



DAS UMFANGREICHSTE STARTER KIT

ANLEITUNG FÜR UNO

V1.0.17.2.11

Packing list

 www.elegoo.com

**Servo Motor
(SG90)
1PC**



**ULN2003 Stepper
Motor Driver
Board
1PC**



**Power Supply
Module
1PC**



**GY-521
Module
1PC**



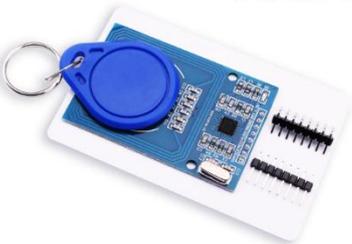
**Elegoo UNO R3
Controller
Board
1PC**



**LCD 1602 Module
(with pin header)
1PC**



**Rc522
RFID Module
1PC**



**Stepper Motor
1PC**

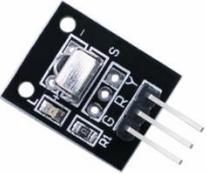


**Prototype Expansion
Board
1PC**



Contact us : service@elegoo.com

**IR Receiver
Module
1PC**



**Joystick Module
1PC**



**DHT11 Temperature
and Humidity
Module
1PC**



**Rotary Encoder
Module
1PC**



**Ultrasonic Sensor
1PC**



**Ds3231 RTC
Module
1PC**



**HC-SR501 PIR
Motion Sensor
1PC**



**Sound Sensor
Module
1PC**



**Water Level Detection
Sensor Module
1PC**



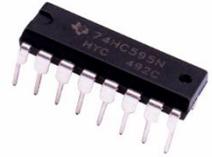
1 Digit 7-Segment
Display
1PC



4 Digit 7-Segment
Display
1PC



74HC595
1PC



L293D
1PC



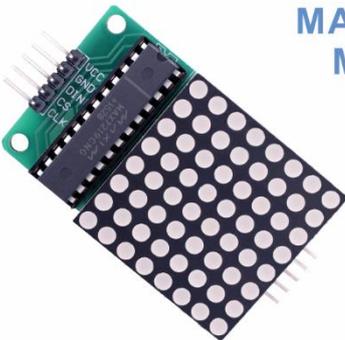
Potentiometer
(10K)
2PCS



Remote
1PC



MAX7219
Module
1PC



Active Buzzer
1PC



Passive Buzzer
1PC





65 Jumper Wire
1PC



Female-to-Male Dupont Wire
20PCS



USB Cable
1PC



830 Tie-Points Breadboard
1PC



Fan Blade and 3V DC Motor (with wire)
each
1PC



Membrane Switch Module
1PC



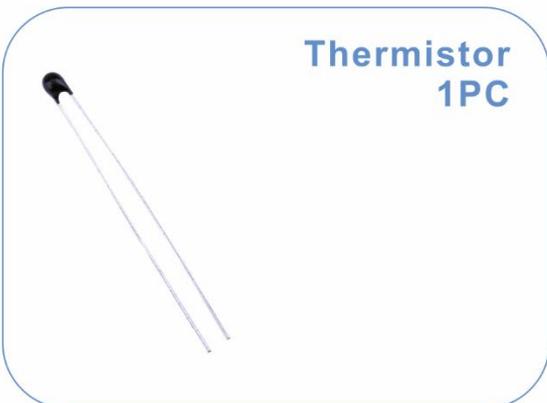
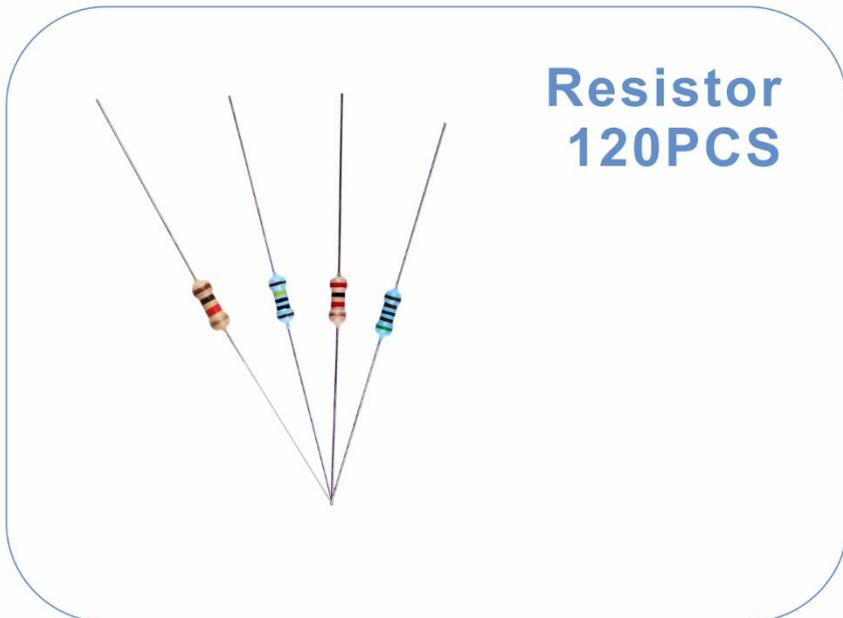
9V Battery With DC
1PC

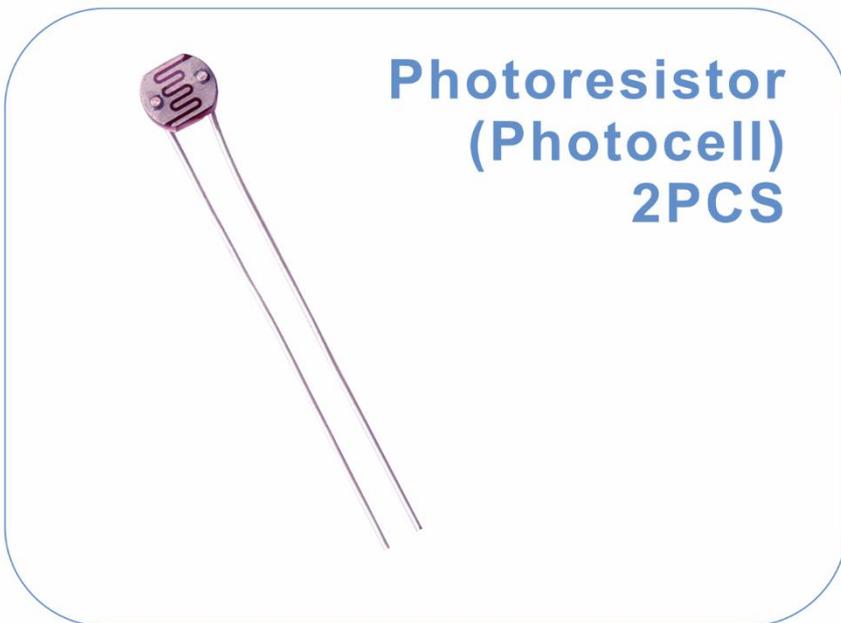
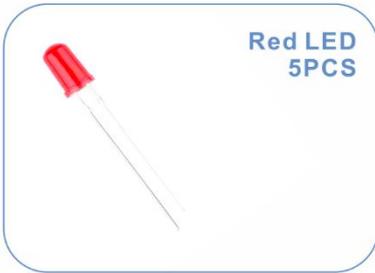


5V Relay
1PC



9V 1A Adapter
1PC





Inhalt

Lektion 0: Installation des IDE.....	10
Lektion 1: Bibliotheken einbinden/ Seriellen Monitor öffnen.....	22
Lektion 2: Blink	31
Lektion 3: LEDs	43
Lektion 4: RGB LEDs	50
Lektion 5: Digitale Eingänge	59
Lektion 6: Aktiver Buzzer	64
Lektion 7: Passiver Buzzer	68
Lektion 8: Neigungssensor.....	72
Lektion 9: Servomotor	76
Lektion 10: Ultraschallsensormodul	80
Lektion 11: Folientastaturmodul.....	85
Lektion 12: DHT11 Temperatur- und Luftfeuchtigkeitssensor	91
Lektion 13: Analoger Joystick	97
Lektion 14: Infrarotempfängermodul	102
Lektion 15: MAX7219 LED-Punktmatrixmodul	108
Lektion 16: MPU-6050 Modul	112
Lektion 17: HC-SR501 Bewegungssensor	121
Lektion 18: Wasserstandssensormodul	131
Lektion 19: Echtzeituhr-Modul.....	136
Lektion 20: Soundsensormodul.....	141
Lektion 21: RC522 RFID-Modul.....	147
Lektion 22: LCD Display	152
Lektion 23: Thermistor	157
Lektion 24: Acht LEDs per 74HC595 ansteuern.....	162
Lektion 25: Der Serielle Monitor	169
Lektion 26: Fotozelle.....	175
Lektion 27: 74HC595 mit Siebensegmentanzeige.....	180

Lektion 28:Digitale 4-Ziffer-Siebensegmentanzeige	186
Lektion 29:Gleichstrommotor	191
Lektion 30: Relais	201
Lektion 31: Schrittmotor.....	207
Lektion 32:Schrittmotor mit Fernbedienung steuern	215
Lektion 33:Schrittmotor mit Drehgeber steuern	219

Lektion 0: Installation des IDE

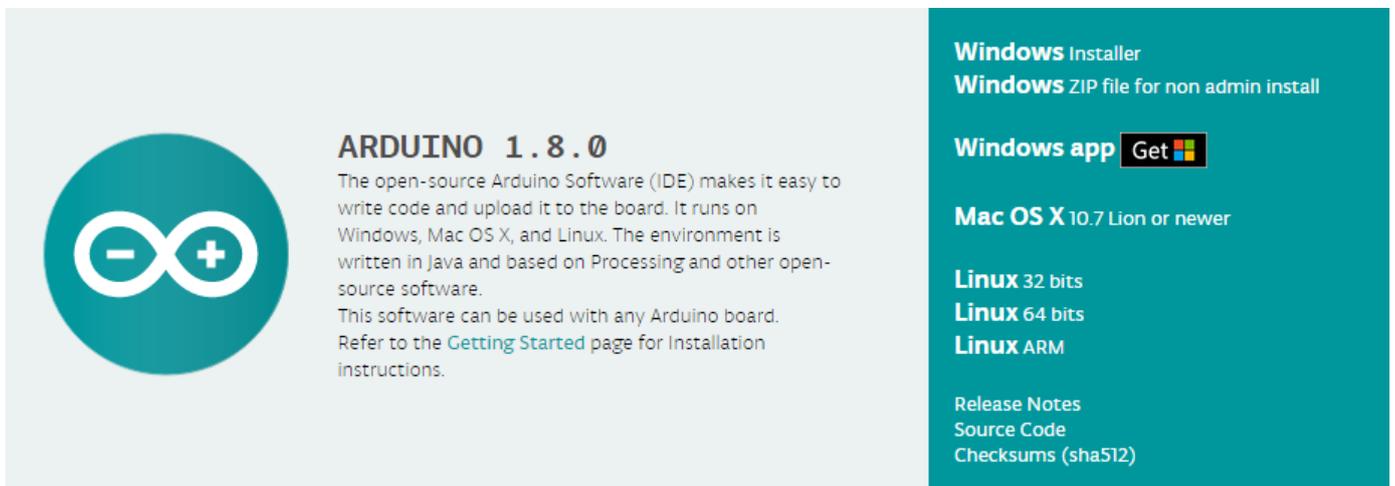
Einführung

Das Arduino IDE („Integrated Development Environment“ = Integrierte Entwicklungsumgebung) ist die Software der Arduino Plattform, mit der man programmiert.

In dieser Lektion werden Sie lernen, wie man den Computer zur Verwendung mit Arduino einrichtet, um die anschließenden Lektionen erfolgreich durchführen zu können.

Die Arduino Software, die Sie zum Programmieren des Arduino benutzen werden, ist für Windows, Mac und Linux verfügbar. Die Installation ist für jede der drei Plattformen unterschiedlich und einige Dinge müssen manuell eingestellt werden, um die Software einzurichten.

Schritt 1: Gehen Sie auf die Seite <https://www.arduino.cc/en/Main/Software> und suchen Sie diese Anzeige:



ARDUINO 1.8.0

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

Windows Installer
Windows ZIP file for non admin install

Windows app [Get](#)

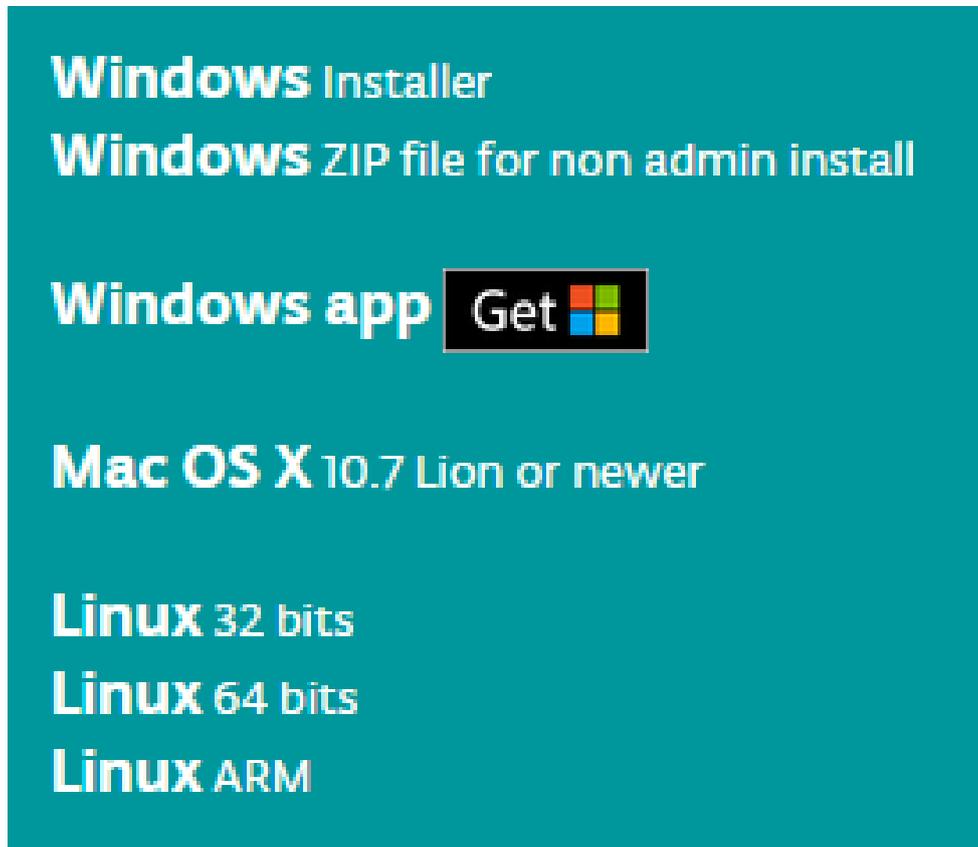
Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

[Release Notes](#)
[Source Code](#)
[Checksums \(sha512\)](#)

Auf dieser Website finden Sie immer die aktuellste Version der Software, daher kann sich die Version im Bild von der Ihnen angezeigten unterscheiden.

Schritt 2 : Laden Sie die für Ihr Betriebssystem vorhergesehene Variante der Entwicklungsumgebung herunter. Als Beispiel nehmen wir die Windows Variante:



The screenshot shows a teal background with white text. At the top, it says "Windows Installer" and "Windows ZIP file for non admin install". Below that, it says "Windows app" next to a "Get" button with the Windows logo. Further down, it lists "Mac OS X 10.7 Lion or newer", "Linux 32 bits", "Linux 64 bits", and "Linux ARM".

Klicken Sie auf *Windows Installer*.

Support the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). Learn more on how your contribution will be used.



The screenshot shows a light gray background. On the left, there are three stylized robot characters representing different Arduino boards. To their right, there is text: "SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 8,808,272 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!". Below the text are six circular buttons with the amounts: "\$3", "\$5", "\$10", "\$25", "\$50", and "OTHER". At the bottom right, there are two buttons: "JUST DOWNLOAD" and "CONTRIBUTE & DOWNLOAD".

Klicken Sie auf *JUST DOWNLOAD*.

Falls es Ihnen nicht möglich ist die neueste Version von der Arduino Website herunterzuladen, können Sie auch die von uns bereitgestellte Version 1.8.0 verwenden, die Sie im Materialpaket finden. Dies ist die aktuellste Version zum Zeitpunkt der Veröffentlichung dieser Anleitung.

 [arduino-1.8.0-linux32.tar.xz](#)

 [arduino-1.8.0-linux64.tar.xz](#)

 [arduino-1.8.0-macosx.zip](#)

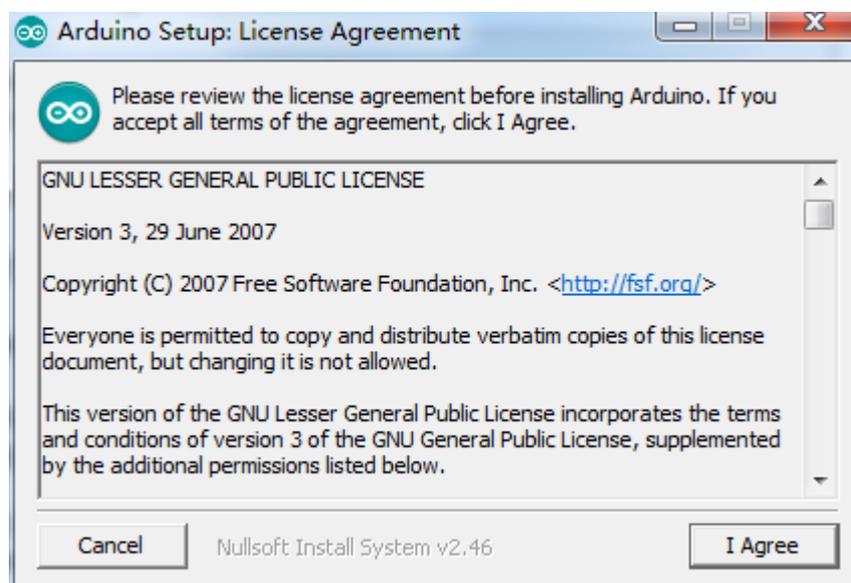
 [arduino-1.8.0-windows.exe](#)

 [arduino-1.8.0-windows.zip](#)

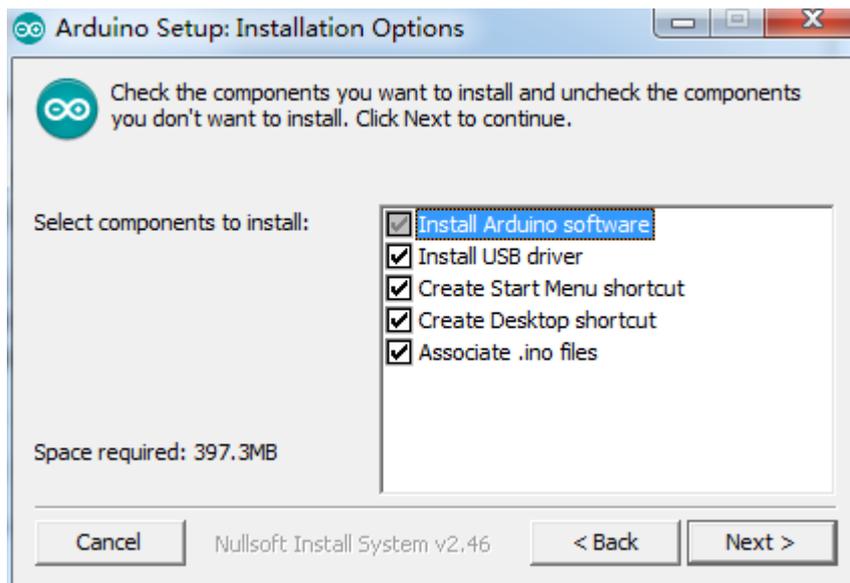
Arduino IDE installieren (Windows)

Installation mit dem Installationspaket (.exe) - *empfohlen* -

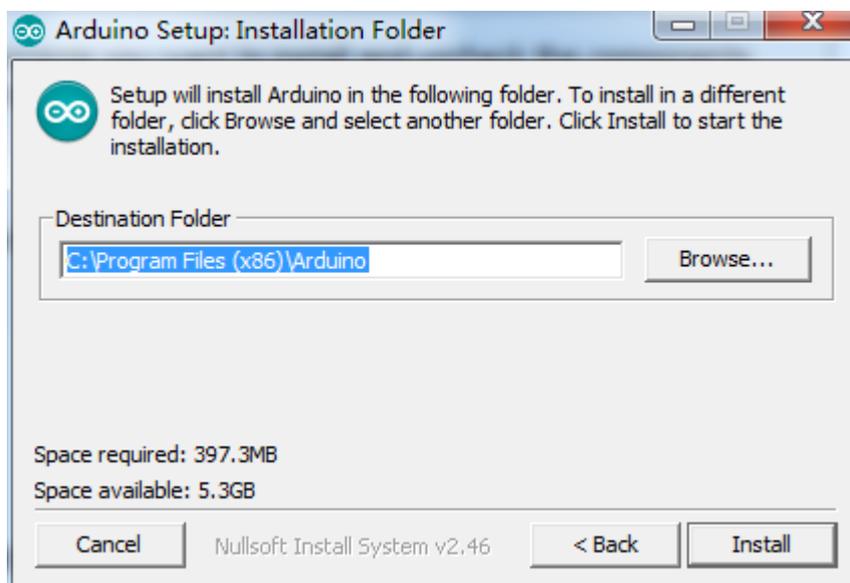
 [arduino-1.8.0-windows.exe](#)



Klicken Sie auf *I Agree*, um fortzufahren.

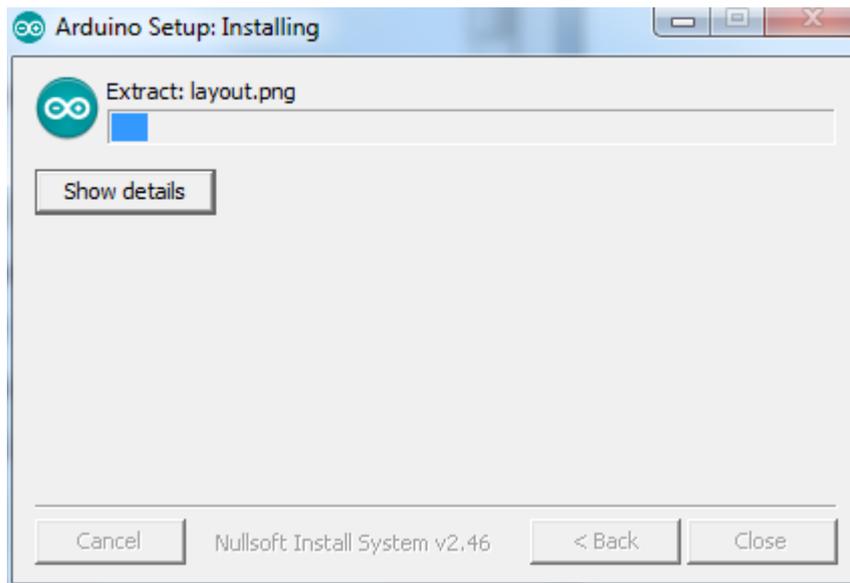


Klicken Sie auf *Next*.

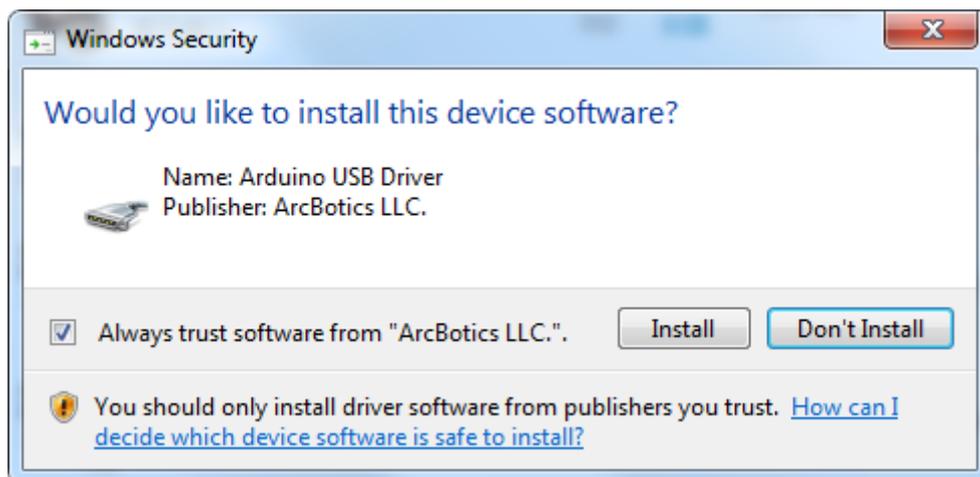


Fortgeschrittene können *Browse...* anklicken, um den Installationspfad zu ändern. Sie können diesen der Einfachheit halber aber auch einfach so belassen, wie er standardmäßig eingetragen ist.

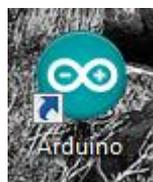
Klicken Sie auf *Install*, um die Installation zu starten.



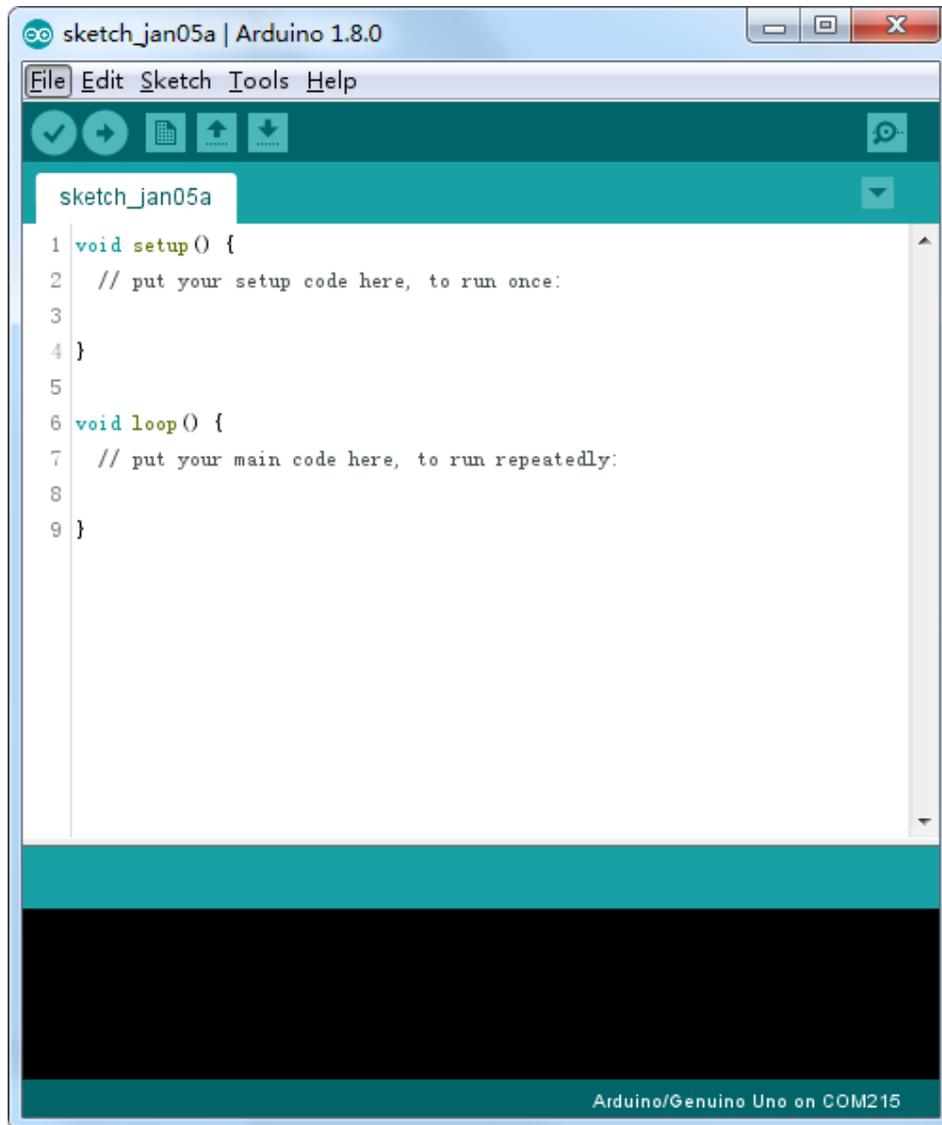
Zum Schluss erscheint diese Meldung. Klicken Sie auf *Installieren* und Sie sind fertig.



Schließlich erscheint folgendes Symbol auf dem Desktop.



Doppelklicken Sie auf das Symbol zum Starten der Arduino Entwicklungsumgebung.



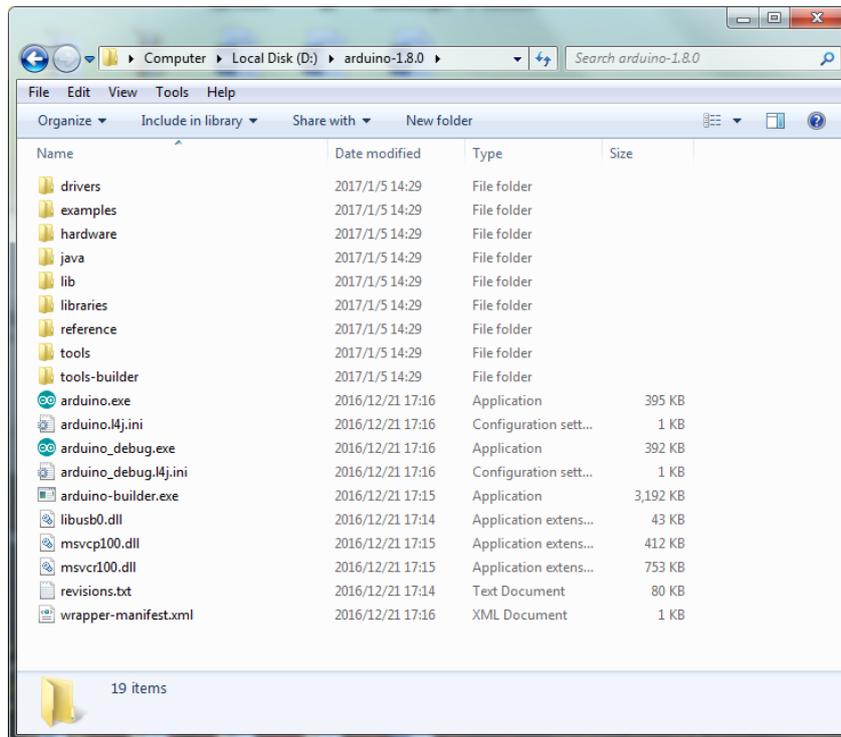
Sie haben die Installation nun erfolgreich abgeschlossen und können zu Lektion 1 gehen. Nachfolgend werden Installationsmethoden für andere Betriebssysteme beschrieben, die Sie überspringen können.

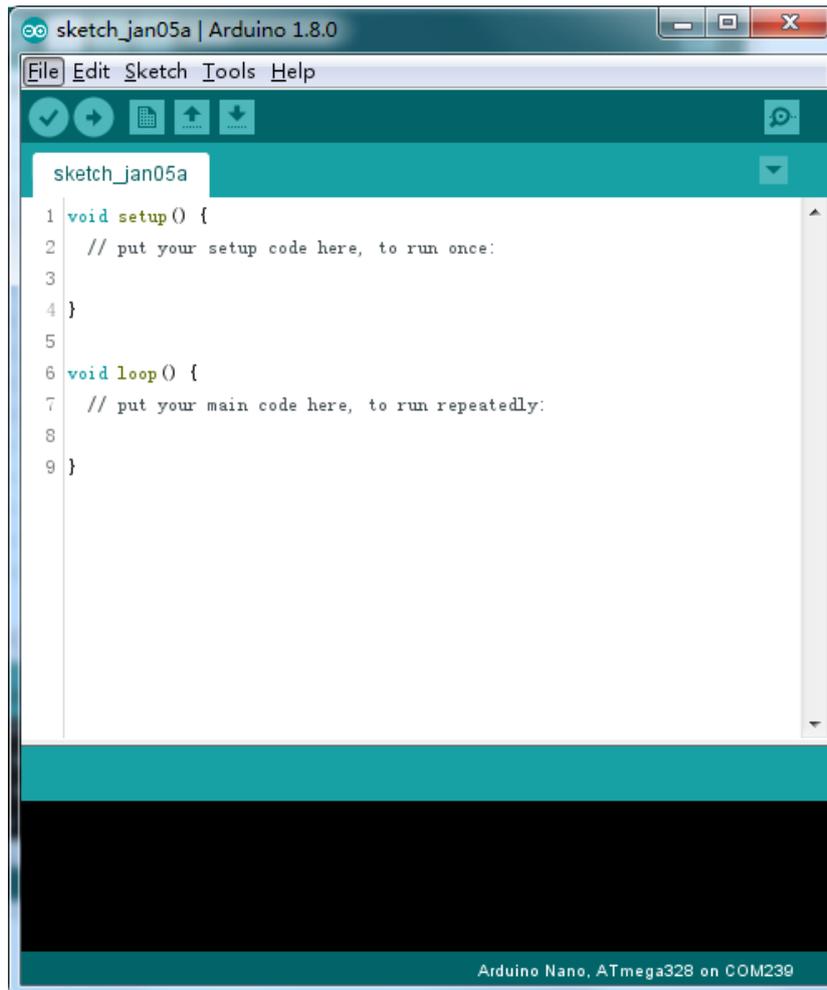
Installation mit dem ZIP-Archiv (.zip) - Für Fortgeschrittene -



Entpacken Sie das Archiv in einen beliebigen Ordner, in dem Sie das Programm

installiert haben wollen. Dies ist Ihr „Arduino-Ordner“. Doppelklicken Sie auf *arduino.exe* im Arduino-Ordner, um die Entwicklungsumgebung zu starten.





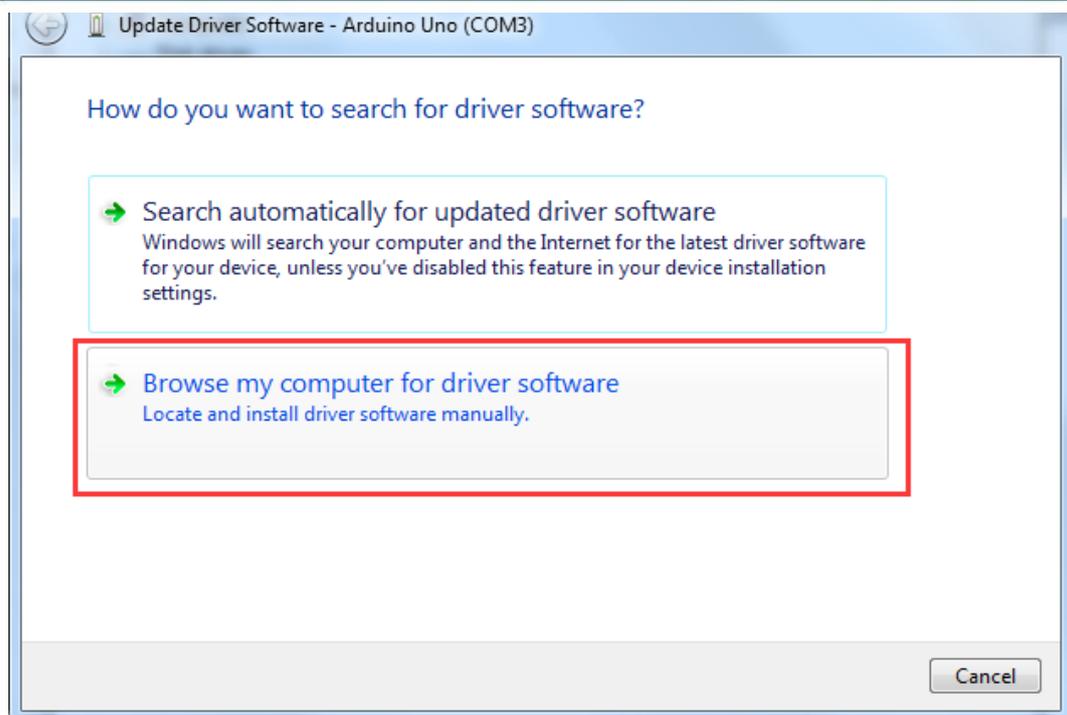
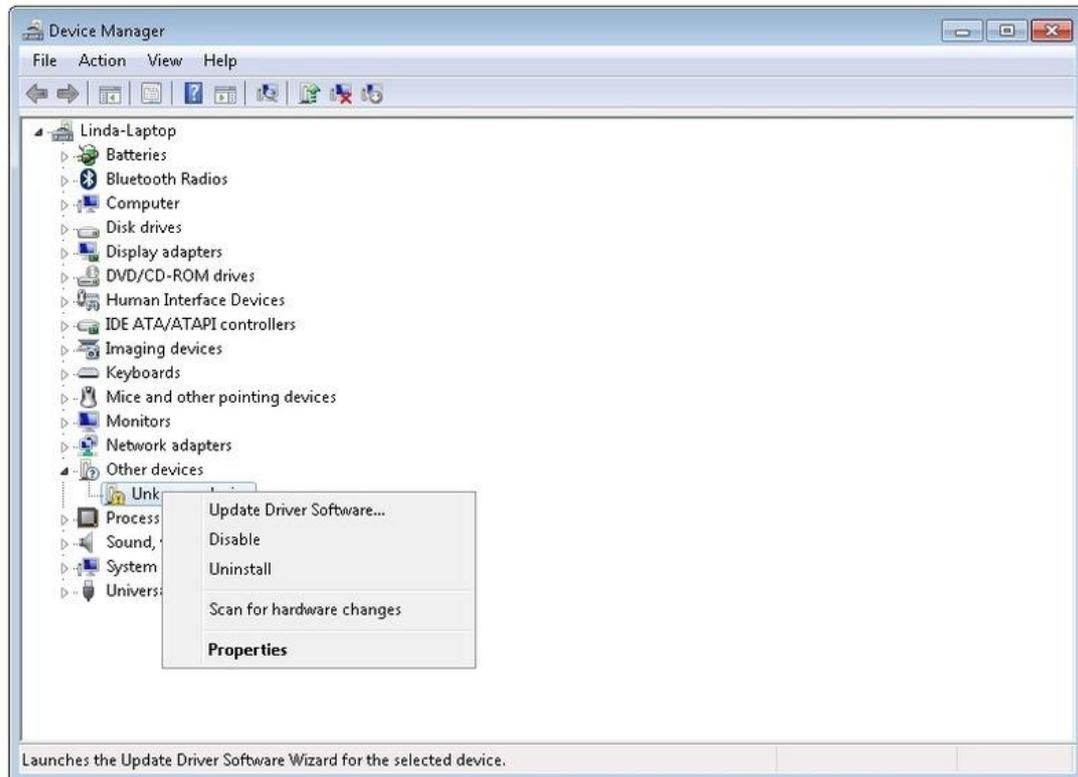
Die Installation war einfach, doch bei dieser Methode müssen die Treiber manuell installiert werden.

Der Arduino Ordnerenthält neben der Entwicklungsumgebung auch die nötigen Treiber, die es ermöglichen den Arduino mit Ihrem Computer via USB-Kabel zu verbinden. Wir schließen die Entwicklungsumgebung wieder und installieren zuerst die USB Treiber.

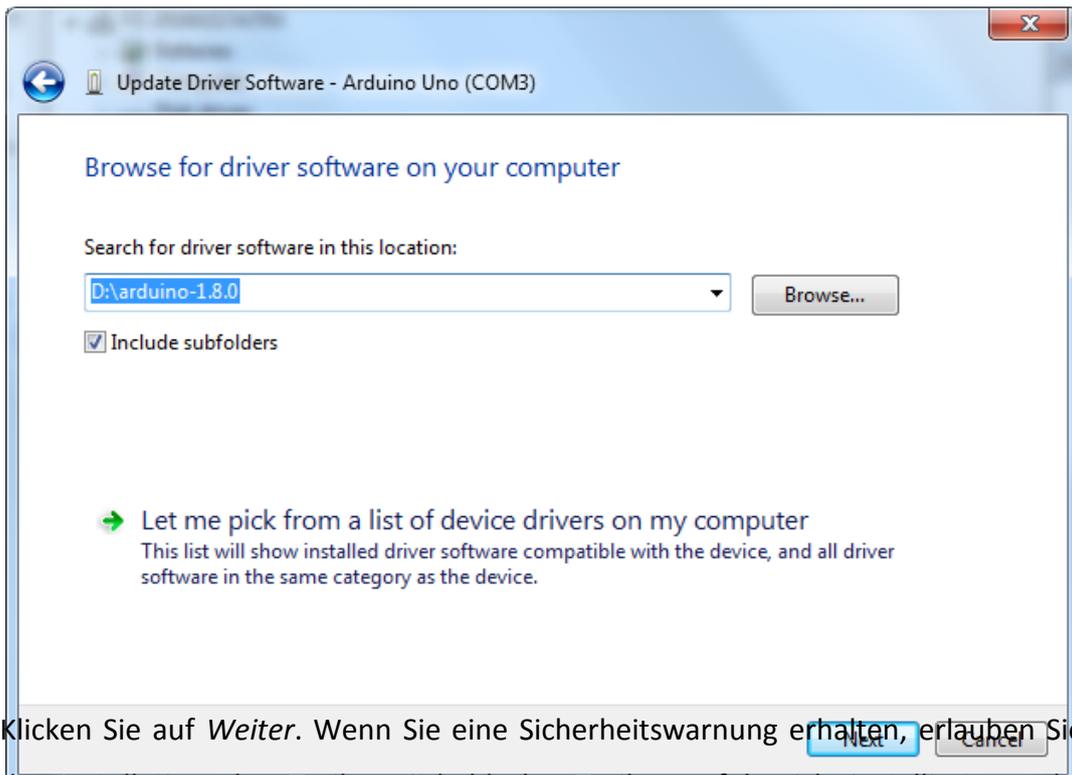
Verbinden Sie das beigelegte USB-Kabel mit dem Arduino und das andere Ende mit dem Computer. Die Power LED des Arduinos leuchtet auf und möglicherweise erhalten Sie eine Nachricht von Windows, dass ein unbekanntes Gerät verbunden wurde. Ignorieren Sie diese Nachricht und schließen alle Anfragen von Windows einen Treiber automatisch installieren zu wollen.

Die sicherste Methode, die USB-Treiber zu installieren, ist über den Geräte-Manager. Dieser kann abhängig von der Windows Version auf unterschiedliche Weisen geöffnet werden. Suchen Sie einfach im Startmenü nach „Geräte-Manager“ und Sie sollten fündig werden. Alternativ können Sie die

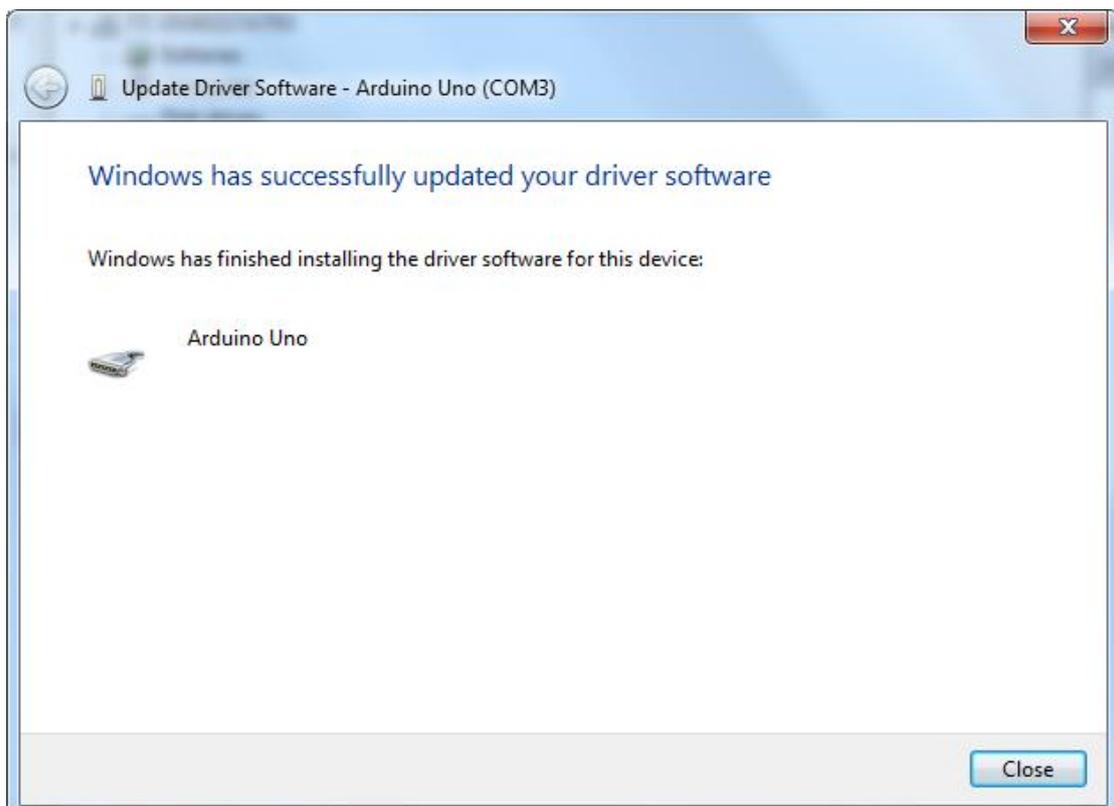
Systemsteuerung öffnen und klicken dort auf *Hardware und Sound*. Unter dem Punkt *Geräte und Drucker* finden Sie ebenfalls den Eintrag zum Geräte-Manager. Ist der Geräte-Manager geöffnet, sehen Sie eine ähnliche Ansicht, wie die untere. Unter „*Andere Geräte*“ sollten Sie einen Eintrag mit einer gelben Warn-Markierung sehen. Dies ist Ihr Arduino.



Machen Sie einen Rechtsklick auf das Unbekannte Gerät und wählen Sie die erste Option (*Treibersoftware aktualisieren...*) aus. Sie haben dann die Optionen „Automatisch nach aktueller Treibersoftware suchen“ und „Auf dem Computer nach Treibersoftware suchen“. Klicken Sie auf letzteres und navigieren Sie anschließend zu Ihrem Arduino-Ordner und dann zu dem darin liegenden Ordner *drivers* (Im Beispiel ist es *D:\arduino-1.8.0\drivers*)



Klicken Sie auf *Weiter*. Wenn Sie eine Sicherheitswarnung erhalten, erlauben Sie die Installation des Treibers. Sobald der Treiber erfolgreich installiert wurde, erhalten Sie eine Bestätigung.



Sie haben die Installation nun erfolgreich abgeschlossen und können zu **Lektion 1** gehen. Nachfolgend werden Installationsmethoden für andere Betriebssysteme beschrieben, die Sie überspringen können.

Arduino IDE Installieren (Mac OS X)



Laden Sie das Archiv herunter und entpacken Sie es anschließend. Doppelklicken Sie auf die *Arduino.app* Datei im entpackten Ordner, um die Entwicklungsumgebung zu starten. Falls Sie eine Anfrage bekommen, ob *Java Runtime Library* installiert werden soll, bestätigen Sie dies. Wenn die Installation abgeschlossen ist, können Sie die Entwicklungsumgebung jederzeit durch einen Doppelklick auf die *Arduino.app* öffnen.

Arduino IDE installieren (Linux) - Fortgeschritten -



Zuerst entpacken Sie das heruntergeladene Archiv (x32 ist für 32bit Betriebssysteme und x64 für 64bit). Sie müssen das Paket dann mit dem *make install* Befehl für Ihr System kompilieren. Dies ist kompliziert und erfordert erweiterte Systemkenntnisse. Bei weiteren Fragen lesen Sie sich bitte die *README* Datei im Archiv durch oder suchen im Internet unter Angabe Ihres Betriebssystems und „*Arduino Installieren*“ nach Hilfe.

Wenn Sie Ubuntu oder ein auf Ubuntu basierendes Linux Derivat benutzen, gibt es eine einfache alternative Möglichkeit die *Arduino IDE* über das Softwarecenter zu installieren. Öffnen Sie dazu das Software Center und geben im Suchfeld „*Arduino*“ ein und klicken anschließend auf *Installieren*.

TIPP: Wenn Sie Probleme bei der Treiberinstallation unter Linux haben, lesen Sie sich bitte das Dokument „*UNO R3, MEGA, NANO DRIVER FAQ*“ durch.



UNO R3, MEGA, NANO DRIVER FAQ

Lektion 1: Bibliotheken einbinden / Seriellen Monitor öffnen

Zusätzliche Arduino Bibliotheken einbinden

Wenn Sie mit der Arduino Entwicklungsumgebung und dessen Funktionen vertraut sind, möchten Sie möglicherweise die Möglichkeiten Ihres Arduinos mit zusätzlichen Bibliotheken erweitern.

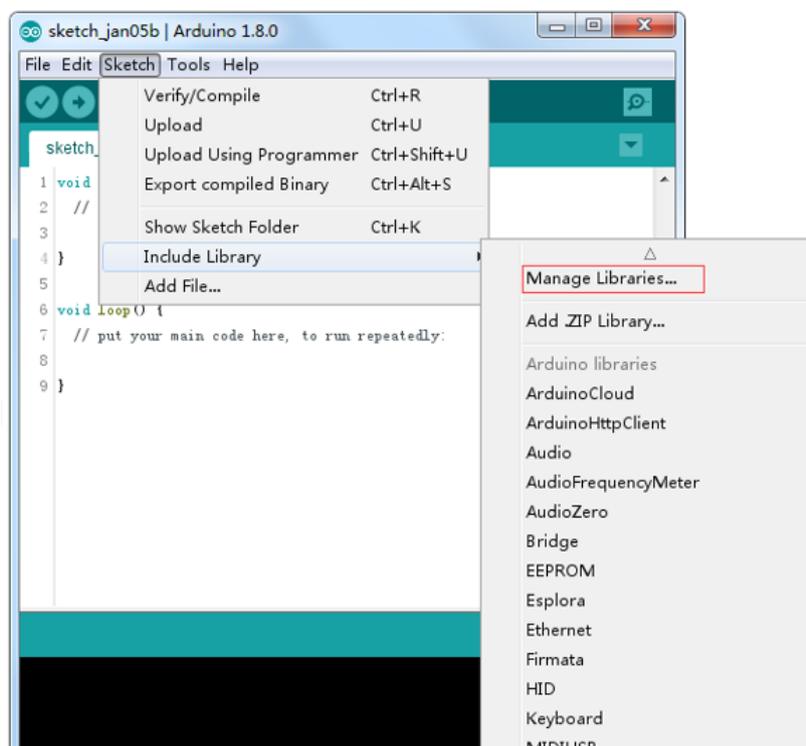
Was sind Bibliotheken?

Bibliotheken sind Sammlungen von vorprogrammiertem Code, die es enorm einfach für Sie machen beispielsweise einen Sensor, ein Display oder ein Modul einzubinden. Zum Beispiel vereinfacht die vorinstallierte *LiquidCrystal* Bibliothek die Kommunikation zu LCD-Zeichenanzeigemodulen. Es gibt hunderte von zusätzlichen Bibliotheken für Arduino im Internet, die Sie sich einfach herunterladen können. Die mit der Arduino IDE mitkommenden Bibliotheken und einige zusätzliche Bibliotheken sehen Sie später zum Vergleich. Um zusätzliche Bibliotheken benutzen zu können, müssen diese erst installiert werden.

Wie installiert man eine Bibliothek?

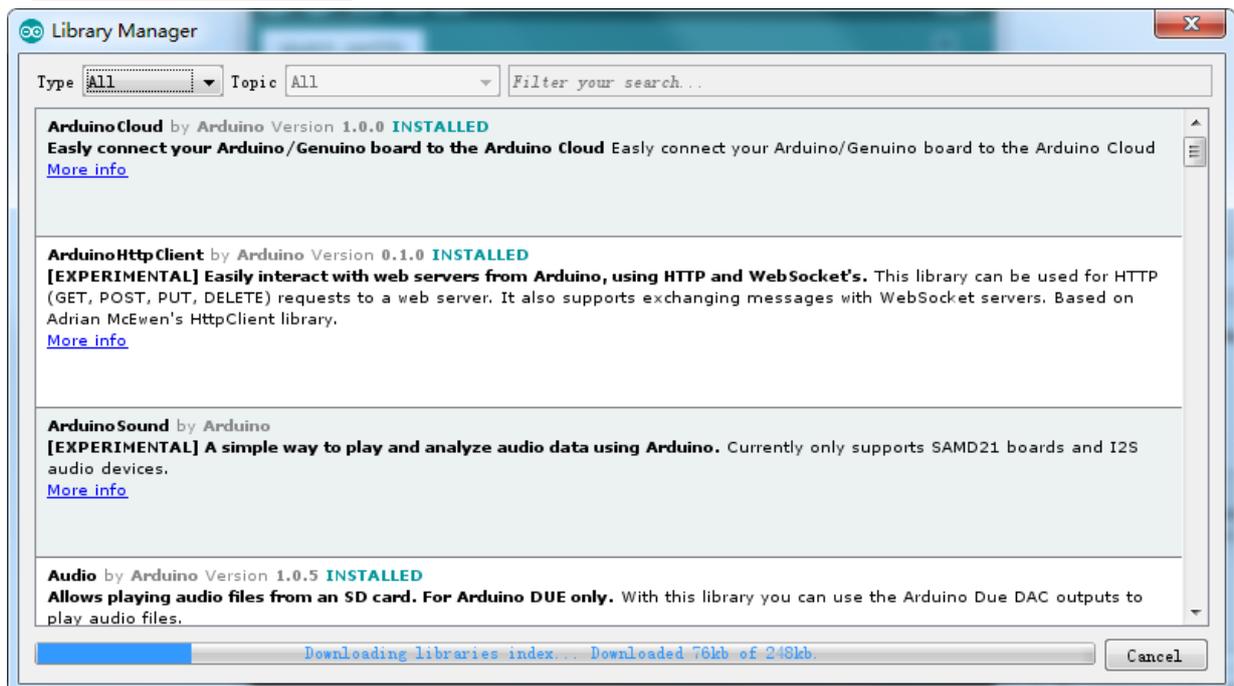
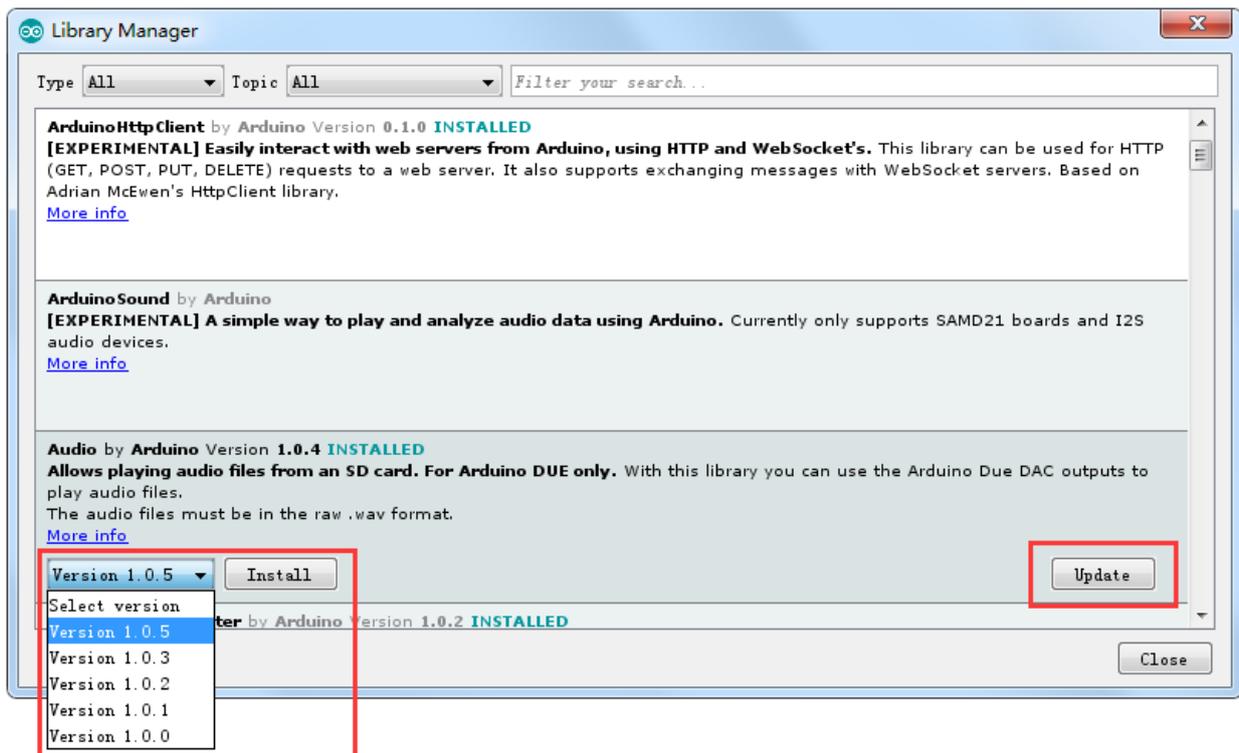
Mit Hilfe des Bibliotheksverwalters

Um eine neue Bibliothek in Ihre Arduino IDE einzubinden, können Sie den *Bibliotheksverwalter* nutzen (verfügbar seit Arduino IDE 1.8.0). Öffnen Sie die IDE und klicken unter *Sketch > Bibliothek einbinden* auf „Bibliotheken verwalten“.

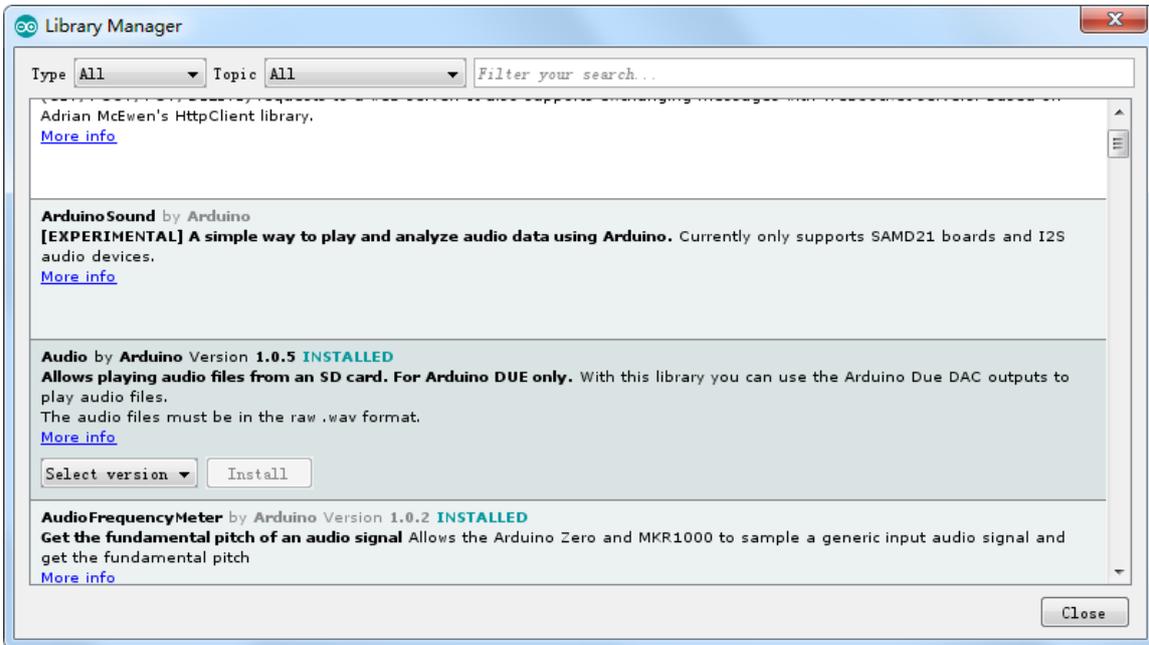


Dann wird sich der Bibliotheksverwalter öffnen und Sie finden eine Liste voller Bibliotheken, die bereits installiert sind oder heruntergeladen werden können. Als Beispiel werden wir die Bibliothek *Audio* installieren. Scrollen Sie die Liste herunter und suchen Sie die Bibliothek. Dann wählen Sie die Version aus, die Sie installieren wollen. Manchmal gibt es nur eine verfügbare Version und es lässt sich keine Version auswählen. Keine Sorge: Das ist normal.

Manchmal müssen Sie ein bisschen geduldig sein, da es sehr viele Bibliotheken gibt. Lassen Sie die Software die Liste aktualisieren und suchen Sie dann den Eintrag.



Klicken Sie zum Schluss auf *Installieren* und warten Sie, bis die Bibliothek installiert ist. Das Herunterladen kann abhängig von Ihrer Internetverbindungs-geschwindigkeit etwas dauern. Sobald die Installation abgeschlossen ist, sollte die Bibliothek im Verwalter den Zusatz „*Installiert*“ haben. Danach können Sie den Bibliotheksverwalter wieder schließen.

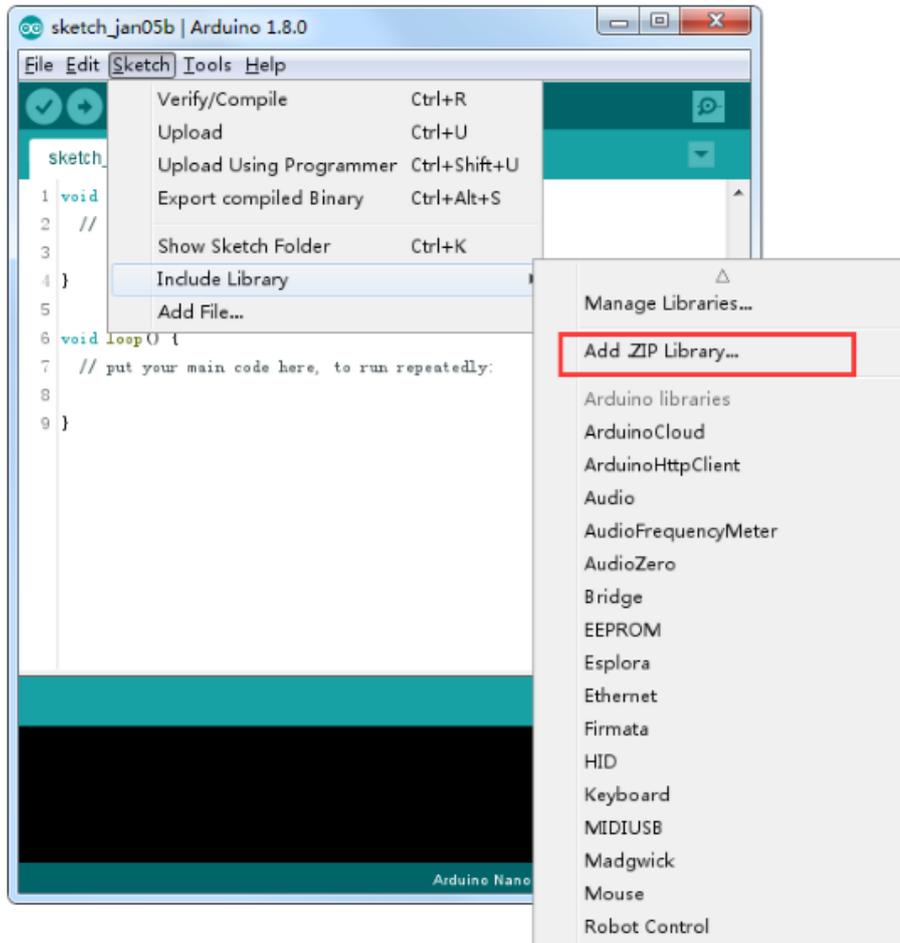


Jetzt können Sie die Bibliothek auch schon über das Einbindungs-menü hinzufügen. Wenn Sie Ihre eigene Bibliothek im Bibliotheksverwalter hinzufügen möchten, erstellen Sie einen Eintrag auf der [Github](#)-Seite von Arduino.

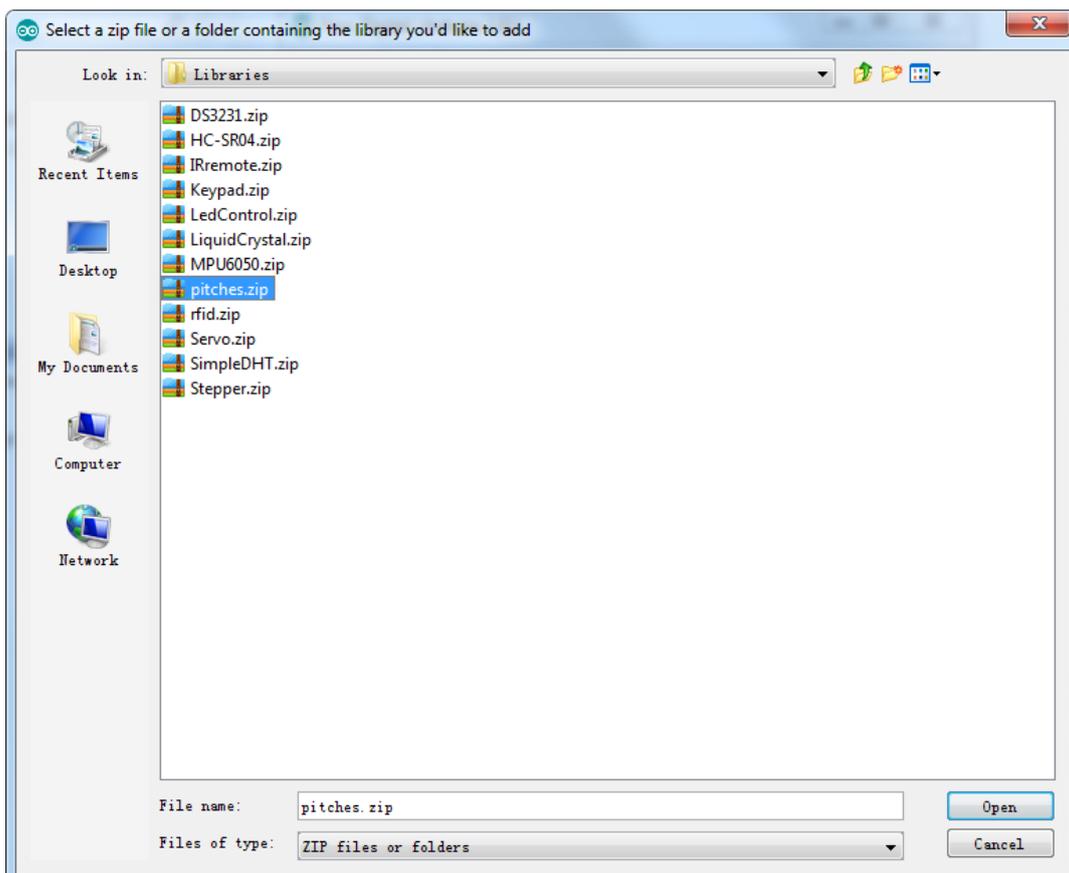
Eine .zip Bibliothek einbinden

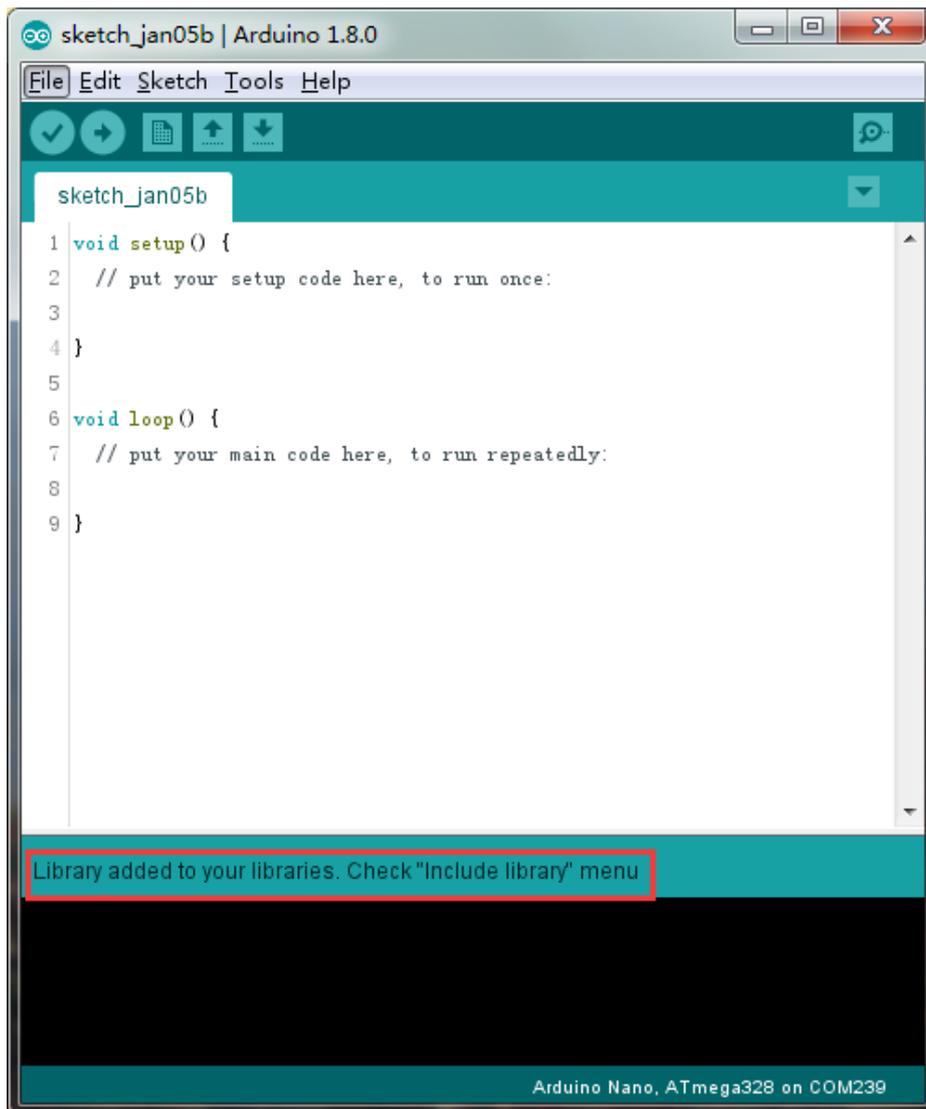
Weitere Bibliotheken werden im Internet häufig als ZIP Dateien zum Download angeboten. Der Name des ZIP Archivs ist meist der Name der Bibliothek. Im Archiv befindet sich eine *.cpp* Datei, eine *.h* Datei und oft eine *keywords.txt* Datei und ein *examples* Ordner mit Code-Beispielen und andere Dateien, die von der Bibliothek benötigt werden. Seit Version 1.0.5 können Drittanbieter-Bibliotheken in der Arduino IDE eingebunden werden. Entpacken Sie die ZIP Dateien der Bibliotheken nicht selber, sondern belassen Sie sie so wie sie sind. Die IDE wird das Archiv automatisch entpacken.

Navigieren Sie in der IDE zu *Sketch > Bibliothek einbinden* und wählen Sie „*ZIP Bibliothek hinzufügen...*“.



Sie werden dazu aufgefordert, die ZIP-Datei auszuwählen, die Sie hinzufügen wollen. Öffnen Sie die heruntergeladene .ZIP-Bibliothek.





Gehen Sie zurück zum Menü *Sketch > Bibliothek einbinden*. Sie sollten Ihre soeben hinzugefügte Bibliothek unten im Drop-Down-Menü wiederfinden. Die Bibliothek kann nun in Ihrem Sketch / Projekt eingebunden werden. Die ZIP-Bibliothek wurde automatisch zu den anderen Bibliotheken in Ihrem Arduino Projektordner hinzugefügt.

Info: Die neue Bibliothek kann direkt in Ihren Sketch eingebunden werden. Die Beispieldateien der neuen Bibliothek sind jedoch erst nach einem Neustart der IDE verfügbar.

Dies sind beiden geläufigsten Wege, um Bibliotheken hinzuzufügen. Bei Mac und Linux Systemen funktioniert es auf die gleiche Art. Die untenstehende manuelle Installation wird nur selten gebraucht und falls Sie diese nicht benötigen, können Sie sie überspringen.

Manuelle Installation

Beenden Sie zuerst die Arduino Software, bevor Sie die Bibliothek manuell installieren. Dann entpacken Sie das ZIP-Archiv der Bibliothek. Wenn Sie zum Beispiel eine Bibliothek namens "ArduinoParty" heruntergeladen haben, entpacken Sie die *ArduinoParty.zip* Datei. Der entpackte Ordner sollte einen Ordner namens *ArduinoParty* enthalten, in welchem sich wiederum Dateien wie *ArduinoParty.cpp* und *ArduinoParty.h* befinden sollten. Wenn die .cpp und .h Dateien nicht in einem separaten Ordner liegen, müssen Sie einen erstellen. In diesem Fall würden Sie einen Ordner namens „ArduinoParty“ erstellen und verschieben alle Dateien, die im Zip-Archiv waren, in diesen Ordner (wie *ArduinoParty.cpp* und *ArduinoParty.h*).

Verschieben Sie den *ArduinoParty* Ordner nun in Ihren Bibliotheksordner. Unter Windows und Mac ist dies standardmäßig unter *Dokumente\Arduino\libraries*. Unter Linux ist es ebenfalls *libraries* Ordner in Ihrem Projektordner.

Ihr Arduino Bibliotheks Ordner sollte nun so aussehen (unter Windows):

Dokumente\Arduino\libraries\ArduinoParty\ArduinoParty.cpp

Dokumente\Arduino\libraries\ArduinoParty\ArduinoParty.h

Dokumente\Arduino\libraries\ArduinoParty\examples

oder so (unter Mac und Linux):

Dokumente/Arduino/libraries/ArduinoParty/ArduinoParty.cpp

Dokumente/Arduino/libraries/ArduinoParty/ArduinoParty.h

Dokumente/Arduino/libraries/ArduinoParty/examples

....

Es können mehr Dateien als nur die .cpp and .h Dateien vorhanden sein, gehen Sie nur sicher, dass sich alle Dateien in Ihrem Ordner befinden. Die Bibliothek wird nicht funktionieren, wenn sich die .cpp und .h Dateien direkt im *libraries* Ordner oder in einem weiteren Unterordner befinden. Zum Beispiel:

Dokumente\Arduino\libraries\ArduinoParty.cpp und

Dokumente\Arduino\libraries\ArduinoParty\ArduinoParty\ArduinoParty.cpp

würden nicht funktionieren.

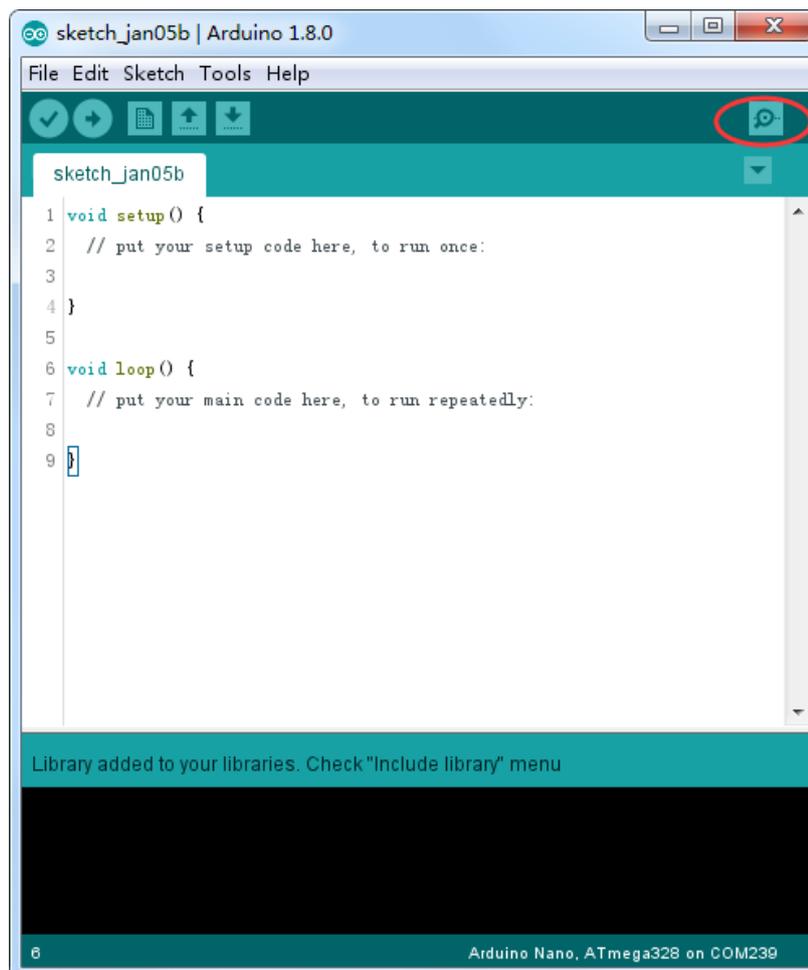
Wenn alles richtig ist, starten Sie die Arduino IDE neu. Stellen Sie sicher, dass die neue Bibliothek unter *Sketch > Bibliothek einbinden* zu finden ist. Dann sind Sie fertig.

Arduino Serieller Monitor (Windows, Mac, Linux)

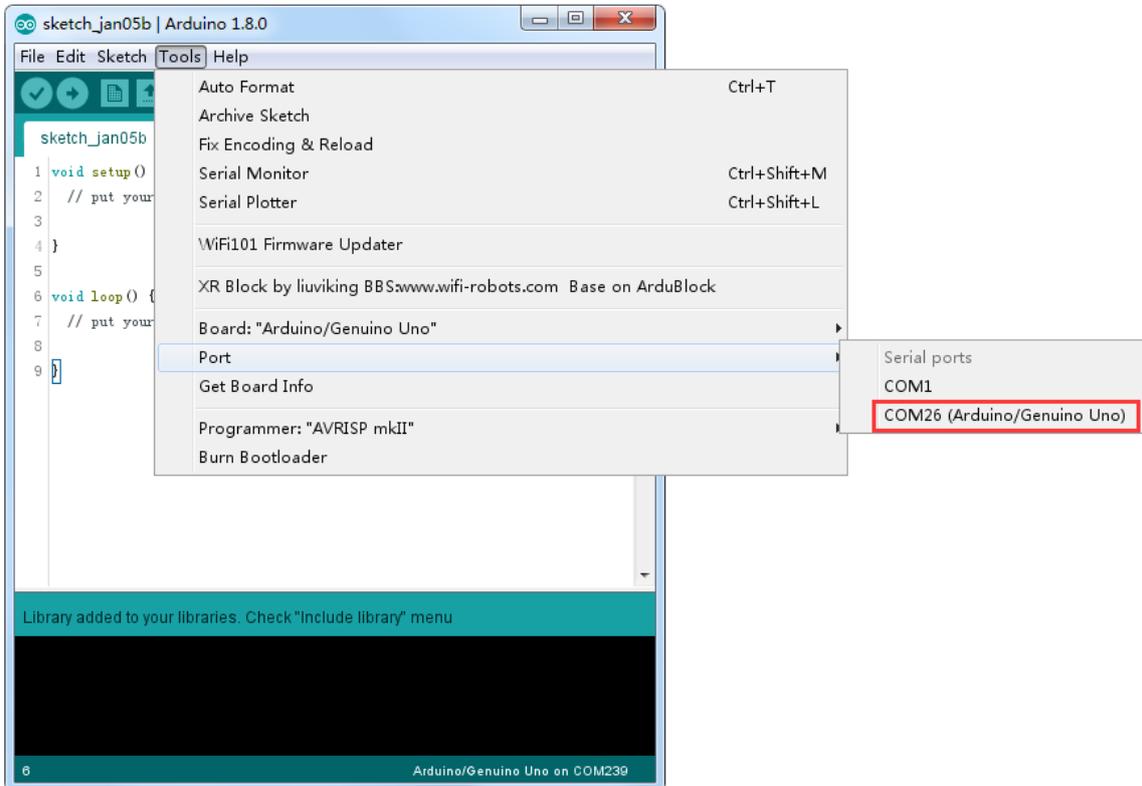
Die Arduino Entwicklungsumgebung (IDE) ist die Software Seite der Arduino Plattform. Und weil das Arbeiten mit einem Terminal ein so bedeutender Teil beim Programmieren mit Arduino und anderen Mikrocontrollern ist, haben sich die Entwickler der Arduino IDE dazu entschieden, ein Terminal (Kommandozeile) zu integrieren. In der Arduino Entwicklungsumgebung wird dies „*Serieller Monitor*“ genannt.

Die Verbindung herstellen

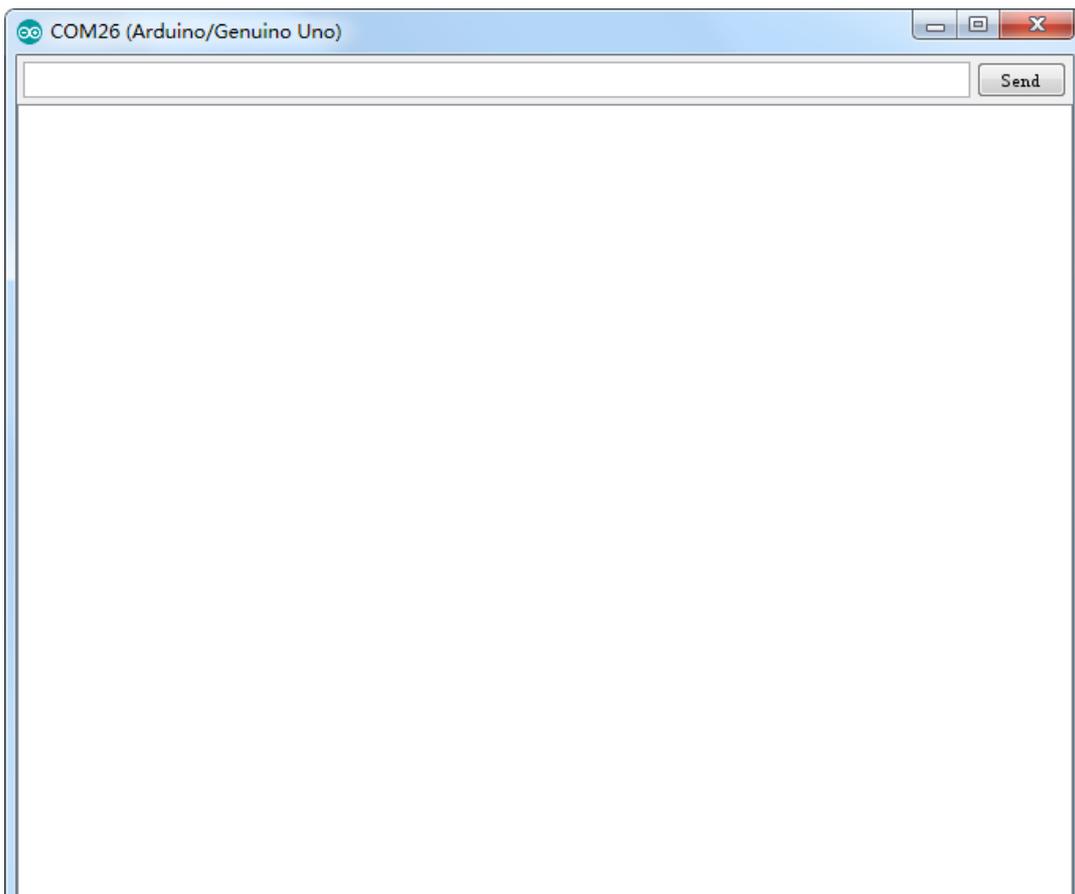
Der Serielle Monitor ist in jeder Version der Arduino IDE vorhanden. Um ihn zu öffnen, müssen Sie lediglich auf das im Bild unten markierte Symbol klicken.



Bevor die Verbindung möglich ist, müssen Sie den richtigen COM-Port auswählen. Gehen Sie zu *Werkzeuge* ->*Port* und wählen Sie den richtigen Port aus. Normalerweise ist nur einer vorhanden. Sonst schauen Sie im Geräte-Manager nach.

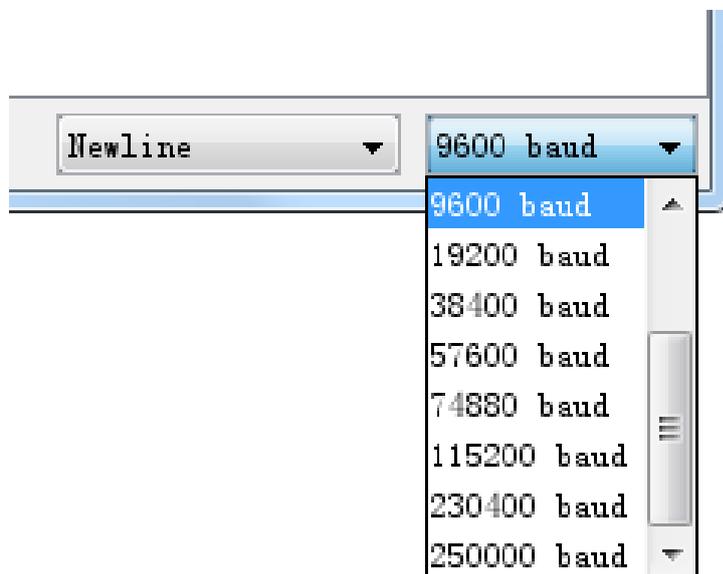


Sobald Sie den Seriellen Monitor dann geöffnet haben, erscheint dies:

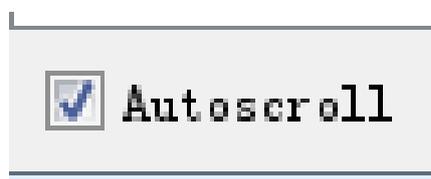


Einstellungen

Die Optionen des Seriellen Monitors sind sehr limitiert, aber reichen aus, um die wichtigsten Optionen der Seriellen Kommunikation festzulegen. Die erste Einstellung, die Sie machen, ist die Festlegung der *Baud-Rate*. Klicken Sie auf das im Bild zu sehende Drop-Down Menü und wählen Sie die richtige Baudrate aus (9600 baud).



Schließlich können Sie durch Anklicken der Checkbox *Autoscroll* einstellen, ob der serielle Monitor automatisch mitscrollen soll.



Vorteile

Der serielle Monitor ist ein schneller und einfacher Weg eine serielle Verbindung zu Ihrem Arduino aufzubauen. Wenn Sie bereits mit der Arduino IDE arbeiten, brauchen Sie kein weiteres Terminal zu installieren, um sich die serielle

Kommunikation Ihres Arduinos anzuschauen.

Nachteile

Die minimalistischen Einstellungen des Seriellen Monitors lassen zu wünschen übrig und für fortgeschrittene Serielle Kommunikation kann dies unzureichend sein.

Lektion 2: Blink

Übersicht

In dieser Lektion lernen Sie, wie Sie Ihr UNO R3 Entwicklungsboard so programmieren, dass die eingebaute LED leuchtet. Außerdem erfahren Sie Schritt für Schritt, wie man ein Programm auf das Arduino Board hochlädt.

