

RAILROAD & Co.[®]

TrainProgrammer[™]



Version 8

Users Guide

November 2012

RAILROAD & Co.[®]

TrainProgrammer[™]

Version 8

Users Guide

November 2012

Copyright© Freiwald Software 1995 - 2012

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

All rights reserved.

The content of this manual is furnished for informational use only, it is subject to change without notice. The author assumes no responsibility or liability for any errors or inaccuracies that may appear in this book.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of the author.

Table of Contents

About this Document	5
RAILROAD & CO. TrainProgrammer™ Users Guide.....	5
Help Menu	6
Quick Start - Step 1: Installation and Program Start	8
Installation.....	8
Program Start	9
Quick Start - Step 2: Reading the digital Address of a Locomotive.....	13
Quick Start - Step 3: Changing the digital Address of a Locomotive	18
Quick Start - Step 4: Changing a short Address to a long Address	20
1 Introduction	24
1.1 Overview.....	24
Supported digital Systems.....	24
Use	25
1.2 Fundamentals of Use	26
The Overall Principle.....	26
Decoders and Devices.....	26
Configuration Options and Decoder Configurations	26
Decoder Database	27
File Handling	27
User Interface Design	27
Window Handling.....	28
Customization of Menus, Tool Bars and Keyboard Accelerators.....	28
Printing.....	29
2 Working with the Decoder Window	31
2.1 General.....	31
2.2 Selecting the right Decoder Configuration	32
Freeware Configuration	32
Standard Configuration	33
Specialized Configurations	34
2.3 Programming the Decoder	34
Editing Decoder Values	36
Reading Values from the Decoder	36
Writing Values to the Decoder.....	36
Profile Mode	36
Automatic Activation of the Programming Track.....	37
Programming on the Main Track	38
Meaning of the Status Symbols.....	38
2.4 Editing Speed Tables	39

2.5	Mapping Function Keys to Output Locations	40
2.6	Custom Editors	41
3	The Test Panel.....	42
4	The Direct Programmer	43
5	Run-In of Locomotives	44
6	Use of a Roller Test Bench	46
7	The Decoder Database	48
7.1	Editing of Decoder Configurations	49
	Configuration Properties.....	51
7.2	Configuration Options	51
	Folder.....	51
	Number	52
	Check Box	52
	Structure	53
	Formula.....	53
	Overview of Option Types	54
	Properties of Configuration Options.....	55
	Reference	55
	Indexed Options.....	56
7.3	Multilingualism, Derivation, Templates	57
	Multilingual Decoder Configurations.....	57
	Derivation	57
	Templates.....	58
Appendix: Troubleshooting		59
	Problems during Reading and Writing of Decoder Values	59
	Compatibility problems	59
Index		60

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.
- **+Net™** is a module, that allows you to control your layout with a network of several computers running **TrainController™**.
- **+4DSound™** is a module, that recreates realistic spatial sound effects for each model railroad layout controlled by **TrainController™** without the need to install on-board sound into each decoder.
- **+SmartHand™** is the world's premium handheld railroad control system designed for computer controlled model railroads.

RAILROAD & CO. TrainProgrammer™ Users Guide

An overview of the basic concepts of **TrainProgrammer™** is provided in this Users Guide. By reading this document one can obtain information about the many features of the product. Additionally you are provided with background information necessary for digital decoder programming with **TrainProgrammer™**.

The document is divided into two parts. Part I provides a quick start tutorial for users, who are in a hurry and are in a fever to start quickly. Part II explains the fundamentals of use.

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, refer to the **Help** menu of **TrainProgrammer™**, please.

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



Basic content. Novice readers should focus on these parts.



Extended content for advanced users. Novice readers should ignore these parts in the beginning.



Important note.

Help Menu

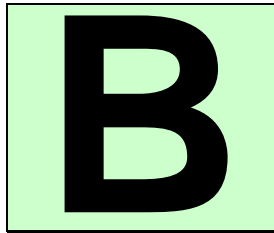
The help menu installed with **TrainProgrammer™** contains detailed reference information necessary for using the program. All menus, dialogs and options are completely described and can be referred to in case of questions or problems.



Please note: no document is complete without the other. If you want to know, what a certain term means or what a certain function does, refer to this Users Guide, please. If you want to know, how a certain object is to be edited or how a specific function is to be executed, call the help menu.

Part I

Quick Start



Quick Start - Step 1: Installation and Program Start

You have obtained **TrainProgrammer™** to conveniently program the decoders and other devices of your digitally controlled model railroad with your computer. It is easily understood, if you are eager to use the program as soon as possible. If you are in a hurry about starting without reading the complete Users Guide first, you can also run through the following quick start tutorial about **TrainProgrammer™**.

Detailed explanations about the concepts, that are the fundamentals of the following, can be found in Part II of this document. It is strongly recommended to study the contents of this part prior to working seriously with **TrainProgrammer™**.

Now let us start:

Installation

The installation file of **TrainProgrammer™**, its name is TPSETUP.EXE, can be downloaded from the download area of the Internet home page of the software (www.freiwald.com) or started from a CD ROM.

After starting TPSETUP.EXE a self-explaining window is displayed, that guides you through the steps, that are necessary to install **TrainProgrammer™** on your computer.

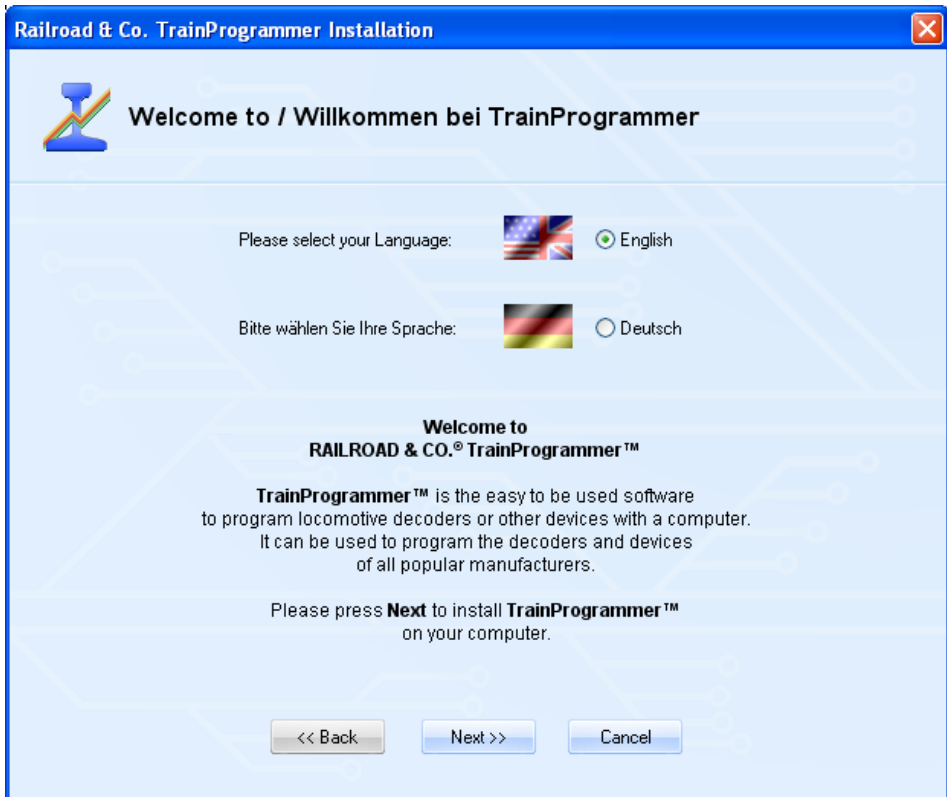


Diagram 1: TrainProgrammer™ Setup Screen

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

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

Program Start

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

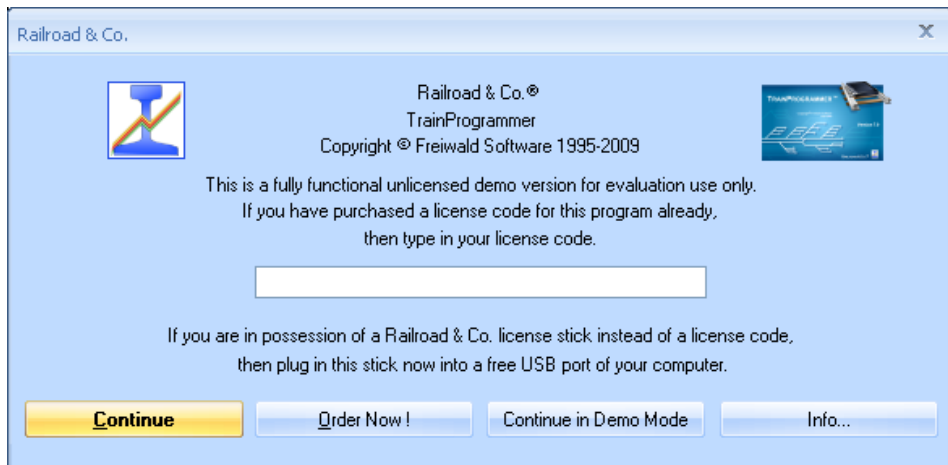


Diagram 2: License Inquiry

After start of the program the software first inquires your license key. Do not be concerned, if you are not yet in possession of such key. Press **Continue unlicensed**, if you want to try the software before buying it.

