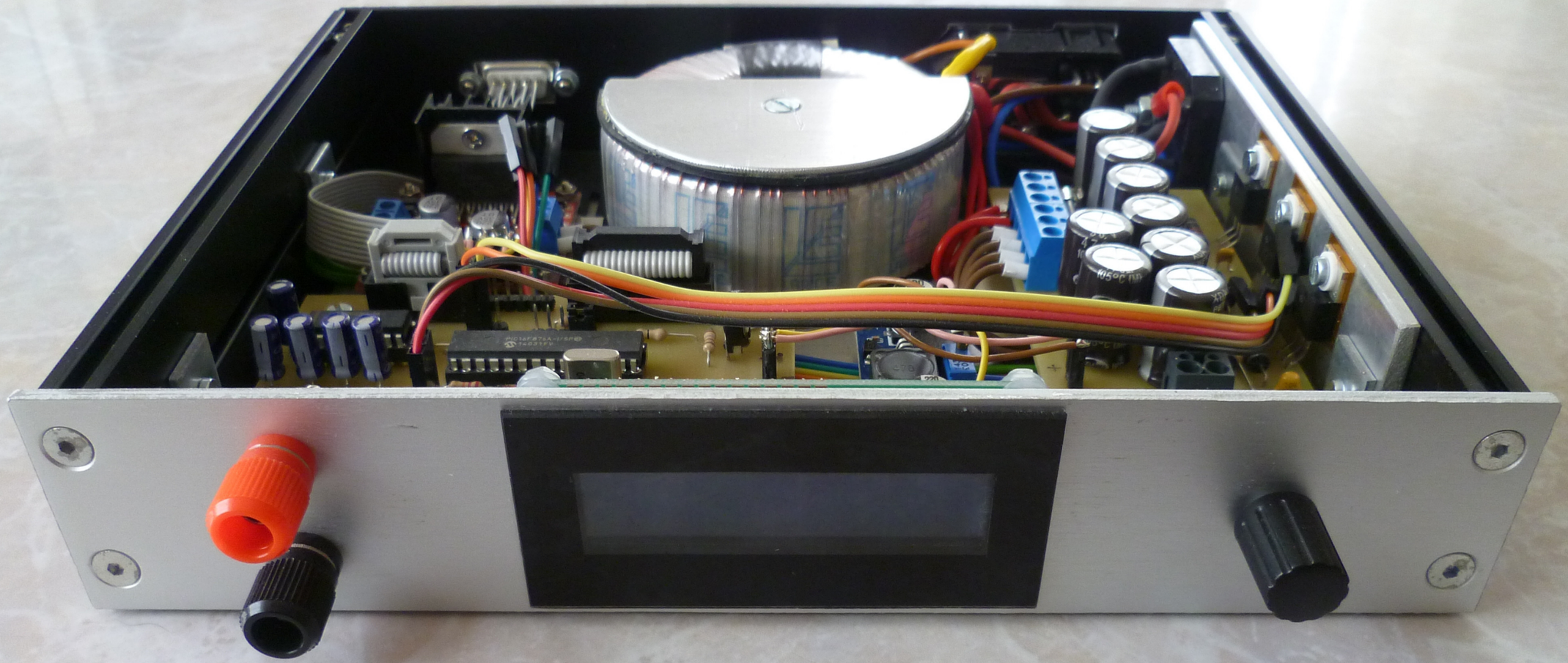
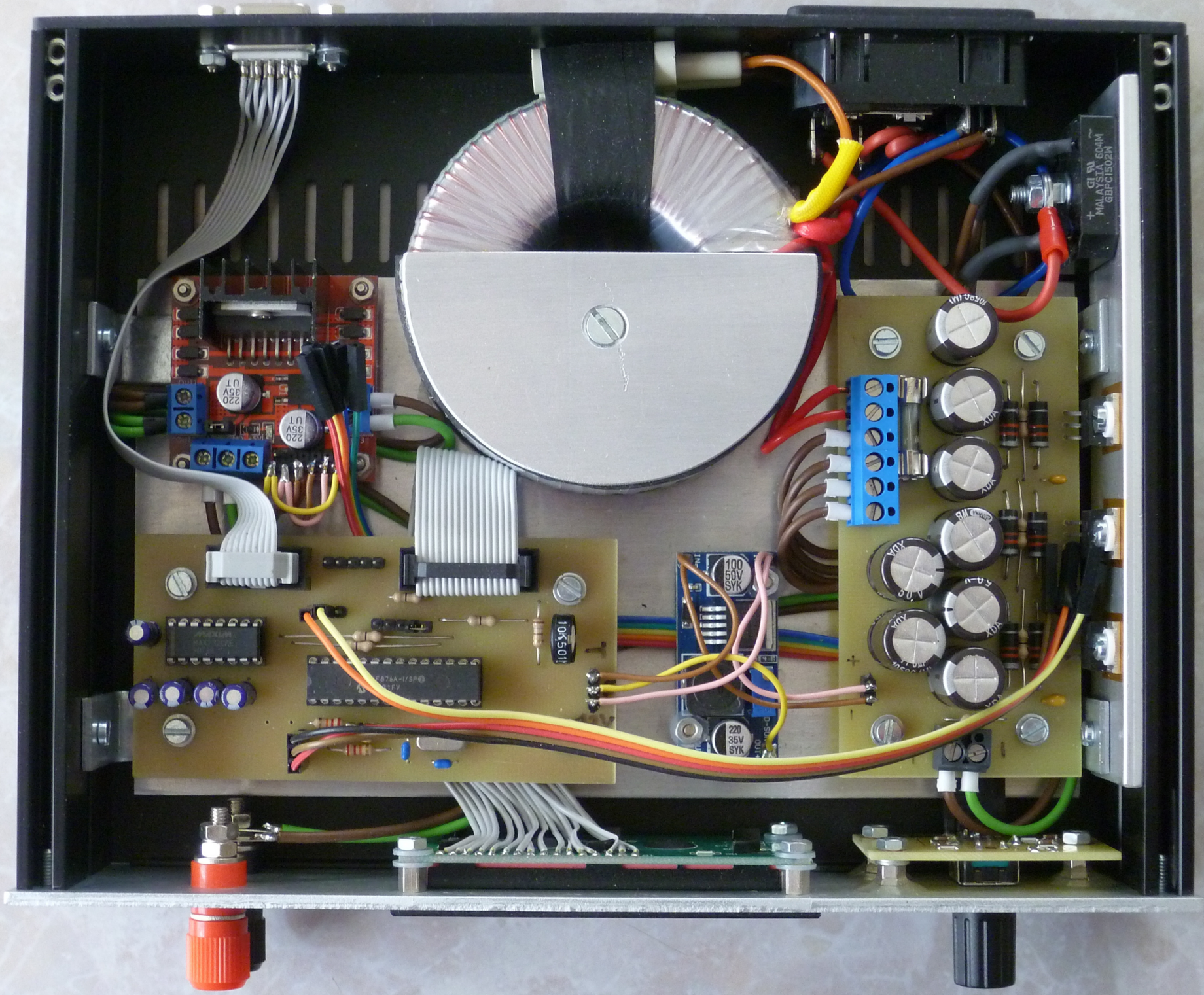
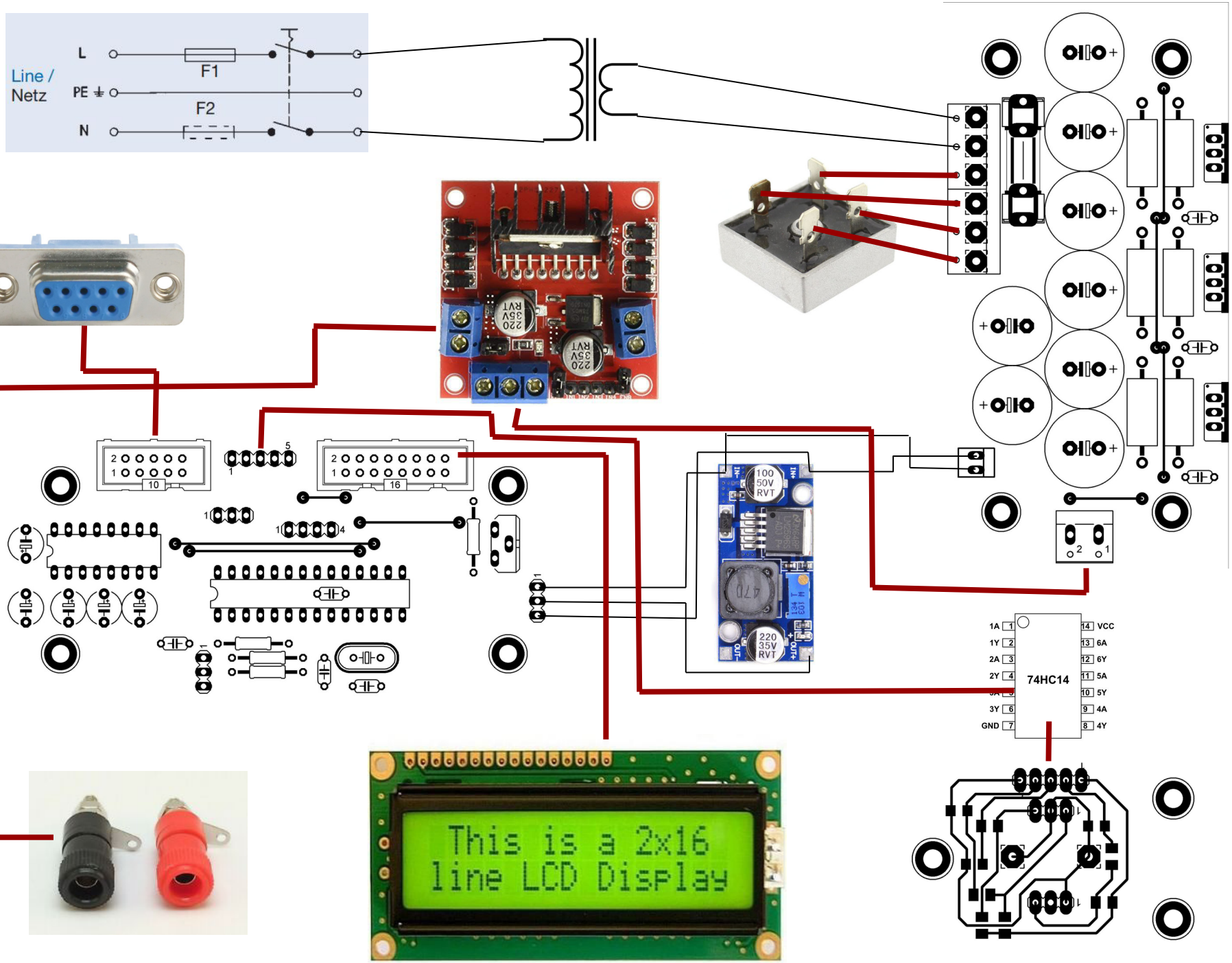
