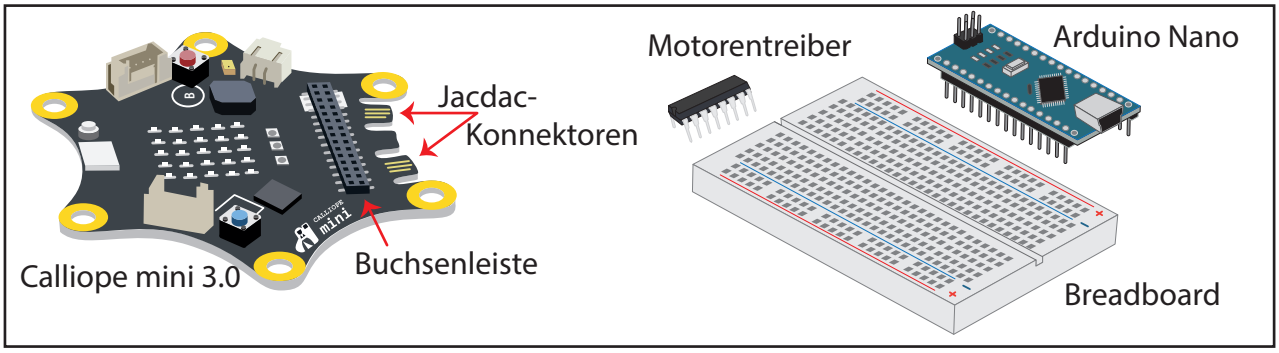
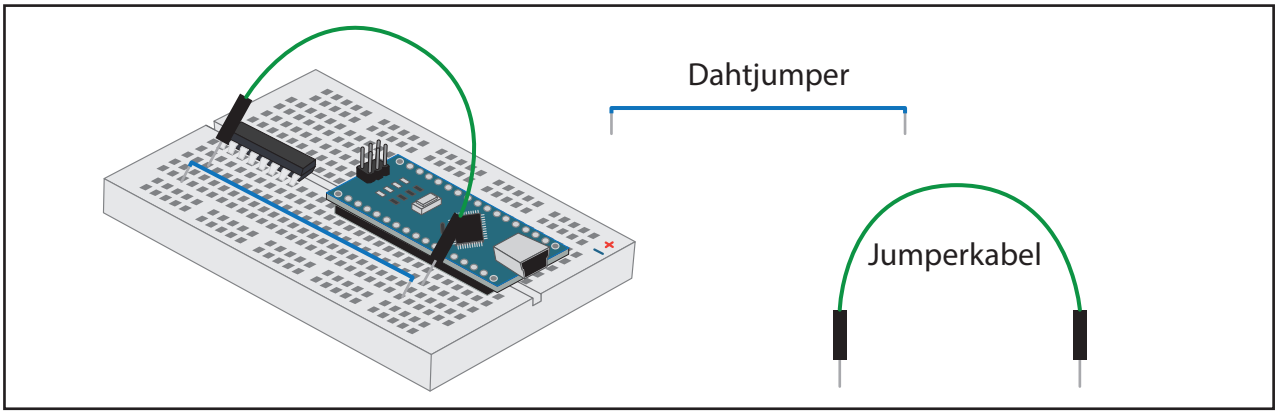
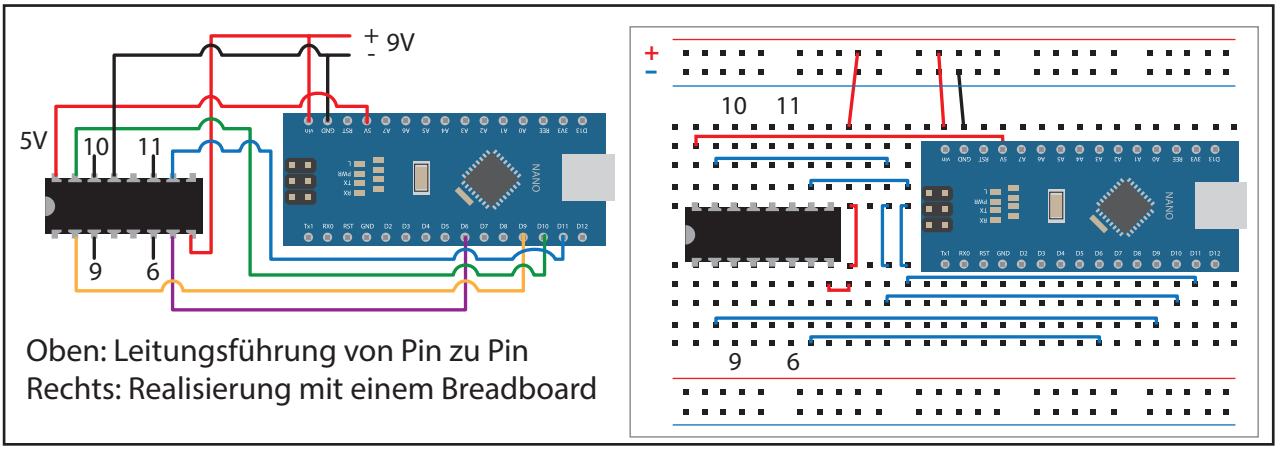
In dem Grundschulprojekt „Vom rollfähigen zum programmierbaren Fahrzeug“ (veröffentlicht im Klinkhardt Verlag, ISBN 978-3-7815-2433-0) lässt sich der letzten Teil statt mit einem Calliope mini auch mit mit einem preisgünstigeren Arduino realisieren. Wie das konkret gelingen kann, wird im Folgenden geklärt.



