

Software für Arduino: Ein universeller DCC-Zubehördecoder

Dies ist eine Software für einen DiY-DCC-Zubehördecoder auf Arduinobasis.

Für den Aufbau sind nur Grundkenntnisse in der Arduino-Programmierung notwendig. Der eigentliche Sketch muss normalerweise nicht angepasst werden. Das Verhalten kann über Konstantendefinitionen in einer Parameterdatei eingestellt werden.

Bei entsprechender externer Beschaltung und mit einer geeigneten Zentrale können die CV-Werte sowohl geschrieben, als auch gelesen werden.

Die Hardware muss selbst erstellt werden, ein Bausatz existiert nicht. Beispiel-Schaltpläne sind im Anhang vorhanden.

Die folgende Zubehörtypen können (auch gemischt) eingerichtet werden. Pro Typ können bis zu 3 Portausgänge konfiguriert werden.

- Servos (opt. mit Relais zur Polarisierung von Weichen)
- Doppelspulenantriebe
- statische Ausgänge
- blinkende Ausgänge (optional Wechselblinker)
- Lichtsignale

Bis zu 12 (aufeinanderfolgende) Zubehöradressen sind ansteuerbar
die 1. Adresse ist per Programmierung einstellbar. Die im Einzelfall nutzbare Zahl der Adressen ist von den verfügbaren Ausgängen abhängig.

Die parametrierbaren Eigenschaften des Decoders lassen sich über eine ‚Steuerdatei‘ (DCC_Zubehordecoder.h) einstellen, ohne den eigentlichen Sourcecode der .ino-Datei ändern zu müssen.

Parametrierbar sind:

- Portbelegung für Startverhalten (analoge Eingänge)
- Zahl und Typ der ansteuerbaren Zubehöradressen
- die Ports für die Zubehöranschlüsse
- initiale Grundeinstellungen für die jeweiligen Zubehörtypen (diese können per CV-Programmierung überschrieben werden).

Das Verhalten der konfigurierten Zubehörtypen wird über CV-Programmierung festgelegt:

- bei Servoausgängen die Endlagen und die Geschwindigkeit
- bei Doppelspulenantrieben die Einschaltzeit der Spulen.
- bei blinkenden Ausgängen das Blinkverhalten
- Bei Lichtsignalen die Zuordnung der Ausgangszustände zu den Signalzuständen.

Inbetriebnahme

Zur Inbetriebnahme kann der Decoder in verschiedene Betriebsmodi gebracht werden. Dies wird über Spannungspegel an 2 analogen Eingängen (Standard:A5/A4) beim Hochlauf bestimmt.

Die im folgenden mit A5 bzw. A4 bezeichneten Eingänge lassen sich per Einstellung in der .h-Datei auch auf andere Analogeingänge verlegen.

```
const byte betrModeP    =  A5;  
const byte resModeP     =  A4;
```

A5:

- 5V (nur Pullup) normaler Betriebsmodus, kein PoM
- 3,3V (Spannungsteiler 1:2) PoM immer aktiv, Adresse immer aus defaults
- 1,6V (Spannungsteiler 2:1) IniMode: CV's werden immer auf init-Werte aus .h-Datei gesetzt
- 0V Programmiermodus / PoM (1. Empfangenes Telegramm bestimmt Adresse)

A4:

- Ist A4 beim Programmstart auf 0, werden alle CV's einschl. interner Statuswerte auf die Defaults zurückgesetzt
- Im Betrieb können mit diesem Eingang zur Justierung die Servos in die Mittelstellung gebracht werden (nur wenn die Justierung per Encoder aktiv ist, s. Kapitel ‚Servo‘)

Definition der Zubehörfunktionen

Welche Zubehörfunktionen den jeweiligen Adressen zugeordnet werden, wird in Konstantendefinitionen festgelegt:

```
const byte iniTyp[] = { FCOIL, FSignal2, FSignal0, FSERVO, FSERVO, FSTATIC, FSTATIC };
const byte out1Pins[] = { A2, 9, 12, A0, A1, 7, 8 };
const byte out2Pins[] = { A3, 10, 5, NC, NC, 6, NC };
const byte out3Pins[] = { NC, 11, NC, NC, NC, NC, NC };
```

Die erste Zeile bestimmt die Funktion der jeweiligen Zubehöradresse, die darunterliegenden Zeilen die zugeordneten Ausgangspins. Es müssen nicht allen Adressen 3 Ausgänge zugeordnet werden. Nicht verwendete Ausgänge werden mit NC gekennzeichnet.

