

# **Befehls-Erkennung**

**(Mot / DCC)**

**der Dekoder-Projekte von**

**www.DIGITAL-BAHN.de**

Prozessor: PIC 4 MHz, insbesondere 12F629 / 16F630 / 16F676

Sprache: PIC Assembler

Dieses Dokument beschreibt die Implementierung der Erkennung von Motorola bzw. DCC Befehlen bei der PIC-Software der auf [www.digital-bahn.de](http://www.digital-bahn.de) veröffentlichten Projekte. Dieses Dokument ist nur für den technisch interessierten gedacht, ein normaler Anwender der Dekoder braucht dieses Dokument nicht zu lesen und erst recht nicht zu verstehen!

Ich gehe bei der Beschreibung von der Kenntnis des Befehls-Aufbaus der Motorola bzw. DCC-Befehle aus. Entsprechende Links zu Beschreibungen der Datenformate sind auf meiner Homepage zu finden (Info -> Technisches..).

Kapitel 1 bis 5 beschäftigen sich mit der Implementierung im PIC. Verwendete Register und Routinen werden beschrieben (Assembler). Dabei wird davon ausgegangen, dass ein 4 MHz PIC verwendet wird, in meinen Projekten sind dies der 12F629 und 16F630 (wobei auch die entsprechenden „analogen“ PICs verwendet werden können, also 16F676).

Kapitel 6 beschreibt hauptsächlich in Tabellen-Form, wie die einzelnen Befehle vom PIC gesehen werden, also welches Bit in welchem Register abgelegt ist und dementsprechend auch, wie die Befehls-Adressen im EEprom (Kapitel 7) abgespeichert werden.

## INHALT:

<b>1</b>	<b>Verwendete Register</b> .....	<b>3</b>
<b>2</b>	<b>Beschreibung der Routinen für Motorola (Schaltbefehle)</b> .....	<b>5</b>
2.1	ISR-Routine Motorola (Schaltbefehle) .....	6
2.2	Routine „check“ für Motorola (Schaltbefehle) .....	7
2.3	Unterprogramm „POLARITY“ .....	8
2.4	Problem Sonderoption 901 (3.Jan.06) .....	8
<b>3</b>	<b>Beschreibung der Routinen für Motorola (Funktionsdekoder)</b> .....	<b>9</b>
3.1	ISR-Routine Motorola (Funktionsdekoder) .....	<b>Fehler! Textmarke nicht definiert.</b>
3.2	Routine „check“ für Motorola (Funktionsdekoder) .....	<b>Fehler! Textmarke nicht definiert.</b>
3.3	Unterprogramm „POLARITY“ für Motorola (Funktionsdekoder) .....	<b>Fehler! Textmarke nicht definiert.</b>
<b>4</b>	<b>Beschreibung der Routinen für DCC (Schaltbefehle)</b> .....	<b>10</b>
4.1	ISR-Routine DCC (Schaltbefehle) .....	10
4.2	Routine „check“ für DCC (Schaltbefehle) .....	11
4.3	Unterprogramm „POLARITY“ für DCC (Schaltbefehle) .....	12
<b>5</b>	<b>Beschreibung der Routinen für DCC (Funktionsdekoder)</b> .....	<b>13</b>
5.1	ISR-Routine DCC (Funktionsdekoder) .....	13
5.2	Routine „check“ für DCC (Funktionsdekoder) .....	14
5.3	Unterprogramm „POLARITY“ für DCC (Funktionsdekoder) .....	15
<b>6</b>	<b>Funktionsbeschreibungen</b> .....	<b>16</b>
6.1	Motorola-Befehle .....	16
6.1.1	Schaltdecoder MM .....	16
6.1.2	Funktionsdekoder MM .....	18
6.2	DCC Befehle .....	22
6.2.1	Schaltdecoder DCC .....	24
6.2.2	Funktionsdekoder DCC .....	26
<b>7</b>	<b>EEPROM Speicherbelegung</b> .....	<b>28</b>
7.1	Motorola Schaltdecoder .....	28
7.2	DCC Schaltdecoder .....	30

**1 Verwendete Register**

Name	Zugriff	R W	Beschreibung	
<b>Adresserkennung</b>				
R_ADR_1 R_ADR_2 R_ADR_3 R_ADR_1a (nur DCC) R_ADR_2a (nur DCC) R_ADR_3a (nur MM)	CHECK	W R	Umladen der im EEPROM gespeicherten Adresse ins RAM  Vergleich mit den Registern R_DEKADR_x_L	
R_DEKADR_1 R_DEKADR_2 R_DEKADR_3 R_DEKADR_4 (nur DCC) R_DEKADR_5 (nur DCC)	ISR	R/W	Zwischenspeicher für die empfangenen Befehle. R_1 bis R_3 werden in ISR bei abgeschlossenem Befehl in die Register R_DEKADR_x_L umgeladen	
R_DEKADR_1_L R_DEKADR_2_L R_DEKADR_3_L	ISR CHECK SAVE_WRITE	W R R	Umladen aus R_DEKADR_x Vergleichen der empfangenen Befehle mit den in EEPROM liegenden Adressen Abspeichern der Adressen im Speicher-Mode	
R_FLAG	#0: F_IDLE (nur MM)	ISR	W R	Wird gelöscht während Einlesen eines Motorola-Befehls Wenn „IDLE“, dann keine Kontrolle des Timings (da erstes Bit)
		POLARITY	W	Wird gesetzt, wenn im Protokoll die Pause erkannt.
	#1: F_READ (nur MM)	ISR	W	Wird gesetzt während Einlesen eines Motorola-Befehls
			W	Wird gelöscht, wenn Timing während Befehls-Empfang nicht OK
			W	Wird gelöscht, wenn Empfang eines Motorola-Befehls abgeschlossen.
		R	Nur wenn „READ“, werden die Befehle eingelesen	
	POLARITY	W	Wird gelöscht, wenn im Protokoll die Pause erkannt	
	#2: F_CMD (MM / DCC)	ISR	W	Wird gesetzt, wenn Empfang eines Motorola-Befehls abgeschlossen.
		CHECK	W	Wird immer gelöscht
		MAIN-LOOP	R	Wenn gesetzt, wird in „CHECK“ gesprungen
#6 (nur F-Dekoder)	Fahrtrichtung			
#7 (nur MM)	CHECK	R/W	Flag um sicherzustellen, dass ein Befehl nur 1 mal abgearbeitet wird. Bei „Off“-Befehl wird das Flag gelöscht	
R_POL	#0 (nur MM, bei DCC immer "0")	POLARITY	W	Wird bei inverser Polarität gesetzt
		CHECK	R	Wenn gesetzt, dann inverse Polarität der Command-Signale (Busbefehle)
		SAVE	R	Wenn gesetzt, dann inverse Polarität der Command-Signale (Busbefehle)
R_FLAG_DCC	#1: byte_1	ISR	R/W	gesetzt wenn Byte 1 (Befehlsbyte) gelesen werden soll
	#2: byte_2	ISR	R/W	gesetzt wenn Byte 2 (Adressbyte / Adressbyte 2 bei langen Adressen) gelesen werden soll
	#3: byte_3	ISR	R/W	gesetzt wenn Byte 3 (Adressbyte 1 bei langen Adressen) gelesen werden soll
	#6: get_byte	ISR	R/W	gesetzt wenn Lesevorgang Byte 4/3/2/1

Name	Zugriff	R W	Beschreibung
R_BITCOUNT (nur MM)	POLARITY	W	wird mit d'18' beschrieben, wenn IDLE-State detected
	ISR	R/W	Counter, wieviel Bits schon eingelesen sind (decrement bis auf ZERO)

