



LED-Systeme effizient steuern – LED-Bussystem LED-B6

Teil 2

Mit dem LED-Bussystem lassen sich vielfältige Lichtsysteme mit LEDs und wenig Verdrahtungsaufwand realisieren. Die kompakt ausgeführte Schaltung verfügt über eine TWI-Schnittstelle, mit deren Hilfe die Module zu einem seriellen Bus zusammengefügt werden können. Jedes Modul verfügt über 6 LED-Ausgänge, wobei jede einzelne LED gezielt angesprochen werden kann. Nach der Hardware-Vorstellung im ersten Teil beschäftigen wir uns jetzt mit der Programmierung sowie einigen praktischen Beispielen.

Das Übertragungsprotokoll

Wie schon im ersten Teil ausführlich besprochen, erfolgt die Kommunikation mit dem LED-Modul über den sogenannten I²C-kompatiblen TWI-Bus. Dieses System ist standardisiert, so dass bestimmte „Regeln“ eingehalten werden müssen. Im Abschnitt „TWI-Bus“ wurde dieses Protokoll ja kurz beschrieben.

Es gibt immer einen Master, der die Daten sendet, und einen Slave, der die Daten empfängt und auswertet. In unserem Fall ist der I²C-USB-Adapter der Master und das LED-Modul der Slave. Es können sich natürlich mehrere „Slaves“ in einem System befinden, die dann unterschiedliche Modul-Adressen aufweisen müssen, um sie gezielt ansprechen zu können.

Wir wollen uns anhand der in Abbildung 12 dargestellten Abläufe sowohl den Schreibvorgang als auch den Lesevorgang einmal genauer anschauen. Im oberen Teil der Abbildung ist ein typischer Schreibvorgang zu sehen.

In den mit „USB-I2C“ gekennzeichneten Zeilen sind die Be-

fehle dargestellt, wie sie von einem Terminalprogramm gesendet werden bzw. wie man sie einzugeben hat.

Hinweis: Das Zahlenformat ist grundsätzlich hexadezimal, also statt 255 dezimal wird hier „FF“ usw. geschrieben. Die Zeile „TWI-Bus“ zeigt die hardwaremäßige Seite mit Aufbau jedes Bytes, diese Information ist vorwiegend für Hardwareprogrammierer von Nutzen.

Jede Datenübertragung beginnt mit der Startbedingung (S), gefolgt von der Slave-Adresse, deren Bit 0 angibt, ob geschrieben oder gelesen werden soll. Die Slave-Adresse ist fest eingestellt und kann nicht verändert werden. Der Wert (Adresse) 34 steht für „Schreiben“, während 35 „Lesen“ bedeutet. Hardwaremäßig wird nach dem Schreiben der Slave-Adresse bzw. nach jedem Byte ein Acknowledge (Bestätigung) vom „Slave“ erzeugt, das dem Master anzeigt, dass das Datenpaket empfangen wurde. Über das Terminalprogramm braucht das Acknowledge nicht ausgewertet zu werden. Hier erfolgt nur das Senden der Zeichenfolge „S 34“, der USB-I2C-Adapter erledigt den Rest. Mehr Information hierzu findet sich in der Dokumentation dieses Adapters.

Beispiel für einen Schreibvorgang

USB - I2C (Befehle)

S	34 (Slave Adresse Schreiben)	05 (Beispiel)	00 (Schreibe Folgebyte in Reg = 0)	0x1C (Schreibe ins Register 00)
----------	-------------------------------------	----------------------	---	--

TWI-Bus

Slave Adresse		Write		LED-B6 - Modul Adresse								Registerauswahl								Register 00h											
B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0
0	0	1	1	1	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
ACK								ACK								ACK															

Slave Adresse :
34h = Schreiben
35h = Lesen

23 (0x25 in Reg. 01)

Register 01h							
B7	B6	B5	B4	B3	B2	B1	B0
0	0	1	0	0	1	0	1
Ack							

06 (0x06 ins Register xx)

Register xxh							
B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	1	1	1	0	0
P							

xx = letztes Register
es ist nicht zwingend erforderlich alle Register zu beschreiben. Es ist auch möglich nur ein Register zu beschreiben.