Das Verhalten der jeweiligen Funktionen wird über 4 (bei Lichtsignalen 5) CV-Werten pro Zubehöradresse angepasst. Für die erste Adresse sind dies CV50-53. Für alle weiteren Zubehöradressen gilt ein Offset von 5. D.h. z.B. für die 3. Zubehöradresse sind die CV's 60-63 zuständig.

Die Initialwerte dieser CV's werden ebenfalls in der Konfigurationsdatei festgelegt. Diese Werte können per CV-Programmierung überschrieben werden.

```
const byte iniFmode[] = { CAUTOOFF, 0, 0b10000, SAUTOOFF, 0, BLKMODE, 0 };
const byte iniPar1[] = { 50, 0b00010, 0b10100, 0, 0, 50, 0 };
const byte iniPar2[] = { 50, 0b10001, 0b11001, 180, 180, 50, 0 };
const byte iniPar3[] = { 0, 50, 8, 8, 0, 100, 0 };
```

Funktionsbeschreibungen

Servos (initTyp = FSERVO)

Der Servo wird je nach Zustand der DCC-Adresse zwischen 2 konfigurierbaren Endpunkten gefahren. Die Geschwindigkeit ist ebenfalls konfigurierbar. Der Servoimpuls wird am 1. zugeordneten Output-Pin ausgegeben. Es kann jeder Digitalpin verwendet werden.

Ist ein 2. Output-Pin definiert, so wird dieser in der Mitte des Fahrweges des Servos umgeschaltet. Dies kann zur Weichenpolarisierung verwendet werden.

Ist auch der 3. Output-Pin definiert, so werden die Relais im Gegentakt angesteuert. Während der Servobewegung sind beide Relais abgefallen.

CV-Werte:

50+Offs	Fmode	Bit0 = 1: AutoOff der Servoimpulse bei Stillstand des Servo (SAUTOOFF)
51+Offs	Par1	Position des Servo für Weichenstellung '0' (in Grad, 0...180)
52+Offs	Par2	Position des Servo für Weichenstellung '1' (in Grad, 0...180)
53+Offs	Par3	Geschwindigkeit des Servo

Justierung der Servo-Ausgänge:

Die Justierung der Endlagen kann direkt durch Programmieren der CV-Werte erreicht werden. Wird der CV-Wert von Par1 oder Par2 verändert, so wird das Servo direkt auf diese neue Position gefahren. Dies

erleichtert die Einstellung.

Alternativ kann zur Justierung ein Drehencoder angeschlossen werden. Dies vereinfacht die Einstellung noch weiter. Dazu müssen die Kommentarzeichen vor dem ‚#define ENCODER_AKTIV‘ entfernt werden. Die Pins des Encoders können dann nicht mehr für andere Zwecke genutzt werden.

```
//#define ENCODER_AKTIV      // Wird diese Zeile auskommentiert, wird der Encoder ...
const byte encode1P        =  A3;      // Eingang Drehencoder zur Justierung.
const byte encode2P        =  A2;
```

Der Drehencoder wirkt immer auf den Servo und die Servolage, die als letztes über einen DCC-Befehl eingestellt wurde. Sobald der Servo über einen DCC-Befehl wieder verstellt wird, wird die aktuell per Encoder eingestellte Position in den CV's gespeichert.

Wird A4 (Eingang ist konfigurierbar, s.o., Kapitel Inbetriebnahme) während der Einstellung auf 0 gezogen, wird der aktuell vom Drehencoder beeinflusste Servo in die Mittellage gebracht. Sobald der Encoder wieder bewegt wird, bewegt sich das Servo wieder zur vorhergehenden Position.

