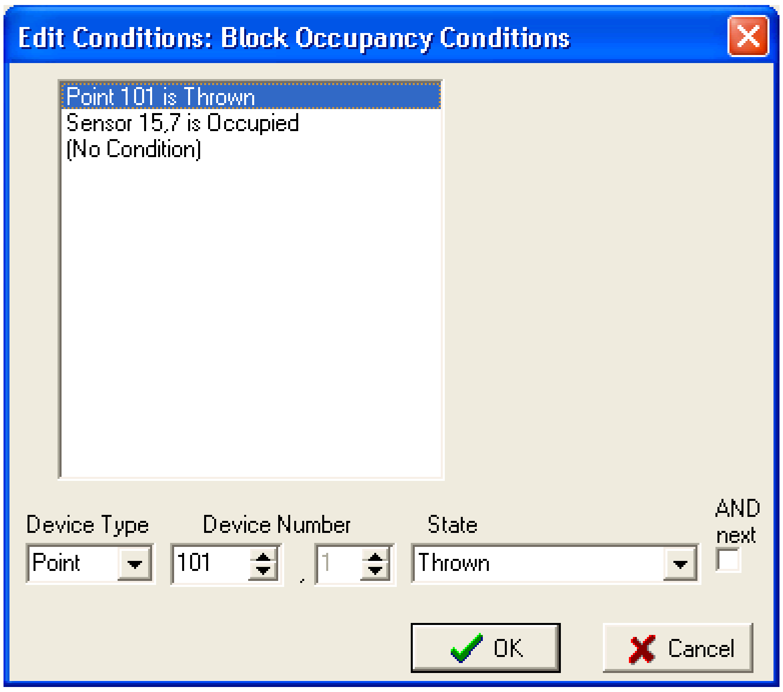
Enter SHUNT Mode When: (No Condition) (No Condition) (No Condition)

| | |
|---|---|
| **Zone Name** | User's name for this zone. This is for information only. |
| **Description** | User's documentation describing the zone. |
| **Change to West-to-East** | A set of conditions that can force the direction of the zone to West-to-East. If one or more of these conditions is met, and none of the East-To-West conditions is met, and none of the "freeze" conditions are met, then the direction of the zone will be changed. Typically these |

| | |
|---|---|
| | conditions would indicate a train is ready to enter a single track from the "West" end. |
| **Change to East-to-West** | A set of conditions that can force the direction of the zone to East-to-West. If one or more of these conditions is met, and none of the West-To-East conditions is met, and none of the "freeze" conditions are met, then the direction of the zone will be changed. Typically these conditions would indicate a train is ready to enter a single track from the "East" end. |
| **Don't change when** | A set of conditions that can freeze the zone state. If one or more of these conditions is met, then the direction of the zone will not be changed automatically. Typically these would be set by block occupancy detectors in the one direction track, to stop its running direction changing when a train was in the single track section. |
| **Enter SHUNT when** | A set of conditions that, if true, buts the zone into "shunt" mode. In this state the zone behaves as if it is reverse running in both directions: the signals are held at red but power isolation is disabled. This will be used to enable shunting operations. |

## 6.8 Condition Editor Form

This form is used to edit condition sets that drive signal behaviour.


Edit Conditions: Block Occupancy Conditions

Point 101 is Thrown
Sensor 15.7 is Occupied
(No Condition)

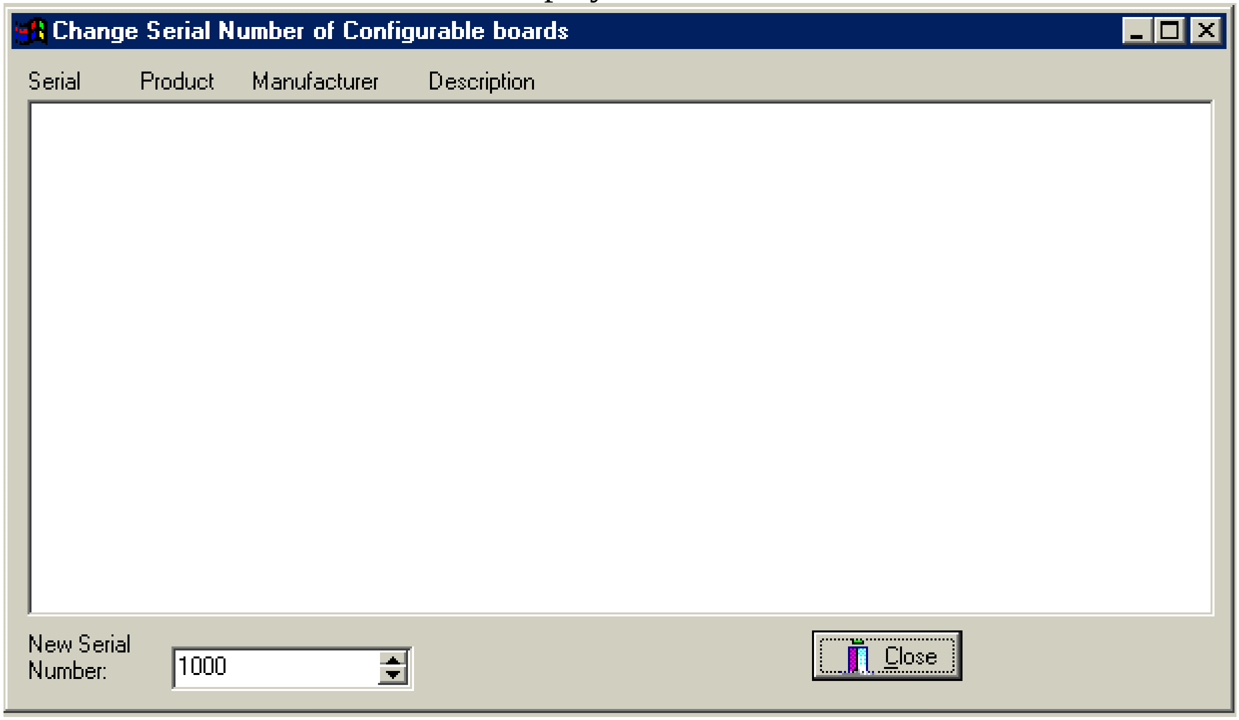
Device Type: Point
Device Number: 101 . 1
State: Thrown
AND next: ☐

