

Erster Test für die Anbindung von Soundmodulen an die Servoplatine

Das ZIP-Archiv enthält folgende Dateien/Ordner:

README.pdf – Diese Anleitung

03.ATTiny85_Sound – Verzeichnis mit dem Programm für den ATTiny85 der Servoplatine

Pattern_Configurator_ServoSound.xlsm – Modifizierter Patterngenerator für die Programmierung des ATTiny85

Prog_Generator_MobaLedLib.xlsm – Angepasster Programmgenerator mit den MP3-Funktionen

MP3.h – Include-Datei für den Programmgenerator

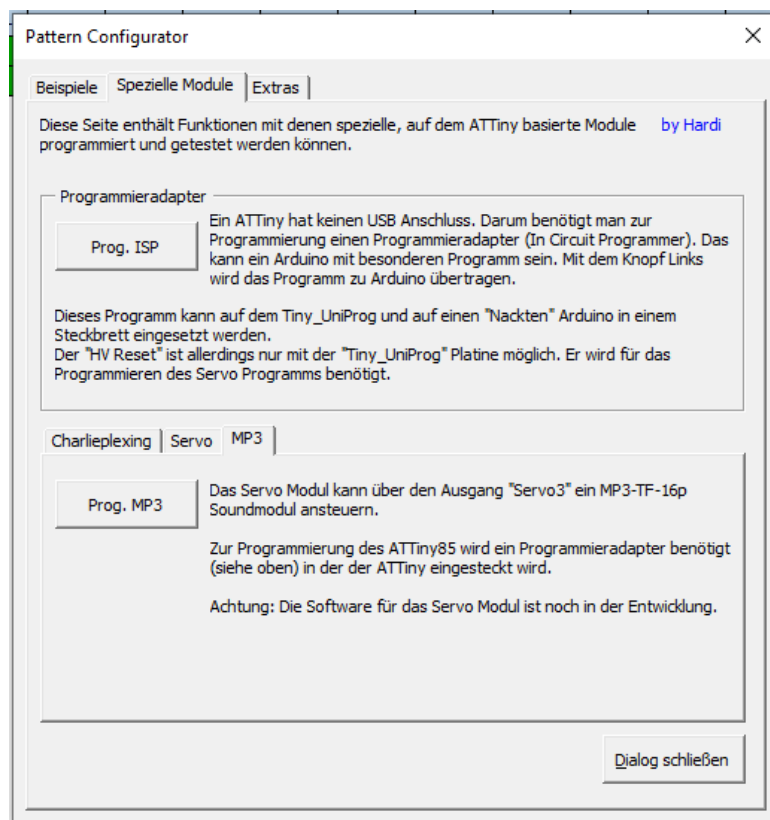
Vorbereitung

Ggf. sind Admin-Rechte erforderlich um die Dateien zu kopieren bzw. Änderungen durchzuführen – je nach Betriebssystem/Rechner.

Das Verzeichnis „**03.ATTiny85_Sound**“ muss in das MobaLEDLib Verzeichnis kopiert werden. Das ist folgendes Verzeichnis:

<C:\Users\<benutzername>\Documents\Arduino\libraries\MobaLedLib\examples\80.Modules>

Anschließend kann die modifizierte Version des Pattern_Configurators gestartet werden. Auf den Kreis oben links klicken. Den Tab „Spezielle Module“ anklicken und unten auf den Tab „MP3“:



Nun den ATTiny-Programmer anschließen, einen ATTiny85 einstecken und mit dem Button „Prog. MP3“ den Prozessor programmieren. Nicht vergessen den programmierten ATTiny anschließend in das Servomodul zu stecken....

