Vorstellung des Laubbläsers von Jürgen mit viel Unterstützung durch Ulrich und Hardi

Der Laubbläser, wie programmieren?

Zunächst mal ein paar Worte zum Laubbläser vorab.
Das Programm ist für meine Hardwarespezifikation entstanden und muss evtl. an einigen Stellen abgeändert werden. So habe ich z.B. noch auf Pos. 1 und Pos.2 auf der SD-Karte Sounds, welche hier nicht berücksichtigt werden, daher kommt der Sound bei mir von der Soundnummer 3,4 und 5.
Mit dem Soundmodul können 14 Sounds (Hardi ich weiß, auch mehr mit tricksen) wiedergegeben werden. Es können also noch weitere Sound für andere Aufgaben genutzt werden, dann darf aber der Laubbläser nicht laufen, da sonst der Sound während der Bewegung nicht wiedergegeben wird.
So nun lasst uns aber starten. Anschauen werden wir als erstes die

Hardwarekonfiguration/ Ausgangslage

Grundplatine 100 bestückt in Grundversion

Adapterplatine 200 mit externem Stromanschluss 5Volt, hier sind angeschlossen:
Stecker 1 Soundmodul MP3-TF-16P Version 1.1 mit Mini-SD-Karte mit 2 MP3-Dateien

- 1x Sound Leerlauf und 1x Sound Vollgas
- 1x MP3-Sound „absolute Stille“

Stecker 2 Servo-Platine mit 1 Servo (Nummer 1)

Was soll passieren?

- Der Mann mit dem Laubbläser startet im Leerlauf
- Der Mann bewegt sich für 9 Sekunden, Sound Vollgas
- Der Mann macht Pause für 10 Sekunden, Sound Leerlauf
- wiederholt das bis der Ausschalter gedrückt wird
- Der Sound ist aus, das Geschehen wird komplett gestoppt

Was gilt es zu programmieren?

- 1. Pattern-Ablauf Bewegung (für den Servo)
- 2. Pattern-Ablauf Sound (für das Soundmodul)
- 3. Programmierung des Prog_Generators

Was müssen wir über das Soundmodul wissen?

Das unter dem Soundmodul befindliche WS2811 Modul wird über die Kanäle Rot und Grün angesteuert und gibt die Schalterbefehle und über Helligkeitswerte an das Modul weiter.
Dabei werden Widerstandskodierte Tasten simuliert, mit denen das Modul normalerweise angesteuert wird.

So nun müssen wir noch die Helligkeitswerte für die Sounds rausbekommen.
Dies „erforschen“ wir über die „MobaLedLib.h“ Bibliothek.
Die Datei finden wir unter Dokumente/Arduino/libraries/MobaLedLib/src/MobaLedLib.h
Diese öffnen wir mit z.B. dem Texteditor und scrollen bis zum

```
//------------------------------------ MP3-TF-16P Sound modul with 4.7uF
capacitor and 2KHz WS2811 ---------------------- 13.10.21:\\
// The new WS2811 modules (model year >2016) generate a 2 kHz PWM Signal
(Old 400 Hz)\\
// Here the filter capacitor could be reduced 4.7uF (Instead of 22uF) to
support also the new\\
// MP3-TF-16P modul which use the GDB3200B chip. With 22uF the new sound
modules didn't work.\\
// Modules with the old MH2024K chip could also be used this macros if the
4.7uF capacitor is used.\\
```

Hier suchen wir dann nach 3 ADKEYs, bei mir ist es Sound 3,4 und 5
Zur Vereinfachung hier die Tabelle für die Helligkeitswerte:

	Helligkeit	**ADKey1**	**ADKey2**
#define SOUND_New_ADKEY10	11	Play Mode	14
#define SOUND_New_ADKEY9	18	U/SD/SPI	13
#define SOUND_New_ADKEY8	23	Loop All	12
#define SOUND_New_ADKEY7	27	Pause/Play	11
#define SOUND_New_ADKEY6	31	Prev/Vol-	10
#define SOUND_New_ADKEY5	39	Next/Vol+	9
#define SOUND_New_ADKEY**4**	**53**	**4**	8
#define SOUND_New_ADKEY**3**	**75**	**3**	7
#define SOUND_New_ADKEY2	148	2	6
#define SOUND_New_ADKEY**1**	**255**	1	**5**

Wir brauchen also die Werte Sound 3 = 75 und Sound 4 = 53, sowie Sound 5 = 255
um die Sounds ansprechen zu können. Diese merken wir uns.