**2 Beschreibung der Routinen für Motorola (Schaltbefehle)**

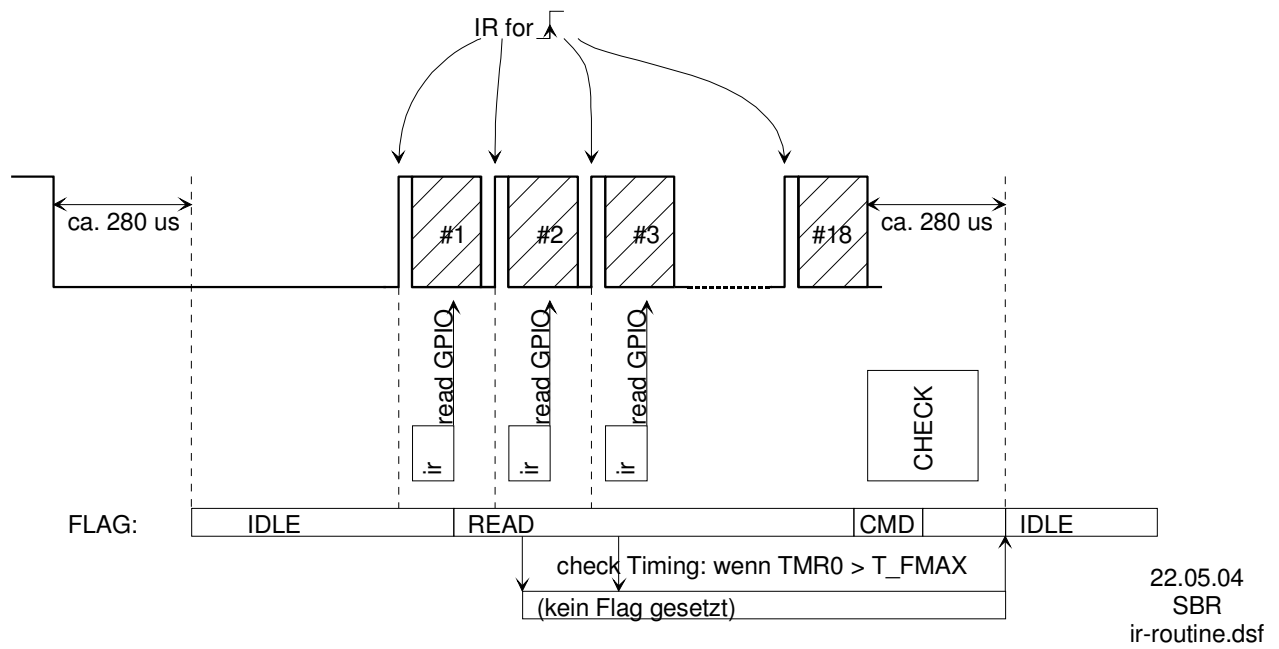
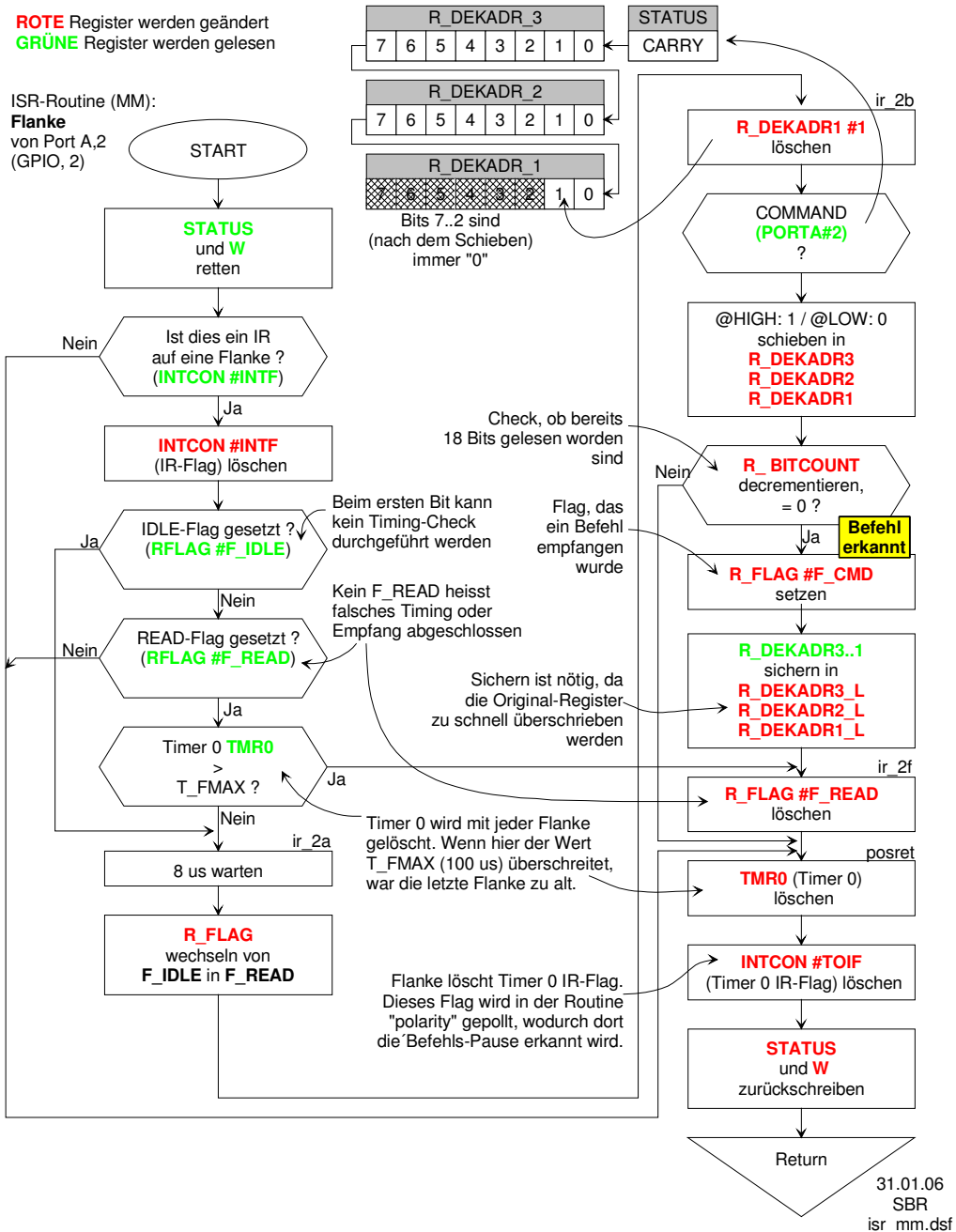


Abbildung 1: Prinzip der MM-Auswertung

**2.1 ISR-Routine Motorola (Schaltbefehle)**

<b>Aufgabe:</b>	Verarbeitung der Busbefehle (MOTOROLA MM)
<b>Aufgerufen:</b>	Durch Interrupt (IB0-fallend oder steigend – wird in „polarity“ umgeschaltet)
<b>Beschreibung:</b>	



**Abbildung 2: ISR-Routine MM**

2.2 Routine „check“ für Motorola (Schaltbefehle)

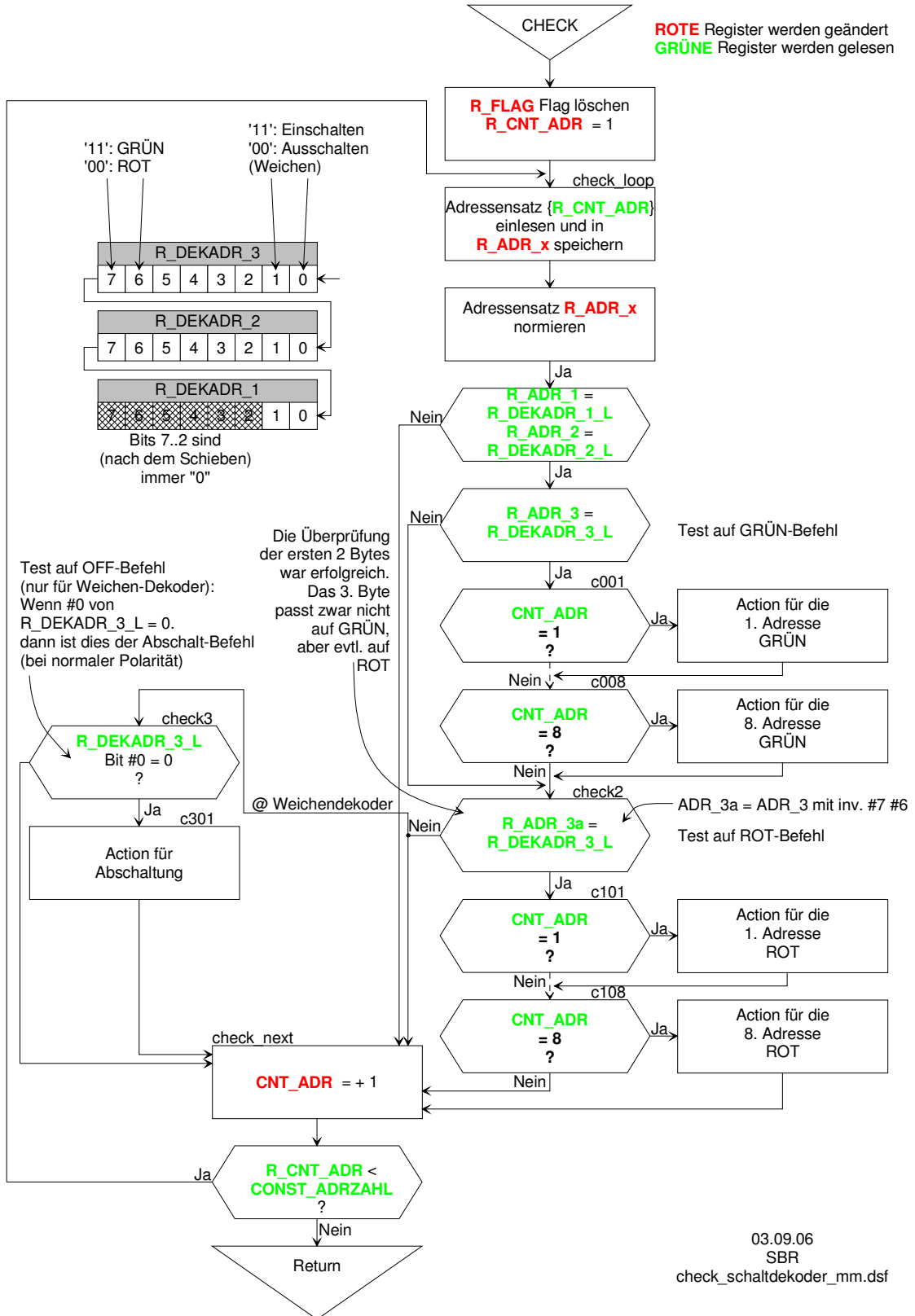


Abbildung 3: Routine "check" (Schaltdekoeder, MM)

