



# MBCAN

Fernsteuerung einer  - Modelleisenbahn

*Nicht-kommerzielles Projekt – Alle Angaben ohne Gewähr*

## Bedienungsanleitung **Servodecoder mbc-84**

*Version 1.2*

*HW 24.01.06, FW 1.00, Parametriercenter ab 2.1.0.0*

*©2007 – 2024 by Dr.-Ing. Thomas Wiesner*

## 1 Inhalt

2	Disclaimer .....	4
3	Revision .....	5
3.1	Bedienungsanleitung .....	5
3.2	Firmware .....	5
4	Einleitung.....	6
5	Funktion.....	7
6	Schaltbild .....	8
7	Bestückung .....	10
8	Bauteileliste.....	12
9	Firmware .....	13
10	Steckverbindungen.....	14
11	Anschlussbeispiele.....	16
12	Inbetriebnahme.....	17
12.1	Modul in Betriebsbereitschaft versetzen.....	17
12.2	Modul konfigurieren .....	17
12.3	Modul mit der CS2 <sup>®</sup> verbinden.....	21
12.4	Modul mit der CS3 <sup>®</sup> verbinden.....	22
13	Modulbilder .....	25
14	Systemarray-Belegung für Eigenentwicklungen .....	26
14.1	Allgemeiner Bereich zum Modul.....	26
14.2	Modulspezifischer Bereich für Funktionsparameter .....	29
15	Befehlssatz zu den Modulen .....	31
15.1	PC_DH_H - Übertragung des Datenbanknamens mit 12 Bytes (1-4).....	33
15.2	PC_DH_M - Übertragung des Datenbanknamens mit 12 Bytes (5-8).....	34
15.3	PC_DH_L - Übertragung des Datenbanknamens mit 12 Bytes (9-12).....	35
15.4	PC_KENNER - Kenner und Identifier für die Module .....	36
15.5	PC_NEU - Neuanmeldeaufforderung .....	37
15.6	PC_NEU_DATA - Rückmeldung PC an Modul während des Neuanmeldeprozesses .....	38
15.7	MD_NEU_DATA - Meldung des Moduls während des Neuanmeldeprozesses .....	39
15.8	PC_RESET - Durchführen eines Hardware-Resets auf dem Modul.....	40
15.9	PC_MD_SEL - Modul aus Datenbank entfernen .....	41



15.10	PC_ALIVE - ALIVE-Abfrage.....	42
15.11	MD_ALIVE - ALIVE-Abfrage.....	43
15.12	PC_ARRAY - Zugriff Systemarray anfragen .....	44
15.13	MD_ARRAY - Zugriff Systemarray freigegeben.....	45
15.14	PC_ARRAY_DATA - Zugriff Systemarray freigegeben .....	46
15.15	MD_ARRAY_DATA - Antwort des Moduls auf Systemarray-Zugriff .....	47
15.16	PC_UPGRADE - Firmware-Upgrade .....	48
15.17	MD_UPGRADE - Firmware-Upgrade freigegeben .....	49
15.18	PC_UPGRADE_DATA - Schreibe Firmware.....	50
15.19	MD_UPGRADE_DATA - Antwort des Moduls auf Schreibe Firmware.....	51
15.20	PC_BOOT - Modul neu Booten .....	52
15.21	MD_S88 - Stellungsmeldung mbc-88 / mbc-90.....	53
16	Post-Code .....	54
17	Quellenverzeichnis .....	56
18	Allgemeine Hinweise zum MBCAN-Projekt .....	57

## 2 Disclaimer

**ACHTUNG: Nur für erfahrene Elektronikbastler geeignet. KEIN Kinderspielzeug!**

Bei Arbeiten an oder mit der aus dieser Dokumentation erstellten Leiterplatte beachten Sie bitte:

- Der Betrieb ist nur an Spannungen kleiner 24 V DC erlaubt. Verwenden Sie ausschließlich geprüfte und zugelassene Steckernetzteile
- Zusammenbau oder Instandsetzungen/Änderungen an der Leiterplatte sind immer im spannungsfreien Zustand durchzuführen
- Betreiben Sie das Gerät nur in trockenen Räumen. Beim Einsatz im Freien sollten Sie entsprechende Maßnahmen zum Schutz gegen Feuchtigkeit ergreifen
- Die zulässigen Ströme an den Schaltausgängen sind einzuhalten. Details finden Sie im jeweiligen Kapitel zur Funktion (vgl. Kapitel 5)
- Dieses Produkt ist nicht für die Nutzung durch Kinder unter 14 Jahren geeignet. Die Anforderungen an Kinderspielzeug werden NICHT erfüllt

Bitte beachten Sie außerdem das Kapitel „Allgemeine Hinweise zum MBCAN-Projekt“ bevor Sie mit dem Nachbau oder der Anwendung der Informationen für eigene Entwicklungen beginnen.

### 3 Revision

#### 3.1 Bedienungsanleitung

1.0	15.02.2024	Erste Version
1.1	01.10.2014	Redaktionelle Anpassungen

#### 3.2 Firmware

FW: 1.00 - Basisversion

## 4 Einleitung

"Machine-to-Machine (M2M) steht für den automatisierten Informationsaustausch zwischen Endgeräten wie Maschinen, Automaten, Fahrzeugen oder Containern untereinander oder mit einer zentralen Leitstelle, zunehmend unter Nutzung des Internets und den verschiedenen Zugangsnetzen, wie dem Mobilfunknetz. Eine Anwendung ist die Fernüberwachung, -kontrolle und -wartung von Maschinen, Anlagen und Systemen, die traditionell als Telemetrie bezeichnet wird. Die M2M-Technologie verknüpft dabei Informations- und Kommunikationstechnik."

[Wikipedia, [https://de.wikipedia.org/wiki/Machine\\_to\\_Machine](https://de.wikipedia.org/wiki/Machine_to_Machine)]

Was für professionelle Systeme gilt, kann für die Automatisierung einer Modelleisenbahn nicht schlecht sein. Auch hier haben wir eine Leitstelle (bei Märklin® die CS2/3® oder MS2®) und verteilte Komponenten, die über den CAN-Bus verbunden sind. Auf dem CAN-Bus finden wir ein von Märklin® definiertes Protokoll vor. Der Austausch von Informationen erfolgt dann automatisch, wobei es keine reine Master-/Slave-Struktur auf dem Bus gibt, sondern ein Multi-Master-System. Das bedeutet, dass sich die mbc-Module bei Änderungen im Prozess, z.B. beim Umstellen der Weiche, selbständig bei der Leitstelle melden (Aktoren). Gleiches gilt für die Rückmelder (Sensoren).

Neben den Sensoren sind vor allem Aktoren für eine ferngesteuerte Modelleisenbahn notwendig. Diese sind entweder Magnetspulenantriebe oder, immer mehr aufkommend, Servomotoren.

Im Folgenden finden Sie hier die Beschreibung eines Decoders zur Ansteuerung von vier Aktoren in Form von Servomotoren.

## 5 Funktion

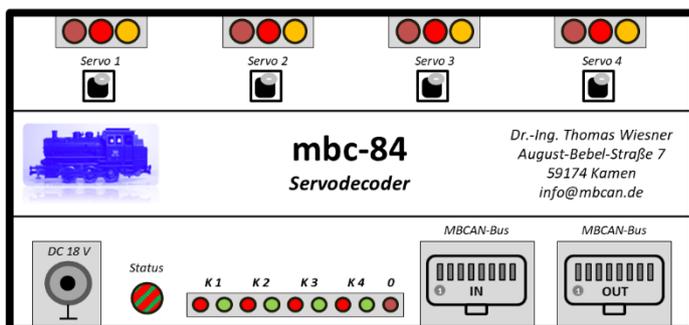


Abbildung 5-1:Modullabel

Dieser Decoder stellt die Schnittstelle für vier Servos zur Ansteuerung von Weichen, Signalen und vergleichbaren Antrieben bereit inkl. Stellungsrückmeldung.

Jeder Ausgang ist mit max. 1 A bei 5 VDC belastbar. Der Weg des Servos ist über das Parametriercenter frei vorgebar.

Es können gängige Servos des Herstellers Graupner verwendet werden, die Steckverbinder des Moduls sind darauf ausgelegt.

<b>Steckverbinder</b>	2x RJ45 MBCAN-Bus / Servos über PINHEAD-Verbinder
<b>Stromversorgung</b>	Steckernetzteil / RJ45-Versorgung
<b>Statusanzeige</b>	Betriebszustand und Traffic wird über Dreifarb-LED angezeigt
<b>Adresszuordnung</b>	Adresse per PC-Software einstellbar
<b>Adressformat</b>	Märklin® Motorola über Gleissignal / CAN-Bus-Protokoll (Loc-ID 0x00003000 - 0x000033ff) möglich
<b>Features</b>	Bei Anschluss an CS2/3® resp. MS2® automatische Rückmeldung / Parametrierung, Firmwareupdate und Auslesung über PC möglich / Anzeige des Moduls in der GUI der CS2/3®

## 6 Schaltbild

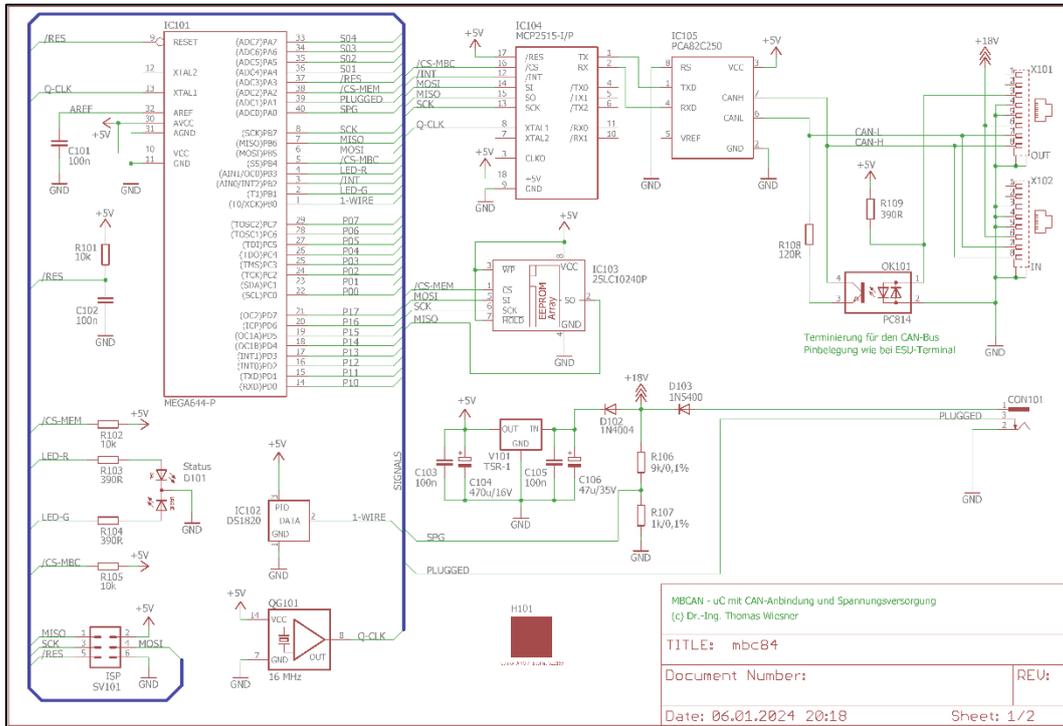


Abbildung 6-1: Prozessor-Schaltbild

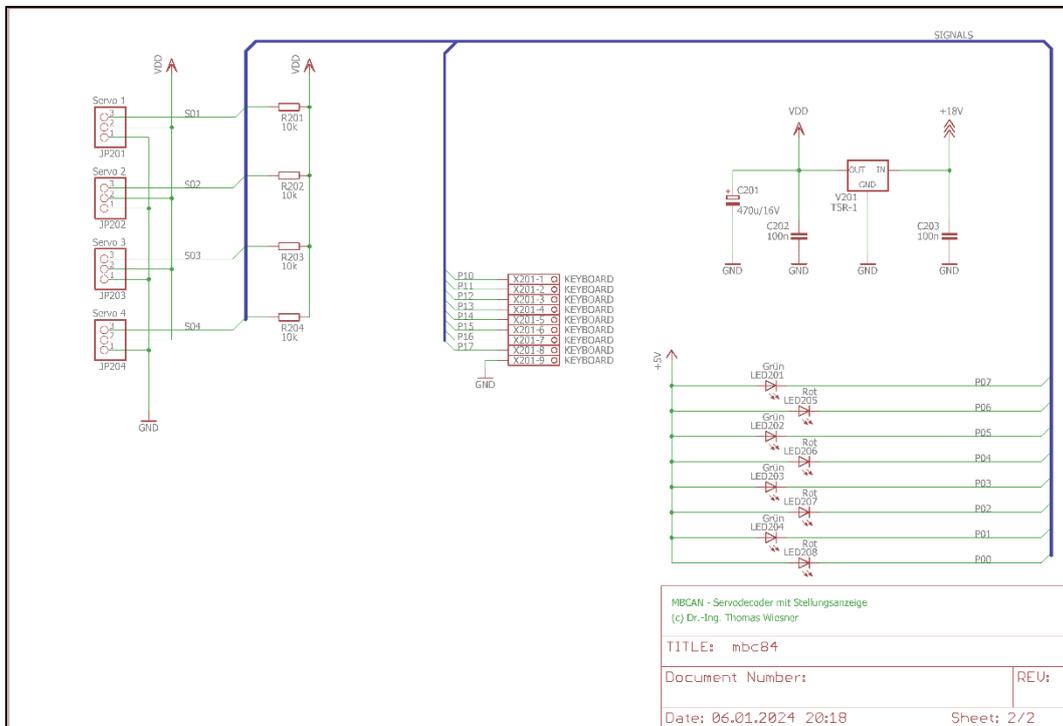


Abbildung 6-2: Sonderfunktionen

Das in Abbildung 6-1 gezeigte Schaltbild zeigt den bei allen Modulen identischen Prozessor-Kern mit Thermosensor DS 1820 als Seriennummer-Lieferant, einem externen EEPROM für die Upgrade-Fähigkeit und dem CAN-Bus-Interface nebst automatischer Terminierung des CAN-Bus.

