RAILROAD & CO.™

TrainController™
Gold and Silver

Version 9

Users Guide

July 2018

RAILROAD & CO.™

TrainController™ Gold and Silver

Version 9

Users Guide

July 2018

Contact: Freiwald Software
Kreuzberg 16 B
D-85658 Egmating, Germany
e-mail: contact@freiwald.com
http://www.freiwald.com

Table of Contents

About this Document

RAILROAD & CO. is the leading product line of computer programs for digitally or conventionally controlled model railroads. It contains the following members:

- **TrainController™** is the world's leading software for model railroad computer control.
- **TrainProgrammer™** is the program, which makes programming of DCC decoders as simple as a few clicks with your mouse.
- **+SmartHand™** is the world's premium handheld railroad control system designed for computer controlled model railroads.
- **+4DSound™** is a module, that recreates realistic spatial sound effects for each model railroad layout controlled by **TrainController™** without the need to install onboard sound into each decoder.
- **+Street™** is a module for control of car systems with **TrainController™**.
- **+Net™** is a module, that allows to control your layout with a network of several computers running **TrainController™**.

RAILROAD & CO. TrainController™ Users Guide

An overview of the basic concepts of **TrainController™** is provided in this Users Guide. By reading this document you can obtain information about the many features of the product. Additionally you are provided with the background information necessary for model railroad computer control with **TrainController™**.

The document is divided into three parts. Part I provides a quick start tutorial for users, who are in a hurry and want to start quickly. Part II explains the fundamentals of use. Knowing the contents of this part you will be able to control your turnouts, signals, routes and trains manually and to perform basic automatic operation. Novice users should focus to this part first and put its content into practice before proceeding with Part III. Part III explains the extended features of the software for professional use of all features by advanced users.

Details of usage are mentioned only if they are necessary to understand the related issues or to point to important features of the program. If you want to know in detail, how specific functions are to be used, please refer to the **Help** menu of **TrainController™**.

Some sections or paragraphs are highlighted with additional markings for novice or advanced readers or to indicate important notes. The markings and their meaning are:

B

Basic content. Novice readers should focus on these parts.

X

Extended content for advanced users. Novice readers should initially ignore these sections.

!

Important note.

Help Menu

The help menu installed with **TrainController™** contains detailed reference information necessary for using the program. All menus, dialogs and options are completely described and can be referred to in the case of questions or problems.

! Please note: the User Guide and the help menu are complementary and should be used together. If you want to know, <u>what</u> a certain term means or <u>what</u> a certain function does, please refer to the Users Guide. If you want to know, <u>how</u> a certain object is to be edited or <u>how</u> a specific function is to be executed, call the help menu.

- This User Guide is <u>not</u> a manual for everyday practical use!
- This User Guide should rather serve to describe the basics, technical terms and important relationships of **TrainController™**.
- You should make it a habit to always consult the help menu during daily work. This is especially true for questions, how to do something, and in particular when problems arise. The full-text search usually leads quite quickly to the relevant information.
- Virtually every dialog box in **TrainController™** is equipped with a help button. Before you change a setting in a dialog box for the first time, it should be natural for you to press the help button in this dialog box, in order to clarify what each currently displayed setting means.

The Editions of TrainController™

TrainController™ is offered in three variants:

- **TrainController™** **Bronze** provides a low-cost entry into computer controlled model railroads. It is primarily designed for users with small and medium size layouts and average requirements. Novice users, who do not know **TrainController™**, may consider doing their first steps with **TrainController™** **Bronze**. The reduced functionality of this variant makes it easier to identify and to learn the basic functions of **TrainController™**.
- **TrainController™** **Silver** is the successor of the established and well-known version **TrainController™ 5**. It addresses users with high demands and also users, who are not reluctant to puzzle to accomplish individual goals.
- **TrainController™** **Gold** is the flagship of the **TrainController™** family and in a class of its own. **TrainController™** **Gold** is primarily designed for users with supreme requirements, who want to operate their layout like the real professionals. While **TrainController™** **Silver** is already able to operate even very large layouts, **TrainController™** **Gold** provides much more convenience, efficiency and security for design and operation – especially for larger layouts.

This document provides an overview of the features of **TrainController™** **Silver** and **Gold**. The features of **TrainController™** **Bronze** are described in a separate document.

All text sections, that describe features of **TrainController™** **Gold**, which are not provided by **TrainController™** **Silver**, are marked with a specific marking on the left side of the text in the same way as this section. Contents marked in this way do not apply to **TrainController™** **Silver**. Users of this program version or readers only interested in **TrainController™** **Silver** can safely ignore these contents.

All text sections, that describe characteristics of **TrainController™** **Silver**, which do not apply to **TrainController™** **Gold**, are marked with a specific marking on the left side of the text in the same way as this section. Users of this program version or readers only interested in **TrainController™** **Gold** can safely ignore these contents.

Unless otherwise indicated all screen shots show the user interface of **TrainController™** **Gold**. This means in particular, that user interface options may be shown, which are not available in **TrainController™** **Silver**.

The Differences between TrainController™ Gold and Silver

The following section lists all features, which are unique to **TrainController™** **Gold**:

Misc:

1. Wildcards in object names. See **Help** menu.

2. A descriptive comment can be added to each object. This comment is displayed in the tool tip window, when the mouse is moved to the element. The comment is also included in the printout of the object details.
3. The menu command **Lock Start** prevents **TrainController™** Gold from terminating an emergency stop state and restarting all interrupted processes, when the start button of the digital system is pressed. This option is useful, if a powered digital system is required to resolve a certain emergency situation, and if it is not desired, that **TrainController™** Gold continues its processing, while the emergency situation is not resolved. Setting this option allows to start the digital system, while **TrainController™** Gold remains stopped.
4. The properties of certain objects (e.g. turnouts, routes, blocks, schedules, engines and cars) can be modified outside of edit mode. The prerequisite is that these objects are previously decommissioned (see section 14.11).
5. With the **Multiple Changes** command it is possible to change the name, color or digital address of several objects at the same time. The change of the name of several objects at once is useful when these names use wildcards with similar construction. In such a case the names of a plurality of objects can be changed in a single step.
   Furthermore, it is also possible to modify the digital addresses of multiple objects at one time. Possible changes include the assignment to another digital system or the block-wise shift of digital addresses.
   To edit multiple objects at once, it is typically most useful to select them in the Explorer window.
6. The **Multiple Changes** command supports the **Memory** tab of indicators, flagman objects and markers. In this way the memory of multiple indicators can be changed in one single step. Multiple markers, in particular, can be set to be turned off by their reference indicator in one single step in this context.
7. The current positions of the trains in the blocks as well as the status of train sets can be stored in a separate file and loaded from there. This is also possible if the project data is not stored at the same time. So it is e.g. no longer necessary, to revert manually the positions of all trains to the current state of the layout after a test with the simulator without connection to the layout (see page 228).
8. The status of the user interface can be optionally stored in a separate file, which allows to edit the project file on different computers with an individual status of the user interface for each of these computers (see page 87).

Switchboard:

9. Switchboard symbols can be displayed in five different sizes ranging from 12x12 to 28x28 pixels per symbol / switchboard cell.
10. Additional track symbols: space saving turnout elements and adequate connecting tracks and crossing symbols. These track symbols do not only allow space saving

arrangement of turnouts, but also reproduction of certain prototypical control panel layouts. See page 100.

11. The name of the associated block can be displayed in block diagrams or the switchboard, too, when edit mode is turned off.
12. It is possible, to override the default colors for the background and frame of text elements by individual settings for each particular text element.
13. Track elements in switchboard windows can be colorized individually.
14. It is possible to insert an empty new line or column at any position into the switchboard with one menu command. The lines and columns right/below the inserted line/column are shifted accordingly. This action can be undone, too. In a similar way it is possible to delete a complete line or column from the switchboard.
15. It is possible to create custom switchboard symbols for signals, push buttons, on/off switches, toggle switches, routes, feedback indicators, flagmen and virtual contacts with an integrated bitmap editor and to assign such custom symbols individually to each according switchboard object. Custom switchboard symbols can be transferred between different data files by export and import.
16. It is possible to create self-provided, inoperable switchboard symbols with an integrated bitmap editor. See page 108.
17. The second digital address of switchboard objects with more than two states (e.g. three-way turnouts or four-aspect signals) can be specified independently from the first address.
18. Mini block symbols can be used to represent blocks in diagonal track sections.
19. Individual blocks can be hidden in the switchboard, when edit mode is turned off.
20. The display of block signals in blocks can be turned on and off individually, too, i.e. on a per-block base.
21. A special switchboard symbol supports the frog polarization of crossings. These crossings may have no tongues. For these crossings, however, the polarity of the frog must be switched according to the set route. For this purpose, there are specific symbols for polarized crossings, which are operated similar to turnouts.
22. With the operations of a route it is possible to set turnouts, signals and other objects to a protective position. As long as the route is active, the objects cannot change their position. But they can be used in other routes, provided that these routes use the object in the corresponding position (see page 288).
23. With self-made extensions arbitrary accessories can be operated in the switchboard and integrated into automatic operations. Examples of such equipment are cranes, machinery and other working models, signals with more than four terms, accessories, which are controlled by more than two turnout addresses and / or locomotive addresses; Selectrix products, which are controlled by several addresses or where more than one bit of a bus address must be changed simultaneously (see the section 14.13).

24. For turnout position control of turnouts with two drives (e.g. double slip switches or three-way turnouts) it is possible to specify two feedback addresses for each state of the turnout (see section 14.12, “ Turnout Position Control”).
25. Route operation by right mouse button clicks to track symbols: clicking with the right mouse button to a track diagram symbol in the switchboard opens a context menu, which contains – among others – a menu of routes, which pass this track symbol. By selecting a route from this menu it is possible to toggle the state of this route.
26. Crossover symbols (see page 101).
27. Text elements can display HTML-based content for text or graphic formatting.

Train Control:

28. **TrainController™** **Gold** supports the operation of train functions controlled by additional function-only decoders without the need to create an artificial multiple-unit as required in **TrainController™** **Silver**.
29. It is possible to specify an individual maintenance interval for each engine or car and an optional operation, that is automatically executed, when the maintenance interval expires. See page 273.
30. For convenient programming of locomotive decoders, a switching command can be specified. With this command and an appropriate relay a selected section of track on the layout can automatically by connected to the programming track output of the central unit whenever functions for the programming track are invoked from **TrainController™** (see page 114).
31. When setting up locomotives with a DCC decoder, the digital address of the locomotive can be read directly from the programming track into the record of the locomotive. Conversely, the digital address stored in the program can be programmed into the decoder (see page 114).
32. With a system operation, locomotive functions (e.g. light) on all vehicles can be turned on or off at the same time (see page 290).
33. With a system operation, all running locomotives can be stopped gently with an adjustable delay (see page 290).
34. Train operations (such as switching light or coupling certain locomotives) can be triggered by buttons or switches in the switchboard (see page 293).
35. With a train operation, trains can be run from their current location automatically to any destination block via AutoTrain (see page 293).
36. With a train operation, an arbitrary schedule can be started with the concerning train (e.g. as engine function from the train window) (see page 293).
37. Engine functions can be directly operated with menu commands in the main window or in the context menu of each vehicle after pointing to that vehicle on the screen. It is not necessary to activate a train window first.

38. It is possible to assign two lists of functions to each engine function, one list for each state of the function (see page 131).
39. The train operation to execute an auxiliary function provides options to specify the vehicle, when applied to a train set. It is possible to select the first, last or all vehicles in a train set, to apply function forwarding as in previous versions or to select the vehicle by marking a position in a train description (see page 269).
40. The train operation **display name** allows to set a variable train name for display on the computer screen, that depends on the currently executed schedule, for example, or other operational aspects (see page 293ff).
41. The train operation **short distance move** allows to move trains by a short distance to a certain direction – also if the train is being controlled by a schedule (see page 293ff).
42. The Engine Function Library (see section 3.6, “Headlights, Steam and Whistle”) provides an option, which causes a function, which is turned on or off in the train window for a vehicle in a train set, to be applied to all vehicles in this train set. In this way it is for example possible to toggle the interior lights of all vehicles in a train set with one click to a function button in the train window.
43. The measurement of the speed profile can be performed by using almost arbitrary third party speed measurement facilities (see page 129). The measurement is by default performed in a semi-automatic guided procedure. But it can also be done in a full automatic mode, if supported by the speed measurement device.

Train Management:

44. Vehicle groups can be optionally defined to exclude all vehicles listed therein. Vehicles are contained in such vehicle group, if they are not listed in this group. See page 257.
45. Powerful train management. It is possible to define cars and to arrange train sets (multiple units, consists) at any time during operation. See section 11.2, “Cars and Train Sets”.
46. Vehicles can be joined to train sets automatically by means of Operations. Train sets can be separated automatically by Operations, too. See page 251.
47. A new schedule rule allows trains to enter reserved destination blocks to join vehicles, that are already located there, to form a new train set. See page 249.
48. Multiple units can be operated with the throttle of the digital system. Multiple units can be created and dissolved with minimum human intervention. Just turning the throttle knob for one participating engine is already sufficient. See page 250.
49. It is possible to specify an individual length for each specific car. This is taken into account for the calculation of the total length of each train set. By adding or removing such cars from/to train sets during operation, the total length of each train can changes automatically, which again is taken into account for stops at the middle of platforms or the extended train guidance based on train length. See page 344.

50. For realistic simulation of train tonnage it is possible to specify the full weight and the empty weight for each car. Cars can be loaded and unloaded manually or automatically at any time during operation. The currently selected weight (load condition) of each car is applied to the calculation of the maximum speed or acceleration momentum of affected train sets. See page 248.
51. It is possible to specify individual contact spots for each specific car and both directions. This is automatically taken into account for proper calculation of braking ramps and distant markers, when a train set is currently being pushed.
52. Forwarding of functions can be turned on or off for each train set at any time during operation. See page 248.
53. Train operations allow to start a spontaneous run with the current train or to terminate the current schedule of a train. These operations can be called automatically by contact indicators, by brake, stop, speed or action markers or by macros.
54. Temporary speed limits can be set with a new train operation. See page 198.
55. The management of vehicle groups can be accessed directly from the main menu.
56. Predefined vehicle groups (e.g. for steam locomotives, diesel locomotives, passenger cars, all locomotives, etc.) support the effective creation and maintenance of extensive vehicle groups and the association with other objects such as blocks, routes, schedules, turntable tracks, COMBI-groups, etc. (see Section 11.3).
57. Individual vehicles or vehicle groups can be excluded from vehicle groups. Using the predefined vehicle groups customized groups can be arranged very quickly, such as the group of all diesel locomotives without rail buses.
58. With the assignment of trains to other objects, the minimum and maximum length of trains can now be specified (see section 11.3). This allows, for example, to arrange specific schedules for trains with a certain length.
59. Also, with the assignment of trains to other objects, the minimum and maximum weight of trains can be specified (see section 11.3). Thus, for example the speed of heavy trains on a slope can be reduced dependent on the current weight of the train, even if the weight of trains is changed during operation train by loading and unloading. Or it is possible to direct loaded trains to other tracks than unloaded trains.
60. Furthermore, with the assignment of trains to other objects, the speed of trains can be specified (see section 11.3). Thus, for example it is possible to evaluate in triggers or conditions, whether a train is moving or how fast it goes.
61. The assignment of trains to other objects can be made dependent on a condition (see section 11.3). This allows for example to lock or release certain blocks temporarily for certain trains with a switch in the switchboard.
62. With the assignment of trains to other objects, the arrangement and orientation of each vehicle in a train set can be specified (see section 11.3).
63. The **Speed Profile** or **Advanced Fine Tuning** dialog box can be accessed directly from the main menu bar.
64. The **Function Library** can be accessed directly from the main menu bar.
65. Clipboard support (copy, cut & paste) for the **Trains** tab.

Dispatcher and Automatic Operation:

66. All lists of objects (blocks, routes, schedules, etc.) in the dispatcher window now provide an optional tree mode. In this mode the objects are not displayed in a plain list, but structured in a tree view. The folders in this tree are inherited from the folders in the explorer window. These folders can be created or deleted in the dispatcher window and will also appear in the explorer window, or vice versa.
67. The name of the blocks can be displayed, too, when edit mode is turned off.
68. It is possible to open more than one dispatcher windows at the same time. This is useful if you want to monitor different block diagrams simultaneously.
69. The **Passenger Ride** command of the **View** menu causes the dispatcher window to follow the selected train as it moves across the layout. The block, in which the train is located at a time, will be automatically highlighted and displayed in the dispatcher window. If the train moves to another block diagram, then the display changes to this diagram, too.
70. With a specific option it is possible to exclude those routes from the calculation of block diagrams, that contain too much turnouts. If two routes between the same two blocks contain a different number of turnouts and this difference exceeds a certain preset value, then the route with the higher number of turnouts is ignored. By using this option the calculated block diagrams will contain only routes with a minimum or accordingly higher number of turnouts.
71. Speed markers provide more control of the location, where speed limits of the subsequent block are applied. See page 166
72. Action markers allow to trigger operations easily at any location in a block without affecting the speed of the passing train. See page 167.
73. The effect of all brake, stop, speed and action markers can be limited to specific trains. In this way it is very simple to let passenger and freight trains stop at different positions. Many other useful applications can be easily achieved, too. See page 179.
74. The effect of all brake, stop, speed and action markers can be limited to specific schedules. In this way it is very simple to let the same train stop at different positions depending on the currently executed schedule. Many other useful applications can be easily accomplished, too. See page 179.
75. It is possible to specify differing brake and stop markers for scheduled stops and for unscheduled stops in the same block. This can be used, for example, to let the same train perform scheduled stops in the middle of a platform and perform unscheduled stops near the block signal at the end of the block, for example. See page 180.
76. It is possible to assign indicators to turnouts for occupancy indication of all routes, that are using the concerning turnouts. See page 344.
77. It is possible to specify a separate set of schedule rules for **AutoTrain** runs. These schedule rules work in the same way as schedule rules for regular schedules. They

can be changed outside edit mode, however, and each change affects all **AutoTrain** runs, which are initiated later.

78. Specific rules for **AutoTrain** can prevent blocks or routes, that are occupied, reserved by other trains, locked to the according direction of travel or locked by an unfulfilled condition, from being included in the path search.
79. **AutoTrain** can be called by operations of other objects. It is in particular possible to trigger **AutoTrain** by start and destination keys; even from external control panels. This allows for automatic train runs on point to point connections without the need to create schedules in advance. See page 206.
80. Individual schedule rules can be applied to all schedules on request.
81. Specific schedule rules prevent schedules from reserving occupied routes and blocks without the need to specify extra conditions for the blocks or routes. In other versions of **TrainController™** it is only possible to prevent trains from running into such blocks or routes. To prevent reservation, too, it is necessary there to specify appropriate conditions.
82. An additional *smart* mode can be optionally applied as a rule to the release of routes in schedules: in this mode passed routes with own occupancy indication are released, when they are no longer reported as occupied. Routes without own occupancy indication are released, when the train reaches a stop marker of a subsequent block. In this way the smart mode automatically selects the release policy, which is optimal for the particular route.
83. Routes, that were already activated prior to reservation by a schedule, are optionally deactivated automatically upon termination of the schedule, if desired. In other versions of **TrainController™** such routes remain always activated and must be turned off explicitly. This is controlled by a new schedule rule.
84. A schedule rule keeps routes and blocks, that could not be released during the normal run of the schedule, reserved upon termination of the schedule, if desired. These blocks or routes are automatically released later, when this is possible. In other versions of **TrainController™** all routes and blocks requested by a schedule and different from the current block of the train are always released upon termination of the schedule.
85. A optional schedule rule causes schedules to select always that route among several routes between the same two blocks, that contains the smallest number of turnouts. This rule is activated by default for new schedules to prevent trains from passing undesired crossovers.
86. With a specific schedule rule it is possible to specify a schedule watchdog. This is the maximum time period between activation of two indicators. If no indicator is triggered within the specified period of time and the train is set to run at non zero speed, then it is assumed, that the train got stuck. In such cases appropriate error information is displayed on the screen. See page 347 .

87. A specific schedule rule provides limited aberration protection. If a train running under control of such schedule is detected in an unexpected block, then appropriate measures are automatically taken by the software. See page 347 .
88. With a specific schedule rule it is possible to specify, that always that path is selected, that contains routes or blocks, which have been visited by the train under control of this schedule the longest time ago (“oldest” block or routes). This option can be used to implement systematic track cleaning trains. See page 348.
89. It is possible to specify a start delay for each schedule, which is applied at the beginning of each schedule and after each stop of a train in a schedule. This delay specifies the time span, which will take place between clearance of a track section ahead of the train and before the train is set in motion. This time span simulates the response time of the engineer.
90. In addition to the global start delay described above, which applies to all stops, scheduled and unscheduled, in equal measure, it is also possible to specify an individual delay for each scheduled stop. Such delay is applied after a scheduled stop and execution of the associated operations and before the train is set in motion. This time span can be utilized to perform additional operations (e.g. playing an announcement, the noise of closing doors or the whistle of the conductor) after a scheduled stop ended and before the train is set in motion. See page199.
91. It is possible to prevent certain schedules (e.g. schedules solely used as successors of other schedules) from being listed in the dispatcher window, when edit mode is turned off. See page 215.
92. Blocks can be defined to be permanently unidirectional. Such blocks can only be passed in a certain direction of travel. Unlike temporary entry locks, which cause a similar effect, this setting is permanently valid and can only be changed in edit mode. See page 156.
93. Blocks, routes, schedules, trains, turnouts and other objects can be decommissioned and excluded from operation at any time during operation. See page 311.
94. Train guidance based on train length: Each train can be prevented from going to destination blocks, that are shorter than the train. See page 344.
95. Train guidance based on train length: Each train can be prevented from stopping in blocks, that are shorter than the train. See page 344.
96. Train guidance based on train length: Each train can be prompted to prefer the shortest destination block, which is long enough to store the train. See page 344.
97. A schedule rule prevents blocks and routes from being released during a running schedule, if the train length indicates, that the train does not fit completely into subsequent blocks.
98. Extended Train Guidance System: Each train can be forced to start a schedule in a certain direction, i.e. forward vs. backward or pulling vs. pushing, respectively. It is also possible to specify, that trains can only be started, if they maintain their current direction of travel. See page 345.

99. Extended options for the selection of trains for schedule successors and schedule selections: schedule successors or schedule selections can be started with specific trains. The option to start a schedule successor with the oldest train can also be applied, if a train change is desired. See page 207.
100. In cases, where control of running trains is passed from a schedule to a successor schedule without stopping the train, a specific schedule option causes allocation of blocks and routes of the successor schedule already, when the train enters the second last block of this schedule. Usually and in other versions of **TrainController™** this allocation is not performed, before the train enters the destination block of the schedule. This option allows for a more fluent change of control between schedules and improved calculation of block signals during this change.
101. Schedule sequences allow sequencing of single schedules, which is usually more flexible then the static chaining of schedules as successors. See page 207.
102. Each schedule can be optionally started with the oldest train.
103. The command **Restart most recent Schedule** allows to restart the schedule, that has been most recently executed by a particular train. This command is for example useful to continue a schedule, that must have been prematurely terminated for certain reasons.
104. The aspect of calculated block signals cannot only be selected for each block or route in each particular schedule, but also preselected in each block, route or turnout once for all schedules. Among others this allows to accomplish the “yellow” signal aspect also for trains run by **AutoTrain™**. See page 196.
105. Speed limits, which depend on calculated block signals, cannot only be preset on the level of blocks for all schedules, but also on the level of routes or turnouts once for all schedules. Among others this allows to accomplish speed limits also for trains run by **AutoTrain™**. Furthermore it is possible to lower the speed limits preset by blocks, routes or turnouts by individual settings for each block or route in a schedule. See page 196.
106. A specific schedule rule can cause trains to reduce their speed to a preset value, when the calculated distant block signal is red due to an unscheduled stop. This causes trains to reduce their speed already in the block before such unscheduled stop applies and can help to improve traffic flow.
107. Turnout position control. See page 312.
108. Conditions and triggers may now contain additional logical groups, that are true, if at least, at most or exactly a certain preset number of items contained in the group have the required state. See page 285.
109. Combined groups can be used in conditions and triggers to check, whether certain trains are located in certain blocks and/or whether these trains are performing certain schedules. They can also be used to check, whether certain blocks are currently involved in certain schedules. See page 286.

110. **Lock all Blocks** is a command, that can be used to interrupt the operation of your layout without causing trains to perform an abrupt stop. See page 211.
111. **Lock all Schedules** is a command, that can be used to terminate the operation of your layout without causing trains to perform an abrupt stop. See page 211.
112. For AutoTrain by Drag & Drop and each section (block or route) it is possible to specify the request of the yellow signal, speed limits, actions and conditions (see page 199 and 340). If, for example, an on/off switch is added to the schedule specific settings of a block for AutoTrain, then the block can be locked or released for AutoTrain at any time during operation.
113. For spontaneous runs and each section (block or route) it is possible to specify the request of the yellow signal, speed limits, actions and conditions (see page 199 and 340). If, for example, a wait time is specified in the schedule specific settings of a block for spontaneous runs, then all trains under control of spontaneous runs will stop here.
114. Beside displaying a message in the message window with a system operation it is an alternative to display a message as a temporarily well visible tool tip on the computer screen (see page 290).
115. With specific system operations, it is possible to automatically select an object at the user interface and/or to execute any menu command (see page 290).
116. With a specific type of operations, the control flow can be controlled within the processing of operations. For example, the execution may be made dependent on preconditions. In addition, also jumps and loops within operations are possible (see page 291).
117. During the execution of operations it is possible to insert delay operations with random duration (see page 291).
118. The execution of operations can also be made dependent on probabilities (see page 291).
119. Optionally, operations can be performed in random order (see page 291).
120. With counters it is possible to count the number of certain processes or events and to evaluate such numbers in conditions or triggers of other objects (see page 302).
121. Concatenation of AutoTrain runs: if an AutoTrain run starts in a block in which no train is located and this block is currently destination of an active AutoTrain run, then the new run be linked as a successor to the already active run. Since the direction of travel can be changed during the transition to successor runs, it is now possible to arrange zigzag runs with AutoTrain.
122. Rules for spontaneous runs can be applied to all trains in one step.
123. A specific schedule rule allows for the release of blocks, already, when a train has completely entered a subsequent block and before it reaches a stop marker in that block. Thus, even without the use of conductive axles on the rear end and additional occupancy sensors in the turnouts, speed limits can be lifted earlier and a higher succession of trains can be accomplished, thus resulting in more prototypical operations (see page 348).

124. With a schedule rule for train length control, entry into blocks that are too short for the train to fit into, can be prevented (see page 348).
125. With a schedule rule, scheduled waits can be limited to those trains which fit in the block (see page 348).
126. With a schedule rule, the destination block of the schedule must already be reserved at the start of the train (see page 348).
127. With a schedule rule, the entire path to the destination block of the schedule must be already reserved at the start of the schedule. All blocks on the path to the destination are treated like ‚critical' blocks (see page 348).
128. With another schedule rule, it can be specified that in any case, only the shortest appropriate destination block is used, even if there are longer destination blocks, which can be reached easier or faster (see page 348). This allows, for example, the optimum use of hidden yard tracks, even if the tracks are not located side by side.
129. With a schedule rule, the evaluation of distances to the destination block or the next obstacle can be turned off for calculation of the optimal path (see page 348).
130. With four other schedule rules, the inclusion of sections of track that are currently occupied, used by other trains, locked, or for which a condition is not met, can be disabled. Such sections are then treated as if they were not contained in the schedule at all (see page 348).
131. With further schedule rules for coupling to vehicles, waiting in the destination block of the schedule, it can be specified, that there must be at least one vehicle in the destination block or that only cars are waiting there but no engine (see page 348).
132. With another schedule rule for the coupling to vehicles, which are waiting in the destination block of the schedule, it can be specified, whether the entering train is joined to the waiting vehicles or whether it remains separated from them (see page 348).
133. With another schedule rule, it can be specified that the end of trains must not stop in critical blocks. A long train must proceed behind a critical section until it completely fits into the blocks that lie beyond the critical section (see page 348).
134. Another schedule rule allows multiple trains running in the same direction of travel, to share the same critical section (see page 348).
135. Train tracking of manually driven trains is able, on demand, to track trains, which reverse in a turnout area (‘zigzag’ movement) (see page 159).
136. Distances and ramps of markers can be specified as formulas. This allows, for example, to vary stop locations in a block, that depend on the length of vehicles already waiting therein. It is also possible to position on the gap between any vehicle of the train (see page 174).

137. With a system operation, all blocks can be locked or unlocked at the same time. (see page 290).
138. With a system operation, all schedules can be locked or unlocked at the same time (see page 290).
139. With buttons or switches in the switchboard, schedules can be started with a certain engine or they can be started from a given starting block (see page 293).
140. For each block in a COMBI-group it is possible to specify, whether this block must be a current block of a train (as in version 7), or whether it is sufficient if the block is reserved, but not the current block (see page 286).
141. In addition to blocks, it is also possible to add routes to COMBI groups (see page 286).
142. Spontaneous runs can be started as operations of blocks. In this way, for example, they can also be started with a push button in the switchboard.
143. For each block in a schedule, it is possible to specify a scheduled stop only for specific trains. This also applies to AutoTrain or spontaneous runs (see page 199).
144. Since the properties of AutoTrain can be edited like regular schedules (see feature 146) it is for example possible to highlight routes used for regular schedules, AutoTrain runs and spontaneous runs in different colors.
145. Occupancy highlighting of blocks can optionally reflect the position and extent of the currently active indicators (according to the position and extent of each indicator in the block editor).
146. AutoTrain can be edited like regular schedules via its own block diagram. It is possible to remove blocks or routes from this diagram, which excludes them from being used in AutoTrain runs. It is furthermore possible to edit the properties of AutoTrain with almost the same range of options (such as start and destination operations, driving mode, rules, permitted trains, condition, etc.) like regular schedules.
147. Spontaneous runs have an own block diagram, which can be edited like the block diagram of other schedules. It is possible to remove blocks or routes from this diagram, which excludes them from being used in Spontaneous runs. In this way it is easily possible to limit spontaneous runs to certain areas of the model railroad layout.
148. Many program settings can be made much more flexible with variables (see section 14.14, “Variables”).
149. System events and states can be evaluated in triggers or conditions (see page 287).
150. The menu command **Restart all Schedules** restarts all schedules, that were active, when the **Terminate All Schedules** menu command, the global **Power Off** command or the edit mode was most recently called.
151. The prerequisite control flow operation allows to evaluate complex conditions (see page 291).
152. The fact, whether an AutoTrain or spontaneous run is active can be evaluated in triggers or conditions of other objects.

153. Recording of blocks and routes with the recorder (e.g. during creation of operation list, triggers or conditions) is also possible via the block diagram and the diagram of schedules.
154. The **Multiple Changes** command supports the **Rules** tab of schedules. In this way several or all rules of multiple schedules can be changed in one single step.
155. The schedule rule **Maximum Detour** (see page 350) .
156. The schedule rule **Use start block as destination block** allows dynamic specification of home blocks for circular schedules (see page 350).
157. The schedule rule **Include turntables** allows turntables to be limited to AutoTrain runs from or to adjacent blocks (e.g. in the attached roundhouse – see page 350).
158. Additional rules allow to affect the calculation of internal block signals (see page 357section).
159. Line-up and automatic move up of multiple trains in a single block with one single sensor.
160. Stations (see section 15.7, "Stations")
161. Specific calculation of internal block signals for shunting moves (see page 370, "Local Schedules and calculated Signals")
162. Booster and Booster Management (see section 15.8, "Booster").
163. The schedule tab for markers and virtual contacts has been extended. It is now possible to limit the validity of markers, for example, not only to certain schedules, but also optionally to trains under control of schedules or not, to AutoTrain runs, Spontaneous runs, local or non local schedules (see page 369, "Local Schedules") or to schedules, which control trains with specific driving modes.
164. The state **reserved by a local schedule** of a block can be evaluated in triggers and conditions. This provides the possibility to distinguish, whether a block is reserved by a local schedule (see page 369, "Local Schedules") or not. This option is for example useful to establish specific signaling or other logic for shunting moves.
165. The internally calculated aspect of distant signals can be evaluated in trigger and conditions (see page 304).
166. Extended accessories can be used to issue complex or non-standard sequences of digital locomotive commands by the functions of an engine or car (see page 320).

Timetable / Clock:

167. The time, date and other settings of the clock/timetable can be altered outside of edit mode, too.
168. It is possible to specify a reset time, which is applied during reset of the complete layout and optionally after begin of each session.
169. The display of the clock can be synchronized with the system clock of the computer.
170. The clock can be automatically started and stopped with system operations called by buttons, macros or even indicators.

171. In edit mode it is possible to limit the display of the timetable to those entries, that are executed at the currently selected date. If this is done, then it is additionally possible to expand the display to show the same content as outside edit mode; i.e. to display the complete operational timetable for the currently selected date.
172. With a system operation the clock can be set to a specific time (see page 290).

Turntable:

173. With a turntable switchboard symbol each turntable or transfer table can be operated and controlled via switchboard windows, too.
174. The automatic calculation of the block diagram covers turntable symbols in switchboard windows, too. Routes across and involving turntables or transfer tables are automatically calculated, too. No specific programming or data input necessary to enable a turntable for **AutoTrain** or automatic train operation.
175. Each turntable track can be optionally marked as forward or backward. This causes the specified locomotives to leave the bridge via the according tracks in forward or backward direction. See page 391.
176. In addition to the above it is possible to override the direction, in which a locomotive leaves the turntable, on an individual per schedule base. See page 391.
177. Segment turntables (see section 17.7).
178. Support of the new NOCH segment turntable.

Traffic Control:

179. It is possible to open more than one traffic control windows at the same time.
180. The traffic control can be pinned to a certain train, to a certain block or to a certain window. This allows several useful applications. See page 216.

Message Window:

181. It is possible to suppress repeated display of undesired Dr.Railroad messages.
182. The content of the message window can be sorted by column. This is in particular useful to sort the logged messages according to the trains, which these messages belong to.

Hardware and Digital Systems:

183. Selectrix systems and derivatives: push button and on/off switch symbols can be arranged to manipulate several bits of the same Selectrix address simultaneously. This function is useful to operate specific Selectrix compatible decoders, that require manipulation of more than one bit of the same address in one step.

Part I

Quick Start

B

Quick Start - Step 1:
Installation and Program Start

You have obtained **TrainController™** to control your model railroad with your computer. It is easily understood, if you are eager to control your layout with your computer as soon as possible. If you are in a hurry about starting without reading the complete Users Guide first, you can also reconstruct the following quick start tutorial about **TrainController™**.

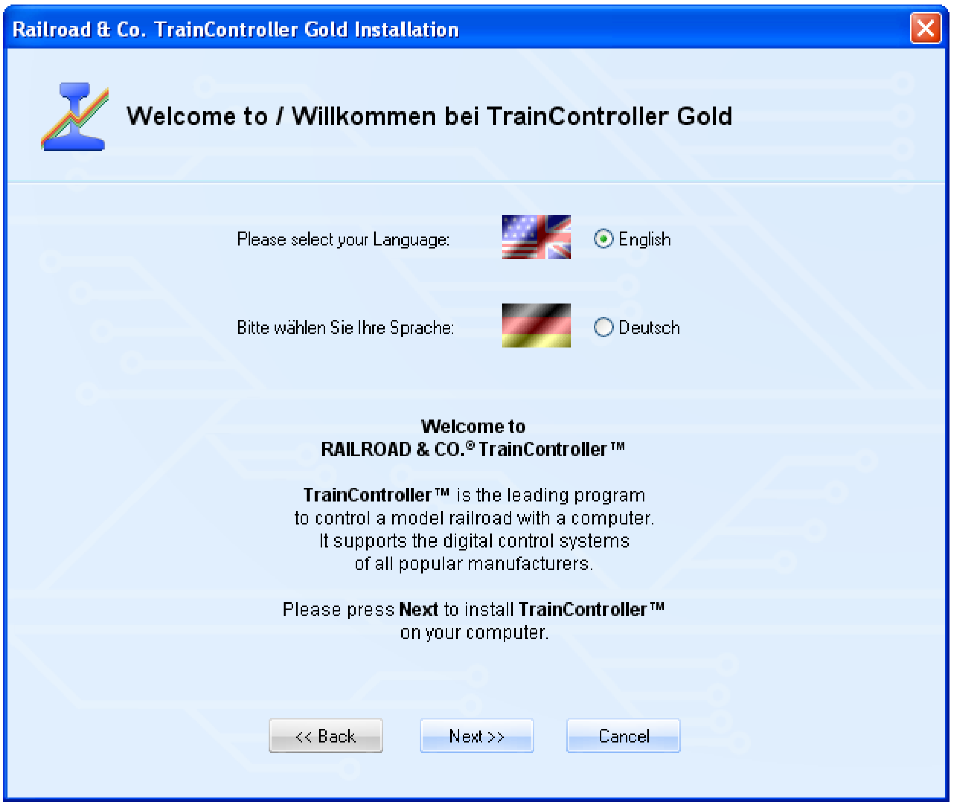
Detailed explanations about the fundamental concepts of **TrainController™** can be found in Part II of this document. It is strongly recommended that you study the contents of Part II prior to working seriously with **TrainController™**.

Now let us start:

Installation

The installation file of **TrainController™**, its name is SETUP.EXE for **TrainController™** **Gold** and TCSSETUP.EXE for **TrainController™** **Silver**, can be downloaded from the download area of the Internet home page of the software (www.freiwald.com).

After starting SETUP.EXE or TCSSETUP.EXE, respectively, a self-explaining window is displayed, that guides you through the steps, that are necessary to install **TrainController™** on your computer.


Railroad & Co. TrainController Gold Installation
Welcome to / Willkommen bei TrainController Gold

Please select your Language: [✓ English]

Bitte wählen Sie Ihre Sprache: Deutsch

Welcome to
RAILROAD & CO.® TrainController™

TrainController™ is the leading program to control a model railroad with a computer. It supports the digital control systems of all popular manufacturers.

Please press **Next** to install **TrainController™** on your computer.

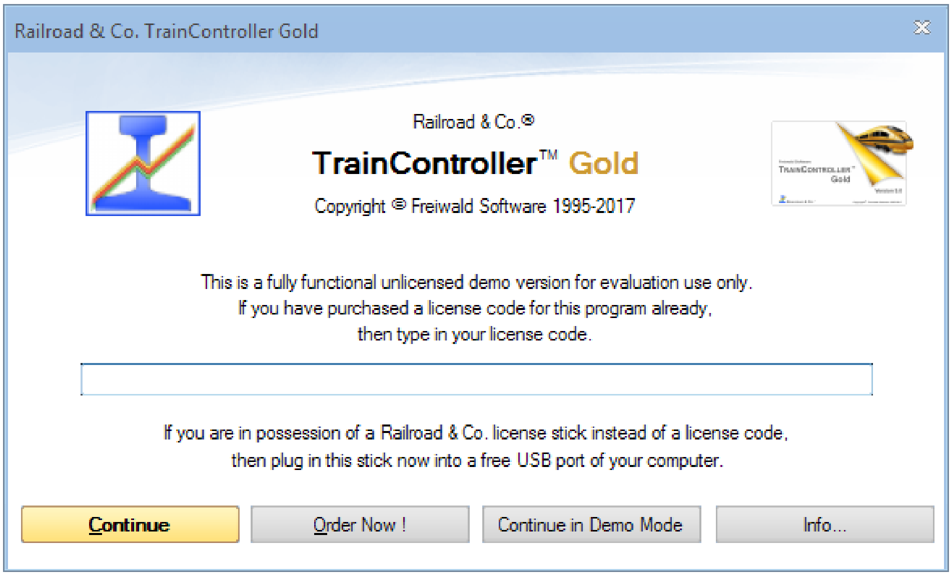
Diagram 1: TrainController™ Setup Screen

Ensure, that you select the right language, because the selected language will also appear later, when running **TrainController**™.

Before you start **TrainController**™ you should connect your digital system which you are using to control your model railroad, to the computer. Please refer to the instructions provided by the manufacturer of your digital system, to see how this is done.

Program Start

After correct installation of **TrainController**™ there should be an entry in the **Start** menu of your Windows system, which you can use to start the software.


Railroad & Co. TrainController Gold

Railroad & Co.®

TrainController™ Gold

Copyright © Freiwald Software 1995-2017

This is a fully functional unlicensed demo version for evaluation use only.
If you have purchased a license code for this program already,
then type in your license code.

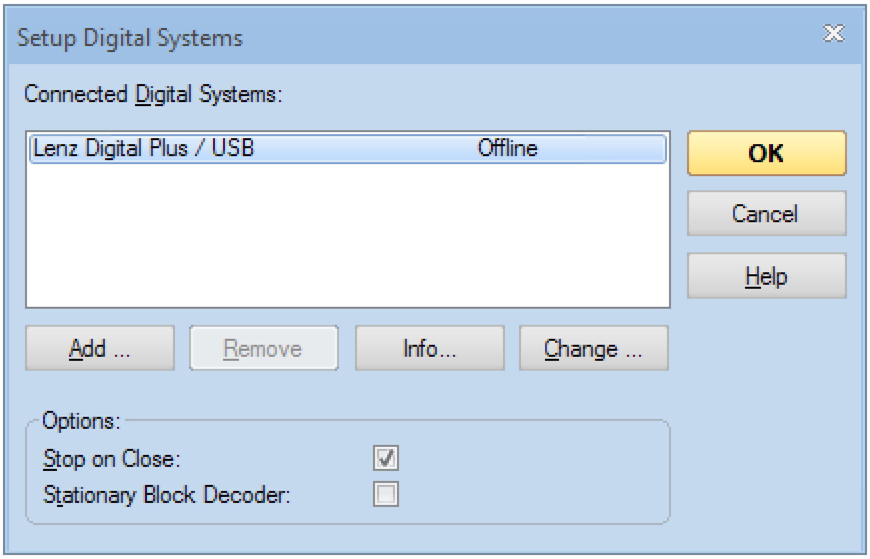
If you are in possession of a Railroad & Co. license stick instead of a license code,
then plug in this stick now into a free USB port of your computer.

Diagram 2: License Inquiry

When the program starts the software first asks for your license key. Do not be concerned, if you are not yet in possession of such key. Press **Continue in Demo Mode**, if you want to try the software before buying it.

If you are already in possession of a license, then enter the license key here or plug in the Railroad & Co. license stick into a free USB port of your computer and **press Continue**.

In the next step the connected digital system is configured. Usually the following screen appears automatically, when the program is started for the first time. If the program starts without displaying the screen shown below, then call the **Setup Digital Systems** command of the **Railroad** tab.


Setup Digital Systems

Connected Digital Systems:

Lenz Digital Plus / USB   Offline

Options:

Stop on Close: ✓

Stationary Block Decoder:

Diagram 3: Setup Digital Systems dialog

If your digital system and/or the serial or USB port of your computer, to which your digital system is connected, is not displayed correctly, press **Change** to select the right settings.

In order to test, whether the connection to the digital system is properly established, play around a little bit with the **Power Off** and **Power On** command of the **Railroad** tab. These commands stop or start your digital system, respectively. Your digital system should respond accordingly to these commands. If your digital system does not respond or if there are some error messages displayed, then do not proceed any further, until this problem is resolved. In case of problems in this area, check very thoroughly, that the digital system is properly connected to the computer according to the manufacturer's instructions.

If the steps outlined above have been performed correctly, you are ready to take the first steps into model railroad computer control.

Quick Start - Step 2:
Controlling a Train

Preparing a Train for Model Railroad Computer Control

First put a train onto the tracks of your layout and run it with your digital system. This step is recommended to verify, that the digital system and the train are running correctly and also to bring the digital address of the train back to your mind. This is needed a few moments later.

Now ensure, that the **Edit Mode** option in the **View** tab is active.

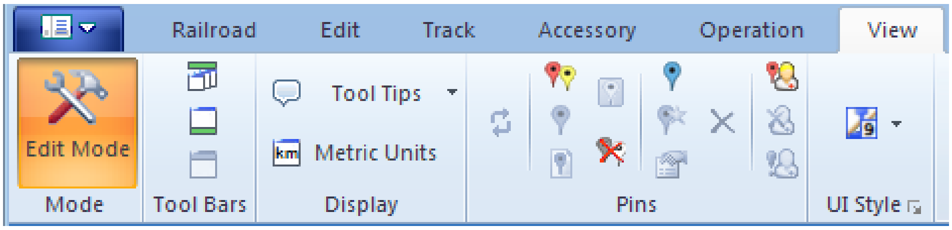


Diagram 4: View Tab

In this mode it is possible to enter new data into the software or to change existing data. This is what we want to do next.

Call the **New Train Window** command of the **Window** tab. If this is done correctly, the following window will appear on your computer screen:

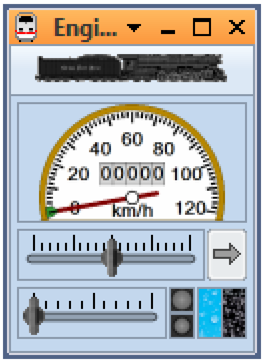


Diagram 5: Train Window

If you want to learn more about the various controls of this window, please refer to chapter 3, “Train Control”.

Now select the **Properties** in the **Edit** tab.

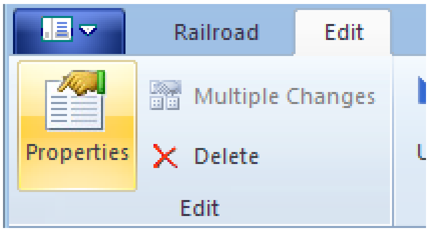
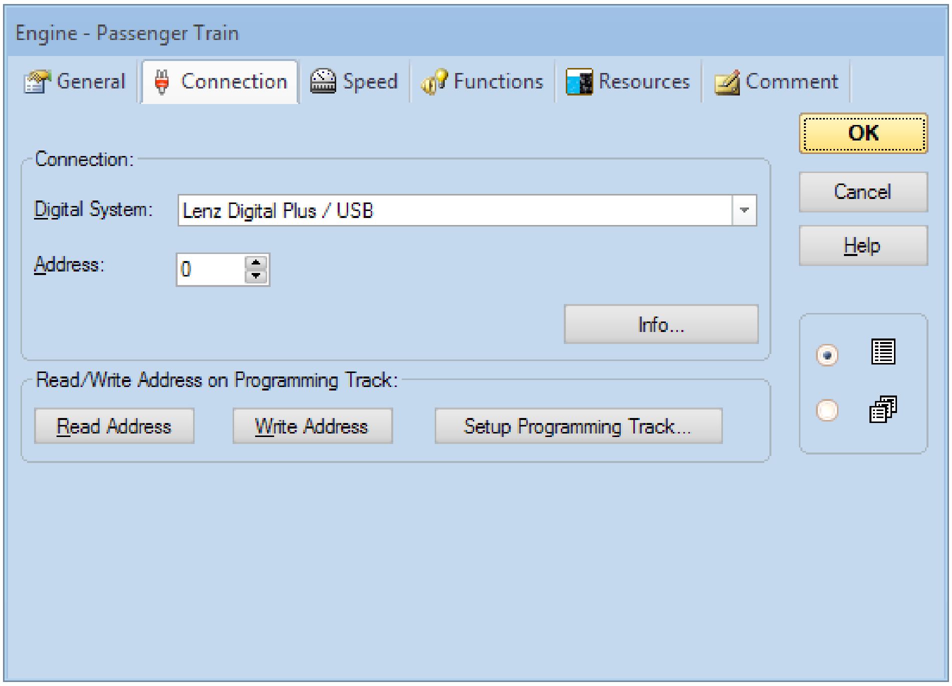


Diagram 6: Edit Tab and Properties Command

This is one of the most important commands of **TrainController**™. It is used for all objects contained in the software (trains, turnouts, signals, routes, etc.), whenever you want to change the settings of a particular object. The following window is displayed now:



Engine - Passenger Train

Connection:

Digital System: Lenz Digital Plus / USB

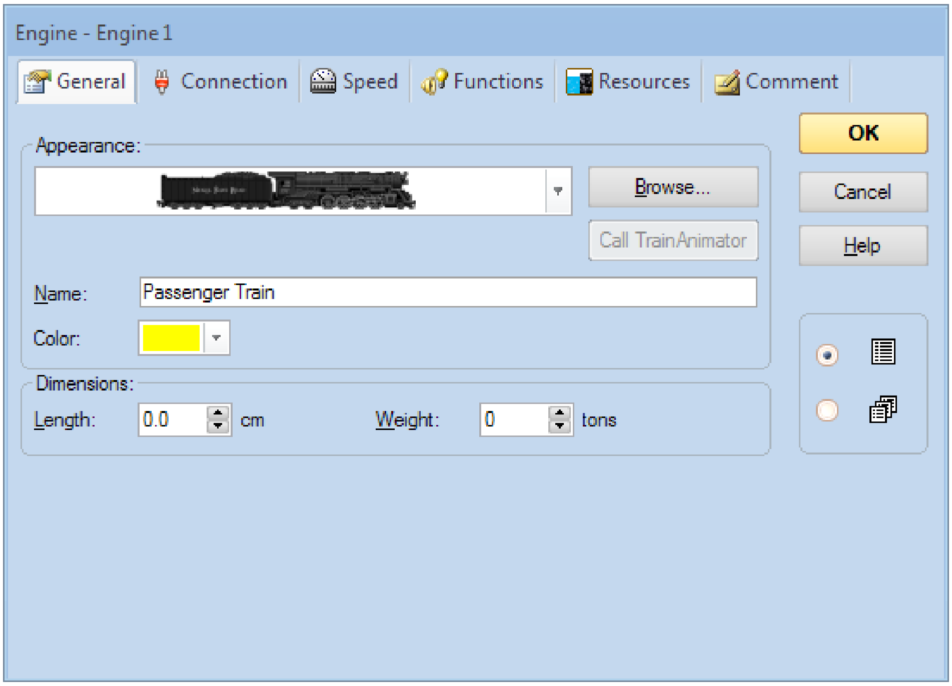
Address: 0

Read/Write Address on Programming Track:

Diagram 7: Specifying the Digital Address

Specify the same address, that you have been using previously to control the train with your digital system, in the field labeled **Address**. If you want to give your engine a name, that is more easy to remember, select the tab labeled **General** and enter an appropriate name. In the following we want to call this train “Passenger Train”.

You can see this name entered into the program in the image displayed below:



Name: Passenger Train

Color: 🟨

Length: 0.0 cm

Weight: 0 tons

Diagram 8: Entering a Name

You may have noticed, that the term “train” is being used here, while the images show the term “engine”. If you want to read more about this difference, refer to section 3.2, “Engines”. In the following we will continue using the more general term “train”.

Now press **OK** to close the dialog and to commit these changes. We will now return to the main screen and are ready to control the train:

Controlling a Train

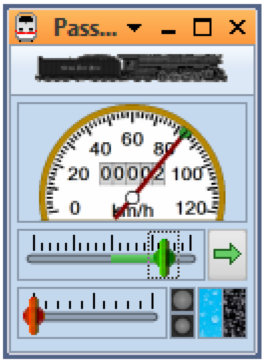


Diagram 9: Train Window

You may notice, that the color of some controls in the train window changed. This happened due to the fact, that we entered a digital address for our train. Now the software knows, how to control the train. To prove this move the mouse to the green control in the centre of the window. Click on it and drag the green control to the right. If everything has been done correctly so far, the train will slowly start to move. We have done the first successful step into model railroad computer control!

Before continuing I suggest that you enjoy playing with the train. Play around with the green control, which is actually an on-screen throttle. Drag it to the right and back to zero, then to the left and watch, how your train responds to these actions. See, how the speedometer needle above of the throttle indicates the scale speed of your running train. Watch the odometer increasing. By clicking the green arrow you will reverse the direction of your train. Dragging the red control, which is located between the throttle and the green arrow, will slow down the train. This control is actually a brake control. It can be used by experienced users, to apply the brake to a running train.

There are many more things, that **TrainController™** can do for realistic control of your trains. You can operate auxiliary functions (light, whistle, coupler, etc.), simulate the consumption of resources, adjust the momentum to your personal needs and scale the speed and distance measurements to the physical characteristics of your train. This is discussed in detail in chapter 3, “Train Control”.

Quick Start - Step 3:
Controlling Turnouts – The Switchboard

Creating a small switchboard control panel

So far the area in the background of the main window of **TrainController™** is still empty. It contains a number of cells, that are arranged in rows and columns. These cells are still empty. We want to fill this empty area with a small switchboard control panel for the following small track layout:

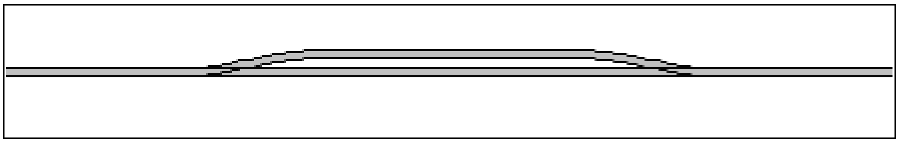


Diagram 10: Small Sample Layout

In the first step we will draw the track diagram in the switchboard window. First ensure that **Edit Mode** in the **View** tab is still turned on (see Diagram 4). Next select the **Draw** mode in the **Track** tab.

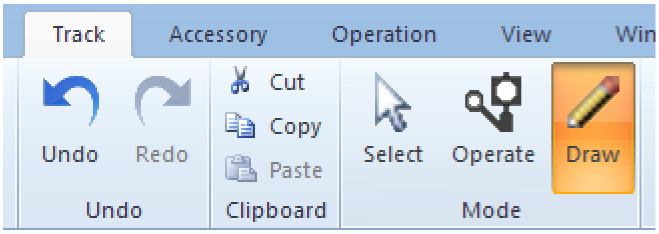


Diagram 11: Track Tab

Now move the mouse to the cell in the switchboard window, where the left end of our track diagram will be located. Click and hold the left mouse button and drag the mouse about 25 cells to the right. Then release the left mouse button. The following image should now be visible in the switchboard window:

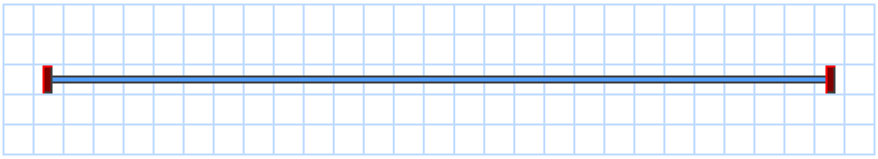


Diagram 12: Straight track section

We have drawn a straight track section. Now move the mouse to a cell on this track section, that is located about one third to the right of the left end. Click the left mouse button and drag the mouse one cell to the right and one cell up. Then release the left mouse button. Now you should see something similar to the following:

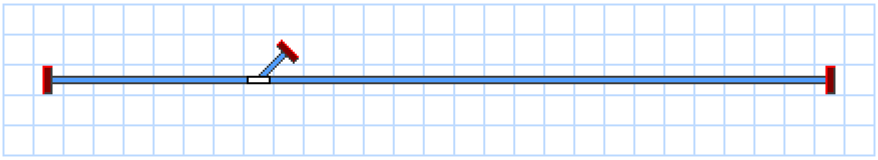


Diagram 13: Track section with turnout

The first turnout in the switchboard has now been created. Now click on the cell, where the diverging route of this turnout ends and drag the mouse to the right to a cell, that is located about one third left of the right end of the straight track section.

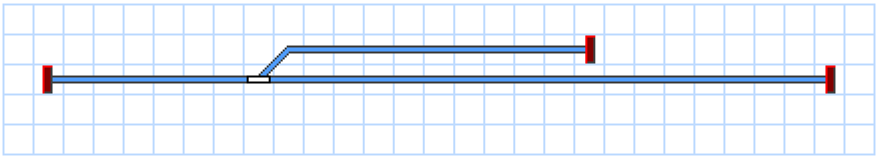


Diagram 14: Extending the track diagram

Finally click on the cell, where the last mouse movement ended, and drag the mouse one cell to the right and one cell down.

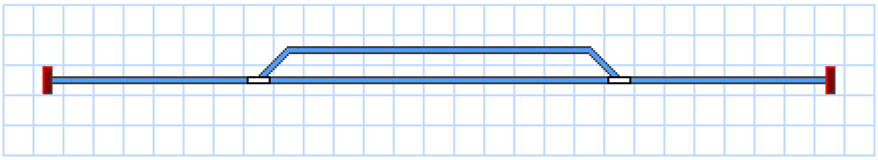


Diagram 15: The complete track diagram

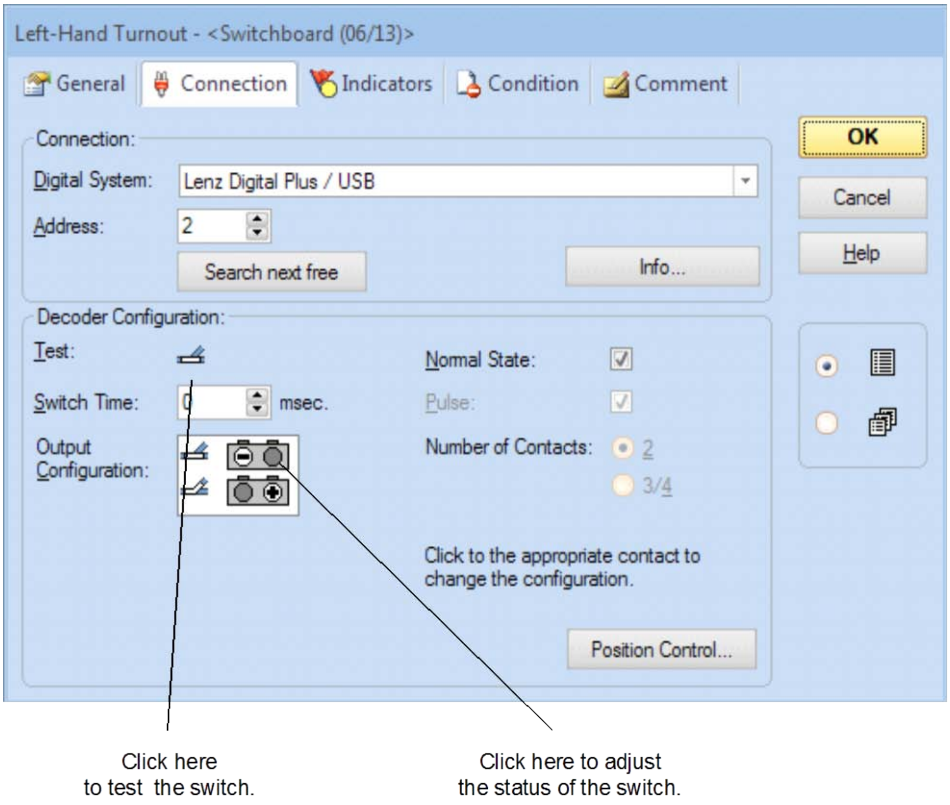
The track diagram of our small sample layout is now complete and should look like Diagram 15.

If you want to operate real turnouts of your existing model railroad with the track diagram control panel just created, try to identify a small area of your layout, that contains a similar track structure with two turnouts as shown above. Now operate these turnouts with your digital system. This step is recommended to verify, that the digital system and the turnouts are correctly working and to bring the digital addresses of the turnouts back to your mind. This is needed in the next step.

Preparing a Turnout for Model Railroad Computer Control

Ensure, that the **Edit Mode** option in the **View** tab is still active (see Diagram 4).

Now click on the symbol of the left turnout in the track diagram and select the **Properties** of the **Edit** tab. Do you remember? This command is used for all objects contained in the software (trains, turnouts, signals, routes, etc.), whenever it is required to change the settings of the particular object. The following window is now displayed:



Connection:

Digital System: Lenz Digital Plus / USB

Address: 2

Decoder Configuration:

Test:

Switch Time: 0 msec.

Output Configuration:

Normal State: ✓

Pulse: ✓

Number of Contacts: (●) 2 ( ) 3/4

Click to the appropriate contact to change the configuration.

Diagram 16: Specifying the Digital Address

Specify the same address, that you have been using previously to control the corresponding real turnout with your digital system, in the field labeled **Address**. Now click on the symbol of the turnout, that is located to the right of the label **Test**. The real turnout on your model railroad layout should now respond. Depending on the wiring of your turnout it is possible, that the image in the software and the physical turnout do not show the same status (closed vs. thrown). If this is the case click on the grey circle in the upper row of the **Output Configuration** to adjust the displayed status (see Diagram 16). The highlighting in the **Output Configuration** should now change and the displayed image of the turnout and the status of the turnout should be in sync, when you test the turnout again. Note, that the layout of this area may vary with the connected digital system.

Some advanced background information: in many cases, dependent on the digital system used, the highlighting in the **Output Configuration** will reflect the keys, that are to be pressed on the handheld of your digital system to set the turnout (or any other accessory, that is operated by turnout commands) to the corresponding state. Whenever the display of the turnout on the computer screen and the status of the turnout on your layout are not in sync, then you should operate the turnout first with your handheld and remember the keystrokes used to achieve a certain state. You should then translate these keystrokes to the **Output Configuration** of this turnout.

If you want to give your turnout a name, that is easier to remember, select the tab labeled **General** and enter an appropriate name.

Now press **OK** to close the dialog and to commit these changes. We will now return to the main screen and are ready to control the turnout. To do this, turn off **Edit Mode** in the **View** tab (see Diagram 4), move the mouse to the symbol of the turnout in the track diagram of the switchboard window, click on this symbol and watch, how the real turnout on your layout responds.

Finally perform the same for the right turnout symbol in the track diagram.

We are now able to control a train and a small layout manually with the computer. I suggest that you run the train back and forth on this small layout a little bit and play with different routes by changing the positions of each turnout prior to each run of the train.

In the next step we will learn, how trains can be operated automatically under control of the computer.

Quick Start - Step 4: Creating Blocks - Tracking Train Positions

Equipping the layout with feedback sensors

The most important prerequisite for controlling trains automatically with your computer or to monitor the movements of trains on the computer screen is equipping the layout with feedback sensors. These sensors are used to report train movements back to the computer. Based on this information **TrainController™** is able to take the right decisions to direct automatically running trains to their destination or to monitor the movement of trains.

Two types of feedback sensors can be used: occupancy sensors and momentary track contacts. Details of this difference and more detailed information about feedback sensors can be found in chapter 4, “Contact Indicators”.

In the following we assume, that occupancy sensors are used to control our small layout and that our layout is divided into four detection sections according to the following image:

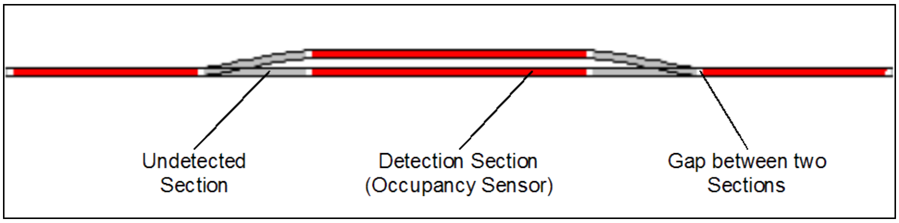


Diagram 17: Detection Sections and Occupancy Sensors

There are other ways to divide a layout into detection sections or to control it with momentary track contacts. Further, the scheme displayed above is also not necessarily the optimal solution. The above scheme has been chosen for this tutorial for reasons of simplicity and because it is sufficient to perform a quick start. Other variants for equipping your layout with feedback sensors are outlined in more detail in section 5.8.

Dividing the layout into Blocks

Another important prerequisite for controlling trains automatically with your computer or to monitor the movements of running trains is separating the layout into logical blocks. Blocks are the base elements for automatic train control and tracking of train positions. There is a close relation between feedback sensors and blocks: each block is associated with <u>one or more</u> feedback sensors.

There are certain guidelines for creation of blocks. They are outlined in detail in section 5.2, “Blocks”. According to these guidelines we divide our small sample layout into blocks as shown below:

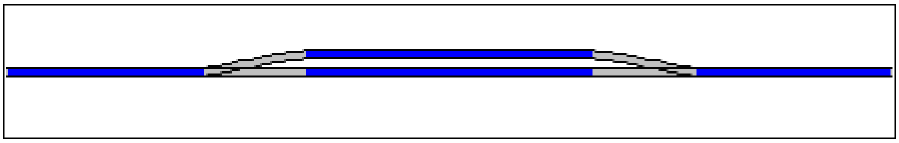


Diagram 18: Dividing a layout into Blocks

As you can see we have applied a 1:1 relation between blocks and detection sections here. Please note, that this is not always the case. In many cases more than one detection section or feedback sensor will be associated with one block. However, it is also possible to control your layout or appropriate parts of it with one feedback sensor per block. For reasons of simplicity and because it is sufficient for the quick start we go with one detection section per block here, too. Please keep in mind, however, that blocks and detection sections are <u>not</u> the same thing.

More details about this topic are outlined in detail in section 5.6, “Blocks and Indicators”.

Entering the locations of Blocks into the Switchboard

Blocks are represented by **TrainController™** on the computer screen by rectangular symbols. To enter the blocks, that are needed to control our train on our sample layout, turn on **Edit Mode** in the **View** tab and select the **Block** command in the **Block** group of the **Accessory** tab.

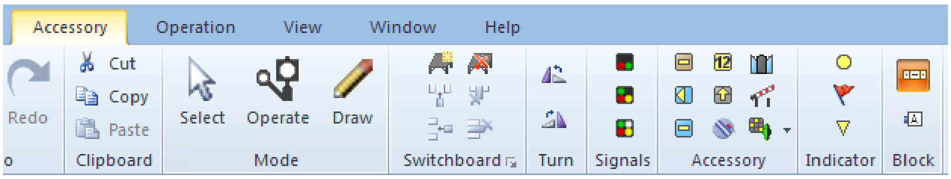


Diagram 19: Accessory Tab

Now click on the cell, that is located right of the cell, that contains the left end of our track diagram. A block will appear at this location.

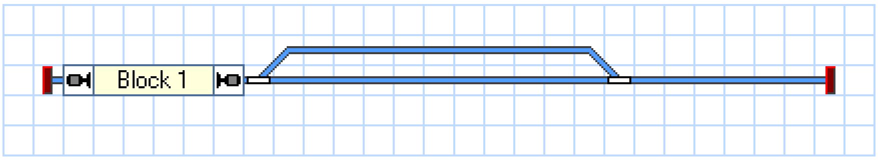


Diagram 20: Block in the Switchboard

Please do the same for the three other blocks. Note, that the cell, where you click, determines the leftmost end of the block. Ensure also, that you click on a cell, that contains a piece of straight track.

You can change the size of each block by dragging its left or right border.

If everything was done correctly, the track diagram should look like the following image:

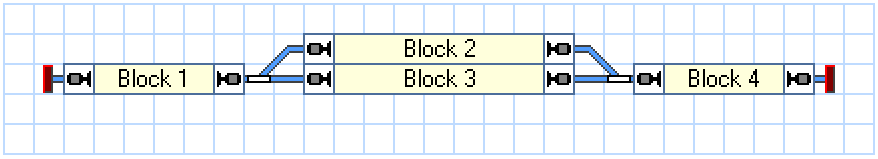


Diagram 21: The complete Track Diagram with all Blocks

Assigning Feedback Sensors to Blocks

There is a close relation between feedback sensors and blocks: each block is associated with one or more feedback sensors. To assign a feedback sensor to a block, select “Block 1” in the switchboard track diagram and call the **Properties** command of the **Edit** tab.

Then select the **Block Editor** tab in the opened dialog box.

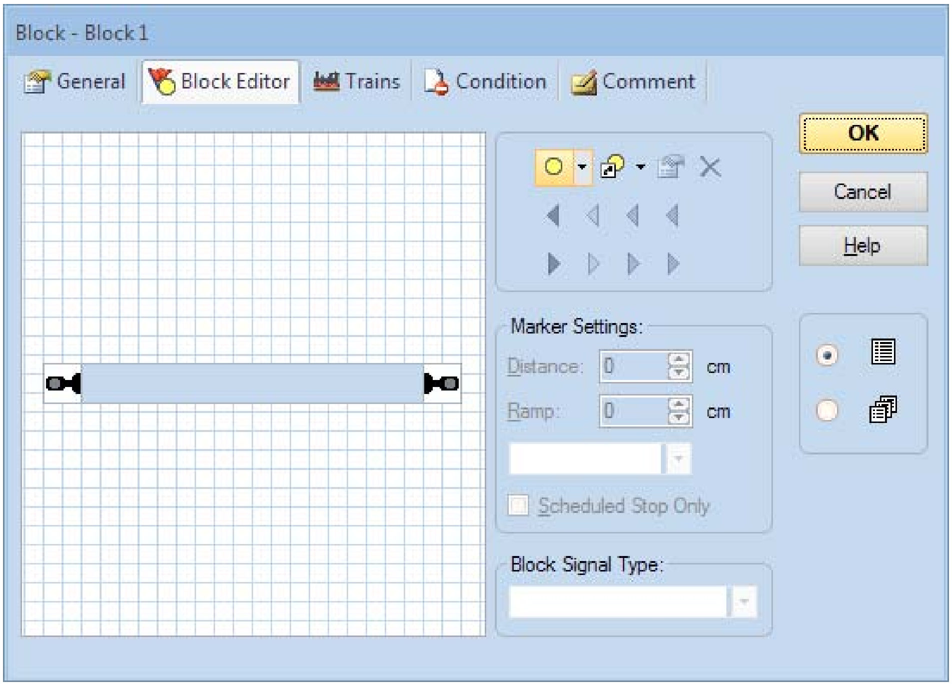
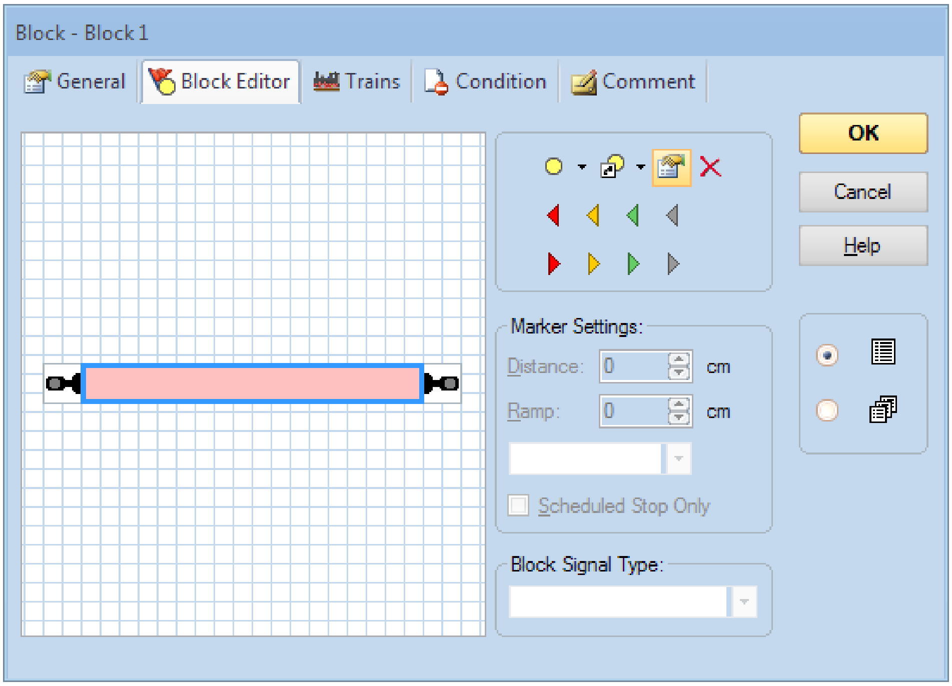


Diagram 22: Block Editor

It shows the properties of the block and indicates, that no sensor is yet assigned to this block.

Click on ○ in the tool bar of the block editor. This is the item, which is highlighted in Diagram 22. The block editor now changes as follows:



Marker Settings:

Distance: 0 cm

Ramp: 0 cm

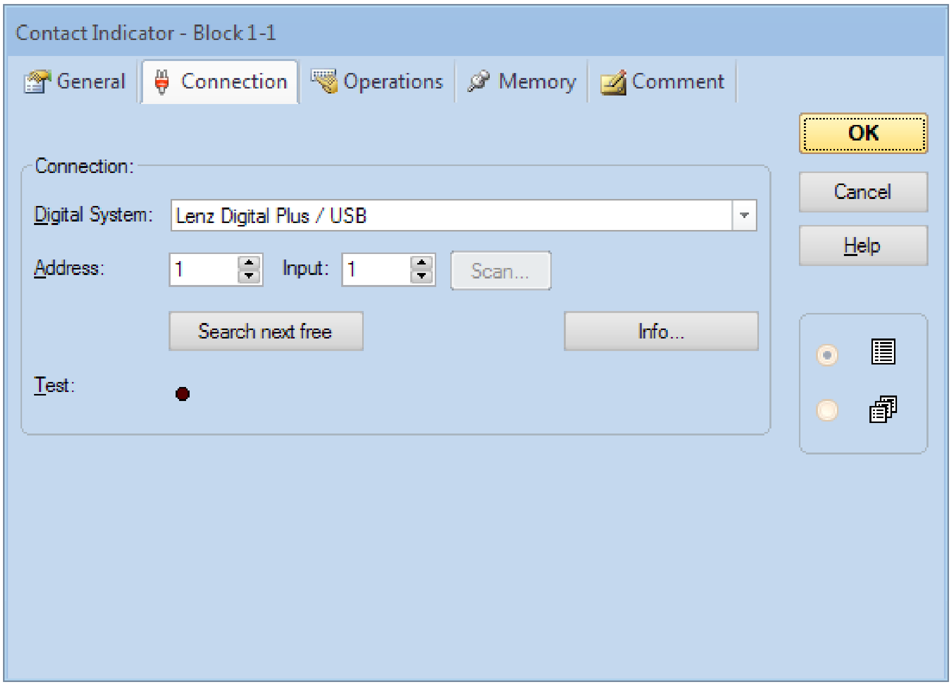
Scheduled Stop Only

Block Signal Type:

Diagram 23: Block Editor with Contact Indicator

The center of the block editor now shows a reddish rectangle. This rectangle is called contact indicator and represents the occupancy section within the block, which is monitored by the feedback sensor.

Now click on the contact indicator (i.e. the reddish rectangle) and then click on the **Properties** command in the tool bar of the block editor. This is the highlighted symbol in Diagram 23. The dialog box displayed below is opened:


Contact Indicator - Block 1-1

General | Connection | Operations | Memory | Comment

Connection:

Digital System: Lenz Digital Plus / USB

Address: 1 Input: 1 Scan...

Search next free | Info...

Test: ●

OK | Cancel | Help

Diagram 24: Specifying the Digital Address of a Contact Indicator

Now specify the digital address of the feedback sensor, that belongs to this contact indicator. In most cases this is the digital address of the feedback decoder and the number of the contact input of this decoder, to which the sensor is connected.

To test your settings, put a train or anything else, that is suited to trigger a feedback event, into the detection section, that corresponds to “Block 1”. The block in the track diagram in the switchboard should now change its color to red:

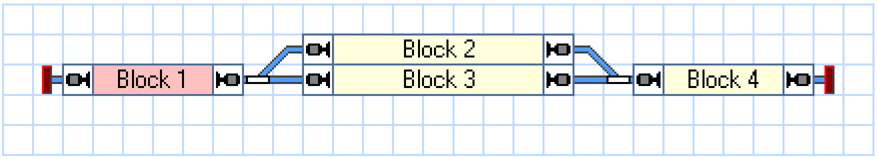


Diagram 25: Indication of an occupied Block

Now create and assign contact indicators to the other three blocks, too.

If this has been done correctly, the blocks in the switchboard will change their color according to the movements of your train on the layout. Play around a little bit with your train and watch how the blocks in the switchboard are indicated.

Displaying train positions on the Computer Screen

Now we are ready for *train tracking*, i.e. displaying of train positions on the computer screen.

To do this move your real train into "Block 1", if it is not located there already. Ensure, that the train is heading towards the other blocks, i.e. that it has to run forward, in order to go to "Block 2" or "Block 3", respectively.

Then turn off **Edit Mode** in the **View** tab (see Diagram 4). Next select "Block 1" in the switchboard and call the **Assign Train** command of the **Block** group in the **Operation** tab according the following image:

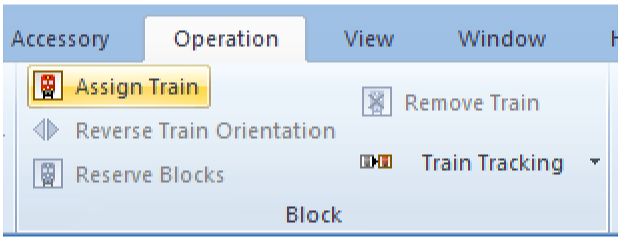


Diagram 26: Operation Tab

In the following dialog select the "Passenger Train" and select the train orientation by marking the option near the arrow pointing to the right.

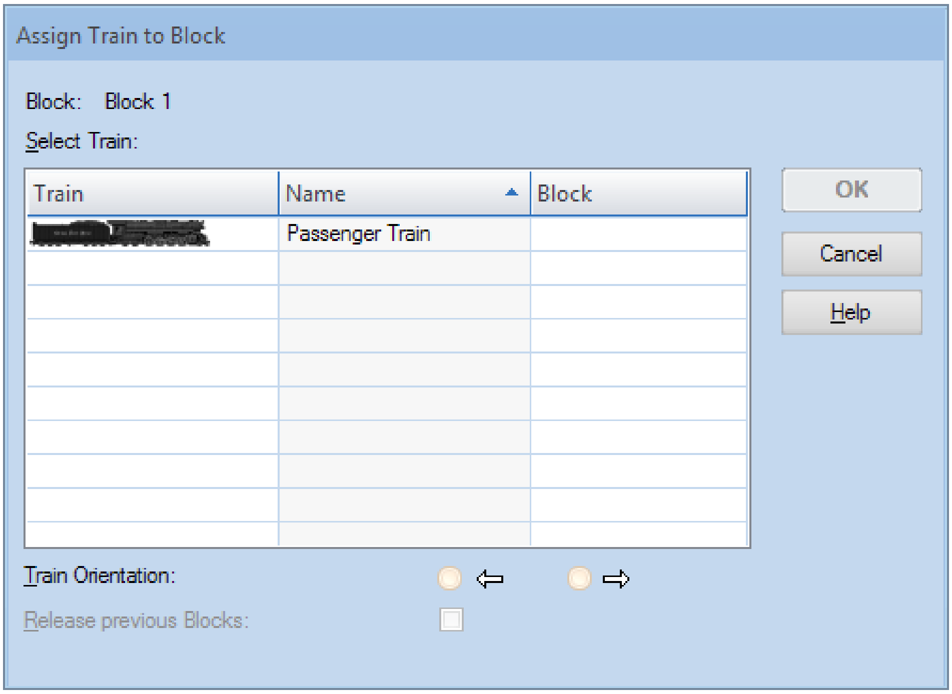


Diagram 27: Assigning a Train to a Block

After pressing **OK** the symbol of the train will appear in "Block 1" in the switchboard control panel:

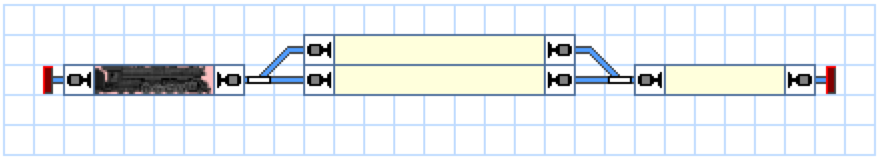


Diagram 28: Display of Train Positions on the Computer Screen

Instead of using the **Assign Train** command you can also drag and drop the train symbol with the mouse from another place on the computer screen to "Block 1", if the train symbol is visible somewhere else.

Now run the train with the on-screen throttle of the train window displayed in Diagram 9. When the train travels to another block, the display should be updated accordingly and the symbol of the train should move to the symbol of the other block. If you are testing this on a bigger layout ensure, that the train does not leave the area, that is controlled by blocks and feedback sensors as described so far.

Simulating Train Movements on the Computer Screen

If no layout is connected, you can also simulate the described movements on the computer screen. For this purpose call the **Simulator** command in the **Windows** Menu of the **Window** tab.

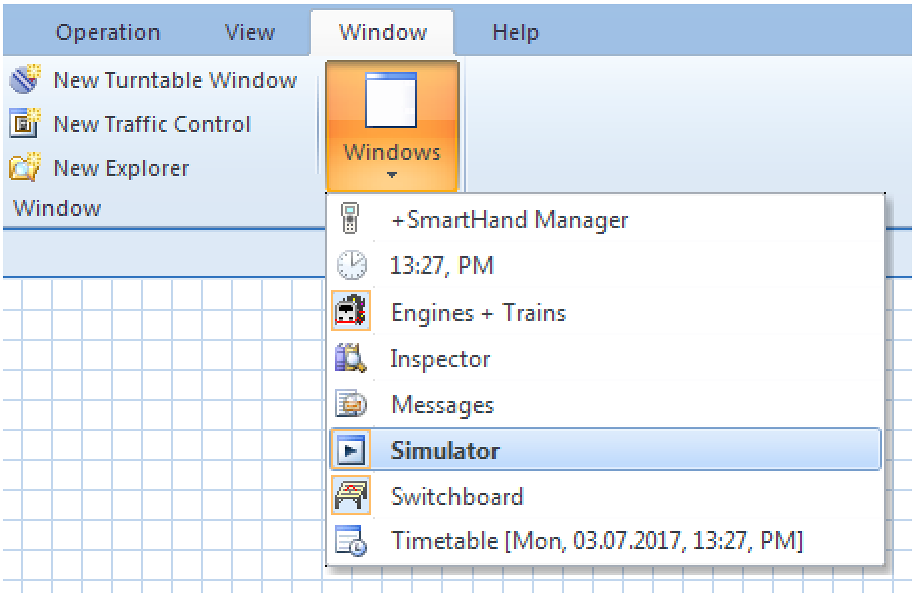


Diagram 29: Window Tab and Window Menu

This opens the Simulator window as displayed below:

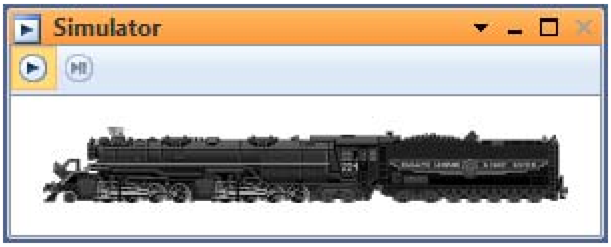


Diagram 30: Simulator

Start the simulator by clicking on the leftmost symbol in the toolbar of the simulator window. This is the highlighted item in Diagram 30.

If you now start the train with the train window in the forward direction, i.e. by dragging the green on-screen throttle in the train window to the right, you will notice, that the symbol of the train moves from block to block on the computer screen. You can even change the turnout positions and watch, how the movement of the train symbol follows accordingly.

If all steps are performed correctly so far, then you are able to control the movement of your train and operate your turnouts with **TrainController™**. You are also able to track the positions of moving trains on the computer screen.

Quick Start - Step 5: Controlling Trains Automatically

Spontaneous Runs

The last part of our quick start tutorial is automatic control of running trains. In the first step a train located in “Block 1” of our small sample layout will run to “Block 4” and stop there. To do this run our train manually back to “Block 1”. Train tracking should ensure, that the display reflects this movement and finally looks like Diagram 28. Please ensure that **Edit Mode** in the **View** tab is turned off (see Diagram 4).

Now select “Block 1”, i.e. the block, where the train image is located, and call the **Spontaneous Run to the Right** command in the **Spontaneous Run** group of the **Operation** tab.

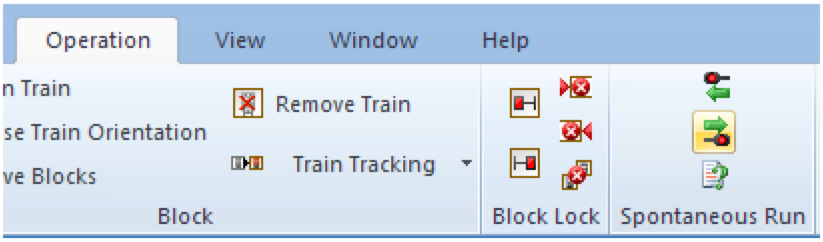


Diagram 31: Start a spontaneous run (to the right)

The display in the switchboard should now change and show something similar to the following:

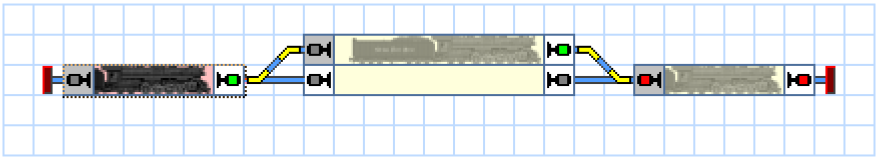


Diagram 32: Spontaneous run

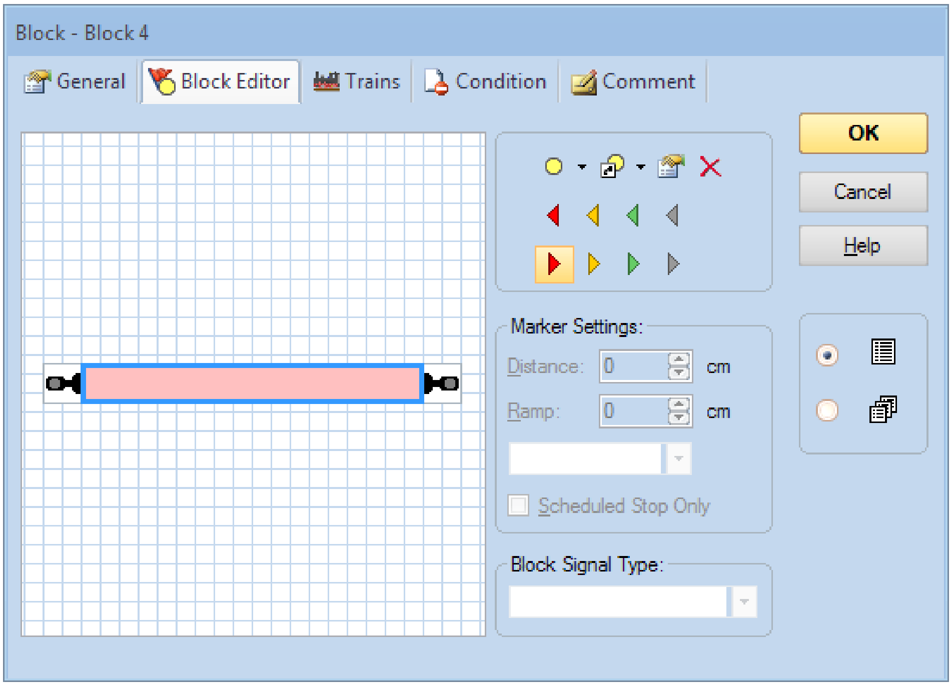
Simultaneously the real train on your layout should start to move now and run through “Block 2” or “Block 3” to “Block 4”, where it should slow down and stop.

The same maneuver can be simulated without a connected model railroad layout by turning on the Simulator (see page 54).

Adjusting the Stop Location

You may have noticed, that the train stopped as soon as the occupancy sensor in "Block 4" was turned on. In order to adjust the location, where the train stops in "Block4", turn on **Edit Mode** via the **View Tab** (see Diagram 4), select "Block 4" and call the **Properties** command of the **Edit** tab.

Then select the **Block Editor** tab as displayed below.



Marker Settings:

Distance: 0 cm

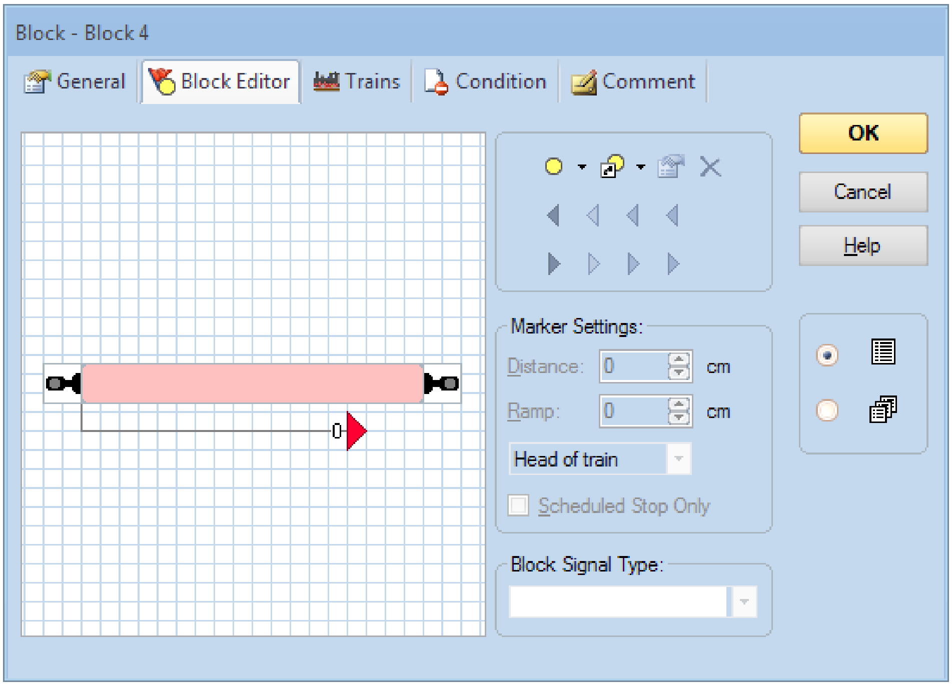
Ramp: 0 cm

Scheduled Stop Only

Diagram 33: Block Editor

Click on the red rectangle (contact indicator) in the center of the block editor and then on the **Insert Stop Marker Right** command ▶ in the tool bar of the block editor (see Diagram 33). A triangle symbol will now appear in the work area of the block editor. Drag this triangle to the right with the mouse.

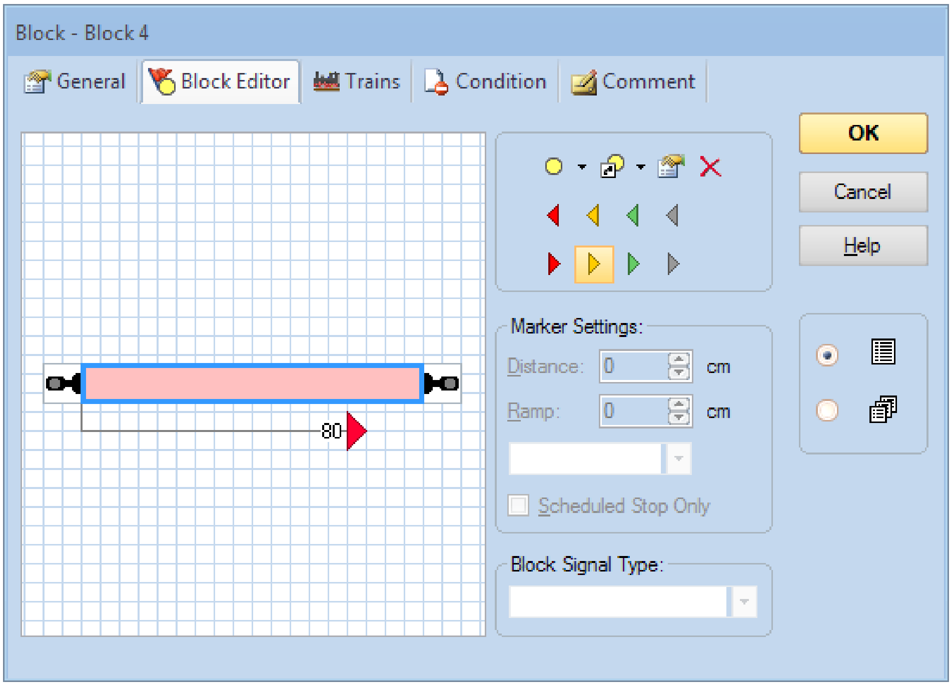
The block editor should look like the following image now:



Marker Settings:
Distance: 0 cm
Ramp: 0 cm
Head of train
☐ Scheduled Stop Only
Block Signal Type:

Diagram 34: Block Editor with Stop Marker

The red triangle marks the point, where the train will stop in “Block 4”. We assume, that this point is located 80 cm away from the left border of the occupancy section. Click on the red triangle, and enter 80 in the **Distance** box.