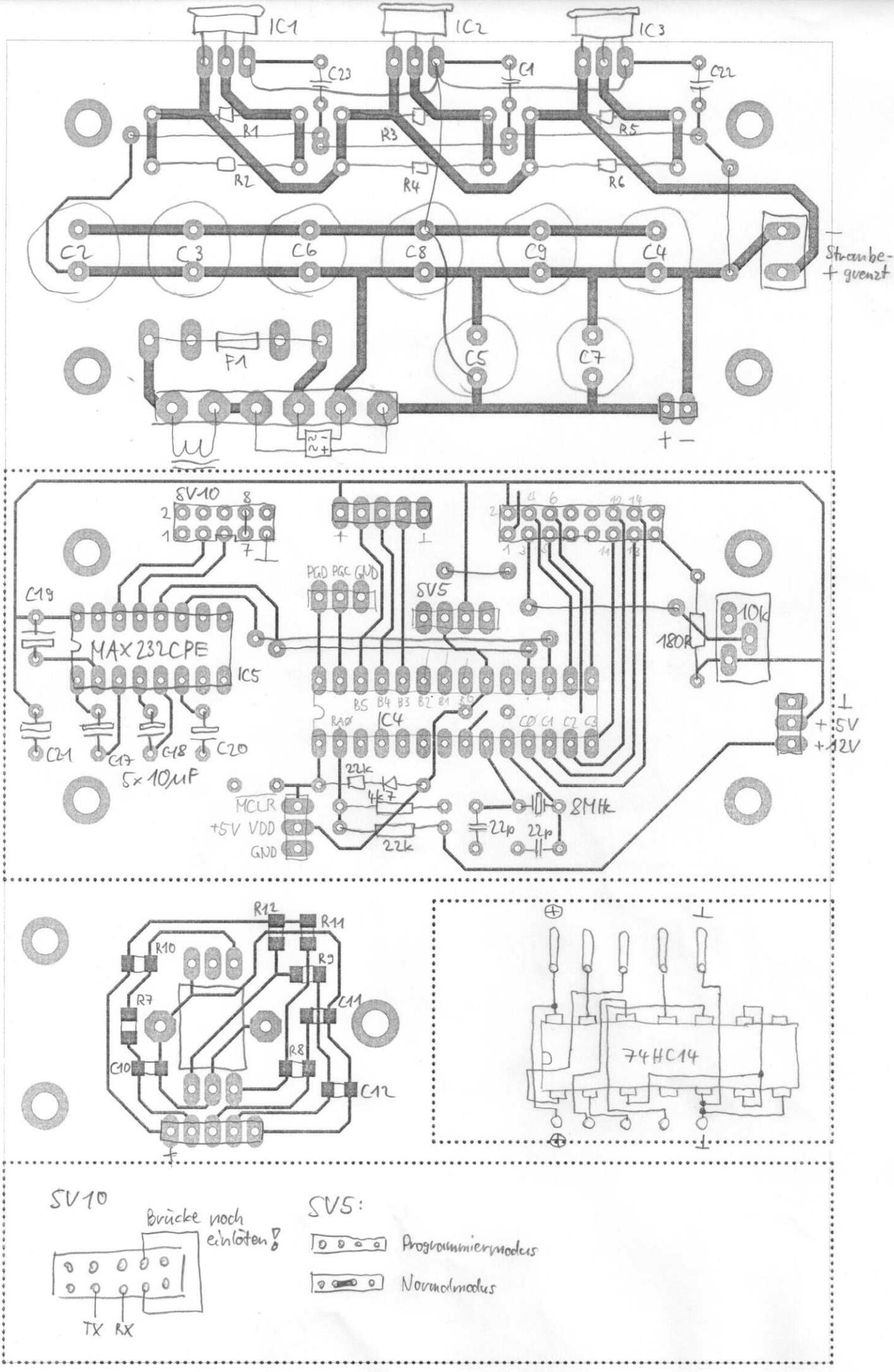
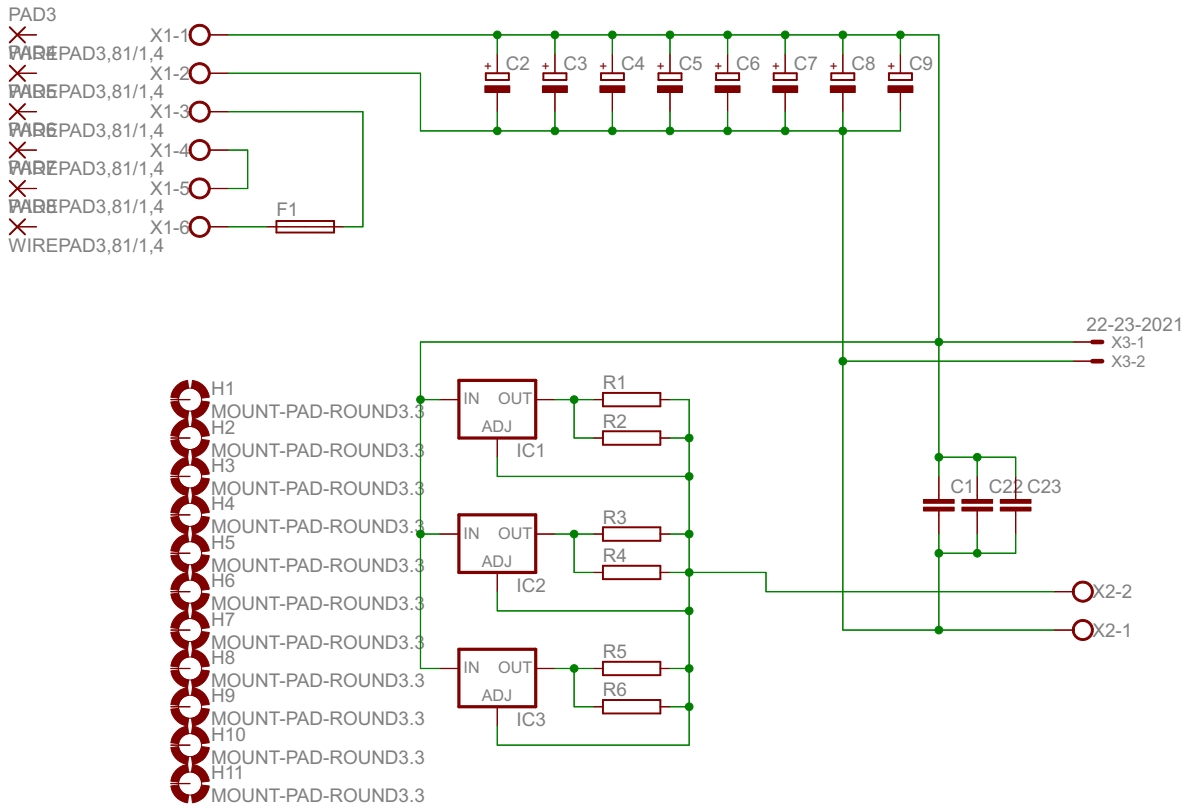
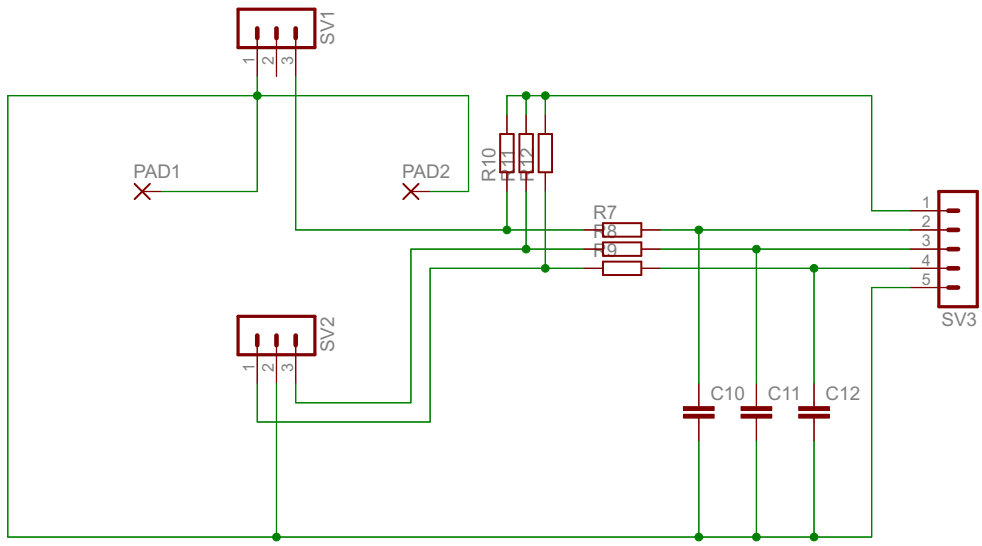


