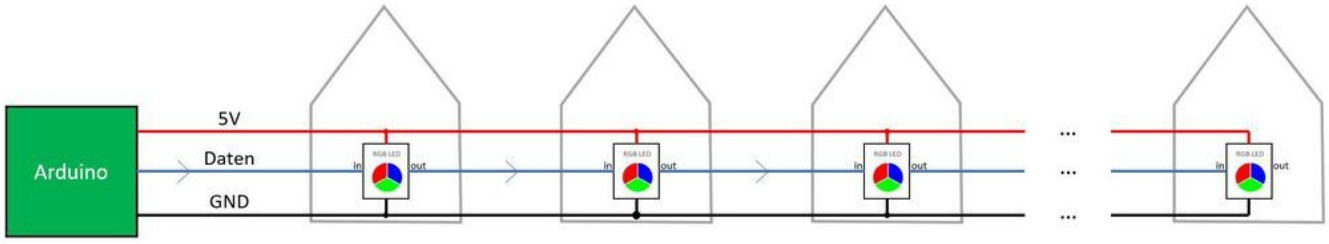
Prinzip der RGB LEDs

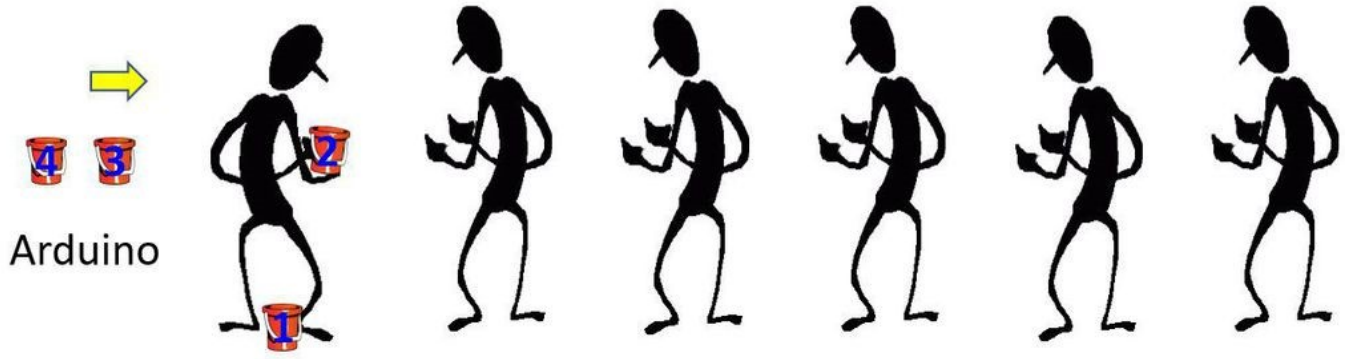
Die LEDs sind in einer Kette angeordnet. Jede LED besitzt einen Daten Eingang und Daten Ausgang. Die erste LED ist mit dem Arduino verbunden. Die zweite LED bekommt ihr Signal vom Ausgang der LED Nummer 1. Sie reicht die Daten über ihren Ausgang an die nächste LED weiter.