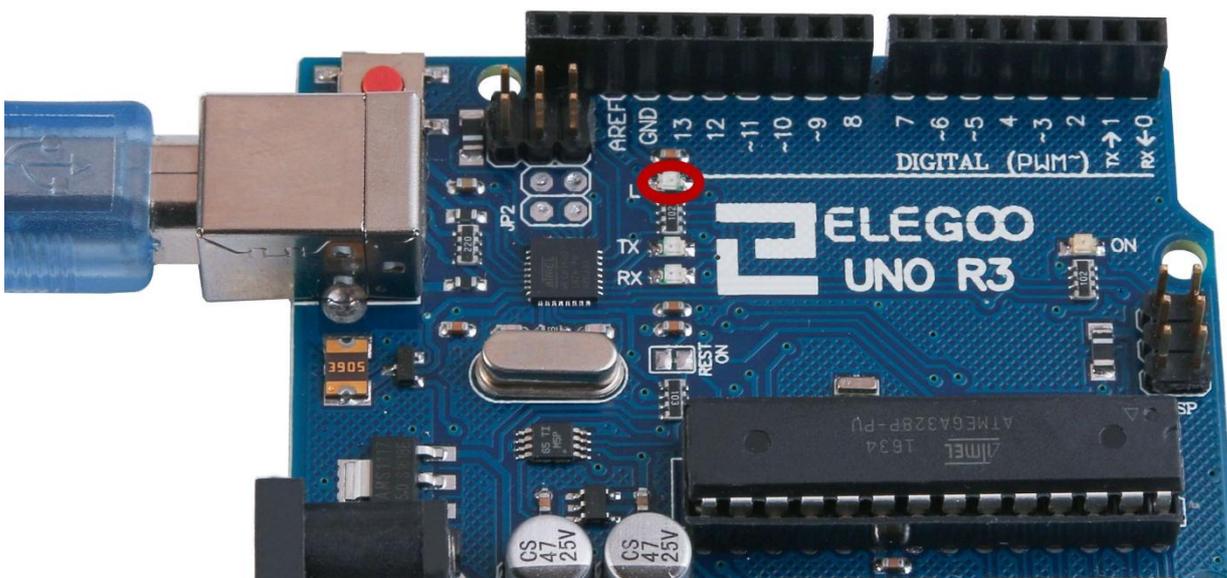
Benötigte Bauteile:

(1) x Elegoo UNO R3

Aufbau

Das UNO R3 Board hat an beiden Seiten Steckplätze, die dazu benutzt werden können, externe elektrische Geräte und Module („Shields“) anzuschließen, um so die technischen Möglichkeiten zu erweitern.

Das Board hat außerdem eine LED, die Sie in Ihren Sketches (Programmen) kontrollieren können. Diese LED ist fest im UNO R3 Board verbaut und wird oft „L“-LED genannt, da sie so auf dem Board gekennzeichnet ist.



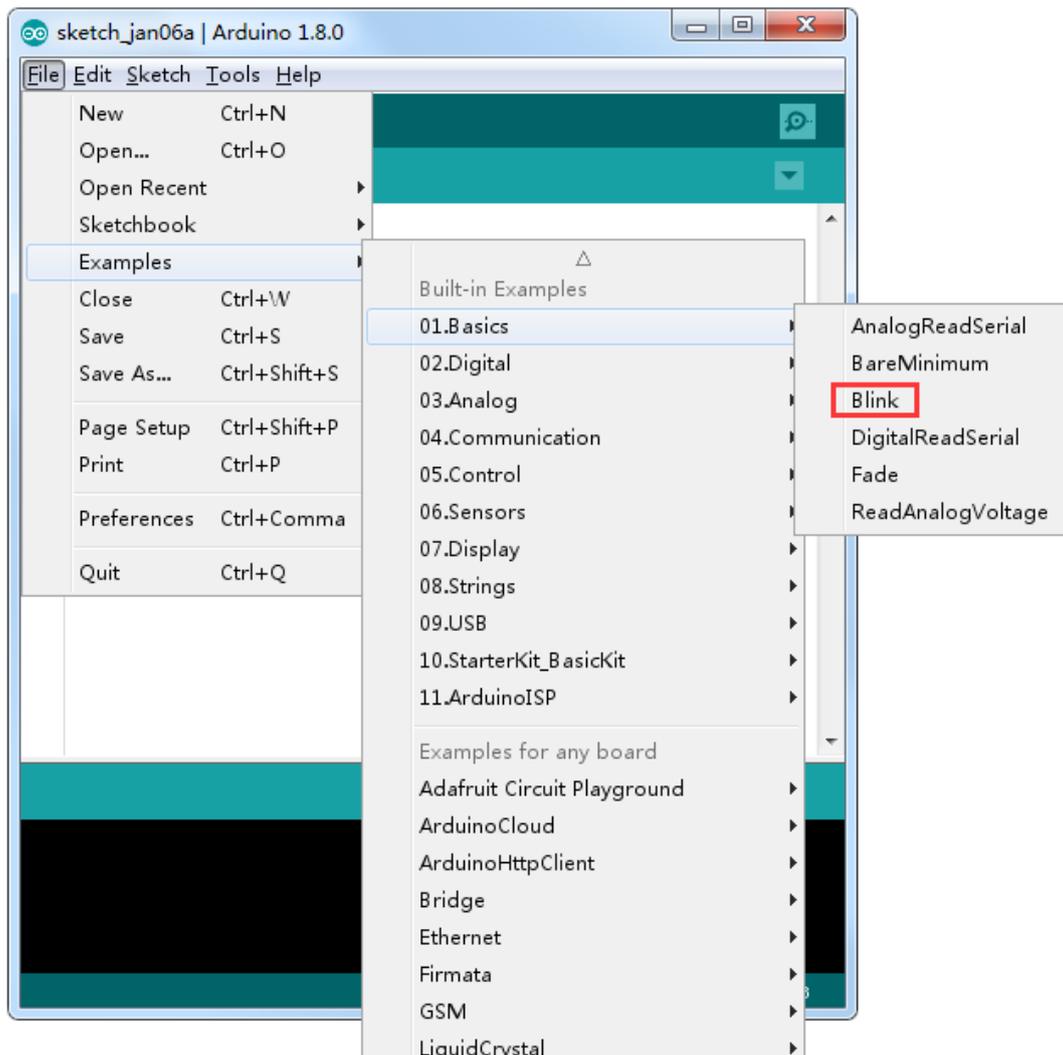
Es kann sein, dass Ihre „L“-LED bereits blinkt, wenn Sie Ihr UNO R3 Board mit einer Stromversorgung verbinden. Das liegt daran, dass die ausgelieferten Boards bereits mit dem „*Blink*“-Sketch vorinstalliert kommen, der die LED blinken lässt.

In dieser Lektion werden wir das UNO R3 Board mit unserem eigens kreierten Blink Sketch programmieren und anschließend die Blinkgeschwindigkeit ändern.

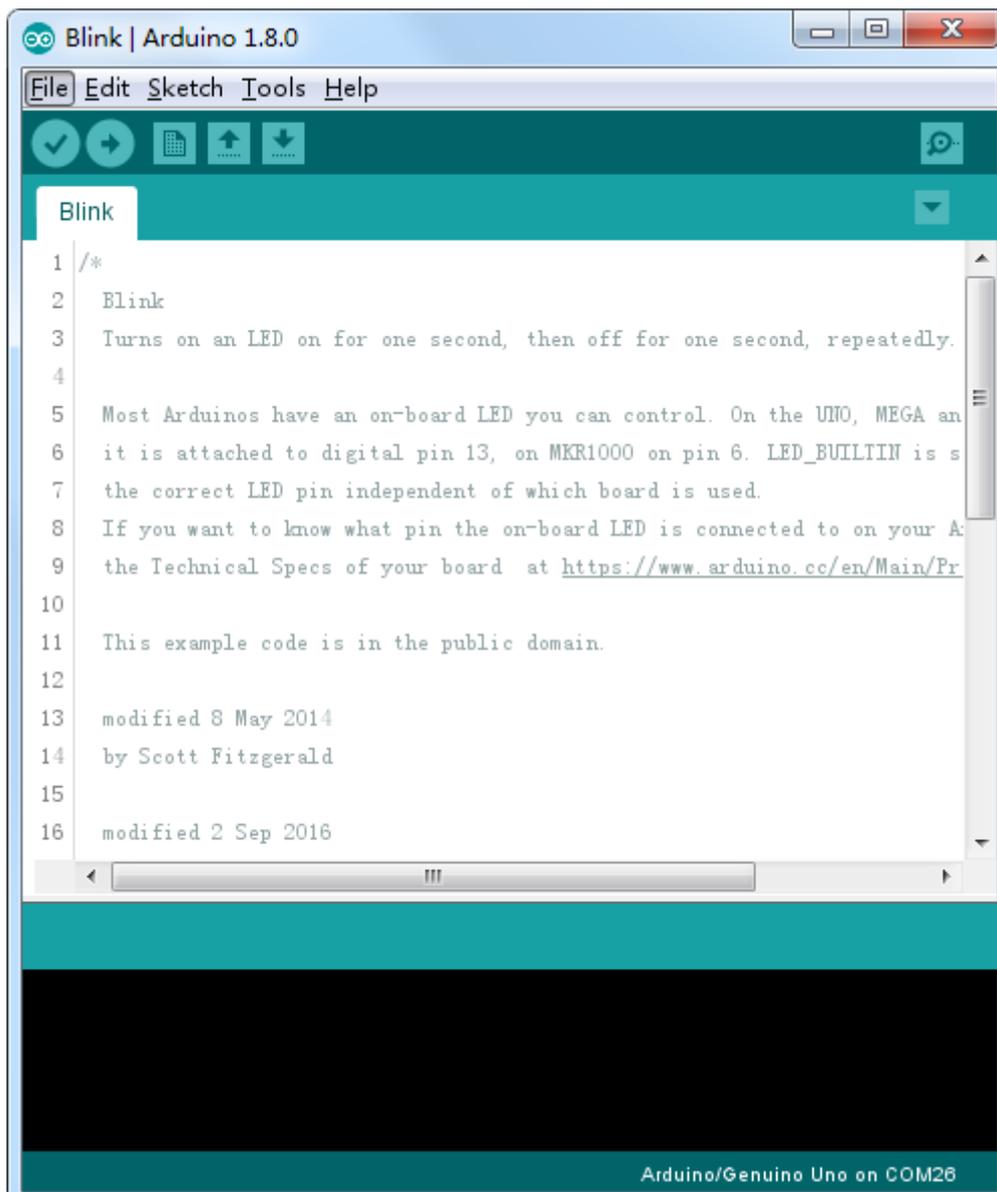
In Lektion 0 haben Sie die Arduino IDE (Entwicklungsumgebung) aufgesetzt und den richtigen Seriellen Port in der IDE eingestellt. Nun werden wir diese Verbindung testen, indem wir das UNO R3 Board erstmals selber programmieren.

Die Arduino IDE kommt mit einer großen Sammlung von Beispiel-Sketches, die Sie einfach öffnen und auf das Board hochladen können. In dieser Sammlung befindet sich auch bereits ein Sketch, der die L-LED zum leuchten bringt.

Laden Sie den Blink-Sketch, den Sie in der IDE unter *Datei > Beispiele > 0.1Basics* finden.



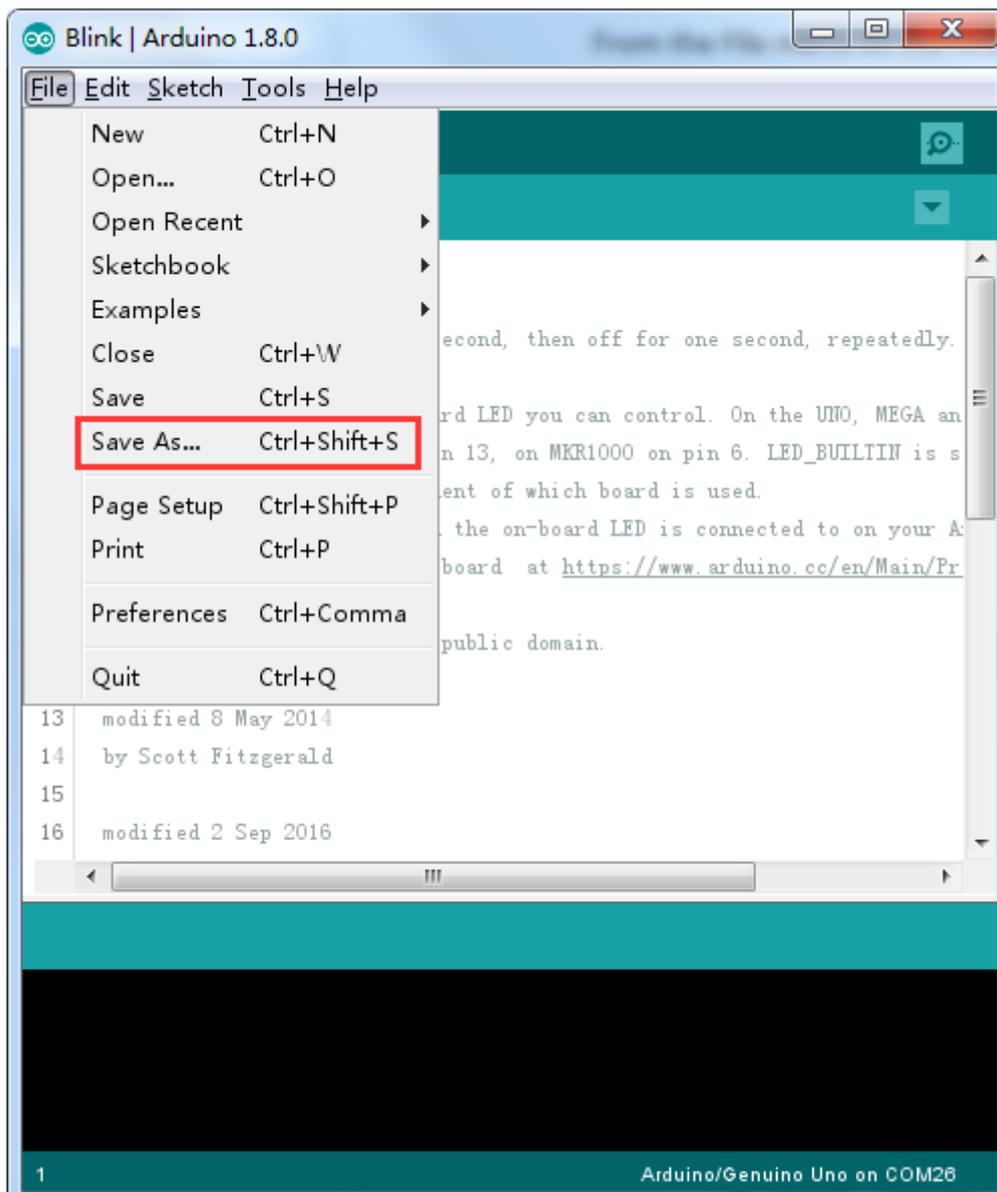
Wenn sich der Blink-Sketch geöffnet hat, sollten Sie das Fenster vergrößern, um sich einen Überblick über den gesamten Programm-Code schaffen zu können.



Die mitinstallierten Beispiel-Sketches sind nicht „schreibbar“. Das heißt, man kann Änderungen nicht abspeichern. Sie können den Sketch abändern und hochladen, aber nicht unter dem gleichen Dateinamen speichern.

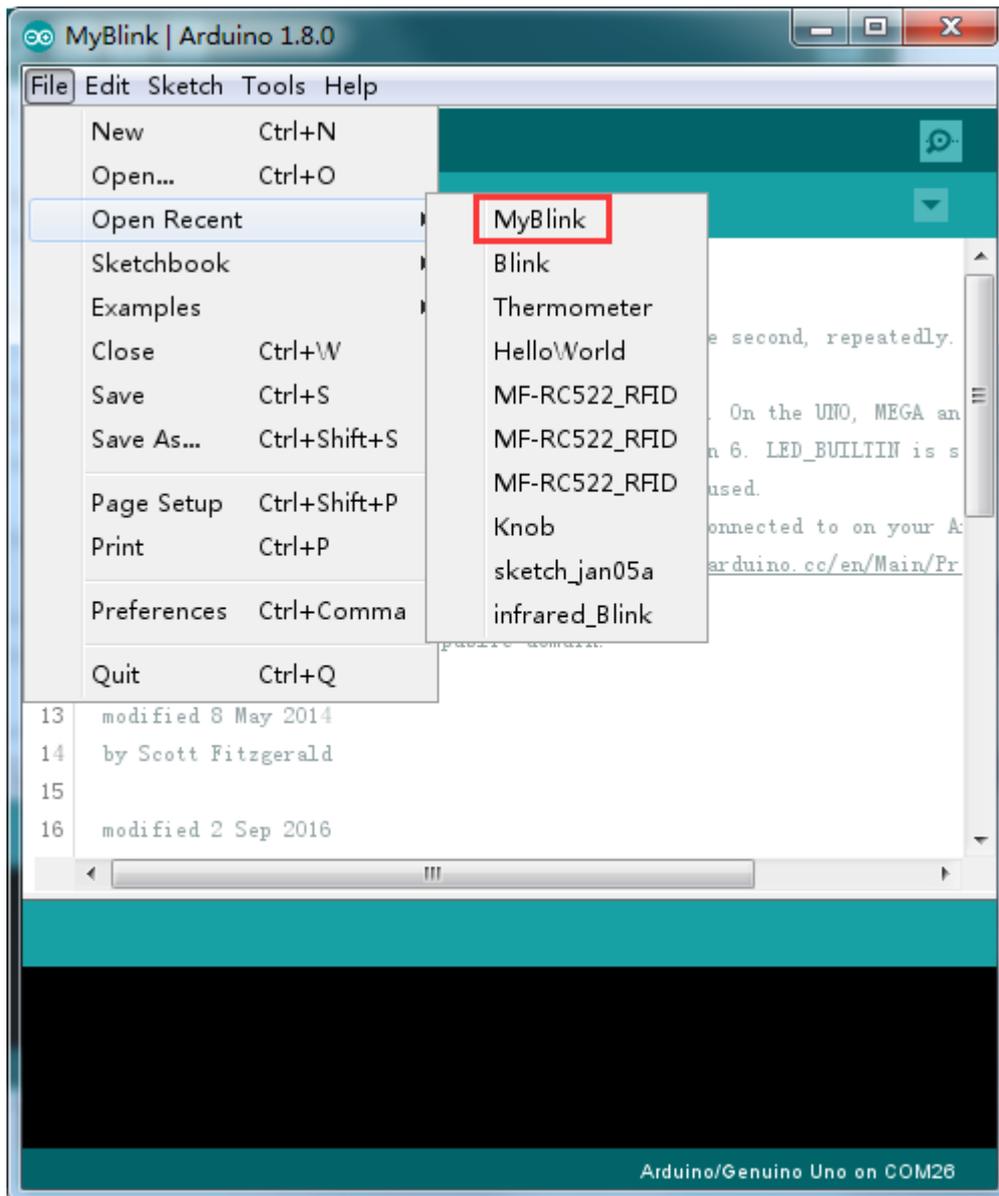
Da wir den Sketch ändern möchten, ist der erste Schritt den Sketch unter neuem Namen abzuspeichern.

Im Arduino IDE Menü wählen Sie *Datei > Speichern unter...* aus und speichern den Sketch dann unter dem Namen „*MyBlink*“.

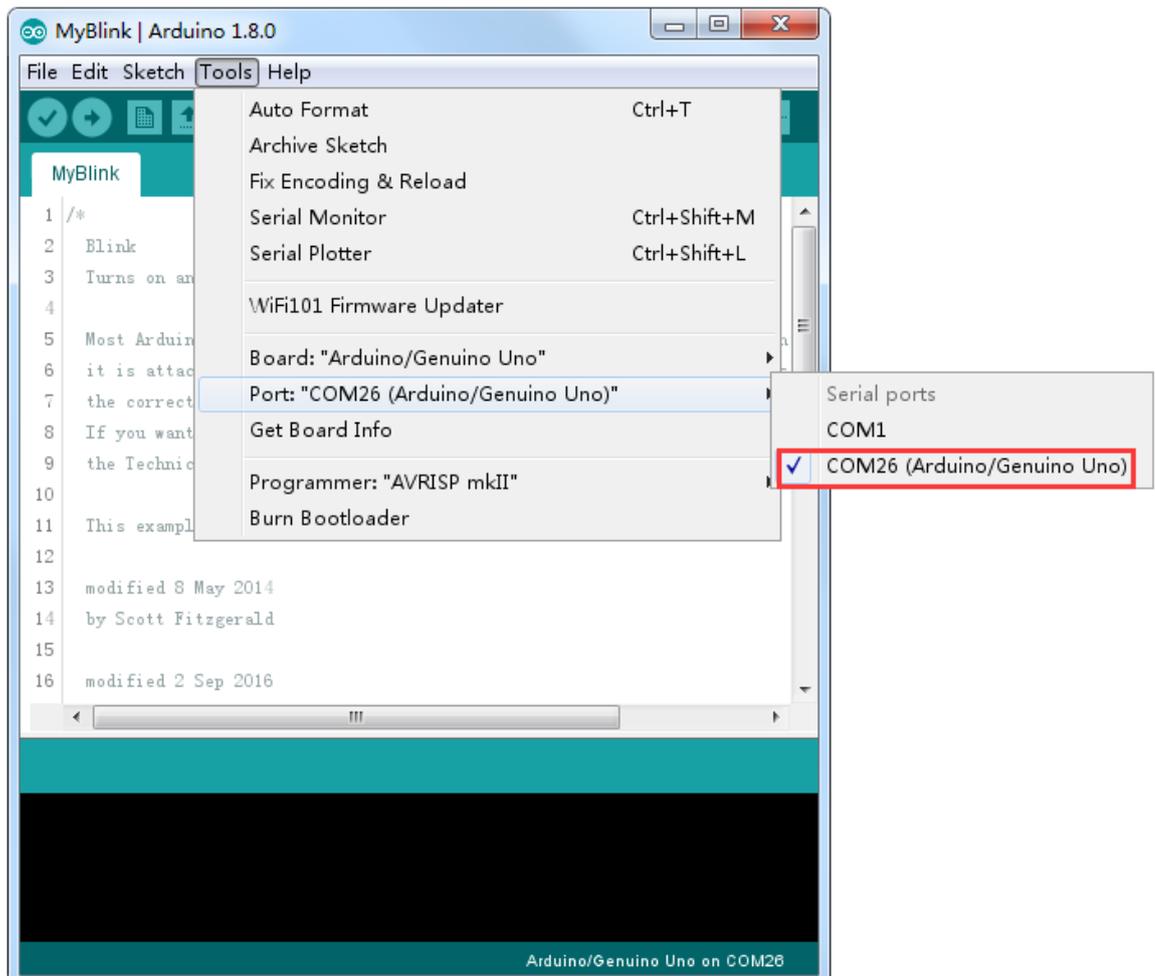
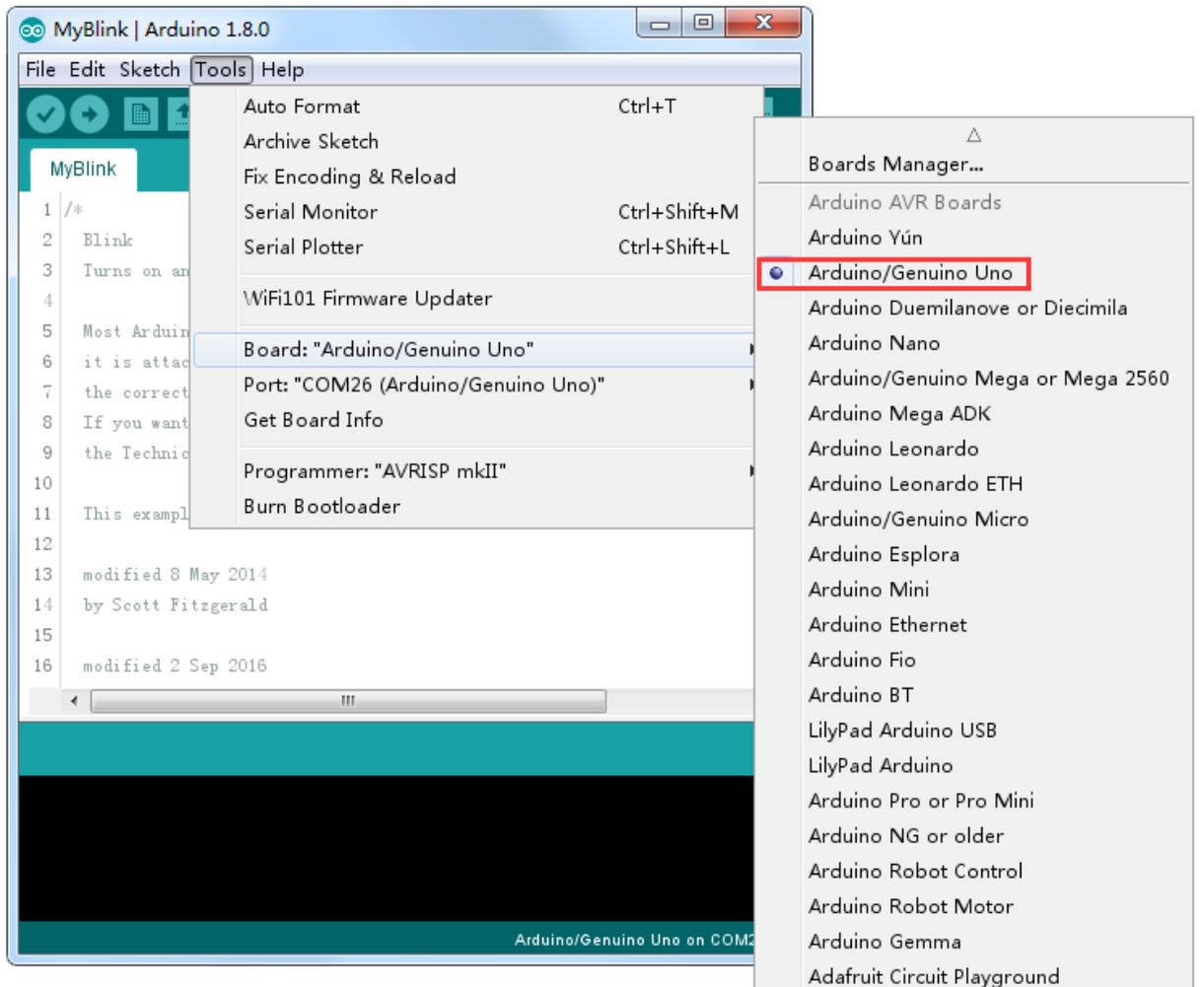




Sie speichern Ihre Kopie des Blink-Sketches in Ihrem Sketchbook (Projektordner) ab. Das bedeutet, dass Sie bei Bedarf den Sketch einfach über *Datei > Sketchbook* im Menü aufrufen können.

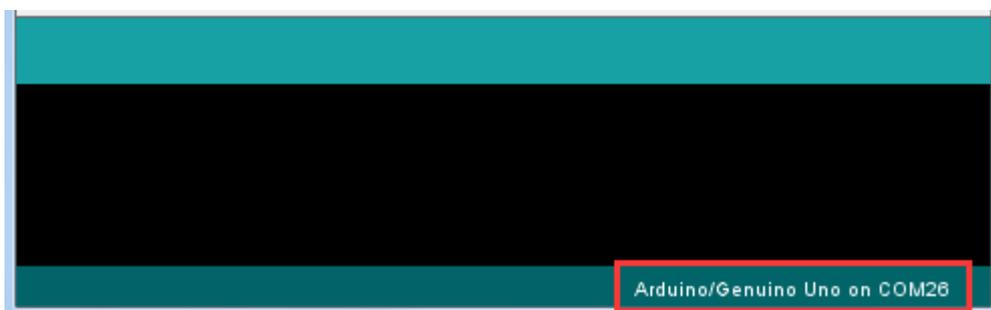


Verbinden Sie Ihr Arduino Board mit einem USB-Kabel mit dem Computer und überprüfen Sie, obder „Board Typ“ und der „Serielle Port“ richtig eingestellt sind,



Achtung: Ihr Board-Typ und der Serielle Port kann sich von den in den Bildern zusehenden Angaben unterscheiden. Wenn Sie das 2560-Board verwenden, dann wählen Sie „Mega 2560“ als Board-Typ, wenn Sie das UNO R3 Board verwenden, wählen Sie „Uno“ als Board-Typ, usw. Der Serielle Port ist bei jedem Gerät unterschiedlich und wird vom Computer zugewiesen. Im Gegensatz zum hier angegebenen Port COM26, könnte es bei Ihnen COM3 oder COM4 sein. Der richtige COM-Port ist durch ein „COMX (arduino XXX)“ gekennzeichnet.

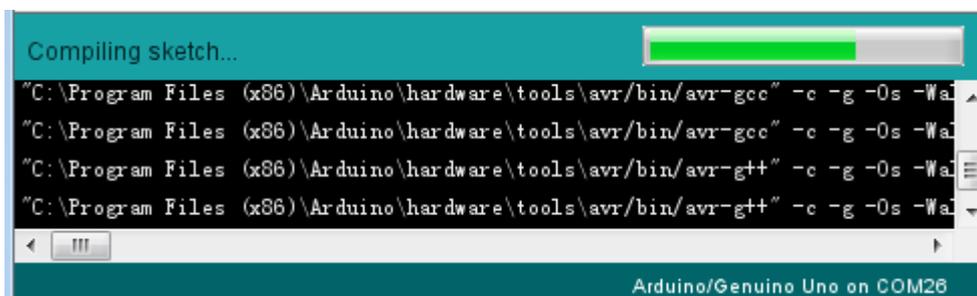
Die Arduino IDE zeigt die aktuellen Einstellungen für das Board im Fenster rechts unten an.



Klicken Sie nun auf die „Hochladen“-Schaltfläche. Dies ist das zweite Symbol von links.



Wenn Sie den Statusbereich der IDE in der unteren Sektion des Fensters beobachten, sehen Sie, wie sich eine Fortschrittsleiste langsam füllt und Meldungen erscheinen. Zuerst erscheint die Meldung „Sketch wird kompiliert...“. Dabei wird der Sketch in ein für das Board passende Format umgewandelt.



Als nächstes ändert sich der Status zu „Hochladen“. An diesem Punkt sollten die LEDs des UNO R3 Boards anfangen zu blinken, da der Sketch auf das Board geladen wird.



Zum Schluss ändert sich der Status zu „Hochladen abgeschlossen“.



Die Meldung darunter teilt uns mit, dass der Sketch 928 Bytes von 32.256 Bytes (der gesamte verfügbare Speicher) belegt. Es ist möglich, dass Sie stattdessen nach dem Schritt „Sketch wird kompiliert...“ eine orange Fehlermeldung erhalten:



Das kann zum einen bedeuten, dass Ihr Board gar nicht mit dem Computer verbunden ist oder aber die Treiber wurden (bei manueller Installation) nicht mitinstalliert. Zuletzt kann auch ein falsch ausgewählter Serieller Port für die Fehlermeldung verantwortlich sein.

Wenn dieses Problem bei Ihnen auftritt, gehen Sie zurück zu Lektion 0 und überprüfen Sie Ihre Arduino IDE Installation.

Hat dagegen alles geklappt, sollte das Board nach Abschluss neustarten und anfangen zu blinken. Öffnen Sie den Sketch.

Beachten Sie, dass ein großer Teil dieses Sketches aus Kommentaren besteht. Dabei handelt es sich um keinen richtigen Code, also Anweisungen, was das Board

zu tun hat, sondern nur um Kommentare, die dem Programmierer erklären, wie der Sketch funktioniert. Die Kommentare dienen Ihnen zur besseren Verständlichkeit.

Alles zwischen den Zeichen `/*` und `*/` am Anfang des Sketches ist ein Block-Kommentar. Dort wird erklärt, wozu der Sketch dient.

Ein-Zeilen-Kommentare starten mit `//` und alles von diesem Zeichen an bis hin zum Ende der Zeile wird als Kommentar interpretiert.

Die erste Zeile des Codes lautet:

```
int led = 13;
```

Wie der Kommentar darüber erklärt, weist dies dem Pin der LED einen Namen zu (*led*). Dieser Pin ist der Pin 13 auf den meisten Arduinos, wie beispielsweise UNO und Leonardo.

Als nächstes haben wir die „*setup*“-Funktion. Der Kommentar erklärt, dass alles was sich in dieser Funktion befindet ausgeführt wird, wenn der *Reset*-Knopf gedrückt wird. Es wird außerdem ausgeführt, wenn das Board aus irgendeinem anderen Grund resettet wird, wie zum Beispiel nach dem Einstecken der Stromversorgung oder nach dem Hochladen eines Sketches.

```
void setup() {  
  // initialize the digital pin as an output.  
  pinMode(led, OUTPUT);  
}
```

Jeder Arduino Sketch muss eine *setup*-Funktion haben. Der Code für diese Funktion muss zwischen den geschwungenen Klammern `{ }` platziert werden.

In diesem Fall gibt es in der *setup*-Funktion nur einen einzigen Befehl, welcher bestimmt, dass das Arduino Board den LED-Pin als einen Ausgang benutzen soll.

Außerdem grundlegend für jeden Sketch ist die „*loop*“-Funktion. Im Gegensatz zur *setup*-Funktion, die nur einmalig nach einem Reset läuft, wiederholt sich die *loop*-Funktion ständig, nachdem Sie alle ihre Befehle ausgeführt hat (*loop* = Schleife).

```
void loop() {  
  digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)  
  delay(1000);                // wait for a second  
  digitalWrite(led, LOW);     // turn the LED off by making the voltage LOW
```

```
    delay(1000);           // wait for a second
}
```

Innerhalb der loop-Funktion wird durch die Befehle der LED-Pin erstmal angeschaltet (HIGH). Darauf folgt eine 1000 Milisekunden (= 1 Sekunde) lange Pause. Dann wird der LED-Pin wieder ausgeschaltet (LOW) und es findet wieder eine 1-sekündige Pause statt.

Jetzt werden Sie die LED dazu bringen schneller zu blinken. Wie Sie bereits geahnt haben könnten, liegt der Schlüssel der Blinkgeschwindigkeit darin, die Zeiten der Pausen anzupassen (die Zahlen innerhalb der Klammern der „*delay*“-Befehle).

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the volt
33   delay(500)                        // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the vo
35   delay(500)                        // wait for a second
36 }
```

Die Zeitspanne der Pausen wird in Milisekunden angegeben (1s = 1000ms). Wenn Sie also die LED doppelt so schnell blinken lassen wollen, müssen Sie die Werte halbieren, also von 1000 auf 500 ms senken. Das sorgt dafür, dass zwischen Ein- und Ausschalten der LED nur noch jeweils eine halbe Sekunde Pause stattfindet.

Laden Sie den Sketch erneut hoch und am Ende sollten Sie die LED schneller blinken sehen.

Sie können die Werte nun beliebig anpassen und den Sketch erneut hochladen und beobachten, wie sich die Blinkgeschwindigkeit jedes mal ändert.

Lektion 3: LEDs

Übersicht

In dieser Lektion werden Sie lernen, wie man die Helligkeit von LEDs mit Hilfe verschiedener Widerstände verändern kann.

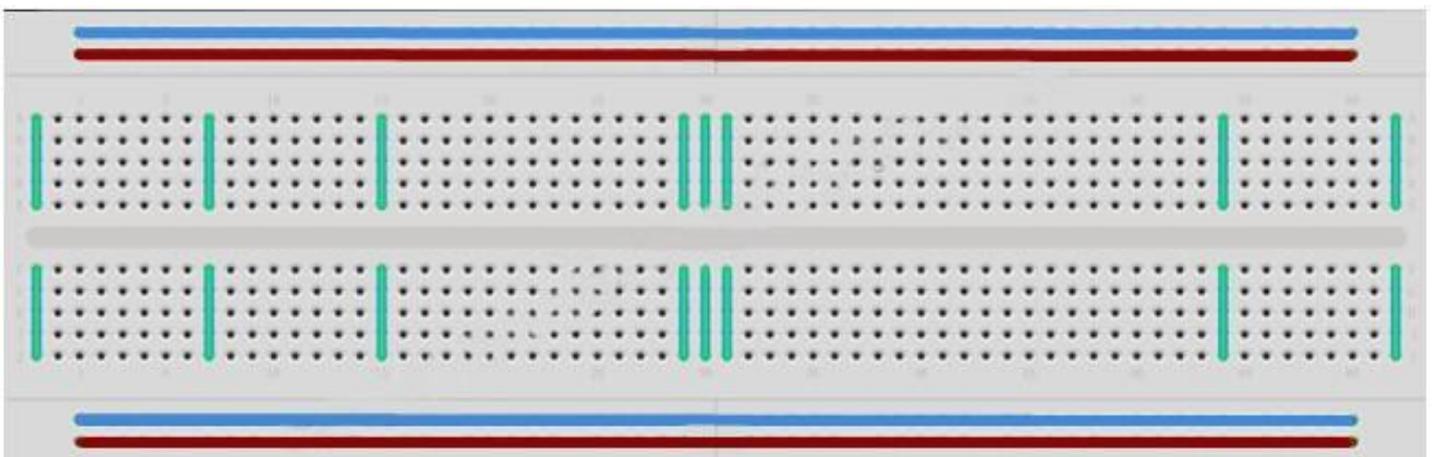
Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 5mm rote LED
- (1) x 220 Ohm Widerstand
- (1) x 1k Ohm Widerstand
- (1) x 10k Ohm Widerstand
- (2) x M-M Kabel (Männlich zu Männlich DuPontJumper Kabel)

Einführung in die Komponenten

BREADBOARD MB-102 :

Ein Breadboard (Steckplatine) ermöglicht es Ihnen, schnell und einfach Schaltungen zu testen, ohne dass gelötet werden muss. Unten finden Sie ein Beispielmodell.



Breadboards gibt es in zahlreichen Größen und Modellen. Die einfachste Art ist ein simples Netz aus Löchern in einem Plastikblock. Innerhalb des Blocks befinden sich Metallstreifen, die die elektrische Verbindung zwischen den Kontaktstellen herstellen. Dabei sind immer die Kontakte innerhalb der selben Reihe (die kurze Seite entlang) miteinander verbunden. Wenn man die Pins von zwei verschiedenen Bauteilen also in die selbe Kontaktreihe steckt, sind sie elektrisch miteinander verbunden. In der Mitte des Blocks befindet sich längs eine Unterbrechung der Kontakte, sodass man die Kontaktreihen links und rechts der Unterbrechung unabhängig voneinander nutzen kann. Manche Breadboards haben an den beiden Außenseiten zwei gesonderte Kontaktreihen, die senkrecht zu den restlichen verlaufen. Diese länglichen Kontaktreihen nennt man „Rails“ (Schienen) und sind dazu da, um die verwendeten Bauteile mit Spannung (meist 5 Volt) zu versorgen.

Während Breadboards sich gut zum Austesten von Prototypen eignen, haben sie einige Limitierungen. Weil die Verbindungen auf Druck bzw eingeklemmten Kabeln basieren und nur temporär sind, sind sie nicht so zuverlässig und haltbar wie gelötete Verbindungen. Wenn bei Ihnen unregelmäßig Fehler in einem Schaltkreis auftreten, könnte es an einem Wackelkontakt auf dem Breadboard liegen.

LEDs:

LEDs (Licht Emittierende Dioden) lassen sich gut als Indikatoren benutzen. Sie verbrauchen sehr wenig Strom und halten so gut wie ewig.

In dieser Lektion werden Sie die wahrscheinlich häufigst vorkommende aller LEDs benutzen: eine 5mm große rote LED. 5mm meinen dabei den Durchmesser der LED. Andere häufige Größen sind 3mm und 10mm.

Man kann eine LED nicht irgendwie an eine Batterie oder Spannungsquelle anschließen. LEDs haben einen positiven und einen negativen Anschluss und leuchten nur auf, wenn sie richtig verbunden sind. Außerdem müssen LEDs mit einem Widerstand benutzt werden, um den fließenden Strom zu begrenzen.



Ohne vorgeschalteten Widerstand wird die LED zerstört, da dadurch zu viel Strom durch sie fließt. Zu hoher Strom erhitzt die LED und lässt sie durchbrennen.

Es gibt zwei Wege zu erkennen, welches der positive und welches der negative Anschluss ist.

Das erste ist die Länge: Der positive Anschluss ist immer länger als der negative.

Außerdem ist an der Stelle, wo der negative Anschluss in das Gehäuse der LED führt, ein abgeflachter Kontakt.

Wenn Sie also eine LED haben, dessen abgeflachter Kontakt gegenüber dem langen Anschluss liegt, ist der lange Anschluss der positive.

Widerstände:

Wie der Name vermuten lässt, setzen Widerstände dem fließenden Strom einen Widerstand entgegen. Je höher der Wert eines Widerstandes, desto mehr Strom wird „aufgehalten“ und desto weniger elektrischer Strom fließt durch die Leitung. Wir benutzen hier einen Widerstand, um zu beeinflussen, wie viel Strom durch die LED fließen wird und daher auch wie hell sie leuchten wird.

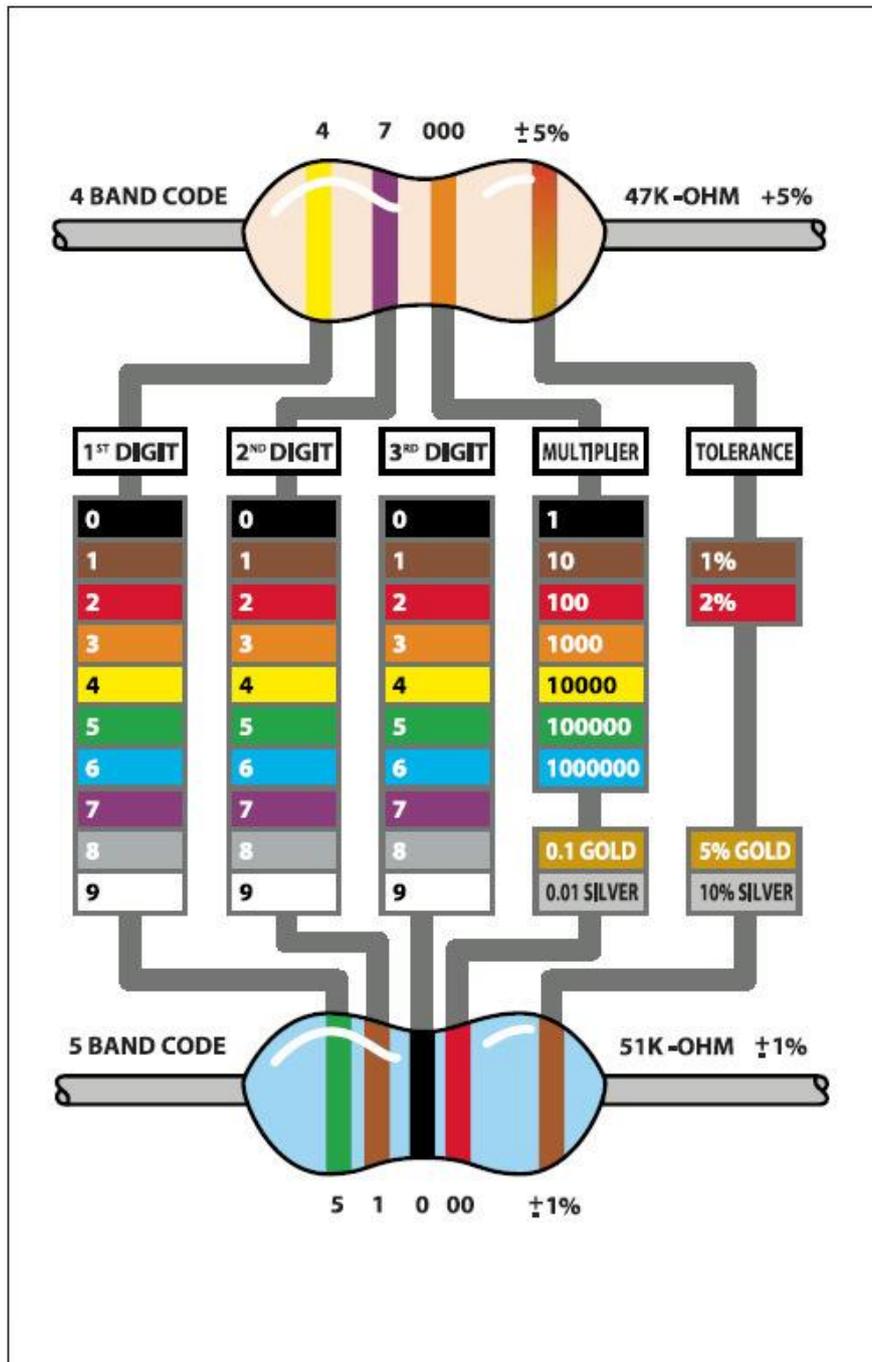


Aber zuerst mehr über Widerstände:

Die Einheit von Widerständen wird „*Ohm*“ genannt, was mit dem griechischen Buchstaben Ω Omega abgekürzt wird. Da 1 Ohm einem sehr geringen Widerstand entspricht, gibt es auch Widerstände in Größen von $k\Omega$ (1.000 Ω) und $M\Omega$ (1.000.000 Ω). Diese Größen werden Kilo-Ohm und Mega-Ohm genannt.

In dieser Lektion benutzen wir drei verschiedene Widerstände: 220 Ω , 1k Ω und 10k Ω . Diese Widerstände sehen, bis auf den aufgemalten Farbcode, alle ziemlich gleich aus. Die bunten Streifen auf dem Widerstand repräsentieren den Wert des Widerstands und können in diesen mit Hilfe einer Tabelle umgerechnet werden.

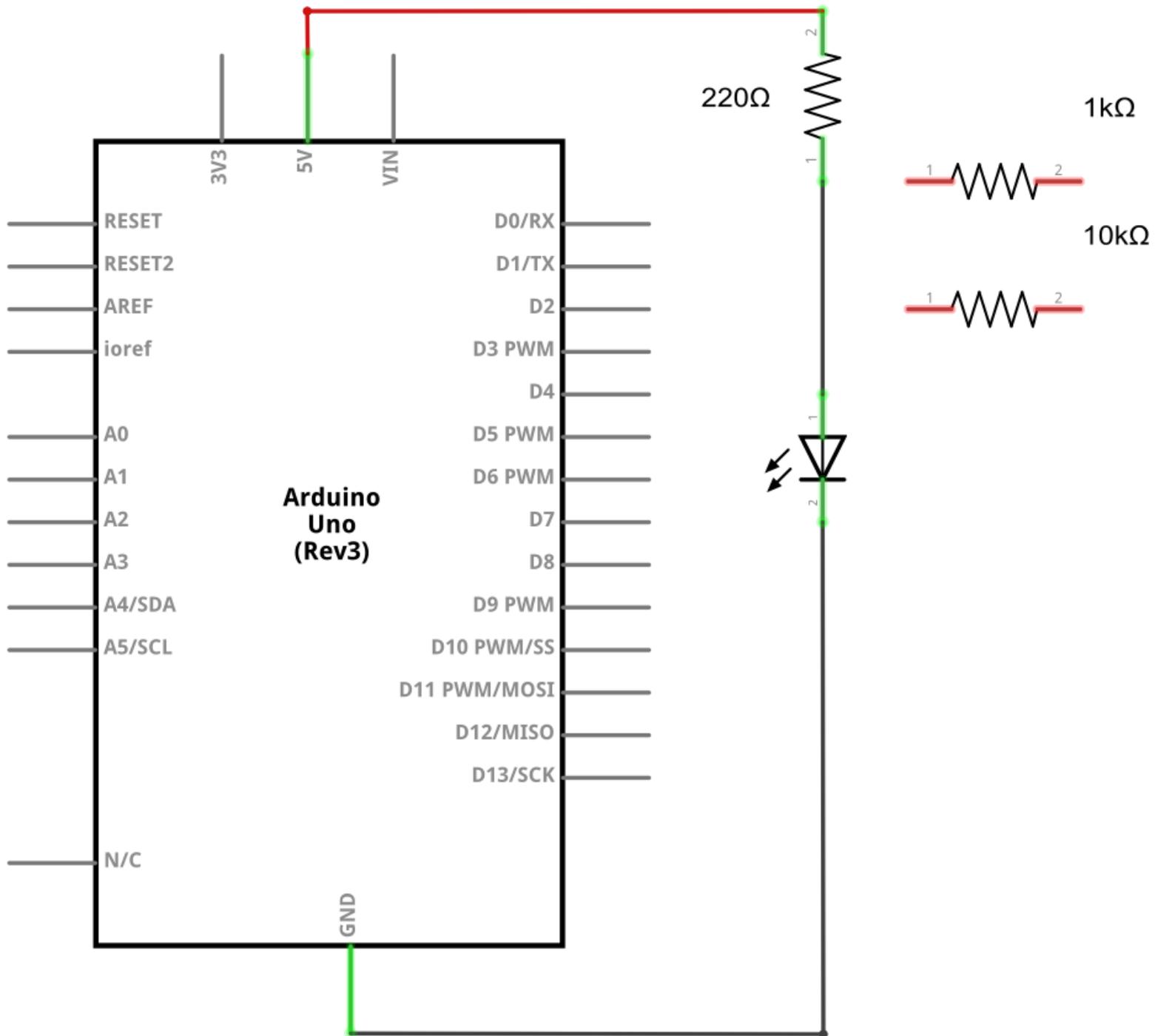
Der Farbcode unseres Widerstandes hat drei farbige Streifen und einen goldenen am Ende.



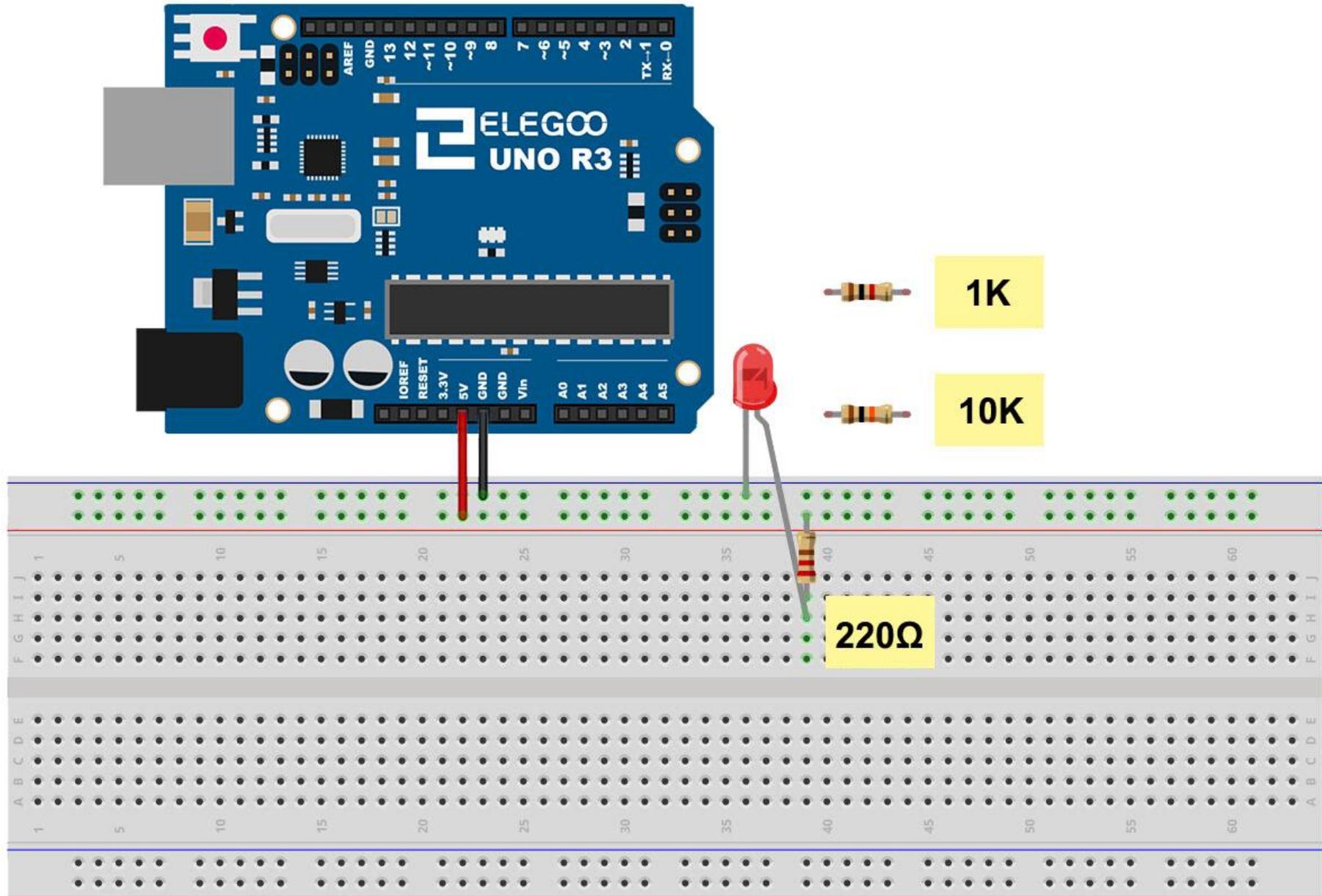
Im Gegensatz zu LEDs haben Widerstände keine positiven und negativen Enden und können daher egal in welche Richtung verbaut werden.

Wenn Ihnen die Methode mit der Tabelle zu kompliziert erscheint, können Sie den Farbcode auf unseren Widerständen direkt ablesen, um seinen Wert zu bestimmen. Alternativ können Sie einen Widerstand auch mit einem Multimeter durchmessen und so den Wert bestimmen.

Verbindungsschema



Schaltplan

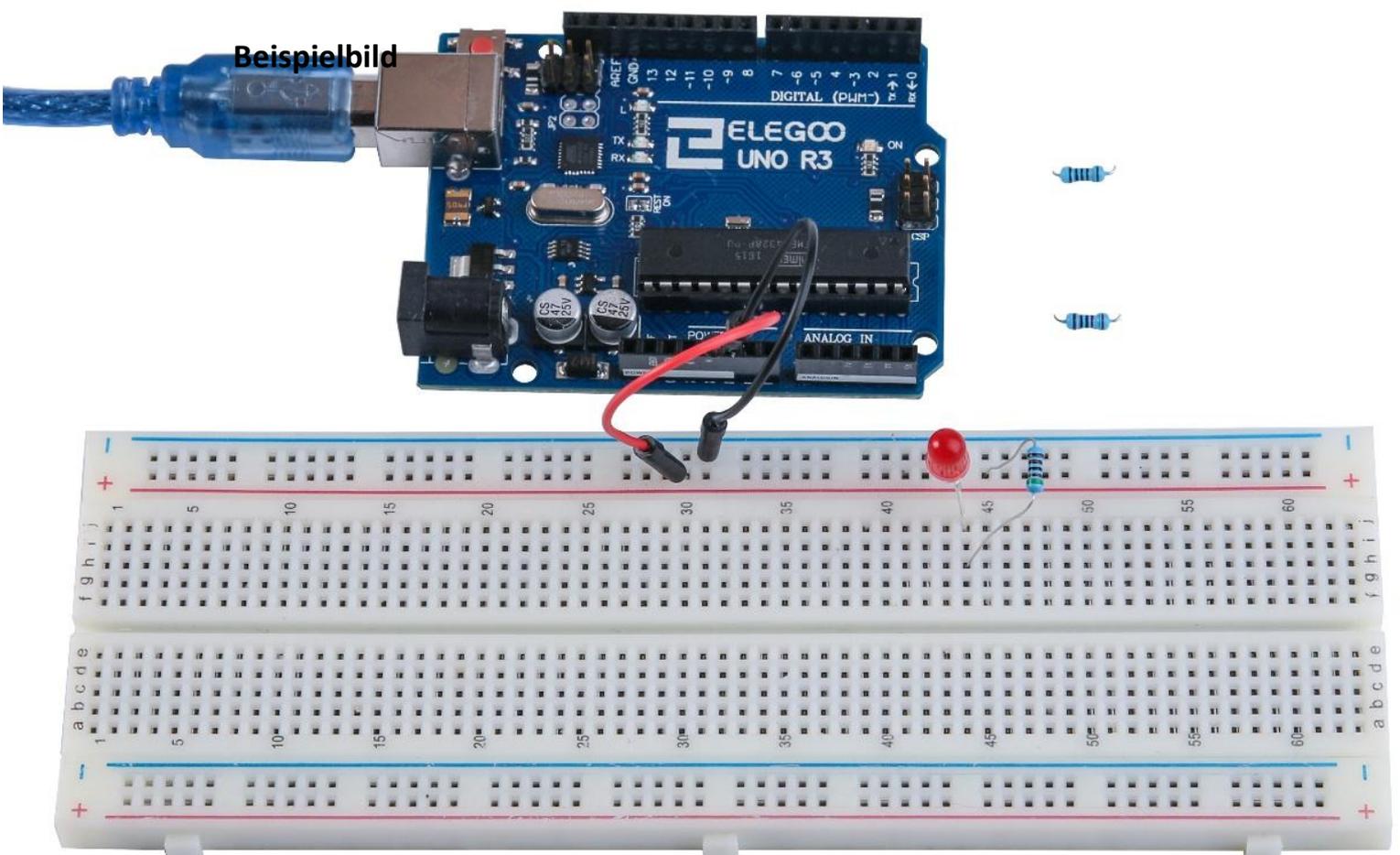


Das UNO Board ist eine praktische Quelle, um 5 Volt Spannung zu erhalten, die wir dafür benutzen werden, um die LED und den Widerstand mit Strom zu versorgen. Sie müssen mit Ihrem Board nichts weiter tun, als es mit einem USB-Kabel zu verbinden.

Mit dem vorgeschalteten 220Ω Widerstand sollte die LED recht hell leuchten. Wenn Sie den 220Ω Widerstand durch den $1k\Omega$ Widerstand ersetzen, sollte die LED ein bisschen dunkler erscheinen. Schließlich, mit dem vorgeschalteten $10k\Omega$ Widerstand, wirkt die LED fast komplett dunkel. Stecken Sie zeitweise das rote Jumper Kabel aus dem Breadboard und wieder ein, um den Unterschied zu einer stromlosen LED zu sehen.

Im Moment haben Sie die positive 5V Spannung mit einem Ende des Widerstandes verbunden, dessen anderes Ende mit der positiven Seite der LED verbunden ist. Die negative Seite der LED geht schließlich an den negativen Anschluss der Stromquelle (auch *GND* = Ground genannt). Wie auch immer, wir können den Widerstand auch so verschieben, dass er nach der LED kommt und es wird trotzdem funktionieren.

Sie können nun den 220Ω Widerstand wiedereinsetzen. Es ist egal an welcher Stelle der LED wir den Widerstand setzen, solange er mit ihr in Reihe geschaltet ist.



Lektion 4: RGB LEDs

Übersicht

RGB LEDs sind ein einfacher Weg, um Farbe in Ihre Projekte zu bringen. Da sich in einer RGB-LED nichts anderes als drei reguläre LEDs in einem befinden, ist die Handhabung dieser ähnlich.

Es gibt sie in zwei Versionen: Mit gemeinsamer Anode und mit gemeinsamer Kathode. Bei gemeinsamer Anode liegt an dem gemeinsam genutzten Pin 5V an, bei gemeinsamer Kathode wird der gemeinsame Pin mit GND (Ground bzw. Erdung) verbunden.

Wie bei jeder LED müssen einige Widerstände vorgeschaltet werden (3 insgesamt), damit der fließende Strom begrenzt wird.

In unserem Sketch lassen wir die LED zuerst in rot leuchten, lassen sie dann nach grün übergehen, danach nach blau und schließlich zurück zur roten Farbe. Dabei werden wir durch den Übergang zwischen den Farben durch einen Großteil aller möglichen Farben schalten.

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x 830 Punkte Breadboard

(4) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

(1) x RGB LED

(3) x 220 Ohm Widerstände

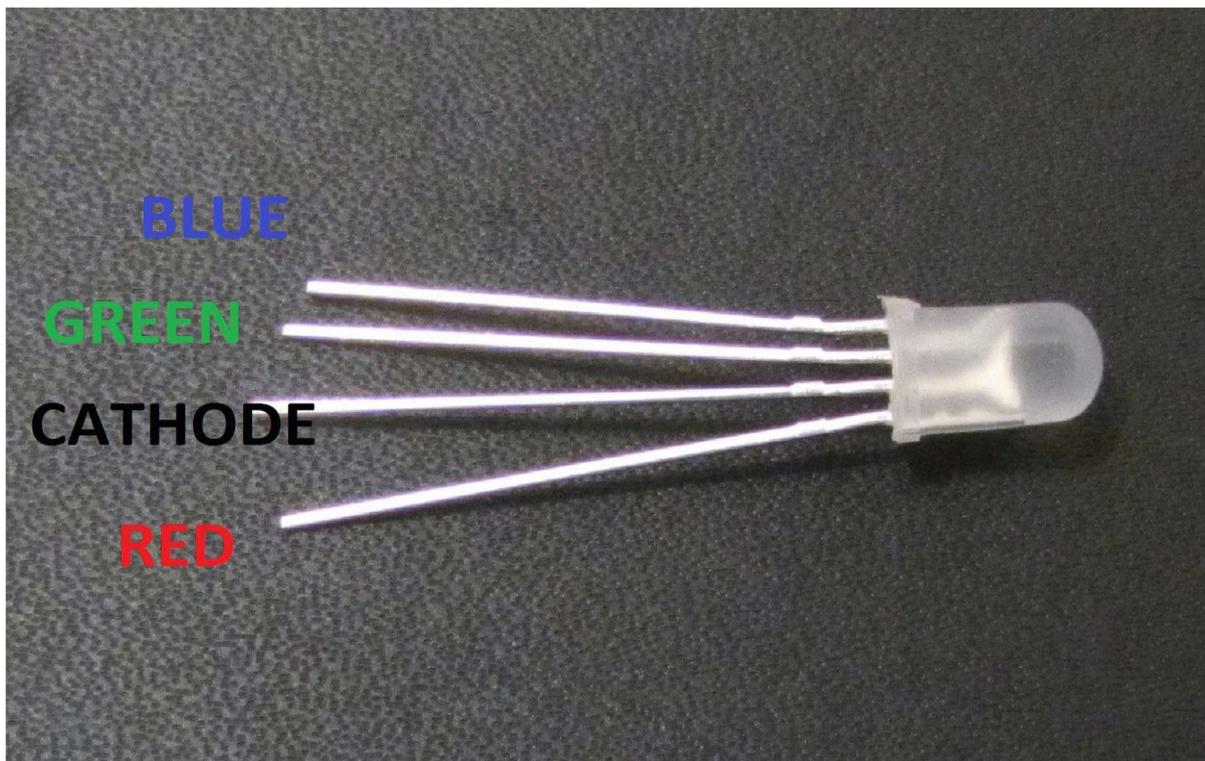
Einführung in die Komponenten

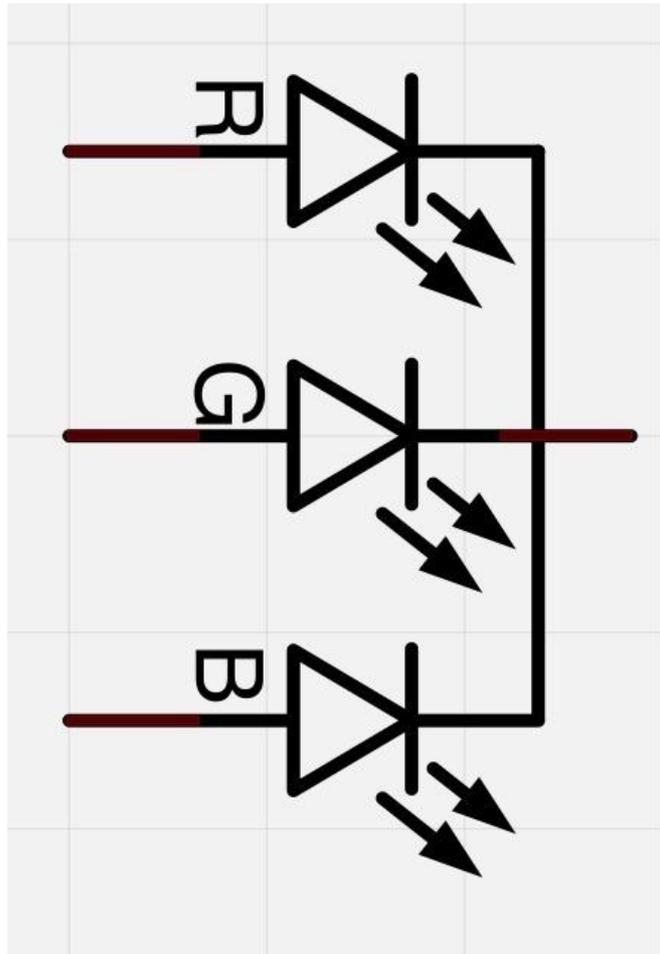
RGB:

Auf den ersten Blick sehen RGB (Rot, Grün, Blau) LEDs wie normale LEDs aus. Innerhalb der RGB-LED befinden sich jedoch drei eigentliche LEDs: eine rote, eine grüne und eine blaue. Durch Anpassen der Helligkeit jeder der drei Farben und zusammenmischen dieser, lassen sich so gut wie alle Farben produzieren.

Die Farben werden dabei ähnlich wie Farbe auf einer Farbpalette gemischt – durch Anpassen der Menge der Grundfarben bzw der drei LEDs. Der schwerste Weg dies zu erreichen wäre durch das Benutzen verschiedener Widerstände oder regelbarer Widerstände, wie wir es bereits in Lektion 2 getan haben, aber das bringt einen großen Aufwand mit sich. Glücklicherweise hat das UNO R3 Board eine *analogWrite*-Funktion, die einem erlaubt die Spannung an den analogen Pins des Boards (mit *a~* markiert) anzupassen.

Die RGB-LED hat vier Kontakte. Es gibt drei Kontakte, die jeweils zum positiven Ende der drei Farb-LEDs gehen und ein Kontakt, der gemeinsam von allen Farben als negativer Anschluss genutzt wird (gemeinsame Kathode).





Hier auf den Bildern sehen Sie eine 4-Elektroden-LED. Die positiven Anschlüsse für Grün, Blau und Rot werden *Anoden* genannt. An diese muss immer der +-Pol angeschlossen werden. Die Kathode wird dagegen immer mit GND (= Ground = Erdung = Minus-Pol) verbunden. Wenn Sie die LED andersherum anschließen, wird sie nicht aufleuchten.

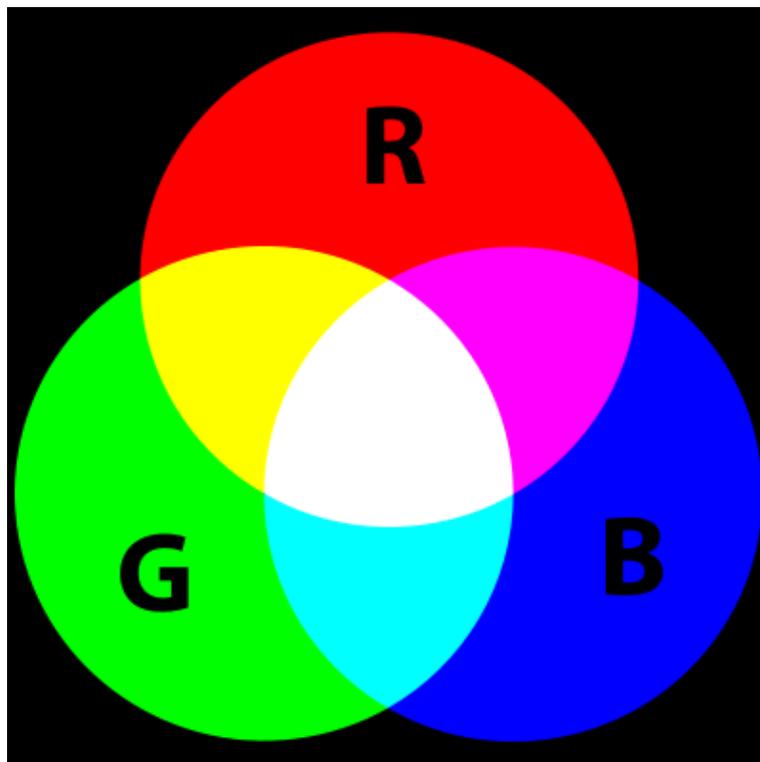
Normalerweise ist der gemeinsame negative Anschluss (gemeinsame Kathode) der zweite Pin von der flachen Seite aus. Es ist außerdem der größte aller vier Pins und wird mit GND verbunden. An jedem der drei positiven Anschlüsse muss ein 220 Ω Widerstand vorgeschaltet werden, um den Strom der LED zu begrenzen.

Die anderen Enden der Widerstände müssen mit dem UNO Board verbunden werden, sodass die LEDs mit dem Board verbunden sind.

Farbe:

Der Grund, warum man durch Anpassen der Helligkeit von rot, grün und blau jede beliebige Farbe mischen kann, ist, dass das menschliche Auge drei verschiedene Lichtrezeptoren (rot, grün und blau) hat. Das Gehirn verarbeitet die Menge von rot, grün und blau und mischt sie zu einer Farbe aus dem Farbspektrum zusammen.

Wir nutzen diesen Trick also aus, um künstlich verschiedene Farben durch Anpassen der rot- grün- und blau-Werte zu erzeugen. Das RGB-Farbmodell wird beispielsweise außerdem in Fernsehern benutzt, bei denen jeder Pixel auf dem LCD-Panel aus drei Farben (rot, grün und blau) besteht.



Wenn wir die Helligkeit aller drei LEDs auf den gleichen Wert einstellen, wird am Ende weiß herauskommen. Wenn wir dann die Helligkeit der blauen LED auf 0 schalten, wird ein gelbes Licht leuchten.

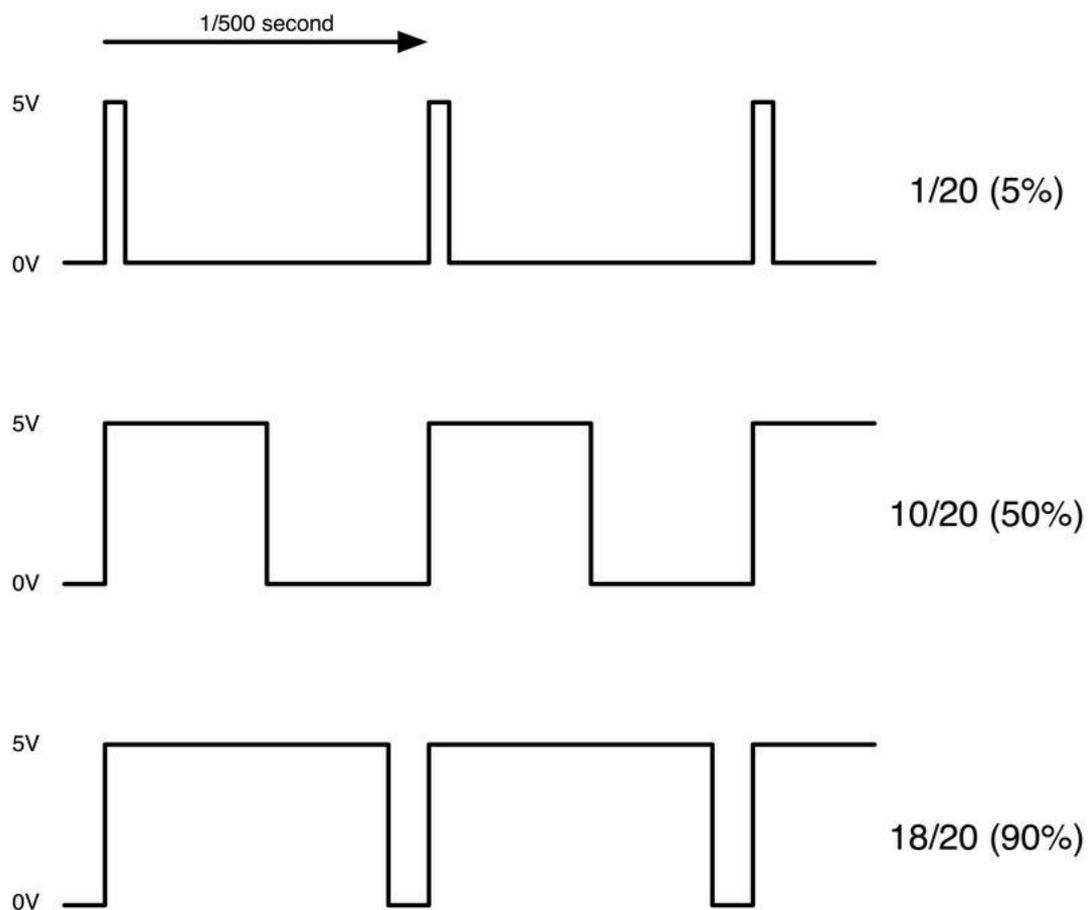
Wir können die Helligkeit der rot, grünen und blauen Teile der LED separat einstellen, was uns ermöglicht jede Farbe, die wir möchten, zusammenzumischen.

Bei Schwarz handelt es sich dagegen um keine eigentliche Farbe, da schwarz im Prinzip nur das Nichtvorhandensein von jeglichem Licht darstellt. Daher können wir mit unserer RGB-LED schwarz nur darstellen, indem wir alle drei Farben ausschalten bzw auf 0 setzen.

PWM-Technik

Die Pulsweitenmodulation (PWM) ist eine Technik, um die elektrische Leistung zu kontrollieren bzw zu begrenzen. Wir benutzen diese Technik auch, um die Helligkeit der einzelnen LEDs zu steuern.

Das Diagramm unten zeigt ein Beispiel eines PWM-Signals aus einem PWM Pin des UNO Boards.

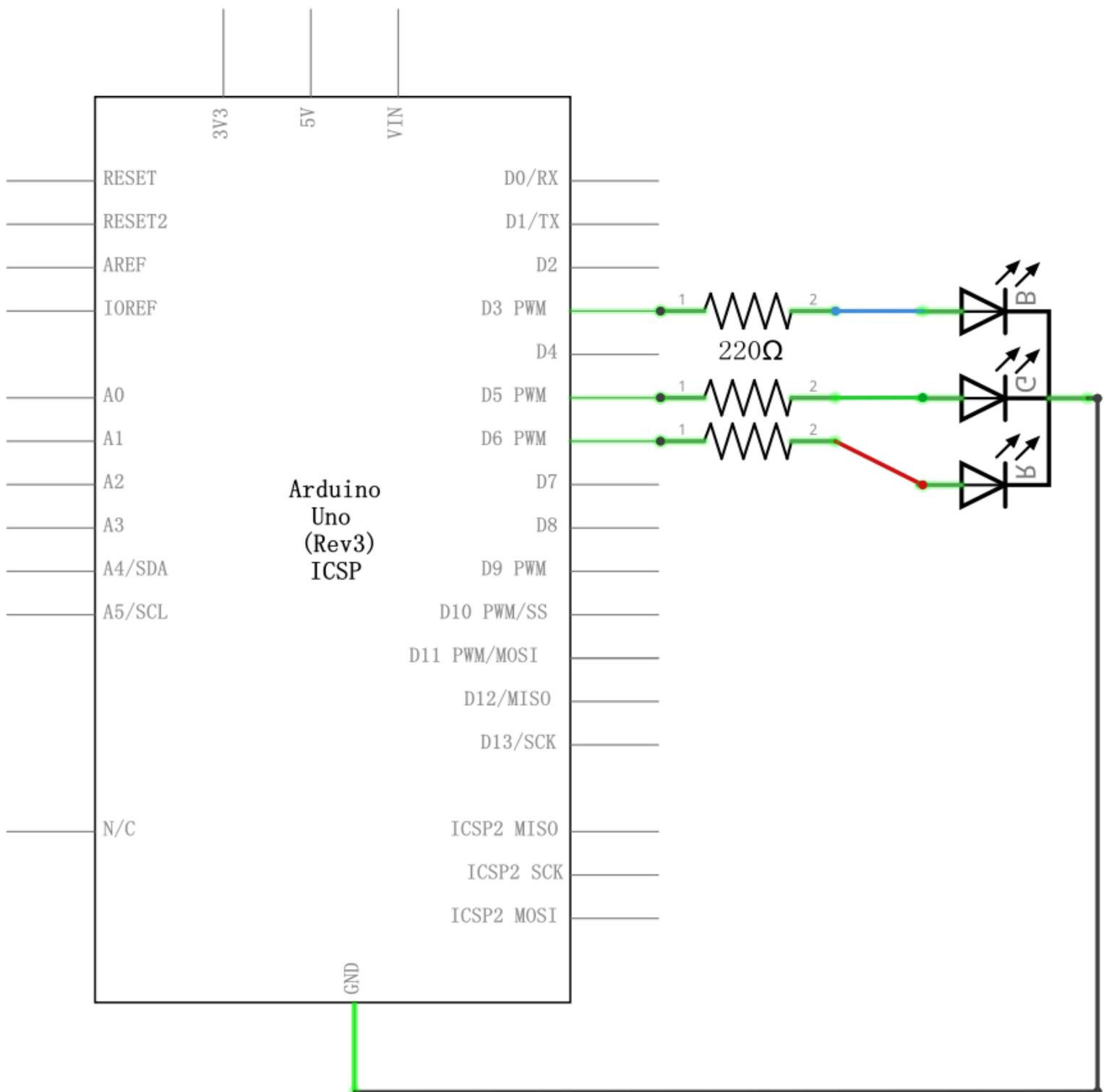


Ungefähr jedes 1/500 einer Sekunde gibt der PWM Ausgang einen Puls aus. Die Länge dieses Pulses wird durch die *analogWrite* Funktion bestimmt. So wird bei „*analogWrite(0)*“ nicht einmal ein Puls ausgegeben, während „*analogWrite(255)*“ einen Puls ausgibt, der die ganze Zeit dauert, bis der nächste Puls ansteht, sodass der Ausgang eigentlich die ganze Zeit eingeschaltet ist.

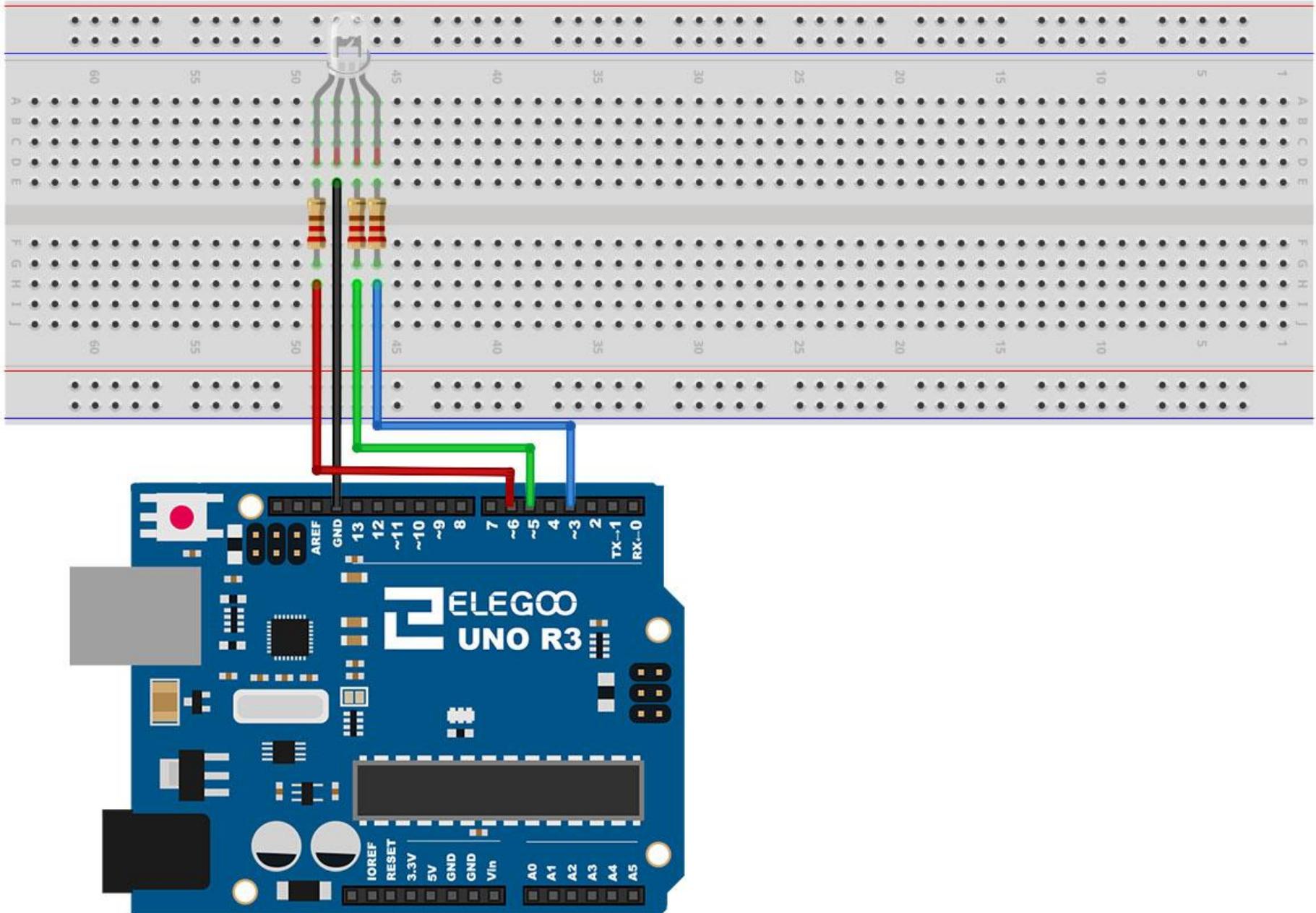
Wenn wir einen Wert irgendwo zwischen 0 und 255 für die *analogWrite*-Funktion festlegen, wird ein Puls produziert. Wenn die Länge des Pulses nur 5% der Zeit beträgt, bekommt das angeschlossene Gerät auch nur 5% der vollen Leistung.

Wenn der Ausgang bei 5V zu 90% der Zeit eingeschaltet ist, wird das Gerät 90% der vollen Leistung erhalten. Trotz des Wechsels zwischen An und Aus, können wir dieses Ein- und Ausschalten nicht an den LEDs erkennen, da unser Auge die schnell wechselnden Zustände zusammenmischt, sodass es für uns aussieht, als würde sich die Helligkeit ändern.

Verbindungsschema



Schaltplan



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 4 RGB LED“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Unser Code wird *FOR-Schleifen* benutzen, um durch die verschiedenen Farben zu wechseln.

Die erste FOR-Schleife wird von rot nach grün gehen.

Die zweite FOR-Schleife geht von grün nach blau.

Die letzte FOR-Schleife geht schließlich von blau zurück nach rot.

Probieren Sie den Sketch einmal aus, bevor wir ihn detaillierter besprechen.

Der Sketch fängt damit an, dass erstmal bestimmt wird, welche Pins für die verschiedenen Farben genutzt werden.

```
// Define Pins
#define BLUE 3
#define GREEN 5
#define RED 6
```

Der nächste Schritt ist die setup-Funktion. Wie wir in den vorherigen Lektionen gelernt haben, wird die setup-Funktion immer nach einem Reset des Arduinos ausgeführt. In diesem Fall ist die einzige Aufgabe der Funktion das Einstellen der Modi der Pins, die wir als Ausgänge für die LEDs benutzen werden.

```
void setup()
{
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
  digitalWrite(RED, HIGH);
  digitalWrite(GREEN, LOW);
  digitalWrite(BLUE, LOW);
}
```

Bevor wir uns die loop-Funktion anschauen, lassen Sie uns zuerst einen Blick auf

die letzte Funktion des Sketches werfen.

Die Variablen definieren

```
redValue = 255; // choose a value between 1 and 255 to change the color.
```

```
greenValue = 0;
```

```
blueValue = 0;
```

Diese Funktion hat drei Argumente, eins für die Helligkeit der roten, eins für die Helligkeit der grünen und eins für die Helligkeit der blauen LED. In jedem möglichen Fall werden die Werte Zahlen im Bereich von 0 bis 255 sein, wobei 0 ausgeschaltet und 255 maximale Helligkeit bedeutet. Die Funktion ruft dann analogWrite auf, um die Helligkeitswerte anschließend auch zu setzen.

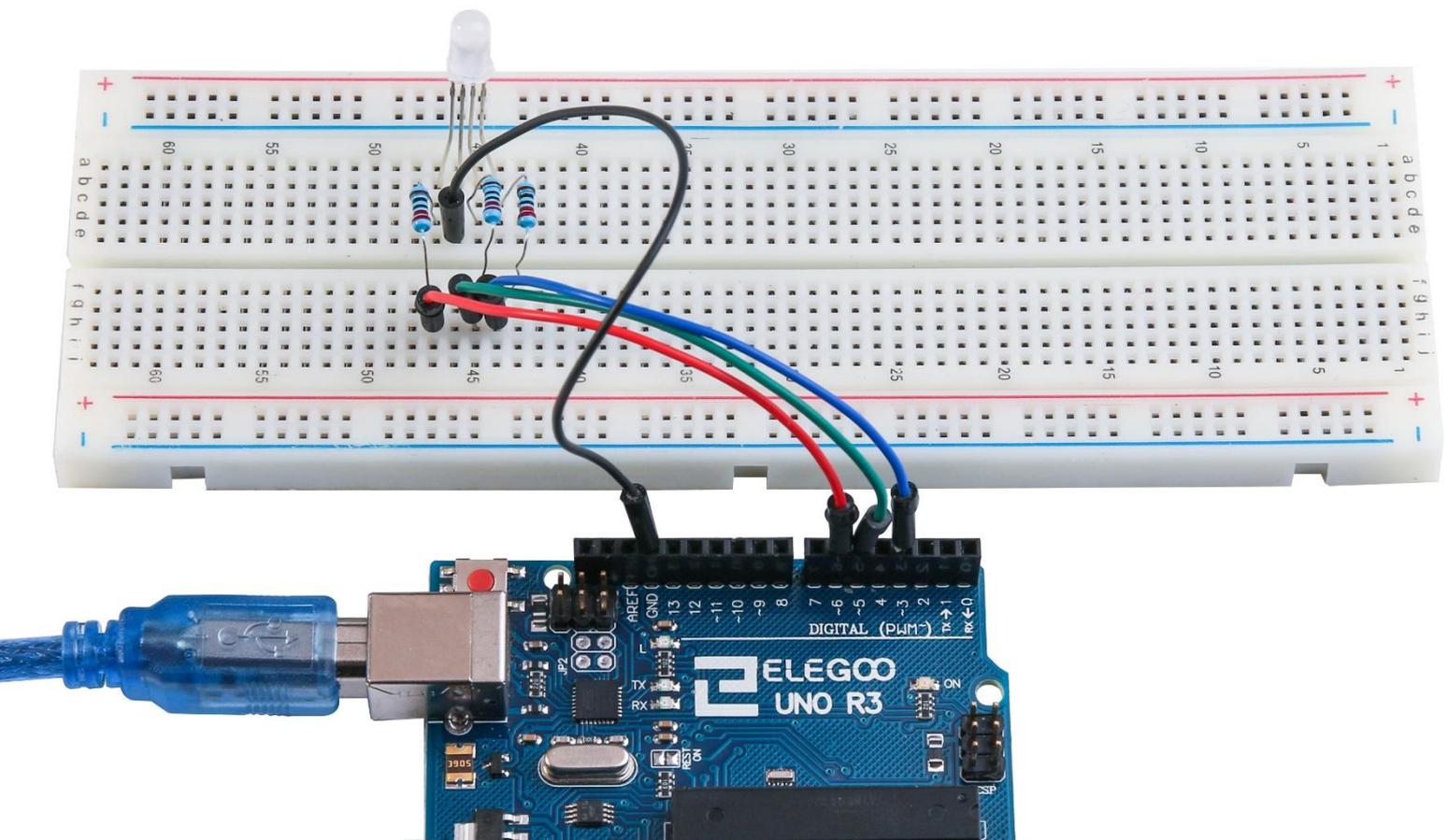
Wenn Sie nun auf die loop-Funktion schauen, können Sie sehen, dass wir die Helligkeiten von rotem, grünem und blauem Licht einstellen und dann für eine Sekunde pausieren, bevor es weiter zur nächsten Farbe geht.