Bn = Datenbit n
S = Start
Sr = Repeated Start
ACK = Acknowledge
NACK = Not Acknowledge
P = Stop
R/W = Read / Write

Beispiel für einen Lesevorgang

USB - I2C (Befehle)

S	34 (Achtung! zuerst Schreibbefehl)	00 (Modul Adresse wählen)	S	35 (Slave Adresse Lesen)	06 (Anzahl der zu lesenden Bytes)
----------	---	----------------------------------	----------	---------------------------------	--

TWI-Bus

Slave Adresse		Write		LED-B6-Modul Adresse								ACK		Slave Adresse (35h)		Read		Byteanzahl								ACK													
B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0
0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0	0	0	1	1	1	0	0	0	0	0	1	1	1	0	0	
ACK								Sr		ACK								ACK																					

xx (das erste Datenbyte wird gelesen)

Datenbyte 00							
B7	B6	B5	B4	B3	B2	B1	B0
x	x	x	x	x	x	x	x
ACK							

es werden insgesamt 7 Bytes gelesen

.....

xx (lese letztes Datenbyte)

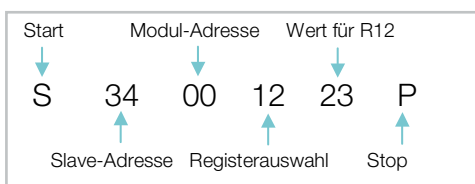
Datenbyte 06							
B7	B6	B5	B4	B3	B2	B1	B0
x	x	x	x	x	x	x	x
P							

Bild 12: Beispiele für einen Schreib- und Lesevorgang auf dem TWI-Bus

Das nächst folgende Byte ist die Modul-Adresse, die im Auslieferungszustand auf 00 steht. Diese Adresse muss bei Betrieb von mehreren Modulen in einem System immer unterschiedlich sein! Mit Hilfe einer speziellen Befehlsfolge kann man diese Adresse umprogrammieren (siehe-> „Register 13–15“).

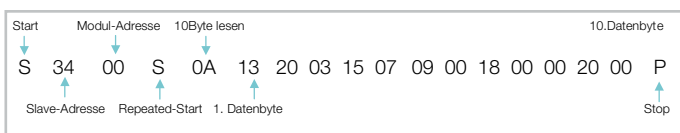
Als Nächstes folgt die „Registerauswahl“. Dieses Byte gibt an, welches Register man beschreiben möchte. Wird hier der Wert „00“ geschrieben, wie in unserem Beispiel, wird das nächste Byte in das Register 00 geschrieben. Ein interner Registerzähler zählt bei jedem Zugriff auf ein Register um 1 weiter, so dass die Registerreihenfolge 00, 01, 02 usw. ist. Man kann auf diese Weise alle Register der Reihe nach beschreiben. Möchte man aber nur ein Register beschreiben, z. B. das Register 12, müssten normalerweise auch alle davor liegenden Register beschrieben werden. Dies kann man dadurch vereinfachen, indem zuerst der Wert „12“ in die „Registerauswahl“ geschrieben und anschließend das Datenpaket für dieses Register gesendet wird. Die „Registerauswahl“ legt somit den Startpunkt zum Beschreiben der Register fest. Den Abschluss jeder Datenübertragung bildet das Stoppsignal bzw. die Stoppbedingung, die als „P“ dargestellt wird.

Hier ein Beispiel dazu: Ein Schreibbefehl für das Register „12“, in das der Wert „23“ geschrieben werden soll, sieht folgendermaßen aus:



Möchte man auch die nachfolgenden Register (13, 14 usw.) beschreiben, sind einfach die entsprechenden Datenbytes anzuhängen, da der Registerzähler automatisch erhöht wird. In ähnlicher Weise wie eben beschrieben können auch Daten gelesen werden. Sind Eingänge am LED-Modul als Eingang konfiguriert, kann eine Abfrage und deren Auswertung nur über eine eigene Applikation (Mikrocontroller) erfolgen. Das Übertragungsprotokoll für einen Lesevorgang ist im unteren Teil der Abbildung 12 zu sehen.

Zu beachten ist hierbei, dass nach der Startbedingung (S) zuerst der Schreibbefehl „34“ und nicht, wie man vermuten könnte, der Befehl 35 gesendet wird. Der Beginn der Übertragung ist bis zur Modul-Adresse identisch mit dem Schreibbefehl. Nach der Modul-Adresse folgt erneut eine Startbedingung, auch „Repeated Start“ genannt. Jetzt wird der Befehl „35“ gesendet, der den „Slave“ dazu veranlasst, Daten zu senden. Zuvor ist aber noch zu übermitteln, wie viele Bytes man lesen möchte. Es können bei unserem LED-Modul maximal 10 Byte gelesen werden. In folgendem Beispiel werden alle verfügbaren Register ausgelesen:



Eine genaue Beschreibung des Datenformats ist im Abschnitt „Register“ nachzulesen.

Terminalprogramm

Für die Kommunikation mit dem USB-I2C-Interface sind beliebige Terminalprogramme oder eigene Software-Entwicklungen einsetzbar, da die Kommunikation über einen Standard-COM-Port erfolgt.

Beispielsweise kann das Terminalprogramm „HTerm“ verwendet werden. Das von Tobias Hammer entwickelte Programm ist unter [1] kostenfrei (auch für kommerzielle Nutzung) als Download erhältlich.