Die Sonderfunktionen des Decoders bestehen aus den Steckverbindern für die Servos und den LED zur Stellungsanzeige sowie einer Schnittstelle zum Anschluss von Stellpulten oder Magnetschaltern für die nachträgliche Digitalisierung von Analoganlagen bzw. Direktsteuerungen.

## 7 Bestückung

Die Bestückung erfolgt wie üblich von den Bauteilen mit der geringsten Höhe (z.B. Widerstände) bis hin zu den höchsten Bauteilen (z.B. Stecker).

Ansonsten ist bei der Bestückung noch zu überlegen, ob der angegebene leistungselektronische Spannungsregler oder ein herkömmlicher Spannungsregler des Typs 7805T verwendet wird. Im letzteren Fall ist allerdings Kühlkörper vorzusehen. Der Platz kann je nach Kühlkörpertyp etwas beengt sein, hier ist auszuprobieren, welcher Kühlkörpertyp der geeignete ist. Außerdem sind dann zwei 100nF-Blockkondensatoren nachzurüsten, die Steckplätze sind vorgesehen. Dies soll Schwingneigungen entgegenwirken. Bei den Modulen des Herstellers werden grundsätzlich die etwas teureren leistungselektronischen Spannungsregler von TRACO verwendet, diese haben die Blockkondensatoren bereits integriert.

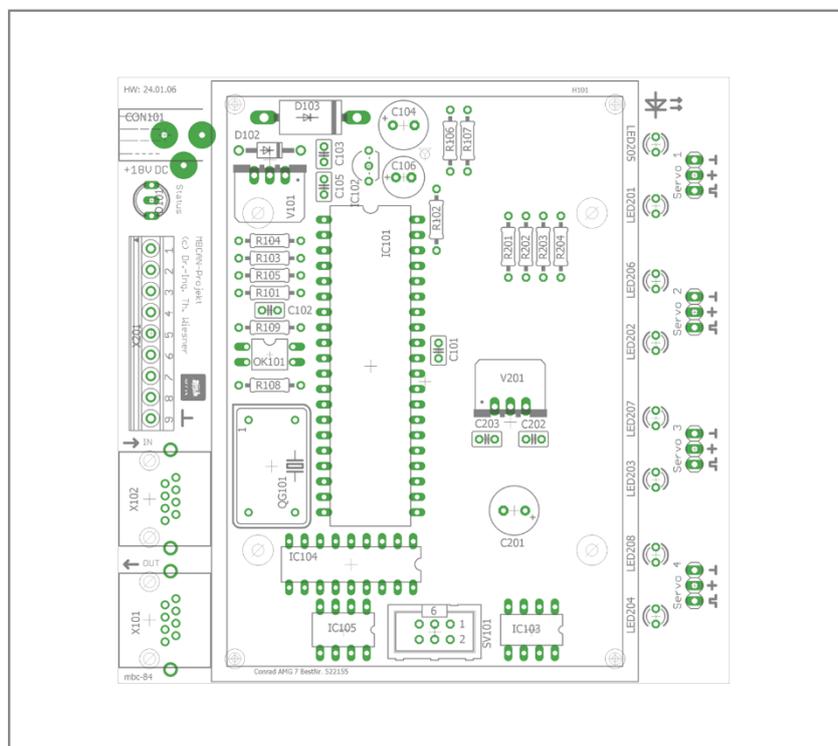


Abbildung 7-1: Bestückung Bauteilnummern

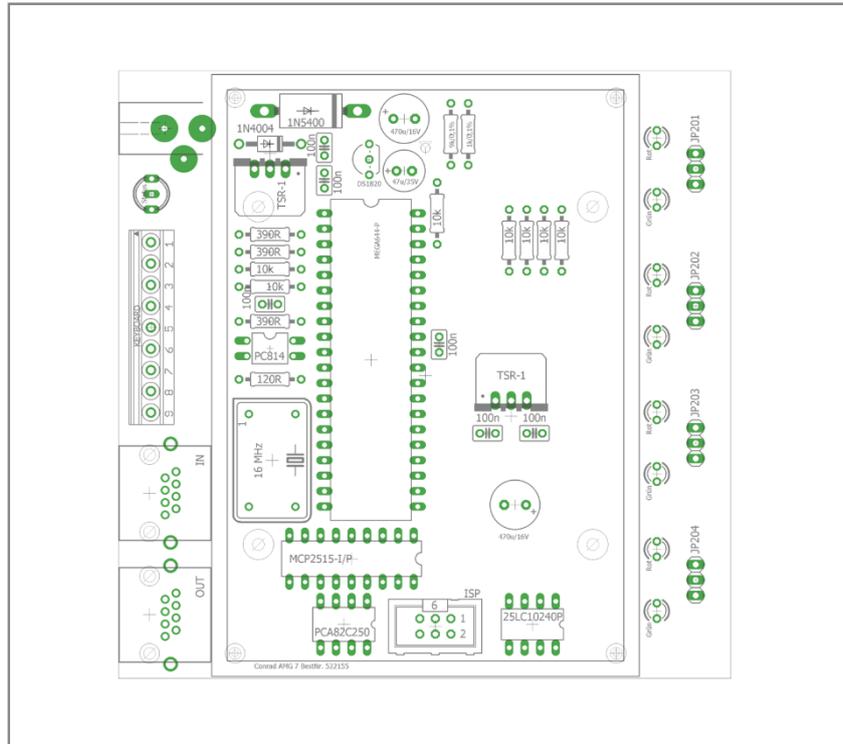


Abbildung 7-2: Bestückung Bauteilwert

## 8 Bauteileliste

Die für die Bestückung benötigten Bauteile sind in nachfolgender Tabelle aufgelistet. Ergänzt sind außerdem ein möglicher Lieferant sowie die zugehörige Bestellnummer. Der Lieferant ist nur ein Vorschlag und ist nicht bindend.

Table 8-1: Stückliste

Part	Value	Lieferant	Bestellnummer	Anzahl
C101 - C103, C105, C202, C203	100n	Reichelt	MKS02-63 100N	6
C104, C201	470u/16V	Reichelt	RAD 470/16	2
C106	47u/35V	Reichelt	RAD 47/35	1
D101	DUOLED R/G 5MM	Reichelt	LED 5 RG-3	1
D102	1N4004	Reichelt	1N 4004	1
D103	1N5400	Reichelt	1N 5400	1
IC101	MEGA644-P	Reichelt	ATMEGA 644P-20PU	1
IC102	DS1820	Reichelt	DS 18S20	1
oder	DS1820	Reichelt	DS 18B20	1
IC103	25LC1024	Reichelt	25LC1024-I/P	1
IC104	MCP2515-I/P	Reichelt	MCP 2515-I/P	1
IC105	PCA82C250	Reichelt	MCP 2551-I/P	1
LED201-204	LED 3mm G 5V	Reichelt	LED 3MM 5V GN	4
LED205-208	LED 3mm R 5V	Reichelt	LED 3MM 5V RT	4
V101, V201	TSR 1-2450	Reichelt	TSR 1-2450	2
OK101	PC814	Reichelt	LTV 814	1
QG101	QG5860	Reichelt	OSZI 16,000000	1
R101, R102, R105, R201 - R204	10k	Reichelt	METALL 10,0K	7
R108	120R	Reichelt	METALL 120	1
R103, R104, R109	390R	Reichelt	METALL 390	3
R106	9k/0,1%	Reichelt	MPR 9,10K	1
R107	1k/0,1%	Reichelt	MPR 1,10K	1
X101, X102	MBCAN	Reichelt	CAT5 T1U 2.8N4N	2
X201-1	AKL 059-04	Reichelt	AKL 059-04	1
X201-2	AKL 059-05	Reichelt	AKL 059-05	1
CON101	Power Jack	Reichelt	LUM NEB 21R	1
SV101	ISP	Reichelt	WSL 6G	1
JP201 - JP204	PINHEAD	Reichelt	MPE 087-1-016	1
ICS-8pol	Socket 8	Reichelt	GS 8P	2
ICS-18pol	Socket 18	Reichelt	GS 18P	1
ICS-40pol	Socket 40	Reichelt	GS 40P	1
Platine	mbc_84.brd	PCBPOOL	mbc_84.brd	1

## 9 Firmware

Die Firmware zum Modul kann entweder direkt onboard via ISP-Schnittstelle oder extern auf den Controller gebracht werden (vgl. Bestückung).

Entsprechende Dateien können von der Webseite heruntergeladen werden. Die Dateitypen sind dabei wie folgt zu unterscheiden:

- Dateien des Typs **mbc\_xx\_xx\_xx\_xx\_isp.hex** sind für die Erstprogrammierung zu verwenden und über die ISP-Schnittstelle aufzuspielen
- Dateien des Typs **mbc\_xx\_xx\_xx\_xx\_upgrade.hex** sind für das Upgrade über das Parametriercenter gedacht. Sie funktionieren NICHT bei der Programmierung über die ISP-Schnittstelle

Die korrekte Einstellung der FUSES ist Abbildung 9-1 zu entnehmen, wenn das AVR Studio verwendet wird.

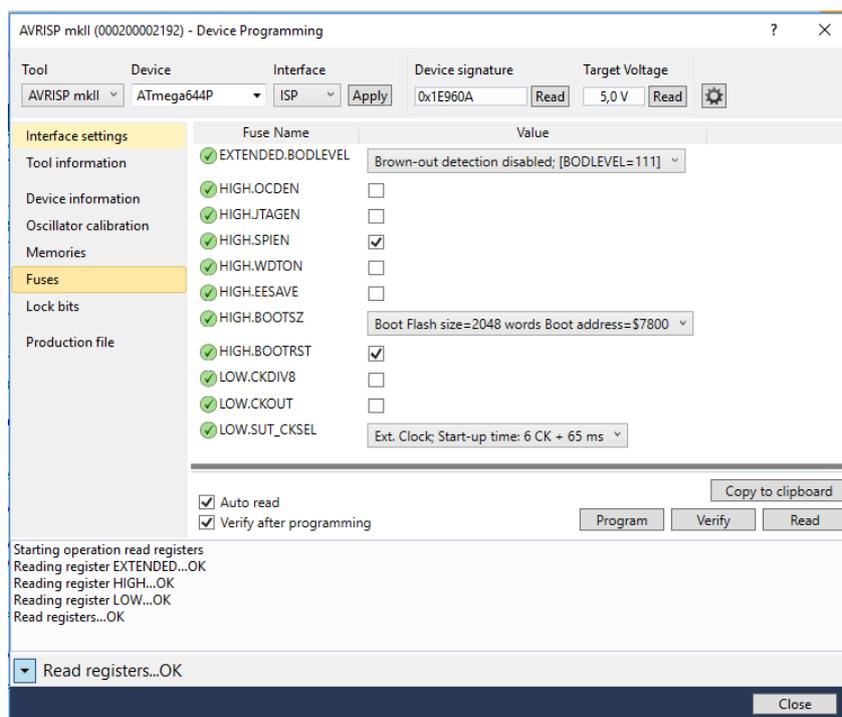


Abbildung 9-1: FUSES im AVR Studio

## 10 Steckverbindungen

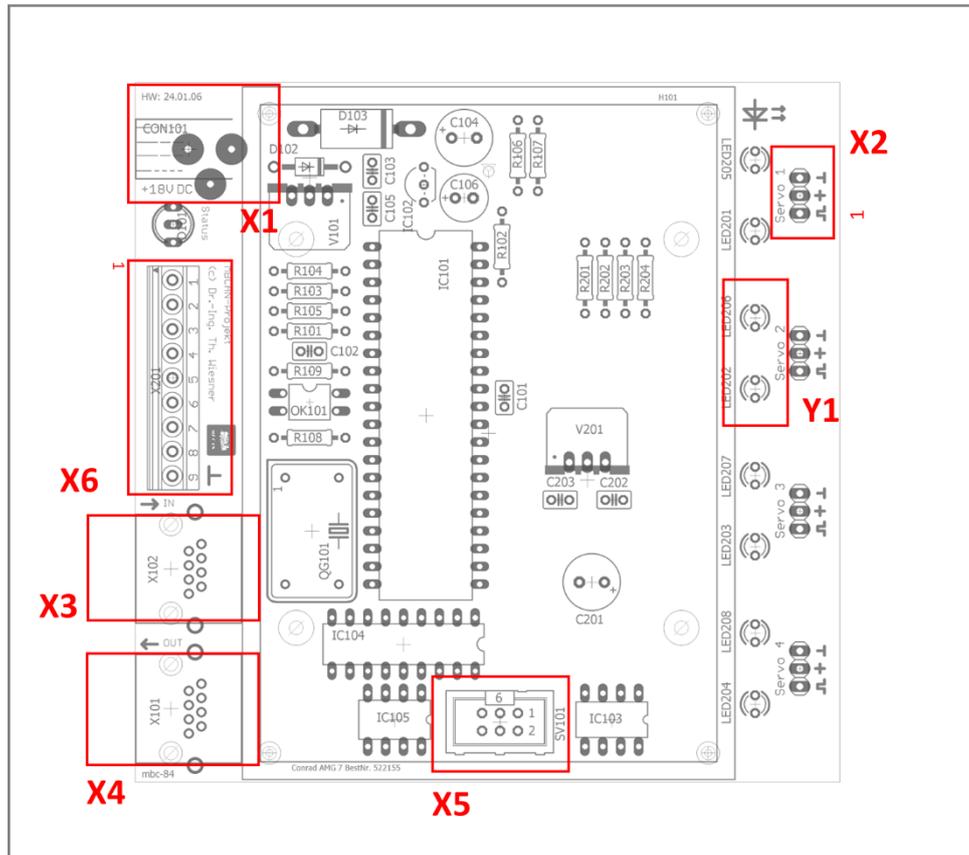


Abbildung 10-1: Steckverbinder

### X1 *dezentrale Spannungsversorgung*

Diese Buchse wird genutzt, um die Basis-Spannungsversorgung in den MBCAN-Bus einzuspeisen. Der Eingang ist verpolungssicher.