Um die Werte schicken zu können, benötigen wir noch die Schaltzeiten,
diese sind **200 ms** und **10 ms**.

Diese finden wir hier im Editor:

```
//Sound functions could be disable / enabled with the variable
SI_Enable_Sound \\
//#define Sound_ADKey( LED, InCh, ADKey1, ADKey2)\\
//PatternTE2(LED,28,InCh,SI_Enable_Sound,2,0,255,0,PM_SEQUENZ_NO_RESTART,200
ms,10 ms,ADKey1,ADKey2,0,0)\\
```

Nun können wir den Editor wieder schließen.
Falls was versehentlich verändert wurde **nicht** speichern!!!

Was müssen wir über die Servoansteuerung wissen?

Auch die Servo-Platine setzt die vom Verteiler kommenden RGB-Signale um.
Somit wird ein Kanal einem Servo auf der Servo-Platine zugeordnet (z.B. Kanal 1 = Rot)
Die Bewegung wird über den Helligkeitswert des Kanals (z.B. rot) in den Werten von 10-210 gesteuert.
(für diejenigen, die es genauer wissen möchten: \\Servo State Diagram)

Die Servo-Einrichtung für die Geschwindigkeit und die Drehwinkel ist im Wiki beschrieben, hier gehe ich nicht gesondert drauf ein. Hier der Link zur Anleitung:
https://wiki.mobaledlib.de/anleitungen/spezial/tiny-uniprog?do=export_pdf

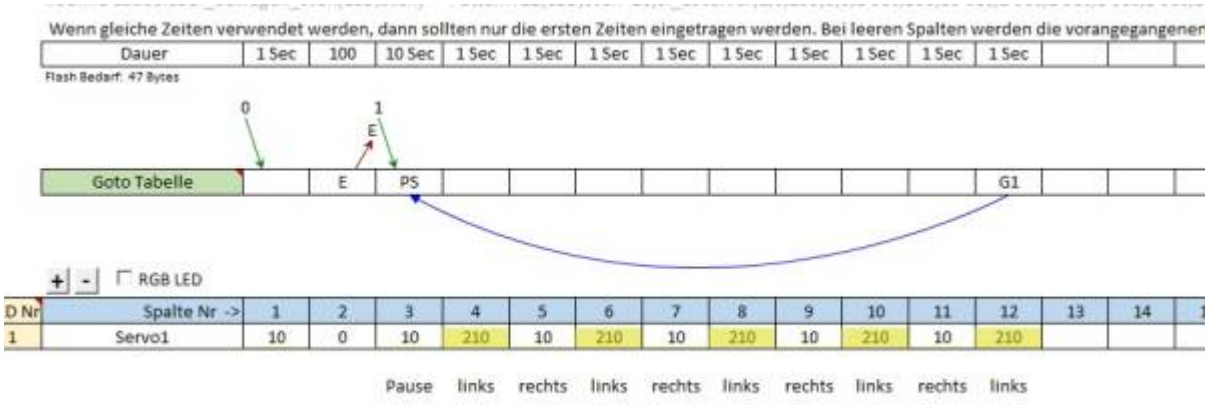
So nun geht es an die eigentliche Programmierung des Pattern 1:

Als erstes machen wir ein neues Blatt (Einstellungen übernehmen? Nein)
Dann geben wir dem Blatt einen Namen wie zum Beispiel: Laubbläser bewegen

Dann setzen wir im gelben Kasten folgende Werte:

Der „Startkanal der RGB LED = 1“ definiert, dass der Servo an den Stecker SV2 angeschlossen ist.
Das ist eine recht verwirrende Geschichte. Tatsächlich ist das Servo an SV1 angeschlossen, aber der ATTiny wurde versehentlich falsch konfiguriert.
Beim Programmieren wurde davon ausgegangen, dass es eine neue Servo Platine ist.

Tatsächlich ist es aber noch eins aus der ersten Charge, bei der der rote und grüne Kanal am WS2811 vertauscht sind.
Das heißt, wenn die **Programmierung richtig** ist, steht der **„Startkanal der RGB LED auf 0“**.
Benennen das Makro mit dem Namen: Laubbläser bewegen.
Danach gehen wir in die Tabelle und füllen diese wie hier aus:



Die Zeile „Spalte Nr.“ benennt unseren Servo (im Beispiel Servo1)
Spalte Nr.1 mit Dauer 1 Sec und Wert 10, sorgt dafür, dass der Servo in die Ausgangsposition fährt

(Damit der Arbeiter an sein Bier kommt 😃)
Spalte Nr. 2 schaltet den Servo ab, damit er nicht brummt.