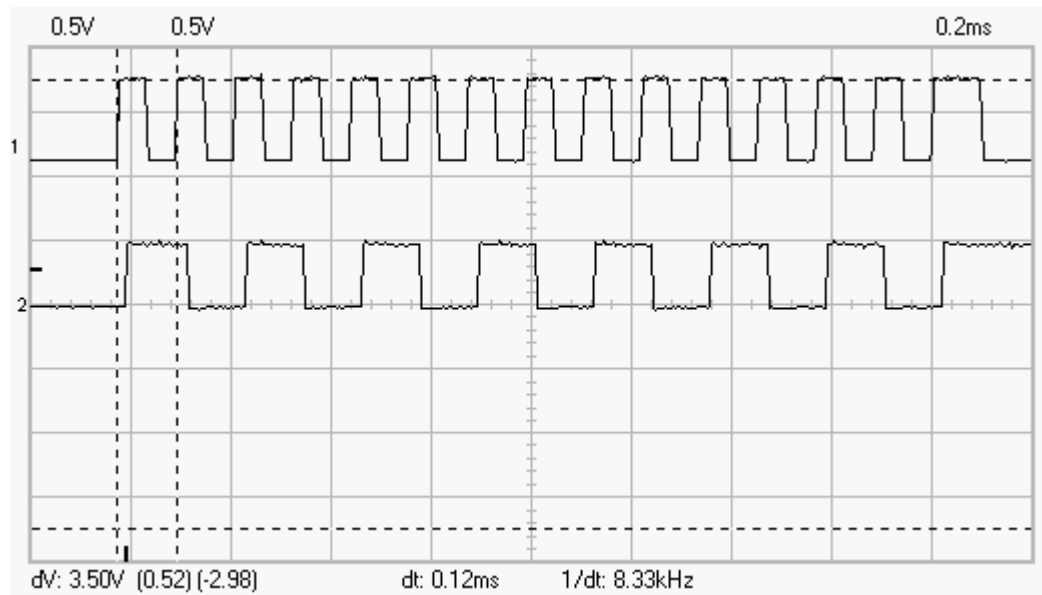
## 2.3 Unterprogramm „POLARITY“

<b>Aufgabe:</b>	Erkennen der Polarität des Busses und der Befehlspause (nur MOTOROLA)
<b>Aufgerufen:</b>	In Hauptroutine „LOOP“
<b>Beschreibung:</b>	Hierfür wird Timer 0 verwendet. Dieser wird in der ISR mit jeder Flanke zurückgesetzt. Erreicht er dennoch den Überlauf (nach 255 us ohne Flanke), so wird dieser in der Routine „POLARITY“ abgefangen, wo das IR-FLAG abgefragt wird. Ist das IR-FLAG gesetzt, so wird das IDLE-Flag im Register R_FLAG gesetzt und R_BITCOUNT zurück auf ‚18‘ gesetzt. Somit ist der Dekoder scharf für die Erkennung des nächsten Motorola-Befehls. Zusätzlich wird die Polarität am Befehls-Eingang überprüft, wodurch eine Online Polaritätserkennung möglich ist. Bei normaler Polarität wird R_POL #0 gelöscht und der IR triggert auf die steigende Flanke. Bei DCC ist R_POL #0 immer „0“.

## 2.4 Problem Sonderoption 901 (3.Jan.06)

Bei der Sonderoption 901 = 3 generiert die IBOX Pausen, es entstehen DCC Impuls-Pakete.

Die DCC-Befehle werden durch das Timeout über T\_FMAX erfasst, denn die Zeit zwischen 2 pos. Flanken beträgt bei MOT = 104 us., kann bei DCC minimal ebenfalls 104 us. betragen. Eine Änderung von T\_FMAX hätte zwar geholfen (Istwert = 100d, OK bei 96d), wäre aber kritisch. Daher verwerfe ich jetzt alle Motorola-Befehle, die „3FF“ sind, und dies nur in der „SAVE“-Routine. Betrachtet wird das Bit-Paar #5 (Bit R\_DEKADR\_2\_L#0), das bei einem Motorola-Befehl eigentlich „00“ sein müsste. Bei einem falschverstandenen DCC-Befehle werden hingegen alle Bits als „1“ erkannt. Das ganze wurde auch verpolt getestet.



**Abbildung 4: Oszillogramm Intellibox**

Hier zu sehen: 18 Bit werden eingelesen (Flanke unteres Signal), es ist aber ein DCC-Befehl (oberes Signal) nach einer Pause. Die erkannte Adresse ist dann „3FF“

### 3 Beschreibung der Routinen für Motorola (Funktionsdekoder)

Bei den Funktionsdekodern ist das Prinzip der Befehls-Erkennung identisch mit der Befehls-Erkennung bei den Schaltdekodern. Lediglich das Timing ändert sich hier, da Befehle für Funktionsdekoder Fahrzeug-Befehle sind dadurch alles „halb so schnell“ läuft.

## 4 Beschreibung der Routinen für DCC (Schaltbefehle)

### 4.1 ISR-Routine DCC (Schaltbefehle)

<b>Aufgabe:</b>	Verarbeitung der Busbefehle (DCC)
<b>Aufgerufen:</b>	Durch Interrupt (IB0-Change), steigende und fallende Flanke
<b>Beschreibung:</b>	
ISR für DCC:	

4.2 Routine „check“ für DCC (Schaltbefehle)

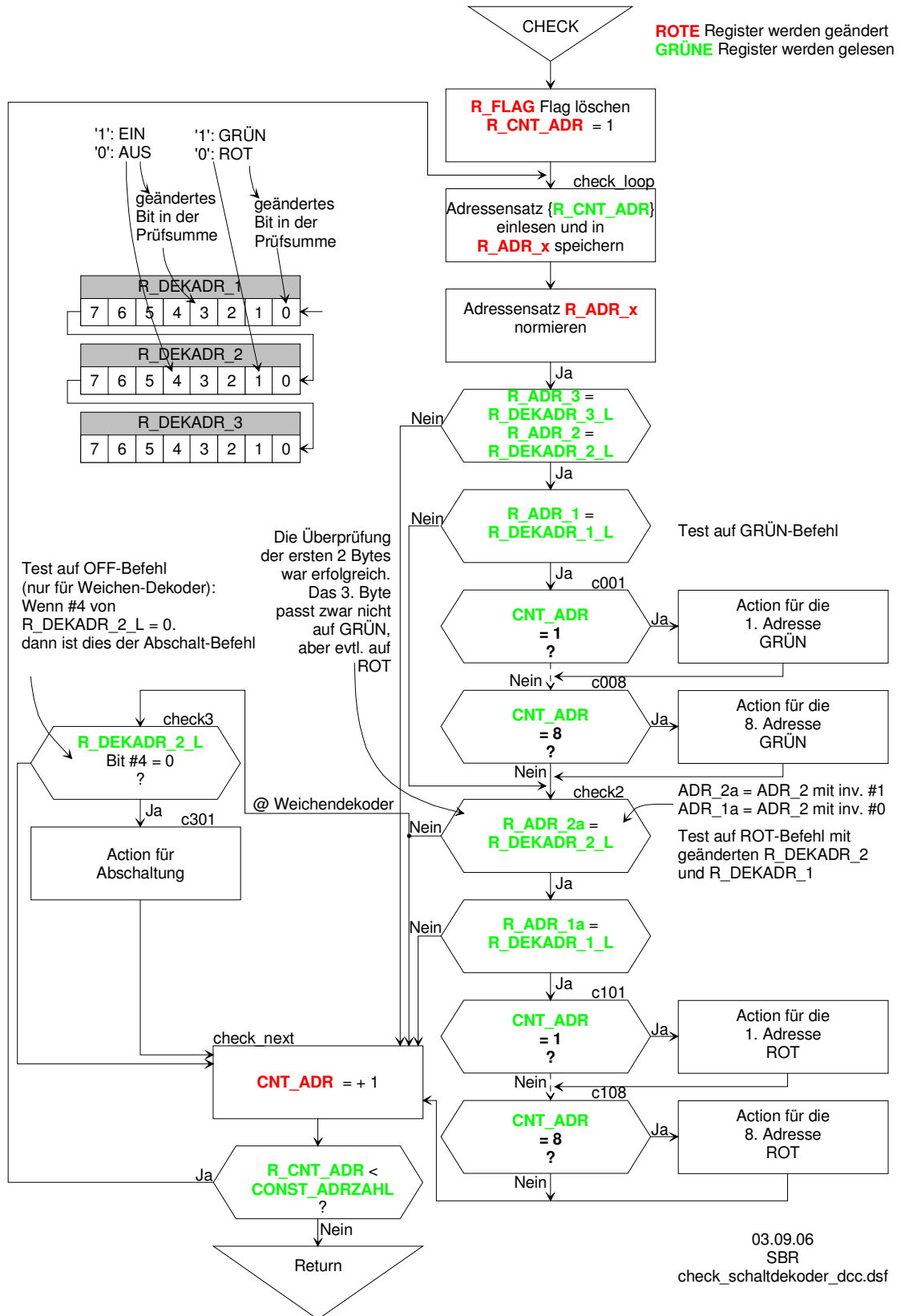


Abbildung 5: Routine "check" (Schaltdekoer, DCC)

## 4.3 Unterprogramm „POLARITY“ für DCC (Schaltbefehle)

Hat keine richtige Funktion im DCC-Format, Polarität wird immer auf „normal“ gesetzt (R\_POL#0 = „0“)

