

Mein Einstieg in das Projekt

MobaLedLib

Ein Erfahrungsbericht
von der Installation
bis zum täglichen Gebrauch
inkl. Anleitung und Tipps

Februar 2021

Inhaltsverzeichnis

Inhaltsverzeichnis	2
Vorstellung	3
Der Weg zur MobaLedLib	5
Was ist die MobaLedLib?	7
Entstehung und Aufbau der MobaLedLib	7
Benötigtes Werkzeug	9
Voraussetzung für den Einsatz der MobaLedLib	10
Etwas Theorie: das Prinzip der RGB LEDs und der „Kette“	11
Anleitung für den ersten Schnelleinstieg zum Ausprobieren	14
Die Platinen	36
Die Hauptplatine (Nr. 100a)	37
Die Verteilerplatine (Nr. 200)	53
Was ist der Programm Generator	59
Das Erstellen von Programmen	66
Ansteuerung der MobaLedLib durch Traincontroller	77
Ideen zum Schalten von Häusern	82
Danksagung	83
Bezugsquellen	84
Links	85

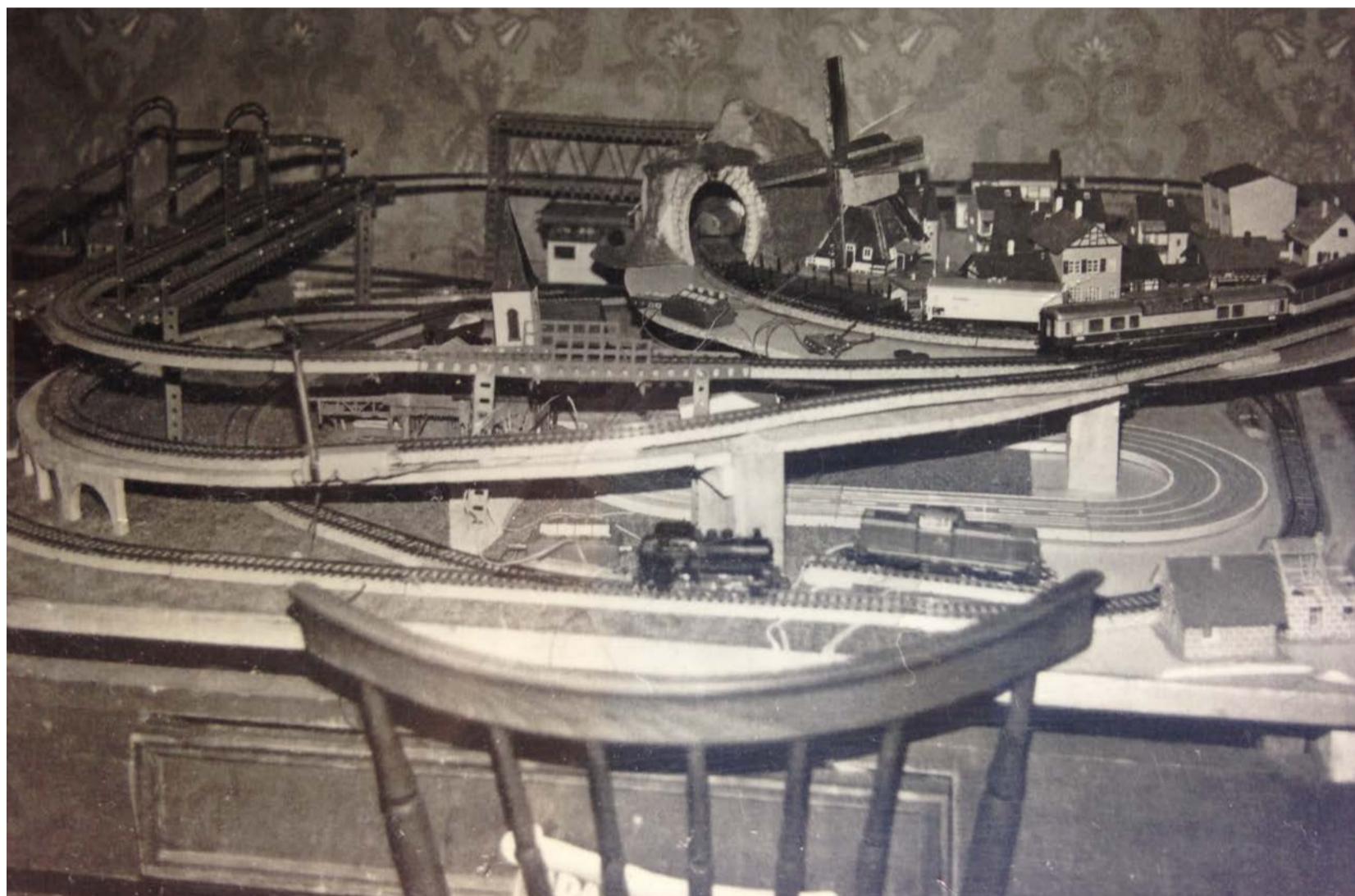
Hier ein Hinweis in eigener Sache:

1. Alle Produkt- oder Firmennamen in meinem Text nenne ich nur des Berichtes halber. Es handelt sich um die Nennungen keinesfalls um Werbung für diese Marken bzw. deren Produkte. Ich habe weder Geld- noch Sachzuwendungen für die Nennungen erhalten.
2. Aus Einfachheitsgründen verzichte ich auf die heute richtigerweise übliche männlich/weiblich/divers-Schreibweisen. Selbstverständlich ist mit dem zum Beispiel mit dem Begriff „Modellbahner“ nicht nur die männliche, sondern auch die weibliche und auch die diverse Ansprache gemeint.
3. Alle von mir beschriebenen Vorgänge, Tätigkeiten, Schaltpläne und sonstige Sachen veröffentliche ich hier ohne Anspruch auf Richtigkeit und Fehlerfreiheit. Alles hier genannte spiegelt lediglich meine Vorgehensweise wieder und stellt keine Garantie dar, dass es bei anderen auch so funktioniert.
4. Einige Texte und Abbildungen habe ich dem MobaLedLib-Wiki (<https://wiki.mobaledlib.de>) entnommen.

Vorstellung

Mein Name ist Jochem Heinen, ich bin Baujahr 1965, und ich übe keinen Beruf aus, der sich mit der Planung, Entwicklung und Herstellung technischer und elektronischer Dinge beschäftigt. Ich bin daher also eigentlich „unvorbelastet“.

Zur Modellbahn bin ich -wie viele andere auch - ganz typisch gekommen: als Kind hatten meine Eltern mir eine kleine Modellbahn geschenkt. Es war eigentlich nur ein Kreis mit einer Ausweichstelle. Im Laufe der Zeit wurde es etwas mehr, aber irgendwann als die Mofa-Zeit begann und ich mobil wurde war Schluss mit Eisenbahn



Meine erste Modellbahn - ca. 1972

Dann ging es weiter: Ausbildung, Freundin, Hochzeit, Kinder, Hausbau. Kinder? Da war doch was? Richtig: ich erinnerte mich der Kartons in der hintersten Ecke im Keller. Schnell waren diese geholt und schon fuhr eine kleine Dampflok mit einigen Anhängern ihre Kreise auf unserem Tisch - ganz zur Freude der Kinder. Naja, die Jungs entwickelten leider keine weiteren Interessen, und so kam die Modellbahn wieder runter in den Keller. Vor einigen Jahren habe ich diese dann für mich wiederentdeckt.

Aber ich hatte irgendwo etwas von Digitalisieren gelesen. Darum beschäftigte ich mich ganz lange mit diesem Thema. Nach ganz vielen YouTube-Filmen begann ich dann, die mittlerweile über 40 Jahre alten Loks zu digitalisieren. Die Steuerung einer eventuell noch aufzubauenden Anlage sollte mittels Computer und entsprechender Software geschehen.

Nach und nach kristallisierte sich folgende Konfiguration heraus, welche ich mir dann auch so zusammenstellte und in einem mittlerweile freigewordenem Kinderzimmer (denn die Jungs sind mittlerweile ausgezogen) aufgebaute:

Gleismaterial: Märklin M-Gleis (noch von früher vorhanden)

Rollmaterial: alles Märklin, alle „alten“ Loks digitalisiert (mit ESU Loci 5)

Digitalzentrale: DiCoStation

Software: Traincontroller Gold

Weichendecoder: Littfinski Magnetartikel-Dekoder S-DEC-4

Rückmelder: Littfinski RM-GB-8-N

Lichtsignal-Decoder: Littfinski LS-DEC-DB

Irgendwann wollte ich dann noch das Beleuchten von vielen Häuschen vorbereiten und dafür legte ich mir noch ein Light-DEC (universelle Anlagenlichtsteuerung von Littfinkis) mit mehreren Light-Display-Modulen (mit jeweils 40 Ausgängen) zu.

Alle Littfinski-Module bestellte ich als Bausätze. So lernte ich wie man lötet und auch die Funktionsweisen der kleinen Bauteile wurden mir langsam wieder klarer.



Hier die digitalisierte V100 - die gleiche Lok ist auf dem vorherigen Bild (1972) im Vordergrund zu sehen.

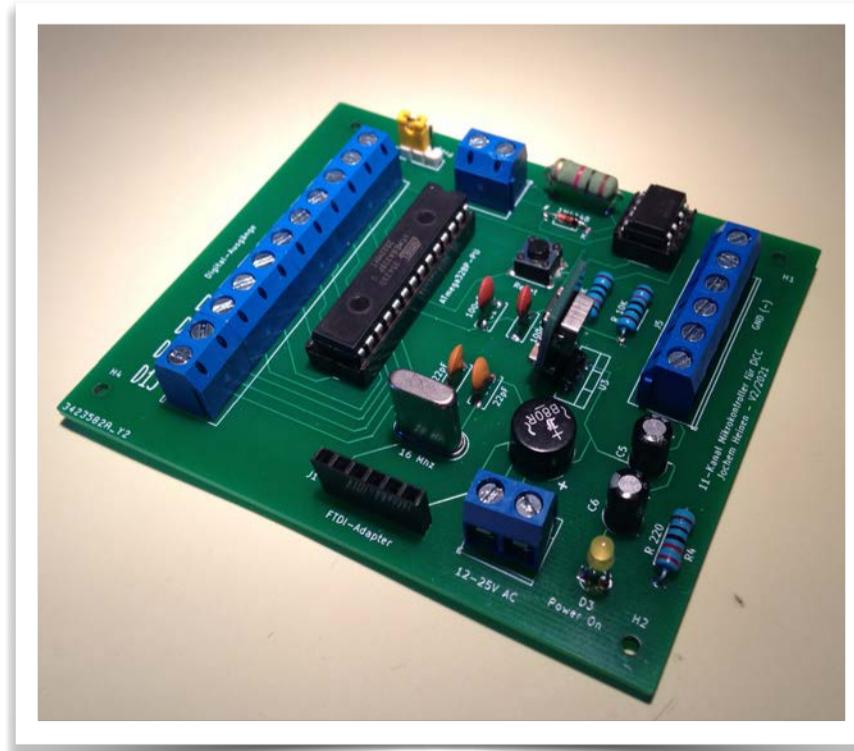
Der Weg zur MobaLedLib

Alles, was ich auf der vorherigen Seite beschrieben habe, funktioniert einwandfrei (Traincontroller, alle Magnetartikel- und Lichtsignaldekoder usw.). Also beschäftigte ich mich dann mit dem Landschaftsbau. Auch hierzu studierte ich viele YouTube-Filme.

Aber irgendwie wollte ich mehr. Elektronische Schaltungen haben mich schon seit jeher sehr interessiert, auch wenn meine Erfahrungen mit solchen Dingen schon mehr als 40 Jahre her sind. Damals konnte ich mich für die sogenannten „Cosmos“-Elektronik-Experimentierkästen begeistern. Aber vieles aus dieser Zeit hatte ich vergessen, bis vor einigen Monaten, als ich zufällig in den Besitz eines Arduino-Startsets kam. Widerstände, Dioden, Transistoren - ich erinnerte mich und es faszinierte mich wieder wie damals. Die ersten Schaltungen waren schnell auf einem Steckbrett aufgebaut und es funktionierte sogar!

Bald versuchte ich dann, diesen Arduino irgendwie in meine Modellbahn zu integrieren. Also suchte ich nach Schaltplänen im Internet, um mittels eines Arduino die DCC-Digitalsignale aus der Zentrale zu empfangen und weiterzuverarbeiten. Diese Schaltpläne fand ich schnell und nachdem die Teile von Elektronik-Versand bei mir ankamen baute ich auf einem Steckbrett die Schaltung (mittlerweile wusste ich auch, was ein Optokoppler ist oder wie bestimmte Mikrokontroller/ICs arbeiten) auf und konnte z. B. mittels eines Schalters in Traincontroller ein Relais umschalten.

Aber das bestückte Steckbrett war für mich nicht alltagsfähig. Also brachte ich mir - wieder mittels YouTube-Videos - die Software „Kicad“ bei. Mit Kicad kann ich Schaltungen im Computer entwerfen und jetzt kommt das Allerbeste: diese Schaltpläne konnte ich sogar in Platinen umwandeln und nachdem ich den einzelnen Elementen entsprechende „Footprints“ zugeordnet hatte konnte ich mir die von mir entwickelten Platinen sogar in 3-D ansehen. Schnell war der nächste Schritt getan: mittels einiger Klicks bestellte ich die Platinen online für kleines Geld und nach einigen Tagen hatte ich diese bei mir auf dem Werktisch. Die zur Bestückung benötigten Einzelteile bestellte ich bei Reichelt und nachdem diese geliefert wurden gab ich mich an das Zusammenstecken und Löten. Und siehe da: nach einiger Zeit hatte ich mit Schaltdecoder „gebaut“, mit denen ich je Platine 11 Funktionen schalten konnte (*Foto rechts*).



Da ich meine Platine weiterentwickelt wollte suchte ich wieder im Internet nach für mich nützlichen Schaltplänen. Dafür melde ich mich auch im sogenannten „Stummiforum“ (www.stummiforum.de) an.

Nach einigen Recherchen in diesem Forum stieß ich auf die „MobaLedLib“. Dies las sich alles sehr interessant und ich wollte im ersten Schritt dieses „System“ ausprobieren und eventuell meine Häuschen auf der Modellbahnanlage abwechslungsreich beleuchten. Aber mehr dazu im nächsten Kapitel.

The screenshot shows a forum post titled "MobaLedLib: 768 LEDs, Servos, Sound, Tageszeitung (siehe #4936)". The post has 5224 replies and is on page 1 of 209. The post content discusses the library and includes a link to a wiki page. It also mentions a Dutch translation of the forum. A profile for user "Hardy" is shown on the right, featuring a photo of a motorcycle and various user statistics.

MobaLedLib: 768 LEDs, Servos, Sound, Tageszeitung (siehe #4936)

Antworten Thema durchsuchen... 5224 Beiträge 1 2 3 4 5 ... 209

MobaLedLib: 768 LEDs, Servos, Sound, Tageszeitung (siehe #4936)

#1 von **Hardy** » Do 20. Dez 2018, 00:47

MobaLedLib: Viel mehr als nur eine Bibliothek zum Ansteuern von LEDs

Einleitung

der MobaLedLib Hauptthread ist verdammt lang geworden. Das schreckt viele ab.
„Das kann man ja niemals alles lesen...“

Das ist auch nicht nötig. Ich betrachte den Thread als so etwas wie eine **Tageszeitung**. Hier findet man alle Neuigkeiten und viele andere Themen zur MobaLedLib.

Wenn Ihr die Seite neu entdeckt habt, dann lest Euch diesen Beitrag hier durch und schaut Euch dann im Wiki um:
<https://wiki.mobaledlib.de/doku.php>

Und verfolgt die neuen Beiträge hier in diesem Thread.

Hier <https://www.stummiforum.de/viewtopic.php...start=4935> habe ich gezeigt wie Ihr ein Abonnement zu diesem Thread einrichten könnt damit Ihr keine Neuigkeiten verpasst.

Der erste Beitrag soll alle Informationen enthalten die man als Einsteiger benötigt.
Oh je, da habe ich mir was vorgenommen...
Aber Ihr könnt mir mit eurem Feedback dabei helfen. Berichtet mir was Ihr vermisst, was Ihr nicht ganz versteht und wie man das Dokument interessanter gestalten kann.

Achtung: Seit Anfang März gibt es ein **eigenes Wiki** für die MobaLedLib:
<https://wiki.mobaledlib.de/>

Hier findet Ihr aller Informationen die Ihr sucht und noch viel mehr. Ganz vielen Dank an Franz (**franz_H0m**) der den Server dazu bereitstellt und an Dominik (**Moba_Nicki**) der sich unermüdlich um die Vervollständigung der Dokumentation kümmert. Und natürlich an alle anderen „Redakteure“.

Sprachen/Languages/Talen
Diese Startseite gibt es dank Misha jetzt auch in Niederländisch:
[Dankzij Misha is deze pagina ook beschikbaar in het Nederlands: \(#1060\)](#)

Screenshot der ersten Seite des mittlerweile sehr umfangreichen Threads über die MobaLedLib. Auf dieser „Einführungsseite“ erklärt der Erfinder der MobaLedLib - Hardy - grundsätzliche Dinge.
Link: <https://www.stummiforum.de/viewtopic.php?f=7&t=165060>

Was ist die MobaLedLib?

Das Wiki der MobaLedLib schreibt darüber:

Mit MobaLedLib lassen sich auf jeder Modellbahn-Anlage Lichteffekte, animierte Figuren, Servo- Motoren und Sound-Module ansteuern. Das System basiert auf den kostengünstigen und millionenfach bewährten Arduino-Microcontrollern. Bis zu 768 Effekte können über nur eine einzige Platine angesteuert werden. Auf jeder Anlage können beliebig viele MobaLedLib-Platinen kombiniert werden. Per DCC oder CAN-BUS kann das System mit vorhandenen Steuerzentralen oder dem PC verbunden werden. Die Programmierung der Effekte erfolgt bequem per Excel-Tabelle am Computer. MobaLedLib-Platinen können über das Forum zum Selbstkostenpreis bezogen und leicht selbst zusammengebaut werden. Bei Fragen hilft die stetig wachsende Gemeinschaft der MobaLedLib-Nutzer im Stummiforum. Bedienungs- und Bauanleitungen sind in diesem Wiki verlinkt. Viel Spaß und willkommen in der Welt von MobaLedLib.

Entstehung und Aufbau der MobaLedLib

Ursprünglich war die MobaLedLib nur eine Arduino Bibliothek mit der man bis zu 256 RGB-LEDs oder 768 einzelne LEDs über eine Datenleitung steuern kann. Das Projekt war zunächst bewusst als Bibliothek und nicht als fertiges Programm gedacht, um die maximale Flexibilität zu erhalten. Eine Lochrasterplatine reichte für die dazugehörige einfache Schaltung anfangs völlig aus. Die populärste Anwendung der MobaLedLib war und ist das „**belebte Haus**“. Über die Steuerung können die in jedem Zimmer verbauten LEDs unabhängig voneinander zufällig ein- und ausgeschalten werden. Durch die Verwendung von RGB-LEDs kann man außerdem jede beliebige Lichtfarbe und Helligkeitsstufe erzeugen. Auf diese Weise sind beliebige Effekte zu simulieren wie Fernseher, sw oder Farbe, offene Kamine, Neonlampen, auch defekte, Gaslaternen, Blitzlichter, Schweißlicht. Alle LEDs werden lediglich über ein vieradriges Kabel miteinander verbunden. Drei Adern für die Stromversorgung und das Steuersignal, die vierte Ader dient als Rückleitung.

Inzwischen hat die MoBaLedLib erheblich an Funktionsumfang gewonnen und kann nicht nur zur Steuerung von LEDs sondern, mit entsprechend verfügbaren Zusatzmodulen, auch für Servos, Schrittmotoren und Soundbausteinen genutzt werden. Die Entwicklung günstiger Platinen (Hauptplatine, Verteilerplatine und

verschiedene Modulplatinen), die über dieses Forum zum Selbstkostenpreis bezogen werden können, machen den Nachbau auch für weniger geübte Bastler leicht möglich.

Auf der Software Seite ist die Unterstützung für den CAN-Bus, das DCC Protokoll und die Selectrix Anbindung hinzugekommen.

Das „**Prog_Generator**“ Programm ist zur zentralen Nutzeroberfläche für alle Softwareeingaben gereift. Damit kann die MobaLedLib ganz ohne Programmierung im vollen Umfang genutzt werden. Im Wesentlichen beschränkt sich die Eingabe auf das Befüllen einer Excel Tabelle mit Hilfe von Auswahlmenüs. Die so festgelegte Konfiguration wird mit einem Tastendruck zum Arduino geschickt. Über der Prog-Generator ist eine Funktion „**LED-Test**“ zum Testen und individuellen Einstellen von LEDs und Soundmodulen aufrufbar. Außerdem können mit dem integrierten **Pattern-Generator** beliebige Muster erzeugt werden. So können beispielsweise auch komplexe Abläufe für Lichtsignale (Charly-Plexing) oder Ampelanlagen individuell entworfen werden.

So weit, so gut. Doch was heißt das für mich? Ich wollte ja erst einmal mit der Beleuchtung starten.

1. Ich benötige eine „Hauptplatine“, auf der zwei Arduino (einer für die Auswertung des DCC-Signals und einer zur Steuerung der LEDs)
2. Zusätzlich brauche ich noch einige „Verteilerplatten“, von denen aus ich dann die einzelnen Leitungen zu den LEDs in den Häuschen ziehe
3. Ich muss NICHT jede einzelne LED mit Kabeln bis zur Steuerplatine anschließen (so wie es bisher bei dem von mir eingesetzten Littfinski-Licht-System war)
4. Die Verbindung zwischen Hauptplatine und den Verteilerplatten und weiter zu den Häuschen geschieht über ein 4- oder 6-poliges Flachkabel
5. Ich kann alle möglichen „Lichteffekte“ auf den LED-Arduino überspielen.

... und das alles möchte ich auf den nächsten Seiten zeigen!

Benötigtes Werkzeug

Für das Zusammenstellen der Platinen benötige ich folgendes Werkzeug:

1. Lötkolben mit feiner Spitze, dazu Lötdraht
2. Pinzette
3. kleinen Seitenschneider
4. Bastelmesser
5. Entlötlitze oder Entlötpumpe (um überflüssiges oder fehlerhaftes Lot zu entfernen)
6. sehr hilfreich ist eine sogenannte „dritte Hand“
7. und mittlerweile benötige ich für manche Arbeiten meine Arbeitslampe mit integrierter Lupe.



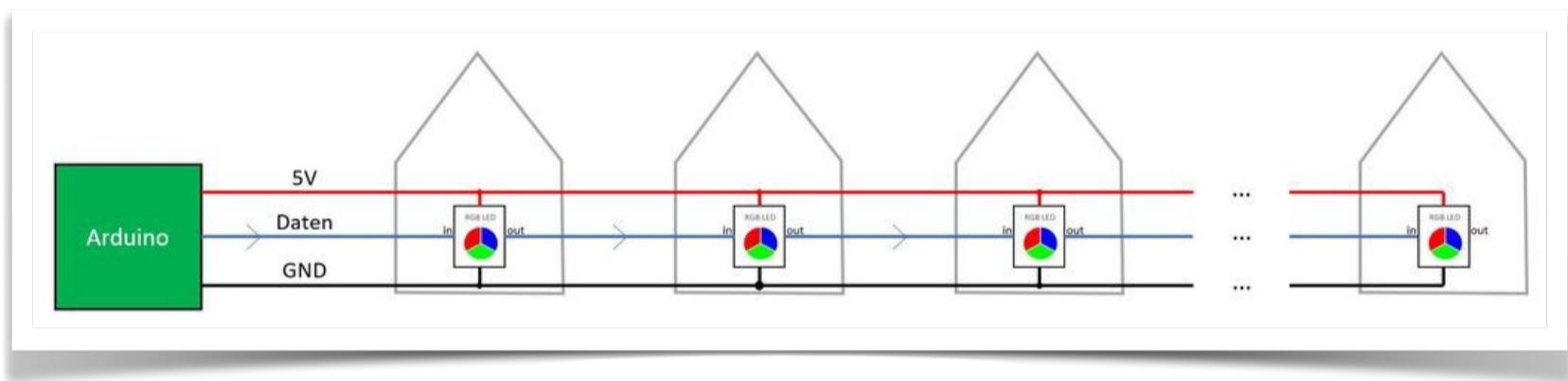
Mein Hobby-Arbeitsplatz - ausnahmsweise einmal aufgeräumt :-)

Voraussetzung für den Einsatz der MobaLedLib

Neben den selbst zu lötenden Platinen benötigt man für die Software einen PC mit aktuellem Betriebssystem (WIN 10). Dazu zwingend Microsoft Excel (für die „Programme“ Prog_Generator und Pattern_Generator usw.) und die Arduino IDE (kostenlos, Hinweise zur Installation folgen auf den nächsten Seiten). MS Excel-Alternative wie OpenOffice oder ähnlich können für die problemlose Nutzung der MobaLedLib nicht empfohlen werden. Als LEDs werden sogenannte WS2812 RGB-LEDs genutzt. Diese kann man z. B. in Form von „Schokoladentafeln“ kaufen, die einzelnen LEDs werden dann einfach von der Tafel getrennt.

Etwas Theorie: das Prinzip der RGB LEDs und der „Kette“

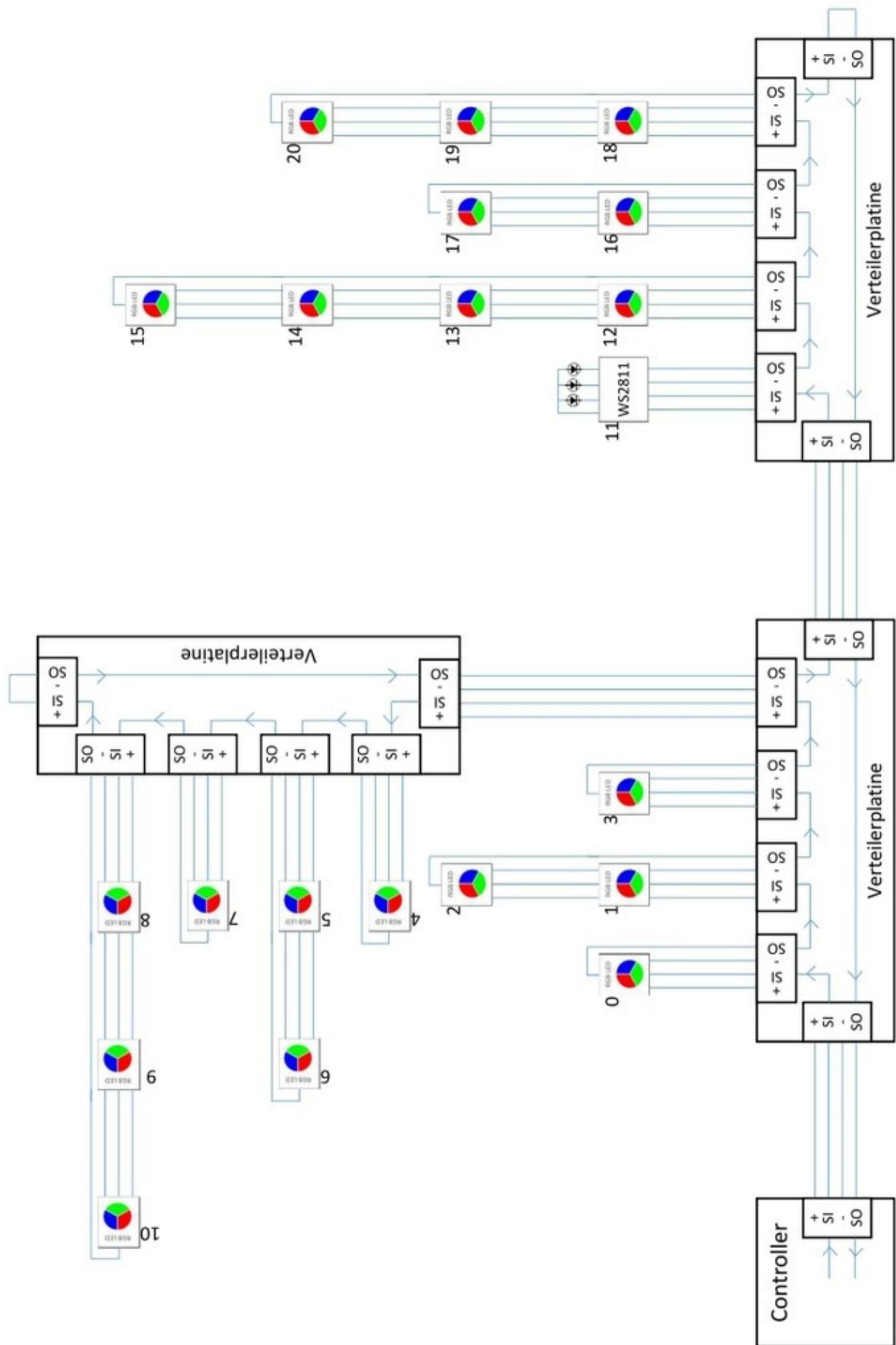
Die LEDs sind in einer Kette angeordnet. Jede LED besitzt einen Daten-Eingang und Daten-Ausgang. Die erste LED ist mit dem Arduino verbunden. Die zweite LED bekommt ihr Signal vom Ausgang der LED Nummer 1. Sie reicht die Daten über ihren Ausgang an die nächste LED weiter.



Alle LEDs haben eine gemeinsame Plusleitung (Rot) und eine gemeinsame Masseleitung (Schwarz). Die Datenleitung verbindet immer nur zwei benachbarte LEDs. Die Teilstücke der Datenleitung sind NICHT elektrisch miteinander verbunden. Das ist ganz entscheidend für das Ansteuerungsprinzip der LEDs.

Die erste RGB LED liest die Daten vom Arduino ein, und merkt sich die drei Helligkeitswerte für ihre Rote, Grüne und Blaue LED. Wenn der zweite Datensatz vom Arduino kommt dann reicht die LED diesen an die nächste LED weiter. Damit bekommt der Chip in der zweiten RGB LED nur den zweiten Datensatz des Arduinos, weil die erste LED den ersten Datensatz für sich beansprucht hat. Die Nummer zwei merkt sich ebenso die ersten empfangenen Helligkeitswerte und reicht alle folgenden weiter. Damit bekommt die dritte LED erst den dritten Datensatz - aber das funktioniert alles automatisch und extrem schnell. Diese Datenübertragung ist so schnell, dass innerhalb von 4 Millisekunden 100 LEDs mit Daten versorgt werden können.

Elektrisch werden die LEDs in einer Kette angeordnet. Der Ausgang einer LED ist mit dem Eingang der nächsten LED verbunden. Auf der Modelleisenbahn ist diese Anordnung aber unpraktisch. Durch die Verwendung einer vierten Leitungen als Daten-Rückleitung vom Ausgang der letzten LED in einem Strang zu der ersten LED im nächsten Strang kann man die LEDs beliebig anordnen. Die Verteiler erleichtern den Aufbau erheblich. Das zeigt das nächste Bild:



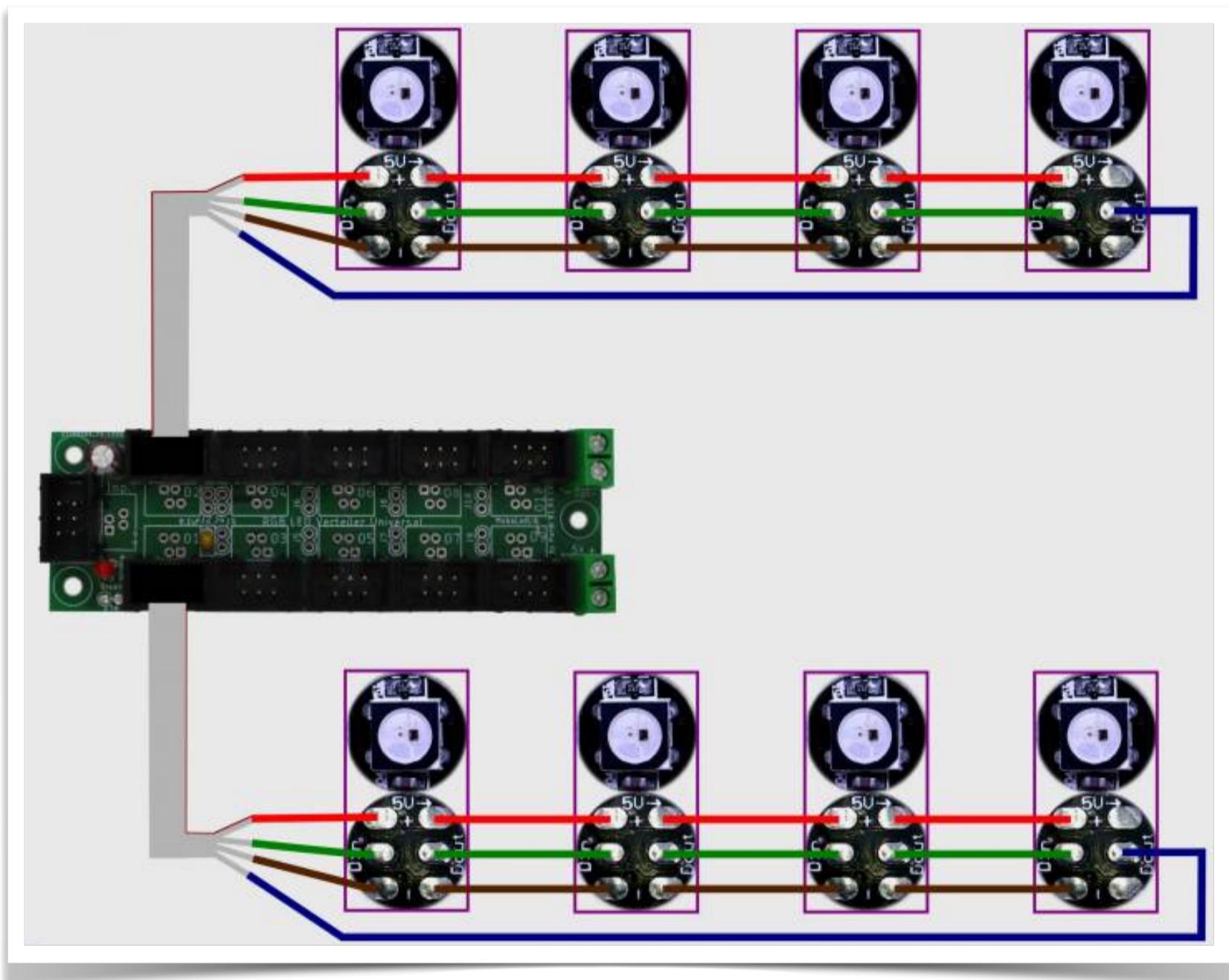
Neben den LEDs sind kleine Nummern welche die „Adresse“ der LED beschreibt.

Hier ein Anschlussbeispiel für zwei Ketten mit WS2812B.

Dabei sind jeweils vier RGB-LEDs in einer Reihe geschaltet.

Von der letzten LED geht ein Kabel zurück zum Flachbandkabel, damit das LED-Signal wieder zurück zum Verteiler kommt.

Jede LED ist dabei mit Ihrer Vorder- und Rückseite abgebildet.



Anleitung für den ersten Schnelleinstieg zum Ausprobieren

Um die MobaLedLib und deren Möglichkeiten testweise ausprobieren zu können benötigt man noch keine Platinen. Nachstehend füge ich die Anleitung aus dem MobaLedLib-Wiki ein. Nach dieser bin ich auch vorgegangen und hatte keinerlei Probleme:

1. Schritt: Installation ARDUINO IDE

Als Einstieg und zum Ausprobieren reichen

- ein Arduino (Original oder preiswerter Clone)
- ein paar WS2812 RGB-LEDs und
- drei Kabel

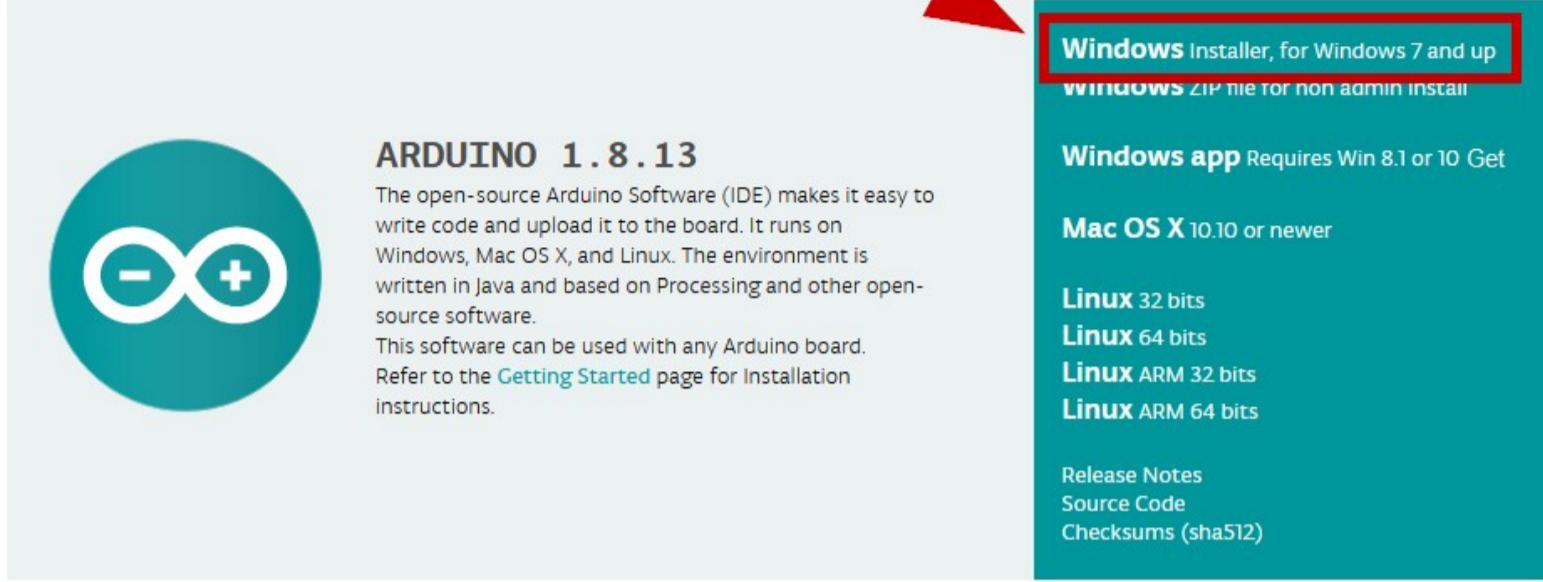
völlig aus. Der Arduino kann ein Uno, Nano oder auch ein Mini Pro o.ä. sein.

Zunächst muss man die aktuelle Arduino Entwicklungsumgebung IDE (Integrated Development Environment) herunterladen und installieren.

Die ARDUINO Download Seite erreicht man über den Link: <https://www.arduino.cc/en/Main/Software>

Die MobaLedLib läuft mit allen aktuellen Versionen der Arduino IDE und wurde mit den Versionen 1.8.8 - 1.8.13 erfolgreich getestet. Für die Unterstützung aller Funktionen der MobaLedLib empfehlen wir die aktuelle Version 1.8.13 zu installieren. Der Arduino WEB-Editor oder die Arduino App können für die MobaLedLib nicht genutzt werden.

Download the Arduino IDE



ARDUINO 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

Windows Installer, for Windows 7 and up
[Windows ZIP file for non admin install](#)

Windows app Requires Win 8.1 or 10 Get

Mac OS X 10.10 or newer

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

[Release Notes](#)
[Source Code](#)
[Checksums \(sha512\)](#)

HOURLY BUILDS

LAST UPDATE
17 August 2020 13:24:0 GMT

Download a **preview of the incoming release** with the most updated features and bugfixes.

[Windows](#)
[Mac OS X](#) (Mac OSX 10.10 or later)
[Linux 32 bit](#), [Linux 64 bit](#), [Linux ARM](#), [Linux ARM64](#)

BETA BUILDS

β BETA

Download the **Beta Version** of the Arduino IDE with experimental features. This version should NOT be used in production.