### X2 *Servoausgänge*

Hier werden die Servos angeschlossen. Pinbelegung:

1 = PWM-Signal, 2 = +5V DC, 3 = GND

### X3 *MBCAN-IN*

Modulverbindung zum nächsten Modul in der Kette über Cat.5-Netzwerkkabel.

### X4 *MBCAN-OUT*

Modulverbindung zum nächsten Modul in der Kette über Cat.5-Netzwerkkabel.

**X5** *Optionale ISP-Schnittstelle*

Programmierschnittstelle für Atmel-Programmieradapter. Wird nur zur initialen Installation oder im Falle eines Modulcrashes benötigt.

**X6** *Stellpult oder externe Kontaktgeber*

Diese Eingänge können zum Vor-Ort-Schalten der angeschlossenen Magnetartikel oder Verbraucher verwendet werden. Dies kann entweder über ein herkömmliches Stellpult, Kontaktgleise o.ä. erfolgen. Die Klemmenbelegung lautet:

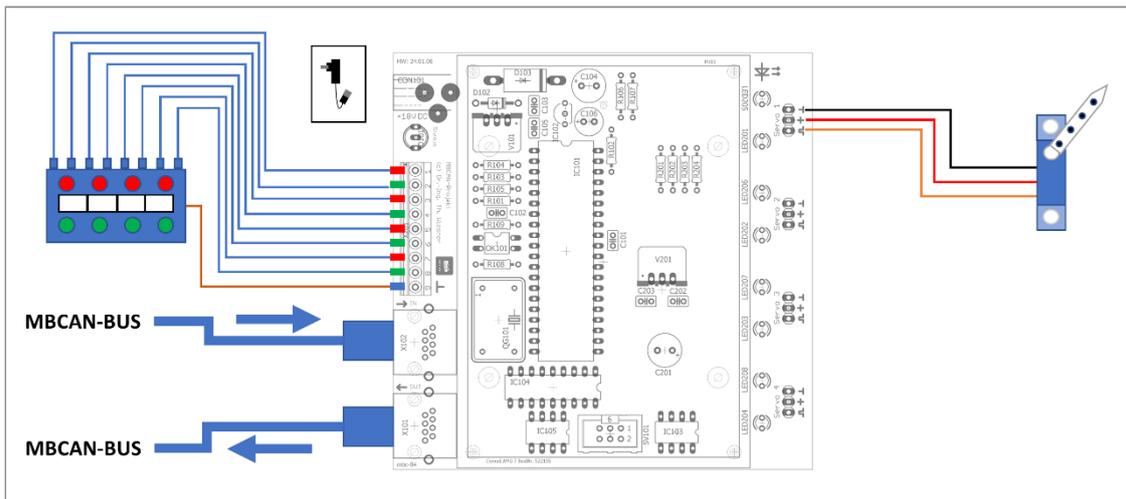
1 = Rot K1, 2 = Grün K1, 3 = Rot K2, ... 9 = Masse/Bezugspotenzial

**Y1** *Statusanzeige*

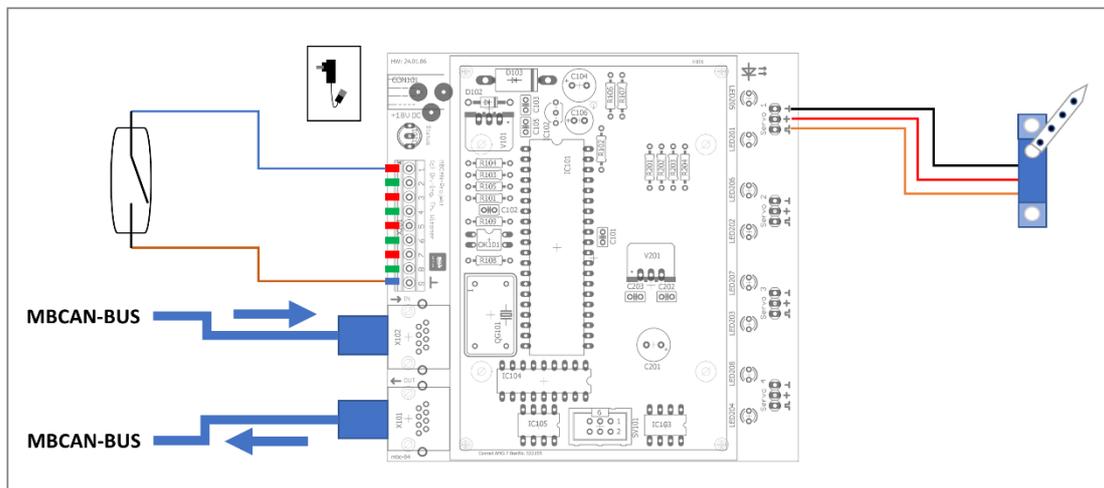
Zeigt die Stellung des Servos an. Die Drehrichtung ist abhängig vom Servotyp. Daher ist eine Zuordnung von Links/Rechts zu Rot/Grün nicht eindeutig möglich. Bitte vor Einbau ausprobieren.

## 11 Anschlussbeispiele

### Digitalisierung von analogen Anlagen



### Kontaktauslösung per REED-Relais oder Schaltgleis



## 12 Inbetriebnahme

### 12.1 Modul in Betriebsbereitschaft versetzen

Nach dem Start des Parametriercenters und dem Anlegen der Spannungsversorgung an die **Steckverbindung X1** meldet sich das Modul automatisch an, sobald eine Verbindungsart zum MBCAN-Bus ausgewählt wurde (siehe Bedienungsanleitung zum Terminaladapter mbc-80).

### 12.2 Modul konfigurieren

Nach erfolgreicher Anmeldung am Parametriercenter und Auslesen des Moduls (vgl. Bedienungsanleitung zum Parametriercenter) erscheint folgender Konfigurationsbereich:

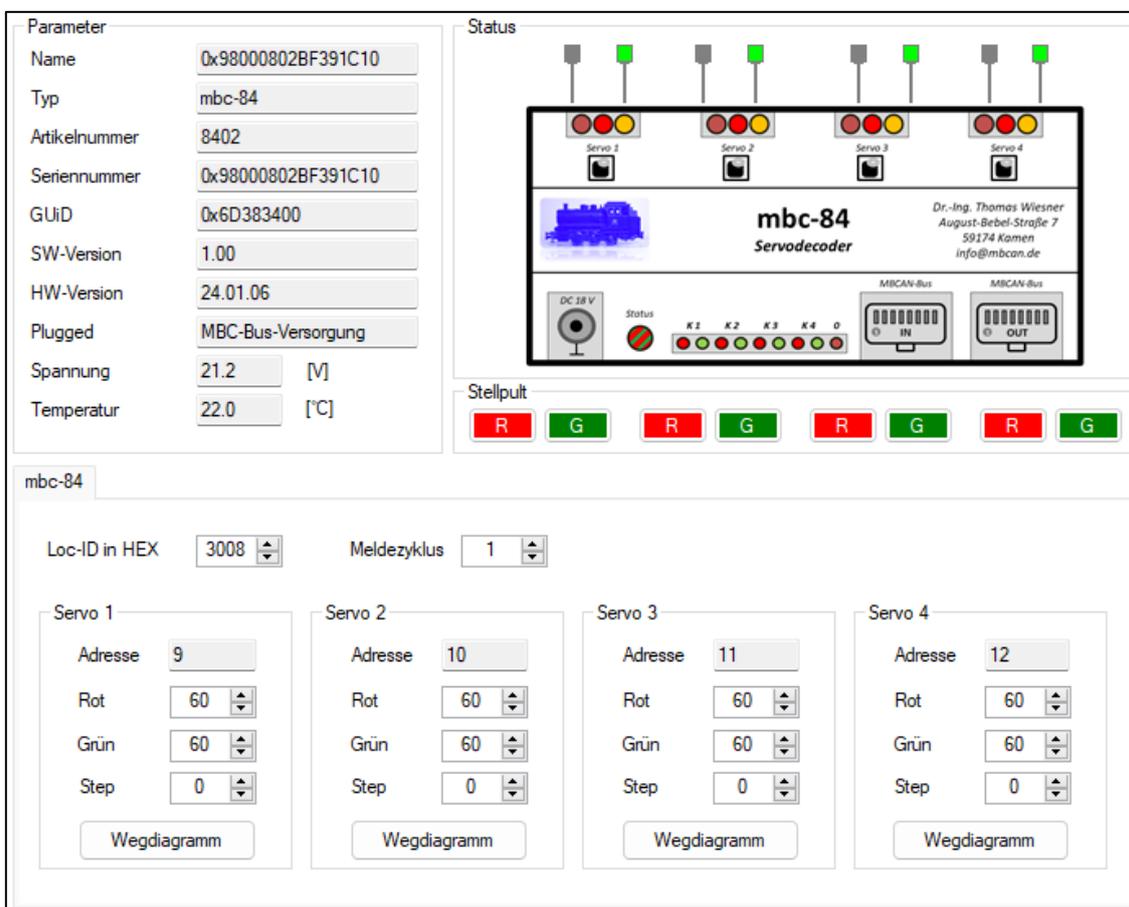


Abbildung 12-1: Konfigurationsbereich

Im obigen Beispiel wurde der Name des Moduls noch nicht geändert, hier wird die Seriennummer zunächst als Name verwendet. Er kann jederzeit über den Modul-Baum angepasst werden (siehe Bedienungsanleitung Terminaladapter mbc-80). Das Stellpult ist aktiviert.

Konfiguriert werden kann die **<Loc-ID>**, unter der das Modul seitens der CS2/3<sup>®</sup>/MS2<sup>®</sup> oder einer PC-Anwendung erreichbar ist. Angegeben wird hier nur die Basisadresse. Bei Veränderung der Loc-ID über die Up-/Down-Buttons wird der Wert um 4 (0x04) angepasst. Die auf der CS2/3<sup>®</sup>/MS2<sup>®</sup> korrespondierende Adresse des entsprechenden Servos kann wie folgt berechnet werden:

**Adresse Servo X = Loc-ID – 0x3000 + X für X = 1 ... 4**

Ergänzende Informationen siehe Protokollbeschreibung von Märklin®.

Das numerische Zählfeld **<Meldezyklus>** gibt an, ob sich das Modul zwecks Synchronisierung an die Zentrale zurückmelden soll oder nicht. Ein kompletter Meldezyklus dauert 256 s (entspricht die Aufforderung an ein Modul pro s). Somit wird der CAN-Bus nicht über Gebühr belastet und trotzdem eine Synchronisierung der unterschiedlichen Module sowie der Zentralen und der PC-Software gewährleistet.

Weitere Konfigurationen der einzelnen Servoausgänge sind die Anschläge der Stellrichtungen **<ROT>** und **<GRÜN>** sowie eine Schritverzögerung.

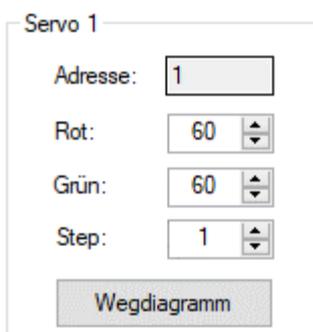


Abbildung 12-2: Servoparameter

Per Default stehen die Endanschläge auf 0, d.h. der Servo verharrt auch bei Richtungswechsel in der Mittenstellung. Dies soll verhindern, dass der Servo oder die daran angeschlossene Mechanik beschädigt wird, wenn unmittelbar in die maximale Ausrichtung angesteuert wird.

Die Anschläge lassen sich schrittweise bis maximal 60 (Zeitintervall 0,95ms) einstellen. Die Übergabe an das Modul erfolgt über den **<Update>**-Button. Es ergeben sich damit insgesamt jeweils 120 Werte zwischen dem linken und dem rechten Anschlag, der für eine Vielzahl von Anwendungen ausreichend sein dürfte.

Wenn Step auf 0 steht, wechselt das Servo bei Richtungsänderung über das Stellpult unmittelbar in den programmierten linken oder rechten Anschlag. Ein Step > 0 verzögert die Ausführung und lässt jeden einzelnen Schritt der 120 möglichen Schritte in verlangsamer Bewegung ausführen. Diese 120 Schritte sind unabhängig von den maximalen Ausschlägen und sind durch den verfügbaren internen EEPROM-Speicher des Moduls begründet (2 x 120 x 4 = 960 Bytes).

Wird ein Wert angepasst, wird das Feld *gelb* hinterlegt und alle Buttons deaktiviert. Dafür werden die Buttons **<Verwerfen>** und **<Update>** aktiviert. Mit **<Verwerfen>** kann die Eingabe rückgängig gemacht werden und der vorher gültige Wert wieder übernommen. Das Feld wird dann wieder in *weiß* hinterlegt. Mit dem Button **<Update>** wird der neue Parameter in das Modul geschrieben.