## 5 Beschreibung der Routinen für DCC (Funktionsdekoder)

### 5.1 ISR-Routine DCC (Funktionsdekoder)

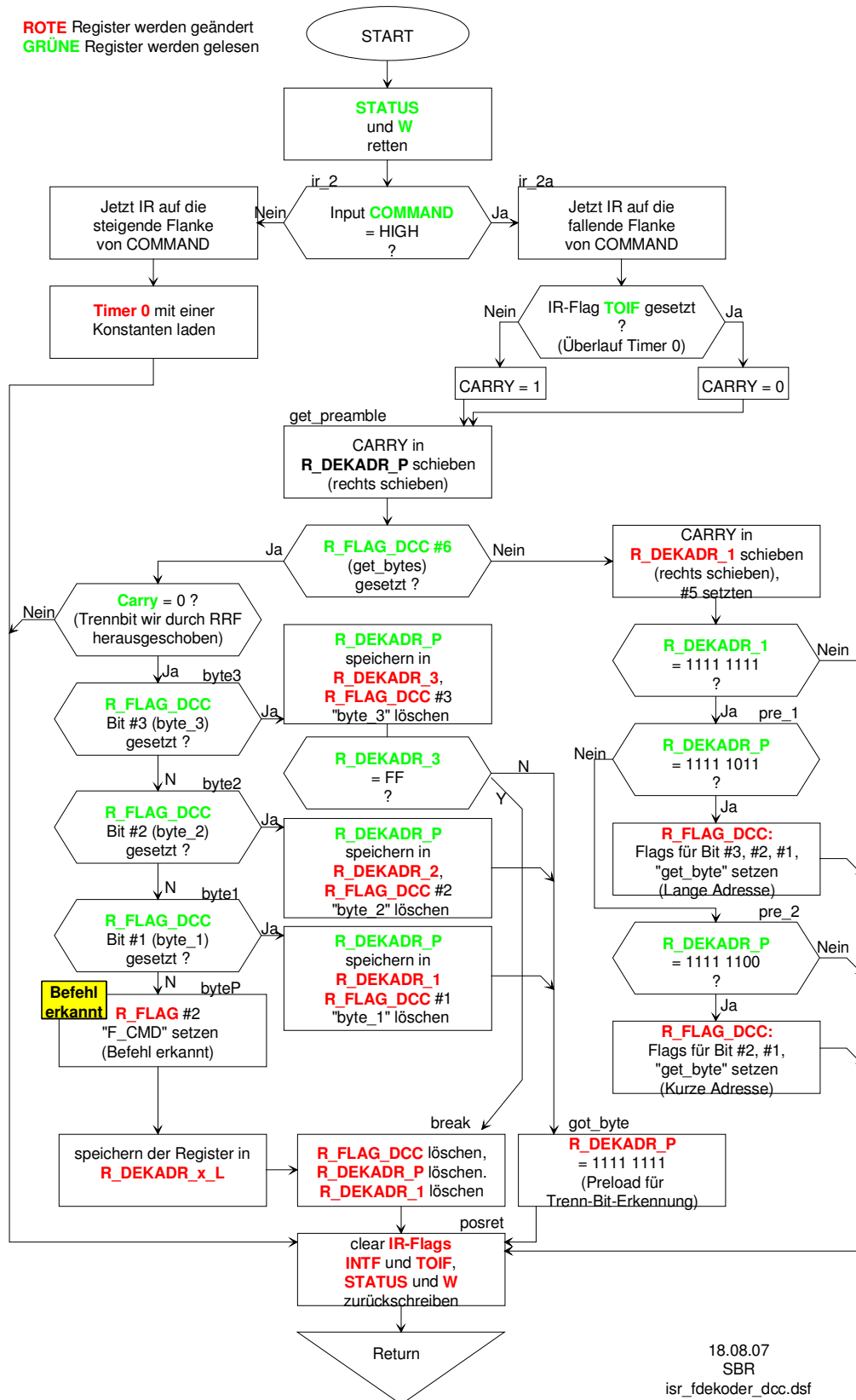
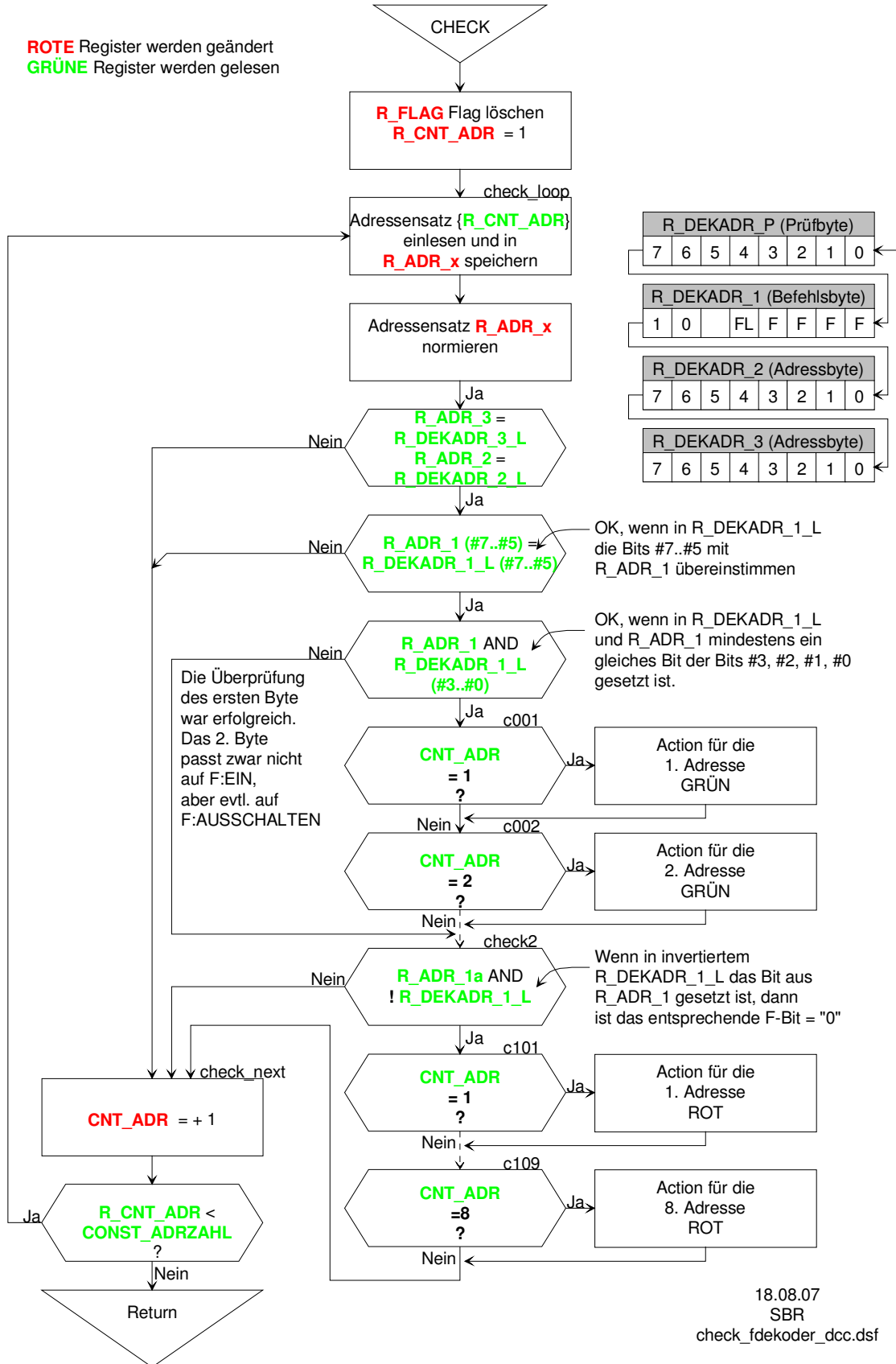


Abbildung 6: ISR für Funktionsdekoder DCC

**5.2 Routine „check“ für DCC (Funktionsdekoder)**

**ROTE** Register werden geändert  
**GRÜNE** Register werden gelesen



18.08.07  
SBR

check\_fdekoder\_dcc.dsf

**Abbildung 7: Routine „check“ (Funktionsdekoder, DCC)**

## **5.3 Unterprogramm „POLARITY“ für DCC (Funktionsdekoder)**

Hat keine richtige Funktion im DCC-Format, Polarität wird immer auf „normal“ gesetzt (R\_POL#0 = „0“)

## 6 Funktionsbeschreibungen

### 6.1 Motorola-Befehle

#### 6.1.1 Schaltdecoder MM

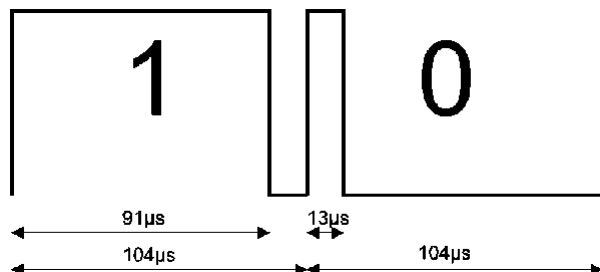


Abbildung 8: Motorola Timing (Schaltbefehle)