Der Calliope mini 3.0 bringt den für das Projekt nötigen Motorentreiber mit. Beim Arduino muss er nachgerüstet werden. Dafür werden neben dem Treiber Steckverbinder und ein Breadboard gebraucht.



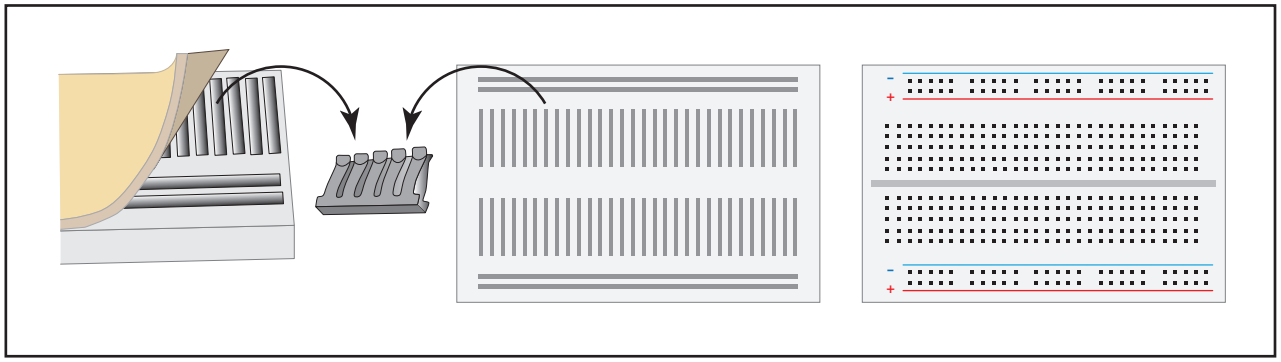
Das Herstellen der Verbindungen mit Drahtjumpern ist mit etwas Mühe verbunden, weil die auf eine passende Länge gebracht werden müssen. Für eine erste Erprobung empfehlen sich deshalb Jumperkabel. Ihr Einsatz führt aber bald zu einem schwer durchschaubaren Kabelgewirr. Wann immer möglich, sollten deshalb die Drahtjumper verwendet werden. Sie führen neben der besseren Durchschaubarkeit auch zu einer höheren Funktionssicherheit.



