Programmierung von Attinys für Servo oder Charlieplexing

Aufgabe/Ziel

Programmierung eines ATTiny85 für die Verwendung in der 510DE-Servo LED WS2811 Platine.
Diese Platine kann in 2 unterschiedlichen Bestückungsvarianten verwendet werden:

- **Variante 1** Ansteuerung von 12 LEDs mit Charlieplexing/Multiplexing-Schaltung
- **Variante 2** Ansteuerung von 3 Servos

Die Programmierung erfolgt mittels der Platine „400-Attiny-Programmer“ in der Standard Bestückung, wie sie in der Wiki Bauanleitung beschrieben ist. Softwaremäßig stehen für jede Bestückungsvariante jeweils 2 Möglichkeiten zur Auswahl:

1. Pattern_Configurator by Hardi
2. Arduino Programmiersoftware

Hardware

1. Arduino UNO
2. 400DE-Attiny-Programmer Platine in der Standardbestückung (Achtung auf die Ausrichtung der 6 LED´s)
3. IC ATTiny85 im DIL8 Gehäuse
4. Je nach Verwendungszweck
    1. 510DE-Servo LED WS2811 Platine in der Variante I. **Eine LED bestückte Charlieplexing-Testplatine (befindet sich auf der 400DE Platine) oder Viessmann Multiplex Signale**
    2. 510DE-Servo LED WS2811 Platine in der Variante II. **3 Servos**
5. Zur Festlegung der Servo-Endpositionen die 100DE-MLL Masterplatine
6. USB-Kabel zum Verbinden des UNO und der MLL Masterplatine mit dem PC.
7. Verbindungskabel von einer 200DE-Verteilerplatine oder der 100DE-Masterplatine.
8. Je nach Anzahl der Servoplatinen und angeschlossenen Servos, ein dem Stromverbrauch angemessenen, zusätzliches 5V Netzteil.
   weitere Info dazu: Sicherheit MobaLedLib, Stromversorgung

Software

1. MS Excel Version ab 2010 (mit Excel 2007 ergaben sich einige Probleme) empfohlen neuere

Versionen.
ACHTUNG: Mit anderen Tabellenkalkulationsprogrammen ergaben sich ebenfalls Probleme und die Makros funktionieren nicht richtig. Infos
2. MobaLedLib by Hardi (In der derzeit aktuellen Version)
3. Arduino 1.8.12

Ablauf

Wie bereits erwähnt gibt es softwaremäßig 2 Möglichkeiten den ATTiny85 zu programmieren.
Im Anschluss ist nur die einfache Variante mit dem Pattern_Config beschrieben.
Um die Programmierung des ATTiny85 mit dem UNO und der aufgesteckten 400DE-Platine durchführen zu können,
muss der UNO für die Tiny-Programmierung vorbereitet werden.

1. Schritt: ATTiny85-Board Installation

Um Fehlermeldungen bei der anschließenden Tiny-Programmierung vorzubeugen, ist es notwendig zu überprüfen ob die richtigen, zusätzlichen Board-Bibliothek in der Arduino IDE installiert sind. Ansonst kommt bei der weiteren Programmierung die folgende Fehlermeldung:



Installation der benötigten Board-Bibliothek in der Arduino IDE:

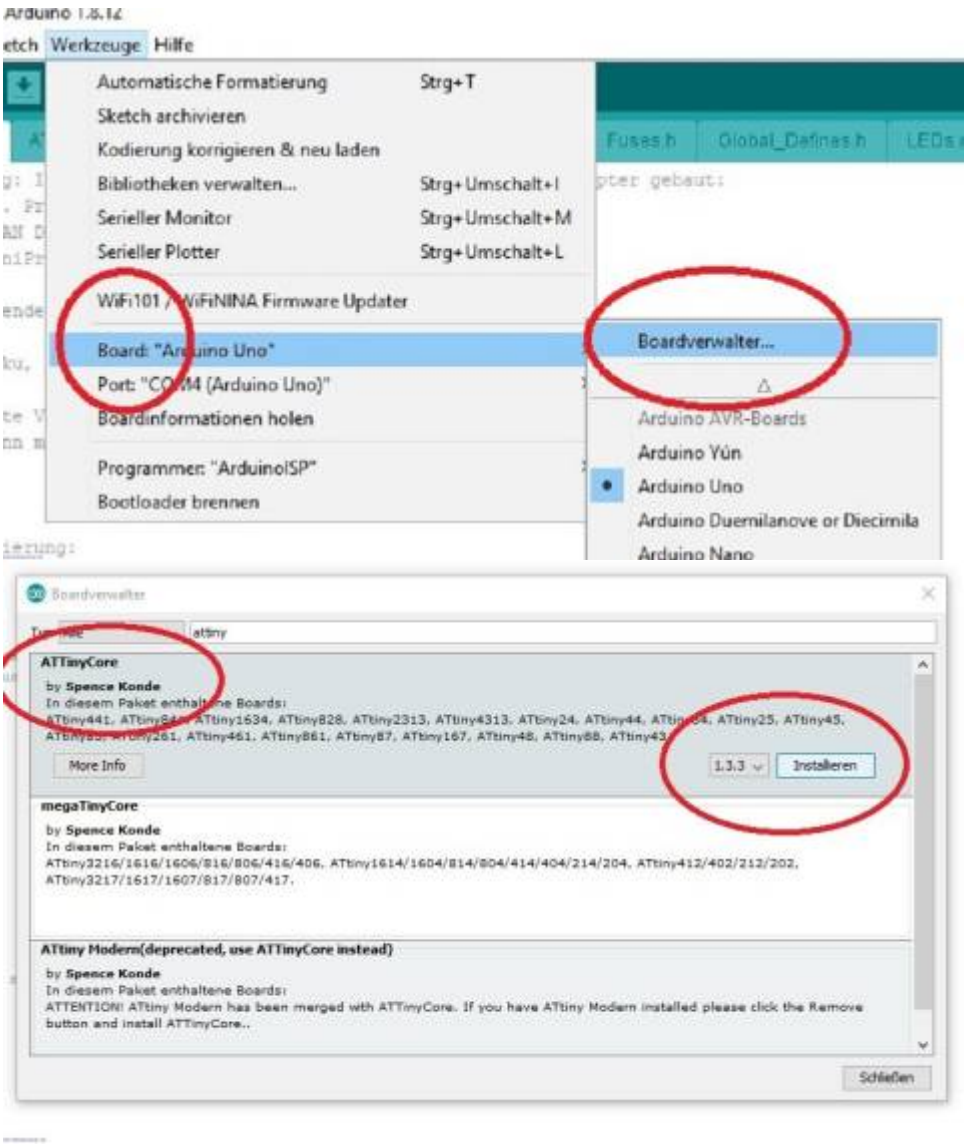
1. Arduino IDE öffnen
2. Datei – Voreinstellungen öffnen