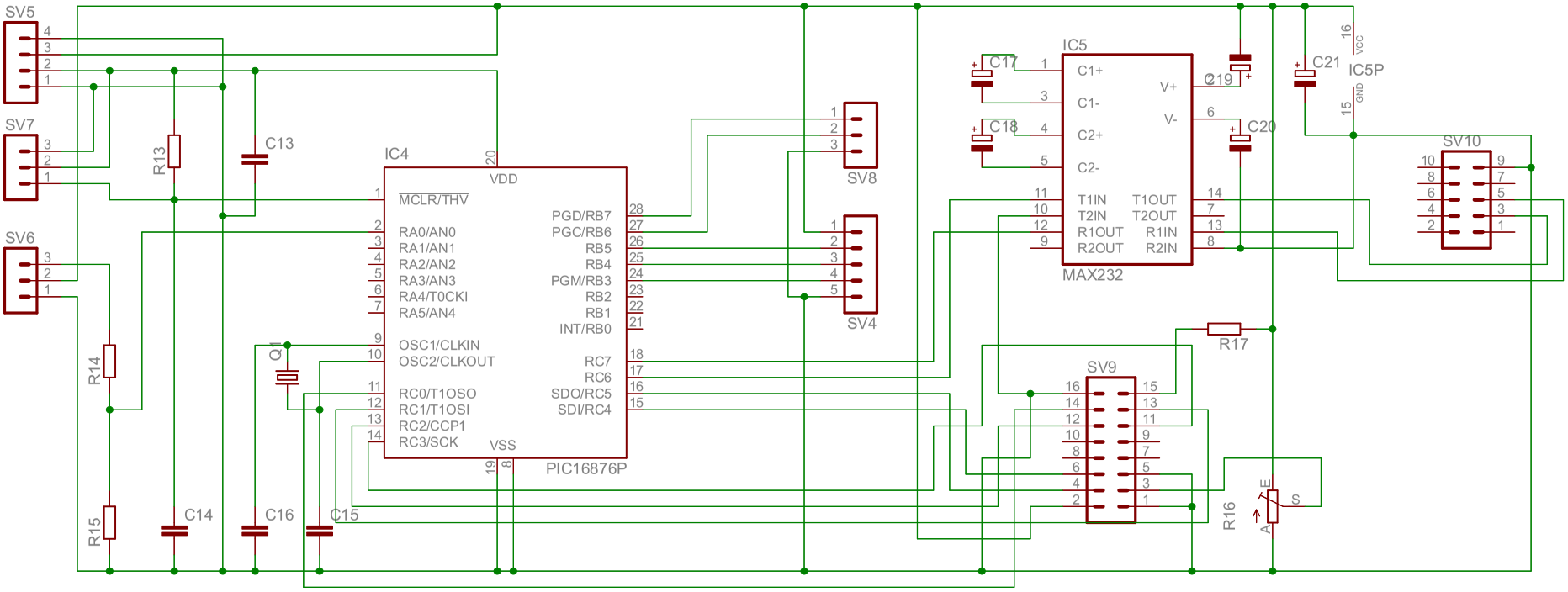

MALAYSIA 604M
GBPC1502W
220
35V
SYK
100
50V
SYK
10K50!


Line /
Netz
L
PE
N
F1
F2
100
50V
RVT
220
35V
RVT
74HC14
1A
1Y
2A
2Y
3Y
GND
VCC
6A
6Y
5A
5Y
4A
4Y
This is a 2x16
line LCD Display


IC1
IC2
IC3
C23
C1
C22
R1
R3
R5
R2
R4
R6
C2
C3
C6
C8
C9
C4
Strombe-
+ grenzt
P1
C5
C7
+ −
SV10
2
1
8
7
PGD PGC GND
SV5
C19
MAX232CPE
IC5
10k
180R
B5 B4 B3 B2 B1 B0
RA0
IC4
C0 C1 C2 C3
C21
C17
C18
C20
5x 10µF
22k
4k7
MCLR
+5V VDD
GND
22k
8MHz
22p
22p
⊥
+5V
+12V
R12
R11
R10
R9
R7
C11
R8
C10
C12
74HC14
SV10
Brücke noch
einlöten!
TX RX
SV5:
Programmiermodus
Normalmodus


PAD3
X1-1
X1-2
X1-3
X1-4
X1-5
X1-6
WIREPAD3,81/1,4
F1
C2
C3
C4
C5
C6
C7
C8
C9
22-23-2021
X3-1
X3-2
H1
H2
H3
H4
H5
H6
H7
H8
H9
H10
H11
MOUNT-PAD-ROUND3.3
IN
OUT
ADJ
IC1
IC2
IC3
R1
R2
R3
R4
R5
R6
C1
C22
C23
X2-2
X2-1


SV1
1
2
3
PAD1
PAD2
R10
R11
R12
R7
R8
R9
SV2
1
2
3
C10
C11
C12
1
2
3
4
5
SV3


SV5
SV7
SV6
R13
C13
IC4
VDD
MCLR/THV
RA0/AN0
RA1/AN1
RA2/AN2
RA3/AN3
RA4/T0CKI
RA5/AN4
OSC1/CLKIN
OSC2/CLKOUT
RC0/T1OSO
RC1/T1OSI
RC2/CCP1
RC3/SCK
VSS
PGD/RB7
PGC/RB6
RB5
RB4
PGM/RB3
RB2
RB1
INT/RB0
RC7
RC6
SDO/RC5
SDI/RC4
PIC16876P
SV8
SV4
C17
C18
IC5
C1+
C1-
C2+
C2-
V+
V-
T1IN
T2IN
R1OUT
R2OUT
T1OUT
T2OUT
R1IN
R2IN
MAX232
C19
C20
C21
IC5P
VCC
GND
SV10
R17
SV9
R16
E
S
A
R14
R15
C14
C16
C15
Q1

Sendebyte:	Bit:	Befehl:	Beschreibung:	
0	0	1	Synchronisationssequenz	
	1	1		
	2	1		
	3	1		
	4	1		
	5	1		
	6	1		
	7	1		
1	0	1		
	1	1		
	2	1		
	3	1		
	4	0	Startbit	Fahrbefehl
	5	0		
	6	A	Adresse	
	7	A		
2	0	A		
	1	A		
	2	A		
	3	A		
	4	A		
	5	0	Startbit	
	6	0	128 Fahrstufen	
	7	0		
3	0	1		
	1	1		
	2	1		
	3	1		
	4	1		
	5	1		
	6	0	Startbit	
	7	D	Richtung	
4	0	S		
	1	S		
	2	S		
	3	S		
	4	S		
	5	S		
	6	S		
	7	0	Startbit	
5	0	P	Prüfbyte	
	1	P		
	2	P		
	3	P		
	4	P		

Sendebyte:	Bit:	Befehl:	Beschreibung:	
	5	P		
	6	P		
	7	P		
6	0	1	Synchronisationssequenz	
	1	1		
	2	1		
	3	1		
	4	1		
	5	1		
	6	1		
	7	1		
7	0	1		
	1	1		
	2	1		
	3	1		
	4	0	Startbit	Schalten der Funktionen
	5	0		
	6	A	Adresse	
	7	A		
8	0	A		
	1	A		
	2	A		
	3	A		
	4	A		
	5	0	Startbit	
	6	1	Befehl	
	7	0		
9	0	B		
	1	B		
	2	B		
	3	B		
	4	B		
	5	B		
	6	0	Startbit	
	7	P	Prüfbyte	
10	0	P		
	1	P		
	2	P		
	3	P		
	4	P		
	5	P		
	6	P		
	7	1	Synchronisationssequenz	
	0	1		
	1	1		

Sendebyte:	Bit:	Befehl:	Beschreibung:	
11	2	1		
	3	1		
	4	1		
	5	1		
	6	1		
	7	1		
12	0	1		
	1	1		
	2	1		
	3	0	Startbit	Schaltdekoder
	4	0		
	5	A	Adresse	
	6	A		
	7	A		
13	0	A		
	1	A		
	2	A		
	3	A		
	4	0	Startbit	
	5	1	Funktion	
	6	0		
	7	F		
14	0	F		
	1	F		
	2	F		
	3	F		
	4	F		
	5	0	Startbit	
	6	P	Prüfbyte	
	7	P		
15	0	P		
	1	P		
	2	P		
	3	P		
	4	P		
	5	P		
	6	1	Synchronisationssequenz	
	7	1		

DCC-Laborbooster

Unterstützt momentan:
-nur 128 Fahrstufen (127, also eigentlich 126)
-Funktionen FL, F1, F2, F3 und F4
-nur 127 Weichenadressen

Menüstruktur:

Durch Drehen am Knopf blättert man durch's Haupmenü. Ein Druck auf den Knopf bringt einen in die Untermenüs. Nach dem letzten Untermenü kommt man zurück ins Haupmenü.

nach dem Einschalten:
-Spannung (die DCC-Spannung vor der H-Brücke)
    -ohne Untermenü

-Fahrstufe
    -0...127 (für DCC und Analog)
        -ENTER => zurück ins Haupmenü

-Funktion (Lokdekoder, Multifunktionsdekoder)
    -FL/F1/F2/F3/F4
        -AUS/EIN
            -ENTER => zurück ins Haupmenü

-Weichen (auch Schaltdekoder, Turnout, Weiche, Signalschaltbefehl, Magnetartikel oder Solenoid genannt)
    -Rot1/Grn1/Rot2/Grn2/Rot3...
        -AUS/EIN
            -ENTER => zurück ins Haupmenü

-Ausgang
    -DCC/AUS/PWM (AUS default aktiviert)
        -ENTER => zurück ins Haupmenü

-Einstellungen
    -Anl-Kor: (Analog-Korrektur)
        -1...127 (ein Offset für die PWM)
    -Wch-Mod: (Weichen-Modus)
        -IB (Intellibox-Modus, Weichen-Adressen bekommen Offset von 4)
        -Roco (Weichen-Adressen werden normal gesendet)
    -Wch-Pls: (Weichen-Puls)
        -1...32 (wie oft/lange der Weichenbefehl gesendet wird, 5 entspricht IB)

-Richtung
    -Vorwärts/Rückwärts (DCC und Analog)

-Info
    -kurze Informationen zur Software (Version)

-wieder zurück zu Spannung

Ueber RS232 kann ferngesteuert werden, die Befehle sollten weitgehend kompatibel zu den P50Xa-Kommandos (ASCII-Commands) sein. Siehe:
https://www.opendcc.de/elektronik/opendcc/opendcc_doc_ib.html
Implementiert sind momentan:

XL:{Lok#,[Speed],[FL],[Dir],[F1],[F2],[F3],[F4]}
XF:{Lok#,[F1],[F2],[F3],[F4],[F5],[F6],[F7],[F8]}
XT:{Trnt#,[Color],[Status]}
Dir: 0 oder r = rückwärts, 1 oder f = vorwärts
Color: 0 oder 'r' = Rot / Abzweig, 1 oder 'g' = Grün / Geradeaus
Status: 1 = ein, 0 oder nicht angegeben = aus
XS
XG