The list box shows the appropriate number of conditions that are allowed for the particular condition group being edited. The selected entry is edited using the controls at the botton of the form. Any conditions not needed should be set to [No Condition]. Please note that any conditions defined *after* a [No Condition] entry will be ignored.

| | |
|---|---|
| Device Type | Lists the type of condition: Point, Sensor, signal, reverse running zone, or none. |
| Device Number | Lists the device number. Points and signals have a single number; sensor numbers are listed in the format *"board number, sensor number"* |
| State | The device's state when the condition is true. |
| And Next | If checked, this condition is only valid of the next condition is valid too. This is used where it is necessary for both things to happen – e.g. "go red if point X is closed AND point Y is thrown". |

## 6.9 "Renumber Boards" Form

This form is used if it is necessary to change a board's pre-programmed serial number, or to allow a list of all boards to be displayed.



When the form is active, a new menu item appears: "Board Numbering". Its menu items have the following actions:

| | |
|---|---|
| Discover Boards | A list of all configurable boards is filled in. This is done by requesting through LocoNet for all such boards to identify themselves. The list on the form will show all boards after the operation completes (this takes a few seconds) |

| | |
|---|---|
| Change Board Address | This option is used to change a board serial number. DO NOT USE THIS COMMAND UNLESS YOU ARE SURE YOU WANT TO PROCEED! |
| Identify Board | The board selected in the list is requested to identify itself (e.g. by beeping, flashing LEDs or whatever action the board manual specifies). This is useful to “find” a specific board on a layout. A SIGM20 will respond by beeping. |

Renumbering boards is only needed if a duplicate board number exists and this will not commonly be necessary. It might be necessary if, for example, a SIGM20 with serial number 1234 is purchased and another configurable board from a different supplier also has the same serial number. In this case, one of the two boards must be renumbered. The steps required are:

1. Decide which board to renumber (the number is used only for PC configuration and has no other effect).
2. Choose a new board number in the edit box at the form’s bottom left. The number can be any number as long as it is unique on the layout.
3. Select the board to be renumbered in the list box on the form.
4. Click the “Change Board Address” menu item. The selected board will be renumbered.
5. Relabel the serial number on the board. This is vital because the board will now only respond to its newly allocated number. Do it now before forgetting! (If this number is forgotten, it can be found later using the “Discover” command).

## 6.10 LocoNet Debug Section

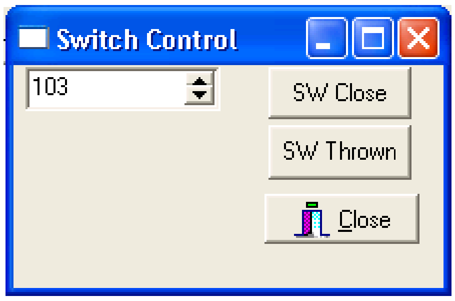
### 6.10.1 Message Display Window

This window displays all received LocoNet messages. It is intended for experienced LocoNet debuggers only.

- If the “Message expand” button is clicked, the message is expanded to report more information for certain message types.
  - This can be useful if you have forgotten which sensor number is which: put a loco onto the track and see which sensor number report is generated.
- Messages can be saved to a file from this window and later viewed using a word processor or text editor for analysis.