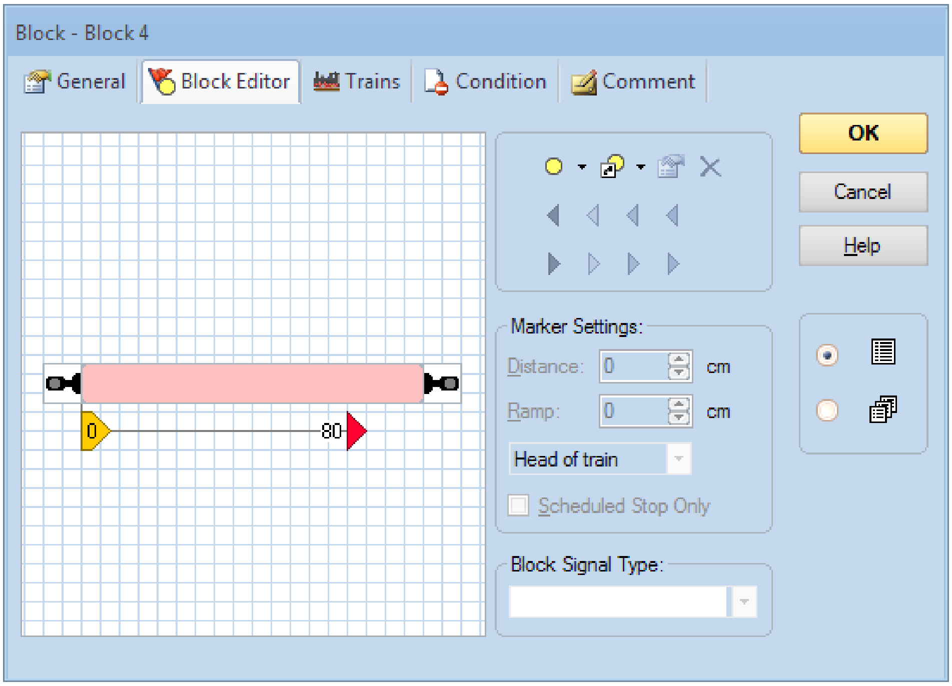


Marker Settings:
Distance: 0 cm
Ramp: 0 cm
Scheduled Stop Only
Block Signal Type:

Diagram 35: Block Editor

Click on the red rectangle (contact indicator) in the center of the block editor and then on the **Insert Brake Marker Right** command in the tool bar of the block editor (see Diagram 35). A second triangle symbol will now appear in the work area of the block editor.

The block editor should now look like the following image:


Block - Block 4

Marker Settings:
Distance: 0 cm
Ramp: 0 cm
Head of train
Scheduled Stop Only
Block Signal Type:

Diagram 36: Block Editor with Brake and Stop Marker

The yellow triangle marks the point, where the train will begin to slow down in “Block 4”. Since we want to slow down the train within 80 cm beginning from the left border of the occupancy section, click on the yellow triangle, and enter 80 in the **Ramp** box.

Now press **OK** and repeat the procedure outlined in the section “Spontaneous Runs”. The train should now begin to slow down, when it arrives at the occupancy section in “Block 4” and stop somewhere inside of “Block 4”.

If the train does not stop at the desired location in “Block 4”, then adjust the ramp and distance settings accordingly as described above. More information can be also found in section 5.6, “Blocks and Indicators”.

During later practice, when everything is configured completely, we expect a train to stop about 80 cm behind the beginning of the occupancy section, if 80 cm is specified as distance for a red triangle. However, this requires installation of an additional sensor at the point, where trains will stop, (see also section 5.8, “Arranging Indicators and Mark-ers in a Block”) or calibration of the speed profile of the locomotive (described in section 3.5, “The Speed Profile”). As far as this tutorial is concerned we are satisfied, if we can get the train to slow down and to stop smoothly somewhere within a block.

Now add a red and a yellow triangle, which point to the left, to “Block 4” in the same way described above and specify similar settings for **Distance** and **Ramp**.

Finally do the same for all other blocks “Block 1, “Block 2” and “Block 3”.

Creating a Commuter Train

In the next step we want a train located in “Block 1” of our small sample layout to run back and forth between “Block 1” and “Block 4”. To do this run your train manually back to “Block 1”. Train tracking should ensure, that the display reflects this movement and finally looks like Diagram 28. Ensure that **Edit Mode** in the **View** tab is turned off (see Diagram 4).

Now select “Block 1”, i.e. the block, where the train image is located, and call the **Rules for Spontaneous Runs** command in the **Spontaneous Run** group of the **Operation** tab.

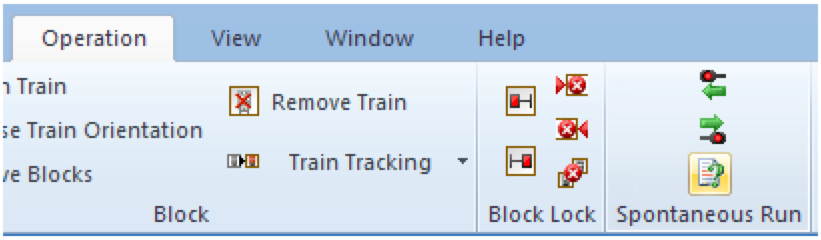
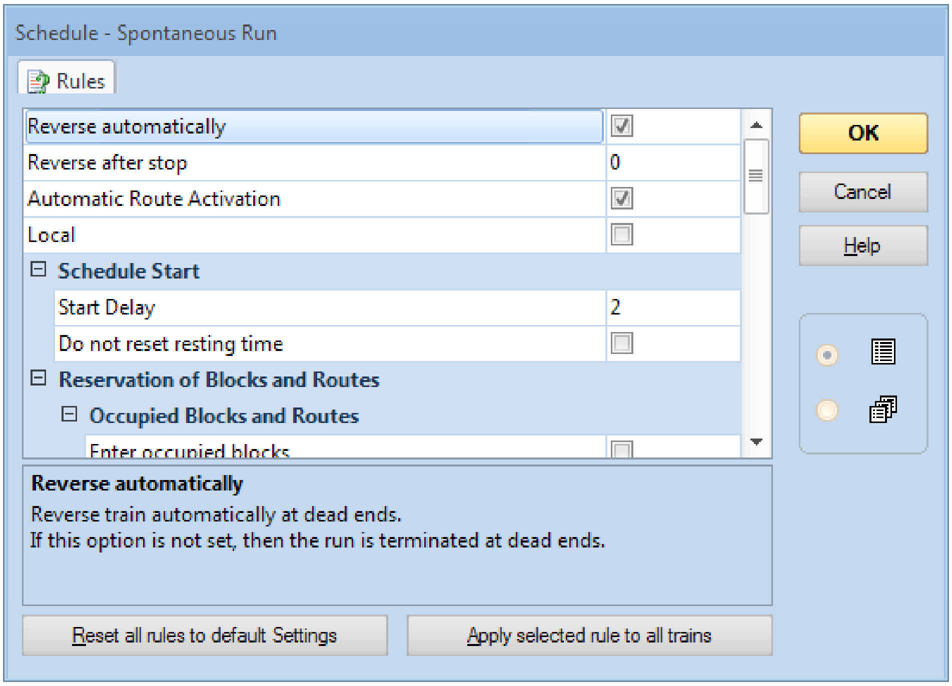


Diagram 37: Rules Command

Then check the option **Reverse automatically**:


Schedule - Spontaneous Run

Rules

Reverse automatically	✓
Reverse after stop	0
Automatic Route Activation	✓
Local	☐
⊟ **Schedule Start**	
Start Delay	2
Do not reset resting time	☐
⊟ **Reservation of Blocks and Routes**	
⊟ **Occupied Blocks and Routes**	
Enter occupied blocks	☐

Reverse automatically
Reverse train automatically at dead ends.
If this option is not set, then the run is terminated at dead ends.

OK
Cancel
Help
Reset all rules to default Settings
Apply selected rule to all trains

Diagram 38: Rules for Spontaneous Runs

This will cause the train to reverse in “Block 4”, because this is a dead end in our small layout, and travel back to “Block 1”. Back in “Block 1” it will reverse again and run back to “Block 4” and so on.

Press OK, select “Block 1”, i.e. the block, where the train image is located, call the **Spontaneous Run to the Right** command of the **Operation** tab and watch, how this works.

This can also be simulated without a connected model railroad layout by turning on the Simulator (see page 54).

AutoTrain™ by Drag and Drop

In the next step we want the train to start in “Block 1” and stop in “Block 3”. This cannot be accomplished with spontaneous runs as outlined above, because under control of a spontaneous run the train may select another path, i.e. “Block 2”, for its travel. Furthermore, it will not stop until it reaches a dead end (here “Block 4”).

To do this run our train manually back to “Block 1”. Train tracking should ensure, that the display reflects this movement and finally looks like Diagram 28. Ensure that **Edit Mode** in the **View** tab is turned off (see Diagram 4).

Now select the **AutoTrain by Drag and Drop** command in the **AutoTrain** group of the **Operation** tab.

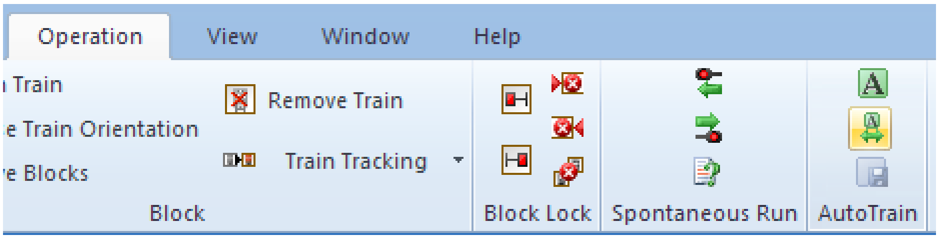


Diagram 39: AutoTrain by Drag and Drop

Then move the mouse pointer to the train symbol located in “Block 1”. The following symbols will appear:

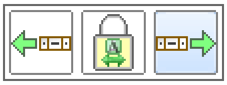


Diagram 40: Determine the start Block of AutoTrain™

Click on the rightmost symbol, hold the mouse button pressed and drag the mouse to “Block 2” until the mouse pointer shows the following symbols:

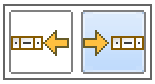


Diagram 41: Determine the destination Block of AutoTrain™

Move the mouse pointer to the rightmost symbol and release the left mouse button. The display in the switchboard should now change and show something similar to the following:

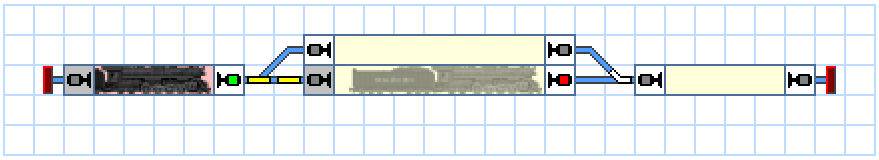


Diagram 42: Running a train automatically with AutoTrain™

Simultaneously the real train on your layout should start to move and run from “Block 1” to “Block 3”, where it should slow down and stop.

After the train has stopped, you can let it run back to “Block 1” automatically by calling the **AutoTrain by Drag and Drop** command once more and dragging the train symbol back to “Block 1”. Please ensure, that the mouse pointer now points to the symbol with the left arrow before clicking and before releasing the left mouse button, since the train should now run to the opposite direction.

Commuter Train with intermediate Stop

As a final step of our tutorial we want to run the train automatically back and forth between “Block 1” and “Block 4” several times. The train will always select the right block with regard to direction of travel, i.e. when running to the right, the train will pass “Block 3”, when running to the left the train will pass “Block 2”. Additionally the train will perform a short intermediate stop in “Block 2” and “Block 3”, respectively, during each pass.

This cannot be accomplished with AutoTrain by Drag and Drop, because this does not allow us to specify intermediate stops.

To do this, run the train manually back to “Block 1”. Train tracking should ensure, that the display reflects this movement and finally looks like Diagram 28. Ensure that **Edit Mode** in the **View** tab is turned off (see Diagram 4).

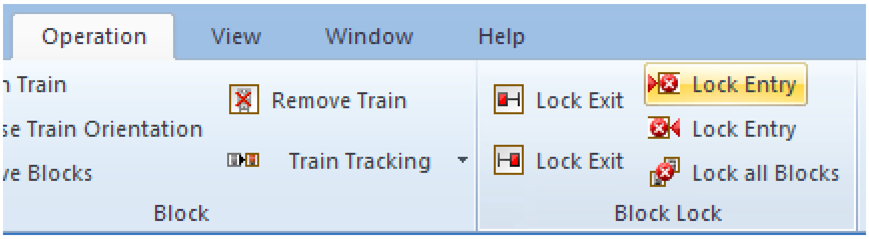


Diagram 43: Locking the left Entry of a Block

Now select “Block 2” and call the **Lock Entry (left)** command in the **Block Lock** group of the **Operation** tab. This ensures, that the train will not pass through “Block 2” on its way to “Block 4”. Then select “Block 3” and call the **Lock Entry (right)** command in the **Block Lock** group of the **Operation** tab.

The switchboard should now look as follows:

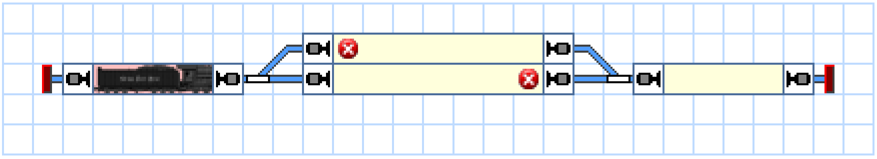


Diagram 44: Locked Block Entries

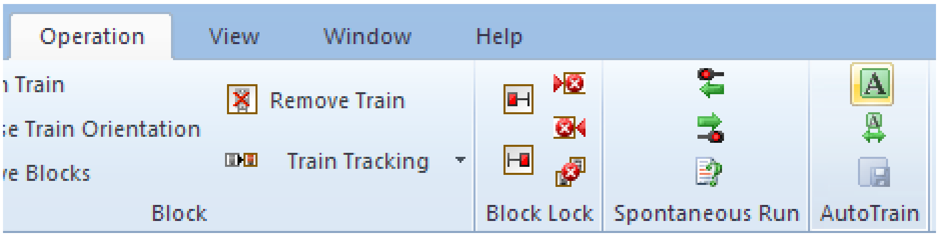


Diagram 45: Operation Tab

Next select “Block 1” and call the **AutoTrain** command in the **AutoTrain** group of the **Operation** tab. This opens the **AutoTrain**™ tool bar as displayed below:

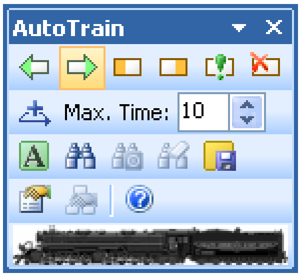
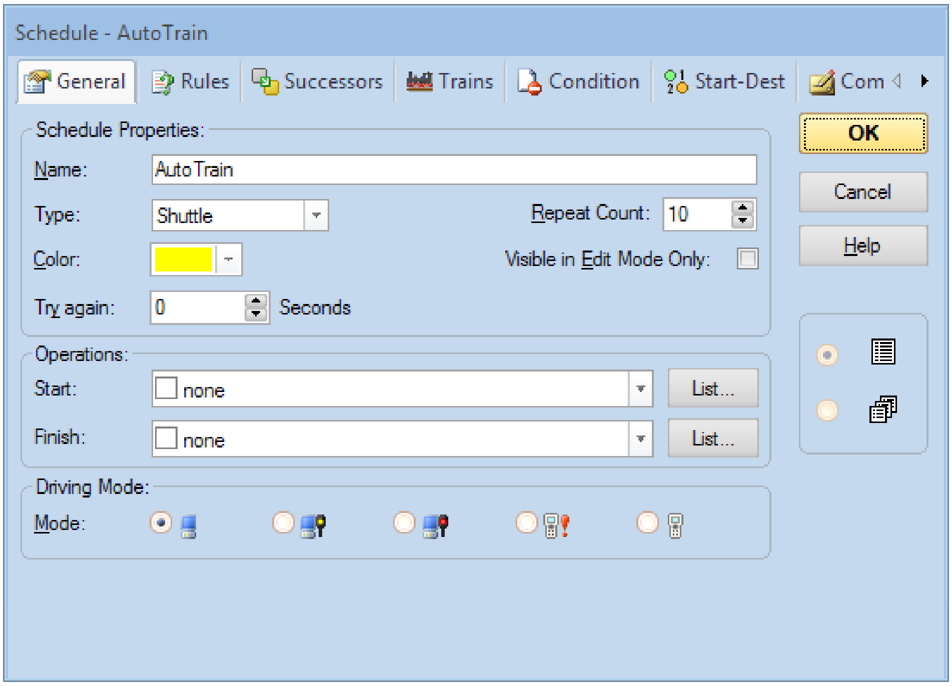


Diagram 46: AutoTrain™ Tool Bar

Ensure that a green marking appears at the right hand side of “Block 1”. This indicates, that we want to start our train in this block and to travel to the right. If this marking is not set, select “Block 1” and press ⇨.

Next select “Block 4” and press ▭. This indicates, that the train will enter “Block 4” from the left to the right and stop here. Now press 🔍. The software now checks, whether there is a path from “Block 1” to “Block 4”. As a result “Block 2” and “Block 3” are displayed on the screen with the same intensity as “Block 1” and “Block 4”. This indicates, that there is a path from “Block 1” to “Block 4”, that passes “Block 2” or “Block 3”, respectively.

Now press . **TrainController™** opens the following dialog:



Schedule - AutoTrain

Schedule Properties:

Name: AutoTrain

Type: Shuttle

Repeat Count: 10

Color: 🟨

Visible in Edit Mode Only: ☐

Try again: 0 Seconds

Operations:

Start: ☐ none List...

Finish: ☐ none List...

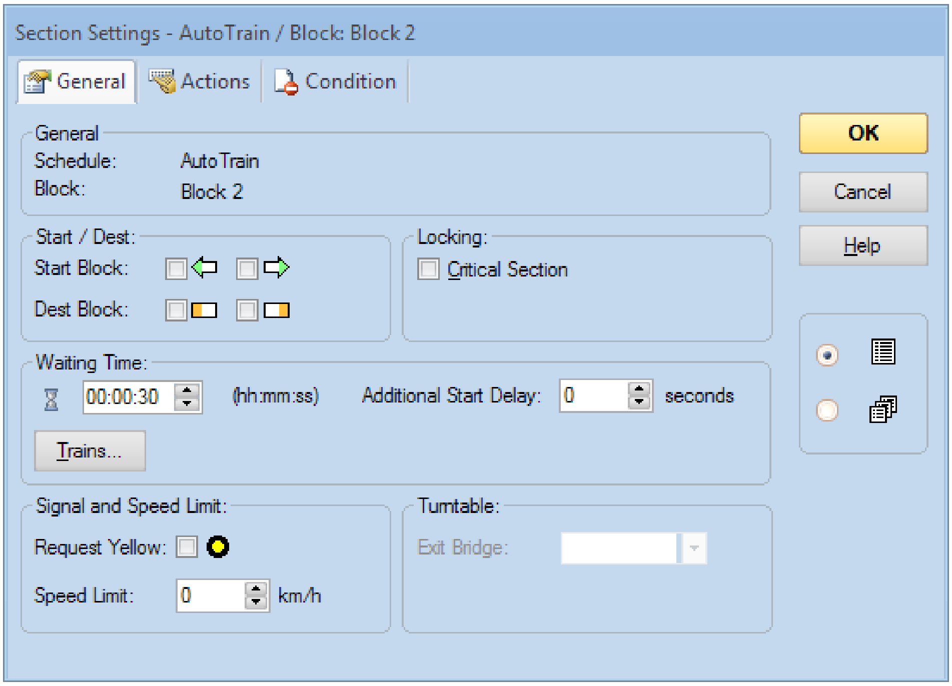
Driving Mode:

Mode: ●

Diagram 47: Specifying a Shuttle Train

Here select **Shuttle** as **Type** and **10** as **Repeat Count**. This tells the software, that you want to create a train, that will run back and forth (shuttle) ten times. You can specify any number as **Repeat Count**. Commit your settings with **OK**.

Now select "Block 2" and press . **TrainController™** opens the following dialog:


Section Settings - AutoTrain / Block: Block 2

General | Actions | Condition

General
Schedule: AutoTrain
Block: Block 2

Start / Dest:
Start Block: ☐ ☐
Dest Block: ☐ ☐

Locking:
☐ Critical Section

Waiting Time:
00:00:30 (hh:mm:ss) Additional Start Delay: 0 seconds

Trains...

Signal and Speed Limit:
Request Yellow: ☐
Speed Limit: 0 km/h

Turntable:
Exit Bridge:

OK | Cancel | Help

Diagram 48: Specifying a Waiting Time

Enter **00:00:30** in the box below **Waiting Time**. This tells the software, that the train is to wait 30 seconds in “Block 2”. Commit your settings with **OK**. Perform the same steps for “Block 3” to specify a waiting time for “Block 3”.

Now press A. The train will now start to move towards “Block 3”. In “Block 3” it slows down and stops for a while. Then it starts again and enters “Block 4”. Here it slows down again, stops and starts in the opposite direction. In “Block 2” it slows down and stops again. After a while it starts again and runs to “Block 1”, where it stops. Then the complete cycle is repeated again.

You are now able to configure control of automatically running trains. Beginning with spontaneous runs you can easily run a train automatically without further measures. AutoTrain by Drag & Drop provides more control over, where trains will go. The AutoTrain Symbol bar, that we used in the last step, provides full control of the train during it’s automatic run.

However, **TrainController™** is able to perform much more complex train control on much more complicated track layouts. **TrainController™** cannot only control perpetually running commuter trains or trains, that run continually around a loop. **TrainController™** can perform intermediate train stops or execute train functions automatically, such as switching on lights or playing sounds. **TrainController™** can operate hidden yards automatically or control trains according to time tables. To learn, how these amazing things can be done with **TrainController™** continue reading part II of this Users Guide.

Part II

Fundamentals

B

1 Introduction

1.1 Overview

TrainController™ is a system which enables you to operate a model railroad layout from a Personal Computer running Microsoft Windows 10, 8, 7, Vista or XP.

TrainController™ provides you with the ease of point and click operation of your turnouts, signals, routes and other accessories displayed on track diagram panels. Track diagram panels are individually created for each yard or section, as desired. You can run your trains with on-screen throttles, external hand held throttles connected to your computer, or with your favorite throttles or hand held throttles supported by your digital system. You can operate digital engines equipped with their own decoders, as well as conventional models without decoders. Digital and conventional engines can run on the same track. Far-reaching automation features make railroad operations manageable by one person and match those found on the largest club layouts. You can see on the screen which engine/train is on which track.

Supported Digital and Control Systems

The software supports major digital and control systems which provide a computer interface. Among others the following systems are supported (list is not exhaustive):

- Digitrax LocoNet
- Lenz Digital Plus
- North Coast Engineering Master Series (NCE)
- Roco Z21, z21, Multizentrale Pro, Interface 10785
- Maerklin Central Station 1, 2 and 3
- Maerklin Digital
- ESU ECoS and Navigator
- CTI
- RCI
- Trix Selectrix
- Müt Digirail
- Rautenhaus Digital
- Uhlenbrock Intellibox, Intellibox 2, Intellibox Basic and IB-COM
- Tams Master Control

- Fleischmann Z1, z21, Twin Center, Multizentrale Pro, Profi-Boss
- Littfinski HSI-88
- Zimo
- Doehler & Haas / MTTM Future Central Control
- and others

For the complete list as well as detailed information about the range of support for certain systems refer to the Help menu of **TrainController™**.

You can run different systems simultaneously on different serial or USB ports. This increases the maximum number of trains, turnouts, signals and feedback indicators that can be operated. If your favorite digital system is not able to report the state of feedback sensors, then you are able to enlarge this system with a second system that is able to do this.

TrainController™ also supports an offline mode that allows trial operation without a connection to a real model railroad. Up to twelve digital and control systems can be connected simultaneously.

! For each digital system additional information is provided that further explains the use of the particular system with **TrainController™**. This information can be found by opening the help menu of **TrainController™** and entering the name of the digital system or the name of the manufacturer as search key.

Modes of Train Operation

With **TrainController™** it is possible to run digitally equipped engines as well as conventional engines that don't have digital decoders. The operation of conventional engines is done with stationary block decoders; i.e. decoders or computer controlled throttles that are mounted at fixed positions on your model railroad rather than in each engine.

This feature is useful if:

- you have a large collection of engines and not all are digitally upgraded.
- you have a conventional - i.e. non-digital - operated model railroad and want to control your layout with your computer without installing a digital decoder in each engine first.
- the models of your engines are very small and the decoders do not fit into the engines (e.g. when you run Z scale).

In all, **TrainController™** provides three methods of operation:

- Operating trains using mobile decoders installed in the engines ("*Computer Command Control*").
- Operating conventional trains using stationary block decoders with static assignment to track sections ("*Computer Section Control*")
- Operating conventional trains using stationary block decoders with dynamic assignment to track sections ("*Computer Cab Control*").

Additionally, it is possible to use these methods simultaneously, i.e. it is possible to run conventional engines and digital engines on the same track - even if your digital system does not support this feature.

Use

TrainController™ is easy to use. It provides an easily learned, intuitive, graphical user interface that is developed according to the following guidelines:

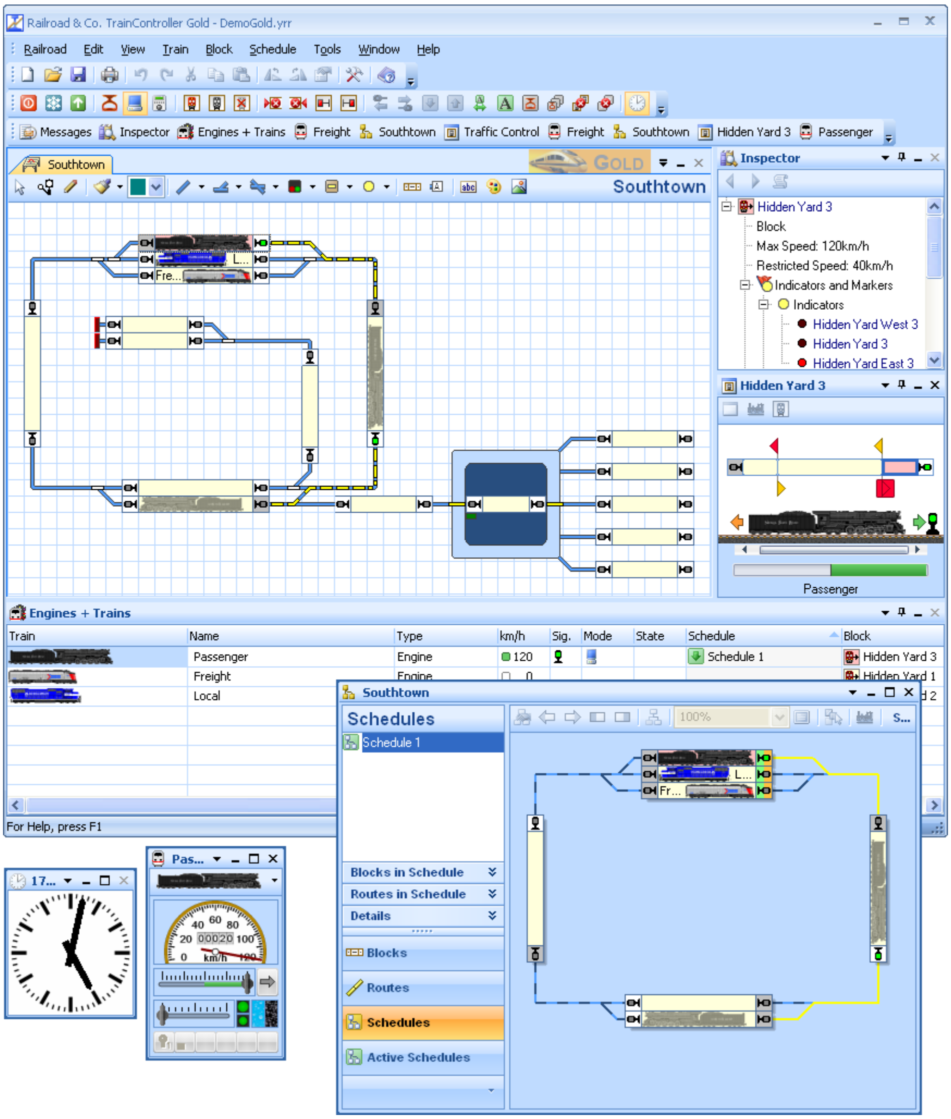
- Use of **TrainController™** is possible without the need to be a computer expert or programmer.
- Graphical items are provided instead of an abstract command syntax.
- Operation is based on natural objects like trains, turnouts, signals, etc. instead of digital addresses or something else.
- Activities are natural - point to a signal and set it to red with a simple mouse click instead of issuing a command like "set contact output of decoder 35 to 1". Accelerate a train to speed 35 mph instead of typing "set speed level of train decoder 16 to 7".
- Automatic Operation can be arranged within minutes without the need to learn a programming language first.

Components

Each component of **TrainController™** has its own special functionality and most of them can be used separately. You only need to concentrate on the components you choose to use. The control of trains and the operation of turnouts and signals is separated.

These are the components of **TrainController™**:

- **The Switchboard:** easy to use control panel editor for the operation of turnouts, signals and other accessories with point and click ease. It allows manual, semi-automatic and fully automatic operation of your accessories.
- **The Train Window:** on-screen throttles and various cab instruments for realistic train operation
- **The Dispatcher:** intelligent monitoring and operation of your entire model railroad, or just parts, that can be arranged within minutes


Railroad & Co. TrainController Gold - DemoGold.yrr
Railroad Edit View Train Block Schedule Tools Window Help
Messages Inspector Engines + Trains Freight Southtown Traffic Control Freight Southtown Hidden Yard 3 Passenger
Southtown
GOLD
Southtown
Inspector
Hidden Yard 3
Passenger
Engines + Trains
Southtown
Schedules
Schedule 1
100%
Blocks in Schedule
Routes in Schedule
Details
Blocks
Routes
Schedules
Active Schedules
17...
Pas...
km/h

Hidden Yard 3
Block
Max Speed: 120km/h
Restricted Speed: 40km/h
Indicators and Markers
Indicators
Hidden Yard West 3
Hidden Yard 3
Hidden Yard East 3

Train	Name	Type	km/h	Sig.	Mode	State	Schedule	Block
	Passenger	Engine	120				Schedule 1	Hidden Yard 3
	Freight	Engine	0					Hidden Yard 1
	Local							d 2

For Help, press F1

Diagram 49: RAILROAD & CO. TrainController™

Automatic Operation

Because you want to control your model railroad with your computer, you are probably interested in operating parts (or all) of your layout automatically. **TrainController™** does not require you to be an experienced programmer or computer expert in order to do this. For this reason, **TrainController™** does not require you to learn a special railroad programming language with a new syntax. Automatic operation can be accomplished by a simple point and click on the objects which are to be operated or monitored. No abstract syntax must be learned. Configuration of automatic operation is as easy as drawing a track diagram.

The number, range and complexity of activities that can be managed by one person is extended substantially. A broad range of operating flexibility is provided that extends from a completely manual operation through to a completely automatic operation (e.g. hidden yards control). Manual and automatic operations can be mixed simultaneously. This applies not only to trains on different areas of your railroad, but also to different trains on the same track, and even to the operation of a single train. The automatic processes are not bound to specific trains. Once specified, they can be performed by each of your trains. Timetable and randomizer functions increase the diversity of your model railroad traffic.

1.2 Variants of Train Control

Train control, i.e. running of model trains on a model railroad layout, is the key aspect of model railroading and hence also for **TrainController™**.

TrainController™ provides a wide range of possibilities to run your trains – from completely manual to completely automatic with a wide range of variants.

The following list provides a brief overview of the different methods to run your trains with **TrainController™**:

(1) Run trains manually, semi-automatically or automatically under full protection, blocking and routing of **TrainController™** along paths and routes, which are automatically activated by the train itself or manually by the end user during the train run. Trains are started ad-hoc, i.e. without specifying destination positions or complete paths in advance (**spontaneous runs**).
(2) Run trains manually, semi-automatically or automatically under full protection, blocking and routing of **TrainController™** by specifying the start and destination positions at any time during operation by dragging a train symbol with the mouse

from its current position to the desired destination position (**AutoTrain™ by Drag & Drop**).

(3) Run trains as before, but specify more than one start and destination position as well as other options such as scheduled waiting times, speed limits etc. at any time during operation just before starting the train (**AutoTrain™ Symbol Bar**).

(4) Run trains manually, semi-automatically or automatically under full protection, blocking and routing of **TrainController™** according to schedules, i.e. sets of options, which specify several start and destination positions as well as other options such as scheduled waiting times, speed limits etc. and which are created prior to the operating session, i.e. during configuration of the layout. Schedules can be started manually, by pressing a button, by start- and destination keys, as part of a sequential chain, automatically triggered or by timetables (**Schedules**).

(5) Run trains manually without any protection, blocking and routing performed by **TrainController™** (**Manual Train Control**).

Spontaneous Runs

This is the most handy method to run your trains under full protection and routing of **TrainController™**. Just put a locomotive on the track and call the menu command **Spontaneous Run**. The train will immediately start to move, provided that the route ahead is clear. It will then select an appropriate path and continue to travel, until it reaches a dead end or until the path ahead is blocked for another reason. At a dead end it will reverse automatically, if desired, and continue to travel to the opposite direction.

Routes can be treated in different manners for spontaneous runs. It is either possible to allow the computer to select and activate all routes requested by the train automatically. It is also possible to leave this to the human operator. In this case the train is stopped in blocks with at least one outgoing route, until one of these outgoing routes is selected and activated by the human operator.

Pros:

- Well suited to accomplish hands-on activity on your model railroad layout including protection, routing and signaling with minimum efforts.
- Easiest way to run trains under full protection and routing.
- Can be spontaneously executed at any time during operation.
- Fastest way to start a train with a **+SmartHand™** handheld under protection of the software.

Cons:

- In general human intervention or specific measures are required to prevent trains from running into tracks, where they must not go.

- Not suited for full automatic operation of the layout, without further measures, because in general human intervention is required to start the train.

AutoTrain™ by Drag & Drop

This is another very convenient method to run trains under full protection and routing of **TrainController**™. Just put a locomotive on the track and drag the symbol of the train on the computer screen with the mouse from its current position to the desired destination position. The train will immediately start to move, provided that the route ahead is clear. It will then select an appropriate path to the specified destination block and travel there, if possible. After arrival at the destination block the train is stopped.

Pros:
- Well suited to move a train spontaneously to a certain location of the layout under full control of the software, protection, routing and signaling, with minimum effort.
- Very easy way to run trains under full protection and routing.
- Can be spontaneously executed at any time during operation.
- Full control of the destination block, where the train will go.

Cons:
- Care has to be taken, that there is a possible path between the current position of the train and the desired destination block.
- Not suited for full automatic operation of the layout, because human intervention is required to start the train.

AutoTrain™ Symbol Bar

This is an extension of **AutoTrain™ Drag & Drop**. Instead of dragging a train symbol from its current position to the desired destination the path of the train and other options are specified via the **AutoTrain™ Symbol Bar**. This symbol bar provides more options than the more simple drag & drop method. The full functionality for automatic running of a train is available here. Among other options it is possible to specify more than one start and destination block, to enforce inclusion or exclusion of certain blocks, to specify scheduled waiting times during the travel, to specify operations, that will be executed during the travel, to determine, whether the train will be controlled manually, automatically or by a mixture of both, and so on. The **AutoTrain™ Symbol Bar** is also useful to predefine train runs for automatic operation of the layout.

Pros:
- Well suited to move a train spontaneously to a certain location of the layout with the possibility to apply the full range of options available for train control.

- Well suited, too, to predefine train runs for full automatic operation with minimum effort.
- Provides the full range of options available for trains running under protection and routing.
- Can be spontaneously executed at any time during operation.
- Full control of the path taken by the train.

Cons:
- Care has to be taken, that there is a possible path between the specified start and destination blocks.
- Not suited for full automatic operation of the layout without further measures, because human intervention is required to start the train.

Schedule

Schedules provide the possibility to predefine train runs in advance and in particular for full automatic operation. Unlike the other methods schedules do not require manual intervention to be started. The full functionality for automatic running of trains is available for schedules, too. Among other options it is possible to specify more than one start and destination block, to predetermine the exact paths, to specify scheduled waiting times during the travel, to specify operations, that will be executed during the travel, to determine, whether the train will be controlled manually, automatically or by a mixture of both, and so on.

Pros:
- Well suited for full automatic operation of trains without human intervention.
- Provides the full range of options available for trains running under protection and routing.
- Can be started automatically without human intervention.
- Full control of the path taken by the train.

Cons:
- Require predefinition prior to operation of the layout.

Manual Train Control

Manual train control is performed by putting a train on the track and by driving it with the throttle of the digital system, with the on-screen throttle of **TrainController**™ or with a **+SmartHand**™ handheld without taking any further measures. Although the position of the train can be tracked by the computer, the computer does not activate routes ahead of the train or take corrective action like stopping the train at a red signal. The human operator is completely responsible for routing and stopping. A train driven in this way, however, is protected against other trains running under control of the computer, while other trains are not automatically protected against this train, i.e. the human operator is responsible for ensuring, that the train operated by him in this way does not run into other trains.

Pros:
- Well suited for manual test runs and basic operation without protection, routing or signaling.
- Can be spontaneously executed at any time during operation.

Cons:
- Low security.
- No automatic routing or signaling.
- Manual control of trains only.
- Number of trains simultaneously operated this way is limited by the skills of the human operator to control and to watch several trains at the same time (usually 1 to 3 per operator).

Comparison Chart

The following chart compares the possibilities of the particular methods and their suitability for certain purposes:

Feature	**(1)** **Spontaneous Runs**	**(2)** **AutoTrain™ by Drag & Drop**	**(3)** **AutoTrain™ Symbol Bar**	**(4)** **Schedules**	**(5)** **Manual Operation**
Block Securing	Yes	Yes	Yes	Yes	No
Automatic Routing	Optional	Yes	Yes	Yes	No
Automatic Signaling	Yes	Yes	Yes	Yes	No
Train Guidance System	Yes	Yes	Yes	Yes	No
Modification by Rules	Yes	Yes	Yes	Yes	No
Automatic Consideration of Speed Limitations	Yes	Yes	Yes	Yes	No
Full functionality for automatic train operation available (e.g. Scheduled Stopovers,…)	Only **Gold**	Only **Gold**	Yes	Yes	No
Can be started with Start and Destination Keys	No	Yes	No	Yes	No
No. of possible Start Blocks per Run	1	1	>=1	>=1	-
No. of possible Destination Blocks per Run	>=1	1	>=1	>=1	-
Start without prior specification of destination Blocks	Yes	No	No	No	Yes
Preset for destination possible	Indirect	Yes	Yes	Yes	Yes
Spontaneous execution w/o prior predefinition	Yes	Yes	Yes	No	Yes
Manual Train Control possible	Yes	Yes	Yes	Yes	Yes
Transfer of Control between operator and computer according to curr. signal status poss.	Yes	Yes	Yes	Yes	No
Automatic Train Control poss.	Yes	Yes	Yes	Yes	No
Effort for Setup/Start	Minimal	Minimal	Medium	Medium	Minimal
Automatic Layout Operation without human intervention	No	No	No	Yes	No
Timetable based Operation	No	No	No	Yes	No

Table 1: Variants of Train Control

All the methods listed above can be used simultaneously and freely combined. The following modes to run trains manually, namely:

- Run trains manually with the throttle of your digital system.
- Run trains manually with the virtual on-screen throttle of **TrainController™**.
- Run trains manually and fully protected with the physical throttles of the **+SmartHand™** handheld control system

can be applied to any manually or semi-automatic operated train for any of the methods listed before. It is also possible to pass each train from manual operation to any of the automatic modes listed before and back or between the particular modes listed above at any time during operation. In short terms: there are almost no limitations.

1.3 Fundamentals of Use

B

The Overall Principle

TrainController™ supports manual, semi-automatic and automatic operation of your model railroad as well as concurrent manual and automatic operation.

Switchboards, Train and *Turntable Windows* provide the controls to operate turnouts, signals, routes, trains and turntables, etc. These controls can be operated manually by the human operator or automatically by the computer.

A human operator is normally only able to operate one or two switchboards and at most two trains at the same time. If multiple control panels or several trains are to be operated at the same time, then either support of additional human operators is required, or a computer running **TrainController™**. The software contains a special component called the *Visual Dispatcher*, which is able to take the place of additional human operators.

Like a human operator the Visual Dispatcher is able to operate turnouts, signals, routes and trains. This is called *automatic operation*.

Manual and automatic operation can be mixed like several human operators can cooperate to control the same layout.

You can also decide to do without the *Visual Dispatcher*, if you want to control everything yourself.

User Interface: Ribbon vs. Menus and Tool Bars

The user interface of **TrainController™** is either controlled via the Ribbon known from the current versions of Microsoft office or via menus and toolbars as in the previous versions of **TrainController™** (classic user interface).

Ribbon:
All the commands in the new user interface are organized into logical groups on a series of tabs called the Ribbon. The Ribbon provides an accessible interface to rich functionality that **TrainController™** provides.

When you start **TrainController™ 9** for the first time, you will notice that the Ribbon has replaced the old style menus and toolbars. Think of the Ribbon as a set of "results-oriented" tabs.

Each tab of the Ribbon is like a rich toolbar organized around a high-level task and contains commands for accomplishing that task. The **Track** tab, for example, contains the frequently-used commands for editing the track layout in the switchboard, while the **Operation** tab includes commands for operation of the model railroad. When a tab is selected, the commands associated with it become visible in the upper part of the screen.

The commands on each tab are ordered into groups to further organize the available features. For example, the **Edit** tab includes groups of commands called **Undo**, **Clipboard**, **Train**, **Schedule**, etc.

You will find all the traditional file menu commands under the **File** menu button left of all tabs in the top left corner of the Ribbon.

Classic User Interface:
The **File** menu also provides a command, which allows to change to the classical user interface with menus and tool bars known from the previous versions of **TrainController™** for users who prefer the old style.

Quick Access Toolbar:
Everyone has a set of commands they use most often, so it is possible to add commands to the **Quick Access Toolbar**, a small toolbar positioned in the title bar of the application window. The **Quick Access Toolbar** provides a location for heavily-used commands that need to be available with one click (regardless of the current Ribbon state). You can add any item to the **Quick Access Toolbar**. From then on you can access the command from wherever you are in the application, and avoid situations where you need to repeatedly switch between Ribbon tabs to accomplish repetitive tasks.

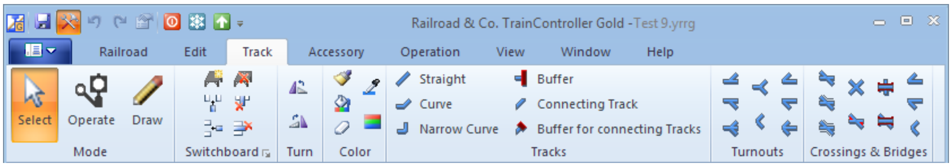


Diagram 50: Ribbon

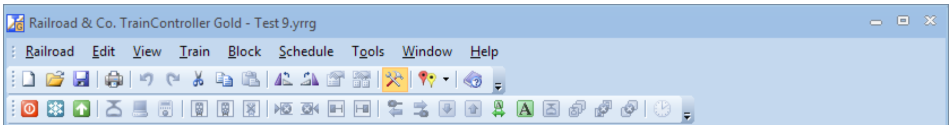


Diagram 51: Classic User Interface

User Interface Design

The user interface of **TrainController™** can be extensively customized to your personal needs and taste.

This begins with the overall layout of the user interface. The user interface can be displayed by applying different visual styles. Among others the following styles are available:

- Several Office 2013 styles
- Several Office 2010 styles
- Several Office 2007 styles
- Visual Studio 2015, 2013 and 2012
- Visual Studio 2010, 2008 and 2005
- Windows 10, Windows 8 and Windows 7
- Native XP
- Office 2003
- Classic Office 2000
- Railroad & Co. 9, 8 and 7
- Several extra styles
- A custom style with the possibility to adjust the primary colors of the user interface to personal taste.

Feel free to select the style, that fits best your personal taste.

Window Handling

The particular functions of **TrainController™** are represented in different windows. Normally you will open several windows for the same model railroad layout. If you want to split the track diagram of your layout into two or more switchboard windows or if you want to control different trains with different train windows then you can open additional windows for the same layout.

Additional windows (switchboards, trains windows, clock, etc.) are opened and closed through the **Window** tab of the software. Each window can be made invisible at any time without loss of data.

Diagram 49 shows an open layout file that contains several windows. The file contains among others, a switchboard window, a train window, a clock window and a Dispatcher window for automatic operation.

The general visual design of all windows is harmonized and the handling of all windows is consistent. The size of all windows is variable and can be adjusted to your personal taste.

Each window can appear in one of the following states:

- Docked to one of the borders of the main window.
- Docked to another window.
- Floating at any location on the computer screen; individually or grouped/docked together with other windows.
- Tabbed with other windows – as one of several tabbed documents in the background of the main window or together with other windows in a floating or docked frame.
- Auto-Hidden while not active with quick access via a button on any side of the main window.
- Floating windows can be maximized or half-screen docked to the boundaries of each computer screen.

The possibility to group related windows together in **TrainController™**, either docked or tabbed, in the main window or in a floating frame somewhere on the computer screen, opens interesting possibilities. It is possible, for example, to arrange a set of related windows for control of one part of the layout together in one group and to arrange another set of related windows for another part of the layout in another group. Such group of related windows can be then moved, resized, hidden, restored and even docked and tabbed together with other groups of windows just like one single window, which makes it very convenient and effective to manage sets of windows, that belong together. An example of such related windows is a switchboard combined with a dispatcher window, that shows just the block diagram of this switchboard (**TrainController™** **Gold** allows you to create multiple block diagrams for multiple switchboards and to open more than one dispatcher window: more about this later). No matter how many windows you need to open to represent your entire layout, you will find a window arrangement, that fits your needs and personal taste.

TrainController™ makes docking of windows intuitive and easy by showing docking markers for each window, that is currently being dragged over the computer screen. The docking markers intuitively indicate, where to move the mouse to dock the dragged window to the desired location. By moving the mouse to a docking marker an additional docking outline provides a clear preview of the docking effect. Thanks to this feature, which has been borrowed from state-of–the-art professional software development systems there is no more puzzling, where a window will be finally docked.

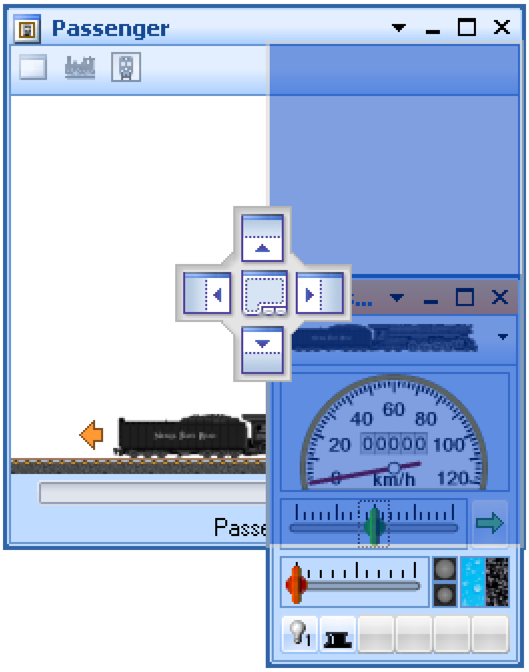


Diagram 52: Docking a Train Window to the right of a Traffic Control

Floating windows can be maximized or half-screen docked to the boundaries of each computer screen. Maximizing and docking as well as later normalizing and undocking can be performed with the mouse by drag & drop (similar to the Aero Snap feature known from the current versions of Microsoft Windows).

In this way a floating switchboard and a floating dispatcher window, for example, can be easily arranged on top of each other by drag & drop with the mouse, each covering half of the computer screen.

TrainController™ stores the window arrangement individually for each project. Especially those users, who work with different data files for different projects will appreciate the possibility to create and to save an individual arrangement of windows for each particular project.

Even for the rare case, that the windows on your computer screen are misaligned and you don't know how to return to a consistent state, provisions have been made: with a specific menu command it is possible to load the current data file once more with a default window status, which can be used again as a starting point for an individual arrangement.

Window Customization

The layout of those windows, which contain the most comprehensive data or which are used most frequently (the switchboard, dispatcher and train window) can be highly customized to personal needs and taste.

You can play with all settings without any risk, because in **TrainController**™ all customizable windows provide the possibility to reset the settings to factory defaults.

Customization of Menus, Tool Bars and Keyboard Accelerators

It is also possible to customize the content of menus and tool bars of the classic user interface as well as to change keyboard accelerators for both kinds of user interface.

New menus and tool bars can be created, commands can be added or removed from menus and tool bars and existing commands can be changed. It is possible to create new menu and tool bar symbols for commands, that do not have a symbol associated with it by default, or to change existing icons with a built-in icon editor.

It is furthermore possible to display all menu and toolbar icons in large size.

Keyboard accelerators can be changed. It is also possible to assign keyboard accelerators to commands, that do not have a keyboard shortcut associated with it by default.

File Handling

The complete data of your model railroad layout is stored in one single file on the hard disk on your computer. This file is called *project file*. You can create as many layout files as you like. For example, this is useful if you have different model railroad layouts or if you want to try and store several variants of the same layout file.

The layout file contains the complete description of your layout, i.e. all track diagrams, routes, trains and all data specified for automatic control of the layout, if any. The file also contains the current status of your model railroad layout, i.e. the current state of all turnouts and signals, the status and positions of trains, and the current time of the clock, etc. Finally the current layout and settings of the user interface are stored in the layout file, too. Please note that all data of the same layout is stored in the same layout file.

Layout files are created, opened and stored through the **Railroad** tab of the software.

Please note the difference between *windows* and *files*. Only one layout file can be opened at the same time and the layout file contains all data and windows that belong to the same layout. The windows belonging to the same layout are contained in one layout file.

TrainController™ Gold provides the option to store the status of the user interface in a separate file, apart from the project file; and to load the status from this separate file.

The status of the user interface concerns all existing windows, whether they are visible or not, their size and position, all customization options for toolbars and menus, and so on.

This option takes effect, when the project file is being stored or loaded. If this option is set, then the status of the user interface is stored twice – once in the project file as usually, and once in the mentioned separate file. When the file is being loaded, then **TrainController™** loads the status of the user interface from the separate file, if available, otherwise from the project file.

This is useful if the project file is edited on several computers. After storing the project file and the status of the user interface in a separate file on computer A and moving the project file (without the separate file) for the first time to computer B, the recent status of the user interface on computer A will be loaded first. This happens, because the status of the user interface (here the status on computer A) is also always stored in the project file and because no separate file is available in this moment on computer B. After storing the project file and the status of the user interface in a separate file on computer B and moving the project file (without the separate file) back to computer A, the original status of the user interface on computer A will be loaded now, because the separate file is still available here.

In this way it is possible to adjust the status of the user interface individually to several computers (e.g. their individual monitor configuration), to move the project file back and forth between several computers and to load on each computer that status of the user interface, which fits best to the particular computer. And if the project file is provided to a new computer, where this file was never loaded, then the most recent status of the user interface of that computer, where the project file was most recently stored, will be loaded on the new computer. This causes the same result, as if this optional feature is not used, which is the default case.

Edit Mode

All changes to be made to the content of your layout file require that **TrainController™** is running in *edit mode*. While edit mode is turned on you can change data, add new data or delete data, that is no longer needed. During operation edit mode is turned off. This protects your data during operation against unintentional changes.

Edit mode can be turned on or off at any time. When edit mode is turned on all automatic operation of your layout, if any, is stopped.

! **In order to input new data as well as to edit or delete existing data edit mode must be turned on.**

Printing

TrainController™ provides very extensive and flexible features to print the data contained in a layout file. It is possible to print a single item on a single piece of paper; but it is also possible, to arrange individual and comprehensive print jobs containing selected items or to print the entire data contained in a layout file.

A print job can contain one or more of the following items:

- Switchboard or track diagrams.
- Block and/or schedule diagrams of the Dispatcher
- Lists of objects grouped by type (e.g. a list of all signals and sorted by name or digital addresses)
- Lists of objects grouped by other aspects (e.g. all turnouts contained in the same switchboard)

- Comprehensive object details

The sorting criteria and order in object lists can follow different aspects (e.g. sorting by name, digital address or date of most recent modification).

Users, who are familiar with HTML and cascaded style sheets (CSS) can even customise the layout of the printed data to personal needs.

Further Steps

In order to control your model railroad layout with **TrainController™,** you need one or more of the digital systems listed in the previous section. These digital systems are connected to an available serial, USB or Ethernet port of your computer.

In the following it is assumed that you are already familiar with the usage of your digital system. For details regarding your digital system, please refer to the documentation provided by the manufacturer.

To create a computer control system with **TrainController™** the following steps are usually performed:

- Creation of *Switchboards* containing control panels based on track diagrams of specific areas
- Entering the data of existing *engines* and *trains*
- Creation of automatic *schedules* with the *Dispatcher*

It is not necessary to perform all steps listed above to control your model railroad with **TrainController™** For model railroad clubs you may only need to arrange the *Switchboards*. In this case, one person may be responsible for controlling the traffic by operating turnouts, signals and routes while other persons are using handheld throttles to control the trains. If you have an existing control panel, then you can use the *Train Windows* independently to take advantage of the realistic train control features of the program.

Switchboards

Usually, you will start configuring **TrainController™** by creating one or more *Switchboards*. Like in real railroads, *Switchboards* are control panels to be used to control turnouts, signals, routes and other accessories like uncouplers or crossing gates. *Switchboards* are made using symbol elements representing *tracks, turnouts, crossings, signals, accessories* and more.

Switchboards are usually created for those parts of the layout that contain turnouts, and signals. Examples of such areas are stations, sidings or hidden yards.

You first insert track elements into the *Switchboards* to create a track diagram that represents the track plan of your entire layout, the main station or any yard etc. For small and medium size layouts it is recommended that you create only one switchboard. This main switchboard is used as a base for quick and easy setup of automatic operation. In the case of larger and more complex layouts you will probably create a separate *Switchboard* per yard. You can create as many additional switchboards as you like.

After you have placed all tracks, turnouts, crossings and bridges in the correct positions, you specify the *digital addresses* of your turnouts.

When this has been done, you are able to control the turnouts of your model railroad with **TrainController™** and your computer.

Your model railroad may contain not only tracks and turnouts but also signals and other accessories. If so, the next step is placing the *signals* at the appropriate locations of your control panel. **TrainController™** provides symbols for *two, three* and *four aspect signals*. Uncouplers, lights, crossing gates or other accessories can be controlled with symbols representing *push buttons, toggle switches* or *on-off switches*.

After you have placed all the signals in the correct positions, you specify the *digital addresses* of your signals and other accessories.

Once you have specified the *digital addresses* of your signals and other accessories, you are able to control these objects manually with **TrainController™**, also.

Text elements can be inserted at arbitrary positions to label your control panel. Images can be placed in the Switchboard as well.

If you want to set up automatic operation for your trains or display train positions in the switchboard, then you will need to insert *blocks* in your switchboard, that represent the blocks of your model railroad.

Switchboards provide even more features for controlling, monitoring and semi-automatic operation. These elements are discussed in more detail later.

Train Windows

The *train window* enables the operation of your *engines* and *trains*. To control several trains simultaneously, you can open as many train windows on your computer screen as desired.

After the selection of the current engine, or train, in the train window, you are able to control the train and monitor its operations with the control instruments.

To operate a certain engine on your layout, create a *Train Window* and specify the digital address of the engine. You do not have to bother with all other options until you want to add more realism to the operation of your trains.

The Visual Dispatcher

The *Visual Dispatcher* is a component that makes large scale railroad operations manageable by one person, matching operations found on the largest club layouts. *Engines* and *trains* can be operated manually or automatically.

Like a human operator must know the overall structure of the model railroad layout the *Visual Dispatcher* needs to know this, too. This structure is represented by a diagram that contains blocks and routes and the track connections between them. This diagram is called *main block diagram* of the layout. The main block diagram describes the track layout of your entire model railroad in rough outline.

The *Visual Dispatcher* manages traffic flow using a blocking system. Blocking ensures that trains do not collide and supports the tracking of train positions. For this purpose, the railroad layout is divided into virtual, logical blocks. That means, you define blocks at locations where traffic control will take place (e.g. scheduled stops in a station).

Usually each track in a station or hidden yard, each siding and appropriate sections of the connections between two yards will form a block.

Dividing the layout into logical blocks does not necessarily imply, that your blocks must be electrically insulated. **TrainController™** does not require such electrical insulation. Whether your blocks must be insulated or not depends solely on the hardware used.

Blocks and connecting *routes* are arranged graphically in the *main block diagram* to specify on which paths trains will travel. *Schedules* describe train movements, i.e. <u>how</u> trains travel. This includes start and destination blocks, scheduled waits, speed limits, etc.

AutoTrain™, an outstanding feature of **TrainController™**, allows you to start trains automatically without the need to define a schedule before or to create new *schedules* while playing with your trains – programming while playing!

Trains can run under full manual control, in which case the human operator will be responsible for obeying the block signals set by the *Dispatcher*; or under full control of the computer; or even with an intermediate level of automation.

For shunting special types of schedules are provided.

Schedules and *timetables* can be arranged with a broad range of flexibility. Since a timetable can be created for each day of the year, up to 365 timetables can be used.

Randomizer functions increase the diversity of your model railroad traffic.

2 The Switchboard

2.1 Introduction

B

By default **TrainController™** displays a *switchboard* in the main window of the software. Additionally it is possible to display as many additional s*witchboards* on your computer screen as desired. A switchboard represents a track diagram control panel of specific parts of your model railroad, i.e. those parts that contain turnouts, and signals. Examples of such areas are stations, sidings or hidden yards. If you intend to control a large layout consider creation of a separate panel for each yard.

Switchboards are used to operate the *turnouts*, *signals*, *routes* and other *accessories*, like crossing gates, on your model railroad. Switchboards are created using different symbol elements that are arranged in rows and columns.

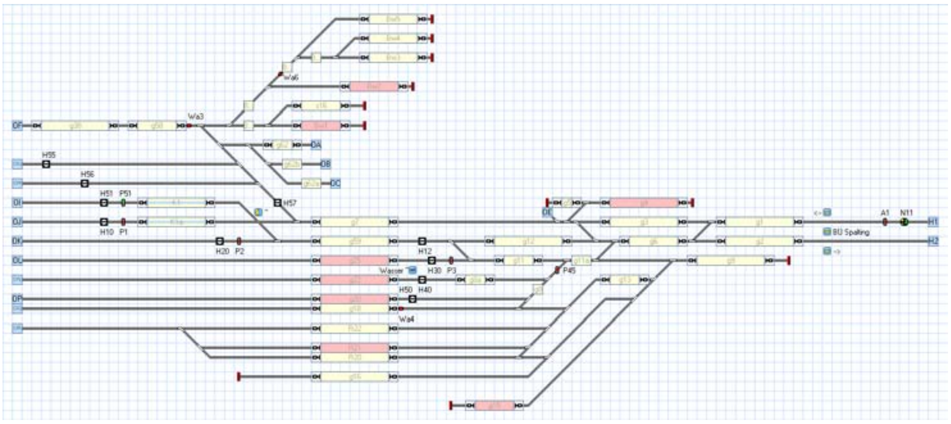


Diagram 53: Switchboard Example

Several types of symbol elements are provided to facilitate the creation of switchboards:

- *Track elements* are used to represent the tracks of your model railroad such as *straight* and *curved* tracks.
- *Turnout elements* are provided as special track elements to enable operation of different types of turnouts like *normal*, *triple* or *slip turnouts*.
- *Signal elements* are used as *two*, *three* or *four aspect signals* to represent and to operate the signals on your model railroad.

- *Accessory elements* of several types – *push buttons*, *toggle switches* or *on-off switches* – operate additional accessories such as uncouplers or lights or can be used to trigger other actions like playing of sound files.
- *Block Symbols* can be used for quick setup of automatic operation and display of train positions.
- *Text elements* can be used as labels, e.g. for tracks in stations.
- *Images* can be inserted into your track diagrams to display landscapes, buildings, streets or other objects of your model railroad.

The following elements can be additionally added to switchboards in specific cases:

- *Route elements* enable manual route operation and locking on layouts.
- *Indicator elements* are provided as *contact indicators* or as more intelligent *flagman indicators* to allow monitoring of the model railroad, creation of semi-automatic and automatic control mechanisms, or tracing of train positions.
- *Virtual Contact Indicators* can be used to reduce the number of track contacts which are needed for automatic operation.

The following steps are performed to create a full functioning switchboard:

- Drawing the track diagram of the related area
- Connecting turnouts and signals
- Inserting block symbols
- Placing signals and accessory elements
- Adding text labels and images

The following steps are mainly performed in the switchboard in those cases where it is desired, to monitor the traffic on the layout to a certain degree, or to achieve semi-automatic operation of the layout without running the *Visual Dispatcher*. If the *Visual Dispatcher* is being used, the following steps are performed in the *Visual Dispatche*r rather than a switchboard.

- Creating manual routes
- Inserting contact indicators
- Arranging semi-automatic control mechanisms

2.2 Size and Appearance

B

For each switchboard it is possible to customize the size, i.e. the numbers of rows and columns, and the appearance individually.

The elements in the switchboard are arranged in a grid based system consisting of rows and columns (see Diagram 53).
The individual preferences with regard to the appearance of track diagram control panels are very different. For this reason **TrainController™** provides many options to customize the appearance of the control panels individually to your convenience and taste. There are options to select the background and track colors, to apply 3 dimensional light and shadow effects to background and tracks and to select the colors in which the states of certain elements are highlighted.

The possibilities are virtually unlimited, a few examples are given below:

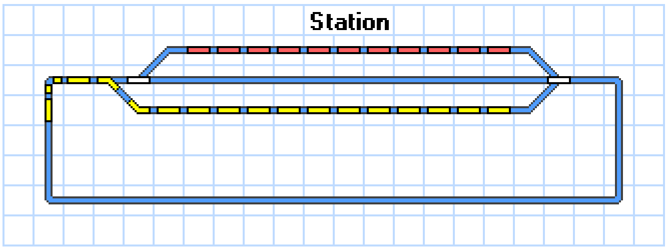


Diagram 54: Standard Format

Diagram 54 shows the standard format for display of control panels. A few examples of the unlimited possibilities to customize the appearance are given in the following:

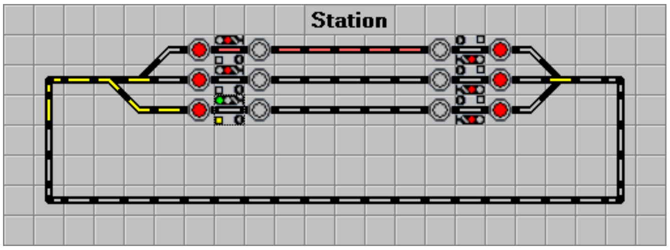


Diagram 55: German Style

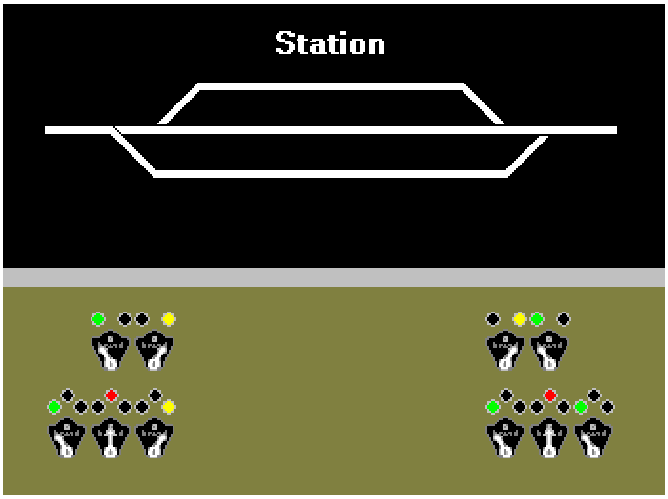


Diagram 56: US CTC Panel Style

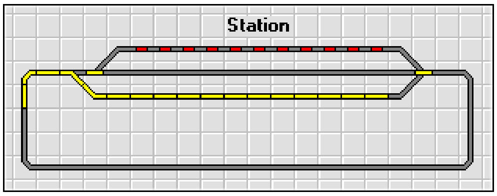


Diagram 57

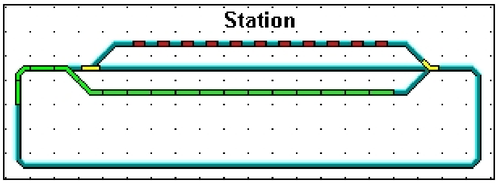


Diagram 58

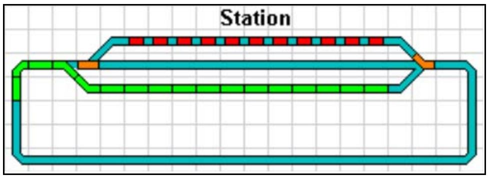


Diagram 59

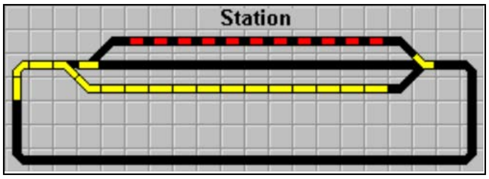


Diagram 60

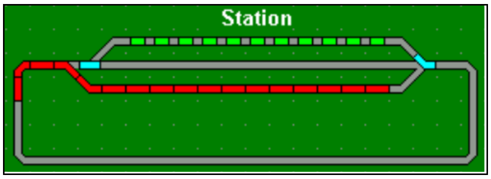


Diagram 61

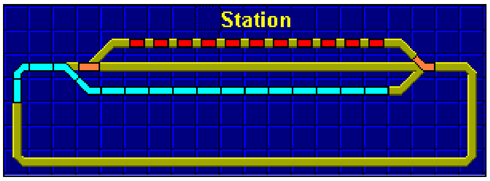


Diagram 62

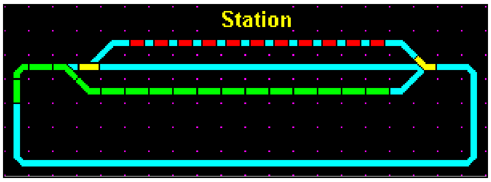


Diagram 63

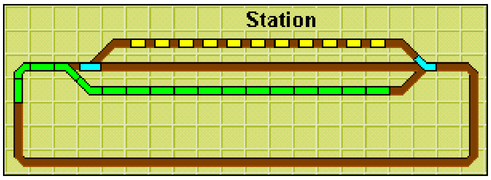
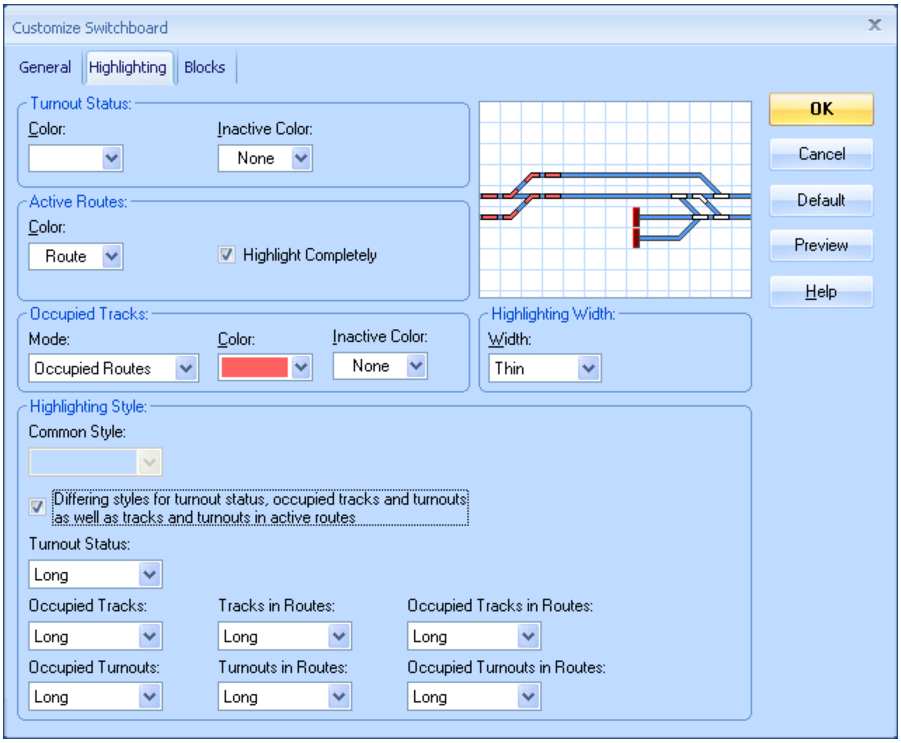


Diagram 64

Among others the following options are provided:

- Six highlighting styles can be applied generally or differing for plain tracks vs. turn-outs, occupied vs. non-occupied track symbols as well as track symbols in active routes. The possibilities to apply different styles and colors to different highlighting aspects provide virtually countless layout combinations and allow reproduction of almost each highlighting style of prototypical plug panels.

- The highlighting of occupied track elements can be turned off globally; or it can follow the activation of routes, i.e. only those track elements are highlighted, which are contained in a currently active route; or the highlighting can be controlled by individual assignments to indicators as in previous versions. The highlight color of occupied track sections can be controlled by the reserving train, if any, as in previous versions, or by the color of the occupied indicator or by specifying a constant color value.
- Active routes can be highlighted with individually specified colors or with the color of the reserving train, if any, or with a color, that is common for all active routes.
- Switchboard symbols can be displayed in five different sizes ranging from 12x12 to 28x28 pixels per symbol / switchboard cell.
- In **TrainController™** **Gold** it is possible to create custom switchboard symbols for signals, push buttons, on/off switches, toggle switches and routes with an integrated bitmap editor and to assign such custom symbols individually to each according switchboard object. Custom switchboard symbols can be transferred between different data files by export and import.


Customize Switchboard

General | Highlighting | Blocks

Turnout Status:
Color:
Inactive Color: None

Active Routes:
Color: Route
☑ Highlight Completely

Occupied Tracks:
Mode: Occupied Routes
Color: 🟥
Inactive Color: None

Highlighting Width:
Width: Thin

Highlighting Style:
Common Style:
☑ Differing styles for turnout status, occupied tracks and turnouts as well as tracks and turnouts in active routes

Turnout Status: Long
Occupied Tracks: Long
Tracks in Routes: Long
Occupied Tracks in Routes: Long
Occupied Turnouts: Long
Turnouts in Routes: Long
Occupied Turnouts in Routes: Long

Diagram 65: Customization of Highlighting in the Switchboard

2.3 Drawing the Track Diagram

B

Creating a *switchboard* starts with drawing the track diagram of the related station, yard or sidings. Using the available *track elements* a schematic image of the tracks of the area is drawn on the computer screen.

The following track elements are available:

- *Straight*
- *Normal* or *narrow Curve*
- *Bumper*
- *Diagonal* or *vertical crossing*
- *Diagonal* or *vertical bridge*
- *Left-* or *right-hand turnout* as well as well as *Wye Turnout*

- *Three way turnout*
- *Single* or *double slip turnout*

You can draw your track diagram in various ways. First, though, the edit mode of the switchboard must be turned on.

Then you have the following options:

- **Inserting single elements:** You can draw your track diagram by inserting single elements successively.
- **Drawing a straight track section with the mouse:** You can draw a straight track section consisting of more than one element very quickly by dragging the section you want to draw with the mouse.
- **Drawing the track diagram with the keyboard:** An additional and fast way to draw the track diagram is the use of the numeric keypad of your computer.

These methods are explained in detail in the help menu.

To adjust the track elements precisely, additional edit facilities such as *copy*, *move* or *turning* of track elements are available.

Space-Saving Turnouts

In addition to the track elements described above the following track elements are available in **TrainController™ Gold**:

- Space-saving turnouts as *left-* and *right-hand turnouts*, *wye turnoust* and *three-way turnouts*
- *Connecting tracks* for space-saving turnouts
- *Left*, *right* and *symmetric crossings* for use with space-saving turnouts and their connecting tracks.

Space-saving turnouts need less space in the switchboard in certain situations, e.g. in the case of crossovers. Furthermore the look of certain prototypical switchboard panels, that use such symbols, too, can now be replicated more realistically.

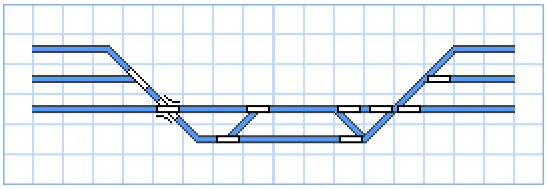


Diagram 66: Normal and Space saving Turnouts

Diagram 66 shows two identical situations, on the left side drawn with normal turnouts and on the right side drawn with space saving turnouts. The left part requires more space, even though a double slip turnout can be used there.

This diagram illustrates also, that normal turnouts and space saving turnouts can be combined in the same track diagram without any problems.

TrainController™ Gold leaves the choice to you, which type of turnout symbols you use, the normal, the space saving or both. This enables you to create a track diagram layout, that fits best to the available screen space, to the prototypical panel style, that you may want to replicate, or just to your personal taste.

Smart Gates and Crossing Gates

Gates are smart switchboard symbols, which can be used for automatic control of the gates of engine sheds or all other gates, which can be passed by trains.

To work properly, a gate must be attached to a straight element or a curve element in the switchboard. The gate is automatically opened, if a route passing this track symbol is activated. The gate is automatically closed again, if this route is deactivated.

By using conditions it is possible to influence the operation of gates furthermore.

Crossing gate symbols can be used for automatic control of crossing gates. Crossing gates can expand over more than one switchboard cell. To work properly, the switchboard cells covered by a crossing gate must contain at least one straight element or curve element. The crossing gate is automatically closed, if a route passing one of these track symbols is activated. The gate is automatically opened again, if the last active route passing the crossing gate is deactivated.

The latter covers proper control of multi-track railway crossings. Even if two trains pass the railway crossing at the same time, the crossing gates remain closed, until the last train has left the crossing and cleared the route across the crossing.

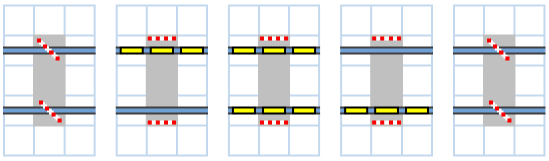


Diagram 67: Crossing Gates and multi-track Railway Crossing

The above picture series shows the crossing gates at a multi-track railway crossing, which are automatically closed, when the first route passing over the railway crossing is activated, and opened again, when the last route is deactivated.

Crossovers

Crossover symbols can be used in **TrainController**™ **Gold** to form crossovers between parallel straight track sections. Crossover symbols are always arranged in pairs.

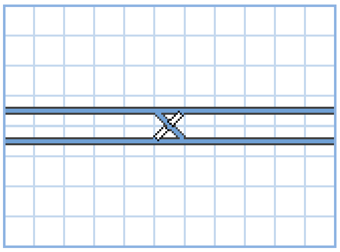


Diagram 68: Crossover formed by two crossover symbols.

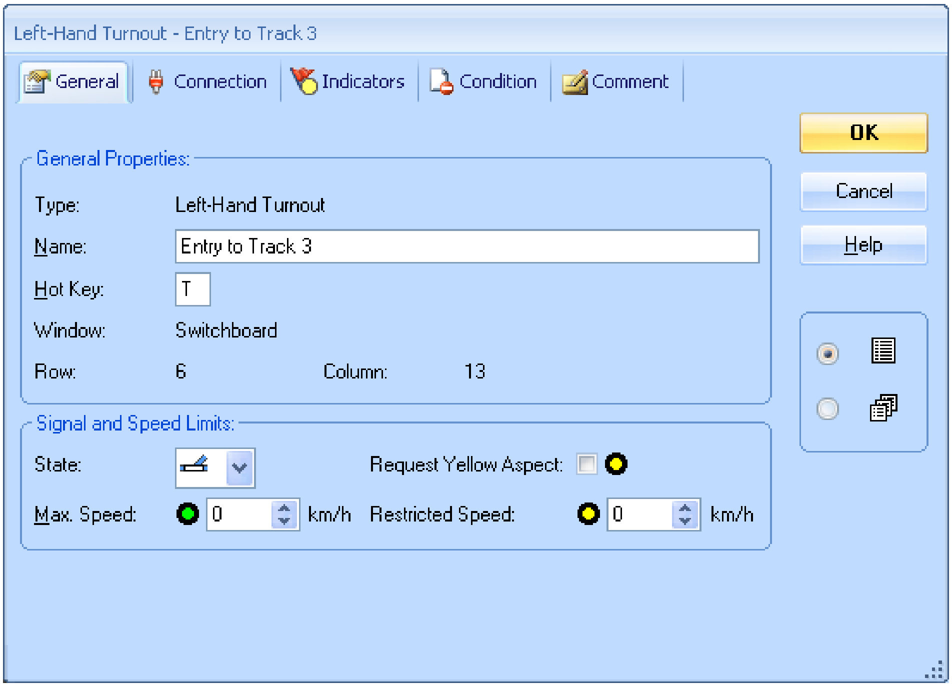
2.4 Connecting the Turnouts

B

When the track diagram is completely drawn, the *digital address* of each turnout, or slip turnout, must be specified. This is the address of the stationery decoder or output device controlling the specified turnout. If several *digital systems* are used, then the system that controls the particular turnout must also be selected.

This is done by selecting the turnout element and using the **Properties** command of the **Edit** tab.

For each turnout you can specify a *name*. This is useful in identifying the turnout when it is referred to later.

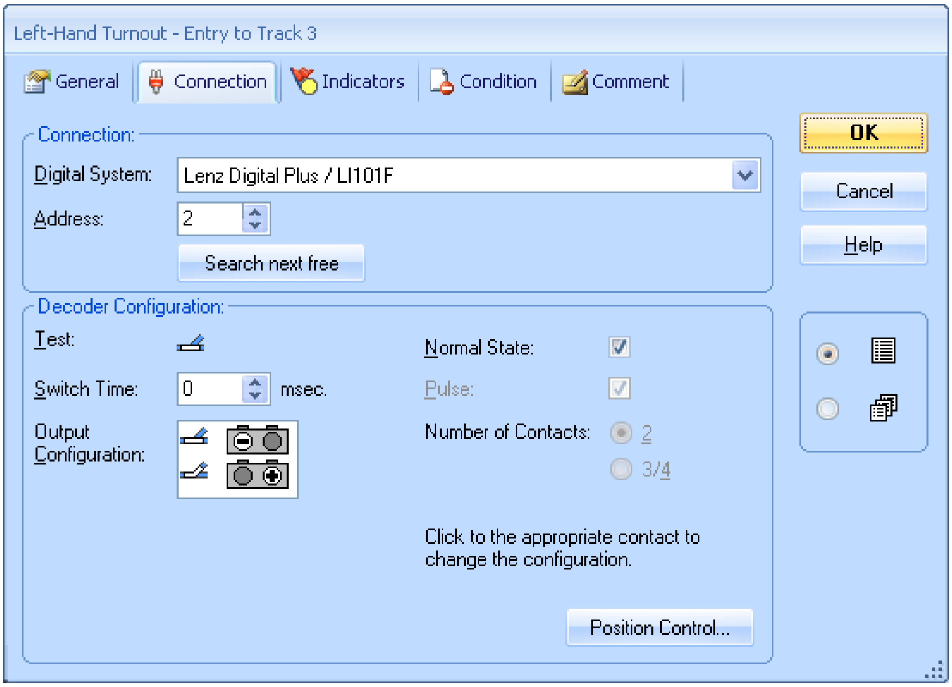


Type: Left-Hand Turnout
Name: Entry to Track 3
Hot Key: T
Window: Switchboard
Row: 6 Column: 13
State:
Request Yellow Aspect: ☐
Max. Speed: 0 km/h
Restricted Speed: 0 km/h

Diagram 69: Specifying the name of a turnout

! Turnouts with more than two states such as *three way turnouts* or *single* or *double slip turnouts* with four solenoids occupy two digital addresses. For simplicity, **TrainController™** always uses the <u>subsequent address</u> by default.

In **TrainController™ Gold**, however, it is also possible to specify two non subsequent addresses, if desired.


Left-Hand Turnout - Entry to Track 3

Connection:

Digital System: Lenz Digital Plus / LI101F

Address: 2

Decoder Configuration:

Test:

Switch Time: 0 msec.

Output Configuration:

Normal State: ✓

Pulse: ✓

Number of Contacts: [✓ 2] 3/4

Click to the appropriate contact to change the configuration.

Diagram 70: Specifying the digital address of a Turnout

For *double slip turnouts* it is possible to specify whether the turnout is operated by two our four solenoids.

Depending on the digital system being used, or the way the turnout is wired, the turnout element in the switchboard may not reflect the correct position of the real turnout. To correct this problem, you do not need to rewire your turnout. The software allows you to setup the configuration of the decoder outputs in any way as required to operate the turnout.

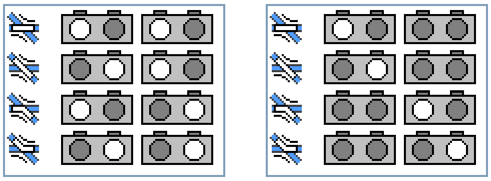


Diagram 71: Decoder Configurations for a double slip turnout

The image above displays two possible configurations for a double slip turnout. In both cases it is assumed that the turnout is operated by two double-solenoid devices with four solenoids in total. For this reason the turnout occupies four output contacts of an accessory decoder. The left image displays a situation, in which both double-solenoid devices must be operated in order to throw the turnout. The right image displays a situation, in which only one double-solenoid device is to be operated to throw the turnout. The bright circles represent the contact outputs of the accessory decoder which are turned on in order to throw the turnout to the corresponding state. The dark circles correspond to the decoder outputs, which remain turned off during operation of the turnout.

For certain digital systems the bright circles are drawn in a color or show an additional sign, which reflects the color or label of the key, that is to be pressed on the keyboard or handheld of this digital system in order to activate the related contact output. If you are familiar with the operation of a certain turnout with your keyboard then these additional markings help you to map the keyboard operation to the correct configuration in the software.

These images display only two possible situations. The decoder outputs can be configured very flexibly as required to operate the turnout.

2.5 Signals and Accessories

After completing the track diagram, the next step is to place the signals in the diagram, as well as the accessory elements such as operating lights, uncouplers or other accessories.

The following elements are provided:

- *Two, three* and *four aspect signals* of different styles
- *Push buttons, toggle switches* or *on-off switches* to operate your accessories

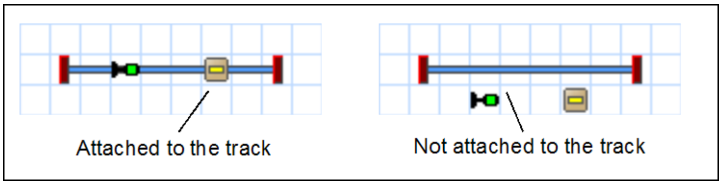


Diagram 72: Attaching signals and accessories to the track

If you want to visualize that a signal or accessory element located in the track diagram is associated with a piece of track on your railroad (for example a signal that controls a track section or a push button that operates an uncoupler), you can *attach* this element to a track element. For the operation of the signal or accessory element, however, it is not important if it is attached to the track or not. The purpose of the attachment is only to visualize the relation between the signal, or accessory element, and the corresponding track.

Signals

Signals are available in different styles. The styles are light or mechanical signals as used by the Deutsche Bundesbahn or light signals of international railroad companies. Additionally, different styles are available for home and distant signals.

The purpose of these different styles is only the indication in the switchboard. For the operation of a signal, it is not important if you select an American or German light signal, or if you select a light or mechanical signal. Feel free to use the style that best fits the respective signal on your model railroad.

It is important for the operation of the signal that you differentiate between the symbol for a *two, three* or *four aspect signal*.

For each signal a special style can be selected, that allows rotated display of the signal symbol. With another style it is possible to display multiple signals in adjacent switchboard cells as if they were fixed to the same mast. With two four aspect signals combined in this way it is possible to display 16 different signal aspects.

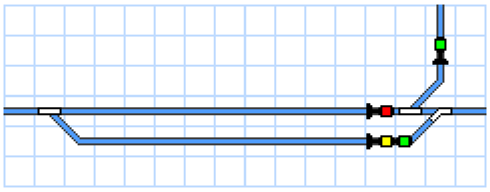


Diagram 73: Rotated and multiple signals

Diagram 73 shows signal symbols that are rotated according to the track elements they are attached to. There is also a multiple signal showing a green over a yellow light. This multiple signal is actually composed by two separate signal symbols. One of them uses a special style that lets it look like it is mounted on top of the other signal on the same mast.

Accessories

Accessory elements are used to control accessories like uncouplers, light or crossing gates. They are available in three different types:

- *Push buttons* are used to turn on a certain contact temporarily – e.g. to control an *uncoupler*
- *Toggle switches* are used to change permanently between two related contacts
- *On-off switches* are used to turn on and off a certain contact permanently – e.g. to turn on and off lights

Push buttons and on-off switches can not only be used to operate a specific contact, but also to control other elements. It is possible, for example, to operate a group of related turnouts and signals with a single click on a push button. More details are outlined in section 14.4, “Operations”.

Connecting Signals and Accessories

Signals and accessories are connected to their real counterpart on the model railroad much like the turnouts as outlined in section 2.4, “Connecting the Turnouts”. This is also done by selecting the symbol of the signal, or accessory, in the switchboard and using the **Properties** command of the **Edit** tab.

For push buttons and on-off switches, which will be used to control other elements, a set of *operations* instead of a digital address needs to be specified. More details are outlined in section 14.4, “Operations”.

2.6 Text Labels

You can place text labels in your control panels. For this purpose, t*ext elements* are provided and can be used to label turnouts, signals or tracks.

2.7 Self-provided Switchboard Symbols and Images

Self-provided Symbols

In **TrainController™ Gold** it is possible to create switchboard symbols with an integrated symbol editor yourself and to display them in the switchboard. Such symbol must only be drawn once and can be displayed many times in the switchboard.

Switchboard symbols are inoperable and mainly used for small switchboard graphics and icons, which extend the default stock of predefined switchboard symbols.

Symbols can be arranged in the background, i.e. behind the track diagram, or in the foreground of the switchboard. Symbols in the background can be covered by track elements. Symbols in the foreground can cover track elements.

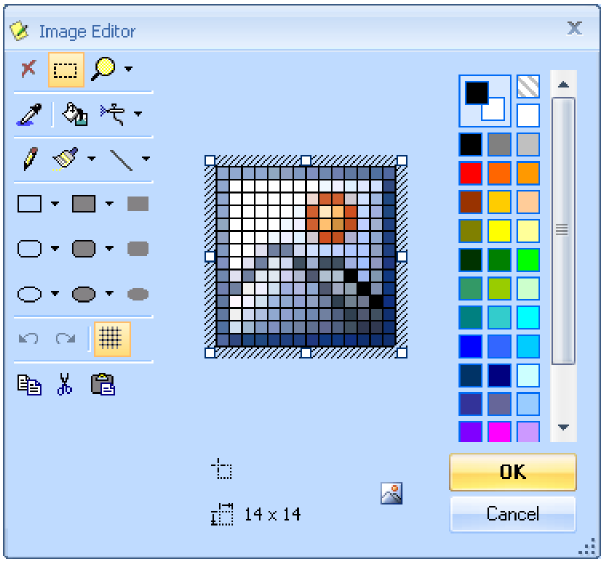


Diagram 74: Creation of self-provided switchboard symbols with the built-in image editor

Self-provided symbols can be transferred between different data files by export and import.

Images

It is possible to display images stored in external bitmap, gif or jpeg files in your switchboard. The following options are provided:

Images can be arranged in the background, i.e. behind the track diagram, or in the foreground of the switchboard. Images in the background can be covered by track elements or by images laying in the foreground. Such images can be used to display landscape structures like meadows or rivers. Images in the foreground can cover track elements and can be used to display buildings, bridges or tunnels.

It is additionally possible to fade out portions of an image, i.e. to draw portions "transparently". This is useful if images with irregular shapes are drawn. This is done by drawing the parts of the image, which will be drawn transparently, with a particular color, which is not used elsewhere in the image.

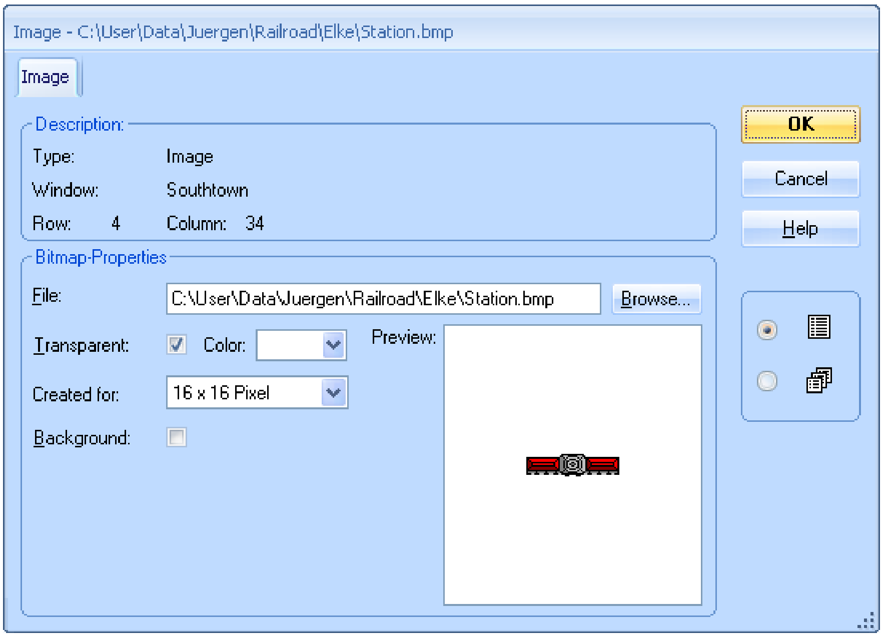


Image - C:\User\Data\Juergen\Railroad\Elke\Station.bmp

Description:
Type: Image
Window: Southtown
Row: 4 Column: 34

Bitmap-Properties
File: C:\User\Data\Juergen\Railroad\Elke\Station.bmp
Transparent: ✓ Color:
Created for: 16 x 16 Pixel
Background: ☐

Diagram 75: Arranging an image

2.8 Highlighting occupied track sections

By default an occupied track section is automatically highlighted in the switchboard, when a route (see section 5.2, “Blocks and Routes”) is currently active, which passes the track section

It is also possible to assign a set of track elements to each occupancy indicator (see chapter 4, “Contact Indicators”). These track elements are highlighted, when the indicator is turned on. In this way it is for example possible to highlight occupied track sections in the switchboard.