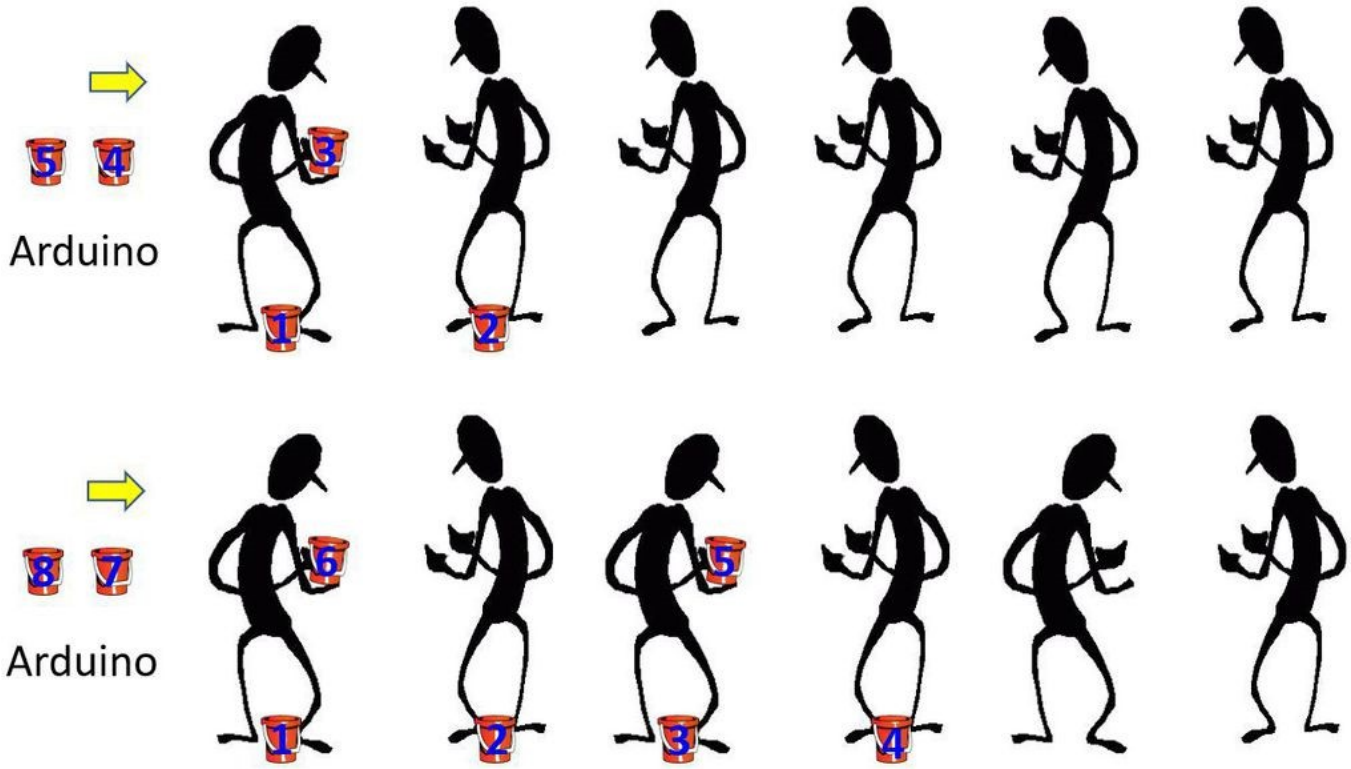


Alle LEDs haben eine gemeinsame Plusleitung (Rot) und eine gemeinsame Masseleitung (Schwarz). Die Datenleitung verbindet immer nur zwei benachbarte LEDs. Die Teilstücke der Datenleitung sind NICHT elektrisch miteinander verbunden. Das ist ganz entscheidend für das Ansteuerungsprinzip der LEDs.

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 wund reicht alle folgenden weiter. Damit bekommt die dritte LED erst den dritten Datensatz…

Das kann man sich wie eine Eimerkette vorstellen. Jeder in der Kette stellt den Ersten Eimer der ihm gereicht wird vor sich hin. Den nächsten Eimer reicht er weiter.





