

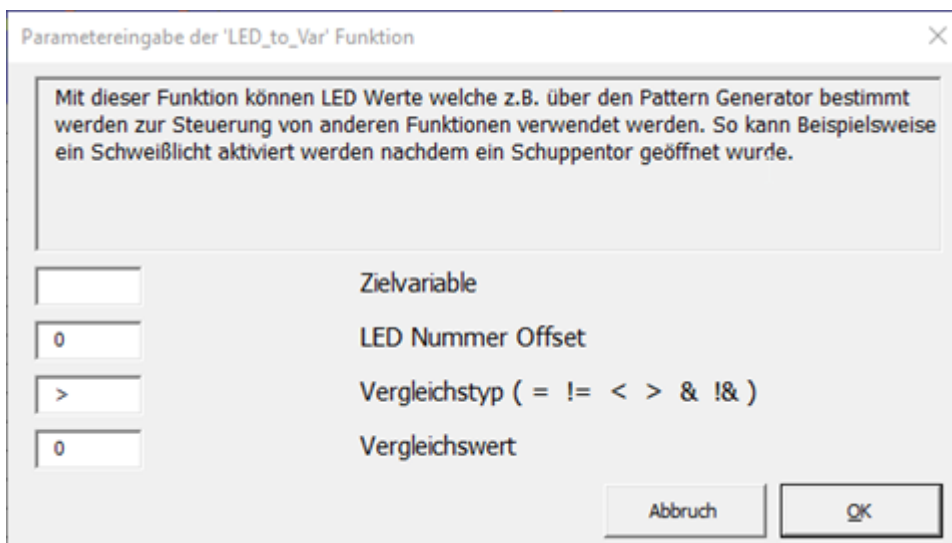
Die Wunderwaffe LED-to-VAR

Wie sich auf einfachste Weise Funktionen beim Erreichen bestimmter Helligkeitswerte schalten lassen und wie man damit jeden Ablauf optimieren kann.

von Michael

Diese Anleitung soll euch die Angst vor einer komplett unterschätzten Funktion der MobaLedLib nehmen. Oftmals wird die Funktion als eine Gimmick für Profis dargestellt, doch im Grunde ist sie ganz einfach zu verstehen.

Los geht es mit dem Menü, das nach dem Auswählen der Funktion **LED-Werte als Variable** (auch bekannt als  **LED_to_Var**) erscheint. Bereits hier zeigt sich eine einfache Parametereingabe.



Zielvariable

Hier ist man also völlig frei und kann als Variable den spezifischen Namen der gewünschten Funktion eingeben, z. B. „Heissluftballon“. Wichtig ist - wie bei allen Variablen - dass der verwendete Name einzigartig ist, denn der Name wird hierdurch zu einem Schalter. Richtig: Statt einer DCC-Adresse schaltet das Wort Heissluftballon zukünftig LEDs oder startet den Gasbrenner zum Abheben.

LED Nummer Offset

Okay, ich gebe zu, das ist ein bisschen unglücklich formuliert und vermutlich ist das die größte Hürde. Was hier eigentlich gemeint ist, ist die Frage nach der referenzierten LED. Also der LED, die wir auslesen wollen bzw. die maßgeblich dafür ist, wann bestimmte Funktionen ausgelöst werden. Damit diese Hürde ein für allemal beseitigt wird, gehe ich in diesem Kapitel möglichst einfach darauf ein.

Wichtig: Die Funktion LED-to-VAR bezieht sich **immer** auf die Funktion, die sich **direkt über** ihr befindet. Um das besser zu verstehen, folgt ein einfaches Beispiel: Wir schalten eine RGB-LED, die nur rot leuchtet, wenn sie eingeschaltet ist und gar nicht leuchtet, wenn sie ausgeschaltet ist. Das ist für den Einstieg sehr einfach. Direkt darunter fragen wir mit der **LED-to-Var**-Funktion genau dieses rote Leuchten ab. Sobald nun die vergebene Variable „Heissluftballon“ aktiv wird (also wenn die rote LED

mit irgendeiner Helligkeit leuchtet, die größer als 0 ist), wird der Gasbrenner im Heißluftballon gezündet und dieser hebt ab.