3. Eintrag [1)] wie abgebildet



4. Anschließend in der Boardverwaltung die Bibliothek „ATTinyCore“ suchen und installieren.



5. Nach der Installation empfiehlt es sich die Arduino IDE neu zu starten, um alle Änderungen zu übernehmen.

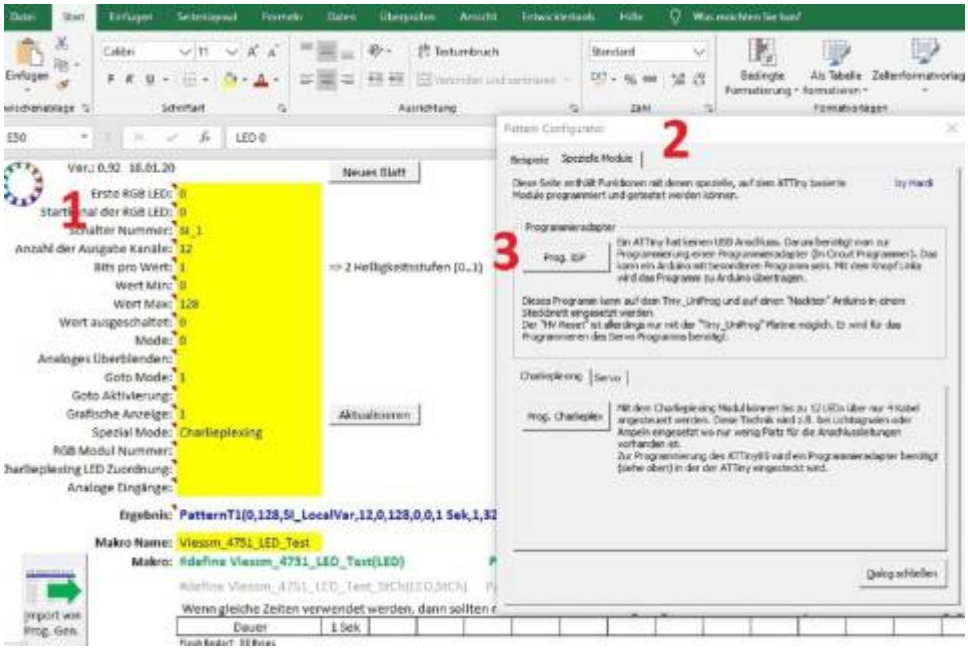
Diese grundlegenden Vorbereitungen sind nur einmal erforderlich.
Außer man wechselt den Rechner....

---

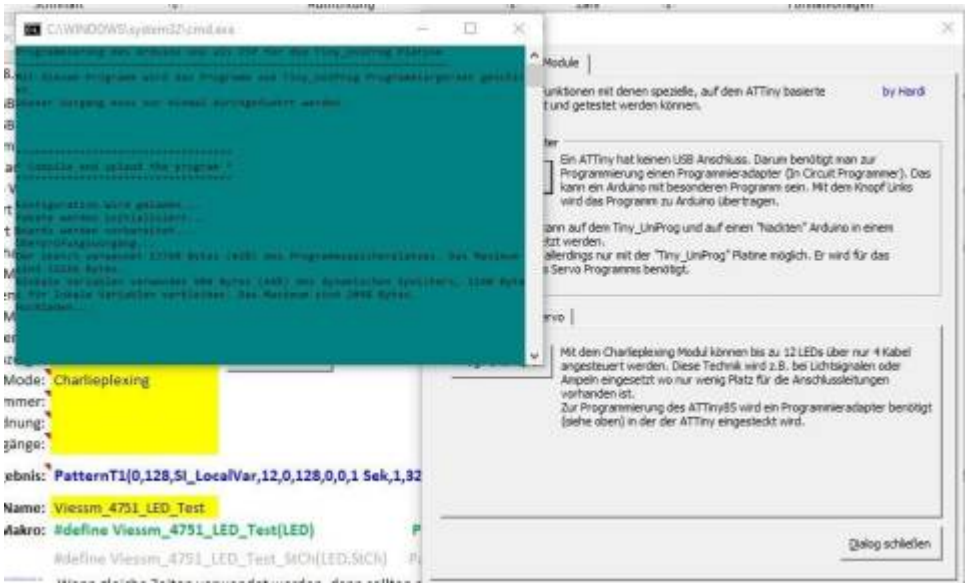
2. Schritt: Mit MobaLedLib-Pattern_Configurator konfigurieren

Verbinden des UNO (400DE-Attiny-Programmer Platine muss noch nicht angesteckt sein) mit dem PC.
Öffnen des Pattern_Configurator:

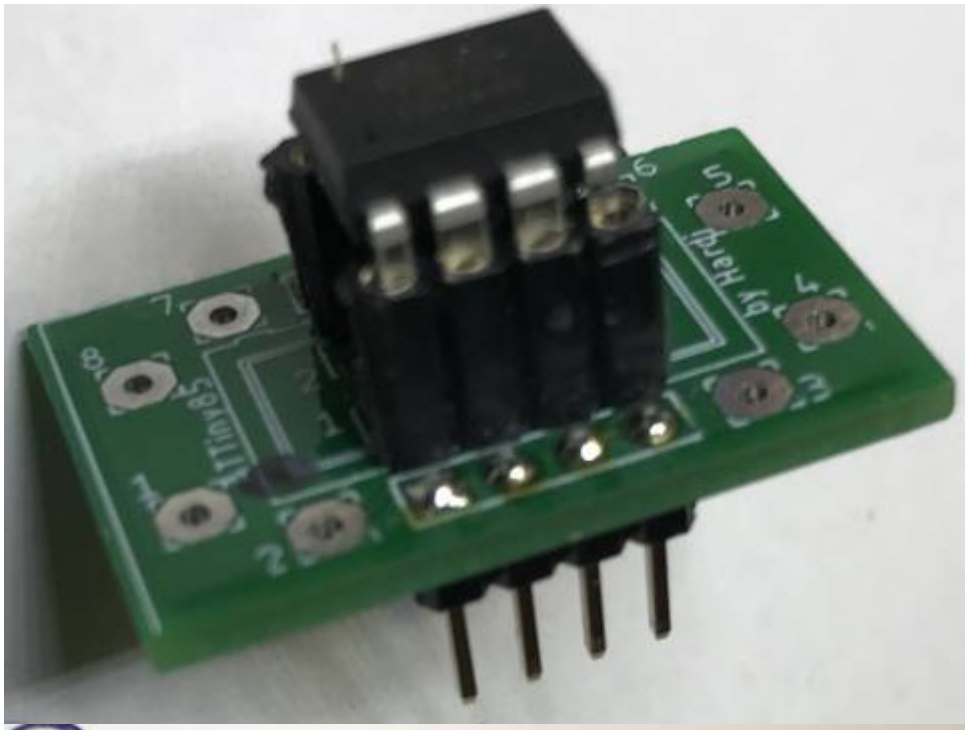
1. Links oben in den Farbkreis klicken.
2. Spezielle Module auswählen.
3. Prog.-ISP drücken.