If there is a known train located in the track section, when the indicator is turned on, the color of this train can be used for highlighting. Otherwise the track elements are highlighted in the same color as the indicator or in an explicitly specified color.

This kind of occupancy indication only serves visual purposes, it does not affect automatic operation of the layout.

2.9 Displaying Train Names and Symbols in the Switchboard

The names or symbols of trains located in a certain block are displayed in the switchboard in the block symbols. These are symbols, that are associated with blocks. Block symbols are able to show the status of the related block as well as an image and/or the name of the train, that is currently located in the block, if any. For further details please refer to section 5.5, “Train Tracking”.

Block symbols are also used for quick and easy setup of automatic operation of your trains. These symbols mark the location of the blocks of your layout in the track diagram.

2.10 Using the Computer Keyboard as a Control Panel

For the convenient operation of *turnouts, signals*, *accessories* and *routes*, it is possible to specify a *hot key* on your computer keyboard. A hot key is one of the keys A to Z or 0 to 9. An element, to which a hot key has been assigned, can be operated conveniently by pressing its hot key.

Hot keys are assigned in the dialog tab displayed in Diagram 69, page 103.

3 Train Control

3.1 Introduction

The Train Window

B

Train windows can be used to operate trains manually with the mouse or keyboard of the computer or to watch the status of running trains during operation.

Train windows contain various controls and instruments, that are used to operate or to monitor each train.

A sample Train Window is displayed below:

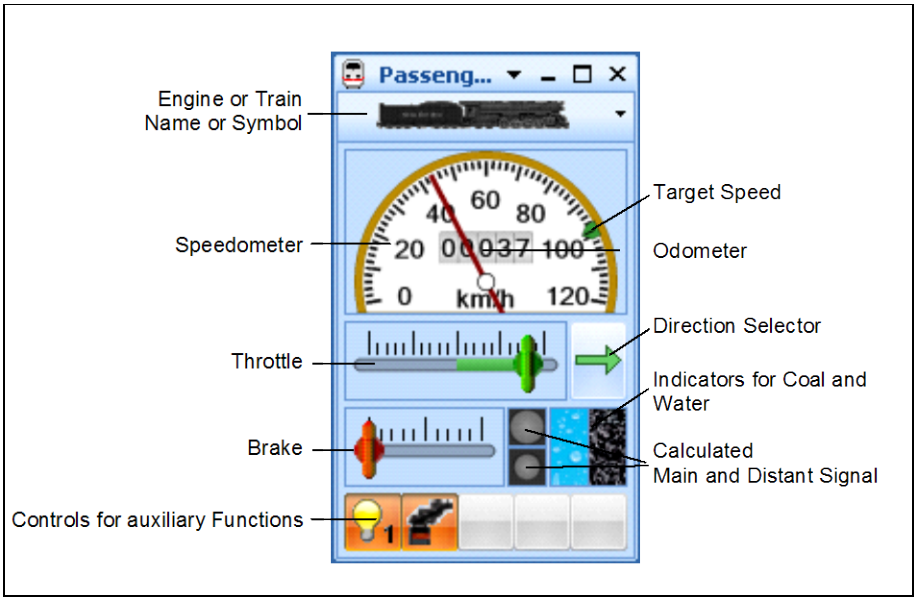

Passeng...
Engine or Train Name or Symbol
Speedometer
Throttle
Brake
Controls for auxiliary Functions
Target Speed
Odometer
Direction Selector
Indicators for Coal and Water
Calculated Main and Distant Signal
0
20
40
60
80
100
120
km/h
00037
1

Diagram 76: Train Window

Train windows provide among others the following features:

- The size of the train window can be continuously adjusted. This is possible on the fly, like any other window, by dragging the borders of the train window with the mouse.
- It is also possible to define an *ideal size* for train windows. The ideal size of each train window can be restored at any time with a single mouse click.
- The symbol or the name of the currently selected train can be displayed.
- The sizes of the particular groups of instruments can be individually adjusted. **TrainController™** is able to switch dynamically between a small and large display mode according to the available space. It is furthermore possible to hide individual and not needed groups of instruments in order to save space on the computer screen.
- The number of steps for operation of the throttle and brake control with the computer keyboard can be individually set. A specific setting allows the throttle to follow the number of physical speed steps of the controlled decoder.
- The throttle can be arranged with the zero position in the middle or on the left.
- The throttle can be set to operate **train oriented** or **layout oriented**. A **train oriented** throttle causes the train always to move <u>forward</u>, when the direction control is pointing to the <u>right</u>, and to move <u>backward</u>, when the direction control is pointing to the <u>left</u>. A **layout oriented** throttle causes the operated train to move <u>to the left on the layout</u>, when the direction control is pointing to the <u>left</u>, and <u>to move to the right on the layout</u>, when the direction control is pointing to the <u>right</u>. This setting is only effective for trains currently assigned to a block (see section 5.2, “Blocks and Routes”). If the block is aligned vertically on the computer screen, then the train will move to the top or bottom, when the direction control is pointing to the top or bottom, respectively. This setting emulates the characteristics of a former throttle for analogue DC railroads.
- The throttle can be set to operate the **speed** of the train (less realistic, but more convenient) or the **power** of the train (more realistic, but less convenient).
  If the throttle controls the speed, then the train is always accelerated with the maximum engine power. Dragging the throttle to a certain position causes the same effect, as if the throttle were first dragged to the maximum position and then reduced to this position, when the train reaches the corresponding speed. If the throttle controls the power, then the train is always accelerated with the power, that corresponds to the slider position. This provides more realistic train control, because many throttles of real railroads actually control the effective power rather than the speed of the train. In such cases the speed indirectly “follows” the effective power. This option, however, also requires more complex user intervention for train control and is less convenient than direct speed control. It is also possible to set individual throttles to operate trains without any momentum.
- The colors and layout of the speedometer and odometer can be individually customized with a wide variety of options.

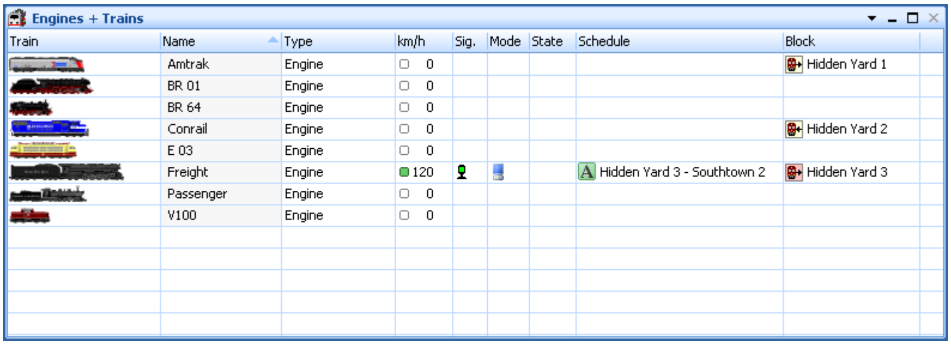
- The speedometer can be made sensitive. In this case the speed of the currently selected train can be changed by clicking to the scale of the speedometer with the left mouse button.
- The currently set target speed can be optionally made visible with a specific marker. The color of this marker can automatically change according to the current direction of travel, if desired, or be set to a constant color.
- All the above settings can be reset to factory defaults at any time, if desired.

Engines + Trains Window

B

The The **Engines + Trains** window of **TrainController™** is used to manage and operate your engines and trains.

The window lists all engines and trains defined in the software and displays additional status information for each train.



Engines + Trains

Train	Name	Type	km/h	Sig.	Mode	State	Schedule	Block
	Amtrak	Engine	▢ 0					Hidden Yard 1
	BR 01	Engine	▢ 0					
	BR 64	Engine	▢ 0					
	Conrail	Engine	▢ 0					Hidden Yard 2
	E 03	Engine	▢ 0					
	Freight	Engine	🟩 120				A Hidden Yard 3 - Southtown 2	Hidden Yard 3
	Passenger	Engine	▢ 0					
	V100	Engine	▢ 0					

Diagram 77: Sample Engines + Trains Window

The **Engines + Trains** window provides the following columns:

- **Train**: an image of the train
- **Name**: name of the train
- **Type**: train type
- **km/h** or **mph**: current speed and direction
- **Sig.**: current signal
- **Mode**: mode of operation
- **State**: status indication
- **Schedule**: currently performed schedule
- **Block**: current location

In the **Engines + Trains** window each engine or train can be selected to change its properties or to operate it.

The data of engines and trains can be exported to a separate file and imported into another **TrainController™** project. In this way it is possible to exchange train data between different layouts or to import train data created at home into a project file belonging to a club layout.

Each item in this list shows the name and the image of the train. To prepare train images for display in **TrainController™** a complementary software program called **TrainAnimator™** is available free of charge.

TrainController™ expects the image data to be stored in a certain format and scaled to a certain size. The images must fit to the proportions of the screen display of **TrainController™**. Additionally the images of several trains should fit together with regard to their scale, regardless of the origin of each image. **TrainAnimator™** is able to process several image formats, among others bitmap, JPEG or GIF. It is also able to extract images, that are stored in application programs or screen savers. **TrainAnimator™** converts the different data formats and image sizes to a standardized and scaled format, which can be used by **TrainController™** without further conversion.

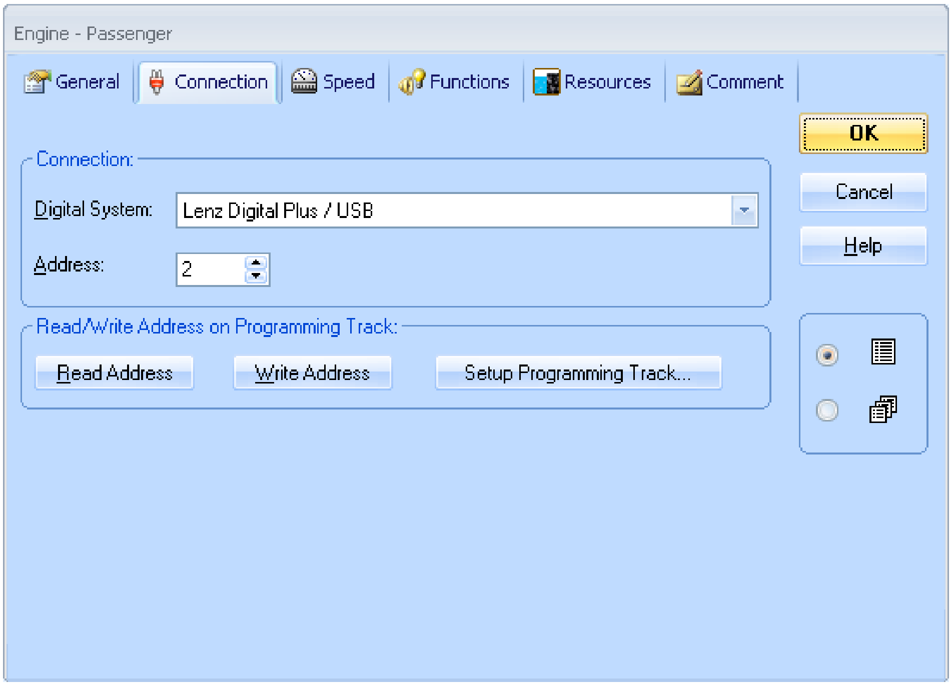
The images displayed in Diagram 77 have been processed with **TrainAnimator™**. Even though the original formats and sizes of the particular images displayed above are very different, they have been converted and scaled to fit together.

3.2 Engines

B

An *engine* describes different properties of one of your model engines. These are prototypical attributes like maximum speed or power, or model related properties like digital address or auxiliary functions.

For operation of your engines it is sufficient to enter each engine with its *digital address* in **TrainController™** To specify the digital address or other attributes mark the appropriate engine in the **Engines + Trains** window or in a Train Window and select the **Properties** command of the **Edit** tab. Once an engine is entered with its digital address it is then possible to control it with the train window.


Engine - Passenger

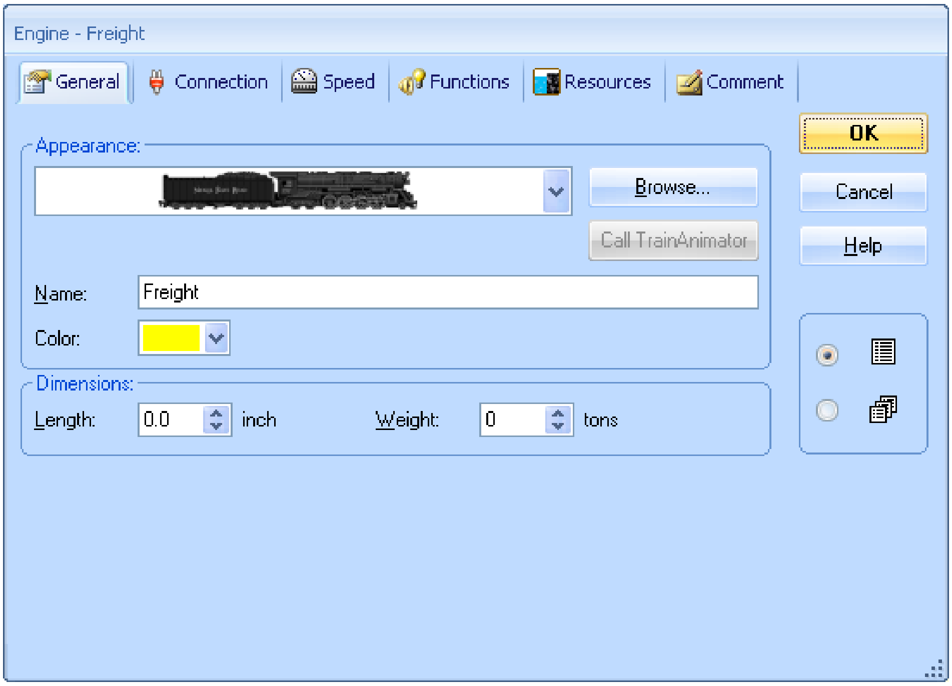
Connection:

Digital System: Lenz Digital Plus / USB

Address: 2

Read/Write Address on Programming Track:

Diagram 78: Digital Address of an Engine



Appearance:

Name: Freight

Color: 🟨

Dimensions:

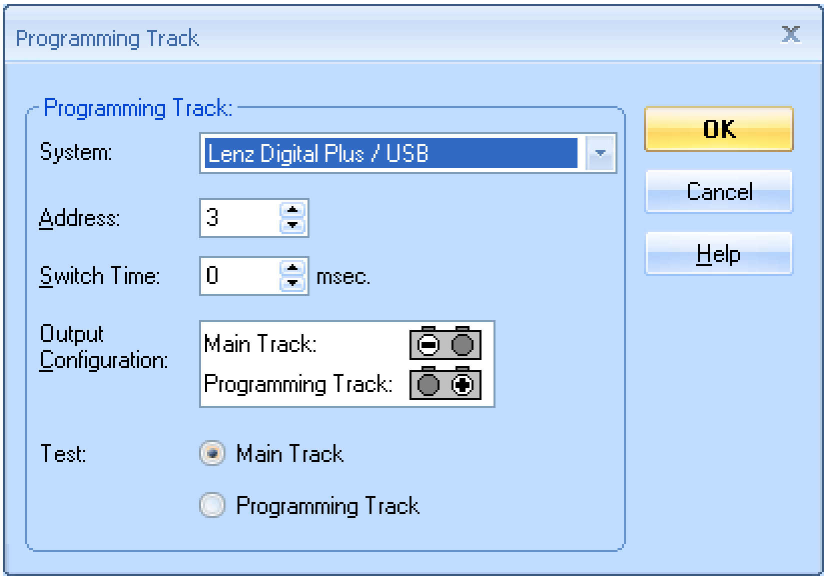
Length: 0.0 inch

Weight: 0 tons

Diagram 79: General Properties of an Engine

TrainController™ **Gold** provides the additional possibility for programming the addresses of locomotives with DCC decoders on the programming track. It is possible to directly read the address from the decoder and to store it in **TrainController™** or vice versa to program the address stored in the program into the decoder. Writing to the decoder, however, requires a license for **TrainProgrammer™**. During programming long addresses, that are stored in several CVs, are automatically converted and processed accordingly, and the concerning bit of CV29 is automatically evaluated or set correctly.

Furthermore, there is the possibility of arranging an appropriate track section of the layout as a temporary programming track. This is done by storing a solenoid address in the software. Through this address a relay or something similar can be operated to connect the track section alternately to the normal main track output of the central unit or the output for the programming track. Whenever **TrainController™** **Gold** sends a programming command to the central unit, it previously operates the relay in order to connect the track section automatically to the programming track output and afterwards it reverts the section back to normal track power.



Programming Track:

System: Lenz Digital Plus / USB

Address: 3

Switch Time: 0 msec.

Output Configuration: Main Track: / Programming Track:

Test: [✓ Main Track] Programming Track

Diagram 80: Arranging the programming Track

This automatic switching of the programming track is also available for the programming functions of the advanced speed profile.

3.3 Throttle and Brake

B

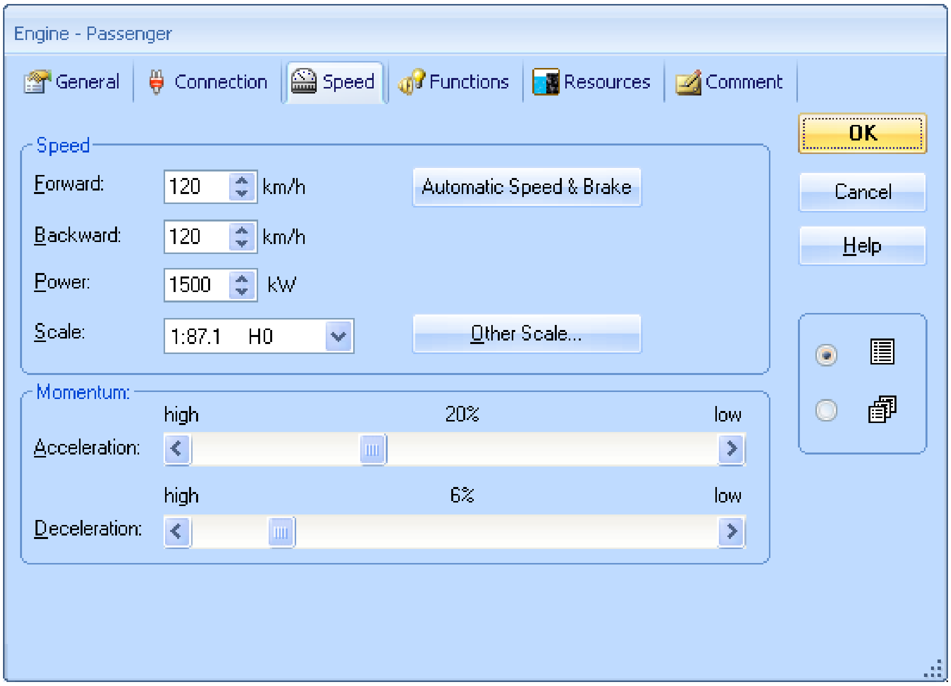
The *throttle* is used to control the speed of each engine. The zero position of the throttle is located in the middle. When the slider of the throttle is in the rightmost position, the train runs forward with maximum speed. Conversely the maximum backward speed is achieved by pulling the slider to the leftmost position.
It is also possible to set the zero position of the throttle to the leftmost position of the throttle control. In this mode the maximum forward or backward speed is achieved by pulling the slider to the rightmost position. The direction of the engine is controlled by the separate direction selector.

The above sections describe the **train oriented** mode of the throttle control. In this mode the throttle causes the train always to move forward, when the direction control is pointing to the right, and to move backward, when the direction control is pointing to the left. In the **layout oriented** mode the throttle causes the operated train to move to the left on the layout, when the direction control is pointing to the left, and to move to the right on the layout, when the direction control is pointing to the right. This setting is only effective for trains currently assigned to a block (see page 155). If the block is aligned vertically on the computer screen, then the train will move to the top or bottom, when the direction control is pointing to the top or bottom, respectively. This setting emulates the characteristics of a former throttle for analogue DC railroads.

In addition to the throttle two additional buttons can be displayed in the train window, if desired, with which the speed of trains can be increased or decreased in single steps, and thus very precisely. If desired a third button can be made visible, with which the throttle can be set directly into the zero position.

An additional instrument to control the speed of a train is the *brake*. Pulling the slider of the brake decelerates the train. The brake is an additional aid. For simplicity it is possible to control the speed with the *throttle* only foregoing the *brake*.

For each *engine* you can specify the *maximum scale speed*. This value is used as the maximum speed with which an engine is controlled by **TrainController**™ To run an engine with maximum speed the throttle slider must be pulled to the maximum position.



Speed
Forward: 120 km/h
Backward: 120 km/h
Power: 1500 kW
Scale: 1:87.1 H0

Momentum:
Acceleration: high 20% low
Deceleration: high 6% low

Diagram 81: Speed Properties of an Engine

For each *engine* you can also specify the *threshold speed*. This is the minimum speed at which the engine runs smoothly. The threshold speed is used if the throttle slider is moved out of the zero position. In this way “dead zones” near the zero position of the slider are avoided. For engines which will run automatically under control of the *Dispatcher* (see chapter 5, “The Visual Dispatcher”) we recommend that you adjust the threshold speed accordingly.

3.4 Speedometer and Odometer

The *speedometer* displays the current *scale speed* of an engine or train. The scale speed is calculated using the real speed on the model railroad layout and the scale of the model. If a train with scale 1:87 (H0) is running with a real speed of 1 mile per hour on the model railroad layout, then this speed corresponds to a scale speed of 87 miles per hour.

In conjunction with the scale factor of the *Clock* (see chapter 13, ” The Clock”) the *simulated distance* is calculated. If the scale factor of the clock is 12, then the duration of one “simulated hour” is 5 real minutes. Our train, which runs with a scale speed of 87 miles per hour passes a distance of 87 simulated miles within 5 real minutes. This simulated mileage is displayed on the *odometer*.

In this way it is possible to simulate very large distances which actually do not exist on your model railroad. Our train, which runs with a real speed of 1 mile per hour passes 87 simulated miles in 5 real minutes or around 1000 simulated miles in one real hour, respectively. This is a scale of 1 to 1000!

3.5 The Speed Profile

To enable the program to display the correct *scale speed* on the speedometer and to perform speed calculations correctly we recommend that you adjust the *speed profile* for each engine.

The speed profile is a table that records which *virtual speed step* corresponds to which *scale speed*. **TrainController™** internally works with 1000 virtual speed steps for each direction regardless of the characteristics of the engine decoder used. When a speed command is sent to the decoder, then the virtual speed step is matched to the appropriate physical speed step of the decoder.

Preparing the decoder

Before adjusting the speed profile, the decoder of the locomotive, if any, should be prepared accordingly. This should be done to achieve the best possible operation. Perform the following steps prior to adjustment of the speed profile:

- Set the start voltage to a value, at which the locomotive begins to run smoothly.
- Trim the maximum speed of the decoder in a way, that the desired maximum scale speed of the locomotive corresponds to the highest speed step of the decoder. If, for example, your decoder has 28 speed steps and the maximum scale speed of the locomotive should be 100 mph, then adjust the maximum speed of the decoder in a way, that the locomotive runs about 100 mph at speed step 28. The procedure to trim the maximum speed setting in the decoder can be conveniently performed with support of **TrainController™**, too. This is explained in more detail in a later section.
- Set the deceleration momentum of the decoder to a minimum value. This is just the value, at which no abrupt speed change of the real locomotive can be noticed anymore, when changing from one speed step to another.
- Adjust the speed table or the mid voltage of the decoder, if any, and its acceleration momentum to any settings, that you feel convenient with.

Note, that the speed profile must be adjusted again, whenever you change the maximum speed, the deceleration momentum, the start or mid voltage or the speed table of the decoder.

The simplified Profile

The software offers two sets of options for adjusting the speed profile of each engine. The first set allows editing of a simplified profile. This simplified profile describes the speed characteristics of your engine very roughly only and with identical settings for both directions of travel. It contains the following entries:

- An entry, that describes the threshold speed. This is the minimum virtual speed step (out of 1000) at which the engine begins to run smoothly. This value is adjusted by letting the engine run as slowly as possible, but also smoothly. If this is achieved the current speed value is stored into the software.
- An entry describing the speed step, that corresponds to a pre-set slow speed. Let the engine run at this speed (e.g. by measuring the speed with a stop watch) and store this value into the software.
- An entry describing the speed step, that corresponds to the maximum speed of the engine. This value is determined and stored in the same way as the other two values.

- An entry describing the braking ramp, that is effective, when the engine is stopped during automatic operation. If the engine is decelerating too slowly or stopping too late during automatic operation, then this value can be easily adjusted.

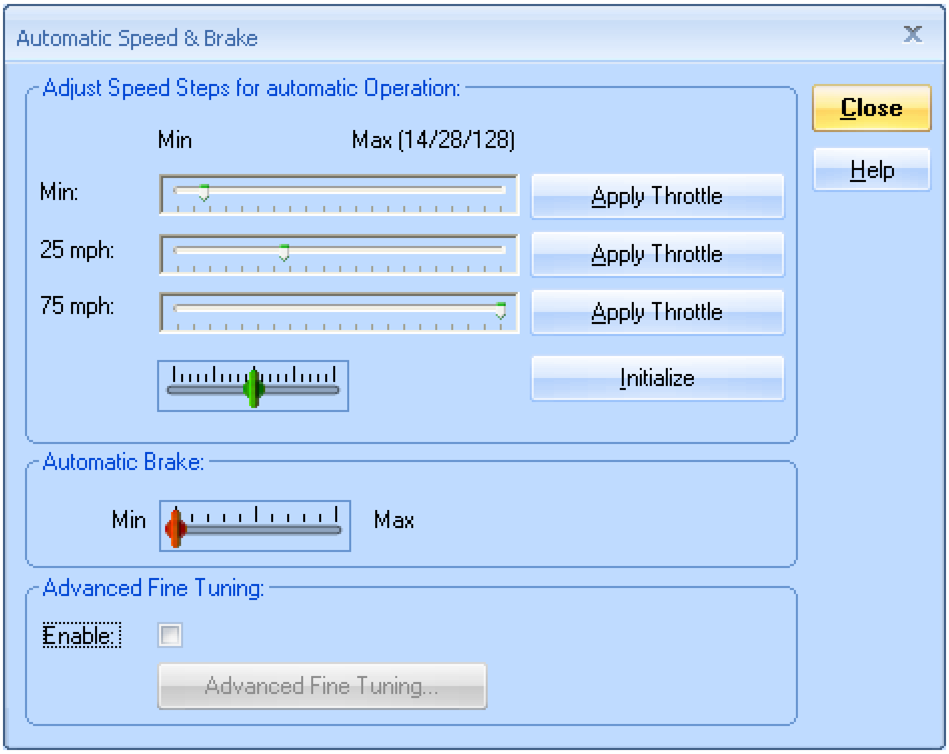


Diagram 82: Adjusting the simplified Profile

This simplified profile describes the speed characteristics of your engine very roughly. This is sufficient for manually operated engines or in many cases during automatic operation, too, if only real stop indicators are being used.

Advanced users, who want to use combined brake/stop indicators (see page 171) or Virtual stop contacts (see section 15.2), should fine tune the speed profiles of their engines as outlined in the following.

! **The settings of the simplified speed profile and the advanced fine tuning affect each other. For this reason it is only possible, to enter options either for the simplified profile or for advanced tuning but not both.**

Advanced Fine Tuning of the Speed Profile

The advanced speed profile is created by measuring the time needed by the affected engine to pass a certain track section. The *scale speed* is calculated using the length of the section and the scale of the model.

For each direction the speed profile contains at most 15 entries for 15 virtual speed steps out of a total of 1000. Intermediate values are calculated accordingly. In this way it is possible to calculate the scale speed for each of the 1000 virtual speed steps.

The entries of the speed profile are distributed equally in the range of the available virtual speed steps. There is no coherence between the number of entries and the number of speed steps of the decoder.

There are five alternative ways to perform the measurement:

Manual measurement of one single speed step (stop watch).

Automatic measurement of one single speed step by running the engine once from one momentary track contact to another. This method is also useful to determine the current maximum scale speed of an engine.

Automatic measurement of one single speed step by running the engine on a track section with three occupancy sensors. This method is also useful to determine the current maximum scale speed of an engine.

Automatic measurement of the complete speed profile with momentary track contacts

Automatic measurement of the complete speed profile with three occupancy sensors

You can measure single values of the speed profile manually like using a stop watch or by running the train once over the measuring section.

TrainController™ also provides the possibility of measuring all relevant values between the *threshold speed* and the *maximum speed* automatically. For this purpose you have to arrange a track section, which is either limited by a momentary track sensor on each side, or which is monitored by track occupation sensors. To each track sensor a *contact indicator* (see section 4, “Contact Indicators”) must be assigned. To measure the speed profile the engine is run back and forth automatically by **TrainController™** The program starts the measurement with the lowest relevant speed. Each time the engine has passed the track section in both directions the engine is accelerated to measure the next speed level. This is repeated until the engine reaches its maximum speed. Monitoring the *contact indicators* **TrainController™** is able to determine, when the engine enters or leaves the track section used for this measurement.

! **Before an automatic measurement of the speed profile is performed, it is important to adjust the threshold speed of the engine. If the threshold speed step is reduced later, after measurement of the speed profile, then the measurement must be repeated.**

The different methods of performing an automatic measurement with momentary track contacts or occupancy sensors are outlined in the following. Further details about the different types of sensors and their usage can be found in section 5.8, “Arranging Indicators and Markers in a Block”.

Measuring with Momentary Track Contacts

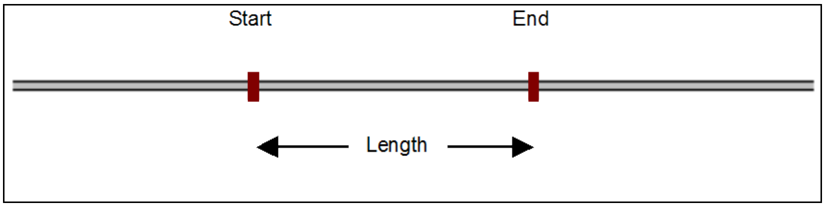


Diagram 83: Measurement with Momentary Track Contacts

For the measurement with momentary contacts two momentary contacts are needed. These contacts are associated with two contact indicators, called “Start” and “End”. The length of the track section used for the measurement is determined by the distance between the two track contacts.

To start the measurement put the engine on the track a certain distance left of indicator “Start” heading to indicator “Start”. The engine will be started in the forward direction. When it reaches the indicator “Start” the measurement of the current speed step begins. When the engine reaches the indicator “End”, then the engine is decelerated and stopped. Now the engine reverses direction and the measurement of the same speed step in the backward direction is performed, now using indicator “End” for the beginning of the measurement and indicator “Start” for the termination. After reaching indicator “Start” the engine is decelerated and stopped and the measurement is repeated for the next speed step in both directions.

The whole procedure is repeated until the speed step that corresponds to the specified maximum speed has been measured.

Please ensure, that both indicators are turned off whenever the engine reverses the direction between two runs of the procedure. There are additional options, with which the run-out and the pause between two passes of the measurement can be adjusted. If an indicator is not turned off when the engine reverses direction then increase the run-out or the pause, respectively.

Measuring with Occupancy Sensors

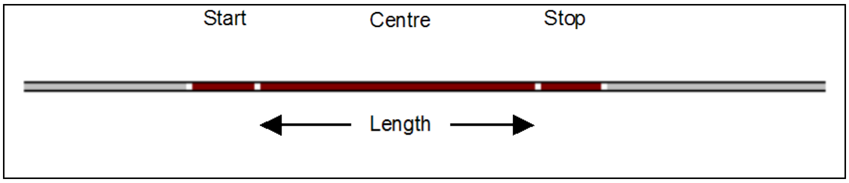


Diagram 84: Measurement with Occupancy Sensors

For the measurement with occupancy sensors three occupancy sensors are needed. These sensors are associated with three contact indicators, called “Start”, “Centre” and “End”. The length of the track section used for the measurement is determined by the length of the occupancy section associated with “Centre”. The length of the other occupancy sections does not matter.

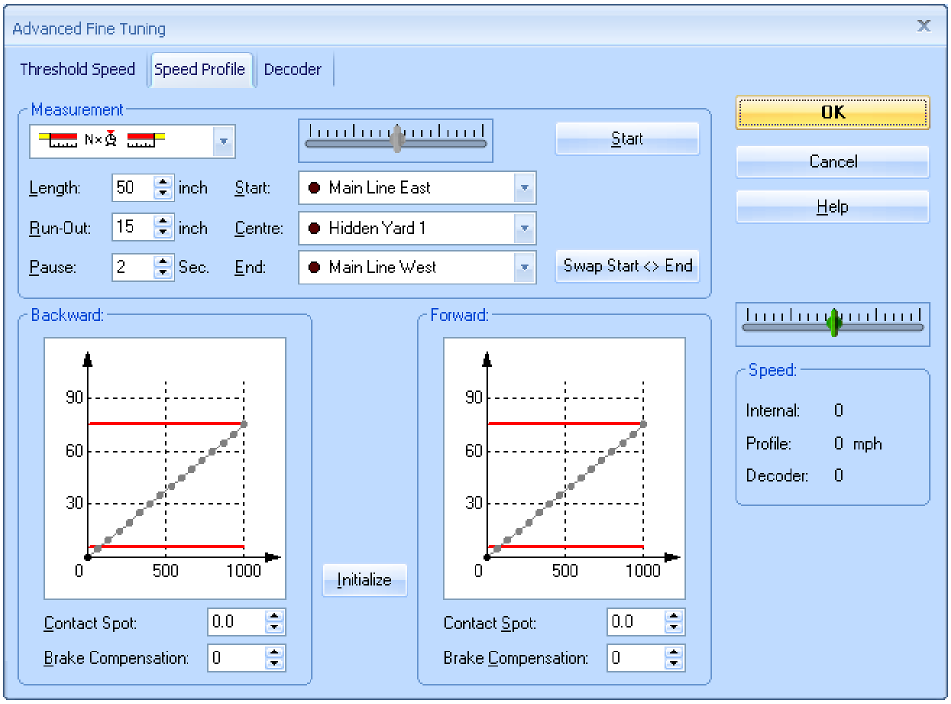
To start the measurement put the engine on the track a certain distance left of section “Centre” heading to section “Centre”. The engine will be started in the forward direction. When it reaches the section “Centre” the measurement of the current speed step begins. When the engine reaches the section “End”, then the engine is decelerated and stopped. Now the engine reverses direction and the measurement of the same speed step in the backward direction is performed, now using indicator “Centre” for the beginning of the measurement and indicator “Start” for the termination. After reaching indicator “Start” the engine is decelerated and stopped and the measurement is repeated for the next speed step in both directions.

The whole procedure is repeated until the speed step that corresponds to the specified maximum speed has been measured.

! **There must not be any "dead gap" between the occupancy sections. That means the track sections must be located close together. Track section "Centre" must begin where the other track sections end and vice versa.**

! **Please ensure, that the indicator associated with "Centre" is turned off whenever the engine reverses direction between two runs of the procedure. There are additional options, with which the run-out and the pause between two passes of the measurement can be adjusted. If indicator "Centre" is not turned off when the engine reverses direction then increase the run-out or pause, respectively.**

It does not matter, though, whether the indicator, that is associated with the track section where the engine just reverses its direction, is turned on or off, when the direction is reversed.



Length: 50 inch
Run-Out: 15 inch
Pause: 2 Sec.
Start: ● Main Line East
Centre: ● Hidden Yard 1
End: ● Main Line West

Backward: Contact Spot: 0.0 Brake Compensation: 0
Forward: Contact Spot: 0.0 Brake Compensation: 0

Internal: 0
Profile: 0 mph
Decoder: 0

Diagram 85: Measuring the Speed Profile with occupancy sensors

The speed profile can also be viewed graphically.

Measuring the speed profile is especially important for all engines, which will run under control of the *Dispatcher* (see chapter 5, “The Visual Dispatcher”). The *Dispatcher* uses the scale speed to control the engines. In this way engines with different characteristics pass the same track sections with identical speed, if the speed profile of each engine is adjusted accordingly.

Trimming the Brake Compensation

In addition to the five procedures to perform the measurement of the speed profile **TrainController™** provides two further procedures, that support the trimming of the brake compensation.

The brake compensation is a value, that describes the braking behavior and deceleration momentum of the physical engine. This value is used to compensate additional deceleration delays - e.g. caused by the engine decoder or by a flywheel mass - when an engine is decelerated. If this engine tends to exceed pre-set braking ramps or stop distances when it is slowed down, then increase this value. The default value is 0, which means, that no compensation is performed. Please note: this option has only an effect in conjunction with distant brake/stop markers, braking ramps or Virtual Contacts and only when engines are decelerated before reaching their location.

The brake compensation cannot be actually measured. The optimal value must be found by trial and error. Nevertheless **TrainController™** can help you to find the optimal value most efficiently. The value can be found with one of the procedures listed below:

Verifying the brake compensation by decelerating an engine from a pre-set speed to zero. The deceleration starts, when a certain momentary contact is triggered.

Verifying the brake compensation by decelerating an engine from a pre-set speed to zero. The deceleration starts, when a certain occupancy sensor is triggered.

The procedure can be performed with the same indicators and the same track sections, that have been used for the measurement of the speed profile. In the case of momentary contacts, the indicator can be used, that marks the start of the measuring track (there called “Start”). In the case of occupancy sensors, the indicator can be used, that marks the measuring track itself (there called “Centre”).

To start the test run put the engine on the track in a certain distance left of the selected indicator (“Start” or “Centre”, respectively) and select a typical speed with the blue slider, with which the affected engine enters those blocks, where it usually has to stop. In the field **Length** of the dialog box specify the length of an estimated average braking ramp used for your blocks.

Then press **Start**, to start the test run. **TrainController™** now accelerates the engine to the specified speed and when the selected indicator is turned on, it tries to decelerate and stop the train at a location, which corresponds to the value specified as **Length**. After the engine stopped, measure the distance between the point, where the engine is located now and the point, where the indicator was turned on. If this distance corresponds to the value specified as **Length**, you are ready. The brake compensation fits.

If the actual distance is higher than the specified **Length**, then increase the brake compensation and repeat the test run. If the actual distance is lower, then decrease the brake compensation and repeat the test run, too.

Repeat the test run, until the brake compensation fits. After this has been done repeat the complete procedure to trim the brake compensation for the backward direction of travel, too.

! **It is important to perform the complete measurement of the speed profile prior to trimming of the brake compensation.**

! **The contact spot of the engine should be determined and specified, too, prior to trimming of the brake compensation.**

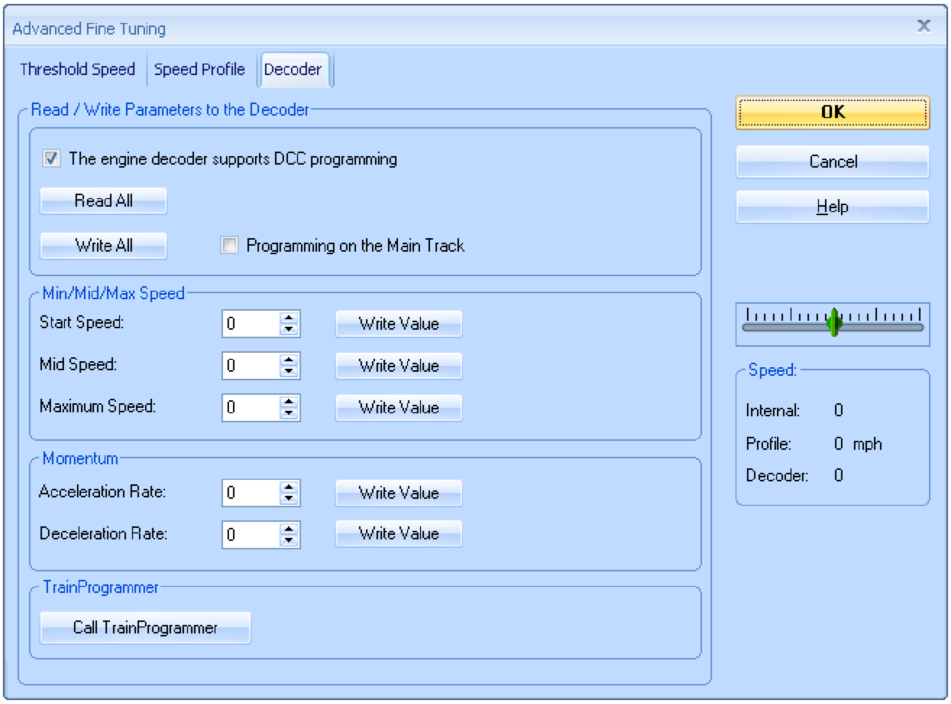
! **In order to achieve maximum accuracy when stopping at shifted stop markers (see section 5.7, “Stop, Brake, Speed and Action Markers”), you should note the following:**

- **A locomotive should enter all blocks, where it has to stop at a shifted stop marker, at a uniform speed.**
- **The length of the brake ramp in these blocks should be as uniform as possible. Different lengths of the blocks can be compensated with corresponding distance settings for the brake markers.**
- **The uniform speed and length of the brake ramp should also be used when adjusting the brake compensation.**

Trimming the Maximum Decoder Speed

The maximum speed of the decoder should be trimmed in a way, that the desired maximum scale speed of the locomotive corresponds to the highest speed step of the decoder. If, for example, your decoder has 28 speed steps and the maximum scale speed of the locomotive should be 100 mph, then adjust the maximum speed of the decoder in a way, that the locomotive runs about 100 mph at speed step 28. The procedure to trim the maximum speed setting in the decoder can be conveniently performed with support of **TrainController™**, too.

If the decoder of the engine can be programmed according to the NMRA DCC standard, then **TrainController™** can change the decoder CVs directly, that are most important for the running characteristics of the engine. This, together with the measurement of the maximum profile value, which represents the maximum speed at the highest decoder speed step, can be used to trim the maximum speed setting in the decoder accordingly: first perform a measurement of the highest speed step. At the end of the measurement the scale speed will be displayed by **TrainController™**. If this scale speed does not match the desired maximum scale speed of the engine, then change the value of the related decoder CV accordingly. This can be done directly via the user interface of **TrainController™**. Then perform the measurement of the highest speed step again and adjust the CV once more, if required, and so on until the scale speed matches. The complete procedure can be conveniently performed via the user interface of **TrainController™**.



[✓] The engine decoder supports DCC programming

[ ] Programming on the Main Track

Start Speed:	0	Write Value
Mid Speed:	0	Write Value
Maximum Speed:	0	Write Value
Acceleration Rate:	0	Write Value
Deceleration Rate:	0	Write Value

Internal:	0
Profile:	0 mph
Decoder:	0

Diagram 86: Programming important speed settings of an NMRA DCC compatible Decoder with TrainController™

If your decoder is not NMRA DCC compatible or if you want to process other decoder CVs, too, then this can be conveniently done by starting **TrainProgrammer**™, which is a separate advanced program for decoder programming.

Note, that a license of TrainProgrammer™ is required to write values into the CVs of a decoder with TrainController™.

!

Even though this procedure has been described here at the end of this section for reasons of the contents, the trimming of the maximum decoder speed step and other speed related decoder CVs must be performed, <u>before</u> the measurement of the speed profile is performed.

Using a third party Speed Measurement Facility

TrainController™ **Gold** provides the possibility to perform all speed measurements with third party measurement facilities, such as roller test benches, tacho wagons, speed measurement devices based on photo sensors, etc. This can normally considerably reduce the time needed for the measurement. For this purpose a speed measurement facility is required, which is able to display the measured speed to the human end user. It is not necessary, that the speed measurement facility can be connected a computer.

The following methods for measurement are available:

Measurement of the complete speed profile with a third party speed measurement facility.

By default the measurement will be performed with a semi-automatic, guided procedure. At first the end user places the locomotive accordingly on the track or on the roller test bench, for example. Then he starts the first test run with a mouse click. **TrainController™** puts the locomotive in motion with the first speed needed to be measured. As soon as the measured speed value is displayed by the measurement device and, if necessary, the locomotive has reached a position, where it can start the next run in opposite direction, the end user stops the locomotive with another mouse click. Then he enters the displayed speed value into **TrainController™**. Afterwards he starts the next test run in opposite direction. These test runs are repeated with different speed until all required speed values are measured.

TrainController™ furthermore provides the ability to perform the complete measurement of all speed values automatically, without the need for intervention by the end user. To support this, the used speed measurement facility must be able to be connected to the computer and must provide some application software on the computer, which is able to (automatically) copy each measured speed value as text to the clipboard. Vendors of speed measurement facilities, which are interested to support the automatic measurement of the speed profile with their products, should contact Freiwald Software for technical details.

When do I have to profile my Locomotive?

Whether it is necessary to capture the speed profile of a locomotive or to adjust its brake compensation depends on the tasks of this locomotive during operation of the layout.

The following table provides an overview:

Locomotive will…	Measuring the Speed Profile	Brake Compensation
stop exactly during automatic operation with one sensor per block	necessary	necessary
stop exactly during automatic operation with separate sensors for stop sections (“three sensors per block”)	not absolutely necessary	not absolutely necessary
run in multiple units	necessary	not necessary
only be controlled manually	not necessary	not necessary

Table 2: When do I have to profile my Locomotive

3.6 Headlights, Steam and Whistle

For each engine an arbitrary number of engine function controls (e.g. light, sound, smoke, etc.) can be defined. Each function can perform one of the following:

- activating a built-in function of an engine decoder
- playing a sound file
- executing a list of operations

Engine Functions can be executed

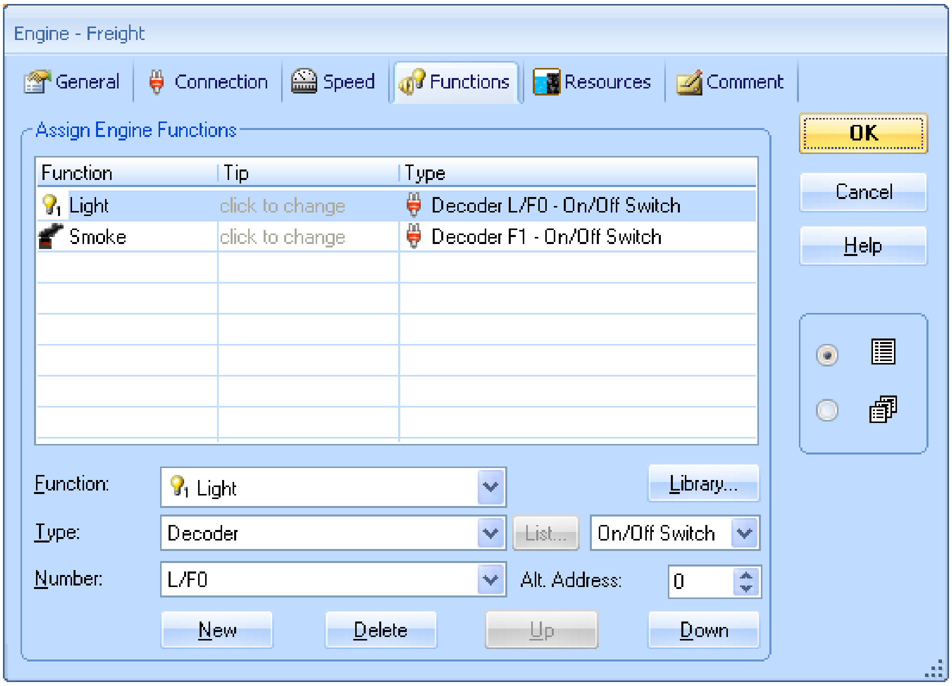
- manually by using the auxiliary function controls of the train window
- when a schedule is executed (see section 5.11, “Schedules”)
- by macros (see section 14.8, “Macros”)

If engine functions are executed by *macros* or *schedules*, then the particular function is identified by its name and symbol (e.g. *Light*, *Smoke*, etc.). If for example a certain schedule has to perform the function symbol *Whistle*, then the function is executed, which is assigned to the engine as *Whistle*. If no such function symbol is assigned, then nothing happens.

You can specify an individual *tip* for each function. This is arbitrary text which is displayed in a small popup window, when the mouse is moved over a function button in the *Train Window*. This tip text helps to distinguish between different functions that are associated with similar function symbols (such as *Light 2*, *Light 3* , …).

The function actually executed may be different from engine to engine. This is illustrated by the following example. It is assumed, that a built-in sound function of the corresponding engine decoder is assigned as *Sound 1* to a diesel engine and playing a sound file with a typical sound of a steam engine is assigned as *Sound 1* to a steam engine. If the function *Sound 1* is executed during automatic operation, then the built-in decoder function is activated for the diesel engine and the specified sound file is played for the steam engine.

Each function, which is assigned to a built-in function of an engine decoder can be turned on permanently (e.g. *Light* or *Steam*) or temporarily (e.g. *Whistle* or *Coupler*). For this reason the auxiliary function controls in the train window can be arranged as on/off switches or push buttons.



Function	Tip	Type
Light	click to change	Decoder L/F0 - On/Off Switch
Smoke	click to change	Decoder F1 - On/Off Switch

Function: Light

Type: Decoder, On/Off Switch

Number: L/F0

Alt. Address: 0

Diagram 87: Arranging Auxiliary Functions

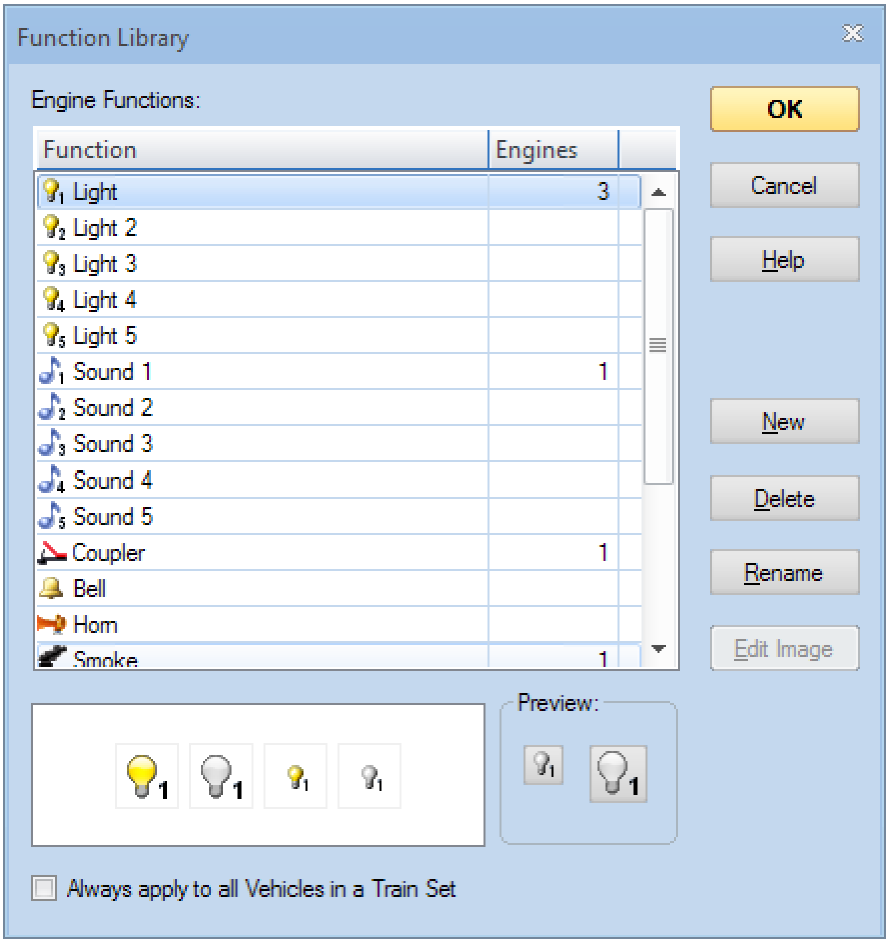
It is also possible to specify certain engine functions as hidden. Such functions are not associated with function controls in the train window and can be controlled automatically by schedules or macros, etc. without consuming space in the train window.

Engine functions can be associated with lists of operations. It is possible to assign two lists of functions to each engine function. The first list of operations is executed, when the engine function is turned on; the other list, when the function is turned off. Engine functions, which are associated with lists of operations, can be configured as push buttons, on/off switches or hidden.

The Engine Functions Library

The engine function library contains the predefined names (e.g. “Light”) and symbols for the available functions. Each function to be assigned to an engine is selected from this library. **TrainController™** is delivered with a default set of predefined functions and symbols. You can also add new functions and draw your own symbols; you can also customise the names and symbols of existing functions to your personal needs.

Each engine function is uniquely identified by the name and symbol stored in the library. If an engine function is executed by a *macro* or a *schedule*, then the particular function is identified by its name and symbol. The function library, however, allows assignment of different actions to the “same” function of different engines. In this way different engines and trains can respond differently to the control of the same common macros and schedules.


Function Library

Engine Functions:

Function	Engines
Light	3
Light 2	
Light 3	
Light 4	
Light 5	
Sound 1	1
Sound 2	
Sound 3	
Sound 4	
Sound 5	
Coupler	1
Bell	
Horn	
Smoke	1

☐ Always apply to all Vehicles in a Train Set

Diagram 88: Engine Functions Library

It is also possible to change the name and symbol of a function stored in the library after the name and symbol has been assigned to an engine, macro or schedule, etc. In this case all references to this function name and symbol are updated accordingly.

By using the option **Always apply to all Vehicles in a Train Set** the currently selected function will always be applied to all vehicles in a train set. This option is useful for functions like interior lights of vehicles, for example, which will be always turned on or off for all vehicles in a train set at the same time. If this option is set for a certain function and this function is turned on or off in the train window for a vehicle in a train set, then this function will be toggled accordingly for all other vehicles in the same train set, too.

Operation of Function Only Decoders

TrainController™ Gold supports the operation of train functions controlled by additional function-only decoders. This is accomplished by specifying an alternative address for each auxiliary function controlled by such decoder.

If a particular locomotive, for example, is equipped with a regular engine decoder with digital address 3 and an additional function decoder with digital address 27, then specify 3 as regular digital address for this engine and set the alternative address to 27 for each function controlled by the function decoder.

The number of alternative addresses, that can be specified for each vehicle, is not limited. Each engine function can carry its own individual alternative address. In other words: **TrainController™ Gold** can operate all decoder functions, regardless, whether they are installed in the main engine decoder or in an additional function decoder, and regardless, how many additional function decoders are installed in a vehicle.

3.7 Passing control between Computer and Digital System

B

Initially, control of each *engine* is assigned to the computer. This means that the software assumes that it has full control over the engine.

With specific menu commands, dependent on the digital systems used, it is possible to pass control from the computer to the digital system and vice versa. For certain digital systems these menu commands are disabled, because these systems are able to pass control automatically (see below).

If control is passed from the computer to the digital system then control of the related digital address is passed to a handheld of the digital system. Additionally – if supported by the digital system - **TrainController™** begins to monitor speed and function changes of this engine and reflects such changes in the Train Window accordingly.

!

For train tracking (see section 5.5, "Train Tracking") of an engine it is important that the software knows the direction and speed of a running engine. If you want to control an engine with a handheld of your digital system under simultaneous train

tracking, then it is necessary to assign control of the engine to the digital system before, if this is not done automatically (see below).

If an automatic schedule of the Dispatcher (see section 5.11, “Schedules”) is executed with an engine currently under control of the digital system, then control of this engine is temporarily passed to the computer. After finishing the schedule, control is passed back to the digital system. Such transfers of control are performed by the software automatically if needed.

! **Assigning control of an engine to the computer is necessary, if you want to control the engine manually with the on-screen throttle.**

! **For those digital systems, where the assignment of control is done automatically, no manual intervention is required. The appropriate menu commands are therefore locked.**

4 Contact Indicators

B

If your digital system is able to report the state of *track contacts*, *reed contacts*, *optical sensors*, *track occupancy sensors* or other feedback sensors to the computer, then you can indicate the status of these contacts and sensors with *contact indicator* symbols. With these indicators, you are able to monitor the state of the feedback sensors on the computer screen.

Contact indicator are used in **TrainController™** for the following purposes:

- Occupancy indication of blocks for tracking and control of trains (see chapter 5, “The Visual Dispatcher I”)
- Occupancy indication of routes (see page 345)
- Execution of actions by passing trains (see section 14.4, “Operations”)
- Individual highlighting of track sections in the switchboard (see section 2.8, “Highlighting occupied track sections”)

If your digital system is not able to report the state of feedback sensors to the computer, then **TrainController™** makes it possible to connect a second or more digital systems to your computer. For this purpose, it is not necessary to purchase another complete digital system that is also able to operate trains and turnouts. **TrainController™** supports special low cost digital systems that are dedicated to the monitoring of feedback sensors. More details about running several digital systems, simultaneously, are outlined in section 18.2, “Operating Several Digital Systems Simultaneously”.

Feedback sensors are divided into *momentary track contacts* and *occupancy sensors*. In **TrainController™** the same symbol is used for both types of contacts. The difference between both types of contacts does not play an important role as long as trains are not operated under control of the *Visual Dispatcher* (see section 5, “The Visual Dispatcher”).

Momentary Track Contacts vs. Occupancy Sensors

Momentary track contacts are turned on for a short period, when a train passes a certain point on the model railroad. They stay turned on only for a short period and are turned off as soon as the train moves any further. In Diagram 89 to Diagram 91 you can see a momentary contact triggered by a passing train. Momentary track contacts indicate that a train is about to pass a certain point. *Occupancy sensors* are turned on when a train enters a certain section on the model railroad. They stay turned on until the train leaves that section completely. Occupancy sensors indicate that a train is located inside a certain track section. In Diagram 92 to Diagram 95 you can see an occupancy sensor turned on and off by a passing train. Occupancy sensors are able to report the presence of a train inside a certain track section even if the train is not moving. Momentary contacts are triggered by moving trains only. Momentary contacts can be made for instance by mechanical track contacts, reed contacts or optical sensors. Occupancy sensors are often based on current sensing in isolated track sections.

Unlike other programs which require occupancy sensors for automatic train control **TrainController™** is also able to control trains if only momentary track contacts are used. Occupancy sensors are safer, though, because with momentary contacts special measures against premature release of blocks and routes must be taken.

The following diagrams show the behavior of a momentary contact in the different phases while a train is passing. The position of the momentary contact is marked with a short vertical line.

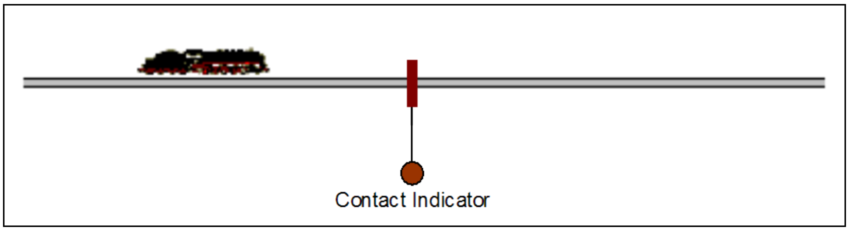


Diagram 89: Train is approaching the momentary contact – the contact is turned off

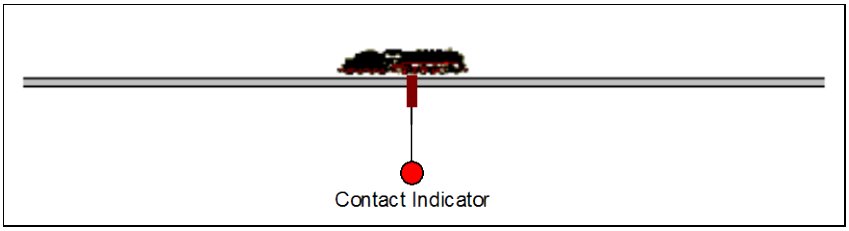


Diagram 90: Train is reaching the momentary contact – the contact is triggered

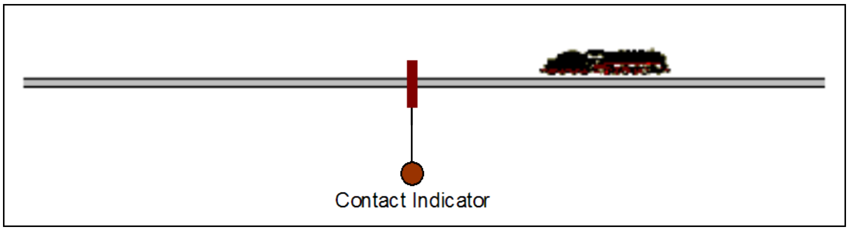


Diagram 91: Train is leaving the momentary contact – the contact is turned off

The following diagrams show the behavior of an occupancy sensor in the different phases while a train is passing. The track section sensed by the occupancy sensor is marked with a horizontal line.

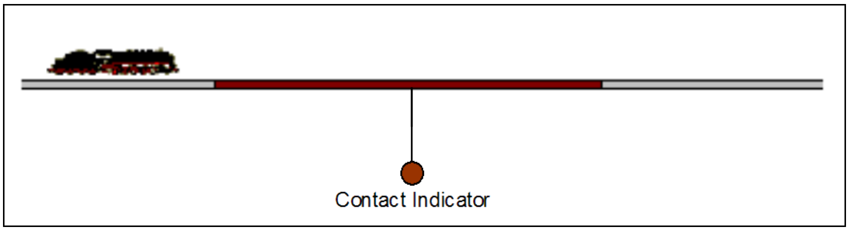


Diagram 92: Train is approaching the occupancy sensor – the sensor is turned off

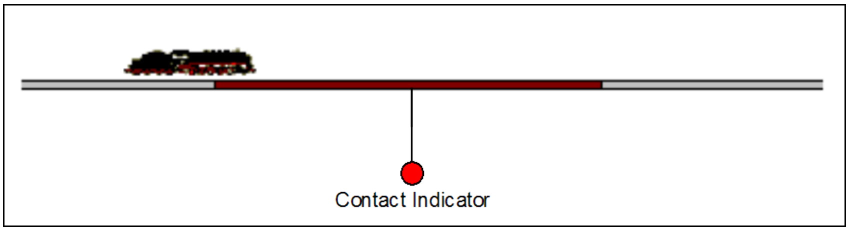


Diagram 93: Train is located inside the sensed section – the sensor is turned on

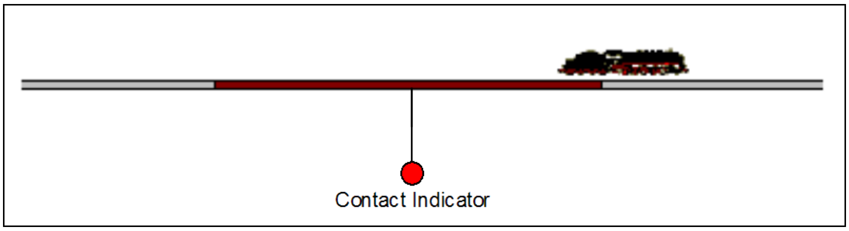


Diagram 94: Train is still located inside the sensed section

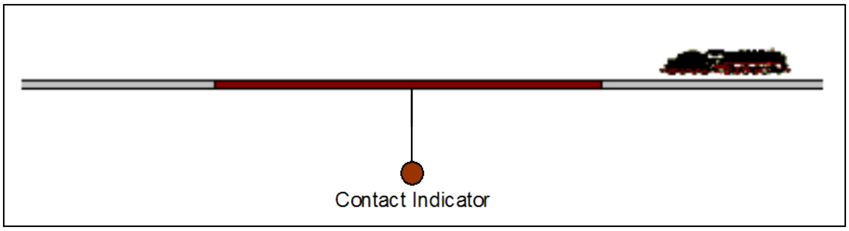


Diagram 95: Train has left the sensed section – the sensor is turned off

There is one major difference between momentary contacts and occupancy sensors to remember: the points at which the indicators are turned on. A momentary track contact is turned on when a train reaches a certain point on the layout regardless of the direction of travel of the passing train. In this way a momentary track contact represents one single sensing point on the model railroad. An occupancy sensor is turned on when a train reaches either end of the sensed track section depending on the current direction of travel of the passing train. In this way an occupancy sensor represents two different sensing points on the model railroad. It depends on the direction of travel of a passing train at which of these two points the train triggers the sensor.

! Even though the software works well with both types of sensors, momentary and occupancy, it is important to ensure, that the indicator symbol, that is associated with a certain sensor, is only turned on once by each passing train, even if the physical sensor is triggered two or more times by the same passing train. Indicator symbols, that are turned on two or more times by the same passing train (“flickering”) may mislead the software and can cause unexpected behavior of the affected trains. This is especially true for trains running under automatic control of the computer. **Each indicator symbol, that is being passed by a train under automatic control of the computer, should be turned on only once by the passing train.**

5 The Visual Dispatcher I

5.1 Introduction

B

A human operator is normally only able to operate one or two switchboards and at most two trains at the same time. If multiple control panels or a certain number of trains are to be operated at the same time, then either support of additional human operators is required, or a component like the *Visual Dispatcher*, which is able to take the place of additional human operators.

The V*isual Dispatcher* (or in a word *Dispatcher*) is a component that makes large scale railroad operations manageable by one person, matching operations found on the largest club layouts.

Like a human operator the Visual Dispatcher is able to operate turnouts, signals, routes and trains. This is called *automatic operation*.

A broad range of operating flexibility is provided from completely manual through to fully automatic operation (e.g. hidden yard control). Manual and automatic operation can be mixed simultaneously. This applies not only to trains on different areas of your railroad, but also to different trains on the same track and even to the operation of a single train. Automatic processes are not bound to specific trains. Once specified they can be performed by each of your trains. There is no need to learn a programming language. Time-table and random functions increase the diversity of your model railroad traffic. Built-in train tracking functions display on the screen which engine/train is on which track.

Like a human operator must know the overall structure of the model railroad layout the *Visual Dispatcher* needs to know this, too. This structure is represented by a diagram that contains blocks and routes and the track connections between them. This diagram is called *main block diagram* of the layout. The main block diagram describes the track layout of your entire model railroad in rough outline.

The *Visual Dispatcher* manages traffic flow using a blocking system. Blocking ensures that trains do not collide and supports the tracking of train positions. For this purpose, the railroad layout is divided into virtual, logical blocks. That means, you define blocks at locations where traffic control will take place (e.g. scheduled stops in a station). Usu-

ally each track in a station or hidden yard, each siding and appropriate sections of the connections between two yards will form a block.
Blocks are arranged graphically and connected by *routes* to specify on which path a train will travel from certain starting blocks to destination blocks. *Schedules* describe train movements, i.e. how trains travel. This includes scheduled waits, speed limits, etc.

Trains can run under full manual control, in which case the operator will be responsible for obeying the block signals set by the *Dispatcher*; under full control of the computer; or even with an intermediate level of automation.

For shunting special types of schedules are provided.

Schedules and *timetables* can be arranged with a broad range of flexibility. Since a time-table can be created for each day of the year, up to 365 timetables can be used.

Random functions increase the diversity of your model railroad traffic.

Creating a model railroad operation system with the *Dispatcher* is done by performing the following steps:

- Divide the model railroad layout into *blocks* and enter these blocks into **TrainController™**
- Arrange blocks and routes between them in the *main block diagram*. This diagram will represent the track layout of your entire model railroad in rough outline.
- Arrange *schedules* and optionally create *timetables*

These steps will be outlined in more detail in the following sections. We will do this by looking at the following sample layout:

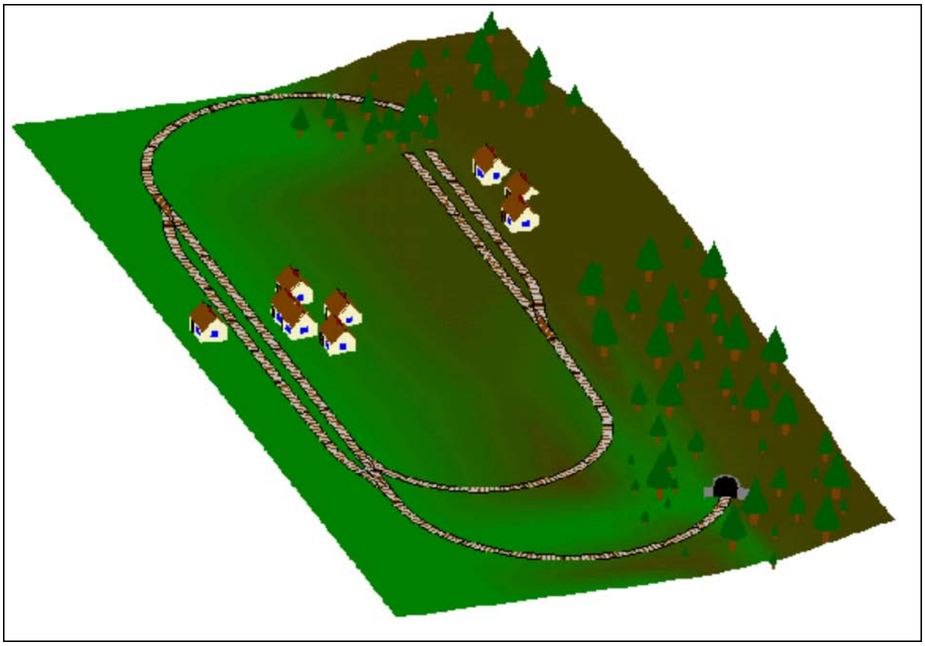


Diagram 96: Sample Layout

The layout has two stations: “Southtown” located on the left side of the layout and “Northville” located at the end of a branch line. There is an additional hidden yard that is covered by the mountain.

This can be seen better in the track plan displayed below:

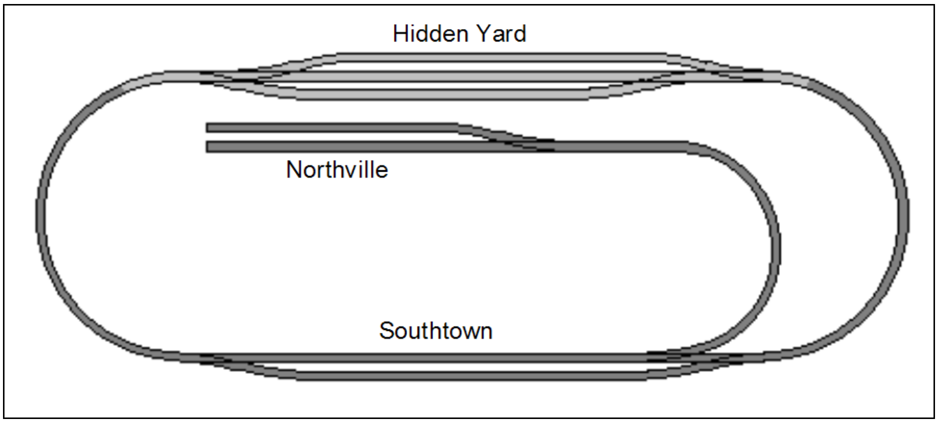


Diagram 97: Track Plan of the Sample Layout

The main line, i.e. the loop that connects “Hidden Yard” and “Southtown”, will be operated automatically under full control of the *Visual Dispatcher*. The branch line from “Southtown” to “Northville” will be operated manually.

The parts of the layout that are covered by structure and therefore invisible are drawn here in a slightly brighter color.

The first step is drawing of a switchboard for the layout displayed above. It looks as follows:

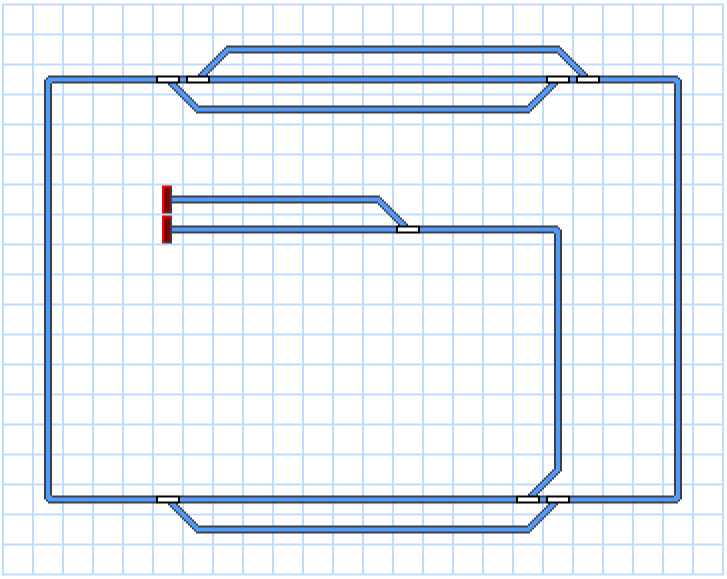


Diagram 98: Switchboard of the sample layout

The next steps, that are required to configure this layout in the *Visual Dispatcher*, are outlined in the following sections.

5.2 Blocks and Routes

Blocks on the Layout

B

The *Visual Dispatcher* manages traffic flow using a *blocking system*. Blocking ensures that trains do not collide. For this purpose the railroad layout is virtually divided in logical blocks. That means, you define blocks at locations, where traffic control will take place (i.e. stopping inside a yard) or where trains are parked. Blocks are also used to determine and to indicate the position of your engines and trains on your tracks.

Typical examples of blocks are

- Tracks at a platform
- Sidings in a (hidden) yard
- Block sections on tracks between two stations

In most cases blocks contain only a straight track section and no turnouts. They are usually limited by two turnouts on both sides or by a turnout and a dead end of the track. Block sections between two stations are often limited by block signals.

Some guidelines for arranging your blocks:

- Blocks may be located anywhere on your railroad.
- Blocks are often limited by turnouts. These turnouts usually do not belong to the blocks.
- Blocks should be long enough to hold each stopping train completely.
- Each location, where the *Visual Dispatcher* will be able to stop a train automatically (e.g. in a station or in front of a signal), should be located in a separate block, i.e. in order to stop two trains at the same time at different locations, these locations must be arranged in different blocks.
- The more blocks are provided the more trains can be run simultaneously under control of the *Visual Dispatcher*.
- Each block can be reserved by at most one train. A specific train may reserve several blocks. A train, that runs under control of the *Visual Dispatcher*, may only enter blocks, that are reserved for this train.
- Blocks only need to be provided for the parts of your model railroad, which will be controlled by the *Dispatcher*. Parts without blocks are not visible to the *Dispatcher*.

Following these guidelines the block structure of the sample layout looks as follows:

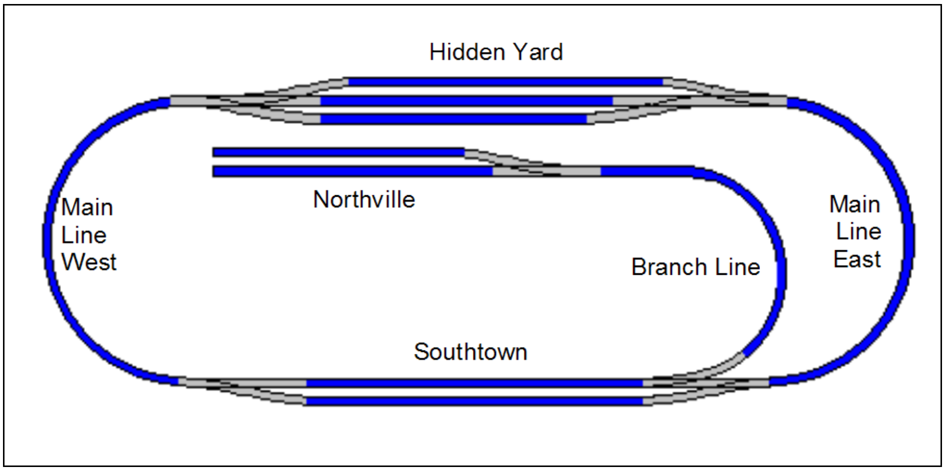


Diagram 99: Block structure of the sample layout

Each blue track section represents a separate block. The blocks on the main line or the branch line between "Southtown" and "Northville" can be subdivided further into more blocks if each of these blocks is long enough to store the longest train. This is useful if you want more than one train to travel on these tracks at the same time.

Block Diagrams

Like a human operator must know the overall structure of the model railroad layout the *visual dispatcher* needs to know this, too. This structure is represented by one or more diagrams, that contain blocks and routes between blocks. These diagrams also display the various itineraries of your trains. Such diagrams are called *block diagrams* of the layout. They describe the track and block layout of your entire model railroad in rough outline.

Block diagrams are displayed in separate windows, the *dispatcher windows*.

Normally each switchboard, that you create for your layout and that contains blocks, corresponds to a block diagram. These block diagrams are created automatically by **TrainController™** by using the track layout drawn in the switchboard and the information about the blocks contained therein. To enable **TrainController™** to create ("calculate") a block diagram for a switchboard, it is necessary to specify the positions of the blocks in the track diagram of the switchboard, if there are any. This is done with the help of *block symbols*.

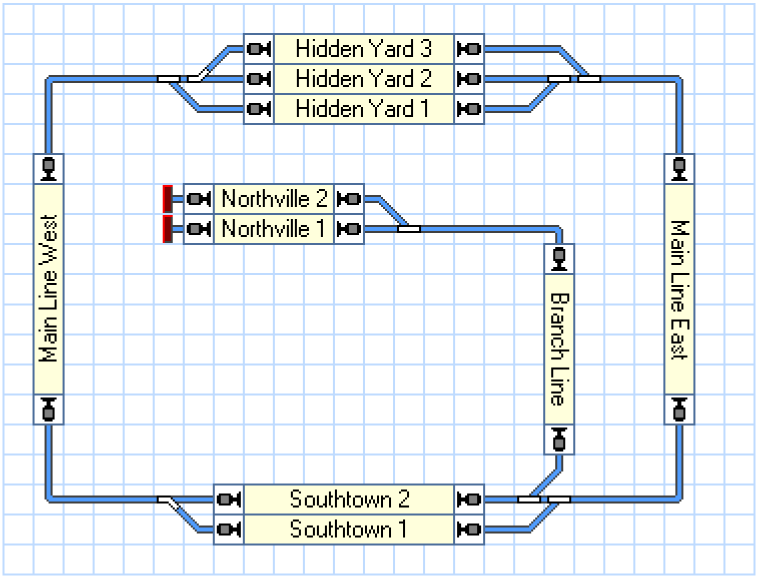


Diagram 100: Switchboard with Blocks

By creating a switchboard, drawing a track diagram in it and inserting block symbols at positions, where blocks are located, **TrainController™** will automatically calculate a block diagram for this switchboard. All connecting routes will be automatically calculated, too, with all contained turnouts. No extra human intervention is necessary to accomplish this.

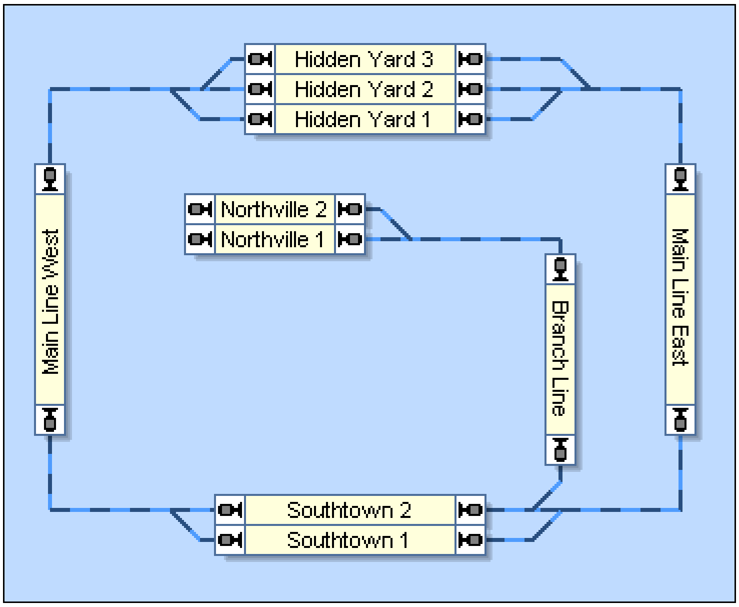


Diagram 101: Block Diagram in the Visual Dispatcher

Blocks are displayed on the computer screen by rectangular boxes. The blocks are connected to each other by routes, which touch each box graphically at a smaller side. These routes are drawn as lines.

Please note that the block diagram represents the track layout in rough outline. The actual track connection between "Main Line West" and "Hidden Yard 3", for example, contains two turnouts. These turnouts are not drawn in the block diagram in detail or as separate objects. Instead a line between the blocks is created, that indicates, that there is a track connection between the blocks.

In order to enable **TrainController™** to calculate the block diagram automatically note the following:

- Draw the complete track diagram of your layout with all turnouts and crossings and without any gaps in a switchboard window.
- Create block symbols for all blocks of the layout, place them according to their location on the actual layout and ensure, that they are turned horizontally or vertically according to the track symbols, to which they are attached.
- Ensure that the blocks are connected to each other by track symbols without any gaps. The connecting tracks must touch the blocks at the smaller sides.

For specific purposes it is also possible in **TrainController™** **Gold** to place blocks on diagonal track symbols. For technical reasons the size of such blocks is automatically shrunken to one single switchboard cell. Adjacent track elements, which will connect such *mini block symbols* with adjacent blocks must touch the block accordingly in appropriate opposite corners.

When working with **TrainController™** you may notice, that switchboards and their corresponding block diagrams seem to look almost identical at first glance. But this is not actually the case. Switchboards contain the details of the track diagram, i.e. each particular track symbol and turnout and also additional objects like signals, push buttons etc. Switchboards are also the base for you to operate your layout, i.e. to perform manual interventions during operation. In contrast, block diagrams display routes between blocks rather than single track or turnout symbols and no additional objects like signals or buttons. Block diagrams mainly serve to manage the blocks and routes and to define and manage predefined itineraries for your trains ("schedules"). They can also be used to monitor the traffic on your layout but are usually not used for manual intervention. In many cases you will display the block diagrams only during edit mode to manipulate your data but hide them during operation.

TrainController™ **Silver** is limited to one block diagram in total, even if more than one switchboard exists. The automatic calculation of the block diagram works only for one selected switchboard.

TrainController™ **Gold** allows to work with as many switchboards and block diagrams as necessary to represent your complete layout.

Even though block diagrams are normally automatically created by the software, it may be necessary under certain circumstances, however, to integrate a part of your layout into the block system of the dispatcher, which cannot be represented in a switchboard window. For this purpose **TrainController™** allows to create additional, manually drawn ("custom") diagrams, too.

Routes between Blocks

In order to let trains run from one block to another the blocks must be linked together. This is done with the help of *routes*. In the block diagrams routes are represented by lines that connect one block with an adjacent block.

Each block has two entries/exits. If a block is passed horizontally, then the entries/exits are graphically located on the left and on the right side of the block. If a block is passed vertically, then the entries/exits are located at the top and at the bottom. Each route begins at the entry/exit of a block and ends at the exit/entry of an adjacent block.

The following image explains the terms once more:

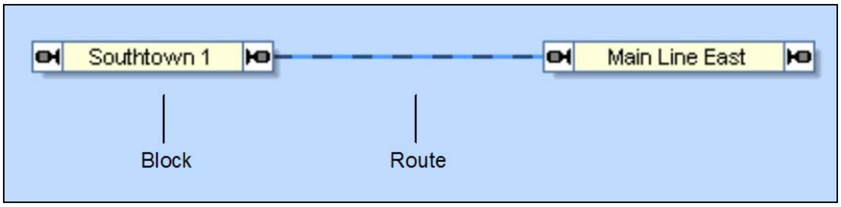


Diagram 102: Blocks and Route

In the diagram displayed above the blocks "Southtown 1" and "Main Line East" are connected with a route.

In many cases the track connection between two blocks contains one or more turnouts. In Diagram 100, for example, the route between Block "Main Line East" and "Southtown 2" contains two turnouts. To enable a train to travel automatically from one block to another the route between both blocks is activated. When this happens, all turnouts contained in the route are operated accordingly. All track elements along the path of the route remain locked in this position until the route is turned off again. As long as these elements are locked, they cannot be operated or used by other routes.

Linking Switchboards together - Connector Symbols

If you are working with more than one switchboard and there are track connections between parts of your layout, that are represented by different switchboards, then these track connections can be represented by *connector* symbols.

Connector symbols are inserted into the track diagrams of your switchboards in a similar way like blocks. Each connector symbol has a name of up to 2 letters or digits, which is displayed in the switchboard, too. To link a certain track symbol in one switchboard to a track symbol in another, insert a connector symbol next to each track symbol in both switchboards and assign the same letters or digits to both. Associated connector symbols are namely identified by identical names. Two connector symbols are associated with each other, if they have got the same name. It is not possible to create more than two connector symbols with the same name.

Connector symbols located in a switchboard are also inherited to the associated track diagram. Furthermore **TrainController™** **Gold** automatically creates a hidden route between

ween each two associated connector symbols. This route represents the said track connection between the two switchboards. From now on **TrainController™** **Gold** is aware, that trains can travel from one switchboard to the other by passing these connectors and the hidden route in between.

If you like you can also use connector symbols to connect one part of your track diagram with another in the same switchboard. In this case **TrainController™** **Gold** will also create a hidden route, if both connectors are contained in the same switchboard diagram. This is sometimes useful for large, complex track diagrams, where omitting certain track connections improves the clarity of the display.

It is also possible to insert connector symbols into custom diagrams to connect such diagrams with switchboards or again other custom diagrams.

5.3 Direction of Travel vs. Engine Orientation

B

It is important to understand the difference between *direction of travel* and the *orientation* of an engine.

Direction of Travel

Direction of travel is seen from the passenger's point of view. For the passenger sitting in a train it is important to know, whether the train runs from the east to the west, from the city to the country, or from the sea to the mountains. The direction of travel has a “geographical” meaning. Each *block* can be passed in one of two directions at a time. For each train controlled by the *Dispatcher* the *Dispatcher* must know the train's intended direction of travel. This information is derived by the *Dispatcher* from the arrangement of the blocks in the related diagrams and the routes that connect these blocks.

TrainController™ draws each block to represent one pair of corresponding directions. Each block can be either passed horizontally (from the left to the right or back) or vertically (from the top to the bottom or back).

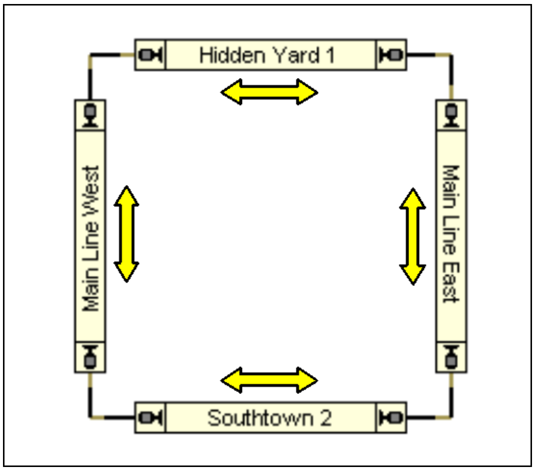


Diagram 103: Block Diagram of a Circular Layout

In the diagram displayed above the direction of travel of each block is here indicated by an arrow. **TrainController™** does not display these arrows, though, but displays small signal symbols on two sides of each block, to mark the direction of travel, that belongs to the block.

The direction of travel will correspond to the drawing of the block in the diagram. A block that is passed horizontally will be drawn as a horizontal rectangle while a block that is passed vertically will be drawn as a vertical rectangle. This is shown in the diagram displayed above.

Engine Orientation

Engine Orientation is seen from the engineer's point of view. It is not important for the passenger. The engine orientation describes the direction of the engine's head. For an engineer, who has to run a train in a certain direction of travel it is also important to know the engine's orientation, i.e. the direction of the engine's head. Depending to the intended direction of travel and the engine's orientation the engineer can decide, whether the engine is to be run forward or backward.

When the *Dispatcher* runs a train, it acts like an engineer. Both items of information - the intended direction of travel and the engine's orientation - must be known by the Dispatcher to start the train correctly.

! The orientation of each engine is specified during assignment of an engine or train to a block. There are several methods to assign trains to a block. The most convenient method is to drag & drop a train icon to the symbol of a block. **Please always check that the current orientation of the engine matches the screen display.** In the case both do not match it is possible to revert the screen display with appropriate menu commands.

Another method for automatic assignment of trains to blocks is the use of train identification or train tracking (see 5.5, "Train Tracking").

5.4 States of a Block

B The different *states* of a block are determined by the fact whether the block is *occupied* or whether it is *reserved* for a certain *engine* or *train*.

Occupied Block

A block is assumed to be *occupied*, if at least one of the *indicators* assigned to the block is turned on.

Reserved Block

Each block can be manually or automatically *reserved* for an *engine* or *train* by the Dispatcher. Reservation serves to support the following goals:

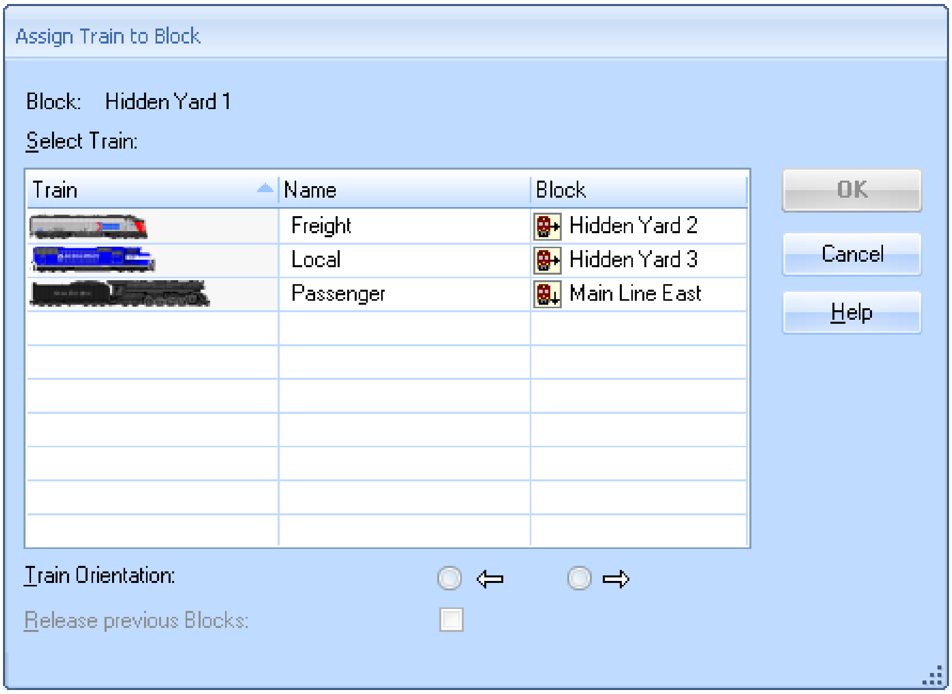
- Since a block can be reserved only for at most one *engine* or *train*, train collisions are avoided if blocks are arranged and reserved correctly.
- The program is able to determine, in which block a certain engine or train is located. This enables operations tied to the locality of trains - for example stopping a train in front of a red signal.
- The use of *block symbols* allows indication of train positions in the *switchboard*.
- Train identification and train tracking is based on dynamic and automatic reservation of blocks, too (see 5.5, "Train Tracking").

