192-CH-DMX-Befehlsinterpreter

Was ist denn das?

Für die Steuerung kleiner Scheinwerfer- und Beleuchtungssysteme werden häufig preiswerte Lichtmischpulte eingesetzt, welche bis zu 192 DMX-Kanäle einer Beleuchtungsanlage einzeln ansteuern können (z.B. Helligkeits- oder Farbwertewerte in einem Bereich zwischen 0 und 255). Die Steuerung ist aufgeteilt auf 12 x 16 Kanäle. Die Aufteilung der Kanäle auf 12 Gruppen, welche über Tasten einzeln oder gemeinsam für die Einstellung durch Schieberegler freigeschaltet werden können, ist dann günstig, wenn sich die zu steuernden Geräte auf jeweils 16 Kanäle (evtl. auch Vielfache davon) aufteilen lassen.

Die Werte für die DMX-Kanäle, welche über die Schieberegler eingestellt werden, sind dann in Szenen speicherbar, die je einem Einzelzustand aller benötigten Kanäle der Lichtanlage entsprechen. Diese Szenen wiederum können als Sequenzen aneinander gereiht werden, welche dann ebenfalls im Lichtmischpult abgespeichert und aufgerufen werden können.

Lichtanlagen zur Beleuchtung kleiner Räume oder von Modellanlagen haben meist farbige Leuchtdioden (LEDs). Besonders flexibel einsetzbar sind RGB-LEDs, welche hintereinander geschaltet und über eine oder wenige Signalleitungen angesteuert werden. Eine entsprechende Bezeichnung solcher LEDs ist „Neopixel“, ein oft eingesetzter Typ ist WS2812.

Zur Steuerung solcher LEDs (z.B. in einem Streifen von LEDs) werden drei Kanäle für die Helligkeit der Farben rot, grün und blau benötigt. Mit den 192 DMX-Kanälen eines Lichtsteuerpults sind so 64 LEDs steuerbar, mit den maximal 512 Kanälen eines DMX-Universums sind es 170 LEDs.

Die Einteilung in 12 x 16 auf die Steuerung zuschaltbare Kanäle ist hier nicht günstig, da mit der ersten Gerätgruppe 5 LEDs gesteuert werden können und der 16. Kanal die rote Lichthelligkeit der 6. LED steuert. Durch die aufeinander folgende Ansteuerung der LEDs muss man diesen 16. Kanal der ersten Gruppe einstellen, dann auf die nächste Gruppe umschalten und dort die restlichen zwei Kanäle der 6. LED steuern.

Es ist deshalb abhängig von der zu steuernden Lichtanlage einfacher, eine andere Möglichkeit der Einstellung der Werte der DMX-Kanäle zu nutzen.

Mit Hilfe des Microcontrollers ESP32 wurde ein Interpreter für eine einfache Steuersprache realisiert, mit deren Hilfe die Kanalwerte von Szenen und Sequenzen festgelegt werden können. Die Bedienung erfolgt über eine Weboberfläche, für die der entsprechende Server ebenfalls vom ESP32 bereitgestellt wird. Damit kann eine Befehlsfolge eingegeben und editiert werden. Wird diese Befehlsfolge dann ausgeführt, so erzeugt der ESP32 DMX-Pakete mit 192 Kanälen, welche per Kabel angeschlossene DMX-Empfänger einer Beleuchtungsanlage für die Lichtsteuerung nutzen.