Es existiert sowohl eine Windows- als auch eine Linux-Version. Die im Weiteren beschriebenen Beispiele wurden unter Zuhilfenahme dieses empfehlenswerten Programms erstellt und getestet.

Informationen zur Installation sind in der Dokumentation des USB-I2C-Interfaces (siehe Link in Teil 1) zu finden. Abbildung 13 gibt einen ersten Eindruck des Terminalprogramms wieder.

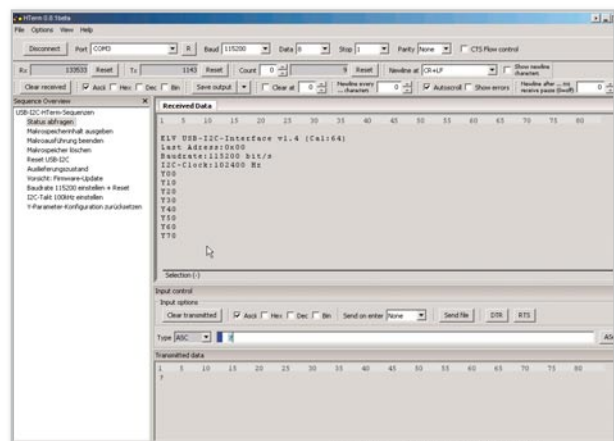


Bild 13: Das komfortable Terminalprogramm „HTerm“

Die Register

Es gibt insgesamt 26 Register, die beschrieben, und 10 Register, die gelesen werden können. Durch das Beschreiben der Register werden bestimmte Operationen ausgeführt wie z. B. das Ein- oder Ausschalten einzelner LEDs oder das Ausführen von Makros usw. Tabelle 1 zeigt eine Aufstellung der beschreibbaren Register.

Im Folgenden wollen wir jede Registergruppe im Einzelnen beschreiben.

Register 0–6

Zum manuellen Schalten der LEDs sind die Register Nr. 0 bis Nr. 6 vorhanden.

Über das Register Nr. 0 können beliebige LEDs mit nur einem Befehl ein- oder ausgeschaltet werden. Jeder LED ist ein Bit zugeteilt, wobei eine „1“ für das Einschalten und eine „0“ für das Ausschalten zuständig ist. Möchte man alle LEDs einschalten, erfolgt dies mit dem Wert 3F. Da die beiden höherwertigen Bits (D 7 und D 8) nicht genutzt werden, kann man auch den Wert „FF“ zum

Register (Schreiben)									
Register Nr.	Datenbits /Datenbytes								Funktion
Hex	D7	D6	D5	D4	D3	D2	D1	D0	
00	x	x	LED6 20	LED5 10	LED4 08	LED3 04	LED2 02	LED1 01	Ein/Ausschalten der LEDs 1 bis 6 Helligkeit wird mit Register 01 bis 06 eingestellt z.B. 0x3F bzw. 0xFF -> alle LEDs an 0x05 LED 0 und LED 3 ein
01	00 bis FF								LED 1 Helligkeit 0 = Aus FF = volle Helligkeit
02	00 bis FF								LED 2 Helligkeit 0 = Aus FF = volle Helligkeit
03	00 bis FF								LED 3 Helligkeit 0 = Aus FF = volle Helligkeit
04	00 bis FF								LED 4 Helligkeit 0 = Aus FF = volle Helligkeit
05	00 bis FF								LED 5 Helligkeit 0 = Aus FF = volle Helligkeit
06	00 bis FF								LED 6 Helligkeit 0 = Aus FF = volle Helligkeit
07	00								Makro's deaktiviert (default)
	01								Ampelsteuerung 1
	02								Fußgängerampel mit Taster
	03								Flackerlicht
	04								Zufallsgenerator 1 (immer nur eine LED eingeschaltet)
	05								Zufallsgenerator 2 (mehrere LEDs eingeschaltet)
	06								RGB-Mode 1
	07								RGB-Mode 2
	08								RGB-Mode 3
	09								Lauflicht 1 (siehe Bild 14)
	0A								Lauflicht 2
	0B								Lauflicht 3
	0C								Lauflicht 4
	0D								Lauflicht 5
	0E								Lauflicht 6
0F								Lauflicht 7	
10								Lauflicht 8	
11 bis 17								diverse Lauflichtmuster 11 bis 17	
08	00 bis FF								Gibt die Geschwindigkeit für Ampelsteuerung Flackerlicht und Lauflicht vor 0= sehr langsam 0xFF = schnell (default = 0x7F)
09	Stunden (Zehner)			Stunden (Einer)			Uhr - Stunden		
0A	Minuten (Zehner)			Minuten (Einer)			Uhr - Minuten		
0B	Sekunden (Zehner)			Sekunden (Einer)			Uhr - Sekunden		
0C	Stunden (Zehner)			Stunden (Einer)			Einschaltzeit Stunden		
0D	Minuten (Zehner)			Minuten (Einer)			Einschaltzeit Minuten		
0E	Sekunden (Zehner)			Sekunden (Einer)			Einschaltzeit Sekunden		
0F	Stunden (Zehner)			Stunden (Einer)			Ausschaltzeit Stunden		
10	Minuten (Zehner)			Minuten (Einer)			Ausschaltzeit Minuten		
11	Sekunden (Zehner)			Sekunden (Einer)			Ausschaltzeit Sekunden		
12	00								Timer off
	01								Timer mit LED (Register 00 bis 06)
	02								Timer mit Makro (Register 07)
13	58								Code zum Ändern der Konfiguration
14	01								Moduladresse ändern (neue Adresse in Reg. 15)
	02								Reset (Softreset)
	03								Werkszustand (stellt alle Register auf Defaultwert)
15	00 bis FF								Neue Moduladresse (erst wirksam nach Reset)