Nun muss noch die Datei „MP3.h“ in das Verzeichnis

C:\Users\<benutzername>\Documents\Arduino\MobaLedLib\Ver_2.0.0\LEDs_AutoProg

kopiert werden. In diesem Verzeichnis findet sich auch eine Datei „LEDs_AutoProg.ino“. In dieser (Textdatei, kann auch mit Notepad++ etc. bearbeitet werden) suchen wir folgende Stelle:

```
09.10.20: - Using port writes to speed up the Mainboard_LED function
          - Added additional pins to the Mainboard_LED function. Now nearly every pin could be
            used as LED pin (New channels 0, 5-16). See definition of the LEDx_PINS below
10.10.20: - Added the compiler switch "KeepDarknessCtr"
11.10.20: - New Mainboard_LED defines which end with the pin number (Example: Mainboard_LED_A0)
*/
|
#ifdef __LEDS_AUTOPROG_H__
  #include "LEDs_AutoProg.h" // This file is generated by "Prog_Generator_MobaLedLib.xlsm"
#endif // __LEDS_AUTOPROG_H__

#define SEND_DISABLE_PIN A1 // Pin A1 is used to stop the DCC, Selectrix, ... Arduino from sending RS232 characters
```

Hier muss nun eine Zeile eingefügt werden:

#include "MP3.h"

Anschließend soll es so aussehen:

```
          - Added additional pins to the Mainboard_LED function. Now nearly every pin could
            used as LED pin (New channels 0, 5-16). See definition of the LEDx_PINS below
10.10.20: - Added the compiler switch "KeepDarknessCtr"
11.10.20: - New Mainboard_LED defines which end with the pin number (Example: Mainboard_LED
*/
#include "MP3.h"
|
#ifdef __LEDS_AUTOPROG_H__
  #include "LEDs_AutoProg.h" // This file is generated by "Prog_Generator_MobaLedLib.xlsm"
#endif // __LEDS_AUTOPROG_H__
```

Hardware

Als Servoplatine eignet sich jede 510DE-Platine, bestückt als Servoplatine.

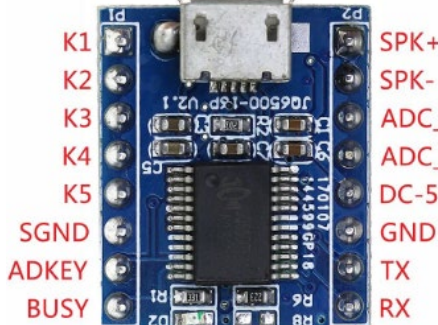
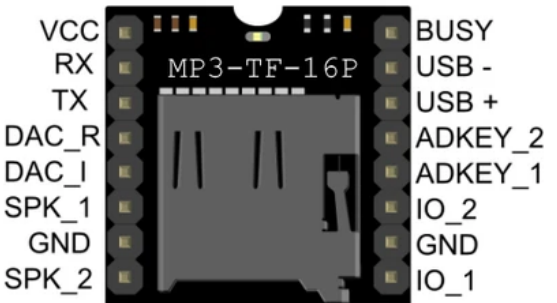
Der Lötjumper für den Servobetrieb SERVO, SERVO1, SERVO2, SERVO3 müssen geschlossen werden.

Es können bis zu drei Soundmodule an eine Servoplatine angeschlossen werden. Wahlweise lassen sich JQ6500 oder MP3-TF-16p verwenden. Das JQ6500 hat den Vorteil das bereits 2 MByte Speicher enthält, was für etliche Soundprojekte ausreichend sein dürfte. Das MP3-TF-16p nutzt eine MicroSD-Karte mit bis zu 32 GByte und vereinfacht durch die mögliche Ordnerstruktur die Verwaltung der Sounddaten.

Folgende Verbindungen sind zwischen 510DE-Platine und Soundmodul herzustellen:

510DE Platine	MP3-TF-16p	JQ6500
Stiftleiste J1/2/3, GND	Pin 7 (GND)	GND
Stiftleiste J1/2/3, VCC	Pin 1 (VCC)	DC-5V
Stiftleiste J1/2/3, SIG	Hier einen Widerstand 1kOhm anschließen. Das andere Ende des Widerstandes kommt an Pin 2 (RX) des Soundmoduls	Hier einen Widerstand 1 kOhm anschließen. Das andere Ende des Widerstandes kommt an RC des Soundmoduls

Zusätzlich muss natürlich ein Lautsprecher an SPK+/SPK- bzw. SPK_1/SPK2 oder ein Verstärker an ADC_L/ADC_R/GND bzw. DAC_R/DACL/GND angeschlossen werden.