Doppelspulenantriebe (initTyp = FCOIL)

Diese Funktion dient zur Ansteuerung von Weichen mit Doppelspulenantrieben. Es müssen 2 Ausgangspins zugeordnet werden. Die Länge des Schaltimpulses und die minimale Pause zwischen 2 Impulsen kann konfiguriert werden.

CV-Werte:

50+offs	Fmode	Bit0 = 1: Spulenausgang automatisch abschalten = 0: Spulenausgang über DCC-Befehl abschalten (noch nicht realisiert)
51+offs	Par1	Einschaltdauer der Spule (in 10ms Einheiten)
52+offs	Par2	minimale Ausschaltdauer der Spule (in 10ms Einheiten)
53+offs	Par3	reserviert

Statische / Blinkende Ausgänge (initTyp = FSTATIC)

Mit dieser Funktion kann ein Ausgang statisch ein/ ausgeschaltet werden (z.B. für Beleuchtung). Ausserdem ist es möglich den Ausgang blinken zu lassen.

Werden 2 Ausgangspins zugeordnet, so schalten diese im Gegentakt um.

Im Blinkmodus blinken sie im Gegentakt (Wechselblinker) bei eingeschalteter Zubehörfunktion. Wird per DCC-Befehl ausgeschaltet, so sind beide Ausgänge inaktiv.

CV-Werte:

50+offs	Fmode	Bit0 = 1: Blinken, 0: statisch Bit1 = 1: Beim Blinken starten erst beide Leds dann Wechselblinken Bit2 = 1: mit weichem Auf/Abblenden
51+offs	Par1	Einschaltzeit des Blinkens (10ms Einheiten)
52+offs	Par2	Ausschaltzeit des Blinkens (10ms Einheiten)
53+offs	Par3	Erste Einschaltzeit beim Start des Blinkens

Lichtsignale (initTyp = F SIGNAL2 / FVORSIG)

Zur Ansteuerung von Lichtsignalen können bis zu 3 Zubehöradressen zu einer Signalfunktion zusammengefasst werden. Der ersten (oder einzigen) Adresse wird der Funktionstyp F SIGNAL2 oder FVORSIG zugeordnet, der bzw. den Folgeadressen der Typ F SIGNAL0.

Aus der Zusammenfassung von bis zu 3 Adressen ergeben sich 2 bis 6 mögliche Signalzustände.

Ausserdem können dem Signal bis zu 8 (bei 3 Adressen) Ausgänge zugeordnet werden.

Über CV-Parameter kann frei konfiguriert werden, welche Ausgangspins bei den jeweiligen Signalzuständen

aktiv (= ON) werden sollen. Dabei ist jedem DCC-Befehl ein CV-Wert zugeordnet. Ausserdem ist für jeden Ausgangspin einzeln einstellbar, ob er weich oder hart umschalten soll.

Die zu schaltenden Ausgangspins werden jeweils in den 8 Bits eines CV-Parameters konfiguriert. Dabei folgende Zuordnung:

Bit0 out1pin der 1. Zubehöradresse des Signals
 Bit1 out2pin der 1. Zubehöradresse des Signals
 Bit2 out3pin der 1. Zubehöradresse des Signals
 Bit3 out1pin der 2. Zubehöradresse des Signals
 . usw

Bei den Ausgängen, die für eine ‚weiche‘ Umschaltung konfiguriert ist, wird zwischen altem und neuem Signalbild eine kurze Pause eingefügt, in der alle Ausgänge ausgeschaltet werden. Diese Überblendzeit ist global im .h File einstellbar. Dies gilt ebenso für die Auf bzw. Abblendzeit. Durch das ‚Nachglühen‘ bedeutet dies bei kurzen Überblendzeiten nicht, dass tatsächlich alle angeschlossenen Led's vollkommen dunkel werden.