In the next step the connected digital system is to be configured. Call the **Setup Digital Systems** command of the **Railroad** menu:

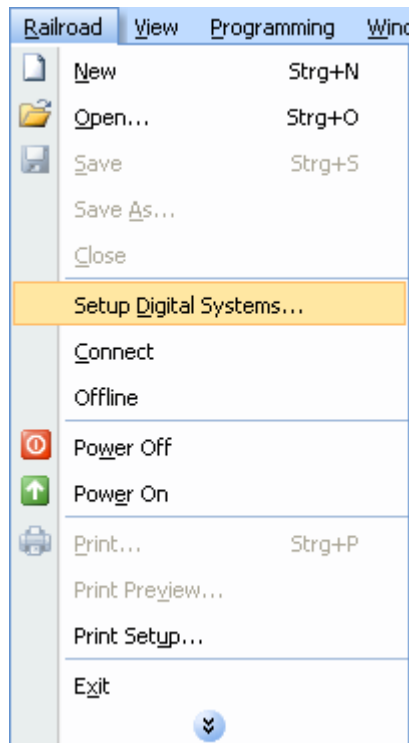


Diagram 3: Railroad menu

The dialog box displayed below will appear now:

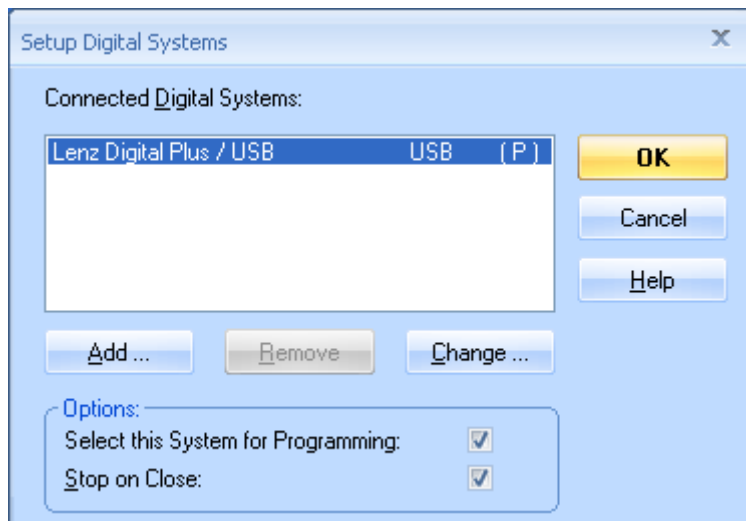


Diagram 4: Setup Digital Systems dialog

If your digital system and/or the USB or serial port of your computer, to which your digital system is connected, is not displayed accordingly, 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** menu. 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 even some error messages, 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 instructions of the manufacturer.

Quick Start - Step 2: Reading the digital Address of a Locomotive

The following steps of the tutorial require a locomotive with a NMRA DCC compatible locomotive decoder.

First put such locomotive onto the programming track and try to read the address programmed into the decoder with your digital system. It is certainly not necessary to do this manual read always when you work with **TrainProgrammer™**, but for this tutorial this step is recommended to verify, that the digital system, the locomotive and the decoder are correctly working. Do not proceed until you have been able to successfully read the address from the decoder with your digital system.

After Step 1 the screen display of **TrainProgrammer™** should look as follows:

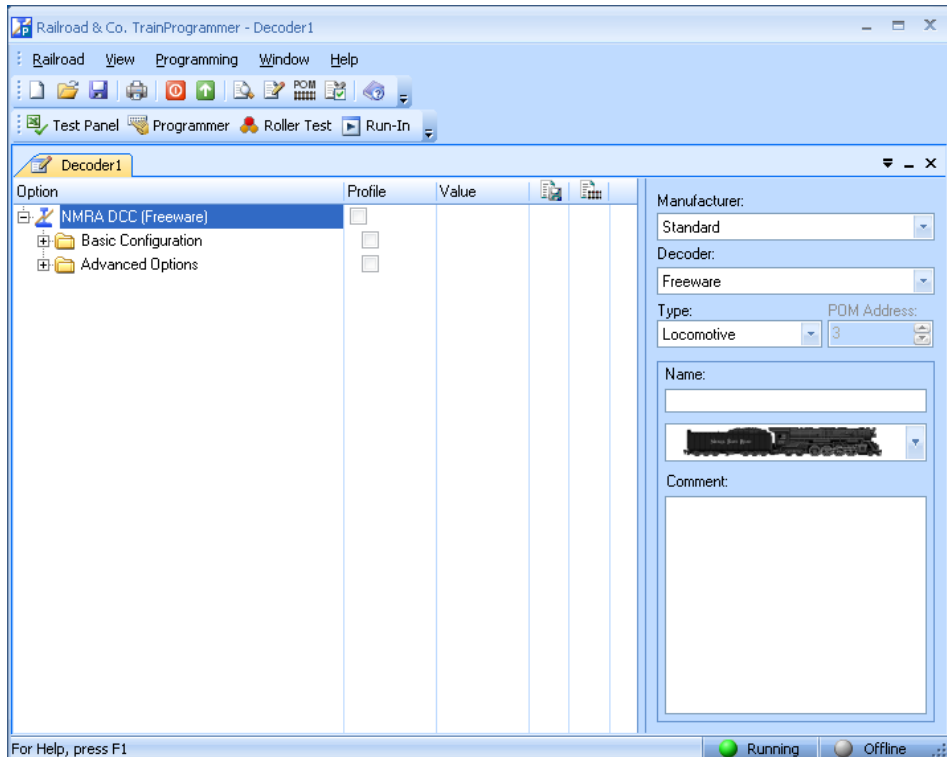


Diagram 5: TrainProgrammer™ initial screen

Press to the '+' marking left of the entry **Basic Configuration** in the **Option** column and then to the same marking left of the entry **Address**.

These options should be expanded now like expanded folder entries in the File Explorer of the Microsoft Windows system.

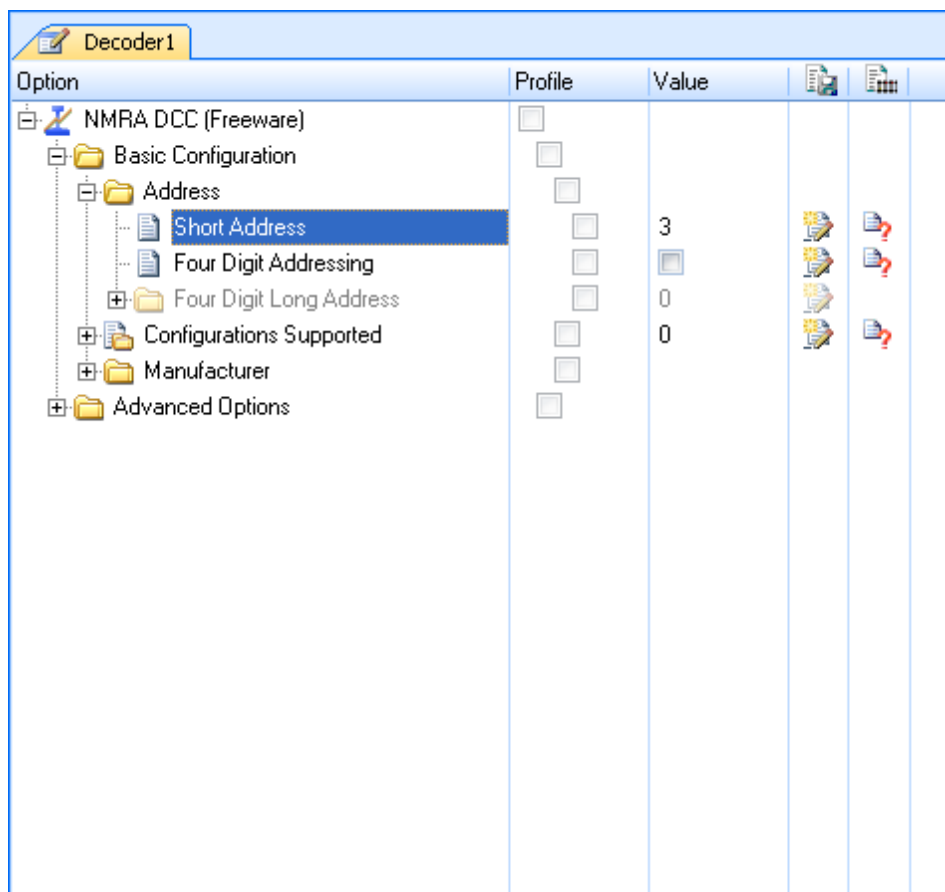


Diagram 6: Selecting a Configuration Option

Now select the entry **Short Address** and call the **Read from Decoder** command of the **Programming** menu.

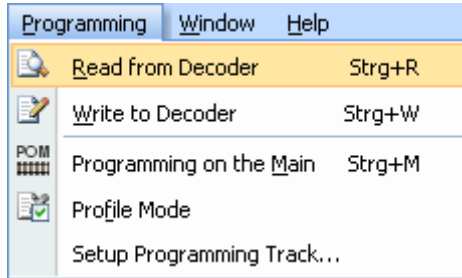


Diagram 7: Reading a Configuration Option

If everything is installed correctly then the address programmed into the decoder will appear in the **Value** column.

Decoder1					
Option	Profile	Value			
[-] NMRA DCC (Freeware)	<input type="checkbox"/>				
[-] Basic Configuration	<input type="checkbox"/>				
[-] Address	<input type="checkbox"/>				
Short Address	<input type="checkbox"/>	10			
Four Digit Addressing	<input type="checkbox"/>				
Four Digit Long Address	<input type="checkbox"/>	0			
Configurations Supported	<input type="checkbox"/>	0			
Manufacturer	<input type="checkbox"/>				
Advanced Options	<input type="checkbox"/>				

Diagram 8: Display of a read Value

In our example the address programmed into the decoder is '10'. In your case another address will probably appear. It should be the same value, however, as previously read with the digital system.

Quick Start - Step 3: Changing the digital Address of a Locomotive

Now we want to program another address into the decoder. Instead of '10' (or the currently displayed value) we want to set the address of the locomotive to '20'.

Click once with the left mouse button to the value of the short address in the **Value** column. The value becomes editable and can be changed to '20':

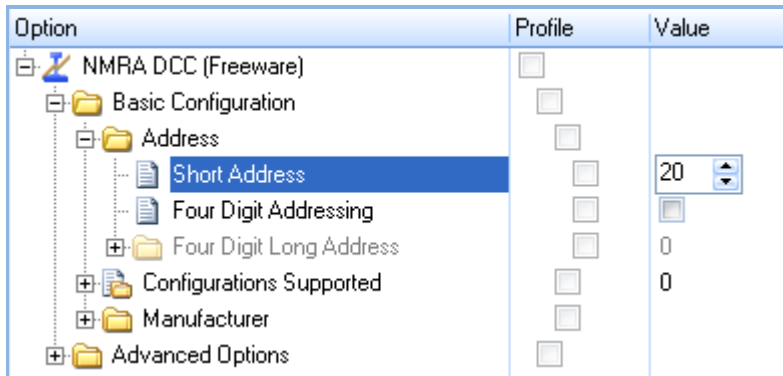


Diagram 9: Changing the value of a Configuration Option

Now commit the changed value by clicking with the left mouse button somewhere on the computer screen.