Die Uebertragungsrate ist 9600Baud, 8N2 und sollte somit zur IB sowie zu Intellibox-Manager kompatibel sein. Siehe:
http://greuphone.magnetmotor.ch/#ibm

noch zu Machen:

```
70  -Stringlänge über RS232 ist (STRING_MAX_LENGTH =) 21 Zeichen, wird aber nicht
    abgefangen
71  -Programm säubern/vereinheitlichen und dokumentieren
72
```

```
'Define für PIC16F1933
'jetzt neu für PIC16F1936 (gleich)
Define CONFIG1 = 0x0fc2
Define CONFIG2 = 0x1dff
Define CLOCK_FREQUENCY = 32  'MHz

Define STRING_MAX_LENGTH = 21

'DCC-Laborbooster

'noch nicht implementiert:
'Stringlänge über RS232 ist STRING_MAX_LENGTH = 21 Zeichen, wird aber nicht abgefangen

'PIC-Register: LCD-Modul: Funktion:
'RC5           4          RS
'GND           5          R/W
'RC4           6          E
'---           7          DB0
'---           8          DB1
'---           9          DB2
'---           10         DB3
'RC0           11         DB4
'RC1           12         DB5
'RC2           13         DB6
'RC3           14         DB7

Symbol EN = PORTB.0
Symbol IN1 = PORTB.1
Symbol IN2 = PORTB.2
Symbol impuls = PORTB.3
Symbol richtung = PORTB.4
Symbol knopf = PORTB.5
'Symbol tx = PORTC.6

Const Roco = 0
Const IB = 1

'LCD Einstellungen
Define LCD_LINES = 2
Define LCD_CHARS = 16
Define LCD_BITS = 4
Define LCD_DREG = PORTC
Define LCD_DBIT = 0
Define LCD_RSREG = PORTC
Define LCD_RSBIT = 5
Define LCD_EREG = PORTC
Define LCD_EBIT = 4
Define LCD_RWREG = 0
Define LCD_RWBIT = 0

Dim spannung As Word
Dim zeichenkette As String
'Dim testzahl As Word
'testzahl = 0
'Dim menu_neu As Byte
'Dim menu_alt As Byte
'menu_neu = 0
'menu_alt = 1
'Dim untermenu As Bit
'untermenu = 0
Dim verzoegerung As Byte
verzoegerung = 0
'Dim untermenu_neu As Byte
'Dim untermenu_alt As Byte
Dim menu(4) As Byte
Dim ebene As Byte
menu(0) = 1
menu(1) = 1
menu(2) = 1
menu(3) = 1
ebene = 0
Dim hilfsvariable As Byte
Dim ausgangsmodus As Byte
```

```
74  ausgangsmodus = 2  'AUS
75  Dim analog_fahrstufe As Byte
76  Dim analog_richtung As Bit
77  Dim analog_pwm As Bit
78  Dim analog_korrektur As Byte
79  Dim weichen_modus As Byte
80  Dim weichen_pulslaenge As Byte
81  Dim weichen_ruecksetzen As Byte
82  Dim parse_zaehler As Byte  'im UP parsen
83  Dim komma_zaehler As Byte  'im UP parsen
84
85  'DCC-Daten vorbereiten und auf 1 setzen
86  Dim dcc_daten(32) As Byte
87  Dim schleife As Byte
88  For schleife = 0 To 31
89      dcc_daten(schleife) = 255
90  Next schleife
91
92  'EEPROM-adressen:
93  '0 DCC-Adresse
94  '1 Fahrstufe
95  '2 Richtung
96  '3 Analog-Korrektur
97  '4 Weichen-Modus
98  '5 Pulslänge Weichen
99
100 Read 2, hilfsvariable
101 analog_richtung = hilfsvariable.0
102 Read 3, analog_korrektur
103 Read 4, weichen_modus
104 Read 5, weichen_pulslaenge
105
106 Dim dcc_modus As Byte
107 Dim dcc_adresse As Byte
108 Dim dcc_richtung As Byte
109 Dim dcc_fahr_funk_bef As Byte
110 Dim dcc_pruefbyte As Byte
111 Dim zwischenergebnis As Byte
112 dcc_modus = 1
113 dcc_adresse = 4
114 dcc_richtung = 1  'Direction: 1= Vorwärts, 0= Rückwärts
115 dcc_fahr_funk_bef = 10
116 analog_fahrstufe = 0
117 analog_richtung = 0
118 analog_pwm = 0
119 analog_korrektur = 0
120 Dim funktionsbefehl_byte As Byte
121 funktionsbefehl_byte = 0
122
123 Gosub datenaufbereitung
124
125
126 'Variablen in INT verwendet:
127 Dim empfangen As String
128 Dim zeichen As Byte
129 Dim zaehler As Byte
130 Dim empfang_fertig As Bit  'Flag aus INT für String fertig empfangen
131 empfang_fertig = 0
132 Dim polaritaet As Bit
133 Dim knopf_gedrueckt As Bit
134 knopf_gedrueckt = 0
135 Dim knopf_warten As Byte
136 knopf_warten = 0
137 Dim drehgeber As Byte
138 drehgeber = 0
139 Dim position_dcc_daten As Byte
140 position_dcc_daten = 0
141 Dim dcc_daten_byte, dcc_daten_bit As Byte
142 Dim dcc_senden As Bit
143 Dim hilfsvariable_int As Byte
144
145 ConfigPin EN = Output
146 Low EN
```

```
147   ConfigPin IN1 = Output
148   ConfigPin IN2 = Output
149   ConfigPin impuls = Input
150   ConfigPin richtung = Input
151   ConfigPin knopf = Input
152   ANSELB = 0  'PortB digital IO
153
154   Hseropen 9600
155   'ConfigPin tx = Output
156
157   'LCD initialisieren:
158   Lcdinit 0
159   'inverses "S":
160   Lcddefchar 0, %10000, %01111, %01111, %10001, %11110, %11110, %00001, %11111
161   'inverses "D":
162   Lcddefchar 1, %00011, %01101, %01110, %01110, %01110, %01101, %00011, %11111
163   'inverses "U":
164   Lcddefchar 2, %01110, %01110, %01110, %01110, %01110, %01110, %10001, %11111
165   'inverses "A":
166   Lcddefchar 3, %10001, %01110, %01110, %00000, %01110, %01110, %01110, %11111
167   'inverses "C":
168   Lcddefchar 4, %10001, %01110, %01111, %01111, %01111, %01110, %10001, %11111
169   'inverses "P":
170   Lcddefchar 5, %00001, %01110, %01110, %00001, %01111, %01111, %01111, %11111
171   'inverses "W":
172   Lcddefchar 6, %01110, %01110, %01110, %01010, %01010, %01010, %10101, %11111
173   'inverses "M":
174   Lcddefchar 7, %01110, %00100, %01010, %01010, %01110, %01110, %01110, %11111
175
176   Lcdcmdout LcdClear
177
178   'Timer0
179   'DCC-Signal logisch 1: >52us, Typ. 58us, <64us
180   '.....................0: >90us, Typ. 116us, <10ms
181   'für 58us:
182   'OPTION_REG  = 0x80
183   'TMR0   = 24
184   'für 116us:
185   OPTION_REG = %10000001
186   TMR0 = 48
187   INTCON.TMR0IE = 1  'Interrupt für Timer0 erlauben
188
189   'Interrupt für PORTB0:
190   INTCON.IOCIE = 1  'allow individual PORTB pins to generate an interrupt
191   IOCBP = %00101000  'enable to detect a rising edge on PB3 and PB5
192   OPTION_REG.7 = 1  '-WPUEN: 1 = PORTB pull-ups are disabled
193   'Interrupt für asynchrone USART:
194   PIE1.RCIE = 1  'Interrupt für USART erlauben
195   INTCON.PEIE = 1  'Interrupt für Peripheriegeräte erlauben
196   'Allgemein:
197   INTCON.GIE = 1  'Interrupt generell erlauben
198
199
200   loop:
201       If drehgeber <> 0 Then
202           hilfsvariable = menu(ebene)
203           menu(ebene) = hilfsvariable + drehgeber
204           drehgeber = 0
205       Endif
206
207       If knopf_warten = 0 Then  'wenn Knopf bereit...
208           'knopf_gedrueckt wird in der ISR gesetzt
209           If knopf_gedrueckt Then  'und gedrückt...
210               ebene = ebene + 1  'ein Untermenu runter
211               knopf_warten = 15  'Anzahl Durchgänge warten bis Knopf wieder bereit
212           Endif
213       Else
214           knopf_warten = knopf_warten - 1
215           knopf_gedrueckt = 0  'Knopf wieder auf Bereitschaft
216       Endif
217
218       If empfang_fertig = 1 Then
219           zeichenkette = empfangen
```

```
220             Gosub parsen
221             empfang_fertig = 0
222         Endif
223 
224         If weichen_ruecksetzen > 0 Then
225             weichen_ruecksetzen = weichen_ruecksetzen - 1
226             If weichen_ruecksetzen = 1 Then
227             'auf Idle setzen
228             dcc_daten(12) = %11110111
229             dcc_daten(13) = %00001111
230             dcc_daten(14) = %11000000
231             dcc_daten(15) = %11111111
232             dcc_daten(28) = %11110111
233             dcc_daten(29) = %00001111
234             dcc_daten(30) = %11000000
235             dcc_daten(31) = %11111111
236             Endif
237         Endif
238 
239         Select Case menu(0)
240         Case 0
241             menu(0) = 9
242         Case 1
243             Gosub menu_spannung
244         Case 2
245             Gosub fahrstufe
246         Case 3
247             Gosub lokadresse
248         Case 4
249             Gosub funktionsbefehl
250         Case 5
251             Gosub weichen
252         Case 6
253             Gosub menu_ausgang
254         Case 7
255             Gosub einstellungen
256         Case 8
257             Gosub menu_richtung
258         Case 9
259             Gosub info
260         Case Else
261             menu(0) = 1
262         EndSelect
263 
264         Select Case ausgangsmodus
265         Case 1  'DCC
266             High EN
267         Case 2  'AUS
268             Low EN
269         Case 3  'PWM
270             High EN
271         EndSelect
272 
273         WaitMs 10
274         'Toggle tx
275         Goto loop
276     End
277 
278 
279     On Interrupt
280         Save System
281         If INTCON.TMR0IF Then  'ist Timer0 überlaufen?
282             INTCON.TMR0IF = 0
283             'High tx  'zum Debuggen
284             '24 für logisch 0, 140 für logisch 1
285             'If dcc_daten_byte.dcc_daten_bit Then
286             Select Case ausgangsmodus
287             Case 1  'DCC, etwa 16-20us Durchlaufzeit INT
288                 If dcc_senden Then
289                     TMR0 = 145  'zurücksetzen
290                 Else
291                     TMR0 = 29  'zurücksetzen
292                 Endif
```

```
293                 If polaritaet Then
294                     Low IN1
295                     High IN2
296                 Else
297                     High IN1
298                     Low IN2
299                 Endif
300                 If polaritaet Then
301                     position_dcc_daten = position_dcc_daten + 1
302                     polaritaet = 0
303                 Else
304                     polaritaet = 1
305                 Endif
306                 hilfsvariable_int = ShiftRight(position_dcc_daten, 3)
307                 dcc_daten_byte = dcc_daten(hilfsvariable_int)
308                 dcc_daten_bit = position_dcc_daten And %00000111
309                 hilfsvariable_int = ShiftRight(dcc_daten_byte, dcc_daten_bit)