[Windows](#)
[Mac OS X](#) (Mac OSX 10.10 or later)
[Linux 32 bit](#), [Linux 64 bit](#), [Linux ARM](#), [Linux ARM64](#)

Anschließend wird man aufgefordert, für das Arduino Projekt zu spenden. Es ist gut, wenn man das großartige Projekt auf diese Weise unterstützt. Das Programm kann aber problemlos auch ohne Spenden heruntergeladen werden. Das heruntergeladene Programm, hier „arduino-1.8.13-windows.exe“ findet man in dem „Downloads,-Ordner seines PC. Die Position des Ordners hängt vom verwendeten Internet Browser und dessen Einstellungen ab.

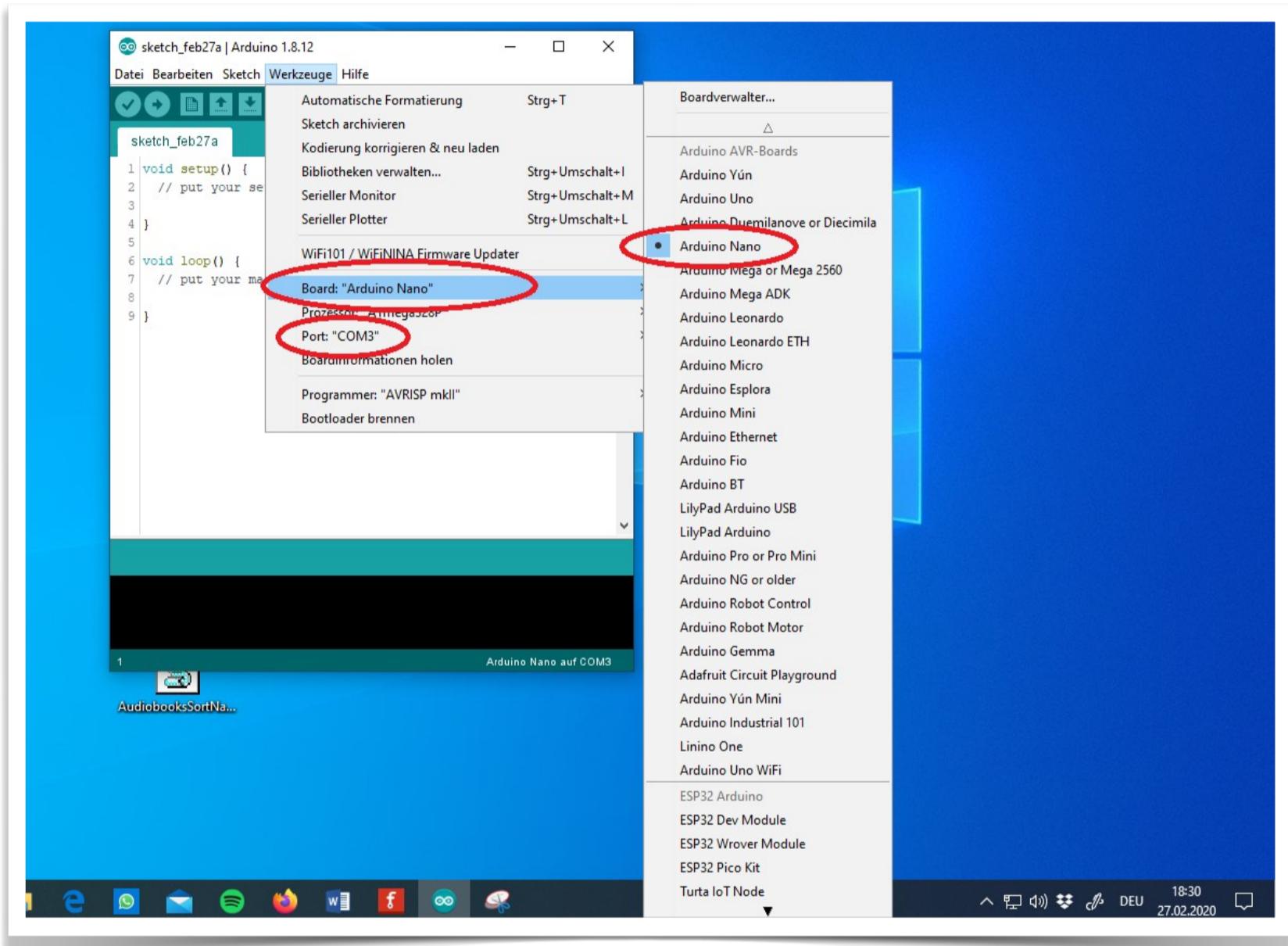
Zur Installation wird das Programm mit einem Doppelklick gestartet. Anschließend befolgt man die Anweisungen auf dem Bildschirm. Zur Installation der IDE findet man bei Bedarf im Internet ausführliche Anleitungen, zum Beispiel:

<https://draeger-it.blog/arduino-ide-installieren/?cn-reloaded=1&cn-reloaded=1>

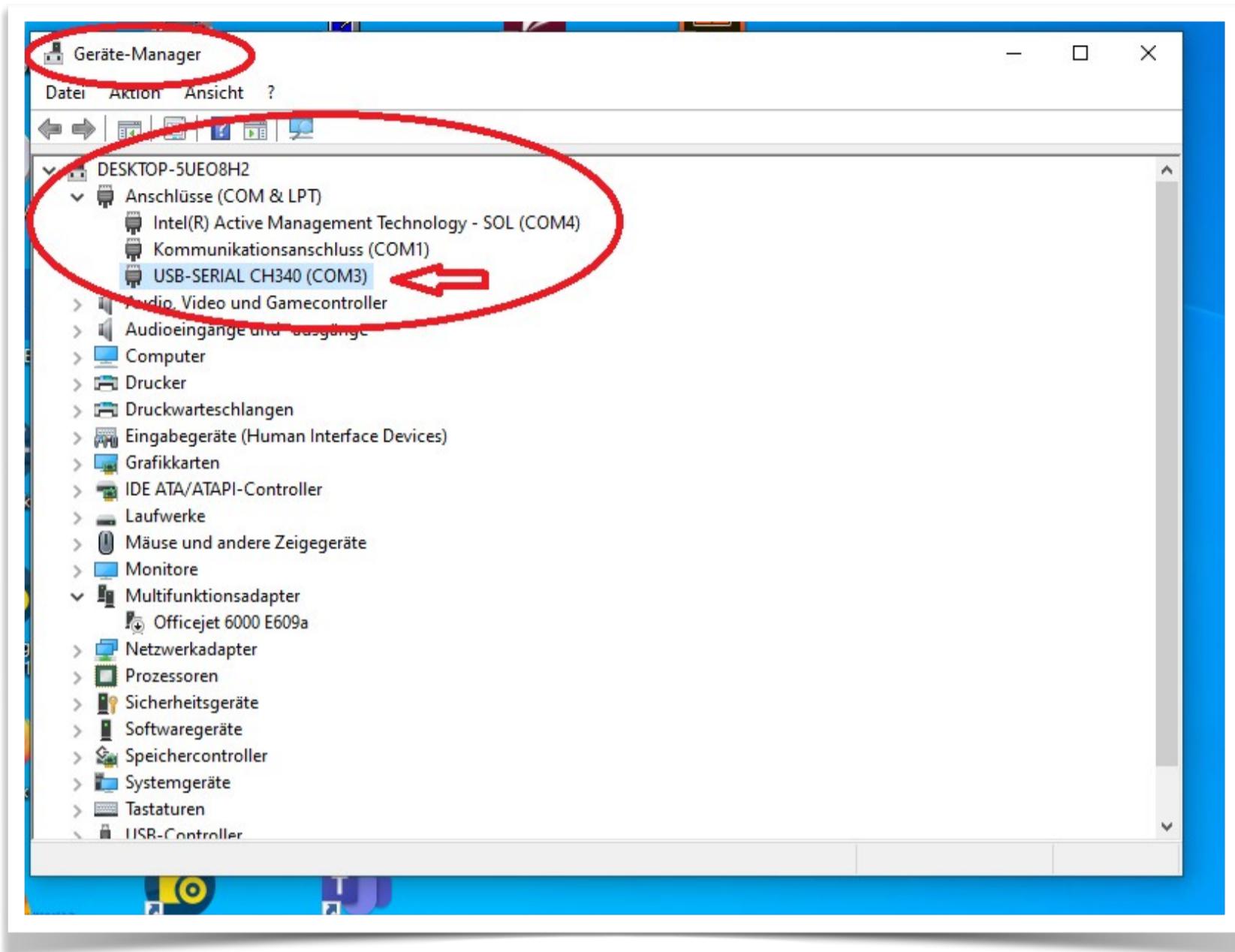
Bei der Installation wird auf dem Desktop eine ICON angelegt, über das die Entwicklungsumgebung gestartet werden kann.



Nun den Arduino an einen freien USB-Port anschließen. WINDOWS installiert nun bei erstmaliger Verbindung USB/Geräte-Treiber für den Arduino. Bei fehlerfreier Installation sollten COM-Port und Arduino, wie im folgenden Bild dargestellt, aufrufbar sein. Die Nummer des COM-Ports ist abhängt von der Belegung der Schnittstellen des PC mit anderen Geräten.



Falls das nicht der Fall ist, sollte zunächst überprüft werden, ob der Arduino erkannt wird und an welchem COM-Port er angeschlossen wurde. Dazu im WINDOWS Geräte-Manager, zu erreichen über die WINDOWS Starttaste mit rechter Maustaste anklicken, unter „Anschlüsse(COM&LPT)“ prüfen ob der Arduino dort aufgelistet ist.



Im Falle von Original-Ardudos erscheint der Name z.B. NANO. Clones werden hingegen mit dem verbauten Chip aufgeführt, hier der CH340. Das liegt daran, dass dort statt der FTDI-Chips die deutlich preiswerteren CH340G USB 2 Serial Chips verbaut werden.

Hilfe, mein Arduino wird nicht erkannt. → siehe Abschnitt [Fehlerbehebung](#) im Wiki.

Tritt beim Kompilieren ein Fehler auf, hilfen vielleicht [FAQ](#) im Wiki weiter.

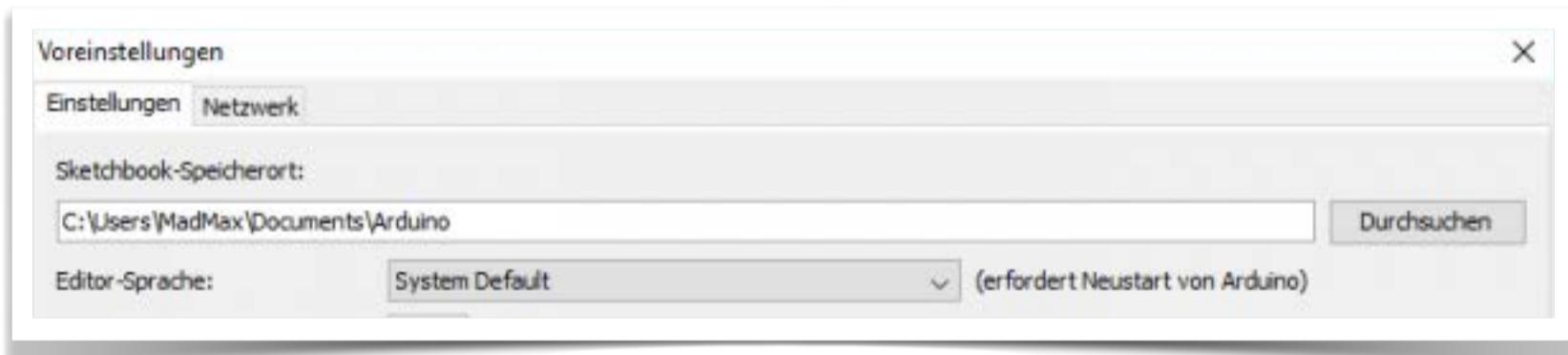
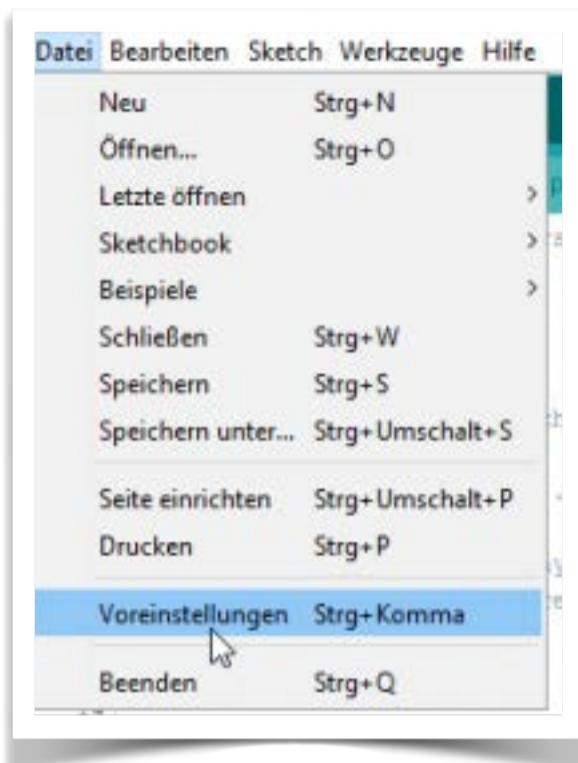
Für Einsteiger ist möglicherweise ein kleiner zusätzlicher Test interessant. Mehr dazu am Ende der Anleitung unter [Test](#) im Wiki

Überprüfung Sketch-Ordner:

Damit alle Funktionen der MobaLedLib aufgerufen werden können, muss der Pfad zum Sketch-Ordner auf dem Standard-Pfad stehen.

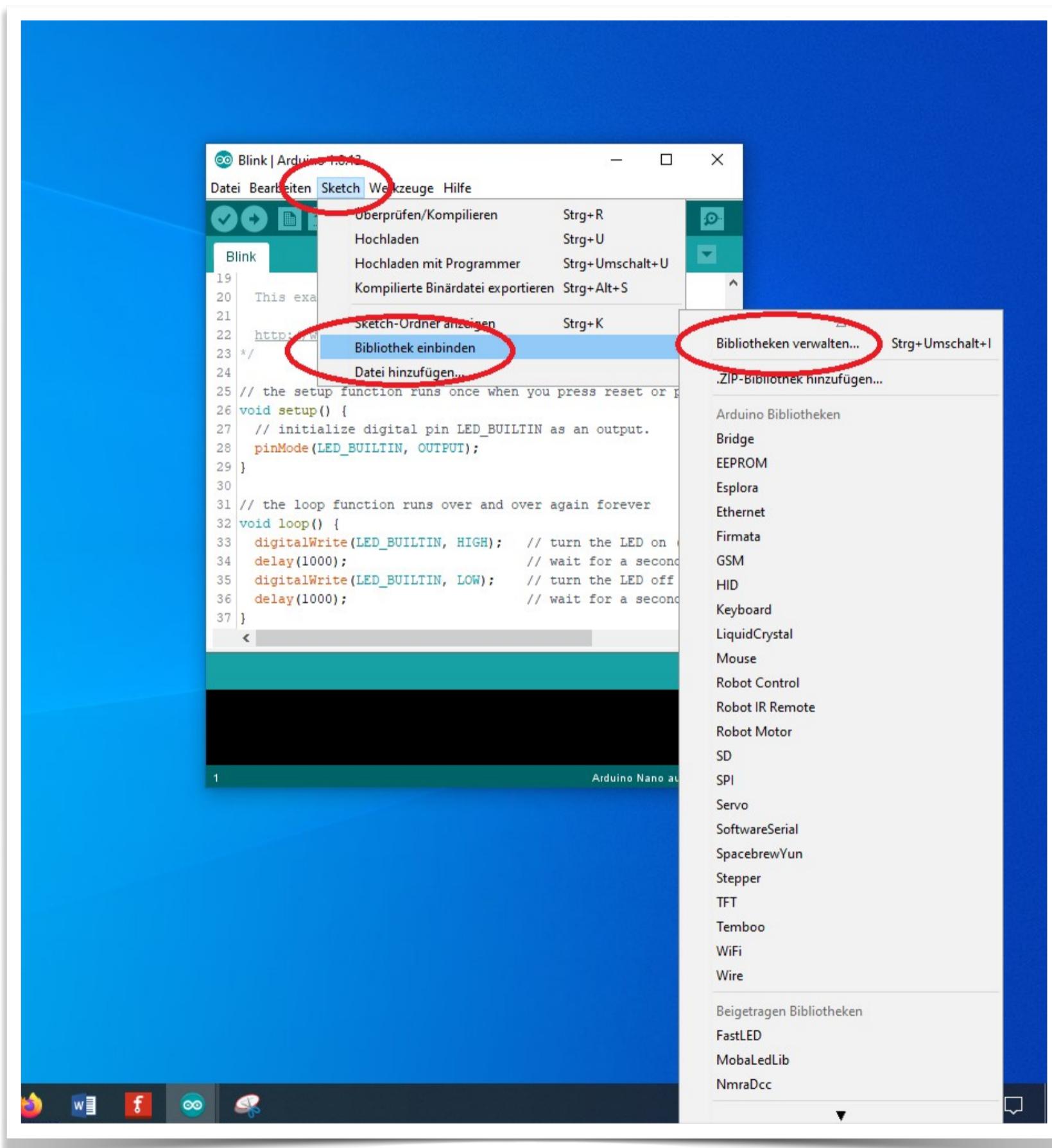
C:\Users**BENUTZERNAME**\Documents\Arduino

Dies kann über das Menü innerhalb der Arduino IDE gemacht werden.

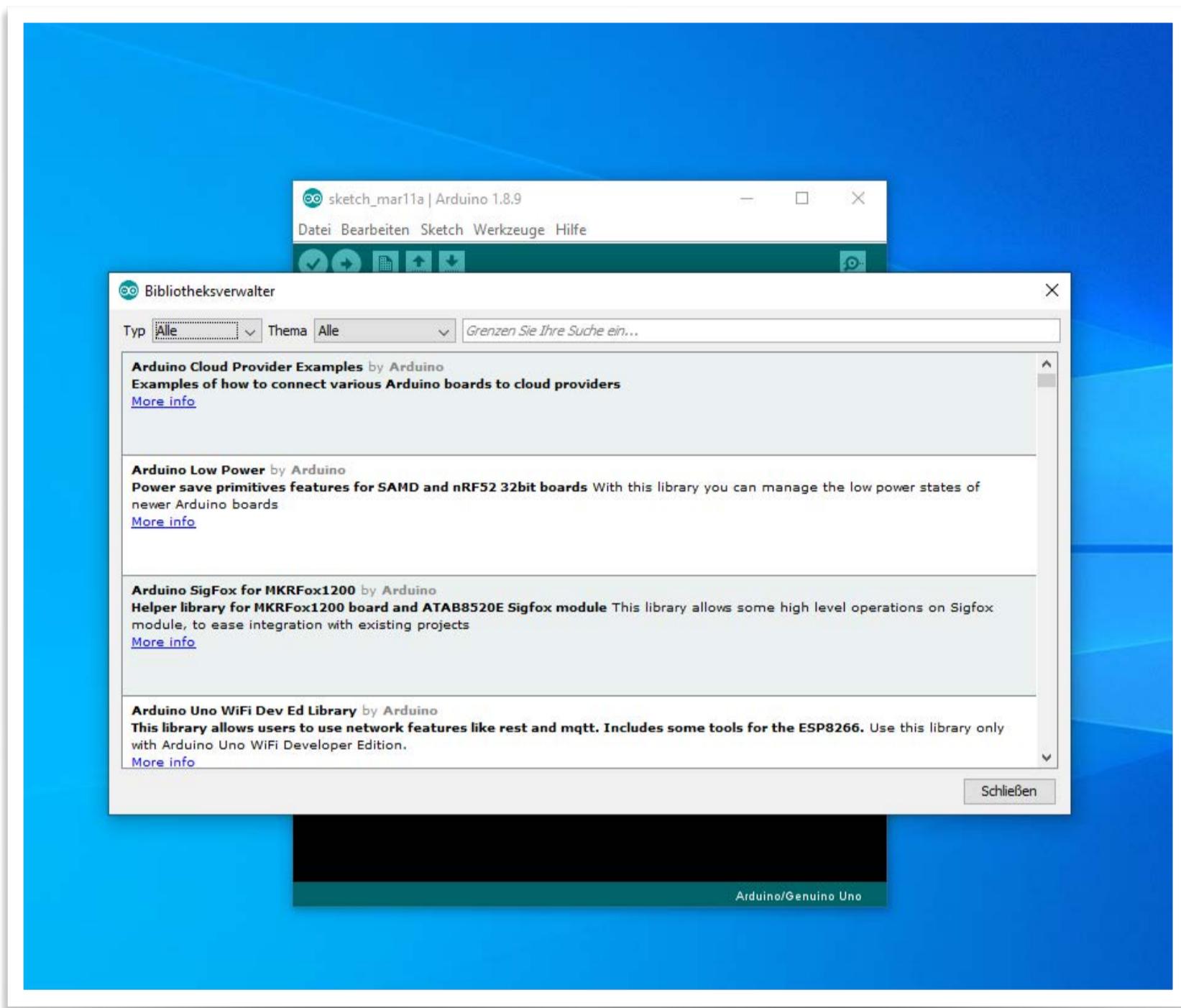


2. MobaLedLib (MLL) einrichten

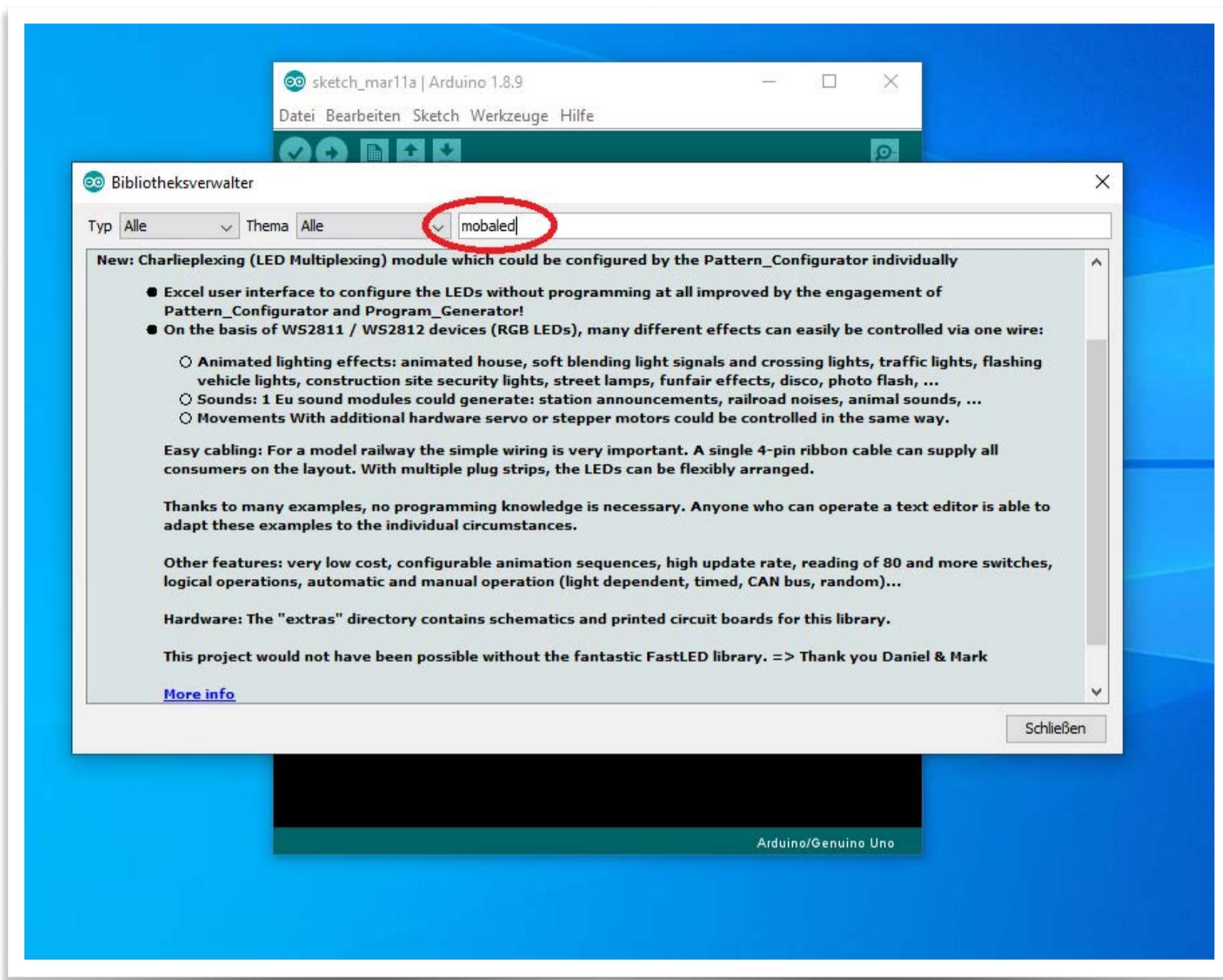
Zunächst muss die Bibliotheksverwaltung der IDE über Sketch → Bibliothek einbinden
→ Bibliothek verwalten geöffnet werden.



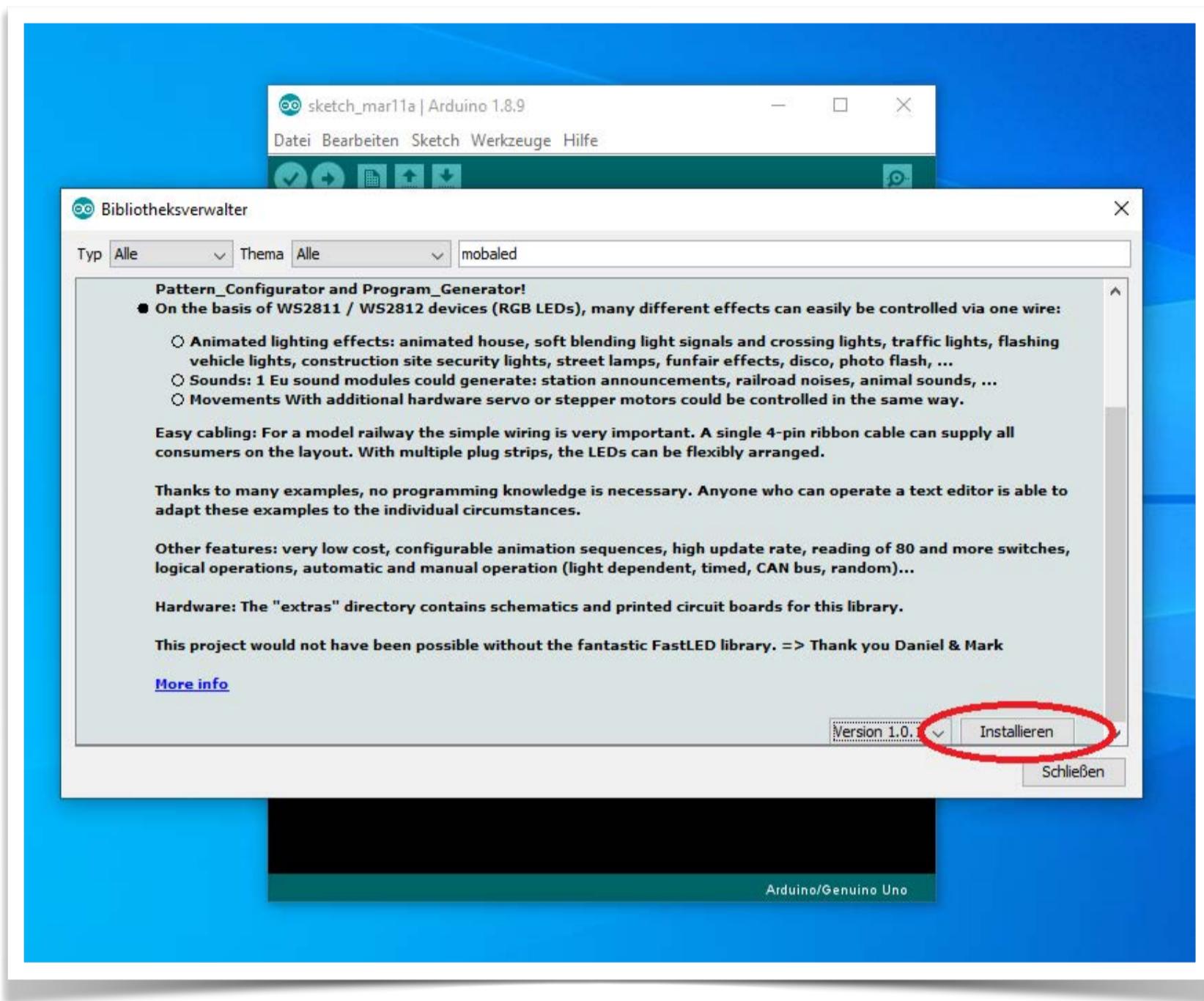
Es öffnet sich der Bibliotheksverwalter.



Nun in das Suchfeld rechts oben „mobaledlib“ eingeben.

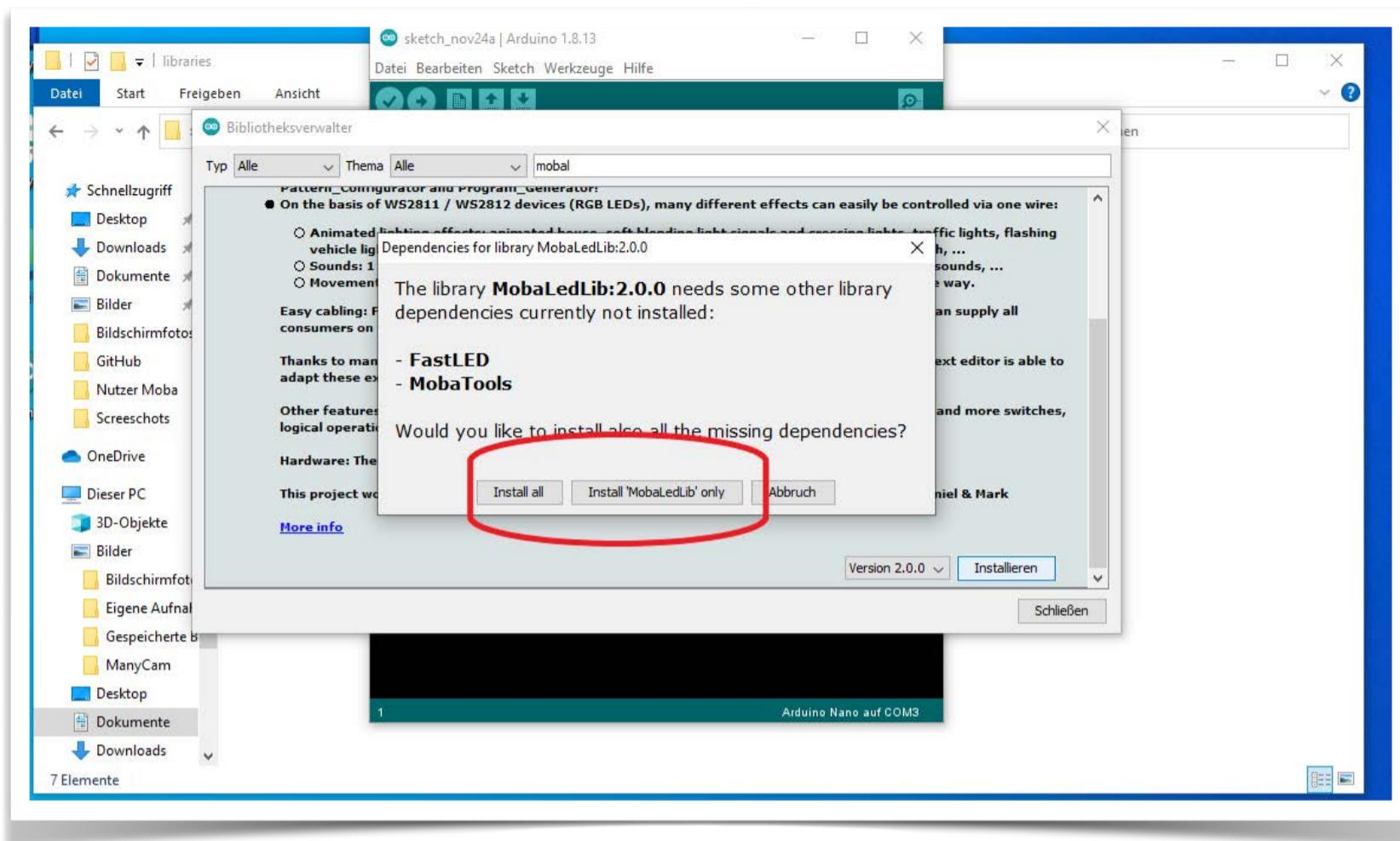


In den gefundenen Eintrag klicken, dann erscheint der Installieren-Knopf



Im Bedarfsfall kann über diesen Menüpunkt auch eine andere Version oder ein Update installiert werden.

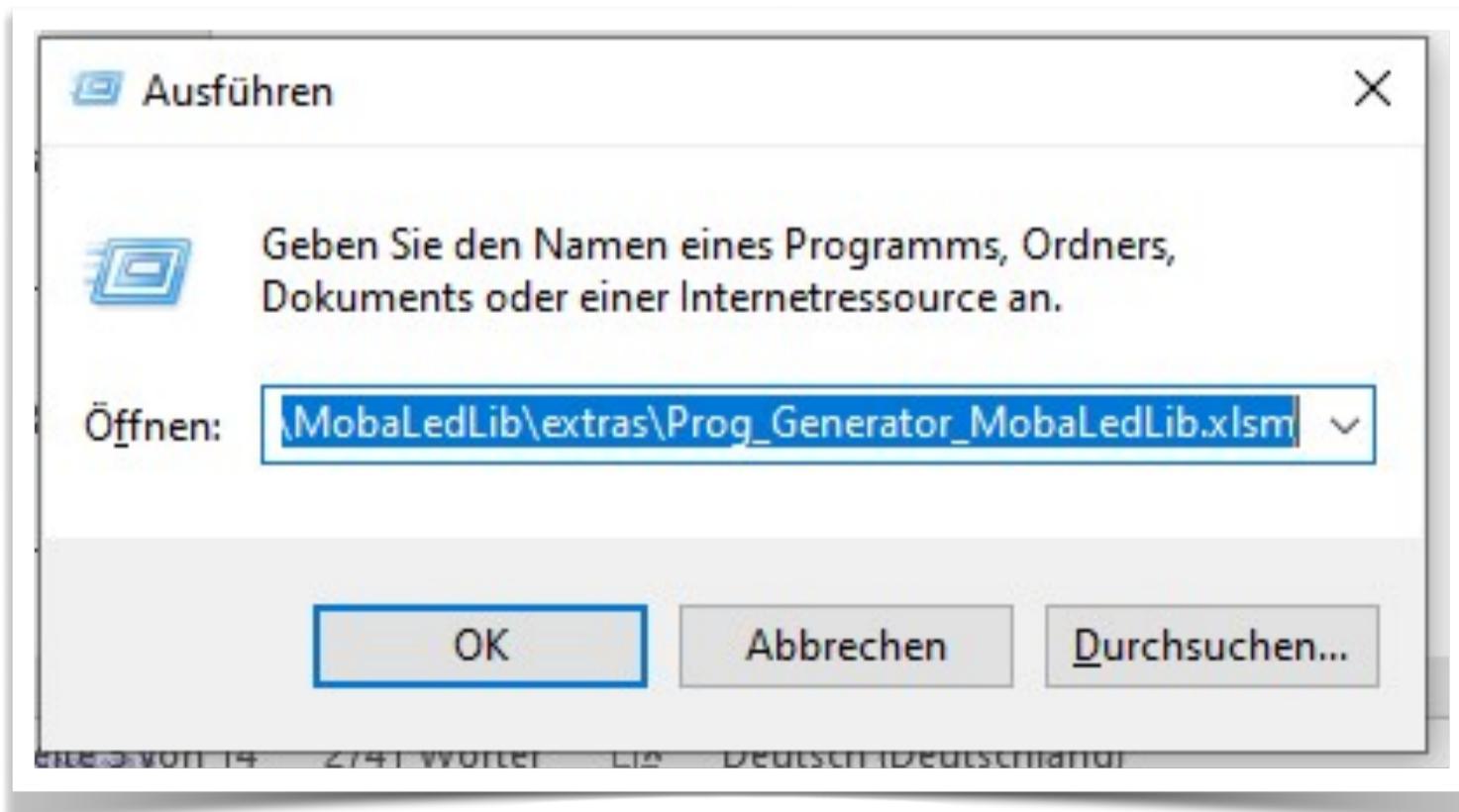
Mit einem Klick auf den Knopf wird die Installation angestoßen. Im folgenden Fenster kann man noch entscheiden ob alle möglicherweise fehlenden Libraries zum Betrieb der MobaLedLib ebenfalls automatisch installiert werden sollen.



Jetzt kann endlich das MobaLedLib-Excel Programm geöffnet werden. Dazu auf der Tastatur die WINDOWS und die „R“ Taste gleichzeitig drücken. In den erscheinenden „Ausführen“ Dialog die folgende Zeile kopieren:

```
%USERPROFILE%
\Documents\Arduino\libraries\MobaLedLib\extras\Prog_Generator_MobaLedLib.xlsxm
```

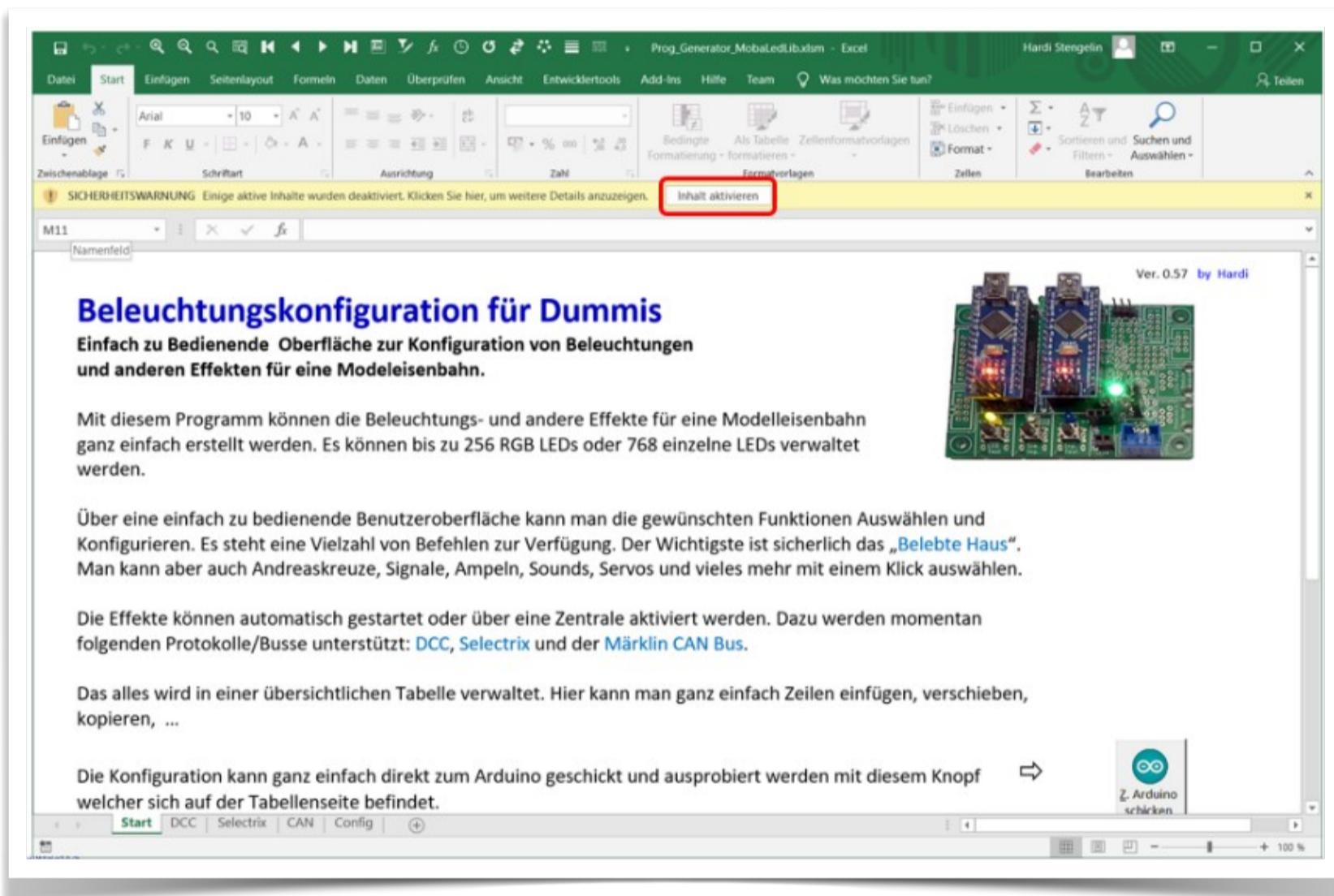
(Achtung: Die Groß- und Kleinschreibung muss exakt stimmen.)



