



**DCC locomotive
decoder
Lokommander II**

User manual
- version 0.1.22 -



No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, without the written permission of Tehnologistic Ltd.

Subject to technical modification



Please read this manual carefully before carrying out the installation!!! Although our products are very robust, incorrect wiring may destroy the module!



During the operation of the device the specified technical parameters shall always be met. At the installation the environment shall be fully taken into consideration. The device must not be exposed to moisture and direct sunshine.



A soldering tool may be necessary for the installation and/or mounting of the devices, which requires special care.



During the installation it shall be ensured that the bottom of the device should not contact with a conductive (e.g. metal) surface!



Content

| | | |
|---------|--|----|
| 1. | Important information..... | 4 |
| 2. | Abbreviations..... | 5 |
| 3. | Contents of the manual | 5 |
| 4. | Main Features | 6 |
| 5. | Technical Specifications..... | 7 |
| 6. | General description of Lokommander II decoders | 8 |
| 7. | Installation of decoder | 13 |
| 8. | Commissioning..... | 15 |
| 9. | Decoder address..... | 16 |
| 10. | Ascertainment of rolling characteristic..... | 18 |
| 10.1. | Linear speed adjustment in 3 points..... | 18 |
| 10.2. | Tabular speed adjustment in 28 steps..... | 19 |
| 11. | Engine Control..... | 20 |
| 12. | Controlled stops..... | 25 |
| 12.1. | Constant brake distance (CBD)..... | 25 |
| 12.1.1. | Fixed deceleration stop | 26 |
| 12.1.2. | Variable deceleration stop..... | 26 |
| 12.2. | Detecting asymmetric DCC signal (Lenz ABC) | 26 |
| 12.3. | Penduling function (Push-pull) | 27 |
| 12.3.1. | Without intermediate stops | 27 |
| 12.3.2. | With intermediate stops | 28 |
| 13. | Function outputs | 28 |
| 14. | Analog operation (DC) | 33 |
| 14.1. | Analog mode 1 | 33 |
| 14.2. | Analog mode 2 | 34 |
| 14.3. | Controlled stop on DC sector | 34 |
| 15. | Bidirectional communication (RailCom) | 35 |
| 16. | Special functions..... | 35 |
| 17. | Electric Uncoupler Configuration..... | 37 |
| 18. | SUSI / Locowire interface | 40 |
| 18.1. | Programing SUSI modules..... | 40 |
| 19. | Using external capacitors or a power pack | 42 |
| 20. | Resetting the decoder..... | 44 |

| | | |
|-----|--|----|
| 21. | Secondary address (decoder lock) | 45 |
| 22. | Firmware update | 46 |
| 23. | Special firmware version for 3V engine | 46 |
| 24. | Accessories | 48 |
| 25. | Technical support..... | 48 |
| 26. | The decoder CV table | 49 |
| 27. | Bits and bytes..... | 88 |

1. Important information



Please read this first chapter

- Lokommander II decoders are designed exclusively for use in model trains. Any other use is forbidden.
- Any connection must be made without the connected power supply. Please make sure that during installation, the locomotive is not powered, not even accidentally.
- Avoid applying blows or mechanical pressure on the decoder.
- Do not remove the heat shrink tube from the decoder (on models fitted with a protective sleeve).
- Ensure that neither the Lokommander II decoder nor the unused wires do not come into electrical contact with the locomotive chassis (short-circuit risk). Insulate the ends of any unused wires.
- Do not solder extension cables on the decoder circuit board except in cases strictly necessary (connections to sound modules, power packs).
- It is forbidden to wrap the decoder in an insulating tape, as this may cause overheating.
- Observe the wiring of the decoder and any external components as recommended in this manual. Wrong wiring / connection can cause damage to the Lokommander II decoder.



- Make sure that there is no wires trapped by the locomotive transmission system when reassembling it.
- Any power source used must be protected by a fuse or electronics to avoid any danger that may arise in a short circuit. Use only transformers or power supplies specially designed for electric trains.
- Do not let Lokommander II decoders to be used by unattended children. Lokommander II decoders are not a toy.
- Do not use Lokommander II decoders in wet environments.

2. Abbreviations

| | | |
|------|---|-------------------------------------|
| DCC | - | Digital Command Control |
| DC | - | Direct Current |
| NMRA | - | National Model Railroad Association |
| CV | - | Configuration Variable |
| PT | - | Programming Track |
| PoM | - | Programming On the Main |
| ABC | - | Automatic Brake Control |
| CBD | - | Constant Braking Distance |
| MSB | - | Most Significant Byte (or Bit) |
| LSB | - | Least Significant Byte (or Bit) |
| FL | - | Front Light |
| RL | - | Rear Light |
| SPP | - | Smart Power Pack |
| n.c. | - | not connected |
| BEMF | - | Back Electro-Motive Force |
| MI | - | Maintenance Interval |

3. Contents of the manual

We congratulate you for purchasing a Lokommander II decoder. This manual is divided into several chapters, which show you step-by-step how to install and customize a Lokommander II decoder. Chapter 4 and 5 provide an overview of the features and parameters



of the decoders. Chapter 6 contains the general description of the decoders. Chapter 7 describes in detail the installation of decoders in locomotives. Please familiarize yourself with the type of engine and the type of interface existing in the locomotive before going through this chapter. Lokommander II decoders can be operated with most commercially available control systems for electric train models.

Factory default values for configuration variables (CVs) and functions are found in Chapter 9. You can change the default settings of your Lokommander II decoder as desired. Chapters 10 - 16 explain configurable parameters, and how to customize them. We recommend reading the chapters 10-12 for setting the address and engine control parameters to be able to customize the decoder for your locomotive optimally.

Chapter 26 contains all the CVs of the decoders.

4. Main Features

- NMRA compliant Generic DCC mobile decoder
- PT or PoM programming modes
- Operation even in analog mode (DC), with configurable active functions
- Short (1-127) and long (128-9999) configurable addresses
- 14, 28/128 speed steps
- Maximum motor current 1000mA
- Load compensation and BEMF
- Speed set at 3 points (V_{min} , V_{mid} , V_{max}) or in tabular form
- Shunting speed (switchable from F3, CV114)
- Acceleration / Deceleration inhibition (switchable from F4, CV115)
- Constant distance braking, activated on ABC or DC sector or at zero speed
- Reduced speed drive on ABC Slow Speed sector
- Penduling function (Push-Pull)
- up to 10 dimmable auxiliary outputs, maximum current 300mA



- Output Mapping to functions F0, F1-F12
- Short-circuit and over-current protection of motor and auxiliary outputs
- Bidirectional communication RAILCOM
- SUSI© and LocoWire© interface
- Outputs for Smart Power Pack (SPP ©)
- Electromagnetic un-coupler drive capability
- Upgradable software via the programmer, even with the decoder mounted in the locomotive
- Reduced dimensions allow for use on the scale H0, TT and N

5. Technical Specifications

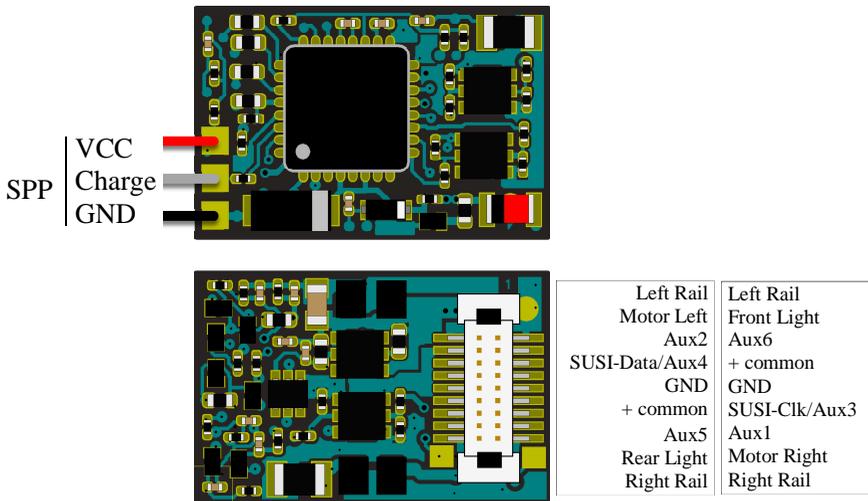
- Supply voltage: 4-24 V, (DCC voltage in the tracks)
- Current consumption without activated outputs: <10 mA
- maximum current for each output: 200 mA
- maximum total current for the decoder: 400 mA
- Dimensions (without wires and connector):

| | |
|-------------------|---------------|
| NEM651: | 14x9x3,3mm |
| NEM652,PLUX12/16: | 19,5x11x3mm |
| NEXT18: | 14,2x9,2x3mm |
| MTC21: | 20x15,3x5mm |
| PLUX22: | 20,5x15x3,5mm |
- weight: 4-6 g
- protection class: IP00
- Operating temperature: 0 ÷ +60 °C
- Storage temperature: -20 ÷ +60 °C
- Humidity: max 85 %

6. General description of Lokommander II decoders

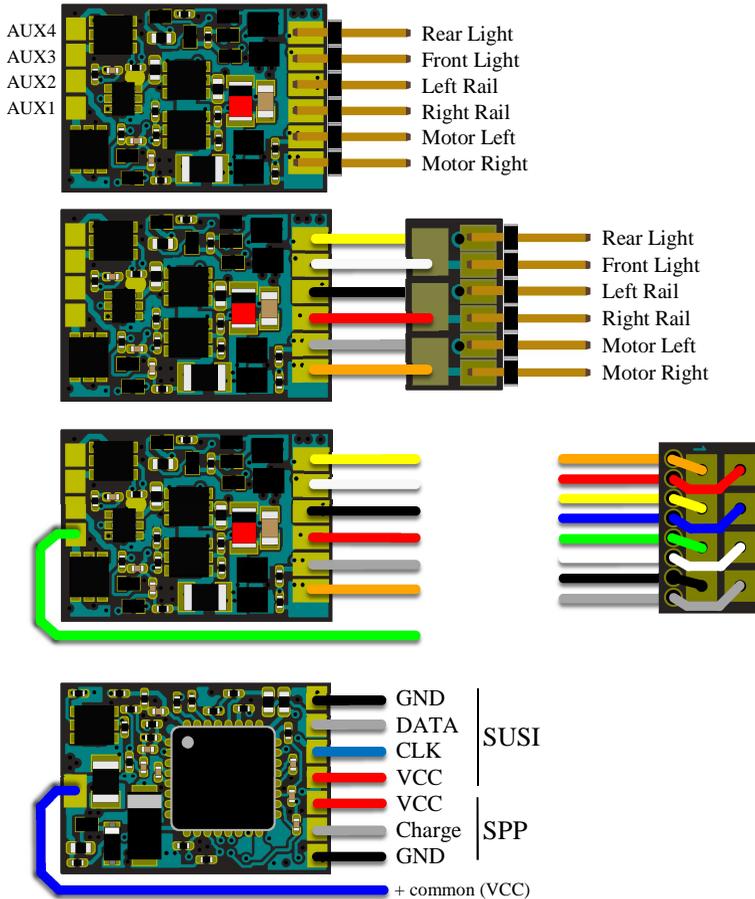
Lokommander II decoders are designed to be used in N, TT, H0, H0e scale models. The different models are differentiated by the physical size, the connector type, the current supplied to the engine and the number of auxiliary outputs available. From the point of view of functioning and programming, they are identical.

The NEXT18 version is 14.2x9.2x3mm



Lokommander II with NEXT18 connector

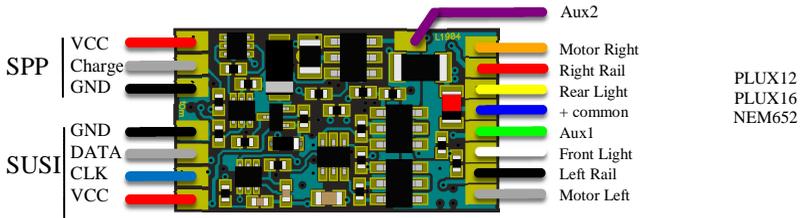
The MICRO version is 14x9x3.3mm and can be delivered with either NEM651 6-pin right or straight-angle connector, soldered directly to the board, or with NEM651/NEM652 connector with wires.



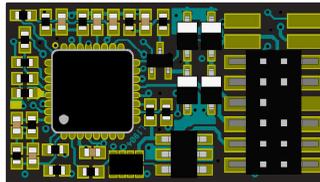
Lokommander II MICRO



The PLUX16 version is 19.5x11x3mm can be delivered with PLUX16 15-pin connector, with PLUX12 11-pin connector or 6-pin NEM651 connector with wires.

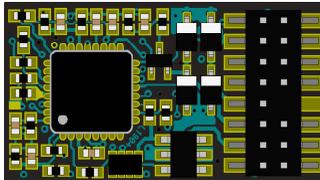


PLUX12

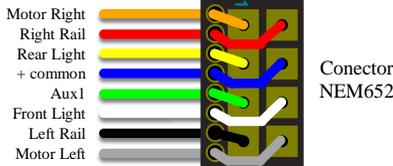


- | | |
|-------------|-------------|
| Motor Right | Front Light |
| Motor Left | + common |
| Right Rail | INDEX |
| Left Rail | Rear Light |
| Aux1 | Aux3 |
| Aux2 | Aux4 |

PLUX16

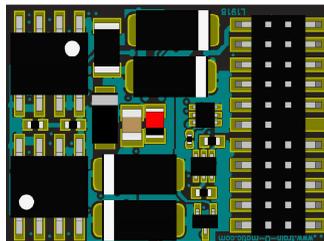
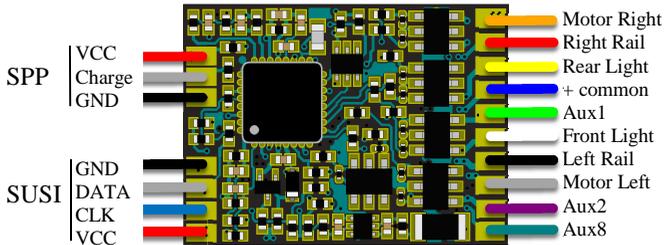


- | | |
|-------------|-------------|
| SUSI-Data | SUSI-Clk |
| + Common | GND |
| Motor Right | Front Light |
| Motor Left | + common |
| Right Rail | INDEX |
| Left Rail | Rear Light |
| Aux1 | Aux3 |
| Aux2 | Aux4 |



Lokommander II with PLUX12/16 or NEM652 connector

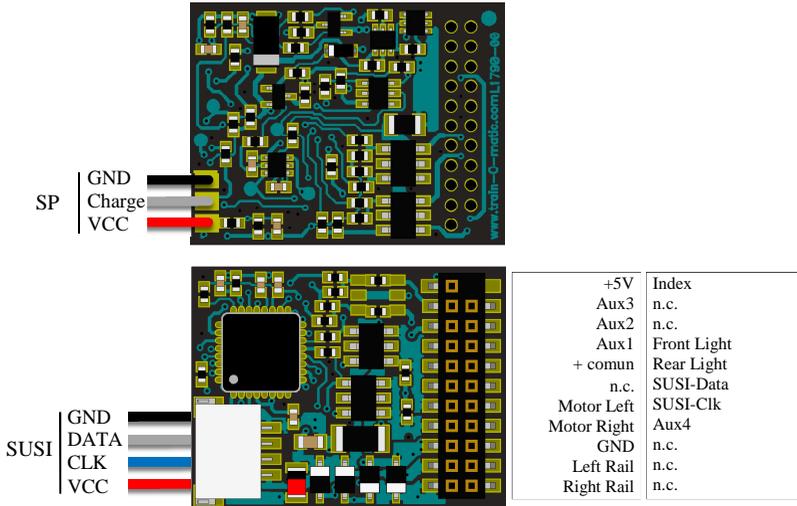
The PLUX22 version is 14.2x9.2x3mm and can be delivered with 21 pin PLUX22 connectors or with 6-pin NEM651 connectors with wires.



| | |
|-------------|-------------|
| Aux3 | Aux8-I/O |
| SUSI-Data | SUSI-Clk |
| + common | GND |
| Motor Right | Front Light |
| Motor Left | + common |
| Right Rail | INDEX |
| Left Rail | Rear Light |
| Aux1 | n.c. |
| Aux2 | n.c. |
| Aux5 | Aux4 |
| Aux6 | Aux6 |

Lokommander II with PLUX22 connector

The decoder version MTC21 is 20x15.3x5mm. It can be delivered with or without a 4-pin SUSI connector (the connector version has the suffix S, MTC21S).