310                 dcc_senden = hilfsvariable_int.0
311             Case 3  'Analog (PWM) 6.2kHz
312                 If analog_fahrstufe > 0 Then
313                     'analog Fahren
314                     If analog_pwm Then
315                         hilfsvariable_int = ShiftLeft(analog_fahrstufe, 1)
316                         TMR0 = hilfsvariable_int
317                     Else
318                         'damit 34us Durchlaufzeit INT:
319                         'TMR0 = 255 - ((analog_fahrstufe * 2) - 2)
320                         'jetzt nur noch 10us
321                         hilfsvariable_int = ShiftLeft(analog_fahrstufe, 1)
322                         TMR0 = 255 - (hilfsvariable_int - 2)
323                     Endif
324                     If analog_richtung Then
325                         'Vorwärts
326                         If analog_pwm Then
327                             Low IN1
328                         Else
329                             High IN1
330                         Endif
331                     Else
332                         'Rückwärts
333                         If analog_pwm Then
334                             Low IN2
335                         Else
336                             High IN2
337                         Endif
338                     Endif
339                     If analog_pwm Then
340                         analog_pwm = 0
341                     Else
342                         analog_pwm = 1
343                     Endif
344                 Else
345                     Low IN1
346                     Low IN2
347                 Endif
348             EndSelect
349             'Low tx  'zum Debuggen
350         Endif
351         If IOCBF.IOCBF3 Then  'fallende Flanke an PB3 (Drehimpuls)?
352             If richtung Then
353                 drehgeber = drehgeber - 1
354             Else
355                 drehgeber = drehgeber + 1
356             Endif
357             IOCBF.IOCBF3 = 0  'RB0-Interrupt-Flag löschen
358         Endif
359         If IOCBF.IOCBF5 Then  'fallende Flanke an PB5 (Knopf)?
360             knopf_gedrueckt = 1
361             IOCBF.IOCBF5 = 0  'RB0-Interrupt-Flag löschen
362         Endif
363         If PIR1.RCIF = 1 Then  'hat die serielle Schnittstelle empfangen?
364             zeichen = RCREG  'Zeichen aus dem Register RCREG lesen
365             If zeichen > 12 Then  'Lf (10) ausblenden, nur Cr (13) durchlassen
```

```
366             If zeichen = 13 Then  'wenn Cr empfangen
367                 'High led  'rote LED ein
368                 empfangen(zaehler) = 0  'String mit 0 abschliessen
369                 empfang_fertig = 1  'Flag setzen dass ein ganzer String empfangen
                    wurde
370                 zaehler = 0  'Zeichenzähler im String zurücksetzen
371             Else
372                 empfangen(zaehler) = zeichen
373                 zaehler = zaehler + 1
374             Endif
375         Endif
376         PIR1.RCIF = 0
377     Endif
378 Resume
379
380 Proc spannung_anzeigen()
381     Adcin 0, spannung
382     spannung = spannung * 27
383     zeichenkette = #spannung
384     Lcdcmdout LcdLine2Pos(1)
385     Lcdout "   "
386     zeichenkette = ReverseStr(MidStr(ReverseStr(#spannung), 4, 2))
387     Lcdout zeichenkette
388     Lcdout "."
389     zeichenkette = MidStr(RightStr(#spannung, 3), 2, 2)
390     Lcdout zeichenkette
391     Lcdout " VDC  "
392 End Proc
393
394 menu_spannung:
395     Select Case ebene
396     Case 0
397         Lcdcmdout LcdLine1Pos(1)
398         Lcdout "<  Spannung:   >"
399         'hier zum schnellen Aktualisieren Code einfügen
400         If verzoegerung = 0 Then
401             Call spannung_anzeigen()
402         Endif
403         verzoegerung = verzoegerung + 1
404         If verzoegerung > 20 Then
405             verzoegerung = 0
406         Endif
407     Case Else
408         ebene = 0
409     EndSelect
410 Return
411
412 fahrstufe:
413     Select Case ebene
414     Case 0
415         Lcdcmdout LcdLine1Pos(1)
416         Lcdout "<  Fahrstufe:  >"
417         Read 1, menu(1)
418         Lcdcmdout LcdLine2Home
419         Lcdout "    ", #menu(1), "        "
420     Case 1
421         Lcdcmdout LcdLine1Home
422         Lcdout "   Fahrstufe:   "
423         Lcdcmdout LcdLine2Home
424         If menu(1) <= 191 Then
425             If menu(1) > 127 Then
426                 menu(1) = 127
427             Endif
428         Endif
429         If menu(1) > 192 Then
430             menu(1) = 0
431         Endif
432         Lcdout "  < ", #menu(1), " >      "
433         dcc_modus = 1
434         Read 0, dcc_adresse
435         Read 2, dcc_richtung  'Direction: 1= Vorwärts, 0= Rückwärts
436         dcc_fahr_funk_bef = menu(1)
437         Gosub datenaufbereitung
```

```
438             analog_fahrstufe = menu(1)
439             analog_fahrstufe = analog_fahrstufe / analog_korrektur
440         Case Else
441             Write 1, menu(1)
442             ebene = 0
443         EndSelect
444     Return
445 
446     lokadresse:
447         Select Case ebene
448         Case 0
449             Lcdcmdout LcdLine1Home
450             Lcdout "< Lokadresse:  >"
451             Read 0, menu(1)
452             Lcdcmdout LcdLine2Home
453             Lcdout "    ", #menu(1), "         "
454         Case 1
455             Lcdcmdout LcdLine1Home
456             Lcdout "  Lokadresse:   "
457             Lcdcmdout LcdLine2Home
458             If menu(1) = 0 Then
459                 menu(1) = 127
460             Endif
461             If menu(1) > 127 Then
462                 menu(1) = 1
463             Endif
464             Lcdout "  < ", #menu(1), " >      "
465         Case Else
466             Write 0, menu(1)
467             dcc_modus = 1
468             Read 0, dcc_adresse
469             Read 2, dcc_richtung  'Direction: 1= Vorwärts, 0= Rückwärts
470             Read 1, dcc_fahr_funk_bef
471             Gosub datenaufbereitung
472             ebene = 0
473         EndSelect
474     Return
475 
476     funktionsbefehl:
477         Select Case ebene
478         Case 0
479             Lcdcmdout LcdLine1Pos(1)
480             Lcdout "<  Funktion:   >"
481             Lcdcmdout LcdLine2Home
482             Lcdout "     Enter      "
483         Case 1
484             Lcdcmdout LcdLine1Home
485             Lcdout "   Funktion:    "
486             Lcdcmdout LcdLine2Home
487             Select Case menu(1)
488             Case 0
489                 menu(1) = 5
490             Case 1
491                 Lcdout "  < FL >      "
492             Case 2
493                 Lcdout "  < F1 >      "
494             Case 3
495                 Lcdout "  < F2 >      "
496             Case 4
497                 Lcdout "  < F3 >      "
498             Case 5
499                 Lcdout "  < F4 >      "
500             Case Else
501                 menu(1) = 1
502             EndSelect
503         Case 2
504             Lcdcmdout LcdLine2Home
505             Select Case menu(1)
506             Case 1
507                 Lcdout "    FL "
508             Case 2
509                 Lcdout "    F1 "
510             Case 3
```

```
511             Lcdout "    F2 "
512         Case 4
513             Lcdout "    F3 "
514         Case 5
515             Lcdout "    F4 "
516         EndSelect
517         Select Case menu(2)
518         Case 0
519             menu(2) = 2
520         Case 1
521             Lcdout "< AUS > "
522         Case 2
523             Lcdout "< EIN > "
524         Case Else
525             menu(2) = 1
526         EndSelect
527     Case Else  'd.h. 3 für Schalten
528         Select Case menu(1)
529         funktionsbefehl_byte.7 = 1  'Funktions-Gruppe 1
530         funktionsbefehl_byte.6 = 0
531         funktionsbefehl_byte.5 = 0
532         Case 1  'FL
533             If menu(2) = 1 Then  'AUS
534                 funktionsbefehl_byte.4 = 0
535             Else  'EIN
536                 funktionsbefehl_byte.4 = 1
537             Endif
538         Case 2  'F1
539             If menu(2) = 1 Then  'AUS
540                 funktionsbefehl_byte.0 = 0
541             Else  'EIN
542                 funktionsbefehl_byte.0 = 1
543             Endif
544         Case 3  'F2
545             If menu(2) = 1 Then  'AUS
546                 funktionsbefehl_byte.1 = 0
547             Else  'EIN
548                 funktionsbefehl_byte.1 = 1
549             Endif
550         Case 4  'F3
551             If menu(2) = 1 Then  'AUS
552                 funktionsbefehl_byte.2 = 0
553             Else  'EIN
554                 funktionsbefehl_byte.2 = 1
555             Endif
556         Case 5  'F4
557             If menu(2) = 1 Then  'AUS
558                 funktionsbefehl_byte.3 = 0
559             Else  'EIN
560                 funktionsbefehl_byte.3 = 1
561             Endif
562         EndSelect
563         dcc_fahr_funk_bef = funktionsbefehl_byte
564         dcc_modus = 2
565         Gosub datenaufbereitung
566         ebene = 0
567     EndSelect
568 Return
569
570 weichen:
571     Select Case ebene
572     Case 0
573         Lcdcmdout LcdLine1Pos(1)
574         Lcdout "<   Weichen:   >"
575         Lcdcmdout LcdLine2Home
576         Lcdout "     Enter      "
577     Case 1
578         Lcdcmdout LcdLine1Home
579         Lcdout "    Weichen:    "
580         Lcdcmdout LcdLine2Home
581         'Läuft durch von 0...255
582         Lcdout "< "
583         hilfsvariable = menu(1)
```

```
584             If hilfsvariable.0 Then