```
#define delayTime 10 // fading time between colors
```

```
Delay(delayTime);
```

Versuchen Sie selber einige Farben aus den drei Grundfarben zu programmieren und schauen Sie sich den Effekt auf der LED an, wenn sie am leuchten ist.

Beispielbild



Lektion 5: Digitale Eingänge

Übersicht

In dieser Lektion werden Sie lernen Drucktaster in Verbindung mit den digitalen Eingängen des UNO Boards zu benutzen, um eine LED an- und auszuschalten.

Durch Drücken des einen Schalters wird die LED eingeschaltet, drückt man den anderen Schalter, wird sie wieder ausgeschaltet.

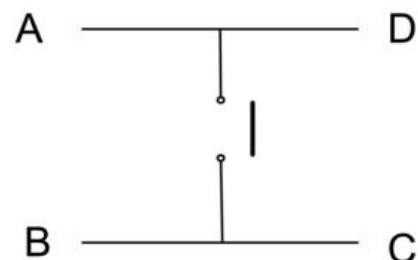
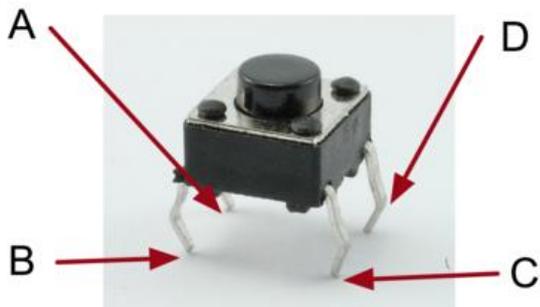
Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x 5mm rote LED
- (1) x 220 Ohm Widerstand
- (2) x Drucktaster
- (7) x M-M Kabel (Männlich zu Männlich/DuPont Jumper Kabel)

Einführung in die Komponenten

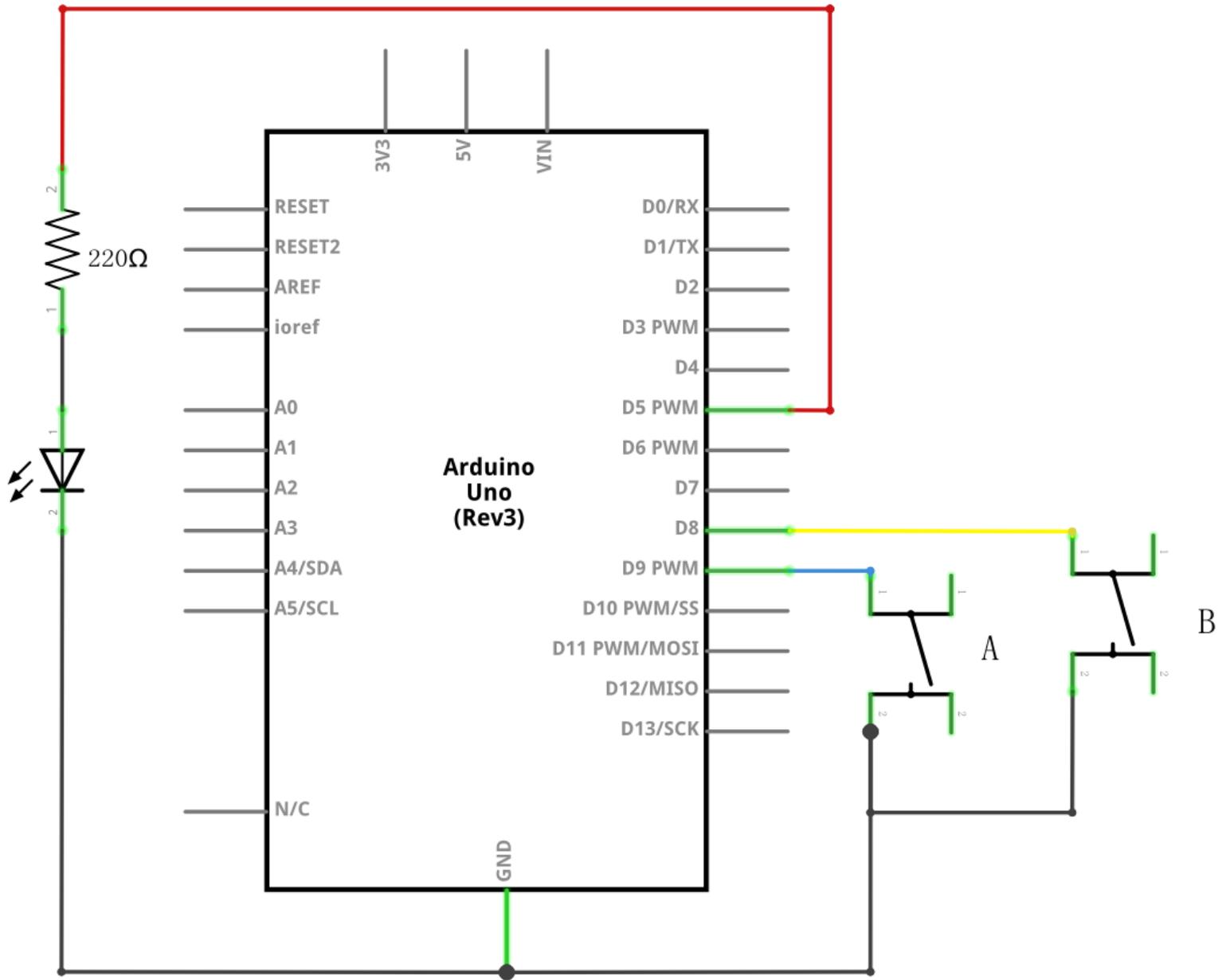
Drucktaster:

Schalter sind sehr einfache Bauteile. Wenn Sie einen Knopf drücken oder einen Hebel umlegen, verbinden sich zwei Kontakte miteinander, sodass elektrischer Strom fließen kann. Die kleinen Drucktaster, die wir in dieser Lektion benutzen werden, haben vier Kontakte, die zuerst kompliziert erscheinen.



Eigentlich gibt es nur zwei richtige elektrische Verbindungen. Innerhalb des Schalters sind die Pins A und D sowie die Pins B und C miteinander verbunden.

Verbindungsschema



Obwohl die Schalter quadratisch sind, können sie nicht in jede Richtung aufgesteckt werden. Das bedeutet, dass die Pins richtig auf dem Breadboard platziert werden müssen, damit sie in die Mitte des Breadboards passen.

Denken Sie daran, dass die LED den positiven Anschluss auf der rechten Seite hat.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 5 Digital Inputs“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Laden Sie den Sketch auf Ihr UNO Board. Durch Drücken des linken Schalters wird die LED eingeschaltet, ein Druck auf den rechten Schalter schaltet sie wieder aus.

Der erste Teil des Sketches definiert drei Variablen für die drei Pins, die benutzt werden. Der „*ledPin*“ ist der Ausgang für die LED, der „*buttonApin*“ verweist auf den Schalter, der näher oben am Breadboard liegt und der „*buttonBpin*“ steht für den unteren Schalter.

Die setup-Funktion definiert den ledPin als einen gewöhnlichen Ausgang (OUTPUT), wohingegen die beiden Schalterpins als Eingänge behandelt werden müssen. In diesem Fall setzen wir den pin-Modus auf „*INPUT_PULLUP*“:

```
pinMode(buttonApin, INPUT_PULLUP);
```

```
pinMode(buttonBpin, INPUT_PULLUP);
```

Der pin-Modus INPUT_PULLUP bedeutet, dass der Pin als Eingang behandelt werden soll. Gleichzeitig bestimmt er, dass wenn nichts an den Eingang angeschlossen ist, der Standardwert für den Eingang „*HIGH*“ ist. In anderen Worten: Der Standardwert für den Pin ist HIGH, außer er wurde durch einen betätigten Schalter auf „*LOW*“ geschaltet.

Das ist der Grund, warum die Schalter mit GND verbunden sind. Wenn ein Taster gedrückt wurde, verbindet er den Eingangs-Pin mit GND, sodass der Eingang nicht länger HIGH geschaltet ist. Da der Eingang im Grundzustand HIGH geschaltet und bei gedrücktem Schalter LOW ist, ist die Logik umgekehrt als eigentlich üblich. Darum werden wir uns in der loop-Funktion kümmern.

```

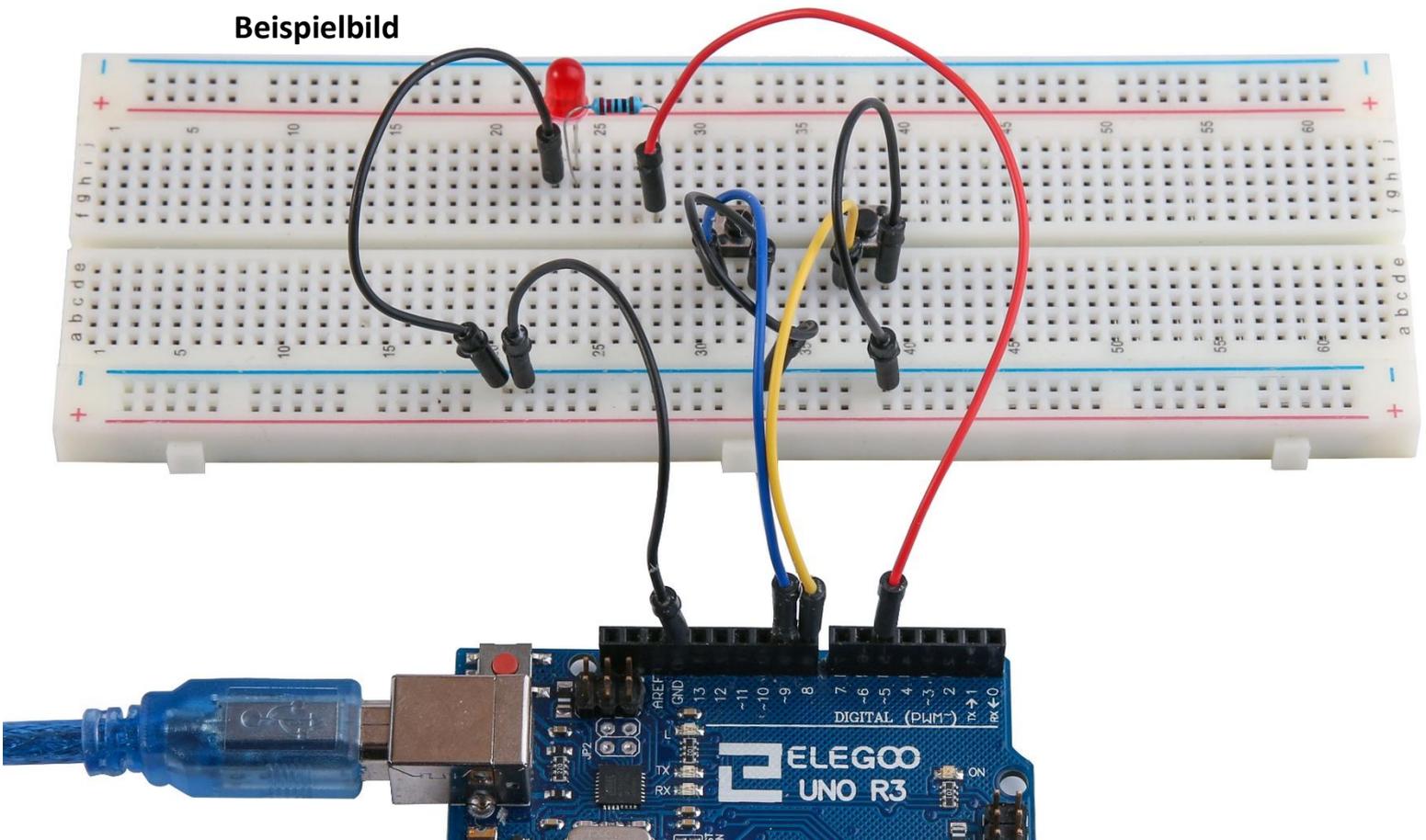
void loop()
{
  if (digitalRead(buttonApin) == LOW)
  {
    digitalWrite(ledPin, HIGH);
  }
  if (digitalRead(buttonBpin) == LOW)
  {
    digitalWrite(ledPin, LOW);
  }
}

```

In der loop-Funktion befinden sich zwei „if“-Statements (Bedingungsanweisungen). Für jeden Schalter eins. Beide if-Statements haben als Bedingung ein „digitalRead“, das den Zustand des jeweiligen Pins beobachtet.

Bedenken Sie, dass bei jedem Druck eines Tasters der Wert dessen Pins LOW sein wird. Wenn der Taster A gedrückt wird, wird *buttonApin* LOW sein. Das erste if-Statement (*digitalRead(buttonApin) == LOW*) wird damit erfüllt sein und die LED zum leuchten bringen (*digitalWrite(ledpin, HIGH)*;). Ähnlich funktioniert es beim zweiten Taster. Wenn Taster B gedrückt wird, erkennt dies das zweite If-Statement (*digitalRead(buttonBpin) == LOW*) und schaltet die LED auf LOW (= aus).

Beispielbild



Lektion 6: Aktiver Buzzer

Übersicht

In dieser Lektion lernen Sie, wie Sie einen Ton aus einem aktiven Buzzer ausgeben.

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x AktiverBuzzer

(2) x W-M Kabel (Weiblich zu Männlich DuPontJumper Kabel)

Einführung in die Komponenten

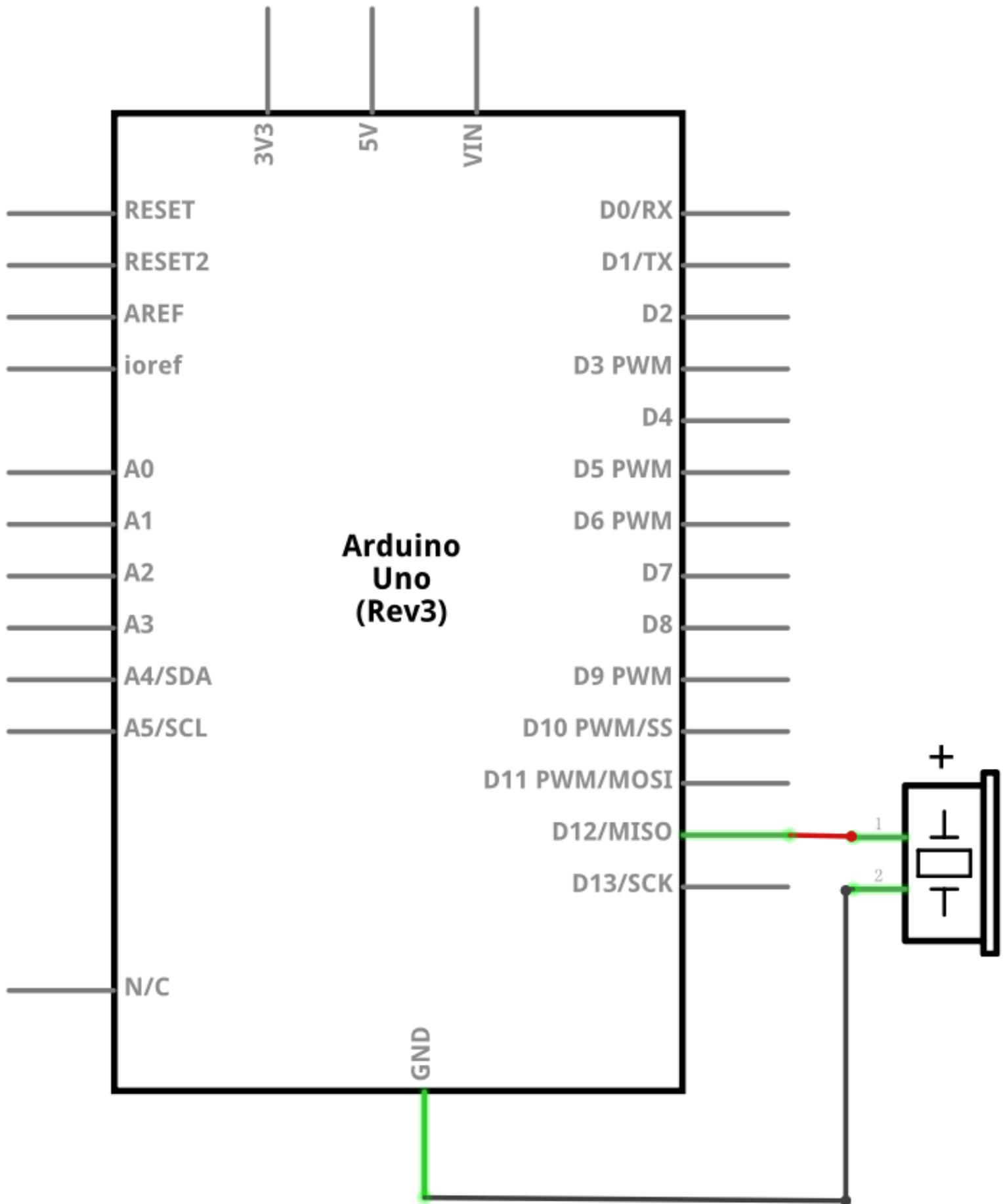
BUZZER:

Elektronische Buzzer (Summer) werden mit Gleichstrom betrieben und kommen mit einem integrierten Schaltkreis. Sie werden häufig in Computern, Druckern, Kopierern, Alarmsystemen, elektronischen Spielzeugen, Automobilelektronik, Telefonen, Timern und anderen elektronischen Geräten benutzt, um Töne auszugeben. Man kann zwischen aktiven und passiven Buzzern unterscheiden. Wenn man einen aktiven mit einem passiven Buzzer vergleicht, stellt man fest, dass der mit dem grünen PCB (Platine) der passive ist, während der mit dem schwarzen PCB der aktive Buzzer ist.

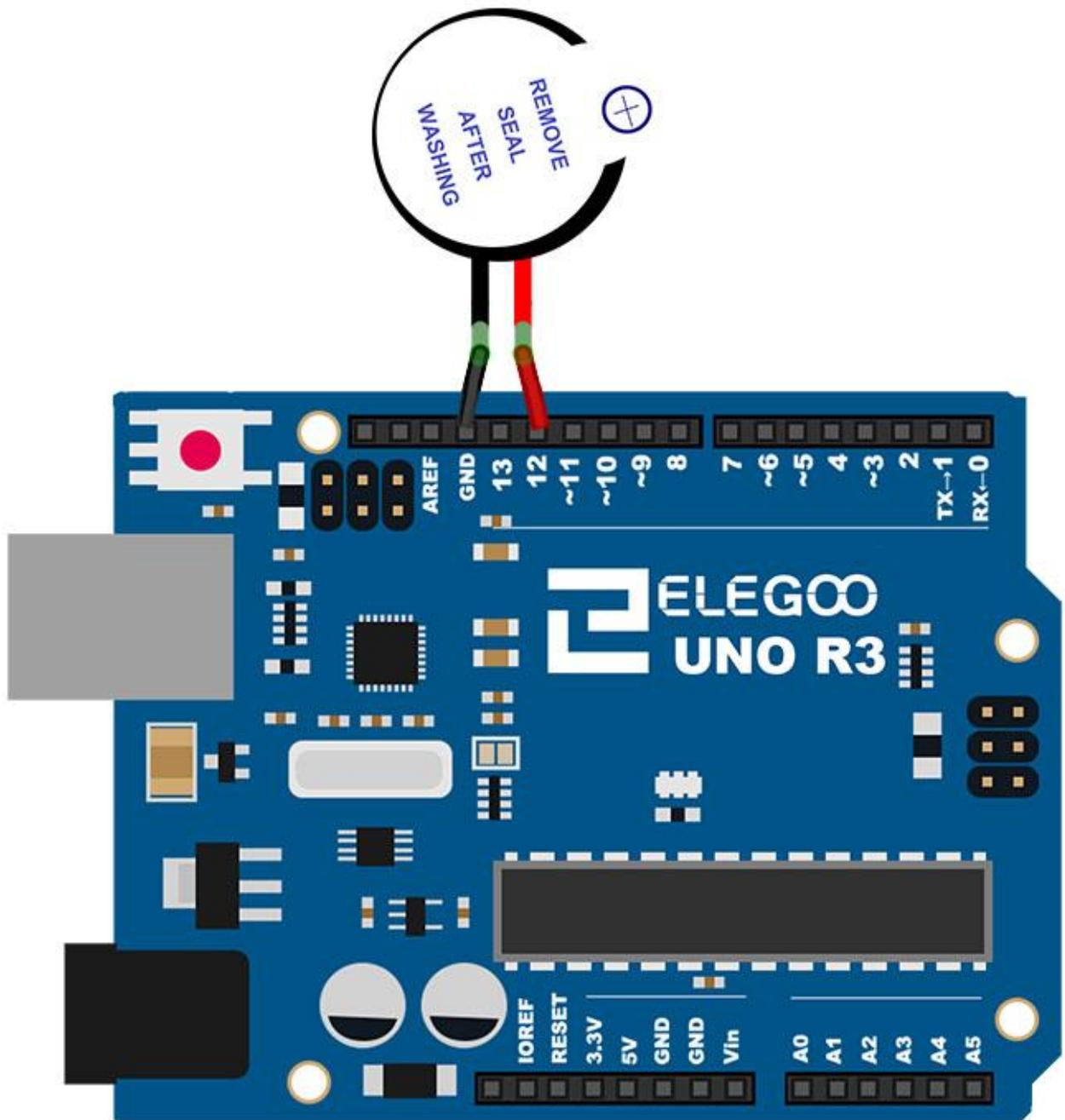
Der Unterschied zwischen den beiden Arten ist, dass der aktive Buzzer eine eingebaute Signalquelle hat, die automatisch einen Ton erzeugt, wenn Gleichstrom angelegt wird. Ein passiver Buzzer hat keine solche Signalquelle und wird daher nicht ertönen, wenn man eine Gleichstromquelle direkt anschließt. Stattdessen muss der passive Buzzer mit Rechteckschwingungen mit einer Frequenz zwischen 2000 und 5000Hz versorgt werden. Ein aktiver Buzzer ist wegen der eingebauten Schaltkreise oft teurer als ein passiver.



Verbindungsschema



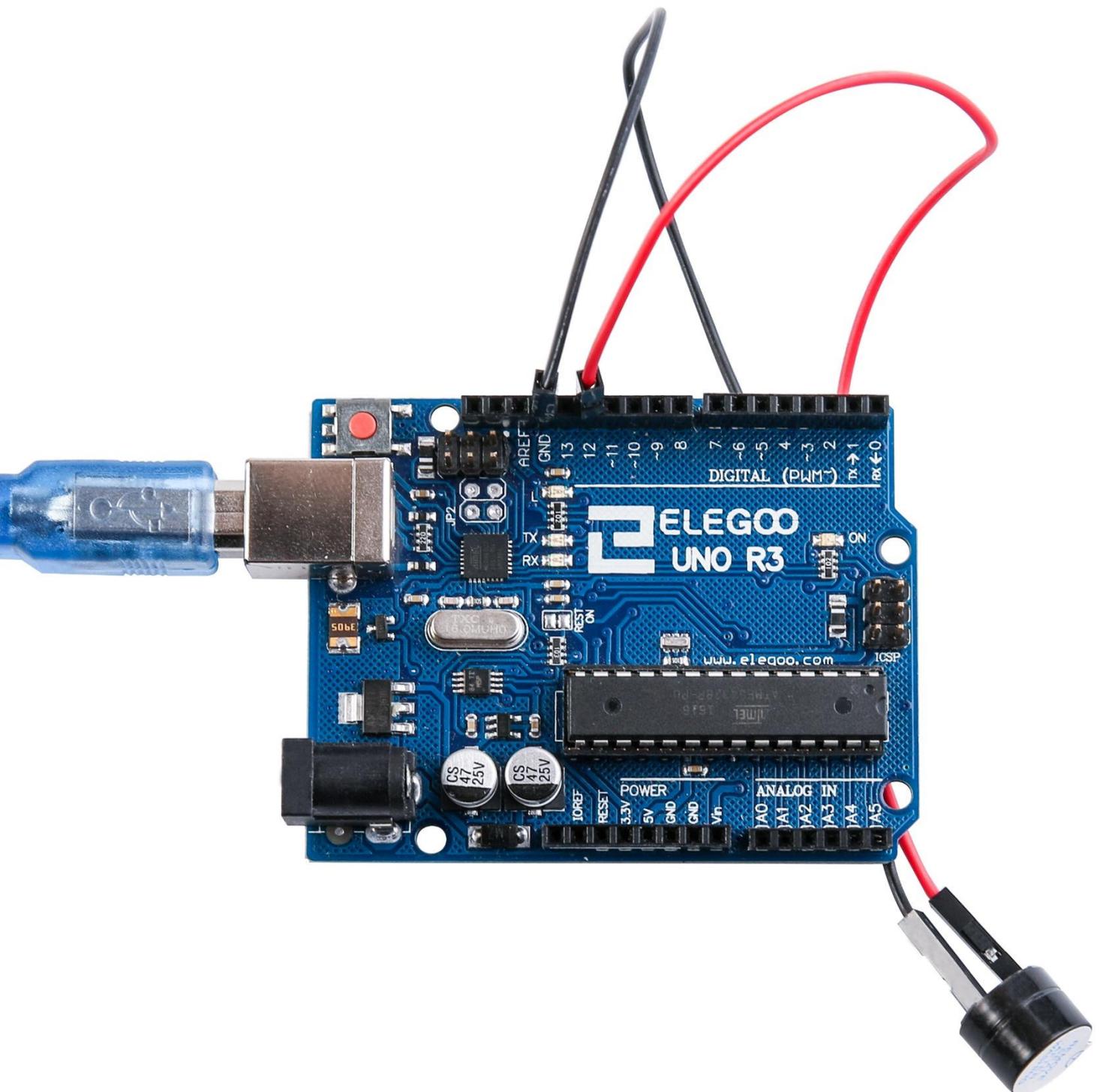
Schaltplan



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 6 Making Sounds“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Beispielbild



Lektion 7:Passiver Buzzer

Übersicht

In dieser Lektion lernen Sie einen passiven Buzzer zu benutzen.

Das Ziel dieser Lektion ist es, acht unterschiedliche Töne erklingen zu lassen, jeden Ton für 0,5 Sekunden:*Alt Do*(523Hz),*Re* (587Hz), *Mi* (659Hz), *Fa* (698Hz), *Sol* (784Hz), *La* (880Hz), *Ti* (988Hz) und das*Soprane Do* (1047Hz).

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x Passiver Buzzer

(2) x W-M Kabel (Weiblich zu Männlich DuPont Jumper Kabel)

Einführung in die Komponenten

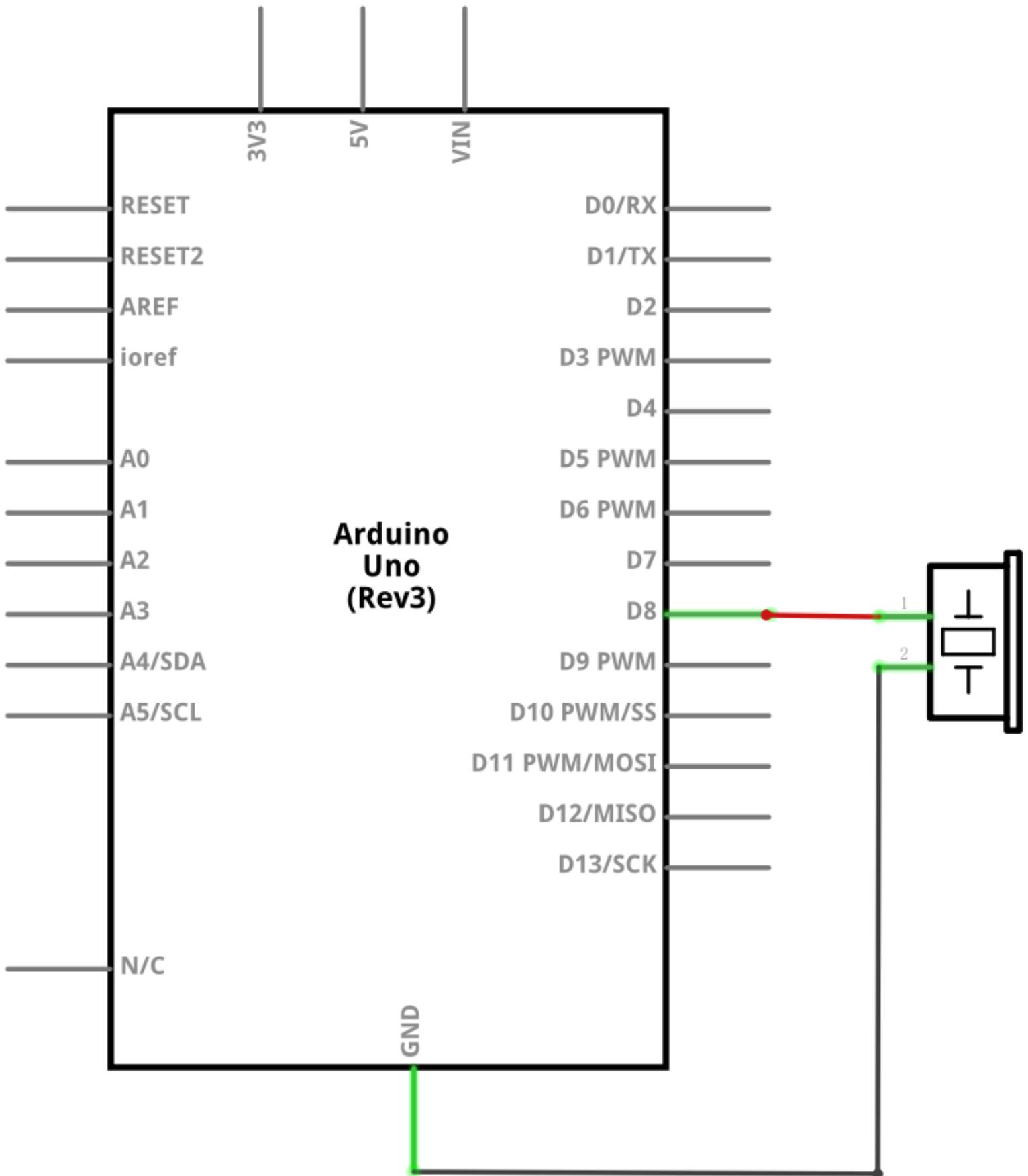
Passiver Buzzer:

Das Funktionsprinzip des passiven Buzzers ist es durch PWM-generierte Audiosignale die Luft zum vibrieren zu bringen. Je nachdem wie hoch die Frequenz des Signals ist, werden verschiedene Töne generiert. Ein Audiosignal von beispielsweise 587Hz (Hertz: Frequenzeinheit) wäre ein *Alt Do*, 587Hz wären ein *Re* und 659Hz wären ein *Mi*. Mit passiven Buzzern kann man durch schnelle Frequenzwechsel Lieder abspielen.

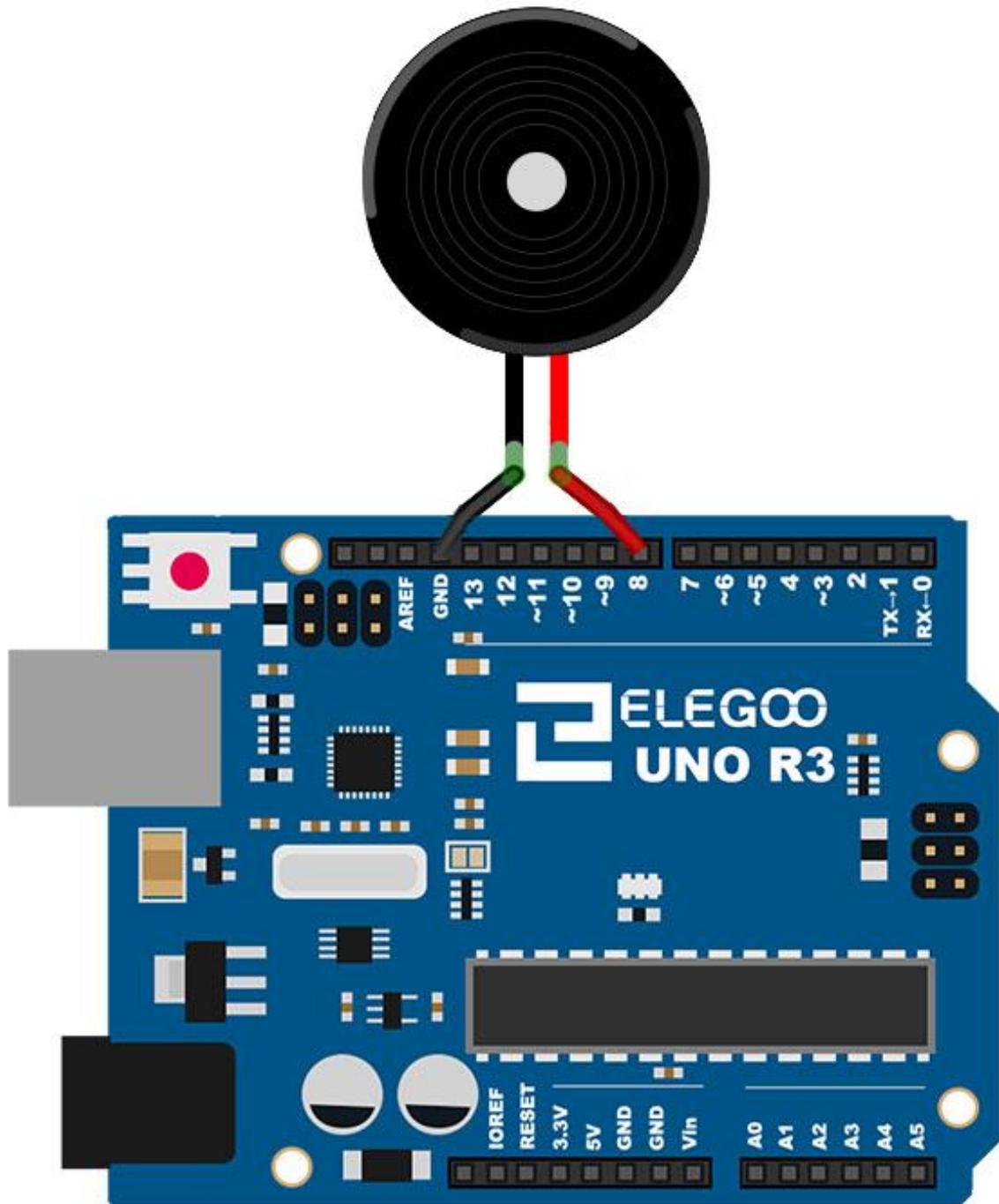
Zum Generieren des Pulses für das PWM-Signal des passiven Buzzers dürfen wir auf keinen Fall die analogWrite-Funktion des Boards benutzen, da ihr Puls auf 500Hz festgelegt ist und somit ein immergleiches Signal ausgegeben werden würde.



Verbindungsschema



Schaltplan



Verbinden Sie den Buzzer mit dem UNO R3 Board. Das rote Kabel muss mit *Pin 8* verbunden werden, das schwarze Kabel mit GND.

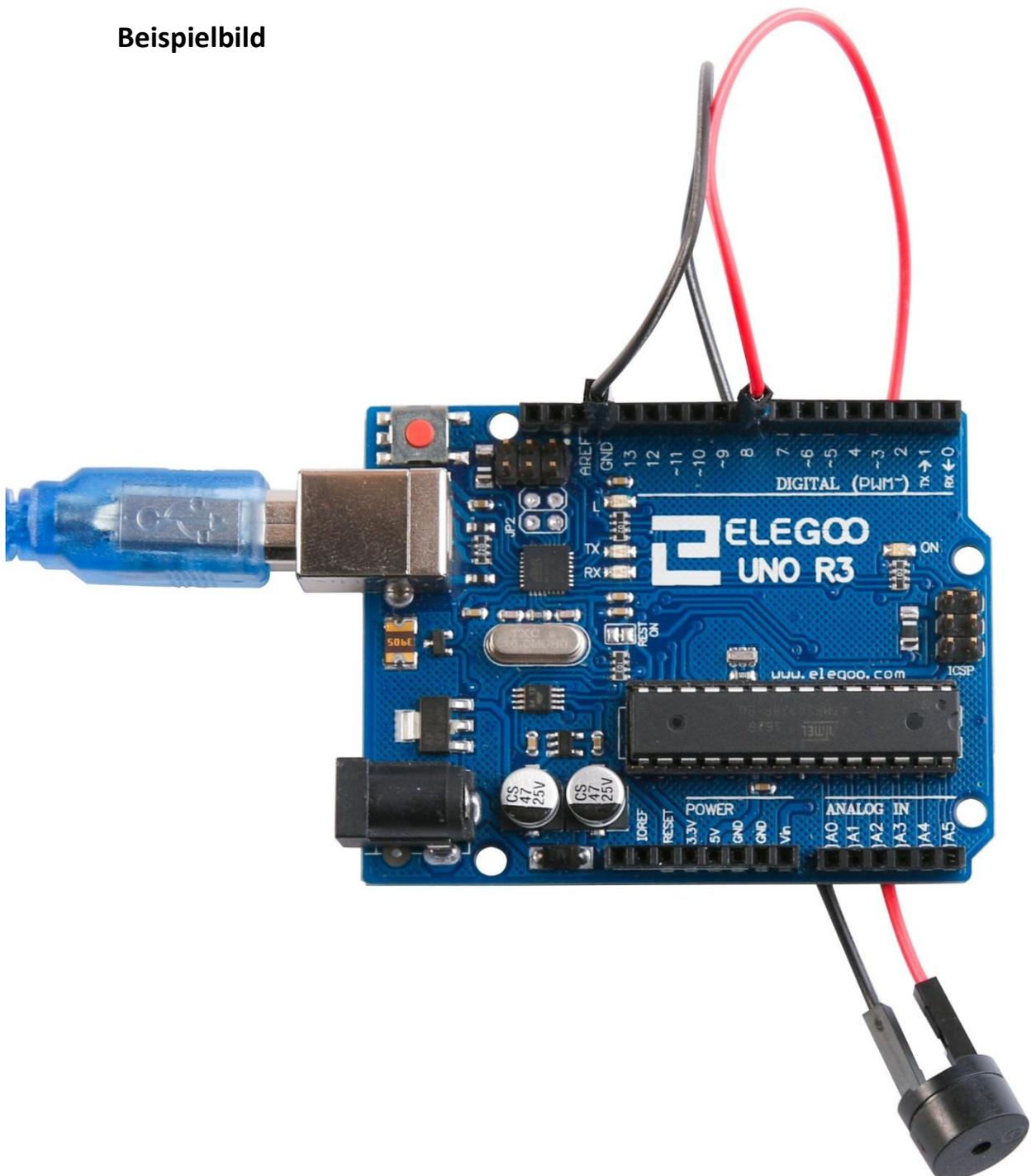
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 7 Passive Buzzer“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „*pitches*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen.

Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Lektion 8: Neigungssensor

Übersicht

In dieser Lektion lernen sie, wie man einen Neigungssensor benutzt, um den Grad der Neigung herauszufinden.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x Neigungssensor
- (2) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)



Einführung in die Komponenten

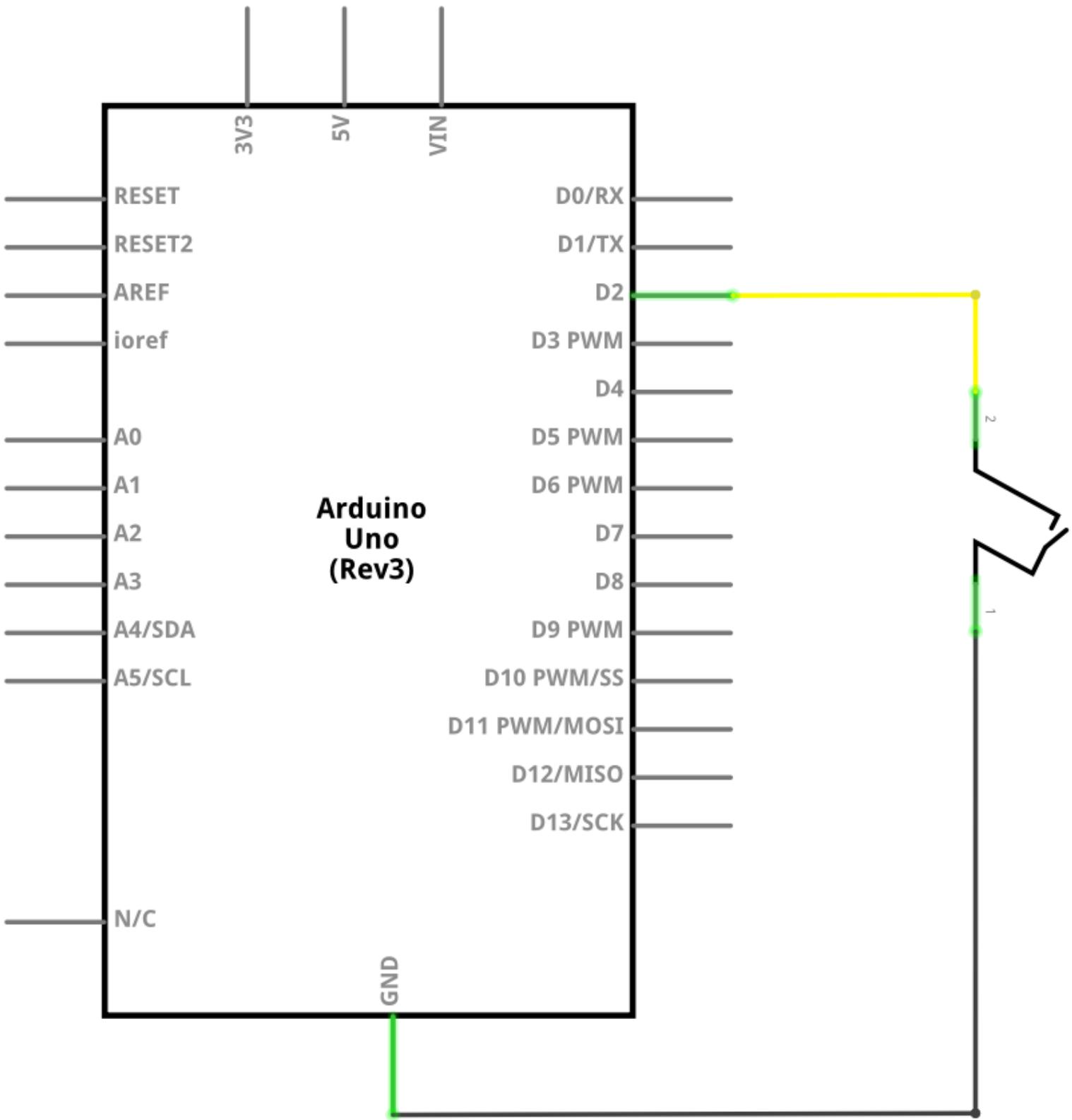
Neigungssensor:

Neigungssensoren erlauben einem die momentane Ausrichtung oder Neigung festzustellen. Sie sind klein, günstig, einfach zu benutzen und verbrauchen wenig Strom. Wenn sie richtig benutzt werden, halten sie ewig. Durch ihren simplen Aufbau werden sie häufig in Spielzeugen, Gadgets und anderen Geräten eingesetzt.

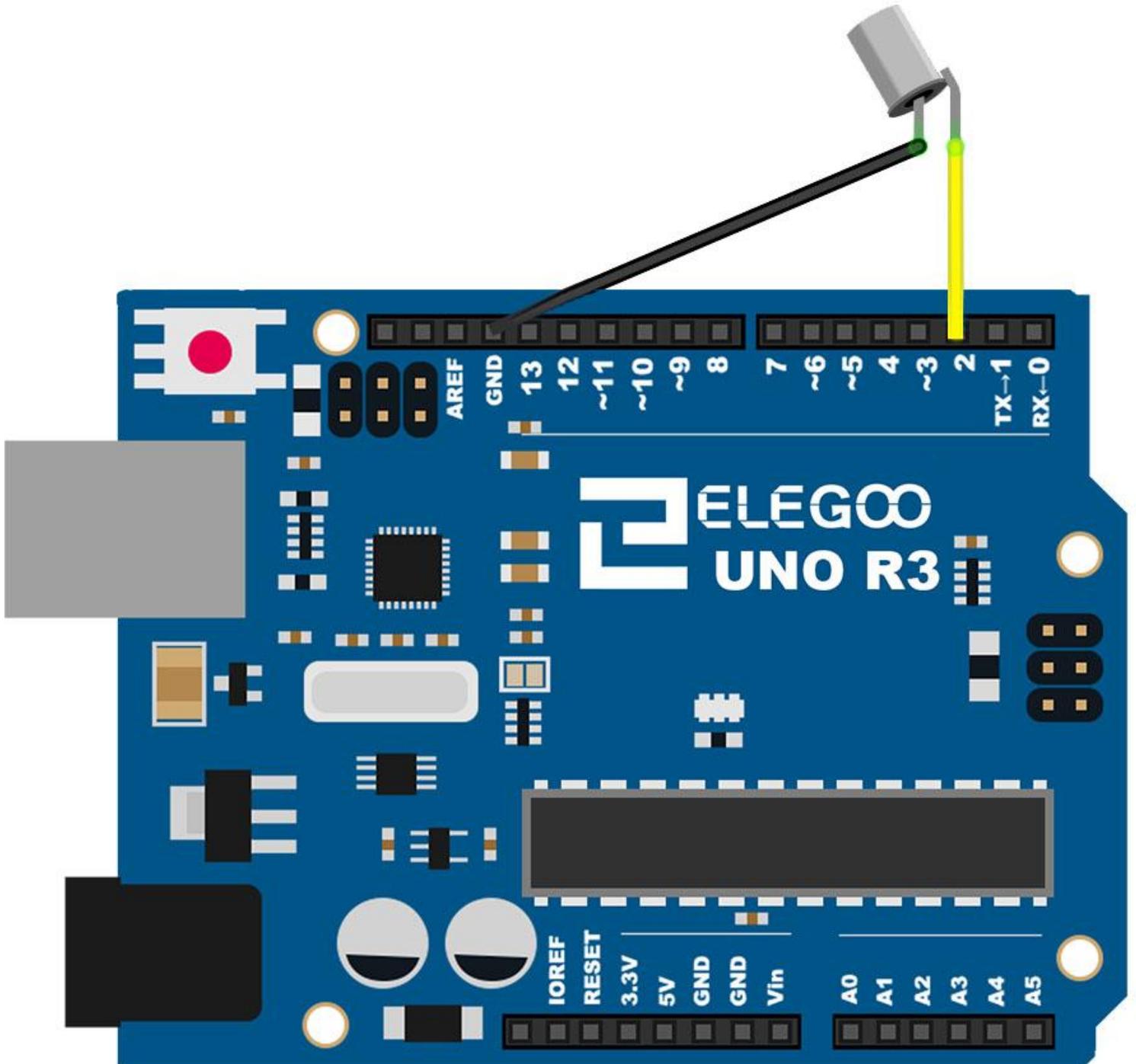
Sie bestehen in der Regel aus einer Art Hohlraum (häufig in Form eines Zylinders) mit einer leitfähigen beweglichen Masse innerhalb. An einem Ende des Hohlrums befinden sich zwei leitende Elemente (Pole). Wenn der Neigungssensor senkrecht, wie im Bild oben, ausgerichtet ist, rollt die bewegliche Masse auf die beiden Pole und verbindet sie miteinander, wie bei einem Schalter.

Obwohl sie nicht so präzise und flexibel wie Beschleunigungssensoren sind, können Neigungssensoren Erschütterungen erkennen oder die grobe Neigung bestimmen. Ein weiterer Vorteil ist, dass große Neigungssensoren direkt die Stromversorgung schalten können. Beschleunigungssensoren geben dagegen digitale oder analoge Spannungswerte aus, die erst durch einen weiteren Schaltkreis oder Mikrocontroller analysiert werden müssen.

Verbindungsschema



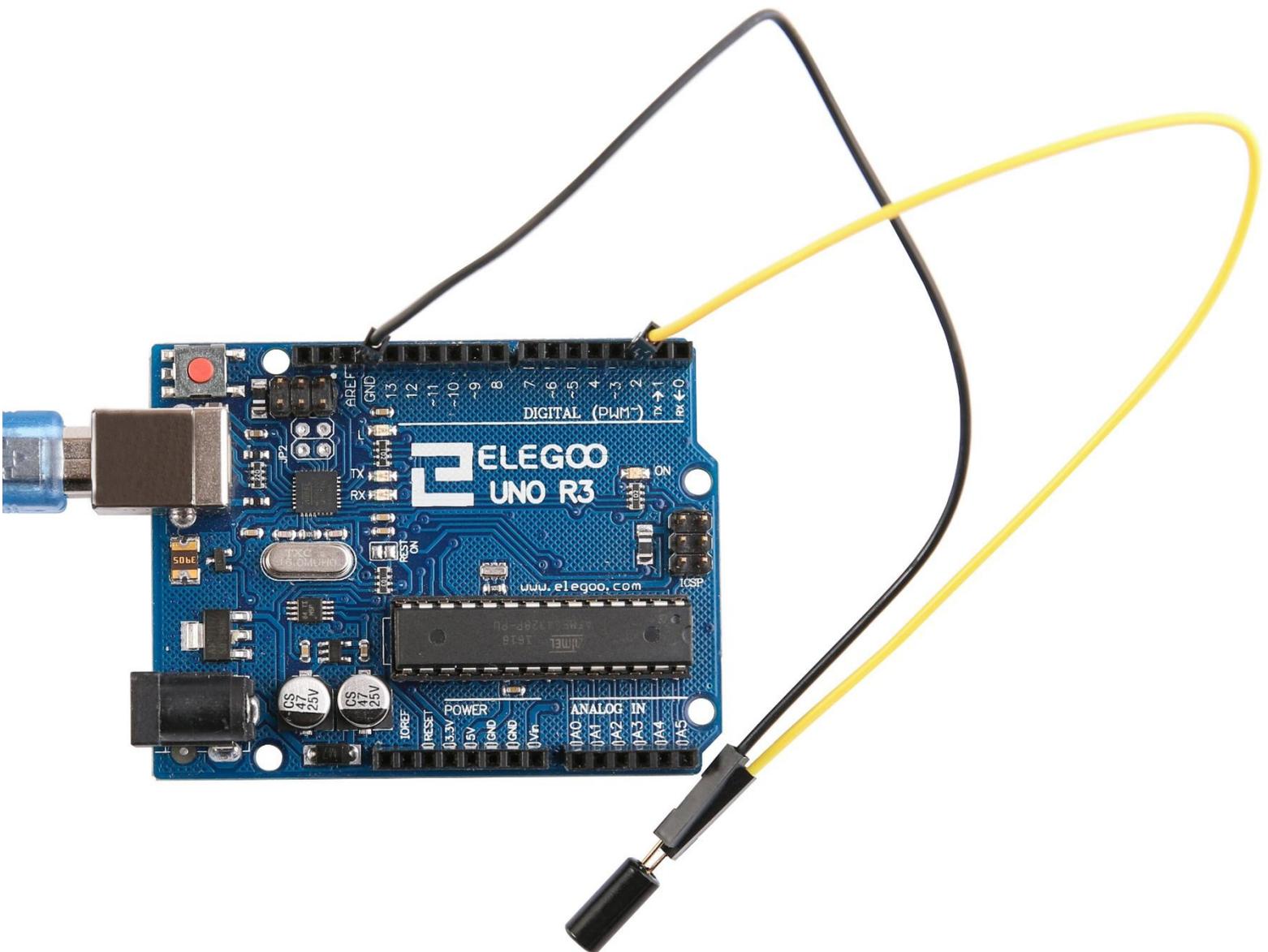
Schaltplan



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 8 Ball Switch“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Beispielbild



Lektion 9: Servo-Motor

Übersicht

Servomotoren sind Motoren, die sich nur um 180 Grad drehen lassen. Sie werden durch das Senden von elektrischen Pulsen vom UNO R3 Board gesteuert. Diese Pulse steuern, zu welcher Position sich der Servo bewegen soll. Der Servomotor hat drei Anschlüsse. Der braune Anschluss wird mit GND verbunden, der rote Anschluss ist der positive Stromanschluss und wird mit einer 5V Quelle verbunden und der orangene Anschluss ist der Signalanschluss, der mit dem Digitalen *Pin 9* Anschluss des UNO Boards verbunden wird.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x Servomotor (SG90)
- (3) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

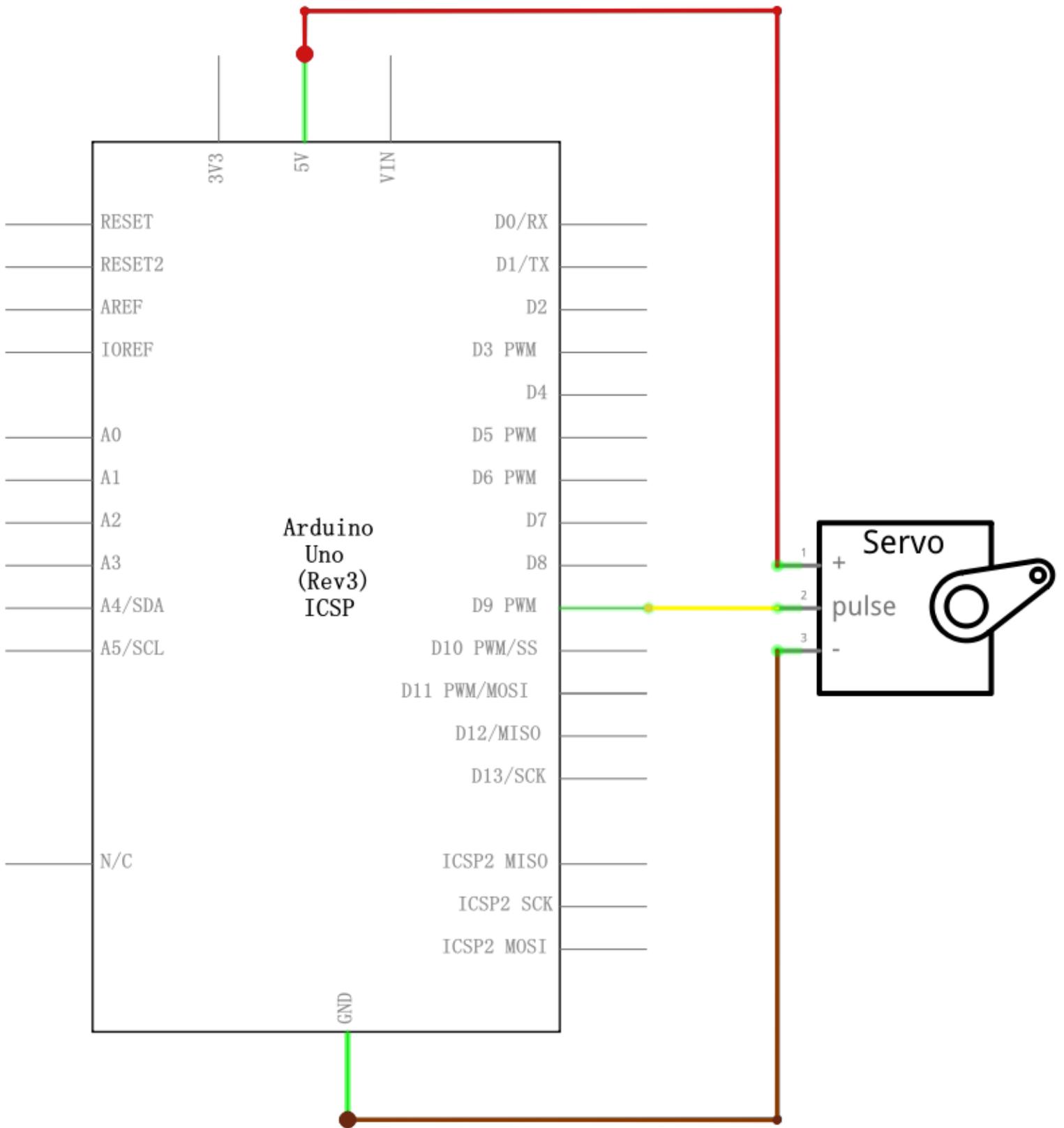
Einführung in die Komponenten

SG90

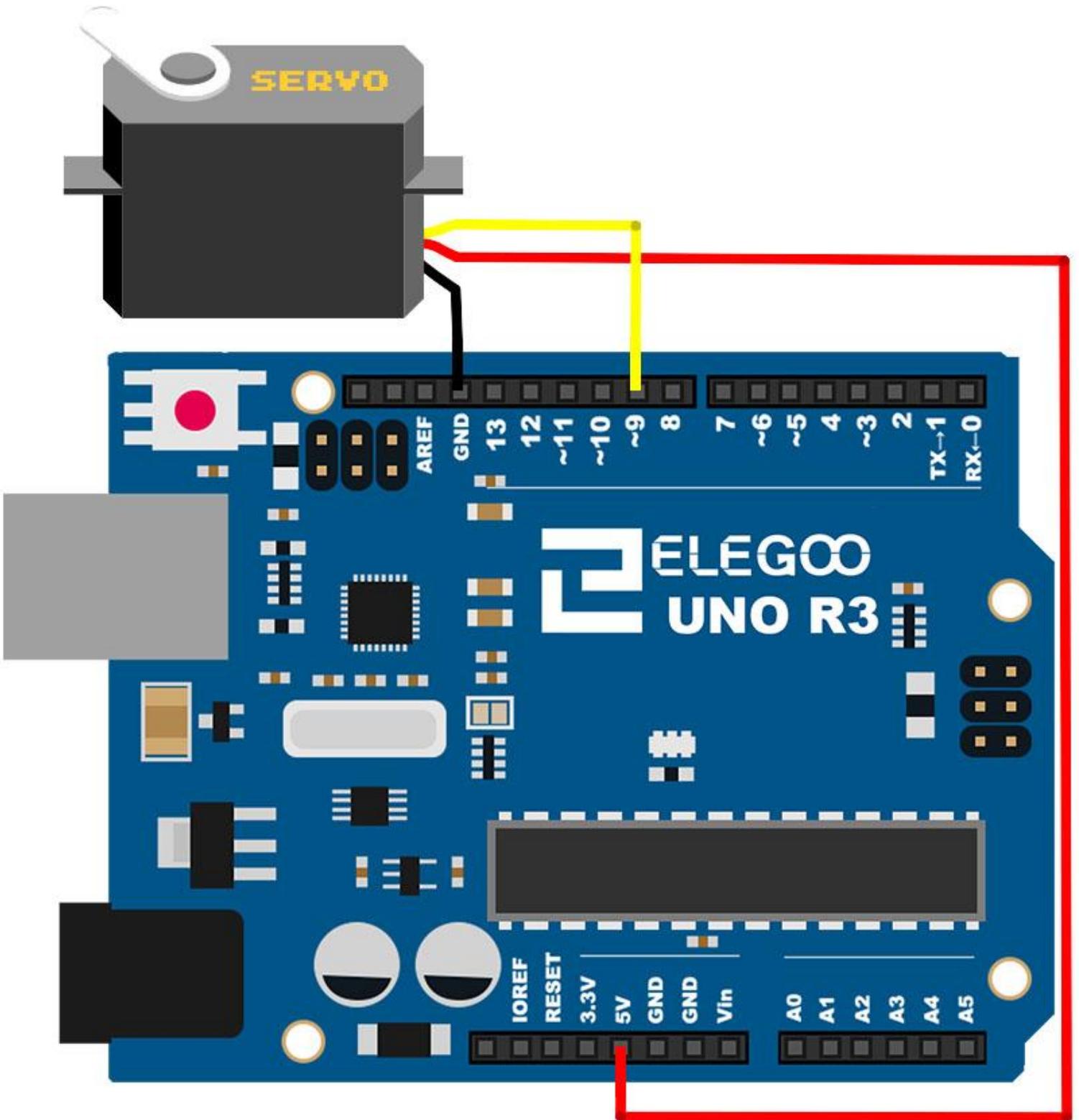
- Universeller Anschluss für JR- und FP-Stecker
- Kabellänge: 25cm
- Geschwindigkeit (ohne Last): 0,12 sec / 60 Grad (4,8V); 0,10 sec / 60 Grad (6,0V)
- Drehmoment (4.8V): 1.6kg/cm
- Betriebstemperatur: -30 ~ 60°C
- Totbandbreite: 5us
- Betriebsspannung: 3,5~6V
- Abmessungen: 3.2 cm x 3 cm x 1.2 cm
- Gewicht: 134g



Verbindungsschema



Schaltplan



Code

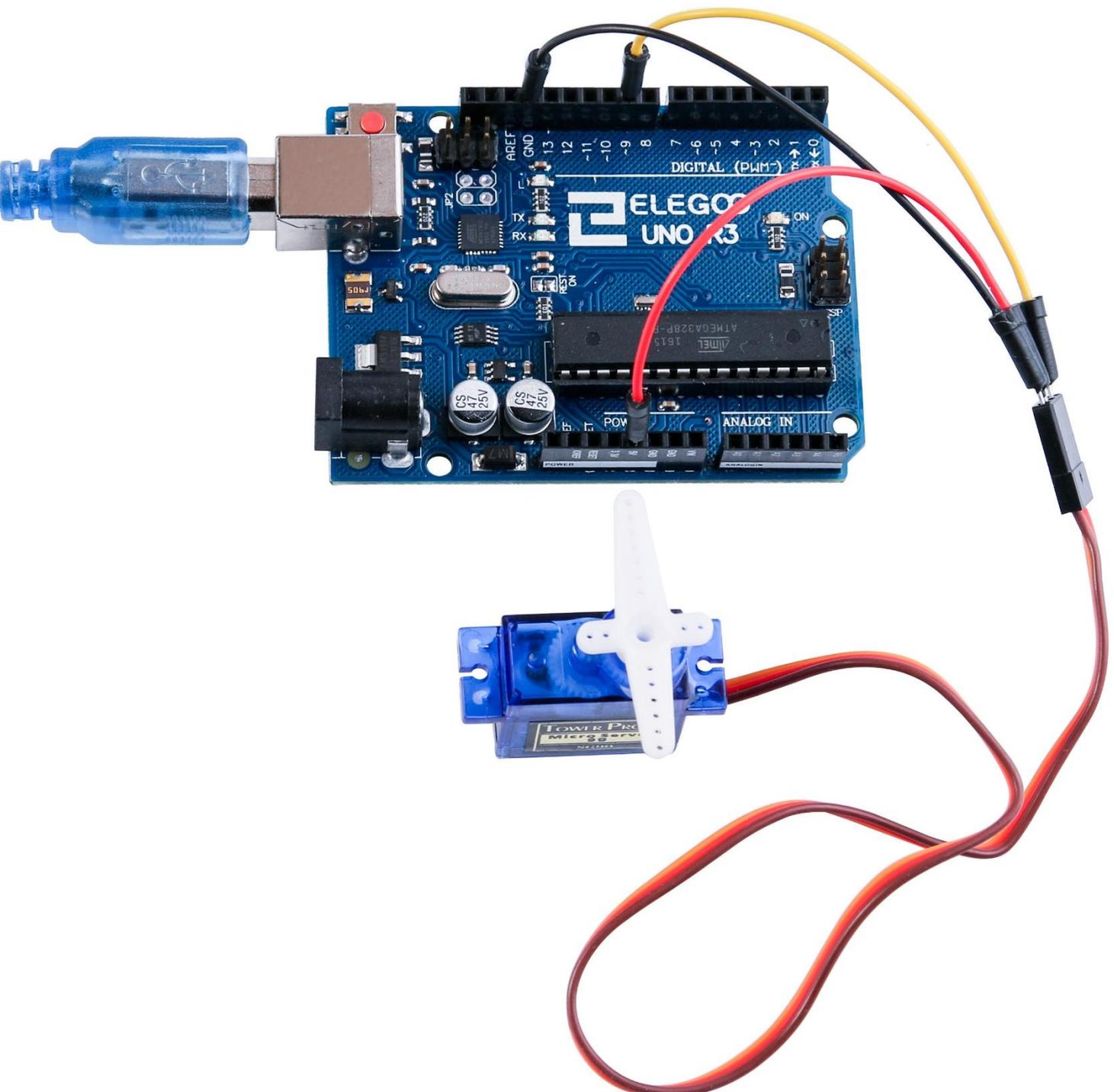
Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 9 Servo“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „Servo“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen.

Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild

Im Bild ist der braune Anschluss des Servos durch ein schwarzes M-M Kabel mit GND, der rote Anschluss durch ein rotes M-M Kabel mit 5V und der orangene Anschluss durch ein gelbes M-M Kabel mit Pin 9 des Boards verbunden.



Lektion 10:Ultraschallsensormodul

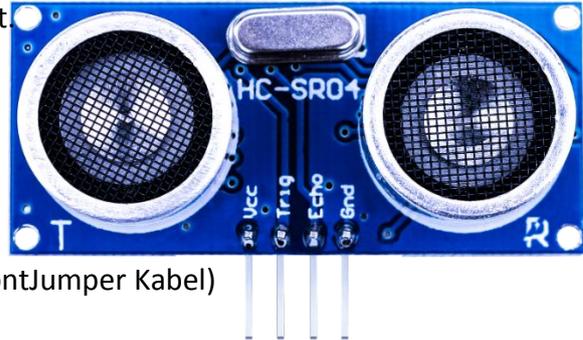
Übersicht

Ultraschallsensoren eignen sich für alle Projekte, bei denen Entfernungen gemessen werden oder Hindernisse umgangen werden müssen.

Das HC-SR04 Modul ist günstig und einfach zu nutzen, da es eine speziell auf diesen Sensor zugeschnittene Bibliothek gibt.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x Ultraschallsensormodul
- (4) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)



Einführung in die Komponenten

Ultraschallsensor

Das Ultraschallsensormodul HC-SR04 erlaubt es uns, Entfernungen von 2-400cm mit einer Genauigkeit bis zu 3mm zu messen. Das Modul beinhaltet Ultraschallsender, -Empfänger und einen Steuerschaltkreis. Das grundlegende Funktionsprinzip ist:

- (1) Ein HIGH-Signal am *Trigger*-Anschluss des Moduls für min. 10µs zum aktivieren.
- (2) Das Modul sendet automatisch acht 40 kHz Schall-Impulse und prüft, ob ein Signal zurückkommt.
- (3) Wenn ein Signal (Schallimpuls) zurückgekommen ist, sendet das Modul auf dem *Echo*-Anschluss ein HIGH-Signal in der Länge, die der Impuls gebraucht hat, um im Modul wiederanzukommen. Dadurch lässt sich die Entfernung zum nächsten Objekt wie folgt berechnen:

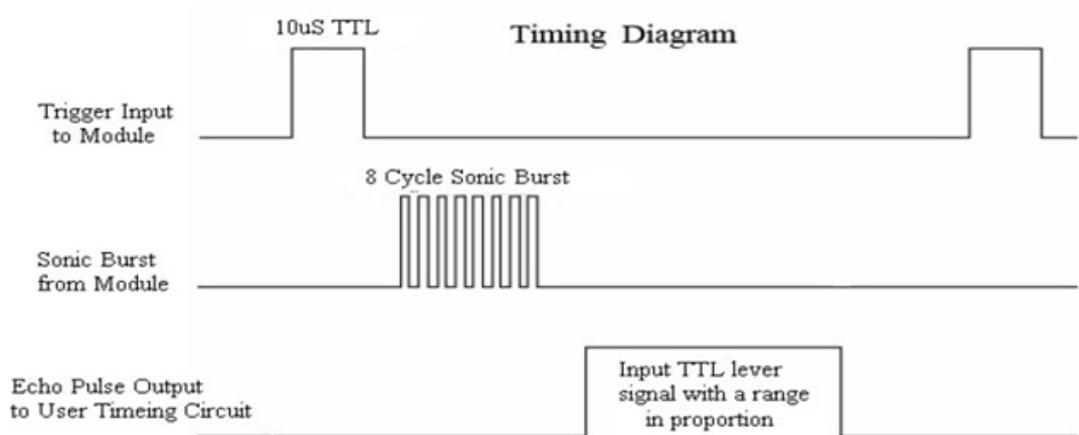
$$\text{Entfernung} = \text{High-Signal-Länge} \times \text{Schallgeschwindigkeit (340 m/s)} / 2$$

Die Entfernung berechnet sich aus der halben Signallänge (daher am Ende geteilt durch 2) multipliziert mit der Schallgeschwindigkeit.

Das Zeitdiagramm sehen Sie auf der nächsten Seite. Sie müssen nur ein kurzes (10µs langes) HIGH-Signal zum Trigger-Pin senden, um die Entfernungsmessung zu starten.

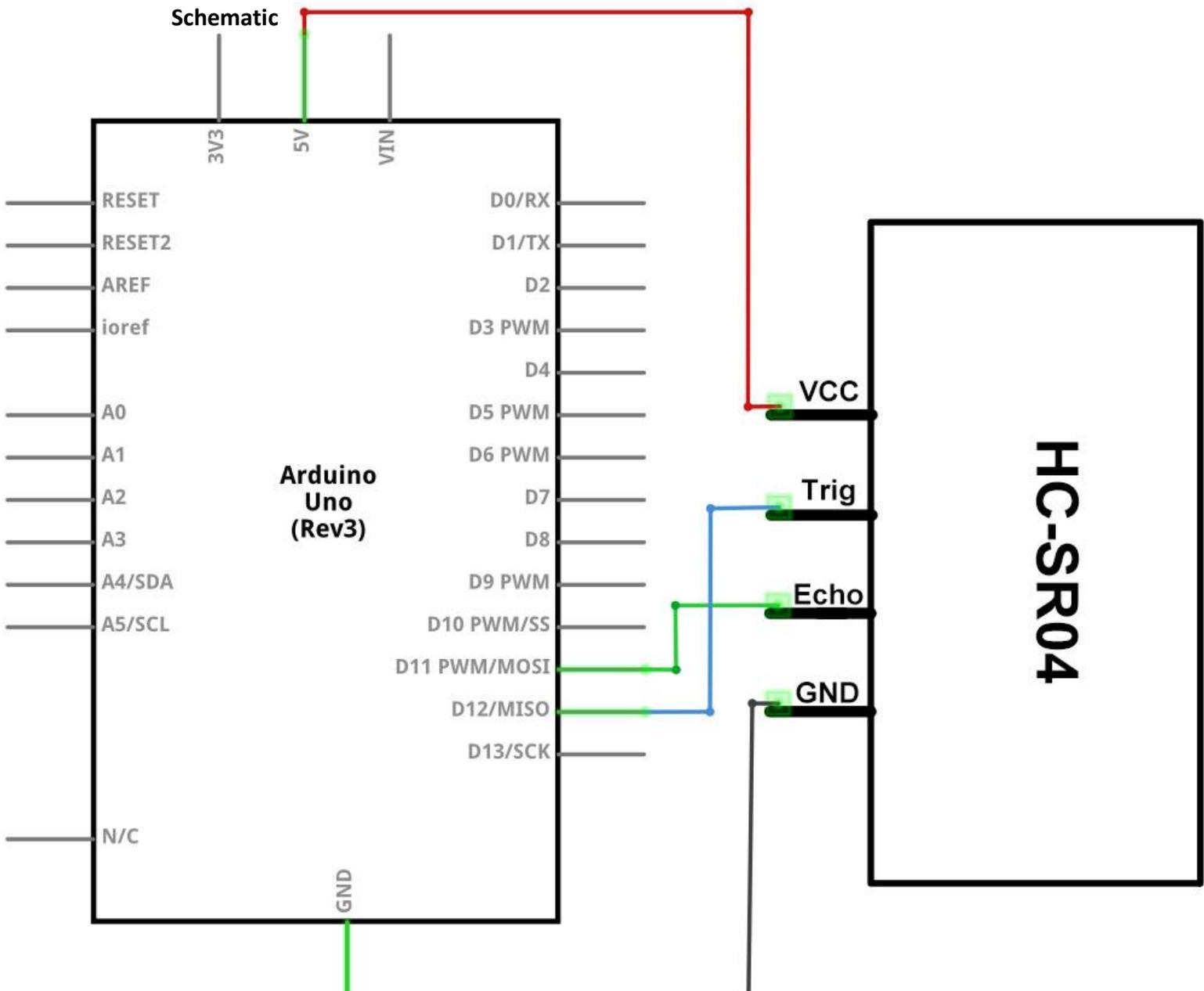
Das Modul sendet dann schnell hintereinander 8 Schallimpulse bei 40 kHz aus und wartet auf das Echo der Impulse und sendet die Dauer des Messvorgangs dann

über den Echo-Pin aus. Wir empfehlen Ihnen im Abstand von 60ms zu messen, um

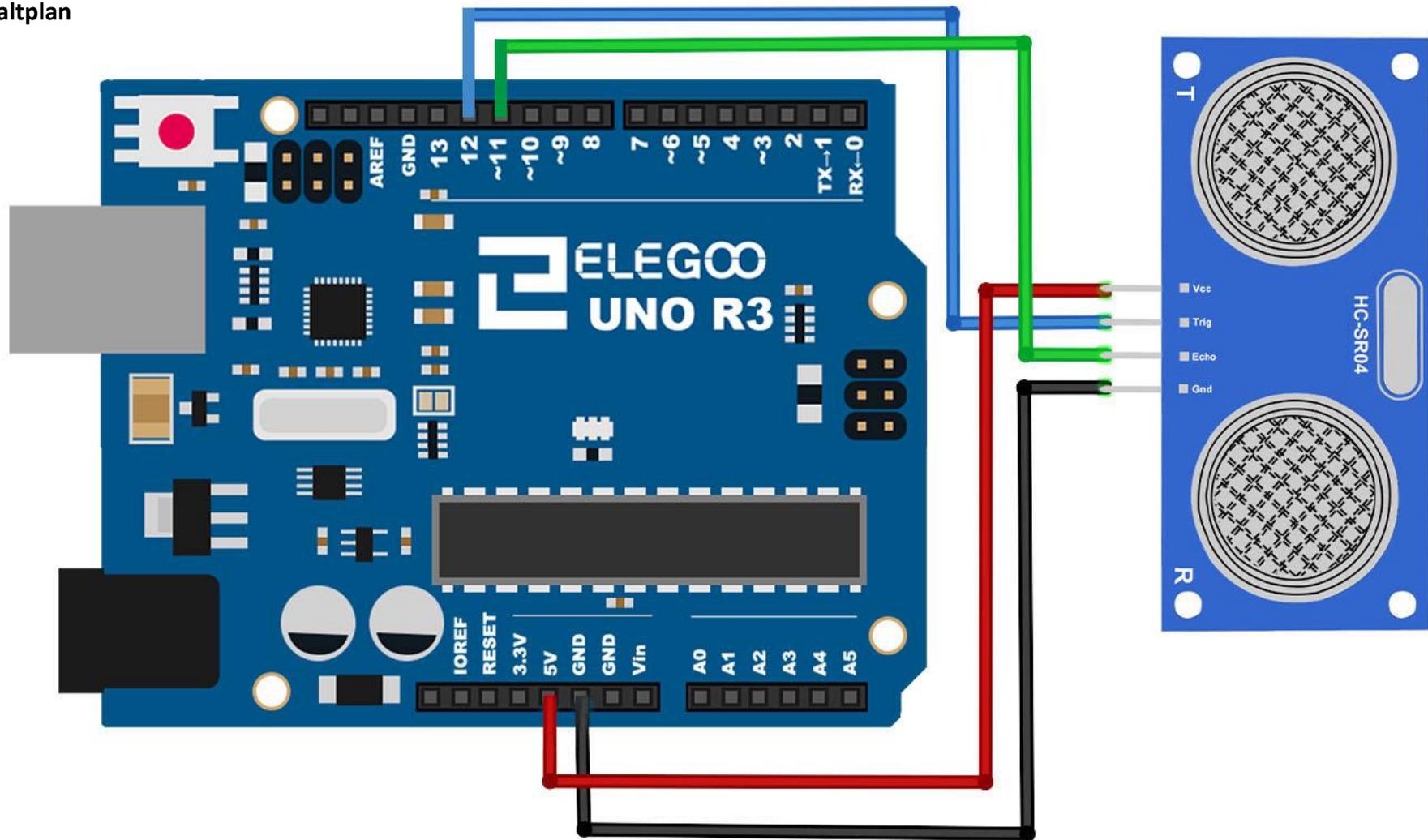


Connection

Schematic



Schaltplan

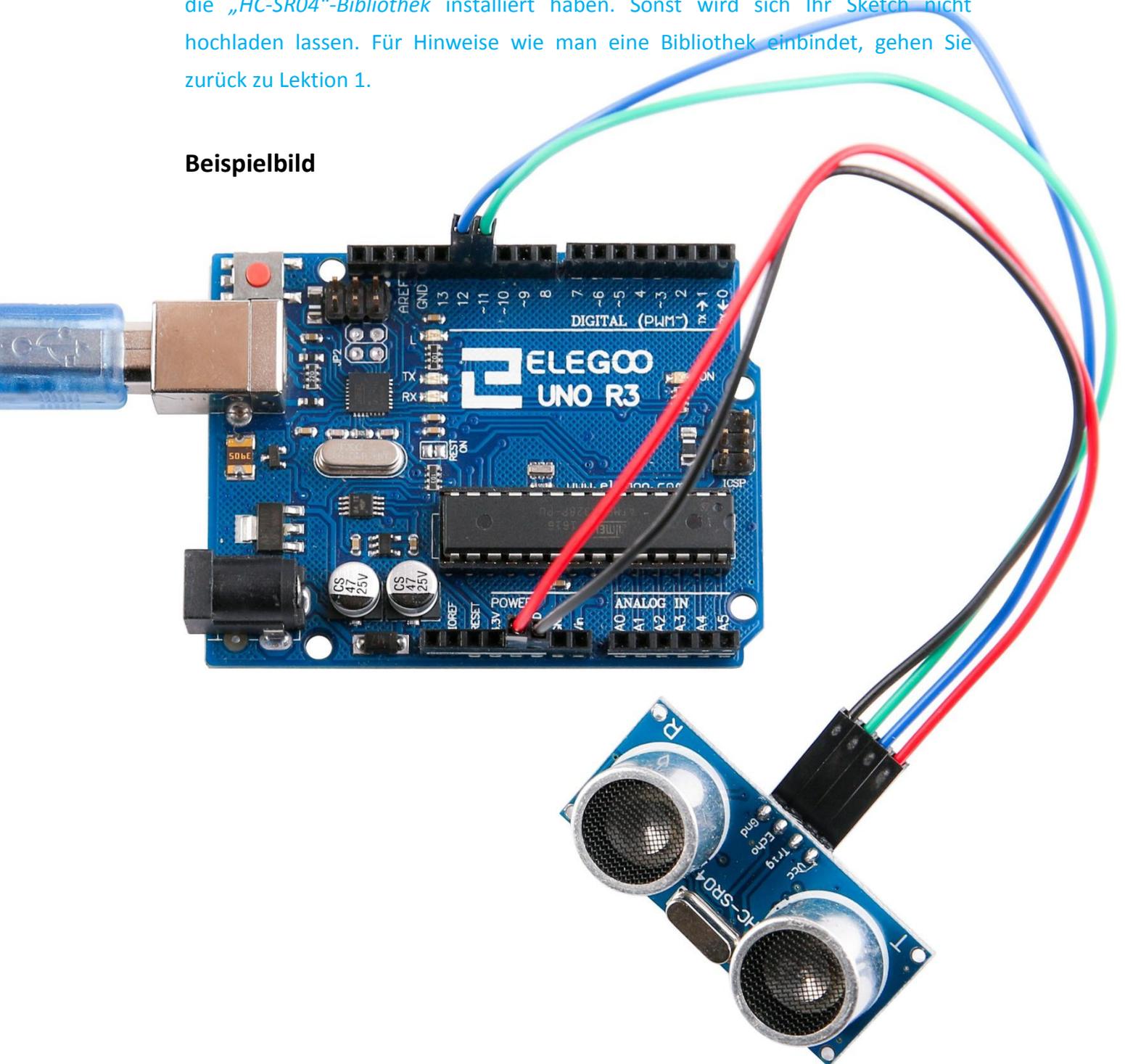


Code

Das Benutzen einer Bibliothek, die an diesen Sensor angepasst ist, macht den Code kurz und einfach. Wir binden die Bibliothek in den ersten Zeilen ein und steuern dann durch einfache Befehle, wie sich der Sensor verhalten soll.

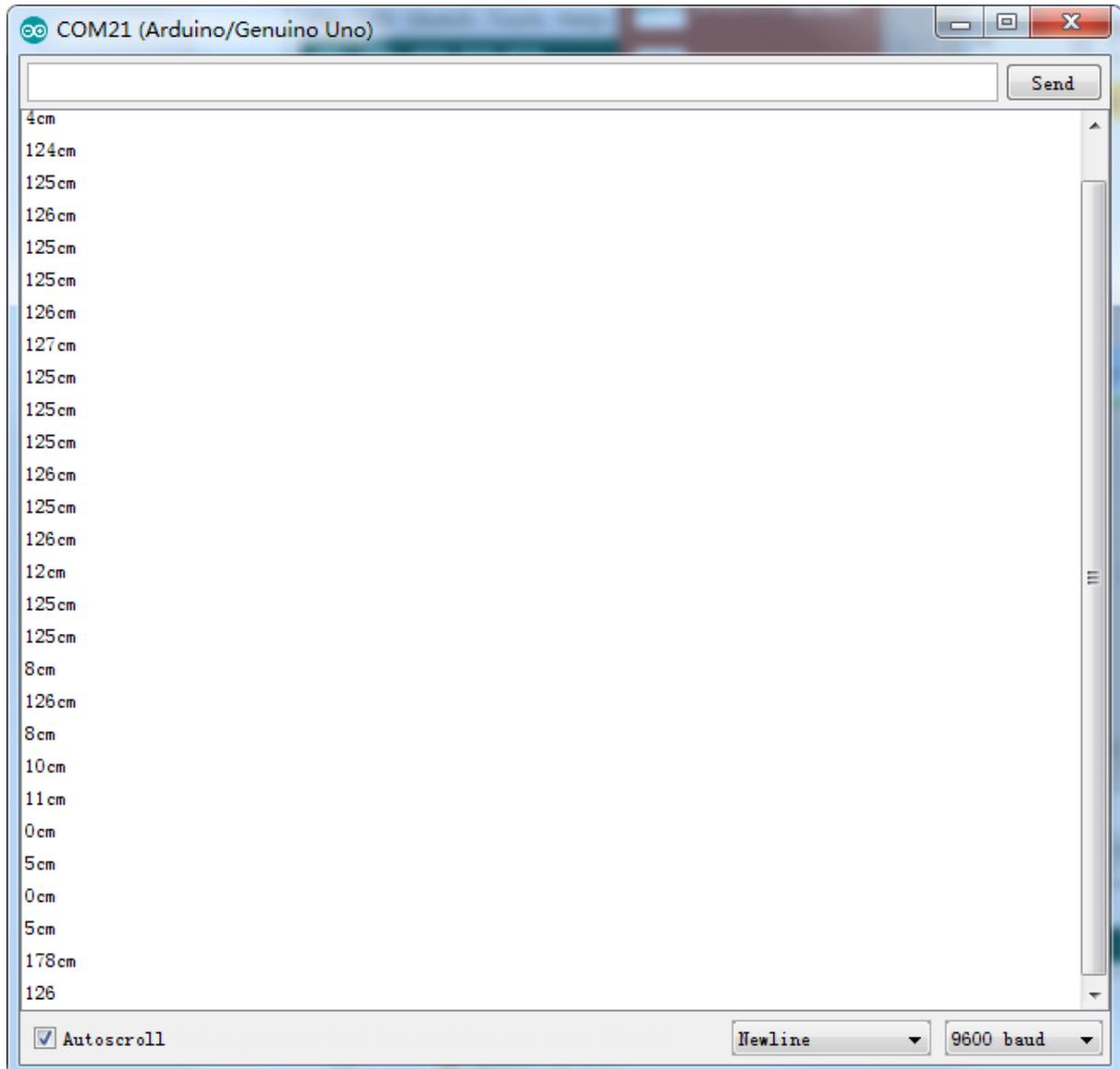
Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 10 Ultrasonic Sensor Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an. Bevor Sie diesen Sketch hochladen können, müssen Sie die „HC-SR04“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen und die momentane Entfernung in Echtzeit angezeigt zu bekommen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



Lektion 11:Folientastaturmodul

Übersicht

In diesem Projekt werden wir eine Folientastatur integrieren, sodass wir mit unserem UNO R3 Board die gedrückten Tasten feststellen können. Tastaturen werden in allen möglichen Geräten verwendet: In Telefonen, Faxgeräten, Mikrowellen, Öfen, Türschlössern, usw. Es gibt sie praktisch überall. Millionen von Geräten benutzen Tastaturen zur Eingabe.

Zu wissen, wie man eine Tastatur mit einem Mikrocontroller wie einem UNO R3 Board verbindet, ist daher sehr nützlich, da viele verschiedene kommerzielle Produkte auf ähnliche Weise funktionieren.

Am Ende, wenn alles richtig miteinander verbunden und programmiert ist, wird auf Knopfdruck der Serielle Monitor am Computer geöffnet werden. Um das Ganze erst einmal einfacher zu halten, starten wir damit, die gedrückte Taste einfach nur anzeigen zu lassen.

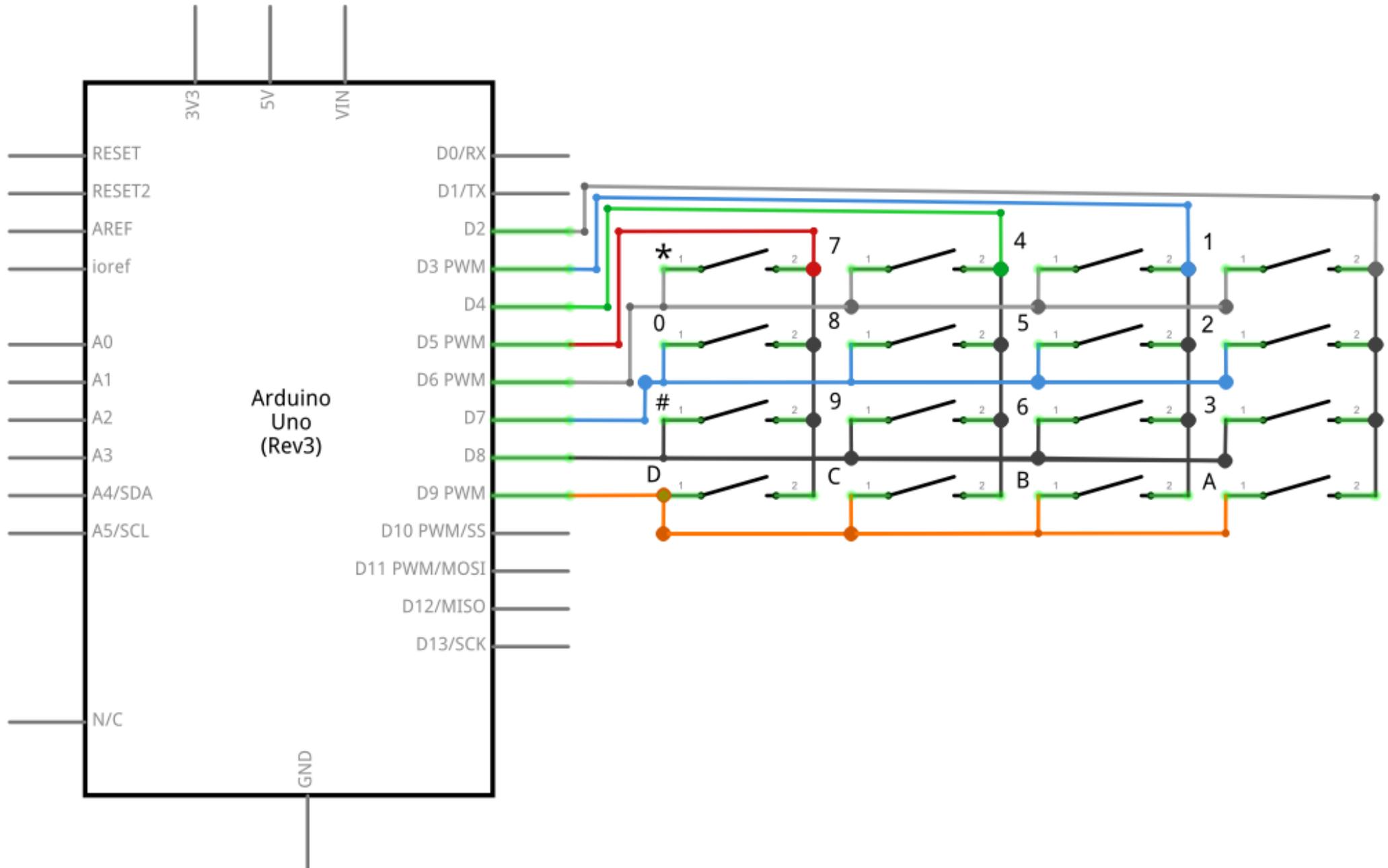
Für dieses Projekt werden wir eine *Matrix Tastatur* benutzen. Das ist eine Tastatur, die die Eingaben kodiert, sodass man mit weitaus weniger Kabeln auskommt, als es bei Einzeltasten der Fall wäre. Die Matrix Tastatur, die wir benutzen, hat beispielsweise 16 Tasten (0-9, A-D und * und #), aber nur 8 Anschlüsse. Bei einer linearen Tastatur bräuchte man 17 Anschlüsse (für jede Taste ein positiver Anschluss und ein gemeinsamer negativer / GND). Die Matrix Kodierung erlaubt es, weniger Anschlüsse zu nutzen und ist daher viel effizienter als lineare Tastaturen.

Benötigte Bauteile:

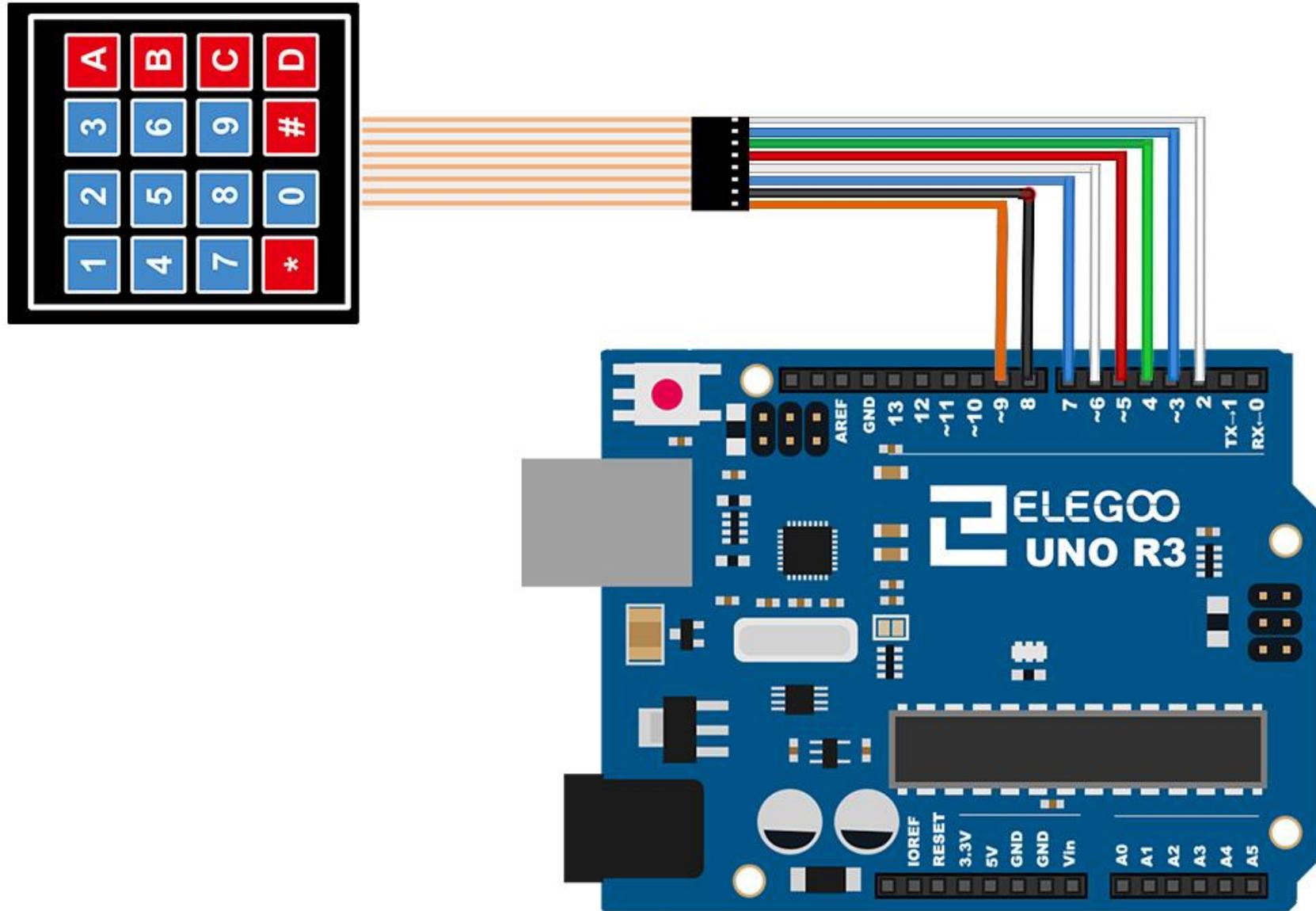
- (1) x Elegoo UNO R3
- (1) x Membrane switch module
- (8) x M-M Kabel (Männlich zu Männlich
DuPont Jumper Kabel)



Verbindungsschema



Schaltplan



Wir schließen die Anschlüsse der Tastatur an die digitalen Ausgänge *D9-D2* des UNO R3 Boards an. Wir verbinden den ersten Pin der Tastatur mit *D9*, den zweiten Pin mit *D8*, den dritten Pin mit *D7*, den vierten Pin mit *D6*, den fünften Pin mit *D5*, den sechsten Pin mit *D4*, den siebten Pin mit *D3* und den achten Pin mit *D2*.

Zum Überblick die Verbindungen noch einmal in einer Tabelle:

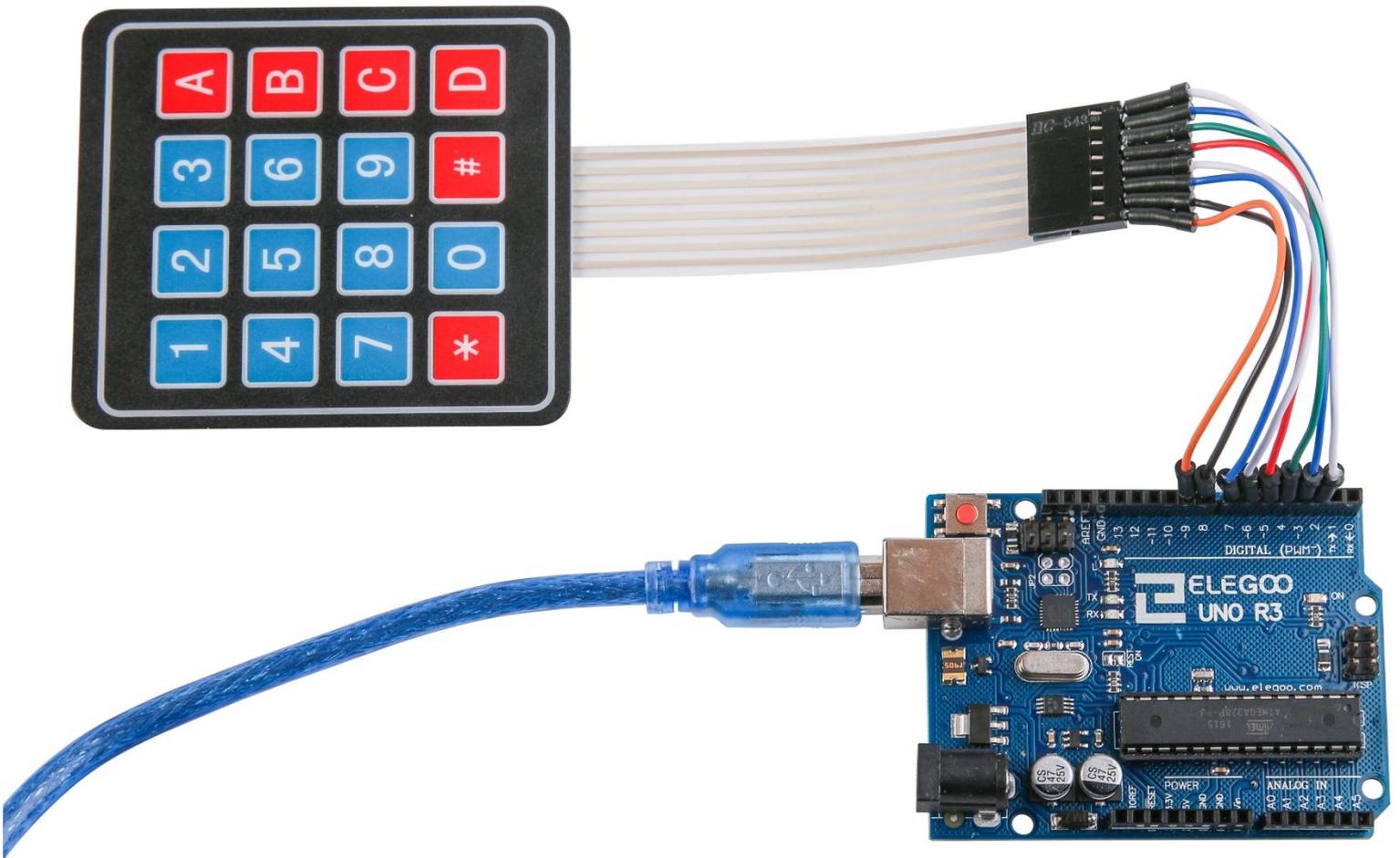
Keypad Pin	Connects to Arduino Pin...
1	D9
2	D8
3	D7
4	D6
5	D5
6	D4
7	D3
8	D2

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 11 Membrane Switch Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

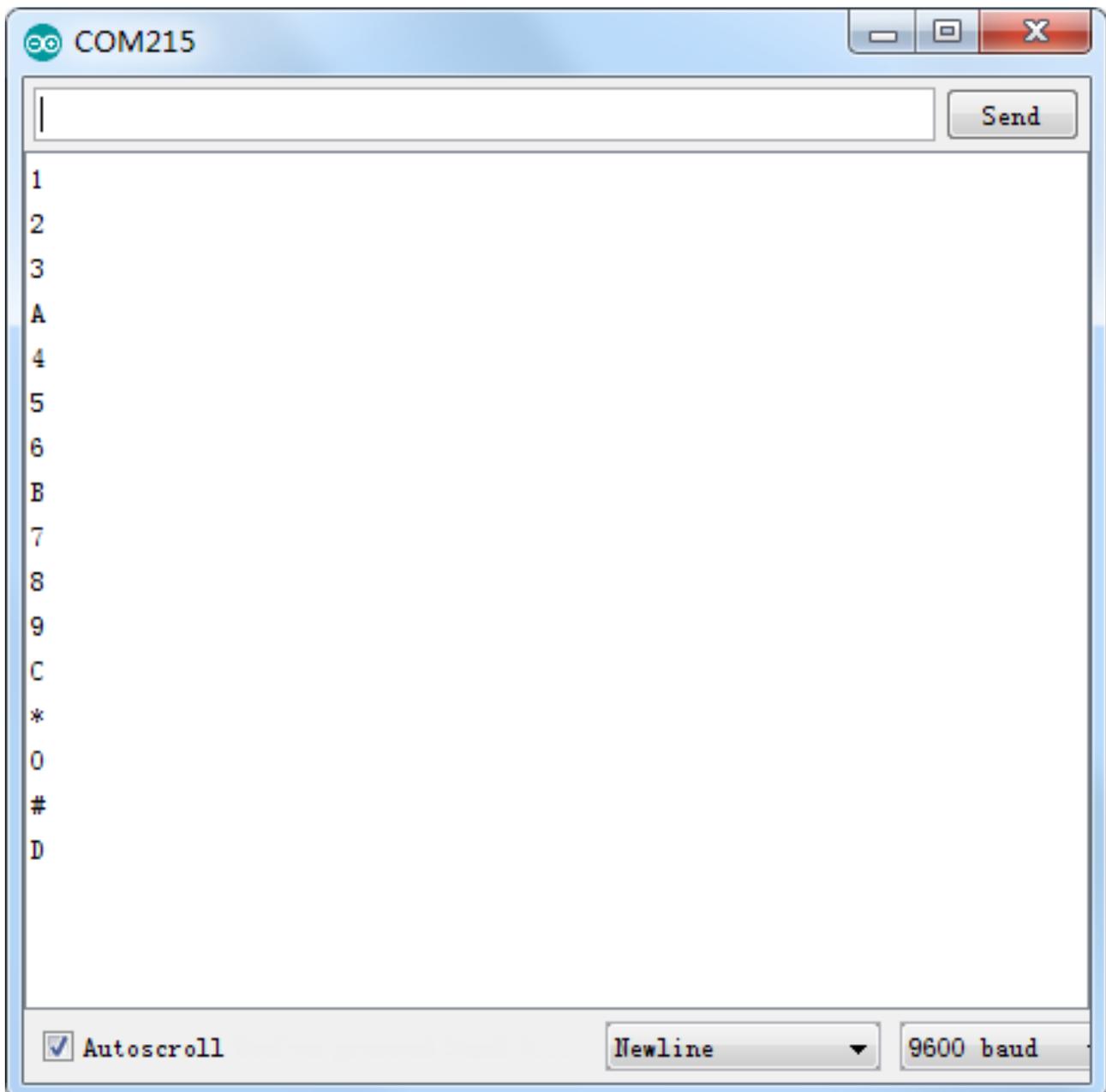
Bevor Sie diesen Sketch hochladen können, müssen Sie die „*Keypad*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Mit diesem Code wird, nachdem der Sketch hochgeladen wurde, bei jedem Knopfdruck die betreffende Taste auf dem Seriellen Monitor der Arduino IDE angezeigt.

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



Lektion12: DHT11 Temperatur- und Luftfeuchtigkeitssensor

Übersicht

In dieser Lektion lernen Sie, wie man einen DHT11 Sensor zum Messen der Temperatur und der Luftfeuchtigkeit benutzt.

Dieser Sensor ist präzise genug für die meisten Projekte, bei denen auf die Temperatur und die Luftfeuchtigkeit geachtet werden muss.

Wir werden wieder eine Bibliothek verwenden, die speziell für diesen Sensor angepasst wurde, sodass unser eigener Code kurz und knapp ausfällt.

Benötigte Bauteile:

(1) x Elegoo UNO R3

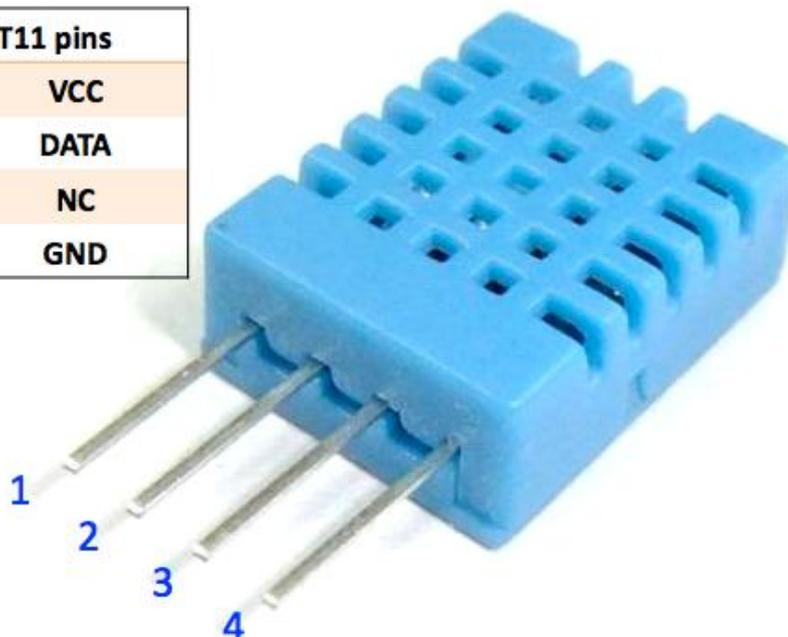
(1) x DHT11 Temperatur- und Luftfeuchtigkeitssensor

(3) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)

Einführung in die Komponenten

Temperatur- und Luftfeuchtigkeitssensor:

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



Der digitale DHT11 Temperatur- und Luftfeuchtigkeitssensor ist ein Verbundsensor, der ein digitales Signal mit den Werten für Temperatur und Luftfeuchtigkeit ausgibt. Die eingebauten digitalen Module sorgen dafür, dass der Sensor

zuverlässige Werte ausgibt und sorgen außerdem für lange Haltbarkeit. Der Sensor enthält ein NTC Temperaturmessgerät und ist an einem 8-bit-Mikrocontroller angeschlossen.

Anwendungsmöglichkeiten:

Heizung, Lüftung, Klimatechnik; Luftentfeuchter; Prüf- und Inspektionsausrüstung; Automobilbau, Wetterstationen, Haushaltsgeräte, Feuchtigkeitsregulatoren und medizinische Geräte.

Produktspezifikationen

Relative Feuchtigkeit

Auflösung: 16Bit

Wiederholbarkeit: $\pm 1\%$ RH

Genauigkeit: Bei $25^{\circ}\text{C} \pm 5\%$ RH

Austauschbarkeit: vollkommen austauschbar

Antwortzeit: 1 / e (63%) von 25°C 6s1m / s Luft 6s

Hysterese: $< \pm 0.3\%$ RH

Langzeitstabilität: $< \pm 0.5\%$ RH / Jahr in

Temperatur

Auflösung: 16Bit

Wiederholbarkeit: $\pm 0.2^{\circ}\text{C}$

Temperaturbereich: Bei $25^{\circ}\text{C} \pm 2^{\circ}\text{C}$

Antwortzeit: 1 / e (63%) 10S

Elektrische Charakteristik

Betriebsspannung: DC 3,5 ~ 5,5V

Betriebsstrom: 0.3mA, standby 60 μA (gemessen)

Abtastabstand: mehr als 2 Sekunden

Pin-Belegung:

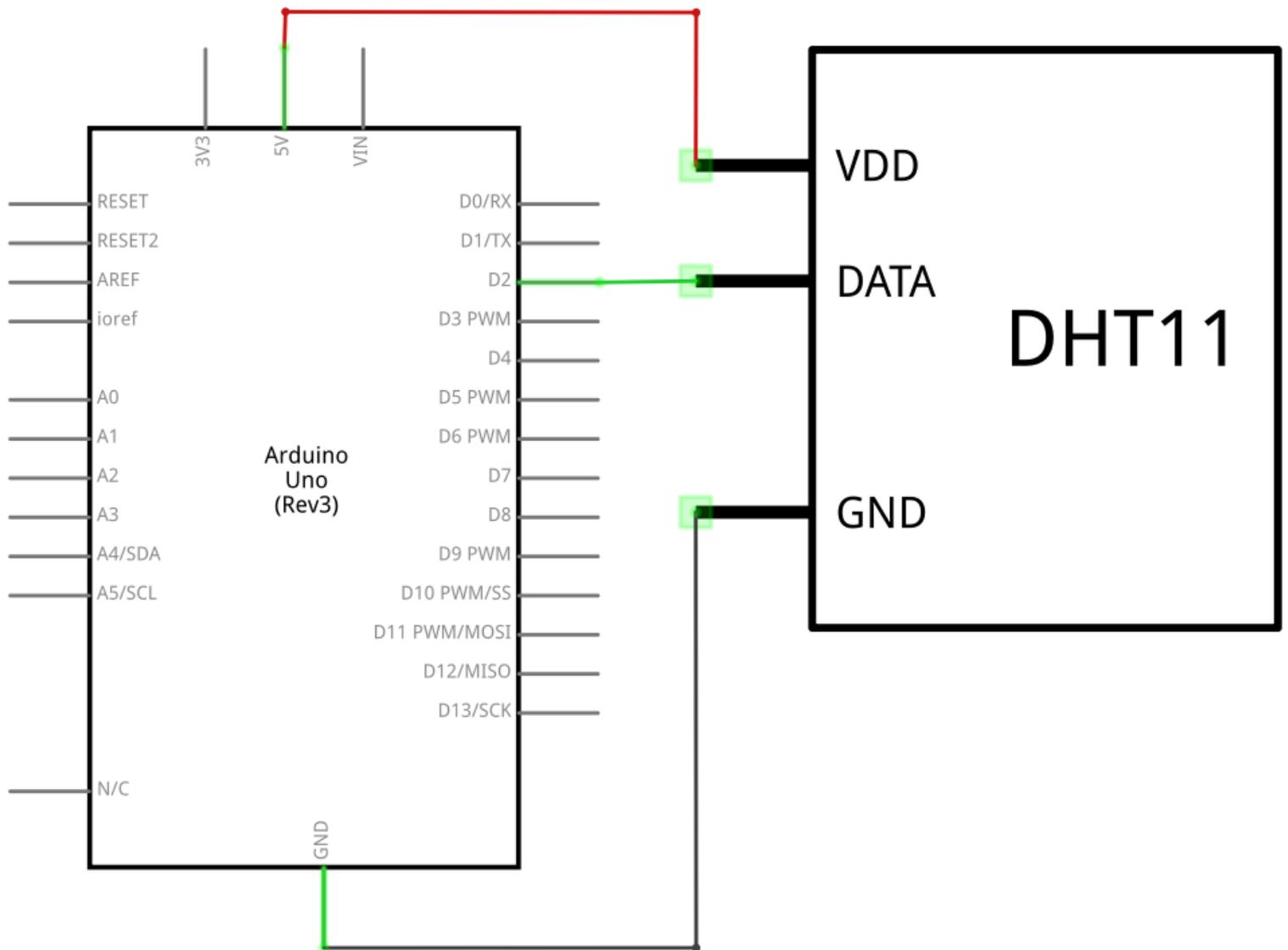
1. VDD Stromanschluss 3,5 ~ 5,5V DC

2. DATA Serieller Datenanschluss

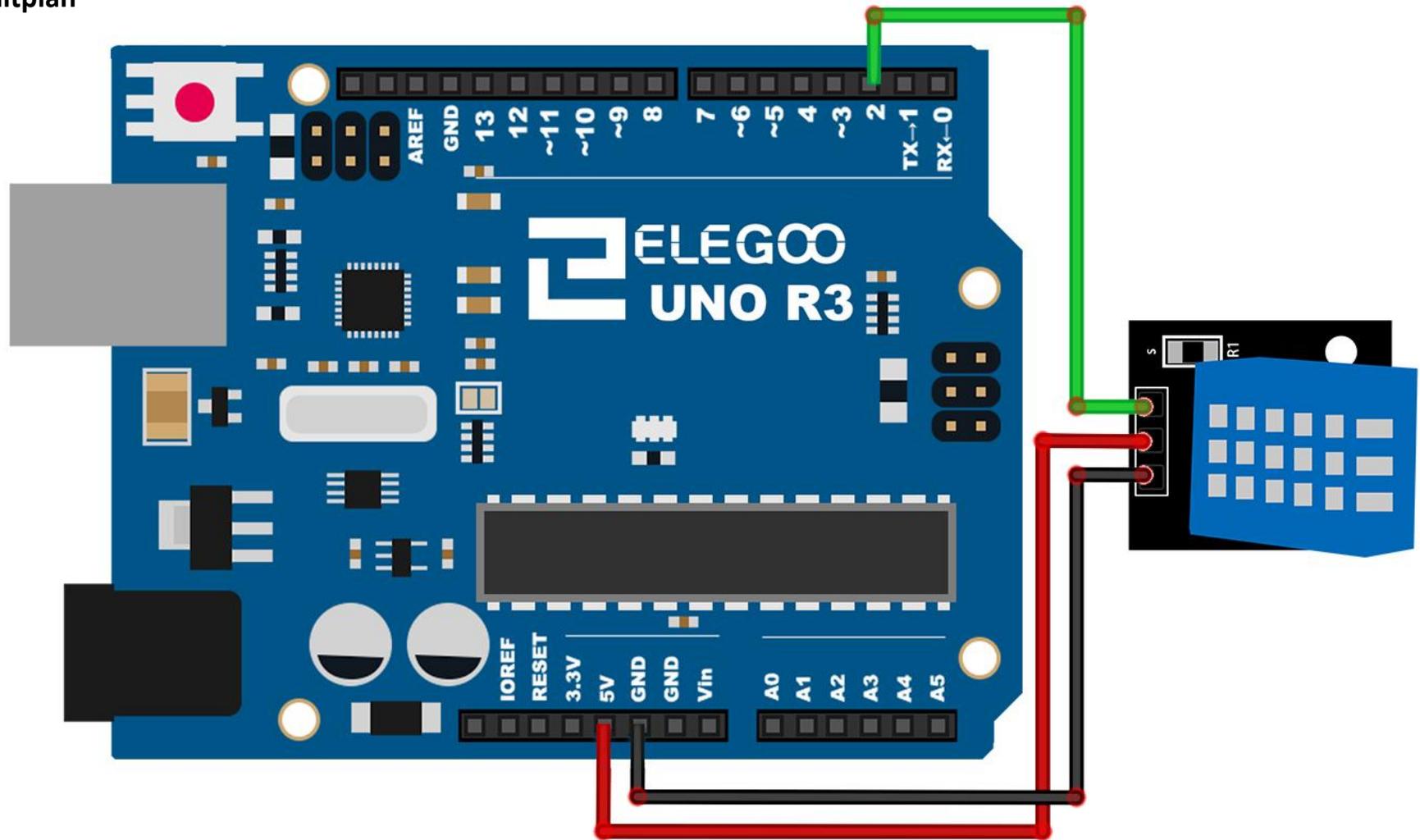
3. NC, unbelegt

4. GND, negativer Anschluss

Verbindungsschema



Schaltplan



Wie Sie sehen können, benötigen wir nur 3 Verbindungen zum Sensor, da ein Pin unbelegt ist.

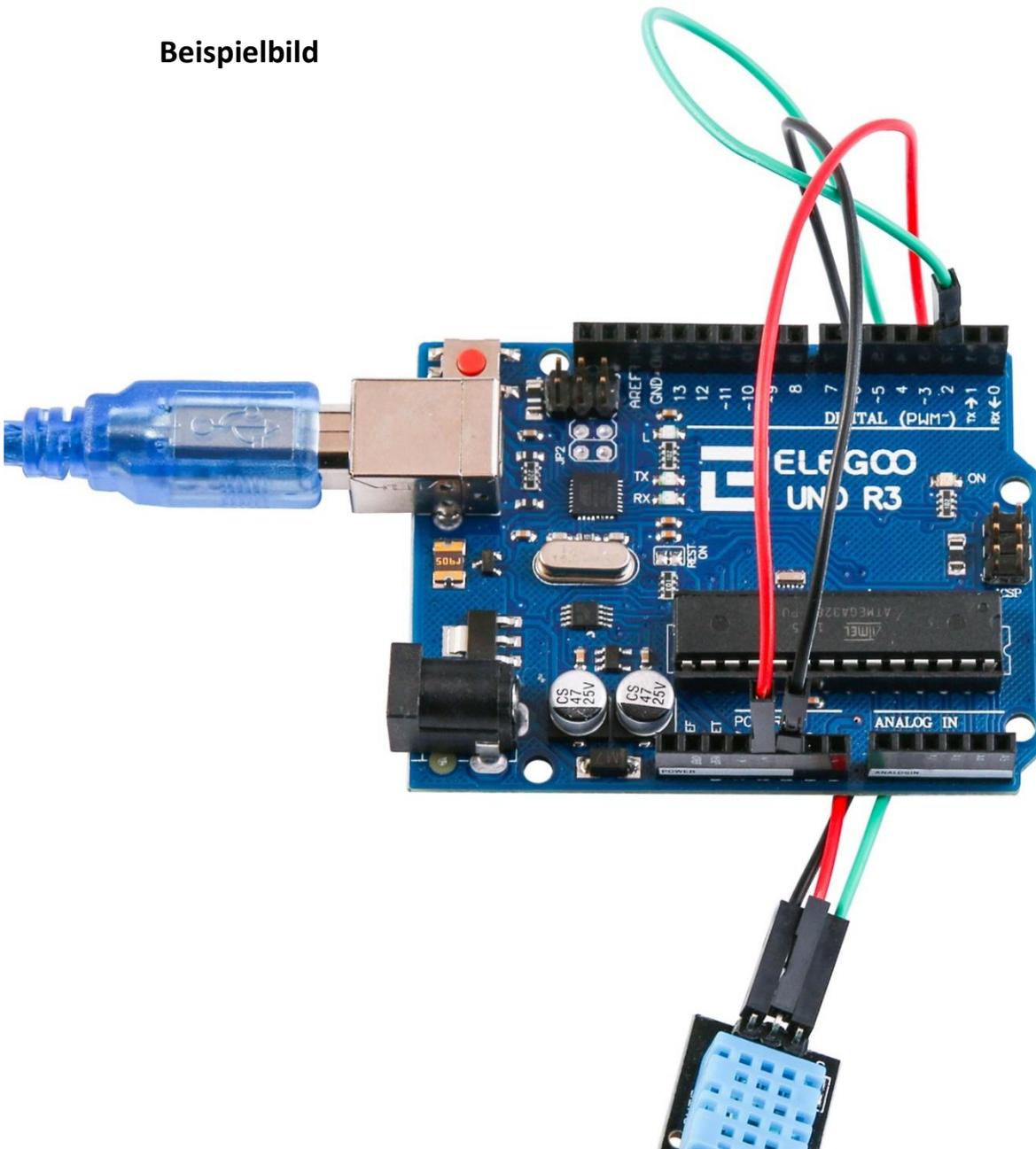
Folgende Pins müssen angeschlossen werden: die Spannungsversorgung, GND und der *Signal*-Anschluss, der an einen beliebigen digitalen Pin des UNO Boards angeschlossen werden kann.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter "Lesson 12 DHT11 Temperature and Humidity Sensor" und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „SimpleDHT“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Laden Sie den Sketch auf Ihr UNO R3 Board hoch und öffnen Sie dann den Seriellen Monitor, sodass Sie ähnliche Werte, wie die unten zu sehenden, angezeigt bekommen. (Es wird die Umgebungstemperatur angezeigt. Im Beispiel sind es 22°C.)

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.