Lokommander II with MTC21 connector

From the table 1. you can find the identification code for each Lokommander II variant.

| Format | Connector | tOm order code | Dimensions without connector |
|--------|-----------------|----------------|------------------------------|
| MICRO | NEM651 straight | 02010220 | 14x9x3,3mm |
| MICRO | NEM651 in angle | 02010221 | 14x9x3,3mm |
| MICRO | Wires + NEM651 | 02010222 | 14x9x3,3mm |
| MICRO | Wires + NEM652 | 02010223 | 14x9x3,3mm |
| NEXT18 | NEXT18 | 02010216 | 14,2x9,2x3mm |
| PLUX22 | PLUX22 | 02010217 | 14,2x9,2x3mm |
| PLUX22 | Wires + NEM652 | 02010218 | 14,2x9,2x3mm |
| PLUX16 | LPUX16 | 02010211 | 19,5x11x3mm |
| PLUX16 | PLUX12 | 02010210 | 19,5x11x3mm |
| PLUX16 | Wires + NEM652 | 02010212 | 19,5x11x3mm |
| MTC21 | MTC21 | 02010208 | 20x15,3x5mm |
| MTC21 | MTC21 + SUSI | 02010209 | 20x15,3x5mm |

Table 1.

7. Installation of decoder

Before installing a digital decoder, especially in older used models, it is a good idea to make sure that the locomotive is operating properly in DC. To do this, perform the following operations:

- Clean of wheels and blades
- Check the engine condition, measure the idling current of the motor powered by 5-10V, which should not exceed 200-300mA, if necessary clean the brushes and the collector.
- Check the transmission system, if necessary, clean and lubricate the axes and the sprockets.
- If the locomotive is equipped with lighting bulb , check that it resists the voltage of 16V, if necessary replace bulbs.

In the case of locomotives prepared for digitization, the installation of the decoders echiped with the connector (PLUX, MTC, NEXT18, MICRO-6, NEM652) is done by extracting the dummy module for analog operation from the connector on the motherboard. In the thus released connector, insert the decoder by following the key (INDEX) to the PLUX and MTC or, if any, the instructions received with the locomotive.

If the NEXT18 decoder is inserted in inversely, direction of travel and the directional lights will be inverse with each other, without the risk of decoder damage.

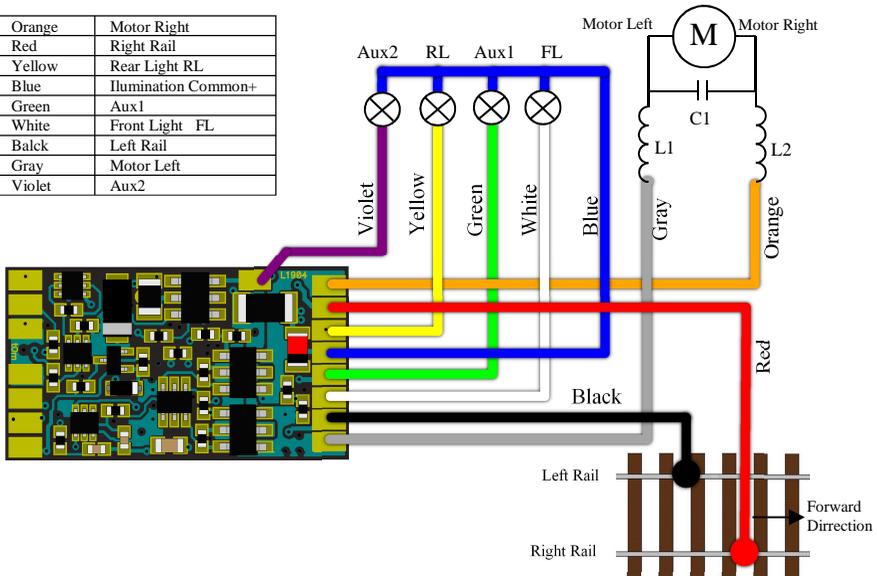
The decoder with NEM651 connector is reverse-tolerant, but it will not work at all.

At the NEM652 connector, the pin at witch the orange wire is connected, will enter the hole 1 marked on the locomotive base plate. Reverse connection does not damage the decoder, the engine will turn in reverse direction and the lights will not work.

In older locomotives that are not ready for digitization, a non-connector decoder can be installed, which has only connection wires.

The color of the wires has an important meaning, the connections will be as shown in the following drawing.

| | |
|--------|----------------------|
| Orange | Motor Right |
| Red | Right Rail |
| Yellow | Rear Light RL |
| Blue | Illumination Common+ |
| Green | Aux1 |
| White | Front Light FL |
| Black | Left Rail |
| Gray | Motor Left |
| Violet | Aux2 |



Firstly establish the forward direction of the locomotive. The connections between the motor and the collector blades are disengaged, retaining the correlation between the motor terminals and the left / right rail. If it exists, the capacitor C1 on the motor terminals is not good to exceed 47nF. If there are no filtration coils, remove even the capacitor C1. It is also eliminated if there are capacitors mounted between the motor terminals and the metallic chassis of the motor. Solder the red / black wires to the wheel collector blades, and the orange / gray wires to the motor terminals respecting the forward direction (right / left).

Lighting or other auxiliary consumers will be connected between the blue (Common +) wire and the wire corresponding to the desired output (FL, RL, Aux1, Aux2, etc.). If these consumers are polarized, for example LEDs, pay attention to the polarity. The positive



terminal (Anode) will be connected to the +Common, the negative terminal (Cathode) to the desired output. Is mandatory that the LEDs to be connected with a current limiting resistor of 1-33K depending on the desired maximum luminous intensity

8. Commissioning

Prior to powering from the digital control station, check the correctness of the connections made to the decoder installation, make sure that there are no short circuits or faulty connections. Check that the connecting wires do not come into contact with the gears or moving parts of them.

At the first power ON of the decoders, it is recommended to perform a reset by entering the value 8 in CV8 to make sure that we start from factory default values of the configuration variables.

Set the address of the locomotive stored in CV1, the initial value being 3, or set an extended address as described in Chapter 10. If we also want to use the consist address, it is recommended to enter it only after all other parameters have been set and tested on the primary address.

When writing/reading CVs on the programming track the decoders confirm to the command station the execution of the received order by issuing acknowledge (confirmation) pulses. During the pulse it is necessary to increase the current consumption of the decoder over 100mA. This is usually done by powering the engine, but there may be situations where this is not possible or the engine consumption is below the required value. In these cases CV165 can configure one or more of the first 8 outputs to send the acknowledgment pulse. At the selected output(s), a consumer (resistor) of the required value must be connected to reach the 100mA current.

9. Decoder address

The Lokommander II decoder can be used either with short addresses (1-127) or long addresses (1-9999). The factory default is short addressing (bit 5 of CV29 is 0), with the address 3 (CV1=3).

The address can be changed by placing the decoder on the Programming Track (PT), and changing the CV1 value, according to the instructions of your Command Station.

If long addressing is needed, the addressing mode has to be changed in the configuration CV of the decoder (bit 5 of CV29). Changing the bit5 value of CV29 to 1 will activate the long addressing mode, and the decoder will respond to the long address specified in CV17 and CV18. Bit5 has a decimal value of 32, so changing bit5 to binary 1 is equivalent with the adding of 32 to the decimal value of CV29 (CV29 has a factory default value of 10, activating bit 5 means, to add 32 to this value, $10+32 = 42$, the new value for CV29 will be 42).

The long addresses will be calculated with the following algorithm (in our example we will consider the long address 2000)

-divide the desired long address with 256 (in our example $2000/256= 7$, remainder = 208)

-add 192 to the result and program it in CV17 ($7+192=199$, program the value of 199 in CV17)

-program the value of the remainder of the division in CV18 (program the value of 208 in CV18)

After programming CV29, CV17 and CV18 to the mentioned values, the decoder can be accessed with the address 2000. To switch back to short addressing, the bit5 of CV29 has to be deactivated.



If we write CV1, the consist address will be automatically deleted and the extended address will be automatically disabled!

Consist address

The Lokommander II decoder can use the Advanced Consist functions. To activate this feature, the consist address has to be set in CV19. When the content of CV19 differs from 0, the decoder will perform functions that are defined in CV21 and CV22 only if they are transmitted to the consist address. All other functions will be performed while they are sent to the base address (defined in CV1 or CV17/18).

Functions declared in CV21(F8-F1), CV22(0,0, F12-F9, F0R, F0F) will not be performed while they are transmitted to the base address. For bit value 0 the function will only be active with the individual address, for value 1 the function will only be active with the consist address.

For example, if we want to use F0 in both directions and F3, F4 with consist address, we will write in CV21 = 12 (00001100) and in CV22 = 3 (00000011).

Consists is useful if we want to run two or more engines in the same train (this means several mobile decoders), as well as multiple traction and want to perform some of the functions individually for each decoder, and other functions globally for all of the decoders.

Speed and direction commands will be sent to all decoders within the same consist. In this way the headlights (of locomotives) and tail light of carriages can be turned on and off, based on the direction commands sent to the consist addresses, while the interior lights in different carriages can be turned on and off based on their individual base addresses.

 Only functions F0, F1-F12 can be used in consist mode. The speed steps setting in CV29 must match the speed step setting of the command station for both base and consist addresses.

10. Ascertainment of rolling characteristic

In this chapter we will describe considerations related to setting the minimum, medium and maximum speed, acceleration and deceleration rates of the locomotive:

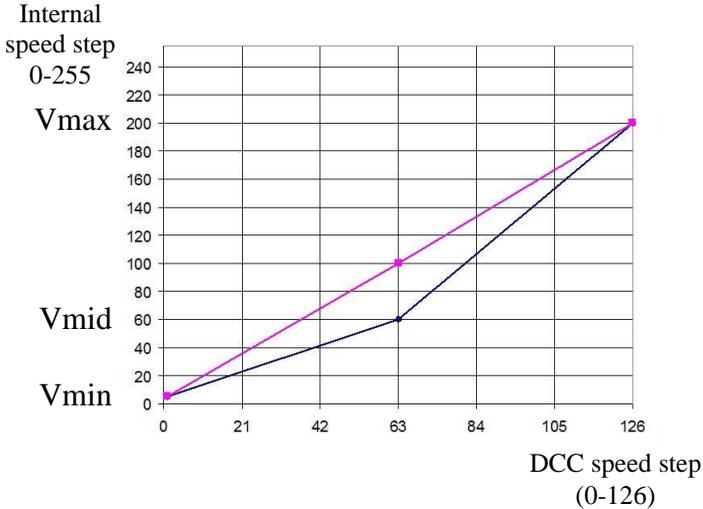
- CV2: engine speed at the lowest speed step
- CV5: engine speed at the highest speed step
- CV6: engine speed at medium speed
- CV3: Acceleration rate
- CV4: deceleration rate

For optimal engine control are recommended usage of 128 speed steps (in DCC). If this is not possible, the decoders also accept DCC commands with 28 or 14 speed steps, but the speed adjustment will be more crude in larger jumps.

The engine speed is set internally in 255 steps. The correlation between the DCC steps and the internal steps can be done in two ways.

10.1. Linear speed adjustment in 3 points

The minimum speed CV2 and maximum speed CV5 are the limits between which we can adjust the engine speed between the first and the last DCC speed step. The engine speed is determined linearly along two straight lines delimited by V_{min} - V_{mid} and V_{mid} - V_{max} respectively. Typically, the first straight segment is selected with a lower slope to have a fine control at slow speeds. This is obtained by choosing for V_{mid} a value less than the average value of the minimum and maximum speeds ($V_{mid} < (V_{min} + V_{max})/2$). If V_{mid} is set to 0, then the average value of the minimum and maximum speed $(CV2 + CV5) / 2$ will be used and the two straight segments will join together to form a single segment.



10.2. Tabular speed adjustment in 28 steps

The engine speed setting is based on the table contained in the CV area, the first step corresponds to CV67 and at the last CV94. By choosing the values in the table, any shape for the engine speed control curve can be set.

If you want fine tuning and speed differentiation according to the travel direction, we can use CV66 for the forward and CV95 for the backward direction. For initial value 0, these CVs have no effect. For other values, the velocity is weighted (multiplied) by CV value/128. If in CV66 (95) we write 128 also does not change the speed. For values below 128, the actual speed will decrease, for higher values it will increase.

In order to achieve a realistic behavior of the railway models, we can determine the acceleration and deceleration rate. In CV3 we can change the acceleration and from CV4 the engine deceleration. If we want to have different acceleration or deceleration depending on the direction of travel, we have CV148-149 for the backward travel direction. If they have the default value zero, for both directions, the

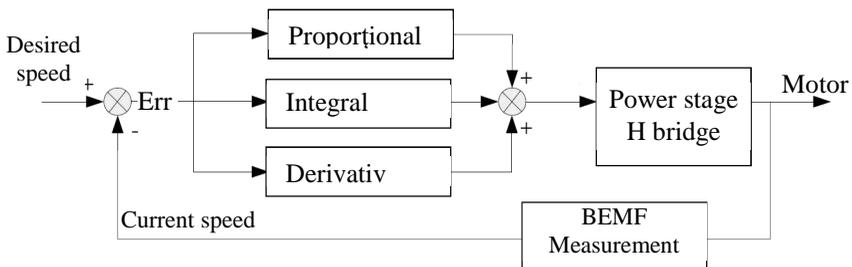
CV3-4 acceleration/deceleration values will be used. For example, if in CV148 we enter a value other than zero it will be the acceleration in the reverse direction and CV3 will be the acceleration in the forward direction.

11. Engine Control

The decoders of the Lokommander family have implemented a PID motor control loop, which uses the generated Electromotive Force (BEMF). This is commonly known as "load compensation" and can be enabled or disabled from Bit 0 of CV60 (factory value bit0 = 1, the PID controller is active).

The motor is connected to one of the diagonals of a H-bridge (made up of 4 FET transistors), the feed is through the other diagonal. The transistors command is provided by the microcontroller in the decoder, using fixed frequency pulse width modulation (PWM) with variable duty cycle. PWM signal frequency is 16/32 kHz, and can be set in Bit 7 of the CV60. The factory value is bit7 = 0, corresponding to the 32kHz frequency. The motor is controlled with PWM pulses regardless of whether the PID controller is activated or not.