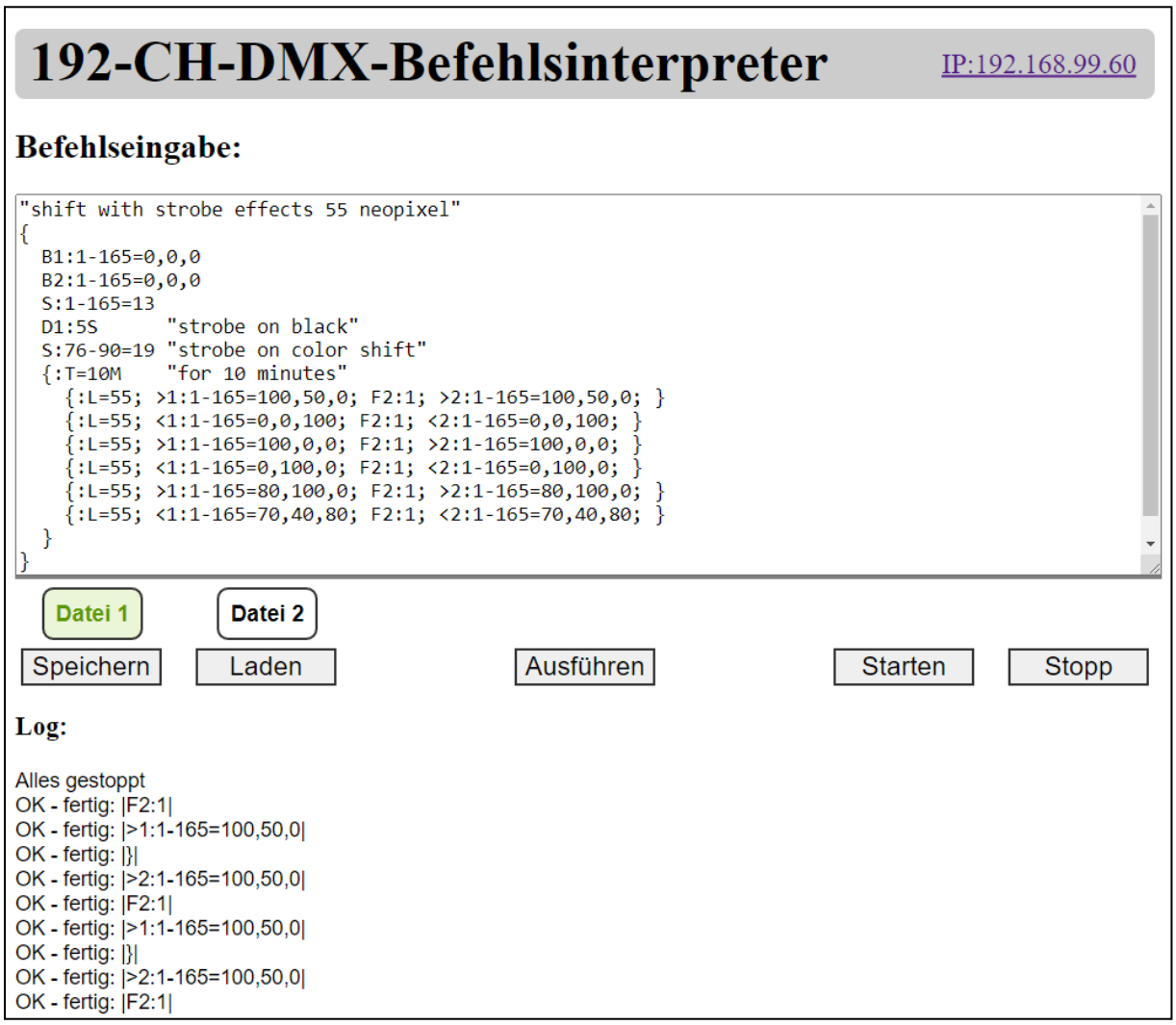
DMX-Befehlsinterpreter

Das Grundkonzept des DMX-Befehlsinterpreters beruht auf zwei Pufferspeichern mit jeweils 192 Kanälen. Für das Setzen der Kanalwerte in diesem zwei Puffern gibt es einen Befehl der flexibel genutzt werden kann, um die Werte aller Kanäle oder von Kanal-Bereichen (Geräten) festzulegen.

Weitere Befehle ermöglichen es dann, zeitgesteuert diese zwei Pufferspeicher als Szenen auf die DMX-Anlage auszugeben und so diese statisch oder dynamisch mittels gleitender Übergänge (Fade) anzusteuern.

Wenn die Werte aus den Pufferspeichern an die DMX-Anlage ausgegeben wurden und gerade kein gleitender Übergang erfolgt, können die Werte in den Pufferspeichern ohne direkte Auswirkung auf die DMX-Anlage geändert werden. So sind durch wiederholtes Setzen der Puffer und dann Ausgeben komplexe Sequenzen realisierbar. Als zusätzliche Effekte können durch zwei weitere Befehle für bestimmte Gruppen von LEDs ein Blitzen (Strobe) festgelegt werden sowie in einem vorzugebenden Bereich die Werte der Kanäle verschoben werden (Shift).

Für die Ablaufsteuerung gibt es noch einen Befehl, um Schleifen bilden zu könne, die dauerhaft, eine bestimmte Zeit oder eine bestimmte Anzahl von Durchläufen lang abgearbeitet werden.