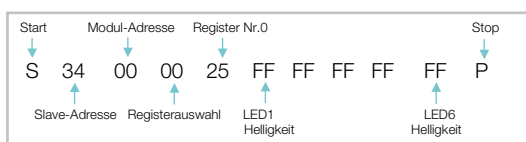
Tabelle 1: Alle Register auf einen Blick

Einschalten aller LEDs verwenden. Möchte man z. B. LED 1, LED 3 und LED 6 einschalten, ist das binäre Bitmuster in eine Hexzahl umzurechnen. Das Bitmuster würde folgendermaßen aussehen: 100101 (binär). Umgerechnet in Hexadezimal ergibt diese Bitfolge die Zahl „25(h)“.

Mit den nachfolgenden Registern Nr. 1 bis Nr. 6 wird die Helligkeit jeder LED bestimmt. So ist das Register Nr. 1 für die LED 1, Register 2 für LED 2 usw. zuständig. Es können alle Werte zwischen 00 und FF verwendet werden. Ein Wert von 80 (128 dezimal) ergibt z. B. eine Helligkeit von 50 %, wobei „FF“ für die volle Helligkeit von 100 % steht.

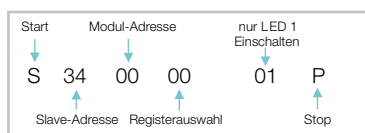
Die LEDs können also wahlweise über das Register Nr. 0 oder mittels der Register 1 bis 6 geschaltet werden. Hat man die Helligkeit der LEDs (Register 1 bis 6) festgelegt, können über Register 0 die LEDs mit einem Byte geschaltet werden. Andersherum können die LEDs auch über das entsprechende Register 1 bis 6 geschaltet werden, indem hier der Helligkeitswert geschrieben wird.

Beispiel: Mit folgender Befehlsfolge werden LED 1, LED 3 und LED 6 mittels Register Nr. 0 eingeschaltet (Wert 25). In die Helligkeitsregister wird jeweils der Wert „FF“ für volle Helligkeit geschrieben:



Hat man die Helligkeitswerte festgelegt, bleiben diese Werte gespeichert, so dass mit dem nächsten Befehl nur das Register Nr. 0 verändert werden braucht, um ein LED-Bitmuster auszugeben.

Beispiel:



Möchte man einen LED-Ausgang zu einem Eingang machen, ist sowohl das entsprechende Bit im Register Nr. 0 als auch der Helligkeitswert auf „00“ zu setzen. Durch einen Lesebefehl wird dieser Port dann kurzzeitig abgefragt. Diese Funktion wird z. B. bei dem später beschriebenen Beispiel-Makro „Fußgängerampel“ genutzt.

Register 7

Dieses Register ist für die internen fertigen Makros zuständig. Ein Makro ist ein selbsttätiges Programm, das eigenständig ohne Steuerung über den TWI-Bus ausgeführt wird, wie z. B. ein Lauflicht oder ein Flackerlicht. Natürlich lässt sich ein Modul, das sich in einem Bussystem befindet, praktisch während des Betriebes „on the Fly“ zu einem „Stand-alone-Gerät“ umprogrammieren. Hierbei gilt es zu beachten, dass, wenn man ein Makro aktiviert, die Register Nr. 0 bis Nr. 6 außer Funktion gesetzt sind.

Einige Beispiele dazu sind im Abschnitt „Praktische Anwendungen“ näher beschrieben.