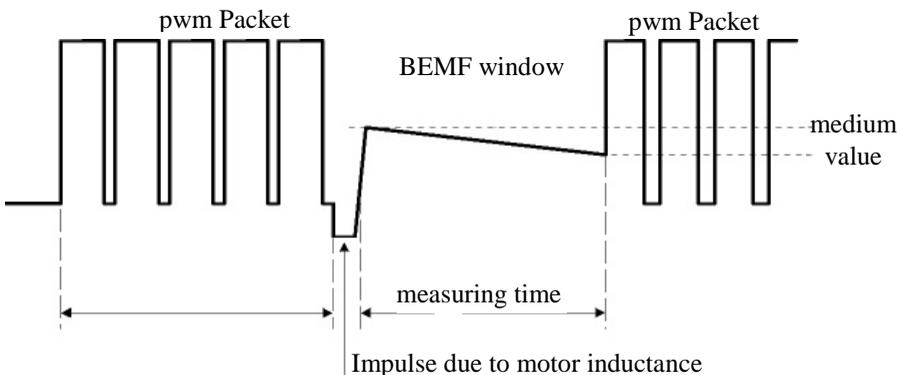
The PID controller is implemented according to the block diagram below:



The reference signal (Desired Speed) is permanently compared to the current speed, and the resulting error signal (Err) is processed by the PID controller, acting on the engine control stage and modifying the PWM signal fill factor so that the error (the difference between desired and current speed) to be minimal.

To determine the current speed, the motor power is interrupted for short periods of time (he is now acting as a generator), and the BEMF voltage is being measured. This voltage is directly proportional to the speed of the motor and is compared to the desired speed for obtaining the error signal.

The engine power interruption period is called a BEMF window. Using too often a BEMF window also has drawbacks, the engine will lose power. So from this point of view, it is desirable to measure not too often, and for a short time as possible the BEMF of the engine. But engine construction requires some time for the BEMF window, which we can not minimize to much.



When power is interrupted, due to the inductance of the motor, there is a pulse at its terminals that compromises the measurement of the bemf voltage. As a result, the bemf measurement will be done after a bemf delay. The width of this impulse (implicitly the required waiting time) depends on the engine construction. Performant motors



(5 or more poles) have this relatively small pulse width compared to older generation motors (3 poles).

During the BEMF measurement, the motor is not supplied with electric power, and due to the mechanical load (gears, locomotive mass, locomotive wagons, etc.) will lose speed, as we can see the downward slope in the previous illustration. In order to obtain a correct BEMF (or current speed) value, the measurements have to be performed several times and mediated.

Also, the number of PWM packets after which a BEMF window is inserted can be variable.

Factory settings ensure proper operation in most applications, but for optimum operation in a given locomotive, we recommend performing the settings described below.

The engine control algorithm in Generation II decoders can be selected from CV9. The default value is 3, with this value the decoder works optimally with most locomotive models, ensuring smooth running without leaps for all speed steps (we recommend using 128 speed steps for optimal BEMF / load compensation). For standard values (CV9 = 0 to 8) corresponds a set of internal control parameters which, in standard mode, are not accessible to the user. Values of 0,1,2 are recommended for low inertia locomotives (Faulhaber motors, small locomotives, etc.). Values 6,7 and 8 can be used in locomotives with high inertia (heavy locomotives, large engines). The values of 3,4 and 5 are used for generic engines.

By selecting a standard set, the user can access only the coefficients of the PID controller (CV61,62,63) and a new set of parameters introduced in the second generation: load compensation weight coefficients (CV137,138,139,140). Practically, these weighting coefficients can determine how strong the load compensation is, depending on the speed of the locomotive. The characteristic of the load compensation is determined by two segments with negative slope, the first between V_{min} (CV2) and

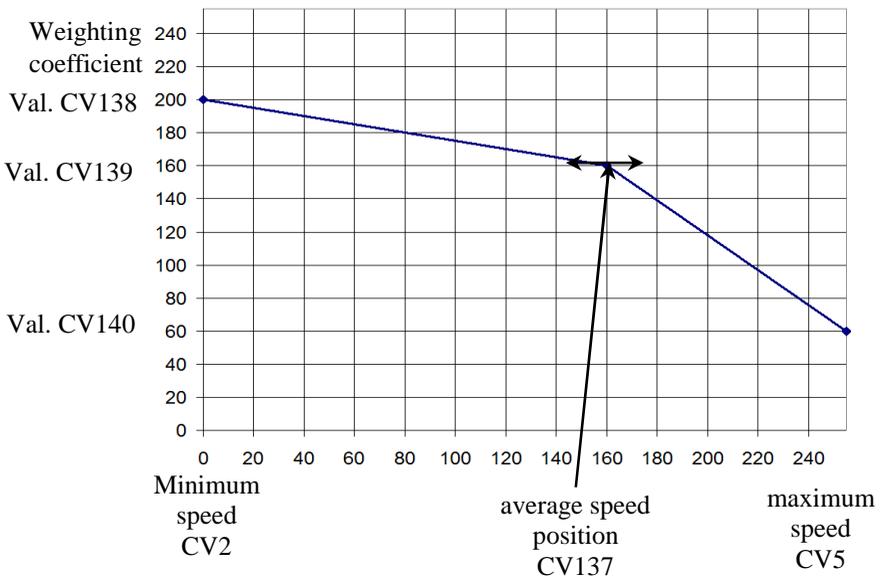


Vmed (CV137), the second between Vmed (CV137) and Vmax (CV5), as seen in the following image.

CV138 sets the load compensation coefficient at the minimum speed (defined in CV2), and CV140 at the maximum speed (defined in CV5). At the average speed of CV137 (which is different from the average speed in CV6) the weighting is set by CV139. The maximum weight is obtained at the value of 255 of CVs 138/139/140.

Practically at high speeds, load compensation no longer counts as high as at low speeds (and extremely low), so the CV140 can drop without causing trouble running the engines.

Experimenting with the CV138 load compensation coefficient values and changing the average speed (CV137) and the weighting coefficient CV 139 good results can be obtained even for problem engines, using the standard control algorithms (eg 3, CV9 = 3) without altering PID parameters (CV61,62,63).





If you want to manually access the parameter set of the control algorithm, it will set CV9 = 9. In this way we will have access to the following CVs:

- CV128: the number of PWM packets after which a BEMF window is inserted. The factory value is 1, its increase is justified only for larger, high inertia motors. The range of numeric values is limited to max 4. The duration of a PWM packet is about 8ms.
- CV130: the BEMF measurement delay. It aims to delay BEMF measurement after the pulse generated by the inductance of the motor after disconnecting from power supply. The factory value is 6. A too low value will have a disastrous result on the BEMF measurement, it will be "drowned" by the motor pulse. In the case of good quality multi-pole motors (such as the Faulhaber or Maxon), the rotor inductance is small, this delay may be reduced. Poor quality motors (such as the 3-pole Piko, Hobby category locomotives) require a longer delay to make BEMF measurements on a stable / clean portion of the generated voltage.
- CV129: The number of mediations during BEMF measuring. The factory value is 6. For better quality motors with multiple poles, the number of mediations may decrease. Increasing the value of over 10 mediations is not practical.
- CV64: PID error limit, ensures the limitation of integral term in PID loop without reducing its response time. Values are allowed in the 1-10 range. A too low value leads to loss of engine power and jerky driving, but too much can lead to instability of the PID loop and excessive engine noise.

Below we present the table with the values of the parameter set according to the CV9 values. In the 9, row of the table, are noted the numeric ranges accepted when using mode 9. Setting the values outside the range will not cause problems, the decoder limiting the values entered in the CVs only at the specified range.



| CV9 | CV128 PWM packets number | CV129 BEMF mediations number | CV130 BEMF measurement delay | observations |
|-----|--------------------------------|------------------------------------|------------------------------------|--------------|
| 0 | 1 | 4 | 1 | |
| 1 | 1 | 4 | 2 | |
| 2 | 1 | 6 | 2 | |
| 3 | 1 | 6 | 2 | Default |
| 4 | 2 | 4 | 1 | |
| 5 | 2 | 4 | 2 | |
| 6 | 2 | 6 | 2 | |
| 7 | 2 | 6 | 4 | |
| 8 | 2 | 8 | 6 | |
| 9 | 1-4 | 1-10 | 1-12 | |

Table 2.

12. Controlled stops

12.1. Constant brake distance (CBD)

Constant Braking Distance stopping allows the locomotive to stop when a command is received on a fixed distance regardless of the travel speed. Stopping can be triggered by 3 factors:

- entering in a sector with asymmetric DCC signal (ABC)- see chap. 12.2.
- entering in a DC sector - see chap. **Error! Reference source not found.**
- receiving a zero speed command

Stopping with CBD when receiving a zero speed command is activated from CV27 Bit7 = 1.

There are two ways to stop on controlled distance:



12.1.1. Fixed deceleration stop

After receiving the stop command, the locomotive traverses a distance calculated at the current speed, then stops with the deceleration set in CV64 (CV150). Runtime with initial speed can be supplemented with a variable delay set in CV65 (CV151) by the formula $\text{Delay} = \text{CV65} * 8\text{ms}$.

12.1.2. Variable deceleration stop

After receiving the stop command, the locomotive will stop with the calculated deceleration based on the speed at the time of receiving the stop command and the stopping distance set by CV153 (CV161). This is a relative distance, being the multiple of the minimum braking distance from the maximum speed obtained with deceleration = 1

If the CV153 is zero (initial value), the fixed deceleration stop from CV64 is selected. If CV64 is also zero, constant distance braking will be disabled. If both CVs are different from zero, priority is the variable deceleration stop set in CV153 (CV161).

All stopping parameters can be differentiated according to the direction of travel. Thus, there are two sets of CVs, one for each direction. If the CV for reverse direction, in brackets, is zero, the value of forward CV will be used for both directions.

Controlled braking distance stopping is inhibited by "Shunting" (F3) or CBD-OFF (F5).

12.2. Detecting asymmetric DCC signal (Lenz ABC)

The asymmetric DCC signal allows exact stopping in front of the signals or in the stations and then passing in the opposite direction. By means of the BM1 or BM2 modules which supply the brake section in front of the signal, the locomotive decoder receives information about the state of the signal according to the direction of travel. Two different information can be transmitted: "Stop" or "Slow Approach".

Upon receiving the "Stop" command, the locomotive will initiate the controlled distance stopping procedure (Chapter 12.1), or if it is disabled, the locomotive will stop with the CV4(CV149) deceleration. Upon receiving the "Slow Approach" command, the speed will be reduced to the value set in CV143 (CV163).

ABC activation is made from CV27:

- Bit0 = 1: Allows ABC signal detection when the right track is more positive
- Bit1 = 1: Allows ABC signal detection when the left track is more positive

ABC typically only works in one direction, but activation for both directions is permitted (except Push-pull).

The sensitivity of the ABC voltage difference detection between the two rails can be changed from CV141. If the initial value does not provide good results at ABC detection, the optimal value can be established experimental in the range 8-16. A too low value causes undesired erroneous detection, and too high will make the detection cumbersome or even impossible.

12.3. Penduling function (Push-pull)

"Push-pull" feature allows you to cycle on a route between two terminal stations repeatedly. Stopping and changing the direction of travel is done when receiving ABC commands at terminal stations. DCC commands only determine the travel speed and possibly the active functions. You can choose between two variants:

12.3.1. Without intermediate stops

The "push-pull" function without intermediate stops requires two separate sections at the ends of the route that generate an ABC "Stop" signal corresponding to the direction the locomotive approaches (the more positive right track). The locomotive arriving in the terminal section stops, inverts the direction (including

directional lights) and, after the waiting time, starts in the new direction. Activation is made from Bit4 (CV122) = 1. From CV142 can be changed the waiting time, in steps of 1 second. On the way may coexist ABC "Slow Approach" sectors , where the locomotive will slow down.

12.3.2. With intermediate stops

"Push-pull" function with intermediate stops requires two separate sections at the ends of the route that generate an ABC "Slow approach" signal corresponding to the direction the locomotive approaches. In the intermediate sectors where the stop is desired, the ABC "Stop" signal will be activated corresponding to the direction from which the locomotive approaches. Intermediate stop last until the ABC "Stop" signal disappear. Activation is made from Bit5 (CV122) = 1. From CV142 you can change the waiting time (in terminal stations), expressed in seconds.

For Push-pull operation, ABC signal detection must be activated in CV27 for one direction (see Chapter 13.2).



ABC activation is not allowed for both directions, this will lead to erroneous operation of "Push-pull" mode! Simultaneous Bit4, Bit5 (CV122) activation is not allowed!

It is recommended to activate one of the constant braking distance methods to ensure that the locomotive stops every time in the same place, regardless of travel speed.

13. Function outputs

Function outputs can command different consumers such as LEDs, bulbs, smoke generator, electromagnetic couplers, etc. Lokommander II decoders have 2 kinds of outputs: power or logic. Power outputs have a transistor that connects to the ground (-) the output at the time of activation. Thus, consumers connect between output and +Vcc (common). Logic outputs provide a voltage of about



+ 5V when active; otherwise it is connected to the ground. The logic outputs cannot exceed the maximum current of 5mA, otherwise there is a risk of destruction of the decoder. A logic output can be used to command 1-2 LEDs with current limiting resistors, or via an external transistor to command larger loads.

To supplement the number of outputs, the SUSI interface can be disabled (CV122 Bit0 = 0) and the corresponding pins can be used as 2 logical outputs. By factory default, they are configured as logical outputs. To use them for the SUSI interface, bit 0 and 1 of CV122 must be set to 1.

Some decoders have a greater number of outputs than those available through the connector. These need to solder additional wires on the marked pads in the drawings from chapter 6.

In table 3 we highlighted the number and type of outputs available on different types of decoders.

| | FL | RL | AUX1 | AUX2 | AUX3 | AUX4 | AUX5 | AUX6 | AUX7 | AUX8 | AUX9 | AUX10 |
|--------|----|----|------|------|-------|-------|-------|-------|------|------|------|-------|
| MICRO | P | P | O,P | O,P | O,P | O,P | O,L,S | O,L,S | | | | |
| NEXT18 | P | P | P | P | L,S | L,S | | | | | | |
| MTC21 | P | P | P | P | L | L | L,S | L,S | | | | |
| Plux12 | P | P | P | P | O,L,S | O,L,S | | | | | | |
| Plux16 | P | P | P | P | L,S | L,S | | | | | | |
| Plux22 | P | P | P | P | P | P | P | P | P | P | L,S | L,S |

P – power output

L – logical output

O – Optional output, accessible by soldering an additional wire

S – output shared with SUSI

Table 3.

For decoders with a maximum of 8 outputs, we use a simplified mapping slightly different from the NMRA standard, which offers greater flexibility (any function can control any output).



| | CV nr. | Default value | AUX 6 | AUX 5 | AUX 4 | AUX 3 | AUX 2 | AUX 1 | FR | FL |
|-----|--------|---------------|------------|-----------|-----------|-----------|----------|----------|----------|----------|
| F0f | 33 | 1 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F0r | 34 | 2 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F1f | 35 | 1 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F1r | 47 | 1 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F2 | 36 | 2 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F3 | 37 | 4 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F4 | 38 | 8 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F5 | 39 | 16 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F6 | 40 | 32 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F7 | 41 | 64 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F8 | 42 | 128 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F9 | 43 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F10 | 44 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F11 | 45 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F12 | 46 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |

Table 4.

