37 SENSOR KIT ANLEITUNG

FÜR UNO V2.0

Vorwort

Unser Unternehmen

Die 2011 gegründete ELEGOOO Inc. ist ein florierendes Technologieunternehmen, das sich der Forschung & Entwicklung, Produktion und Vermarktung von Open-Source-Hardware widmet. Mit Sitz in Shenzhen, dem Silicon Valley in China, sind wir auf über 150 Mitarbeiter angewachsen und verfügen über eine ca. 1000 Quadratmeter große Fabrik.

Unsere Produktlinien umfassen DuPont-Drähte, UNO R3-Boards und komplette Starterkits, die für Kunden jeden Niveaus entwickelt wurden um Arduino-Kenntnisse zu erlernen. Darüber hinaus verkaufen wir auch Raspberry Pi Zubehör wie TFT-Touchscreens. Weiterin planen wir, unser Angebot um weitere Technologien zu erweitern, darunter auch Produkte rund um den 3D-Druck. Alle unsere Produkte entsprechen internationalen Qualitätsstandards und werden von unseren Kunden auf den verschiedensten Märkten weltweit gelobt.

Offizielle Webseite:

http://www.elegoo.com

US Amazon storefront: http://www.amazon.com/shops/A2WWHQ25ENKVJ1

CA Amazon storefront: http://www.amazon.ca/shops/A2WWHQ25ENKVJ1

UK Amazon storefront: http://www.amazon.co.uk/shops/AZF7WYXU5ZANW

DE Amazon storefront: http://www.amazon.de/shops/AZF7WYXU5ZANW

FR Amazon storefront: http://www.amazon.fr/shops/AZF7WYXU5ZANW

ES Amazon storefront: http://www.amazon.es/shops/AZF7WYXU5ZANW

IT Amazon storefront: http://www.amazon.it/shops/AZF7WYXU5ZANW

Unsere Anleitung

Diese Anleitung ist für Anfänger gedacht. Sie werden alle Grundlagen über das Arduino Board, Sensoren und andere Komponenten lernen. Wenn Sie sich tiefergehend mit der Arduino Plattform beschäftigen wollen, empfehlen wir Ihnen das „Arduino Kochbuch“ von Michael Margolis.

Einiger Code in dieser Anleitung wurde von Simon Monk geschrieben. Simon Monk ist der Autor einer Reihe von Büchern zum Thema Open Source Hardware. Einige seiner Werke sind “30 Arduino Selbstbau-Projekte” und “Raspberry Pi programmieren” und können im Buchhandel erworben werden. Weitere Titel sind auf Englisch erhältlich.

Kundenservice

Als ein kontinuierlich und schnell wachsendes Technologieunternehmen sind wir bestrebt, Ihnen exzellente Produkte und einen qualitativ hochwertigen Service zu bieten. Sie erreichen uns per E-Mail unter service@elegoo.com oder EUservice@elegoo.com. Wir würden uns freuen, von Ihnen zu hören und jegliche Kommentare und Anregungen sind für uns sehr wertvoll.

Alle Probleme oder Fragen, die Sie mit unseren Produkten haben, werden von unseren erfahrenen Ingenieuren innerhalb von 12 Stunden (24 Stunden an Feiertagen) umgehend beantwortet.

Packing list

www.elegoo.com

JOYSTICK

RGB
LED

IR
RECEIVER

18B20
TEMP

AVOIDANCE

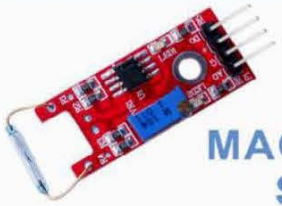
FLAME

TRACKING

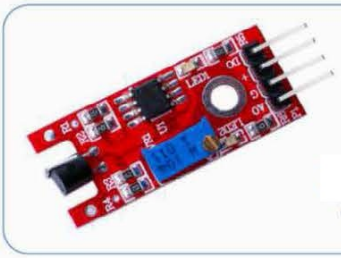
PHOTO
RESISTOR

IR
EMISSION

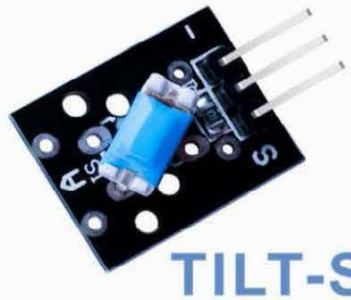
Contact us : service@elegoo.com



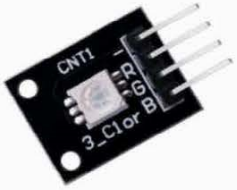
MAGNETIC SPRING



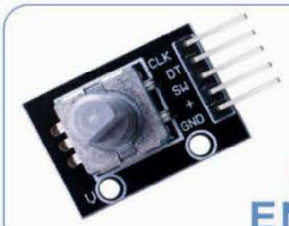
METAL TOUCH



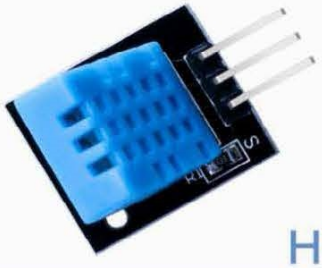
TILT-SWITCH



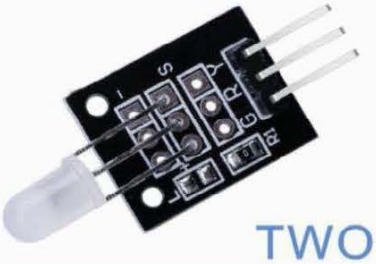
SMD RGB



ROTARY ENCODER



TEMP AND HUMIDITY



TWO-COLOR

Contact us : service@elegoo.com


ULTRASONIC SENSOR


DS1307 RTC Module


GY-521 MODULE


PHOTO-INTERRUPTER

TAP MODULE


7 COLOR FLASH


BUTTON


ACTIVE BUZZER


LASER EMIT


MEMBRANE SWITCH


BIG SOUND


WATER LEVEL SENSOR


DIGITAL TEMPERATURE

LCD 1602
MODULE
(WITH PIN HEADER)

RELAY

LINEAR
HALL

POWER
SUPPLY
MODULE

SMALL
SOUND

SHOCK

PASSIVE
BUZZER

HC-SR501 PIR
MOTION SENSOR

Contact us : service@elegoo.com

Inhaltsverzeichnis

Lektion 0 Installieren der Arduino IDE (Entwicklungsumgebung)

Einführung

Die integrierte Entwicklungsumgebung (IDE) von Arduino ist die Softwareseite der Arduino Plattform.

In dieser Lektion lernen Sie, wie Sie Ihren Computer für die Verwendung von Arduino einrichten.

Die Arduino-Software, mit der Sie Ihren Arduino programmieren, ist für Windows, Mac und Linux erhältlich. Der Installationsprozess ist für alle drei Plattformen minimal unterschiedlich und es gibt eventuell einige manuelle Schritte bei der Installation der Software.

Schritt 1: Gehen Sie zu https://www.arduino.cc/en/Main/Software und scrollen Sie bis zu folgender Stelle:

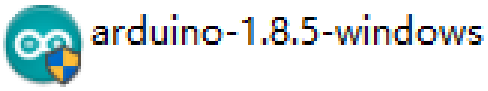
Die auf dieser Website verfügbare Version ist in der Regel die neueste Version, und die aktuelle Version ist möglicherweise neuer als die Version auf dem Bild in dieser Anleitung.

Schritt 2： Laden Sie die Entwicklungssoftware herunter, die mit dem Betriebssystem Ihres Computers kompatibel ist. <u>In diesem Beispiel nehmen wir Windows.</u>

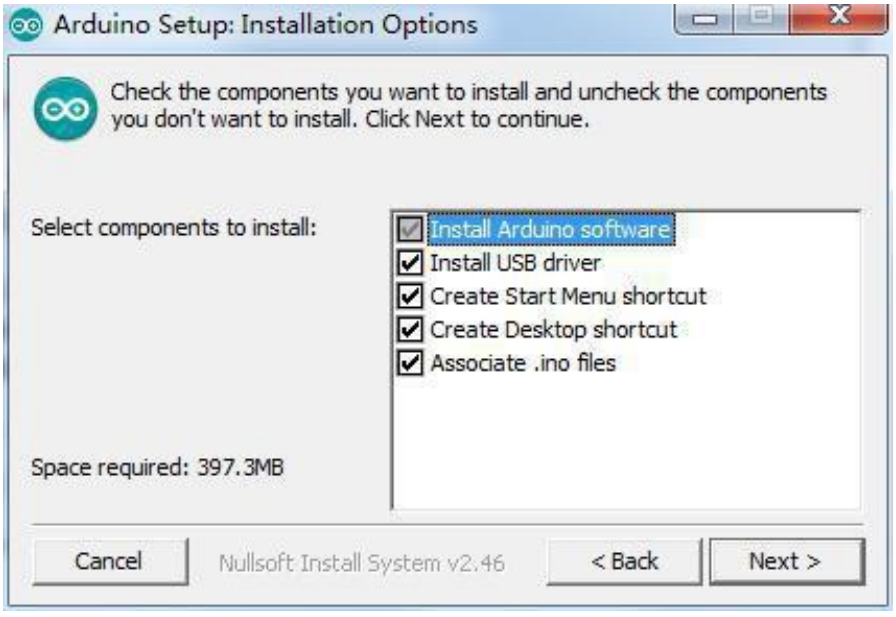
Klicken sie auf „Windows installer“

Klicken Sie auf „JUST DOWNLOAD“

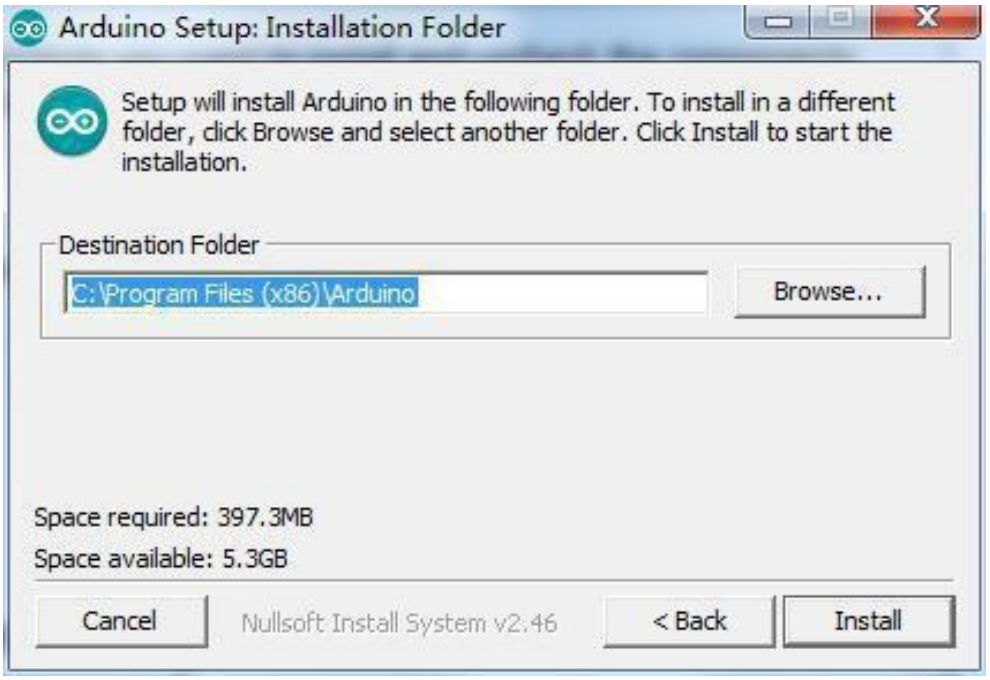
Nachdem Sie die Arduino IDE heruntergeladen haben, starten Sie das Installationsprogram (zur Zeit der Erstellung dieser Anleitung „arduino-1.8.5-windows.exe“).



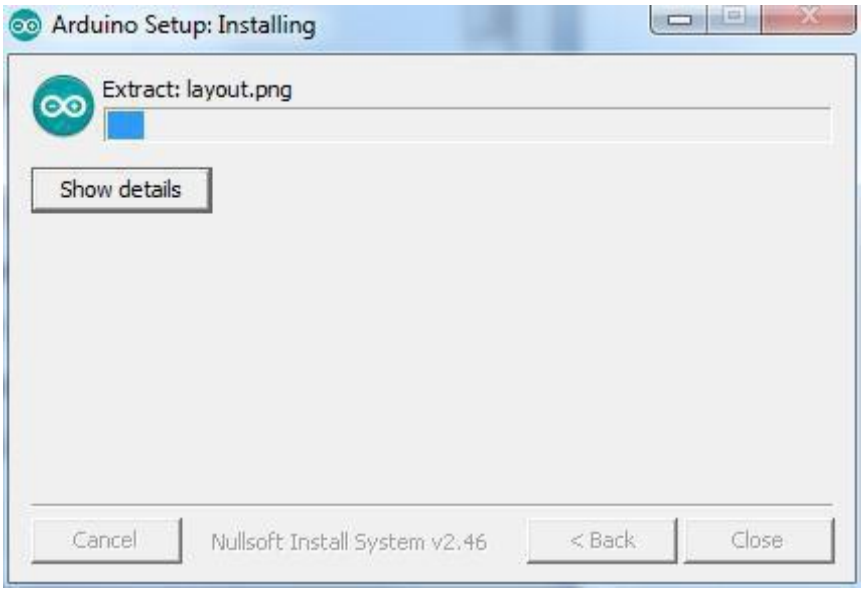
Klicken Sie auf „I agree“



Klicken Sie auf „Next“



Sie können entweder den Standardinstallationspfad benutzen (für Anfänger empfohlen) oder per Klick auf den „Browse-Button“ einen eigenen Installationspfad auswählen. Klicken Sie auf „Install“ um den Installationsprozess zu starten.



Anschließend erscheint der folgende Dialog zur Treiberinstallation. Klicken Sie auf „Install“, um die Installation abzuschließen.

Nach dem Ende der erfolgreichen Installation erscheint auf dem Desktop das Arduino Icon (siehe unten). Mittels dieses Icons können Sie die Arduino IDE starten.

Doppelklicken Sie nun das Icon um die Arduino Entwicklungsumgebung zu starten.

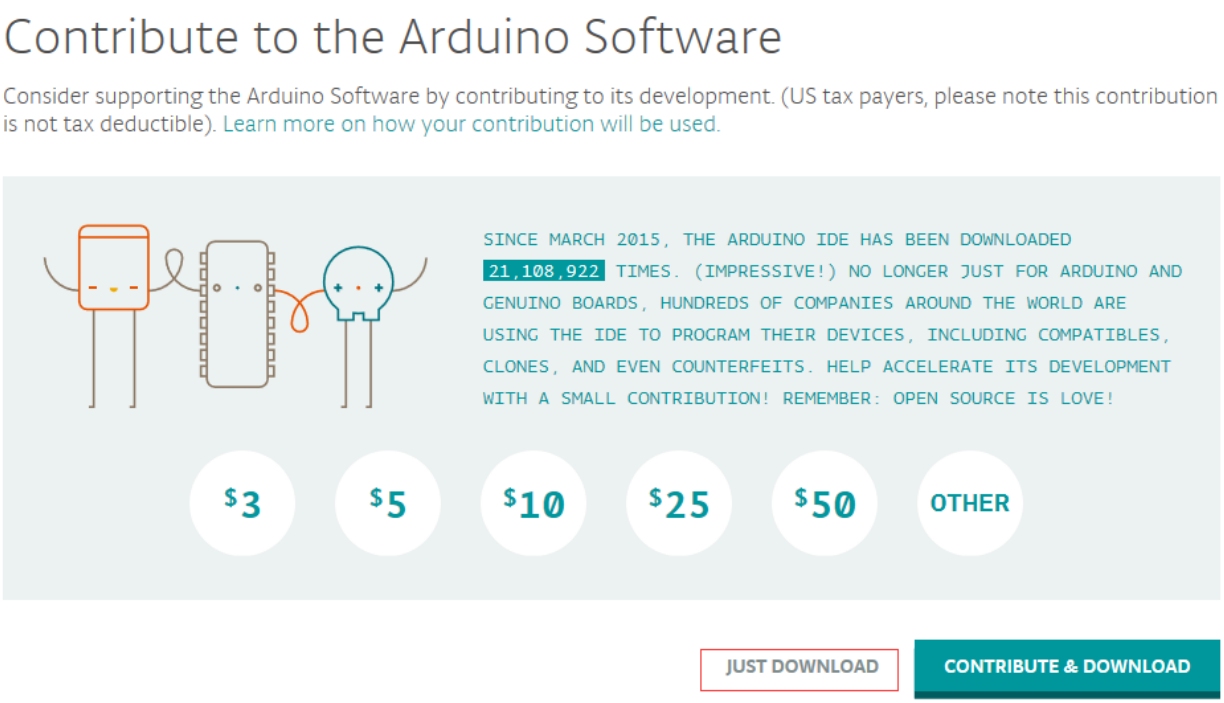
Windows-Benutzer können die Installationsanweisungen für Mac-Systeme überspringen und zu Lektion 1 übergehen. Mac-Benutzer sollten den follgenden Abschnitt lesen.

Installation Arduino (Mac OS X)

Schritt 1: Download der Arduino Software (IDE)

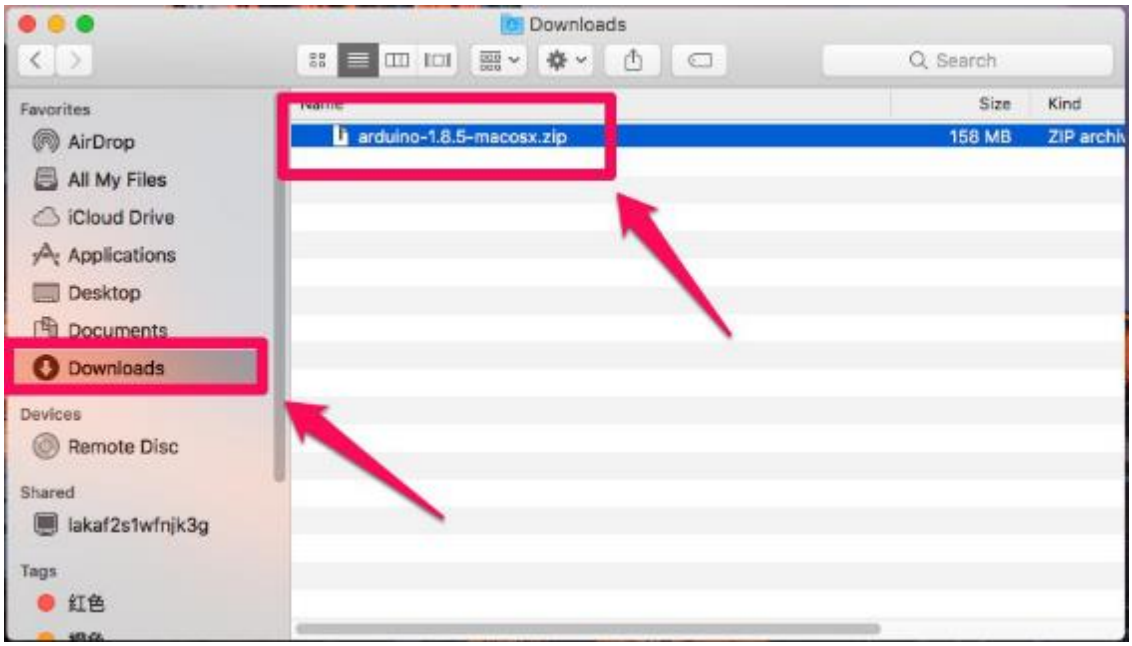
·Öffnen Sie die folgende URL: **https://www.arduino.cc/en/Main/Software**

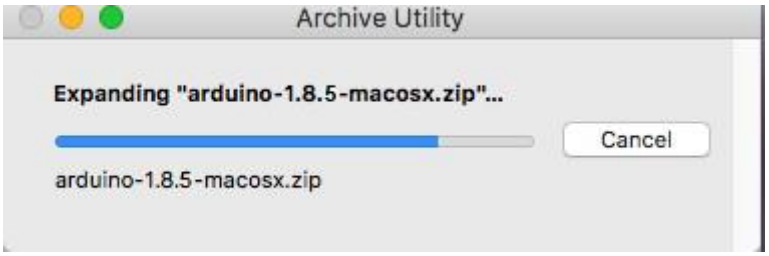
·Klicken Sie auf „Mac OSX 10.7 Lion or neuer“



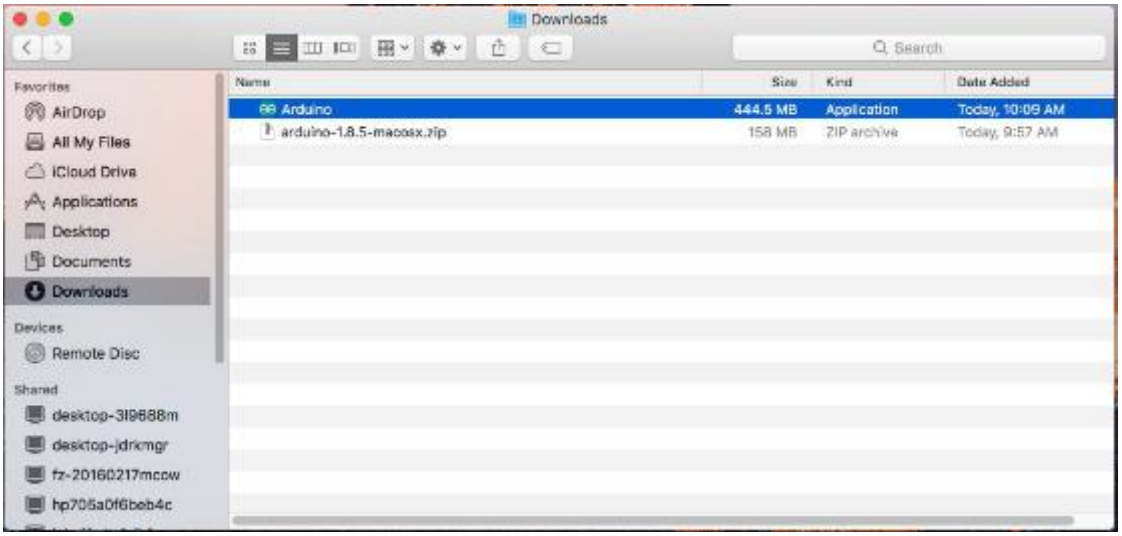
·Klicken Sie auf „JUST DOWNLOAD“

Laden Sie die Zip-Datei herunter und entpacken Sie sie; doppelklicken Sie Arduino.app, um die Arduino IDE zu öffnen. Das System wird Sie auffordern die Java-Laufzeitbibliothek zu installieren, wenn diese nicht schon auf Ihrem Computer installiert ist. Sobald die Installation abgeschlossen ist, können Sie die Arduino IDE starten.





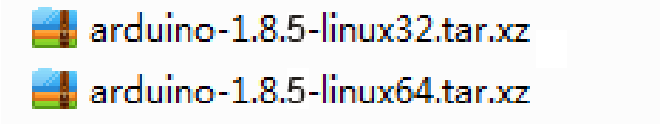
· Die Arduino Datei ist die Arduino Entwicklungsumgebung. Doppelklicken Sie die Datei um Arduino zu starten.



·Arduino Entwicklungsumgebung (IDE)

Installation Arduino (Linux)

Sie müssen den Befehl "make install" verwenden. Wenn Sie das Ubuntu-System verwenden, wird empfohlen, Arduino IDE vom Ubuntu Software Center aus zu installieren.



Lektion 1 Bibliotheken hinzufügen

Installation anderer Arduino Bibliotheken

Sobald Sie mit der Arduino-Software vertraut sind und die eingebauten Funktionen nutzen, können Sie die Möglichkeiten Ihres Arduino mit zusätzlichen Bibliotheken erweitern.

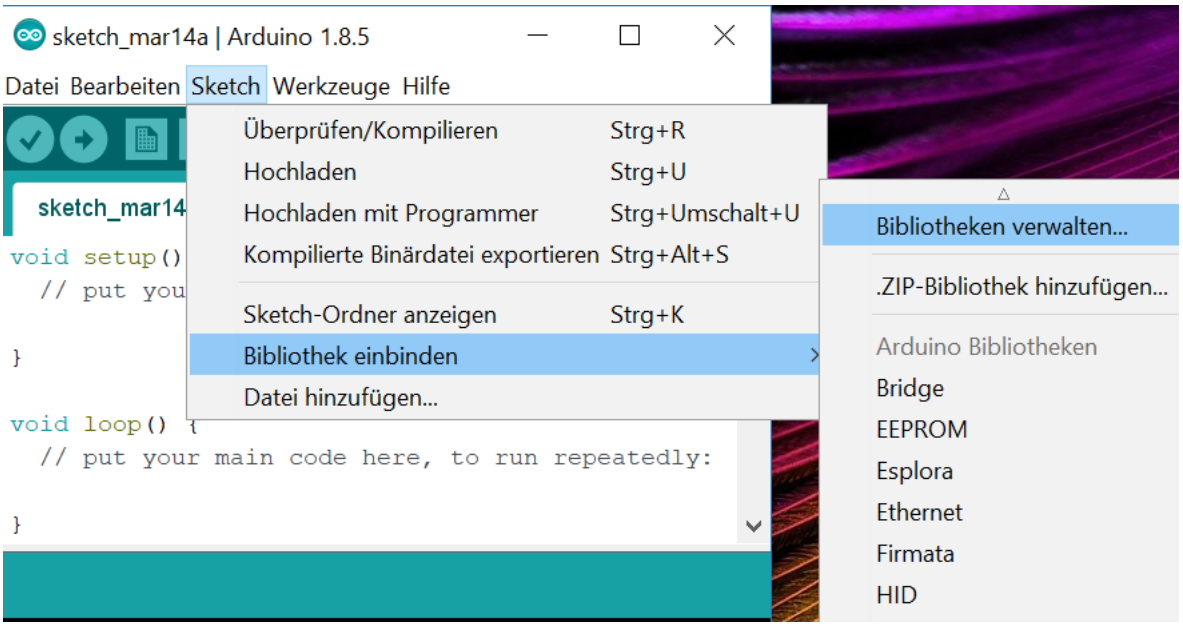
Was sind Bibliotheken?

Bibliotheken sind eine Sammlung von Code, die es Ihnen leicht macht, einen Sensor, ein Display, ein Modul oder ähnliches anzuschließen. Die eingebaute Flüssigkristallbibliothek macht es zum Beispiel einfach, mit LCD-Bildschirmen zu sprechen. Es gibt Hunderte von zusätzlichen Bibliotheken, die im Internet zum Download zur Verfügung stehen. Die eingebauten Bibliotheken und einige dieser zusätzlichen Bibliotheken sind in der Referenz aufgeführt. Um die zusätzlichen Bibliotheken nutzen zu können, müssen Sie diese installieren.

Wie man eine Bibliothek installiert?

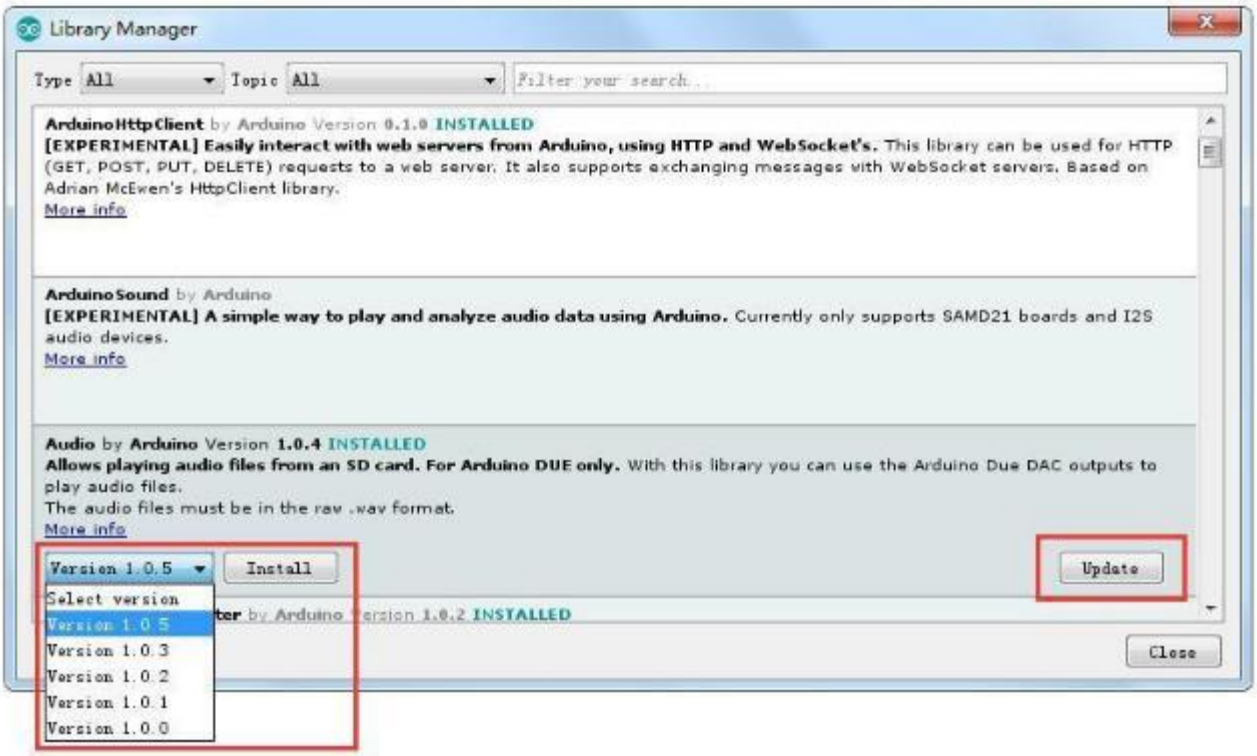
Mittels des Library Managers

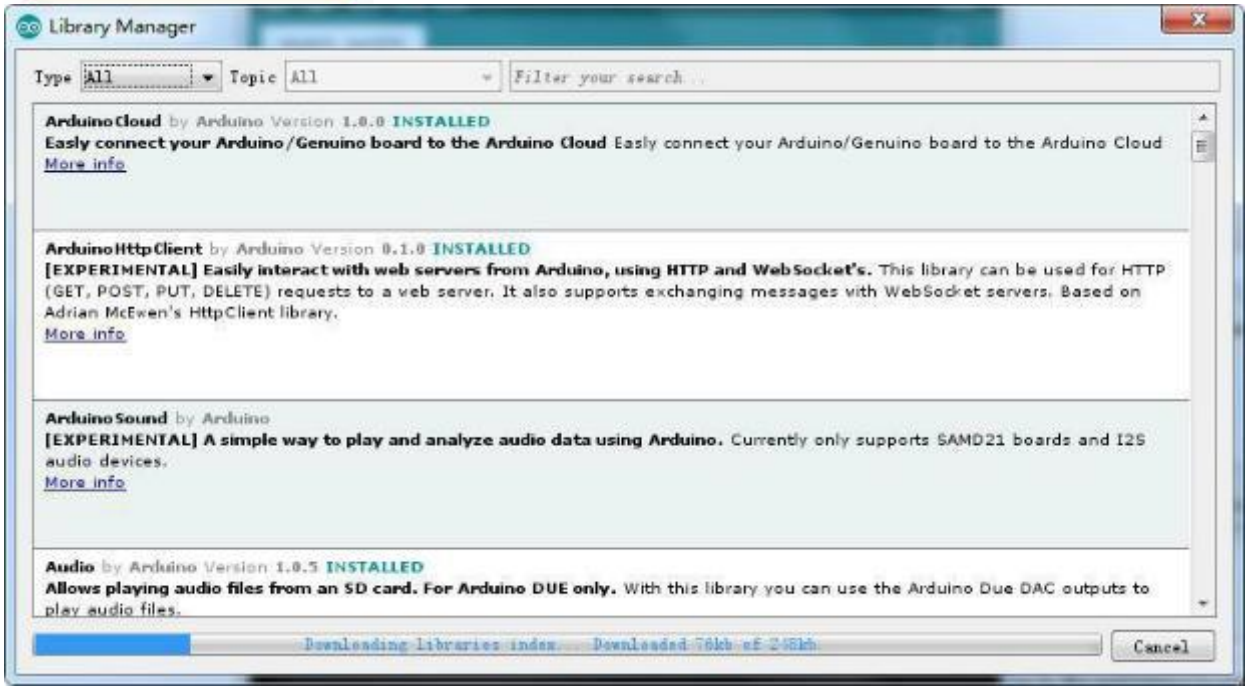
Um eine neue Bibliothek in Ihre Arduino IDE zu installieren, können Sie den Library Manager verwenden (verfügbar ab IDE Version 1.8.5). Öffnen Sie die IDE und klicken Sie auf das Menü "Sketch" und dann auf Bibliothek einbinden > Bibliotheken verwalten.



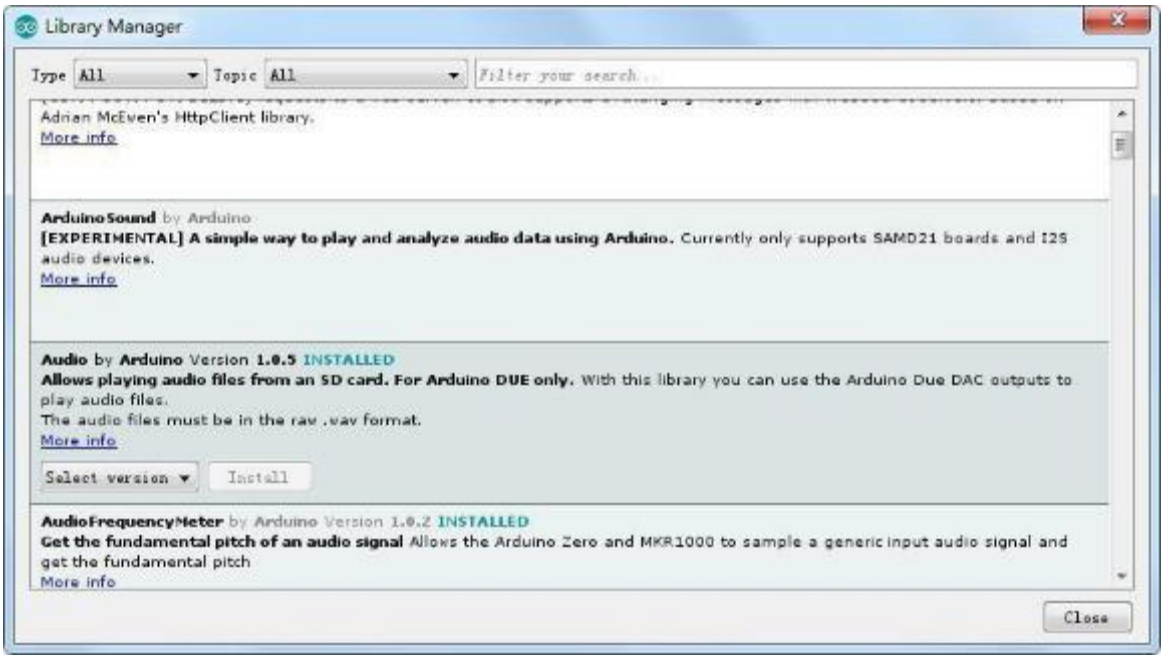
Dann öffnet sich der Library Manager und Sie finden eine Liste der Bibliotheken, die bereits installiert sind oder installiert werden können. In diesem Beispiel werden wir die Bridge-Bibliothek installieren. Scrollen Sie in der Liste, um sie zu finden und wählen Sie dann die Version der Bibliothek aus, die Sie installieren möchten. Manchmal ist nur eine Version der Bibliothek verfügbar. Wenn das Versionsauswahlmenü nicht erscheint, machen Sie sich keine Sorgen: Dann gibt es eben nur genau eine Version der Bibliothek.

Mittels des „Update Buttons“ können die neusten Versionen vom Server abgerufen werden.


Library Manager
Type All
Topic All
Filter your search...
ArduinoHttpClient by Arduino Version 0.1.0 INSTALLED
[EXPERIMENTAL] Easily interact with web servers from Arduino, using HTTP and WebSocket's. This library can be used for HTTP (GET, POST, PUT, DELETE) requests to a web server. It also supports exchanging messages with WebSocket servers. Based on Adrian McEwen's HttpClient library.
More info
ArduinoSound by Arduino
[EXPERIMENTAL] A simple way to play and analyze audio data using Arduino. Currently only supports SAMD21 boards and I2S audio devices.
More info
Audio by Arduino Version 1.0.4 INSTALLED
Allows playing audio files from an SD card. For Arduino DUE only. With this library you can use the Arduino Due DAC outputs to play audio files.
The audio files must be in the raw .wav format.
More info
Version 1.0.5
Install
Update
Select version
Version 1.0.5
Version 1.0.3
Version 1.0.2
Version 1.0.1
Version 1.0.0
Close

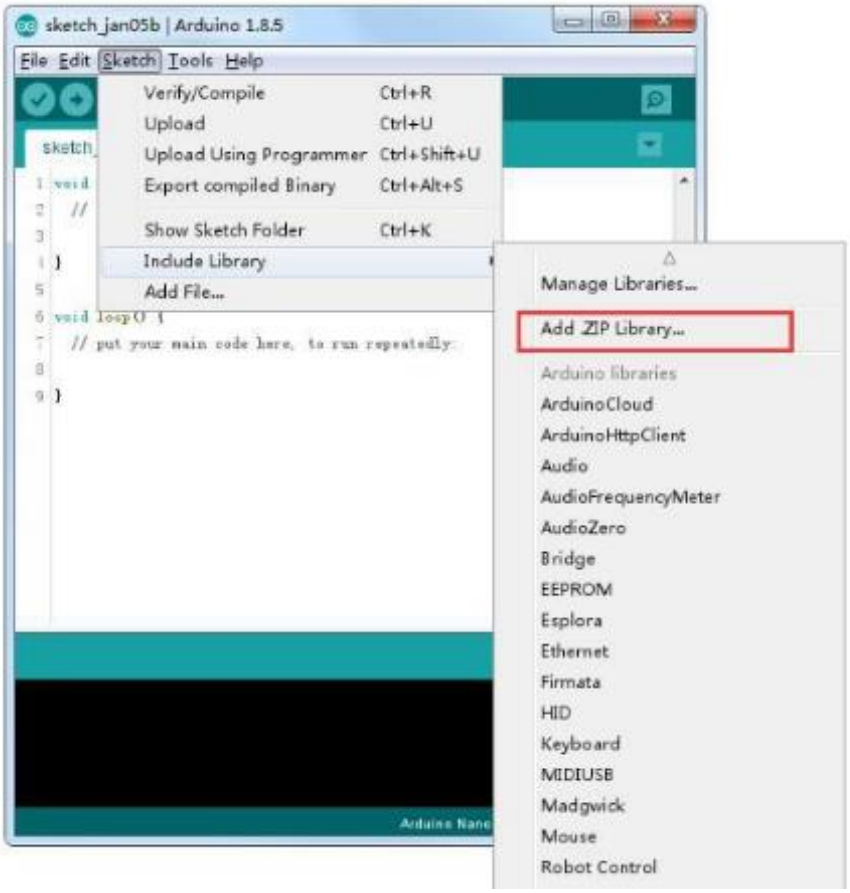

Library Manager
Type All
Topic All
Filter your search...
ArduinoCloud by Arduino Version 1.0.0 INSTALLED
Easily connect your Arduino/Genuino board to the Arduino Cloud Easily connect your Arduino/Genuino board to the Arduino Cloud
More info
ArduinoHttpClient by Arduino Version 0.1.0 INSTALLED
[EXPERIMENTAL] Easily interact with web servers from Arduino, using HTTP and WebSocket's. This library can be used for HTTP (GET, POST, PUT, DELETE) requests to a web server. It also supports exchanging messages with WebSocket servers. Based on Adrian McEwen's HttpClient library.
More info
ArduinoSound by Arduino
[EXPERIMENTAL] A simple way to play and analyze audio data using Arduino. Currently only supports SAMD21 boards and I2S audio devices.
More info
Audio by Arduino Version 1.0.5 INSTALLED
Allows playing audio files from an SD card. For Arduino DUE only. With this library you can use the Arduino Due DAC outputs to play audio files.
Cancel

Klicken Sie abschließend auf "Installieren“ und warten Sie, bis die IDE die neue Bibliothek installiert hat. Das Herunterladen kann einige Zeit in Anspruch nehmen und hängt von Ihrer Verbindungsgeschwindigkeit ab. Wenn es fertig ist, sollte das Wort „Installed“ neben der Bridge-Bibliothek erscheinen (Die Arduino IDE ist nur teilweise ins Deutsche übersetzt). Sie können den Library Manager nun schließen. Anmerkung: In manchen Installationen ist die Bridge Bibliothek schon installiert; dieser Abschnitt dient nur dazu das Prinzip aufzuzeigen)

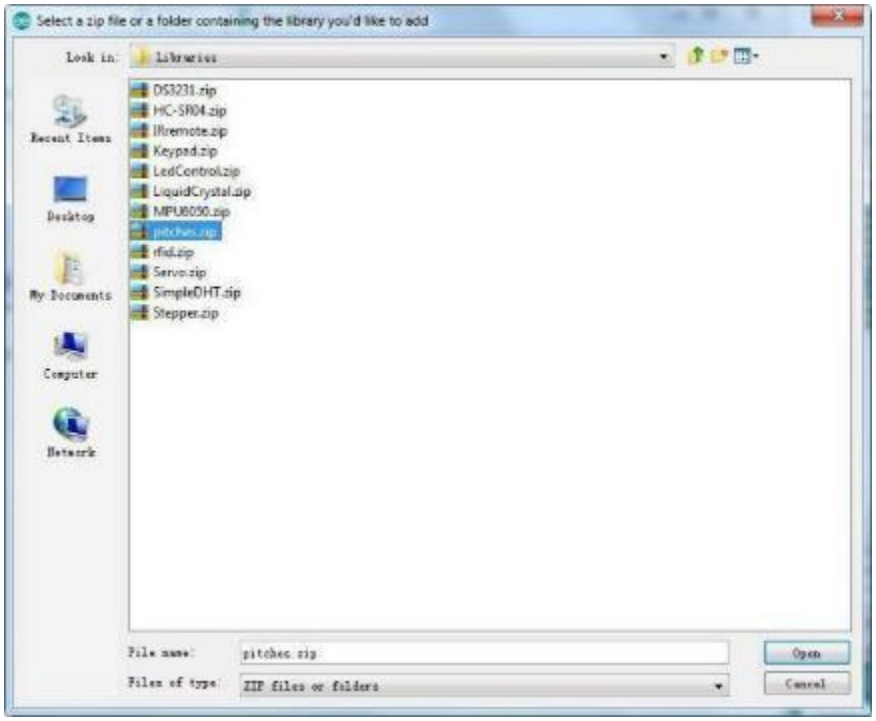


Importieren einer .zip Bibliothek

Bibliotheken werden oft als ZIP-Datei oder Ordner verteilt. Der Name des Ordners ist der Name der Bibliothek. Innerhalb des Ordners befindet sich eine.cpp-Datei, eine.h-Datei, oft eine keywords.txt-Datei, ein Beispielordner und andere Dateien, die von der Bibliothek benötigt werden. Ab Version 1.0.5 können Sie Bibliotheken von Drittanbietern in der IDE installieren. Entpacken Sie die heruntergeladene Bibliothek nicht, sondern belassen Sie sie so, wie sie ist.
Navigieren Sie in der Arduino-IDE zu Sketch > Bibliothek einbinden. Wählen Sie oben in der Dropdown-Liste die Option ".ZIP Bibliothek hinzufügen" (englisch: Add .ZIP library).



Sie werden aufgefordert, die Bibliotheken auszuwählen, die Sie hinzufügen möchten. Navigieren Sie zum Speicherort der.zip-Datei und öffnen Sie diese.



Kehren Sie zum Menü Sketch > Bibliothek einbinden zurück. Sie sollten nun die Bibliothek am unteren Rand des Dropdown-Menüs sehen. Die Bibliothek ist nun bereit für die Verwendung in Ihrem Sketch. Die Zip-Datei wird im Bibliotheksordner in Ihrem Arduino Sketchverzeichnis installiert. Bemerkung: die Bibliothek kann in Sketches verwendet werden, aber die Beispiele der Bibliothek werden erst nach einem Neustart der IDE angezeigt unter Datei > Beispiele.

Auf MAC und Linux Systemen funktioniert die Installation genauso. Die manuelle Installation, die im Folgenden als Alternative vorgestellt wird, wird selten verwendet und Anfänger können das Kapitel getrost überspringen.

Manuelle Installation

Um die Bibliothek zu installieren, beenden Sie zuerst die Arduino-Anwendung. Dann entpacken Sie die ZIP-Datei mit der Bibliothek. Wenn Sie zum Beispiel eine Bibliothek mit dem Namen "ArduinoParty" installieren, dekomprimieren Sie ArduinoParty.zip. Die Datei sollte einen Ordner namens ArduinoParty enthalten, in dem sich Dateien wie ArduinoParty.cpp und ArduinoParty.h befinden. (Wenn sich die.cpp- und.h-Dateien nicht in einem Ordner befinden, müssen Sie einen Ordner erstellen. In diesem

Fall erstellen Sie einen Ordner mit dem Namen "ArduinoParty" und verschieben dorthin alle Dateien, die in der ZIP-Datei enthalten sind, wie z.B. ArduinoParty.cpp und ArduinoParty.h.).

Ziehen Sie den ArduinoParty-Ordner in diesen Ordner (Ihren Bibliotheken-Ordner). Unter Windows wird er wahrscheinlich "Dokumente\Arduino\Bibliotheken" heißen. Für Mac-Benutzer wird er wahrscheinlich "Dokumente/Arduino/libraries" heißen. Unter Linux ist es der Ordner "libraries" in Ihrem Sketchbuch.

Ihr Arduino-Bibliotheksordner sollte nun so aussehen (Windows):

Dokumente\Arduino\libraries\ArduinoParty\ArduinoParty.cpp

Dokumente\Arduino\libraries\ArduinoParty\ArduinoParty.h

Dokumente\Arduino\libraries\ArduinoParty\examples

Oder so (Mac und Linux):

Dokumente/Arduino/libraries/ArduinoParty/ArduinoParty.cpp

Dokumente /Arduino/libraries/ArduinoParty/ArduinoParty.h

Dokumente /Arduino/libraries/ArduinoParty/examples

....

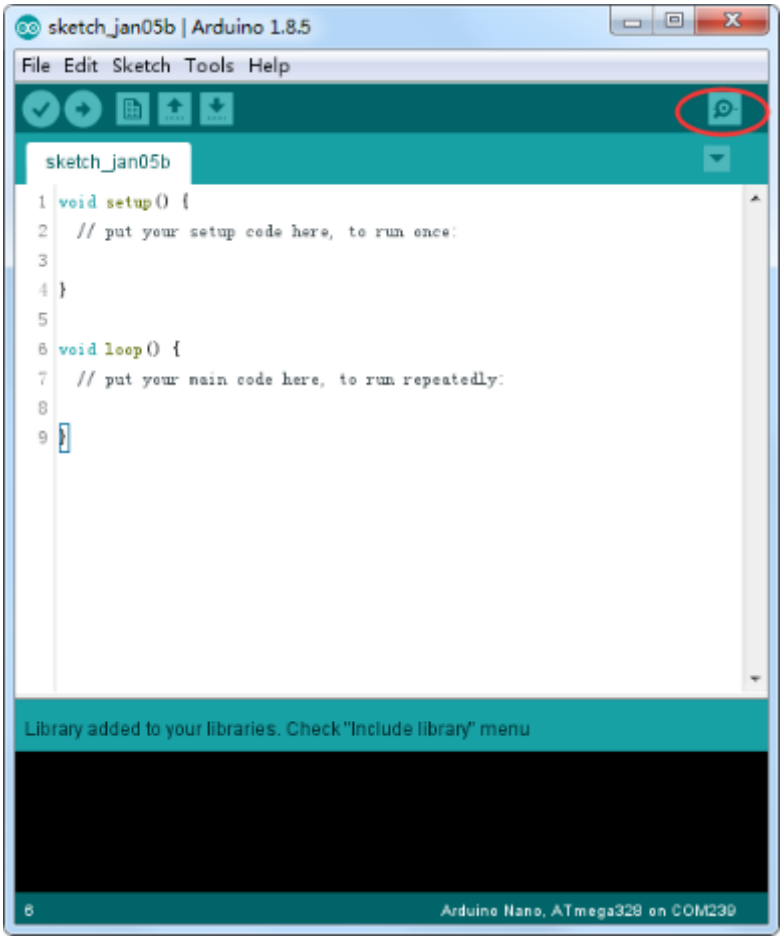
Es kann mehr Dateien geben als nur die .cpp und .h Dateien. Stellen Sie sicher, dass sie alle da sind. (Die Bibliothek funktioniert nicht, wenn Sie die .cpp- und .h-Dateien direkt in den Bibliotheksordner legen oder wenn sie in einem extra Ordner verschachtelt sind. Zum Beispiel: Dokumente\Arduino\ArduinoParty.cpp und Dokumente\Arduino\Arduino\ArduinoParty\ArduinoParty\ArduinoParty\ArduinoParty.cpp wird nicht funktionieren.) Starten Sie die Arduino Applikation neu. Stellen Sie sicher, dass die neue Bibliothek im Menüeintrag Sketch > Bibliothek einbinden angezeigt wird. Geschafft! Sie haben eine Bibliothek installiert!

Lektion 2 Öffnen des seriellen Monitors (Serial Monitor)

Die integrierte Entwicklungsumgebung (IDE) von Arduino ist die Softwareseite der Arduino-Plattform. Und weil die Verwendung eines Terminals (zur seriellen Kommunikation) einen so großen Teil der Arbeit mit Arduinos und anderen Mikrocontrollern ausmacht, haben die Entwickler beschlossen, ein serielles Terminal in die Software zu integrieren. In der Arduino Umgebung wird dieses als serieller Monitor bezeichnet. Die serielle Kommunikation zwischen Arduino Hardware und der Arduino IDE findet normalerweise per USB Kabel statt.

Verbindung aufbauen

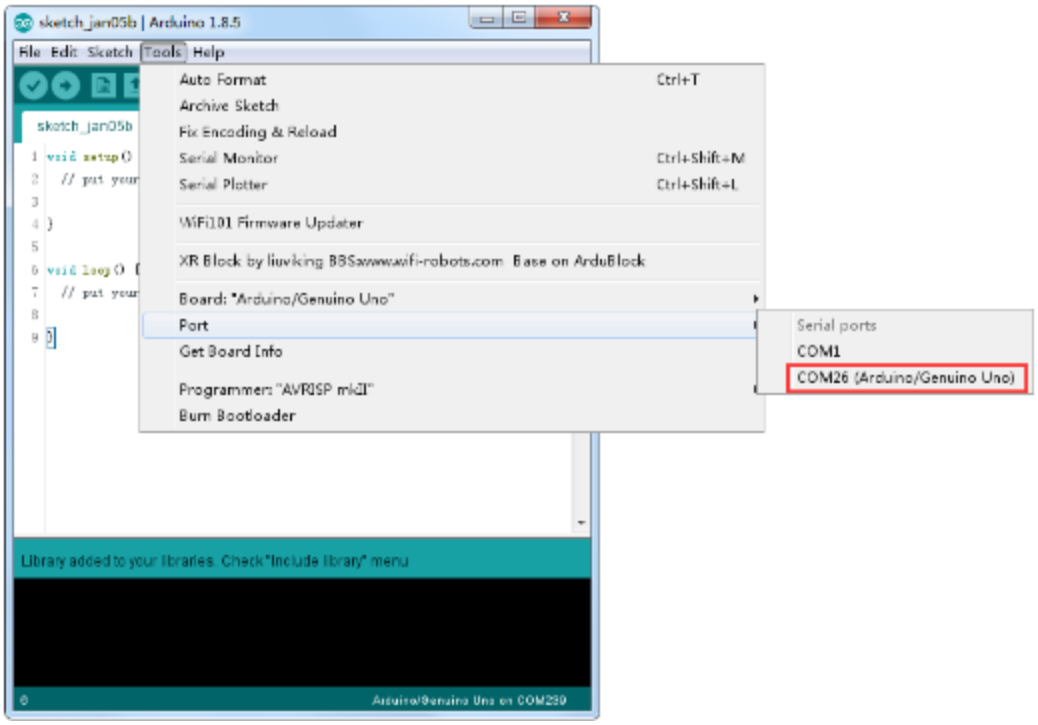
Der serielle Monitor wird mit allen Versionen der Arduino IDE mitgeliefert. Um ihn zu öffnen, klicken Sie einfach auf das Symbol Serieller Monitor (siehe Screenshot).



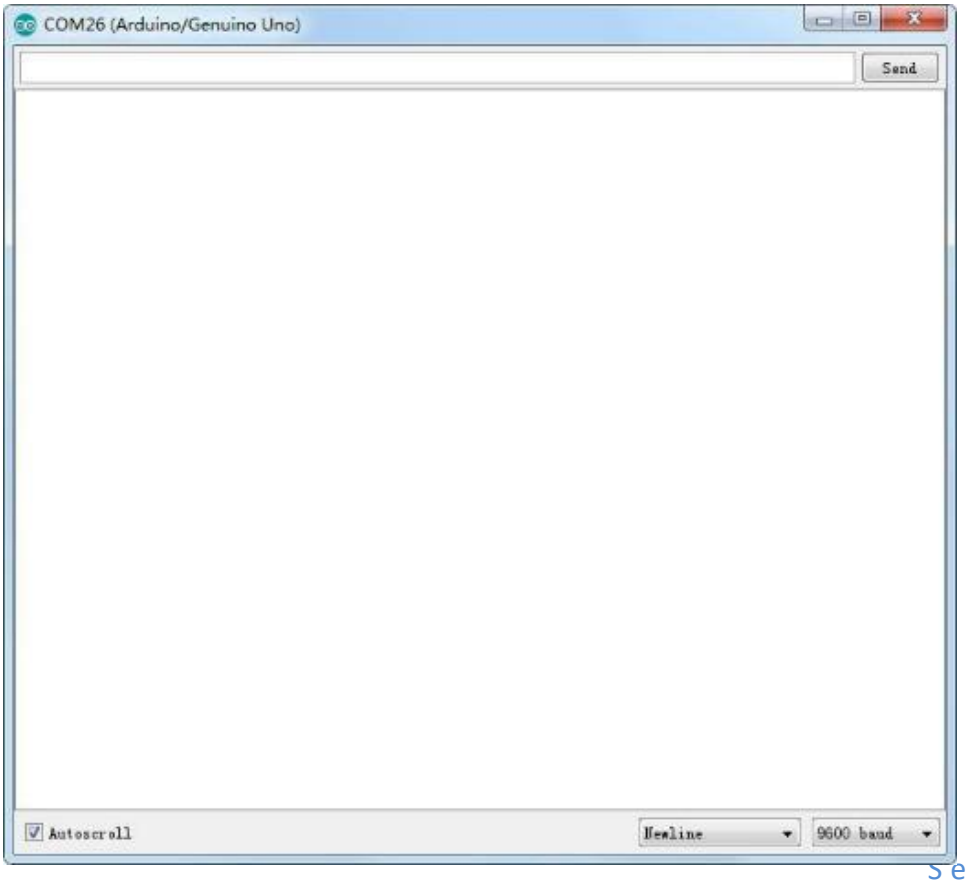
Der Port für den seriellen Monitor wird genauso ausgewählt, wie der Port zum Laden von Code auf die Arduino Hardware. (Werkzeuge -> Port, und wählen Sie den Port aus).

Hinweise:

- An manchen PCs wird der Port erst angezeigt im Menü, wenn die Arduino Hardware angeschlossen ist
- Wie im Screenshot unten zu sehen ist, steht normalerweise der Name des Arduino Boards neben dem Com Port. So sollte der passende Port leicht zu finden sein.

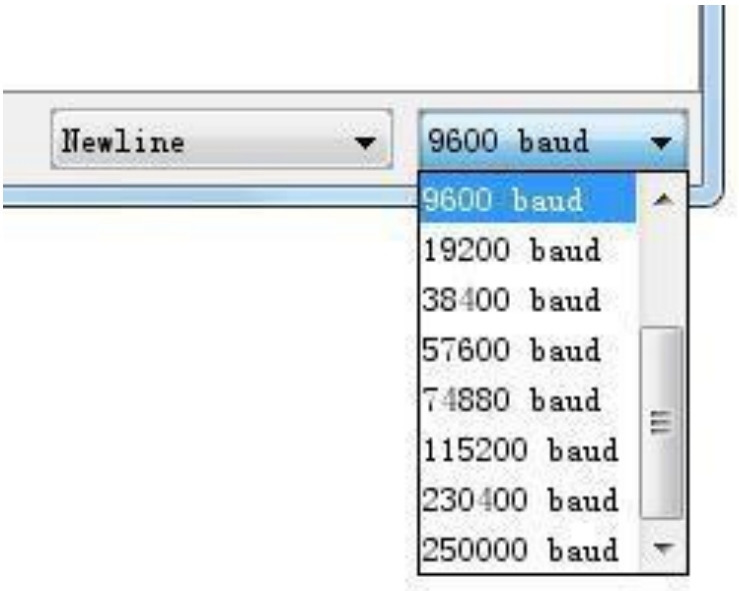


Geöffnet sieht der Monitor ungefähr folgendermaßen aus:



Einstellungen

Der serielle Monitor hat nur wenige Einstellmöglichkeiten, aber genug, um die meisten Ihrer Anforderungen an die serielle Kommunikation zu erfüllen. Die erste Einstellung die Sie ändern können ist die Baudrate. Klicken Sie auf das Dropdown-Menü Baudrate, um die gewünschte Baudrate auszuwählen. Die Baudrate im seriellen Monitor muss zur Baudrate im Arduino Code passen.



Weiterhin können Sie das Terminal auf Auto scrollen einstellen, indem Sie das Kontrollkästchen in der linken unteren Ecke aktivieren.

Vorteile

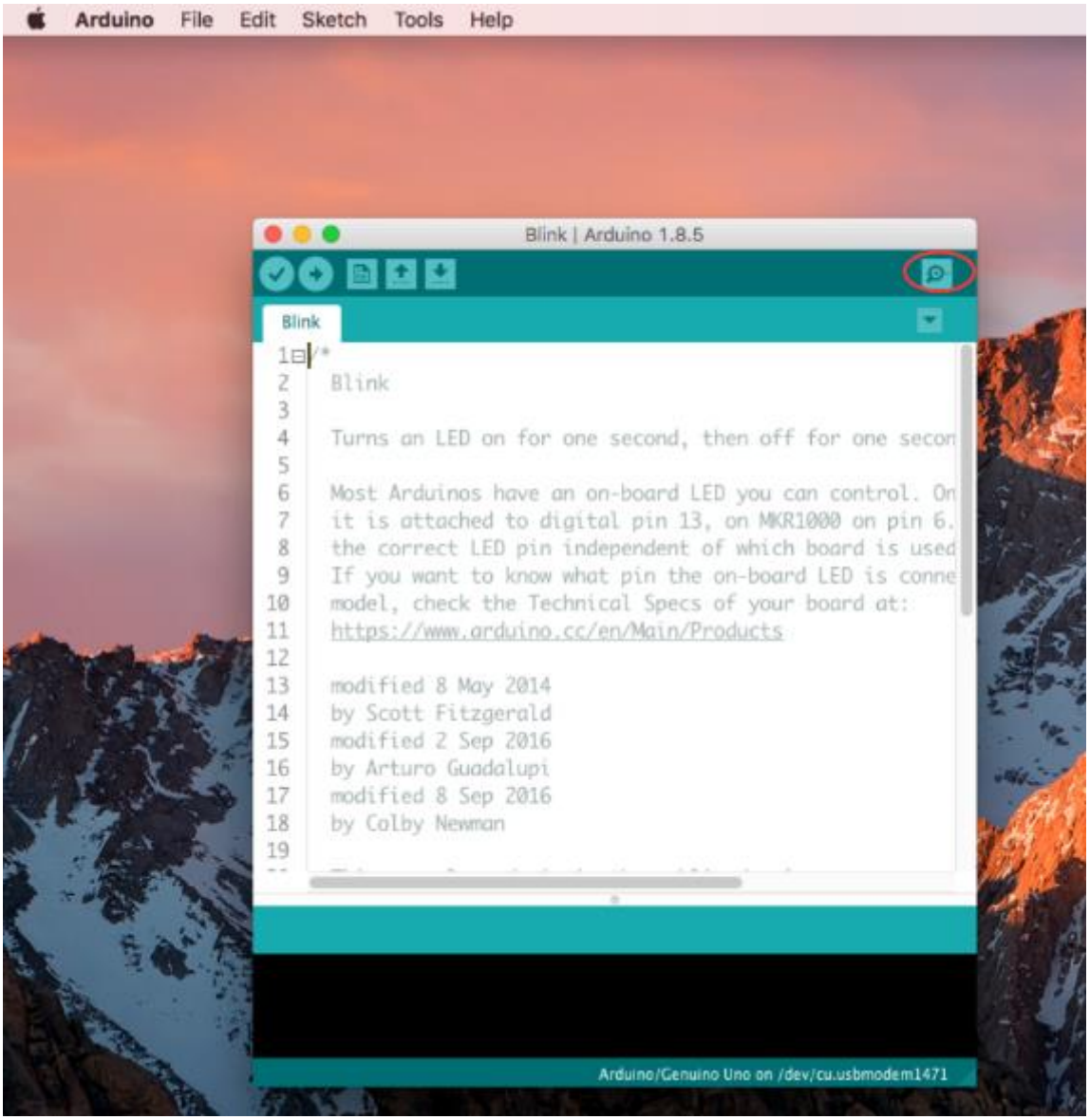
Mit dem seriellen Monitor können Sie schnell und einfach eine serielle Verbindung mit Ihrem Arduino herstellen. Wenn Sie bereits in der Arduino IDE arbeiten, ist es normalerweise nicht nötig ein separates Terminal für die Anzeige von Daten zu öffnen.

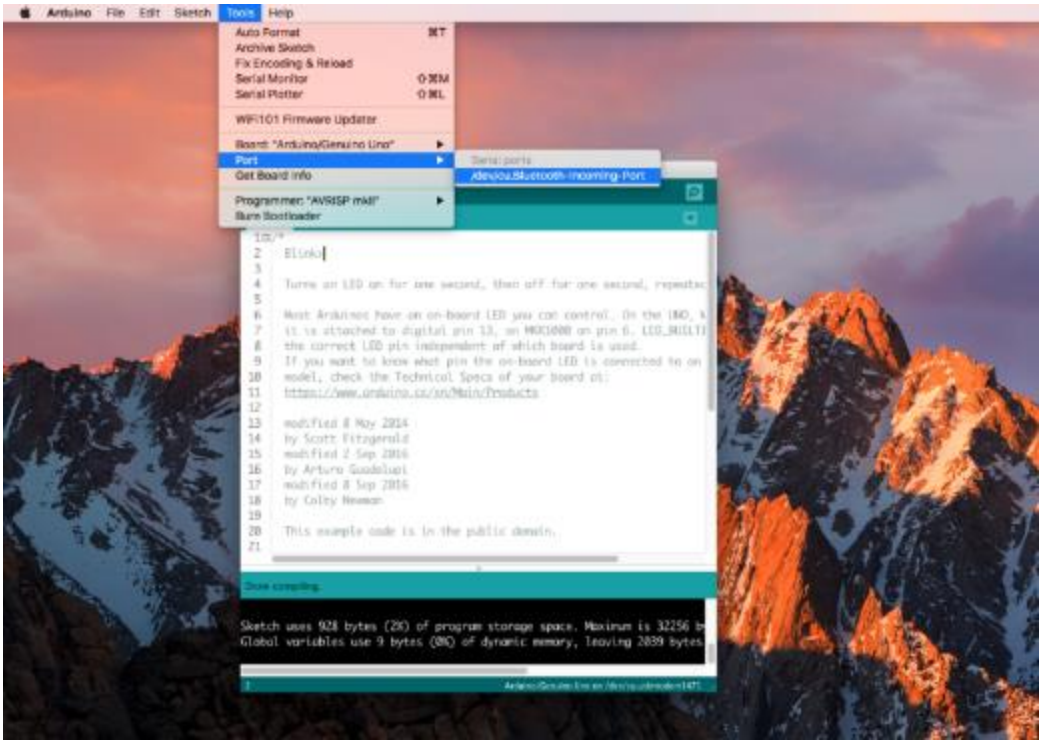
Nachteile

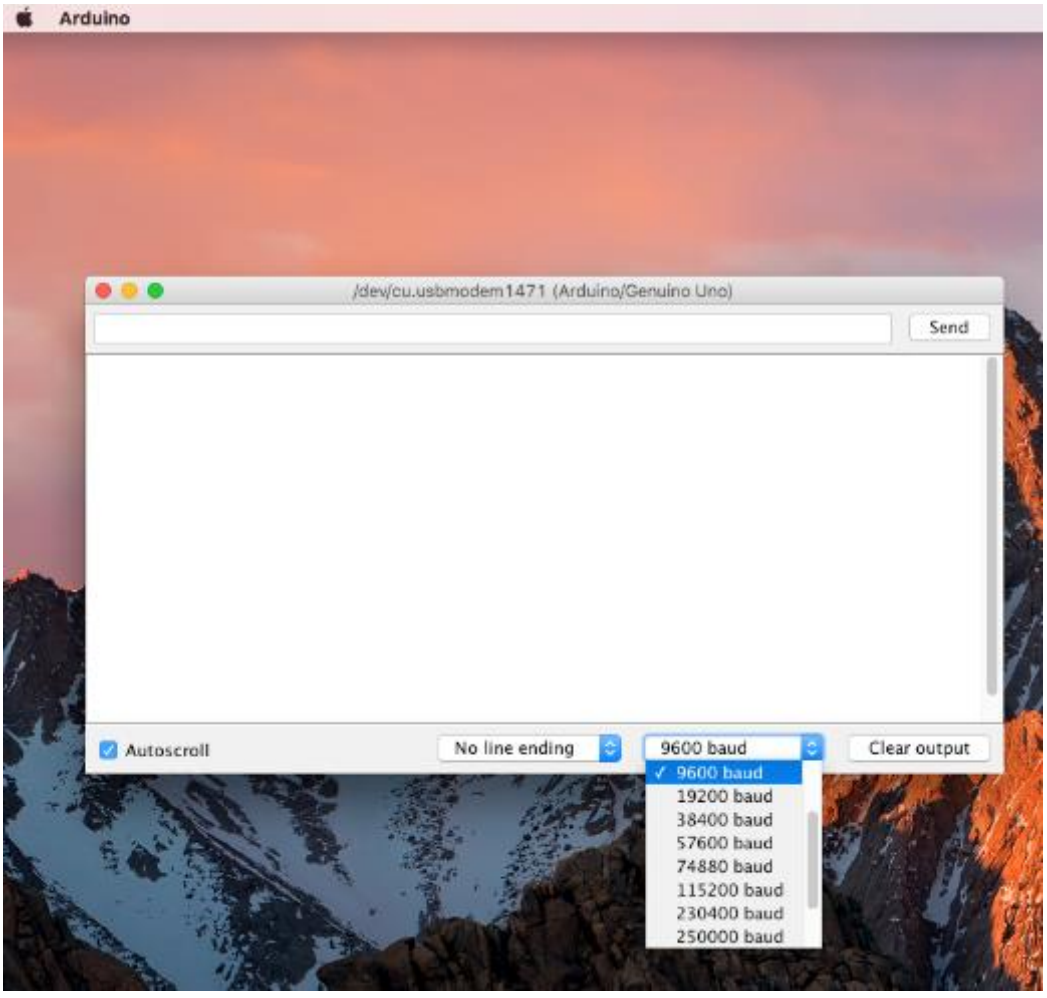
Einige Einstellungen fehlen im seriellen Monitor weshalb eventuellen fortgeschrittene Szenarien nicht möglich sind.

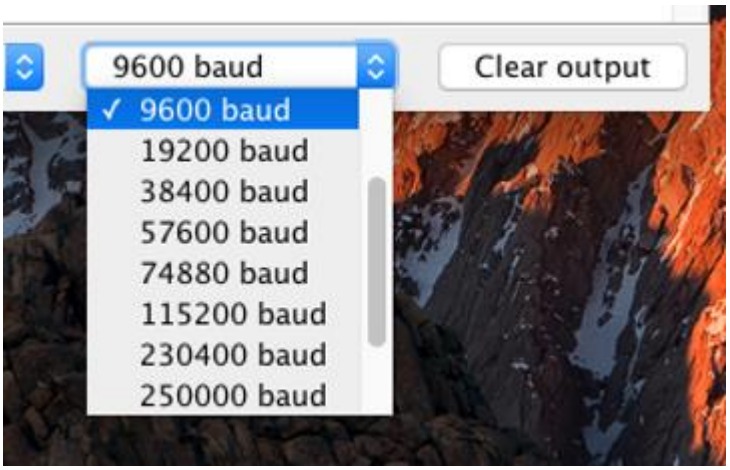
Mac Umgebung:

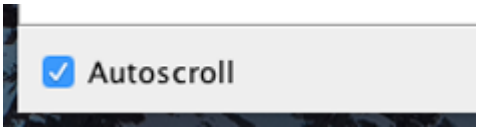
Der serielle Monitor verhält sich auf dem Mac genauso wie unter Windows. Im Folgenden sind trotzdem die entsprechenden Screenshots zu sehen.




Arduino File Edit Sketch Tools Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⇧⌘M
Serial Plotter ⇧⌘L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
/dev/cu.Bluetooth-Incoming-Port
Blink
Turns an LED on for one second, then off for one second, repeatedly.
Most Arduinos have an on-board LED you can control. On the UNO, it's attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN sets the correct LED pin independent of which board is used.
If you want to know what pin the on-board LED is connected to on your model, check the Technical Specs of your board at:
https://www.arduino.cc/en/Main/Products
modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman
This example code is in the public domain.
Done compiling.
Sketch uses 928 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables.


Arduino
/dev/cu.usbmodem1471 (Arduino/Genuino Uno)
Send
Autoscroll
No line ending
9600 baud
Clear output
✓ 9600 baud
19200 baud
38400 baud
57600 baud
74880 baud
115200 baud
230400 baud
250000 baud


9600 baud
Clear output
✓ 9600 baud
19200 baud
38400 baud
57600 baud
74880 baud
115200 baud
230400 baud
250000 baud

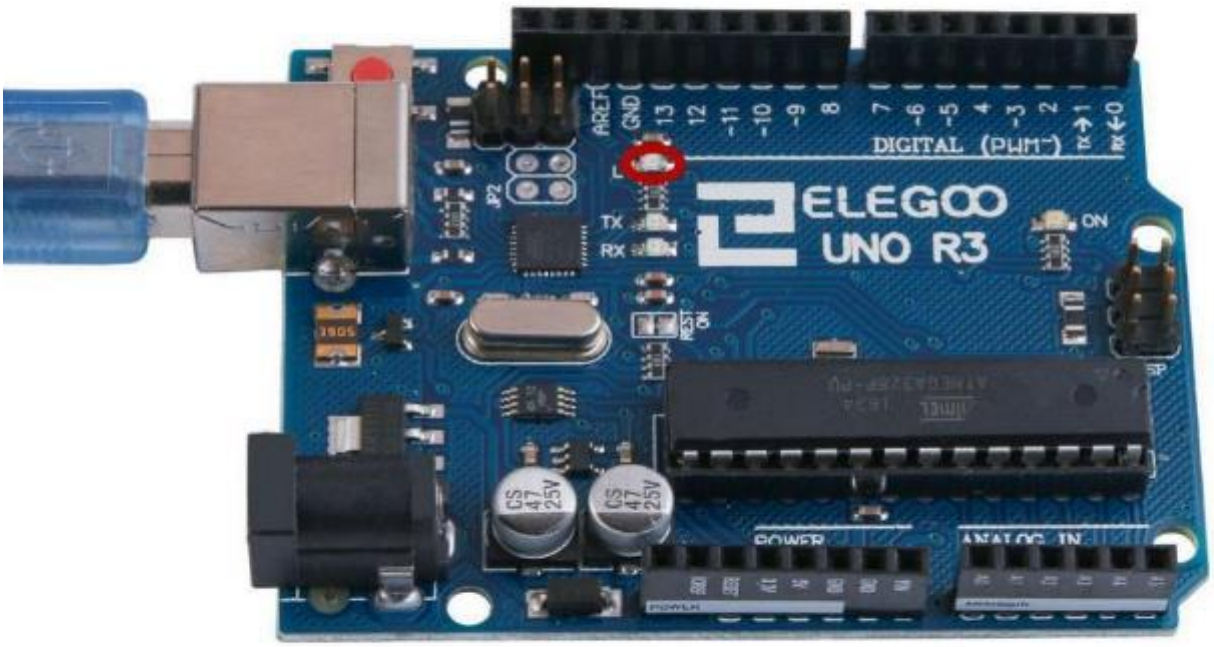

Autoscroll

Lektion 3 Blinkende LED

Übersicht

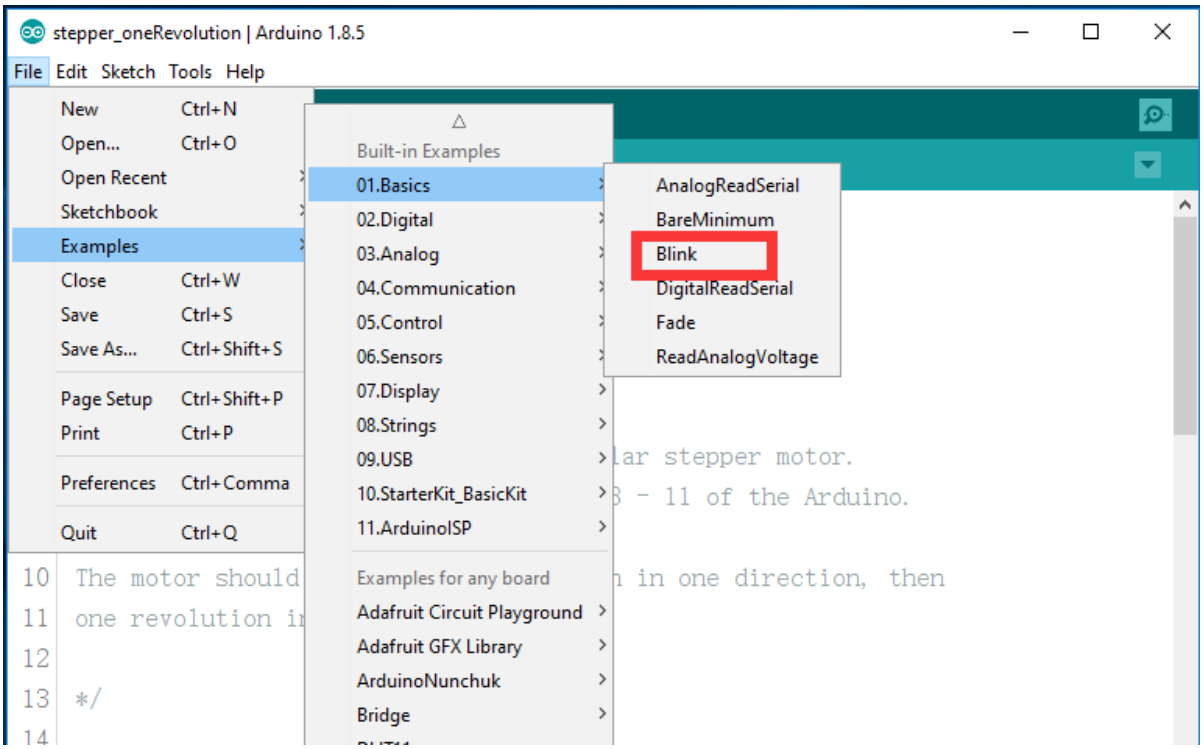
In dieser Lektion lernen Sie, wie Sie Ihr UNO R3-Controllerboard so programmieren, dass es die eingebaute LED des Arduino blinkt und wie Sie Programme herunterladen (also auf den Arduino) können.

Das UNO R3-Board hat auf beiden Seiten Steckverbinder, die für die Verbindung zu anderer Elektronik im allgemeinen oder aufsteckbaren "Shields" gedacht sind und die seine Leistungsfähigkeit erweitern (z. B. Shields zur Motorsteuerung). Das Board hat auch eine LED, die Sie von Ihren Sketches aus steuern können. Diese LED ist auf der Platine UNO R3 fest eingebaut und wird oft als die "L"-LED bezeichnet, da sie so auf der Platine beschriftet ist.

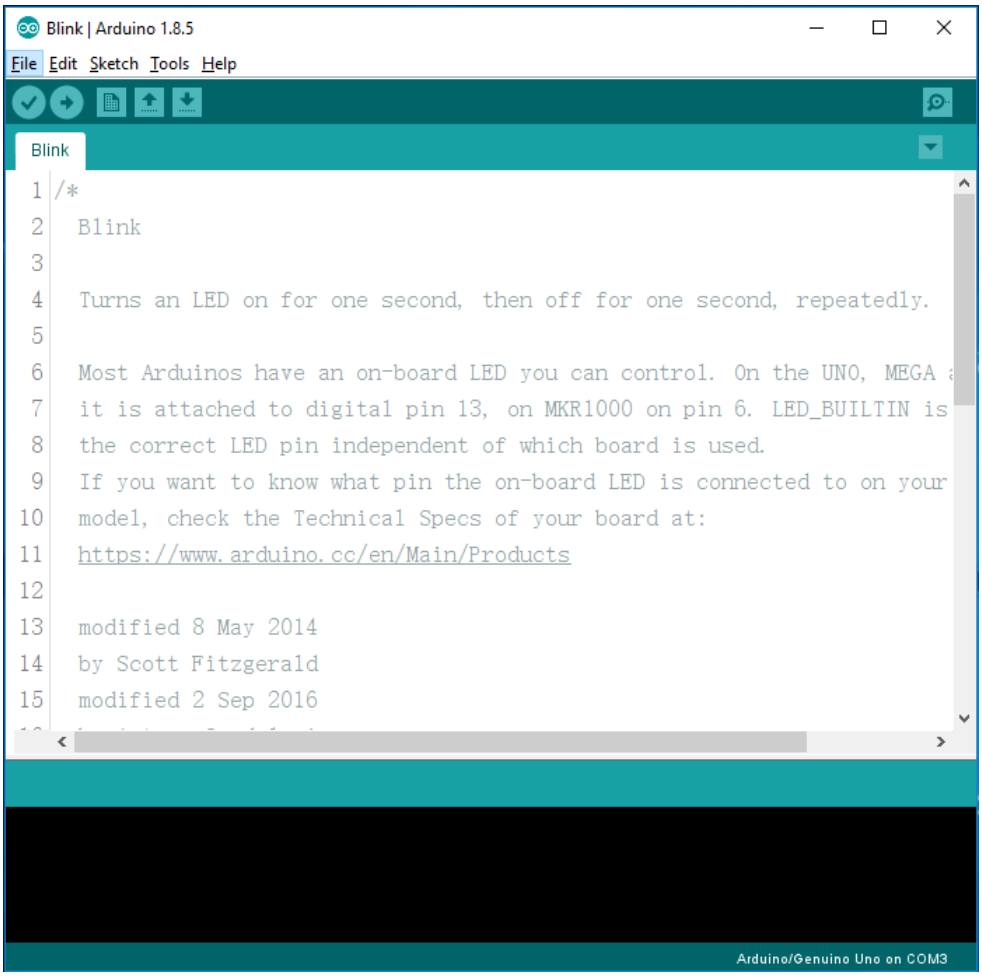


Sie werden vielleicht feststellen, dass die L-LED Ihres UNO R3 Boards bereits blinkt, wenn Sie es an einen USB-Stecker anschließen. Das liegt daran, dass die Platinen in der Regel mit dem vorinstallierten Sketch "Blink" ausgeliefert werden. In dieser Lektion werden wir das UNO R3-Board mit unserem eigenen Blink-Sketch neu programmieren und dann die Blinkfrequenz ändern.

