

SAnD-4

SW-Beschreibung (Stand V3.35)

Technische Dokumentation

Entwicklungsgrundlage

Dies ist keine Aufbau-Anleitung oder Bedienungs-Anleitung. Die hier beschriebenen Informationen sind für den Aufbau und den Betrieb des Moduls normalerweise nicht nötig.

INHALT:

1	Grundsätzliches	2
2	Verwendete Register	4
2.1	Reservierte Register f. Dekoder-Funktion.....	4
2.2	Reservierte Register f. Servo-Dekoder	5
3	Beschreibung der Software-Routinen	7
3.1	Routine „restore_servo“	7
3.2	Routine „update_4094“	9
3.3	Routine „UMUL0707L“	9
3.4	Routine „generate_servo_impuls“.....	10
3.5	Routine „change_mode“	11
3.6	Routine „check_switches“	12
3.7	Routine „statemachine_sand_4“ (State Machine)	13
4	Funktionsbeschreibungen	15
4.1	Parametrisierung.....	15
4.2	Servo Impuls-Generierung.....	16
4.3	Totzeit-Generierung	17
5	HEX Speicherbelegung	18
6	EEPROM Speicherbelegung	20

1 Grundsätzliches

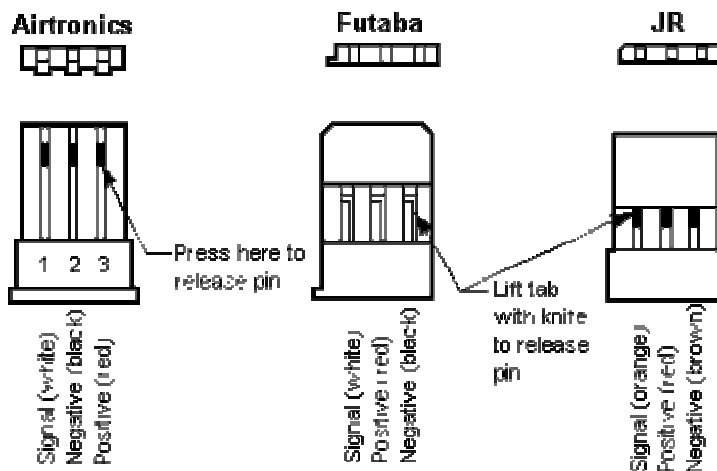
1.1 Servos

Quelle: <http://www.rn-wissen.de/index.php/Servos>

Handelsübliche Servomotoren besitzen 3 Anschlüsse:

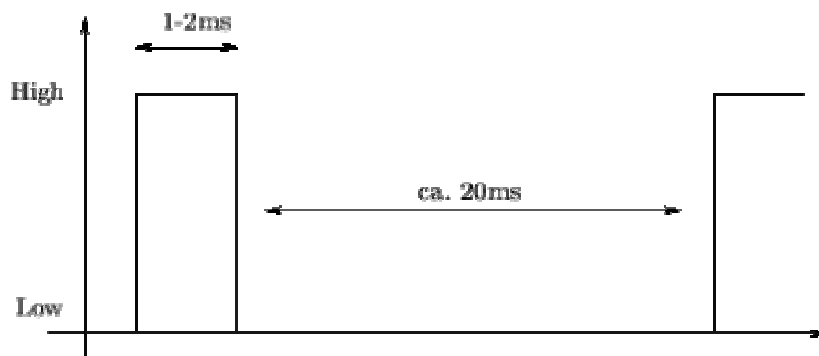
- GND
- PWM
- +5V

Stock Connectors



Ansteuerung

An die PWM-Leitung wird ein pulswidenmoduliertes Signal angeschlossen. Die Repetition-Period(Periode) entspricht bei den meisten Modellen 20ms. Innerhalb/zu Beginn dieser 20ms wird ein Puls erwartet, der sich zwischen 1ms und 2ms bewegt, wobei diese Werte jeweils den Endlagen des Servos entsprechen. D.h. 1ms ist ganz links und 2ms ist ganz rechts(Einige Sevos haben in diesem Wertebereich jedoch nicht die volle Bewegungsfreiheit ausgenutzt, die Werte, bei denen der Servo ganz links/rechts ist können auch unter 1ms/über 2ms liegen). 1,5ms würde demnach die Mittelstellung bedeuten. Aufgrund der Pulslänge lässt sich also eine direkte Aussage über die Position des Servos treffen. Der Motor sorgt dann intern mithilfe des Potis dafür, dass die Position gehalten wird.



1.2 Was kann SAnD-4

- Ansteuerung bis zu 4 Servos
- Ansteuerung bis zu 4 Relais
- es können 4 Kurven mit jeweils bis zu 100 Punkte frei definiert werden
- Es gibt bis zu 8 Digital-Adressen (DCC oder MM). Jeder Adresse kann eine „Aktion“ zugeordnet werden
- Für jede „Aktion“ kann der zu verwendene Servo, die Kurven-Nummer und der Relais-Ausgang festgelegt werden. Optional kann die Kurve auch in einer „Loop“ ständig wiederholt werden
- Auslösen einer Aktion auch durch 2 Taster (jeweils öffnen / schließen, also bis zu 4 Aktionen) möglich
- Konfigurierbar, ob nach Ablauf die Servo-Spannung und/oder die Servo-Impulse abgeschaltet werden sollen. Dadurch kann ein ständiges Nachjustieren (und damit Brummen) des Servos verhindert werden
- Einstellen der Servo-Endpositionen durch Justage-Routine möglich
- Digital-Spannung wird nicht durch Servo-Strom belastet (galvanische Trennung)

Die Konfiguration wird über den HEX-Manipulator durchgeführt, Download unter:

http://www.digital-bahn.de/bau_pic/hexmanipu.htm

2 Verwendete Register

2.1 Reservierte Register f. Dekoder-Funktion

Die Register im HEX-Bereich \$20-\$3f sind für die allgemeine Dekoder-Funktion reserviert