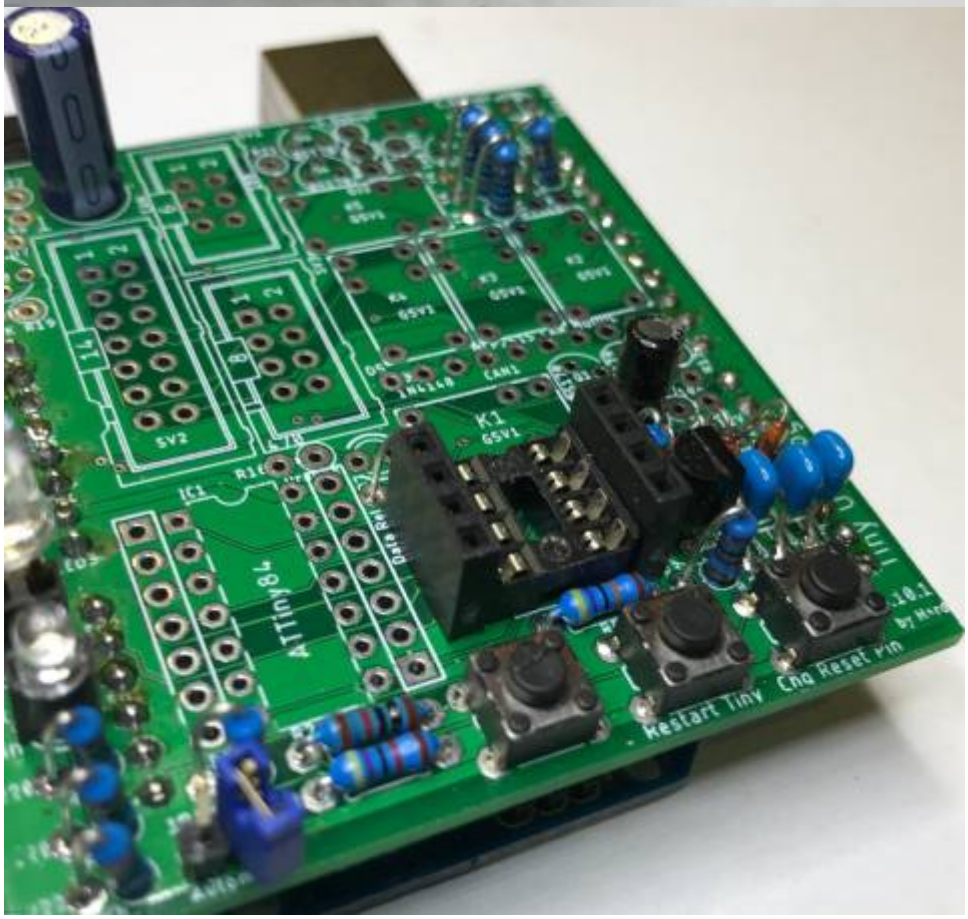


Anschließend wird vermutlich nach dem COM Anschluss des UNO gefragt. Wenn dieser definiert ist, wird der UNO für die Programmierung durch das Hochladen der entsprechenden .ino vorbereitet (C:\Users\MadMax\Documents\Arduino\libraries\MobaLedLib\examples\90.Tools\02.Tiny_UniProg). Die Einstellung werden automatisch durch den Pattern_Config. durchgeführt. Die LED auf dem UNO flackert. Wenn der UNO nur für diesen Zweck verwendet wird ist dieser Vorgang nur einmal durchzuführen.



Nachdem die Programmierung des UNO erfolgreich abgeschlossen wurde ist er jetzt bereit für die ATTiny85 Programmierung.
Die 400DE-Platine mit dem UNO verbinden und einen ATTiny85 in den dafür vorgesehenen Sockel in der richtigen Ausrichtung einstecken.
Empfehlung: Programmiert man öfter IC´s könnte der eingelötete Sockel durch das mehrmalige Herausnehmen und Hineinstecken Schaden nehmen. Wenn man einen zusätzlichen IC-Sockel mit dem eingesteckten ATTiny85 verwendet kann man vorbeugen. Oder man verwendet die neben dem Sockel vorgesehenen Buchsenleisten mit einer auf der 400DE vorhandenen Adapterplatine.





Die grüne Heartbeat-LED blinkt und die weiße LED leuchtet.

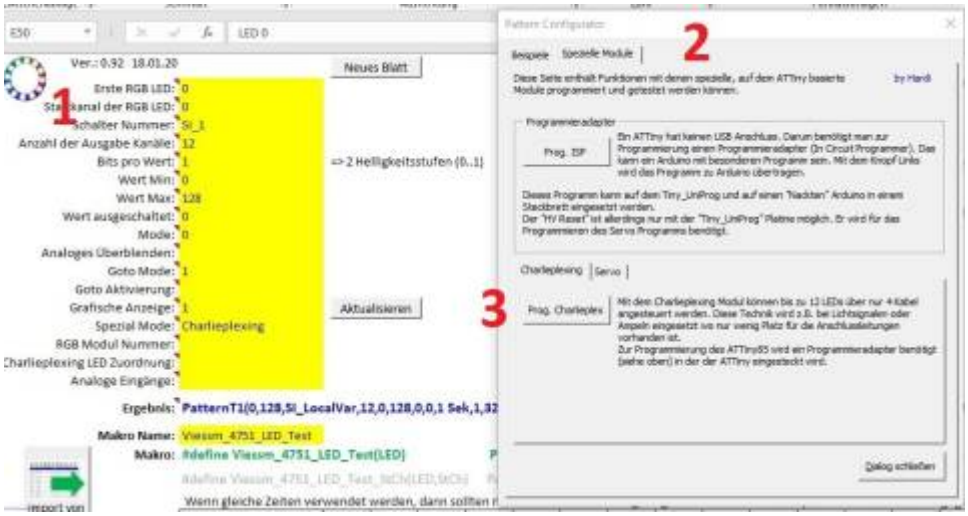
---

3. Schritt: Für welche Anwendung wird der ATTiny85

verwendet?

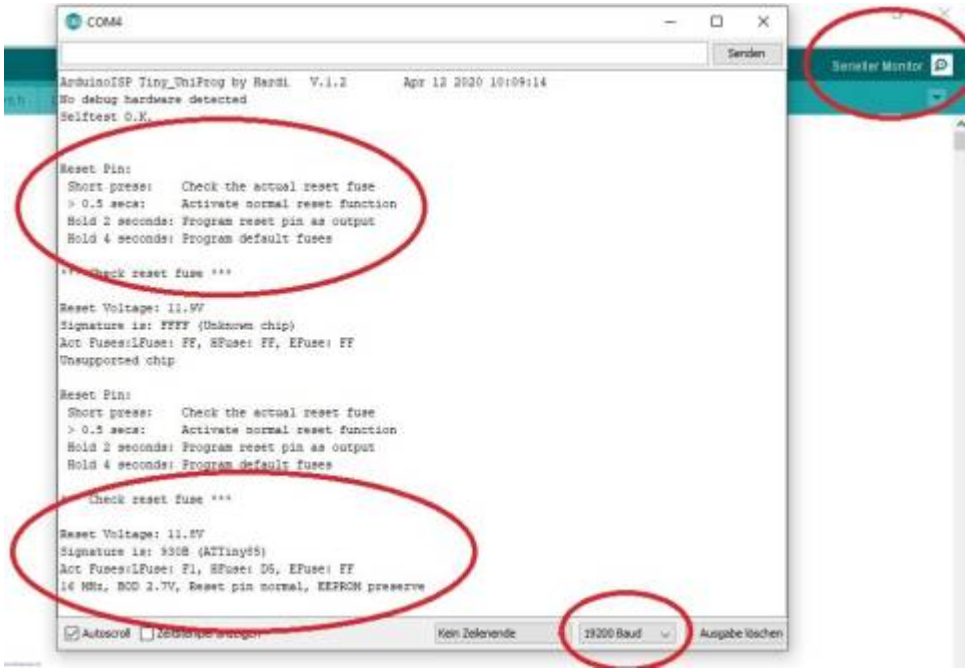
Charlieplexing/Multiplexing-Modul

1. Öffnen des Pattern_Configurators
2. Links oben in den Farbkreis klicken.
3. Spezielle Module auswählen.
4. Charlieplexing auswählen.
5. Prog.Charlieplex anklicken.