Ensure, that the entry **Short Address** is still selected and call the **Write to Decoder** command of the **Programming** menu.

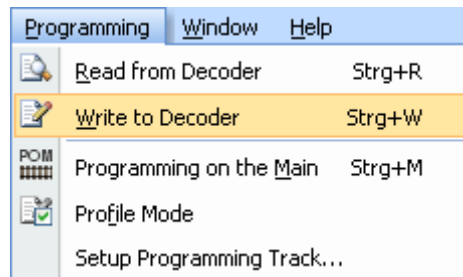


Diagram 10: Writing a Configuration Option

If everything is installed correctly then the address '20' will now be written into the decoder. You can verify this by repeating the steps of Step 2 of this tutorial or by reading the address with your digital system.

Quick Start - Step 4: Changing a short Address to a long Address

Now we want to program a long address, e.g. '2365' into the decoder. Experienced users know, that doing this with the digital system requires complex calculation of two decoder values (CV 17 and CV 18) as well as a lot of keystrokes for writing of a least 3 decoder values (CV17, CV18, CV29).

With **TrainProgrammer™** such procedures are much more convenient and effective and require just a few mouse clicks.

At first set a tick mark in the **Value** column right of the entry **Four Digit Addressing**. Then expand the option **Four Digit Long Address**. Change the current value '0' right of **Four Digit Long Address** to '2365'. The screen display should look as follows now:

Option	Profile	Value
NMRA DCC (Freeware)	<input type="checkbox"/>	
Basic Configuration	<input type="checkbox"/>	
Address	<input type="checkbox"/>	
Short Address	<input type="checkbox"/>	20
Four Digit Addressing	<input checked="" type="checkbox"/>	
Four Digit Long Address	<input type="checkbox"/>	2365
High Byte	<input type="checkbox"/>	201
Low Byte	<input type="checkbox"/>	61
Configurations Supported	<input type="checkbox"/>	32
Manufacturer	<input type="checkbox"/>	
Advanced Options	<input type="checkbox"/>	

Diagram 11: Changing the long Address of a Decoder

Now select the entry **Address** (important!) and call the **Write to Decoder** command of the **Programming** menu.

The programming procedure is now executed and after a while all required values for the long address are written into your decoder. That's all !

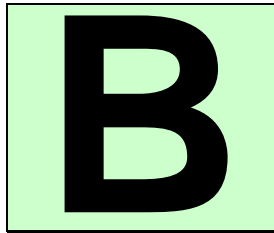
Some Background Information:

The options of a decoder are arranged and viewed in **TrainProgrammer™** like the files and folders in the File Explorer of Microsoft Windows. Some options can ‘contain’ other options (like folders containing files or other folders). The **Address** option, for example, contains all options and CVs, that are relevant for the address of a decoder, among others CV1, CV17, CV18 and that bit of CV29, that controls, whether the long or short address is valid. By selecting an option, that ‘contains’ other options it is possible to read or write all contained options in one step. By selecting the **Address** option we are able to write all options, that specify the long address, to the decoder in one step. By selecting the topmost option in the **Option** column we would be even able to read or write all options of the decoder in one single step.

This feature is one of the keys to make programming of decoders with **TrainProgrammer™** so convenient and effective.

Part II

Fundamentals



1 Introduction

1.1 Overview

B **TrainProgrammer™** is a system for convenient and effective programming of decoders or devices for digital control of model railroads with a Personal Computer running Microsoft Windows 7, XP or Windows Vista.

TrainProgrammer™ can deal with decoders or devices, that support at least one of the following programming methods:

- NMRA DCC direct mode
- NMRA DCC paged mode
- NMRA DCC register mode
- Loconet Option Switch (OPSW) programming
- programming of Selectrix compatible locomotive decoders

TrainProgrammer™ is easy to use. It provides an intuitive and quickly to learn graphical user interface and the ease of point & click to read and write the settings of your decoders. Many decoder information like speed tables or function setups are displayed graphically and can be read, drawn and written with a few mouse clicks. The decoder information can be printed or saved on your computer. In this way the same speed table, for example, can be loaded into several decoders.

If supported by the connected digital system it is also possible to program decoders on the main track. This means that you can program decoders without having to put the locomotive on a separate programming track. This method is also known as “Operations Mode Programming” or “Programming on Main Track”.

TrainProgrammer™ is being delivered with a decoder-database, that contains the configurations of frequently used decoder types. It is possible to create new configurations, to change or delete existing configurations or to add configurations, that are created by others or the manufacturer of your decoder, to the database.

Supported digital Systems

TrainProgrammer™ supports among others the following digital systems for decoder programming (list is not exhaustive):

- Lenz Digital Plus
- Digitrax LocoNet
- NCE
- Roco Digital
- Maerklin Central Station 2
- Uhlenbrock Intellibox, Intellibox 2, IB Basic, IB COM
- Fleischmann Twin Center
- Tams
- Trix Selectrix
- Rautenhaus Digital (SLX and RMX)
- Muet Digirail
- D&H / MTTM Future-Central-Control
- Massoth
- Zimo

RAILROAD & CO. supports use of other applications at the same time sharing the same connections. If you have installed **TrainProgrammer™** together with other **RAILROAD & CO.** applications, then the same configuration of digital systems can be used by all **RAILROAD & CO.** applications at the same time without the need to specify the type of systems or the connecting ports again. Programming of decoders is only possible with one of these connected digital systems at a time, though. The configuration of connected digital systems may contain a digital system not listed here, which is used by another **RAILROAD & CO.** application. In this case this particular system cannot be selected for decoder programming.

TrainProgrammer™ also supports an offline mode, which allows trial operation without a connection to a real model railroad.

Use

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

- Use of **TrainProgrammer™** is possible without the need to be a computer expert or programmer.
- Graphical items are provided instead of an abstract command syntax.
- Procedures are effective and fast.

1.2 Fundamentals of Use

The Overall Principle

B The concept of **TrainProgrammer**[™] is intended to support convenient and effective programming of decoder configurations.

The options used for programming are arranged on the computer screen in a hierarchical structure like the folders and files of the Microsoft Windows operating systems. Options, that belong together, are grouped together. All options, that belong together, for example all options, that describe the speed and running characteristics of a decoder, can be read from or written to the decoder in one step.

Decoders and Devices

A *decoder* is a piece of equipment, that is usually used to control the items found on a model railroad such as locomotives, turnouts, signals, feedback sensors etc. A *device* is a piece of equipment, that is used to control decoders or other devices of a digitally controlled model railroad such as central units, handheld controls, control panels, interfaces, boosters, reversing loop controllers, etc. In this document the term decoder is often used as synonym for device, too.

Configuration Options and Decoder Configurations

A *configuration option* describes an aspect of the settings of a decoder or a group of such aspects. A DCC configuration variable, for example, may be associated with one or more configuration options (for example DCC CV1, which may be associated with a configuration option called “Short Address”). A configuration option is distinguished by several properties, among others its name, a short explanation of its meaning, allowed content, etc. Configuration options can be grouped together according to their meaning. In case of a loco decoder, for example, the options, that control the speed of the loco are usually grouped together and separated from the options, that control the auxiliary functions of the decoder.

A *decoder configuration* is a set of all configuration options, that describe a particular type of decoder or device. There are specialised decoder configurations, that describe just one specific type of decoder (e.g. a certain locomotive decoder brand of a certain manufacturer). There are also general decoder configurations, that describe common properties of a plurality of similar decoders (e.g. a configuration, that describes common settings of NMRA DCC compatible locomotive decoders).

Please do not mistake decoder configurations for decoders. A decoder configuration contains information about one or more types of decoders. A decoder is a piece of equipment used to control your model railroad, e.g. the particular decoder, that is installed in a certain steam locomotive.

Decoder Database

A *decoder database* is a collection of one or more decoder configurations. **TrainProgrammer**[™] is delivered with a default decoder database, that contains the configurations of frequently used decoder types. With **TrainProgrammer**[™] it is also possible, to create your own decoder configurations, to customise existing configurations or to delete not needed configurations from the database. It is also possible to add decoder configurations created by others, e.g. the manufacturer of the decoder, to your own database. In the “goody box” of our Internet web site (www.freiwald.com/pages/goody.htm) an additional decoder database is available for download, that contains more than 400 decoder configurations of popular decoders (courtesy of JMRI). Also ask the manufacturer of your decoder for configurations, that are not yet contained in the database!

File Handling

The complete data, that belongs to one decoder or device is stored in one single file on the hard disk on your computer. This file is called *decoder file*. In the most common case, that the decoder belongs to a locomotive, one can also visualise the decoder file as the file, that contains the data of one locomotive. You can create as many decoder files as you like – for example a separate file for each locomotive in your collection.

The decoder file contains all values stored in the physical options (CVs) of your decoder, an optional name and image (e.g. the name and image of the locomotive, where the decoder is installed), an optional commenting text and other information.

Decoder files are created, opened and stored through the **Railroad** menu of the software.

User Interface Design

The user interface of **TrainProgrammer**[™] 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 2007 styles
- Visual Studio 2008 and 2005
- Native XP
- Office 2003
- Classic Office 2000

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

Window Handling

The information, that belongs to a decoder, and that is stored in one file (see above) is displayed in one window, the *decoder window*. In the most common case, that the decoder belongs to a locomotive, one can also visualise the decoder window as the display of all data, that belongs to one locomotive. You can open as many decoder windows as you like at the same time.

Decoder windows and opened decoder files are associated with each other.

