

# **Description of the mfx®Rail format**

*Author: Stefan Krauss*

*Team: Rainer Müller, Mike F. Schwarzer, Stefan Krauß,  
Knut Schwichtenberg, Erwin Stegmaier*

*Version 2.3 from 20170617*

mfx is a registered trademark of Gebrüder Märklin & Cie. GmbH,  
73033 Göppingen, Germany

## **Notice**

Out of private curiosity, some technically interested model railroaders (the team) created the mfx®Format. This document describes the relevant observations and conclusions; it reflects the private opinion of the authors. This description is not an official format specification and is not endorsed by those involved in the development or marketing of mfx®participating companies authorized.

The document is aimed at technically interested model railroaders and should be understood as a journalistic work. No commercial interests are pursued.

## **Disclaimer**

The author and team exclude any claims for damages that may arise from incorrect, incomplete or inaccurate information in this document. In particular, it is pointed out that all information is for informational purposes only, suitability for a specific purpose cannot be guaranteed and third-party intellectual property rights may be affected. It is to be expected that the document is incomplete and partially incorrect.

## **Copyrights**

The author holds the copyright to this document. All texts, drawings and images may not be used in whole or in part without the written consent of the author. The author can be contacted via the following email address or website:

[stefan@skrauss.de](mailto:stefan@skrauss.de)  
<http://www.skrauss.de>

The unmodified PDF file may be distributed privately and printed for private purposes. Commercial use is prohibited.

This copyright notice does not refer to the facts and technical processes presented. These may be protected by copyright, patent or other intellectual property rights of third parties. Reproduction in this document is for information purposes only.

## **Trademark rights**

Märklin, mfx and mfx+ are registered trademarks of Gebrüder Märklin & Cie. GmbH, 73033 Göppingen, Germany. Trix is a registered trademark of Märklin Holding GmbH, 73037 Göppingen, Germany.

ESU and ECoS are registered trademarks of ESU electronic solutions ulm GmbH & Co. KG, 89081 Ulm, Germany.

Motorola is a registered trademark of Motorola Inc., Schaumburg, Illinois, USA.

RDS is a registered trademark of THOMSON Licensing SA, Boulogne Billancourt, France.

Other names may be subject to third-party property rights. Trademarks and other protected names are used in this document for information purposes only.

## Table of contents

1 Introduction.....	5
1.1 Purpose of the document.....	5
1.2 Document history.....	5
1.3 Conventions.....	6
1.4 Abbreviations.....	6
1.5 Property rights.....	6
1.6 References and literature.....	7
2 Signal and coding.....	8
2.1 Transmission principle.....	8
2.2 Control channel.....	9
2.2.1 Bit coding and synchronization.....	9
2.2.2 Stuffing.....	10
2.2.3 Bit order.....	11
2.2.4 Structure of the data frames.....	11
2.2.5 Checksum.....	13
2.2.6 Principle of feedback.....	13
2.2.7 Feedback window 1BitFeedback.....	14
2.2.8 Feedback window data feedback.....	15
2.2.9 Embedding control commands in MM format.....	16
2.3 Return channel.....	16
2.3.1 Modulation.....	17
2.3.2 Coding of the 1-bit feedback.....	17
2.3.3 Coding for data feedback.....	17
2.3.4 Checksum for data feedback.....	18
3 Commands.....	20
3.1 Commands.....	20
3.1.1 No command.....	20
3.1.2 Command 000: Drive (short) .....	20
3.1.3 Command 001: Drive.....	20
3.1.4 Command 010: Functions (short).....	21
3.1.5 Command 011: Functions (extended) .....	21
3.1.6 Command 100: Functions (individual control) .....	21
3.1.7 Command 101: reserved.....	21
3.1.8 Command 110: reserved.....	21
3.1.9 Command 111: Configuration.....	21
3.2 Advanced configuration commands.....	22
3.2.1 Command 111 000: Read CV.....	22
3.2.2 Command 111 001: Write CV.....	22
3.2.3 Command 111 010: DecoderSearch.....	22
3.2.4 Command 111 011: Assignment of rail address.....	22
3.2.5 Command 111 100: Existence query (Ping, mfx®Verify).....	23
3.2.6 Command 111 101: Headquarters.....	23

3.2.7 Command 111 110: reserved.....	23
3.2.8 Command 111 111: reserved.....	23
4 Protocol.....	24
4.1 Sleep mode.....	24
4.2 Switching on: Decoder receives power.....	25
4.3 Registration.....	25
4.4 Log out.....	27
4.5 Resetting the control panel.....	27
4.6 Driving and operating.....	28
4.7 Multiple traction.....	28
4.8 Emergency stop.....	28
4.9 Configure.....	29
4.10 Protocol switching.....	30
5 Extension mfx+®.....	31
5.1 Functional principle.....	31
5.2 Procedures and processes.....	32
5.2.1 Mode setting.....	32
5.2.2 Resource management.....	32
5.2.3 Emergency operation.....	33
5.2.4 Operational recording.....	33
5.2.5 Interaction with the headquarters.....	33
6 Decoder configuration.....	35
6.1 Addressing.....	35
6.2 Encoding and byte order.....	35
6.3 Structure.....	36
6.3.1 Structure of the configuration space.....	36
6.3.2 Rules for the structures for fast registration.....	38
6.4 Description of the configuration variables.....	39
6.4.1 Block 01: Decoder properties.....	39
6.4.2 Block 02: Functions.....	44
6.4.3 Block 03: Automatic functions.....	44
6.4.4 Block 04: Function assignment.....	45
6.4.5 Block 05: Driving characteristics.....	49
6.4.6 Block 06: Hardware functions.....	51
6.4.7 Block 07: Additional protocols.....	53
6.4.8 Block 08: Sound settings.....	56
6.4.9 Block 09: Special options.....	58
6.4.10 Block 0A: mfx+®.....	58
6.4.11 Decoder reset.....	61

# 1 Introduction

## 1.1 Purpose of the document

With the introduction of the new digital system for controlling model railways from Märklin® In 2004, a new rail format was introduced. The new format and the associated decoders are called mfx®. The technical background and the exact functionality of the new digital format were discussed by the companies Märklin® and ESU®. Unfortunately, very little has been published. An official technical specification is not available.

Some technically interested and experienced model railroaders – the team – have painstakingly examined the rail signal and tried to understand the functionality of mfx®. This has been largely successful. This document contains the results of these investigations. It describes the mfx® Rail signal as interpreted by the team. This also explains the gaps and the errors that are certainly included; some of it may also be speculation, as the intention of the original developers is not known.

The informative presentation is intended to give the technically interested model railroader an insight into the functionality of mfx®. Even if this knowledge is certainly not necessary in everyday model railway life, it does help to explain and understand certain processes.

## 1.2 Document history

version	Date	author	Changes
1.0	20090201	S. Krauss	Document newly created, all results of the team R. Müller, MF Schwarzer, S. Krauß, K. Schwichtenberg and E. Stegmaier summarized CVs broken down in detail
1.1	20090301	S. Krauss	Minor corrections
2.0	20140613	S. Krauss	Revision of the entire document, incorporation of the information from various craft projects CV meaning further analyzed and missing descriptions added or existing ones corrected mfx+® analyzed and described
2.1	20140629	S. Krauss	Bugs corrected, some new findings (especially mfx+®) incorporated
2.2	20161105	S. Krauss	mfx® Command for 32 functions added, MM coding corrected
2.3	201706xx	S. Krauss	Explanations Configuration Space expanded, quick login described

## 1.3 Conventions

Some details have not yet been clarified or investigated experimentally. These open questions are generally not explicitly noted in this document. Where parts that are not yet known in detail are described, two cases occur, which are marked as follows:

**reserved:** As *reserved* the parts are described which are logically possible and sensible full but could not be observed experimentally (eg certain command markings). These possibilities may be defined later by the system developers or have already been defined but not yet used.

**unknown:** As *unknown* the parts are described that were observed but for which an explanation is still missing.

Since the names and designations used by the developers are unknown, an attempt was made to find the most meaningful designations possible. In cases of doubt, however, the designations used can also be misleading, namely when a designation implicitly associates facts that are not correct. Caution is therefore advised when deriving information from the designations.

The terms decoder and locomotive are used synonymously in this document, because so far mfx®capable decoders have only been available as locomotive decoders. However, the explanations also apply to all other conceivable mfx®Decoder, e.g. function decoder.

## 1.4 Abbreviations

<b>CA</b>	Configuration Array	Substructure of the configuration variables.	see 6.1
<b>CV</b>	Configuration variable (Configuration Variables)	Storage location for configuration settings in the decoders.	see 6.1
<b>DCC</b>	Digital Command Control	Alternative rail format.	see 2.1
<b>MM</b>	Marklin®Motorola®	Alternative rail format.	see 2.1
<b>SID</b>	Rail address (RailIdentifier)	For addressing mfx®Addresses used by decoders via the rail signal.	see 2.2.4
VAT number	Participant recognition (Unique Identifier)	Unique 32-bit serial number for central units, decoders and other digital components such as boosters; assigned by Märklin.	cf. CS2 CANProto coll Description exercise

## 1.5 Property rights

The processes described may be subject to third-party protection rights, in particular copyright and patent rights. The patent application EP

1 555 054 A1, which obviously relates to certain applications of the mfx®rail format, especially the automatic registration. Märklin has other patents and patent applications from subsequent years that deal with various aspects of control via mfx®and mfx+®. However, some of these seem to be further developments that have not yet been implemented.

Therefore, no statement can be made about the free usability of the procedures described here.

## 1.6 References and literature

The majority of the information was obtained by the team through their own observations, experiments and conclusions. Some clues can also be found in the following places:

**Rainer Müller's website:** originally <http://www.mue473.de> , for a domain

Moving now under <http://www.alicedsl.net/mue473/> This detailed description was partly created in parallel with the work of the team and was published before Rainer Müller joined the team.

**CS2\_canprotokoll\_10.pdf**(26.9.2008): From Märklin® on <http://www.maerklin.de> published clear description of the communication protocol (CAN/Ethernet) of the Märklin® CentralStation 2.

**cs2CANProtokoll2\_0.pdf**(7.2.2012): Second, extended version of the CAN protocol Description of Märklin®, also on <http://www.maerklin.de> published.

**Patent application EP 1 555 054 A1:** By ESU® Patent application filed in 2004, which obviously familiar with the procedures of mfx®. The patent application was abandoned and no valid patent was created.

**Patent DE 10 2011 003 472 A1 or patent application EP 2 481 458 A1:** Märklin Patent, which deals with the procedures for calculating resource consumption etc. in mfx+. Both documents are essentially the same in terms of content.

**Patent DE 10 2009 058 296 A1 or patent application EP 2 329 870 B1:** Märklin be writes a possible extension of the handling of addresses in mfx®Format. Both documents are essentially the same in content.

**Patent application EP000002431083A1:** Patent application by Märklin, which deals with the return message in mfx®Signal busy.

**Brochure "Let's talk about ... Märklin®systems":** When introducing mfx®2004 by ESU® information sheet published, contains a section with technical details about mfx®.

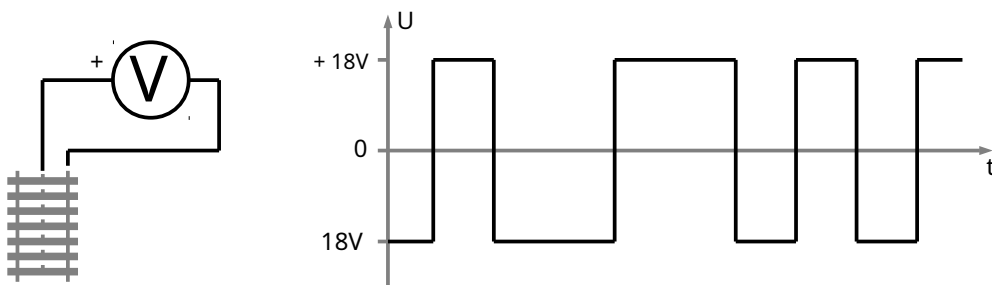
The patent documents can be obtained via the patent search at the German Patent and Trademark Office (on the Internet via <http://depatisnet.dpma.de/> to reach).

## 2 Signal and coding

### 2.1 Transmission principle

The rail signal consists of a square wave alternating voltage of approximately  $\pm 18$  V. In the vehicles and stationary decoders, the supply voltage for the decoder electronics and the connected consumers is obtained after rectification.

The polarity changes serve to transmit information. The same principle is also used in other digital systems such as the digital system in the Märklin®Motorola®Format (MMFormat, previously from Märklin®used format and thus in a way predecessor of mfx®) or DCC (Digital Command Control, a standardized format that is particularly popular in two-wire model railway systems). From a physical point of view, the rail signal is therefore compatible with these systems.



The rail signal only changes polarity, there are no gaps (no voltage periods). However, the polarity can remain the same for some time. Such pauses are used to separate Motorola® Commands and mfx®Commands and feedback. However, these pauses are less than 100 ms (maximum value is not exactly known).

Longer periods with negative potential (limit not exactly known, probably  $> 100$  ms) on the center conductor are understood as a braking section as in the previous digital system. In this case, the decoder regulates according to the braking curve until it comes to a standstill and starts moving, taking the acceleration curve into account, when information is received again. The braking section function does not have to be implemented and can be switched off.

The Märklin®Some decoders also recognize braking distances at positive potential, but the original definition requires negative potential. So that the principle can also be used on two-wire systems where there is no clear assignment of the two connections, these decoders do not differentiate between positive and negative potential.

The rail signal also has a return channel, via which mfx®Decoders can report information back to the control center. This happens in special pauses during which a voltage of constant polarity or with short polarity changes is applied to the track for synchronization. The RDS modulation method is used.®(RadioDataSystem, a system for transmitting additional digital information such as the station name in the radio broadcast). The feedback is generated by a consumer in the decoder. This produces short current pulses at a certain frequency, which can be coupled out again at the central station. The consumer in the decoder is stimulated by a square wave signal.