Das Bild zeigt die Weboberfläche des Befehlsinterpreters. Rechts oben steht die aktuelle Webadresse. Darunter erfolgt die Eingabe der Befehle für die DMX-Steuerung. Diese können zeilenweise (kein zusätzliches Trennzeichen erforderlich) oder innerhalb einer Zeile durch '**;**' getrennt eingegeben werden. Das Einrücken von Zeilen mittels Leerzeichen zur Verbesserung der Übersichtlichkeit ist möglich (die Tabulatortaste hat im Webbrowser eine andere Funktion). Kommentare können beliebiger Stelle eingefügt werden, sie sind durch '"' (Anführungsstriche) zu klammern.

Im Eingabebereich eingegebene Befehle können in zwei Dateien abgelegt werde, welche jeweils die Hälfte des 4096 Zeichen großen EEPROM-Speichers belegen. Somit kann jede speicherbare Datei ca. 2040 Zeichen groß sein. Die Auswahl, ob Dateispeicher 1 oder 2 genutzt werden soll, erfolgt über die Auswahlbuttons. Darunter befinden sich die Buttons zum „Speichern" des Eingabebereichs in der ausgewählten Datei oder zum „Laden" aus der Datei in den Arbeitsspeicher. Wird beim Speichern festgestellt, dass das Programm zu groß ist, die für die Programmausführung erzeugte komprimiertere Form aber in den Dateispeicher passt, wird die komprimierte Fassung gespeichert.

Ein Doppelklick auf den Eingabebereich wählt seinen gesamten Inhalt aus. Es ist damit leicht, den Inhalt in einen Editor zu transferieren, dort zu bearbeiten sowie (bei mehr als zwei oder zu großen Programmen) getrennt zu speichern.

Beim Einschalten der Spannung des ESP32 arbeitet der Befehlsinterpreter sofort die in Datei 1 abgespeicherte Befehlsfolge ab. Es ist somit möglich, den Befehlsinterpreter sofort betriebsbereit nach Einschalten der Betriebsspannung zur Steuerung einer Lichtanlage zu nutzen.

Der mittlere Button „Ausführen" ermöglicht es, Befehle zum Setzen und Verändern der Kanalwerte in den zwei Pufferspeichern sofort auf die Lichtanlage auszugeben. Auch wenn das Setzen der Kanalwerte nur die Pufferspeicher betrifft, kann so die Wirkung der Kanalwerte z.B. für rot, grün und blau von RGB-LEDs sofort eingeschätzt und angepasst werden, ohne dafür erst ein Programm abarbeiten zu müssen. Für das sofortige Ausführen von Befehlen muss der Eingabecursor irgendwo unter dem entsprechenden Befehl stehen und dann der Button „Ausführen" angeklickt werden. Nicht für alle Befehle ist dieses direkte Ausführen möglich.

Der Button „Starten" macht eben dies für das im Eingabebereich stehende Programm. Die Ausgabe in eine der zwei Dateien ist vorher nicht erforderlich, das Programm wird direkt aus dem Eingabebereich heraus gestartet. Im Bereich „Log" links unten werden (nach unten scrollend) Bestätigungen zur Befehlsabarbeitung ausgegeben und es werden Fehler angezeigt. Ist wegen eines Fehlers keine weitere Abarbeitung möglich, hält die Verarbeitung an.

Das Programm wird aus dem Eingabebereich gelesen, in eine komprimierte Form konvertiert (überflüssige Zeichen und Kommentare werden entfernt – deshalb können Kommentare an beliebiger Stelle im Text stehen). Die Abarbeitung erfolgt in diesem separaten Speicher. Ein Editieren des Eingabebereichs ist auch während der Programmausführung möglich, allerdings sind die Lade- und Speicherfunktion blockiert, bis die Programmausführung gestoppt wird.

Der Button „Stopp" hat zwei Funktionen. Er unterbricht ein gerade laufendes Programm (so auch das beim Start automatisch ausgeführte Programm aus Datei 1). Ein Doppelklick auf den Button setzt außerdem in der angeschlossenen Lichtanlage alle Kanäle auf den Wert 0 (Blackout).

Befehle

Die Befehle bestehen aus einem Zeichen am Anfang, dahinter stehen abhängig von der Art des Befehls die im Folgenden dargestellten Informationen und Parameter. Für die eindeutige Lokalisierung werden innerhalb der Befehle Trennzeichen **: , - =** benutzt.
Befehle müssen voneinander getrennt sein, bei mehreren Befehlen in einer Zeile müssen sie durch Semikolon ‘;‘ getrennt werden. Befehle in aufeinanderfolgenden Zeilen sind durch das Zeilenende getrennt (kein Semikolon hinter dem letzten Befehl in einer Zeile).