Each decoder window can appear in one of the following states:

- Docked to one of the borders of the main window.
- Docked to another decoder 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.

Customization of Menus, Tool Bars and Keyboard Accelerators

It is also possible to customize the content of menus and tool bars and to change keyboard accelerators.

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.

Printing

The information, that is stored in a decoder file, can be printed in a clearly arranged manner. **TrainProgrammer™** will print the general properties associated with a decoder (e.g. name and image) as well as a table of all configuration options. In this way you can create your own collection of printed data sheets for your decoders and devices.

Railroad & Co. TrainProgrammer Decoder Data Sheet



Description		
Type:	NMRA DCC (Freeware)	
Name:		
Comment:		
Basic Configuration		
Address	CV -1	
Short Address	CV 1	3
Four Digit Addressing	CV 29	1
Four Digit Long Address	CV -1	2365
High Byte	CV 17	201
Low Byte	CV 18	61
Configurations Supported	CV 29	32
Inverted Direction	CV 29	0
28 Speed Steps	CV 29	0
Power Source Conversion	CV 29	0
Adv. Acknowledgment	CV 29	0
Enable Speed Table	CV 29	0
Four Digit Addressing	CV 29	1
Manufacturer	CV -1	
Manufacturer ID	CV 8	0
Version Number	CV 7	0
Advanced Options		
User Identification	CV -1	
ID #1	CV 105	0
ID #2	CV 106	0

Diagram 12: Decoder Data Sheet

2 Working with the Decoder Window

2.1 General

B

TrainProgrammer™ displays all information, that belongs to a decoder, in the *decoder window*. If the decoder belongs to a locomotive, one can also visualise the decoder window as the display of all data belonging to one locomotive.

It is possible to open several decoder windows at the same time.

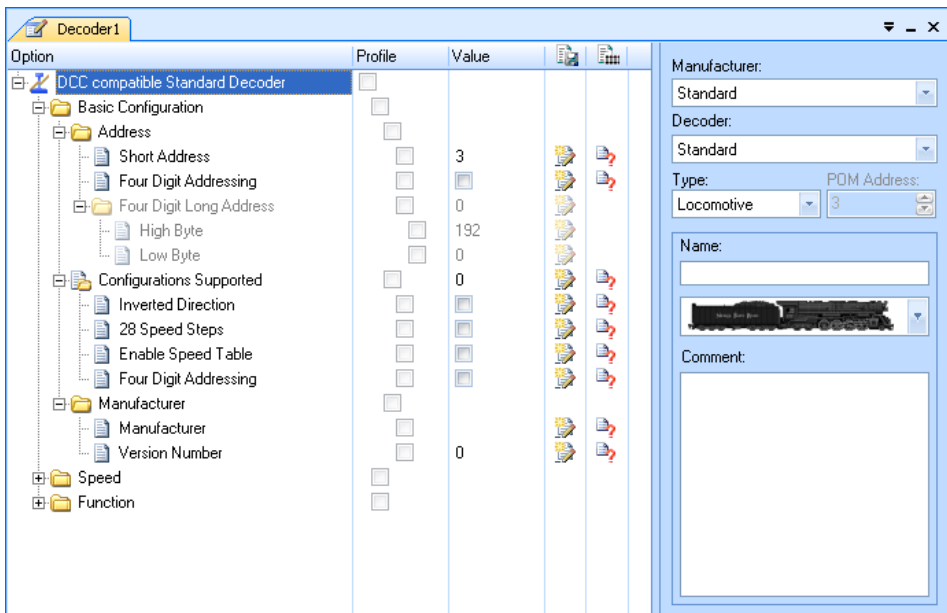


Diagram 13: Decoder Window

The most important part of the decoder window is the *explorer window* in the left part. The explorer window shows all configuration options of the decoder.

The controls in the upper right part of the decoder window are used to select the appropriate decoder configuration.

The lower right part of the decoder window may change according to the currently selected configuration option. By default this part shows controls to enter general data of the decoder, for example the name and image of the locomotive, which the decoder is installed in. This area of the decoder window may also show special editors for speed table editing or configuration of function maps.

2.2 Selecting the right Decoder Configuration

A decoder configuration is a set of all configuration options, that describe a particular type of decoder or device.

There are specialised decoder configurations, that describe just one specific type of decoder (e.g. a certain locomotive decoder brand or a certain manufacturer). There are also general decoder configurations, that describe common properties of a plurality of similar decoders (e.g. a configuration, that describes common settings of NMRA DCC compatible locomotive decoders).

Decoder configurations do not only contain configuration options. Additionally they contain information, how these options shall be arranged in the explorer window, how these options shall be displayed on the computer screen and how these options shall be edited. In this way a decoder configuration also describes the user interface for interaction with the particular options of the configuration and the values stored in the decoder.

Decoder configurations are distinguished by the name of the manufacturer and the list name of the decoder. Some general configurations are listed with the manufacturer name “Standard”.

Freeware Configuration

B The is a configuration, that can be used for almost all locomotive decoders. It contains the options required to change the address of the decoder, some basic configuration options and to read manufacturer and version information from the decoder, if any.

You can use this configuration free of charge. No license of **TrainProgrammer™** is required to use this configuration for writing the supported options into the decoder. This configuration can be used to change the address of a decoder, to change the speed steps (14 vs. 28) of the decoder or some other very basic settings free of charge.

All other configurations only support reading of the supported options from the decoder, if no license of **TrainProgrammer™** is available. Writing of decoder values with other configurations than the freeware configuration requires a valid license. The freeware configuration is listed in the default database of **TrainProgrammer™** with the manufacturer name “Standard”.

Standard Configuration



The is a configuration, that can be used for almost all NMRA DCC compatible locomotive decoders. It contains the most commonly used configuration options.

This configuration can be used to change and view the address of a decoder, common speed and running characteristics, speed tables, function mappings and other common configuration settings.

Users, especially those which are novice in decoder programming, should preferably use the standard decoder configuration. If in doubt, which configuration to use for a particular decoder, consider to try first, whether the standard configuration doesn't do the job already.



The standard configuration provides a user friendly, intuitive and standardised user interface for access to the most common configuration options of the majority of all NMRA DCC compatible locomotive decoders.



One aspect of the standard configuration has to be kept in mind, however: unlike those specialised decoder configurations, which are customised to the specific characteristics of a certain decoder type, the standard configuration does not take into account any specific limitations with regard to value ranges and uses default or common value ranges instead. The value range of DCC CV1, “Short Address”, for example, is usually 1 to 127. If the address of a certain decoder type must not exceed 99, then this specific limitation of this decoder is not taken into account by the standard configuration. Before writing any value into your decoder you are responsible to ensure, e.g. by referring to the manual of the decoder, that the value to be written is within the permitted range. This does not only, but especially apply to the standard configuration. Usually you will not encounter any problems, though the supplier of this program shall not be liable to you for any problems or damages that may arise, if not allowed values are written into a configuration variable with **TrainProgrammer™** - regardless which decoder configuration has been used.

On the other hand the standard configuration or the use of **TrainProgrammer™** in general does not establish any new hazards, that are not already established by the conventional method of programming of decoders with the handheld of the digital system.

The above statement should therefore be mainly understood to discharge the supplier of this program from liability.

Specialized Configurations



Specialised configurations are customized to a certain type of decoder. Usually they contain the complete set of configuration options supported by this decoder as well as their value ranges.

Experienced users and users, who want to adjust specific characteristics of their decoder, which are not covered by the standard configuration, can use these configurations.

TrainProgrammer™ is delivered with a default decoder database, that contains a number of specialized configurations for frequently used decoder types. The “goody box” of our Internet web site (www.freiwald.com/pages/goody.htm) provides an additional decoder database with more than 400 specialized decoder configurations of popular decoders (courtesy of JMRI).

2.3 Programming the Decoder

The most important tool to read configuration values from the decoder or to write values into the decoder is the *explorer window* in the left part of the decoder window.

Option	Profile	Value		
DCC compatible Standard Decoder	<input type="checkbox"/>			
Basic Configuration	<input type="checkbox"/>			
Address	<input type="checkbox"/>			
Short Address	<input type="checkbox"/>	3		
Four Digit Addressing	<input type="checkbox"/>			
Four Digit Long Address	<input type="checkbox"/>	0		
High Byte	<input type="checkbox"/>	192		
Low Byte	<input type="checkbox"/>	0		
Configurations Supported	<input type="checkbox"/>	0		
Inverted Direction	<input type="checkbox"/>			
28 Speed Steps	<input type="checkbox"/>			
Enable Speed Table	<input type="checkbox"/>			
Four Digit Addressing	<input type="checkbox"/>			
Manufacturer	<input type="checkbox"/>			
Manufacturer	<input type="checkbox"/>			
Version Number	<input type="checkbox"/>	0		
Speed	<input type="checkbox"/>			
Vstart	<input type="checkbox"/>	7		
Vhigh	<input type="checkbox"/>	1		
Vmid	<input type="checkbox"/>	1		
Acceleration Rate	<input type="checkbox"/>	0		
Deceleration Rate	<input type="checkbox"/>	0		
28 Speed Steps	<input type="checkbox"/>			
Enable Speed Table	<input type="checkbox"/>			
Speed Table Values	<input type="checkbox"/>			
Function	<input type="checkbox"/>			

Diagram 14: Explorer Window

The configuration options are arranged in a hierarchical manner. Configuration options (folders) can contain other options. The entries on the lowest level of the hierarchy, i.e. the entries, which do not contain any other entries, correspond to the physical settings, configuration variables (CV), registers or option switches of your decoder. Some of these entries represent only one bit or “binary switch” of one variable or register. In this document the term configuration variable is frequently used as synonym for a physical setting.

With the **Expand** or **Collapse** commands of the **View** menu it is possible to display or hide the contents of particular folders. By clicking to the '+' or '-' markers left of the option names it is also possible to expand or collapse particular folders.