Nun sollte sich endlich die aktuelle Version des Programm Generators der MobaLedLib öffnen.

Sollte dies nicht der Fall sein, hat der Virenschanner die MobaLedLib blockiert.

→ siehe Abschnitt [Fehlerbehebung](#) im Wiki.

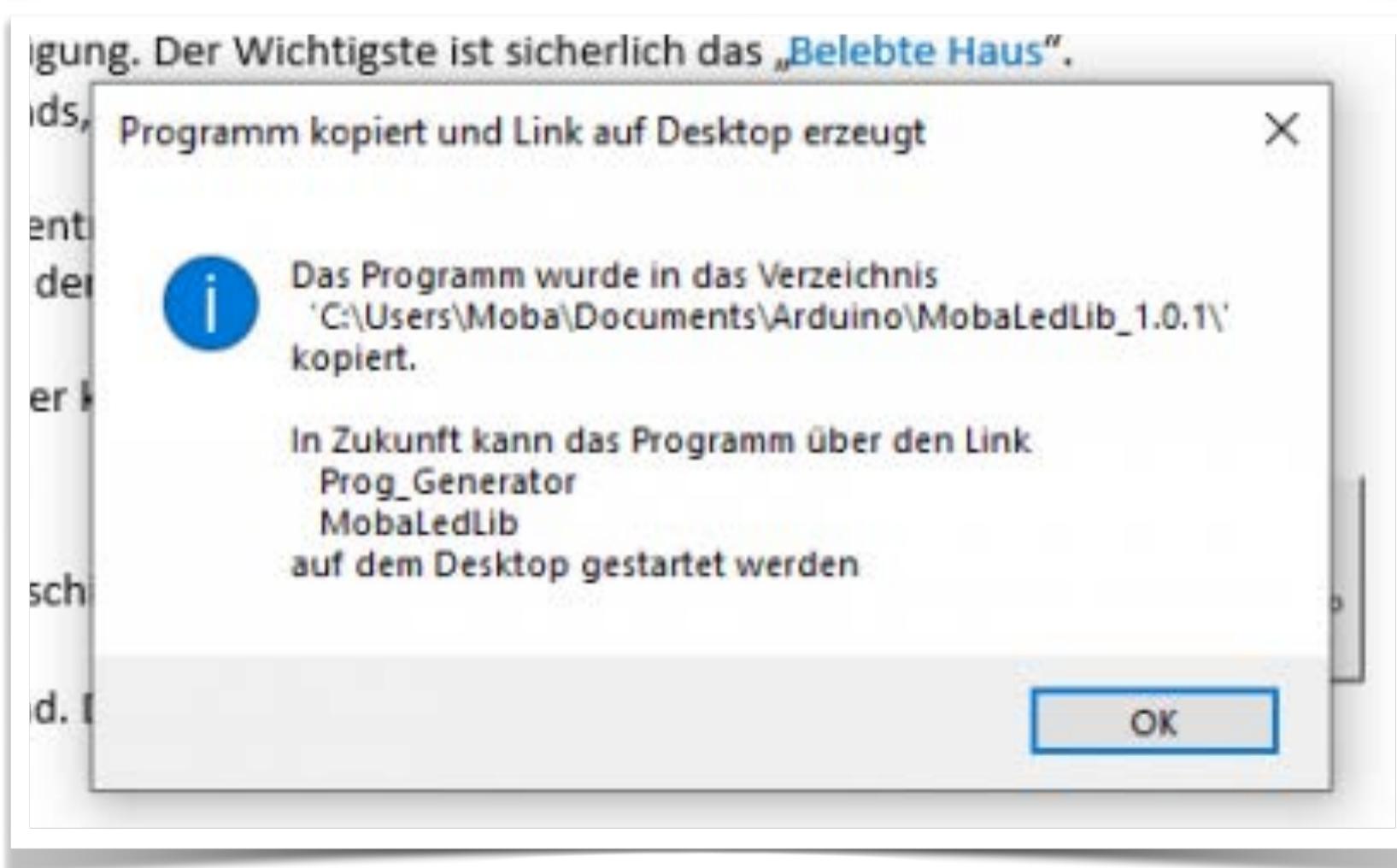
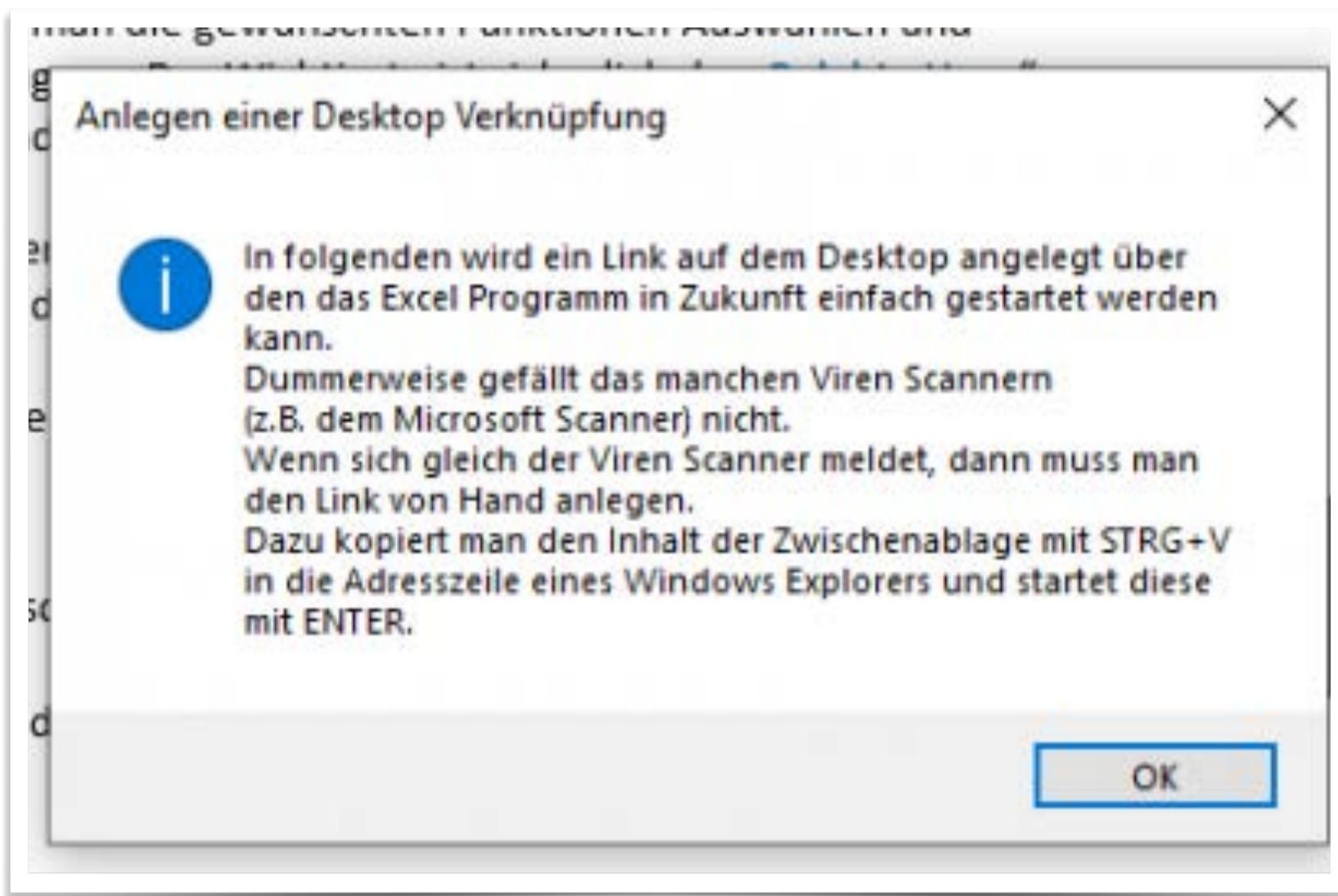


Da es sich um ein Excel Programm mit eingebetteten Makros handelt, wird beim ersten Start eine Sicherheitswarnung angezeigt. Mit einem Klick auf „Inhalte aktivieren“ werden die Makros ausführbar gemacht.

Danach wird das Programm automatisch in das Verzeichnis

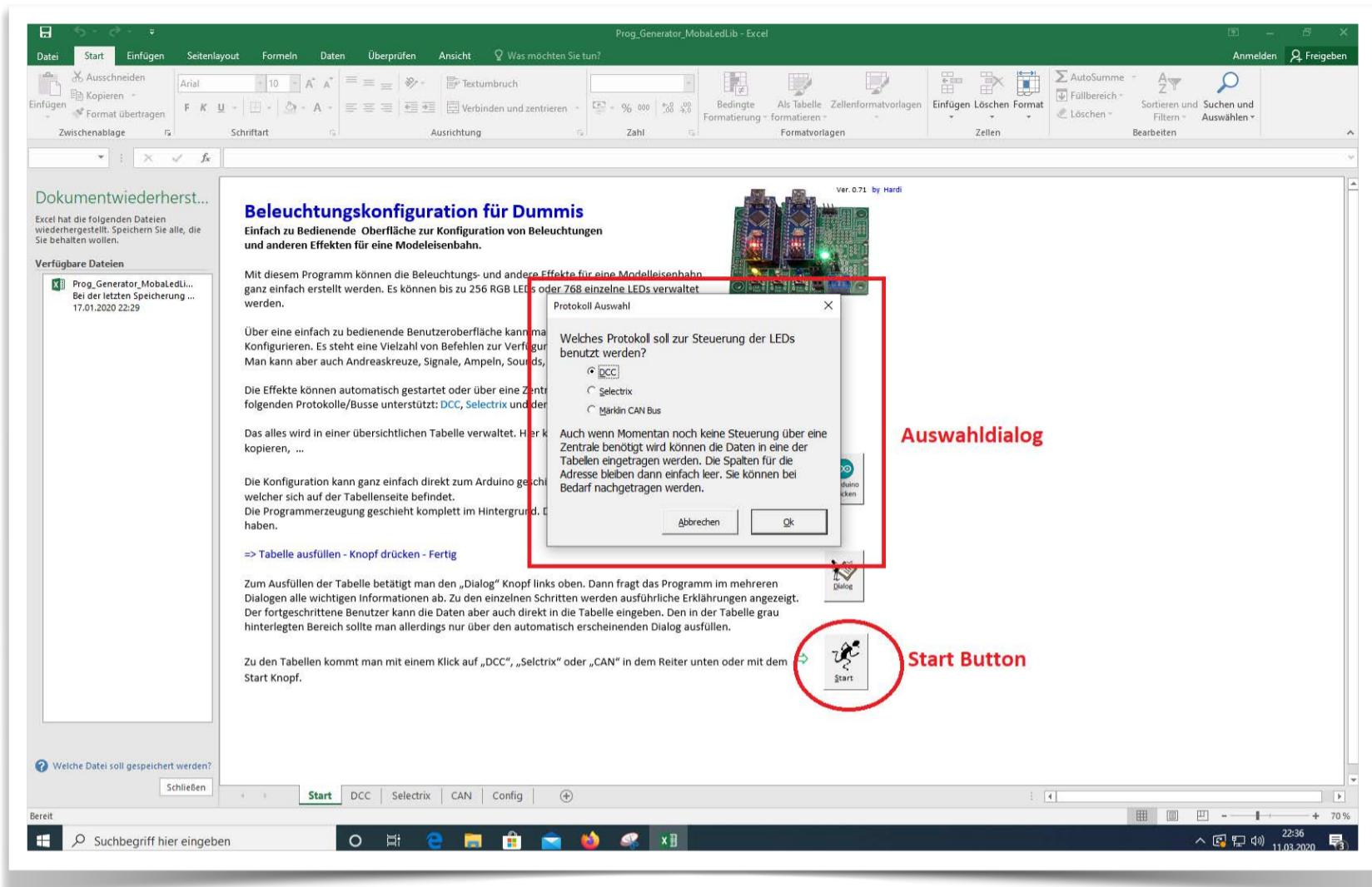
%USERPROFILE%\Documents\Arduino\MobaLedLib_x.y.z\LEDs_AutoProg

kopiert und ein Icon auf dem Desktop angelegt (x.y.z entspricht der Versionsnummer). Darüber kann der Prog-Generator künftig gestartet werden.

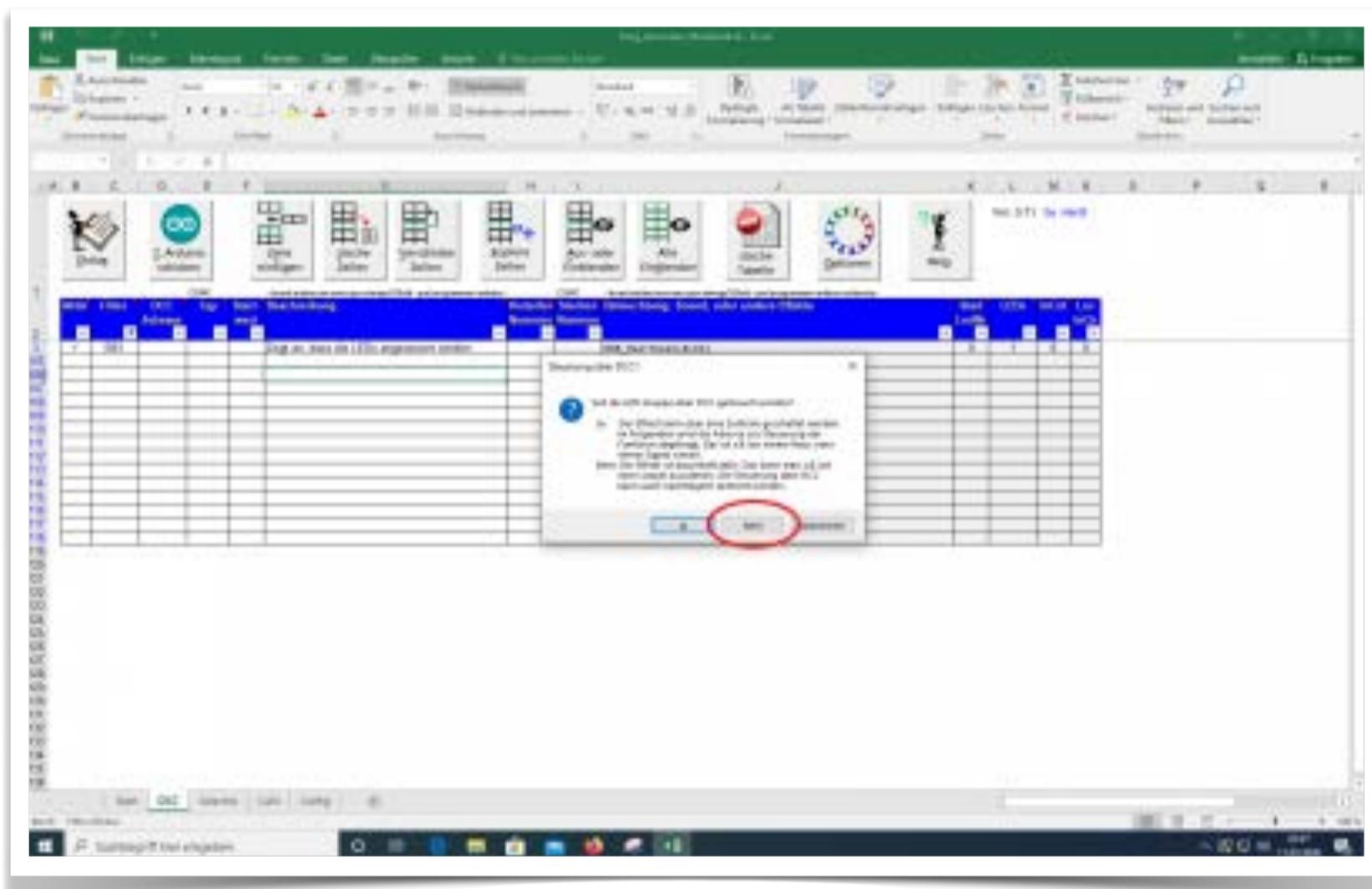
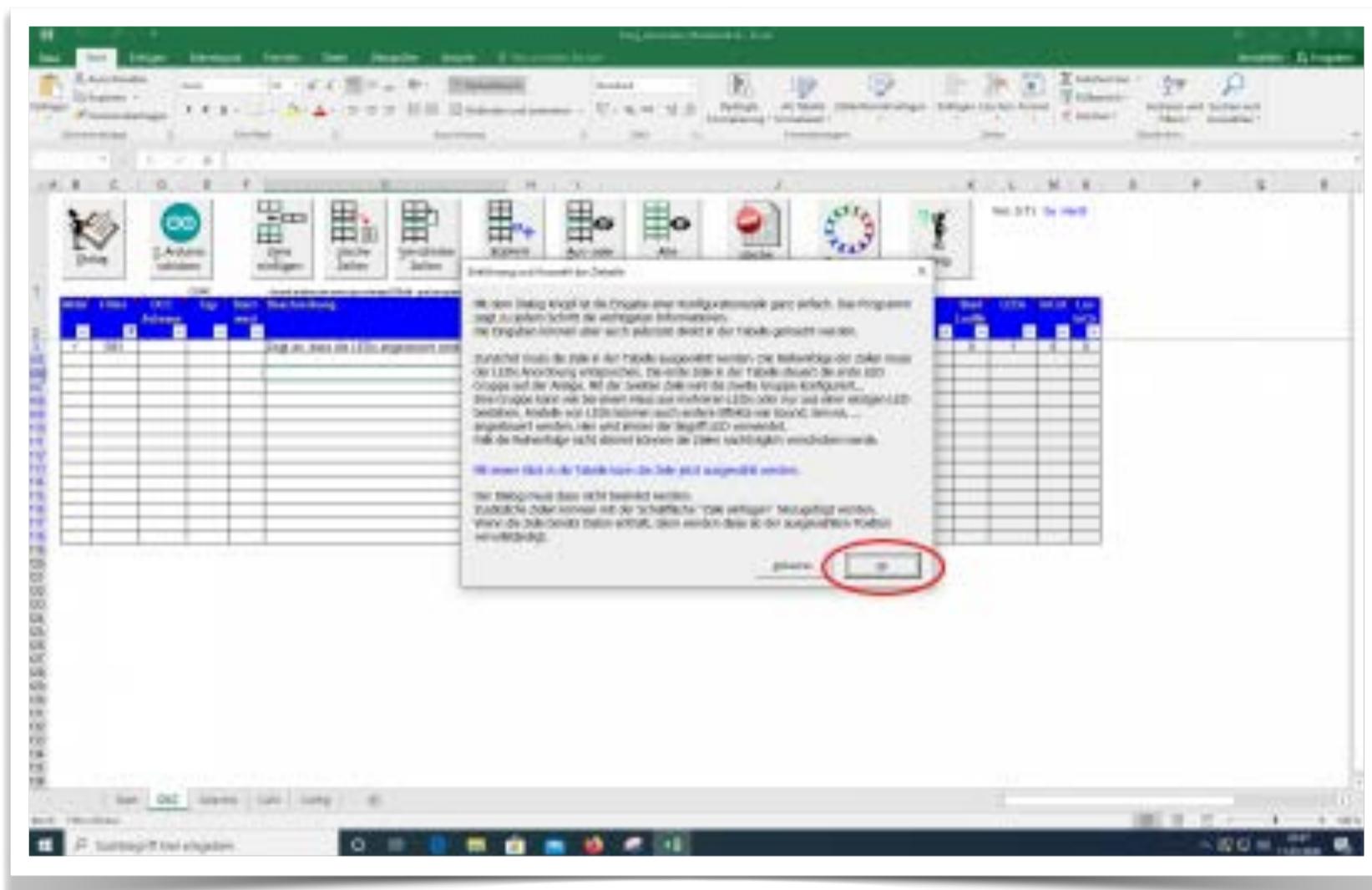


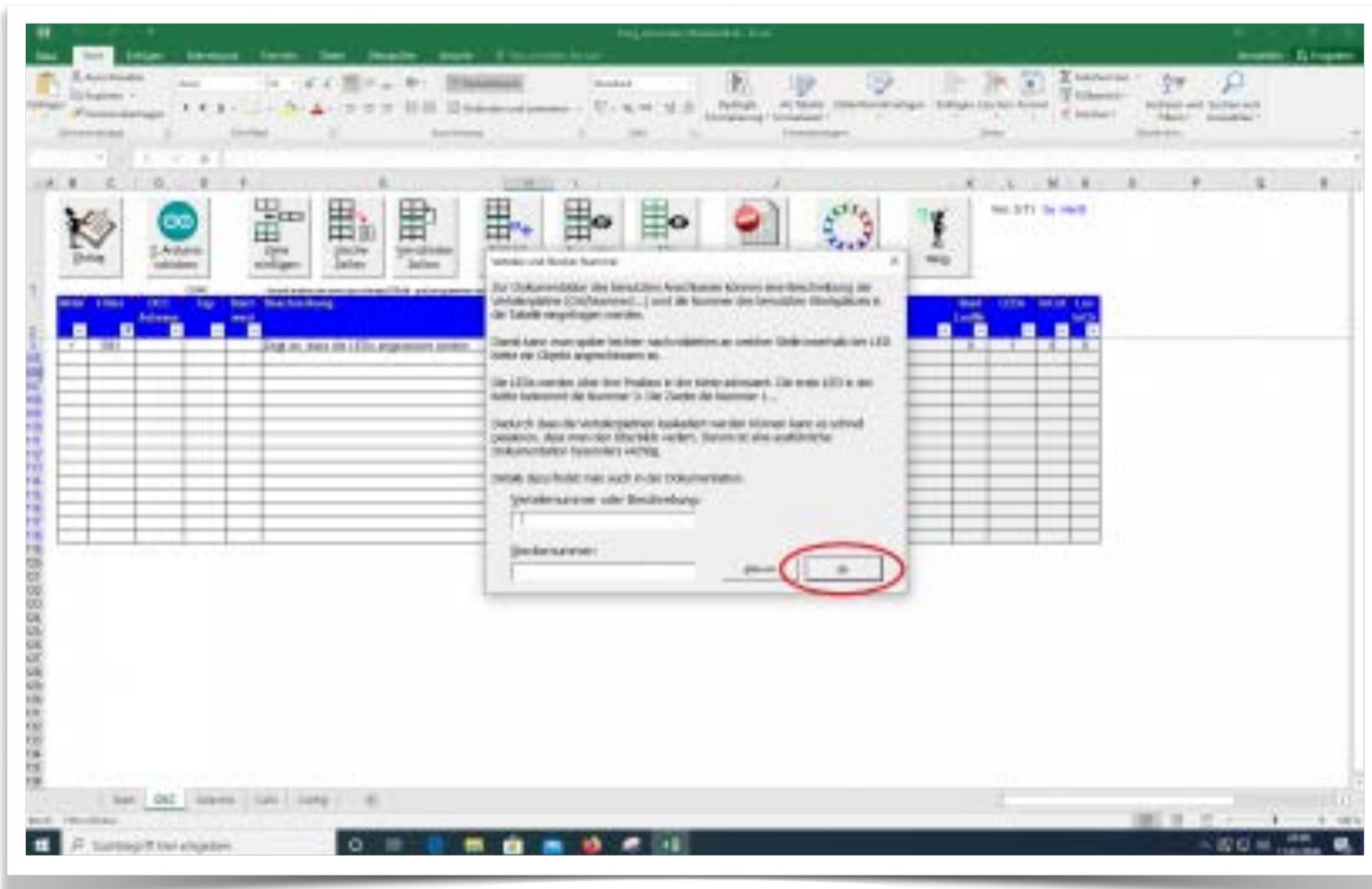
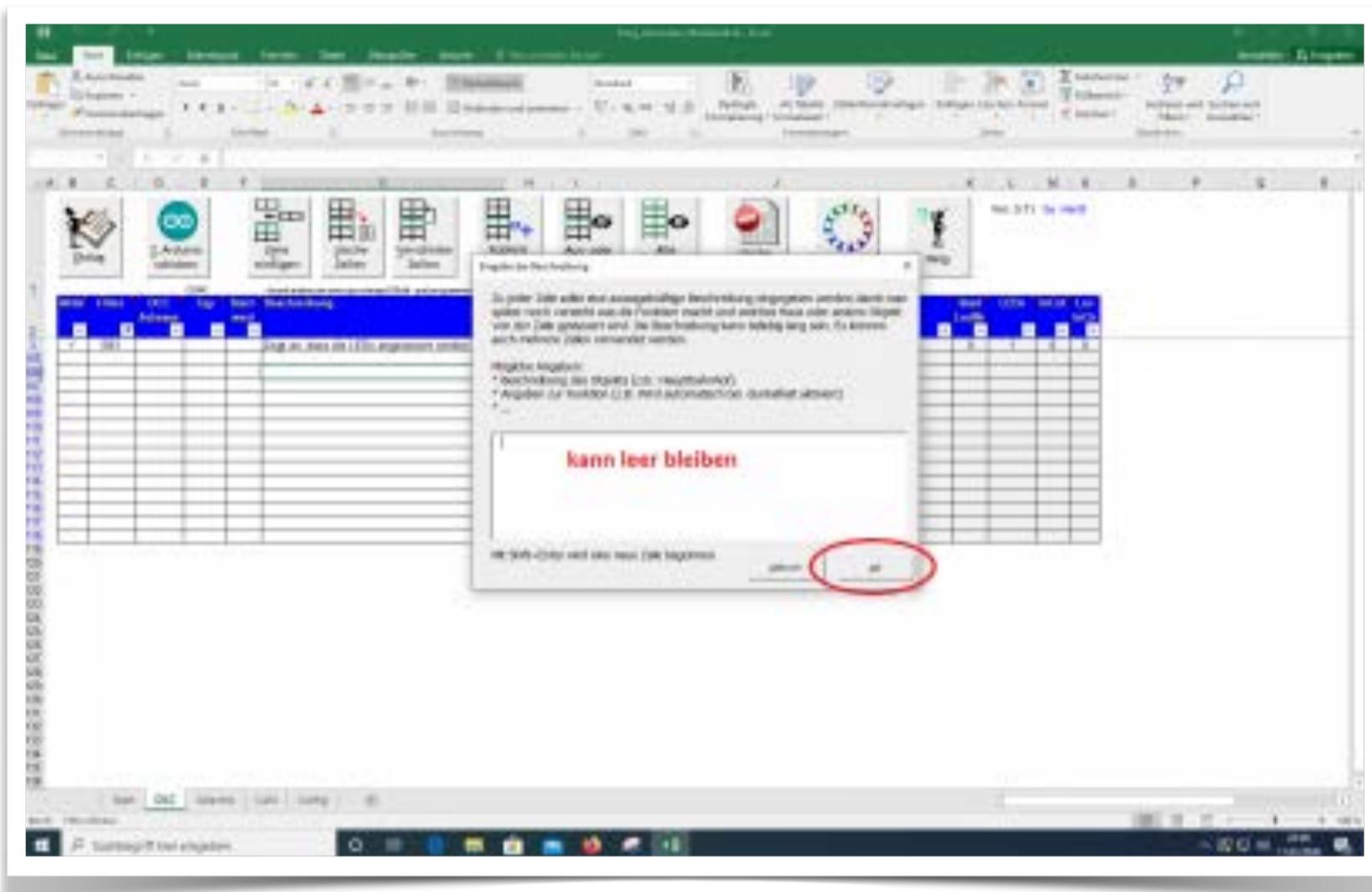
Ein Klick auf OK führt zu einem ersten Auswahldialog.

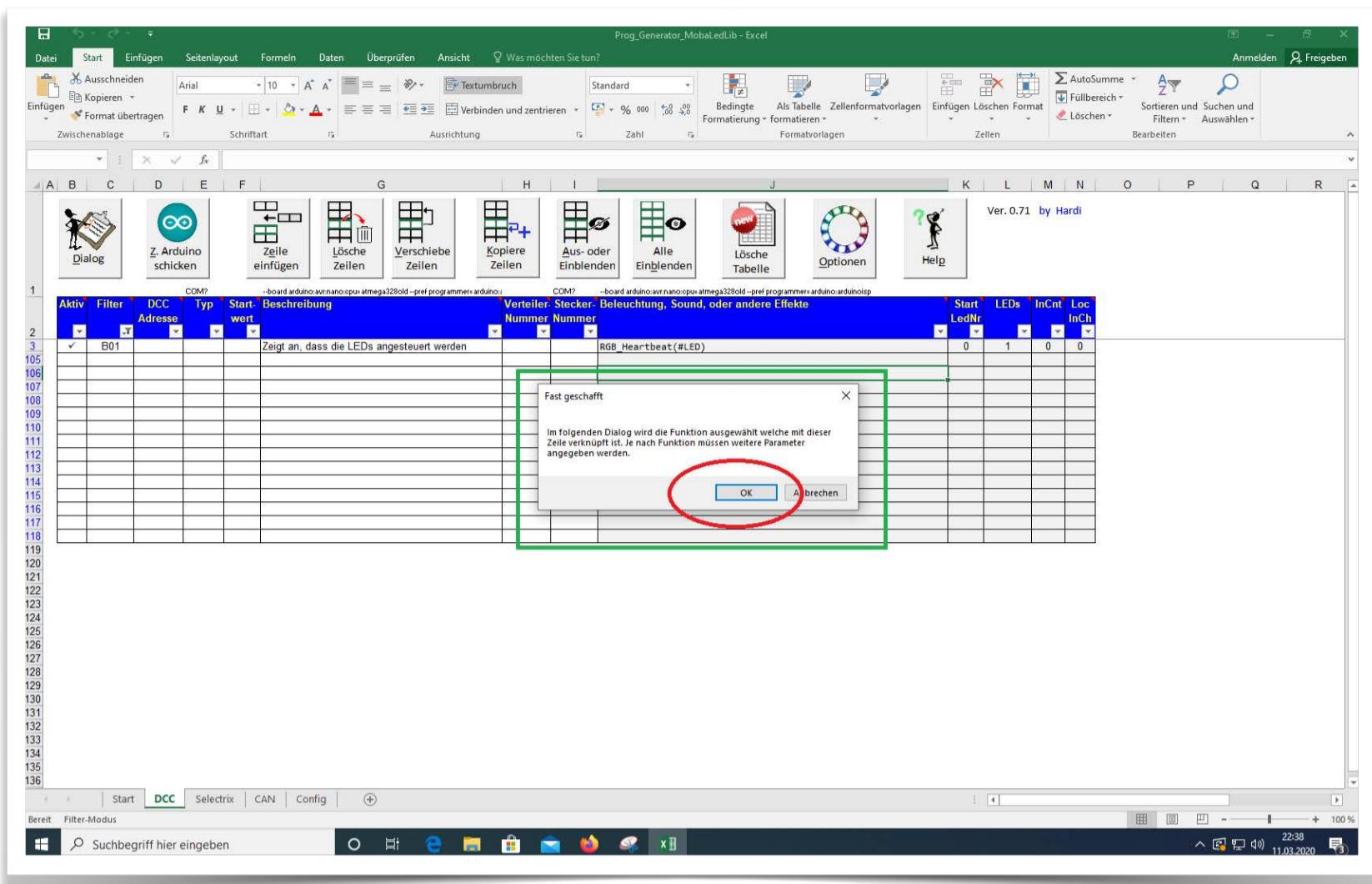
Für erste Versuche ist es zunächst unerheblich welche der drei angebotenen Möglichkeiten (DCC-, Selektix- oder CAN-Steuerung) ausgewählt wird.



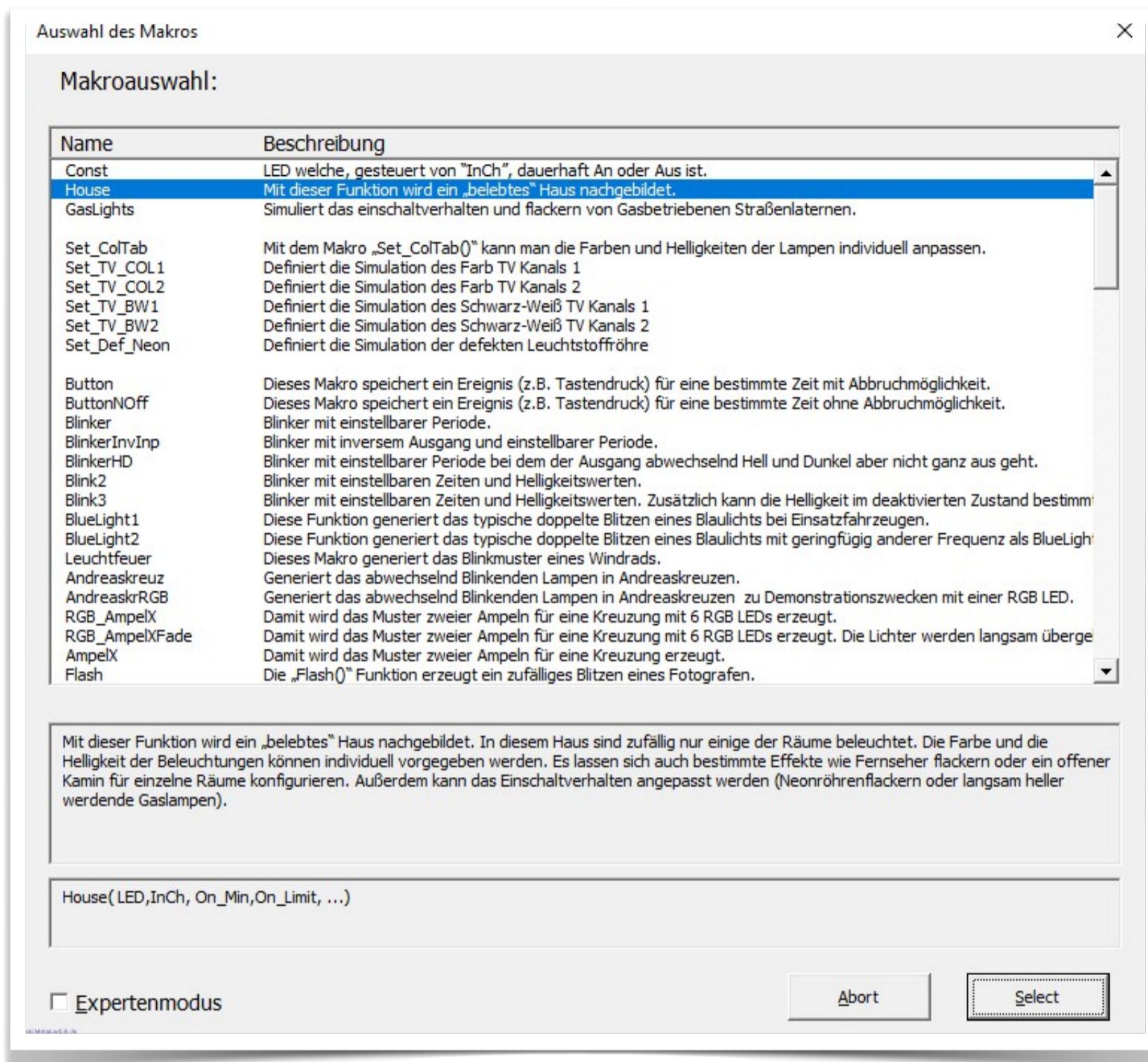
Nach Anklicken des Start-Knopfes werden weitere Dialoge geöffnet. Eingaben sind für den ersten Einstieg nicht notwendig. Es genügt, die rot markierten Optionen auszuwählen. Erläuterungen zu den Dialogen sind in der ausführlichen Anleitung zum Prog_Gen im Wiki zu finden.





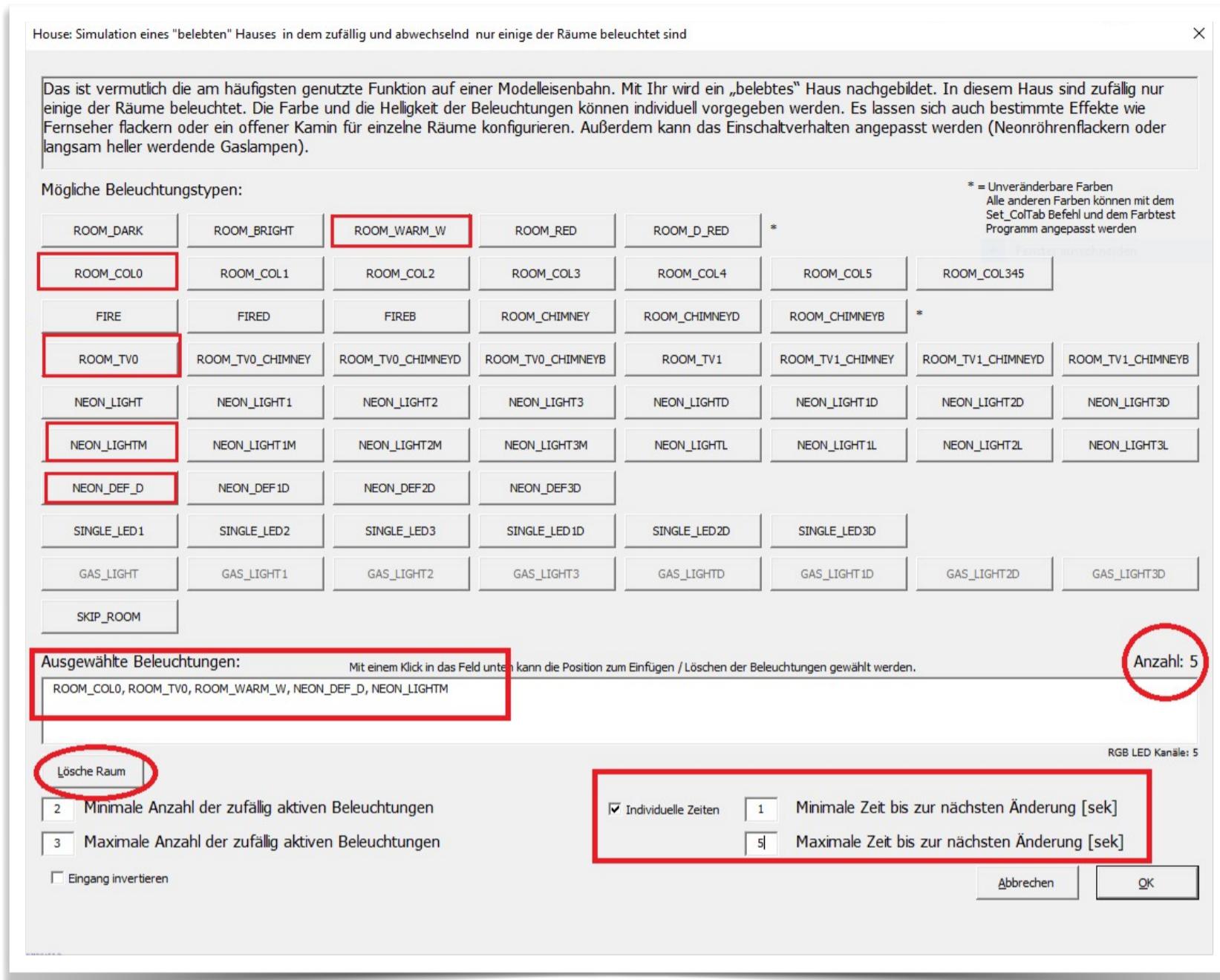


Die letzte Schaltfläche führt uns schließlich zur Makroauswahl.