JQ6500	MP3-TF-16P
 <p> K1 K2 K3 K4 K5 SGND ADKEY BUSY SPK+ SPK- ADC_L ADC_R DC-5V GND TX RX </p>	 <p> VCC RX TX DAC_R DAC_L SPK_1 GND SPK_2 BUSY USB - USB + ADKEY_2 ADKEY_1 IO_2 GND IO_1 </p>

Die Stromversorgung des Soundmoduls erfolgt durch über das Servomodul. Alternativ kann man auf die Verbindung der Stromversorgung verzichten und das Soundmodul auch direkt versorgen. Je nach verwendetem Lautsprecher kann das Modul theoretisch kurzzeitig bis zu 3 Watt verbrauchen!

Nutzung

Das Soundmodul wird über „Befehle“ angesprochen. Dabei wird der Befehl im Rot-Kanal übertragen, eventuelle Parameter im Grün- und Blaukanal.

Die Verzeichnisstruktur der Sounddateien auf der SD-Karte ist relativ starr vorgeschrieben. Es können Sounddateien (wahlweise im mp3- oder wav-Format) in folgende Verzeichnisse der SD-Karte abgelegt werden:

- Wurzelverzeichnis (Dateinamen 4-stellig 0001.mp3 etc.)
- Verzeichnis ADVERT (Dateinamen 4-stellig)
- Verzeichnis mp3 (Dateinamen 4-stellig)
- Verzeichnisse 01 bis 32 – (Dateinamen 3-Stellig 001.mp3 etc.)

Es werden nur die ersten Zeichen der Dateinamen ausgewertet. Der tatsächliche Dateiname kann länger sein. Damit sind Dateinamen der Art

0023Yesterday.mp3

möglich. Dieser Dateiname wird vom Soundmodul als „0023.mp3“ behandelt. Das erleichtert den Umgang mit den Sounddateien deutlich und sollte ausgiebig genutzt werden.

Je nach Speicherort der Sounddateien müssen unterschiedliche Befehle verwendet werden um eine Sounddatei anzusprechen.

Nach jedem „Befehl“ an das Soundmodul sollte ein NOP-Befehl (RGB-Wert 0/0/0) gesendet werden. Der ATTiny erkennt nur „Änderungen“ von Befehlen. Zwischen den Befehlen sollte jeweils eine kurze Pause sein – das MP3-Modul ist nicht „unendlich schnell“ – besonders dann nicht, wenn sehr viele MP3-Dateien auf der SD-Karte liegen. Werden Befehle mit unterschiedlichen Codes (Rotwert) hintereinander gesendet kann auf den NOP-Befehl zur Trennung verzichtet werden.

Die Daten für das MP3-Modul müssen nach folgender Formel in Helligkeitswerte umgerechnet werden:

Wert * 8 + 4 = Helligkeitswert

Wert	0	1	2	3	4	5	6	7	8	9	10	11	12	...	31
Helligkeit	0	12	20	28	36	44	52	60	68	76	84	92	100	...	252

Bei der Angabe von Ordernummern oder Track-/Dateinummern gibt es noch eine Besonderheit. Das MP3-Modul zählt beginnend mit 1, die Datenübertragung beginnt mit 0. Wenn wir also Ordner 15 ansprechen möchten müssen wir eine 14 senden.

Im roten Kanal wird jeweils der Befehlscode gesendet. In der Dokumentation unten finden sich diese Codes zu Beginn jeder Funktionsbeschreibung. Der Befehlscode muss entsprechend obiger Formel in eine Helligkeit konvertiert werden. Aus Befehlscode 3 wird also ein Helligkeitswert 28.

