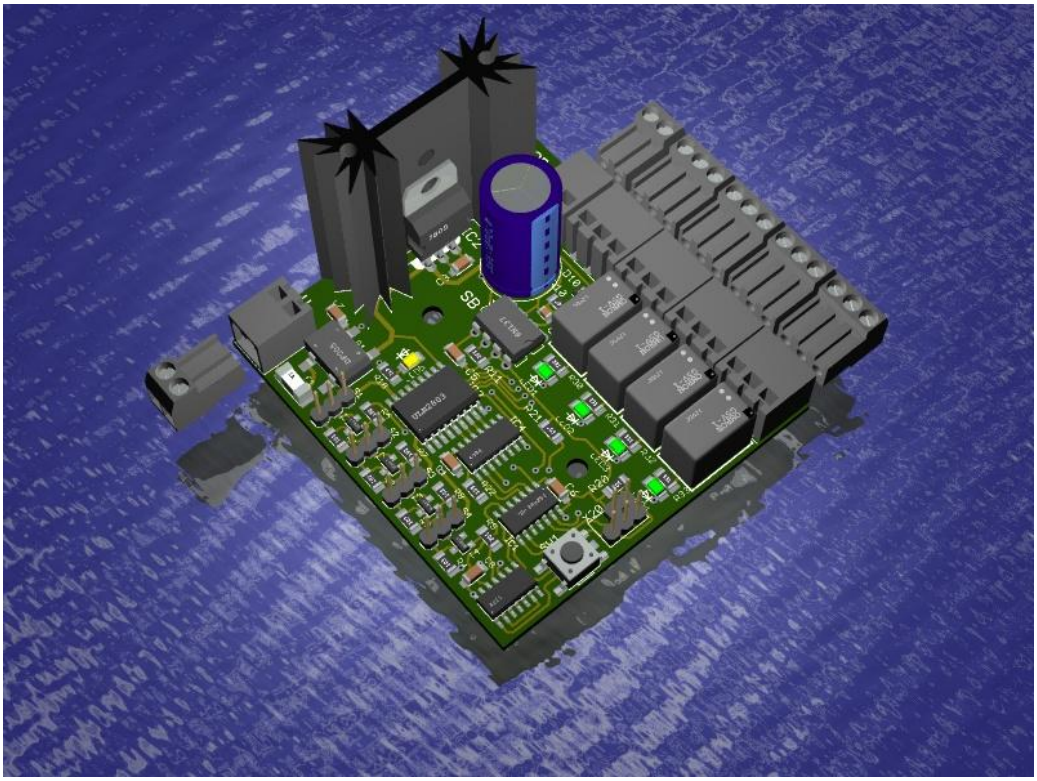
SAnD-4 - Servo Ansteuerungs-Dekoder (4 Servos)



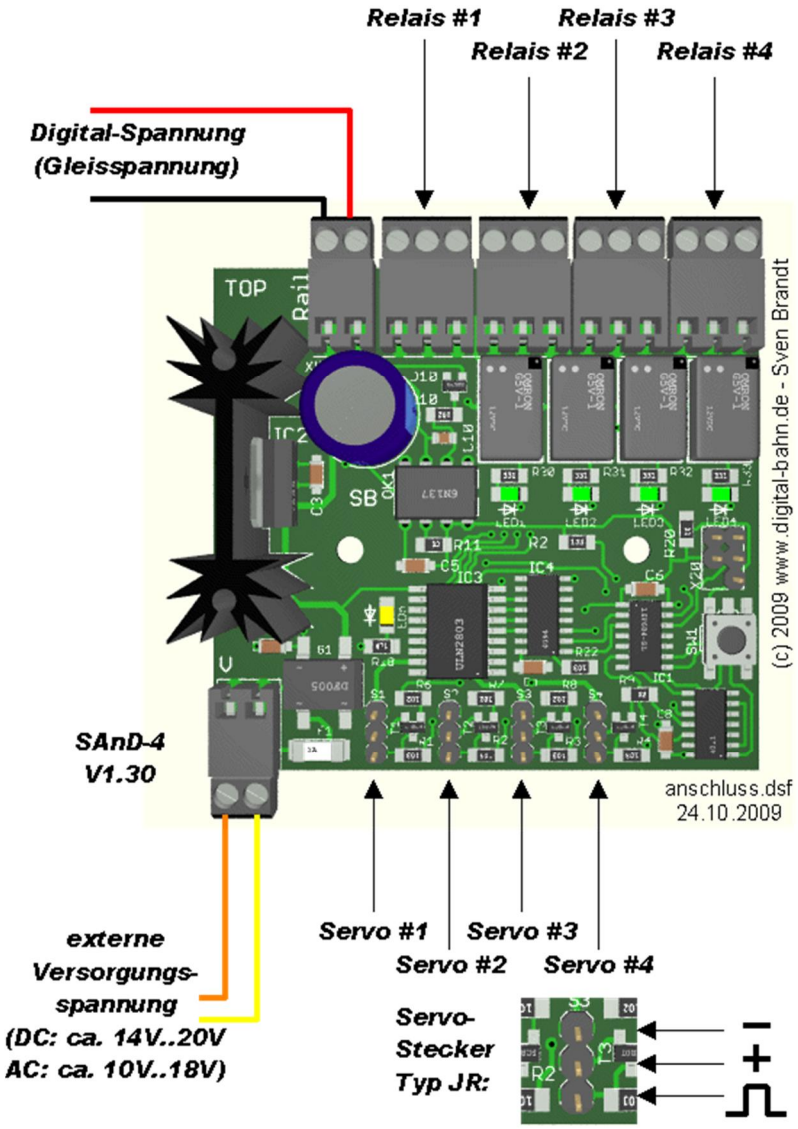
Hier die Eigenschaften des SAnD:

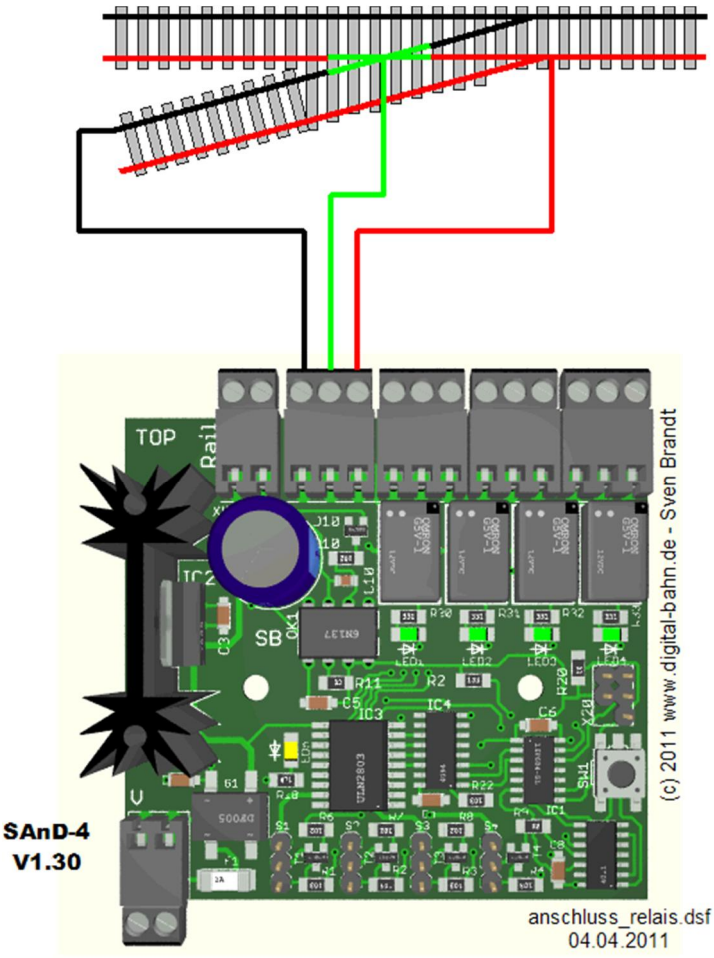
- Für Motorola (Märklin) oder DCC
- Digitale Ansteuerung von 4 Servos mit nahezu beliebigen Kurvenformen
- Über 8 Digital-Adressen können 4 unterschiedliche Kurven abgerufen werden. Dabei ist die Zuordnung zu den Servos frei konfigurierbar
- Kurven können einmalig oder wiederholt ("Loop") ablaufen
- Jeder Ablauf kann bis zu 100 Schritte haben. Dabei ist eine Schritt-Breite von 20 ms bis zu 4 s möglich, d.h. ein Ablauf kann bis zu 6.6 min dauern.
- erweiterter Einstellbereich für den Servo (gängig sind 1 ms .. 2 ms). Die minimale Impulsbreite kann 0.4 ms, die maximale Impulsbreite 4 ms betragen, wobei ein maximales Delta von ca. 2 ms möglich ist (also z.B. 0.4 ms .. 2.4 ms).
- Konfiguration der Kurvenform und Parameter via PC-Software
- Zusätzlich Ansteuerung von 4 Relais (Umschalter), um das Weichen-Herzstück der 2-Leiter Fahrer mit der richtigen Spannung zu versorgen
- 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.
- Anwendungsbeispiel 1: Umlegen von Weichen und Signalen inkl. "Umfassen" (das ist eine kleine Pause im Ablauf, in der ein Stellwerker den Hebel neu greifen muss)
- Anwendungsbeispiel 2: Ansteuerung von Schranken (inkl. "Nachwippen")
- Anwendungsbeispiel 3: Öffnen / Schließen von Toren etc.
- Anwendungsbeispiel 4: Ansteuerung von Spielplatz-Wippen etc. ("Loop"-Funktion)
- Einstellen der Servo-Endpositionen durch Justage-Routine möglich
- Digital-Spannung wird nicht durch Servo-Strom belastet (galvanische Trennung)

- Die Adressen können beliebig vergeben werden
- Address-Learning Funktion: Nach dem Druck auf die Taste gelangt der Dekoder in den "Lern-Modus".
- Keine DIP-Schalter notwendig, also auch keine Adress-Tabellen
- Polung der Eingangssignale beliebig, keine Verpolungsmöglichkeit
- Platinen-Grösse: 67 mm x 62 mm
- Vorbereitet für Montage auf DIN-Hutschienen (dadurch entfällt das Anschrauben unter der Anlage)
- Konfigurierbar ab V3.35: Ansteuern von bis zu 2 Servos auch durch Tasten möglich

Anschluss

Für den Anschluss an den Servo sind 3-Polige Stiftleisten im RM 2.54 ("JR-Stecker") vorgesehen. Die Spannungsversorgung kann über eine externe DC oder AC-Spannung vorgenommen werden, der Digital-Eingang hat eine galvanische Trennung über einen Optokoppler.



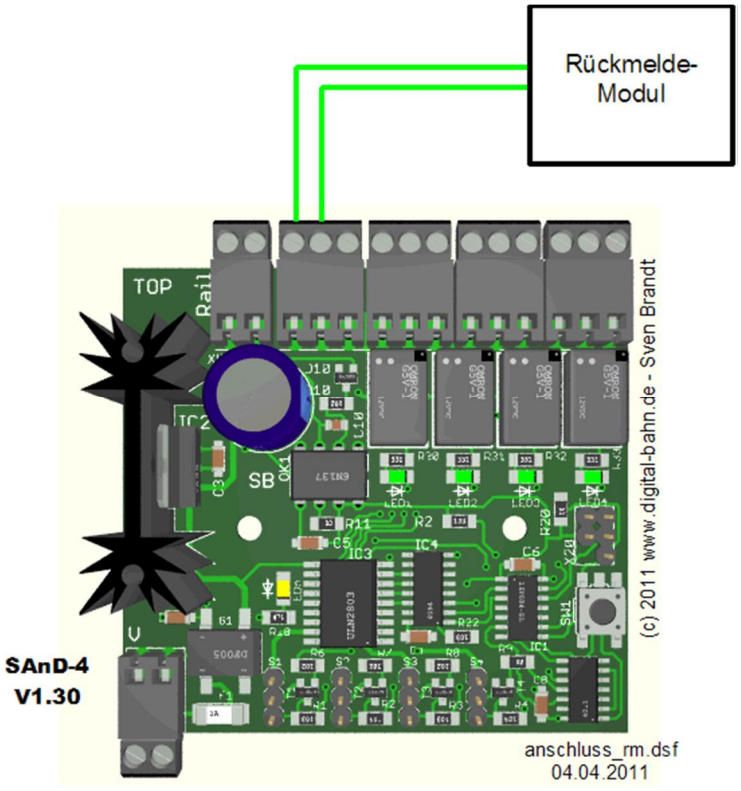


Anschluss-Beispiel Herzstück-Polarisierung

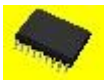
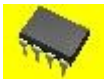
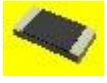
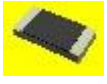
2-Leiter Fahrer haben das Problem, dass das Herzstück einer Weiche je nach Weichenstellung eine andere Polarität haben muss. Mit Hilfe der Relais kann das Herzstück sehr einfach mit dem richtigen Pol der Gleis-Spannung versorgt werden. Hier im Beispiel für Relais 1 aufgezeigt:

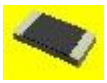
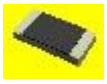
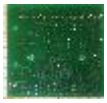
Anschluss-Beispiel Rückmeldung

Die Relais können für eine Rückmeldung der Weichenstellung verwendet werden. In dem Beispiel ist der Kontakt in der einen Weichenstellung offen, in der anderen geschlossen. Auch eine 2-Polige Rückmeldung ist möglich (Weichenstellung 1 = Kontakt 1 geschlossen, Weichenstellung 2 = Kontakt 2 geschlossen). Der Anschluss an den Rückmelder ist vom Rückmelder selber abhängig. Geeignet sind alle Rückmelder, die einen Kontakt offen/geschlossen erfassen können wie z.B. Masse-Sensoren. Strom-Sensoren eignen sich also weniger.



Diese Art der Rückmeldung ist ein Kompromiss zwischen Betriebssicherheit und einfacher Verdrahtung. Ein Versagen des Servos (z.B. Servo Defekt, Kabel ab) wird hier NICHT erkannt. Hier ist ein Ansetzen "weiter hinten" in der Wirkungskette nötig, z.B. den entsprechenden Kontakt durch ein Mikroschalter auslösen, der wiederum mechanisch durch den Servo ausgelöst wird.