For PLUX22 decoders with 10 outputs, we used an NMRA standard mapping.

| | CV nr. | Default value | AUX 10 | AUX 9 | AUX 8 | AUX 7 | AUX 6 | AUX 5 | AUX 4 | AUX 3 | AUX 2 | AUX 1 | FR | FL |
|-----|--------|---------------|--------|------------|-----------|-----------|------------|-----------|-----------|-----------|----------|----------|----------|----------|
| F0f | 33 | 1 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F0r | 34 | 2 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F1f | 35 | 4 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F2 | 36 | 8 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F3 | 37 | 16 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F4 | 38 | 4 | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | |
| F5 | 39 | 8 | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | |
| F6 | 40 | 16 | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | |
| F7 | 41 | 32 | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | |
| F8 | 42 | 64 | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | |
| F9 | 43 | 16 | | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | | | |
| F10 | 44 | 32 | | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | | | |
| F11 | 45 | 64 | | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | | | |
| F12 | 46 | 126 | | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) | | | | | |
| F1r | 47 | 4 | | | | | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |

Table 5.



Mapping in grays field are impossible. Settings are only possible within the white field (the factory settings are colored green).

The functions F0 (f = forward, r = reverse), F1 (f = forward, r = reverse) F2 and F3 can control only FL, FR and AUX1,2,3,4,5,6 outputs. For NMRA compatibility, CV35 is F1 for forward (forward) and CV47 for reverse (reverse) direction. Functions F4,5,6,7,8 can only control AUX2,3,4,5,6,7,8 and 9 outputs. Functions F9,10,11,12 can only control AUX outputs 5,6,7,8,9 and 10.

The PWM factor of the 12 outputs is set in CV48-59 (see Chapter 26).

From CV112, we can set the start-up time (Fade-IN), respectively CV113, the fall-off time (Fade-OUT) of the PWM signal applied to outputs. These times can be set in steps of 8ms and represent the time in which the output PWM fill factor rise from 0 to 255, or vice versa. If CV48-59 establishes a fill factor lower than the maximum value 255, the rise and fall times decrease proportionally. These two parameters are common to all outputs. This function is useful when we want to simulate the slow turn on of incandescent bulbs.

If we want any output to be commanded with a continuous signal (without variable fill factor PWM) in CV117(CV185) we can set to value 1 the bit corresponding to the desired output(s). On Lokommander II versions with more than 8 outputs, the continuous command of outputs 9-12 can be set from CV185 bits 0-3.

Starting with software version 3.5.207, functions F0 (f / r), F1 (f / r) and F2-F12 can be configured to inhibit one or more output(s) FL, FR, AUX1, ... AUX 6.



| | CV nr. | Default value | AUX 6 | AUX 5 | AUX 4 | AUX 3 | AUX 2 | AUX 1 | FR | FL |
|-----|--------|---------------|------------|-----------|-----------|-----------|----------|----------|----------|----------|
| F0f | 166 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F0r | 167 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F1f | 168 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F2 | 169 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F3 | 170 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F4 | 171 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F5 | 172 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F6 | 173 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F7 | 174 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F8 | 175 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F9 | 176 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F10 | 177 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F11 | 178 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F12 | 179 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |
| F1r | 180 | 0 | Bit7 (128) | Bit6 (64) | Bit5 (32) | Bit4 (16) | Bit3 (8) | Bit2 (4) | Bit1 (2) | Bit0 (1) |

Table 6.

According to *table 6*, if we want that a function to inhibit one of the outputs, the corresponding output bit must be set to 1 in the CV corresponding to the function. Functions F0 and F1 can inhibit FL, FR, AUX1, ... AUX6 outputs depending on travel direction. CVs 166/168 set the inhibition of some outputs if the locomotive moves in the forward direction, respectively in CVs 167/180 sets the inhibition of some outputs if the locomotive moves in the reverse direction.



14. Analog operation (DC)

The decoder allows the locomotive to run even with classical speed controllers providing continuous power (DC). They can be of two types: filtered and pulsed (PWM).

To enable DC operation, it is necessary to enter the value "1" in Bit2 / CV29.

From CV13 and CV14 we can determine which function is to be activated if we use the decoder in analog mode(DC current). In the following table we find the meaning of each bit of the two CVs. If the bit has a value of 1, that function will be active in analog mode.

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|------|------|------|------|------|------|------|------|------|
| CV13 | F8 | F7 | F6 | F5 | F4 | F3 | F2 | F1 |
| CV14 | F14 | F13 | F12 | F11 | F10 | F9 | RL | FL |

Table 7.

There are two modes of analog (DC) operation:

14.1. Analog mode 1

Mode 1 can be used with variators that provide filtered continuous voltage. Depending on the voltage in the rails, the desired speed is set and motor control via the PID loop is provided. You can get smooth run even and at very low speeds, as in DCC mode. For example, at the sudden application of the maximum voltage, the engine will reach the maximum speed with the acceleration rate in CV3 (CV148).

The correlation between rail voltage and travel speed is linear in function of 3 CVs:

- CV145: starting threshold, the motor starts when the voltage in the rails reaches this value



- CV146: turn-off threshold, the motor stops when the rail voltage drops below this value, may be lower than the starting threshold.
- CV147: maximum speed, at this rail voltage will reach the maximum speed

The value written in these CVs is calculated by multiplying the value of the desired voltage by 10. For example, for the maximum voltage of 14V, in CV146 will be written 140.

This analog mode will not work properly with pulsed(PWM) speed variators!

To choose analog mode 1, the zero value is written in CV164.

14.2. Analog mode 2

In this mode, the motor is controlled by a high frequency pulsed (PWM) voltage. The PWM duty cycle is fixed and set by CV164. For the maximum value of 255, virtually all rail voltage is applied to the engine. If a lower value is set, the voltage applied to the motor will be less than that in the rails (allows the use of motors with lower rated voltage). To choose mode 2, a value other than zero is entered in CV164. Only this mode can be used with pulsed (PWM) speed controllers.

14.3. Controlled stop on DC sector

Continuous current can also be used in conjunction with DCC to supply DC brake sections. Thus, if a DCC-powered locomotive reaches a DC sector, it will stop if the following conditions are met: Bit4 or Bit5 in CV27 have the value "1", CV29 Bit2 = 0 and the voltage in the rails is higher than the threshold set in CV162.

The threshold set in CV162 (default value 100 => 10V) is useful when using a power pack simultaneously with the DC-brake function. So if the supply voltage is below the threshold we are in SPP mode and the locomotive will stop after the expiration of the time set in CV123. If the voltage exceeds the threshold, the DC brake function

is activated and the locomotive will stop at a controlled distance (see chapter 12.1)

15. Bidirectional communication (RailCom)

"Bidirectional" means that the transfer of information under the DCC protocol is not only to the decoder but also in the opposite direction. Thus the decoder can send messages such as confirmation of receipt of commands, address, actual speed, internal temperature, load and other status information.

The RailCom operating principle is based on the introduction of a cutout by the control station at the end of each DCC package where it interrupts the power supply and short-circuits the two lines. In these windows the decoders send a few bytes of data that are received by detector connected between locomotive and control station or by control station itself (if its capable to receive railcom informations).

The data packet is divided into two channels. On the first channel, the address (short, long, or consist) of the decoder is transmitted. On the second channel, CV handling POM responses are delivered (reading, writing result).

RailCom communication can be deactivated from CV29-Bit3 (0 - RailCom inactive, 1 - RailCom active). Channels 1 and 2 are enabled in CV28 Bit1 and Bit2.

16. Special functions

By calling our special functions we can get information about:

- The internal temperature of the decoder
- the quality of the received DCC signal
- number of hours and minutes of operation
- the time stamp (hour) at which the last locomotive maintenance was performed

In order to save the values of these parameters to the non-volatile memory (eeprom) of the decoder, must be enabled in the user-accessible CV area. Bit7 / CV122 enables or disables the save function (bit7 = 0, save function disabled, bit7 = 1, save function enabled). Saving the instantaneous values is done by calling the F5 function from the control station (or tOm Programmer).



Without calling F5 (On, then Off), the values in the corresponding CVs are not updated!

The internal (saved) temperature of the decoder can be read from CV133. The temperature is given in degrees Celsius.

The DCC Signal Quality Indicator (QoS = Quality of Signal) is read from CV135. The read value is given in percent (in the range 0-100%). The lowest QoS value detected by the decoder from the last reading, it's writed in CV136. To reset the minimum value, enter CV136 value 100 [%]. (before reading, call the save function via F5 On, F5 Off).

The number of hours and minutes of operation are read from CV156, 157 and 158 thus:

- The number of minutes of operation is the value read from CV156
- The number of hours of operation is the sum of the value read from CV157 multiplied with 256 and the value read from CV158. (before reading, call the save function with F5 On, F5 Off).

Maintenance period:

The decoder may retain the time stamp of the locomotive maintenance and may indicate the exceedance of a set number of hours since the last maintenance.

This function can be activated and configured in CV154 (see Chapter 26). The maintenance interval is specified in hours in CV155. The factory value is 40 hours. The value can be changed by the user in the range 0-255. After resetting the decoder, the value of CV155 will be 40 (hours).

The time at which the last maintenance was confirmed can be read from CV159 and 160 thus:

$$\text{Hours} = (\text{CV159 Value}) + 256 * (\text{CV160 Value})$$

To confirm the maintenance, the so-called pseudo programming is used: the value 128 is entered in CV8 (it is not equivalent to a decoder reset!). As a result of this operation, the maintenance time mark is saved and the new maintenance interval will be calculated from this time stamp.



If the exceeded maintenance interval has been signaled by setting CV30 bit 3, after confirmation of the maintenance CV30 must be reset (to 0). The CV30 is not automatically erased by the maintenance confirmation procedure.

17. Electric Uncoupler Configuration

The Lokommander II decoder allows the use of any physical output for the action of electromagnetic couplers. If a logical output is chosen it is necessary to use an external transistor, the output supplying an insufficient current for actuating the coupler. The Krois® and Roco® couplings require a high-frequency PWM signal supply to avoid burning of the coils of the couplings. The automatic decoupling function of the decoder provides this command signal.



The automatic decoupling function can only be activated with the stationary locomotive.

The automatic decoupling function is a physical function (not logical, such as maneuvering speed, inactivation of acceleration and deceleration, etc.), and for its configuration it is done the following way:

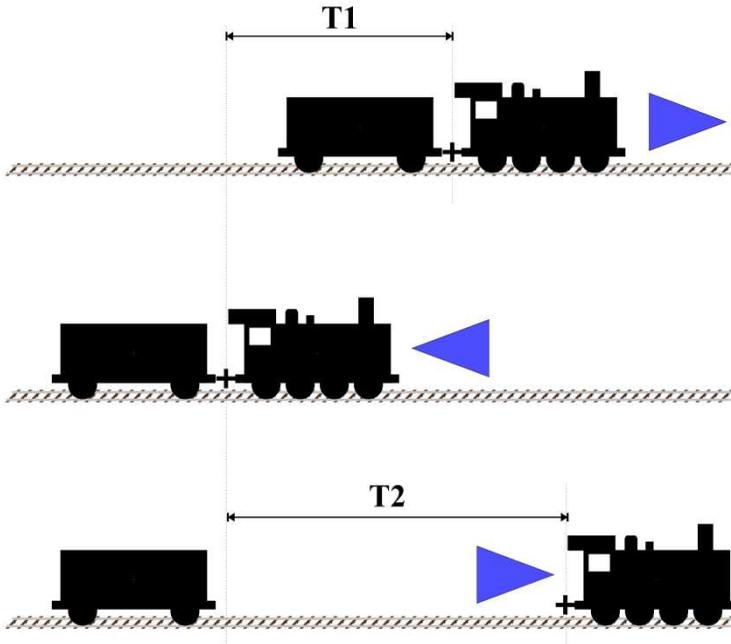
Choose an F function to be used for the automatic decoupling function (it can be a function used for other commands, for example sound).



For the selected function, from CVs 33-47, we make the assignment (mapping) of the physical output, to which the electric coupler is connected, to this function (for example, if we choose the F8 function for automatic decoupling and the coupler electromagnets are connected to the physical output Aux2 / purple wire, in CV43 will enter the value 8, which means that when the F8 function is activated, the physical output Aux2 will be turned ON).

For electromagnetic decoupling, it must be specified in CV118 on which outputs are applied the decoupling function. As for Aux2, we will write in CV118 the value 4 (in CV118 specify the output number: for FL value 1, for RL value 2, for Aux1 value 3, for Aux2 value 4 ... for Aux8 value 10).

From CV124 we can select the type of coupling used (DC or PWM) as follows: Bit0-0 output HF PWM; Bit0-1 continuous output. From Bit1-CV124 we can choose the engine control mode during uncoupling. Thus bit1-0 the engine will be commanded instantly, making a sudden movement; bit1-1 the motor will be controlled by the PID loop with deceleration acceleration according to CV3-4.



Once we mapped logic F to physical output, we can change configuration variables to optimize automatic decoupling. The stationary locomotive (after stopping) will have the direction set according to the figure above. The maximum travel speed during the automatic decoupling function is set in CV121 (value 0 means that the displacement will not be performed, but only the electromagnetic coupling will be activated). When calling the function, the locomotive will actuate the electric coupler and travel for a period of time $T1$ in the opposite direction to the direction set before calling the function. The travel distance can be controlled by the travel speed (CV121) and the backward motion time (CV119). After this movement, the locomotive stops, changes the direction of travel (which will be the same as before calling the function) and will travel

for a duration of T2, after which it will stop moving and disable the electric couplings. To determine the distance of travel in the direct sense, we also have 2 parameters, the travel speed (CV121) and the time period T2 (CV120). From these 2 parameters we can reduce or increase the distance traveled directly. Functions activated before dialing the decoupling function remain active during decoupling.

The decoupling function is called when the function is activated(ON) and after a complete cycle is turned off, even if the F function has not been deactivated. To turn the function on again, the OFF command will be sent, followed by ON command.



The polarity of the electromagnetic couple's wires is important. If they are not properly connected, the movement (lift) is reversed!

18. SUSI / Locowire interface

You can connect to the SUSI / Lokowire interface any sound or function decoder that meets the interface specifications. For connection, 4 contacts are provided on the top of the decoder (see figures from Chap.6). For versions with Plux16, Plux22, MTC21 and NEXT18 connectors, these connection points are available among the pins of the connector, and no additional wires are needed. These contacts respect the order / meaning of the SUSI and Lokowire interface respectively. We recommend the use of specific color conductors



Attention! Improper connection of the SUSI / Lokowire module may cause damage to the SUSI / Lokowire module

18.1. Programing SUSI modules

Like locomotive decoders, SUSI sound modules can be personalized by changing some operating parameters. The values of these parameters are stored in configuration variables (CVs) ranging from CV897 to CV1024. The SUSI sound module is programmed via



the Lokommander II decoder. Depending on the CV number, the Lokommander II decoder will identify whether this CV should be written or read from a SUSI module connected to the decoder interface. To program the various configuration variables of the SUSI module, please refer to its manual.

Write CVs of SUSI modules can be done in PT or PoM mode. Because some digital systems allow writing and reading of CVs only in the 1-255 range, a special mechanism for these digital systems has been implemented in the Lokommander II decoder, with the help of which two CVs provide access to the CVs of the SUSI modules. CV126 is used as index, and CV127 is used as transport CV. So in CV126 we write the difference between the address of the CV we want to access and 800. By reading or writing CV127 we read or write the CV with the address $800 + CV126$.

Examples:

- If you want to write value 1 in CV897 of the SUSI module, you have to write 97 ($897-800 = 97$) in CV126 and value 1 in CV127. After entering value 1 in CV127, the Lokommander II decoder will transmit a command on the SUSI interface to the sound module (or the function decoder) to write the value 1 in CV 897.
- If you want to read the content of CV 902 from the SUSI module connected to the Lokommander II decoder interface, enter the value 102 ($902-800 = 102$) in CV126, and read the CV127 value. This value is equal to the value contained in CVC 902 of the sound module (or function decoder) connected to the Lokommander II decoder.