Parameter der Funktionen werden im Grün- und Blaukanal übertragen. Auch hier wieder umgesetzt in Helligkeiten. Möchte man die Lautstärke also auf den Wert 12 stellen muss im Grünkanal eine 100 übertragen werden. Werte liegen grundsätzlich im Bereich 0-31.

Einige Befehle unterstützen mehr als 32 Werte, z.B. die Befehle um Tracks abzuspielen. Hier wird dann die Tracknummer auf Grün und Rot aufgeteilt. Grün enthält die höherwertigen 5 Bits (MSB), Blau die niederwertigen 5 Bits.

Beispiel:

Gewünschte Tracknummer: 39

Minus 1 = >38

Zerlegen: $38 = 1 * 32 + 6$, also 1 als MSB, 6 als LSB

Umwandeln in Helligkeiten: Grün=12, Blau=44

Bei Verwendung der Definitionen unten wird das bereits berücksichtigt. Man kann dann direkt die Nummer des Tracks (ohne „minus 1“) angeben. Die beschriebene Berechnung wird nur benötigt wenn man z.B. über für Patterngenerator oder „manuell“ die RGB-Werte ermitteln möchte

Es gibt derzeit fünf unterschiedliche Grundbefehle für das Soundmodul:

MP3_CMD	Einfache Befehle ohne Parameter
MP3_SET	Einstellbefehle, Parameter 0-31 im Grünkanal
MP3_TRACK	Befehle um Sounds abzuspielen mit MSB/LSB
MP3_PLAY_FOLDER_TRACK	Abspielen von Sound <Blaukanal> im Ordner <Grünkanal>
MP3_SELECT_MODULE	Setzt den aktuell genutzten Anschluss (1-3)
MP3_SET_TYPE	Setzt den Typ des angeschlossenen Moduls

Nach einer Neuprogrammierung des ATtiny sind die Ausgänge wie folgt vorgelegt:

SERVO1	JQ6500	PIN 5 des ATtiny PB0(MOSI)
SERVO2	MP3-TF-16p	PIN 6 des ATtiny PB1(MISO)
SERVO3	JP6500	PIN 7 des ATtiny PB2(SCK/ADC1)

Möchte man andere Modultypen anschließen, so muss man **einmalig** die verwendeten Module mit dem Befehl MP3_SET_TYPE einstellen. Der ATtiny merkt sich diese Einstellung, daher kann man das einmal nach der Installation mit ein paar Zeilen im Programmgenerator machen:

Aktiv	Filter	Adresse oder Name	Typ	Startwert	Beschreibung	Verteilungs-Nummer	Stecker-Nummer	Beleuchtung, Sound, oder andere Effekte	Serv-LeadV	LEDs	inCh	Loc inCh	LED Ausw
✓		SwitchD1			Anschluss Servo1 setzen			MP3_SET_TYPE(123, #InCh, 1, MP3_TF_16p)	0	C1-1	1	0	0
✓		SwitchD2			Anschluss Servo2 setzen			MP3_SET_TYPE(123, #InCh, 2, MP3_JQ6500)	0	^ C1-1	1	0	0
✓		SwitchD3			Anschluss Servo3 setzen			MP3_SET_TYPE(123, #InCh, 3, MP3_TF_16p)	0	^ C1-1	1	0	0

Als LED-Adresse muss statt „123“ natürlich die tatsächliche Adresse für das Servomodul eingetragen werden und als Modultyp wählt man das jeweils angeschlossene Soundmodul. Mit den 3 Tastern auf der Hauptplatine können die Anschlüsse dann geändert werden.

Die Verwendung funktioniert dann für beide Modultypen ähnlich. Allerdings unterstützt das JQ6500-Modul nicht alle Funktionen.

Sequenzen

Startsequenz

Da bis zu drei Module über eine LED-Adresse angesprochen werden muss man manchmal mehrere Befehle hintereinander senden, also Befehlssequenzen bilden. Das kann mit der MLL auf verschiedene Arten passieren.