The specification of the current to be generated is not exactly known, but current decoders generate current pulses of around 140 mA. A transistor controlled by a processor output short-circuits the two signal lines via a 100 Ohm resistor. Attention: Depending on the decoder hardware, the transistor and resistor cannot be loaded to any extent; a large number of feedback signals in a short period of time can therefore damage the decoder.

For comparison with DCC: Here an acknowledgement pulse is defined as feedback, which also works with current pulses, but which are usually generated by the motor connected to the decoder. Here the current must be at least 60 mA; at the central unit (programming track!) it is limited to around 250 mA by a resistor. The pulse is not modulated and must be around 6 ms long.

## 2.2 Control channel

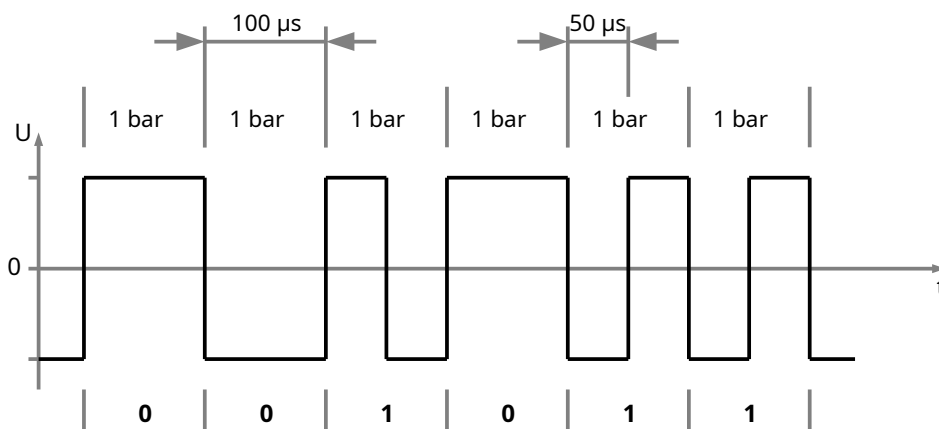
The control channel is the transmission of information from the control center to the decoder via the rail.

### 2.2.1 Bit coding and synchronization

The bits are biphase mark encoded. The clock signal is therefore included in the data stream and all bits have a fixed length. A transmission clock of 10 kBaud is used, so one bit is 100  $\mu$ s long.

The bit coding can be described by the following rules:

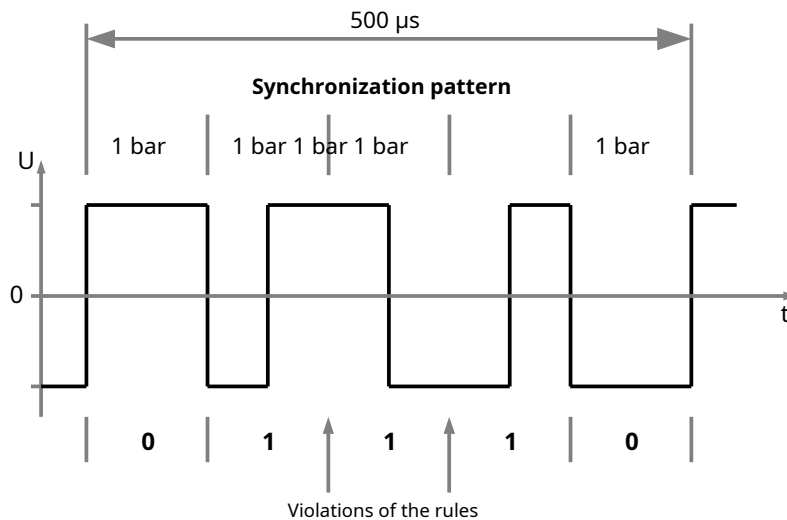
- Each clock boundary is marked by a polarity change.
- A one is represented by a polarity change in the middle of the clock cell.
- A zero is represented by a clock cell without polarity change.



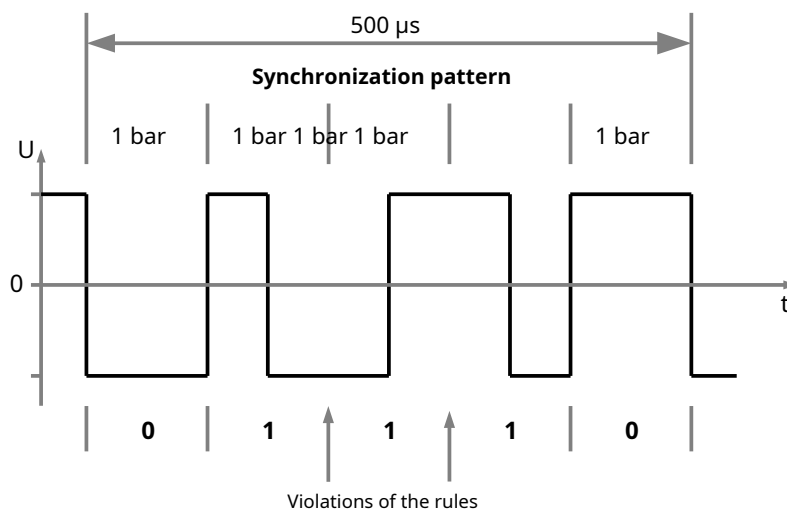
To synchronize the transmission and to mark the beginning and end of the frame, a synchronization pattern (this can be seen as a preamble for the frames) is used that deliberately violates the coding rules. The synchronization pattern is therefore easy to recognize regardless of the data sent.

The 500  $\mu$ s long bit sequence 01110 is used as the synchronization pattern, whereby the polarity change between the 2nd and 3rd and the 3rd and 4th bit is omitted. At these two

The obligatory polarity change that indicates the end of a clock cell is omitted. This is a violation of the biphase mark rules.



Depending on the level at the beginning of the synchronization pattern, this can of course also occur with opposite levels. The direction of the polarity change is not particularly important.



## 2.2.2 Stuffing

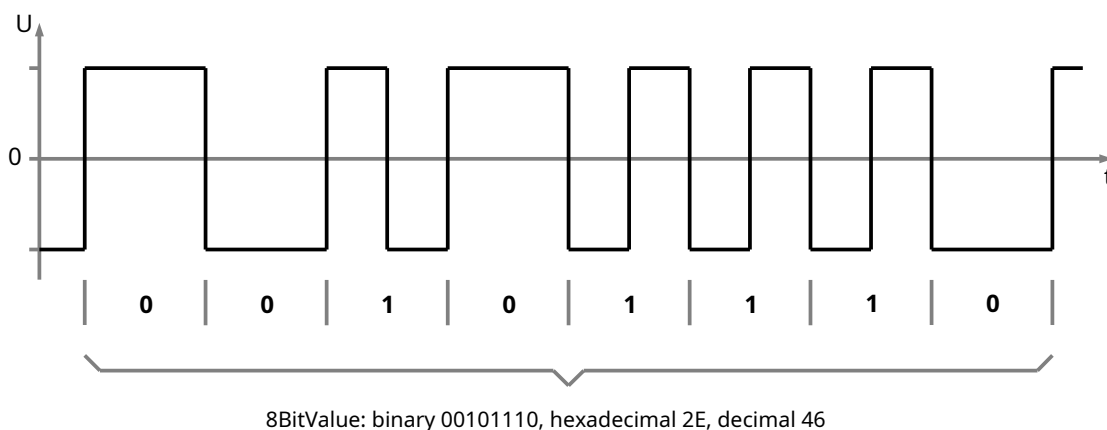
Stuffing means inserting bits into the data stream to avoid certain bit patterns in the normal data stream. This is used because, for example, these bit patterns are used for synchronization or because confusion with other protocols is possible. The synchronization of the data and frames of the mfx®The rail signal is independent of the transmitted data due to the special synchronization pattern, so certain bit sequences in the data cannot be confused with the synchronization pattern. Bit stuffing would therefore not actually be necessary.

Nevertheless, bit stuffing is used, which prevents more than 9 one-bits from being transmitted in direct succession. The one-bits would be recognized by a DCC decoder as one-bits, even if the specified lengths only overlap very slightly. A long series of one-bits is considered a synchronization signal for DCC frames in the DCC system. This is why in the mfx®A stuff bit has been inserted into the data stream to prevent such constellations from occurring and thus to maintain interoperability with the DCC system. This means that DCC and mfx can be used on the same track.®transmitted alternately without causing confusion in the decoder.

The stuff bit rule is: After 8 transmitted ones, a zero is always inserted. This is inserted after the checksum calculation in the central unit and removed again before the checksum test in the decoder. The synchronization pattern is not taken into account, so a stuff bit is never inserted in the synchronization pattern. If all 8 bits at the end of the frame are one, a stuff bit zero is appended to the end of the frame. This means that the stuff bit can also be removed at the end of the frame without special treatment.

### 2.2.3 Bit order

The bits are transmitted in big endian order, meaning the most significant bit of a value is sent first.



This order applies to every single value, regardless of its length. The Big Endian bit order also applies to the bits of the individual bytes in the response.

The response uses the Little Endian byte order. This means that byte 0 is sent first, then byte 1, and so on.

Note: When storing configuration variables, larger individual values are sometimes created from several configuration variables, i.e. several bytes. The Big Endian byte order is usually used here (see Chapter 6).

### 2.2.4 Structure of the data frames

Data frames are started by the synchronization pattern and always end with a synchronization pattern. However, only one synchronization pattern needs to be output between two data frames. However, several synchronization patterns can also be output.

synchronization patterns follow one another or the last polarity change is omitted in the last synchronization pattern before a pause.

The data frames are structured as follows:

Address Commands Checksum

The frame can contain a command with parameters, but it can also contain no command at all or several commands in a row. The following patterns are also valid frames:

Address Command1 Checksum

Address Checksum

Address Command1 Command2 Checksum

Two commands only occur in the combination of travel command plus functional state. Presumably only this combination is permitted. In particular, more complex commands with feedback do not work in combination with other commands.

A rail address, the so-called SID, is used to address the decoders. It is dynamically assigned to the decoders during registration (addresses starting from 1, whereby the central unit is free to decide which addresses it assigns). The address is 14 bits long (= 16383 addresses + 0 as broadcast address). However, not all address bits are always transmitted in order to save transmission time; the upper bits are then implicitly 0. The format can be clearly identified by the first bits.

The address formats in different lengths (LSB on the far right):

10 AAAAAA	7 bit address
110 AAAAAAAA	9 bit address
1110 AAAAAAAAAA	11 bit address
1111 AAAAAAAAAAAAAA	14 bit address

The address 0 is the broadcast address and addresses all decoders, especially decoders that have not yet been registered.

The checksum is always 8 bits long and forms the last 8 bits of the frame:

CCCCCCCC

The commands always begin with a 3-bit command identifier. This is followed by the parameter bits of the command.

CCC PPPP...

The length of the command with parameters is determined when reading from left to right from the command and possibly the parameter values. The length is not necessarily fixed (but usually is).

Some commands use a subcommand identifier of an additional 3 bits in the first position of the parameters (extended commands).

CCC UUU PPPP...

However, the structure is basically the same; the subcommand bits could be considered as parameters.

### 2.2.5 Checksum

The checksum is a CRC code that is created over all bits of the frame, but only after any existing stuff bits have already been deleted. The CRC code is created bit by bit, since the frame length is not necessarily divisible by 8.

CRCAlgorithm: simple, bitwise CRCCode with XOR

CRCPolynomial:  $x_8 + x_2 + x + 1$  (= 100000111) Start value: 127  
(= 01111111)

Final value: 0

When reading, all bits of the frame without stuff bits are pushed into the CRC register. The register is first initialized with the start value. At the end of the frame, i.e. when the 8 bits of the checksum have been processed by the CRC register, the CRC register must contain 0.

There are many implementations for the CRC calculation. The following CCode shows an example of a way to check the checksum (datacontains the data bits in ascending order, NUMBERis the number of data bits):

```
uint8_t data[NUMBER];          // Value only 0 or 1
uint16_t crc;                  // not just 8 bits to get carry

crc = 0x007F;
for (int i = 0; i < COUNT; i++) {

    crc = (crc << 1) + data[i]; if ((crc &
    0x0100) > 0)
        src = (src & 0x00FF) ^ 0x07;
}
```

In this algorithm implementation, the polynomial is represented by 0x07, the 9th bit position is omitted. `crccat` the end is 0, then the test was successful.

The CRC calculation can be carried out using the same code. The data bits must also contain the 8 bits that will later contain the CRC. These bits must initially be set to 0.

### 2.2.6 Principle of feedback

The decoder is able to send information to the central unit. This is called feedback. The data transfer between the decoder and the central unit is always initiated and controlled by the central unit. Certain commands from the central unit expect a feedback. The decoder therefore only ever responds to these commands.

After each command that expects a response, a response window is sent, which gives the decoder time to place the response on the track. The decoder only sends data in this window.

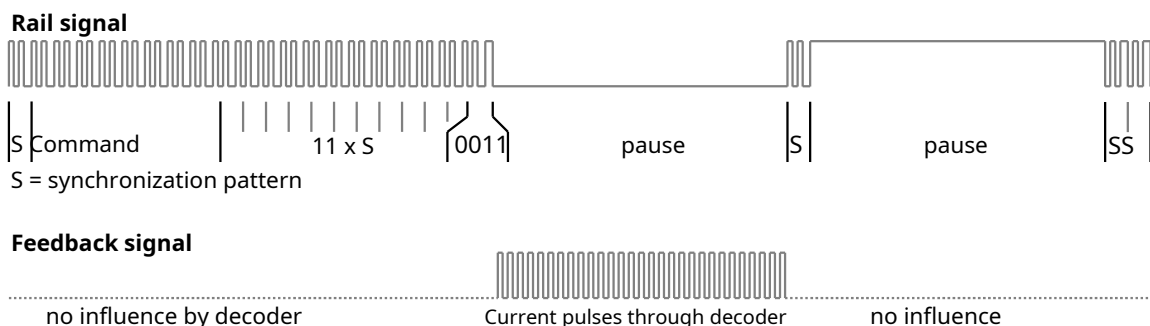
There are two types of feedback: a simple 1-bit feedback (like a yes/no response) and the data feedback, where several bytes can be transmitted. Which feedback is expected depends on the command. The feedback window differs slightly between the two types of feedback.