The Lokowire interface does not require programming of the configuration variables. The Lokommander II decoder is factory-shipped with SUSI configured interface (CV122 - bit1 = 1). To activate the Lokowire interface, set CV122 - bit1 = 0.

19. Using external capacitors or a power pack

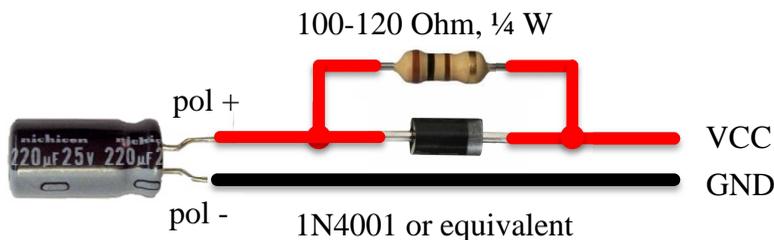
In some model layouts, due to the wear of rails and deposits of different materials on the tracks, the contact between rails and wheels are imperfect. They cause power outages, with jerky running, especially at low speed. These inconveniences can be eliminated using buffer capacitors (220 uF / 25V or for better results higher but not exceeding 2200 uF) or uninterruptible power supplies SPP.

To connect these devices, the Lokommander II decoder has 3 contacts on one side of the PCB. The position of the 3 contacts to which the wires are attached can be determined from the images of the different types of decoders in Cap. 6.



Installing these devices requires quality soldering equipment and experience. Our warranty does not cover defects due to inappropriate interventions and soldering

The capacitors are charged in series thru a 100 ohm resistors, limiting their charging current, therefore the digital control systems will not interpret the capacitor charging current as short-circuit. The diode is designed to provide the required power or the maximum current available to the internal circuits of the decoder in the absence of tension on the rails. The diode and resistor are external components, not included in the Lokommander II decoder. The connections should to be done according to the images from Chapter 6. The black wire will be soldered to the GND and the red one to the Vcc (the two extremes from the group of 3 pads reserved to SPP). After making the connections, we can use a heat shrinkable tube or insulating tape for insulation.



Disconnect / remove buffer capacitors before programming the decoders. The use of buffer capacitors does not facilitate the programming of decoders CVs

Uninterruptible power supplies SPP (Smart Power Pack or equivalent) removes this inconvenience, allowing both write and read of CVs in the traditional way without having to remove them. Switching off the SPP source during programming is done automatically by the Lokommander II through the third wire (Charge).

To connect the SPP modules, use the 3 contacts located on one of the Lokommander II sides, as can be seen in the pictures in chapter 6. Details of the connection can be found in the manual of the SPP.

SPP sources work only in digital mode, in analog mode they are disabled (see CV29 configuration). In order to avoid very high consumption, due to the simultaneous loading of non-interruptible sources, when powering model layout with multiple SPPs, there is a start delay. Thus, in CV152 we can set in seconds the time after which the SPP module is started from the moment of the power is applied in the track. When multiple decoders are used at the same track, this time will be set different in order to avoid the simultaneous start of all SPPs. The SPP modules allow locomotives to run for up to 4 seconds without DCC power from rails (fully loaded, depending on locomotive consumption). This duration is set in CV123, in steps of 16ms (default value 16, CV_value * 16ms =

0.25 seconds). After this period expired in the absence of the DCC signal, even if the SPP its not fully discharged, the locomotive will perform an emergency stop (as a safety measure). The movement will be resumed only after witch the DCC signal reappears.



Note that when powering the locomotiveshiped with SPP, charging the capacitors from the uninterruptible power source can consume a current of about 300 mA for a maximum of 2 minutes. For more details, please refer to the SPP uninterruptible power supply manual.

20. Resetting the decoder

You can reset the decoder to the default settings at any time. By using any DCC command station, it is enough to enter any numerical value (other than 128) in CV8, as a result of this reset, all CVs will have the default value (see the Default Value column in the CV table). The decoders can be reset using the tOm Programmer; for the same result, press "Reset CVs" button in the Firmware TAB.

There are 2 CVs that are an exception, their content is not deleted in case of a reset. These are CV105 and CV106, intended to store user-specific information (serial number, identifier, inventory number, etc.). Their content will be changed by direct writing, resetting the decoder will not alter the content of these CVs.



A firmware upgrade will enter the default values in CV105 and CV106. In order to preserve their value, make a backup of the CVs before firmware upgrad (using the tOm Programmer)



By resetting the Lokommander II decoder, the CVs of the connected SUSI modules will not be reset.

21. Secondary address (decoder lock)

When using multiple decoders within the same housing, it is useful to use a secondary address that will allow the selection of the decoder in question. In this way any of the decoders that are inside the same housing (carriage body) can be programmed on the Programming Track without removing it. The secondary addresses are programmed into CV16 before the decoders (in our case the Shine Maxi Digi 02 with the onboard decoder) is being assembled in their housing.

The ranges of secondary addresses are 1-7 (value of 0 means that secondary addressing is not used). This permits the use of maximum 7 decoders in the same carriage or locomotive housing, which is more than enough.

If the value of CV16 is not equal to zero, the decoders will accept programming commands only if the secondary address of decoder that is intended to be programmed is programmed prior in CV15, and it matches the value in CV16 (it should be the same as CV16 of the decoder in question).



WARNING: even CV16 can be programmed only if the correct value is programmed in CV15.

Using secondary addressing is important to know that the only CV that can read and written without knowing the secondary address is CV15. For this reason the values used are limited to the range 1-7. If the secondary address of the decoder is forgotten within 7 iterations it can be found.

This way of accessing / programming of the decoder CVs is useful in case of railcars, or permanently connected sets, which have more decoders built in, and it would be very inconvenient their programming in the traditional way (on Programming Track all decoders would be programmed with the same CV values, what most likely is not desired).



Assigning secondary addresses to each decoder of the railcar or carriage sets, when placing them on the Programming Track, only the decoder for which the CV15 = CV16 will be programmed. In this way we can program several decoders independently, even if they are on the programming track in same time.

22. Firmware update

You can update the Lokommander II decoder operating software (called firmware) at any time. New firmware versions are designed either to eliminate bugs in the operation of decoders or because new functions are implemented. This update can be made by you without removing the decoder from the locomotive via the tOm Programmer. The tOm Programmer's operating software and firmware upgrade files can be downloaded from the train-O-matic site. For the operation of the upgrade, please refer to the tOm Programmer's user manual.

The firmware version can be found by reading the following CVs:

- CV253 firmware version (3)
- CV254 firmware subversion (5)
- CV254 build version, upper byte (0)
- CV256 build version, lower byte (200)

For users of decoders with older firmware versions, some functions in this manual may be inaccessible.

23. Special firmware version for 3V engine

This special firmware version of Lokomander II it's useful for railroad models which, for various reasons (gauge, physical dimensions), do not allow the use of regular motors, only of low-voltage motors.

In order to ensure the command of the 3V engine, modifications were made to the Lokomander II firmware and the use of additional



extra hardware. By reading Chapter 12 (in the Lokomande II user manual) we can understand the principle of engine control. It is noticeable that the electromotive voltage (BEMF) measurement is made at regular intervals. For the 3V engine this voltage is much lower. To overcome this inconvenience we used an external amplifier. Reading the amplified voltage is done on one of the logic outputs (AUX6) of the decoder, which in software is configured for analog input. Also, the command of the BEMF voltage divider is made with another logic output (AUX5).

The 3-volt motor is also powered by PWM pulses of 12-16 V with variable duty cycle, as regular motors. Firmware limit the fill factor to prevent burning of the 3V engine. To limit the current peaks that can occur on the 3V motor when applying the PWM signal, two resistors are connected in series with the motor.

The AUX2 output is used for the SPP (external capacitor package) command .

To control the lights and auxiliary consumers remain fewer with 3 output. For decoder with NEXT18 connector, FL, RL, AUX1 and AUX3-AUX4 (only if the SUSI interface is not used) outputs remain available for the lights.

This special firmware can be used with Lokomander II of one of the following types: NEXT18, MTC21, PLUX22.

Decoders shipped with this special firmware can only be used for 3V motor control and only with the related hardware circuits. By firmware update you can go back to using the usual 12-16V motors.



24. Accessories

- tOm programmer is a PC interface used to program DCC mobile decoders.
- shine FDT, shine LT, shine micro are LED modules for locomotive and wagon lighting
- shine mini / midi / maxi digi / ana are LED sticks for interior lighting of wagons
- TD Maxi, TD Roco are decoders for macros

For details on accessories and a complete list of railroad products, visit the page: www.train-o-matic.com/

25. Technical support

If you have any questions or suggestions about train-o-matic products, you can write to us at support@train-o-matic.com

Any positive or negative criticism is welcome. We are continually working on firmware optimization by adding new functionality and correcting any bugs that may still exist.



26. The decoder CV table

In the table on the following pages are listed all the CVs of Lokommander II decoders. We recommend that you change your CVs only if you are sure of their function and the impact of your action. Incorrect CV settings can negatively affect the performance of the decoder or cause incorrect responses to the commands transmitted to the decoder. The “CV” column contains the CVs number, the “Default Value” column contains the "factory" value of the CVs (after a decoder reset, all CVs will have the appropriate value in this column), the column "Value Range" contains the range of usable values for each CV and the "Description" column contains the name (if there is an established name) and information about the CV function as well as the reference to the related chapter

| CV | Default Value | Value Range | Description |
|----|---------------|-------------|---|
| 1 | 3 | 0-127 | Decoder Adresse Short, 7 bits |
| 2 | 3 | 1-255 | Vstart |
| 3 | 5 | 0-63 | Acceleration Rate, 0=Fastest acceleration |
| 4 | 5 | 0-63 | Deceleration Rate, 0=Fastest deceleration |
| 5 | 200 | 1-255 | Vhigh |
| 6 | 60 | 0-255 | Vmid =[25%-75%]Vhigh |



| | | | |
|----|-----|-------|---|
| 7 | 105 | - | Software Version (only readable) |
| 8 | 78 | - | Manufactured ID/RESET (readable 78 = train-O-matic, any written value, excepting 128, will reset the decoder to the factory default values) |
| 9 | 3 | 0-9 | Motor Control Algorithm, 0-8 User defined = 9 (see cap.12 and CV's 128-130) |
| 13 | 0 | 0-255 | Analog Mode, Alternate Mode Function Status F1-F8 Bit 0 = 0(0): F1 not active in Analog mode = 1(1): F1 active in Analog mode Bit 1 = 0(0): F2 not active in Analog mode = 1(2): F2 active in Analog mode Bit 2 = 0(0): F3 not active in Analog mode = 1(4): F3 active in Analog mode Bit 3 = 0(0): F4 not active in Analog mode = 1(8): F4 active in Analog mode Bit 4 = 0(0): F5 not active in Analog mode = 1(16): F5 active in Analog mode Bit 5 = 0(0): F6 not active in Analog mode = 1(32): F6 active in Analog mode Bit 6 = 0(0): F7 not active in Analog mode = 1(64) F7 active in Analog mode Bit 7 = 0(0): F8 not active in Analog mode = 1(255): F8 active in Analog mode |



Lokommander II

User Manual firmware 3.5.195

Version

0.1.22

| | | | |
|----|---------------|-------|--|
| 14 | 3= 1+ 2 | 0-255 | Analog Mode, Alternate Mode Function. Status F0f, F0r, F9-F14 Bit 0 = 0(0): F0f not active in Analog mode = 1(1): F0f active in Analog mode Bit 1 = 0(0): F0r not active in Analog mode = 1(2): F0r active in Analog mode Bit 2 = 0(0): F9 not active in Analog mode = 1(4): F9 active in Analog mode Bit 3 = 0(0): F10 not active in Analog mode = 1(8): F10 active in Analog mode Bit 4 = 0(0): F11 not active in Analog mode = 1(16): F11 active in Analog mode Bit 5 = 0(0): F12 not active in Analog mode = 1(32): F12 active in Analog mode Bit 6 = 0(0): F13 not active in Analog mode = 1(64): F13 active in Analog mode Bit 7 = 0(0): F14 not active in Analog mode = 1(255): F14 active in Analog mode |
| 15 | 0 | 0-7 | LockValue: Enter the value to match Lock ID in CV16 to unlock CV programming. No action and no ACK will be performed by the decoder when LockValue is different from LockID. In this situation only CV15 write is allowed. |



| | | | |
|----|-----|---------|--|
| 16 | 0 | 0-7 | LockID: To prevent accidental programming use unique ID number for decoders with same address. |
| 17 | 192 | 192-255 | Extended Address, High Byte |
| 18 | 3 | 0-255 | Extended Address, Low Byte |
| 19 | 0 | 0-127 | Consist Address If CV #19 > 0: Speed and direction is governed by this consist address (not the individual address in CV #1 or #17+18); functions are controlled by either the consist address or individual address, see CVs #21 + 22. |
| 21 | 0 | 0-255 | Functions defined here will be controlled by the consist address. Bit 0 = 0(0): F1 controlled by individual address = 1(1): by consist address Bit 1 = 0(0): F2 controlled by individual address = 1(2): by consist address Bit 2 = 0(0): F3 controlled by individual address = 1(4): by consist address Bit 3 = 0(0): F4 controlled by individual address = 1(8): by consist address Bit 4 = 0(0): F5 controlled by individual address = 1(16): by consist address Bit 5 = 0(0): F6 controlled by individual address = 1(32): by consist address Bit 6 = 0(0): F7 controlled by individual address |



