PlayGround

Nicht alles was in der MobaLedLib möglich ist, lässt sich auch im Dialogfeld unterbringen. Auch sind die Wünsche der einzelnen Anwender doch sehr unterschiedlich. In diesem Kapitel sollen Abläufe/Sketches vorgestellt werden, die funktionieren und selbst in den Prog_Generator zu erfassen sind. Positiver Nebeneffekt, durch die Erfassung wird eine gewisse Lernkurve erzielt, die im Idealfall dazu führt, dass man durch Änderung oder Verwendung einzelner Befehle eigene Abläufe generiert, die dann gerne wieder hier veröffentlicht werden können.

- Beispiel einer Discobeleuchtung:

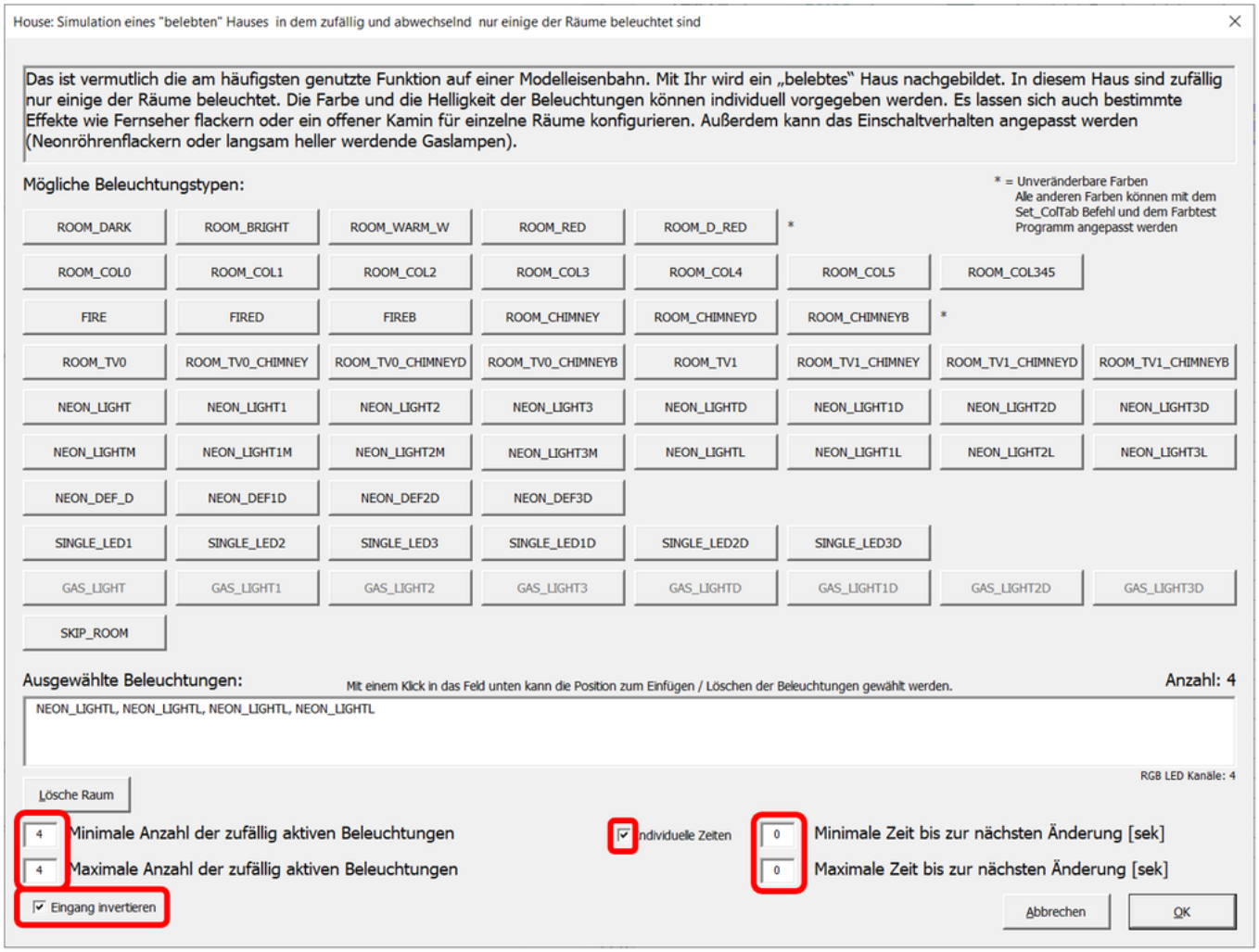
https://www.youtube.com/embed/UjEwNN2QuH8