Editing Decoder Values

The displayed values can be changed by clicking to the displayed value in the column labeled **Value**. Depending on the meaning of the selected entry additional controls might be displayed in the lower right part of the decoder window. With these additional controls the decoder values can be changed in a more comfortable, more meaningful way. If, for example, a speed table entry is selected in the explorer window, then the Speed Table Editor is displayed to the right of the explorer window which allows graphical viewing and editing of the speed table instead of entering single numerical values.

Reading Values from the Decoder

Values can be read from the decoder by selecting an entry and using the **Read from Decoder** command of the **Programming** menu. If the selected entry is a folder, then the values of all options contained in this folder are read. If the entry in the first line of the explorer window is selected, then all values of the decoder are read. In this way it is possible to read the complete speed table of a decoder or even all values stored in a decoder in one step.

Writing Values to the Decoder

Values can be written to the decoder by selecting an entry and using the **Write to Decoder** command of the **Programming** menu. If the selected entry is a folder, then the values of all options contained in this folder are written. If the entry in the first line of the explorer window is selected, then all values of the decoder are written.

Profile Mode

Programming of selected entries contained in different folders in one step is also possible by defining a profile of values, which will be read from or loaded into the decoder. This is done by turning on the profile mode with the **Profile Mode** command of the **Programming** menu. After enabling the profile mode the values to be processed can be

selected by checking the related boxes in the column labeled **Profile** and finally using the **Read from Decoder** or **Write to Decoder** command of the **Programming** menu.

The profile mode is also useful for programming a common profile of settings into a plurality of decoders.

Automatic Activation of the Programming Track

TrainProgrammer™ provides 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 **TrainProgrammer™** 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.

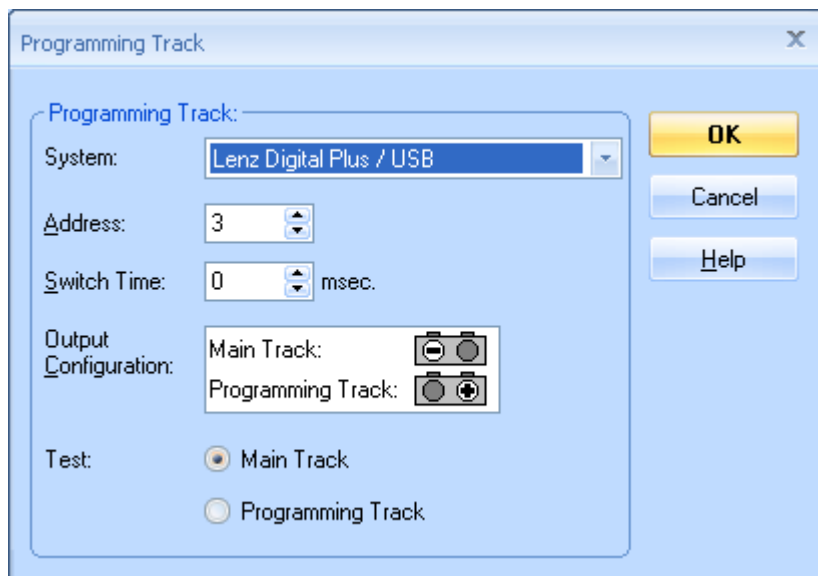


Diagram 15: Arranging the programming Track

Programming on the Main Track

TrainProgrammer™ also supports programming on the main track or main track programming (POM). If programming on the main track is turned on, then all programming commands are sent to a certain decoder on the main track. This decoder is selected by specifying the address of the decoder, the POM address. If this mode is turned off, then all programming commands are sent to the decoder on the separate programming track.

Usually it is only possible to write decoder values to decoders on the main track. With the introduction of (Digitrax) transponding and bidirectional communication, however, there is a growing number of digital systems, which also support reading of values from decoders, that are located on the main track.

Not all decoders or all digital systems support programming on the main track. Please refer to the documentation of your decoder and digital system if it supports this mode.

Meaning of the Status Symbols

The two rightmost columns of the explorer window show option specific status information as outlined below:



The value of this option has been saved to disk.



The value of this option has been changed, but it has not yet been saved to disk.



The value of this option matches the value stored in the decoder. This symbol is displayed after each successful read or write access to the related option in the decoder.



The value of this option might not match the value stored in the decoder. This symbol is displayed after each change of the decoder value.



The last read or write access to the option in the decoder failed.

2.4 Editing Speed Tables

With the speed table editor it is possible to view and draw the speed table of a locomotive decoder. The speed table becomes visible in the lower right part of the decoder window, if an entry associated with the speed table of a locomotive decoder is selected in the explorer window.

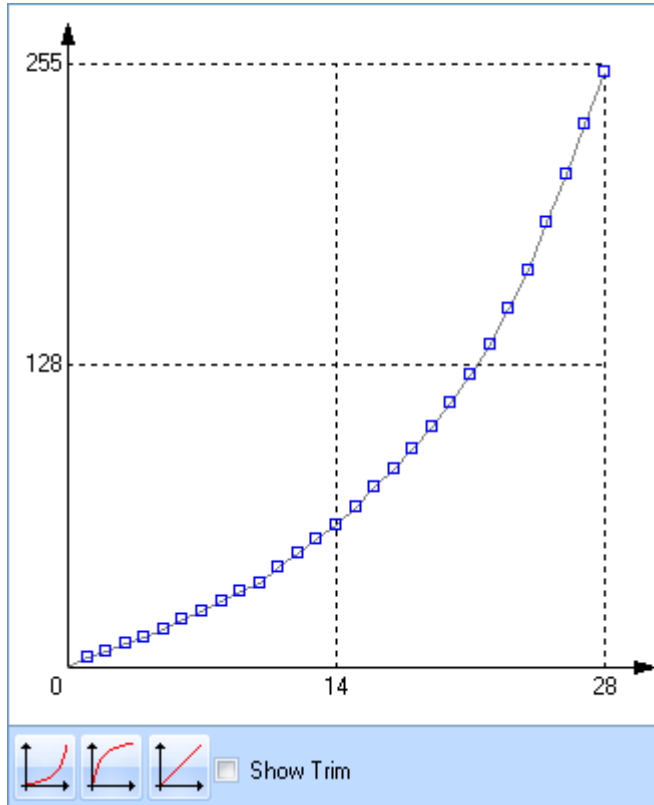


Diagram 16: Speed Table Editor

The curve shows the user defined speed table of the decoder. Each dot can be dragged with the mouse in order to change the speed table.

There are also features to change the overall shape of the curve. The complete curve can be made more or less steep with one mouse click. It is also easily possible to create a linear curve.

By selecting a configuration option in the explorer window, that contains all speed table values, it is possible to read or write a complete speed table from or to the decoder in one step.

2.5 Mapping Function Keys to Output Locations



With this editor it is possible to view and edit the mapping of function keys to output locations of a locomotive decoder. The editor becomes visible in the lower right part of the decoder window, if an entry associated with the function mapping of a locomotive decoder is selected in the explorer window.

	Lf	Lr	F1	F2	F3	F4	F5	F6	F7	F8
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>						
2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>						
3	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
11								<input type="checkbox"/>	<input type="checkbox"/>	
12								<input type="checkbox"/>	<input type="checkbox"/>	
13								<input type="checkbox"/>	<input type="checkbox"/>	
14								<input type="checkbox"/>	<input type="checkbox"/>	

Diagram 17: Mapping Function Keys to Output Locations

The grid of check boxes shows the mapping of function keys to output locations. By setting or removing a tick mark it is possible to link or unlink a function key to an output location.

By selecting a configuration option in the explorer window, that contains all options related to the function mapping, it is possible to read or write the complete map from or to the decoder in one step.

Due to the disadvantage, that the various manufacturers don't follow a common mapping scheme it is not possible to provide this convenient editor in the standard decoder

configuration (see page 33). This editor is only available for specialised decoder configurations.

2.6 Custom Editors



TrainProgrammer™ can be extended by custom editors to support convenient and intuitive editing for specific features of arbitrary decoder types. For this purpose **TrainProgrammer™** offers an open programming interface (API) for plug-in of such custom editors.

There is no limitation with regard to the features and look & feel of such custom editors.

Information about the programming interface for implementation of custom editors for **TrainProgrammer™** are available on request.

3 The Test Panel

The Test Panel is an auxiliary window, that can be opened separately via the **Test Panel** command of the **View** menu.

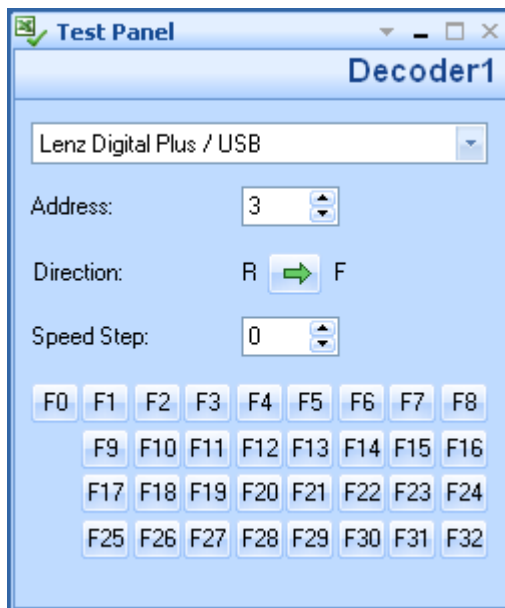


Diagram 18: Test Panel

The Test Panel provides controls for the most important locomotive functions such as speed, direction and auxiliary functions.

The Test Panel can be used to test the settings of the currently selected locomotive directly via the computer screen.

4 The Direct Programmer



The Direct Programmer is an auxiliary window, that can be opened separately via the **Direct Programmer** command of the **View** menu.

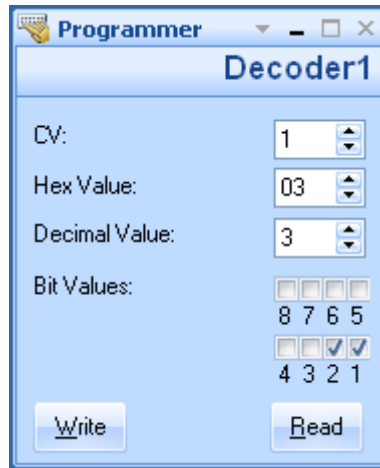


Diagram 19: Direct Programmer

Experienced users can use the Direct Programmer to exchange decimal, hexadecimal or binary option values directly with the decoder by bypassing the explorer window. This feature can also be used to process configuration variables, that are not supported by the currently selected decoder configuration.