Die gelbe Prog-LED blinkt, anschließend flackert die orange Read-LED. Das Hochladen ist abgeschlossen, wenn die weiße LED leuchtet *(C:\Users\MadMax\Documents\Arduino\libraries\MobaLedLib\examples\80.Modules\02.CharlieplexTiny)* .

Dann ist der ATTiny85 für die Verwendung im Charlieplex-Modul einsetzbar.
Das Ergebnis kann man mit dem seriellen Monitor in der Arduino IDE überprüfen bzw. einsehen.



1. Das serielle Monitor Fenster öffnen.

2. Die Taste auf der ATTiny-Uni-Platine drücken (rechte äußere Taste) – Länge je nach Zweck
3. Wenn der ATTiny nicht gleich erkannt wird dann nochmals versuchen.
   Die angezeigten Werte der Fuses und die Frequenzeinstellung 16MHz sind für die Charlieplex Verwendung.

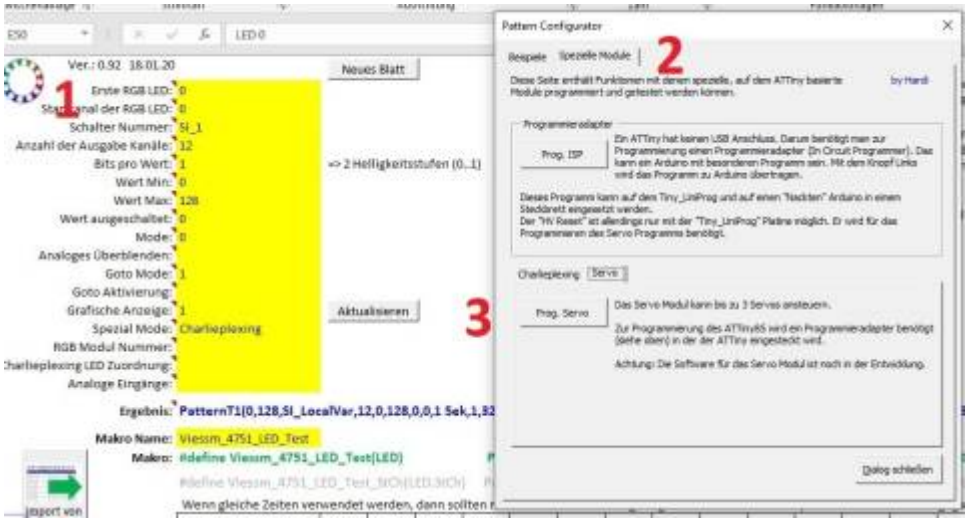
Reset pin normal = weiße LED leuchtet

Die weitere Programmierung für die speziellen Anforderungen ist jetzt mit dem Pattern_Configurator möglich.
Beispiele sind angeführt und eine nähere Beschreibung von Hardi ist unter dem folgenden Link zu finden.

Servo-Modul

Vorgehensweise wie bei der Charlieplex-Modul Programmierung.

1. Öffnen des Pattern_Configurators
2. Links oben in den Farbkreis klicken.
3. Spezielle Module auswählen.
4. Servo auswählen



Anschließend kommt die normale Routine wie sie vom Charlieplex-Modul bekannt ist.
Die gelbe Prog-LED blinkt, anschließend flackert die orange Read-LED.
Hochladen abgeschlossen, wenn die blaue LED leuchtet.

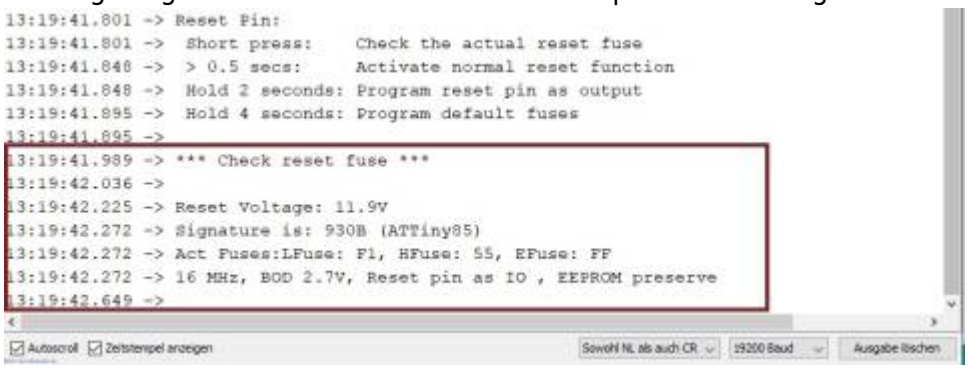
//
(C:\Users\MadMax\Documents\Arduino\libraries\MobaLedLib\examples\80.Modules\01.ATTiny85_Servo
)//.

Bei der Verwendung einer alten Version vom Programm-Generator kann es passieren, das der Reset-Pin nicht automatisch als Ausgang definiert wird.
Dann bitte den Reset-Taster auf dem Tiny_UniProg drücken, bis die blaue LED kurz aufblinkt.

Die richtige Programmierung kann man überprüfen, indem man folgende Schritte macht

1. Das serielle Monitor Fenster öffnen.
2. Die Reset-Taste auf der Platine kurz drücken (rechte äußere Taste)
3. Wenn der ATTiny nicht gleich erkannt wird dann nochmals versuchen.

Die angezeigten Werte der Fuses und die Frequenzeinstellung 16 MHz sind für die Servo Verwendung.