```
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0010 0000 0000 0001 0111 0000 0000 0100 1001
Sample OK: 23 *C, 50 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
=====
Sample DHT11...
Sample RAW Bits: 0011 0011 0000 0000 0001 0110 0000 0000 0100 1001
Sample OK: 22 *C, 51 %
```

Lektion 13: Analoger Joystick

Übersicht

Analoge Joysticks eignen sich gut, um Ihre Projekte präzise steuern zu können. In dieser Lektion lernen Sie, wie man einen analogen Joystick benutzt.

Benötigte Bauteile:

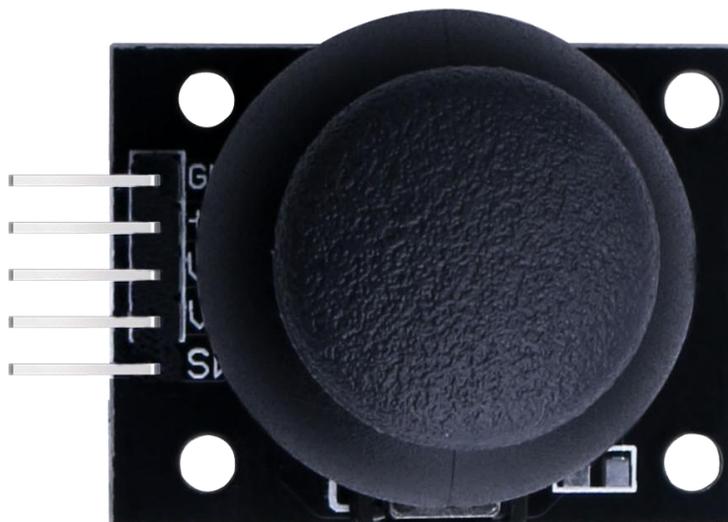
- (1) x Elegoo UNO R3
- (1) x Joystick Modul
- (5) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)

Einführung in die Komponenten

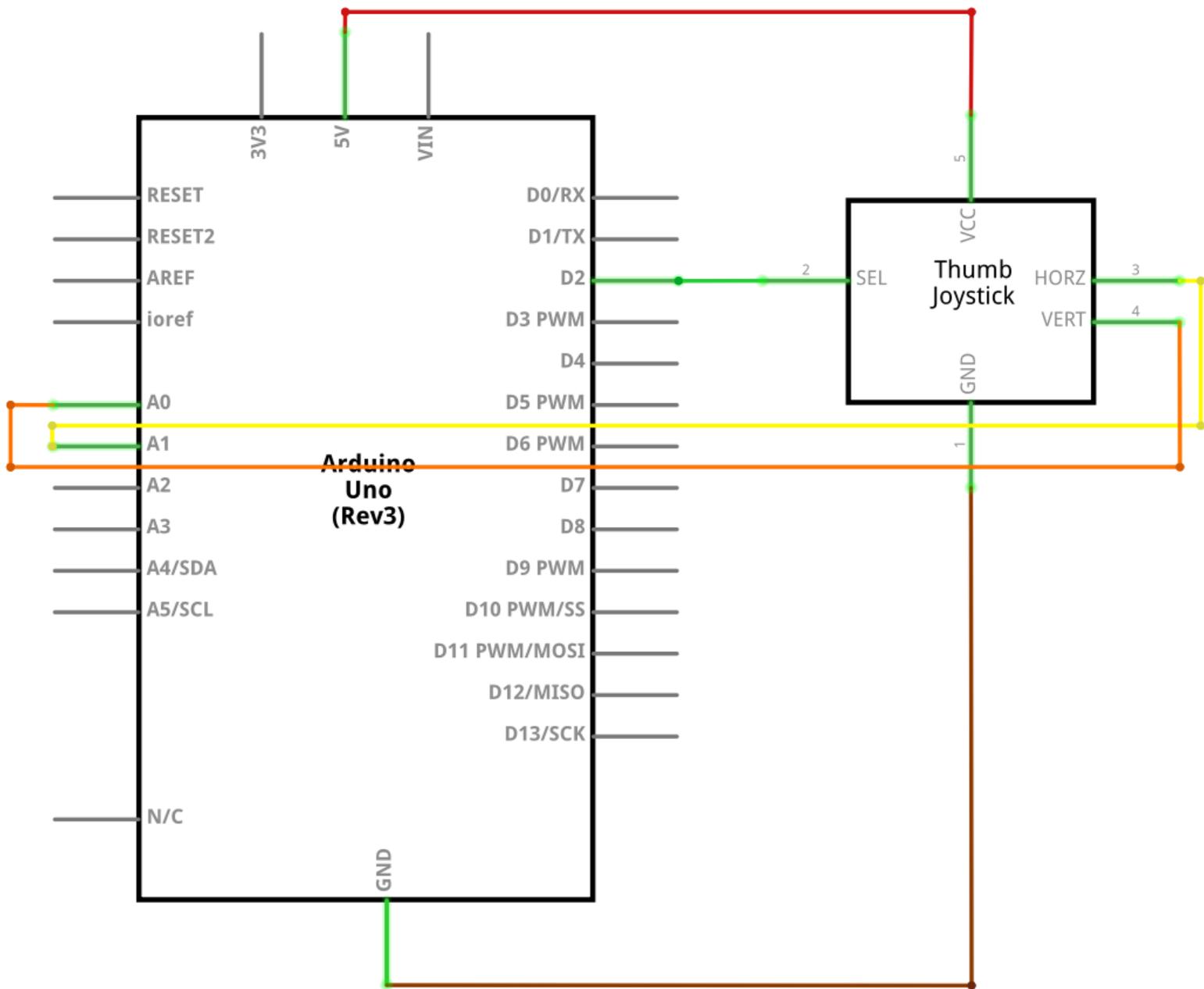
Joystick

Das Modul hat 5 Pins: *VCC* (Stromversorgung), *GND*, *X*, *Y* und *Key*. Beachten Sie, dass die Bezeichnungen bei Ihrem Modell leicht abweichen können, je nachdem wo das Modul gekauft wurde. Der Joystick ist analog und erlaubt eine viel präzisere Steuerung als bei einfachen mechanischen Schaltern. Zusätzlich zur horizontalen und vertikalen Steuerung, können Sie den Joystick herunterdrücken, wobei der Joystick als Schalter fungiert.

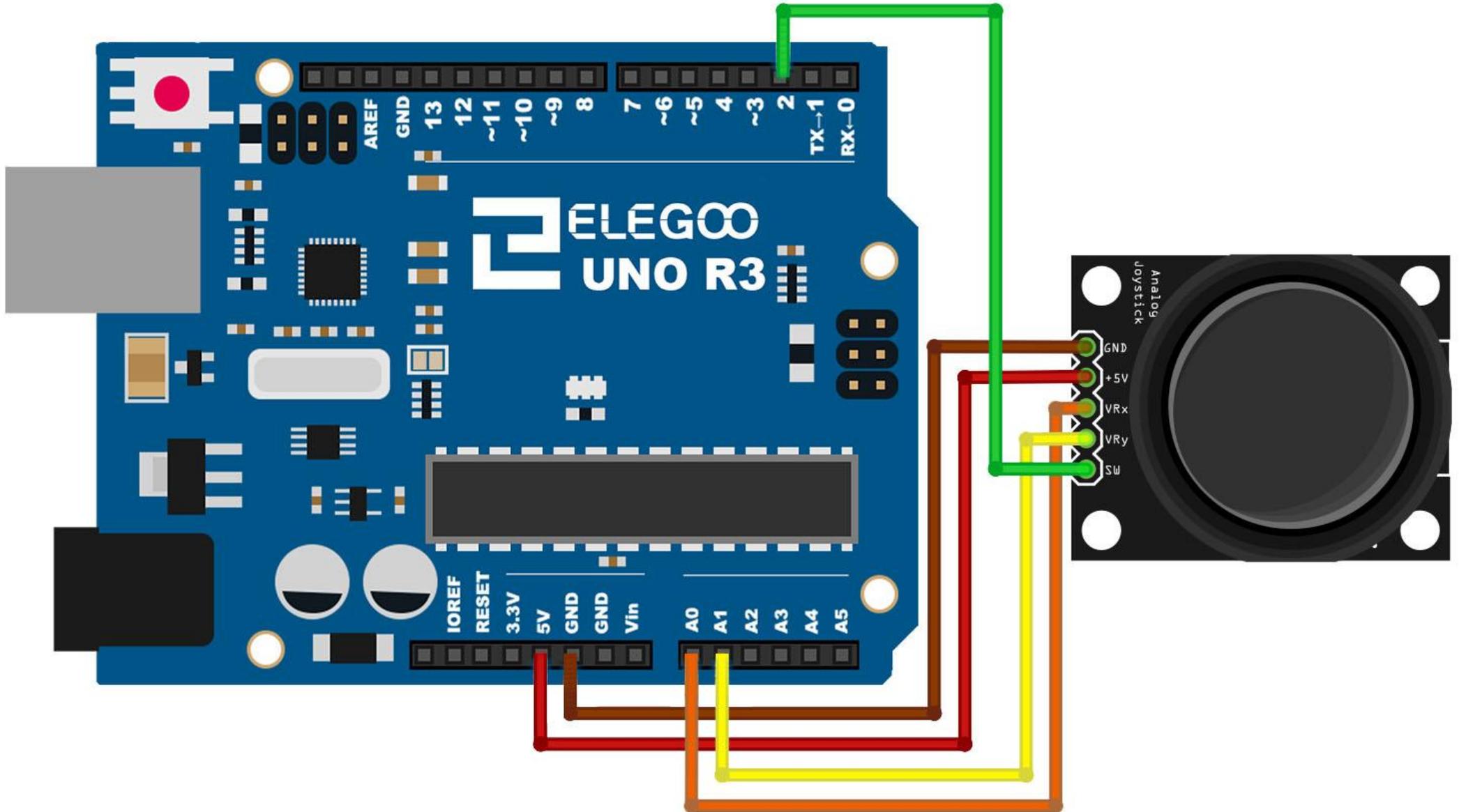
Wir müssen die analogen Pins des UNO Boards nutzen, um die analogen Werte des Joysticks auf der X- und Y-Achse empfangen zu können und einen digitalen Pin für den Schalter. Der *Key*-Pin ist im normalen Zustand frei und wird bei Drücken des Joysticks mit *GND* verbunden. Um zuverlässige Werte zu erhalten, muss der *Key*-Pin mittels eines Pullup-Widerstandes mit dem Board verbunden werden. Dabei können wir die eingebauten Pullup-Widerstände des UNO Boards benutzen. Für eine Anleitung, wie man Pins des UNO R3 Boards als Eingang mit Pullup-Widerstand benutzt, schauen Sie sich den Code-Teil von Lektion 5 erneut an.



Verbindungsschema



Schaltplan



Wir brauchen fünf Verbindungen zum Joystick:

Key, Y, X, VCC und GND.

X und Y sind analog und Key ist digital. Wenn Sie den Schalter nicht benutzen, können Sie den Key-Pin weglassen und benötigen nur die restlichen vier Anschlüsse.

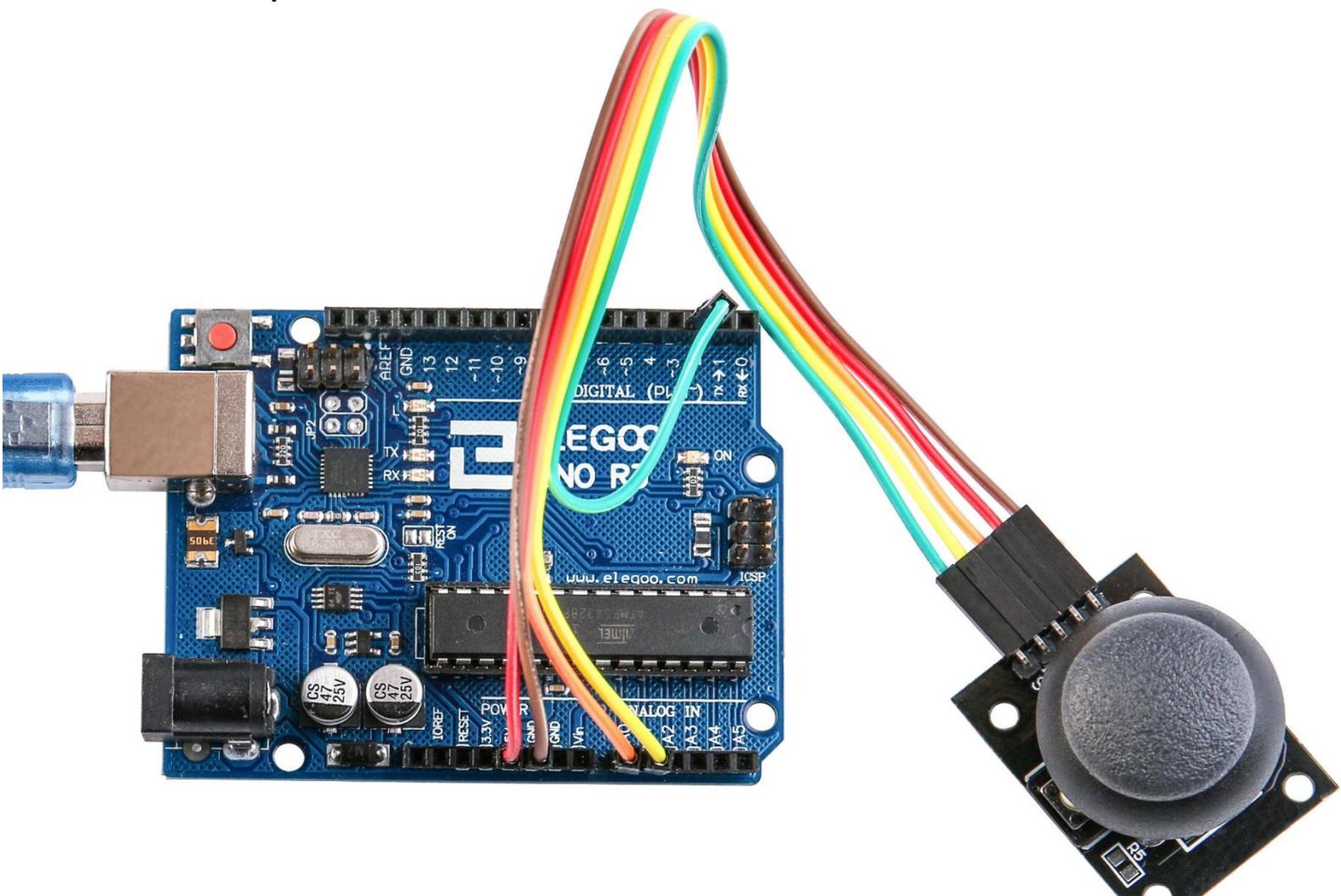
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 13 Analog Joystick Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Analoge Joystick sind im Grunde Potentiometer, daher geben sie analoge Werte zurück. Wenn sich der Stick in der Mitte befindet (Standardposition), sollte er am X- und am Y-Pin einen Wert von 512 ausgeben. Der Wertebereich geht von 0 – 1024.

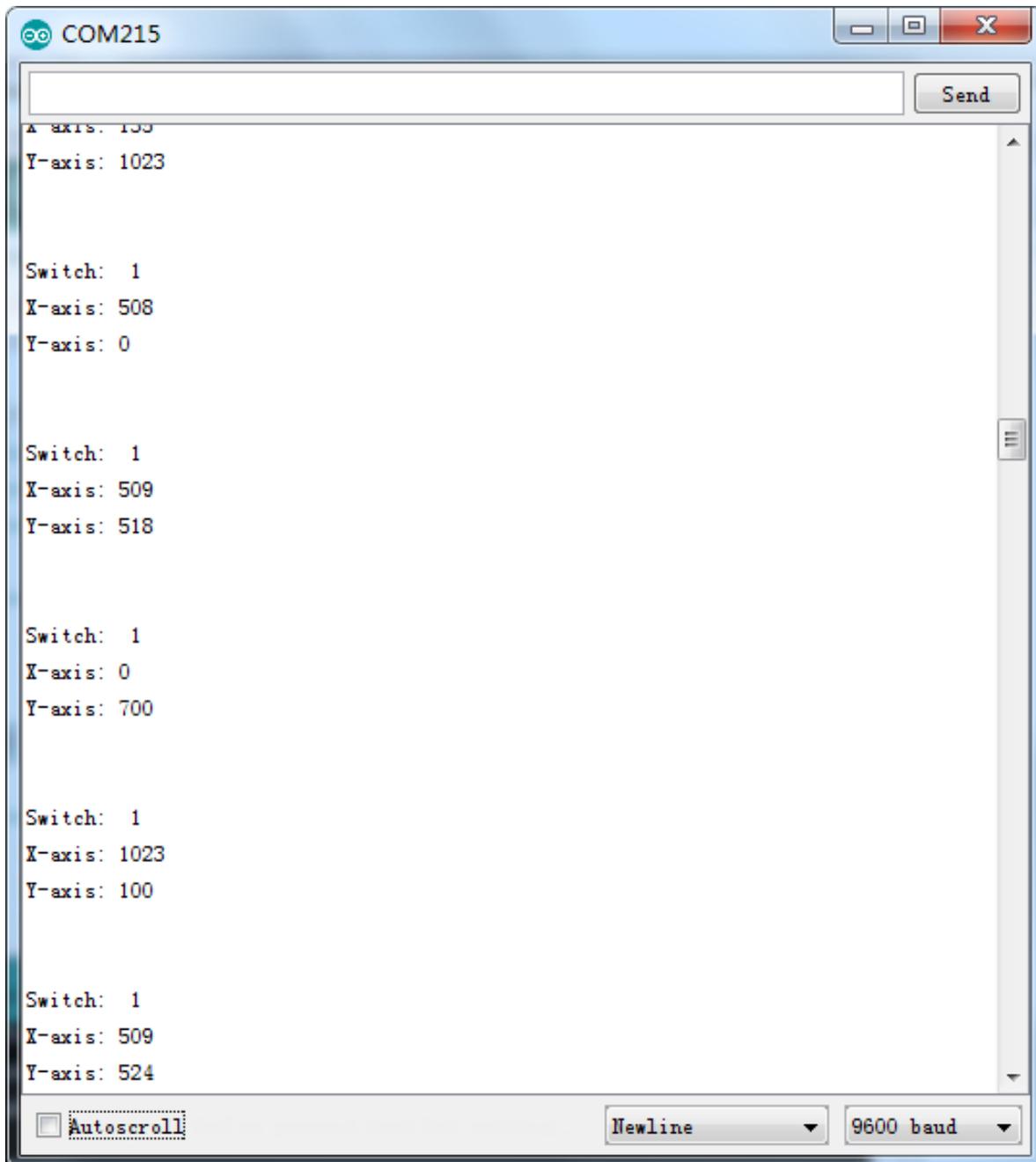
Auf der X-Achse ist ganz links = 0 und ganz rechts = 1024, auf der Y-Achse ist ganz unten = 0 und ganz oben = 1024.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



Lektion 14: Infrarotempfängermodul

Übersicht

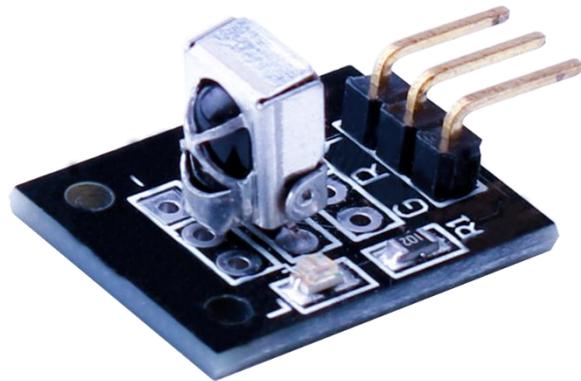
Durch eine *IR* (Infrarot) Fernbedienung lässt sich Ihr Projekt auf einfache Weise kabellos steuern.

IR Fernbedienungen sind einfach aufgebaut. In dieser Lektion werden wir einen Infrarotempfänger mit dem UNO Board verbinden und ihn dann mittels einer geeigneten Bibliothek einbinden.

In unserem Sketch sind bereits alle hexadezimalen IR Codes für alle Tasten der Fernbedienung eingetragen. Wir werden im Sketch überprüfen, welche Taste gedrückt wurde und ob sie im Moment gedrückt wird.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x IR Empfängermodul
- (1) x IR Fernbedienung
- (3) x W-MKabel (Weiblich zu Männlich
DuPontJumper Kabel)



Einführung in die Komponenten

IR Empfänger (Sensor):

Infrarotempfänger sind kleine Mikrochips mit einer Fotozelle, die darauf getrimmt ist Infrarotsignale zu erkennen. Die Infrarottechnik wird in zahlreichen Geräten eingesetzt: jeder TV und jeder DVD Player hat einen Infrarotempfänger auf der Vorderseite, der auf das Infrarotsignal der Fernbedienung reagiert. In der dazugehörigen Fernbedienung befindet sich eine passende IR LED, die verschiedene Infrarotimpulse ausgibt, je nachdem welche Taste gedrückt wurde und ob der TV an oder ausgeschaltet oder einfach nur der Kanal gewechselt werden soll. Infrarotlicht ist unsichtbar für das menschliche Auge, daher sehen wir die IR LED nicht leuchten.

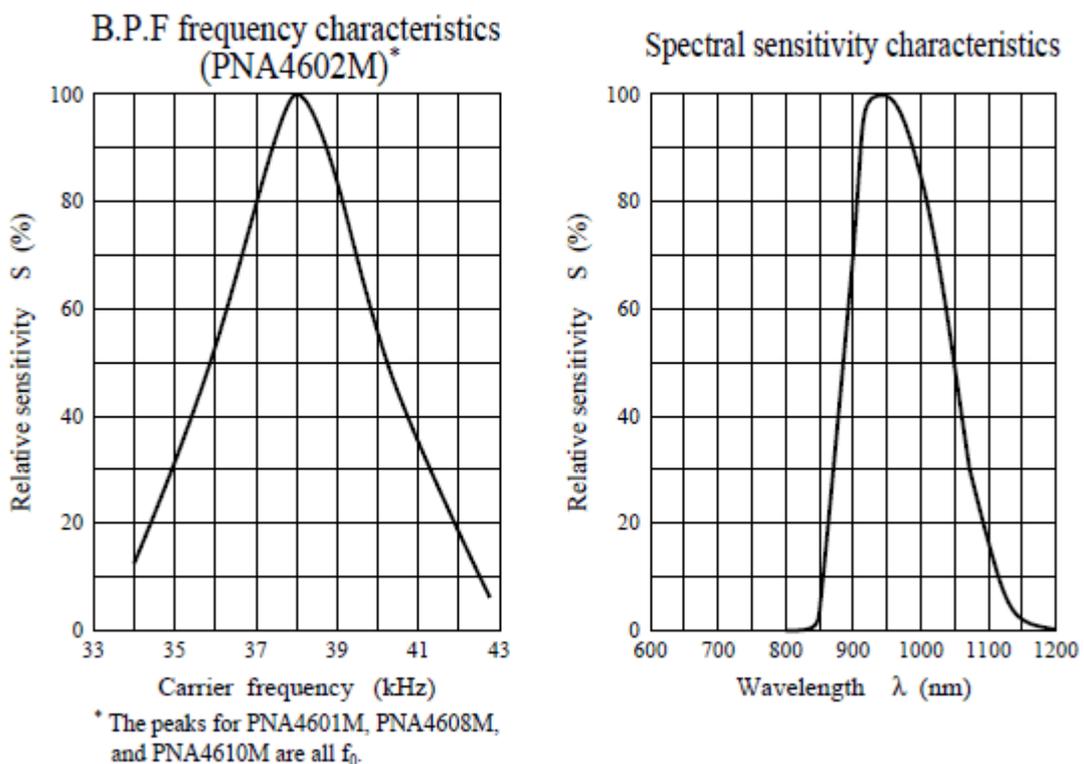
IR Empfänger unterscheiden sich in einigen Punkten zu normalen, sogenannten „CdS“-Fotozellen. IR Empfänger filtern lediglich Infrarotlicht heraus und können kein sichtbares, normales Licht registrieren. Auf der anderen Seite sind CdS-Fotozellen gut dafür geeignet gelb/grünes sichtbares Licht zu erkennen,

funktionieren aber nicht bei Infrarotlicht.

Infrarotempfänger haben einen Demodulator eingebaut, der auf ein moduliertes Infrarotsignal bei 38kHz reagiert. Eine einfach blinkende Infrarot-LED würde nichts bewirken, das Licht muss PWM-moduliert sein und eine Frequenz von 38kHz haben. Fotozellen haben dagegen gar keinen eingebauten Demodulator und können jede Frequenz feststellen, die in ihrer Frequenzzeit liegt (etwa 1kHz).

Infrarotempfänger geben ein digitales Signal aus. Wenn sie ein 38kHz IR-Signal empfangen, geben sie ein LOW-Signal (0V) aus, während sie wenn sie nichts empfangen HIGH (5V) ausgeben. Fotozellen dagegen funktionieren wie Widerstände. Ihr Widerstand ändert sich abhängig davon, wie viel Licht sie empfangen.

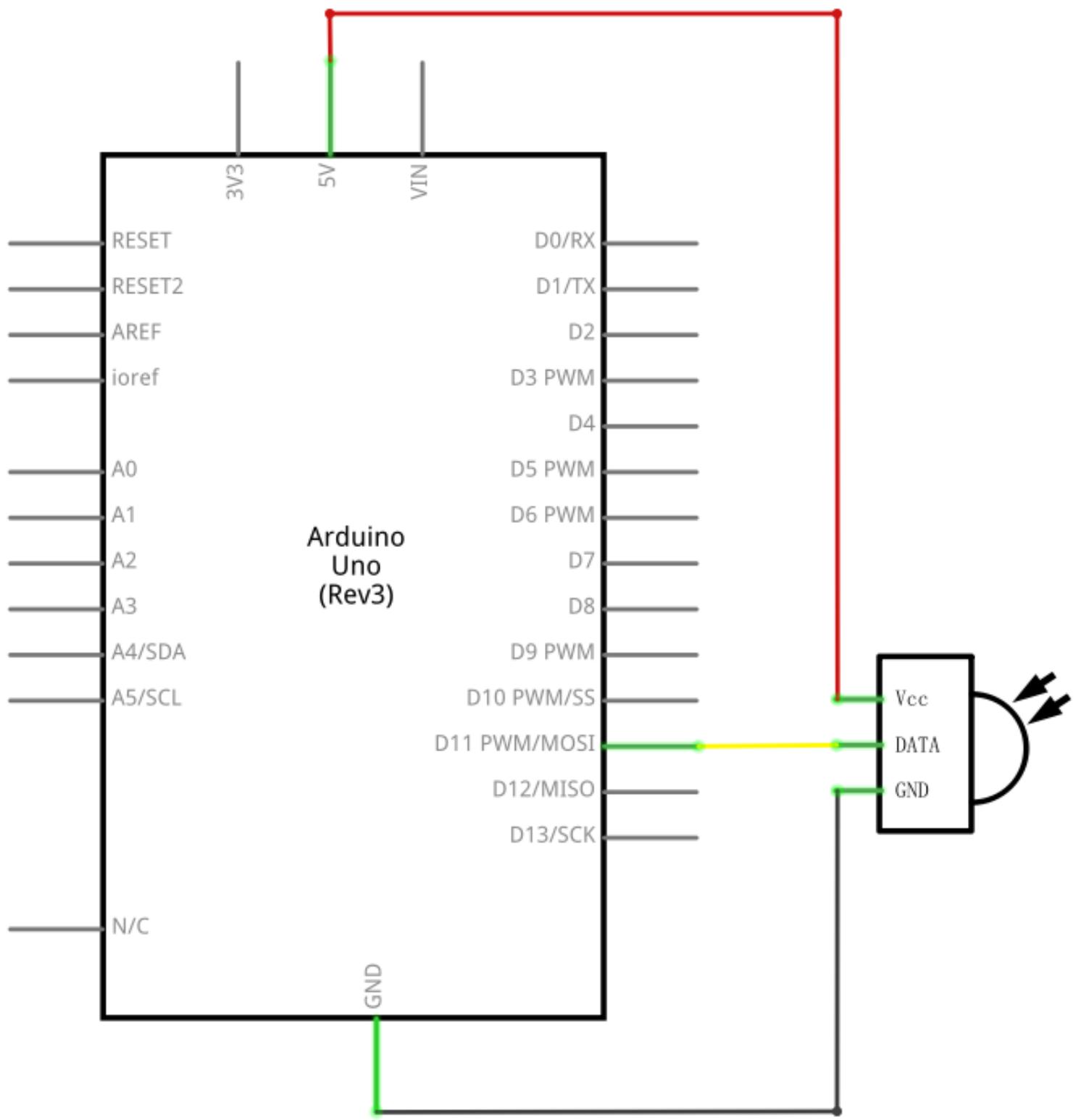
Messgrößen



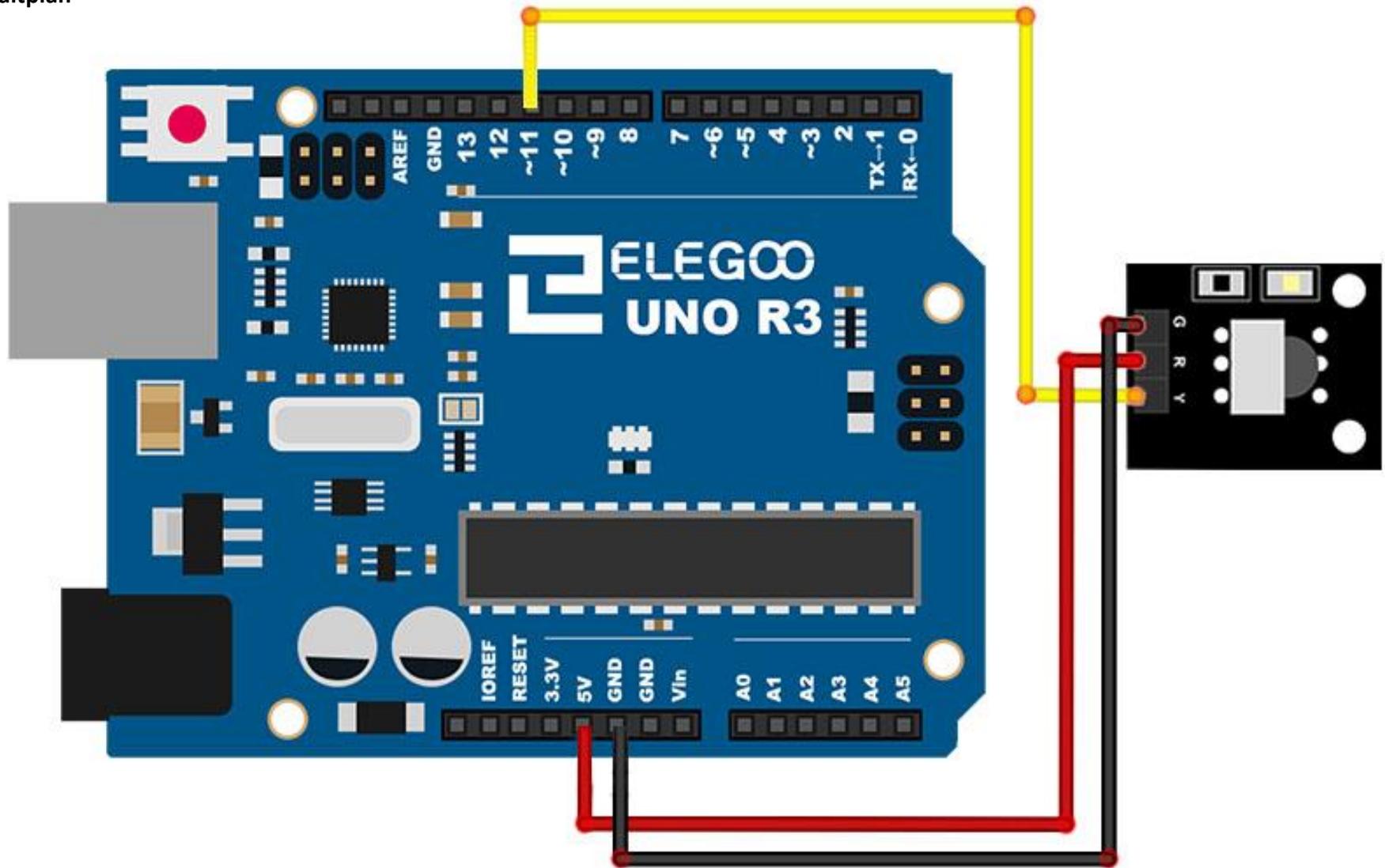
Wie Sie in den Graphen erkennen können, liegt die Spitzenfrequenz bei 38kHz und die optimale Wellenlänge des Lichtes beträgt 940nm. Es können zwar alle Frequenzen im Bereich von 35kHz bis 41kHz und alle Wellenlänge von 850 bis 1100nm benutzt werden, aber die Empfangsgenauigkeit nimmt ab und die Signale würden nicht so gut erkannt werden. Daher gehen Sie bitte sicher, passende LEDs zum senden zu benutzen. Ob Ihre LED die richtige Wellenlänge hat, können Sie im

Datenblatt der LED überprüfen. Der optimale Wert wäre 940nm (nicht sichtbar).

Verbindungsschema



Schaltplan



Es gibt drei Verbindungen zum IR-Empfänger.

Die Anschlüsse sind: *Signal*, *Spannung* und *GND*.

Das “-” steht für GND, “S” ist das Signal, und der mittlere Pin ist die Spannung (5V).

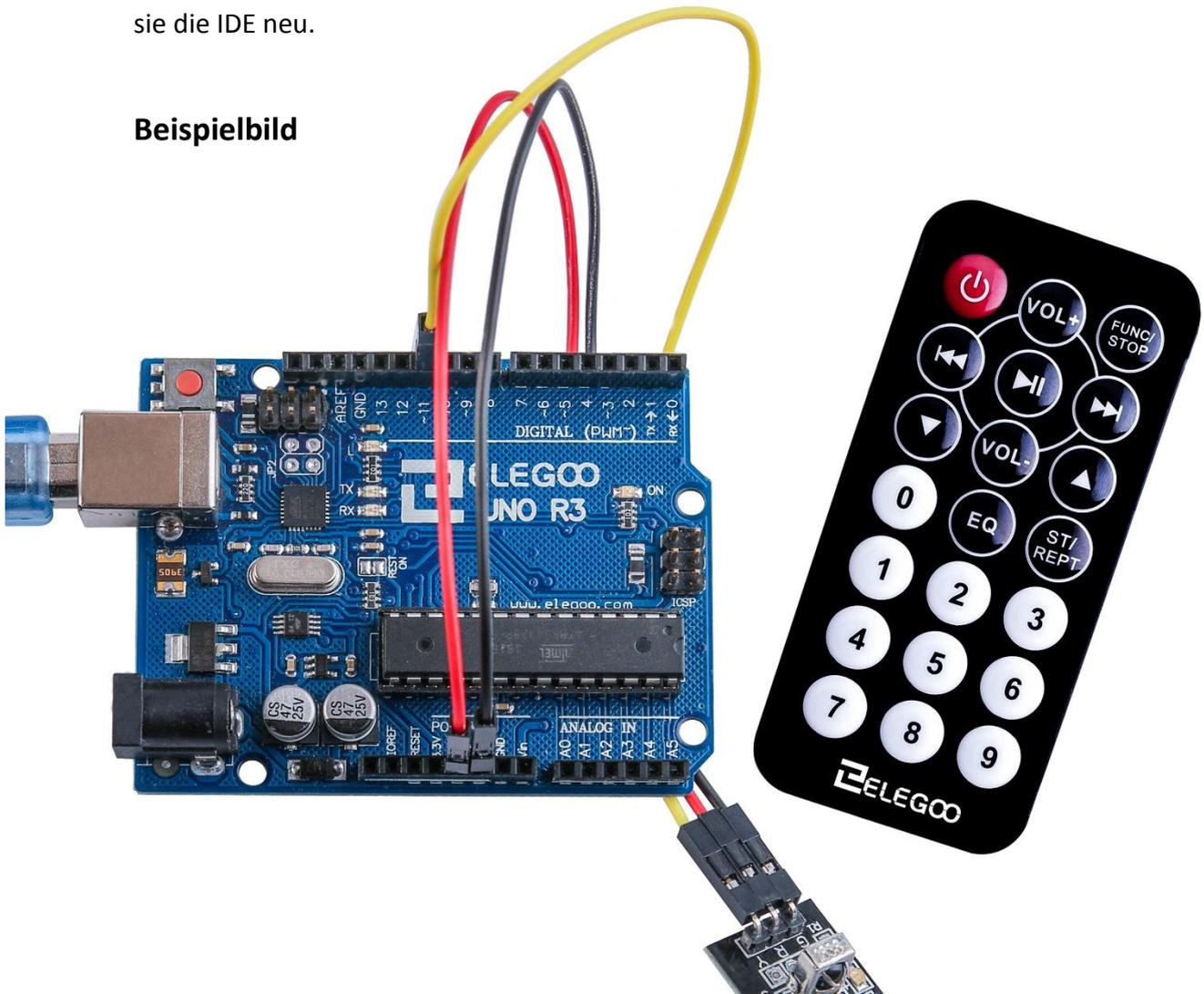
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 14 IR Receiver Module*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „*IRremote*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

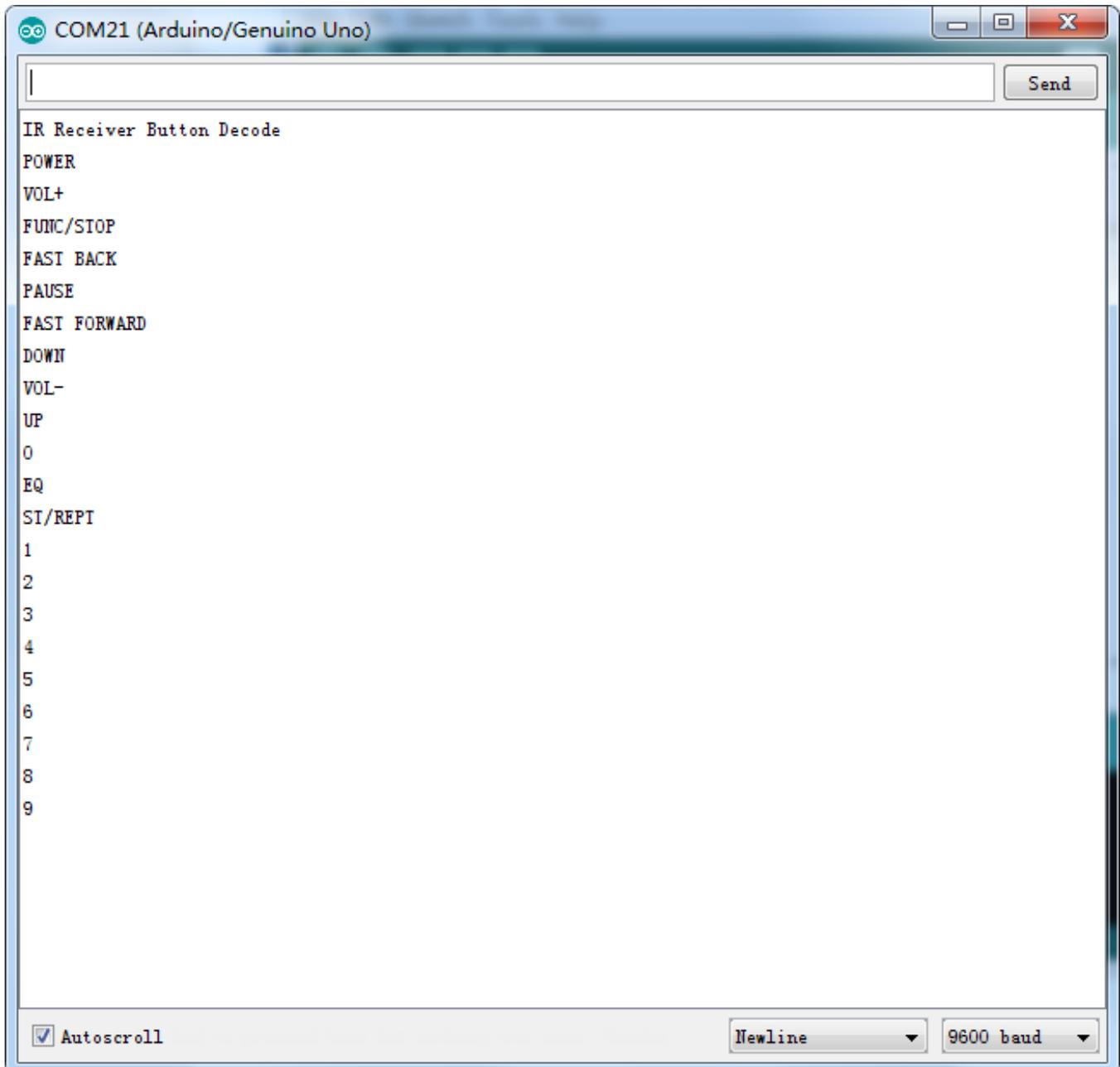
Als nächstes verschieben wir zeitweise den Ordner „*RobotIRremote*“ aus dem Bibliotheksordner der IRremote-Bibliothek, da wir zum Programmieren eine andere Version dessen benutzen werden, was sonst zu Konflikten führen würde. Sie können den Ordner nach dem Hochladen wieder zurück in den Bibliotheksordner verschieben. Wenn Sie die Bibliothek installiert haben, starten sie die IDE neu.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



Lektion 15: MAX7219 LED-Punktmatrixmodul

Übersicht

In dieser Lektion werden wir ein MAX7219-Modul mit unserem UNO Board verbinden und Text auf diesem anzeigen lassen.

Da diese Module den MAX7219 LED-Treiber benutzen, können wir jede einzelne der 64 LEDs getrennt steuern, während wir nur 3 Verbindungen zum Board brauchen.

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x MAX7219 Module

(5) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)

Einführung in die Komponenten

MAX7219 LED-Punktmatrixmodul

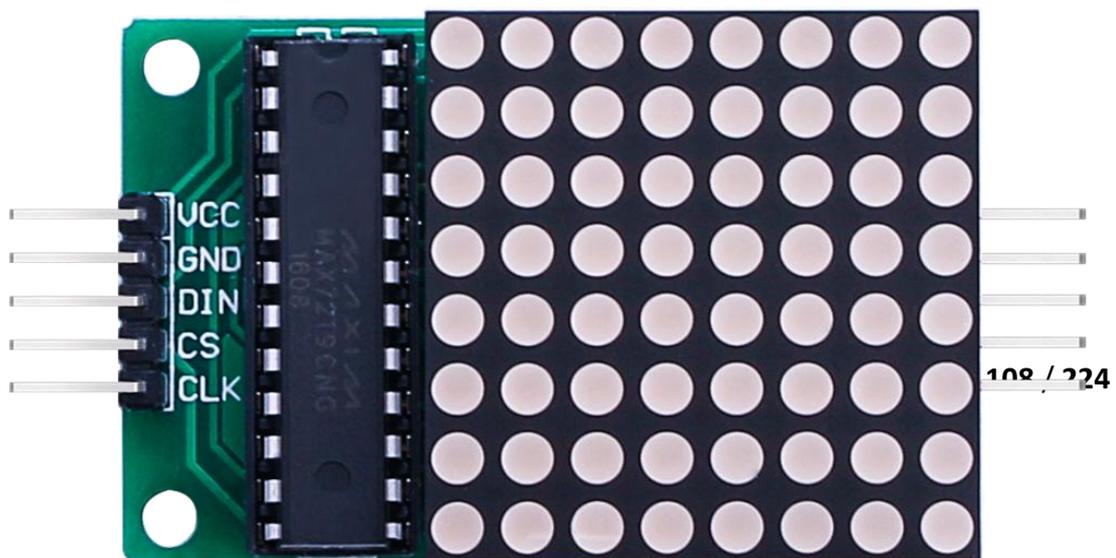
Unser Projekt besteht aus unserem Arduino Board mit einem seriell angeschlossenen MAX7219-Chip, der eine 8x8 LED-Matrix kontrolliert. Der MAX7219 ist ein serieller Displaytreiber mit gemeinsamer Kathode, der eine bis zu 8-Segment große Ziffernanzeige, ein Balkendiagramm oder einfach 64 individuelle LEDs steuern kann. Der Einfachheit halber benutzen wir eine 8x8 LED-Matrix mit einem integrierten MAX7219 Chip, der bereits vollständig angeschlossen ist. Die typischen technischen Spezifikationen lauten:

Betriebsspannung: DC 4,7V – 5,3V

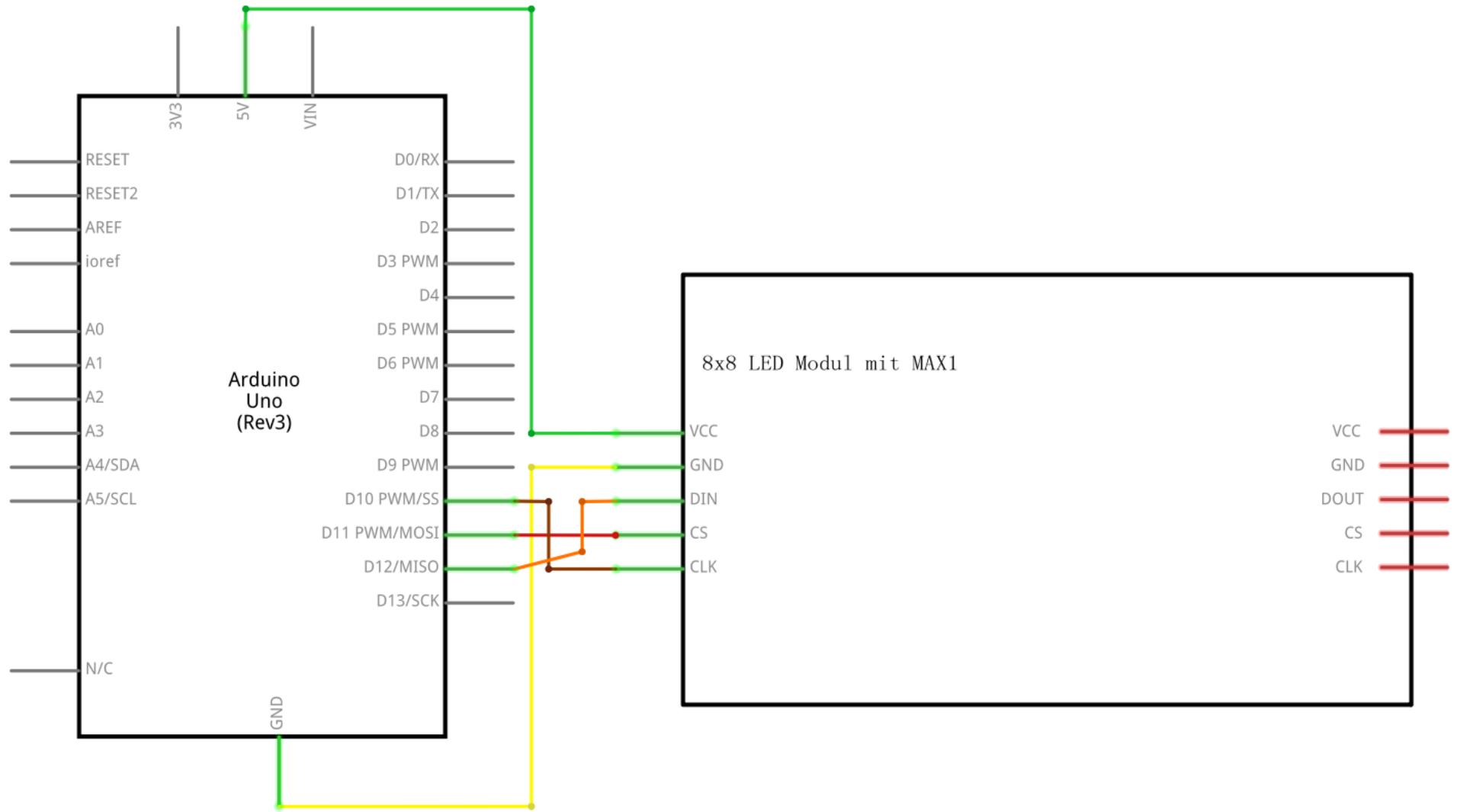
Typische Spannung: 5V

Betriebsstrom: 320mA

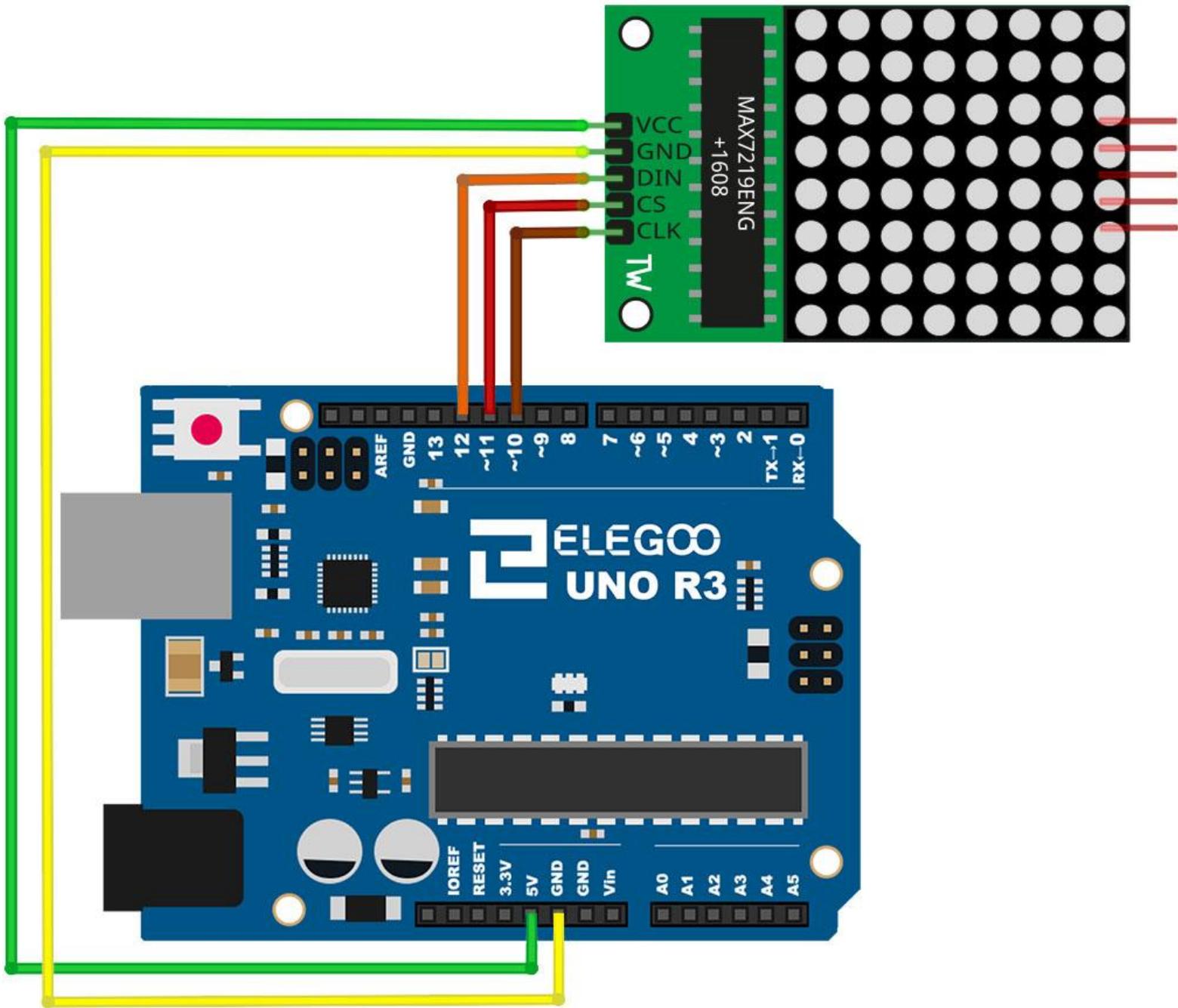
Maximaler Stromfluss: 2A



Verbindungsschema



Schaltplan



VCC und GND werden mit dem Arduino verbunden.

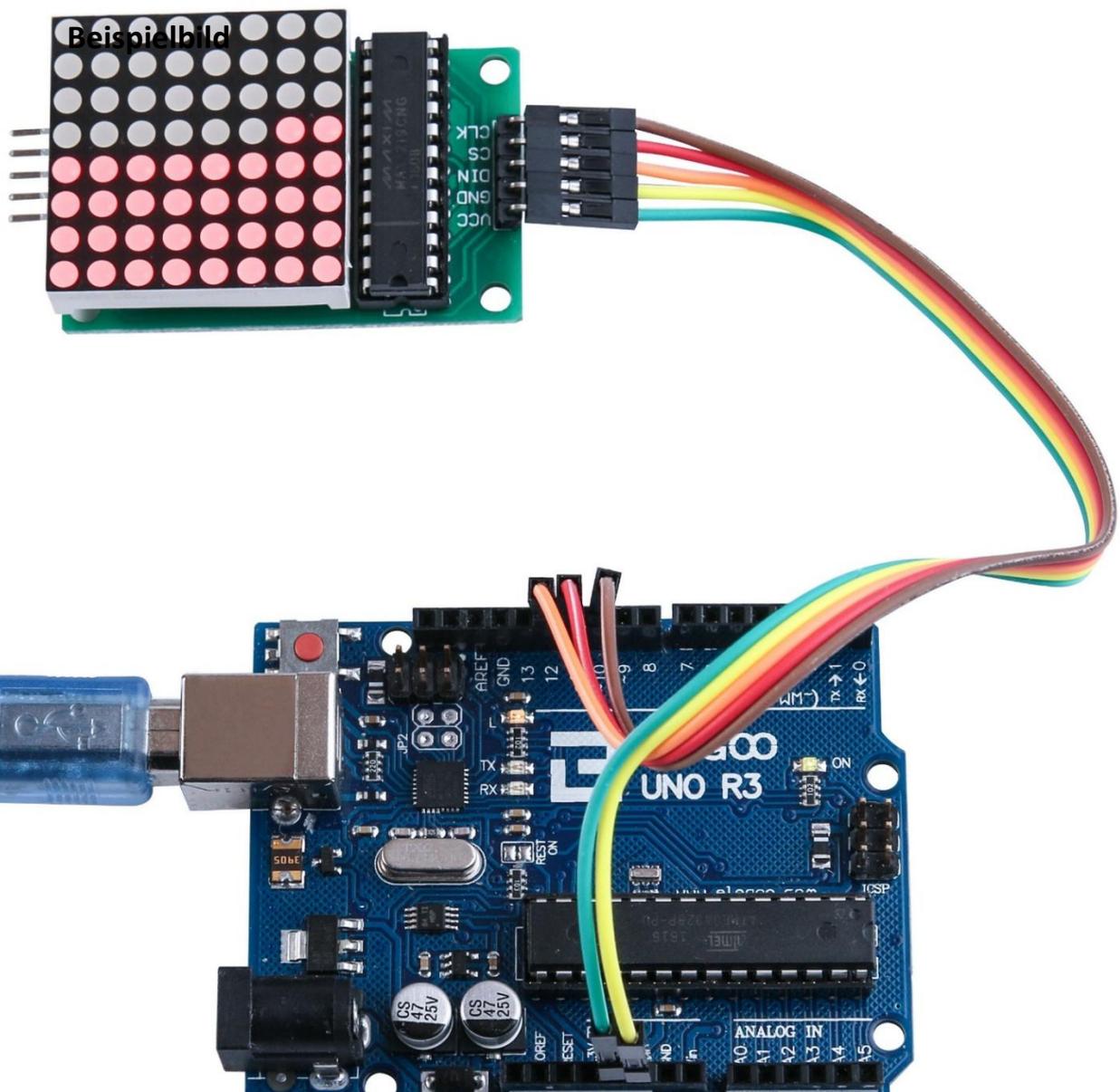
Pin 12 wird mit *DIN* und Pin 11 mit *CS* verbunden. *CLK* wird an Pin 10 angeschlossen.

Code

Unser Sketch benutzt die „*Maxmatrix*“-Bibliothek, um mit dem MAX7219 Modul zu kommunizieren.

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 15 MAX7219 LED Dot Matrix Module*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „*LedControl*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.



Lektion 16:MPU-6050 Modul

Übersicht

In dieser Lektion lernen Sie, wie man ein MPU6050 Modul, eines der besten Trägheitsmessgeräte, das für Arduino verfügbar ist, benutzt. *IMU* (Trägheits-)Sensoren wie der MPU 6050 werden in selbstbalancierenden Robotern, Drohnen, Smartphones und vielen weiteren Geräten benutzt.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x MPU6050 Modul
- (4) x W-M Kabel

Einführung in die Komponenten

MPU6050 Sensor

Der *InvenSense MPU-6050* Sensor beinhaltet einen *MEMS* Beschleunigungssensor und ein *MEMS* Gyroskop in einem einzigen Chip. Der Sensor ist sehr genau, da er eine 16-bit analog zu digital Umwandlung je Kanal durchführt. Daher kann er die X-, Y und Z-Werte zeitgleich messen und ausgeben. Der Sensor benutzt eine I2C-Schnittstelle für die Verbindung zum Arduino. Der MPU-6050 ist nicht teuer, gerade wenn man bedenkt, dass er einen Beschleunigungssensor und ein Gyroskop enthält.

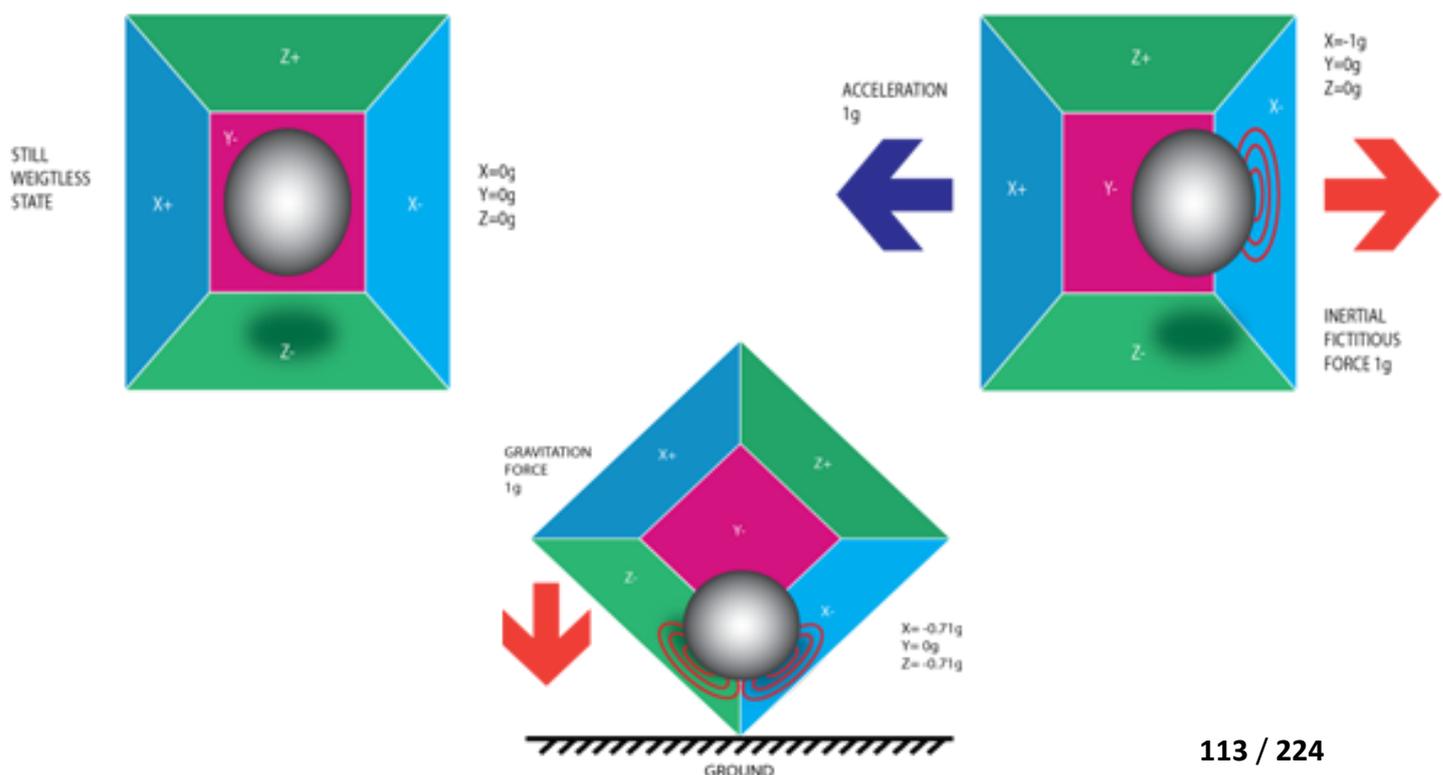


IMU-Sensoren sind eine Sensorart, die für uns heutzutage unverzichtbar ist, da sie in jeglichen elektronischen Geräten verbaut werden. Sie werden in Smartphones benutzt, Wearables, Spielecontrollern und vielen weiteren Geräten. IMU Sensoren ermöglichen es uns, an einem Objekt befestigt, die Ausrichtung des Objektes im 3-dimensionalen Raum zu bestimmen. Diese Werte werden normalerweise in Winkeln gemessen, sodass wir die Ausrichtung und Haltung berechnen können. Daher werden sie zum Beispiel in Smartphones eingesetzt, um die Ausrichtung festzustellen und wenn nötig das Display mitzudrehen. Auch in Wearables, wie dem Nike Fuel Band oder dem Fitbit-Armband werden IMU-Sensoren zum Erkennen von Bewegungen benutzt.

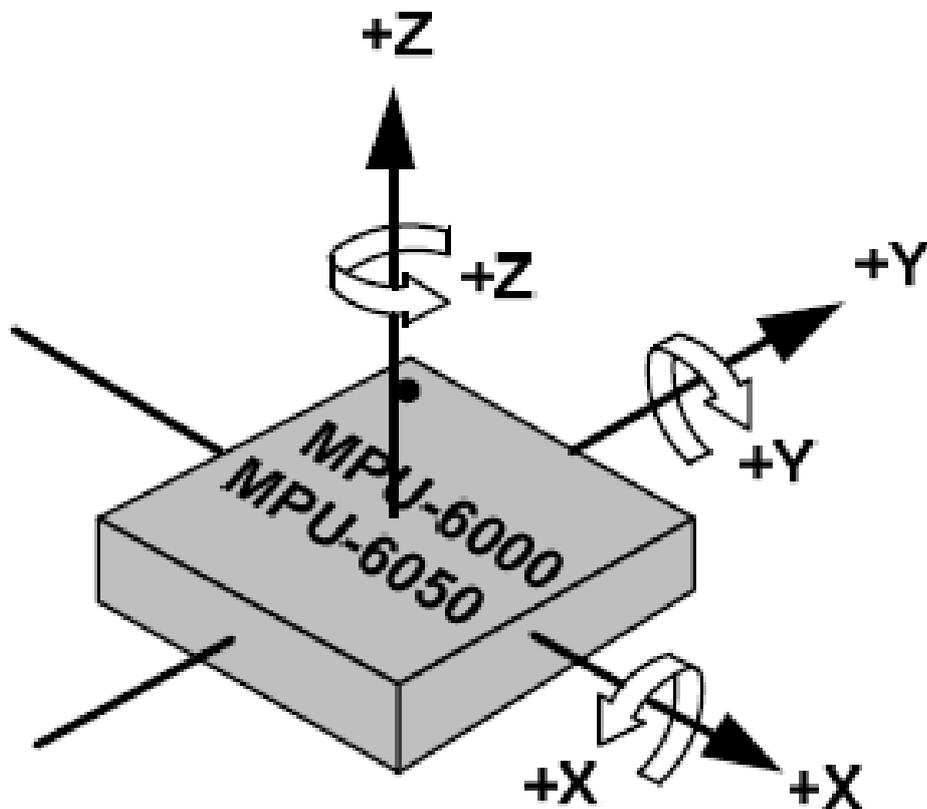
Wie funktioniert es?

IMU-Sensoren bestehen normalerweise aus zwei oder mehr Teilen. Nach Priorität geordnet sind diese: der Beschleunigungssensor, das Gyroskop, ein Magnetometer und ein Höhenmesser. Der MPU 6050 ist ein 6 DOF- (Degrees of Freedom) bzw ein 6-Achsen-IMU-Sensor, was bedeutet, dass der 6 Werte ausgibt. Drei Werte vom Beschleunigungssensor und drei Werte vom Gyroskop. Der Sensor basiert auf MEMS-Technik (Micro Electro Mechanical Systems). Der Beschleunigungssensor und das Gyroskop sind in einem einzelnen Chip eingebettet. Der Chip benutzt das I2C-Protokoll (Inter Integrated Circuit) zur Kommunikation.

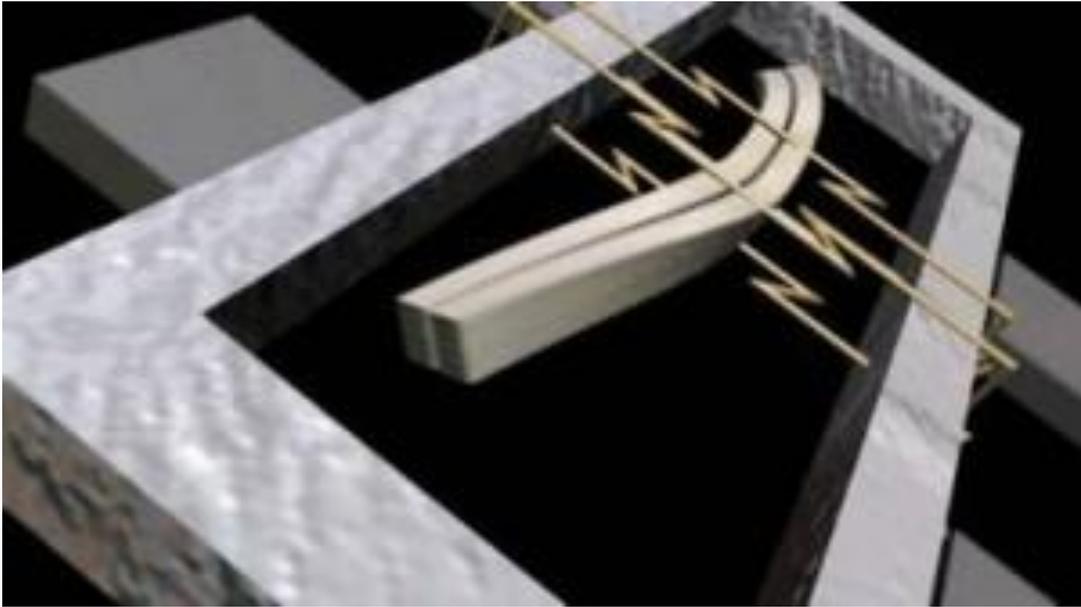
Wie funktioniert ein Beschleunigungssensor?



Ein Beschleunigungssensor funktioniert auf dem Prinzip des Piezoelektrischen Effektes. Stellen Sie sich eine würfelförmige Box vor, mit einem kleinen Ball innerhalb, wie im Bild unten. Die Wände der Box bestehen aus piezoelektrischen Kristallen. Immer, wenn Sie die Box neigen, bewegt sich der Ball durch die Schwerkraft in eine Richtung zu einer Wand hin. Die Wand, auf die der Ball stößt, generiert winzige piezoelektrische Ströme. Es gibt insgesamt drei parallele Wändepaare in einem Würfel. Jedes Paar entspricht einer Achse im dreidimensionalen Raum: die X-, die Y- und die Z-Achse. Abhängig von der generierten Stromstärke, können wir die Richtung der Neigung und ihre Größe berechnen. Zur besseren Verständlichkeit schauen Sie sich bitte das Bild an.

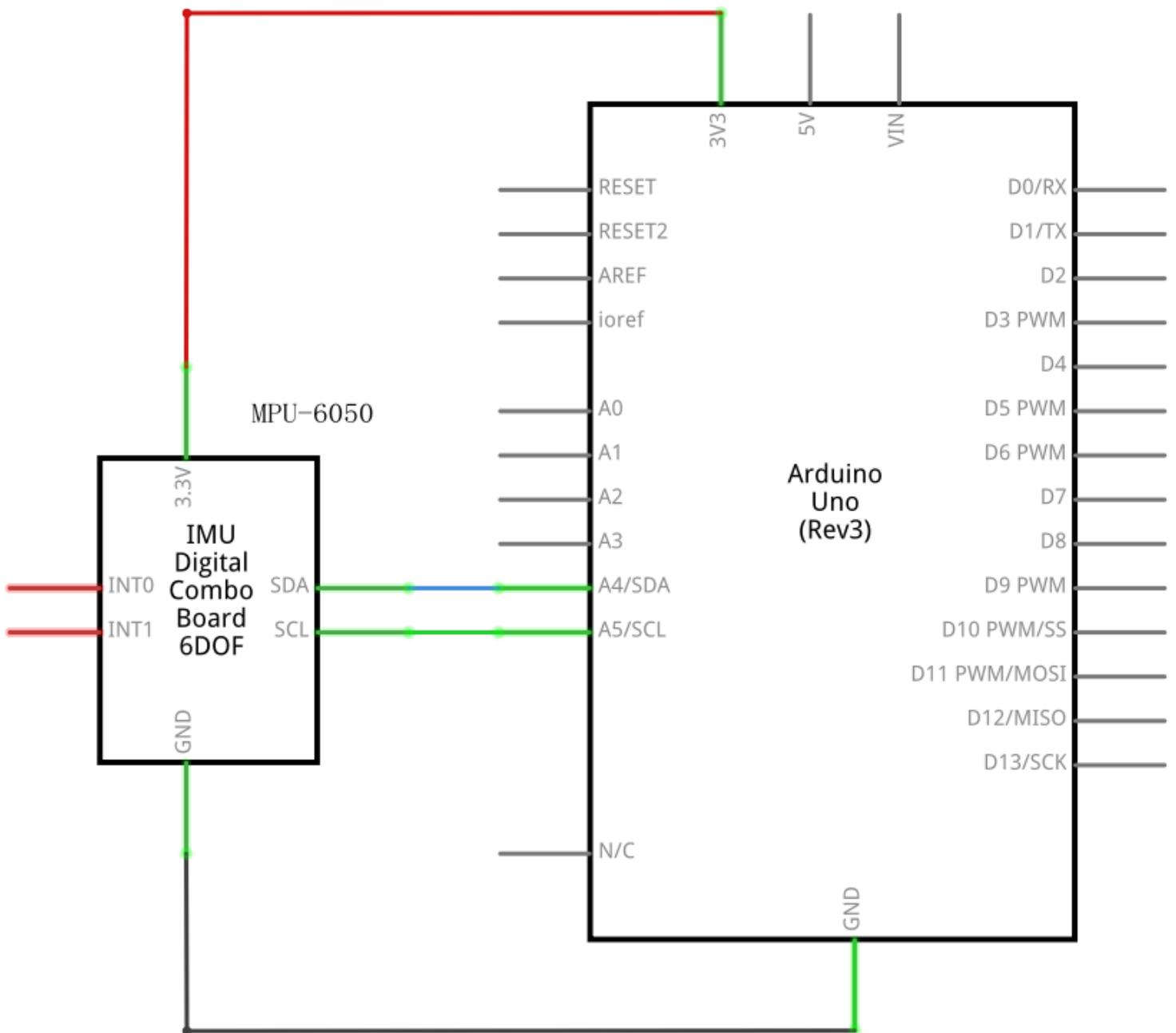


Wie funktioniert ein Gyroskop?

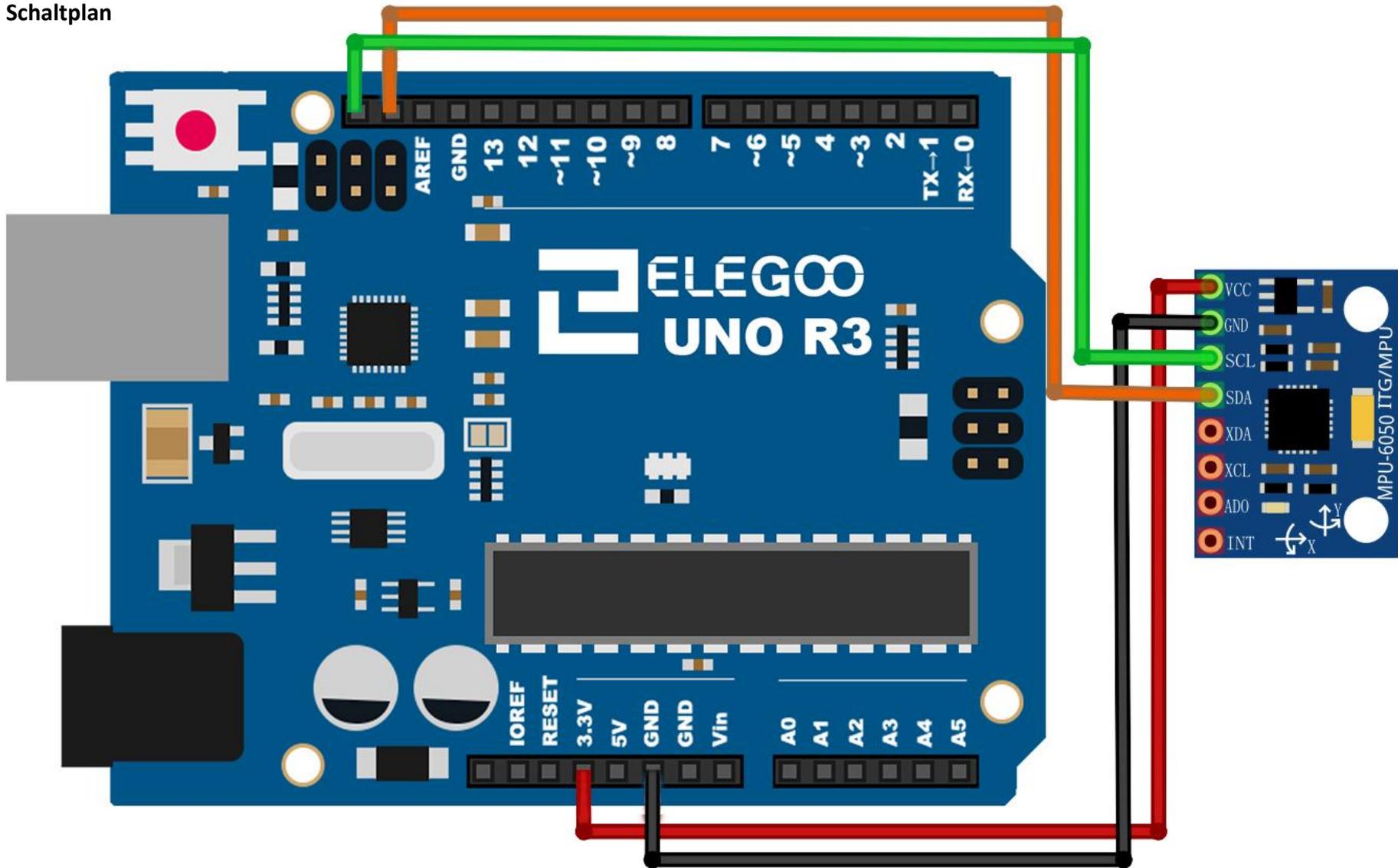


Gyroskope funktionieren auf Grund des Prinzips der Coriolis-Beschleunigung. Stellen Sie sich vor es gibt eine gabelförmige Struktur, die sich konstant hin und her bewegt. Sie wird durch piezoelektrische Kristalle an ihrem Platz gehalten. Wann immer sie versuchen diese Struktur zu neigen, wirkt auf die Kristalle eine Kraft in Richtung der Neigung. Das passiert aufgrund der Trägheit der sich bewegenden Gabel. Die Kristalle produzieren daher einen piezoelektrischen Strom, dessen Stärke dann extern verstärkt wird. Die Werte werden dann durch einen Mikrocontroller wiederaufbereitet und lesbar gemacht.

Verbindungsschema



Schaltplan



Der MPU 6050 kommuniziert mit dem Arduino durch das I2C Protokoll. Die MPU 6050 wird mit dem Arduino wie im obigen Diagramm verbunden.

Das Programm, dass wir benutzen werden, nutzt den *Interrupt*-Pin des Arduinos. Daher muss der digitale Pin *D2* des Arduinos mit dem als „*INT*“ beschrifteten Pin des MPU 6050 verbunden werden. Als nächstes müssen wir die I2C Verbindungen herstellen. Dafür müssen wir den Pin *SDA* des MPUs an den analogen *Pin 4* und den Pin *SCL* des MPUs an den analogen *Pin 5* des Arduinos anschließen. Und das war es auch schon. Sie haben den MPU 6050 erfolgreich verbunden.

Benötigte Bibliotheken

MPU6050

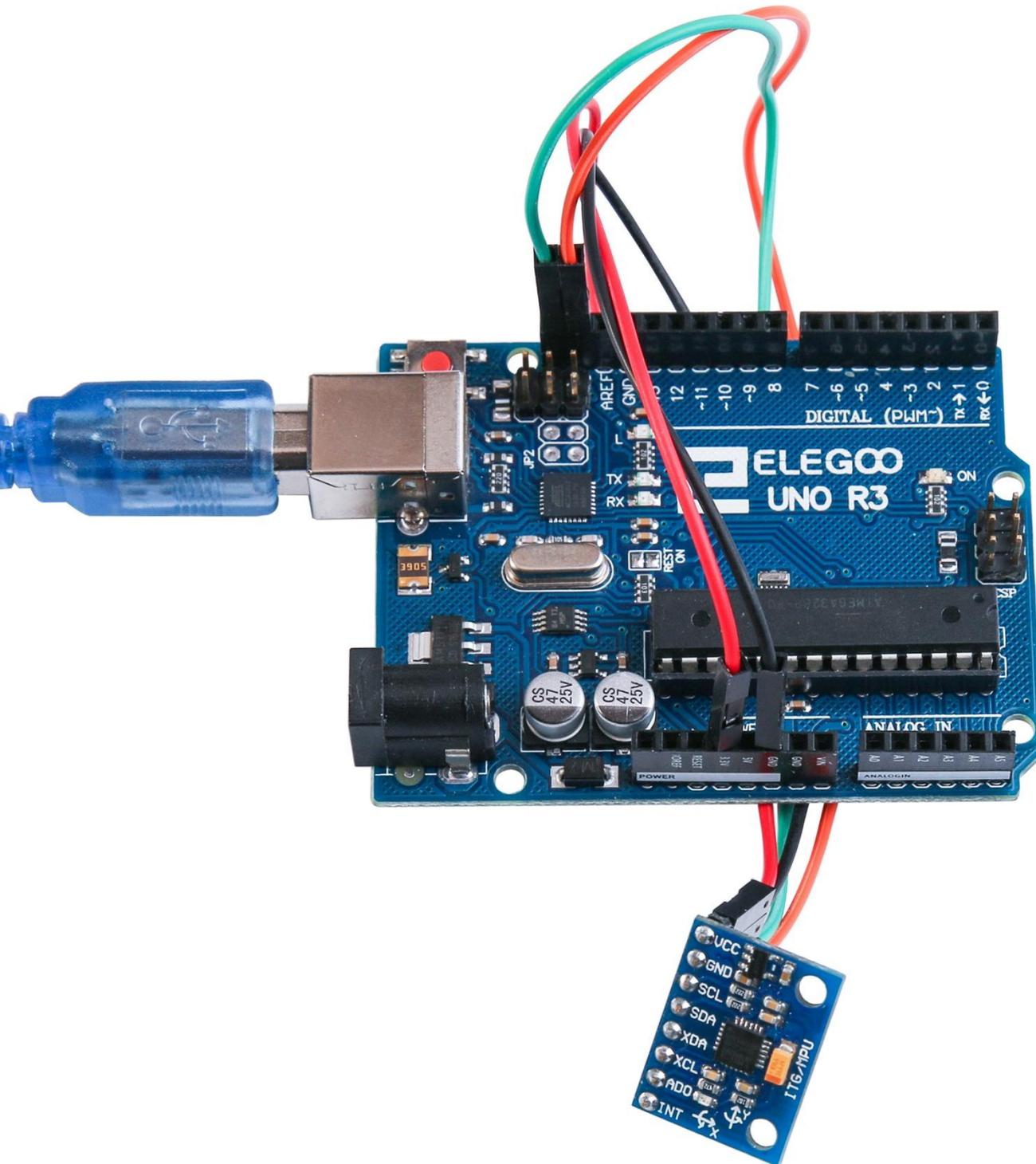
Der Code

Der kurze Beispielsketch ist ein sehr einfacher Sketch und gibt die ganzen Rohwerte aus (Vom Beschleunigungssensor und vom Gyroskop, ebenso die Temperatur). Das sollte mit dem Arduino UNO, Nano, Leonardo und auch dem Due funktionieren.

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 16 MPU-6050 Module*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

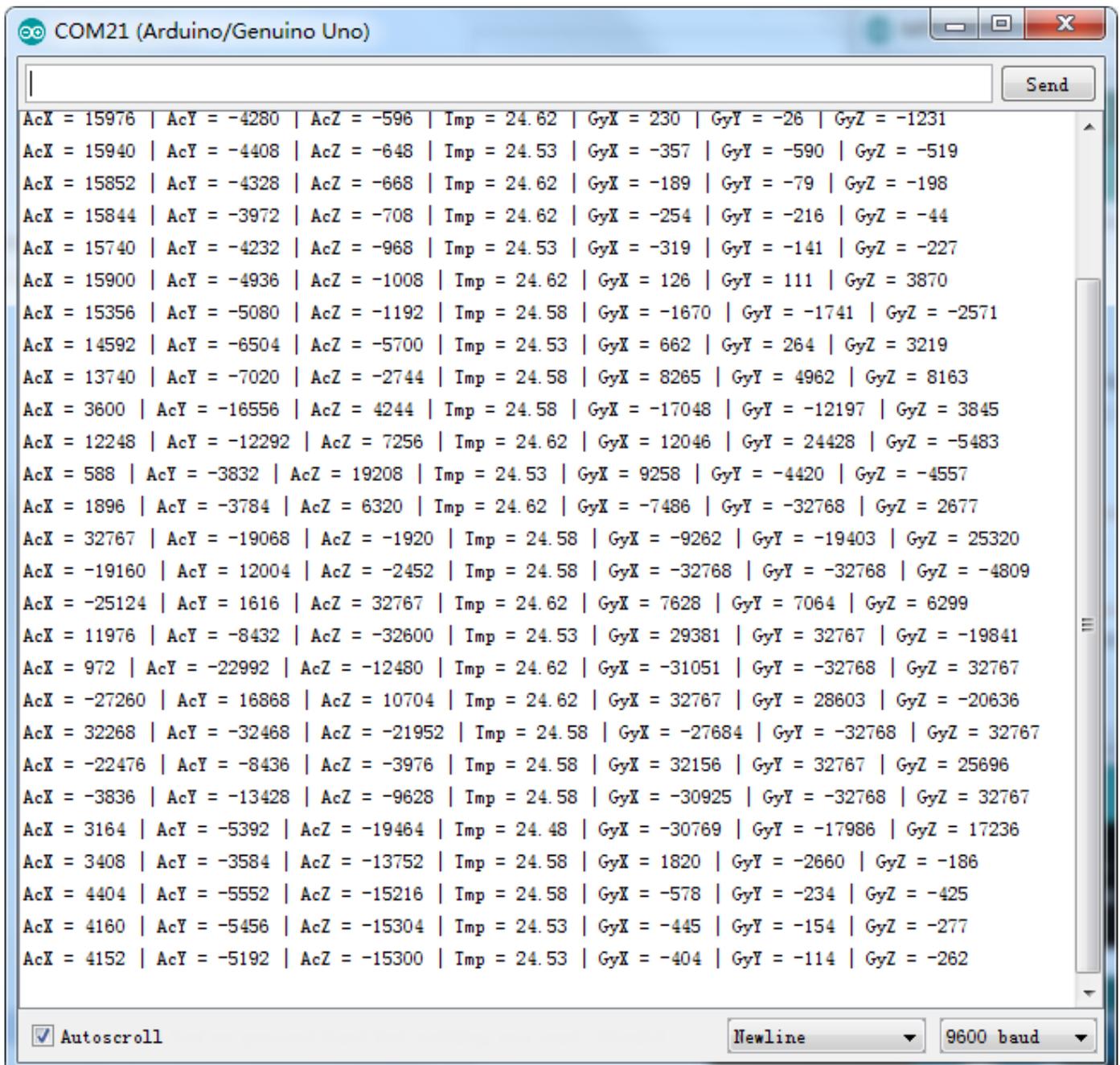
Bevor Sie diesen Sketch hochladen können, müssen Sie die „*MPU6050*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



The screenshot shows the 'COM21 (Arduino/Genuino Uno)' Serial Monitor window. The window title bar includes the Arduino logo, the port name, and standard window controls. A 'Send' button is located in the top right corner of the text area. The main area contains a list of 30 lines of sensor data, each line representing a single data point. The data is formatted as follows: `AcX = [value] | AcY = [value] | AcZ = [value] | Imp = [value] | GyX = [value] | GyY = [value] | GyZ = [value]`. At the bottom of the window, there is a checked 'Autoscroll' checkbox, a 'Newline' dropdown menu, and a '9600 baud' dropdown menu.