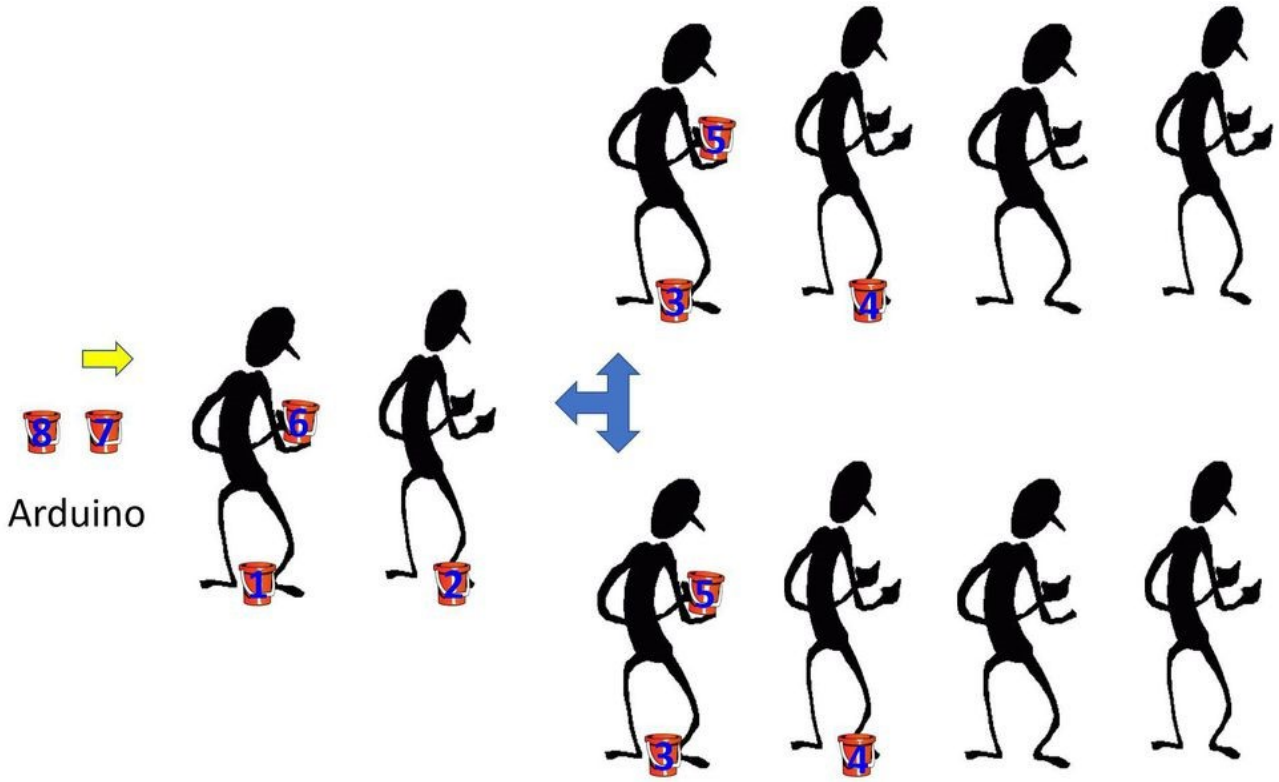
Am Ende hat jeder einen Eimer vor sich stehen. Dann kommt der Befehl vom Kommandanten die Eimer in das Feuer zu schütten.

So funktioniert das auch beim Arduino. Die Daten werden genauso weitergereicht und dann auf Befehl des Arduinos gleichzeitig angezeigt. Dabei ist die Datenübertragung so schnell, dass innerhalb von 4 Millisekunden Hundert LEDs mit Daten versorgt werden können.

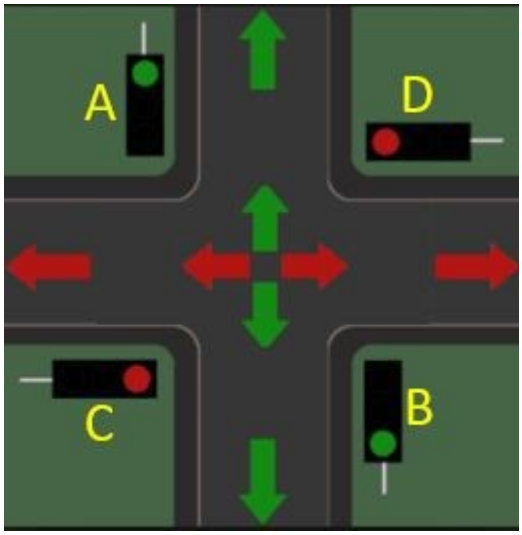
Das Geniale dabei ist, dass man den LEDs keine Adresse per DIP-Schalter oder so geben muss. Das erste Datenpaket kommt bei der ersten LED an und das Zweite bei der zweiten LED…

Ein weiterer Vorteil des Verfahrens ist, dass das elektrische Signal von jeder LED neu generiert wird. Dadurch können sich Störungen nicht aufaddieren. Das ist gerade in unserem Umfeld wichtig. Das Digitalsignal der Eisenbahn ist eine große Störquelle welche über die Schienen überall auf der Anlage sendet.