Qty	Parts	Bezeichnung	Gehäuse	Bezug	Bestellnummer	Preis	Anmerkung
1	IC1	Prozessor 16F684-I/SL , SOIC-14		Reichelt	PIC 16F684-I/SL	ca. 1.20 Euro	
1	IC2	Spannungsregler 7805 , TO-220		Reichelt	µA 7805	ca. 0.17 Euro	
1	IC3	Treiber ULN2803AD , SOIC-18		Reichelt	ULN 2803 D	ca. 0.45 Euro	
1	IC4	Logic-IC 4094 , SOIC-16		Reichelt	SMD 4094	ca. 0.16 Euro	
1	IC6	Logic-IC 4011 , SOIC-14		Reichelt	SMD 4011	ca. 0.15 Euro	
1	OK1	Optokoppler 6N137 , DIP8		Reichelt	6N 137	ca. 0.49 Euro	
4	T1..T4	Transistor PNP BC807 , SOT23		Reichelt	BC 807-40 SMD	ca. 0.05 Euro	
1	D10	Doppel-Diode BAV99 , SOT23		Reichelt	BAV 99 SMD	ca. 0.04 Euro	
1	G1	Gleichrichter DF 005		Reichelt	SMD DF005	ca. 0.21 Euro	
5	LED1..LED5	LED , 1206		Reichelt	SMD-LED 1206 xx	ca. 0.11 Euro	xx = RT/GN/GE
4	R1..R4	Widerstand 10k , 1206		Reichelt	SMD 1/4W 10k	ca. 0.10 Euro	
3	R20..R22	Widerstand 10k , 1206		Reichelt	SMD 1/4W 10k	ca. 0.10 Euro	
6	R6..R11	Widerstand 1 k , 1206		Reichelt	SMD 1/4W 1,0k	ca. 0.10 Euro	

4	R40..R43	Widerstand 1 k , 1206		Reichelt	SMD 1/4W 1,0k	ca. 0.10 Euro	alternativ zu K1..K4, siehe Ergänzungs-Stückliste 1)
1	R18	Widerstand 470R , 1206		Reichelt	SMD 1/4W 470	ca. 0.10 Euro	
4	R30..R33	Widerstand 330R , 1206		Reichelt	SMD 1/4W 330	ca. 0.10 Euro	
1	C1	Elko 470uF, 35V, radial (stehend)		Reichelt	RAD 470/35	ca. 0.14 Euro	
6	C3..C8	Keramik-Kond. 100 nF , 1206		Reichelt	X7R-G1206 100N	ca. 0.10 Euro	
1	C10	Keramik-Kond. 47 pF , 0805		Reichelt	NPO-G0805 47p	ca. 0.05 Euro	
1	F1	SMD-Sicherung 1A Superflink		Reichelt	SMD-SF 1,0A	ca. 0.40 Euro	Info Sicherungen
1	SW1	Taster Schurter LSG 1301.9313, 6.2 x 6.8 mm		Reichelt	Taster 9313	ca. 0.28 Euro	
2	X5, X9	Stecksystem 2-polig, RM 3.5 mm		Reichelt	AKL 182-02 / AKL 169-02	ca. 0.35 + 0.46 Euro	
4	S1..S4	Stiftleiste 1x3-polig gerade, RM 2.54 mm		Reichelt	(SL 1x40G 2,54) = 40-polig	ca. 0.18 Euro	für Servo (JK), abschneiden
1	X20	Stiftleiste 2x3-polig, RM 2.54 mm		Reichelt	(SL 2x40G 2,54) = 2x40-polig	ca. 0.28 Euro	Programmierstecker
1	LP1	Platine, ca. 67 mm x 61.5 mm x 1.6 mm		Shop	pcb_sand4	Staffelpreis lt. Shop	
1	KK1	Kühlkörper PR32 38,1mm		Reichelt	V PR32/38,1	ca. 1.25 Euro	excl. Montage-Material (M3)
1	Z1	DIN-Schienen-Halter		Reichelt	BOPLA TSH35	ca. 2.30 Euro	optional für Hutschienen-Montage

Ergänzungs-Stückliste 1: Relais und Anschluss-Stecker

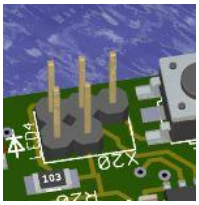
Die Relais K1..K4 sind nicht für jeden brauchbar. Wer jedoch 2-Leiter Schienen verwendet und mit dem Servo eine Weiche umschalten möchte, der kann Herzstück (und evtl. Weichenzunge) über das Relais umpolen. Auch Rückmeldungen via S88 (oder ähnliches) sind durch das Relais leicht zu realisieren.

Wer die Relais nicht braucht, kann diese (und auch X1..X4) weglassen. Dafür sollten dann aber R40..R43 bestückt werden, da man dann wenigstens die LEDs LED1..LED4 zur Anzeige verwenden kann. Im Justage-Modus werden diese LEDs dafür benötigt, den aktiven Servo zu signalisieren. Im Betrieb könnte man über die LEDs dann eine aktive Bewegung signalisieren (dies muss dann nur in der Kurven-Form entsprechend vorgegeben werden).

Qty	Parts	Bezeichnung	Gehäuse	Bezug	Bestellnummer	Preis	Anmerkung
4	K1..K4	Relais Omron G5V-1 12V		Reichelt	G5V-1 12V	ca. 1.35 Euro	Relais baugleich
		Relais Fujitsu-Takamisawa SY 12V		Reichelt	SY 12W K	ca. 1.30 Euro	
4	X1..X4	Stecksystem 3-polig, RM 3.5 mm		Reichelt	AKL 182-03 / AKL 169-03	ca. 0.35 + 0.66 Euro	
4	R40..R43	entfallen					

In Circuit Programmierung

Über den 5-poligen Stecker X20 kann die bestückte Dekoder Platine programmiert werden.



Adressen lernen (Address-Learning Funktion)

Durch den Druck auf den Programmier-Taster auf der Platine fangen die LEDs der Ausgänge an zu blinken. Zuerst blinkten diese im 1er Rhythmus, jetzt wird die Adresse für den Befehl#1 abgefragt. Als nächstes (2er Blinken) möchte der Dekoder die Adresse für den Befehl#2 wissen, es folgt die Abfrage für Befehl#3 bis Befehl#8. Jede Lernphase kann mit der Taste übersprungen werden, dann wird die bereits programmierte Adresse nicht verändert.