In die Goto Tabelle geben wir den Wert E für Ende ein, das beendet die erste Startsequenz, welche wir später für den Sound brauchen.
Spalte Nr. 3 wartet für 10 Sec ohne den Servo zu bewegen, in die Goto Tabelle geben wir den Wert P für Position Start und S für Start ein.
(die Werte beim Servo sind 10 bis 210 (min bzw. max.) um hier die Bewegungen zu erzeugen d.h. Wert 10 der z.B. linke Anschlag und 210 z.B. der rechte Anschlag.
Den tatsächlichen Anschlag bestimmt man im Farbtest Programm über die Servo Einstellungen.
Die Werte 10 und 210 liegen außerhalb des normalen Bewegungsbereichs.
Das wird gemacht, damit der Servo nicht brummt. Das Brummen entsteht weil die Messung der WS2811 Helligkeiten und die Messung der Position im Servo geringfügige Störungen haben.

Spalte Nr. 4 bewegt in 1 Sec den Servo auf den anderen Ausschlag
Spalte Nr. 5 bewegt in 1 Sec den Servo wieder zurück zum ersten Anschlag
Spalten 6-12 machen das Gleiche wie die Spalten 3 bis 5.

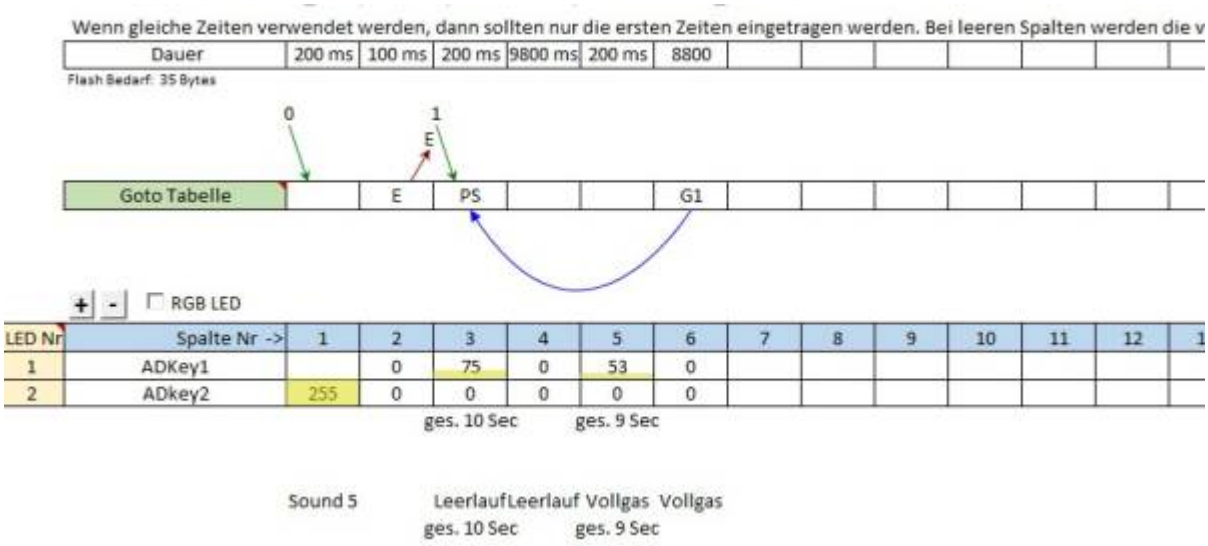
In Spalte 12 tragen wir in die Goto-Tabelle noch den Wert G1 ein, dieser bewirkt das ab der Spalte 3 das Ganze Spiel bis zur Reihe 12 wiederholt wird, bis es beendet wird z.B. durch einen Taster.
Nun müsste die die Tabelle wie oben ausgefüllt aussehen.
Die Anmerkungen unter der Tabelle sind nur zur Veranschaulichung was wo passiert.

Speichern nicht vergessen!!

So nun geht es an die Programmierung des Pattern 2 für den Sound

Hierfür machen wir ein neues Blatt (Einstellungen übernehmen? Nein)
Dann tragen wir wieder die Werte in den gelben Kasten ein:

Dann geben wir dem Blatt wieder einen Namen wie zum Beispiel: Laubbläser Sound
Danach gehen wir wieder in die Tabelle und füllen diese wie folgt aus:



Beginnen wir beim Ausfüllen neben LED-Nr. 1 und LED-Nr.2 die Spaltennamen mit ADKey1 und ADKey2 aus.
Wir erinnern uns, das sind die Ansteuerwerte aus der Editor Geschichte von oben.