In Lektion 0 haben Sie Ihre Arduino IDE eingerichtet und sichergestellt, dass Sie die richtige serielle Schnittstelle für den Anschluss an Ihr UNO R3-Board finden. Nun ist es an der Zeit, diese Verbindung zu testen und Ihr UNO R3-Board zu programmieren. Die Arduino IDE enthält eine große Sammlung von Beispielsketches, die Sie laden und verwenden können. Dazu gehört auch ein Beispielsketch für das Blinken der LED 'L'.
Laden Sie den Sketch "Blink", den Sie im Menüsystem der IDE unter Datei > Beispiele > 01.Basics finden.

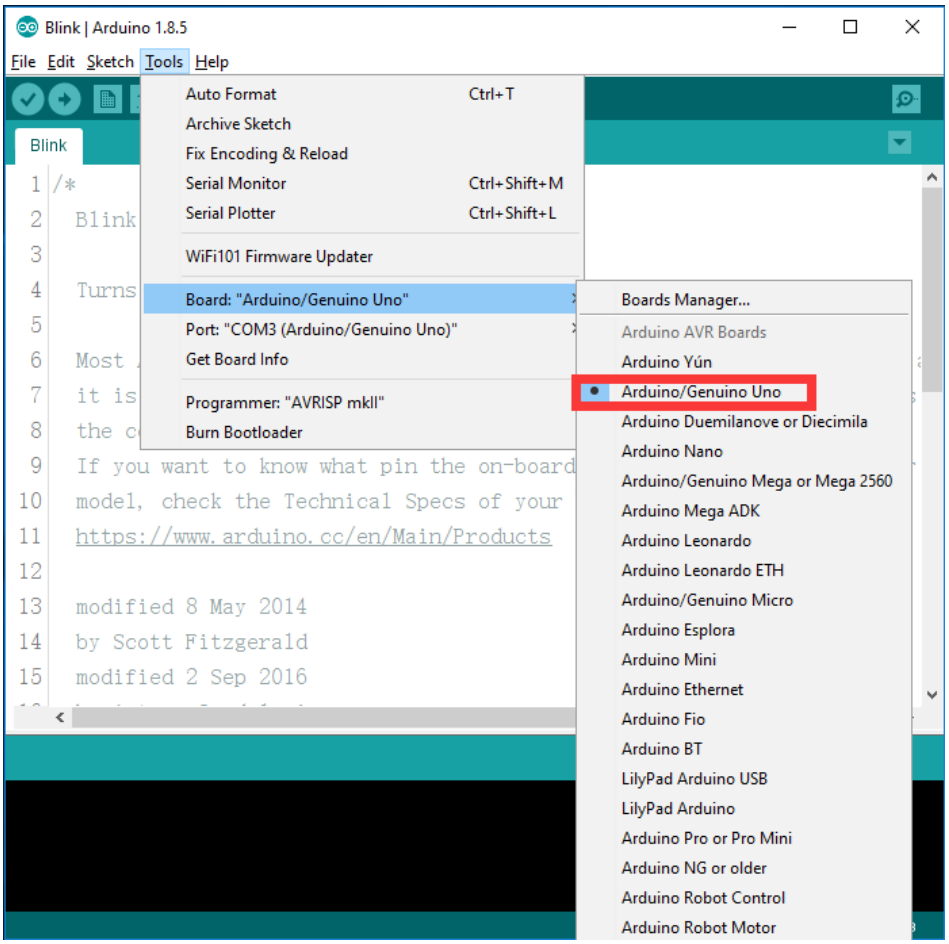


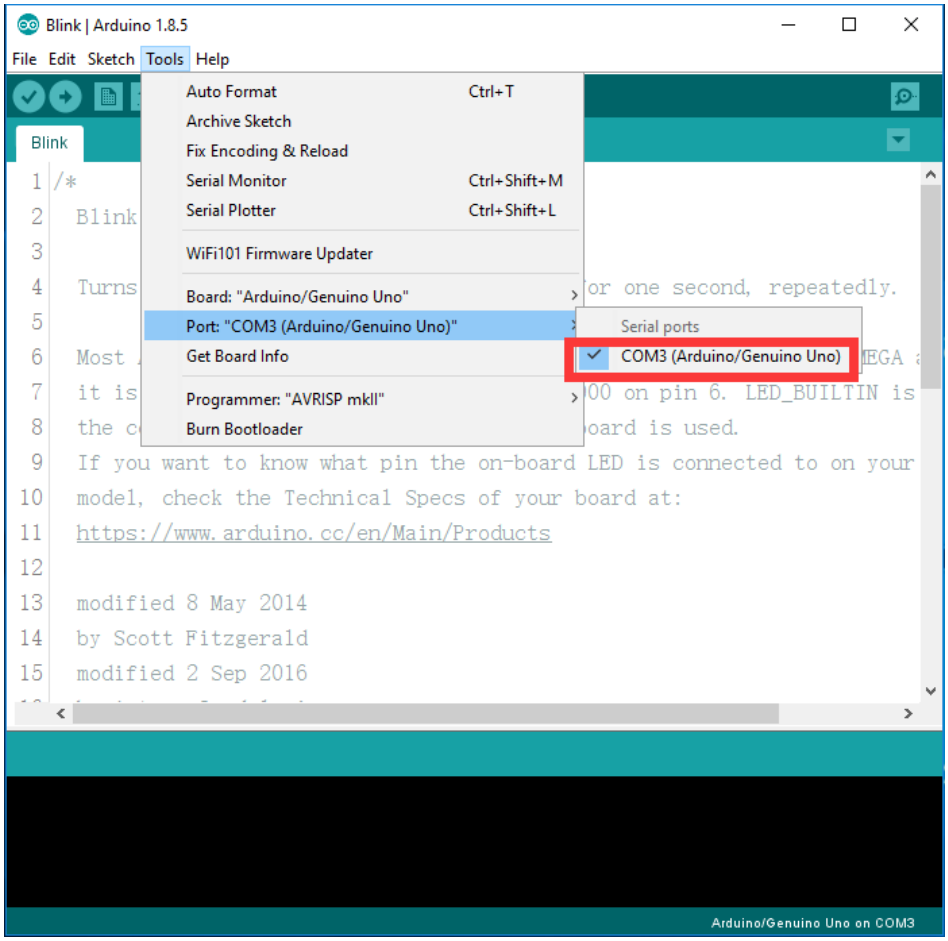
Wenn sich das Sketchfenster öffnet vergrößern Sie es, so dass Sie den gesamten Sketch im Fenster sehen können.



Die Beispielskizzen (oder Sketches), die der Arduino IDE beiliegen, sind schreibgeschützt. Das heißt, Sie können sie auf ein UNO R3-Board hochladen, aber wenn Sie sie ändern, können Sie sie nicht unter dem selben Dateinamen speichern.

Schließen Sie Ihr Arduino-Board mit dem USB-Kabel an Ihren Computer an und überprüfen Sie, ob die Einstellungen für "Board" und "Port" korrekt sind.


Blink | Arduino 1.8.5
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM3 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora
Arduino Mini
Arduino Ethernet
Arduino Fio
Arduino BT
LilyPad Arduino USB
LilyPad Arduino
Arduino Pro or Pro Mini
Arduino NG or older
Arduino Robot Control
Arduino Robot Motor
Blink
/*
Blink
Turns
Most
it is
the c
If you want to know what pin the on-board
model, check the Technical Specs of your
https://www.arduino.cc/en/Main/Products
modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016

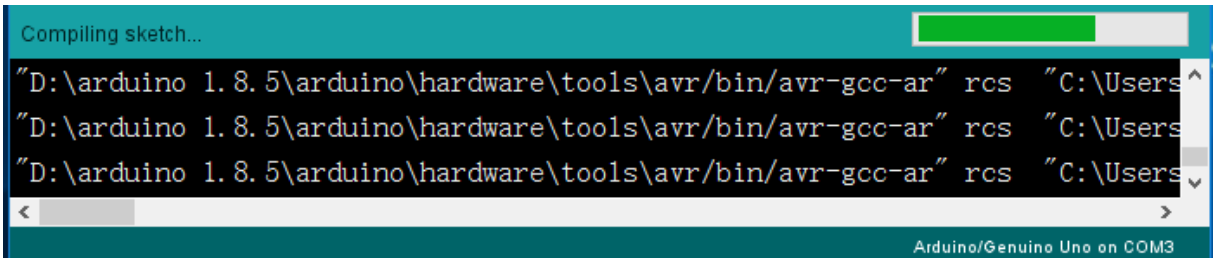


Hinweis: Der Boardtyp und die serielle Schnittstelle bei Ihnen sind nicht notwendigerweise identisch mit denen in der Abbildung. Wenn Sie z.B. ein Arduino Mega Board verwenden, dann müssen Sie Mega 2560 als Board Type wählen. Dasselbe gilt entsprechend für andere Arduino Boards. Und der serielle Port, ist auf jedem Computer ein anderer. Unter Windows lauten die Ports allerdings alle „COM“ und eine Zahl. In unserem Beispiel COM3. Nachdem der Boardname hinter dem Port angezeigt wird, ist der passende Port leicht zu finden. Bitte beachten Sie auf manchen Computern wird der COM Port erst aktiviert, wenn das Arduino Board an einen USB Port angeschlossen ist.

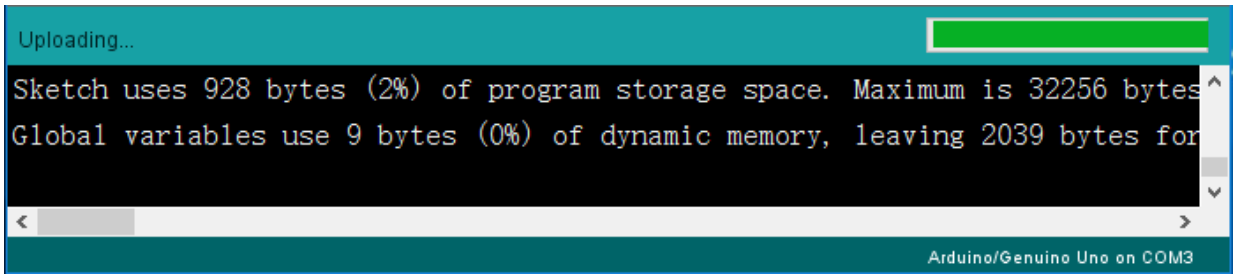
Die Arduino IDE zeigt die aktuellen Einstellungen für das Board am unteren Rand des Fensters an.

Klicken Sie auf die Schaltfläche "Hochladen“ (englisch: Upload). In der Symbolleiste ist dies die zweite Schaltfläche von links, dargestellt durch einen Pfeil, der nach rechts zeigt.

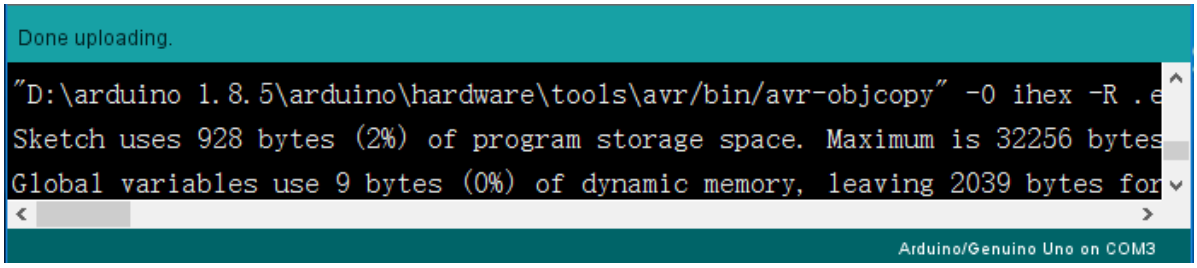
Wenn Sie sich den Statusbereich der IDE ansehen, sehen Sie einen Fortschrittsbalken und eine Reihe von Meldungen. Zuerst wird es heißen: "Sketch wird kompiliert...". Dadurch wird der Sketch in ein Format konvertiert, das für den Upload auf das Board geeignet ist.



Als nächstes wechselt der Status auf „Hochladen“. An diesem Punkt sollten die LEDs auf dem Arduino anfangen zu flackern, während der Sketch (Skizze, also das Programm) übertragen wird.



Abschließend wechselt der Staus zu "Hochladen abgeschlossen".



Die andere Nachricht sagt uns, dass der Sketch 928 Bytes der 32.256 Bytes verwendet, die zur Verfügung stehen. Falls das Hochladen nicht funktioniert, erhalten Sie nach dem Compilieren des Sketches die folgende oder eine ähnliche Fehlermeldung:

Dieser Fehler ist allgemeiner Natur und kann eines von drei Dingen bedeuten: Entweder ist Ihr Board nicht angeschlossen, es wurde die falsche serielle Schnittstelle ausgewählt oder die passenden Treiber wurden nicht installiert. Wenn letzteres der Fall ist, gehen Sie zurück zu Lektion 0 und überprüfen Sie Ihre Installation.

Sobald das Hochladen abgeschlossen ist, sollte das Board automatisch neu gestartet werden und anfangen zu blinken.

Anmerkung：Ein großer Teil dieses Sketches besteht aus Kommentaren. Dies sind keine eigentlichen Programmanweisungen, sondern sie erklären nur, wie das Programm funktioniert. Sie sind für Sie da. Alles zwischen /* und */ am Anfang des Sketches ist ein Blockkommentar, der erklärt wofür der Sketch gedacht ist. Die Kommentare der Beispiele, die von Arduino mitgeliefert werden sind nur auf Englisch verfügbar. Soweit Code in diesem Dokument vorgestellt wird, werden die Kommentare übersetzt.

Einzeilige Kommentare beginnen mit // und alles bis zum Ende der Zeile wird als Kommentar betrachtet. Vor dem Kommentar können Programmanweisungen stehen.

Die erste Programmzeile ist die 'setup' Funktion. Sie wird ausgeführt, wenn die Reset-Taste gedrückt wird. Sie wird auch immer dann ausgeführt, wenn das Board aus irgendeinem Grund neu gestartet wird, z.B. wenn es erstmals mit Strom versorgt wird oder nachdem ein Sketch hochgeladen wurde.

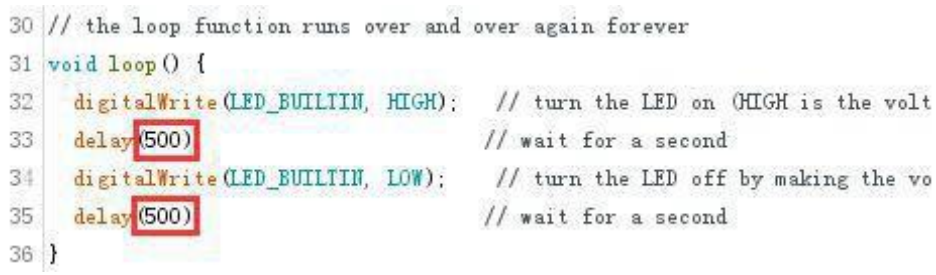
```
void setup() {
   // Initialisiert die eingebaute LED als Ausgang, LED_BUILTIN ist eine Konstante für PIN13
   pinMode(LED_BUILTIN, OUTPUT);
}
```

Jeder Arduino-Sketch muss über eine „Setup“-Funktion verfügen und Sie können eigene Anweisungen zwischen den beiden geschweiften Klammern einfügen.
In diesem Fall gibt es nur einen einzigen Befehl, der wie der Kommentar sagt, dem Arduino-Board mitteilt, dass wir den LED-Pin als Ausgang verwenden werden.

Es ist auch zwingend erforderlich, dass ein Sketch über eine „loop“-Funktion verfügt. Im Gegensatz zur Funktion „Setup“, die nur einmal ausgeführt wird, wird die „loop“-Funktion solange ausgeführt, bis ein Neustart des Boards erfolgt. D.h. sobald die Ausführung der Anweisungen innerhalb der „loop“ Funktion beendet sind, fängt die Funktion von vorne an.

```
void loop() {
   digitalWrite(LED_BUILTIN, HIGH);     // schaltet die LED an
   delay(1000);                         // 1000ms=1s warten
   digitalWrite(LED_BUILTIN, LOW);      // schaltet die LED ab
   delay(1000);                         // 1000ms=1s warten
}
```

Innerhalb der Loop-Funktion schalten die Anweisungen zuerst die LED ein, dann wird eine Sekunde gewartet, dann wird die LED wieder ausgeschaltet und wieder eine Sekunde gewartet. Anschließend geht es wieder von vorne los.
Sie werden jetzt Ihre LED schneller blinken lassen. Wie Sie vielleicht schon vermutet haben, liegt der Schlüssel dazu in der Änderung des Parameters in () für den Befehl „delay“.



Diese Verzögerungszeit ist in Millisekunden angegeben. Wenn Sie also möchten, dass die LED zwei Mal so schnell blinkt, ändern Sie den Wert von 1000 auf 500.

Laden Sie den Sketch erneut hoch und Sie sollten sehen, dass die LED schneller blinkt.

Lektion 4 DHT11 Temperatur- und Luftfeuchtigkeitssensor

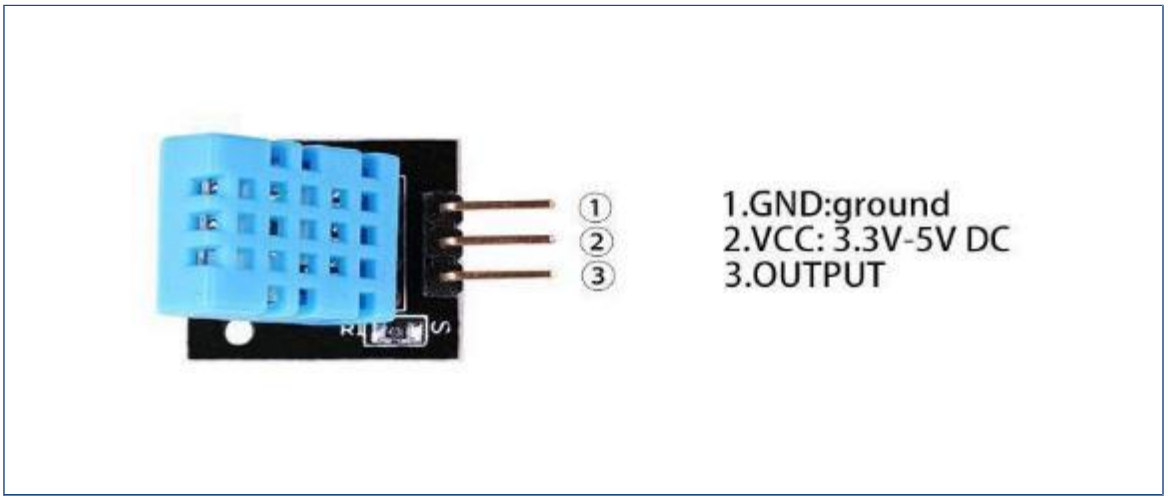
Übersicht

In diesem Tutorial lernen wir, wie man einen DHT11 Temperatur- und Feuchtesensor verwendet. Er ist genau genug für die meisten Projekte, die nur einen groben Überblick über die Feuchte- und Temperaturwerte haben müssen.

Wir werden eine Bibliothek verwenden, die speziell für diese Sensoren entwickelt wurde, so dass unser Code kurz und einfach wird.

Temperatur und Luftfeuchtigkeit

Ein Modul mit einem Temperatur-/Feuchtefühler Typ DHT11, Temperaturbereich: -20 - 60°C (+/-1°C), Rel. Luftfeuchtigkeit: 5-95% (+/-5%), Versorgungsspannung: 3V bis 5,5V. Mit eingebautem 10 K Ohm Pullup-Widerstand.



Benötigte Komponenten:

1 x Elegoo Uno R3

1x DHT11 Modul

3 x F-M Drähte (Female to Male DuPont wires)

Auszug aus dem Datenblatt des Sensors

Temperatur- und Feuchtesensor:

Produktparameter relative Feuchte:

Auflösung: 16Bit Wiederholgenauigkeit: ±1% RH

Genauigkeit: Bei 25℃ ±5%RH

Antwortzeit: 1 / e (63%) von 25℃ 6s

Hysterese: <± 0.3% RH

Produktparameter Temperatur:

Auflösung: 16Bit Wiederholgenauigkeit: ±0.2℃

Genauigkeit: Bei 25℃ ±2℃

Antwortzeit: 1 / e (63%) 10S

Elektrische Eigenschaften

Spannungsversorgung: DC 3.5～5.5V

Stromaufnahme: Messung 0.3mA, Stand-by 60μA

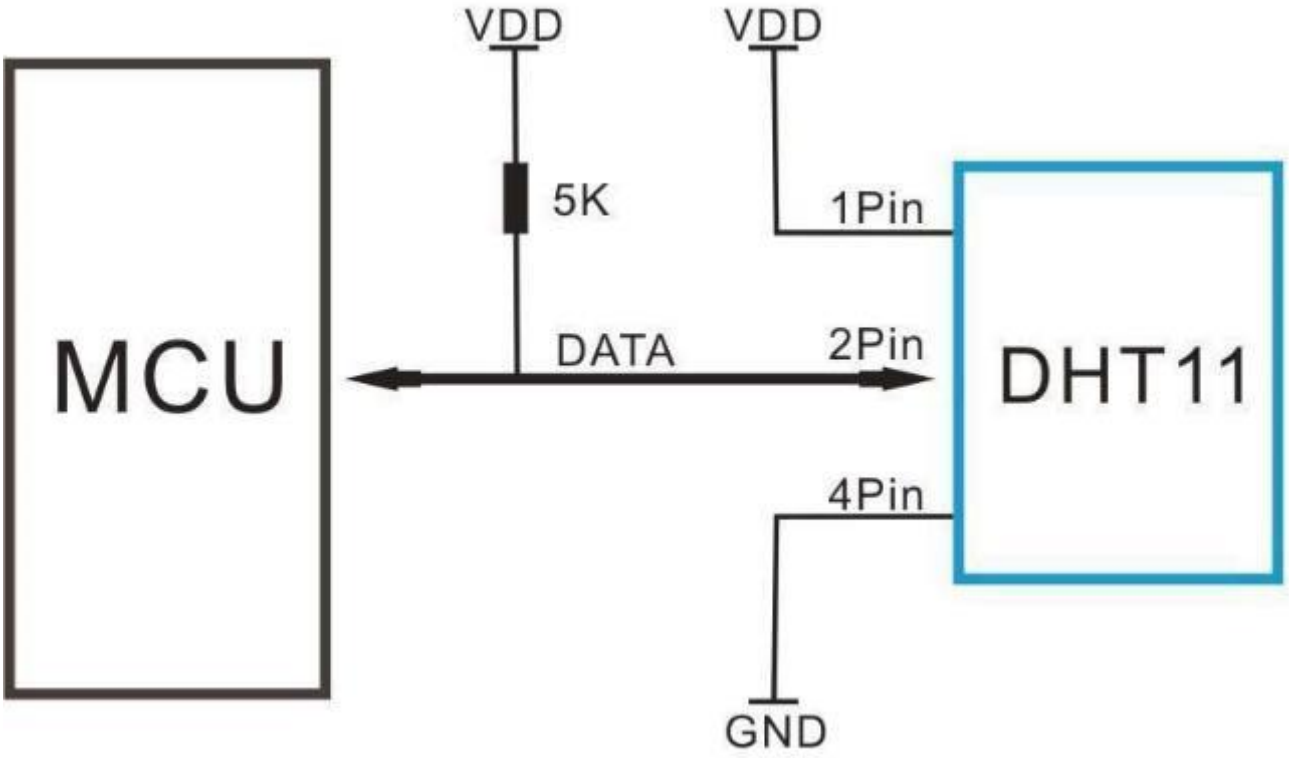
Abtastzeit: > 2s

Pinbeschreibung:

1. Versorgungsspannung 3.5～5.5V DC (DC=direct current=Gleichstrom)
2. DATA, Datenbus
3. NC, nicht verbunden
4. GND ground, Masse

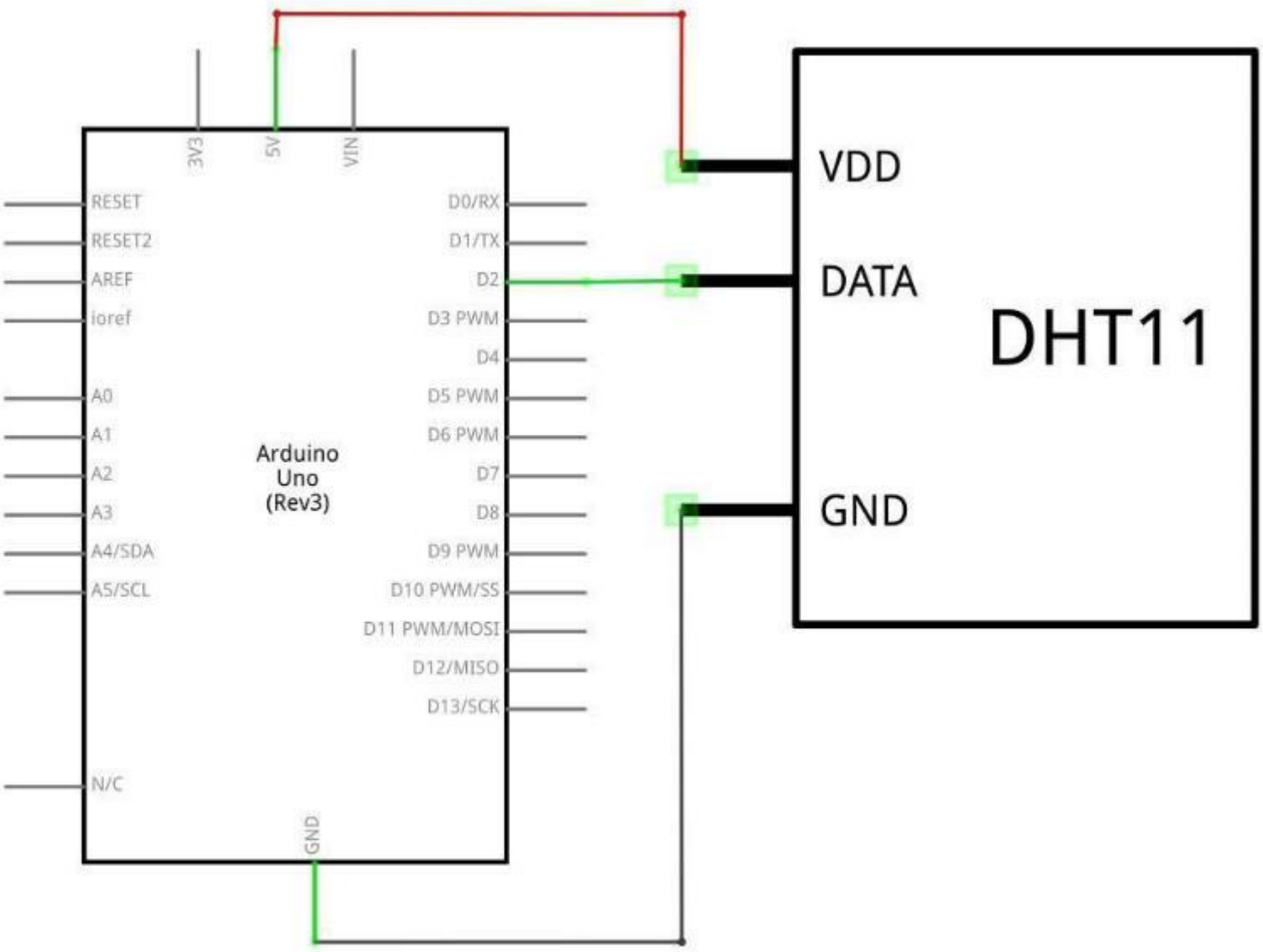
Ein paar Worte bezüglich ground/Masse. Die Masse ist der Bezugspunkt für alle Spannungen. Er wird willkürlich auf 0V gesetzt. Spannungen können sowohl größer als Masse sein oder kleiner. Als Anfänger kann man sich bei Arduino-Gleichstromanwendungen die Masse als Nullpunkt oder auch als Minuspol (bei symmetrischen Versorgungsspannungen gilt dies allerdings nicht) vorstellen.

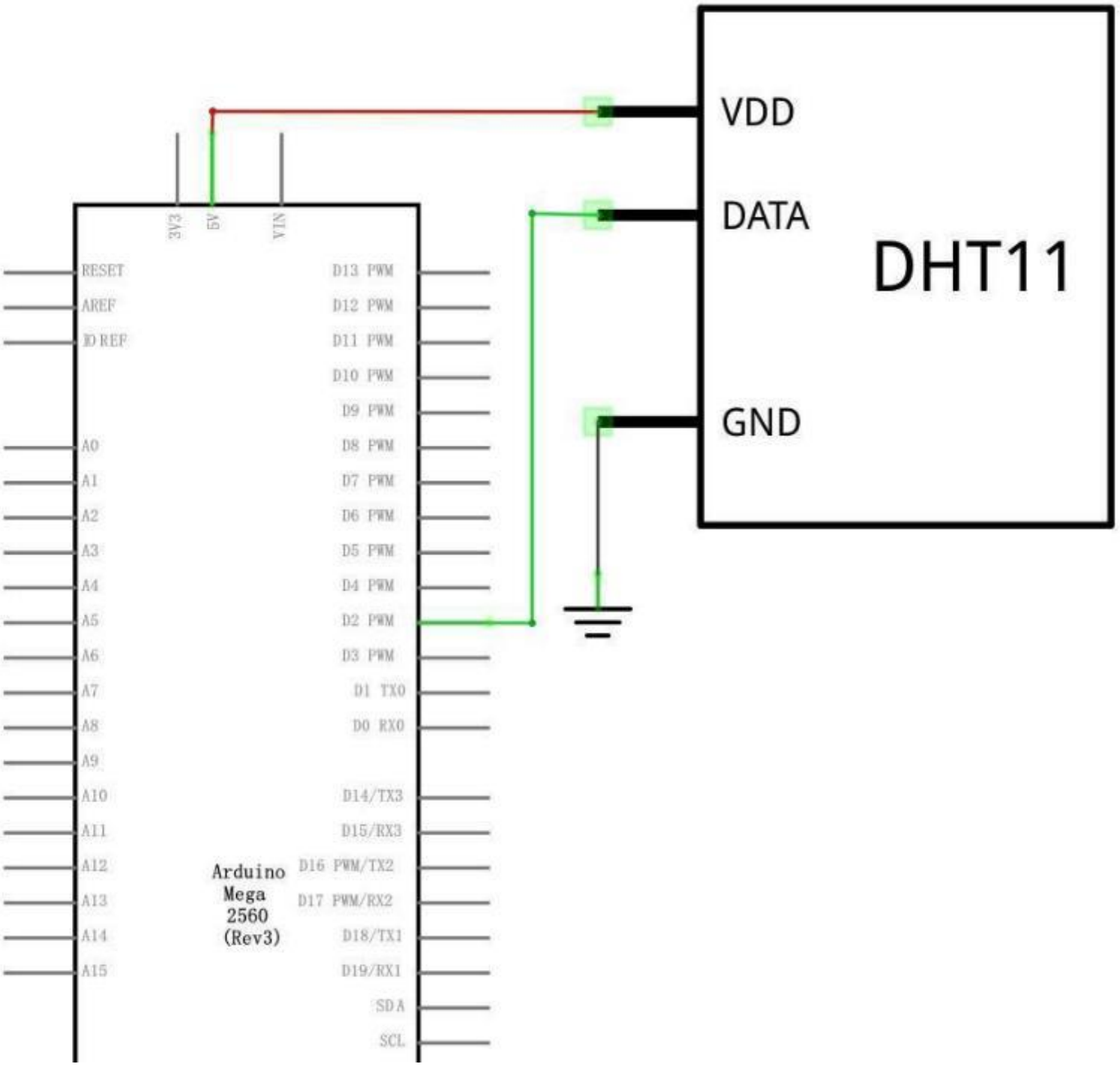
Typische Anwendung



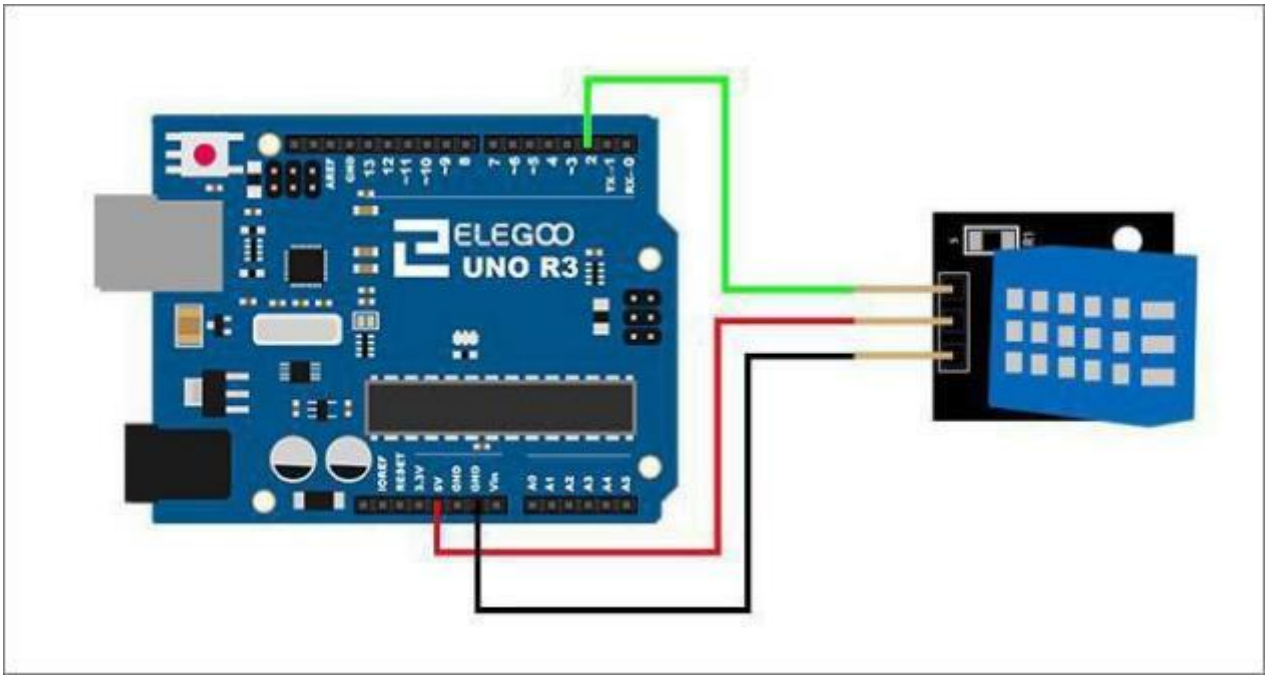
Verbindung

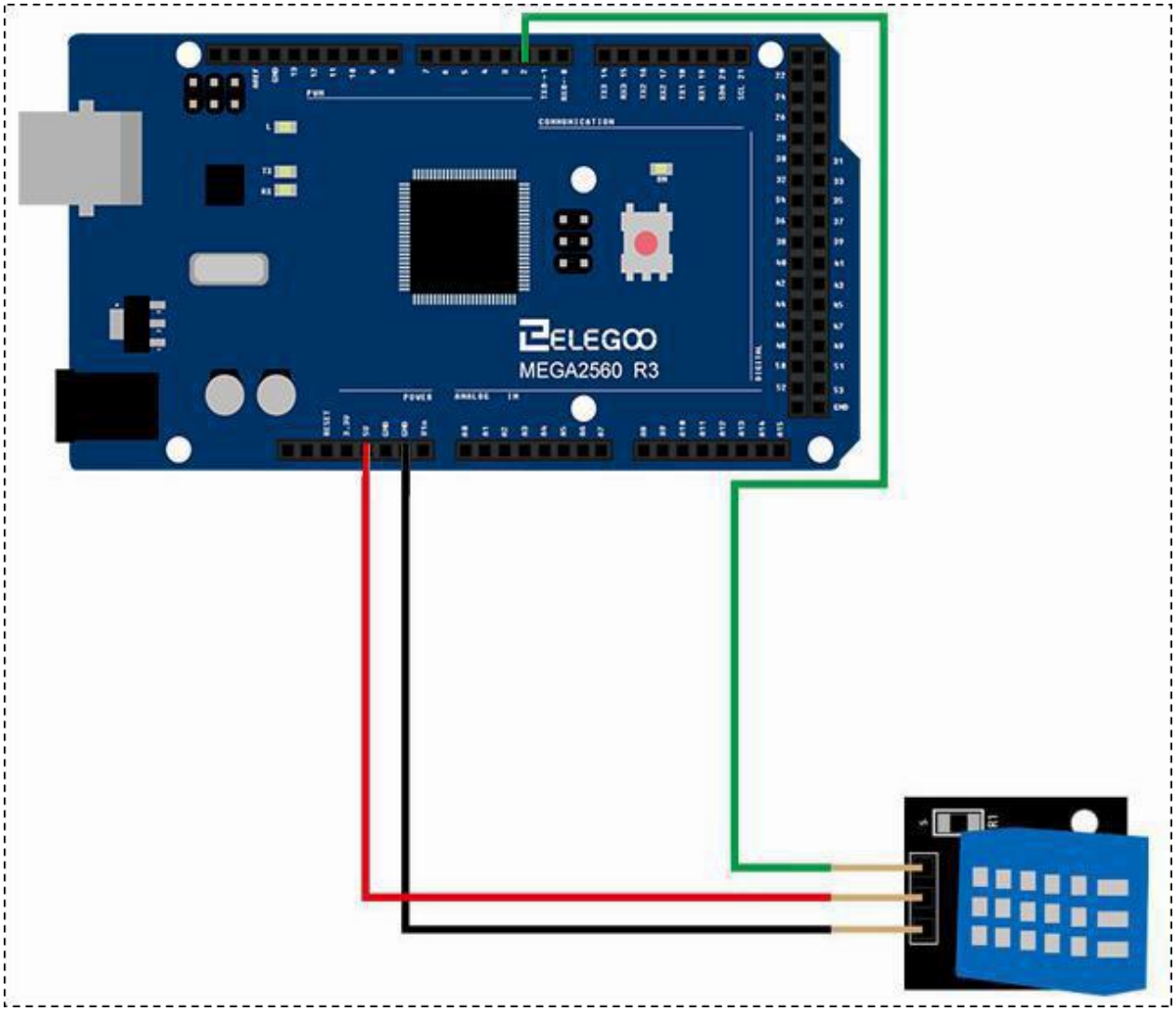
Schaltplan




VDD
DATA
GND
DHT11
3V3
5V
VIN
RESET
AREF
IOREF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
Arduino
Mega
2560
(Rev3)

Verdrahtungsplan



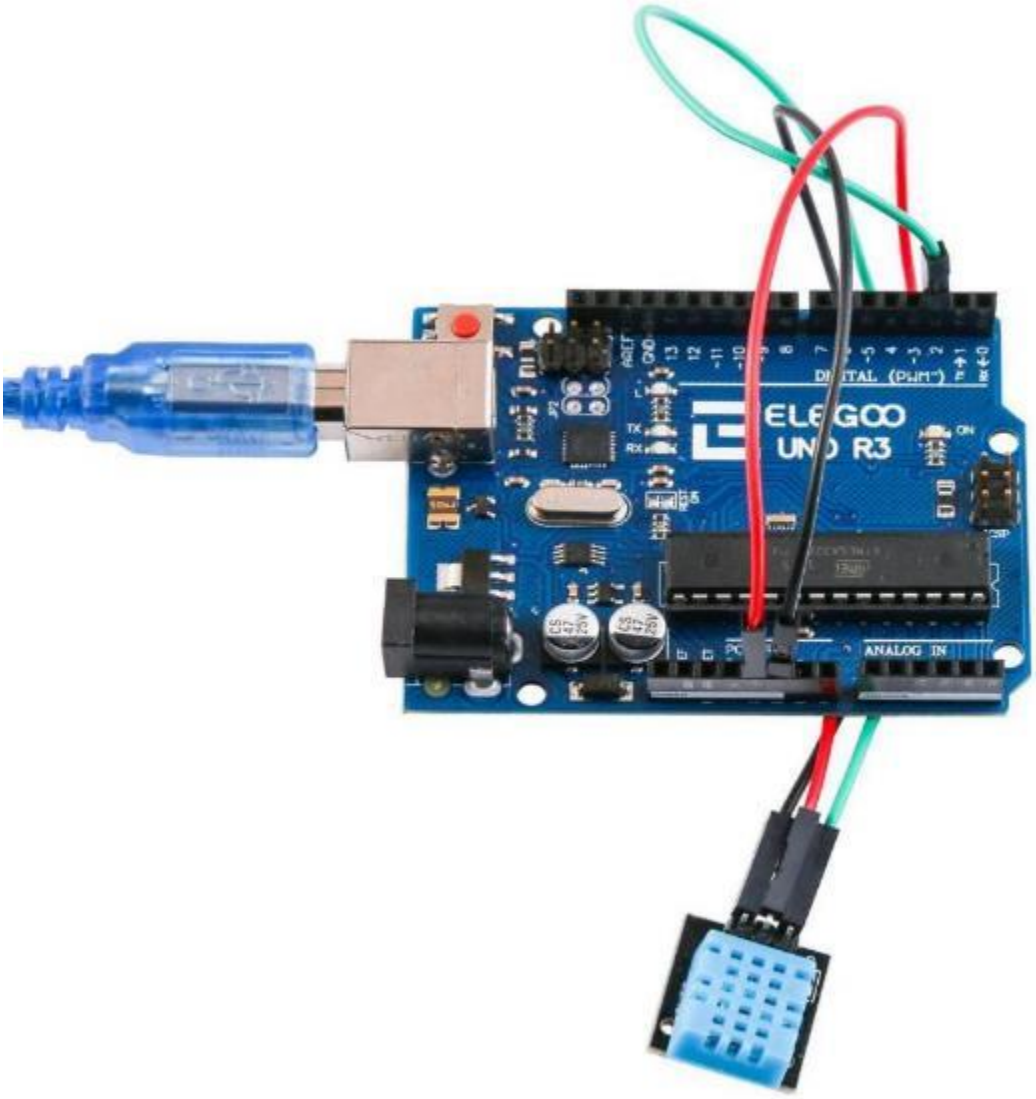


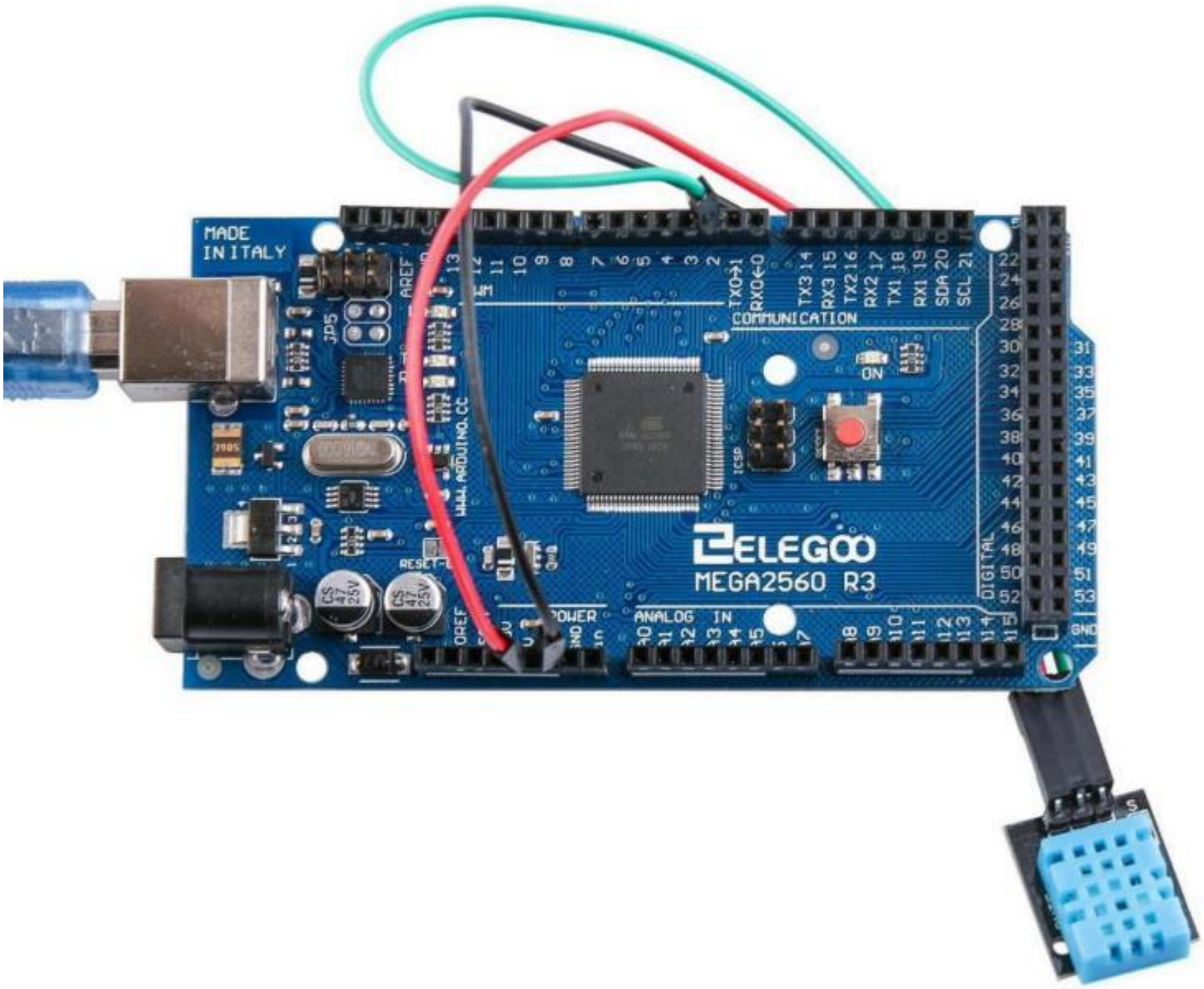
Wie Sie sehen, benötigen wir nur 3 Anschlüsse zum Sensor, da einer der Pins nicht verwendet wird. Die Anschlüsse sind: Spannung, Masse und Signal, die an jeden beliebigen Pin unserer UNO angeschlossen werden können.

Code

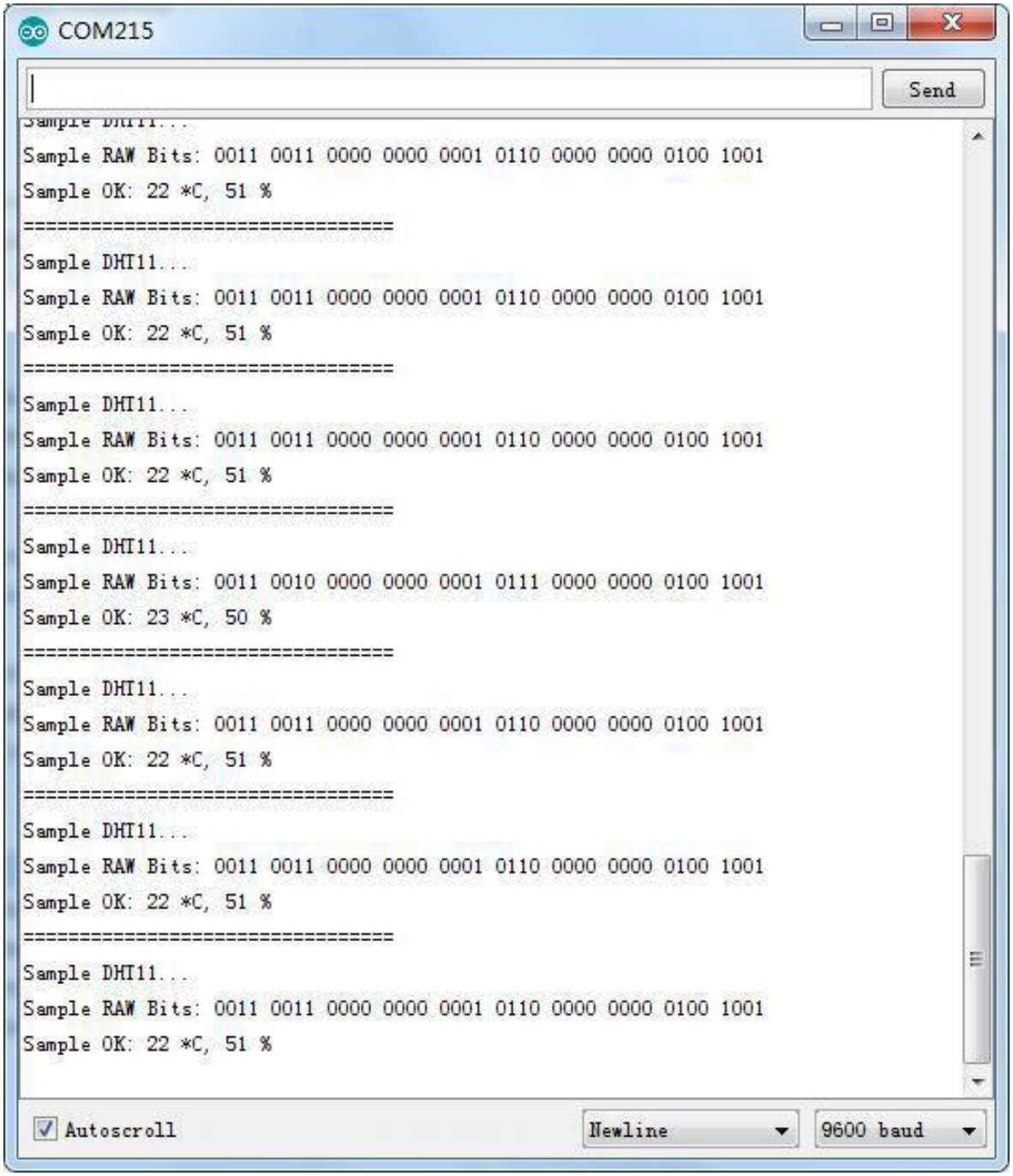
Nach der Verdrahtung öffnen Sie bitte das Programm im Code-Ordner (Lesson 4 TEMP AND HUMIDITY MODULE) und klicken Sie auf „Hochladen“, um das Programm hochzuladen.
Vergewissern Sie sich, dass Sie die < SimpleDHT>-Bibliothek installiert haben oder installieren Sie sie gegebenenfalls neu. Andernfalls wird Ihr Code nicht funktionieren (Kompilierfehler).
Ausführliche Informationen zum Tutorial zum Laden von Bibliotheksdateien finden Sie in Lektion 1.

Ergebnis




MADE
IN ITALY
COMMUNICATION
ON
ELEGOO
MEGA2560 R3
POWER
ANALOG IN
DIGITAL

Laden Sie das Programm hoch und öffnen Sie den seriellen Monitor. Der serielle Monitor sollte ungefähr so aussehen wie auf dem nachfolgenden Screenshot. Im Beispiel hat der Raum 22° und 51% Luftfeuchtigkeit. Details über den seriellen Monitor finden sind in Lektion 2.



Im Folgenden wird der Code abgedruckt und erklärt:

```
#include <SimpleDHT.h>    //Bindet die SimpleDHT Bibliothek ein
/*Wir definieren den Datenpin als PIN 2 am Arduino; der Datenpin des Moduls wird also an Pin2 des
Arduino angeschlossen.
int pinDHT11 = 2;
SimpleDHT11 dht11;
/*setup () Die Setup-Funktion () wird aufgerufen, wenn das Arduino-Board gestartet wird. Benutzen Sie
es, um Variablen zu initialisieren und anderen Code, der nur einmal ausgeführt werden soll. Diese
Funktion läuft nur einmal beim Einschalten des Boards. */
void setup()
 {
   /*Setzen der Baudrate des seriellen Ports auf 9600; Das ist auch der Standardwert im seriellen
Monitor*/
   Serial.begin(9600);
}
void loop () {
/* Alles was mit Serial.print anfängt wird im seriellen Monitor angezeigt, falls er geöffnet ist */
   Serial.println ("=================================");
   Serial.println ("Sample DHT11 with RAW bits...");
   byte temperature = 0;
   byte humidity = 0;
   byte data[40] = {0};
   int err = SimpleDHTErrSuccess;
   if ((err = dht11.read(pinDHT11, &temperature, &humidity, data)) != SimpleDHTErrSuccess) {
      Serial.print("Read DHT11 failed, err=");
```

```
        Serial.println(err);
        delay(2000); // Laut Datenblatt sollte die Abfragefrequenz >= 2s sein
        return;
    }
    Serial.print("Sample RAW Bits: ");
    for (int i = 0; i < 40; i++) {
        Serial.print((int)data[i]);
        if (i > 0 && ((i + 1) % 4) == 0) {
            Serial.print(' ');
        }
    }
    Serial.println("");

    Serial.print("Sample OK: ");
    Serial.print((int)temperature); Serial.print(" *C, ");
    Serial.print((int)humidity); Serial.println(" H");
    // Laut Datenblatt sollte die Abfragefrequenz >= 2s sein
    delay(2000);
}
```

Im obigen Beispielprogramm haben wir die folgenden Kontrollstrukturen kennengelernt:

(1)if

„if“ überprüft, ob eine bestimmte Bedingung erfüllt ist; Das Format einer „if“ Bedingung ist:

if (Bedingung) { // Code der ausgeführt wird wenn die Bedingung erfüllt ist }

Falls die Bedingung wahr ist, wird der Code in geschweiften Klammern ausgeführt, ansonsten wird er übersprungen.

Die geschweiften Klammern können weggelassen werden, dann gilt alles bis zum nächsten Semikolon als Code, der ausgeführt werden soll, falls die Bedingung wahr ist. Dies ist allerdings fehleranfällig, insofern empfiehlt es sich insbesondere für Anfänger, die geschweiften Klammern immer zu setzen.

Im Folgenden drei Beispiele von korrekten ifs:

if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LEDpin, HIGH);}

Falls die Bedingung ein Zahlenvergleich ist, wird einer der folgenden Operatoren benötigt:

Operatoren: x == y (x ist gleich y), x != y (x ist ungleich y), x < y (x ist kleiner als y), x > y (x ist größer als y), x <= y (x ist kleiner oder gleich y), x >= y (x ist größer oder gleich y)

Warnung:

Achten Sie darauf, dass Sie nicht versehentlich das einzelne Gleichheitszeichen verwenden (z.B. if (x = 10)). Das einzelne Gleichheitszeichen ist der Zuweisungsoperator und setzt x auf 10. Verwenden Sie stattdessen das doppelte Gleichheitszeichen (z.B. if (x == 10)), das der Vergleichsoperator ist und überprüft ob x gleich 10 ist oder nicht. Die letzte Aussage ist nur wahr, wenn x gleich 10 ist, aber die erste Aussage wird immer wahr sein. Das liegt daran, dass C die Anweisung, if (x=10) wie folgt von innen nach außen auswertet: 10 wird x zugeordnet, also enthält x jetzt 10. Damit ist die Bedingung 10, was immer TRUE/wahr ist, da jede Zahl ungleich Null als TRUE/wahr ist (gilt für die meisten Programmiersprachen). Folglich ist if (x = 10) immer TRUE, was nicht das gewünschte Ergebnis ist, wenn

man eine if-Anweisung verwendet. Zusätzlich wird die Variable x auf 10 gesetzt, was ebenfalls keine gewünschte Aktion ist.

if kann auch Teil einer verzweigten Kontrollstruktur sein, indem Sie das if...else Konstrukt verwenden.

(2)for Anweisungen

Die for-Anweisung wird verwendet, um einen Block von Anweisungen, die in geschweifte Klammern eingeschlossen sind, zu wiederholen. Ein Zähler wird in der Regel zum Inkrementieren und Beenden der Schleife verwendet. Die for-Anweisung ist für jede sich wiederholende Operation nützlich und wird oft in Kombination mit Arrays verwendet.

Der for-Schleifenkopf besteht aus drei Teilen: for (Initialisierung; Bedingung; Inkrement) { //Anweisung(en); } Die Initialisierung erfolgt als erstes und genau einmal. Jedes Mal, wenn die Schleife durchlaufen wird, wird die Bedingung getestet; wenn sie wahr ist, wird der Anweisungsblock und das Inkrement ausgeführt, dann wird die Bedingung erneut getestet. Wenn die Bedingung falsch wird, endet die Schleife.

Beispiel:

Die Schleife wird 255-mal durchlaufen. Jedes Mal wird die Methode analogWrite aufgerufen, wobei der zweite Parameter jedes mal um eins höher ist. D.h. die Spannung am PIN steigt it jedem Schleifendurchlauf an.

```
for (int i=0; i < 255; i++)
{
        analogWrite(PWMpin, i);
        delay(10);
}
```

Tipp:

Die for-Schleifen in C sind viel flexibler als in einigen anderen Programmiersprachen, wie z.B. BASIC. Jeder der drei Elemente im Header kann ausgelassen werden, solange die Semikolons vorhanden sind. Alle drei Anweisungen können beliebige Ausdrücke sein. Auch wenn so eine Verwendung ungewöhnlich ist, kann sie hilfreich sein bestimmte Programmierprobleme zu lösen.

Lektion 5 DS18B20 TEMPERATURMODUL

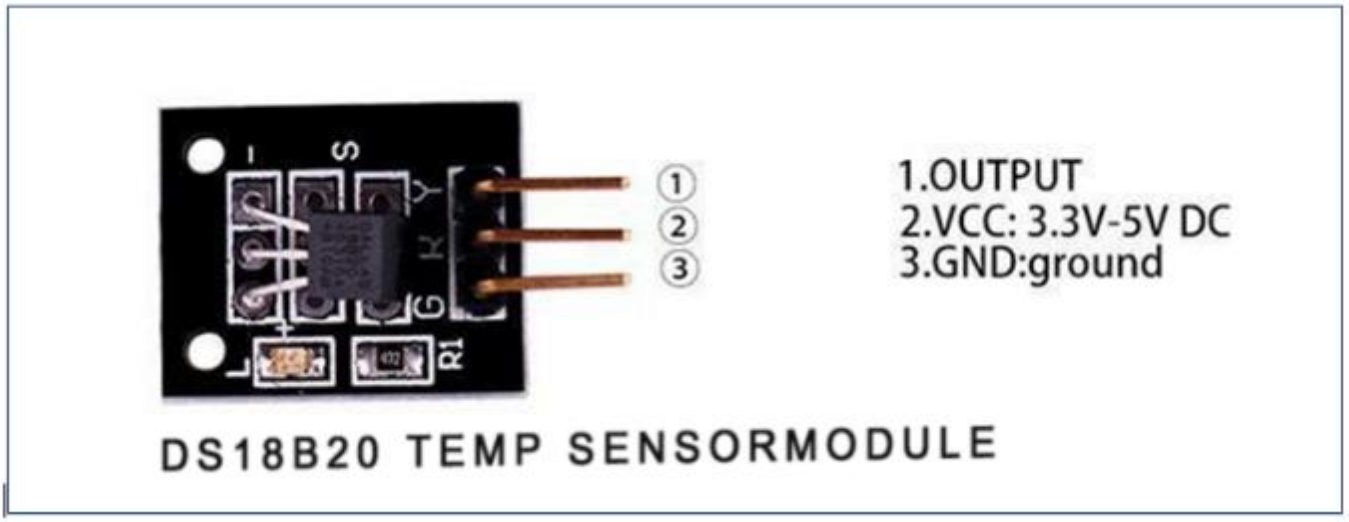
Überblick

In diesem Experiment werden wir lernen, wie man das DS18B20 Modul benutzt, um die Umgebungstemperatur zu messen.
Das Modul kommuniziert mittels des sogenannten 1-Wire Buses mit dem Arduino. D.h. der Sensor misst die Temperatur und schickt sie mittels eines einzigen Datenpins an den Arduino. An den Bus können auch mehrere Sensoren mit unterschiedlichen Adressen gleichzeitig angeschlossen werden (was wir aber in diesem Tutorial nicht machen werden). Die gesamte Auswertelogik übernehmen das Modul und entsprechende schon vorhandene Arduino Bibliotheken. Das Auslesen der Temperaturwerte wird dadurch sehr einfach. Eine ausführliche Beschreibung des 1-Wire Buses würde den Rahmen dieses Tutorials sprengen.

Beschreibung des Moduls

Ein Modul mit einem digitalen '1-Wire' Temperatursensor (DS18B20). Ein 4,7K Ohm Pullup-Widerstand für das Bussignal ist auf dem Modul schon enthalten.
Zusätzliche Sensoren können an den Bus angeschlossen und individuell adressiert werden. Unabhängig von der Anzahl der angeschlossenen Sensoren sollte nur ein Pullup-Widerstand an den Bus angeschlossen werden.

- Temperaturbereich: -55 to +125°C
- Genauigkeit: 0.5°C
- Auflösung: 9-12Bit, abhängig vom Programm



Benötigte Komponenten:

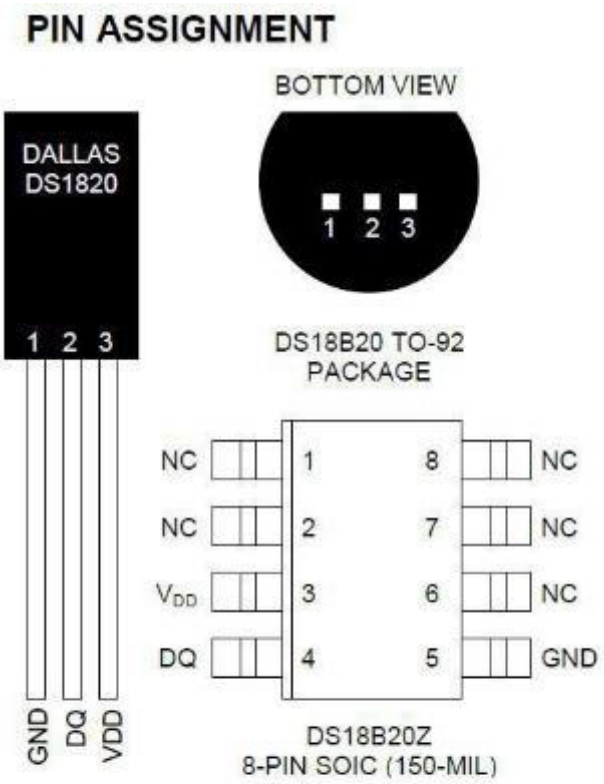
1 x Elegoo UnoR3

1 x USB Kabel

1 x DS18B20 Modul

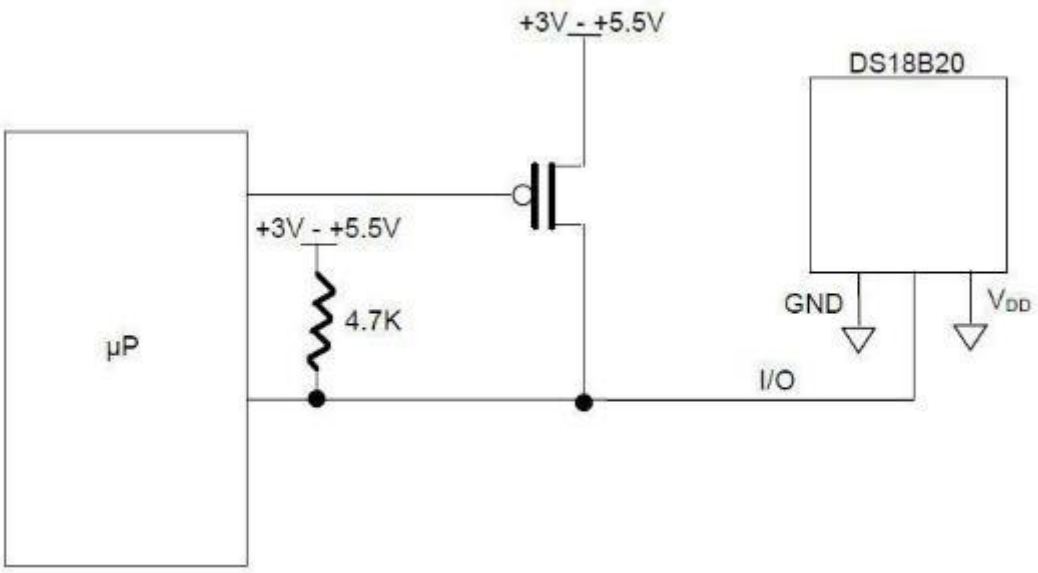
3 x F-M Drähte

Komponenteneinführung DS18B20:



PIN DESCRIPTION

GND	- Ground
DQ	- Data In/Out
V_{DD}	- Power Supply Voltage
NC	- No Connect



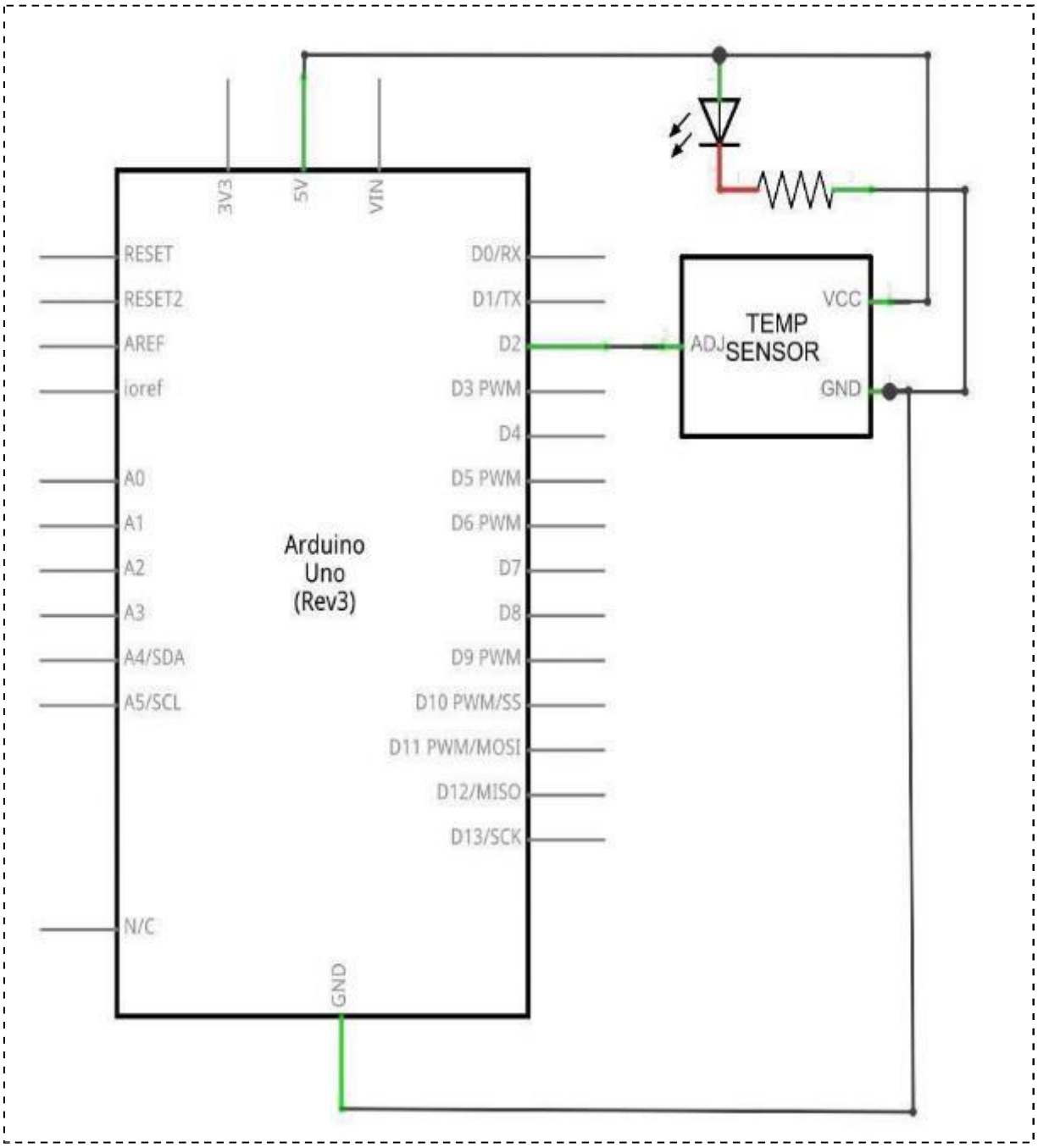
Prinzip

Das Modul DS18B20 verwendet einen Bus. Der Versorgungsspannungsbereich geht von 3,0V bis 5,5V. Es kann Temperaturbereiche von -55 Grad bis +125 Grad mit einer Genauigkeit von +/-0,5°C messen. Die Temperatur selbst wird durch 12 Bit dargestellt und kann maximal alle 750 Millisekunden abgefragt werden.

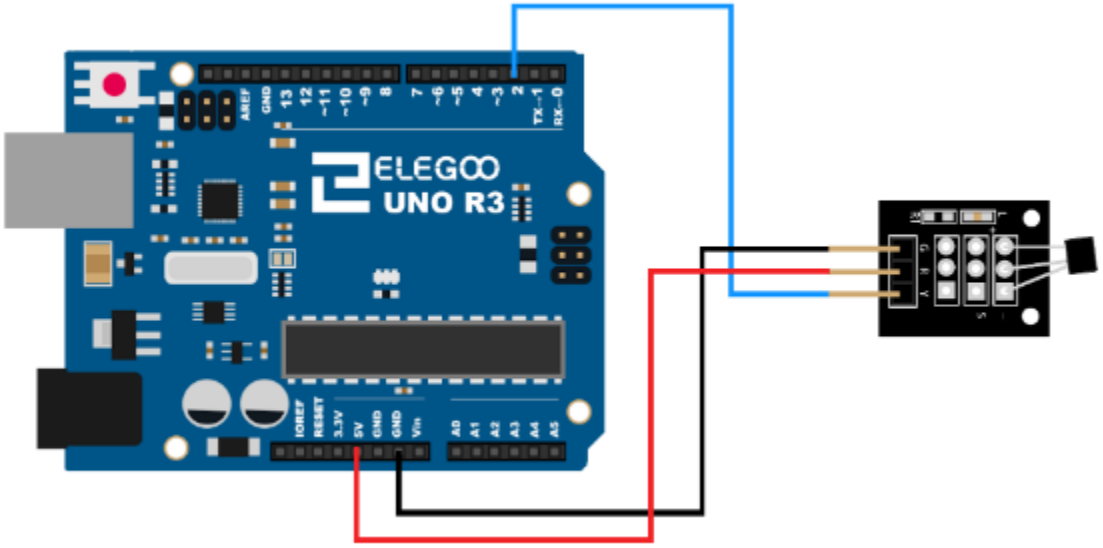
Jeder DS18B20 enthält eine eindeutige Nummer als Identifikator, so dass mehrere DS18B20-Chips an einem Bus vorhanden sein können.

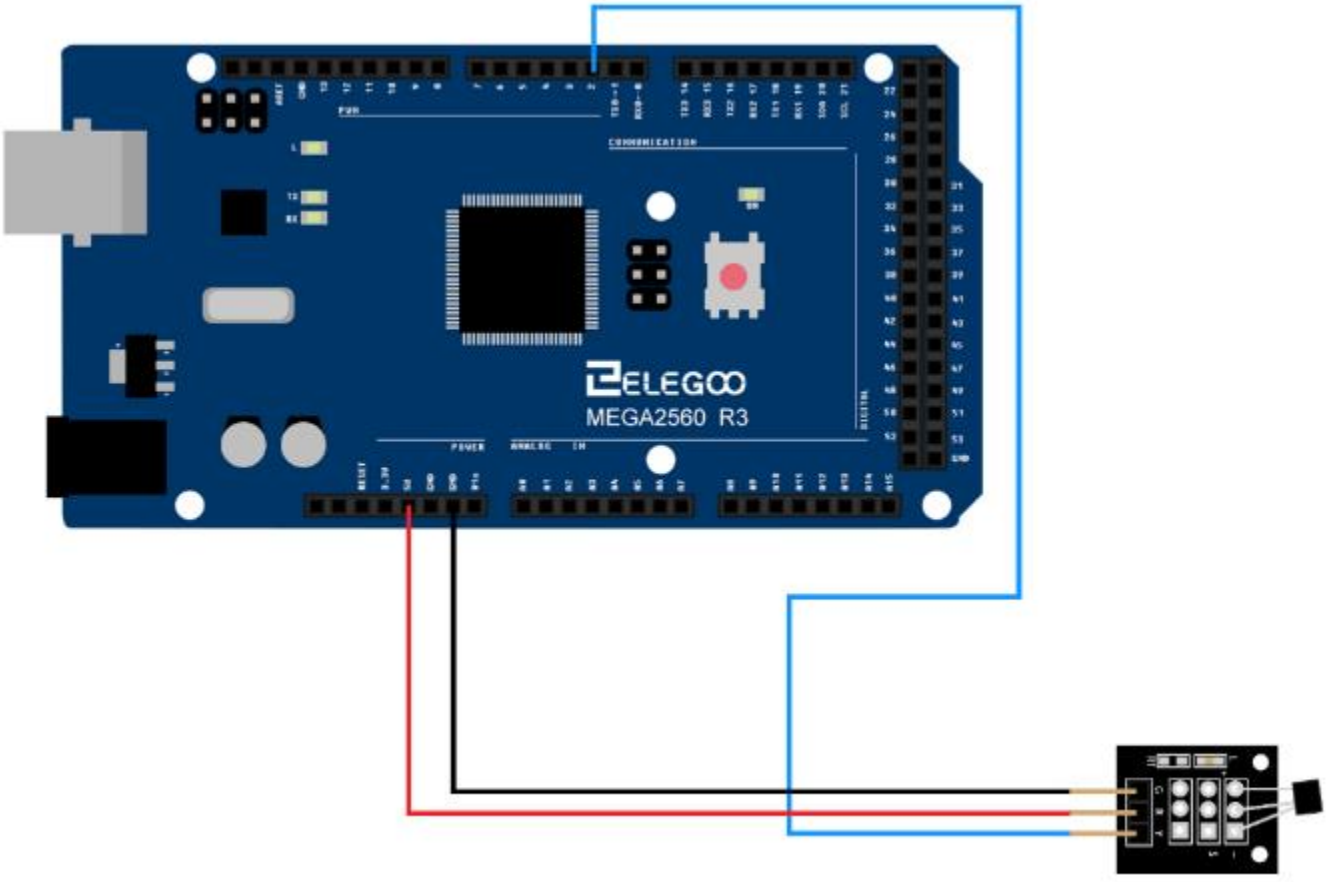
Verbindung

Schaltplan



Verdrahtungsplan





Code

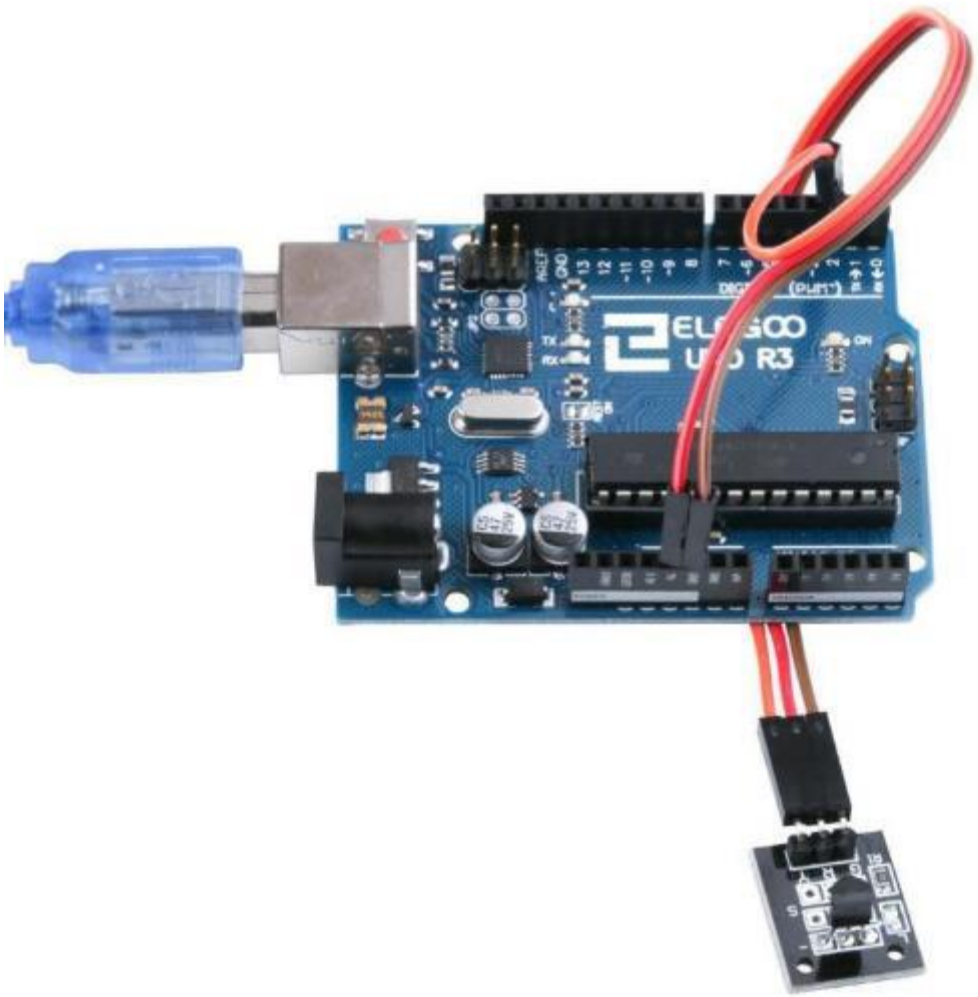
Nach der Verdrahtung öffnen Sie bitte das Programm im Codeordner - （Lesson 5 DS18B20 DIGITAL TEMPERATURE SENSOR MODULE ） and Klicken Sie auf „Hochladen“, um das Programm hochzuladen.