Befehl B – Setzen von Werten für die DMX-Kanäle in den Pufferspeichern 1 und 2

Aufbau: **B** <p> : <Startkanal> , oder - <Endkanal> = <Wert1> , <Wert2>, … ,<Wert48>

<p> -> 1 oder 2, danach als Trennzeichen ein Doppelpunkt
<Startkanal> <Endkanal> -> 1…192
Trennzeichen zw. <Startkanal> und <Endkanal> -> alternativ Komma oder Bindestrich
Trennzeichen zwischen Kanalnummern und Werten -> Gleichheitszeichen
<Werte> -> -1, 0, … , 255 (es sind 1 bis 48 Werte möglich, Trennung durch Kommata)

Im Befehl B steht hinter dem Anfangsbuchstaben als nächstes Zeichen die Angabe des Pufferspeichers 1 oder 2, dann folgt ein Doppelpunkt. Danach ist der Bereich der DMX-Kanäle anzugeben, deren Werte im Pufferspeicher geändert werden sollen. Zwischen Startkanal und Endkanal kann als Trennzeichen ein Komma oder ein Bindestrich geschrieben werden. Es ist zulässig, nur den Startkanal anzugeben, wenn nur der Wert für einen Kanal im Pufferspeicher geschrieben werden soll. Die Zahl des Endkanals darf nicht kleiner als die des Startkanals angegeben werden. Hinter der Angabe des Kanalbereichs folgt ein Gleichheitszeichen.

Die Werte (durch Kommata getrennt) nach dem Gleichheitszeichen werden aufeinander folgend in den angegebenen Kanalbereich geschrieben. Es muss mindestens ein Wert angegeben werden, maximal sind 48 Werte möglich. Wie viele dieser Werte tatsächlich in DMX-Kanäle des Pufferspeichers geschrieben werden, hängt ausschließlich von der Länge des Bereichs ab, der zwischen Startkanal und Endkanal (beide einschließlich) liegt. Ist nur ein Startkanal angegeben, wird nur der erste Wert geschrieben, weitere angegebene Werte werden ignoriert. Ist die Länge des Bereichs zwischen Startkanal und Endkanal größer als die Zahl der angegebenen Werte, werden die Werte entsprechend oft zyklisch genutzt, bis alle Kanäle des angegebenen Bereichs einen Wert erhalten haben. Wird ein Strang von 64 RGB-LEDs vom Befehlsinterpreter gesteuert, so werden bei der Kanalangabe 1-192 und der Angabe von drei Werten 0,80,0 alle 64 RGB-LEDs mit etwa einem Drittel der maximalen Helligkeit auf grün gesetzt. Es ist so mit einer Angabe weniger Werte das Setzen vieler Kanäle möglich. Beim Wert -1 wird immer dann, wenn dieser Wert benutzt wird, der alte Wert im Puffer nicht verändert.

Der Befehl B schreibt nur die Werte in dem angegebenen Pufferspeicher und nicht in den Ausgabepuffer an die DMX-Lichtanlage. Es gibt bei Abarbeitung des Befehls in einem Programm nur die Bestätigung im Log, aber keine erkennbare Reaktion der Anlage. Für das Finden passender Farbwerte für die Kanäle während der Befehlseingabe ist der Button „Ausführen“ zu benutzen, um sofort die Wirkung der Werte in den betreffenden Kanälen bewerten zu können. Üblich werden mehrere Befehle B für unterschiedliche Farben in Teilen der Lichtanlage hintereinander benutzt.

Befehle > und < - Verschieben der Werte eines Bereichs im Pufferspeicher

Aufbau: **<** oder **>** <p> : <Startkanal> , oder - <Endkanal> = <Wert1>,<Wert2>, … ,<Wert48>

<p> -> 1 oder 2, danach als Trennzeichen ein Doppelpunkt
<Startkanal> <Endkanal> -> 1…192
Trennzeichen zw. <Startkanal> und <Endkanal> -> alternativ Komma oder Bindestrich
Trennzeichen zwischen Kanalnummer und Werten -> Gleichheitszeichen
<Werte> -> -1, 0, … , 255 (es sind 1 bis 48 Werte möglich, Trennung durch Kommata)