The behavior of the control channel, i.e. the signal from the central unit to the decoder, for the 1-bit feedback is described in the following section 2.2.7, and for the data feedback in 2.2.8. The return channel, i.e. the signal from the decoder to the central unit, is the subject of subsection 2.3.

### 2.2.7 Feedback window 1BitFeedback

The feedback window for a 1-bit feedback consists of the following sequence, which follows directly after a corresponding command:

- 11 or 11.5 synchronization pattern
- Bit sequence 0011
- After the last edge, another edge change can occur after a very short time (approx. 2 – 3  $\mu$ s; this is inserted by the control center in order to achieve a certain polarity of the pause)
- Pause with negative/positive potential of 6.4 ms
- 1 or 1.5 synchronization patterns
- Pause with equal or inverse potential of 6.4 ms
- 2 synchronization patterns as conclusion; the last synchronization pattern may form the start of a new frame



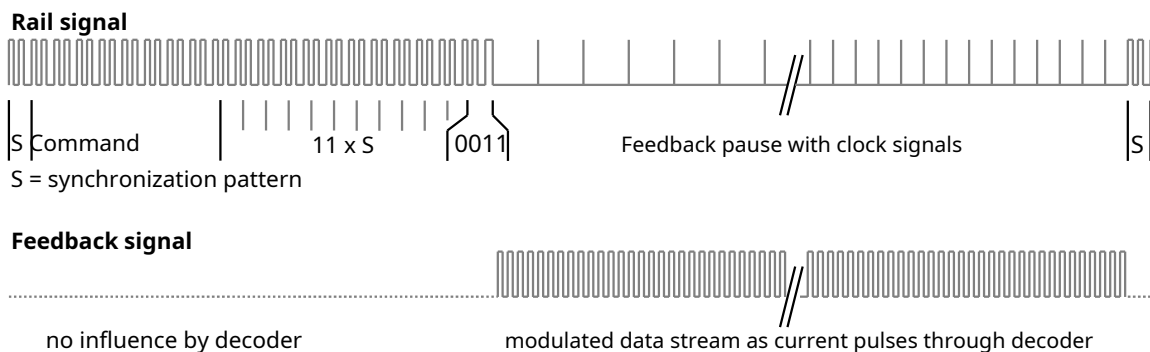
With simple feedback there is no synchronization with the RDS®Clock as with the data feedback. The feedback by creating the RDS®The carrier signal (generation of current pulses by the decoder, see 2.3) is generated during the first pause. In the second pause, the decoder no longer generates the signal. The second pause could serve as a reference for error detection (if a carrier is measured here, an error has occurred, e.g. due to a scattered signal). It also ensures that a pure mfx®The signal is symmetrical except for the data feedback, i.e. the positive and negative voltage components are on average the same length. The signal then contains no DC voltage component, i.e. it is DC-free. However, this does not apply to the data feedback.

The bit sequence 0011, which indicates the start of the feedback window, was chosen because it enables a clear distinction between the start of a command (always starts with 1 and then the address bits) and a synchronization pattern (starts with 01, with the 1 missing the last edge). This makes it easier to decode the data stream in the decoder, which can then recognize the start of the feedback window synchronously with the data stream.

### 2.2.8 Feedback window Data feedback

The feedback window for a data feedback consists of the following sequence, which follows directly after a corresponding command:

- 11 or 11.5 synchronization pattern
- Bit sequence 0011
- After the last edge, another edge change can occur after a very short time (approx. 2 – 3  $\mu$ s; this is inserted by the control center in order to achieve a certain polarity of the pause)
- Pause with predominantly negative/positive potential, length depends on the expected bits, short clock pulses are generated during the pause
- 1 synchronization pattern as conclusion; may belong to the following data frame as start synchronization pattern



With the short pulses during the feedback pause, the central unit tells the decoder the position of the possible phase changes for the clock and data signals (clock and data are encoded by phase changes, see 2.3). A possible phase change is indicated by the central unit with a short, 25  $\mu$ s long pulse (positive or negative, depending on the potential of the pause signal).

The decoder can orient itself on these signals, but it does not have to. New Märklin® Decoders probably do without this. However, for compatibility reasons, the central units still send the 25 $\mu$ s pulses.

First, the central unit sends pulses at intervals of approximately 912  $\mu$ s. The decoder modulates the clock signal at these points by changing the phase. During this time, no user data is sent; the data sent corresponds to a sequence of ones. 23 clock pulses are sent without data. This gives the RDS®Receiver in the control center has the opportunity to synchronize to the signal. If this works in exceptional cases

If it does not work once, for example because the decoder signal happens to start very unfavorably, the response from the control center cannot be read and the query must be repeated.

Then the actual data transmission begins. To do this, the central unit also places a 25  $\mu$ s pulse in the middle of the clock pulses. For a zero, the decoder would have to insert another phase change at this point (see 2.3). The pulses from the central unit now come at intervals of 456  $\mu$ s.

The number depends on the amount of data expected (3 for start pattern 010 + number of data bits + 8 for checksum + 4 extra at the end). When transmitting one data byte, there are 23 clock cycles (= 46 pulses at intervals of 456  $\mu$ s), and for 2 data bytes, there are 31 clock cycles. The entire pause with the short clock signals is therefore around 42 ms long for a 1-byte response, and just over 49 ms for 2 bytes.

Then the pause ends, the final synchronization pattern is sent, the decoder no longer sends feedback signals and switches off the carrier signal.

Note on timing: The interval between the start of the 25 $\mu$ s pulses is approximately 912  $\mu$ s or 456  $\mu$ s. However, the intervals are not very precise. At the beginning they tend to be longer (approx. 930  $\mu$ s), later they tend to be less (approx. 880  $\mu$ s). However, very precise timing is not important here.

### 2.2.9 Embedding control commands in MM format

In addition to the mfx, the data stream of the rail signal can also be embedded with MM control commands. For this purpose, the data stream of the mfx®Frames are completed by at least two synchronization patterns, such that the last synchronization pattern ends with a negative potential. This is followed by a pause with a negative potential, then an MM command and then another pause with a negative potential. Then the transmission of mfx begins again.®Frames with a synchronization pattern.

- Pause with negative potential, about 6.25 ms
- an MM command consisting of 9 double bits (= 18 bits)
- Pause with negative potential, about 1.57 ms
- Repeating the MM command
- Pause with negative potential, about 6.18 ms

The repetition of the MM command and the pause between the MM command and the repetition are a necessary part of the MM definition (the command is only accepted if the command is repeated directly after an appropriate pause).

It is also possible to send several different MM commands one after the other before returning to the mfx®data stream is switched. In addition, commands of other data formats could also be embedded in the same way, e.g. DCC commands.

## 2.3 Return channel

The return channel is the transmission of information from the decoder to the control center via the rail.

### 2.3.1 Modulation

Basically, the return channel is like an RDS®Signal is coded (bit coding and modulation). This makes it possible to use a commercially available RDS®DecoderChip to be used. This is also implemented in this way.

The carrier frequency does not exactly correspond to that for RDS®defined carrier frequency. A slightly lower frequency is used, which, in contrast to the RDS®Carrier frequency can be derived from a quartz with an "even" frequency value (e.g. 4 MHz). The carrier frequency is 52.63 kHz (results from 4 MHz divided by 76, instead of 4.332 MHz divided by 76 = 57 kHz for RDS®).

As described in 2.2.6, there are two types of feedback: a simple 1-bit feedback and the data feedback. Both use the RDS®DecoderChip, but in a different way. Only the actual data feedback uses the actual RDS®Transmission technology.

### 2.3.2 Coding of the 1-bit feedback

The simple 1-bit feedback is coded by the fact whether there is a response from the decoder at all. If the decoder sends a response, this is counted as one ("yes"). If the central unit does not detect any response, this means zero ("no"). The special thing about it: This type of response can be sent by several decoders at the same time, the responses are ORed. The response is therefore recognized as existing if at least one decoder sends a response.

As feedback, the decoder sends the RDS®carrier signal, thus generating a square signal with a frequency of 52.63 kHz.

Technically, it is an ASK signal (Amplitude Shift Keying). The carrier signal is amplitude modulated with the data stream. There are only two states to be transmitted: Bit = 0 means the amplitude of the carrier signal is 0, with Bit = 1 the carrier signal is sent with full amplitude without phase change.

The headquarters takes advantage of the fact that the RDS®DecoderChip reports the presence of a carrier signal independently of transmitted data.

### 2.3.3 Coding for data feedback

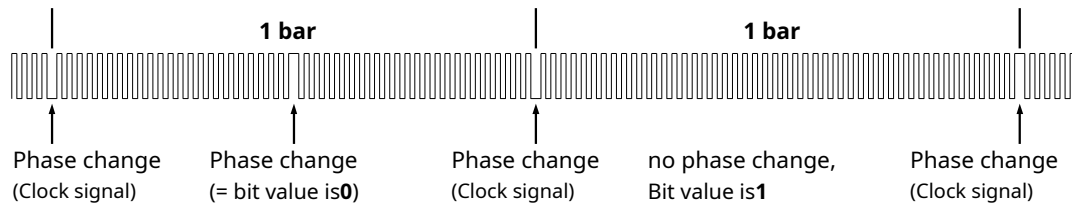
When data is returned, a stream of data bits is sent, which the central unit interprets. 1, 2, 4 or 8 data bytes and a checksum byte are transmitted. In tests, only 1, 2 and 4 bytes could be transmitted. However, information in the CAN documentation for the Märklin Central Station 2 and the command structure lead to the conclusion that 8 bytes are also possible.

The feedback may only be provided by one decoder at a time, which is requested to do so by the corresponding command.

To encode clock and data, the RDS®defined biphase modulation is used. With each clock signal, the phase position of the carrier signal changes by 180°. One clock lasts 912 µs, so it is sent at 1096.5 baud (= carrier signal divided by 48). The bits are also coded by phase changes:

- **Bits = 1:**the phase position does not change within one cycle.
- **Bits = 0:**the phase of the carrier signal changes by 180° in the middle of a clock (i.e. 456 µs after the phase change of the clock).

The coding is very similar to the biphase mark coding used in rail signals.



Technically, this is PSK modulation (Phase Shift Keying). Information is transmitted by changing the phase position of the signal. In this case, the clock signal and the data signal are transmitted using biphase coding with two phase positions.

The data stream of the return channel transmits one bits in the idle state. The data frame is structured as follows:

... 11111 010 Byte1 (Byte2 ... Byte8) Checksum 111...

010 is the preamble, it marks the start of the data bytes. Then 1 to 8 data bytes follow without any further separators. At the end there is always an 8-bit checksum. Bit stuffing or similar does not occur.

The specified byte order is used, starting with byte 1. The response normally contains the contents of consecutive configuration variables. Byte 1 is the configuration variable at the requested address, byte 2 is the CV with an address increased by one, etc.

### 2.3.4 Checksum for data feedback

The last byte of the data read back forms a checksum that has certain similarities to a CRC code. This checksum is formed from all the data bytes of the response and then appended.

The following algorithm (in C) can be used to calculate the checksum from the data bytes (data contains the data bytes, NUMBER is the number of data bytes, i.e. 1, 2, 4 or 8):

```
uint8_t data[NUMBER];
uint16_t checksum;

checksum = 0x00FF;
for (int i = 0; i < COUNT; i++) {

    checksum = checksum ^ (checksum << 1) ^ (checksum << 2); checksum =
    checksum ^ data[i];
    if (checksum & 0x0100)
        checksum = checksum ^ 0x0107; if
    (checksum & 0x0200)
        checksum = checksum ^ 0x020E;
}
```

The code does not include loading the data, only the algorithm is shown. An actual implementation can of course differ from this.

When responding with one byte, the checksum check is simplified to:

```
if ((data byte ^ checksum) == 0xF3)
    OK
```

## 3 Commands

This chapter describes the various commands that can be used in an mfx®Frame can be sent from the central unit to the decoders via the track signal.

For all commands, only the data bits of the command identifier and the command parameters are specified here. For a complete mfx®For this purpose, the frame must be supplemented with a synchronization pattern, address (SID) and checksum according to the scheme described in the previous chapter.

### 3.1 Commands

#### 3.1.1 No command

Data frames without a command (i.e. only address + checksum) expect a simple 1-bit response from the addressed decoder. This can be used to determine whether a decoder is currently accessible at the specified address (SID).

This command has not yet been observed in any central unit, but has been experimentally used in the first decoders with mfx®It is not known whether all decoders behave accordingly.

#### 3.1.2 Command 000: Drive (short) 000

R SSS

This command enables extremely short driving commands, but is limited to certain driving levels. It is used in particular when the locomotive is stationary (i.e. at driving level 0). The direction of travel and the speed of the locomotive are specified.

R = 0: forward, R = 1: backward

S = speed level calculated from  $S * 16$  (i.e. the 8 speed levels: stop, 16, 32, 48, 64, 80, 96, 112); bits 4, 5 and 6 of the speed level are transmitted, bits 0, 1, 2 and 3 are implicitly 0.

mfx®uses 127 speed levels. Speed level 1 causes an emergency stop regardless of the direction; the locomotive stops immediately without taking the set braking delay into account. However, speed level 1 cannot be transmitted with this command.

#### 3.1.3 Command 001: Drive 001

R SSSSSS

This command specifies the direction of travel and speed in full resolution, so all possible speed levels can be transmitted.

R = 0: forward, R = 1: backward

S = speed level (127 speed levels, speed level = 1 is emergency stop)

**3.1.4 Command 010: Functions (short) 010**

FFFF

This command specifies the state of the functions. The first 16 functions are implicitly controlled, but only the first 4 functions are specified explicitly. The other 12 functions are implicitly "off".

FFFF = F3 F2 F1 F0; F4 – F15 are "off" F = 0:

function "off", F = 1: function "on"

**3.1.5 Command 011: Functions (extended) 011**

0 FFFFFFFF

011 1 FFFFFFFFFFFFFFFF

This command also transmits the status of the functions of the locomotive or decoder. There are two variants that explicitly specify either the first 8 or the first 16 functions.