Die Bedeutung der anderen Buttons entnehmen Sie bitte der Bedienungsanleitung zum Terminaladapter mbc-80.

Der Weg zwischen den beiden Anschlägen ist per Default linear, kann aber geändert werden. Soll ein anderer Bewegungsablauf zwischen den Anschlägen ausgeführt werden, lässt sich über dem **<Wegdiagramm>**-Button ein individueller Weg programmieren.

Das in Abbildung 12-3 gezeigte Fenster erscheint nach Auswahl des genannten Buttons und bietet neben Einstellmöglichkeiten des Weges auch dessen grafischen Darstellung sowie die Programmiermöglichkeit des Moduls.

Jeder Servoweg von <ROT> nach <GRÜN> und umgekehrt ist jeweils in Einzelsteps von 120 mit maximal der Summe der beiden Maximalanschläge (hier auch 120) aufgelöst. Bei geringeren Maximalanschlägen entsprechend weniger, dies wird auch in der Grafik dargestellt. Die gelben Linien zeigen horizontal die Mittenstellung und vertikal die Drehrichtungsänderung an.

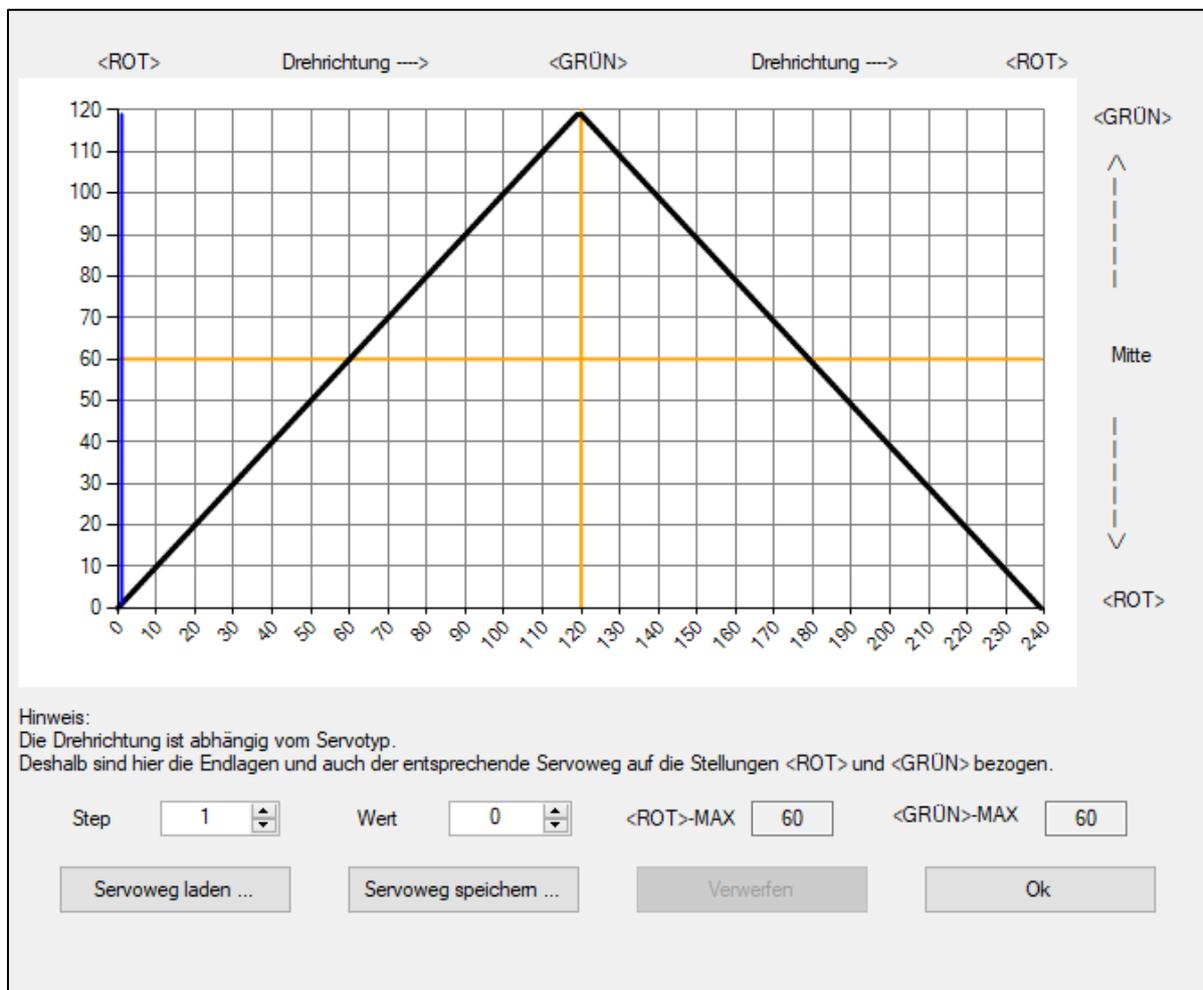


Abbildung 12-3: Programmierbare Servowege

Die Einstellung der einzelnen Steps erfolgt über das numerische Zählerfeld **<Step>**, der Ausschlag des Servos dann über das Feld **<Wert>**. Zur besseren Orientierung läuft ein blauer vertikaler Linealstrahl durch die Grafik; der vorhandene resp. der angepasste Wert des Ausschlags wird unmittelbar in der Grafik abgebildet.

Einmal erstellte Bewegungsabläufe lassen sich speichern oder laden. Es wird eine einfache Textdatei erstellt und im Ordner MBCAN beim entsprechenden User des PC unter **„Dokumente/MBCAN/Servos“** abgelegt. So lassen sich auch unabhängig vom Parametriercenter Servowege vorbereiten und auf einfache Weise importiert und an das Modul übergeben.

Soll die Änderung zurückgenommen werden, kann über den **<Verwerfen>**-Button der im Modul für den Ausgang hinterlegte Servoweg wieder ausgelesen werden. Andernfalls wird mit **<Update>** der gezeichnete Servoweg an das Modul übergeben und im EEPROM gespeichert.

Nachfolgend eine Abbildung, an der die Funktionalität erkennbar ist.

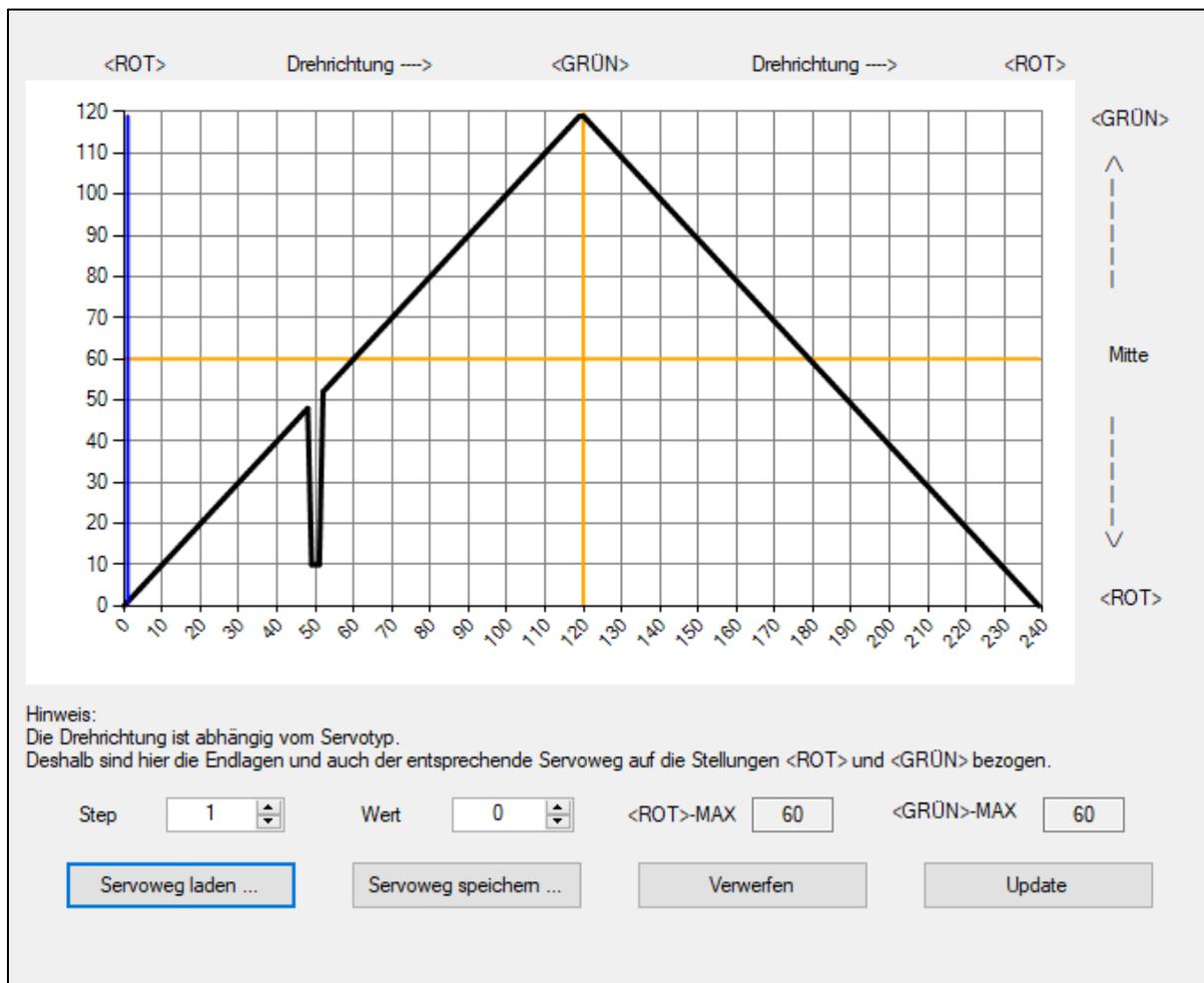


Abbildung 12-4: Angepasster Servoweg

#### HINWEIS:

Nach einer Neuprogrammierung über ISP sind alle Servos auf die Stellung <GRÜN> und <MITTELSTELLUNG> programmiert. D.h., wenn die Endlagen eingestellt werden, verstellt sich der Servo automatisch auf die Endlage <GRÜN>. Es sei denn, vorher wurde eine Stellungänderung auf <ROT> übermittelt via CAN oder Switchboard. Dann verstellt sich das Servo automatisch auf die neue Endlage <ROT>.

### 12.3 Modul mit der CS2® verbinden

Ist das Modul über den Terminaladapter mbc-80 mit der CS2® verbunden, meldet es sich über seine GUID an und ist sowohl im **<Info>-Bereich** als auch im **<Info>-Konfigurationsbereich** mit den Stammparametern anzeigbar. Im **<Info>-Bereich** werden die Artikelnummer, die Version in der Märklin®-Konvention, die aktuelle Spannung und die Gehäuseinnentemperatur des Moduls angezeigt, letztere dynamisch. Im **<Info>-Konfigurationsbereich** werden Artikelnummer, die Seriennummer, die Firm- und die Hardwareversion, angezeigt.

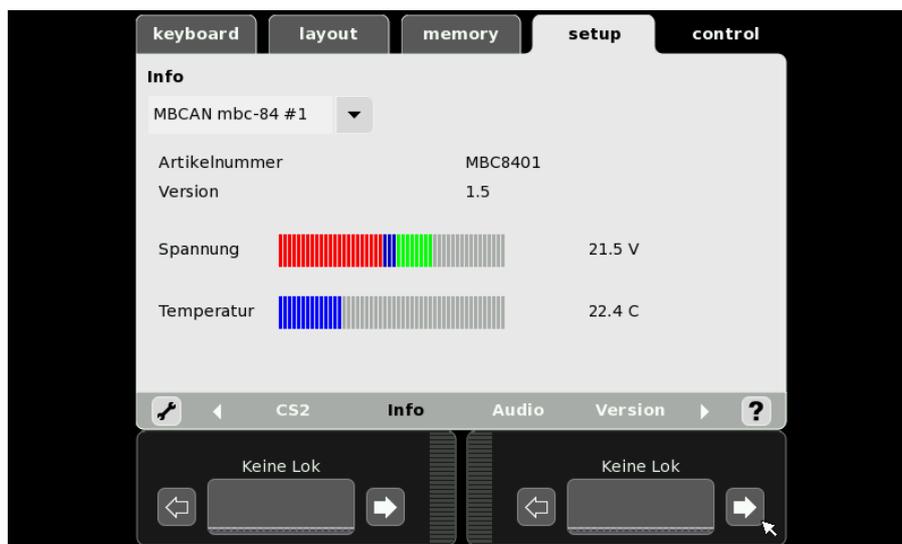


Abbildung 12-5: <Info>-Bereich



Abbildung 12-6: <Info>-Konfigurationsbereich

Der Button **<CAN-Test>** und das UpDown-Feld **<Ergebnisse>** haben keine besondere Funktion und werden standardmäßig in dieser Registerkarte angezeigt.

Die Zuordnung der Servos zum Modul erfolgt über die Konfigurationsmaske zum Layout.