Beim Programmstart sind alle Lichtsignale im Zustand 0, der Zustand beim Ausschalten des Decoders wird bei Lichtsignalen nicht gespeichert.

Vorsignale

Vorsignale verhalten sich im Prinzip wie Lichthauptsignale. Bei Vorsignalen kann jedoch eine zusätzliche Adresse parametrierbar werden, auf die das Signal auch reagiert. Dies ist in der Regel die Adresse des Hauptsignals, das vom Vorsignal angekündigt wird. Damit kann das Vorsignal automatisch mit dem zugehörigen Hauptsignal geschaltet werden. Hat das Haupt- und Vorsignal mehr als 2 Begriffe, muss die entsprechende Adresse auch bei den Folgeeinträgen angegeben werden

Dunkeltastung:

Befindet sich das Vorsignal zusammen mit einem Hauptsignal an einem Mast, kann es automatisch dunkelgeschaltet werden, wenn das Hauptsignal Hp0 / Hp00 anzeigt. Dazu wird beim Hauptsignal der Index des Vorsignals eingetragen (Position in der Funktionstabelle, beginnend mit 1, 0= kein Vorsignal). Ausserdem ein Bitmuster, das festlegt bei welchen Signalbildern des Hauptsignals das Vorsignal dunkelgeschaltet wird.

CV-Werte:

50+offs	Fmode	Signalmodus: Bit7=1 : invertiert die Softled-Ausgänge (HIGH=OFF) (MobaTools ab V0.9) Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0) Bit0: out1Pin, ... Bit2: out3Pin
51+offs	Par1	Bitmuster der Ausgänge für Befehl 1.Adresse 0 (rot)
52+offs	Par2	Bitmuster der Ausgänge für Befehl 1.Adresse 1 (grün)
53+offs	Par3	FVORSIG: low Byte der Hauptsignaladresse FSIGNAL: Index des Vorsignals am gleichen Mast (0)
54+offs		FVORSIG: high Byte der Hauptsignaladresse FSIGNAL: Bitmuster der Zustände, bei denen das Vorsignal dunkel ist: Bit 0: Befehl 1.Adresse 0 (rot) Bit 1: Befehl 1.Adresse 1 (grün) Bit 2: Befehl 2.Adresse 0 (rot) u.s.w.
CV's der 1. Folgeadresse:		
55+offs	Fmode	Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0) Bit0: out1Pin, ... Bit2: out3Pin
56+offs	Par1	Bitmuster der Ausgänge für Befehl 2.Adresse 0 (rot)
57+offs	Par2	Bitmuster der Ausgänge für Befehl 2.Adresse 1 (grün)
58+offs	Par3	reserviert

59+offs		reserviert
CV's der 2. Folgeadresse:		
60+offs	Fmode	Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0) Bit0: out1Pin, ... Bit2: out3Pin
61+offs	Par1	Bitmuster der Ausgänge für Befehl 3.Adresse 0 (rot)
62+offs	Par2	Bitmuster der Ausgänge für Befehl 3.Adresse 1 (grün)
63+offs	Par3	reserviert
64+offs		reserviert

CV-Werte - Übersicht

folgende CV's werden vom Zubehördecoder verwendet:

CV-Nr	Bedeutung
1/9	1. Adresse (Decoderadresse oder Weichenadresse)
29	Betriebsarteneinstellung nach DCC (Decodertyp, Adressierungsart: Decoder / Output-Adressierung)
47	Kennung für Erstinitiiierung, allgemeine Optionen die für den gesamten Decoder gelten Bit7-4 = 0x50 andere Werte führen beim Hochlauf zur Grundinitiiierung Bit0=1: Automatische Adresserkennung im Programmiermodus Bit1=1: Roco-Modus (dcc-Adresse 0 = Weichenadresse 1) Bit1=0: Std-Modus (dcc-Adresse 4 = Weichenadresse 1)
48/49	PoM-Adresse
50-54	Parameter für 1. Weichenadresse
55-59	Parameter für 2. Weichenadresse
...	
Bedeutung der CV's bei den verschiedenen Funktionen (CV-Nummern für 1. Weichenadresse)	
Servo:	
50	Bit0 = 1: AutoOff der Servoimpulse bei Stillstand des Servo
51	Position des Servo für Weichenstellung '0' (in Grad, 0...180)
52	Position des Servo für Weichenstellung '1' (in Grad, 0...180)
53	Geschwindigkeit des Servo
54	aktuelle Weichenstellung (nicht manuell verändern!)
Doppelspulenantrieb: (derzeit nur mit automatischer Abschaltung)	
50	Bit0 = 1: Spulenausgang automatisch abschalten = 0: Spulenausgang über DCC-Befehl abschalten
51	Einschaltdauer der Spule (in 10ms Einheiten)
52	minimale Ausschaltdauer der Spule (in 10ms Einheiten)
53	-
54	aktuelle Weichenstellung (nicht manuell verändern!)
statischer/Blinkender Ausgang	
50	Bit0 = 1: Blinken, 0: statisch Bit1 = 1: Beim Blinken starten erst beide Leds dann Wechselblinken Bit2 = 1: mit weichem Auf/Abblenden
51	Einschaltzeit des Blinkens (10ms Einheiten)
52	Ausschaltzeit des Blinkens (10ms Einheiten)
53	Erste Einschaltzeit beim Start des Blinkens

54	aktueller Zustand (nicht manuell verändern!)
Lichtsignale	
50	Signalmodus: Bit7=1 : invertiert die Softled-Ausgänge (HIGH=OFF) (MobaTools ab V0.9 Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0). Bit0: out1Pin, ... Bit2: out3Pin
51	Bitmuster der Ausgänge für Befehl 1.Adresse 0 (rot)
52	Bitmuster der Ausgänge für Befehl 1.Adresse 1 (grün)
53	FVORSIG: low Byte der Hauptsignaladresse FSIGNAL: Index des Vorsignals am gleichen Mast (0)
54	FVORSIG: high Byte der Hauptsignaladresse FSIGNAL: Bitmuster der Zustände, bei denen das Vorsignal dunkel ist: Bit 0: Befehl 1.Adresse 0 (rot) Bit 1: Befehl 1.Adresse 1 (grün) Bit 2: Befehl 2.Adresse 0 (rot) u.s.w.
55	Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0). Bit0: out1Pin, ... Bit2: out3Pin
56	Bitmuster der Ausgänge für Befehl 2.Adresse 0 (rot)
57	Bitmuster der Ausgänge für Befehl 2.Adresse 1 (grün)
58	reserviert
59	reserviert
60	Bit 0..2: Bitmuster hard/soft gibt an, welche Ausgänge 'hart' umschalten (Bit=1) und welche Ausgänge weich überblenden (Bit=0). Bit0: out1Pin, ... Bit2: out3Pin
61	Bitmuster der Ausgänge für Befehl 3.Adresse 0 (rot)
62	Bitmuster der Ausgänge für Befehl 3.Adresse 1 (grün)
63	reserviert
64	reserviert

Schaltbild:

Beispiel einer Decoderschaltung für 4 Servos (3 mit Polarisierung), 1 Doppelspulenantrieb und 2 statischen Ausgängen.

Zugehörige Konfiguration aus der .h-Datei:

```
const byte iniTyp[] = { FSERVO, FSERVO, FSERVO, FSERVO, FSTATIC, FSTATIC, FCOIL };
const byte out1Pins[] = { A0, A1, 11, 12, 7, 8, 10 };
const byte out2Pins[] = { 3, 5, NC, NC, 6, NC, 9 };

const byte iniFmode[] = { SAUTOOFF, SAUTOOFF, SAUTOOFF, 0, BLKMODE, 0, CAUTOOFF };
const byte iniPar1[] = { 0, 0, 0, 0, 50, 0, 50 };
const byte iniPar2[] = { 180, 180, 180, 180, 50, 0, 50 };
const byte iniPar3[] = { 8, 8, 8, 0, 100, 0, 0 };
```