HEX	Name	
22	R_ADR_1	
23	R_ADR_2	
24	R_ADR_3	
25	R_ADR_3a	
26	R_ADR_2a	
27	R_ADR_1a	
28	R_DEKADR_1	
29	R_DEKADR_2	
2A	R_DEKADR_3	
2B	R_DEKADR_4	
2C	R_DEKADR_5	
2D	R_DEKADR_1_L	
2E	R_DEKADR_2_L	
2F	R_DEKADR_3_L	
30	R_BITCOUNT	
31	R_FLAG	
32	R_POL	
33	R_CNT1_TGL	
34	R_CNT2_TGL	
35	R_CNT3_TGL	
36	R_CNT_TGL	
37	R_CNT_ADR	
38	R_TGLA	
39	R_BUTTON	
3A	R_EEDAT	
3B	R_EEADR	
3C	R_MODE	
3D	R_STATUS	
3E	R_SAVE_F	
3F	R_TGLC	

2.2 Reservierte Register f. Servo-Dekoder

Die Register im HEX-Bereich \$40-\$7f sind für die speziellen Dekoder-Funktion für „SERVO“ reserviert

HEX	Name	Beschreibung
40	R_COUNT_01	Counter, Servo-Spezifisch Dieser Counter zeigt auf die Tabellen-Zeile des aktuellen Kurven-Wertes
41	R_COUNT_02	
42	R_COUNT_03	
43	R_COUNT_04	
44	R_KURVE_NUMBER_01	Reload f. Anzahl der Kurven-Punkte, Servo-Spezifisch In diese Register wird die Anzahl der Kurven-Punkte der zu fahrenden Kurve geladen
45	R_KURVE_NUMBER_02	
46	R_KURVE_NUMBER_03	
47	R_KURVE_NUMBER_04	
48	R_COUNT_STRETCH_01	Counter, Servo-Spezifisch Dieser Counter zeigt auf die zählt während einer Stretchung
49	R_COUNT_STRETCH_02	
4A	R_COUNT_STRETCH_03	
4B	R_COUNT_STRETCH_04	
4C	R_KURVE_STRETCH_01	Reload f. Stretch-Wert, Servo-Spezifisch In diese Register wird die Anzahl Wiederholungen je Kurven-Punkt der zu fahrenden Kurve geladen
4D	R_KURVE_STRETCH_02	
4E	R_KURVE_STRETCH_03	
4F	R_KURVE_STRETCH_04	
50	R_SERVO_ACTIVE	beinhaltet diese Flags: SERVO1_ACTIVE R_SERVO_ACTIVE, 0 SERVO1_LOOP R_SERVO_ACTIVE, 4 SERVO2_ACTIVE R_SERVO_ACTIVE, 1 SERVO2_LOOP R_SERVO_ACTIVE, 5 SERVO3_ACTIVE R_SERVO_ACTIVE, 2 SERVO3_LOOP R_SERVO_ACTIVE, 6 SERVO4_ACTIVE R_SERVO_ACTIVE, 3 SERVO4_LOOP R_SERVO_ACTIVE, 7
51	R_ADR_FLAGS_SERVO	USE_SERVO1 R_ADR_FLAGS_SERVO, 0 USE_SERVO2 R_ADR_FLAGS_SERVO, 1 USE_SERVO3 R_ADR_FLAGS_SERVO, 2 USE_SERVO4 R_ADR_FLAGS_SERVO, 3 USE_SERVOx_BREAK R_ADR_FLAGS_SERVO, 5 USE_SERVOx_LOOP R_ADR_FLAGS_SERVO, 4 USE_SERVOx_STOP R_ADR_FLAGS_SERVO, 6
52	R_ADR_FLAGS_MAP	USE_CURVE1 R_ADR_FLAGS_MAP, 4 USE_CURVE2 R_ADR_FLAGS_MAP, 5 USE_CURVE3 R_ADR_FLAGS_MAP, 6 USE_CURVE4 R_ADR_FLAGS_MAP, 7 USE_RELAIS1 R_ADR_FLAGS_MAP, 0 USE_RELAIS2 R_ADR_FLAGS_MAP, 1 USE_RELAIS3 R_ADR_FLAGS_MAP, 2 USE_RELAIS4 R_ADR_FLAGS_MAP, 3
54	R_4094_VALUES	R1_ON R_4094_VALUES, 0 R2_ON R_4094_VALUES, 1 R3_ON R_4094_VALUES, 2 R4_ON R_4094_VALUES, 3 V1_ON R_4094_VALUES, 6 V2_ON R_4094_VALUES, 7 V3_ON R_4094_VALUES, 5 V4_ON R_4094_VALUES, 4
55	R_TABLEREAD	Transfer-Register für Tabellen-Zugriffe über die Routine „TABLE_READ1“, „TABLE_READ2“, „TABLE_READ3“, „TABLE_READ4“ Beinhaltet die Zeilen-Nummer, die gelesen werden soll
56	R_PWMVALUE	Transfer-Register für die Routine „generate_servo_impuls“