Entscheiden aber ist, dass man diese Kette nicht teilen kann. Bei einem T-Stück müssten die „Eimer“ verdoppelt werden. Bei elektrischen Signalen funktioniert das. Dann bekommen aber immer zwei Leuchtdioden die gleichen Helligkeitswerte.

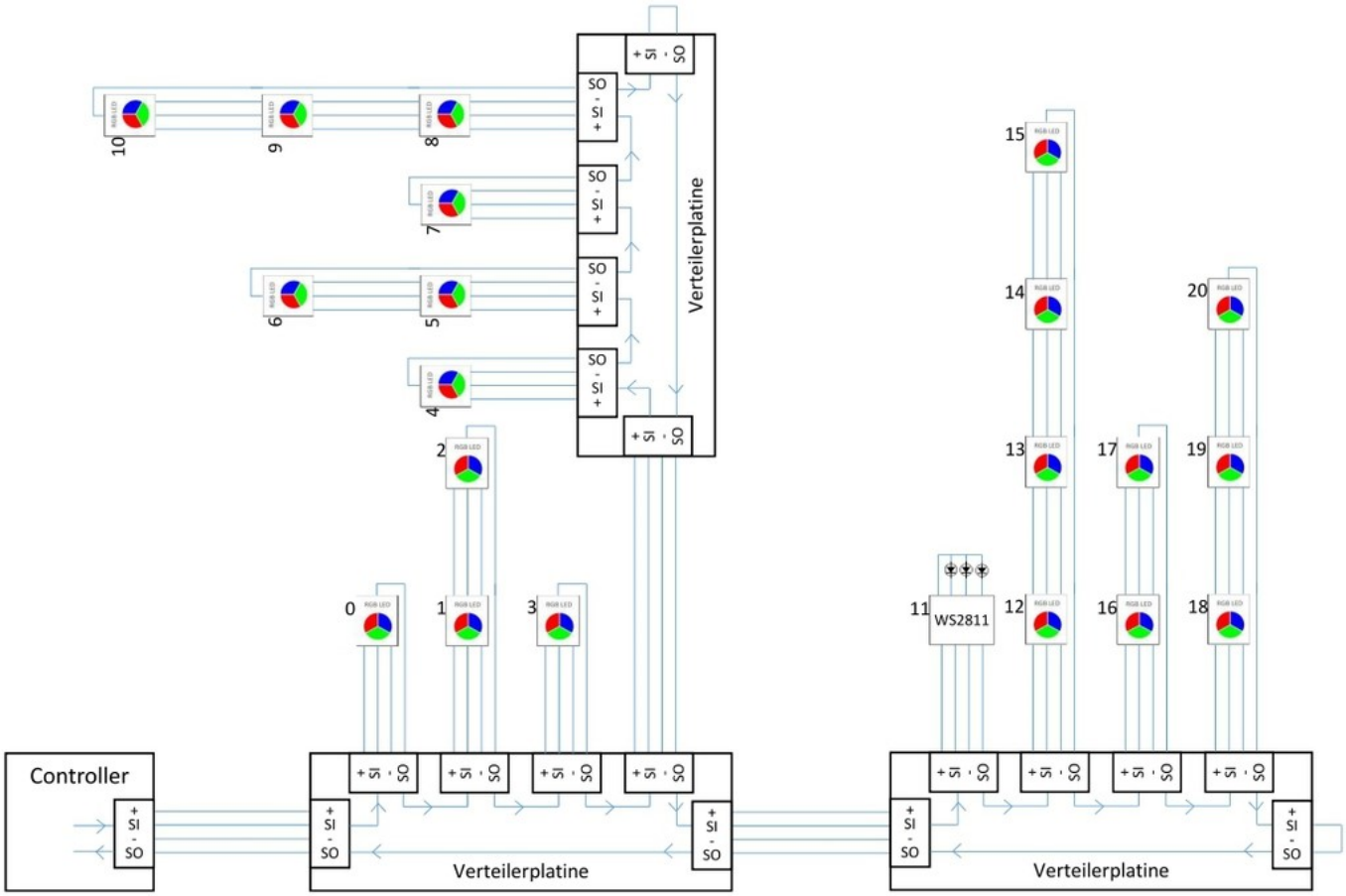


So eine Duplizierung der Signale kann man z.B. bei einer Ampel nutzen. Bei einer Kreuzung sollen die gegenüber liegenden Ampeln das gleiche Bild Anzeigen:



Ampel A = B und C = D. Hier könnte man so eine Verzweigung einbauen. Das wird man aber direkt bei den Ampeln machen und diese mit einem 4-poligen Kabel an einen Verteiler anschließen.