FFFFFFF = F7 F6 F5 F4 F3 F2 F1 F0; F8 – F15 are "off"

FFFFFFFFFFFFFFF = F15 – F0

F = 0: function "off", F = 1: function "on"

**3.1.6 Command 100: Functions (individual control) 100**

NNNNNNN 0 F

This command can be used to control the state of exactly one function. The states of all other functions are not affected. The command must therefore be sent for all existing functions in order to transmit the complete functional state.

Up to 128 functions, i.e. F0 – F127, can be switched. However, central units and decoders currently only support a maximum of 16 or 32 functions.

NNNNNNN = Function number F0 – F127 F

= 0: Function "off", F = 1: Function "on"

**3.1.7 Command 101: reserved****3.1.8 Command 110: reserved****3.1.9 Command 111: Configuration**

111 CCC ....

The configuration of the decoder (reading and writing) is carried out via the extended command 111. The action is defined by a subcommand CCC



This command is only executed after registration; once registered, the track address assigned by a specific control unit is saved.

### 3.2.5 Command 111 100: Existence query (Ping, mfx®Verify) 111

100 UUU

- + simple 1-bit feedback

This command checks whether a registered decoder exists under this SID with the stored UID. The decoder acknowledges this with a simple response if the decoder has the specified UID U.

The decoder remembers the track address assigned to it. With this command, it can recognize that the assignment of track address and UID made by the control center is valid. The decoder registers again if it is addressed using this track address, but the UID is incorrect; no feedback is given. If the track address and UID are correct, the decoder acknowledges this via the 1-bit feedback. This makes it clear to the control center that the corresponding decoder can be addressed.

### 3.2.6 Command 111 101: Headquarters

[illegible]

The control center sends this command at regular intervals with the broadcast address 0 and thus displays its UID U and the new registration counter Z.

The decoder remembers the track address, the UID of the central unit and the status of the new registration counter. This command tells it the UID of the current central unit and the status of the new registration counter. If one of the stored values differs from the two values provided by the central unit, the decoder immediately switches off all functions and registers again (see behavior when switching on).

The new registration counter is increased by one each time the central unit is reset (via a corresponding menu item such as "Reset to factory settings" or similar, not a reset in the electrical sense by switching the central unit off and on again) and each time a decoder is deregistered. This change then forces all decoders that have not actively noticed the increase to register again when they first contact the central unit. However, if the counter is increased while the track signal is being received continuously, the decoder simply saves the new counter reading. In this case, a new registration does not take place. The decoder recognizes this because it has already received a "central unit" command before the change.

### 3.2.7 Command 111 110: reserved

### 3.2.8 Command 111 111: reserved

## 4 Protocol

This chapter describes processes that occur in operations. The description is based primarily on observation of available systems. There may be specific requirements and rules for the processes that are not known.

### 4.1 Hibernation

If the operator does not change anything, the system is in a sleep state in which certain information is transmitted cyclically:

- Command "Central": With this, the central office announces its UID and the new registration counter.
- DecoderSearch: Gives decoders the opportunity to register.
- Decoder query for each registered mfx®Decoder: Through the corresponding feedback, the control center knows which registered decoders can be addressed with feedback.
- Driving and switching commands for each registered mfx®Decoder: This refreshes the status of the decoder.
- If MMDecoders are also controlled by the central unit, an MM signal is also generated and repeated.

The temporal behavior depends strongly on the respective situation, especially the number of registered mfx®and MMDecoder dependent.

Example: The following conditions were determined for a system with three registered mfx®Locomotives and one MMLok observed (roughly):

- The central unit sets its UID/new registration counters approximately every 150 ms (Märklin®Mobile Station 1<sub>1</sub>) or 500 ms (Märklin®Central Station 1<sub>2</sub>) available.
- Decoder search and decoder query alternate, with the decoder query being carried out cyclically across all registered locomotives. The sequence "Search, query SID 1, search, query SID 2, search, query SID 3" can be observed here. The interval between two commands is approximately 100 ms.
- The decoder states are transmitted more frequently, the interval between two commands for the same decoder is about 50 – 80 ms. Decoder states for mfx®and MMDecoder are treated in approximately the same way and sent cyclically (i.e. mfx®SID 1, 2, 3, then MM, then SID 1, 2, 3 again, etc.).

---

1 About the first generation of Märklin®To distinguish the Mobile Station from the successor generation introduced in 2008, this document refers to the Mobile Station 1. Officially, this marking or numbering is used by Märklin®not used. As with the Märklin®Mobile Station 1 is also the first generation of Märklin®Central Station is meant and therefore marked with an officially unused number 1.

## 4.2 Switching on: Decoder receives power

As soon as the decoder receives power (even after short power interruptions) it behaves as follows:

- Depending on the setting of the special options in the decoder, it initially continues with the last settings for motor and functions.
- It waits for 2 correct "Central" commands (the UID of the central unit and the new registration counter must match the stored values). Only when it is certain that the decoder and the central unit already know each other correctly does the decoder accept further commands and also respond to the decoder request (ping) from the central unit; before that, it ignores it.

Originally, it was intended that a decoder would accept commands addressed to its SID after it was switched on, even if it could not yet check whether these came from the known central unit. After receiving an incorrect central unit identification, the locomotive would then have stopped and registered again. This was intended to prevent any interruptions in the control of the locomotives during brief power interruptions. However, the safer method described above was implemented in the first decoders.

- The decoder can also be put into operating mode by assigning an address (assigning a SID).
- If a different central UID, a change in the registration counter or an incorrect address assignment is detected in the decoder request, the decoder registers with the central unit again.

If the decoder was not yet registered when it was switched off or was in the process of registering, the decoder remembers this. In this case, the decoder registers again immediately when it receives power again.

## 4.3 Registration

At regular intervals, the central unit sends the command DecoderSearch with C=0, U=0. All decoders that have not yet registered then respond, i.e. all decoders that are ready to register. Decoders are ready to register through the following events:

- The decoder has not yet registered or has not registered again after being ready to register. In this case, no central UID is stored in the decoder.
- A "Central" command is received where either the UID of the central unit or the new registration counter do not match the stored values.

If the new registration counter is changed while the decoder is receiving the data (i.e. the decoder has seen a "central" command with the old and then the new registration counter value), it only saves the new value of the counter but does not register again.

- A decoder query is received for the own track address (SID) in which the own UID is not specified.

The central unit uses this after deleting a locomotive in the central unit to prompt it to immediately re-register if the locomotive is currently on the track (in this case it would have recognised the increase in the new registration counter and saved it without re-registering). After deleting, the central unit sends a decoder query with UID 0 to the freed-up SID.

- The decoder will not receive any decoder queries for its own decoder for approximately 5 to 10 minutes.

As soon as (at least) one decoder responds to the first search request, the UID of the decoder is determined via a sequence of search request commands.

This process works as follows: First, all decoders whose highest bit is 0 are searched for with C=1, U=000... If one responds, the next bit (C=2) is searched for. If no one responds, the query continues with C=1, U=100... to a 1-bit. All 32 bits of the UID are queried in this way. If several decoders try to register at the same time, only the decoder with the smallest ID is found. The decoder that is not recognized registers with the next query.

After the decoder UID has been determined, the central unit assigns a 14-bit rail address (SID) to the decoder using the "Assign rail address" command. This command is sent to broadcast address 0, since the decoder does not yet know the SID. The decoder then goes into operating mode with the relevant UID and can then be addressed using this SID.

The central unit now reads out a whole series of CVs. During this time, no decoder queries need to be sent, as the reading operations will provide feedback anyway.

The central unit begins to send the actual driving information and decoder statuses as soon as it has read out enough information about the decoder. It does not wait until the reading process is complete. In this way, a locomotive can be operated at the central unit before the entire registration process has been completed.

As soon as the central unit has read all the CVs it needs, the registration process is complete. From now on, the decoder status and the decoder query are sent regularly.

There are various strategies used by the central units for reading the decoder CVs after registration. The first generation of central units (Märklin®Central Station 1, Märklin®Mobile Station 1, ESU®ECoS®) reads all CVs of the decoder during registration. Depending on the decoder and the central unit, this takes roughly between 1 and 3 minutes. In the second central unit generation from Märklin®(Central Station 2, Mobile Station 2) only the parameters required for control are read out (in particular name and functions). Registration only takes about 10 to 15 seconds. However, the remaining CVs must be read out in a corresponding amount of time if the user wants to adjust parameters and calls up the corresponding menu.

In addition, reading can be done at different speeds on different decoders (see 4.9 Configuration), which can be exploited by the central unit.

## 4.4 Log out

Once a locomotive has been automatically registered, it remains stored in the control center until it is manually deleted from the control center. As long as an mfx®Decoder is registered, it is addressed by the central unit and remains in the refresh cycle of the central unit.

With the command "existence query" a control center can detect which registered locomotive is actually on the track. Originally it was intended that no commands would be sent to mfx®to send locomotives that cannot be reached. However, when checking whether the locomotive is in a section operated by a booster that does not have feedback capability, the control center cannot distinguish whether the locomotive is actually not on the track. This option is therefore not used.

Therefore, an mfx®The decoder registers automatically with the central unit, but it must be deregistered and thus removed from the central unit's refresh cycle manually.

Each time you log off (= delete an mfx®If a locomotive is logged off in the control center, the control center increases the new registration counter by one. A locomotive logged off "in absence" recognizes that it must log on again (new registration counter does not match the stored value).

However, all mfx that are not on the track at the time of deregistration will also report®Decoder again. It is not clear to them which locomotive was deregistered.

However, locomotives on the track at the time of deregistration do not register again because they immediately notice the counter change and correct the stored counter value.

This would also apply to the locomotive that was just deregistered. It would therefore only be able to tell that it had been deleted if it did not receive any commands (in which case it would register again after a few minutes).

To ensure that the deleted locomotive no longer feels addressed and registers again if necessary, the central unit sends a decoder query with the (false) UID 0 to the deleted locomotive immediately before increasing the new registration counter.

## 4.5 Resetting the control unit

When resetting, the new registration counter is increased by one, just like when deleting a locomotive. However, the track signal is also interrupted for about 10 seconds. The increase in the new registration counter cannot therefore be actively detected by any decoder. All mfx®Decoders will therefore register again the next time the track signal is switched on.

## 4.6 Driving and operating

Every driving command (= change of direction or speed) is immediately transmitted in an mfx®Frame is sent to the decoder (once). Other status information such as the status of the functions is not transmitted. For example:

Driving (speed) or

Functions (functional status of all functions)

There is no need to wait for the cyclical transmission of the decoder state. However, the next time the complete decoder state is transmitted, the change will of course be taken into account.

Decoder status and driving commands are also sent to a registered mfx®Decoder sent if no response is received to the decoder existence requests, which are also sent cyclically (see 4.1 Idle state and 4.4 Logout).

## 4.7 Multiple traction

Some central units can be used to create and drive multiple traction units. However, this is a feature of the central unit and has no equivalent in the rail protocol. The central unit simply sends the same control information to all decoders belonging to the multiple traction unit, regardless of the protocol.

## 4.8 Emergency stop

A reduction in driving speed does not normally take effect immediately, but is achieved gradually in accordance with the braking deceleration. This also applies when the driving speed is set to zero or the direction of travel is changed. In the latter case, the speed is controlled in accordance with the braking deceleration until the vehicle comes to a standstill and then in accordance with the acceleration time to the set speed.

In an emergency, however, this is not enough; the locomotive should come to a standstill immediately. This is achieved by setting speed level 1. In this case, the locomotive brakes as quickly as possible until it comes to a standstill. The set direction of travel is irrelevant.

The Märklin®For safety reasons, Mobile Station 1 sends the command “speed level 0” four times in a row and also changes the direction of travel (the button for emergency stop is identical to the one for changing the direction of travel).

**A notice:**With MM decoders, changing the direction of travel from driving causes an emergency stop. With MM1 decoders, speed level 1 is used for changing the direction of travel. With MM2 decoders, there is another, separate bit for the direction of travel. Speed level 1 is only used for the emergency stop here, the direction of travel bit is decisive for the direction of travel.

## 4.9 Configure

The decoders are configured during operation. The corresponding mfx®Commands configuration variables (CV, see chapter 6) are read and written in the decoder.

At mfx®However, there is no clear assignment between CV address and meaning. Therefore, the central unit must first determine the data structure and thus the CVs responsible for certain tasks by reading certain variables. The central units therefore do not allow the programming of arbitrary CVs, but only allow the modification of properties that the central unit has determined in the CV structure of the decoder in question.

The control unit reads the current configuration of a decoder when logging in or when calling up the programming menu. Only the configuration options available and supported by the control unit are then displayed.

When reading, the values are transmitted via the return channel. Since this represents a thermal load for the decoder hardware, two CV reading processes must not follow one another as quickly as possible. The first CV readings from ESU®systems manufactured by Märklin have a time interval of approximately 120 ms between two readings. The® manufactured, newer decoders (such as Märklin®mLD and mSD) have improved hardware that does not rely on these breaks.

When writing the configuration, various processes were observed:

- Märklin®Mobile Station 1: For security reasons, writing is carried out using three identical write commands in direct succession. The exact CV is known through registration, but the written values are not read back. There is therefore no control as to whether the value was actually written successfully.  
CVSet(CV, Value)  
CVSet(CV, Value)  
CVSet(CV, Value)
- Märklin®Central Station 1: The Central Station 1 does not store the CV addresses of all the setting options of a decoder, but only the start of the configuration blocks (the structure is described in Chapter 6). Therefore, the corresponding CV number is first searched for in the respective block (reading index 00 of each CV number belonging to the block until the CV number sought is found). Then all occupied CVs (identified by CV number + index) are written under this CV number, not just the ones that have actually been changed. Finally, all written CVs are read again for checking. Only 1 byte is ever written and read.
- Märklin®Central/Mobile Station 2: Multi-byte commands are used (usually 4 byte blocks), after writing the value is read back for checking.

In the first systems, the existing multi-byte read and write commands were not used. In the newer systems such as the Märklin®However, Central/Mobil Station 2 makes use of this.