Um z.B. die Lautstärke und/oder die Equalizer beim Start des Systems einzustellen kann man Monoflops nutzen. Man beginnt dann zum Beispiel so:

```
MonoFlopInv(Booted1, #InCh, 2 Sek)
MonoFlopInv(Booted2, #InCh, 3 Sek)
```

Damit wird eine Variable „Booted1“ erzeugt die 2 Sekunden nach dem Start der MLL ein Signal erzeugt sowie eine Variable „Booted2“ welche nach 3 Sekunden ein Signal erzeugt.

Diese Signale können wir nun nutzen um weitere Variablen zu setzen:

```
MonoFlop(Init1, #InCh, 0.5 Sek)
MonoFlop(Init2, #InCh, 0.5 Sek)
```

Diese Zeile setzt eine Variable „Init1“ für 0,5 Sekunden auf „An“.

Die Variablen „Init1“ und „Init2“ können wir nun (endlich...) nutzen um einen Befehl z.B. für ein Soundmodul auszulösen:

✓	Sound1	Init1		Select Module 1		MP3_SELECT_MODULE(#LED, #InCh, 1)
✓	Sound1	Init2		Play Track 1		MP3_SET(#LED, #InCh, MP3_SET_VOLUME, 15)

Das Ganze lässt sich (fast) beliebig erweitern. Die Befehle für Soundmodule welche an einem einzelnen Servomodul angeschlossen sind müssen zwingend nacheinander ausgelöst werden. Mehrere Servomodul können aber zeitgleich angesteuert werden. Ist also an LED#1 und LED#2 jeweils ein Soundmodul angeschlossen kann man Init1/Init2 für die Initialisierung beider Module verwenden.

Pattern-Configurator

Alternativ kann man den Pattern-Configurator nutzen um Befehlssequenzen an Soundmodule zu senden. Dazu brauchen wir (meistens) einen Ablauf welcher alle 3 Farben einer LED getrennt ansteuert. Wir tragen daher bei „Anzahl der Ausgabekanäle“ eine 3 ein und als „Bits pro Wert“ eine 8. „Wert Max“ muss auf 255 stehen.

Nun können wir die Sequenzen eintragen. In der ersten Zeile (Rotwert) können wir den Mnemo-Code (siehe Befehlsübersicht unten) des gewünschten Befehls eintragen. In der ersten Spalte typischerweise den Befehl um einen Ausgang (ein Modul) anzuwählen:

```
MP3_SELECT_CMD
```

Die Modulnummer welche aktiviert werden soll tragen wir in den Grünkanal ein. Diese Nummer muss mit dem Makro MP3_BYTE aufbereitet werden. Um das Modul am Anschluss SERVO2 anzuwählen also:

```
MP3_BYTE(2)
```

Die dritte Zeile kann leer bleiben.

In der nächsten Spalte sollten wir einen NOP-Befehl eintragen. In die erste Zeile kommt also:

```
MP3_NOP
```

In Spalte 3 können wir nun zum Beispiel die Lautstärke einstellen. In die erste Zeile kommt daher:

MP3_SET_VOLUME

In die zweite Zeile der gewünschte Lautstärkewert (hier 15) – wieder mit MP3_BYTE aufbereitet:

MP3_BYTE(15)

So kann man nach und nach mehrere Kommandos an die Soundmodule senden. Allerdings verbraucht das einigen Speicher im LED-Arduino.

Es empfiehlt sich mit dem Zeittakt etwas zu experimentieren. Für die meisten Befehle genügt es jeweils „0.1 Sek“ als Dauer einzutragen. Manche Befehle (abspielen von Sounds, besonders wenn viele Sounds auf der SD-Karte sind) brauchen etwas länger. Das muss man ggf. ausprobieren.

Befehlsübersicht

MP3_CMD – MP3_NOP – Nichts machen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode 0 (0x00)

Rot-Kanal: 0

Mnemo: MP3_NOP

Es wird kein Befehl an das Soundmodul übertragen. Das Modul ist bereit für einen neuen Befehl. Insbesondere wenn identische Befehlscode hintereinander übertragen werden muss dazwischen ein MP3_NOP gesendet werden.