In this way the value of a CV, which is not supported by the standard decoder configuration (see page 33), can be read or write on the fly without the need to change the decoder configuration.

5 Run-In of Locomotives



The run-in of new or repaired locomotives is supported by an auxiliary window called „Run-In“.

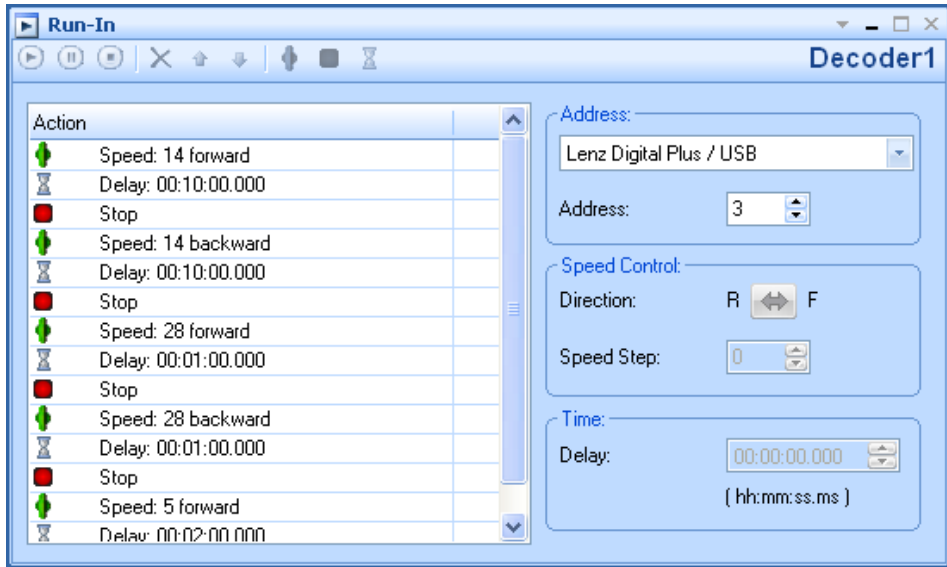


Diagram 20: Run-In

The window provides the ability to create a sequence of actions for the run-in of locomotives. These actions are executed successively to support the automatic run-in of locomotives. Possible actions are:

- **Speed and direction:** this action sets the locomotive in motion in a certain direction.
- **Stop:** stops the locomotive.
- **Delay:** this action inserts a pause between two actions. If the previous action sets the speed of the locomotive, for example, then the locomotive runs with the set speed until the time specified as delay in the subsequent action has passed. In the above illustration the locomotive is started at speed step 14 and runs with constant speed for 10 minutes before it is stopped.

The toolbar of this window provides options to edit the actions in the list to edit and to start, stop or interrupt the execution of the specified actions.

The actions should be performed with a locomotive on a roller test bench or on a circular layout to enable the locomotive to run at constant speed in the same direction for a longer time.

6 Use of a Roller Test Bench



An additional auxiliary window titled “Roller Test” supports programming of locomotive decoders on a roller test bench.

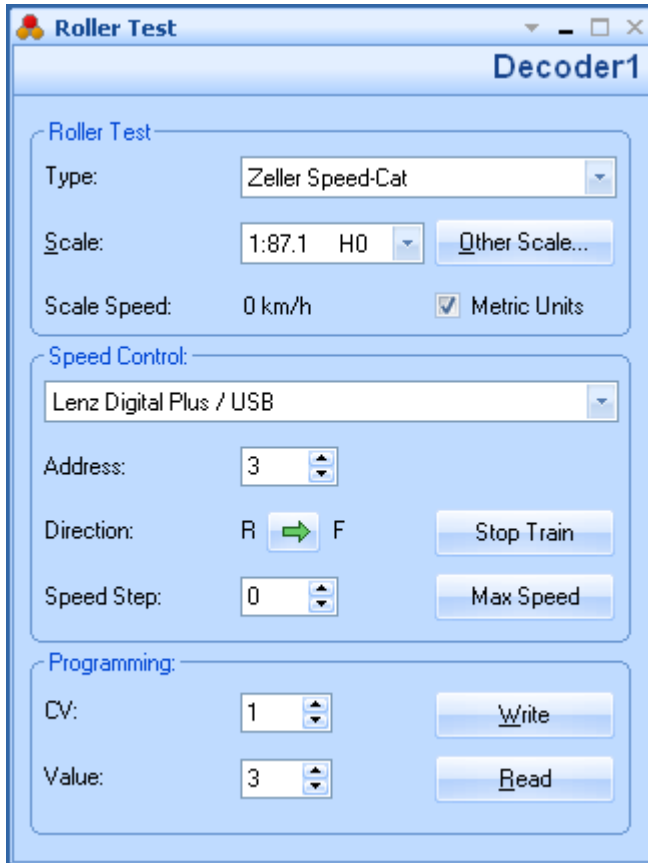


Diagram 21: Programming on a Roller Test Bench

The window offers the possibility to select the type of the roller test bench, to set the speed and direction of the locomotive and to write or read individual decoder settings. While the locomotive runs, its current scale speed (mph or km/h) is displayed. This requires a roller test bench with PC interface and the properly adjusted scale of the locomotive.

The following roller test benches meet this requirement and are currently supported:

- Roller test bench from Zeller with connection to the PC via Speed-Cat

Roller test benches are in particular very useful to adjust the maximum speed of a locomotive very efficiently. If the maximum speed is set and the decoder option (CV), which controls the maximum speed, is chosen, then by writing different values to the decoder the maximum speed can be changed and the effect of the setting can be directly reviewed by inspecting the displayed scale speed..

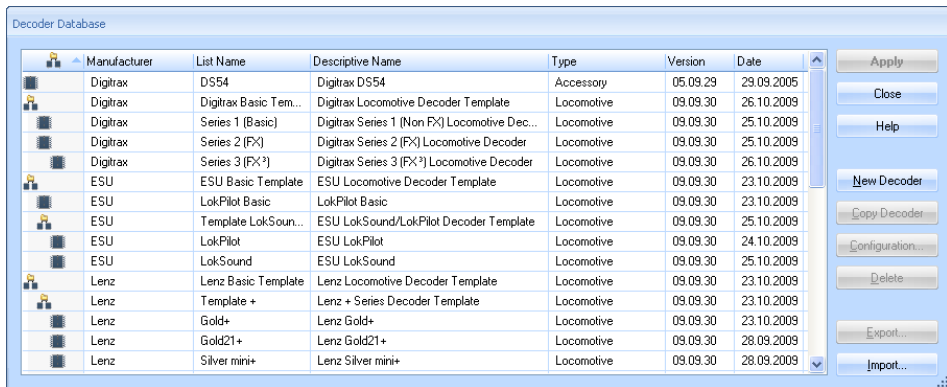
7 The Decoder Database

X

This chapter provides information for experienced users, who want to create their own custom decoder configurations.

TrainProgrammer™ contains a collection of decoder configurations. This collection is called *decoder database*. A decoder configuration is a set of all configuration options, in case of DCC also called decoder variables (CV), of a decoder or device. This description contains among others the name of each option or variable, respectively, a short explanation of its meaning, allowed content, etc. The particular options can be grouped together according to their meaning. In case of a loco decoder, for example, the options, that control the speed of the loco are usually grouped together and separated from the options, that control the auxiliary functions of the decoder.

TrainProgrammer™ is delivered with a decoder database, that contains the configurations of frequently used decoder types. With **TrainProgrammer™** it is also possible for experienced users, to create own decoder configurations, to customise existing configurations or to delete not needed configurations. It is also possible to add decoder configurations created by others, e.g. the manufacturer of the decoder, to the database. Ask the manufacturer of your decoder for configurations, that are not yet contained in the database!



Manufacturer	List Name	Descriptive Name	Type	Version	Date
Digitrax	DS54	Digitrax DS54	Accessory	05.09.29	29.09.2005
Digitrax	Digitrax Basic Tem...	Digitrax Locomotive Decoder Template	Locomotive	09.09.30	26.10.2009
Digitrax	Series 1 (Basic)	Digitrax Series 1 (Non FX) Locomotive Dec...	Locomotive	09.09.30	25.10.2009
Digitrax	Series 2 (FX)	Digitrax Series 2 (FX) Locomotive Decoder	Locomotive	09.09.30	25.10.2009
Digitrax	Series 3 (FX?)	Digitrax Series 3 (FX?) Locomotive Decoder	Locomotive	09.09.30	26.10.2009
ESU	ESU Basic Template	ESU Locomotive Decoder Template	Locomotive	09.09.30	23.10.2009
ESU	LokPilot Basic	LokPilot Basic	Locomotive	09.09.30	23.10.2009
ESU	Template LokSoun...	ESU LokSound/LokPilot Decoder Template	Locomotive	09.09.30	25.10.2009
ESU	LokPilot	ESU LokPilot	Locomotive	09.09.30	24.10.2009
ESU	LokSound	ESU LokSound	Locomotive	09.09.30	25.10.2009
Lenz	Lenz Basic Template	Lenz Locomotive Decoder Template	Locomotive	09.09.30	23.10.2009
Lenz	Template +	Lenz + Series Decoder Template	Locomotive	09.09.30	23.10.2009
Lenz	Gold+	Lenz Gold+	Locomotive	09.09.30	23.10.2009
Lenz	Gold21+	Lenz Gold21+	Locomotive	09.09.30	28.09.2009
Lenz	Silver mini+	Lenz Silver mini+	Locomotive	09.09.30	28.09.2009

Diagram 22: Decoder Database

Each decoder configuration can contain language dependent contents (such as names, tool tips, etc.) for different languages at the same time. The language actually used depends on the language of the user interface of **TrainProgrammer™**, which is selected during installation of the software. The options of the decoder database allow editing of

the language dependent contents for all supported languages. In this way it is possible to create one single decoder configuration for several languages. If such decoder configuration is used in the English version of **TrainProgrammer™**, then the English contents of the decoder configuration are displayed. If the same decoder configuration is used in the German version of **TrainProgrammer™**, then the German contents of the decoder configuration are displayed.