Deko der #	von	bis	0/1	2/3	4/5	6/7	8/9	10/ 11	12/ 13	14/ 16	16/ 17			R1	R2	R3
			R1	R2				R3				Weiche	Spule			
1	1	4	11	00	00	00	00	11	00	00	11	1	1 (grün)	03	00	C3
			00					00	00	00	11	1	2 (rot)	03	00	03
			11					11	11	00	11	2	1 (grün)	03	00	F3
			00					00	11	00	11	2	2 (rot)	03	00	33
			11					11	00	11	11	3	1 (grün)	03	00	CF
			00					00	00	11	11	3	2 (rot)	03	00	0F
			11					11	11	11	11	4	1 (grün)	03	00	FF
			00					00	11	11	11	4	2 (rot)	03	00	3F
						00	00	00	00	1-4	aus	03	00	00		
2	5	8	10	00	00	00	00							02	00	00
3	9	12	00	11	00	00	00							00	C0	00
4	13	16	11	11	00	00	00							03	C0	00
5	17	20	10	11	00	00	00							02	C0	00
6	21	24	00	10	00	00	00							00	80	00
7	25	28	11	10	00	00	00					wie	oben	03	80	00
8	29	32	10	10	00	00	00							02	80	00
9	33	36	00	00	11	00	00							00	30	00
10	37	40	11	00	11	00	00							03	30	00
11	41	44	10	00	11	00	00							02	30	00
12	45	48	00	11	11	00	00							00	F0	00
13	49	52	11	11	11	00	00							03	F0	00
14	53	56	10	11	11	00	00							02	F0	00
15	57	60	00	10	11	00	00							00	B0	00
16	61	64	11	10	11	00	00							03	B0	00
17	65	68	10	10	11	00	00							02	B0	00
18	69	72	00	00	10	00	00							00	20	00
19	73	76	11	00	10	00	00							03	20	00
20	77	80	10	00	10	00	00							02	20	00
21	81	84	00	11	10	00	00							00	E0	00
22	85	88	11	11	10	00	00							03	E0	00
23	89	92	10	11	10	00	00							02	E0	00
24	93	96	00	10	10	00	00							00	A0	00
25	97	100	11	10	10	00	00							03	A0	00

Deko der # von bis	0/1	2/3	4/5	6/7	8/9	10/ 11	12/ 13	14/ 16	16/ 17			R1	R2	R3
	R1	R2			R3			Weiche	Spule					
26	101	104	10	10	10	00	00					02	A0	00
27	105	108	00	00	00	11	00					00	0C	00
28	109	112	11	00	00	11	00					03	0C	00
29	113	116	10	00	00	11	00					02	0C	00
30	117	120	00	11	00	11	00					00	CC	00
31	121	124	11	11	00	11	00					03	CC	00
32	125	128	10	11	00	11	00					02	CC	00
33	129	132	00	10	00	11	00					00	8C	00
34	133	136	11	10	00	11	00					03	8C	00
35	137	140	10	10	00	11	00					02	8C	00
36	141	144	00	00	11	11	00					00	3C	00
37	145	148	11	00	11	11	00					03	3C	00
38	149	152	10	00	11	11	00					02	3C	00
39	153	156	00	11	11	11	00					00	FC	00
40	157	160	11	11	11	11	00					03	FC	00
41	161	164	10	11	11	11	00					02	FC	00
42	165	168	00	10	11	11	00					00	BC	00
43	169	172	11	10	11	11	00					03	BC	00
44	173	176	10	10	11	11	00					02	BC	00
45	177	180	00	00	10	11	00					00	2C	00
46	181	184	11	00	10	11	00					03	2C	00
47	185	188	10	00	10	11	00					02	2C	00
48	189	192	00	11	10	11	00					00	EC	00
49	193	196	11	11	10	11	00					03	EC	00
50	197	200	10	11	10	11	00					02	EC	00
51	201	204	00	10	10	11	00					00	AC	00
52	205	208	11	10	10	11	00					03	AC	00
53	209	212	10	10	10	11	00					02	AC	00
54	213	216	00	00	00	10	00					00	08	00
55	217	220	11	00	00	10	00					03	08	00
56	221	224	10	00	00	10	00					02	08	00
57	225	228	00	11	00	10	00					00	C8	00
58	229	232	11	11	00	10	00					03	C8	00
59	233	236	10	11	00	10	00					02	C8	00
60	237	240	00	10	00	10	00					00	88	00
61	241	244	11	10	00	10	00					03	88	00
62	245	248	10	10	00	10	00					02	88	00
63	249	252	00	00	11	10	00					00	38	00
64	253	256	11	00	11	10	00					03	38	00

**Tabelle 1: Registerbelegung für Motorola Schaltbefehle**

## 6.1.2 Funktionsdekoder MM

Die Funktions-Befehle sind in den Fahrzeug-Befehle integriert. Das Timing dieser ist doppelt so lang wie bei den Schaltdekodern.

Dekoder # Lokadresse	0/1	2/3	4/5	6/7	8/9	R_DEKADR_3_L	R_DEKADR_3_L	R_DEKADR_3_L
	R_DE KADR 1 L	R_DEKADR_2_L				hex	hex	hex
1	11	00	00	00		03	00	
2	10	00	00	00		02	00	
3	00	11	00	00		00	C0	
4	11	11	00	00		03	C0	siehe folgende Tabellen
5	10	11	00	00		02	C0	
6	00	10	00	00		00	80	
7	11	10	00	00		03	80	
8	10	10	00	00		02	80	
9	00	00	11	00		00	30	
10	11	00	11	00		03	30	
11	10	00	11	00		02	30	
12	00	11	11	00		00	F0	
13	11	11	11	00		03	F0	
14	10	11	11	00		02	F0	
15	00	10	11	00		00	B0	
16	11	10	11	00		03	B0	
17	10	10	11	00		02	B0	
18	00	00	10	00		00	20	
19	11	00	10	00		03	20	
20	10	00	10	00		02	20	
21	00	11	10	00		00	E0	
22	11	11	10	00		03	E0	
23	10	11	10	00		02	E0	
24	00	10	10	00		00	A0	
25	11	10	10	00		03	A0	
26	10	10	10	00		02	A0	
27	00	00	00	11		00	0C	
28	11	00	00	11		03	0C	
29	10	00	00	11		02	0C	
30	00	11	00	11		00	CC	
31	11	11	00	11		03	CC	
32	10	11	00	11		02	CC	
33	00	10	00	11		00	8C	
34	11	10	00	11		03	8C	
35	10	10	00	11		02	8C	
36	00	00	11	11		00	3C	
37	11	00	11	11		03	3C	
38	10	00	11	11		02	3C	
39	00	11	11	11		00	FC	
40	11	11	11	11		03	FC	
41	10	11	11	11		02	FC	
42	00	10	11	11		00	BC	
43	11	10	11	11		03	BC	
44	10	10	11	11		02	BC	
45	00	00	10	11		00	2C	
46	11	00	10	11		03	2C	
47	10	00	10	11		02	2C	
48	00	11	10	11		00	EC	
49	11	11	10	11		03	EC	
50	10	11	10	11		02	EC	
51	00	10	10	11		00	AC	
52	11	10	10	11		03	AC	
53	10	10	10	11		02	AC	
54	00	00	00	10		00	08	
55	11	00	00	10		03	08	
56	10	00	00	10		02	08	
57	00	11	00	10		00	C8	
58	11	11	00	10		03	C8	
59	10	11	00	10		02	C8	
60	00	10	00	10		00	88	
61	11	10	00	10		03	88	
62	10	10	00	10		02	88	
63	00	00	11	10		00	38	
64	11	00	11	10		03	38	
65	10	00	11	10				
80	00	00	00	00				
81	01	00	00	00				
82	01	10	00	00				
! 191 !	01	01	00	00				
84	01	11	00	00				
85	01	00	10	00				
86	01	10	10	00				
87	01	01	10	00				

11 = Function ON / 00 = Function OFF

Dekoder # Lokadresse	0/1	2/3	4/5	6/7	8/9	R_DEKADR_3_L	R_DEKADR_3_L	R_DEKADR_3_L
	R_DE KADR 1_L	R_DEKADR_2_L				hex	hex	hex
88	01	11	10	00				
89	01	00	01	00				
90	01	10	01	00				
91	01	01	01	00				
92	01	11	01	00				
93	01	00	11	00				
94	01	10	11	00				
95	01	01	11	00				
96	01	11	11	00				
97	01	00	00	10				
98	01	10	00	10				
99	01	01	00	10				
100	01	11	00	10				
101	01	00	10	10				
102	01	10	10	10				
103	01	01	10	10				
104	01	11	10	10				
105	01	00	01	10				
106	01	10	01	10				
107	01	01	01	10				
108	01	11	01	10				
109	01	00	11	10				
110	01	10	11	10				
111	01	01	11	10				
112	01	11	11	10				
113	01	00	00	01				
114	01	10	00	01				
115	01	01	00	01				
116	01	11	00	01				
117	01	00	10	01				
118	01	10	10	01				
119	01	01	10	01				
120	01	11	10	01				
121	01	00	01	01				
122	01	10	01	01				
! 192 !	01	01	01	01				
124	01	11	01	01				
125	01	00	11	01				
126	01	10	11	01				
127	01	01	11	01				
128	01	11	11	01				
129	01	00	00	11				
130	01	10	00	11				
131	01	01	00	11				
132	01	11	00	11				
133	01	00	10	11				
134	01	10	10	11				
135	01	01	10	11				
136	01	11	10	11				
137	01	00	01	11				
138	01	10	01	11				
139	01	01	01	11				
140	01	11	01	11				
141	01	00	11	11				
142	01	10	11	11				
143	01	01	11	11				
144	01	11	11	11				
145	00	01	00	00				
146	00	01	10	00				
147	00	01	01	00				
148	00	01	11	00				
149	00	01	00	10				
150	00	01	10	10				
151	00	01	01	10				
152	00	01	11	10				
153	00	01	00	01				
154	00	01	10	01				
155	00	01	01	01				
156	00	01	11	01				
157	00	01	00	11				
158	00	01	10	11				
159	00	01	01	11				
160	00	01	11	11				
161	11	01	00	00				
162	11	01	10	00				
163	11	01	01	00				
164	11	01	11	00				
165	11	01	00	10				
166	11	01	10	10				
167	11	01	01	10				
168	11	01	11	10				