Lokommander II

User Manual firmware 3.5.195

Version

0.1.22

| | | | |
|----|---------------|------|---|
| | | | <p>= 1(64): by consist address</p> <p>Bit 7 = 0(0): F8 controlled by individual address</p> <p>= 1(255): by consist address</p> |
| 22 | 0 | 0-63 | <p>Functions defined here will be controlled by the consist address.</p> <p>Bit 0 = 0(0): F0 (forw.) controlled by individual address</p> <p>= 1(1): by consist address</p> <p>Bit 1 = 0 (0): F0 (rev.) controlled by individual address</p> <p>= 1(2): by consist address</p> <p>Bit 2 = 0(0): F9 controlled by individual address</p> <p>= 1(4): by consist address</p> <p>Bit 3 = 0(0): F10 controlled by individual address</p> <p>= 1(8): by consist address</p> <p>Bit 4 = 0(0): F11 controlled by individual address</p> <p>= 1(16): by consist address</p> <p>Bit 5 = 0(0): F12 controlled by individual address</p> <p>= 1(32): by consist address</p> |
| 27 | 51= 1+ | 0-7 | <p>Decoder Automatic Stopping Configuration</p> <p>Bit0 = 0(0) Disable Auto Stop in the presence of an asymmetrical DCC signal which is more positive on the right rail</p> <p>= 1(1) Enable Auto Stop in the presence of an asymmetrical DCC signal which is more positive on the right rail</p> <p>Bit1 = 0(0) Disable Auto Stop in the presence of an asymmetrical DCC</p> |



| | | | |
|----|---------------|------|--|
| | 2+ | | signal which is more positive on the left rail = 1(2) Enable Auto Stop in the presence of an asymmetrical DCC signal which is more positive on the left rail Bit2 – not used Bit3 – not used Bit4 = 0(0) Disable Auto Stop in the presence of reverse polarity DC = 1(16) Enable Auto Stop in the presence of reverse polarity DC Bit5 = 0(0) Disable Auto Stop in the presence of reverse polarity DC = 1(32) Enable Auto Stop in the presence of reverse polarity DC Bit6 – not used Bit7 = 0(0) Disable Auto Stop in the presence of zero sped brake = 1(128) Enable Auto Stop in the presence of zero sped brake |
| 28 | 3 | 0-3 | RailCom configuration Bit 0 = 0(0): CH1 Address Broadcast Off = 1(1): CH1 Address Broadcast On Bit 1 = 0 (0): CH2 Data transmission Off = 1(2): CH2 Data transmission On |
| 29 | 10= 2+ | 0-63 | Configuration Data Bit 0 = 0(0): Locomotive Direction normal = 1(1): Locomotive Direction reversed Bit 1 = 0(0): FL controlled by bit 4 in Speed and Direction instructions = 1(2): FL controlled by bit 4 in function group one instruction |



| | | | |
|----|---|-------|--|
| | 8 | | Bit 2 = 0(0): Power Source Conversion NMRA Digital Only (only DCC) = 1(4): Power Source Conversion Enabled (DC + DCC) Bit 3 = 0(0): Bi-Directional Communications disabled = 1(8): Bi-Directional Communications enabled. Bit 4 = 0(0): speed table set by configuration variables #2,#5, and #6 = 1(16): speed table set by configuration variables #66-#95 Bit 5 = 0(0): one byte addressing (short addressing) = 1(32): two byte addressing (extended/long addressing) Bit 6 - Not used Bit 7 - Not used |
| 30 | 0 | 0-15 | Error Information: = 0 - No error occurred. = 1 (Bit0) - Motor Short Protection occurred = 2 (Bit1) - Aux Output Short Protection occurred = 4 (Bit2) - Overtemperature occurred = 8 (Bit3) - The maintenance period has been exceeded If error occurred the value must be cleared with programming "0" to CV30 |
| 33 | 1 | 0-255 | F0, Forward move mapping Bit 0 = 0(0): Out1 not active on F0 forward = 1(1): Out1 active on F0 forward Bit 1 = 0(0): Out2 not active on F0 forward = 1(2): Out2 active on F0 forward |



| | | | |
|----|---|-------|--|
| | | | <p>Bit 2 = 0(0): Out3 not active on F0 forward = 1(4): Out3 active on F0 forward Bit 3 = 0(0): Out4 not active on F0 forward = 1(8): Out4 active on F0 forward Bit 4 = 0(0): Out5 not active on F0 forward = 1(1): Out5 active on F0 forward Bit 5 = 0(0): Out6 not active on F0 forward = 1(2): Out6 active on F0 forward Bit 6 = 0(0): Out7 not active on F0 forward = 1(4): Out7 active on F0 forward Bit 7 = 0(0): Out8 not active on F0 forward = 1(8): Out8 active on F0 forward</p> |
| 34 | 2 | 0-255 | <p>F0, Backward move mapping Bit 0 = 0(0): Out1 not active on F0 backward = 1(1): Out1 active on F0 backward Bit 1 = 0(0): Out2 not active on F0 backward = 1(2): Out2 active on F0 backward Bit 2 = 0(0): Out3 not active on F0 backward = 1(4): Out3 active on F0 backward Bit 3 = 0(0): Out4 not active on F0 backward = 1(8): Out4 active on F0 backward Bit 4 = 0(0): Out5 not active on F0 backward</p> |



Lokommander II

User Manual firmware 3.5.195

Version

0.1.22

| | | | | |
|----|-------------------|-------|--|----|
| | | | <p>= 1(1): Out5 active on F0 backward Bit 5 = 0(0): Out6 not active on F0 backward = 1(2): Out6 active on F0 backward Bit 5 = 0(0): Out7 not active on F0 backward = 1(4): Out7 active on F0 backward Bit 6 = 0(0): Out8 not active on F0 backward = 1(8): Out8 active on F0 backward</p> | |
| 35 | 1 4 for PLUX22 | 0-255 | <p>F1, Forward move mapping Bit 0 = 0(0):Out1 not active on F1 f.w. = 1(1):Out1 active on F1 forward Bit 1 = 0(0):Out2 not active on F1 f.w. = 1(2):Out2 active on F1 forward Bit 2 = 0(0):Out3 not active on F1 f.w. = 1(4):Out3 active on F1 forward Bit 3 = 0(0):Out4 not active on F1 f.w. = 1(8):Out4 active on F1 forward Bit 4 = 0(0):Out5 not active on F1 f.w. = 1(1):Out5 active on F1 forward Bit 5 = 0(0):Out6 not active on F1 f.w. = 1(2):Out6 active on F1 forward Bit 6 = 0(0):Out7 not active on F1 f.w. = 1(4):Out7 active on F1 forward</p> | 35 |



| | | | | |
|----|-----------------------|-------|---|----|
| | | | Bit 7 = 0(0):Out8 not active on F1 f.w. = 1(8):Out8 active on F1 forward | |
| 36 | 1 8 for PLUX22 | 0-255 | F1, Backward move mapping Bit 0 = 0(0):Out1 not active on F1 b.w. = 1(1):Out1 active on F1 b.w. Bit 1 = 0(0):Out2 not active on F1 b.w. = 1(2):Out2 active on F1 b.w. Bit 2 = 0(0):Out3 not active on F1 b.w. = 1(4):Out3 active on F1 b.w. Bit 3 = 0(0):Out4 not active on F1 b.w. = 1(8):Out4 active on F1 b.w. Bit 4 = 0(0):Out5 not active on F1 b.w. = 1(1):Out5 active on F1 b.w. Bit 5 = 0(0):Out6 not active on F1 b.w. = 1(2):Out6 active on F1 b.w. Bit 6 = 0(0):Out7 not active on F1 b.w. = 1(4):Out7 active on F1 b.w. Bit 7 = 0(0):Out8 not active on F1 b.w. = 1(8):Out8 active on F1 b.w. | 36 |
| 37 | 2 16 for PLUX22 | 0-255 | F2 mapping Bit 0 = 0(0): Out1 not active on F2 = 1(1): Out1 active on F2 | 37 |



| | | | | |
|----|-------------------|-------|--|----|
| | | | Bit 1 = 0(0): Out2 not active on F2 = 1(2): Out2 active on F2 Bit 2 = 0(0): Out3 not active on F2 = 1(4): Out3 active on F2 Bit 3 = 0(0): Out4 not active on F2 = 1(8): Out4 active on F2 Bit 4 = 0(0): Out5 not active on F2 = 1(1): Out5 active on F2 Bit 5 = 0(0): Out6 not active on F2 = 1(2): Out6 active on F2 Bit 6 = 0(0): Out7 not active on F2 = 1(4): Out7 active on F2 Bit 7 = 0(0): Out8 not active on F2 = 1(8): Out8 active on F2 | |
| 38 | 4 4 for PLUX22 | 0-255 | F3 mapping Bit 0 = 0(0): Out1 not active on F3 = 1(1): Out1 active on F3 Bit 1 = 0(0): Out2 not active on F3 = 1(2): Out2 active on F3 Bit 2 = 0(0): Out3 not active on F3 = 1(4): Out3 active on F3 Bit 3 = 0(0): Out4 not active on F3 | 38 |



| | | | | |
|----|-------------------|-------|---|----|
| | | | <p>= 1(8): Out4 active on F3 Bit 4 = 0(0): Out5 not active on F3 = 1(1): Out5 active on F3 Bit 5 = 0(0): Out6 not active on F3 = 1(2): Out6 active on F3 Bit 6 = 0(0): Out7 not active on F3 = 1(4): Out7 active on F3 Bit 7 = 0(0): Out8 not active on F3 = 1(8): Out8 active on F3</p> | |
| 39 | 8 8 for PLUX22 | 0-255 | <p>F4 mapping Bit 0 = 0(0): Out1 not active on F4 = 1(1): Out1 active on F4 Bit 1 = 0(0): Out2 not active on F4 = 1(2): Out2 active on F4 Bit 2 = 0(0): Out3 not active on F4 = 1(4): Out3 active on F4 Bit 3 = 0(0): Out4 not active on F4 = 1(8): Out4 active on F4 Bit 4 = 0(0): Out5 not active on F4 = 1(1): Out5 active on F4 Bit 5 = 0(0): Out6 not active on F4 = 1(2): Out6 active on F4</p> | 39 |



| | | | | |
|----|------------------------|-------|--|----|
| | | | Bit 6 = 0(0): Out7 not active on F4 = 1(4): Out7 active on F4 Bit 7 = 0(0): Out8 not active on F4 = 1(8): Out8 active on F4 | |
| 40 | 16 16 for PLUX22 | 0-255 | F5 mapping Bit 0 = 0(0): Out1 not active on F5 = 1(1): Out1 active on F5 Bit 1 = 0(0): Out2 not active on F5 = 1(2): Out2 active on F5 Bit 2 = 0(0): Out3 not active on F5 = 1(4): Out3 active on F5 Bit 3 = 0(0): Out4 not active on F5 = 1(8): Out4 active on F5 Bit 4 = 0(0): Out5 not active on F5 = 1(1): Out5 active on F5 Bit 5 = 0(0): Out6 not active on F5 = 1(2): Out6 active on F5 Bit 6 = 0(0): Out7 not active on F5 = 1(4): Out7 active on F5 Bit 7 = 0(0): Out8 not active on F5 = 1(8): Out8 active on F5 | 40 |
| 41 | 32 | 0-255 | F6 mapping | 41 |



| | | | | |
|----|------------------------|-------|--|----|
| | 32 for PLUX22 | | Bit 0 = 0(0): Out1 not active on F6 = 1(1): Out1 active on F6 Bit 1 = 0(0): Out2 not active on F6 = 1(2): Out2 active on F6 Bit 2 = 0(0): Out3 not active on F6 = 1(4): Out3 active on F6 Bit 3 = 0(0): Out4 not active on F6 = 1(8): Out4 active on F6 Bit 4 = 0(0): Out5 not active on F6 = 1(1): Out5 active on F6 Bit 5 = 0(0): Out6 not active on F6 = 1(2): Out6 active on F6 Bit 6 = 0(0): Out7 not active on F6 = 1(4): Out7 active on F6 Bit 7 = 0(0): Out8 not active on F6 = 1(8): Out8 active on F6 | |
| 42 | 64 64 for PLUX22 | 0-255 | F7 mapping Bit 0 = 0(0): Out1 not active on F7 = 1(1): Out1 active on F7 Bit 1 = 0(0): Out2 not active on F7 = 1(2): Out2 active on F7 Bit 2 = 0(0): Out3 not active on F7 | 42 |



| | | | | |
|----|-------------------------|-------|--|----|
| | | | <p>= 1(4): Out3 active on F7 Bit 3 = 0(0): Out4 not active on F7 = 1(8): Out4 active on F7 Bit 4 = 0(0): Out5 not active on F7 = 1(1): Out5 active on F7 Bit 5 = 0(0): Out6 not active on F7 = 1(2): Out6 active on F7 Bit 6 = 0(0): Out7 not active on F7 = 1(4): Out7 active on F7 Bit 7 = 0(0): Out8 not active on F7 = 1(8): Out8 active on F7</p> | |
| 43 | 128 16 for PLUX22 | 0-255 | <p>F8 mapping Bit 0 = 0(0): Out1 not active on F8 = 1(1): Out1 active on F8 Bit 1 = 0(0): Out2 not active on F8 = 1(2): Out2 active on F8 Bit 2 = 0(0): Out3 not active on F8 = 1(4): Out3 active on F8 Bit 3 = 0(0): Out4 not active on F8 = 1(8): Out4 active on F8 Bit 4 = 0(0): Out5 not active on F8 = 1(1): Out5 active on F8</p> | 43 |



| | | | | |
|----|-----------------------|-------|--|----|
| | | | Bit 5 = 0(0): Out6 not active on F8 = 1(2): Out6 active on F8 Bit 6 = 0(0): Out7 not active on F8 = 1(4): Out7 active on F8 Bit 7 = 0(0): Out8 not active on F8 = 1(8): Out8 active on F8 | |
| 44 | 0 32 for PLUX22 | 0-255 | F9 mapping Bit 0 = 0(0): Out1 not active on F9 = 1(1): Out1 active on F9 Bit 1 = 0(0): Out2 not active on F9 = 1(2): Out2 active on F9 Bit 2 = 0(0): Out3 not active on F9 = 1(4): Out3 active on F9 Bit 3 = 0(0): Out4 not active on F9 = 1(8): Out4 active on F9 Bit 4 = 0(0): Out5 not active on F9 = 1(16):Out5 active on F9 Bit 5 = 0(0): Out6 not active on F9 = 1(32):Out6 active on F9 Bit 6 = 0(0): Out7 not active on F9 = 1(64): Out7 active on F9 Bit 7 = 0(0): Out8 not active on F9 | 44 |



| | | | | |
|----|------------------------|-------|---|----|
| | | | = 1(128): Out8 active on F9 | |
| 45 | 0 64 for PLUX22 | 0-255 | F10 mapping Bit 0 = 0(0): Out1 not active on F10 = 1(1): Out1 active on F10 Bit 1 = 0(0): Out2 not active on F10 = 1(2): Out2 active on F10 Bit 2 = 0(0): Out3 not active on F10 = 1(4): Out3 active on F10 Bit 3 = 0(0): Out4 not active on F10 = 1(8): Out4 active on F10 Bit 4 = 0(0): Out5 not active on F10 = 1(16):Out5 active on F10 Bit 5 = 0(0): Out6 not active on F10 = 1(32):Out6 active on F10 Bit 6 = 0(0): Out7 not active on F10 = 1(64):Out7 active on F10 Bit 7 = 0(0): Out8 not active on F10 = 1(128): Out8 active on F10 | 45 |
| 46 | 0 128 for PLUX22 | 0-255 | F11 mapping Bit 0 = 0(0): Out1 not active on F11 = 1(1): Out1 active on F11 Bit 1 = 0(0): Out2 not active on F11 | 46 |



| | | | | |
|----|-------------------|-------|--|----|
| | | | <p>= 1(2): Out2 active on F11 Bit 2 = 0(0): Out3 not active on F11 = 1(4): Out3 active on F11 Bit 3 = 0(0): Out4 not active on F11 = 1(8): Out4 active on F11 Bit 4 = 0(0): Out5 not active on F11 = 1(16):Out5 active on F11 Bit 5 = 0(0): Out6 not active on F11 = 1(32):Out6 active on F11 Bit 6 = 0(0): Out7 not active on F11 = 1(64):Out7 active on F11 Bit 7 = 0(0): Out8 not active on F11 = 1(128):Out8 active on F11</p> | |
| 47 | 0 4 for PLUX22 | 0-255 | <p>F12 mapping Bit 0 = 0(0): Out1 not active on F12 = 1(1): Out1 active on F12 Bit 1 = 0(0): Out2 not active on F12 = 1(2): Out2 active on F12 Bit 2 = 0(0): Out3 not active on F12 = 1(4): Out3 active on F12 Bit 3 = 0(0): Out4 not active on F12 = 1(8): Out4 active on F12</p> | 47 |