Befehl:

MP3_CMD(LED, InCh, MP3_NOP)

MP3_SELECT_MODULE – Anschluss auswählen

Kompatibilität: JQ6500, MP3-TF-16P

Empfänger: ATTiny

Befehlscode 31 (0x1f)

Rot-Kanal: 252

Mnemo: MP3_SELECT_CMD

Stellt den aktuell zu nutzenden Anschluss für das MP3-Modul ein. An jeden Servoausgang kann ein MP3-Modul angeschlossen werden. Wertebereich 1-3. Beim Einschalten ist der Anschluss SERVO1 aktiv.

Befehl:

MP3_SELECT_MODULE(LED, InCh, <anschlussnummer>)

MP3_SET_TYPE – Modultyp für einen Anschluss wählen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode 31 (0x1e)

Rot-Kanal: 252

Mnemo: MP3_TYPE_CMD

Stellt den Modultyp ein der an einem Anschluss angeschlossen ist.

Befehl:

MP3_SELECT_MODULE(LED, InCh, <anschlussnummer>, <modultyp>)

Modultyp kann dabei MP3_TF_16p oder MP3_JQ6500 sein. Die Werte werden intern gespeichert, man muss sie also nicht in das normale MLL-Programm aufnehmen sondern nur einmal zur Konfiguration setzen.

MP3_CMD - MP3_NEXT – Nächsten Sound abspielen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 1 (0x01)

Rot-Kanal: 12

Mnemo: MP3_NEXT

Spielt die nächste verfügbare Sounddatei der aktuellen Wiedergabeart ab (aktueller Ordner, Wurzelverzeichnis etc.). Ist keine weitere Datei vorhanden wird wieder bei der ersten Datei begonnen. Die Reihenfolge entspricht der Kopierreihenfolge beim Beschreiben der SD-Karte

Befehl:

MP3_CMD(LED, InCh, MP3_NEXT)

MP3_CMD – MP3_PREV – Vorigen Sound abspielen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 2 (0x02)

Rot-Kanal: 20

Mnemo: MP3_PREV

Schaltet entsprechend „NEXT“ rückwärts durch die Dateien.

Befehl:

MP3_CMD(LED, InCh, MP3_PREV)

MP3_PLAY_TRACK - Dateien aus dem Wurzelverzeichnis abspielen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 3 (0x03)

Rot-Kanal: 28

Mnemo: MP3_PLAY_TRACK

Parameter: 0-1023 (entsprechend Datei erste bis 1024. Datei im Wurzelverzeichnis)

Spielt eine Sounddatei aus dem Wurzelverzeichnis ab. Die Dateien sind entsprechend der Reihenfolge nummeriert in der sie auf die SD-Karte kopiert wurden. Etwas unübersichtlich, aber zur Kompatibilität mit SD-Karten die für die Sound-Platine mit analoger Ansteuerung bespielt wurden taugt es.

Befehl:

MP3_TRACK(LED, InCh, MP3_PLAY_TRACK, <tracknummer>)

MP3_CMD - MP3_INCREASE_VOLUME – Erhöhen der Lautstärke

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 4 (0x04)

Rot-Kanal: 36

Mnemo: MP3_INCREASE_VOLUME

Erhöht die Lautstärke um 1.

Befehl:

MP3_CMD(LED, InCh, MP3_INCREASE_VOLUME)

MP3_CMD - MP3_DECREASE_VOLUME – Reduzieren der Lautstärke

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 5 (0x05)

Rot-Kanal: 44

Mnemo: MP3_DECREASE_VOLUME

Reduziert die Lautstärke um 1.

Befehl:

MP3_CMD(LED, InCh, MP3_DECREASE_VOLUME)

MP3_SET - MP3_SET_VOLUME – Lautstärke einstellen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 6 (0x06)

Rot-Kanal: 52

Mnemo: MP3_SET_VOLUME

Setzt die Lautstärke auf einen Wert 0-31.