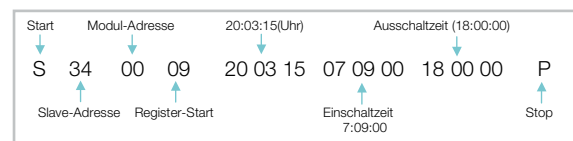
Register 8

Dieses Register legt die Geschwindigkeit für die im Register beschriebenen Makros fest. Der Wertebereich umfasst 00 bis FF, wobei der Defaultwert (Werkseinstellung) 80 (128 dez.) ist. Je höher der Wert, desto höher ist auch die Geschwindigkeit.

Register 9 bis 11

In diesen Registern werden Uhrzeit sowie die Einschalt- und Ausschaltzeiten für die Timerfunktion gespeichert. Das Speicherformat ist das BCD-Format, so dass in den oberen 4 Bit eines Bytes die Zehnerstelle und in den unteren 4 Byte die Einerstelle gespeichert wird.

Hier ein Beispiel:



Es wird die Uhr auf 20:03:15, die Einschaltzeit auf 7:09:00 und die Ausschaltzeit auf 18:00:00 gesetzt.

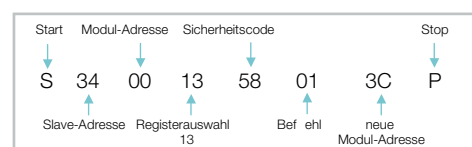
Register 12

Über dieses Register werden die Timerfunktionen aktiviert. Wird hier der Wert „00“ geschrieben, ist die Timerfunktion ausgeschaltet (Default). Mit dem Wert „01“ ist der Timer aktiv und die LEDs werden entsprechend von den gespeicherten Ein- und Ausschaltzeiten in Abhängigkeit der Register 00 bis 06 geschaltet. Möchte man bestimmte Makros per Timer aktivieren, muss das Register 12 mit dem Wert „02“ beschrieben werden. Das entsprechende Makro wird vorher mit Register 07 ausgewählt.

Wichtig: Die Timerfunktion hat höchste Priorität, d. h. alle anderen Funktionen sind nicht aktiv. Ist die Timerfunktion nicht aktiv, hat die Makrofunktion die oberste Priorität.

Register 13 bis 15

Diese Register finden sich ab der Registeradresse „13“, mit ihnen lassen sich Sonderfunktionen ausführen. Zu diesen Sonderfunktionen gehören das Ändern der Modul-Adresse, Reset und das Zurücksetzen in den Werkszustand. Damit man diese Funktionen ausführen kann, ist der Wert „58“ in das Register 13 zu schreiben. Dies ist eine Sicherheitsmaßnahme, damit man nicht unbeabsichtigt die Modul-Adresse ändert. Hier ein Beispiel, das zeigt, wie man die Modul-Adresse auf „3C“ umprogrammiert:



Wichtig! Die neue Modul-Adresse wird ab dem „Stop-Signal“, also nach Beenden der Befehlssequenz, aktiviert. Schreibt man den Befehl „03“ nach der „58“, wird ein Zurücksetzen in den Werkszustand ausgelöst.

Register (Lesen)									
Register Nr.	Datenbits								Funktion
Hex	D7	D6	D5	D4	D3	D2	D1	D0	
00	-	-	LED6	LED5	LED4	LED3	LED2	LED1	LED-Status (nur lesen)
01	Stunden (Zehner)			Stunden (Einer)			Uhr - Stunden		
02	Minuten (Zehner)			Minuten (Einer)			Uhr - Minuten		
03	Sekunden (Zehner)			Sekunden (Einer)			Uhr - Sekunden		
04	Stunden (Zehner)			Stunden (Einer)			Einschaltzeit Stunden		
05	Minuten (Zehner)			Minuten (Einer)			Einschaltzeit Minuten		
06	Sekunden (Zehner)			Sekunden (Einer)			Einschaltzeit Sekunden		
07	Stunden (Zehner)			Stunden (Einer)			Ausschaltzeit Stunden		
08	Minuten (Zehner)			Minuten (Einer)			Ausschaltzeit Minuten		
09	Sekunden (Zehner)			Sekunden (Einer)			Ausschaltzeit Sekunden		

Tabelle 2: Register, die auch ausgelesen werden können

Register auslesen

In Tabelle 2 sind die Register dargestellt, die auch gelesen werden können.

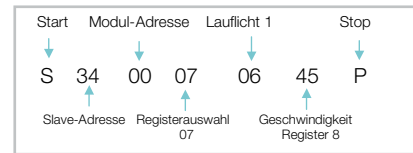
Der dazugehörige Befehl ist im Abschnitt „Das Übertragungsprotokoll“ erklärt. Das Auslesen von Uhrzeit und Timerzeiten dient eigentlich nur zur Kontrolle und kann über das Terminalprogramm nicht weiter ausgewertet werden.