Der Befehl > verschiebt die Werte der Kanäle in Richtung des Endkanals. Der Befehl < verschiebt die Werte im angegebenen Bereich in Richtung des Startkanals. Wie viele Kanäle diese Verschiebung ausmacht, hängt von der Anzahl der angegeben Werte hinter dem Gleichheitszeichen ab. Der Bereich für die Verschiebung, welcher mittels Startkanal und Endkanal angegeben ist, muss mindestens so groß sein, wie neue Werte hinter dem Gleichheitszeichen angegeben werden. Sind drei Werte angegeben so muss der Bereich zwischen Startkanal und Endkanal auch mindestens die Größe drei haben. Bei einem größeren angegebenen Bereich werden alle Werte um 3 Kanäle von ihrer alten Position in die Befehlsrichtung weg kopiert. Die neu im Befehl angegebenen Werte werden immer auf die freigewordenen Positionen in der Reihenfolge vom niedrigen zum höheren Kanal geschrieben, d.h. bei einer typischen DMX-Lichtanlage ist für die drei Kanäle einer RGB-LED der Wert für die rote Farbe immer der erste angegebene Wert, unabhängig davon in welcher Richtung der frei gewordene Bereich frei geschoben wurde. Wird als ein Wert -1 angegeben, bleibt der alte Wert, der bei Verschieben nur dupliziert wurde, erhalten.

Auch die Auswirkung dieser Befehle kann direkt mit dem Button „Ausführen“ bei der Eingabe überprüft werden. Für unterschiedliche Teile der Lichtanlage können entsprechend mehrere Verschiebebefehle nacheinander angegeben werden.

Befehl D – Pufferspeicher anzeigen (auf DMX-Anlage ausgeben)

Aufbau: **D** <p> : <Anzeigedauer>

<p> -> 1 oder 2, danach als Trennzeichen ein Doppelpunkt
<Anzeigedauer> -> Zahl (dann 10tel Sekunden, alternativ mit Maßeinheit)

Der Befehl D zeigt den angegeben Pufferspeicher für eine Zeitdauer an. Dazu wird die Interpretation des Programms für diese Zeitdauer angehalten. Wird als Anzeigedauer nur eine Zahl angegeben, so wird diese Zahl als 10tel-Sekunden bewertet. Wird direkt an die Zahl der Buchstabe ‘S‘ angehängt, ist die Zeitdauer in Sekunden, ist der Buchstabe ‘M‘ an die Zahl angehängt, so ist die Angabe in Minuten.

Befehl F – Überblenden zwischen den zwei Pufferspeichern

Aufbau: **F** <p> : <Anzeigedauer>

<p> -> 1 oder 2, danach als Trennzeichen ein Doppelpunkt
<Anzeigedauer> -> Zahl (x 0,25 Sekunden, alternativ mit Maßeinheit <n>S oder <n>M)

Der Befehl F1 startet das Überblenden von Pufferspeicher 1 hin zu 2, der Befehl F2 macht dies umgekehrt. Für das Überblenden sind ca. 250 Schritte erforderlich, die minimal in je einer

Millisekunde ausgeführt werden, d.h. das gesamte Überblenden dauert minimal ca. 0,25 Sekunden. Diese minimale Überblendzeit kann durch Angabe einer Zahl als Faktor verlängert werden. Bei längeren Überblendzeiten ist jedoch günstiger, durch Angabe einer Zahl mit Maßeinheit ‘S‘ für Sekunden oder ‘M‘ für Minuten die Dauer festzulegen. Der Faktor wird dann automatisch ermittelt.

Befehl S – Blitzeffekt steuern (Strobe)

Aufbau: **S** <p> : <Startkanal> , oder - <Endkanal> = <Steuerwert>

<p> -> 0, 1 oder 2, danach als Trennzeichen ein Doppelpunkt
<Startkanal> <Endkanal> -> 1…192
Trennzeichen zw. <Startkanal> und <Endkanal> -> alternativ Komma oder Bindestrich
Trennzeichen zwischen Kanalnummern und Werten -> Gleichheitszeichen
<Steuerwert> -> 0, 1 … 10, 11 … 19 (nur ein Wert, steuert das Blitzverhalten)