Danach gehen wir in Spalte 1 unter Dauer und geben hier 200ms ein (den Wert haben wir oben als Zeitwert rausgesucht) und bei ADkey2 die 255 für den Helligkeitswert für Sound 5 ein. Nun haben wir Sound 5 („Stille“-Ton) definiert.

Als nächstes folgt in Spalte 2 wieder für 100 ms kein Signal, also „Taste“ loslassen.
Das wird benötigt, damit das Soundmodul den „Tastendruck“ akzeptiert.

In der Goto-Tabelle wird mit E das Ende der 1. Sequenz definiert.

Weiter geht es mit der Spalte 3 in der wir bei ADKey1 den Wert 75 als Helligkeitswert
für den Sound 3 ein und aktivieren diesen mit 200 ms, in der Goto-Tabelle tragen wir PS ein,
Bedeutung wie oben: Wert P für Position-Start und S für Start der 2. Sequenz (Schleife)

Spalte 4 füllen wir die Dauer mit 9800 ms und die beiden ADKeys mit den Werten 0
Nun haben wir die Gesamtdauer für den Leerlauf definiert (Spalte 3+4 geben die 10 Sec)

Spalte 5 füllen wir wieder mit 200 ms und den ADKey1 mit dem Wert 53 als Helligkeitswert für Sound 4
und noch die Spalte 6 mit Dauer 8800 ms und die beiden ADKeys wieder mit den Werten 0
Nun haben wir die Gesamtdauer für Vollgas definiert (Spalte 5+6 ergeben die 9 Sec),
jetzt geben wir in Goto-Tabelle noch G1 ein um die 2. Sequenz dauerhaft zu wiederholen

Die Zeile ADKey2 füllen wir von Zelle 2-6 mit je einer 0

Das Speichern nicht vergessen!!

jetzt noch Programmierung im Prog_Generator

im Prog_Generator machen wir uns, indem wir unten auf dem Reiter DCC einen Rechtsklick
machen unter „Verschieben oder Kopieren“ das Unterfenster auf und setzen einen Haken
bei „Kopie erstellen“ dann Klick auf „OK„

Zeile eins sollte die Heartbeat beinhalten und angehakt sein in Spalte B

Nun gehen wir nochmal in den Pattern_Generator und übertragen.
Die Datei „Laubbläser Sound“, diese fügen wir im Prog_Generator unter der Zeile z.B. 4 ein.
Dann geben wir unter „Adresse oder Name“ eine DCC-Adresse ein (z.B. 1) und wählen auf
Nachfrage den Schalter Rot (es werden automatisch rot und grün angelegt)

Dann in der nächsten Zeile 5 machen wir einen Doppelklick, dann sollte sich das
Kontextmenü mit den Auswahlmöglichkeiten öffnen.
Unter Manipulation wählen wir LED manipulieren aus (dazu muss der Expertenmodus aktiviert sein!!)
Parameter 0 als Änderung von StartLedNr. eingeben und Kanal 0 lassen.

Nun gehen wir nochmal in den Pattern_Generator und übertragen die
Datei „Laubbläser bewegen“, wählen die Zeile 6 aus und fügen diese ein.
Dann geben wir wieder wie zuvor die gleiche DCC-Adresse (z.B. 1) ein und wählen auf Nachfrage den Schalter Rot aus. Es werden wieder Rot und Grün angelegt.

Hier auch wieder **das Speichern nicht vergessen!!**

So sollte das Programm im Prog_Generator aussehen:

Nun können wir das Programm an den Arduino schicken.

Jetzt müsste es, nachdem man in der Excel den Grünen Punkt bei „Laubbläser Sound“ oder „Laubbläser bewegen“ anklickt losgehen.

Stoppen kann man das Ganze dann mit dem Roten Punkt.

Der Servo läuft in die Endposition und der Sound 5 „Stille“ ist zu hören 😬 für eine Sec.

So nun viel Spaß beim Nachbau und Laubblasen

Hier habe ich noch ein Video des „rohen“ Zusammenbaus für Euch.
Die RGB-Matrix zeigt, dass außer dem Laubbläser auch noch viele andere Dinge gleichzeitig angesteuert werden können.

Die Sounds als ZIP-File zum Download

projekt_laubblaeser_sound.zip