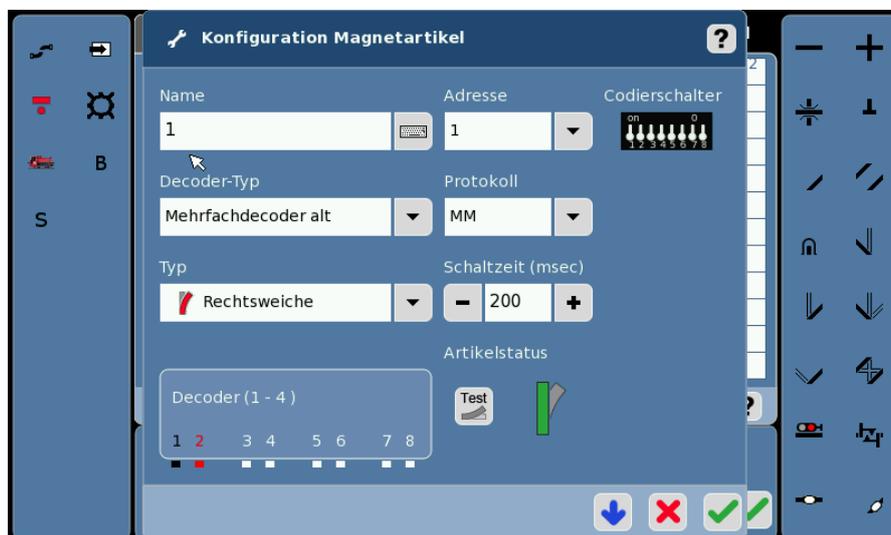


Abbildung 12-7: Zuordnung der Magnetartikel zum Modul

Der **<Name>** des Ausgangs kann vergeben werden. Als **<Protokoll>** ist auf jeden Fall „**MM**“ auszuwählen. Ansonsten werden die CAN-Nachrichten nicht im Loc-ID-Bereich von **0x3000** bis **0x33FF** gesendet. Die **<Schaltzeit>** spielt keine, da der Servoweg separat vorgegeben wird.

Die **<Adresse>** des Servos kann über folgende Gleichung der Modul-Loc-ID zugeordnet werden:

$$\text{Adresse Servo } X = \text{Loc-ID} - 0x3000 + X \text{ für } X = 1 \dots 4$$

Beispiele:

Loc-ID	Adresse Servo 1	Adresse Servo 2	Adresse Servo 3	Adresse Servo 4
0x3000	1	2	3	4
0x300C	13	14	15	16
0x3100	257	258	259	260
0x3120	289	290	291	292
0x33FC	1021	1022	1023	1024

## 12.4 Modul mit der CS3® verbinden

Ist das Modul über den Terminaladapter mbc-80 mit der CS3® verbunden, meldet es sich über seine GUID an und ist im **<System/Einstellungen>-Bereich** unter **<sonstige Geräte>** anzeigbar. Im **<Info>-Bereich** werden die Artikelnummer, die Version in der Märklin®-Konvention, die Seriennummer in der Märklin-Konvention (Modulnummer des Modultyps = letzte Zahl der GUID + 1), die aktuelle

Spannung und die Gehäuseinnentemperatur des Moduls angezeigt, letztere dynamisch. Im **<Einstellungen>-Bereich** werden der Name des Moduls (Nickname kann angepasst werden), die Seriennummer, die Firmware und die Hardwareversion in der MBCAN-Konvention angezeigt.

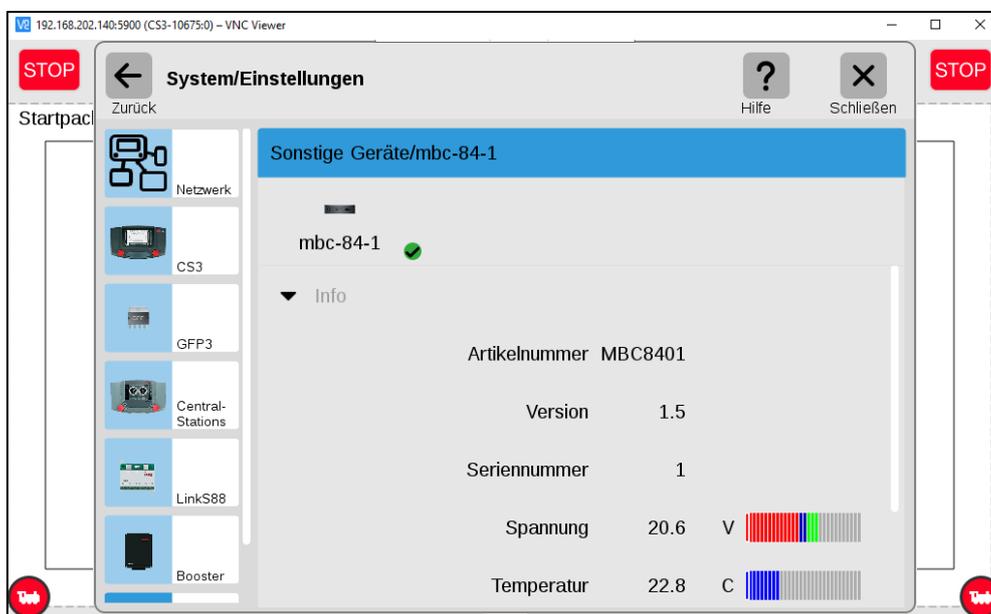


Abbildung 12-8: <Info>-Bereich

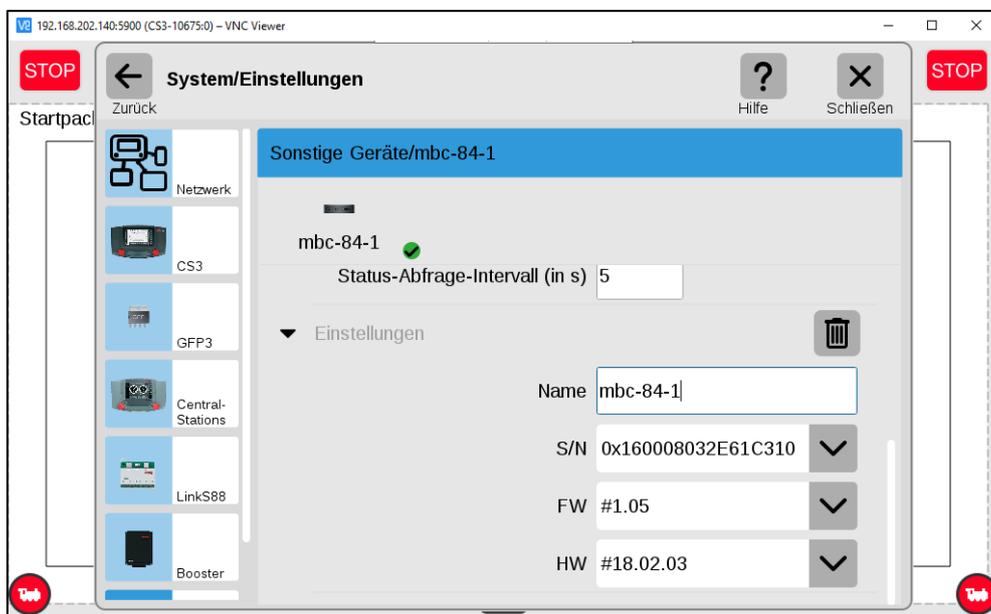


Abbildung 12-9: <Einstellungen>-Bereich

Das **<Status-Abfrage-Intervall>-Feld** regelt die Abfrage auf dem CAN-Bus zu diesem Modul. Es sollte auf dem vorgeschlagenen Wert belassen werden.

Die Zuordnung der Servos zum Modul erfolgt über die Konfigurationsmaske zum Layout.

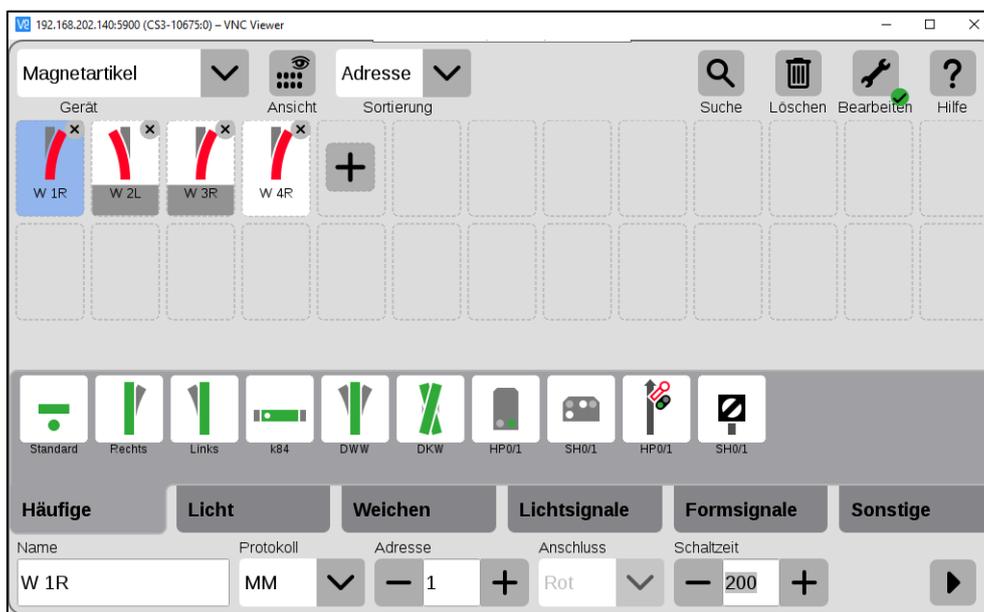


Abbildung 12-10: Zuordnung der Magnetartikel zum Modul

Der **<Name>** des Ausgangs kann vergeben werden. Als **<Protokoll>** ist auf jeden Fall „**MM**“ auszuwählen. Ansonsten werden die CAN-Nachrichten nicht im Loc-ID-Bereich von  $0x3000$  bis  $0x33FF$  gesendet. Die **<Schaltzeit>** spielt keine untergeordnete Rolle, da die Servowege separat vorgegeben werden.

Die **<Adresse>** des Servos kann über folgende Gleichung der Modul-Loc-ID zugeordnet werden:

$$\text{Adresse Servo } X = \text{Loc-ID} - 0x3000 + X \text{ für } X = 1 \dots 4$$

**Beispiele:**

Loc-ID	Adresse Servo 1	Adresse Servo 2	Adresse Servo 3	Adresse Servo 4
0x3000	1	2	3	4
0x300C	13	14	15	16
0x3100	257	258	259	260
0x3120	289	290	291	292
0x33FC	1021	1022	1023	1024

## 13 Modulbilder



Abbildung 13-1: Fertiges Modul inkl. Gehäuse

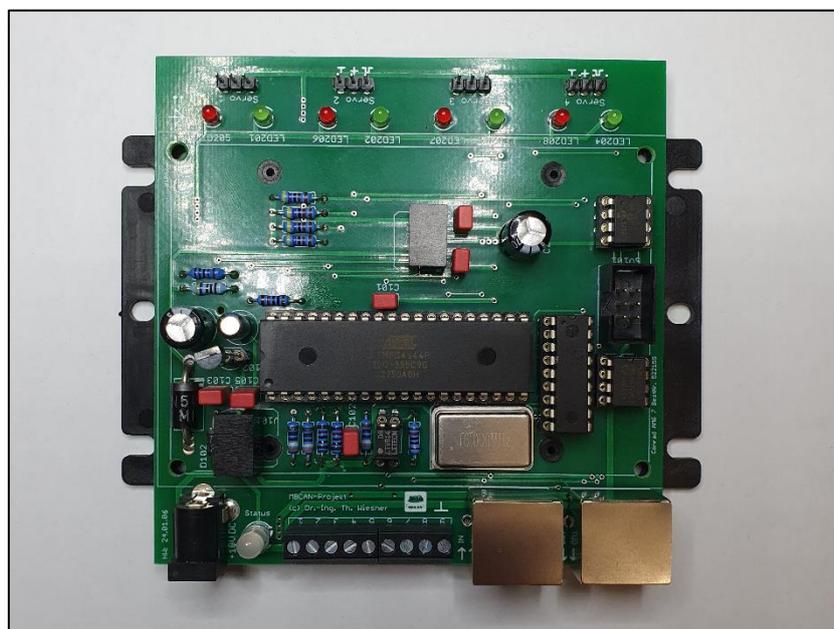


Abbildung 13-2: bestückte Platine

## 14 Systemarray-Belegung für Eigenentwicklungen

Nachfolgend ist die Belegung des Systemarrays abgebildet. Dies erleichtert bei Eigenentwicklungen von Software, die notwendigen Informationen des Moduls auslesen und parametrieren zu können.