Befehl:

MP3_SET(LED, InCh, MP3_SET_VOLUME, <lautstärke>)

Beispiel:

MP3_SET(1, SwitchD1, MP3_SET_VOLUME, 20)

Setzt die Lautstärke des Soundmoduls/Servoplatine an LED 1 auf 20 wenn auf der Hauptplatine der linke Taster betätigt wird.

MP3_SET - MP3_SET_EQ – Equalizer einstellen

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 7 (0x07)

Rot-Kanal: 60

Mnemo: MP3_SET_EQ

Stellt den Equalizer ein. Mögliche Werte (Grünkanal):

Mnemo	Wert	Bedeutung
MP3_EQ_NORMAL	0	Normale Wiedergabe
MP3_EQ_POP	1	Popmusik
MP3_EQ_ROCK	2	Rockmusik
MP3_EQ_JAZZ	3	Jazz
MP3_EQ_CLASSIC	4	Klassische Musik
MP3_EQ_BASS	5	Bass verstärken

Befehl:

MP3_SET_EQ(LED, InCh, MP3_SET_EQ, <wert>)

Beispiel:

MP3_SET_EQ(1, SwitchD1, MP3_SET_EQ, MP3_EQ_ROCK)

Setzt den Equalizer des Soundmoduls/Servoplatine an LED 1 auf „Rock“ wenn auf der Hauptplatine der linke Taster betätigt wird. Der Befehl wird intern umgesetzt in: R=60, G=20, B=0.

MP3_TRACK - MP3_PLAY_TRACK_REPEAT – Track mit Wiederholung abspielen, Wurzelverz.

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 8 (0x08)

Rot-Kanal: 68

Mnemo: MP3_PLAY_TRACK_REPEAT

Spielt einen Track aus dem Wurzelverzeichnis in einer Endlosschleife ab. Werte 1-1024. Reihenfolge entspricht der Kopierreihenfolge beim erstellen der SD-Karte.

Befehl:

MP3_TRACK(LED, InCh, MP3_PLAY_TRACK_REPEAT, <tracknummer>)

Beispiel:

MP3_TRACK(1, SwitchD1, MP3_PLAY_TRACK_REPEAT, 1)

Spielt die erste Datei aus dem Wurzelverzeichnis des Soundmoduls/Servoplatine an LED 1 ab wenn SwitchD1 betätigt wird und wiederholt sie ständig.

MP3_CMD - MP3_STANDBY – Standby-Modus aktivieren

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 10 (0x0a)

Rot-Kanal: 84

Mnemo: MP3_STANDBY

Schaltet das MP3-Modul in den Stromsparmodus.

Befehl:

MP3_CMD(LED, InCh, MP3_STANDBY)

MP3_CMD - MP3_RESET – Setzt das MP3-Modul auf den Ausgangszustand zurück

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 12 (0x0c)

Rot-Kanal: 100

Mnemo: MP3_RESET

Setzt das Soundmodul zurück (Keine Wiedergabe, Lautstärke 30).

Befehl:

MP3_CMD(LED, InCh, MP3_RESET)

MP3_CMD - MP3_PLAY – Wiedergabe starten

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 13 (0x0d)

Rot-Kanal: 108

Mnemo: MP3_PLAY

Startet die Wiedergabe des aktuellen Tracks (z.B. nach PAUSE).

Befehl:

MP3_CMD(LED, InCh, MP3_PLAY)

MP3_CMD - MP3_PAUSE – Pause

Kompatibilität: JQ6500, MP3-TF-16P

Befehlscode: 14 (0x0e)

Rot-Kanal: 116

Mnemo: MP3_PAUSE

Pausiert die aktuelle Wiedergabe.

Befehl:

MP3_CMD(LED, InCh, MP3_PAUSE)

MP3_PLAY_FOLDER_TRACK – Sounddatei aus einem Ordner abspielen

Kompatibilität: MP3-TF-16P

Befehlscode: 15 (0x0f)

Rot-Kanal: 124

Mnemo: MP3_STANDBY

Spielt die Sounddatei xxx.mp3/wav aus dem Verzeichnis yy der SD-Karte ab. Ordernamen 01 bis 32 (entsprechen Daten 0-31), Dateinamen 001.mp3/wav bis 032.mp3/wav.