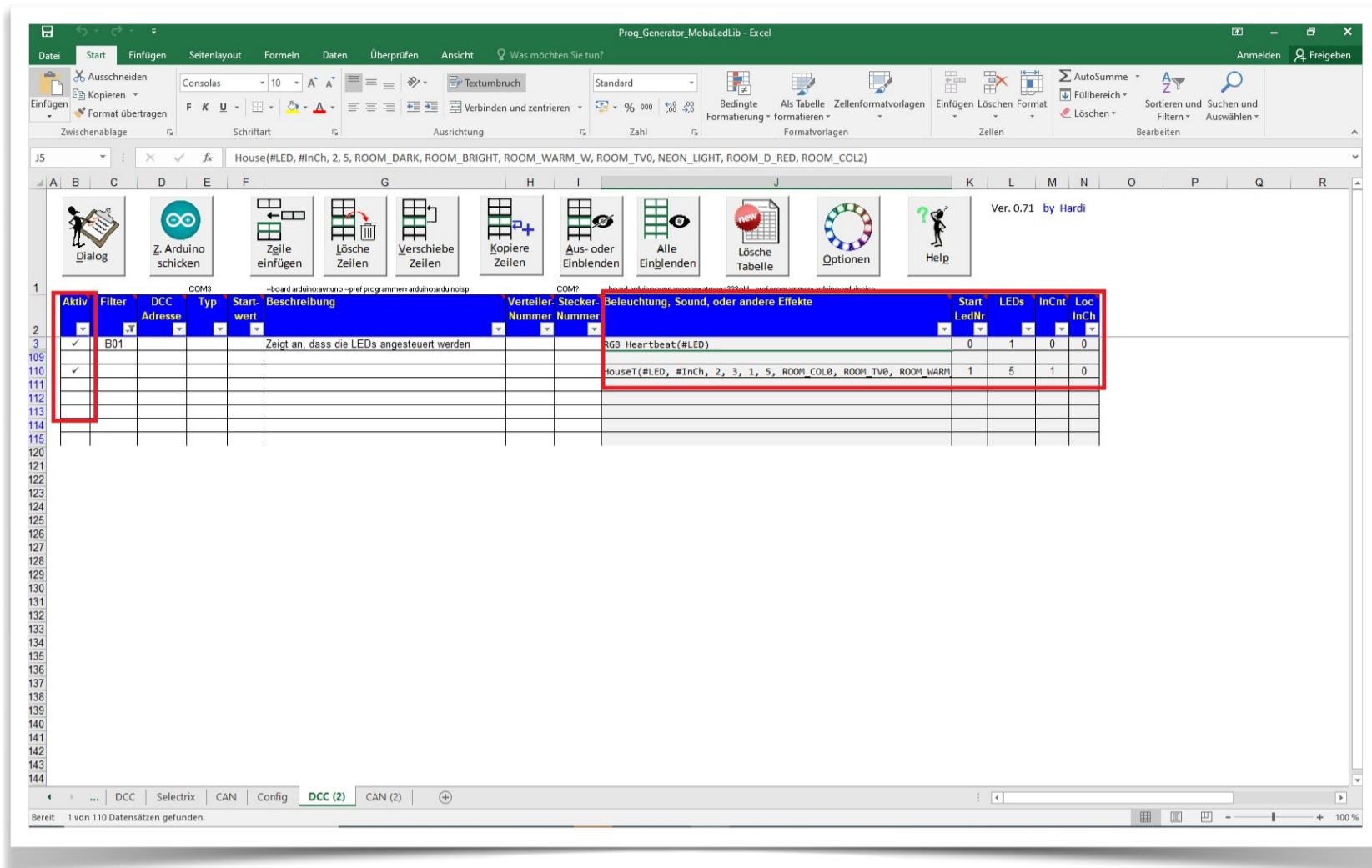


Auswahl eines Beispiels (belebtes Haus)

Für unseren ersten Test wählen wir mit dem Select-Knopf das „House“ aus und erhalten die folgende Seite zur weiteren Auswahl der Funktionen:



Über das Tastenfeld „mögliche Beleuchtungstypen“ können unterschiedliche Beleuchtungen für die Räume eines Hauses ausgewählt werden. Wenn in einem Gebäude fünf Räume beleuchtet sind, müssen fünf Beleuchtungen ausgewählt werden. Auswahl und Anzahl werden angezeigt. Über die Taste „lösche Raum“ lassen sich Räume löschen um eine andere Beleuchtung auszuwählen. Durch Anklicken der Option „individuelle Zeiten“ lassen sich die Zeiten für den Beleuchtungswechsel den eigenen Bedürfnissen entsprechend anpassen. Für Testzwecke bietet es sich an, die „Maximale Zeit bis zur nächsten Änderung“ auf 5 [Sec] zu setzen. Nun über „OK“ die Auswahl bestätigen. Damit werden die Werte in die Excel-Tabelle eingefügt und die Seite sollte wie folgt aussehen:

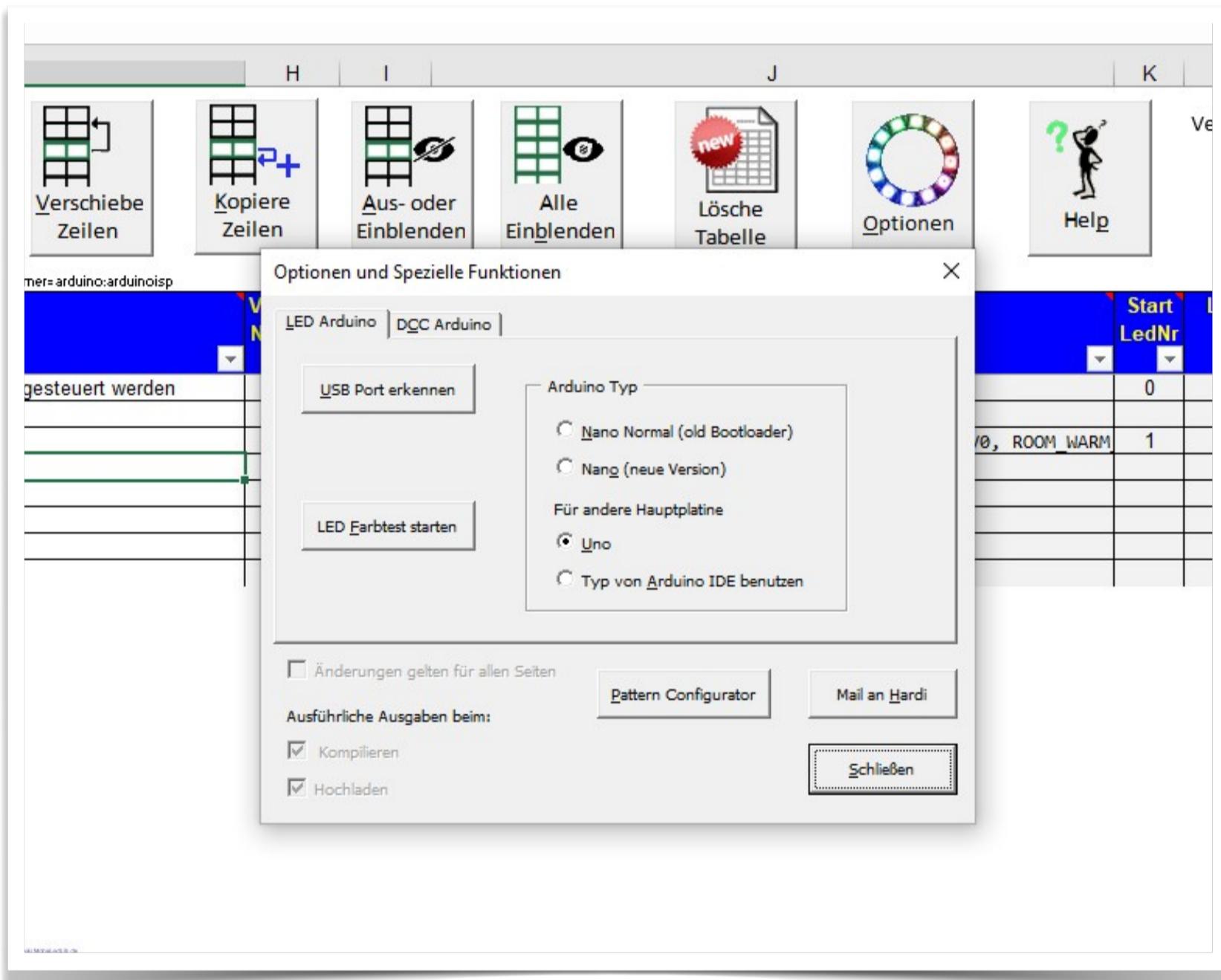


Zwei Zeilen sind aktiv geschaltet. Das ist an dem per Mausklick gesetzten Häkchen zu erkennen. Nicht-aktive Zeilen werden nicht zum Arduino übertragen. Die Felder in den Spalten Filter, DCC Adresse, Typ und Startwert bleiben frei. Das Feld „Beschreibungen“ kann zur Erläuterung für eigene Zwecke genutzt werden. Der grau hinterlegte Bereich ist automatisch befüllt worden und kann bzw. sollte nicht geändert werden. Über die roten Dreiecke in den Feldern könne zur weiteren Erklärung Tooltips aufgerufen werden.

Die erste Zeile mit dem Beispiel B01 „RGB_Heartbeat(#LED)“ wird standardmäßig gesetzt. Damit wird die erste LED in der Kette genutzt, um zu signalisieren, dass die Übertragung des Programms an den Arduino erfolgreich war und das System „lebt“. Falls die folgenden LEDs dann trotzdem nicht so arbeiten wie erwartet, hat man irgendwo in der Auswahl für die LEDs einen Fehler gemacht oder in der Verdrahtung der LEDs liegt ein Fehler vor.

Im nächsten Schritt wird unsere Auswahl zum ARDUINO geschickt. Beim ersten Sendeversuch erfolgt die Aufforderung, den benutzten COM-Port festzulegen. Einfach den Anweisungen folgen und anschließend den „Z.Arduino schicken“ Knopf drücken.

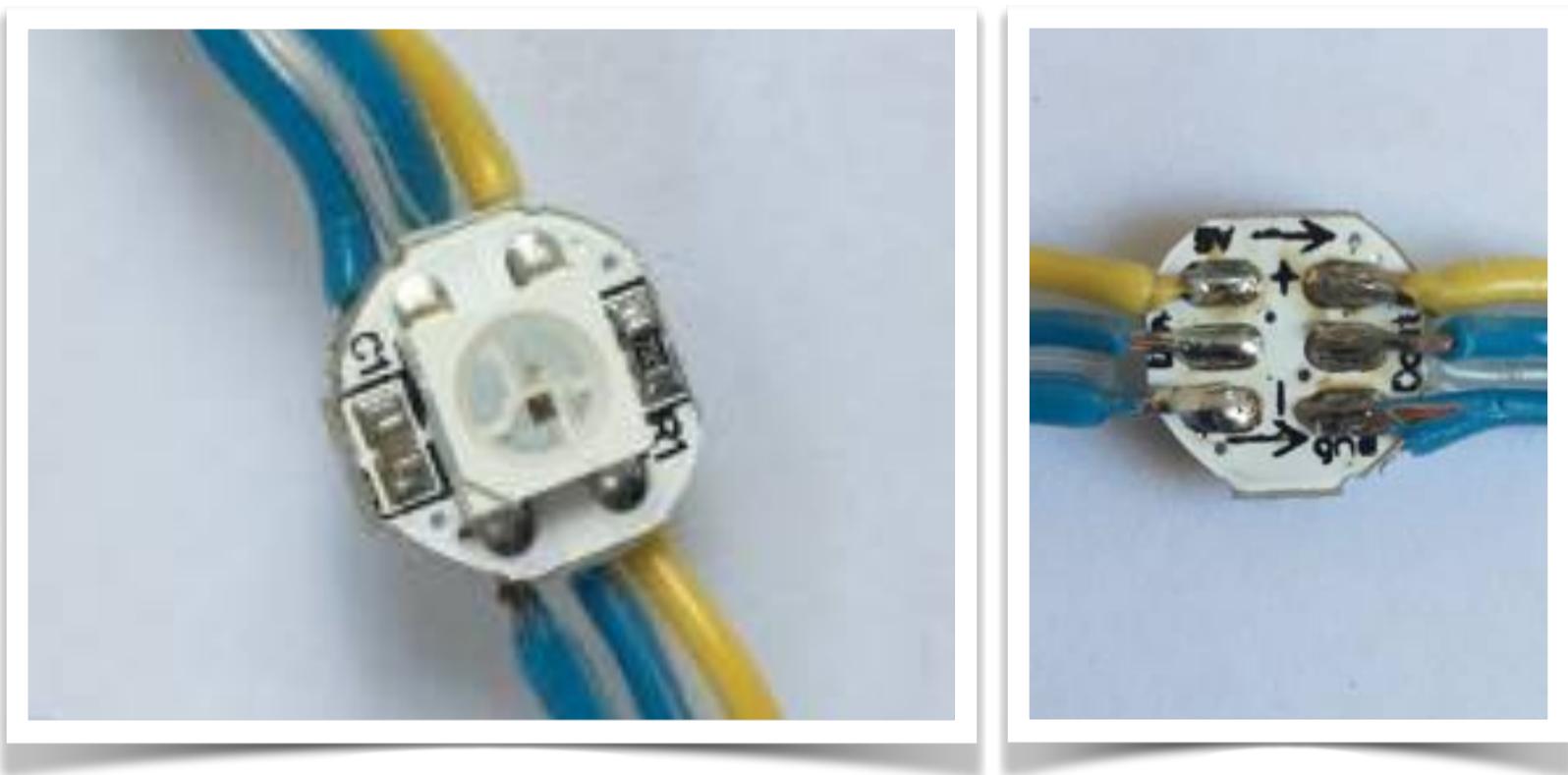
Der Vorgang zum Erkennen des COM-Ports kann auch über „Optionen“ → „USB Port erkennen“ angestoßen werden. Für eine fehlerfreie Übertragung zum Arduino muss im Auswahlmenü unbedingt der tatsächlich genutzte Typ eingetragen werden. Für Arduino Nano Clones i.d.R. „Nano Normal (old Bootloader)“ auswählen. Ist der Arduino Typ nicht aufgeführt, sollte über das Arduino IDE der Typ ausgewählt werden und hier übernommen werden: „Typ von Arduino IDE benutzen“.



Hat bis hierher alles wie gewünscht funktioniert ist jetzt zwar das Programm auf dem Arduino aber es fehlen noch die LEDs.

3. Anschluss der RGB LEDs an einen Arduino

Im Abschnitt [Bezugsquellen](#) sind Händler aufgeführt, die schnell und etwas teurer bzw. „billiger“ und mit längeren Lieferzeiten die Bauteile liefern.



Für den ersten Versuche reicht es, einfach ein paar RGB LEDs zusammen zu löten und direkt mit den Anschlüsse des Arduinos zu verbinden. Vorsichtshalber vorher den Arduino vom Computer trennen. Aber nicht übertreiben. Wenn der Arduino mit einem normalen USB-Netzteil betrieben wird stehen garantierte 300 mA am 5V-Anschluss zur Verfügung. Das entspricht 5 RGB-LEDs mit maximaler Helligkeit auf allen Kanälen. Verbindet man mehr LEDs, reicht die Stromversorgung über den Arduino ggf. nicht mehr aus.

Bitte beachten, dass hierfür unbedingt RGB-LED verwendet werden müssen und keine RGBW-LED. Das W steht hierbei für einen zusätzlichen Weiß-Kanal, der von der Software nicht unterstützt wird. Wird zwischen mehreren RGB-LED eine RGBW-LED in Reihe geschaltet, kann es passieren, dass die Kanalprogrammierung aus dem Prog-Generator nicht mehr die richtigen LED anspricht, weil er nacheinander drei Farb-Kanäle erwartet, wo tatsächlich aber vier Kanäle (Rot, Grün, Blau plus Weiß) sind. RGBW- und RGB-LED sehen äußerlich gleich aus und auch in den Produktbeschreibungen ist nicht immer eindeutig erkennbar, welche Art von LED angeboten wird.

Es werden nur drei Kabel benötigt:

- 5 Volt von der ersten LED zum 5V Pin des Arduino
- GND von der ersten LED zu einem GND Pin des Arduino
- In von der ersten LED zum Pin D6 des Arduino.

Nun den Arduino wieder über das USB Kabel mit dem Computer verbinden. Sind keine Änderungen vorgesehen, reicht auch eine 5 Volt Spannungsquelle aus. Die LEDs sollten nun leuchten, wie oben über das „House“ festgelegt.



Nun können weitere Macros ausprobiert werden, um erste Erfahrungen zu sammeln. Einfach im House Macro andere Einstellungen wählen und erneut zum Arduino schicken. Ebenso können zusätzliche Zeilen eingefügt werden und vorhandene aktiviert bzw. deaktiviert werden. Bitte bedenken, dass die Anzahl der im MACRO ausgewählten LEDs die der tatsächlich angeschlossenen LEDs für ein sinnvolles Ergebnis nicht übersteigen sollte.

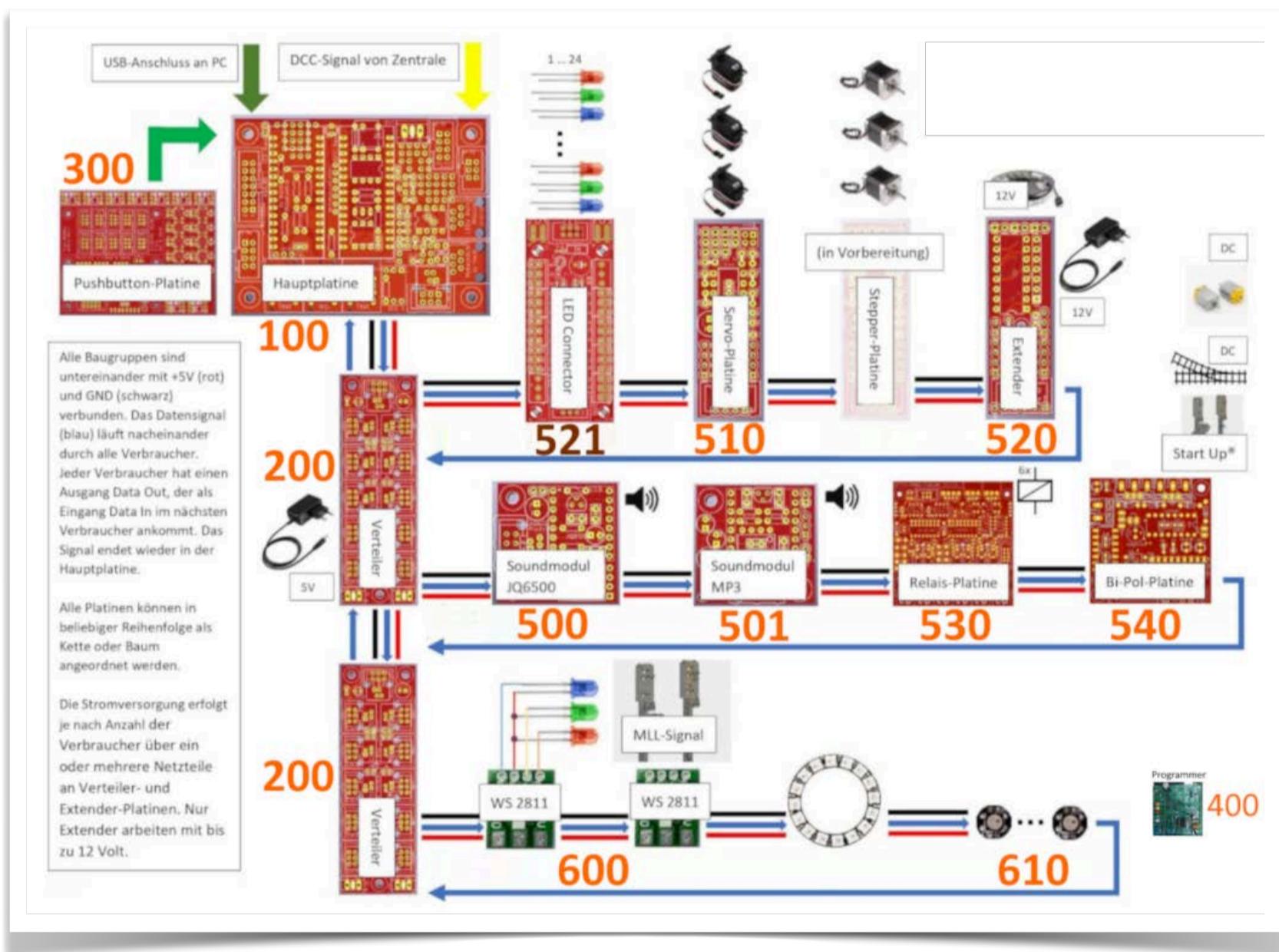
(Ende Anleitung aus Wiki)

Die Platinen

Es gibt mittlerweile viele verschiedenen Platinen für die verschiedensten Aufgaben. Ich möchte hier erst einmal nur auf die Beleuchtung eingehen. Im Wiki findet man eine Übersicht über alle Platten. Interessierte können diese Platten zum Selbstkostenpreis bestellen, die Bezugsadresse gebe ich auf einer separater Seite an.

Passenderweise sind im Wiki auch schon die entsprechenden Einkaufskörbe (für den Elektronik-Versender Reichelt) für die benötigten elektronischen Einzelteile hinterlegt. Hier kann man einfach durch Klicken und übernehmen der Einkaufsliste diese Teile bei Reichelt bestellen.

Hier einmal eine Übersicht über die zur Zeit möglichen Platten und deren Zweck:



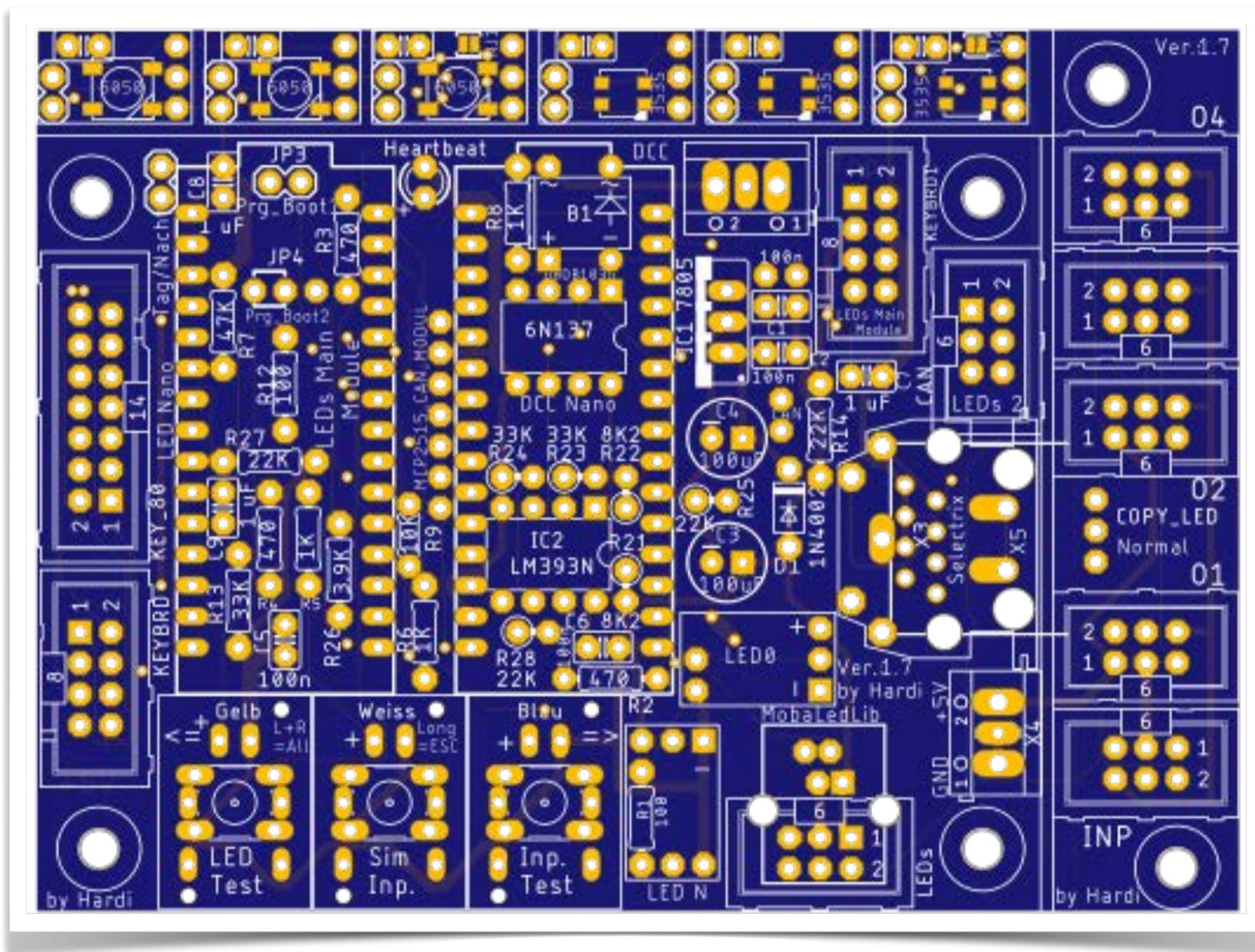
Für mein Beleuchtungs-Projekt benötige ich einmal eine Hauptplatine für DCC-Ansteuerung (Nr. 100) und zusätzlich noch einige Verteilerplatten (Nr. 200).

Die Hauptplatine (Nr. 100a)

Die Hauptplatine ist die wichtigste Platine im MobaLedLib-Universum. Diese steuert die LEDs an und sorgt auch für die Kommunikation mit der Digitalzentrale per DCC (es ist auch CAN oder Selectrix möglich, diese Protokolle setze ich aber nicht ein und darum gehe ich auch nicht näher darauf ein. Interessierte finden im MobaLedLib-Wiki viele weitere Informationen dazu).

Nachfolgend ein Bild der noch unbestückten Hauptplatine. Aber bitte nicht Verzweifeln: viele auf der Platine aufgedruckte Elemente werden bei der Grundversion für DCC nicht benötigt.

Übrigens: Ich verwende die zur Zeit aktuelle Platinen-Version 1.7.



Die obige Hauptplatine ist für 5 Euro plus Versand erhältlich (Stand 2/2021).

Als nächstes benötigt man zwei Arduino Nano. Diese erhält man aus vielen Quellen für kleines Geld. Ich habe diese z. B. bei Amazon, 3 Stück für 9,99 Euro, bestellt. Dann noch 2 WS2812 LEDs in der Bauform 5050 oder 3535. Auch diese habe ich über Amazon bestellt, wahrscheinlich gibt es dies aber auch bei Conrad oder Reichelt. Jetzt benötigt man nur noch die aufzulögenden Teile. Für diese ist im Wiki praktischerweise direkt ein Warenkorb für den Versender Reichelt hinterlegt. Dieser enthält folgende Teile (ungefährer Warenkorbwert ca. 5,- Euro + Versandkosten):

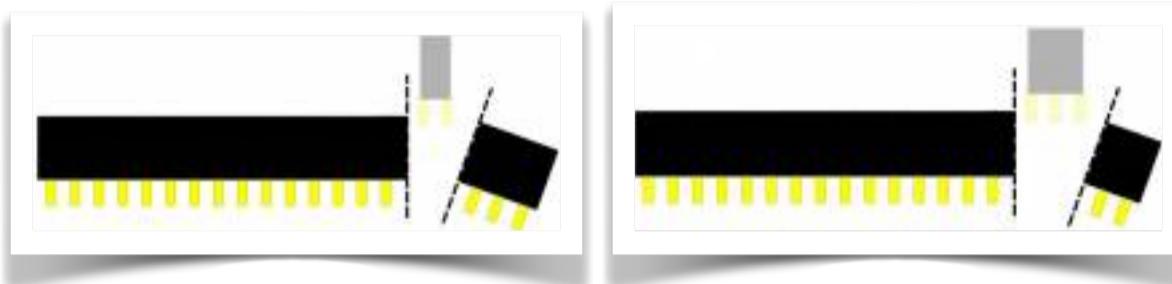
Anzahl	Bezeichnung	Beschreibung	Bestellnummer	Alternativen, Bemerkungen
1	B1	Diode 1N4148	1N 4148	
4	C5, C6, C10, C12	Keramikkondensator, 100nF, RM 2.5mm	Z5U-2,5 100N	Zur Info Die Kondensatoren C10, C12, C14, C16, C18 und C20 sind die Kondensatoren auf der Oberseite der Miniplatinen für die WS2812-LEDs
1	DCC	Lötbare Schraubklemme - 2-pol, RM 5 mm, 90°	RND 205-00045	Alternativ: RND 205-00232
1	IC3	Single Bus Buffer Gate mit 3-State-Ausgängen, SOT-23-5	SN 74LVC1G12 5DBV	Anstelle des IC3 und den Widerständen R2 und R9, kann auch der Widerstand R26 mit 3,90 KΩ bestückt werden. Zusätzlich muss dann der Lötjumper „SJ2“ auf der Unterseite geschlossen werden.
1	R2	Widerstand, 47Ω, 0.6W, 1% Gelb-Lila-Schwarz-Gold-Braun	METALL 47,0	
1	R9	Widerstand, 10KΩ Braun-Schwarz-Schwarz-Rot-BRAUN	METALL 10,0K	
5	J1, JP3, JP4, CON2, CON3	Stiftleiste, 2-pol	MPE 087-1-002	
2	CON1, CON4	Stiftleiste, 3-pol	MPE 087-1-003	
2	JP3, JP4	Jumper 2,54	JUMPER 2,54 BL JUMPER 2,54 SW	Der Warenkorb enthält je einen blauen und einen schwarzen Jumper

1	OK1	Sockel für Optokoppler 6N137, 8-pol	GS 8P	
1	OK1	Optokoppler 6N137	6N 137	
1	R8	Widerstand, 1,00KΩ Braun-Schwarz-Schwarz-Braun-Braun	METALL 1,00K	
1	R13	Widerstand, 4,70KΩ, 1%, 0.6W Gelb-Lila-Schwarz-Braun-Braun	METALL 4,70K	Der Widerstand muss je nach Bedarf an den verwendeten Lichtsensor angepasst werden. Hilfreich ist dabei diese Tabelle.
1	LDR	Photowiderstand - GL5506	GL5506 (AliExpress)	
1	SV3	Wannenstecker, 6-pol	WSL 6G	
2	U1, U4	Buchsenleiste, 2-pol	BL 1X20G8 2,54	Die vier Buchsenleisten werden aus einer langen Leiste gefertigt.
2	U1, U4	Buchsenleiste, 3-pol		
4	U2, U3	Buchsenleiste, 15-pol	BL 1X20G8 2,54	Diese Buchsenleiste muss leider geteilt werden. Bei Conrad ist auch die 15-polige Variante erhältlich.

ACHTUNG: die Position 4 (IC3, Single Bus Buffer Gate) ist extrem klein, geschätzt 2 x 3 mm. Bei meiner Bestellung hatte ich direkt 3 Stück bestellt, da ich Angst hatte, dass ich diese nicht richtig aufgelötet bekomme. Dazu aber mehr an späterer Stelle.

Buchsenleiste teilen

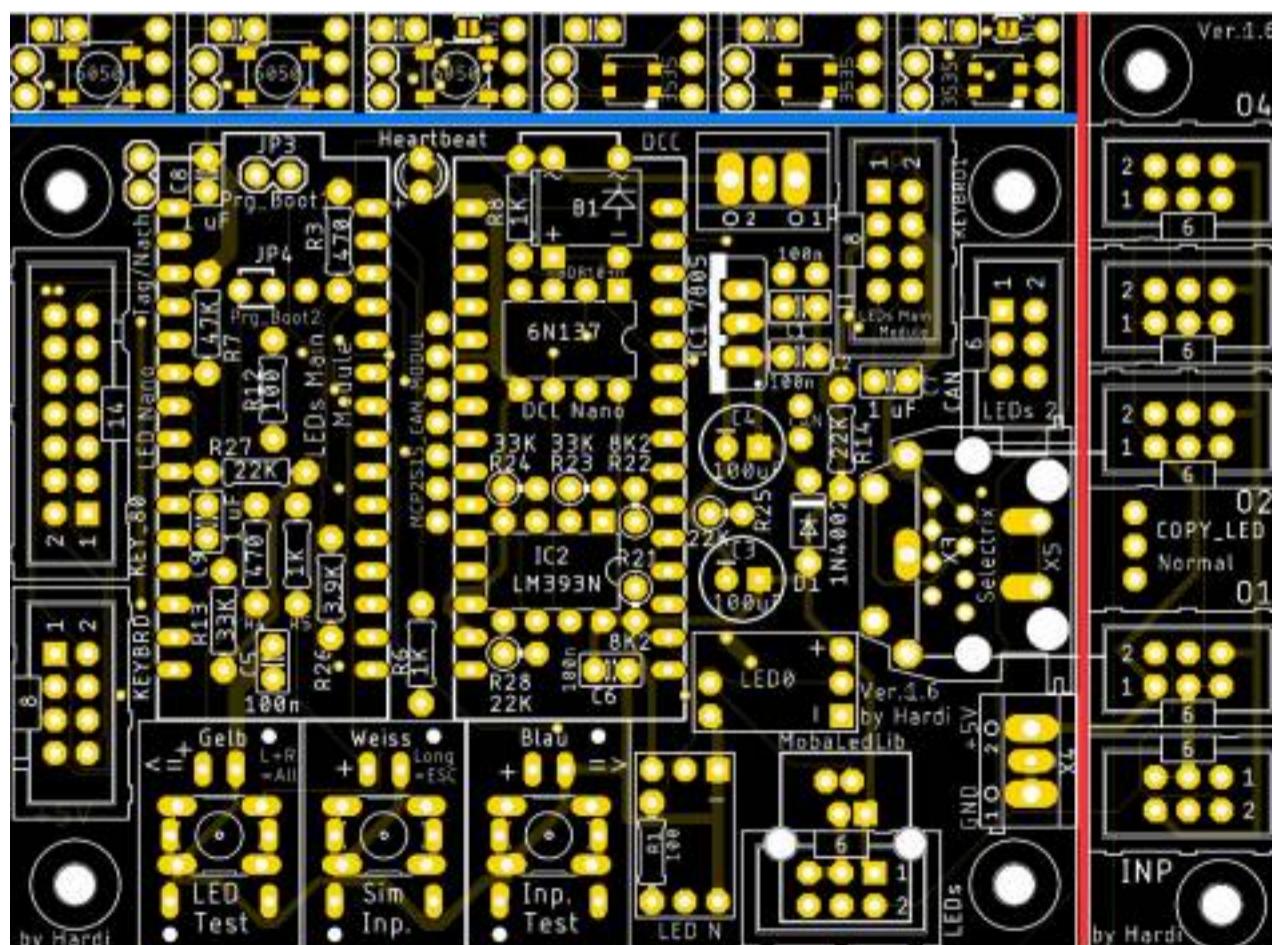
Die 20poligen Buchsenleiste für U1 und U4 wird per Säge auf die notwendigen Teilstücke ab gelängt (jeweils etwa 1mm hinter dem letzten benötigten Bein absägen). Aus einer 20poligen Leiste werden je zwei 2polige und zwei 3polige Buchsenleisten für die LED erstellt. Die übrig gebliebenen Innenstücke werden nicht benötigt. Dies ist notwendig, da die günstigen bereits fertigen Buchsenleisten nicht mehr erhältlich sind.



Bestückung - Aufbauanleitung

Platine teilen

Als erstes teilte ich die die Platine mit Dremel entlang der unten rot und blau markierten Stellen. Dabei führte ich zuerst den roten, danach den blauen Schnitt aus.



Die rechte abgetrennte Platine kann als „kleine Verteilerplatine“ genutzt werden (habe ich aber nicht gemacht) und die obere Platine ergibt 6 kleine LED-Träger. Diese werden wie folgt geteilt:

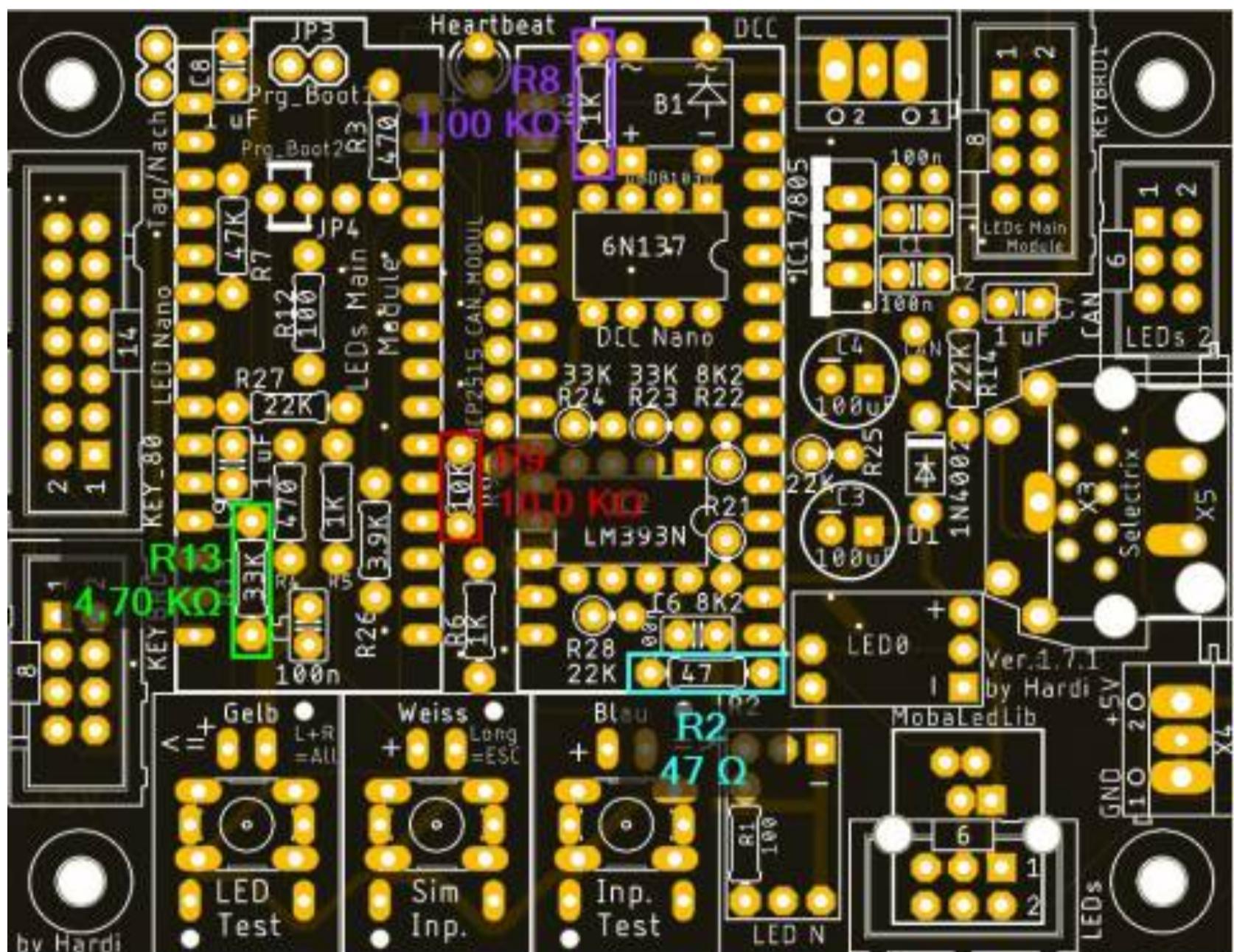


Die Bauanleitung hierfür folgt tiefer.