```
AcX = 15976 | AcY = -4280 | AcZ = -596 | Imp = 24.62 | GyX = 230 | GyY = -26 | GyZ = -1231
AcX = 15940 | AcY = -4408 | AcZ = -648 | Imp = 24.53 | GyX = -357 | GyY = -590 | GyZ = -519
AcX = 15852 | AcY = -4328 | AcZ = -668 | Imp = 24.62 | GyX = -189 | GyY = -79 | GyZ = -198
AcX = 15844 | AcY = -3972 | AcZ = -708 | Imp = 24.62 | GyX = -254 | GyY = -216 | GyZ = -44
AcX = 15740 | AcY = -4232 | AcZ = -968 | Imp = 24.53 | GyX = -319 | GyY = -141 | GyZ = -227
AcX = 15900 | AcY = -4936 | AcZ = -1008 | Imp = 24.62 | GyX = 126 | GyY = 111 | GyZ = 3870
AcX = 15356 | AcY = -5080 | AcZ = -1192 | Imp = 24.58 | GyX = -1670 | GyY = -1741 | GyZ = -2571
AcX = 14592 | AcY = -6504 | AcZ = -5700 | Imp = 24.53 | GyX = 662 | GyY = 264 | GyZ = 3219
AcX = 13740 | AcY = -7020 | AcZ = -2744 | Imp = 24.58 | GyX = 8265 | GyY = 4962 | GyZ = 8163
AcX = 3600 | AcY = -16556 | AcZ = 4244 | Imp = 24.58 | GyX = -17048 | GyY = -12197 | GyZ = 3845
AcX = 12248 | AcY = -12292 | AcZ = 7256 | Imp = 24.62 | GyX = 12046 | GyY = 24428 | GyZ = -5483
AcX = 588 | AcY = -3832 | AcZ = 19208 | Imp = 24.53 | GyX = 9258 | GyY = -4420 | GyZ = -4557
AcX = 1896 | AcY = -3784 | AcZ = 6320 | Imp = 24.62 | GyX = -7486 | GyY = -32768 | GyZ = 2677
AcX = 32767 | AcY = -19068 | AcZ = -1920 | Imp = 24.58 | GyX = -9262 | GyY = -19403 | GyZ = 25320
AcX = -19160 | AcY = 12004 | AcZ = -2452 | Imp = 24.58 | GyX = -32768 | GyY = -32768 | GyZ = -4809
AcX = -25124 | AcY = 1616 | AcZ = 32767 | Imp = 24.62 | GyX = 7628 | GyY = 7064 | GyZ = 6299
AcX = 11976 | AcY = -8432 | AcZ = -32600 | Imp = 24.53 | GyX = 29381 | GyY = 32767 | GyZ = -19841
AcX = 972 | AcY = -22992 | AcZ = -12480 | Imp = 24.62 | GyX = -31051 | GyY = -32768 | GyZ = 32767
AcX = -27260 | AcY = 16868 | AcZ = 10704 | Imp = 24.62 | GyX = 32767 | GyY = 28603 | GyZ = -20636
AcX = 32268 | AcY = -32468 | AcZ = -21952 | Imp = 24.58 | GyX = -27684 | GyY = -32768 | GyZ = 32767
AcX = -22476 | AcY = -8436 | AcZ = -3976 | Imp = 24.58 | GyX = 32156 | GyY = 32767 | GyZ = 25696
AcX = -3836 | AcY = -13428 | AcZ = -9628 | Imp = 24.58 | GyX = -30925 | GyY = -32768 | GyZ = 32767
AcX = 3164 | AcY = -5392 | AcZ = -19464 | Imp = 24.48 | GyX = -30769 | GyY = -17986 | GyZ = 17236
AcX = 3408 | AcY = -3584 | AcZ = -13752 | Imp = 24.58 | GyX = 1820 | GyY = -2660 | GyZ = -186
AcX = 4404 | AcY = -5552 | AcZ = -15216 | Imp = 24.58 | GyX = -578 | GyY = -234 | GyZ = -425
AcX = 4160 | AcY = -5456 | AcZ = -15304 | Imp = 24.53 | GyX = -445 | GyY = -154 | GyZ = -277
AcX = 4152 | AcY = -5192 | AcZ = -15300 | Imp = 24.53 | GyX = -404 | GyY = -114 | GyZ = -262
```

Lektion17:HC-SR501 Bewegungssensor

Übersicht

In dieser Lektion lernen Sie, wie man einen Bewegungssensor (*PIR-Sensor*) mit dem UNO Board verbindet.

Das UNO Board ist das Herz dieses Projekts. Es ist mit dem Bewegungssensor verbunden und wenn eine Bewegung registriert wird, soll eine LED aufleuchten.

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x HC-SR501 PIR Bewegungssensor

(3) x W-M Kabel (Weiblich zu MännlichDuPontJumper Kabel)

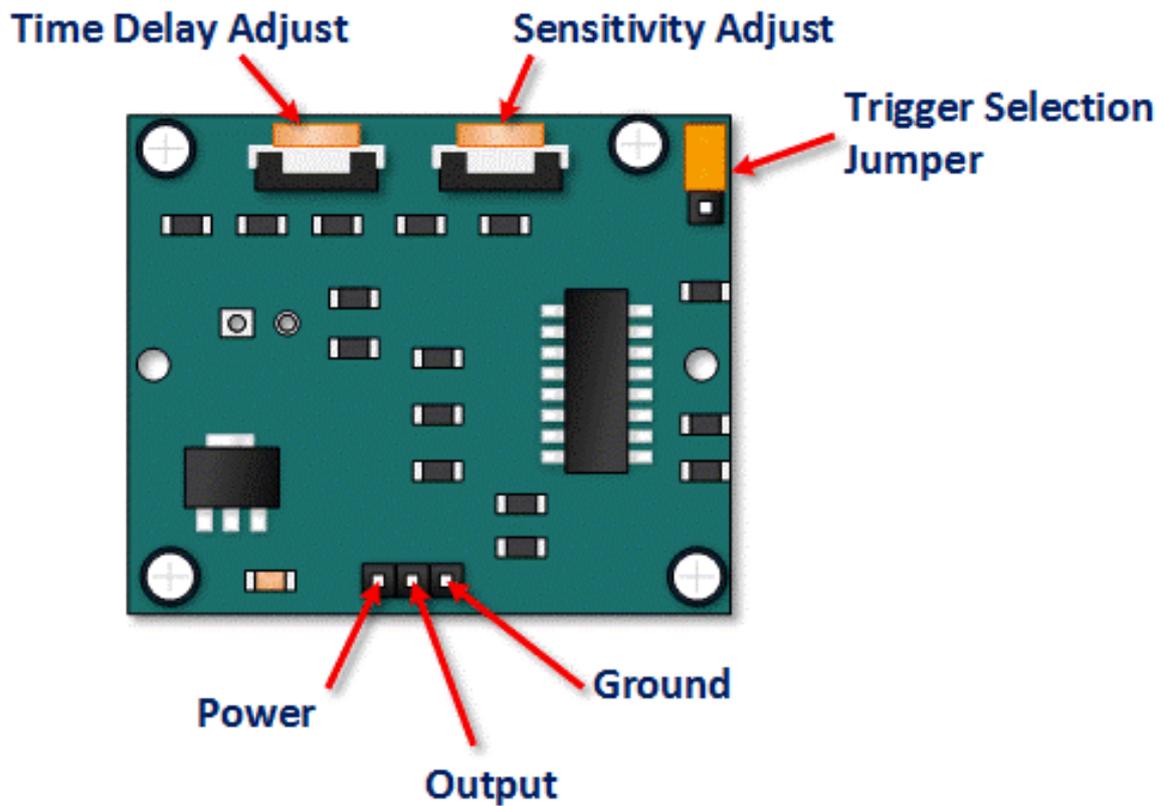
Einführung in die Komponenten

PIR Sensor:

PIR Bewegungssensoren sind deutlich komplizierter aufgebaut als viele der anderen hier vorgestellten Sensoren (wie Fotozellen und Neigungssensoren), weil viele variablen Werte das Verhalten des Sensors beeinflussen.

Der PIR-Sensor selbst hat zwei Sensorpunkte. Jeder der beiden Punkte besteht aus einem speziellen Material, das auf Infrarotlicht reagiert. Die Linse, die hier benutzt wird, dient lediglich dazu mehr Lichtstrahlen einzufangen, ist aber nicht zwingend notwendig. Die zwei Sensorpunkte können eine gewisse Distanz weit „sehen“ (abhängig von der Empfindlichkeit des Sensors). Wenn der Sensor im Ruhezustand ist, registrieren beide Punkte die gleiche Menge an IR-Licht, dem IR-Licht, das von der Umgebung abgestrahlt wird. Wenn ein warmer Körper, wie ein Mensch oder ein Tier, an dem Sensor vorbeiläuft, ändert sich erst die Infrarotlichtmenge am ersten Sensorpunkt und es entsteht eine positive Differenz. Wenn der Körper den Sensorbereich verlässt, passiert genau das Gegenteil: Die Infrarotlichtmenge am ersten Punkt nimmt ab und am zweiten zu und es entsteht eine negative Differenz. Diese wechselnden Pulse sind das, was der Sensor erkennt.





Pin / Steuerelement	Fuktion
Time Delay Adjust	Stellt ein, wie lange der Ausgang nach einer erkannten Bewegung HIGH bleibt: 5 Sekunden bis 5 Minuten.
Sensitivity Adjust	Stellt die Empfindlichkeit von 3 – 7 Metern ein.
Trigger Selection Jumper	Stellt ein, ob der Bewegungssensor mehrmals ausgelöst werden kann.
Ground Pin	GND- / Erdungs-Anschluss
Output Pin	LOW bei keiner Bewegung. HIGH wenn eine Bewegung erkannt wird (HIGH = 3,3V)
Power Pin	5 – 20 Volt Gleichstromversorgung

HC SR501 PIR Funktionsbeschreibung

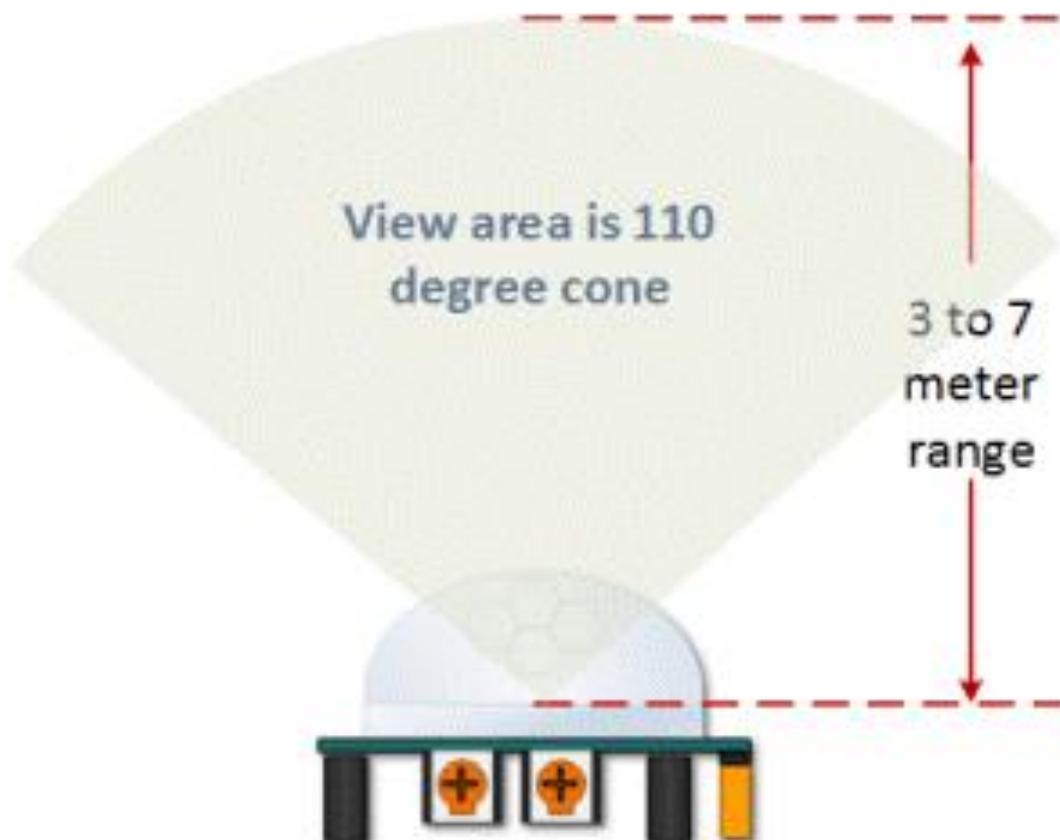
Der SR501 Sensor registriert IR-Licht-Schwankungen und interpretiert diese als Bewegungen und setzt seinen Ausgang anschließend auf LOW. Was als Bewegung und was als keine interpretiert wird, hängt stark von den variablen Einstellmöglichkeiten ab.

Geräte-Initialisierung

Der Sensor benötigt fast eine Minute, um sich zu initialisieren. Während dieser Zeit gibt er möglicherweise falsche Signale aus. Der Schaltkreis bzw die Programmierung des Boards sollte deshalb diese Einschaltdauer berücksichtigen und die Werte in dieser Zeit ignorieren.

Reichweite des Sensors

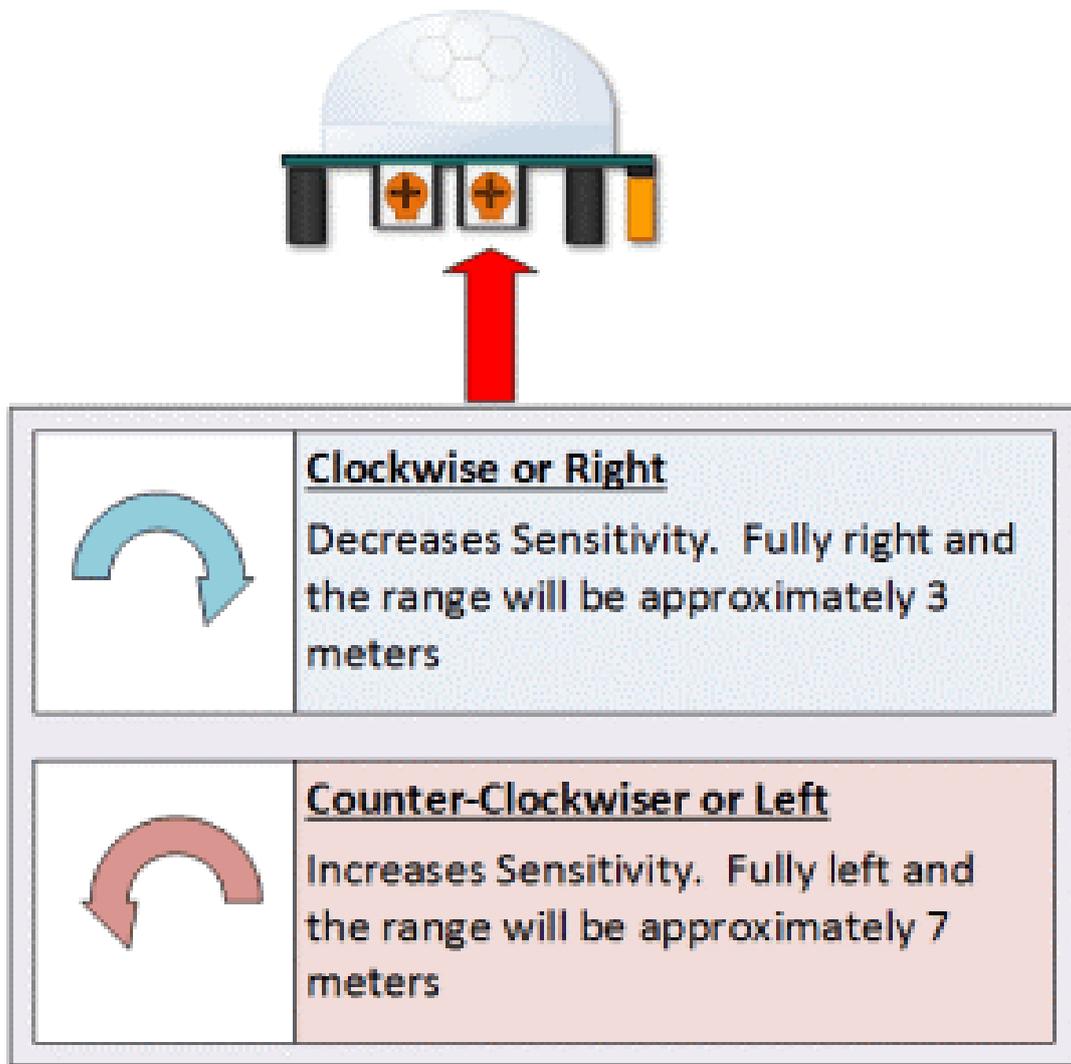
Der Sensor erkennt Bewegungen in einem Winkel von 110° mit einer Entfernung von 3 - 7 Metern.



HC SR501 Reichweite

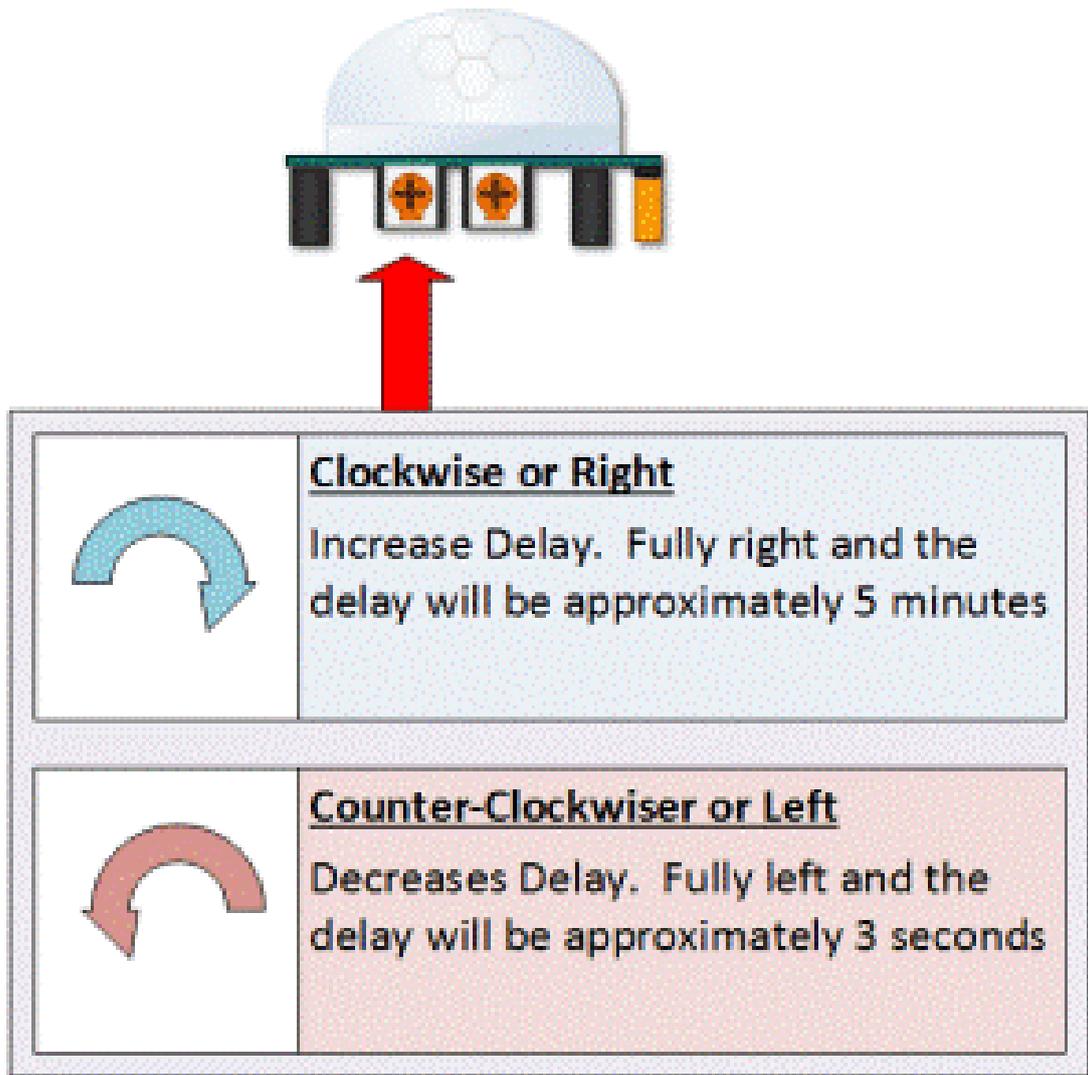
PIR Reichweite (Sensitivity) einstellen

Wie erwähnt kann die Reichweite in einem Bereich von ungefähr 3 bis 7 Metern eingestellt werden. Die Abbildung unten zeigt, wie man die Reichweite anpasst.



HC SR501 Einschaltdauer anpassen

Die Einschaltdauer legt fest, wie lange der Ausgang des PIR Sensormoduls HIGH bleibt, nachdem eine Bewegung erkannt wurde. Der Einstellbereich beträgt 3 Sekunden bis 5 Minuten. Die Abbildung unten zeigt, wie man die Einschaltdauer anpasst.



HC SR501 Einschaltdauer Hinweis – Wichtig!

Der Ausgang des Sensors wird 3 Sekunden nachdem die eingestellte Einschaltdauer abgelaufen ist durchgehend LOW senden. Mit anderen Worten: In diesen 3 Sekunden nach der letzten Aktivierung wird KEINERLEI Bewegung erkannt.

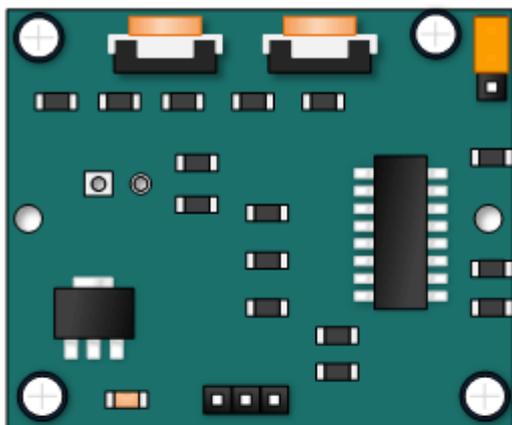
Beispiel:

Stellen Sie sich vor, Sie haben die Einschaltdauer auf 5 Sekunden festgelegt. Der PIR-Sensor erkennt nun eine Bewegung und wird für 5 Sekunden aktiv (= sendet HIGH). Nach diesen 5 Sekunden sendet der PIR-Sensor 3 Sekunden lang LOW, egal ob er in der Zeit eine Bewegung erkennt oder nicht. In dieser Zeit werden keine Bewegungen registriert. Nach den drei Sekunden kann der Sensor wieder Bewegungen erkennen und schaltet bei registrierter Bewegung wieder auf HIGH.

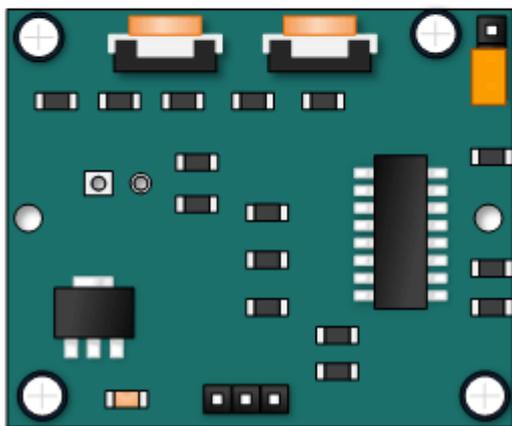
Trigger Modus Jumper

Der Trigger Jumper erlaubt Ihnen zwischen zwei Modi umzuschalten: *Single Trigger* (einmalige Aktivierung) und *Repeatable Trigger* (wiederholbare Aktivierung). Diese Einstellung beeinflusst die Einschaltdauer des Sensors.

- **SINGLE TRIGGER** / Einmalige Aktivierung – Die Einschaltdauer beginnt mit der ersten registrierten Bewegung und läuft solange, bis die Zeit abgelaufen ist.
- **REPEATABLE TRIGGER** / Wiederholbare Aktivierung – Die Einschaltdauer wird mit jeder neuen registrierten Bewegung zurückgesetzt.



Single Trigger Mode – Time Delay is started immediately upon detecting motion. Continued detection is blocked



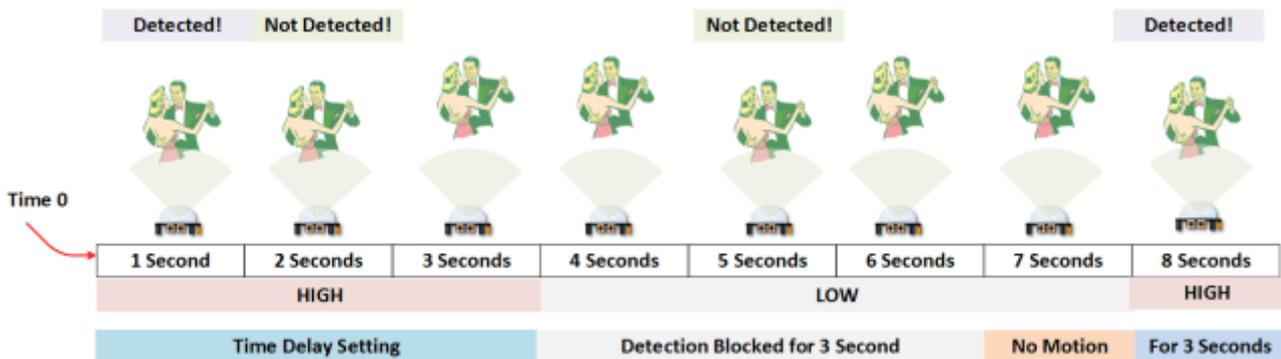
Repeatable Trigger Mode – Time Delay is re-started every time motion is detected.

HC-SR501 Anwendungsbeispiel Tanzfläche

Stellen Sie sich vor Sie wollen das Licht nur an den Stellen einer Tanzfläche einschalten, wo sich Leute befinden. Dabei muss man berücksichtigen, wie sich die Einschaltdauer und der Trigger Modus auf das Verhalten des Sensors auswirken, damit sich die Beleuchtung so verhält, wie es gewollt ist.

Beispiel 1

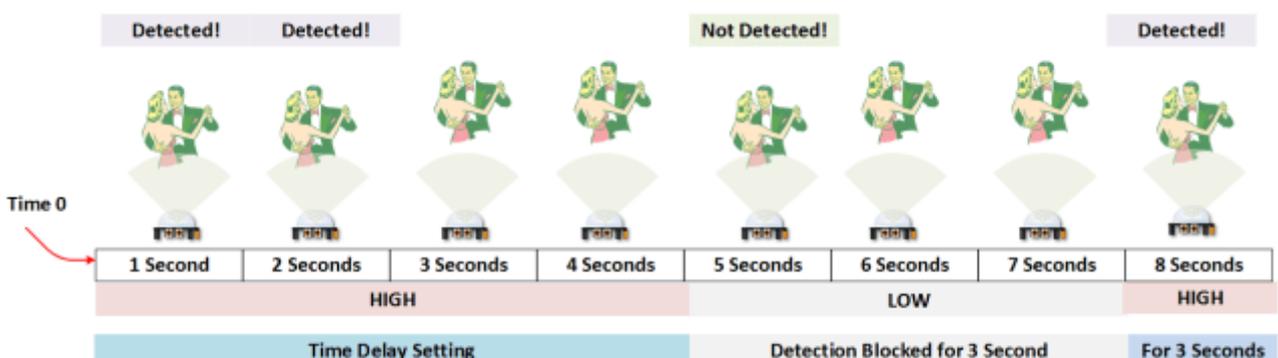
Im ersten Beispiel ist die Einschaltdauer auf 3 Sekunden gesetzt und der Trigger Modus auf Single Trigger. Wie sie der Abbildung entnehmen können, wird die Bewegung nicht immer erkannt. Es gibt im Endeffekt eine Zeitspanne von 6 Sekunden, in der der Sensor nicht auf Bewegungen reagiert.



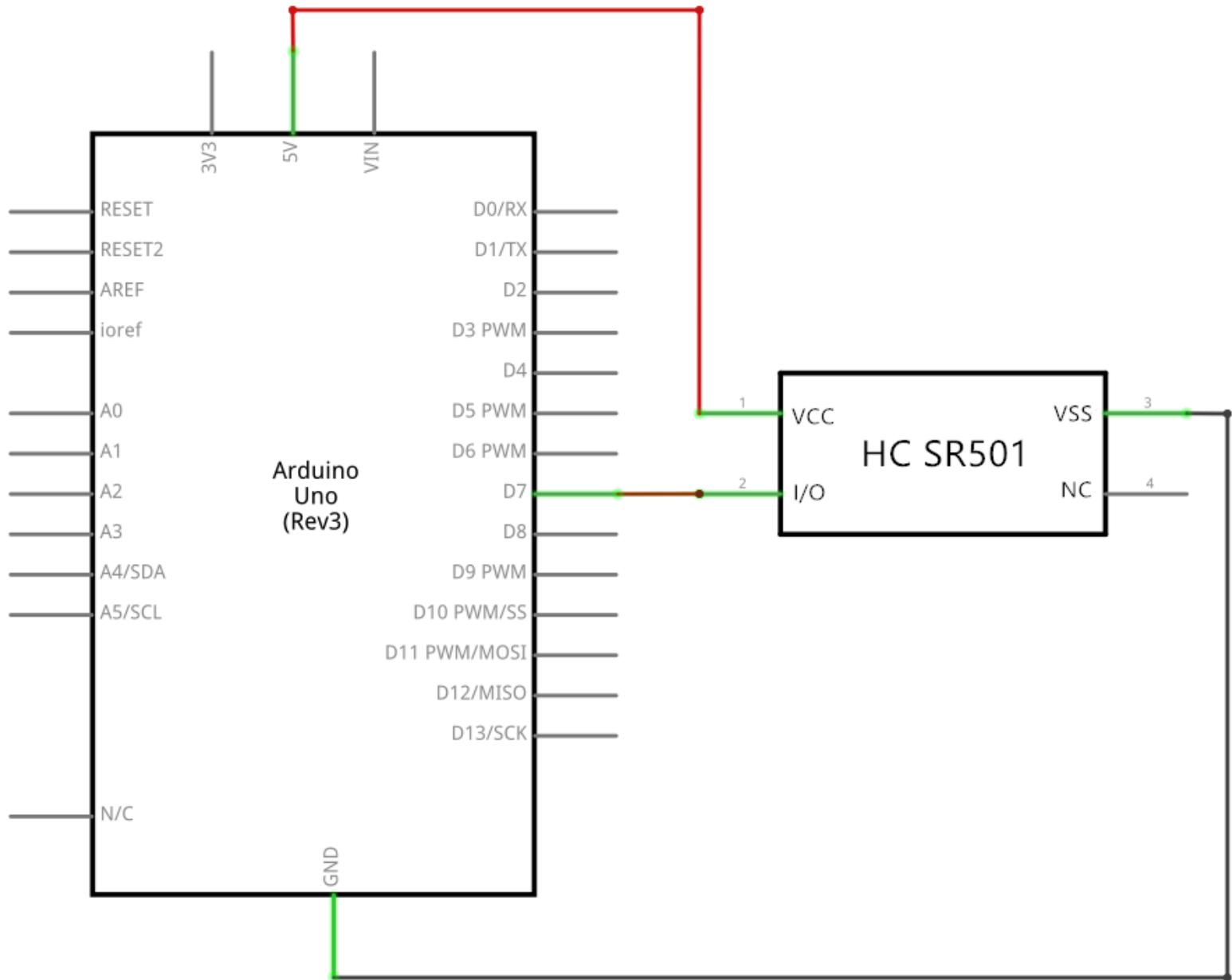
Beispiel 2

Im zweiten Beispiel ist die Beleuchtungsdauer ebenfalls bei drei Sekunden gesetzt, aber der Trigger Modus steht dagegen auf repeatable Trigger. In der Abbildung unten sehen Sie, dass die Beleuchtungsdauer zurückgesetzt wird. Nichtsdestotrotz wird die Erkennung nach Ablauf der Dauer für drei Sekunden blockiert.

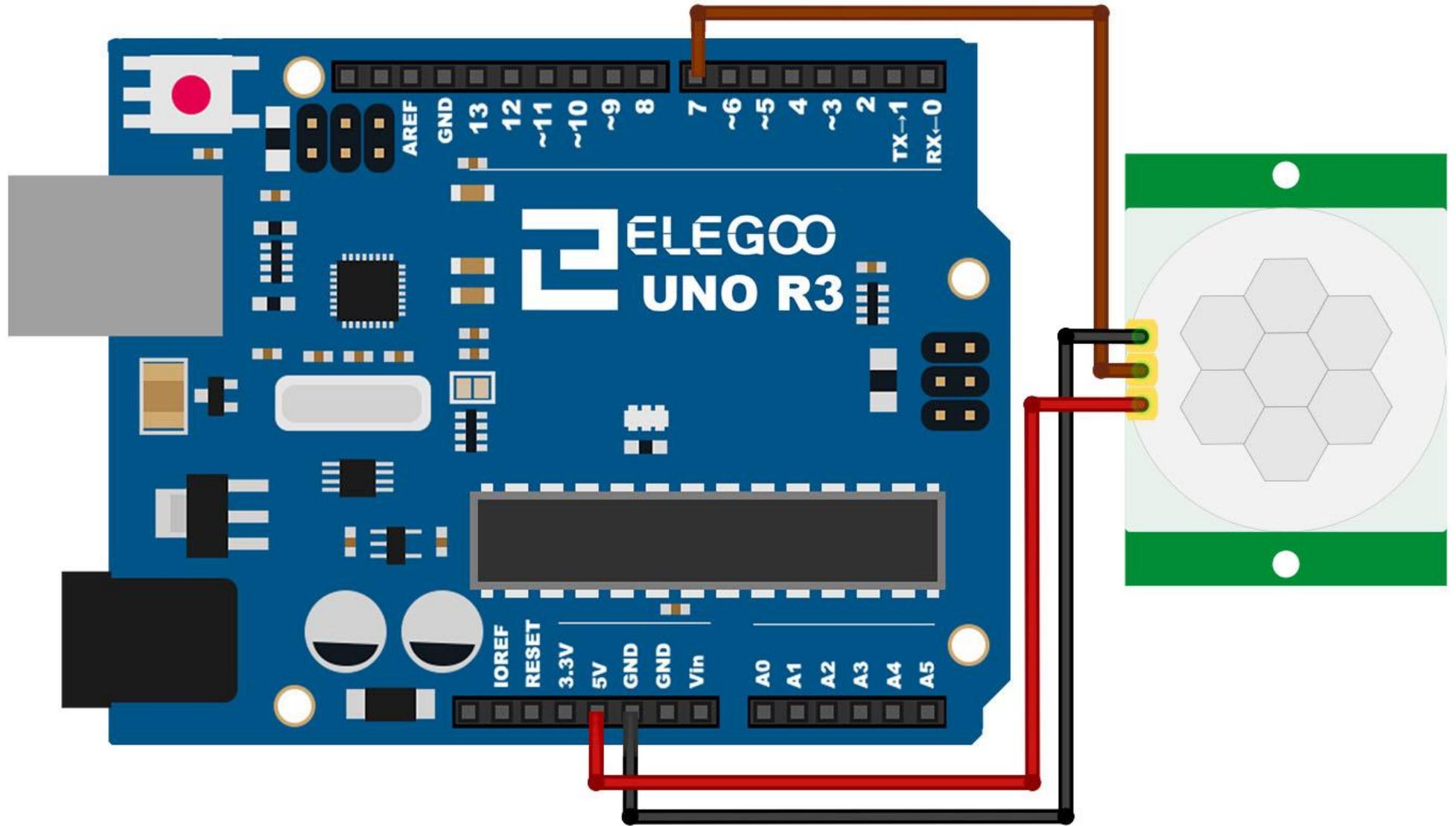
Die 3 Sekunden Ruhezeit könnte man durch eine kreative Programmierung überbrücken. Beachten sie allerdings, dass nicht alle elektronischen Geräte ein ständiges schnelles An- und Ausschalten vertragen.



Verbindungsschema



Schaltplan



PIR-Sensoren mit einem Mikrocontroller zu verbinden ist sehr einfach. Der PIR-Sensor fungiert als digitaler Ausgang. Also müssen Sie lediglich auf ein LOW- (keine Bewegung) bzw ein HIGH-Signal (Bewegung erkannt) warten.

Normalerweise ist eine wiederholbare Aktivierung (repeatable Trigger) erwünscht, also setzen sie den Jumper in die entsprechende Position. Verbinden Sie den Sensor mit 5V und GND und anschließend den Ausgang mit einem beliebigen digitalen Pin des UNO Boards. In diesem Beispiel nutzen wir den *Pin 7*.

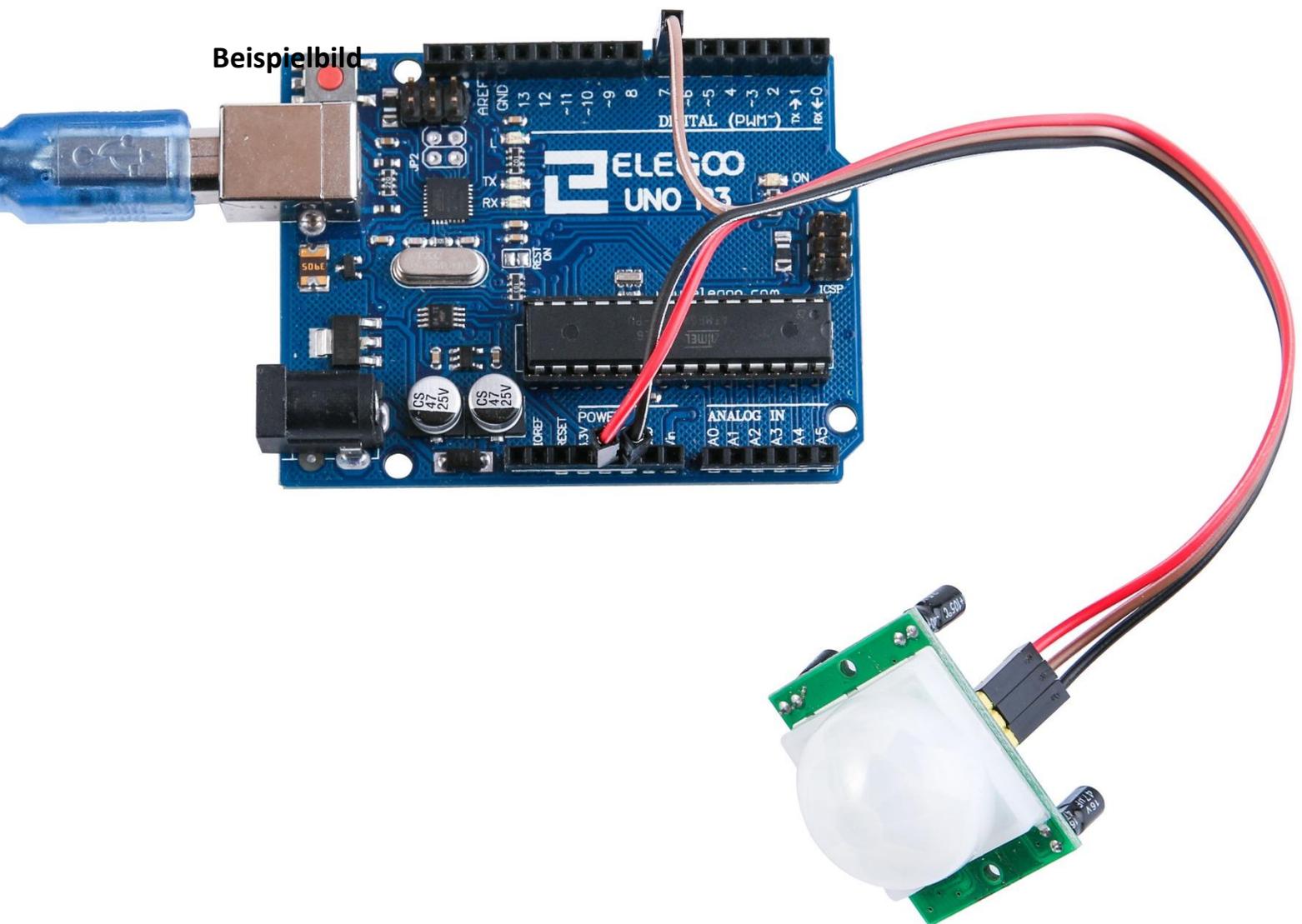
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 17 HC-SR501 PIR Sensor“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte [Lektion 2](#) noch einmal an.

Der Sketch wird einfach die Arduino LED, die mit Pin 13 verbunden ist, einschalten, wenn eine Bewegung erkannt wurde.

Achten Sie darauf die Initialisierungszeit von 1 Minute in Ihrem Code zu überbrücken.

Beispielbild



Lektion 18:Wasserstandssensormodul

Übersicht

In dieser Lektion lernen Sie, wie man einen Wasserstandssensor benutzt.

Dieses Modul kann den Wasserstand abschätzen. Der Sensor besteht im Wesentlichen aus einem Verstärkerschaltkreis bestehend aus einem Transistor mit mehreren kammförmig verteilten Antennen auf dem PCB (Platine). Wenn der Sensor in Wasser getaucht wird, fungieren diese Antennen als Widerstand, der sich ändert, je nachdem wie hoch bzw. tief das Wasser steht. Dann wird aus diesem Widerstandswert der Wasserstand berechnet und anschließend in ein elektrisches Signal umgewandelt und wir können die Veränderung des Wasserstands über die ADC-Funktion des UNO Boards beobachten.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (3) x W-M Kabel (Weiblich zu Männlich/DuPont/Jumper Kabel)
- (1) x Wasserstandssensormodul

Einführung in die Komponenten



Wassersensor:

Wassersensoren wurden entwickelt, um anliegendes Wasser erkennen, was auf viele verschiedene Arten genutzt werden kann, wie zum Beispiel zum Erkennen von Regen, dem Wasserstand oder auslaufender Flüssigkeiten. Der Sensor besteht im Groben aus drei Teilen: Ein elektronischer Anschluss, ein 1 MΩ Widerstand und mehrere blanke Leiter / Antennen.

Dieser Sensor funktioniert durch eine Reihe an freiliegenden Leitungen, die mit GND verbunden sind. Zwischen diesen GND-Leitungen liegen die Sensorleitungen.

Die Sensorleitungen sind an einen schwachen Pullup-Widerstand mit 1 MΩ angeschlossen. Der Widerstand sorgt dafür, dass der Ausgang des Sensors solange HIGH ist, bis ein Wassertropfen oder ähnliches eine Sensorleitung mit einer GND-Leitung verbindet. Dieser Sensor kann an einen digitalen Pin des UNO Boards angeschlossen werden, womit erkannt werden kann, ob Wasser anliegt. Wenn man ihn dagegen an einen analogen Pin anschließt, kann man zusätzlich noch den

Wasserstand bzw die Wassermenge herrausfinden.

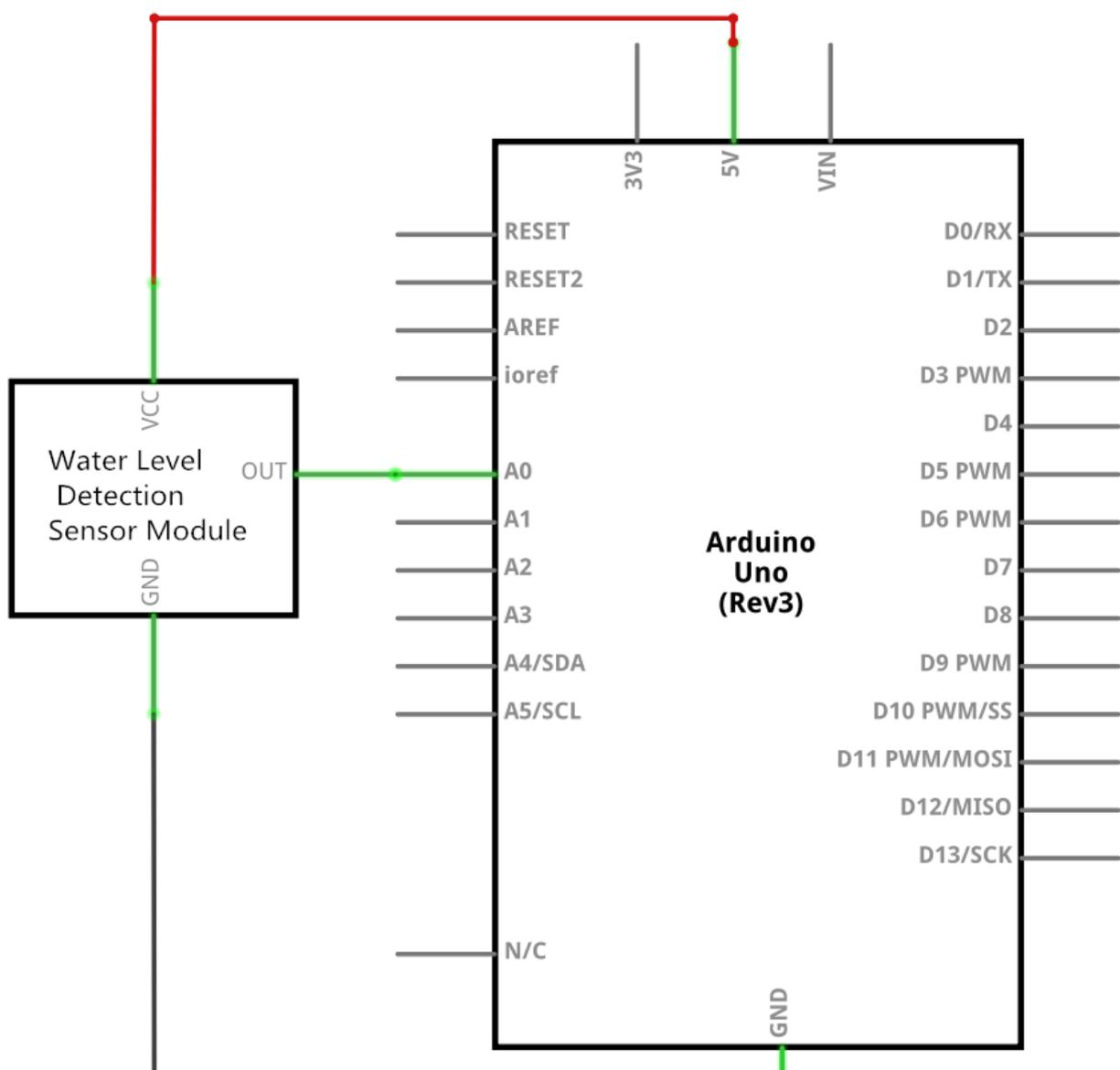
Dieser Sensor kann den Wasserstand durch eine Reihe an freiliegenden parallelgeschalteten Leitungen messen. Er kann die Wassermenge zu einem analogen Signal berechnen, das direkt im Programmcode benutzt werden kann, um den Wasserstand zu berechnen.

Er hat einen geringen Stromverbrauch, aber dennoch eine hohe Empfindlichkeit.

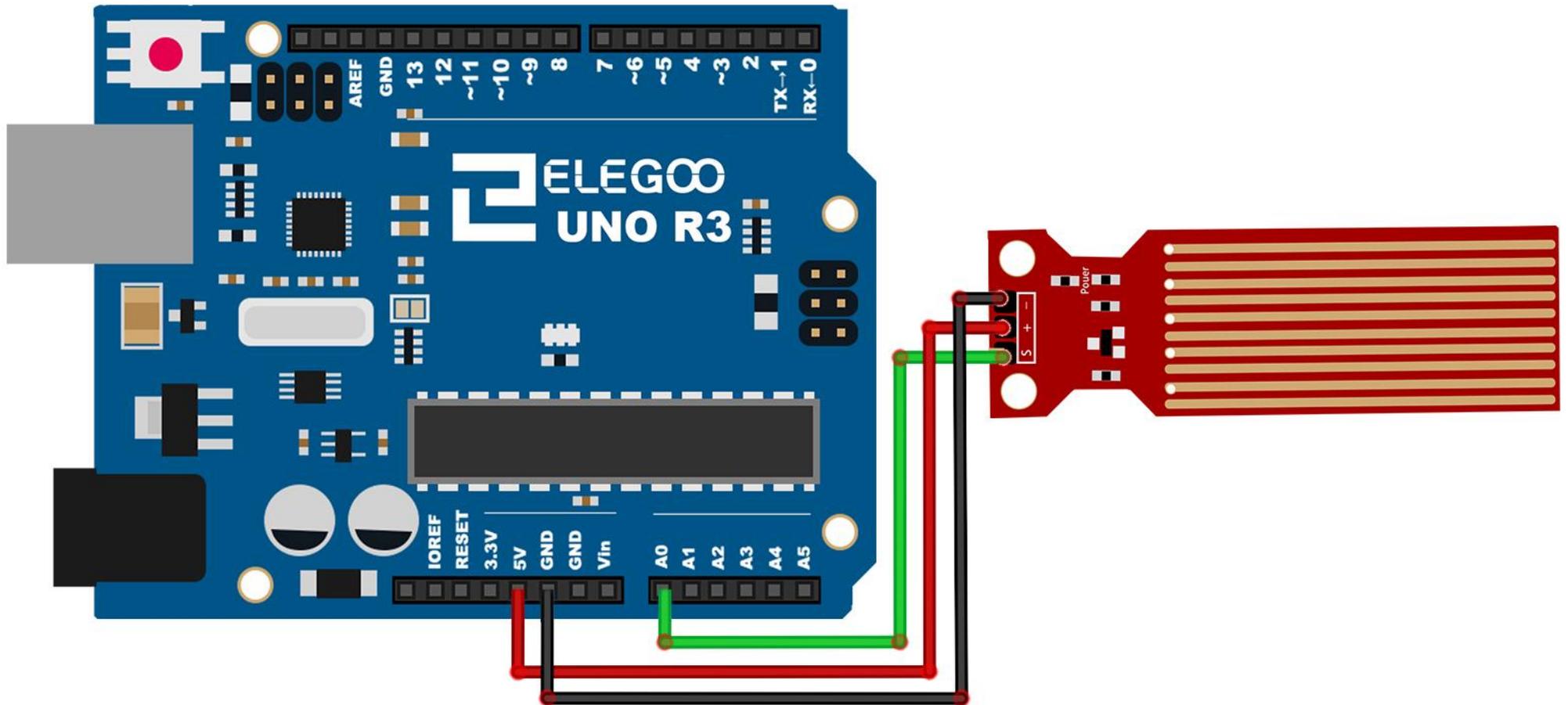
Features:

- 1、 Betriebsspannung: 5V
- 2、 Betriebsstrom:<20ma
- 3、 Schnittstelle: Analog
- 4、 Erkennungsbereich: 40mm×16mm
- 5、 Betriebstemperatur: 10°C~30°C
- 6、 Ausgangsspannung des Signals: 0~4,2V

Verbindungsschema



Schaltplan

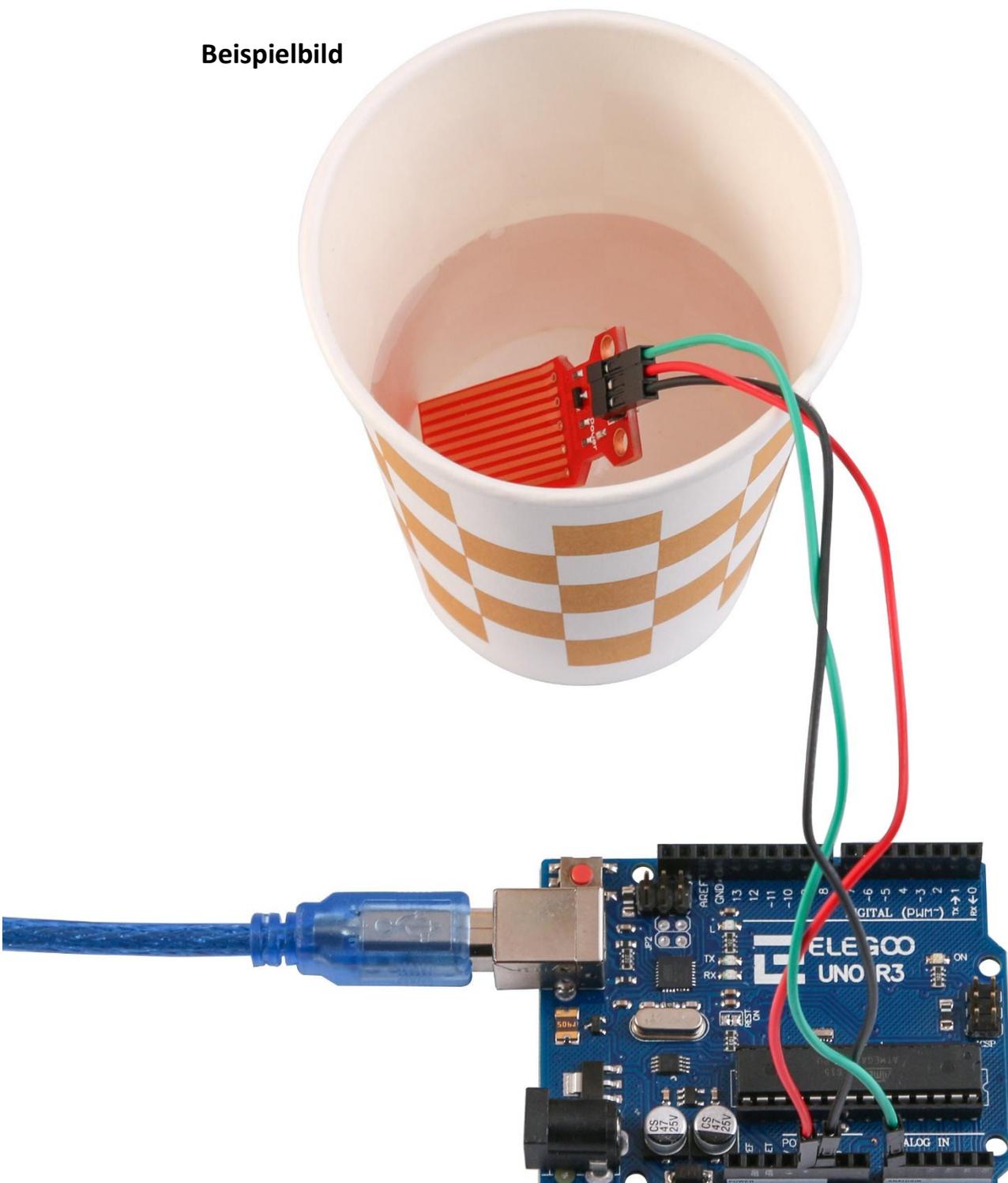


Anschluss des Sensors: Die Stromversorgung (+) wird mit dem 5V-Pin des UNO R3 Boards, der negative Anschluss (-) wird mit dem GND-Pin verbunden. Der Signalausgang (S) wird mit einem der Analogen Pins (A0-A5) des Boards verbunden.

Code

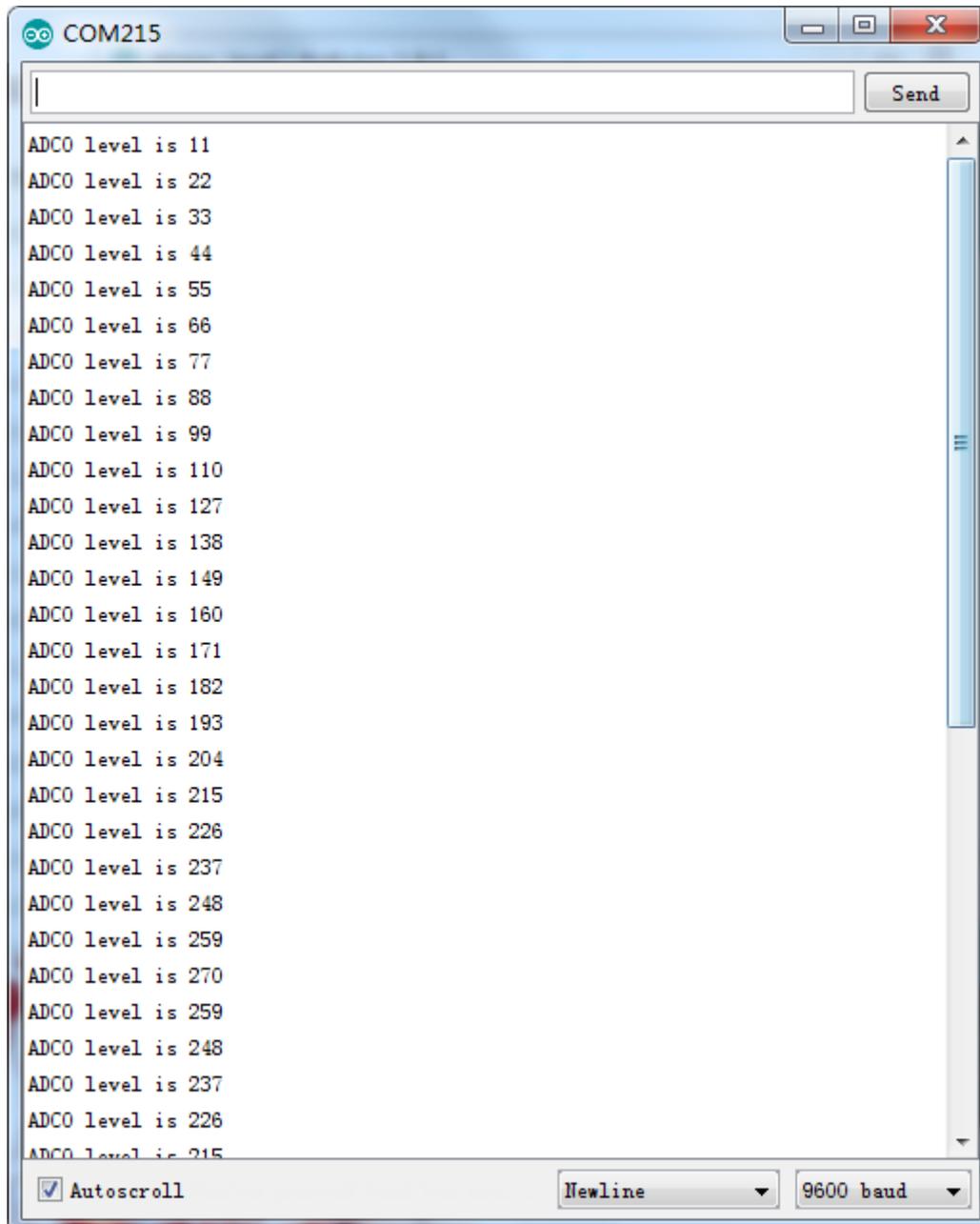
Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 18 Water Level Detection Sensor Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



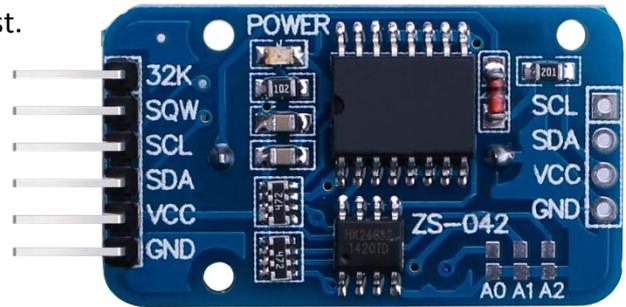
Lektion19:Echtzeituhr-Modul

Übersicht

In dieser Lektion lernen Sie, wie man das DS3231 RTC (Echtzeituhr) -Modul, das das aktuelle Jahr, den Monat, die Woche, den Tag sowie Stunde, Minute und Sekunde anzeigen kann. Es wird mit einer kleinen Backupbatterie betrieben, wenn es nicht an ein UNO Board angeschlossen ist.

Benötigte Bauteile:

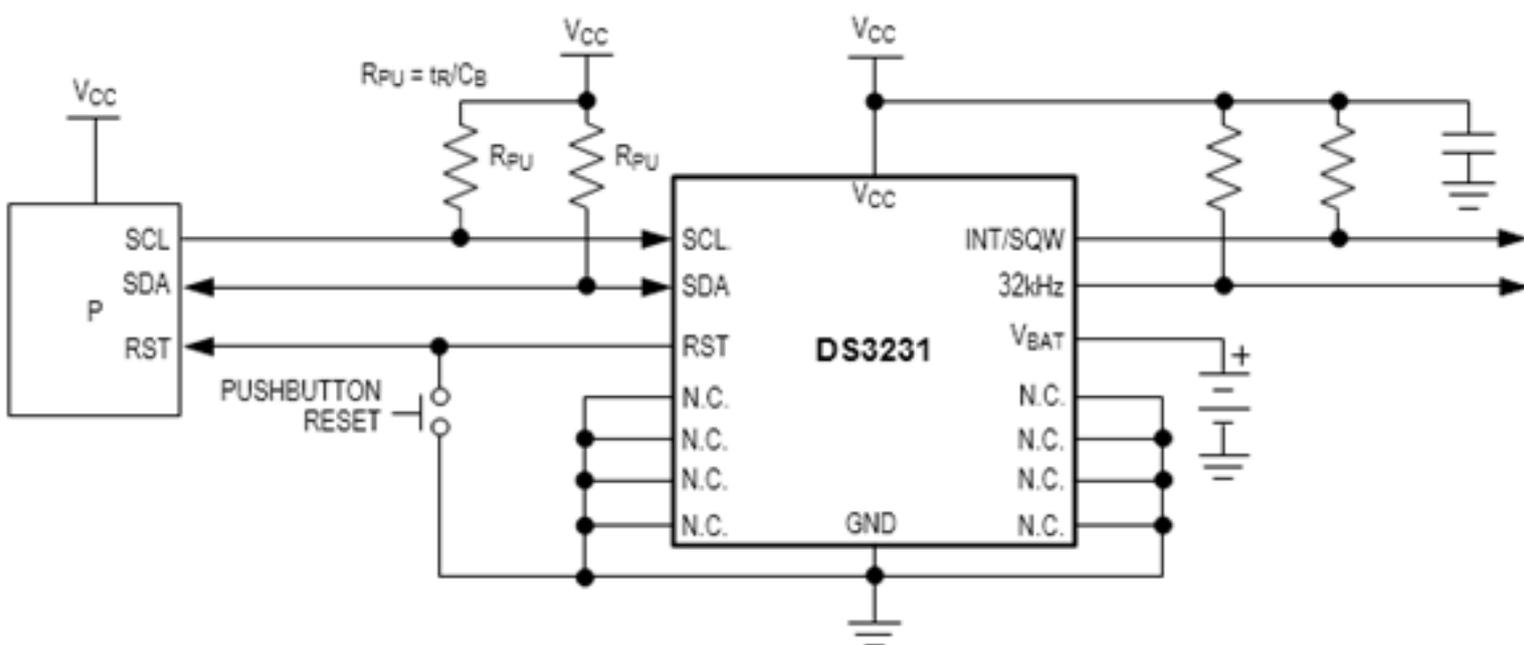
- (1) x Elegoo UNO R3
- (1) x DS3231 RTCModul
- (4) x W-M Kabel (Weiblich zu MännlichDuPontJumper Kabel)



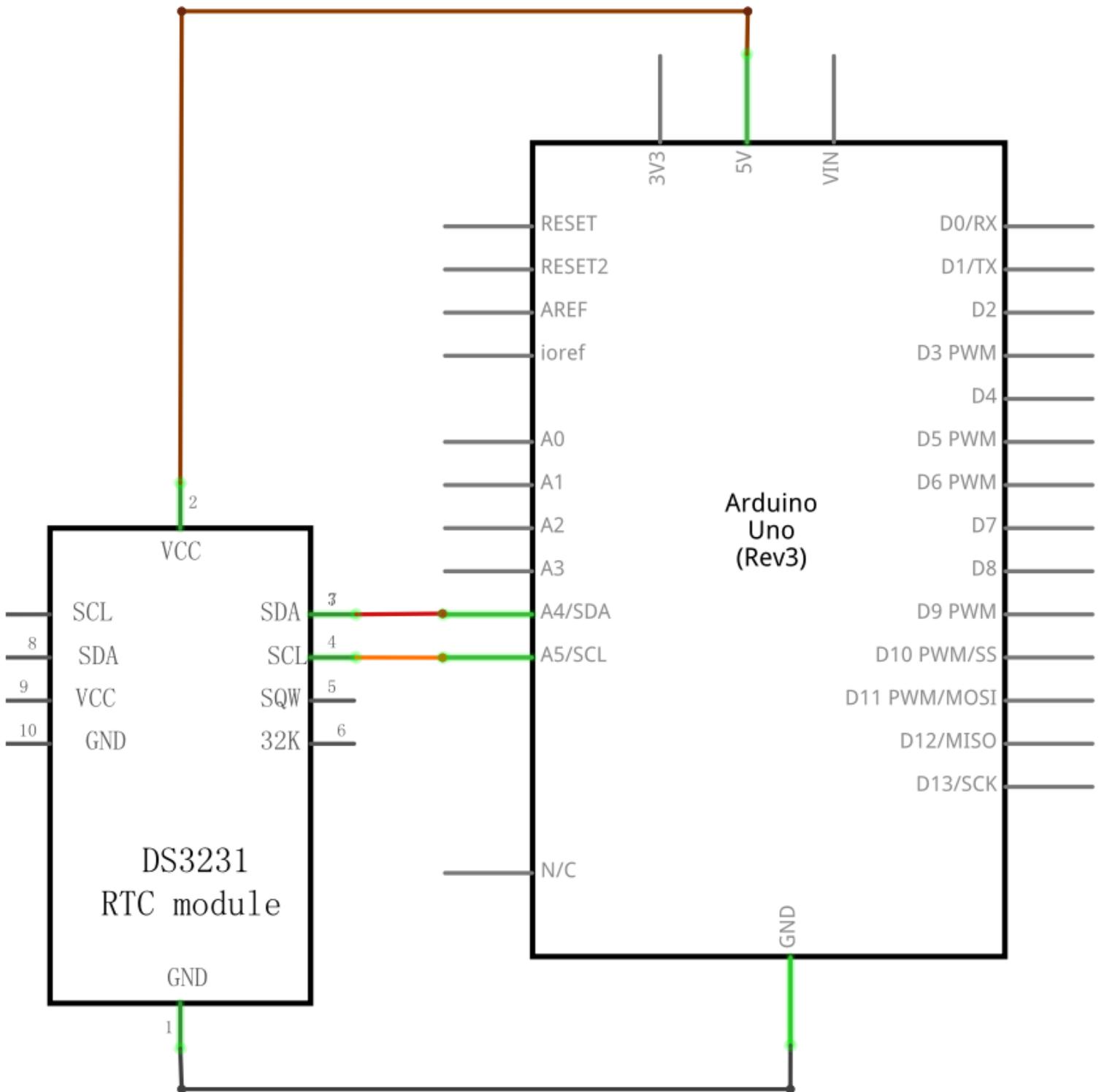
Einführung in die Komponenten

DS3231

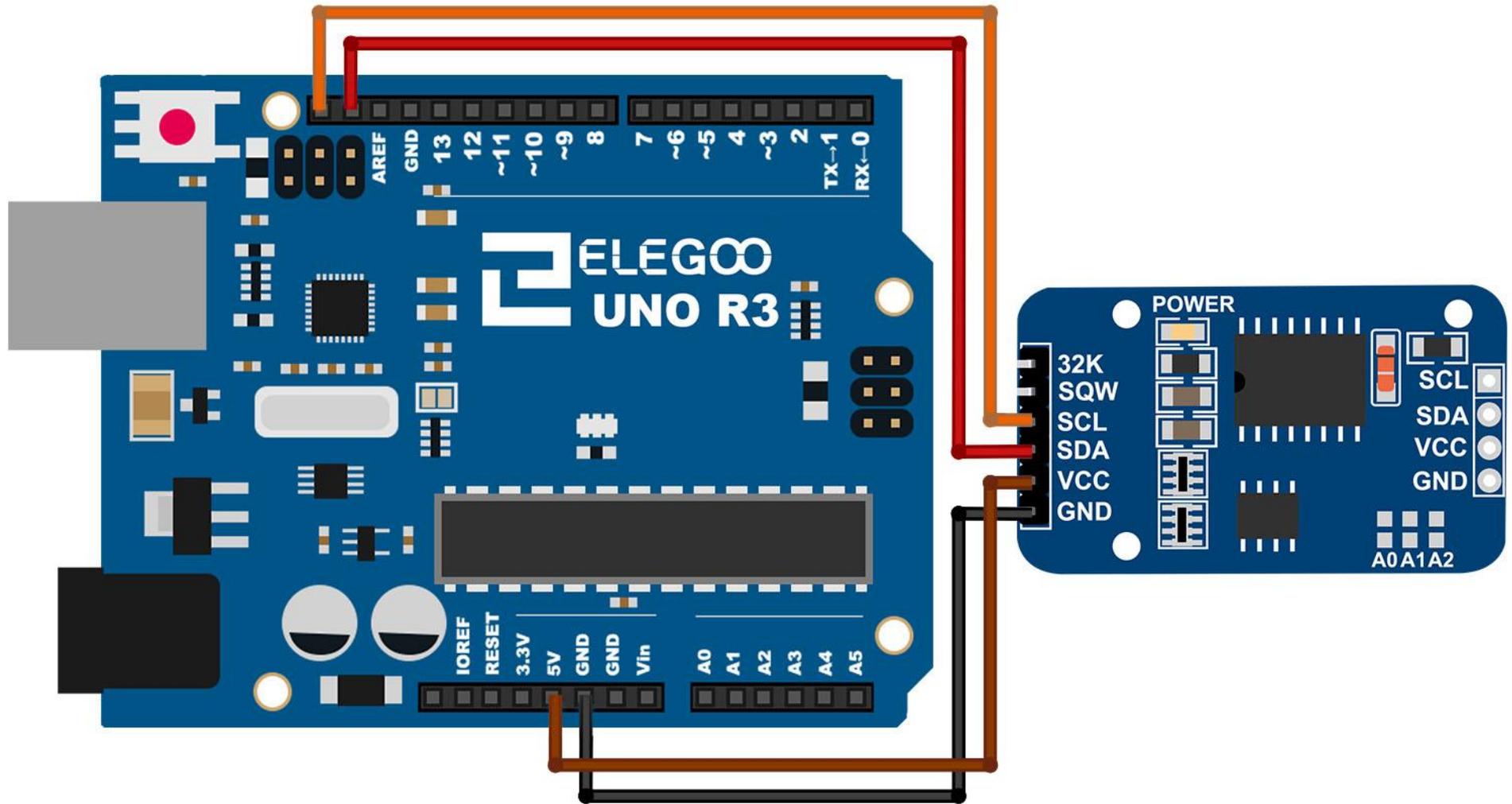
Der DS3231 ist ein einfacher Chip, der die Zeit anzeigen kann. Er hat eine kleine integrierte Batterie, sodass die Uhr weiterarbeitet, auch wenn der Chip nicht mehr mit dem UNO Board verbunden ist.



Verbindungsschema



Schaltplan



Schließen Sie das Modul wie im folgenden Bild an.

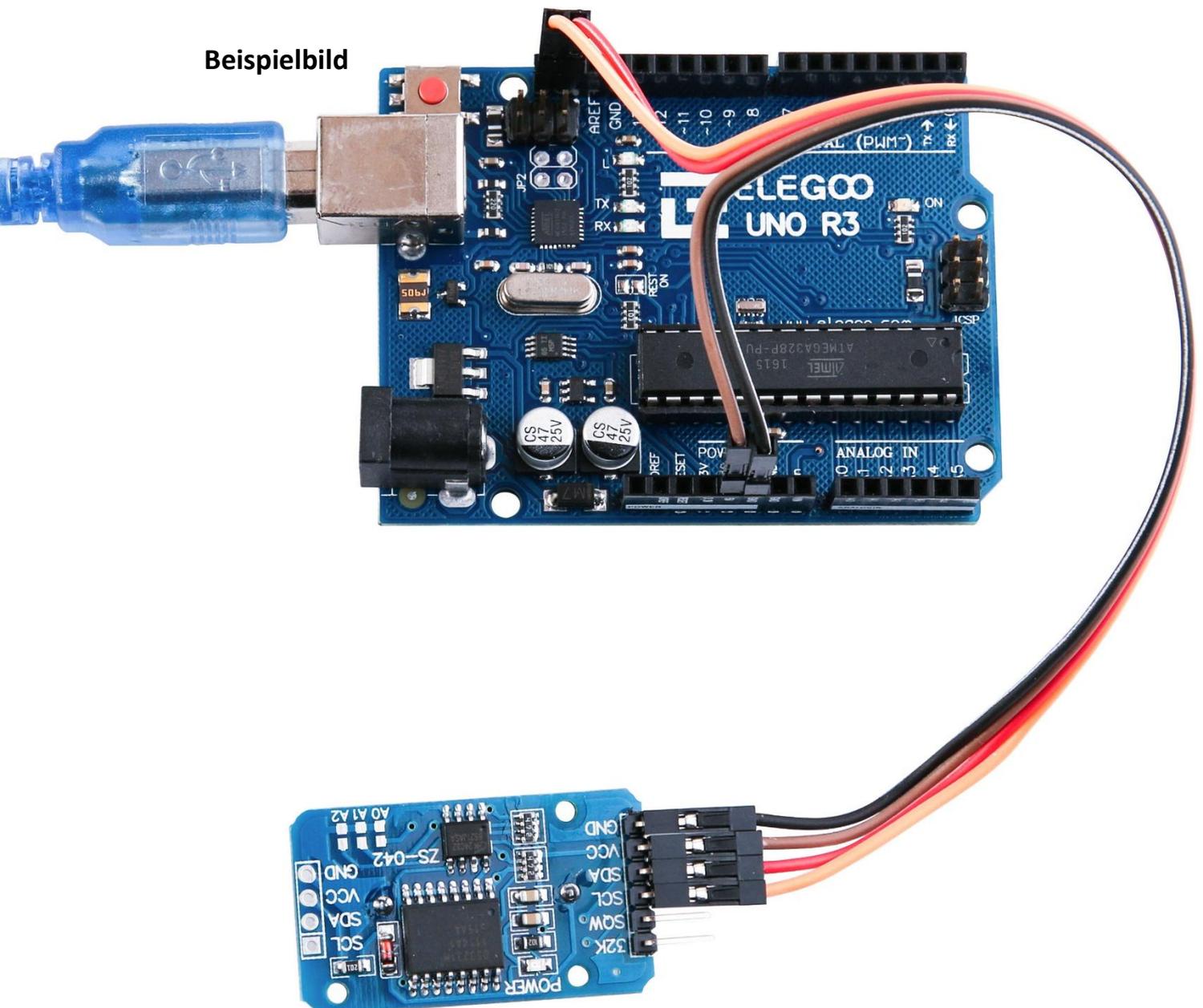
Ignorieren Sie den *32K* und den *SQW* Pin, sie werden nicht benötigt. Verbinden sie den *SCL*-Pin mit dem *SCL*-Pin Ihres UNO R3 Boards und den *SDA*-Pin mit dem gleichnamigen *SDA*-Pina uf Ihrem Board. Der *VCC* Pin wird wie immer mit dem 5V-Pin und der *GND*-Anschluss mit dem *GND*-Pin des Boards verbunden.

Code

ANach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 19 Real Time Clock Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 nocheinmal an.

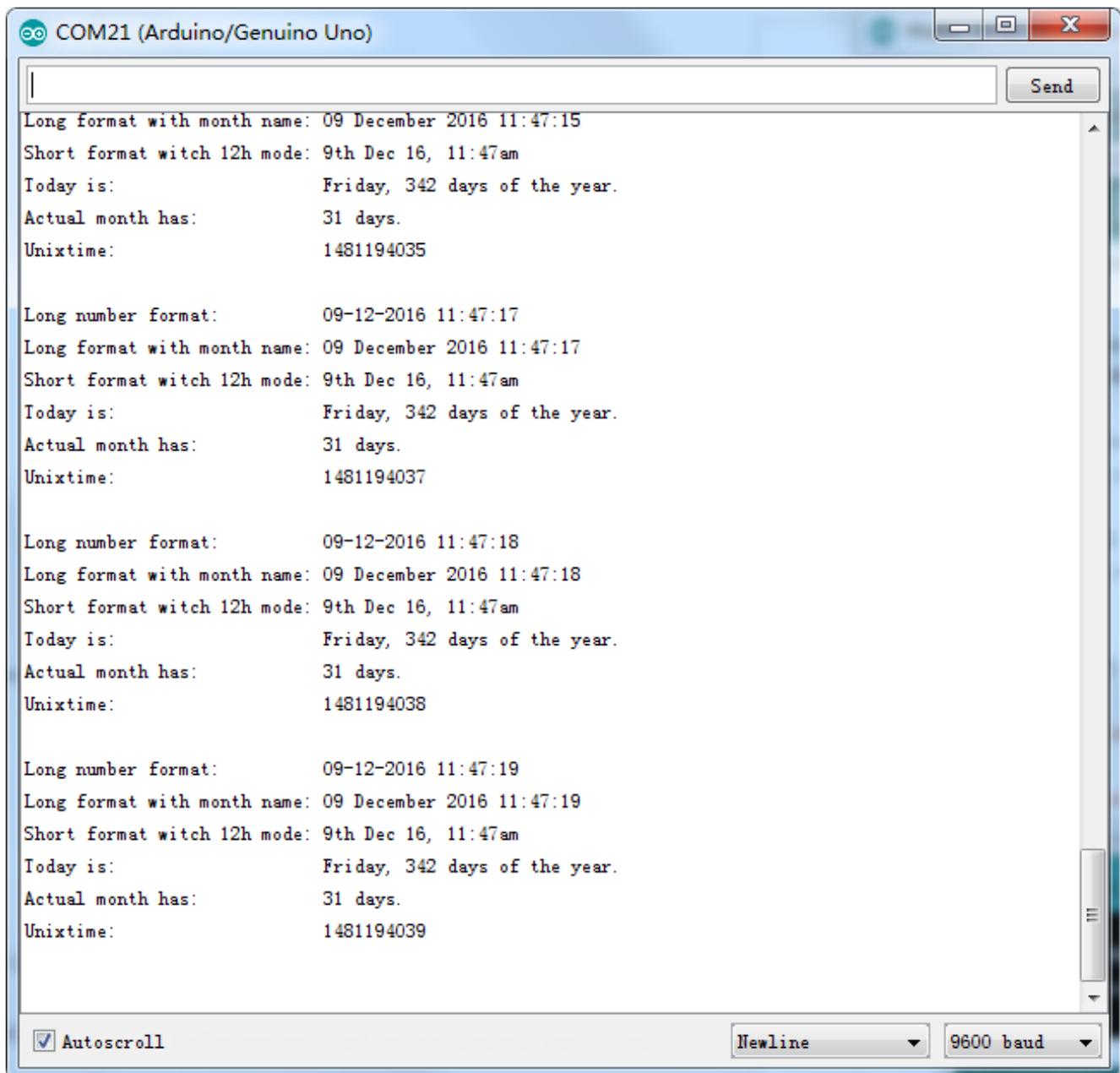
Bevor Sie diesen Sketch hochladen können, müssen Sie die „DS3231“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.

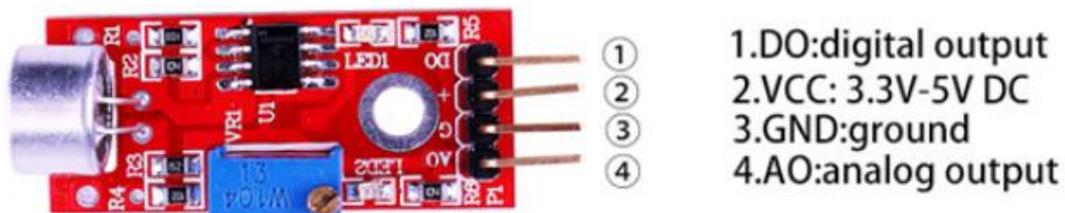


Lektion20:Soundsensormodul

Übersicht

In dieser Lektion lernen Sie, wie man ein Soundsensormodul benutzt. Dieses Modul hat zwei Ausgänge: Einen analogen Ausgang (AO) und ein Ausgang mit dem direkten Signal vom Mikrofon (DO). Wenn das Lautstärkelevel eine gewisse Schwelle überschreitet bzw unterschreitet, wird ein HIGH bzw ein LOW Signal ausgegeben. Die Lautstärkeschwelle kann dabei über das Potentiometer eingestellt werden. Um sicher gehen zu können, dass das Mikrofon Ihre Stimme erkennt, versuchen Sie die Empfindlichkeit durch Drehen des blauen Potentiometers am Modul anzupassen.

Um die Genauigkeit zu erhöhen, führt das Modul 10 Messungen hintereinander aus.



Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x Soundsensormodul
- (4) x W-M Kabel (Weiblich zu MännlichDuPontJumper Kabel)

Einführung in die Komponenten

Mikrofon

Wandler sind Geräte, die Energie von einer Form in eine andere umwandeln. Ein Mikrofon ist ein Wandler, der Sound bzw Schallwellen in elektrische Signale umwandelt.Es arbeitet gegensätzlich zu Lautsprechern. Mikrofone gibt es in verschiedenen Formen und Größen.Je nach Ausführung nutzt ein Mikrofon verschiedene Technologien, um Töne in elektrische Signale umzuwandeln. Bei diesem Modul handelt es sich um ein elektrisches Kondensatormikrofon, welches

weit verbreitet in Smartphones, Laptops usw. vorkommt.

Wie der Name suggeriert, besteht ein Kondensatormikrofon aus einem Plattenkondensator und beruht auf dem Prinzip der variablen Kapazität. Es besteht aus zwei Platten: einer fixierten Platte und einer beweglichen. Zwischen den beiden Platten befindet sich ein kleiner Abstand. Ein elektrisches Potential lädt die fixierte Platte auf. Wenn jetzt Schallwellen auf die bewegliche Platte stoßen und diese anfängt sich zu bewegen, ändert sich die Kapazität zwischen den Platten, was dazu führt, dass der fließende Strom je nach Intensität und Art des Tons variiert.



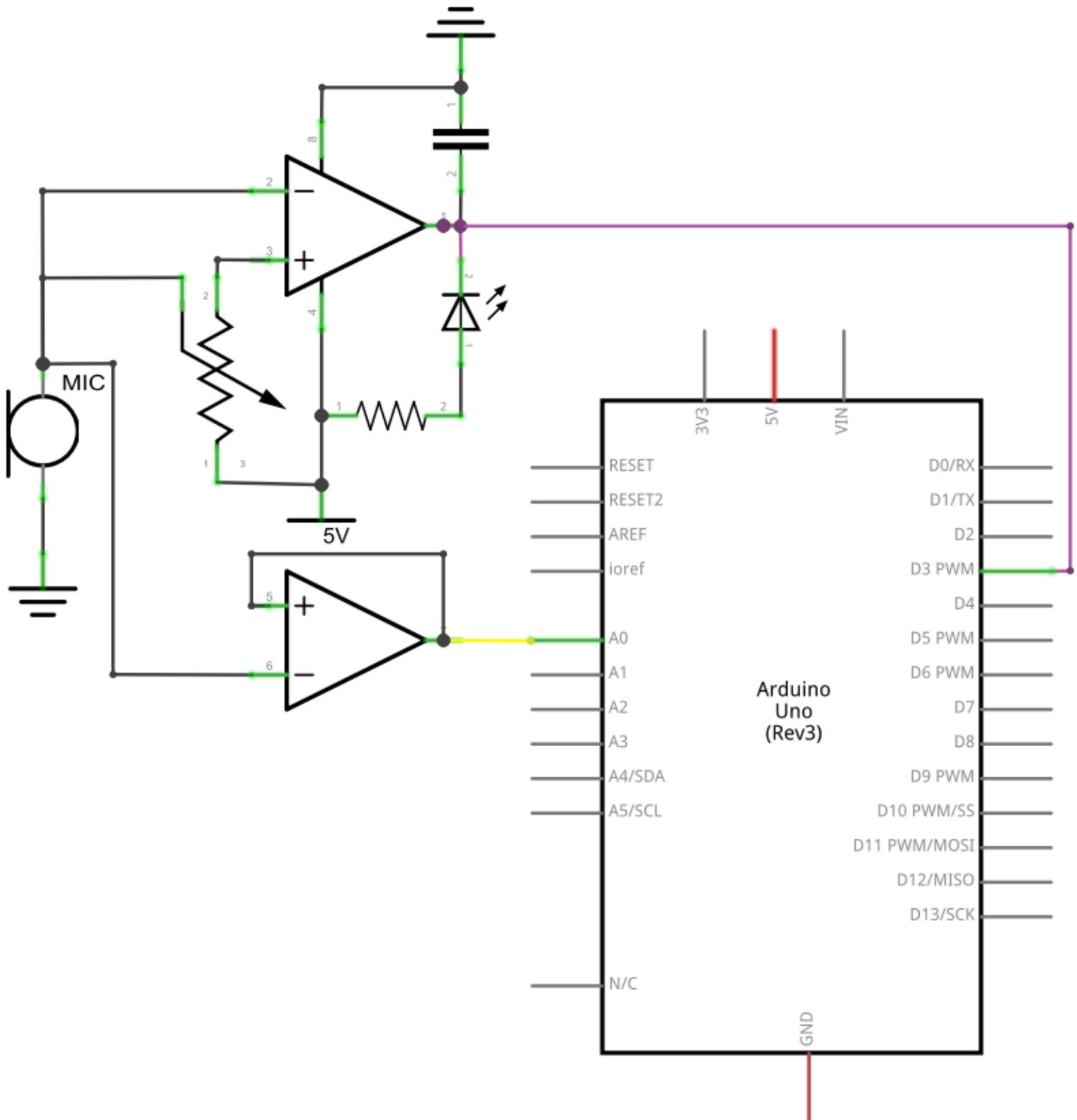
Diese Kondensatormikrofone werden in vielen Schaltkreisen verbaut, um Töne oder Luftschwingungen zu erkennen, die anschließend in elektrische Signale zur Weiterverarbeitung umgewandelt werden. Die zwei Anschlüsse im Bild oben dienen zur Verbindung mit dem Rest des Schaltkreises.



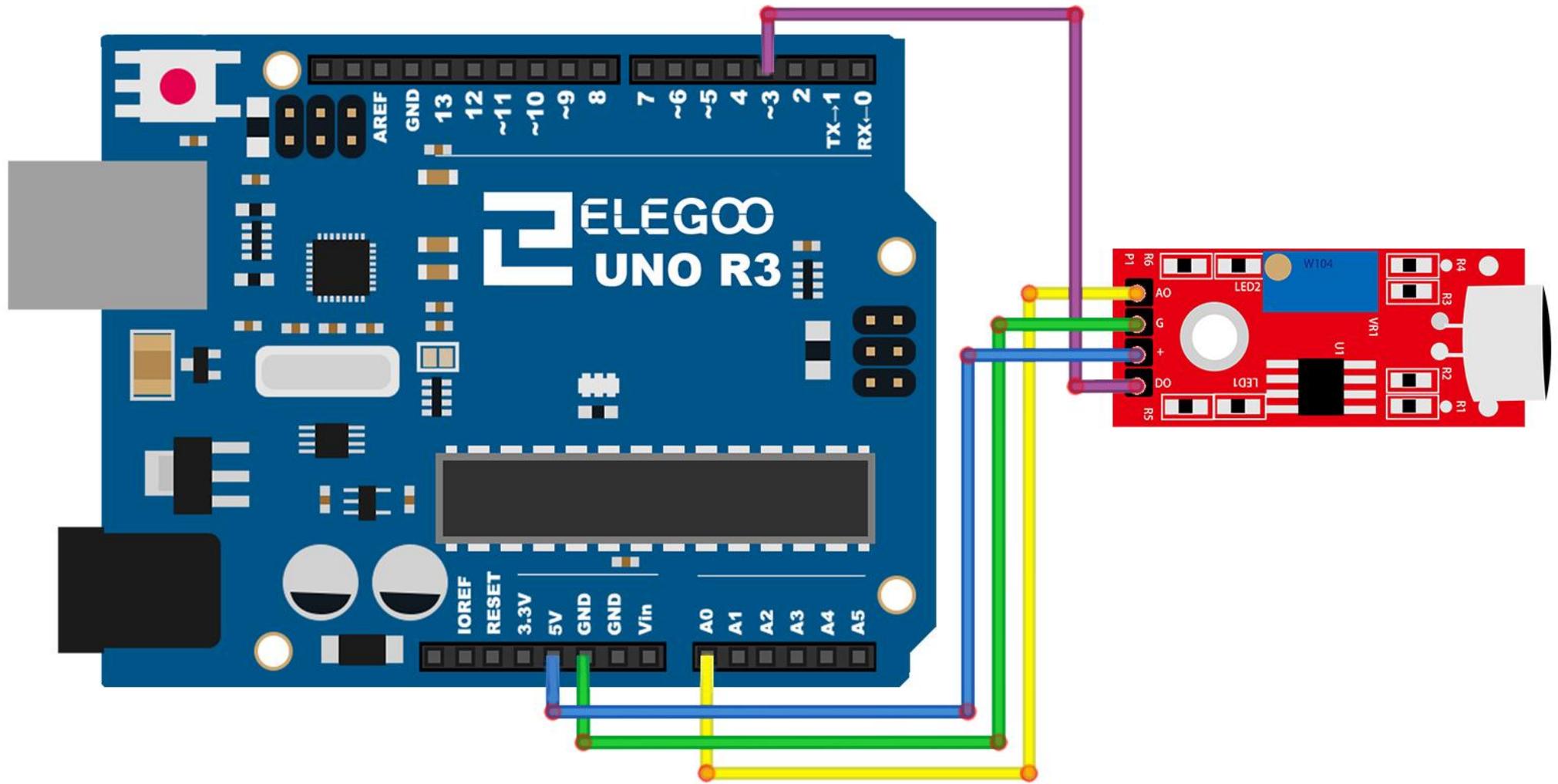
Ein festes leitendes Metallgehäuse umschließt die verschiedenen Komponenten des Mikrofons. Die Oberseite ist mit einem durchlässigen Material beklebt, das als

Staubfilter dient. Die Soundsignale / Schallwellen durchdringen den Filter und bringen die bewegliche Platte zum schwingen.

Verbindungsschema



Schaltplan

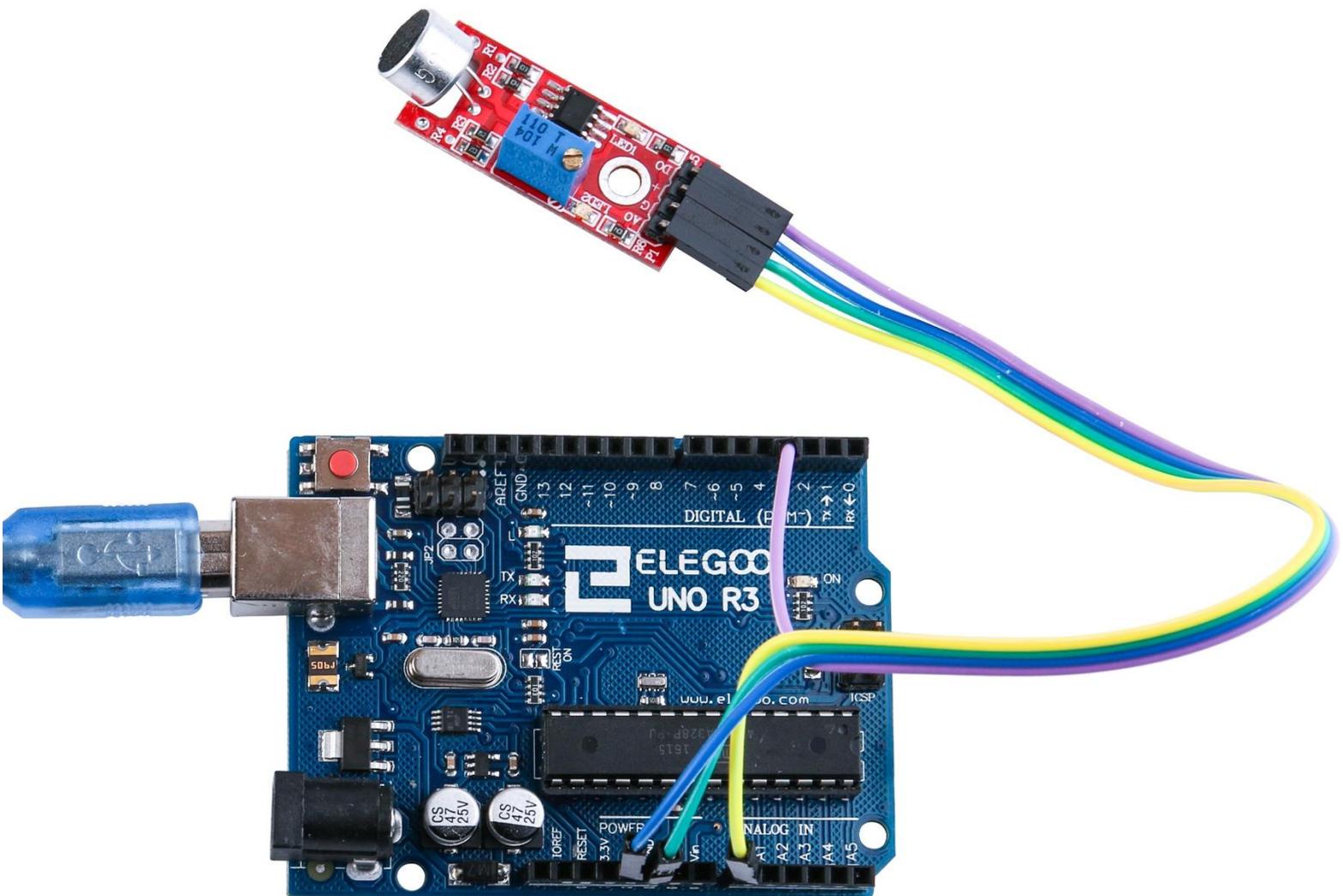


Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 20 Sound Sensor Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte [Lektion 2](#) noch einmal an.

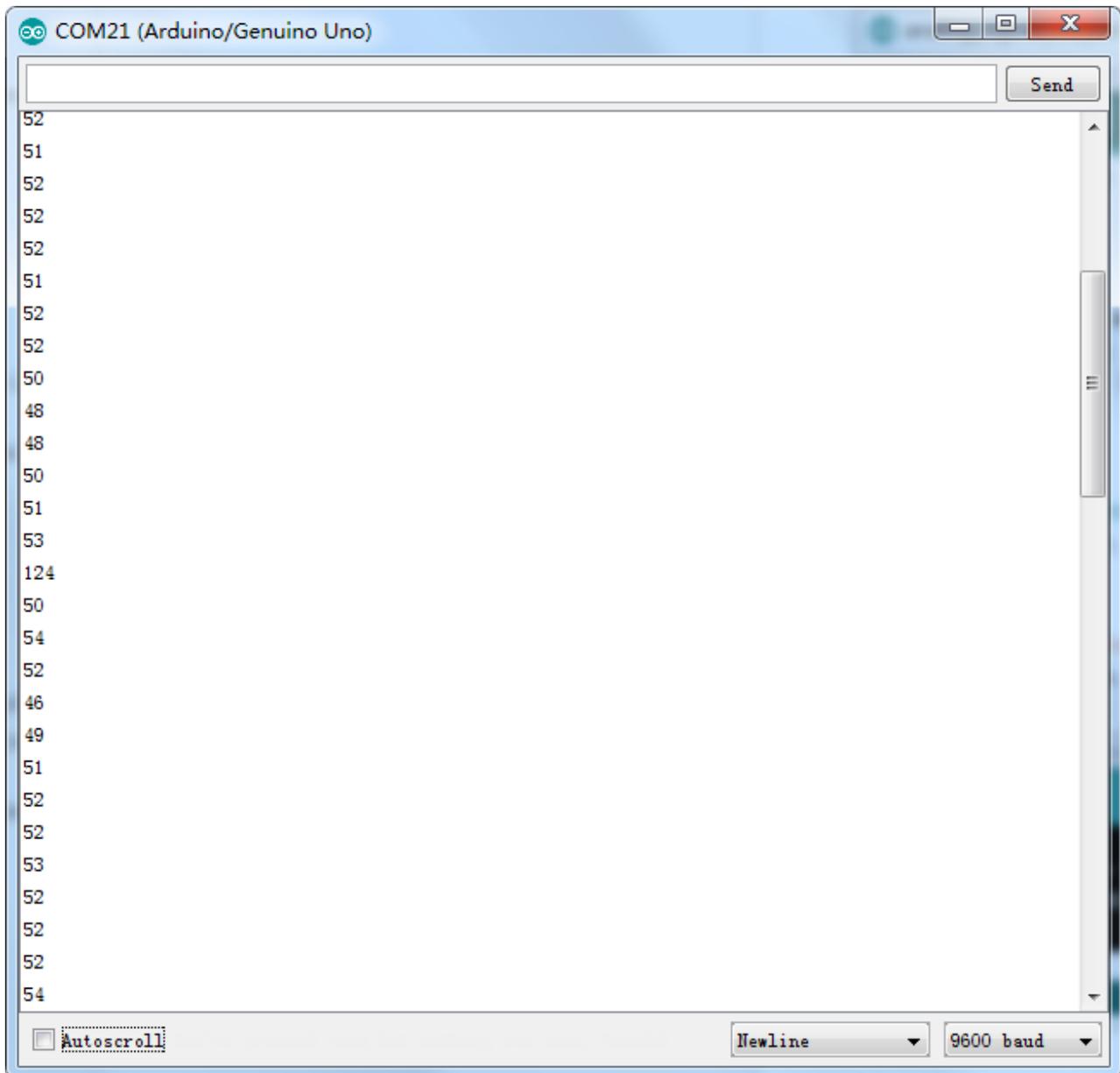
Dieses Modul kann sowohl als digitaler als auch als analoger Ausgang dienen, wobei je nach Modus der Code angepasst werden muss. Wir stellen Ihnen Codebeispiele für beide Modi bereit: `digital_signal_output` und `analog_signal_output`. Der Code `digital_signal_output` gibt, wenn das Lautstärkelevel eine gewisse Schwelle überschreitet, ein HIGH-Signal auf dem digitalen *Pin 11* des Arduinos aus und lässt gleichzeitig die L-LED aufleuchten. Der Schwellwert kann, wie oben beschrieben, durch drehen am Potentiometer verstellt werden. Der Code `analog_signal_output` liest die analogen Werte des Moduls und gibt sie direkt auf dem Seriellen Monitor aus. Die Empfindlichkeit kann hierbei auf gleiche Weise geändert werden.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



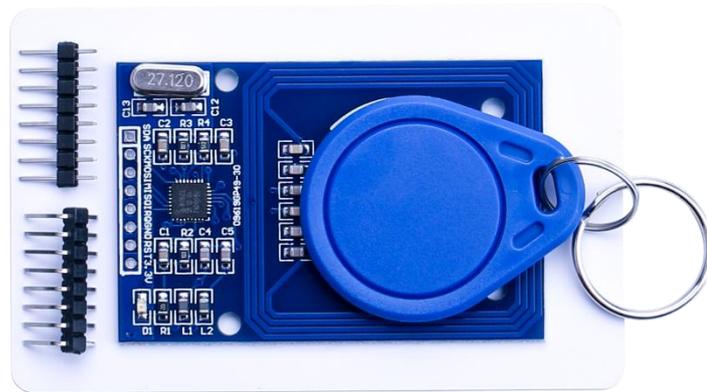
Lektion21:RC522 RFID-Modul

Übersicht

In dieser Lektion lernen Sie, wie man das RC522 RFID-Modul mit dem UNO R3 verbindet. Dieses Modul benutzt die SPI- (Serial Peripheral Interface) Schnittstelle, um mit Mikrocontrollern wie dem Raspberry Pi, Arduino oder Beagle Board zu kommunizieren.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x RC522 RFID-Modul
- (7) x W-M Kabel (Weiblich zu Männlich
DuPont JumperKabel)



Einführung in die Komponenten

RC522

Das MFRC522 ist ein hochintegriertes Lese-/Schreibgerät zur kontaktlosen Kommunikation bei 13,56 MHz. Der MFRC522-Chip unterstützt ISO 14443A und den MIFARE® Modus.

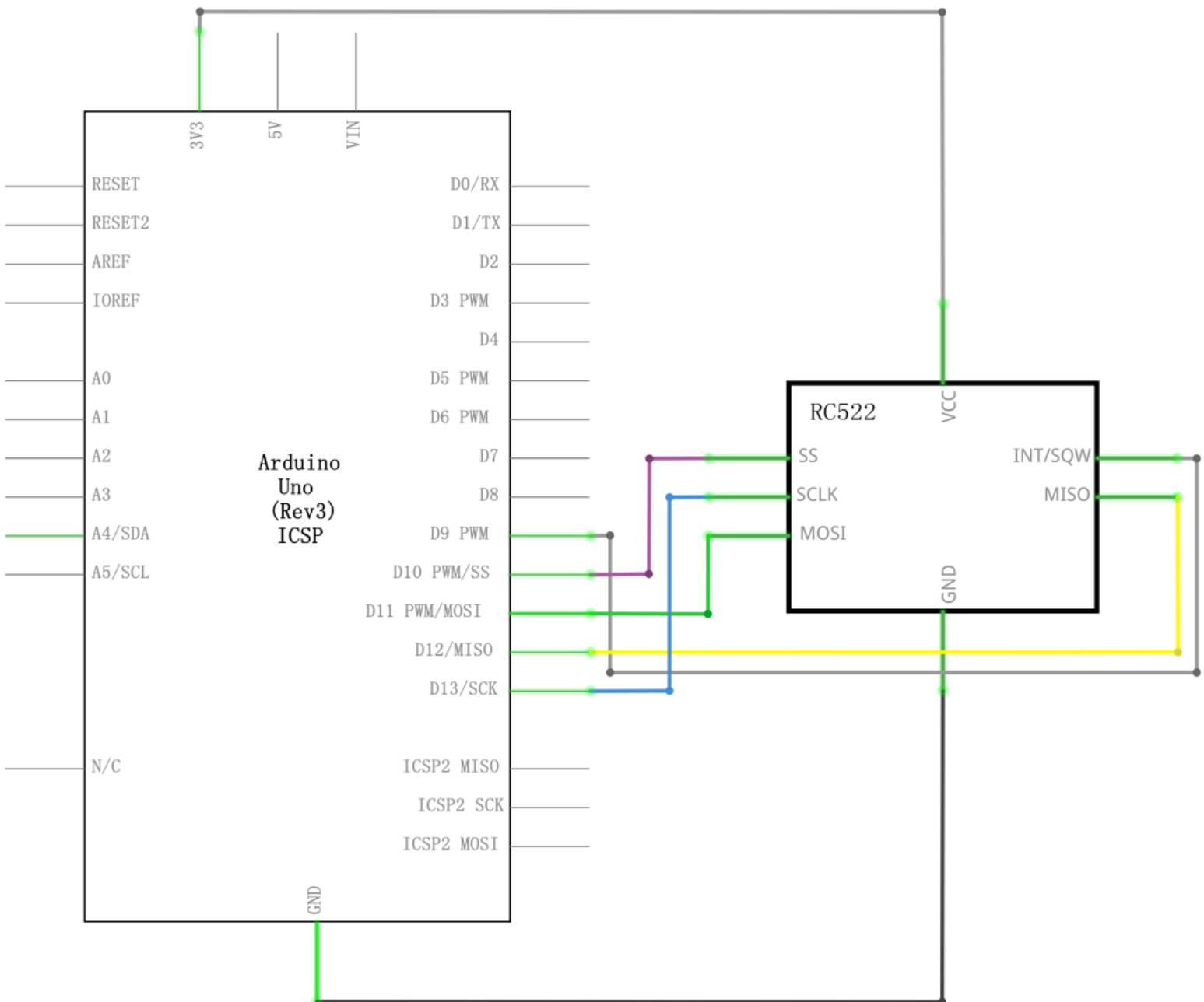
Der Sendeteil des MFRC522-Chips kann eine Antenne zur Kommunikation mit ISO/IEC 14443A bzw. MIFARE® Karten und Transpondern betreiben, ohne dass diese Karten eine externe Stromversorgung benötigen. Der Empfangsteil beinhaltet einen Schaltkreis zur Demodulierung und Entschlüsselung des Signals von ISO/IEC 14443A/MIFARE® kompatiblen Karten und Transpondern. Der digitale Teil übernimmt die komplette Modulierung und Fehlererkennung (Parity & CRC). Es werden MIFARE® Classic (z.B. MIFARE® Standard) Produkte unterstützt. Der Chip unterstützt kontaktlose Kommunikation mit Übertragungsraten mit bis zu 848 kBit/s in beide Richtungen.

Verschiedene Schnittstellen sind verfügbar:

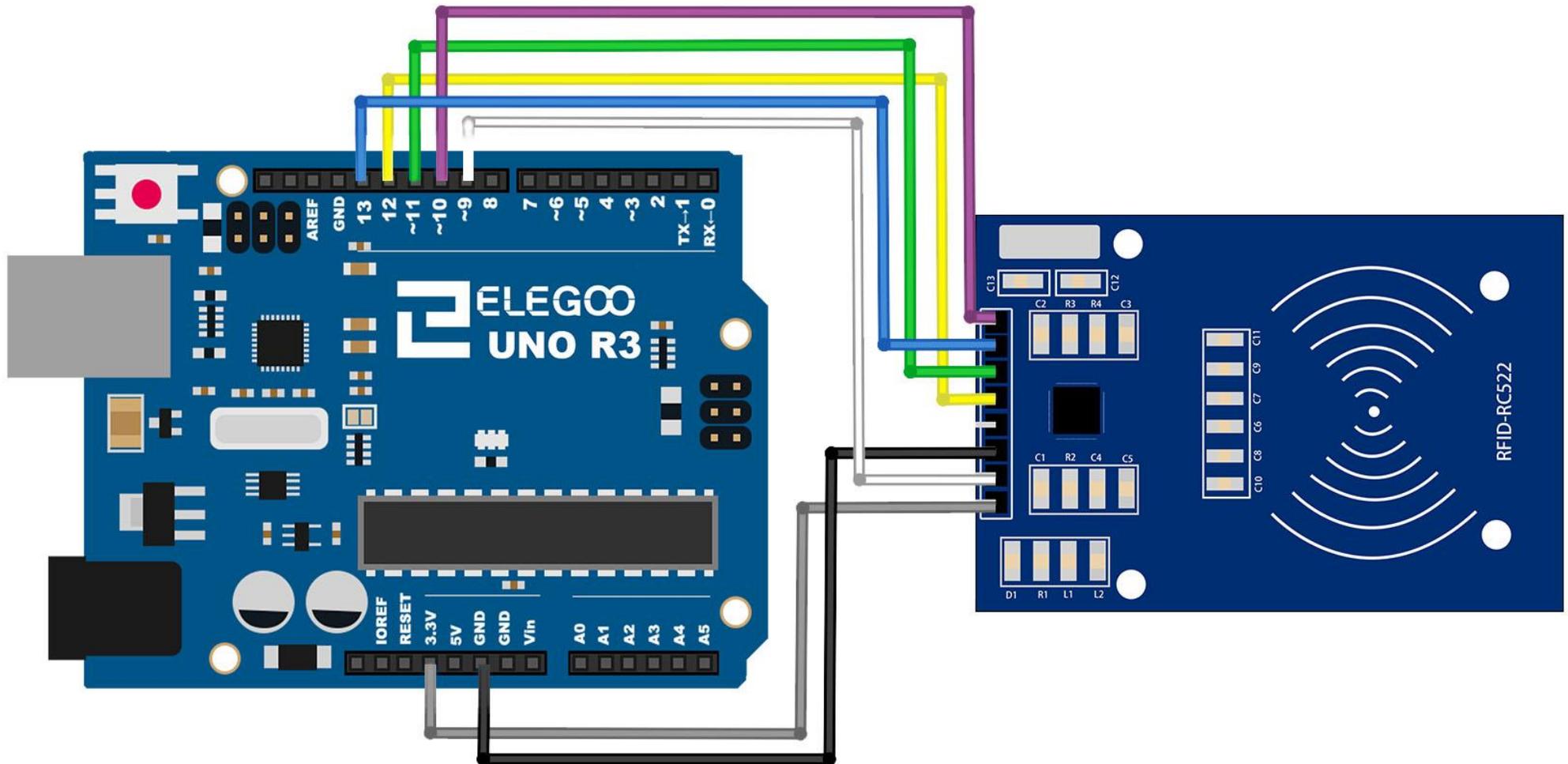
- SPI Schnittstelle
- Serielles UART (ähnlich wie RS232)
- I2C Schnittstelle.

Die Abbildung unten zeigt ein typisches Schalt diagram zur Verbindung des MFRC522s unter Verwendung einer externen Antenne.

Verbindungsschema



Schaltplan



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 21 RC522 RFID Module“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an. Bevor Sie diesen Sketch hochladen können, müssen Sie die „rfid“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

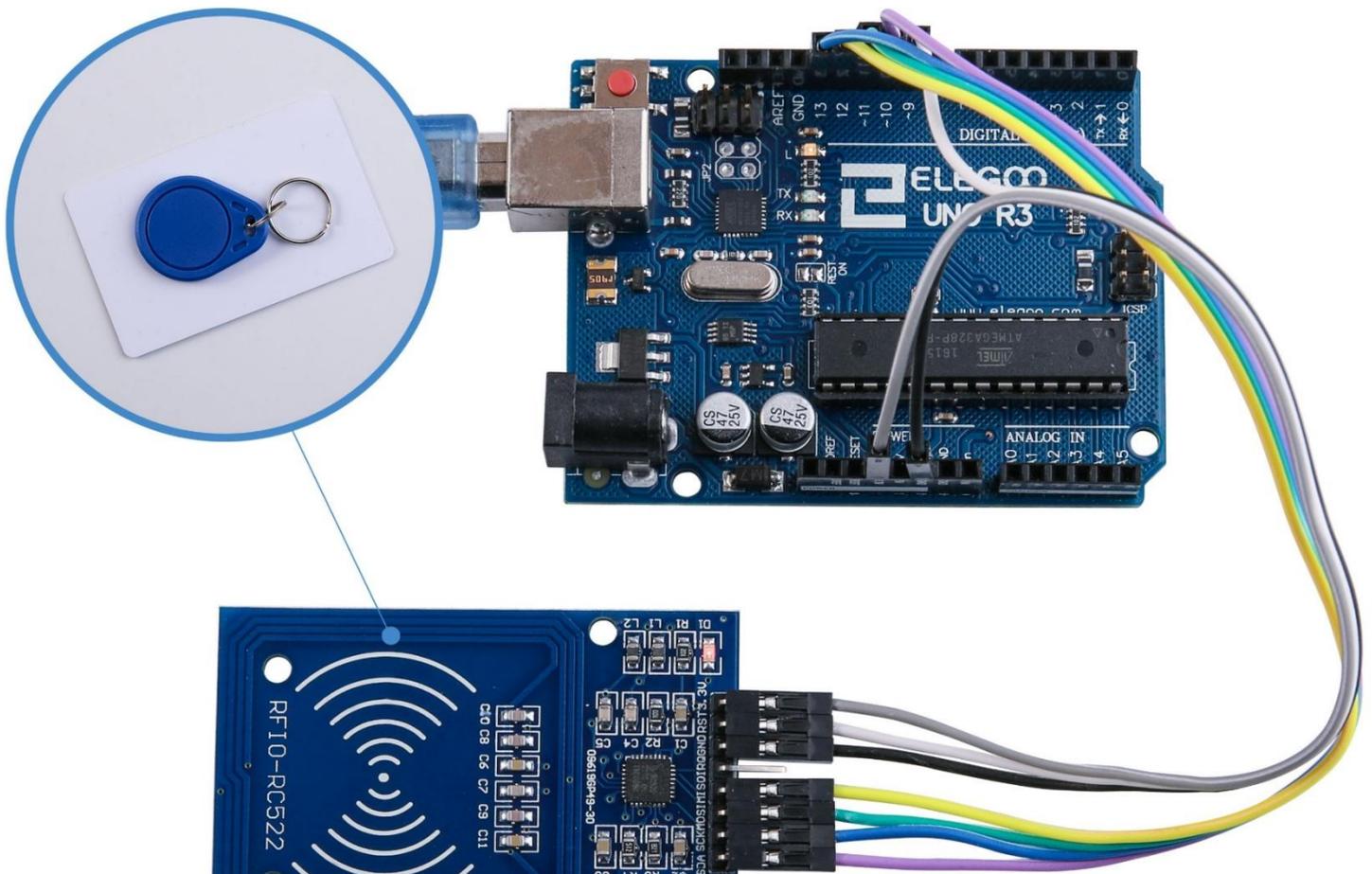
	MFRC522	Arduino	Arduino	Arduino	Arduino	Arduino
	Reader/PCD	Uno	Mega	Nano v3	Leonardo/Micro	Pro Micro
Signal	Pin	Pin	Pin	Pin	Pin	Pin
RSI/Reset	RSI	9	5	D9	RESEI/ICSP-5	RSI
SPI SS	SDA(SS)	10	53	D10	10	10
SPI MOSI	MOSI	11 / ICSP-4	51	D11	ICSP-4	16
SPI MISO	MISO	12 / ICSP-1	50	D12	ICSP-1	14
SPI SCK	SCK	13 / ICSP-3	52	D13	ICSP-3	15