Mit einer oder mehreren RGB-LED kann der Discobetrieb sowie das „Arbeitslicht“ bei Partyende dargestellt werden. Für die Discobeleuchtung wird das Flashlight benutzt und bei jeder LED die 3 Kanäle mit verschiedenen Zeiten einzeln angesteuert. Die LED ist doppelt zugewiesen, es ist entweder nur Disco oder Tagesmodus aktiviert. Die Umschaltung wurde über einen einfachen Schalter gelöst, ist aber auch über DCC-Ansteuerung möglich. Damit Ihr das gleiche Ergebnis wie in dem Video bekommt, sind im Prog_Generator folgende Zeilen zu erfassen:

Aktiv	Filter	DCC Adresse	Typ	Start-wert	Beschreibung	Verteiler-Nummer	Stecker-Nummer	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs	InCnt	Loc InCh
✓								#define TEST_TOGGLE_BUTTONS			0	0
✓		1	AnAus					Flash(#LED, C_ALL, #InCh, #LocInCh, 5 Sek, 120 Sek)	0	1	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	1	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	1	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	1	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	2	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	2	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	2	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	3	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	3	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	3	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	4	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	4	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	4	C3-3	1	1
✓												
✓		1	AnAus					HouseT_Inv(#LED, #InCh, 4, 4, 0, 0, NEON_LIGHTL, NEON	5	4	1	0

Das Umschalten funktioniert auch ohne DCC. Der Trick ist die erste Zeile. Der Befehl „#define TEST_TOGGLE_BUTTONS“ aktiviert die drei Taster der Hauptplatine zum Simulieren der ersten drei DCC Befehlen. (funktioniert nicht, bei Minimalbelegung der Hauptplatine!) Das geht auch wenn kein zweiter (DCC) Arduino vorhanden ist.

Die Vorgaben für den Flash-Befehl werden über den Button Dialog in den Prog_Generator gezogen. Pro RGB-LED ist es notwendig drei Zeilen zu generieren (Rot / Grün / Blau). Je nach Größe der Disco können mehrere RGB-LED eingesetzt werden. Die Zeilen mit Flash... C1, C2, C3 sooft kopieren wie RGB-LED´s angesteuert werden sollen Für das Arbeitslicht kann man auf den Dialog „House“ zurückgreifen. Wichtig ist hier, dass der Eingang invertiert wird. Dann ist entweder die Disko oder das normale Licht an.



Dadurch dass die „Minimale und Maximale Anzahl der zufällig aktiven Beleuchtungen" gleich der Anzahl der LEDs ist werden alle LEDs angeschaltet. Durch die Verwendung der „Individuellen Zeiten" von 0 werden die Lichter sofort aktiviert. Mit dem Schalter „Eingang invertieren" ist die Beleuchtung an, wenn die Disco aus ist. Hier wurde als Beispiel der Effekt „Neon_LIGHTL" verwendet, mit dem mehrere Neonlichter in einem größeren Raum simuliert werden. (Hier können auch andere Effekte ausgewählt werden-einfach mal ausprobieren!)

Wenn man auch einen „Tag" Modus haben will bei dem weder Neonlichter noch Disco aktiv ist, dann kann man zwei DCC Adressen vergeben. Das sieht dann so aus:

Aktiv	Filter	DCC Adresse	Typ	Start-wert	Beschreibung	Verteiler-Nummer	Stecker-Nummer	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs	InCnt	Loc InCh
✓								#define TEST_TOGGLE_BUTTONS			0	0
✓		1	AnAus					Flash(#LED, C_ALL, #InCh, #LocInCh, 5 Sek, 120 Sek)	0	1	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	1	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	1	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	1	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	2	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	2	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	2	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	3	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	3	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	3	C3-3	1	1
✓		1	AnAus					Flash(#LED, C1, #InCh, #LocInCh, 100 ms, 750 ms)	4	C1-1	1	1
✓		1	AnAus					Flash(#LED, C2, #InCh, #LocInCh, 150 ms, 800 ms)	4	C2-2	1	1
✓		1	AnAus					Flash(#LED, C3, #InCh, #LocInCh, 75 ms, 900 ms)	4	C3-3	1	1
✓												
✓		2	AnAus					HouseT(#LED, #InCh, 4, 4, 0, 0, NEON_LIGHTL, NEON_LI	5	4	1	0

Man könnte die Disco noch weiter perfektionieren. Es könnten verschiedene Beleuchtungsvarianten erstellt werden welche zu verschiedenen Musikstücken passen (Langsam, Schnell, ...). Man könnte eine Spiegelkugel einbauen und, und, und ...

