

Sound Servoplatine

++ Diese Seite wird derzeit überarbeitet ++

Über die Servo-Platine 510 können drei Soundmodule angesteuert werden. Es können JQ6500 Module oder MP3-TF16-p/DFPlayer Mini Module verwendet werden. Dabei spielt es keine Rolle ob nur JQ6500 oder nur MP3-TF16-p/DFPlayer Mini oder diese beliebig gemischt, eingesetzt werden. Die drei Module können auch parallel je einen Sound abspielen.

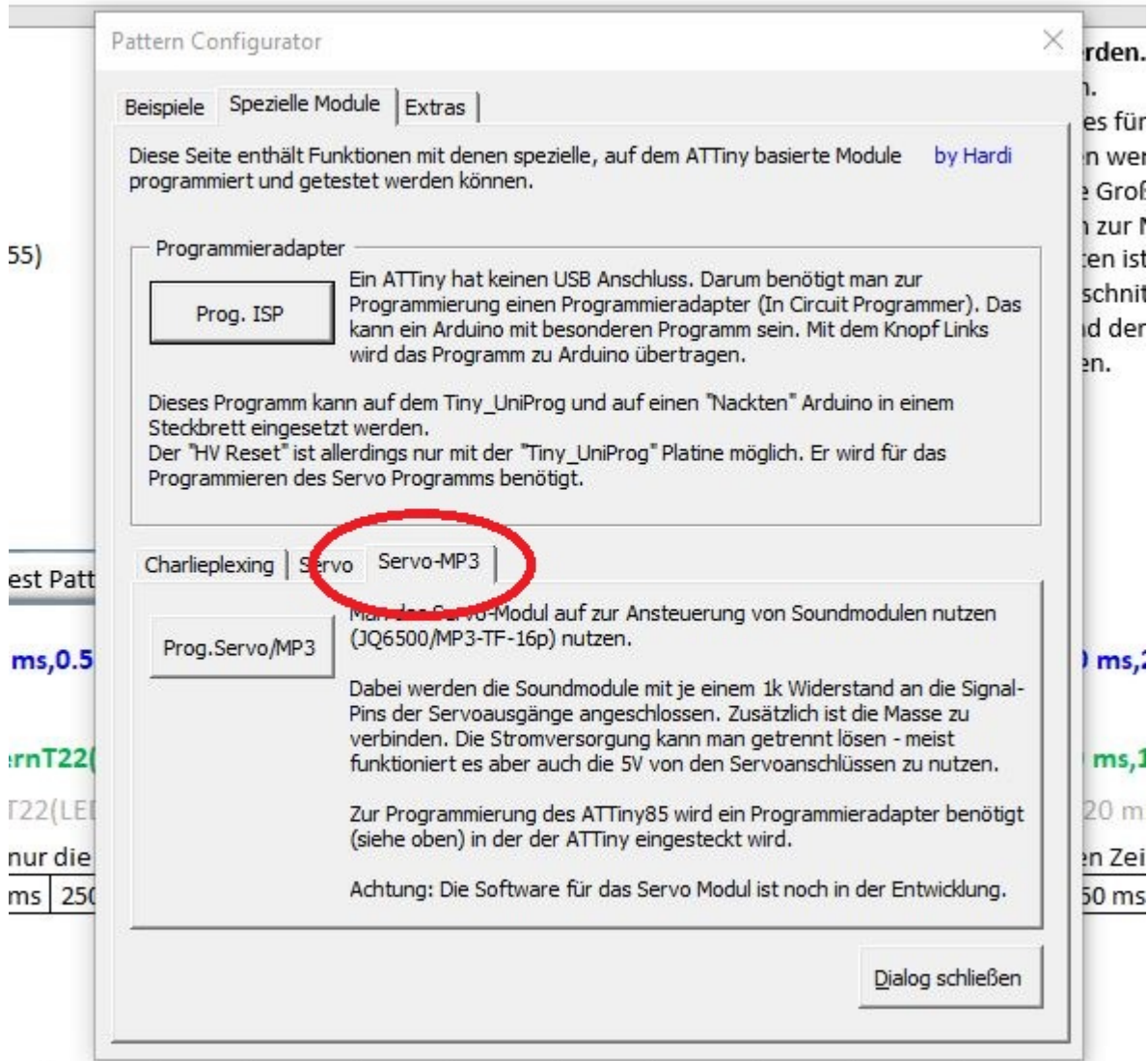
Vorbereitungen Hardware

ATTiny 85

Zunächst muss der ATTiny für die Servoplatine programmiert werden.