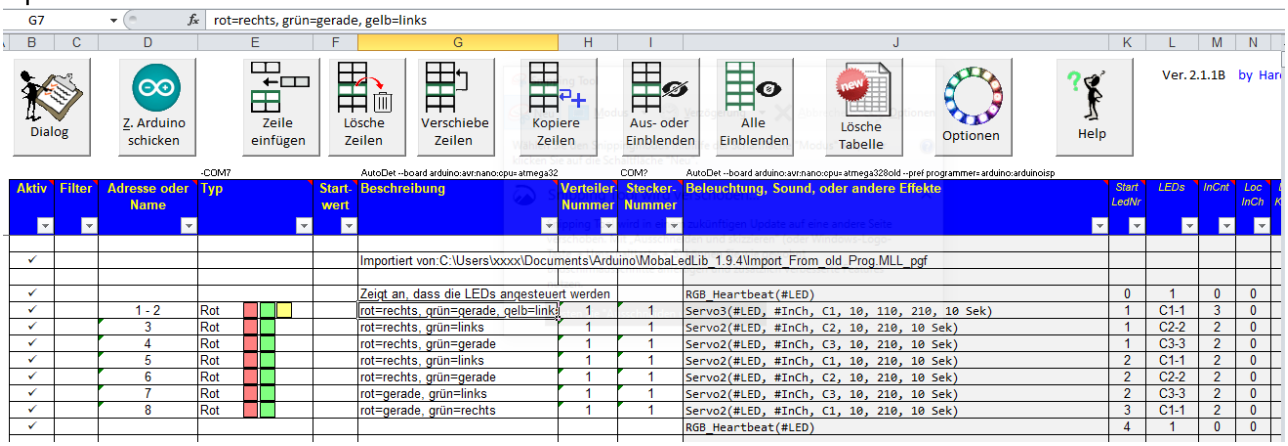


** Reset pin IO = blaue LED leuchtet.**

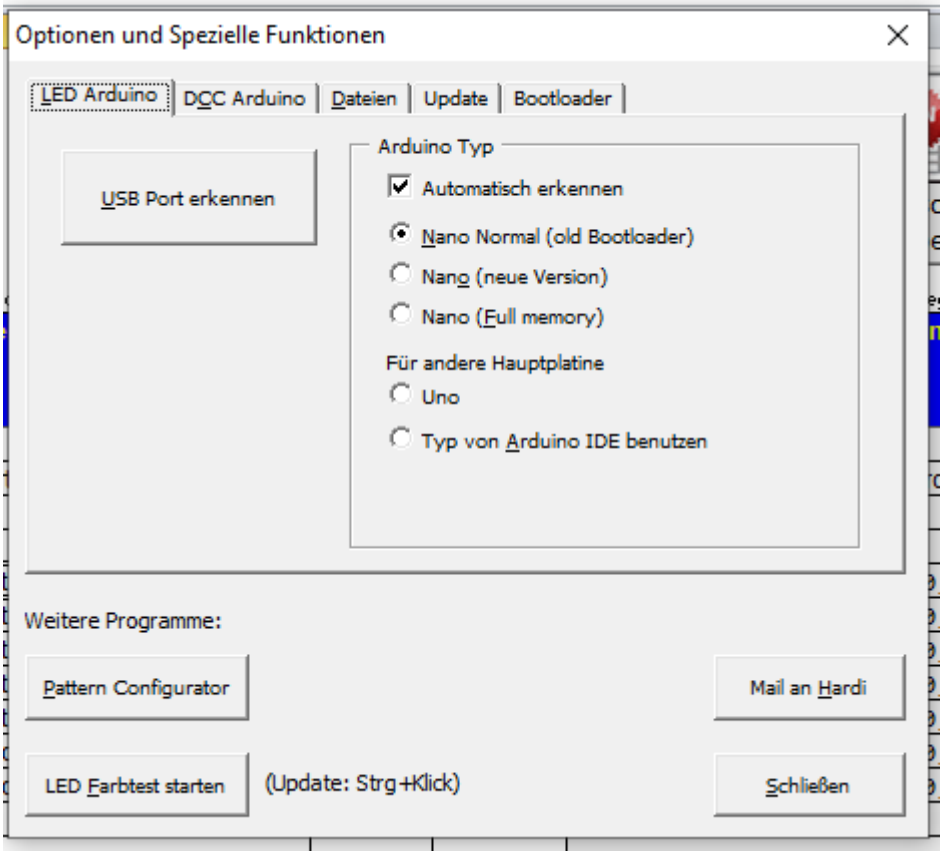
Servo-Positionen mit dem Farbtestprogramm definieren

Um den Servo-ATTiny85 zu verwenden müssen noch die Endpositionen der Servos definiert werden. Das ist mit dem Farbtestprogramm des Programm-Generator möglich.

1. Programm-Generator starten
2. Optionen aufrufen



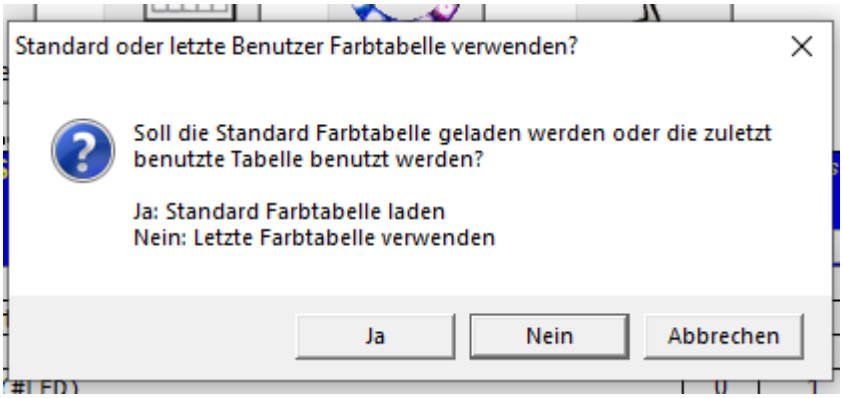
3. LED Farbtest starten



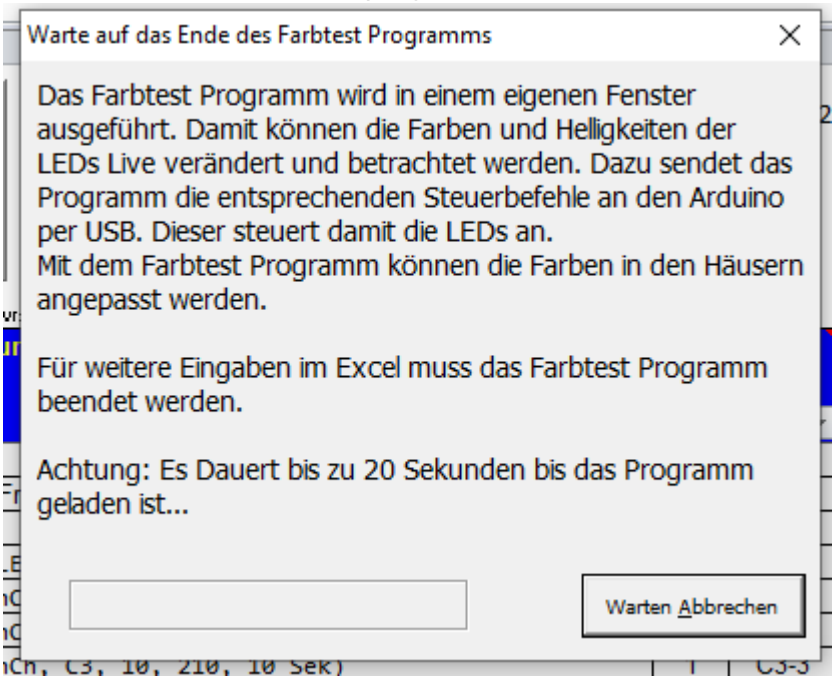
4. Auswahl des COM Ports bestätigen



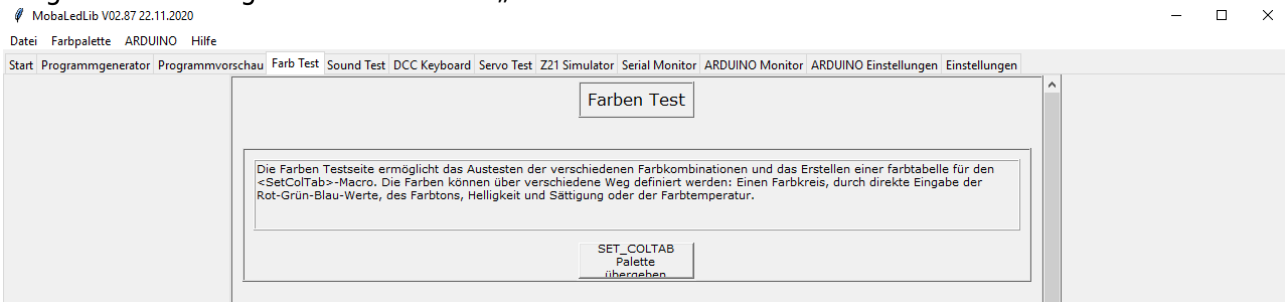
5. Abfrage „Standard oder letzte Benutzer Farbtabelle verwenden?“ bei der erstmaligen Abfrage mit „Ja“ bestätigen.