For *shunting* or similar manual operation it is possible to reserve a group of related blocks manually. In this case the *Dispatcher* takes care, that automatically controlled trains do not enter these blocks. If reserved blocks are no longer needed, they can be *released* by yourself or automatically by the *Dispatcher*.

Current Block

Among the blocks, which are reserved for a train, there is a special block, where the head of the train is assumed to be located. This block is called the *current block* of the train. Through the current block all block related operations which affect the speed of a train (like running with restricted speed) are performed.

In the beginning you must manually assign each engine or train to its current block. Afterwards this assignment is adjusted automatically by **TrainController**™ according to the position changes of the affected trains. Even after terminating and restarting of the program this assignment is automatically updated. Only if an engine or train is moved by hand to another track you must assign the engine or train to its new current block again.


Assign Train to Block

Block: Hidden Yard 1

Select Train:

Train	Name	Block
	Freight	Hidden Yard 2
	Local	Hidden Yard 3
	Passenger	Main Line East

Train Orientation: ○ ⇦ ○ ⇨

Release previous Blocks: ☐

Diagram 104: Assigning a train to the Current Block

When an engine or train is assigned to its current block, the current *engine orientation* must be specified. **TrainController**™ needs to know this orientation to be able to determine, if an engine will run forward or backward. **TrainController**™ adjusts the engine orientation accordingly even if an engine changes its orientation by passing a *reversing loop*.

TrainController™ provides several methods to assign a train to a block. The most convenient is to drag a train from the **Engines + Trains** window to the symbol of a block. The initial assignment of an engine or train to a block can also be done automatically without manual interaction, if a *train identification device* is used (see section 5.5, "Train Tracking"). If this device is associated with a block then each engine or train detected by the train identification device will be automatically assigned to this block.

A reserved block must not necessarily be occupied. This is also true for the current block. If for example a train leaves its current block and temporarily no other blocks, that are reserved for this train, are occupied, then the current block is not changed, before the train enters another block and this block is indicated as occupied.

Display of Train Positions

The states of a block outlined are indicated by the concerning block symbols in the switchboard. In this way you can control in the switchboard, too, if a certain block is occupied or reserved. Block symbols display the name and/or the image of the train, that is currently located in the related block, in the switchboard. For further details refer to 5.5, "Train Tracking", please.

Unidirectional Blocks

In **TrainController™** **Gold** each block can be specified to be unidirectional. An unidirectional block can only be passed in a certain direction of travel.

Locking the entries of Blocks

Each block can be temporarily locked during operation. Locked blocks cannot be reserved by running trains. A train, that is already located in a block, when the block is locked, might stay there, though, and leave the block later. A lock does not also have an effect for a train, that has already reserved the block, before the lock is set. This train may proceed into the locked block.

Locks are directional. It is possible to set an individual lock for a particular direction of travel. This permits trains to pass the block only in one direction of travel. For this reason these locks are also called entry locks. The lock prevents trains from entering the block via the locked entry, while trains approaching the block from the opposite direction are not affected by this lock.

Please note that locking of a block affects all trains.

In **TrainController™** **Gold** locking the entry of a block causes a similar effect like an unidirectional block (see above), i.e. both prevent trains from passing a block in a certain direction of travel. There are some important differences, however, between unidirectional blocks and blocks, which are locked to certain direction of travel:

- An entry lock can be set and removed at any time during operation. Unidirectional blocks can only be changed in edit mode. Hence an entry lock prevents a train from passing the affected block in a certain direction only temporarily while an unidirectional block does this permanently.
- Entry locks are treated as temporary obstacles. It is possible to establish a path for a train run (e.g. via **AutoTrain™**), that passes a block in a (temporarily) locked direction. The train may even approach a locked block, awaiting that this lock is lifted sooner or later.
- Unidirectional blocks are treated as permanent obstacles. It is not possible to establish a path for a train run (e.g. via **AutoTrain™**), that passes a block in a (permanently) disabled direction.

If a certain track section is intended to be always used in a certain direction of travel, then define the according block as unidirectional. If you want to prevent trains from passing a block in a certain direction of travel for a limited period of time only during operation, then use an entry lock. An entry lock can for example be used to lock the opposing entry of a bi-directional single-track section, which is currently occupied by a train, against opposing trains. Using unidirectional blocks for such track section would not be adequate, because this would not allow to operate the track section alternately in both directions.

Locking the exit of Blocks

Each exit of any block can be temporarily locked during operation. A block cannot be left through a locked exit. Trains may enter such blocks and may stay there, but they cannot leave a block through a locked exit.

It is possible to lock either exit of each block individually and independently from the opposite exit.

Please note that locking of a block exit affects all trains.

5.5 Train Tracking

TrainController™ is able to indicate the positions of your engines and trains on the computer screen. This is always and automatically done in the screens of the *Visual Dispatcher*, such as the main block diagram or the particular schedule diagrams.

The *block symbols* in the switchboard also display the state of the associated block and optionally the name and/or image of the train that is located in this block.

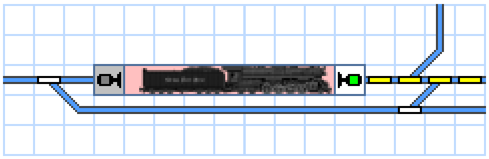


Diagram 105: Block Symbol in the Switchboard

Train Tracking

B

The *Visual Dispatcher* uses the *main block diagram* to perform automatic train tracking.

Whenever a block is reported as occupied, because one of the indicators assigned to it is turned on, then the *Dispatcher* checks, whether there is an appropriate train in an adjacent block. An adjacent block is a block that is connected with the current block with a route in the *block diagram*.

If there is such train, then the train is moved to this block. This is done by automatic assignment of the train to the new block and releasing of the previous block.

As a result of this movement the name and/or image of the engine or train appears in the block symbol of the related block in the *Visual Dispatcher*. Additionally, the train disappears from the symbol of the previous block. If there are *block symbols* in a switchboard window associated with these blocks, then the train movement will also be shown in these symbols.

If there is more than one train located in adjacent blocks, then the Dispatcher tries to determine the most probable candidate. For this calculation the speed of each train and the direction of travel, if known, or the occupancy state of each adjacent block is taken into account.

In order to achieve precise results it is important to assign the initial position and orientation of each train correctly. Additionally, you should always ensure that the software is able to track the direction and speed of each train. The control of trains that you operate with the throttle of your digital system should properly be assigned to the digital system (see 3.7, “Passing control between Computer and Digital System”).

Train tracking can also be disabled for certain blocks or temporarily for the complete layout. This is useful for areas or in situations, when you put new engines or trains manually on the physical track and you want to avoid unintentional train tracking caused by the resulting occupancy messages.

!

Attention: train tracking is turned on by default. Unintended triggering of indicators, that are assigned to blocks, might cause train assignments to be moved in the block diagram. If this is not desired in certain situations then train tracking can be (temporarily) disabled for the complete layout.

- Under the conditions listed below train tracking works for each engine or train on the layout, which has been previously assigned to a block.
- The initial assignment of trains to blocks can be done manually or automatically by train identification (see section 15.1, “Train Identification”). Train identification rids you from performing the initial assignment manually; train identification is, however, not a prerequisite of train tracking.
- Train tracking is based on the *block diagrams* of the Visual Dispatcher and follows the specified routes between the blocks. The tracking of manually operated trains, such as those trains that you control with the throttle of your digital system, is only possible, if you create an appropriate main block diagram, that contains the proper routes between your blocks.

!

For train tracking of an engine it is important that the software knows the direction and speed of a running engine. If you want to control an engine with a handheld of your digital system under simultaneous train tracking, then it may be necessary to assign control of the engine to the digital system before you do this (see section 3.7, “Passing control between Computer and Digital System”).

Tracking of Back and Forth Shunting

During normal train tracking train movements are only tracked from a block to another block when the two blocks are adjacent, i.e. if they are connected directly by a route.

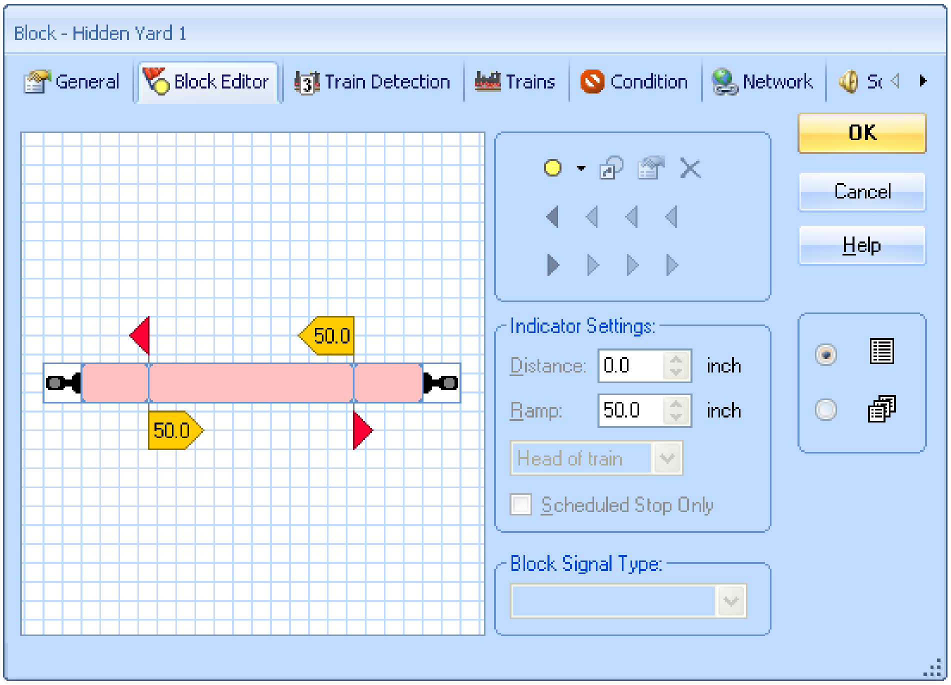
In **TrainController™** **Gold** , there is also the possibility to track trains from one block to another, when the train changes its direction in a turnout area during a shunting movement (‘zigzag’). If, for example, an engine moves from a track in a station to a parallel track, then it can be tracked to the parallel track even if it changes its direction in the turnout area between the blocks. In this case, both blocks are not connected directly by a route.

Tracking of such movements is enabled with the **Track Zigzag Moves between Blocks** command in the **Block** group of the **Operation** tab.

Only simple zigzag moves can be tracked, however. That means, if the engine changes the direction of travel more than once after leaving the first block and before it enters into the second block, this movement cannot be tracked. After a change of direction in the turnout area, a shunting locomotive should therefore re-enter into a block, to be located again. From here it can then start, if required, the next stage of the zigzag move and so on.

5.6 Blocks and Indicators

For proper operation the *dispatcher* must be able to detect, whether a train occupies a specific section of your railroad or when a train passes a specific point on your railroad. This detection is done with *contact indicators*.



Indicator Settings:
Distance: 0.0 inch
Ramp: 50.0 inch
Head of train
Scheduled Stop Only
Block Signal Type:

Diagram 106: The Block Editor

In order to establish a block, *contact indicator symbols*, which represent the track sensors located in this block, are created and assigned to the block. This is done with the *block editor*, which is displayed in Diagram 106. If at least one of the indicators contained in a block is turned on, then the block is assumed to be occupied. The actual layout positions of the sensors assigned to the block determine also the location of the block on your railroad.

The block editor shows an edit area with the current configuration of the block. Contact indicators are displayed as red rectangles in the center of the editor. Usually these rectangles represent the occupancy sections associated with each indicator (in the case of occupancy sensors) or the point in the block, where the indicator is triggered (in the case of momentary contacts like reed turnouts, mechanical contacts, etc.). Each physical sensor located in the block is represented by one indicator rectangle. The location and size of these indicator rectangles can be customized and do not have any impact for the operation of the program, but if properly arranged they can visualize the section, that is covered by a specific sensor.

In order to have control over the exact location, where a train will stop or change its speed inside a block, certain sections can be marked as stop, brake or speed sections (see section 5.7, "Stop, Brake, Speed and Action Markers") or combinations of these.

To establish a block on your railroad, it is necessary to install the required sensors. Depending on the principle of the contact sensors used it may be necessary to electrically insulate the track section belonging to each contact sensor from adjacent sections. Whether electrical insulation is necessary or not depends solely on the contact sensors being used. The software does not require electrical insulation of your blocks.

- The software does not require that a block is electrically insulated from other blocks. However, the sensors used might require this.
- Blocks usually contain several indicators. If these indicators represent isolated or separate track sections, then several track sections are contained in the same block (see also 5.8, "Arranging Indicators and Markers in a Block").
- The same indicator <u>cannot</u> be assigned to several blocks. In particular you should install your sensors on your layout in a way such that each sensor section is associated with at most one block. Further, if you use a train identification system (see 5.5, "") then each train identification section or zone, respectively, must be associated with at most one block.
- Even though it is possible to assign indicator symbols to a block, which are already contained in other windows, this feature is mainly provided for reasons of compatibility to previous software versions or for very specific purposes. Usually you should create each indicator symbol, which is contained in a block, with the block editor displayed in Diagram 106.

5.7 Stop, Brake, Speed and Action Markers

B

A block is established by creating and assigning one or more *indicators* to it. If at least one of these indicators is turned on, then the block is assumed to be *occupied*. The indicators are used for indication of occupancy.

It may be required, that a train has to stop or to change its speed when passing a certain block. This is for example the case, when the block ahead is not available, when the train will stop inside the block for a certain amount of time or when another speed limit applies in the subsequent block. The exact locations, where trains will stop or change their speed inside the block are determined by marking certain indicators with *stop*, *brake* or *speed markers*.

Stop and Brake Markers

Let us assume that a train approaches a certain block. That means, that none of the assigned indicators was activated before and that at least one of these indicators is activated now. The block is now marked as occupied and the train continues with unchanged speed. If the train reaches a location in the block, which is marked by a brake marker for the current *direction of travel* (see section 5.3, “Direction of Travel vs. Engine Orientation”) and the train has to stop inside this block, then the train is decelerated to its *threshold speed*. The braking ramp can be set as desired individually for each brake marker. If the train reaches a position, which corresponds to a stop marker for the current *direction of travel* and the train has to stop inside this block, then the train is stopped here.

A stop marker determines a point in a block, where trains stop. Stop markers are represented in **TrainController**™ by red arrowheads pointing to the direction of travel, in which they apply. A brake marker determines a point in a block, where trains, that have to stop in a block, begin to slow down. Brake markers are represented in **TrainController**™ by yellow arrowheads.

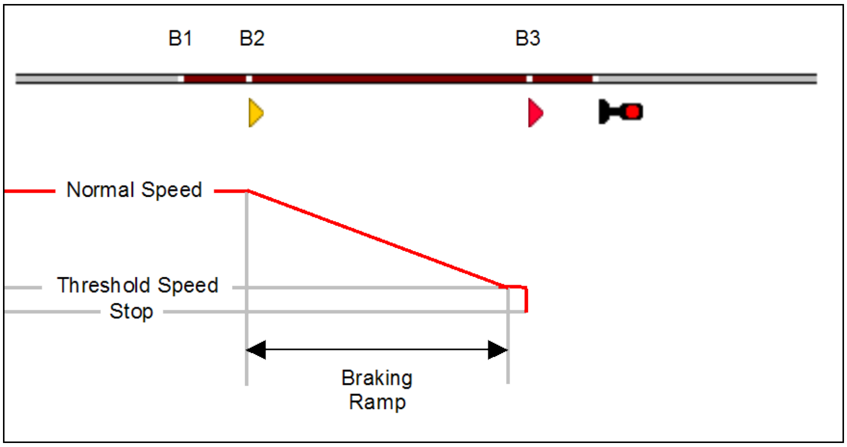


Diagram 107: How Brake and Stop Markers work – Occupancy Sensors

Diagram 107 shows a block, which is equipped with three occupancy sensors. The left entries to the sensed track sections are labeled with B1, B2 and B3.

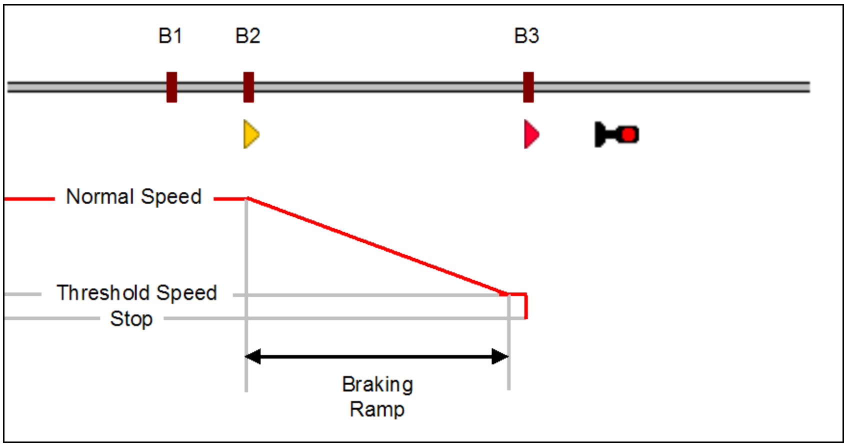


Diagram 108: How Brake and Stop Indicators work – Momentary Track Contacts

An alternative, but for this discussion almost equivalent situation is shown in Diagram 108. It contains a block equipped with momentary contacts. These contacts are labeled with B1, B2 and B3, too.

B3 is marked with a stop marker (▶) effective for trains travelling to the right. B2 is marked with a brake marker (▷) effective in the same direction. B1, which applies only to the first diagram, is neither marked as brake nor as stop marker. B1 is used only for occupancy detection.

The red line shows the speed of the train. It is assumed that the train will stop in this block, at B3. When the train enters the block at B1 nothing happens, because B1 is only used to report the entry into the block. When the train reaches B2, it is decelerated to its threshold speed. The braking ramp can be specified individually for each brake marker. After deceleration the train proceeds at threshold speed until it reaches B3. When the train reaches B3, it is stopped immediately.

Diagram 106 shows the same situation as Diagram 107 configured in the block editor.

If the train does not have to stop in this block, then it passes all indicators and markers without any speed change.

If the stop marker B3 is missing, then the train will run with normal speed to B2 and stop there. If no stop marker is assigned to a block, then the first appropriate brake marker is used as stop marker. If B1 is the only indicator and there are no markers in the block, then the train will be stopped immediately at B1. If no marker is assigned to a block, then the first triggered indicator implicitly defines a stop marker. If necessary, a train is stopped in a block anyway, even if only indicators and no brake and stop markers are assigned.

! **This examples also illustrates that proper operation of brake markers requires correct adjustment of threshold speed of each affected train! If this is not the case, the train will be decelerated to an undefined threshold speed. Normally this speed will be too low to run the train properly and the train will stop before reaching the stop marker.**

A stop, brake or speed marker is always associated with an indicator. Usually this is a contact indicator, that represents an occupancy section or momentary contact installed on your layout. A stop, brake or speed marker is valid for a particular direction of travel. The marker usually takes effect, when a train running in this direction enters the associated occupancy section or touches the associated momentary contact. It is also possible to specify a distance for each stop, brake or speed marker. In this case the marker takes effect, when the train has passed this distance after entering the associated occupancy section or after touching the associated momentary contact. Such markers are called *shifted* stop, brake or speed markers.

While each marker is always associated with exactly one indicator, it is possible to use the same indicator with several markers. The same occupancy section, for example, can be used to slow down passing trains (brake marker) and to stop trains in a certain distance behind the border of the section (shifted stop marker). This is accomplished by adding a brake and a stop marker to the same contact indicator, which represents the occupancy section, and by specifying an appropriate distance for the stop marker. It is even possible to add more than one brake, stop or speed marker to the same indicator or to the same block. The assignment of two stop markers to the same indicator, for example, is useful, if different trains will stop at different positions (e.g. advance of freight trains to the block signal at the end of the block while passenger trains stop at the middle of the platform). For this purpose the validity of a stop, brake or speed marker can be limited to certain trains.

! **Please note that a brake marker is only effective if the train has to stop in the same block. As a consequence brake and stop markers that belong together must be contained in the same block.**

The same indicator can be marked with stop or brake markers for one or both directions of travel. It is even possible, that a certain indicator is associated with a stop marker in one direction and with a brake marker in the opposite direction.

It is recommended that the sensors corresponding to stop markers are located at positions, which ensure, that even long trains completely fit into the block.

If an engine or train passes a sequence of blocks and a certain block is not available or must be passed at restricted speed, then the train is stopped or decelerated in the previous block. Brake and stop markers control, if a train may exit a certain block. For this reason **TrainController™** always assumes, that stop markers are usually located near the exit of each block with reference to the direction of travel in which they are effective.

When a train enters a block, the dispatcher checks if there is a route before the next block. In this case, the route is activated if this has not already been done. If the activation is not completed when the train reaches the brake or stop marker in this block then the train is decelerated or stopped, respectively, in order to wait for the activation of the route. If there is only one indicator without any markers in this block, then the same indicator is used for indication of entry into the block, activation of the route and also implicitly as stop marker. In this case, the train is always stopped briefly because the activation of a route takes some time.

! **To avoid such stops it is important to use different locations in the block for brake and stop markers.**

Speed Markers

A speed marker determines a point in a block, where a speed limit of the subsequent block is applied. Speed markers are represented in **TrainController™ Gold** by green arrowheads. If restricted speed applies in a certain block, then the train is decelerated at the first speed marker of the previous block. If no speed marker is assigned to this previous block, then the train is decelerated at the brake or stop marker, which takes effect first.

It is possible to specify a braking ramp for each speed marker, too. This ramp works in the same way as braking ramps of brake markers (see above).

TrainController™ assumes that a train ready to be started is located with its head near the exit of its current block. It is also assumed that the train will exit its current block and enter the next block just after being started. For this reason all speed conditions up to the next block are applied at the start of a schedule.

! **All speed changes take place at the appropriate markers of the previous block.**

Action Markers

All markers described so far are also able to perform additional operations, e.g. to toggle the headlights of the passing train or to open a crossing gate, etc. These markers, however, may also change the speed of the train. If it is desired to perform operations while ensuring, that the speed of the passing train remains unchanged, action markers can be used. An action marker determines a point in a block, where operations can be performed without affecting the speed of a train. In a certain sense action markers comprise the common sub set of all other markers, namely the ability to perform operations as well as certain other properties, but unlike all other markers they do not have the built-in ability, to affect the speed of passing trains.

Action markers are represented in **TrainController™** **Gold** by gray arrowheads.

5.8 Arranging Indicators and Markers in a Block

B

This section describes the different types of sensors and how to use them to operate a block.

Arranging Momentary Track Contacts and Occupancy Sensors in a Block

B

In the following it is assumed that the track section between the turnouts in the following diagrams is a block. Several methods of arranging indicators and markers in a block are discussed below. The pros and cons of each method are outlined as well.

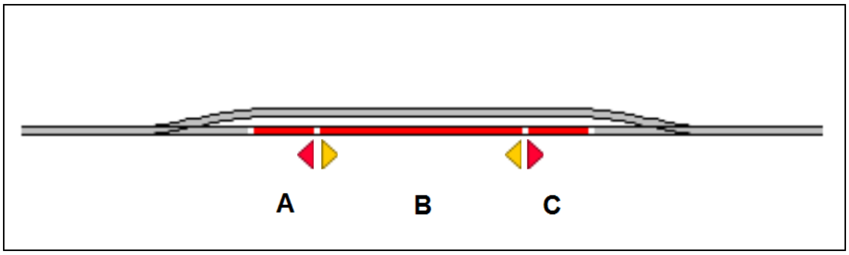


Diagram 109: Block with three occupancy sensors

Diagram 109 shows a block equipped with three occupancy sensors. Each of these sensors is associated with a contact indicator in the software called A, B and C. All indicators are assigned to the same block in the software. The block is indicated as occupied as soon as a train enters section A from the left or section C from the right. The block remains occupied until the train leaves the opposite section. A stop marker has been defined for indicator A for trains running to the left, C is marked with a stop marker for trains running to the right. The trains are stopped at the boundary between B and A or C, respectively. The indicator B is associated with two brake markers for both directions. Trains begin to slow down when entering B from either direction. The sections A and C should be long enough such that each train is safely stopped before touching one of the turnouts. On the other side the longest train should completely fit into the block when being stopped. For this reason the boundaries between B and A or C, respectively, where trains are stopped, must be located close enough to the boundaries of the complete block.

The configuration displayed in Diagram 109 is the optimal and recommended solution. The block is indicated as occupied as long as a train is located in one of the three occupancy sections. Additionally it would be even possible to distinguish in which of the three sections A, B or C a train is located. Lost or parked cars can be detected, too, if they cause an occupancy indication. Pushed trains can also be treated, too, if the first pushed car generates an occupancy indication. This method requires the effort, however, of cutting the rails at the boundaries of each occupancy section.

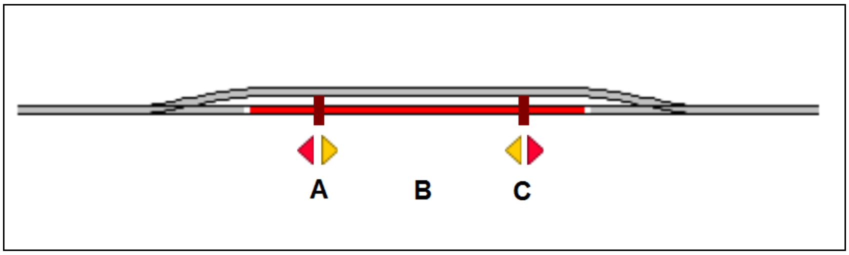


Diagram 110: Block with an occupancy and two momentary sensors

Diagram 110 shows a block equipped with one occupancy (B) and two momentary sensors (A and C). Each of these sensors is associated with a contact indicator in the software called A, B and C. All indicators are assigned to the same block in the software. The block is indicated as occupied as soon as a train enters section B from any direction. The block remains occupied until the train leaves section B. The indicator A additionally corresponds to a stop marker for trains running to the left, C is marked with a stop marker for trains running to the right. Both indicators are additionally marked with brake markers for the opposite direction, respectively. The location of A and C should ensure, that each train is safely stopped before touching one of the turnouts. On the other side the longest train should completely fit into the block when being stopped. For this reason A or C, respectively, where trains are stopped, must be located close enough to the boundaries of the complete block.

Application of Diagram 110 has take into account, that momentary contacts tend to be less reliable than occupancy sensors.

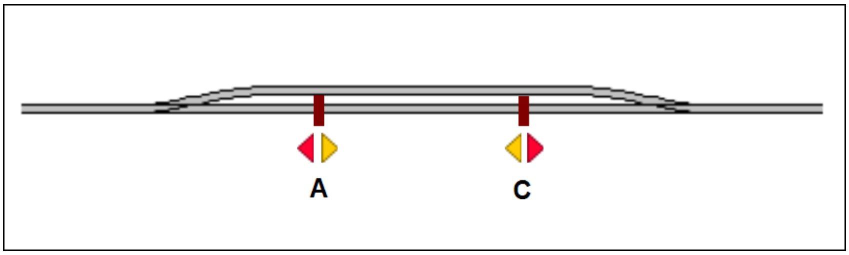


Diagram 111: Simple Block with two momentary sensors

Diagram 111 shows a simple configuration of a block equipped with two momentary sensors. Both sensors are associated with a contact indicator in the software called A and C. Both indicators are assigned to the same block in the software. The indicator A is additionally marked with a stop marker for trains running to the left, C is associated with a stop marker for trains running to the right. Both indicators are additionally marked with brake markers for the opposite direction, respectively. The location of A and C should ensure, that each train is safely stopped before touching one of the turnouts. On the other side the longest train should completely fit into the block when being stopped. For this reason A or C, respectively, where trains are stopped, must be located close enough to the boundaries of the complete block.

The configuration displayed in Diagram 111 is very simple and inexpensive but has also some disadvantages. Block occupancy is not indicated. As long as the block is reserved for a train located inside this block this causes no major problem, because the dispatcher will not allow another train to enter this block. But certain measures are to be taken to avoid premature reservation of this block for another train when a train leaves the block. There is also a disadvantage for passing trains. Let us assume that a train is passing the block from the left to the right and that a route is to be activated before the block ahead, to the right of this block. As soon as the passing train enters the block at A the route is activated. In the same moment the train begins to slow down, because A defines also a brake marker and the train has to wait, until the route is reported to be activated which takes time. This could be avoided by adding an additional contact as shown in the following diagram:

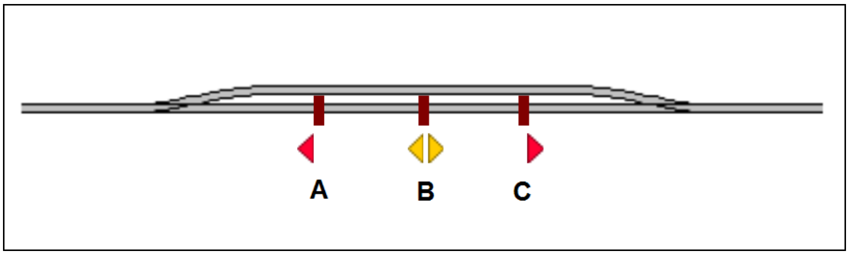


Diagram 112: Block with three momentary sensors

In Diagram 112 the indicator A defines a stop marker for trains running to the left, C acts as stop marker for trains running to the right. Indicator B is marked with brake markers for trains running in both directions. In this configuration block occupancy is not indicated, too, and as in Diagram 111 certain measures are to be taken to avoid premature reservation of this block for another train when a train leaves this block. But trains can pass this block without any speed changes, even if there is a route to be activated before the block ahead – provided the distance between A and B or C and B, respectively, is large enough such that the route can be activated after passing A or C, respectively, and before reaching B.

All examples discussed so far can be applied for blocks passed by trains in both directions. The configuration can be made simpler if trains pass a block only in one direction. This is shown in the following:

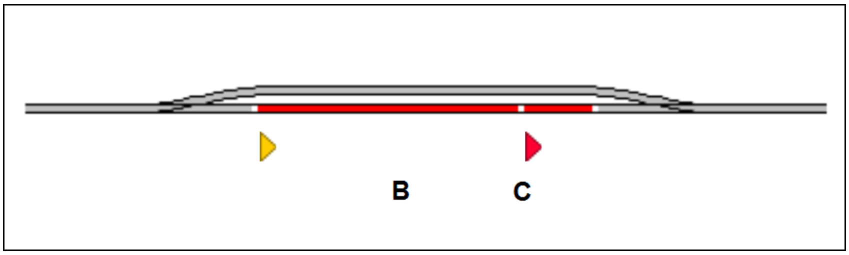


Diagram 113: Block with two occupancy sensors

Diagram 113 has been derived from Diagram 109 by eliminating sensor A. It is assumed that the block is only passed from the left to the right. B acts as brake marker and C as stop marker for trains running to the right.

The different configurations discussed in this section are only examples. Configurations similar to Diagram 113 can also be made with momentary contacts instead of occupancy sensors or with a mixture of both types similar to Diagram 110. One can think also of other configurations. There is no best way to setup a block. The optimal solution does not only depend on technical requirements but also which equipment you already have and how much money you want to spend on new equipment.

One Sensor per Block: Shifted Brake or Stop Markers

In the examples discussed so far, all locations, where trains stop or begin to brake are identical to the entry of an occupancy section or to the point, where a momentary track contact is triggered. In Diagram 112 we even installed an extra sensor to isolate the location, where the entrance into the block is reported, from the location, where the train begins to brake to gain time for activation of subsequent routes.

But it is not essential to install extra sensors for this purpose. It is also very easy to specify, that a stop marker is located in a certain distance from the point, where the associated sensor is turned on. This is done by specifying a *distance* for such marker. This creates a *shifted stop marker.*

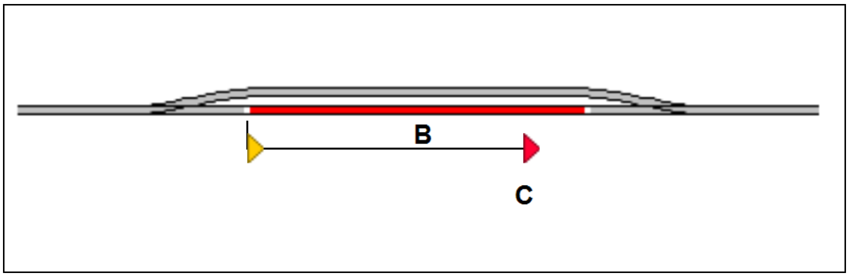


Diagram 114: Shifted Stop Marker

If your trains run very precisely and have been profiled accordingly, then it is not necessary to install a separate sensor to mark the stop point C. Instead it is possible to mark the occupancy sensor B with a brake marker and a shifted stop marker for stop point C.

Assume that in the above example the desired stop point C is located in a distance of 50 inches from the left border of the occupancy section B. If it is desired, that trains decelerate and stop within 50 inches after entrance into B, then contact B is marked with a (shifted) stop marker with a distance of 50 inches. Additionally a brake marker should be added to B with a braking ramp of just under 50 inches to accomplish smooth deceleration.

If a train, that has to stop in this block, enters the occupancy section B from the left, it will be decelerated to threshold speed within 50 inches of the left border of B. When it arrives at the point C, which is 50 inches away from the entrance to B, the train will be stopped automatically.

In other words: the shifted stop marker associated with B works exactly like an additional sensor marked with a stop marker located 50 inches behind the entrance into section B.

This principle can also be applied to the opposite direction. In this way one single occupancy sensor (sensor B in this example) can be marked with brake markers and shifted stop markers for both directions. For reasons of simplicity the markers for the opposite direction have been omitted in Diagram 114.

In Diagram 114 each train, that has to stop in this block, will begin to slow down just as it enters the track section B. As mentioned earlier, this can cause trains to slow down temporarily, if a subsequent route is to be activated. To avoid this, it is possible to specify a distance for the brake marker, too, which leads to a *shifted* brake marker. The principle is shown in the diagram below:

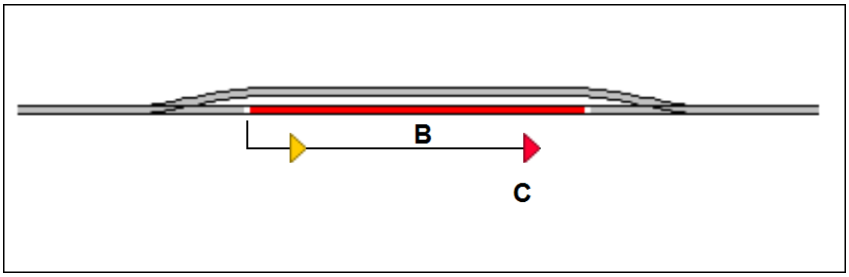


Diagram 115: Shifted Brake and Stop Markers

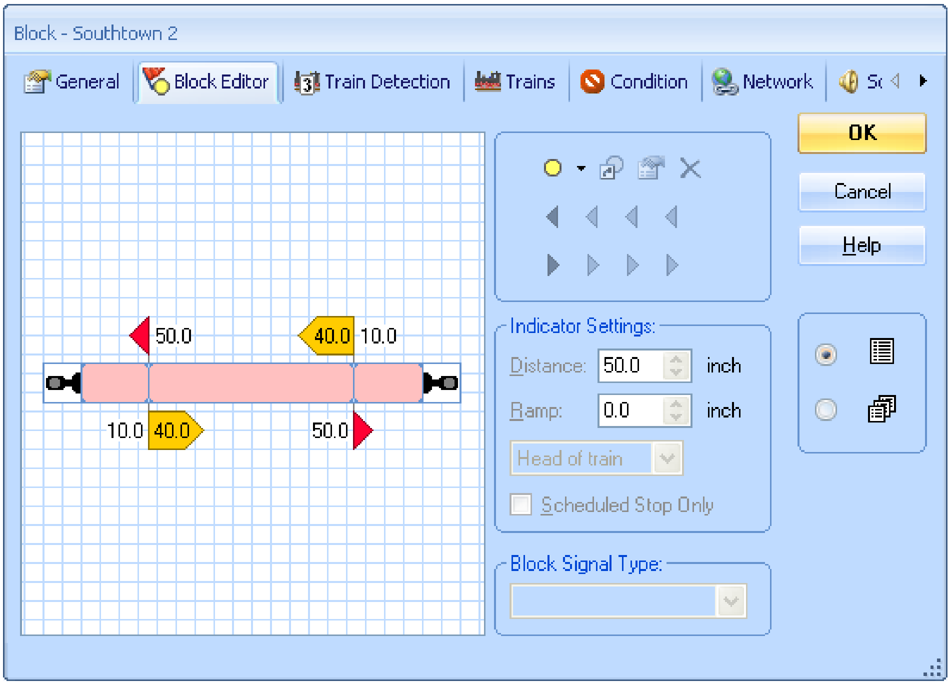


Diagram 116: Editing Shifted Brake and Stop Markers in the Block Editor

Diagram 116 demonstrates, how shifted brake and stop markers are arranged for both directions in the block editor. Trains, that will stop in this block, will begin to slow down 10 inches behind the entrance of the occupancy section. The braking ramp is set to 40 inches, thus trains will reach threshold speed 50 inches behind the entrance, where they also stop, because this is exactly the distance of the shifted stop markers.

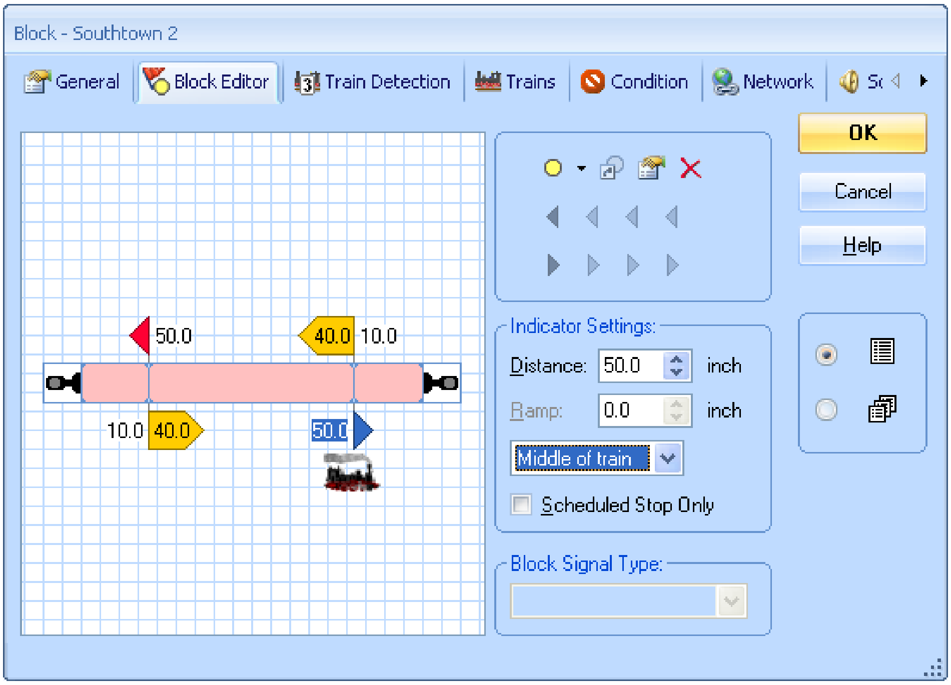
The complete configuration displayed above can be created with the block editor for both directions in no time at all with a few mouse clicks.

! **Shifted brake or stop markers allow operation of a complete block with one single sensor and indicator symbol. Proper Functioning of shifted brake and stop markers require appropriate profiling of the affected locomotives.**

Stopping a Train in the Middle of a Platform

With the block editor it is quite simple and straightforward to accomplish a train stop in the middle of a platform.

- Create a block, that represents the track passing the platform.
- Open the block editor and create an indicator, that represents the sensor installed in this block.
- Mark the indicator with a stop and a brake marker for each affected direction of travel.
- Specify appropriate distances for these markers and an appropriate ramp for the brake marker.
- Apply the option **Middle of train** to the stop marker.



Indicator Settings:
Distance: 50.0 inch
Ramp: 0.0 inch
Middle of train
[ ] Scheduled Stop Only

Diagram 117: Specify a shifted Stop Marker for Centred Stop

That’s it. Each train, that has to stop in this block will automatically stop centred with respect to the location determined by the stop marker.

Variable Stop Locations in a Block – Stopping for Coupling and Line-Up

X

Using the settings **Head**, **Middle** or **Tail of Train**, it is possible to stop trains at a fixed location such as stop at a signal or in the center of a platform. For shunting and exchange of locomotives, it is often necessary to let the train stop at a certain point with a particular vehicle, or to take into account the length and position of the vehicles already waiting in a block, when the train enters the block.

For this purpose it is possible in **TrainController™ Gold**, to specify a formula for each marker in a block for the calculation of variable distances for markers and variable braking ramps.

Such a formula is set up according to the usual rules of mathematics. It can contain numbers and the operators **+**, **-**, ***** and **/**. As usually multiplication and division first, then addition and subtraction. But it is also possible, with the help of brackets **(** and **)** to give priority to certain operators. The numbers may also contain decimal points, in order to specify even a fraction of the basic units of centimeters or inches.

In addition, each formula can contain wildcards. Thus, the length of the train entering into the block or the length of the vehicles already located there can be considered. It is also possible that a wildcard refers only to a part of a train or to individual vehicles.

Wildcards begin with one of the characters **%**, **#** or **?**.

%	The wildcard represents the total length of the vehicles, which this wildcard refers to.
#	The wildcard represents the number of the vehicles, which this wildcard refers to.
?	The wildcard is replaced by the value **1**, if one or more vehicles refer to it. If no vehicle refers to this wildcard, then the value is **0**.

Next follows the letter **B** (for block) if the wildcard refers to vehicles, which are already located in the block. If the character **B** is not specified, then the wildcard refers to the train currently entering the block.

Next is a type code that determines the vehicles, which the wildcard refers to. The following types exist:

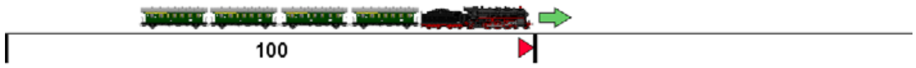
• **A**: All vehicles of the train
• **L**: pulling locomotives
• **H**: pushing locomotives (‘helpers’). These are all locomotives in a train set, which are separated from the pulling locomotives by at least one car. If there are no cars in a train, then there are no pushing locomotives. If there are only cars in front of the first locomotive (seen in the current direction of travel), then there are only pushing, but no pulling locomotives. %**L** is thus always 0 for pushed trains.
• **C**: car

If the type code is followed by the letter **R**, then the wildcard refers to the last vehicles of the specified type (seen in the direction of the entering train). If **R** is not specified, then the wildcard refers to the first vehicles in driving direction.

At the end of the wildcard an optional number can follow. This number indicates how many vehicles the wildcard refers to. If no number is specified, all vehicles of that type are meant.

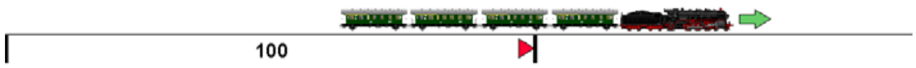
Examples:

- **100**:



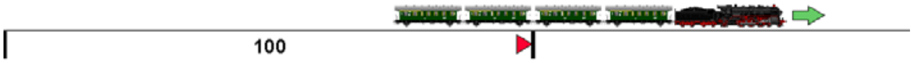
This formula works in the same way as the standard option 100 cm with **Head of Train**. The result of a formula is namely always applied to the head of the train entering into the block.

- **100+%A/2**:



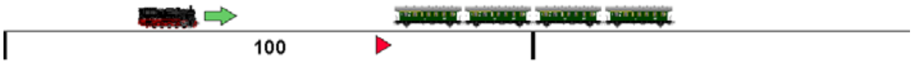
This formula can e.g. be used to stop a train in centered at 100 cm. This formula works in the same way as the standard option 100 with **Middle of Train**.

- **100+%L+%C/2**:



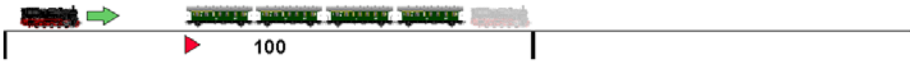
This formula can e.g. be used to stop the cars of a train centered at 100 cm. It does not matter how long the pulling locomotive or locomotives are.

- **100-%BC/2**:



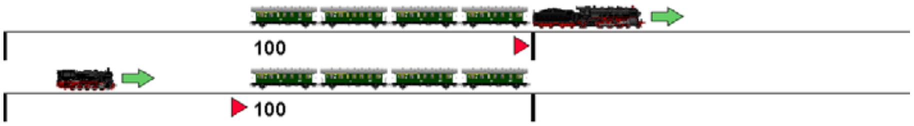
If the cars of a train were stopped centered at 100 cm with the formula specified in the preceding example, then you can use this formula to stop with a locomotive at the other end of the parked cars.

- **100-%L-%BA**:



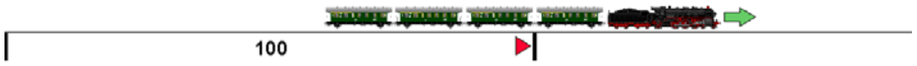
This formula is useful when moving a locomotive to the opposite end of the train. If the incoming train was stopped with its head at 100 cm with a normal stop marker, then the locomotive can be stopped with this formula at the other end of the cars.

- **100+%L**
  100-%BC:



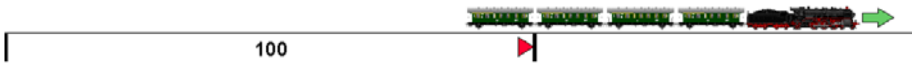
These formulas are useful for the change of locomotives, when the new locomotive will stop at the rear of the waiting cars and the lengths of the locomotives involved in the exchange may be different. With the first formula, the first car of the entering train is stopped at 100 cm. With the second formula, the new locomotive is moved to the other side at the end of the waiting cars. This works even if the original locomotive remains in the block.

- **100+%L+%C1**:



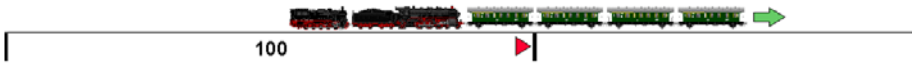
Using this formula, a train can be separated with an uncoupling track located at 100 cm behind the first car.

- **100+%A-%CR1**:



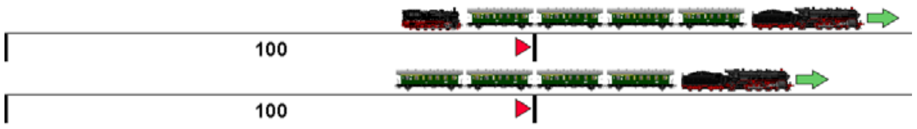
Using this formula, a train can be separated with an uncoupling track located at 100 cm before the last car.
In order to understand a formula with plus and minus signs, it is helpful first to mentally move the train forward according to the positive part of the formula and then back according to the negative part. In the above example you mentally move the train initially to the right according to its total length. The rear of the train is then (mentally) located at the red mark. Then you move the train back by the length of the last car.

- **100+%A-%H-%CR1**:



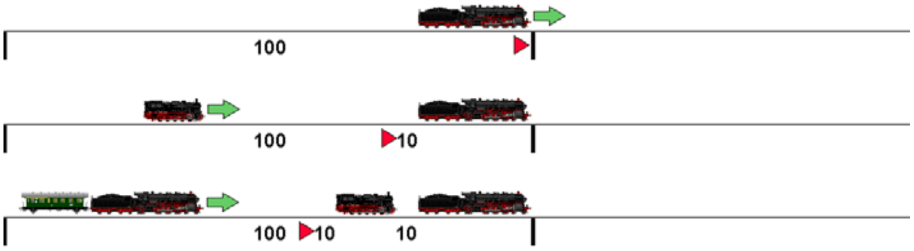
Using this formula, a pushed train can be separated with an uncoupling track located at 100 cm before the first car.

- **100+%A-%AR2:**



Using this formula, a train can be separated with an uncoupling track located at 100 cm before the last two vehicles, regardless, whether these vehicles are locomotives or cars.

- **100-%BA-?BA*10:**



This formula can be used for line-up of vehicles in a block (see also page 253). ?BA*10 describes the gap between the lined-up vehicles.
When the first train enters the block, %BA and ?BA are both 0 and the train stops at 100 cm.
When the second train enters the block, %BA equals to the length of the first train and ?BA is 1. This causes the second train to stop 10 cm behind the first.
When the third train enters the block, %BA equals to the total length of the waiting vehicles (including the gap between them) and ?BA is still 1. This causes the third train to stop 10 cm behind the second.

The interpretation of all distances as cm or inches depends on the setting of the **Metric Units** command in the **View** tab. If **Metric Units** is turned off, then all values are interpreted as inches. The above examples apply accordingly by using inches as unit instead of cm.

The variable wildcard **%V** (see page 328) can be used in these formulas, too. It has a specific meaning here. These wildcards are replaced in the first pass of the calculation by the content of the corresponding variable. The resulting formula is then calculated in the second pass in the usual manner like a formula without **%V** wildcards.

Stopping different Trains at different Positions

Passenger trains will stop in the middle of the platform while freight trains will advance and stop at the end of the block? This can be easily accomplished with **TrainController™ Gold**, too:

- Add a further stop marker to the configuration described in the previous section.
- Apply **Head of train** to this additional marker.
- Select this stop marker and call the **Properties** command in the toolbar located in the upper right corner of the block editor.

- Select the (freight) trains, to which the additional stop marker will apply.
- Repeat the last two steps for the stop marker created in the previous section, which is responsible for stopping trains in the middle of the platform, and specify the (passenger) trains, to which this marker will apply.

If this has been done correctly all passenger trains will automatically stop centered with regard to the location determined by the first stop marker, while freight trains will advance and stop at the position defined by the second block marker.

It is not only possible to specify, that a certain marker is only valid for specific trains, it is also possible to specify, that a certain marker is only triggered in conjunction with certain schedules. These schedules are specified in a similar way like the associated trains. Limiting markers to specific schedules is useful, for example, if the same train will stop at different locations depending on the currently executed schedule. It can be also useful in cases, where it is simpler to select a few schedules rather than a plurality of trains in order to specify different stop points for different trains.

If a certain marker is limited to specific trains and to specific schedules, then the marker is only triggered, if one of these trains passes this marker under control of one of these schedules. The marker remains turned off, if the train does not match or if the schedule does not match or both.

It is possible to define an arbitrary number of stop, brake and speed markers in a block. It is also possible to specify, that a certain marker takes effect only under certain conditions. In this way it is for example possible to define different stop points for different operational situations.

The features described above and their combinations provide virtually unlimited possibilities to determine, where trains will stop, slow down or change their speed in the particular blocks.

Markers for scheduled Stops vs. Markers for unscheduled Stops

It is possible to specify differing brake and stop markers for scheduled stops and unscheduled stops in the same block.

Markers specified for scheduled stops are only triggered, if the train has to perform a scheduled stop in this block. If the train stops for another reason, e.g. because the exit of the block is currently locked, then such marker is not triggered.

If at least one stop marker in a block is specified for scheduled stops, then all other stop markers in this block, which are not specified for scheduled stops, are only triggered for unscheduled stops. This applies accordingly to brake markers, too.

This feature can be used to specify different stop points for the same train in the same block, which depend on, whether the train has to perform a scheduled stop or an unscheduled stop in this block. A good example is a passenger train, which will sometimes perform a scheduled stopover in the middle of a certain block, and which will pass this block without scheduled stop otherwise. This can be accomplished by specifying a stop marker located in the middle of the block for scheduled stops and another stop marker at the location of the block signal for unscheduled stops. A passing train, that has to perform a scheduled stop in this block, will stop in the middle of this block. In other cases, when this train does not have to perform a scheduled stop, but must stop here for any other operational reason, the train will stop at the block signal.

5.9 Block Signals

General

Traffic Blocking is used on real railroads to prevent two trains from running into each other by dividing the track into sections protected by signals. These signals (here called *block signals*) indicate to a train whether it can enter the block which begins beyond the signal. If the block ahead is occupied the driver of a train approaching the signal protecting that block sees a red stop light. If the section in front is unoccupied and the train has permission to enter it the driver sees a green signal light. In addition to the signal for the next block the driver is usually also presented with a distant signal which indicates the status of the block beyond that which is being entered. If the distant signal indicates green, it means that the subsequent block is free to be entered; otherwise the block ahead is occupied and the train should proceed into the next block with caution and be prepared to stop at a red light.

When a train is running under control of the *Dispatcher*, **TrainController™** automatically calculates signal aspects taking into account the availability of *blocks* and *routes* in front of the train. These signal aspects are displayed in the block diagrams and as home and distant signals in the Train Window (see chapter 3, “Train ”). The signals indicate, whether the current block may be left and how the following block must be entered. The *brake* and *stop indicators* assigned to a block ensure that a train is stopped at the appropriate location. Since **TrainController™** assumes that the brake and stop indicators belonging to a block are located near the exit of the block, this is also assumed for the imaginary location of block signals.

TrainController™ displays the signal aspect currently valid for a block, when the first indicator assigned to this block is reached. It is possible to say: “The engineer is able to see the block signal at the end of a block when the train enters the block”.

Signal Aspects

TrainController™ uses five different signal aspects - each is associated with a specific color:

Color	Meaning
Red	Stop
Green	Proceed
Yellow	Proceed Restricted
White	Shunting
Grey	Signal not available

Table 3: Signal Aspects

For each train under its control the *Dispatcher* calculates the aspect of the next block signal and the distant signal. The signal aspect is calculated depending how the train is operated.

During shunting (see section 5.11, “Schedules”) white is indicated for all blocks reserved for this train.

When a train is executing a schedule the availability of the next two blocks in front of the *current block* of the train is calculated as home and as distant signal. If the train must not enter the block, then the signal of the previous block is set to “red”. If the train may enter the block, then the signal is set to “green”. If the block is available and reserved for traveling with restricted speed, then the signal is set to “yellow”.

Similarly the same signal aspects are valid for the *distant signal*, which indicates in advance, whether the train is allowed to leave the block behind the *current block* and how the next block behind this advance block must be entered.

“Grey” is used, if the other colors do not apply. This is also the case, if the train is not running under control of the *Dispatcher*.

The calculated state of the home signal of each block is displayed on the relevant side in the symbol of the block.

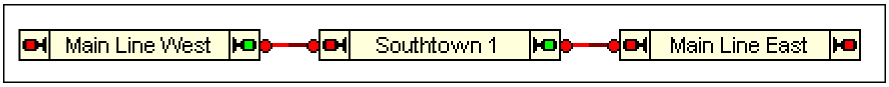


Diagram 118: Block Signals

In the example displayed above a train may leave “Southtown 1” and proceed to “Main Line East”. The signal symbol on the right side of the block shows green. The signal on the other side displays red, because it is assumed that the train must not enter “Main Line West”.

The states of the home and distant signal (if available) are additionally indicated in the cab signal controls of the Train Window (see chapter 3, “Train Control”), when a train is running under control of the *Dispatcher*.

How to use Signals on the Model Railroad Layout

TrainController™ does not need any signals on your model railroad to control trains. But for realistic operation it should be possible to indicate the calculated signal aspects with appropriate signal models on your model railroad. For this purpose it is possible to create two signals within each block, one for each direction of travel. These signals are called *integrated block signals*. Integrated signals can be linked to physical signal models on your layout. They are automatically operated according to the calculated signal aspects of the block they belong to.

! **These signals are only used for indication. They do not need any facilities to control trains, because the trains are controlled by the Dispatcher.**

It does also not matter, if the used signal models represent home or distant signals, because the models are only used for display. Selecting the appropriate signal model and location you are free to decide, where home and where distant signals will be visible. These signal models are of course operated dependent on the *direction of travel*. For this reason you can create a signal for each direction of travel.

How Block Signals Work

The following example shows for blocks A to D, which are subsequently passed by two trains.

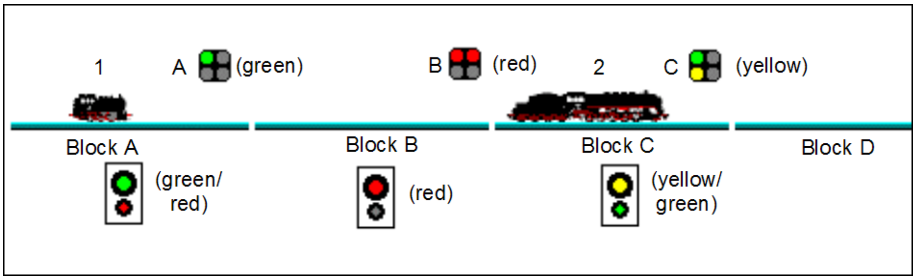


Diagram 119: Block Signals

The internally calculated signal aspects are indicated inside the black rectangles below the block name. These signals are also indicated as cab signals in the Train Window, while the train is inside the relevant block. Above the track there are signal elements controlled according to these aspects. For example the block signal labeled "B" is assumed to belong to block B.

Train 1 may enter block B but not block C, because block C is still reserved and occupied by train 2.

The calculated block signal for train 1 in block A is green, because train 1 may leave block A and enter block B without any restrictions. This is also indicated by signal A, which is assumed to belong to block A.

Since train 1 must not enter block C, the block signal of block B is calculated as red (in this case an distant signal is not indicated in the Train Window). This state is indicated by signal B of block B.

Train 2 may leave block C and enter block D only with restricted speed. For this reason the calculated block signal for train 2 in block C is yellow. This state is indicated by signal C, that belongs to block C.

Additional Notes

! The internal signaling system of **TrainController™ does not claim to simulate realistic signaling systems of the prototype**. For each block the software only calculates, whether a train may leave this block in the related direction and whether any speed restrictions apply. This calculation is only done for those blocks, that are currently in the focus of an active schedule.

By linking integrated block signals to signal models on your layout the internally calculated aspects can be made visible on the layout if desired. This simplified signaling system is easily arranged and meets the requirements of playful model railroad operation.

If a signaling system according to the rules of the prototype is desired, then this can be achieved by using the calculated block signals, reservation and occupancy states of related blocks, turnout positions as well as logical associations based on *triggers* and *conditions* as outlined in section 14.7, “Prototypical Signaling”.

5.10 Spontaneous Runs

After arranging the block system in **TrainController™** as outlined in the previous sections it is possible to run trains under full protection and routing of the computer. Put a locomotive on the track, assign its symbol to the associated block and call the menu command **Spontaneous Run**. The train will immediately start to move, provided that the route ahead is clear. It will then select an appropriate path and continue to travel, until it reaches a dead end or until the path ahead is blocked for another reason. At a dead end it will reverse automatically, if desired, and continue to travel in the opposite direction.

With this method routes can be treated in different manners. It is either possible to allow the computer to select and activate all routes requested by the train automatically. It is also possible to leave this to the human operator. In this case the train is stopped in blocks with at least one outgoing route, until one of these outgoing routes is selected and activated by the human operator.

If no further measures are taken, trains may run anywhere on your layout. By permitting certain blocks for certain trains only it is possible to direct trains to specific locations. Another way to control the path each train takes is the definition of schedules. This is outlined in the following section.

5.11 Schedules

B

Schedule Diagrams

After drawing your block diagram you will specify the desired train movements. This is done with the help of *schedules*.

Schedules describe how trains travel from selected starting blocks to destination blocks.

The base of each schedule is a *schedule diagram*. This diagram contains all blocks and routes of the main block diagram, that the train will use on its journey. This diagram can be displayed on the computer screen, too. This is done by displaying those parts of the main block diagram, that do not belong to the schedule, transparently in the background of the computer screen as shown below:

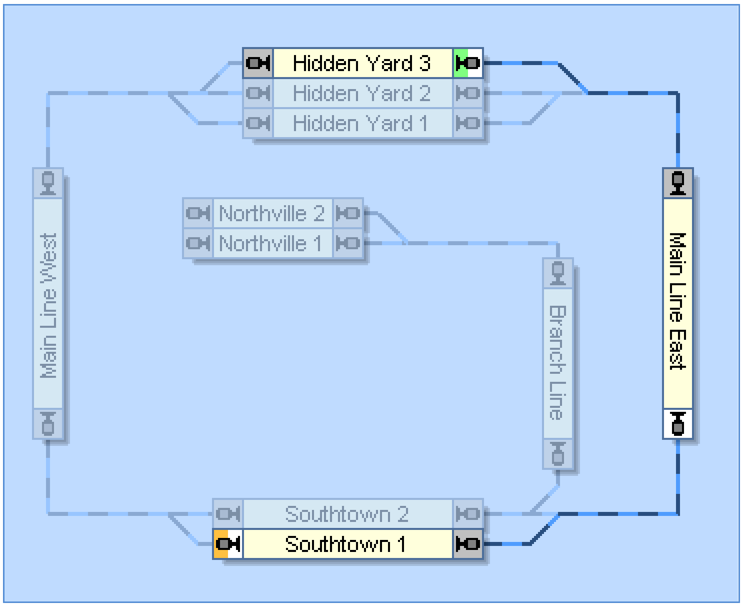


Diagram 120: Schedule Diagram

Diagram 120 shows the diagram of a schedule, that starts in “Hidden Yard 3”, passes “Mainline East” and ends in “Southtown 1”. The blocks and routes, that belong to this schedule, are drawn with normal intensity, while the objects, that do not belong to the schedule are drawn transparently in the background. In a specific mode of the software you can easily pick and add them to the current schedule with a click of the mouse on these objects.

Additionally one or more starting blocks and optionally one or more destination blocks are to be specified. Starting blocks are marked in the schedule diagram with a small green marking, destination blocks with an orange or red marking. In the diagram above "Hidden Yard 3" is marked as a starting block and "Southtown 1" is marked as a destination block.

In order to start this schedule, assign an arbitrary train to block "Hidden Yard 3", select the schedule on the computer screen and call the appropriate start command of **TrainController™**. The *Visual Dispatcher* will automatically allocate the blocks and activate the routes, that belong to this schedule and will automatically start the train. When the train reaches the stop indicator in "Southtown 1", the schedule is terminated.

A schedule can only contain elements, that are also contained in the main block diagram. The location of each element in the display is determined by the location of the referenced element in the main block diagram. If an element in the main block diagram is changed, moved or deleted then this change is reflected in all schedule diagrams. In this way multiple schedules can be conveniently maintained by changes to the main block diagram.

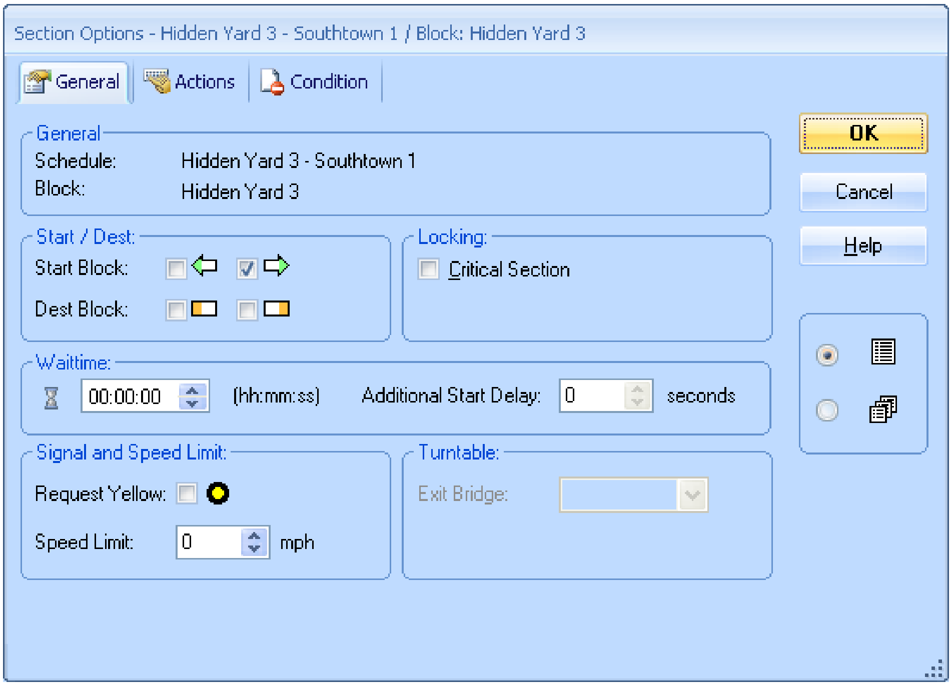
Start and Destination of a Schedule

Each schedule contains one or more start blocks and one or more destination blocks. Start blocks are marked in the schedule diagram with a small green marking, destination blocks with an orange marking.

! **It is required that you mark the desired start and destination blocks otherwise the schedule cannot be started.**

In Diagram 120 "Hidden Yard 3" is marked as start block to the right and "Southtown 1" is marked as destination block to the left.

Start, destination and other schedule specific section settings are entered in the dialog box displayed below.



Section Options - Hidden Yard 3 - Southtown 1 / Block: Hidden Yard 3

General | Actions | Condition

General
Schedule: Hidden Yard 3 - Southtown 1
Block: Hidden Yard 3

Start / Dest:
Start Block: ☐ ⇦ ☑ ⇨
Dest Block: ☐ ☐

Locking:
☐ Critical Section

Waittime:
00:00:00 (hh:mm:ss) Additional Start Delay: 0 seconds

Signal and Speed Limit:
Request Yellow: ☐ ●
Speed Limit: 0 mph

Turntable:
Exit Bridge:

Diagram 121: Schedule Specific Block Settings

Passage through each Block

The direction, in which blocks are passed by trains running from a start to a destination block of the schedule is marked with grey and highlighted color at the exits of each block. Trains on this schedule pass each block from the grey to the highlighted exit. A train running the schedule in Diagram 120 from “Hidden Yard 3” to “Southtown 1” will pass “Main Line East”, for example, from the grey exit at the top to the highlighted exit at the bottom. Blocks, that are passed in both directions in the same schedule, are displayed with both exits highlighted. Blocks, that are not included in the schedule or that cannot be reached by a train executing this schedule, are drawn with two grey exits. If a block, that is included into your schedule, is drawn with two grey exits, then no path exists from a starting block to a destination block, that touches this block.

Note that schedules can also be started to the opposite direction, i.e. from a destination block to a starting block. In such case each block is passed from the highlighted to the grey exit.

Alternative Paths

One of the most outstanding features of the *Visual Dispatcher* is the ease of specifying alternatives for the path a train has to take when executing a schedule.

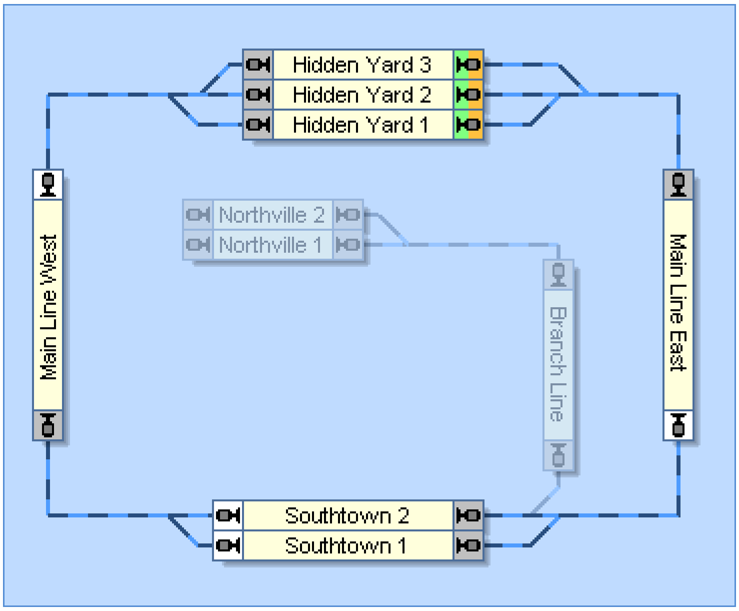


Diagram 122: Schedule Diagram with alternative Paths

Diagram 122 shows a schedule for train movements, that begin in of the three blocks in "Hidden Yard", proceed on the mainline in a clockwise direction, pass "Southtown" through one of the two blocks and end again in "HiddenYard".

To start the schedule, assign a train to one of the blocks in "Hidden Yard", select the schedule on the computer screen and call the appropriate start command of **TrainController**™. The *Dispatcher* will automatically allocate the blocks and activate the routes that belong to this schedule and will automatically start the train. If there is more than one train located in "Hidden Yard" and both can be used with this schedule, one of the trains will be selected automatically. It is also possible for you to pre-select the train before starting the schedule.

The *Dispatcher* will also look for an appropriate path through "Southtown" and will select a block in "Southtown" as well as appropriate routes to this block, that are available. If both blocks of "Southtown" are currently available, then the Dispatcher will perform a random selection. In the same way an appropriate block in “Hidden Yard” is selected, when the train approaches the destination.

Further, each schedule can be started in either direction. If the schedule is started in the opposite direction, then the specified destination blocks of the schedule are used as starting blocks and the starting blocks become destination blocks. The schedule of Diagram 122 can also be started in the counter-clockwise direction.

Since the start and destination blocks are identical in this example the trains will start and end in “Hidden Yard”. In Diagram 120, though, a train will start in “Hidden Yard 3” and end in “Southtown 1”, if the schedule is started in the normal direction. Starting the same schedule in the opposite direction will cause these two blocks to swap their meaning. “Southtown 1” will become the starting block and the train will end in “Hidden Yard 3”.

The terms *start* and *destination* are mainly used to describe, from where to where the trains travel on this schedule and where trains end. The actual starting block of a train can also be located in the inside of the schedule. In Diagram 122 the *Dispatcher* will first try to find an available train in “Hidden Yard”. If there is no appropriate train in “Hidden Yard” the Dispatcher can be instructed to start a waiting train in “Southtown”, if desired. If you select a waiting train in “Southtown” and start a schedule with that train, the Dispatcher will use this train, even though it is not located in the starting block of the schedule.

The destination blocks are always used as the end point of each schedule. In other words: a train can be started in any block of the schedule and it will always make its way to an appropriate destination block, that can be reached from where it is started.

Looking at Diagram 122 we realize, that with one single schedule diagram and by picking a few blocks and routes from the main block diagram, we can describe all possible train movements in both directions on the main line of this layout.

- The start and destination blocks of each schedule are to be specified manually.
- It would for example be possible to explicitly specify “Southtown 1” as an additional destination block in Diagram 122. If Southtown 1 is available, then each train coming from “Main Line East” will select “Southtown 1” as its destination. If “Southtown 1” is not available, the train will automatically proceed via “Southtown 2” to “Hidden Yard”.
- It is not possible to reverse a train within a schedule. If, for example, a train enters “Southtown 1” from “Main Line West” it is not possible to leave “Southtown 1 to “Main Line West” without first terminating the current schedule and starting another

schedule. This other schedule can, however, be another run of the same schedule diagram.
- It is not possible to change a train within a schedule.

! **Schedules describe train movements of one train from blocks to other blocks without changes of trains and without changes of direction.**

You can create as many schedules as you need.

Schedules are not bound to specific trains. In principle, each schedule can be executed by any train. In this way, by specifying only a few schedules it is possible to achieve varied operation for many different trains. To start a schedule with a specific train, the train must, however, be currently located in a block of this schedule.

To run your trains with realistic speed it is very important that the speed profile of each affected engine is set (see section 3.5, “The Speed Profile”).

5.12 Execution of Schedules

B For varied operation or special situations you can specify among others the following attributes for each schedule:

- If the schedule will be executed manually or automatically controlled by the computer.
- A time period for which the Dispatcher repeatedly tries to start the schedule, if the first attempt to start the schedule fails.
- Whether certain blocks or routes of the schedule will be passed with restricted speed.
- Operations, which are executed at the beginning, at the end or during the schedule.
- Whether and how often the schedule will be repeated as a cycle or by a shuttle train.
- A selection of other schedules, which are started after finishing the schedule with regard to availability or by random selection

Starting a Schedule

B Each schedule can be started during operation of the layout in either of the two possible directions, i.e. from the starting to the destination blocks or vice versa.

When a schedule is started, the *Dispatcher* searches the starting (destination) blocks of the schedule until it finds a *current block* of a train, which is not already running on another schedule.

If there is no such block then the *Dispatcher* can optionally continue the search in other blocks, that are located on the path from a starting to a destination block (or back) to find a train that can be started from there. The attributes of each schedule contain an option with which you can specify, whether the *Dispatcher* may start a train from other blocks than the explicitly marked starting (destination) blocks or not.

If no train is found on a block of the schedule or all trains are already running other schedules then the start of the schedule fails. It is possible to specify a time period for which the Dispatcher repeatedly tries to start the schedule, if the first attempt to start the schedule fails.

!

A schedule is always started with one train. If you want to start the same schedule with several trains, then the start of the schedule must be executed several times according to the number of trains to be started. This repeated start can be automated by Operations of Buttons and Macros (see section 14.4, “Operations”).

Reservation of Blocks and Routes

B

When a train is started on a schedule, the *Dispatcher* tries to reserve at least the *current block* and the next block in front of the train. Also, when a train enters a block, the block ahead is reserved.

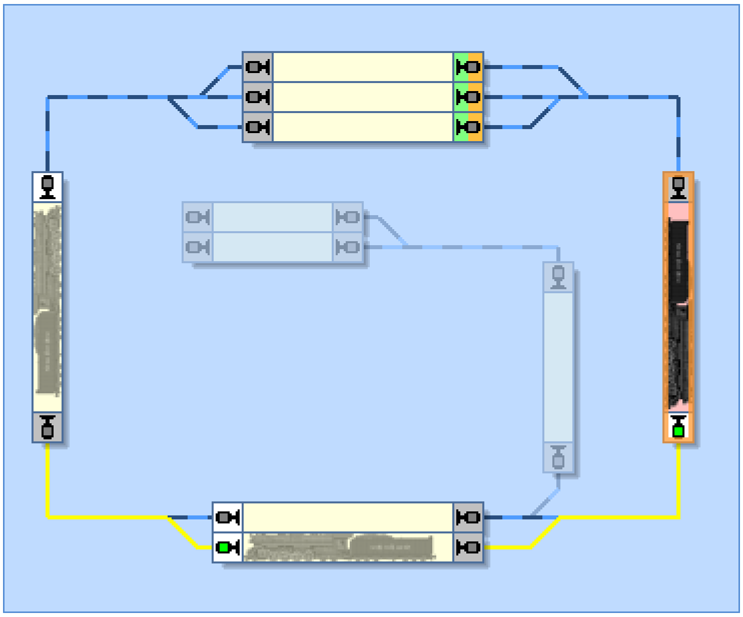


Diagram 123: Reservation of the Block ahead

In the situation displayed above the train has just entered block “Main Line East” (displayed in red). The block ahead is reserved for the train.

The route located between the current block and the block ahead is activated, too. A route is assumed to be located between two blocks, if it connects these blocks in the schedule diagram.

If it is not possible to reserve at least one block ahead of the train or if the route to this block cannot be activated, then the signal at the related block exit is set to red and the train must not proceed.

The *Dispatcher* follows different strategies to reserve the next blocks and routes.

By default, the *Dispatcher* applies a *smart* mode. This means: when a block directly ahead of the train is about to be reserved, then the *Dispatcher* checks, whether there is a route ahead of the block ahead. If this is the case, then this route and the block ahead of this route are reserved. This is done to reserve and activate the route in time to prevent unintentional train stops caused by long lasting route activation.

The diagram displayed above shows the smart mode. On entry into block “Main Line East” the Dispatcher does not only reserve block “Southtown 2” at the bottom. The Dispatcher also checks, whether there is a route directly ahead of “Southtown 2”. Since this is the case, this route and the block ahead of this route are reserved, too. This is done to avoid unintentional train stops in “Southtown 2” due to the fact, that the train must not leave “Southtown 2” before the route to “Main Line West” is activated.

Smart reservation avoids unintentional train stops caused by long lasting route activation.

What happens, if “Main Line West” is currently not available in this situation? This is no problem. The *Dispatcher* only <u>tries</u> to reserve the additional route and the block ahead of “Southtown 2”. If this is currently not possible, then the train is allowed to simply proceed to “Southtown 2”.

It is also possible to instruct the *Dispatcher* not to apply the smart mode to a schedule. In this case it is possible to specify a fixed number of blocks, that the Dispatcher will try to reserve during execution of the related schedule. If, for example, the number of blocks to be reserved ahead is set to 2, then the *Dispatcher* will always try to maintain the next 2 blocks in front of the train reserved for this train. If it is not possible to reserve the specified number, then the *Dispatcher* will allow the train to proceed, if at least one block in front of the train is available.

Using a fixed number of 2 as look ahead for the block reservation ensures that the distant signal assigned to the block ahead always shows a correct state. If it is desired to install a signaling system based on the internally calculated signal aspects of the *Dispatcher*, then this option might be useful, especially if distant signals are used.

By increasing the look ahead any further for certain schedules you can also give some trains a higher priority. When a train is able to reserve the complete path to the destination when its schedule starts, then it cannot be blocked by other trains during its journey. It has received a high priority to reach its destination.

Path Selection

The *Dispatcher* follows a smart strategy, when it has to select one of several possible paths. In Diagram 122, for example, the *Dispatcher* has to select one of three possible paths, when a train approaches the “Hidden Yard” from the west or from the east.

In the following the criteria which influence the selection of a path are listed. The following aspects lower the chance of a path being used or prevent a path from being selected at all:

- Other trains, that reserve one or more blocks and routes ahead of the train.
- Locks applied to the entry or exit of certain blocks (see page 156).
- Blocks or routes, that are reported as occupied by unknown objects; more severe, if the rules specified for the schedule do not allow the entry of occupied blocks or routes.
- Conditions, that prevent a block from being reserved or a route from being activated (see the following section).
- The distance to an appropriate destination block.
- Superfluous loops.

There are also criteria, that raise the chance of a certain path being selected:

- Blocks ahead of the train that have already been reserved for this train.
- Activated routes ahead of the train , that are not reserved by other trains.
- The distance to the nearest obstacle listed in the previous list.

At first the *Dispatcher* evaluates each possible path according to the criteria listed above. Two paths are equivalent with regard to these criteria, if exactly the same aspects apply. If two paths are equivalent, then the Dispatcher performs a random selection.

The criteria listed above do not prevent a path from being selected. They lower the chance of a path to be selected, though, but the Dispatcher might select a path, which is affected by a negative criterion, if there is no "better" alternative.

Special attention should be paid to the distance to an appropriate destination block. If the distances to appropriate destination blocks of two alternative paths are different, then the Dispatcher will probably select the shorter path. If the shorter path is currently locked by an obstacle, then it depends on the difference of these distances, whether the Dispatcher uses the longer path or decides to try to pass through the shorter path in the hope, that the obstacle soon disappears. In other words: the Dispatcher does not select a free path under all circumstances, especially not, if the free path is much longer than other alternatives, that are currently not available.

Release of Blocks and Routes

In general a block or route reserved by a schedule is released when the train has reached a block ahead of this block/route and when this block/route is not indicated as occupied anymore. In Diagram 122, for example, block “Main Line East” is not released before a train coming from “Hidden Yard” has reached “Southtown”. If “Main Line East” is still indicated as occupied when the train reaches “Southtown” release of “Main Line East” is further delayed until the occupancy indication of “Main Line East” is turned off.

In detail the following rules apply:

- By default a block is assumed to be reached, when the train arrives at a stop marker assigned to this block. Certain settings, however, allow to prepone this point (see page 353, “Release of Blocks and Routes”).
- An occupied block or route is not released. (An exception of this rule is outlined below.)
- A block or route is not released until the train has reached a block behind of this block/route.
- When a train reaches a block all non-occupied blocks/routes located before this block, but not located behind another occupied and reserved block/route, are released. If, for example, “Main Line East” in Diagram 122 is still reserved and occupied when the train reaches “Main Line West”, then the used block of “Southtown” is not released, regardless whether it is occupied or not. If both, “Main Line East” and the related block in “Southtown”, are unoccupied when the train reaches “Main Line West”, then both blocks are released.
- When the train reaches the destination position of the current schedule, i.e. the stop indicator in a destination block of this schedule, then all blocks and routes apart from this last block are released, regardless whether they are currently occupied or not.

Preset Block Signals and Speed Limits

As outlined in section 5.9, “Block Signals” **TrainController™** automatically calculates signal aspects for all trains running under control of the *Dispatcher*. These signal aspects take into account the availability of blocks and routes ahead of the train. If the train must not enter a block, then the signal of the previous block is set to “red”. If the train can enter the block, then the signal is usually set to “green”. It is additionally possible, however, to cause **TrainController™** to display “yellow” instead of “green”, if desired.

For this purpose it is possible to select an individual signal aspect (yellow or green) for each block or each route in a schedule. Dependent on this setting **TrainController™** will automatically apply the selected color to the calculated block signal, if the train may proceed.

These signal settings are specified at the level of blocks and routes in a schedule. That means: the same block or route may have differing signal settings in different schedules.

It is additionally possible to adapt the train speed to the selected signal aspect. This is done by specifying speed limits for the green and yellow signal aspect in the properties of each block. For each block it is preset in this way, at which maximum speed each train may pass this block dependent on the currently selected signal.

Assume a block with the maximum speed (green signal) set to 80 mph and the restricted speed (yellow signal) set to 30 mph. If the signal for this block in schedule “A” is set to green, then the train will pass this block at 80 mph, when schedule “A” is executed. If the signal for this block in schedule “B” is set to yellow, then the train will pass this block at 30 mph, when schedule “B” is executed.

The available speed limits for the green and yellow signal aspect are specified globally on the level of each block. In a schedule it is then selected, which of the two speed limits applies for this block in this schedule.

The above describes the default policy. **TrainController™** **Gold** provides even more possibilities to adapt the calculated block signals and applied speed limits to personal needs.

TrainController™ **Gold** it is not only possible to select the desired signal indication (green or yellow) for each block or route in a schedule, it is also possible to specify once for all schedules, that the calculated block signal of certain blocks or routes will be yellow. And even more: it is possible to preselect the yellow signal indication individually for each particular position of a turnout. The signal indication specified for a specific turnout position is then accordingly propagated to all routes, that contain this turnout in this position.

The hierarchy of the various signal settings is as follows: to calculate the block signal indication for a certain route or block in a schedule, **TrainController™** **Gold** at first checks, whether the yellow signal has been selected for this block or route in the settings of the schedule. If this is not the case, then the properties of the block or route are checked instead. In the case of a route the preselected block signals for the according positions of all turnouts contained in this route are checked, too. If at least one of the checked objects requests a yellow block signal, then the resulting block signal is yellow, too. In all other cases the resulting block signal is green.

The possibility to preselect signal indications on the level of blocks, routes or turnouts provides several advantages:

- In cases, where a specific block or route will always be passed with the same signal indication, it is much more convenient to preselect the indication once for all schedules in the properties of the block or route.
- The possibility to preselect a signal on the level of turnouts is useful, if specific turnouts will always be passed with the same signal indication. It is much more convenient to preselect this indication once for all routes and schedules, which use this turnout, in the properties of the turnout, rather than being forced, to repeat the same selection in all affected routes or schedules.
- The possibility to specify differing signal indications for the particular positions of a turnout is useful, if the indication of the calculated signal will depend on the turnout position.
- The signal indication preselected for a block, route or turnout is also applied to trains run by **AutoTrain™**. In other versions of **TrainController™** the signal indication for trains run by **AutoTrain™** is always green in all blocks or routes and cannot be changed.

As outlined earlier there is a close connection between calculated block signals and speed limits. The allowed speed of a train depends on the currently valid signal indication (green or yellow). While other versions of **TrainController™** only allow to specify speed limits on the level of blocks, **TrainController™ Gold** allows to specify speed limits also on the level of schedules, routes and even turnout positions. The hierarchy of the various settings is similar to that of the signal settings described above. To calculate the speed limit for a certain route or block in a schedule, **TrainController™ Gold** at first determines the speed limit specified for this block or route in the settings of the schedule. Additionally the speed limit preset in the properties of the block or route is determined, too. In the case of a route the speed limits for the according positions of all turnouts contained in this route are determined, too. The final speed limit results from the minimum of all determined limits. If no speed limit is specified for a certain object in this chain, then the settings of this object do not affect the resulting speed limit.

The possibility to preset speed limits on the level of routes or turnouts provides the same advantages as listed above. Especially it is possible with **TrainController™ Gold** to propagate speed limits valid for a certain position of a turnout to all affected schedules and also to trains run by **AutoTrain™**.

Temporary Speed Limits

A temporary speed limit can be accomplished by executing a specific train operation, e.g. by a marker in a block. Temporary speed limits can be applied for trains running under control of a schedule, **AutoTrain** or a spontaneous run.

If the current speed of the train, to which the speed limit is applied, exceeds the specified value, then the speed of the train is reduced to the specified speed as soon as this operation is executed. If 0 is specified as speed, then an effective speed limit, if any, is cleared. When a train terminates a schedule, then an effective speed limit, if any, is automatically cleared, too. This is also true, if control of the train is passed to a successor schedule. Temporary speed limits are only effective in the scope of the current schedule of the affected train.

Waiting Time

You can specify a *waiting time* for each block contained in a schedule in order to perform scheduled stops in certain blocks of a schedule.

In **TrainController™ Gold** it is furthermore possible to specify an individual delay for each scheduled stop. Such delay is applied after a scheduled stop has ended, while associated operations are executed and before the train is set in motion. This time span can be utilized to perform additional operations (e.g. playing an announcement, the noise of closing doors or the whistle of the conductor) after a scheduled stop ended and before the train is set in motion (see below).

In **TrainController™ Gold** it is additionally possible to specify the trains, which are affected by the waiting time. In this way it can be specified, for example, that passenger trains are stopped during execution of the schedule while freight trains go by. The definition is done with a train description (see section 11.3, “Approved Trains”).

In **TrainController™ Gold** a waiting time for each block can be specified for AutoTrain runs and spontaneous runs, too.

Additional Operations

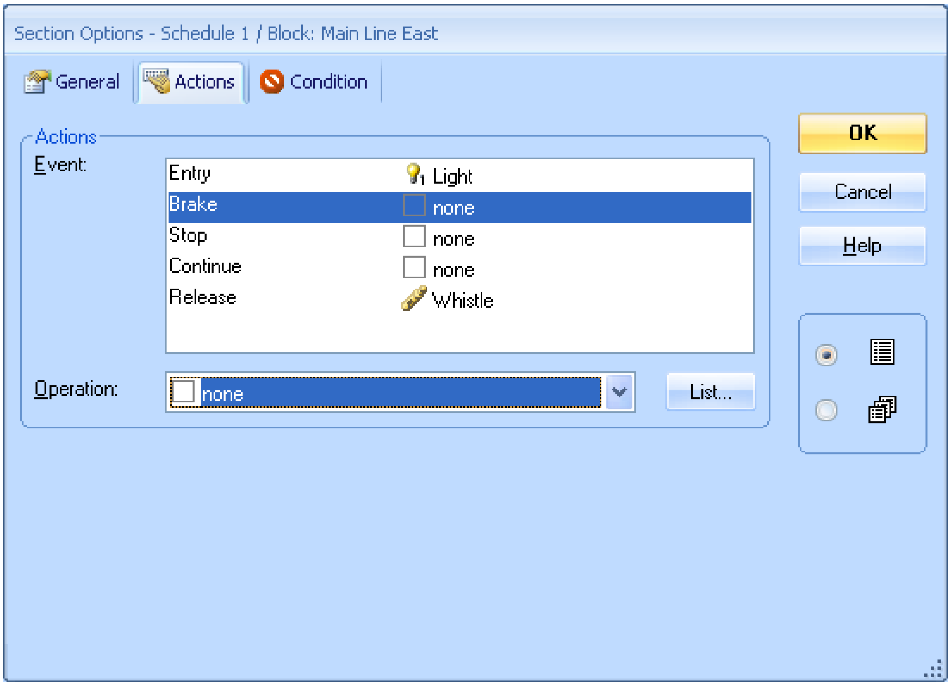
Finally it is possible, to assign Operations to each block of a schedule. Possible operations are turning on or off an *engine function* (see section 3.6, “Headlights, Steam and Whistle”), execution of certain train operations or execution of a list of operations in order to perform a sequence of actions.

These operations can optionally be performed when

- the train enters the block
- the train reaching a brake indicator has to reduce its speed
- the train has to stop
- the train starts again after a stop

- the block is released after the train has left the section

Additionally it is possible to perform operations before starting or after finishing the schedule.



Actions

Event:

Entry	Light
Brake	none
Stop	none
Continue	none
Release	Whistle

Operation: none

Diagram 124: Specifications of the Section of a Schedule

In the example displayed above each train entering the related block will turn on the light. Additionally it will blow its whistle when the block is released later.

If a function symbol specified here is not configured for an engine, then this engine will do nothing, when it executes this schedule. If, for example, the function symbol *Whistle* is only assigned to steam engines in the example displayed above, then diesel engines will remain quiet when executing this schedule.

These operations are specified on a per-schedule base. It is possible to specify different operations for different schedules.

In **TrainController™** **Gold** such operations can also be specified for AutoTrain runs and spontaneous runs.

Type of a Schedule - Shuttle and Cycle Trains

There are different types of schedules.

Normally – when no special type is selected – the journey of the train ends in a destination block of the schedule.

If a train will repeat the schedule as a *shuttle train*, it will be started again after arriving in a destination block and will run back in the opposite direction to an appropriate start block. It is possible to specify a repeat count to control, how often the schedule will be repeated.

It is also possible to repeat the schedule as a *cycle* based on a circular diagram. In this case the train is started again on the same schedule after arriving at the destination block of the schedule. The train repeats traveling on the schedule in the same direction as before. As for shuttle trains it is possible, to specify how many times the cycle will be repeated.

! **When repeating schedules as a cycle it is necessary that these schedules are circular, i.e. destination blocks must also be start blocks.**

Shunting

An additional type of schedules is *shunt*.

If a schedule is provided for shunting then all *blocks* and *routes* of the schedule are reserved, when the *Dispatcher* starts the schedule. The blocks can be passed in an arbitrary direction. Shunting trains are operated manually and it is also allowed to reverse a train while shunting and to leave a block in the opposite direction. The *Dispatcher* does not intervene; it takes care only, that other engines or trains under its control do not enter the blocks reserved for the shunting train.

If a schedule is provided for shunting, then all blocks contained in schedule are reserved, when a train is started. Since each block can be reserved for only one train, at most one train can run simultaneously on this schedule.

Running Trains manually under Control of a Schedule

For each schedule you can specify its *driving mode*. If desired you can control engines and trains on the schedule completely manually. In this case the computer reserves the blocks, activates the routes and calculates the block signals. You are – like a real engineer – responsible for obeying the indicated signals and following the speed conditions. But it is also possible to transfer the control over the schedule completely to the computer. In this case all engines and trains on this schedule are operated automatically. Finally it is also possible to share the engineer's job with the computer. In this way it is for example possible, that the train is running under your manual control, but that the computer is able to intervene to stop a train in front of a red signal.

Driving Mode	Explanation
🖥	Trains are completely controlled by the computer
🖥🟡	The computer intervenes when restricted speed is prescribed or when the train approaches a red signal requesting the train to stop.
🖥🔴	The computer intervenes, when the train approaches a signal requesting the train to stop.
🎛!	Trains are almost completely controlled manually. If the human operator fails to stop the train in time before reaching the stop marker in front of a red signal, then the computer performs an emergency stop of the train.
🎛	Trains are completely controlled manually.

Table 4: Driving Modes of a Schedule

It is possible to use different modes for different schedules, regardless whether these schedules share the same blocks and routes or not. This enables full automatic operation of one part of your layout and running trains manually under computer control in another part.