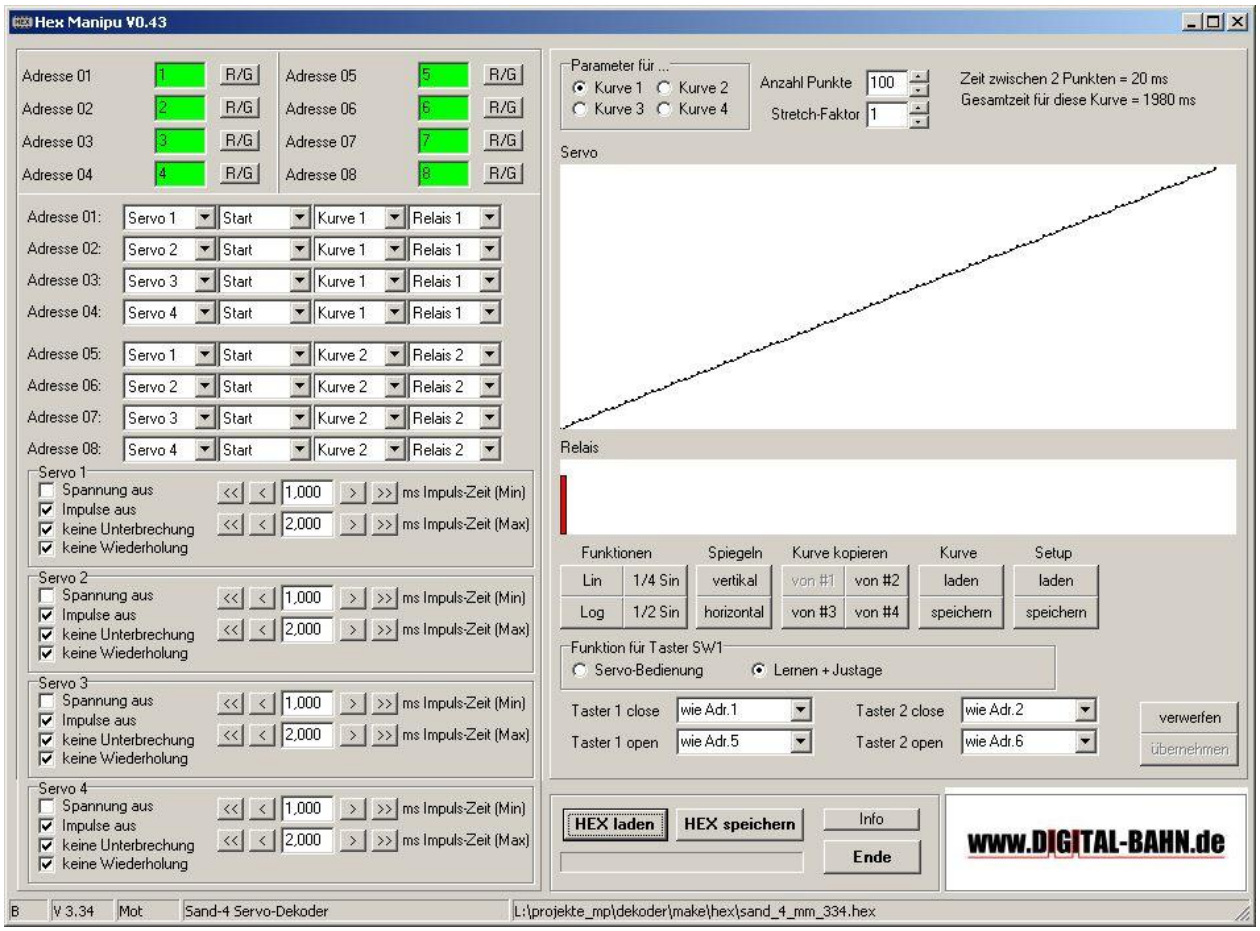
Solange die Kurven und die Zuordnung nicht über den Manipulator geändert wurden, ist die Abfolge der Adressen wie folgt definiert:

Adresse 1 = Servo 1 hin, Adresse 2 = Servo 2 hin, Adresse 3 = Servo 3 hin, Adresse 4 = Servo 4 hin
Adresse 5 = Servo 1 zurück, Adresse 6 = Servo 2 zurück, Adresse 7 = Servo 3 zurück, Adresse 8 = Servo 4 zurück

SAnD - Servo Ansteuerungs-Dekoder: Konfiguration

Ein Servo hat naturgemäss eine ganze Menge Einstell-Möglichkeiten. Für die verschiedenen Ansteuerungen möchte der professionelle Anwender (und wer Servos einsetzen will, den kann man wohl zu dieser Gruppe zählen..) z.B. die Kurvenform anpassen.
Das Einstellen einer Kurve über CVs oder ähnliches möchte ich mir lieber gar nicht erst vorstellen (müssen), dies ist einfach nicht praktikabel. Da bei meinen Projekten ja alles ein bischen anders ist, habe ich an dieser Stelle auch eine ganz spezielle Möglichkeit: die Manipulation des HEX-Files, welches dann in den Dekoder-PIC gebrannt wird. Das geht natürlich nur, wenn man das HEX-File dann auch in den PIC übertragen werden (sprich "brennen") kann, aber diese Fähigkeit haben wohl inzwischen die meisten Anwender meiner Projekte.

Zunächst muss das entsprechende HEX-File in den Manipulator geladen werden. Es erscheint jetzt dieser Blidschirm:



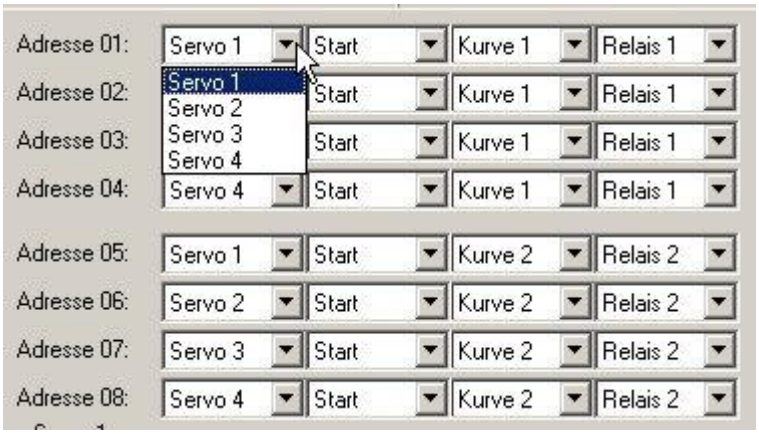
Adressen einstellen

Der Dekoder hat (bis zu) 8 Adressen. Die Adressen können hier eingestellt werden, aber auch später "am Gleis" über die normale Adress-Lern Routine geändert werden.

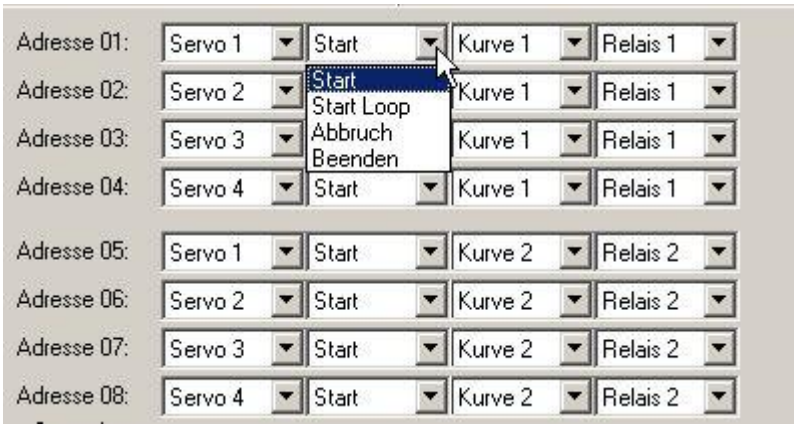
Mapping einstellen

Jetzt wird es schon ein bischen komplizierter: das sogenannte "Mapping" definiert, durch welche Adresse bei bestimmter Servo mit einer bestimmten Kurve gefahren wird.

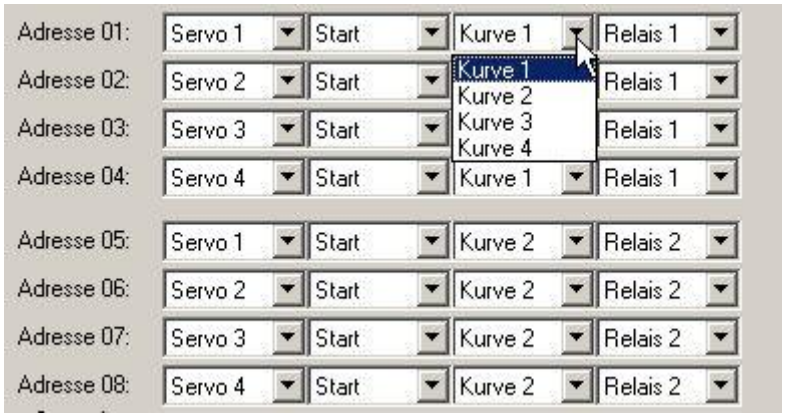
Es wird zunächst definiert, welcher Servo zu berwenden ist:



Anschließend kann bestimmt werden, was mit diesem Servo zu tun ist, also ob eine Kurve einmalig oder zyklisch (START LOOP) zu starten ist. Laufende Loops können mit der ABBRUCH-Funktion sofort gestoppt werden, während ein BEENDEN die Kurve noch beendet und dann die LOOP nicht wieder startet.