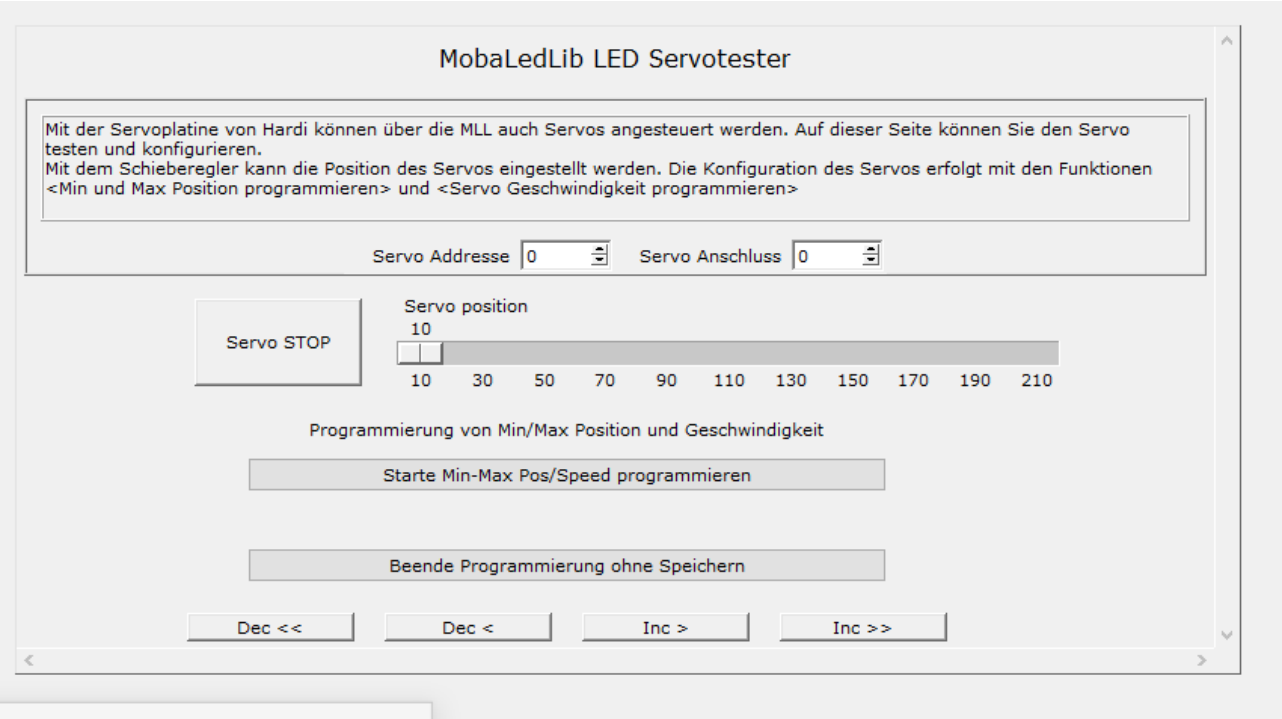
6. Auf das Laden des Farbtestprogramms warten