Different schedules with different modes can be arranged for the same part of your layout, too. It is for example possible to create two schedules for the main track of your layout. The first schedule is used for automatically running trains, while the second schedule uses the same track for trains operated manually under control of the computer. In this way you can operate your favorite train manually while other trains in front of or behind this train are controlled automatically.

Driving modes can also be specified individually for each particular engine. If this is done, then the driving mode of the engine overrides the setting of the schedule. This is useful if you want to run different trains in different driving modes with the same schedules.

5.13 AutoTrain – Start of Schedules made Easy

B

AutoTrain™ is another outstanding feature of **TrainController™**. With **AutoTrain™** you can run automatic trains at any time during operation without the need to define schedules in advance.

AutoTrain™ is especially useful in the following cases:

- If a train will automatically run somewhere during operation and you did not specify an appropriate schedule to perform this task in advance.
- If you want to define a new schedule quickly from scratch.

Auto Train by Drag & Drop

The fastest way to run **AutoTrain™** is Drag & Drop with the mouse:

- Select the **Operation** tab (or the **Schedule** menu in the classic user interface) and call the **AutoTrain by Drag and Drop** command.
- Move the mouse cursor to the start block.
- Press and hold the left mouse button on one of the following symbols:
  , if you want the train to exit the start block to the left.
  , if you want the train to exit the start block to the right.
- Hold the left mouse button pressed and drag the mouse to the block in the block diagram or in the switchboard, where the train will stop.
- Release the left mouse button over one of the following symbols:
  , if you want the train to enter the destination block from the right to the left.
  , if you want the train to enter the destination block from the left to the right.
- The train will now start and run automatically to the destination block.

Some additional options are available:

Click this option, if you want to leave the **AutoTrain by Drag and Drop** command turned on after releasing the mouse button. In this case another AutoTrain run can be initiated by drag and drop without the need to call the **AutoTrain by Drag and Drop** command once more.

By pressing and holding the mouse button on this symbol it is possible to specify all blocks in a station (see section 15.7, "Stations") as start blocks with exit of the train to the left. This station is determined by the block, where the mouse is pointing to. It is the station, where the block is located in.
This option works in the same way as if all blocks in the said station are specified as start blocks of a schedule.

Same as above but with exit of the train to the right.

By releasing the mouse button over this symbol it is possible to specify all blocks in a station as destination blocks with entry of the train from the right to the left. This station is determined by the block, where the mouse has been dragged to. It is the station, where the block is located in.
This option works in the same way as if all blocks in the said station are specified as destination blocks of a schedule.

Same as above but with entry of the train from the left to the right.

Click this option, if you want to perform a local AutoTrain run (see page 369, "Local Schedules").

The above options allow to perform an AutoTrain run from a station to a station with only one single drag and drop operation. It is also possible to mix the options for blocks and stations. In this way an AutoTrain run can be performed from an individual block to a station or vice versa.

The above symbols are valid for horizontal blocks. Symbols for vertical blocks look accordingly.

Auto Train Toolbar

With the **AutoTrain™** toolbar you have more options for individual customization before the train is actually started. To run a train with the **AutoTrain™** toolbar the following steps are performed:

- Open the **AutoTrain™** toolbar .
- Select the locations (blocks) on the layout, where the train will start.
- Select the locations (blocks) on the layout, where the train will stop.
- Optionally specify additional options that influence the execution of the **AutoTrain™**, such as waiting time, operations, cycle, shuttle, etc.
- Start **AutoTrain™**.

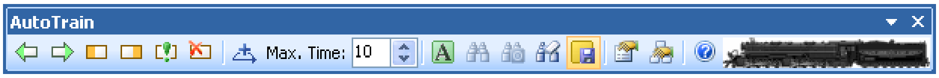


Diagram 125: AutoTrain Tool Bar

After starting **AutoTrain™** automatically tries to find a path from the specified start block to the specified destination blocks. If a train is located in the start block, it is automatically started to run in the selected direction.

A started **AutoTrain™** is very similar to a schedule which is currently executed. It has one starting block and one ore more destination blocks, that are selected before **AutoTrain™** is started.

There are some additional options:

- After selection of the start and destination blocks you can let **AutoTrain™** try to find a path from the start to the destination blocks without starting a train. This is useful in *edit mode*, especially if no train is located in the start block. This is also useful if you want to check the resulting path before actually starting the train. Together with another option, that allows you to store the current **AutoTrain™** as a permanent schedule for later use, this is a very fast method to create new schedules by letting the software calculate the appropriate paths for you.
- It is possible to select certain blocks or routes to be included in the schedule prior to starting the search for an appropriate path. Each path found will then pass through these blocks or routes, if possible. This gives you more control over the resulting path.
- It is also possible to exclude certain blocks or routes from **AutoTrain™** prior to starting the search for an appropriate path. This also gives you additional control over the resulting path.
- You can also specify, whether only the shortest possible paths from the start to the destination blocks will be taken into account or all possible paths.
- Additionally it is possible to limit the search time. This option is useful in the case of large or complex layouts and slow computers, where the search may take a while.

While an **AutoTrain™** is active you can also store it as a schedule to execute it later, e.g. as part of a time table.

! **AutoTrain™ requires the prior calculation or creation of a block diagram.**

! With regard to selection of blocks and routes **AutoTrain™** can follow the same policies, that are valid for schedules. That means: as well as allowing you to include blocks or routes, which are currently locked, into a schedule in edit mode, **AutoTrain™** can also include blocks or routes, which are currently unavailable. In this way it is possible to create schedules with **AutoTrain™** for later use, which contain blocks or routes, that are currently not available. With certain settings it is possible, however, to cause **AutoTrain™** to take into account only those routes or blocks, that are currently available for a train run.

AutoTrain with Start and Destination Keys

In **TrainController™** **Gold** it is also possible to perform **AutoTrain™** as operation of other objects. This is in particular useful, if you want to start **AutoTrain™** with push button symbols by using them as start and destination keys.

AutoTrain™ operations are always associated with certain blocks.

AutoTrain™ operations do not distinguish between the start and the destination of the run. If two **AutoTrain™** operations are called for Block A first and then for Block B, then the train runs from A to B. If the same operations are called in the opposite order, then the train runs from B to A. **AutoTrain™** operations should be always called in pairs. The first operation determines the start block and the direction, in which the train starts. The second operation specifies the destination block and the direction, in which the train enters the destination block. The second operation also starts the train.

You can use **AutoTrain™** operations in a macro, for example, to run a train from one block of your layout to another. In this case you should ensure, however, that two **AutoTrain™** operations, one to specify the start block and one to specify the destination block, are contained in the macro.

An interesting use of these operations and the actual reason, why these operations are provided, is the use with push button symbols as start and destination keys. To accomplish this, assign one **AutoTrain™** operation to the operations of each related push button. **AutoTrain™** operations should be executed in pairs. This is accomplished during operation by pressing one push button with such operation assigned and then another. The first push button determines the start block and the direction, in which the train starts. The second push button determines the destination block and the direction, in which the train enters the destination block. The second push button also starts the train. It is the order, in which the operations are executed, which is relevant for the determination of start and destination, not the operations themselves.

AutoTrain™ with start and destination keys is useful to run trains in a default manner, i.e. without any specific additional actions, on point to point connections. No prior definition of schedules is necessary, which keeps the schedule list less populated. By assigning **AutoTrain™** operations to the operations of feedback indicator symbols, which are again associated with push buttons on external control panels, it is even possible to trigger **AutoTrain™** from such remote panels.

In addition to the possibilities described above it is also possible in **TrainController™ 9 Gold** to perform AutoTrain with start and destination keys also from an arbitrary block in a station or to an arbitrary block in a station. These operations are also applied to blocks like the operations described above and work in a similar way. But instead of selecting only the block, where this operation is applied to, all blocks in the same station are selected, where this block is located in. Therefore these additional operations do not only select one single block as start or destination block of the AutoTrain run, but all blocks in the same station. It is also possible to mix the operations effective for individual blocks with the operations effective for all blocks in a station. In this way it is possible to perform an AutoTrain run from an individual block to a station or vice versa.

5.14 Schedule Sequences

With schedule sequences a series of schedules can be executed in **TrainController™ Gold** one after the other with the same train. Schedule sequences contain a list of other schedules. When the sequence is started, then the first schedule in the list is started, too. After termination of the first schedule the second schedule in the list is started with the same train and without stopping the train, if possible. After termination of the second schedule in the sequence the third schedule is started and so on, until the last schedule in the list is completed.

With schedule sequences it is possible to create long schedules by combining several shorter schedules. Sequences are for example useful to create a plurality of long schedules with a library of several short schedules as building blocks.

5.15 Successors of a Schedule

X For each schedule it is possible to specify a set of successor schedules, one of which will be started after the schedule is finished.

Several options allow you to specify how control of the train is passed from a schedule to its successor:

- The successor can be selected **by order** or **randomly**.
- Additionally you can select to **keep the train**, i.e. to enforce that the successor continues with the same train as before, or to enforce a **train change**.
- **TrainController™ Gold** also allows to specify a train description. If a train description is specified, then the successor is started with a train, to which the train description applies (see also page 257).
- It is additionally possible to specify, that the successor schedule will be started with the **oldest train**. The oldest train is the train, which has not been operated by a schedule for the longest time. **TrainController™** allows you to combine this option with the other options. If this option is combined with the option to perform a train change, then the successor is started with the *oldest* train, that differs from the previous train. If this option is combined with the specification of a train description, then the *oldest* train, to which the train description applies, is started.
- It is also possible to specify that **all** listed successors are started. These successors are started simultaneously, when the previous schedule is about to be terminated.

With schedule successors it is possible to control a hidden yard automatically. A train arriving in a hidden yard can be enabled to select another waiting train, which will leave the hidden yard.

! **If it is intended to start the successor with the same train, then it is recommended, that the successor starts with a destination block of the previous schedule. In this block the control of the train is transferred to the successor.**

! **If several schedules will be executed in a sequence, e.g. schedule 2 will be executed after schedule 1 and schedule 3 will be executed after schedule 2, then schedule 2 is to be specified as successor of schedule 1 and schedule 3 as successor of schedule 2.**

Since it is not possible to reverse a train or to change trains during the execution of a schedule successors must be used if

- a train will be reversed
- trains will be changed

Schedule Sequences vs. Schedule Successors vs. Long Schedules

Will complex train runs be specified as schedule sequences, as a chain of schedule successors or as one complex schedule? The answer to this question depends on the individual case and is also a question of personal taste.

A train run, for example, that starts in the hidden yard of the block diagram displayed in Diagram 101, passes both blocks in “Southtown” and ends again in the hidden yard can be specified as a schedule sequence with three or four schedules, as a chain of schedule successors or as one big schedule which alternate paths in “Southtown”.

These are the pros and cons of the particular approaches:

Schedule Sequences:

- Only available in **TrainController™ Gold**.
- No change of trains is possible between two subsequent schedules in the sequence.
- Usually used as a replacement for a single more complex schedule. Schedule sequences can be used to create a plurality of longer, more complex schedules by using several shorter, less complex schedules as building blocks.
- No static linkage between a schedule and the subsequent schedule in the sequence. A schedule can precede different subsequent schedules in different schedule sequences.
- The look ahead to select an optimal path out of several possible alternate paths is limited by the end of the current schedule in the sequence. This can improve the performance of the path selection, but may lead to selection of non-optimal paths.
- Schedule sequences can be started in reverse direction, i.e. beginning with a destination block of the last schedule in the sequence end ending in a start block of the first schedule in the sequence.
- During the change from one schedule of the sequence to the next certain minor limitations may apply with regard to calculation of block signal aspects and speed limits due to technical reasons.

Schedule Successors:

- Change of train is possible between two subsequent schedules in the chain of successor schedules.
- Static linkage between a schedule and the subsequent schedules in the chain of successors.
- Like schedule sequences the look ahead to select an optimal path out of several possible paths is limited by the end of the current schedule with the same consequences as with schedule sequences.

- Chains of schedule successors cannot be started in reverse direction.
- During the change from one schedule to a successor schedule without train change certain minor limitations may apply with regard to calculation of block signal aspects and speed limits due to technical reasons.

Single long Schedules:

- Change of train is not possible, until the schedule is terminated.
- The look ahead to select an optimal path out of several possible alternate paths can take into account the complete path to the block, where the train will finally stop. This supports the selection of optimal paths at the expense of program performance.
- Single schedules can be started in reverse direction, i.e. from a destination block to a start block.
- Single schedules can be repeated as cycle or as shuttle train.

Conclusions:

- If you want to change the running train between two subsequent schedules, then both schedules must be chained as successors. This is for example useful, if a train entering a hidden yard will trigger another train to leave this yard.
- If the train will not be changed, then it is usually better to create a schedule sequence (**TrainController™** **Gold** only) or a complex, long schedule rather than a chain of schedule successors.
- If a schedule will be repeated as a cycle or commuter train, then use a single schedule for this purpose.
- Critical sections cannot span different schedules. They must be completely contained in the same schedule.
- If a single schedule becomes very complex or long, then consider to split it into a schedule sequence of several more simple schedules.
- If the track plan of your layout allows to derive a plurality of more complex schedules from a relatively small set of simple basic schedules, then consider to create the complex schedules as sequences of these simple schedules.

It is possible, though not very recommended, to mix schedule successors and schedule sequences. This is treated by the software in the following way: chaining by successors has higher priority than chaining by sequences. That means: if schedule B is specified as successor of schedule A, and schedule A and a third schedule C are listed consecutively in a schedule sequence, then A is executed first by this sequence, then B (as successor of A) and finally C (as the second member of the sequence).

5.16 Schedule Selections

Sometimes it is desirable to select one of several schedules. This is supported by *schedules selections*. A *schedule selection* enables the selection of certain schedules out of a selection of several other schedules. Even though there is no schedule diagram associated with a schedule selection such selection can be started like any other normal schedule. It can be used wherever a normal schedule can be used. When a schedule selection is started then one or more of the schedules contained in the selection are selected and started. This selection may also include other schedule selections.

5.17 Operation Interruption - Termination of Schedules

There are several methods to interrupt the running operation or to terminate schedules. These methods can be accessed by different menu commands. They are described in the following:

- Global **Stop**: This command performs an emergency stop of all connected digital systems and terminates all running schedules. This is the most drastic method to terminate operation and should only be used in very rare, extreme emergency cases. Since all schedules are terminated the computer releases control of all previously running trains. If the emergency stop of the connected digital system is released later, then the software does not have control over any trains.
- **Freeze**: This command performs an emergency stop of all connected digital systems and interrupts all running schedules. This is the recommended method to stop operation in emergency cases. The software keeps control over all previously running trains. After resolving the emergency situation and clearing the freeze state of the software the operation can be continued at the position, where it was interrupted. All previously running trains are automatically restarted.
- **Stop Train**: This command stops the selected train abruptly, but does not terminate any running schedule. It can be used to clear an emergency, where only one single train is affected.
- **Stop All Trains**: This command stops all trains abruptly, but does not terminate any running schedule. All affected trains must be manually set in motion again later.
- **Terminate Schedule / Run**: This command stops the selected train abruptly and terminates its current schedule or spontaneous run, respectively. It can be used to terminate a running schedule or spontaneous run prematurely.
- **Terminate all Schedules**: This command stops all trains abruptly and terminates their current schedules or spontaneous runs, respectively.
- **Lock all Blocks:** The methods listed above stop the affected train always abruptly. To stop trains smoothly additional measures must be taken or custom methods must

be configured by the end user in versions other than **TrainController™ Gold**. In **TrainController™ Gold**, however, this method can be used to interrupt the operation of the layout by stopping all trains smoothly at the next appropriate location. If this method is applied, then all currently not reserved blocks are prevented from being reserved. Each train, which is currently controlled by a schedule or spontaneous run, will process the blocks, which it has already reserved and will then stop smoothly without reserving any additional blocks.

- **Lock all Schedules:** This method can be used to terminate the operation of the layout while stopping all trains smoothly at the destination of their current schedule. If this method is applied, then all schedules are prevented from being started. Each train, which is currently controlled by a schedule, will process to the next destination block of its current schedule and will then stop smoothly without starting any additional schedules.

Especially the last two methods, which are available in **TrainController™ Gold** only, are well suited to interrupt or terminate operation without causing trains to perform an abrupt stop. If **Freeze** is applied additionally after all trains have smoothly come to rest, it is possible to terminate the operating session completely and to start the next session again at another day and at the same position.

5.18 Putting it all together – The Dispatcher Window

The *dispatcher window* serves as display for the block system of your layout. It lists and displays all diagrams, blocks, routes and schedules.

In **TrainController™ Gold** the dispatcher window furthermore displays all stations (see section 15.7, “Stations”) and boosters (see section 15.8, “Booster”).

TrainController™ Gold allows to open as many dispatcher windows as you like. It is for example possible to open a separate dispatcher window for each existing block diagram. By grouping a dispatcher window together with a switchboard window within the same parent frame window (either docked side by side or tabbed) it is possible to create several grouped windows, that contain a switchboard each together with the associated block diagram. Especially if these windows are tabbed it is possible to toggle conveniently between the switchboard view and the block diagram view of the related part of your layout.

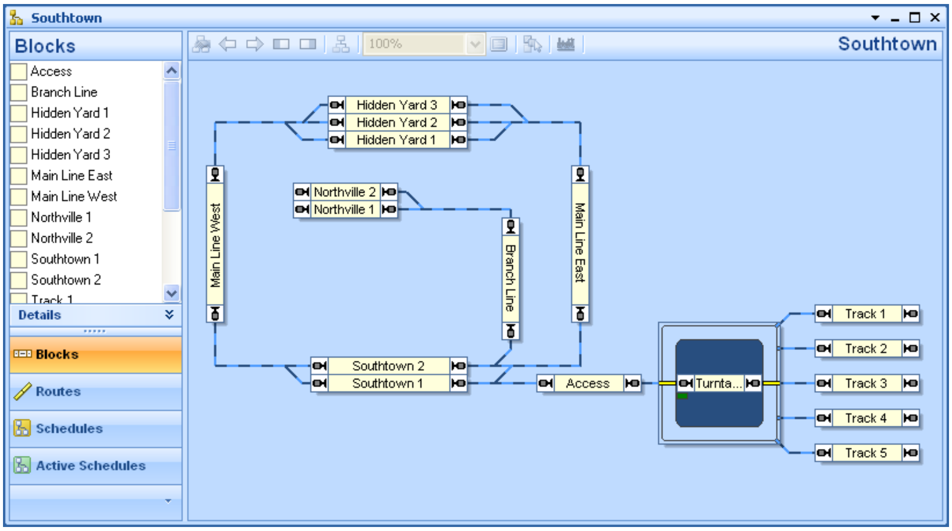


Diagram 126: Dispatcher Window

The dispatcher window is split into two parts. The left part lists the blocks, routes or schedules of your layout.

The left part of the dispatcher window of **TrainController**™ **Gold** furthermore lists all stations and boosters.

With the particular controls of the page navigator it is quite easy to switch from one view to another. Depending on the selected view additional detail information is also available. The block and route view, for example, additionally provide an optional view of the indicators and markers contained in the currently selected block or route. The schedule view provides optional lists of the blocks and routes contained in a selected schedule and also allows you to display a view of all indicators and markers, that are contained in a certain block or route of this schedule.

The optional indicator view furthermore provides another interesting feature: in offline mode, i.e. if the indicator symbols in the view are currently not connected to an actual digital system, it is possible to turn the status of these symbols on and off by clicking on them with the left mouse button. In this way the sensor events generated by passing trains can be conveniently simulated.

The right part of the dispatcher window displays the currently selected block or schedule diagram.

The right part of the dispatcher window of **TrainController**™ **Gold** may also display the currently selected station or booster diagram.

It is possible to switch from one diagram to another by using the diagram selector menu in the upper right corner of the dispatcher window. Click on the name of the current diagram, which is displayed by well visible letters in the upper right corner of the dispatcher window to open the menu of available diagrams and to change to another diagram.

All routes displayed in the dispatcher window, whether displayed in the route list or in the block diagram, can be operated with mouse click, too, when edit mode is turned off.

5.19 Customizing the Dispatcher Window

General

The dispatcher window can be freely resized and zoomed. This allows you to let the display fit the dimensions of the displayed block diagram optimally.

The colors of the window background, blocks and connecting routes can be adjusted to personal taste, too.

The display of block signals and train images can be turned on or off.

Additionally to these general customization features, which were also available in previous versions of the software, **TrainController**™ provides the following additional customization features:

- A new option allows the reset of all display options to factory defaults.
- Active routes can be displayed with individually specified colors (as in previous versions), or with the color of the reserving train, if any, or with a color, that is common for all active routes.
- The highlight color of occupied routes can be controlled by the reserving train, if any, as in previous versions, or by the color of the occupied indicator or by specifying a constant color value.
- The display intensity of blocks and routes, that do not belong to the currently selected schedule can be dimmed to fit personal taste and to support low contrast display environments.

- It is optionally possible to display the names of blocks in the block diagram, when edit mode is turned off. Other versions of the software displayed block names solely in edit mode.

Visibility of Schedules

The visibility of schedules in the dispatcher window can be limited during operation to those schedules, that you actually want to see listed then. This is controlled by the option **Visibility in Edit mode only** in the properties of each schedule. If this option is turned on, then the schedule does not occur in the schedule list in the dispatcher window, when edit mode is turned off. This is for example useful, if you want to exclude those schedules from being listed, that are started as successor of other schedules or by start/destination keys or if you want to limit the list to those schedules, that you want to start explicitly after selecting them from the list.

6 The Traffic Control

B

During operation of a layout the *Traffic Control* shows the status of the currently selected train, block or route and the current status of the indicators, that have been assigned to the current object.

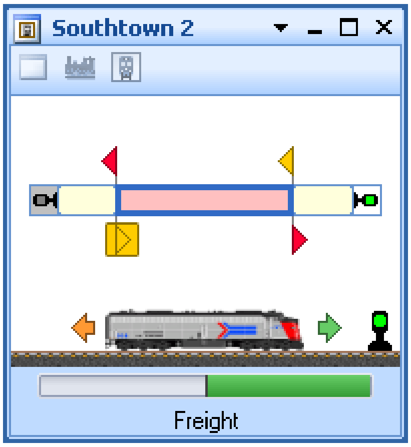


Diagram 127: Traffic Control

Here all important information about the currently selected train and its current location is assembled. When you select a train on the computer screen, this train and the block, where it is located, are displayed. When you select a block or a route, this block/route and the train, which is currently there, if any, are displayed.

The speed of the train is made visible with a colored rectangle. The status of the block, whether occupied or not, and the status of the block signals on both exits are displayed as well.

Additionally the indicators and markers, that have been assigned to the block or to the route, are displayed. The status of each indicator, whether occupied or not, and the usage of each marker as a brake or stop marker for a certain direction are displayed here, too.

If the digital system, to which these indicators belong, is running in offline mode, then you can toggle the state of each indicator by clicking on it with the mouse. In this way the movements of trains can be simulated very conveniently: simply select the block that you want to look at on the computer screen and click on the occupancy, brake or stop indicator to simulate what happens if a train passes this indicator. Please refer also to chapter 9, “The Simulator”, for further details about simulation.

In **TrainController™** **Gold** it is furthermore possible, to open as many traffic control windows as desired. Other versions of the software are limited to display of only one traffic control window at a time.

The following options are additionally available in **TrainController™** **Gold**:

- **Pin to current Window:** With this option the traffic control becomes associated to the currently active window. Even if that window becomes inactive the traffic control will only display objects, that are selected in this window. This option can be used to permanently watch the status of objects, that are selected in a certain window. If a traffic control is pinned to a certain train window, for example, then this traffic control will display only those trains, that are driven by this train window. This is for example useful, if a traffic control is grouped together with a certain train window (e.g. docked side by side or tabbed). In this way it is possible to create a “super” train window, that contains a regular train window combined with a traffic control, that always shows the status of the train, which is currently selected in the train window.

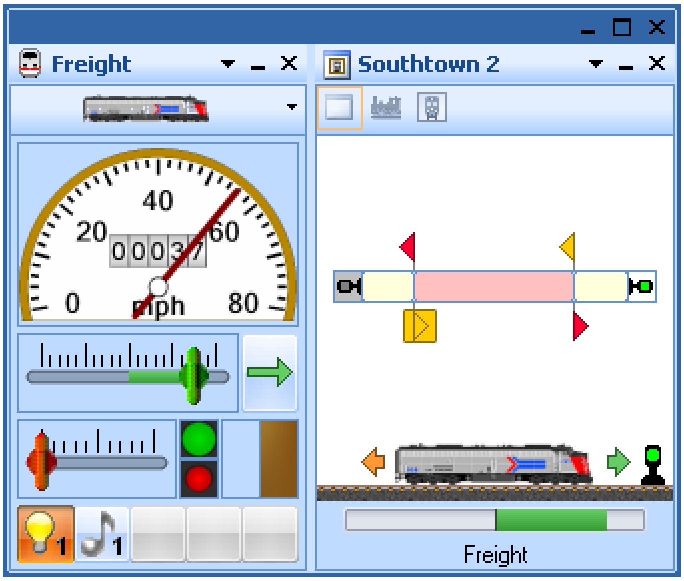


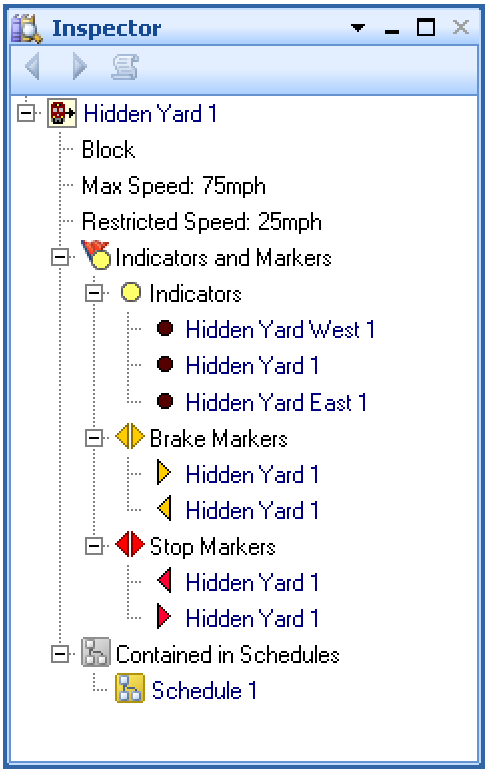
Diagram 128: Grouped Train Window and Traffic Control

- **Pin to current Train:** With this option the traffic control becomes associated to the currently selected train. Even if another object is selected this traffic control will continue to display the status of this train. This option can be used to permanently watch the status of a certain train.
- **Pin to current Block:** With this option the traffic control becomes associated to the currently selected block. Even if another object is selected this traffic control will continue to display the status of this block. This option can be used to permanently watch the status of a certain block (e.g. the entrance to a station).

7 The Inspector

B

The *Inspector* helps you to have an overview of the objects of your model railroad - this is especially very useful in the case of large layouts with many *turnouts*, *signals*, *routes*, *engines*, *trains*, *blocks*, *schedules*, etc. The Inspector clearly displays the properties of the currently selected object. The references to other objects (for example turnouts in routes or blocks in schedules, etc.) are visible, too. With a click it is possible to skip to other referenced objects, to view their properties. Important attributes like the name or digital address of objects can be edited directly in the Inspector without the need to go through separate dialog boxes.



```
Hidden Yard 1
    Block
    Max Speed: 75mph
    Restricted Speed: 25mph
    Indicators and Markers
        Indicators
            Hidden Yard West 1
            Hidden Yard 1
            Hidden Yard East 1
        Brake Markers
            Hidden Yard 1
            Hidden Yard 1
        Stop Markers
            Hidden Yard 1
            Hidden Yard 1
    Contained in Schedules
        Schedule 1
```

Diagram 129: Inspector

8 The Message Window and Pins

8.1 The Message Window

B

With the *Message Window* you can keep yourself up to date about the events occurring in **TrainController™** while operating your model railroad with the computer. In certain situations **TrainController™** displays informative, warning or error messages in the *Message Window*.

Most of these messages are generated by the *Dispatcher* (see chapter 5, “The Visual Dispatcher”). A special mode enables displaying of additional informative messages, which are useful to search for errors during creation of your automatic control system with the *Dispatcher*.

Using *system operations* (see page 286) it is additionally possible to display user defined messages in the *Message Window*.

The different types of messages are marked with different symbols.

Symbol	Meaning
✓	Informational message. This type of message is often displayed, when a certain operation has been completed successfully.
⚠	Warning. The related action is performed, but certain problems may occur.
✖	Error. The execution of the related action is aborted.
☠	Fatal error. This message is for example displayed, when an object needed to perform the current action, has been deleted by the user. Normally a user intervention is necessary, to correct the data.
?	Question. Recommendation to check, whether a certain setting is intended or not.
⌛	Planned wait.
🚦	An engine or train is ready to be controlled manually.

✎	Custom message – generated by *system operation*.
🔍	Detail message. Messages of these type can be optionally displayed to ease the search for errors when the control system is created.
ⓘ	Additional information.

It is also possible, to copy the text of messages to the clipboard or to save it to a text file.

Dr. Railroad

! Dr. Railroad is another outstanding feature of **TrainController™.** This function checks all data entered into **TrainController™** with artificial intelligence and detects automatically logical and other failures, lists them in the message window and gives hints to correct them.

In **TrainController™** **Gold** it is possible to store a Dr. Railroad message with own text in the properties of each object. When calling Dr. Railroad this message is then displayed together with the name of the object in the message window.

In this way you can for example mark objects, which must be still edited. With the help of Dr. Railroad these objects are displayed in the message window and their properties can be accessed directly from here.

8.2 Pins

Pins are an outstanding and unique feature of **TrainController™**. Pins make error diagnostics and debugging significantly easier and much more intuitive by showing information, where the events happen, rather than in a plain list like in the message window. Thereby troubleshooting becomes literally much more targeted.

Pins are an important expression of our aspirations to make train operations not only as realistic as possible, but also to make the path to this goal as easy and intuitive as it can be.

Pins are displayed in different types and colors on the computer screen with small markers.

Pins can be made visible with commands in the **Pins** group of the **View** tab of **TrainController™**. While pins are visible, the color of the windows, where pins are displayed, is being changed to gray scale. Thereby it is easier to locate the pins.

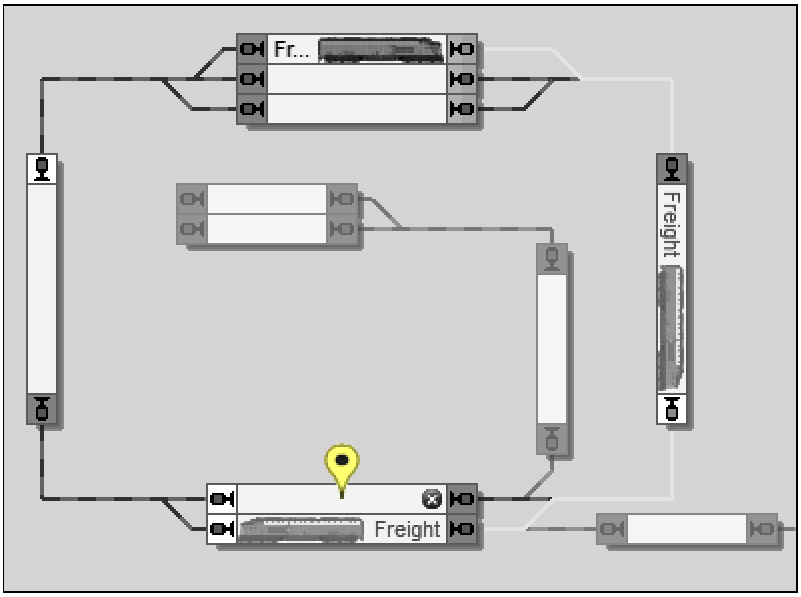
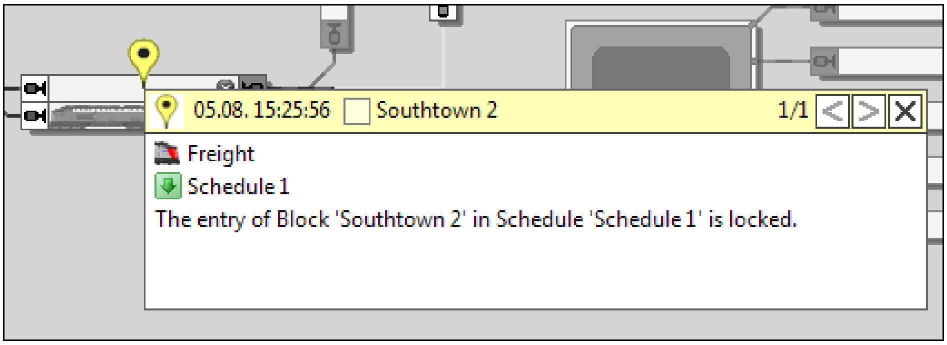


Diagram 130: Display of a pin in the Dispatcher Window

By clicking on the marker of a pin the pin can be expanded and collapsed. An expanded pin shows a window, which displays the information of this pin.


05.08. 15:25:56 Southtown 2 1/1

Freight

Schedule 1

The entry of Block 'Southtown 2' in Schedule 'Schedule 1' is locked.

Diagram 131: Expanded pin in the Dispatcher Window

Diagram 130 shows an active schedule, which contains the blocks “Southtown 1” and “Southtown 2” at the bottom. The train chooses “Southtown 1” for a certain reason and a yellow pin is displayed in block “Southtown 2”. If you want to know, why the train did not choose “Southtown 2”, click to the pin. The pin is expanded and displays the reason (Diagram 131): “Southtown 2” was not choosen, because its entry was locked.

This is a very simple example, but in more complex situations you will come to appreciate, that important information is displayed, where the events happen.

There are the following types of pins:

- System pins
- Dr. Railroad pins
- User pins

System Pins

System pins are automatically created by **TrainController™** during the operation of the layout. In many cases system pins are created together with the messages, which can be made visible in the message window. But there are also many pins, which do not have a corresponding message.

System pins are implicitly filtered to the context of the window, where they are displayed. By selecting a certain train in the switchboard or dispatcher window, for example, only the systems pins associated with this train are displayed. The diagram of a schedule in the dispatcher window, for example, displays only the pins which were generated by the execution of this schedule. System pins can also be filtered explicitly to the pins generated in the current session or during the most recent execution of a schedule. These features support the focus on the key information for the current debugging.

System pins are displayed in three colors:

- **Red / Error:**
  Red pins appear, if a certain operation fails, e.g. the execution of a schedule.
- **Yellow / Warning:**
  Yellow pins appear, if a certain operation is performed with limitations or another operation is chosen instead.
- **Green / Ok:**
  Green pins indicate the successful execution of operations. Green pins can be used, for example, to track the path of a train under control of a schedule.
  The display of green pins is turned off by default to allow you to focus on possible problems.
  The diagram below shows the same situation like Diagram 130, this time with the display of green pins turned on. The path chosen by the train is clearly visible now.

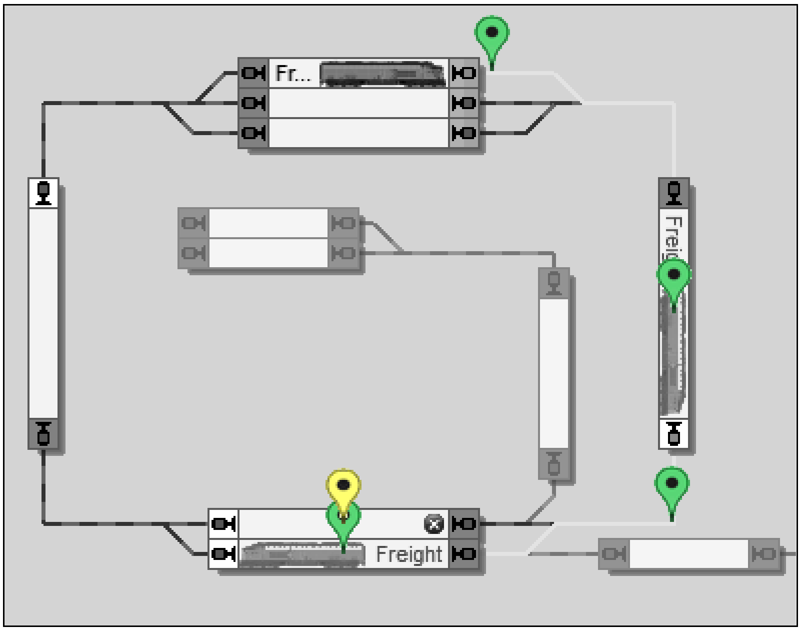


Diagram 132: Green pins marking the path of a train in the Dispatcher Window

Dr. Railroad Pins

The information generated by Dr. Railroad can be made visible with pins, too. These pins correspond to Dr. Railroad errors and questions, which are normally displayed in the message window. Dr. Railroad pins appear at the location of erroneous objects on the computer screen.

Dr. Railroad pins are displayed in two colors:

- **Red / Warning**
- **Yellow / Question**

User Pins

User pins can be created by the end user. After selecting an object on the computer screen and creating a user pin a marker appears at the location of this object. The text, which belongs to each user pin, can be freely edited.

User pins are displayed with blue color. They can be deleted at any time and are stored in the project file.

User pins are very useful to leave notes for further processing of the data for the end user himself or other users.

If an end user has to interrupt editing of data of a specific object, for example, he can create a user pin for this object with a note, what to do next, when the work is continued.

If an end user has a problem with the execution of a certain schedule by a certain train, and wants to consult an expert user, for example, he can create one or more user pins with details about his problem, e.g. for the block of the schedule, where the train is currently located, leave these pins expanded, store the project file and send the file to the expert user. When the expert user opens the project file, the user pins showing the location as well as the detail information of the problem will become visible in expanded state on the screen at once. This is a significant improvement for the exchange of relevant information in such cases.

9 The Simulator

With **TrainController™** it is possible to simulate the operation of a model railroad automatically and without human intervention.

The traffic control (see chapter 6, “The Traffic Control”) allows you to simulate the movement of running trains by triggering of the contact indicators, that belong to the particular blocks. Simulated triggering of the contact indicators is accomplished by clicking on the particular indicators with the mouse.

The simulator window can run such simulation automatically without the need for manual clicks on indicators. To start the simulation open the simulator window via the **Window** tab and press the **Start** button in the simulator window.

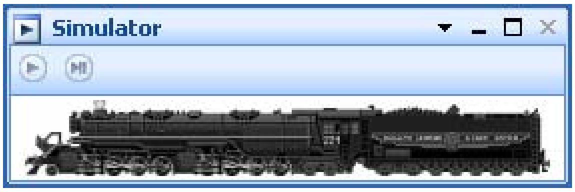


Diagram 133: The Simulator Window

The following prerequisites must be fulfilled to run the simulation:

- The software must run in offline mode, i.e. the computer must not be connected to a digital system.
- The software must run outside edit mode.

The simulator solely simulates the triggering of indicators by running trains. It does not operate anything. In particular it does not affect the speed or direction of running trains directly nor does it start or stop any trains. The speed of trains is set by the usual means – e.g. by running schedules or by using the controls of the train window. For running trains, however, the simulator is able to calculate, which contact indicator will be triggered next and when. These calculations are based on the current position of each running train and the path, which it is about to take. Note, that only those contact indicators are simulated, that are directly assigned to a block (see section 5.6, „Blocks and Indicators“).

Starting the Program in Offline Mode

If the END key of your keyboard is pressed and held at the start of a session or when a file is being loaded, then the session will start in offline mode. This is useful for working with the simulator or when the most recently used digital systems are currently off or not connected. In this case, a failing connection with possibly subsequent errors is avoided. Moreover, all the settings of the digital systems (e.g. COM port numbers) remain unchanged for later online operation.

Saving and Restoring of Train Positions

In connection with the simulator, but not only here, it is useful to be able to save and restore the current positions of all trains in the blocks as well as the compositions of train sets in separate files.

TrainController™ Gold supports this with special menu commands.

At the end of a session even the current train positions and configurations of train sets are stored and reloaded at the beginning of the next session. This works here only because the layout data does usually not change between sessions.

Additional efforts, however, may be seen in the following cases:

- If the connection to the layout is closed for testing purposes, to try out new settings with the simulator, the positions of trains or composition of train sets in the program may change in the simulation. After reconnecting with the layout it may happen, that the new train positions stored in the program do no longer match the actual positions of the trains on the layout. Rather than updating all train positions and possibly all compositions of train sets by hand, it is now possible in **TrainController™ Gold** to save the status of trains in a separate file with a special menu command before starting the test, and to restore the train status after the test and before the operation with the real layout is continued.
- However, this feature is not only useful when testing with the simulator. Sometimes new settings are first entered into the project on a trial basis and their test is performed with the layout connected. Here, the actual positions of trains on the layout are changed, of course. If the new settings are later discarded for some reason, the last saved state of the project is simply loaded. Here, however, also the old train positions and composition of train sets are restored from the previous project state, too. The changes of the train positions on the layout made through the test should therefore be updated again by hand. Instead it is also possible to save the new train positions in a separate file just before going back to the original project status. After

loading the original project data, the current train positions are then restored from that file.

10 A Sample Layout

B

General

The layout displayed below will be operated with **TrainController™**:

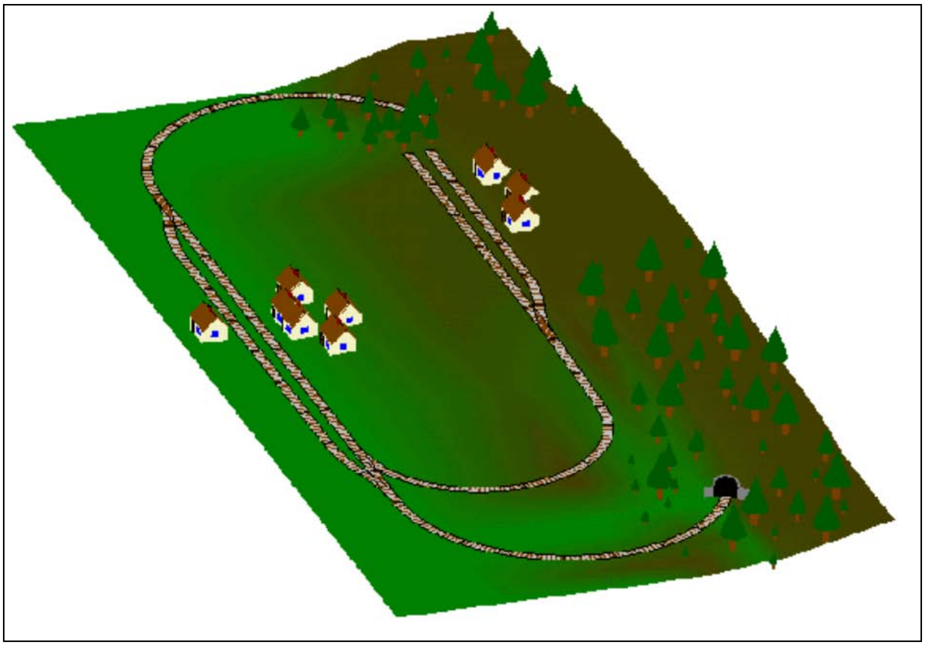


Diagram 134: Sample Layout

The layout has two stations: “Southtown” located on the left side of the layout and “Northville” located at the end of the branch line. There is an additional hidden yard that is covered by the mountain.

This can be seen better in the track plan displayed below:

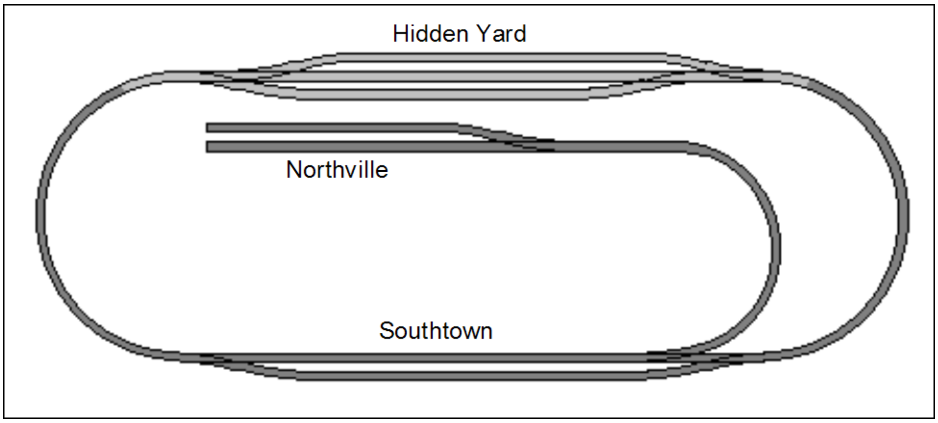


Diagram 135: Track Plan of the Sample Layout

The main line, i.e. the loop that connects “Hidden Yard” and “Southtown”, will be operated automatically under full control of the *Dispatcher*. The branch line from “Southtown” to “Northville” will be operated manually.

! In the following the necessary steps to control this layout are explained. **TrainController™** is installed with a set of sample files called STEP1.YRR to STEP5.YRR. Each of these file corresponds to the content of one of the following sections. By loading these files into **TrainController™** you can reconstruct for yourself, how the particular steps are performed.

Step 1: Creating the Switchboard

The first steps are creation and drawing of the *switchboard*.

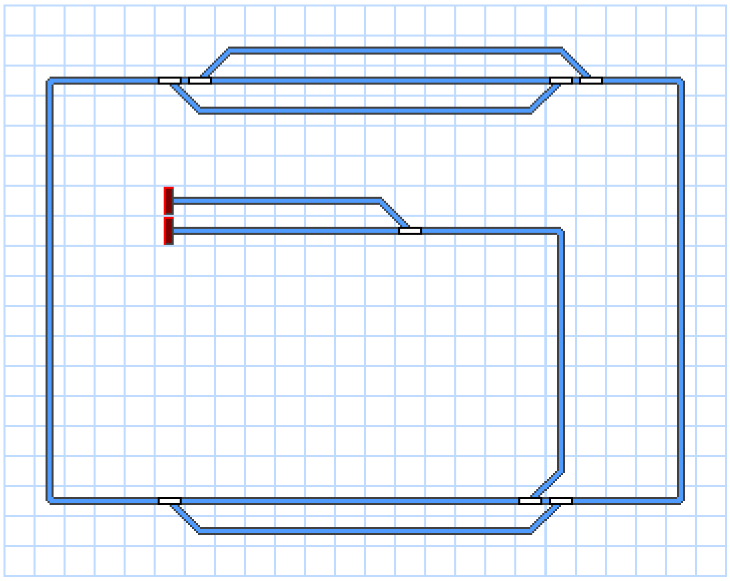


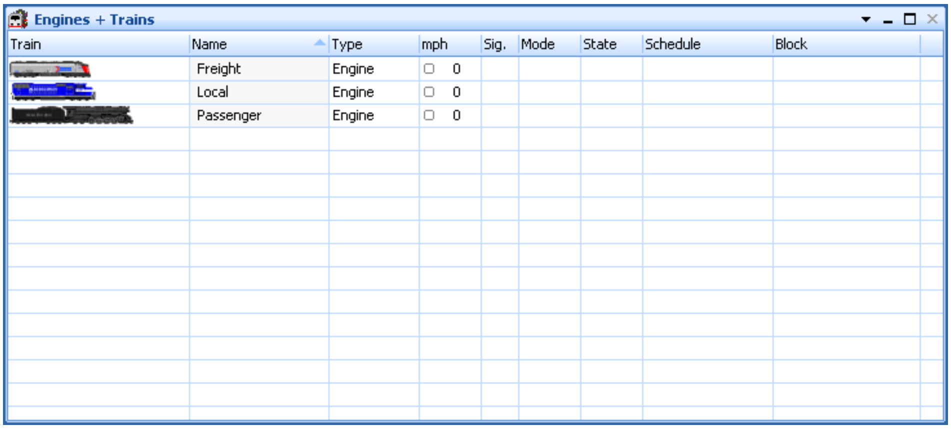
Diagram 136: Switchboard Southtown

Diagram 136 shows the switchboard of the sample layout. All turnouts get appropriate names. The related digital addresses are assigned, too.

At this stage we are able to control all turnouts on our sample layout.

Step 2: Defining the Engines

Our switchboard is now completed and we are going to create the entries for the engines that we want to run on the layout. We want to run three trains, a passenger and a freight train that can run on the main line only, and an additional train that can also go to Northville. The trains are entered into the Train Window as displayed below:


Engines + Trains

Train	Name	Type	mph	Sig.	Mode	State	Schedule	Block
	Freight	Engine	▢ 0					
	Local	Engine	▢ 0					
	Passenger	Engine	▢ 0					

Diagram 137: Engine list

By editing the properties of each engine we assign a digital address to each engine and can additionally specify engine functions, measure the threshold speed and the speed profile and edit other properties. This is not outlined in detail here, because it is not important for understanding of this sample layout. Further details can be found in chapter 3, “Train Control”.

The images have been prepared with **TrainAnimator**™.

Through the **Window** tab of the software you can open additional Train Windows, if you want to control each train through a separate Train Window.

At this stage of the sample we are able to control our trains manually with the computer on all parts of the sample layout.

Step 3: Creating Blocks

At first we divide our layout into logical blocks. We follow the guidelines on page 145. The resulting block structure looks as follows:

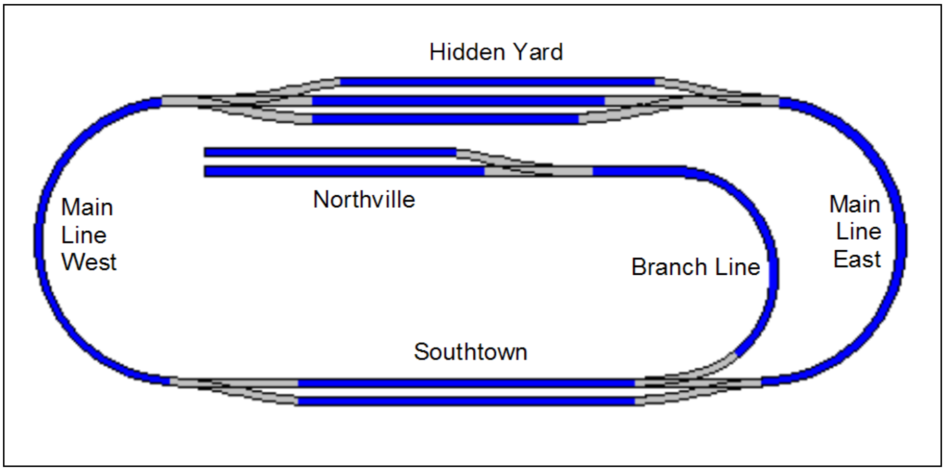


Diagram 138: Block structure of the sample layout

Each blue track section represents a separate block.

Based on this diagram we insert a block symbol for each block into the switchboard. The resulting switchboard is displayed in the next diagram:

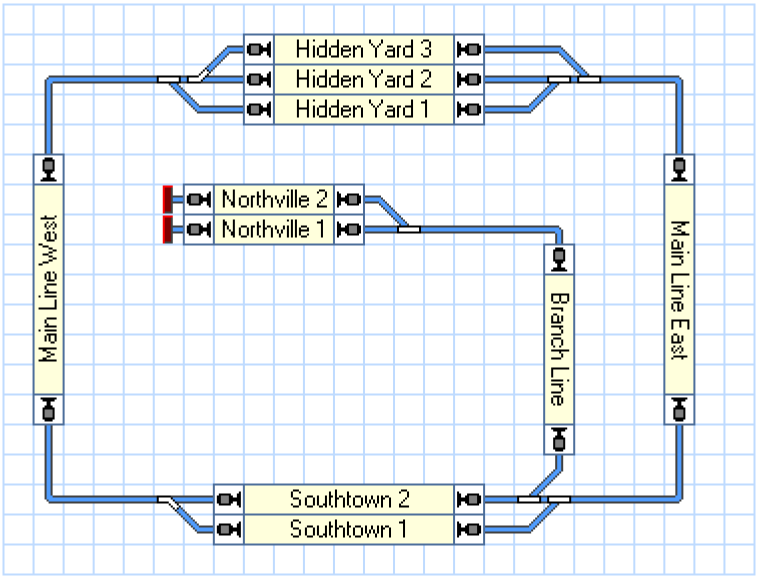


Diagram 139: Switchboard with Blocks

Based on this switchboard the *Visual Dispatcher* automatically calculates the following block diagram:

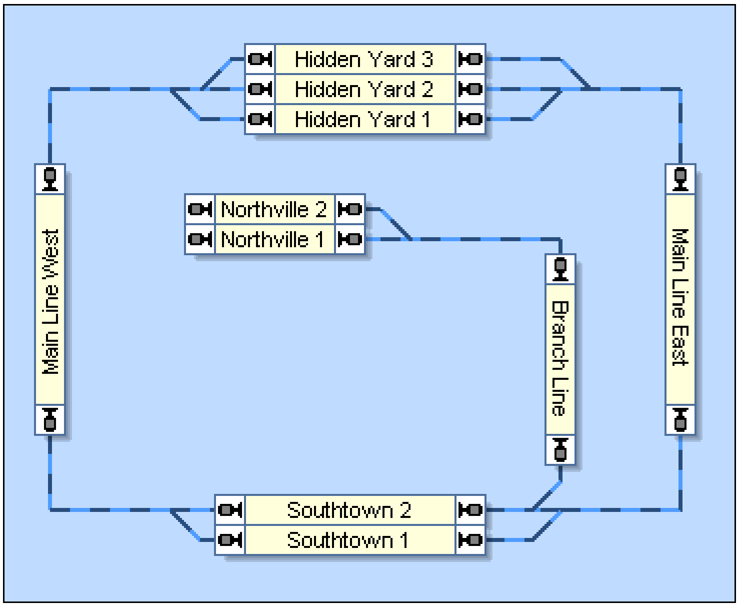


Diagram 140: Block Diagram in the Visual Dispatcher

Please note that the block diagram represents the track layout in rough outline. The actual track connection between "Main Line West" and "Hidden Yard 3", for example, contains two turnouts. These turnouts are not drawn in the block diagram in detail. Instead a route between both blocks is created.

All necessary routes between all blocks are created and recorded automatically.

Step 4: Contact Indicators

We want to equip each block on the main loop with three occupancy sensors. The arrangement of indicators of each block follows Diagram 109 (please refer to page 167). The occupancy sensor in the center of each block (dark red zones in Diagram 141) will be used as brake indicator for both directions; the sensors on both sides of each block will be used as stop indicator for the related direction (light red zones in Diagram 141).

The branch line to "Northville" contains 3 blocks. Since we do not want to run automatic trains there it is sufficient to install one occupancy sensor in each of these blocks for train tracking of manual trains.

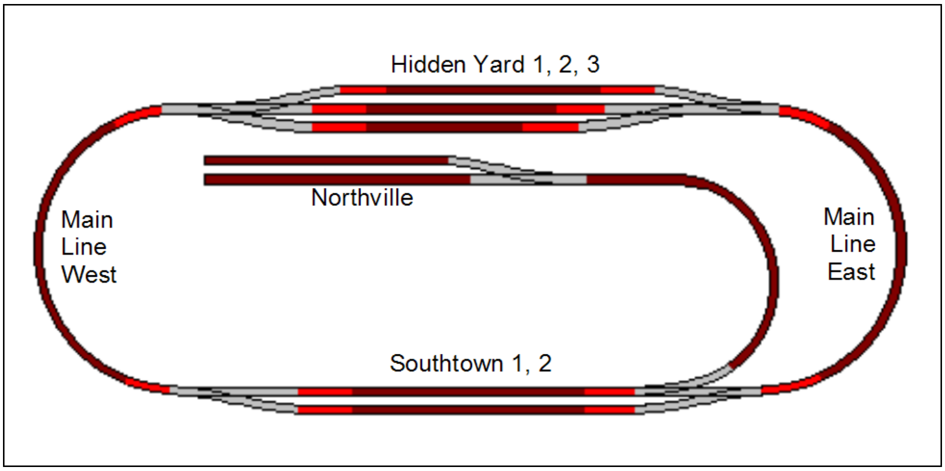


Diagram 141: Indicator arrangement of the sample layout

The grey tracks in Diagram 141 are not contained in any block. They are part of routes, which are assumed to be located between the blocks.

Indicators are created for each block according to the following table:

Block	Indicator	Markers
Hidden Yard 1	Hidden Yard 1	◁▷
	Hidden Yard East 1	▶
	Hidden Yard West 1	◀
Hidden Yard 2	Hidden Yard 2	◁▷
	Hidden Yard East 2	▶
	Hidden Yard West 2	◀
Hidden Yard 3	Hidden Yard 3	◁▷
	Hidden Yard East 3	▶
	Hidden Yard West 3	◀
Main Line East	Main Line East	△▽
	Hidden Yard East Entry	▲
	Southtown East Entry	▼
Main Line West	Main Line West	△▽
	Hidden Yard West Entry	▲
	Southtown West Entry	▼
Southtown 1	Southtown 1	◁▷
	Southtown East 1	▶
	Southtown West 1	◀
Southtown 2	Southtown 2	◁▷
	Southtown East 2	▶
	Southtown West 2	◀
Northville 1	Northville 1	◁▷◀▶
Northville 2	Northville 2	◁▷◀▶
Branch Line	Branch Line	◁▷◀▶

Table 5: Indicator Configuration

The small icons indicate in which direction of travel a certain indicator is effective as brake or stop marker. The indicator “Hidden Yard 1”, for instance, marked by ◁ and ▷ is used as brake marker of block “Hidden Yard 1” for both directions of travel. The indicator “Southtown East Entry”, marked by ▼ is used as stop indicator of block “Main Line

East” for trains that pass this block from the top to the bottom of the layout, i.e. from the Hidden Yard to Southtown. For trains running to the opposite direction this indicator reports that the train enters the block.

Step 5: Creating Schedules

One single schedule is sufficient to describe all train movements on the main line of the sample layout:

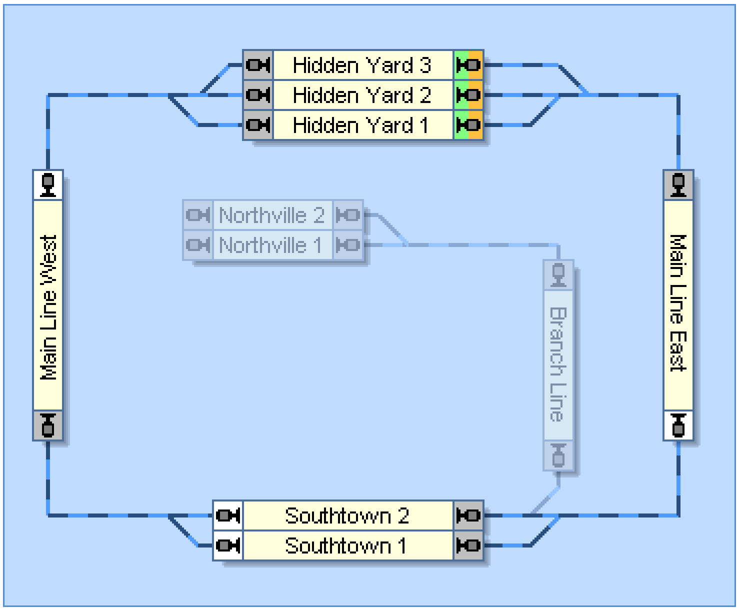


Diagram 142: Schedule Diagram of the Sample Layout

The blocks in “Hidden Yard” are marked as start blocks of the schedule. Since the schedule forms a closed loop these blocks are automatically calculated as destination blocks, too. The schedule can be started to both directions, i.e. trains can run clockwise or counter-clockwise under control of this schedule. Depending on a specific setting of this schedule, it allows either for train movements that start in Hidden Yard or for train movements that start in any other block of the main loop. All train movements will end in “Hidden Yard”, though.

Manual Operation

The branch line from “Southtown” to “Northville” and back will be operated manually.

All precautions for train tracking have been already done by integrating the blocks of the branch line into the main block diagram accordingly.

Trains waiting in “Southtown” and bound to “Northville” will release block “Southtown 2” as soon as they leave “Southtown”. They will be automatically tracked to “Northville” and back. All is done by proper drawing of the main block diagram, no further actions are needed. A train that comes from “Northtown” and arrives in “Southtown” will automatically reserve block “Southtown 2” again.

From there it can be started by the schedule shown in the previous section and automatically travel to the “Hidden Yard”. This can even be done automatically on arrival in “Southtown” without further intervention by the human operator.

Further Steps

Now the framework has been completed, we can add varied operation.

It is for example possible to assign the schedule to the operations of a push button symbol in a switchboard in order to start automatic operation manually when desired.

Another schedule can be added with “Southtown 2” as destination block. This schedule will lead a train to “Southtown 2” where it can be taken over to perform a manually operated travel to “Northville”.

It is also possible to start the schedules created here according to a time table.

A hint in case you have configured endless automatic operation: by adding a global on-off switch symbol located somewhere in a Switchboard to the conditions of some or all of your schedules you can implement a global power off mechanism. Lets assume that the schedules can only be started, if this global on-off switch is on. If this switch is turned off during automatic operation then all trains will finish the run of the current schedule and will not start another run of any schedule that is restricted by this on-off switch. In this way you can smoothly shut-down your automatic operation in a very clean way.

These and more advanced techniques will be explained in Part III of this Users Guide.

Part III

Extensions

X

This Part III of the Users Guide explains the extended features of **TrainController™**. These features enable advanced users to make professional use of the software.

Novice users should focus to the previous Part II first and should not read any further, before they put the content of Part II into practice. With the possibilities outlined in Part II you can control your complete layout manually and perform basic automatic operation of your trains.

11 Advanced Train Control

11.1 Train Sets in TrainController™ Silver

A train set is composed of a couple of engines. Train sets can be created, arranged and dissolved at any time during operation of the layout.

Train sets are used to accomplish realistic *multiple unit* operation, i.e. operation of trains, that contain more than one engine.

Similar to real railroads each single engine, which is operated separately, or each car, which is located isolated on your layout, can also be seen as a train. For this reason the term train is usually used in **TrainController™** as a generic term for each engine, isolated car or complete train set.

Train sets can be arranged in the **Engines + Trains** window by dragging the symbols of engines with the mouse. They can also be arranged with the **Train Set dialog box**.

To operate the speed or direction of a train set select an arbitrary engine currently contained in this train set in the **Train Window**. Changing the speed or direction of this vehicle will also be applied accordingly to all other engines contained in the train set. The speed and direction controls of the Train Window always reflect the status of the selected individual vehicle rather than the complete train set. If there are several engines with different speed characteristics assigned to a train set, i.e. the engines run with different speed at the same speed step, then **TrainController**™ is able to balance out the different behavior of the engines. This requires correct adjustment of the *speed profile* of each affected engine, however (see section 3.5, “The Speed Profile”).

Multiple unit operation via train sets is also possible with the throttle of your digital system. To operate the speed or direction of a train set select an arbitrary engine currently contained in this train set on the throttle of your digital system. Changing the speed or direction of this vehicle with the digital throttle will also be applied accordingly by **TrainController**™ to all other engines contained in the train set.

To operate the auxiliary functions of a certain engine select this particular vehicle in the Train Window, too. The operation of functions, however, only applies to the currently selected vehicle. The function buttons of the Train Window always reflect the status of the currently selected vehicle. In other words: for manual operation of functions it does not matter, if the vehicle is currently contained in a train set or not. The effect is limited to the particular vehicle.

Like in real railroads a certain vehicle can only run as part of one train at a time. If a vehicle is successfully added to a train set, then it is automatically removed from its previous train set, if any.

Operation of Additional Function Only Decoders in TrainController™ Silver

Function only decoders are often used to add additional functions to a decoder controlled locomotive or to other rolling stock. An example is lighting in passenger cars. These decoders can be controlled with **TrainController**™ **Silver**, too.

This is done by setting up a “dummy engine” with the digital address of the function only decoder. The speed settings of this decoder don't matter in this case. The function setup of this decoder is done as outlined in section 3.6, “Headlights, Steam and Whistle”.

Manual operation of the extra functions provided by the function only decoder is done by selecting the “dummy engine” in the **Train Window** and operating the function buttons of this engine.

For automatic operation of the extra functions provided by a function only decoder it is necessary to arrange a train set and to setup the “dummy engine” representing the function only decoder along with the actual engine as multiple unit. If different function symbols are used for the functions of the actual engine and the functions provided by the function only decoder, then it is possible to select and activate the specific functions of the function only decoder automatically without affecting the functions of the actual engine.

Example: Automatic Car Lighting in TrainController™ Silver

The following example demonstrates how a train can be prepared in **TrainController**™ **Silver** in order to operate the car lighting in this train automatically. It is assumed that the lighting is controlled by an additional function only decoder. Perform the following steps:

- Create and setup an engine “Loco” for the actual engine heading the train.
- Create an additional engine “Dummy” and specify the digital address of the function only decoder.

- Setup the function symbols for the functions provided by the function only decoder in the engine “Dummy”. Use a unique function symbol for the car lighting that has not been already used for functions of the actual engine “Loco”.
- Arrange the two engines created above as train set.
- Assign the function symbol representing the car lighting to the operations of a *schedule*, a *macro* or an *indicator* (see also Diagram 124 or Diagram 166) as desired.

11.2 Cars and Train Sets

Cars

Cars represent vehicles of your model railroad, that are not equipped with a motor. Examples are passenger cars or freight cars. For each car you can specify the following attributes among others:

- digital decoder address, if the car is equipped with a function decoder
- name and image
- length and weight
- maximum allowed speed
- auxiliary functions (e.g. light, smoke or coupler)

Cars are mainly used for trains, that change their formation during operation, and to accomplish the following tasks with these trains:

- the maximum speed of a certain locomotive will vary and depend on the pulled cars (e.g. fast passenger train vs. slow freight train, both pulled by the same engine at different times)
- the same locomotive will be directed to different tracks according to the cars it is currently pulling (e.g. passenger train may go to the platform track while a freight train pulled by the same locomotive at another time must not go there)
- trains will be directed to different tracks according to their current length (see also page 344)
- trains will be always able to stop in the middle of a block (e.g. a platform) even if they pull trains of varying length
- the tonnage of trains will be simulated realistically according to the current weight of the cars contained in the train

Cars are mainly needed to accomplish the purposes outlined above and in general advanced operation of your model railroad. For reasons of simplicity novice users should postpone the usage of cars until they are familiar with the program.

!

Even though it is possible to create a car record in TrainController™ Gold for each particular car of your model railroad it is recommend to go with as few cars as possible.
If a certain train does not change its formation during operation, then it is sufficient to create a simple engine record for such train. Engines also have a length and a weight and if these attributes do not change during operation you can specify the length or weight of the complete train pulled by this engine in the properties of the engine, too.
If a certain set of cars is always coupled together but pulled by different engines, then one should create only one single car object in TrainController™ for this set and assign an appropriate name, image, length and weight to it, which represent the according attributes of the complete set of cars.

Train Sets

A train set is composed of a couple of engines or cars. Train sets can be created, arranged and dissolved at any time during operation of the layout.

Train sets are not only used to operate cars together with engines to accomplish the purposes outlined in the previous section. Train sets are also used to accomplish realistic *multiple unit* operation, i.e. operation of trains, that contain more than one engine.

Similar to real railroads each single engine, which is operated separately, or each car, which is located isolated on your layout, can also be seen as a train. For this reason the term train is usually used in **TrainController™** Gold as a generic term for each engine, isolated car or complete train set.

Train sets can be arranged in the **Engines + Trains** window by dragging the symbols of engines and cars with the mouse. They can also be arranged with the **Train Set dialog box** displayed below:

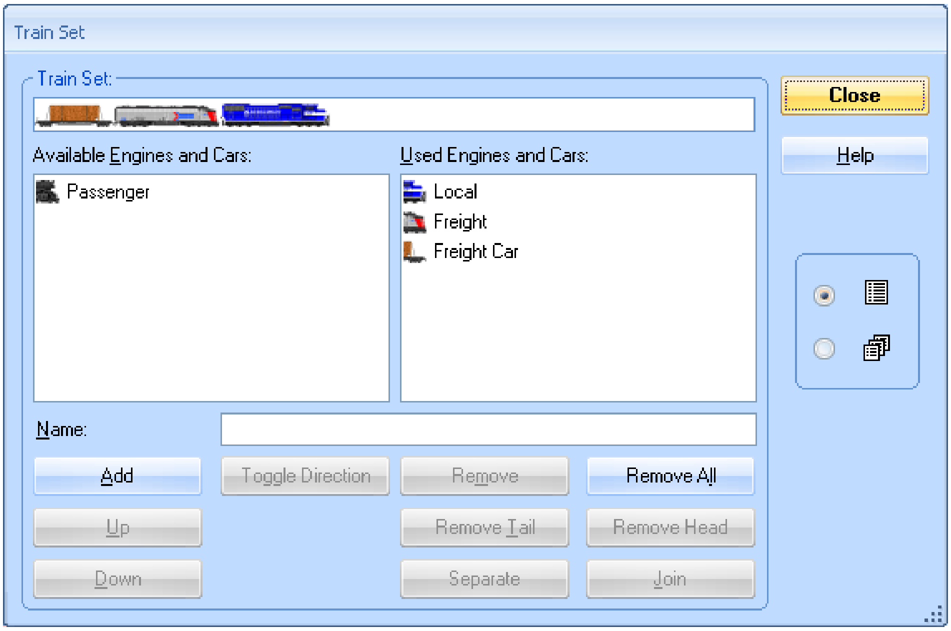


Diagram 143: Arranging a Train Set

The options of this dialog allow to add engines or cars to a train set, to remove vehicles, to change the direction of a vehicle within the train set or to split train sets into two other train sets.

To operate the speed or direction of a train set select an arbitrary engine currently contained in this train set in the Train Window. Changing the speed or direction of this vehicle will also be applied accordingly to all other engines contained in the train set. The speed and direction controls of the Train Window always reflect the status of the selected individual vehicle rather than the complete train set. If there are several engines with different speed characteristics assigned to a train set (multiple unit), i.e. the engines run with different speed at the same speed step, then **TrainController™ Gold** is able to balance out the different behavior of the engines. This requires correct adjustment of the *speed profile* of each affected engine, however (see section 3.5, "The Speed Profile").

Multiple unit operation via train sets is also possible with the throttle of your digital system. To operate the speed or direction of a train set select an arbitrary engine currently contained in this train set on the throttle of your digital system. Changing the speed or direction of this vehicle with the digital throttle will also be applied accordingly by **TrainController™ Gold** to all other engines contained in the train set.

To operate the auxiliary functions of a certain engine or car select this particular vehicle in the Train Window, too. The operation of functions, however, only applies to the currently selected vehicle. The function buttons of the Train Window always reflect the status of the currently selected vehicle. In other words: for manual operation of functions it does not matter, if the vehicle is currently contained in a train set or not. The effect is limited to the particular vehicle.

Like in real railroads a certain vehicle can only run as part of one train at a time. If a vehicle is successfully added to a train set, then it is automatically removed from its previous train set, if any.

Control Cars

Cars can also be characterized as control cars. Control cars differ from normal cars in the following characteristics:

- They can also send direction commands to an associated decoder. In this way it is possible to operate direction-dependent functions (e.g. headlights or tail light) also on a car.
- When they are included in a train set, they can be selected in the train window just like locomotives to control the speed and direction of the entire train set.

Cars and Load

For realistic simulation of train tonnage it is possible to specify the full weight and the empty weight for each car. With a specific menu command it is possible to toggle between both weights, i.e. it is possible to simulate loading and unloading of cars. The currently selected weight (load condition) of each car is applied to the calculation of the maximum speed or the acceleration momentum of affected train sets.

Cars can also be loaded or unloaded automatically during operation, e.g. during a running schedule.

Automatic loading and unloading of cars provide also the possibility to decrease the acceleration momentum of trains running in hidden areas of the layout, if desired. To accomplish this automatically, specify appropriate schedule operations, that unload all cars, when they enter the hidden area, and that load all cars again, when they leave these areas.

Forwarding of Functions in Train Sets

Automatic operation of auxiliary functions called for a train set are normally performed by the first engine or car only. The first engine or car in a train set is that one, which is displayed at the rightmost position in the image of the train set in the **Engines + Trains** window.

In order to allow auxiliary functions to be performed by other vehicles in a train set, too, it is possible, to turn on **function forwarding**. This is accomplished with a certain train operation, with which function forwarding can be turned on or off. This operation is usually used in macros. Assume a macro, with a first operation to turn function forwarding on and a second operation to call an auxiliary function to turn on the lights. If this macro is called for a train set, then the lights are turned on in all vehicles in the train set, which are able to perform this function.

The opposing operation, i.e. turning off function forwarding, is also available in order to return to the default policy.

Joining and Separation of Train Sets

Vehicles can be added and removed from train sets by using certain menu commands. Additionally **TrainController™** **Gold** is able to add or remove vehicles from train sets automatically during operation without explicit human intervention.

To accomplish this each train set can be separated into two parts at a time. This can be done manually by calling an according menu command or automatically. If a train is separated, then **TrainController™** **Gold** draws a red triangular marker between the two separated vehicles. It is also possible to join a separated train set.

If the speed of an engine contained in a separated train set is changed, then this change only applies to the engines contained in the <u>same part</u> of the separated train set. If, for example, a train set consisting of two locomotives is separated and the first engine is accelerated, then the second engine remains standing still. If the first engine is detected in an adjacent block, then the train set is finally dissolved and the second engine remains in its current block.

Note, that for reasons of simplicity only one separation is supported per train set. If a train set, which is separated between the second and third vehicle, for example, is additionally separated between the fifth and the sixth vehicle, then the red marker is moved from its previous position to the gap between the fifth and sixth car and the second and third car are joined again.

If a train set will be divided into three or even more parts, then it has to be separated into two parts first. Before it is possible to separate one of these parts further it is necessary, that the other part leaves the current block first.

Arranging Train Sets by Train Tracking

If one part of a separated train set is manually moved to another block and detected there by train tracking, then the moved vehicles are automatically removed from the train set. The other vehicles form a new train, which remains in the previous block. The moved vehicles also form a new train, which is now located in the new block.

The opposing feature, i.e. joining trains by manual driving a locomotive or a train into a reserved block with other vehicles waiting there, is also possible. This is accomplished by checking the menu item **Enable Join by Train Tracking** in the **Train Set** group of the **Operation** tab for a specific locomotive or car. If such locomotive is manually driven into a reserved block with other vehicles already located there, then the locomotive and the vehicles are automatically joined and form a new train set.

The joining process begins, when the approaching train enters the block, where another train is already located. In this moment both trains are displayed in the block with a red triangle between them. Actually both trains form a single train set now, which is still separated, however. The separated train set is finally joined and the red rectangle disappears, when the approaching train stops. From this moment all engines in the new train set are operated together.

The maneuver outlined above even works, when the approaching train is operated with the throttle of your digital system. After the join has been completed, all engines of the new train set can be driven with the throttle of the digital system by operating one of the engines with the throttle. The speed of the other engines is automatically synchronised by the computer software. In this way multiple units can be created very conveniently and without additional intervention by simply driving trains to positions, where other trains are already located.

Even though the menu item **Enable Join by Train Tracking** can be set for locomotives or cars only, it has also an affect for train sets. If a train set with a vehicle at the according end with this menu item checked enters a reserved block, then the train set and the vehicles located in this block are automatically joined and form a new train set, too.

In order to make the above maneuver to work properly it is necessary, that the train, which enters the block, can be detected in this block. For this reason it is necessary that the entering train can trigger an indicator in this block, even though there are already other vehicles located there.

Arranging Train Sets by Schedules

It is also possible to start a schedule with a separated train set. Depending on the direction of travel forced by the schedule, however, only the corresponding part of the separated train set will start to move. When this part of the train set enters the next block, the train set will be dissolved by leaving the non-moving vehicles in the start block of the schedule.

The opposing feature, i.e. joining trains at the end of a schedule, is also possible. This is accomplished with a specific schedule rule, that allows trains to enter destination blocks of schedules, that are reserved by other trains. If the train, which executes a schedule with this rule turned on, enters a destination block of this schedule reserved by another train, then both trains are automatically joined to form a new train set. This is the only exception from the basic principle, that no train may enter a block, which is reserved by another train. This exception is also only available for destination blocks of schedules. In order to make this maneuver to work properly it is necessary, that the train, which runs the schedule and which will join the other train already sitting in the destination block, can be detected in the destination block. For this reason it is necessary that the entering train can trigger a brake and a stop marker in the destination block, even though there is already another train located in this block.

Arranging Train Sets By Operations

Train sets can also be joined or separated by operations, macros and engine functions. The following actions are possible:

- join a separated train set
- separate the first locomotive located at a certain end of the train from the train set
- separate all locomotives located at a certain end of the train from the train set
- separate the vehicle (locomotive or car) located at a certain end of the train from the train set
- separate a train set at a certain side of a vehicle

The end of the train, that will be separated from the train, is always specified based on the direction of travel relative to the layout (left vs. right; top vs. bottom). It is for example possible, to separate the first locomotive on the right end of the train. This is for example useful in a scenario, where an auxiliary locomotive is temporarily added to a train set to push a heavy train uphill. The direction of travel, which the block at the top of the grade is passed to by trains coming from the bottom, is always the same; e.g. from the right to the left. In such case the pushing locomotive is always located at the right end of the train., regardless whether the heading locomotive is running forward or backward. For this reason it is more useful to be able to separate the locomotive at the right end of the train rather than the locomotive at the tail of the train, which may point upwards, if the train set runs backwards.

In the above list the first four operations can be applied to train sets, usually as operations or macros executed at the beginning, at the end or during execution of a schedule.

The fifth operation can only be executed by vehicles (engines or cars), that are currently part of a train set. It is usually performed by a macro, which is called as an auxiliary function of the engine or car. This operation allows interesting maneuvers. Assume a train, that will be separated left of the caboose, when it enters a certain block. If the operation to separate a train set at the left side of a vehicle is added to a macro and this macro is specified as an auxiliary function of the caboose, then each train set containing the caboose can be separated left of the caboose by calling this auxiliary function; e.g. at the end of a schedule. Thereby it doesn't matter, at which position of the train set the caboose is placed.

! Note, that the features described in this section, i.e. joining and separation of trains, are *logical* features. They are needed by the software to perform book keeping of the arrangements of train sets. At any rate it is also necessary, to care about the *physical* coupling or uncoupling of the particular vehicles contained in a train set. This is to be done by additional means not covered here.

Operation of Additional Function Only Decoders in TrainController™ Gold

Function only decoders are often used to add additional functions to a decoder controlled locomotive or to other rolling stock. An example is lighting in passenger cars. These decoders can be controlled with **TrainController™**, too.

This is done by setting up a car with the digital address of the function only decoder. The function setup of this decoder is done as outlined in section 3.6, "Headlights, Steam and Whistle".

Manual operation of the extra functions provided by the function only decoder is done by selecting the car in the Train Window and operating the function buttons of this car.

Example: Automatic Car Lighting in TrainController™ Gold

The following example demonstrates how a train can be prepared in order to operate the car lighting in this train automatically. It is assumed that the lighting is controlled by an additional function only decoder. Perform the following steps:

- Create and setup an engine for the actual engine heading the train.
- Create a car and specify the digital address of the function only decoder.
- Setup the function symbols for the functions provided by the function only decoder in the car. Use a unique function symbol for the car lighting, that has not been already used for functions of the actual engine.
- Arrange the engine and car in a train set.
- Assign the function symbol representing the car lighting to the operations of a *schedule*, a *macro* or an *indicator* (see also **Diagram 124**) as desired.

Line-Up of Vehicles and Trains in a single Block

With **TrainController™ Gold** it is possible to line up more than one vehicle or train in a single block without connecting them to a continuous train set. This is for example useful, if you want to park several locomotives in a single track of a hidden yard.

Several features of **TrainController™ Gold** must be combined in order to accomplish this.

First the schedule rule **Line up in destination block** (see page 356) must be used in a schedule. This rule enables trains under control of this schedule to enter a destination block of this schedule, even if this block is already reserved by other vehicles. This rule works similar to the rule **Enter reserved destination block for joining** with the difference, that the incoming vehicles are expected to be stopped in a certain distance from the already waiting vehicles. This distance must be specified to activate the rule described above. The incoming vehicles are not connected with the waiting vehicles to a continuous train set. Both groups of vehicles remain separated. It is furthermore possible to line up vehicles in more than two separated groups.

If a manually operated train enters a block, where several vehicles are already lined-up in separated groups as described above, then the incoming vehicles are added to the waiting vehicles as a new group, which remains separated from the waiting vehicles.

Note, that line-up of vehicles in a certain block can only be initiated with a schedule and the rule described above, but not by train tracking. Once the vehicles in a block are lined-up in at least two separated groups, then each manually operated train subsequently tracked into this block by train tracking will form a new separated group.

Line-up by a schedule requires, that the distances and ramps of stop and brake markers in the concerning block are configured accordingly. Appropriate formulas should be used to calculate the space already occupied by the waiting vehicles and the additional gap between the waiting vehicles and the incoming vehicles (see also page 174). This is very important, because each incoming train under control of a schedule is expected to be stopped in the distance, which has been specified to activate the rule for line-up.

The assumed distances are used, when the lined-up vehicles are moved up. This happens <u>automatically</u>, when a group of vehicles at one of the two sides of the block leaves the block. If a group of vehicles at one end of the queue is moved to another block, then the group, which previously was the second group in this queue, is <u>automatically</u> moved towards the end of the block. The distance of this move results from the length of the vehicles in the first group plus the gap between both groups. Once the second group has completed this move, then the group, which previously was the third group in the queue, performs the same move, afterwards the fourth group, and so on. To enable the software to calculate the distances of this move accordingly, it is very important to specify all distances and lengths correctly. This includes also the maximum train length of the concerning block, all distances and ramps of all involved markers as well as the length of each involved vehicle, engine or car.

!

Note also, that line-up and move up of vehicles can only be performed in one direction at a time.

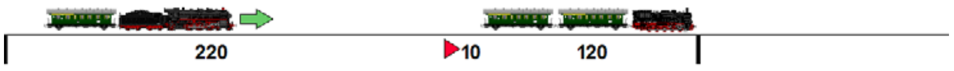


Diagram 144: Line-Up – Step 1

Assume, for example, that a train with length 100 cm is directed by a schedule from the left to the right into a destination block with maximum train length of 350 cm and another train with length 120 cm already waiting there.

The distance specified in the schedule rule for line-up is 10 cm. In this case the software assumes, that the waiting train is located at the right end of the block and the remaining space in the left part of the block is 220cm (350 cm minus 120 cm minus 10cm).

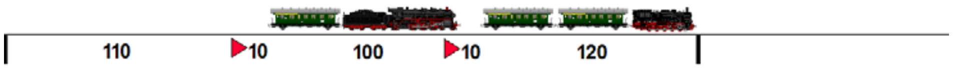


Diagram 145: Line-Up – Step 2

The incoming train will be stopped in a distance of 10 cm to the waiting train. Now 110 cm (220 cm minus 100cm minus 10cm) are still available in the left part of the block.

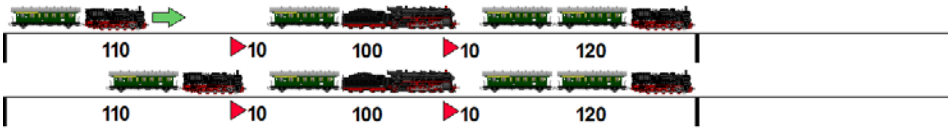


Diagram 146: Line-Up – Step 3

With the same schedule a third train with a length of up to 110 cm can be directed here – but also only if it enters the block from the left to the right.

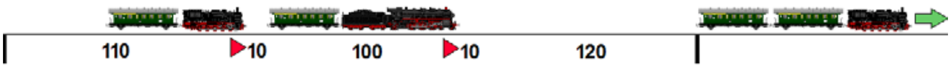


Diagram 147: First train has left the block.

When the first train (length 120 cm) leaves the block to the right later, then free space of 130 cm (120 cm plus 10 cm) becomes available in the right part of the block.

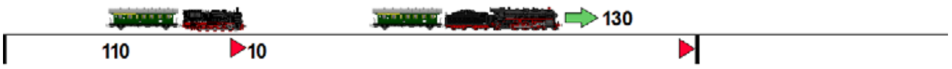


Diagram 148: Move up – Step 1

The second train in the queue will now automatically move up by 130 cm to the right.

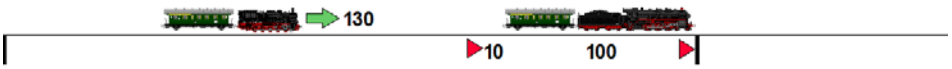


Diagram 149: Move up – Step 2

When this has been done, the third train will perform the same move.

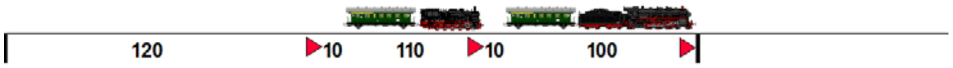


Diagram 150: Move up finished

Finally both trains are located at the right side of the block with 120cm free space in the left part of the block. Now another train with a length of up to 120 cm can be directed here - if it enters the block from the left to the right.

Line-up and move up for a certain block can by performed only in one direction – following a strict FIFO (first in – first out) principle.

!

For reasons of simplicity the software does not record the information, in which direction line-up and move up have been performed most recently. It just compares the available and the required space with each other. It assumes, that waiting trains are always located at suitable positions in the block and it derives the distance for the move up of the remaining trains solely from the length and the gap of the leaving train.

To perform line-up and move up accordingly, perform the following:

- Specify the maximum train length of the concerning block accordingly.
- Specify the length of each involved vehicle – engine and car – correctly.
- Capture the speed profile and brake compensation of each involved engine.
- Specify a sufficient value for the gap between two groups of vehicles in the schedule rule for line-up (at least 10cm or 4 inches – better more).
- Specify correct formulas for the distances and ramps of all involved markers (see also page 174). With according formulas, line-up in a block can be controlled with one single brake and one stop marker (for each direction).
- Ensure that the required markers are triggered. Line-up can be performed with one single sensor in a block. The sensor, however, which triggers the brake and stop markers used for line-up, must not be occupied by waiting trains.
- Perform line-up and move up in one direction (FIFO principle).

Additional information: the default value assumed by the software for the gap between vehicles, which are lined-up by train tracking is 10cm (4 inches). Hence if you drive a train manually to a block, where it is lined up with waiting vehicles, ensure, that the train is stopped with a distance of about 10cm (4 inches) to that vehicles.

11.3 Approved Trains

X

The use of blocks, routes, schedules, etc., as well as performing other functions can be limited to certain locomotives, cars or train sets. In this manner it can be e.g. ensured that certain schedules will be carried out with passenger trains or that electrical engines do not enter track sections without overhead line.

If you want e.g. to open blocks in the staging yard only for certain trains, then enter these trains in the properties of the corresponding blocks. As a result, only these trains will enter these blocks, while other trains are directed elsewhere.

Among others, each block, route, schedule, brake or stop marker, or certain tracks of a turntable can be associated with trains. Depending on the type of object this assignment serves different purposes. In the case of blocks, routes and schedules it controls, which train may use the block, the route or the schedule. In the case of brake or stop markers it is determined thereby, to which train the marker applies. In conjunction with turntable bridges this can be used to turn locomotives to a certain direction.

Train Descriptions in TrainController™ Silver

In **TrainController™** Silver it is possible for blocks, routes and schedules to specify those engines or train sets, that are allowed to use the object in question. This is done with train descriptions.

A train description consists of a list of engines.

For a train description it is also possible to specify, whether at least one vehicle or whether all vehicles in a train set must be included in this list. If a train consists of a single locomotive, this information plays no role. But when it comes to a train set, it can thus be determined, whether the train set may contain engines, which are not contained in the list, or whether all the engines in the train set must be covered by this list.

If, for example, all trains on the layout do not contain more than one engine, and the non-electrified branch line is only open for steam and diesel engines, then this can be covered with a train description, which contains only steam and diesel engines. For a train with an electric locomotive, but without steam and diesel engine on the other hand, the description does not apply.

It is also possible to exclude individual entries from the list of vehicles in a train description.

If one and the same (extensive) list of vehicles is to be entered for several objects, then this can be simplified with the aid of vehicle groups. A vehicle group consists of one or more engines and/or again further vehicle groups. Instead of assigning a list of individual engines to an object, it is also possible to store a list which contains vehicle groups in which these vehicles are directly or indirectly contained. An engine is included in a vehicle group when the engine is entered directly in the vehicle group, or if the vehicle group contains another vehicle group, in which in the engine is included.

Vehicle Groups in TrainController™ Gold

In **TrainController™** **Gold** there are several predefined vehicle groups. Each locomotive or train is automatically assigned to the appropriate groups according to its type.

The following predefined vehicle groups are available:

- Group of all steam engines
- Group of all diesel engines
- Group of all electric engines
- Group of all engines (= steam engines + diesel engines + electric engines)
- Group of all freight cars
- Group of all passenger cars
- Group of all cars (= freight cars + passenger cars)
- Group of all vehicles (= engines + cars)

Exclusion of Vehicles from Vehicle Groups

In **TrainController™** **Gold** it is also possible, to exclude vehicles from a group.

To form, for example, a new vehicle group of all diesel engines, which does not contain the rail busses, the predefined group of all diesel engines is added to this new group as well as all rail busses. The entries for the rail busses in this new group are additionally marked as excluded.

Of course it is also possible to exclude an entire group of vehicles from another group. In the above example it is also possible to form a group with all rail busses, to enter this group to the new group instead of the individual rail busses and to mark the entry for the group of rail busses as excluded.

Train Descriptions in TrainController™ Gold

In **TrainController™ Gold** it is not only possible for blocks, routes and schedules, but also for many other objects and functions to specify those engines, cars or train sets, that are allowed to use the object in question or for which the function in question is valid.

The following specifications are for example possible (list is not exhaustive):

- Which trains are allowed to use certain blocks and routes. This can be used to direct certain types of trains to specific tracks.
- With which trains certain schedules may be executed.
- Which trains specific markers in a block apply to. This allows for example certain actions to be carried out by selected trains or to specify different stop points in the same block for different types of trains.
- With which trains the successor schedules of certain schedules are executed.
- Which trains a waiting time in a block of a schedule applies to.
- Which locomotives are rotated to a certain direction, when they leave the bridge of a turntable via a certain track.

These specifications are not limited, however, to simple lists of vehicles. In particular when train sets are used it is also possible, to describe the characteristics and composition of the train set accurately. This is done with train descriptions.

Simple Train Descriptions

A simple train description consists of a list of engines and cars. This list may also include vehicle groups, where other engines, cars or again vehicle groups are included.

For a simple train description it is also possible to specify, whether at least one vehicle or whether all vehicles in a train set must be included in this list. If a train consists of a single locomotive, this information plays no role. But when it comes to a train set, it can thus be determined, whether the train set may contain vehicles, which are not contained in the list, or whether all the vehicles in the train set must be covered by this list.

If, for example, all train sets on the layout do not contain more than one engine, and the non-electrified branch line is only open for steam and diesel engines, then this can be covered with a simple train description, which contains only steam and diesel engines and which prescribes, that at least one of these vehicles must be included in the train set. Train sets with additional cars are then also covered by this description. For a train with an electric locomotive, but without steam and diesel engine on the other hand, the description does not apply.

If you want to create a train description for a train set with a certain diesel engine X with several cars of type Y, for example, which does not apply, when the diesel engine X pulls other cars than Y, then enter X and all cars of type Y to the train description and specify, that all vehicles of the train set must match the list.