### 14.1 Allgemeiner Bereich zum Modul

Der allgemeine Teil des Systemarrays ist bei allen mbc-Modulen gleich. Dargestellt ist die C-Schreibweise:

```
//=====
//= Systemarray mit Parametern zum Zustand =
//=====

// MBC_ARRAY_START          Start des belegten Systemarrays

#define MBC_ARRAY_START      1

// MBC_ALLG_START          Start des Allgemeinblocks

#define MBC_ALLG_START       MBC_ARRAY_START

// MBC_INFO_START          Start des Neuanmeldeblocks

#define MBC_INFO_START       MBC_ARRAY_START

// MBC_L_SNR               Laenge der Seriennummer
// MBC_S_SNR               Index Start Seriennummer
// MBC_SNR                  Seriennummer

#define MBC_L_SNR             8
#define MBC_S_SNR             MBC_ARRAY_START
#define MBC_SNR               sys_array[MBC_S_SNR]

// MBC_L_ART               Laenge der Artikelnummer
// MBC_S_ART               Index Start Artikelnummer
// MBC_ART_1               Artikelnummer Byte 4
// MBC_ART_2               Artikelnummer Byte 3
// MBC_ART_3               Artikelnummer Byte 2
// MBC_ART_4               Artikelnummer Byte 1

#define MBC_L_ART             4
#define MBC_S_ART             (MBC_L_SNR + MBC_S_SNR)
#define MBC_ART_1             sys_array[MBC_S_ART]
#define MBC_ART_2             sys_array[MBC_S_ART + 1]
#define MBC_ART_3             sys_array[MBC_S_ART + 2]
#define MBC_ART_4             sys_array[MBC_S_ART + 3]

// MBC_L_SW                Laenge Softwareversion
// MBC_S_SW                Index Start Softwareversion
// MBC_SW_1                Softwareversion Byte 3
// MBC_SW_2                Softwareversion Byte 2
// MBC_SW_3                Softwareversion Byte 1

#define MBC_L_SW              3
#define MBC_S_SW              (MBC_L_ART + MBC_S_ART)
#define MBC_SW_1              sys_array[MBC_S_SW]
#define MBC_SW_2              sys_array[MBC_S_SW + 1]
```

```
#define MBC_SW_3 sys_array[MBC_S_SW + 2]

// MBC_L_HW Laenge Hardwareversion
// MBC_S_HW Index Start Hardwareversion
// MBC_HW_1 Hardwareversion Byte 6
// MBC_HW_2 Hardwareversion Byte 5
// MBC_HW_3 Hardwareversion Byte 4
// MBC_HW_4 Hardwareversion Byte 3
// MBC_HW_5 Hardwareversion Byte 2
// MBC_HW_6 Hardwareversion Byte 1

#define MBC_L_HW 6
#define MBC_S_HW (MBC_L_SW + MBC_S_SW)
#define MBC_HW_1 sys_array[MBC_S_HW]
#define MBC_HW_2 sys_array[MBC_S_HW + 1]
#define MBC_HW_3 sys_array[MBC_S_HW + 2]
#define MBC_HW_4 sys_array[MBC_S_HW + 3]
#define MBC_HW_5 sys_array[MBC_S_HW + 4]
#define MBC_HW_6 sys_array[MBC_S_HW + 5]

// MBC_L_NAMEBLOCK Laenge des Modulnamens
// MBC_S_NAMEBLOCK Index Start Modulname
// MBC_NAME Name des Moduls

#define MBC_L_NAMEBLOCK 20
#define MBC_S_NAMEBLOCK (MBC_L_HW + MBC_S_HW)
#define MBC_NAME sys_array[MBC_S_NAMEBLOCK]

// MBC_INFO_ENDE Ende des Neuanmeldeblocks

#define MBC_INFO_ENDE (MBC_L_NAMEBLOCK + MBC_S_NAMEBLOCK - 1)

// MBC_L_GUID Laenge der GUID
// MBC_S_GUID Index Start GUID
// MBC_UiD_1 GUID Byte 4
// MBC_UiD_2 GUID Byte 3
// MBC_UiD_3 GUID Byte 2
// MBC_UiD_4 GUID Byte 1

#define MBC_L_GUID 4
#define MBC_S_GUID (MBC_L_NAMEBLOCK + MBC_S_NAMEBLOCK)
#define MBC_UiD_1 sys_array[MBC_S_GUID]
#define MBC_UiD_2 sys_array[MBC_S_GUID + 1]
#define MBC_UiD_3 sys_array[MBC_S_GUID + 2]
#define MBC_UiD_4 sys_array[MBC_S_GUID + 3]

// MBC_L_DB Laenge der Datenbankversion
// MBC_S_DB Index Start Datenbankversion
// MBC_DB Datenbanknummer des PC

#define MBC_L_DB 12
#define MBC_S_DB (MBC_L_GUID + MBC_S_GUID)
#define MBC_DB sys_array[MBC_S_DB]

// MBC_L_KN Laenge CS2-Geraetekennung
// MBC_S_KN Index Start CS2-Geraetekennung
// MBC_KN_H CS2-Geraetekennung HIGH
// MBC_KN_L CS2-Geraetekennung LOW
// MBC_AK_H CS2-Autokennung HIGH
```

```
// MBC_AK_L           CS2-Autokennung LOW
// MBC_CS2_GER        CS2-Geraetegruppe

#define MBC_L_KN       5
#define MBC_S_KN       (MBC_L_DB + MBC_S_DB)
#define MBC_KN_H       sys_array[MBC_S_KN]
#define MBC_KN_L       sys_array[MBC_S_KN + 1]
#define MBC_AK_H       sys_array[MBC_S_KN + 2]
#define MBC_AK_L       sys_array[MBC_S_KN + 3]
#define MBC_CS2_GER    sys_array[MBC_S_KN + 4]

// MBC_L_PARA         Laenge Parametersatz
// MBC_S_PARA         Index Start Parametersatz
// MBC_PLUGGED        Steckernetzteil gesteckt
// MBC_SPG_1          Spannung 10er Digit
// MBC_SPG_2          Spannung 1er Digit
// MBC_SPG_3          Spannung 0.1er Digit
// MBC_TMP_1          Temperatur 10er Digit
// MBC_TMP_2          Temperatur 1er Digit
// MBC_TMP_3          Temperatur 0.1er Digit

#define MBC_L_PARA     7
#define MBC_S_PARA     (MBC_L_KN + MBC_S_KN)
#define MBC_PLUGGED    sys_array[MBC_S_PARA]
#define MBC_SPG_1      sys_array[MBC_S_PARA + 1]
#define MBC_SPG_2      sys_array[MBC_S_PARA + 2]
#define MBC_SPG_3      sys_array[MBC_S_PARA + 3]
#define MBC_TMP_1      sys_array[MBC_S_PARA + 4]
#define MBC_TMP_2      sys_array[MBC_S_PARA + 5]
#define MBC_TMP_3      sys_array[MBC_S_PARA + 6]

// MBC_ALLG_ENDE      Ende des Allgemeinblocks

#define MBC_ALLG_ENDE  (MBC_L_PARA + MBC_S_PARA - 1)

// MBC_BOOT_START     Anfang des BOOT-Bereiches

#define MBC_BOOT_START (MBC_L_PARA + MBC_S_PARA)

// BOOT-Array-PAGE-Size Groesse der EEPROM-PAGE

#define MBC_BOOT_PGSZ  256

// Anzahl der Pakete fuer die BOOT-Array-Uebertragung

#define MBC_BOOT_PGMAX (MBC_BOOT_PGSZ / 4)

// MBC_L_BOOT         Laenge BOOT-Array
// MBC_S_BOOT         Index Start BOOT-Array
// MBC_BOOT           BOOT-Array

#define MBC_L_BOOT     (2 + MBC_BOOT_PGSZ)
#define MBC_S_BOOT     MBC_BOOT_START
#define MBC_BOOT_ARRAY sys_array[MBC_BOOT_START]

// MBC_BOOT_ENDE     Ende des BOOT-Bereiches

#define MBC_BOOT_ENDE  (MBC_L_BOOT + MBC_S_BOOT - 1)
```

```
// Je nach Modultyp ist das sys_array unterschiedlich lang. Alle Module
// haben einen gemeinsamen Teil mit einer Laenge von 69 Bytes und einem Boot-Block
// von 256 Bytes. Bei den anderen Modulen kommt noch ein General-Purpose-Array mit
// einer Laenge von 2048 Bytes hinzu, respektive 512 Bytes beim mbc-80.

// MBC_L_GENPURP           Laenge des General Purpose Arrays
// MBC_S_GENPURP           Index Start des General-Purpose-Arrays
// MBC_ARRAY_MAX           Laenge des Systemarrays

#ifdef _mbc_80_
    #define MBC_L_GENPURP           512

    #define MBC_S_GENPURP           (MBC_BOOT_ENDE + 1)
    #define MBC_ARRAY_MAX           (MBC_L_GENPURP + MBC_S_GENPURP)
#else
    #define MBC_L_GENPURP           2048
    #define MBC_S_GENPURP           (MBC_BOOT_ENDE + 1)
    #define MBC_ARRAY_MAX           (MBC_L_GENPURP + MBC_S_GENPURP)
#endif

// MBC_ARRAY_ENDE           Ende des Systemarrays

#define MBC_ARRAY_ENDE           (MBC_ARRAY_MAX - 1)
```

Listing 14-1: Modulparameter

Um die geänderten Parameter in das interne EEPROM zu schreiben, werden folgende Indizes verwendet:

```
//=====
//= Parameter fuer die EEPROM-Steuerung =
//=====

#define EE_NAME           0x01           // ID Block Name
#define EE_GUID           0x02           // ID Block GUID
#define EE_KENNUNG           0x03           // ID Block Geraetekennung
#define EE_DB           0x04           // ID Block Datenbank
```

Listing 14-2: EEPROM-Indizes Modul

## 14.2 Modulspezifischer Bereich für Funktionsparameter

Der modulspezifische Teil des Systemarrays beinhaltet die für die Funktion des Moduls vom Standard abweichenden Parameter. Diese liegen im GENPURP-Bereich des Systemarrays.

Neben der Loc-ID und der Aktorenstellung sind dies die Anschläge je Servo und Richtung, die Center-Stellung aller Servos sowie die Schritverzögerung für jeden Servo.

Dargestellt ist die C-Schreibweise:

```
//=====
//= Sonderbelegung des Systemarrays nach dem Allgemeinblock =
//=====

// Laenge modulspezifische Parameter
// Index Start modulspezifische Parameter
// MAddr HIGH
// MAddr LOW
// Stellung der Aktoren

#define MBC_84_L_MM          3
#define MBC_84_S_MM          MBC_S_GENPURP
#define MBC_84_MM_H          sys_array[MBC_84_S_MM]
#define MBC_84_MM_L          sys_array[MBC_84_S_MM + 1]
#define MBC_84_MM            sys_array[MBC_84_S_MM + 2]

// Laenge Maximalwerte Servos, Center und Schrittweite
// Index Start Maximalwerte Servos, Center und Schrittweite
// Servoanschlaege (Lx,Rx,Center,VZx)

#define MBC_84_L_SVMAX       13
#define MBC_84_S_SVMAX       (MBC_84_L_MM + MBC_84_S_MM)
#define MBC_84_SVMAX         sys_array[MBC_84_S_SVMAX]

// Laenge Servowege 120 Steps pro Servo und Richtung
// Index Start Servowege
// Servowege

#define MBC_84_L_SVWEG       960
#define MBC_84_S_SVWEG       (MBC_84_L_SVMAX + MBC_84_S_SVMAX)
#define MBC_84_SVWEG         sys_array[MBC_84_S_SVWEG]
```

Listing 14-3: Funktionsparameter

Um die geänderten Parameter in das interne EEPROM zu schreiben werden folgende Indizes verwendet:

```
//=====
//= Parameter fuer die EEPROM-Steuerung =
//=====

#define EE_84_MAddr          0x0A // ID Block Adresse
#define EE_84_AKTOR          0x0B // ID Block Aktorenstellung
#define EE_84_SERVOSMAX      0x0C // ID Block Servomaximalwerte
#define EE_84_SERVOSWEG      0x0D // ID Block Servowege
```

Listing 14-4: EEPROM-Indizes Funktionen

## 15 Befehlssatz zu den Modulen

Um die Module des MBCAN-Projektes auf dem CAN-Bus ansprechen und parametrieren zu können, bedarf es neben der Geräte-UID auch einen adäquaten Befehlssatz. Der Befehlssatz von Märklin® setzt sich aus Kommandos zusammen, die im CAN-Header integriert sind. Da dieser Header sehr sensibel auf Fehler reagiert, fällt er für eigene Befehlsübertragungen aus.

Märklin® hat aber eine Möglichkeit geschaffen, dass Privatpersonen, Vereine o.ä. freie Adressräume in der Loc-ID (Local ID, nicht Lokomotiv-ID) nutzen können. Diese liegen im Adressraum 0x00001800 bis 0x00001BFF (Datenbytes 1 bis 4 der CAN-Nachricht) und sind u.a. über das Schaltkommando 0x0B (= 0x16 im CAN-Header) verfügbar.