Die Anleitung dazu [Programmierung von Attinys für Servo oder Charlieplexing](#) ist im WIKI zu finden.

Im Pattern-Configurator unter spezielle Module **Servo-MP3** auswählen.



Das folgende Bild zeigt die Pins am Ausgang der Servo-Platine. Der SIG-J1 - Pin ist mit dem Eingang des ersten Soundmodules zu verbinden, entsprechend J2 mit Modul 2 und J3 mit Modul3.

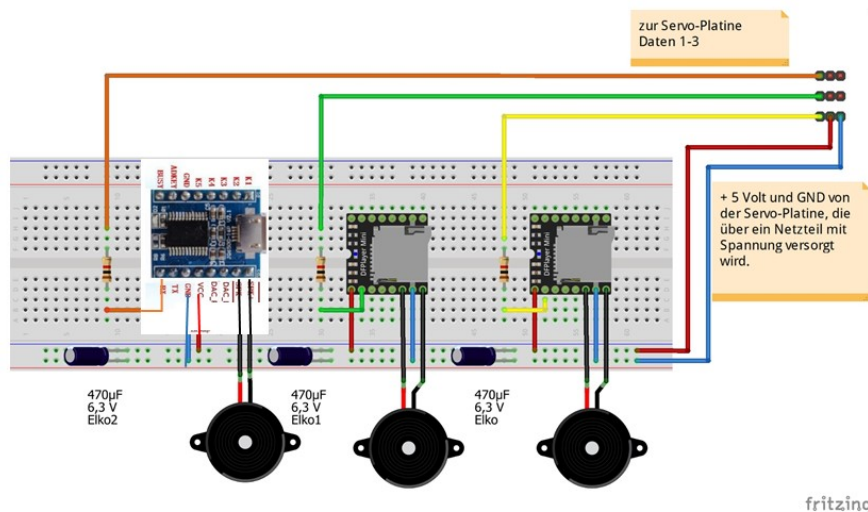


JQ6500 und MP3-TF16-p/DFPlayer Mini



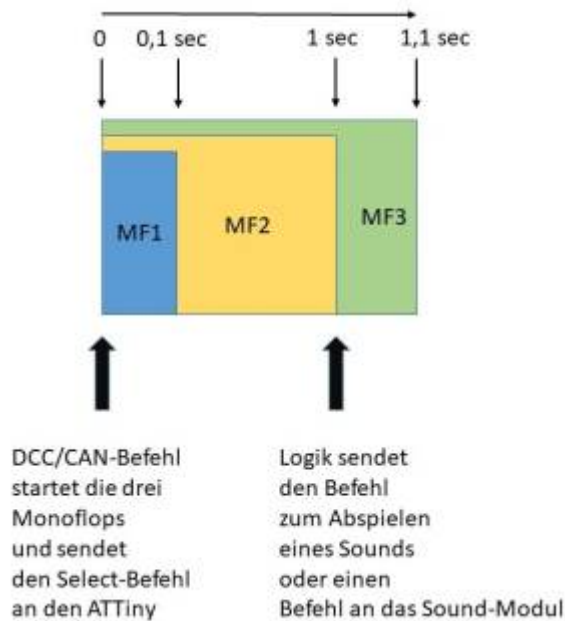
Schaltung

Die drei Ausgänge der Servo-Platine werden nun über je einen 1 kOhm Widerstand mit den Eingängen der Sound-Module verbunden. Die Versorgungsspannung sollte über eine Verteilerplatine mit angeschlossener stabiler Spannungsversorgung erfolgen, da die Leistung an der Hauptplatine zur Versorgung der Sound-Module nicht ausreicht. Die 470uF Elektrolytkondensatoren dienen als Puffer für die recht hohen Einschaltströme der Soundmodule. Eine separate 5V Spannungsversorgung ist ebenfalls möglich. Dann müssen GND und -5 Volt verbunden werden.