Similar to vehicle groups, it is also possible to exclude individual entries from the list of vehicles in a simple train description.
The train description for the blocks of the non-electrified branch line can alternatively contain the group of all engines and an excluding entry for the group of all electric engines.

Train descriptions are edited in the **Trains** tab.

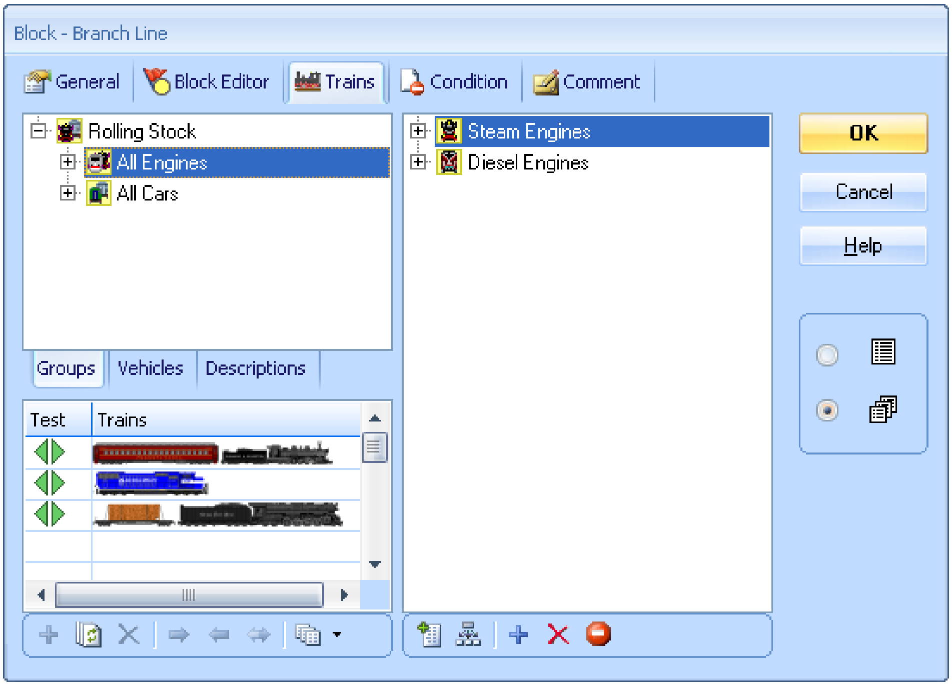


Diagram 151: Editing a simple Train Description

This tab is divided into the following areas:

- The top left area contains the inventory of available vehicles. This area itself consists of three tabs. From the **Groups** tab, vehicle groups can be selected. The groups can be accessed via a tree structure. At the lowest level of the tree the individual vehicles can also be selected from here. The tree structure allows not only the transfer of entire groups to the other areas, but also to locate individual vehicles according to their type.
  In the **Vehicles** tab, all vehicles are presented in a list.
  To enter a vehicle or group from this area in another area, select it here and click the option to add it in the target area, or drag the symbol with the mouse there.
- The area to the right contains the vehicles and vehicle groups, which are assigned to the train description.
- In the lower left area it is possible to compose virtual train sets for testing purposes. Green and red markings indicate if the train description applies to this train or not. In this way, the validity of a train description can be tested very comfortable.

Diagram 151 shows three train sets in the test area. The first train is a steam locomotive with a passenger car. The second train consists of a single diesel locomotive. The third train consists of a steam engine with a freight car. The train description prescribes that every train must have at least a steam or diesel locomotive. Since this is valid for all three train sets, everywhere green markers appear.

Train Descriptions as independent Objects

Train descriptions can also be created as independent objects, if for some reason the same train description is to be used several times. In order to assign the steam and diesel locomotives to all blocks of the branch line, a simple train description, which contains the steam and diesel locomotives, is created first as an independent object. Then in the settings of the blocks this separate train description is entered. If this description is changed at a later time, then this change automatically affects all blocks in which this train description has been used.

Train descriptions, that are created as separate objects, offer additional possibilities to specify the following characteristics of the affected trains:

• the minimum and maximum length
• the minimum and maximum weight
• the range for the current speed of trains

This allows for example to selectively enable or disable certain blocks for trains that fall below or exceed a predetermined length.

It is also conceivable to create a train description for heavy freight trains by specifying a minimum weight. By assigning this train description to an action marker, which in turn belongs to a block in a slope, heavy trains can then prototypically decelerate in this slop. This also functions adequately if the arrangement of a train set is not changed, but its weight is changed by the operations for loading and unloading.

Furthermore, specific actions can be triggered when a passing train falls below or exceeds a certain speed range.

Conditional Train Descriptions

For train descriptions, that are created as separate objects (see previous section), a condition can also be specified. If this condition is not fulfilled, no train fits the train description. So even a train, that meets all other criteria of the train description, will not fit the train description, if the condition is not satisfied.

In this way it is for example possible, to disable or enable certain blocks, routes, etc. for specific trains with a condition. Assume separate train descriptions for freight and passenger trains. Assume furthermore, that a certain toggle switch in the switchboard is contained in the conditions of the train description for freight trains. If both train descriptions are assigned to a block, then the block can be either enabled for passenger trains only or for both types of trains. This depends on the position of the toggle switch. This results in additional interesting opportunities to intervene in the ongoing operation of the layout.

Extended Train Descriptions

In the Train tab it is also possible to specify, whether the train description is a simple description, as described in the previous section, or whether it is extended train description, which offers further possibilities discussed in this section.

Extended train descriptions are e.g. used to specify references to separately created train descriptions as described above. In the example of the previous section, the separately created train description is assigned to a block by activating the extended train description in the **Train** tab of the block first and then entering the reference to the external train description.

Extended train descriptions also allow you to determine the arrangement of train sets to the last detail.

If train sets are not used on a layout or not modified during the operation, then all necessary requirements are already met by the simple train descriptions outlined above. A study of possibilities explained below is then not required.

An extended train description consists of one or several lines. Each of these lines represents a pattern for the composition of a train set. This pattern in turn consists of a string of positions, which represents the string of vehicles in the train set. Each position in a pattern is in turn associated with a list of vehicles (consisting of locomotives, cars or vehicle groups) and the number of times the specified vehicles will occur at that position.

It is also possible, to enter the reference to a separately created train description in one line of a train description instead of a pattern string. Hence a train description is a collection of patterns and references to other train descriptions.

If a train description includes more than one line, then the whole train description matches a train, if at least one row fits this train.

Extended train descriptions are also edited in the **Trains** tab. After switching to an extended description the tab has the following structure:

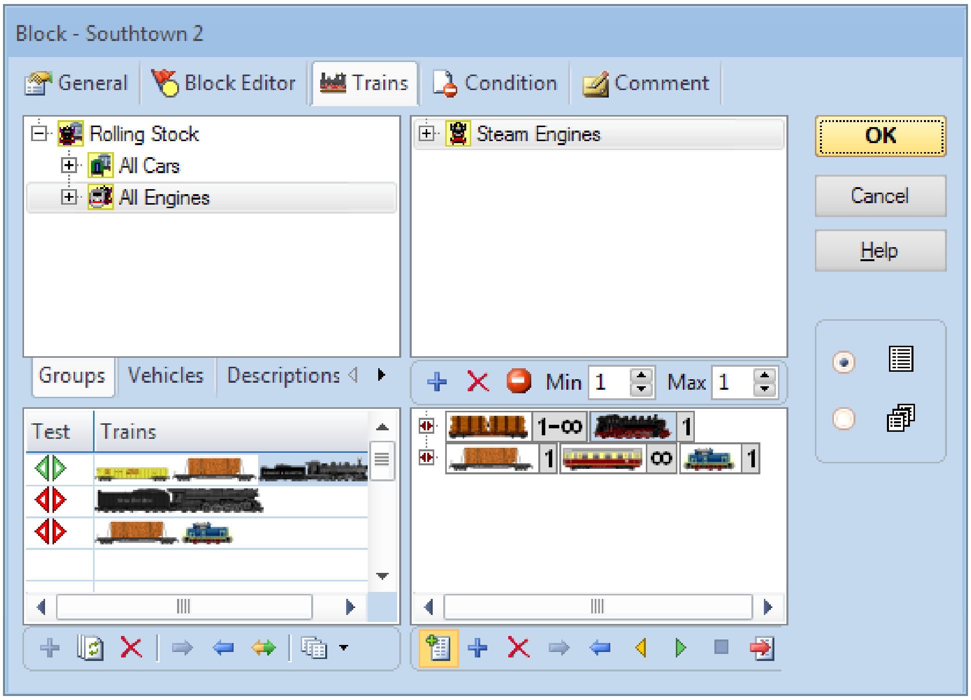


Diagram 152: Editing an extended Train Description

This tab is divided into four areas:

- The top left area contains the inventory of available vehicles. This area itself consists of three tabs. From the **Groups** tab, vehicle groups can be selected. In the **Vehicles** tab, all vehicles are presented in a list.
  The **Descriptions** tab contains all train descriptions created as separate objects.
- To enter an item from this area in another area, select it here and click the option to add it in the target area, or drag the symbol with the mouse there.
- The upper right area contains the vehicles and vehicle groups, which are assigned to one position of the extended train description.
- The lower right contains the individual lines of the train description. If a position is selected in a line of this description, then the vehicles belonging to this position are listed above.
- In the lower left area it is possible to compose virtual train sets for testing purposes. Green and red markings indicate if the train description applies to this train or not. In this way, the validity of a train description can be tested very comfortable.
  Only the pure train compositions are tested here; train lengths and weights remain disregarded.

After this survey the features will now be discussed in more detail.

The individual positions of a train are depicted by a symbol, which includes a picture of a vehicle as well as a number indication.

The train description in the above diagram contains two lines. The first line consists of two positions, the second entry of three positions. The lines are read from right to left in each case, and the rightmost position corresponds to the head of the train set as seen in the forward direction of travel.

In the above example the front, rightmost position of the first line is selected. The list of vehicles belonging to this position consists of the predefined vehicle group of all steam locomotives.

The image in a position always tries to represent the vehicles contained in the associated list. The software always tries to find a symbol for each position, which reflects the contents of this position as accurately as possible. If a predefined vehicle group is used, then one of the images displayed in the following table is used.

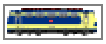
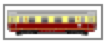
Symbol	Predefined Vehicle Group
	All Steam Engines
	All Diesel Engines
	All Electric Engines
	All Engines
	All Freight Cars
	All Passenger Cars
	All Cars
	All Vehicles

Table 6: Symbols for predefined Vehicle Groups used in Train descriptions

If individual vehicles or self-defined vehicle groups are used instead of predefined vehicle groups, then the program will show the picture of any assigned vehicle. This is e.g. the case at the rightmost position of the second line in the above diagram. Here a locomotive with the symbol has been assigned.

The numbers displayed at a position indicate how often you want the associated vehicles to appear at the corresponding position.

Symbol	Meaning
1	Exactly 1 steam engine
1-3	At least 1 and at most 3 diesel engines
5	Exactly 5 cars
2-∞	At least 2 freight cars
0-5	At most 5 passenger cars (no passenger cars possible, too)
∞	Any number of vehicles (no vehicle possible, too)

Table 7: Numerical Data in Train Descriptions

From the positions the descriptions of matching train sets are composed:

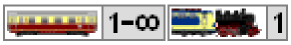
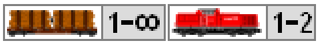
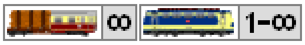
Symbol	Meaning
	Passenger train with 1 engine and at least 1 passenger car
	Freight train with 1 or 2 diesel engines
	Arbitrary train pulled by electric engines
	Freight train with double heading and helper
	Train with 2 steam engines but without cars

Table 8: Sample Train Descriptions

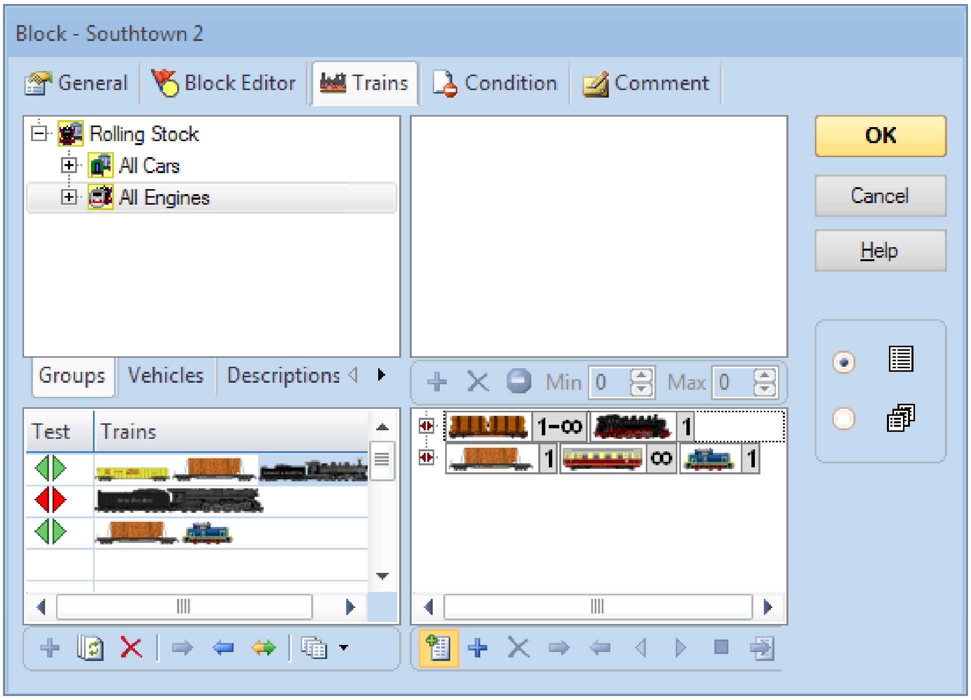


Diagram 153: Testing an extended Train Description

At the bottom left of the **Trains** tab virtual train sets are assembled to see if the created train descriptions fit as desired or not on certain trains.

A green check mark indicates, that the description fits to the train concerned, while a red marker indicates that the description does not fit. A green-white flag indicates that the currently selected line of the description fits on the train (see Diagram 152). With the help of the different green markers it can be determined, if necessary, which line in particular fits on a train.

In Diagram 153 in the test area, three train sets were arranged. The first train is a steam locomotive with two freight cars. This train fits exactly on the first line of the train description. A green marker appears.

The second train being tested is a single steam locomotive. The description always requires at least one car. Therefore, here a red mark appears.

The third train matches the second line of the train description. This line contains passenger cars in the center position, but because the minimum number of passenger cars at this position is 0 and therefore passenger cars may be missing, the entry fits on this train and a green marker appears.

Directional Train Descriptions

Very advanced users also have the following options:

- By default, all lines of a train description are not directional. This means the direction of travel of the train in question does not matter, i.e. whether the locomotive pulls or pushes the train. It does not matter whether a train matches the positions of the entry from right to left or from left to right. If you want to distinguish this as well, so there is the possibility to define the train description as directional. A directional train description considers the positions of the vehicles in a train set only in the order of the positions shown on the screen. With directional train description pulled and pushed trains can be distinguished. This direction specification is only evaluated when the train actually moves. A directional train description does not fit on a train, when the train is moving in the wrong direction. For a stopped train, it does not matter whether the train description is directional or not. It is enough if it applies to one of two possible directions.
- By default, all positions within a line of a train description are not directional. It does not matter whether the locomotive operates the train with the funnel or the tender first. If you want to distinguish this, too, then you can set each position individually as well as forwards or backwards. Using this information it is for example possible to

specify that a train description applies only to a train if it is pulled by a steam locomotive with funnel ahead. If a train set includes directional positions, then the orientation of the individual vehicles also plays a role in a stopped train.

The colored markers ◀▶ in the test window consist of arrowheads, which represent the two possible directions of the tested train. In a simple or non-directional train description both markers always show the same color, because the direction is of no importance here. With a directional train description you can use this to check whether the relation to the direction has been set correctly.

In a non-directional line of a train description the icon ◂▸ is displayed on the left side of the line. When this icon is clicked with the mouse, the order of the positions of this line is reversed. This does not change the effect of the train description, however.

In a directional line of a train description the icon ▸ is displayed on the left side of the line. When this icon is clicked with the mouse, the order of the positions of this line is reversed. This also changes the effect of the train description for running trains.

For directional train descriptions the direction of the train plays only a role when the train actually moves.
If individual positions within the train description are directional, then the orientation of the vehicles within the train set is also effective when the train stops.

Note that extended train descriptions are only needed in very specific cases. In the majority of cases it is sufficient to use simple train descriptions, which are also much easier to set up.

Train Descriptions with marked Vehicle Positions

For specific purposes it is possible to mark selected positions in a train description. These markings can be used to apply certain train operations to individual vehicles in a train set.

Examples are:

- Execution of a vehicle function (e.g. operation of interior lights only by individual cars in a train set.
- Separating a train set left or right of a certain vehicle

.

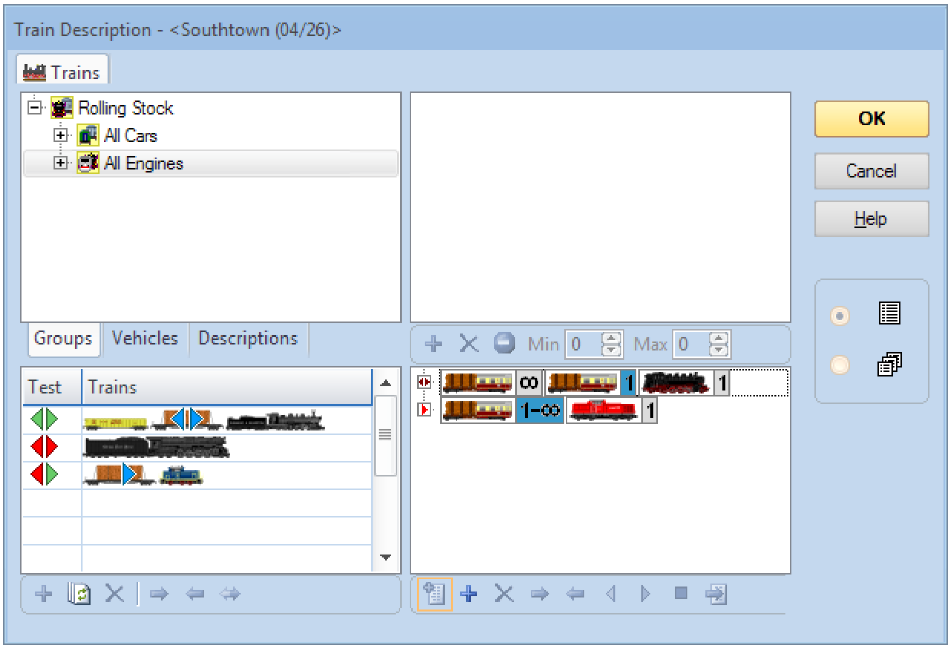


Diagram 154: Train Description with marked Vehicle Positions

Diagram 154 shows a train description, which applies to trains with one engine and one or more cars. The position of the first car is marked. This is indicated with a blue marking in the lower right area of the dialog box.

In the test window in the lower left area of the dialog box the matching cars are marked with a blue marking for each direction of travel. Note, that the matching cars may be different for the different directions of travel.

If a train description contains more than one line, then the marked positions in the first line, which matches a given train, apply to the vehicles of this train.

In Diagram 154 in the test area, three train sets were arranged. The first train is a steam locomotive with two freight cars. This train fits exactly on the first line of the train description. A green marking appears. The first car is marked with a blue arrow for both directions of travel, because this car matches the marked position in the first line for both directions of travel.

The second train being tested is a single steam locomotive. The description always requires at least one car. Therefore, here a red marking and no blue markings appears.

The third train matches the second line of the train description, but only in forward direction. Therefore the matching car is displayed with a blue marking only for the forward direction.

11.4 Acceleration and Train Tonnage

An additional feature of **TrainController™** is the realistic simulation of *momentum*, i.e. *acceleration* and *deceleration* of engines and train sets.

For each *engine* you can specify the *power* (see also Diagram 81). The power affects the acceleration of the engine. An engine with more power is able to accelerate faster. The acceleration is also affected by the *type* of the engine. Usually an electric engine is able to accelerate faster than a steam engine with identical power. This fact is also taken into account when the acceleration is calculated.

Cars and train sets provide even more realistic simulation of momentum. It is namely possible to specify the *weight* of each car. The higher the total weight of all vehicles in a train set, the longer is the time needed to accelerate the train to a certain speed or to decelerate the train. The maximum speed of a train is also limited by the total weight of the train.

If several engines are running as a *multiple unit*, then the *power* of each engine is added to the total power of the multiple unit. Since the total power is higher than the individual power of each particular engine the multiple unit is able to accelerate faster and to run with a certain train tonnage at a higher maximum speed.

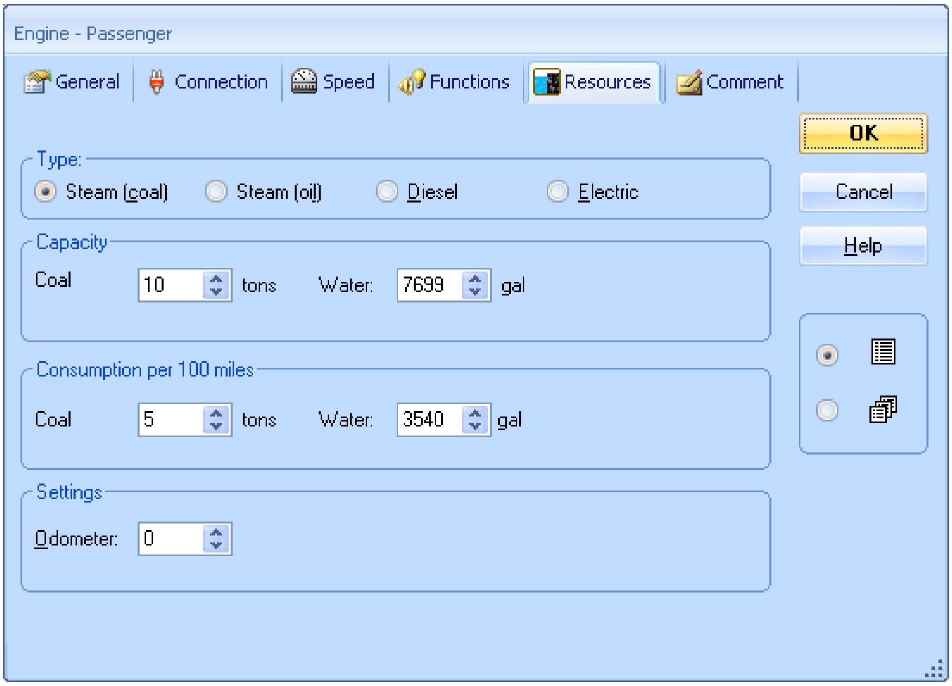
The time needed to accelerate or decelerate an engine or train set is additionally scaled and shortened using the scale factor of the *Railroad Clock*. If for example the scale factor of the clock is 10, then the calculated times are shortened to the tenth part. Even this shortening, however, results in timing, which is often found too slow. For this reason it is possible to adjust the *inertia* of each engine individually. In this way it is possible to accelerate or decelerate an engine without inertia or with the inertia of a real engine. Any setting between these extreme cases can be selected. It is also possible to adjust the inertia for acceleration and deceleration separately (see Diagram 81).

Do not be concerned if this sounds too complicated - especially in the beginning. For each engine which is created **TrainController™** assumes default settings for power, train tonnage and momentum. You are not required to set it. The default values result in a moderate behavior for acceleration and deceleration which can be adjusted with the *inertia* as desired. The additional features discussed in this section are only needed if you want to simulate the behavior of real trains.

11.5 Coal, Water and Diesel

You can specify the *type* of each *engine*. This attribute describes how the engine is powered. Possible choices are *steam engine*, *diesel engine* or *electric engine*.

Using this *engine type* **TrainController**™ calculates the consumption of *coal*, *oil*, *water* or *diesel*, if desired. It is possible to specify the capacity and the consumption per 100 miles of coal, oil, diesel or water.



Type: [✓ Steam (coal)] Steam (oil) Diesel Electric

Capacity
Coal 10 tons Water: 7699 gal

Consumption per 100 miles
Coal 5 tons Water: 3540 gal

Settings
Odometer: 0

Diagram 155: Arranging Consumption of Coal and Water

This calculation can be toggled on or off as desired. If toggled on, **TrainController**™ calculates the consumption of resources while the engine is running. By selecting specific menu items the resources can be reset to full, for example after the affected engine has visited an engine yard.

If an engine runs out of resources it is stopped. The affected resource must be reset to full before it is possible to start the engine again.

For electric engines no consumption of resources is calculated.

11.6 Monitoring the Maintenance Interval

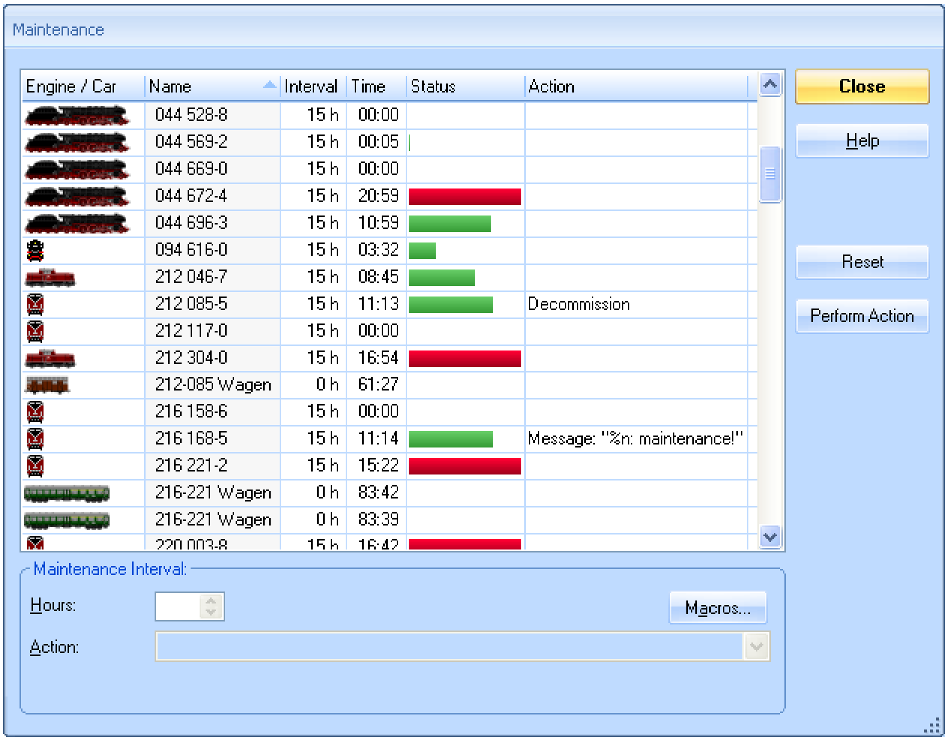
For each engine and car the elapsed operation time since the last *maintenance* is tracked by **TrainController™ Gold**. This time is increased accordingly when a vehicle is running.

Based on the recommendations of the manufacturer of your engine or car, you can determine when it is time to lubricate the wheels or to change the carbon brushes. After maintenance you can reset the elapsed time to 0 (see Diagram 156).

For each vehicle it is possible to specify an individual maintenance interval and an optional operation, that will be automatically performed, when the maintenance interval expires. The following operations are possible:

- Decommission of the vehicle.
- Display of a message in the Message Window.
- Execution of a macro.
- Execution of a schedule .

Especially the latter allows for very interesting features. It is for example possible to specify a certain schedule (*maintenance schedule*), which directs each vehicle automatically to a certain track of your layout, when the maintenance interval expires. If this feature is applied to a car, then the train set, which the car currently belongs to, is started by the maintenance schedule. The maintenance schedule should be equipped with an appropriate trial time. This ensures, that the schedule is also executed in cases, where the vehicle is currently busy in another schedule, when the maintenance interval expires. In this case the vehicle will first terminate its current regular schedules and then start the maintenance schedule.


Maintenance

Engine / Car	Name	Interval	Time	Status	Action
	044 528-8	15 h	00:00		
	044 569-2	15 h	00:05	🟩	
	044 669-0	15 h	00:00		
	044 672-4	15 h	20:59	🟥	
	044 696-3	15 h	10:59	🟩	
	094 616-0	15 h	03:32	🟩	
	212 046-7	15 h	08:45	🟩	
	212 085-5	15 h	11:13	🟩	Decommission
	212 117-0	15 h	00:00		
	212 304-0	15 h	16:54	🟥	
	212-085 Wagen	0 h	61:27		
	216 158-6	15 h	00:00		
	216 168-5	15 h	11:14	🟩	Message: "%n: maintenance!"
	216 221-2	15 h	15:22	🟥	
	216-221 Wagen	0 h	83:42		
	216-221 Wagen	0 h	83:39		
	220 003-8	15 h	16:42	🟥	

Maintenance Interval:

Hours:

Action:

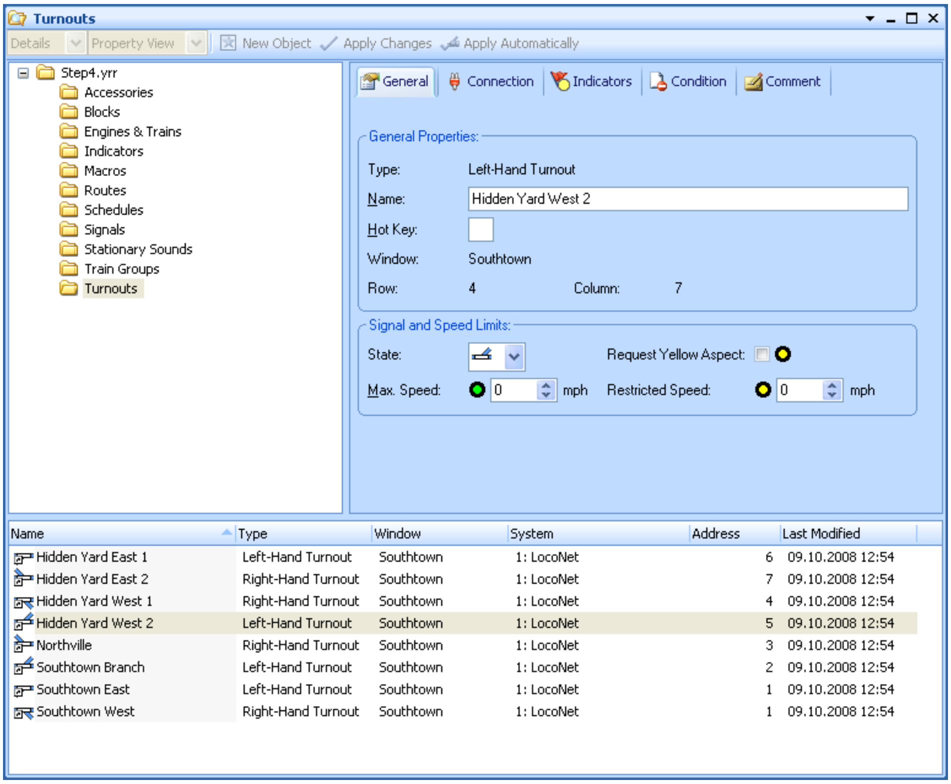
Diagram 156: Vehicle Maintenance

12 The Object Explorer

The *Object Explorer* allows effective management and editing of all objects stored in **TrainController™** in a fashion, which is similar to the Microsoft Windows File Explorer. The Explorer is especially useful for experienced users with complex layouts.

Several Explorer windows can be opened simultaneously through the **Windows** tab. The Explorer window shows three window panes:

- the folder pane in the upper left corner
- the list pane in the lower area
- the detail pane in the upper right corner



Type: Left-Hand Turnout
Name: Hidden Yard West 2
Hot Key:
Window: Southtown
Row: 4 Column: 7

State:
Request Yellow Aspect: ☐
Max. Speed: 0 mph
Restricted Speed: 0 mph

Name	Type	Window	System	Address	Last Modified
Hidden Yard East 1	Left-Hand Turnout	Southtown	1: LocoNet	6	09.10.2008 12:54
Hidden Yard East 2	Right-Hand Turnout	Southtown	1: LocoNet	7	09.10.2008 12:54
Hidden Yard West 1	Right-Hand Turnout	Southtown	1: LocoNet	4	09.10.2008 12:54
Hidden Yard West 2	Left-Hand Turnout	Southtown	1: LocoNet	5	09.10.2008 12:54
Northville	Right-Hand Turnout	Southtown	1: LocoNet	3	09.10.2008 12:54
Southtown Branch	Left-Hand Turnout	Southtown	1: LocoNet	2	09.10.2008 12:54
Southtown East	Left-Hand Turnout	Southtown	1: LocoNet	1	09.10.2008 12:54
Southtown West	Right-Hand Turnout	Southtown	1: LocoNet	1	09.10.2008 12:54

Diagram 157: Object Explorer

12.1 Folders

All objects are grouped by folders in the Explorer. These folders are similar to the file folders of the Windows Explorer. Each object is contained in exactly one folder.

By default all objects are grouped by type; i.e. there are separate folders for turnouts, routes, blocks, engines and trains, etc. These *default folders* cannot be deleted, they are the default home folder for all newly created objects of the concerning type.

It is also possible to create additional *user folders* and to arrange all objects in a customised folder structure. User folders can be created at any place in the folder structure, each object can be moved into each folder. In this way it is for example possible to create a separate folder for each switchboard, where all objects are stored, that are contained in this switchboard.

Since all newly created objects need an initial default home folder it is not possible to delete the predefined default folders. However, if you want to work with a completely own folder structure, then you can keep the default folders empty and move them to a separate auxiliary user folder of your folder structure. Note however, that all objects, that are newly created in another window, will be still initially stored in the default folder, that belongs to the type of the object. From here the object must be explicitly moved to another folder in your custom folder structure, if desired.

12.2 Objects and Links

The second pane lists all items, that are contained in a folder, and optionally their most important properties. This provides a quick overview of groups of related objects and their most important attributes; e.g. which digital address is being used by which engine or which routes have been modified most recently.

Objects are either directly stored in the Explorer or indirectly as so called *links*. A link is associated with an object, that is stored in another window. Turnouts, for example, which are always located in switchboard windows, are listed as links in the Explorer. Whenever a new turnout is created in a switchboard, then a link associated with this turnout is automatically created in the default turnout folder in the Explorer. If a turnout is deleted from a switchboard, then the associated link in the Explorer disappears, too. The same is true for all other objects hosted by other windows, such as signals, blocks, routes or schedules, etc. This ensures, that all objects stored in **TrainController™** are also visible in the Explorer.

Even though a link to the object does not represent the object itself it is possible to edit the properties of an object through its associated link. In this way it is possible to view and access with the Explorer the properties of all objects stored in **TrainController™**.

Note however, that it is not possible to create or to delete links in the Explorer. The creation of a turnout link, for example, is not possible, because it would not be clear, in which switchboard and at which position in this switchboard this turnout should be located. The same is true for all other objects located in windows other than the Explorer. Deletion of links is also not possible because such links would be gone forever and would make these objects inaccessible in the Explorer, which again would violate the rule, that all objects should be accessible in the Explorer. If you try to delete a link from the Explorer, then the link is not actually deleted; instead the link is being moved to the default folder, that belongs to the type of the associated object (e.g. the turnout folder, if the link points to a turnout). For reasons of simplification there is exactly one link in the Explorer to each object stored in another window. It is neither possible to delete this link nor to create additional links to the same object.

TrainController™ displays all links and certain references to objects listed in a window, that are actually stored in another window, with the same small marking, that is used by the Microsoft Windows File Explorer to distinguish links to files from other items.

However, the Explorer does not only store links; it is also possible, to create and store certain types of objects directly in the Explorer. Among others the following objects can be directly created and stored in the Explorer:

- Folders
- Engines, Trains and Vehicle groups
- Contact, Flagman and Virtual Indicators
- Push Buttons, On-Off Switches and Toggle Switches
- Routes (manual routes only, see page 307)
- Macros
- Sound objects (**+4DSound™**)

This allows to create objects especially for semi-automatic or automatic operation without occupying space in other windows such as the switchboards or the Dispatcher. It is, for example, possible to create routes in the Explorer for manual operation with start and destination keys, which are in turn located in a switchboard, without needing any switchboard space for the route symbol itself.

Indicators, which are being used to monitor the occupancy state of routes for reliable automatic operation, for example, and which are not contained in a block, can be created in the Explorer, too. In this way they do not occupy any space in the switchboard.

Other applications are push buttons, on-off switches or toggle switches, that are created to control the outputs of accessory decoders automatically, but that are only operated by other objects (e.g. routes or macros) rather than manually, and that will not occupy any space in a switchboard.

In general the Explorer is able to act as a container for certain objects, that will not occupy space in another window. Note, however, that unlike links the deletion of directly stored objects actually removes these objects from the **TrainController**™ data, since there is no other location, where these objects are stored.

Note also, that the Explorer serves for management and editing of objects. It is not possible to operate objects manually through the Explorer.

12.3 Object Details

The third pane shows the details of the object, that is currently selected in the list pane. There are two modes to view these details. The first view is the **Inspector View**. This view shows the details of the selected object in the same fashion as the separate Inspector window (see chapter 7, “The Inspector”). The second view is the **Properties View**, it allows to edit and view the properties of the currently selected object, as if the separate Properties window of that element were opened.

Since several Explorer windows can be opened simultaneously it is possible to view and edit the properties of different elements at the same time. This is for example useful, if comparison of the properties of different objects is an issue.

The **Properties View** is also available, when edit mode (see page 88) is turned off. This allows to view the properties of each object even during operation of the layout, however it is not possible to edit the data, while edit mode is turned off.

13 The Clock

TrainController™ can display a fast clock on your computer screen. Using a fast clock time spaces are stretched artificially. This simulates more realistic timing.

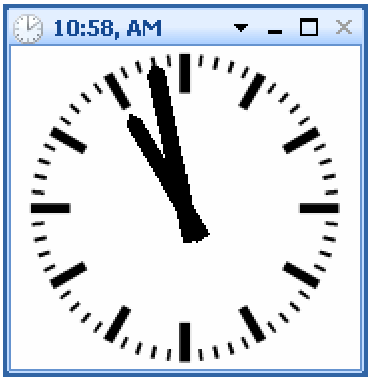


Diagram 158: The Clock Window

The clock is used to perform a timetable based operation with the *Dispatcher* (see chapter 15, “ The Visual Dispatcher II”). It is also used for realistic simulation of inertia when a train is accelerated or decelerated. *Simulated distances* are calculated using the clock, too.

Additionally the clock provides a perpetual calendar, with which an arbitrary date between 1830 and 2030 can be selected. In this way it is possible to play in your favorite epoch and to run different timetables, for example varying between working days or holidays.

The clock is permanently active and runs always in the background of the program. If desired you can display the clock on the computer screen. If the clock is visible it can be stopped, if desired, or its settings - such as *scale factor, current time* or *date* - can be changed.

An additional useful feature is skipping time intervals without operation. If you run a timetable in which no trains are running at night then you can skip such a period. In this way you can shorten those intervals as desired.

14 Extended Control and Monitoring Functions

With the mechanisms outlined in this chapter you can extend manual control to semi-automatic control of your layout with switchboards. Additionally some of the mechanisms explained here can be applied to the *Visual Dispatcher* or can be used to influence automatic control individually.

This is the reason why they are discussed in a separate chapter.

14.1 Indicator Symbols in Switchboards

Indicator symbols cannot only be created in the block editor (see section 5.6, “Blocks and Indicators”), it is also possible for specific purposes, to create indicator symbols in a switchboard or in the Explorer (see chapter 12, “The Object Explorer”). Locating an indicator symbol in a switchboard is especially useful in cases, where the *Visual Dispatcher* is not being used at all or if the switchboard represents an area of your layout, which is not controlled by the *Visual Dispatcher*. Locating an indicator symbol in the Explorer is useful for auxiliary indicators, which should not be visible in a switchboard.

14.2 The Memory of Indicators

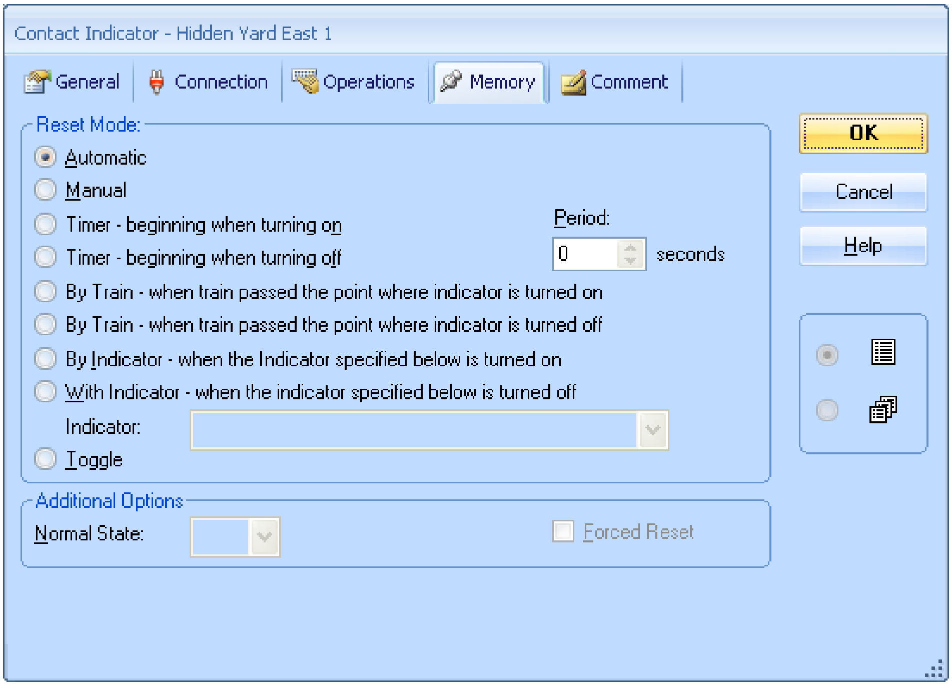
In the simplest case, an *indicator* is automatically turned on and off by the associated track contact or occupancy sensor (see chapter 4, “Contact Indicators”). Additionally, *indicators* provide a *memory* in which the event, that has occurred, can be “stored” for a longer period. This possibility is especially useful to prevent an indicator from unwanted flickering during automatic operation (see below).

For this purpose you can select one of the following methods to turn off the *indicator*:

- **Automatic:** this is the default method. In this case the *indicator* is automatically turned on and off by the associated track contact or occupancy sensor.
- **Manual:** in this case, the *indicator* remains turned on until you turn it off – by clicking on it with the mouse.
- **Timer:** in this case, the *indicator* remains turned on for a certain period. This can be used to reset a signal a few seconds after a train has passed it.
- **By Train:** if this option is selected, then the *indicator* remains turned on until a train has passed the associated track contact or occupancy sensor or another point on your

layout. With this option it is for example possible to use a momentary track contact for *Virtual Occupancy Indication*.

- **By Indicator:** if this option is selected, then the indicator remains turned on until another indicator is turned on.
- **With Indicator:** if this option is selected, then the indicator remains turned on until another indicator is turned off. Previous software versions only provided the option of leaving the indicator turned on, until another indicator is turned on.
- **Toggle:** if this option is selected, then the indicator is alternately turned on and off. With this option, it is possible to create a track occupancy detector with two momentary track contacts. This is explained in more detail in the example on page 301 "Simple Track Occupancy Detection".


Contact Indicator - Hidden Yard East 1

General | Connection | Operations | Memory | Comment

Reset Mode:
- (●) Automatic
- ( ) Manual
- ( ) Timer - beginning when turning on
- ( ) Timer - beginning when turning off
- ( ) By Train - when train passed the point where indicator is turned on
- ( ) By Train - when train passed the point where indicator is turned off
- ( ) By Indicator - when the Indicator specified below is turned on
- ( ) With Indicator - when the indicator specified below is turned off
- Indicator:
- ( ) Toggle

Period: 0 seconds

Additional Options
Normal State:
☐ Forced Reset

Diagram 159: Memory of a an Indicator

Normally, the *indicator* is turned off when the condition that causes the activation of the indicator no longer applies, e.g. if **Timer** is selected and the specified time period has passed, then the *indicator* is turned off only if the condition no longer applies. If the *condition* is still valid, then the *indicator* remains turned on even if the specified time period has passed. Sometimes it is useful to turn off the *indicator* regardless of the current state of the *condition*. For this purpose the additional option **Forced Reset** can be selected.

Note, that state of an indicator stored by **Memory** settings is only valid to the end of the current session. If **TrainController™** is terminated and started again, then the indicator may be set to another state after begin of the next session.

Example: Preventing an Indicator from Flickering

In the following example it is assumed that a certain momentary track contact is triggered by each axle of a passing train. It shows how the indicator symbol can be prevented from flickering. Finally, the indicator will be turned on only once by a passing train.

- Create a *contact indicator* and link it to the momentary track contact.
- Set the *memory* of the indicator to **Timer 2 Seconds**.

	Memory
Indicator	Reset: after 2 seconds

Table 9: Preventing an Indicator from Flickering

When the first axle of a passing train touches the track contact, then the indicator is turned on. When this axle leaves the track contact, the indicator remains turned on until the 2 seconds have passed. If the next axle of the train touches the track contact before the timer expires then the indicator will remain turned on for another 2 seconds and so on. The indicator is turned off when no further axle of the train touches the track contact, i.e. when the train has passed the contact completely. In the software the indicator is turned on only once regardless of how many cars and axles the train contains.

!

Preventing contacts from flickering is especially recommended when feedback indicators are being used for automatic control of trains. **Each indicator symbol, that is passed by a train under automatic control of the computer, should be turned on only once by the passing train.** Indicators, that are turned on two or more times by the same passing train (“flickering”) may mislead the software and can cause unexpected behavior of the affected trains.

14.3 Protection and Locking with Conditions

In addition to the locking mechanisms provided by *routes*, there are even more possibilities for locking and protection. It is possible to restrict the operation of *turnouts, signals, accessories* and *routes* to certain conditions called *conditions*. For instance, it is possible to specify that a certain turnout may be operated only if a certain dependent signal is red. Even more complex conditions, which depend on the combination of several objects, can be specified. For instance, it is possible to specify that a certain signal may be turned to green only if the turnout behind the signal is closed <u>and</u> the track section behind the turnout is unoccupied.

Such conditions are specified by assigning a *condition* to the respective element. This is done by selecting the symbol of the element and using the **Properties** command of the **Edit** tab. In the following dialog select the tab labelled **Conditions**. Now select the state that will be affected by the condition – in the second example mentioned above, the state *green* of the signal - as well as the elements that will be checked to verify whether the condition applies or not. Also in the example, you would have to select the *turnout* that will be closed and an appropriate *contact indicator* that indicates, whether the track section behind the turnout is occupied.

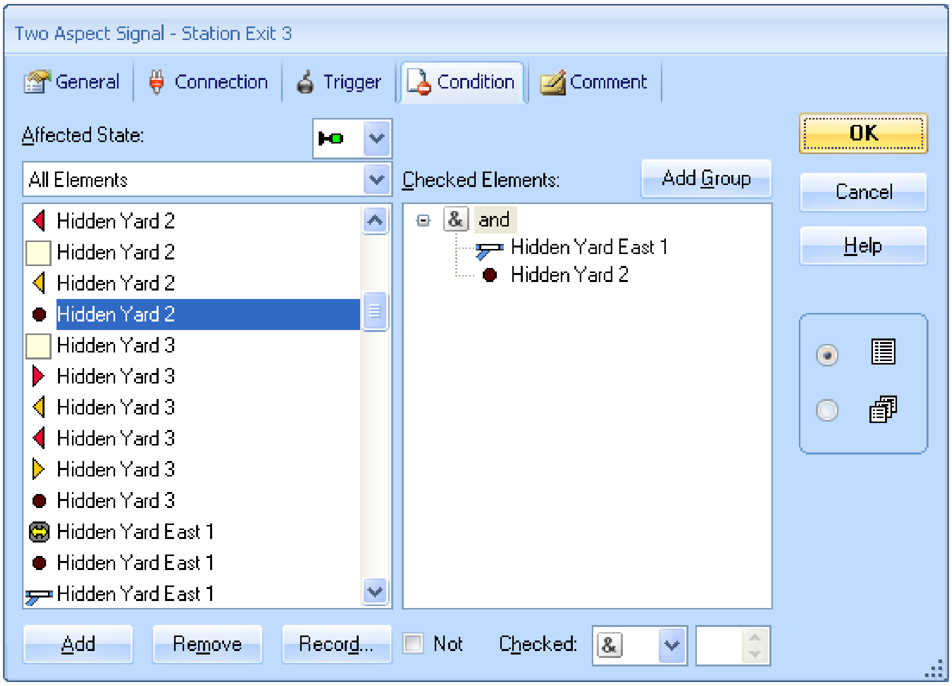


Diagram 160: Conditions of a signal

By selecting **and** or **or** in the first entry of the condition it is possible to adjust the condition to your special needs. If **and** is selected, then all listed elements must have the required state to meet the condition. If **or** is selected, then the condition applies if at least one of the listed elements has the required state.

In the example displayed above, it is possible to turn the signal to green only if the turnout "Hidden Yard East 1" is closed and the contact indicator "Hidden Yard" is turned off.

The elements that are part of the *condition* and the element that is to be restricted, may be located at arbitrary locations of your model railroad. It should be noted that it is not necessary that the elements are placed in the same switchboard window.

Complex Conditions

It is also possible to create complex conditions by mixing 'and' and 'or'. This is done by including AND-groups or OR-groups into a condition. Such groups can also contain other AND- or OR-groups, respectively. In this way it is possible to create conditions of virtually unlimited complexity. If a group is contained in another group, then the inner (contained) group is checked first, the result of this calculation is then taken into account for calculation of the containing (outer) group. And so on, if the outer group is again contained in another group.

Each condition establishes itself an AND- or OR-group. Diagram 160 shows a condition, which is an AND-group. The simple condition in this example does not contain any other groups.

It is also possible to inverse the meaning of each checked state, of a group or of the whole condition with the NOT-option. If this is done, then the appropriate item of the condition is fulfilled, when the related object is not in the specified state or, if NOT is applied to a group or the condition itself, if the result calculated for this group or condition is ‘wrong’. Among others the NOT-option is interesting for objects with more than two states. An example is a condition, that will be fulfilled, if a three way turnout is set to one of the two diverging states. Instead of assigning the two diverging states of the turnout to the condition, it is also possible to use the third state (main leg) and apply the NOT-option to it („if not set to main leg“).

Numerical Groups

In addition to the AND- and OR-groups provided by other **TrainController™** versions **TrainController™ Gold** provides three additional types of groups:

- AT-LEAST-group: such group meets the condition, if at least a certain preset number of items contained in this group have the required state.
- AT-MOST-group: such group meets the condition, if at most a preset specified number of items contained in this group have the required state.
- EXACT-group: such group meets the condition, if exactly a certain preset number of items contained in this group have the required state.

These groups can be used to evaluate, if the number of items, which are in a required state, exceeds, falls below or fits a certain preset number. This option is useful, for example, to start a certain schedule, when at least three trains are waiting in a station, or to prevent trains from running to a hidden yard, if at least 5 trains are already stored there, etc.

Combined Groups

Using numerical or logical groups in conditions or triggers it can e.g. be checked whether any train is currently in a block, but it is not possible to determine which train it is. This is made possible in **TrainController™** **Gold** with COMBI groups.

A COMBI group consists of a list of blocks, routes or schedules and is always associated with a train description (see section 11.3, “Approved Trains”)).

With COMBI groups it can be checked whether certain trains are in certain blocks, whether certain sections (blocks or routes) are reserved for those trains and / or whether certain schedules are executed with these trains. It can also be determined whether certain sections are being used by certain schedules.

- A COMBI group satisfies the condition, when a train for which the train description of the COMBI group applies, reserves at least one of the specified sections (blocks or routes), and when this train is controlled by one of the specified schedules. A COMBI group applies to a train when the train description, that is associated with the COMBI group, applies on the train.
- If no block and route is specified, then the COMBI group meets the condition if a train to which the COMBI-group applies, is controlled by one of the specified schedules.
- If no schedule is entered, then the COMBI group meets the condition if a train to which the COMBI-group applies, reserves at least one of the specified sections (blocks or routes).
- If no train description was given, then the COMBI group applies to all trains. In this case the COMBI group meets the condition, if at least one of the specified sections will be used by at least one of the specified schedules.
- For a block it is additionally possible to specify whether the block needs to be the current block of the corresponding train or whether it is sufficient when the block is reserved, but not the current block.

This may sound more complicated than it actually is. The following examples help you to understand:

Examples:

- A COMBI group, that is valid for freight trains and which contains the block “Mainline East” and the schedule “Local Freight”, meets the condition, if a freight train is located in block “Mainline East” <u>and</u> if this train is currently executing the schedule “Local Freight”.

- A COMBI group, that is valid for the engine “Big Boy” and which contains the block “Northville Branch” as reserved block, meets the condition, if the “Big Boy” reserves the block “Northville Branch”.
- A COMBI group, that is valid for the car “Passenger Car” and the schedule “Rheingold”, meets the condition, if “Passenger Car” is currently executing the schedule “Rheingold”.
- A COMBI group, that contains the route “Southtown Branch” and the schedule “Southtown - Northville”, meets the condition, if the route is reserved by a train, which is currently executing the schedule “Southtown - Northville”.

The following features and limitations apply to COMBI-groups:

- COMBI-groups can be included in other groups (such as AND-groups or OR-groups). COMBI-groups are the only type of groups, which are associated with train descriptions. In addition to the associated train description COMBI-groups may only contain blocks, routes or schedules. Other entries, including other logical groups, contained in COMBI-groups are ignored.

System Events and States

With **TrainController™ Gold** it is possible to evaluate the occurrence of certain global events or states in conditions or triggers (e.g. of flagman or signal objects).

A system event or state occurs globally and is not bound to the status of an individual object.

System events signal the occurrence of a certain system wide situation. They are visible for evaluation only for a short moment and can be evaluated only in triggers of objects.

System states indicate, that the system is currently in a certain state. System states cannot only be evaluated in triggers, but also in the conditions of objects.

Among others the following system events and states can be evaluated:

- **Begin of Session (Event):**
  This event occurs at the beginning of a session, after the project file has been completely loaded and everything has been completely initialized. By using this event in a trigger of a flagman, for example, it is possible to execute certain operations automatically at the beginning of each session.
- **Continue after Stop or Freeze (Event):**
  This event occurs, when operation is continued after freeze or stop.

- **Outside Edit Mode (State):**
  This state indicates, whether the software is running outside of the edit mode.
- **Offline Mode (State):**
  This state indicates, whether the software is running in offline mode or not.
- **Open Connection to Digital Systems (Event):**
  This event occurs, when the connections to the attached digital systems have been opened.
- **Loss of Connection to a Digital System (Event):**
  This event occurs, when the connection to one of the attached digital systems got lost.
- **Clock Running (State):**
  This state indicates, whether the clock is currently running or not.
- **Active Schedules (State):**
  This state indicates, whether at least one schedule is currently running or not.
- **Schedule Aberration (Event):**
  This event occurs, when the limited aberration protection detects an aberration in an active schedule.
- **Schedule lost Car (Event):**
  This event occurs, when an apparent lost car has been detected during an active schedule.
- **Schedule Watchdog (Event):**
  This event occurs, when the schedule watchdog detects a train under control of a schedule, that apparently got stuck.
- **Turnout Position Control (Event):**
  This event occurs, when turnout position control of a turnout used in an active route detects an erroneous turnout position.

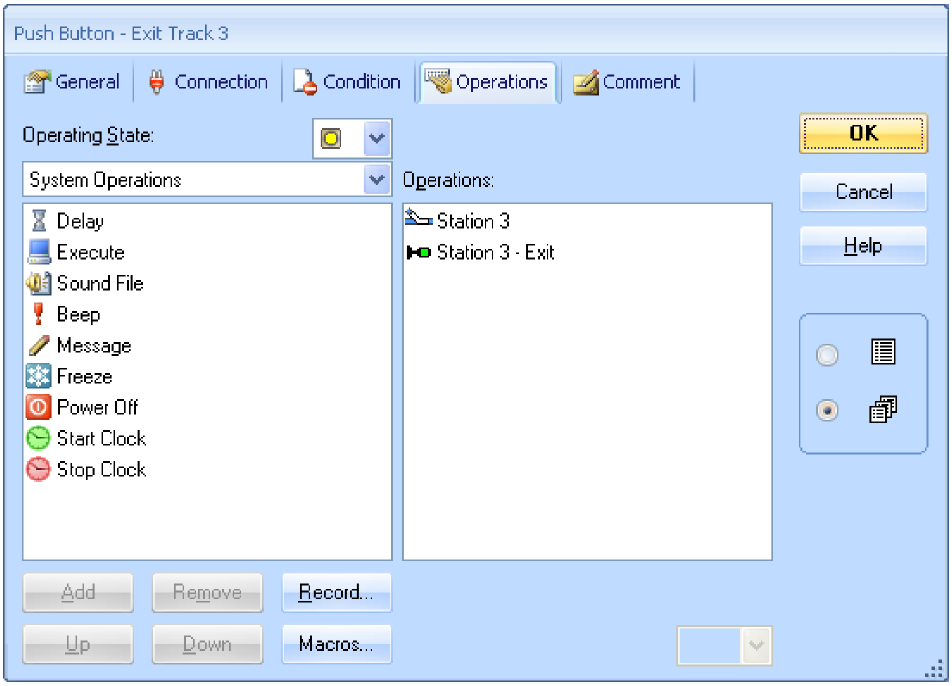
! **For security reasons the software asks the user for the permission to evaluate system events or states in triggers, when a project file is being loaded, which contains triggers with system events or states. You should grant this permission only, if you are loading only files from yourself or from people you trust.**

14.4 Operations

X It is possible to assign several *operations* to a *push button* or *on-off switch* instead of a digital address. By doing this, you are able to operate several elements with one single push button or on-off switch. It is, for example, possible to change the state of several signals, simultaneously, with one single on-off switch.

Each push button or on-off switch provides two sets of *operations* – one set for each state (on/off) of the push button or on-off switch. In this way, you can turn a group of related signals to green by turning on a certain on-off switch. The signals can be turned to red again by turning off this switch.

Operations are specified by selecting the symbol of the push button, or on-off switch, in the switchboard and using the **Properties** command of the **Edit** tab. In the following dialog select the tab labeled **Operations**. Now select the state that will trigger the operation – e.g. the state "on" of an on-off switch - as well as the elements that will be operated.


Push Button - Exit Track 3

General | Connection | Condition | Operations | Comment

Operating State:

System Operations

Operations:

- Delay
- Execute
- Sound File
- Beep
- Message
- Freeze
- Power Off
- Start Clock
- Stop Clock

Operations:
- Station 3
- Station 3 - Exit

Diagram 161: Operations of a push button

In the example displayed above, a turnout and a related signal are operated by pressing a push button.

Operations not only can be executed by push buttons and on-off switches, but also by other elements such as *indicators*, *routes* or during execution of *schedules*.

If *operations* are assigned to *contact indicators*, then passing trains are able to trigger other operations automatically. For example, a passing train can open or close certain crossing gates automatically. Another possibility is playing sound files triggered by passing trains. Since operations can also contain features of the *Dispatcher* – e.g. starting of a schedule (see chapter 5, “The Visual Dispatcher”) – virtually unlimited possibilities for automatic operation are provided.

A special application of *operations* is performed by *routes*. *Turnouts*, *signals* and other elements that are operated by routes can be locked until the corresponding route is released again. As long as they are locked with the route the affected objects cannot change their state or be used in other routes. Locked objects are thus also treated as turnouts, which lie on the path of the route.

TrainController™ **Gold** provides the additional option to set other objects via operations of a route in a protective position, which yet allows the use of the objects in other routes. The safety position remains active until the route will be deactivated. Objects cannot change their state, but they can be used in other routes, provided that the other routes use the objects in the appropriate state.

System Operations

An additional feature is *system operations*.

With these system operations, you are able to create, for example, an emergency stop button in your switchboard.

Among others the following system operations are available:

- Playing of sound files
- Execution of an external program
- Output of a warning tone with the speaker of the computer
- Turning off the power of the digital system
- Emergency stop of all trains
- Display of a message in the message window.
- Display of a message as a temporary appearing tooltip.
- Stop all trains with adjustable delay (smooth emergency stop).
- Locking and unlocking of all blocks.
- Locking and unlocking of all schedules.
- Start and stop of the clock
- Set the clock to a particular time.

- Operation of locomotive functions on all vehicles:
  With this operation, an arbitrary function of the engine functions library can be switched on or off on all vehicles in which this function is specified. In this way it is e.g. possible, to switch the interior lights in all passenger cars with a single button press. In conjunction with the timetable, this can also be done automatically at specific times.
- Select an object.
- Execute a menu command:
  This operation can trigger automatically any of the menu commands. By a preceding operation the object can also be selected, which is the subject of the following menu command. Thus, in automatic mode also those functions of the software can be performed, which are only accessible via the menu. Note, however, that selecting the object, or the execution of the menu command run exactly as if both are called via user interface (including the change of windows to the selected object, call of subsequent dialog boxes, etc.). But these side effects can also be utilized to, for example, change from one window to another by selecting an object.

Several system operations allow to use a variable or formula (see section 14.14, “Variables”) as value. More information can be found in the **TrainController™ Gold** Help menu.

Control Flow Operations

With control flow operations it is possible to affect the execution of operations.

TrainController™ provides the following operations to control the flow of operations:

- **Delay:**
  With this operation it is possible to insert a delay with a fixed duration between two operations.
- **Random Delay:**
  This operation is similar to the delay. The delay caused by this operation, however, is randomly varied between 0 and the specified time.
  To accomplish a random delay with a minimum duration (e.g., 5 to 7 seconds), specify a fixed delay of 5 seconds followed by a random delay of 2 seconds.
- **Goto and Label:**
  These two operations are always used together. With **Goto** the operation can be continued with another operation rather than the operation next in the list.
  This other operation is marked by an operation, which places a label in the list of operations. It is possible to skip forward, i.e. to a label, which is placed in the list below the **Goto** operation. And it is also possible to skip backward, i.e. to a label, which is

located above of the **Goto**. In the second case a loop is formed. A label is identified by a name with up to 4 characters. Uppercase and lowercase letters do not matter.

- **Prerequisite**:
  This operation is always associated with a condition. If the condition is valid, then that operation is executed, which follows directly after the prerequisite. If the condition is not valid at the time of execution of the operation, the next operation will be ignored and the subsequent operation will be executed.
  If the operation immediately following a prerequisite is a **Goto**, then the execution of blocks of operations can be made dependent on the prerequisite.
- **Probability:**
  With this operation, the probability of execution of the operations specified below can be influenced. Normally, all operations specified in a list are carried out with 100% certainty. With this operation, you can influence the likelihood of subsequent operations. These operations will then be executed only with the specified probability. The probability affects all subsequent operations in the list until a new probability operation is specified. The probability acts individually. That is, depending on the specified probability some of the subsequent operations are performed and some not. If an operation is specified with 100% probability, all subsequent operations are carried out as usual. In this way it is also possible to specify different probabilities to the operations in a list.
- **Random Order / Sequential Order:**
  With these operations, the order of execution of the following operations can be influenced. Normally, all operations are performed in the listed order. The **Random Order** operation allows to affect the execution of the following operations in a way that a random order is chosen. The order of subsequent operations will be mixed together until a new **Sequential Order** operation is applied.
  Operations for inserting delays or setting probabilities are unaffected by the change in the order. They maintain their position in the list. So, if for example three schedules start with an interval of one minute, and the second schedule will run with 50% probability, then with this operation the order of the schedules can be randomly shuffled. The delays between the schedules remain intact, as well as the fact that the schedule, which is started as the second, will only run with a probability of 50%.
- **Access to Variable:**
  This operation can be used to access a variable (see section 14.14, “Variables”).

Note, that for certain technical reasons the only control flow operation, which can be performed by routes, is the **Delay** operation.

Some control flow operations allow to use a formula as value. More information can be found in the **TrainController™** Gold Help menu.

Train Operations

Train operations can be applied to trains. They are often executed by indicators, markers or schedules. They can also be executed by macros, which again are executed by schedules or as a train function.

Among others the following train operations are available:

- Execution of a train function

  In **TrainController™** **Gold** this train operation allows to specify, which vehicle performs the operation, if this operation is called for a train set.

  First Vehicle:
  The auxiliary function of the first vehicle in the train set is operated.

  Last Vehicle:
  The auxiliary function of the last vehicle in the train set is operated.

  All Vehicles:
  The auxiliary function of all vehicles in the train set is operated.

  Forwarding:
  The vehicles are determined by the state of the function forwarding.

  Description:
  The vehicles are determined by a train description with marked vehicle positions (see page 269).

- Stop a train with or without momentum.
- Set the train direction.
- Start a spontaneous run with a train
- Termination of the schedule, which is currently executed by the train
- Set a temporary speed limit.
- Turn on or off function forwarding.
- Join or separate train sets.
- Automatic load or unload of cars.
- Starting an AutoTrain run from the current block of the train to any destination block. For this operation, the destination block and the direction of the entrance to this block is specified. When executing the train operation (e.g. by a button or a indicator) an AutoTrain run from the current location of the corresponding train to the

specified destination is started.
In this way it is for example possible to specify an individual home block for each train. By executing the train operation (e.g. at the end of a session) each train can then automatically be moved into his home block.
Another application is to run a train on special occasions to a particular block (e.g. for maintenance).

- Execution of a particular schedule.
  With this train operation it is for example possible to start a schedule with a specific train out of the train window.
  This operation also allows to start a certain schedule after termination of another schedule with the same train as before. With the ability to integrate these train-operation in the control flow of operations, there are further possibilities that go far beyond the existing possibilities of the successor function of schedules.
- **Display Name:**
  This train operation sets a display name for the corresponding train. This name is used instead of the stored name of the train for display on the computer screen. This operation allows to display variable train names, which depend on schedules, for example, or other operational aspects. This train operation can only be used for trains under control of a schedule (e.g. at the beginning of a schedule). When the train terminates its current schedule the display name is deleted.
- **Short distance move:**
  Performs a move of the train at slow speed by the specified distance to the specified direction. Unlike all other train operations, which set directly the speed or direction of the train, this operation is also allowed for trains under control of a schedule. However, a train under control of a schedule must perform a scheduled wait, when this operation is initiated and the short distance move must be completed, before the scheduled wait ends. Furthermore a train under control of a schedule must not leave its current block due to execution of this operation.
  This train operation can be used for coupling maneuvers during active schedules.

In **TrainController™ Gold** train operations can be performed by buttons or switches in the switchboard. For this purpose it is possible to associate any buttons or switches to a block. The train operation is then executed by that train, which is currently in the block.

In this way it is e.g. possible to set up a push button in a switchboard, with which the light or the coupling can be operated of that locomotive, which is currently in the associated block.

If you combine the ability to assign a button to a block with the train operation to start a schedule, then another interesting possibility occurs: When the button is pressed , then the schedule will be started with the train, which is currently located in the block. In this way it is possible to create a button, which does not only start the schedule, but which also selects the start block of this schedule.

! Certain train operations are not executed when the train in question is under the control of an active schedule. The time of execution plays a role, but not from where the train operation is invoked. Thus, it is e.g. possible to call a train operation to execute another schedule from an active schedule, if appropriate delays ensure that this train operation is actually executed after the current active schedule ended. The following train operations are affected (not exhaustive):

- Train operations, which set directly the speed or direction of the train
- Train operations for execution of another schedule (or AutoTrain run or spontaneous run)

Several train operations allow to use a variable or formula (see section 14.14, “Variables”) as value. More information can be found in the **TrainController™ Gold** Help menu.

Lists of Operations

Lists of operations can be used at certain parts of the software to execute a sequence of actions or to execute very specific operations, which are not available in the default selection or operations provided there. Among others lists of operations can be assigned as engine functions (see section 3.6, “Headlights, Steam and Whistle”) or they can be executed as part of schedules (see page 199).

Example: Automatic Reset of Signals

The following example explains how a signal can be reset to red after a train has left an occupancy section.

- Place or select a signal in the control panel.
- Create a *contact indicator* and link it to the occupancy section.
- Specify the signal in the “red” state as an *operation* of the indicator. This is performed when the indicator is toggled off.

	Operations	
Indicator	● **On**	-
	● **Off**	■ **Signal**

Table 10: Automatic Reset of Signals

When the train reaches the occupancy section, then the indicator is turned on. When the train leaves the section, the indicator is turned off. This also resets the signal. This is performed by the *operations* of the indicator.

Example: Emergency Stop Button

The following example explains how a push button symbol can be used to perform an emergency stop of the model railroad layout. It is also shown how the emergency stop can be triggered by pushing a certain key (here ‘S’) on the computer keyboard.

- Place or select a push button symbol in the control panel.
- Assign ‘S’ as a hot key to the push button (see section 2.6).
- Specify the system operation “Power Off” as an *operation* of the push button. This is performed when the push button is pressed.

	Hot Key	Operations	
Push Button	**‘S’**	🟥 **On**	✋ **Power Off (System Operation)**
		■ **Off**	-

Table 11: Emergency Stop Button

When the push button is pressed, either by clicking on it with the mouse or by pressing the ‘S’ on the computer keyboard, then the complete model railroad is stopped.

14.5 Semi-Automatic Control Mechanisms using Flagman Elements

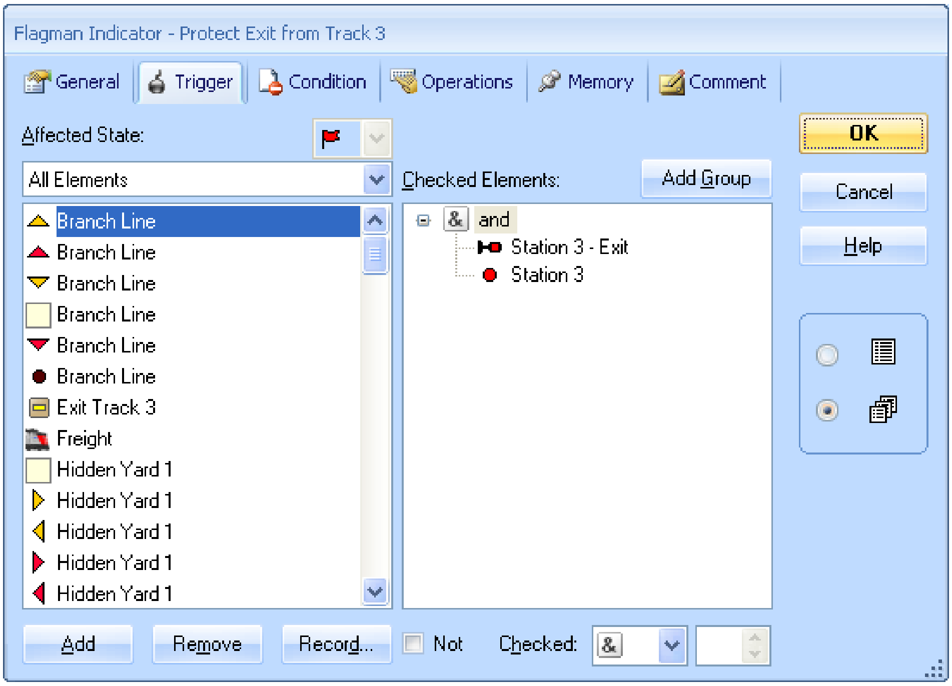
The Flagman

X

With the possibilities described in the previous sections, it is already possible to create many and diverse semi-automatic control mechanisms. Even more powerful functions are provided by the *flagman indicators* introduced in this section. This will be made clearer by the examples provided in this section. *Flagman indicators* work like intelligent relays that are turned on under certain conditions. *Flagmen* are able to indicate certain events and to execute *operations* automatically.

Flagman indicators are somewhat similar to *contact indicators*. While a contact indicator indicates if a certain feedback sensor is activated or not, a *flagman* indicates that a certain more complex event has occurred. A *flagman* is able, for example, to indicate that a train is waiting in front of a red signal. The event to monitor this is assigned to each *flagman* as a *trigger*. A *trigger* contains a set of elements whose states are to be monitored. In the example mentioned above, the *trigger* would contain the red signal and a *contact indicator* that monitors the track section in front of the signal. When the signal is red and a train touches the contact indicator, then the flagman is turned on by its *trigger*.

A *trigger* is specified by selecting the symbol of the flagman and using the **Properties** command of the **Edit** tab. In the following dialog select the tab labeled **Trigger**. Now select the elements that will be monitored.


Flagman Indicator - Protect Exit from Track 3

Affected State:

All Elements

Branch Line
Branch Line
Branch Line
Branch Line
Branch Line
Branch Line
Exit Track 3
Freight
Hidden Yard 1
Hidden Yard 1
Hidden Yard 1
Hidden Yard 1
Hidden Yard 1

Checked Elements:

and
Station 3 - Exit
Station 3

Diagram 162: Trigger of a Flagman

By selecting **and** or **or** in the first entry of the trigger additional possibilities are available. If **and** is selected, then <u>all</u> elements listed in the *trigger* must be in the required state to turn on the *flagman*. If **or** is selected, then the *flagman* is turned on, if at least one of the elements has the required state.

In the example displayed above, the *flagman* is turned on if a signal is red and a track section is occupied.

It is also possible to create complex triggers by mixing ‘and’ or ‘or’. This is done by means of AND and OR groups. These groups are described in detail on page 284. They work in the same way in triggers as they do in conditions.

It is furthermore possible to include a *flagman* in the trigger of other *flagmen*. This function provides the ability to specify *trigger* conditions with virtually unlimited complexity.

In **TrainController™ Gold** it is also possible to use numerical groups or COMBI-groups in triggers.

Flagmen and Operations

It is possible to assign a set of *operations* to each state (on/off) of a *flagman* (see section 14.4, “Operations”). In this way it is possible to operate a set of elements automatically when a certain event occurs. This feature enables flexible semi-automation of your switchboards.

Flagmen and Conditions

It is also possible to assign a *condition* to a *flagman* (see section 14.3, “Protection and Locking with Conditions”). The *condition* is additionally checked each time, after the *trigger* of the flagman has been activated and before the *flagman* is turned on. If the condition does not apply the *flagman* remains turned off. This is also the same for the triggers and conditions of signals (see section 14.7).

The following example demonstrates an application of this feature.

Example: Detecting Train Direction

The *condition* of a *flagman* can be used to detect the direction of a passing train.

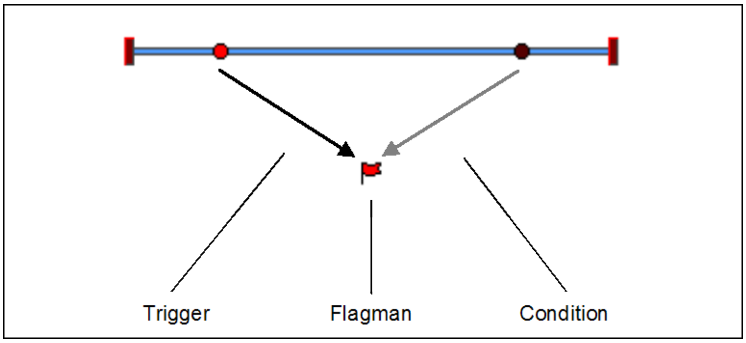


Diagram 163: Detecting Train Direction

On the track section displayed above, an operation should be performed by trains running from the left to the right. If a train passes from the right to the left, then nothing should happen. For this purpose, a detection mechanism that is activated by trains running from the left to the right only is needed.

In order to create this detection mechanism, two track sensors are placed on the track section. The distance between these sensors should be smaller than the length of the shortest train passing this section.

The following steps should be executed:

- Create a switchboard and draw the track diagram displayed above.
- Place two *contact indicators* (see section 4, “Contact Indicators”) in the track diagram and specify the digital addresses of the respective track sensors.
- Create a *flagman*.
- Specify the left contact in the “on” state as the *trigger* of the flagman.
- Specify the right contact in the “off” state as the *condition* of the flagman.

	Trigger	**Condition**
Flagman	● Left Contact	● Right Contact

Table 12: Detecting Train Direction

If the left sensor is passed by a train coming from the left, then this event is reported to the *flagman* by the *trigger*. The flagman then checks its *condition* and detects that the right contact is turned off. Since the *condition* applies, the *flagman* is turned on as required.

If the right sensor is passed by a train coming from the right, then nothing happens because the right contact is not part of the *trigger*. If the train passes the left contact a few moments later, then this event is again reported to the *flagman* by the *trigger*. The flagman again checks its *condition* and detects that the right contact is still turned on. Since the *condition* does not apply, the *flagman* is not turned on.

By assigning *operations* to the *flagman*, it is possible to operate other elements depending on the direction of passing trains.

Example: Detecting uncoupled Cars

The following example demonstrates how inadvertently uncoupled cars can be detected. This mechanism is useful at the entry to hidden yards.

For this mechanism, a track occupancy detector and two additional *flagman indicators* are needed. In the following, these flagmen are called “timer” and “alarm”.

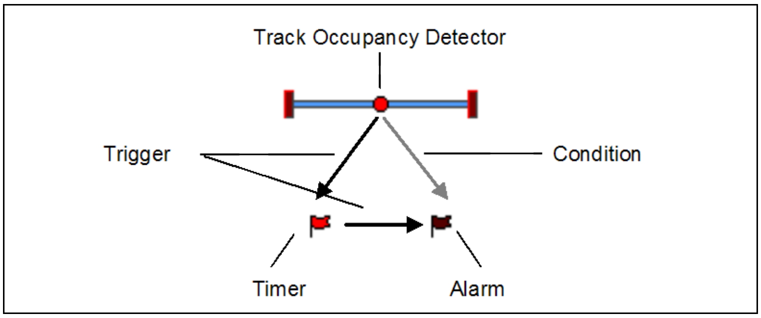


Diagram 164: Detecting uncoupled Cars

- Place or select a *contact indicator* in the track diagram and specify the digital address of the respective track occupancy detector.
- Create the two *flagmen* “Timer” and “Alarm”.
- Specify the occupancy detector in the “on” state as *trigger* of flagman “Timer”.
- Set the *memory* of flagman “Timer” to turning off **After 30 Seconds** and select the **Forced Reset** option.
- Specify “Timer” in the “off” state as *trigger* of flagman “Alarm”.
- Specify the track occupancy detector in the “on” state as *condition* of flagman “Alarm”.
- Specify the appropriate operations to be performed when flagman “Alarm” is turned on (e.g. emergency stop of all trains).

	Trigger	Conditions	Operationen	Memory
Timer	● Occupancy Detector	-	-	Forced Re-set: After 30 Seconds
Alarm	⚑ Timer	● Occupancy Detector	appropriate emergency operations	-

Table 13: Detecting uncoupled Cars

When the track occupancy detector is passed by a train, the flagman "timer" is turned on by its *trigger*. It remains on for 30 seconds. After 30 seconds, the "timer" is turned off again, even if the track occupancy detector is still turned on – this is done because the **Forced Reset** option is selected. Turning off flagman "timer" is reported to the flagman "alarm" by the *trigger* of "alarm". The flagman "alarm" now checks its *condition*, i.e. if the track occupancy detector is still turned on by some inadvertently uncoupled cars. If this is the case then "alarm" is turned on and performs the emergency operations.

The time period specified as memory of flagman "timer" must be large enough to enable the longest/slowest train to leave the occupancy detector. Otherwise a false alarm would be triggered. On the other hand the time period must be shorter than the interval between two successive trains. Otherwise it could happen that the next train has already turned on the occupancy sensor when the "timer" is turned off.

It is obvious that this mechanism only works if uncoupled cars can be detected by the track occupancy detector. If necessary, the axles at the end of each train can be made conductive using an appropriate resistor.

Example: Simple Track Occupancy Detection

The following example demonstrates how a track occupancy detection is made possible using temporary track sensors.

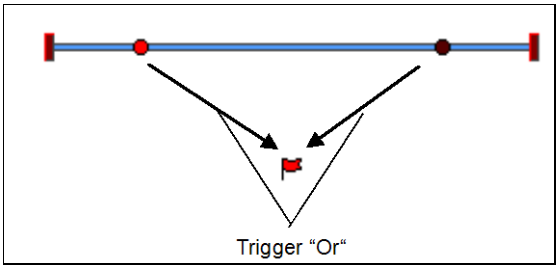


Diagram 165: Simple Track Occupancy Detection

In addition to the *contact indicators,* an additional *flagman indicator* is needed for indication of occupancy.

- Place or select the *contact indicators* into the track diagram and specify the digital addresses of the respective track sensors.
- Create a *flagman* for indication of occupancy.
- Specify both contact indicators as *trigger* of the flagman using the option **Or**.
- Set the *memory* of the flagman to **Toggle**.

	Trigger	**Memory**
Flagman	🔴 Left **OR** 🔴 Right	**Toggle**

Table 14: Simple Track Occupancy Detection

When a train enters the track section between the track sensors the *flagman* is turned on by its *trigger*. When the train leaves the track section, the corresponding *contact indicator* is turned on. This event is again reported to the flagman through its *trigger*. The option **Toggle** now makes sure that the *flagman* is turned off.

This mechanism also works if the train enters and leaves the track section on the same side.

14.6 Counter

Counters are special symbols in the switchboard of **TrainController™** **Gold**. They can be used to count events and processes and to evaluate their number in conditions and triggers.

Each counter has a variable start value, a switch-on and a switch-off value. Initially the start value is stored in the counter. With each click of the mouse the value stored in the counter is increased by 1. When the value reaches the switch-on value, then the counter is turned on. If the value continues to increase and exceeds at some point the switch-off value, then the counter is turned off.

The use of counters is especially interesting in automatic operation. For this purpose there are special operations, with which you can increase or decrease the counter value by 1 or reset it to the start value. Further, it is possible to evaluate in conditions and triggers of other objects, whether the counter is turned on or off.

With the switch-on value it can be determined, at which value the counter is switched on. In this way it is possible to evaluate certain minimum numbers in the conditions or triggers of other objects.

With the switch-off value it can be determined, at which value the counter is switched off. In this way it is possible to evaluate certain maximum numbers in the conditions or triggers of other objects.

By resetting the counter, the counter is reset to the start value. The start value is also the minimum value that can be stored in the counter. If the value stored in the counter is equal to the starting value, then the value cannot be reduced further.

The switch-on value should always be set at least as high as the start value, the switch-off value should always be set at least as high as the switch-on value.

When it is desired to evaluate a specific fixed counter value, the switch-on and the switch-off values are set to this fixed value.

14.7 Prototypical Signaling

Almost each prototypical signaling system can be modeled by applying the **Triggers** and **Conditions** (see page 296 and section 14.3) to signals. Beside Flagman elements (see section 14.5) signals are another type of elements, to which these features can be applied.

Especially by applying triggers to the particular states of a signal symbol, it is possible to let this signal symbol respond to arbitrary situations with the display of an appropriate aspect.

The following rules apply to the triggers of signals:

- The signal may change the displayed aspect, whenever an element changes its state, that is contained in a trigger of one of the signal aspects.
- If the triggers of two signal aspects apply at the same time, then the software may freely select one of these aspects.
- Empty triggers always apply. However, when a non-empty trigger applies at the same time, the software selects the signal aspect associated with a non-empty trigger. Non-empty triggers have got priority.
- Since non valid conditions may prevent a valid trigger from becoming effective (see page 298) and a later change of the condition does not override the concerning state change, a signal always reflects the situation when one of the triggers fired the last time; but it does not necessarily always reflect the current situation.

According to the above the following recommendations apply:

- Specify the particular triggers assigned to the various aspects of the same signal in a way, that no two different non-empty triggers become valid at the same time. Make use of the NOT-option (see page 284) to exclude the triggers of other signal aspects from a particular trigger.
- Leave the trigger of exactly one signal aspect empty, if possible. This is the “else” case and specifies the signal aspect in all cases, where no trigger of another signal aspect applies.
- To extend the simplified signaling system, which was introduced in section 5.8, and which is based on a 1 to 1 mapping of the internally calculated block signal aspects, with your own rules, you can apply these internally calculated aspects to triggers and conditions of signals (and other elements), too, and combine them with the status of other elements.

Evaluating the state of distant Signals

The internally calculated aspect of distant signals can be directly evaluated in **TrainController™ Gold** in triggers and conditions. This is useful for distant signals, which reflect the state of more than one main signal.

Assume a distant signal located at the entry into a station with five tracks. The state of the distant signal reflects the state of the main signal located at that track, which will be passed by the train. In total there are five main signals. In order to evaluate the right main signal in the trigger of the distant signal it is necessary to combine the status of all main signals with the status of the possible routes to the five tracks. This results in a relatively complex structure of AND and OR conditions containing the aspects of the five main signals as well as the status of all possible routes to the five tracks. Such structure must be created for each aspect of the distant signal.

To make this much easier **TrainController™** **Gold** provides the possibility to evaluate directly the internally calculated aspect of the distant signal for the said entry block. In the trigger of each aspect of the distant signal located at the exit of a certain block, the complex structure described above can be replaced by one single entry, which represents the internally calculated aspect of the distant signal for this block.

14.8 Macros

Macros are used to operate other elements.

They are very similar to *push buttons* in the *switchboard* (see section 2.5, “Signals and Accessories”). Like push buttons they are also able to perform operations (see section 14.4, “Operations”). Unlike push buttons they are not placed in a *switchboard*. Instead macros are for example used in the operations called by other elements (see section 14.4, “Operations”), executed in *schedules* (see section 5.11, “Schedules”), executed in *timetables* (see section 16, “ Timetables”) or assigned to *engines* as part of their functions (see section 3.6, “Headlights, Steam and Whistle”).

In this way, macros are invisible and work in the background of the program.

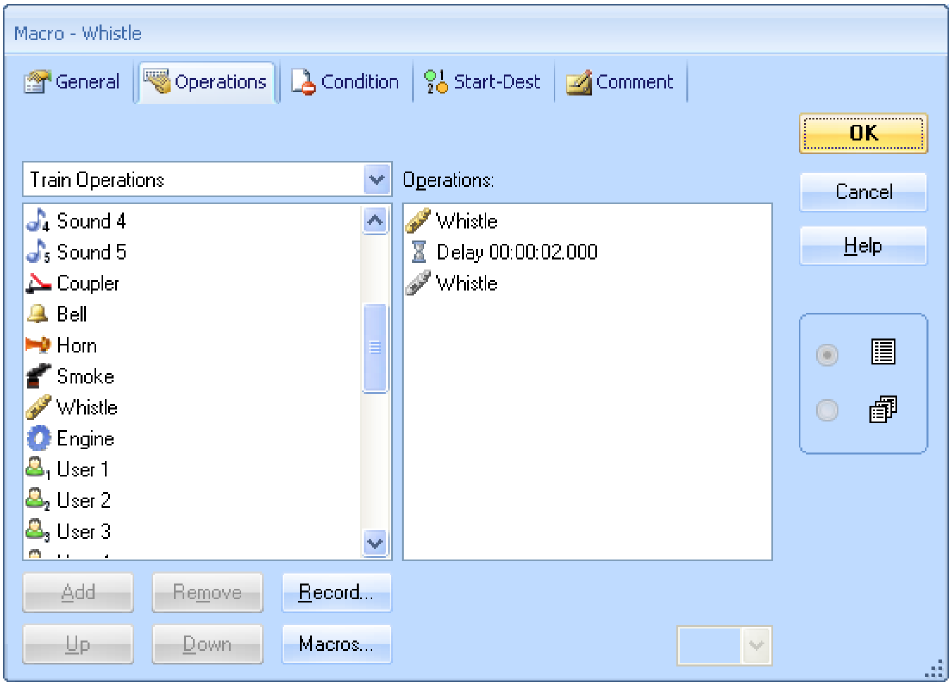
Engine functions assigned to macros can only be executed, if the macro is executed in the context of an engine. This is the case, if the macro is assigned to another engine function (in this way engine functions can be triggered indirectly by other engine functions), or if the macro is assigned to a schedule. If the macro is not executed in the context of an engine (e.g. by a timetable), then all engine functions contained in the macro are ignored.

Example: Automatic Engine Whistle

Engine or Trains running a schedule will blow their whistle for exactly two seconds when passing a certain section.

This is done in the following way:

- Open the macro list and create a new *macro* “Whistle”.
- Specify “Whistle on”, “Delay 2000 milliseconds” and “Whistle off” as *operations* of the macro according to Diagram 166.
- Assign macro “Whistle” as engine function or as an operation to be performed by a schedule.



Operations:
Whistle
Delay 00:00:02.000
Whistle

Diagram 166: Setting up macro Whistle

For a detailed discussion about engine functions refer to section 3.6, “Headlights, Steam and Whistle”.

Macros vs. Lists of Operations

Lists of operations are used to reduce the number of macros required. Most special purpose or single purpose sequences of operations can be applied directly as a list of operations to engine functions (see section 3.6, “Headlights, Steam and Whistle”) or to schedule operations (see page 199). Macros can be used to create general purpose sequences of operations wherever needed. Additionally it is possible to limit the use of macros by specifying appropriate conditions (see section 14.3, “Protection and Locking with Conditions”).

14.9 Extended Route Operation

Route Symbols in the Switchboard

In cases, where the Dispatcher is not being used, but it is desired to operate routes manually via a switchboard it is also possible to create *routes* in switchboards. Such routes are used to operate and lock the *tracks*, *turnouts* and *signals*, that belong to the route. Routes are operated in the switchboard like *on-off switches*. If the route is turned on, then all turnouts and signals of the route are operated. All track elements and signals along the path of the route remain locked in this position until the route element is turned off again. As long as these elements are locked, they cannot be operated or used by other routes.

Manual Routes vs. Automatic Routes

TrainController™ distinguishes between *manual routes* and *automatic routes*. Automatic routes can be operated automatically by the *Visual Dispatcher*. Manual routes can only be operated through their route control. They cannot be operated automatically by the *Visual Dispatcher*.

A manual route is created by inserting a route symbol into a switchboard at an arbitrary location. The location of the route symbol in a switchboard does not matter. Especially the location of the route symbol must not relate to the location of the tracks, turnouts and signals contained in this route. Manual routes are created, if the *Visual Dispatcher* is not being used at all or for those areas of your layout, which are only controlled manually with switchboards but not with the *Visual Dispatcher*.

An automatic route is always created as part of the *block diagram* of the *Visual Dispatcher* (for more details see section 5.2, “Blocks and Routes”).

With the exception, that manual routes cannot be operated automatically by the *Visual Dispatcher*, there are no further differences between manual and automatic routes.

Recording of Routes

All routes, that are not created by the automatic block diagram calculation in the dispatcher (see section 5.2, "Blocks and Routes"), require recording of the path of the route. This is done by selecting the route and using the **Properties** command of the **Edit** tab. In the following dialog, select the tab labeled **Route** and then press the button labeled **Record**.

This procedure starts the *switchboard recorder* and the path of the route can be recorded. The running switchboard recorder shows the small control panel displayed below:

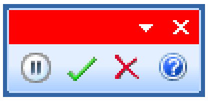


Diagram 167: The control panel of the switchboard recorder

The control panel contains four buttons with the following meaning (from left to right):

- **Break**: Recording is interrupted and no elements are recorded until this button is pressed once more
- **Stop with Save**: Recording is terminated and the recorded elements are stored.
- **Cancel**: Recording is terminated and the recorded elements are not stored.
- **Help**: Display help information about the recorder.

After starting the switchboard recorder, you are able to record the route. First select the switchboard in which the intended path of the route is located. Then, click on the track where the route will begin. Finally, click on the track element, where the route will end. **TrainController™** displays the tracks along the route as if the route were activated, but only if it is possible to reach the destination track from the starting track.