Der Befehl S bereitet den Blitzeffekt nur vor. Die Ausführung erfolgt denn in Verbindung mit den Befehlen D und F. Die Festlegung, wann und mit welchen DMX-Kanälen geblitzt wird, erfolgt durch die Angabe des Pufferspeichers (1,2) und durch den Bereich zwischen Startkanal und Endkanal. Wird hinter dem Befehlsbuchstaben S die Zahl 1 angegeben, so erfolgt der Blitzeffekt nur, wenn gerade ein Befehl D1 oder F1 abgearbeitet wird. Entsprechendes gilt für S2 hinsichtlich der Befehle D2 und F2. Wird jedoch der Befehl S0 ausgeführt, ist das Blitzen nicht auf einen der zwei Pufferspeicher eingeschränkt. Kompatibel zu einer älteren Programmvariante wo der Blitzeffekt noch nicht auf einen Pufferspeicher beschränkt werden konnte, kann die 0 auch weggelassen werden - S: entspricht S0:

Es kann immer aktuell nur ein S-Befehl aktiv sein, jeder neue S-Befehl (z.B. S1:10-12=15) beendet einen eventuell schon früher gesetzten S-Befehl (z.B. S0:17-21=3).

Für die Angabe von Startkanal und Endkanal gelten die gleichen Festlegungen wie beim Befehl B, es kann ein Bereich oder auch nur einen Kanal angegeben werden. Damit ein weißer Blitz erzeugt wird, muss die Kanalauswahl bei RGB-LEDs jedoch mindestens die drei Farben rot, grün und blau beinhalten. Ein einzelner Kanal, welche eine rote LED steuert kann auch nur rot blitzen.

Nach der Kanalangabe und dem Gleichheitszeichen steht ein Steuerwert. Ist der Steuerwert 0, so werden alle Blitzeffekte ausgeschaltet (egal ob S:0, S0:0, S1:0, S2:0).

Liegt der Steuerwert im Bereich von 1 bis 10 wird ein regelmäßiger Blitzeffekt erzeugt. Dabei sind die Abstände der Blitze beim Wert 1 am kürzesten und beim Wert 10 am längsten.

Bei den Steuerwerten von 11 bis 19 werden zufällige Abstände erzeugt, wobei aber auch hier die Abstände der Blitze durchschnittlich beim Wert 11 am geringsten und beim Wert 19 am größten sind.

Befehle { - Schleifenbeginn und } - Schleifenende

Aufbau: **{** -> Anfang einer Endlosschleife

{ : T = <Zeitdauer> -> Anfang einer zeitgesteuerten Schleife

{ : L = <Durchlaufanzahl> -> Anfang einer Schleife mit Anzahl der Durchläufe

} -> Ende jeder Schleife

Es gibt keine abweisenden Schleifen. Bei einer zeitgesteuerten Schleife (T hinter dem Doppelpunkt) wird die Zeitdauer entweder als Zahl (größer 0) ohne Maßeinheit in 10tel-Sekunden angegeben oder wie bereits bei anderen Befehlen mit 'S' für Sekunden bzw. mit 'M' für Minuten direkt hinter der Zahl. Bei einer Schleife mit vorgegebener Durchlaufanzahl (L hinter dem Doppelpunkt, Zahl auch größer 0) wird die Schleife entsprechend oft durchlaufen.

Ob eine Schleife nochmals zu durchlaufen oder zu verlassen ist, wird erst bei Erreichen des Befehls } für das Schleifenende geprüft. Wenn eine Schleife wegen der in ihr enthaltenen Befehle lange läuft, kann es bei zeitgesteuerten Schleifen sein, dass die Verweilzeit in der Schleife bereits abgelaufen ist, die Schleifenbearbeitung aber seit dem Zeitablauf noch nicht wieder das Schleifenende erreicht hat und deshalb die Schleife auch über die vorgegebene Zeit weiter läuft.

Wegen der Prüfung am Schleifenende, ob eine Schleife nochmals ausgeführt wird, sind als abgearbeitete Befehl im Log nur das Schleifenende und dann wieder der nach dem Schleifenanfang folgende Befehl zu sehen.

Kommentare

Kommentare müssen am Beginn und am Ende ein Anführungszeichen haben. Sie erläutern das im Eingabebereich stehende Programm. Sie haben keinerlei Funktion als Trennzeichen oder sind dahingehend beschränkt, wo sie im Text der Befehle stehen (auch mitten in einem Befehl). Beim Starten der Programmausführung wird für die Abarbeitung eine gesondert gespeicherte, komprimierte Form des Programmtextes erstellt. Hierbei werden auch die Kommentare komplett entfernt. Ist ein Programm im Eingabebereich zu groß (mehr als 2040 Zeichen), um in seiner für die Darstellung gewählten Form in den Dateien 1 oder 2 gespeichert zu werden, so wird automatisch versucht, den Inhalt des Eingabebereichs in komprimierter Form in der Datei zu speichern. Hierbei gehen auch die Kommentare nicht gespeichert. Diese spezielle Speicherung wird sofort im Log angezeigt. Da im Eingabeberiech immer noch das vollständige Programm steht, kann es entweder auf eine passende Größe verkleiner werden oder in einen anderen Editor kopiert und dort gespeichert werden.