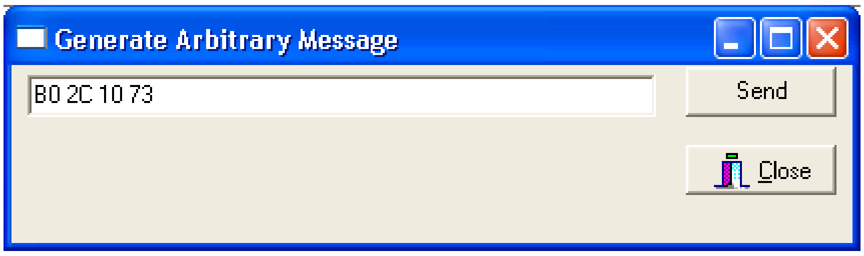
### 6.10.2 Generate SWRQ Window

This window allows a Switch Request message to be generated to change a point on the layout.



### 6.10.3 Generate Arbitrary Message Window

For experienced LocoNet debuggers only!



B0 2C 10 73

This window allows the user to generate an arbitrary message and send it to LocoNet. The message is entered in hexadecimal format including the checksum byte at the end.

# Appendix A.LocoNet Interrogation & Startup Sequence

The SIGM20 participates in the LocoNet® accessory and sensor interrogation sequence which is controlled by some command stations (e.g. DCS100). This sequence is invoked after power is applied to the layout to find out the initial state of all points and track occupancy sensors.

In a Digitrax system controlled by a DCS100, the command station sends accessory commands to special addresses. The accessory decoders respond by reporting their state over LocoNet®. The SIGM20 monitors these messages to discover the initial state of all attached devices. The signal aspect LEDs are then lit accordingly.

After this sequence is complete – i.e. after all point and sensor type devices have reported – the signals begin to report their states. This allows correct linkage between boards: i.e. it allows a signal on one board to set is amber aspect correctly depending on the state of the next signal that could be on another board.

The SIGM20 also reports the reverse running zone directions using “output state” messages for the DCC addresses used to manually set the reverse running zone states. This means that other devices on LocoNet will be able to discover the running direction.

Where the accessory decoders do not have a LocoNet® connection – for example the Digitrax DS44 or decoders from other manufacturers, the initial state can only be discovered after DCC control packets have been sent to the device. If the points are controlled by a DTM30 (or the older DTM16) board, its “initialise” function can be used to drive all points to a known initial state which will then be known by the SIGM20 boards.

The SIGM20 can also be programmed to generate the responses automatically after power is applied. This can be useful in non-Digitrax systems but should not be selected when installed as part of a Digitrax controlled layout.

## IMPORTANT INFORMATION

### DISPOSAL OF WASTE ELECTRICAL & ELECTRONIC EQUIPMENT

European law requires that household waste from electrical and electronic equipment is collected separately from normal domestic waste and then properly recycled.

The *SIGM20* is marked with the crossed out wheeled bin symbol (shown opposite), in accordance with the European directive 2002/96/EC, which means that you must return any damaged or “end-of-life” units to your local authorised collection centre. *GFB Designs* will provide advice and assistance regarding the correct disposal procedures.

### WARRANTY

All *GFB Designs* products are guaranteed against defects in materials and workmanship for a period of 24 months from the date of purchase. Should a defect arise within this period, the product will be repaired or replaced free of charge PROVIDED:-

- The product has been used and maintained in accordance with our instructions and (where applicable) any electrical ratings.
- No accidental physical damage has occurred whilst the product is in the customer’s possession.
- The fault has not been caused by any other form of misuse or neglect.

This guarantee is in addition to, and does not diminish, your statutory rights – for further information please contact your local Trading Standards Authority.

### DECLARATION OF CONFORMITY

The undersigned declares that the *SIGM20* Colour-light Signal Controller - manufactured by *GFB Designs* (contact details below) - is, where applicable, in accordance with the European directives 93/68/CEE (CE marking) and 2014/30/EU (electromagnetic compatibility). Relevant parts of the following standards have been used to demonstrate electromagnetic compatibility:-

BS EN 61000-6-1:2007 & BS EN 61000-6-3:2007

[illegible]

**Fraser Black, Proprietor**

### INTELLECTUAL PROPERTY

“*Sig-naTrak*” is a registered trademark of GFB Designs, Stockport, UK.

The following are registered trademarks of Digitrax Inc., Norcross,GA, USA:-

“LocoNet”, “DCS100”.

Parts of LocoNet® technology, and of other devices which may be connected to a LocoNet® network, are covered by US Patents 6,275,739, 6,367,742, 6,533,224.

“BD4”, “BDL168”, “DB150”, “BDL16”, “BDL162”, “BD1”, “UP3”, “PR3” and “MS100” are Digitrax products which incorporate additional intellectual property and patents owned by Digitrax Inc.

“Microsoft Windows” is a trademark of Microsoft Corporation.

Sig-naTrak®

Model railway electronics by GFB Designs

CE

***This product has been designed and manufactured in the UK.***

***Sig-naTrak*® by GFB Designs**

***www.signatrak.co.uk***

67 Boddens Hill Road, Stockport, SK4 2DG, United Kingdom

T: +44(0)161 883 2022 | F: +44(0)161 883 2077 | E: info@signatrak.co.uk