Siehe Lektion 2 für Details über das Hochladen von Programmen falls Fehlermeldungen auftreten.

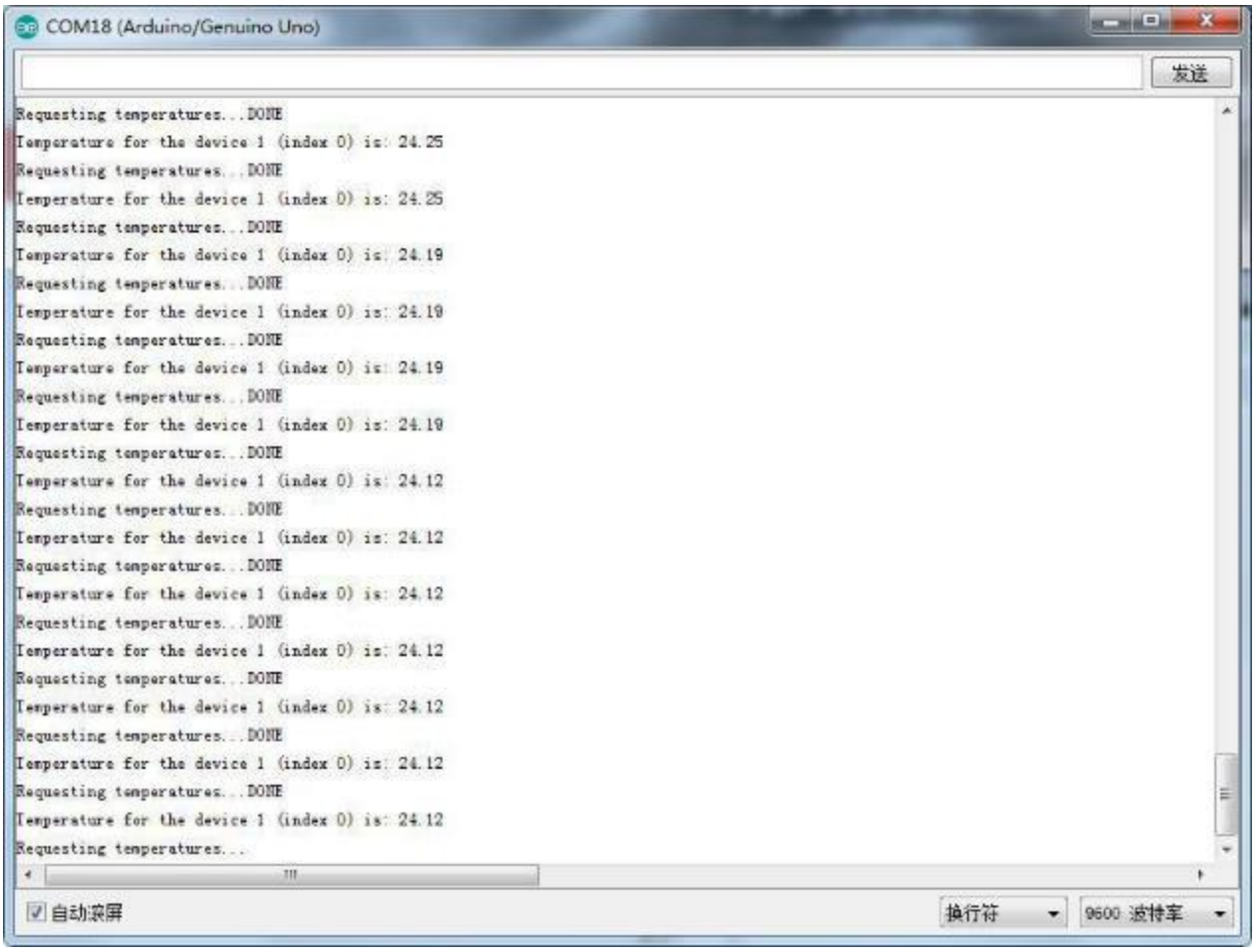
Bitte stellen Sie sicher, dass Sie die < DallasTemperature> und die <OneWire> Bibliothek installiert haben. Andernfalls wird Ihr Code nicht funktionieren.

Details zum Laden von Bibliotheksdateien finden Sie in Lektion 1

Ergebnis



Unten sehen Sie ein Beispiel, wie der serielle Monitor bei uns aussieht:



Im Folgenden finden Sie den Code für diese Lektion

```
/* Benötigte Bibliotheken*/
#include <OneWire.h>
#include <DallasTemperature.h>

/* Der Datenpin ist verbunden mit Pin 2 des Arduino*/
#define ONE_WIRE_BUS 2

/* Einrichten einer OneWire Instanz zur Kommunikation mit beliebigen OneWire Geräten (nicht nur Maxim/Dallas-Temperatur-ICs)
OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);

/*
 * Die Setup Funktion zum initialisieren des Sensors
 */
void setup (void)
{
   /* Seriellen Monitor starten*/
   Serial.begin(9600);
   Serial.println("Dallas Temperature IC Control Library Demo");

   /*Sensor initialisieren*/
   sensors.begin();
}

/*
 * Auslesen und anzeigen der Temperatur
 */
```

```
void loop(void)
{
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures(); /*Temperatur vom Sensor auslesen */
  Serial.println("DONE");
  /* Nachdem wir die Temperaturen nun haben, können wir sie hier ausgeben.
  /* Wir haben nur einen Sensor von dem wir einen Wert bekommen können */
  Serial.print("Temperature for the device 1 (index 0) is: ");
  Serial.println(sensors.getTempCByIndex(0));
}
```

Aus dem obigen Programm lernen wir, wie man die serielle Kommunikation programmiert:

(1) begin()

Beschreibung:

Stellt die Datenrate in Bits pro Sekunde (Baud) für die serielle Datenübertragung ein. Verwenden Sie für die Kommunikation mit dem Computer eine der folgenden Datenraten: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 oder 115200. Sie können aber auch andere Raten angeben, z.B. um über die Pins 0 und 1 mit einer Komponente zu kommunizieren, die eine bestimmte Baudrate benötigt. 9600 sind eine gute Standardeinstellung solange keine große Datenmenge übertragen werden soll.

Ein optionales zweites Argument konfiguriert die Daten-, Paritäts- und Stoppbits. Der Standard ist 8 Datenbits, keine Paritätsbit, ein Stoppbit.

Sowohl Baudrate als auch Partitäts- und Stoppbits müssen bei Sender und Empfänger gleich sein. Wenn beispielsweise nicht der serielle Monitor von Arduino als Empfänger dienen soll, sondern ein anderes Programm, könnte dieses bestimmte Einstellungen erfordern. Solange Sie nur mit dem seriellen Monitor von Arduino kommunizieren, können Sie die Standardeinstellungen beibehalten.

Syntax:

Serial.begin(speed)

Serial.begin(speed, config)

Nur Arduino Mega: (Er hat mehr als einen seriellen Port)

Serial1.begin(speed)

Serial2.begin(speed)

Serial3.begin(speed)

Serial1.begin(speed, config)

Serial2.begin(speed, config)

Serial3.begin(speed, config)

Parameter:

speed: Geschwindigkeit in bits pro Sekunde (baud) – Datentyp: long

config: setzt Daten-, Paritäts- und Stoppbits. Gültige Werte sind:

SERIAL_5N1

SERIAL_6N1

SERIAL_7N1

SERIAL_8N1 (Standardeinstellung)

SERIAL_5N2

SERIAL_6N2

SERIAL_7N2

SERIAL_8N2

SERIAL_5E1

SERIAL_6E1

SERIAL_7E1

SERIAL_8E1

SERIAL_5E2

SERIAL_6E2

SERIAL_7E2

SERIAL_8E2

SERIAL_5O1

SERIAL_6O1

SERIAL_7O1

SERIAL_8O1

SERIAL_5O2

SERIAL_6O2

SERIAL_7O2

SERIAL_8O2

Rückgabewert:

keiner

Beispiel:

Arduino Uno Beispiel:

```
void setup() {
  Serial.begin(9600);
}
```

Arduino Mega Beispiel:

```
void setup(){
  Serial.begin(9600);
  Serial1.begin(38400);
  Serial2.begin(19200);
  Serial3.begin(4800);
  Serial.println("Hello Computer");
  Serial1.println("Hello Serial 1");
  Serial2.println("Hello Serial 2");
  Serial3.println("Hello Serial 3");
}
```

(2) println()

Sendet Daten über den seriellen Port als menschenlesbare ASCII Zeichen gefolgt von einem Zeilenumbruch. Dieser Befehl hat die selbe Form wie Serial.print(), abgesehen vom Zeilenumbruch.

Syntax:

Serial.println(val)

Serial.println(val, format)

Parameter:

val: der zu übertragende Wert – kann ein beliebiger Datentyp sein

format: gibt die Zahlenbasis (bei ganzzahligen Datentypen) oder die Anzahl der Dezimalstellen (bei Gleitkomma-Typen) an.

Rückgabewert:

size_t (long): println() gibt die Anzahl der gesendeten Bytes zurück

Beispiel:

```
int analogValue = 0;
void setup() {
  Serial.begin(9600);
}
void loop() {
  analogValue = analogRead(0);

  // Senden in verschiedenen Formaten
  Serial.println(analogValue);          // senden als ASCII codierter Dezimalwert
  Serial.println(analogValue, DEC);   // senden als ASCII codierter Dezimalwert
  Serial.println(analogValue, HEX);   // senden als ASCII codierter Hexadezimalwert
  Serial.println(analogValue, OCT);   // senden als ASCII codierter Oktalwert
   Serial.println(analogValue, BIN);   // senden als ASCII codierter Binärwert

  delay(10);
}
```

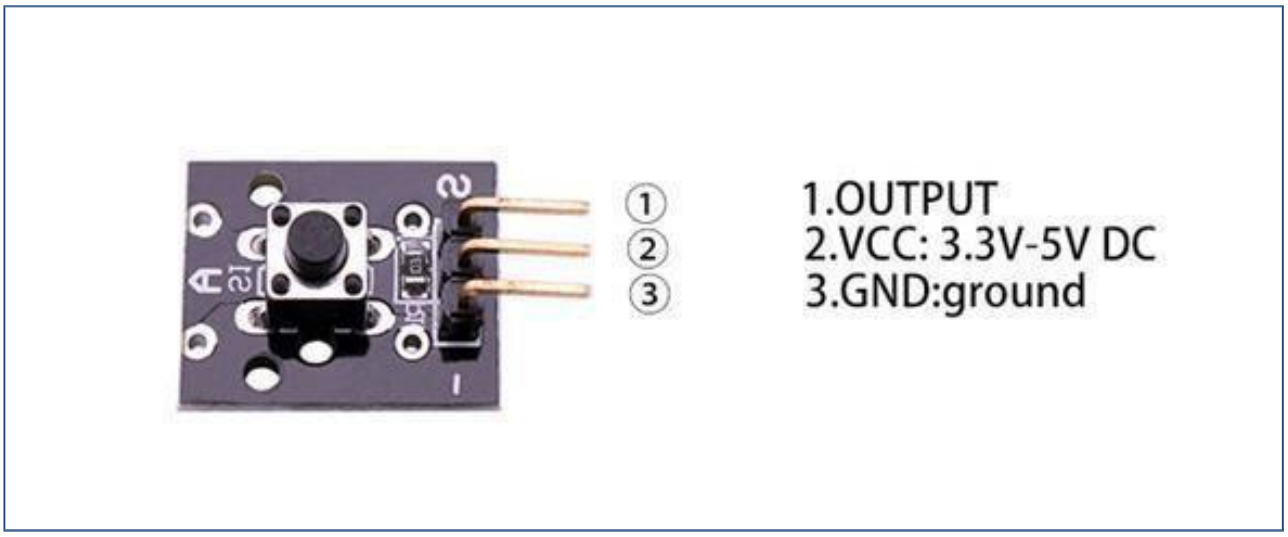
Lektion 6 Tastermodul

Überblick

In dieser Lektion lernen wir, wie man einen Taster verwendet.

Auf einer Leiterplatte montierter Taster

Ein eingebauter 10 K Ohm Widerstand ist zwischen dem mittleren Pin und dem Pin 1 vorhanden und kann als Pullup- oder Pulldown-Widerstand verwendet werden. Der Taster verbindet die beiden äußeren Pins. Wenn Sie die Versorgungsspannung an Pin 2 anschließen wirkt der Widerstand als Pullup; d.h. bei nicht gedrücktem Taster liegt die Versorgungsspannung am Ausgang. Wenn Sie die Versorgungsspannung an Pin 3 anschließen, wirkt der Widerstand als Pulldown; d.h. bei nicht gedrücktem Taster liegt Masse/Ground am Ausgang. In diesem Fall wären VCC und GND also vertauscht gegenüber dem Bild unten.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

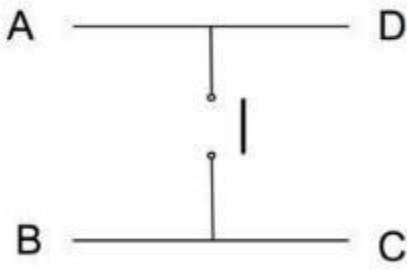
1x Tastermodul

3x F-M Drähte

Komponenteneinführung

Taster:

Schalter sind wirklich einfache Komponenten. Wenn Sie einen Knopf drücken oder einen Hebel umlegen, verbinden sie zwei Kontakte miteinander, so dass Strom durch sie fließen kann. Die kleinen Tastschalter, die in dieser Lektion verwendet werden, haben vier Anschlüsse was ein wenig verwirrend sein kann.



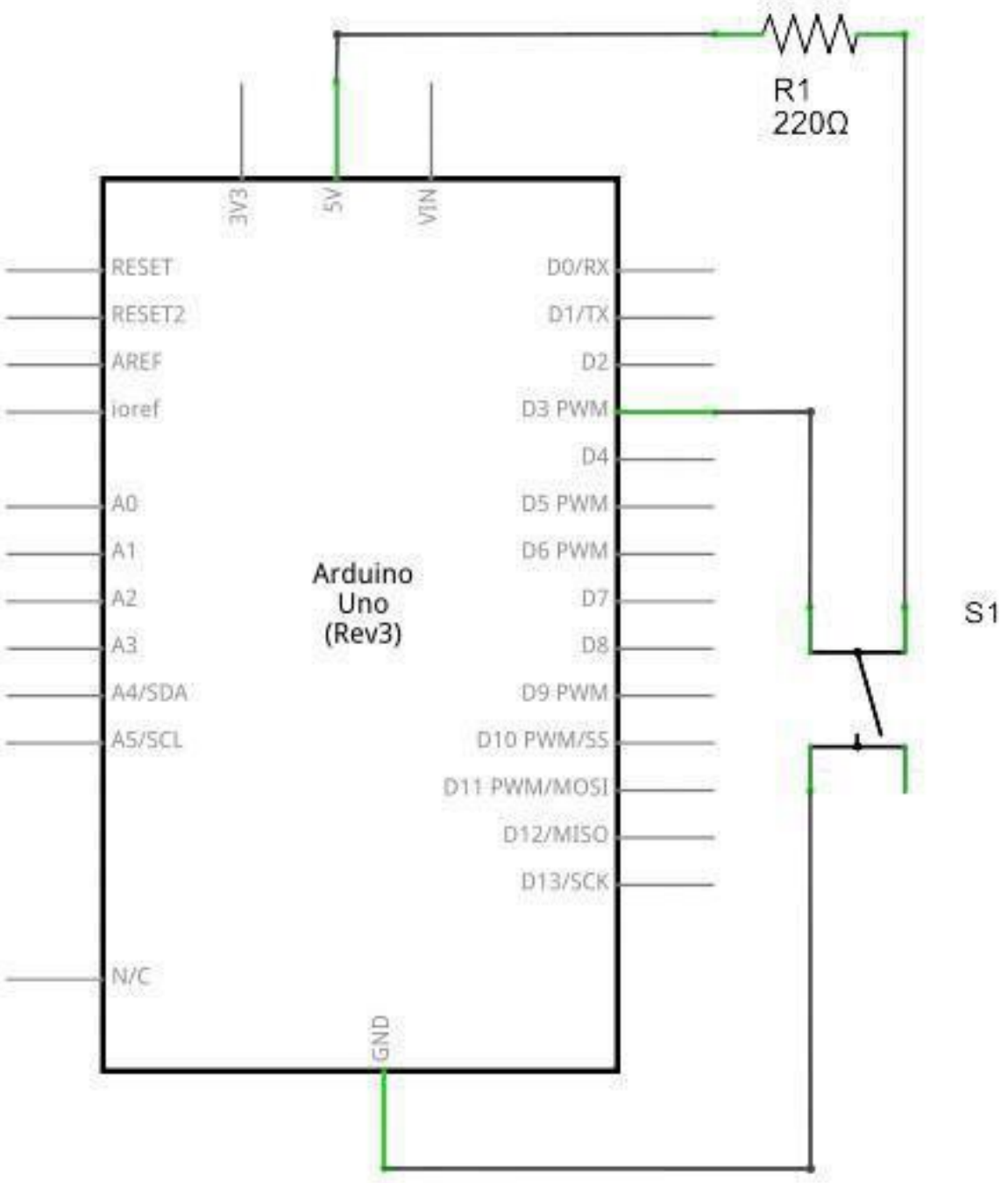
Eigentlich gibt es nur zwei elektrische Anschlüsse, da innerhalb des Schalters die Pins B und C sowie A und D miteinander verbunden sind.

Prinzip

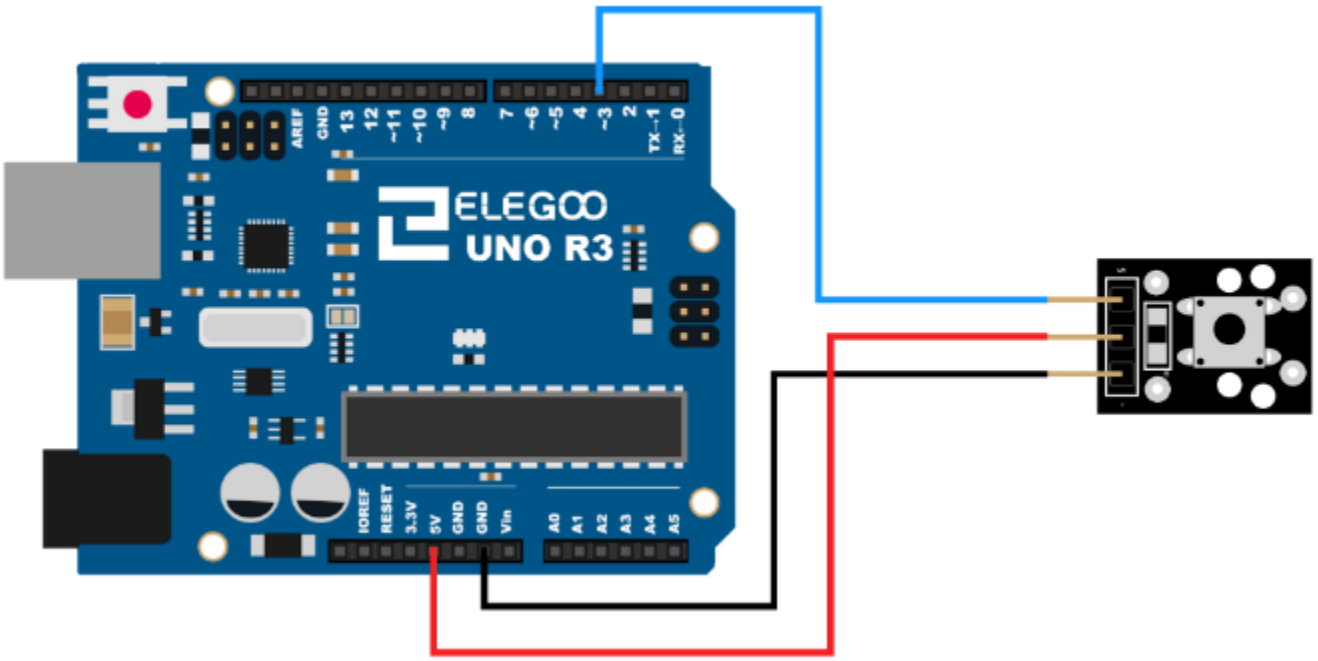
Mit dem Taster und der Arduino Pin 13 LED können wir eine einfache Schaltung aufbauen. Ein Wechsel der Schalterposition ändert den Status der LED. Genau wie bei einem Lichtschalter im Haus.

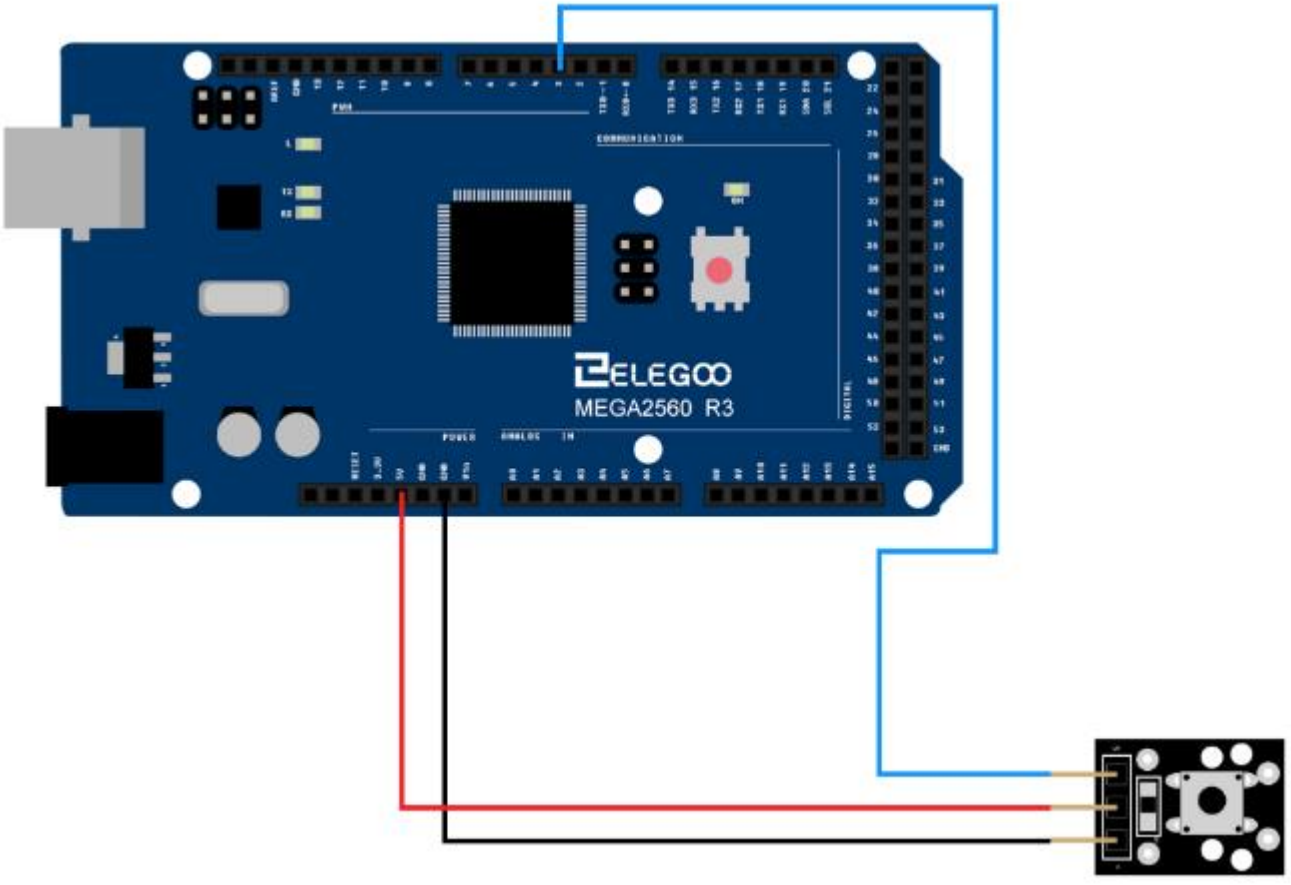
Verbindung

Schaltplan

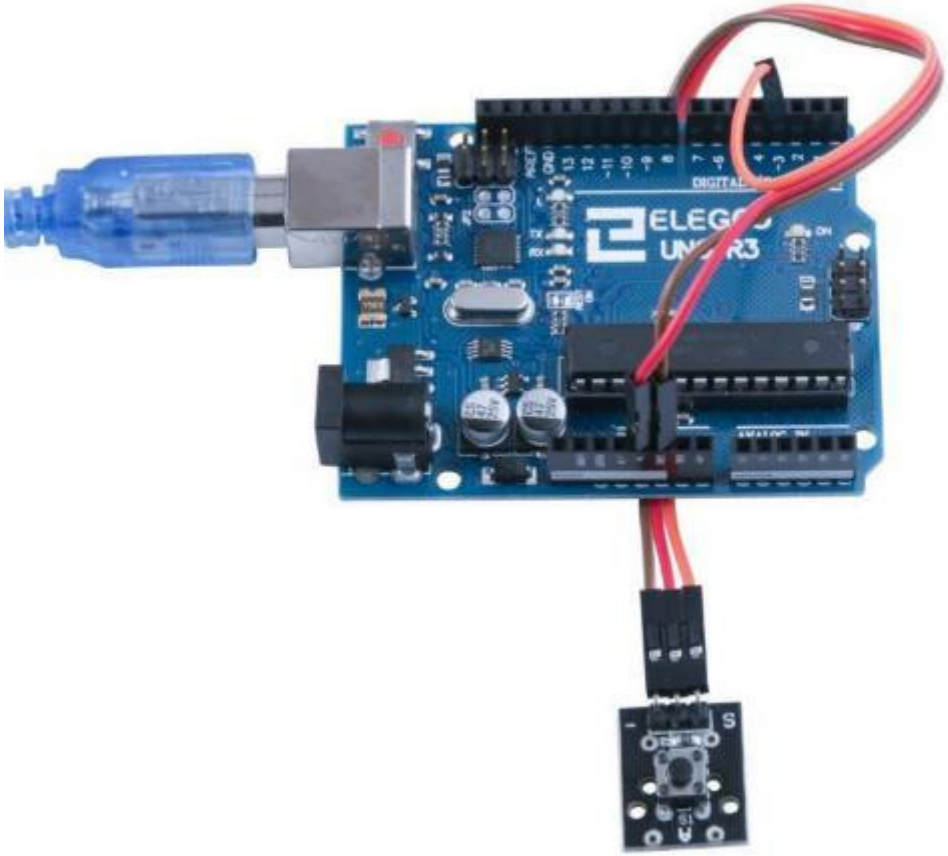


Verdrahtungsplan





Beispielbild



Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Codeordner（Lesson 6 BUTTON SWITCH MODULE）und klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen, falls es irgendwelche Fehlermeldungen gibt.

Im Folgenden finden Sie den Code:

```
int Led = 13;    //Port der eingebauten Arduino LED, die wir zur Anzeige verwenden
int Shock = 3;  //Port an dem der Taster angeschlossen wird
int val;          //Variable, die den Schalterzustand speichert
void   setup()
{
   pinMode(Led, OUTPUT);    //Definieren des LED Pins als Ausgang
   pinMode(Shock, INPUT);    //Definieren des Pins an dem der Taster angeschlossen ist als Eingang
}
void   loop()
{
    val = digitalRead(Shock);    //Abfragen des Schalterzustands
    if (val == HIGH)                 //Falls der Taster nicht gedrückt ist
    {
        digitalWrite(Led, LOW); //LED ausschalten
    }
    Else                                 //Falls der Taster gedrückt ist
    {
        digitalWrite(Led, HIGH); //LED einschalten
    }
}
```

Anmerkung:
Wenn der Taster nicht gedrückt ist, liegen am Pin 3 des Arduino über den eingebauten 10k Widerstand (im Bild oben steht fälschlicherweise 200 Ohm) die 5V/3,3V der Versorungsspannnung an. Er ist also „high“ Wenn der Taster gedrückt wird, ist der Pin 3 mit der Masse verbunden, liegt also auch 0V. Er ist also „low“.

Im obigen Programm haben wir das if/else gelernt

if/else ermöglicht eine größere Kontrolle über den Codefluss als die einfache if-Anweisung, indem mehrere Tests gruppiert werden können. Beispielsweise könnte ein Analogeingang getestet und eine Aktion durchgeführt werden, wenn der Eingang kleiner als 500 ist, und eine weitere Aktion, wenn der Eingang größer als 500 ist. Der Code würde so aussehen:

```
if (pinFiveInput < 500)
{    // action A }
else
{    // action B }
```

else kann einen weiteren if-Test durchführen, so dass mehrere, sich gegenseitig ausschließende Tests gleichzeitig durchgeführt werden können:

```
if (pinFiveInput < 500)
{    // do Thing A }
else if (pinFiveInput >= 1000)
{    // do Thing B }
else
  {    // do Thing C }
```

Sie können eine unbegrenzte Anzahl solcher Zweige haben. (Eine andere Möglichkeit verzweigte sich gegenseitig ausschließende Tests auszudrücken, ist die switch case Anweisung.)

Hinweis: Wenn Sie if/else verwenden und sicherstellen wollen, dass immer eine Standardaktion ausgeführt wird, ist es eine gute Idee Ihre Tests mit einer else-Anweisung zu beenden, die Ihr gewünschtes Standardverhalten ausführt.

Lektion 7 Verschiedene Schaltertypen

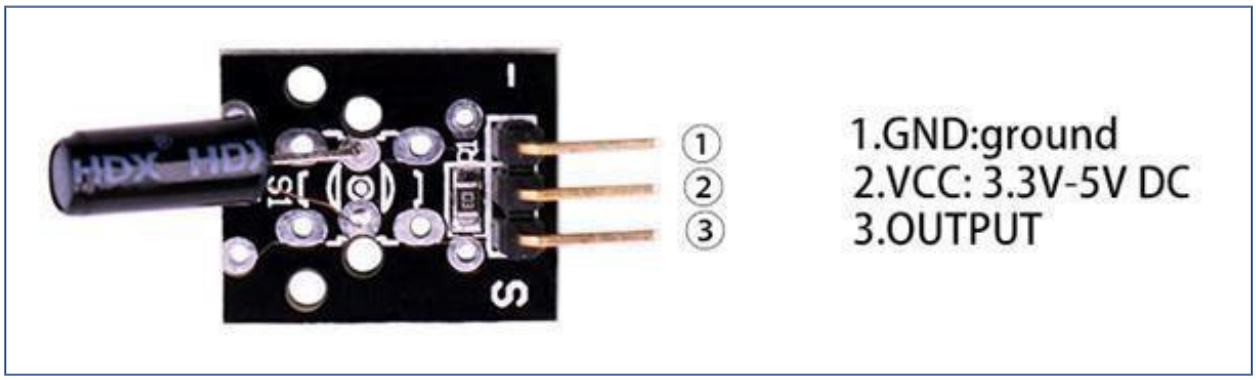
Überblick

In diesem Versuch lernen wir den Umgang mit Schaltermodulen.

Inklusive Erschütterungsschalter, Neigungsschalter und Vibrationsschalter.

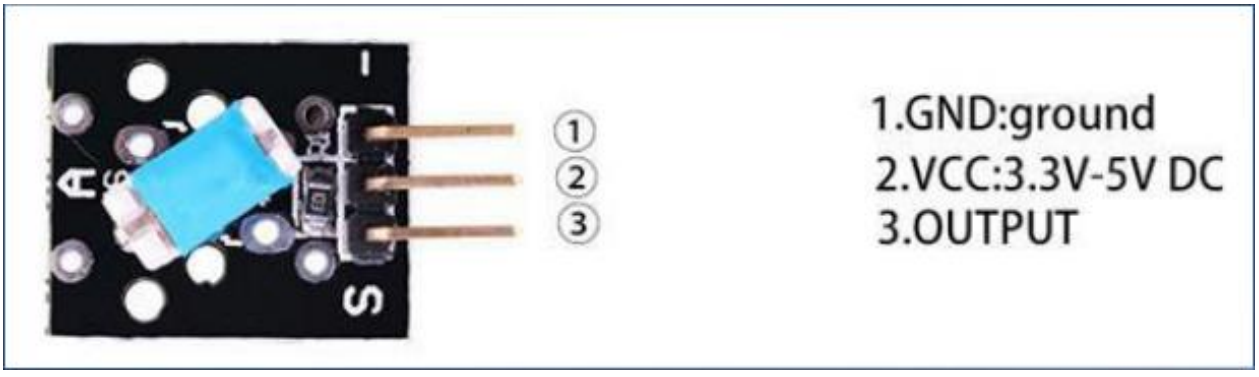
Erschütterungssensor

Ein eingebauter 10 K Ohm Widerstand ist zwischen dem mittleren Pin und Pin 3 vorhanden und kann als Pullup- oder Pulldown-Widerstand verwendet werden. Der Taster verbindet die beiden äußeren Pins.



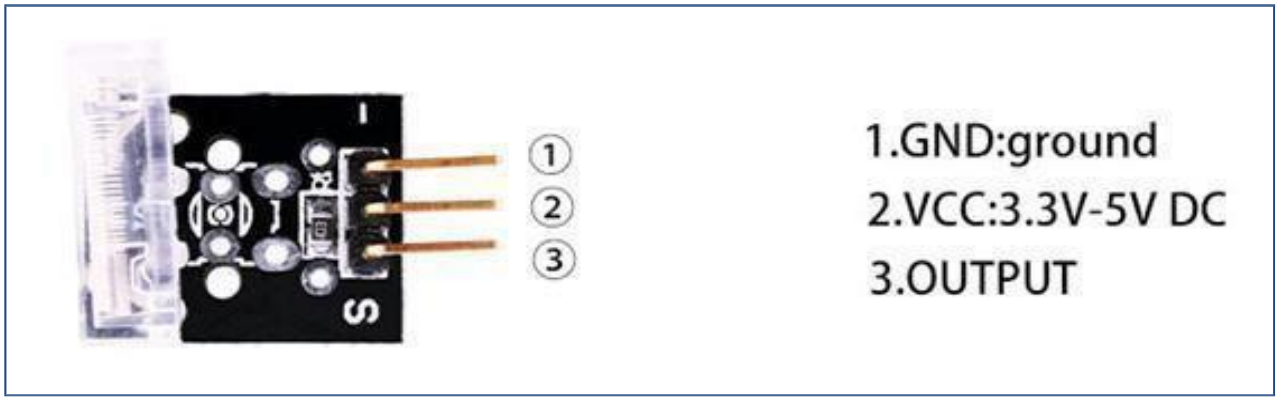
Neigungssenor mit rollender Kugel

Ein eingebauter 10 K Ohm Widerstand ist zwischen dem mittleren Pin und Pin 3 vorhanden und kann als Pullup- oder Pulldown-Widerstand verwendet werden. Der Taster verbindet die beiden äußeren Pins.



Vibrationsschaltermodul

Vibrationsschaltermodul. Ein eingebauter 10 K Ohm Widerstand ist zwischen dem mittleren Pin und Pin 3 vorhanden und kann als Pullup- oder Pulldown-Widerstand verwendet werden. Der Taster verbindet die beiden äußeren Pins.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

1 x Schockschaltermodul

1 x Neigungsschaltermodul

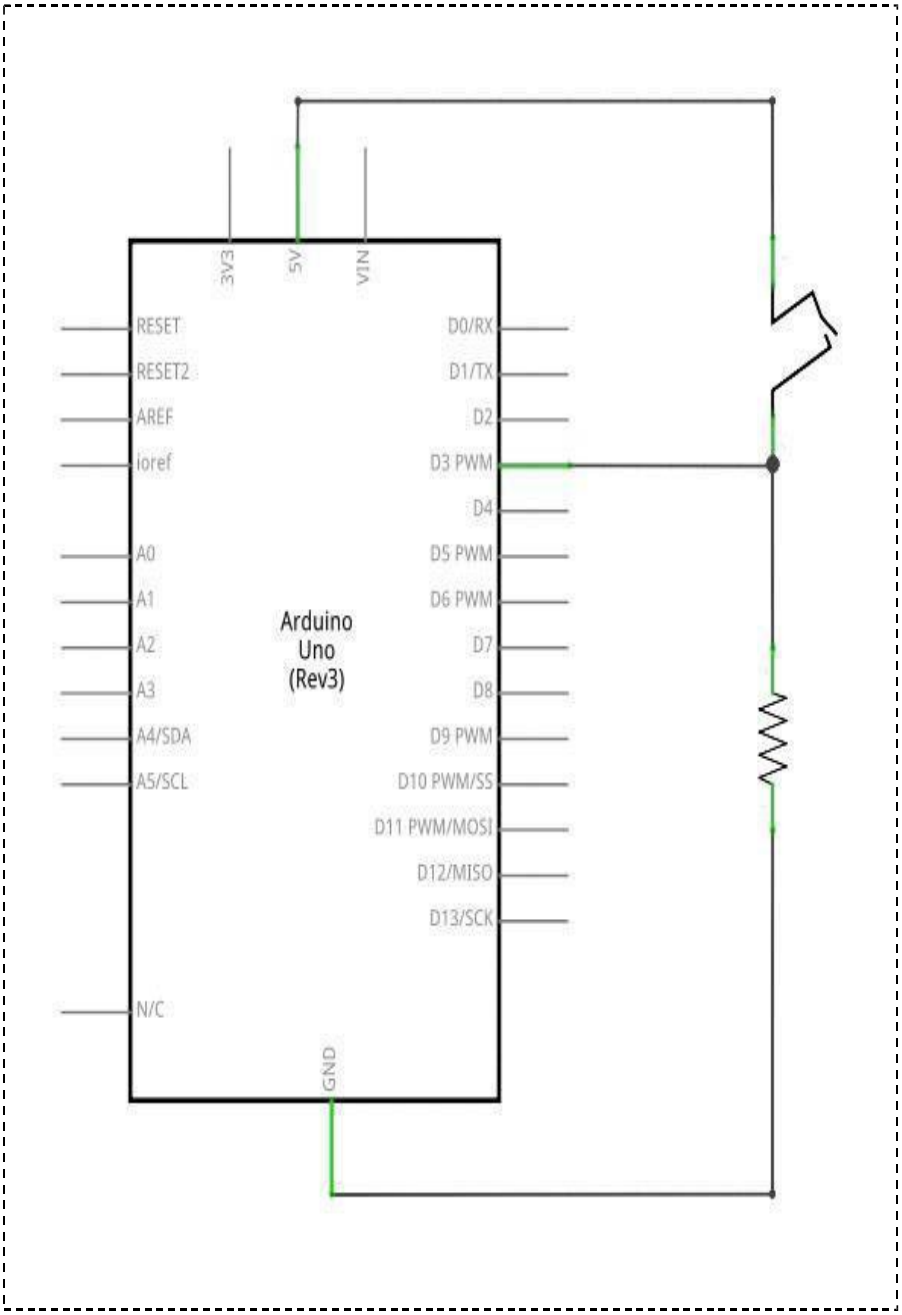
1 x Vibrationsschaltermodul

3 x F-M Drähte

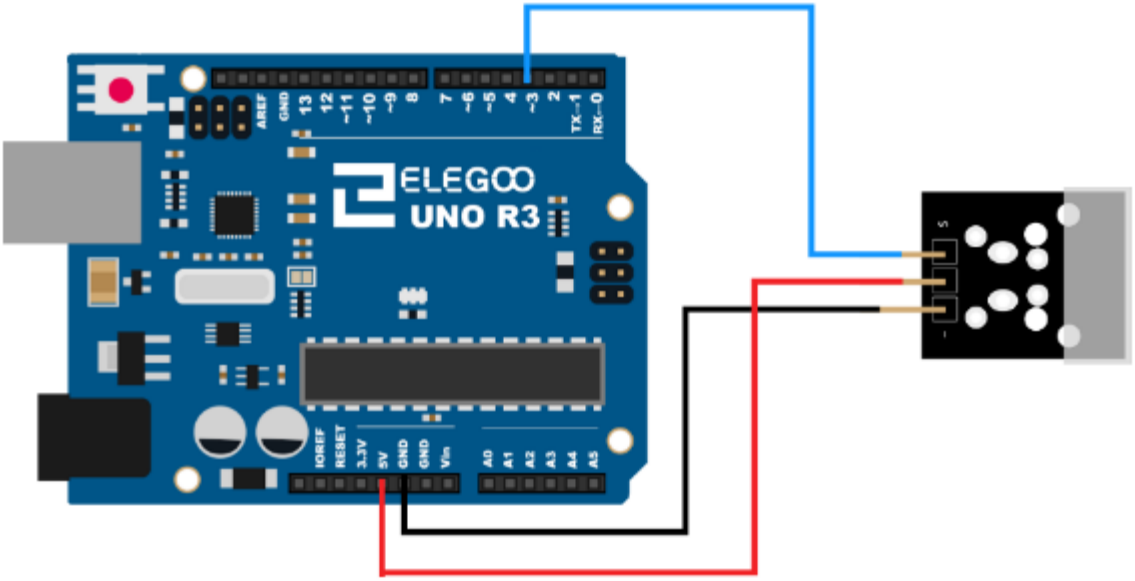
Prinzip

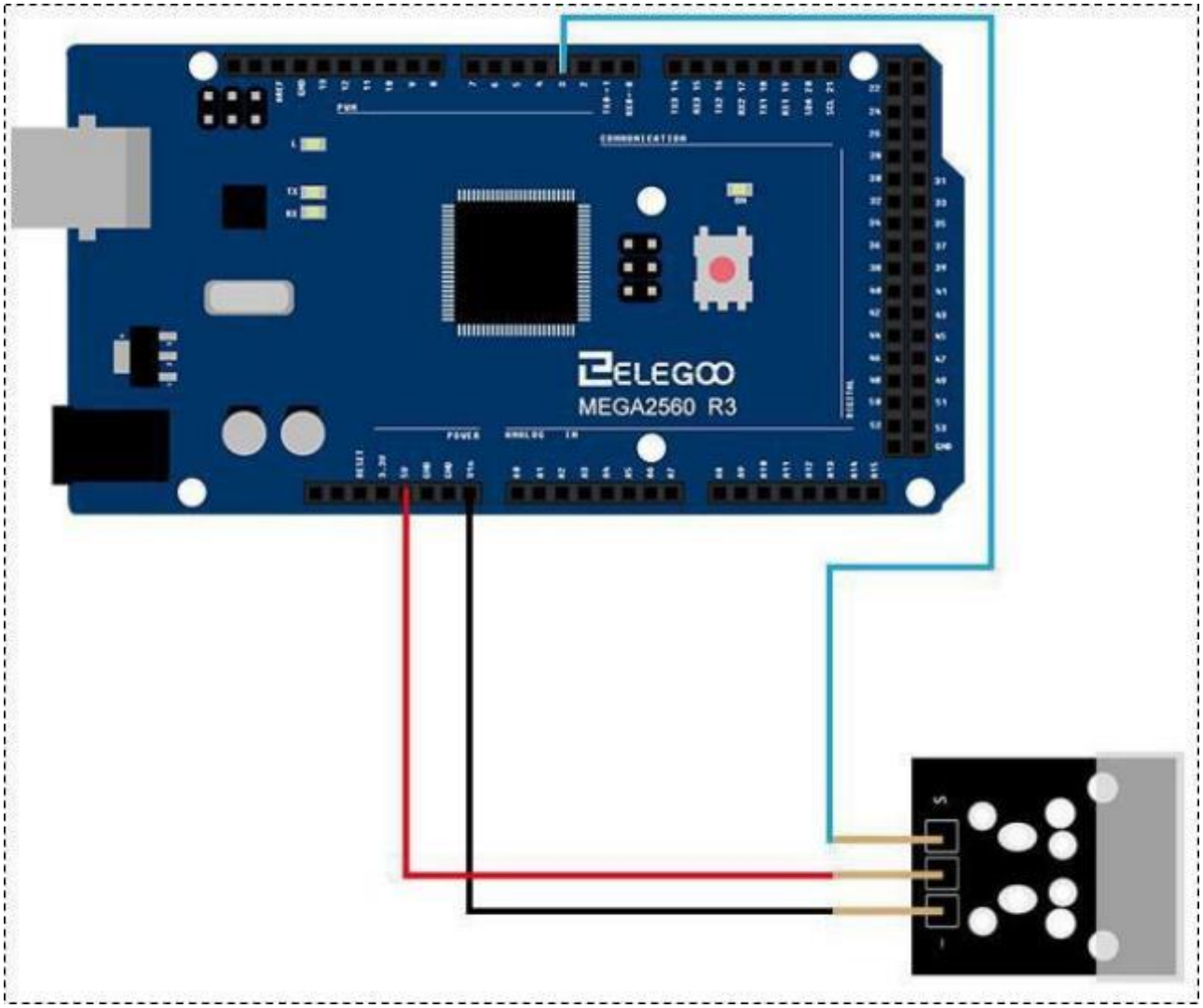
Port 13 auf dem UNO Board (oder Mega 2560 Board) verfügt über eine eingebaute LED. Um die LED bei Schalterbetätigung blinken zu lassen, müssen Sie lediglich den "S"-Pin des Switch-Moduls mit dem digitalen Port 3 des Elegoo Uno Boards (oder des Mega 2560 Boards) verbinden.

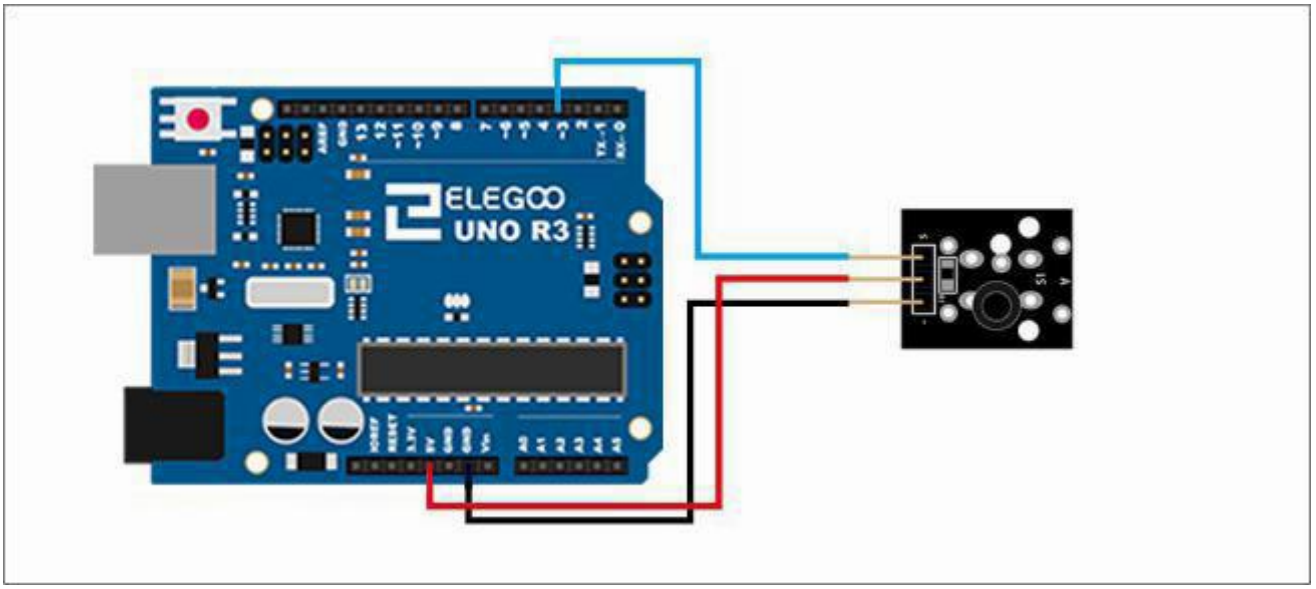
Verbindung
Schaltplan

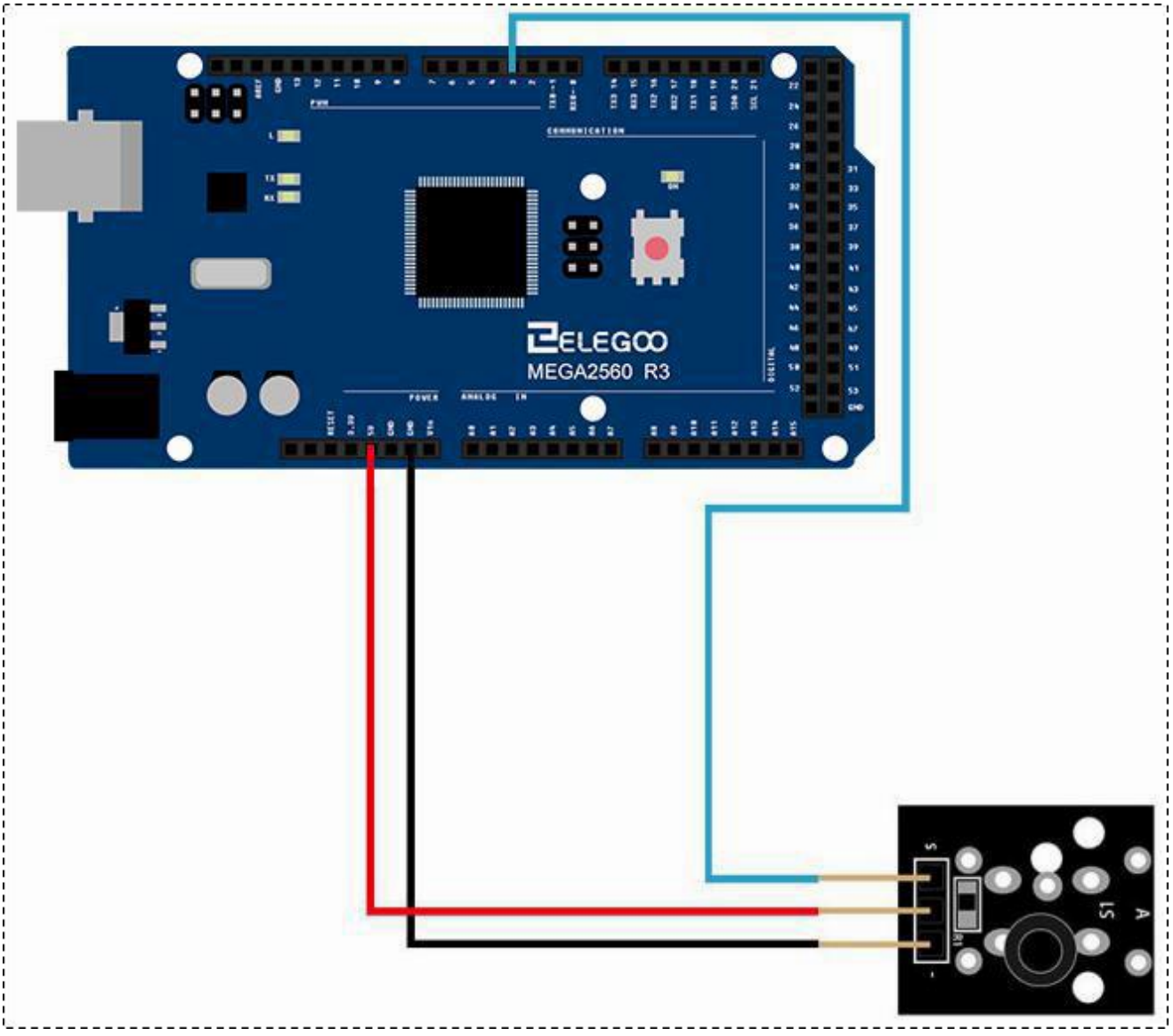


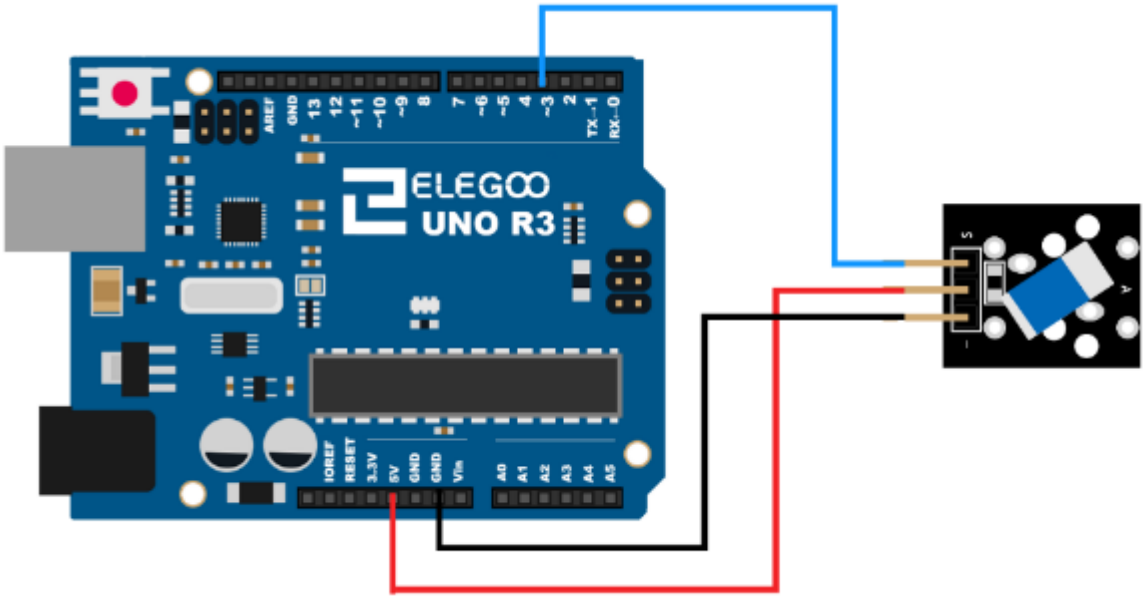
Verdrahtungsplan

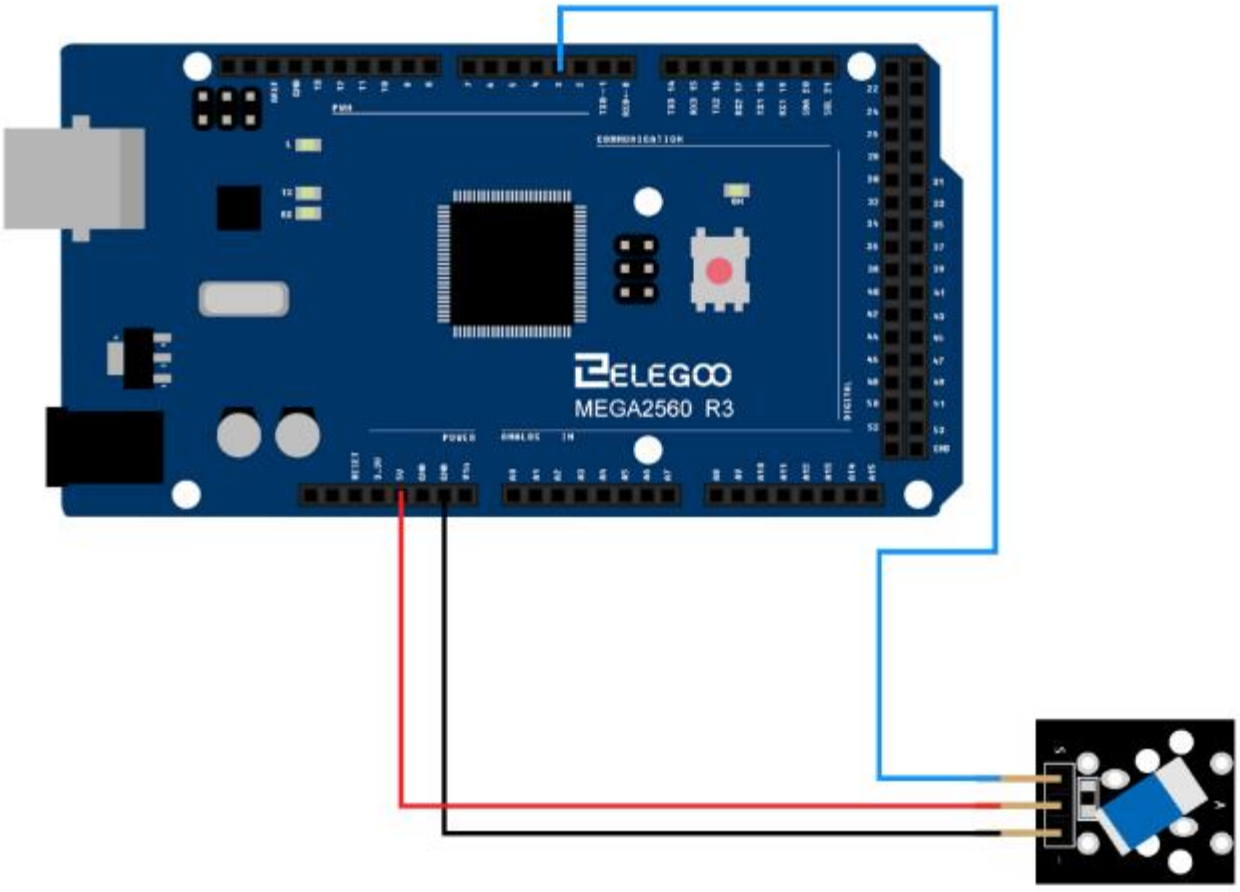





ELEGOO
UNO R3


ELEGOO
MEGA2560 R3

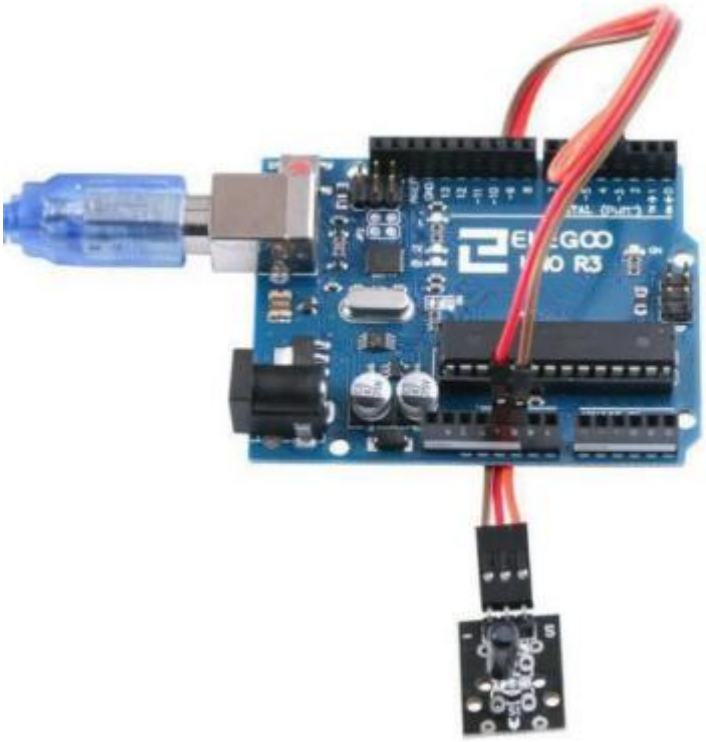

ELEGOO
UNO R3

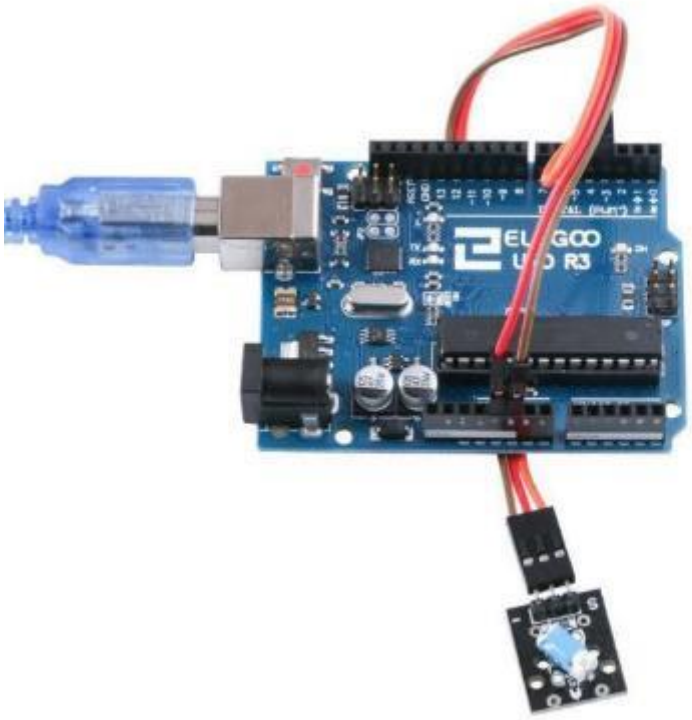

ELEGOO
MEGA2560 R3

Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Codeordner -（Lesson 7 Each type of vibration switch）and klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen, wenn es irgendwelche Fehlermeldungen gibt.

Beispielbild





Im Folgenden finden Sie den Code:

```
// Port für eingebaute LED definieren
int Led=13;
//Port definieren, an dem der Schalter angeschlossen ist
int buttonpin=3;
//Variable definieren, in der der Schalterzustand gespeichert wird
int    val;
void    setup()
{
//LED Pin als Ausgang festlegen
pinMode(Led,OUTPUT);

//Pin an dem der Schalter angeschlossen wird als Eingang festlegen
pinMode(buttonpin,INPUT);}
void    loop()
{
//Schalterzustand einlesen
val=digitalRead(buttonpin);
//Falls der Schalter geschlossen ist…
if(val==HIGH)
{
//LED einschalten
digitalWrite(Led,HIGH);
}
else //Falls der Schalter offen ist…
{
//LED ausschalten
digitalWrite(Led,LOW);
}
}
```

Lektion 8 Infrarot (IR) Sender und Empfänger

Überblick

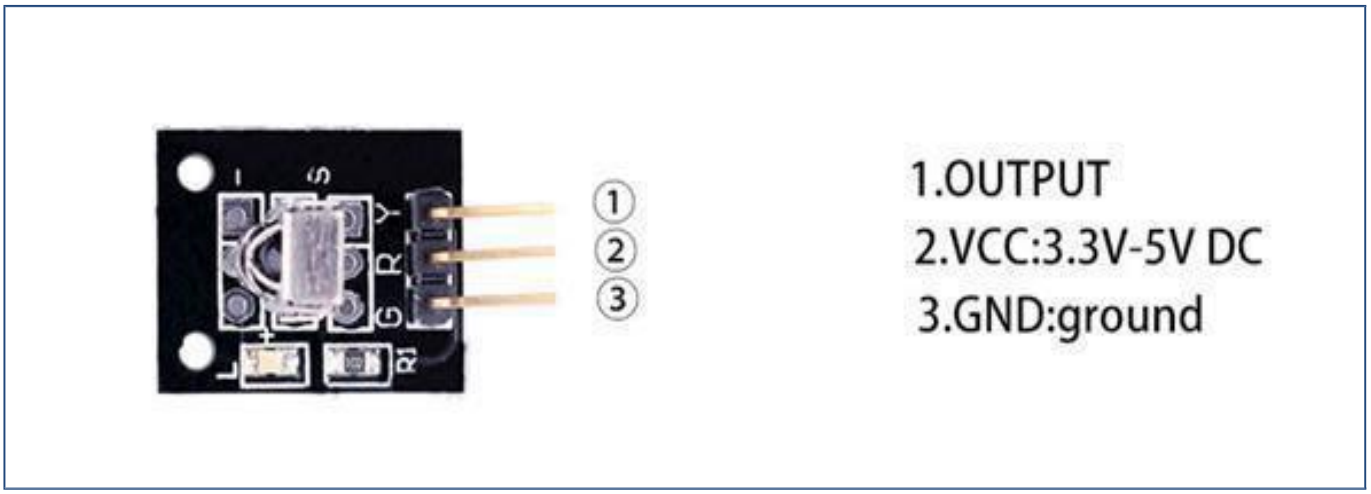
In diesem Versuch lernen wir, wie man ein Infrarotsender und Empfängermodul verwendet.

Solche Sender und Empfänger begegnen uns im Alltag sehr häufig in Form von Fernbedienungen.

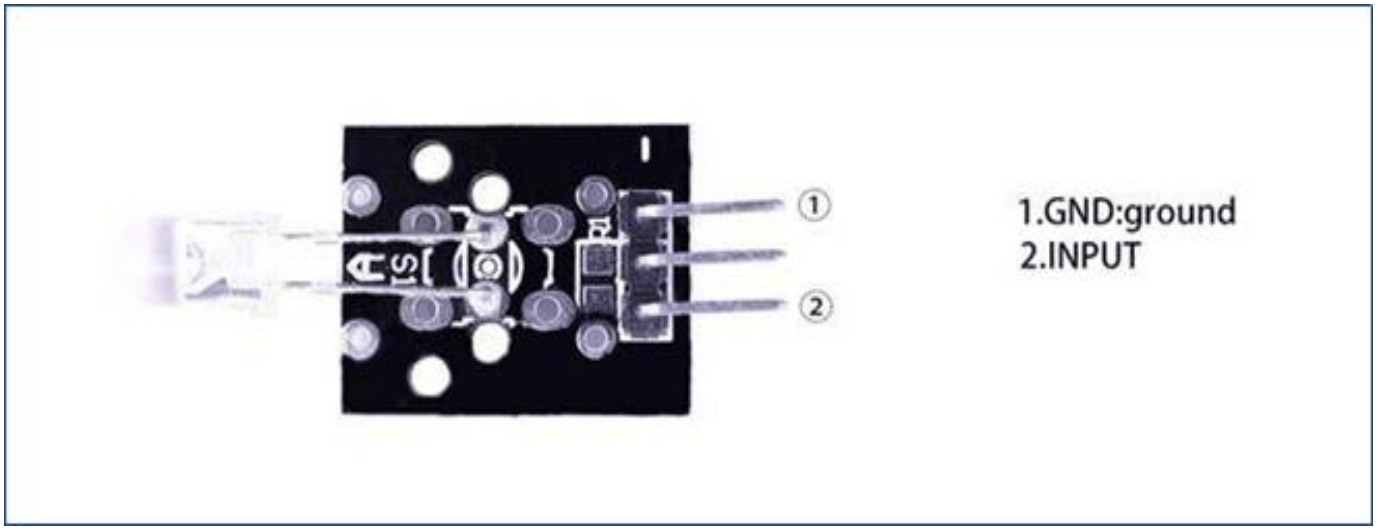
IR Empfänger

Infrarotsensor vom Typ 1838 für 38KHz Infrarotsignale.

- Spannungsversorgung: 2.7 to 5.5 V
- Frequenz: 37.9 KHz
- Reichweite: 18m (typisch)
- Empfangswinkel: 90°



IR Sender



Benötigte Komponenten:

2x Elegoo Uno R3

2x USB Kabel

1x IR Empfängermodul

1x IR Sendermodul

5 x F-M Drähte

Komponenteneinführung

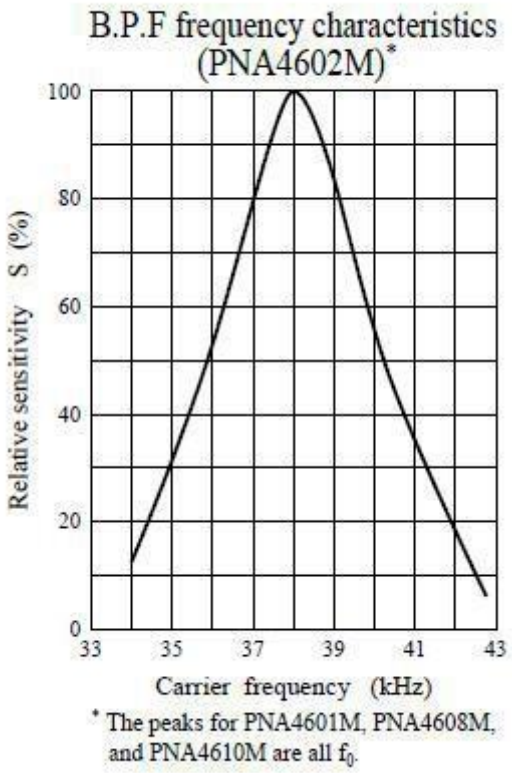
IR EMPFANGSSENSOR:

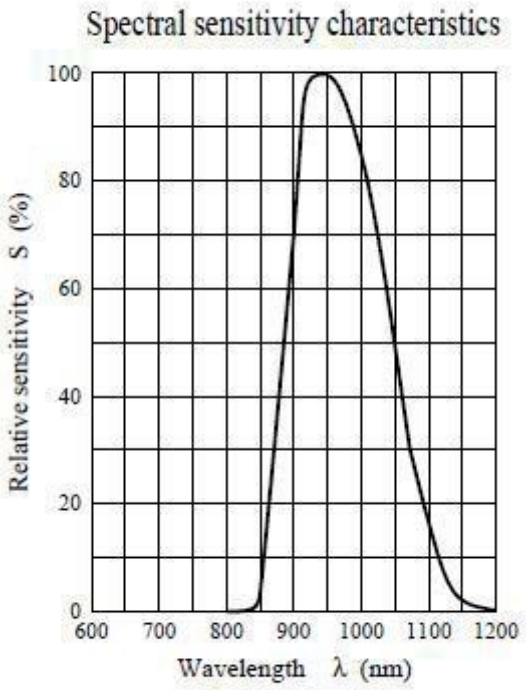
IR Detektoren sind kleine Mikrochips mit einem Lichtsensor, die auf Infrarotlicht abgestimmt sind. Sie werden sehr oft zur Erkennung von Fernbedienungssignalen verwendet; jeder Fernseher und DVD-Player hat einen davon auf der Vorderseite, um das IR-Signal der Fernbedienung zu empfangen. Im Inneren der Fernbedienung befindet sich eine passende IR LED, die IR-Impulse aussendet, um dem Fernseher Befehle zum Ein-, Ausschalten oder Wechseln der Kanäle zu geben. IR Licht ist für das menschliche Auge nicht sichtbar, weshalb es ein wenig mehr Arbeit erfordert, um eine Schaltung zu testen. Es gibt ein paar Unterschiede zwischen diesen Detektoren und einer Fotozelle:

IR-Detektoren enthalten einen Filter, der nur Infrarotlicht durchlässt und sichtbares Licht ausfiltert. Im Gegensatz dazu sind Fotozellen gut im Erkennen von sichtbarem Licht, aber nicht gut im Erkennen von IR Licht.

- IR-Detektoren haben einen Demodulator im Inneren, der nach moduliertem Infrarotlicht bei 38 KHz sucht und nur diese auswertet. Fotozellen haben keinen Demodulator und können jede Frequenz (oder auch durchgängiges Licht) innerhalb der Ansprechgeschwindigkeit der Fotozelle (ca. 1KHz) erkennen.
- IR-Detektoren haben einen digitalen Ausgang - entweder erkennen sie ein 38KHz IR Signal und der Ausgang ist low (0V) oder sie erkennen keins und geben high (5V) aus. Fotozellen wirken wie Widerstände, deren Widerstand sich je nach Lichteinfall ändert.

Was man messen kann



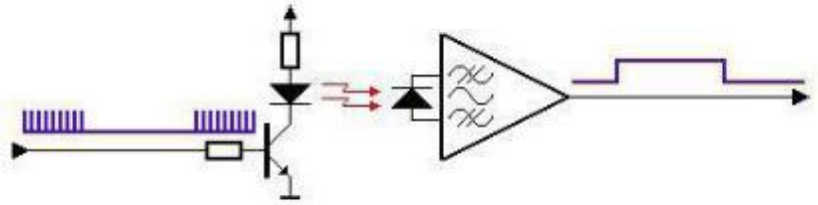


Wie Sie diesen Datenblattgrafiken entnehmen können, liegt die Frequenz, die am besten erkannt wird bei 38 KHz und die am besten erkannte Lichtwellenlänge liegt bei 940 nm. Man kann den Empfänger von ca. 35 KHz bis 41 KHz verwenden, aber die Empfindlichkeit sinkt, so dass Signale aus der Ferne nicht mehr sicher erkannt werden. Ebenso können Sie LEDs mit einer Wellenlänge von 850 bis 1100 nm verwenden, aber sie funktionieren nicht so gut wie 900 nm bis 1000 nm. Also stellen Sie sicher, dass Sie passende Sende-LEDs benutzen. Überprüfen Sie das Datenblatt Ihrer IR LED auf die Wellenlänge. Versuchen Sie eine LED mit 940 nm zu erhalten.

Prinzip

Lassen Sie uns zunächst einen Blick auf die Struktur des Infrarotempfängers werfen. Es gibt zwei wichtige Elemente im Inneren, ein IC (integrierter Schaltkreis) und und eine Photodiode. Die Hauptfunktionen des ICs sind Filterung, Dekodierung und Verstärkung. Die Photodiode ist dazu da, das Infrarotsignal zu empfangen.

Nachfolgend finden Sie einen Prinzipschaltplan (Auf der linken Seite der Sender mit IR LED und rechts der Empfänger mit einer Photodiode, die an einem IC hängt):



Die Infrarot LED sendet ein mit 38 kHz moduliertes Signal und der Infrarot Empfänger empfängt, dekodiert, filtert und verstärkt das Signal.

Verbindung

Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Code-Ordner （Lesson 8 IR Receiver and IR Emission） and klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen falls Fehlermeldungen auftreten. Bitte stellen Sie sicher, dass Sie die Bibliothek < IRremote> installiert haben. Details zum Laden von Bibliotheksdateien finden Sie in Lektion 1.

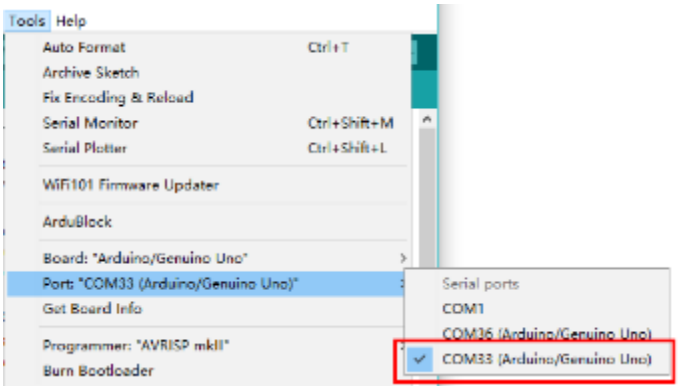
Nehmen Sie zunächst ein UNO-Board und schließen Sie das Infrarotsendemodul gemäß dem beigefügten Bild an und laden Sie das Programm“IR_Emission“

IR_Emission

Als nächstes nehmen Sie eine weiteres UNO-Board und schließen Sie das Infrarot-Empfängermodul wie unten gezeigt an und laden Sie das Programm „IR_Receiver“.

IR_Receiver

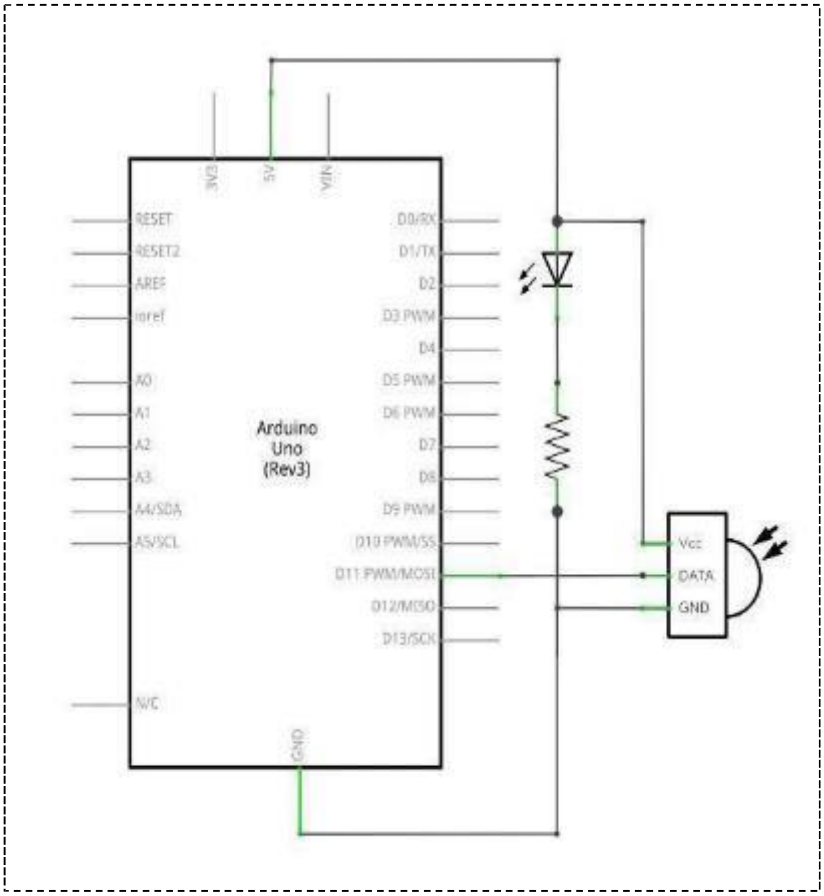
Hinweis: Stellen Sie vor dem Herunterladen sicher, dass Sie den Download Port auswählen. Es gibt zwei serielle Schnittstellen, wenn Sie zwei UNO-Karten anschließen. Wir müssen den jeweils passenden auswählen, wie im Bild gezeigt.