Der Befehlssatz von MBCAN baut auf diesem Adressraum und das Märklin®-Schaltkommando auf. Anders als bei Märklin® üblich, werden nur uni-direktionale Befehle generiert. D.h., dass das Response-Bit im CAN-Header nicht genutzt wird.

```
//=====
//= CAN-Befehlsnummern PC-Kommunikation initialisieren           =
//= Dieser ist der zweite Teil in der Addr der CAN-Nachricht 0x18xx =
//=====

// PC_DB_H           PC - Datenbanknummer HIGH
// PC_DB_M           PC - Datenbanknummer MID
// PC_DB_L           PC - Datenbanknummer LOW
// PC_KENNER         PC - Geraetekenner und Identifier
// PC_NEU            PC - Neuanmeldungsanforderung des PC
// PC_NEU_DATA       PC - Neuanmeldungs kanal des PC
// MD_NEU_DATA       MD - Neuanmeldungs kanal des Moduls
// PC_RESET          PC - Reset durch PC
// PC_MD_DEL         PC - Modul wurde aus Datenbank geloescht
// PC_ALIVE          PC - Alivemeldung durch PC angefordert
// MD_ALIVE          MD - Acknowledge des Moduls auf PC_ALIVE
// PC_ARRAY          PC - Anfordern, auf das Systemarray des Moduls
//                  zuzugreifen
// MD_ARRAY          MD - Acknowledge des Moduls auf PC_ARRAY
// PC_ARRAY_DATA     PC - Schreiben/Lesen und ggf. Wert und Systemarray-
//                  index uebergeben
// MD_ARRAY_DATA     MD - Ack des Moduls auf PC_ARRAY_DATA und Wert aus
//                  dem Systemarray uebergeben
// PC_UPGRADE        PC - Anfordern, auf das Systemarray des Moduls
//                  zuzugreifen
// MD_UPGRADE        MD - Acknowledge des Moduls auf PC_UPGRADE
// PC_UPGRADE_DATA   PC - Schreiben/Lesen und ggf. Wert und Systemarray-
//                  index uebergeben
// MD_UPGRADE_DATA   MD - Ack des Moduls auf PC_UPGRADE_DATA und Wert
//                  aus dem Systemarray uebergeben
// PC_BOOT           PC - Modul mit neuer Firmware starten
// MD_S88            MD - S88-Stellungsmeldung

#define PC_DB_H      0x00
#define PC_DB_M      0x01
#define PC_DB_L      0x02
#define PC_KENNER    0x03
#define PC_NEU       0x04
#define PC_NEU_DATA  0x05
```

```
#define MD_NEU_DATA      0x06
#define PC_RESET        0x07
#define PC_MD_DEL       0x08
#define PC_ALIVE        0x09
#define MD_ALIVE        0x0A
#define PC_ARRAY        0x0B
#define MD_ARRAY        0x0C
#define PC_ARRAY_DATA   0x0D
#define MD_ARRAY_DATA   0x0E
#define PC_UPGRADE      0x0F
#define MD_UPGRADE      0x10
#define PC_UPGRADE_DATA 0x11
#define MD_UPGRADE_DATA 0x12
#define PC_BOOT         0x13
#define MD_S88          0x14
```

*Listing 15-1: Befehlssatz der MBCAN-Module*

Die Nachrichten auf dem MBCAN-Bus zur Kommunikation der Module untereinander und zum Parametriercenter entsprechen wie beschrieben der Märklin®-Konvention mit einer Datenlänge von 8 Byte (vgl. UDP-Datenformat bei Kopplung mit der CS2®):

Beispiel: **00 16 5F 38 08 00 00 18 09 6D 38 34 01**

#### Übersetzung:

*PRIO: 0x00* = Normale Priorität der Nachricht

*KOMMANDO: 0x16* = Schaltkommando

*HASH: 0x5F38* = HASH des Senders aus der GUID gemäß Märklin®

*DLC: 0x08* = Länge der Nachricht

*Loc-ID: 0x00001809* = ALIVE-Anfrage (0x1800 als Basis und 0x0009 als Befehl MD\_ALIVE)

*GUID: 0x6D383401* = Anfrage an mbc-84 #1

Weitere Informationen zu den CAN-Nachrichten gemäß Märklin®-Konvention siehe Quellenangabe unten.

Nachfolgend sind die Befehle und ihre Funktionen aufgeführt.

## 15.1 PC\_DH\_H - Übertragung des Datenbanknamens mit 12 Bytes (1-4)

<b>Befehl</b>	PC_DB_H
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001800
<b>Funktion</b>	Übertragung des Datenbanknamens mit 12 Bytes (1-4)
<b>Beschreibung</b>	Bytes 1 bis 6 stellen das Datum, Bytes 7 bis 12 die Uhrzeit dar. Beispielstring: "070917235340" = am 07.09.2017 um 23:53:40 wurde die Datenbank erstellt. Die einzelnen Bytes werden in ASCII-Werte übersetzt und dann übertragen. Dieser String findet sich auch im Dateinamen der exportierten Datenbank aus dem Parametriercenter.
<b>Genutzte Datenbytes</b>	D0 – D3: Loc-ID 0x00001800
	D4 – D7: Bytes 1-4 des Datum-/Uhrzeit-Strings
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 00 30 37 30 39
<b>Antwort</b>	-/-

## 15.2 PC\_DH\_M - Übertragung des Datenbanknamens mit 12 Bytes (5-8)

<b>Befehl</b>	PC_DB_M
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001801
<b>Funktion</b>	Übertragung des Datenbanknamens mit 12 Bytes (5-8)
<b>Beschreibung</b>	Bytes 1 bis 6 stellen das Datum, Bytes 7 bis 12 die Uhrzeit dar. Beispielstring: "070917235340" = am 07.09.2017 um 23:53:40 wurde die Datenbank erstellt. Die einzelnen Bytes werden in ASCII-Werte übersetzt und dann übertragen. Dieser String findet sich auch im Dateinamen der exportierten Datenbank aus dem Parametriercenter.
<b>Genutzte Datenbytes</b>	D0 – D3: Loc-ID 0x00001801
	D4 – D7: Bytes 5-8 des Datum-/Uhrzeit-String
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 00 31 37 32 33
<b>Antwort</b>	-/-

### 15.3 PC\_DH\_L - Übertragung des Datenbanknamens mit 12 Bytes (9-12)

<b>Befehl</b>	PC_DB_L
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001802
<b>Funktion</b>	Übertragung des Datenbanknamens mit 12 Bytes (9-12)
<b>Beschreibung</b>	Bytes 1 bis 6 stellen das Datum, Bytes 7 bis 12 die Uhrzeit dar. Beispielstring: "070917235340" = am 07.09.2017 um 23:53:40 wurde die Datenbank erstellt. Die einzelnen Bytes werden in ASCII-Werte übersetzt und dann übertragen. Dieser String findet sich auch im Dateinamen der exportierten Datenbank aus dem Parametriercenter.
<b>Genutzte Datenbytes</b>	D0 – D3: Loc-ID 0x00001802
	D4 – D7: Bytes 5-8 des Datum-/Uhrzeit-Strings
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 00 35 33 34 30
<b>Antwort</b>	-/-

## 15.4 PC\_KENNER - Kenner und Identifier für die Module

<b>Befehl</b>	PC_KENNER
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001803
<b>Funktion</b>	Kenner und Identifier für die Module
<b>Beschreibung</b>	<p>Die Kennung der MBCAN-Module folgt strikt dem Format der Geräte-UiD von Märklin. In der GUID stellt die erste Stelle die Kennung dar. Zurzeit verwendet MBCAN die Kennung "m" (0x6D). Im Parametriercenter kann die Kennung angepasst werden, falls Märklin den Kenner "m" für seine eigene Module reklamiert.</p> <p>Darüber hinaus bekommt jedes Modul noch einen Identifier, mit dem es sich an der GUID der CS2/3 als „Sonstige Geräte“ anmelden kann. Zurzeit ist dies „AAAA“ (0xAAAA). Im Parametriercenter kann die Kennung angepasst werden, falls Märklin den Identifier "m" für seine eigene Module reklamiert. Ausgenommen sind die Module mbc-80 (Identifier 0x0040) und mbc-82 (Identifier 0x0000) die von Märklin fest vorgegeben sind. Diese Identifier sind in der Firmware der Module bereits fest integriert.</p>
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001803 D4: 0x00 D5: Kennung D6 – D7: Identifier
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 03 00 6D AA AA
<b>Antwort</b>	-/-

## 15.5 PC\_NEU - Neuanmeldeaufforderung

<b>Befehl</b>	PC_NEU
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001804
<b>Funktion</b>	Neuanmeldeaufforderung
<b>Beschreibung</b>	Zyklische Aufforderung an neu am MBCAN-Bus angeschlossene und noch nicht angemeldete Module, sich am Parametriercenter anzumelden. Dies gilt auch für Module, die über das Parametriercenter zurückgesetzt wurden.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001804
	D4 - D7: 0x00
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 04 00 00 00 00
<b>Antwort</b>	MD_NEU_DATA

## 15.6 PC\_NEU\_DATA - Rückmeldung PC an Modul während des Neuanmeldeprozesses

<b>Befehl</b>	PC_NEU_DATA
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001805
<b>Funktion</b>	Rückmeldung des PC an das Modul während des Neuanmeldeprozesses
<b>Beschreibung</b>	Der PC sendet das empfangene Seriennummer-Byte auf den MBCAN-Bus zurück als Quittierung. Das entsprechende Modul reagiert dann mit dem nächsten Byte der Seriennummer, alle anderen Module schalten in den Listen-Modus und reagieren erst nach einer weiteren PC_NEU-Nachricht, falls sie noch nicht erfolgreich angemeldet waren.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001805
	D4 - D6: 0x00
	D7: n-tes Byte xx der Seriennummer
<b>Nachricht</b>	00 16 5F 38 08 00 00 18 05 00 00 00 xx
<b>Antwort</b>	MD_NEU_DATA

## 15.7 MD\_NEU\_DATA - Meldung des Moduls während des Neuanmeldeprozesses

<b>Befehl</b>	MD_NEU_DATA
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x00001806
<b>Funktion</b>	Meldung des Moduls während des Neuanmeldeprozesses
<b>Beschreibung</b>	Wenn das Modul noch nicht am Parametriercenter angemeldet war, reagiert es mit dieser Nachricht an den PC. Es sendet sein erstes Byte seiner Seriennummer an den PC. Reagiert der PC mit der Nachricht PC_NEU_DATA mit exakt dem gleichen Byte, sendet es weitere Bytes seiner Seriennummer, bis entweder alle Bytes übertragen wurden (erfolgreiche Anmeldung) oder der PC gerade ein anderes Modul initiiert. Stimmt das Byte nicht überein, geht es in den Listen-Modus und wartet auf eine weitere PC_NEU-Nachricht.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001806
	D4 - D6: 0x00
	D7: n-tes Byte xx der Seriennummer
<b>Nachricht</b>	00 16 2B 17 08 00 00 18 06 00 00 00 xx
<b>Antwort</b>	PC_NEU_DATA

## 15.8 PC\_RESET - Durchführen eines Hardware-Resets auf dem Modul

<b>Befehl</b>	PC_RESET
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001807
<b>Funktion</b>	Durchführen eines Hardware-Resets auf dem Modul
<b>Beschreibung</b>	Über das Parametriercenter können Module gezielt einem RESET unterzogen werden. Die Identifizierung der Module geschieht über ihre GUID.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001807 D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01 00 16 5F 38 08 00 00 18 07 6D 38 34 01
<b>Antwort</b>	-/-

## 15.9 PC\_MD\_SEL - Modul aus Datenbank entfernen

<b>Befehl</b>	PC_MD_DEL
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001808
<b>Funktion</b>	Modul aus Datenbank entfernen
<b>Beschreibung</b>	Das Modul wurde aus der Datenbank entfernt und kann sich an dieser Datenbank auch nicht mehr neu anmelden. Wird in der Regel nur bei Modulen verwendet, die sich in der Datenbank befinden aber nicht mehr am Bus angeschlossen werden sollen. Wird nur einmal gesendet, wenn das Modul im Parametriercenter gelöscht wird. Ist das Modul nicht am Bus und wird nach einem Neustart der Software wieder am Bus angeschlossen, meldet es sich nicht mehr neu an, es sei denn, die Datenbank wird neu erstellt.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001808
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 5F 38 08 00 00 18 08 6D 38 34 01
<b>Antwort</b>	-/-

## 15.10 PC\_ALIVE - ALIVE-Abfrage

<b>Befehl</b>	PC_ALIVE
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001809
<b>Funktion</b>	ALIVE-Abfrage
<b>Beschreibung</b>	Zyklische Abfrage über die GUID, ob das betreffende Modul sich noch am MBCAN-Bus befindet. Es antwortet mit der Nachricht MD_ALIVE.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001809
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 5F 38 08 00 00 18 09 6D 38 34 01
<b>Antwort</b>	MD_ALIVE

### 15.11 MD\_ALIVE - ALIVE-Abfrage

<b>Befehl</b>	MD_ALIVE
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x0000180A
<b>Funktion</b>	ALIVE-Abfrage
<b>Beschreibung</b>	Zyklische Abfrage über die GUID, ob das betreffende Modul sich noch am MBCAN-Bus befindet. Es antwortet mit der Nachricht MD_ALIVE.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180A
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 2B 17 08 00 00 18 0A 6D 38 34 01
<b>Antwort</b>	-/-

## 15.12 PC\_ARRAY - Zugriff Systemarray anfragen

<b>Befehl</b>	PC_ARRAY
<b>Sender</b>	PC
<b>Loc-ID</b>	0x0000180B
<b>Funktion</b>	Zugriff Systemarray anfragen
<b>Beschreibung</b>	Der PC fragt über die GUID an, ob er auf das Systemarray des Moduls zugreifen darf.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180B
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 5F 38 08 00 00 18 0B 6D 38 34 01
<b>Antwort</b>	MD_ARRAY

### 15.13 MD\_ARRAY - Zugriff Systemarray freigegeben

<b>Befehl</b>	MD_ARRAY
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x0000180C
<b>Funktion</b>	Zugriff Systemarray freigegeben
<b>Beschreibung</b>	Antwort des durch die GUID im Befehl PC_ARRAY adressierten Moduls mit Freigabe des Zugriffs. Das Modul geht dann in die Wartestellung, alle anderen Module werden die folgenden Anfragen des PC nicht mehr aus. Ausgenommen sind Anfragen des PC außerhalb des Befehls PC_ARRAY_DATA.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180C
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 2B 17 08 00 00 18 0C 6D 38 34 01
<b>Antwort</b>	-/-