```
#define RST_PIN 9 // Configurable, see typical pin layout above
```

```
#define SS_PIN 10 // Configurable, see typical pin layout above
```

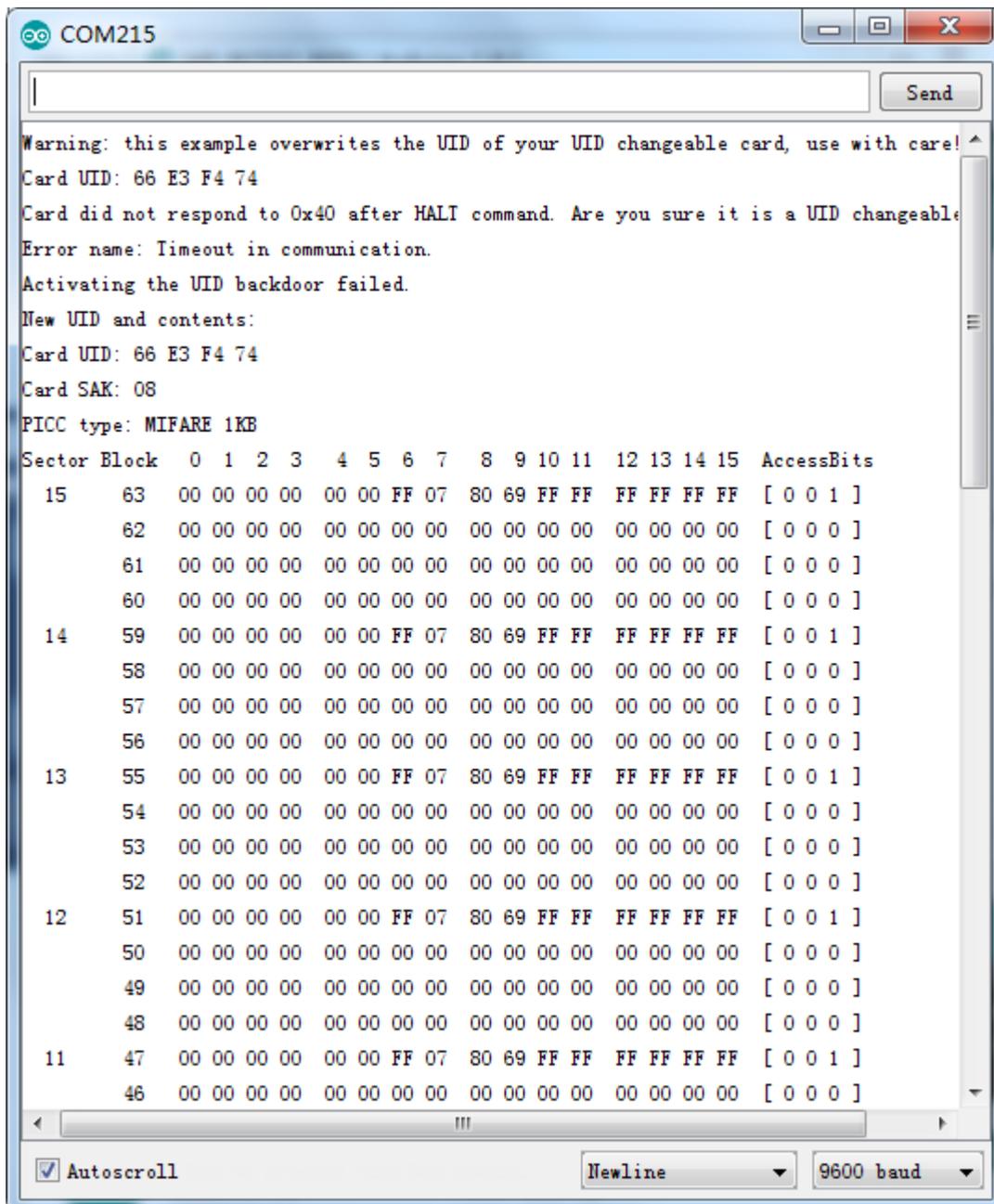
Die SPI-Pins befinden sich je nach Modul an unterschiedlicher Stelle und Sie müssen eventuell im Code kleine Anpassungen an der Pinbelegung vornehmen.

Beispielbild



Öffnen Sie den Seriellen Monitor:

Klicken Sie auf das Symbol des Seriellen Monitors, um ihn zu öffnen. Die grundlegenden Informationen zum Seriellen Monitor haben Sie in Lektion 1 kennengelernt.



Warning: this example overwrites the UID of your UID changeable card, use with care!

Card UID: 66 E3 F4 74

Card did not respond to 0x40 after HALT command. Are you sure it is a UID changeable?

Error name: Timeout in communication.

Activating the UID backdoor failed.

New UID and contents:

Card UID: 66 E3 F4 74

Card SAK: 08

PICC type: MIFARE 1KB

Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	AccessBits
15	63	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	62	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
14	60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	59	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	58	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	57	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
13	56	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	55	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	54	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	53	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
12	52	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	51	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	49	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
11	48	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	47	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	46	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]

Autoscroll Newline 9600 baud

Lektion 22: LCD Display

Übersicht

In dieser Lektion lernen Sie, wie man ein alphanumerisches LC-Display verbindet. Das Display hat eine LED-Hintergrundbeleuchtung und kann auf zwei Reihen je bis zu 16 Zeichen darstellen. Man kann die viereckigen Zeichenfelder sowie die Pixel, mit denen die Zeichen dargestellt werden, bei genauerer Betrachtung erkennen. Das Display zeigt die Zeichen weiß auf blauem Hintergrund an und ist dazu gedacht, um Text anzuzeigen. In dieser Lektion werden wir den Arduino Beispielsketch der LCD-Bibliothek benutzen. In der nächsten Lektion erweitern wir dies, indem wir auf dem Display durch das Benutzen von Sensoren die aktuelle Temperatur anzeigen lassen.



Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x LCD1602 Modul
- (1) x Potentiometer (10K)
- (1) x 830 PunkteBreadboard
- (16) x M-M Kabel (Männlich zu Männlich DuPont JumperKabel)

Einführung in die Komponenten

LCD1602

Pinbelegung des LCD1602:

VSS: Verbindung zu GND

VDD: Verbindung zur +5V Stromversorgung

VO: Anpassen des Kontrasts der Anzeige

RS: Kontrolliert, an welcher Stelle des Speichers Daten geschrieben werden sollen. Dabei können Daten ins Datenregister geschrieben werden, was den anzuzeigenden Text beinhaltet, oder ins Befehlsregister, wo die Befehle, die der Controller ausführen soll, gespeichert werden.

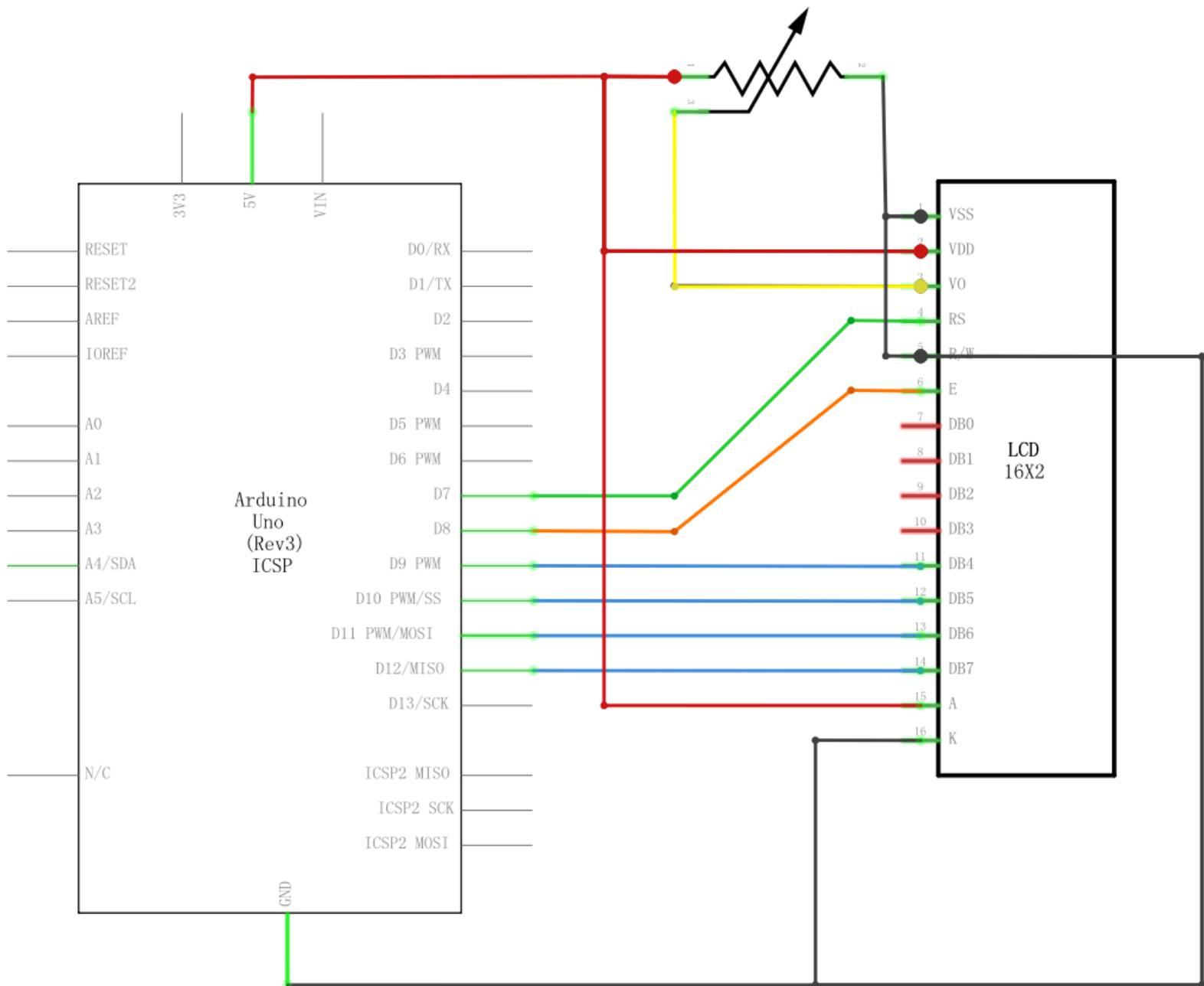
R/W: Umschaltung zwischen Lese- und Schreibmodus

E: Pin, der bei einem LOW-Signal vordefinierte Instruktionen ausführt.

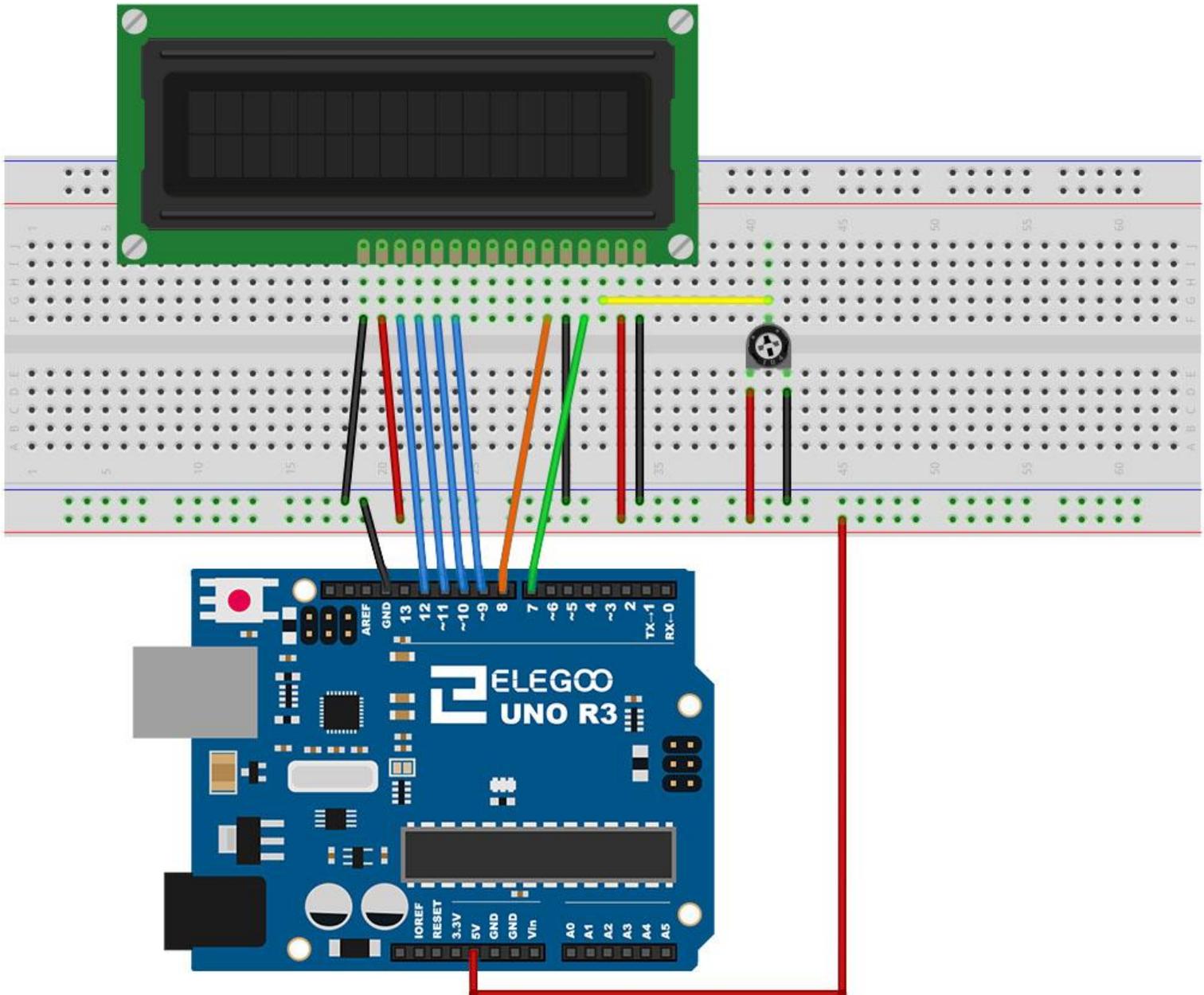
D0-D7: Übertragung der Daten (Lesen & Schreiben)

A and K: Anpassen der LED-Hintergrundbeleuchtung

Verbindungsschema



Schaltplan



Das LCD benötigt sechs Pins am Arduino, eingestellt als digitale Ausgänge. Zusätzlich müssen natürlich +5V und GND zur Stromversorgung verbunden werden.

Es müssen einige weitere Anschlüsse verbunden werden. Das Display sollte am oberen Ende des Breadboards ausgerichtet werden, damit es später einfacher fällt die Pins zu identifizieren. Vergessen Sie nicht das lange gelbe Kabel, das das Potentiometer mit dem Pin 3 des Displays zum Einstellen des Kontrasts verbindet. Es kann sein, dass Sie Ihr Display ohne bereits angelötete Headerpins bekommen haben. Dann folgen Sie bitte den Hinweisen in der nächsten Sektion.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 22 LCD Display*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „*LiquidCrystal*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Laden Sie den Code auf Ihr Arduino Board hoch und Sie sollte folgende Nachricht auf dem Display sehen: „*Hello, World!*“, gefolgt von einer Reihe an Zahlen, die von 0 hochzählen.

Die erste wichtige Zeile im Sketch ist:

```
#include <LiquidCrystal.h>
```

Dies meldet der IDE, dass wir die LiquidCrystal Bibliothek benutzen möchten.

Die nächste Zeile, die wir ändern müssen, stellt ein, welcher Pin vom Arduino mit welchem Pin am Display verbunden ist.

```
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
```

Nach dem Hochladen des Codes stellen Sie bitte sicher, dass die Hintergrundbeleuchtung des LCDs aufleuchtet und drehen Sie solange am Potentiometer, bis der Kontrast richtig eingestellt ist und Sie den Text sehen können.

In der setup-Funktion sind zwei Befehle:

```
lcd.begin(16, 2);
```

```
lcd.print("Hello, World!");
```

Der erste Befehl teilt der Bibliothek mit, wie viele Spalten und wie viele Zeilen das Display hat. Der zweite Befehl bestimmt die Nachricht, die auf dem Display angezeigt werden soll.

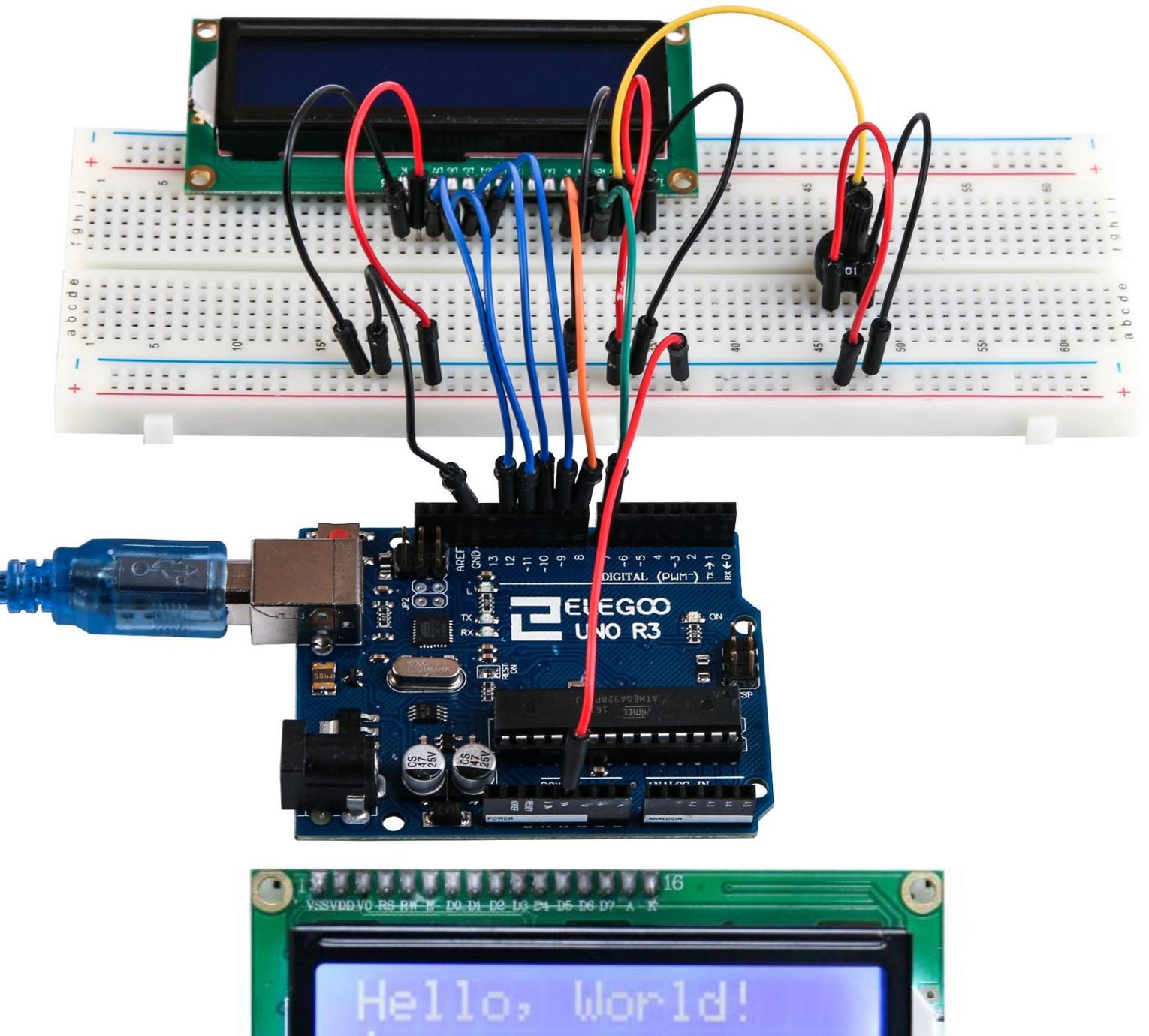
In der loop-Funktion sind ebenfalls zwei Befehle:

```
lcd.setCursor(0, 1);
```

```
lcd.print(millis()/1000);
```

Der erste Befehl setzt die Cursor Position (wo der nächste Text erscheinen soll) zu Spalte 1 in Zeile 2. Die Anzahl der Spalten und Zeilen fängt bei 0 an zu zählen, nicht bei 1. Wenn man die erste Zeile nutzen will, muss man im Code also „0“ benutzen. Der zweite Befehl gibt die Anzahl der Millisekunden, die vergangen sind, seit der Arduino zum letzten Mal resettet wurde, in Zeile 2 auf dem Display aus.

Beispielbild



Lektion 23:Thermistor

Übersicht

In dieserLektion werden wir das LCD benutzen, um die Temperatur anzuzeigen.

Benötigte Bauteile:

(1) x Elegoo UNO R3

(1) x LCD1602 Modul

(1) x 10KOhm Widerstand

(1) x Thermistor

(1) x Potentiometer

(1) x 830 PunkteBreadboard

(18) x M-M Kabel (Männlich zu MännlichDuPont JumperKabel)

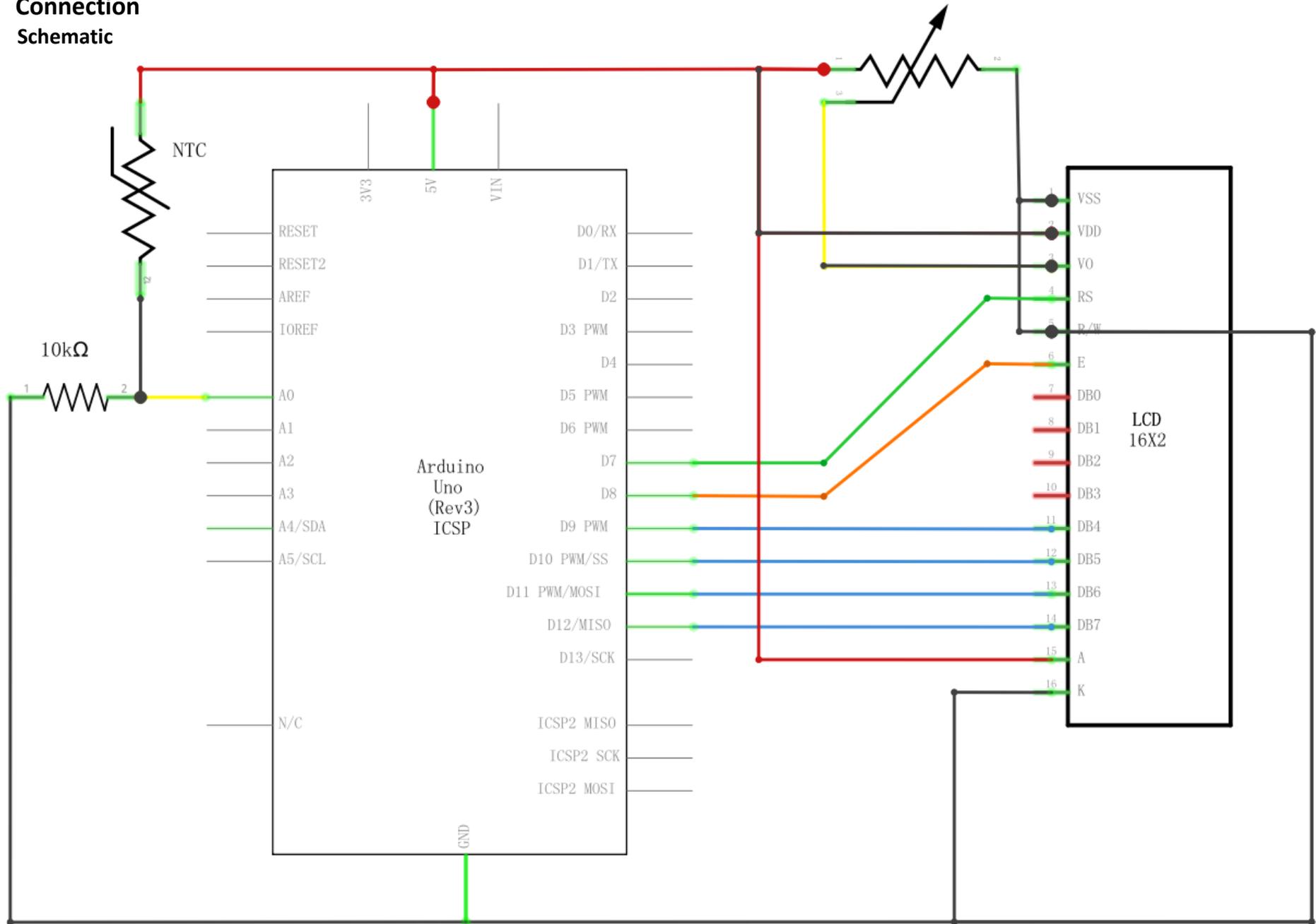
Einführung in die Komponenten

Thermistor

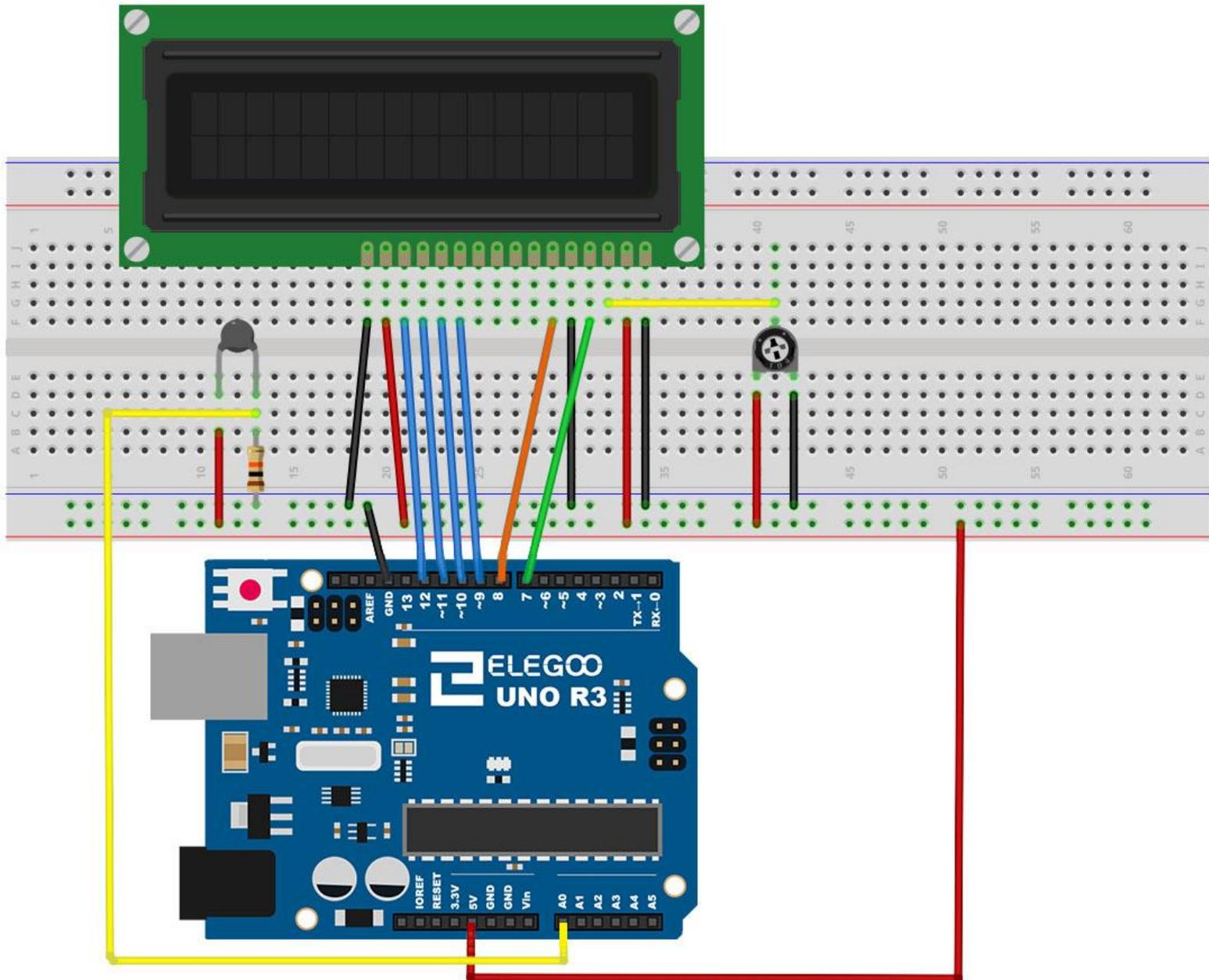
Ein Thermistor (thermal resistor) ist ein thermischer Widerstand – ein Widerstand, der seinen Wert mit der Temperatur ändert. Technisch gesehen sind alle Widerstände Thermistoren, ihr Widerstand ändert sich leicht abhängig von der Temperatur. Aber die Widerstandsänderung ist nur minimal und schwer zu messen. Thermistoren sind so gebaut, dass sie ihren Widerstand stark abhängig von der Temperatur ändern, sodass ein 1-Grad-Unterschied mehr als 100 Ohm Widerstandsunterschied bedeuten können!

Es gibt zwei Arten von Thermistoren:*NTC* (Negativer Temperaturkoeffizient) und *PTC* (Positiver Temperaturkoeffizient) -Thermistoren. Normalerweise werden *NTC*-Thermistoren zur Temperaturmessung genutzt. *PTCs* werden oft als wiederverwendbare Sicherungen benutzt. Bei steigender Temperatur steigt ebenfalls der Widerstand an. Da durch steigenden Strom die Temperatur steigt, begrenzt der *PTC*-Thermistor somit durch den Widerstand den Strom wieder, bis normale Temperaturen erreicht sind. Dieser Effekt eignet sich gut zum Sichern von Schaltkreisen.

Connection Schematic



Schaltplan



Das Breadboard Layout basiert auf dem Layout aus Lektion 22, sodass Sie nicht viel ändern müssen, wenn sie seitdem nichts angepasst haben.

Es gibt ein paar Jumper Kabel in der Nähe des Potentiometers, die im Vergleich zur letzten Lektion leicht verschoben wurden.

Der 10 kΩ Widerstand und der Thermistor wurden dem Schaltkreis neu hinzugefügt.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 23 Thermometer“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „LiquidCrystal“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Der Sketch basiert ebenfalls auf dem Sketch aus Lektion 22. Laden sie den neuen Sketch auf Ihren Arduino und die aktuelle Temperatur wird auf dem Display angezeigt. Wenn Sie ihren Finger auf den Thermistor auflegen, sehen Sie, wie die Temperatur auf dem Display steigt.

Ein Kommentar über dem lcd Befehl ist nützlich, falls man später mal die Pin-Belegung anpassen will.

```
// BS E D4 D5 D6 D7  
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
```

In der loop-Funktion passieren nun zwei interessante Dinge. Zum einen muss das analoge Signal des Thermistors in die eigentliche Temperatur umgewandelt werden und als zweites muss die berechnete Temperatur dann auch angezeigt werden.

Zuerst aber schauen wir uns an, wie man die Temperatur berechnet:

```
int tempReading = analogRead(tempPin);  
double tempK = log(10000.0 * ((1024.0 / tempReading - 1)));  
tempK = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * tempK *  
tempK)) * tempK );  
float tempC = tempK - 273.15;
```

```
float tempF = (tempC * 9.0)/ 5.0 + 32.0;
```

Wechselnde Werte auf einem LC-Display anzuzeigen ist kompliziert. Das Hauptproblem ist, dass sich die Anzahl der Ziffern je nach Temperatur ändern kann. Wenn die Temperatur also beispielsweise von 101,50°C auf 99,00°C sinkt, dann kann es passieren, dass die zusätzliche Ziffer des ersten Wertes auf dem Display stehen bleibt, wenn vergessen wird sie zu überschreiben.

Um das zu verhindern, werden wir bei jeder Änderung die gesamte Zeile neu setzen.