Befehl:

MP3_PLAY_FOLDER_TRACK(LED, InCh, <folder>, <track>)

Beispiel:

MP3_PLAY_FOLDER_TRACK(1, SwitchD1, 5, 15)

Spielt die Sounddatei \05\015.mp3 bzw. \05\015.wav ab wenn SwitchD1 betätigt wird (LED-Nummer 1).

Übertragen wird in diesem Fall:

Rot = 124 (Befehlscode), Grün = 36 (interne Ordernummer 4), Blau = 116 (interne Dateinummer 14)

MP3_TRACK - MP3_PLAY_MP3 – Track aus Ordner mp3 abspielen

Kompatibilität: MP3-TF-16P

Befehlscode: 18 (0x12)

Rot-Kanal: 148

Mnemo: MP3_PLAY_TRACK_MP3

Spielt einen Track aus dem Verzeichnis „mp3“ der SD-Karte ab. Dateinamen vierstellig 0001.mp3/wav bis 1024.mp3/wav.

Befehl:

MP3_TRACK(LED, InCh, MP3_PLAY_MP3, <tracknummer>)

Beispiel:

MP3_TRACK(1, SwitchD1, MP3_PLAY_MP3, 1)

Spielt die Datei „\mp3\0001.mp3“ des Soundmoduls/Servoplatine an LED 1 ab wenn SwitchD1 betätigt wird.

MP3_TRACK - MP3_PLAY_ADVERT – Advert abspielen

Kompatibilität: MP3-TF-16P

Befehlscode: 19 (0x13)

Rot-Kanal: 156

Mnemo: MP3_PLAY_ADVERT

Die aktuell laufende Wiedergabe wird unterbrochen und stattdessen ein Track aus dem Verzeichnis „ADVERT“ der SD-Karte abgespielt. Dateinamen vierstellig 0001.mp3/wav bis 1024.mp3/wav. Nach der Wiedergabe des „Adverts“ wird die ursprüngliche Wiedergabe fortgesetzt

Befehl:

MP3_TRACK(LED, InCh, MP3_PLAY_ADVERT, <tracknummer>)

MP3_CMD - MP3_STOP_ADVERT – Advert abbrechen

Kompatibilität: MP3-TF-16P

Befehlscode: 21 (0x15)

Rot-Kanal: 172

Mnemo: MP3_STOP_ADVERT

Stoppt die Wiedergabe des aktuellen Adverts (siehe MP3_PLAY_ADVERT) und fährt mit der Wiedergabe der ursprünglichen Sounddatei fort.

Befehl:

MP3_CMD(LED, InCh, MP3_STOP_ADVERT)

MP3_TRACK - MP3_PLAY_FOLDER_REPEAT – Dateien aus einem Ordner endlos abspielen

Kompatibilität: MP3-TF-16P

Befehlscode: 23 (0x17)

Rot-Kanal: 188

Mnemo: MP3_PLAY_FOLDER_REPEAT

Spielt alle Tracks aus einem Verzeichnis nacheinander ab und beginnt dann von vorne. Ordernamen 01 bis 99.

Befehl:

MP3_TRACK(LED, InCh, MP3_PLAY_FOLDER_REPEAT, <ordnernummer>)

MP3_SET - MP3_SET_REPEAT_CURRENT – Aktuellen Track wiederholen

Kompatibilität: MP3-TF-16P

Befehlscode: 25 (0x19)

Rot-Kanal: 204

Mnemo: MP3_SET_REPEAT_CURRENT

Schaltet die Wiederholung des aktuellen Tracks an oder ab. Flag kann den Wert 0 (nicht wiederholen) oder 1 (endlos wiederholen) haben.

Befehl:

MP3_SET(LED, InCh, MP3_SET_REPEAT_CURRENT, <flag>)