57	R_SERVO_MAP_01	Servo-Spezifische Flags, beinhaltet jeweils: SERVO_0x_USE_C1 R_SERVO_MAP_0x, 4 SERVO_0x_USE_C2 R_SERVO_MAP_0x, 5 SERVO_0x_USE_C3 R_SERVO_MAP_0x, 6 SERVO_0x_USE_C4 R_SERVO_MAP_0x, 7 SERVO_0x_USE_R1 R_SERVO_MAP_0x, 0 SERVO_0x_USE_R2 R_SERVO_MAP_0x, 1 SERVO_0x_USE_R3 R_SERVO_MAP_0x, 2 SERVO_0x_USE_R4 R_SERVO_MAP_0x, 3	
58	R_SERVO_MAP_02		
59	R_SERVO_MAP_03		
5A	R_SERVO_MAP_04		
5B	R_COUNT_TEMP	not used	
5C	R_SERVO_FLAGS_01	Servo-Spezifische Flags, beinhaltet jeweils: S0x_SERVO_FLAG_I_OFF R_SERVO_FLAGS_0x, 2 S0x_SERVO_FLAG_NOINT R_SERVO_FLAGS_0x, 6 S0x_SERVO_FLAG_ONLY1 R_SERVO_FLAGS_0x, 7 S0x_SERVO_FLAG_V_OFF R_SERVO_FLAGS_0x, 1 S0x_SERVO_FLAG_WAIT R_SERVO_FLAGS_0x, 0	
5D	R_SERVO_FLAGS_02		
5E	R_SERVO_FLAGS_03		
5F	R_SERVO_FLAGS_04		
60	R_OFFSET_SERVO_01	diese Register beinhalten Offset und Faktor je Servo. Geladen aus den EEprom-Adressen: R_OFFSET_SERVO_01 \$60 R_OFFSET_SERVO_02 \$62 R_OFFSET_SERVO_03 \$64 R_OFFSET_SERVO_04 \$66 R_FAKTOR_SERVO_01 \$61 R_FAKTOR_SERVO_02 \$63 R_FAKTOR_SERVO_03 \$65 R_FAKTOR_SERVO_04 \$67	
61	R_FAKTOR_SERVO_01		
62	R_OFFSET_SERVO_02		
63	R_FAKTOR_SERVO_02		
64	R_OFFSET_SERVO_03		
65	R_FAKTOR_SERVO_03		
66	R_OFFSET_SERVO_04		
67	R_FAKTOR_SERVO_04		
68	R_ACTIVATOR		Übergabe mit der Nummer des zu verändernden Servos von den Routinen „gruen_x“ und „status_to_output“ an „restore_servo“
69	R_LEDSIGNAL		verwendet für die LED Signalisierung
70	R_FLAG_01	F_LEDON R_FLAG_01, 6 F_ONLY1 R_FLAG_01, 5 F_RESTORE R_FLAG_01, 7 F_V_was_off R_FLAG_01, 0	
71	R_LOOPCOUNT	In der Routine „change_mode“ und in der „statemachine_sand_4“ verwendet. Beinhaltet, welcher State gerade aktiv ist.	
72	R_SWITCH01	Verwendte in der Routine „check_switches“, realisiert dort das Entprellen der Switches	
73	R_SWITCH02		
74	R_FLAG_SWITCH	F_SW1_SW2_close R_FLAG_SWITCH, 4 F_SW1_close R_FLAG_SWITCH, 0 F_SW1_open R_FLAG_SWITCH, 1 F_SW2_close R_FLAG_SWITCH, 2 F_SW2_open R_FLAG_SWITCH, 3	
75	R_FAKTOR	Transfer-Register für die Routine „generate_servo_impuls“	
76	R_OFFSET		

3 Beschreibung der Software-Routinen

3.1 Routine „restore_servo“

Aufgabe	Status eines Servos verändern
Aufgerufen	Diese Routine wird aufgerufen, wenn der Status eines Servos zu verändern ist. Dies kann passieren durch <ol style="list-style-type: none">1. Digital-Befehl empfangen2. entsprechende Tasten-Betätigung3. während der Initialisierung („Restore“ der letzten Servo-Position)
Beschreibung	In dieser Routine werden die Flags und Register des zu aktivierenden Servos entsprechend der Parametrisierung (Werte im Tabellen-Bereich des HEX-Files) geladen.