Tipp für eine Variante: Nutzt statt „Flash“ den Dialog „Blinker“ oder „Blinker-HD“, dann habt Ihr auch eine Variante für Schmuse-Songs!

Dieses Beispiel haben Dominik (Moba_Nicki) und Hardi zur Verfügung gestellt.

Beispiel Schweißlicht von Holger

Mit diesem Beispiel soll die Erzeugung eines schweißlichtähnlichen Lichteffekts mit einer RGB-LED und das zeitgleiche Abspielen der entsprechenden MP3-Sounddatei gezeigt werden. Der Sound liegt als MP3-Datei mit einer Dauer von ca. 3 Sekunden auf ein JQ6500 Modul vor. Details zum JQ6500 Modul im WIKI-Link. Entsprechend wird das Schweißlicht ebenfalls eine Dauer von ca. 3 Sekunden haben. Längere Sounddateien können über entsprechende MP3-Bearbeitungsprogramme gekürzt und an eigene Vorstellungen angepasst werden. Harte Schnitte sollten im Fall des Schweißgeräuschs kein Problem sein.

Zunächst den Pattern_Configurator öffnen und ein neues Blatt anlegen, die Einstellungen nicht übernehmen und einen Namen nach eigener Wahl vergeben, hier Welding_2. (Bild Neues Blatt)

In dem neuen Blatt sind im gelb unterlegten Teil die Einstellungen wie im folgenden Bild gezeigt vorzunehmen.

(Bild Einstellungen)

Anmerkungen dazu:

- Ausgabekanäle 4: 3 x für die RGB-LED, der vierte für die Ansteuerung des Sound-Modules. Beim JQ6500-Modul wird der rote Kanal für den Sound gebraucht, gelb und blau sind herausgeführt und können für LEDs, natürlich auch weiße, genutzt werden. Das wird in diesem Beispiel im ProgGen auch so genutzt.
- Bits pro Wert 8: mit der 8 Bit Auflösung lassen sich die 256 Helligkeitsstufen darstellen. Für ein Schweißlicht mit harten Übergängen und wenigen Helligkeitsstufen würde eine geringere Auflösung zur Darstellung völlig ausreichen. Mit 4 Bit ergibt sich nach meiner Einschätzung keine wesentlicher Unterschied im erzeugten LED-Licht. Damit lassen sich ein paar Byte kostbarer Speicher einsparen. Allerdings brauchen wir für die Ansteuerung des Sound-Modules einen exakten Wert, der sich evtl. mit einer geringeren Auflösung nicht erzeugen lässt.
- Analoges Überblenden: ist nur für die Ansteuerung des Sound-Moduls notwendig, für ein Schweißlicht mit kurzen Lichtimpulsen eigentlich nicht. (Ohne den eingeschalteten Analogmodus hat bei mir der Sound nicht funktioniert.)
- Goto Aktivierung: definiert die Methode zum Einlesen der Eingänge im GOTO-Modus. Durch einen Doppelklick auf das gelbe Feld öffnet sich das im Bild gezeigte Fenster. Wir wählen N_Buttons1 aus, da wir mit der Aktion bei „1“ starten wollen also erst bei der Betätigung eines Tasters oder der Ausführung eines Befehls aus einem Programm heraus und nicht bereits „0“, also ohne eine Eingabe.
- Grafische Anzeige: eine „1“ schaltet die grafische Anzeige in den Tabellen ein und erzeugt die „GOTO Tabelle“.

Nun können wir die Tabellen befüllen.

(Bild Pattern_Conf)

Die von mir eingetragenen Werte für die Zeiten und die Werte sind nur ein Beispiel und können natürlich nach eigenen Vorstellungen fast beliebig angepasst werden.