| | | | | |
|----|-----|---------------------|--|--|
| | | | Bit 4 = 0(0): Out5 not active on F12 = 1(16):Out5 active on F12 Bit 5 = 0(0): Out6 not active on F12 = 1(32):Out6 active on F12 Bit 6 = 0(0): Out7 not active on F12 = 1(64):Out7 active on F12 Bit 7 = 0(0): Out8 not active on F12 = 1(128): Out8 active on F12 | |
| 48 | 255 | 0-255 | Out 1 Light intensity, [1-255] | |
| 49 | 255 | 0-255 | Out 2 Light intensity, [1-255] | |
| 50 | 255 | 0-255 | Out 3 Light intensity, [1-255] | |
| 51 | 255 | 0-255 | Out 4 Light intensity, [1-255] | |
| 52 | 255 | 0-255 | Out 5 Light intensity, [1-255] | |
| 53 | 255 | 0-255 | Out 6 Light intensity, [1-255] | |
| 54 | 255 | 0-255 | Out 7 Light intensity, [1-255] | |
| 55 | 255 | 0-255 | Out 8 Light intensity, [1-255] | |
| 56 | 255 | 0-255 | Out 9 Light intensity, [1-255] | |
| 57 | 255 | 0-255 | Out 10 Light intensity, [1-255] | |
| 60 | 1 | 0,1, 128, 129 | Motor PID and PWM Control Bit 0 = 0(0): PID Control Disabled = 1(1): PID Control Enabled | |



| | | | |
|-------|-----|-------|---|
| | | | Bit7 = 0(0): Motor PWM Frequency 32kHz = 1(128): Motor PWM Frequency 16kHz |
| 61 | 80 | 0-255 | PID P constant |
| 62 | 120 | 0-255 | PID I constant |
| 63 | 40 | 0-255 | PID D constant |
| 64 | 1 | 0-15 | Brake, 0-No brake, 1-15 Braking rate, Value influence Constant Braking Distance, 1-Value Shortest Braking distance from maximum Speed, Increase value to increase braking distance, Distance=Value x Shortest Distance |
| 65 | 0 | 0-255 | BrakeDelay, 0-No Delay, To increase stopping distance in small amount increase the value, BrakeDelay = Value * 8ms (ms) Extra Distance = MaxSpeed * BrakeDelay Ex: 200ms(delay)=8(ms)*25(value) |
| 67 | 2 | 1-255 | Speed Table Step 1 Value |
| | | | |
| 94 | 240 | 1-255 | Speed Table Step 28 Value |
| 95 | 1 | 1-10 | PID error limit |
| 105 | 0 | 0-255 | USER data |
| 106 | 0 | 0-255 | USER data |
| 112 | 50 | 1-127 | Light Effect Fade ON (turn on delay), ex.:1=8ms, 15=120ms 125=1000ms |
| 113 | 25 | 1-127 | Light Effect Fade OFF (turn off delay) |



| | | | |
|-----|----|-------|---|
| 114 | 4 | 0-255 | Shunting speed, Function mapping F1-F8, F3 default |
| 115 | 8 | 0-255 | Switch Off Acceleration Deceleration, Function mapping, F4 default |
| 116 | 16 | 0-255 | Disable Constant Braking, Function mapping, F5 default |
| 117 | 0 | 0-255 | No Effect(Fading) on AUX, continues signal, Output Mapping Bit 0 = 0(0): FL could be dimmed and faded (PWM signal) = 1(1): continues signal with no fading on FL Bit 1 = 0(0): RL could be dimmed and faded (PWM signal) = 1(2): continues signal with no fading on RL Bit 2 = 0(0): AUX1 could be dimmed and faded (PWM signal) = 1(4): continues signal with no fading on AUX1 Bit 3 = 0(0): AUX2 could be dimmed and faded (PWM signal) = 1(8): continues signal with no fading on AUX2 Bit 4 = 0(0): AUX3 could be dimmed and faded (PWM signal) = 1(16): continues signal with no fading on AUX3 Bit 5 = 0(0): AUX4 could be dimmed and faded (PWM signal) = 1(32): continues signal with no fading on AUX4 Bit 6 = 0(0): AUX5 could be dimmed and faded (PWM signal) = 1(64): continues signal with no fading on AUX5 Bit 7 = 0(0): AUX6 could be dimmed and faded (PWM signal) = 1(128): continues signal with no fading on AUX6 |
| 118 | 0 | 0-12 | Electrical Coupler Output mapping. Only one of the outputs can be configured as ECoupler Output |



Lokommander II

User Manual firmware 3.5.195

Version

0.1.22

| | | | |
|-----|----|-------|--|
| | | | CV118 = 0, None of the AUX selected for ECoupler operation CV118 = 1, FL selected for ECoupler operation CV118 = 2, RL selected for ECoupler operation CV118 = 3, AUX1 selected for ECoupler operation CV118 = 12, AUX10 selected for ECoupler operation |
| 119 | 50 | 0-255 | Electrical Coupler, Kick_time = Val*8ms, ex: 400ms=50*8ms |
| 120 | 50 | 0-255 | Decoupling, Locomotive move Time=Val*8ms, ex: 400ms=50*8ms |
| 121 | 50 | 0-255 | Decoupling Locomotive moving speed |
| 122 | 71 | 0-255 | Configuration: Bit 0 = 0(0): SUSI pins used as PWM Outputs (AUX) = 1(1): SUSI pins used as SUSI CLK/SUSI DATA or Locowire Bit 1 = 0(0): Locowire Interface active = 1(2): SUSI Interface active Bit 2 = 0(0): No Load transmission ower SUSI = 1(4): Load transmission ower SUSI active Bit 3 = 0(0): Motor PWM weighting OFF = 1(8): Motor PWM weighting with variation of track voltage Bit 4,5= 00(0): No Push-Pull operation = 10(16) Push-Pull operation without intermediate stop active = 01(32) Push-Pull operation with intermediate stop active = 11(48) Not permitted, must be avoided! |



| | | | |
|-----|-----|--------|---|
| | | | Bit 6 = 0(0): FL/RL inactive during firmware update = 1(64): During Firmware update the FL/RL outputs blinks Bit 7 = 0(0): No saving = 1(128): Enable saving QoS and Temperature to Eeprom |
| 123 | 16 | 0-255 | SPP (Smart Power Pack) Timeout=16ms*Value Ex: =16ms*16=256ms |
| 124 | 0 | 0-1 | ECoupler Mode CV124 = 0, PWM Output CV124 = 1, Full Output on selected AUX in CV118 |
| 126 | 102 | 0-255 | SUSI CV transport, SUSI CV=800+Value |
| 127 | | 0-255 | SUSI DATA transport, Data write to CV=800+cv126 |
| 128 | 1 | 1-4 | If CV9 = 9, user mode PWM Periode |
| 129 | 6 | 1-10 | If CV9 = 9, user mode Bemf average |
| 130 | 6 | 1-12 | If CV9 = 9, user mode Bemf delay |
| 133 | | | Chip temperature read out. Prior the readout F5 function must be switched On and Off |
| 134 | 100 | 60-120 | TempLimit, 85o C Temperature protection |
| 135 | | 0-100% | QoS (Quality of Service) current value (only after saving with F5, if the function is enabled in CV122 bit 7. Only Readable |
| 136 | | 0-100% | Worst QoS (Quality of Service) value (only after saving with F5, if the function is enabled in CV122 bit 7. Only Readable |
| 137 | 60 | 0-255 | Vmedium for Load Compensation (medium speed for Load Compensation) |



| | | | |
|-----|-----|-------|--|
| 138 | 150 | 0-255 | Load Compensation at Vmin |
| 139 | 100 | 0-255 | Load Compensation at Vmedium (it could be different than Vmid) |
| 140 | 80 | 0-255 | Load Compensation at Vmax |
| 141 | 14 | 0-50 | ABC Sesity, ABC detection treshold in 0.1V (14 => 1,4V treshold) |
| 142 | 10 | 0-255 | Wait time when stationary in pendulling mode, in seconds |
| 143 | 15 | 0-255 | Forward speed, when ABC slow speed it's triggered in forward direction |
| 144 | 50 | 0-255 | ABC start delay after power-up |
| 145 | 85 | 0-255 | DC starting treshold voltage, in 0,1V steps (85 => 8,5V) |
| 146 | 65 | 0-255 | DC stop treshold voltage, in 0,1V steps (65 => 6,5V) |
| 147 | 160 | 0-160 | DC max speed treshold voltage, in 0,1V steps (160 => 16V) |
| 148 | 0 | 1-63 | Acceleration Rate Reverse: 0 = use CV3 value |
| 149 | 0 | 0-63 | Deceleration Rate Reverse 0 = use CV4 value |
| 150 | 0 | 0-15 | CBD Brake Reverse, 0-use CV64 value, 1-15 Braking rate reverse, Value influence Constant Braking Distance in reverse movement, 1-Value Shortest Braking distance from maximum Speed, Increase value to increase braking distance, Distance=Value x Shortest Distance |
| 151 | 0 | 0-255 | CBD BrakeDelay reverse, 0-use CV65 value, To increase stopping distance in small amount increase the value, BrakeDelay = Value * 8ms (ms) Extra Distance = MaxSpeed * BrakeDelay Ex: 200ms(delay)=8(ms)*25(value) |
| 152 | 10 | 0-255 | SPP (Smart Power Pack) start delay in seconds, default 10s |



| | | | |
|-----|-----|-------|---|
| 153 | 0 | 0-255 | CBD Brake_distance forward |
| 154 | 0 | 0-15 | Maintenance Configuration Bit0 = 0 (0): maintenance function disabled = 1 (1): maintenance function enabled Bit1 = 0 (0): MI overrun is not signaled in CV30, bit3 = 1 (2): MI overrun is signaled in CV30, bit3 Bit2 = 0 (0): MI overrun is not signaled with FL/RL = 1(4): MI overrun is signaled by FL/RL alternating with low frequency Bit3 = 0 (0): exceeding MI by 50% is not signaled by FL/RL = 1(8): exceeding MI by 50% is signaled by FL/RL alternating with high frequency |
| 155 | 40 | 0-255 | Maintenance Interval MI (hours) |
| 156 | | 0-59 | Work minutes |
| 157 | | 0-255 | Work hours low byte |
| 158 | | 0-255 | Work hours high byte |
| 159 | | 0-255 | Last maintenance hour low byte |
| 160 | | 0-255 | Last maintenance hour high byte |
| 161 | 0 | 0-255 | CBD Brake_distance reverse |
| 162 | 100 | 0-255 | DC brake threshold voltage – see chap. Error! Reference source not found. |



| | | | |
|-----|---------|-------|---|
| | | | Vcc > threshold => DC brake Vcc < threshold => SPP timeout |
| 163 | 15 | 0-255 | Reverse speed, when ABC slow speed it's triggered in reverse direction |
| 164 | 255(64) | 0-255 | Motor PWM value for DC-mode2 |
| 165 | 255 | 0-255 | CV operation acknowledge mapping on outputs - see chapter 8 |
| 166 | 0 | 0-255 | Inhibition of outputs with F0f (F0 forward movement) – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F0 (forward) = 1(1): FL inhibited by F0 (forward) Bit 1 = 0(0): FR not inhibited by F0 (forward) = 1(2): FR inhibited by F0 (forward) Bit 2 = 0(0): Aux1 not inhibited by F0 (forward) = 1(4): Aux1 inhibited by F0 (forward) Bit 3 = 0(0): Aux2 not inhibited by F0 (forward) = 1(8): Aux2 inhibited by F0 (forward) Bit 4 = 0(0): Aux3 not inhibited by F0 (forward) = 1(16): Aux3 inhibited by F0 (forward) Bit 5 = 0(0): Aux4 not inhibited by F0 (forward) = 1(32): Aux4 inhibited by F0 (forward) Bit 6 = 0(0): Aux5 not inhibited by F0 (forward) = 1(64): Aux5 inhibited by F0 (forward) Bit 7 = 0(0): Aux6 not inhibited by F0 (forward) |



| | | | |
|-----|---|-------|---|
| | | | = 1(128): Aux6 inhibited by F0 (mers înainte) |
| 167 | 0 | 0-255 | Inhibition of outputs with F0r (F0 backward movement) – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F0 (backward) = 1(1): FL inhibited by F0 (backward) Bit 1 = 0(0): FR not inhibited by F0 (backward) = 1(2): FR inhibited by F0 (backward) Bit 2 = 0(0): Aux1 not inhibited by F0 (backward) = 1(4): Aux1 inhibited by F0 (backward) Bit 3 = 0(0): Aux2 not inhibited by F0 (backward) = 1(8): Aux2 inhibited by F0 (backward) Bit 4 = 0(0): Aux3 not inhibited by F0 (backward) = 1(16): Aux3 inhibited by F0 (backward) Bit 5 = 0(0): Aux4 not inhibited by F0 (backward) = 1(32): Aux4 inhibited by F0 (backward) Bit 6 = 0(0): Aux5 not inhibited by F0 (backward) = 1(64): Aux5 inhibited by F0 (backward) Bit 7 = 0(0): Aux6 not inhibited by F0 (backward) = 1(128): Aux6 inhibited by F0 (backward) |
| 168 | 0 | 0-255 | Inhibition of outputs with F1f (F1 forward movement) – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F1 (forward) |



| | | | |
|-----|---|-------|--|
| | | | <p>= 1(1): FL inhibited by F1 (forward) Bit 1 = 0(0): FR not inhibited by F1 (forward) = 1(2): FR inhibited by F1 (forward) Bit 2 = 0(0): Aux1 not inhibited by F1 (forward) = 1(4): Aux1 inhibited by F1 (forward) Bit 3 = 0(0): Aux2 not inhibited by F1 (forward) = 1(8): Aux2 inhibited by F1 (forward) Bit 4 = 0(0): Aux3 not inhibited by F1 (forward) = 1(16): Aux3 inhibited by F1 (forward) Bit 5 = 0(0): Aux4 not inhibited by F1 (forward) = 1(32): Aux4 inhibited by F1 (forward) Bit 6 = 0(0): Aux5 not inhibited by F1 (forward) = 1(64): Aux5 inhibited by F1 (forward) Bit 7 = 0(0): Aux6 not inhibited by F1 (forward) = 1(128): Aux6 inhibited by F1 (forward)</p> |
| 169 | 0 | 0-255 | <p>Inhibition of outputs with F2 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F2 = 1(1): FL inhibited by F2 Bit 1 = 0(0): FR not inhibited by F2 = 1(2): FR inhibited by F2 Bit 2 = 0(0): Aux1 not inhibited by F2</p> |