Oben: Leitungsführung von Pin zu Pin
Rechts: Realisierung mit einem Breadboard

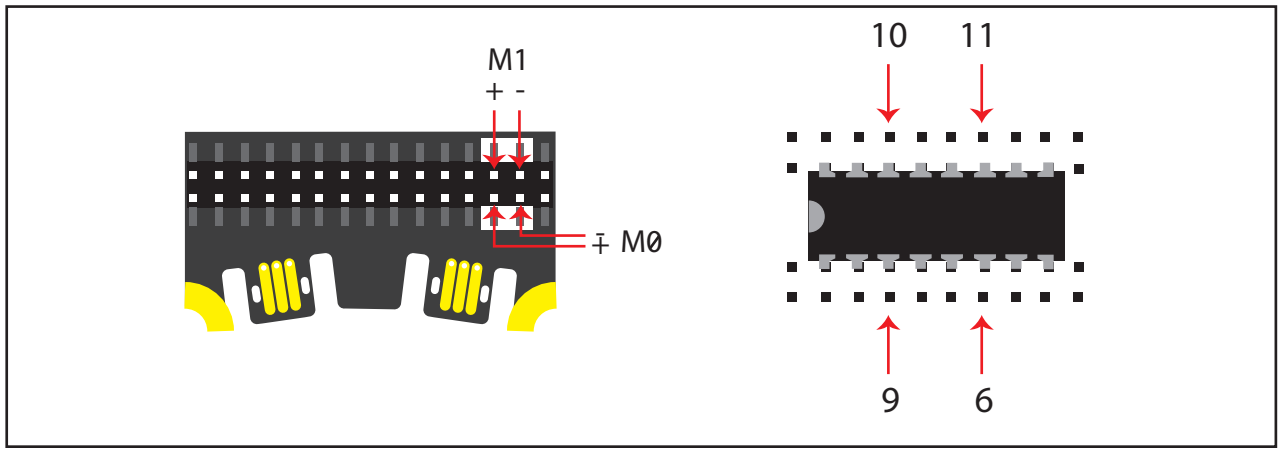
Größe und Abstände der Pins am Arduino Nano und am Motorentreiber sind genormt und lassen sich so problemlos in das Breadboard einstecken. Die einzelnen Anschlüsse müssen dabei elektrisch voneinander getrennt bleiben.

Das gelingt, wenn die Bauteile mittig auf dem Breadboard platziert werden.

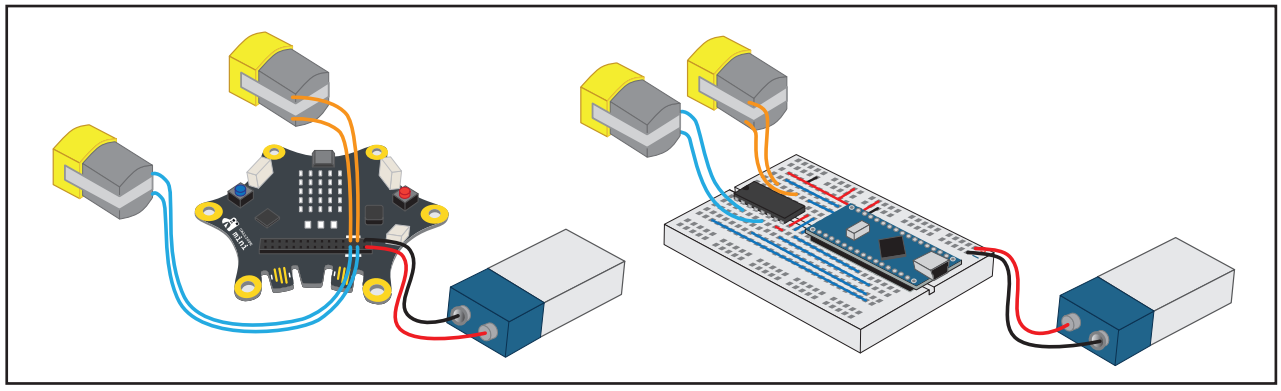


Wenn man das doppelseitige Klebeband von der Unterseite abzieht, werden Kontaktbleche sichtbar. In der Zeichnung oben ist ein solches Blech dreidimensional, in der Draufsicht sind sie dunkler, die isolierenden Teile heller dargestellt. Dieses Setting ermöglicht es, über die Löcher an jeden Pin mehrere Anschlüsse zu legen. Die Randstreifen dienen zu Stromversorgung.

Sind Motorentreiebr und Nano verkabelt, ist eine die Vergleichbarkeit der beiden Systeme gegeben.