Note: The strategy of when and which CVs are read and written depends largely on the respective central software. It must therefore be expected that the exact procedures change depending on the software version.

## 4.10 Protocol switching

All previously known mfx®Decoders are multi-protocol capable. In addition to mfx, they can also be operated in MM format and in analogue mode (AC, i.e. with alternating current). The decoder switches automatically between the available operating modes.

The decoder decides in a few seconds which protocol it will listen to. The switchover can also take place during operation.

Basically, there are different strategies for how the protocols are recognized and which commands are responded to by which protocol. The following decoder procedures were observed, among others:

- First generation decoders and Märklin®mfx®Decoders use priorities:

The decoder uses a specific protocol if it detects valid data packets for this protocol on the bus. It is irrelevant whether these data packets are addressed to the decoder itself. However, the data packet must be valid in itself, for example, it must have the correct checksum.

If several protocols are detected on the track, the decoder selects the protocol used according to priorities: mfx®has the highest priority. The decoder never listens to several protocols at the same time, even if data packets from different protocols are sent to it with regard to the set addresses.

After about 3 seconds of driving with mfx®the decoder no longer switches to MM, even if it does not have a valid mfx®Frame is no longer recognized. After a short power interruption (approximately 50 ms is sufficient), a new decision is made as to which protocol is to be used.

- ESUDecoders version 4 do not use priorities, but listen to the available protocols simultaneously (at least to some combinations).

Dealing with different protocols is essentially a matter of decoder implementation and not specified in the protocols.

## 5 Extension mfx+®

### 5.1 Functional principle

Marklin® markets a group of functions introduced in 2013 under the name mfx+® or “game world”. At its core, mfx+ is based® that more realistic game functions depend on operating states that are stored in the decoder itself and updated depending on usage. mfx+® This affects various components of the digital system:

- Decoder

An mfx+® The decoder measures a number of operating parameters and uses them to derive operating states, which it continually recalculates and saves. For example, the locomotive measures the distance travelled using the engine speed or a fictitious energy consumption using the total engine power. These values are queried by the control center and used to represent a more realistic game operation. For example, a diesel locomotive must be refilled with fuel after a certain amount of use. When this is the case is calculated using a simulated fuel consumption in the locomotive.

The decoder has two tasks: Firstly, it automatically records the operating parameters and calculates the fictitious fuel reserves. Secondly, the locomotive automatically goes into a kind of emergency mode when the reserves are running low.

- Headquarters

Only the central unit makes the possibilities of the mfx+ decoder a more realistic process. To do this, it queries the status of the locomotives, in particular their current operating fluid levels, at regular intervals and offers the operator appropriate actions, such as refueling.

In addition, the possibilities of mfx+ include® Locomotives at Märklin® also the control via a graphically displayed driver's cab at the control center. However, this is a functionality of the control center and technically does not require mfx+® Decoder. The central unit's software simply restricts use to appropriately equipped locomotives.

- mfx®Rail format

The functions of mfx+® expand the mfx rail format® in the narrower sense, it is just a few configuration variables. These variables can be used to configure the decoder's internal recording of operating states and to query the current operating state. The actual possibilities of mfx+® are achieved by the central unit and the extended behavior of the decoders.

In Chapter 6, especially subsection 6.4.10, the mfx+® The processes and procedures for mfx+® concerning the decoder and its interaction with the central unit are described in this chapter.

## 5.2 Procedures and processes

### 5.2.1 Mode setting

An mfx+®Decoder has two modes that can be switched via a configuration variable:

- Standard mode: In this mode, the special features of mfx+®switched off.
- mfx+®mode

There is no distinction between different mfx+®Types of play such as "professional", "semi-professional" or "specialist". There is also no distinction between the types of traction (steam, diesel, electric). These are settings that are only made on the central unit and control the behavior of the central unit. They are not saved in the decoder.

The mode is used on every mfx+®The decoder automatically resets the registration process to "Standard". This ensures compatibility with central units that use mfx+®not know. A control center that supports the mfx+®If you want to use this mode, you must explicitly activate it after logging in.

There are two approaches to selecting the driver's cab and the equipment used (for example, an electric locomotive has no coal supply):

- mfx+®Decoder, old firmware version (no driver's cab is stored in the configuration area under "stock"): The control center recognizes the type of traction from the symbol stored for the Mobile Station. If you want to use a different driver's cab, you only need to change the Mobile Station symbol in the configuration menu of the control center. However, only the three standard driver's cabs for steam, diesel and electric locomotives can be used here.
- mfx+®Decoder with new firmware version: Here the driver's cab, from which the type of traction is derived, is explicitly stored in the decoder under a corresponding configuration variable (see chapter 6.4.10).

If the decoder is operated via a different data format (e.g. MM), the mfx+®Settings ignored (no resource consumption, no emergency operation, etc.).

### 5.2.2 Resource management

The mfx+®Decoder manages three types of operational inventories:

- Fuel A: Water for steam locomotives, diesel for diesel locomotives
- Fuel B: Coal for steam locomotives
- sand

When driving, the decoder "consumes" these supplies. The aim is to simulate behavior that is as realistic as possible. When driving under load, for example, more diesel is consumed than when simply rolling. The decoder therefore uses other measured values in addition to time for the calculation, such as the engine speed and the power requirement. How exactly the consumption is calculated is stored in the decoder's firmware and is not known. How

How quickly the individual resources are consumed can, however, be influenced via configuration variables.

The fictitious resources are only available in mfx+®mode. However, the last status remains stored in standard mode and is available after switching back to the mfx+®Mode is available again.

The consumption of resources is independent of which type of traction the decoder is simulating. There is no corresponding setting. The decoder always calculates the consumption for all three resources. What exactly these resources represent and whether they are displayed at all is up to the central unit.

For a resource that is not required (e.g. the coal supply in an electric locomotive), the consumption factor must be set to 0. The resource is then not taken into account.

#### 5.2.3 Emergency operation

If the supply of a resource falls below certain limits, the locomotive goes into emergency mode. This means that the driving speed is limited to a certain value. There are two emergency modes:

- Low level: The limit can be set for each resource. If the limit is exceeded for at least one resource, the locomotive limits its speed to an adjustable value (referred to as the "reserve level").
- Resource empty: If the current supply for a resource has reached 0, the speed is limited to a second adjustable value (called the "empty level").

The emergency mode, like the resource management, is only available in mfx+®Mode active.

An equipment with a factor of 0 is not taken into account and does not trigger an emergency operation.

#### 5.2.4 Operational recording

The mfx+®The decoder records the distance travelled (measured via speed and set gear ratio) and the travel time. The total distance and total travel time can be called up as a value via configuration variables. The recording (or more precisely: the increase in the value) takes place regardless of the mode.

#### 5.2.5 Interaction with the headquarters

The Märklin®Central Station 2/3 with the corresponding software version offers locomotives with mfx+®Decoders have extended functions. It recognizes an mfx+®Decoder when logging on to the additional mfx+®Configuration block.

The control center queries the current status of the three devices at regular intervals by reading the corresponding configuration variables. The query is essential for the function of the mfx+®Decoder is irrelevant. It only serves to implement the central unit functionality.

The following behavior of the Märklin was observed®Central Station 2: There is a 10 second interval between two queries. Only those locomotives are queried

which can be controlled by the user and are therefore visible on the display. The query is also omitted when the configuration screen is open and the locomotive is not visible. The query interval is maintained even when several locomotives are visible, and the individual locomotives are then queried less frequently.

Note: In order to be able to carry out the query as quickly as possible, four configuration variables are read out simultaneously using a read command. Since there are only three operating states, a fourth, unused variable is also read out. However, this has no meaning and is always 0.

When the operating supplies are replenished ("fueling"), the control center writes the new value into the corresponding configuration variables (current status of the operating supplies).

After changing values in the CV configuration menu or after the refueling process, the control center reads the entire configuration of the resource stock again (i.e. current value, maximum value, factor, etc.). It uses these values to show the corresponding converted resource stocks in the display.

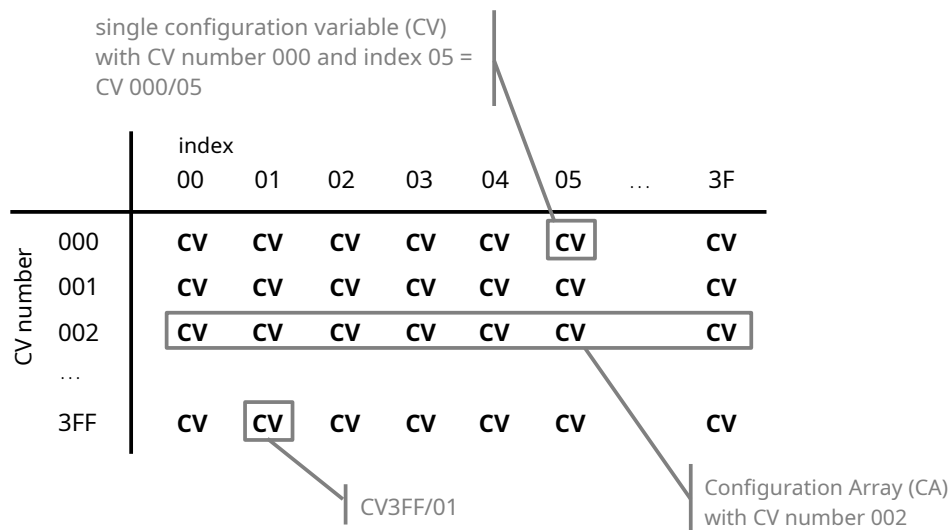
## 6 Decoder configuration

### 6.1 Addressing

The addresses of the configuration variables (CV) are 16 bits long and consist of two parts:

VVVVVVVVIIIIII

The top 10 bits indicate the number of the configuration variable, the bottom 6 bits represent an additional index. V/I is used to write the address. A group of configuration variables, identified by the CV number V, is referred to here as a configuration array (CA) to better distinguish it from individual CVs.



As shown in the sketch, the individual configuration variables are organized in a two-dimensional structure (all information in hexadecimal).

### 6.2 Encoding and byte order

A CV comprises 8 bits, i.e. it contains 1 data byte.

Values that require a larger range are distributed over several consecutive CVs. The byte order is big endian.

The MSB is therefore stored at the smaller (index) address, as the following example shows.

Example		index						
		00	...	05	06	07	08	...
CV number	...							
	002	16		83	00	91	EEE	
	...							

Article number, 24-bit value hex  
0091EE = decimal 37358

Basically, mfx®all CVs at all conceivable addresses can be read and written. However, not all addresses have CVs with a meaningful value stored there and not all CVs can be written to. Corresponding write commands are ignored. When a CV without a defined value is read, 0 is returned. If a single, undefined CV is read, 0 is returned or no response is given at all (depending on the decoder). However, there is no reliable mechanism for detecting undefined CVs.

All numbers in this chapter are hexadecimal unless explicitly stated otherwise.

## 6.3 Structure

### 6.3.1 Structure of the configuration space

The meaning of the CVs is not determined by the CV address. Instead, the structure and meaning of the stored information itself can be read out via CVs. These CVs, which carry meta information, cannot be written, but only read.

When reading and writing, the central unit must navigate through the CVs using the meta information. This is somewhat more complicated than with a fixed assignment of CV addresses to configuration parameters, but it is more flexible and easy to expand. The central unit hides the complexity of the structures from the user and normally only accesses defined CVs.

The CVs of a decoder are structured according to the following rules:

- The address space of the CVs is managed in blocks that are a multiple of 4 CAs long. Any CAs that are not required at the end remain unused.

The first block begins with the first CA, i.e. the CA with CV number 000. To read the complete configuration space, the control center begins by reading CVs in this CA.

- The CV at index 00 specifies the CA type in each CA, i.e. the meaning of the configuration array. The following are defined:

00 = CA not used 01 =

Block description

10 ... XX = CA type depends on the block

The meaning of the CA type is defined within a block; in different blocks the same CA type has a different meaning.

The CV at index 00, i.e. the CA type, is always read-only.

- The CA with the block description is always the first CA in a block and is completely read-only. It is structured as follows (i.e. the CVs at index 00, 01, 02, etc.):

01 BB VV 00 GG LL 01 is the CA

type, see above. BB stands for

the block type, i.e.

01 = Decoder basic setting 02 =  
Functionality

03 = automatic functions 04 =  
Function Mapping  
... etc.

The block types, their structure and the meaning of their CAs are described in Chapter 6.4. GG and LL see next bullet point.

VV stands for the version of the block structure. Typically it is 01.

- Each block is divided into equal-sized groups of CAs.

GG = Number of groups LL =  
Number of CAs per group

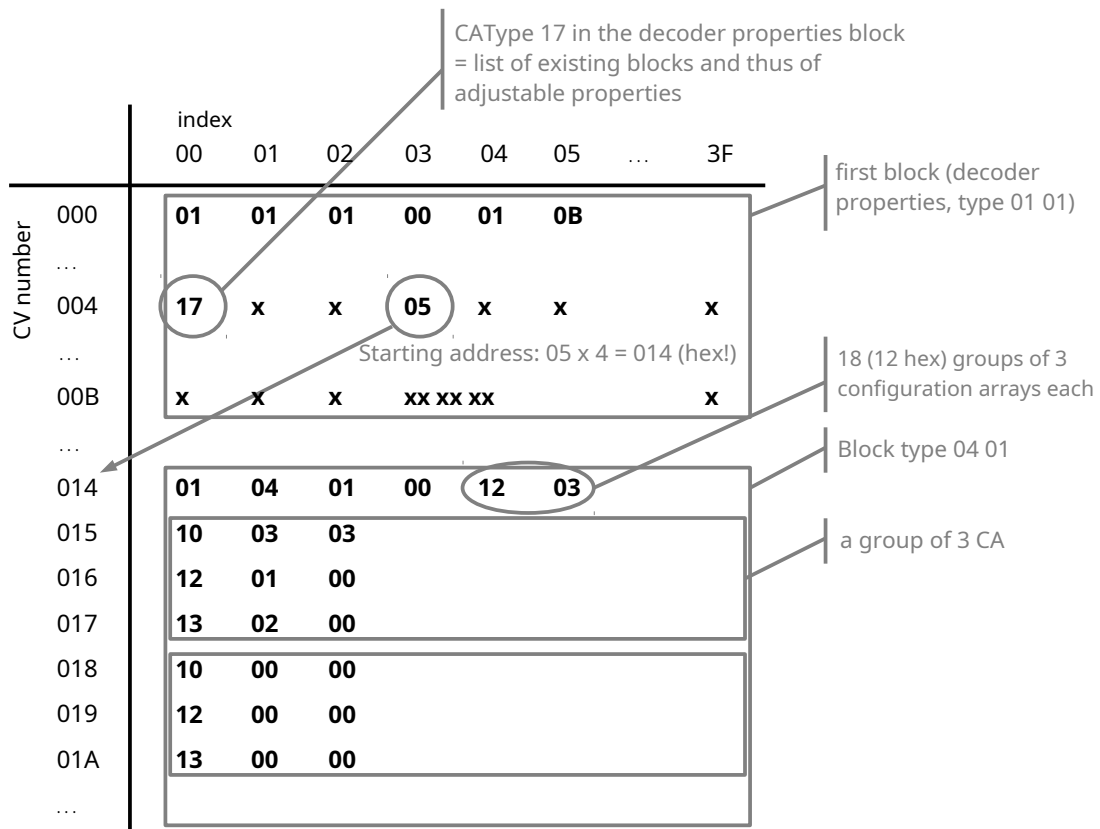
- Within a block, the order of the CAs (or groups of several CAs) can be significant.