Jetzt wird noch definiert, welche der 4 Kurven denn nun verwenden ist:



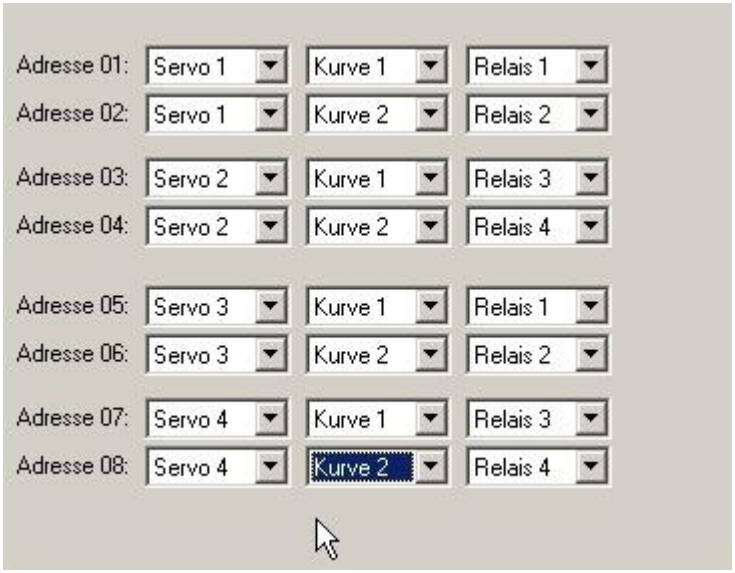
Und nun noch, welcher Relais-Ausgang geschaltet werden soll:

Ich erkläre das am besten mal an ein paar Beispielen: Mal angenommen, Sie wollen Form-Signale (2-begriffig) stellen, und zwar an allen 4 Servos. Dann brauchen Sie ja im Prinzip 2 Kurven-Formen, nämlich je eine für "Hoch" und für "Runter". Dadurch kann man dann in der "Hoch"-Kurve auch das "Umfassen" realisieren. Nun würde sich dann das folgende Mapping ergeben:

Adresse 1 fährt Signal 1 hoch, also Kurve 1 an Servo 1
Adresse 2 fährt Signal 1 runter, also Kurve 2 an Servo 1
Adresse 3 fährt Signal 2 hoch, also Kurve 1 an Servo 2
Adresse 4 fährt Signal 2 runter, also Kurve 2 an Servo 2
Adresse 5 fährt Signal 3 hoch, also Kurve 1 an Servo 3
Adresse 6 fährt Signal 3 runter, also Kurve 2 an Servo 3
Adresse 7 fährt Signal 4 hoch, also Kurve 1 an Servo 4
Adresse 8 fährt Signal 4 runter, also Kurve 2 an Servo 4

könnse folgen? Gut! Dieses Mapping würde dann hier so aussehen, wobei wir die Einstellungen für

die Relais ersteinmal ignorieren können:



Servo-Parameter einstellen

Es gibt noch ein paar Parameter, die Servo-Spezifisch sind:

"Impuls-Zeit (MIN)" und "Impuls-Zeit (MAX)" stellen die End-Anschläge des Servos da. Hier können Werte eingetragen werden, wenn man diese schon kennt. Ansonsten würde man diese Parameter eher durch die Justage-Funktion (Endpositionen einstellen) ermitteln.

"keine Unterbrechung" führt dazu, eine der Servo während einer Bewegung nicht durch einen neuen Befehl unterbrochen werden kann. Dies ist eigentlich meistens sinnvoll, da ansonsten ziemlich wilde Servo-Bewegungen entstehen können.

"keine Wiederholung" führt dazu, die selbe Kurve nicht 2x hintereinander gefahren werden darf. Dies ist z.B. bei Weichen sinnig, denn wenn diese zuletzt von LINKS nach RECHTS gefahren wurde, dann muss sie ja erst wieder von RECHTS nach LINKS gefahren werden. Eine Wiederholung der Bewegung sähe ziemlich blöd aus.

"Spannung aus" führt dazu, dass nach Ablauf der Kurve die Versorgungs-Spannung des Servos abgeschaltet wird.

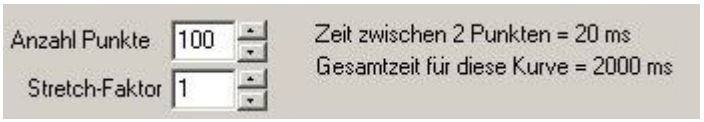
"Impulse aus" führt dazu, dass nach Ablauf der Kurve die Ansteuer-Impulse des Servos abgeschaltet werden.

Die letzten beiden Einstellungen haben folgenden Hintergrund: Wird ein Servo mit Impulsen versorgt, so versucht er, die Stellung zu halten. Dies kann dazu führen, dass der Servo ständig brummt, obwohl er eigentlich kar nichts zu tun hat. Also kann man die Ansteuer-Impulse abschalten. Dies wiederum mag nicht jeder Servo. Es gibt sog. "Fail-Save"-Servos, die gehen in einen sicheren Grund-Zustand, wenn die Ansteuerung weggenommen wird. Auch bei digitalen Servos kann das Abschalten der Impulse nicht helfen, weil diese sich die Soll-Stellung einfach merken und trotzdem nachregeln. Also hilft hier das Abschalten der Versorgungs-Spannung. Dies jedoch kann bei den Analogen Servos wiederum problematisch sein, weil diese beim Wieder-Einschalten der Spannung teilweise wilde Zuck-Bewegungen ausführen.
Fazit: Hier muss jeder je nach Anwendung und Servo die richtige Abschalt-Taktik rausfinden!

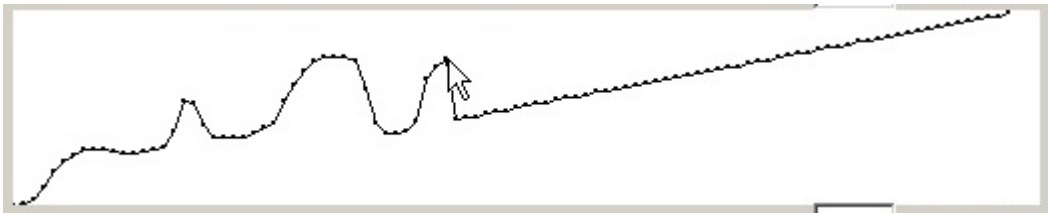
Kurvenform einstellen

Beim Mapping haben wir ja schon davon gesprochen das man bestimmte "Kurven fahren". Nun wollen wir mal zeigen, wie man so eine Kurve nun definiert. Zunächst einmal gibt es bis zu 4 Kurven, die man definieren kann. Die Auswahl, welche dieser Kurven gerade angezeigt und bearbeitet wird, erfolgt hier:

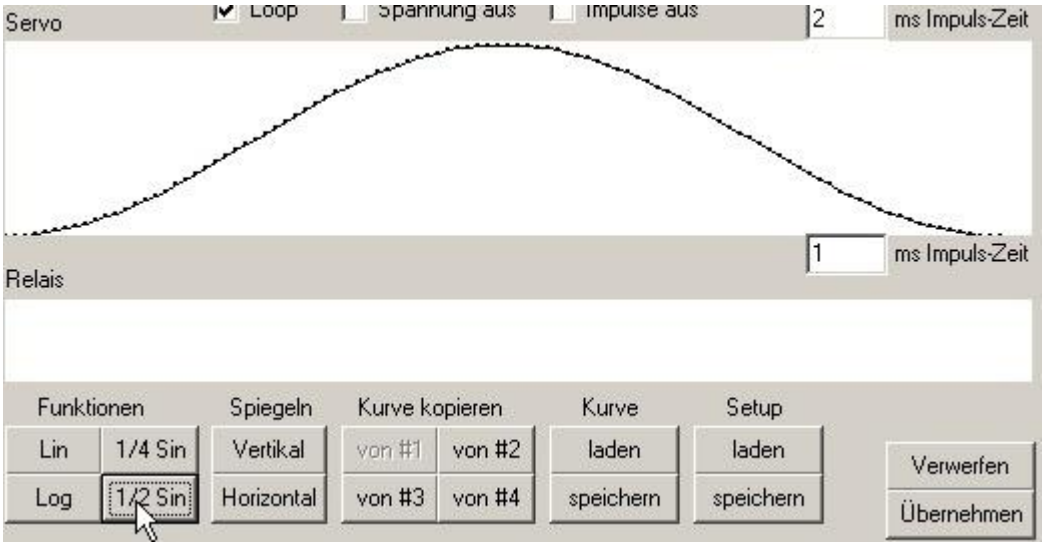
Dann gibt es noch 2 allgemeine Angaben je Kurve, nämlich die Anzahl der Punkte sowie ein "Stretch-Faktor", mit dem man den Ablauf einer Kurve langsamer machen kann. Diese werden hier eingestellt und die daraus resultierenden Zeiten angezeigt:



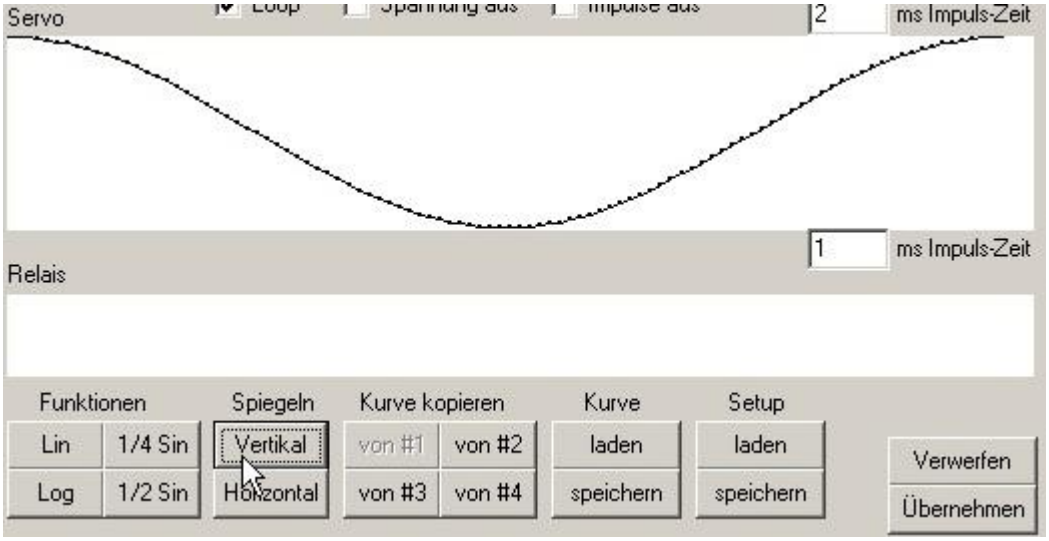
So, nun aber endlich zur eigentlichen Kurve! Die Kurve wird angezeigt und kann via Maus (Klick und Ziehen) angepasst werden. Ich denke dass mit dieser Darstellung jeder klarkommen sollte (wenn nicht: unten und oben in der Grafik entsprechen den beiden End-Positionen des Servos. Die Kurve wird von links nach rechts abgefahren, d.h. die einzelnen Punkte werden dem Servo nacheinander als Soll-Position übergeben).



Was gibt es sonst noch? Es können Kurve über einige mathematische Funktionen vorgegeben werden:

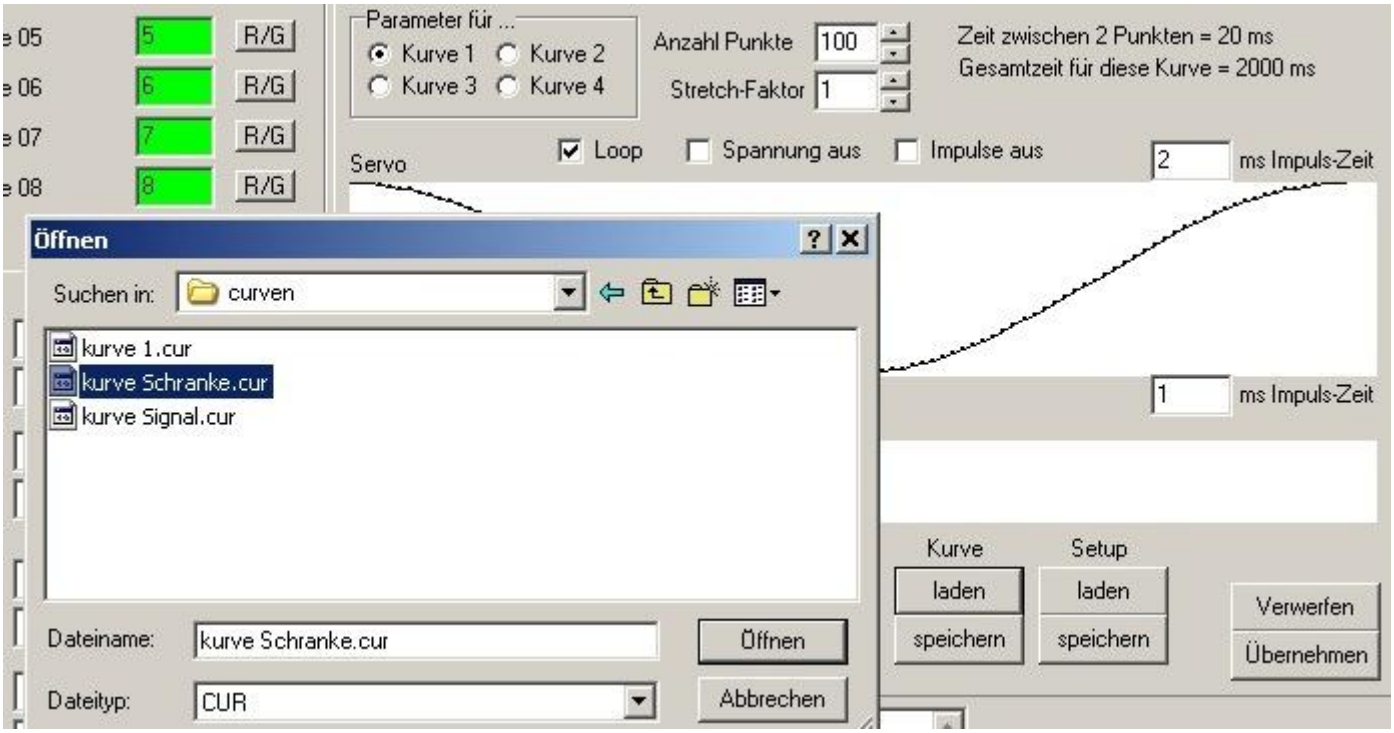


.. es gibt horizontale und vertikale Spiegel-Möglichleiten:



Eine Kurve kann in eine andere kopiert werden:

Eine Kurve und ein komplettes Setup kann geladen und gespeichert werden. Bei einer Kurve handelt es sich hier nur um die einzelne Kurve, bei einem Setup werden alle 4 Kurven und auch die dazugehörigen Parameter wie der Stretch-Faktor, das Mapping usw. geladen:



Adressen lernen (Address learning function)

Durch den Druck auf den Programmier-Taster können die Adressen auch über den Lern-Mode angelernt werden. Zunächst fängt die LED 5 (Grün) im 1er Rhythmus an zu blinken, wodurch signalisiert wird, dass Adresse 1 gerade gelernt werden kann. Wird jetzt ein Schalt-Befehl (Weichen-Befehl) auf das Gleis gelegt, so wird diese Adresse abgespeichert. Automatisch geht es gleich weiter mit Adresse 2 (signalisiert durch 2er-Blinken). Dito dann für Adresse 3 bis 8.
Sollen Adressen beim Lernen übersprungen werden, kann man dies durch erneutes Drücken der Programmier-Taste erreichen. Der Lern-Vorgang kann jederzeit durch Wegnehmen der Versorgungs-Spannung beendet werden, bereits gelernte Adressen sind dann bereits gespeichert. Ein Abklemmen der Digital-Spannung hat diese Wirkung hingegen nicht.

Justage-Modus: Endpositionen einstellen

Da die End-Positionen der Servos bei einigen Konstruktionen nach dem Einbau justiert werden muss, habe ich eine entsprechende Routine eingebaut. Hierfür muss am Programmier-Stecker eine Justage-Schaltung aus ein paar Tastern und Dioden angeschlossen werden, mit der die Justage durchgeführt wird.

Die Routine wird durch Drücken des Programmier-Tasters während des Startens (also beim Anlegen der Versorgungsspannung für die Platine) aufgerufen.

Hier der Ablauf, um die Endpositionen zu justieren:

1. während des Startens der Platine (=Anlegen der Versorgungs-Spannung) wird der Programmier-Taster oder der Taster SW3 auf der Justage-Platine gedrückt
2. Jetzt wird der Servo 1 an die End-Position Nr. 1 gefahren (dies ist die Position, die der Servo mit dem 1 ms-Impuls anfährt. Ob dies der rechte oder der Linke Anschlag ist, ist Servo-Abhängig). Die LED 1 des Relais 1 ist jetzt an, die LED 5 blinkt langsam
3. Jetzt kann man mit den Tasten SW1 und SW2 auf der Justage-Platine die Position verstellen, während Servo 1 entsprechend reagiert. Die Schrittweite beträgt hier 8 us.
4. Durch Druck auf SW3 der Justage-Platine kommt man in die Einstellung der 2. End-Position (Vorgabe 2 ms). LED 5 blinkt jetzt deutlich schneller
5. wie Punkt 3
6. Nach erneutem Tastendruck geht das ganzs Spiel wieder von vorne los, aber diesmal für Servo 2, 3 und 4
7. nach der Justage aller Servos (LED5 blinkt nicht mehr, LED1-4 idR. aus) befindet sich die Platine wieder im normalen Betriebs-Modus.

Zu beachten ist hierbei, dass ein Verstellen der End-Position 1 auch die End-Position 2 entsprechend verschiebt. Also wird die 1 ms-Grenze um +200 us vergrössert, so wird auch die 2 ms-Grenze um +200 us vergrössert. Dadurch ist es möglich, dass man im Arbeitsschritt 2 den kompletten Arbeitsbereich des Servos verschieben kann, ohne dass man auch die End-Position 2 ebenfalls neu justieren muss.

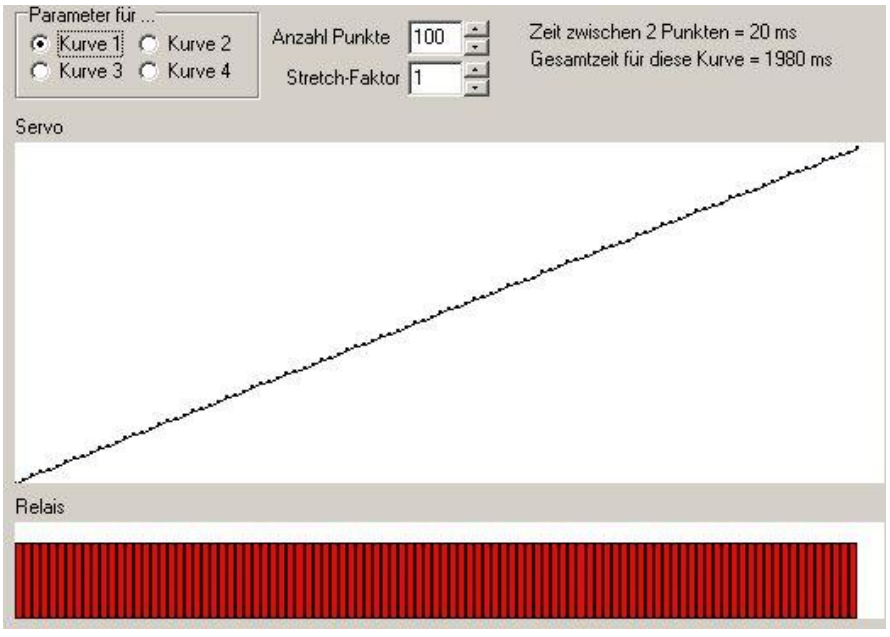
Man sieht: auch wer keine Relais verwendet, ist in dieser Situation gut beraten, die LEDs durch Bestückung von R40..R43 betriebsbereit zu halten.

Variationsmöglichkeiten

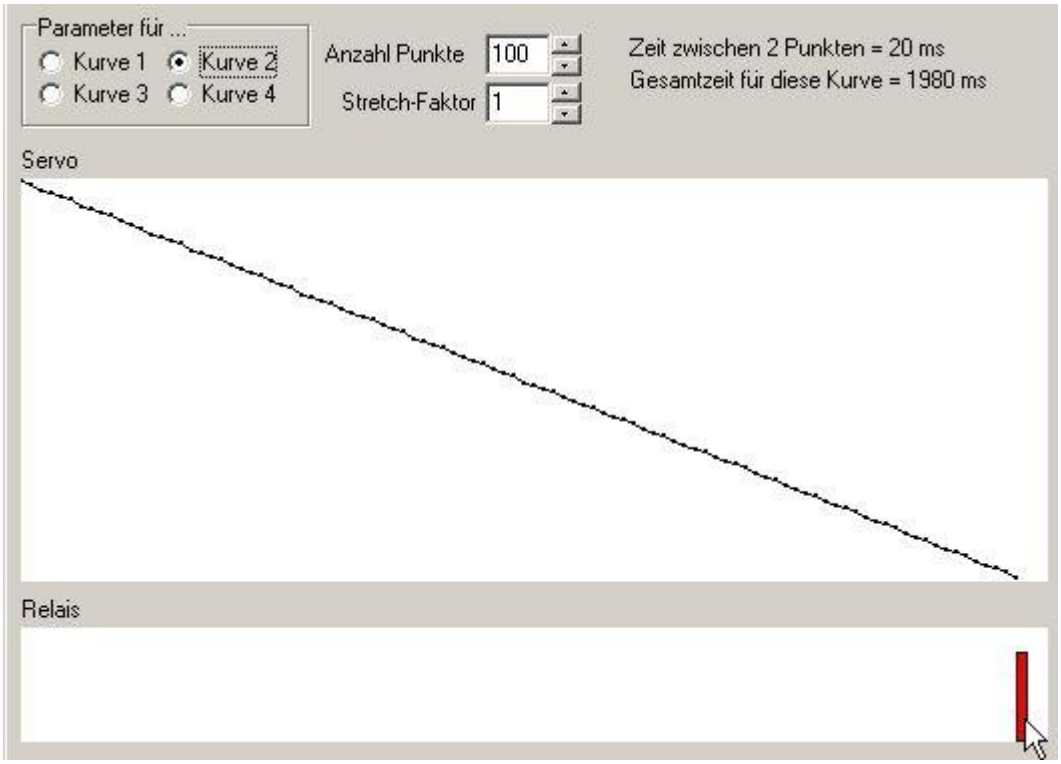
Verwenden der Relais zur Rückmeldung der Weichenstellung