The decoder database is opened with the **Decoder Database** command of the **Railroad** menu. After opening the database it is possible to create or delete decoder configurations, to change configurations or to copy an existing configuration as a starting point for another customised decoder configuration.

It is also possible to import the content of other decoder databases or to export the configurations of selected decoders to a separate database. The latter is useful for users or decoder manufacturers, who created new or customised existing configurations and want to publish these configurations in order to share these configurations with other users.

7.1 Editing of Decoder Configurations

By selecting a decoder configuration in the table of contents of the opened decoder database and calling the **Configuration** command it is possible to edit this configuration.

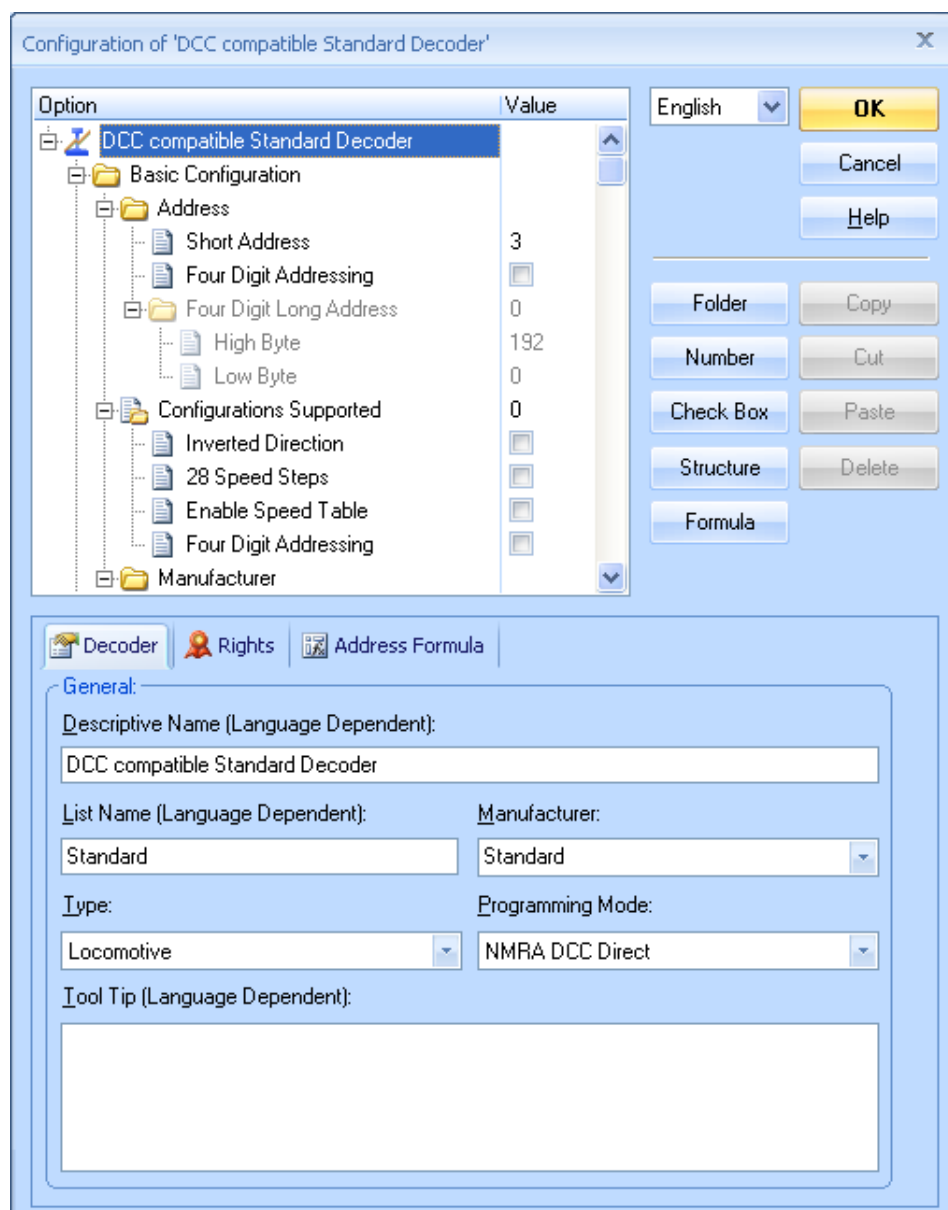


Diagram 23: Editing a Decoder Configuration

Configuration Properties

Each decoder configuration has certain general properties, such as the list name given by the manufacturer, a longer descriptive name, the name of the manufacturer, a specification, whether the configuration describes locomotive or accessory decoders or something else, information how the decoder is programmed and several others.

It is also possible to specify author, copyright and version information and to protect the configuration with a password. Password protected configurations can only be changed, if the password is known. Note, that all configurations delivered in the default database of **TrainProgrammer™** are password protected to protect them from unauthorised changes. These configurations can be copied, however, to allow changes in the duplicates of these configurations.

7.2 Configuration Options

An important information of each decoder configuration is established by the contained configuration options. It is possible to create new options, to delete them, to copy existing options, to move them to another location in the option hierarchy and to edit the content of each option. The particular types of options are described in the following.

Folder

Folders are used to group options together in the explorer window, that belong together according to their meaning, or to separate options from each other. In case of a loco decoder, for example, the options, that control the speed of the loco are usually grouped together and separated from the options, that control the auxiliary functions of the decoder. This is achieved by creating different folders for the two groups of options.

The options contained in a folder are also called sub options of this folder. Folders may contain other folders, too. This is the base for creation of meaningful option hierarchies.

Even though folders are not associated with physical configuration options of the decoder or device, such as configuration variables, folders may be also associated with numeric values. A typical example of such folder is the long address in an NMRA DCC compatible loco decoder, that contains the two numeric options associated with CV 17 and CV 18 as sub options. The folder groups these two options, which are closely related to each other, together. Additionally this folder can be enabled to display the long address, that results from the content of CV17 and CV 18, as plain text. Vice versa changes of the value associated with this folder are then propagated accordingly to the sub options. In this way one can intuitively edit a long address as a plain 4-digit number

without the need to bother with the resulting physical values stored in CV 17 and CV 18.

Because folders are not directly associated with an actual physical configuration option the value displayed with a folder is never stored into the decoder. Instead this value can be provided to make editing of certain configuration options more intuitive. The folder, that represents the long address of an NMRA DCC compatible decoder, for instance, can be used for 4-digit editing of this long address. If the value 1234 is entered here, then the sub options CV 17 and CV 18 are automatically set to 196 and 210, respectively. The value 1234 is not stored into the decoder, though, because the decoder does not offer memory to do that. Instead the two values of CV 17 and CV 18 are stored into the decoder. The value of the folder is provided for intuitive editing of 4-digit numbers.

The values of folders can be displayed as numbers or as a list of selectable choices in the explorer window.

Number

Numbers represent numeric configuration options of the decoder or device or parts of such options. A typical example of such numeric option is the DCC CV 1 in an NMRA DCC compatible loco decoder, that contains the 2-digit digital address of the decoder.

The values of numeric options are displayed as numbers or as a list of selectable choices in the explorer window.

Numbers cannot contain any sub options.

Check Box

Check boxes or binary options represent configuration options, that can only accept two different values, 0 and 1. Such options are usually used to enable or disable a certain function in a decoder or device. A typical example of such binary option or bit is the 6th bit of the DCC CV 29 in an NMRA DCC compatible loco decoder, with which the decoder can be set to the short or long addressing mode.

The values of binary options are displayed as check boxes in the explorer window.

Binary options cannot contain any sub options. It is also not allowed to define binary options without defining a number, structure or formula (see below) for the physical option, which the corresponding bit belongs to. For example it is not allowed to create a binary option/check box for the 6th bit of CV 29 in an NMRA DCC compatible loco

decoder without defining a number or structure for CV29 itself. It is not necessary, though, that the binary option is a direct sub option of the number or structure. The binary option may be located anywhere in the option hierarchy.

Structure

Structures represent numeric options of the decoder or device, that are composed by other sub options. Structures contain other sub options, that are associated with the particular bits of this numeric configuration option. A typical example of such structure is DCC CV 29 in an NMRA DCC compatible loco decoder, that contains global settings of the decoder. The particular settings can be represented as binary sub options or bits. If a certain decoder provides five option switches in CV 29, for example, then this can be represented in the explorer window by a structure representing CV 29 with 5 check boxes or bits as sub options.

The values of structures are displayed as numbers in the explorer window. If the value of the structure changes, then the value of the contained sub options are changed accordingly. If, vice versa, the value of a sub option changes, then the value of the structure changes accordingly, too.

Structures can not only contain binary options, but also numbers. A typical example of such structure is DCC CV 19 in an NMRA DCC compatible loco decoder, that contains the consist address in the lower 7 bits and a setting for direction reversing in the highest bit. This option can be represented in the explorer window by a structure, that contains a seven bit wide numeric option and a check box (bit) as sub options.

Structures must have sub options.

Formula

Formulas represent numeric options of the decoder or device, that are calculated based on a mathematical formula according to the values of their sub options. Formulas contain other sub options. An example of such formula is DCC CV 115 in the ZIMO MX60 loco decoder. It controls the “decoupling definition” and can accept 2-digit values between 0 and 99. The upper digit controls the “full voltage pulse time” and accepts values between 0 and 9. The lower digit controls the “percentage of voltage after pulse”. Setting the lower digit to 3 and the upper digit to 7 results in a value of 73 for this CV. In other words: the value of the CV is calculated according to the formula $10 \cdot A + B$, where A represents the value of the “full voltage pulse time” and B represents the value of the “percentage of voltage after pulse”.