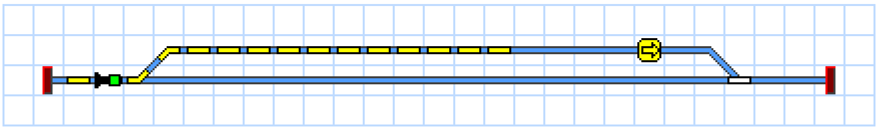


Diagram 168: Active route with turnout and signal

If you specify the start and end of a route in this way, then **TrainController™** tries to find an arbitrary suitable path. Alternatively, you can also specify a path from the start to the destination of the route. To do this, move the mouse to the starting track. Press and hold the left mouse button and drag the mouse along the desired path to the destination of the route. After reaching the destination release the left mouse button. Again **TrainController™** indicates the tracks along the route as if the route were activated.

To extend an existing route, additionally press and hold the Shift key during the procedure outlined above.

! **Note, that routes, which are generated automatically by the block diagram calculation in the dispatcher (see section 5.2, "Blocks and Routes") do not require any manual recording. The paths of these routes are automatically recorded during calculation of the block diagram.**

Signals in Routes and Protection of Routes

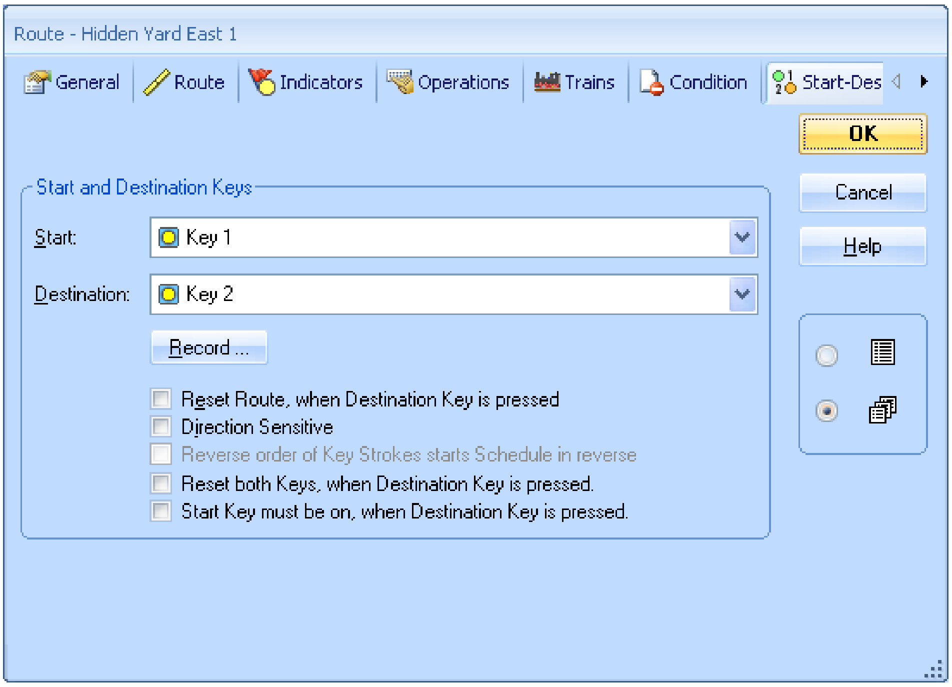
If *signals* will be operated in addition to the turnouts along the route, then you can add the related signals to the *operations* of the route. More details about *operations* can be found in section 14.4, "Operations". Signals included in these *operations* can be locked if desired until the specific route is turned off again.

In this way you can also *protect* the route. All turnouts outside the path of the route, which have been additionally assigned to the *operations* of the route, are operated accordingly and can be locked until the route is turned off. In this way you can lock turnouts outside of routes in appropriate positions to protect trains running on the route against collisions.

Operation of Routes with Start and Destination Keys

On control panels of real railroads, routes are often operated by first pressing a key near the starting point of the intended *route* and then pressing a key near its destination point. The operation of routes in this way can also be done with **TrainController™**.

For this purpose it is possible to assign a start and a destination key to each route. This is done by selecting the route and using the **Properties** command of the **Edit** tab. In the following dialog select the tab labeled **Start-Dest**. Here select the desired start and destination key.


Route - Hidden Yard East 1

Start and Destination Keys

Start: Key 1

Destination: Key 2

- [ ] Reset Route, when Destination Key is pressed
- [ ] Direction Sensitive
- [ ] Reverse order of Key Strokes starts Schedule in reverse
- [ ] Reset both Keys, when Destination Key is pressed.
- [ ] Start Key must be on, when Destination Key is pressed.

Diagram 169: Assigning start and destination key to a route

- It is possible to select *Push buttons*, *On-off switches* and *Contact Indicators* (see section 4, “Contact Indicators”) as start and destination keys. Especially with contact indicators it is possible to operate routes with start and destination keys in an external control panel (see section 14.10, “ External Control Panels”).
- Several options are provided to adjust the operation with start and destination keys to your needs. For example it is possible to specify that the route is released when the destination key is hit again after activation of the route. It is also possible to specify that the start key must be pressed until the destination key is pressed.

14.10 External Control Panels

Running an external control panel simultaneously with your computer is made possible by **TrainController**™. One solution is to not connect the buttons of your external control panel directly to your model railroad, but indirectly through the feedback decoders of your digital system. If a button on your external control panel is pressed, then this event is reported to the computer as a feedback signal by the digital system. With

TrainController™, you are able to create appropriate *contact indicators* to monitor these feedback signals. By assigning appropriate *operations* to these contact indicators, the accessories on your model railroad can be operated as desired.

Please note, that a push button or switch on your external control panel is associated with a contact indicator in TrainController™.

To operate a turnout with two buttons in an external control panel perform the following steps:

- Create a turnout symbol in your switchboard.
- Connect the real push buttons in your external control panel with two input contacts of a feedback decoder of your digital system.
- Create two contact indicators and assign the addresses of the two input contacts to them.
- Assign the first state of the turnout symbol to the *operations* of the first contact indicator and the second state of the turnout symbol to the *operations* of the second contact indicator.

A very useful application is the operation of *routes*. Without using a computer, the installation of expensive equipment would be necessary to operate *routes* with an external control panel. The option to assign contact indicators as start and destination keys to routes (see section 14.9, “Extended Route”) is very useful as well.

To operate a route with start and destination keys in an external control panel perform the following steps:

- Create a route symbol in your switchboard.
- Connect the real push buttons in your external control panel with two input contacts of a feedback decoder of your digital system.
- Create two contact indicators and assign the addresses of the two input contacts to them.
- Assign the contact indicators as start and destination keys to the route symbol.

14.11 Decommissioning of Objects

Blocks, routes, schedules, trains, turnouts and other objects can be decommissioned in **TrainController™** **Gold** at any time during operation. Decommissioned objects are excluded from operation. Decommissioned schedules, for example, cannot be started; decommissioned routes cannot be activated; decommissioned trains cannot be driven. A decommissioned turnout cannot change its state; such turnout can still be used in routes, however, if the current state of the turnout matches the turnout state required by the route.

Decommissioning is useful for each object, that will be prevented from being operated during operation of the layout. An object can be decommissioned at any time during operation. It is not necessary, to turn on edit mode before.

Objects are decommissioned by clicking on the object with the right mouse button and then selecting the command **Decommission** in the popup menu. If this command is applied to a decommissioned object, then the object is put into operation again.

Even though decommissioned objects cannot be operated, they can still be accessed and selected in lists or referred to by other elements without limitation. In particular decommissioned objects do still exist.

The decommissioning of objects also serves to edit the properties of certain objects when edit mode is turned off. To change a block, route, schedule, engine or car during the running operation, the object is first decommissioned. While the object is decommissioned, it is possible to change its properties, without the need to turn on edit mode. Ongoing operations are thereby not affected. However, the edited cannot be involved in the operation, as long as the properties are edited and the object is decommissioned.

14.12 Turnout Position Control

Turnout position control can be used to protect turnouts, that are currently locked in routes, against outside interferences or operation failures.

Turnout position control is based on different categories of turnout status:

a) The digital system stores and reports back the most recent turnout command (logical turnout state). This information can be for example used to detect, if a turnout is operated by an external handheld.
b) The turnout decoder can report back the electrical status of the turnout drive. This feature usually requires a turnout decoder, which is able to report back the current status to the digital system, and certain circuitry associated with the turnout drive, which reports the turnout status back to the decoder. This information can be used to detect, if the turnout drive failed to execute a turnout command issued by the digital system.

c) The electrical status of the turnout is reported back to the digital system or the computer, respectively, by feedback input contacts, which are associated with a separate feedback encoder. This information can be used, too, to detect, if the turnout drive failed to execute a turnout command issued by the digital system.

TrainController™ **Gold** provides turnout position control, too. For this purpose it supports all methods (a) to (c) listed above.

Turnout position control is usually only relevant for turnouts currently locked in routes. Since routes are just the tool to assure certain turnout positions, it should not matter, if a turnout changes its position, if it is not locked by a route. For this reason turnout position control in **TrainController™** **Gold** only applies to turnouts locked in routes, too.

If turnout control is turned on for a specific turnout, then **TrainController™** **Gold** automatically evaluates the turnout status reported back by the digital system (method (a) and (b)) to determine, whether the turnout position is in line with the associated route, if any.

If the digital system is not able to report back to the computer the status of turnouts, or in certain other situations it may be useful , to add method (c). To support this method **TrainController™** **Gold** allows to assign an individual feedback address and feedback status (on or off) to each particular position of a turnout. The feedback status is then additionally evaluated to determine, whether the turnout position is in line with the associated route or not.

By specifying a certain delay for each turnout **TrainController™** can be caused to check, whether a turnout requested by a route is in line with the route, when the specified delay has passed. The route is only activated, when the delays of all turnouts in the route have passed and if each turnout is in line with the route.

Additionally to the delayed check of turnout positions prior to final activation of each route **TrainController™** also reacts to turnout or feedback reports, which indicate, that a certain turnout has changed its position and that it is no longer in line with the route, that currently locks the turnout.

For turnouts with two drives (e.g. double slip switches or three way turnouts) it is possible to use two feedback addresses for each individual state of the turnout.

Note the following, however: whenever the state of one of the two feedback sensors changes, the software waits for a short time before a reaction or error handling takes place. In practice it will often happen, that the second drive and therefore the second feedback changes its state subsequently. Only after the operation of the second drive the final state of the two monitored feedback sensors is valid. Therefore, the reaction does not take place, until a certain short period without any reaction from the participating sensors has passed.

Error Processing

Turnout position control does not make any sense, if there is no reaction to failing turnouts.

One of the following mandatory reactions must be individually selected for each turnout. The selected reaction is performed, when the turnout is locked by a route for a train in a schedule:

- Search alternate path: if this option is selected, then **TrainController™** tries to continue the affected schedule with an alternate path.
- Lock block exit: if this option is selected, then **TrainController™** locks the exit of the block, where the affected train is currently located. This will cause the train to stop in this block and enables the human operator to clear the turnout problem.
- Stop schedule: if this option is selected, then **TrainController™** terminates the affected schedule. This is another, more drastic measure to prevent the train from passing the failing turnout and to enable the human operator to resolve the problem first.

Additionally and optionally it is possible to decommission the failing turnout in its current position and/or to execute a macro to perform other actions. Note, that decommission of the turnout does not prevent the turnout from being used by routes, that match the current position of the turnout. If a turnout, that failed to go to the thrown position, for example, is decommissioned, then it can be still requested by other routes in the closed position.

Limits of Turnout Position Control

While the methods (a) to (c) listed above concern the logical or electrical status of a turnout, the actual physical/mechanical position of the turnout, i.e. the actual position of the switch blade, can differ from the electrical state. This is for example the case, if the turnout drive operated the turnout correctly, but a small piece of ballast prevents the blade from following the drive completely. Such mechanical problems usually remain undetected or require at least complex and uneconomical changes of the turnout construction, which enable the blade to report back its position to a feedback input according to method (c). For this reason turnout position control can usually only be used to solve problems related to the logical or electrical status of turnouts, e.g. unauthorized operation of locked turnouts by external handhelds or electrical problems in conjunction with the decoder or turnout drive. Turnout position control can usually not solve mechanical turnout problems.

Because of these undetected mechanical problems and because the error processing of failing turnouts detected by turnout position control always causes an actual unwanted intervention into the normal operation of the layout, all measures to prevent turnouts from failing should be utilized first. Turnout position control is a measure, that can be added as an additional security measure for normally reliably operating turnouts. It should not be misunderstood as a compensation for unreliably working turnouts!

14.13 Extended Accessories, Cranes and Functional Models

In **TrainController™** **Gold** the predefined types of switchboard symbols (e.g. for turnouts, signals, switches, buttons, etc.) can be extended by self-created symbol types. These so-called extended accessories can be used to control the following items:

- Accessories having more than two digital addresses.
- Accessories (e.g. signals) with more than four states.
- Accessories, which are equipped with locomotive decoders and which are operated with speed, direction or function commands.
- Accessories, which are controlled with different decoders and commands (turnout and locomotive commands).
- Accessories, which are not only controlled with commands, but also return feedback to the PC.
- Selectrix accessories, which are controlled by several bus addresses or in which more than a bit of a bus address must be changed simultaneously.

The above features can occur in any combination. This allows control of arbitrary equipment (e.g. cranes, machines, working models, lighting, etc.) with unlimited possibilities via the switchboard.

All accessories can also be integrated into automated processes.

Use of extended Accessories

Extended accessories always consist of two components. The appearance and functionality of the equipment, e.g. how the device responds on which turnout command, is stored as separate data. This data is called *template of the extended accessory*. This template can be created by each user himself, but it can also be created by the manufacturer of the equipment or by experienced users and universally provided. Therefore it is possible to store the template in separate files, and distribute it over the internet.

The user of the device then only needs to load the appropriate template of an extended accessory into his project. He can then place one or more *symbols for the extended accessory* in the switchboard and has only to specify the corresponding digital addresses. This is sufficient to control his equipment. For the end user the use of an extended accessory is practically as easy as using one of the standard types (signals, switches, buttons, etc.).

The separate authoring of the functionality of the equipment allows, that this process must be done only once – e.g. by an expert, while novice users can then very easily use the equipment in their projects in any number.

The end user specifies one or more digital addresses for each instance of the article in his project. Depending on the configuration of the accessory these are a turnout address, a locomotive address and/or the address of a feedback sensor. If the accessory requires multiple addresses of the same type, e.g. several turnout addresses, then the different addresses are formed by adding an offset to the specified base address. The value of the offset is stored by the expert in the data for each accessory. Users of accessories which do not create such data, do not have to worry about this.

Creation of extended Accessories

The look & feel and the functionality of extended accessories, e.g. how the item responds to which turnout command, is stored as separate data (template) and can also be distributed in separate files (e.g. by the manufacturer of the equipment).

Extended accessories are always composed of basic building blocks (controls). For these compositions, there are no restrictions so that any combination may be created.

Following controls are available:

- Switches for switching of states with the mouse. Such switches have at least two positions (e.g. on/off, or left/right, or red/green, etc.). Switches can also have more than two states. This can be used, for example, to control signals with an arbitrary number of aspects.
- Push Buttons for temporarily activating a state with the mouse.
- Contact Indicators, which are associated with feedback sensors.
- Speed Sliders, with which speed steps on a locomotive decoder can be set by dragging a slider with the mouse.

- Indicator Controls for displaying states. These have at least two states, but may also have more states like switches. But unlike switches they are used for display purposes only and cannot be operated with the mouse.

Arranging the look of an extended Accessory

The authors of extended accessories are provided with many features to arrange the look of the accessory.

First, the size of the accessory can be freely selected. The size is used to define how many columns and lines the symbols, which represent these accessories, occupy in the switchboard.

Furthermore, for each available grid size 12x12, 16x16, 20x20, 24x24 and 28x28 a background image in the overall size of the accessory can be created with the built-in image editor. If the accessory serves for control of a crane, for example, then a stylized image of a crane cab can be selected as background image.

For each control of the accessory and each state of the control individual symbols can be assigned. These symbols can be either selected from the reservoir of built-in symbols or be created with the built-in image editor.

If the author of the accessory does not provide a background or symbol image for a specific grid size, then the image for the next best matching grid size is being used and scaled accordingly.

Finally, it is also possible to hide individual controls of an accessory. This is useful if these controls are only needed for the internal logic of the accessory, but are not required for operation or supervision by the end user. Such hidden controls are not visible for the end user. Also note that hidden controls are not taken into account, when calculating the actual column width and row height of the accessories. So it is quite possible to create a complex accessory with several controls, which occupies only one single switchboard cell.

Operations

The controls of an extended accessories can perform the following operations:

- Control of other controls of the same accessory.
- Control flow operations (see page 291).
- Special operations for extended accessories.

In particular the latter operations for extended accessories are essential, because these operations specify which turnout or locomotive commands the respective control sends to the connected decoder.

The following operations for extended accessories are available:

- Switch command to switch between the two states of a turnout decoder (e.g. from position red to green, from plus to minus, from closed to thrown, etc).
- Command for turning on or off of the switching current at a single contact of a turnout decoder (e.g. turn on the switching current for position red).
- Speed command to set speed step on a locomotive decoder.
- Direction command (forward / reverse) for locomotive decoders.
- Function command (e.g. light/F0 on, F3 off, etc.) for locomotive decoders.
- Simultaneous changing of one or more bits of a Selectrix bus address (only for Selectrix compatible systems).

These commands can be combined with each other and with the other commands listed above in arbitrary sequences.

Theoretically, it is possible, e.g. to send several turnout and locomotive commands with a single switch.

For each digital command an address offset is to be specified. This value describes what number is added to the base address specified by the end user to determine the actual digital address to be controlled. Assume a decoder for signals which is controlled by four successive turnout addresses. The end user can program the decoder to a base address (for example 31). The decoder is then automatically associated with the addresses 31 to 34. In the operations, which control the address 33 (base address plus 2), the address plus 2 is specified by the author of the extended accessory. If the end user of the accessory specifies 31 as the base turnout address, then all operations with address plus 2 will be applied automatically to address 33.

For push button and feedback controls separate sets of operations can be assigned to the two states (on and off) of the control. For switches and indicator controls an individual set of operations can be assigned to each state of the control.

Speed slider controls cannot perform operations. They are only used to send a speed command to a locomotive decoder according to the position of the slider.

With the above operations, the internal logic of an extended accessory is described (internal operations). Additionally, the end user of the accessories can add operations, with which other objects in his individual project can be controlled (external operations). Internal operations are always specified by the author of an extended accessory. External operations are always specified by the end user. Assume an extended accessory for a signal with more than four aspects. With the internal operations the author of the accessory specifies, how the signal works internally, e.g. which switch commands are sent to the decoder for which signal aspect. In addition, the end user can specify with an external operation , that the exit of a certain block is locked, when a particular signal symbol of this type shows red. In this way, extended accessories can be integrated into automatic processes.

Triggers

For switch and indicator controls it is possible to specify triggers, with which the states of other controls of the same extended accessory can be evaluated. This is in particular essential for indicator controls, because this is the only way to use them effectively.

For each state of a switch or indicator control an individual set of triggers can be specified. One trigger, however, should be left blank. It describes the default state of the control, which is set if none of the other triggers apply.

With the above triggers, the internal logic of an extended accessory can be described (internal trigger). Additionally, the end user can also add individual triggers to a switch or indicator controls, with which the states of objects in his individual project can be evaluated (external trigger). Assume an extended accessory for a signal with more than four aspects. With internal triggers the author of the accessory can specify, how signals of this type work internally. In addition, the end user can specify with an external trigger , that the signal shows red, when a particular block cannot be left. This is a further way to integrate extended accessories into automatic.

Internal and external triggers are linked by **Or**, i.e. the associated state of the control is set if either the internal or the external or both triggers are met.

Conditions

With the exception of feedback contact controls conditions can be specified for all controls of an extended accessory. In these conditions, the states of other controls of the same accessory are evaluated.

For each state of a switch or a indicator control an own condition can be set. For push button and speed slider controls the condition describes, whether the button may be turned on or whether the position of the slider may be changed.

With the above conditions, the internal logic of an extended accessory can be affected (internal conditions). It is for example possible to prescribe, that a signal may be switched from green to yellow only if it is turned to red in the meantime. The end user can also specify additional conditions, with which the states of other objects in his own project can be evaluated (external conditions). He can, for example, prescribe that the permission to pass a signal may only be shown, when a specific block may be left.

Internal and external conditions are combined by **And**, i.e. the condition is fulfilled when both the internal and the external condition is fulfilled.

Use in Operations, Triggers and Conditions

The controls of an extended accessory can not only be used internally, i.e. in the operations, triggers and conditions of other controls of the same accessory.

For the end user, it is also possible, to use the controls of an instance of this accessory in the switchboard in the operations, triggers and conditions of other objects of his own project. This bidirectional linking of extended accessories with the other objects of his own project allows a virtually unlimited integration of functional models, complex signals and other advanced equipment in automatic processes.

Using Extended Accessories in Engine Functions

In order to understand the following description, we assume a fictive locomotive decoder in which specific functions are not controlled by digital function commands, but by speed or direction commands. For the sake of simplicity, we assume a decoder, with which the couplers of a locomotive are controlled by the locomotive function F1. The side (front or rear) of the vehicle is selected by setting the direction of travel in the decoder to forward (for selecting the front coupler) or reverse (for the rear coupler) before calling F1.

We assume that this decoder is installed in a locomotive in addition to the decoder for motor control.

The couplings of the locomotive should now be controlled by the function buttons in the locomotive.

The two decoders cannot be controlled as a multiple unit or train set, since otherwise the control of the couplers would influence the driving direction of the locomotive, which of course is not desired. It is also not possible to transmit digital commands for the direction of travel directly via train operations or the like.

However, extended accessories can be used for this purpose. For this purpose, a simple extended accessory with a switch with two states ("front coupler" and "rear coupler") as the only control is sufficient. In the operations for the "front coupler" state, operations are entered for setting the direction of travel to forward and turning on F1, and, if necessary, after a delay, to turn off F1. The operations for the "rear coupler" state are arranged correspondingly.

For each engine in which this decoder is installed as an additional function decoder, a symbol of this extended accessory with the corresponding engine address as the basic address must be created. The call to this symbol is then entered as an operation for a locomotive function of the corresponding engine.

Let us assume, for example, that these are the three locomotives with the addresses 14, 24 and 34, and that the digital address of the function decoder is by 1 higher (i.e. 15, 25 and 35). In order to control the couplers in these three locomotives, three symbols of the extended accessory must be created, which control the addresses 15, 25 and 35 accordingly with their operations,
The call of the symbol for the address 15 is then entered as an operation for a locomotive function of engine 14. The other engines are configured accordingly.

Extended Accessories and Variables

In the example of the previous section, for each additional locomotive with the described function decoder a symbol of the extended accessory must be created again.

It would be more elegant and clearer if a single symbol of this extended accessory could be used for all such engines. For this purpose, however, the address of the calling engine would have to be evaluated when calling the symbol in an engine.

To support this, formulas with variables can be used instead of fixed values in certain operations of extended accessories (see section 14.14, “Variables”).

The values, which result from the calculation of these formulas, can not only hold the parameters of the digital command (such as speed step or function number), but also the address offset, which is added to the base address of the extended accessory to determine the actual digital address to be controlled.

If such digital command is executed by the operations of an extended accessory, which again is called by an engine function, then it is even possible to store the digital address of the vehicle, which is currently executing these operations, in a variable (by means of the context vehicle; see page 324). The value of this variable can then be used to calculate the address offset for the issued digital command.

In short terms: The digital address of a vehicle, which calls an extended accessory in its functions, can be used to calculate the address, which the digital commands are issued to by the extended accessory.

Thus, in the example described above, it is possible to use a single symbol of the extended accessory. The digital address of the corresponding function decoder (15, 25, or 35) can then always be derived from the address of the calling locomotive (14, 24 or 34).

It is also possible to evaluate the position and orientation of a vehicle in its current train set (if any) with variables. This information can then even be used in the example described above to control not only the coupler in terms of “front” and “rear”, but also in terms of “right or left side of the locomotive”, regardless how the train set is standing on the track and how the locomotive is set up in the train.

14.14 Variables

General

Variables can be used to make operations, conditions, triggers and many other options more flexible. By using a variable instead of a fixed value for a specific option it is possible to change the value of this option at runtime and depending of the current operational situation.

Variables can be used

- to count events happening during operation and evaluate the status
- to change the name of trains displayed in blocks dynamically
- to perform arithmetic calculations and evaluate the result
- to select an object out of a plurality of identical objects for operation with one simple generic macro
- to change the delays or probabilities of operations in a list of operations dynamically

- to affect the speed of temporary speed limits dynamically
- to affect the distances or ramps of markers or the distances of short distant moves dynamically
- and so on …

The possibilities are virtually not limited.

Variables are usually created by editing the operations, which access the variables. With the exception of local variables (see page 330) all other variables can be created, edited and deleted via the Explorer window.

Type of Variables

Each variable has a specific type, which is determined during creation of the variable. The following types of variables are available:

- **Number:**
  Number variables are used to store numeric values. Numeric variables can be used among others as counters, for numeric calculations and for program options, which numeric values are assigned to (e.g. the train operation **temporary speed limit** with a variable speed setting).
- **Text:**
  Text variables are used to store text strings. Text variables can be used among others for messages or for other program options, which text strings are assigned to (names of sound files, labels of goto operations, display name of trains, etc).
- **Time:**
  Time variables are used to store time values. Time variables can be used among others for operations, which use a time value (e.g. to set the current time of the clock, delay operations, etc.). The unit of the values stored in time variables is milliseconds.

- **Object:**
  Object variables are used to store references to objects. Object variables can be used to perform operations with the objects stored therein. Object variables are always bound to a specific type of object (e.g. two aspect signal), which is determined during creation of the variable. Only references to objects with this type can be stored in the variable. With object variables it is for example possible to create multi-purpose macros.
  Assume a macro, which contains an operation of a variable for two aspect signals in its list of operations. This operation sets the signal currently stored in the variable to green. This single macro can be used to operate a plurality of two aspect signals, one at a time. If the reference to a specific signal is assigned to the said variable prior to the call of the macro, then this signal will be operated by the macro. Assigning another signal prior to the next call of the macro will cause the macro to operate the other signal.

Context Objects of Variables

If a train operation is executed by a marker in **TrainController**™, then this operation is applied to the train reserving the block in which the marker is located. At the time of the operation, this is the *context* train. This already known principle is extended for variables as follows:

Variables are mainly used in the operations, triggers or conditions of objects. This determines also the *context* objects, which an access to a variable applies to.

The following table describes the conjunction between the access of a variable and the particular context objects for the most common cases:

Use of a variable by:	**Context Train**	**Context Block**	**Context Route**	**Context Schedule**
train operations	the train itself	current block of the context train	-	current schedule of the context train
engine function	the containing train set, if any; otherwise the engine itself	current block of the context train	-	current schedule of the context train
Block	the train, which is currently reserving the block	the block itself	-	current schedule of the context train
marker or indicator in a block	the train reserving the context block	the block, which contains the marker or sensor	-	current schedule of the context train
push button or switch associated with a block	the train reserving the context block	the block, which the push button or switch is associated with	-	current schedule of the context train
Route	the train, which is currently reserving the route	current block of the context train	the route itself	current schedule of the context train
Schedule	the train executing the schedule (only after start)	the current block of the context train	-	the schedule itself
Turnout	the train, which is currently reserving the context route	-	the currently active route across the turnout	current schedule of the context train
Turntable	the train reserving the block on the bridge	the block, which is associated with the bridge	the currently active route, across the turntable	current schedule of the context train
Macro	context train of the caller	context block of the caller	context route of the caller	context schedule of the caller

Table 15: Context Objects

The context vehicle exists only in conjunction with the execution of engine functions. It is the vehicle, which executes these functions. Note, that in many cases the context engine and the context train may be identical.

There are furthermore context stations and context boosters. These are just the stations (see section 15.7, “Stations”) and boosters (see section 15.8, “Booster”), respectively, which the context block (see above) is located in.

Operations for Access to Variables

The access to values to variables is performed by the control flow operation **Access to Variable** (see page 291).

Among others the following types of access operations are available:

- **= (Assignment):**
  This operation is available for all types of variables.
  It assigns a fixed value or the content of another variable to a variable.
  For number and time variables it is also possible to assign the result of a calculation based on a formula.
  It is furthermore possible to assign system wide status values, the status of the system clock or the fast clock, or certain status values of context objects (such as the speed of the context train or the number of trains located in the area of the context booster) to a variable. More details can be found in the **Help** menu of **TrainController**™.
- **+ (Add):**
  This operation is available for number, text and time variables.
  It increments the value stored in the variable by a fixed value or the content of another variable. In case of text variables the value is appended to the text already stored in the variable.
  For number and time variables it is also possible to increment the stored value by the result of a calculation based on a formula.
- **- (Subtract):**
  This operation is available for number and time variables.
  It decrements the value stored in the variable by a fixed value, the content of another variable or the result of a calculation based on a formula.

- **Random:**
  This operation is available for number variables.
  It assigns a random number between 0 and a specified value to a variable. This value can be a fixed number, stored in another variable or the result of a calculation based on a formula.
  If, for example, the specified value is 12, then a random number between 0 and 11 will be assigned to the variable. With a subsequent **+ (Ad**d) operation it is also possible to create a random value in a range with a lower boundary, which differs from 0.
- **! (Operation):**
  This operation is only available for object variables.
  It performs a specific operation (e.g. set signal to green) with the object currently stored in the variable.
- **= (Query):**
  This operation can only be applied to object variables.
  It is available in various forms, among others **= (Query Train)**, **= (Query Block)**, **= (Query Schedule)**.
  The operation can be used to retrieve an object currently linked to a given object, e.g. the train currently reserving a given block, or the current schedule of this train, or another object. This object is then stored in the variable.
  To turn on the lights of that train, which is currently reserving block “Southtown 3”, the following operations are executed:
  First the operation **= (Query Train)** with specification of block "Südstadt 3" is executed with an object variable. Thus, the train which is currently reserving the block "Südstadt 3" is stored in this variable. Then the operation **! (Operation)** with specification of the "light on" train operation is executed with this variable. This turns the light on of that train, which is currently stored in the variable, i.e. the train, which is currently located in "Südstadt 3".
- **@ (Reference):**
  This operation is available for all types of variables.
  It assigns a reference to another variable to a variable.
  This operation should be used only by very experienced users. If a reference to variable X is assigned to variable Y, then variable Y acts as a kind of “pointer” to variable X. Whenever variable Y is accessed or evaluated, then actually variable X is accessed or evaluated. If the reference to X is stored in Y and Y is incremented by 5 with the **+ (Add)** operation, for example, then actually the value stored in X is incremented.

Use of Variables in Operations

Many operations support the use of formulas instead of fixed values. One example is the **Delay** control flow operation. Instead of specifying a fixed delay it is also possible to specify a formula. During execution of this operation the value, which results from the calculation of the formula, determines the time of the delay.

Such formula consists of a text which contains numbers, arithmetic operators **+**, **-**, *****, and **/** or brackets like an ordinary mathematical formula. The formula can also contain variable wildcards (see below).

Evaluation of Variables in Triggers and Conditions

The content of variables can be evaluated in triggers or conditions.

- **Numeric, Time and Text Variables:**
  The value stored in numeric, time or text variables can be evaluated with the logic operations **= (equal)**, **<> (not equal)**, **>= (greater or equal)**, **> (greater than)**, **<= (less or equal)**, **< (less than)** with fixed values or the value currently stored in another variable.
  For numeric and time variables it is also possible to compare with the result of a calculation based on a formula.
  For text values the comparison is not case sensitive.
- **Object Variables:**
  The object referred to by object variables can be tested, whether it is equal or not equal to a fixed object or to the object referred to by another variable.
  It is also possible to evaluate the states of the stored object.
  For object variables with the object type train it is furthermore possible to check, whether the train currently stored in the variable matches a certain train description.

Variable Wildcards

In many options of the program with text as value it is also possible to insert the content of variables into the text by means of wildcards. Whenever this text is used (e.g. during execution of the system operation **Message**), the wildcard is replaced by the current content of the variable.

The content of number and time variables will be accordingly converted to text. The content of object variables will be converted to empty text.

To enter a variable wildcard into a text field type the two characters **%V**. If the particular text field supports the variable wildcard, these two characters are immediately extended to **%V[?]**. From now on this wildcard can only be edited as a whole, like one single character. Double clicking to this wildcard opens a dialog box, in which the variable to be used for the wildcard can be selected. After confirming the selection the '?' in the wildcard is replaced by the name of the variable.

If the text fragment **%V** is not automatically extended to **%V[?]**, then the corresponding text option does not support variable wildcards.

Example:

Assume, that the following message is to be displayed during operation with the **Message** system operation:
"Currently active schedules X – Currently activated routes Y", where X and Y will be replaced by the corresponding numbers.
Assume, that the number of active schedules is counted in the number variable "Count-S"; and the number of active routes is counted in the variable "Count-R".
Create a **Message** system operation and type in the following text:
"Currently active schedules %V – Currently activated routes %V"
This text will be automatically extended to
"Currently active schedules %V[?] – Currently activated routes %V[?]"
Double click the left instance of **%V[?]** and select the variable "Count-S". Double click the right instance of **%V[?]** and select the variable "Count-R".
The text now changes to
"Currently active schedules %V[Count-S] – Currently activated routes %V[Count-R]".
Assume that the value stored in "Count-S" is 5 and the value stored in "Count-R" is 11, when the **Message** system operation is executed. Then the following text will be displayed:
"Currently active schedules 5 – Currently activated routes 11".

Variable wildcards are supported by the following features (list not exhaustive):

- System operations with text as value (such as **Message**, **Sound File**, etc.).
- Content of text labels in the switchboard.
- Formulas of ramps and distances of markers in blocks (see page 174).
- Assignments to variables or evaluation of variables with text or a formula as operand.

Scope

The scope of a variable describes, in which context a value of a variable can be evaluated. Variables are able to store more than one value. The scope of a variable specifies, which of these values is accessed in the current situation.

There are the following scopes:

- **Global:**
  Variables with global scope can store one single value. This value can be accessed at all locations of the program, where variables can be used. If the value of a global variable is changed at one location of the program, this change will become effective at all other locations, too, where this variable is accessed.
- **Private:**
  Variables with private scope can store an individual ("private") value for each object in the program. This value can be set by the operations performed by this object and evaluated in conditions or triggers of the same object.
  The same private variable can be used by different objects. The private value stored in this variable for a specific object, however, is not visible for another object.
  Assume, for example three push buttons. The first push button may be pushed once, the second twice and the third three times. To achieve this, the same variable with private scope can be incremented (by one) in the operations of each push button. This will store a private value for each push button in the variable. In the conditions of each push button the variable can be compared with the individual limit (1, 2, or 3, respectively) of each button. The actual value used for this comparison is the private value of this push button stored in the variable.
  Macros are an exception. For macros no private value is stored. If a variable with private scope is used in the operations or conditions of a macro, then the private value of that object is used, which has called the macro. This allows, for example, to increment the private counters of the three push buttons mentioned above in a macro called by these push buttons.
- **Train:**
  Variables with train scope can store an individual value for each train in the program. The value stored in this variable for a particular train can be used in each object, which is currently associated with this train.
  Examples of such objects are blocks or routes reserved by this train; indicators or markers contained in blocks reserved by a train; turnouts contained in routes reserved by this train; etc.
  If, for example, a variable with train scope is contained in the condition of a marker, then the value stored in this variable of that train is used, which is currently reserving the block, where the marker is contained in.

- **Block:**
  Variables with block scope can store an individual value for each block in the program. The value stored in this variable for a particular block can be used in each object, which is currently associated with this block. Examples of such objects are the train located in this block; indicators or markers contained in this block; push buttons linked to this block etc.
- **Schedule:**
  Variables with schedule scope can store an individual value for each schedule in the program. The value stored in this variable for a particular schedule can be used in each object, which is currently associated with this schedule. Examples of such objects are the trains currently executing this schedule; blocks reserved by a train, which is currently executing this schedule, as well as all indicators and markers therein; routes reserved by a train, which is currently executing this schedule, as well as all turnouts contained therein; etc.
- **Route:**
  Variables with route scope can store an individual value for each route in the program. The value stored in this variable for a particular route can be used in each object, which is currently associated with this route. Examples of such objects are the turnouts contained in this route.
- **Extended Accessory:**
  Variables with extended accessory scope can store an individual value for each symbol of an extended accessory in the program. The value stored in this variable for a particular extended accessory can be used in each control of the extended accessory.
- **Local:**
  Variables with local scope can store one single value. Local variables are created in the context of operation lists. Their value can only be accessed by operations in the same list of operations. Local variables are not managed via the Explorer window. A local variable is automatically deleted, when there is no operation anymore, which refers to it.

TrainController™ 9 Gold tries to resolve references to variable values as far as possible. If, for example, a variable with schedule scope is used in the operations of a macro, which is called by a marker in a block, then the software tries to find out, which train is reserving this block and which schedule is controlling the train. If there is such schedule, then the value stored in this variable for this schedule is used. If there is no such schedule, then an empty value (0, empty string, etc.) is used instead.

15 The Visual Dispatcher II

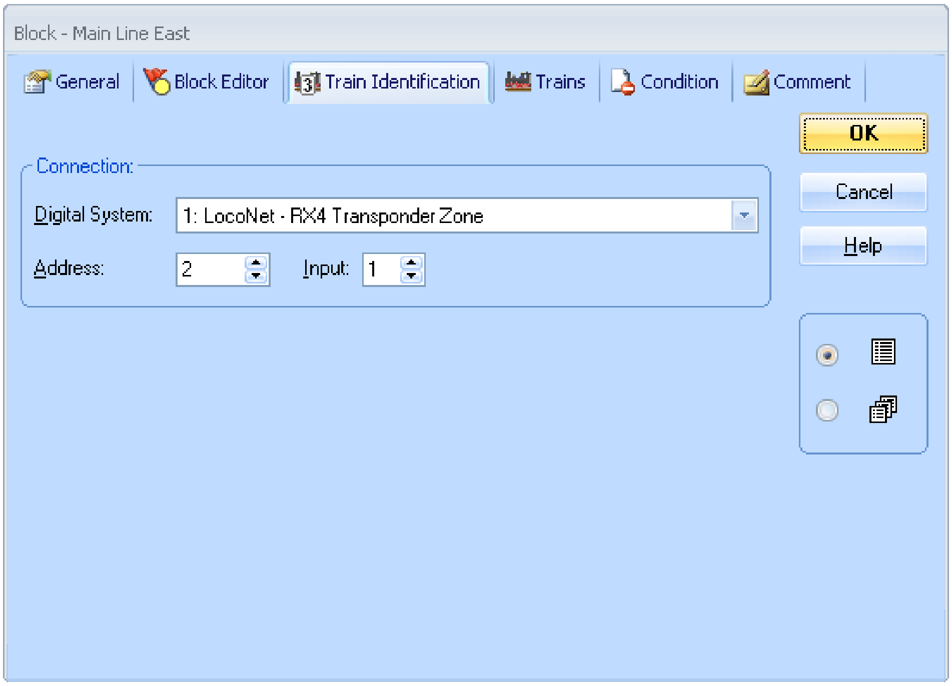
15.1 Train Identification

Track sensors are normally used to determine, if a track section is occupied by any *engine* or *train*. Some digital systems, however, are not only able to report occupancy, but also the identity of the occupying train. Examples of such systems are RailCom, Digitrax Transponding, Muet or HELMO. If a block is associated with an appropriate train identification device, then it is possible to determine the train, that is occupying a certain track section or detection zone, respectively. Associating blocks with train identification devices is called *train identification* in **TrainController™**.

It is very simple to setup a train identification system in **TrainController™**.

At first activate the option **Use Train Identification** in the **Setup Digital Systems** dialog for that digital system, where your train identification devices (readers, detectors, etc.) are connected.

On the computer screen each train identification device/train identification zone is linked to a block. To configure such device or zone in **TrainController™** simply enter the digital address of the train identification device/zone into the block properties (Diagram 170).



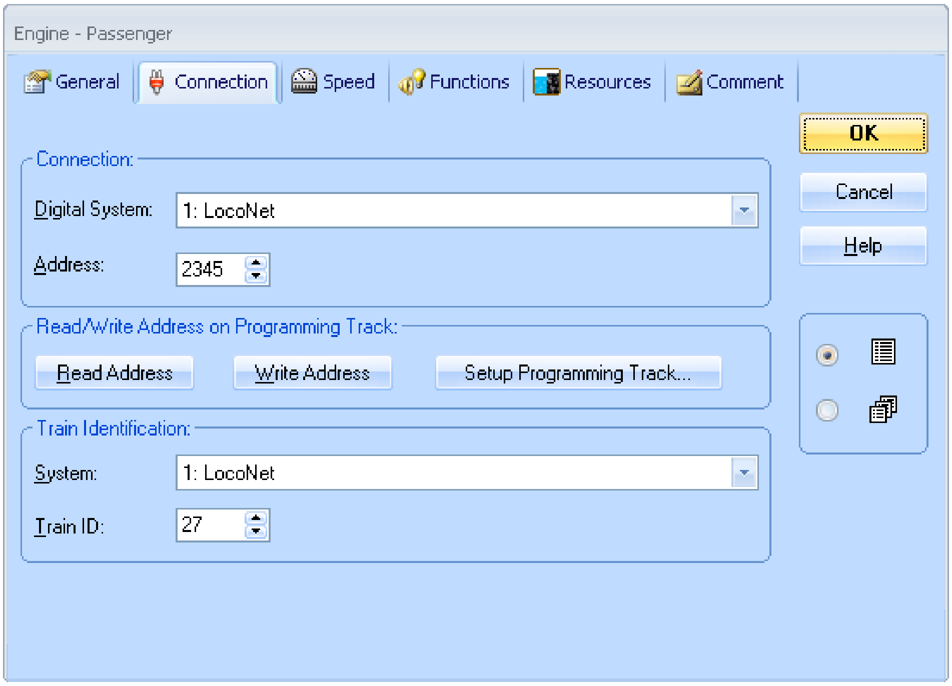
Connection:

Digital System: 1: LocoNet - RX4 Transponder Zone

Address: 2 Input: 1

Diagram 170: Specifying the digital address of a train identification device

Finally the train IDs are entered in the properties of the related engines or trains. The properties of each engine and train (see chapter 3, “Train ”) provide special options to specify an individual train ID for each engine or train. This is displayed in the following image.


Engine - Passenger

General | Connection | Speed | Functions | Resources | Comment

Connection:
Digital System: 1: LocoNet
Address: 2345

Read/Write Address on Programming Track:

Train Identification:
System: 1: LocoNet
Train ID: 27

Diagram 171: Specifying the digital connection and the train ID of an engine

In this example the train ID 27 is assigned to the steam engine with the digital address 2345. Of course digital address and transponder number do not have to be identical. Especially in the case of multiple units or if the transponder is mounted on a car rather than an engine it is very useful, that decoder addresses and transponder numbers are treated independently.

For *trains* in **TrainController™ Silver** (see section 11.1, "Train Sets") an additional option, **Use ID of Engines**, is provided. If this option is set then the train is not associated with an own train ID. Instead the IDs of the engines, that have been assigned to this train, are used. If the train is running and the ID of one of its engines is detected then this ID is mapped to the running train.

The screen displayed above might slightly change its appearance dependent on the actual capabilities of the connected train identification system. For some train identification systems you will not explicitly have to assign a train ID, instead there is a kind of auto-capture mechanism, with which train Ids can be automatically read from a passing train.

If more than one digital system is connected (**TrainController™** allows simultaneous operation of up to 12 digital systems) then it is even possible to use different digital systems for train steering and train identification. In this way it is possible to use a system like Digitrax as an additional train identification system, even if a digital system of another manufacturer is already installed. It is furthermore possible to use a train identification system like Digitrax on model railroads, which are controlled conventionally (in which case only transponder numbers and no digital decoder addresses are to be specified for the particular engines).

Specifying the train ID for each affected engine is the only additional effort with regard to the configuration of engines or trains. Nothing else is to be done.

Here is again a list of the necessary steps to configure train identification:

- Activate the option **Use Train Identification** in the **Setup Digital Systems** dialog for that digital system, where your train identification devices (readers, detectors, etc.) are connected.
- Assign the digital address of the train identification device/transponding zone to the properties of the associated block.
- Specify the engine/train ID of each engine or train, that you want to use for train identification, in the properties of each related engine or train.

When these steps have been done then the name and/or image of the train, that passes a certain train identification device or detection zone will automatically appear in the block symbol of the *Visual Dispatcher*. If there are one or more optional block symbols in a switchboard window associated with this block, then the train will appear in these boxes, too.

By assigning the digital address of a train identification device/detection zone to a block a relation is established between such detection zone and a block in **TrainController™**. This relation is used to perform an automatic train-to-block assignment, when a train is being detected in a train identification zone.

This relation should also be taken into account, when you wire your layout. Like a regular occupancy sensor each train identification device or detection zone can only belong to one block in **TrainController™** (see also section 5.8, "Arranging Indicators and Markers in a Block"). When a train is being detected in a train identification zone, it must be possible to clearly determine the block, to which the train is to be assigned.

TrainController™ does not only use train identification for automatic train-to-block assignment, but also for more complex safety functions. The *Dispatcher* uses train identification as a redundant anti collision protection in addition to the train tracking algorithms implemented in the software (see next section). If a train is reported in a block, which does not correspond to one of the “expected” positions calculated by the software, then the user is alerted and affected trains are stopped if desired.

Registration of unknown Trains

If a connected train identification device identifies a train ID , which does not belong to a known engine in **TrainController™**, then this train ID is displayed in the block symbol, which belongs to the associated block. This is illustrated in the image below.

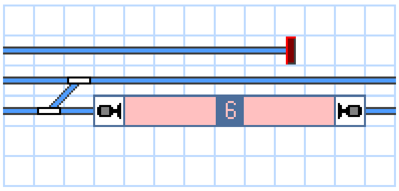


Diagram 172: Train ID of an unknown Train

This information cannot only be used to determine unknown train Ids or digital addresses of trains, respectively. It is also possible to use this information for fast creation of new engine records. While the unknown train is located in the block and its train ID is visible in the associated block symbol, it is possible to create a new engine record very quickly by clicking to the block symbol with the right mouse button and calling the **Create Engine for detected Train ID** command. This creates a new engine record, assigns the symbol of the new engine to the block and opens the **Engine Properties** dialog box, with which it is possible to specify a name and symbol for the new engine or to alter other attributes. In this way it is conveniently possible to register new engines in **TrainController™**.

15.2 Virtual Contacts and Virtual Occupancy Indication

General

X

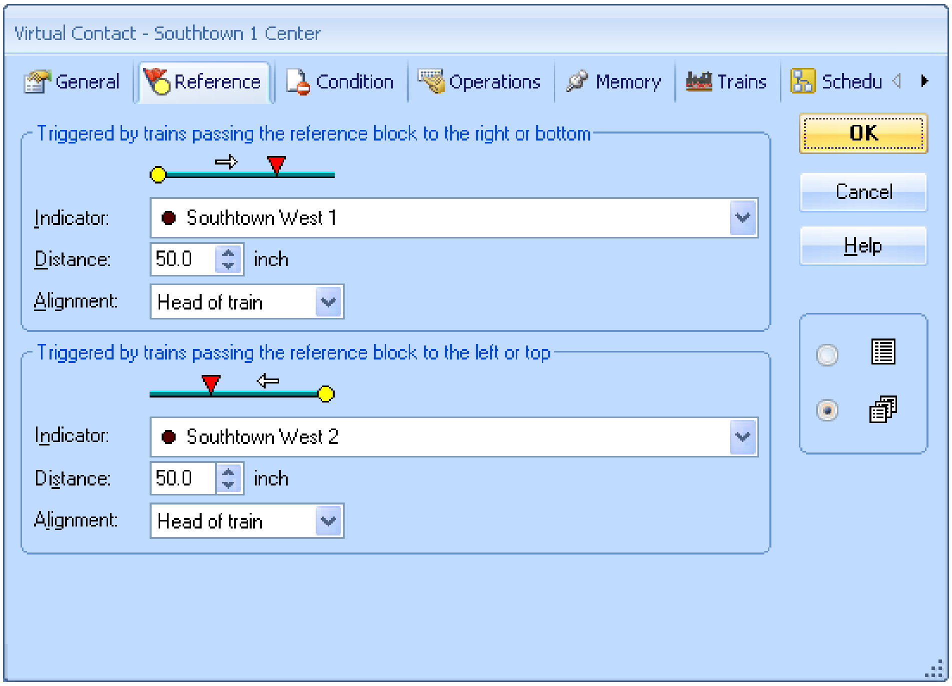
Virtual Contacts are similar to normal *contact indicators* (see section 4, “Contact Indicators”). But unlike contact indicators there is no related track contact or real sensor on the model railroad. Instead virtual contacts are assumed to be located at a specific distance from another indicator, which is called the reference indicator.

Virtual contacts can be used to reduce the number of necessary track sensors on your model railroad. Typical applications are triggering of operations by passing trains a certain distance from an existing sensor (see also section 14.4, “Operations”).

After creation of a virtual contact the following properties are specified:

!

- a maximum of two reference indicators, one for each direction of travel (see section 5.3, “Direction of Travel vs. Engine Orientation”). **These indicators must already be assigned to a block.**
- the distance from the specified reference indicator
- whether the virtual contact will be turned on, when the head, the middle or the tail of a train passes the point, where the virtual contact is assumed to be located


Virtual Contact - Southtown 1 Center

Triggered by trains passing the reference block to the right or bottom

Indicator: Southtown West 1
Distance: 50.0 inch
Alignment: Head of train

Triggered by trains passing the reference block to the left or top

Indicator: Southtown West 2
Distance: 50.0 inch
Alignment: Head of train

Diagram 173: Arranging a Virtual Contact

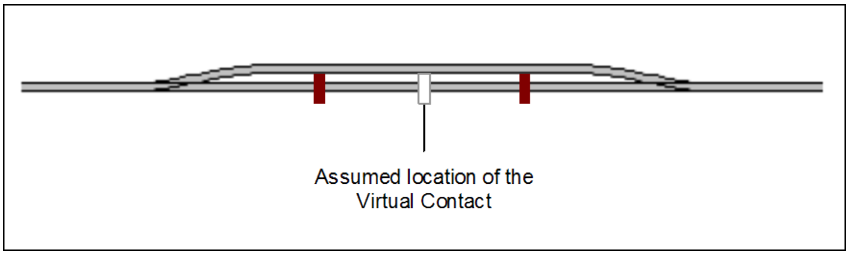


Diagram 174: Virtual Contact with two reference indicators

The image above shows a virtual contact (white rectangle) with two reference indicators (dark red markers). When a train passes the left indicator from the left to the right, then the current scale speed of the train and the distance of the virtual contact to this indicator is taken into account to calculate the time, at which the train passes the assumed location of the virtual contact. Even if the train changes its speed after passing the left indicator, this is taken into account and the resulting time is adjusted accordingly. When a train passes the left indicator from the right to the left, the virtual contact is not turned on.

Virtual contacts work only under the following conditions:

- if a train is stopped or changes its direction after passing the reference indicator and before arriving at the assumed location of the virtual contact, then the virtual contact is not turned on, even if the train continues traveling in the original direction and passed this location.
- it is very important, that the scale speed of the passing engines and trains can be calculated correctly. For this reason it is recommended that you adjust the speed profile of each affected engine accordingly (see section 3.5, “The Speed Profile”).
- it is very important, that the direction of travel of each passing engine or train is known. Otherwise virtual contacts could be turned on by trains traveling in the wrong direction. For this reason it is also essential to determine, which engine or train passes the reference indicators assigned to the virtual contact. This is only possible, if the affected trains are running under control of the Dispatcher, and if the reference indicators are assigned to blocks.

!

Virtual contacts can only be turned on by engine and trains running under control of the *Visual Dispatcher*. The reference indicator must be assigned to a block.

In conjunction with Virtual Contacts the difference between momentary contacts and occupancy sensors has to be taken into account. If an indicator representing a momentary contact is used as a reference indicator of a Virtual Contact then the one and only sensing point represented by the momentary contact is used as the base of the distance from the reference indicator to the Virtual Contact.

If an indicator representing an occupancy sensor is used as a reference indicator of a Virtual Contact then the sensing point reached first by trains running into the particular direction is used as the base of the distance from the reference indicator to the Virtual Contact. In Diagram 93, for instance, the left boundary of the sensed track section is used as the base of the distance for trains running from the left to the right.

Using Virtual Contacts as Indicators in a Block

Virtual contacts can be used to stop trains in a block in cases, where the actual occupancy sensor in this block is already turned on, e.g. by waiting cars. The reference contact of such virtual contact could be an indicator of the previous block, in which case the entry into the related block is indicated a certain time after a train passed the indicator of the previous block.

Virtual Occupancy Indication

X

If an indicator is associated with a momentary contact then this contact can be upgraded with the *memory* of the indicator to a *virtual occupancy* sensor (see section 14.2, "The Memory of Indicators"). If this is done the indicator stays on after activation of the contact until the complete train has passed the point where the contact is located. It is possible to take into account the point where the contact is activated or the point where the contact is deactivated. In this way it is for example possible to avoid premature release of routes in cases when long trains pass a route and only momentary contacts are used. This option works only for trains under control of the Dispatcher and it relies on correct specification of the length of each train.

Virtual Contacts can be combined with Virtual Occupancy Indication, too. The Memory is namely also available for Virtual Contacts. In this way a Virtual Contact will be turned on when a train reaches a certain point on the layout. And the Virtual Contact will stay turned on, until the last car of the train has passed this point.

!

Please note the difference between Virtual Contacts and Virtual Occupancy Indication. A Virtual Contact marks a certain point on your model railroad, i.e. a Virtual Contact is turned <u>on</u>, when a train is assumed to arrive at a certain point. Virtual Occupancy Indication is used to turn a certain real or Virtual Contact <u>off</u> when a train has passed a certain point completely.

15.3 Controlling the traffic flow in Schedules

Limiting the Reservation of Blocks and Routes in certain Schedules

For each *block* and each route in a schedule it is possible to specify a *condition*. This is a condition, which must be valid when a block or route is about to be reserved during a running schedule. As long as the condition does not apply it is not possible to reserve the block or route. How conditions work is outlined in section 14.3, “Protection and Locking with Conditions”.

This feature provides additional control. It is for example possible to specify, that a certain block may only be reserved, if a certain *on/off switch* is toggled off. Turning off or on this switch you can intervene into the traffic flow at any time and lock or release the affected block.

Such conditions can be specified on a global or a per-schedule base. Global conditions are specified as part of the properties of blocks or routes as outlined in section 14.3, “Protection and Locking with Conditions”. They are valid for all schedules, that use these blocks or routes.

Conditions can also be specified on a per-schedule basis, while editing the diagram of a schedule. Conditions specified in this way apply only when this schedule is being executed. Such local per-schedule conditions are always only valid for the schedule they have been specified for, other schedules are never affected by these local conditions.

In **TrainController™** **Gold** it is additionally possible to specify conditions, which apply only to AutoTrain runs or only to spontaneous runs.

Critical Sections

In the diagram displayed below “Main Line East” and “Main Line West” are specified as critical sections. Critical sections are displayed on the computer screen with a blue marking.

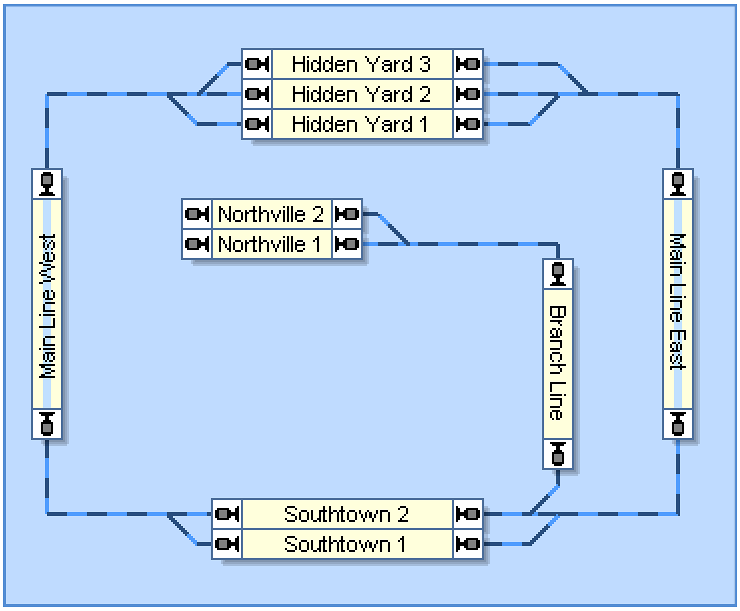


Diagram 175: Critical Sections

The most usual application of critical sections is to protect opposing trains from deadlocking each other. If the *Dispatcher* encounters during reservation of the next blocks ahead, that a block is marked as critical section, it will continue reserving further blocks, until a block is reached, that is not marked as a critical section.

If in the diagram displayed above, block “Main Line East” is reserved for a train that is about to leave a block in “Hidden Yard”, then the *Dispatcher* continues with reservation of a block in “Southtown”. If it is not currently possible, to reserve a block in “Southtown”, because both blocks in “Southtown” are already reserved by other trains, then the *Dispatcher* will not even reserve “Main Line East” and the train does not get permission to leave “Hidden Yard”.

A train may enter a critical section only if it is sure that it can leave the critical section on the other side.

If a critical section contains more than one block, then either all blocks of this section <u>plus</u> the first block behind this section are reserved in one step or no block is reserved at all and the train must not proceed.

A typical example of a critical section is a single track line between two stations, which can be traveled in both directions. If there is one or more blocks between these two stations, then these blocks should be marked as critical sections. A train, which is about to leave one of the two stations towards the other station will not leave this station, if it is not sure, that a block in the other station is available, i.e. if it is not sure, that it can leave the critical section on the other side. This prevents trains from deadlocking each other on the single line track between the stations.

There is a specific option, though, that allows trains, that are executing the same schedule at the same time, to share a critical section. In this way it is possible to queue several trains heading in the same direction in the same critical section while opposing trains must wait, until the complete critical section is clear. This allows several trains to follow each other on a single track line while locking opposing trains.

Critical sections can be assigned to blocks on a per-schedule base or alternatively in the main block diagram. A block that is marked as a critical section in the main block diagram will be treated as such in all schedules, that contain this block. A block, that is marked as critical section in certain schedules only will only affect those trains, that are controlled by this schedule.

The Train Guidance System

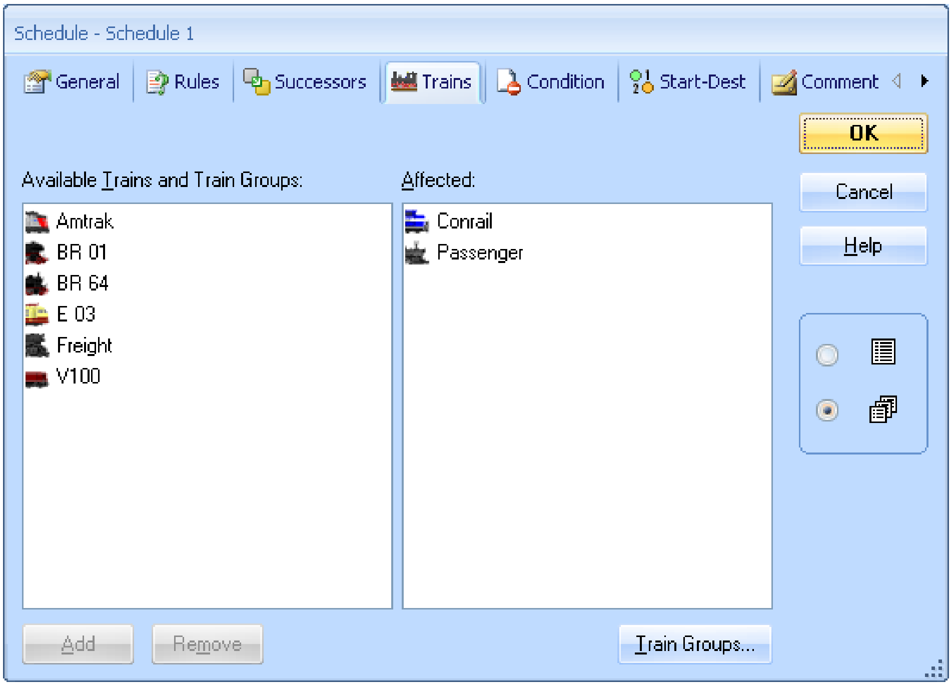
X

The usage of blocks, routes and schedules can be restricted to certain engines, trains or vehicle groups. In this way it is possible to ensure, that certain schedules are only started with passenger trains or to avoid electric engines entering tracks without overhead cables. This feature can also be used to make sure, that trains automatically entering a hidden yard are directed to tracks, which are long enough to hold the train.

If no engine or train is explicitly specified as a permitted train, then the block, route or schedule can be used by all trains.

A schedule can only be started when a train can be found in a block of this schedule and this train has the permission to use this schedule.

If you want to define *home blocks* for certain trains, e.g. in a hidden yard, then assign these trains to their desired home blocks as permitted trains. As a consequence only these trains will enter and stop in the affected blocks, while other trains will be automatically directed to other blocks.



Available Trains and Train Groups:

- Amtrak
- BR 01
- BR 64
- E 03
- Freight
- V100

Affected:

- Conrail
- Passenger

Diagram 176: Specifying the Trains permitted to use a Schedule

Vehicle groups are useful in conjunction with schedules in order to put together related engines or trains in groups. For example you can create the group of all *passenger trains*, or all *freight trains* or all *electric engines*. If you want to create several schedules only for freight trains, then you do not have to specify the particular trains as permitted trains for each affected schedule. It is much more convenient to create a vehicle group for your freight trains and assign only this group to the affected schedules.

Vehicle groups can contain other vehicle groups, too. In this way the vehicle group of all passenger trains can contain the group of all local trains and the group of all express trains.

For each block, route or schedule it is additionally possible to specify a *condition*. This is a condition, which must be valid when the block is reserved, the route is activated or the schedule is started. As long as the condition does not apply it is not possible to use the block, route or schedule. How conditions work is outlined in section 14.3, "Protection and Locking with Conditions".

This feature provides additional control. It is for example possible to specify, that a certain schedule may only be used, if a certain *on/off switch* is toggled off. Turning off or on this switch you can intervene into the traffic flow at any time and lock or release the affected schedule. It is not possible to start a locked schedule.

Train Guidance based on Train Length

The length of locomotives, cars and train sets can affect, how schedules are executed.

For this purpose it is possible to specify a maximum train length for each block. This maximum train length describes, up to which length trains fit into the particular block. Together with the length specified for each train and car or calculated for each train set, respectively, **TrainController™** **Gold** can determine, whether a train fits into a particular block or not.

This is used to accomplish the following goals:

- Trains can be prevented from driving to destination blocks of schedules, which are shorter than the length of the train.
- Trains can be prevented from performing unscheduled stops in blocks, which are shorter than the length of the train. If such blocks are reserved by longer trains, then these blocks will be treated in a similar way like critical blocks.
- Trains can be prompted to prefer the shortest destination block, which is long enough to store the train.

The features listed above can be activated by specifying a maximum train length for the concerning blocks and by checking certain schedule rules.

These features are in particular very useful in conjunction with train sets, that change their formation during operation. If such changes also affect the length of the train set, then this may change the destination, to which trains run or where trains perform unscheduled stops. If trains entering a hidden yard always prefer to drive to the shortest destination block in this yard, which is longer than the train itself, it is possible to utilize the available track space optimally. In such case no train will waste track space by running into a track, that is longer than necessary. Since **TrainController™** **Gold** is able to calculate the length of train sets, the track space in the hidden yard can be even used optimally for train sets, which change their formation during operation.

The following aspects should be also noted:

- The maximum train length specified for each block does not have any effects to the braking ramp of trains or to the location where trains stop in the block. There is no correlation between the maximum train length of a block and the distances specified for shifted brake and stop markers (see page 171).
- If a zero length is specified for a vehicle or all vehicles in a train set, respectively, then this train fits into all blocks. Zero length is the initial default setting for each vehicle.
- If a zero length is specified for a block, then all trains fit into this block. The block is assumed to have an "unlimited length". Zero length is the initial default setting for each block.

Forcing a Train to start a Schedule in a certain Direction

Normally each train, which is suited to execute a certain schedule, can be started in both directions, i.e. forward or backward. Therefore it does not matter, with which orientation the train is standing on the track. It will be started in the right direction of travel in each case.

With specific schedule rules, however, it is possible to force all trains started by the concerning schedule to start in a certain direction, i.e. forward, backward or by maintaining their current direction.

If the rule to start in forward direction is activated, for example, then a train will not be started by the concerning schedule, if it had to run in backward direction. The above is correct for single locomotives. For train sets the rule works slightly different. In conjunction with train sets 'forward' is interpreted as 'pulling' and 'backward' means 'pushing'. If the rule to start in forward direction is activated, for example, then a train set will only be started by the concerning schedule, if a locomotive is located at the end of the train, which corresponds to the requested direction of travel. In other words: the train set will only start, if there is a locomotive, which will <u>pull</u> the train set. If the rule to start in backward direction is activated, then a train set will only start, if there is a locomotive, which will <u>push</u> the train set. Care has to be taken, if there are locomotives located at both ends of a train set. In this case the train will be pulled <u>and</u> pushed, regardless in which direction it is started. In this case the rule does not have any affect and will not prevent this train set from being started, regardless of the requested direction of travel.

Routes with separate occupancy indication

Release of routes can be controlled individually and independently from the occupancy state of adjacent blocks. It is possible to assign a set of indicators to each turnout or route. These indicators determine whether a route is occupied or not. If at least one of these indicators is turned on, then the route is assumed to be occupied. It is possible to assign the same indicator to more than one turnout or route.

The assignment of indicators to turnouts is only supported by **TrainController™** **Gold**.

- A route is assumed to be occupied, if at least one of the indicators is turned on, which are assigned to the route.
- A route is also assumed to be occupied, if it contains one or more turnouts and at least one of the indicators is turned on, which are assigned to these turnouts.

It does not matter, whether a certain indicator is directly assigned to a route or assigned to a turnout contained in this route (indirect assignment). The assignment of indicators to turnouts is more convenient in cases, where many routes pass the same common turnout. To accomplish occupancy indication for all these routes in one step it is sufficient to assign the indicator to the common turnout. Assignment of indicators to routes, on the other hand, is useful for routes, that do not contain turnouts at all, or if the occupancy indication of the route depends on indicators, that cannot by associated with turnouts.

Indicators associated with turnouts and routes, respectively, should be preferably created in **TrainController™** as part of the properties of each turnout or route, respectively, rather than as separate switchboard symbols.

Occupancy indication of routes allow routes to be released independently from the occupancy state of adjacent blocks. Blocks or routes are usually not released, until the train reaches a stop indicator in a subsequent block. If your routes are equipped with an own occupancy indication, it is possible to turn off this rule. In this case unoccupied routes can be already released, when the train reaches the first indicator at the entrance of a subsequent block. In this case the track area covered by such routes is available for other trains earlier.

! The rule to release preceding blocks and routes of a certain schedule at the stop indicator of subsequent blocks should be turned off only, if the routes contained in this schedule are equipped with an own occupancy indication. Furthermore the tail of each train should be able to trigger the sensors used for occupancy indication of the routes. Usually this requires, that the trailing cars of affected trains are lighted or the wheel sets of these cars are conductive.

! Occupancy indication of routes can also be used to detect cars lost in a turnout area or to prevent routes from being prematurely released, when a long train is completely filling the subsequent block, but the tail of the train is still located on the route. In this case the route is not released, even if this long train has reached the stop indicator of this block (exception: the block is the destination block of a schedule, in which case all routes contained in this schedule may be released).

Schedule Watchdog and Limited Aberration Protection

With a specific schedule rule it is possible to specify a schedule watchdog. This is the maximum time period between activation of two indicators. If no indicator is triggered within the specified period of time and the train is set to run at non zero speed, then it is assumed, that the train got stuck. In such cases a warning is displayed in the message window and an error indication is raised in the **Engines + Trains** window.

It is furthermore possible to activate a limited aberration protection for each schedule. If the train under control of this schedule is detected by train tracking in block, which does not belong to the schedule, then the train is stopped immediately, a warning is displayed in the message window and an error indication is raised in the **Engines + Trains** window. Such condition can occur, for example, if a turnout contained in the schedule does not work correctly and the train is directed to a wrong block.

!

The limited aberration protection can detect certain, but not all possible conditions, where trains run into wrong blocks. In particular it usually cannot protect the train against collision, if the wrong block is already reserved by another train. Aberration protection should not be misunderstood as collision protection and it does not rid you from arranging for reliably working hardware.

Detection of lost Cars

In **TrainController™** **Gold** a special schedule rule allows monitoring of lost cars. When a preset number of blocks behind the current block of the train remains reserved, then the internally calculated signal of the current block is set to red and the train is stopped.

In this way lost cars can be detected when they have conductive axles and when they cause an occupancy event in the block or route, where they got stuck. The appropriate sections are not released and sooner or later the minimum number of not released blocks is reached. Then the train is stopped and a corresponding status information is displayed on the screen.

Track Cleaning Trains

With a specific schedule rule it is possible to specify, that always that path is selected, that contains routes or blocks, which have been visited by the train under control of this schedule the longest time ago (“oldest” block or routes).

If two or more identical paths in a schedule are available, then the selection is usually performed by random. By using the option to select the “oldest” block or route the schedule will be performed in a more regular or systematic manner. This option ensures, that a train under control of this schedule selects another path each time it passes a certain branching. This option is only effective, however, if the available paths are identical and no obstacle is blocking a certain path. The option is also only effective on a per schedule and per train base. This means in particular, that all time stamps are cleared, when the schedule is terminated. Thus this option is only useful, if the train passes the same branching in the same schedule run several times; for example in cycle or shuttle schedules.

This option can also be used to arrange automatic schedules for track cleaning trains. Due to the factor, that the train will prefer to go to that blocks or routes, that have been visited by this train the longest time ago, it will sooner or later visit all blocks and routes it can reach in this schedule, assumed the schedule is specified as cycle or shuttle with an appropriate number of repetitions. This is a good qualification for complete and systematic track cleaning.

15.4 Overview of all Schedule Rules

In **TrainController™** **Silver** and **Gold**, the execution of schedules, AutoTrain or spontaneous runs can be customized to individual needs with a variety of rules.

The rules are divided into the following categories:

Schedule Start

This category includes rules that specify the conditions under which the train can be started.

- **Only select trains in start blocks:**
  Only trains are started, which are located in a start block of the schedule. If this rule is not activated, then the run can be started also with trains, which are located in an arbitrary block of the schedule.

- **Train may stay in start block:**
  The schedule can also be executed , even if the train cannot leave the current block.

- **Start Delay:**
  This rule specifies a delay (in seconds) after clearing the track ahead of the train and before setting the train in motion. It is applied at the beginning of the schedule and after each stop..

- **Start the oldest train:**
  This rule is effective if it is possible to execute the schedule with more than one train. Specify the number of oldest trains, where the train to be started will be selected from. The oldest trains are those, which are resting at their current position for the longest time.
  If **0** is specified for this rule, then this rule is disabled and the train to be started is selected by random from the available trains. However, with a schedule, which starts and end in a hidden yard and which is permanently repeated, this setting could cause a train to be started again very soon after terminating a run, or not to be started for a very long time.
  If **1** is specified for this rule, then actually the oldest train is always started. However, with a schedule, which starts and end in a hidden yard and which is permanently repeated, this setting causes the trains to be started in predictable order, which may become boring after a while.
  Specifying an appropriate value greater than **1** can provide a good balance between coincidence and predictability.

- **Do not reset resting time:**
  This rule is useful in conjunction with the use of the rule **Start the oldest train** in this or other schedules. The oldest trains are those, which are resting at their current position for the longest time. This resting time is usually reset whenever a schedule with the concerning train is terminated. By setting this rule the resting time is not reset, when this schedule is terminated. This is useful for schedules, which control the moving up of trains in a hidden yard, for example, or similar local maneuvers. The train which entered a certain yard first, for example, remains the oldest train in this yard, even if it is moved within this yard by a schedule with this rule set.

- **Train may only start in current direction:**
  Trains are only started, if they maintain their current direction of travel..

- **Train may only start in forward direction:**
  Single locomotives are only started in forward direction.
  Train sets are only started, if this results in a pulled train.

- **Train may only start in backward direction:**
  Single locomotives are only started in backward direction.
  Train sets are only started, if this results in a pushed train.

- **Control car pulls:**
  This rule is only effective together with the rule **Train may only start in forward direction**. Train sets are not only started, if this results in a pulled train, but also, if this results in a train with a control car at its head.
  In other words: If both rules are activated, then train sets are only started, if this results in a train with an engine or a control car at its head.
  If only the rule **Train may only start in forward direction** is activated, then train sets are only started, if this results in a train with an engine at its head.

Reservation of Blocks and Routes

This category includes rules that specify, how blocks and routes ahead are reserved and entered.

- **Enter occupied blocks:**
  Trains may enter occupied blocks.

- **Enter occupied routes:**
  Trains may and enter occupied routes.

- **Reserve occupied blocks:**
  Occupied blocks may be reserved for this schedule.

- **Reserve occupied routes:**
  Occupied routes may be reserved for this schedule.

- **Select route with least turnouts:**
  If there are more than one routes between two blocks, then the route with the smallest number of turnouts is selected. This rule is useful for double crossovers between two blocks, for example, to prevent the train from changing the tracks while travelling from one block to the next.

- **Select oldest block or route:**
  The software select the path via those blocks and routes, that have not been visited by the current train for the longest time. This rule can be used for track cleaning trains or to accomplish more varied operation.

- **Ignore distances:**
  The distances to the destination blocks and obstacles, i.e. the number of blocks and routes between the train and the destination block or obstacle, do not matter, when selecting the optimal path.

- **Maximum Detour:**
  This rule specifies the maximum number of blocks, that are accepted as detour on the path to the destination block. This rule is useful in all cases, where the train is allowed to select a path, which is longer than the shortest path (in terms of numbers of blocks), when the shortest path is locked by an obstacle. The previously existing rule **Ignore distances** could be used in such cases, too. But that rule often leaded to inacceptable long detours and required also much CPU capacity. The rule **Maximum Detour** is a much more accurate option for intervention.

- **Unavailable Blocks and Routes:**
  Sections (blocks or routes) which are currently not available, are normally considered as usable for the calculation of paths to the destination. It is assumed that they are only temporarily not available, and a path may be established through such sections if no better, free path exists. The train then selects a path via such sections, but may eventually stop until the assumed obstacle no longer exists. This was the default behavior in version 7.
  With the rules in this section, the inclusion of such sections can be turned off depending on the nature of the obstacle. This has an effect as if the sections were not part of the schedule or would be decommissioned. One must note, however, that already another train driving ahead of a train can prevent the start of a schedule completely.
  Include occupied: Occupied blocks and routes are included into the path search.
  Include reserved: Blocks, which are reserved by another train, are included into the path search.
  Include locked: Blocks with locked entries are included into the path search.
  Include restricted: Blocks and routes, which cannot be reserved due to an unfulfilled condition are included into the path search.

- **Smart look ahead:**
  At least one block ahead is always reserved ahead of the train. If there is a route behind the next block then this route and the subsequent block are reserved together with the next block, too.

- **Fixed look ahead:**
  The software reserves always a fixed number of blocks ahead of the train.

- **Reserve destination block at start:**
  The selected destination block is already reserved at start of the train. The train must not leave its current block, if the destination block cannot be reserved..

- **Reserve complete path to destination:**
  The train is only started, if the path to the destination block can be reserved completely. All non-destination blocks of the schedule are treated as 'critical'. Since this is an intensification of the previous rule, only one of these two rules can be activated.

- **Use start block as destination block:**
  This rule forces a train controlled by a circular schedule with several start and destination blocks, which are identical to the start blocks, to use that block as destination of the schedule, where the train started from. This rule is useful for example for schedules, which start and end in a hidden yard. In this way it is possible to define an implicit “pseudo home block” in this yard for each train starting and returning there without the need to specify this block explicitly as home for each train. By using this rule the pseudo home block is automatically (implicitly) assigned to the train, when the train is located in this block or when trains swap their (home) position in the yard by hand.

- **Include turntables:**
  This rule applies only to AutoTrain runs. If this rule is set, then turntables are included into the path search, even if they are not directly connected to a start or destination block of the AutoTrain run.
  Deactivating this rule prevents turntables from being included into paths from a distant start block to a distant destination. Turntables are limited to be used in paths, which start or end in an adjacent destination block (e.g. in the attached roundhouse).

Release of Blocks and Routes

This category includes rules that specify, how already passed blocks and routes are released. It is for example possible to specify different variants for the time, at which passed blocks or routes are released.

- **Time of Release - At Stop Marker:**
  Blocks and routes in this schedule are not released, before the train reaches a stop marker in a subsequent block.

- **Time of Release - Upon complete Entry:**
  Passed blocks and routes in this schedule are not released, before the train has entered the subsequent block completely. This rule assumes, that the length of the train is known. If the length of the train has not been specified, then the sections are released when the train reaches a stop marker in a subsequent block.

- **Time of Release - Smart:**
  Passed sections with own occupancy indicators are released, if they are no longer occupied. Sections without indicators are released at the stop marker of the subsequent block. This rule can only be used, if the trains are equipped with conducting wheel sets at the rear end.

- **Time of Release - Smart or upon Entry:**
  Passed sections with own occupancy indicators are released, if they are no longer occupied. Sections without indicators are released, when the train has entered the subsequent block completely (if the length of the train is known) or at the stop marker in a subsequent block (if the train length is not known). This rule can only be used, too, if the trains are equipped with conducting wheel sets at the rear end.

- **Time of Release - By Occupancy:**
  Passed blocks or routes are released, if they are not occupied, regardless whether they contain own indicators or not. This rule is only available for compatibility with earlier versions. If it is possible to release sections using occupancy sensors, then there is normally nothing to be said against using the **Smart** setting.

  Time of Release - Entry Offset:
  If sections are released upon complete entry of the train, then the point, where the complete entry is reported, can be displaced by a certain distance into the block for reasons of security.

- **Release destination block:**
  The destination block is released, when the schedule is terminated. This rule is useful if the train moves into a part of the layout, which is not supervised by the computer.

- **Keep previously active Routes:**
  Routes, that were already active prior to the reservation by the schedule, remain turned on when they are left or when the schedule is terminated. If this rule is not activated, all routes are turned off, when they are left or when the schedule is terminated.

- **No enforced release of blocks or routes upon termination:**
  Normally, all of the blocks and routes except the current block of the train are released at the end of a schedule (unless the previous rule is used). If this rule is activated, all blocks or routes, that cannot be released during the normal run of the schedule, are also not released, when the schedule is terminated.

Train Length

This category includes rules that specify, how the length of trains impacts the execution of the schedule.

- **Trains must fit into destination blocks:**
  Only those blocks are used as destination blocks, that are long enough to store the train.

- **Prefer shortest destination block:**
  Trains are preferably directed to the shortest available destination block, which is long enough to store the train. This is a very weak criterion, however. It is only effective, if the available destination blocks are 'comparable' with respect to other criteria or conditions. This rule is for example useful to select the shortest fitting track in a hidden yard with multiple parallel (and therefore 'comparable') tracks.

- **Enforce shortest destination block**:
  Trains are forced to go to the shortest available destination block, which is long enough to store the train. Shorter and longer destination blocks are generally not considered, regardless of whether they are available or not. If this rule is used, then the rule **Trains must fit into destination blocks** is automatically activated, too. In many cases, when using this rule, it makes also sense, to turn on the rule **Reserve destination block at start** to ensure that the right destination block is available before departure of the train or the execution of the schedule.

- **Do not enter short blocks:**
  Trains must not enter into blocks, which are too short. These blocks are treated as if they were not included in the schedule. This rule only affects blocks, for which a maximum train length is specified.

- **Do not stop in short blocks:**
  Trains must not stop in blocks, that are too short. These blocks are treated as 'critical' blocks.

- **No scheduled stop in short blocks:**
  Trains must not perform a scheduled stop in blocks, that are too short. Specified waiting times are ignored in such blocks..

- **Train must fit in uncritical blocks:**
  The tail of the train must not be stopped in a critical block. After passing a critical section a long train must proceed to the point where it completely fits into the blocks that lie beyond the critical section. Blocks which lie behind a critical section, but are too short to take the train are considered 'critical', too.

- **Do not release blocks or routes under long trains:**
  Blocks or routes are not released, if the train does not completely fit into subsequent blocks. If you want the affected sections to remain reserved even after the end of the schedule, the rule **No enforced release of blocks or routes upon termination** must also be switched on.

Train Sets

This category includes rules that specify, how train sets are treated.

- **Joining in destination blocks - Enter reserved destination block for joining:**
  Trains on this schedule may enter reserved destination blocks to join to vehicles already located there. If this rule is being used, the rules that allow to reserve and enter occupied blocks should be activated, too.

- **Joining in destination blocks - Join in destination block:**
  After entry into the destination block the incoming train joins to vehicles already located there. If this rule is turned off, then the entering train and the waiting vehicles remain separated. This can be used to e.g. move a shunting engine into a block with already waiting vehicles, without joining them together.

- **Joining in destination blocks - Use only reserved destination blocks**:
  Destination blocks are only used, if they are reserved by vehicles already located there. With this rule it is possible to prescribe, that a schedule, which is used for coupling to waiting vehicles, ends only in destination blocks, where vehicles are actually located. If this rule is not turned on, then the train can also be directed to an empty destination block.

- **Joining in destination blocks - Join with cars:**
  Destination blocks are only used, if they are reserved by cars already located there but no engines. With this rule it is possible to prescribe, that a schedule, which is used for coupling to waiting vehicles, ends only in destination blocks, where only cars are actually located, but no engines. If this rule is not turned on, then the train can also be directed to destination blocks, in which engines are waiting or which are empty.

- **Line up in destination block:**
  This rule allows trains to be lined up with other trains already waiting in the destination block. In this way several trains (usually locomotives) can be parked in a staging track without being connected with each other in a continuous train set. The value entered for this rule specifies the gap between the already waiting and the incoming vehicles.

- **Permit start without engine:**
  The schedule can be started with trains, that contain only cars. This allows to use schedules for the control of marshalling humps.

Signals

With the following rules it is possible to affect the calculation of the internal block signals:

- **Request yellow:**
  This rule requests the yellow signal for all blocks and routes in the schedule.

- **Request yellow for local schedule:**
  Same as before, but only if the schedule is a local schedule (see page 369, “Local Schedules”).

- **Reject yellow:**
  All requests for the yellow signal issued by blocks, routes or turnouts in the schedule are rejected. The internal block signal will be always set to green, when the train may proceed.

- **Reject yellow for local schedule:**
  Same as before, but only if the schedule is a local schedule (see page 369, “Local Schedules”).

- **Replace green by white:**
  The internal block signal is set to white, whenever the calculation of the signal results in a green aspect. This rule provides an optional alternate signal aspect for green – e.g. for alternate signaling of shunting moves.

- **Replace green by white for local schedule:**
  Same as before, but only if the schedule is a local schedule (see page 369, “Local Schedules”).

- **Replace yellow by white:**
  The internal block signal is set to white, whenever the calculation of the signal results in a yellow aspect. This rule provides an optional alternate signal aspect for yellow – e.g. for alternate signaling of shunting moves.

- **Replace yellow by white for local schedule:**
  Same as before, but only if the schedule is a local schedule (see page 369, “Local Schedules”).

Security

This category includes rules that increase the security of operation.

- **Watchdog:**
  This rule prescribes a period of time (in seconds). If this period passes without any expected feedback activity, then it is assumed, that the train got stuck somewhere. In this case a warning marker appears on the screen.

- **Limited Aberration Protection:**
  The train is stopped, if it is tracked by train tracking into a block, where it is no expected due to the block diagram of the schedule. With this rule only certain but not all possible aberration conditions can be detected.

- **Detect lost Cars:**
  The train is stopped, if a specified number of blocks behind the train cannot be released (see also page 347). Setting the value to 0 disables this rule.

Misc

This category includes rules that do not fit into other categories.

- **Adaptive Braking:**
  This rule activates the Adaptive Braking Procedure (ABP) for the schedule. ABP improves the exactness of calculated stop locations. ABP causes the train to slow down with individual speed and ramp values, when it has to stop in a block. These values are adapted to the individual characteristics of the train to stop the train as exactly as possible at the desired stop location.
  However, ABP may cause each train to slow down in blocks prior to the block, where the train has to stop. If this effect is not desired, then Adaptive Braking can be deactivated for each schedule.

- **Critical Sections - Share critical sections in this schedule:**
  When two trains are controlled by the same schedule at the same time, then they may enter the same critical sections.

- **Critical Sections - Share critical sections in the same direction:**
  When two trains are running in the same direction at the same time, then they may enter the same critical sections.

- **Anticipate Stop:**
  With this rule the speed of the train can be reduced to the specified speed, when the distant signal is red due to unscheduled stop. This supports provident driving of trains and reduces the ‘concertina effect’ if trains follow each other too closely.

Spontaneous Run

TrainController™ provides the following rules for spontaneous runs. These rules are available in all editions of **TrainController™**:

- **Reverse automatically:**
  With this rule trains are automatically reversed at dead ends. If this option is not set, then the run is terminated at dead ends..

- **Reverse after stop:**
  With this rule, a time can be set. If a train in a spontaneous run cannot proceed for the specified period of time train, then it is automatically reversed. This rule is useful to dissolve blockages of opposing trains automatically.

- **Automatic Route Activation:**
  If this rule is set then routes are automatically selected and activated as needed. If this option is not set, then all routes must be activated manually.

- **Local:**
  This rule causes the associated train to perform a local spontaneous run (see page 369, “Local Schedules”).

15.5 Examples

Example: Manual Control of Station Entry

Trains will run automatically on the small layout displayed below. Before entering the station trains will wait until the human operator selects a destination track with start and destination key.

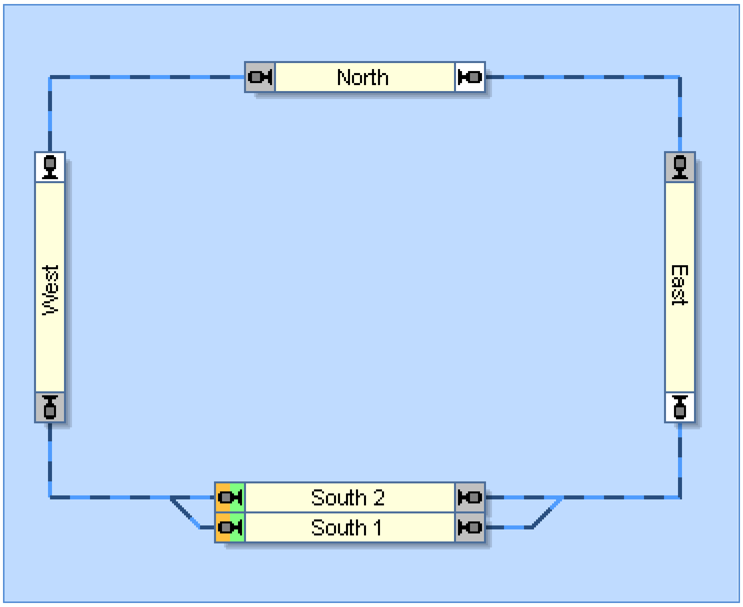


Diagram 177: Manual Control of Station Entry

This situation can be controlled with one single schedule, that is displayed in Diagram 177, too. The start and destination blocks of this schedule are located in block “South 1” and “South 2”. Since each schedule can be started in two directions, this schedule is able to control trains that travel clockwise, as well as trains that travel counter-clockwise.

When the schedule is being started in either direction, then we take care, that the entry to both, “South 1” and “South 2” is locked. The started train will proceed to “East” or “West”, respectively, and stop, if the lock has not been released until then.

By activating a route you can pre-select a path to “South 1” or “South 2”, respectively. This route can be associated with a pair of start and destination keys. If additionally the release of both blocks in “South” is performed as *operation* by each route, too, you are not only able to pre-select a path with the start and destination key, but also to release the lock which causes the train to continue its journey.

There are several possible variants. Instead of locking the entries to “South 1” and “South 2” you can also lock the bottom exits of “East” and “West”, respectively. You can also terminate the schedule in “East” or “West” and start another schedule with start and destination key, that directs the train from “East“/“West” to “South 1” or “South 2”.

Example: Manual Control of Station Exit

The exit of the hidden yard will be controlled manually in the following way: it will be possible to select the train, which is to be started by a schedule, by selecting the track, from which the train will start.

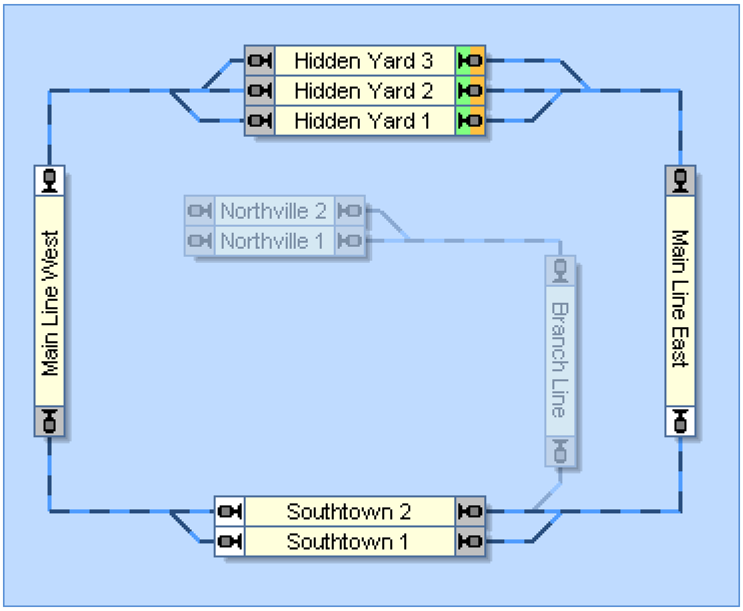


Diagram 178: Manual Control of Station Exit

The most simple solution is to assign an *operation*, that starts the schedule, to the operations of all routes, that connect the blocks in “Hidden Yard” with the blocks “Main Line East” or “Main Line West”, respectively. Details about operations can be found in section 14.4, “Operations". Instead of starting the schedule directly we activate the route. Through the operations of the route the schedule is started after activation of the route. This configuration additionally uses a trick, that was explained on page 194. When selecting one or several alternatives to start or continue a schedule **TrainController™** prefers to select an alternative that prefers an already activated route. Since we activate the desired route first before the schedule is started, the activated route will be selected as the route to be used by the schedule and the train waiting in the block next to this route will be started.

An alternative solution uses macros (see section 14.8, “Macros”) and the possibility to lock the exit of each block (see page 156). For each track and each possible path a separate macro is defined. Additionally, the exits of each block in “Hidden Yard” are held locked by default. This can be done by manually locking all exits manually at the beginning. Appropriate operations are assigned to the macro, that first remove the lock from the exit of the related block and then start the schedule. For example, the macro, that controls the exit from “Hidden Yard 2” to the right, executes removing the exit lock of “Hidden Yard 2” to the right plus start of the schedule. In the same way the other macros are configured, one macro for each track and each side of the “Hidden Yard”. With an appropriate operation, that restores the lock to the exits and executed by the schedule, when the block is released (see page 196), we can ensure that the exit lock of the start block is restored to the default.