Wenn bis hierhin alles klar ist, schauen wir uns genau an dieser Stelle den **LED Nummer Offset** genauer an. Wie wir oben gelernt haben, fangen wir bei der ersten LED mit null an zu zählen, die direkt über der LED-to-VAR-Funktion steht (im Folgenden LED-Nr. genannt), in unserem Fall also die rote RGB-LED, deren roter Kanal im aktiven Zustand leuchtet. Man erkennt sie im Screenshot an der rot unterlegten **255** die darüber mit **Kanal 0** gekennzeichnet ist. Die grün unterlegte **0** bezieht sich exakt auf diesen Kanal. Gelb unterlegt ist lediglich die Wenn-Bedingung (größer als null), worauf ich weiter unten eingehe.

Aktiv	Filter	Adresse oder Name	Typ	Start wert	Beschreibung	Verteiler Nummer	Stecker Nummer	Icon	Name	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs
✓									Heartbeat LED	RGB_Heartbeat(#LED)	0	1
									Pins LED Bus definieren	// Set_LED_OutPinList(6 V)		
✓		1	AnAus	0	rote RGB-LED mit voller Helligkeit Aktiv, wenn Kanal rot größer als 0 ist				RGB-LED einstellbar LED-Werte als Variable	ConstRGB(#LED, #InCh, 0, 0, 0, 255, 0, 0) LED_to_Var(heissluftballon, 0, >, 0)	1	1
✓		heissluftballon			Gasbrenner starten				Feuer	Fire(#LED, #InCh, 3, 200)	2	3

In einer Tabelle dargestellt teilt sich die erste LED auf drei Kanäle auf. Kanal 0 entspricht hier also dem roten Kanal dieser LED:

LED-Nr.	Kanal	Offset
1	Rot	0
1	Grün	1
1	Blau	2

Das war bisher ein Kinderspiel, oder? Dann darf's jetzt ein bisschen mehr sein, oder? Sehr gern.

Bevor ich aber jetzt auf Werte unterhalb der LED-Variable eingehe (ja, das geht auch), schauen wir uns an, wie es sich verhält, wenn wir eine Funktion abfragen müssen, die aus mehr als einer RGB-LED besteht oder anders gesagt aus mehr als drei Kanälen. Das könnte beispielsweise ein belebtes Haus sein, immerhin kann die LED-to-VAR-Funktion bis zu 31 Kanäle abfragen, was also locker für zehn RGB-LEDs innerhalb einer Funktion ausreicht. Doch ich mache lieber mit zwei WS2811 bzw. sechs Kanälen weiter.

Im Folgenden nehmen wir uns die [MicroStepper Pro-Platine](#) als Referenz. Das ist im Übrigen ein sehr praktisches Beispiel, weil es sehr wahrscheinlich ist, dass beim Bewegen des Steppermotors auch passende Licht- oder Soundfunktionen ausgelöst werden sollen. Auf die Erklärungen zum [Pattern Configurator](#) verzichte ich an dieser Stelle. Wichtig ist hier nur, dass der Steppermotor sechs Sekunden lang in die eine Richtung fährt und am Ziel für zehn Sekunden pausiert, bevor er für sechs Sekunden wieder zurück fährt. Genau das zeigt der folgende Screenshot.

Der aufmerksame Leser wird jetzt bereits erkennen, dass die Kontroll-LED in jeder dieser Phasen einen anderen Helligkeitswert hat - dazu später mehr.

Dauer		20	6000	10000	6000	
Flash Bedarf: 49 Bytes						
+ -		☐ RGB LED				
LED Nr	Spalte Nr ->	1	2	3	4	5
1	MS1					
2	MS2					
3	MS3		255		255	
4	Bewegung		127	0	127	0
5	Richtung		255			
6	Kontroll-LED	0	1	2	3	0

Eigentlich bleibt alles wie beim vorangegangenen Beispiel. Wir referenzieren lediglich einen anderen Kanal, nämlich den blauen Kanal des zweiten WS2811, der sich auf der Stepperplatine befindet. Die grün unterlegte **5** bezieht sich exakt auf diesen blauen Kanal. Gelb unterlegt ist wieder die Wenn-Bedingung (größer als null).

Aktiv	Filter	Adresse oder Name	Typ	Start wert	Beschreibung	Verteiler-Nummer	Stecker-Nummer	Icon	Name	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs
✓									Heartbeat LED	RGB_Heartbeat(#LED)	0	1
									Pins LED Bus definieren	// Set_LED_OutpinList(6 V)		
✓		1	AnAus	0	Steppermotor (pc)				Muster Pattern Configurator	PatternT4(#LED, 28, #InCh, 6, 0, 255, 0, PM_SEQUENZ	1	2
✓					Aktiv, wenn Kanal rot größer als 0 ist				LED-Werte als Variable	LED_to_Var(heissluftballon, 5, >, 0)		
✓		heissluftballon			Gasbrenner starten				Feuer	Fire(#LED, #InCh, 3, 200)	3	3

Doch wie kommt man auf diese 5?