Dekoder # Lokadresse	0/1	2/3	4/5	6/7	8/9	R_DEKADR_3_L	R_DEKADR_3_L	R_DEKADR_3_L
	R_DE KADR 1 L	R_DEKADR_2_L				hex	hex	hex
169	11	01	00	01				
170	11	01	10	01				
171	11	01	01	01				
172	11	01	11	01				
173	11	01	00	11				
174	11	01	10	11				
175	11	01	01	11				
176	11	01	11	11				
177	10	01	00	00				
178	10	01	10	00				
179	10	01	01	00				
180	10	01	11	00				
181	10	01	00	10				
182	10	01	10	10				
183	10	01	01	10				
184	10	01	11	10				
185	10	01	00	01				
186	10	01	10	01				
187	10	01	01	01				
188	10	01	11	01				
189	10	01	00	11				
190	10	01	10	11				
! 83 !	10	01	01	11				
! 123 !	10	01	11	11				
193	00	00	01	00				
194	00	00	01	10				
195	00	00	01	01				
196	00	00	01	11				
197	11	00	01	00				
198	11	00	01	10				
199	11	00	01	01				
200	11	00	01	11				
201	10	00	01	00				
202	10	00	01	10				
203	10	00	01	01				
204	10	00	01	11				
205	00	11	01	00				
206	00	11	01	10				
207	00	11	01	01				
208	00	11	01	11				
209	11	11	01	00				
210	11	11	01	10				
211	11	11	01	01				
212	11	11	01	11				
213	10	11	01	00				
214	10	11	01	10				
215	10	11	01	01				
216	10	11	01	11				
217	00	10	01	00				
218	00	10	01	10				
219	00	10	01	01				
220	00	10	01	11				
221	11	10	01	00				
222	11	10	01	10				
223	11	10	01	01				
224	11	10	01	11				
225	10	10	01	00				
226	10	10	01	10				
227	10	10	01	01				
228	10	10	01	11				
229	00	00	00	01				
230	11	00	00	01				
231	10	00	00	01				
232	00	11	00	01				
233	11	11	00	01				
234	10	11	00	01				
235	00	10	00	01				
236	11	10	00	01				
237	10	10	00	01				
238	00	00	11	01				
239	11	00	11	01				
240	10	00	11	01				
241	00	11	11	01				
242	11	11	11	01				
243	10	11	11	01				
244	00	10	11	01				
245	11	10	11	01				
246	10	10	11	01				
247	00	00	10	01				
248	11	00	10	01				
249	10	00	10	01				

Dekoder # Lokadresse	0/1	2/3	4/5	6/7	8/9	R_DEKADR_3_L	R_DEKADR_3_L	R_DEKADR_3_L
	R_DEKADR_1_L	R_DEKADR_2_L				hex	hex	hex
250	00	11	10	01				
251	11	11	10	01				
252	10	11	10	01				
253	00	10	10	01				
254	11	10	10	01				
255	10	10	10	01				
???	10	10	10	10				

**Tabelle 2: Adressteil MM – Funktionsdekoder (Lokdekoder)**

Speed Step	R_DEKADR_3_L							
	7	6	5	4	3	2	1	0
14	1		1		1		1	
13	0		1		1		1	
2	1		1		0		0	
1	0		1		0		0	
Reverse	1		0		0		0	
Stopp	0		0		0		0	

**Tabelle 3: Speed Information**

Die Geschwindigkeit ist Mot 1 kompatibel jeweils im ersten Bit des Pärchens abgelegt. Die Fahrrichtungs-Info ist nicht absolut.

Geschwindigkeits - Bereich	R_DEKADR_3_L							
	7	6	5	4	3	2	1	0
-14 .. -7		1		0		1		0
-6 .. -0		1		0		1		1
+0 .. +6		0		1		0		1
+7 .. +14		0		1		0		0

**Tabelle 4: Speed Information**

Zusatz-Bytes für Mot 2: Hier werden die 2. Bits der Pärchen verwendet. Es ist zum einen die Fahrtrichtung absolut definiert, zum zweiten ist die Anzahl der Fahrstufen auf 28 je Fahrtrichtung erhöht worden.

Funktion	R_DEKADR_3_L							
	7	6	5	4	3	2	1	0
F1		1		1		0		X
F2		0		0		1		X
F3		0		1		1		X
F4		1		1		1		X
.. ON								1
.. OFF								0

**Tabelle 5: Status F1..F4**

Weiterhin ist in den 2. Bits der Pärchen der Status der Funktionstasten F1..F4 abgelegt.

6.2 DCC Befehle

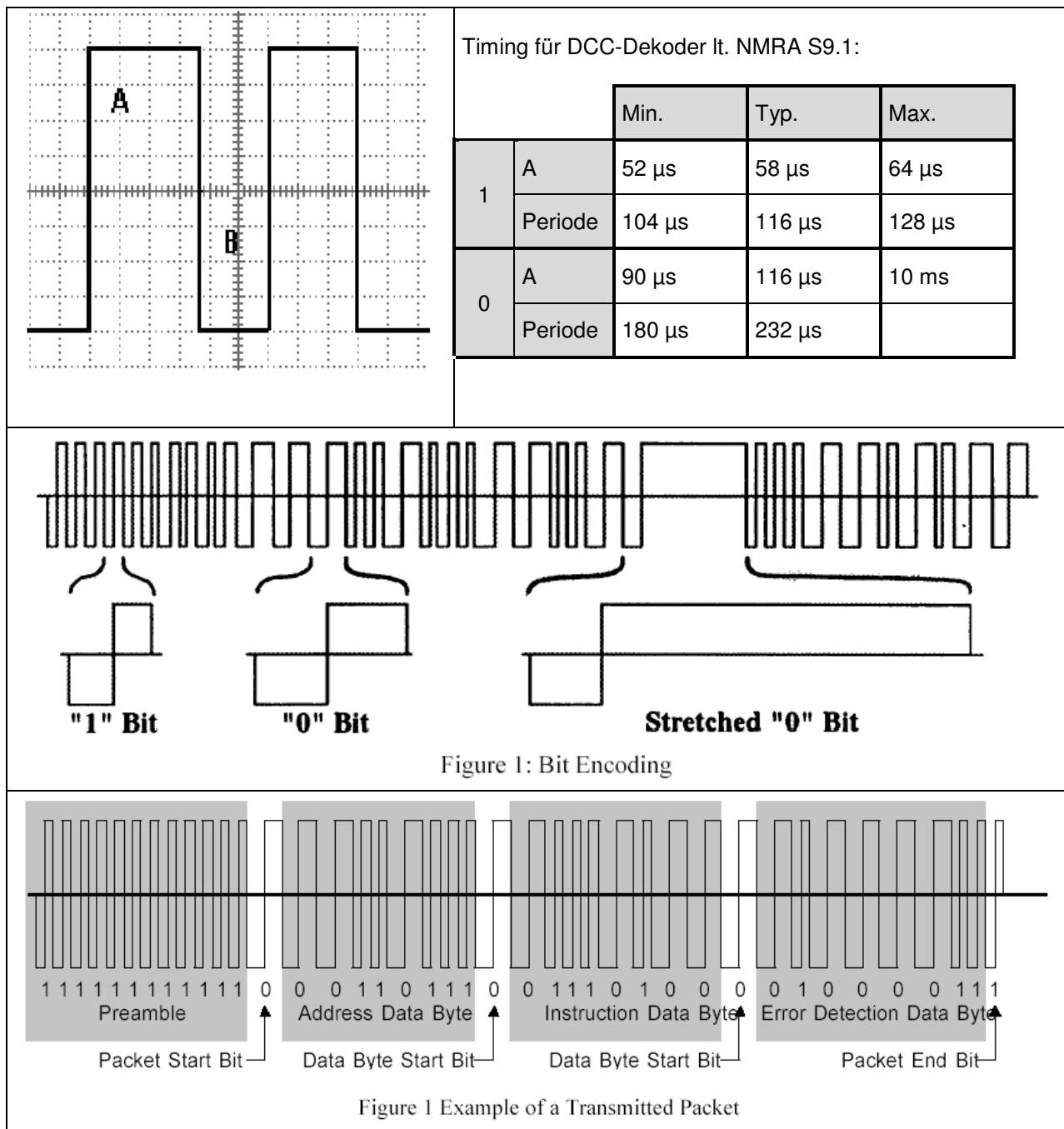
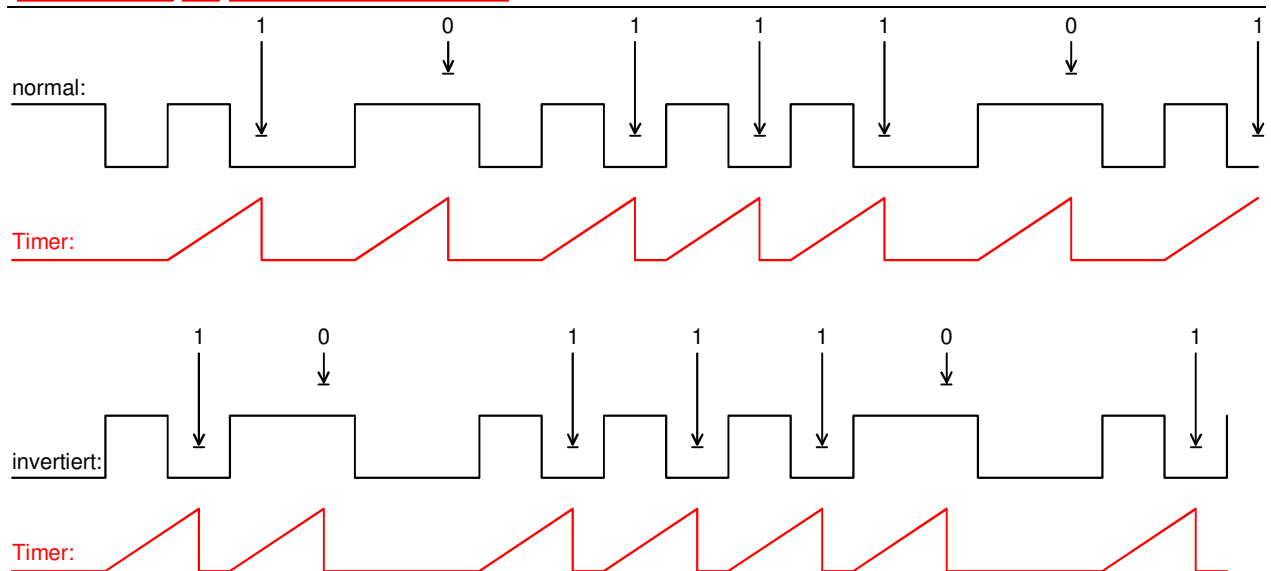