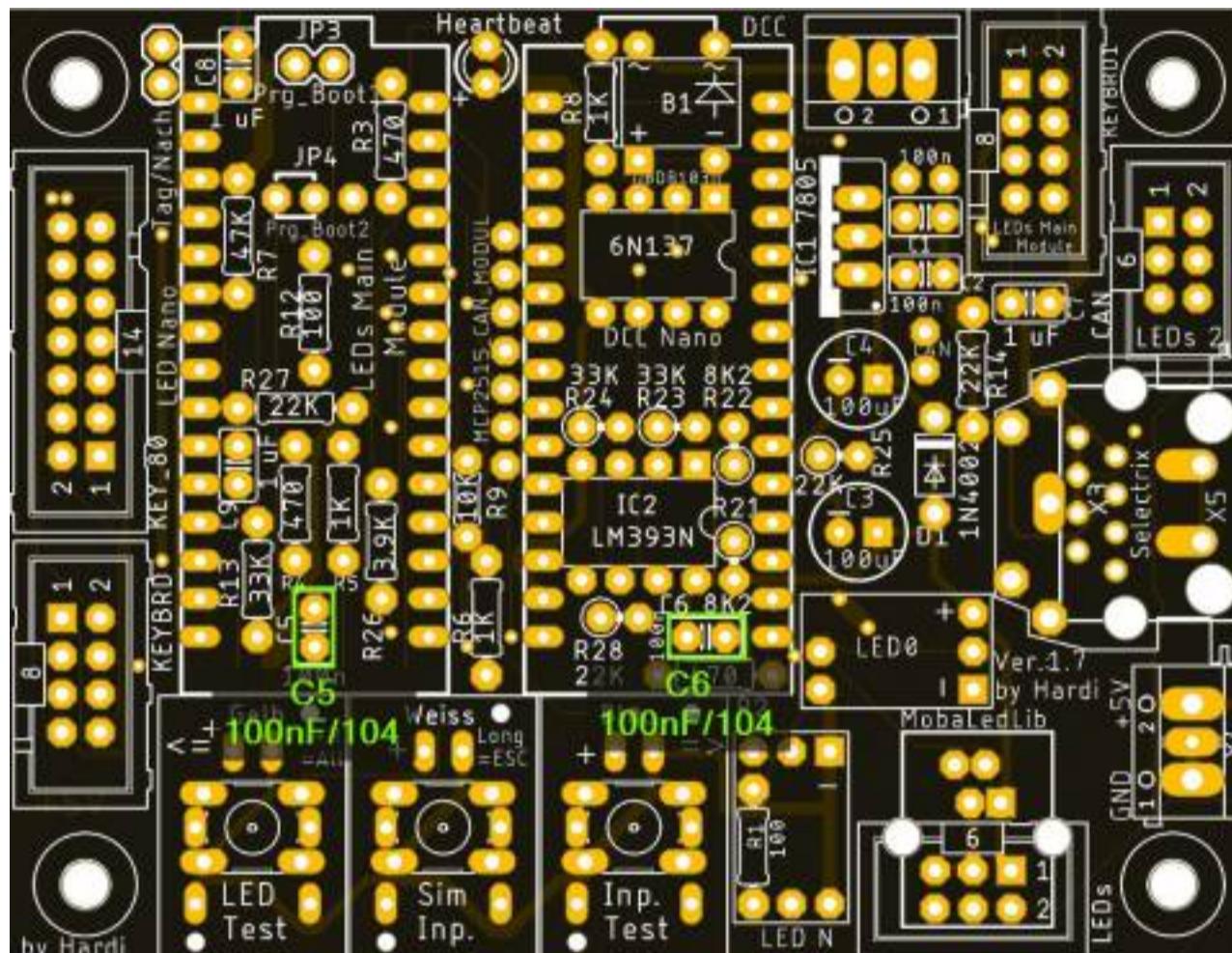
Bestückung Oberseite

Bitte nicht den Arduino direkt auf die Hauptplatine auflöten. Andernfalls ist ein Austausch bei einem Defekt nicht möglich.

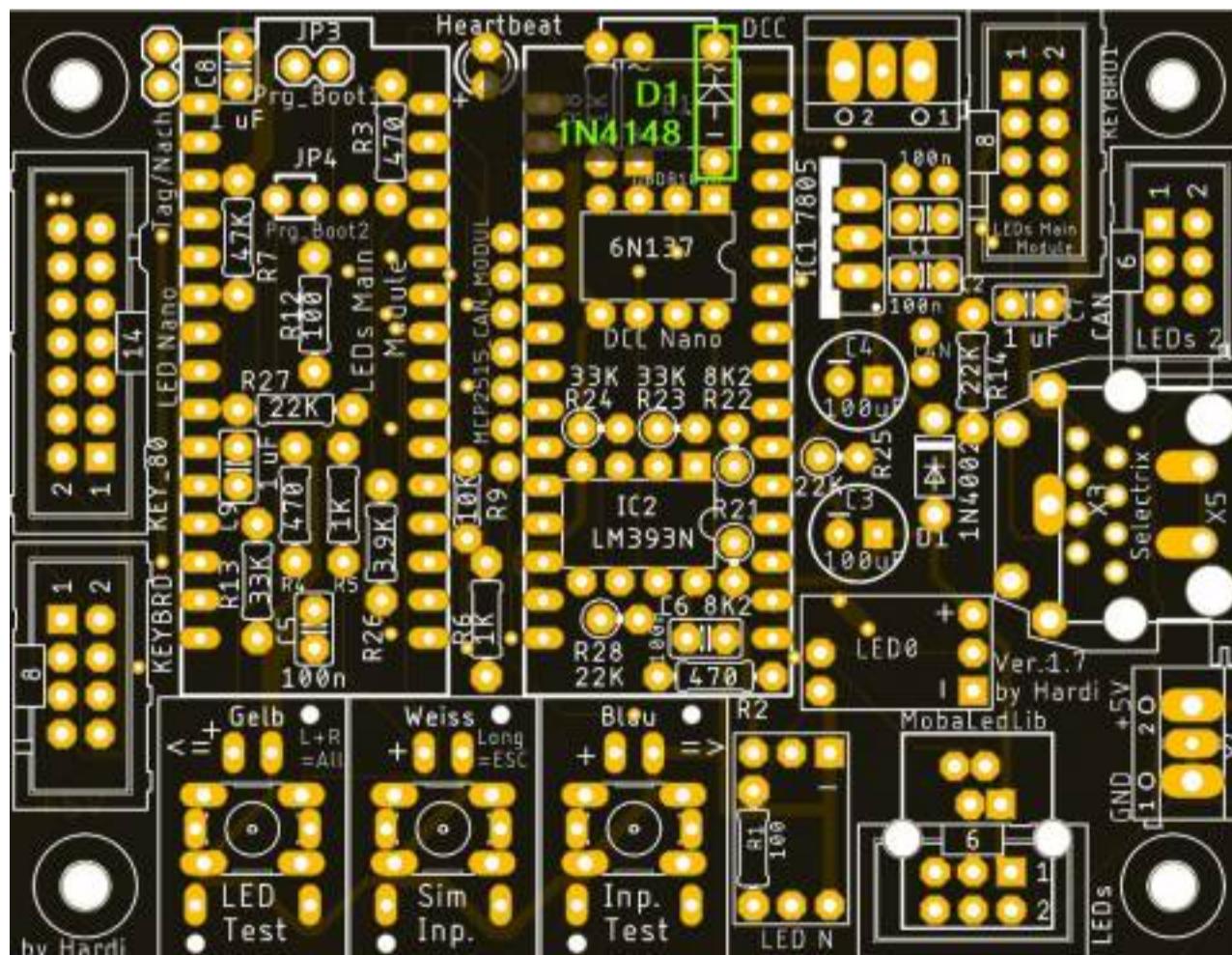
Den Anfang auf der Oberseite der Platine machen die Widerstände R2 ($47\ \Omega$), R8 ($1,0\ K\Omega$), R9 ($10,0\ K\Omega$) und R13 ($4,70\ K\Omega$)



. . . gefolgt von den Keramikkondensatoren C5 und C6 (je 100nF) . . .

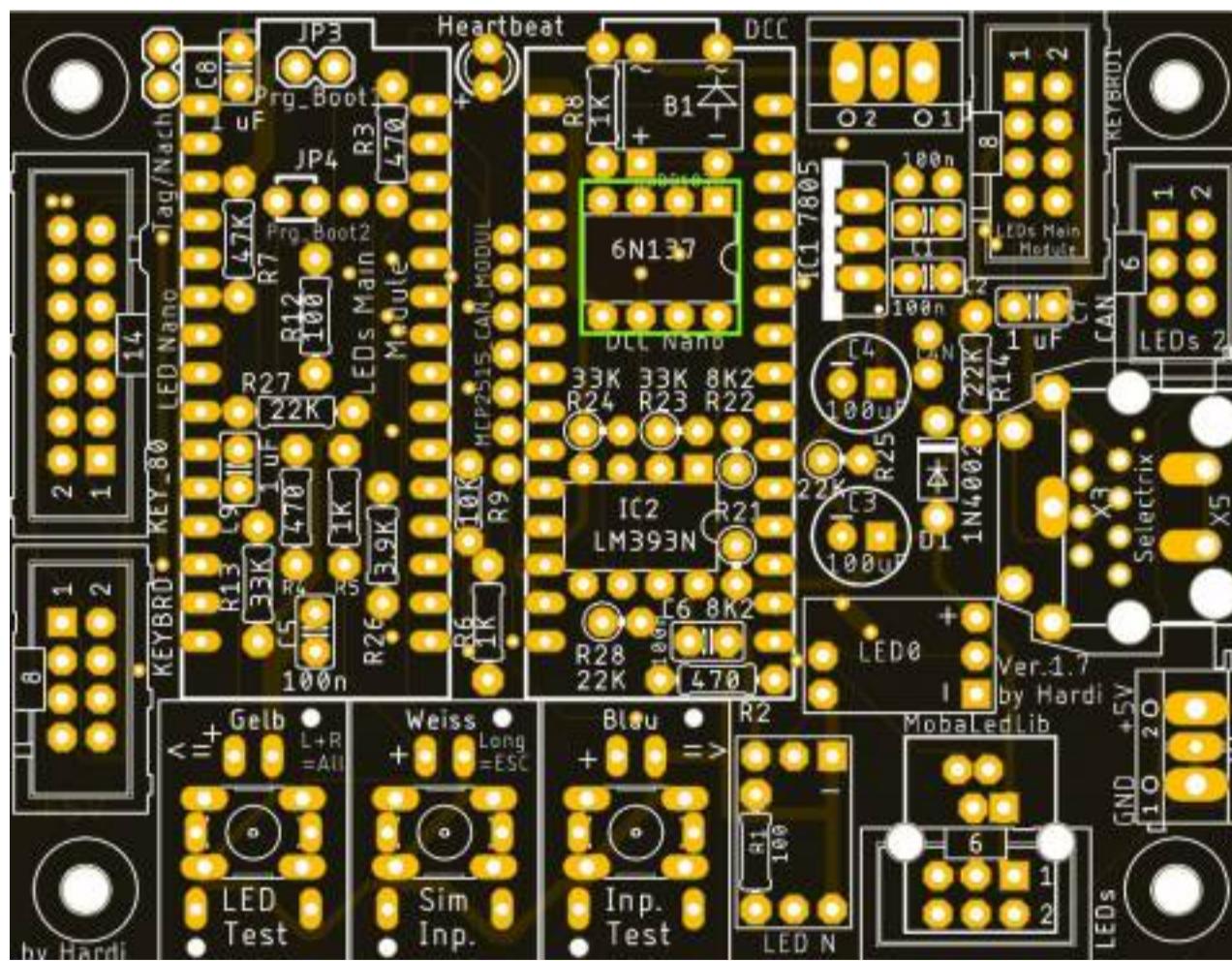
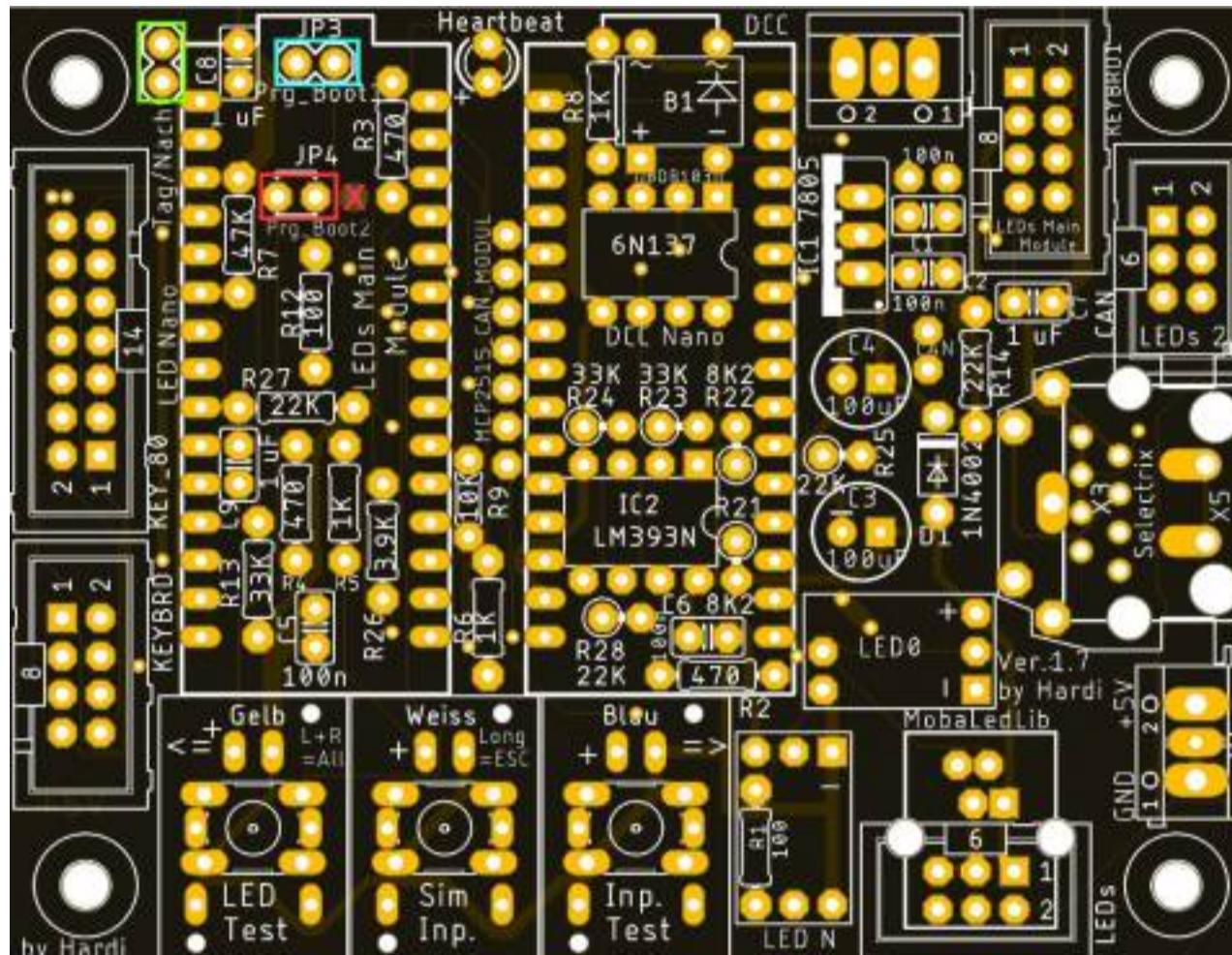


. . . und der Diode B1 (1N4148) - hier bitte unbedingt Pooling beachten (Strich ist oben)



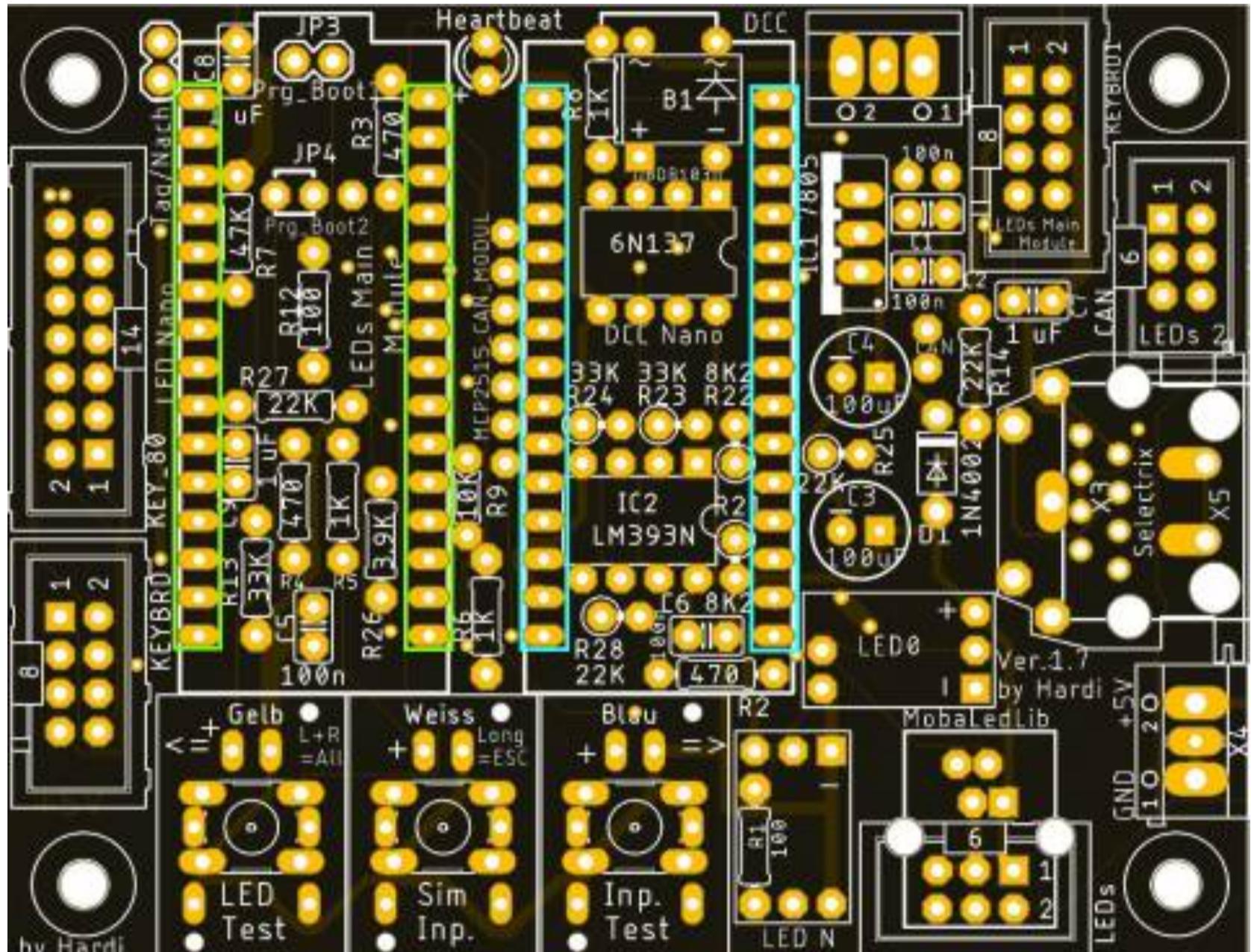
Die Stifteleisten J1, JP3 und JP4, sowie der Sockel für OK1 folgen danach.

Bei dem Jumper JP4 gibt es eine Besonderheit. Dort wird eine 2-Polige Stifteleiste anstelle der 3-Poligen verwendet und der rechte Kontakt bleibt frei. Dieser ist durch ein rotes X Markiert.



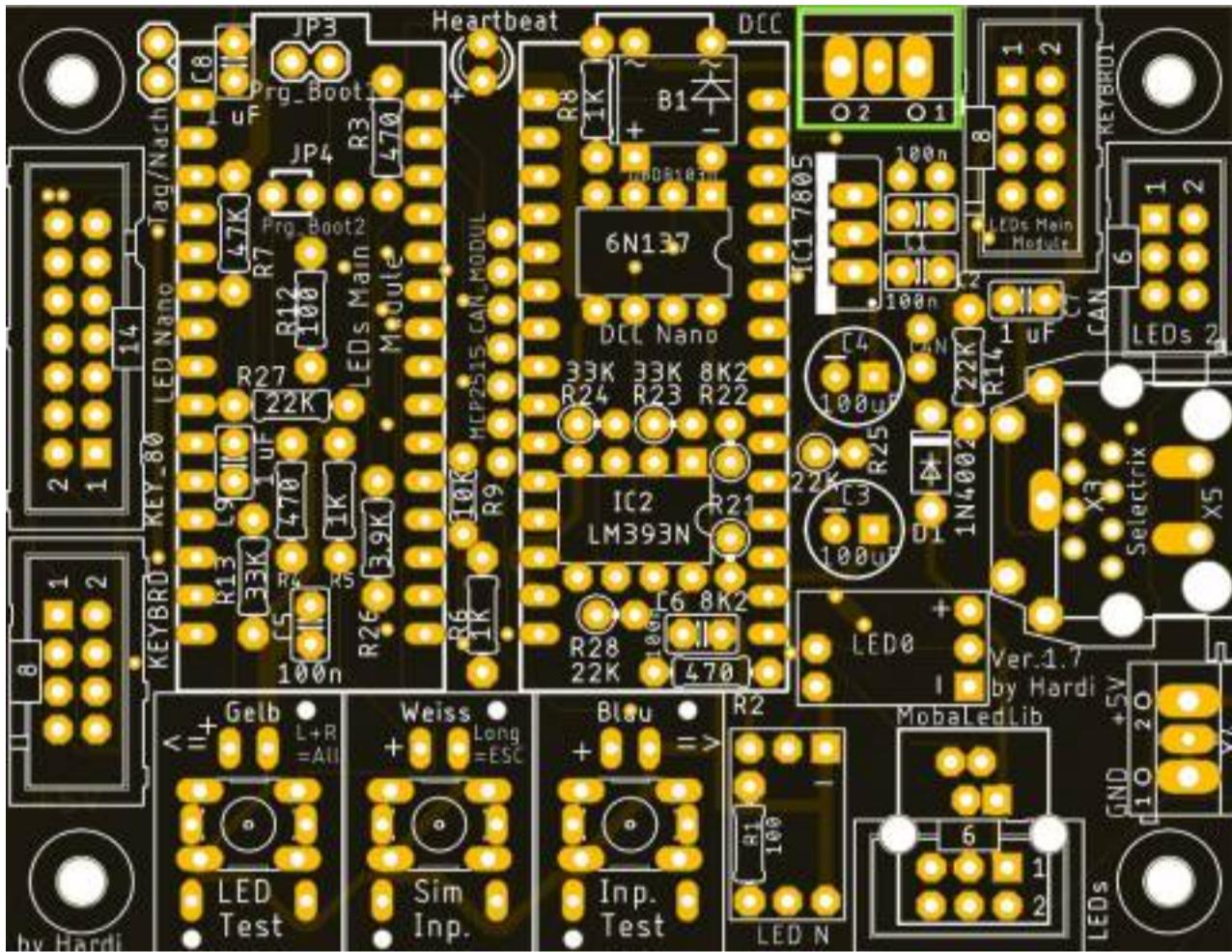
Die nächsten Bauteile sind die vier Buchsenleisten für die beiden Arduinos.

Damit diese passen, müssen die Leisten, sollte man die aus dem Reichelt Warnkorb gekauft haben, auf 15 Pole gekürzt werden.

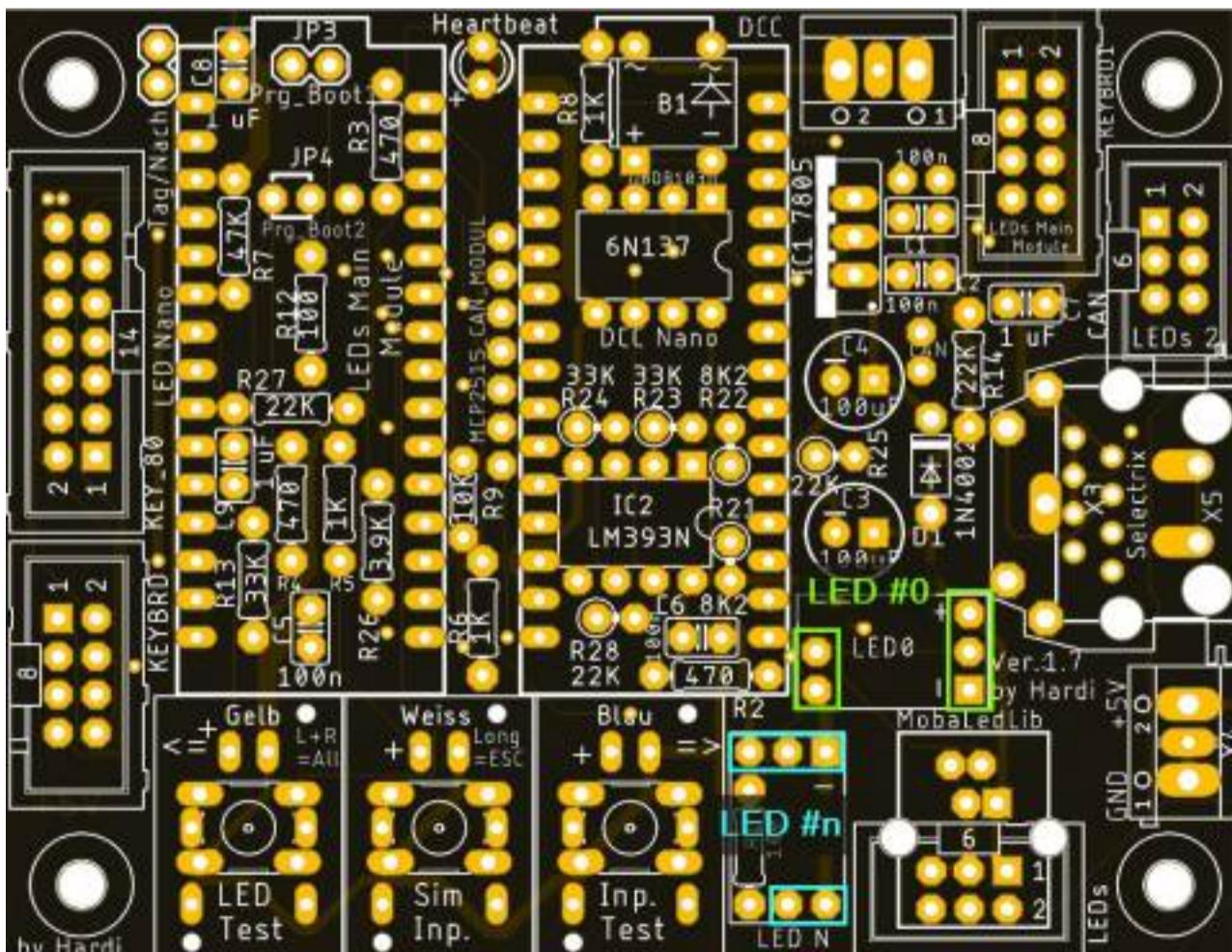


Bei den Buchsenleisten für die Arduinos kann man sich die Arbeit vereinfachen, indem man die Leisten auf die Arduinostifftleisten steckt und dieses dann in die Hauptplatine steckt.

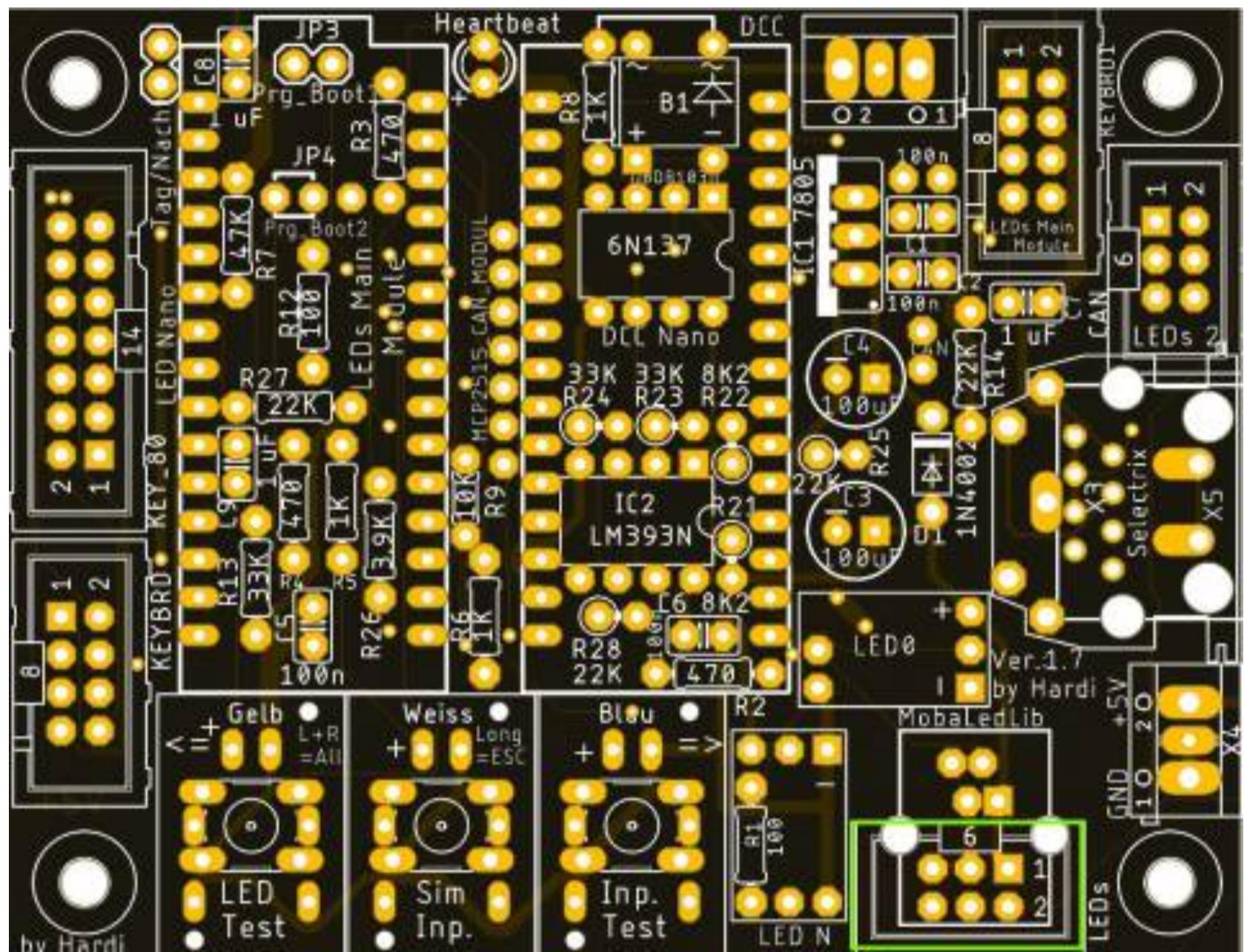
Nun folgt das Einlösen der Anschlussklemmen für das DCC-Signal . . .



und die Buchsenleisten für die WS2812-LED-Platinen. Dies klappt am leichtesten, wenn man die LED-Platinen bereits nach der entsprechenden Anleitung (siehe tiefer) zusammengebaut hat.

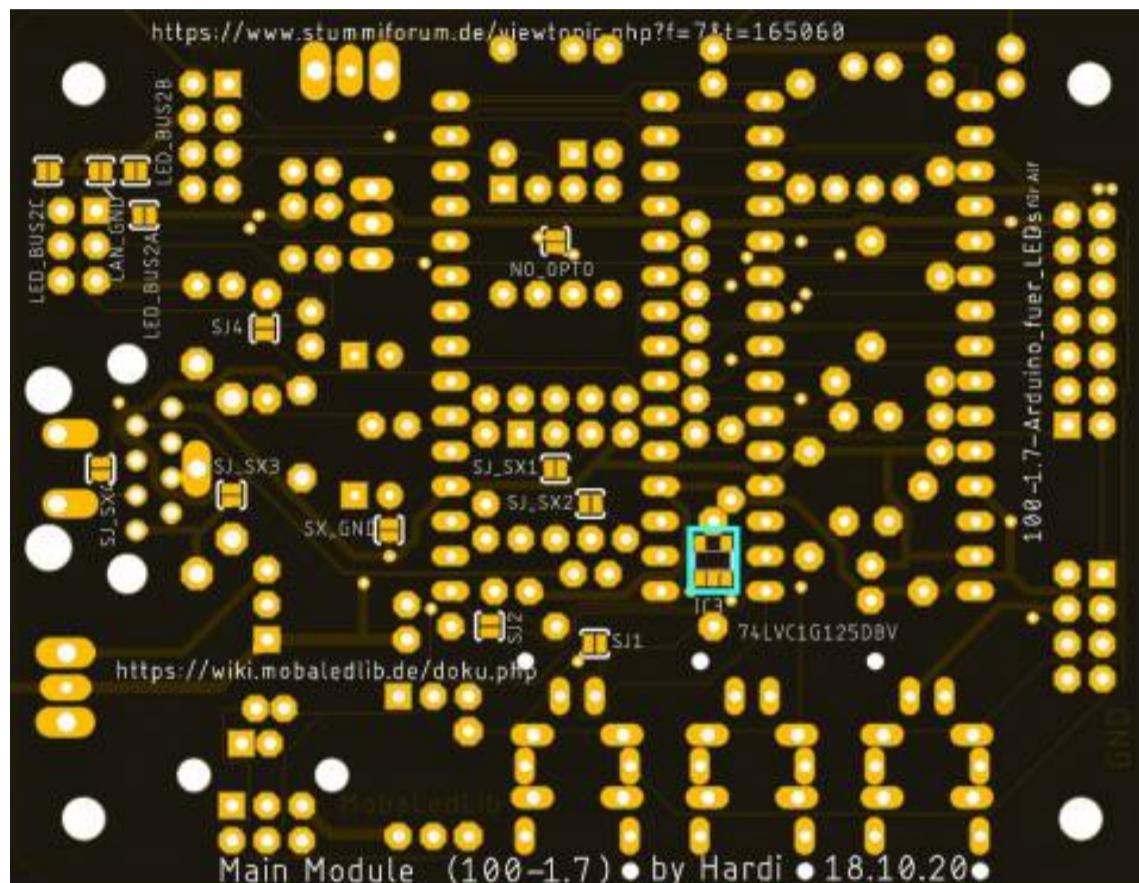


Als letztes Bauteil auf der Oberseite kommt der Wannenstecker für die LEDs an die Reihe.



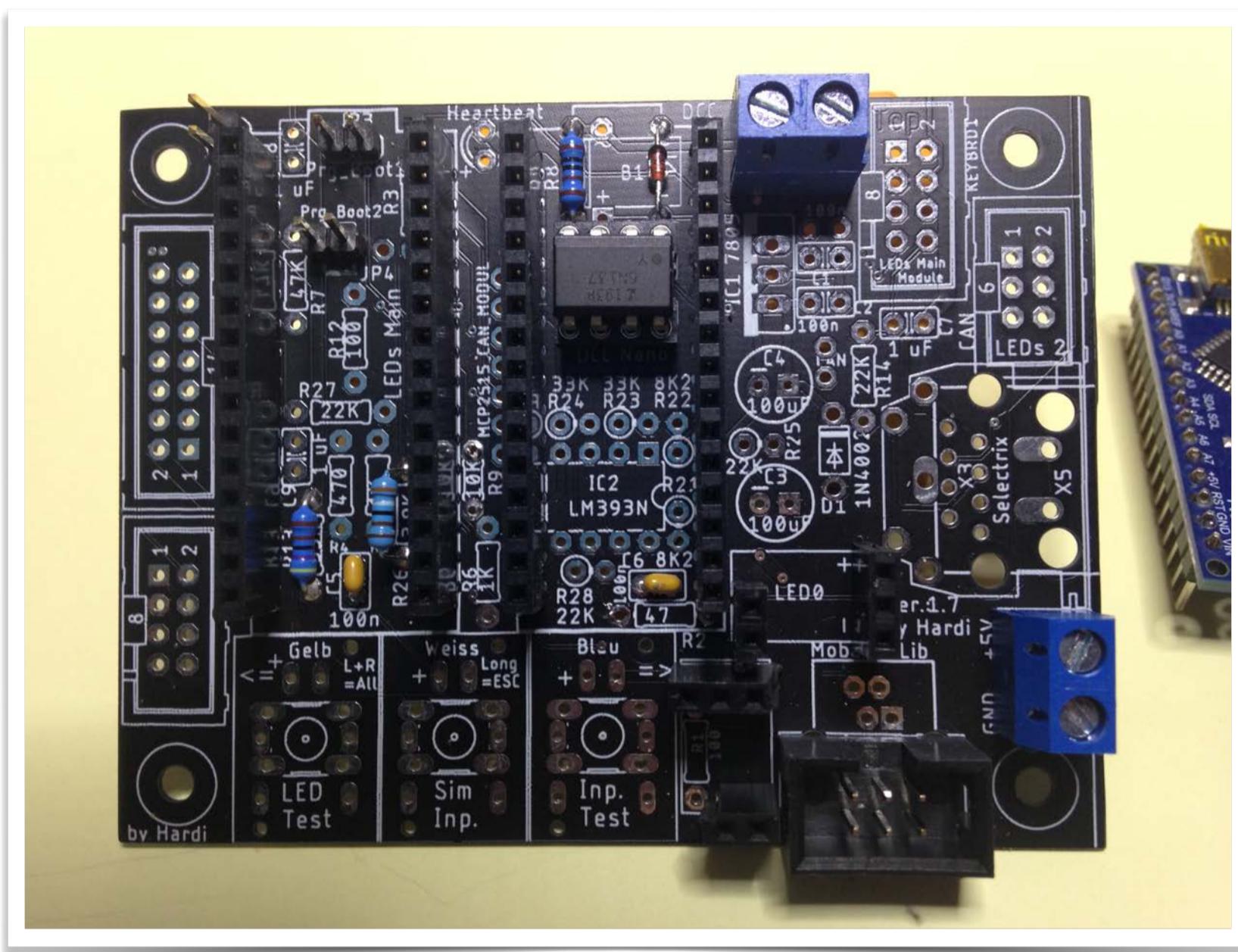
Bestückung Unterseite

Auf der Unterseite wird IC3 bestückt. Dieser sorgt zusammen mit R21 und R9 auf der Oberseite dafür, dass sich der LED-Nano mit der Software bespielen lassen, auch wenn der DCC-Nano dabei bislang Probleme bereitet hat. Da es sich dabei um ein Bauteil im Formfaktor „SOT-23-5“ handelt, bitte ein besonderes Augenmerk auf die feinen Pinabstände beim Löten geben. Nach dem Einlöten unbedingt mit einer guten Lupe die Lötstellen kontrollieren.