Doch jetzt zurück zu der Kette. 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 dieses Bild:



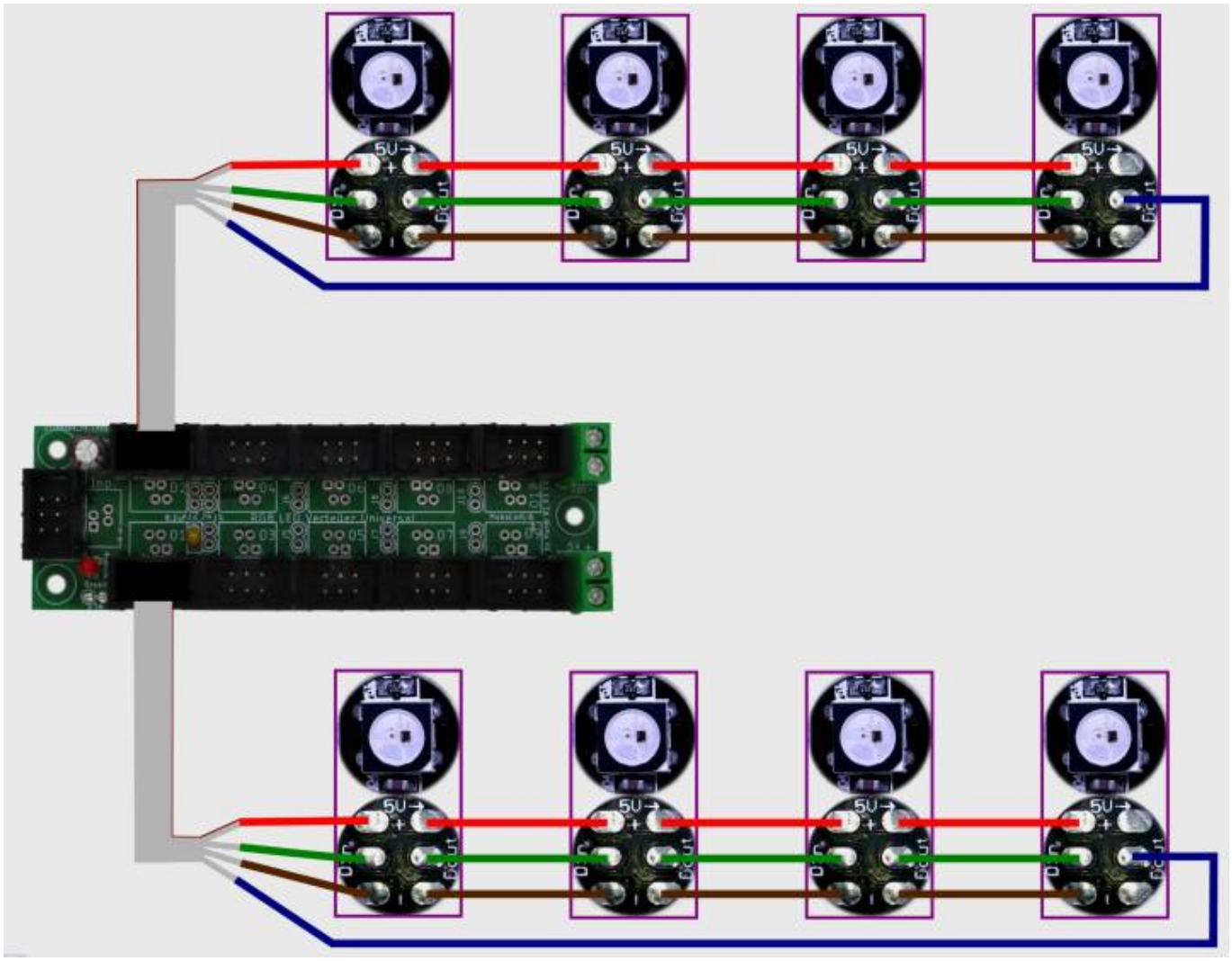
Neben den LEDs sind kleine Nummern welche die „Adresse“ der LED beschreibt. Praktischerweise gibt es die „Intelligenz“ der RGB-LEDs auch als einzelnes IC ohne LED bzw. als kleine Platine mit montiertem IC. Das wird im Rahmen der MobaLedLib genutzt um andere Dinge anzusteuern: Servos, Motoren, Relais, Soundmodule. Diese WS2811-Schaltkreise verhalten sich wie LEDs und haben Anschlüsse an denen man dann einzelne LEDs - oder eben andere Verbraucher - anschließen kann.