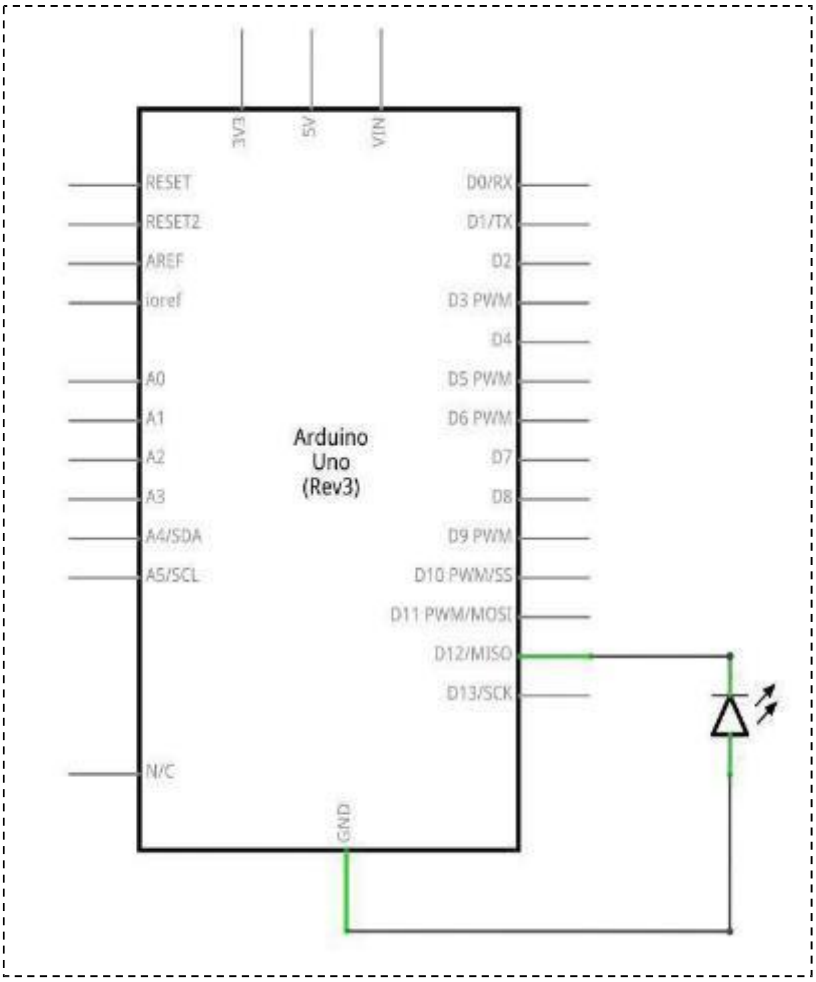


Nach erfolgreichem Hochladen richten Sie das Infrarotsendemodul auf das Infrarotempfangsmodul aus. Beobachten Sie den UNO des Empfängers, wenn Sie die L-LED blinken sehen können heißt es, dass der Versuch erfolgreich ist.

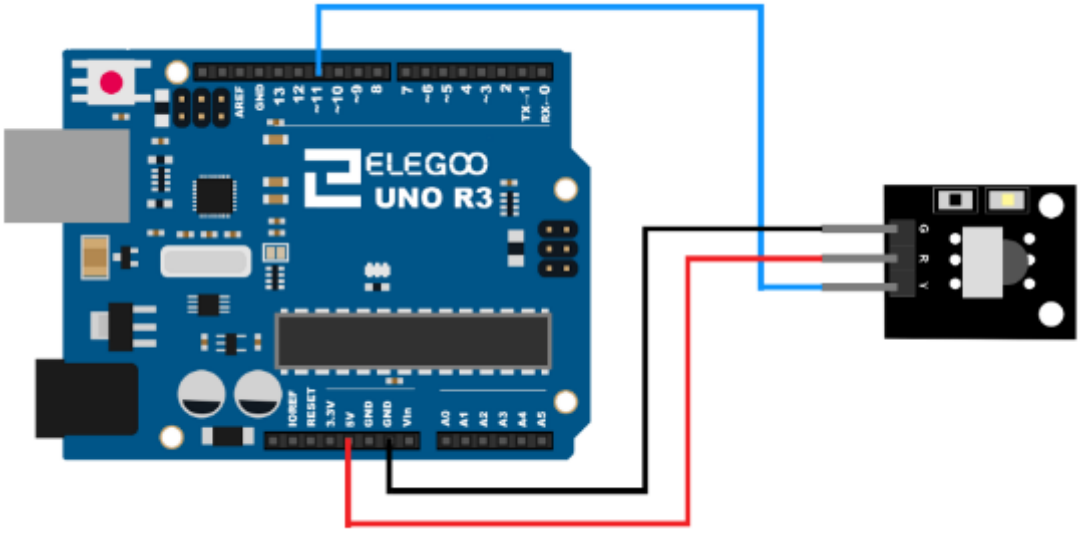
Das Prinzip des Versuchs besteht darin, dass das Infrarotsendemodul ein bestimmtes codiertes Signal sendet und nach dem Empfang des Infrarotempfängermoduls die L-LED zum Blinken gebracht wird.

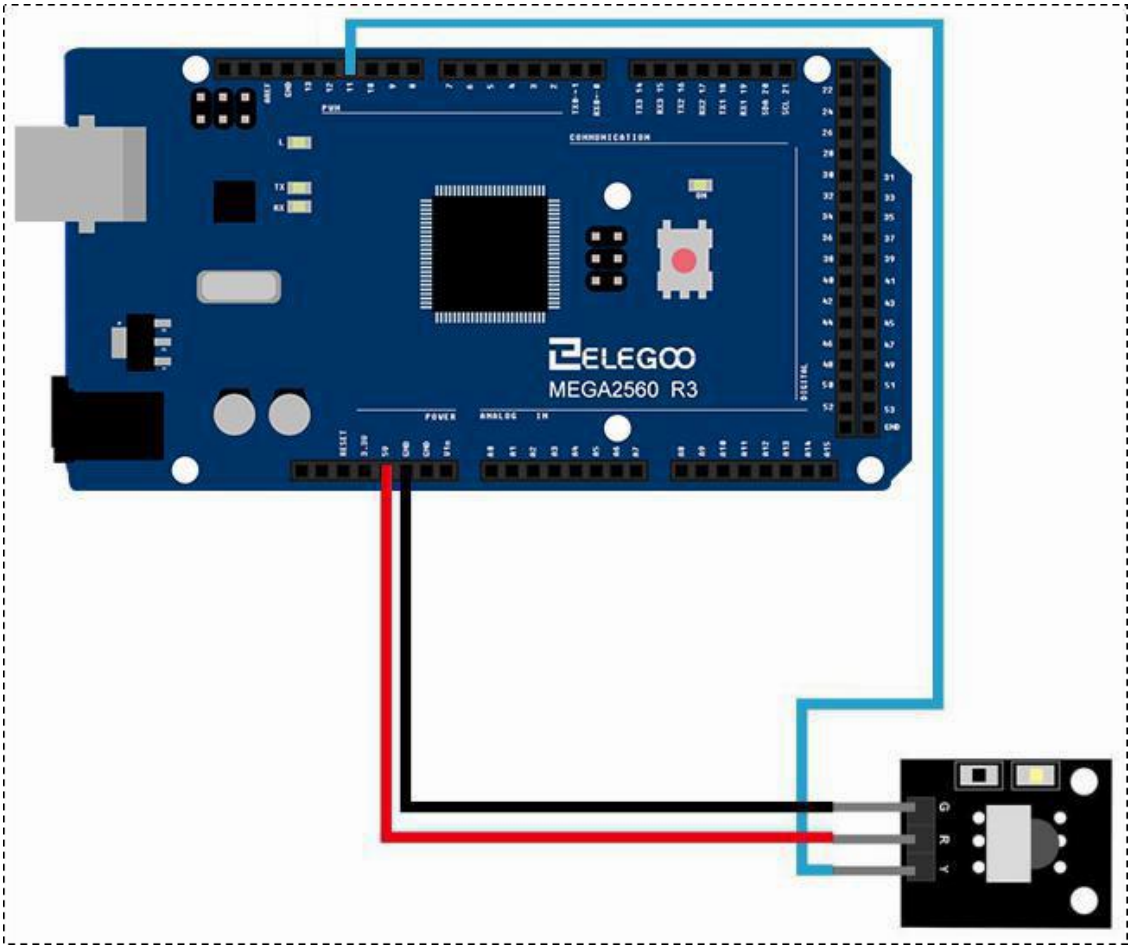
Schaltplan

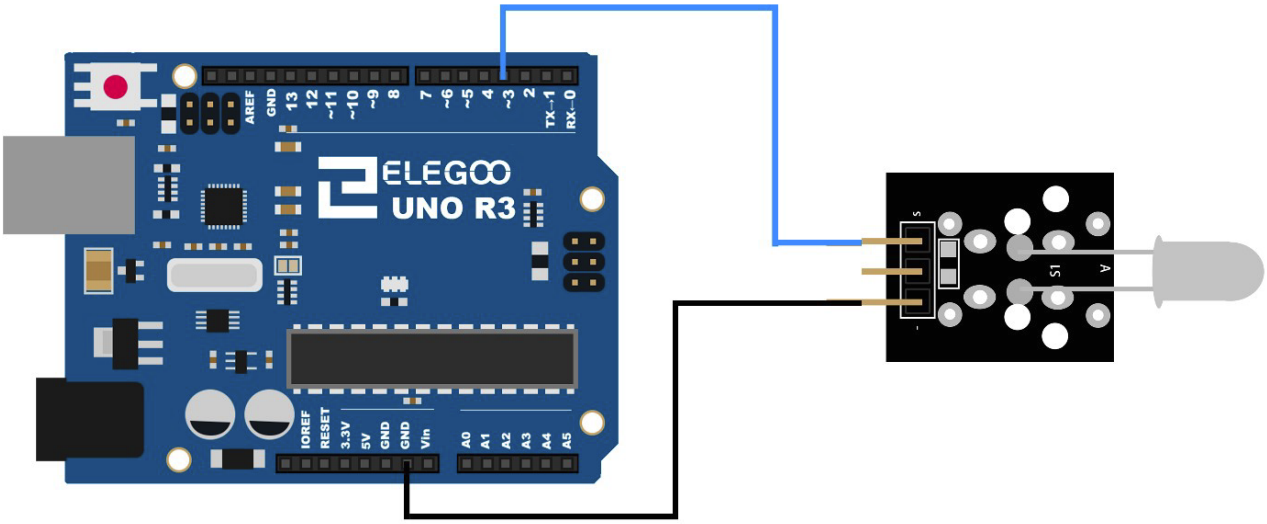

3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
GND
Arduino
Uno
(Rev3)
Vcc
DATA
GND

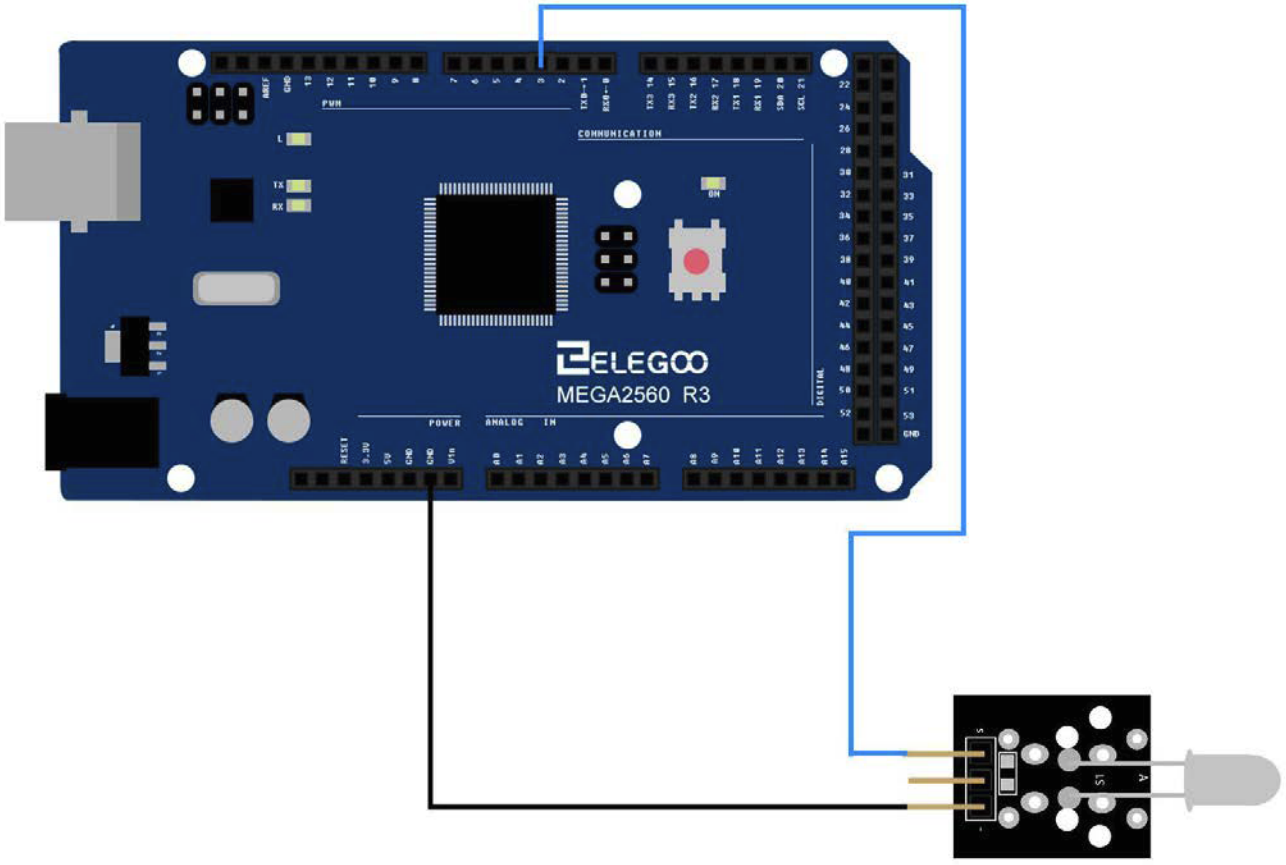

3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
GND
Arduino
Uno
(Rev3)

Verdrahtungsplan

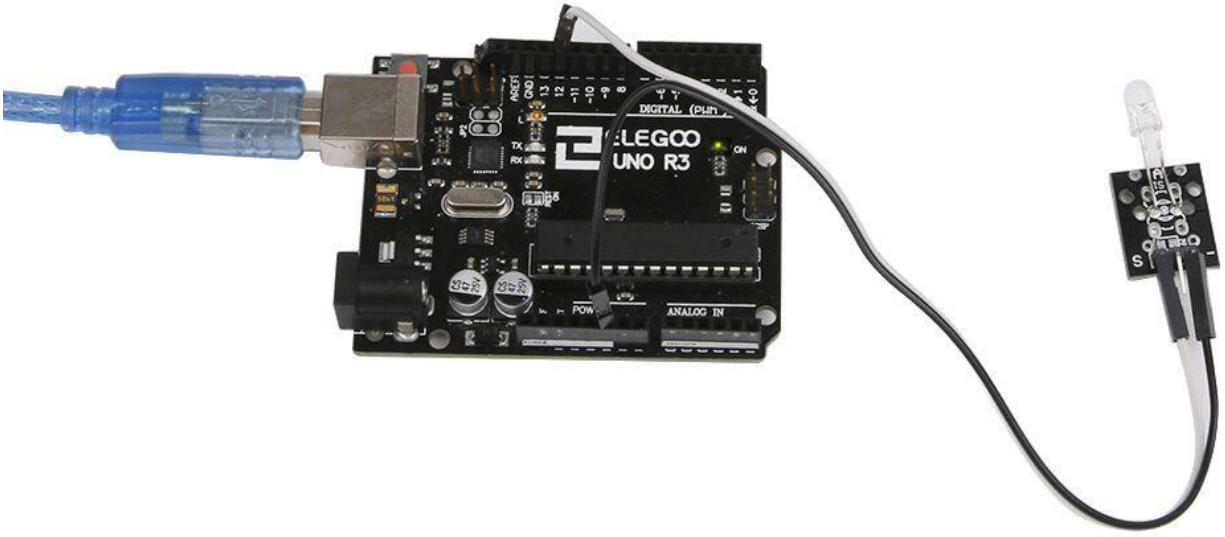





ELEGOO
UNO R3


ELEGOO
MEGA2560 R3

Beispielbild



Fehlersuche

Falls die L-LED nicht blinkt, handelt es sich meist um einen Verdrahtungsfehler oder die Module sind nicht auf einander ausgerichtet.

Aber wenn Sie sicher sind, dass die Verbindungen und die Programme korrekt sind, gibt es zwei Möglichkeiten, um festzustellen, ob das IR-Empfängermodul und das IR-Sendemodul beschädigt sind.

1. Empfängermodul

Richten Sie ihre TV Fernbedienung auf den Empfänger und drücken Sie beliebige Tasten auf der Fernbedienung. Falls die L-LED blinkt funktioniert das Modul andernfalls ist es defekt.

2. Sendemodul

Öffnen Sie die Kamera Ihres Handys und halten Sie sie vor die LED des IR-Senders. Wenn Sie ein violettes Licht sehen können, funktioniert ihr Sendemodul.

Sollten die Module defekt sein, setzen Sie sich bitte mit uns in Verbindung und wir senden Ihnen Ersatz zu.

Im Folgenden finden Sie den Code für Sender und Empfänger

(1) Sender

```
/* Die IR LED muss mit dem Arduino PWM (Pulsweitenmodulation) pin 3 verbunden werden.*/
/* <IRremote.h> Bibliothek einbinden*/
 #include <IRremote.h>
 IRsend irsend;
 void setup()
 {}
 void loop()
 {
  irsend.sendRC5(0x0, 8); /* sende 0x0 (8 bits)*/
  delay(200);
  irsend.sendRC5(0x1, 8); /* sende 0x1 (8 bits)*/
  delay(200);
  }
```

(2) Empfänger:

```
/* <IRremote.h> Bibliothek einbinden*/
#include <IRremote.h>
   /* Empfangspin ist Pin 11 */
 #define RECV_PIN    11
   /* Pin der eingebauten LED ist Pin 13 */
 #define LED             13
 IRrecv irrecv(RECV_PIN);
 decode_results results;
 void setup() {
/* LED als Ausgang setzen */
     pinMode(LED, OUTPUT);
     Serial.begin(9600);
/* Empfänger starten */
     irrecv.enableIRIn();
 }
   void loop() {
   if (irrecv.decode(&results))    {
        int state;
           if ( results.value == 1 )     {       /* Falls 0x01 empfangen wird → eingebaute LED einschalten */
              state = HIGH;
           }
 else{
             State = LOW;  /* Andernfalls eingebaute LED ausschalten */
           }
   digitalWrite( LED, state );
  Serial.println(results.value);
  irrecv.resume();
    }
 }
```

Lektion 9 Aktiver Summer

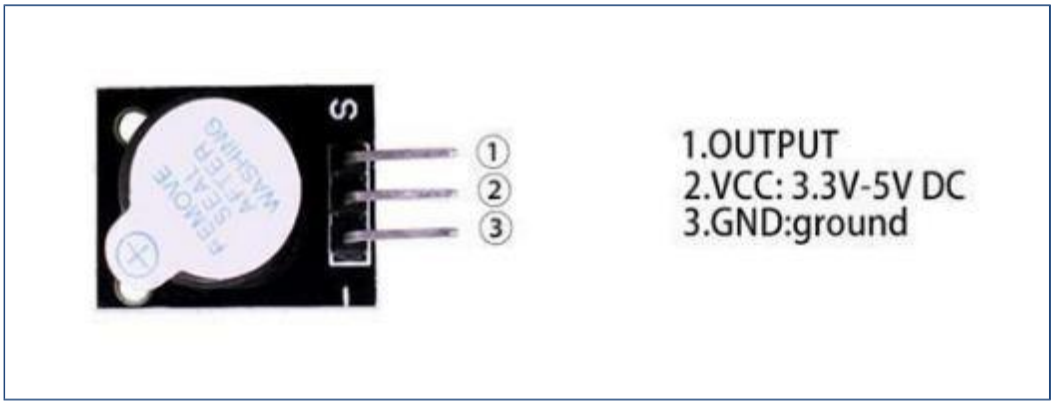
Überblick

In diesem Versuch lernen wir den Umgang mit einem Summer Modul. Wir werden mit einer Schaltung Lärm machen. Die am meisten verwendeten Komponenten, die Töne ausgeben können sind Lautsprecher und Summer. Im Vergleich zum Lautsprecher ist der Summer einfacher anzusteuern.

Komponentenübersicht

Summer:

Elektronische Summer werden mit Gleichstrom gespeist. Sie werden häufig in Computern, Druckern, Fotokopierern, Alarmen, elektronischem Spielzeug, elektronischen Geräten der Automobilindustrie, Telefonen, Zeitschaltuhren und vielen weiteren elektronischen Produkten verwendet. Die Summer lassen sich in aktive und passive einteilen. Der Summer mit dem weißen Klebeband auf der Oberseite ist der aktive, der ohne Klebeband der passive Summer.



Benötigte Komponenten:

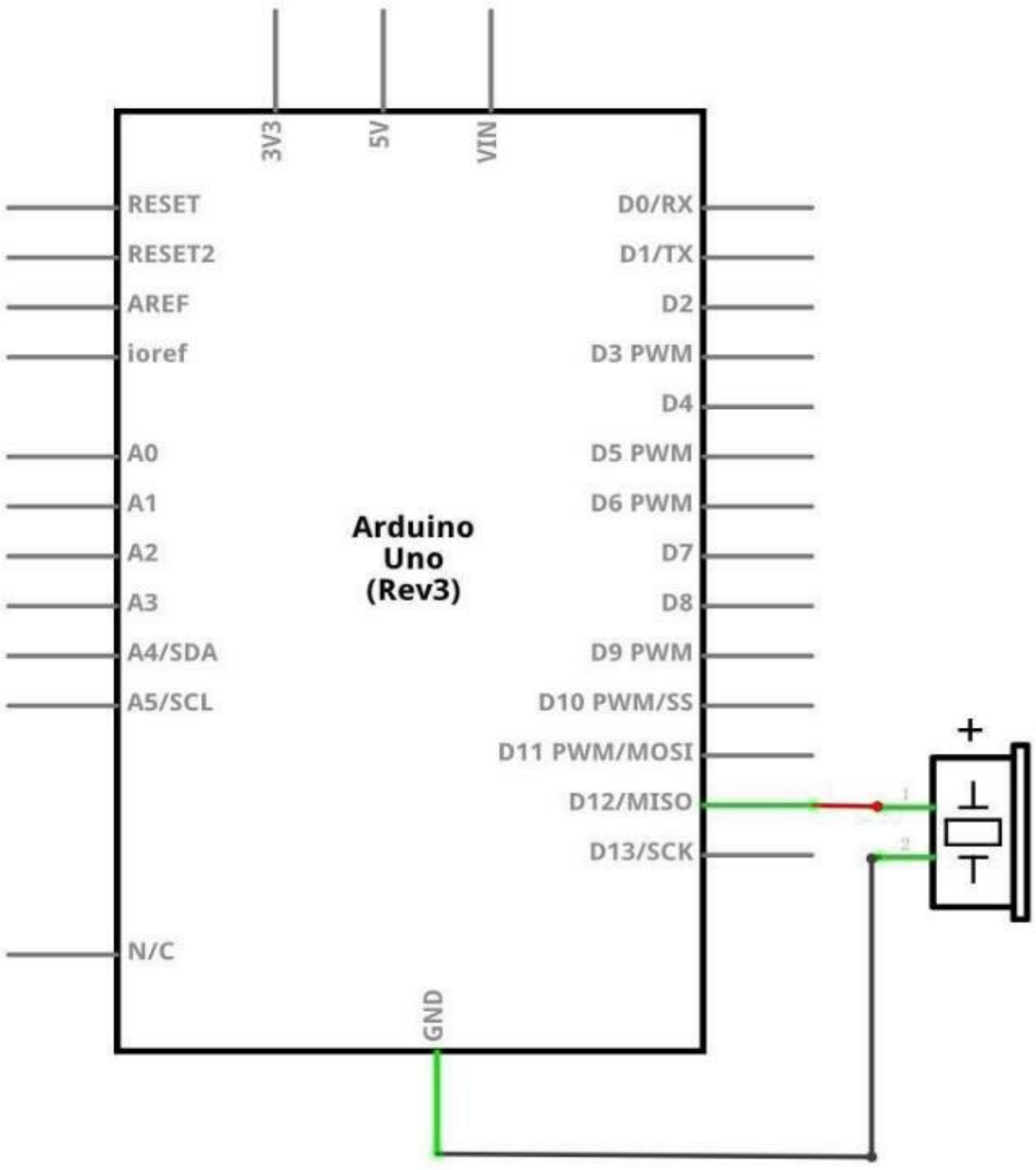
1 x Elegoo Uno R3

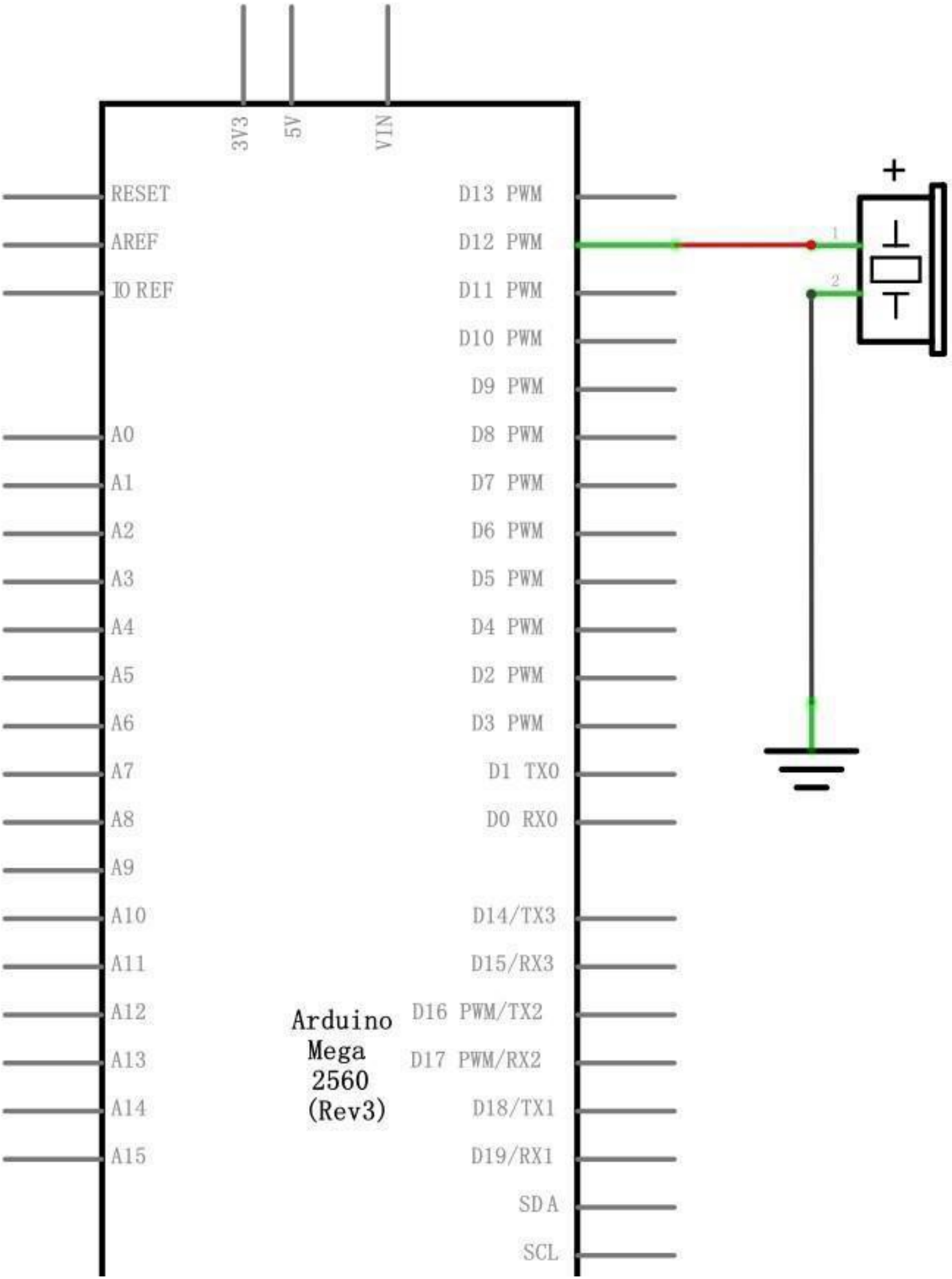
1x Aktiver Summer

3 x F-M Drähte

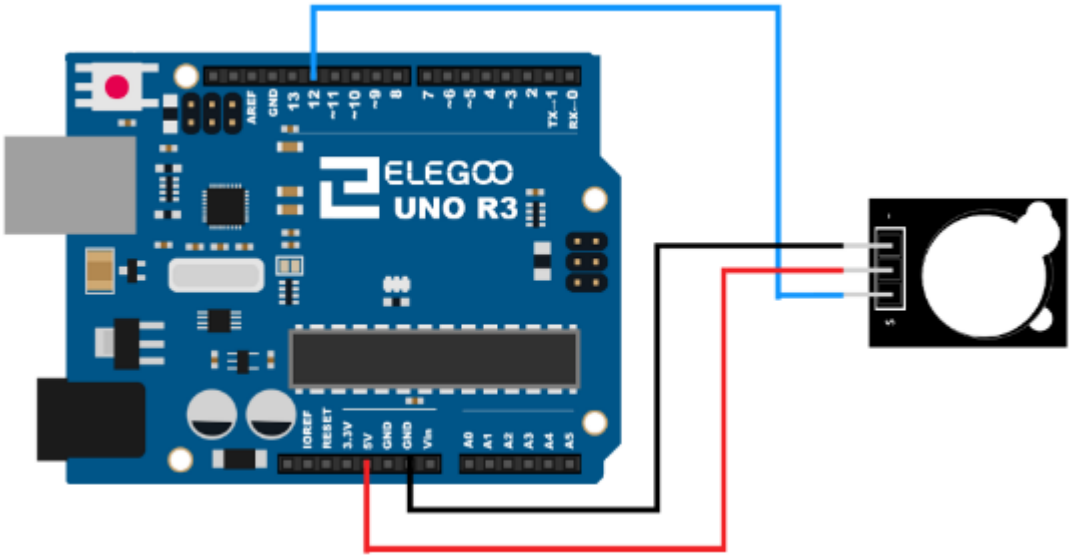
Verbindung

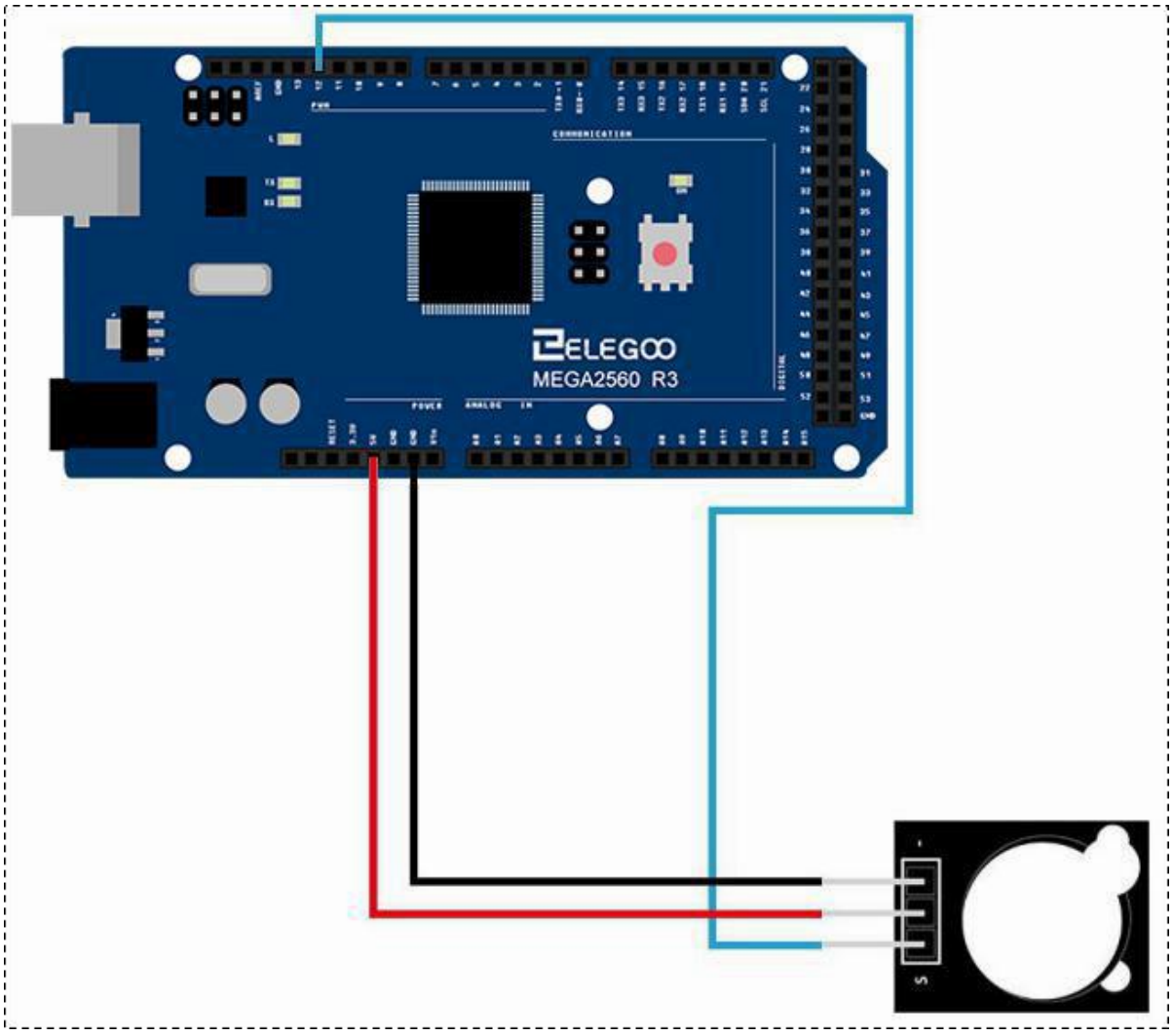
Schaltplan




3V3
5V
VIN
RESET
AREF
IO REF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
Arduino
Mega
2560
(Rev3)
+
1
2

Verdrahtungsplan

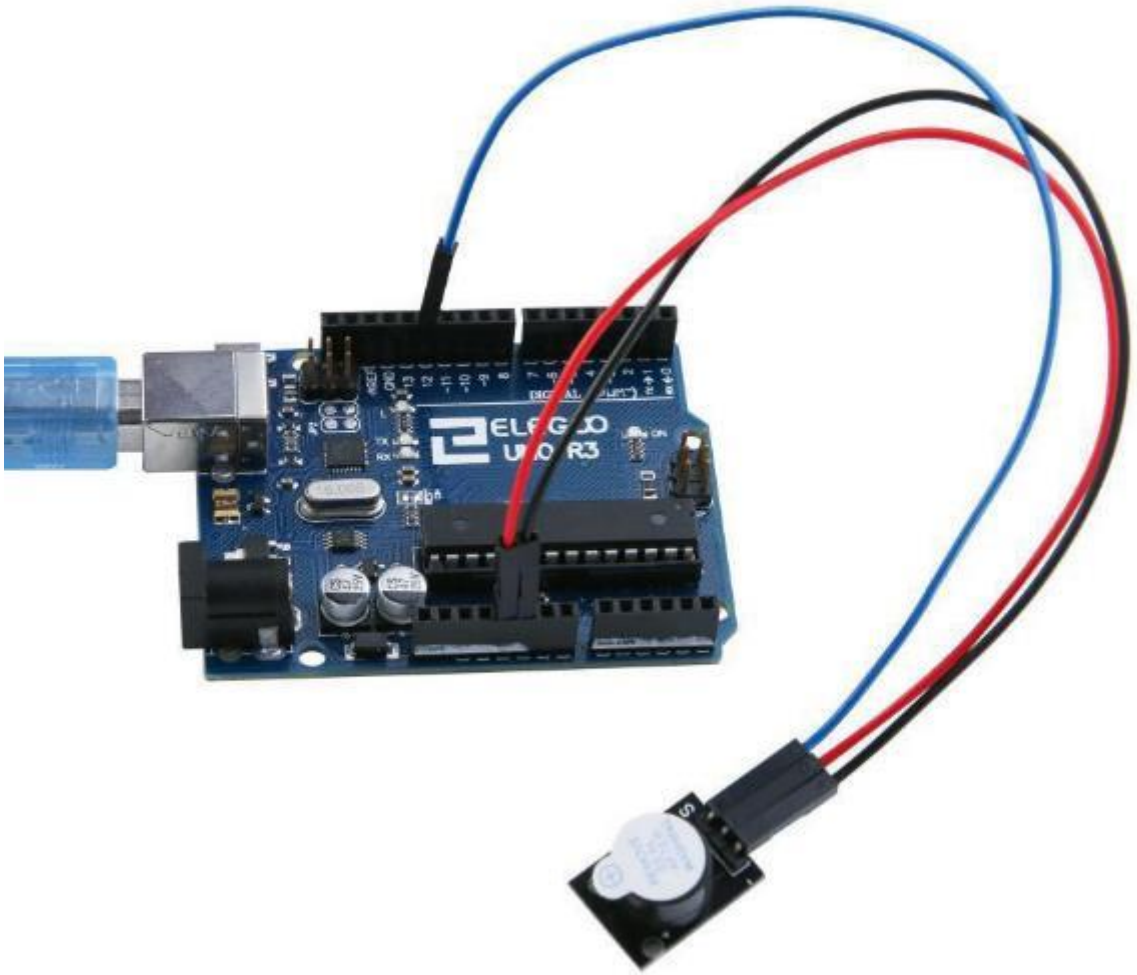




Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Code-Ordner（Lesson 9 Active buzzer）und klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen falls Fehlermeldungen auftreten.

Beispielbild



Im Folgenden finden Sie den Code

```
/* der Pin für den aktiven Summer */
int buzzer = 12;
void setup()
{
/* Pin an dem der Summer angeschlossen ist als Ausgang definieren */
  pinMode(buzzer,OUTPUT);
}
void loop()
{
  unsigned char i;
  while(1)
  {
    /* Ausgeben einer Frequenz */
      for(i=0;i<80;i++)
      {
        /*Summer aktivieren*/
        digitalWrite(buzzer,HIGH);

        delay(1);                    //1ms warten
        digitalWrite(buzzer,LOW); /*Summer ausschalten*/
        delay(1);                  //1ms warten
        }
          /* Ausgeben einer anderen Frequenz */
          for(i=0;i<100;i++)
            {
               digitalWrite(buzzer,HIGH);
               delay(2);                  //2ms warten
               digitalWrite(buzzer,LOW);
               delay(2);                  //2m warten
            }
   }
```

Aus dem obigen Programm haben wir die Schleifensyntax gelernt:

while Schleifen

Beschreibung

„while Schleifen“ laufen solange, bis der Ausdruck in der Klammer () falsch wird. Etwas muss die überprüfte Variable ändern, sonst wird die while-Schleife nie beendet. Dies kann in Ihrem Code z.B. eine inkrementierte Variable oder ein externes Ereignis, z.B. der Wert eines Sensors sein.

Syntax

```
while(Ausdruck){    // Anweisung(en) }
```

Parameter

Ausdruck – ein boolescher C – Ausdruck, der als wahr oder falsch ausgewertet werden kann

Beispiel

```
var = 0; while(var < 200)
{
        // mach etwas 200 mal
        var++;
}
```

Lektion 10 Passiver Summer

Überblick

In dieser Lektion lernen Sie, wie man einen passiven Summer verwendet.

Ziel des Experiments ist es, acht verschiedene Klänge zu erzeugen, die jeweils 0,5 Sekunden dauern vom zweigestrichenen C (523Hz) bis zum dreigestrichenen C (1047 Hz).

Benötigte Komponenten:

1x Elegoo Uno R3

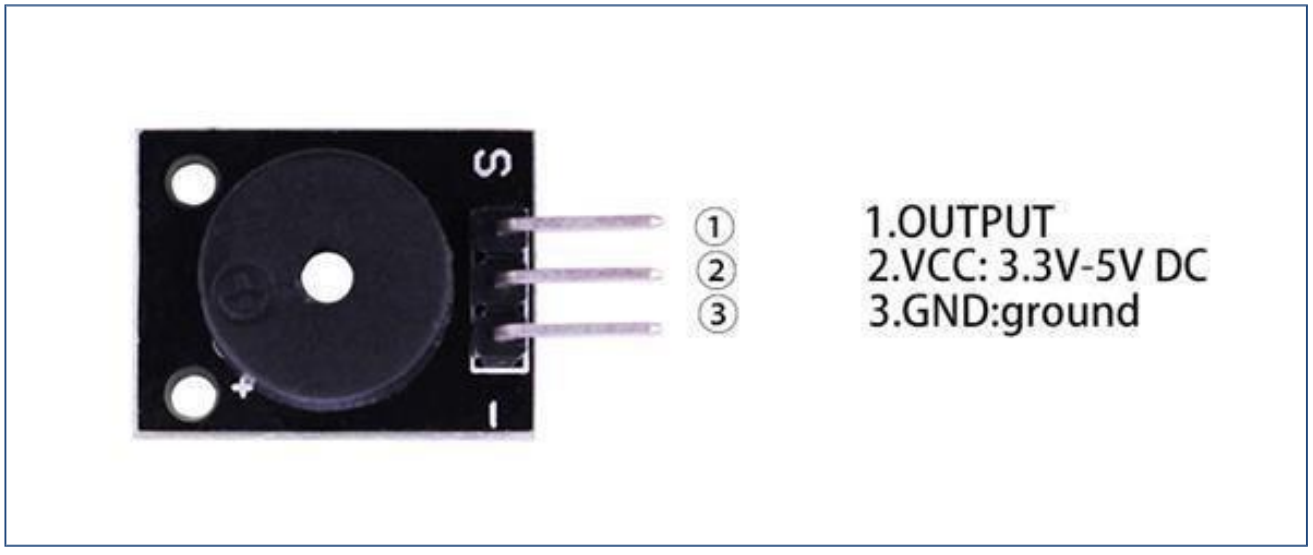
1x Passiver Summer

3x F-M Drähte

Komponenteneinführung

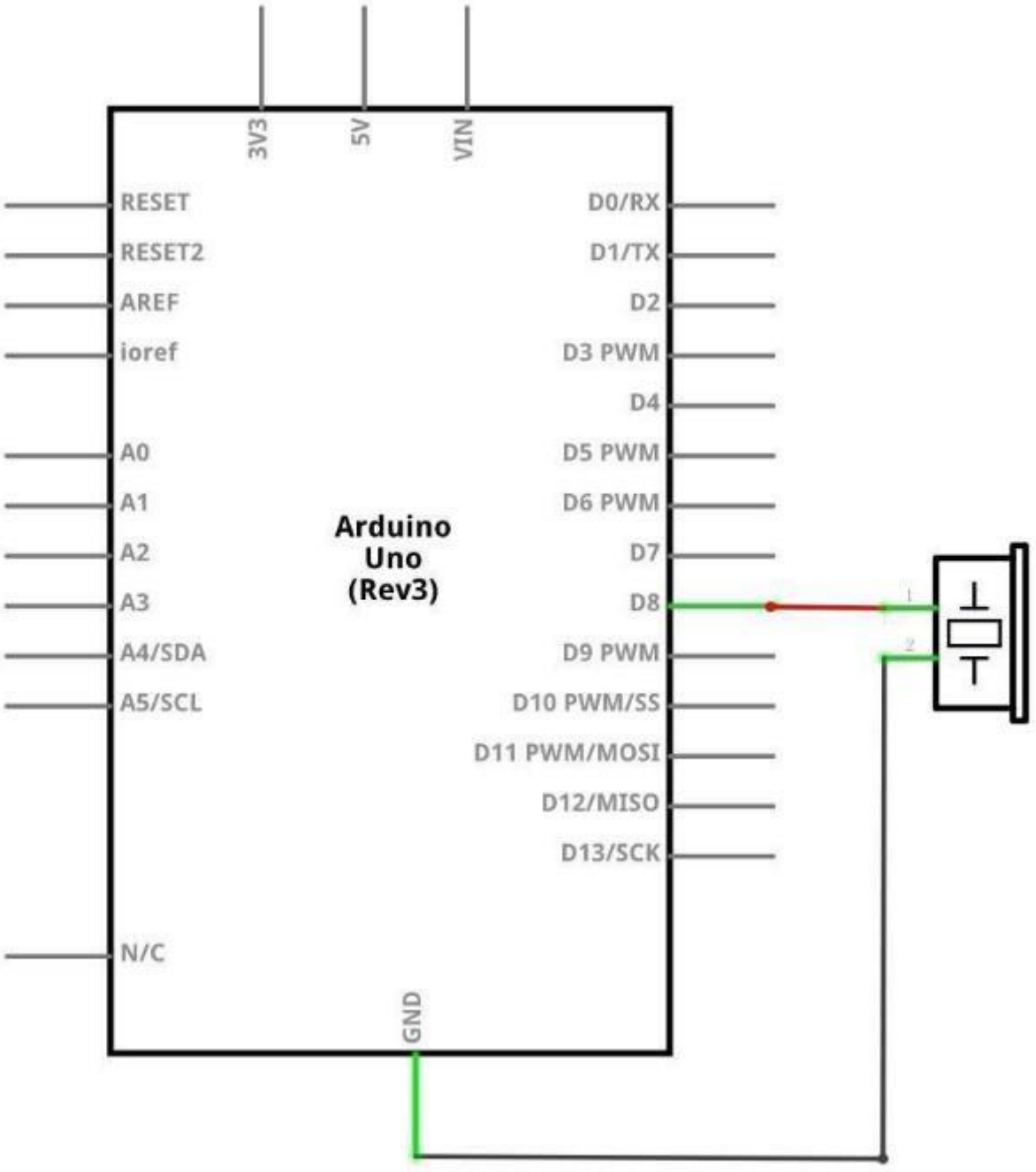
Passiver Summer:

Das Funktionsprinzip des passiven Summers besteht darin, die Luft mit der Frequenz zum Schwingen zu bringen, mit der er angesteuert wird.

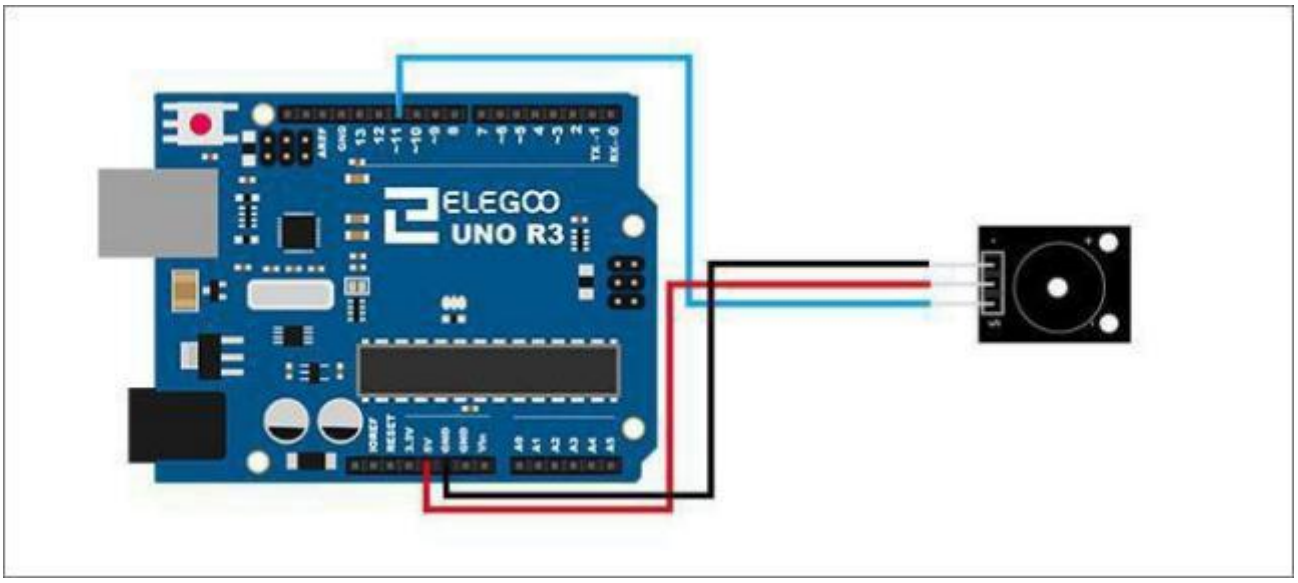


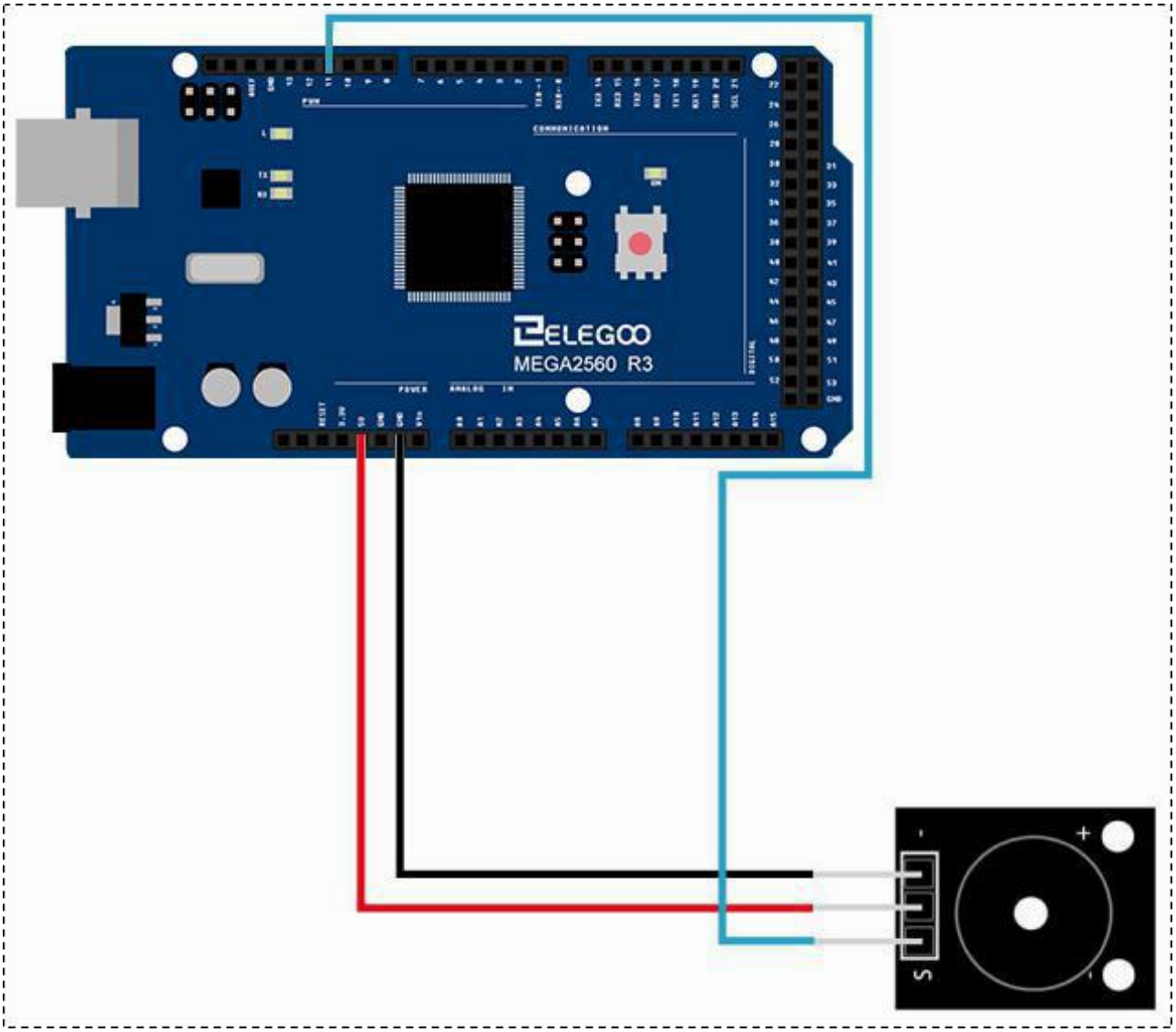
Verbindung

Schaltplan



Verdrahtungsplan





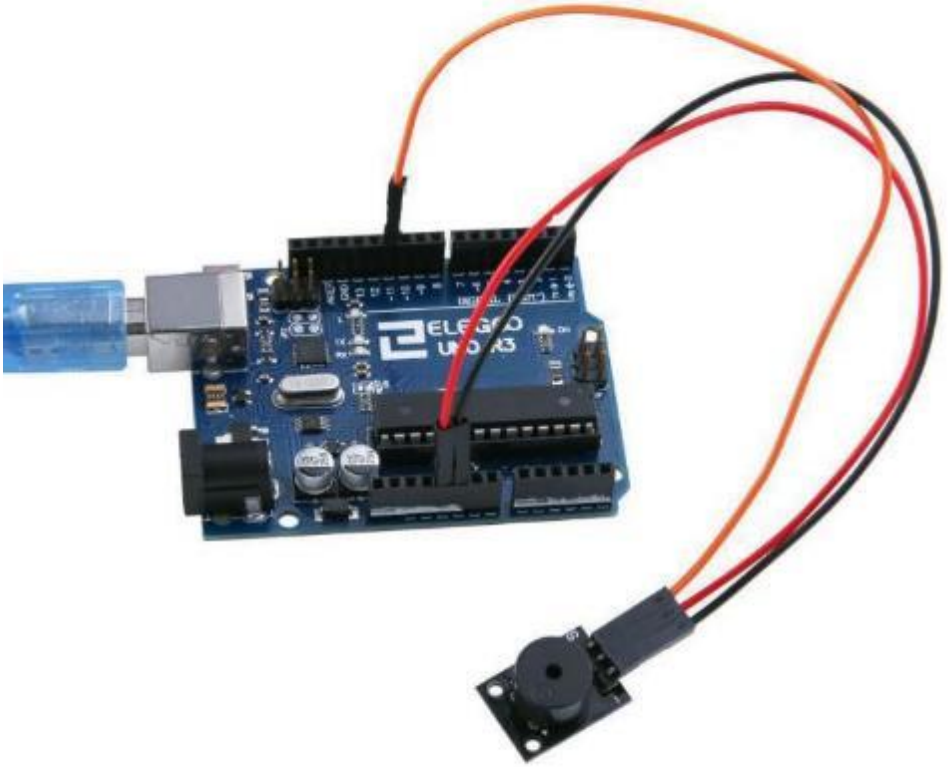
Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Codeordner -（Lesson 10 - Passive Buzzer）and Klicken Sie auf „Hochladen“, um das Programm hochzuladen.

Stellen Sie sicher, dass die Bibliothek < pitches> installiert ist. Andernfalls wird Ihr Code nicht funktionieren.

Details zum Laden von Bibliotheksdateien finden Sie in Lektion 1.

Beispielbild



Im Folgenden finden Sie den Code für das Experiment

```
#include "pitches.h"

// die zu spielenden Noten
int melody[] = {
   NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5, NOTE_C6};
int duration = 500;    // 500 Millisekunden

void setup() {

}

void loop() {
   for (int thisNote = 0; thisNote < 8; thisNote++) {
      // Ausgabe an Pin11 (D8), jede Note dauert 0,5 s
      tone(11, melody[thisNote], duration);

      // Nach jeder Note eine Sekunde warten
      delay(1000);
   }

   // Nach 2s Wartezeit geht es von vorne los
   delay(2000);
}
```

Hinweis: Pitches.h enthält nur Konstanten für Noten. Dabei werden diversen Noten die zugehörigen Frequenzen zugewiesen. Z.B. NOTE_C5 entspricht 523 Hz.

Die Methode tone() kommt direkt aus dem Arduino Framework und erwartet als ersten Parameter einen Pin an dem der Ton ausgegeben wird, als zweiten Parameter die Frequenz und als optionalen dritten Parameter die Zeitdauer des Tons.

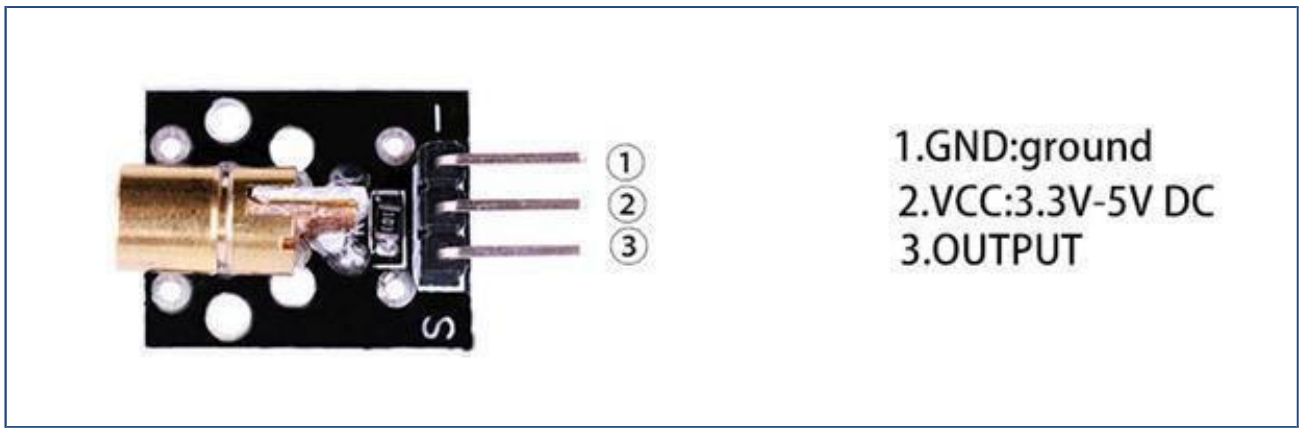
Lektion 11 LASER MODUL

Überblick

In diesem Versuch lernen wir, wie man ein Laser Modul verwendet.

Laserstrahler

Verbinden Sie den ("-") Pin (Masse, Pin Nummer 1 im nachfolgenden Bild) auf dem Laser-Modul mit dem GND-Pin auf dem Arduino Board, den zweiten/mittleren Pin mit dem 5V Pin auf dem Arduino Board und den dritten ("S") Pin mit dem Digitalport 9 auf dem Arudino Board.
Wellenlänge: 650nm



Benötigte Komponenten:

1x Elegoo Uno R3

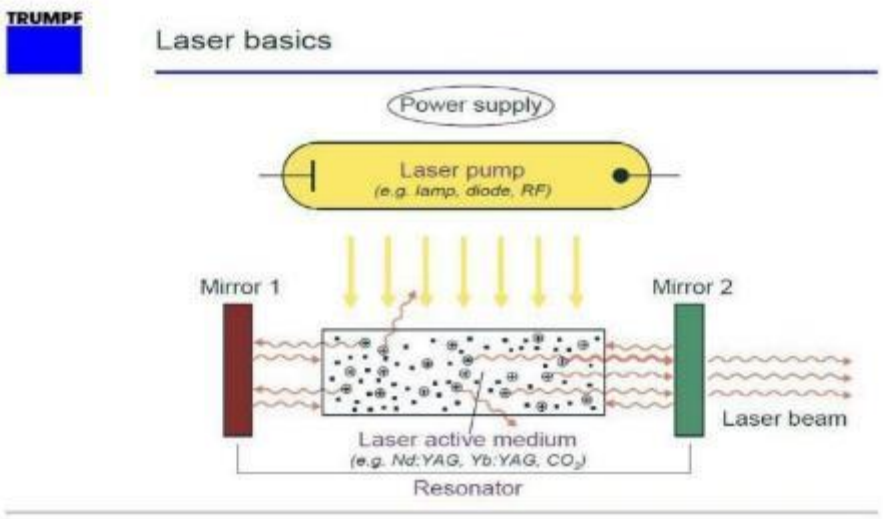
1x USB Kabel

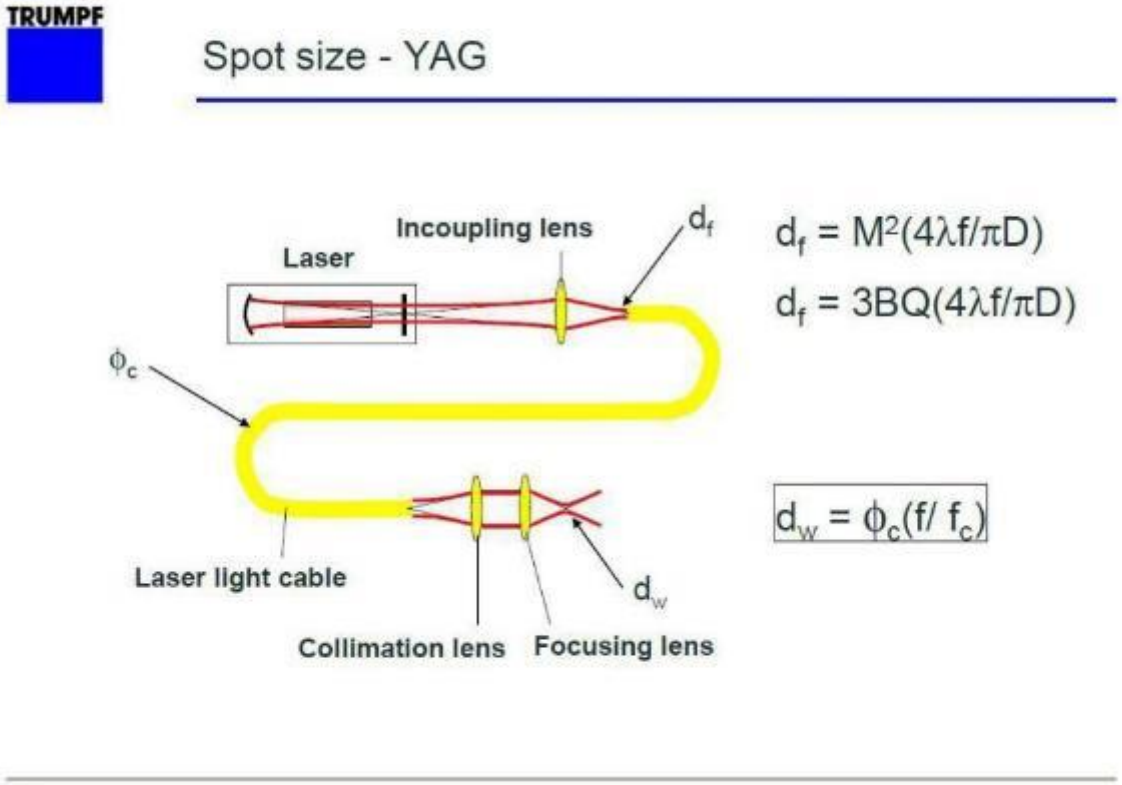
1x Laser Modul

3x F-M Drähte

Komponenteneinführung:

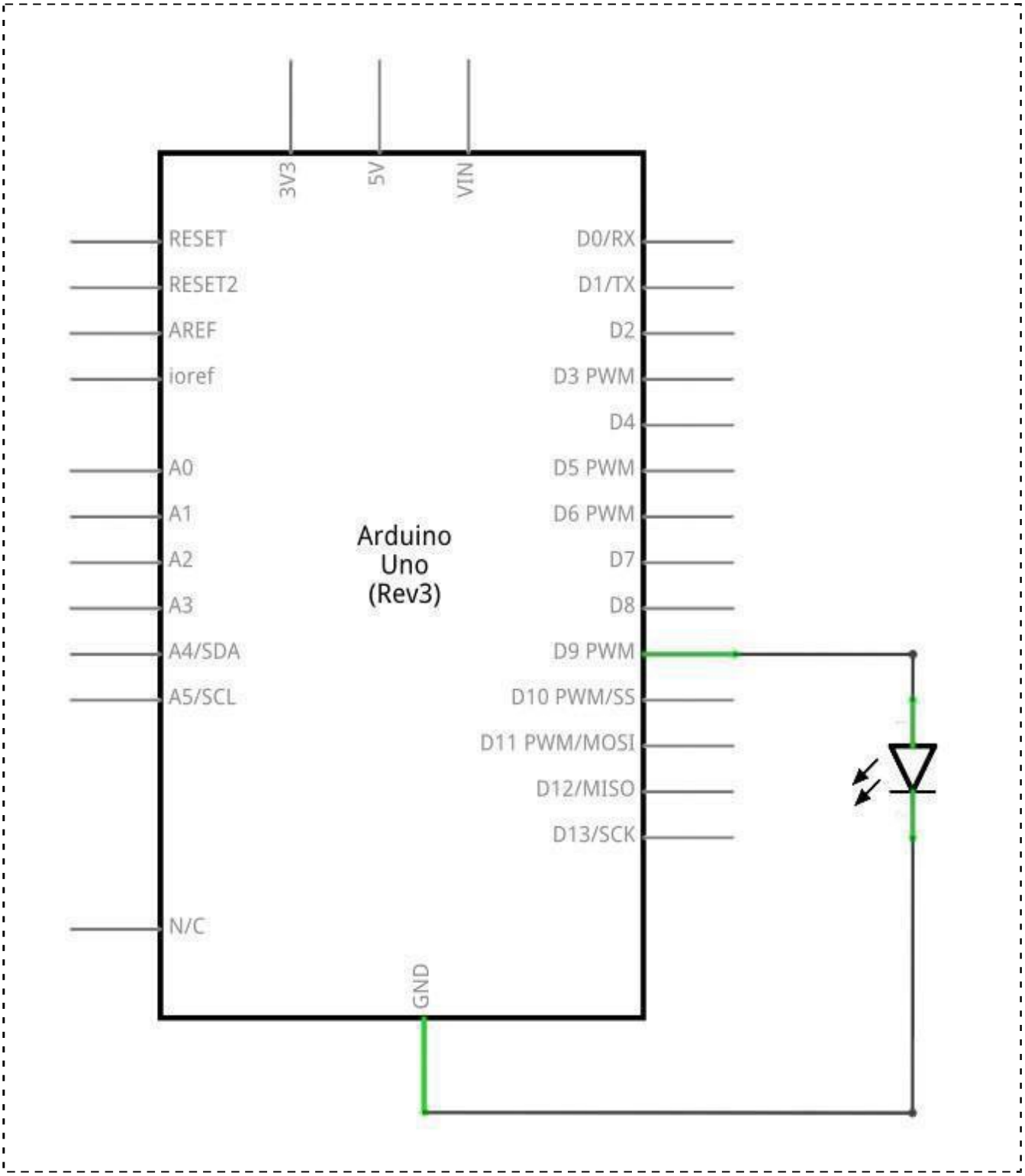
Laser: Auszug aus dem Datenblatt



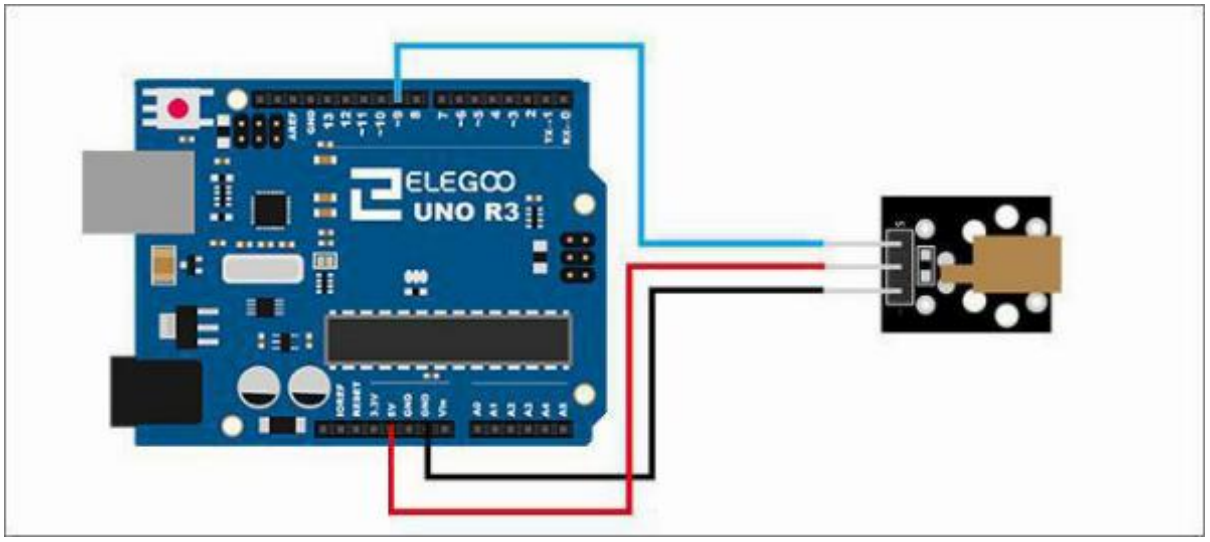


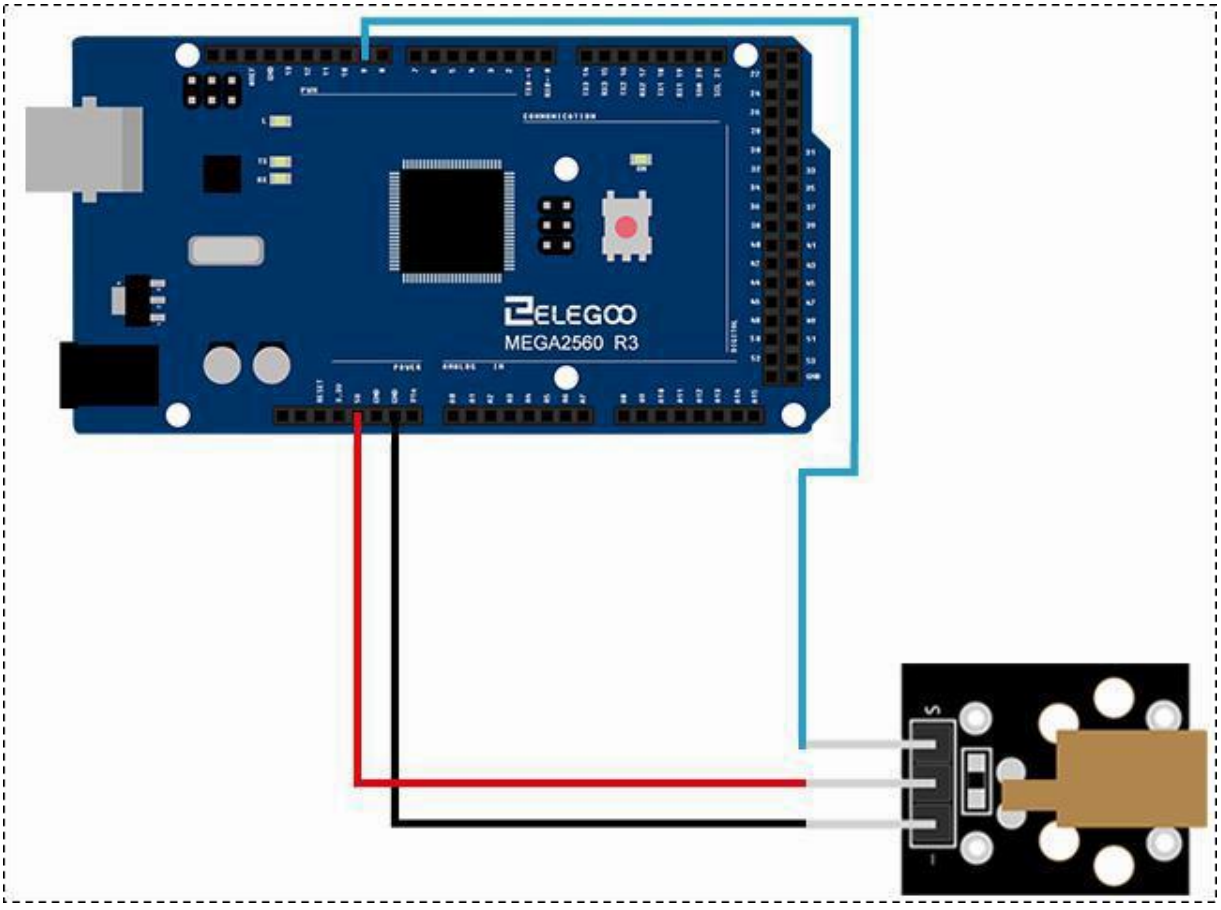
Verbindung

Schaltplan



Verdrahtungsplan

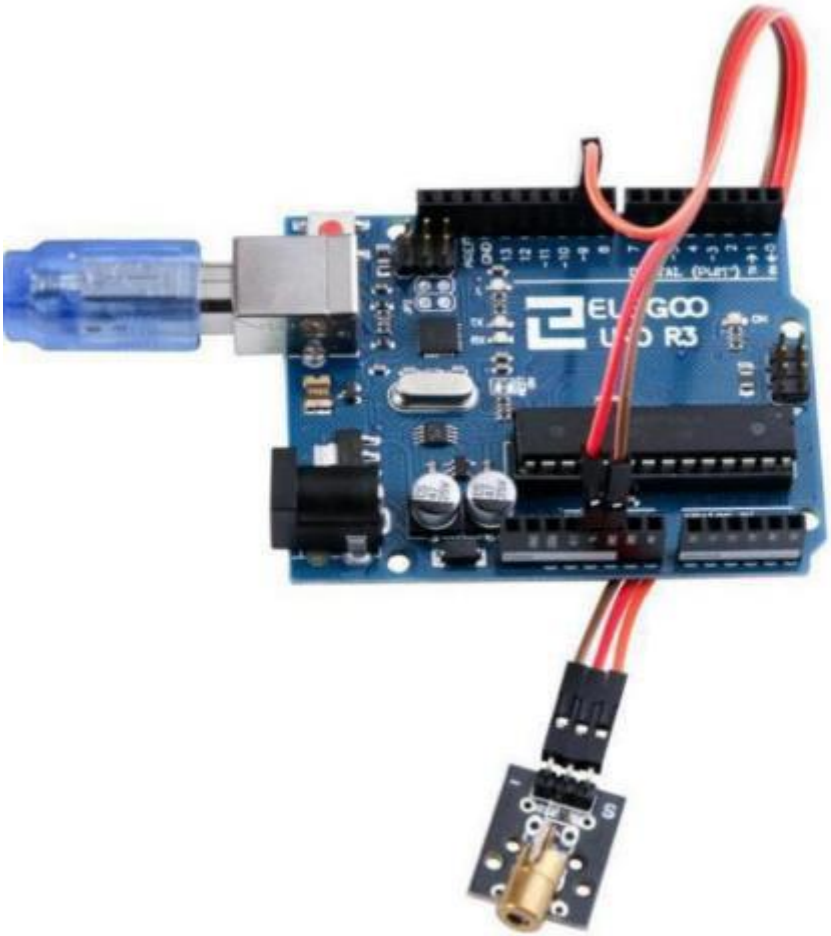




Code

Nachdem Sie die Verdrahtung abgeschlossen haben, laden Sie bitte das Programm "Lesson 11 LASER MODULE" im Ordner "code" auf Ihren Arduino. Wir sehen, dass das Modul rotes Laserlicht aussendet. Achtung: Bitte nicht direkt in die Lichtquelle sehen und Schutzbrille aufsetzen (nicht beiliegend).

Beispielbild



Im Folgenden finden sie den Code und einige Erklärungen

```
int. pos = 0;

void setup()

{

/*Pin 9 als Ausgang festlegen*/
   pinMode(9,OUTPUT);

}


void loop()

{

   for (pos = 0; pos <= 255; pos += 1) //Aufsteigend (heller werdend) von 0 bis 255

   {

      / *Mittel Pulsweitenmodulation das Laser Modul ansteuern. Je höher der , pos ‘ Wert ist, desto heller leuchtet das Laser Modul*/
      analogWrite(9, pos);

      delay(25);

   }

   for (pos = 255; pos >= 0; pos -= 1) //Absteigend (dunkler werdend) von 255 bis 0

   {

      analogWrite(9,pos);

      delay(25);

   }

}
```

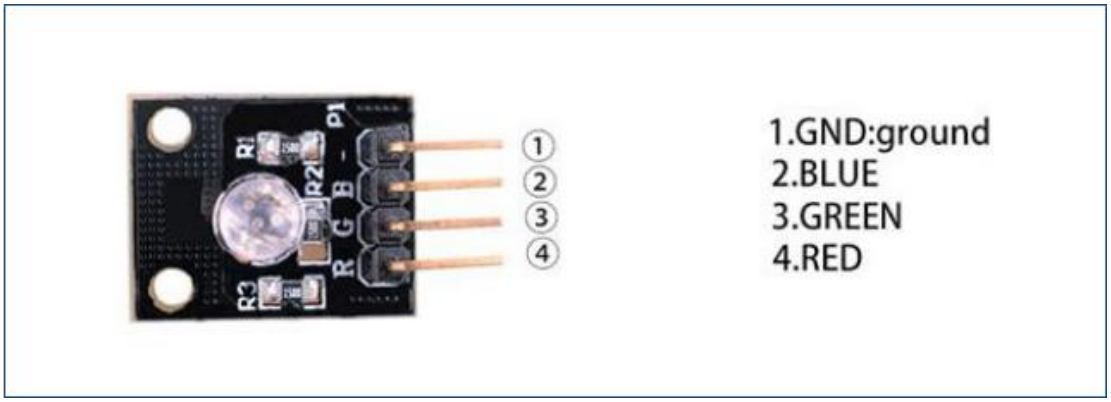
Lektion 12 SMD RGB MODUL UND RGB MODUL

Überblick

In diesem Versuch lernen wir den Umgang mit dem SMD RGB Modul und dem RGB (RGB steht für rot, grün, blau) Modul. Eigentlich sind die Funktionen der beiden Module SMD-RGB und RGB nahezu identisch, aber die Gehäuseform ist unterschiedlich. SMD-RGB-LED-Modul und RGB-Modul bestehen aus Vollfarb LEDs (in rot, grün und blau). Durch die Spannung an den Pins R, G, B können wir die Stärke der drei Primärfarben (rot/blau/grün) so einstellen, dass das als Ergebnis die Wirkung einer einzigen Farbe entsteht. **D.h. das Modul kann in jeder beliebigen Farbe leuchten und die Farbe kann via Arduino eingestellt werden.**

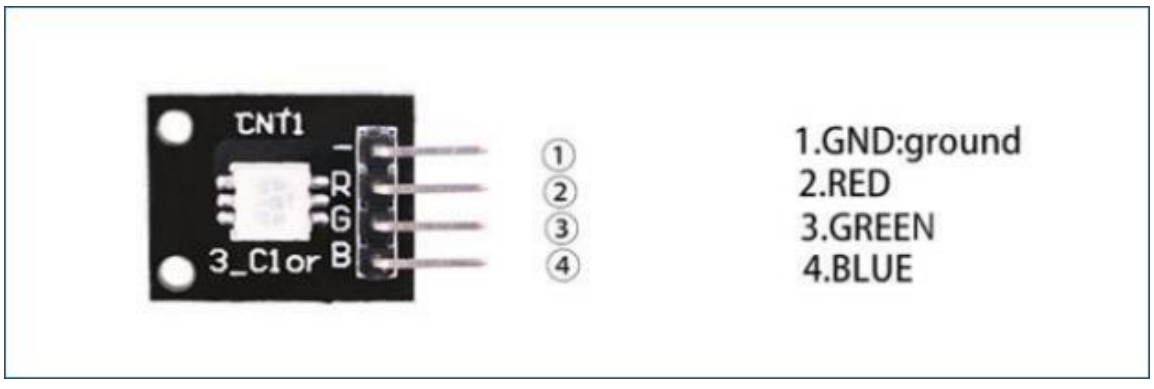
RGB LED

RGB-LED mit klarer Linse und eingebautem 150 Ohm Vorwiderstand für 5 V Betrieb.