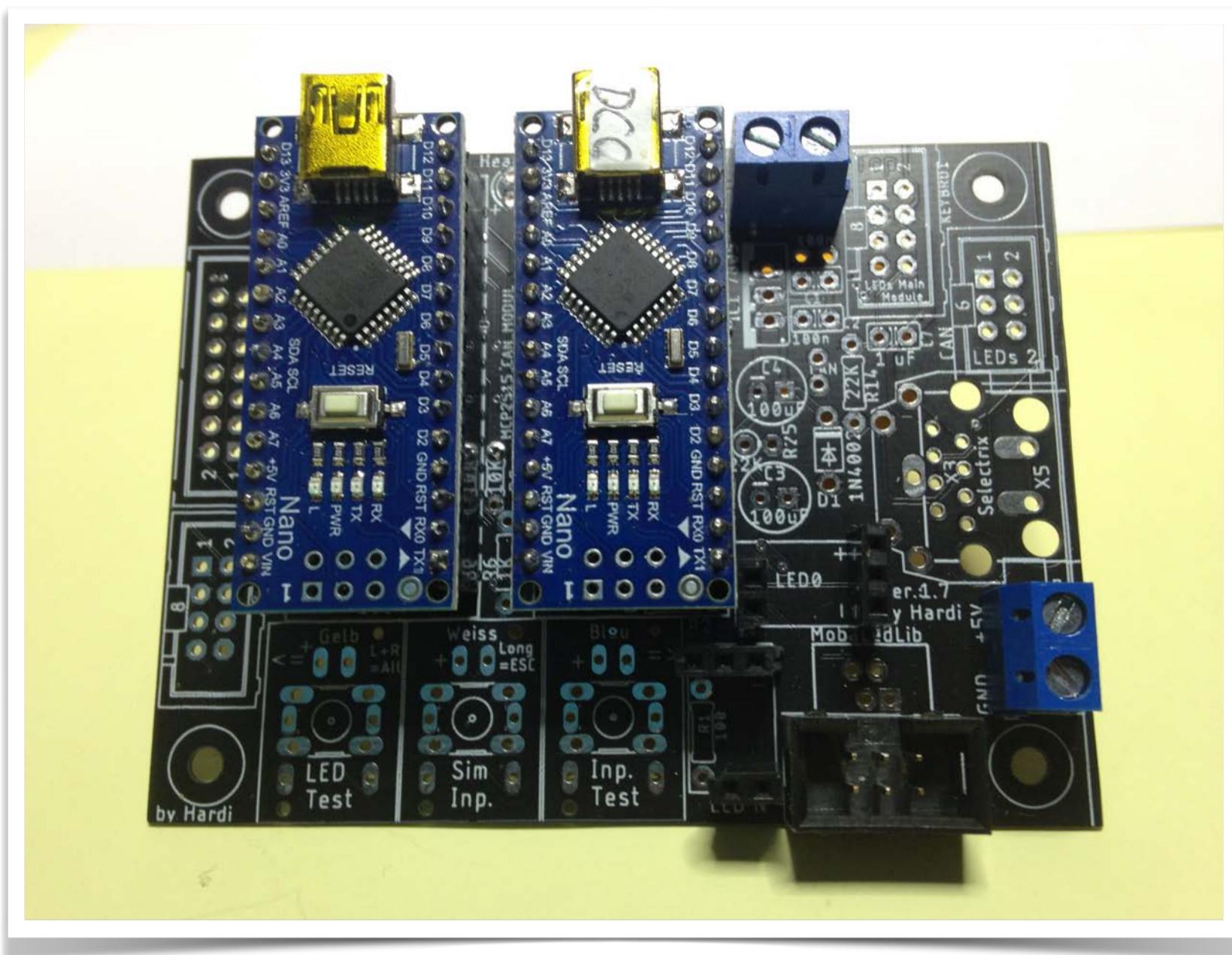


Trotz Lupe hatte ich kein gutes Gefühl beim Einlöten dieses extrem kleinen Bauteils. Unter der Vergrößerung sah erst einmal alles gut aus. Später beim ersten Test der Platine ohne DCC funktionierte alles perfekt. Als ich dann LEDs per DCC-Befehl einschalten wollte, tat sich nichts. Ich hatte das Gefühl, dass der linke LED-Arduino richtig arbeitet und nach dem Anschließen des rechten DCC-Arduino per USB an meinen Rechner konnte ich auf dem „Serial Monitor“ sehen, dass dieser auch die DCC-Signale von meiner Digitalzentrale empfängt. Nur leider fand keine Kommunikation zwischen dem rechten DCC-Arduino und dem linken LED-Arduino statt. Nachdem ich mein Problem im Stummiforum geschildert hatte kamen prompt die Antworten. Letztendlich stellte sich heraus, dass der kleine IC3 von mir nicht richtig eingelötet war. Da dieser nicht unbedingt erforderlich ist sollte ich diesen entfernen, den Widerstand R26 mit 3,7 kΩ (ACHTUNG: nicht im Reichelt Warenkorb enthalten) einlösen, die beiden Widerstände R2 und R9 wieder entfernen und zusätzlich die Lötbrücke „SJ2“ auf der Platinenrückseite mit einem Tropfen Lot schließen. Nachdem ich dies durchgeführt hatte funktionierte die Hauptplatine einwandfrei. Ich wurde aber auch darauf hingewiesen, dass es bei der Programmierung der Arduino eventuell Probleme bei fehlendem IC3 geben könne, dann hätte ich die Arduino aus der Platine ausstecken, mittels USB programmieren und wieder einstecken sollen. Aber es funktioniert auch so ohne Fehler.

So sieht meine fertige Platine ohne Arduino aus:

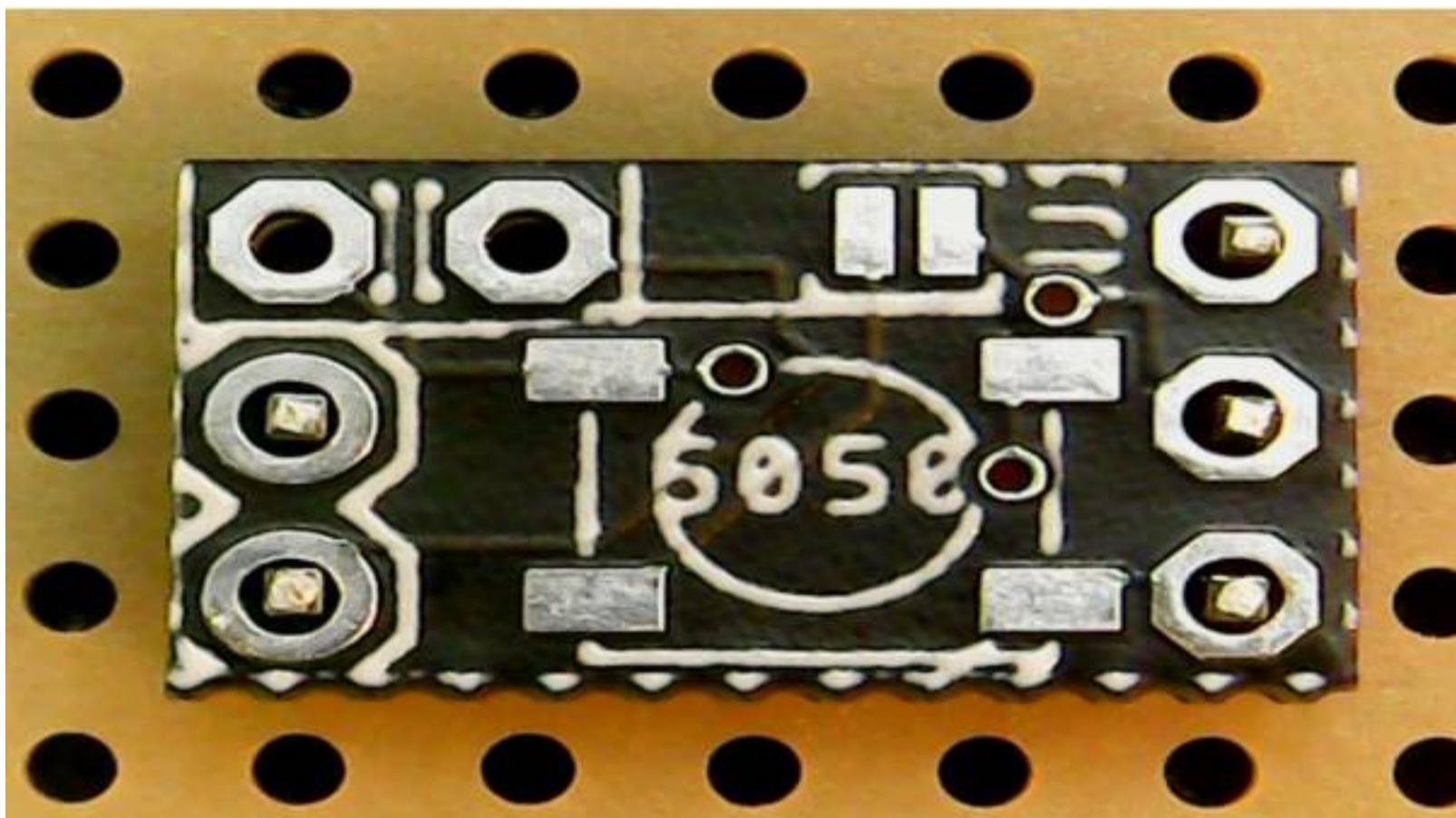


Und so mit den beiden Arduino (der linke ist der „LED-Arduino“, der rechte der „DCC-Arduino“:

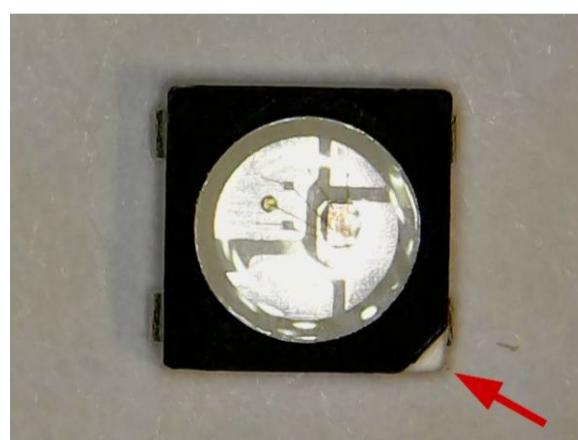
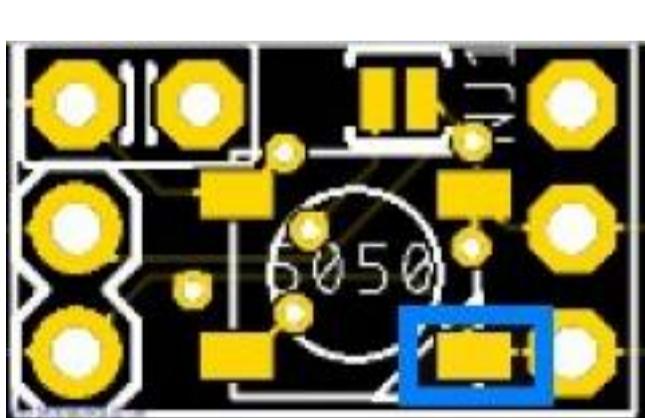


Als nächstes baue ich die beiden Mini-LED-Platinen zusammen.

Den Anfang machen die beiden Stiftleisten auf der Unterseite der Platine. Am einfachsten ist es die Platinen entweder in die Hauptplatine einzustecken, oder mitsamt den Stiftleisten in eine Lochrasterplatine. Dadurch wird es zum Kinderspiel die Miniplatine und die Stiftleisten zu verlöten.



Im Anschluss wird die LED auf der Vorderseite aufgelötet. Dies kann auch per Handlötzung erfolgen, wenn es noch nicht zu viel Kaffee zum Frühstück gab. Als erstes auf das Lötpad unten links etwas Lötzinn geben und kurz auskühlen lassen. Nun mit einer breiten Pinzette die LED nehmen und grob ausrichten, dass die Markierung der LED zu dem Lötpad unten rechts ausgerichtet ist.



Das Lötzinn mit dem Lötkolben wieder schmelzen und die LED mit der Pinzette auf das Lötpad mit dem geschmolzenen Lötzinn legen und den Lötkolben entfernen. Nach ca. 5 Sekunden ist das Lötzinn wieder fest und man kann die LED loslassen. Wenn die Position und passt, kann man nun die anderen Lötpads mit dem Lötkolben erhitzen und ganz wenig Lötzinn auftragen. Durch die Adhäsion fließt das Lötzinn von selber zwischen die jeweiligen Lötpads der Platine und der LED. Sobald alle vier Kontakte verlötet sind, die Platine für ein paar Minuten auskühlen lassen. **Auf Grund der Größe, wird die Platine sehr schnell sehr heiß und man verbrennt sich die Finger.** Vor dem Einlöten des Kondensators empfiehlt es sich die Lötungen der LED mit einer guten Lupe zu kontrollieren.

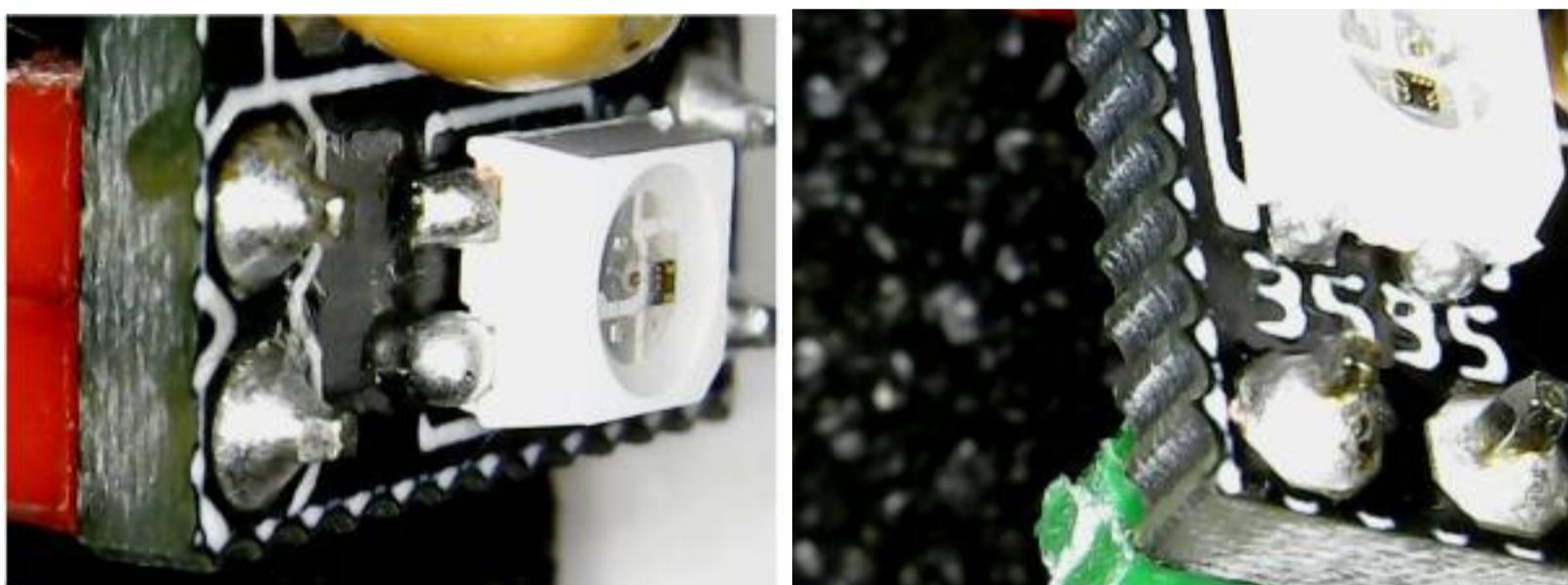
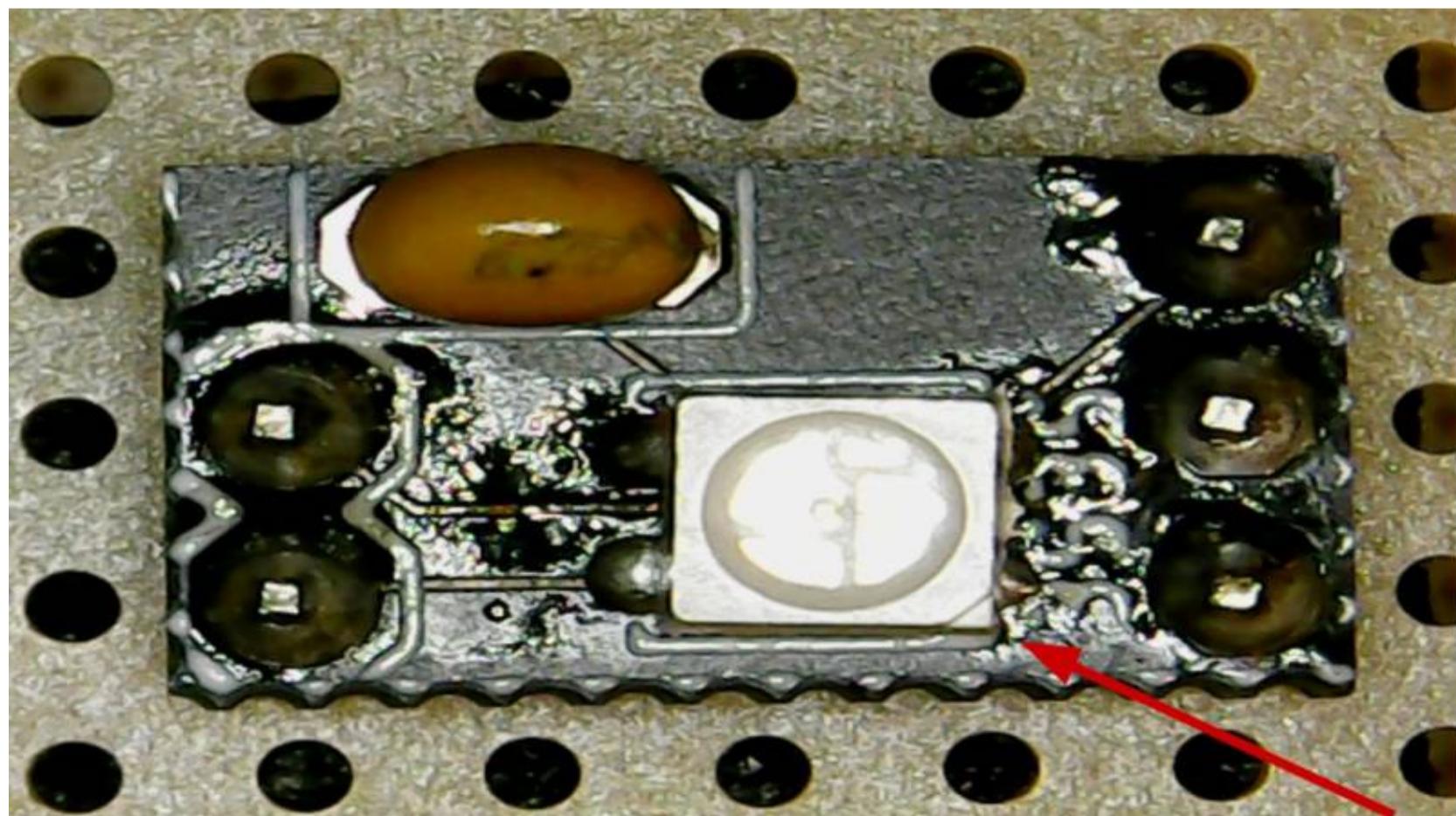


Bild der fertigen Platine



Damit der LED-Bus einwandfrei funktioniert muss noch ein Lötjumper geschlossen werden. Ich habe mich für die Version „Normalbetrieb“ entschieden und darum muss der Lötjumper „NJ1“ auf der Platinenvorderseite noch geschlossen werden.

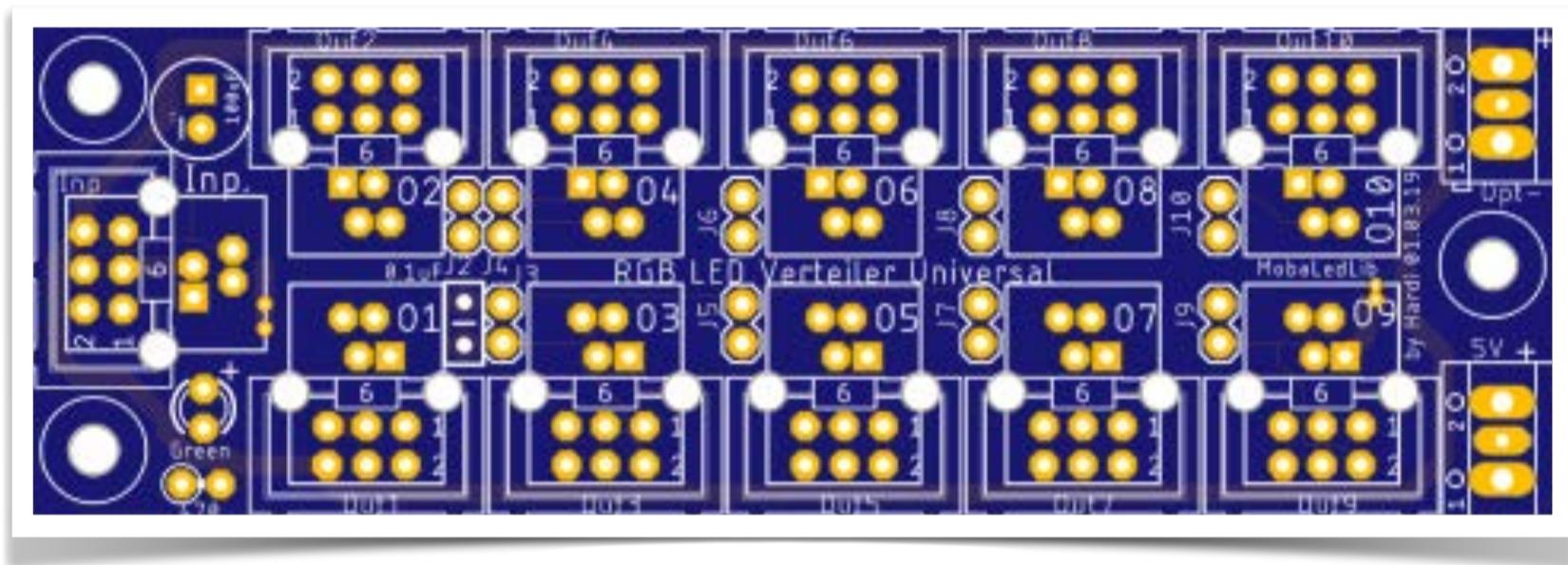
Jetzt können beide LED-Platinen auf die Hauptplatine gesteckt werden.

Wofür sind die beiden LED-Platinen da? Beide dienen der Kontrolle des LED-Bus. Wenn man das nachher richtig im Prog_Generator definiert blickt die erste LED-Platine (das ist die rechts neben dem DCC-Arduino) regelmäßig (sogenannter „Heartbeat“ / Herzschlag) und die letzte LED des Busses (das ist dann die unterhalb des DCC-Arduino) blinkt entsprechend mit. Blinkt nur die erste und nicht die letzte, dann gibt es irgendwo im LED-Bus eine Unterbrechung.

Die Verteilerplatine (Nr. 200)

An die Verteilerplatine können mehrere Häuser, Ampeln, Soundmodule, Servomodule usw. angeschlossen werden.

Hier ein Bild der umbestückten Platine:



Diese Verteilerplatine kostet einzeln 3,00 Euro + Versand, auch hierfür ist wieder ein Reichelt-Warenkorb (ca. 6,- Euro + Versand) hinterlegt:

Stückliste

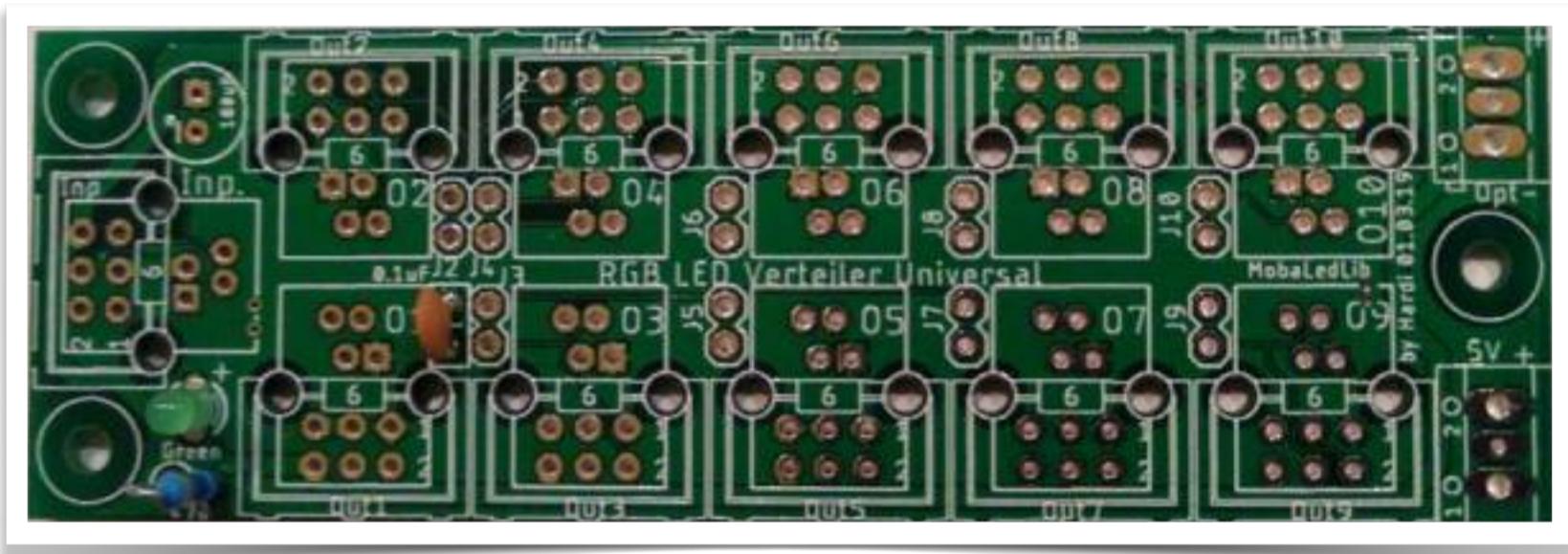
Anzahl	Bezeichnung	Beschreibung	Bestellnummer	Bemerkung
1	Platine	200_Verteilerplatine	Alf	Platine kann über Alf bezogen werden
11	INP, O1, O2, O3, O4, O5, O6, O7, O8, O9, O10	Wannenstecker, 6-polig, gerade	WSL 6G	
11	INP, O1, O2, O3, O4, O5, O6, O7, O8, O9, O10	Pfostenbuchse, 6-polig	PFL 6	
1	LED Green	LED, 3 mm, grün, 20 mcd, 40°	LED 3MM GN	
1	R21	Widerstand, 470 Ω	METALL 470	
1	C9	Keramikkondensator, 100nF, RM 2.5mm	Z5U-2,5 100N	

1	C8	Elko, radial, 100 μ F, 16 V	SM 100/16RAD	
2	5V, Opt IN	Lötbare Schraubklemme 2-pol, RM 5 mm, 90°	RND 205-00045	Alternativ: RND 205-00232
9	J2, J3, J4, J5, J6, J7, J8, J9, J10	Stiftleisten, 2-polig	MPE 087-1-002	
1	JUMPER	JUMPER 2,54 SW	JUMPER 2,54 SW	Jumper 2,54, mit Griffflasche, schwarz JUMPER 2,54GL SW

Bestückungsanleitung Verteilerplatine

Grundsätzlich sollte man zuerst die niedrigen/flachen Bauteile einlöten.

Auf der Oberseite zuerst den Widerstand R21 (470Ω), LED Green und den Keramikkondensator C9 (100nF),



dann 9 x die Stiftleisten (J2 - J10),



dann den Elektrolytkondensator C8 (100µF, ACHTUNG: Polarität beachten: „-“ ist auf der Platine und dem Bauteil markiert) und anschließend die 11 Wannenstecker INP, O1 bis O10 und zum Schluss die beiden Schraubklemme „5V“ & „Opt IN“.

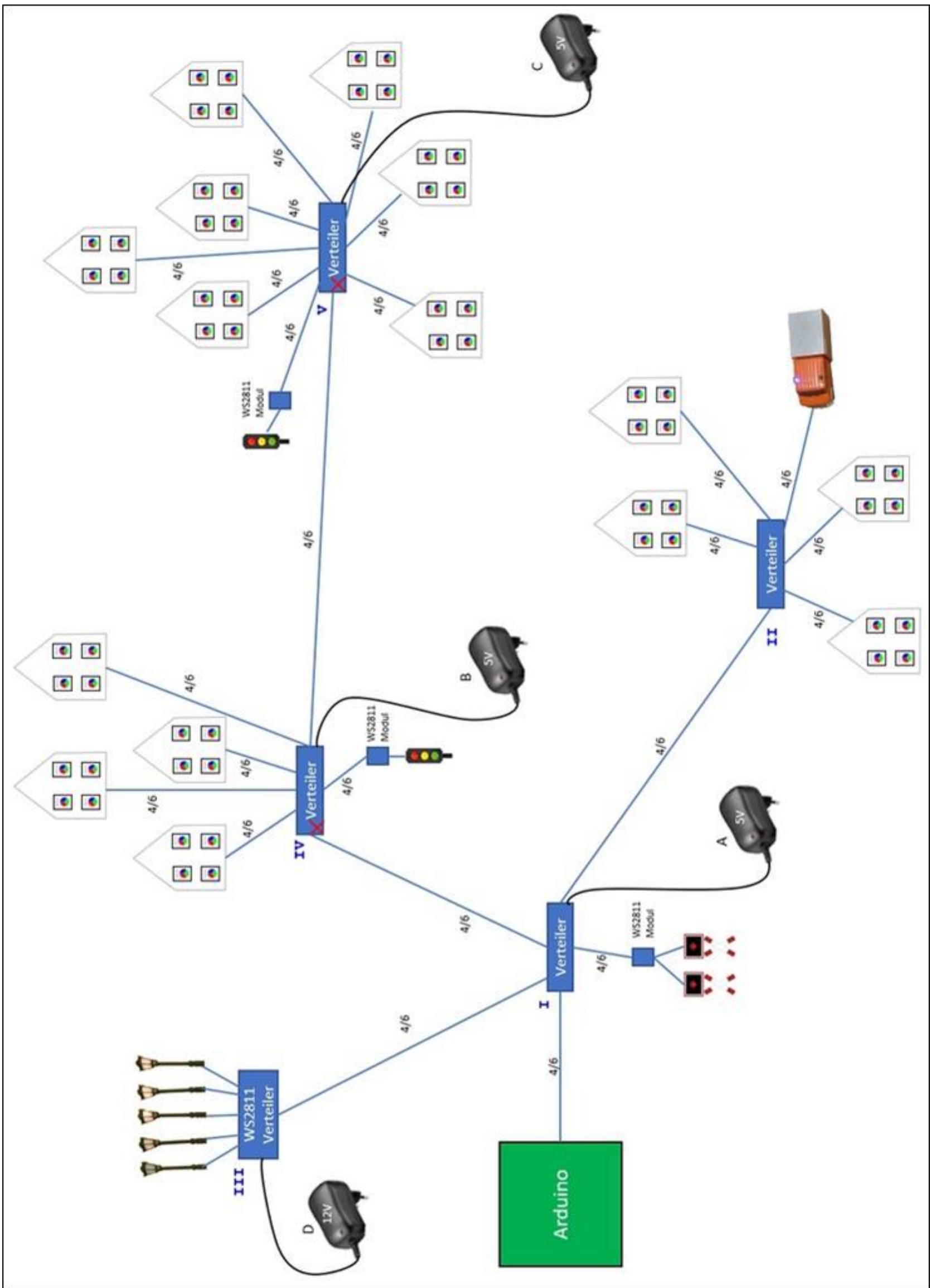
Ich habe die angegebenen 6-poligen Stecker genutzt. Möglich sind auch 4-polige Stecker oder RJ10-Anschlüsse. Hierzu mehr im MobaLedLib-Wiki.

So sieht die Verteilerplatine fertig aus:



Zum Anschluss eines Netzteile (5V, 2A) sind die beiden unteren rechten Schraubklemmen vorgesehen. Wenn die Hauptplatine mit der Verteilerplatine verbunden ist wird diese auch von dem an der Verteilerplatine angeschlossenem Netzteil mit Spannung versorgt. Für weitere Tests reicht das jetzt erst einmal aus. Später, wenn die einzelnen Platinen unter meiner Modellbahnanlage montiert sind und alle Stecker belegt sind muss ich auf der Rückseite der Verteilerplatten den Jumper „J-Power“ mit einem scharfen Messer auftrennen. So kann ich die Spannungsversorgung in einzelne „Inseln“ unterteilen und mehrere Einspeisungen mittels Netzteil durchführen. Auch hierzu weitere Informationen im bereits so oft genannten Wiki. ACHTUNG: mit Strom ist nicht zu spaßen!

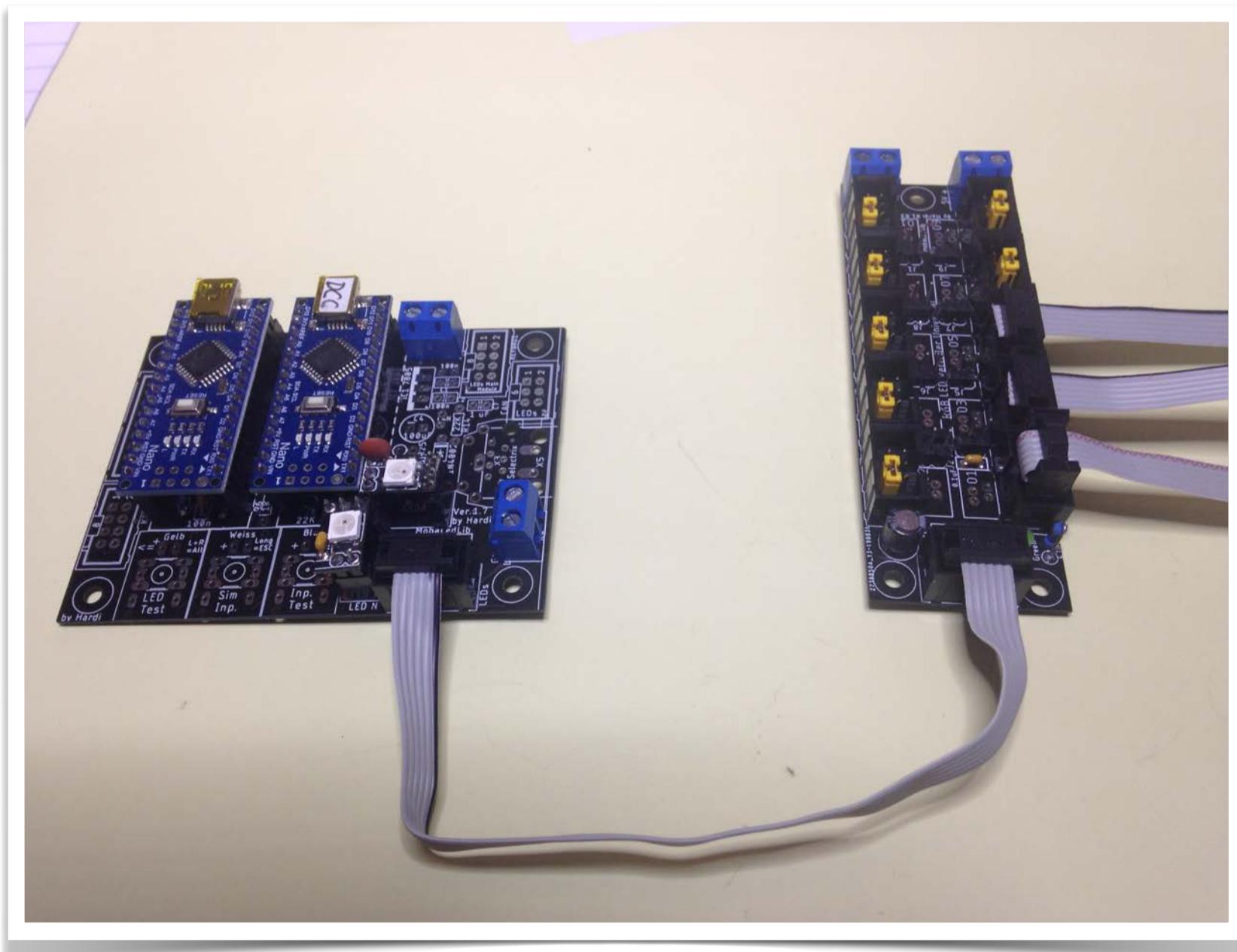
Das Bild auf der nächsten Seite zeigt eine mögliche Unterteilung in verschiedenen Bereiche.



Die Unterteilung kann sich auch an der Anordnung der LEDs auf der Anlage orientieren. Man kann für jeden Bereich ein eigenes kleines Netzteil vorsehen. Im nebenstehenden Bild versorgt das Steckernetzteil A den Arduino und den Verteiler I und II. Die Straßenlaternen sollen mit 12 - 16 V versorgt werden. Sie bekommen ihre Versorgungsspannung über das 12 V Netzteil D. Die Häusergruppe an Verteiler IV werden von der Spannungsquelle B gespeist. Das rote X auf der Verteilerplatine symbolisiert das der Jumper „J_Power“ hier getrennt wurde. Auch der Verteiler V ist getrennt. Damit wird die dritte „Stadt“ separat versorgt. Die LEDs auf den Verteilerplatinen leuchten, wenn die Versorgungsspannung anliegt. Sie sind hinter dem Jumper „J_Power“ angeschlossen.

Als letztes habe ich mich noch einige Verbindungskabel aus 6-poligem Flachkabel angefertigt (hierzu gibt es eine Anleitung auch im Wiki). Mit einem verbinde ich die Hauptplatine und eine Verteilerplatine. An der Hauptplatine schließe ich die Häuser an. Je Haus habe ich vier WS2811-LEDs eingebaut.

So sieht das jetzt aus:



Damit das Digitalsignal sowohl die gesamte Verteilerplatine durchläuft als auch wieder zurück zur Hauptplatine (damit die 2. Heartbeat-LED die Richtigkeit der LED-Kette anzeigt) geht müssen die Digital-IN und Digital-OUT Pins in den nicht genutzten Steckern mittels Jumper (in meinem Foto sind das die kleinen gelben Steckerchen in den Wannensteckern) verbunden werden. Falls ich jetzt von dieser Verteilerplatine ein Verbindungskabel zur nächsten Verteilerplatine verlege und ich in einem leeren Stecker den Jumper vergessen habe, so ist das Digitalsignal dort unterbrochen und gelangt nicht in die nächste Verteilerplatine und auch nicht mehr zurück zur Hauptplatine zur 2. Heartbeat-LED und somit blinkt diese nicht mehr. Genial nicht wahr?

So, das war erst einmal alles.

- die Software ist installiert und funktioniert
- die Platinen sind zusammengelötet, kontrolliert und verbunden
- die LEDs in den Häuschen sind richtig angeschlossen und ich habe mit genau notiert, wo welche LED in welchem Haus ist.

Dann kann es jetzt mit der Programmierung losgehen!

Viel Spaß!

Was ist der Programm Generator

Mit diesem Programm können die Beleuchtungs- und andere Effekte für eine Modelleisenbahn ganz einfach erstellt und verwaltet werden. Es können bis zu 256 RGB LEDs oder 768 einzelne LEDs verwaltet werden.

Über eine einfach zu bedienende Benutzeroberfläche kann man die gewünschten Funktionen Auswählen und Konfigurieren. Es steht eine Vielzahl von Befehlen zur Verfügung. Der Wichtigste ist sicherlich das „Belebte Haus“. Man kann aber auch Andreaskreuze, Signale, Ampeln, Sounds, Servos und vieles mehr mit nur einem Klick auswählen.

Die Effekte können automatisch gestartet oder über eine Zentrale aktiviert werden. Dazu werden momentan folgenden Protokolle/Busse unterstützt: **DCC**, **Selectrix** und der **Märklin CAN Bus**.

Das alles wird in einer übersichtlichen Tabelle verwaltet. Hier kann man ganz einfach Zeilen einfügen, verschieben, kopieren und nach seinen Bedürfnissen anpassen.

Die Konfiguration kann von dort direkt zum Arduino geschickt und ausprobiert werden. Die Programmerzeugung geschieht komplett im Hintergrund. Der Benutzer muss keinerlei Programmiererfahrung haben.

Wichtige Hinweise

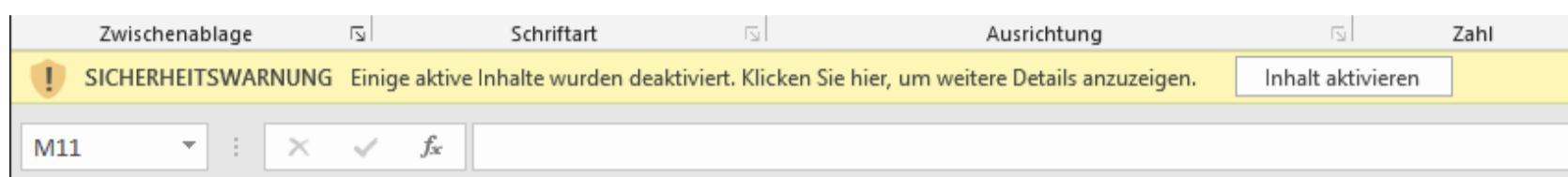
Excel-Versionen: Der Programm-Generator funktioniert nur mit Microsoft Excel in der Desktop-Version. Die Verwendung von OpenSource-Varianten¹⁾, sowie der „Web Version von Excel“²⁾, ist wegen der enthaltenen Makros nicht fehlerfrei möglich.