585                 Lcdout "Grn "
586             Else
587                 Lcdout "Rot "
588             Endif
589             hilfsvariable = ShiftRight(hilfsvariable, 1)
590             hilfsvariable = hilfsvariable + 1
591             Lcdout #hilfsvariable, " >      "
592         Case 2
593             Lcdcmdout LcdLine2Home
594             hilfsvariable = menu(1)
595             If hilfsvariable.0 Then
596                 Lcdout "  Grn "
597             Else
598                 Lcdout "  Rot "
599             Endif
600             'R und G sind jeweils schon eine Adresse, Bereich jetzt 0...127:
601             hilfsvariable = ShiftRight(hilfsvariable, 1)
602             'Bereich jetzt 1...128 (R1, G1, R2, G2, ... R128, G128):
603             hilfsvariable = hilfsvariable + 1
604             Lcdout #hilfsvariable
605             Select Case menu(2)
606             Case 0
607                 menu(2) = 2
608             Case 1
609                 Lcdout " < AUS > "
610             Case 2
611                 Lcdout " < EIN > "
612             Case Else
613                 menu(2) = 1
614             EndSelect
615         Case Else
616             hilfsvariable = menu(1)
617             Select Case weichen_modus
618             Case Roco
619                 'im Roco-Modus nichts machen
620             Case IB
621                 'im IB-Modus 8 Adressen dazu, fängt erst bei R5 an
622                 'geht dafür aber bis G132 (angezeigt G128):
623                 hilfsvariable = hilfsvariable + 8
624             EndSelect
625             dcc_adresse.7 = 1
626             dcc_adresse.6 = 0
627             dcc_adresse.5 = 0
628             dcc_adresse.4 = hilfsvariable.7
629             dcc_adresse.3 = hilfsvariable.6
630             dcc_adresse.2 = hilfsvariable.5
631             dcc_adresse.1 = hilfsvariable.4
632             dcc_adresse.0 = hilfsvariable.3
633             dcc_fahr_funk_bef.7 = 1
634             dcc_fahr_funk_bef.6 = 1
635             dcc_fahr_funk_bef.5 = 1
636             dcc_fahr_funk_bef.4 = 1
637             Select Case menu(2)
638             Case 1
639                 dcc_fahr_funk_bef.3 = 0
640             Case 2
641                 dcc_fahr_funk_bef.3 = 1
642             EndSelect
643             dcc_fahr_funk_bef.2 = hilfsvariable.2
644             dcc_fahr_funk_bef.1 = hilfsvariable.1
645             dcc_fahr_funk_bef.0 = hilfsvariable.0
646             dcc_modus = 3
647             Gosub datenaufbereitung
648             weichen_ruecksetzen = weichen_pulslaenge
649             ebene = 0
650         EndSelect
651     Return
652
653     menu_ausgang:
654         Select Case ebene
655         Case 0
656             Lcdcmdout LcdLine1Home
```

```
657             Lcdout "<   Ausgang:   >"
658             Lcdcmdout LcdLine2Home
659             Lcdout "     Enter      "
660             menu(1) = 2  'beim Wechseln ins Untermenu gleich AUS aktiv
661         Case 1
662             Lcdcmdout LcdLine1Home
663             Lcdout "    Ausgang:    "
664             Select Case menu(1)
665             Case 0
666                 menu(1) = 1
667             Case 1
668                 Lcdcmdout LcdLine2Pos(1)
669                 Lcdout 1, 4, 4, "   AUS   PWM "
670             Case 2
671                 Lcdcmdout LcdLine2Pos(1)
672                 Lcdout "DCC   ", 3, 2, 0, "   PWM "
673             Case 3
674                 Lcdcmdout LcdLine2Pos(1)
675                 Lcdout "DCC   AUS   ", 5, 6, 7, " "
676             Case Else
677                 menu(1) = 3
678             EndSelect
679         Case Else
680             Select Case menu(1)
681             Case 1
682                 ausgangsmodus = 1  'DCC
683             Case 1
684                 ausgangsmodus = 2  'AUS
685             Case 2
686                 ausgangsmodus = 3  'PWM
687             EndSelect
688             ebene = 0
689         EndSelect
690     Return
691
692     einstellungen:
693         Select Case ebene
694         Case 0
695             Lcdcmdout LcdLine1Pos(1)
696             Lcdout "<Einstellungen:>"
697             Lcdcmdout LcdLine2Pos(1)
698             Lcdout "     Enter      "
699         Case 1
700             Lcdcmdout LcdLine1Home
701             Lcdout " Einstellungen: "
702             Select Case menu(1)
703             Case 0
704                 menu(1) = 3
705             Case 1  'Analog-Korrektur
706                 Read 3, menu(2)
707                 Lcdcmdout LcdLine2Pos(1)
708                 Lcdout "Anl-Kor:   ", #menu(2), "    "
709             Case 2
710                 Read 4, menu(2)
711                 Lcdcmdout LcdLine2Pos(1)
712                 Select Case menu(2)
713                 Case 0
714                     Lcdout "Wch-Mod:  Roco  "
715                 Case 1
716                     Lcdout "Wch-Mod:   IB   "
717                 Case Else
718                     'wenn im EEPROM 255 steht oder so
719                     Lcdout "Wch-Mod:        "
720                 EndSelect
721             Case 3
722                 Read 5, menu(2)
723                 Lcdcmdout LcdLine2Pos(1)
724                 Lcdout "Wch-Pls:   ", #menu(2), "    "
725             Case Else
726                 menu(1) = 1
727             EndSelect
728         Case 2
729             Select Case menu(1)
```

```
730             Case 1
731                 Lcdcmdout LcdLine2Home
732                 If menu(2) = 0 Then
733                     menu(2) = 127
734                 Endif
735                 If menu(2) > 127 Then
736                     menu(2) = 1
737                 Endif
738                 Lcdout "Anl-Kor: < ", #menu(2), " >   "
739             Case 2
740                 Lcdcmdout LcdLine2Home
741                 Select Case menu(2)
742                 Case 255
743                     menu(2) = 1
744                 Case 0
745                     Lcdout "Wch-Mod: <Roco> "
746                 Case 1
747                     Lcdout "Wch-Mod: < IB > "
748                 Case Else
749                     menu(2) = 0
750                 EndSelect
751             Case 3
752                 Lcdcmdout LcdLine2Home
753                 If menu(2) = 0 Then
754                     menu(2) = 32
755                 Endif
756                 If menu(2) > 32 Then
757                     menu(2) = 1
758                 Endif
759                 Lcdout "Wch-Pls: < ", #menu(2), " >   "
760             EndSelect
761         Case Else
762             Select Case menu(1)
763             Case 1
764                 Write 3, menu(2)
765                 analog_korrektur = menu(2)
766             Case 2
767                 Write 4, menu(2)
768                 weichen_modus = menu(2)
769             Case 3
770                 Write 5, menu(2)
771                 weichen_pulslaenge = menu(2)
772             EndSelect
773             ebene = 0
774         EndSelect
775     Return
776
777     menu_richtung:
778         Select Case ebene
779         Case 0
780             Lcdcmdout LcdLine1Pos(1)
781             Lcdout "<   Richtung:  >"
782             Read 2, menu(1)
783             Lcdcmdout LcdLine2Home
784             If menu(1) = 1 Then
785                 Lcdout "    Vorwaerts   "
786             Else
787                 Lcdout "   Rueckwaerts  "
788             Endif
789         Case 1
790             Lcdcmdout LcdLine1Pos(1)
791             Lcdout "    Richtung:   "
792             Lcdcmdout LcdLine2Home
793             Select Case menu(1)
794             Case 255
795                 menu(1) = 1
796             Case 0
797                 Lcdout "<  Rueckwaerts >"
798             Case 1
799                 Lcdout "<   Vorwaerts  >"
800             Case Else
801                 menu(1) = 0
802             EndSelect
```

```
    Case Else
        Write 2, menu(1)
        hilfsvariable = menu(1)
        analog_richtung = hilfsvariable.0
        ebene = 0
    EndSelect
Return

info:
    Select Case ebene
    Case 0
        Lcdcmdout LcdLine1Home
        Lcdout "<    Info:     >"
        Lcdcmdout LcdLine2Home
        Lcdout "     Enter      "
    Case 1
        Lcdcmdout LcdLine1Home
        Lcdout "     Info:      "
        Lcdcmdout LcdLine2Home
        Select Case menu(1)
        Case 0
            menu(1) = 3
        Case 1
            Lcdout "<  Version 9   >"
        Case 2
            Lcdout "<  von Chregu  >"
        Case 3
            Lcdout "<  02.01.2020  >"
        Case Else
            menu(1) = 1
        EndSelect
    Case Else
        ebene = 0
    EndSelect
Return

datenaufbereitung:
'dcc_modus ist Byte mit 1...6
'Modi intern: 1: Fahrbefehl, 2: Schalten der Funktionen, 3: Schaltdekoder
'Modi extern: 4: Fahrbefehl, 5: Schalten der Funktionen, 6: Schaltdekoder
'dcc_adresse ist 6 oder 7 Bits
'dcc_richtung ist 1 Bit: 1= Vorwärts, 0= Rückwärts
'dcc_fahr_funk_bef ist 6 oder 7 Bits

    Select Case dcc_modus
    Case 1, 4  'Fahrbefehl
        If dcc_modus = 1 Then
            hilfsvariable = 0
        Else  'bei 4
            hilfsvariable = 16
        Endif
        dcc_adresse.7 = 0  'zur Sicherheit Bit löschen
        dcc_fahr_funk_bef.7 = dcc_richtung.0  'Richtung auf Fahrstufenbyte übertragen
        zwischenergebnis.0 = 1  '9.  1
        zwischenergebnis.1 = 1  '10. 1
        zwischenergebnis.2 = 1  '11. 1
        zwischenergebnis.3 = 1  '12. 1
        zwischenergebnis.4 = 0  'Packet Start Bit
        zwischenergebnis.5 = 0  'MSB Adresse
        zwischenergebnis.6 = dcc_adresse.6