7. Im geöffneten Programm zum Reiter „Servo Test“ wechseln

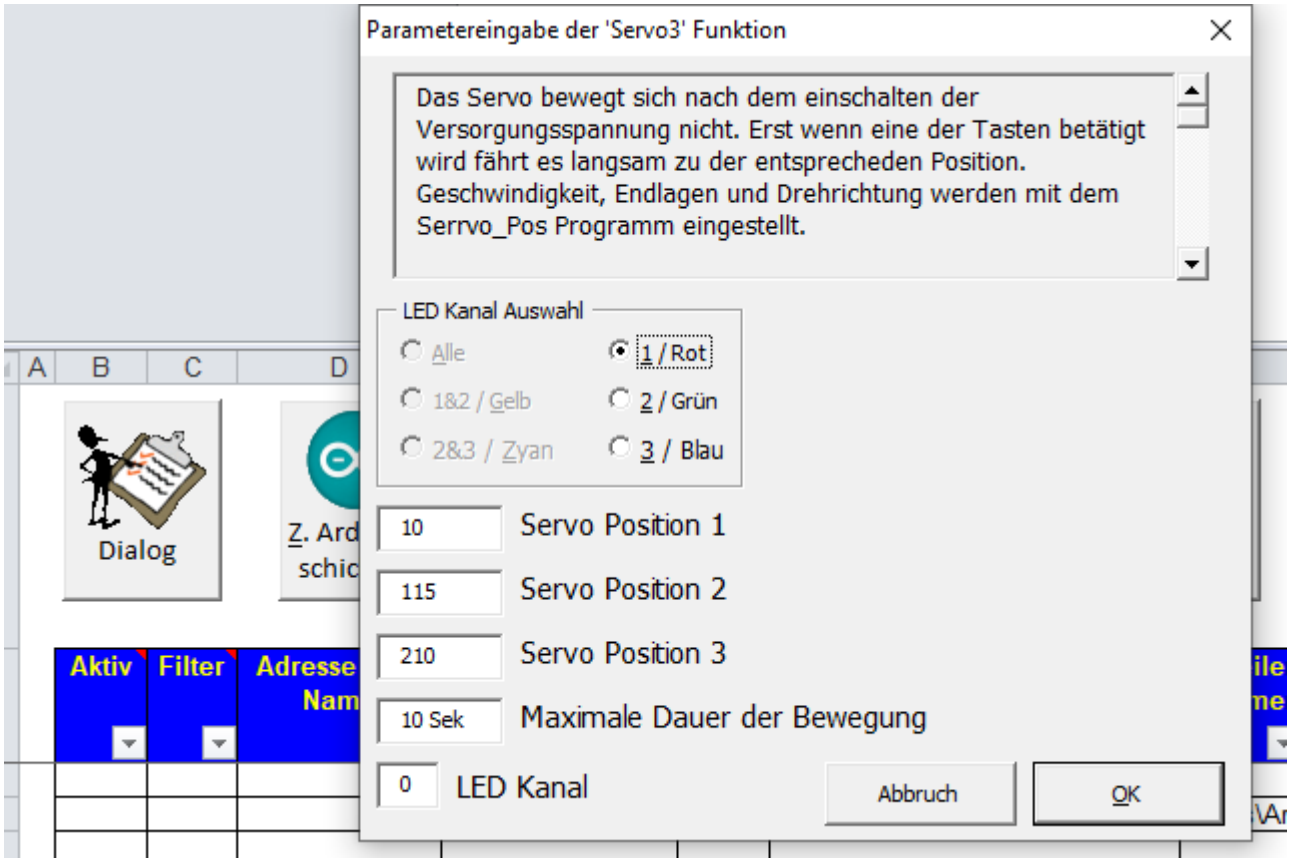


8. Wir sollten jetzt folgende Darstellung sehen



9. Folgende Vorgehensweisen haben sich bereits bei mehreren Anwendern bewährt. Die Werte werden hierbei direkt in den Attiny geschrieben:
10. 1. Servo Addresse und Servo Anschluss angeben. (Servo Adresse „0“ ist die HeartBeat LED auf der Hauptplatine. **Servo Addresse** „1“ ist die erste Servoplatine. **Servo Anschluss** „0“ ist der erste von drei möglichen Servos pro Servoplatine.)
11. 2. Das Servo (ohne Ruderhorn) mittels des Schiebebalkens (unter Servo position) auf 110 (Mittenposition) setzen.
12. 3. Dann das Ruderhorn des Servo montieren.
13. 4. Folgend (unter Programmierung von Min/Max Position und Geschwindigkeit) den Button „Starte Min-Max Pos/Speed programmieren“ drücken um die erste äußere Position (Min) des Servo mittels der Pfeilfelder „Dec «“, „Dec <“, „Inc >“ und „Inc »“ zu bestimmen. Nach erneutem Drücken des Button kann die zweite äußere Position (Max) eingestellt werden. Erläuterung: Dec=Decrease=verringern, Inc=Increase=erhöhen. „<“ und „>“=kleine Schritte, „«“ und „»“=größere Schritte.
14. 5. Nach nochmaligem Drücken des Buttons (auf der Hauptplatine blinkt die weiße mittlere LED) kann die Geschwindigkeit des Servos ebenso mittels der Pfeilfelder eingestellt werden.
15. 6. Weitere Servos können jetzt unter der Eingabe von Servo Addresse und Servo Anschluss eingestellt werden (zurück zu Punkt 1.).
16. 7. Schließen des Farbtestprogramms. Eine extra Speicherung ist nicht notwendig und es gibt auch keinen Button hierfür.

Sollten pro Servo mehr als zwei Stellungen benötigt werden, so können die weiteren Stellungen über den Programm-Generator (Dialog) eingestellt werden, siehe auch erstes Bild in dieser Rubrik. Dort ist als erstes ein Servo mit drei Stellpositionen aufgeführt, danach mehrere Servos mit 2 Positionen. Bei der Servobestimmung bitte die Kanalauswahl (1/Rot, 2/Grün, 3/Blau) beachten. Pro Servoplatine = 3 Servos = 3 Kanäle



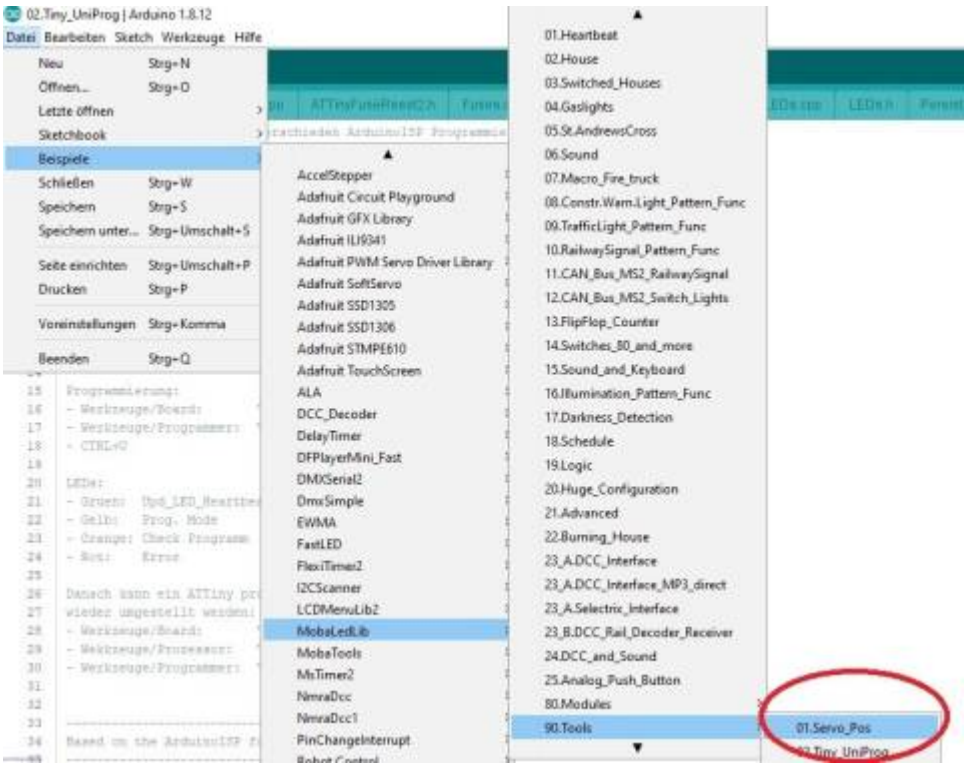
Bitte auch die Bauanleitung „510DE - Servo & LEDs“ beachten. Hier sind wichtige Tipps zu den Servoplatinen-Anschlüssen zu finden

---

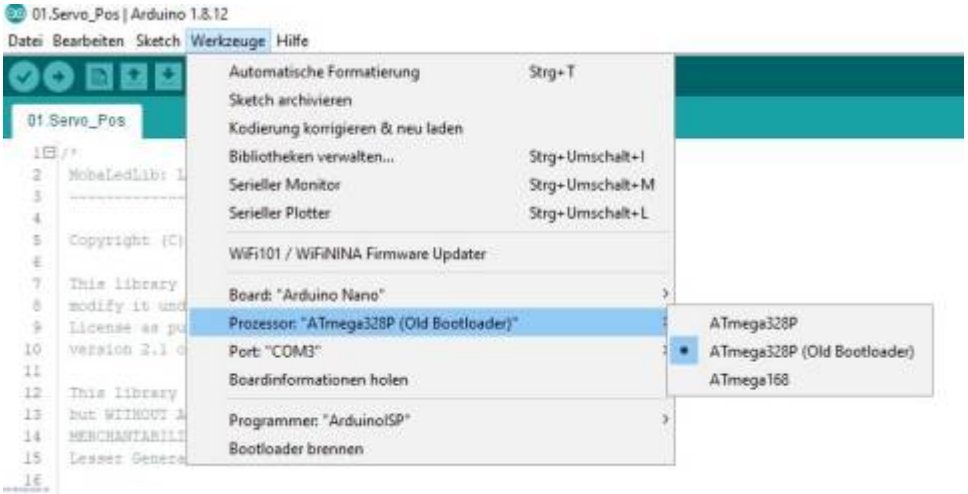
Servo-Positionen mit der Arduino IDE definieren

Um den Servo-ATTiny85 zu verwenden müssen noch die Endpositionen der Servos definiert werden. Das ist mit der Arduino IDE möglich.