Die Programmausführung auf der Grundlage des im Eingabebereich stehenden Programms ist nicht auf die Größe von ca. 2040 Zeichen beschränkt, diese sind nur eine Grenze für die Speicherung in den Dateien 1 und 2 und damit auch für die automatische Ausführung nach dem Einschalten.

Realisierung

Die Hardware des Befehlsinterpreters besteht minimal aus einem ESP32 Board (es wurden das „DOIT ESP32 DevKit“ und „LOLIN 32“ getestet) und einem Schaltkreis MAX485 zur Ansteuerung des DMX-Busses. Die Stromversorgung des MAX485 erfolgt mit 5 Volt. Zur Sicherheit wurde deshalb der DO-Ausgang des MAX485 vor der Verbindung mit dem RX2-Eingang des ESP32 über zwei Widerstände im Pegel auf ca. 3,3 Volt gesenkt. Für den Aufbau von DMX-Ansteuerungen gibt es viele Anregungen im Internet hinsichtlich des Einsatzes von Optokopplern, einer getrennten DC/DC-Spannungsversorgung des MAX485, der Verwendung von BIAS-Widerständen und eines Abschlusswiderstandes.

Die Übersetzung der Software erfolgte über die Arduino-IDE mit dem entsprechenden ESP32-PlugIn. Die Anpassung des Befehlsinterpreters hinsichtlich des Zugangs zum WIFI und die Vorgabe einer geeigneten IP-Adresse bzw. eine DHCP-Nutzung sind im Quelltext festzulegen. Es ist eine einfache OTA-Verwaltung verfügbar, welche ebenfalls im Quelltext zu konfigurieren ist. Für die Weboberfläche stehen zwei Sprachdateien in Deutsch und Englisch zur Verfügung, die über eine Variablendefinition ausgewählt werden.

Die angepasste und compilierte Software ist über OTA oder seriell auf den ESP32 zu laden.

Funktionell sind über Variablen ebenfalls Anpassungen einfach möglich, welche allerdings bislang nicht getestet wurden. Sie ist z.B. die Erhöhung der Größe der gesendeten DMX-Pakete auf mehr als 192 Kanäle möglich. Dabei muss auch der Zeitabstand des Sendes von DMX-Paketen, welcher für die speziell eingesetzte DMX-Hardware auf 10 ms eingestellt ist, eventuell vergrößert werden.

In der Log-Anzeige werden kaum interne Informationen über die Arbeit des Programms ausgegeben. Einige elementare Meldungen können über die serielle Schnittstelle empfangen werden. Nur in einem Fall wird eine Meldung in der Log-Anzeige im Browser erfolgen, die das Senden der DMX-Pakete betrifft. Aktuell wurde das Sende-Interval auf 10 ms eingestellt, wobei allerdings gewartet wird, ob das vorher gesendete Paket abgesandt werden konnte. Wenn diese Wartezeit die Hälfte des normalen Sende-Intervals (d.h. aktuell 5 ms) überschreitet, erfolgt eine Meldung in der Log-Anzeige.

Die realisierten Befehle orientieren sich an den Funktionen eines einfachen Mischpultes einer Lichtanlage. Um das Projekt nicht zu komplex werden zu lassen, wurde auf eine umfangreichere Programmsprache, Variablen sowie die Nutzung des SPIFFS-Dateisystems verzichtet. Da die unverändert übernommenen Funktionen zur DMX-Paketübertragung auf RTOS aufbauen, wäre es möglicherweise auch eleganter, die Zeitsteuerung des Befehlsinterpreters nicht über die millis()-Funktion sondern auch direkt über RTOS-Funktionen zu realisieren. Hier fehlen mir entsprechende Erfahrungen. Hinsichtlich der Weboberfläche würde möglicherweise die Steuerung des Webclients durch den ESP32-Server mittels AJAX-Funktionen eine komfortablere Bedienung ermöglichen. Auch dies ist einer Aufwand/Zeit-Optimierung zum Opfer gefallen.