- In der untersten Tabelle für die Werteeingabe muss die Spalte Nr 1 leer bleiben, in der entsprechenden Spalte der Goto-Tabelle tragen wir ein „E“ als „GoEnd-Anweisung“ ein, die Funktion wartet auf ein Ausführungskommando, der logische Zustand ist „0“. Die Dauer ist unbedeutend daher „10 ms“. Bitte beachten: zwischen Zahlenwert und Einheit muss ein Leerzeichen eingegeben werden.
- In der zweiten Spalte der Goto-Tabelle geben wir ein „S“ als Startspalte ein. Der Einsprung in der Programmausführung erfolgt mit der logischen „1“ = Tastendruck, wie grafisch dargestellt.
- In der vierten Spalte der Wertetabelle habe ich den Wert 70 eingegeben, da bei meinem Sound-Modul die entsprechende MP3-Datei über diesen Wert aufgerufen wird. Andere Werte können als Anhalt der folgenden Tabelle entnommen werden. Die Werte können wegen der Bauteiltoleranzen für jedes Sound-Modul unterschiedlich ausfallen und müssen evtl. durch Tests ermittelt werden.

Anhaltswerte für den **JQ6500**

```
* SOUND 5    37
* SOUND 4    49
* SOUND 3    70
* SOUND 2    134
* SOUND 1    255
```

- Die Dauer von 850 ms habe ich eingestellt, damit das Sound-Modul den Befehl sicher erkennt und reagiert.
- In der nächsten Spalte habe ich eine Pause von 400 ms eingefügt, da das Sound-Modul gegenüber der LED träge reagiert und etwas Zeit benötigt bis das gewünschte Geräusch zu hören ist. Der Sound wird nur einmal ausgelöst und läuft dann über 3 Sekunden ab. Ein Abbruch wäre nur über den Aufruf eines anderen Sounds bzw. eines „leeren“ Sounds möglich.
- In den folgenden Spalten sind Werte für die Ansteuerung der RGB-LED eingetragen. Die Werte können zwischen 0 und 255 liegen, x = Maximalwert. Zunächst ist das Licht etwas bläulich, zum Ende des Schweißvorgangs ist das das rötliche Nachglühen zu sehen. Der Umsetzung eigener Vorstellungen und Wünschen steht hier (fast) nichts im Wege.
- Am Ende muss in die Goto-Spalte der Wert „E“ eingegeben werden.
- Sound und Licht können leicht über die Verlängerung der Brenndauer der LED synchronisiert werden. Hier muss man etwas experimentieren.

Nun können wir die Konfiguration mit einem Klick auf die entsprechende Taste an den Progarmm_Generator schicken.

(Bild Prog_Gen_1)

Wir wollen jetzt das Programm nicht direkt zum Arduino schicken und auch nicht direkt zurück zum Pattern_Configurator zurückkehren. Also im Text aufgefordert nur eine Zeile im ProgGen anklicken, in die das Muster geschrieben werden soll. Im nächsten Fenster übernehmen wir den vorgegebenen Wert „0“, Standard LEDs. Nach der Bestätigung mit „OK“ sollte das Ergebnis wie in Zeile 47 aussehen:

(Bild Pg_Gen_3)

Nach der Eingabe einer Adresse und Auswahl eines Tasters können wir unser Ergebnis wie gewohnt zum Arduino schicken und testen.

(Bild Prog_Gen_4)

Mit jedem Aufruf wird die Funktion einmal ausgeführt. Ich habe die Funktion noch mit der Random-Funktion verknüpft, die über einen Ein/Aus-Befehl aktiviert wird und Schweißlicht und Geräusch in unregelmäßigen Abständen aufruft.

(Bild Prog_Gen_5)

In der Random-Funktion habe ich folgende Werte eingegebe:

(Bild Pro_Gen_5_random)

Über die Zielvariable WL1 wird unsere Schweißlichtfunktion aufgerufen. In meinem Beispiel kann über die DCC-Befehle 50 bis 56 weiterhin das JQ6500-Modul wie gewohnt gesteuert werden. Dazu muss über dir NEXT_LED-Funktion der Sprung zur nächsten LED in der Kette rückgängig gemacht werden – NEXT_LED -1! Wie anfangs erwähnt können die beiden verbleibenden Ausgänge auf dem Sound-Modul für LEDs, zum Beispiel Außenbeleuchtung des Lokschuppens, unter Verwendung von „NEXT_LED -1“ genutzt werden, Beispiel in der Programmzeile 41 bis 44.