SMD RGB

RGB-LED im SMD-Gehäuse ohne Vorwiderstand.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

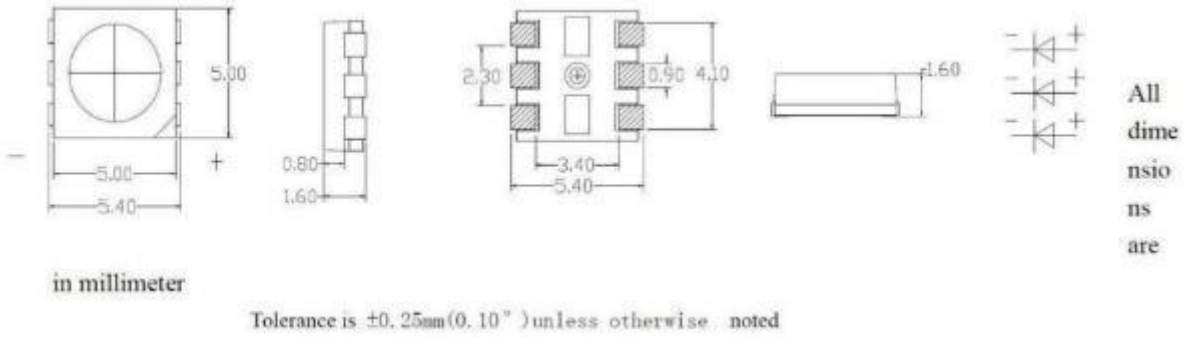
1x SMD RGB Modul

1x RGB Modul

4x F-M Drähte

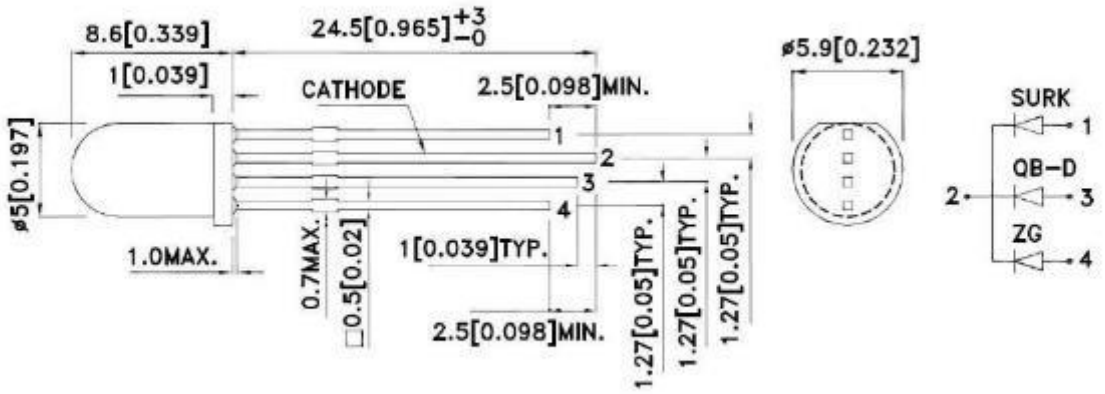
Komponenteneinführung (Auszüge aus dem Datenblatt)

SMD RGB:



Parameter	Symbol	Value	Unit
Forward Current	If	20	mA
Reverse Voltage	Vr	5	V
Operating Temperature	Topr	-25～+85	℃
Storage Temperature	Tstg	-35～+85	℃
Soldering temperature	Tsol	260±5℃ (for4sec)	℃
Power Dissipation	Pd	R=40	mW
Pulse Current	I_{FP}	100	mA

RGB:



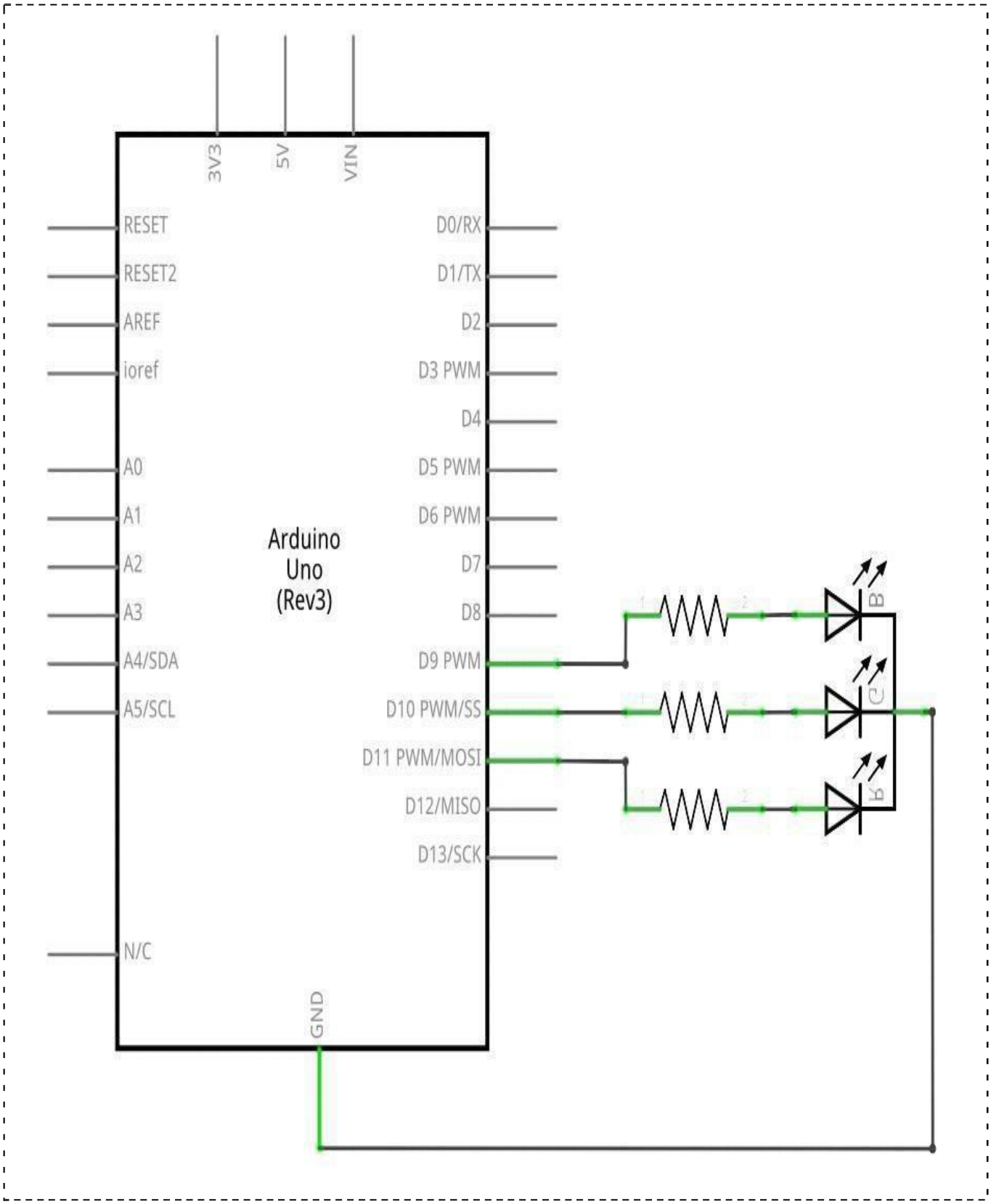
Electrical / Optical Characteristics at TA=25°C

Symbol	Parameter	Device	Typ.	Max.	Units	Test Conditions
λpeak	Peak Wavelength	Hyper Red Blue Green	650 468 515		nm	IF=20mA
λD [1]	Dominant Wavelength	Hyper Red Blue Green	630 470 525		nm	IF=20mA
Δλ1/2	Spectral Line Half-width	Hyper Red Blue Green	28 25 30		nm	IF=20mA
C	Capacitance	Hyper Red Blue Green	35 100 45		pF	VF=0V;f=1MHz
VF [2]	Forward Voltage	Hyper Red Blue Green	1.95 3.3 3.3	2.5 4 4.1	V	IF=20mA
IR	Reverse Current	Hyper Red Blue Green		10 50 50	uA	VR=5V

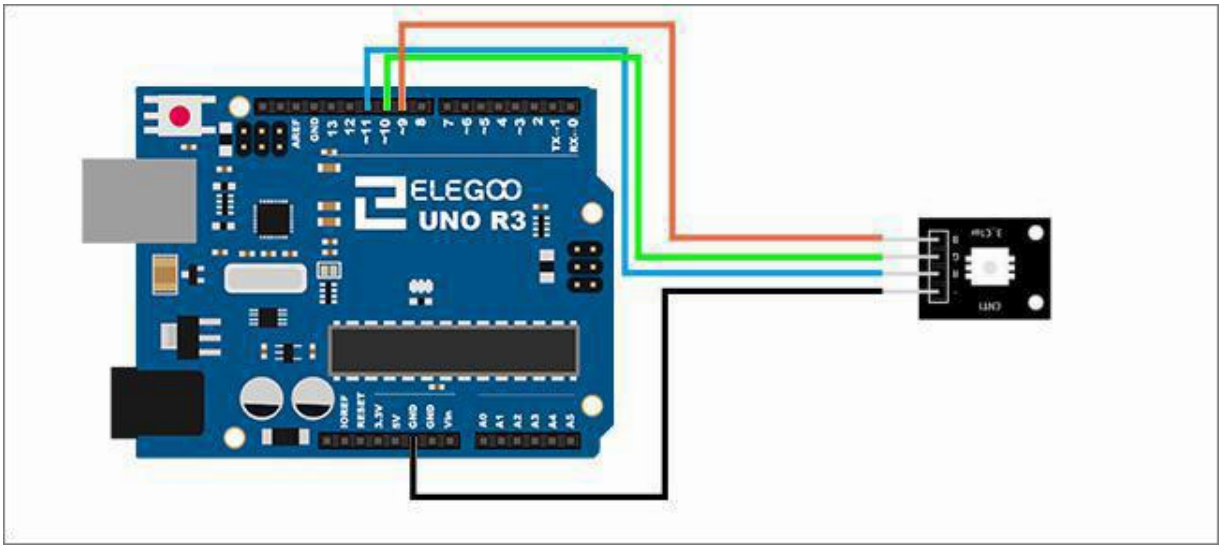
Notes:
1.Wavelength: +/-1nm.
2. Forward Voltage: +/-0.1V.

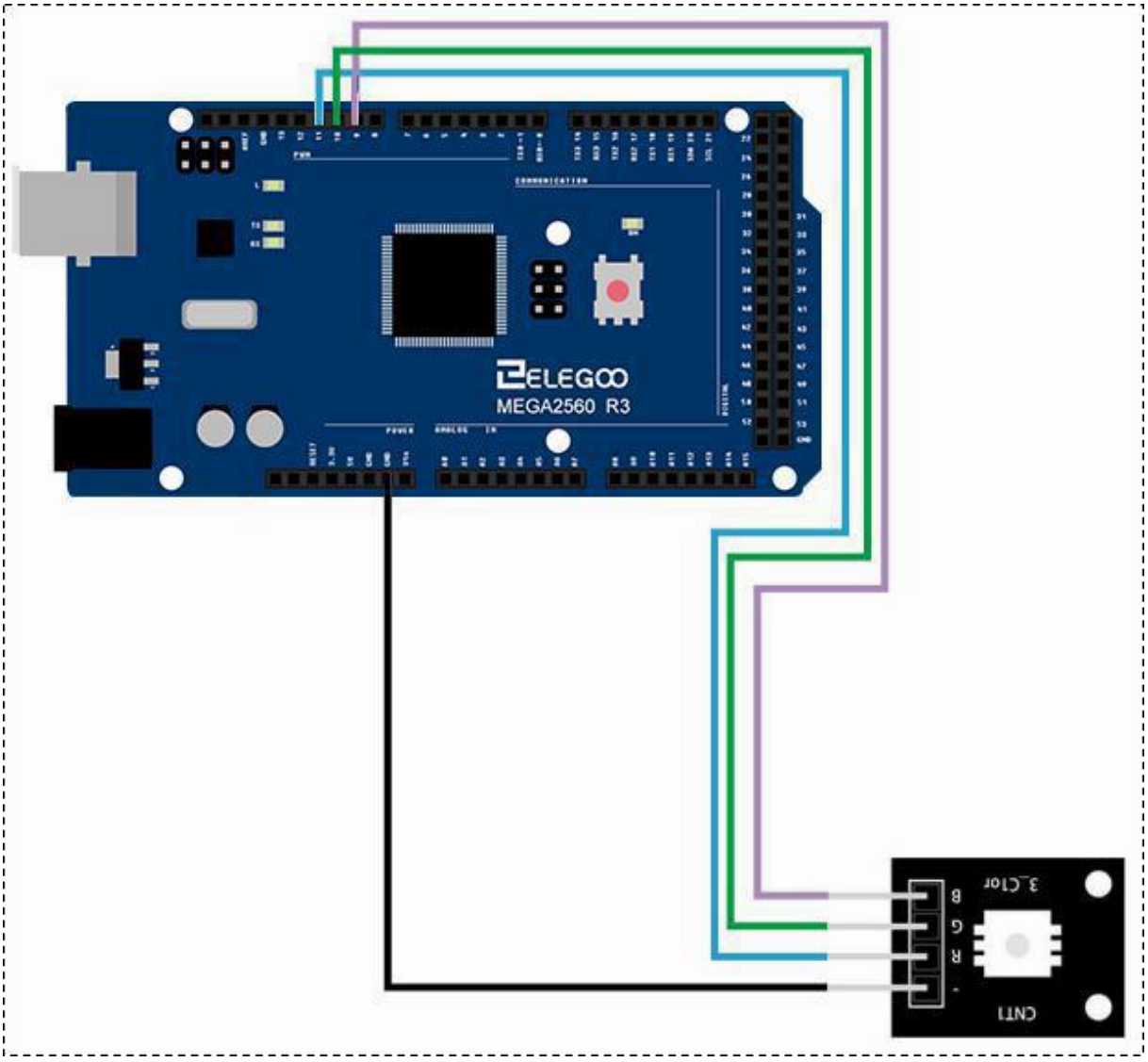
Verbindung

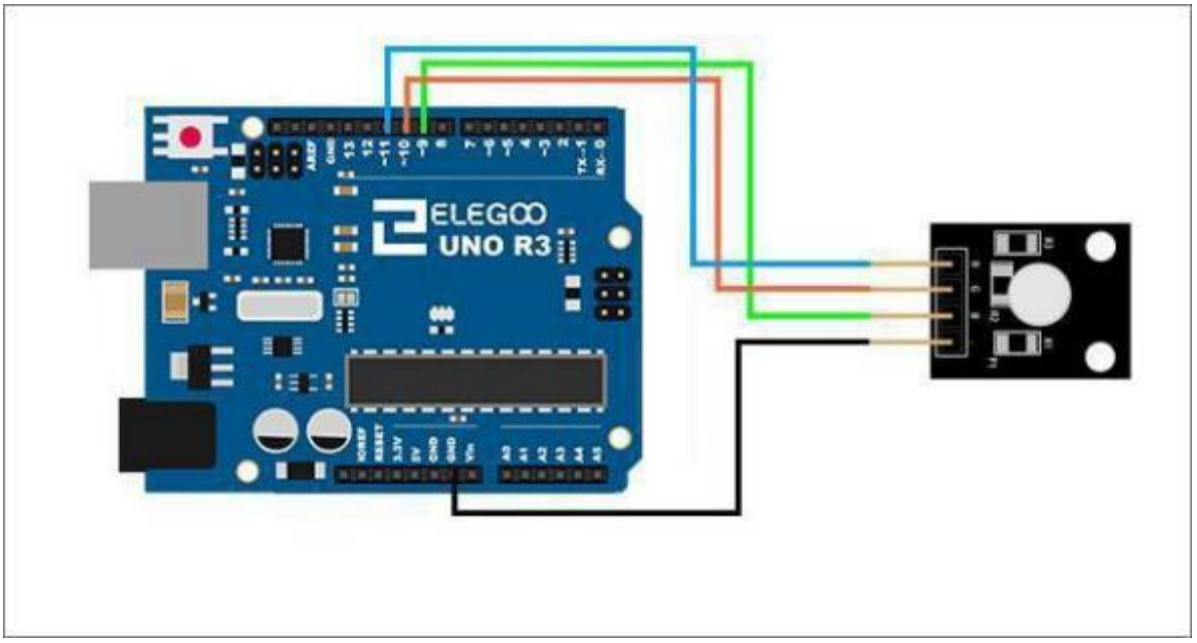
Schaltplan

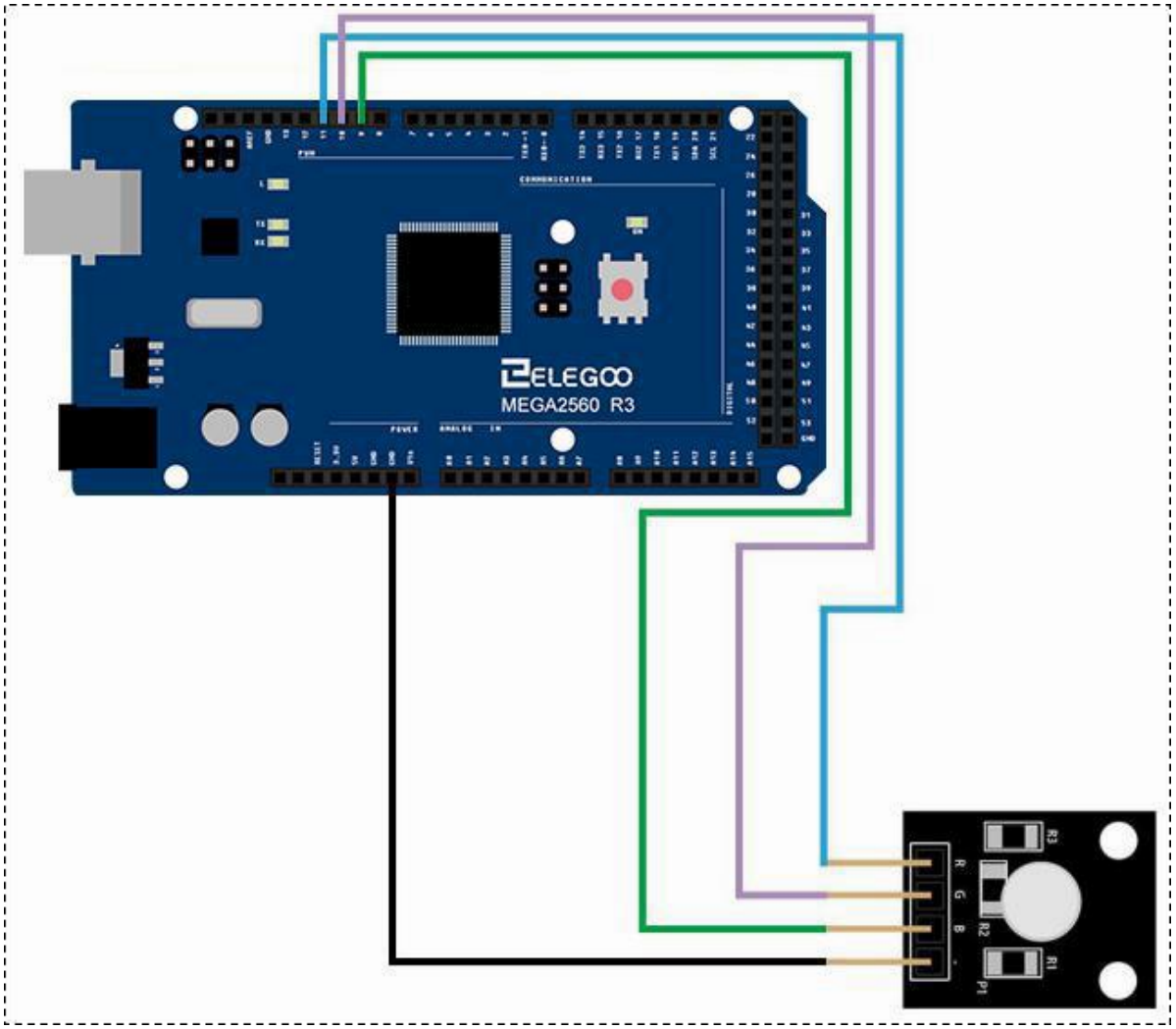


Verdrahtungsplan





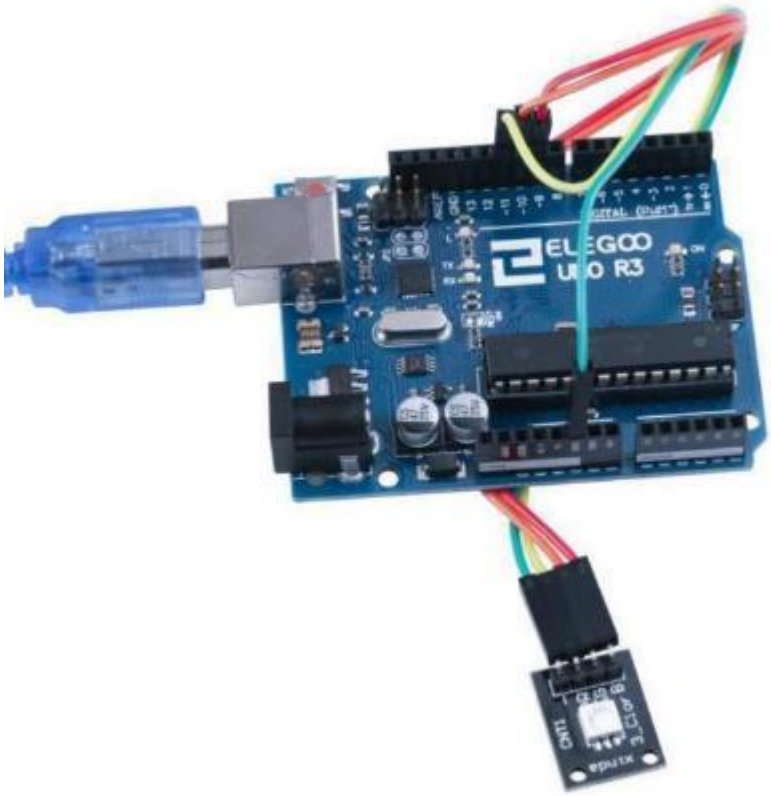

ELEGOO
UNO R3

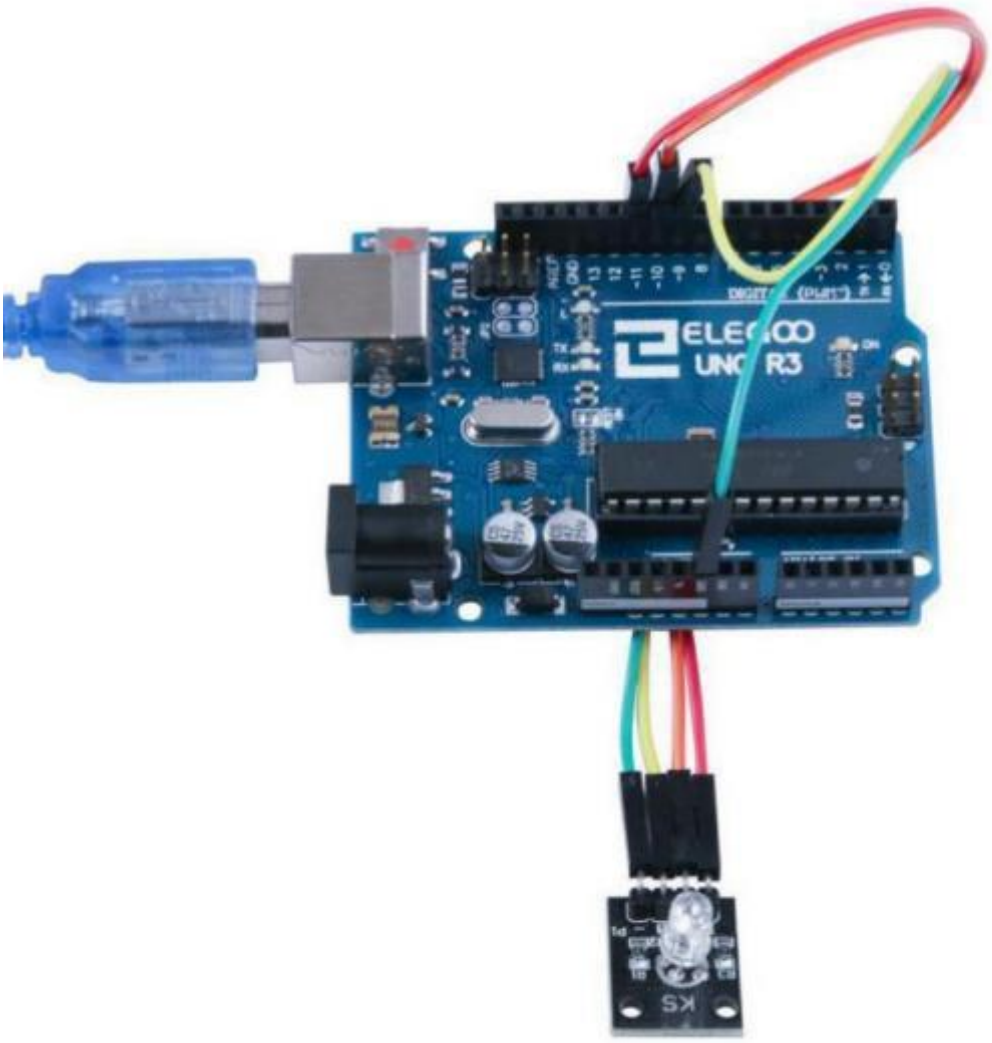

ELEGOO
MEGA2560 R3

Code

Nachdem Sie die Verdrahtung abgeschlossen haben, öffnen Sie bitte den Ordner "Lesson 12 SMD RGB MODULE AND RGB MODULE" und laden Sie das Programm auf Ihren Arduino. Sie können sehen, dass das Modul seine Farbe wie programmiert ändert. Wenn Sie die Farbe auf andere Weise ändern möchten, können Sie den Code anpassen.

Beispielbild




ELEGOO
UNO R3

Im Folgenden finden Sie den Code und einige Erklärungen

```
int. redpin = 11; / * der Pin der roten LED wird an Port 11 angeschlossen */
int greenpin =10; / * der Pin der grünen LED wird an Port 10 angeschlossen */
int bluepin =9; / * der Pin der blauen LED wird an Port 9 angeschlossen */
int val;
void setup(){
/*Pins 9 bis 11 als Ausgänge definieren*/
  pinMode(redpin, OUTPUT);
  pinMode(bluepin, OUTPUT);
  pinMode(greenpin, OUTPUT);
  Serial.begin(9600);
}
void loop(){
for(val=255; val>0; val--)/*Absteigend von 255 bis 1 */
{
  analogWrite(redpin, val); /*Die rote LED wird mit jedem Schleifendurchlauf heller*/
  analogWrite(greenpin, 255-val); /*Die grüne LED wird mit jedem Schleifendurchlauf dunkler*/
/*Die blaue LED wird mit jedem Schleifendurchlauf dunkler. Eigentlich kein guter Code, weil es hier zu
negativen Werten kommt. Bitte nicht nachmachen ☺*/
  analogWrite(bluepin, 128-val); /*Die blaue LED wird mit jedem Schleifendurchlauf dunkler.*/
  delay(50);
}
for(val=0; val<255; val++) /*Aufsteigend von 0 bis 254 */
{
  analogWrite(redpin, val);
  analogWrite(greenpin, 255-val);
  analogWrite(bluepin, 128-val);
  delay(50);
}
Serial.println(val, DEC);
}
```

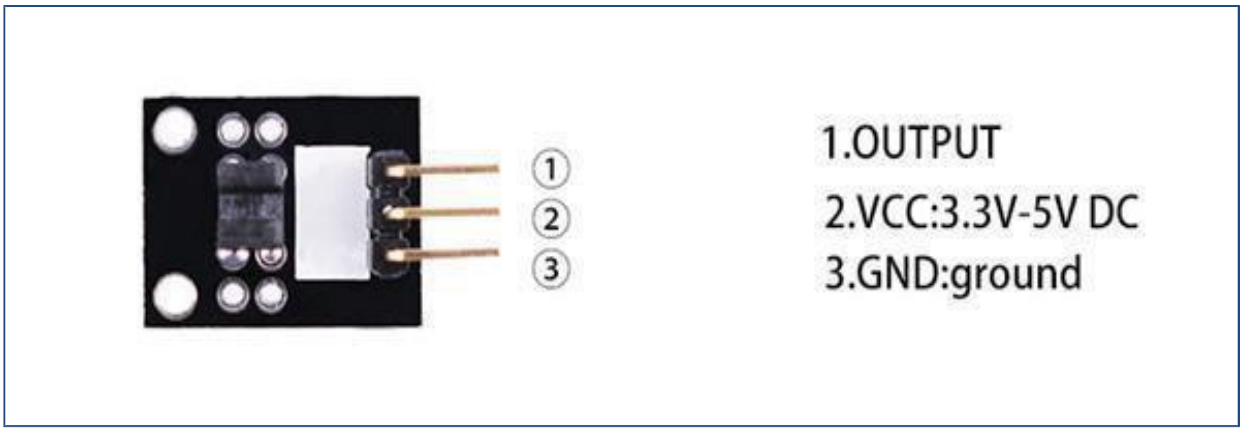
Lektion 13 Gabellichtschrankenmodul

Überblick

In diesem Versuch lernen wir, wie man ein Gabellichtschrankenmodul verwendet.

Lichtschranke

Gabellichtschranke. Der mittlere Pin wird an die + 5 V Versorgung des Arduino angeschlossen und der mit '-' gekennzeichnete Pin an die Masse (Ground). Das Ausgangssignal (mit einem 10 K Ohm Pullup gegen +5 V) liegt an Pin ‚1' an.



Benötigte Komponenten:

1 x Elegoo Uno R3

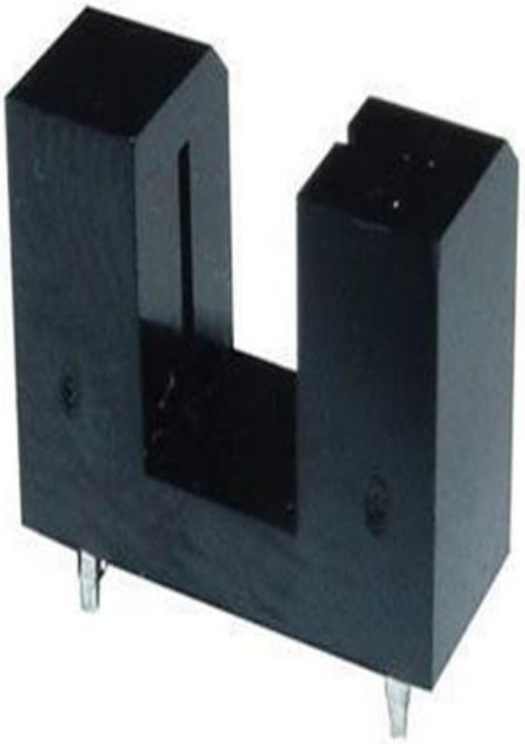
1 x USB Kabel

1 x Gabellichtschrankenmodul

3 x F-M Drähte

Komponenteneinführung

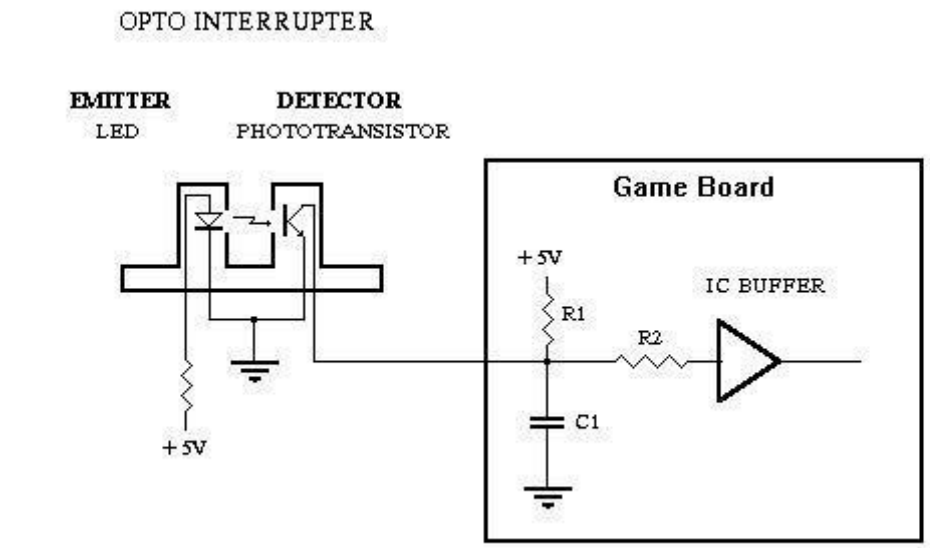
Optischer Unterbrechungssensor：

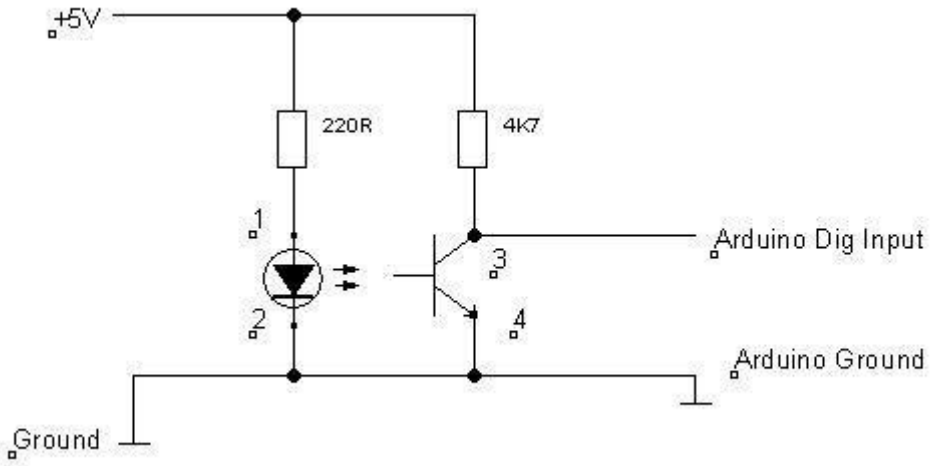


Wenn kein Hindernis zwischen Sender und Empfänger (=im Bereich zwischen den Zinken der Gabel) ist, schaltet der Lichtstrahl den Fototransistor durch, so dass 0V am Ausgang des Moduls anliegt. Dies ist sehr schön im nachfoldenen Bild zu sehen. Der Phototransistor schaltet in diesem Fall Knoten 3 auf die selbe Spannung wie Knoten 4.

Wird der Lichtstrahl dagegen unterbrochen sperrt der Fototransistor und Knoten 3 liegt (über einen Kollektorwiderstand; in diesem Fall auch Pull-Up Widerstand) auf 5V.

Prinzip

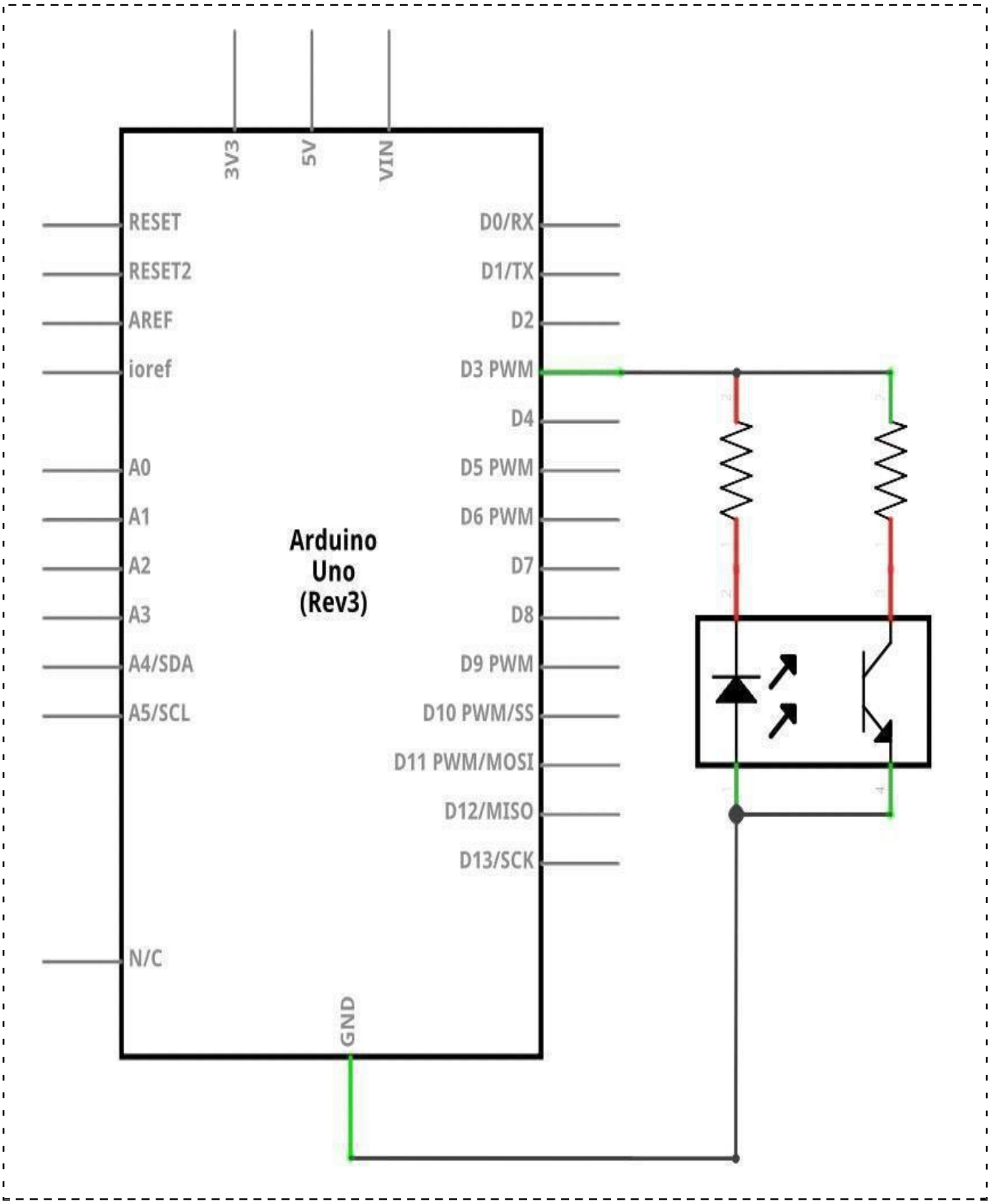




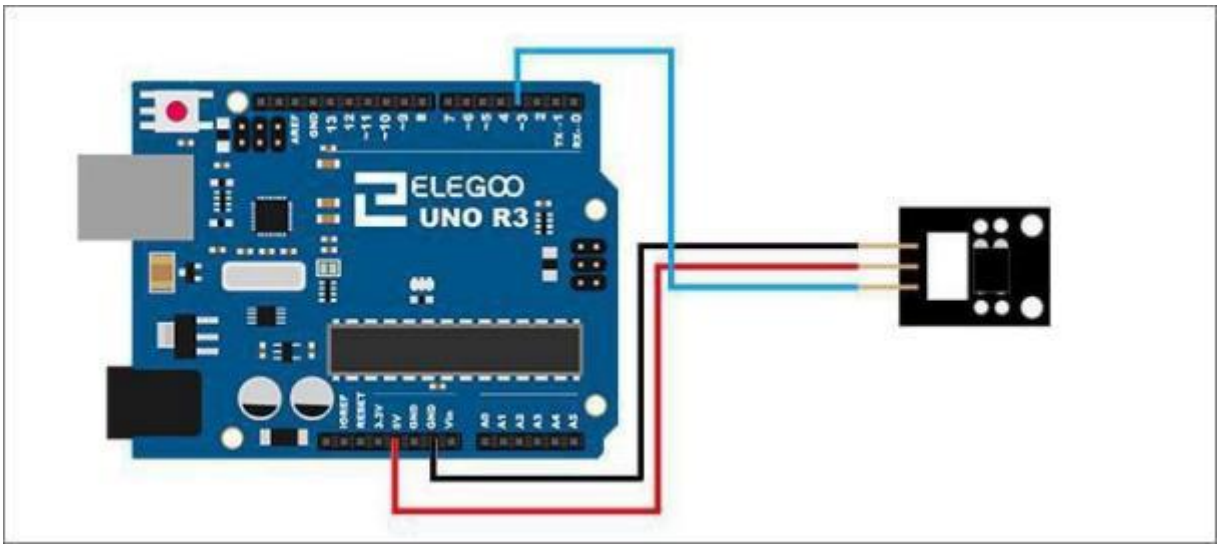
In unserem Beispielcode weiter hinten werden wir die interne LED des Arduino benutzen um den Zustand der Lichtschranke anzuzeigen.

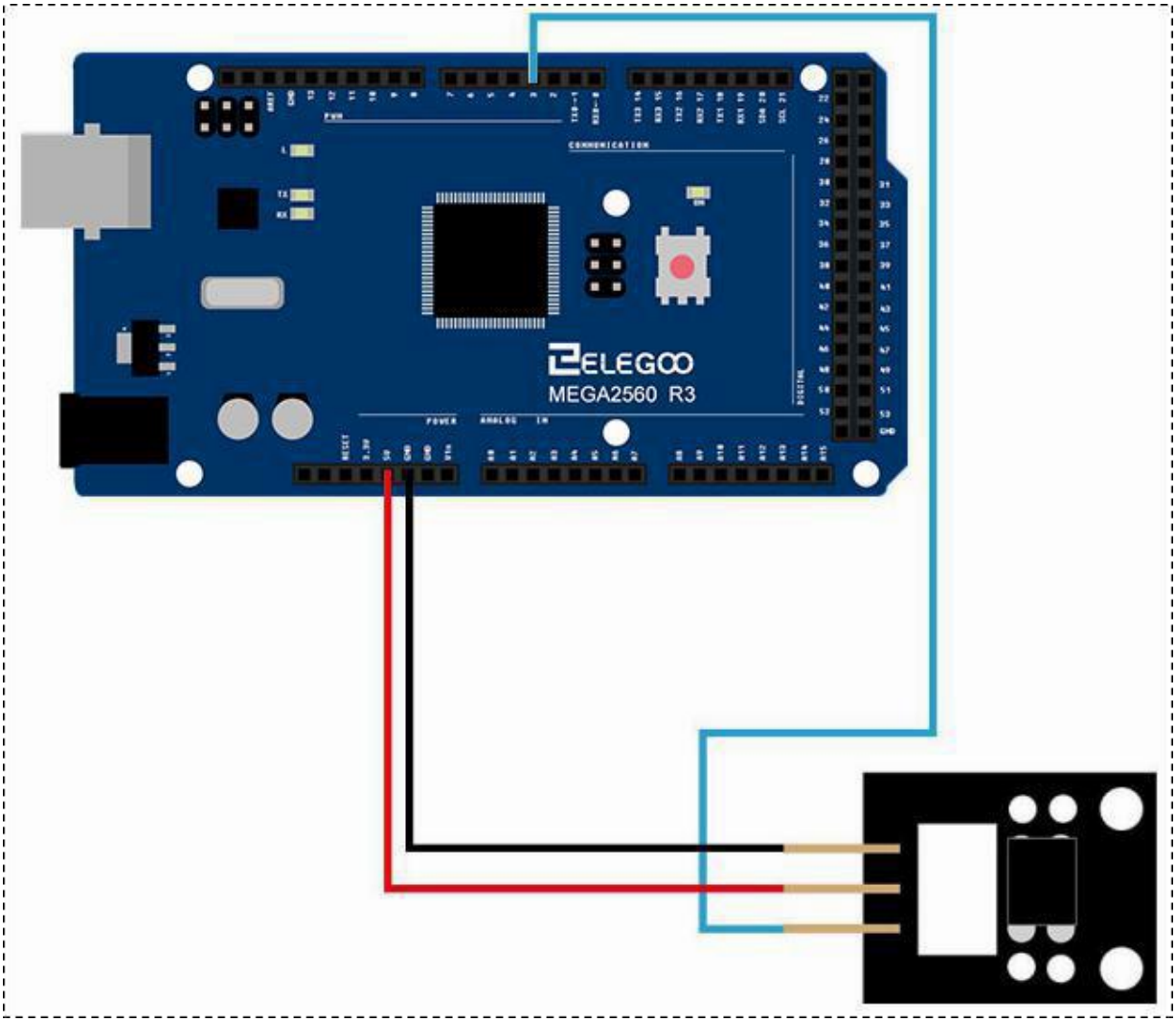
Verbindung

Schaltplan



Verdrahtungsplan



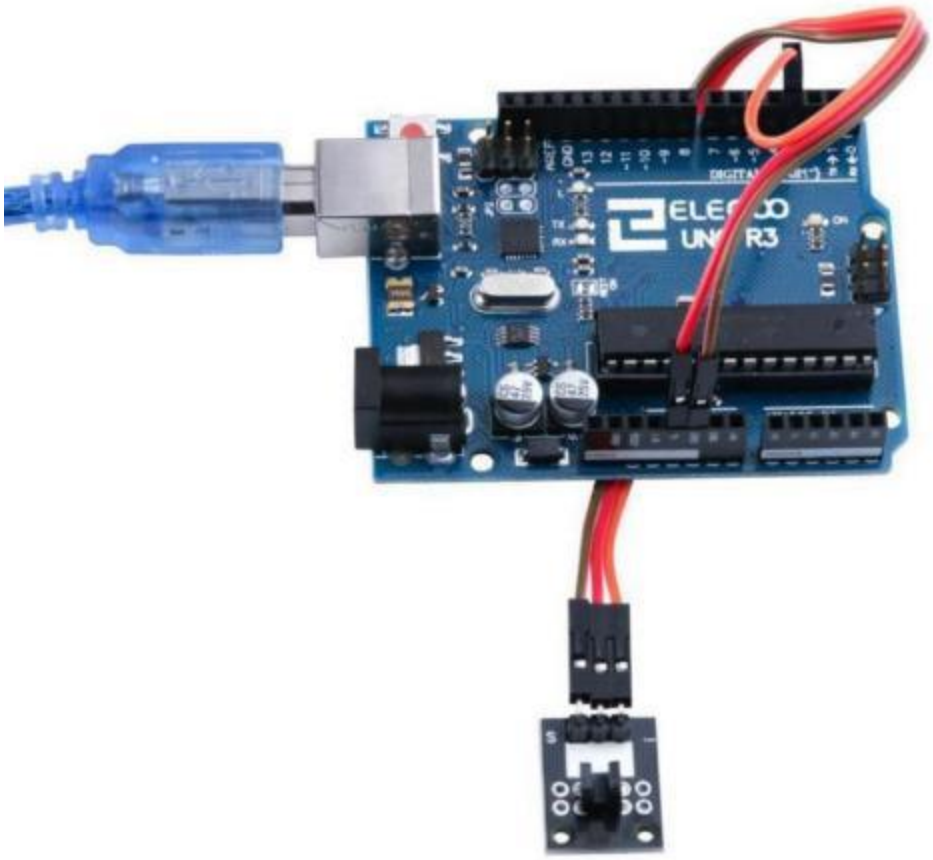


Code

Nachdem Sie die Verkabelung abgeschlossen haben, gehen Sie zu Code > Lektion 13 Photo-interrupter MODULE und laden Sie das Programm (die .ino-Datei) auf den Arduino.

Legen Sie ein Stück Papier in die Nut des Moduls und nehmen Sie es wieder weg. Sie sehen, dass die LED13 erlischt und wieder aufleuchtet.

Beispielbild



Im Folgenden finden Sie den Code und einige Erklärungen:

```
/* Die LED zur Anzeige liegt auf Pin 13. Das ist die eingebaut LED des Arduino */
 int Led=13;
 /* Das Gabellichtschrankenmodul wird an Pin 3 angeschlossen*/
 int buttonpin=3;
 int val;
 void    setup()
 {
 /* Die LED an Pin 13 als Ausgang definieren */
 pinMode(Led,OUTPUT);
 /* Den Pin 3, an dem die Gabellichtschranke angeschlossen ist, ist ein Eingang */
 pinMode(buttonpin,INPUT);
 }
 void    loop()
 {
 /* Wert einlesen, den die Gabellichtschranke ausgibt (0V falls die Lichtschranke unterbrochen ist, 5V
 wenn die Lichtschranke nicht unterbrochen ist*/
   val=digitalRead(buttonpin);
 if(val==HIGH)
 {
 / * LED ausschalten, wenn die Lichtschranke unterbrochen ist*/
 digitalWrite(Led,LOW);
 }
 else
 {
 / * LED anschalten, wenn die Lichtschranke nicht unterbrochen ist*/
 digitalWrite(Led,HIGH);
 }
 }
```

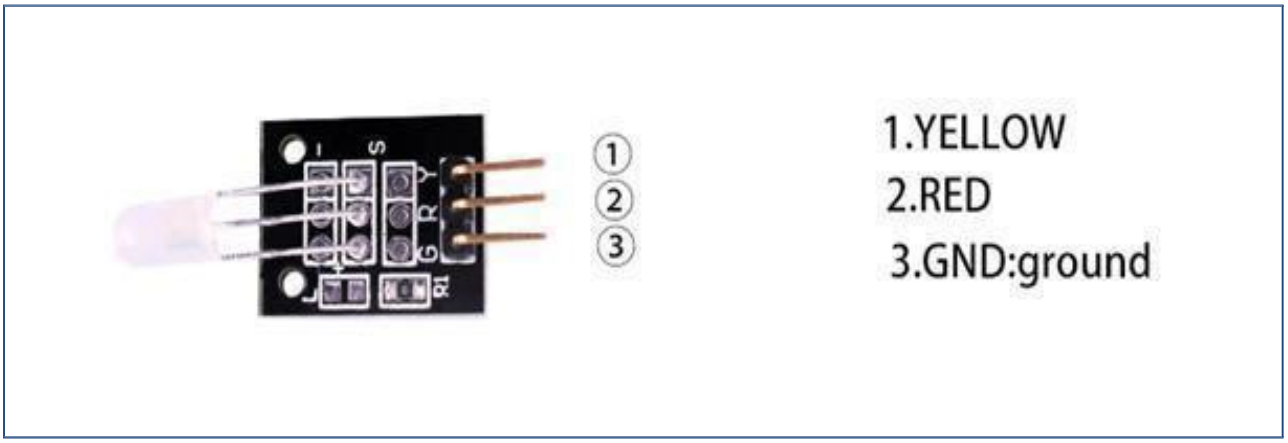
Lektion 14 ZWEIFARBIGE LED

Übersicht

In diesem Versuch lernen wir eine zweifarbige LED zu benutzen. Auf der Platine sitzt ein Bauteil, das intern eine rote und eine gelbe LED besitzt. Mit dem Arduino können wir beide LEDs unabhängig voneinander ansteuern. Da die beiden LEDs in einem gemeinsamen Gehäuse untergebracht sind, erscheinen sie wie eine LED, deren Farbe wir ändern können (zumindest im Farbspektrum von gelb und rot).

Zweifarbige LED 5mm

Die 5mm LED hat eine gemeinsame Kathode, die mit dem 'G' Pin der Platine verbunden ist. Der mittlere Pin ist mit der roten Anode und der ‚Y' Pin mit der gelben Anode verbunden.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

1x Zweifarbige LED mit gemeinsamer Kathode

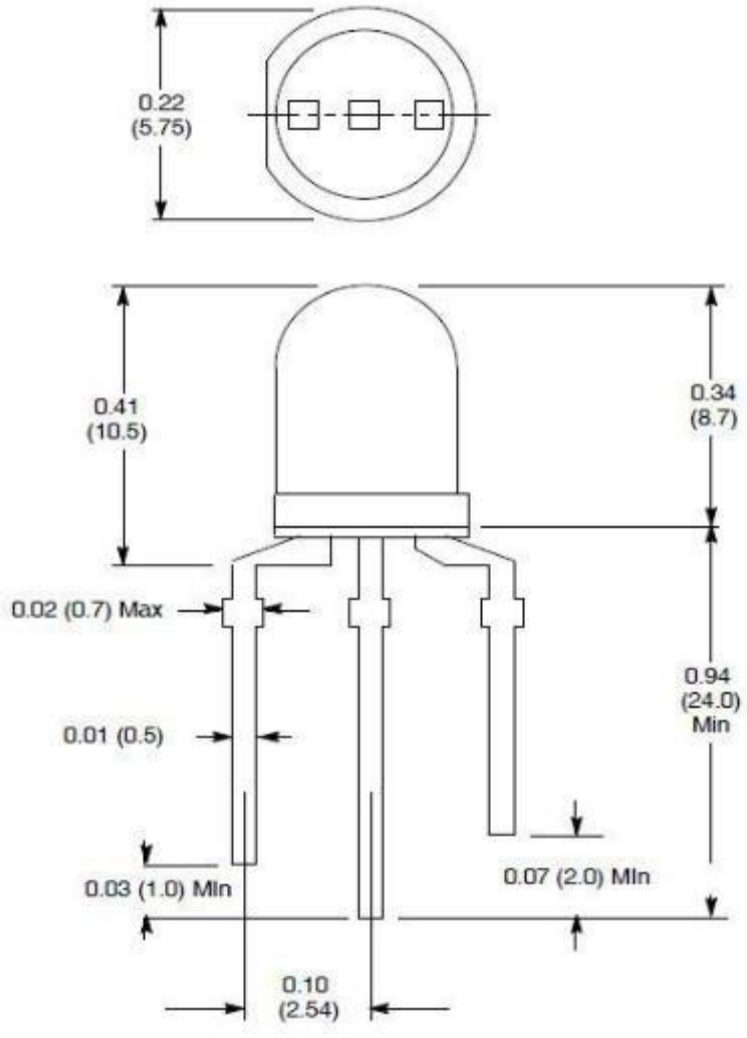
3x F-M Drähte

Komponenteneinführung

Zweifarbige LED mit gemeinsamer Kathode:

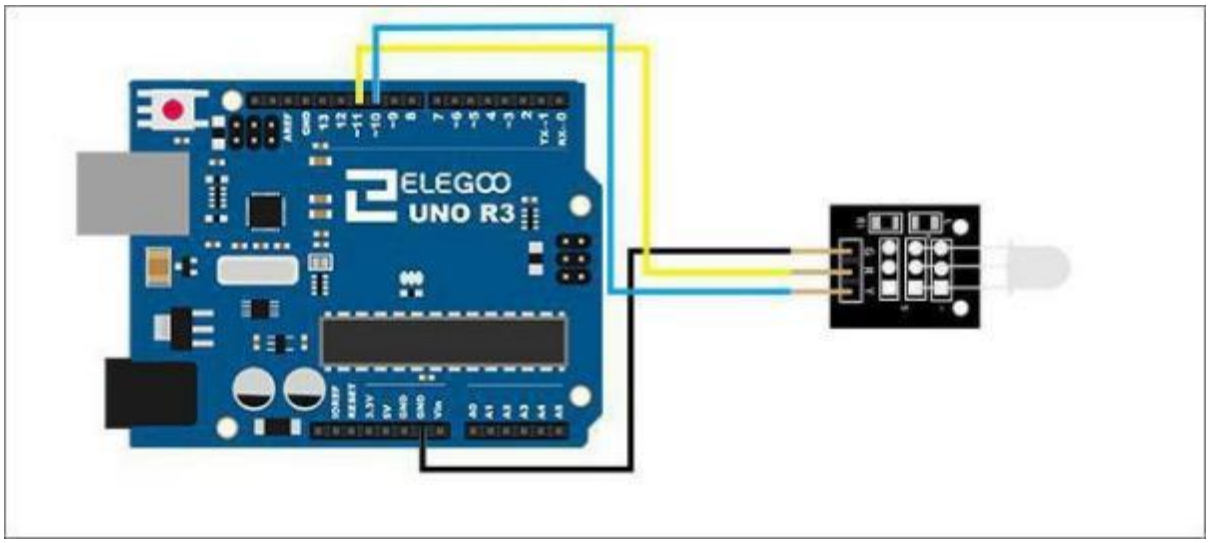
Electro-Optical Characteristics: (T_A = +25°C unless otherwise specified)

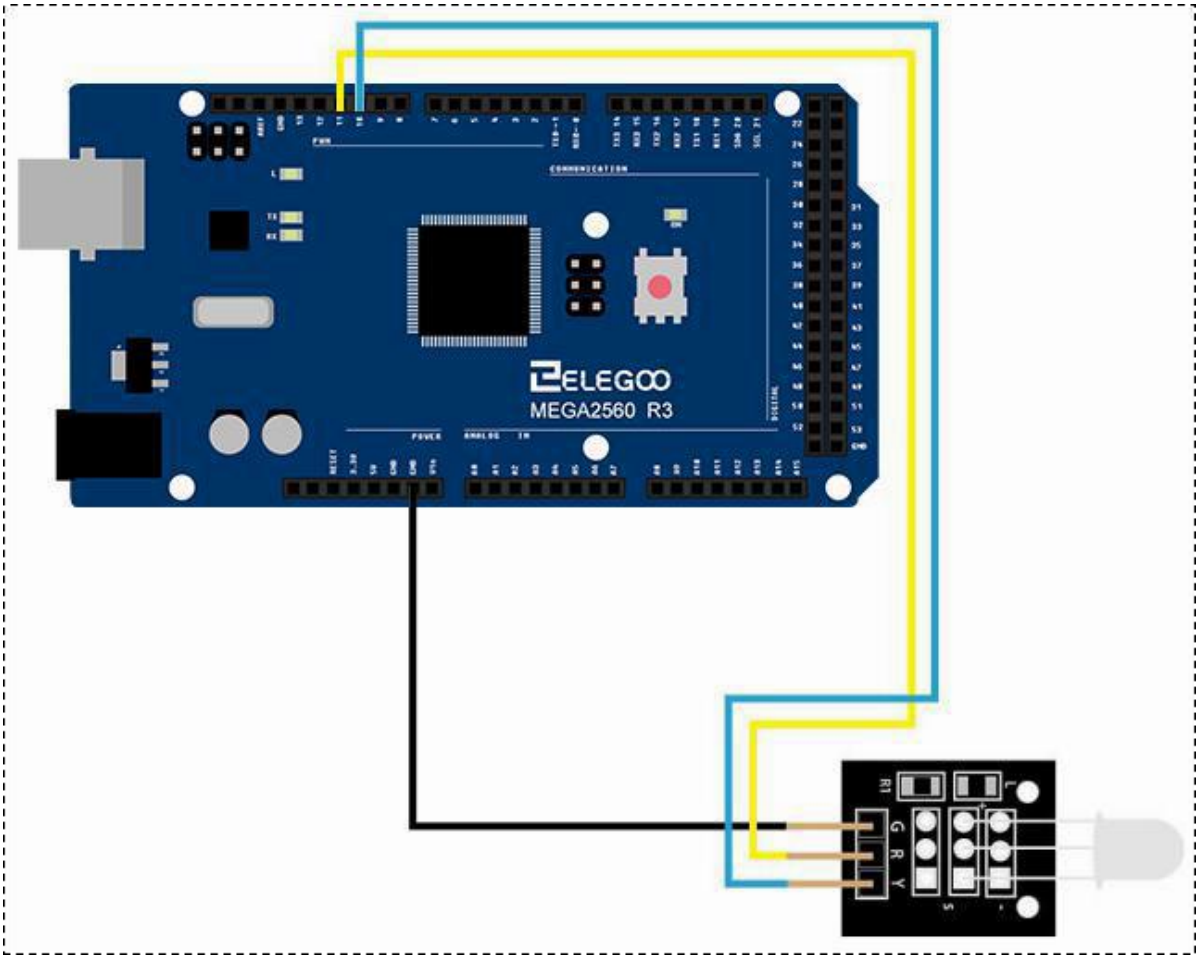
Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
View Angle of Half Power	2Θ1/2	IF = 20mA	–	40	–	deg
Forward Voltage High Efficiency Red	VF	IF = 20mA	–	2.05	2.80	V
Yellow-Green			–	2.15	2.80	V
Luminous Intensity (Note 1)	IV	IF = 20mA	35	60	–	mcd
Peak Emission Wavelength High Efficiency Red	λp	IF = 20mA	–	625	–	nm
Yellow-Green			–	570	–	nm
Dominate Wave Length (Note 2) High Efficiency Red	λd(HUE)	IF = 20mA	–	618	–	nm
Yellow-Green			–	567	–	nm



1. Red +
2. Common Lead –
3. Green +

Verdrahtungsplan



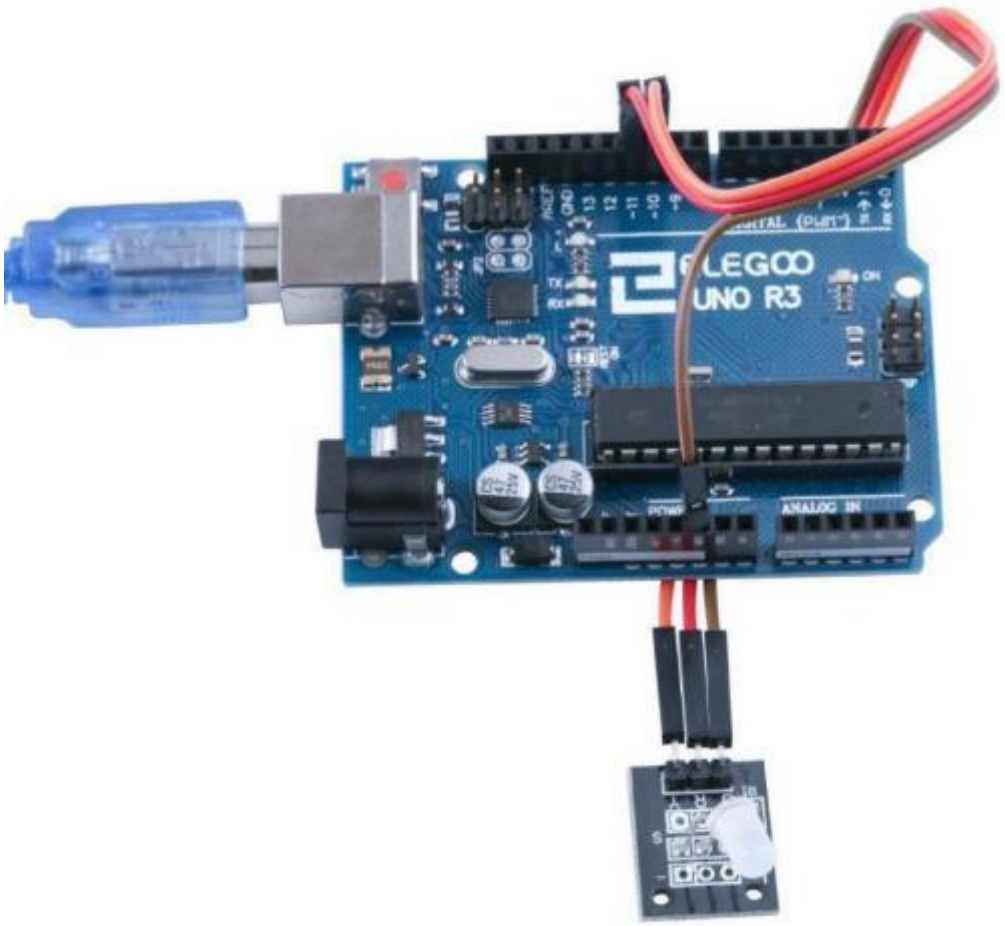


Code

Nachdem wir die Schaltung angeschlossen haben, laden wir das Programm "Lesson 14 Dual-color Common-Kathode LED", auf den Arduino.

Wir können sehen, dass das Modul seine Farbe kontinuierlich ändert. Wenn Sie die Farbe auf andere Weise ändern möchten, können Sie den Code anpassen. Er sollte leicht zu verstehen sein.

Beispielbild



Im Folgenden finden Sie den Code und einige Erklärungen.

```
/* Pin11 des Arduino wird mit der roten LED verbunden */
 int redpin =    11;
/* Pin10 des Arduino wird mit der gelben LED verbunden */
 int    yellowpin    =10;
 int    val;
 void setup()
 {
    pinMode(redpin, OUTPUT); /* Der Pin an dem die rote LED angeschlossen ist, ist ein Ausgang*/
    pinMode(yellowpin, OUTPUT); /* Der Pin an dem die gelbe LED angeschlossen ist, ist ein Ausgang*/
    Serial.begin(9600);
 }
 void loop()
 {
    for (val = 255;    val > 0;    val--) /*Von 255 bis 1 */
    {
        /* Die Helligkeit der beiden Farben ändert sich alle 15 ms, d.h. die Farbe mit der die LED leuchtet
 ändert sich ständig */
        analogWrite(11, val);
        analogWrite(10, 255 - val);
        delay(15);
    }
    for (val = 0; val < 255;    val++) /*Von 0 bis 254 */
    {
 /* Die Helligkeit der beiden Farben ändert sich alle 15 ms, d.h. die Farbe mit der die LED leuchtet ändert
 sich ständig */
        analogWrite(11, val);
        analogWrite(10, 255 - val);
       delay(15);
    }
 }
```

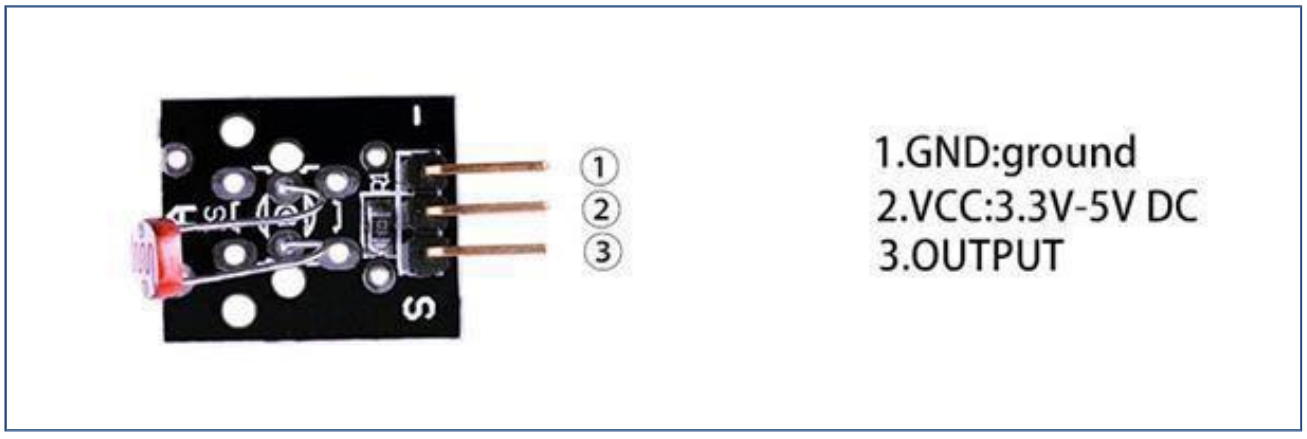
Lektion 15 Fotowiderstandsmodul

Überblick

In diesem Versuch lernen wir den Umgang mit einem Fotowiderstandsmodul. Fotowiderstände sind in unserem täglichen Leben sehr verbreitet. Sie werden hauptsächlich in intelligenten Schaltern verwendet, um etwas Komfort in unser Leben zu bringen.

Fotowiderstand

LDR (Lichtabhängiger Widerstand / Fotowiderstand / engl: Light Dependant Resistor). Widerstand bei Dunkelheit >20 Megaohm, bei Tageslicht < 80 Ohm. Die beiden äußeren Pins sind mit dem Fotowiderstand verbunden. Ein 10 Kiloohm Wiederstand ist zwischen den Pins 2 und 3 (‚S') angeschlossen auf unserem Modul. Dieser Widerstand erleichtert es uns Messchaltungen aufzubauen.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

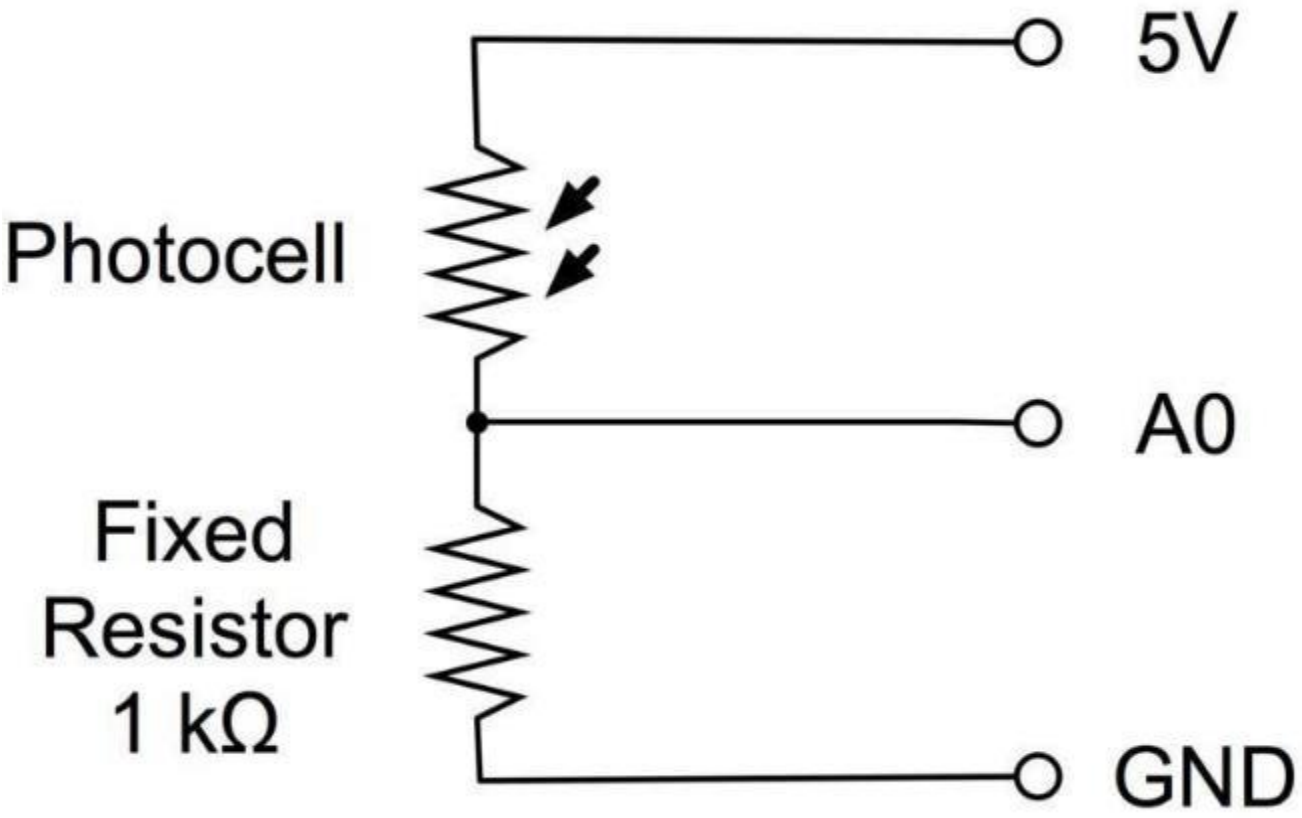
1 x Fotowiderstandsmodul

3 x F-M Drähte

Komponenteneinführung

Fotowiderstand:

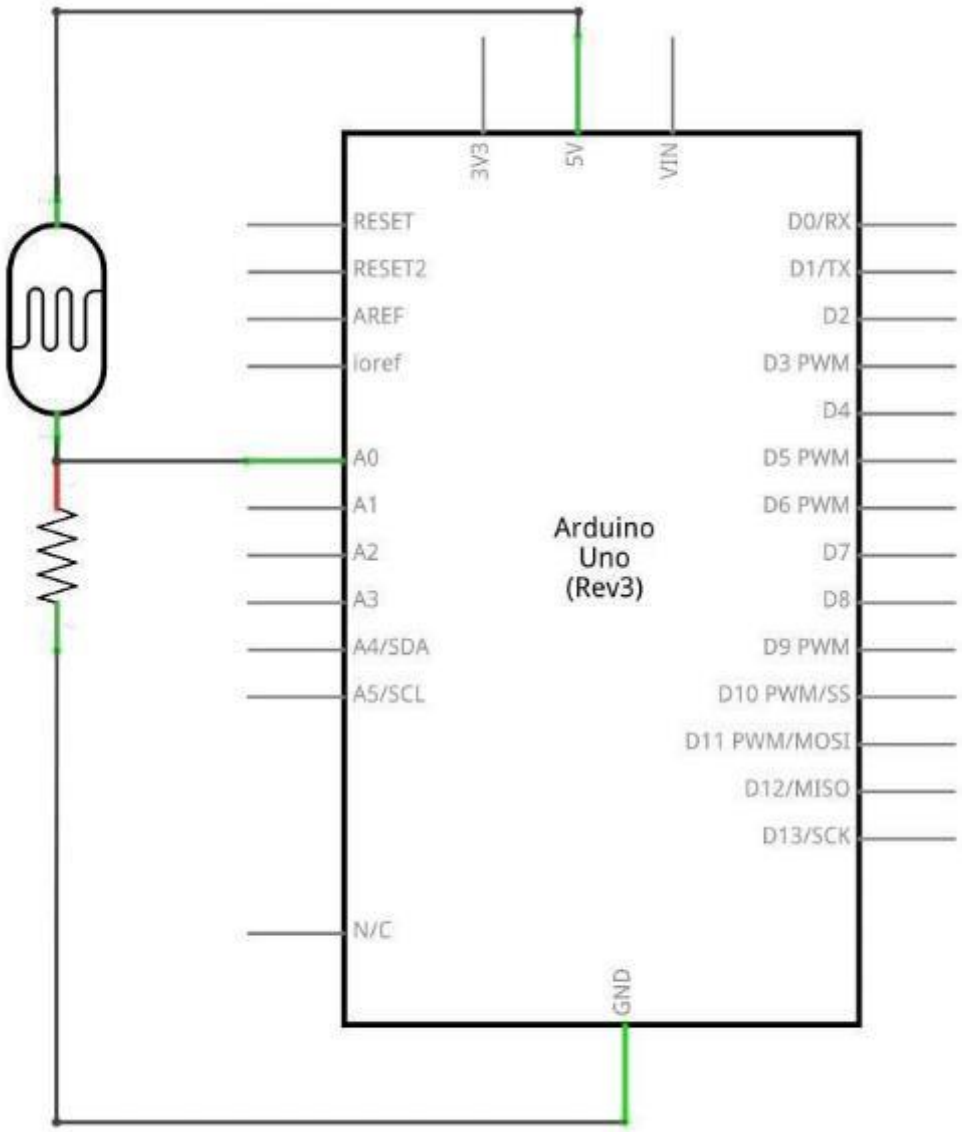
Ein Fotowiderstand wird auch Fotozelle oder lichtabhängiger Widerstand – auf englisch LDR – genannt. Wie der Name schon sagt, verhalten sich diese Komponenten wie ein Widerstand, nur dass sich der Widerstand in Abhängigkeit davon ändert, wie viel Licht auf sie fällt. Der Widerstand hat einen sehr hohen Widerstandswert bei Dunkelheit und einen sehr niedrigen bei starker Helligkeit. Der Widerstandswert gibt also an, wie wieviel Licht auf die Fotozelle fällt. Nun kann man Widerstände nicht direkt messen, sondern nur indirekt. Was wir allerdings messen können ist die Spannung, die an einem Widerstand anliegt, da diese proportional zum Widerstandswert ist. Am einfachsten baut man einen Spannungsteiler zwischen einem Festwiderstand und dem Fotowiderstand.



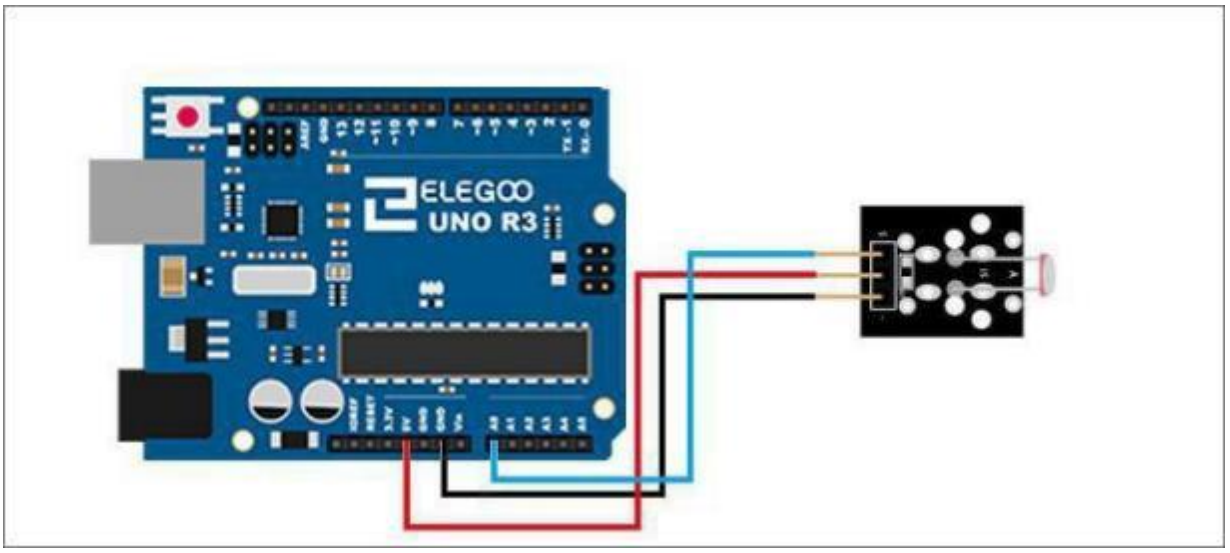
Wenn viel Licht auf den Fotowiderstand fällt, ist sein Widerstand klein im Verhältnis zum Festwiderstand. An dem kleineren Widerstand liegt weniger Spannung an, als am größeren. In der nachfolgenden Schaltung heißt das, dass wir bei großer Helligkeit am Arduino Pin A0 einen Spannungswert nahe 5V messen werden. Bei vollkommener Dunkelheit, liegt fast die gesamte Spannung am Fotowiderstand an und wir messen nahezu 0V.