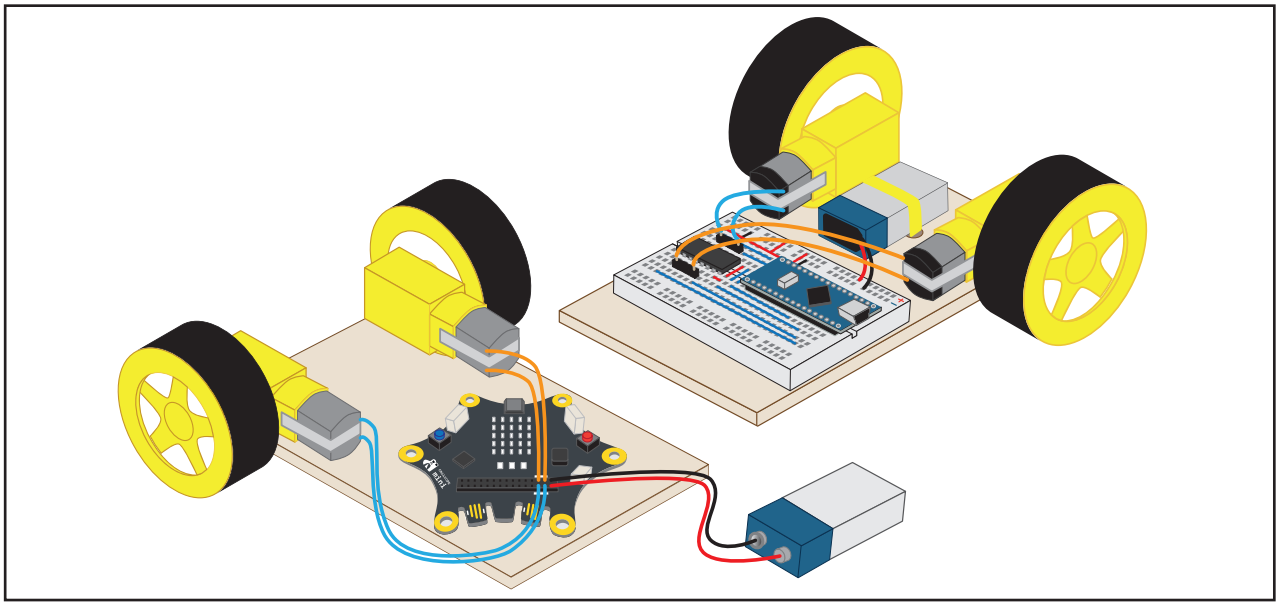
Für die Motoren sind 4 Anschlüsse vorgesehen, 2 für den einen und 2 für den anderen Motor. Die Beschriftung + und - an den Pins des Calliope ist dabei irreführend. Plus- und Minuspol werden über eine noch zu erläuternde Programmierung festgelegt.



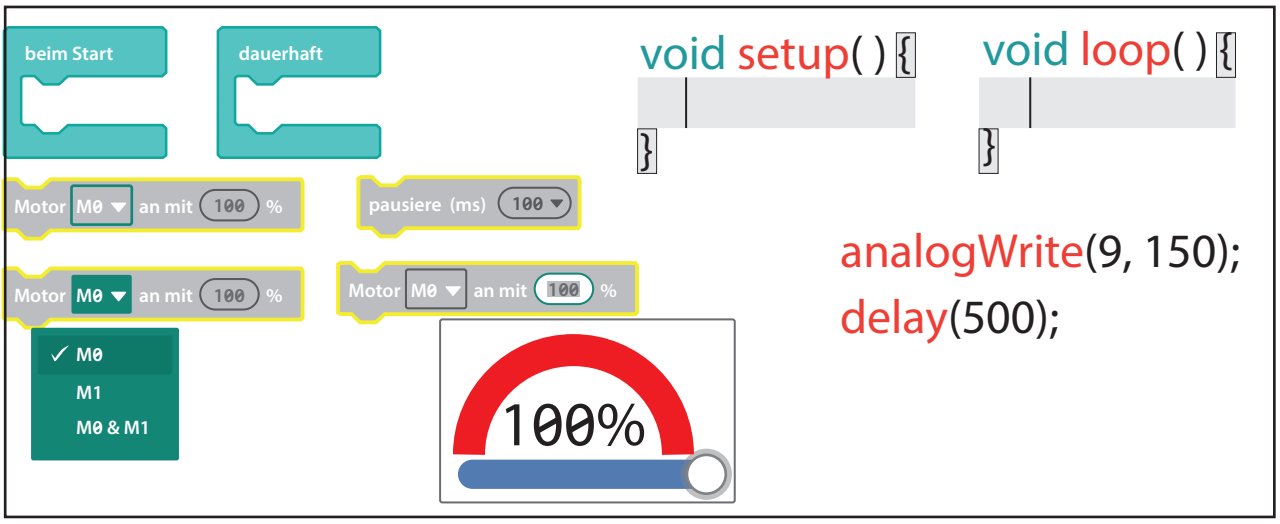
Strom wird von einer 9 V Blockbatterie geliefert. Sie hat einen Plus- (+) und einen Minuspol (-). Damit die Motoren laufen können, muss ein Anschluss mit (-) und der andere mit (+) verbunden sein. Sollen die Motoren gleichsinnig drehen, um das Fahrzeug geradeaus fahren zu lassen, muss die Polung übereinstimmen. Am besten kann das durch Ausprobieren herausgefunden werden.

Wenn die Motoren nicht wie gewünscht drehen, einfach an einem Motor die Anschlüsse tauschen.