        zwischenergebnis.7 = dcc_adresse.5
        'hilfsvariable = hilfsvariable + 1
        ASM:        INCF hilfsvariable,F
        dcc_daten(hilfsvariable) = zwischenergebnis
        zwischenergebnis.0 = dcc_adresse.4
        zwischenergebnis.1 = dcc_adresse.3
        zwischenergebnis.2 = dcc_adresse.2
        zwischenergebnis.3 = dcc_adresse.1
        zwischenergebnis.4 = dcc_adresse.0
        zwischenergebnis.5 = 0  '1. Daten Byte Start Bit
        zwischenergebnis.6 = 0  'Bit 0: 128 Fahrstufen ganzes Byte
        zwischenergebnis.7 = 0  'Bit 1
        hilfsvariable = hilfsvariable + 1
```

```
876             dcc_daten(hilfsvariable) = zwischenergebnis
877             zwischenergebnis.0 = 1  'Bit 2
878             zwischenergebnis.1 = 1  'Bit 3
879             zwischenergebnis.2 = 1  'Bit 4
880             zwischenergebnis.3 = 1  'Bit 5
881             zwischenergebnis.4 = 1  'Bit 6
882             zwischenergebnis.5 = 1  'Bit 7
883             zwischenergebnis.6 = 0  '2. Daten Byte Start Bit
884             zwischenergebnis.7 = dcc_richtung.0  'Richtung: 1= Vorwärts, 0= Rückwärts
885             hilfsvariable = hilfsvariable + 1
886             dcc_daten(hilfsvariable) = zwischenergebnis
887             zwischenergebnis.0 = dcc_fahr_funk_bef.6
888             zwischenergebnis.1 = dcc_fahr_funk_bef.5
889             zwischenergebnis.2 = dcc_fahr_funk_bef.4
890             zwischenergebnis.3 = dcc_fahr_funk_bef.3
891             zwischenergebnis.4 = dcc_fahr_funk_bef.2
892             zwischenergebnis.5 = dcc_fahr_funk_bef.1
893             zwischenergebnis.6 = dcc_fahr_funk_bef.0
894             zwischenergebnis.7 = 0  'Prüf Byte Start Bit
895             hilfsvariable = hilfsvariable + 1
896             dcc_daten(hilfsvariable) = zwischenergebnis
897             dcc_pruefbyte = dcc_adresse Xor %00111111
898             dcc_pruefbyte = dcc_pruefbyte Xor dcc_fahr_funk_bef
899             zwischenergebnis.0 = dcc_pruefbyte.7
900             zwischenergebnis.1 = dcc_pruefbyte.6
901             zwischenergebnis.2 = dcc_pruefbyte.5
902             zwischenergebnis.3 = dcc_pruefbyte.4
903             zwischenergebnis.4 = dcc_pruefbyte.3
904             zwischenergebnis.5 = dcc_pruefbyte.2
905             zwischenergebnis.6 = dcc_pruefbyte.1
906             zwischenergebnis.7 = dcc_pruefbyte.0
907             hilfsvariable = hilfsvariable + 1
908             dcc_daten(hilfsvariable) = zwischenergebnis
909         Case 2, 5  'Schalten der Funktionen
910             If dcc_modus = 2 Then
911                 hilfsvariable = 6
912             Else  'bei 5
913                 hilfsvariable = 6 + 16
914             Endif
915             dcc_adresse.7 = 0  'zur Sicherheit Bit löschen
916             zwischenergebnis.0 = 1  '9.  1
917             zwischenergebnis.1 = 1  '10. 1
918             zwischenergebnis.2 = 1  '11. 1
919             zwischenergebnis.3 = 1  '12. 1
920             zwischenergebnis.4 = 0  'Packet Start Bit
921             zwischenergebnis.5 = 0  'MSB Adresse
922             zwischenergebnis.6 = dcc_adresse.6
923             zwischenergebnis.7 = dcc_adresse.5
924             hilfsvariable = hilfsvariable + 1
925             dcc_daten(hilfsvariable) = zwischenergebnis
926             zwischenergebnis.0 = dcc_adresse.4
927             zwischenergebnis.1 = dcc_adresse.3
928             zwischenergebnis.2 = dcc_adresse.2
929             zwischenergebnis.3 = dcc_adresse.1
930             zwischenergebnis.4 = dcc_adresse.0
931             zwischenergebnis.5 = 0  '1. Daten Byte Start Bit
932             zwischenergebnis.6 = dcc_fahr_funk_bef.7  'Funktionen
933             zwischenergebnis.7 = dcc_fahr_funk_bef.6
934             hilfsvariable = hilfsvariable + 1
935             dcc_daten(hilfsvariable) = zwischenergebnis
936             zwischenergebnis.0 = dcc_fahr_funk_bef.5
937             zwischenergebnis.1 = dcc_fahr_funk_bef.4
938             zwischenergebnis.2 = dcc_fahr_funk_bef.3
939             zwischenergebnis.3 = dcc_fahr_funk_bef.2
940             zwischenergebnis.4 = dcc_fahr_funk_bef.1
941             zwischenergebnis.5 = dcc_fahr_funk_bef.0
942             zwischenergebnis.6 = 0  'Packet Start Bit
943             dcc_pruefbyte = dcc_adresse Xor dcc_fahr_funk_bef
944             zwischenergebnis.7 = dcc_pruefbyte.7  'Prüf Byte Start Bit
945             hilfsvariable = hilfsvariable + 1
946             dcc_daten(hilfsvariable) = zwischenergebnis
947             zwischenergebnis.0 = dcc_pruefbyte.6
948             zwischenergebnis.1 = dcc_pruefbyte.5
```

```
            zwischenergebnis.2 = dcc_pruefbyte.4
            zwischenergebnis.3 = dcc_pruefbyte.3
            zwischenergebnis.4 = dcc_pruefbyte.2
            zwischenergebnis.5 = dcc_pruefbyte.1
            zwischenergebnis.6 = dcc_pruefbyte.0
            zwischenergebnis.7 = 1  '1. Bit der Synchronisationssequenz
            hilfsvariable = hilfsvariable + 1
            dcc_daten(hilfsvariable) = zwischenergebnis
        Case 3, 6  'Schaltdekoder
            If dcc_modus = 3 Then
                hilfsvariable = 11
            Else  'bei 6
                hilfsvariable = 11 + 16
            Endif
            dcc_adresse.7 = 1  'zur Sicherheit Bit setzen
            dcc_adresse.6 = 0  'zur Sicherheit Bit löschen
            zwischenergebnis.0 = 1  '10. 1
            zwischenergebnis.1 = 1  '11. 1
            zwischenergebnis.2 = 1  '12. 1
            zwischenergebnis.3 = 0  'Packet Start Bit
            zwischenergebnis.4 = 1  'MSB Adresse
            zwischenergebnis.5 = 0
            zwischenergebnis.6 = dcc_adresse.5
            zwischenergebnis.7 = dcc_adresse.4
            hilfsvariable = hilfsvariable + 1
            dcc_daten(hilfsvariable) = zwischenergebnis
            zwischenergebnis.0 = dcc_adresse.3
            zwischenergebnis.1 = dcc_adresse.2
            zwischenergebnis.2 = dcc_adresse.1
            zwischenergebnis.3 = dcc_adresse.0
            zwischenergebnis.4 = 0  '1. Daten Byte Start Bit
            dcc_fahr_funk_bef.7 = 1  'zur Sicherheit Bit setzen
            zwischenergebnis.5 = 1  'Bit 7: Befehl
            zwischenergebnis.6 = dcc_fahr_funk_bef.6
            zwischenergebnis.7 = dcc_fahr_funk_bef.5
            hilfsvariable = hilfsvariable + 1
            dcc_daten(hilfsvariable) = zwischenergebnis
            zwischenergebnis.0 = dcc_fahr_funk_bef.4
            zwischenergebnis.1 = dcc_fahr_funk_bef.3
            zwischenergebnis.2 = dcc_fahr_funk_bef.2
            zwischenergebnis.3 = dcc_fahr_funk_bef.1
            zwischenergebnis.4 = dcc_fahr_funk_bef.0
            zwischenergebnis.5 = 0  'Packet Start Bit
            dcc_pruefbyte = dcc_adresse Xor dcc_fahr_funk_bef
            zwischenergebnis.6 = dcc_pruefbyte.7  'Prüf Byte Start Bit
            zwischenergebnis.7 = dcc_pruefbyte.6
            hilfsvariable = hilfsvariable + 1
            dcc_daten(hilfsvariable) = zwischenergebnis
            zwischenergebnis.0 = dcc_pruefbyte.5
            zwischenergebnis.1 = dcc_pruefbyte.4
            zwischenergebnis.2 = dcc_pruefbyte.3
            zwischenergebnis.3 = dcc_pruefbyte.2
            zwischenergebnis.4 = dcc_pruefbyte.1
            zwischenergebnis.5 = dcc_pruefbyte.0
            zwischenergebnis.6 = 1  '1. Bit der Synchronisationssequenz
            zwischenergebnis.7 = 1
            hilfsvariable = hilfsvariable + 1
            dcc_daten(hilfsvariable) = zwischenergebnis
        EndSelect
    Return

    'XL:{Lok#,[Speed],[FL],[Dir],[F1],[F2],[F3],[F4]}
    'XF:{Lok#,[F1],[F2],[F3],[F4],[F5],[F6],[F7],[F8]}
    'XT:{Trnt#,[Color],[Status]}
    'Dir: 0 oder r = rückwärts, 1 oder f = vorwärts
    'Color: 0 oder 'r' = Rot / Abzweig, 1 oder 'g' = Grün / Geradeaus
    'Status: 1 = ein, 0 oder nicht angegeben = aus
    parsen:
        parse_zaehler = 0
        komma_zaehler = 0
        If zeichenkette(parse_zaehler) = "X" Then
            'Hserout "X empfangen!", CrLf
            parse_zaehler = parse_zaehler + 1
```

```
1022        Endif
1023        Select Case zeichenkette(parse_zaehler)
1024        Case "L"
1025            'Lokfunktionen
1026            dcc_adresse = 0
1027            dcc_fahr_funk_bef = 0
1028            'Hserout "Lokfunktion", CrLf
1029            parse_zaehler = parse_zaehler + 1
1030            While zeichenkette(parse_zaehler) <> 0
1031                If zeichenkette(parse_zaehler) = "," Then
1032                    komma_zaehler = komma_zaehler + 1
1033                    parse_zaehler = parse_zaehler + 1
1034                Else
1035                    Select Case komma_zaehler
1036                    Case 0
1037                        'Loknummer
1038                        dcc_adresse = dcc_adresse * 10
1039                        zwischenergebnis = StrValB(MidStr(zeichenkette, parse_zaehler + 1
                            , 1))
1040                        dcc_adresse = dcc_adresse + zwischenergebnis
1041                    Case 1
1042                        'Geschwindigkeit
1043                        dcc_fahr_funk_bef = dcc_fahr_funk_bef * 10
1044                        zwischenergebnis = StrValB(MidStr(zeichenkette, parse_zaehler + 1
                            , 1))
1045                        dcc_fahr_funk_bef = dcc_fahr_funk_bef + zwischenergebnis
1046                    Case 2
1047                        'FL
1048                        funktionsbefehl_byte.7 = 1  'Funktions-Gruppe 1
1049                        funktionsbefehl_byte.6 = 0