```
lcd.setCursor(0, 0);
```

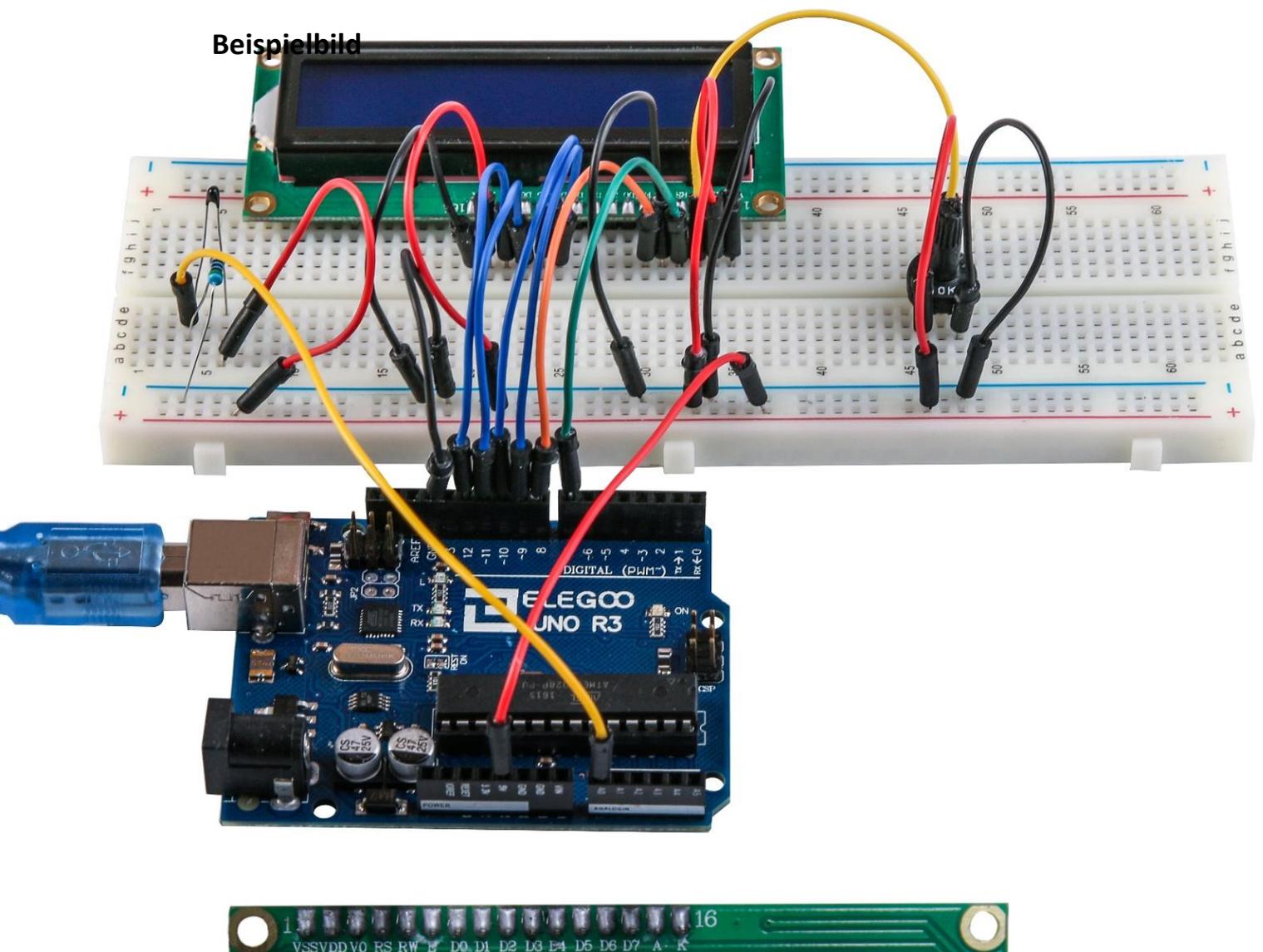
```
lcd.print("Temp      C  ");
```

```
lcd.setCursor(6, 0);
```

```
lcd.print(tempF);
```

(Der eher seltsame Kommentar soll Sie daran erinnern, dass das Display 16 Spalten hat). Es wird eine Vorlage als Zeichenfolge an das Display gesendet mit Leerzeichen, die später durch die Temperatur ersetzt werden. Danach wird der Cursor an den Start der Temperaturanzeige (Spalte 7) gesetzt und die Temperatur eingefügt.

Beispielbild



Lektion 24:Acht LEDsper 74HC595 ansteuern

Übersicht

In dieser Lektion lernen Sie, wie man acht rote LEDs mit dem UNO Board verbinden kann, ohne dafür 8 Pins belegen zu müssen.

Sie könnten die acht LEDs auch einzeln mit vorgeschaltetem Widerstand mit dem UNO Board verbinden, Ihnen würden dabei aber sehr schnell die Pins ausgehen. Wenn Sie nur wenige Komponenten mit Ihrem Board verbinden möchten, funktioniert das noch, aber meistens möchte man noch weitere Schalter, Sensoren, Servos usw. anschließen und früher oder später gehen Ihnen die Anschlüsse aus, weil alle belegt sind. Also statt alle LEDs einzeln anzuschließen, werden wir in dieser Lektion den 74HC595 Chip benutzen. Der 74HC595 ist ein Seriell zu Parallel Wandler. Der Chip hat acht Ausgänge (was für unser Vorhaben perfekt ist) und drei Eingänge, über die er mit Daten versorgt werden kann.

Durch den Chip lassen sich die LEDs ein bisschen langsamer ansteuern, so kann man die LEDs nur noch 500.000 pro Sekunde umschalten statt 8.000.000 pro Sekunde, wie es ohne Chip der Fall wäre. Das ist nichtsdestotrotz immer noch enorm schnell, sodass man den Unterschied mit dem Auge niemals wahrnehmen wird.



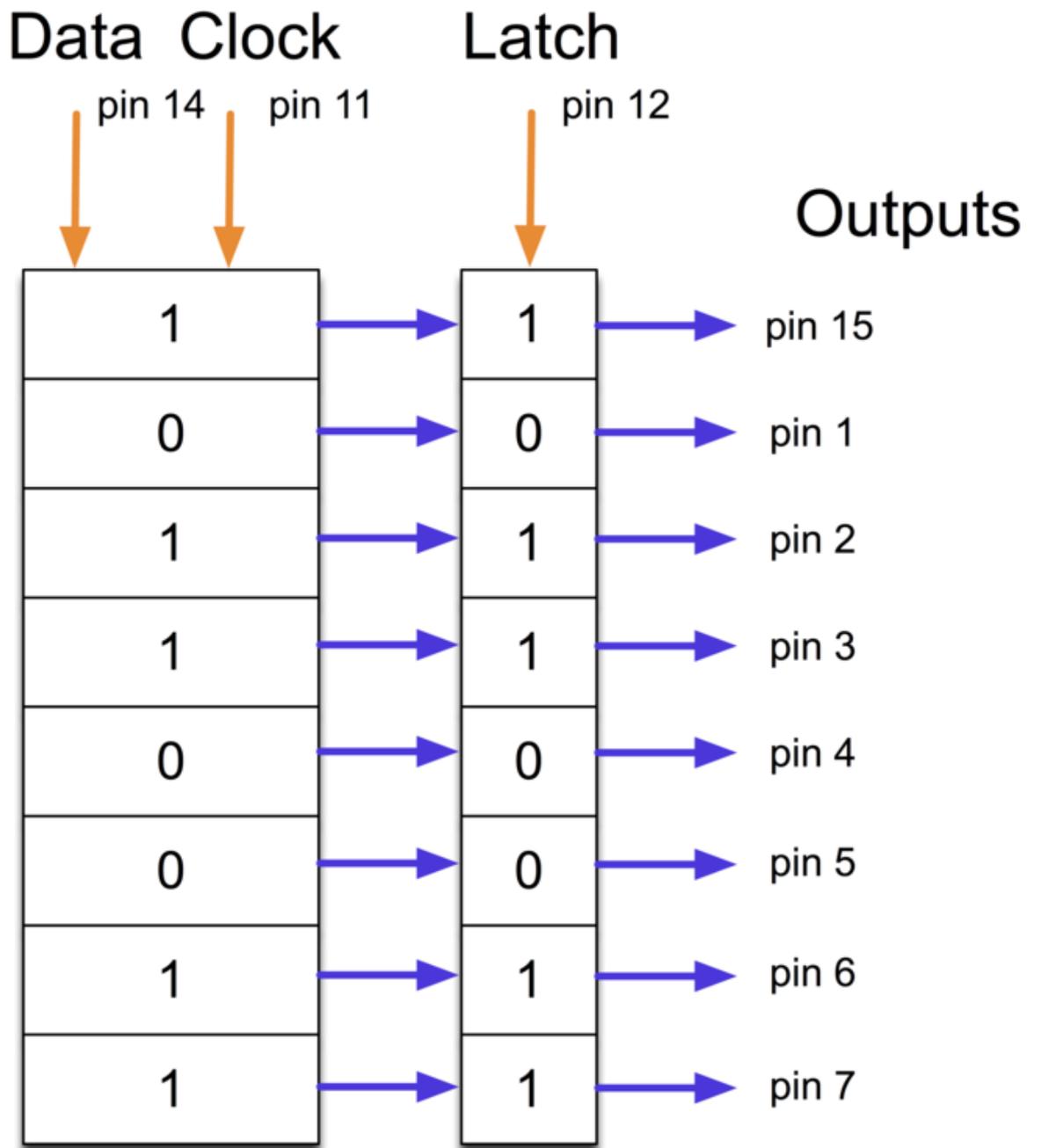
Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (8) x Rote LEDs
- (8) x 220 Ohm Widerstände
- (1) x 74HC595 IC
- (14) x M-M Kabel (Männlich zu MännlichDuPont JumperKabel)

Einführung in die Komponenten

74HC595 Schieberegister:

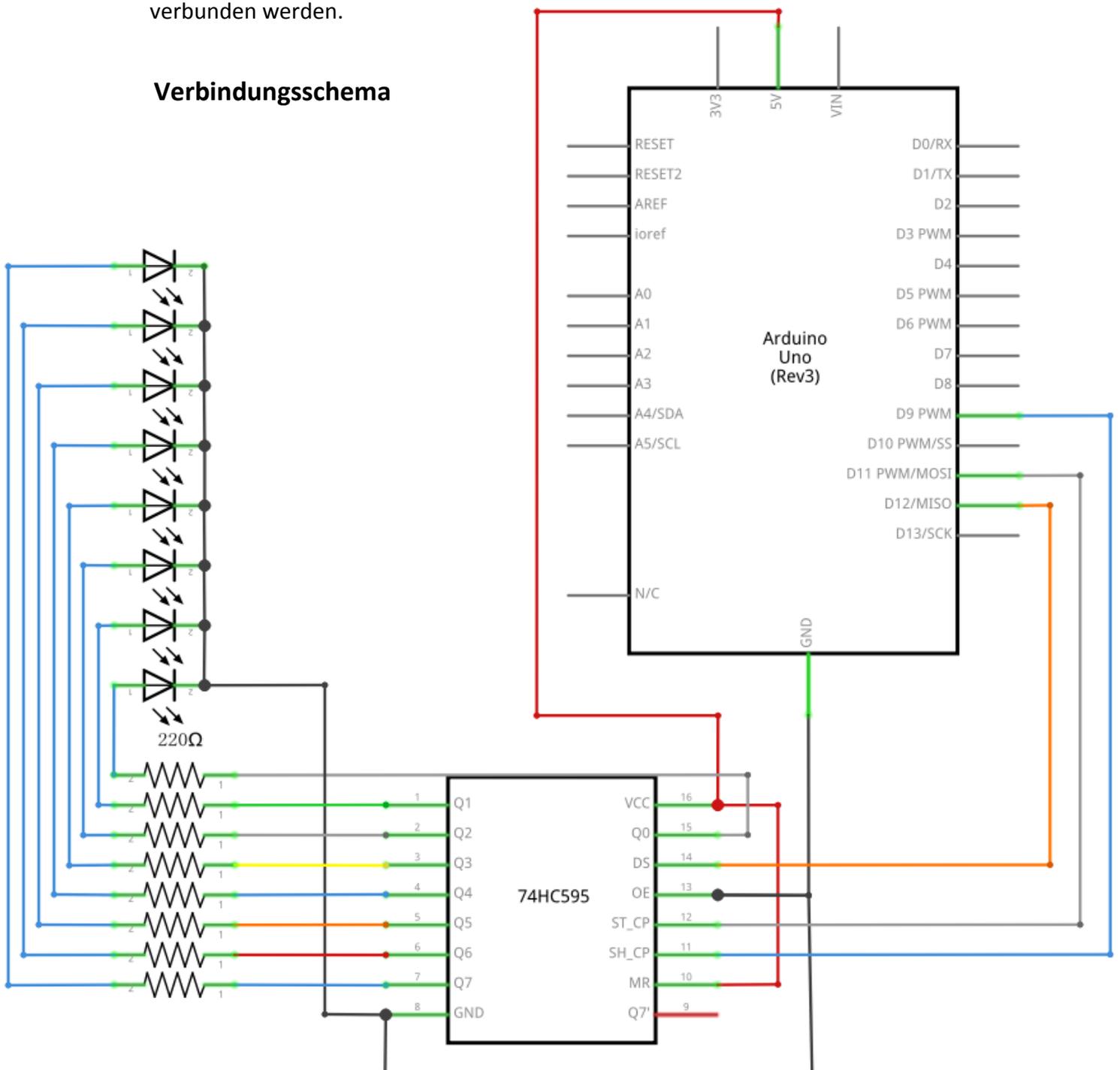
Ein Schieberegister ist ein Chip, der acht Speicherstellen hat, die jeweils entweder eine 1 oder eine 0 speichern können. Um die Werte zu verändern, werden wir die Daten über den „Data“- und den „Clock“-Anschluss des Chips senden.



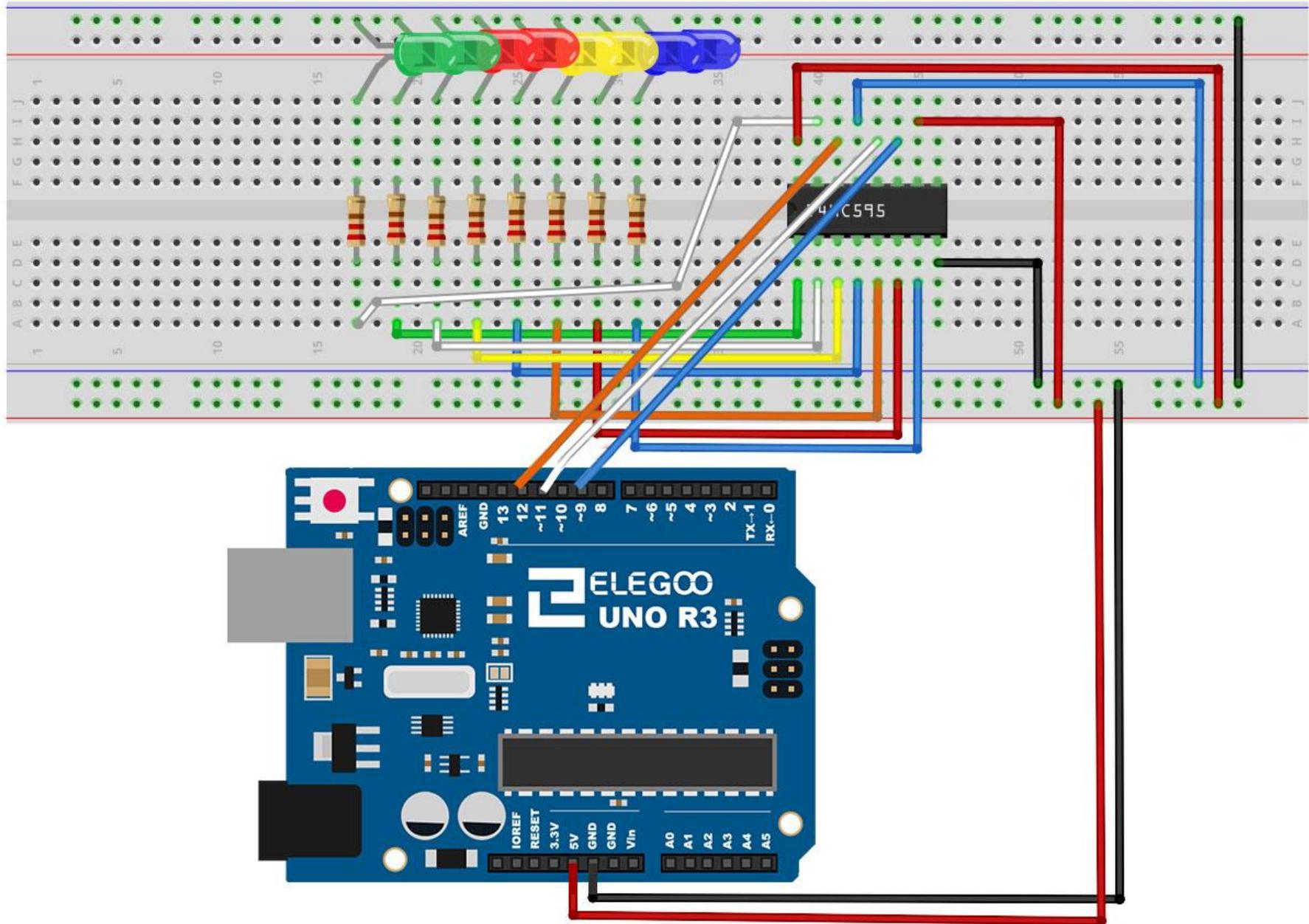
An den Clock-Pin müssen acht Pulse geschickt werden. Bei jedem Puls wird der Data-Pin überprüft. Wenn der Data-Pin HIGH geschaltet ist, wird eine 1 in das Schieberegister an die jeweilige Stelle geschrieben. Wenn der Pin LOW ist, wird eine 0 geschrieben. Nachdem alle 8 Pulse empfangen wurden, muss der „Latch“-Pin aktiviert werden und die acht Werte werden im Register gespeichert. Das ist unbedingt notwendig, da sonst die Daten an die falschen Stellen geschrieben werden könnten und die falschen LEDs aufleuchten würden.

Der Chip hat zusätzlich einen „Output Enable“ (OE)-Pin, durch den alle Ausgänge auf einmal an bzw. ausgeschaltet werden können. Sie könnten diesen Pin an einen PWM-fähigen Pin des Boards anschließen und mit der analogWrite-Funktion die Helligkeit der LEDs kontrollieren. Der OE-Pin ist aktiv LOW, also muss er mit GND verbunden werden.

Verbindungsschema



Schaltplan



Da wir insgesamt acht LEDs und acht Widerstände anschließen müssen, müssen einige Verbindungen hergestellt werden.

Am einfachsten ist es, wenn man den 74HC595-Chip zuerst einsetzt, da so ziemlich alles mit ihm verbunden wird. Setzen sie den Chip so ein, dass die kleine Einkerbung zum oberen Ende (der kurzen Seite) des Breadboards zeigt. Der Pin 1 befindet sich am Chip unter dieser Einkerbung.

Der digitale *Pin 12* des UNOs geht an *Pin 14* des Schieberegisters, der digitale *Pin 11* des UNOs geht an *PIN 12* des Schieberegisters und der digitale *Pin 9* des UNOs geht an *PIN 11* des Schieberegisters.

Alle Ausgänge außer einem sind auf der unteren Seite des Chips. Daher werden wir die LEDs der Einfachheit halber auf der linken Seite platzieren.

Nach dem Chip müssen die Widerstände an ihre Stelle gesetzt werden. Sie müssen aufpassen, dass die Kontakte der verschiedenen Widerstände sich nicht berühren. Sie sollten dies noch einmal überprüfen, bevor Sie das Board später mit Strom versorgen. Wenn es zu schwer ist die Widerstände einzusetzen, ohne dass sich dessen Kontakte berühren, kann es helfen die Kontakte zu kürzen, sodass die Widerstände tiefer am Breadboard anliegen.

Als nächstes platzieren Sie die LEDs auf dem Breadboard. Die langen positiven Enden müssen alle in Richtung Chip zeigen.

Schließen Sie die Jumper Kabel wie im obigen Bild gezeigt an. Vergessen Sie nicht das Kabel, das vom Pin 8 des Chips zum GND-Anschluss des Breadboards führt.

Laden Sie den Sketch hoch und probieren Sie es aus. Alle LEDs sollten nacheinander aufleuchten, bis alle an sind. Danach schalten sich alle LEDs aus und das Muster wiederholt sich anschließend.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 24 Eight LED with 74HC595*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte [Lektion 2](#) noch einmal an.

Als erstes im Sketch bestimmen wir die drei Pins, die wir benutzen werden. Dabei handelt es sich um die digitalen Ausgänge des UNO Boards, die mit den Latch-, Clock- und Data-Pins vom 74HC595 verbunden werden.

```
int latchPin = 11;
```

```
int clockPin = 9;
```

```
int dataPin = 12;
```

Als nächstes wird eine Variable mit dem Namen „*leds*“ gesetzt. Diese Variable werden wir dazu nutzen, um zu speichern welche LEDs gerade ein- und welche ausgeschaltet sind. Variablen des Typs „byte“ repräsentieren acht Bit große Zahlen. Jeder Bit kann dabei an oder aus sein. Also perfekt für die Zustände unserer LEDs.

```
byte leds = 0;
```

Die setup-Funktion bestimmt nur, dass wir unsere drei Pins als Ausgänge nutzen.

```
void setup()
{
    pinMode(latchPin, OUTPUT);
    pinMode(dataPin, OUTPUT);
    pinMode(clockPin, OUTPUT);
}
```

Die loop-Funktion schaltet zu Beginn erst einmal alle LEDs aus, indem die *leds*-Variable auf 0 gesetzt wird. Dann wird „*updateShiftRegister*“ aufgerufen, das den Wert der *leds*-Variable an das Schieberegister sendet, was somit alle LEDs ausschaltet. Um die Funktionsweise der *updateShiftRegister*-Funktion kümmern wir uns später. Die loop-Funktion wird durch eine 0,5 sekündige Pause unterbrochen und beginnt dann in einer *for*-Schleife in Durchgängen von 0 bis 7 zu zählen, wobei die Variable *i* die aktuelle Zahl repräsentiert. Bei jedem Durchgang wird die Funktion „*bitSet*“ aufgerufen, die die Zahl in der *leds*-Variable ändert. Danach wird wieder *updateShiftRegister* aufgerufen, damit der Chip die LEDs anpasst. Dann gibt es wieder eine 0,5 sekündige Pause und danach folgt der nächste Durchgang.

```
void loop()
{
    leds = 0;
    updateShiftRegister();
    delay(500);
    for (int i = 0; i < 8; i++)
    {
        bitSet(leds, i);
        updateShiftRegister();
        delay(500);
    }
}
```

```
}  
}
```

Die Funktion `updateShiftRegister` setzt zuerst den latch-Pin auf LOW, und ruft dann die Funktion „`shiftOut`“ auf, bevor sie den latch-Pin wieder auf HIGH umschaltet.

Die `shiftOut`-Funktion benötigt vier Parameter. Die ersten beiden sind die Pins, die an Data und Clock angeschlossen sind.

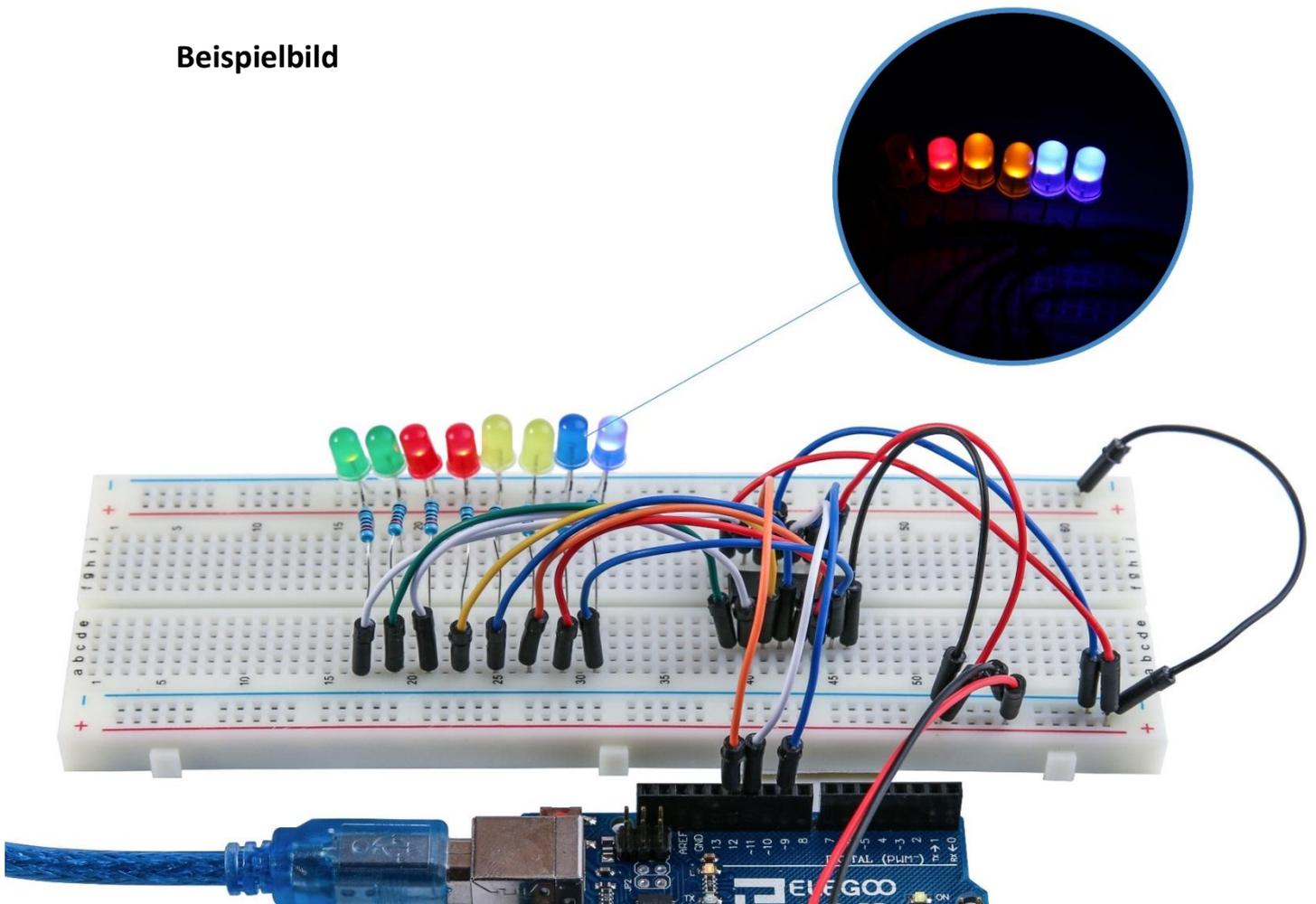
Der dritte Parameter bestimmt an welchem Ende wir starten wollen. Wir starten bei dem Bit ganz rechts, der „*Least Significant Bit*“ (LSB) genannt wird.

Der letzte Parameter sind die eigentlichen Daten, die wir an das Schieberegister übertragen wollen. Bei uns ist der letzte Parameter die `leds`-Variable.

```
void updateShiftRegister()  
{  
    digitalWrite(latchPin, LOW);  
    shiftOut(dataPin, clockPin, LSBFIRST, leds);  
    digitalWrite(latchPin, HIGH);  
}
```

Wenn sie stattdessen lieber eine LED aus- statt einschalten möchten, müssen sie eine ähnliche Arduino-Funktion, die „`bitClear`“-Funktion, mit der `leds`-Variable als Parameter aufrufen. Diese Funktion setzt den entsprechenden Bit auf 0. Danach müssen Sie nur noch die `updateShiftRegister`-Funktion aufrufen.

Beispielbild



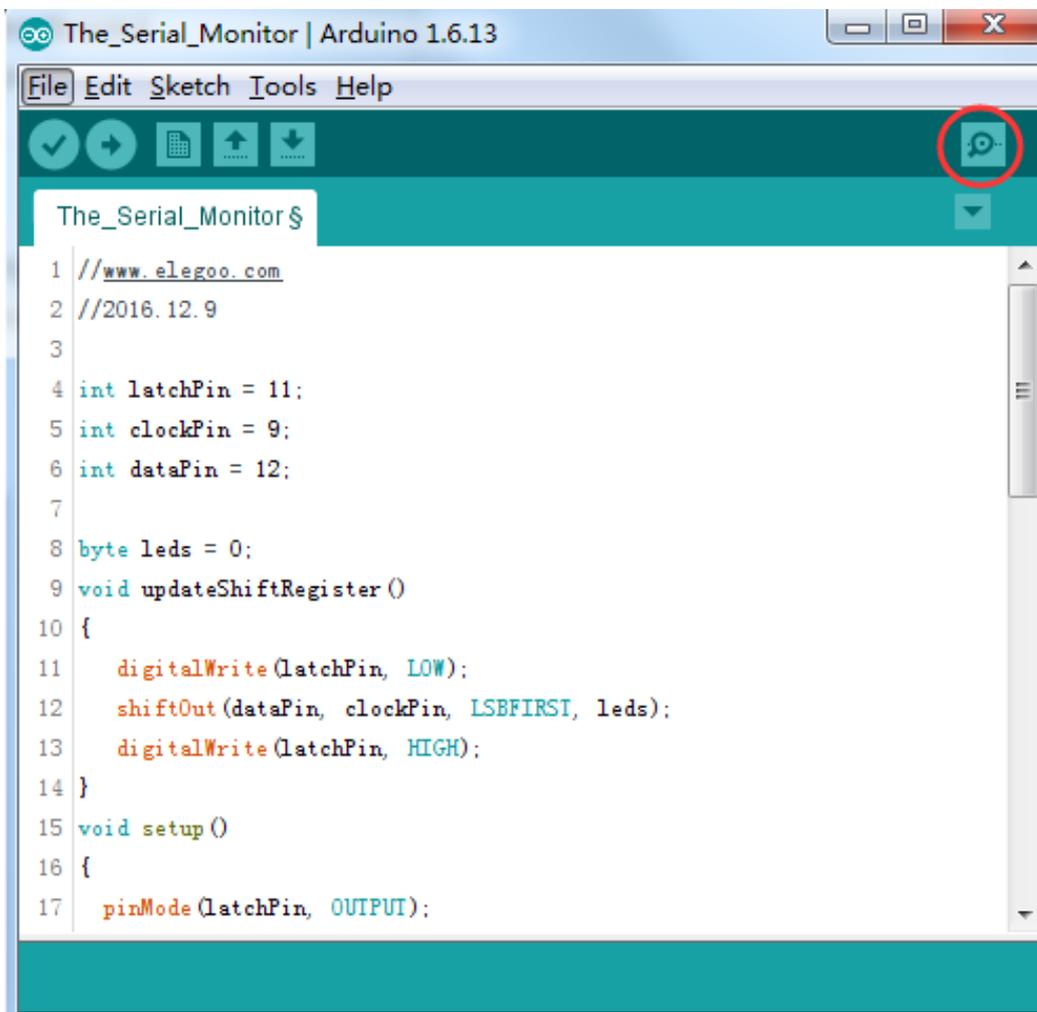
Lektion 25:Der Serielle Monitor

Übersicht

In dieser Lektion werden wir auf Lektion 24 aufbauen. Wir werden die Möglichkeit einbauen die LEDs vom Computer aus über den Seriellen Monitor zu steuern. Der Serielle Monitor ist die Verbindung zwischen Ihrem Computer und dem UNO Board. Durch ihn können Sie Textnachrichten senden und empfangen, was nützlich zum debuggen und zum steuern des UNO Boards sein kann. Zum Beispiel können Sie von Ihrem Computer einen Befehl senden, der die LEDs einschalten wird. In dieser Lektion werden die selben Teile und ein ähnliches Breadboard Layout wie in Lektion 24 benutzt. Lesen Sie zuerst Lektion 24, falls Sie es noch nicht getan haben.

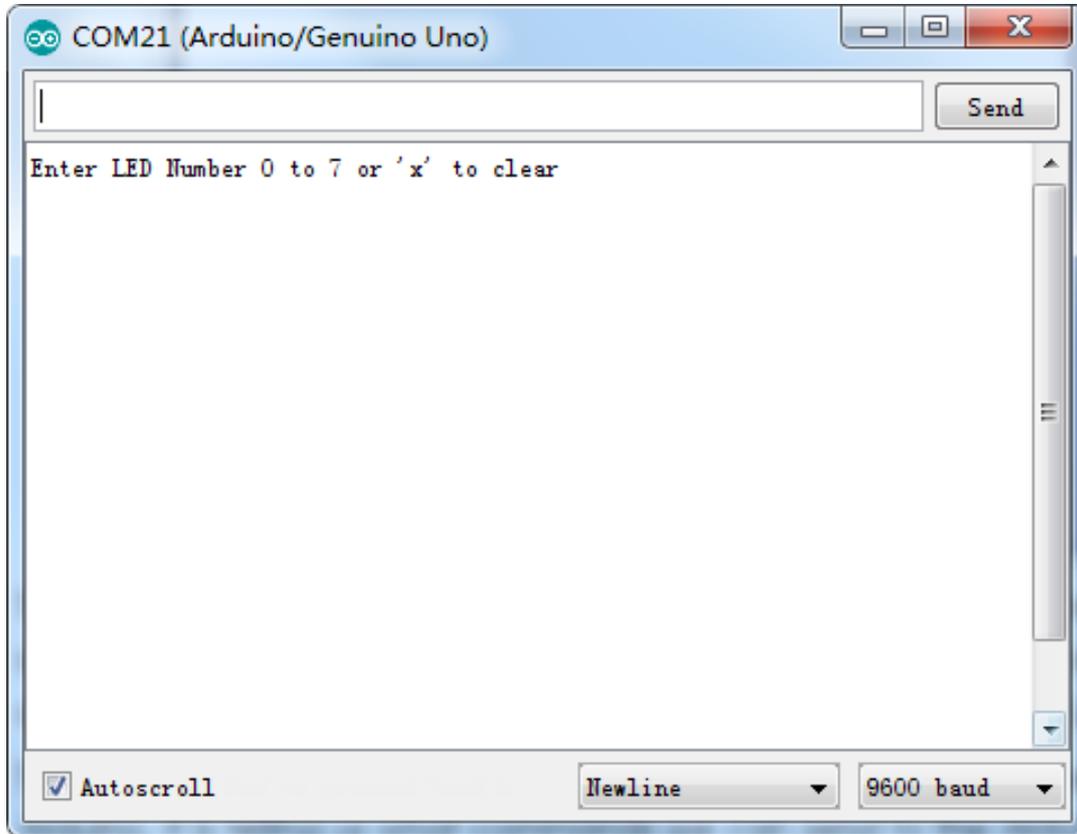
Durchführung

Nachdem Sie den Sketch auf Ihr Board hochgeladen haben, klicken Sie in der IDE auf das Symbol des Seriellen Monitors (unten in der Abbildung markiert).



Das folgende Fenster wird sich öffnen.

Die grundlegenden Informationen zum Seriellen Monitor haben Sie in [Lektion 1](#) kennengelernt.



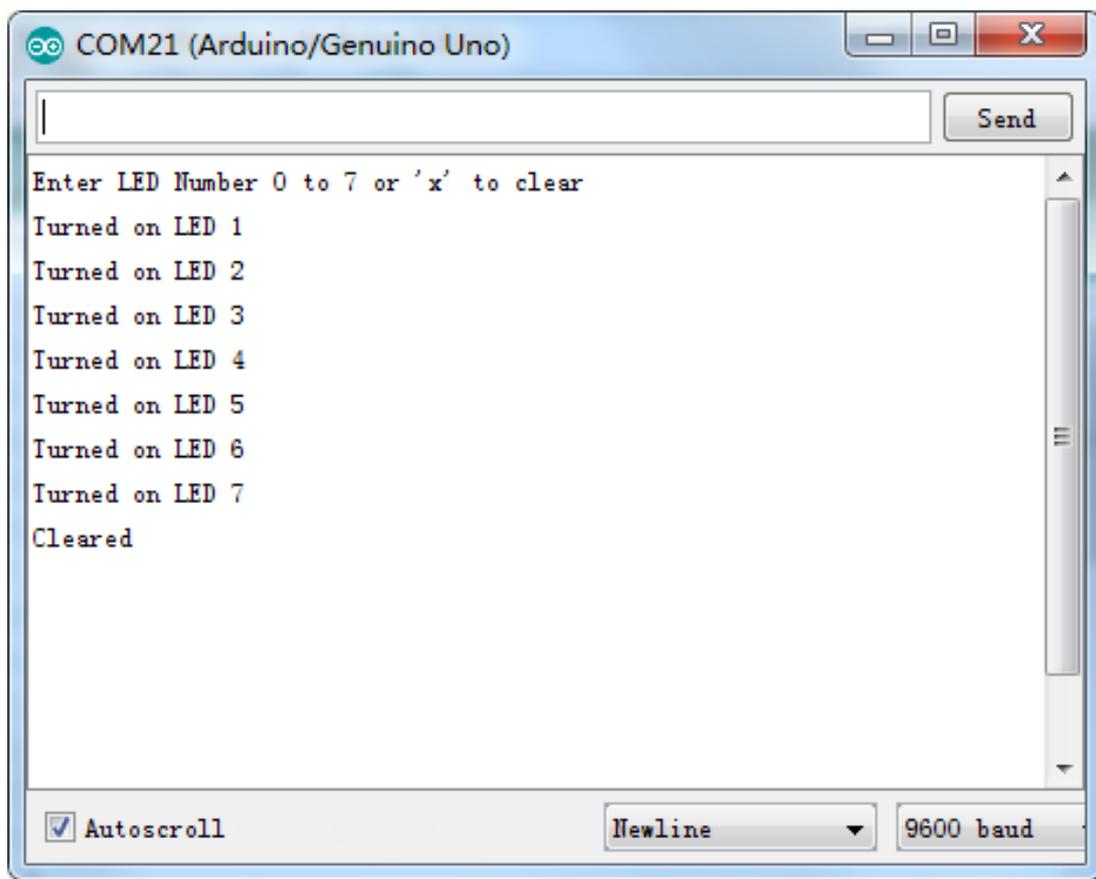
Dieses Fenster ist der Serielle Monitor, der es Ihnen erlaubt sowohl Nachrichten von Ihrem Computer an das UNO Board zu senden, als auch Nachrichten vom UNO Board zu empfangen. Die erste Nachricht, die Sie bekommen, lautet: „*Enter LED Number 0 to 7 or 'x' to clear*“.

Die Nachricht teilt uns mit, welche Befehle wir an das Board senden können: Ein „x“ schaltet alle LEDs aus. Eine beliebige Zahl von 0 – 7 schaltet die jeweilige LED ein, wobei 0 die LED ganz links ist und 7 die LED ganz rechts.

Versuchen Sie die folgenden Befehle nacheinander über das Eingabefeld oben im Seriellen Monitor an Ihr Board zu senden. Drücken Sie nach jedem Befehl „Senden“: *x 0 3 5*

Der erste Befehl (x) wird nichts auslösen, da die LEDs bereits alle ausgeschaltet sind. Beim Senden der Zahlen sollte jedoch die entsprechende LED aufleuchten und Sie sollten eine Bestätigungsnachricht vom Board im Seriellen Monitor erhalten. Die Anzeige im Seriellen Monitor sollte dann bei Ihnen ähnlich wie hier

aussehen:



Geben Sie erneut „x“ ein und klicken Sie auf „Senden“, um alle LEDs auszuschalten.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „[Lesson 25 The Serial Monitor](#)“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte [Lektion 2](#) noch einmal an.

Wie Sie wahrscheinlich bereits geahnt haben, basiert unser Sketch auf dem Sketch aus Lektion 24. Also werden wir hier nur die neuen Teile aufgreifen.

In der setup-Funktion gibt es drei neue Zeilen am Ende.

```
void setup()
{
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  updateShiftRegister();
}
```

```

Serial.begin(9600);
while (! Serial); // Wait until Serial is ready - Leonardo
Serial.println("Enter LED Number 0 to 7 or 'x' to clear");
}

```

Zuerst haben wir den Befehl „Serial.begin(9600)“. Dieser leitet die Serielle Kommunikation mit einer Baudrate von 9600 ein. Die Baudrate ist die Geschwindigkeit der Datenübertragung. Sie können die Baudrate auf einen höheren Wert stellen, müssen dabei aber auch die Einstellung im Seriellen Monitor anpassen. Zunächst belassen Sie den Wert aber bitte bei 9600.

Die Zeile mit der „while“-Schleife wartet solange, bis eine Serielle Verbindung zum Computer hergestellt ist und fährt dann fort.

Ohne diesen Code könnte es passieren, dass die erste Nachricht zu früh gesendet und daher nicht empfangen wird. Dies ist allerdings nur beim Arduino Leonardo notwendig, da beim UNO (ihrem Board) das Board automatisch zurückgesetzt wird, wenn Sie den Seriellen Monitor öffnen.

Die letzte Zeile in der setup-Funktion sendet die „Begrüßungsnachricht“ aus, die wir später oben im Seriellen Monitor sehen werden.

Die ganze Steuerung der LEDs passiert in der loop-Funktion:

```

void loop()
{
  if (Serial.available())
  {
    char ch = Serial.read();
    if (ch >= '0' && ch <= '7')
    {
      int led = ch - '0';
      bitSet(leds, led);
      updateShiftRegister();
      Serial.print("Turned on LED ");
      Serial.println(led);
    }
    if (ch == 'x')
    {
      leds = 0;
      updateShiftRegister();
    }
  }
}

```

```

        Serial.println("Cleared");
    }
}
}

```

Alles, was innerhalb der loop-Funktion passiert, ist in einem if-Statement mit dem Parameter „*Serial.available()*“. Es wird also nur etwas passieren, wenn die Funktion *Serial.available()* „*true*“ (= wahr) zurückgibt.

Serial.available() gibt *true* zurück, wenn Nachrichten über die Serielle Verbindung empfangen wurden. Eingehende Serielle Nachrichten werden in einem sogenannten „*Buffer*“ (= Puffer) gespeichert. Wenn dieser Buffer Nachrichten enthält, gibt *Serial.available()* *true* zurück.

Wenn eine Nachricht empfangen wurde, kann man sie über diesen Befehl lesen:

```
char ch = Serial.read();
```

Die Funktion „*Serial.read()*“ liest das nächste Zeichen aus dem Buffer und entfernt es aus diesem. Hier wird dieses Zeichen gleichzeitig der Variable „*ch*“ zugewiesen. Die *ch*-Variable ist eine Variable des Typs „*char*“, was für Charakter bzw Zeichen steht. Eine *char*-Variable kann also genau ein Zeichen abspeichern.

Wenn Sie sich an die Instruktionen der ersten Nachricht im Seriellen Monitor halten, kann es sich bei dem empfangenden Character um den Buchstaben *x* oder eine Zahl zwischen 0 und 7 handeln.

Das if-Statement in der nächsten Zeile überprüft ob es sich um eine Zahl zwischen 0 und 7 handelt, indem es überprüft ob es sich um eine Zahl handelt, die größer oder gleich 0 und gleichzeitig kleiner oder gleich 7 ist. Da es sich bei einem Zeichen auch um einen Buchstaben handeln kann, wird nicht direkt das Zeichen verglichen, sondern der sogenannte „*ASCII*“-Wert. (American Standard Code for Information Interchange). Jedes Zeichen hat einen einzigartigen vordefinierten ASCII-Wert in Form einer Zahl und diese Zahlen kann man vergleichen und so herausfinden, um welches Zeichen es sich handelt.

Wenn das if-Statement wahr ist, kommen wir zur nächsten Zeile.

```
int led = ch - '0';
```

Nun rechnen wir mit Charaktern! Wir subtrahieren die 0 vom Charakter, den die *ch*-Variable gerade enthält und speichern das Ergebnis in der Variable „*led*“. Wenn über die Serielle Verbindung eine 0 empfangen, wird (weil $0-0 = 0$) eine 0 in der *led*-Variable gespeichert. Bei einer 7, würde $7-0 = 7$ in der Variable gespeichert.

Da wir nun die Nummer der LED wissen, die angeschaltet werden soll, können wir den entsprechenden Bit in der leds-Variable setzen und das Schieberegister updaten.

```
bitSet(leds, led);
```

```
updateShiftRegister();
```

Die nächsten zwei Zeilen geben eine Bestätigungsnachricht an den Seriellen Monitor zurück.

```
Serial.print("Turned on LED ");
```

```
Serial.println(led);
```

Die erste Zeile nutzt die Funktion „*Serial.print()*“ statt „*Serial.println()*“. Der unterschied zwischen den beiden Funktionen ist, dass die letztere zusätzlich zur angegebenen Nachricht noch einen Zeilenumbruch ausgibt, also die nächste Ausgabe in einer neuen Zeile anfangen wird. Erstere Funktion dagegen macht keinen Umbruch und man kann danach mit dem nächsten print-Befehl genau an der Stelle weiterschreiben, wo der letzte print-Befehl aufgehört hat. Wir benutzen in der ersten Zeile *Serial.print()*, da wir keinen Umbruch wollen, weil das nur der erste Teil der Nachricht ist. Mit der zweiten Zeile wird die LED-Nummer angehängt und die Nachricht ist fertig, weshalb wir *Serial.println()* nutzen, was am Ende automatisch in die nächste Zeile wechselt.

Die Nummer der LED wird dabei in der led-Variable, eine Variable des Typs „*int*“ (Zahlenvariable) gespeichert. Eigentlich benötigt die *Serial.println*-Funktion einen „*String*“ (Text) als Parameter. Das ist aber nicht schlimm, da die Funktion *int*-Variablen automatisch in Text umwandelt.

Nach dem if-Statement, dass sich um Zahleneingaben von 0 bis 7 kümmert, folgt das if-Statement, dass sich darum kümmert was passiert, wenn ein x empfangen wird (*ch = x*):

```
if (ch == 'x')
{
    leds = 0;
    updateShiftRegister();
    Serial.println("Cleared");
}
```

Bei einem „x“ werden alle Bits der leds-Variable auf 0 gesetzt, das Schieberegister geupdatet und eine Bestätigungsnachricht geschickt, dass alle LEDs ausgeschaltet wurden.

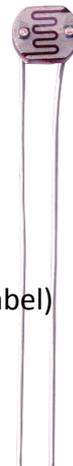
Lektion 26:Fotozelle

Übersicht

In dieser Lektion lernen Sie, wie man die Lichtintensität mit Hilfe einer Fotozelle über einen analogen Ausgang misst. Wir bauen auf Lektion 24 auf und kontrollieren mit Hilfe der Lichtintensität / Helligkeit die Anzahl der LEDs, die aufleuchten sollen. Die Fotozelle wird links am Breadboard platziert.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (8) x LEDs
- (8) x 220 Ohm Widerstände
- (1) x 1K Ohm Widerstand
- (1) x 74HC595 IC
- (1) x Fotowiderstand (Fotozelle)
- (16) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)



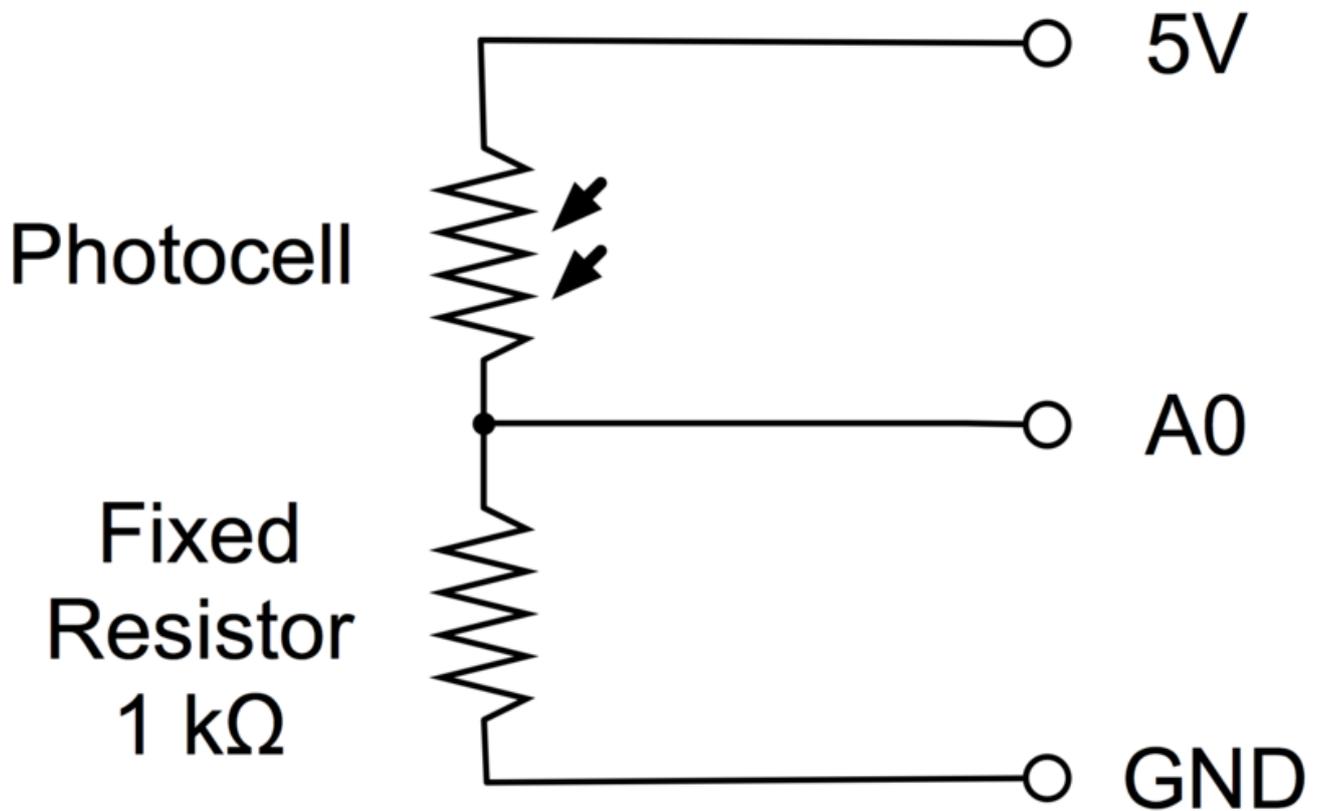
Einführung in die Komponenten

Fotozelle:

Eine Fotozelle ist eine Art lichtabhängiger Widerstand, der manchmal auch „LDR“ genannt wird. Dieses Bauteil arbeitet wie ein normaler Widerstand, mit der Ausnahme, dass der Widerstandswert sich mit der Lichtmenge, die auf ihn strahlt, ändert.

Diese Fotozelle hat in fast kompletter Dunkelheit einen Widerstand von 50 k Ω und in hellem Licht einen Widerstand von etwa 500 Ω . Damit wir den Widerstandswert mit unserem UNO R3 Board über einen analogen Eingang messen können, müssen wir ihn irgendwie in eine Spannung umwandeln.

Der einfachste Weg dies zu tun ist, die Fotozelle mit einem festen Widerstand zu kombinieren.

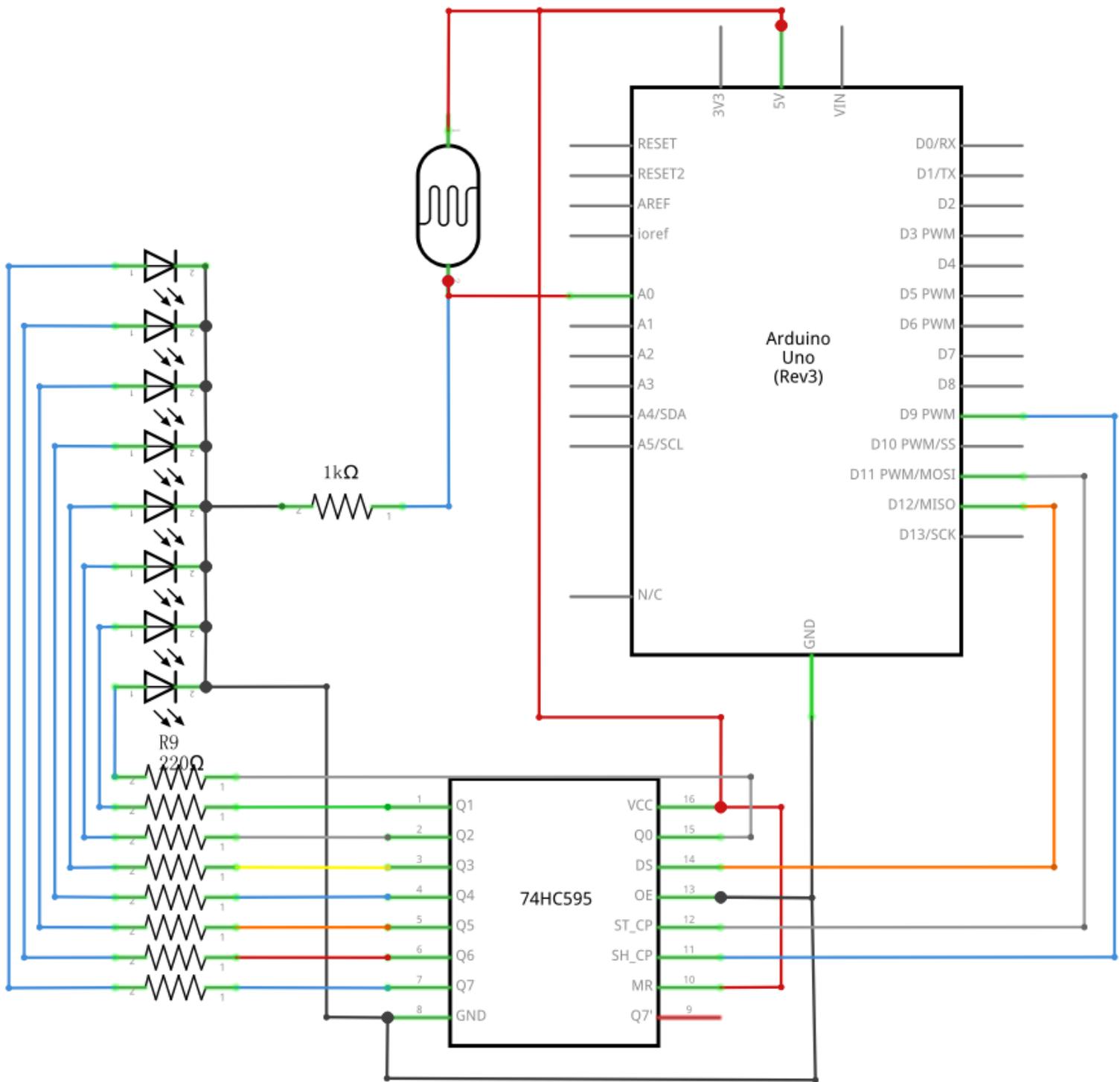


Der feste Widerstand und die Fotozelle zusammen arbeiten ähnlich wie ein Potentiometer. Wenn das Licht hell ist, ist der Widerstand der Fotozelle im Vergleich zum festen Widerstand sehr gering, also verhält es sich wie wenn ein Potentiometer auf das Maximum gestellt ist.

Wenn die Fotozelle sich in dunkler Umgebung befindet, wird ihr Widerstand größer als der des festen 1 kΩ Widerstands und es verhält sich wie wenn ein Potentiometer auf das Minimum (nach GND) gestellt ist.

Laden Sie den Sketch hoch und probieren Sie einmal Ihren Finger auf die Fotozelle zu halten und danach die Fotozelle in die Nähe einer Lichtquelle zu positionieren.

Verbindungsschema



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter [Lektion 26 Photocell](#) und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte [Lektion 2](#) noch einmal an.

Das erste, was wir beachten müssen, ist, dass der Name des analogen Eingangs „lightPin“ heißt.

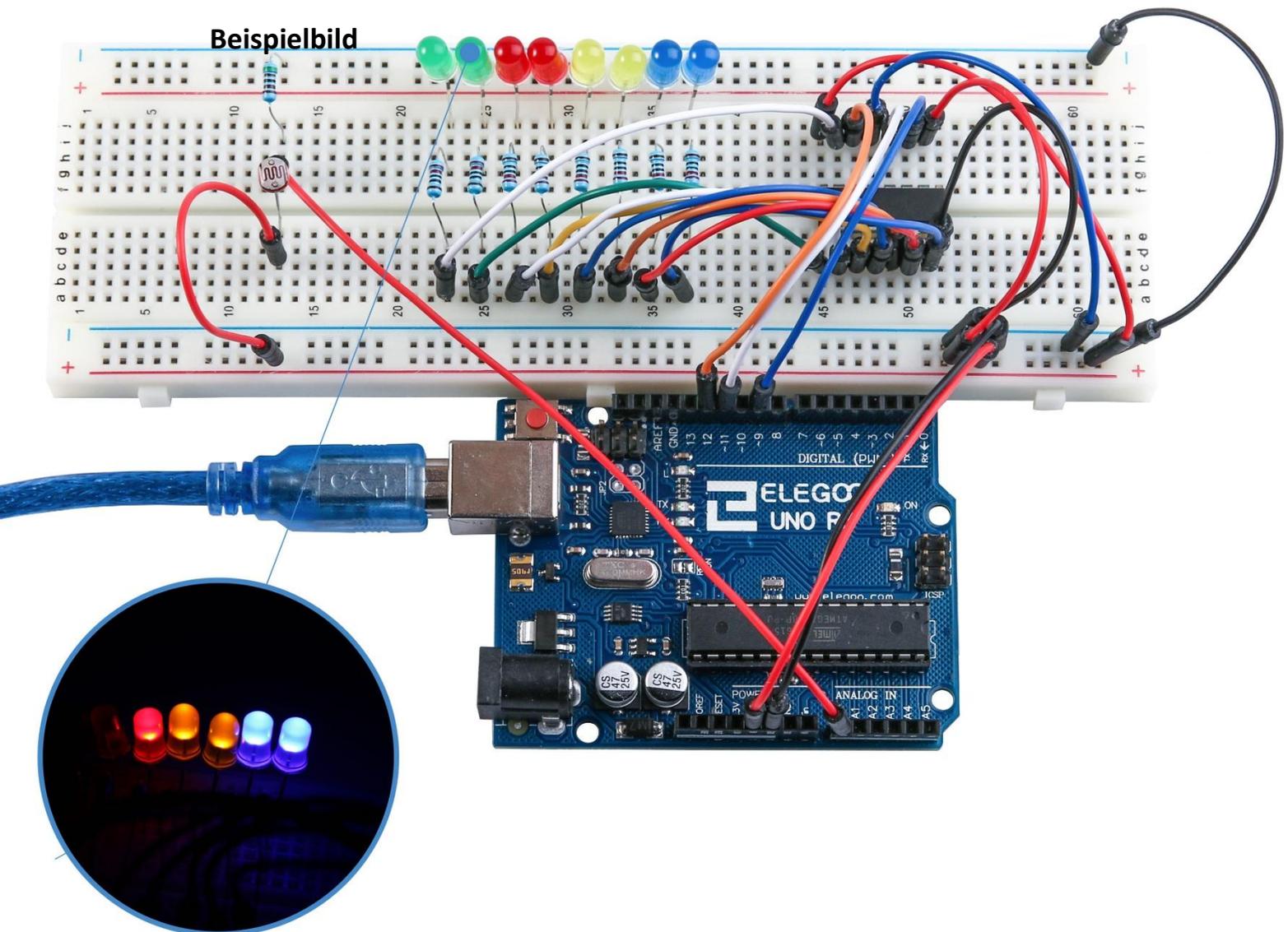
Die nachfolgende Zeile berechnet, wie viele LEDs aufleuchten sollen.

```
int numLEDSLit = reading / 57; // all LEDs lit at 1k
```

Wir teilen den Rohwert, den wir am Pin erhalten durch 57. In anderen Worten teilen wir den Widerstandsbereich in neun Zonen auf, von alle LEDs ausgeschaltet bis alle LEDs eingeschaltet. Wenn am Eingang nun ein Widerstand von 1 kΩ anliegt (also die Fozelle komplett beleuchtet ist), wird der Rohwert etwa $1023 / 2 = 511$ betragen. Bei diesem Wert alle LEDs aufleuchten. Und die „numLEDSLit“-Variable würde

$1023 / 57 = 9$ betragen.

Beispielbild



Lektion 27: 74HC595 mit Siebensegmentanzeige

Übersicht

Nach Lektion 24, 25 und 26, erweitern wir unser Projekt erneut und werden mit dem 74HC595 Schieberegister eine Segmentanzeige steuern. Die Siebensegmentanzeige kann Zahlen von 0 – 9 anzeigen.

Benötigte Bauteile:

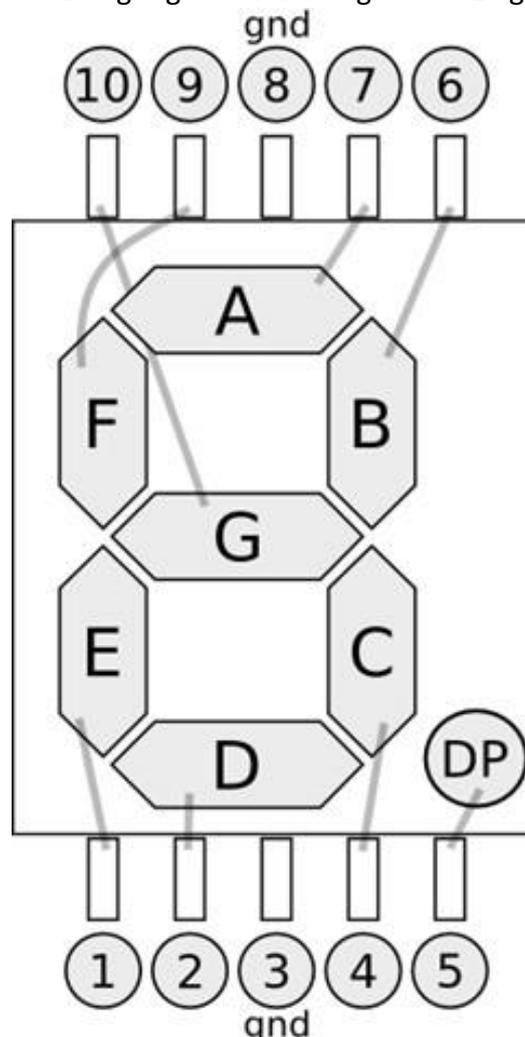
- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x 74HC595 IC
- (1) x 1-Ziffer-Siebensegmentanzeige
- (8) x 220 Ohm Widerstände
- (26) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)



Einführung in die Komponenten

7-Segment-Anzeige

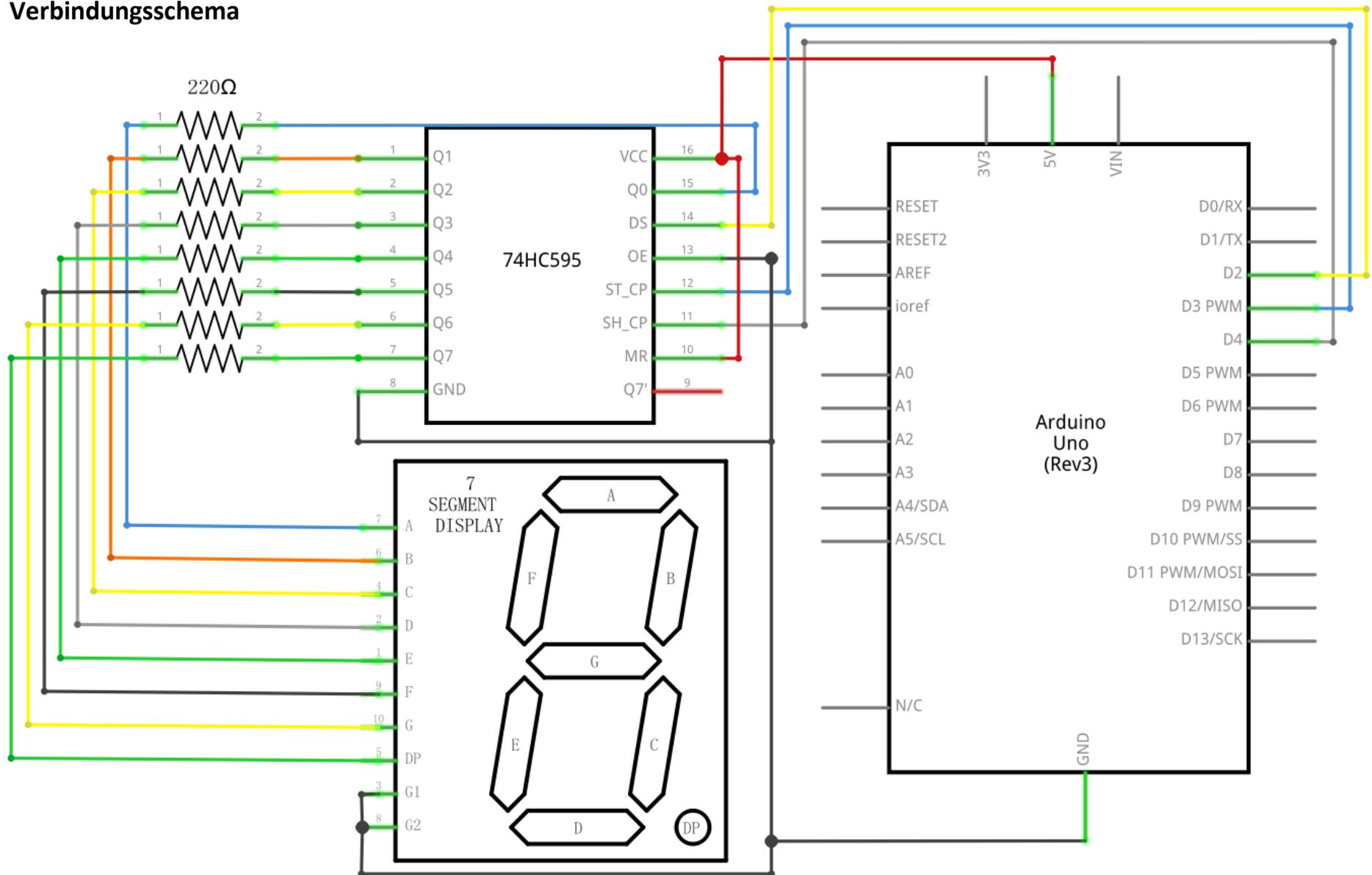
Unten sehen Sie die Pin-Belegung der Siebensegmentanzeige.



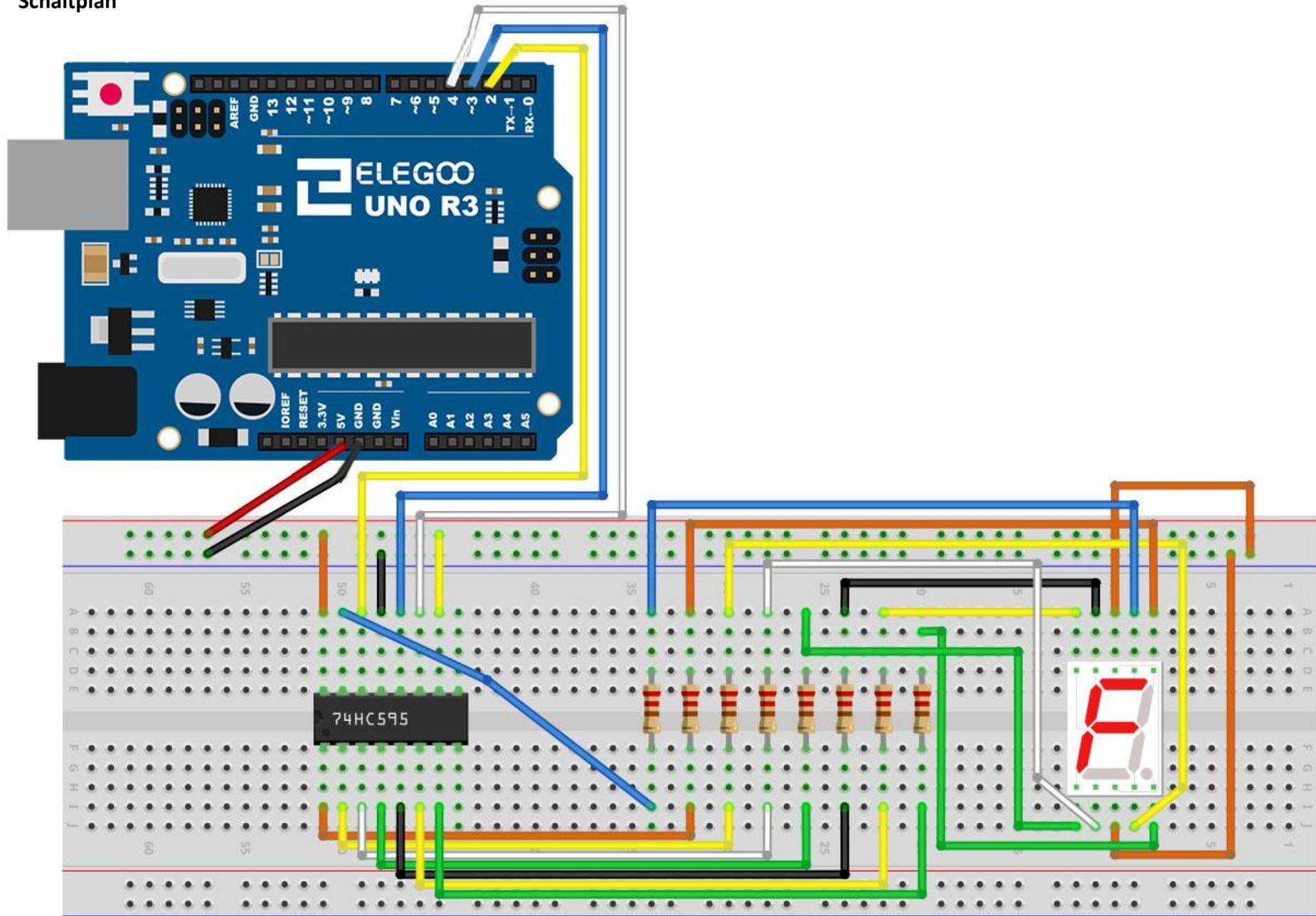
10 Ziffern (0-9) können wie folgt mit der Segmentanzeige dargestellt werden (Die folgende Tabelle gilt für Siebensegmentanzeigen mit gemeinsamer Kathode. Bei gemeinsamer Anode müssen Anpassungen durchgeführt werden):

Ziffer	dp	a	b	c	d	e	f	g
0	0	1	1	1	1	1	1	0
1	0	0	1	1	0	0	0	0
2	0	1	1	0	1	1	0	1
3	0	1	1	1	1	0	0	1
4	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1
6	0	1	0	1	1	1	1	1
7	0	1	1	1	0	0	0	0
8	0	1	1	1	1	1	1	1
9	0	1	1	1	1	0	1	1

Verbindungsschema



Schaltplan



Die folgende Tabelle zeigt, welcher Pin vom 74HC595-Chip mit welchem Pin an der Siebensegmentanzeige verbunden werden muss.

74HC595 Pin	Siebensegmentanzeige Pin
Q0	7 (A)
Q1	6 (B)
Q2	4 (C)
Q3	2 (D)
Q4	1 (E)
Q5	9 (F)
Q6	10 (G)
Q7	5 (DP)

Schritt 1: Verbinden des 74HC595

Zuerst verbinden wir den Chip mit der Stromversorgung und GND:

VCC (Pin 16) und **MR** (Pin 10) werden mit 5V verbunden.

GND(Pin 8) und **OE** (Pin 13) werden mit GND verbunden.

DS, **ST_CP** und **SH_CP** verbinden:

DS (Pin 14) wird mit dem UNO R3 Board *Pin 2 verbunden* (gelbes Kabel)

ST_CP(Pin 12, Latch-Pin) wird mit dem UNO R3 Board *Pin 3 verbunden* (blaues Kabel)

SH_CP(Pin 11, Clock-Pin) wird mit dem UNO R3 Board *Pin 4 verbunden* (weißes Kabel)

Schritt 2: Verbinden der Siebensegmentanzeige

Pin 3 und *Pin 8* gehen an den **GND**-Pin vom UNO R3 Board (Bei gemeinsamer Kathode. Bei gemeinsamer Anode müssen die Pins 3 und 8 mit +5V verbunden werden)

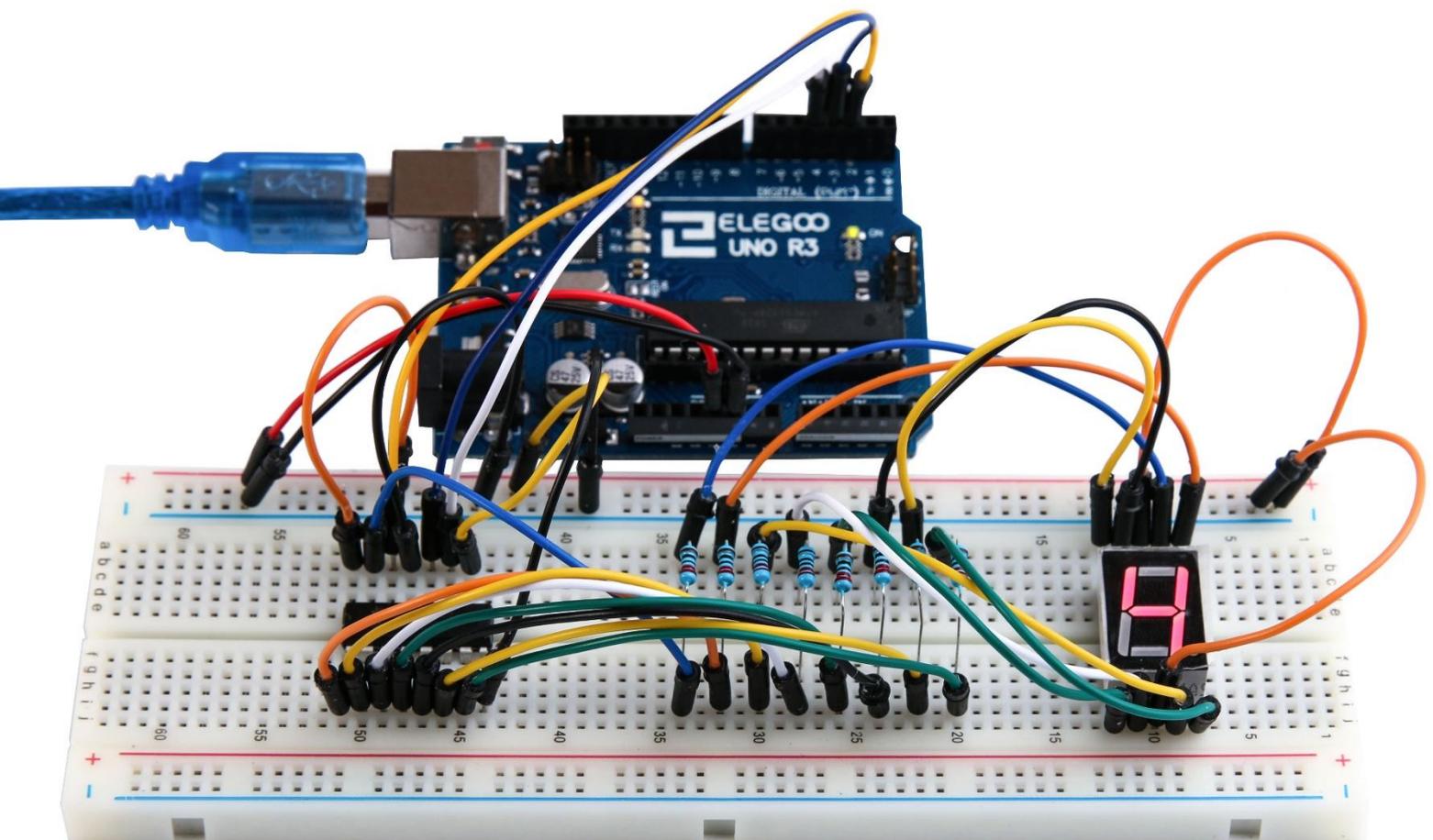
Verbinden Sie die die *Q0 – Q7 Pins* vom 74HC595-Chip mit den entsprechenden in der obigen Tabelle gelisteten Pins der Siebensegmentanzeige (*A-G* und *DP*).

Schalten Sie bei jeder Verbindung zu den Segmenten einen 220 Ohm Widerstand vor.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 27 74HC595 And Segment Display“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Beispielbild



Lektion 28: Digitale 4-Ziffer-Siebensegmentanzeige

Übersicht

In dieser Lektion lernen Sie, wie man eine vierziffrige Siebensegmentanzeige benutzt.

Beim Benutzen einer einziffrigen Siebensegmentanzeige wird die gemeinsame Kathode mit GND verbunden. Handelt es sich um eine gemeinsame Anode, wird diese mit der Stromversorgung verbunden.

Wenn wir eine vierziffrige Siebensegmentanzeige nutzen, wird der gemeinsame Pin (egal ob Anode oder Kathode) verwendet, um zu bestimmen welche Ziffer angesprochen wird. Auch wenn nur eine Ziffer gleichzeitig gesteuert wird, sieht es mit dem Auge am Ende so aus, als ob sich alle vier Ziffern auf einmal ändern würden, da die Änderungsintervalle so klein sind.

Benötigte Bauteile:

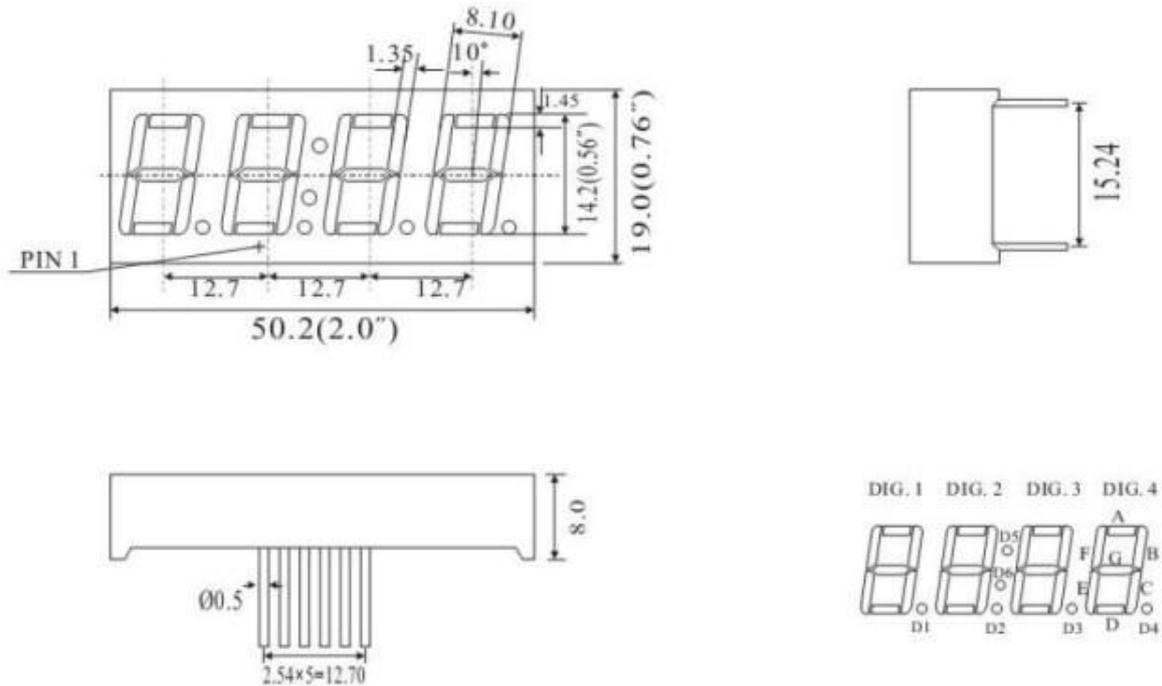
- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x 74HC595 IC
- (1) x 4-Ziffer-Siebensegmentanzeige
- (4) x 220 Ohm Widerstände
- (23) x M-M Kabel (Männlich zu Männlich DuPont JumperKabel)



Einführung in die Komponenten
Vierziffrige Siebensegmentanzeige

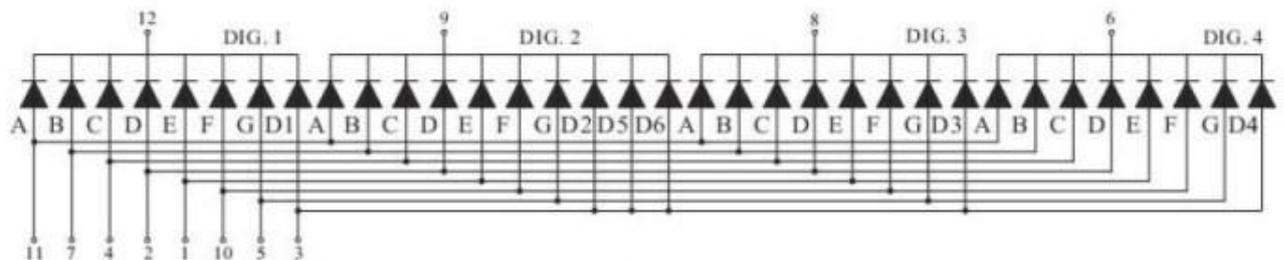
Package Dimensions

CPS05643AB

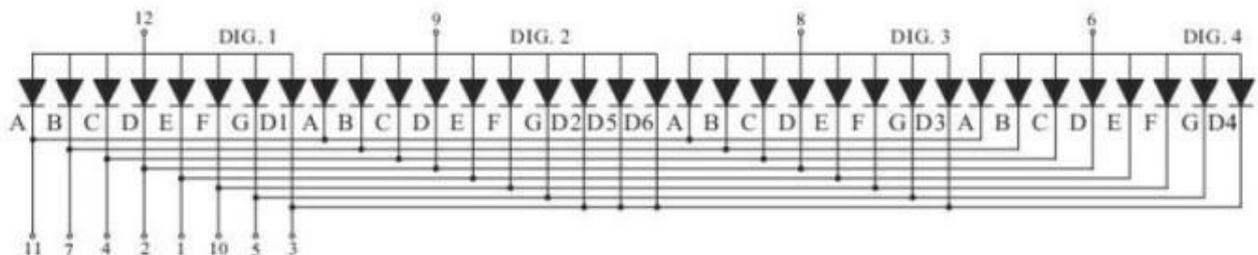


UNIT: MM(INCH) TOLERANCE: $\pm 0.25(0.01")$

Internal Circuit Diagram



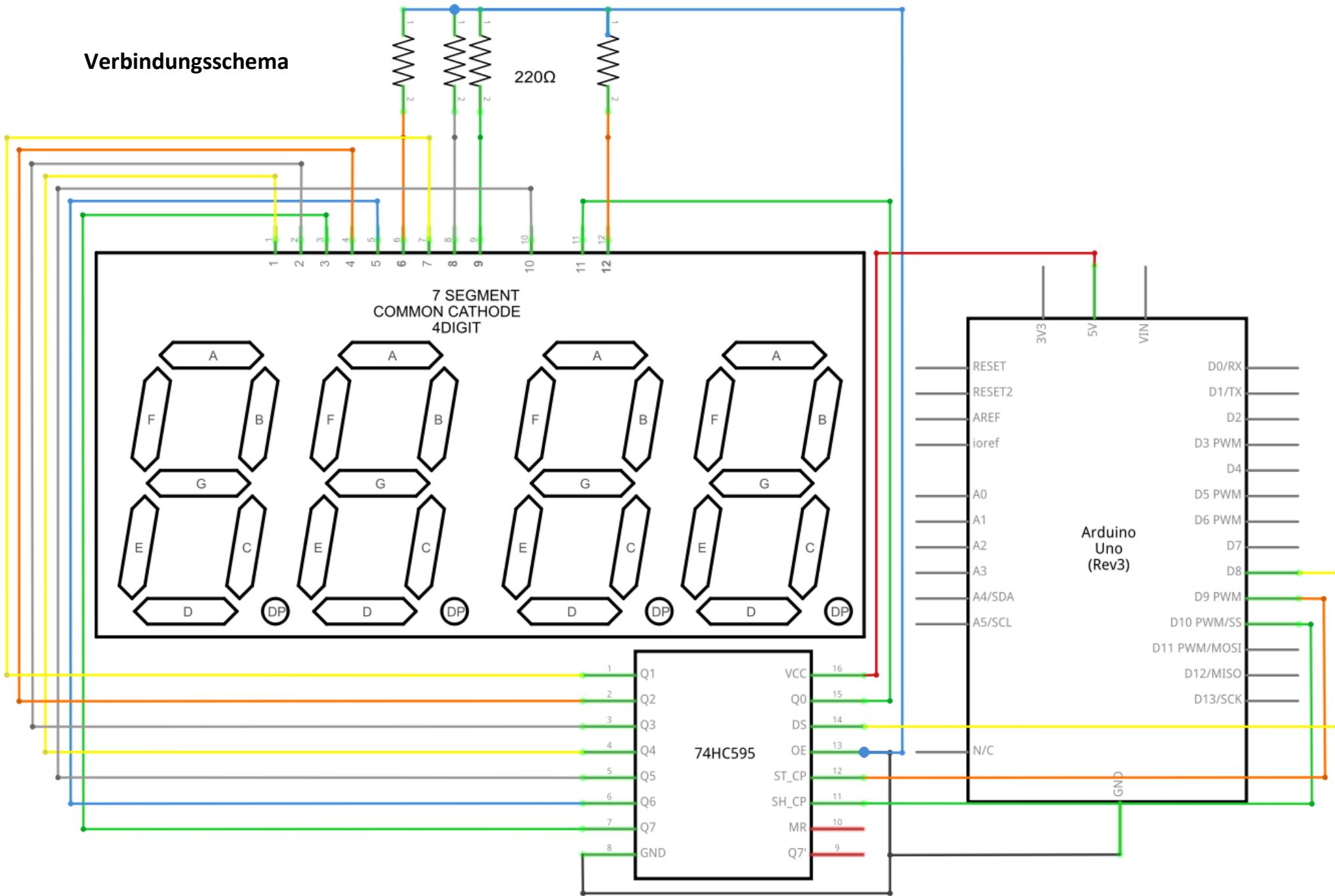
5643A



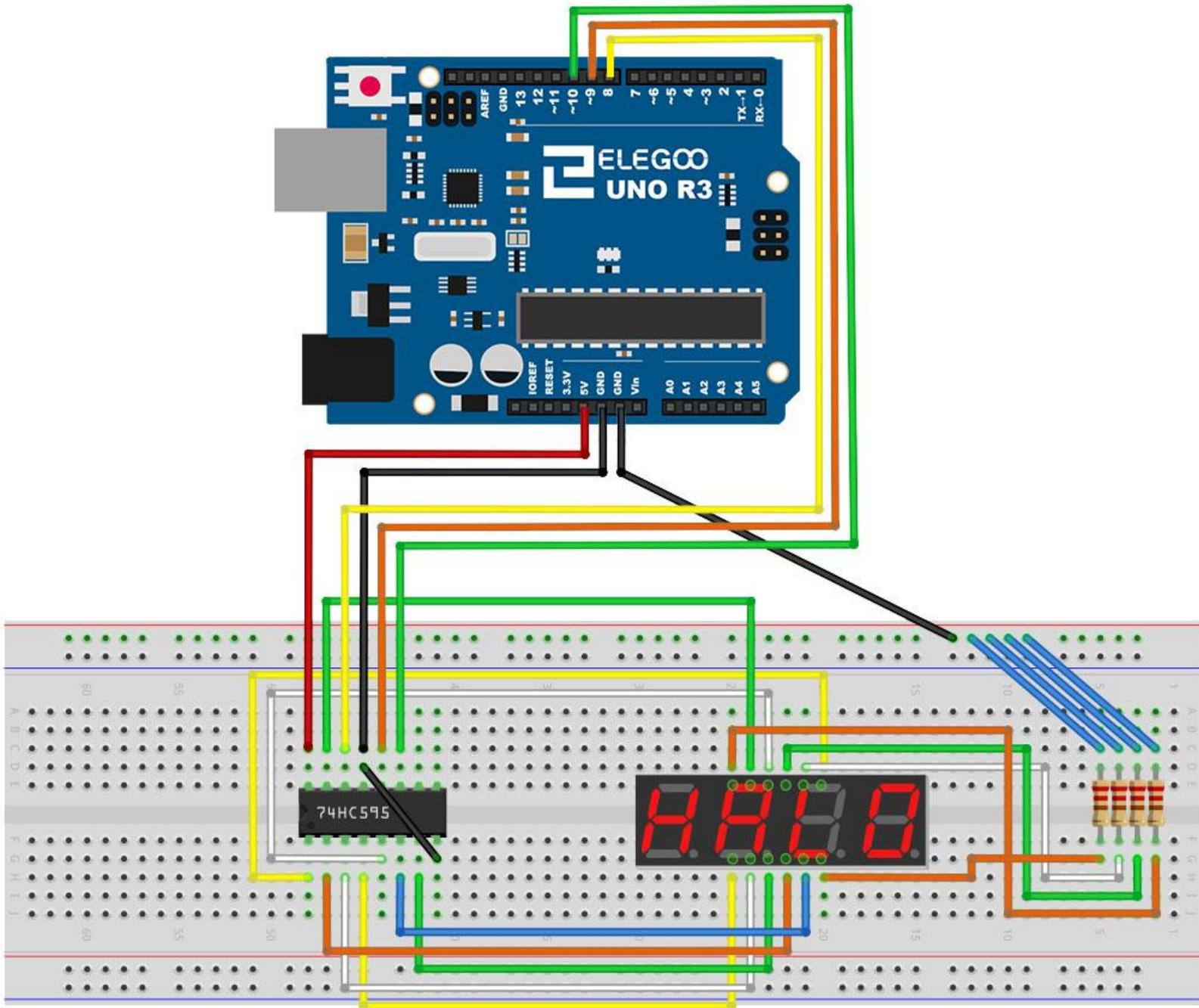
5643B

Four Digits Displays Series

Verbindungsschema



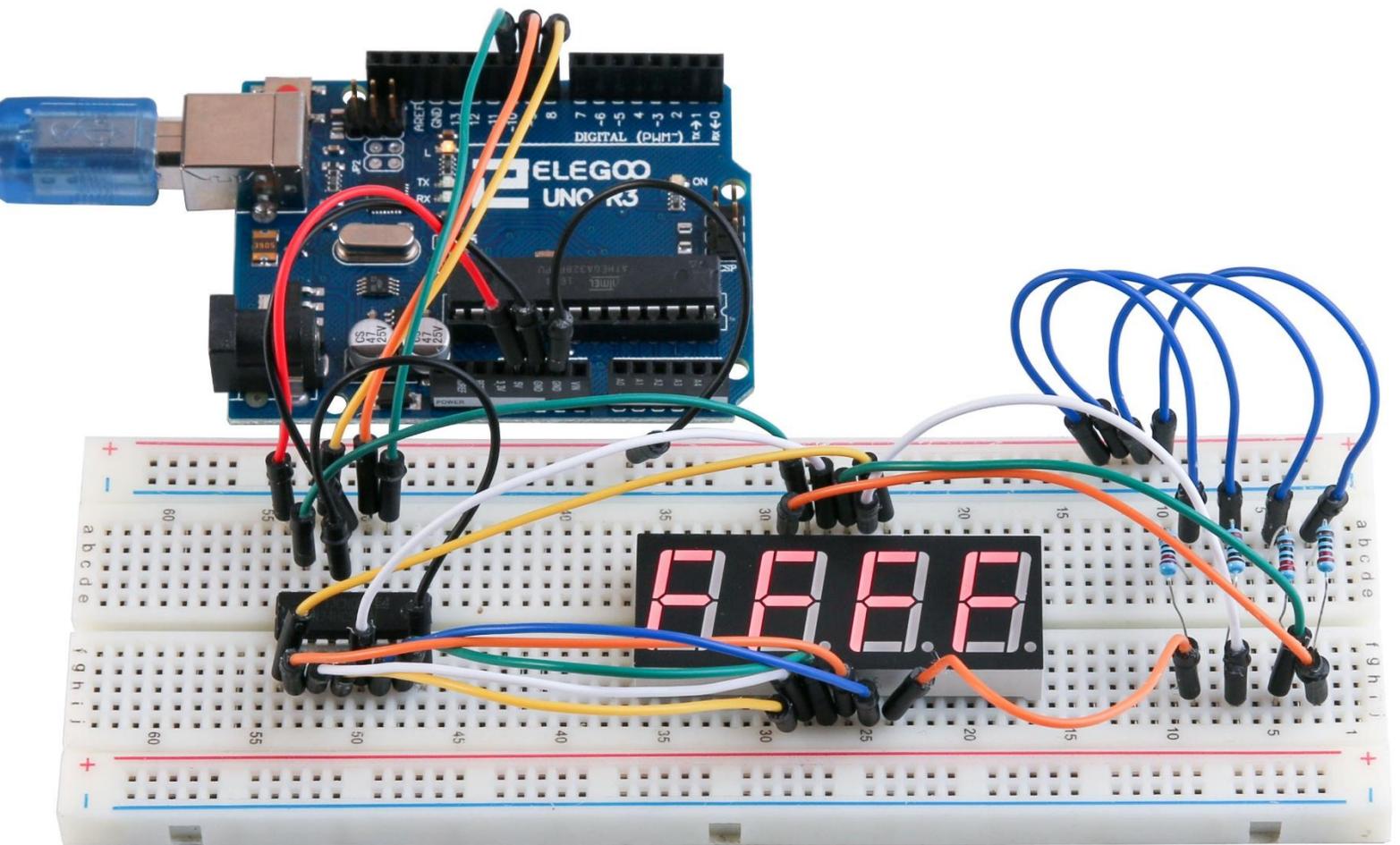
Schaltplan



Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 28 Four Digital Seven Segment Display*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Beispielbild



Lektion29: Gleichstrommotor

Übersicht

In dieser Lektion lernen Sie, wie man einen kleinen DC Motor (DC = direct current = Gleichstrom) mit Hilfe eines Transistors mit einem UNO Board benutzen.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x L293D IC
- (1) x 3-6V DC Motor mit Lüfterblättern
- (5) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)
- (1) x Breadboard Stromversorgungsmodul
- (1) x 9V1A Netzteil

Einführung in die Komponenten

Breadboard Stromversorgung

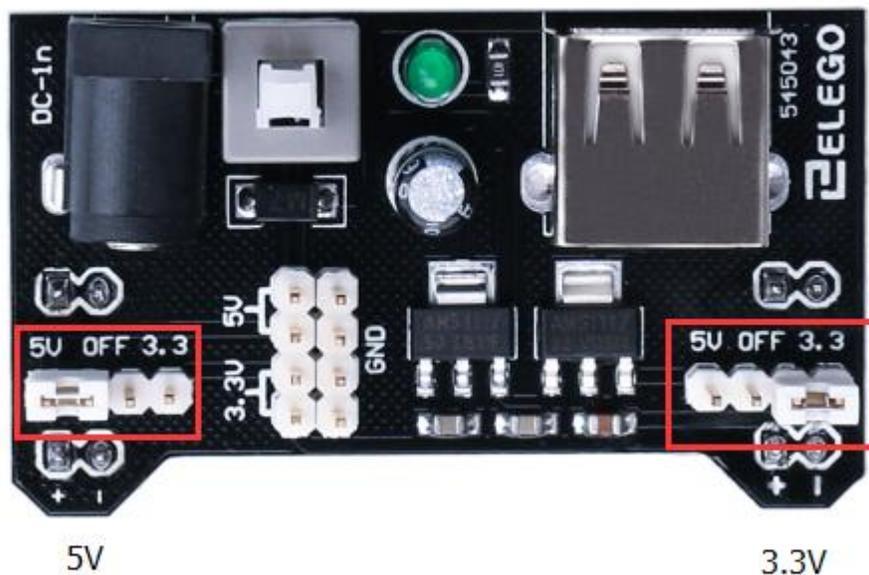
Der kleine DC Motor verbraucht mehr Strom, als unser UNO R3 Board über seine Ausgänge bereitstellen kann. Um das Board nicht zu beschädigen, verwenden wir ein externes Netzteil.



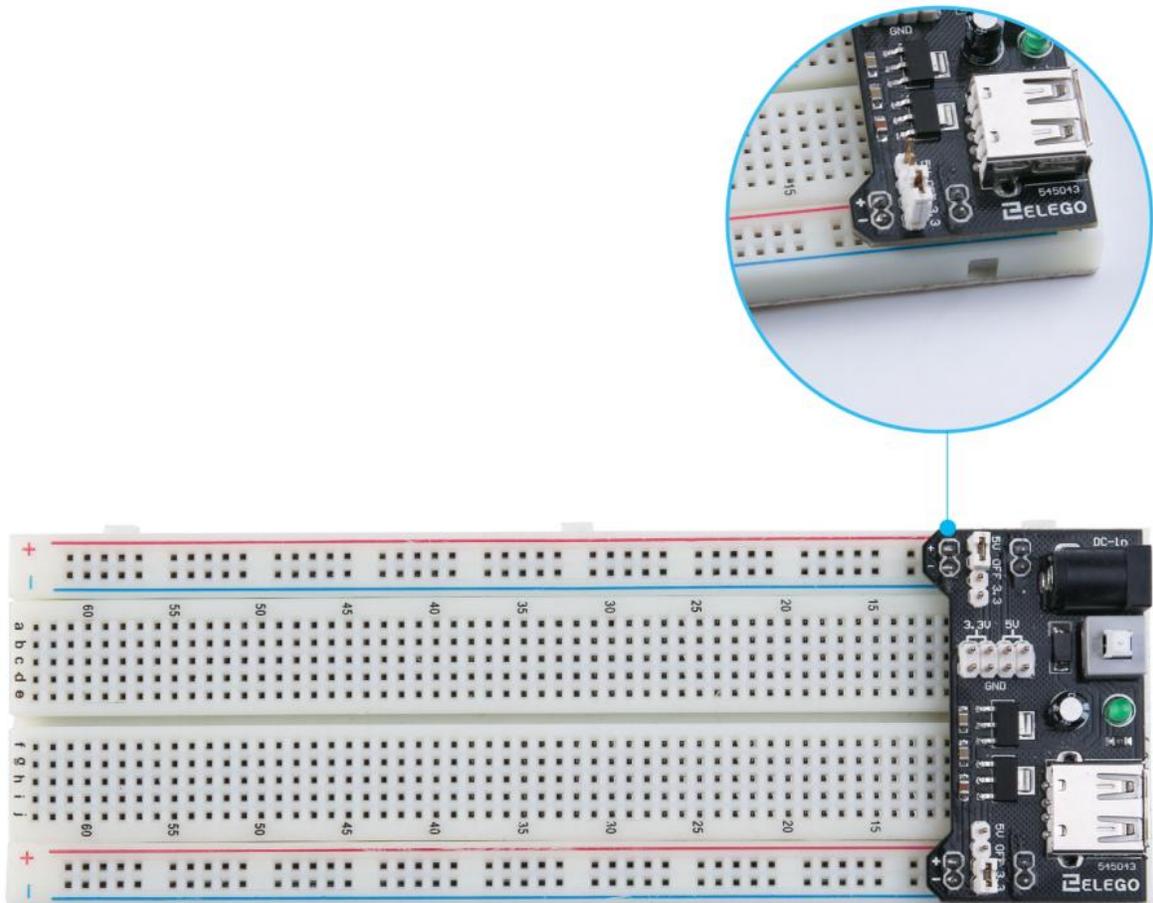
Produktspezifikationen:

- Einrastender Ein/Aus-Schalter
- LED Betriebsanzeige
- Eingangsspannung: 6,5-9V(DC) an 5,5mmx2,1mm Buchse
- Ausgangsspannung: 3,3V/5V
- Maximaler Ausgangsstrom: 700mA
- Unabhängige Stromschienen für das Breadboard: 0V, 3,3V, 5V
- Header Pins zum einfachen Verbinden
- Größe: 5,3cm x 3,6cm
- USB-Anschluss zur Stromversorgung von externen Geräten

Ausgangsspannung einstellen:



Die linke und rechte Ausgangsspannung kann unabhängig voneinander eingestellt werden. Um die Ausgangsspannung einzustellen, müssen sie den Jumper auf die entsprechenden Pins setzen. Beachten Sie: Wenn beide Stromschienen auf „OFF“ geschaltet sind, werden die Breadboard Anschlüsse nicht mit Strom versorgt und es wird nicht funktionieren.

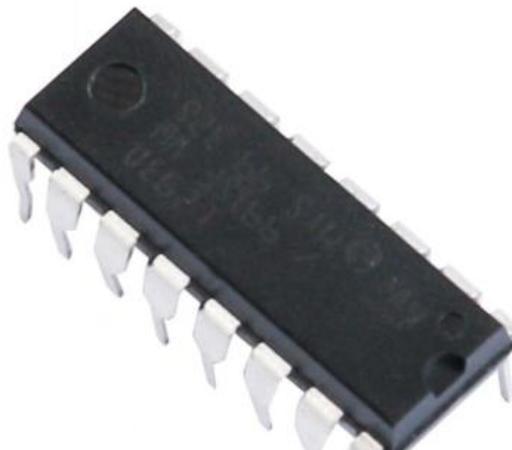


Wichtiger Hinweis:

Stellen Sie sicher, dass Sie das Modul korrekt auf dem Breadboard ausrichten. Der negative Pin des Moduls muss sich an der blauen Linie (-) und der positive Pin des Moduls muss sich an der roten Linie (+) des Breadboards befinden. Eine falsche Ausrichtung könnte dafür sorgen, dass der Strom verkehrt herum angeschlossen wird.

L293D

Der L293D ist ein vielseitiger Chip. Er kann zwei Motoren unabhängig von einander steuern. Wir benutzen in dieser Lektion nur eine Hälfte des Chips, die meisten Pins auf der rechten Seite des Chips werden nur für die Steuerung eines zweiten Motors benötigt.



Produktspezifikationen:

- Variable Spannungsversorgung: 4,5 V bis 36 V
- Separater Logikeingang
- Interner ESD-Schutz
- Überhitzungsschutz
- Gute Eingangsstörsicherheit
- Ausgangsstrommax. 1000mA pro Kanal
- Spitzenausgangsstrom 2000 mA

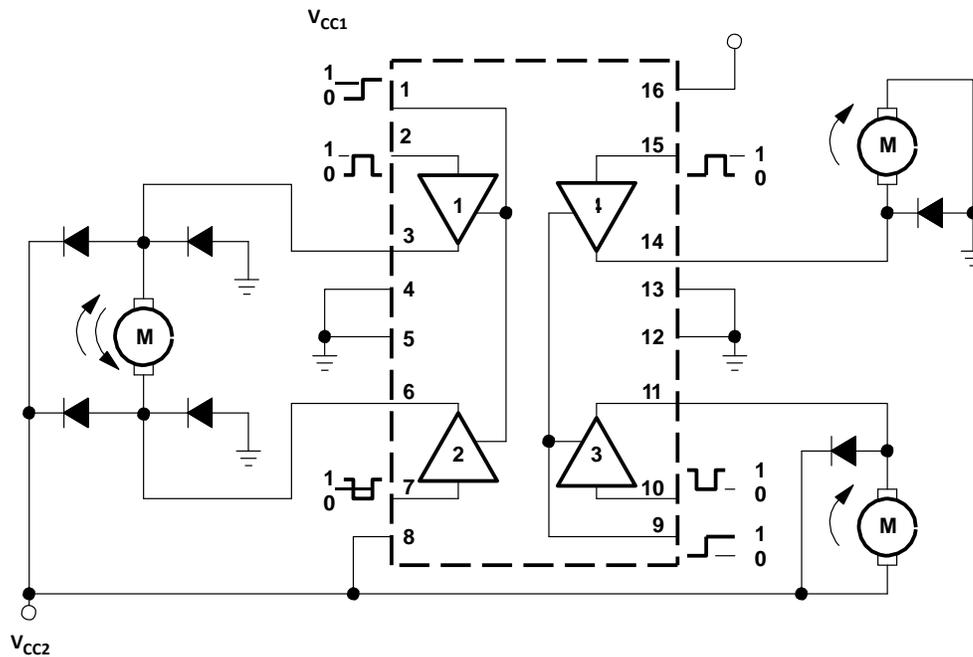


Beschreibung

Der L293D ist ein 4-Kanal Hochstromtreiber. Er ist dazu konstruiert bidirektionale Ströme von bis zu 1000 mA bei Spannungen von 4,5V bis 36V zu generieren. Der L293D ist dazu geeignet induktive Lasten wie Relays, Magnetspulen, DC- und bipolare Schrittmotoren und weitere Lasten mit hohem Strom- bzw Spannungsbedarf zu betreiben.

Alle Eingänge sind *TTL* kompatibel. Jeder Ausgang ist ein Totem-Pole-Ausgang mit einem Darlington Transistor und einer pseudo-Darlingtonquelle. Die Treiber sind in Paaren geschaltet. Die Treiber 1 und 2 sind zu 1,2EN geschaltet und die Treiber 3 und 4 zu 3,4EN. Wenn ein Aktivierungseingang HIGH geschaltet ist, werden die entsprechenden Treiber aktiviert und ihre Ausgänge eingeschaltet und in Phase mit ihren Eingängen gesetzt. Wenn der Aktivierungseingang LOW ist, werden die entsprechenden Treiber deaktiviert und ihre Ausgänge ebenfalls ausgeschaltet und in einen Hochimpedanzmodus versetzt. Mit den richtigen Dateneingängen formen beide Treiberpaare ein volles H (bzw Brücke), passend für Magnetspulen bzw Motoren.

Blockdiagramm



Wir haben uns mit nicht entzifferbaren Pinbelegungsdiagrammen herumgeärgert, also entschieden wir uns dafür ein eigenes Diagramm zu erstellen, das die Informationen übersichtlicher wiedergibt.

Es gibt 3 Verbindungen zum Arduino, 2 Verbindungen zum Motor und eine Verbindung zum Netzteil.

L293D

M1 PWM	1	16	Battery +ve
M1 direction 0/1	2	15	M2 direction 0/1
M1 +ve	3	14	M2 +ve
GND	4	13	GND
GND	5	12	GND
M1 -ve	6	11	M2 -ve
M1 direction 1/0	7	10	M2 direction 1/0
Battery +ve	8	9	M2 PWM

Motor 1

Motor 2

Wie man dieses Pinbelegungsdiagramm verwendet:

Die linke Seite versorgt den ersten Motor und die zweite Seite kann einen zweiten Motor

versorgen. Es funktioniert natürlich auch mit nur einem angeschlossenen Motor.

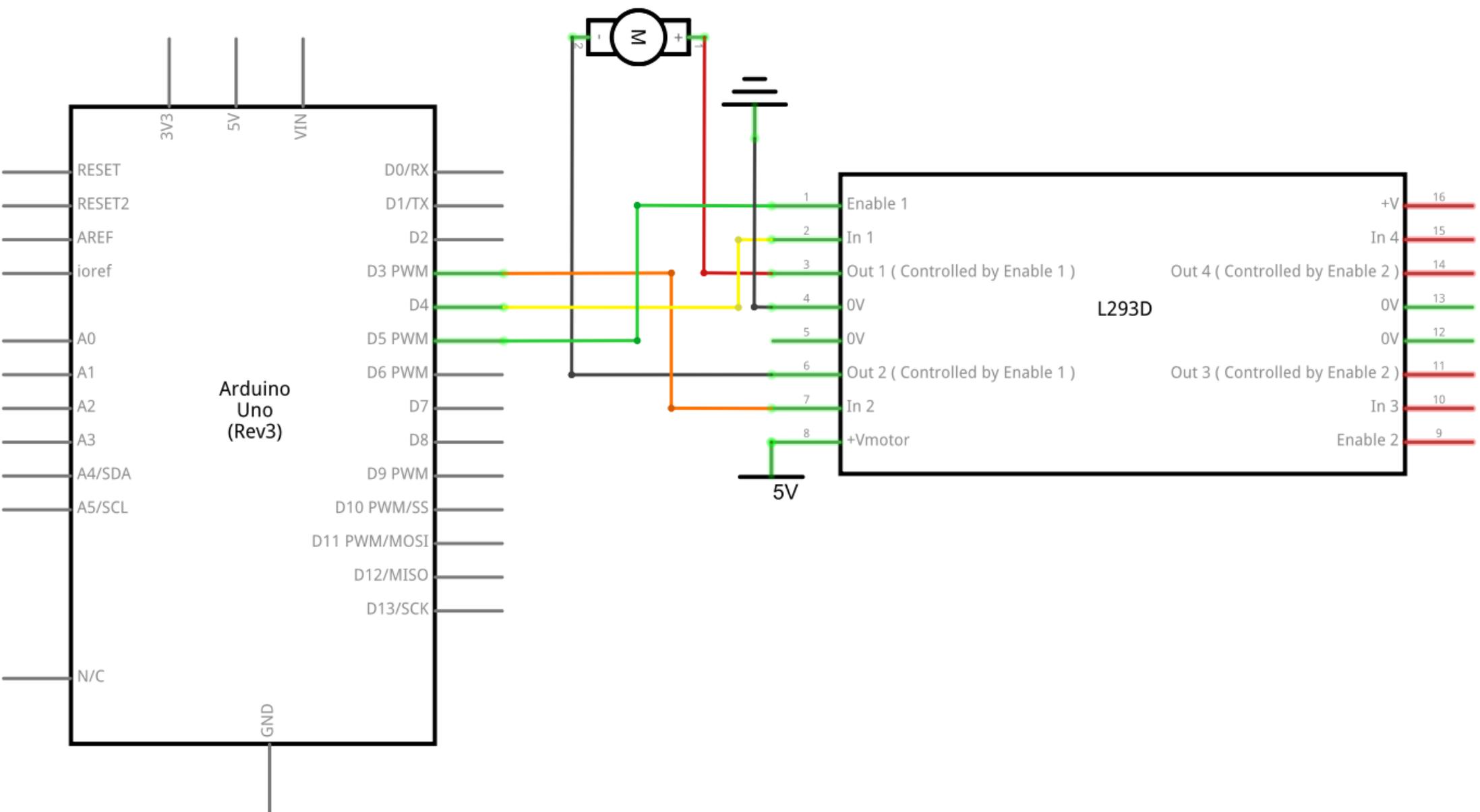
Arduino Verbindungen

Der *M1 PWM* des Chips wird mit einem beliebigen digitalen PWM Pin des Arduinos verbunden, z.B. *Pin 5*. Der Ausgangswert zwischen 0 und 255 bestimmt die Geschwindigkeit des Motors, wobei 0 ausgeschaltet wäre, 128 die Hälfte der Geschwindigkeit und 255 die maximale Geschwindigkeit.

M1 direction 0/1 und *M1 direction 1/0* werden jeweils mit einem digitalen Arduino Ausgang verbunden. Auf einem der Pins muss HIGH und auf dem anderen muss LOW gesendet werden, damit der Motor sich in eine Richtung dreht.

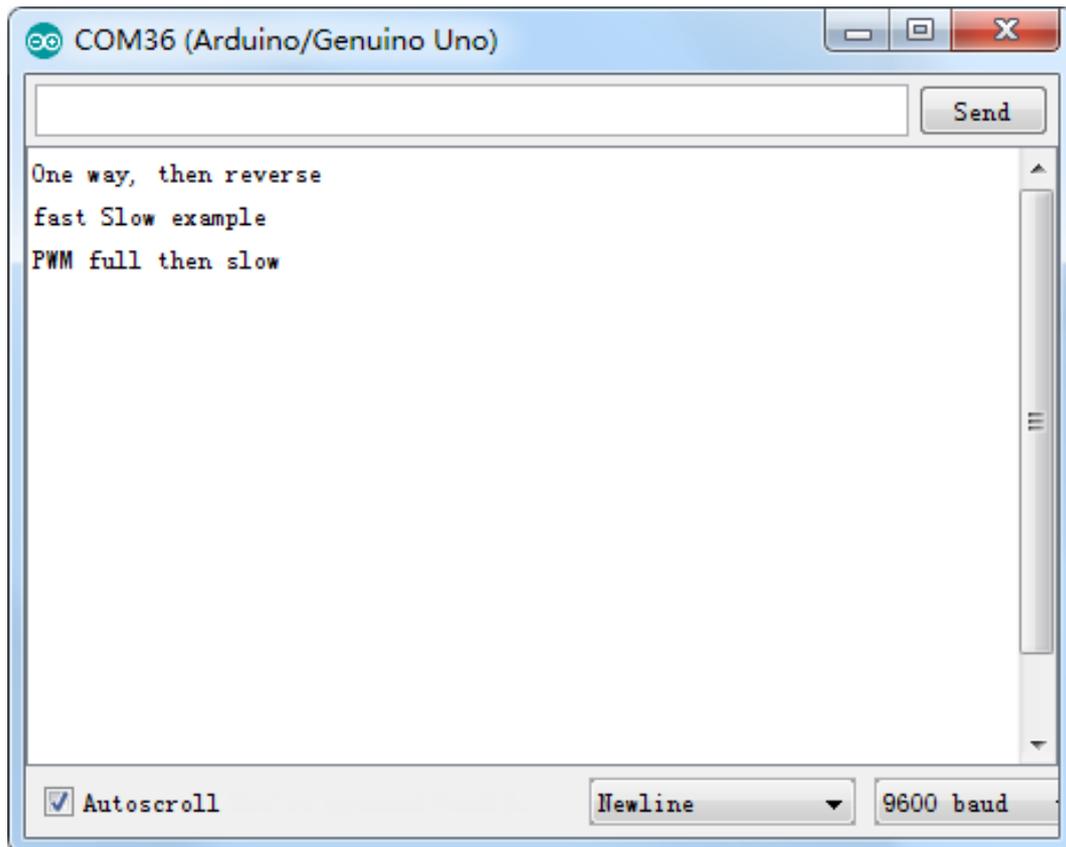
Wenn Sie die HIGH- und LOW-Zustände umschalten, dreht der Motor in die andere Richtung.

Verbindungsschema



Der untenstehende Code benutzt statt einer externen Stromversorgung den Arduino als Stromquelle. Beachten Sie, dass dies wie geschildert riskant ist und stattdessen lieber ein externes Netzteil in Kombination mit dem L293D Chip benutzt werden sollte.

Sie sollten einen Motor niemals direkt mit dem Arduino verbinden, da beim Ausschalten eines Motors ein elektrischer Strom zurückfließt. Bei einem kleinen Motor beschädigt dies den Arduino, bei einem großen passiert Schlimmeres.