Abbildung 9: DCC Timing aus NMRA S 9.1 / 9.2

	Periode Min.	Periode Max.
„1“		140 µs
„0“	170 µs	

Tabelle 6: Implementierte Timing-Parameter



**Abbildung 10: Befehls-erkennung DCC (weitere Idee, nicht implementiert)**

Idee für Befehls-erkennung (**verworfen**, dann dann der Timer-IR benötigt und dieser bei MM nicht verwendet wird):

Steigende Flanke am COMMAND-Pin löst IR aus, Startet in ISR den Timer.

Nach Ablauf des Timers löst dieser IR aus. in ISR wird Polarität des COMMAND-Signals betrachtet.

### 6.2.1 Schaltdekoder DCC

		Preamble								0	Adressbyte							0	Befehlsbyte							0	Prüfbyte														
←		1	1	1	1	1	1	1	1	1	0	1	0	A	A	A	A	A	A	0	1	A	A	A	C	D	D	D	0	P	P	P	P	P	P	P	P				
		R_DEKADR 5								R_DEKADR 4								R_DEKADR 3 L								R_DEKADR 2 L								R_DEKADR 1 L							
R1		1	1	1	1	1	1	1	1	1	0	1	0	0000 0	0101 5	1	1	1	x	0	0	0	0	0	1	1	1	x	0	0	1	E/F	0	7							
G1		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	0	0	1	0	0	1	1	1	x	0	0	0	0	E/F	2	7						
R2		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	0	1	0	0	0	1	1	1	x	0	1	1	1	E/F	4	7						
G2		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	0	1	1	0	0	1	1	1	x	0	1	1	0	E/F	6	7						
R3		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	0	0	0	0	1	1	1	x	1	0	1	1	E/F	8	7						
G3		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	0	1	0	0	1	1	1	x	1	0	0	0	E/F	A	7						
R4		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	1	0	0	0	1	1	1	x	1	1	1	1	E/F	C	7						
G4		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	1	1	0	0	1	1	1	x	1	1	1	0	E/F	E	7						
R5		1	1	1	1	1	1	1	1	1	0	1	0	0000 0	1001 9	1	1	1	x	0	0	0	0	7	1010 = A																
G5		1	1	1	1	1	1	1	1	0	1	0	1			1	1	x	0	0	1	0	7	1011 = B																	
R6		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	0	1	0	0	7	1000 = 8																
G6		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	0	1	1	0	7	1001 = 9																
R7		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	0	0	0	7	1110 = E																
G7		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	0	1	0	7	1111 = F																
R8		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	1	0	0	7	1100 = C																
G8		1	1	1	1	1	1	1	1	1	0	1	0			1	1	1	x	1	1	1	0	7	1101 = D																
R9		1	1	1	1	1	1	1	1	1	0	1	0	0000	1101	1	1	1	x	0	0	0	0																		
R12		1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	1	1	x										E/F	1									
R13		1	1	1	1	1	1	1	1	1	0	1	0	0001 1	D	0	1	0	1	1	1	1	x									E/F	5								
R16		1	1	1	1	1	1	1	1	1	0	1	0			1	0	0	1	1	1	1	x										E/F	9							
R17		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R20		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R21		1	1	1	1	1	1	1	1	1	0	1	0	0010 2	D	0	1	0	1	1	1	1	x									E/F	5								
R24		1	1	1	1	1	1	1	1	1	0	1	0			1	0	0	1	1	1	1	x										E/F	9							
R25		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R28		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R29		1	1	1	1	1	1	1	1	1	0	1	0			0	1	0	1	1	1	1	x										E/F	5							
R32		1	1	1	1	1	1	1	1	1	0	1	0			1	0	0	1	1	1	1	x										E/F	9							
R33		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R36		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R37		1	1	1	1	1	1	1	1	1	0	1	0	0011 3	D	0	1	0	1	1	1	1	x									E/F	5								
R40		1	1	1	1	1	1	1	1	1	0	1	0			1	0	0	1	1	1	1	x										E/F	9							
R41		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R44		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R45		1	1	1	1	1	1	1	1	1	0	1	0	0100 4	D	0	1	0	1	1	1	1	x									E/F	5								
R48		1	1	1	1	1	1	1	1	1	0	1	0			1	0	0	1	1	1	1	x										E/F	9							
R49		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R52		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R53		1	1	1	1	1	1	1	1	1	0	1	0	0100 4	D	1	1	0	1	1	1	1	x									E/F	5								
R56		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	9							
R57		1	1	1	1	1	1	1	1	1	0	1	0			1	1	0	1	1	1	1	x										E/F	D							
R60		1	1	1	1	1	1	1	1	1	0	1	0			0	0	0	1	1	1	1	x										E/F	1							
R61		1	1	1	1	1	1	1	1	1	0	1	0																			E/F	1								
R64		1	1	1	1	1	1	1	1	1	0	1	0																			E/F	1								

**Tabelle 7: Implementierung bei DCC Schaltdekoder**

x: 1 = ein, 0 = aus

Prüfbyte ist die EOR Verknüpfung von Adressbyte und Befehlsbyte

Befehlsbyte Bit 6..4	Uhlenbrock IBox		DCC-Norm / Multimaus	
	Adr. von	Adr. bis	Adr. von	Adr. bis
111	1	252	1	256
110	253	508	257	512
101	509	764	513	768
100	765	1020	769	1024
011	1021	1276	1025	1280
010	1277	1532	1281	1536
001	1533	1788	1537	1792
000	1789	2044	1793	2048

Tabelle 8: Definition der Adress-Bereiche über das Befehls-Byte

Adresse Schaltdekode		Preamble										Adressbyte (Address Data Byte)								Befehlsbyte (Instruction Data Byte)								Prüfbyte (Error-Byte)									
Uhlenbrock Ibox	Roco	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0		
-	1..4	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	A	A	A	C	D	D	D	0	E	E	E	E	E	E	E	E
R1	R5	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	Das Prüfbyte ist die EXOR-Verknüpfung von Adress-Byte und Befehls-Byte			
G1	G5	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
R2	R6	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	1	0	0	0	0	0				
G2	G6	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	1	1	0	0	0	0				
R3	R7	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	0	0	0	0	0	0				
G3	G7	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	0	0	0				
R4	R8	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	1	0	0	0	0	0				
G4	G8	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	1	1	0	0	0	0				
5	9	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
6	10	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	1	0	0	0	0	0				
7	11	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	0	0	0	0	0	0				
8	12	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	1	1	0	0	0	0	0				
9..16	13..20	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
17..24	21..28	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
25..32	29..36	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
33..40	37..44	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
41..48	45..52	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
49..56	53..60	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
57..120	61..124	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
121..184	125..188	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
185..252	189..256	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0				
253..508	257..512	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0				
509..764	513..768	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0				
765..1020	769..1024	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0				
1021..1276	1025..1280	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0				
1277..1532	1281..1536	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0				
1533..1788	1537..1792	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0				
1789..2044	1793..2048	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0				
2044	2048	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0				

Tabelle 9: Preamble, Adressteil und Befehlssteil DCC – Schaltdekode

**6.2.2 Funktionsdekode DCC**

		R_DEKADR_2_L																	
Adresse	Preamble										Adressbyte 1 (Address Data Byte)								
	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
	1	1	1	1	1	1	1	1	1	1	0	0	A	A	A	A	A	A	0
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0
127	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0

Dekoder-Realisierung:  
R\_DEKADR\_3\_L ist hier „00“

**Tabelle 10: Preamble und Adressteil DCC – kurze Adressen**

Es stehen hier 7 Bits zur Verfügung, d.h. es sind  $2^7 = 128$  Adressen ansprechbar.