05.01.2022

Beispiel 1

```
"shift with strobe effects 55 neopixel"
{
 B1:1-165=0,0,0
 B2:1-165=0,0,0
 S:1-165=13
 D1:5S     "strobe on black"
 S:76-90=19 "strobe on color shift"
 {:T=10M    "for 10 minutes"
  {:L=55; >1:1-165=100,50,0; F2:1; >2:1-165=100,50,0; }
  {:L=55; <1:1-165=0,0,100; F2:1; <2:1-165=0,0,100; }
  {:L=55; >1:1-165=100,0,0; F2:1; >2:1-165=100,0,0; }
  {:L=55; <1:1-165=0,100,0; F2:1; <2:1-165=0,100,0; }
  {:L=55; >1:1-165=80,100,0; F2:1; >2:1-165=80,100,0; }
  {:L=55; <1:1-165=70,40,80; F2:1; <2:1-165=70,40,80; }
 }
}
```

Beispiel 2

```
"crossfading and shifting 55 neopixel"
{
 B1:1-33=100,80,0;B2:1-33=0,20,100
 B1:34-66=30,0,100;B2:34-66=30,80,20
 B1:67-99=0,80,80;B2:67-99=120,0,0
 B1:100-132=20,100,0;B2:100-132=10,70,90
 B1:133-165=120,0,0;B2:133-165=100,100,0
 {:T=2M "fade all"
  D1:10;F1:10;D2:10;F2:10
 }
 {:T=2M "fade segments one by one"
  B2:1-33=0,20,100;B2:34-66=30,0,100;B2:67-99=0,80,80;B2:100-132=20,100,0;B2:133-165=120,0,0
  F1:5;D2:3
  B1:1-33=0,20,100;B1:34-66=30,80,20
  F2:5;D1:3
  B2:34-66=30,80,20;B2:67-99=120,0,0
  F1:5;D2:3
  B1:67-99=120,0,0;B1:100-132=10,70,90
  F2:5;D1:3
  B2:100-132=10,70,90;B2:133-165=100,100,0
  F1:5;D2:13;F2:5;D1:3
  B2:100-132=20,100,0;B2:133-165=120,0,0
```

```
 F1:5;D2:3
 B1:67-99=0,80,80;B1:100-132=20,100,0
 F2:5;D1:3
 B2:34-66=30,0,100;B2:67-99=0,80,80
 F1:5;D2:3
 B1:1-33=100,80,0;B1:34-66=30,0,100
 F2:5;D1:13
}
B2:1-33=100,80,0;B2:34-66=30,0,100;B2:67-99=0,80,80;B2:100-132=20,100,0;B2:133-165=120,0,0
{:T=2M "shift all segments"
 {:L=11
  >2:1-33=0,20,100;<2:34-66=30,80,20;>2:67-99=120,0,0;<2:100-132=10,70,90;>2:133-
165=100,100,0
  F1:3;D2:2
  >1:1-33=0,20,100;<1:34-66=30,80,20;>1:67-99=120,0,0;<1:100-132=10,70,90;>1:133-
165=100,100,0
 }
 {:L=11
  <2:1-33=100,80,0;>2:34-66=30,0,100;<2:67-99=0,80,80;>2:100-132=20,100,0;<2:133-165=120,0,0
  F1:3;D2:2
  <1:1-33=100,80,0;>1:34-66=30,0,100;<1:67-99=0,80,80;>1:100-132=20,100,0;<1:133-165=120,0,0
 }
}
{:T=2M "shift pixel one by one"
 D1:10;{:L=11;>2:1-33=0,20,100;F1:1;>1:1-33=0,20,100;}
 D1:10;{:L=11;>2:34-66=30,80,20;F1:1;>1:34-66=30,80,20;}
 D1:10;{:L=11;>2:67-99=120,0,0;F1:1;>1:67-99=120,0,0;}
 D1:10;{:L=11;>2:100-132=10,70,90;F1:1;>1:100-132=10,70,90;}
 D1:10;{:L=11;>2:133-165=100,100,0;F1:1;>1:133-165=100,100,0;}
 D1:10;{:L=11;<2:133-165=120,0,0;F1:1;<1:133-165=120,0,0;}
 D1:10;{:L=11;<2:100-132=20,100,0;F1:1;<1:100-132=20,100,0;}
 D1:10;{:L=11;<2:67-99=0,80,80;F1:1;<1:67-99=0,80,80;}
 D1:10;{:L=11;<2:34-66=30,0,100;F1:1;<1:34-66=30,0,100;}
 D1:10;{:L=11;<2:1-33=100,80,0;F1:1;<1:1-33=100,80,0;}
}
}
```