## 15.14 PC\_ARRAY\_DATA - Zugriff Systemarray freigegeben

<b>Befehl</b>	PC_ARRAY_DATA
<b>Sender</b>	PC
<b>Loc-ID</b>	0x0000180D
<b>Funktion</b>	Zugriff Systemarray freigegeben
<b>Beschreibung</b>	<p>Der PC stellt die Zugriffsanfrage. Dies kann entweder ein Lese- oder ein Schreibzugriff sein. Außerdem ist der Systemarray-Index enthalten, der gelesen oder beschrieben werden soll.</p> <p>Beispiel:                  D4 = 0 -&gt; Lesen, 1 -&gt; Schreiben                  D5 + D6 = Systemarray-Index                  D7 = zu schreibender Wert, bei lesendem Zugriff irrelevant</p> <p>Über den Index des Systemarrays wird außerdem das Ende einer Datenübertragung angezeigt. Liegt der Index über der Maximallänge des Systemarrays und entspricht es einem bestimmten Wert, wird die Wartestellung des Modus für weitere Datenübertragungen aufgehoben und alle anderen Module können wieder auf einen PC_ARRAY-Zugriff angesprochen werden.</p>
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180D D4: 0 -> Lesen, 1 -> Schreiben D5 - D6: Systemarray-Index D7: zu schreibender Wert, beim Lesen n.c.
<b>Nachricht</b>	Wert 0x0A an die Stelle 0x0001 im Systemarray schreiben 00 16 5F 38 08 00 00 18 0D 01 00 01 0A
<b>Antwort</b>	MD_ARRAY_DATA

### 15.15 MD\_ARRAY\_DATA - Antwort des Moduls auf Systemarray-Zugriff

<b>Befehl</b>	MD_ARRAY_DATA
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x0000180E
<b>Funktion</b>	Antwort des Moduls auf Systemarray-Zugriff
<b>Beschreibung</b>	Bei einem lesenden Zugriff übergibt das Modul auf D7 den Inhalt des Systemarrays, bei einem schreibenden Zugriff ist D7 irrelevant. Die anderen Datenbytes der Nachricht (D4 ... D6) sind identisch mit der Nachricht des PC.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180E
	D4: 0 -> Lesen, 1 -> Schreiben
	D5 - D6: Systemarray-Index
	D7: gelesener Inhalt des Systemarrays, beim Schreiben n.c.
<b>Nachricht</b>	Gelesener Wert 0x07 aus der Stelle 0x0108 im Systemarray
	00 16 2B 17 08 00 00 18 0E 00 01 08 07
<b>Antwort</b>	-/-

## 15.16 PC\_UPGRADE - Firmware-Upgrade

<b>Befehl</b>	PC_UPGRADE
<b>Sender</b>	PC
<b>Loc-ID</b>	0x0000180F
<b>Funktion</b>	Firmware-Upgrade
<b>Beschreibung</b>	Der PC fragt über die GUID an, ob er die Firmware des Moduls upgraden darf. Ist nur aktiv bei Modulen der 3. Generation und nicht gültig für die Module des Typs mbc-91.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x0000180F
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 5F 38 08 00 00 18 0F 6D 38 34 01
<b>Antwort</b>	MD_UPGRADE

### 15.17 MD\_UPGRADE - Firmware-Upgrade freigeben

<b>Befehl</b>	MD_UPGRADE
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x00001810
<b>Funktion</b>	Firmware-Upgrade freigeben
<b>Beschreibung</b>	Antwort des durch die GUID im Befehl PC_UPGRADE adressierten Moduls mit Freigabe des Zugriffs. Das Modul geht dann in die Wartestellung, alle anderen Module werden die folgenden Anfragen des PC nicht mehr aus. Ausgenommen sind Anfragen des PC außerhalb des Befehls PC_UPGRADE_DATA.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001810
	D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01
	00 16 2B 17 08 00 00 18 10 6D 38 34 01
<b>Antwort</b>	-/-

## 15.18 PC\_UPGRADE\_DATA - Schreibe Firmware

<b>Befehl</b>	PC_UPGRADE_DATA
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001811
<b>Funktion</b>	Schreibe Firmware
<b>Beschreibung</b>	Der PC übermittelt die Upgrade-Daten. Das Modul speichert diese in das externe EEPROM zur Vorbereitung der Neuprogrammierung. Die Daten werden PAGE-weise (je 64 Byte) vom Parametriercenter übertragen, so dass der BOOTLOADER hinterher die Daten aus dem externen EEPROM auch korrekt auslesen kann.
<b>Genutzte Datenbytes</b>	HASH: Laufende Nummer in der jeweiligen PAGE D0 - D3: Loc-ID 0x00001811 D4 - D7: 4 zu schreibende Bytes
<b>Nachricht</b>	Schreibe im laufenden Index 2 die Werte 0x01, 0x00, 0x01 und 0x0A fortlaufend in das externe EEPROM 00 16 03 02 08 00 00 18 11 01 00 01 0A
<b>Antwort</b>	MD_UPGRADE_DATA

### 15.19 MD\_UPGRADE\_DATA - Antwort des Moduls auf Schreibe Firmware

<b>Befehl</b>	MD_UPGRADE_DATA
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x00001812
<b>Funktion</b>	Antwort des Moduls auf Schreibe Firmware
<b>Beschreibung</b>	Das Modul antwortet mit der exakten Datenstruktur der gesendeten Nachricht und signalisiert damit, dass es die Upgrade-Daten im externen EEPROM gespeichert hat.
<b>Genutzte Datenbytes</b>	HASH: Laufende Nummer in der jeweiligen PAGE
	D0 - D3: Loc-ID 0x00001812
	D4 - D7: 4 zu schreibende Bytes
<b>Nachricht</b>	Schreibe im laufenden Index 2 die Werte 0x01, 0x00, 0x01 und 0x0A fortlaufend in das externe EEPROM
	00 16 03 02 08 00 00 18 12 01 00 01 0A
<b>Antwort</b>	-/-

## 15.20 PC\_BOOT - Modul neu Booten

<b>Befehl</b>	PC_BOOT
<b>Sender</b>	PC
<b>Loc-ID</b>	0x00001813
<b>Funktion</b>	Modul neu Booten
<b>Beschreibung</b>	Nach erfolgreicher Übertragung der neuen Firmware signalisiert der PC einen Hardwarereset des Moduls. Dies geschieht nicht über den Befehl PC_RESET, da vorher noch Identifier in das externe EEPROM gespeichert werden müssen die anzeigen, dass eine neue Firmware vorliegt.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001813 D4 - D7: GUID des Moduls
<b>Nachricht</b>	GUID mbc-84 #1 = 6D 38 34 01 00 16 5F 38 08 00 00 18 13 6D 38 34 01
<b>Antwort</b>	-/-

## 15.21 MD\_S88 - Stellungsmeldung mbc-88 / mbc-90

<b>Befehl</b>	MD_S88
<b>Sender</b>	Modul
<b>Loc-ID</b>	0x00001814
<b>Funktion</b>	Stellungsmeldung mbc-88 / mbc-90
<b>Beschreibung</b>	Sendet bei Statusänderung eines PINs die Stellung auf den MBCAN-Bus, so dass sowohl des Parametriercenter als auch andere Module diese ggf. weiterverarbeiten können. Ist ein Relikt aus den ersten beiden Generationen der MBCAN-Modulreihe und sollte bei Eigenentwicklungen durch Auswertung der 0x22/23-CAN-Kommandos von Märklin® ersetzt werden.
<b>Genutzte Datenbytes</b>	D0 - D3: Loc-ID 0x00001814
	D4 - D5: Modulnummer (BUS 1 1...31, BUS 2 32...62, BUS 3 63...93)
	D6: Kontaktnummer (1...16)
	D7: Stellung
<b>Nachricht</b>	Modul 16, Kontakt 2 hat Stellung 1
	00 16 2B 17 08 00 00 18 14 00 10 02 01
<b>Antwort</b>	-/-

## 16 Post-Code

Jedes Modul besitzt eine Dreifarb-LED zur Anzeige des Betriebsstatus. Dies ist notwendig, da die Module ansonsten ohne Bus-Verbindungen keine Möglichkeiten haben zu sagen "wie es ihnen gerade geht". Ähnlich dem Post-Code bei den PC, wo über Töne beim Booten die einzelnen Schritte bestätigt oder Fehler akustisch ausgegeben wurden, habe ich mir einen Licht-Code für die Dreifarb-LED einfallen lassen.

Die LED-Anzeige wird mit 500 ms getaktet und ist je Botschaft 7 s lang; d.h., dass im Grundsatz 6 Blinkschematas zu je einer der drei Farben ROT, ORANGE und GRÜN möglich sind, Mischungen mal ausgenommen. Die Farbe der LED sind drei Klassen von Botschaften resp. Stati zugeordnet:

*ROT: Fehler im Modul*    *ORANGE: Konfiguration des Moduls*    *GRÜN: Bestätigung von Prozessen*

Unregelmäßiges Aufflackern der orangenen LED-Farbe bei ansonsten grüner LED zeigt Datentrain auf dem CAN-Bus an bzw. während des Upgrades aus dem externen EEPROM entsprechende Schreib-/Lesezugriffe. Damit ist erkennbar, ob der auf dem Modul implementierte CAN-Baustein Nachrichten verarbeitet.

Stand heute sind folgende Post-Codes implementiert:

Table 16-1: LED-Signalbedeutung

<b>Normalbetrieb (kein Blinken)</b>												
												
<b>Neuanmeldung PC erfolgreich (1x grün blinken)</b>												
												
<b>Neuanmeldung CS2/3 erfolgreich (2x grün blinken)</b>												
												
<b>Modulupdate erfolgreich (3x grün blinken)</b>												
												
<b>BT-Schreiben erfolgreich (4x grün blinken)</b>												
												

<b>FW-Upgrade erfolgreich (5x grün blinken)</b>												
												
<b>MCP-CAN-Baustein defekt oder nicht vorhanden (1x rot blinken)</b>												
												
<b>Versorgungsspannung zu niedrig (2x rot blinken)</b>												
												
<b>Firmware-Upgrade abgebrochen, da Fehler beim Parsen des neuen Programms. Das im Controller gespeicherte Programm wird wieder ausgeführt. (4x rot blinken)</b>												
												
<b>Firmware-Upgrade hat einen allgemeinen Systemfehler erzeugt. Das Modul muss über die ISP-Schnittstelle komplett neu aufgesetzt werden. (5x rot blinken)</b>												
												
<b>Interne Firmware wird gestartet (nur bei Modulen mit Upgrade-Funktion) (1x orange blinken)</b>												
												
<b>Firmware-Upgrade - Probedurchlauf wird durchgeführt (2x orange blinken)</b>												
												
<b>Firmware-Upgrade - Programmierung des internen EEPROM wird durchgeführt (3x orange blinken)</b>												
												
<b>Modul konfiguriert die interne Hardware (10x orange blinken)</b>												
												

## 17 Quellenverzeichnis

Bei der Erstellung der Hard- und Software sowie der Dokumente und Texte zum MBCAN-Projekt sind u.a. folgende Fundstellen verwendet worden:

- [01] Märklin: „Kommunikationsprotokoll CAN transportierbar über Ethernet“, 2012
- [02] Märklin: „Einstieg in Märklin Digital“, 1994
- [03] Atmel: „ATMega644P - 8-bit AVR“, 2008
- [04] Microchip: „MCP2515 - Stand-Alone CAN Controller With SPI™ Interface“, 2003
- [05] Schmitt: „Mikrocomputertechnik mit Controllern der Atmel AVR-RISC-Familie“, 2008
- [06] Luis: „C/C++ - Das komplette Programmierwissen für Studium und Job“, 2004
- [07] CAN: „<http://www.kreatives-chaos.com/artikel/can>“
- [08] MM-Protokoll: „<http://home.snafu.de/mgrafe/Programme/Signalerzeugung - Froitzheim.pdf>“
- [09] Eagle: „<http://www.cadsoft.de>“
- [10] Microsoft: „<https://www.visualstudio.com/products/visual-studio-dev-essentials-vs>“
- [11] Atmel: „<http://www.atmel.com/microsite/atmel-studio/>“
- [12] Forum: „<http://www.mikrocontroller.net>“
- [13] Wolff: „HTML5 und CSS3 - Das umfassende Handbuch“, 2016
- [14] SelfHTML: „<https://wiki.selfhtml.org/wiki/CSS/Tutorials/Bildergalerie>“, 2018

## 18 Allgemeine Hinweise zum MBCAN-Projekt

Dies ist eine Dokumentation zu meiner privaten, nicht-kommerziellen Internetseite zum MBCAN-Projekt und dient ausschließlich der Darstellung meines Hobbys. Dazu gehören auch die dort zum Download angebotenen Dokumente und Softwarepakete.

Die Ausführungen beziehen sich auf die Internetpräsenz "mbcan.de".

### Herausgeber:



Dr.-Ing. Thomas Wiesner  
August-Bebel-Str. 7  
59174 Kamen  
eMail: [info@mbcan.de](mailto:info@mbcan.de)

### Haftungshinweis:

Die Inhalte der Internetpräsenz "mbcan.de", die Dokumentation, deren Inhalt sowie die Ideen dürfen nur für den privaten Gebrauch genutzt werden. Der Nachbau der gezeigten Schaltungen oder Anwendung der Software geschieht auf eigene Gefahr. Ich übernehme keine Haftung für eventuell durch die Anwendung entstandenen Sach-, Vermögens- oder Personenschäden.

### Copyrights:

Die auf den Internetseiten und in den Dokumenten ggf. verwendeten jeweiligen Warenzeichen sind Eigentum der jeweiligen Unternehmen. Alle ggf. damit verbundenen Rechte werden durch mich uneingeschränkt anerkannt.

Soweit nicht durch Copyrights Dritter geschützt, liegt das Copyright bei allen hier gezeigten Texten, Bildern, Schaltungen und Quellcode bei Dr.-Ing. Thomas Wiesner. Eine Verwendung auf anderen Webseiten oder jegliche andere Veröffentlichung, auch auszugsweise, wird hiermit ausdrücklich untersagt.

Kamen, 20.10.2024

gez. Dr.-Ing. Thomas Wiesner