Motoren und Batterien sind mit Treiber und Mikrocontroller über Kabel verbunden. Die Batterie kann zwischen den Motoren platziert und z. B. mit einem Kabelbinder gesichert werden.



Zur ersten Erprobung soll das Fahrzeug so programmiert werden, dass es ein kurzes Stück vor- und die gleiche Strecke zurückfährt. Für den Calliope mini steht dazu Block-, für den Arduino Textcode zur Verfügung . Hier die benötigten Codebausteine: links Block, rechts Textcode.

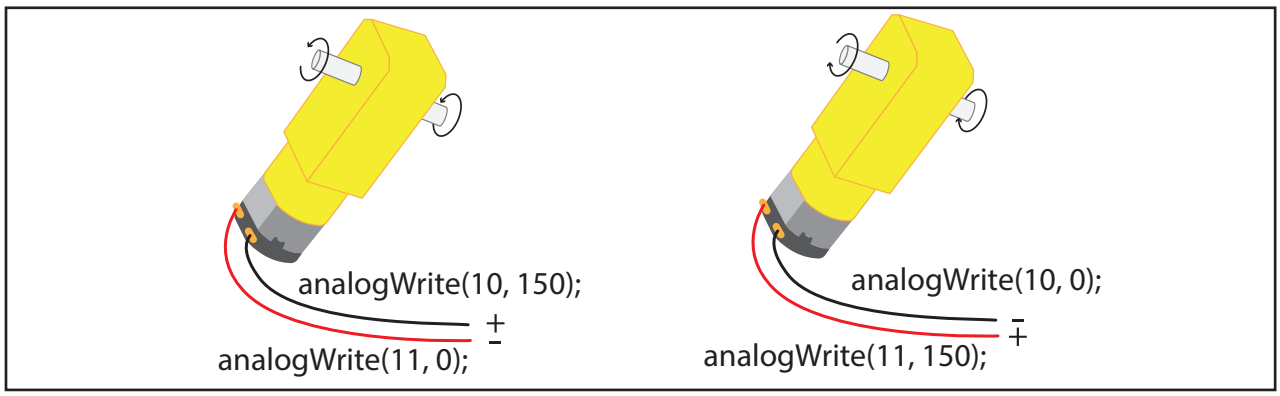


Zur Lösung der Aufgabe müssen beide Motoren so programmiert werden, dass sie gleichzeitig starten, nach kurzer Zeit in den und Rücklauf versetzt werden und dann stoppen.

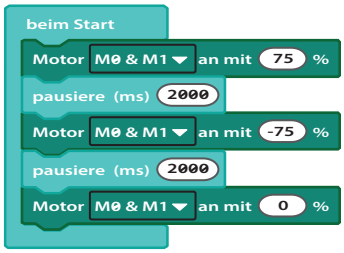
Beim Calliope zum Betreiben von Motoren ein Blockbaustein vorgesehen. Mit diesem ist es möglich, auf jeden Motor einzeln oder auf beide gleichzeitig zuzugreifen. Vor- und Rücklauf können mit einem Regelschieber oder durch Zahlen zwischen -100 und 100 eingestellt werden. 100 bedeutet dabei volle Kraft v0raus, -100 volle Kraft zurück. Eine 0 bedeutet Stillstand. Die Zahlen zwischen 0 und 100 haben Einfluss auf die Drehzahlen der Motoren und damit auf die Geschwindigkeit des Fahrzeugs.

Beim Arduino geht der Blick auf die Anschlusspins der Motoren: Einer muss auf 0, der andere

auf einen Zahlwert =/>150 gesetzt werden. Ein auf Null gesetzter Pin wird zum Minus-, der mit der größeren Zahl der Pluspol. Mit den Zahlen kann die Drehgeschwindigkeit beeinflusst werden. Beide Pins auf 0 schaltt den Motor ab. Ein Tausch der beiden Pole (Umpolung) bewirkt eine Umkehr der Drehrichtung.