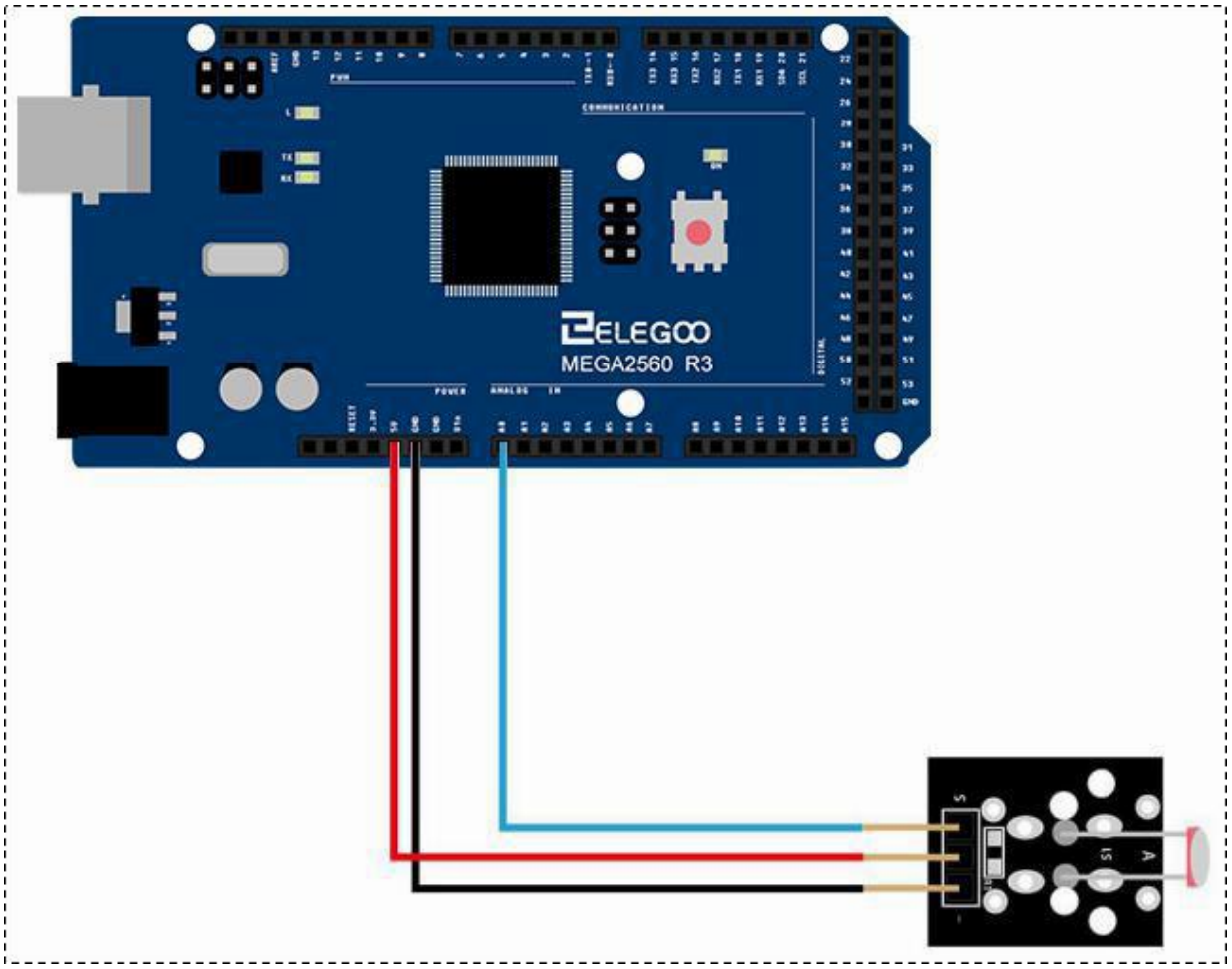
Verbindung

Schaltplan



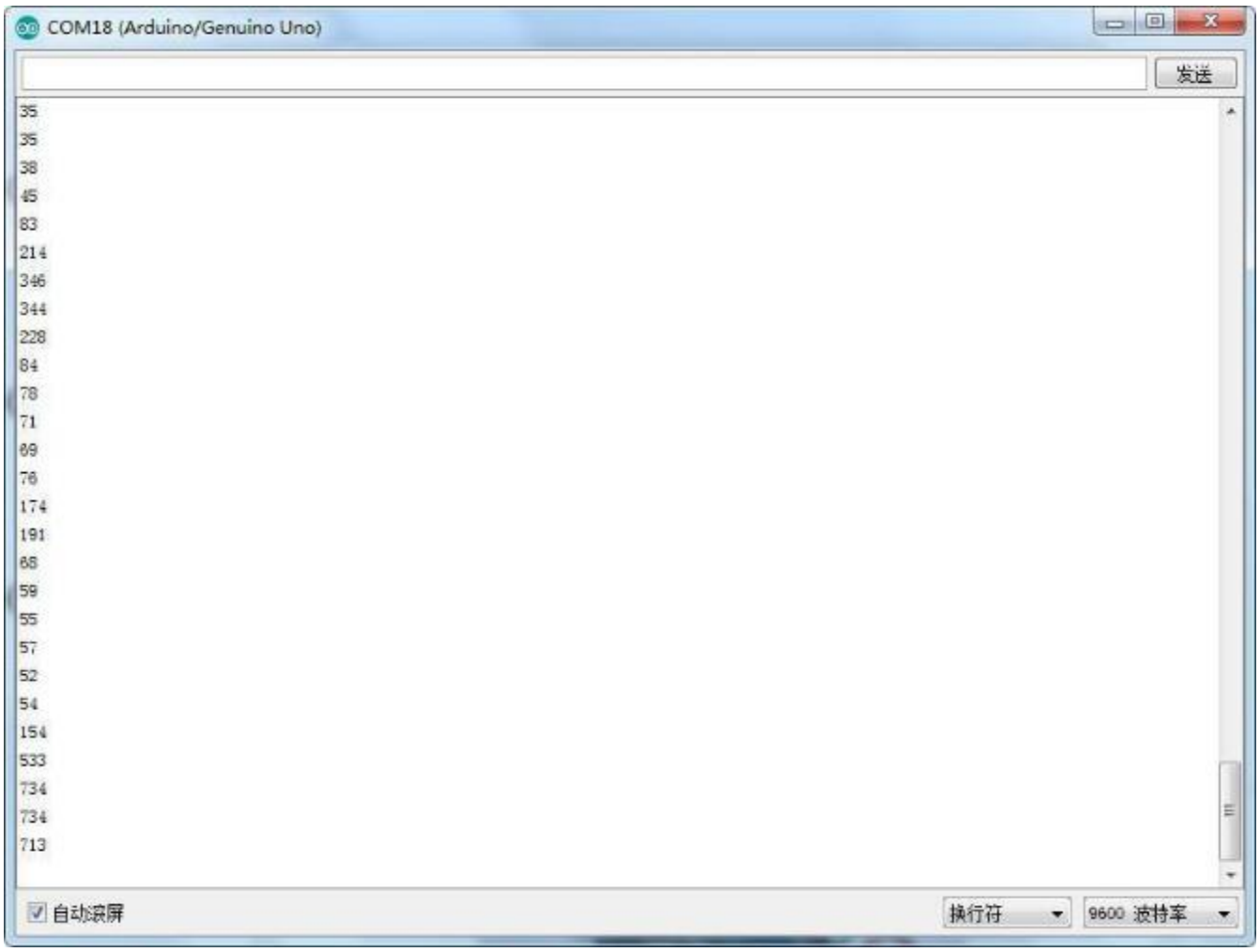
Verdrahtungsplan





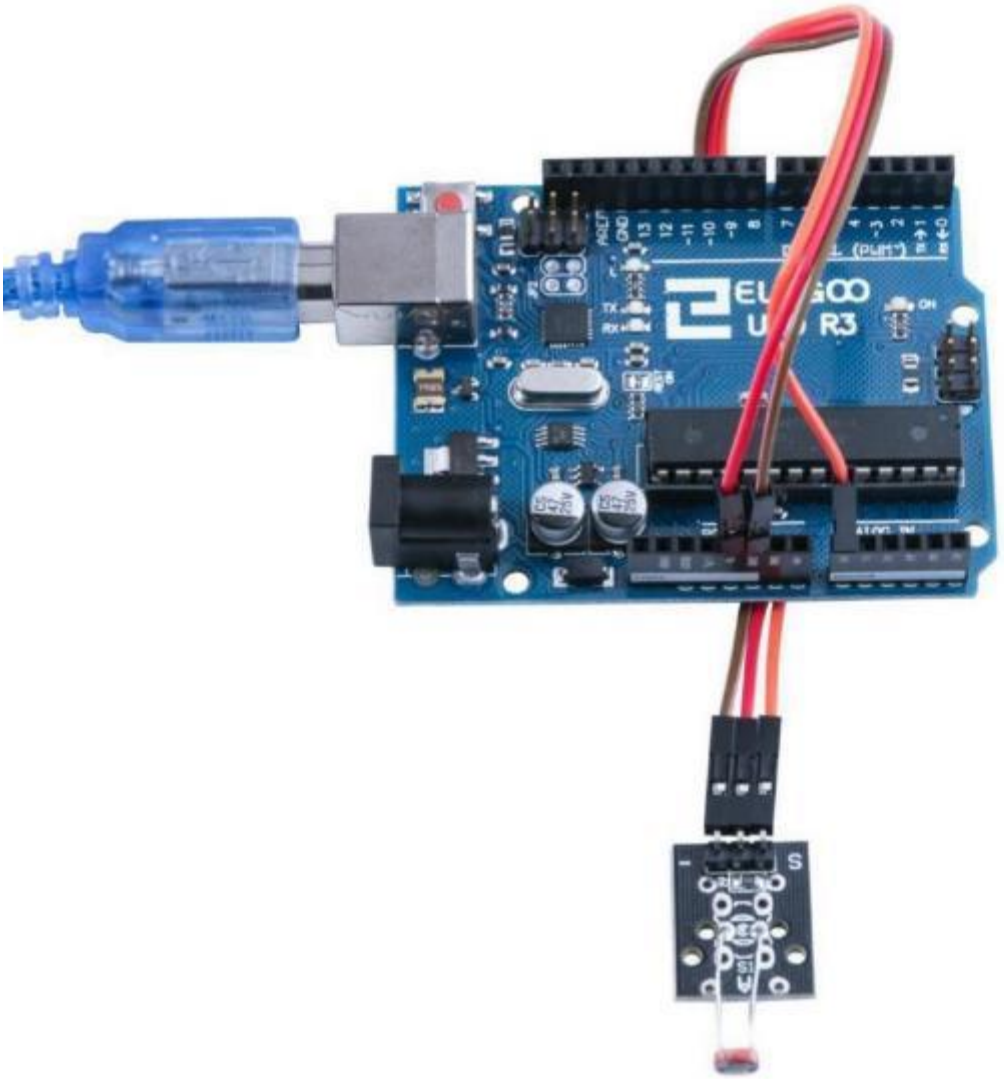
Code

Nachdem Sie die Verkabelung abgeschlossen haben, laden Sie bitte das Programm Code > Lektion 15 PHOTO RESISTOR MODULE auf den Arduino. Sehen Sie in Lektion 2 nach, wie Sie den seriellen Monitor öffnen können, falls Sie es vergessen haben.



Bitte beachten Sie, dass der Bereich 0 – 5V hier abgebildet wird auf die Werte 0 bis 1023 (2 hoch 10, da der Arduino einen 10 Bit Analog-Digital Wandler an Board hat)

Beispielbild



Im Folgenden finden Sie den Code und ein paar Erklärungen

```
int sensorPin    =    A0;       /* Wir messen die Spannung an Pin A0 */
int    ledPin    =    13;      /* Zur Ausgabe benutzen wir die eingebaute LED an Pin 13 */
int sensorValue =    0;
void setup()
{
        pinMode(ledPin,OUTPUT); /*Die eingebaute LED ist ein Ausgang*/
        Serial.begin(9600);
}
void loop()
{
/* Spannungswert messen zwischen Fotowiderstand und Festwiderstand*/
sensorValue = analogRead(sensorPin);
digitalWrite(ledPin, HIGH); /*LED einschalten*/
/*Warten. Die Wartezeit ist proportional zur gemessen Spannung; zwischen 0 und 1023
Millisekunden */
delay(sensorValue);

digitalWrite(ledPin, LOW); /*LED ausschalten*/
/*Warten. Die Wartezeit ist proportional zur gemessen Spannung; zwischen 0 und 1023
Millisekunden */
delay(sensorValue);
Serial.println(sensorValue,    DEC);
}
```

Anmerkung: Wir haben also eine Blinkschaltung gebaut. Bei Dunkelheit (oder wenn wir einen Finger auf den Fotowiderstand halten) blinkt die eingebaute LED sehr schnell (eventuell zu schnell für das menschliche Auge). Bei Helligkeit blinkt sie sehr langsam.

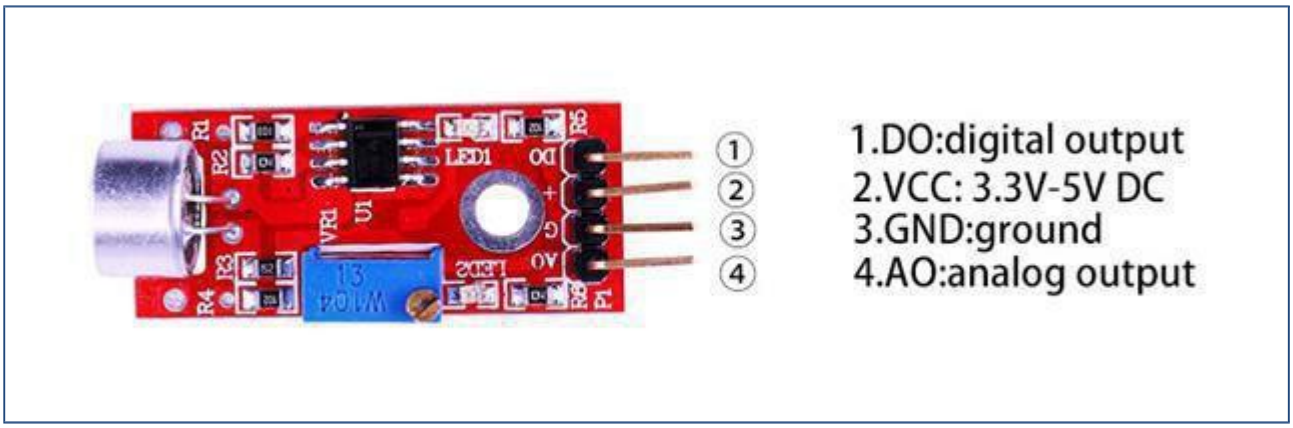
Lektion 16 KLEINES UND GROSSES SCHALLWANDLERMODUL

Überblick

In diesem Versuch lernen wir den Umgang mit Schallwandlermodulen.

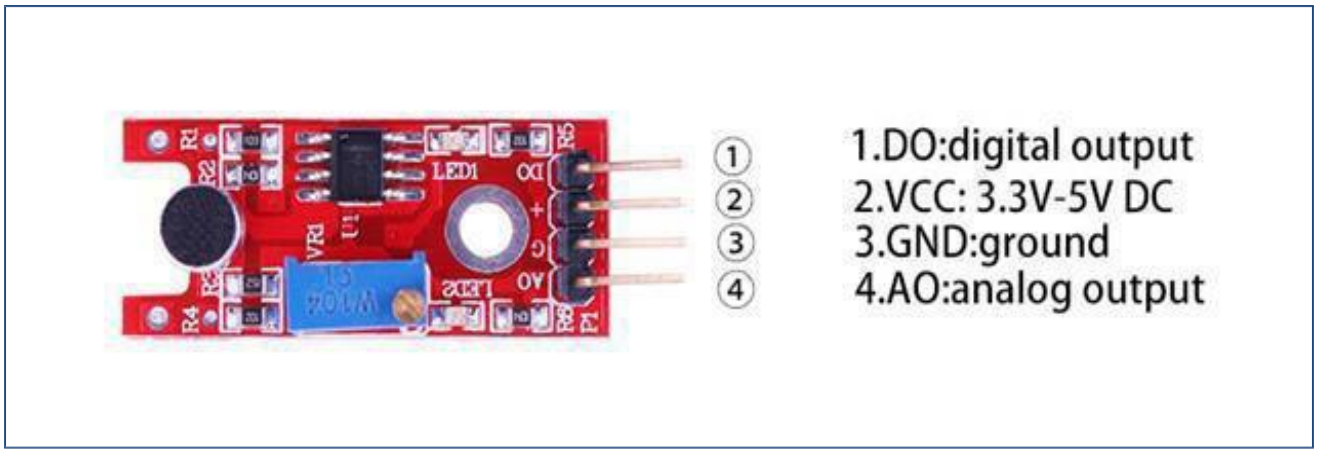
Modul mit großem Mikrofon

Ein Schallwandlermodul mit einer großen Mikrofonkapsel. Es hat sowohl einen digitalen, als auch einen analogen Ausgang. Der digitale Ausgang schaltet um, wenn der Schall ein bestimmtes Level erreicht. Das Schwelllevel kann mittels eines Potentiometers eingestellt werden. Der analoge Ausgang gibt ein gepuffertes Signal aus, dass proportional zum empfangenen Schall ist.



Modul mit kleinem Mikrofon

Abgesehen von der geringeren Größe des Mikrofons und seiner geringeren Empfindlichkeit ist das Modul identisch zu dem Modul mit der größeren Mikrofonkapsel.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

1 x großes Schallsensormodul

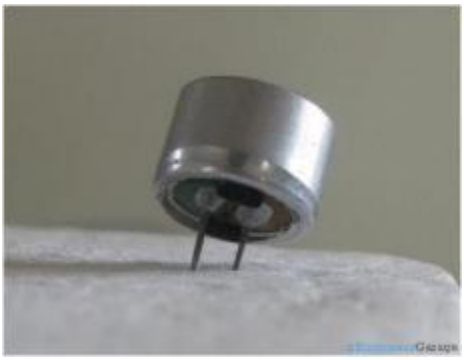
1 x kleines Schallsensormodul

4 x F-M Drähte

Komponenteneinführung

Mikrofon

Wandler sind Geräte, die Energie von einer Form in eine andere umwandeln. Ein Mikrofon ist ein Wandler, der Schallenergie in elektrische Signale umwandelt. Es funktioniert genau andersherum als ein Lautsprecher (ein Lautsprecher wandelt elektrische Energie in Schallenergie). Mikrofone sind in verschiedenen Formen und Größen erhältlich. Je nach Anwendung kann ein Mikrofon verschiedene Technologien verwenden, um Schall in elektrische Signale umzuwandeln. Hier verwenden wir ein Elektret-Kondensatormikrofon, das in Mobiltelefonen, Laptops usw. weit verbreitet ist. Der Marktanteil der Elektrokondensatormikrofone an den Mikrofonen liegt bei ca. 90%. Wie der Name schon sagt, ist das Elektretkondensatormikrofon ähnlich einem Plattenkondensator aufgebaut. Es funktioniert nach dem Prinzip einer veränderlichen Kapazität. Es besteht aus zwei Platten von denen eine fest angebracht ist und die andere beweglich aufgehangen ist. Wenn Schallwellen auf die bewegliche Membran treffen, bewegt sich diese und die Kapazität (die zum Abstand der Platten proportional ist) verändert sich analog zu den Schallwellen.



Diese Mikrofone werden häufig in elektronischen Schaltungen eingesetzt, um kleine Geräusche oder Luftschwingungen zu erkennen, die wiederum in elektrische Signale für die weitere Verwendung umgewandelt werden. Die beiden Beinchen, wie im Bild oben gezeigt, dienen zur elektrischen Verbindung mit der Schaltung.

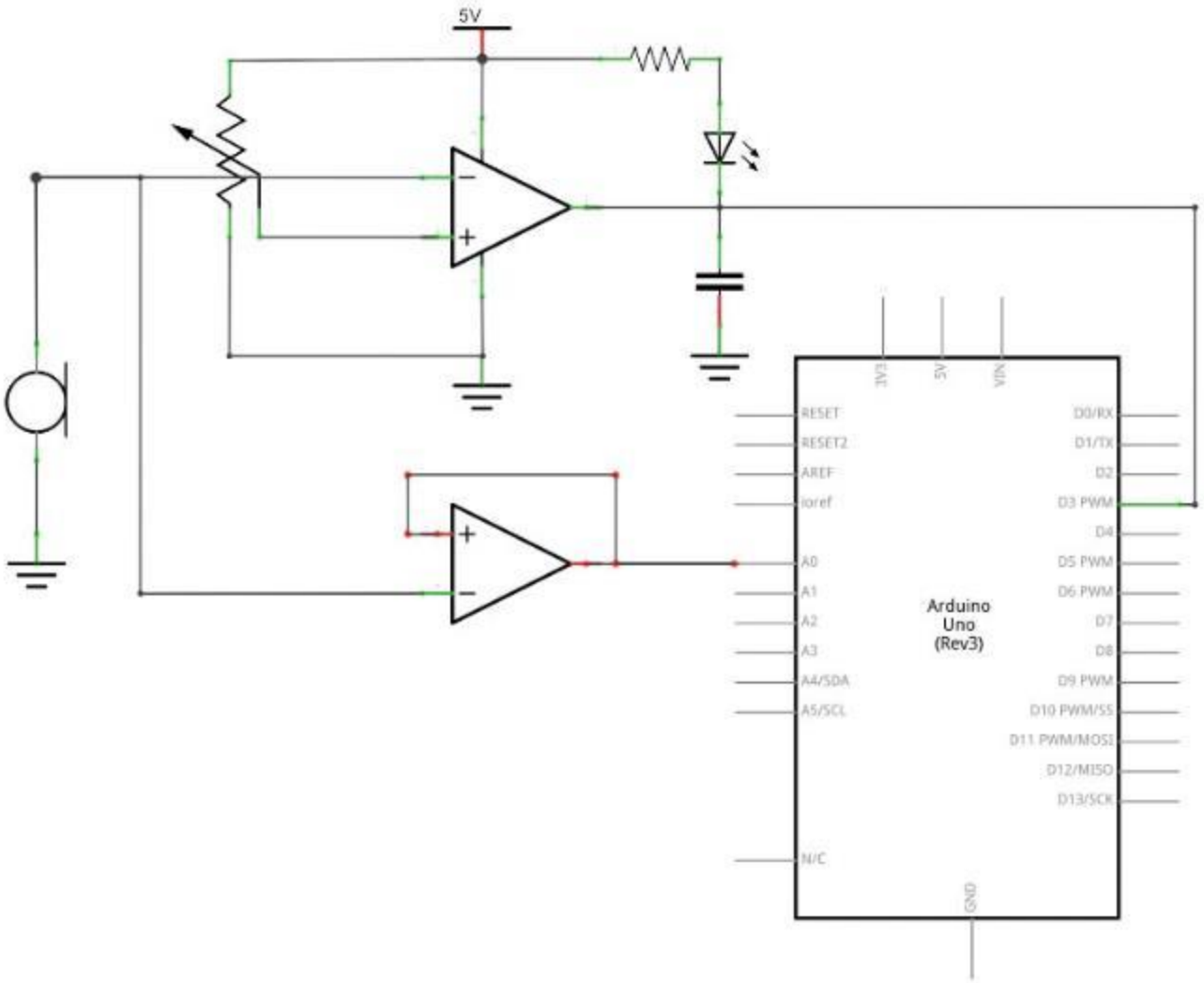
Ein leitfähiger Metallkörper umschließt die inneren Teile des Mikrofons. Auf die Oberseite wird ein poröses Material geklebt. Es dient als Filter für Staubpartikel. Die Schallsignale/Luftschwingungen durchdringen das poröse Material und treffen auf die Membran im Inneren.

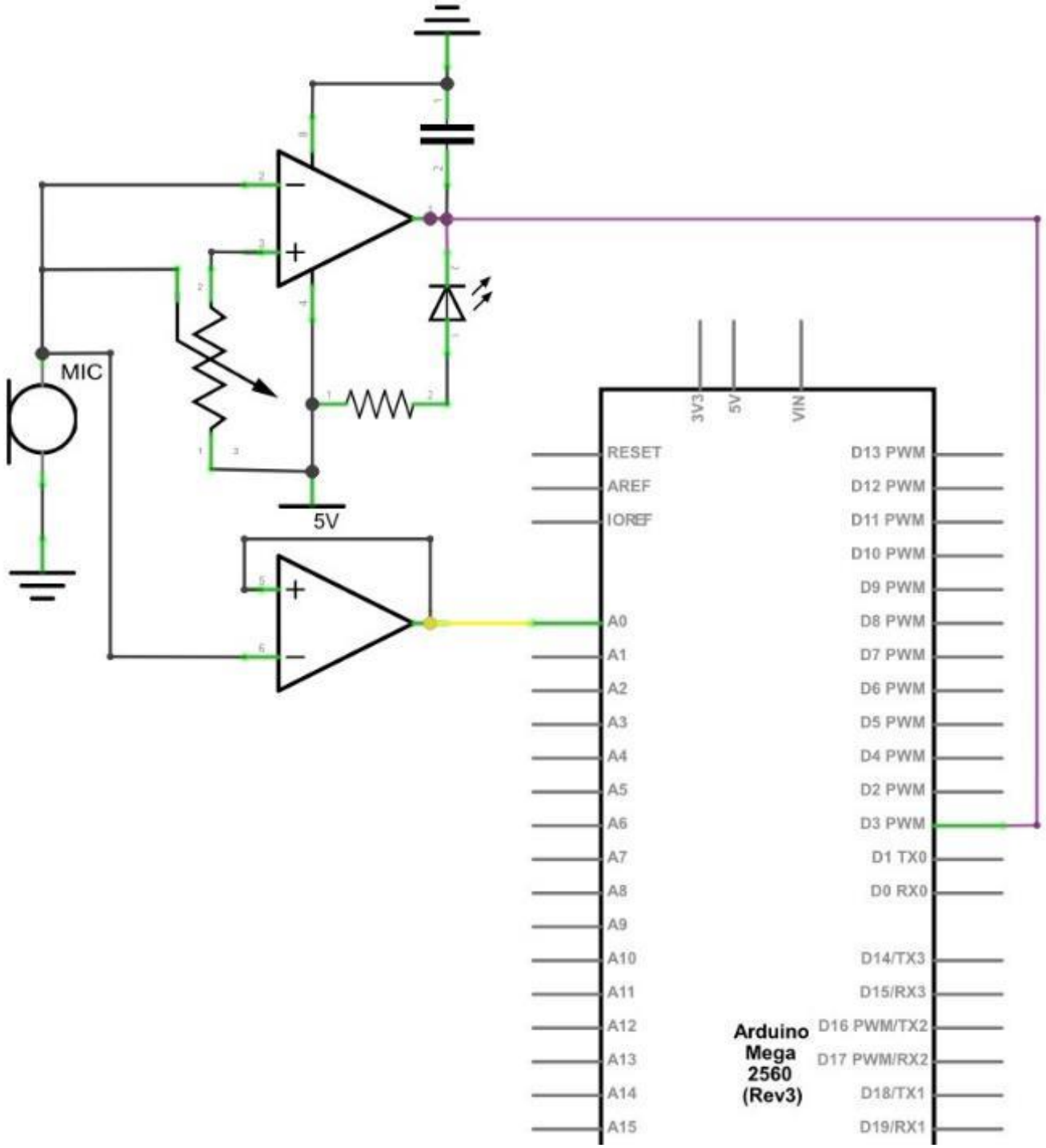
Schallsensormodul:

Das Schallsensormodul bietet eine einfache Möglichkeit, Schall zu erkennen und wird in der Regel zur Erfassung der Schallintensität verwendet. Dieses Modul kann für Sicherheits-, Schalt- und Überwachungsanwendungen eingesetzt werden. Seine Empfindlichkeit lässt sich einfach und bequem einstellen. Es verwendet ein Mikrofon, das das Eingangssignal zu einem Verstärker, einem Peak-Detektor (Spitzenwertdetektor) und einem Puffer liefert. Wenn der Sensor einen Ton erkennt, erzeugt das Modul eine Ausgangsspannung, die an einen Mikrocontroller gesendet wird, der dann die notwendige Verarbeitung durchführt.

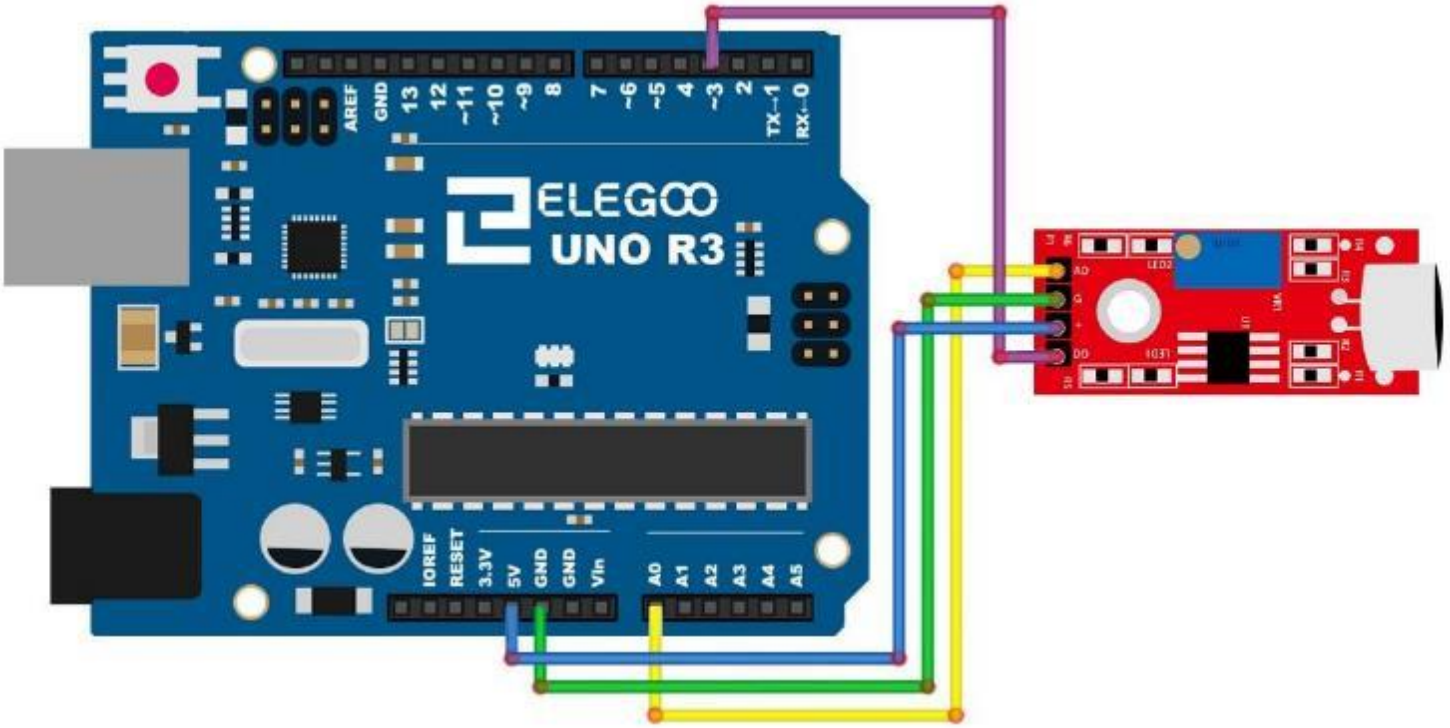
Verbindung

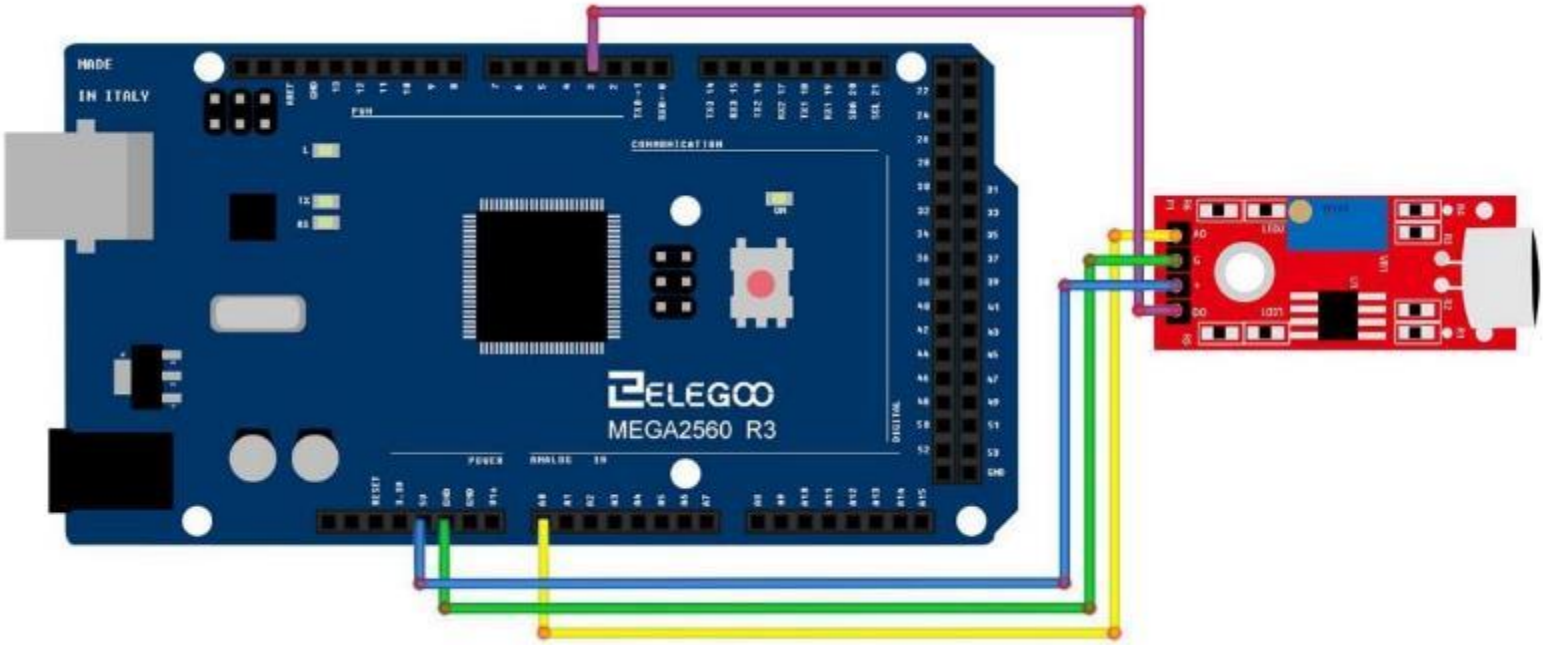
Schaltplan




MIC
5V
3V3
5V
VIN
RESET
AREF
IOREF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
Arduino
Mega
2560
(Rev3)

Verdrahtungsplan





Code

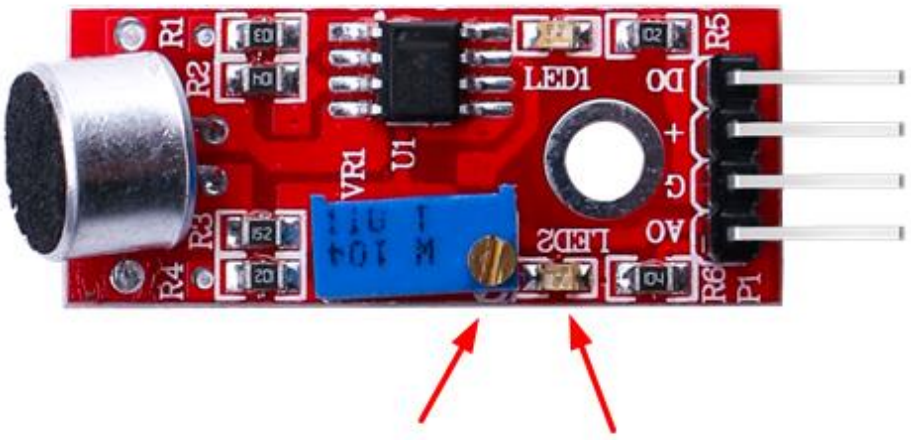
Nach der Verdrahtung laden Sie bitte das Programm（Lesson 17 BIG SOUND SENSOR MODULE AND SMALL SOUND SENSOR MODULE）auf Ihren Arduino.

Das Modul hat zwei Ausgänge:

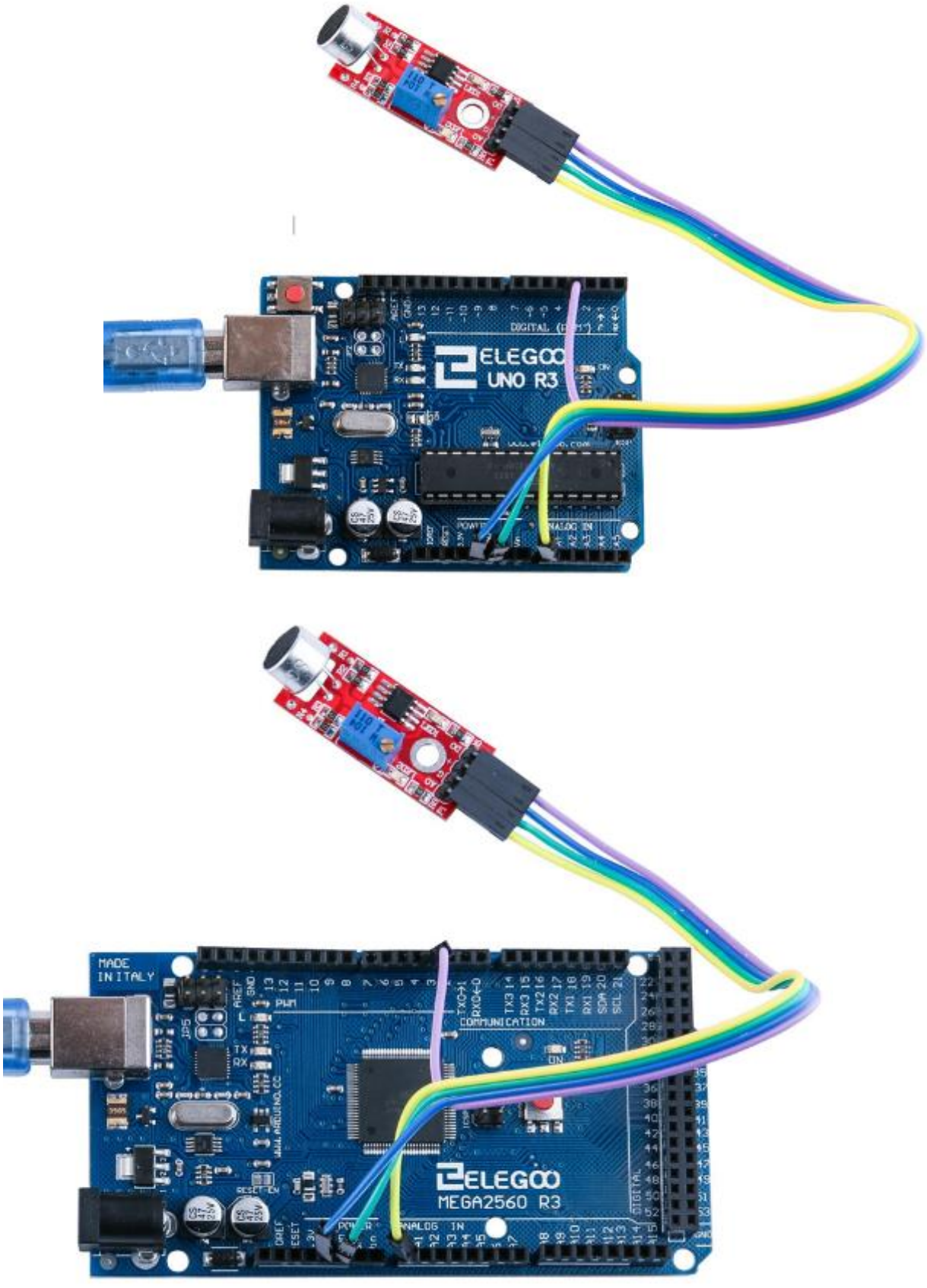
AO, analoger Ausgang, er liefert ein Echtzeit-Ausgangsspannungssignal des Mikrofons. Wie im Schaltplan zu sehen ist, ist das Signal gepuffert. Wir werden im Weiteren den analogen Ausgang verwenden um den Schall auf dem seriellen Plotter des Arduino sichtbar zu machen.

DO, Digitaler Ausgang, wenn die Intensität des Schalls einen bestimmten, einstellbaren Schwellwert erreicht, wird der Ausgang ‚high', ansonsten ist er ‚low'. Es handelt sich hier also um einen Komparatorausgang, wobei der Schwellwert über ein Potentiometer einstellbar ist.

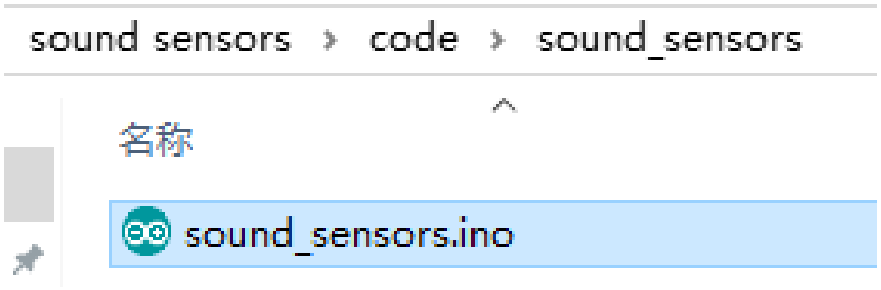
Bitte beachten Sie, dass Sie das 10k-Potentiometer mit dem Schraubendreher durch Drehen der Schraube gegen den Uhrzeigersinn einstellen müssen, bis die LED2 erlischt.

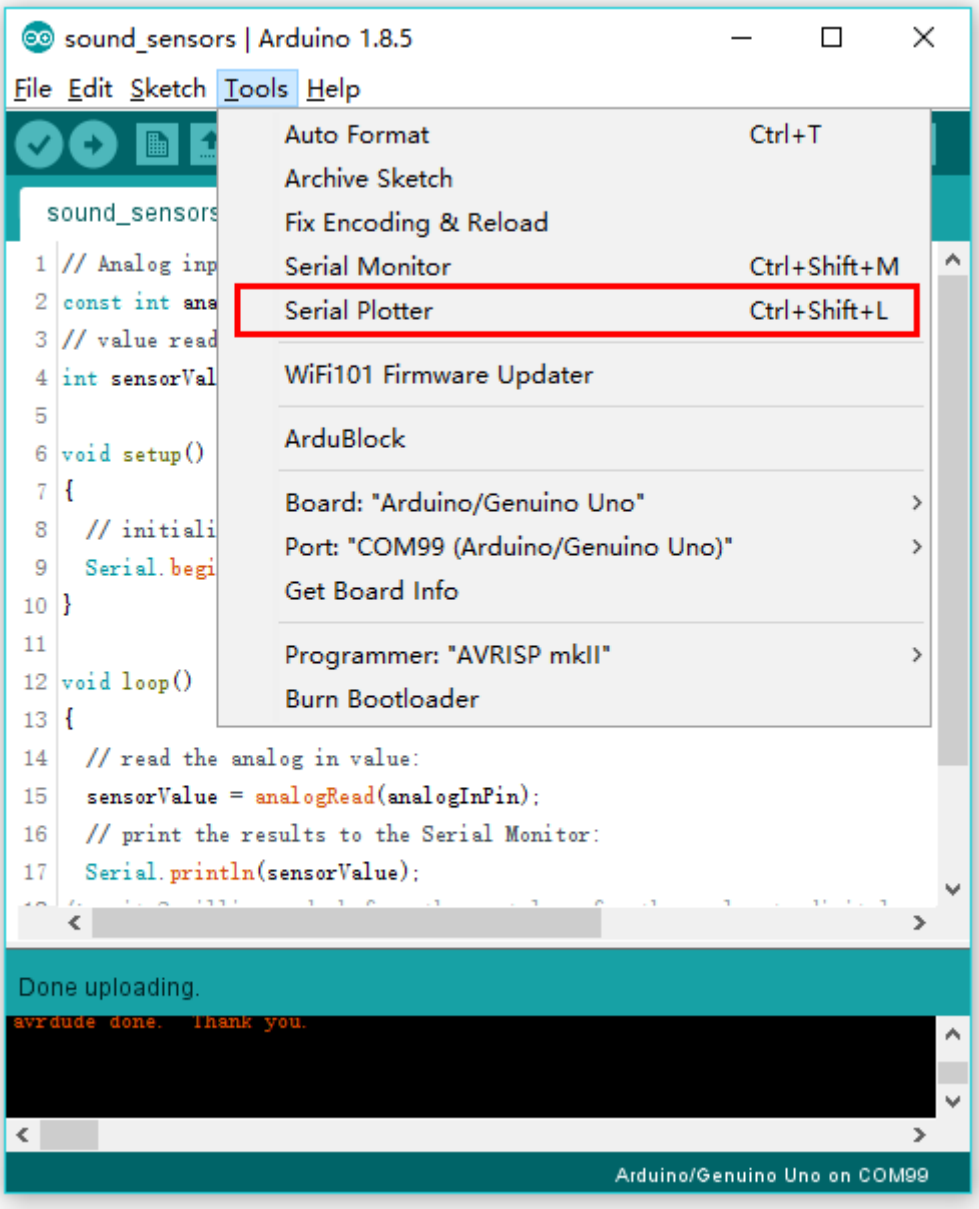


Beispielbild

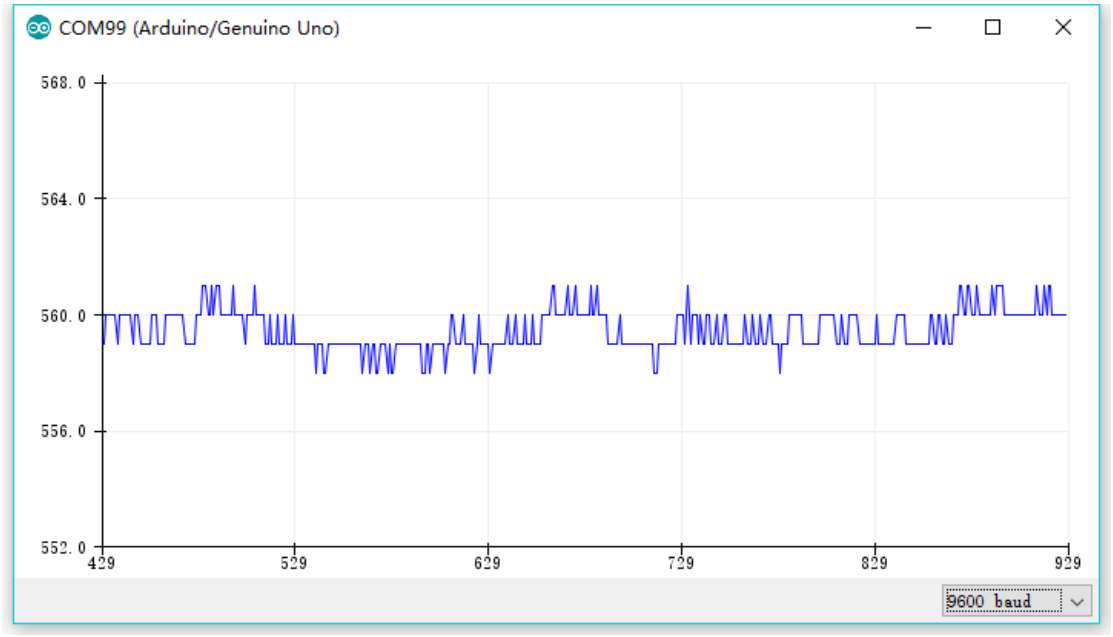


Nach dem Laden des Arduino klicken Sie in der Entwicklungsumgebung auf „Werkzeuge“, „Serieller Plotter“. Der nachfolgende Screenshot stammt von einem englischen Betriebssystem.

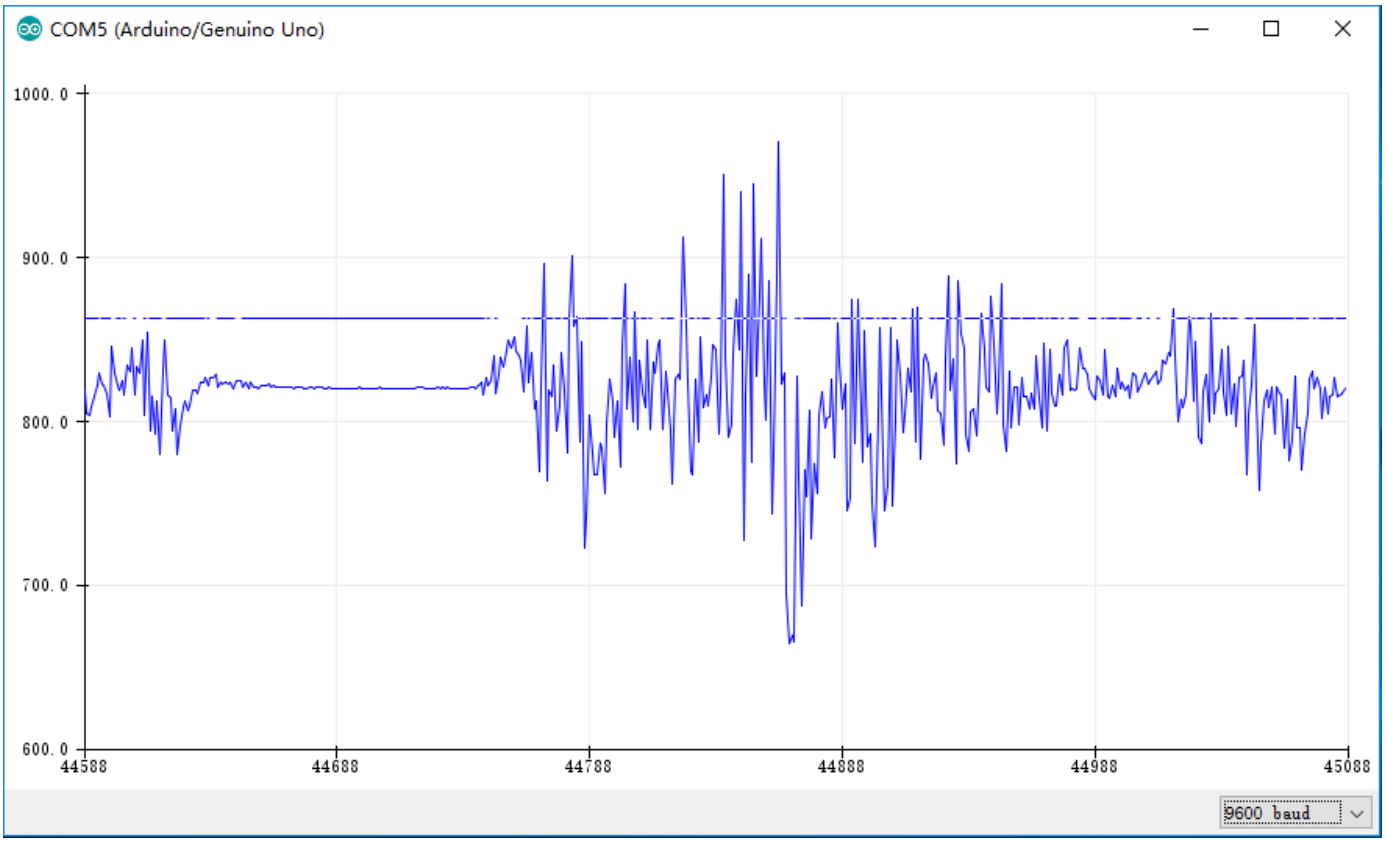




Bei Ruhe sehen wir ein Diagramm, dass dem folgenden ähnlich sein sollte:



Wenn Sie ins Mikrofon sprechen oder pusten, sollte das Diagramm so ähnlich aussehen.



Im Folgenden finden Sie den Code und einige Erklärungen dazu:

```
// Der analoge Ausgang des Moduls ist mit dem Pin A0 des Arduino verbunden
const int analogInPin = A0;
int sensorValue = 0;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  // Wir lesen den analogen Wert vom Schallmodul ein. Bitte beachten: Der Wertebereich geht von 0 bis 1023
  // Im seriellen Plotter wird der Wert auf der vertikalen Y-Achse dargestellt
  sensorValue = analogRead(analogInPin);
  Serial.println(sensorValue);
  // Nur eine Messung pro 2 ms um dem Analog/Digital Wandler Zeit zu geben
  delay(2);
}
```

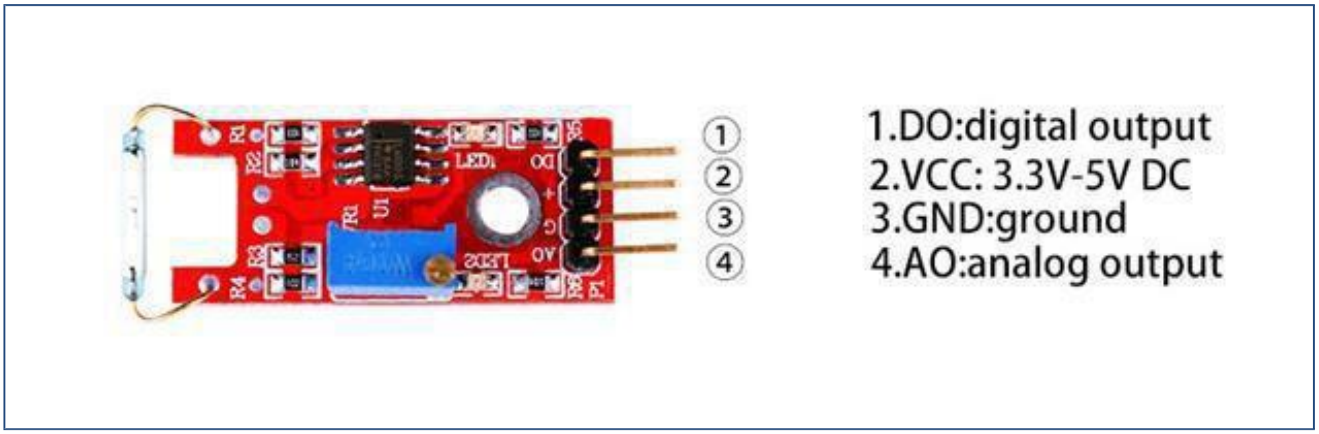
Lektion 17 REEDSCHALTERMODUL

Überblick

In diesem Versuch lernen wir, wie man mit einen Reedschalter nutzen kann.

Reedschalter

Dieser Reedschalter bietet sowohl eine analoge als auch eine digitale Schnittstelle. Ein aufgelötetes Potentiometer wird als Pull-Up Widerstand verwendet. Sie können durch Verstellen des Potentiometers die Schwelle einstellen, ab wann das Modul schaltet (also sich öffnet oder schließt). Dies ist nur relevant für den digitalen Ausgang. Der gepufferte Analogausgang wird dadurch nicht beeinflusst.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

1x Reedschaltermodul

4 x F-M Drähte

Komponenteneinführung

Reedschalter und Reedschalteraktivierung:

Obwohl ein Reedschalter prinzipiell auch durch Einlegen in eine elektrische Spule aktiviert werden kann, werden viele Reedschalter und Reedsensoren zur Näherungserkennung verwendet und durch einen Magneten aktiviert. Wenn der Magnet in die Nähe des Reed-Sensors/Schalters gebracht wird, wird dieser Sensor/Schalter aktiviert. Wenn der Magnet aus der Nähe des Reed-Sensors/Schalters entfernt wird, schaltet sich dieser wieder ab. Was im inneren des Schalters vor sich geht bei der Ansteuerung ist jedoch nicht offensichtlich. Eine Möglichkeit sich dies vorzustellen ist, dass der Magnet magnetische Pole in die Metallteile des Reedschalters induziert und die daraus resultierende Anziehung zwischen den elektrischen Kontakten den Reedschalter aktiviert.

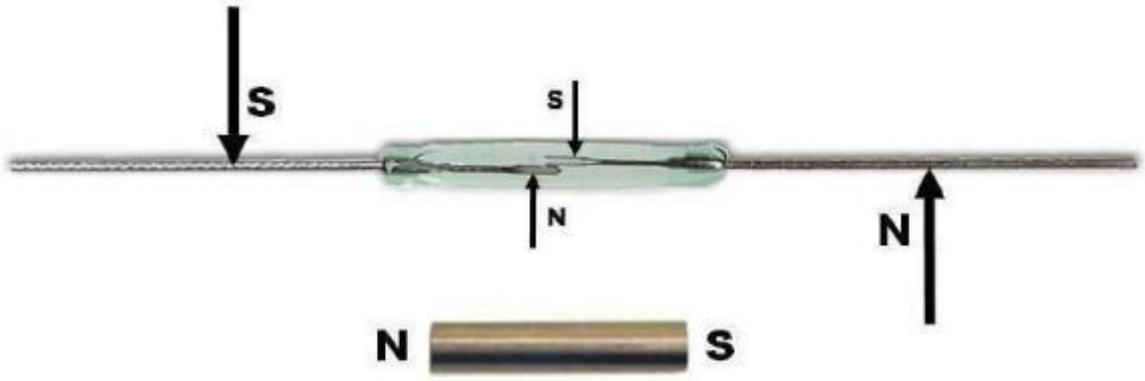


Figure 1 – Magnetic Induction

Eine weitere gleichwertige Denkweise über die Wechselwirkung zwischen einem Magneten und einem Reedschalter lautet, dass der Magnet magnetischen Fluss durch die elektrischen Kontakte induziert. Wenn der Magnetfluss hoch genug ist, bewirkt die magnetische Anziehung zwischen den Kontakten das Schließen des Reedschalters.

Figure 2 – Magnetic Flux

Unterschied zwischen dem Reedschaltermodul und dem Mini-Reedschaltermodul

Das große Modul kann auf zwei Arten Werte ausgeben: digital und analog. Das Minimodul kann nur digitale Werte ausgeben.

Prinzip

Die nachfolgenden Bilder zeigen bei verschiedenen Lagen des Magneten, bei welchen Abständen der Reedschalter geschlossen ist. Beachten Sie dabei die Achsenbeschriftung und die entsprechenden Pole des Magneten (N bzw. S). Wir sehen zum Beispiel, dass wenn der Magnet rechtwinklig und mittig zum Reedschalter liegt, wird der Schalter nicht schließen. Erst wenn der Magnet ein bisschen nach links oder rechts bewegt wird, schließt der Schalter (zu sehen in Figure 4).

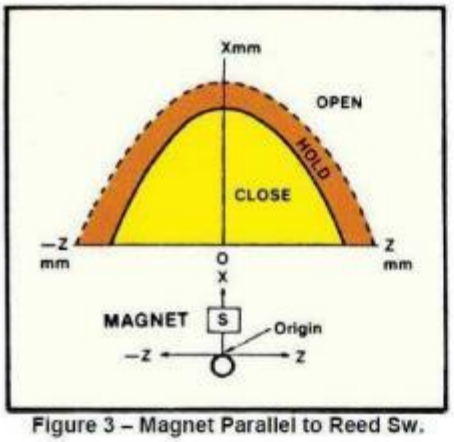


Figure 3 – Magnet Parallel to Reed Sw.

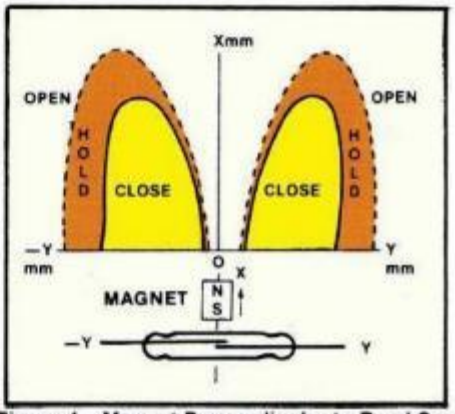


Figure 4 – Magnet Perpendicular to Reed Sw.

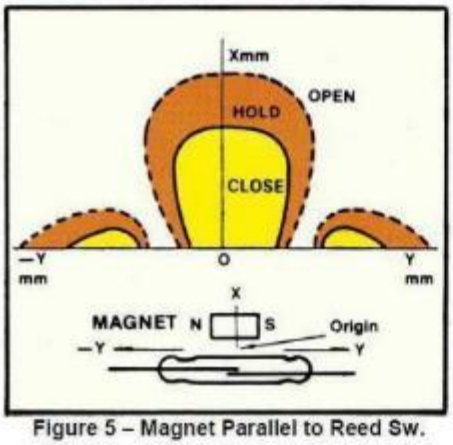
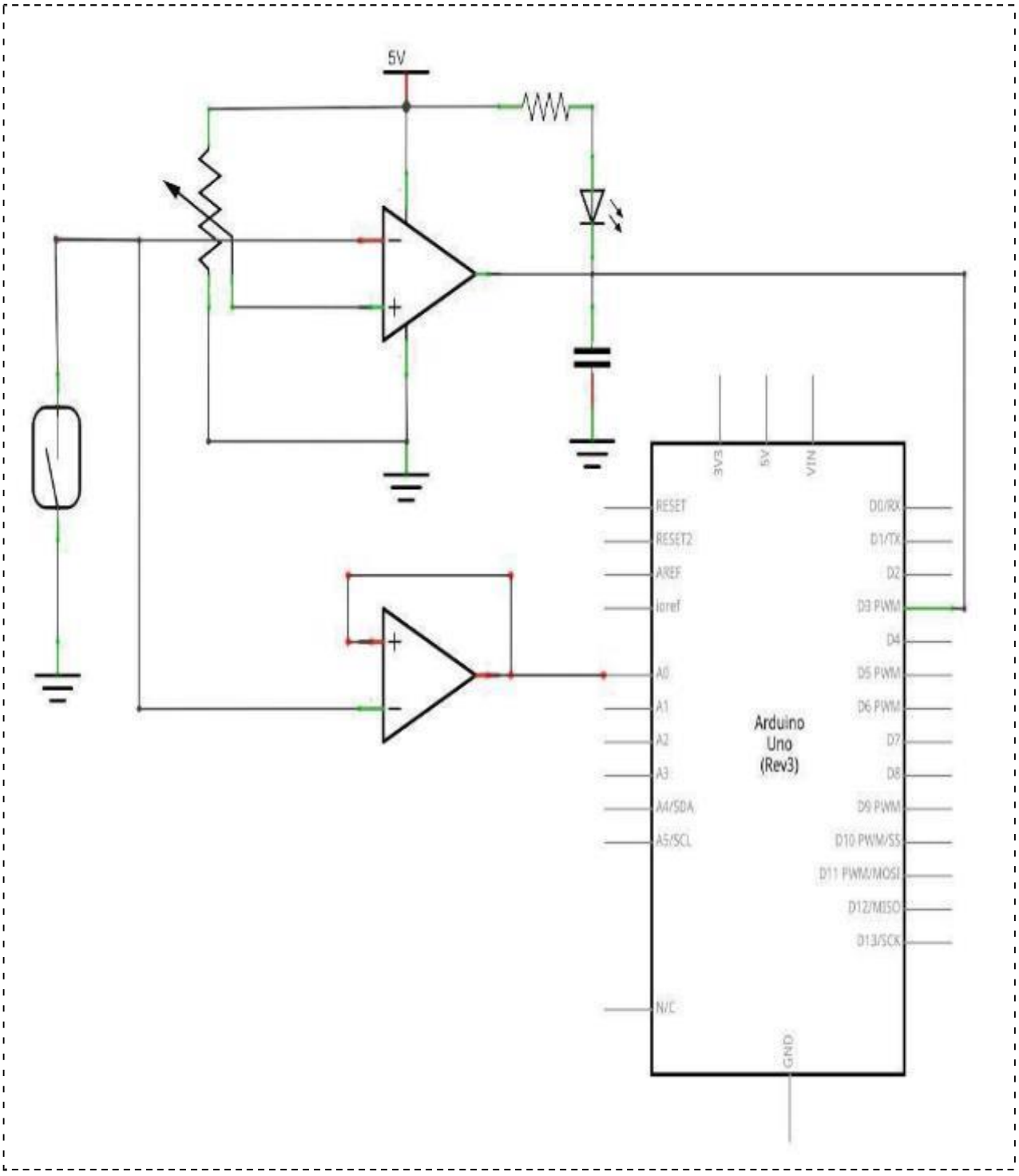


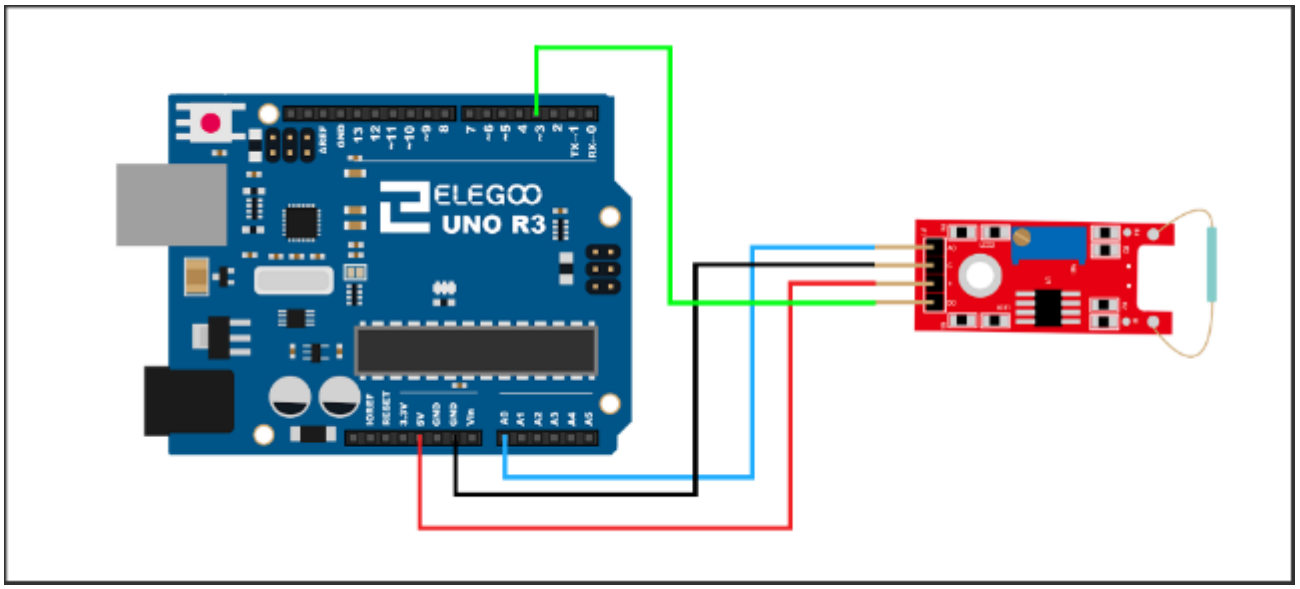
Figure 5 – Magnet Parallel to Reed Sw.

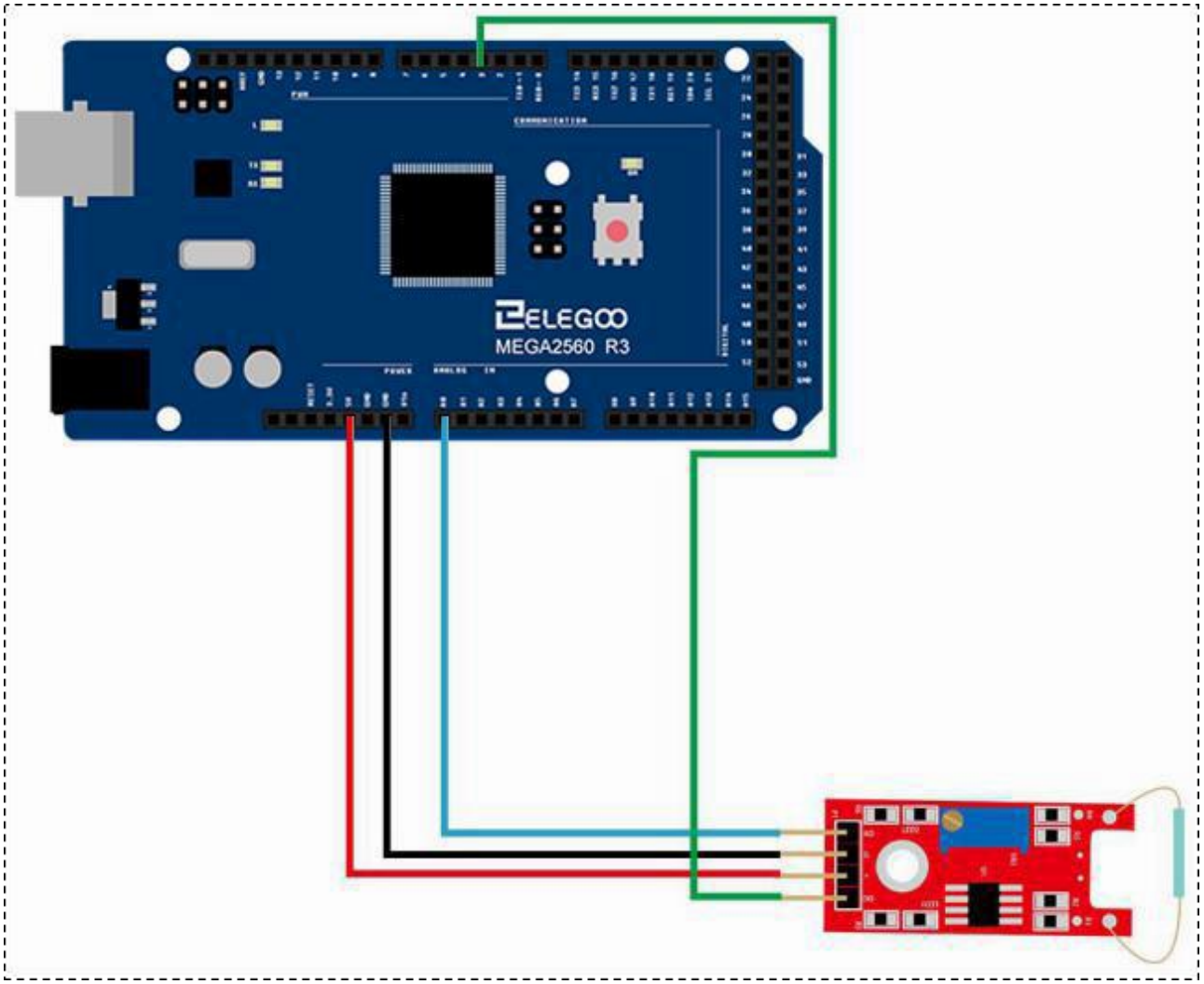
Verbindung des Reedschaltermoduls

Schaltplan



Verdrahtungsplan

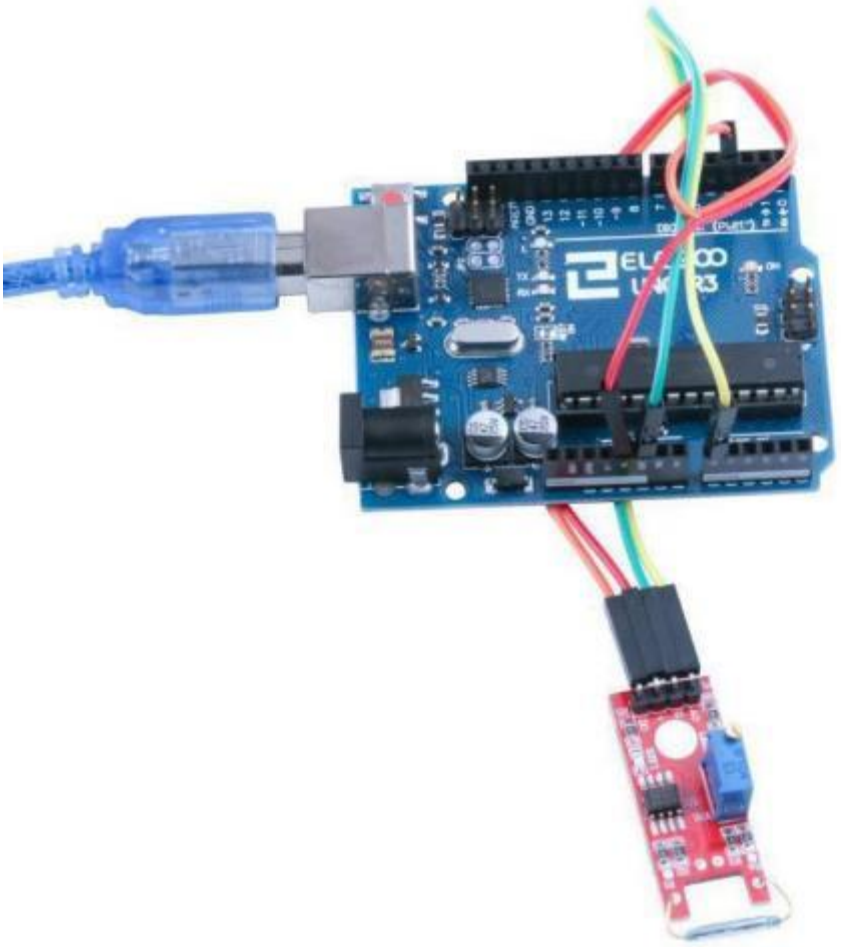


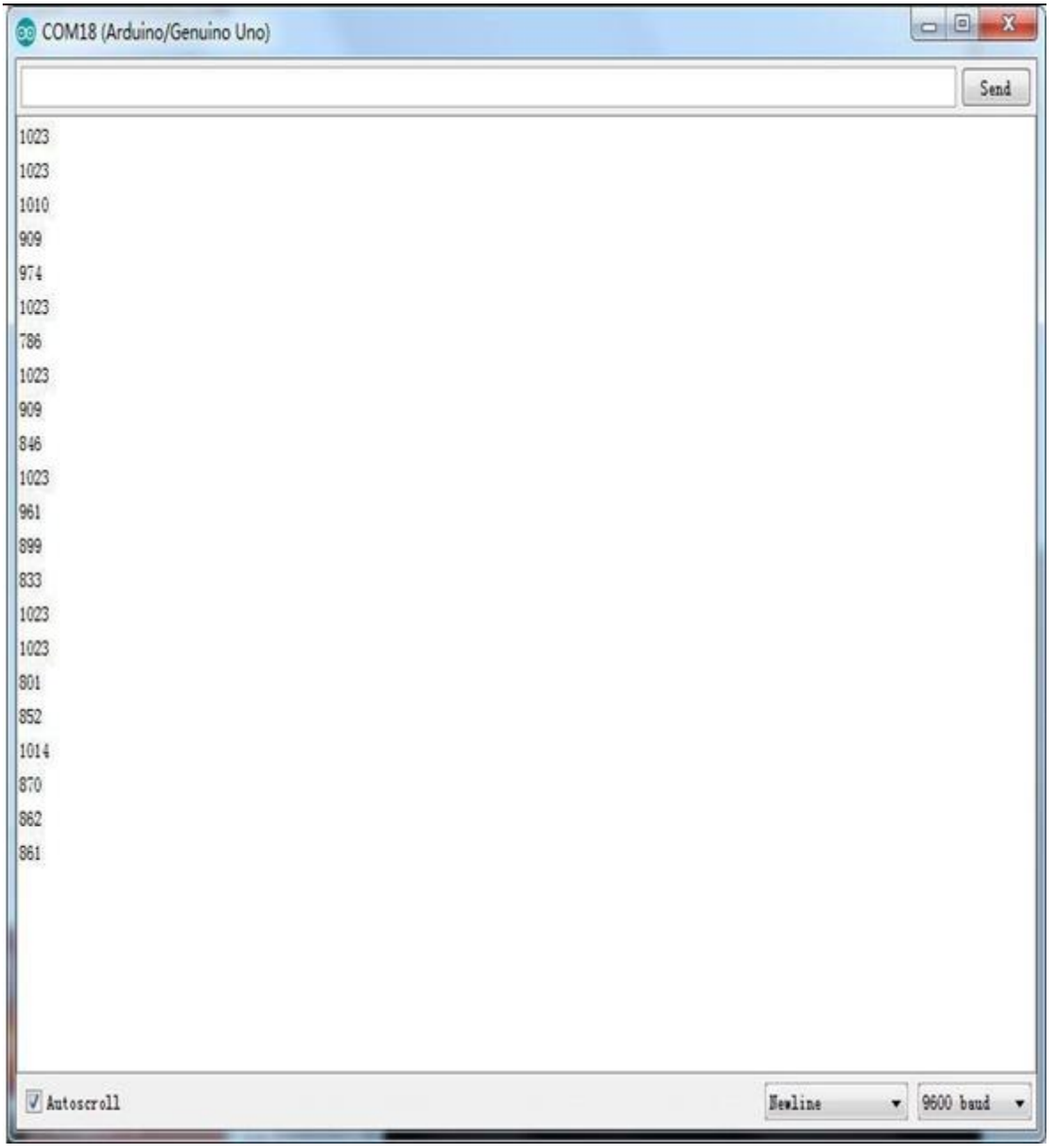


Code

Nachdem das Modul mit dem Ardunio verbunden ist, laden Sie bitte das Programm "Lesson 17 REED SWITCH AND MINI REED SWITCH MODULE ". Wenn der Sensor ein Magnetfeld erkennt gibt das Modul die Werte aus, die die Stärke des Magnetfelds widerspiegeln. Die Werte liegen zwischen 0 und 1023.

Beispielbild




COM18 (Arduino/Genuino Uno)
Send
1023
1023
1010
909
974
1023
786
1023
909
846
1023
961
899
833
1023
1023
801
852
1014
870
862
861
Autoscroll
Newline
9600 baud

Im Folgenden finden Sie den Code und einige Erläuterungen

Wir lassen die eingebaute LED an Pin 13 blinken. Die Blinkfrequenz hängt dabei vom Magnetfeld ab. Je stärker das Magnetfeld ist, desto langsamer blinkt die LED. Sie können also die Blinkfrequenz ändern, indem sie mit einem Magneten näher zum Modul oder weiter weg vom Modul gehen.