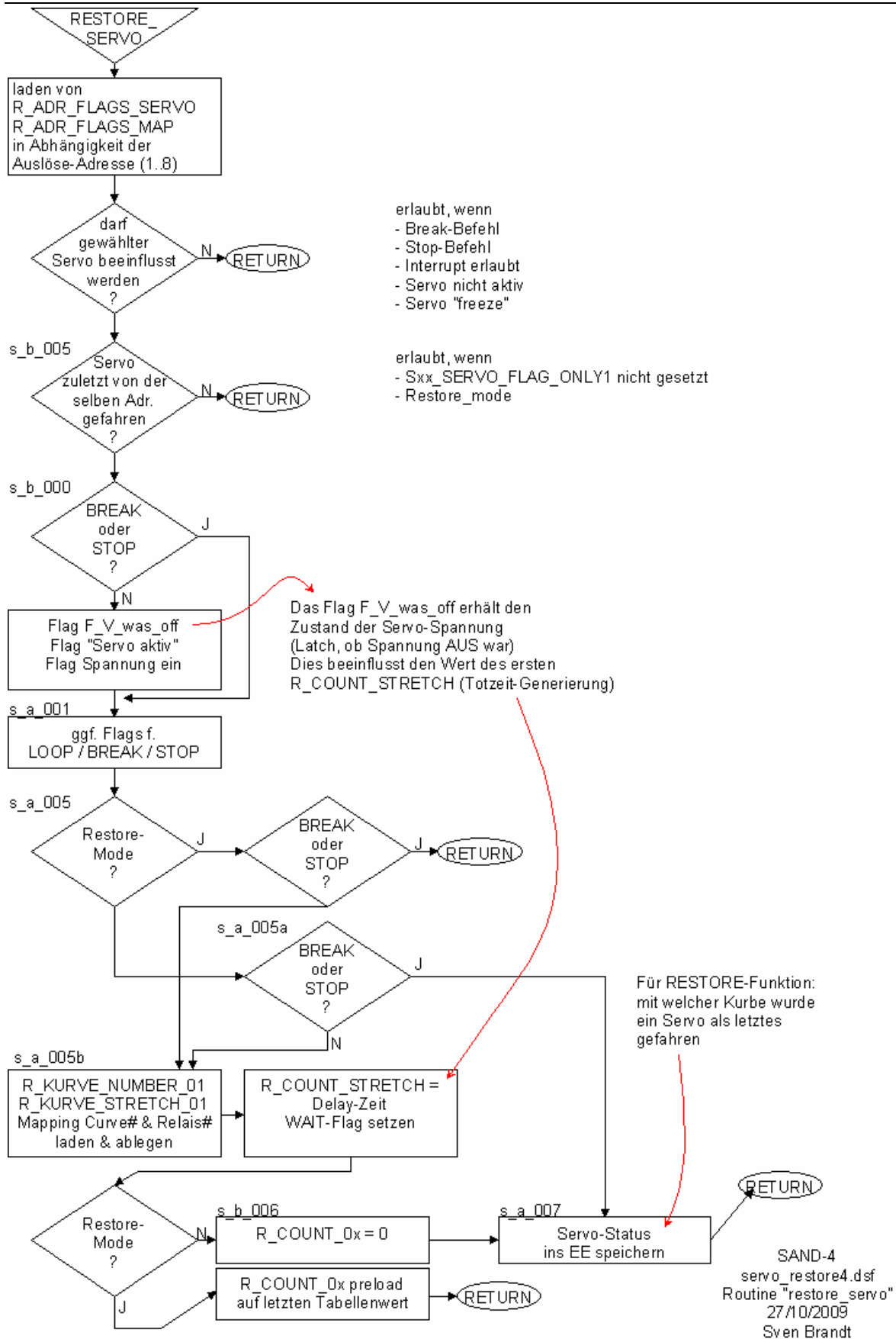


Abbildung 1: Flow-Chart „restore_servo“

3.2 Routine „update_4094“

Aufgabe	Ansteuerung des 4094 (serielles Schieberegister)
Aufgerufen	change_mode state_servo_005
Beschreibung	Die Ausgänge des 4094 werden entsprechend Register R_4094_VALUES gesetzt. Der Status der LED (da selber Ausgang wie der DATA_4094) wird gelatched und restored.

3.3 Routine „UMUL0707L“

Aufgabe	Multiplikations-Routine 7bit x 7 bit (Quelle: Microchip):
Aufgerufen	aufgerufen von „generate_servo_impuls“
Beschreibung	$AARGB0 \times BARGB0 = AARGB0[MSB] AARGB1[LSB]$

3.4 Routine „generate_servo_impuls“

Aufgabe Berechnen der capture compare (CCP) Register
Aufgerufen change_mode

- state_servo_001
- state_servo_002
- state_servo_003
- state_servo_004

Beschreibung input register
 - R_PWMVALUE
 - R_FAKTOR
 - R_OFFSET

Details zu der Berechnung der Impuls-Breite siehe Kap. 4.2

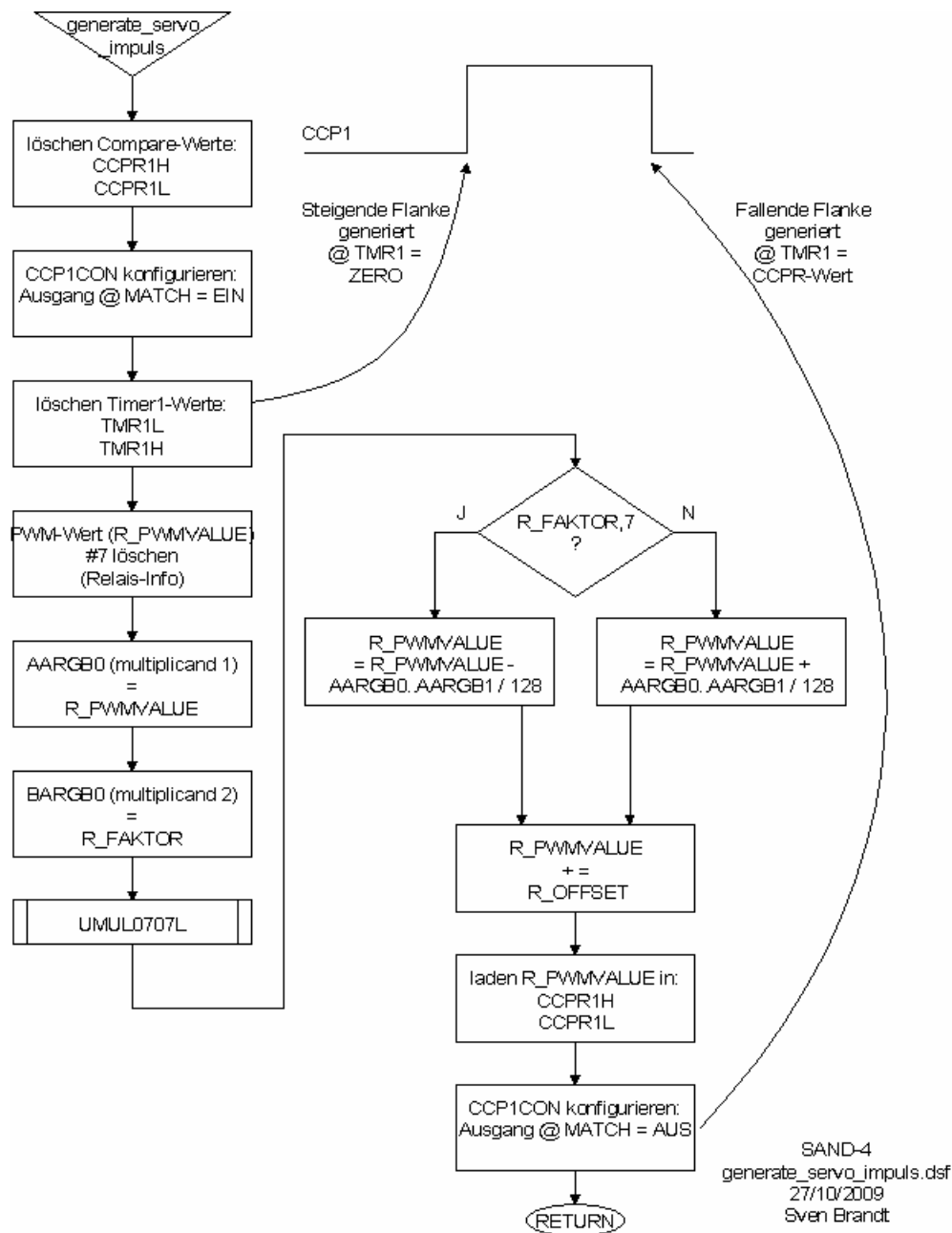


Abbildung 2: Flow-Chart „generate_servo_impuls“

3.5 Routine „change_mode“

Aufgabe In dieser Routine können die Anschläge des Servos justiert werden
Aufgerufen Routine wird aufgerufen, wenn der SW1 beim Starten aktiv (LOW) ist
Beschreibung

3.6 Routine „check_switches“

Aufgabe Einlesen der Taster SW1 und SW2
Aufgerufen change_mode
 statemachine_sand_4
Beschreibung Wird ein CLOSE erkannt, wird das Flag F_SWx_close gesetzt (einmalig)
 Wird ein OPEN erkannt, wird das Flag F_SWx_open gesetzt (einmalig)

der Counter R_SWITCH0x läuft hier bei gedrücktem Taster nach Oben, dabei werden nur die geraden Zahlen verwendet. Oben (d.h. bei 120) angekommen wird dann die Taster-Aktion ausgelöst und der Counter auf 121 gesetzt. diesem Fall muss der Counter erst wieder runter (auf < 21), dort wird der Wert dann wieder auf die gerade Zahl '20' gesetzt. Dadurch wird erreicht, dass nach der Aktion für Tastendruck die Taster erst losgelassen werden muss, ehe eine erneute Aktion durch Counter = 120 ausgelöst werden kann.