Lokommander II

User Manual firmware 3.5.195

Version

0.1.22

| | | | |
|-----|---|-------|---|
| | | | <p>= 1(4): Aux1 inhibited by F2 Bit 3 = 0(0): Aux2 not inhibited by F2 = 1(8): Aux2 inhibited by F2 Bit 4 = 0(0): Aux3 not inhibited by F2 = 1(16): Aux3 inhibited by F2 Bit 5 = 0(0): Aux4 not inhibited by F2 = 1(32): Aux4 inhibited by F2 Bit 6 = 0(0): Aux5 not inhibited by F2 = 1(64): Aux5 inhibited by F2 Bit 7 = 0(0): Aux6 not inhibited by F2 = 1(128): Aux6 inhibited by F2</p> |
| 170 | 0 | 0-255 | <p>Inhibition of outputs with F3 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F3 = 1(1): FL inhibited by F3 Bit 1 = 0(0): FR not inhibited by F3 = 1(2): FR inhibited by F3 Bit 2 = 0(0): Aux1 not inhibited by F3 = 1(4): Aux1 inhibited by F3 Bit 3 = 0(0): Aux2 not inhibited by F3 = 1(8): Aux2 inhibited by F3 Bit 4 = 0(0): Aux3 not inhibited by F3</p> |



| | | | |
|-----|---|-------|---|
| | | | <p>= 1(16): Aux3 inhibited by F3 Bit 5 = 0(0): Aux4 not inhibited by F3 = 1(32): Aux4 inhibited by F3 Bit 6 = 0(0): Aux5 not inhibited by F3 = 1(64): Aux5 inhibited by F3 Bit 7 = 0(0): Aux6 not inhibited by F3 = 1(128): Aux6 inhibited by F3</p> |
| 171 | 0 | 0-255 | <p>Inhibition of outputs with F4 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F4 = 1(1): FL inhibited by F4 Bit 1 = 0(0): FR not inhibited by F4 = 1(2): FR inhibited by F4 Bit 2 = 0(0): Aux1 not inhibited by F4 = 1(4): Aux1 inhibited by F4 Bit 3 = 0(0): Aux2 not inhibited by F4 = 1(8): Aux2 inhibited by F4 Bit 4 = 0(0): Aux3 not inhibited by F4 = 1(16): Aux3 inhibited by F4 Bit 5 = 0(0): Aux4 not inhibited by F4 = 1(32): Aux4 inhibited by F4 Bit 6 = 0(0): Aux5 not inhibited by F4</p> |



| | | | |
|-----|---|-------|--|
| | | | <p>= 1(64): Aux5 inhibited by F4 Bit 7 = 0(0): Aux6 not inhibited by F4 = 1(128): Aux6 inhibited by F4</p> |
| 172 | 0 | 0-255 | <p>Inhibition of outputs with F5 – see chap. Error! Reference source not found.</p> <p>Bit 0 = 0(0): FL not inhibited by F5 = 1(1): FL inhibited by F5 Bit 1 = 0(0): FR not inhibited by F5 = 1(2): FR inhibited by F5 Bit 2 = 0(0): Aux1 not inhibited by F5 = 1(4): Aux1 inhibited by F5 Bit 3 = 0(0): Aux2 not inhibited by F5 = 1(8): Aux2 inhibited by F5 Bit 4 = 0(0): Aux3 not inhibited by F5 = 1(16): Aux3 inhibited by F5 Bit 5 = 0(0): Aux4 not inhibited by F5 = 1(32): Aux4 inhibited by F5 Bit 6 = 0(0): Aux5 not inhibited by F5 = 1(64): Aux5 inhibited by F5 Bit 7 = 0(0): Aux6 not inhibited by F5 = 1(128): Aux6 inhibited by F5</p> |
| 173 | 0 | 0-255 | <p>Inhibition of outputs with F6 – see chap. Error! Reference source not</p> |



| | | | |
|-----|---|-------|---|
| | | | <p>found.</p> <p>Bit 0 = 0(0): FL not inhibited by F6 = 1(1): FL inhibited by F6</p> <p>Bit 1 = 0(0): FR not inhibited by F6 = 1(2): FR inhibited by F6</p> <p>Bit 2 = 0(0): Aux1 not inhibited by F6 = 1(4): Aux1 inhibited by F6</p> <p>Bit 3 = 0(0): Aux2 not inhibited by F6 = 1(8): Aux2 inhibited by F6</p> <p>Bit 4 = 0(0): Aux3 not inhibited by F6 = 1(16): Aux3 inhibited by F6</p> <p>Bit 5 = 0(0): Aux4 not inhibited by F6 = 1(32): Aux4 inhibited by F6</p> <p>Bit 6 = 0(0): Aux5 not inhibited by F6 = 1(64): Aux5 inhibited by F6</p> <p>Bit 7 = 0(0): Aux6 not inhibited by F6 = 1(128): Aux6 inhibited by F6</p> |
| 174 | 0 | 0-255 | <p>Inhibition of outputs with F7 – see chap. Error! Reference source not found.</p> <p>Bit 0 = 0(0): FL not inhibited by F7 = 1(1): FL inhibited by F7</p> <p>Bit 1 = 0(0): FR not inhibited by F7</p> |



| | | | |
|-----|---|-------|--|
| | | | <p>= 1(2): FR inhibited by F7 Bit 2 = 0(0): Aux1 not inhibited by F7 = 1(4): Aux1 inhibited by F7 Bit 3 = 0(0): Aux2 not inhibited by F7 = 1(8): Aux2 inhibited by F7 Bit 4 = 0(0): Aux3 not inhibited by F7 = 1(16): Aux3 inhibited by F7 Bit 5 = 0(0): Aux4 not inhibited by F7 = 1(32): Aux4 inhibited by F7 Bit 6 = 0(0): Aux5 not inhibited by F7 = 1(64): Aux5 inhibited by F7 Bit 7 = 0(0): Aux6 not inhibited by F7 = 1(128): Aux6 inhibited by F7</p> |
| 175 | 0 | 0-255 | <p>Inhibition of outputs with F8 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F8 = 1(1): FL inhibited by F8 Bit 1 = 0(0): FR not inhibited by F8 = 1(2): FR inhibited by F8 Bit 2 = 0(0): Aux1 not inhibited by F8 = 1(4): Aux1 inhibited by F8 Bit 3 = 0(0): Aux2 not inhibited by F8</p> |



| | | | |
|-----|---|-------|--|
| | | | <p>= 1(8): Aux2 inhibited by F8 Bit 4 = 0(0): Aux3 not inhibited by F8 = 1(16): Aux3 inhibited by F8 Bit 5 = 0(0): Aux4 not inhibited by F8 = 1(32): Aux4 inhibited by F8 Bit 6 = 0(0): Aux5 not inhibited by F8 = 1(64): Aux5 inhibited by F8 Bit 7 = 0(0): Aux6 not inhibited by F8 = 1(128): Aux6 inhibited by F8</p> |
| 176 | 0 | 0-255 | <p>Inhibition of outputs with F9 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F9 = 1(1): FL inhibited by F9 Bit 1 = 0(0): FR not inhibited by F9 = 1(2): FR inhibited by F9 Bit 2 = 0(0): Aux1 not inhibited by F9 = 1(4): Aux1 inhibited by F9 Bit 3 = 0(0): Aux2 not inhibited by F9 = 1(8): Aux2 inhibited by F9 Bit 4 = 0(0): Aux3 not inhibited by F9 = 1(16): Aux3 inhibited by F9 Bit 5 = 0(0): Aux4 not inhibited by F9</p> |



| | | | |
|-----|---|-------|--|
| | | | <p>= 1(32): Aux4 inhibited by F9 Bit 6 = 0(0): Aux5 not inhibited by F9 = 1(64): Aux5 inhibited by F9 Bit 7 = 0(0): Aux6 not inhibited by F9 = 1(128): Aux6 inhibited by F9</p> |
| 177 | 0 | 0-255 | <p>Inhibition of outputs with F10 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F10 = 1(1): FL inhibited by F10 Bit 1 = 0(0): FR not inhibited by F10 = 1(2): FR inhibited by F10 Bit 2 = 0(0): Aux1 not inhibited by F10 = 1(4): Aux1 inhibited by F10 Bit 3 = 0(0): Aux2 not inhibited by F10 = 1(8): Aux2 inhibited by F10 Bit 4 = 0(0): Aux3 not inhibited by F10 = 1(16): Aux3 inhibited by F10 Bit 5 = 0(0): Aux4 not inhibited by F10 = 1(32): Aux4 inhibited by F10 Bit 6 = 0(0): Aux5 not inhibited by F10 = 1(64): Aux5 inhibited by F10 Bit 7 = 0(0): Aux6 not inhibited by F10</p> |



| | | | |
|-----|---|-------|---|
| | | | = 1(128): Aux6 inhibited by F10 |
| 178 | 0 | 0-255 | Inhibition of outputs with F11 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F11 = 1(1): FL inhibited by F11 Bit 1 = 0(0): FR not inhibited by F11 = 1(2): FR inhibited by F11 Bit 2 = 0(0): Aux1 not inhibited by F11 = 1(4): Aux1 inhibited by F11 Bit 3 = 0(0): Aux2 not inhibited by F11 = 1(8): Aux2 inhibited by F11 Bit 4 = 0(0): Aux3 not inhibited by F11 = 1(16): Aux3 inhibited by F11 Bit 5 = 0(0): Aux4 not inhibited by F11 = 1(32): Aux4 inhibited by F11 Bit 6 = 0(0): Aux5 not inhibited by F11 = 1(64): Aux5 inhibited by F11 Bit 7 = 0(0): Aux6 not inhibited by F11 = 1(128): Aux6 inhibited by F11 |
| 179 | 0 | 0-255 | Inhibition of outputs with F12 – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F12 |



| | | | |
|-----|---|-------|---|
| | | | <p>= 1(1): FL inhibited by F12 Bit 1 = 0(0): FR not inhibited by F12 = 1(2): FR inhibited by F12 Bit 2 = 0(0): Aux1 not inhibited by F12 = 1(4): Aux1 inhibited by F12 Bit 3 = 0(0): Aux2 not inhibited by F12 = 1(8): Aux2 inhibited by F12 Bit 4 = 0(0): Aux3 not inhibited by F12 = 1(16): Aux3 inhibited by F12 Bit 5 = 0(0): Aux4 not inhibited by F12 = 1(32): Aux4 inhibited by F12 Bit 6 = 0(0): Aux5 not inhibited by F12 = 1(64): Aux5 inhibited by F12 Bit 7 = 0(0): Aux6 not inhibited by F12 = 1(128): Aux6 inhibited by F12</p> |
| 180 | 0 | 0-255 | <p>Inhibition of outputs with F1r (F1 backward) – see chap. Error! Reference source not found. Bit 0 = 0(0): FL not inhibited by F1 (backward) = 1(1): FL inhibited by F1 (backward) Bit 1 = 0(0): FR not inhibited by F1 (backward) = 1(2): FR inhibited by F1 (backward) Bit 2 = 0(0): Aux1 not inhibited by F1 (backward)</p> |



| | | | |
|-----|---|------|---|
| | | | <p>= 1(4): Aux1 inhibited by F1 (backward) Bit 3 = 0(0): Aux2 not inhibited by F1 (backward) = 1(8): Aux2 inhibited by F1 (backward) Bit 4 = 0(0): Aux3 not inhibited by F1 (backward) = 1(16): Aux3 inhibited by F1 (backward) Bit 5 = 0(0): Aux4 not inhibited by F1 (backward) = 1(32): Aux4 inhibited by F1 (backward) Bit 6 = 0(0): Aux5 not inhibited by F1 (backward) = 1(64): Aux5 inhibited by F1 (backward) Bit 7 = 0(0): Aux6 not inhibited by F1 (backward) = 1(128): Aux6 inhibited by F1 (backward)</p> |
| 181 | | | Saving the last state of functions and outputs: 0 - disabled, 1 - enabled |
| 182 | | | Status saved FL, RL |
| 183 | | | Status saved F1 - F8 |
| 184 | | | Status saved F9 - F16 |
| 185 | 0 | 0-15 | <p>No Effect(Fading) on Outputs, continues signal, Aux7 ... Aux10 Bit 0 = 0(0): Aux7 could be dimmed and faded (PWM signal) = 1(1): continues signal with no fading on Aux7 Bit 1 = 0(0): Aux8 could be dimmed and faded (PWM signal) = 1(2): continues signal with no fading on Aux8 Bit 2 = 0(0): Aux9 could be dimmed and faded (PWM signal) = 1(4): continues signal with no fading on Aux9</p> |



| | | | |
|--|--|--|--|
| | | | Bit 3 = 0(0): Aux10 could be dimmed and faded (PWM signal) = 1(8): continues signal with no fading on Aux10 |
|--|--|--|--|



27. Bits and bytes

If we want to modify the values of the configuration variables (CV), it is good to keep a few notions regarding the representation of numbers in binary format. In binary format we have only two digits 0 and 1. A binary number is called a bit. An 8-bit group will call a byte, representing a binary number of 8 binary digits. Configuration variables, CV, are bytes stored in non-volatile memory of decoders. The bits of a byte are numbered from 0 to 7. Bit 0, it's the least significant (LSB), has the decimal value of 1 and bit (7) it's the most significant (MSB), has the decimal value of 128.

Some control stations, used to modify CVs, display the value and allow entry only in decimal format. In this case, it is good to know how to find the state of a bit from the decimal value read, or how to calculate the decimal value you have to write in the CV based on the desired bit configuration.

| | MSB | | | | | | | LSB |
|--------------|------|------|------|------|------|------|------|------|
| Bit position | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Bit value | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

If we know the bit configuration and we want to find out the decimal value, we use the following calculation formula:

$$\text{Dec} = B7 * 128 + B6 * 64 + B5 * 32 + B4 * 16 + B3 * 8 + B2 * 4 + B1 * 2 + B0$$

where B0 ... B7 represents the value of the respective bit (0 or 1). For example, if B7 = 1, B5 = 1, B2 = 1, in the rest 0, we will have:

$$\begin{aligned} \text{Dec} &= 1 * 128 + 0 * 64 + 1 * 32 + 0 * 16 + 0 * 8 + 1 * 4 + 0 * 2 + 0 = \\ &= 128 + 32 + 4 = 164 \end{aligned}$$

If we want to find the bits configuration from the decimal value, we do the opposite. We try to subtract from the decimal value the bits value begin with MSB and we keep the difference for the next subtractions until we get zero. For possible subtractions, with a



positive result, the bit will have a value of 1. For the impossible subtractions, when the difference is negative, we abandon the operation (the value of the bit will be zero) and continue with the next decrease.

For example, we want to find the bits configuration for decimal value 73:

| | |
|------------------|------------------------|
| $73 - 128 = -55$ | \Rightarrow Bit7 = 0 |
| $73 - 64 = 9$ | \Rightarrow Bit6 = 1 |
| $9 - 32 = -23$ | \Rightarrow Bit5 = 0 |
| $9 - 16 = -7$ | \Rightarrow Bit4 = 0 |
| $9 - 8 = 1$ | \Rightarrow Bit3 = 1 |
| $1 - 4 = -3$ | \Rightarrow Bit2 = 0 |
| $1 - 2 = -1$ | \Rightarrow Bit1 = 0 |
| $1 - 1 = 0$ | \Rightarrow Bit0 = 1 |

CV tool

Cv tool is a small utility program to convert the value of decimal bits to binary and vice versa or to calculate the value of extended addresses.

It can be downloaded from the following address:

<https://train-o-matic.com/downloads/software/cvTool.zip>



CV Tool V 1.01.005 ☒

CV / Bit Value Calculator

bit 0, value = 2 to the power of 0 = 1

bit 1, value = 2 to the power of 1 = 0

bit 2, value = 2 to the power of 2 = 4

bit 3, value = 2 to the power of 3 = 0

bit 4, value = 2 to the power of 4 = 16

bit 5, value = 2 to the power of 5 = 32

bit 6, value = 2 to the power of 6 = 0

bit 7, value = 2 to the power of 7 = 128

CV Value [0-255]

Long Address Calculator

Long Address [0-10239]

CV17 Value [192-231]

CV18 Value [0-255]



Copyright © 2019 Tehnologistic Ltd.
All rights reserved
**The information in this document is subject to change without
notice**

“train-o-matic” and the  logo are registered
trademarks of Tehnologistic Ltd.

www.train-O-matic.com

**ABC Technology and RailCom are registered trademarks
of Lenz electronics**

<http://www.digital-plus.de>

**SUSI and the  are registered trademarks of DIETZ
ELEKTRONIK**

<http://www.d-i-e-t-z.de>

**Tehnologistic SRL
Str. Libertatii 35A
407035 Apahida
Romania**