Eine andere Versinnbildlichung wäre diese:

Das kannst du dir vorstellen wie einen langen Zug mit Güterwagen. Der Zug hält an der ersten LED und der erste Wagen wird abgekoppelt, die LED schickt den Zug, mit neuer Kraft zur zweiten. Dort wird der zweite Wagen, der nun ja der Erste ist, abgekoppelt. Und so geht das reihherum bis die Schublok wieder im Depot angekomm ist und sagt: „Arbeit verrichtet, Chef.“

WS2812

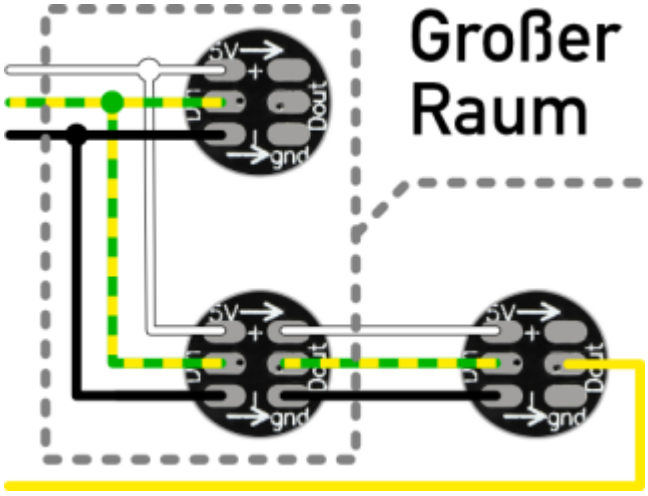
Hier ein Anschlussbeispiel für zwei Ketten mit WS2812B.
Dabei sind jeweils vier RGB-LEDs in einer Reihe geschaltet.
Von der letzten LED geht ein Kabel zurück zum Flachbandkabel, damit das LED-Signal wieder zurück zum Verteiler kommt.
Jede LED ist dabei mit Ihrer Vorder- und Rückseite abgebildet.