In the first case a route is started instead of the schedule and in the second case a macro. In either case the schedule is indirectly started by the route or the macro, respectively, by their operations. The actions performed before the schedule is actually started ensure, that the right train leaves the “Hidden Yard”.

In both cases we can additionally trigger the route or the macro, respectively, by appropriate start- and destination keys (see page 309) from a switchboard or an external control panel. This provides the possibility to select the train to be started from such panel, too.

Example: Hidden Yard with Train Length Control and Automatic Bypass

The hidden yard displayed below will be operated automatically in the following way:

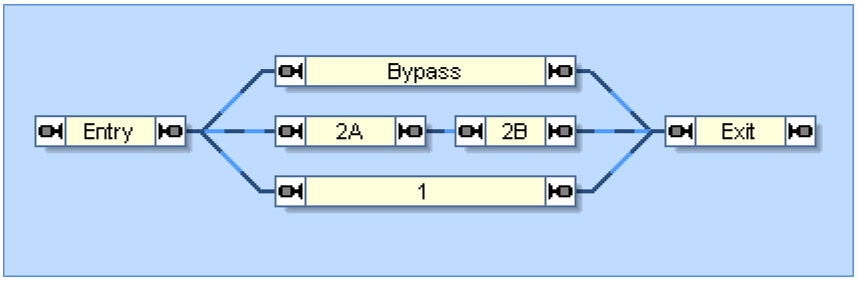


Diagram 179: Hidden Yard with Train Length Control and Bypass

- Trains enter the hidden yard through block “Entry” on the left and leave the yard through block “Exit” on the right.
- Long trains should go to block 1, if this track is available. If track 1 is already occupied, then long trains will bypass the hidden yard via block “Bypass” and leave the hidden yard at once. Long trains must not enter track 2.
- It is assumed that two short trains fit into track 2. Short trains will stop in block “2 B”, if track 2 is available. If there is already a train that is occupying “B 2”, then the next short train will enter track 2, too, and stop in block “2 A”. If there are already two trains waiting in track 2 then the next short train will go to track 1. If both tracks 1 and 2 are completely filled then a short train will bypass the hidden yard via block “Bypass” and leave the hidden yard without stopping.
- When a short train that has been waiting in “2 B” leaves the yard then another short train waiting in “2 A”, if any, will automatically move up to “2 B”.

The following schedules are created:

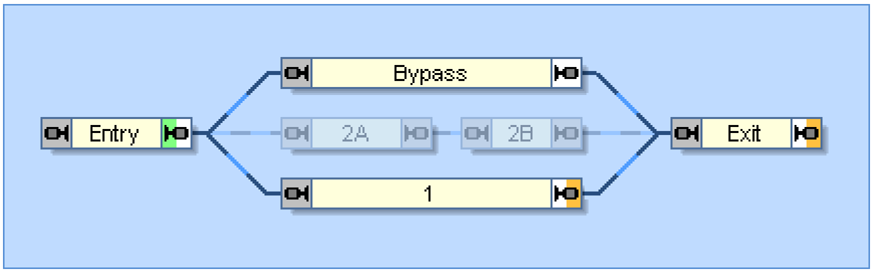


Diagram 180: Schedule for long Trains “Entry Long Trains“

The schedule to describe the entry of long trains is displayed in Diagram 180. The start block of this schedule is block “Entry”. The destination blocks are alternatively block “1” or block “Exit” via “Bypass”.

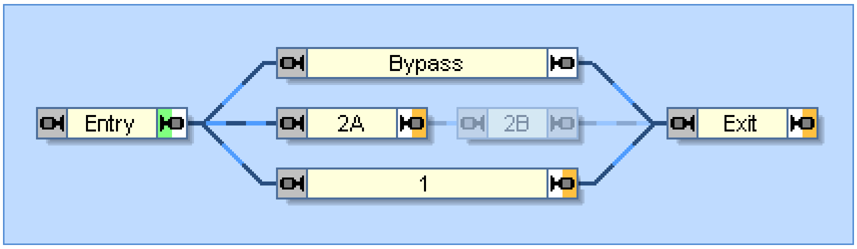


Diagram 181: Schedule for short Trains “Entry Short Trains“

The schedule for short trains is displayed in Diagram 181. The start block of this schedule is block “Entry” again. The destination blocks are alternatively block “2A”, block “1” or block “Exit” via “Bypass”.

The conditions listed in the following table ensure that the tracks are filled as required:

Schedule	Block	Condition	Remark
Entry Long Trains	Bybass	Block 1	May go to “Bypass”, only if block “1” is in use
Entry Short Trains	Block 1	Block 2A	May go to “Block 1”, only if block “2A” is in use
	Bypass	Block 1 **AND** Block 2A	May go to “Bypass”, only if block “1” and block “2A” are in use

Table 16: Conditions of Block Reservation

The schedule, that controls the exit of waiting trains, is shown below:

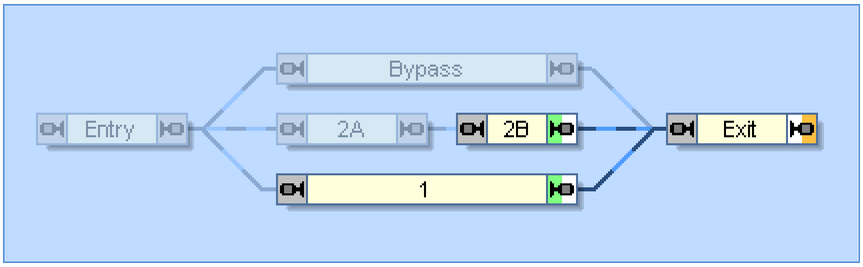


Diagram 182: Schedule “Exit“

For moving up trains from block “2A” to “2B” we need another schedule:

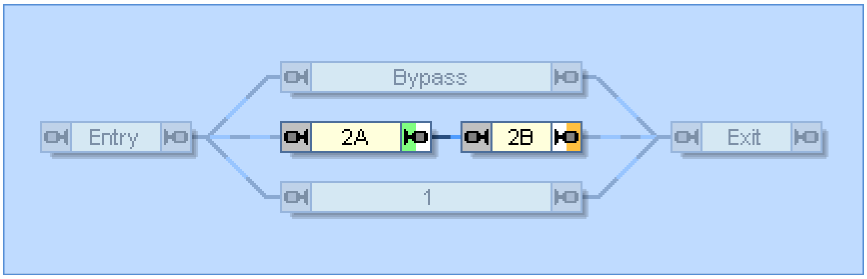


Diagram 183: Schedule “2A to 2B“

This schedule can be linked to schedule “Exit” as a successor. In this way each train that leaves the hidden yard controlled by schedule “Exit” will try to execute schedule “2A to 2B”, when it arrives in block “Exit”. If there is a train waiting in “2A” and if “2B” is free in this moment then this train will proceed to “2B”.

The two entry schedules can be linked as successors to other schedules that move a train from anywhere to block “Entry”. In the same way other schedules, that move a train from block “Exit” to anywhere can be linked as successors to schedule “Exit”. In this way the small configuration shown here can be integrated into the operation of a complete layout.

In **TrainController™** **Gold** extended train guidance based on train length (see page 344) can be used to direct short and long trains to different tracks. We didn’t apply these possibilities here to illustrate an example, that will work in **TrainController™** **Silver**, too.

Example: Optimal Length Control for Hidden Yards

Trains entering a hidden yard shall always be directed to shortest possible block into which they just fit.

At first, the lengths of all vehicles and the maximum train lengths of the blocks in the hidden yard are entered into the program.

The simplest solution is now to mark the rule **Prefer shortest destination block** in all schedules, that end in the hidden yard. Additionally, the rule **Trains must fit into destination blocks** should be set, too, to ensure that all used destination blocks are actually long enough.

This works for simple hidden yards where all tracks are located equally side by side.

But if e.g. two hidden yards are located behind the other, and additional sections (blocks or routes) must be passed to reach the second yard, then trains are directed to free, long-enough tracks of the first yard. The rule **Prefer shortest destination block** is so weak that it cannot overcome the greater distance to the second yard. In such a case it makes sense to enable the rule **Enforce shortest destination block**. This ensures in any case that always the shortest appropriate destination block is targeted, no matter how long the distance is, or what are the obstacles on the way.

But this rule is problematic in cases where the only best fitting destination block is already used by another train. The entering train would move to this block anyway, and wait before its entry, until the destination block becomes free.

To handle this case also optimally, two schedules are created, which end in <u>both</u> hidden yards. These schedules are entered as the successors of the schedules that terminate before the hidden yard. The following rules are set in these two schedules:

Schedule 1:
- **Enforce shortest destination block**
- **Reserve destination block at start**
- **Train may stay in start block** must <u>not</u> be set.

Schedule 2:
- **Prefer shortest destination block**
- **Trains must fit into destination blocks**

These two schedules are listed in this order as successors in other schedules with the option **By Order**.

During operation, the software tries to start schedule 1 first for each arriving train. This schedule searches the best matching destination block in one of the two hidden yards and reserves this block at once. If this is not possible, e.g. because this destination block is already in use by another train, then the train must not proceed. Since the rule **Train may stay in start block** is not set, schedule 1 fails. Therefore, now schedule 2 is started and another suitable destination block is selected.

The order of the schedules ensures that a non-optimal matching destination block is approached only when really no optimal fitting destination block is available in both hidden yards.

15.7 Stations

General

Stations can be used as an optional feature to control the traffic on your layout and the aspects of calculated signals for shunting moves.

Stations can control the traffic flow by preventing trains from the execution or termination of schedules in a station in certain situations. Furthermore stations can impair the calculation of internal block signal aspects for schedules, which are limited to the area of a single station (shunting moves).

Stations are established by assigning blocks and routes to them. This is done in a similar way like assigning blocks and routes to schedules. Each block or route can only be assigned to at most one station. It is not possible to assign a block or route to two stations or more. A route, however, can connect two blocks, which are located in different stations.

Stations are displayed in the station view of the Dispatcher Window (see section 5.18).

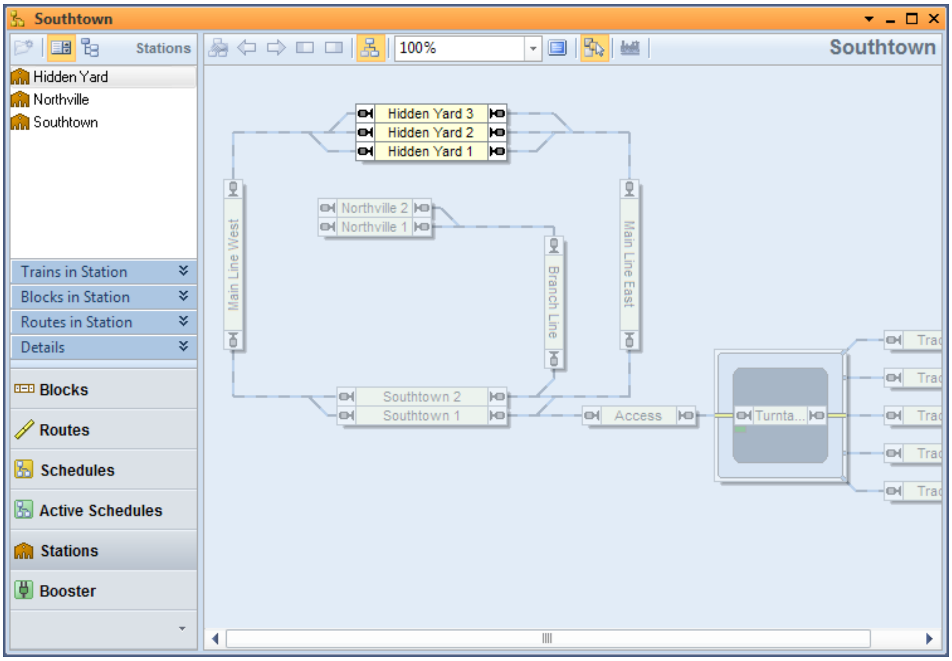


Diagram 184: Station View in the Dispatcher Window

Diagram 156 shows a layout with three stations. The station "Hidden Yard" is currently selected and the station diagram of "Hidden Yard" is currently visible in the right part of the dispatcher window. The blocks "Hidden Yard 1", "Hidden Yard 2" and "Hidden Yard 3" are located in station "Hidden Yard".

Minimum and Maximum Number of Trains

For each station it is possible to specify a minimum and maximum number of trains. This allows to control the start and termination of schedules in a station.

The minimum and maximum number of trains are related to inactive trains, i.e. to trains not under control of a schedule.

If the number of inactive trains currently located in the station is smaller or equal than the minimum number of trains, then no schedule can be started with one of these trains. If the number of inactive trains currently located in the station is greater or equal than the maximum number of trains, then no schedule can be terminated in this station.

Note, that passing of trains through a station is not affected by these numbers. That means: a train may always pass a station – regardless of the values of these numbers.

Conditions

For each station two conditions can be specified, one for the start of schedules in this station and one for the termination of schedules in this station.

If the condition to start schedules in this station is not true, then no schedule can be executed with a block located in this station as start block.

If the condition to terminate schedules in this station is not true, then no schedule can be executed with a block located in this station as destination block.

Note, that passing of trains through a station is not affected by these conditions. That means: a train may always pass a station – regardless whether these conditions are true or not.

Stations, Trains and Schedules

It is also possible to assign trains and schedules to a station.

If trains are assigned to a station, then only these trains may start or terminate a schedule in this station.

If schedules are assigned to a station, then only these schedules may be started or terminated in this station.

Local Schedules

A local schedule is a schedule, which contains only blocks located in the same station as well as blocks not located in a station, which are directly connected to blocks in this station by routes. In other words: a local schedule contains only blocks located in the same station or directly adjacent to the same station.

A schedule with blocks located in different stations is no local schedule. This is a schedule, which connects different stations.

A schedule with routes, which connect two blocks not contained in a station, is no local schedule. This is a schedule with routes on a main track.

Trains controlled by a local schedule can only move within one station or into blocks adjacent to this station. Typical examples are switching moves.

Regular schedules are made implicitly local or non-local by their schedule diagrams, i.e. by the blocks and routes contained in the schedule and the stations, where these blocks and routes are located in.

A schedule established by AutoTrain (see section 5.13) can be forced to be local with a specific option. Such AutoTrain run will only contain those blocks and routes, which result in a local schedule according to the above description. This option is useful to force an AutoTrain run to be limited to a certain station.

Spontaneous runs can be forced to be local, too, by applying a specific rule for spontaneous runs to selected trains (see page 359).

Local Schedules and calculated Signals

Local schedules are often used to perform shunting moves. In the real world signals are often set differently for shunting moves on one side and trips from one station to another on the other side. For these reasons **TrainController™ Gold** provides a set of schedule rules, with which the internally calculated block signals can be specifically affected for local schedules (see page 357).

15.8 Booster

General

Boosters can be used as an optional feature to establish a booster management for your layout. This booster management allows – among others – to automatically stop trains, to prevent trains from starting, or to lock and release certain areas of the layout depending on the load put on the track or status of physical boosters.

Boosters are established by assigning blocks and routes to them. This is done in a similar way like assigning blocks and routes to schedules. Each block or route can only be assigned to at most one booster. It is not possible to assign a block or route to two boosters or more at the same time. A route, however, can connect two blocks, which are assigned to different boosters.

Boosters are displayed in the booster view of the Dispatcher Window (see section 5.18).

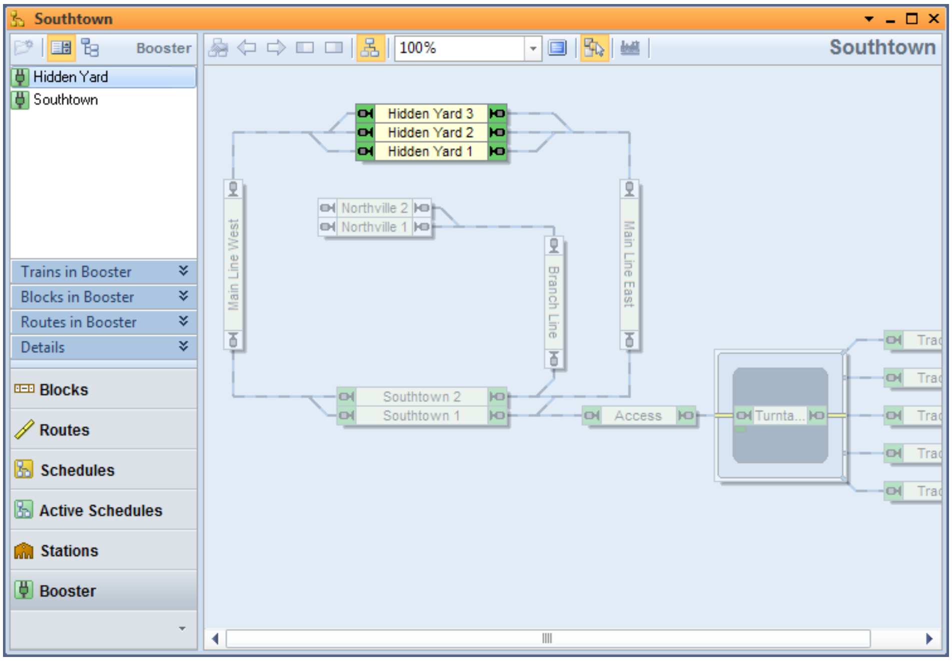


Diagram 185: Booster View in the Dispatcher Window

Diagram 185 shows a layout with two boosters. The booster "Hidden Yard" is currently selected and the block diagram of "Hidden Yard" is currently visible in the right part of the dispatcher window. The blocks "Hidden Yard 1", "Hidden Yard 2" and "Hidden Yard 3" are assigned to booster "Hidden Yard".

Boosters can be used to control the traffic on the model railroad as a function of the load currently put on the tracks or the number of trains currently busy in certain areas of the layout. In many cases the booster management is established to prevent physical boosters from being overloaded. For this purpose booster objects in **TrainController™** are usually associated with physical boosters connected to the track. It is also possible, however, to establish a booster management for traffic control based on virtual boosters without physical counterpart.

States of a Booster

Each booster provides the following states:

Symbol	State	Meaning
🟩	On	The booster is turned on. No problems exist in the area of the booster. This is the normal state.
⬜	Off	The booster is turned off. The traffic is currently stopped in the area of the booster.
🟨	Warning	The booster is turned on. Certain action may be required to avoid subsequent problems.
🟥	Error	A severe problem occurred in the area of the booster (e.g. short circuit or overload of the associated physical booster, too many active trains, etc.)
☒	Disabled	All features of the booster object are disabled. The booster object does not impair the traffic in the area associated with it.

Table 17: States of a Booster

The colors associated with each booster state are also displayed in the exit boxes of the blocks in the block diagram of this booster (see Diagram 185).

Rules

In **TrainController™ Gold**, the effect of each booster can be customized to individual needs with a variety of rules.

The rules are divided into the following categories:

Off State:

This category includes rules that specify the behavior of the booster object, when it is turned off:

- **Freeze Trains:**
  All trains located in blocks associated with the booster object are frozen, when the booster is turned off. Frozen trains are stopped. Schedules, which control these trains, remain active.

- **Stop Trains and Schedules:**
  All trains located in blocks associated with the booster object are stopped, when the booster is turned off. Schedules, which control these trains, are terminated.

- **Lock Exits:**
  The exits of all blocks associated with the booster object are locked, when the booster is turned off. This causes all trains in the booster area to stop, when they reach the next stop marker in their current block (unless the physical booster associated with the booster object cuts the power). Trains, that are currently stopped cannot be put in motion by schedules.

- **Lock Entries:**
  All entries of blocks located at the boundary of the booster area are locked, when the booster is turned off. This prevents trains currently not located in the booster area to enter the booster area.

Error State:

This category includes rules that specify the behavior of the booster object, when it is in the error state:

- **Freeze Trains:**
  All trains located in blocks associated with the booster object are frozen, when the booster changes to the error state. Frozen trains are stopped. Schedules, which control these trains, remain active.

- **Stop Trains and Schedules:**
  All trains located in blocks associated with the booster object are stopped, when the booster changes to the error state. Schedules, which control these trains, are terminated.

- **Lock Exits:**
  The exits of all blocks associated with the booster object are locked, when the booster changes to the error state. This causes all trains in the booster area to stop, when they reach the next stop marker in their current block (unless the physical booster associated with the booster object cuts the power). Trains, that are currently stopped cannot be put in motion by schedules.

- **Lock Entries:**
  All entries of blocks located at the boundary of the booster area are locked, when the booster changes to the error state. This prevents trains currently not located in the booster area to enter the booster area.

- **Off:**
  The booster is turned off, when it changes to the error state.

Warning State:

This category includes rules that specify the behavior of the booster object, when it is in the warning state:

- **Keep Frozen Trains:**
  Trains frozen in the error state are not released. This option is useful to prevent the booster from changing back to the error state right after changing from the error to the warning state.

- **Lock Exits:**
  The exits of all blocks associated with the booster object are locked, when the booster changes to the warning state. This causes all trains in the booster area to stop, when they reach the next stop marker in their current block. Trains, that are currently stopped cannot be put in motion by schedules.

- **Lock Entries:**
  All entries of blocks located at the boundary of the booster area are locked, when the booster changes to the warning state. This prevents trains currently not located in the booster area to enter the booster area.

- **Do not execute Schedules:**
  Do not execute any additional schedules with trains located in the booster area, while the booster is in warning state.

Threshold Values:

This category includes threshold values, which causes the booster to change to the error or warning state, when the actual value reaches the threshold value:

- **Running Trains:**
  The warning or error state, respectively, is turned on, when the number of trains running in the booster area at non-zero speed, is greater or equal than the specified threshold value. This option is useful to prevent the associated physical booster from being overloaded, when no other information from the booster (such as current or temperature) is available.

- **Current:**
  The warning or error state, respectively, is turned on, when the current of the track output of the associated physical booster is greater or equal the specified threshold value.

- **Temperature:**
  The warning or error state, respectively, is turned on, when the temperature of the associated physical booster is greater or equal the specified threshold value.

Physical Connections of a Booster

In order to implement a booster management in conjunction with physical boosters, the following addresses can be specified for each booster object:

- Booster address: this address allows to associate the booster object in **TrainController™** directly with a physical booster connected to the layout. This provides the possibility to turn on or off the physical booster or to determine values for the current or temperature of the physical booster.
  The availability of certain features depends on the type of the physical booster. Further information can be found in the Help menu of **TrainController™**.

- Turnout commands: for the on and the off state it is possible to specify an individual turnout command (turnout address and state). This command is sent to the specified turnout address, when the booster object is turned on or off. This option is useful for those physical boosters, which provide the possibility to turn the track power on or off from the distance via a turnout command.

- Feedback addresses and state: For each state of a booster object (on, off, error and warning) it is possible to specify an individual feedback address and state. This causes the booster object to change to the according state each time, when the specified feedback contact changes to the specified state. This option is useful for those physical boosters, which report their status via regular feedback events.

Trigger

For each state of a booster object (on, off, error and warning) it is possible to specify an individual trigger. This causes the booster object to change to the according state each time, when the trigger specified for this state becomes true.

This feature is useful for those physical boosters, which report their state with more complex information than plain feedback events.

This feature can also be used to implement a virtual booster management without physical boosters (see below).

Virtual Booster Management

TrainController™ provides the possibility to establish a booster management for traffic control without the existence of physical boosters, too, or where there is no possibility to specify a connection to a physical booster.

Such virtual booster management can be accomplished by specifying a trigger for each state of a booster object. In this way it is possible to automatically turn a booster object to error state and to stop the traffic in a certain area of the layout, for example, when a certain logical condition in **TrainController**™ becomes true.

The option to change the booster to warning or error state, when the number of active trains in the booster area reaches or exceeds a certain threshold value, and to stop the traffic in this area subsequently, is another way to control the traffic on your model railroad layout, which can be used without the existence of physical boosters or without a connection to them.

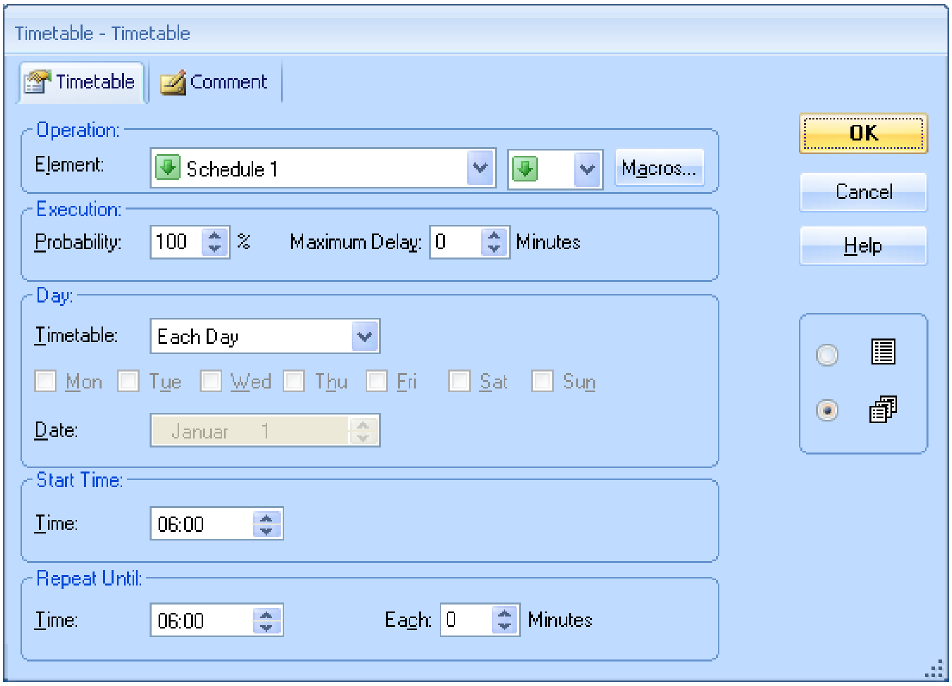
Boosters and other Objects

Each state of a booster object can be evaluated in the trigger or condition of other objects. Furthermore booster objects can perform operations, when they change their state. These features provide virtually unlimited possibilities to integrate booster objects into the automatic control of your model railroad layout and to accomplish a smart booster management or traffic control.

16 Timetables

It is possible to execute *schedules* or *macros* (see section 14.8, “Macros”) at specific times. Using a time table entry you can specify, on which days or at which times a schedule or macro will be executed.

Schedules can be started daily, on specific days of the week or at a specific date, as desired. The latter feature enables creation of up to 365 different timetables. In this case the valid timetable is selected by setting the date of the *Clock*.



Operation:
Element: Schedule 1

Execution:
Probability: 100 %   Maximum Delay: 0 Minutes

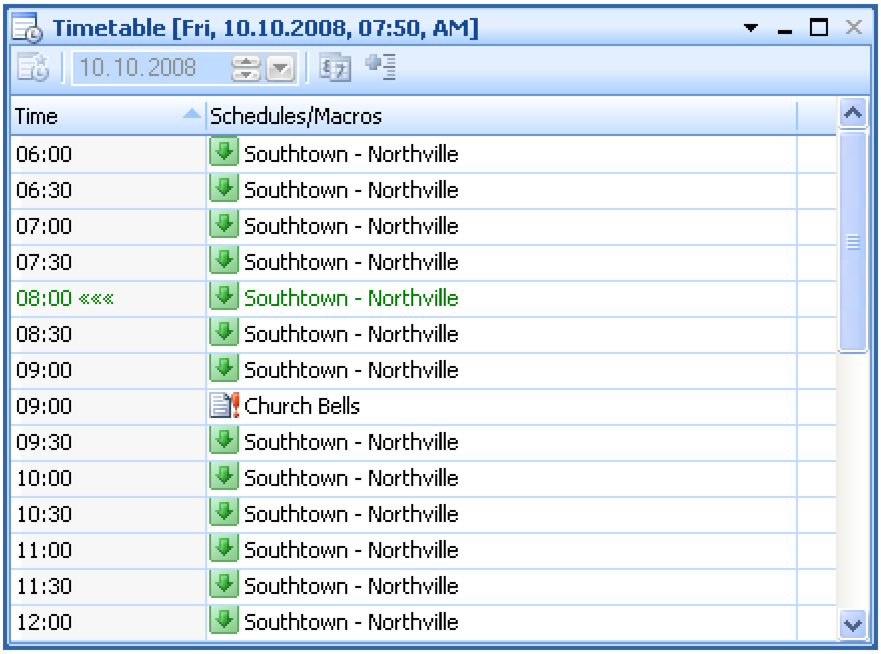
Day:
Timetable: Each Day
Mon Tue Wed Thu Fri Sat Sun
Date: Januar 1

Start Time:
Time: 06:00

Repeat Until:
Time: 06:00   Each: 0 Minutes

Diagram 186: Specifying the start time of a Schedule

Using these specifications **TrainController™** creates a *timetable* for the current day. The current day is determined by the date, which is currently displayed by the *Clock* (see chapter 13, ” The Clock”). **TrainController™** starts the particular schedules or macros dependent to the time currently displayed by the *Clock*.



Timetable [Fri, 10.10.2008, 07:50, AM]

10.10.2008

Time	Schedules/Macros
06:00	Southtown - Northville
06:30	Southtown - Northville
07:00	Southtown - Northville
07:30	Southtown - Northville
08:00 «««	Southtown - Northville
08:30	Southtown - Northville
09:00	Southtown - Northville
09:00	Church Bells
09:30	Southtown - Northville
10:00	Southtown - Northville
10:30	Southtown - Northville
11:00	Southtown - Northville
11:30	Southtown - Northville
12:00	Southtown - Northville

Diagram 187: Timetable Window

Using macros in timetables allow interesting effects. It is for example possible to turn on or off the lights on the model railroad or to play sound files at certain times.

The additional features to execute timetable entries only by chance or to insert random delays provide even more variety.

17 Turntables and Transfer Tables

17.1 Introduction

Turntables and *transfer tables* are used in **TrainController™** to operate real turntables and transfer tables on your model railroad with the computer. In this document the term "turntable" is mostly used synonymously for both, turntables and transfer tables.

TrainController™ provides a separate turntable window, which provides a graphical representation of each turntable or transfer table and which allows manual operation of turntables.

Different turntable windows can be opened simultaneously to control several turntables/transfer tables at the same time. The number of turntable windows is only limited by the capacity of your computer.

Each turntable object can be configured to operate a turntable or to operate a transfer table as displayed below:

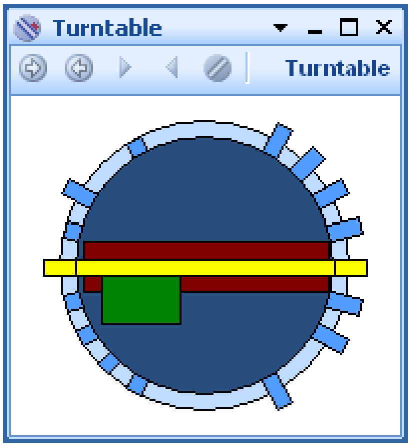


Diagram 188: Turntable Window

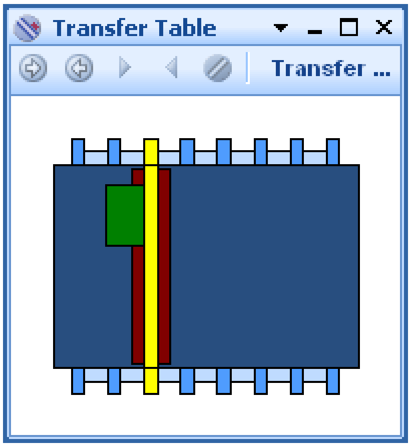


Diagram 189: Transfer Table

Special features are:

- up to 80 tracks on each turntable or transfer table
- each track can be individually configured as active or inactive as well as removed completely
- each turntable can be operated manually via the turntable window
- predefined software drivers for all leading turntable types
- generic turntables and transfer tables allow adaptation to custom driven devices
- each turntable/transfer table can be operated semi-automatically by operations of push buttons, macros, indicators or routes
- the operation of turntables and transfer tables can be easily integrated into schedules, **AutoTrain**™ or spontaneous runs

Supported Turntable/Transfer Table Commands

TrainController™ supports the following turntable/transfer table commands:

- permanent move in either direction
- stop of permanent move with automatic alignment to the next active track
- step to the next or previous active track
- direct selection of specific tracks *(indexing)*
- 180° turn (turntables only)
- dedicated adjustment of locomotive direction during automatic operation (turntables only, see page 391)

Integrating Turntables into the Switchboard and the Operation of the Layout

In **TrainController™** **Gold** turntables are created by inserting a turntable symbol at an appropriate location in a switchboard. This symbol allows to operate the turntable with the mouse via the switchboard. It is also optionally possible to open one or more *turntable windows* via the **Window** tab of **TrainController™**. The turntable symbol in the switchboard is optimised for space-saving display. For this reason inactive tracks, i.e. tracks without connection to the layout, are not displayed by turntable symbols in the switchboard. Because the symbol has furthermore to fit into the arrangement of switchboard cells, the layout of turntable symbols is compulsorily a bit schematic. The display in the turntable window on the other hand is more realistic. Both views can be used alternatively or simultaneously for manual operation and control of turntables.

The turntable symbol in the switchboard provides some advantages, however, that are not provided by the stand-alone turntable window:

- the turntable symbol integrates more smoothly into manual switchboard operation than manual operation via a separate window
- the turntable symbol visualises the linkage to adjacent blocks
- the turntable symbol supports simple integration of the turntable into automatic operation of the layout, because turntable symbols are taken into account by the automatic calculation of the block diagram (see section 5.2, “Blocks and Routes”). All possible paths from the bridge to adjacent blocks or back are automatically captured as routes
- turntable symbols are also visible in the associated calculated block diagrams
- since turntable symbols are usually associated with blocks, they can also display, which train is currently located on the bridge and how the train is oriented.

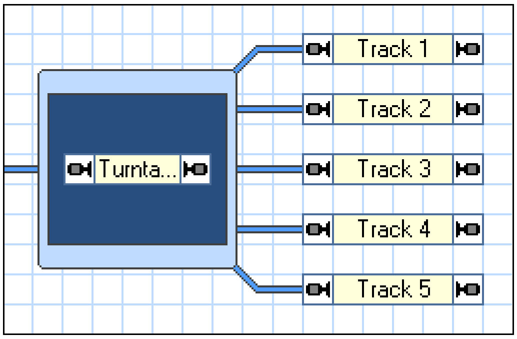


Diagram 190: Turntable Symbol in the Switchboard

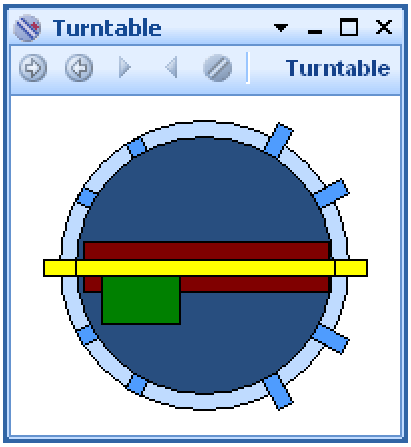


Diagram 191: Corresponding Turntable Window

Arranging the Layout of the Turntable Symbol in the Switchboard

In **TrainController™ Gold**, it is possible, to arrange the color design of the turntable symbol in the switchboard to your own taste.

It is also possible to influence the content of the indication. Inside the turntable symbol the following can be displayed:

- only the block (as in version 7)
- a thumbnail image of the turntable or transfer table as in the turntable window
- or both.

The content can also be designed differently, depending on whether the bridge is moving or not. It is e.g. possible to display the thumbnail view during the movement of the bridge to track the change in position visually. While at standstill the block is visible to indicate the orientation of vehicles on the bridge.

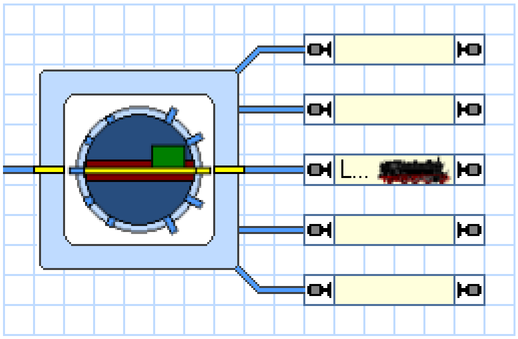


Diagram 192: Turntable Symbol with thumbnail Image of the Turntable

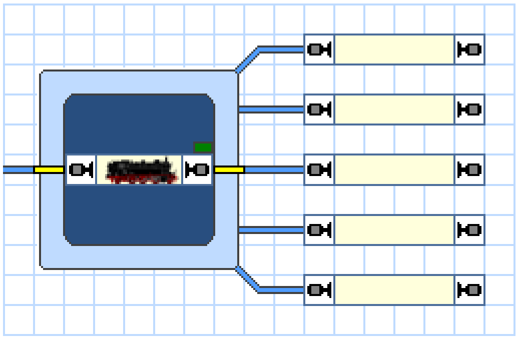


Diagram 193: Turntable Symbol with displayed Block

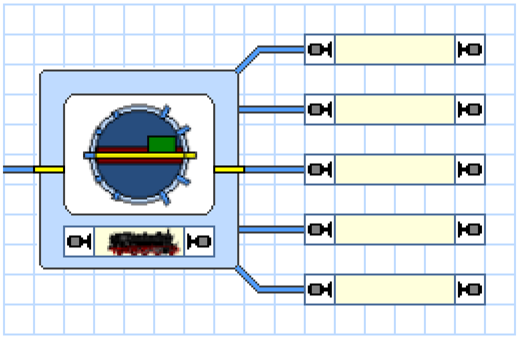
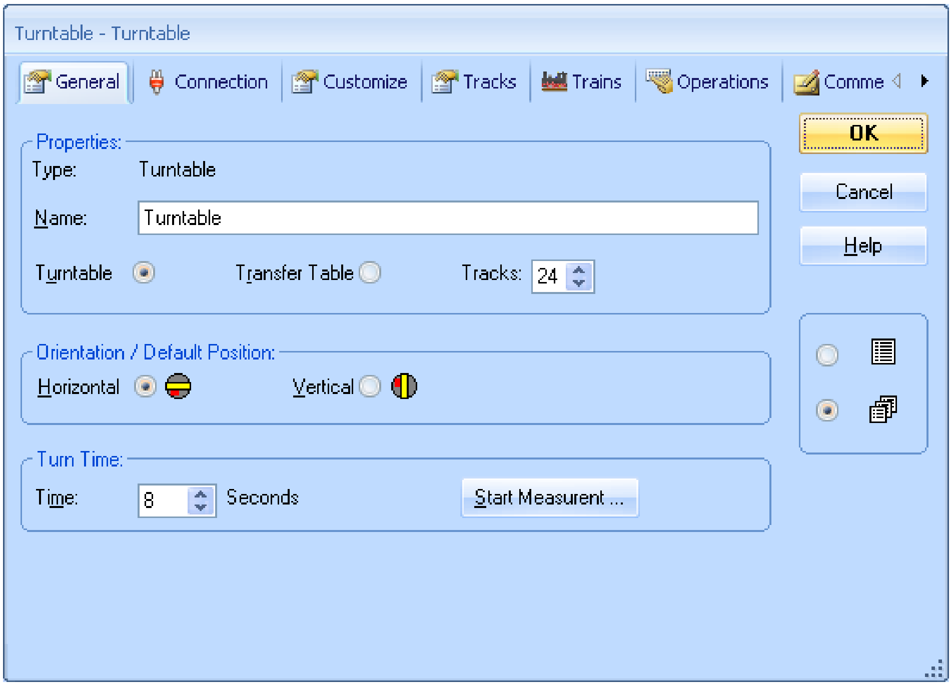


Diagram 194: Turntable Symbol with thumbnail Image and Block

17.2 Configuring a Turntable or Transfer Table

To configure a turntable or transfer table use the **Properties** command of the **Edit** tab. Next, select if you want to control a turntable or transfer table and how many tracks can be connected to the turntable/transfer table.

Additionally you can specify a **name** for the turntable/transfer table This is useful in identifying the turntable/transfer table when it is referred to later. By specifying or measuring the **turn time** of the turntable/transfer table, you can assure that the movement of the bridge on the computer screen is synchronized with the movement of the actual bridge on your layout.


Turntable - Turntable

Properties:
Type: Turntable
Name: Turntable
Turntable [✓] Transfer Table Tracks: 24

Orientation / Default Position:
Horizontal [✓] Vertical

Turn Time:
Time: 8 Seconds Start Measurent ...

Diagram 195: Specifying general properties of a turntable

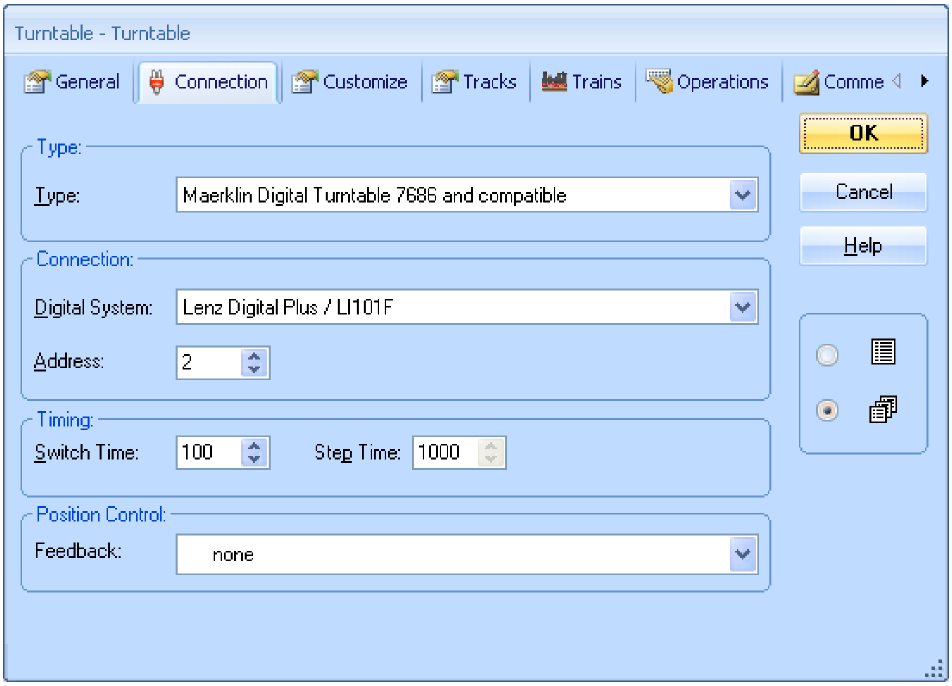
17.3 The Type of a Turntable/Transfer Table

Digital Turntable

A turntable is called a *digital turntable* if it is driven by a (built-in) digital turntable decoder. Examples of digital turntables are

- Marklin Digital Turntable 7686 and compatibles
- Marklin Turntable 7286 with digital turntable decoder 7687
- Turntable driven by the Digital Turntable Decoder Rautenhaus SLX815

! **Digital turntables support all commands listed on page 380. Specifically, they support the direct selection of specific tracks (*indexing*). Since indexing is vital for automatic operation digital turntables can be operated automatically without any limitations or special measures to be taken.**



Type:

Type: Maerklin Digital Turntable 7686 and compatible

Connection:

Digital System: Lenz Digital Plus / LI101F

Address: 2

Timing:

Switch Time: 100 Step Time: 1000

Position Control:

Feedback: none

Diagram 196: Specifying the type and digital address of a turntable

A digital address must be specified for each digital turntable. This is the digital address of the digital turntable decoder.

Analog Turntables/Transfer Tables

A turntable/transfer table is called an *analog turntable/transfer table* if it supports the following limited subset of turntable commands:

- permanent move in either direction
- stop of permanent move

Examples of analog turntables/transfer tables are:

- Marklin Turntable 7186
- Marklin Transfer Table 7294

The turntables/transfer tables listed above are not intended by the manufacturer to be operated by a digital system. Nevertheless, they can be made controllable by the computer. In this case, they must be wired to accessory decoders and, optionally, latching relays. When wired in this way, they are accessed via a digital turnout address. For details about digital addresses and wiring diagrams for the particular analog turntable/transfer table types, refer to the **Help** menu of **TrainController™**, please.

! **Analog turntables do usually not support indexing and cannot be used for automatic operation without further measures.**

It is also possible to configure analog turntables/transfer tables for support of indexing, too. In this way, it is possible to upgrade an analog turntable to a (pseudo-)digital turntable by means of the software. If this is done they can be used for automatic operation like digital turntables. For further details refer to section 17.6, "Turntable Operations", please.

Generic Turntables

Generic turntables/transfer tables are all turntables/transfer tables, that are not explicitly listed as devices supported by **TrainController™**. An example is a home- made turntable driven by custom hardware.

Generic turntables are not associated with a certain digital address. Instead they are only able to perform certain operations when one of the turntable/transfer table commands listed on page 380 is given. If no operation is specified for a certain command then a generic turntable/transfer table does nothing when this command is given.

Usually you will assign the operation of push buttons, on-off switches or toggle switches located anywhere in one of your Switchboards to a turntable/transfer table command. In this way the associated element is operated when the command is given. The associated element again can then operate the actual turntable on the model railroad layout accordingly.

Generic turntables can be setup to operate like analog turntables and – if operations for indexing are added as well – even like digital turntables.
For further details about the operations assigned to a generic turntable refer to section 17.6, "Turntable Operations", please.

17.4 Automatic Operation of Turntables

! **Please note, that the turntable/transfer table must be able to go to specific tracks (indexing), if requested. If you are using an analog or generic turntable/transfer table, then setup this turntable for indexing according to section 17.6, “Turntable Operations”. Digital turntables support indexing and no further measures are necessary.**

Automatic Operation in TrainController™ Gold

In **TrainController™ Gold** turntables and transfer tables can be easily integrated into automatic operation by using turntable symbols in switchboards. These symbols are associated with a block and taken into account by the automatic calculation of the block diagram (see section 5.2, “Blocks and Routes”. All possible paths from the bridge to adjacent blocks or back are automatically captured as routes.

Right after configuring a turntable symbol accordingly in a switchboard it is possible to run trains automatically over the turntable. The routes, which connect the bridge (more precisely: the block associated with the turntable) with adjacent blocks, can be used by AutoTrain or in other schedules like any other route. Actually with regard to automatic operation there is no fundamental difference between turntables and turnouts.

For each track it is possible to specify, that certain locomotives may leave the bridge via this track only with a certain orientation (forward or backward). In this way it is for example possible to force steam locomotives to enter the roundhouse only with a certain orientation, while Diesel or electrical locomotives may still enter the roundhouse with an arbitrary orientation.

17.5 The Track Layout of a Turntable/Transfer Table

Active and Passive Tracks of Turntables

Each physical turntable or turntable decoder, respectively, can support a maximum number of track exits or tracks. The maximum number of tracks of the Maerklin digital turntable 7686, for instance, is 48. Usually only a fraction of the possible tracks are actually used.

The used tracks are divided into active tracks and passive tracks.

Active tracks correspond to those track exits of the turntable, which are connected to existing tracks of the layout. Engines can enter and leave the turntable via active tracks.

Passive tracks correspond to those track exits of the turntable, where the bridge of the turntable can be turned to, but which are <u>not</u> connected to existing tracks of the layout. In many cases there is only a short stub track associated with a passive track. Engines cannot enter and leave the turntable via passive tracks.

Diagram 191, for instance, shows a turntable with 6 active and 4 passive tracks. The total number of active and passive tracks must be always even.

Note, that all active <u>and</u> passive tracks are usually important in conjunction with physical control of the turntable bridge and the turntable decoder. The decoder does not care, whether an engine can leave the bridge via a certain track exit or not. For this reason the difference between active and passive tracks is irrelevant for the decoder. But the bridge must be able to turn the house to each existing track exit, regardless whether the exit is passive or not. In Diagram 191, for instance, there are 10 track positions, where the house of the bridge can go and thus all 10 positions, i.e. the number of active <u>and</u> passive tracks, must be programmed into the decoder, if any, as different positions.

Synchronizing the Turntable Symbol

The turntable symbol in the switchboard of **TrainController™** **Gold** only displays active track exits. In the switchboard it is important to save place and to visualize, how the turntable tracks are connected to the circumjacent track layout. For this reason the passive tracks, which do not have a connection to the track layout, are not displayed by the turntable symbol in the switchboard.

In order to work properly the turntable symbol in the switchboard must be synchronised with the track layout of the physical turntable.

Diagram 197 illustrates, how this is done:

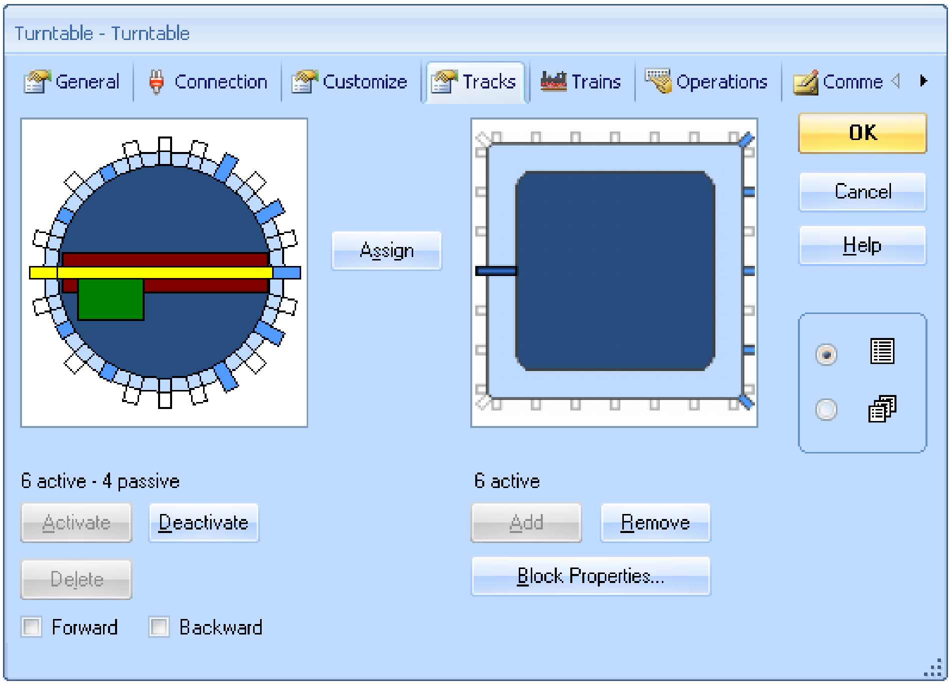


Diagram 197: Synchronizing the Turntable Symbol

The left image in Diagram 197 represents the track layout of the physical turntable. It has 6 active and 4 passive tracks, 10 significant track positions in total. The right image shows the schematic track layout of the turntable symbol in the switchboard. The number of active tracks must be identical in both images. Passive track exits of the physical turntable, that can be addressed with the bridge or the turntable decoder, respectively, but do not have a track connection to the rest of the layout, are not displayed and not taken into account by the schematic turntable symbol displayed in the switchboard. This is done to reduce the switchboard space required to display the turntable symbol.

To perform the synchronisation ensure first, that the number of active tracks in both windows is identical. Then select a track in the left image and a track in the right symbol, that will be mapped to each other. Then press **Assign**. The subsequent procedure automatically iterates clockwise through the active tracks of both symbols and automatically maps the tracks of the physical turntable displayed in the left image to appropriate tracks of the symbol to the right.

Forward and Backward Tracks of Turntables

Each active track of the physical turntable, i.e. those tracks, that are actually connected to the layout, can be marked as a *forward track* or *backward track*.

These markings are taken into account during automatic operation of a turntable and do not apply to transfer tables. If a track is marked as forward track, then all affected locomotives, that leave the turntable bridge via this track during automatic operation, are automatically turned with their head to this track, so that they leave the bridge in forward direction. If a track is marked as backward track, then locomotives are automatically turned with their rear to this track, i.e. they will leave the bridge via this track in backward direction.

It is furthermore possible to specify, which locomotives are affected by these markings. This is done by filling the associated list of enabled trains (see also page 257) accordingly. In this way it is for example possible to force steam locomotives to enter the roundhouse only with a certain orientation, while Diesel or electrical locomotives may still enter the roundhouse with an arbitrary orientation.

Turning Locomotives automatically to an individual Direction

If the turntable is operated manually, then the provided commands provide full control over the direction, in which locomotives are turned.

For automatic operation it is possible to mark each active track of the turntable as forward or backward track as outlined in the previous section. This setting is usually valid for all trains, that exit the turntable via the tracks, which are marked in such a way. This feature is useful for tracks, that will <u>always</u> be passed in a certain direction, e.g. if certain locomotives will enter a roundhouse always in forward direction. In particular these settings apply to all schedules, that contain the turntable, in the same way.

Tracks, which are not marked as forward or backward direction are usually accessed by the turntable bridge on the shortest possible way. For such tracks the direction, in which the locomotive leaves the turntable, cannot be predicted. It is sometimes desirable, however, to have control over the direction as the case arises. For this reason there is an additional option, with which the direction, in which locomotives leave the turntable, can be set on a per schedule basis. In this way trains, that perform a certain schedule, can be caused to leave the turntable in forward direction; and trains, that perform other schedules, can be caused to leave the turntable via the <u>same</u> tracks in backward direction.

The following priority scheme applies for all locomotives, that pass a turntable under control of a schedule:

- If a certain direction for the exit of the turntable is set in the schedule specific settings of the block, that belongs to the turntable, then all locomotives under control of this schedule leave the turntable in the specified direction. This setting applies to all tracks of the turntable.
- if the above does not apply then the locomotive leaves the turntable in the direction specified for the according turntable track.
- If no direction is specified neither for the schedule nor for the exit track, then this track is accessed by the turntable bridge on the shortest possible way. In this case it cannot be predicted, whether the locomotive leaves the bridge in forward or backward direction.

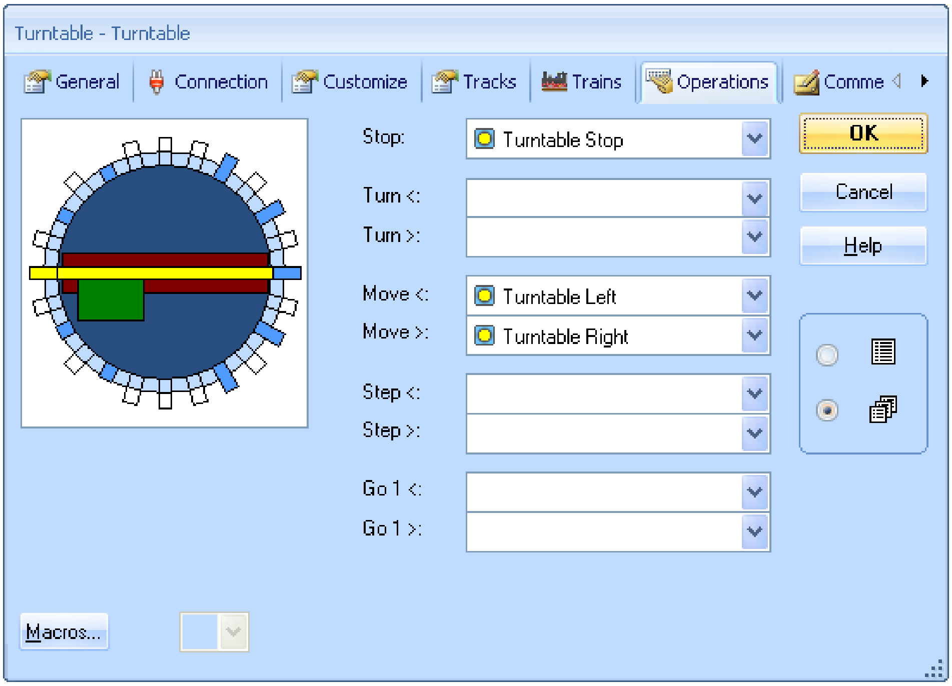
17.6 Turntable Operations

For each command listed on page 380 it is possible to specify a certain operation, that is executed when this command is given (see also section 14.4, “Operations”). These operations are mainly intended to be used for upgrading of analog turntables to operate like digital turntables. This is done by adding operations for *indexing*.

And these operations are used to setup a generic turntable/transfer table to operate like an analog or digital turntable.

They can be used additionally by digital turntables for special purposes as well, if desired.

Usually you will assign the operation of push buttons, on-off switches or toggle switches located anywhere in one of your Switchboards to a turntable/transfer table command. In this way, the associated element is operated when the command is given. The associated element can then operate the actual turntable on the model railroad layout accordingly, e.g. via relays wired to accessory decoders.



Turntable - Turntable

Stop: Turntable Stop
Turn <:
Turn >:
Move <: Turntable Left
Move >: Turntable Right
Step <:
Step >:
Go 1 <:
Go 1 >:

Diagram 198: Assigning operations of buttons to a turntable

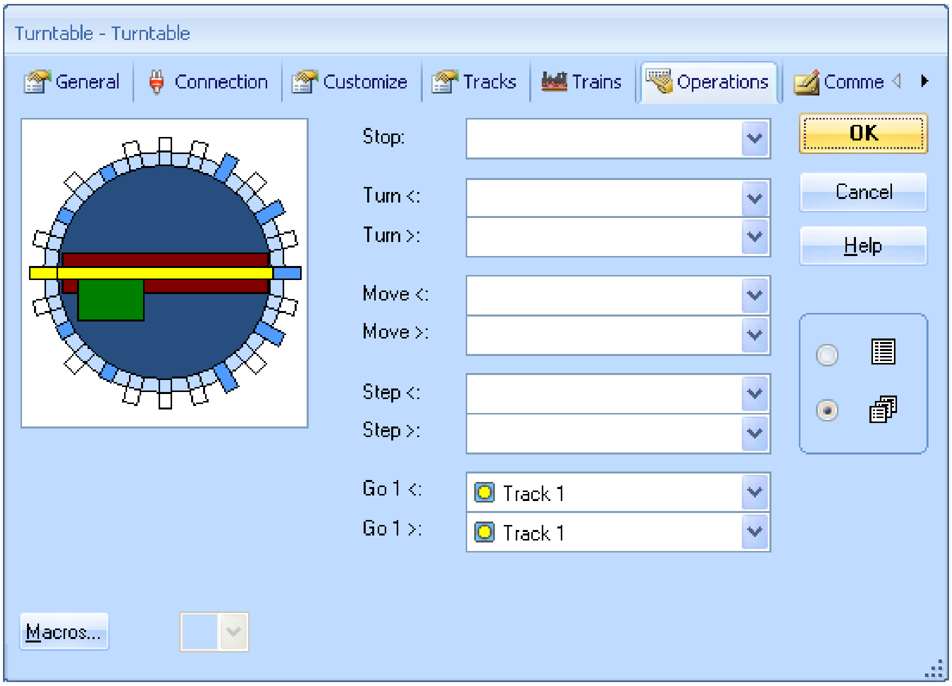
If, for example, operations are assigned to a generic turntable according to Diagram 198, then this turntable can be operated like an analog turntable. With this setup of operations, a generic turntable can perform exactly the same commands like an analog turntable.

It is also possible to operate macros. In conjunction with evaluation of indicator elements and restricted execution of operations (see 14.3, "Protection and Locking with Conditions"), it is even possible to setup *indexing* for analog or generic turntables/transfer tables. This is demonstrated in the following example.

Example: Indexing of an Analog Turntable

This example explains how an analog turntable, such as the Fleischmann Turntable or the Marklin Turntable 7186, can be setup for indexing in order to be controlled automatically. It is assumed that the track layout of the turntable is identical to Diagram 191. In the following, it is explained how indexing is setup for track 1. The setup for the other tracks is done accordingly.

- Wire the analog turntable according to the instructions in the **Help** menu of **TrainController™**.
- A Turntable Window is created and the track layout of the turntable is configured accordingly.
- Create a feedback indicator "Track 1" that is turned on, when the physical bridge of the turntable reaches the position of track 1. Of course you need an appropriate physical sensing device on your model railroad layout that is able to detect and report when the bridge reaches this position. This indicator is used to trigger stopping of the bridge at the destination position.
- Create an on-off switch "Track 1". This on-off switch is used to trigger turntable movement and to act as a memory in order to stop the bridge at the correct position.
- Assign the move of the turntable bridge (in any direction) as operations to the on-off switch "Track 1".
- Assign the on-off switch "Track 1" as turntable operation to the turntable according to the image below.



Stop:
Turn <:
Turn >:
Move <:
Move >:
Step <:
Step >:
Go 1 <: Track 1
Go 1 >: Track 1

Diagram 199: Assigning operations of a on-off switch to a turntable

- Create a flagman “Track 1” and assign the indicator “Track 1” as trigger. This flagman is used to stop the turntable when the bridge reaches track “1”.
- Assign the state “on” of the on-off switch “Track 1” as condition to the flagman. In this way it is ensured that the bridge is stopped at track “1” only if it should do so.
- Assign the command to stop the turntable as operation to the flagman.
- Via the operations of the flagman the on-off switch “Track 1” should be turned off again in order to return to the initial state.

How it works:

If the turntable is instructed to go to track 1, then the on-off switch “Track 1” is turned on. This on-off switch starts the bridge to move. When the physical bridge of the turntable reaches track 1, the feedback indicator “Track 1” is turned on. This again triggers the flagman, that is turned on, because the on-off switch “Track 1” acting as a memory is still turned on. The flagman then stops the turntable.

Notes:

This is a very rough explanation of the setup. Detailed instructions would be out of proportions of this manual. This example should give you a first idea how the mechanism works in principle.

The key is the usage of the on-off switch as a memory. It is turned on at the beginning of the move to the destination track and it ensures that the turntable is correctly stopped at the right track.

Normally you will create two on-off switches for each track, one for each direction.

A problem might arise due to the fact that in many cases stopping of the bridge must be triggered just before the bridge reaches the destination track rather than just after arrival in order to stop the movement in time.

Transfer tables are setup for indexing accordingly.

Setup of a generic turntable for indexing is done in the same way. The only further measure to be taken is setup of additional operations for the commands normally supported by analog turntables as displayed in Diagram 198.

17.7 Segment Turntables

In **TrainController™** **Gold** turntables can be set up as segment turntables. In this case the first and the last of those tracks are specified, which cannot be reached by the bridge.

A segment turntable always turns with the house to all active tracks. Forward and reverse tracks, as well as turning to a certain track with a certain orientation of the bridge are not possible.

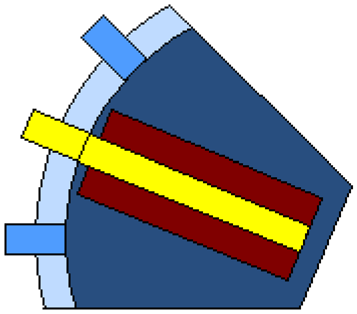


Diagram 200: Small Segment Turntable with three Tracks

Small segment turntables with few sidings and a small angle between the two outermost tracks are displayed as shown in Diagram 200. The house is not displayed in this variant for reasons of space.

If the segment turntable has many tracks or if the angle between the two outermost tracks is large, then the turntable is displayed as shown below:

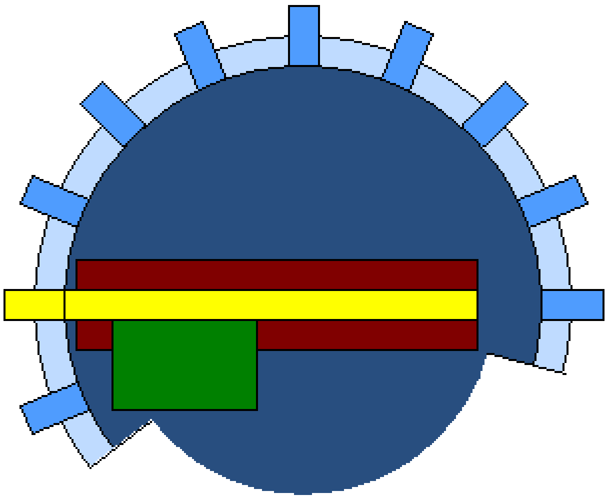


Diagram 201: Large Segment Turntable

As you can see in both diagrams, the bridge of a segment turntable can only be accessed from one side. **TrainController**™ **Gold** automatically takes this into account in the calculation and control of paths across the bridge.

The arrangement of a segment turntable and in particular the synchronization with a switchboard symbols works, apart from identifying the two tracks, which mark the non-covered area of the turntable, in the same way as for other turntables.

18 Special Applications

18.1 Mixing manual and automatic Operation

X

TrainController™ will not supersede you – the human operator. The software can make large scale railroad operations manageable by one person, matching operations found on the largest club layouts. In many cases, several trains will run automatically under control of the computer while certain other trains remain under manual control of the human operator.

Very often certain parts of the layout are controlled fully automatically by the computer (e.g. hidden yards) while other parts of the layout remain under complete control of the human operator (e.g. fiddle yards). In this section, it is outlined how trains can be passed from manual to automatic control or vice versa.

A typical example is displayed in the block diagram below:

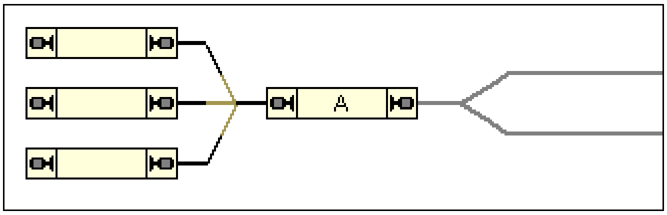


Diagram 202: Mixing manual and automatic operation

On the left side of the layout, a hidden yard is located. This hidden yard is operated fully automatically by the computer. On the right side of the layout, a small yard is located that is operated manually.

The left part – the automatic part of the layout – is equipped with indicators in each siding. A block diagram with blocks and routes and additional schedules have been created to control entry and exit of trains into and out of the left part of the layout automatically.

The right part – the manual part of the layout – is not included into the main block diagram. The track layout is indicated in Diagram 202 with gray lines.

Passing trains from manual to automatic control

The key is the block marked with an “A”. It marks the interface between the manual and the automatic part of the layout. If trains leaving the manual part of the layout will be passed to automatic control without further interaction a *train identification device* is needed here (see section 5.5, “Train Tracking”). Such a device is able to detect which train is about to enter the automatic part of the layout. If block “A” is associated with the train identification device according to Diagram 170, then **TrainController™** will perform the assignment of each detected train to block “A” automatically.

Additionally, you can assign a *schedule* of the Dispatcher as *operation* to a *flagman indicator*, which is triggered, when block “A” is reserved. If this is done, then the manually operated train passing the train identification device is not only detected and assigned to block “A”, but a schedule of the Dispatcher is also started, that runs the train automatically to a free block in the hidden yard.

In this way the train is passed from manual to automatic control without further interaction.

In many cases, the manual part of the layout is not even known to **TrainController™**. Indeed it is not necessary to include the parts of the layout, that are not operated by the computer, in the block diagram. Only the automatic part of the layout including all engines and trains, that are to be operated by the computer, must be known to the *Visual Dispatcher*. Control of each engine can be assigned to the digital system (see section 3.7, “Passing control between Computer and Digital System”). When an engine passes block “A” on its way from the manual to the automatic part of the layout and a schedule is started with this engine at block “A”, then the software will gain control of the engine automatically. When the schedule is finished, control is given back to the digital system and the engine can then be controlled manually.

Passing trains from automatic to manual control

With the features outlined above the automatic pass of engines from manual to automatic control is supported.

There is a special option for the opposite direction as well. This option is called **Release Last Block** and should be set as a property of all automatic schedules ending in block “A”. Normally – if this option is not set – each engine finishing a schedule in block “A” will keep this block reserved permanently - even after the engine has been taken over by manual control. As long as this block remains reserved no other engine will be able to perform another schedule ending in this block. To prevent you from being forced to release such blocks manually set this option for all automatic schedules ending in blocks where engines are passed to manual control. If this option is set for a schedule then the destination block is automatically released when the schedule is finished.

Passing control of trains without a train identification system

It is also possible to pass trains from manual to automatic control without the use of a train identification system. This is done by means of *train tracking*. In this case the manually operated part of the layout must be equipped with track sensors and this part of the layout must be included into the block diagram, too. An example how this is done is outlined on page 240.

18.2 Operating Several Digital Systems Simultaneously

X

With **TrainController™** it is possible to operate several digital systems in parallel. This is for example useful, if

- Your favorite digital system does not support monitoring of track sensors and feedback events.
- All digital addresses provided by your digital systems are already in use and you need more capacity to operate additional items.
- Your digital system is too slow for efficient monitoring of track sensors – especially in the case of larger model railroad layouts.
- You want to use separate digital systems for engine and accessory operation.

TrainController™ supports the simultaneous operation of up to 12 digital systems. During operation it does not matter, to which system particular items are connected. **TrainController™** handles all connected digital systems like one large system. All features can be used without any conditions as if only one large system were connected. It is for example not important if the turnouts contained in a certain *route* are connected to the same digital system or to different systems.
Only when the digital address of an engine, turnout, track sensor, etc. is specified, then you have to take care, that the correct digital system is selected (see Diagram 78).

18.3 Operation of Modular Layouts

X

Modular layouts can be controlled with all **TrainController™** editions.

If the composition of the modules frequently changes, however, and the track diagram of the complete layout therefore consistently changes, too, then this is very well supported by **TrainController™ Gold**. For this purpose a separate switchboard track diagram is created for each particular module and the corresponding block diagram is automatically calculated by the software. These modules are linked to each other with the so called connector symbols available in **TrainController™ Gold**. If the composition of modules is changed, it is sufficient to rename the connector symbols accordingly. **TrainController™ Gold** then calculates the new resulting map of the whole module system instantly and automatically. Tracking of trains on the computer screen is then immediately possible without further manipulations.

If trains are to be driven by the computer, then spontaneous runs (see section 5.10, “Spontaneous Runs”) are well suited on such layouts. Unlike schedules, spontaneous runs do not depend on predefined itineraries, which may change, if the composition of modules changes. The use of schedules would require additional adjustments after each change of the module composition, which are not necessary for spontaneous runs. The manual operation of routes, block exit locks, limiting of blocks to specific trains or to a specific direction of travel provide a variety of possibilities for spontaneous runs to influence the automatic operation or to intervene.

18.4 Running Conventional Engines without Decoder

Stationary Block Decoders

TrainController™ provides the possibility to control conventional engines, i.e. locomotives without an own engine decoder. This is done with *stationary block decoders*, i.e. decoders or computer controlled throttles, which are mounted at fixed positions on your model railroad rather than in each locomotive.

This feature is useful,

- if you have a large collection of locomotives and not all are digitally upgraded.
- if you have a conventional - i.e. non-digital - operated model railroad and want to control it with your computer without installing an engine decoder in each locomotive first.
- if the models of your engines are very small and the decoders do not fit into the engines (e.g. when you run Maerklin Mini Club).

In all **TrainController™** provides three methods of operating your trains, which are explained below:

- Operating trains with individual engine decoders (“Computer Command Control”).
- Operating trains with stationary block decoders with static assignment to track sections (“Computer Section Control”).
- Operating trains with stationary block decoders with dynamic assignment to track sections (“Computer Cab Control”).

Additionally it is possible to use all these methods simultaneously, i.e. it is possible, to run conventional engines and digital engines at the same time.

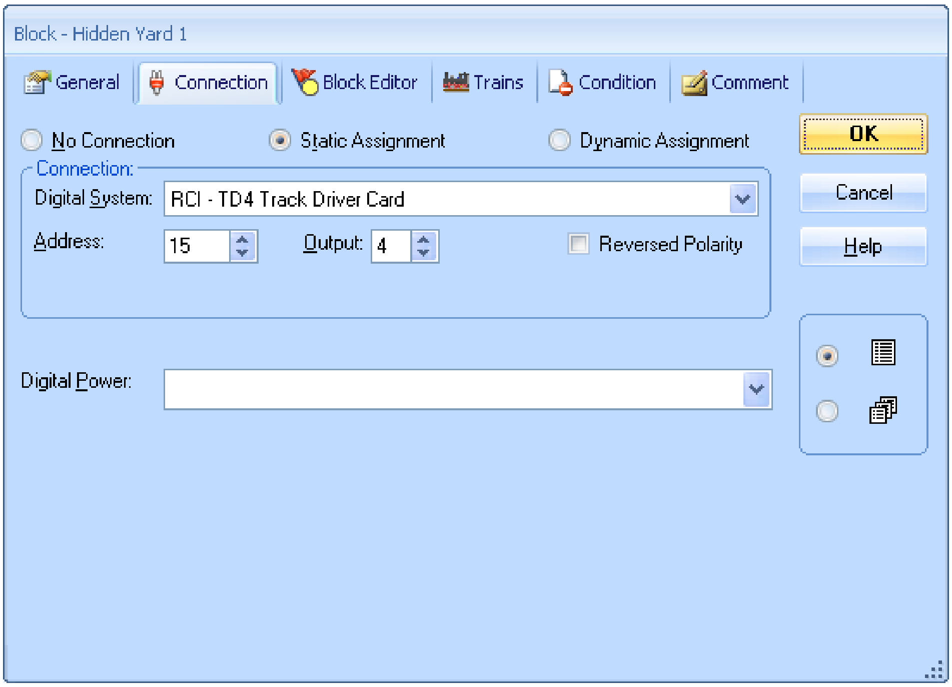
Computer Command Control

This is the method supported by most of the digital systems of today. This is also the only method supported in the first versions of **TrainController™** In this case each engine is equipped with an individual engine decoder and can be operated directly by sending speed or function commands to the decoder. More details are explained in the documentation of the digital system.

Computer Section Control

This method is also called “*Computer Block Control*” or “*One throttle per section*”. In **TrainController™** this kind of operation is based on the *blocks* of the *Dispatcher*. Unlike *Computer Command Control* it is possible to operate conventional locomotives with this method.

In this case all *blocks*, in which conventional locomotives should be able to run, must be electrically insulated from each other. Additionally each block is electrically connected to a specific decoder, which is mounted at a fixed position of your model railroad. The power in each block is controlled by the associated decoder. This results in a static assignment between each block and a stationary block decoder. To assign a block to its own stationary block decoder you have to assign a digital address to each block - namely the address of the connected stationary block decoder. Whenever a block is reserved for an engine or train, then all consecutive engine commands are sent to the stationary block decoder, which is connected to the block, instead to the engine itself. Since several blocks can be reserved for an engine or train, **TrainController™** sends engine commands to all affected blocks.


Block - Hidden Yard 1

No Connection [✓ Static Assignment] Dynamic Assignment

Connection:

Digital System: RCI - TD4 Track Driver Card

Address: 15 Output: 4 Reversed Polarity

Digital Power:

Diagram 203: Computer Section Control - Specifying the Digital Address of a Block

Computer Cab Control

This method is also called "*Progressive Cab Control*". In **TrainController**™ this kind of operation is based on the *blocks* of the *Dispatcher*. Unlike *Computer Command Control* it is possible to operate conventional locomotives with this method. This method supports also the possibility to run digital and conventional engines on the same track.

Unlike *Computer Block Control* there is no permanent electrical connection between blocks and *stationary block decoders*. For this reason the number of stationary block decoders may be lower than the number of affected blocks.

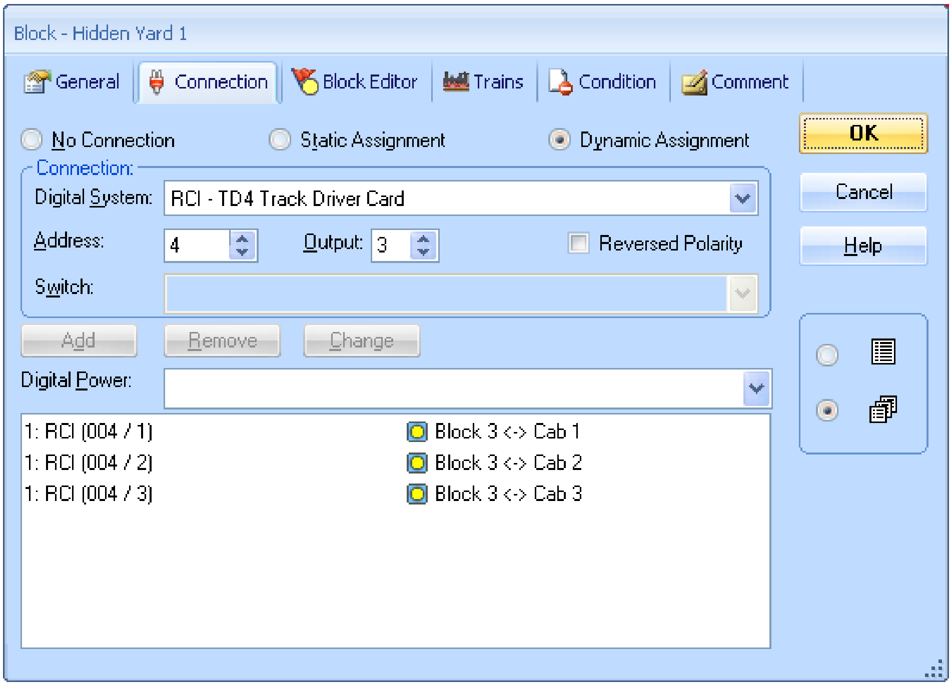
All blocks in which conventional locomotives should be able to run, must be electrically insulated from each other. The electrical connection between blocks and decoders is established when required. This results in a dynamic assignment between each block and one of several stationary block decoders, which are mounted at fixed positions of your model railroad. The power in each block is controlled by a dynamically assigned decoder.

In order to arrange Computer Cab Control for a specific block, you have to specify a list of digital addresses - namely the digital addresses of the stationary block decoders, from which one should be dynamically selected. But there is one more thing to do: when a stationary block decoder is selected for a specific block, then the power generated by this decoder must be routed to the block. In order to establish the electrical connection you have to specify an *on-off switch* (see section 2.5, “Signals and Accessories”) for each stationary block decoder, which will be used to turn on or off the connection between the block and the decoder. In most cases a sequence of switching operations (e.g. a sequence of several relays) must be operated to establish the connection between a block and a stationary block decoder. In this case make use of the possibility, to assign a set of *operations* (see section 14.4, “Operations”) to an on-off switch.

Whenever a block is reserved for an engine or train, then the Dispatcher searches an appropriate stationary block decoder. If a decoder was found, then the on-off switch, which is associated with the connection between the block and the decoder, is automatically turned on. When the block is released, this on-off switch is automatically turned off again.

If you have arranged your blocks correctly, then you do not have to take care of the dynamic assignment of decoders to blocks and the routing of the electrical power from the decoders to the affected blocks. This is done automatically by the Dispatcher.

Of course it is possible to arrange your blocks in a way, that one decoder can control several blocks simultaneously, which are reserved for the same train.


Block - Hidden Yard 1

No Connection [ ] Static Assignment [ ] Dynamic Assignment [✓]

Connection:

Digital System: RCI - TD4 Track Driver Card

Address: 4 Output: 3 Reversed Polarity [ ]

Switch:

Digital Power:

1: RCI (004 / 1)	● Block 3 <-> Cab 1
1: RCI (004 / 2)	● Block 3 <-> Cab 2
1: RCI (004 / 3)	● Block 3 <-> Cab 3

Diagram 204: Arranging a Block for Computer Cab Control

Adjusting the Polarity of each Block

!

In order to direct each train to the correct direction of travel and in order to avoid shortcuts **TrainController**™ applies a *logical polarity* attribute to each block. For each block **TrainController**™ assumes that the following is true:

If a train is located in a block heading to the right/bottom and if the train is directed to move forward then the train moves to the right/bottom.

Unlike computer command control, where this condition is usually true, if the decoder is installed properly, this is not always true when stationary block decoders are used. The direction in which the train moves depends on the wiring of each block. In order to let each block meet the above rule without rewiring of your layout **TrainController**™ provides an option to adjust the *logical polarity* of each block in the software (see Diagram 203 and Diagram 204).

It is very easy to adjust the polarity of each block in **TrainController™**. Perform the following steps:

- Put a train on the track inside of the block.
- Make sure that the train is heading to the right or bottom, respectively.
- Assign the train to the block in the *Visual Dispatcher*.
- Make sure, that the train image in the block symbol of the *Visual Dispatcher* is also heading to the right or bottom, respectively.
- Select the train in the *Train Window*.
- Drag the speed slider in the *Train Window* to the right.
- If the actual train on the layout is now moving to the right or bottom, respectively, then the polarity of the block is correctly adjusted. Otherwise open the properties of the block and change the polarity of the block by toggling the **Reverse Polarity** option.

Look at the following example:

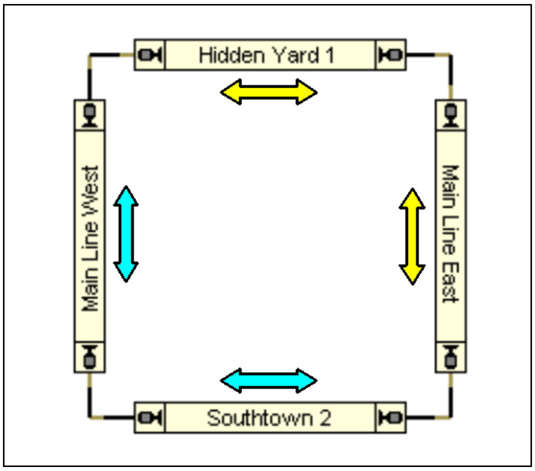


Diagram 205: Block Diagram of a Circular Layout

It can be assumed that the physical wiring of the layout displayed above is done in a way that the track power will not change its polarity when a train cycles around the loop. In other words: the <u>physical</u> polarity of all blocks in the above diagram can be assumed to be identical.

The situation in **TrainController™** is different. **TrainController™** does not want to rely on the fact, that the layout has been wired in a certain way. Additionally, the structure of many layouts is much more complicated. It can contain reversing loops or several levels, it can be based on a modular structure, etc.

For this reason **TrainController™** uses the *logical polarity scheme* described above. If the layout displayed above is wired in a way that the track power will not change its polarity when a train cycles around the loop, then the train will pass “Hidden Yard 1” and “Southtown” to different logical directions (left or right), even though the physical polarity of the track power remains unchanged. A train that passes “Hidden Yard” to the right at positive track polarity will pass “Southtown 2” to the left at the same track polarity. As a consequence “Hidden Yard 1” and “Southtown 2” have different *logical polarity* from the point of view of the software. The differences with regard to the *logical polarity* of the particular blocks are marked with a yellow or blue arrow in the diagram displayed above.

Running conventional and digital Engines on the same Track

This is supported with an additional option. Each block, on which conventional engines as well as digital engines will be able to run, must be arranged for dynamic decoder assignment (*Computer Cab Control,* unless the RCI system is used - see below). Additionally it is possible, to assign one extra *on-off switch* to each affected block (see Diagram 204). This additional on-off switch is used to turn on and off the “digital power” for this block. Whenever the block is reserved for a conventional engine, then the block is automatically connected to an appropriate stationary block decoder as outlined in the section before. When the block is reserved for a digital engine, then the extra on-off switch is used to turn on the “digital power” for this block.

In this way it is even possible to run conventional and digital engines in different blocks of the same track at the same time.

The Track Driver Cards of the RCI system provide a built-in feature to route DCC power directly to the output points. This feature is used, when a block is statically assigned to a stationary block decoder on a Track Driver Card (*Computer Section Control*). Whenever a block is reserved for an engine with an own DCC decoder, then the DCC mode is automatically turned on for the Track Driver point connected to this block. When the block is released, then the DCC mode is turned off.

Notes

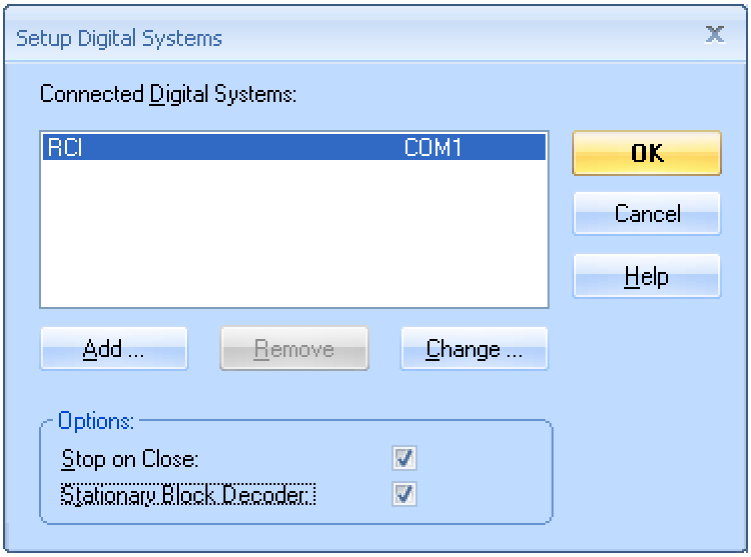
You can use regular engine decoders of any digital system as stationary block decoders. To use an engine decoder as stationary decoder, mount it at a fixed position of your model railroad and connect the wires, which are normally connected to the motor, to the track instead. To be on the safe side you should ask the dealer or manufacturer of the engine decoder, if it can be used as stationary block decoder without the risk to be damaged. The supplier of the program will not be liable to you for any damages.

TrainController™ supports also digital systems, which provide computer controlled throttles dedicated to be used as stationary block decoders (e.g. the systems RCI or CTI).

The operation of conventional locomotives with stationary block decoders is based on the blocks of the *Dispatcher* (see chapter 5, "The Visual Dispatcher"). As a consequence engines or trains can only be operated with stationary block decoders, if they are running under control of the *Dispatcher*. In return the *Dispatcher* guarantees, that traveling engines and trains are operated by the appropriate stationary block decoders. Because the *Dispatcher* is able to reserve blocks automatically according to the progress of traveling engines and trains under its control, it can also assign the appropriate stationary block decoders automatically to the engines.

Additional Options

In order to operate *stationary block decoders*, select the Option **Stationary Block Decoder** in the **Setup Digital Systems dialog box** (see Diagram 206).



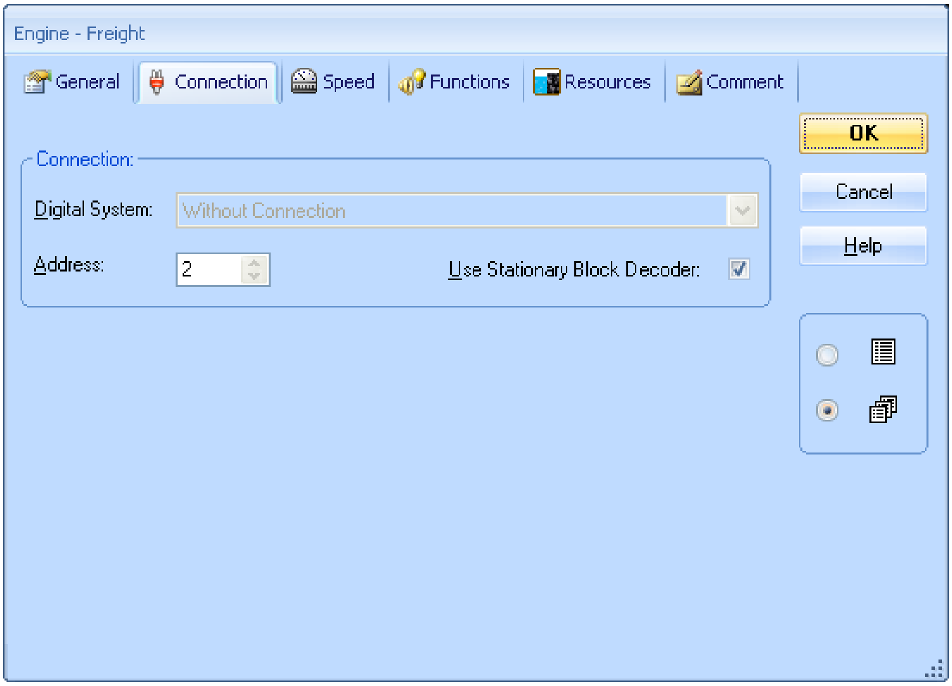
RCI COM1

Stop on Close: ✓
Stationary Block Decoder: ✓

Diagram 206: Arranging the Digital Systems to use Stationary Block Decoders

When stationary block decoders are used, then in the **Block Dialog Box** an additional tab labeled **Connection** appears (see and). Here the digital addresses of the stationary decoders, which are associated with this block, are to be specified.

For each conventional engine select the Option **No Connection** in the tab labeled **Connection** of the **Engine Dialog Box.**



Connection:

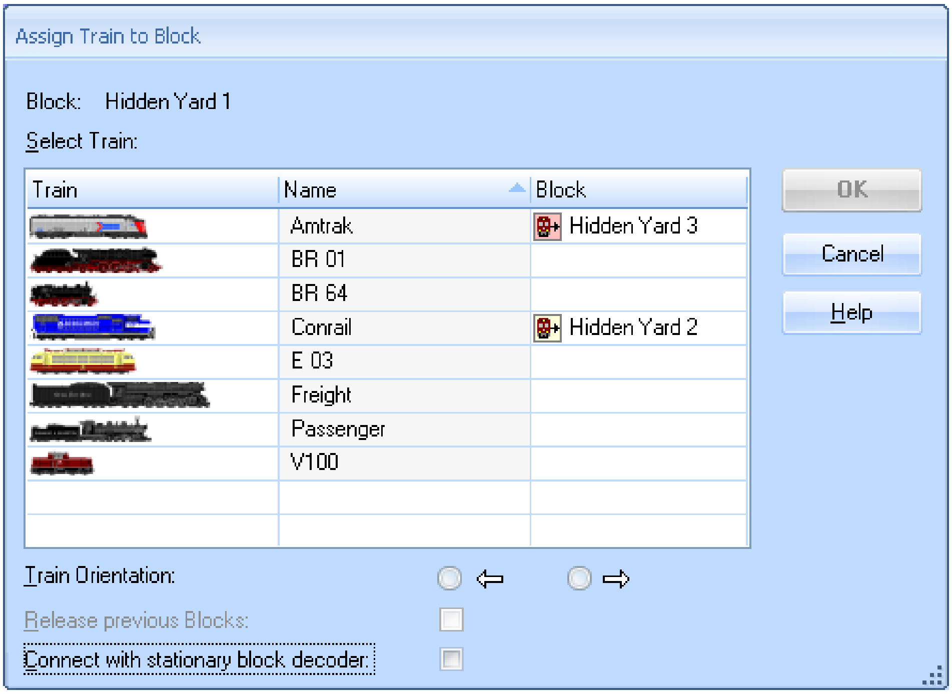
Digital System: Without Connection

Address: 2

Use Stationary Block Decoder: ✓

Diagram 207: Entering a Conventional Engine

When a train is assigned to a block which is setup for *Computer Cab Control* through the **Assign Train to Block dialog box**, then the additional option **Connect with stationary block decoder** is provided. Select this option, if the block should be connected to an available stationary block decoder during this assignment. In this case the stationary block decoder is reserved for this train. Until the block is released this decoder cannot be used by other trains. If this option is not selected then **TrainController**™ tries to reserve an appropriate stationary decoder, when the train starts running on a *schedule* or when additional blocks are reserved for this train.


Assign Train to Block

Block: Hidden Yard 1

Select Train:

Train	Name	Block
	Amtrak	Hidden Yard 3
	BR 01	
	BR 64	
	Conrail	Hidden Yard 2
	E 03	
	Freight	
	Passenger	
	V100	

Train Orientation: ○ ⇦ ○ ⇨

Release previous Blocks: ☐

Connect with stationary block decoder: ☐

Diagram 208: Reserving a Block for a Conventional Engine

List of Examples

Index