Example: In the configuration of the decoder outputs there are 27 groups with one CA each. There are 6 CAs with the meaning 10 (here: configuration of a hardware output). The first CA stands for the first output (front light), the second for the second output (rear light) and so on up to the sixth output (AUX 4). So here it is purely the order that decides, there is no additional CV that indicates which output it is.

The function definitions in block 04 are also listed in order starting from F0. In this case, this does not have to be the case, because there is actually an index table for this in block 02. However, this is usually ignored by the control centers and a corresponding order is assumed in block 04. This saves read accesses and therefore time when reading out the configuration.

- In the first block, which is always present and begins with CV number 000, a list with references to all other blocks is kept in a CA with CA type 17.
- The beginning of a block is always on a CV number that is divisible by 4. The index in the block table can therefore be limited to 8 bits, since the two lower bits are always 0 and are not stored. The index must therefore be multiplied by 4 to find the CV number of the beginning of the block.

The following example illustrates the structure of the CVs.



The following subsections list the meaning of the known CA types, blocks and configuration variables.

### 6.3.2 Rules for the structures for fast registration

To speed up the registration process, it is sufficient to read only the information required for immediate operation from the decoder. Märklin has developed this "quick registration" with the Märklin®Central Station 2 introduced.

In principle, the central unit could, for example, identify which decoder it is from the UID. In this case, it could even dispense with reading the structure CVs and directly access the values of interest if it knows the decoder and how its CV structure is constructed. This would make it easier to register an mfx®Accelerate the locomotive further.

Newer SW (from 3.0) from Märklin®Central Station 2 and Central Station 3 seem to work in this way to some extent. Certain structural information is no longer read out because it is assumed that certain information can be found at fixed CVs (absolute or relative to others).

In the first step, the following information is read out in this order:

- CV address 004: block table (block 01, CA type 17).

Only the CVs at index 01 – 09 are read, i.e. the index of the first 9 blocks.

- In each block description of the 9 blocks, the first two CVs are read (index 0 and 1). Index 0 is used to recognize that it is really the beginning of a block; the value must be 01. Index 1 describes the block type.
- CV address 008: Decoder manufacturer (block 01, CA type 10). Only index 4 is read here, i.e. the manufacturer identification.

The mfx®The configuration space must therefore always set CA type 17 in block 1 to CV address 004 so that the quick login works.

The Märklin®For example, Central Station 2 reads the following information during quick login (SWStand 4.1.2(3)):

- CV address 002: Locomotive identification/locomotive symbol (Block 01, CAType 16).
- CV address 003: locomotive name (block 01, CA type 18).
- Start of block 7 + 12 (= 4th CA in the 3rd group in block 7): MM address (CA type 13). Only index 1, i.e. the first MM address, is read.
- Start of block 5 + 2: Acceleration/braking deceleration (CAType 13).
- Start of blocks 5 + 3: speed characteristic curve (CA type 17). Only the first two values are read, i.e. starting speed and maximum speed.
- Start of block 8 + 1: volume (CA type 10).
- Start of block 4 + 1 ... + 48 in steps of 3, total 16 CA: function assignment (the first 16 groups, first CA in each case, CA type 10).
- Start of Block A + 12: Driver's cab for mfx+®(CAType 1B).

The order listed is only partially correct, but it is not important.

## 6.4 Description of the configuration variables

### 6.4.1 Block 01: Decoder properties

#### CAType 10: DecoderManufacturer

10	HH	HH	HH	HH	PP	PP	PP	PP
----	----	----	----	----	----	----	----	----

HH = Manufacturer identification (ESU®Terminology: manID)

00 00 00 97 = ESU®

00 00 00 83 = Trix®/Märklin®

The last byte corresponds to the manufacturer ID from the DCC standard.

PP = product identifier (ESU®Terminology: proID)

At Märklin®The spare part number of the decoder is stored here (spare part number as it can be found in the spare parts sheet).

Some values of ESU®Decoders, but there are also others: 02 00 00 0E = mfx®Decoder as installed in the first Märklin®mfx®Locomotives

02 00 80 10 = LokPilot mfx®(V2.0, older ESU®61600) 02

00 00 28 = LokPilot V3.0 mfx®

FF FF FF FF means: value is not set. All bytes can only

be read.

### CAType 11: Production identifier

The 8 bytes contain information about the manufacturing process. They are used differently by different manufacturers.

Importance in ESU®:

11	A.A.	BB	CC	CC	DD	DD	DD	DD
----	------	----	----	----	----	----	----	----

AA.BB.CCCC = Production identifier (ESU®Terminology: pInfo) DD

= Production Date (ESU®Terminology: pDate)

The date is given in seconds since 1.1.2000. Meaning at

Märklin®:

11	DD	DD	DD	DD	IN	IN	IN	IN
----	----	----	----	----	----	----	----	----

DD = Date of manufacture

Date as a number; Example: the number 20120314 stands for March 14, 2012 IN

= manufacturing information (usually not used)

FF FF FF FF means: value is not set. All bytes can only

be read.

### CAType 12: BootloaderVersion (Firmware Part B)

8 bytes to identify the bootloader version, use depends on the manufacturer. Meaning for ESU®:

12	A.A.	BB	CC	CC	DD	DD	DD	DD
----	------	----	----	----	----	----	----	----

AA.BB.CCCC = Firmware version B (ESU®Terminology: bCode) DD

= Date of creation of firmware B (ESU®Terminology: bDate)

The date is given in seconds since 1.1.2000. Meaning at

Märklin®:

12	A.A.	BB	CC	VV	DD	DD	DD	DD
----	------	----	----	----	----	----	----	----

AA.BB.CC.VV = BootloaderVersion

DD = Bootloader creation date

Date as a number; example: the number 20120314 stands for March 14, 2012

FF FF FF FF means: value is not set. All bytes can only be read.

### CAType 13: FirmwareVersion (Firmware Part A)

8 bytes to identify the firmware version, use depends on the manufacturer. Meaning for ESU®:

13	A.A.	BB	CC	CC	DD	DD	DD	DD	TT	TT	TT	TT
----	------	----	----	----	----	----	----	----	----	----	----	----

AA.BB.CCCC = Firmware version A (ESU®Terminology: aCode) DD

= Firmware A creation date (ESU®Terminology: aDate)

The date is given in seconds since 1.1.2000. TT = Firmware type (ESU®Terminology: aType) Meaning at Märklin®:

13	A.A.	BB	CC	VV	DD	DD	DD	DD	EEE	FF	GG	HH
----	------	----	----	----	----	----	----	----	-----	----	----	----

AA.BB.CC.VV = Firmware version

DD = Firmware creation date

Date as a number; Example: the number 20120314 stands for March 14, 2012

EE.FF.GG.HH = FirmwareInfo

FF FF FF FF means: value is not set. All bytes can only be read.

### CAType 14: ProtocolInformation

14	P1	P2	PL	MS6 ... MS1 GP	0000 WR1 WR2 RD1 RD2
----	----	----	----	----------------	----------------------

P1.P2 = Protocol version of the mfx®Protocol (only 1.0 observed so far)

PL = maximum packet length in bits of the mfx®Commands (only command without Synchronization pattern, without address and without CRC, also without StuffBits; maximum packet lengths of 63 bits and PL of 64 and 80 have been observed so far.

Write delay:

MS (meaning unclear)

G = Total (meaning unclear) P =

Passive (meaning unclear)

MultiInfo:

WR = maximum number of CVs that can be written in one command RD =

maximum number of CVs that can be read in one command

Attention: The order here is different than usual. The same coding is assumed as in the Read/Writemfx®Commands. However, the information does not match the observations.

All bytes are read only.

### CAType 15: unknown

4 bytes can be stored and read again. This data has no observable effect on the decoder, perhaps only the central unit can store information here (such as the locomotive name, which is stored in this way in type 16). Previously unused, all bytes contain 0.

### CAType 16: Locomotive identification (symbol, manufacturer, article number)

The control center attempts to display a suitable locomotive symbol or locomotive image from this information. Since users can upload their own images to the control center, no unique identification can be stored for the image. Instead, a suitable image must be found using the available information and heuristics.

16	SS	SS	YY	YY	HH	A.A.	A.A.	A.A.
----	----	----	----	----	----	------	------	------

The first 4 bytes define a locomotive symbol for the first Märklin®Device generation and the ESU®Devices.

The Märklin®Mobile Station 1 only knows a few fixed symbols ("generic symbols"). These are determined by the first two bytes. Mobile Station 1 also only reads the first two bytes.

SS SS = generic symbol

01 08 = no picture

01 09 = ELok

01 0A = diesel locomotive

01 0B = steam locomotive

Observation: If only a generic symbol is stored by the Mobile Station 1, the manufacturer identification of ESU®used and YY YY = 00 00. The Central Station 1 also uses YY YY, it is then usually the manufacturer identification of Märklin®filed.

SS SS YY YY = Symbol of Märklin®Central Station 1. In this use it may

Also for SS SS there may be values that differ from the table above.

HH = Manufacturer identification of the locomotive (according to NMRA, like DCC)

97 = ESU®

83 = Trix®/Märklin®

A = Item number of the locomotive

At Märklin®Central Station 2 mainly uses the article number and the name to select a symbol or locomotive image (the name often contains the series number, for which there are then corresponding locomotive images).

**CAType 17: BlockTable**

17	FF	...	0
----	----	-----	---

Null-terminated list of available blocks.

FF = Index of the block start ( $FF \times 4$  = CV number of the first CA of the corresponding block)

**CAType 18: LocomotiveName**

18	CC	...	0
----	----	-----	---

Null-terminated character string with locomotive name. Locomotive names can be up to 16 bytes long. The locomotive name is encoded in UTF8. If only ASCII characters are used, the name can therefore be 16 characters long. If umlauts and other special characters are used, the maximum length is shortened accordingly because these characters are encoded using multiple bytes.

If all 16 bytes are used, the terminating 0 character can be omitted. This would be represented anyway by the non-writable CV at index 17, which, like all undefined CVs, is always read as 0.

CC = Character of the locomotive name in UTF8 encoding

**CAType 19: User data**

A 0-terminated character string of up to 16 bytes can be stored and read again. The data may only be read by the central unit up to the first 0 character. This data has no effect on the decoder; only the central unit can store information here.

No use is known so far.

**CAType 1A: unknown**

1B	??	??	??	??	??	??	??	??
----	----	----	----	----	----	----	----	----

unused or previously only measured with 00 as user data

It could be a 32-bit index register + a 32-bit data register or a register for authenticating the control center and updating firmware or sound. The true meaning and function are unknown.

**CAType 1B: HardwareVersion**

1B	A.A.	BB	CC	DD	EEE	FF	GG	HH	HW	HW	HW	HW
----	------	----	----	----	-----	----	----	----	----	----	----	----

AA.BB.CC.DD.EE.FF.GG.HH = HardwareVersion

HW = Part number (at Märklin®: Number of the unprogrammed HW part, the program HW is then completely managed under the spare part number, as it is also listed in the spare parts sheet)

### 6.4.2 Block 02: Functions

#### CAType 10: Driving function

10	XX
----	----

XX = driving function present (value 01, not present: 00).

This may also be an index if several motors can be configured in block 05.

#### CAType 11: Switching function

11	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

XX = Index in the function assignment table (see block 04) for the functions F0 to F15 (in this order). 00 for non-existent function.

### 6.4.3 Block 03: Automatic functions

#### CAType 10: Automatic switching function when stationary (FS)

10	XX
----	----

XX = Index in the function assignment table (see block 04) for the function FS

The FS function is a pseudo function that is automatically switched on internally when the train is stationary.

#### CAType 11: Automatic switching function when driving (FF)

11	XX
----	----

XX = Index in the function assignment table (see block 04) for the function FF

The FF function is a pseudo function that is automatically switched on internally when the train is moving.

#### CAType 12: Sensor input (I)

12	XX
----	----

XX = Index in the function assignment table (see block 04) for the sensor input.

The sensor input is treated like a function that is triggered by a signal at the corresponding input of the decoder. There can be several sensor inputs. Therefore, several targets of type 12 can also occur; the order is determined by the sensor input number (I1, ...).

### 6.4.4 Block 04: Function assignment

For each function (16 pieces) + FS and FF (functionality when stopped and when driving) + sensor inputs (if available), one group each with the three CAs of type 10, 12 and 13.

#### CAType 10: Function symbol

10	M GGGGGGG	S1	S2
----	-----------	----	----

M = instantaneous (1) or continuous function (0)

For momentary functions, the control unit switches the function on when the button is pressed and off when the button is released. For permanent functions, the function is switched on when the button is pressed for the first time and switched off when the button is pressed for the next time.