- Alle Befehle an diese ATTiny Soundplatine müssen über eine LED Adresse laufen, im Beispiel LED 1. Sollte es (ungewollt) eine Verschiebung geben mit dem Befehl „next LED -1“ wieder auf die LED Adresse der Sound-Platine zurück gehen.
- Über den [Kleinen Verteiler](#) mit der Copy-Funktion, Einstellung über den Jumper, kann man parallel Test-LEDs anschließen und optisch die Funktion überprüfen.

Steuerung über DCC/CAN-Befehle



Vor jedem Sound-Befehl muss das Modul ausgewählt werden auf dem sich die Sound-Datei befindet. So können Sound-Dateien in unterschiedlicher Reihenfolge von den drei Modulen abgespielt werden. Die Logik stellt sicher, dass zunächst über den ATTiny das Modul ausgewählt wird, der Befehl umgesetzt werden kann und dann, mit zeitlichem Verzug, der Track ausgewählt oder eine andere Funktion des Moduls aufgerufen wird.

Beispiel:

Im folgenden Beispiel wird bei Aufruf des:

- DCC-Befehl „3“ der erste Sound des ersten Moduls (JP6500) abgerufen
- DCC-Befehl „4“ der erste Sound des zweiten Moduls (DFPlayer Mini) abgerufen
- DCC-Befehl „5“ der erste Sound des dritten Moduls (DFPlayer Mini) abgerufen

Ver. 3.1.0 by Hardi

Dialog, Z. Arduino schicken, Zeile einfügen, Lösche Zeilen, Verschiebe Zeilen, Kopiere Zeilen, Aus- oder Einblenden, Alle Einblenden, Lösche Tabelle, Optionen, Help

Aktiv	Filter	Adresse oder Name	Typ	Startwert	Beschreibung	Verteiler-Nummer	Stecker-Nummer	Icon	Name	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs	InGrd	Loc InCh	LED/ Sound Kanal
					Modul 1 JQ6500										
✓		3	Rot		startet MonoFloP (MF) 1			🔴	! Mono-Flop	MonoFloP(MF11, #InCh, 0.1 Sek)				1	0
✓		MF11	Rot		sendet Select Befehl für Modul 1			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 1)	1	^ C1-1		1	0
✓		3	Rot		startet MonoFloP (MF) 3			🔴	! Mono-Flop	MonoFloP(MF13, #InCh, 1.1 Sek)				1	0
✓		MF12	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF12u13, NOT #InCh AND MF13)				1	0
✓		MF12u13	Rot		spielt Track 1 vom JQ6500 ab (Welding)			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0
					Modul 2 DFPlayer Mini										
✓		4	Rot		startet MF1			🔴	! Mono-Flop	MonoFloP(MF21, #InCh, 0.1 Sek)				1	0
✓		MF21	Rot		sendet Select Befehl für Modul 2			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 2)	1	^ C1-1		1	0
✓		4	Rot		startet MF2			🔴	! Mono-Flop	MonoFloP(MF22, #InCh, 1 Sek)				1	0
✓		4	Rot		startet MF3			🔴	! Mono-Flop	MonoFloP(MF23, #InCh, 1.1 Sek)				1	0
✓		MF22	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF22u23, NOT #InCh AND MF23)				1	0
✓		MF22u23	Rot		spielt Track 1 aus Root von DFPlayer 1 ab			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0
					Modul3 DFPlayer Mini										
✓		5	Rot		startet MF1			🔴	! Mono-Flop	MonoFloP(MF31, #InCh, 0.1 Sek)				1	0
✓		MF31	Rot		sendet Select Befehl für Modul 3			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 3)	1	^ C1-1		1	0
✓		5	Rot		startet MF2			🔴	! Mono-Flop	MonoFloP(MF32, #InCh, 1 Sek)				1	0
✓		5	Rot		startet MF3			🔴	! Mono-Flop	MonoFloP(MF33, #InCh, 1.1 Sek)				1	0
✓		MF32	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF32u33, NOT #InCh AND MF33)				1	0
✓		MF32u33	Rot		spielt Track 1 aus Root von DFPlayer 2 ab			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0
					Welding Modul 1 JQ6500										
✓		MF12u13	Rot		Imported_Pattern (pc)			🔧	Muster Pattern_Configurator	// Activation: N_Buttons1InCh to TempVar1(#InCh, 1)PatternT26	2	1	1	0	0

Über parallel angeschlossene Test-LEDs, Stichwort Mini-Verteiler, kann der Ablauf optisch sehr gut verfolgt werden. Andere/kürzere Zeitintervalle für die MonoFlops sind möglich und ggf. durch Tests zu ermitteln.