Dazu erweitern wir einfach mal die oben gezeigte Tabelle um drei Kanäle, denn die Funktion über der Abfrage besteht jetzt aus zwei LEDs.

LED-Nr.	Kanal	Offset
1	Rot	0
1	Grün	1
1	Blau	2
2	Rot	3
2	Grün	4
2	Blau	5

Bevor wir das Kapitel **LED Nummer Offset** schließen, zeige ich noch - wie oben versprochen - das Auslesen von LEDs, die **nach** der LED-to-VAR-Funktion aufgeführt werden. Auch wenn diese Funktion gegeben ist, so bleibt es bei der wichtigsten Regel: Die LED-to-VAR-Funktion bezieht sich **immer** auf die Funktion, die sich **direkt über** ihr befindet. Das muss mindestens **ein** Kanal sein. Im folgenden Screenshot wollen wir den grünen Kanal der unteren RGB-LED abrufen. Für die Zählweise muss man die drei Kanäle über der LED-to-VAR-Funktion mitzählen und dann bei der darunter positionierten LED weiterzählen.

Aktiv	Filter	Adresse oder Name	Typ	Start wert	Beschreibung	Verteiler-Nummer	Stecker-Nummer	Icon	Name	Beleuchtung, Sound, oder andere Effekte	Start LedNr	LEDs
✓									Heartbeat LED	RGB_Heartbeat(#LED)	0	1
									Pins LED Bus definieren	// Set_LED_OutpinList(6 V)		
✓		1	AnAus	0	rote RGB-LED mit voller Helligkeit				RGB-LED einstellbar	ConstRGB(#LED, #InCh, 0, 0, 0, 255, 0, 0)	1	1
✓					Aktiv, wenn Kanal rot größer als 0 ist				LED-Werte als Variable	LED_to_Var(heissluftballon, 4, >, 0)		
✓		2	AnAus	0	grüne RGB-LED mit voller Helligkeit				RGB-LED einstellbar	ConstRGB(#LED, #InCh, 0, 0, 0, 255, 0, 0)	2	1
✓		heissluftballon			Gasbrenner starten				Feuer	Fire(#LED, #InCh, 3, 200)	3	3

In der Tabelle gestaltet sich das nun so:

LED-Nr.	Kanal	Offset
1	Rot	0
1	Grün	1
1	Blau	2
LED-to-VAR-Funktion		
2	Rot	3
2	Grün	4
2	Blau	5

Vergleichstyp (= != < > & !&)

Hier stehen sechs Operatoren zur Verfügung, mit denen man so gut wie jeden Zustand abfragen kann. Die ersten vier Operatoren sind selbsterklärend, die letzten beiden sind extrem mächtig, da sich damit unterschiedliche Funktionen auf einen Helligkeitswert legen lassen. Bevor ich näher darauf eingehe, schauen wir uns die Funktion der ersten vier Operatoren in der Tabelle an:

Operator	Bedeutung	Beispiel für „wahr“
=	Abgefragter LED-Wert und Bedingung müssen gleich sein	Aktiv, wenn der abgefragte Helligkeitswert einer LED erreicht wird (z. B. „Kanal 0 = 255“)
!=	Abgefragter LED-Wert und Bedingung dürfen nicht gleich sein	Aktiv, wenn der abgefragte Helligkeitswert einer LED nicht zutrifft (z. B. „Kanal 0 ≠ 0“)
<	Abgefragter LED-Wert kleiner sein als Bedingung	Aktiv, solange der abgefragte Helligkeitswert einer LED unterschritten bleibt (z. B. „Kanal 0 < 127“)
>	Abgefragter LED-Wert größer sein als Bedingung	Aktiv, sobald der abgefragte Helligkeitswert einer LED überschritten wird (z. B. „Kanal 0 > 127“)
&	Binärer Vergleich	siehe unten
!&	Binärer Vergleich negiert	siehe unten

Vergleichswert

From:
<https://wiki.mobaledlib.de/> - MobaLedLib Wiki

Permanent link:
https://wiki.mobaledlib.de/anleitungen/prog_gen/led-to-var?rev=1791925102

Last update: **2026/10/13 20:58**