The function symbols are defined in three parts, which allows a gradual differentiation between the symbols depending on the display options of the central station. While the mobile station 1 only evaluates the function group (G), the mobile station 2 already reads the function group and part 1 of the symbol information (G S1). The central station evaluates all 3 parts (G S1 S2).

G = Functional group

00 = not assigned to any function group, there is only one specific symbol 02  
 = non-specific function (shown as F)  
 03 = Light  
 04 = Interior lighting  
 05 = Outdoor lighting  
 06 = Mechanical functions 07  
 = Sound  
 08 = Background Sound  
 09 = Voice output  
 0A = shunting gear  
 0B = Direct control (ABV off)

S1 S2 = Function symbol

The following table lists all detailed symbols (specifying G S1 S2; however, the group assignment does not always have to be present, the group could also be 00):

00 00 00 = no function (no symbol)  
 02 02 00 = undefined (question mark or F symbol, on Central Station 2 with number)  
 03 03 00 = Headlight (light symbol)  
 03 03 40 = Driver's cab lighting rear 03 03  
 80 = Driver's cab lighting front  
 04 04 00 = General interior lighting 04 04  
 01 = Interior lighting in the car 04 04 04 =  
 Nostalgic interior lighting 04 04 05 = Ball  
 table lamp (Epoch II)

04 04 06 = Table lamp cone (Epoch III) 04 04  
 07 = Table lamp cylinder (Epoch IV)  
 05 05 00 = Exterior lighting (general) 05  
 05 01 = License plate lighting 05 05 02 =  
 Train destination sign lighting 05 05 03 =  
 Chassis lighting 05 05 06 = Flashing light  
  
 05 05 07 = Staircase lighting 05 05 09  
 = Firebox lighting  
 06 06 00 = Mechanical function (general function symbol)  
 07 07 00 = Sound (SoundSymbol)  
 07 07 01 = Brake noise  
 07 07 02 = Pantograph noise 07 07  
 05 = Coupling noise 07 07 06 = Rail  
 joint  
 07 07 08 = conductor's whistle 07  
 07 09 = switching steps 07 07 0A  
 = blow out cylinder 07 07 0D =  
 shaking grate  
 07 07 0E = Generator 07  
 07 0F = Buffer shock 07  
 07 10 = Shaking grate  
 07 07 11 = general sound (symbol musical note)  
 07 07 13 = let off steam  
 07 07 14 = Engine noise 07  
 07 2E = Sanden  
 07 07 2F = Switch off brake sound  
  
 08 08 00 = Background sound (symbol locomotive from the front)  
 09 09 00 = Announcement (loudspeaker symbol)  
 09 09 01 = Announcement (loudspeaker symbol with mounting rod pointing downwards) 09  
 09 02 = Announcement (loudspeaker symbol with mounting rod pointing upwards) 09 09 04 =  
 Announcement (mast symbol)  
 09 09 05 = Speech output (speech bubble symbol)  
  
 0A 0A 00 = shunting gear (turtle symbol)  
 0A 0A 01 = Shunting (symbol: no stop before red signal)  
  
 0B 0B 00 = direct control (mass symbol) 0B 0B xx  
 = alternative speed characteristic  
  
 02 1F 00 = User defined function  
  
 06 20 00 = Telex coupling 00 20  
 00 = Telex coupling 00 20 40 =  
 Telex coupling rear 00 20 80 =  
 Telex coupling front

06 21 00 = Smoke generator

00 21 00 = Smoke generator

06 22 00 = Pantograph 00 22

00 = Pantograph 00 22 40 =

Pantograph rear 00 22 80 =

Pantograph front

03 23 00 = High beam 00 23

00 = High beam 00 23 40 =

High beam rear 00 23 80 =

High beam front

07 24 00 = Bell 00

24 00 = Bell

07 25 00 = Horn

00 25 00 = Horn

07 26 00 = Whistle

00 26 00 = Whistle

00 27 00 = Open doors 00

27 01 = Close doors

00 28 00 = Fan 00

28 01 = Fan

00 29 00 = Pump 00

29 01 = Air pump

00 29 02 = Fan (intake) 00 29

04 = Injector

00 29 05 = Water pump

00 2A 00 = Coal shovels

00 2B 00 = crane hook

00 2B 01 = Arrows for tilting (forward/up) 00 2B 02 =

Arrow up/down

00 2B 03 = Arrow turn (front to right) 00 2B

04 = Arrow right/left

00 2B 05 = double hook

00 2B 06 = magnet

00 2B 07 = Arrow up 00 2B

08 = Arrow down 00 2B 09 =

Arrow left 00 2B 0A = Arrow

right

00 2B 0C = Turn arrow (front to left)

00 2C 00 = thick arrow pointing up

00 2C 01 = Sound off (crossed-out speaker symbol)

For some function symbols, a line with group assignment and another with group 00 was inserted because both combinations were observed. In principle, however, group membership should be independent of the symbol.

#### CAType 12: Function when driving forward

12	RG DS A4 A3 A2 A1 LH LV	S4 S3 S2 S1 S0 SE AS BA
S12 ... S5		0 0 0 0 0 S15 .. S13

The functions marked with 1 are active when the activated function is switched on while driving forward.

The description of the meaning of the bits refers here to the first ESU®/Märklin® mfx® and mfx® Sound decoder with 4 AUX outputs. It should therefore be seen as an example configuration. Newer decoders even define 6 AUX outputs (for operation as a function decoder, function outputs instead of motor), the bits shift accordingly. In addition, up to 28 sound slots are now offered.

Basically, an assignment to the hardware functions takes place (bit 0, index 01 = hardware function 0; bit 1, index 01 = hardware function 1, etc.). The existing hardware functions and thus the order are determined by the information in block 06. For newer decoders with more hardware outputs (AUX), the scheme must be adapted accordingly.

LV = Front light output LH = Rear

light output A4 ... A1 = AUX 4 to

1 output

DS = Direct control (acceleration curve & braking curve off = ABV off) RG =

Shunting gear

BA = Switch off brake noise

AS = Switch on alternative sound or mute SE =

Switch on sound (switch on driving sound) Sxx =

SoundSlotNumber xx

#### CAType 13: Function when reversing

13	RG DS A4 A3 A2 A1 LH LV	S4 S3 S2 S1 S0 SE AS BA
S12 ... S5		0 0 0 0 0 S15 .. S13

The functions marked with 1 are active when the activated function is switched on when reversing.

The meaning of the fields corresponds to that for forward travel, see above for CAType 12.

### 6.4.5 Block 05: Driving characteristics

#### CAType 10: Engines

10	A.A.
----	------

AA = number of motor outputs, 0 for function decoders that do not control a motor

#### CAType 11: Engine type

11	00 AR DDDD
----	------------

The motor type determines how the motor is controlled by the decoder.

A special form is the motor type "AUX". In this mode, the two motor connections act as additional function outputs (AUX 5 and AUX6).

A = Control in analogue mode (1 = yes, 0 = no) R =

Control dynamically reduced (1 = yes, 0 = no) D =

Motor type:

- 0 = AUX, motor output stage is used as two function outputs 1 = SDSMotor, unregulated
- 2 = unregulated motor
- 3 = regulated DC motor (high-performance drive C90)
- 4 = regulated bell-armature motor
- 5 = regulated DC motor (soft) 6 = regulated DC motor (hard) 7 = regulated DC motor (track 1)

The entire 6 bits of the engine type can also be considered as a unit and then indicate an identification number for the engine type from 0 – 63.

The SDS motor has its own control and therefore must not be controlled by the decoder. If type 1 = SDS is set, the control parameters must be set to 0 (especially the control influence parameter).

#### CAType 12: Motor control frequency

12	FF	FF
----	----	----

FF FF = PWM frequency of the motor control in 10 Hz steps

Depending on the decoder, only a few frequencies are possible. For example, the following were observed:

- 06 1A = 15kHz
- 0C 35 = 30kHz

**CAType 13: Acceleration time, braking deceleration**

13	A.A.	BB
----	------	----

AA = Acceleration time (AA x 0.25 sec.) BB =

Braking deceleration (BB x 0.25 sec.)

The time required for the locomotive to accelerate or brake from 0 to  $V_{\max}$  is given.  
needed.

**CAType 14: Trim**

14	FF	RR
----	----	----

FF = Forward trim (FF x 0.0078125 x motor voltage) RR =

Reverse trim (RR x 0.0078125 x motor voltage)

The trim is used to  $V_{\max}$  to be set differently for forward and reverse travel. This accommodates locomotives that have different maximum speeds forward and backward.

With a value of 80 (decimal 128) there is no influence (factor is then 1.0), with other values the motor voltage derived from the speed characteristic is multiplied by the factor specified here. This then gives the motor voltage actually used.

**CAType 15: Load control**

15	EE	RR	KK	II
----	----	----	----	----

EE = control influence (EE x 0.39216%)

RR = control reference (RR x 0.1 volt), absolute  $V_{\max}$  for motor characteristic KK =  
control parameter K

II = Control parameter I

**CAType 16: Braking distance**

16	00 AB 000 C
----	-------------

A = Braking distance due to direct current on the track (opposite direction of travel, normal  
Activate 2-wire braking behavior (value 1)

B = braking distance due to direct current on the track (in direction of travel, inverse  
Activate 2-wire braking behavior (value 1)

If AB = 11, i.e. both are activated, the direction-independent braking with direct current (3-wire behavior) is activated.

C = Märklin® Activate braking section (value 1, this bit is only available in the first ESU®/  
Märklin®mfx®Decoders)

**CAType 17: Speed characteristic**

17	MM	A.A.		...	KK
----	----	------	--	-----	----

MM = Starting speed  $V_{\min}(0255)$  AA =

Maximum speed  $V_{\max}(0255)$

KK = 26 support points of the speed characteristic

The speed characteristic has 28 points, whereby the first and last points are constant (01 and FF) and are therefore not saved. The support points are specified as factors between 01 and FF.

**CAType 18: Speedometer**

18	TT	TT
----	----	----

TT = Maximum speed displayed on the speedometer of the control center in km/h

Attention: Unlike other CVs, the speedometer value is stored in the Little Endian byte order. Example: 120 km/h maximum speed is represented with TT TT = 78 00.

**CAType 19: Reverse operation**

19	0000000R
----	----------

R = Activate reverse operation (reverse the direction of the locomotive in relation to Direction of travel and light)

**6.4.6 Block 06: Hardware functions**

Groups of one CA each. Each group and therefore each CA represents an existing hardware function that can be configured in this CA. The order determines which bit represents the hardware function in the function assignment table.

There are several groups of most CA types. The order is important because it determines the assignment to the hardware (eg first CA of CA type 10 controls output 1 alias front light, sixth CA of CA type 10 controls output 6 alias AUX4).

**CAType 10: Configuration of hardware outputs**

10	MM	DD	FF
----	----	----	----

Configuration of a hardware output. The order of the information determines the assignment: 1. entry front light, 2. entry rear light, 3rd to 6th entry AUX1 to AUX4.

MM = Output mode

00 = no function 01 =

continuous light

02 = Flashing light (on/off time is the same length)

03 = Flashing light (opposite phase, i.e. is off when flashing light 02 is on and vice versa)

04 = Strobe light effect (short flash of light at the beginning of each period)  
 05 = Light effect Double Strobe (two short flashes of light at the beginning of each period)  
 06 = Random light effect (for firebox flickering)  
 07 = Control of a smoke generator  
 08 = Permanent light, slowly fades in and out when switched on/off  
 09 = Marslight light effect (for American locomotives)  
 0A = Gyrolight light effect (for American locomotives)  
 0B = Rule 17 forward light effect (for American locomotives)  
 0C = Rule 17 reverse light effect (for American locomotives)  
 0D = Neon light (flickers for a while when switched on, goes out immediately)  
 0E = Energy saving lamp light (takes some time to reach full brightness)  
 0F = Telex ("SoftTelex" for the newer Telex couplings, reduces the current in the Holding phase)

DD = Dimming (0255), valid for all effects

FF = flashing frequency (FF x 0.05 sec.) or time depending on the light effect:

Effect 02 – 05: Period duration Effect  
 08: Fade up/down time Effect 09 and  
 0A: Speed Effect 0B and 0C:  
 Dimming time Effect 0D: Flickering  
 time  
 Effect 0E: Time until full brightness is reached when switching on  
 Effect 0F: Coupling time  
 all other effects do not use the specification

Note: in the tested first Märklin®/ESU®mfx®Decoders only allow one flashing frequency (presumably because the microcontroller lacks the necessary resources). If you set the flashing frequency for a specific output on these decoders, this value is automatically adopted for all other outputs.

Note: For newer Märklin®mfx®Decoders there is no difference between flashing lights 02 and 03, in particular they do not light up in opposite phase.

### CAType 11: Configuration of internal functions

11	TT	PP
----	----	----

TT = Type of internal function

01 = Direct control (switch off ABV)  
 02 = Shunting gear  
 03 = Brake noise  
 04 = Alternative sound (on older ESU®/Märklin®mfx®Sound decoders)  
 06 = Mute  
 07 = Alternative sound (on newer Märklin®mfx®Sound decoders)  
 meanwhile: brake dynamics (= brake squealing game world)  
 If a, a brake squeal is also triggered when the locomotive is slowly braked over many speed levels (with PC control or ABV simulation in the Märklin®Central Station.

PP = Parameter for the corresponding function

Function 01: presumably a divisor that determines by which part the ABV is reduced

Function 02: Divider for the speed in shunting mode (bit 7 = ½, bit 6 = ¼ etc.;  
normally 80 for half normal speed) Function 03, 04,  
(07 old): Volume  
Function 06, 07: no meaning

### CAType 12: Configuration of sound functions

12	A.A.	LL
----	------	----

Configuration of the sound functions, the order determines the number of the sound slot (the first entry with AA = 02 describes the sound slot number 1, etc.).

AA = Type of sound

01 = Operating sound (driving sound)  
02 = SoundSlot

LL = volume (0 = off, FF = full volume)

### 6.4.7 Block 07: Further protocols

Blocks with 4 CAs each; each block configures an additional protocol. Which one can be found in CAType 10.