Praktische Anwendungen von Makros

Im Folgenden wollen wir einige Beispiele für praktische An-

wendungen betrachten. Wir beginnen mit dem Makro „Lauflicht“, das wie die beiden anderen Beispiele bereits in die Firmware integriert ist (siehe Tabelle 1). Die Anschlussbelegung und die verschiedenen Modi sind in Abbildung 14 zu sehen.

Mit folgender Befehlssequenz wird diese Funktion aktiviert:



Auch im Bereich der Modellbahn-Technik gibt es viele Anwendungsfälle. Benötigt man z. B. eine Ampelsteuerung für seine Modellbahnlandschaft, kann dies wie in Abbildung 15 dargestellt realisiert werden.


Da nicht genügend LED-Ausgänge zur Verfügung stehen, müssen die sich diagonal gegenüberliegenden LEDs der Ampel parallel geschaltet werden. Normalerweise ist das direkte Parallelschalten von LEDs tabu, da die Kennlinien nie ganz identisch sind und somit die Stromverteilung unsymmetrisch ist. Aber kommen die LEDs vom gleichen Hersteller und sind auch noch aus der gleichen Charge, dann ist dies durchaus möglich. Da sich die Vorwiderstände auf der Platine des LED-Moduls befinden und nicht so leicht ausgetauscht werden können, müssen wir mit dem kleinen Nachteil leben, dass sich hierdurch der LED-Strom halbiert – bei den Lichtstärken moderner LEDs und der hier angestrebten Anwendung sicher kein Problem.

Die zweite Variante der Ampelsteuerung ist die Fußgängerampel, wie sie in Abbildung 16 dargestellt ist.

Hier kann mit einem Taster interaktiv in den Programmablauf eingegriffen werden. Betätigt man den Taster, schaltet die Ampel nach einer gewissen Zeit um. Die Ablaufgeschwindigkeit kann mit dem Wert in Register 8 variiert werden.

Zu beachten ist, dass in den Abbildungen 15 und 16 für die Ampelsteuerung die Signalleitungen symbolisch dargestellt sind, sie bestehen natürlich jeweils aus zweiadriger Litze.

Eine weitere interessante Anwendung ist der Einsatz von RGB-LEDs. Zur Steuerung gibt es drei fertige Makros (Register 07), mit der automatische Farbverläufe dargestellt werden können. Hierbei werden durch additive Farbmischung der drei Grundfarben Rot, Grün und Blau nahezu alle möglichen Farben dargestellt. Die Geschwindigkeit, mit der die Farben wechseln, kann über das Register 08 eingestellt werden.

Wie man die LEDs anschließt, zeigt die Abbildung 17. Pro Modul können zwei RGB-LEDs angeschlossen werden. Hierbei ist darauf zu achten, dass neben einzelnen LEDs nur solche RGB-LEDs verwendet werden können, die über eine gemeinsame Katode bzw. deren integrierte LEDs einzeln herausgeführt sind. Beide Varianten sind in Abbildung 17 dargestellt. Wie die Farberzeugung funktioniert, ist in Abbildung 18 dargestellt. Je nach Lichtintensität der einzelnen LEDs werden unterschiedliche Farben erzeugt. 