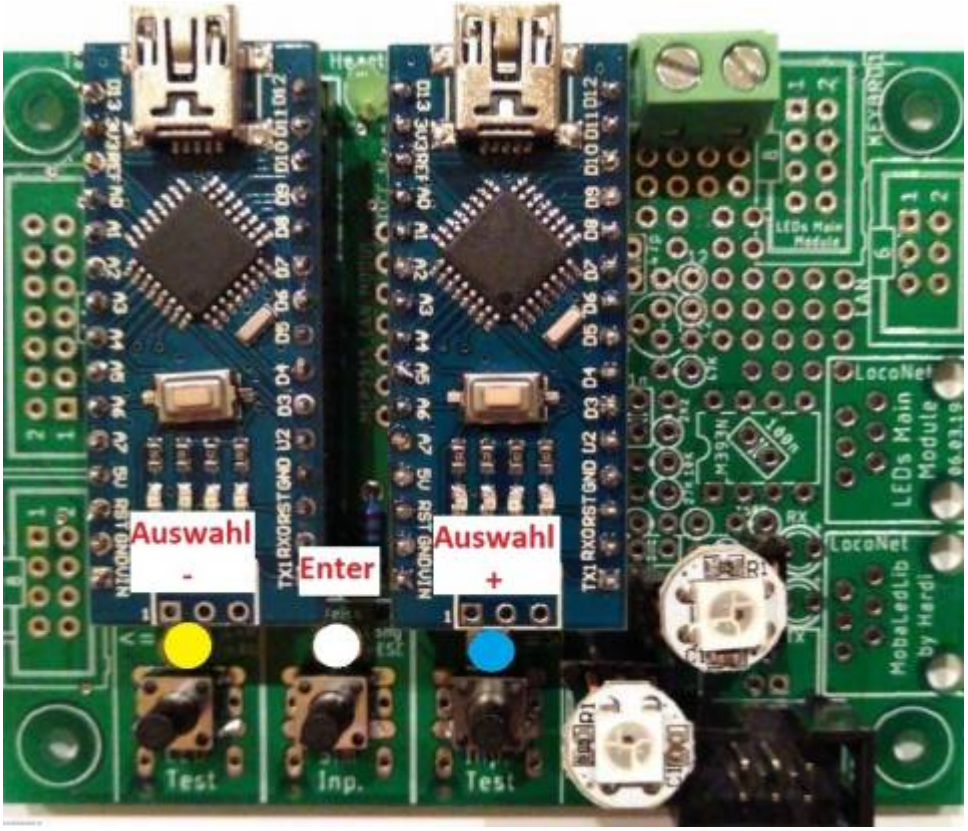
1. Arduino Software öffnen
2. Datei – Beispiele … 01.Servo_Pos öffnen - siehe unten.



3. Bevor der Sketch auf den LED-Nano geladen wird, sollte man die Board Einstellungen überprüfen. Je nach Nano Fabrikat werden unterschiedliche Bootloader-Versionen durch die Lieferanten verwendet.



4. Anschließend den Sketch auf den Nano hochladen.
5. Jetzt sollten auf der Masterplatine die gelbe und die blaue LED bei den drei Tasten abwechselnd blinken.
6. Der LED-Nano ist jetzt bereit für die Einstellung der Servo-Positionen.



7. Mit der rechten(+) Taste (blaue LED) wählt man das entsprechende Servo aus. (In den meisten Fällen ist die LEDNr 0 die Heartbeat-LED). Im seriellen Monitor sieht man die ausgewählte LEDNr. und den RGB Kanal bzw. das ausgewählte Servo. Mit der linken(-) Taste (gelbe LED) kann man in der Auswahl zurück navigieren. \\Das ausgewählte Servo zuckt leicht hin und her. Anschließend mit der mittleren Taste bestätigen.
8. Jetzt blinkt die linke gelbe LED – bedeutet die Min. Position kann eingestellt werden. Mit der linke(-) oder rechten(+) Taste bewegt sich das Servo in die jeweilige Position.
   ACHTUNG: das Servo sollte nicht bis zum äußersten Anschlag eingestellt werden. Es könnte sonst Schaden nehmen und die Funktion ist nicht mehr einwandfrei gewährleistet. Mit der mittleren Taste wird bestätigt.

Im seriellen Monitor werden keine Werte angezeigt.

1. Nach der Bestätigung blinkt die blaue LED – bedeutet die Max.Position kann eingestellt werden.
2. Nach der Bestätigung der Max. Position blinkt die weiße mittlere LED – bedeutet die Geschwindigkeit des Servos kann eingestellt werden.
   - Gelbe Taste(–) langsamer
   - Blaue Taste(+) schneller
3. Nach der Bestätigung ist das Servo fertig eingestellt und das nächste Servo kann ausgewählt werden.

INFO: wenn das Servo-Modul mit einem SMD-WS2811 Chip auf der Rückseite bestückt ist, ändert sich die Reihenfolge der Servos (OUTGrün und OUTRot vertauscht) gegenüber der DIP8 Version des WS2811. Eine erweiterte Programmierung für spezielle Anforderungen ist mit dem Pattern_Configurator möglich. Derzeit gibt es keine Beispiele und auch keine nähere Beschreibung von Hardi.

Man kann mit dem Beispiel_Main experimentieren.

- https://www.stummiforum.de/viewtopic.php?f=7&t=165060&sd=a&start=935
- https://www.stummiforum.de/viewtopic.php?f=7&t=165060&sd=a&start=1790
- https://www.stummiforum.de/viewtopic.php?f=7&t=165060&sd=a&start=1818

Bekannte Fehler

- Der COM Anschluss wird nicht richtig erkannt – Anschluss überprüfen und evtl USB-Port wechseln.
- Verzeichnis „“ nicht gefunden – fehlende Bibliothek in der Boardverwaltung der Arduino IDE siehe
- Wenn man alle Servos (mehr als drei) mit nur einer Servoplatine programmiert (Servos austauscht), bleiben die Einstellungen nicht erhalten - Man konfiguriert nicht das Servo sondern die Ausgänge des ATTiny auf der Servoplatine.
- Der Programmer erzeugt die 12V für den HV-Reset nicht. Dies kann einer der folgenden Ursachen haben
  - Nicht bestückter Widerstand R10
  - Falsche Beschriftung des Plus Pols der LEDs (Dieser muss Links sein). Das hatte Hardi zunächst nicht gemerkt und die Software so geschrieben, dass sie zu der falschen Beschriftung passt. In der aktuellen Version der Platine vom 30.10.19 ist die Beschriftung dann korrigiert. Dummerweise ist in der offiziellen Version der Bibliothek noch die alte Software. Eine korrigierte Version gibt es hier: https://github.com/Hardi-St/MobaLedLib_Docu/blob/master/Quelldateien/02.Tiny_UniProg.zip
  - Falsche Kondensatoren. Die Beschriftung der Einheit auf dem Board verursacht Verwirrung.
    Die Angabe auf der Platine ist 0.22uF. Dies sind 220nF, bitte prüfen ob es sich um die richtigen Werte handelt[2)].
  - Lötbrücke zwischen einem Pad und einer Durchkontaktierung. Dummerweise haben die Durchkontaktierungen keinen Lötstopplack.
  - Falsche bestückter Spannungsteiler (R8 wurde versehentlich mit 47K anstelle von 470K bestückt).

1)
http://drazzy.com/package_drazzy.com_index.json
2)
Aufdruck 224 = 220nF, Falsch ist 223 = 22nF

From:
https://wiki.mobaledlib.de/ - **MobaLedLib Wiki**

Permanent link:
https://wiki.mobaledlib.de/anleitungen/spezial/tiny-uniprog?rev=1616329655

Last update: **2021/03/21 13:27**