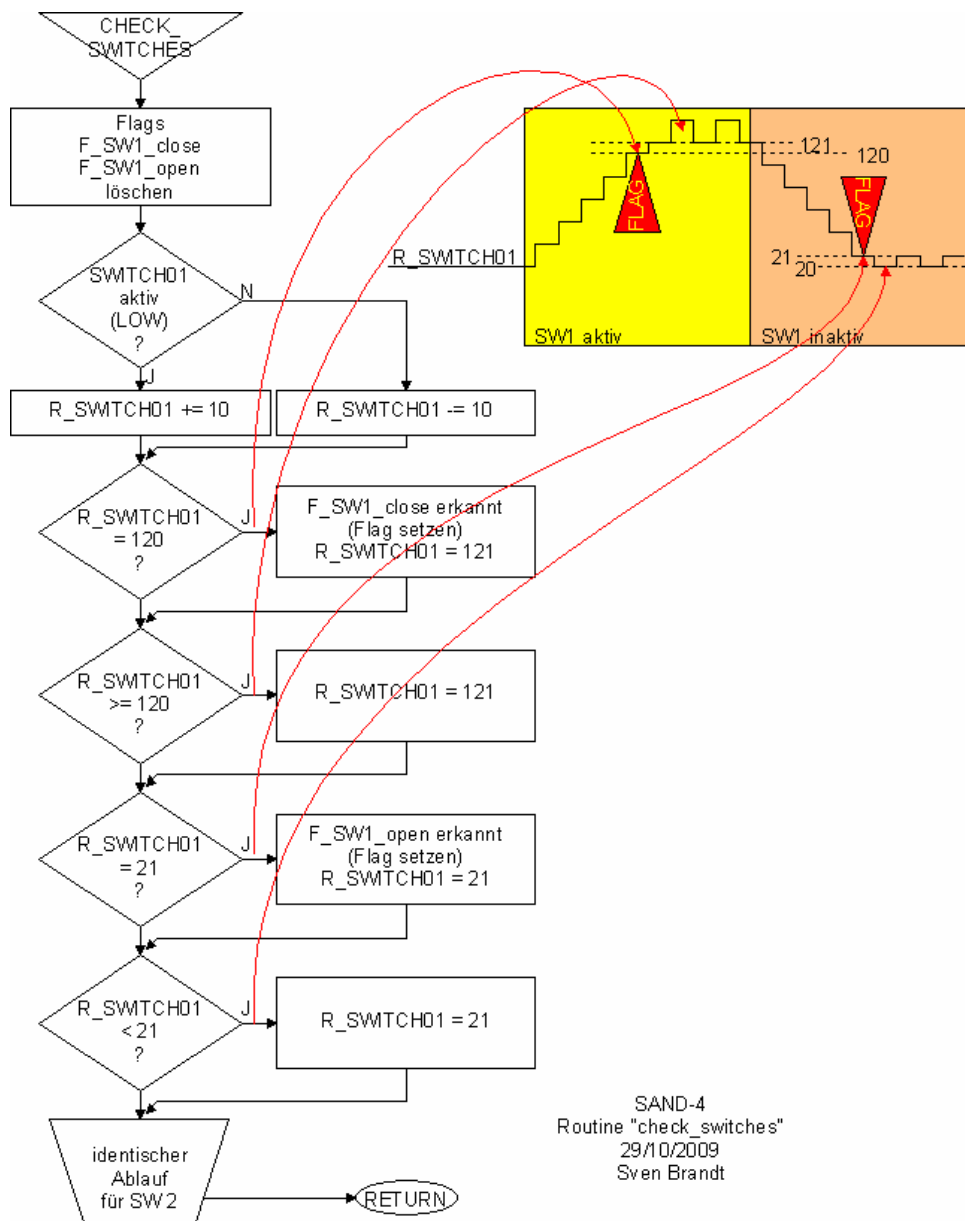


Abbildung 3: Flow-Chart „check_switches“

3.7 Routine „statemachine_sand_4“ (State Machine)

Aufgabe Servo Ansteuerung, Taster-Abfrage, LED-Ansteuerung, 4094-Ansteuerung
Aufgerufen zyklisch aus loop, Ablauf in 4ms Raster (Verwendung von Timer 1)
Beschreibung