Der SanD besitzt 4 Relais. Wer diese nicht zur Polarisierung von Weichen benötigt (also z.B. die Märklin-Fahrer), kann damit eine Rückmeldung z.B. via S88 (oder andere Bus-Systeme) realisieren.

Zunächst definiert man hierfür die Relais-Stellung bei Kurve 1 (diese stellt z.B. die Bewegung nach RECHTS dar). Zu beachten: Es sind alle Relais-Balken gesetzt, bis auf den letzten. Dadurch wird das Relais ausgeschaltet, wenn die Kurve abgelaufen ist.



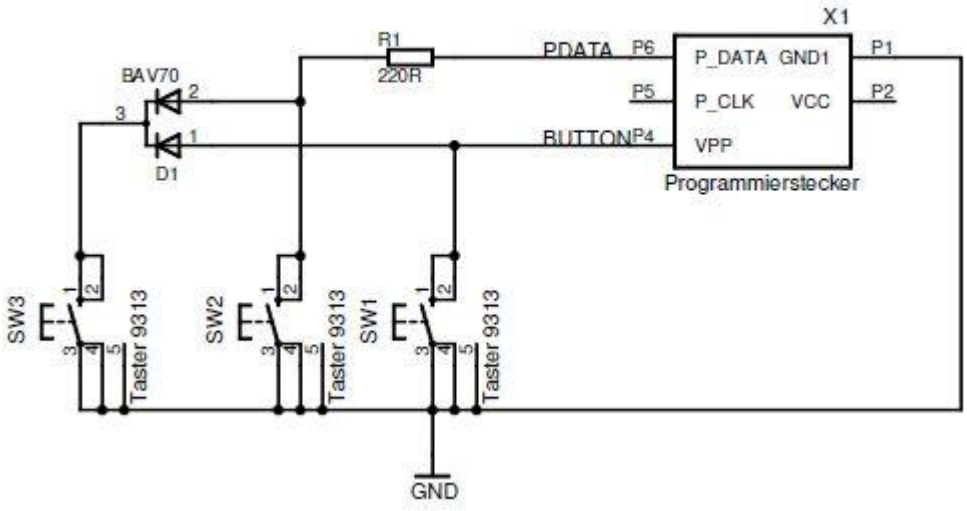
Und jetzt zu Kurve 2 (diese stellt z.B. die Bewegung nach LINKS dar). Zu beachten: Es sind keine Relais-Balken gesetzt, bis auf den letzten. Dadurch wird das Relais eingeschaltet und gehalten, wenn die Kurve abgelaufen ist.



Jetzt hat man folgendes realisiert: Das Relais schaltet am Ende der Bewegung nach RECHTS aus und nach Ende der Bewegung nach LINKS ein. Allerdings gibt es keinen Kontakt für IN_BEWEGUNG oder ähnliches, den es ist immer entweder der Rückmeldekontakt LINKS (= Relais angezogen) oder aber RECHTS (= Relais abgefallen) aktiv. Ein "3. Zustand" (also RECHTS / LINKS / Ich-bin-in-Bewegung-und-weder-Links-noch-Rechts) ist nun mal nicht mit einem Relais zu realisieren, da müsste man dann schon 2 dafür opfern.
Allerdings müsste PC-Steuerung ja wissen können: Wenn ich RECHTS sende und der Rückmelde-Kontakt noch nicht RECHTS anzeigt, dann ist der Servo auch nicht rechts. Entweder bewegt der sich noch, oder er hat den Befehl verpennt, oder....

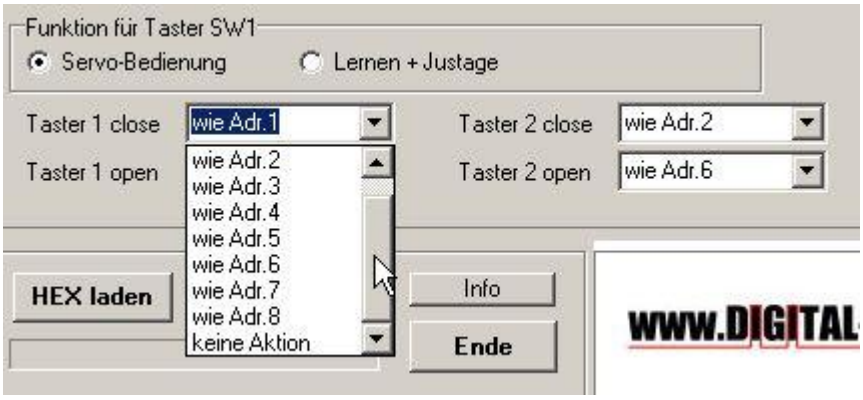
Verwenden der Taster-Eingänge für die Servo-Bewegung

Auf der Justage-Platine sind ja 3 Taster vorhanden. Hier könnte man 2 davon zur Servo-Aktivierung verwenden (der 3. Taster SW3 auf der Justage-Platine stellt lediglich den Zustand "beide Taster gedrückt" dar und kann uns hier leider nicht dienen). Hier noch mal der Schaltplan der Justage-Platine:



Die Taster SW1 und SW2 können also wie hier gezeigt an den SAND angeschlossen und zur Servo-Bedienung verwendet werden. Allerdings gibt es mit SW1 noch einen Haken: dieser Taster entspricht dem SW1 auf der Haupt-Platine SAnD und ist demnach eigentlich für die Adress-Lern Funktion reserviert.

OK. Jetzt genug der Vorrede, goto Praxis. Der Manipulator bietet (ab V0.43) diese Eingabe:



Hier kann ausgewählt werden, ob der SW1 für Servo-Bedienung oder für die "normalen" Aufgaben (Adressen Lernen und Starten des Justage-Modes während der Bootens) verwendet wird oder ob er für die Servo-Steuerung verwendet werden soll.
In diesem Beispiel nun soll SW1 für den Servo verwendet werden. Für jeden Taster gibt es nun 2 Auslöse-Punkte: öffnen und schließen. Tritt nun ein solches Ereignis ein, so reagiert der SanD einfach so, als wenn er die hier definierte Digital-Adresse empfangen hätte. Dadurch kann man z.B. beim Öffnen des Taster die Kurve 1 (= Weiche LINKS), beim Schließen die Kurve 2 (= Weiche RECHTS) abfahren. Somit sind dann z.B. 2 Weichen komplett via Taster (oder eigentlich wären es hier dann eher Schalter) steuerbar.