In Zeile 133 wird über die Variable MF12u13 zeitgleich mit dem dazu gehörigen Geräusch vom Sound-Modul 2 ein Schweißlicht ausgelöst. Geräuschlänge und Länge des Lichts können leicht durch Anpassung des **Schweisslicht** über den Pattern-Configurator angepasst werden.

Damit man den Schweißler nicht immer persönlich wecken muss, hier eine Lösung mit der Zufallsschaltung (Random-Funktion).

					Modul 1 JQ6500										
✓		3	AnAus	0				🎲	Zufallsschaltung 1 Ausgang	Random(WL1, #InCh, RN_NORMAL, 3 sec, 15 sec, 1 sec, 1 sec)				1	0
✓		WL1	Rot		startet MonoFloP (MF) 1			🔴	! Mono-Flop	MonoFloP(MF11, #InCh, 0.1 Sek)				1	0
✓		MF11	Rot		sendet Select Befehl für Modul 1			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 1)	1	^ C1-1		1	0
✓		WL1	Rot		startet MonoFloP (MF) 2			🔴	! Mono-Flop	MonoFloP(MF12, #InCh, 1 Sek)				1	0
✓		WL1	Rot		startet MonoFloP (MF) 3			🔴	! Mono-Flop	MonoFloP(MF13, #InCh, 1.1 Sek)				1	0
✓		MF12	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF12u13, NOT #InCh AND MF13)				1	0
✓		MF12u13	Rot		spielt Track 1 vom JQ6500 ab (Welding)			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0
✓		MF12u13	Rot		Imported_Pattern (pc)			🔧	Muster Pattern_Configurator	// Activation: N_Buttons1InCh to TempVar1(#InCh, 1)PatternT26	2	1	1	0	0
✓								🔧	LED Nummer manipulieren	// Next_LED(-2)	3	-2	0	0	0
					Modul 2 DFPlayer Mini										
✓		4	Rot		startet MF1			🔴	! Mono-Flop	MonoFloP(MF21, #InCh, 0.1 Sek)				1	0
✓		MF21	Rot		sendet Select Befehl für Modul 2			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 2)	1	^ C1-1		1	0
✓		4	Rot		startet MF2			🔴	! Mono-Flop	MonoFloP(MF22, #InCh, 1 Sek)				1	0
✓		4	Rot		startet MF3			🔴	! Mono-Flop	MonoFloP(MF23, #InCh, 1.1 Sek)				1	0
✓		MF22	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF22u23, NOT #InCh AND MF23)				1	0
✓		MF22u23	Rot		spielt Track 1 aus Root von DFPlayer 1 ab			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0
					Modul3 DFPlayer Mini										
✓		5	Rot		startet MF1			🔴	! Mono-Flop	MonoFloP(MF31, #InCh, 0.1 Sek)				1	0
✓		MF31	Rot		sendet Select Befehl für Modul 3			🔵	Pin MP3-Modul definieren	MP3_SELECT_MODULE(#LED, #InCh, 3)	1	^ C1-1		1	0
✓		5	Rot		startet MF2			🔴	! Mono-Flop	MonoFloP(MF32, #InCh, 1 Sek)				1	0
✓		5	Rot		startet MF3			🔴	! Mono-Flop	MonoFloP(MF33, #InCh, 1.1 Sek)				1	0
✓		MF32	Rot		logische Verknüpfung der drei MFs			🔗	Logische Verknüpfung	Logic(MF32u33, NOT #InCh AND MF33)				1	0
✓		MF32u33	Rot		spielt Track 1 aus Root von DFPlayer 2 ab			🎵	Titel # abspielen	MP3_TRACK(#LED, #InCh, MP3_PLAY_TRACK, 1)	1	^ C1-1		1	0

From: <https://wiki.mobaledlib.de/> - MobaLedLib Wiki

Permanent link: https://wiki.mobaledlib.de/anleitungen/spezial/sound_servoplatine?rev=172719992

Last update: 2024/09/24 17:46