### CAType 10: Protocol

10	PP
----	----

PP = further protocol

01 = Analogue AC  
02 = Analogue DC  
03 = MM  
04 = DCC

### CAType 11: ProtocolConfiguration

The configuration bits depend on the protocol (except bit 0 = activation).

Analog AC or DC:

11	0000000A
----	----------

A = Protocol activated (1)

MM:

11	000 F DD HA
----	-------------

F = Use follow-up addresses (value = 1) or freely definable addresses (value = 0)

This setting defines how the 2nd and all subsequent addresses are determined. Starting from the set MM address, the subsequent addresses can be used automatically or all addresses can be explicitly defined in the decoder.

D = number of addresses (00 = 1 address, 01 = 2 addresses, 10 = 3 addresses, 11 = 4 addresses)

The 1st address controls the speed level and F0 – F4, the 2nd address F5 – F8 etc. H = allow half-steps (value 1 means: 27 speed levels via half-step method) A = protocol activated (1)

DCC:

11	00 E 00 PSA
----	-------------

E = use short addresses (0) or long addresses (1) P = Power  
(function unknown)

S = Use 14 speed steps (0) or 28/128 speed steps (1) A =  
Protocol activated (1)

### CAType 12: Enabled functions

12	F15 F14 ... F8	F7 F6 ... F0
----	----------------	--------------

Set bit means: This function is automatically activated when driving in this protocol.

### CAType 13: Addresses in MM format

13	M1	M2	M3	M4
----	----	----	----	----

M1 = MM address (driving and functions F0 to F4)

M2 = 2nd MM address (functions F5 – F8 through MM function F1 to F4) M3 =

3rd MM address (functions F9 – F12 through MM function F1 to F4) M4 = 4th

MM address (functions F13 – F15 through MM function F1 to F3) Coding

according to the signal form of the MM rail signal:

0 = means short pulse (1/8 of the bit length high, 7/8 low) 1  
= long pulse (7/8 of the bit length high, 1/8 low)

The coding of the first 81 addresses was done by Märklin® and corresponds to the original MM format (Märklin®). However, only addresses 1 to 80 are used; address 0 is sometimes referred to as address 81 and indicates that the decoder is operated analogously in some decoders). All bit patterns have now been defined, so that 255 addresses are available (0 is still not used). This results in the following values:

Address 0 – 9: 55, 03, 01, 0C, 0F, 0D, 04, 07, 05, 30  
Addr. 10 – 19: 33, 31, 3C, 3F, 3D, 34, 37, 35, 10, 13

Addr. 20 – 29:	11, 1C, 1F, 1D, 14, 17, 15, C0, C3, C1 CC, CF,
Addr. 30 – 39:	CD, C4, C7, C5, F0, F3, F1, FC FF, FD, F4, F7,
Address 40 – 49:	F5, D0, D3, D1, DC, DF DD, D4, D7, D5, 40, 43,
Address 50 – 59:	41, 4C, 4F, 4D 44, 47, 45, 70, 73, 71, 7C, 7F,
Address 60 – 69:	7D, 74
Addr. 70 – 79:	77, 75, 50, 53, 51, 5C, 5F, 5D, 54, 57 00, 02,
Address 80 – 89:	06, E9, 0E, 12, 16, 1A, 1E, 22 26, 2A, 2E, 32,
Addr. 90 – 99:	36, 3A, 3E, 42, 46, 4A 4E, 52, 56, 5A, 5E, 62,
Addr. 100 – 109:	66, 6A, 6E, 72 76, 7A, 7E, 82, 86, 8A, 8E, 92,
Addr. 110 – 119:	96, 9A 9E, A2, A6 , F9, AE, B2, B6, BA, BE, C2
Address 120 – 129:	C6, CA, CE, D2, D6, DA, DE, E2, E6, EA EE, F2,
Addr. 130 – 139:	F6, FA, FE, 08, 18, 28, 38, 48 58, 68, 78, 88,
Addr. 140 – 149:	98, A8, B8, C8, D8, E8 F8, 0B, 1B, 2B, 3B , 4B,
Addr. 150 – 159:	5B, 6B, 7B, 8B 9B, AB, BB, CB, DB, EB, FB, 09,
Address 160 – 169:	19, 29 39, 49, 59, 69, 79, 89, 99, A9, B9, C9
Address 170 – 179:	D9, 0A, AA, 20, 60, A0, E0, 23, 63, A3 E3, 21,
Address 180 – 189:	61, A1, E1, 2C, 6C, AC, EC, 2F 6F, AF, EF, 2D,
Addr. 190 – 199:	6D, AD, ED, 24, 64, A4 E4, 27, 67, A7, E7, 25,
Addr. 200 – 209:	65, A5, E5, 80 83, 81, 8C, 8F, 8D, 84, 87, 85,
Addr. 210 – 219:	B0, B3 B1, BC, BF, BD, B4, B7, B5, 90, 93, 91
Addr. 220 – 229:	9C, 9F, 9D, 94, 97, 95
Address 230 – 239:	
Address 240 – 249:	
Address 250 – 255:	

Attention: In MM format, bit 0 is sent first. This corresponds to bit 0 in the address definition above.

### CAType 13: Addresses in DCC format

13	AD	EA (MSB)	EA (LSB)	RTTTTTTT
----	----	----------	----------	----------

AD = DCC address (short address) EA

= extended (long) DCC address

T = Traction address (0 = no multiple traction)

The DCC address is ignored in multiple traction. In order to be able to drive the locomotive individually again, the traction address must be set to 0.

R = Direction reversal in multiple traction (value = 1, affects direction of travel and Light function)

### CAType 14: Voltage range in analog mode

14	MM	A.A.
----	----	------

MM = starting voltage (0255, proportion of the available voltage)

AA = Voltage at maximum speed (0255, proportion of available voltage)

### 6.4.8 Block 08: Sound settings

Depending on the sound decoder, not all setting options are available. The interpretation of the parameters also differs slightly.

#### CAType 10: Volume

10	LL
----	----

LL = Volume (0255)

#### CAType 11: Sound type (diesel locomotive/electric locomotive)

At ESU®Sound decoders:

11	TT	TT
----	----	----

TT TT = sound type, the bit fields are interpreted as follows:

00000001 0AAAAAAA = Diesel locomotive electric / ELok

A = Distance between gears (1127) 00000001 10000000

= Diesel locomotive hydraulic 00000000 0AAAAAAA = Diesel

locomotive with manual transmission

A = Number of switching stages (1127)

At Märklin®Sound decoders:

11	A1	A2
----	----	----

A1 = Distance between steam blasts at speed 1

For wheel sensor: 0 = at every pulse

For ELok/diesel locomotive: Speed modulation 1  
= step-wise, linear modulation

A2 = Distance between steam blasts at speed > 1

For wheel sensor: 0 = at every pulse, >0 = at every xth pulse

For ELok/diesel locomotives: speed modulation  
0 = stepwise linear, fixed distance between the speed steps

#### CAType 12: Sound type (steam locomotive)

Only used by ESU®Sound decoders supported.

12	TT	TT
----	----	----

TT TT = sound type, the bit fields are interpreted as follows:

0MMMMMMM 0AAAAAAA = Steam locomotive without external sensor

M = Distance between steam blasts at speed level 1 (M x 0.064 sec.; 1127) A

= Distance between steam blasts at speed level > 1 (slope; 1127)

00000000 0TTTTTTT = Steam locomotive with external sensor

T = TriggerImpulse (1127)

### CAType 13: Speed of sound

13	MM	A.A.
----	----	------

MM = speed of driving noise at lowest gear ( $MM \times 0.78125 \%$ ) AA = speed of driving noise at highest gear ( $AA \times 0.78125 \%$ )

### CAType 14: Random noise

14	MM	A.A.
----	----	------

If MM and AA are greater than 0, the random sound is activated. MM

= minimum distance in seconds

AA = maximum distance in seconds

### CAType 15: Brake noise

15	SS
----	----

SS = Threshold for brake noise (1255)

The larger the value, the earlier the braking sound starts, and the smaller the value, the later the braking sound starts. However, if the value is too small, the braking sound will never be triggered.

### CAType 16: Automatic Sounds

16	RW	0 RLK AAAA
----	----	------------

RW = number of the sound that is played when the direction changes

Configuration of the multi-station announcement. This must be provided in the sound configuration. When the corresponding sound slot is played, a different announcement is played. The order is defined in the CV.

R = Order of announcements (0 = start with announcement 1 in ascending order, 1 = descending order) L = Reverse order when the direction of travel is changed (1)

For commuter trains: If the train turns around at the terminus, the announcements are played in reverse order.

K = Reverse order when reaching the last announcement (final announcement) (1)

For reversing loops: When the last station (= last announcement) is reached, the train turns around in a reversing loop and travels to the stations again in reverse order. Accordingly, the announcements at the end are played in reverse order (the last announcement, the final announcement, is only played once).

A = number of stations (= number of announcements)

### 6.4.9 Block 09: Special options

#### CAType 10: Storage Options A

10	0 0 0 0 0 RSZ
----	---------------

R = After reset (loss of voltage) start again with acceleration (1)  
or take the last speed without taking the acceleration curve into account (0); only has an effect if the speed has also been saved

S = Save current target speed when switching off (1) Z = Save  
state of functions when switching off (1)

#### CAType 11: Storage Options B

11	0 0 0 0 0 0 F
----	---------------

F = Save direction of travel when switching off (1)

#### CAType 12: Configuration of outputs (ports)

12	0 A6 A5 A4 A3 A2 A1 A3 A4L A5L A6L A7 A8
----	--

A6 = Use push/pull output stage for AUX6 (1) A5 = Use  
push/pull output stage for AUX5 (1)

A4 = Use AUX4 as logic output (1), otherwise amplified output A3 = Use AUX3  
as logic output (1), otherwise amplified output G = Swap track connections

L = Swap front/rear light outputs M = Swap  
motor connections

### 6.4.10 Block 0A: mfx+®

#### CAType 10: Current equipment inventory

10	FA	Facebook	SD
----	----	----------	----

Percentage of operating fluids still available (0 = 0%, empty; FF = 100%, full). FA =

Current operating fluids for Fuel A (water, diesel)

FB = Current operating resource stock for Fuel B (coal) SD

= Current operating resource stock for sand

**CAType 11: Maximum operating resource stock**

11	FA	FA	Facebook	Facebook	SD	SD
----	----	----	----------	----------	----	----

Absolute value of the resource stock (each as a 16-bit value) when the current stock is 100%.

FA = Maximum operating fluid stock for Fuel A in liters FB =

Maximum operating fluid stock for Fuel B in kilograms SD =

Maximum operating fluid stock for sand in kilograms

**CAType 12: Factor for resource consumption**

12	FA	Facebook	SD
----	----	----------	----

The current resource supply is "consumed" depending on usage, i.e. it is gradually reduced. In these CVs, the step size for each resource can be defined and thus consumption can be controlled.

FA = step size for Fuel A (reduction per interval) FB =

step size for Fuel B (reduction per interval)

SD = step size for sand (reduction per start from standstill)

With step size 0, the operating resource is ignored (no consumption, no emergency operation).

**CAType 13: Operating Resources Reserve**

13	FA	Facebook	SD
----	----	----------	----

If the respective reserve supply (threshold value) is undercut, the decoder goes into emergency mode ("reserve level").

FA = Operating resource reserve for Fuel A

FB = Operating resource reserve for Fuel B

SD = Operating resource reserve for Sand

**CAType 14: Emergency speed**

14	LS	SS
----	----	----

LS = Maximum speed level in emergency mode when a supply of operating fluid is empty, i.e. 0 ("EmptyLevel")

SS = Maximum speed in emergency mode, when a supply of operating fluids exceeds the reserve level has fallen below ("reserve level")

If the respective threshold is undershot, the maximum speed level defined here will be used, even if a higher speed level is selected via the drive command.

**CAType 15: Driving time**

15	DD	DD	DD	DD
----	----	----	----	----

D = total travel time in seconds; i.e. the total time the locomotive was running

The total driving time is also recorded in standard mode. This value is not reset when resetting. The value can only be read.

**CAType 16: Distance**

16	SS	SS	SS	SS
----	----	----	----	----

S = total distance travelled

The distance travelled is calculated from the engine speed and the gear ratio. If the gear ratio is set accordingly, the distance can be measured in mm. However, slippage and measurement inaccuracies can result in a relatively high error.

The total distance travelled is also recorded in standard mode. This value is not reset when resetting. The value can only be read.

**CAType 17: Load**

17	DD	DD
----	----	----

D = Measure of the current load

The cyclic load measurement only works in the mfx+®mode and only if Fuel A is used (Fuel A factor > 0). The value is continuously recalculated by the decoder based on the driving data.

Note: In an earlier version, the value was referred to as a measure of the maximum speed. The value is read-only.

**CAType 18: Gear reduction**

18	UU	UU
----	----	----

U = Factor for calculating the distance travelled from the engine speed (see distance travelled)

The factor can be viewed as a gear reduction. The counter-EMF voltage is used as a measure of the motor speed, and the factor is used to further calculate the distance. The actual mechanical conditions cannot therefore be determined from the factor.

**CAType 19: Interval**

19	IA	IB
----	----	----

Controls the intervals (value 00 ... FF) at which the two resources Fuel A and Fuel B are reduced. Together with the factor, this can be used to define consumption.

IA = Interval for Fuel A

IB = Interval for Fuel B

#### CAType 1A: mfx+®mode

1A	MM
----	----

Mode in mfx+®Operation; is activated when logging on to an mfx®Central unit automatically deleted (= standard mode set).

00 = Standard mode, no resource consumption and no emergency operation

01 = mfx+®mode

#### CAType 1B: Driver's cab

1B	FF	FF
----	----	----

FF = Number of the driver's cab displayed by Central Station 2

If the control center does not know the cab number, one of the three standard cabs is selected depending on the MobileStation symbol.

#### 6.4.11 DecoderReset

The decoder reset is generated by writing the value 0xC6 into CV 000/01. Normally, however, structural information is read from this register, so the value cannot be read back.