The values of formulas are displayed as numbers in the explorer window.

Formulas must have sub options. These sub options are numbers (see above).

Overview of Option Types

The following table summarises the meaning of the particular option types:

Type	Has Sub Options	Displayed in the explorer window as	Corresponds to a physical configuration option (such as a CV)	Purpose	Example (NMRA DCC)
Folder	Yes	Number or List	No	Groups related options together.	Folder, that contains all options related to loco speed
Number	No	Number or List	Yes	Represents numeric values, that are stored in an physical configuration option (CV) or in a part of such option.	CV1, CV18, Consist Address stored in the lower 7 bits of CV19
Check Box	No	Check Box	Yes (one bit)	Represents a single bit of a physical configuration option.	Bits of V29, Highest bit of CV19
Structure	Yes	Number	Yes	Represents an option, that is put together bitwise by other options.	CV29, CV19
Formula	Yes	Number	Yes	Represents an option with values, that are calculated based upon a mathematical formula	CV115 (Zimo MX60)

Properties of Configuration Options

Each configuration option is described by a set of properties. These properties fall in different categories. These properties describe, how the option is displayed in the explorer window, how the value of the option is edited and how the value of the option is linked to the value stored in the associated physical option (CV), if any. The properties therefore describe the look & feel of the option and the characteristics of its storage.

General properties specify the name, tool tip help information and other attributes of an option. Physical properties describe the number of the associated physical option (CV number), the value range, the occupied bits etc. It is furthermore possible to specify a formula for the calculation of the option value (e.g. DCC long address, which is calculated by the values of CV17 and CV 18) or to specify a list of textual choices, if the value of the option shall be intuitively edited via a list of possible choices.

It is also possible to specify other options, which enable or disable the selected option depending on their current value. The options associated with the long address of an NMRA DCC locomotive decoder, for example, can be disabled in the explorer window, when the 6th bit of DCC CV 29, which selects the short or long addressing mode, is cleared.

Finally it is possible to specify, whether a specific editor, for example the speed table editor (see section 2.4), shall be displayed in the decoder window, when this option is currently selected. This includes also the possibility to link the option to a custom editor (see section 2.6).

Reference

References represent almost exact copies of other options. References do not establish an own type of option. They can be used, however, to reduce the work to create a series of almost identical options. An example are configuration options, which describe the configuration of auxiliary functions of a DCC locomotive decoder. In many cases there is a configuration option or a group of options (preferably grouped in a folder), which belongs to a certain physical function output or a logical DCC function (such as F1, F2, ...). In total all function outputs or all logical DCC functions are represented by several options or groups of options, which differ only by their name and their number (e.g. CV number). In order to avoid multiplication of almost identical configuration options, which differ only by name and number, it is sufficient, to create only one of these options completely first and to create a reference for the second, third, and so on almost identical option, which refers to the first option.

References are useful, if a series of options is to be created, where the particular options only differ by name and number. It is also possible to create references to folders. In this case the names of all options contained in this folder are derived from the names of the original folder. The numbers of these options are adjusted automatically.

Unlike copies of options created by means of the Windows Clipboard, references have the advantage that an effect of subsequent revisions and changes in the original option automatically apply also to all associated references.

Indexed Options

Some decoders have so many settings that a large number of configuration options (CVs) is required. On the other hand the number of options, which are directly addressable over the track signal, is limited. In order to store more options in a decoder than these limits allow, a so-called “indexed access” is done. This means that the meaning and content of a CV changes, depending on the value of a so-called “index register”. By changing the value in the index register the value and the meaning of the indexed option changes, too. In this manner, an option with a certain number (for example the CV 280) can be used several times and solves the lack of directly addressable CVs.

For example, if the CV 260 is used as an index register for the CV 280, then with each change of CV 260 the meaning of CV 280 changes, too. If CV 260 supports the five values from 0 to 4, then five different values with different meanings can be stored in CV 280. To write a specific value to CV 280 or to read from it, the corresponding value between 0 and 4 must be previously written to CV 260. If 0 is written into CV 260, then the first value in CV 280 can be accessed, if 1 is written into CV 260, then the second value in CV 280 can be accessed, and so on.

In the decoder database of **TrainProgrammer™** indexed options can also be arranged. During the programming process, the necessary values are always automatically written to the options, that are used for indexed access, before accessing an indexed option.

In the decoder database it is for example possible to set CV 260 as the index register for CV 280. For the five values from 0 to 4 five different configuration options can be defined and associated with CV 280. These five options may have different names, meanings, values, ranges and other properties. For each of these options it will be specified, which CV acts as the index register (in this example, CV 260), and which value (in this example, from 0 to 4) must be written into the index register to access that option. The same option can also be indexed with more than one index register. In this case, several values must be written into these index registers prior to programming the indexed option.

In **TrainProgrammer**TM an index register may even be indexed itself. However, so far no decoder is known, which uses this. Furthermore, it is also possible to create references to indexed options and thus to reproduce indexed options with little effort.

The supply of the index registers with the correct values is automatically performed by **TrainProgrammer**TM during the programming process prior to accessing an indexed option. For the end user, the programming of indexed options is therefore practically as comfortable and easy as the access to regular, non-indexed options.

7.3 Multilingualism, Derivation, Templates

The features outlined in this section are useful for editors of decoder configurations, which are not only created for personal use, but also intended to be used by a plurality of other users.

Multilingual Decoder Configurations

Each decoder configuration can contain language dependent contents (such as names, tool tips, etc.) for different languages at the same time. The language actually used depends on the language of the user interface of **TrainProgrammer**TM, which is selected during installation of the software. The options of the decoder database, however, allow editing of the language dependent contents for all supported languages. In this way it is possible to create one single decoder configuration for several languages. If such decoder configuration is used in the English version of **TrainProgrammer**TM, then the , **TrainProgrammer**TM, then the English contents of the decoder configuration are displayed. If the same decoder configuration is used in the German version of **TrainProgrammer**TM, then the German contents of the decoder configuration are displayed.

The currently supported languages are English and German.

Derivation

It is possible to derive a decoder configuration from another. This is very useful and time saving if several similar decoder configurations are to be created. In many cases the decoder configurations of the decoders of the same manufacturer are very similar, if not even identical. In such case it is useful to create a decoder configuration for one *typical* representative of this *family* of similar decoders first and then to derive the other decoders of this family from the first.

Initially derivation causes all decoder options of the base decoder configuration to be inherited by the derived configuration. It is furthermore possible to modify selected options in the derived configuration in order to support those parts of the derived configuration, that are different to the base configuration.

The following modifications are possible in derived configurations:

- Options, which are contained in the base configuration, but which are not supported by the derived configuration, can be **excluded** from the derived configuration.
- New options, which are supported by the derived configuration, but not contained in the base configuration, can be **added**.
- Options of the base configuration including their child options can be completely **replaced** in the derived configuration. This causes the same effect as excluding an option of the base configuration and adding a new one.
- Selected parts of the content of an option of the base configuration can be **overwritten** in derived configurations. This is useful, if the derived option is almost identical to the base option with just small differences like default value, minimum or maximum allowed values, names or tool tips, etc.

Templates

Templates complete the concept of derivation. Assume a family of almost identical decoders with a large subset of common options. In such cases it is useful to create a decoder configuration, which represents the common subset as a base configuration and to derive all other configurations from this base. If the base configuration, however, does not represent a real decoder itself, this configuration can be marked as a **template**. Such Templates are only used to derive other decoders from them in the decoder database, but they are not provided for programming of decoders during normal use of **TrainProgrammer™**.

Appendix: Troubleshooting

Problems during Reading and Writing of Decoder Values

Problems during reading and writing of decoder values are in most cases caused by contact problems between track and wheels. With **TrainProgrammer**TM usually a group of values is read or written, while with the digital system only one value can be processed in one step.

And this makes the difference!

For programming with the digital system the locomotive is put on the track, one decoder value is written or read and if this does not work at once the locomotive is slightly pushed and everything works fine.

With **TrainProgrammer**TM the locomotive is put on the track and reading/writing is started for a bunch of decoder values. In many cases this works for several values and after processing some values the process is interrupted and an error message is displayed. But the small push makes the difference!! During programming the locomotive performs small micro movements. For this reason tracks and wheels must be very (!) clean. If this is not the case the locomotive might lose contact during processing of several decoder values. Error messages or wrong decoder values are never generated by **TrainProgrammer**TM itself. This information is passed directly from the digital system to the user interface.

Compatibility problems

TrainProgrammerTM does not communicate with the decoder directly. Instead it sends a command to the digital system, that directs the digital system to send commands to the track for reading or writing of decoder values. In case of read or write errors the digital system sends an error message back to the computer. This error message is finally displayed by **TrainProgrammer**TM. If contact problems as outlined above can be ruled out as reason of read/write problems, then check with the manufacturer of the decoder or the digital system, whether there are known problems with regard to the compatibility between both.

Index

- binary option 52
- check box 52
- configuration
 - decoder configuration 26
- configuration option 26
- custom editors 41
- database
 - decoder database 27, 48
- decoder 26
- decoder configuration 26
- decoder database 27, 48
- decoder file 27
- decoder programming 34
- decoder window 28, 31
- derivation 57
- device 26
- direct programmer 43
- edit decoder values 36
- editors
 - custom editors 41
- explorer window 31, 34
- file
 - decoder file 27
- folder 51
- formula 53
- function keys
 - map to output locations 40
- function outputs
 - mapping to function keys 40
- indexed configuration option 56
- main track programming 38
- map
 - function/output map 40
- multilingual decoder configurations 57
- number 52
- option
 - configuration options 26
- POM 38
- POM address 38
- profile mode 36
- programming
 - decoder 34
- programming on the main track 38
- programming track 37
- read decoder values 36
- reference 55
- roller test bench 46
- run-in 44
- speed table 39
- structure 53
- style of user interface 28
- template 58
- test panel 42
- user interface
 - style 28
- window
 - decoder window 28, 31
 - explorer window 31, 34

write decoder values 36