```
COM36 (Arduino/Genuino Uno)
Send
One way, then reverse
fast Slow example
PWM full then slow
Autoscroll Newline 9600 baud
```

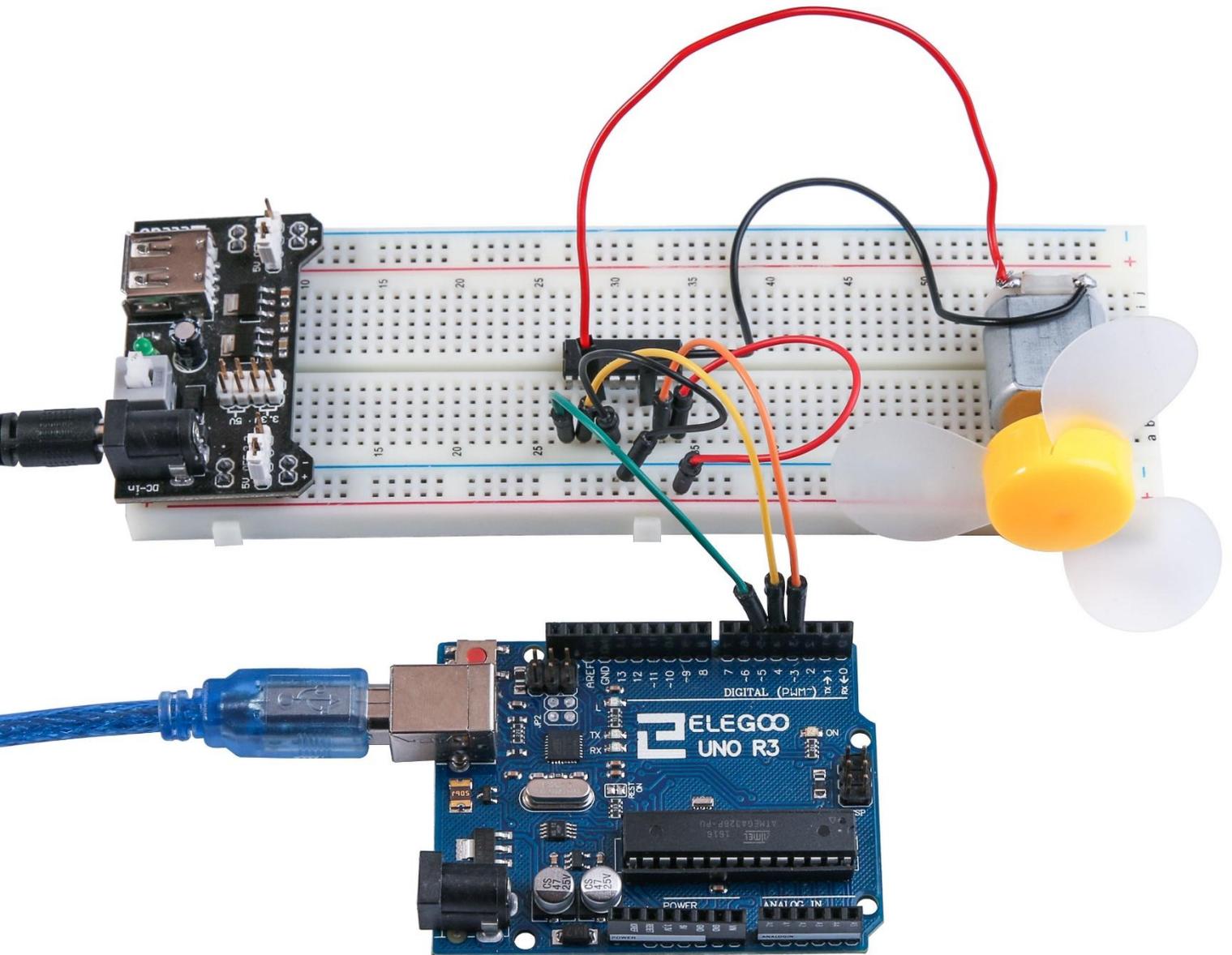
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 29 DC Motors“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Nach dem Hochladen des Sketches schalten sie alle Schalter der Stromversorgung ein. Der Motor wird am Anfang fünfmal leicht im Uhrzeigersinn und anschließend gegen den Uhrzeigersinn drehen. Danach fängt er an sich schnell in Richtung Uhrzeigersinn zu drehen, macht eine kurze Pause und dreht sich dann schnell in Richtung gegen den Uhrzeigersinn. Dann sendet das Board zur Steuerung ein PWM-Signal, wobei der Motor seine Geschwindigkeit immer weiter reduzieren

und danach wieder ansteigen lassen wird. Nach einer 10s-Pause geht es von vorne los.

Beispielbild



Lektion 30:Relais

Übersicht

In dieser Lektion lernen Sie, wie man ein Relay benutzt.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x 3-6V DC Motor mit Lüfterblättern
- (1) x L293D IC
- (1) x 5VRelais
- (1) x Breadboard Stromversorgungsmodul
- (1) x 9V1A Netzteil
- (8) x M-M Kabel (Männlich zu MännlichDuPont JumperKabel)



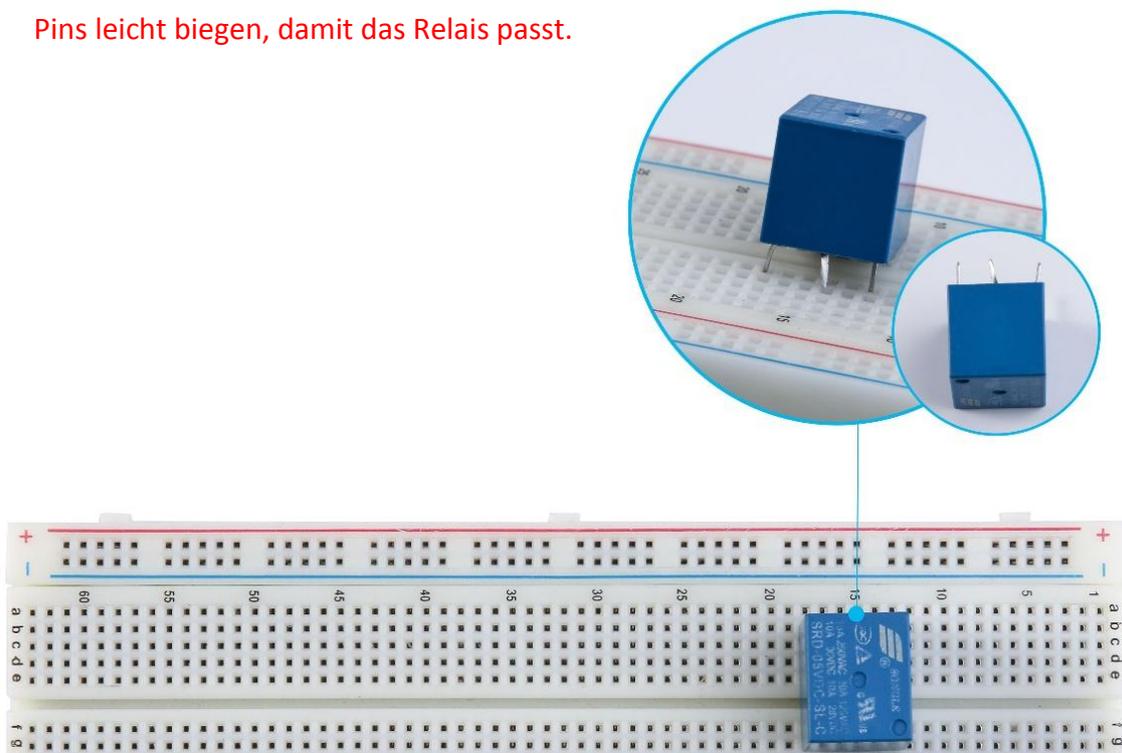
Einführung in die Komponenten

Relais:

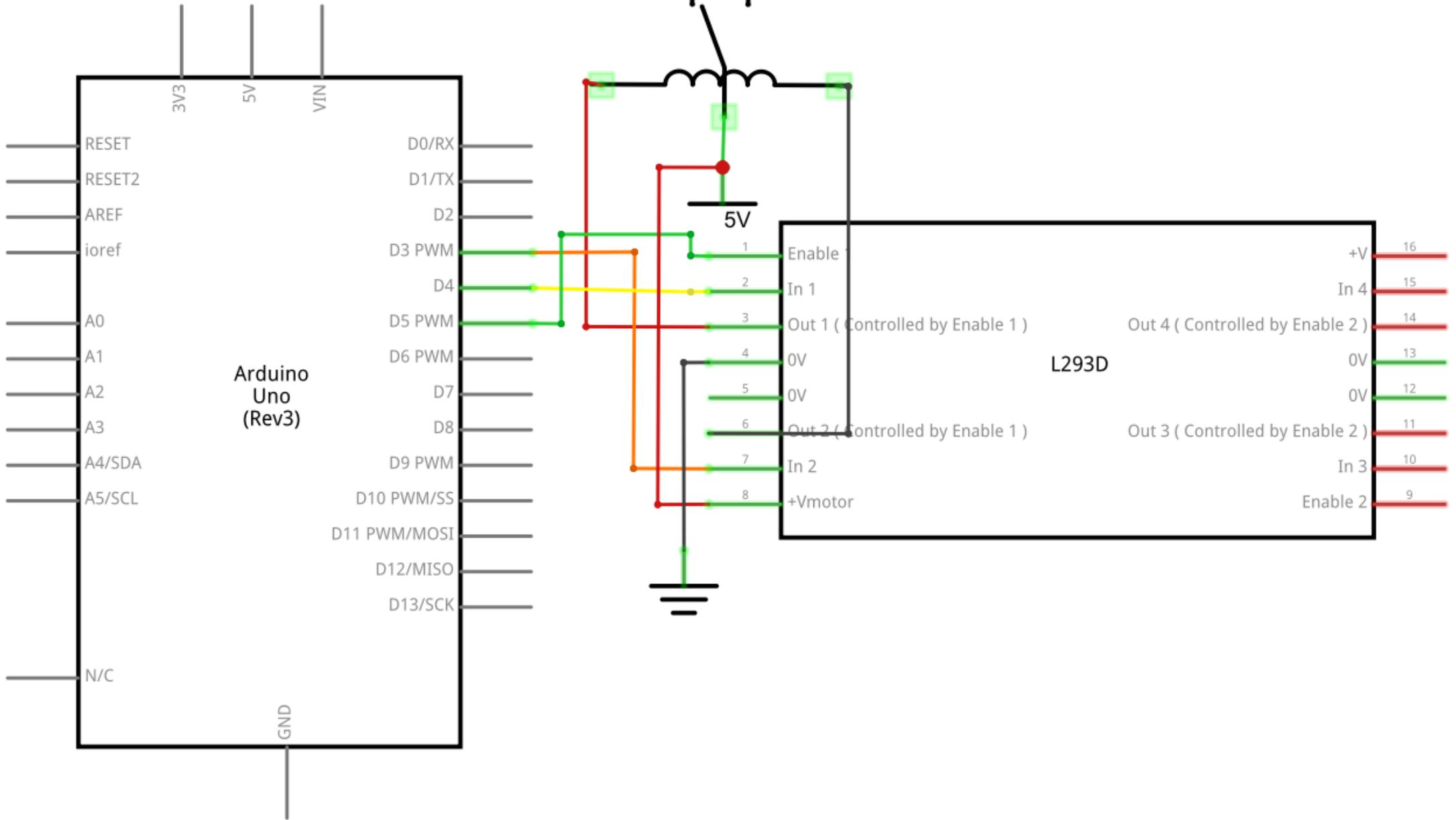
Ein Relais ist ein durch Elektrizität ausgelöster Schalter. Viele Relais benutzen einen Elektromagneten, der bei angelegter Spannung mechanisch den Schaltkreis schließt. Ein anderes Prinzip wird in Halbleiterrelais / Solid State Relais (SSRs) benutzt. Relais benutzt man, wenn man den Schaltkreis mit einem schwächeren Steuerstromkreis schalten will oder aber, wenn man mit einem Steuersignal viele Schaltkreise auf einmal schalten will. Der Steuerstromkreis und der zu steuernde Stromkreis sind dabei elektrisch vollkommen voneinander isoliert. Die ersten Relais wurden in Langstreckentelegraphen als Verstärker benutzt. Sie verstärkten das Signal, das sie von einem anderen Schaltkreis erhielten und übertrugen es auf einen anderen Schaltkreis. Relais wurden massenweise bei Anrufvermittlungsstellen und frühen Computermodellen eingesetzt, um logische Operationen auszuführen.

Eine Relais Art, die die hohen Ströme schalten kann, die zum Betrieb eines elektrischen Motors oder einer anderen hochstromigen Last nötig sind, sind die sogenannten Schützrelais. Halbleiterrelais dagegen steuern Ströme, ohne dass sie bewegliche Teile benötigen. Stattdessen wird ein Halbleiterchip zum Schalten verwendet. Relais mit speziell angepassten Betriebseigenschaften und manchmal sogar mehreren Spulen werden benutzt, um elektrische Schaltkreise vor Überlastungen und Fehlern zu schützen. In modernen elektrischen Systemen werden diese Funktionen von sogenannten Schutzrelais übernommen. Unten sehen sie, wie man ein Relay am Arduino betreibt.

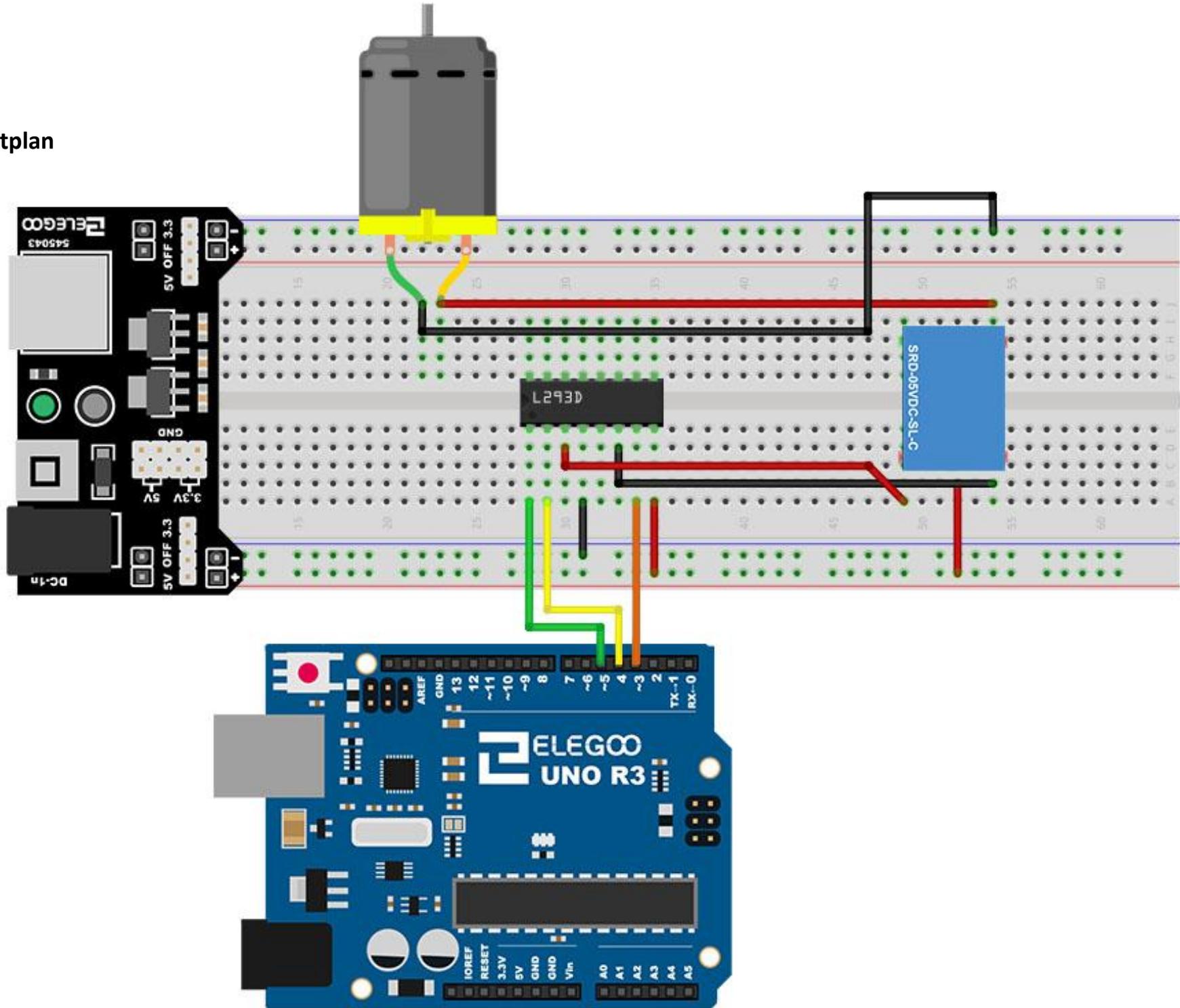
Wahrscheinlich werden Sie Schwierigkeiten dabei haben das Relais auf das Breadboard zu setzen. Wie Sie im Bild unten sehen können, müssen Sie einen der Pins leicht biegen, damit das Relais passt.



Verbindungsschema



Schaltplan

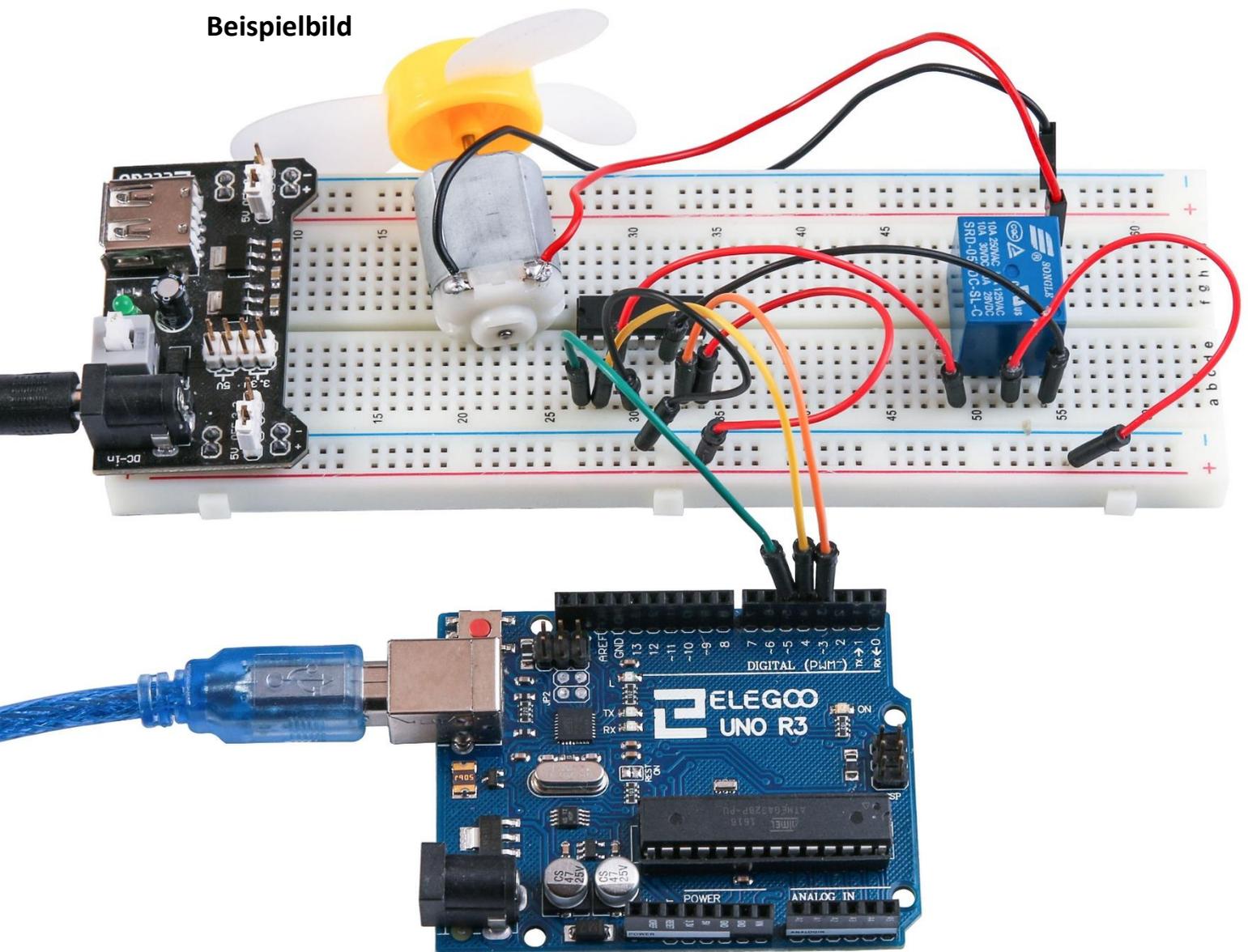


Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 30 Relay“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Nach dem Hochladen des Programmes, schalten sie die Stromversorgung ein. Das Relais wird schalten und dabei ein Klick-Geräusch als Feedback von sich geben, worauf der Motor anfängt sich zu drehen. Nach einer Zeit wird das Relais ausgeschaltet und der Motor stoppt.

Beispielbild



Lektion31:Schrittmotor

Übersicht

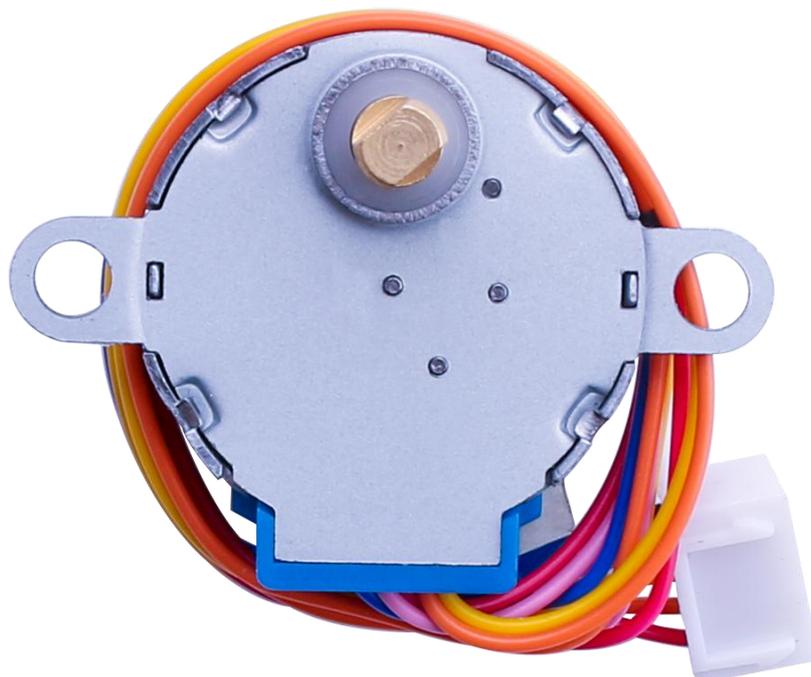
In dieser Lektion lernen sie einen einfachen Weg kennen einen Schrittmotor zu betreiben. Der Stepper kommt mit einem integrierten Treibermodul, was es für uns einfach macht ihn mit unserem UNO Board zu verbinden.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x ULN2003 Schrittmotortreibermodul
- (1) x Schrittmotor
- (1) x 9V1A Netzteil
- (1) x Breadboard Stromversorgungsmodul
- (6) x W-M Kabel (Weiblich zu Männlich DuPont JumperKabel)
- (1) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

Einführung in die Komponenten

Schrittmotor



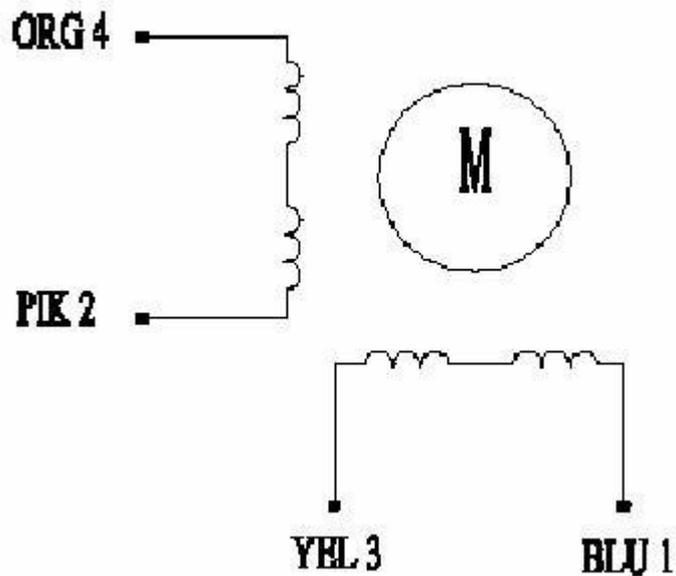
Ein Schrittmotor ist ein elektromechanisches Gerät, das elektrische Pulse in diskrete mechanische Bewegungen umwandelt. Die Axe des Motors dreht in kleinen mechanischen Schritten, wenn ein passender elektrischer Puls an ihn gesendet wird. Die Rotation des Motors ist stark abhängig von dem Eingangssignal bzw den Pulsen. Die Sequenz der Pulse bestimmt direkt die Richtung der Rotation, während die Länge der Rotation von der Anzahl und Frequenz der Pulse abhängt. Eine der größten Vorteile von Schrittmotoren ist, dass sie präzise in einem Schleifensystem gesteuert werden können. Das ist möglich, weil kein Feedback über die Position benötigt wird, da man immer genau vorher weiß, auf welche Position sich der Motor drehen wird. Diese Art der Steuerung macht Sensoren und Rückmeldegeräte wie optische Encoder überflüssig. Die Position kann ermittelt werden, in dem man die gesendeten Pulse im Auge behält.

Schrittmotor 28BYJ-48 Parameter

- Modell: 28BYJ-48
- Nennspannung: 5VDC
- Phasenanzahl: 4
- Geschwindigkeitsänderungsverhältnis: 1/64
- Schrittwinkel: $5,625^\circ / 64$
- Frequenz: 100Hz
- DC Widerstand: $50\Omega \pm 7\% (25^\circ\text{C})$
- Leerlaufinfraktionsfrequenz: $> 600\text{Hz}$
- Leerlaufauszugsfrequenz: $> 1000\text{Hz}$
- Wechselwirkungsmoment $> 34.3\text{mN.m} (120\text{Hz})$
- Selbstpositionierungsmoment $> 34.3\text{mN.m}$
- Reibungsmoment: 600-1200 gf.cm
- Kippmoment: 300 gf.cm
- Isolierter Widerstand: $> 10\text{M}\Omega (500\text{V})$
- Isolierte elektrische Leistung: 600VAC/1mA/1s
- Isolierungsgrad: A
- Temperaturanstieg: $< 40\text{K} (120\text{Hz})$
- Rauschen: $< 35\text{dB} (120\text{Hz}, \text{No load}, 10\text{cm})$

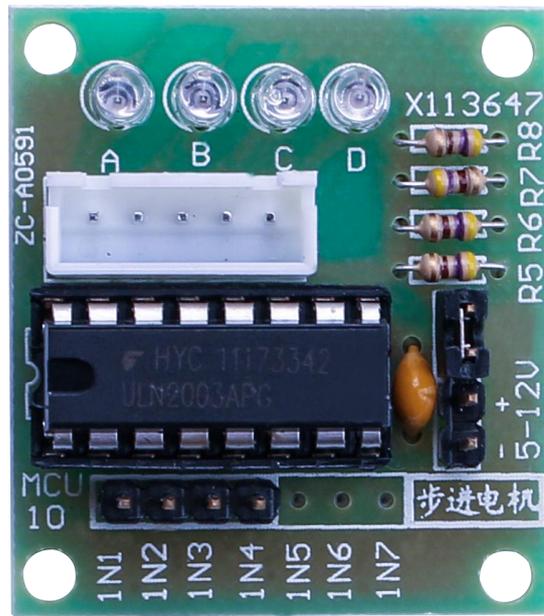
Schnittstellenschaltkreise

WIRING DIAGRAM



Der bipolare Schrittmotor hat normalerweise vier Anschlüsse. Im Gegensatz zu unipolaren Schrittmotoren, haben bipolare keine gemeinsame zentrale Verbindung, sondern stattdessen zwei unabhängige Spulen. Sie können die beiden Arten von einander unterscheiden, indem Sie den Widerstand zwischen den Anschlüssen messen. Sie sollten bei der Messung zwei Kabelpaare mit gleichem Widerstand finden. Dann handelt es sich um einen bipolaren Schrittmotor. Wenn sie an zwei nicht miteinander verbundenen Kabeln den Widerstand messen, wird ihnen ein unendlicher Widerstand bzw gar nichts angezeigt.

ULN2003 Treibermodul



Modulbeschreibung

- Größe: 42mmx30mm
- Verwendet den ULN2003 Treiber Chip, 500mA
- Die LEDs A, B, C und D indizieren den Status der 4 Phasen des Schrittmotors
- Der weiße Anschluss ist ein Standard-Vierphasenschrittmotor-Anschluss
- Die Stromversorgungspins sind separat zugänglich
- Die restlichen Pins des ULN2003 Chips sind für Erweiterungen verfügbar

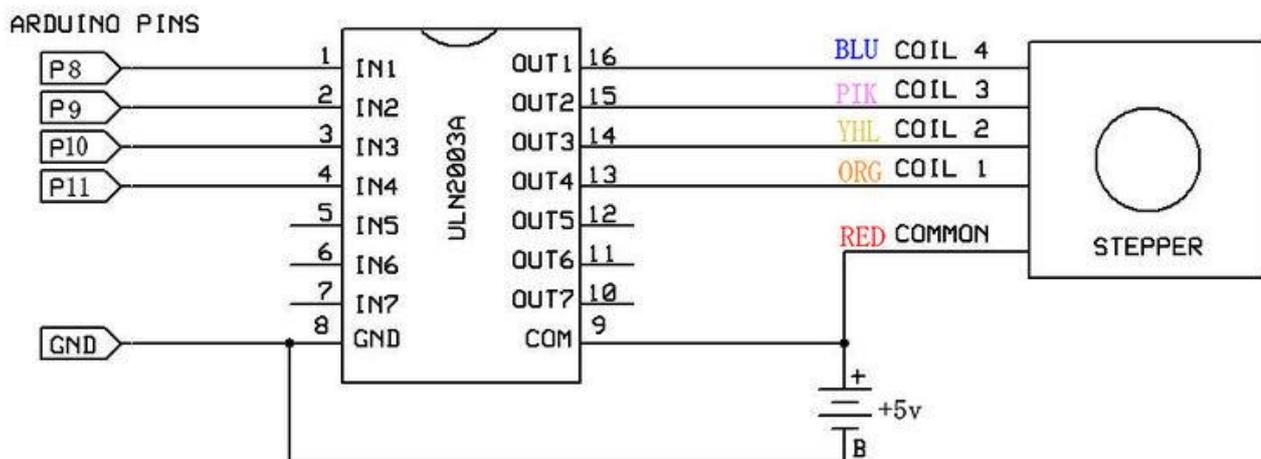
Der einfachste Weg, um einen unipolaren Schrittmotor mit einem Arduino anzusteuern ist eine Verbindung im Transistor Array des ULN2003A zu überbrücken. Der ULN2003A-Chip enthält sieben Darlington Transistoren und hat zusätzlich sieben weitere TIP120 Transistoren an Board. Er kann bis zu 500 mA pro Kanal ausgeben und hat einen internen Spannungsabfall von etwa 1V im eingeschalteten Zustand. Er enthält außerdem interne Klemmdioden, um

Spannungsspitzen zu verlagern, wenn induktive Lasten betrieben werden. Um den Schrittmotor zu kontrollieren, muss in einer speziellen Frequenz jede der Spulen mit Spannung versorgt werden.

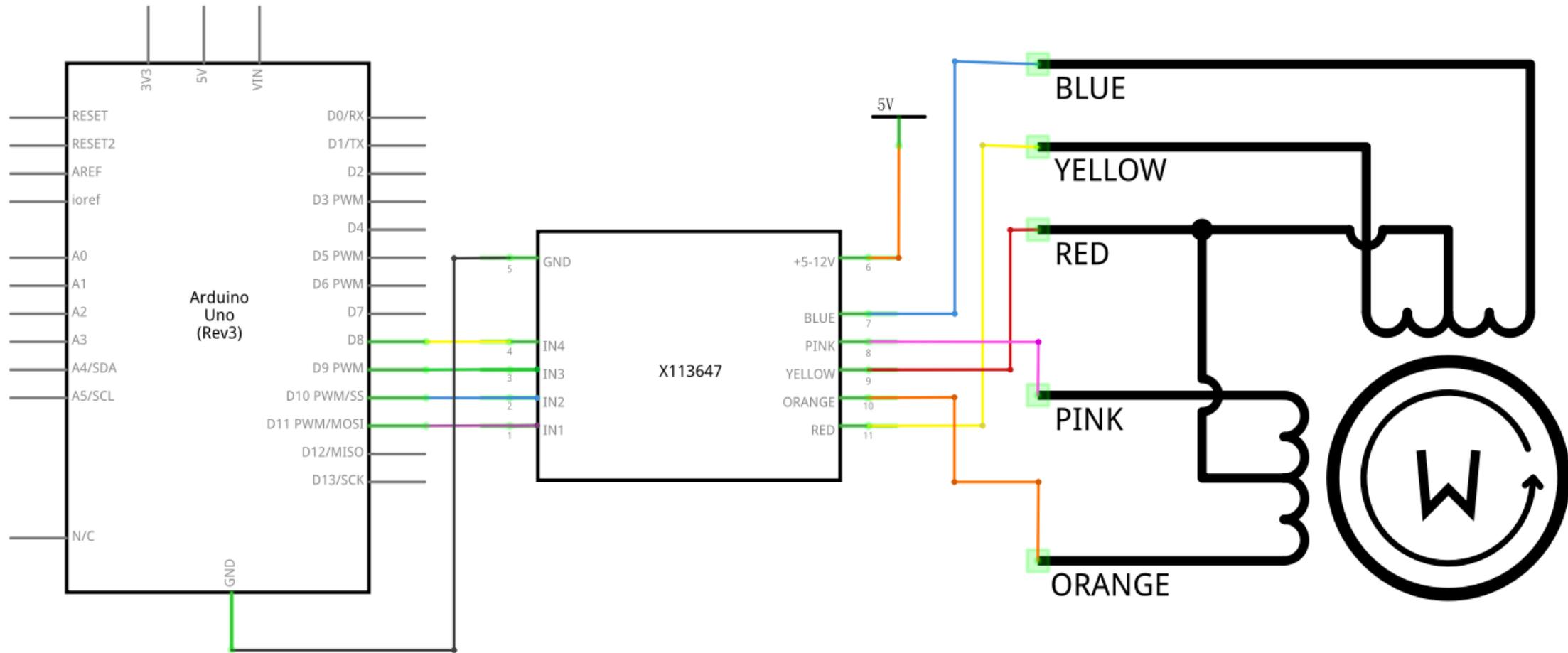
Die Sequenz würde in etwa so aussehen:

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 ORG	-	-						-
3 YEL		-	-	-				
2 PIK				-	-	-		
1 BLU						-	-	-

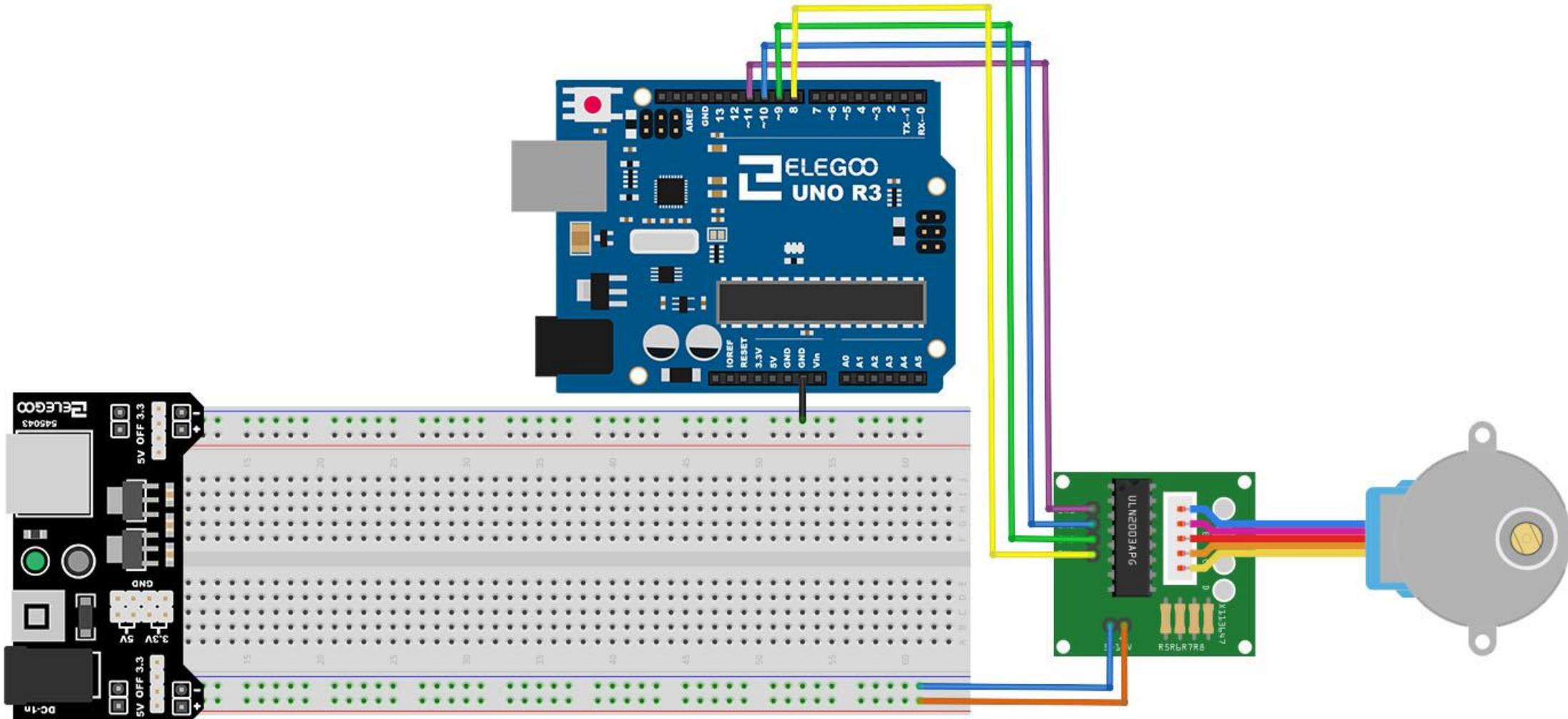
Hier sind einige Diagramme, die zeigen wie man einen unipolaren Schrittmotor mit einem ULN2003A an einem 4-Pin-Anschluss verbindet.



Verbindungsschema



Schaltplan



Wir benutzen 4 Pins, um den Schrittmotor zu steuern. Wir benutzen die *Pins 8 -11* unseres UNO Boards.

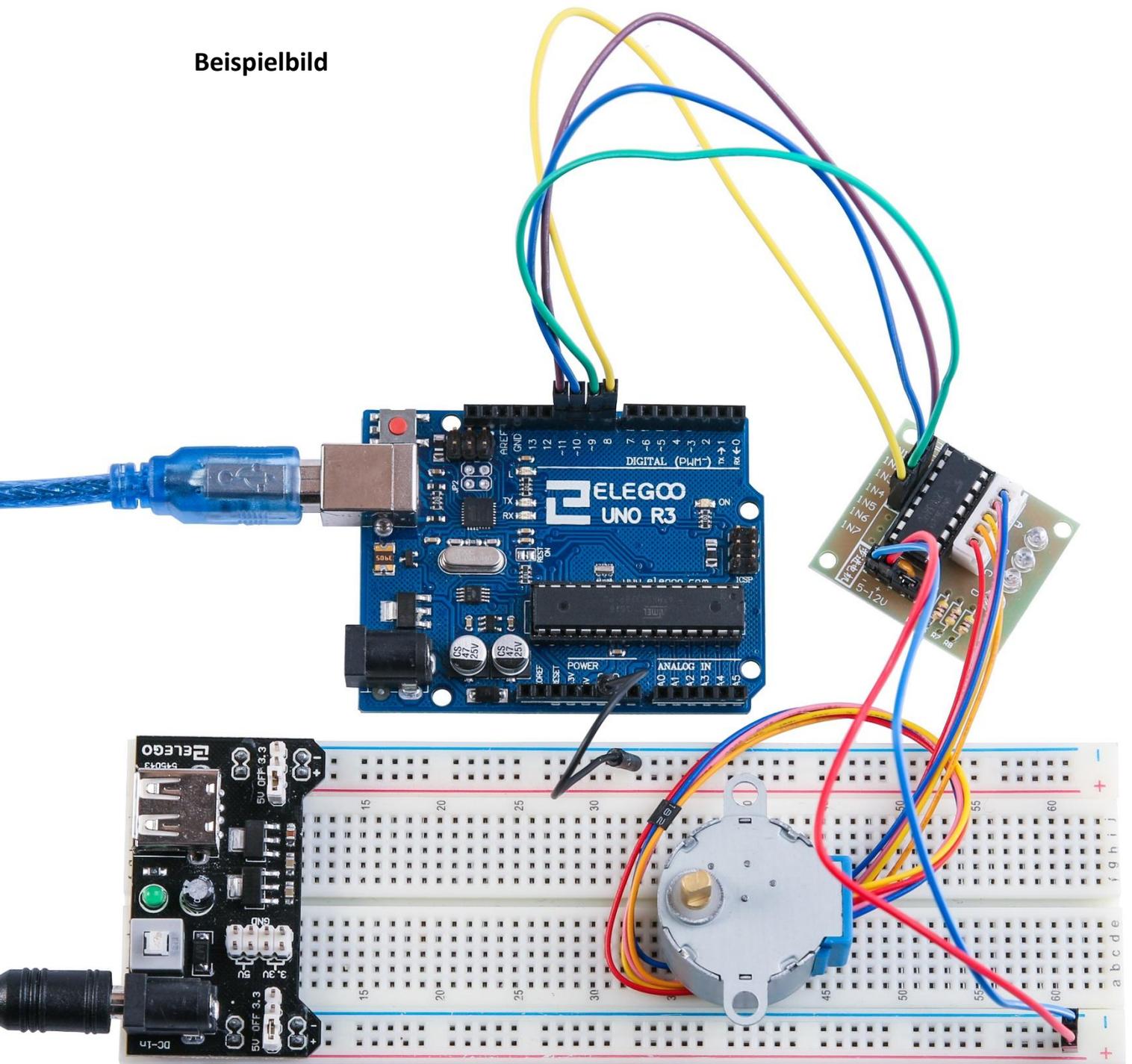
Wir verbinden GND vom Board mit dem GND-Anschluss des Moduls.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Lesson 31 Stepper Motor“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „Stepper“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Beispielbild



Lektion32:Schrittmotor mit Fernbedienung steuern

Übersicht

In dieser Lektion lernen Sie, wie man einen Schrittmotor aus Entfernung mit einer IR-Fernbedienung steuern kann.

Da der Schrittmotor ein eigenes Treibermodul hat, ist die Verbindung mit unserem UNO Board einfach.

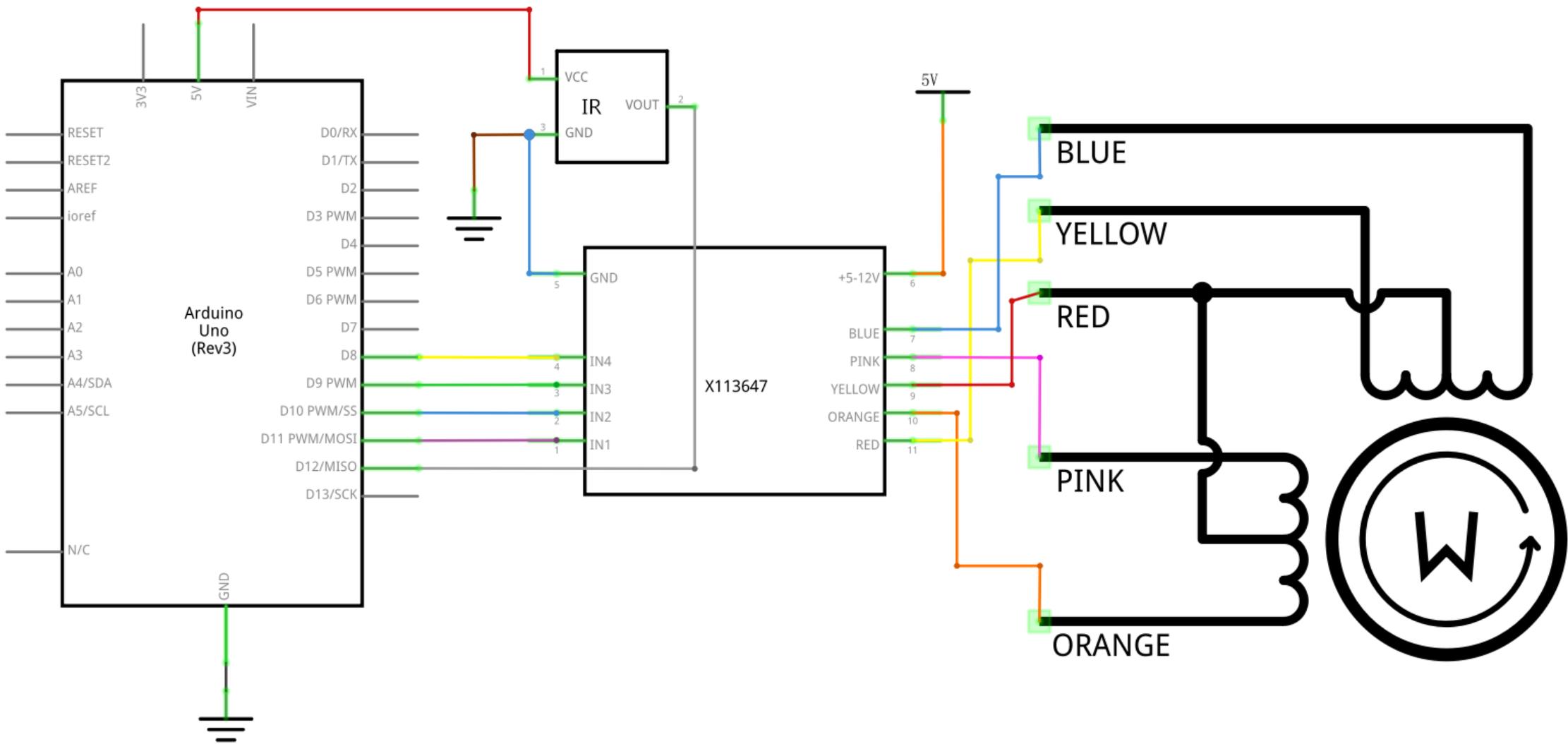
Wir wollen den Schrittmotor diesmal nicht direkt über die Stromversorgung des Boards betreiben, sondern nutzen die externe Breadboardstromversorgung mit dem 9V 1A Netzteil.

Da der Infrarotsensor kaum Strom verbraucht, wird er direkt an das UNO Board angeschlossen.

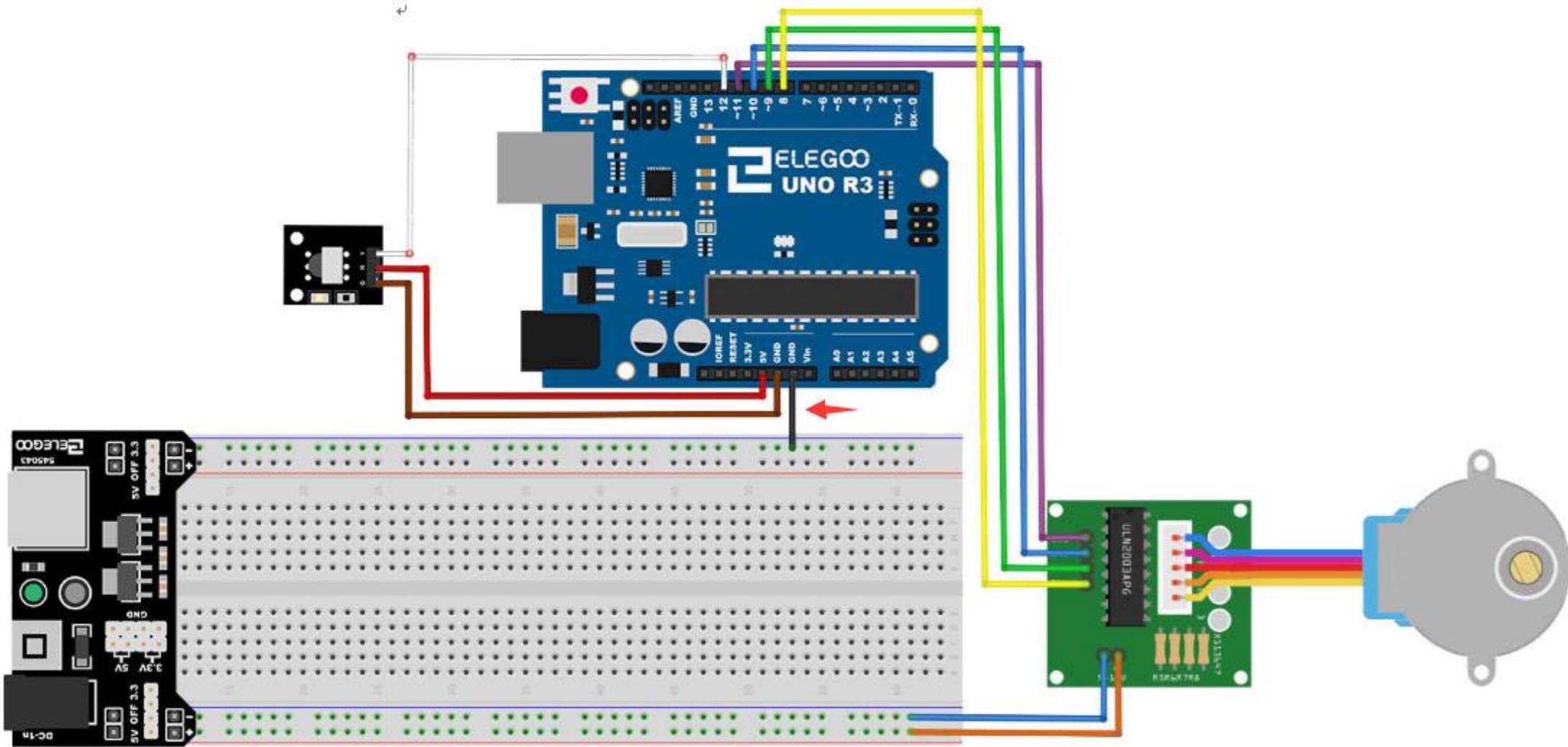
Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x IR-Empfängermodul
- (1) x IR-Fernbedienung
- (1) x ULN2003 Schrittmotortreibermodul
- (1) x Schrittmotor
- (1) x Breadboard Stromversorgungsmodul
- (1) x 9V1A Netzteil
- (9) x W-M Kabel (Weiblich zu Männlich DuPont Jumper Kabel)
- (1) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

Verbindungsschema



Schaltplan



Wir brauchen 4 Pins für den Schrittmotor und 1 Pin für den IR-Empfänger.

Den Schrittmotor schließen wir an die *Pins 8-11* und den IR-Empfänger an den *Pin 12* des UNO Boards an.

Wir verbinden zur Stromversorgung den *5V-* und den *GND* Anschluss des IR-Empfängers mit den entsprechenden Pins auf unserem Board. Um das Board nicht zu beschädigen, versorgen wir den Schrittmotor über die externe Breadboard Stromversorgung mit Strom.

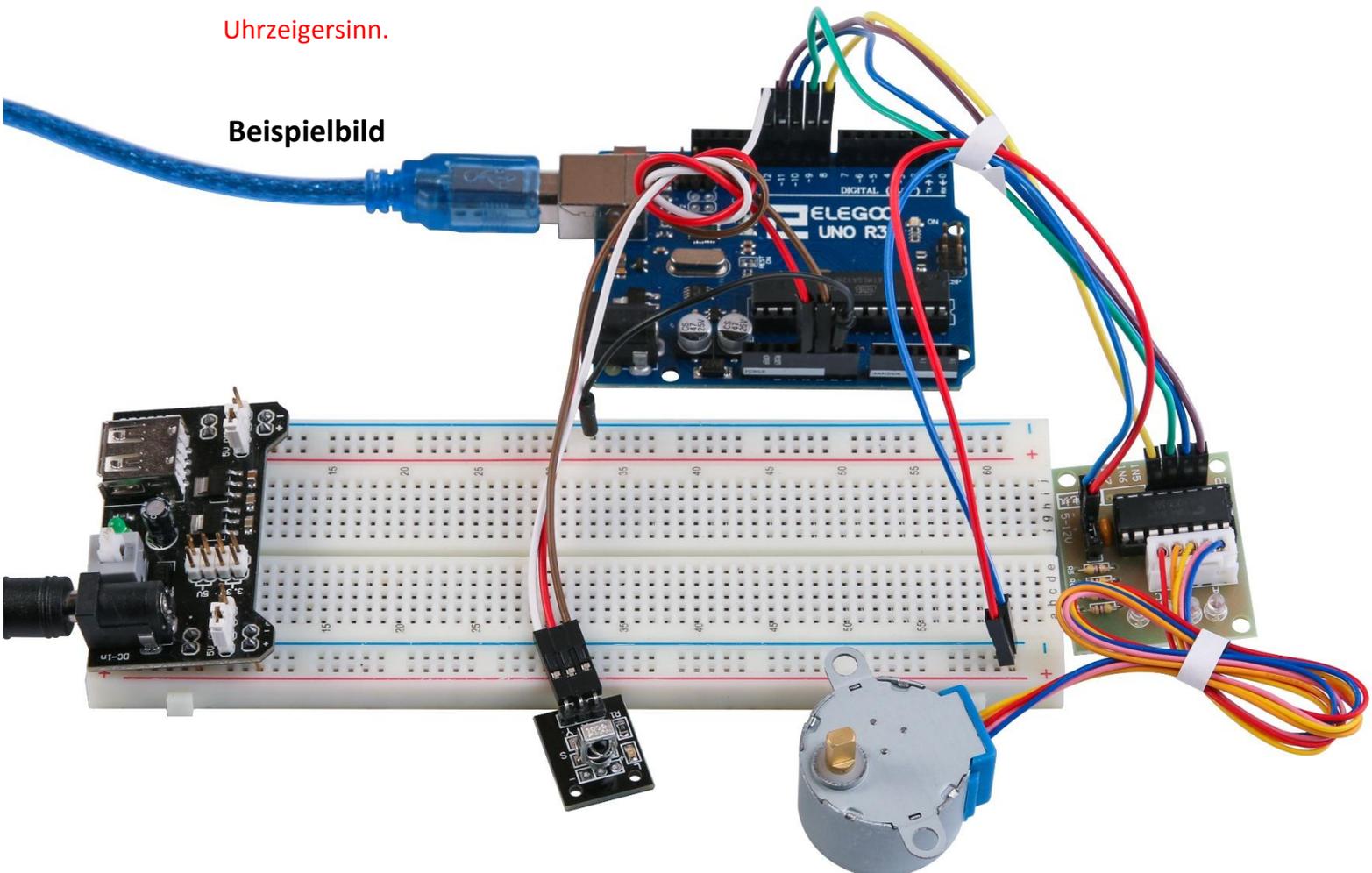
Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 32 Controlling Stepper Motor With Remote*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an. Bevor Sie diesen Sketch hochladen können, müssen Sie die „*IRremote*“- und die „*Stepper*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Der Sketch erkennt nur zwei Tasten der Fernbedienung: *VOL+* und *VOL-*.

Wenn *VOL+* gedrückt wird, dreht der Motor sich eine Umdrehung im Uhrzeigersinn. Beim Klicken auf *VOL-* macht er eine volle Umdrehung gegen den Uhrzeigersinn.

Beispielbild



Lektion 33:Schrittmotor mit Drehgeber steuern

Übersicht

In dieser Lektion lernen Sie, wie man einen Schrittmotor mit einem Drehgeber steuert. Wir verwenden wieder den 28BYJ-48 Schrittmotor mit seinem ULN2003 Stromversorgungsmodul.

Der 28BYJ-48 Motor ist nicht sehr schnell und auch nicht sehr stark, aber er eignet sich gut für Anfänger zum Experimentieren mit der Steuerung von Schrittmotoren mit Arduino. Wir erstellen einen Sketch, in dem der Motor sich in die Richtung dreht, in die wir den Drehgeber drehen. Außerdem zählt der Sketch, wie viele Schritte der Motor gegangen ist, sodass wir am Ende, durch Herunterdrücken des Drehgeberschalters, auf die exakte Startposition des Schrittmotors zurückkehren können.

Benötigte Bauteile:

- (1) x Elegoo UNO R3
- (1) x 830 Punkte Breadboard
- (1) x Drehgebermodul
- (1) x ULN2003 Schrittmotortreibermodul
- (1) x Schrittmotor
- (1) x Breadboard Stromversorgungsmodul
- (1) x 9V 1A Netzteil
- (9) x W-M Kabel (Weiblich zu Männlich DuPont Jumper Kabel)
- (1) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

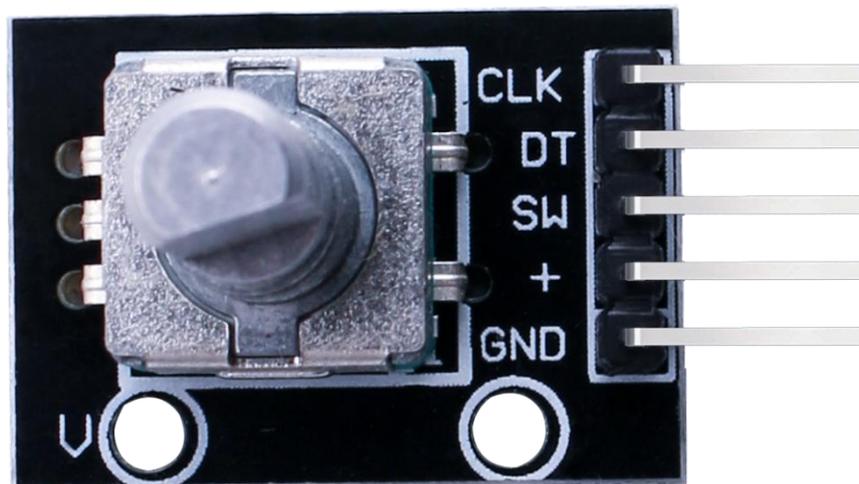
Einführung in die Komponenten

Drehgeber

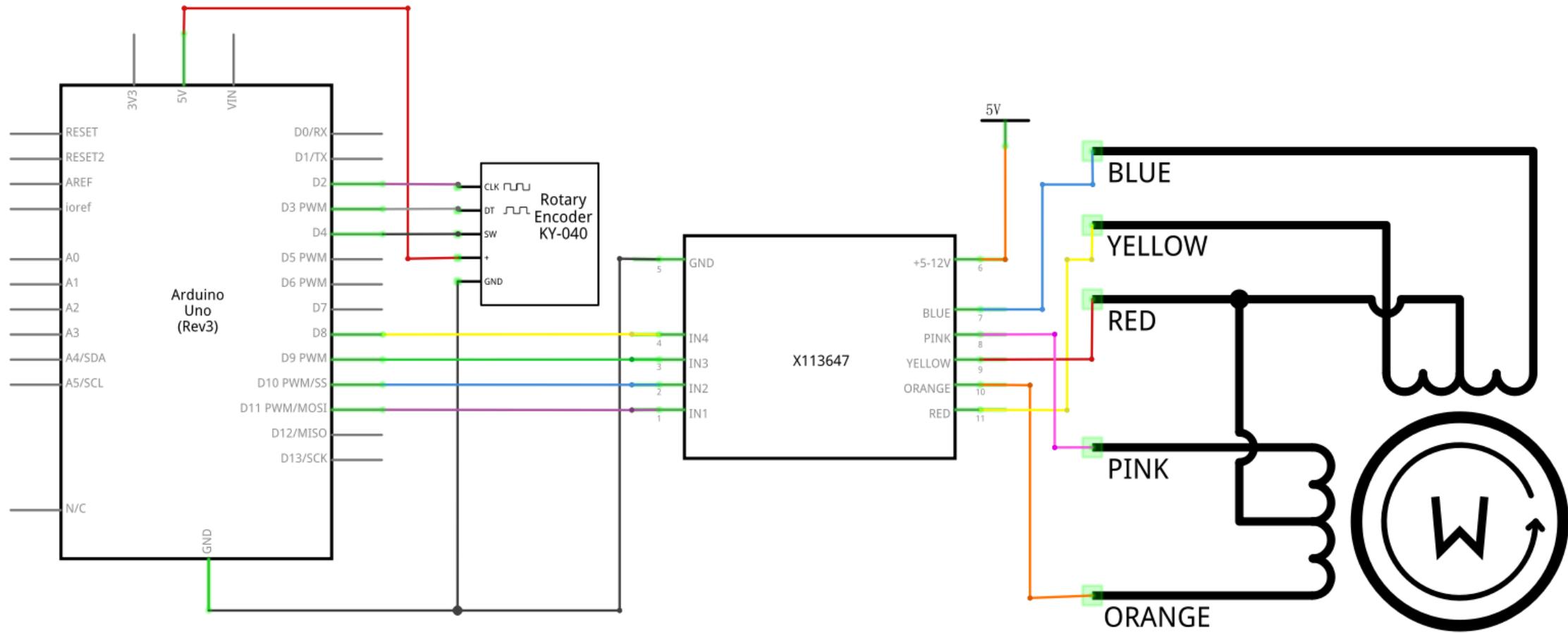
Ein Drehgeber, auch Drehregler genannt, ist ein elektromechanisches Gerät, das die Winkelposition einer Achse in ein analoges oder digitales Signal umwandelt.

Es gibt zwei Haupttypen von Drehgebern: absolute und inkrementale (relative) Drehgeber. Absolute Drehgeber geben den aktuellen Winkel der Achse aus, während inkrementale Drehgeber Informationen über die Bewegung der Achse ausgeben, wodurch anschließend die Geschwindigkeit, Distanz und Position berechnet werden können.

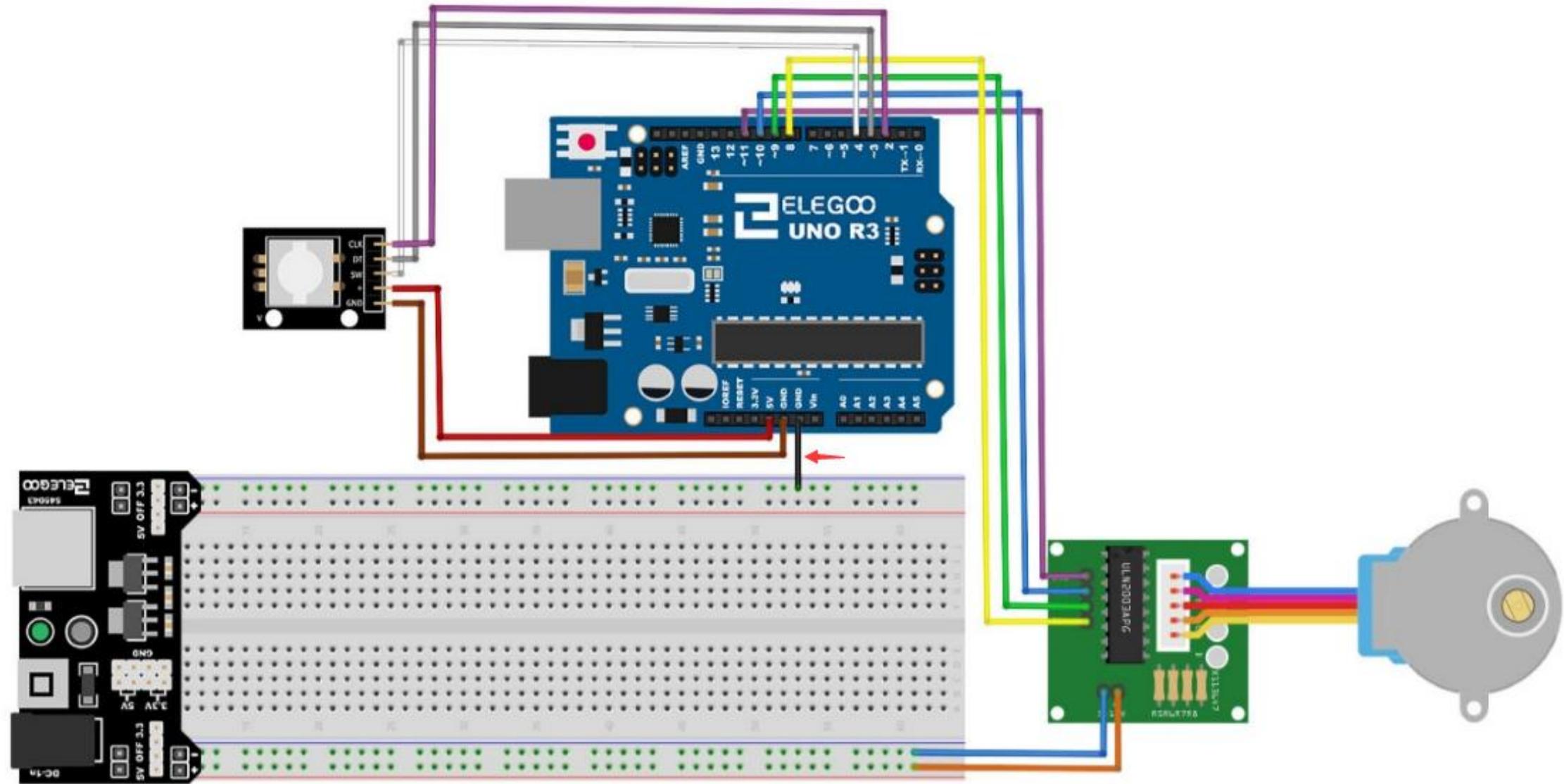
Drehgeber werden überall dort verwendet, wo eine präzise Steuerung mit endloser Drehung benötigt wird, unter anderem in industriellen Steuerungen, Robotern, speziellen Objektiven, Computereingabegeräten wie Mäusen und Trackballs, Rheometern und Drehradars.



Verbindungsschema



Schaltplan



Wir benötigen 4 Pins für die Steuerung des Schrittmotors und 3 Pins für das Drehgebermodul.

Den Schrittmotor schließen wir an die *Pins 8-11* und den Drehgeber an die *Pins 2-4* des UNO Boards an.

Wir verbinden zur Stromversorgung den *5V-* und den *GND* Anschluss des Drehgebers mit den entsprechenden Pins auf unserem Board. Um das Board nicht zu beschädigen, versorgen wir den Schrittmotor über die externe Breadboard Stromversorgung mit Strom. Wir verbinden außerdem den *GND-Anschluss* des UNO Boards mit dem *GND-Anschluss* des Breadboards zum Potentialausgleich der beiden Stromversorgungen.

Code

Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „*Lesson 33 Controlling Stepper Motor With Rotary Encoder*“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 2 noch einmal an.

Bevor Sie diesen Sketch hochladen können, müssen Sie die „*Stepper*“-Bibliothek installiert haben. Sonst wird sich Ihr Sketch nicht hochladen lassen. Für Hinweise wie man eine Bibliothek einbindet, gehen Sie zurück zu Lektion 1.

Wir benutzen einige Variablen, um die aktuelle Position des Schrittmotors zu speichern, damit wir die Schrittzahl im Auge behalten und am Ende zur Startposition zurückkehren können.

Außerdem ist eine Fehlerüberprüfung eingebaut, um sicher zu gehen, dass der Drehgeber keine Schritte überspringt, da dies die Schrittmotorposition ungenau machen würde.