Aktuell werden folgenden Versionen unterstützt:

Excel 2013
Excel 2016
Excel 2019
Excel 365 Desktop-App

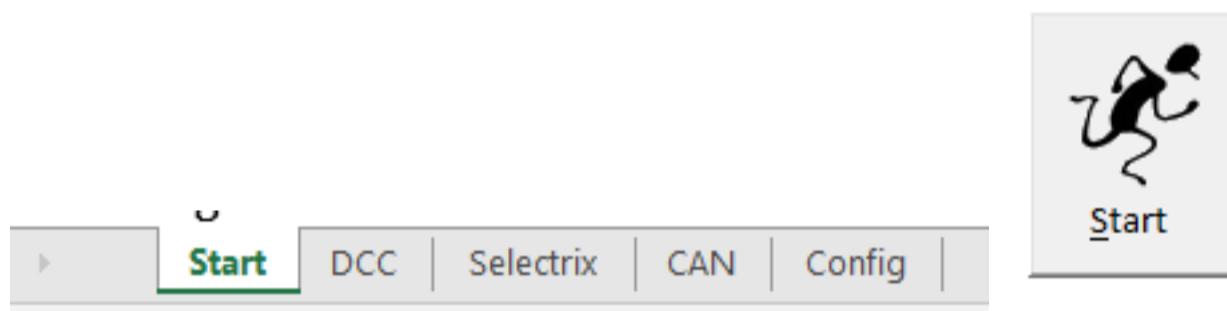
Erster Start

Beim ersten Start wird von Excel eine Warnung ausgegeben, dass sich aktive Elemente („Makros“) in der Tabelle befinden und das diese aus Sicherheitsgründen deaktiviert wurden. Um den Programm-Generator nutzen zu können ist das Aktiveren der Inhalte notwendig. Dazu einfach oben auf den Button „Inhalt aktivieren“ klicken.

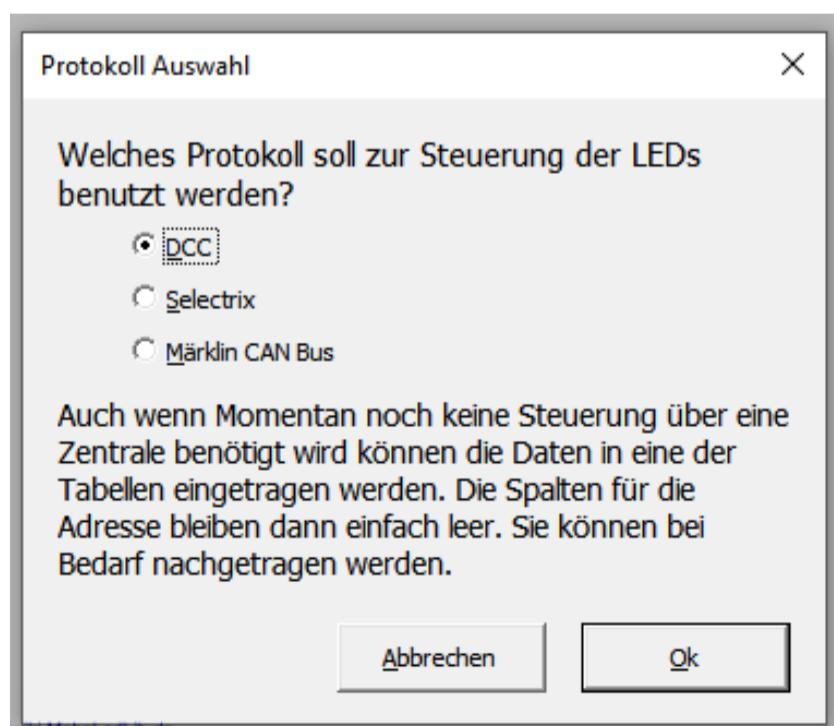


Auswahl des Systems zur Einbindung

Je nachdem welches Steuerungssystem eingesetzt wird kann man am unteren Rand, das gewünschte System auswählen oder einfach auf den Button „Start“ klicken.



Bei einem Klick auf den Button „Start“ öffnet sich ein Auswahlmenü, welches ermöglicht eines der verschiedenen Systeme zur Steuerung zu verwenden. Ein Wechsel auf ein anderes System ist jederzeit möglich. Da ich per DCC steuern möchte wähle ich dies hier aus oder klicke im unteren Bereich auf den Reiter „DCC“



Die Auswahl eines Steuerungssystems ist notwendig um die Beleuchtungen zu verwalten. Allerdings muss man keines besitzen um die MobaLedLib oder den Programm-Generator verwenden zu können.

Verwendung des Programm-Generators

Aufbau der Tabelle

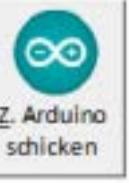
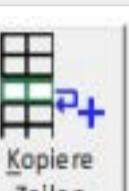
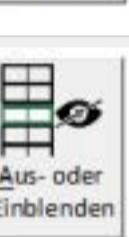
Das Hauptfenster des Programm-Generators teilt sich in drei Hauptbereiche auf.

- Dem grünen Bereich für alle Buttons welche die Steuerung des Generators übernehmen.
- Der Filterliste, welche sich in dem roten Bereich befindet.
- sowie dem orangenen Bereich in dem alle Befehle gespeichert und verwaltet werden.

Aktiv	Filter	DCC Adresse	Typ	Startwert	Beschreibung	Verteiler Nummer	Stecker Nummer	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs	InCnt	Loc InCh
✓	B01				Zeigt an, dass die LEDs angesteuert werden			RGB_Heartbeat(#LED)	0	1	0	0
✓	B02				Animiertes Haus mit 7 Räumen. Es sind zufällig zwischen 2 und 5 Zimmer beleuchtet.			House(#LED, #InCh, 2, 5, ROOM_DARK, ROOM_BRIGHT, ROOM_WARM_W, ROOM_TV0, NEON_LIGHT, ROOM_D_RED, ROOM_COL2)		7	1	0
✓	B04				6 (Gas) Straßenlaternen welche zufällig nacheinander angehen. Die Helligkeit der Lampen nimmt langsam zu. Manchmal flackert eine Lampe. Die 3. Lampe ist "defekt". Sie leuchtet schwächer. Diese Zeile verwendet RGB LEDs zu Testzwecken.			GasLights(#LED, #InCh, GAS_LIGHT, GAS_LIGHT, GAS_LIGHTD, GAS_LIGHT, GAS_LIGHT, GAS_LIGHT)		6	1	0
	B04				6 (Gas) Straßenlaternen welche zufällig nacheinander angehen. Die Helligkeit der Lampen nimmt langsam zu. Manchmal flackert eine Lampe. Die 3. Lampe ist "defekt". Sie leuchtet schwächer. Diese Zeile verwendet einzelne LEDs welche über WS2811 Module angesteuert werden.			GasLights(#LED, #InCh, GAS_LIGHT1, GAS_LIGHT2, GAS_LIGHT3D, GAS_LIGHT1, GAS_LIGHT2, GAS_LIGHT3)		2	1	0
✓	B05				Andreaskreuz mit zwei abwechselnd blinkenden Lampen. Diese Zeile verwendet RGB LEDs zu Testzwecken.			AndreaskreuzRGB(#LED, #InCh)		2	1	0
	B05				Andreaskreuz mit zwei abwechselnd blinkenden Lampen. Diese Zeile verwendet einzelne LEDs welche über ein WS2811 Modul angesteuert werden.			Andreaskreuz(#LED, C12, #InCh)		C1-2	1	0
✓	B08				Beispiel eines Baustellenlichts mit 6 Lampen. Diese Zeile verwendet RGB LEDs zu Testzwecken.			ConstrWarnLightRGB6(#LED, #InCh, 5, 255, 100 ms, 0 ms, 500 ms)		6	1	0
✓	B08				Beispiel eines Baustellenlichts mit 6 Lampen. Diese Zeile verwendet einzelne LEDs welche über WS2811 Module angesteuert werden.			ConstrWarnLight6(#LED, #InCh, 5, 255, 100 ms, 500 ms)		C1-6	1	0

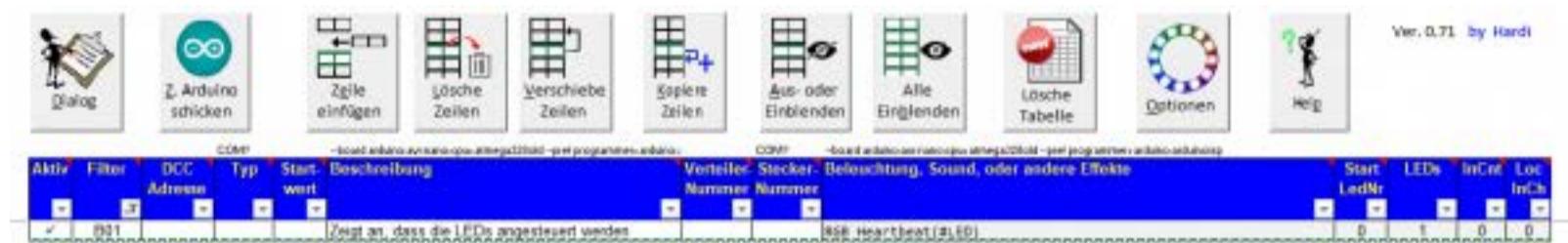
Erklärung der Buttons:

Button	Beschreibung
	Der wichtigste Button von allen. Hierhinter verbirgt sich die meiste Magie und ermöglicht das Erstellen der Beleuchtungs-, Sound und Bewegungseffekte.

 Z. Arduino schicken	wandelt die Befehle in C++-Code um und überträgt diese im Anschluss an den LED Arduino.
 Zeile einfügen	fügt eine neue leere Zeile über der aktuell markierten Zeile ein
 Lösche Zeilen	löscht wie die Aufschrift bereits erklärt alle markierten Zeilen. Die Löschung erfolgt aber erst nach einer erneuten Bestätigung in dem aufgehendem Fenster.
 Verschiebe Zeilen	Hiermit können eine oder mehrere Zeilen in der Reihenfolge verschoben werden. Die ausgewählten Zeilen werden oberhalb des grünen Balkens eingefügt und das Programm berechnet die Reihenfolge der LEDs neu. Dies ist hilfreich wenn man z.B.: einen zusätzlichen Verteiler in einer vorhanden Verkabelung hinzufügt oder diesen woanders platziert.
 Kopiere Zeilen	Hiermit lassen sich Zeilen kopieren für den Fall, das es z.B.: ein identisches oder ähnliches Beleuchtungskonzept für ein anderes Haus gibt.
 Aus- oder Einblenden	Hiermit lassen sich Zeilen aus- und auch wieder einblenden. Diese werden nicht gelöscht sondern nur versteckt.
 Alle Einblenden	Dieser Button zeigt alle Zeilen, die vorher versteckt wurden, wieder an.
 Lösche Tabelle	löscht alles in dem Programmreich und legt eine komplett neue Tabelle an. Das Programm fragt aber vorher ob es wirklich gewünscht ist. Durch eine Filterliste ausgeblendete Zeilen werden nicht gelöscht.
 Optionen	Öffnet das Menü für die Optionen, in dem man die Einstellungen für den LED Arduino und den DCC Arduino ändern kann. Es gibt auch die Möglichkeit, die Datei zu speichern oder die Beta Version der MobaLedLib zu installieren.
 Help	Öffnet die Hilfeseite

Filterliste

Die blauen Filterlisten ermöglichen es, die Anzahl der Zeilen zu verringern und nur bestimmte Zeilen anzuzeigen. Dies wird z.B.: gemacht um nur das Beispiel „B1“ beim ersten Aufrufen der Tabelle „DCC“, „Selectrix“ oder „CAN“ anzuzeigen.



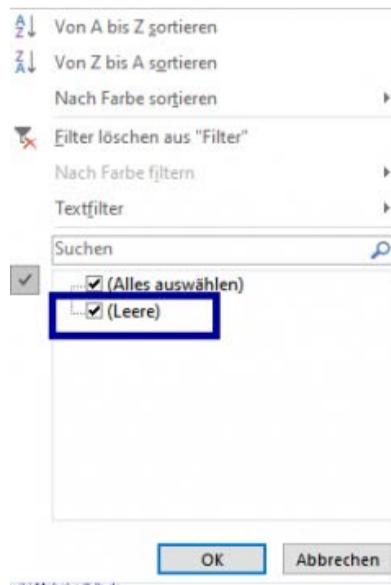
Erklärung der einzelnen Spalten

Bezeichnung	Erklärung
Aktiv	Mit dieser Spalte kann man eine Zeile (zu Testzwecken) deaktivieren. Zeilen welche mit dem „Aus- oder Einblenden“ Knopf oder dem Autofilter ausgeblendet sind werden ebenfalls nicht benutzt.
Filter	Diese Spalte kann zum Filtern nach bestimmten Gruppen benutzt werden. Für jede Gruppe kann hier ein gleicher Wert eingetragen werden. Dann kann man mit dem Autofilter bestimmte Gruppen aktivieren.
Adresse oder Name	Hier wird die DCC Adresse zwischen 1 und 10240 eingetragen. Achtung nicht alle Zentralen unterstützen Adressen > 9999. Alternativ kann hier ein Schalter (z.B. „SwitchB7“) oder eine selbst definierte Variable (z.B. „HausA“) eingetragen werden.
Typ	Typ des Eingangs: AnAus (Schalter) Rot (Taster) Grün (Taster)
Startwert	Definiert den Startwert des Eingangskanals nachdem die Versorgungsspannung eingeschaltet wurde. Er ist gültig bis die entsprechende DCC Nachricht empfangen wird. Ist die automatische Speicherung des Letztzustandes aktiviert kann durch Eingabe von '0' die Zustandsspeicherung für diese Zeile deaktiviert werden. Durch Eingabe von '*' wird die Zustandsspeicherung für diese Zeile erzwungen, sofern die Funktion einen Zustand speichern kann, die Standardeinstellung dies aber deaktiviert (z.B. alle Counter mit Timeout).
Beschreibung	Hier sollte ein beliebiger Text zur Dokumentation eingegeben werden damit man sich später wieder zurechtfindet. Mit Alt+Enter kann mit einer neuen Zeile begonnen werden.
Verteiler-Nummer	In dieser Spalte kann eine Nummer oder eine Bezeichnung des Verteilers eingetragen werden an dem die LEDs angeschlossen sind.
Stecker-Nummer	Hier kann die Nummer des Steckplatzes der Verteilerplatine eingetragen werden welche von der angeschlossenen Baugruppe benutzt wird.

Beleuchtung, Sound, oder andere Effekte	Hier werden mit einem Doppelklick oder dem „Dialog“ Knopf die Funktionen zum ansteuern der LEDs oder der anderen Verbraucher eingetragen. Achtung: Die Einträge sollten nur von Experten manuell verändert werden.
Start LedNr	Diese Spalte enthält die Startnummer der ersten LED dieser Zeile. Die Nummer ergibt sich aus der Zeilenposition und der Anzahl der vorangegangenen LEDs. Die Zahl kann nicht verändert werden.
LEDs	Hier wird automatisch die Anzahl der von dieser Zeile angesteuerten LEDs eingetragen. Achtung: Nicht manuell ändern.
InCnt	Sie enthält die Anzahl der lokal benutzten InCh Kanäle. Sie wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.
Loc InCh	Sie enthält die Anzahl der lokal benutzten InCh Kanäle. Sie wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.
LED Kanal	Benutzer LED Kanal. 0 = Standard 1 = Taster Wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.
Start Tast LED	LED Nummer im Taster Kanal. Wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.
Start LED G2	LED Nummer für den benutzerdefinierten Kanal 2. Wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.
Start LED G3	LED Nummer für den benutzerdefinierten Kanal 3. Wird automatisch vom Programm geschrieben. Achtung: Nicht manuell ändern.

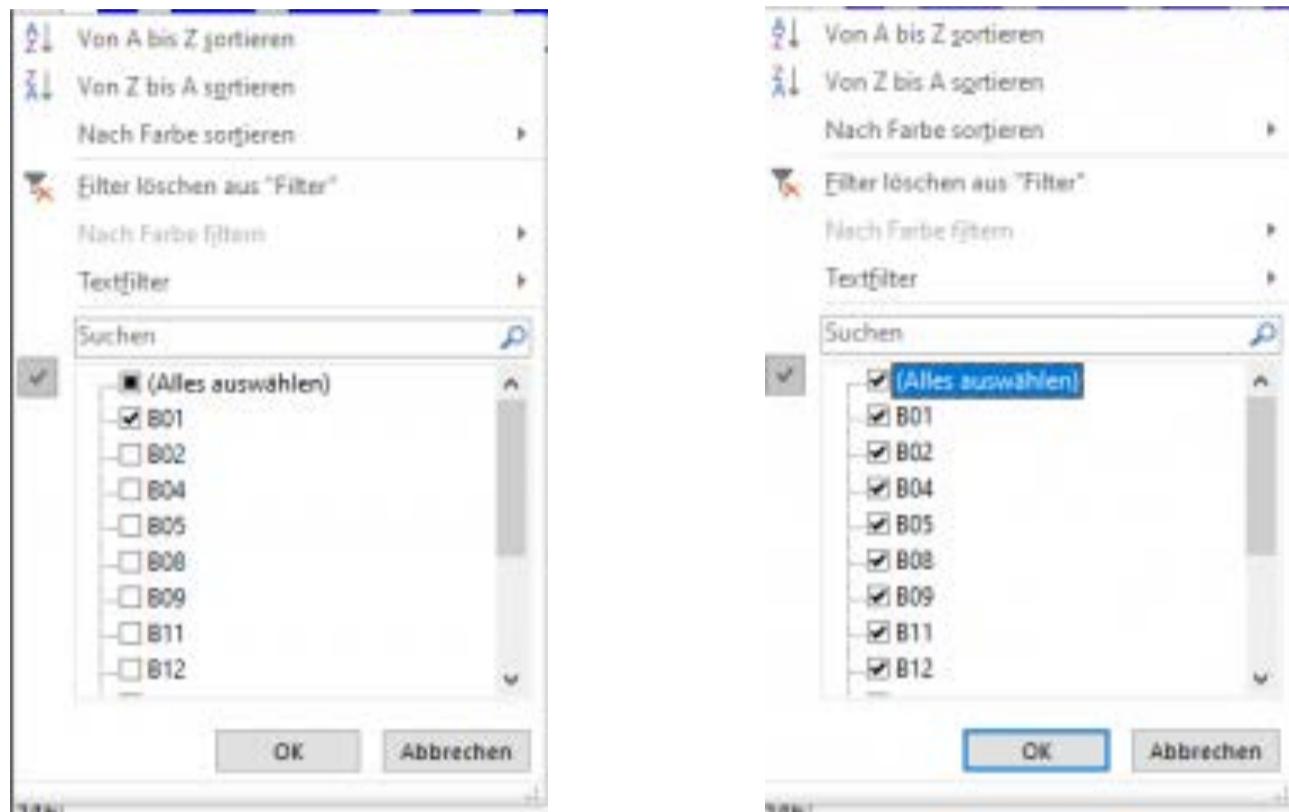
Eventuelle Probleme mit Filterliste

Bei einem Löschen von Zeilen oder dem ganzen Blatt, kann es vorkommen das nicht alles gelöscht wurde oder die Tabelle danach komplett leer ist. Um dieses Problem zu beheben bitte alle Filter löschen und das Kästchen (Leere) aktivieren.



Mitgelieferte Beispiele

Die Beispiele befinden sich ab sofort auf der Extraseite „Examples“. Beim ersten Aufruf der Tabelle ist nur ein Beispiel mit dem Namen „B01“ vorhanden. Um alle sehen zu können in der Filterliste den Filter auf „Alle auswählen“ setzen.



Danach wird eine längere Liste von Beispielen im Bereich für den Programmablauf angezeigt.

Wenn nicht wurden die Beispiele bereits aus dem Programm gelöscht.

Eine aktuelle und saubere Version des Programm-Generators kann jederzeit wieder heruntergeladen werden.

Das Erstellen von Programmen

Aufrufen des Dialogs

Der Anfang geht ganz schnell. Dazu oben einfach auf den Button „Dialog“  klicken.



Auswahl der Ansteuerung

Hier kann ausgewählt werden, ob der Effekt über eine Zentrale (DCC, Selectrix oder Märklin CAN) angesteuert wird oder dauerhaft aktiv ist.

Beispiele für Ansteuerung über eine Zentrale

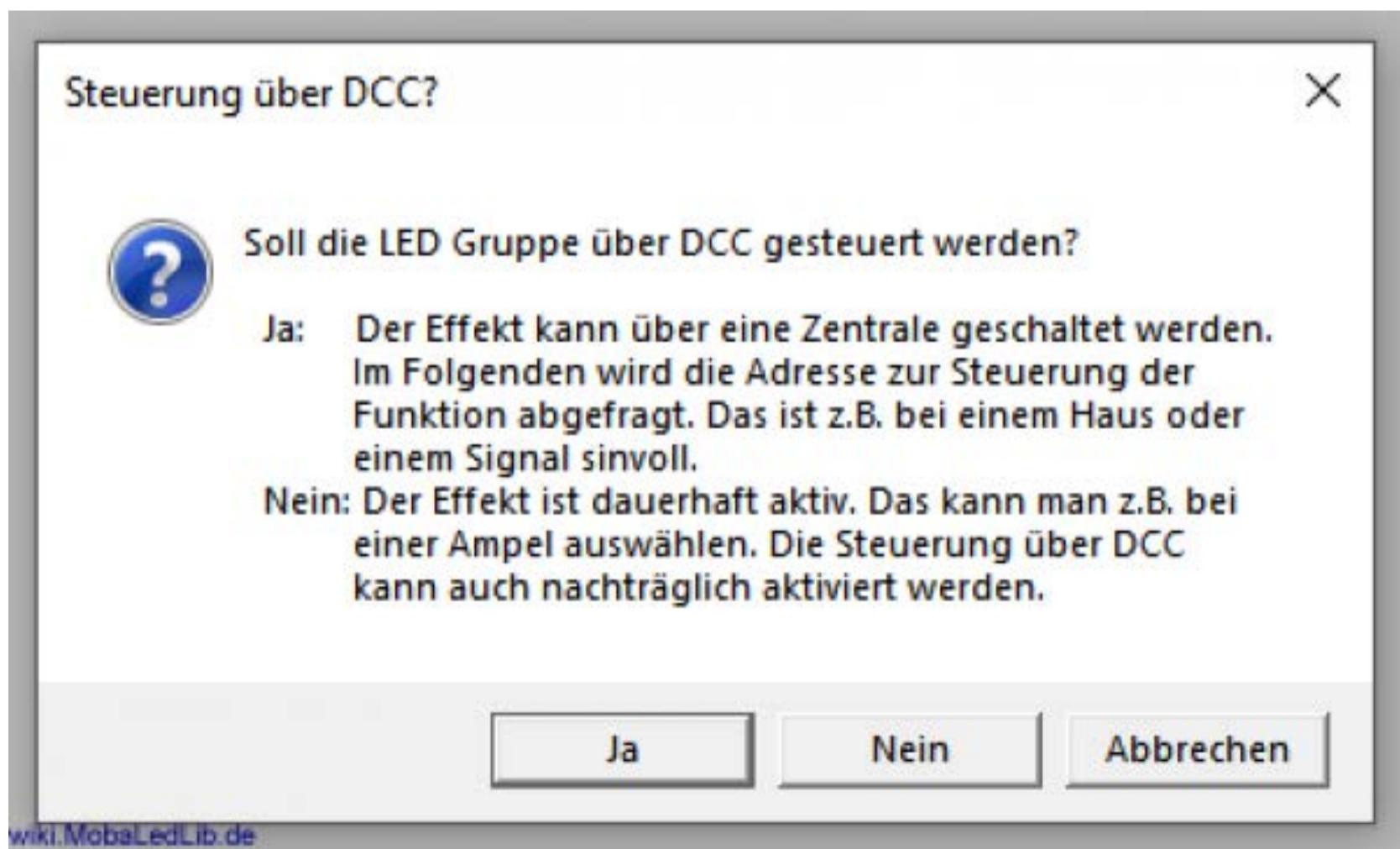
- Licht- oder Formsignale
- Beleuchtung von Häusern

- Soundeffekte wenn ein Zug im Bahnhof steht oder durchfährt
- Tagsüber Beleuchtung als normale Gaststätte, Abends Discobeleuchtung
- [Das brennende Haus](#) - Licht-, Sound und Raucheffekt

Beispiele für dauerhaft aktive Effekte

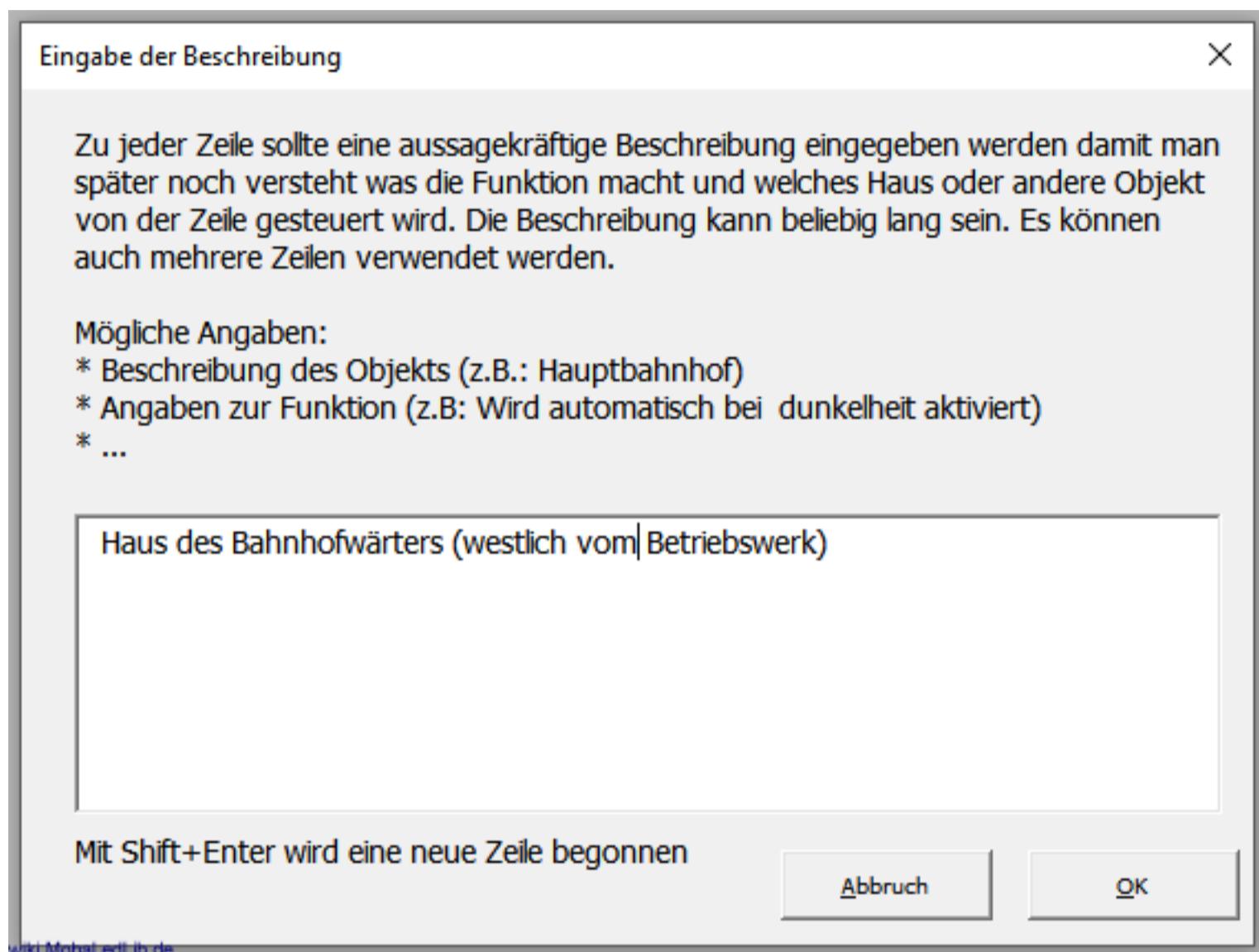
- Ampeln
- Schweißlicht im Bahnbetriebswerk
- Blitzlicht eines Fotografen bei Veranstaltungen
- Blinklichter auf hohen Gebäuden und [Windrädern](#)
- [Baustellenwarnbaken](#)
- [Bewegung mit der MobaLedLib](#)
- [Schornsteinfeger bei der Arbeit](#)

Alle Beispiele können natürlich auch dauerhaft aktiv oder von einer Zentrale gesteuert werden. Möglich ist auch eine Verwendung der „Push Buttons“-Erweiterung für die MobaLedLib, um Effekte per Druckknöpfe welche auf der ganzen Anlage verteilt werden können, auszulösen. Ich habe hier erst einmal auf „Ja“ geklickt um diese Gruppe über DCC zu steuern.



Vergabe eines aussagekräftigen Namens

In diesem Dialogfenster, sollte dem Effekt ein aussagekräftiger Name vergeben werden. Dies ermöglicht es später diesen leichter wiederzufinden. Wenn alles nur „Beleuchtung Haus“ heißt, kennt sich später niemand mehr aus.



Beispiele für gute Bezeichnungen

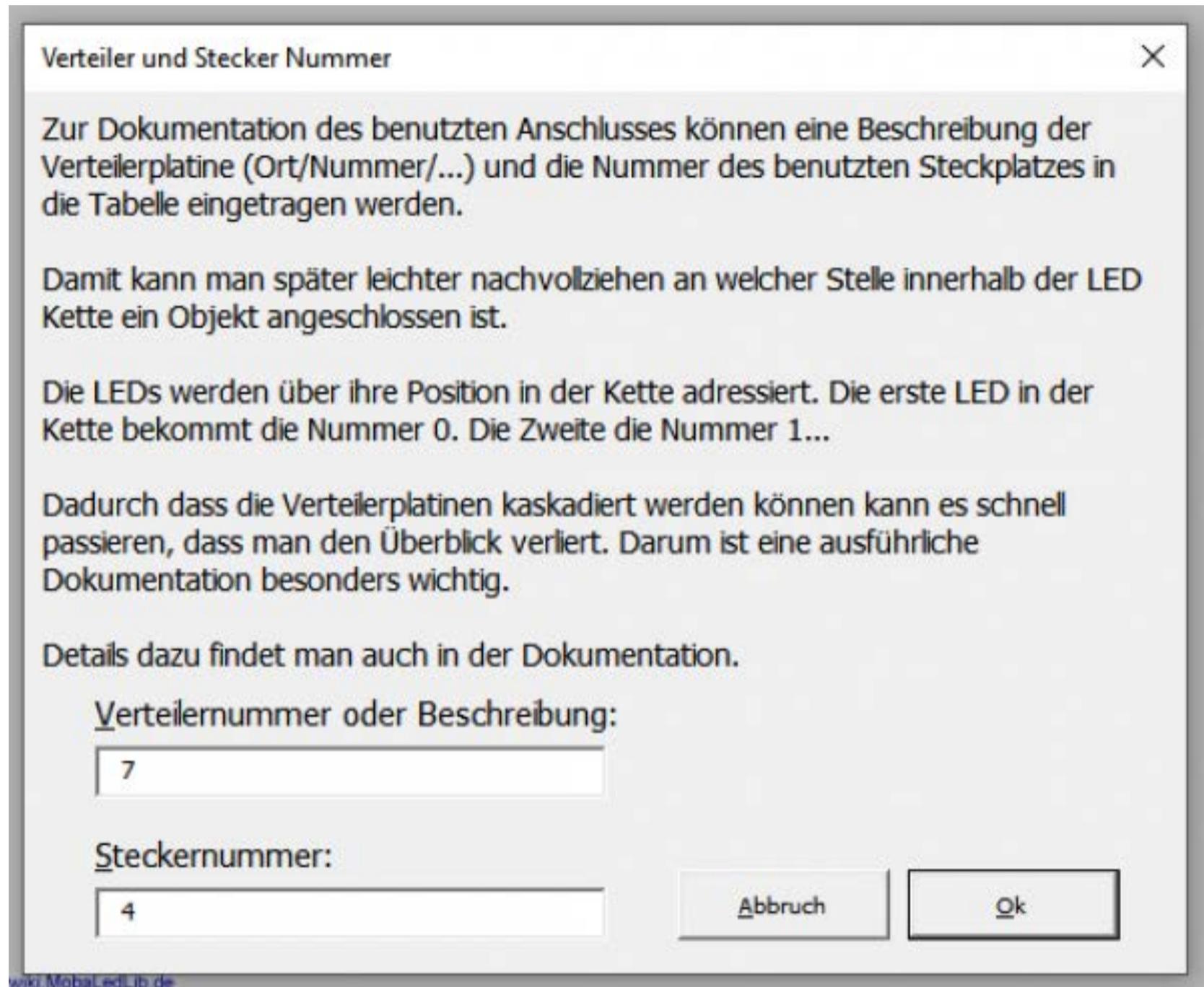
- Kirche am kleinen Marktplatz
- Bahnhofs „Meckershäusen“
- Gasthof „zum schnellen Koch“
- Wohnhaus auf dem Berg
- Straßenlaternen Hauptstraße Meckershäusen Südseite
- Beleuchtung Bahnsteig Hauptbahnhof

Beispiele für schlechte Namen

- Haus 1, Haus 2, Haus 3
- Straßenlaternen 1, Laternen 2

Vergabe der Verteiler- und Steckplatznummer

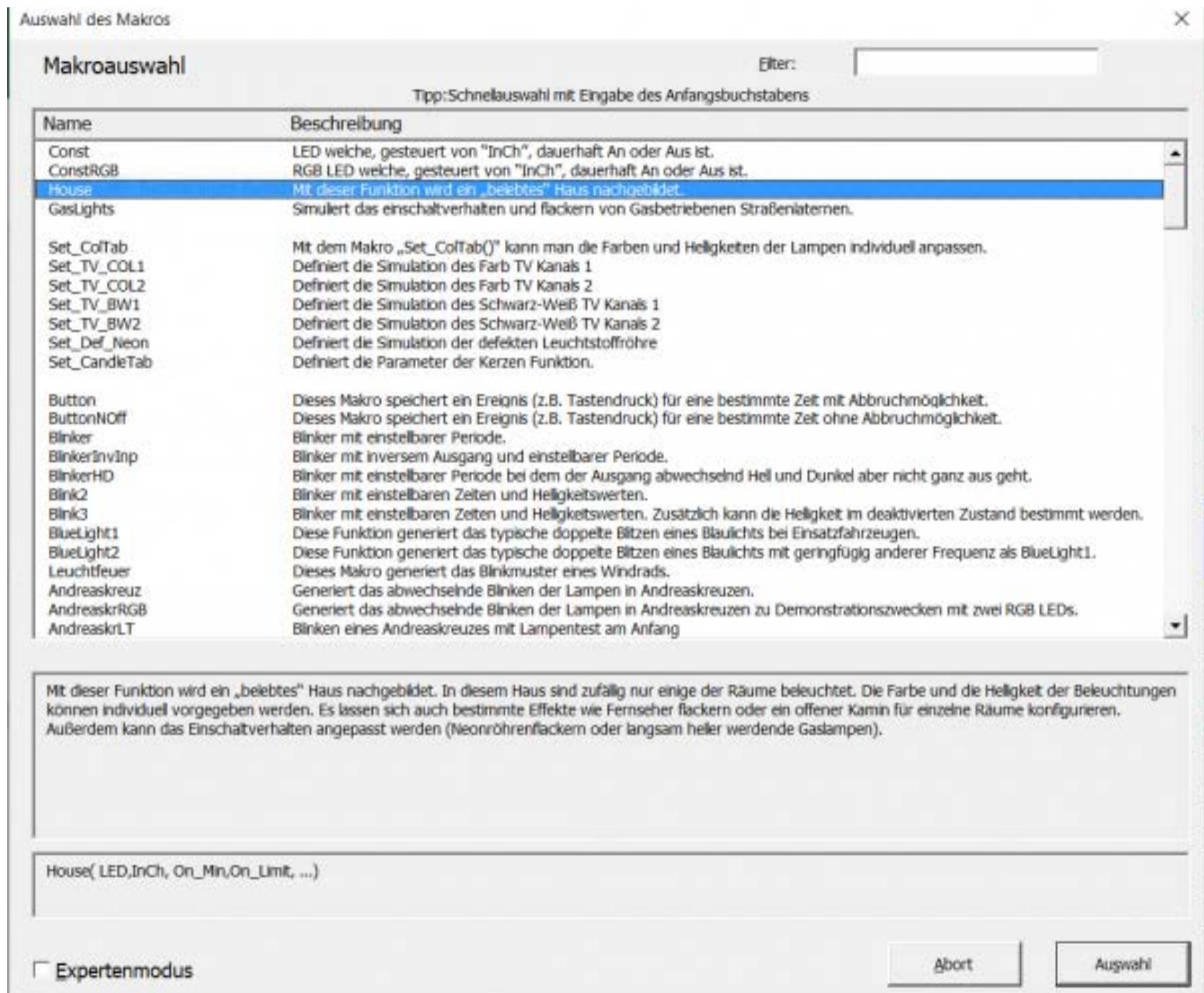
Das nächste Fenster wünscht die Vergabe einer Verteiler- und Steckplatznummer. Dies ermöglicht die Übersicht zu behalten, wenn später z. B. das Haus nicht mehr links auf der Anlage steht sondern rechts oder ein zusätzlicher Verteiler für das neue Dorf hinzugefügt wird. Eine Eingabe ist aber nicht zwingend notwendig.



Auswahl des Effekts

Die nachfolgenden Effekte sind in der Standardeinstellung verfügbar.

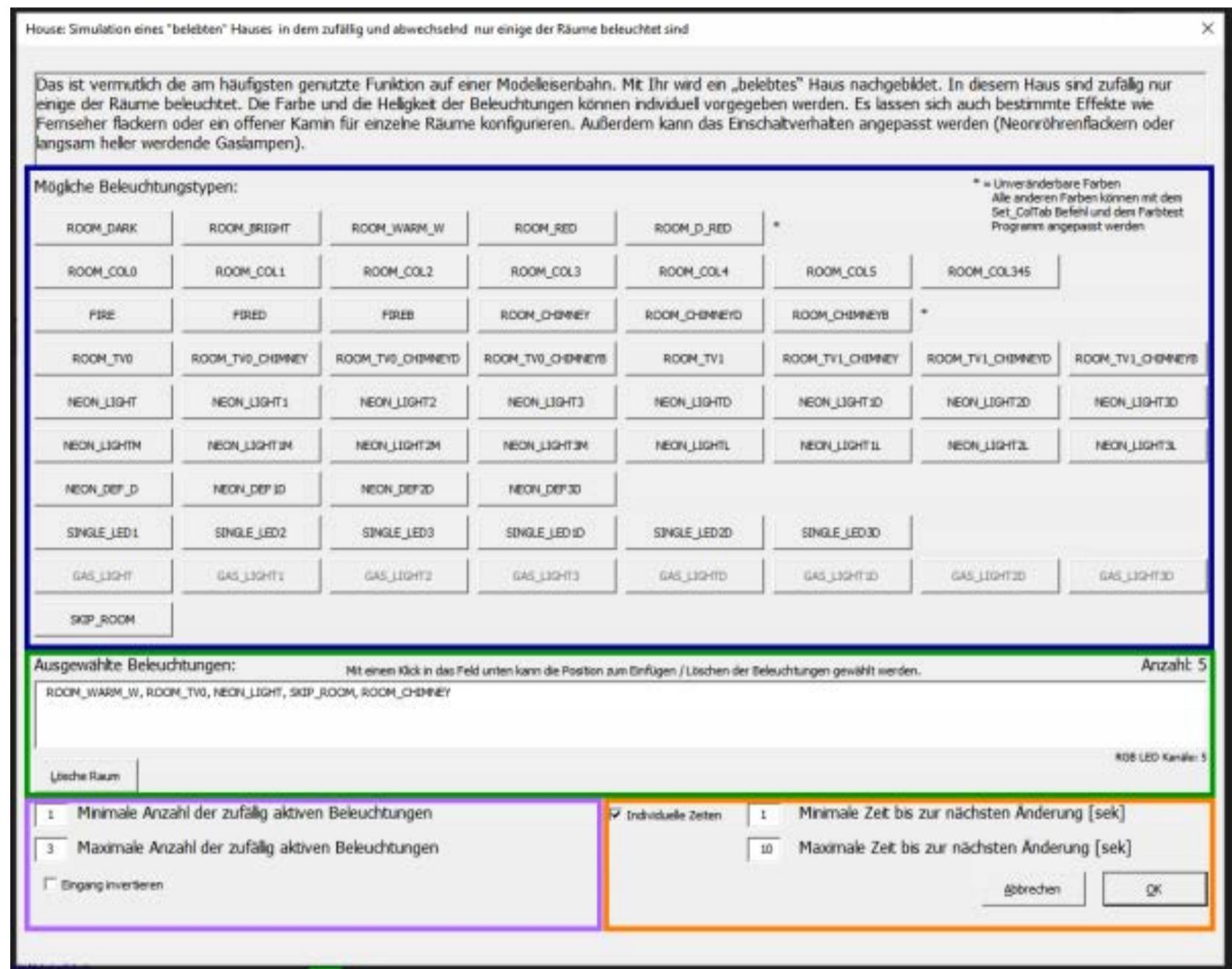
Die gesamte Liste der Effekte mit ausführlichen Erklärungen kann man im Wiki nachschlagen.