(1) Analoge Auswertung

```
// Der Analogausgang des Moduls wird am Arduinopin A0 angeschlossen
int    sensorPin    =    A0;
// Wir benutzen zur „Ausgabe“ die eingebaute LED 13
int    ledPin    =    13;
int    sensorValue =    0;

void setup()
{
        pinMode(ledPin,OUTPUT);
        Serial.begin(9600);
}
void loop()
{
        sensorValue = analogRead(sensorPin); //Sensorwert auslesen
        digitalWrite(ledPin,    HIGH); //LED 13 anschalten
        delay(sensorValue);              //Warten, Wartezeit ist proportional zur Magnetfeldstärke
        digitalWrite(ledPin,    LOW); //LED 13 ausschalten
        delay(sensorValue);              //Warten, Wartezeit ist proportional zur Magnetfeldstärke
        Serial.println(sensorValue,    DEC);
}
```

(2) Digitale Auswertung

Dieses Programm ist sogar noch einfacher. Die LED schaltet sich nur ein bzw aus, wenn das Reedmodul schließt bzw. sich öffnet. Es wird also der Vorgang des Schaltens simuliert, während das vorherige Programm eine Näherungsmessung durchgeführt hat.

```
// Wir benutzen zur „Ausgabe“ die eingebaute LED 13
int Led=13;
// Der Digitalgausgang des Moduls wird am Arduinopin 3 angeschlossen
int buttonpin=3;
int    val;
void    setup()
{
        pinMode(Led,OUTPUT);
        pinMode(buttonpin,INPUT);
}
void    loop()
{
        //Auslesen des Wertes vom Reedsensormodul
        val=digitalRead(buttonpin);
        //Wenn der Reedschalter geschlossen ist, LED einschalten, wenn er offen ist, LED ausschalten
        if(val==HIGH)
        {
                digitalWrite(Led,HIGH);
        }
        else
        {
                digitalWrite(Led,LOW);
        }
}
```

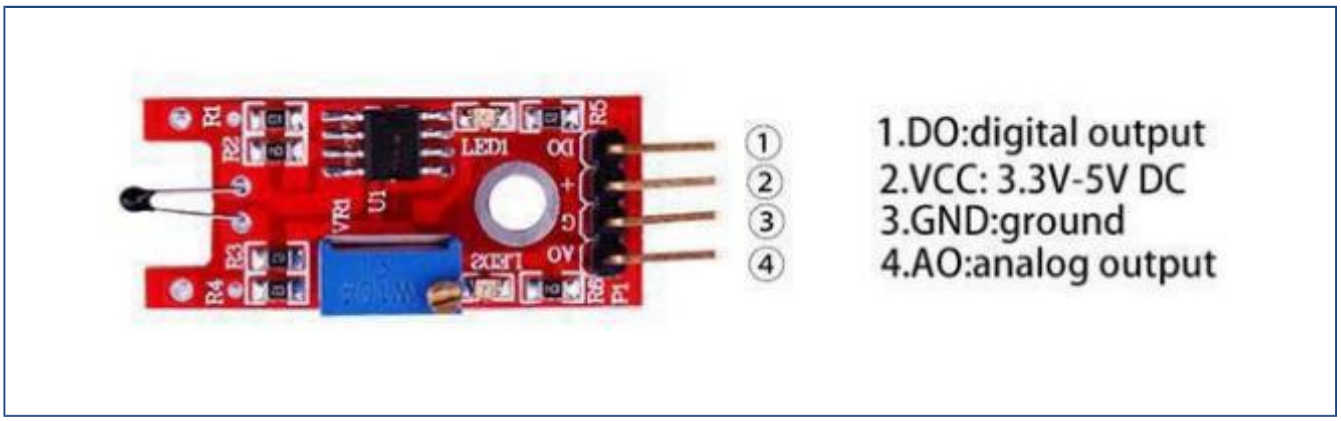
Lektion 18 DIGITALES TEMPERATURMODUL

Überblick

In diesem Versuch lernen wir mit einem digitalen Temperaturmodul umzugehen.

Digitales Temperaturmodul

Temperaturfühlermodul mit einem NTC-Thermistor. Der digitale Ausgang 'DO' ist high, wenn die eingestellte (mittels Potentiometer) Temperatur erreicht ist. Am Pin'AO' steht ein analoges Ausgangssignal des Sensors zur Verfügung.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

1x Digitales Temperaturmodul

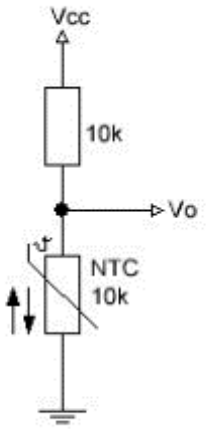
4x F-M Drähte

Komponenteneinführung

Thermistor (=temperaturabhängiger Widerstand) :

Was ist ein NTC Widerstand?

Thermistoren sind Temperaturfühler aus gesintertem Halbleitermaterial. Der Widerstandswert kann ermittelt werden, indem man einen bekannten Gleichstrom durch den NTC schickt und den Spannungsabfall am NTC misst (wie im nachstehenden Bild zu sehen).



NTC Widerstände sind nichtlineare Widerstände, die ihr Widerstandsverhalten mit der Temperatur verändern. Der Widerstand von NTCs nimmt mit steigender Temperatur ab. Deshalb werden sie auch Heißleiter genannt. Die Abnahme des Widerstands steht im Zusammenhang mit einer Konstante, die in der Elektrotechnik als Beta oder ß bekannt ist. Beta wird in K (=Kelvin) angegeben.

Wenn wir nun einen Spannungsteiler machen, der typischerweise zwei Widerstände in Reihe zwischen Vcc und Masse und dem analogen Port in der Mitte hat, hängt der Messwert vom Verhältnis der beiden Widerstände ab: Wenn sie gleich sind, ist der Messwert 512 bzw. 511 (Die ganzzahlige Hälfte von 1023). Wenn einer der Widerstände ein NTC ist variieren die Werte am Analogport mit der Temperatur:

Wenn die Temperatur sinkt, steigt der Wert des Widerstandes und damit auch die gemessene Spannung am Analogausgang.

Nehmen wir an, wir haben einen 10K Widerstand in Reihe mit einem NTC, den wir von jetzt an „R“ nennen.
Dann beträgt die Spannung (Vo) in der Mitte zwischen den Widerständen:
Vo=R/(R+10K) * Vcc

Der Analogeingang des Arduino skaliert diese Spannung auf einen Wert zwischen 0 und 1023. Daraus kann der Spannungswert folgendermaßen berechnet werden.

Wert des Analog-Digital-Wandlers = Vo*1023/Vcc

oder

Wert des Analog-Digital-Wandlers = 1023 *(Vo/Vcc)

Durch Einsetzen der beiden Formeln in einander erhalten wir folgendes:

Wert des Analog-Digital-Wandlers = (R/(R+10K))*Vcc*1023/Vcc

Wie man sieht, kürzt sich Vcc heraus, woraus sich folgendes ergibt:

Wert des Analog-Digital-Wandlers = (R/(R+10k))*1023

Aufgelöst nach R bekommen wir:

R=10k/(1023/ADC-1)

Bitte beachten Sie, die Formel gilt nur für eine Pull-Up Konfiguration, wie sie am Anfang des Kapitels gezeigt wurde. D.h. der NTC liegt mit einem Beinchen an Masse und der andere Widerstand liegt an der Versorgungsspannung. Für deine Pull-Down-Konfiguration – also NTC an der Versorgungsspannung und der andere Widerstand an Masse – gilt folgende Formel:

R = 10k/(1023/ADC – 1));

Wie sieht das ganze als Code aus?

```
byte NTCPin = A0;
const int SERIESRESISTOR = 10000; //Der Reihenwiderstand hat 10k
void setup()
{
        Serial.begin(9600);
}
void loop()
{
        float ADCvalue;
        float Resistance;
        ADCvalue = analogRead(NTCPin);
        Serial.print("Analog ");
        Serial.print(ADCvalue); //Wert des Analog-Digital-Wandlers
        Serial.print(" = ");
        Resistance = (1023 / ADCvalue) - 1; //Nenner der obigen Formel für pull-up Konfiguration …
        Resistance = SERIESRESISTOR / Resistance; //…auf zwei Berechnungen aufgesplittet
        Serial.print(Resistance);
        Serial.println(" Ohm");
        delay(1000);
}
```

Den Widerstand des NTC zu kennen ist ja ganz nett, aber wir kennen die Temperatur immer noch nicht, oder?

Nun, viele NTC's haben einen Nennwert, der bei 25 Grad Celsius gemessen wird. Wenn Sie also einen 10k NTC haben und 10k messen bedeutet das, dass die Temperatur in diesem Moment 25 Grad beträgt. Das hilft natürlich erstmal nicht weiter, wenn die Messung einen anderen Wert ergibt.

Sie könnten eine Tabelle anlegen, in die Sie für einige Widerstandswerte die dazugehörige Temperatur eintragen. Solche Tabellen sind recht genau (je nach Anzahl der Werte in der Tabelle), aber die Erstellung ist aufwändig und die Tabellen benötigen viel Speicher.

Es gibt jedoch eine Formel, die Steinhart-Hart-Gleichung, die eine gute Annäherung liefert. Sie ist für die meisten Zwecke absolut ausreichend.

Die Gleichung sieht in der vereinfachten Form folgendermaßen aus:

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{B} ln \left(\frac{R}{R_0} \right)$$

To ist Nenntemperatur (in den meisten Fällen und auch bei unserem Modul) 25 °C in Kelvin (= 298.15 K). B ist ein vom NTC abhängiger Parameter (bei unserem Modul 3950). Ro ist der Nennwert des NTC (bei der Nenntemperatur). R ist der gemessene Widerstandswert.

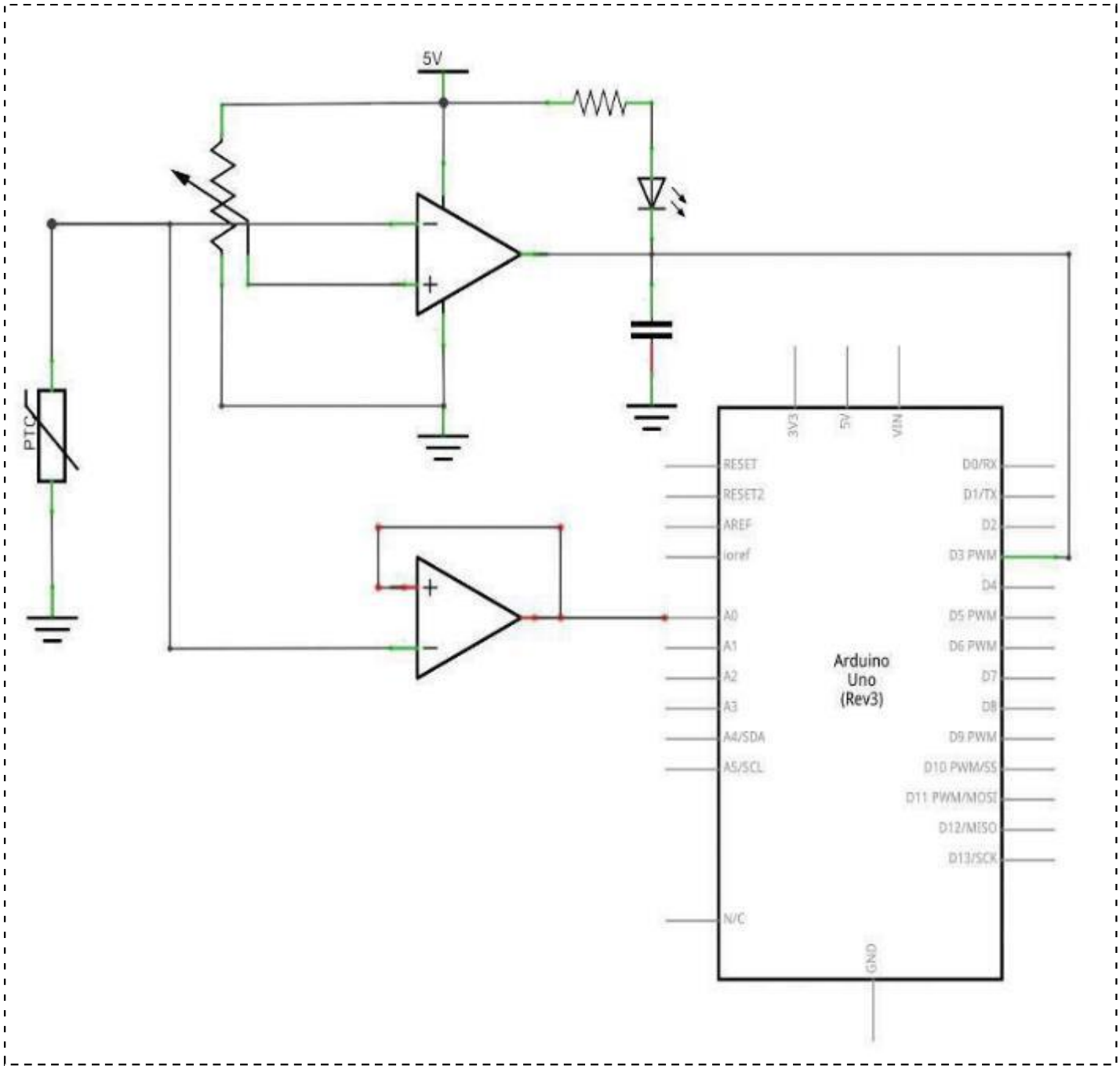
Hinweis: ln ist der logarithmus naturalis, also der Logarithmus zur Basis e (der Euler'schen Zahl).

Diese Formel können wir jetzt in unserem Programm anwenden um endlich die Temperatur messen zu können.

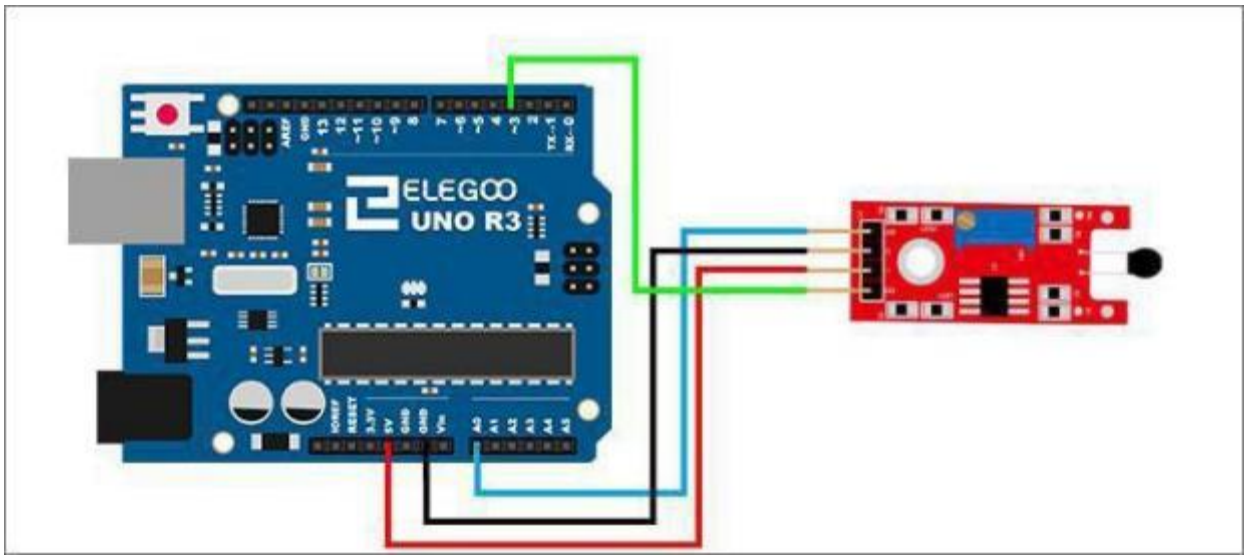
```
byte NTCPin = A0;
#define SERIESRESISTOR 10000
#define NOMINAL_RESISTANCE 10000
#define NOMINAL_TEMPERATURE 25
#define BCOEFFICIENT 3950
void setup(){
        Serial.begin(9600);
}
void loop(){
        float ADCvalue;
        float Resistance;
        ADCvalue = analogRead(NTCPin);
        Serial.print("Analoge ");
        Serial.print(ADCvalue);
        Serial.print(" = ");
        Resistance = (1023 / ADCvalue) - 1;
        Resistance = SERIESRESISTOR / Resistance;
        Serial.print(Resistance);
        Serial.println(" Ohm");
        //Ab hier beginnt die Umrechnung von Widerstandswert zu Temperatur
        float steinhart;
        steinhart = Resistance / NOMINAL_RESISTANCE; // (R/Ro)
        steinhart = log(steinhart); // ln(R/Ro)
        steinhart /= BCOEFFICIENT; // 1/B * ln(R/Ro)
        steinhart += 1.0 / (NOMINAL_TEMPERATURE + 273.15); // + (1/To)
        steinhart = 1.0 / steinhart; // Invertieren
        steinhart -= 273.15; // Von Kelvin nach Celsius umrechnen
        Serial.print ("TemperatureSerial.print(steinhart)”);
        Serial.println(" oC");
        delay(1000);
}
```

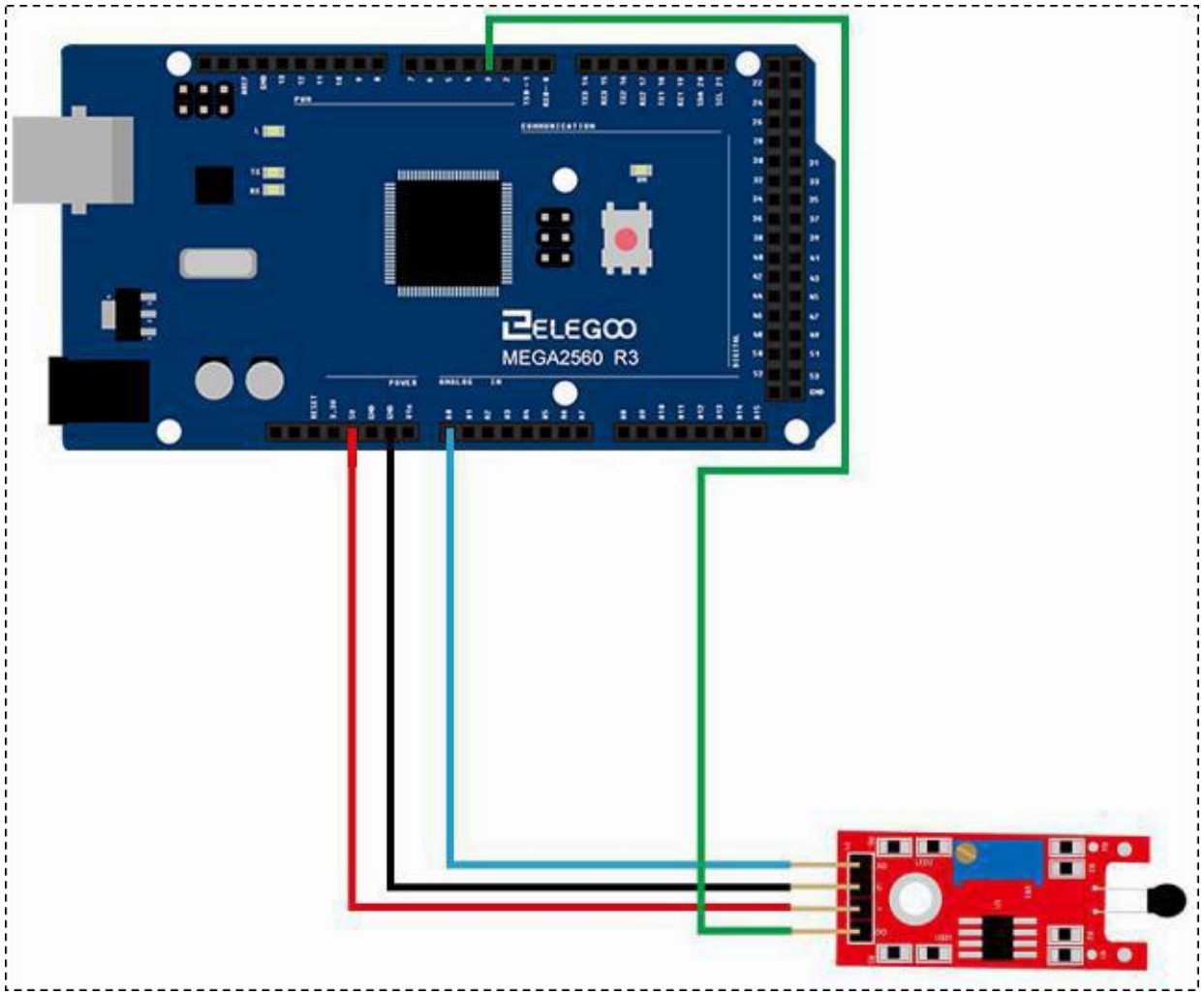
Verbindung des Moduls mit dem Arduino

Schaltplan



Verdrahtungsplan



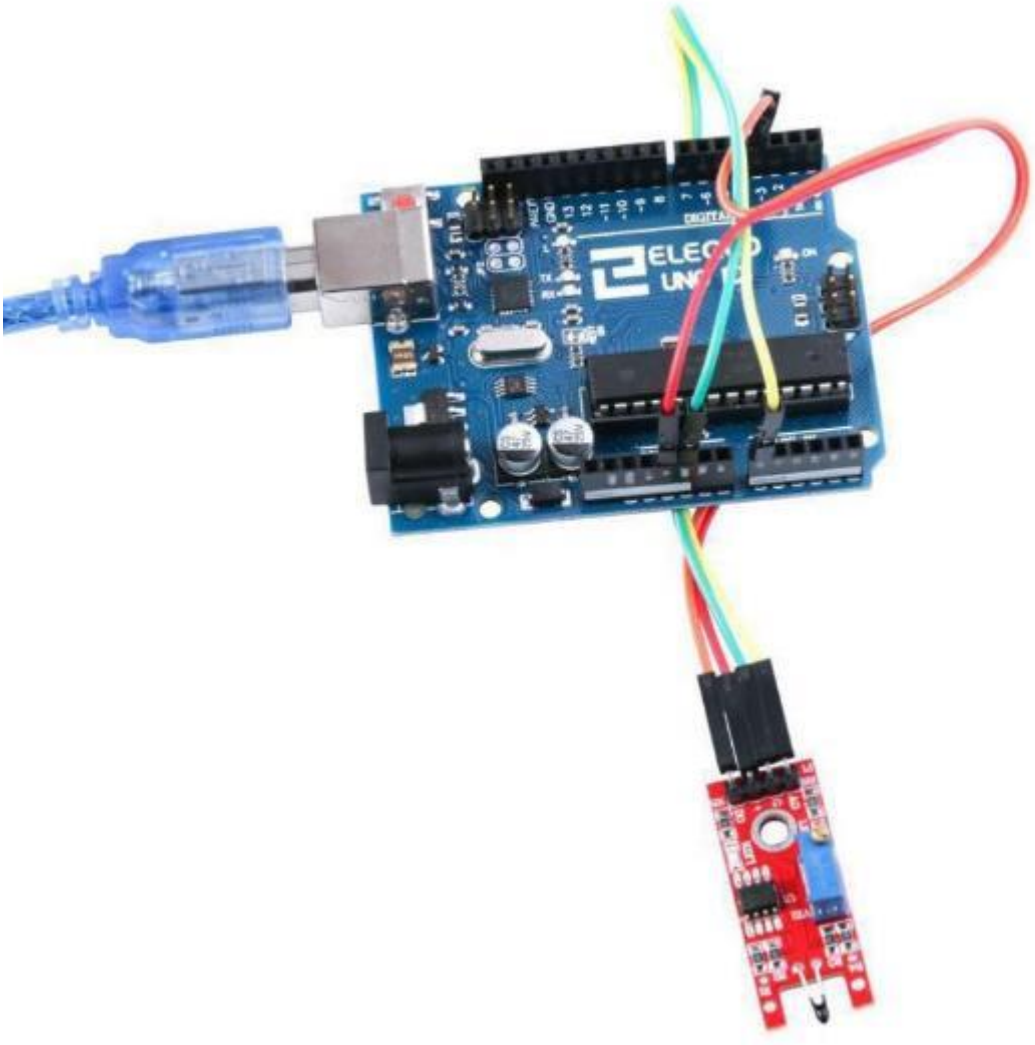


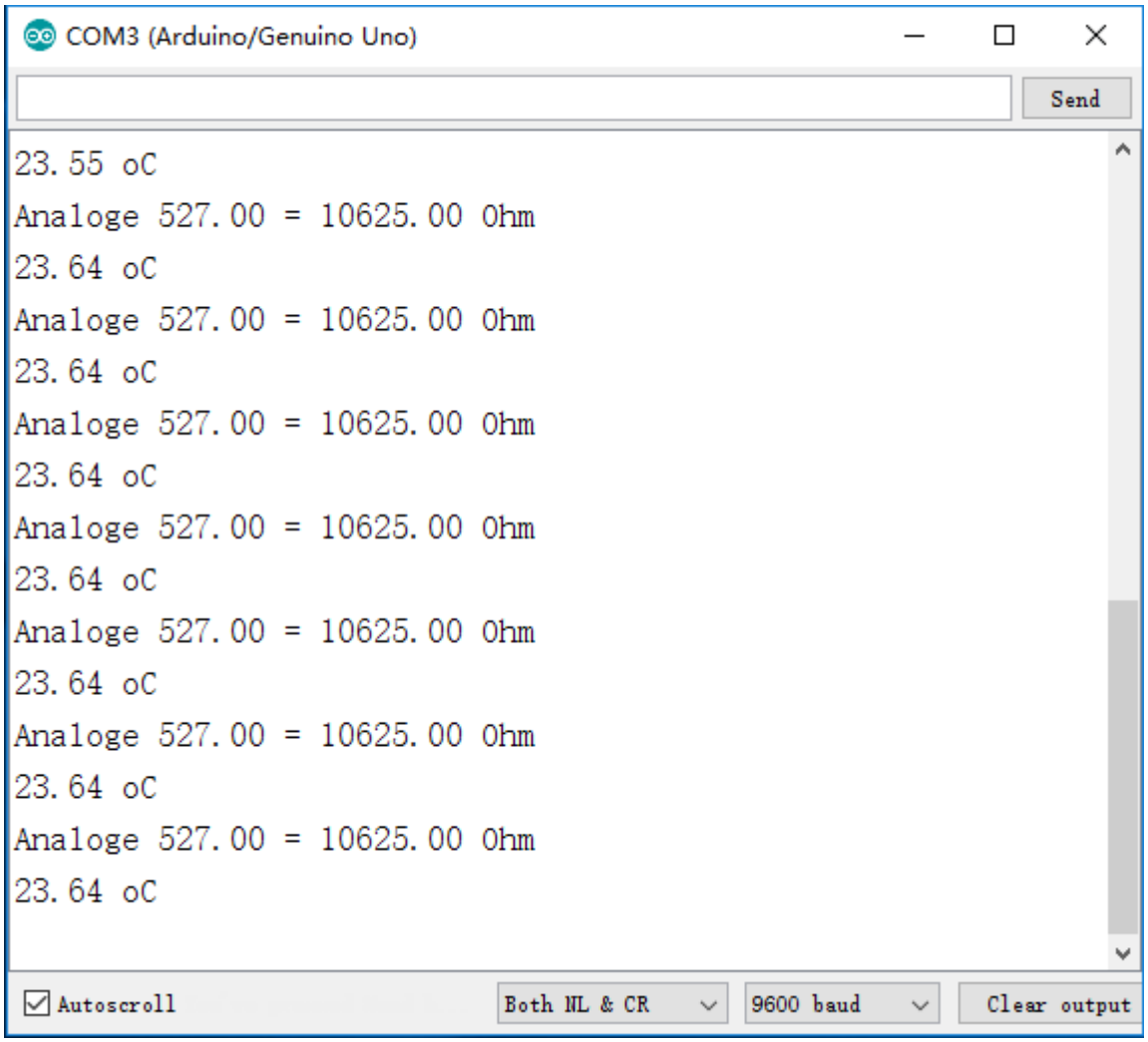
Code

Nachdem die Schaltung angeschlossen ist, laden Sie bitte das Programm " Lesson 18 DIGITAL TEMPERATURE MODULE " auf den Arduino.

Öffnen Sie den seriellen Monitor um die Messwerte zu sehen.

Beispielbild




COM3 (Arduino/Genuino Uno)
Send
23.55 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Analoge 527.00 = 10625.00 Ohm
23.64 oC
Autoscroll
Both NL & CR
9600 baud
Clear output

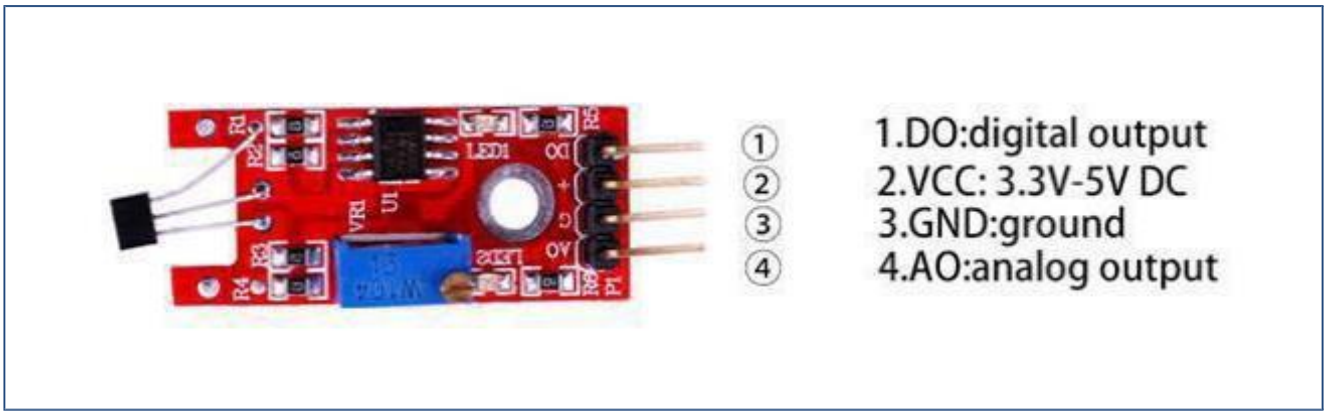
Lektion 19 LINEARES HALLSENSOR MODUL

Überblick

In diesem Versuch lernen wir, wie man ein lineares Hallsensormodul benutzt.

Linearer Hallsensor

Das lineare Hallsensormodul dient dazu zu erkennen, ob sich ein Magnetfeld in der Nähe des Sensors befindet. Variablen wie Feldstärke, Polarität und Position des Magneten relativ zum Sensor beeinflussen die Messung. Wie auch schon in den vergangenen Lektionen hat das Modul wieder einen analogen Ausgang und einen digitalen Ausgang, dessen Schaltschwelle/Empfindlichkeit mittels eines Potentiometers eingestellt werden kann.



Benötigte Komponenten:

1x Elegoo Uno R3

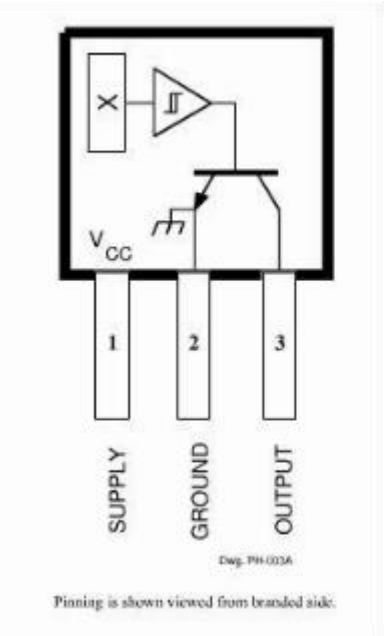
1x Lineares Hallsensormodul

1x USB Kabel

4x F-M Drähte

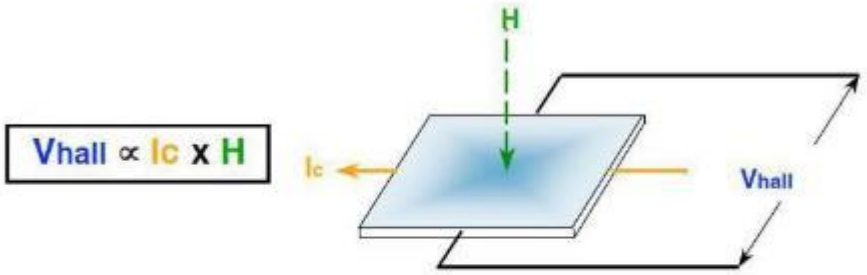
Komonenteneinführung

Hall Sensor (Auszug aus dem englischen Datenblatt):

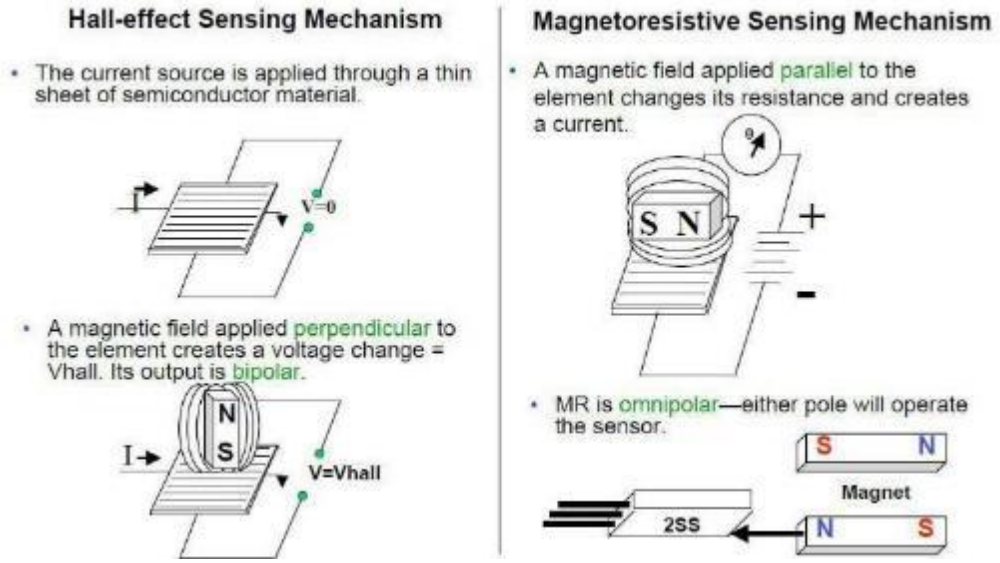


ABSOLUTE MAXIMUM RATINGS
at T_A = +25°C

Supply Voltage, V_{CC}	28 V
Reverse Battery Voltage, V_{RCC}	-35 V
Magnetic Flux Density, B	Unlimited
Output OFF Voltage, V_{OUT}	28 V
Reverse Output Voltage, V_{OUT}	-0.5 V
Continuous Output Current, I_{OUT}	25 mA
Operating Temperature Range, T_A	
Suffix 'E–'	-40°C to +85°C
Suffix 'L–'	-40°C to +150°C
Storage Temperature Range, T_S	-65°C to +170°C



Vhall = Output Hall-effect voltage
H = Magnetic Flux created by magnet or current-carrying conductor
Ic = Constant supply current

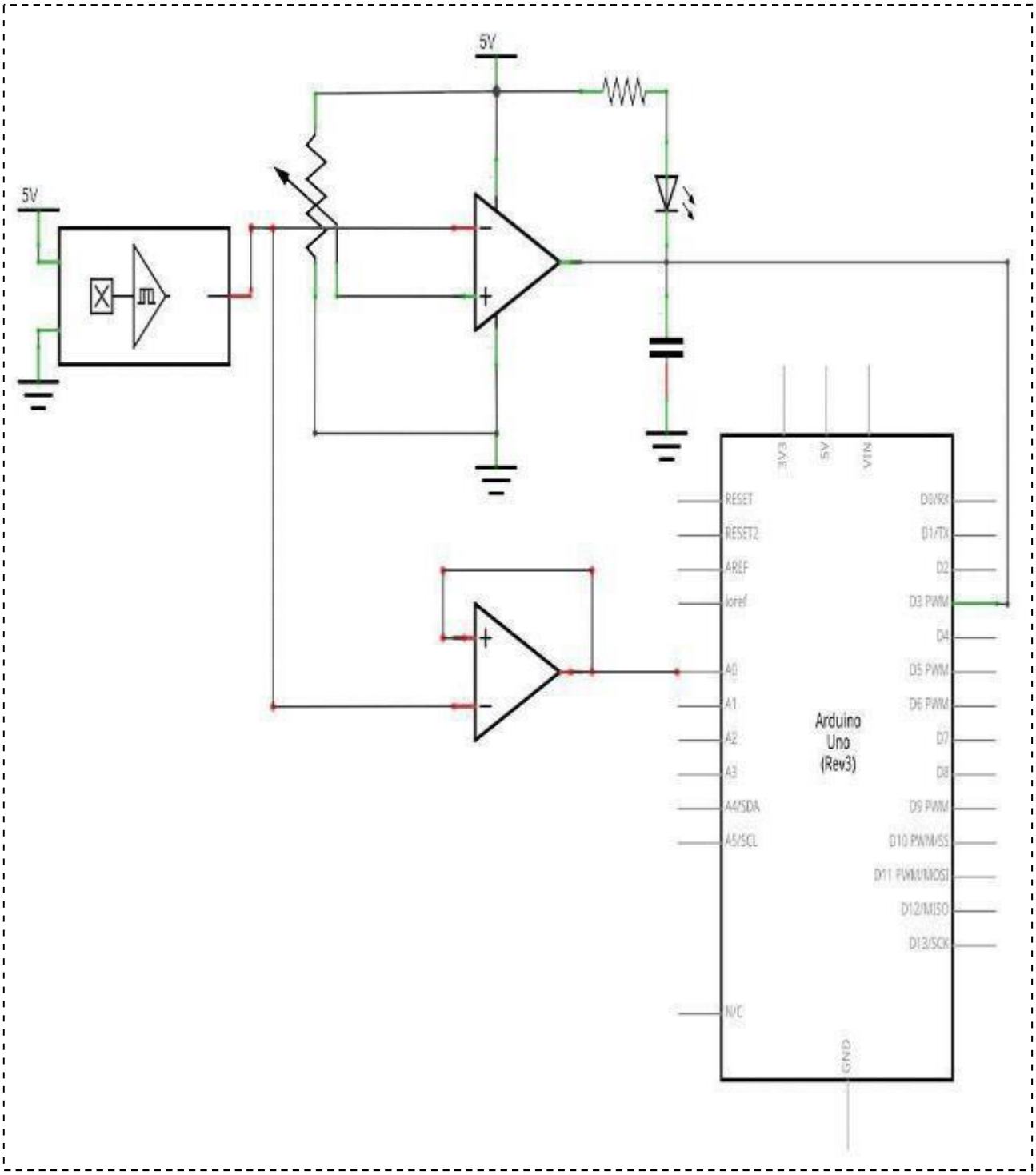


Typen von Hallsensoren:

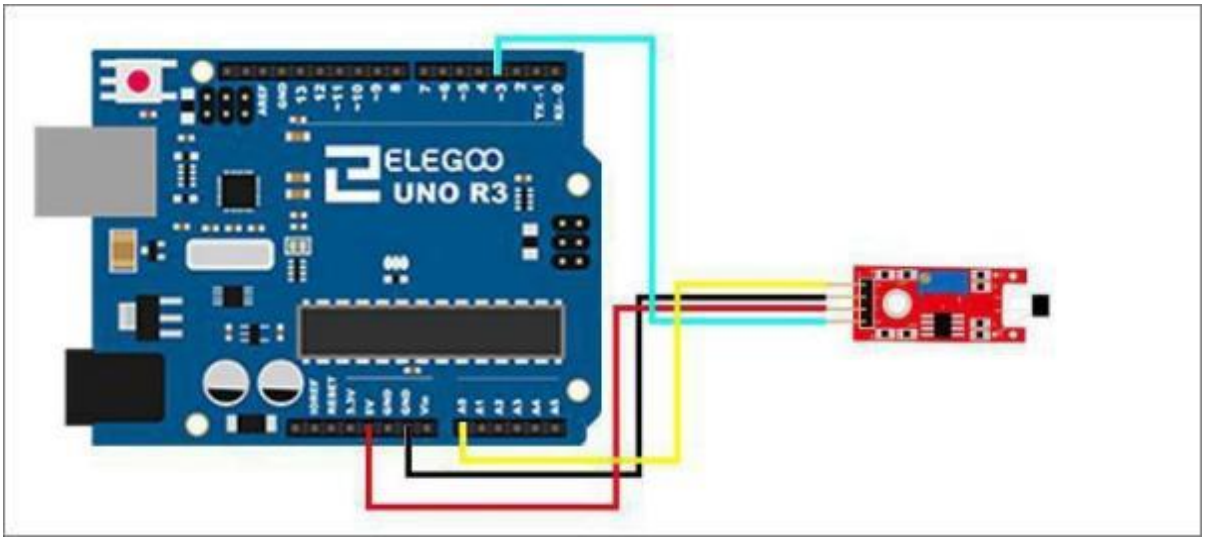
Unipolar: Nur ein magnetischer Südpol wird vom Sensor erkannt.
Bipolar: Der Sensor ist polabhängig. Ein Südpol aktiviert den Sensor und ein Nordpol deaktiviert ihn.
Latching: Der Sensor speichert sein Ergebnis und wird erst wieder durch ein Magnetfeld anderer Polirität umgeschalten werden.
Omnipolar: Der Sensor schaltet ab einer bestimmten Feldstärke unabhängig von der Polarität.

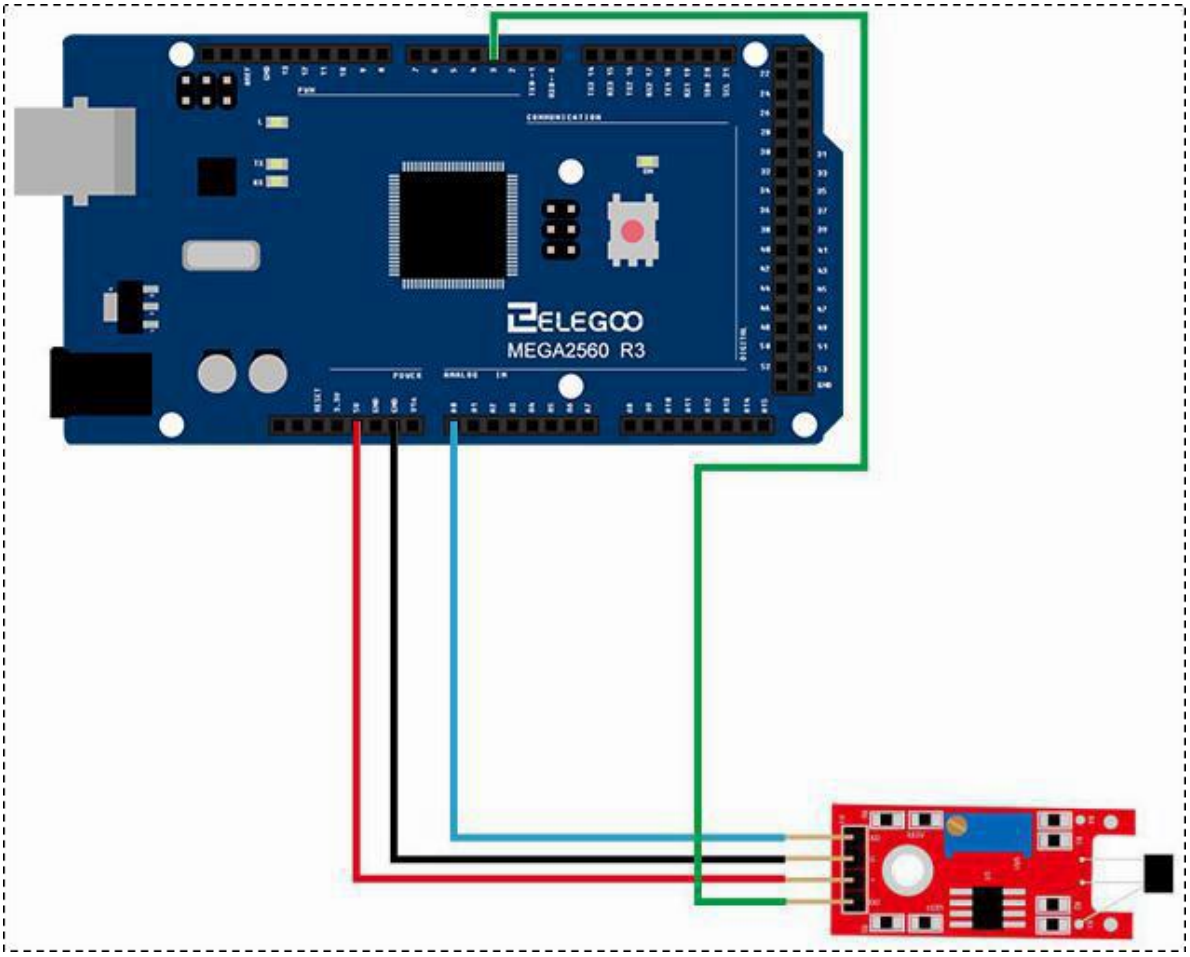
Verbindung

Schaltplan



Verdrahtungsplan



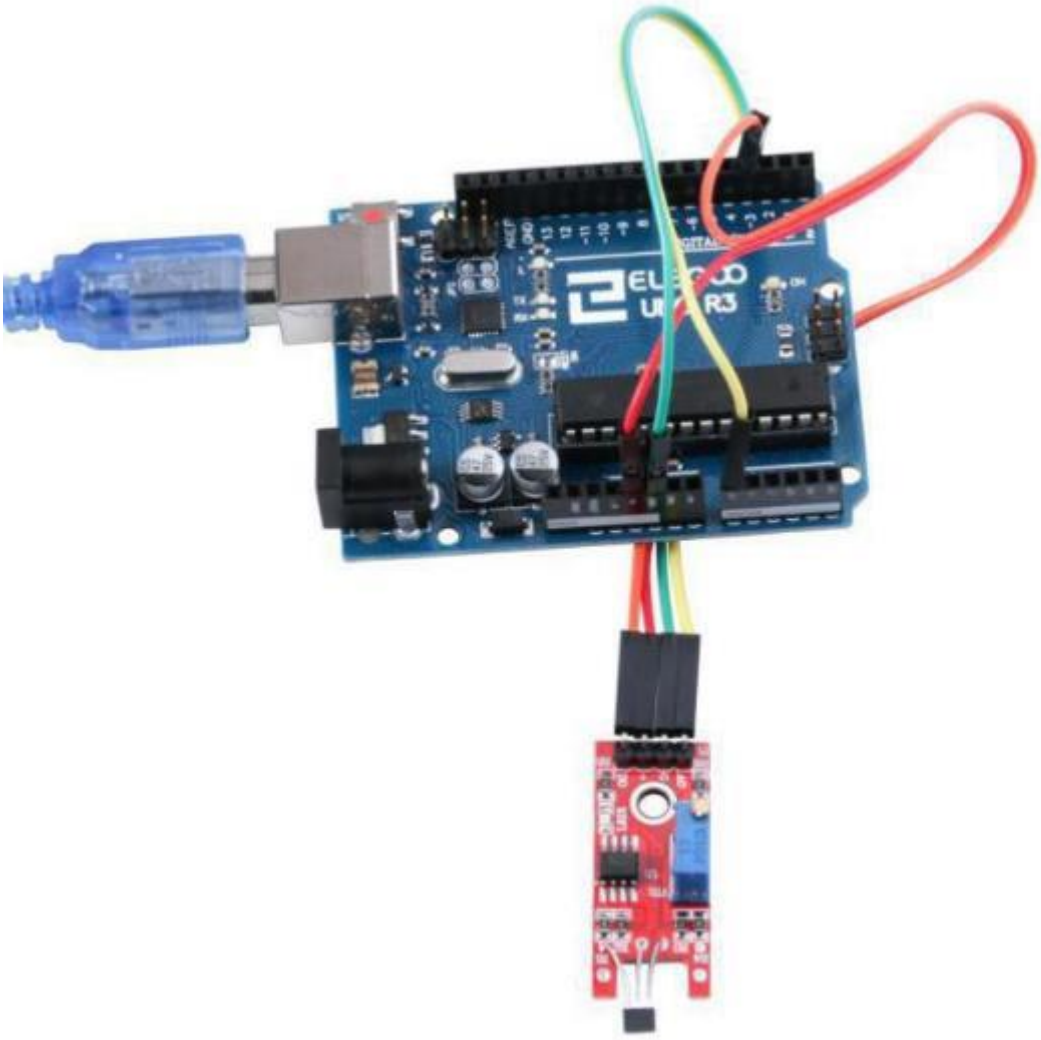


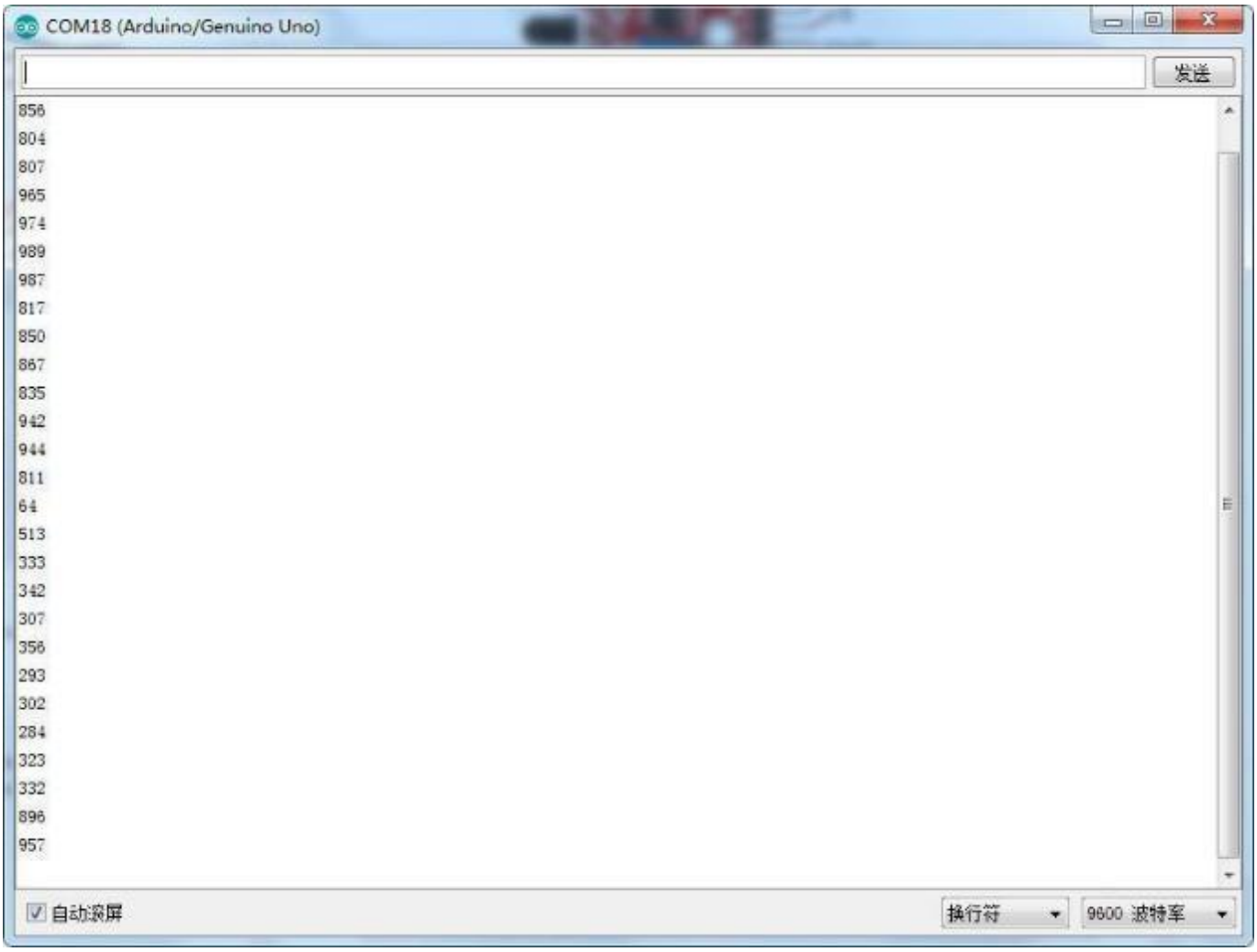
Code

Nachdem Sie die Schaltung angeschlossen haben, laden Sie bitte das Programm "Lesson 19 LINEAR HALL AND ANALOG HALL MODULE" auf den Arduino.

Beim Hallsensormodul können wir den digitalen oder den analogen Ausgang wählen. Wir benutzen im ersten Beispielprogramm den digitalen Ausgang und lassen eine LED aufleuchten, wenn das Magnetfeld in der Nähe des Sensors einen Schwellwert überschreitet, der mittels des Potentiometers eingestellt werden kann. Im zweiten Beispielprogramm benutzen wir den analogen Ausgang.

Beispielbild





Im Folgenden finden Sie den Code mit ein paar Erklärungen

(1)digitaler Ausgang

```
//Wir benutzen wieder die eingebaute LED 13
int Led=13;
//Der Digitalausgang des Hallsensormoduls wird an Pin 3 des Arduino angeschlossen
int buttonpin=3;
int val;
void  setup()
{
    //die LED ist ein Ausgang
    pinMode(Led,OUTPUT);
    //der Pin an dem das Hallsensormodul angeschlossen ist, ist ein Eingang
    pinMode(buttonpin,INPUT);
}
```

```
void    loop()
{
      //Einlesen des binären Wertes vom Hallsensormodul
      val=digitalRead(buttonpin);
      //Wenn der Sensor ein Magnetfeld erkannt hat, schalten wir die LED an …
      if (val==HIGH)
      {
            digitalWrite(Led,HIGH);
      }
      else
      {
      // …und wenn das Magnetfeld unterhalb des Schwellwertes liegt, schalten wir die LED wieder aus
            digitalWrite(Led,LOW);
      }
}
```

(2) analoger Ausgang

```
//Der Analogausgang des Hallsensormoduls wird an Pin A0 des Arduino angeschlossen
int    sensorPin    =    A0;
//Wir benutzen wieder die eingebaute LED 13
int    ledPin    =    13;
int sensorValue =    0;
void setup()
{
      pinMode(ledPin,OUTPUT);
      Serial.begin(9600);
}
void loop()
{
      /*Je höher das Magnetfeld ist, desto langsamer blinkt die LED. Dies wird erreicht, da nach jedem
Schaltvorgang der LED so lange gewartet wird in ms wie der Wert des Analog-Digital-Wandlers ist
```

```
 (zwischen 0 und 1023)*/
      sensorValue = analogRead(sensorPin);
      digitalWrite(ledPin, HIGH);
      delay(sensorValue);
      digitalWrite(ledPin, LOW);
      delay(sensorValue);
      Serial.println(sensorValue, DEC);
 }
```

Lektion 20 Flammensensormodul

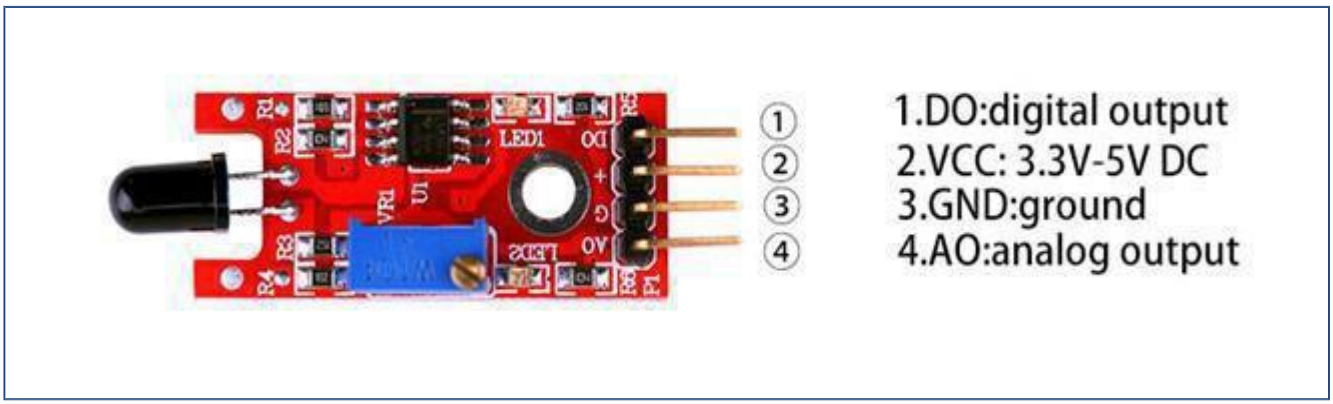
Überblick

In diesem Versuch lernen wir den Umgang mit dem Flammensensormodul. Dieses Modul ist flammen- und strahlungsempfindlich. Es kann auch gewöhnliche Lichtquellen im Bereich von 760nm-1100nm detektieren. Der Erfassungsabstand beträgt bis zu einem Meter. Der Flammensensor kann ein digitales oder analoges Signal ausgeben. Er kann als Flammenmelder oder in Feuerlöschrobotern eingesetzt werden.

Flammen

Ein Sensormodul zur Erkennung von Flammen. Die spektrale Empfindlichkeit des Sensors ist optimiert, um die Strahlung von offenen Flammen zu erkennen. Das Ausgangssignal 'DO' wird high, wenn eine Flamme erkannt wird. Die Schaltschwelle ist über ein Potentiometer einstellbar. Am Pin 'AO' steht ein analoges Ausgangssignal des Sensors zur Verfügung.

- Typischer Detektionsbereich: 720-1100nm
- Typischer Detektionswinkel: 60°



Benötigte Komponenten:

1 x Elegoo Uno R3
1 x USB Kabel
1 x Flammensensormodul
4 x F-M Drähte

Komponenteneinführung

Flammensensor:

Erkennt eine Flamme oder eine Lichtquelle mit einer Wellenlänge im Bereich von 760nm-1100nm

Erfassungsabstand: 20cm (4,8V) ~ 100cm (1V)

Erfassungswinkel ca. 60 Grad

Der Komparatorchip LM393 macht die Modulmesswerte stabil.

Einstellbarer Erfassungsbereich.

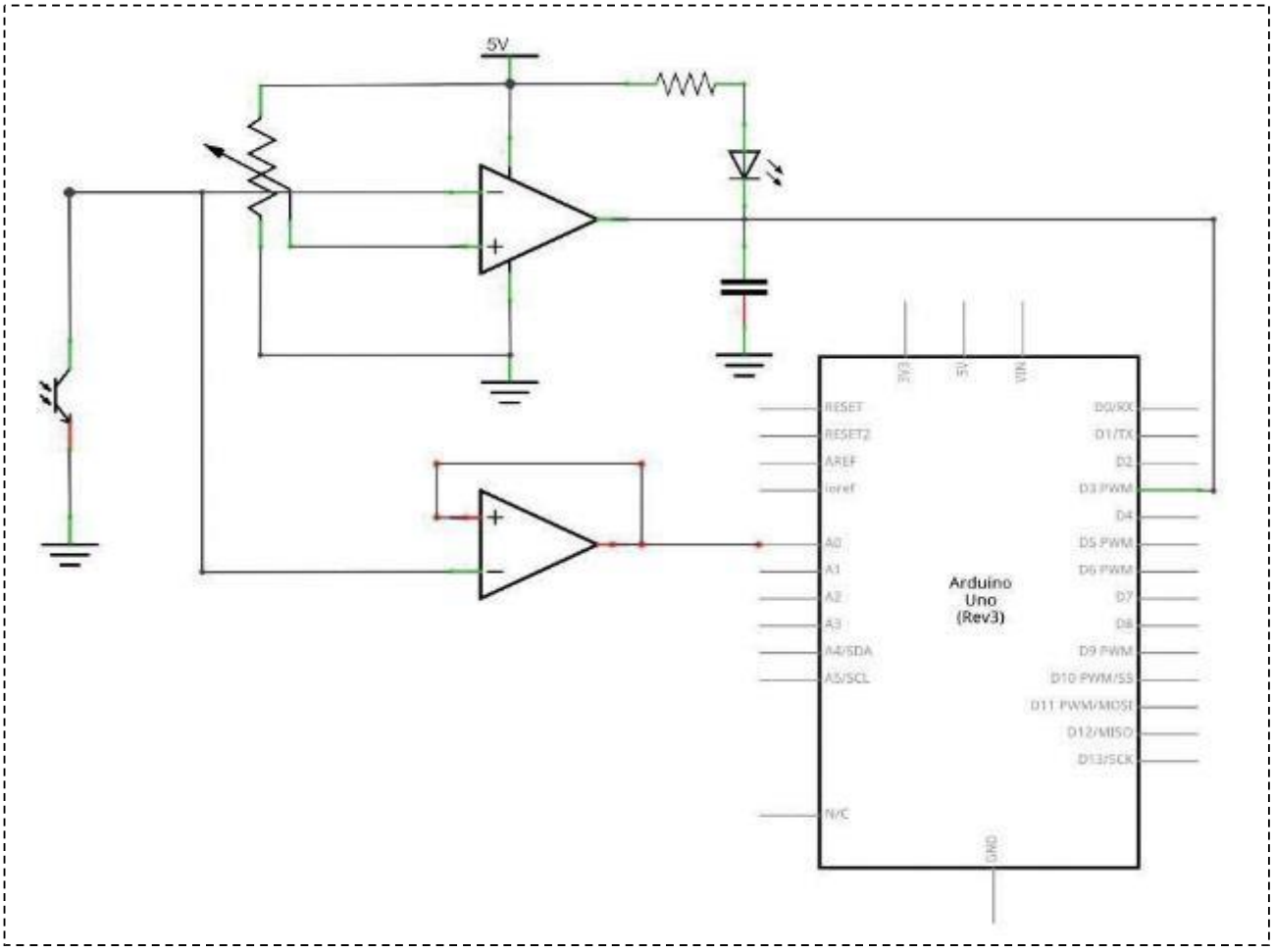
Betriebsspannung 3,3V-5V

Digitaler und analoger Ausgang

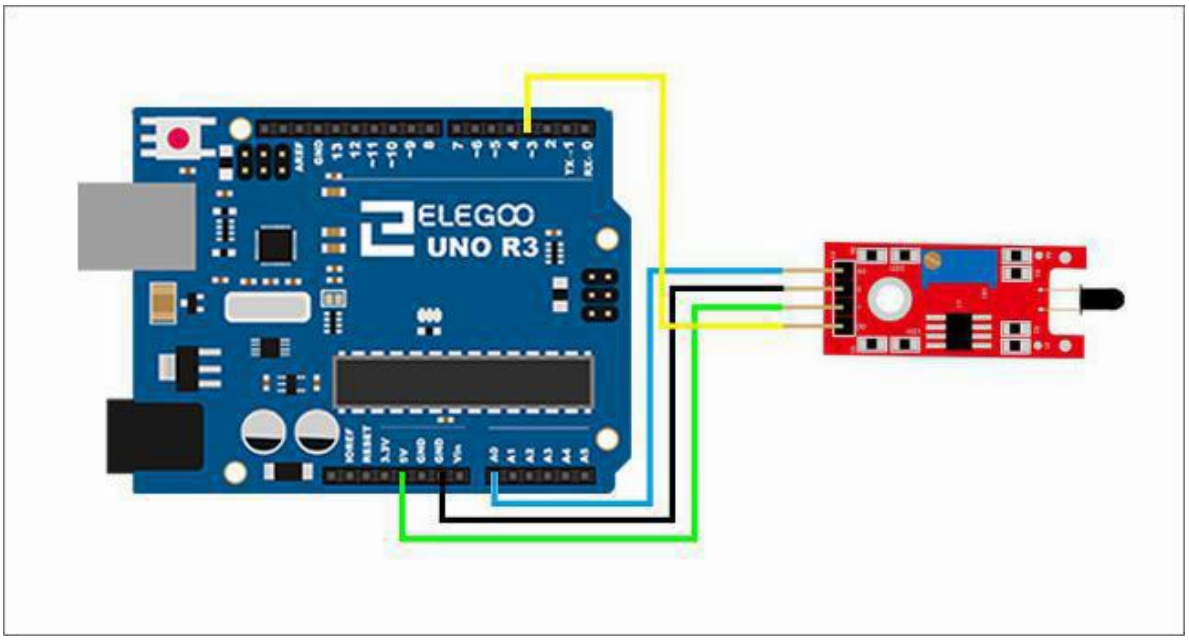
Anzeige (LED) jeweils für Spannungsversorgung und digitalen Schaltvorgang

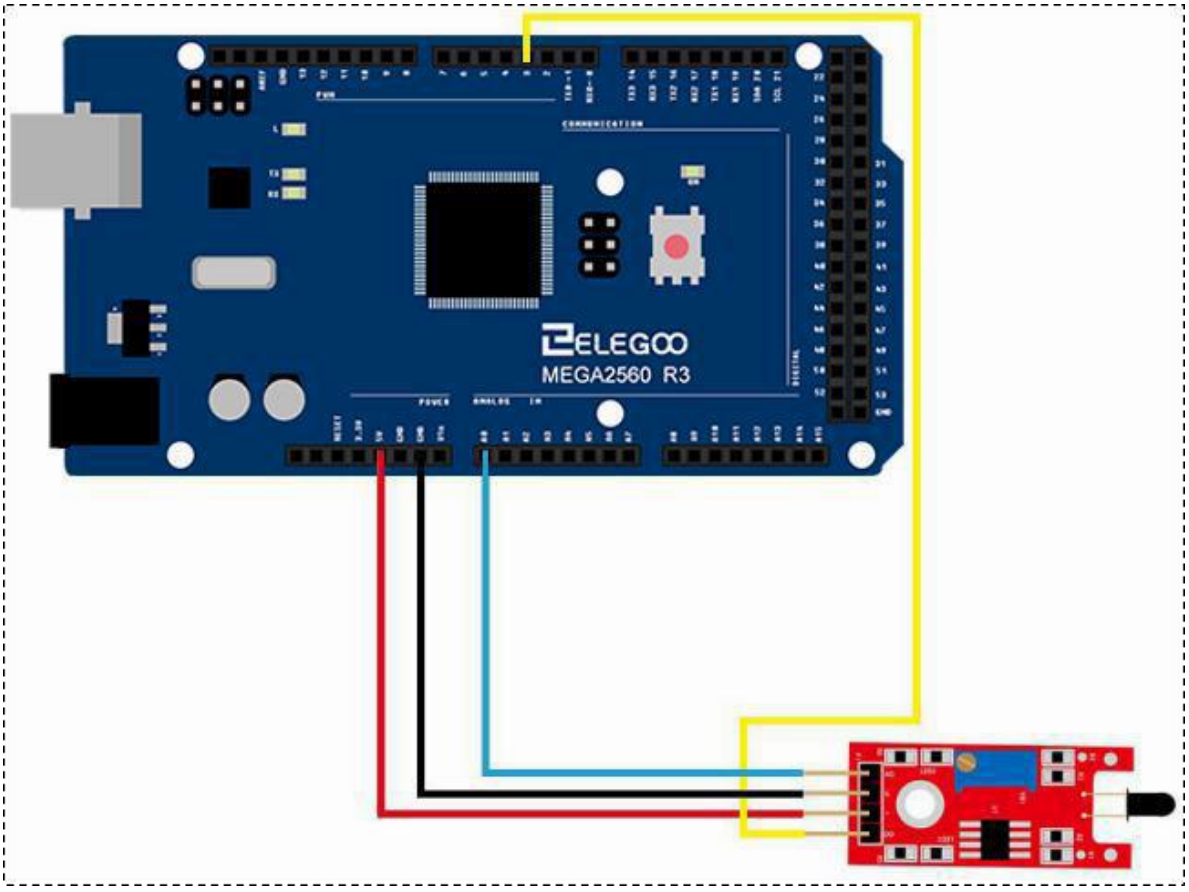
Verbindung

Schaltplan



Verdrahtungsplan



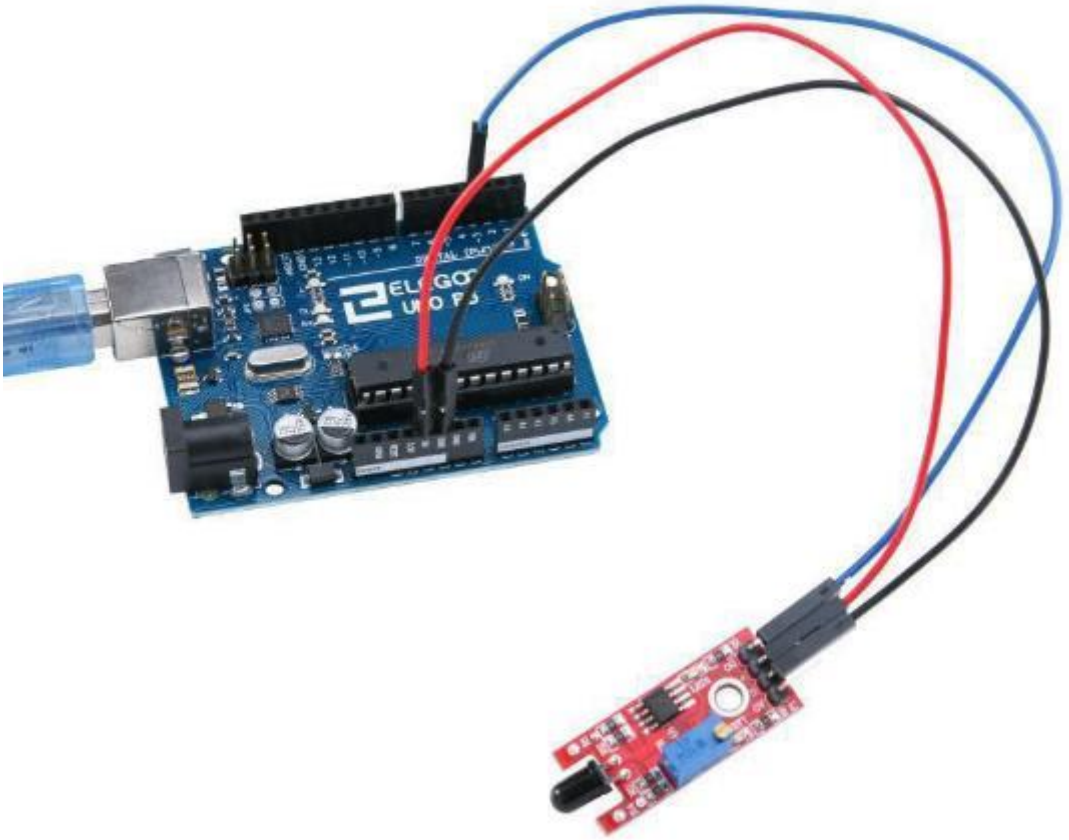


Code

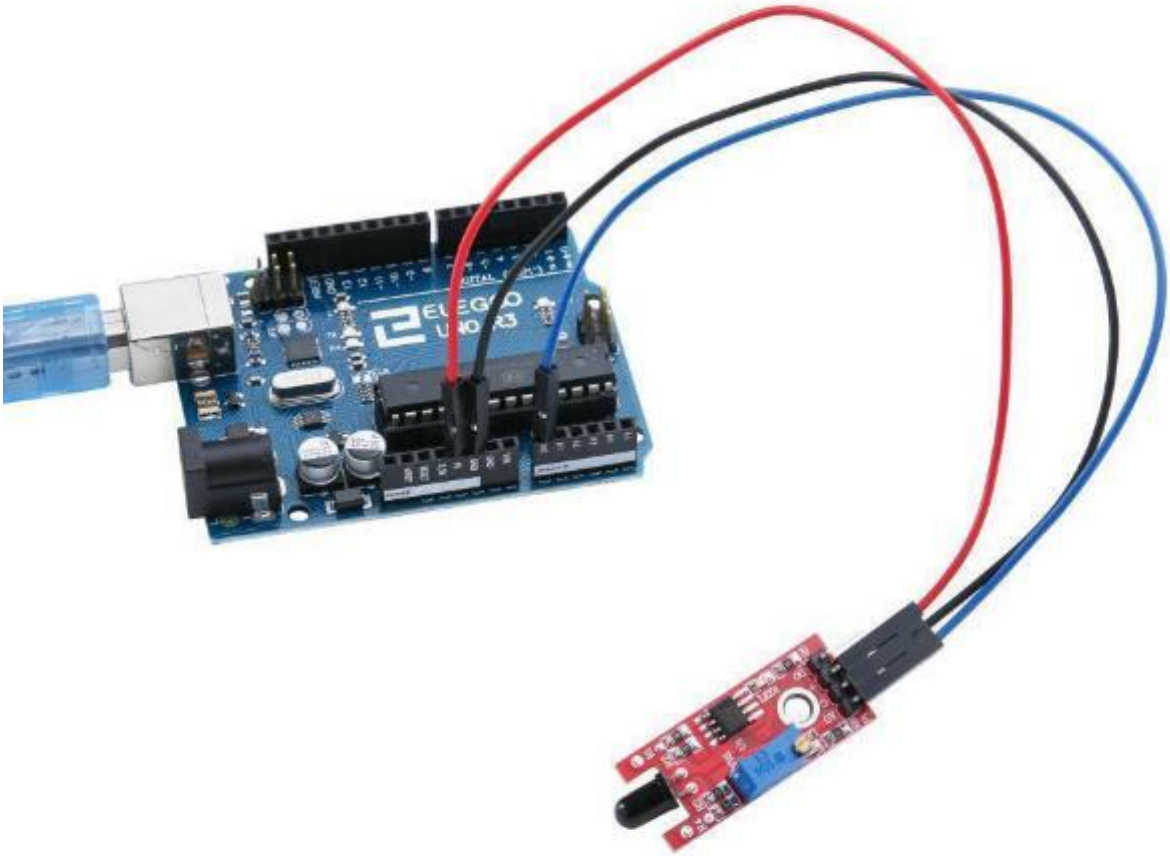
Nachdem die Schaltung verbunden ist, laden Sie bitte das Programm "Lesson 20 FLAME SENSOR MODULE" auf den Arduino.

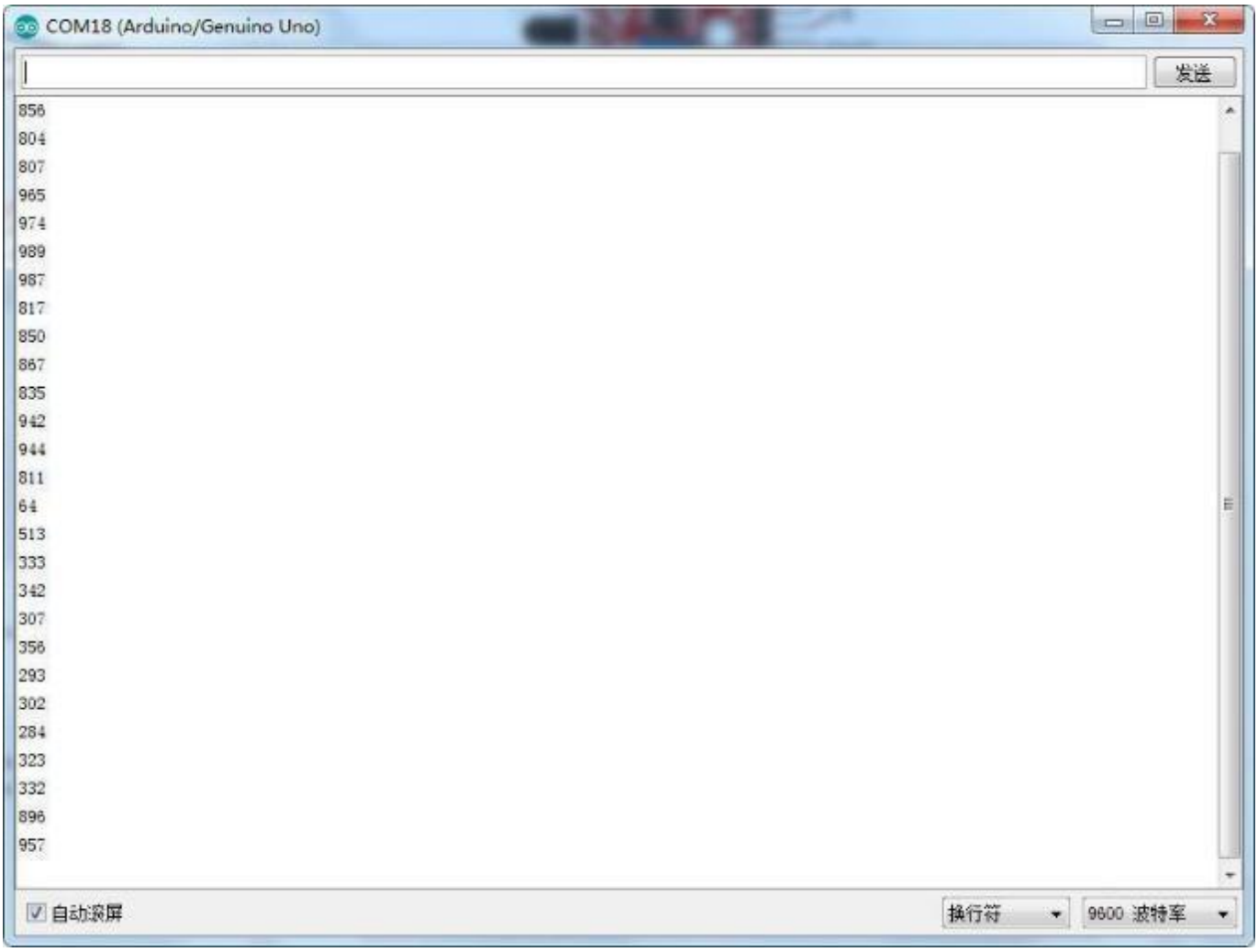
Für das Flammensensormodul können wir zwischen zwei Ausgängen wählen: Digitalausgang oder Analogausgang. In der folgenden Abbildung verwenden wir den Digitalausgang. So können wir sehen, dass wenn der Flammensensor die Flamme erfasst, die LED am Arduino aufleuchtet.

Beispielbild



Im folgenden Bild verwenden wir den Analogausgang. In diesem Fall können wir die Stärke der Flamme messen in einem Bereich zwischen 0 und 1023. Das übernächste Bild zeigt die Ausgabe der Analogwerte im seriellen Monitor.