WS2812 parallel

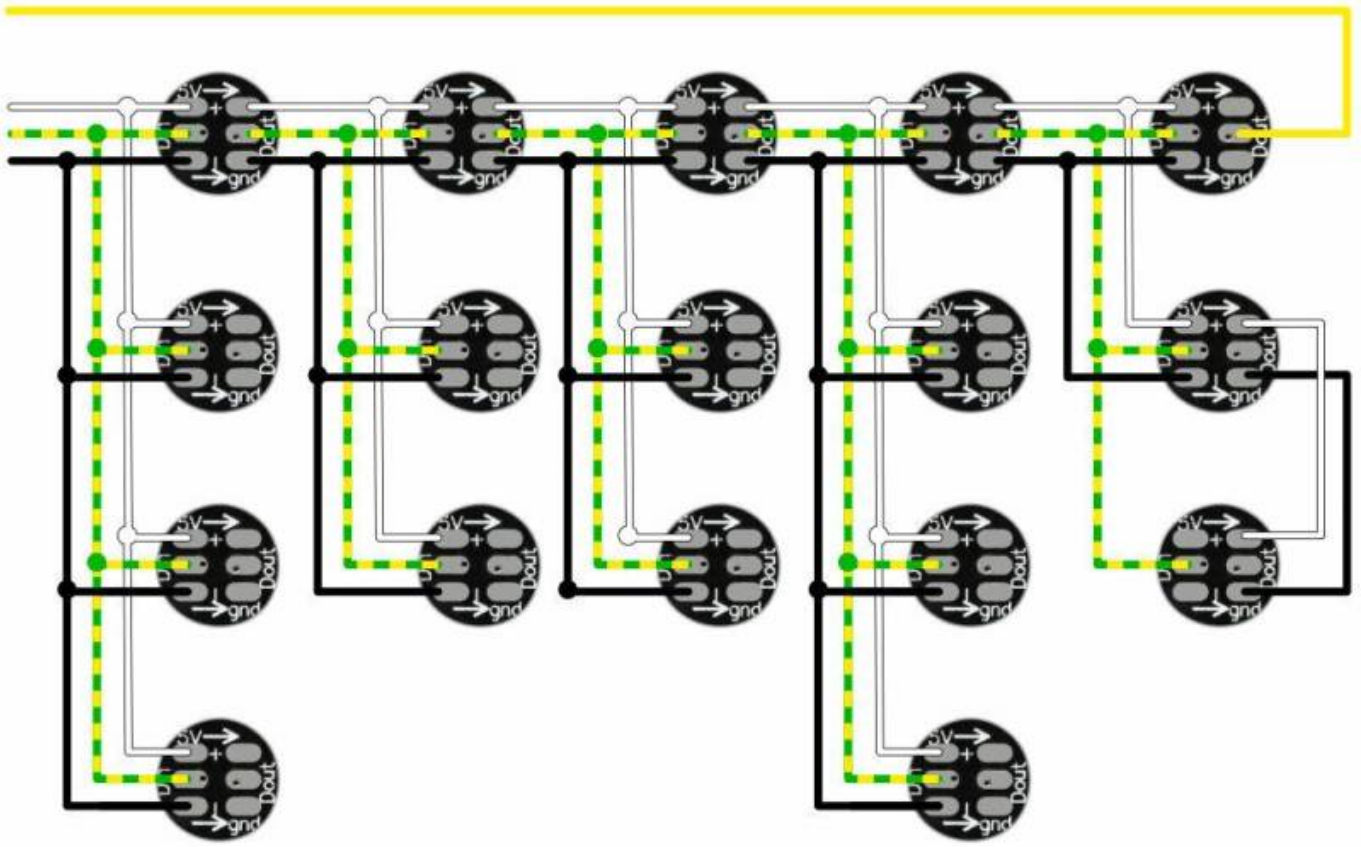
Hier ein Anschlussbeispiel für zwei parallel geschaltete WS2812B.

Wenn in einem Raum zwei oder mehr RGB-LEDs gebraucht werden, um den Raum beispielsweise gleichmäßig auszuleuchten, können WS2812B-LEDs ganz einfach parallel angeschlossen werden. Diese parallel angeschlossenen LEDs bekommen alle das gleiche Signal für den Dateneingang und reagieren somit auch synchron.



Das folgende Bild zeigt fünf große Räume. Im Programm Generator werden nur die fünf LEDs der oberen Reihe definiert. Alle LEDs ab der zweiten Reihe abwärts sind für den Arduino unsichtbar. Das ist ein zusätzlicher Vorteil ggü. dem CopyLED-Befehl, denn im gezeigten Beispiel unten belegen 12 der 17 LEDs keinen Speicherplatz auf dem Arduino.

Diese Anschlussmöglichkeit ist immer dann empfehlenswert, wenn die LED immer und in jedem Fall exakt das gleiche tun soll wie eine andere. Es bietet sich auch bei einem Farbwechsel an, der bspw. eine Burg oder ein Viadukt beleuchtet. Wenn von vornherein klar ist, dass alle LEDs, die das Objekt anstrahlen immer dieselbe Farbe erzeugen sollen, könnte man hier ganz viele LEDs zu einer zusammenfassen.



ACHTUNG: Das Signal für den Datenausgang (D Out) darf in diesem Fall nur von einer dieser parallel geschalteten WS2812B weiter gereicht werden.