		R_DEKADR_3_L										R_DEKADR_2_L														
Adresse	Preamble										Adressbyte 1 (Address Data Byte 1)								Adressbyte 2 (Address Data Byte 2)							
	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	1	1	1	1	1	1	1	1	1	1	0	1	1	A	A	A	A	0	0	A	A	A	A	A	A	0
128	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0
129	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	1
255	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0
256	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
512	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0
9999	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0
16127	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0

**Tabelle 11: Preamble und Adressteil DCC – lange Adressen**

Es stehen hier jetzt 14 Bits zur Verfügung, d.h. es sind  $2^{14} = 16384$  Adressen ansprechbar (aber Adress-Byte 1 = ‚FF‘ ist nicht zulässig, daher nur 16128 Adressen möglich). Verwendet für Adressen > 128 Das Prüfbyte ist hier die EXOR-Verknüpfung der beiden Adress-Bytes und des Befehls-Bytes

		R_DEKADR_1_L								R_DEKADR_P_L							
Funktion	Befehlsbyte (Instruction Data Byte)								Prüfbyte (Error-Byte)								
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
	1	0	D	D	D	D	D	0	E	E	E	E	E	E	E	0	
FL	1	0	0	1				0								0	
F1	1	0	0				1	0								0	
F2	1	0	0			1		0								0	
F3	1	0	0		1			0								0	
F4	1	0	0		1			0								0	
F5	1	0	1	1			1	0								0	
F6	1	0	1	1			1	0								0	
F7	1	0	1	1		1		0								0	
F8	1	0	1	1	1			0								0	
F9	1	0	1	0			1	0								0	
F10	1	0	1	0			1	0								0	
F11	1	0	1	0		1		0								0	
F12	1	0	1	0	1			0								0	

**Rules:**  
R3 kann nicht ‚FF‘ sein!  
R1 kann nicht ‚00‘ sein!  
R1 kann nicht ‚FF‘ sein!

**Tabelle 12: Befehlsbyte und Prüfbyte DCC**

Die fahrtrichtungsabhängige Funktion FL setzt das Bit #4 im Befehlsbyte nur, wenn das DCC-28 oder DCC-128 Format verwendet wird (dann ist Bit #1 von CV29 =“1“!). Zusätzlich muss dafür die Fahrtrichtung aus den Fahrbefehlen abgeleitet werden.

Funktion	Befehlsbyte (Instruction Data Byte)								Expansion Byte								R_DEKADR_P Prüfbyte (Error-Byte)								
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
	1	1	0	1	1	1	1	C	0	D	D	D	D	D	D	D	0	P	P	P	P	P	P	P	
F13	1	1	0	1	1	1	1	0	0																1
F14	1	1	0	1	1	1	1	0	0																1
F15	1	1	0	1	1	1	1	0	0																1
F16	1	1	0	1	1	1	1	0	0																1
F17	1	1	0	1	1	1	1	0	0																1
F18	1	1	0	1	1	1	1	0	0																1
F19	1	1	0	1	1	1	1	0	0																1
F20	1	1	0	1	1	1	1	0	0	1															1
F21	1	1	0	1	1	1	1	1	0																1
F22	1	1	0	1	1	1	1	1	0																1
F23	1	1	0	1	1	1	1	1	0																1
F24	1	1	0	1	1	1	1	1	0																1
F25	1	1	0	1	1	1	1	1	0																1
F26	1	1	0	1	1	1	1	1	0																1
F27	1	1	0	1	1	1	1	1	0																1
F28	1	1	0	1	1	1	1	1	0	1															1

Das Prüfbyte ist die EXOR-Verknüpfung von Adress-Byte(s) und Befehls-Byte

Dekoder-Realisierung:  
Die erweiterten Funktionen sind nicht implementiert.

**Tabelle 13: Befehlsbyte und Prüfbyte DCC – erweiterte Funktionen**

## 7 EEPROM Speicherbelegung

### 7.1 Motorola Schaltdekoder

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	RM	A01 _1	A02 _1	A03 _1	A04 _1	A05 _1	A06 _1	A07 _1	A08 _1							
	0x01	0x03	0x03	0x03	0x03	0x02	0x02	0x02	0x02	0xff	0xff	0xff	0xff	0xff	0xff	0xff
10	RS	A01 _2	A02 _2	A03 _2	A04 _2	A05 _2	A06 _2	A07 _2	A08 _2							
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xff	0xff	0xff	0xff	0xff	0xff	0xff
20		A01 _3	A02 _3	A03 _3	A04 _3	A05 _3	A06 _3	A07 _3	A08 _3							
	0xff	0xc3	0xf3	0xcf	0xff	0xc3	0xf3	0xcf	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
30																
	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
		01 g	02 g	03 g	04 g	05 g	06 g	07 g	08 g							
40																
	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
50																
	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
60																
	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
70																
	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff

**Tabelle 14: EEprom Speicherbelegung bei Motorola Schaltdekoder**

--: not used  
 RS: R\_STATUS (Status)  
 RM: R\_MODE (Mode)  
 A01..A08: Addresses

## 7.2 Motorola F-Dekoder

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	RM	A01 _1	A02 _1	A03 _1	A04 _1	A05 _1	A06 _1	A07 _1	A08 _1							
	0x01	0x03	0x03	0x03	0x03	0x02	0x02	0x02	0x02	0xff	0xff	0xff	0xff	0xff	0xff	0xff
10	RS	A01 _2	A02 _2	A03 _2	A04 _2	A05 _2	A06 _2	A07 _2	A08 _2							
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xff	0xff	0xff	0xff	0xff	0xff	0xff
20		A01 _3	A02 _3	A03 _3	A04 _3	A05 _3	A06 _3	A07 _3	A08 _3							
	0xff	0x51	0x05	0x15	0x55	0x51	0x05	0x15	0x55	0xff	0xff	0xff	0xff	0xff	0xff	0xff
30	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
		01 F1	01 F2	01 F3	01 F4	02 F1	02 F2	02 F3	02 F4							
40	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
50	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
60	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
70	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff

**Tabelle 15: EEprom Speicherbelegung bei Motorola F-Dekoder**

--: not used  
 RS: R\_STATUS (Status)  
 RM R\_MODE (Mode)  
 A01..A08: Addresses

## 7.3 DCC Schaltdekoder

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	RM	A01 _1	A02 _1	A03 _1	A04 _1	A05 _1	A06 _1	A07 _1	A08 _1							
	0x01	0x78	0x7a	0x7c	0x7e	0x7b	0x79	0x7f	0x7d	0xff	0xff	0xff	0xff	0xff	0xff	0xff
10	RS	A01 _2	A02 _2	A03 _2	A04 _2	A05 _2	A06 _2	A07 _2	A08 _2							
	0x00	0xf2	0xf6	0xfa	0xfe	0xf2	0xf6	0xfa	0xfe	0xff	0xff	0xff	0xff	0xff	0xff	0xff
20		A01 _3	A02 _3	A03 _3	A04 _3	A05 _3	A06 _3	A07 _3	A08 _3							
	0xff	0x05	0x05	0x05	0x05	0x09	0x09	0x09	0x09	0xff	0xff	0xff	0xff	0xff	0xff	0xff
30	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
		01 g	02 g	03 g	04 g	05 g	06 g	07 g	08 g							
40	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
50	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
60	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
70	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff

**Tabelle 16: EEprom Speicherbelegung bei DCC Schaltdekoder**

--: not used  
 RS: R\_STATUS (Status)  
 RM R\_MODE (Mode)  
 A01..A08: Addresses

## 7.4 DCC F-Dekoder

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	RM	A01 _1	A02 _1	A03 _1	A04 _1	A05 _1	A06 _1	A07 _1	A08 _1							
	0x01	0x81	0x82	0x84	0x88	0xb1	0xb2	0xb4	0xb9	0xff	0xff	0xff	0xff	0xff	0xff	0xff
10	RS	A01 _2	A02 _2	A03 _2	A04 _2	A05 _2	A06 _2	A07 _2	A08 _2							
	0x00	0x01	0x01	0x01	0x01	0x01	0x01	0x01	0x01	0xff	0xff	0xff	0xff	0xff	0xff	0xff
20		A01 _3	A02 _3	A03 _3	A04 _3	A05 _3	A06 _3	A07 _3	A08 _3							
	0xff	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xff	0xff	0xff	0xff	0xff	0xff	0xff
30	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
		01 F1	01 F2	01 F3	01 F4	01 F5	01 F6	01 F7	01 F8							
40	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
50	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
60	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff
70	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff	0xff

**Tabelle 17: EEPROM Speicherbelegung bei DCC F-Dekoder**

--: not used  
 RS: R\_STATUS (Status)  
 RM R\_MODE (Mode)  
 A01..A08: Addresses