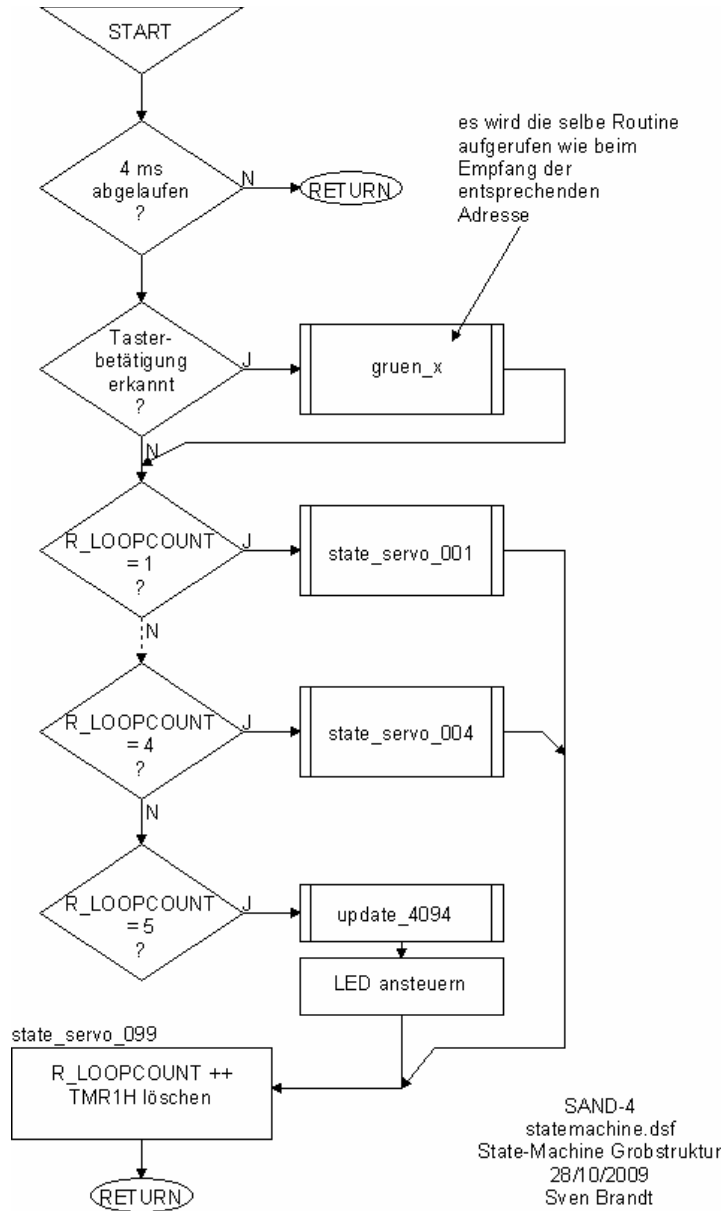


Abbildung 4: State-Machine Grobstruktur

4 Funktionsbeschreibungen

4.1 Parametrisierung

Grundsätzlich können die Parameter in ihre Funktionen unterteilt werden:

Kurven-Definition	es können 4 Kurven mit je bis zu 100 Werte definiert werden.
Servo Parameter	enthält Flags, die an den Servo (Nummer 1..4) gebunden sind. Diese Werte werden beim Start des Programms aus der Tabelle gelesen.
Curve Parameter	beinhalten Informationen über die Kurve (Anzahl der Werte und Stretch-Faktor). Diese Informationen werden abgerufen, sobald eine Adresse (bzw. Taster-Betätigung) erfolgt ist und Zuordnung der Adresse zur Kurve (Mapping) erfolgt ist (Routine „restore_servo“).
Adress Parameter	beinhaltet das Mapping, also die Zuordnung von Servo / Relais und Kurve zu der entsprechenden Adresse. Diese Informationen werden abgerufen, sobald eine Adresse (bzw. Taster-Betätigung) erfolgt ist (Routine „restore_servo“).
Switch Parameter	beinhaltet das Mapping der 2 Taster, also welche Aktion (wie beim Adress-Empfang) auszuführen ist. Diese Informationen werden abgerufen, sobald eine Taster-Betätigung erkannt wurde.

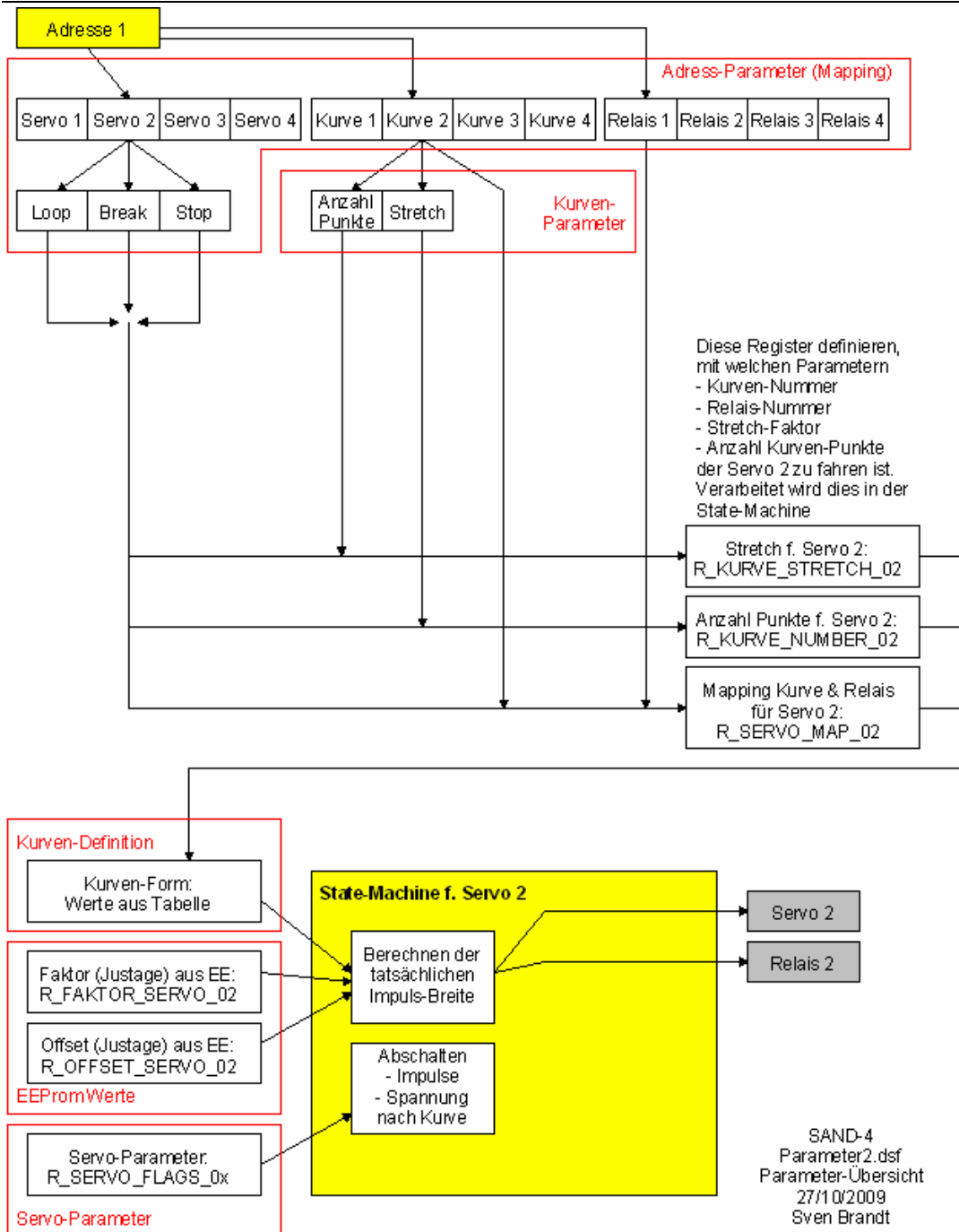


Abbildung 6: Übersicht Servo-Parametrisierung (Data-Flow)