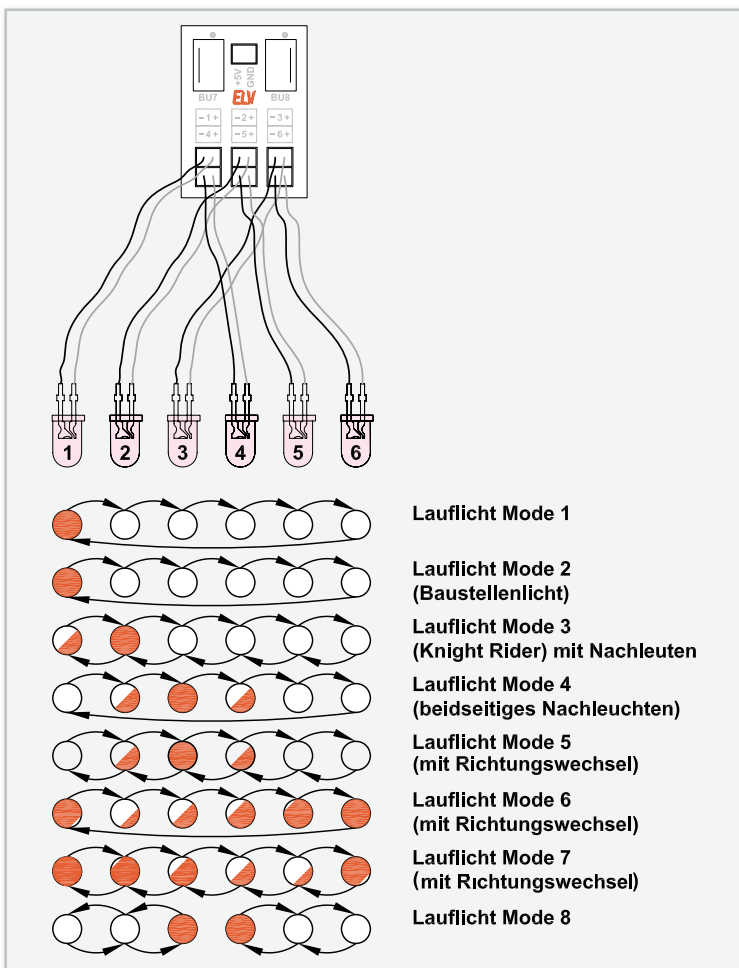


Bild 14: Anschlussbelegung und Modi im Lauflicht-Betrieb

[1] www.der-hammer.info/terminal/

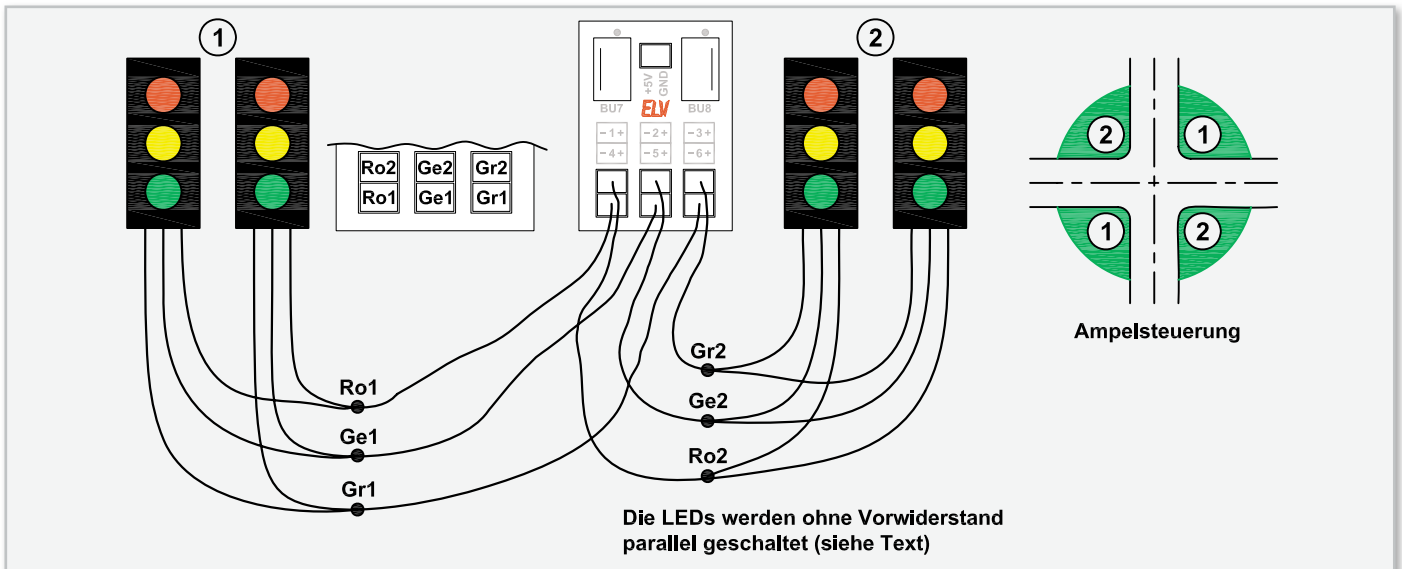


Bild 15: Realisierung einer Ampelsteuerung

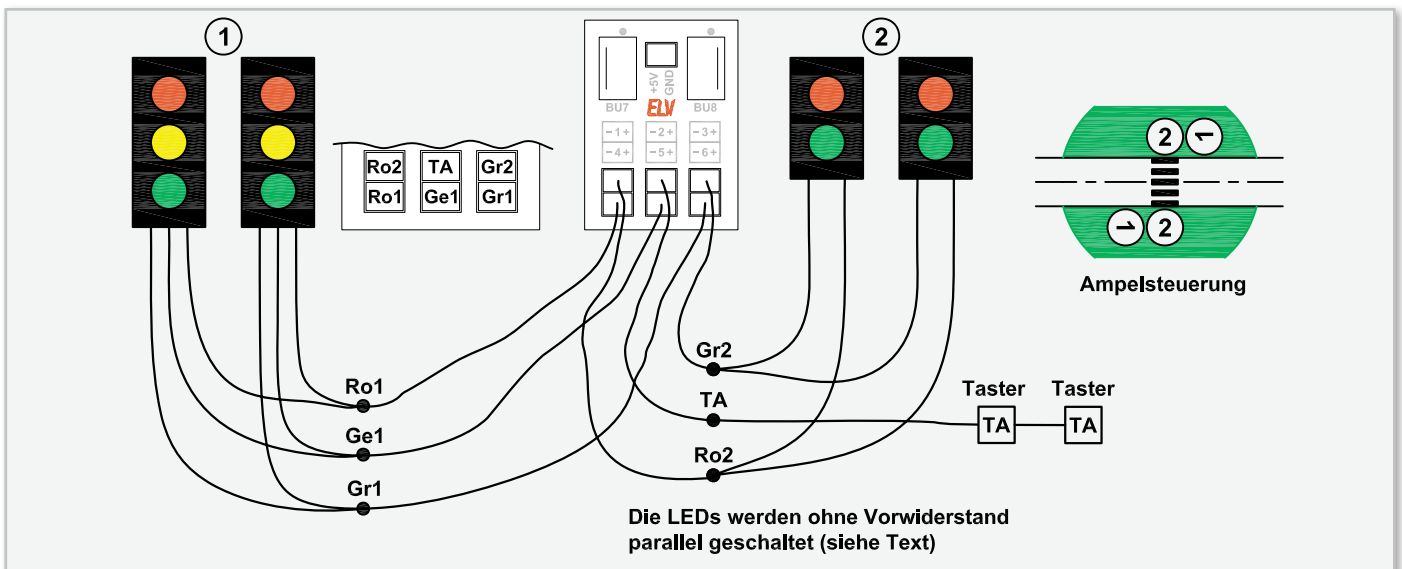