Konfiguration des Effekts

Auswahl eines Beispiels (belebtes Haus)

Für unseren ersten Test wählen wir mit dem Select-Knopf das „House“ aus und erhalten die folgende Seite zur weiteren Auswahl der Funktionen:



Über das Tastenfeld „mögliche Beleuchtungstypen“ (blauer Rahmen) können unterschiedliche Beleuchtungen für die Räume eines Hauses ausgewählt werden. Wenn in einem Gebäude fünf Räume beleuchtet sind, müssen fünf Beleuchtungen ausgewählt werden. Die ausgewählten Effekte werden in dem grünen Bereich angezeigt. Über die Taste „Lösche Raum“ lassen sich Räume löschen um eine andere Beleuchtung auszuwählen.

In dem violetten Rahmen lässt sich einstellen wie viele Beleuchtungen mindestens und wie viele maximal gleichzeitig leuchten sollen. Das Kästchen „Eingang invertieren“ dreht die Logik des verbunden Schalters /Steuerkanals um. Dies wird z.B. in der Disco verwendet um tagsüber eine normale Gaststätte zu haben und Nachts über die gleichen LEDs die Beleuchtung einer Disco zu simulieren.

Durch Anklicken der Option „Individuelle Zeiten“ (orangener Rahmen) lassen sich die Zeiten für den Beleuchtungswechsel den eigenen Bedürfnissen entsprechend anpassen. Für Testzwecke bietet es sich an, die „Maximale Zeit bis zur nächsten Änderung“ auf 5-10 [Sec] zu setzen.

Mit einem Klick auf „OK“ werden die Einstellungen übernommen und in dem Programmbereich angezeigt.

Aktiv	Filter	DCC Adresse	Typ	Start wert	Beschreibung	Vorlieger Nummer	Stecker Nummer	Beleuchtung, Sound, oder anderes Effekte	Start LedNr	LEDs	InCat	Loc InCh
✓					Taster auf der Hauptplatine für DCC-Simulation			#define TEST_TOGGLE_BUTTONS			0	0
✓	B 01				RGB_Heartbeat(#LED)	0	0	RGB_Heartbeat(#LED)	0	1	0	0
✓		1	AnAus	1	Haus des Bahnhofwärters (westlich vom Betriebswerk)	7	4	HouseT(#LED, #InCh, 1, 3, 1, 2B, ROOM_WARM_H, ROOM_TV_B, WARM_LIGHT, SKIP_ROOMS, ROOM_CHIMNEY)	1	5	1	0
✓					Heartbeat letzte LED	999	0	RGB_Heartbeat(#LED)	6	1	0	0

Wenn man möchte, können im Anschluss weitere Effekte angelegt werden.



In dem Bild oben ist erkennbar, das vier Zeilen Aktiv geschaltet sind. In der obersten Zeile ist die Verwendung der Testbuttons von der Hauptplatine eingetragen. In der dritten Zeile steht das gerade erstellte Haus mit den getroffenen Einstellungen. Das ist an dem Haken in der Spalte „Aktiv“ zu erkennen. Zeilen können mit einem einfachen Mausklick in die Spalte aktiviert und deaktiviert werden. Nicht-aktive Zeilen werden nicht zum Arduino übertragen. Der grau hinterlegte Bereich ist automatisch befüllt worden und kann bzw. sollte nicht geändert werden. Über die roten Dreiecke in den Feldern können zur weiteren Erklärung Tooltips aufgerufen werden.

Die zweite Zeile mit dem „Heartbeat 1. LED“ sollte in jedem Projekt verwendet werden. Damit wird die erste LED in der Kette genutzt, um zu signalisieren, dass die Übertragung des Programms an den Arduino erfolgreich war und das System „lebt“. Falls die folgenden LEDs dann trotzdem nicht so arbeiten wie erwartet, hat man

irgendwo in der Auswahl für die LEDs einen Fehler gemacht oder in der Verdrahtung der LEDs liegt ein Fehler vor.

Zusätzlich kann auch auf der Hauptplatine die letzte LED installiert und für die vierte Zeile verwenden werden. Wenn auch diese im Regenbogenfarben blinkt, kann man davon ausgehen, das die Verkabelung stimmt und alles funktioniert.

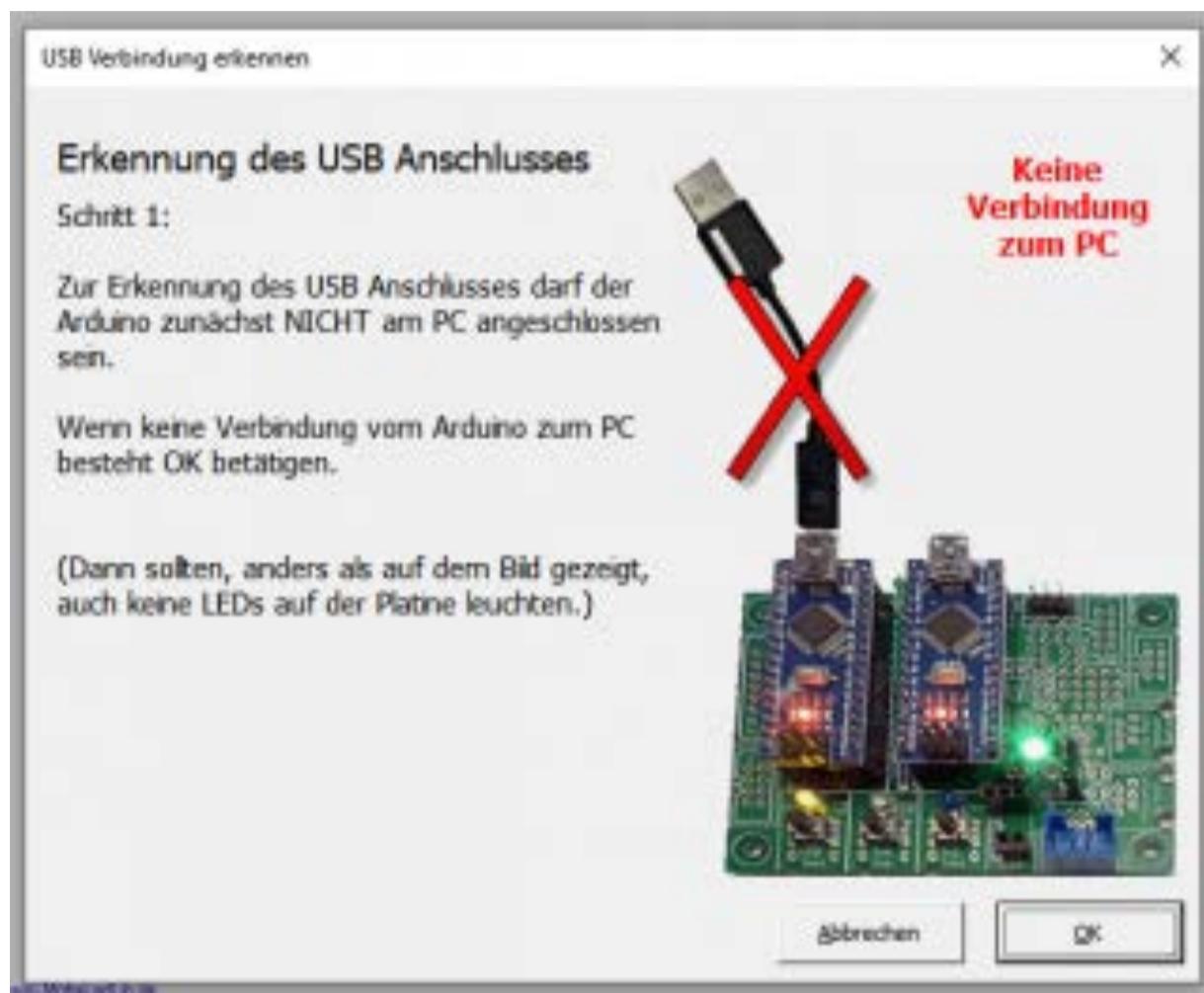
Besonderheit beim ersten Hochladen

Beim ersten Hochladen eines Programmes mit DCC-Steuerung wird gefragt, ob das Programm für den DCC-Arduino schon auf diesem installiert wurde. Sollte es schon erledigt sein, kann man das Dialogfenster mit einem Klick auf „Ja“ einfach schließen und der Upload zum LED-Arduino wird nun gestartet. Wenn man es noch nicht gemacht hat, kann man die Schaltfläche „Installieren“ auswählen, dann erfolgt die Installation des notwendigen Programmes. Dies ist nur einmalig notwendig oder wenn eine neue Version der Software veröffentlicht wird.

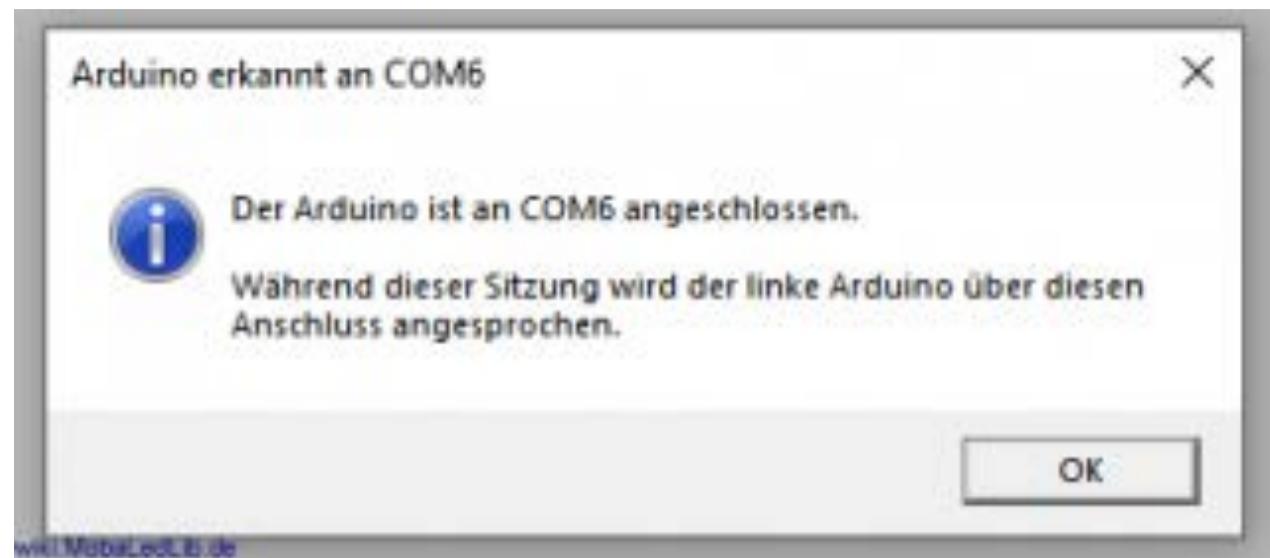


Im nächsten Schritt wird unsere Auswahl zum ARDUINO geschickt.

Beim ersten Sendeversuch erfolgt die Aufforderung, den benutzten COM-Port festzulegen. Einfach den Anweisungen folgen und anschließend den „Z.Arduino schicken“ Knopf drücken.



Wenn alles geklappt hat, sieht es so aus.



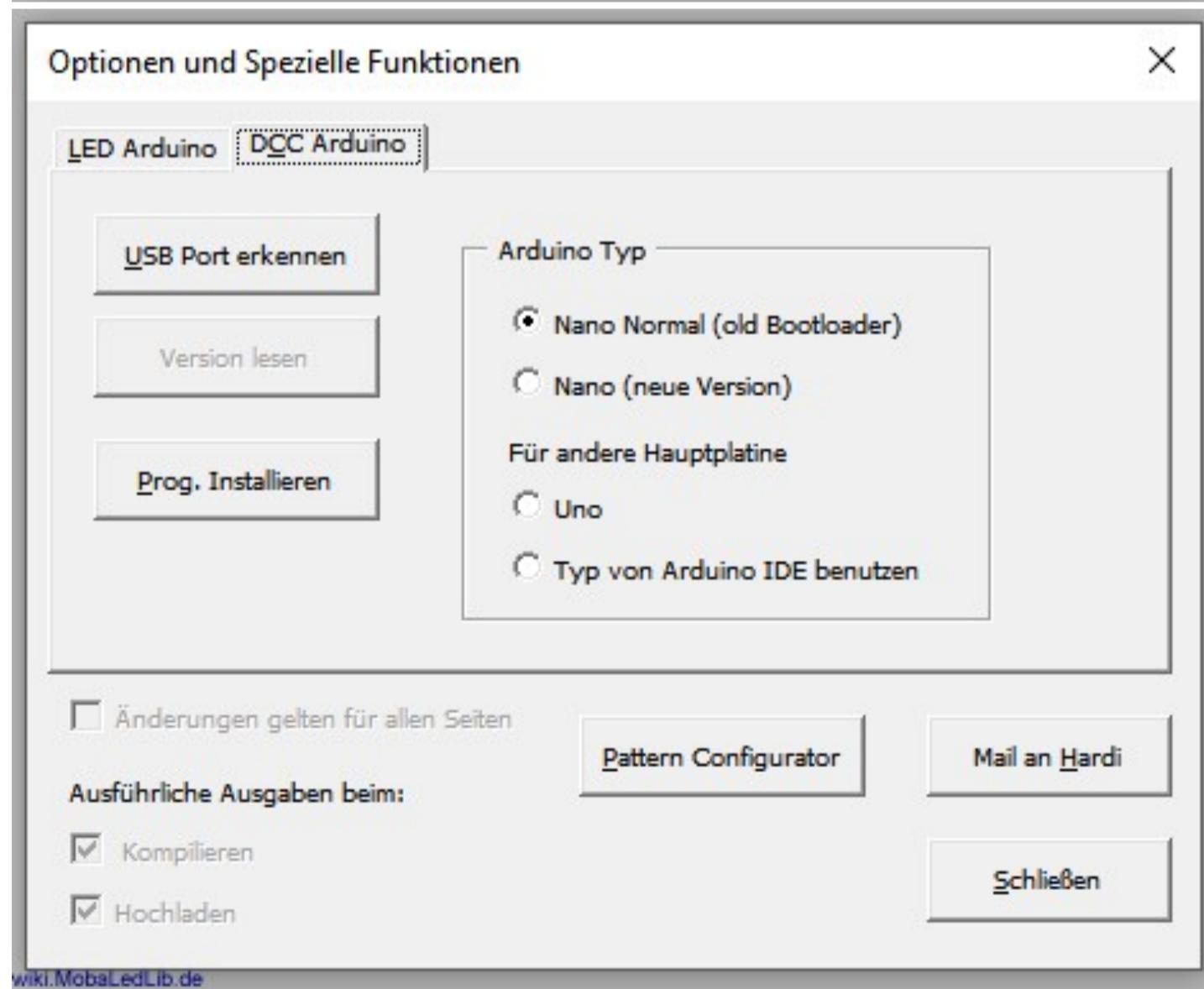
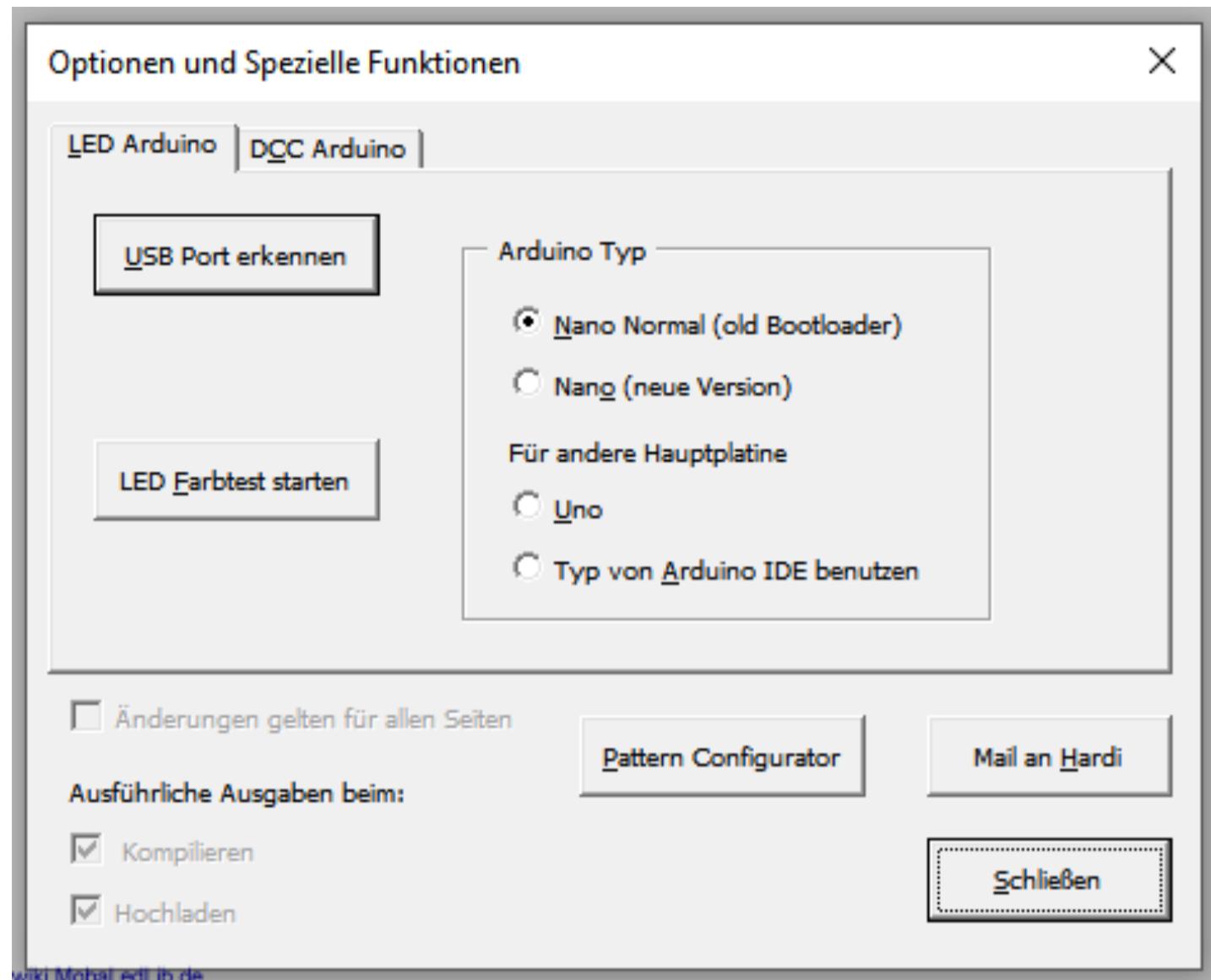
Bei Problem sieht es so aus.



Der Vorgang zum Erkennen des COM-Ports kann auch über „Optionen“ → „USB Port erkennen“ angestoßen werden.

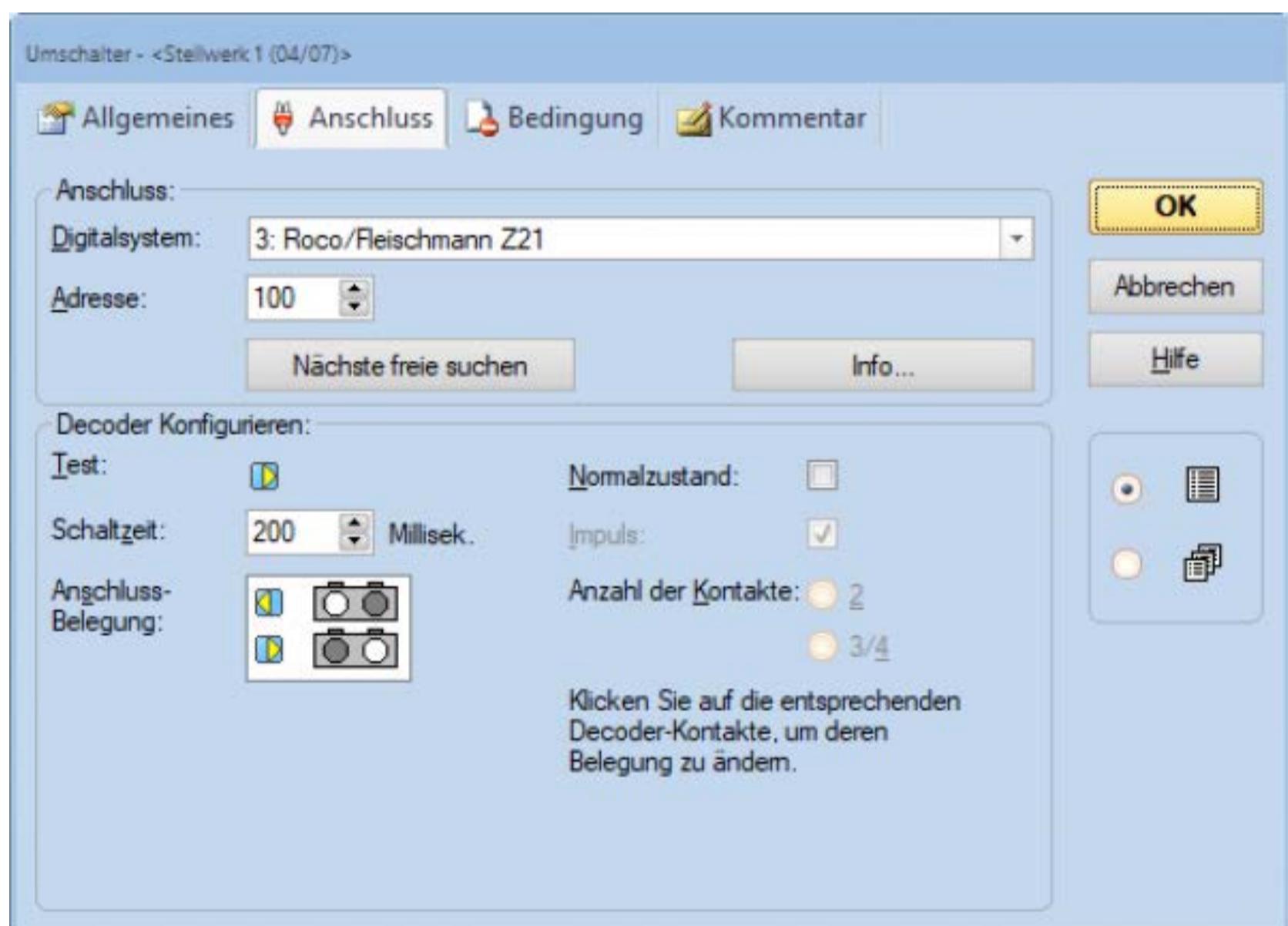


Für eine fehlerfreie Übertragung zum Arduino muss im Auswahlmenü unbedingt der tatsächlich genutzte Typ eingetragen werden. Für Arduino Nano Clones i.d.R. „Nano Normal (old Bootloader)“ auswählen. Ist der Arduino Typ nicht aufgeführt, sollte der Punkt „Typ von Arduino IDE benutzen“ ausgewählt werden und hier übernommen werden. Die Auswahl des Arduino-Types muss für den LED-Arduino und für den Steuer-Arduino (DCC, Selectrix, LocoNet,...) separat erfolgen.



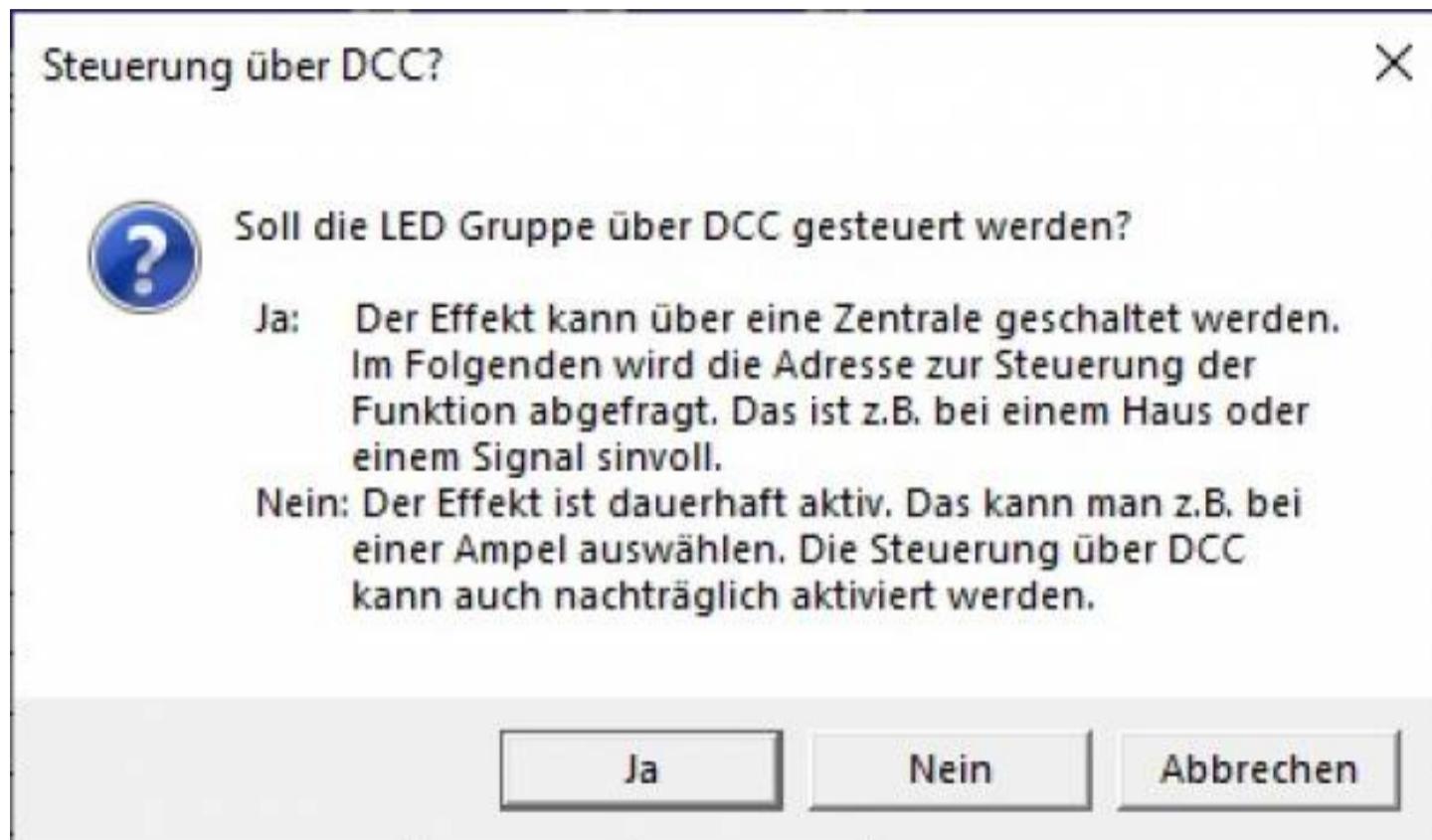
Ansteuerung der MobaLedLib durch Traincontroller

Will man einzelne Effekte (MobaLedLib-Makros) durch Befehle von Traincontroller ein- bzw. ausschalten, dann muss man diese als Schalter, Signal oder Weiche in TC einrichten. Ich habe das über Umschalter gemacht z. B. hier ein Schalter mit der Adresse „100“:

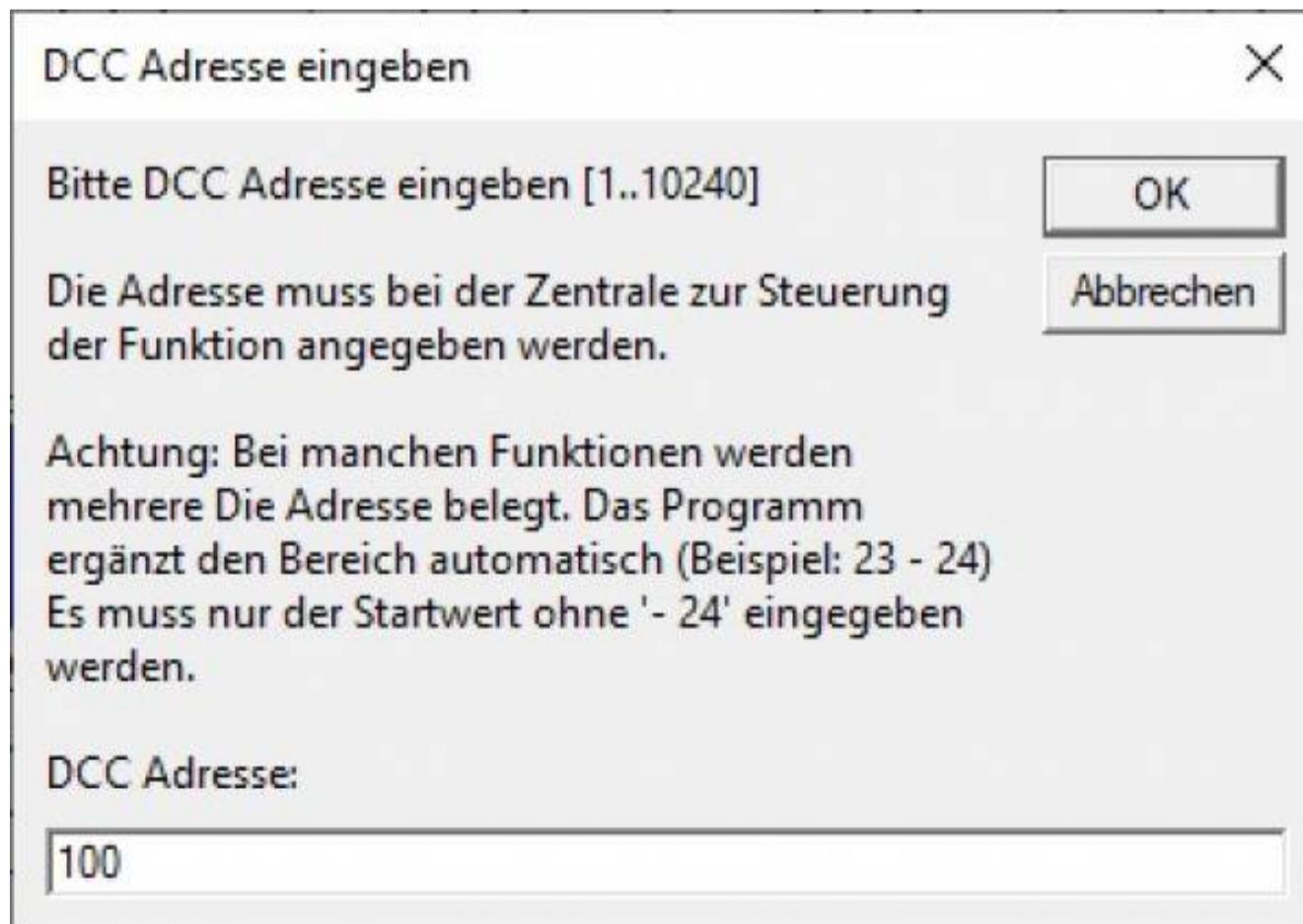


Bei der Programmierung der MobaLedLib-Macros kann man jetzt diese Adresse mit angegeben, entweder als Antwort auf die Frage nach einer Adresse oder direkt in der Excel-Tabelle

Beispiel für die Abfrage:



Hier mit „Ja“ beantworten



Hier dann die DCC Adresse 100 eingeben

Auswahl des Eingabe Typs



Ein Steuereingang von der Zentrale kann entweder als Ein- / Ausschalter oder als Taster (Rot/Grün) interpretiert werden.

- * Ein- / Ausschalter verwendet man z.B. bei Häusern.
- * Taster werden Beispielsweise bei Sounds eingesetzt.

Beim Taster können zwei verschiedene Aktionen über einer Adresse ausgelöst werden.

Anstelle von "Rot" und "Grün" kann je nach Zentrale auch "Abbiegen" oder "Gerade", 0 oder 1 angezeigt werden.

- An / Aus
 Rot
 Grün

"Rot" und "Grün" ist symbolisch zu verstehen und hat nichts mit den LED Farben zu tun.

Eingabetyp Ein/Aus ist für Schalter und Effekte, bei Signalen kann es auch „Rot“ oder „Grün“ sein. Da ich in TC einen Umschalter definiert habe klicke ich hier z.B. „Grün“ an. Damit wird der Effekt bei Schalterlage „Grün“ eingeschaltet und bei „Rot“ wieder aus.

Definition des Startwerts

Startwert des Eingangs eingeben

OK

Abbrechen

Der Startwert bestimmt das Verhalten nach dem Einschalten in Verbindung mit DCC, CAN oder Selectrix.

Normalerweise sind die Funktionen beim Start deaktiv. Erst wenn der erste DCC Einschaltbefehl von der Zentrale kommt wird die Zeile aktiviert.

Wenn eine bestimmte Funktion bereits beim Einschalten der Anlage einen definierten Wert haben soll kann das über den Startwert vorgegeben werden. Die meisten Funktionen haben einen Eingang mit dem sie Ein- oder Ausgeschaltet werden. Hier wird eine 1 zum Einschalten angegeben.

Bei Funktionen mit mehreren Eingängen (z.B. Signale) ist der Wert ist Bitkodiert. Hier wird der erste Eingang mit einer 1, zweite Eingang mit einer 2 und der dritte Eingang mit einer 4 aktiviert.

Startwert: (Keine Eingabe wenn nicht benötigt)

Der Startwert sollte leer gelassen werden. Traincontroller kümmert sich um die richtige Einstellung.

Alternativ kann man diese Eingaben auch direkt in der Excel-Tabelle vornehmen:

Aktiv	Filter	Adresse oder Name	COMP: Typ	Start Wert	Beschreibung	Verteiler Nummer	Stecker Nummer	Beleuchtung, Sound, oder andere Effekte	Start LED4	LED4 Wert	Ende LED4
✓					Zeigt an, dass die LEDs angesteuert werden.			RGB_Heartbeat [#LED]	0	1	0
✓		100 AnAus [■]			TestDCC			Const [#LED, C_ALL, #InCh, B, 127]	1	1	1

Die Adresse wird in die Spalte „Adresse oder Name“ eingetragen. Der Typ in die Spalte „Typ“ eingeben.

Anschließend einfach wieder oben auf den Knopf „zum Arduino übertragen“ klicken und nach einiger Zeit ist das Programm fertig übertragen. Jetzt kann mittels des Umschalters in Traincontroller der entsprechende Effekt manuell ein- bzw. ausgeschaltet werden.

Nutze ich den „Fahrplan“ in Traincontroller um zeitliche Abläufe automatisch aufzurufen, dann muss ich jetzt noch entsprechende Makros in TC anlegen. Dies könnte zum Beispiel ein Makro „Bahnhof Licht ein“ und ein Makro „Bahnhof Licht aus“ sein. In den Eigenschaften des Makros hinterlege ich dann den entsprechenden Umschalter und die richtige Schalterstellung. Dieses Makro kann ich dann im „Fahrplan“ zu einer definierten Zeit aufrufen. Z. B. um 8.00 Uhr das Makro „Bahnhof Licht aus“ (damit es ausgeschaltet wird falls es beim Start der Bahnhofsuhrt eingeschaltet war), um 18.00 dann wieder das Makro „Bahnhof Licht ein“ und so weiter. Das funktioniert hervorragend!

Ideen zum Schalten von Häusern

Wenn ich mehrere Häuser oder sogar eine kleine Stadt beleuchtet habe, dann gibt es mehrere Möglichkeiten, diese per DCC einzuschalten:

Ich kann natürlich jedes einzelne Haus einzeln mit einer eigenen DCC-Adresse ein -/ ausschalten. Aber bei Straßenzügen oder bei einer kleinen Stadt ist das natürlich umständlich.

Da das „belebte Haus“ nicht unbedingt das physikalisch vorhandene Haus abbilden muss kann ich ein „belebtes Haus“ mit soviel Räumen einrichten wie ich LEDs in mehreren zu schaltenden Häusern habe. Somit kann ich mit einer DCC-Adresse z. B. ein „belebtes Haus“ mit 16 Räumen (= tatsächlich sind das dann 4 Häuser à 4 Räume) anlegen. Diese Vorgehensweise ergibt natürlich gewisse Schwierigkeiten, wenn ich in die entsprechende LED-Kette neue Häuser/LEDs einfüge.

Alternativ kann ich im „Programm_Generator“ jedes Haus einzeln anlegen und auch entsprechend benennen. Lediglich bei der DCC-Adresse gebe ich bei allen gleichzeitig einzuschaltenden Häusern die gleiche DCC-Adresse ein. Somit werden alle Häuser, die die gleiche DCC-Adresse haben beim Schalten in Traincontroller ein- bzw. ausgeschaltet.

Und jetzt viel Spaß beim Ausprobieren!

Danksagung

Ich möchte mich ganz herzlich beim „Erfinder“ der MobaLedLib - Hardi - bedanken. Außerdem ein großes Dankeschön an alle, die mit der MobaLedLib zu tun haben:

Alfred - für das Versenden der Platinen

Dominik - für das tolle Wiki

. . . und alle anderen, die ich namentlich noch nicht kenne, die aber zur Entwicklung der MobaLedLib viel beigetragen haben. Außerdem bedanke ich mich bei allen, die regelmäßig im Stummiforum Rede und Antwort bei Problemen stehen. Ihr leistet einen super Support!

Vielen Dank!

Bezugsquellen

Die im Folgenden angegebenen Bezugsquellen sind nur Beispiele. Die Teile gibt es sicherlich auch bei vielen anderen Herstellern.

- WS2812 LEDs schnell: <https://www.amazon.de/Kuman-100pcs-WS28...2BHS6HGA50>
- WS2812 (Billiger aber lange Lieferzeit): <https://www.aliexpress.com/item/3269459...4c4d1oivPG>
- WS2812 LEDs mit angelöteten Kabeln: <https://de.aliexpress.com/item/19410663...4c4dXk8Cxy>
- WS2811 Module zum ansteuern normaler LEDs: <https://www.aliexpress.com/item/3275599...4c4d1R2J3z>
- 4-Polige Wannenstecker: #133: <https://www.stummiforum.de/viewtopic.php...&start=132>
- Sound Modul JQ6500: <https://de.aliexpress.com/item/32712836075.html>
- Sound Modul MP3-TF-16P: <https://de.aliexpress.com/item/32919672331.html>
- 64 WS2812 LEDs für Tests am Schreibtisch: <https://www.amazon.de/AZDelivery-Matrix...73&sr=8-19>
- Momentan sind die LEDs bei diesem Lieferanten vergriffen und bei Anderen sind nur noch wenige Vorrätig.
- => Google „WS2812 64 LEDs“
- Platinen: eMail an Alf (Stumminame: aftpriv): LedLib@yahoo.com
- Siehe [Verfügbare Platinen und Preisliste \(#500\)](#) im Wiki

Links

Und ganz zum Schluss noch einige Links:

MobaLedLib im Stummiforum:

<https://www.stummiforum.de/viewtopic.php?f=7&t=165060>

MobaLedLib-Wiki:

<https://wiki.mobaledlib.de>

Jochem Heinen

Alte Landstraße 44
53902 Bad Münstereifel
Telefon 02253 / 930166
Telefax 02253 / 952323

Rückfragen zu dieser Schrift bitte nur per eMail an:
jochem@familieheinen.name