Hier der zur Erfüllung der Aufgabe erforderliche Code für beide Controller:




```
void setup( ) {
  analogWrite(6, 0);
  analogWrite(9, 150);
  analogWrite(10, 0);
  analogWrite(11, 150);
  delay(2000);
  analogWrite(6, 150);
  analogWrite(9, 0);
  analogWrite(10, 150);
  analogWrite(11, 0);
  delay(2000);
  analogWrite(6, 0);
  analogWrite(10, 0);
}
```

Beide Motoren an vorwärts

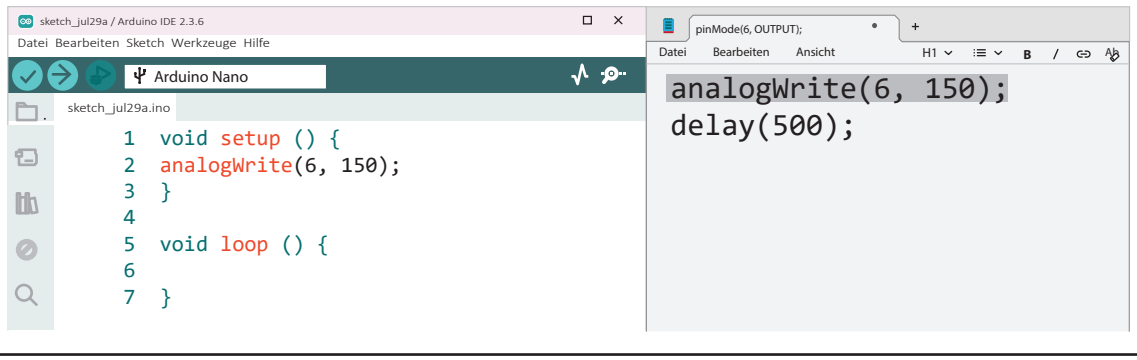
Laufdauer

Beide Motoren an rückwärts (Polung umgekehrt)

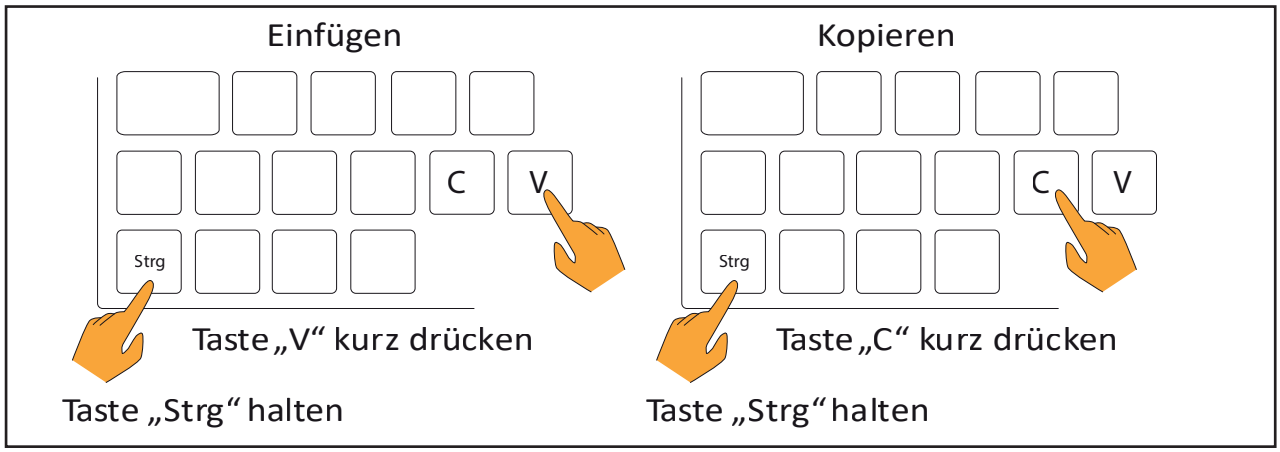
Laufdauer

Beide Motoren aus

Beim Textcode muss auf Rechtschreibung und Zeichensetzung geachtet werden. Um solche Fehler zu vermeiden, empfiehlt es sich, die Befehle in einem Texteditor vorzuhalten, sodass sie per Copy & Paste in den Editor eingefügt und auch vervielfältigt werden können. So müssen nur die Zahlen angepasst werden .



Der Befehl wird im Texteditor angewählt (das Gewählte wird grau hinterlegt) und durch Drücken der Tasten Strg + C in den Zwischenspeicher kopiert. Dann wird der Cursor in die passende Zeile der Arduino IDE gesetzt und der Code durch Drücken der Tasten Strg + V eingefügt.



Die Vorgaben „void setup“ (beim Start) und „void loop“ (dauerhaft) werden von der Arduino IDE automatisch generiert beim Aufrufen von --> Datei und --> Neuer Sketch. Die Erklärung auf Englisch sollte angewählt und durch Drücken der Entf-Taste gelöscht werden.

Weitere Details finden sich Internet unter „mint-unt.de“.