1050                        funktionsbefehl_byte.5 = 0
1051                        If zeichenkette(parse_zaehler) = "1" Then
1052                            funktionsbefehl_byte.4 = 1
1053                        Else
1054                            funktionsbefehl_byte.4 = 0
1055                        Endif
1056                    Case 3
1057                        'Dir
1058                        If zeichenkette(parse_zaehler) = "1" Then
1059                            dcc_richtung = 1
1060                        Else
1061                            dcc_richtung = 0
1062                        Endif
1063                        analog_fahrstufe = dcc_fahr_funk_bef
1064                        dcc_modus = 4
1065                        Hserout #dcc_adresse, ":", #dcc_fahr_funk_bef, ":", #dcc_richtung
                            , CrLf
1066                        Gosub datenaufbereitung
1067                    Case 4
1068                        'F1
1069                        If zeichenkette(parse_zaehler) = "1" Then
1070                            funktionsbefehl_byte.0 = 1
1071                        Else
1072                            funktionsbefehl_byte.0 = 0
1073                        Endif
1074                    Case 5
1075                        'F2
1076                        If zeichenkette(parse_zaehler) = "1" Then
1077                            funktionsbefehl_byte.1 = 1
1078                        Else
1079                            funktionsbefehl_byte.1 = 0
1080                        Endif
1081                    Case 6
1082                        'F3
1083                        If zeichenkette(parse_zaehler) = "1" Then
1084                            funktionsbefehl_byte.2 = 1
1085                        Else
1086                            funktionsbefehl_byte.2 = 0
1087                        Endif
1088                    Case 7
1089                        'F4
1090                        If zeichenkette(parse_zaehler) = "1" Then
1091                            funktionsbefehl_byte.3 = 1
```

```
1092                    Else
1093                        funktionsbefehl_byte.3 = 0
1094                    Endif
1095                    dcc_fahr_funk_bef = funktionsbefehl_byte
1096                    dcc_modus = 5
1097                    Gosub datenaufbereitung
1098                    Hserout "Ok", Cr
1099                EndSelect
1100                parse_zaehler = parse_zaehler + 1
1101            Endif
1102        Wend
1103    Case "F"
1104        'Funktionsbefehl
1105        dcc_adresse = 0
1106        dcc_fahr_funk_bef = 0
1107        parse_zaehler = parse_zaehler + 1
1108        While zeichenkette(parse_zaehler) <> 0
1109            If zeichenkette(parse_zaehler) = "," Then
1110                komma_zaehler = komma_zaehler + 1
1111                parse_zaehler = parse_zaehler + 1
1112            Else
1113                Select Case komma_zaehler
1114                Case 0
1115                    'Loknummer
1116                    dcc_adresse = dcc_adresse * 10
1117                    zwischenergebnis = StrValB(MidStr(zeichenkette, parse_zaehler + 1
                        , 1))
1118                    dcc_adresse = dcc_adresse + zwischenergebnis
1119                Case 1
1120                    'F1
1121                    If zeichenkette(parse_zaehler) = "1" Then
1122                        funktionsbefehl_byte.0 = 1
1123                    Else
1124                        funktionsbefehl_byte.0 = 0
1125                    Endif
1126                Case 2
1127                    'F2
1128                    If zeichenkette(parse_zaehler) = "1" Then
1129                        funktionsbefehl_byte.1 = 1
1130                    Else
1131                        funktionsbefehl_byte.1 = 0
1132                    Endif
1133                Case 3
1134                    'F3
1135                    If zeichenkette(parse_zaehler) = "1" Then
1136                        funktionsbefehl_byte.2 = 1
1137                    Else
1138                        funktionsbefehl_byte.2 = 0
1139                    Endif
1140                Case 4
1141                    'F4
1142                    If zeichenkette(parse_zaehler) = "1" Then
1143                        funktionsbefehl_byte.3 = 1
1144                    Else
1145                        funktionsbefehl_byte.3 = 0
1146                    Endif
1147                Case 5  'ab hier müsste der Befehl mehrmals gesendet werden
1148                    'F5   ab hier noch nicht implementiert
1149                    If zeichenkette(parse_zaehler) = "1" Then
1150                        'für später
1151                    Else
1152                        'für später
1153                    Endif
1154                Case 6
1155                    'F6
1156                    If zeichenkette(parse_zaehler) = "1" Then
1157                        'für später
1158                    Else
1159                        'für später
1160                    Endif
1161                Case 7
1162                    'F7
1163                    If zeichenkette(parse_zaehler) = "1" Then
```

```
1164                        'für später
1165                    Else
1166                        'für später
1167                    Endif
1168                Case 8
1169                    'F8
1170                    If zeichenkette(parse_zaehler) = "1" Then
1171                        'für später
1172                    Else
1173                        'für später
1174                    Endif
1175                    dcc_fahr_funk_bef = funktionsbefehl_byte
1176                    dcc_modus = 5
1177                    Gosub datenaufbereitung
1178                    Hserout "Ok", Cr
1179                EndSelect
1180                parse_zaehler = parse_zaehler + 1
1181            Endif
1182        Wend
1183    Case "T"
1184        'Weichenbefehl
1185        dcc_adresse = 0
1186        dcc_fahr_funk_bef = 0
1187        'Hserout "Lokfunktion", CrLf
1188        parse_zaehler = parse_zaehler + 1
1189        While zeichenkette(parse_zaehler) <> 0
1190            If zeichenkette(parse_zaehler) = "," Then
1191                komma_zaehler = komma_zaehler + 1
1192                parse_zaehler = parse_zaehler + 1
1193            Else
1194                Select Case komma_zaehler
1195                Case 0
1196                    'Weichennummer
1197                    dcc_adresse = dcc_adresse * 10
1198                    zwischenergebnis = StrValB(MidStr(zeichenkette, parse_zaehler + 1
                        , 1))
1199                    dcc_adresse = dcc_adresse + zwischenergebnis
1200                Case 1
1201                    'Farbe
1202                    If zeichenkette(parse_zaehler) = "1" Or zeichenkette(
                        parse_zaehler) = "g" Then
1203                        dcc_fahr_funk_bef.0 = 1
1204                    Else  'wenn "0" oder "r"
1205                        dcc_fahr_funk_bef.0 = 0
1206                    Endif
1207                Case 2
1208                    'Status
1209                    If zeichenkette(parse_zaehler) = "1" Then
1210                        dcc_fahr_funk_bef.3 = 1
1211                    Else
1212                        dcc_fahr_funk_bef.3 = 0
1213                    Endif
1214                    'Daten vorbereiten:
1215                    Select Case weichen_modus
1216                    Case 0  'Roco
1217                        dcc_adresse = dcc_adresse - 1
1218                    Case 1  'IB
1219                        dcc_adresse = dcc_adresse + 3
1220                    Case Else
1221                        'nichts machen
1222                    EndSelect
1223                    hilfsvariable = dcc_adresse * 2
1224                    dcc_adresse.7 = 1
1225                    dcc_adresse.6 = 0
1226                    dcc_adresse.5 = 0
1227                    dcc_adresse.4 = hilfsvariable.7
1228                    dcc_adresse.3 = hilfsvariable.6
1229                    dcc_adresse.2 = hilfsvariable.5
1230                    dcc_adresse.1 = hilfsvariable.4
1231                    dcc_adresse.0 = hilfsvariable.3
1232                    dcc_fahr_funk_bef.7 = 1
1233                    dcc_fahr_funk_bef.6 = 1
1234                    dcc_fahr_funk_bef.5 = 1
```

```
                    dcc_fahr_funk_bef.4 = 1
                    dcc_fahr_funk_bef.2 = hilfsvariable.2
                    dcc_fahr_funk_bef.1 = hilfsvariable.1
                    dcc_modus = 6
                    Gosub datenaufbereitung
                    Hserout "Ok", Cr
                    weichen_ruecksetzen = weichen_pulslaenge
                EndSelect
                parse_zaehler = parse_zaehler + 1
            Endif
        Wend
    Case "S"  'STOP
        If InStrRev(zeichenkette, "T") > 0 Then
            ausgangsmodus = 2
            Hserout "Pwr off", Cr
        Endif
    Case "G"  'GO
        If InStrRev(zeichenkette, "O") > 0 Then
            ausgangsmodus = 1
            Hserout "Pwr on", Cr
        Endif
    Case "Y"  'Status
        If InStrRev(zeichenkette, "Y") > 0 Then
            Select Case ausgangsmodus
            Case 1
                Hserout "Pwr on", Cr
            Case 2
                Hserout "Pwr off", Cr
            Case 3
                Hserout "Analog", Cr
            EndSelect
        Endif
    Case "V"  'Version
        If InStrRev(zeichenkette, "V") > 0 Then
            Hserout "Version 9", Cr
        Endif
    EndSelect
Return
```