4.2 Servo Impuls-Generierung

Für die Impuls-Generierung wird das CCP-Feature des PIC verwendet. Der CCP-Ausgang kann dadurch unabhängig von der Programm-Laufzeit eine gezielte Impuls-Breite ausgeben. Verwendet wird für dieses Verfahren der Timer 1.

In der Routine „generate_servo_impuls“ wird das CCP-Modul entsprechend konfiguriert. Die steigende Flanke wird sofort generiert (TMR1 = ZERO). Dann werden die Compare -Register CCPR1L und CCPR1H mit dem Wert entsprechend der Impuls-Breite geladen. Bei einem Match von Timer 1 mit den Compare-Registern erzeugt der PIC dann die fallende Flanke.

Der Timer 1 ist auf eine Step-Size von 8 µs konfiguriert.

Folgende Berechnungen sind nötig, um aus den Tabellenwert (Bereich 0..127) sowie dem Offset- und Faktor-Werten (Justage) auf die Compare-Werte zu kommen::

Beispiel 1:

Offset = 130
Faktor = 70
Tabellenwert = 60
Produkt = Faktor x Tabellenwert = 4200
Quotient = Produkt / 128 = 32
Compare-Wert = Quotient + Offset + Tabellenwert = 32 + 130 + 60 = 222
→ CCPR1H = 0
→ CCPR1L = 222

Beispiel 2:

Offset = 130
Faktor = 70
Tabellenwert = 127
Produkt = Faktor x Tabellenwert = 8890
Quotient = Produkt / 128 = 69
Compare-Wert = Quotient + Offset + Tabellenwert = 69 + 130 + 127 = 326
→ CCPR1H = 1
→ CCPR1L = 70

Beispiel 3: ist der Faktor >127, so ändert sich die Berechnung:

Offset = 130
Faktor = 130
Tabellenwert = 60
Produkt = (255-Faktor) x Tabellenwert = 125 x 60 = 7500
Quotient = Produkt / 128 = 58
Compare-Wert = - Quotient + Offset + Tabellenwert = - 58 + 130 + 60 = 132
→ CCPR1H = 0
→ CCPR1L = 132

4.3 Totzeit-Generierung

Um einem Servo das „hochfahren“ zu ermöglichen, muss eine Totzeit zwischen einschalten der Servo-Spannung und des ersten Impulses generiert werden.

Jeder Servo hat hierfür ein Flag S01_SERVO_FLAG_WAIT. Dieses servo-Spezifische Flag wird gesetzt, sobald der Servo gestartet wird. In der State-Machine wird dadurch ein Stretch-Zyklus ohne Impuls-Generierung durchfahren, danach wird dieses Flag gelöscht und die „echte“ Kurve gefahren.

Der Stretch-Counter kann beim Servo-Start wird mit unterschiedlichen Werte geladen werden. Dies hängt davon ab, ob die Versorgungs-Spannung des Servos abgeschaltet war (Laden mit = *delay_v_i*) oder nicht (Laden mit = *delay_v_i_short*)

Dadurch ergibt sich eine Tot-Zeit zwischen Einschalten der Versorgungs-Spannung der Servos und des ersten "echten" Impulses von (*'delay_v_i'* x 20 ms)

5 HEX Speicherbelegung

In dem HEX-File sind verschiedene Speicher-Bereiche reserviert. In diesen kann via HEX Manipulator parametrisiert werden.

	Zeile		
\$500..\$563	0..99	Kurve 1	100 Werte zur Definition der Kurve 1 #7 definiert, ob das Relais eingeschaltet wird
\$564	100	Servo Parameter (Servo 1)	#1 switch V off after RUN #2 switch Impulse off after RUN #6 reject interrupt of running servo #7 reject repetition of same adr.
\$565	101	Curve Parameter (Curve 1)	Number of Steps
\$566	102		Stretch-Faktor
\$567	103	Adress Parameter (Adress 1)	Flags 1 (Mapping Servo) #0 Adress 1 use Servo 1 #1 Adress 1 use Servo 2 #2 Adress 1 use Servo 3 #3 Adress 1 use Servo 4 #4 Loop #5 abbruch (sofort, Kurve wird abgebrochen) #6 beenden (Stopp nach Ablauf der Kurve)
\$568	104		Flags 2 (Mapping Relais / Kurve) #0 Adress 1 use Relais 1 #1 Adress 1 use Relais 2 #2 Adress 1 use Relais 3 #3 Adress 1 use Relais 4 #4 Adress 1 use Kurve 1 #5 Adress 1 use Kurve 2 #6 Adress 1 use Kurve 3 #7 Adress 1 use Kurve 4
\$569	105	not used	
\$56a	106	Adress Parameter (Adress 5)	Flags 1 (Mapping Servo) #0 Adress 5 use Servo 1 #1 Adress 5 use Servo 2 #2 Adress 5 use Servo 3 #3 Adress 5 use Servo 4 #4 Loop #5 abbruch (sofort, Kurve wird abgebrochen) #6 beenden (Stopp nach Ablauf der Kurve)
\$56b	107		Flags 2 (Mapping Relais / Kurve) #0 Adress 5 use Relais 1 #1 Adress 5 use Relais 2 #2 Adress 5 use Relais 3 #3 Adress 5 use Relais 4 #4 Adress 5 use Kurve 1 #5 Adress 5 use Kurve 2 #6 Adress 5 use Kurve 3 #7 Adress 5 use Kurve 4
\$56c	108	not used	
\$56d	109	not used	
\$56e	110	Switch Parameter	Info switch 1 close #0 like Adress 1 #1 like Adress 2 #2 like Adress 3 #3 like Adress 4 #4 like Adress 5 #5 like Adress 6 #6 like Adress 7 #7 like Adress 8
\$56f..\$57f	111..127	not used	

entsprechendes gilt für den Speicherbereich \$580..\$5ff / \$600..\$67f / \$680..\$6ff:

Kurven-Definition	<p>es können 4 Kurven mit je bis zu 100 Werte definiert werden. Diese Werte liegen im HEX-File an den folgenden Positionen:</p> <p>Kurve 1: \$500..\$563 Kurve 2: \$580..\$5E3 Kurve 3: \$600..\$663 Kurve 4: \$680..\$6E3</p> <p>Diese Werte werden während des Kurven-Ablaufes aus der Tabelle gelesen</p>
Servo Parameter	<p>enthält Flags, die an den Servo (Nummer 1..4) gebunden sind.</p> <p>Servo 1: \$564 (Zeile 100) Servo 2: \$5E4 (Zeile 100) Servo 3: \$664 (Zeile 100) Servo 4: \$6E4 (Zeile 100)</p> <p>Diese Werte werden beim Start des Programms aus der Tabelle gelesen:</p>
Curve Parameter	<p>beinhalten Informationen über die Kurve (Anzahl der Werte und Stretch-Faktor):</p> <p>Kurve 1: \$565..\$566 (Zeile 101..102) Kurve 2: \$5E5..\$5E6 (Zeile 101..102) Kurve 3: \$665..\$666 (Zeile 101..102) Kurve 4: \$6E5..\$6E6 (Zeile 101..102)</p> <p>Diese Informationen werden abgerufen, sobald eine Adresse (bzw. Taster-Betätigung) erfolgt ist und Zuordnung der Adresse zur Kurve (Mapping) erfolgt ist (Routine „restore_servo“).</p>
Adress Parameter	<p>beinhaltet das Mapping, also die Zuordnung von Servo / Relais und Kurve zu der entsprechenden Adresse</p> <p>Adresse 1: \$567..\$568 (Zeile 103..104) Adresse 2: \$5E7..\$5E8 (Zeile 103..104) Adresse 3: \$667..\$668 (Zeile 103..104) Adresse 4: \$6E7..\$6E8 (Zeile 103..104) Adresse 5: \$56A..\$56B (Zeile 106..107) Adresse 6: \$5EA..\$5EB (Zeile 106..107) Adresse 7: \$66A..\$66B (Zeile 106..107) Adresse 8: \$6EA..\$6EB (Zeile 106..107)</p> <p>Diese Informationen werden abgerufen, sobald eine Adresse (bzw. Taster-Betätigung) erfolgt ist (Routine „restore_servo“).</p>
Switch Parameter	<p>beinhaltet das Mapping der 2 Taster, also welche Aktion (wie beim Adress-Empfang) auszuführen ist.</p> <p>SW1 close:: \$56E (Zeile 110) SW1 open:: \$5EE (Zeile 110) SW2 close:: \$66E (Zeile 110) SW2 open:: \$6EE (Zeile 110)</p> <p>Diese Informationen werden abgerufen, sobald eine Taster-Betätigung erkannt wurde.</p>

6 EEPROM Speicherbelegung

	0	1	2	3	4	5	6	7	8	9	0a	0b	0c	0d	0e	0f
0x	RM	R1	R2	R3	R4	R5	R6	R7	R8	R9	Ra	FF	FF	FF	FF	FF
1x	RS	R1	R2	R3	R4	R5	R6	R7	R8	R9	Ra	FF	FF	FF	FF	FF
2x	MS	R1	R2	R3	R4	R5	R6	R7	R8	R9	Ra	FF	FF	FF	FF	MN
3x	RS2	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
4x	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
5x	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
6x	o1	f1	o2	f2	o3	f3	o4	f4	FF	FF	FF	FF	FF	FF	FF	FF
7x	L1	L2	L3	L4	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	DF

FF: not used

R1..Ra: Adresses R1..R10

RS: \$ 10: R_STATUS (Status)

RM: \$ 00: R_MODE (#0 gesetzt → Funktion SW1 = Lernen)

RS2: \$ 30: R_STATUS_2 (Status #2, for led_schalten only)

MN: \$ 2F: R_MINOUT (lad_haus only)

DF: \$ 7F: Digital-Format (info only): aa=MOT / dd=DCC / 00=NONE

o1..o4 \$ 60/62/64/66: Offset value for servo 1..4

f1..f4 \$ 61/63/65/67: faktor value for servo 1..4

L1..L4 \$ 70/71/72/73: last activator for servo 1..4

Motorola:

```

0x01 0x03 0x03 0x03 0x03 0x02 0x02 0x02 0x02 0x00 0x00 0x00 0x00 0xff 0xff 0xff
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xc0 0xc0 0xc0 0xc0 0xff 0xff 0xff
0x01 0xc3 0xf3 0xcf 0xff 0xc3 0xf3 0xcf 0xff 0xc3 0xf3 0xcf 0xff 0xff 0xff 0xff
0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff

0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0x7d 0xfc 0x7d 0xfc 0x7d 0xfc 0x7d 0xfc 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0x00 0x00 0x00 0x00 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xaa

```

DCC:

```

0x01 0x78 0x7a 0x7c 0x7e 0x7b 0x79 0x7f 0x7d 0x7a 0x78 0x7e 0x7c 0xff 0xff 0xff
0x00 0xf2 0xf6 0xfa 0xfe 0xf2 0xf6 0xfa 0xfe 0xf2 0xf6 0xfa 0xfe 0xff 0xff 0xff
0x01 0x05 0x05 0x05 0x05 0x09 0x09 0x09 0x09 0x0d 0x0d 0x0d 0x0d 0xff 0xff 0xff
0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff

0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0x7d 0xfc 0x7d 0xfc 0x7d 0xfc 0x7d 0xfc 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
0x00 0x00 0x00 0x00 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xaa

```