Bild 16: Realisierung einer Fußgänger-Ampelsteuerung

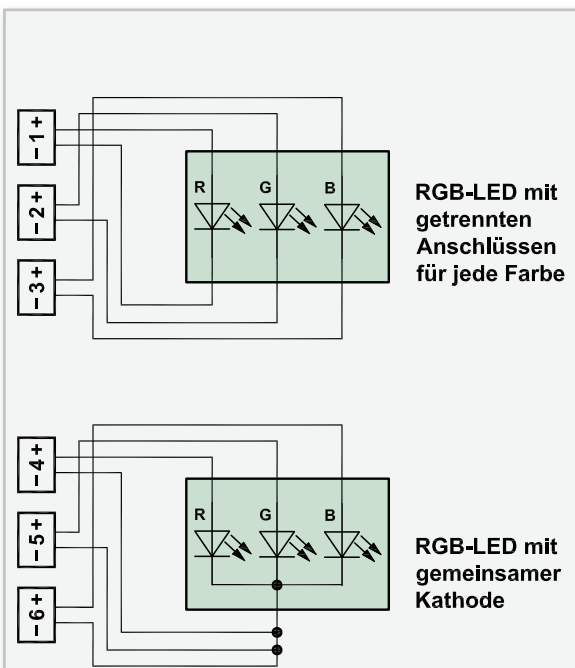


Bild 17: Anschlussbelegung RGB-LEDs

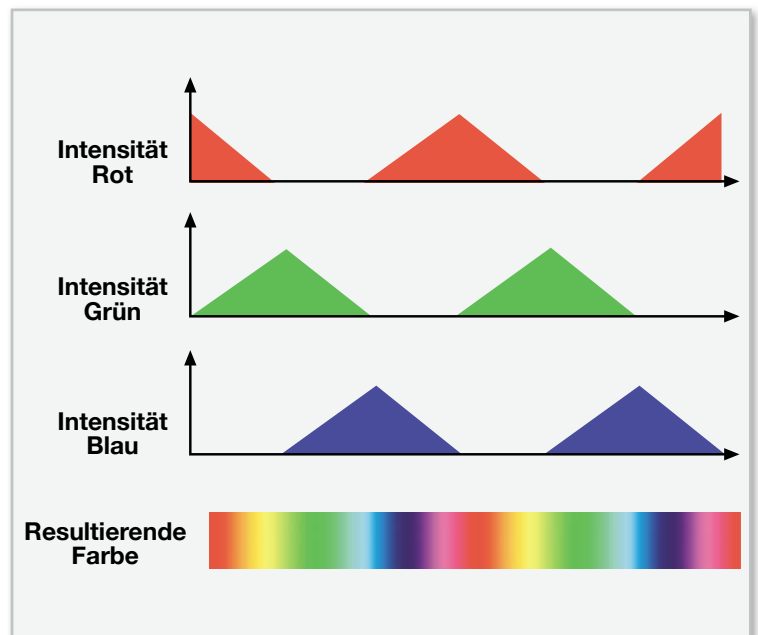


Bild 18: Der Zusammenhang zwischen den Intensitätsverläufen der RGB-Signale und dem resultierenden Farbverlauf