COM18 (Arduino/Genuino Uno)
发送
856
804
807
965
974
989
987
817
850
867
835
942
944
811
64
513
333
342
307
356
293
302
284
323
332
896
957
自动滚屏
换行符
9600 波特率

Im Folgenden finden Sie den Code und einige Erklärungen dazu

(1)Digitalausgang

```
//Die eingebaute LED des Arduino verwenden wir zu Ausgabe
int Led=13;
//Das Flammensensormodul schließen wir an Pin 3 an.
int buttonpin=3;
int    val;
void    setup()
{
        pinMode(Led,OUTPUT);
        pinMode(buttonpin,INPUT);
}
void    loop()
{
        //Einlesen des Ausgangs des Flammensensormoduls
        val=digitalRead(buttonpin);
        //Wenn das Modul eine Flamme erkennt, LED an Pin13 aufleuchten lassen …
        if(val==HIGH)
        {
                digitalWrite(Led,HIGH);
        }
        Else //wenn das Modul keine Flamme erkennt, LED wieder ausschalten
        {
                digitalWrite(Led,LOW);
        }
}
```

(2)Analogausgang

```
// Das Flammensensormodul schließen wir an Pin A0 an.
int    sensorPin    =    A0;

// Die eingebaute LED des Arduino verwenden wir zu Ausgabe
int    ledPin    =    13;
int    sensorValue =    0;


void setup()
{
      pinMode(ledPin,OUTPUT);
      Serial.begin(9600);
}
void loop()
{
      sensorValue = analogRead(sensorPin); //Analogwert vom Sensor einlesen per ADC
      //LED einschalten. Je stärker eine Flamme erkannt wird, desto länger bleibt die LED an
      digitalWrite(ledPin, HIGH);
      delay(sensorValue);
      //LED ausschalten. Je stärker eine Flamme erkannt wird, desto länger bleibt die LED aus
      digitalWrite(ledPin, LOW);
      delay(sensorValue);
      Serial.println(sensorValue, DEC);
      //Je stärker eine Flamme erkannt wird, desto langsamer blinkt also die LED
}
```

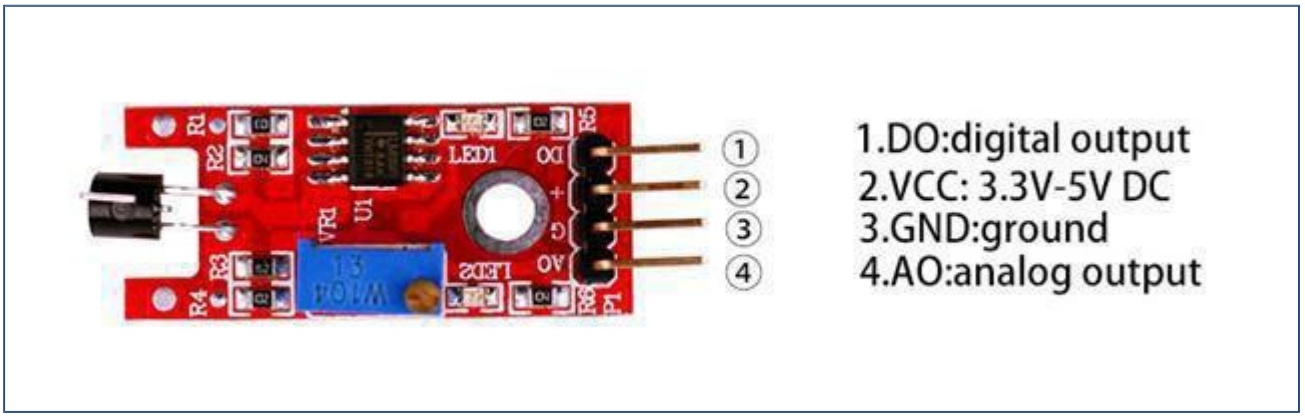
Lektion 21 BERÜHRUNGSSENSITIVER SCHALTER

Überblick

In diesem Versuch lernen wir, wie man ein berührungssensitives Modul benutzt.

Berührungssensitiver Schalter

Durch Berühren des Sensor-Pins wird ein high Signal am 'DO'-Pin erzeugt. Der Ausgang ist kein sauberes Signal, sondern enthält 50 Hz netzinduzierte Signale ("Netzbrummen"). Das Ausgangssignal ist high-aktiv und die Empfindlichkeit der Schaltung kann mit einem Potentiometer eingestellt werden. Am Pin 'AO' steht ein analoges Ausgangssignal des Sensors zur Verfügung.



Benötigte Komponenten:

1 x Elegoo Uno R3

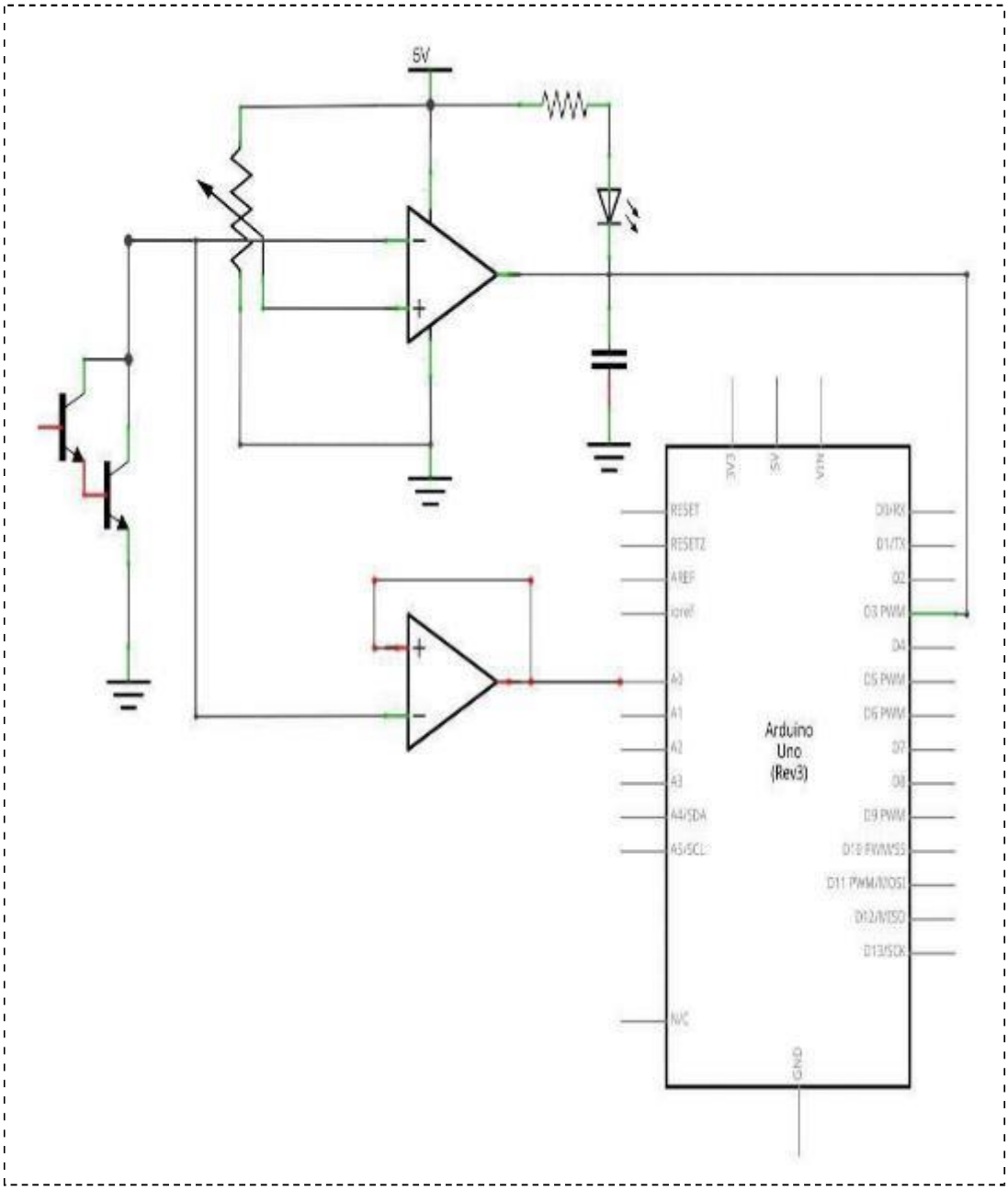
1 x USB Kabel

1 x Berührungssensitives Modul

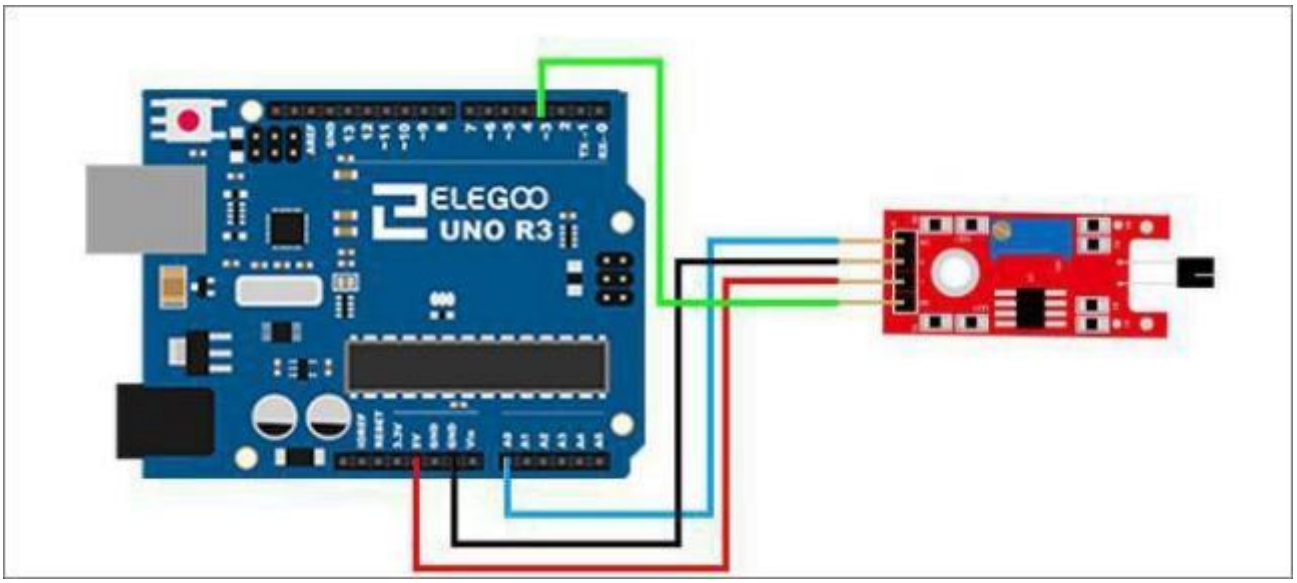
4 x F-M Drähte

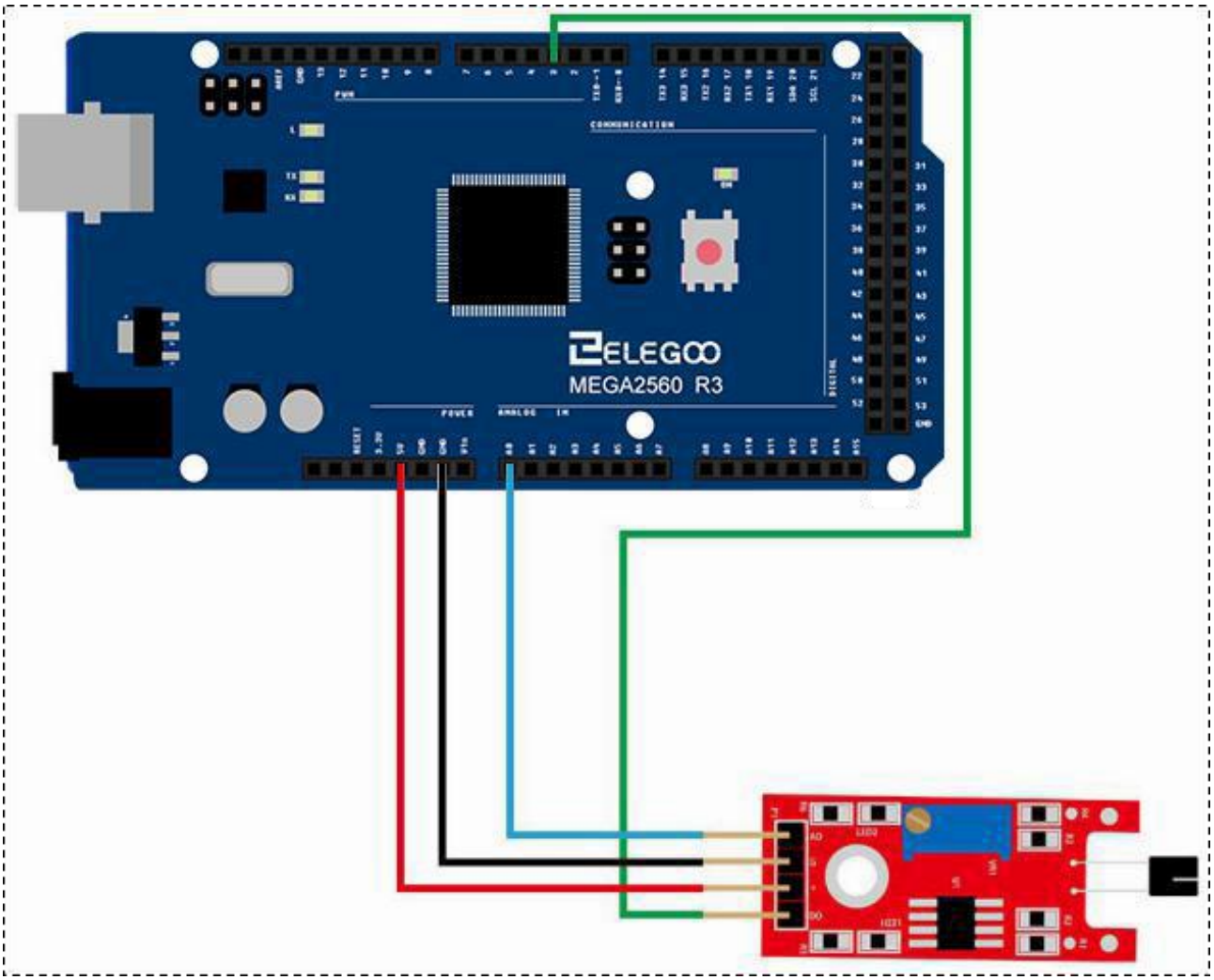
Verbindung

Schaltplan



Verdrahtungsplan

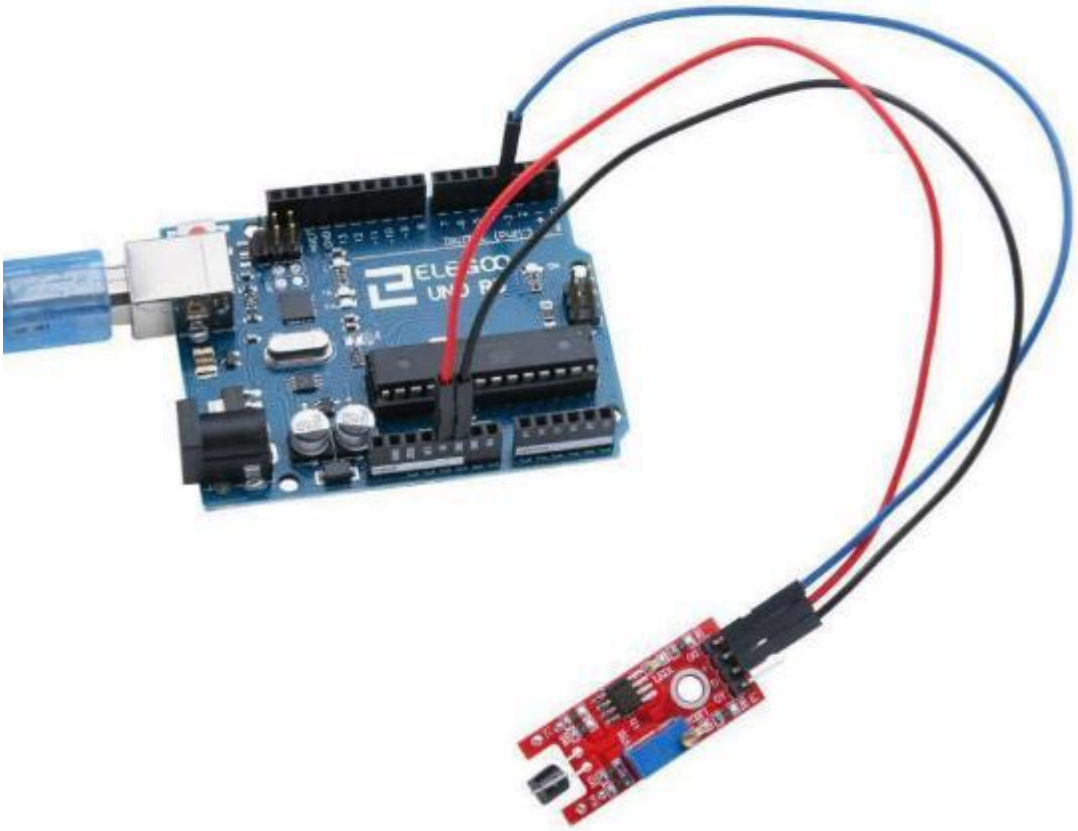




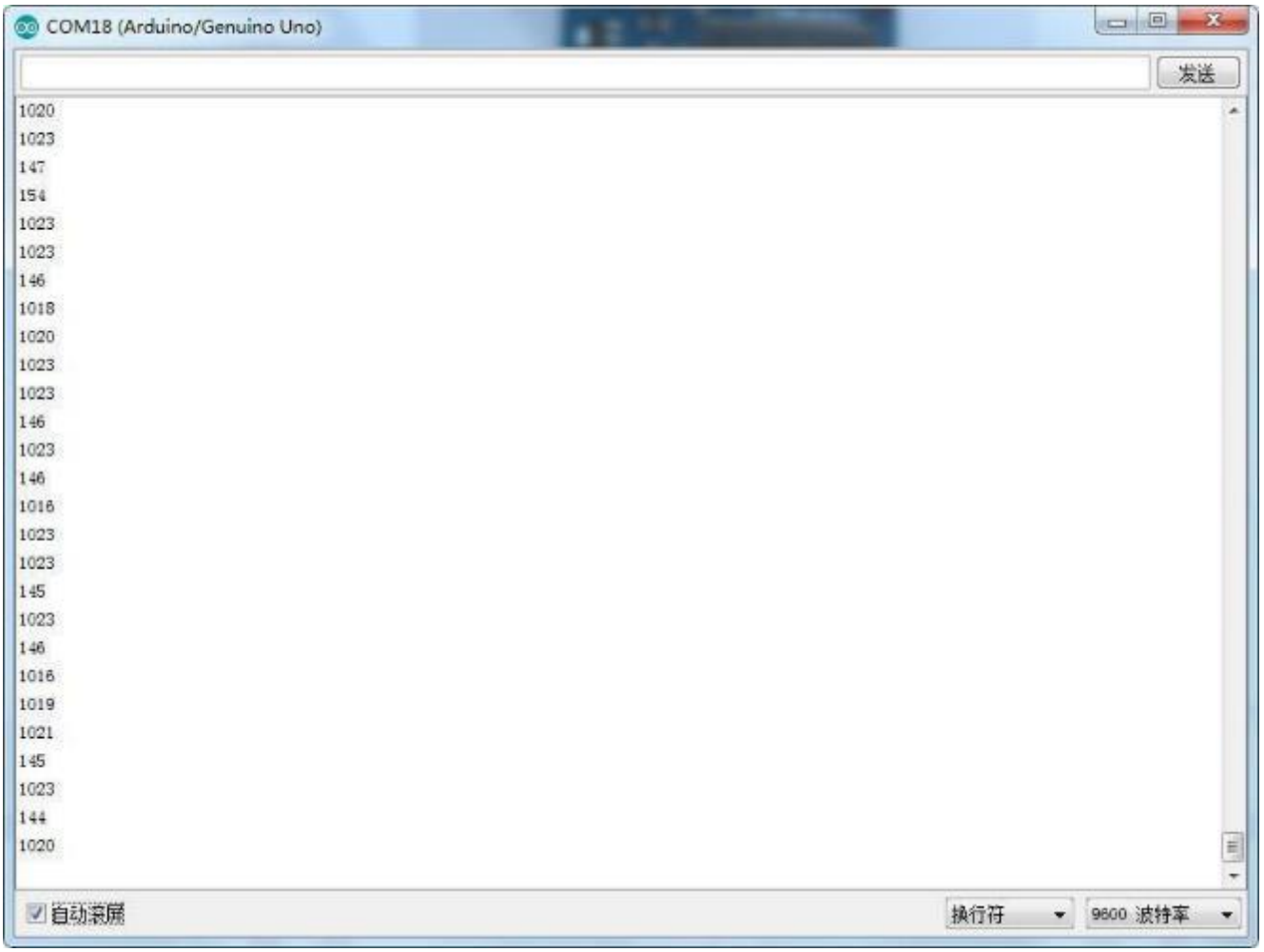
Code

Nachdem wir die Schaltung angeschlossen haben, öffnen Sie den Ordner "Code" und laden Sie das Programm "Lesson 21 METAL TOUCH MODULE" auf den Arduino. Für das Modul haben wir die Möglichkeit, entweder den Digitalausgang oder den Analogausgang zu benutzen. In der folgenden Abbildung verwenden wir den digitalen Ausgang.

Beispielbild



Im folgenden Bild verwenden wir den AO-Port zur Ausgabe. Je nach Berührung werden Zahlen zwischen 0 und1023 ausgegeben.



Im Folgenden finden Sie den Code und einige Erklärungen.

（1）Digitalausgang

```
//Die eingebaute LED des Arduino verwenden wir zu Ausgabe
int Led=13;
//Das Modul schließen wir an Pin 3 an.
int buttonpin=3;
int    val;
void    setup()
{
      pinMode(Led,OUTPUT);
      pinMode(buttonpin,INPUT);
}
void    loop()
{
      //Einlesen des Ausgangs des Moduls
      val=digitalRead(buttonpin);
      //Wenn das Modul eine Berührung erkennt, LED an Pin13 aufleuchten lassen …
      if(val==HIGH)
      {
            digitalWrite(Led,HIGH);
      }
      else //wenn das Modul keine Berührung erkennt, LED wieder ausschalten
      {
            digitalWrite(Led,LOW);
      }
}
```

（2）Analogausgang

```
// Das Flammensensormodul schließen wir an Pin A0 an.
int sensorPin    = A0;
// Die eingebaute LED des Arduino verwenden wir zu Ausgabe
int ledPin    = 13;
int sensorValue = 0;
void setup()
{
      pinMode(ledPin,OUTPUT);
      Serial.begin(9600);
}
void loop()
{
      sensorValue = analogRead(sensorPin); //Analogwert vom Sensor einlesen per ADC
      //LED einschalten. Je stärker eine Berührung erkannt wird, desto länger bleibt die LED an
      digitalWrite(ledPin, HIGH);
      delay(sensorValue);
      //LED ausschalten. Je stärker eine Berührung erkannt wird, desto länger bleibt die LED aus
      digitalWrite(ledPin, LOW);
      delay(sensorValue);
      Serial.println(sensorValue, DEC);
      //Je stärker eine Berührung erkannt wird, desto langsamer blinkt also die LED

}
```

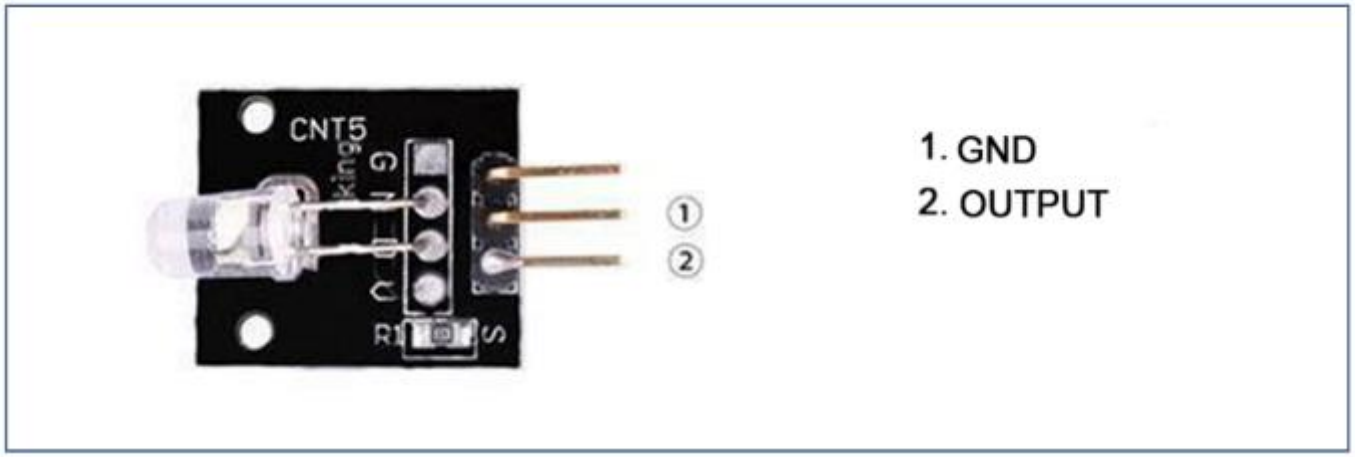
Lektion 22 7 FARBEN FLASH LED MODUL

Überblick

In diesem Versuchen lernen wir ein 7 Farben Flash Modul zu verwenden.

7 Farben Flash

Klare 5mm LED für direkten Betrieb an 5V, die LED-Farbe wechselt automatisch durch eine siebenfarbige Sequenz. Die 5-V-Versorgung wird an den S-Pin angeschlossen und Ground am mittleren Pin.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

1 x 7 Farben Flash LED Modul

2 x F-M Drähte

Komponenteneinführung

7 Farben Flash Led:

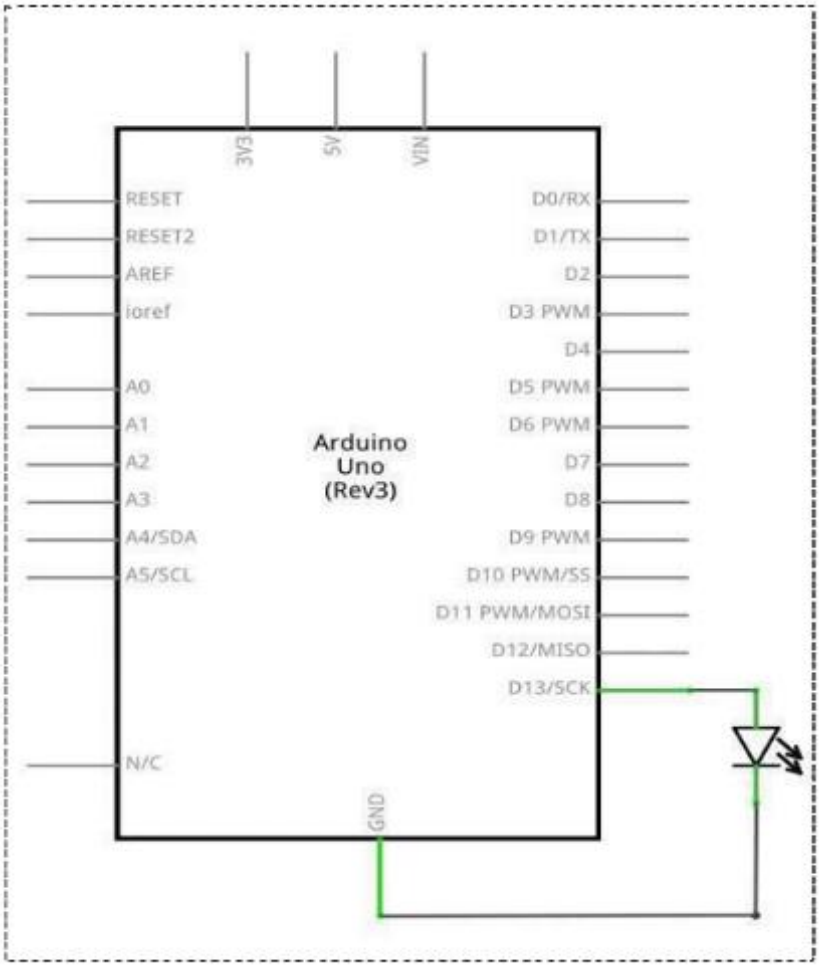
Das Modul enthält eine 7-farbige 5mm LED mit einem eingebauten Chip, der jede Farbe nacheinander blinken lässt.

Vorwärtsspannung: 2,5V - 6V

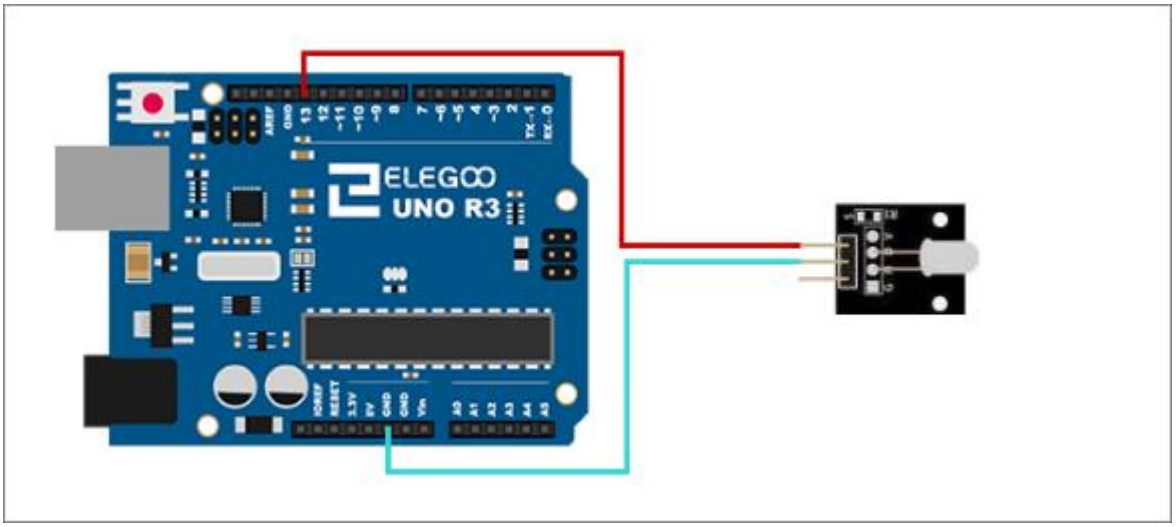
Durchlassstrom: 40mA

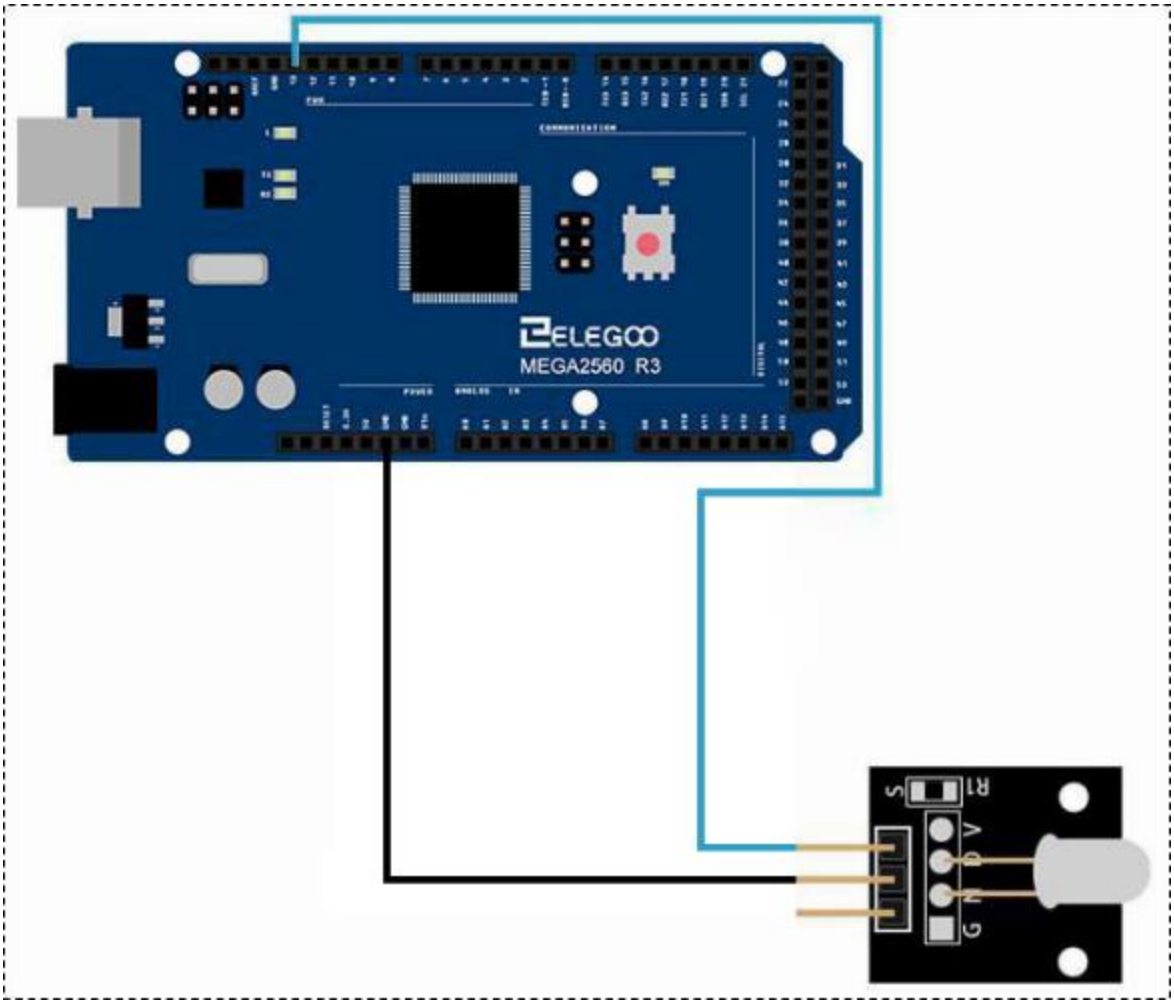
Verbindung

Schaltplan



Verdrahtungsplan

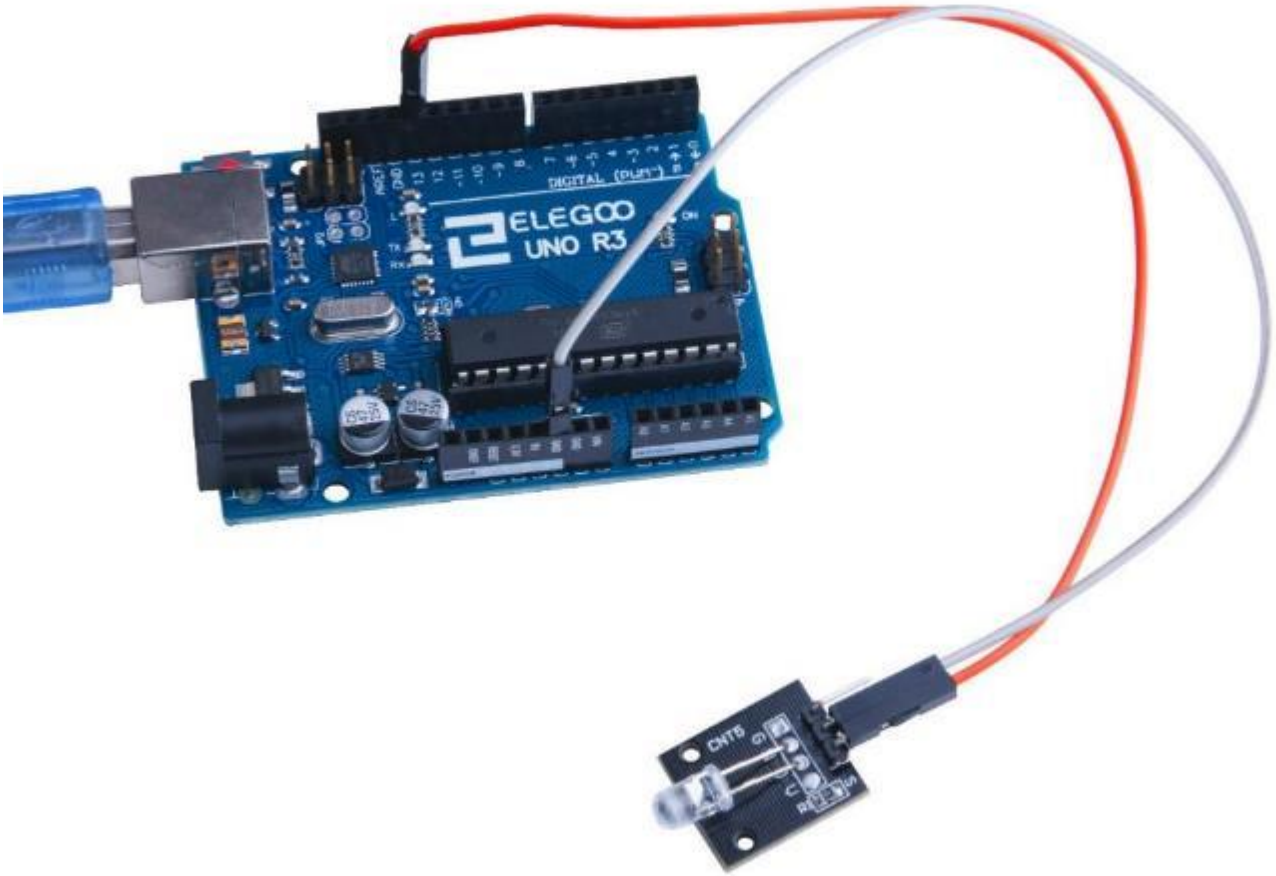




Code

Nachdem wir die Schaltung aufgebaut haben, laden Sie bitte das Programm "Lesson 22 7 COLOR FLASH LED MODULE" auf den Arduino. Wir können nun das Farbwechselspiel der LED beobachten.

Beispielbild



Im Folgenden finden Sie den einfachen Code

```
void setup()
{
    // Wir benutzen Port 13 als Ausgang um das Modul mit Spannung zu versorgen
    pinMode(13, OUTPUT);
}

void loop()
{
    // Port 13 liegt während der gesamten Programmausführung auf 5V. Dies ist möglich, weil der
    // Farbwechsel vom Modul selbst ausgeführt wird
    digitalWrite(13, HIGH);
}
```

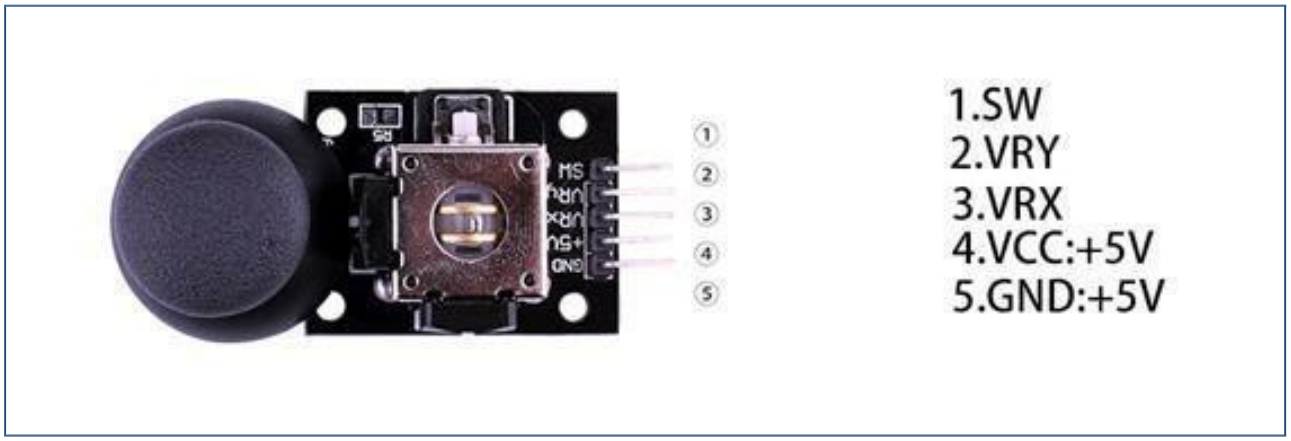
Lektion 23 JOYSTICK MODUL

Überblick

Das Modul funktioniert wie ein Joystick auf einer Spielkonsole. Mit diesem Joystick Modul können Sie x-, y- und z-Koordinaten steuern. Es kann als Kombination von zwei Potentiometern und einem Taster betrachtet werden. In x- und y-Richtung handelt es sich um analoge Eingangssignale und in z-Richtung um ein digitales Signal. Dadurch werden die x- und y-Achsen mit analogen Pins des Arduino verbunden, während der z-Port mit einem digitalen Pin verbunden wird. Das heißt weiterhin, dass die z-Achse nur gedrückt bzw. nicht gedrückt kennt, wohingegen in x- und y-Richtung 1024 Abstufungen erkannt werden.

Joystick

Ein analoger 2-Achsen-Joystick mit 2x 10K Ohm Potis und Tasterfunktion. Die Steckerbezeichnungen sind auf der Platine aufgedruckt. Ein aufsteckbarer Bedienknauf ist im Lieferumfang des Moduls enthalten.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

1 x Joystickmodul

5 x F-M Drähte

Komponenteneinführung

Joysticksensor:

Viele Roboterprojekte benötigen einen Joystick. Dieses Modul bietet eine kostengünstige Lösung. Durch einfaches Anschließen an zwei analoge Eingänge können Sie Ihren Roboter in X, Y-Richtung steuern. Das Modul hat auch zusätzlich einen Schalter, der mit einem digitalen Pin verbunden werden kann.

Spezifikation

Versorgungsspannung: 3,3V bis 5V

Schnittstelle: Analog x2, Digital x1 Größe: 40*28mm

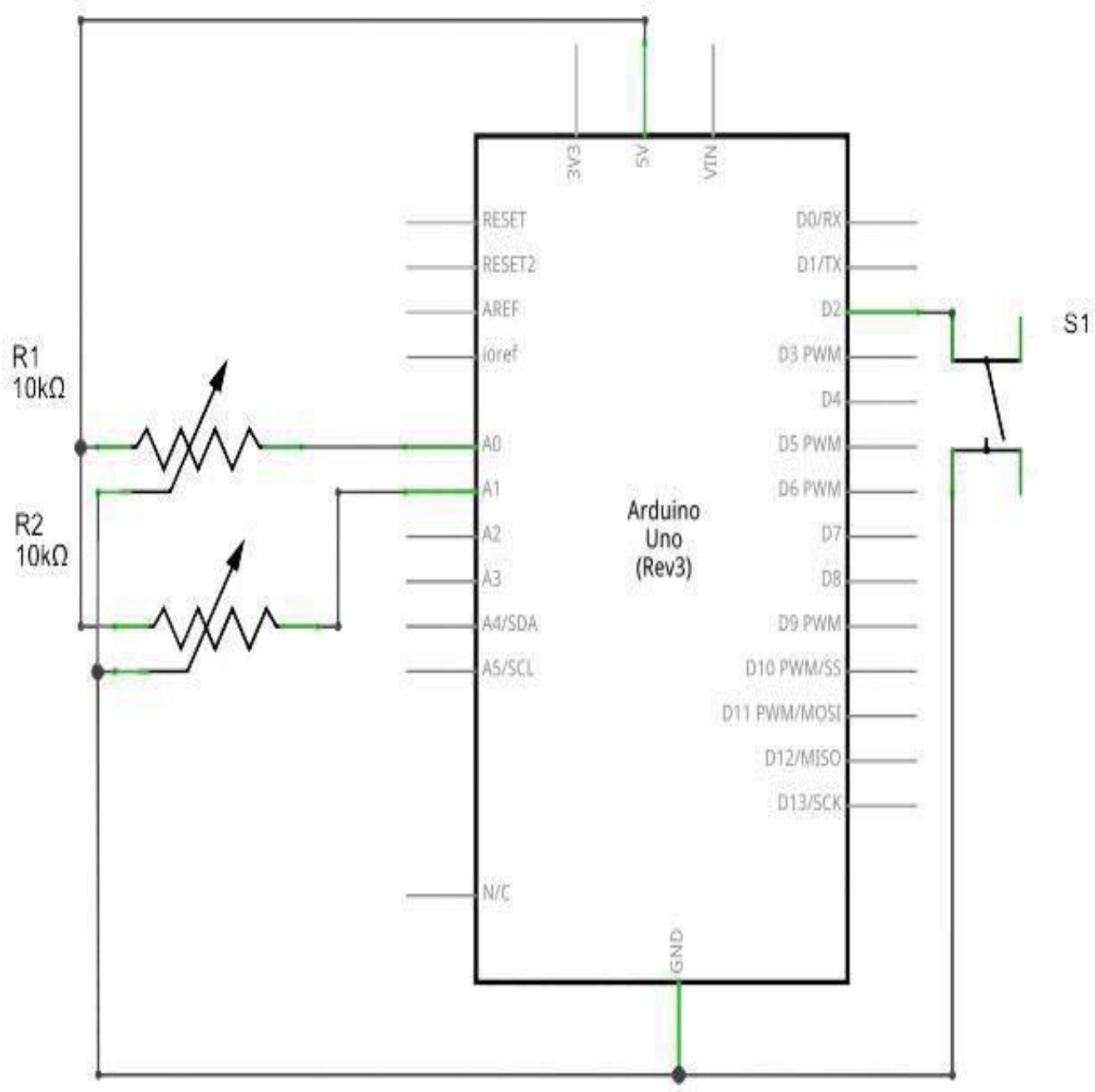
Gewicht: 12g

Das Modul hat 5 Pins: Vcc, Masse, X, Y, Taster (SW). Beachten Sie, dass sich die Beschriftungen auf Ihrem Modul leicht unterscheiden können je nachdem woher Sie das Modul erhalten haben. Der Daumenstick ist analog und sollte genauere Messwerte liefern als einfache digitale Joysticks, die einfach nur links, rechts, mitte als Wert liefern können. Zusätzlich können Sie den Joystick nach unten drücken um einen Taster zu aktivieren.

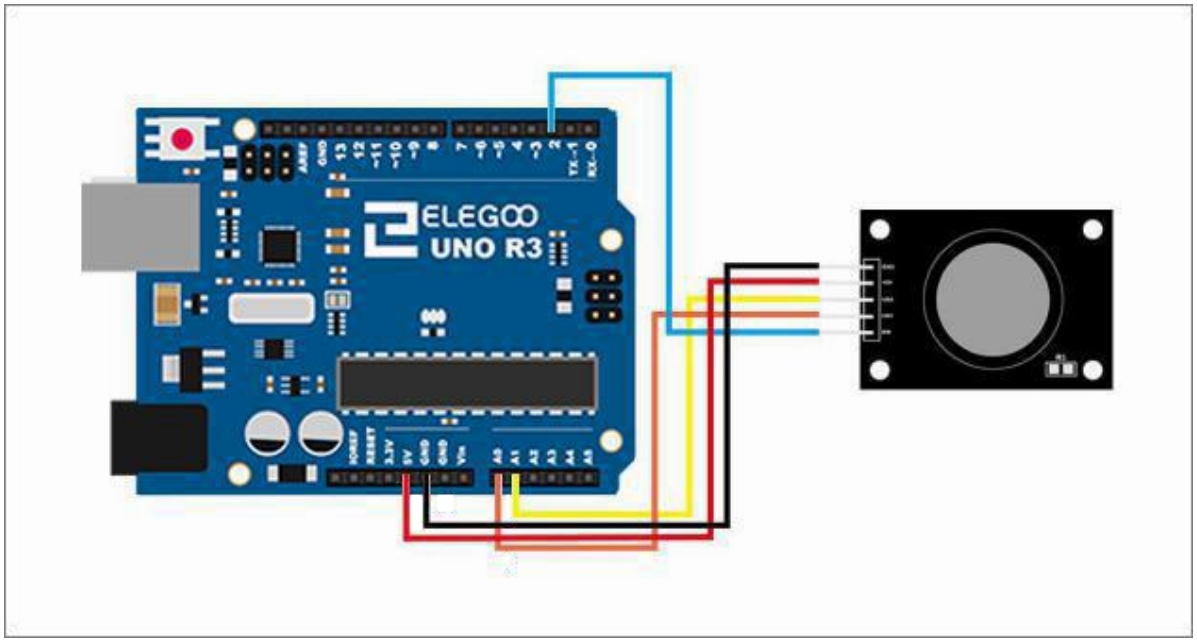
Wir müssen analoge Arduino-Pins verwenden, um die Daten von den X/Y-Pins zu lesen, und einen digitalen Pin um den Tasterzustand zu lesen. Der Taster ist mit Masse verbunden, wenn der Joystick gedrückt wird, ansonsten hängt er in der Luft. Um sinnvolle Messwerte vom Taster zu erhalten, muss dieser über einen Pull-Up Widerstand mit Vcc verbunden werden.

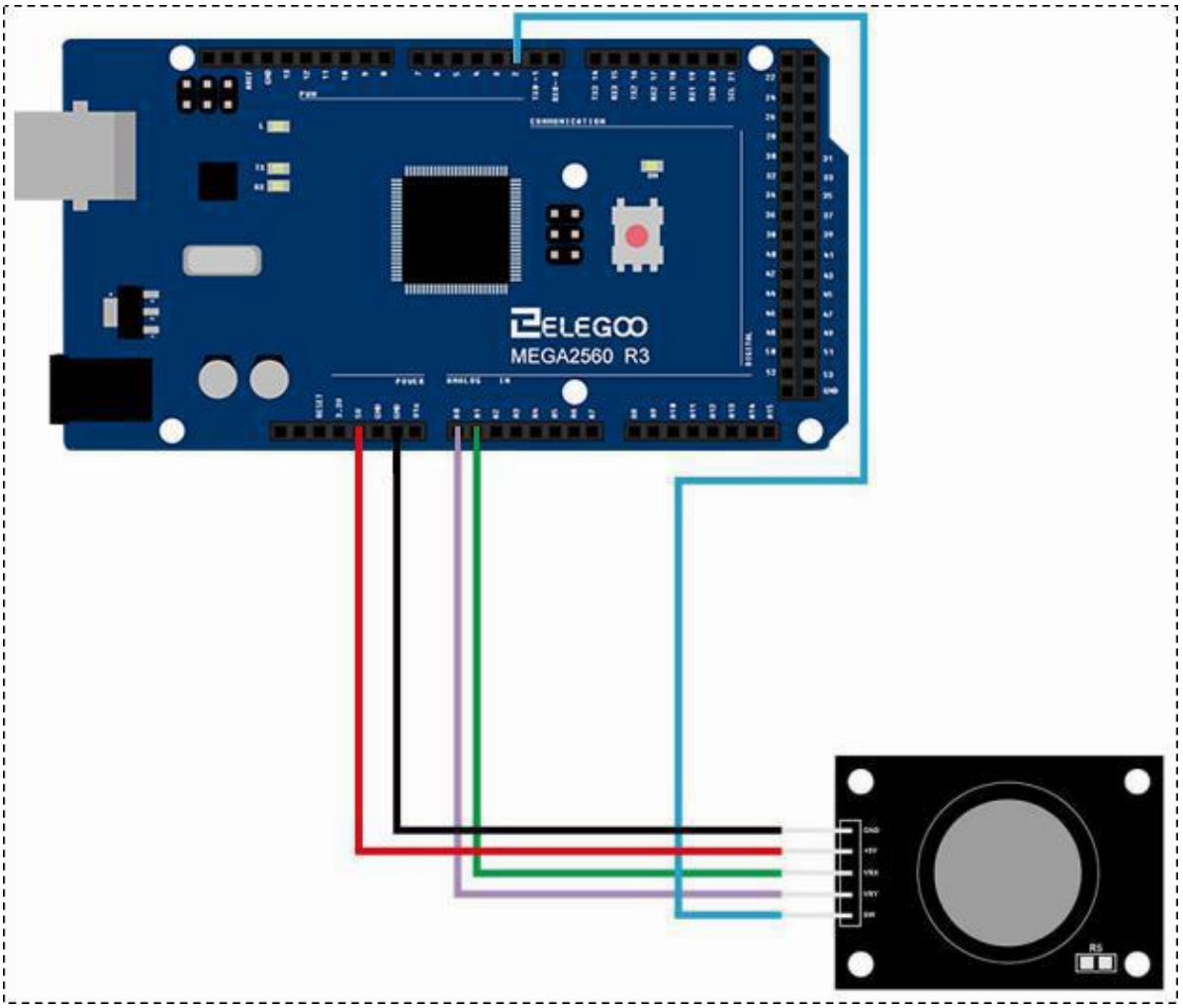
Verbindung

Schaltplan



Verdrahtungsplan

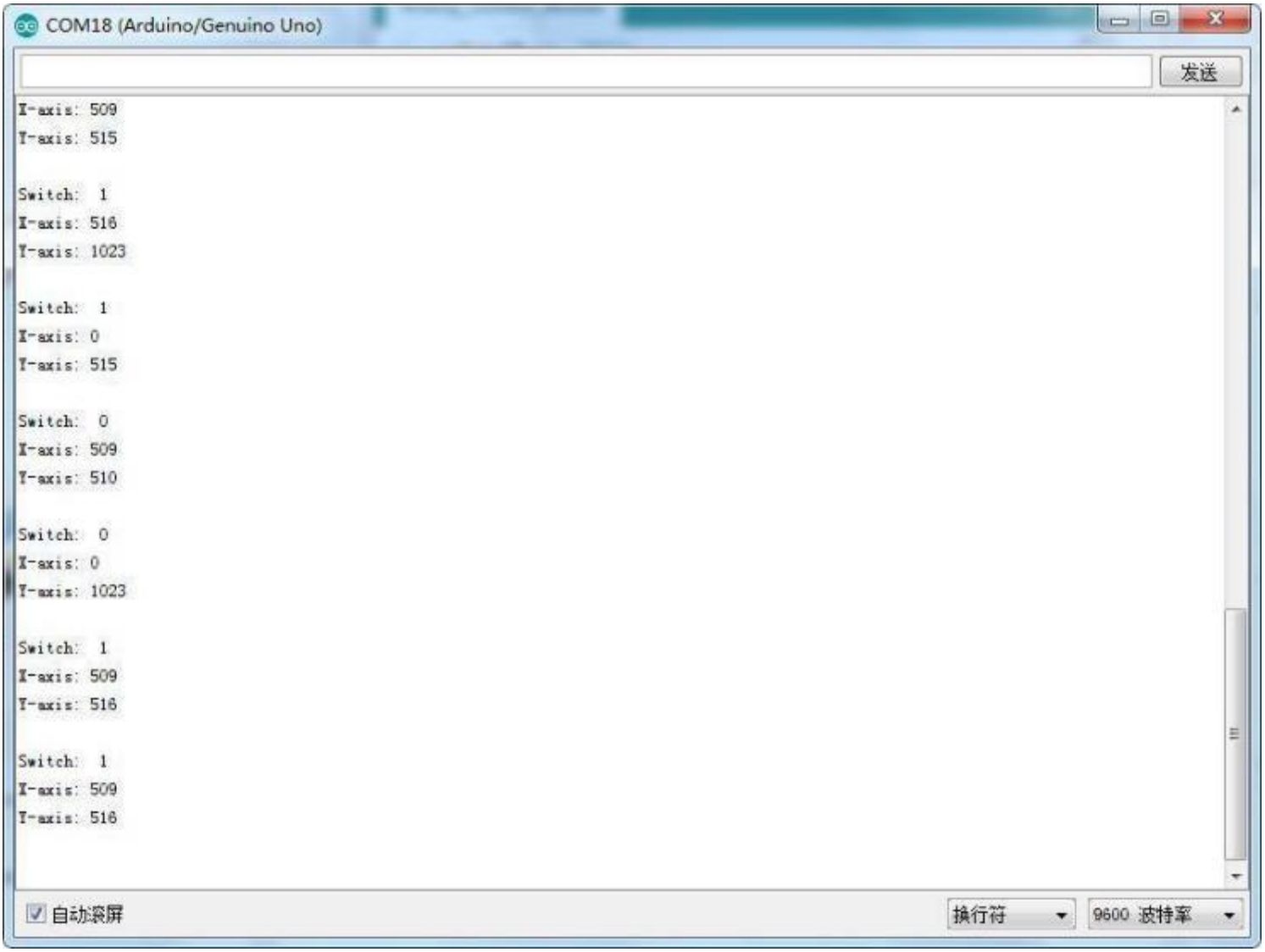




Code

Analoge Joysticks sind prinzipiell Potentiometer. Wenn sich der Joystick in der Ruhe- oder Neutralstellung befindet, sollte ein Wert von etwa 511/512 zurückgegeben werden, also der Hälfte von 1023. Zusätzlich ist am Pin SW (Z-Achse) ein digitales Signal vorhanden, das mit einem digitalen Port verbunden ist und einen Pull-Up Widerstand integriert hat. Werte von SW: ‚high' bedeutet nicht gedrückt, ‚low' bedeutet gedrückt.
Nachdem Sie die Schaltung verbunden haben, laden Sie bitte das Programm "Lesson 23 JOYSTICK MODULE" auf den Arduino. Der serielle Monitor könnte wie folgt aussehen:

Beispielbild


COM18 (Arduino/Genuino Uno)
发送
X-axis: 509
Y-axis: 515
Switch: 1
X-axis: 516
Y-axis: 1023
Switch: 1
X-axis: 0
Y-axis: 515
Switch: 0
X-axis: 509
Y-axis: 510
Switch: 0
X-axis: 0
Y-axis: 1023
Switch: 1
X-axis: 509
Y-axis: 516
Switch: 1
X-axis: 509
Y-axis: 516
自动滚屏
换行符
9600 波特率

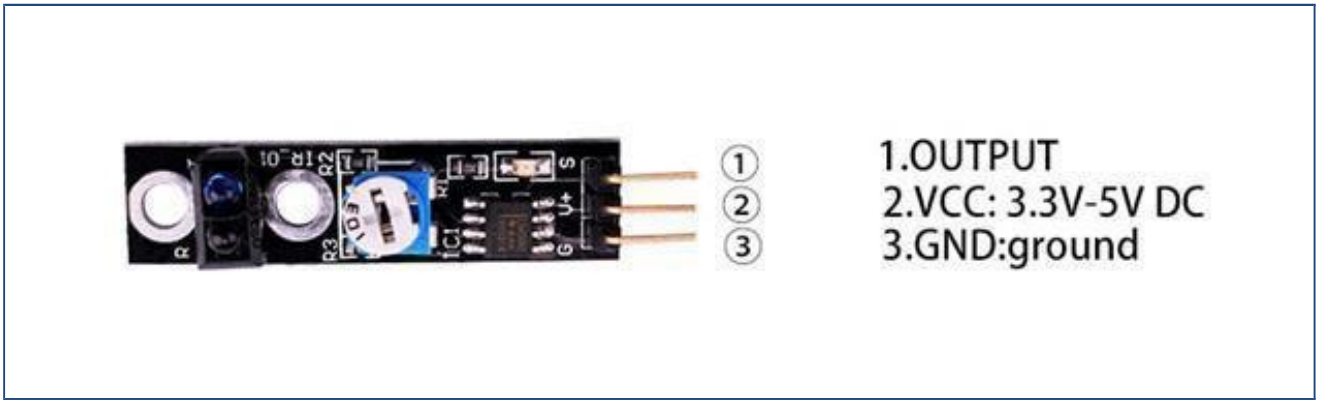
Im Folgenden finden Sie den Code und einige Erklärungen

```
// Pin, der mit dem z-Achsen Schalter verbunden ist
const int SW_pin = 2;
// Pin, der mit der X-Achse verbunden ist
const int X_pin = A1;
// Pin, der mit der Y-Achse verbunden ist
const int Y_pin = A0;

void setup() {
  pinMode(SW_pin, INPUT);
  digitalWrite(SW_pin, HIGH);
  Serial.begin(9600);
}

void loop() {
  // Das Programm fragt zweimal pro Sekunde die Werte für alle drei Achsen vom Modul ab und gibt
  // sie über die serielle Schnittstelle aus
  Serial.print("Switch:  ");
  Serial.print(digitalRead(SW_pin));
  Serial.print("\n");
  Serial.print("X-axis: ");
  Serial.print(analogRead(X_pin));
  Serial.print("\n");
  Serial.print("Y-axis: ");
  Serial.println(analogRead(Y_pin));
  Serial.print("\n");
  delay(500);
}
```

Lektion 24 TRACKING MODUL

Überblick

In diesem Versuch lernen wir, wie man das Tracking-Modul verwendet. Die Infrarotdiode sendet ständig Infrarotlicht aus. Schwarze Gegenstände absorbieren Licht stark. Wenn das Infrarotlicht auf eine schwarze Oberfläche trifft wird weniger Licht reflektiert und somit werden weniger IR-Strahlen vom Phototransistor empfangen. Im Gegensatz dazu reflektieren helle Gegenstände das Licht besser, so dass mehr Licht vom Modul empfangen wird. Das Modul kann also dazu verwendet werden schwarze Gegenstände zu erkennen. Nachdem das Modul eine Lichtschranke enthält, kann damit natürlich auch ein Hindernis erkannt werden. Allerdings beträgt die Reichweite des Moduls nur wenige Millimeter. Ein Hindernis direkt vor den Sender-/Empfängerdioden führt dazu, dass der 'OUTPUT'-Pin nach Masse gezogen wird (active low). Über ein Potentiometer kann die Empfindlichkeit der Schaltung eingestellt werden. Der Erfassungsabstand kann bis zu ca. 1 cm betragen. Die Sende/Empfangsdioden sind nach unten gerichtet. Das Modul zielt also darauf ab, auf den Boden gerichtet zu werden.



Benötigte Komponenten:

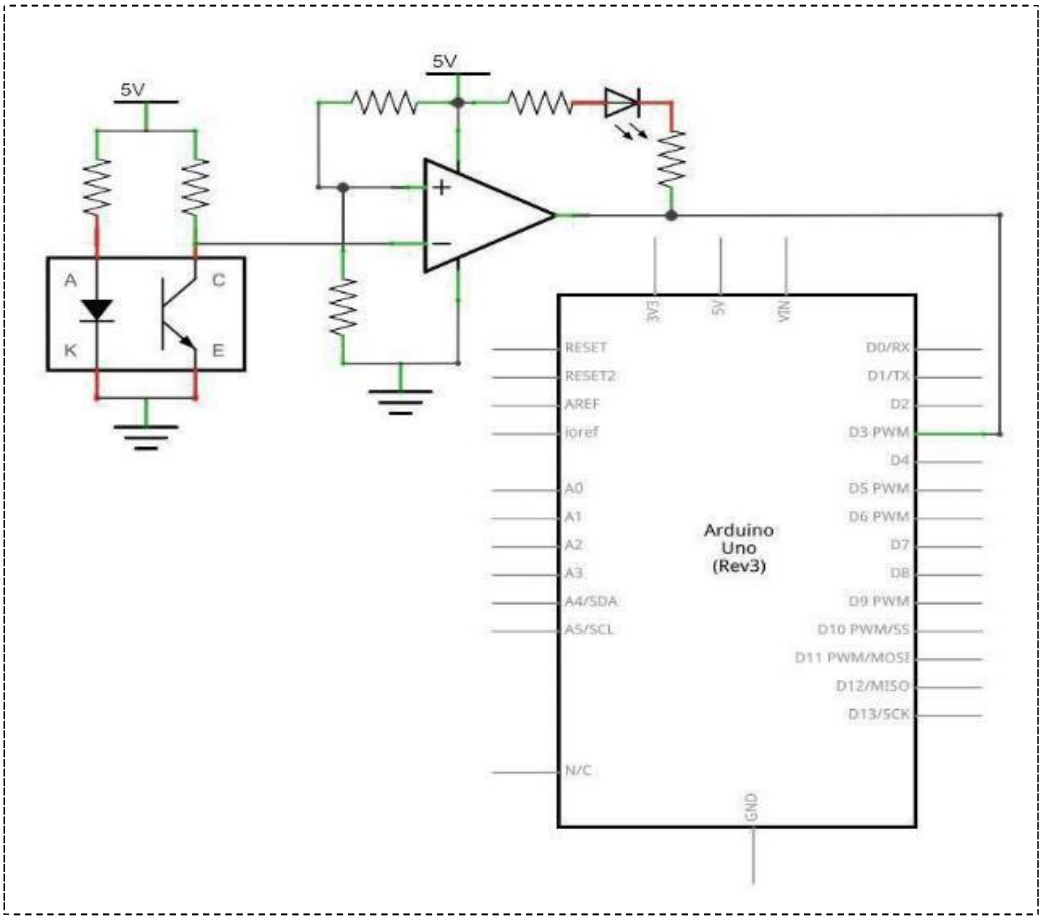
1x Elegoo Uno R3

1x USB Kabel

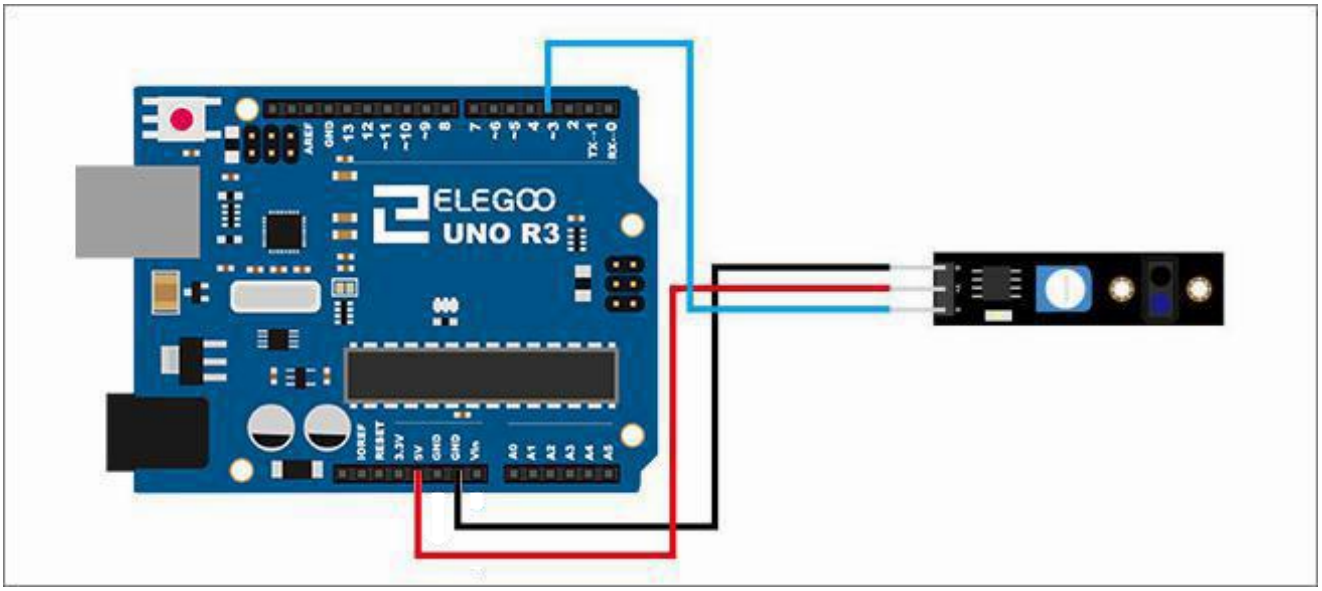
1x Trackingmodul

3x F-M Drähte

Schaltung



Verdrahtungsplan



Code

Nachdem die Schaltung verdrahtet ist laden Sie bitte das Programm "Lesson 24 TRACKING MODULE" auf den Arduino. Es ist normal, dass die auf dem Arduino eingebaute LED (Pin 13) leuchtet, wenn nichts die Lichtschranke des Trackingmoduls behindert. Wenn Sie das Modul auf ein Blatt weißes Papier legen, wird die LED ausgehen (natürlich abhängig davon wie die Empfindlichkeit des Moduls per Potentiometer eingestellt ist). Wenn Sie nun einen dicken schwarzen Strich auf das Papier malen oder das Modul auf eine schwarze Fläche richten, sollte die LED wieder angehen, weil zu wenig Licht von der schwarzen Fläche reflektiert wird.

Beispielbild

Im Folgenden finden Sie den Code und ein paar Erklärungen:

```
int Led=13;
int buttonpin=3;
int val;
void setup()
{
        pinMode(Led,OUTPUT);
        pinMode(buttonpin,INPUT);
        Serial.begin(9600);
}
void  loop()
{
        //Wert des Ausgangs des Moduls lesen
        val=digitalRead(buttonpin);
        if(val==HIGH)
        {
                digitalWrite(Led,HIGH);
                Serial.println("HIGH!");
        }
        else
        {
                digitalWrite(Led,LOW);
                Serial.println("LOW!");
        }
}
```

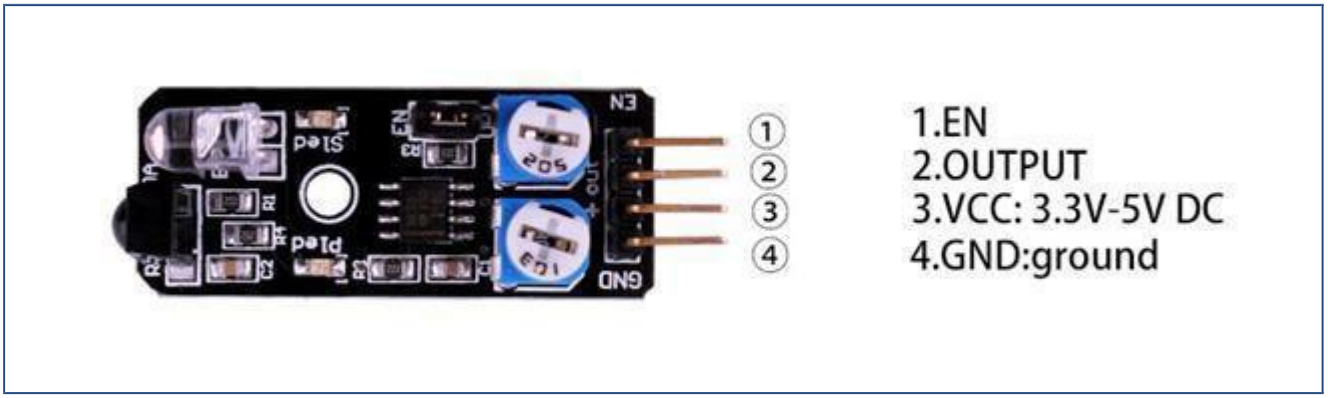
Lektion 25 Infrarot 38 KHz Hindernisvermeidungsmodul

Überblick

Im folgenden Versuch lernen wir ein Hindernisvermeidungsmodul kennen. Es sendet mit 38kHz Signale aus und wertet aus, ob das Signal reflektiert und somit wieder empfangen wurde. Das Modul in der letzten Lektion war mehr zur Linienerkennung auf dem Boden gedacht. Das Modul dieser Lektion ist dazu da Objekte parallel zum Boden zu erkennen. Dies erkennt man auch daran, dass die Sende-LED des Moduls nach vorne zeigt. Insofern ist die Reichweite dieses Moduls auch deutlich größer.

Hindernisvermeidung

IR-Reflexionssensor geeignet zur Hindernisvermeidung. Wenn sich ein Hindernis vor dem IR-Sender/Empfänger befindet, wird der 'Out'-Pin auf low geschaltet. Die Empfindlichkeit der Schaltung kann mit einem Potentiometer eingestellt werden. Für den Dauerbetrieb kann eine Brücke/Jumper (EN) eingesetzt werden. Durch Entfernen der EN-Brücke kann ein externes Logiksignal (am EN-Pin) den Detektor ein- und ausschalten (low = aktiv, high = aus).



Benötigte Komponenten

1x Elegoo Uno R3

1x USB Kabel

1x Hinderniserkennungsmodul

4x F-M Drähte

Komponenteneinführung

Hindernisvermeidungsmodul:

Technische Daten:

Betriebsspannung: DC 3.3V-5V

Betriebsstrom: ≥ 20mA

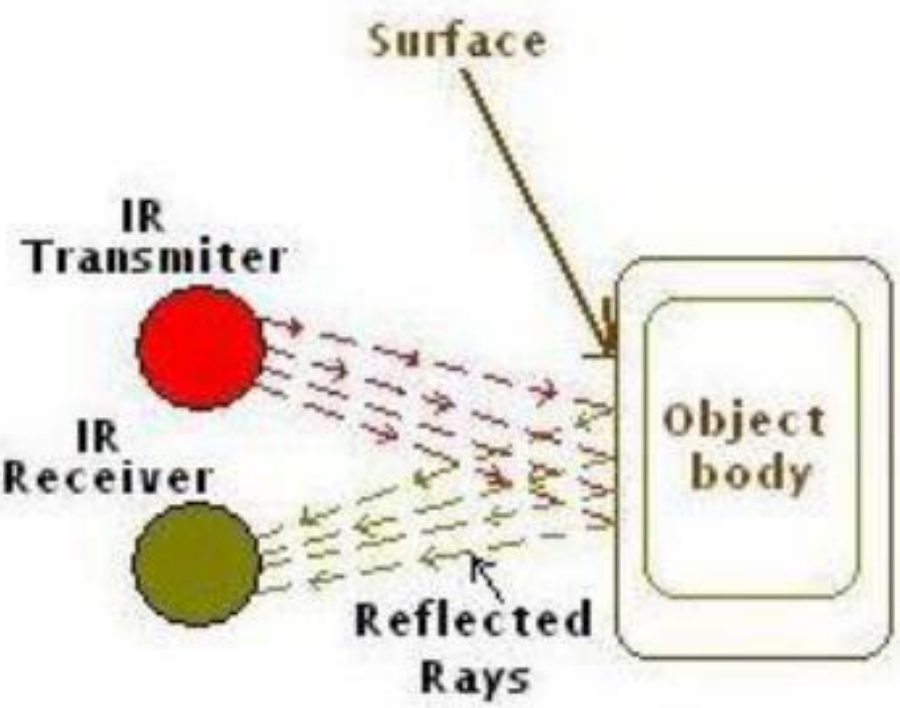
Betriebstemperatur: -10 ℃ - +50 ℃

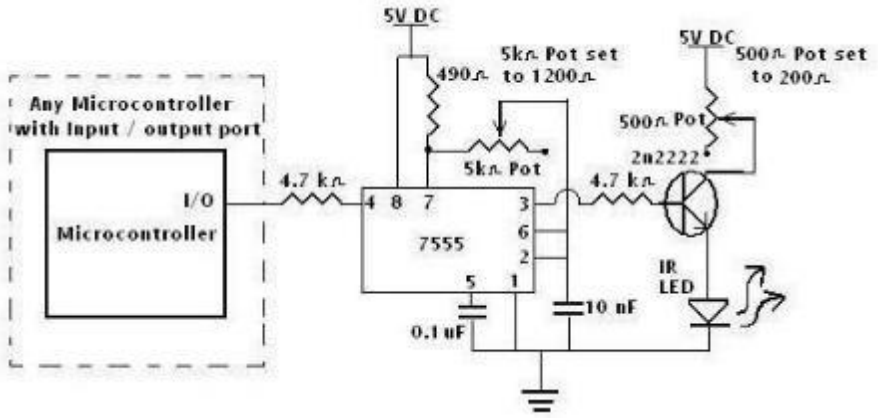
Erfassungsabstand: 2-40cm

IO-Schnittstelle: 4-Draht-Schnittstellen (- / + / S / EN)

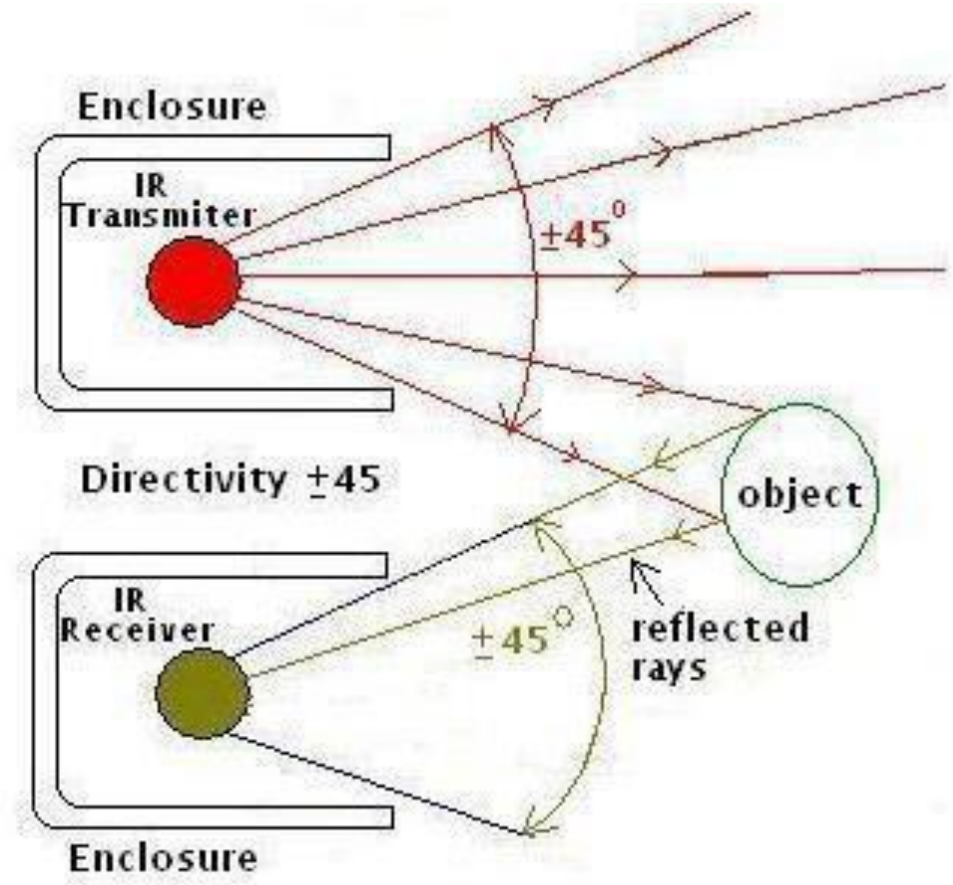
Ausgangssignal: TTL-Pegel (low Pegel bei Hindernis, kein Hindernis high)

Das Grundkonzept der IR(Infrarot)-Hinderniserkennung besteht darin, dass ein IR-Signal ausgesendet wird. Falls das Signal von einem Gegenstand reflektiert wird, wird es von einem Empfänger wiedererkannt.



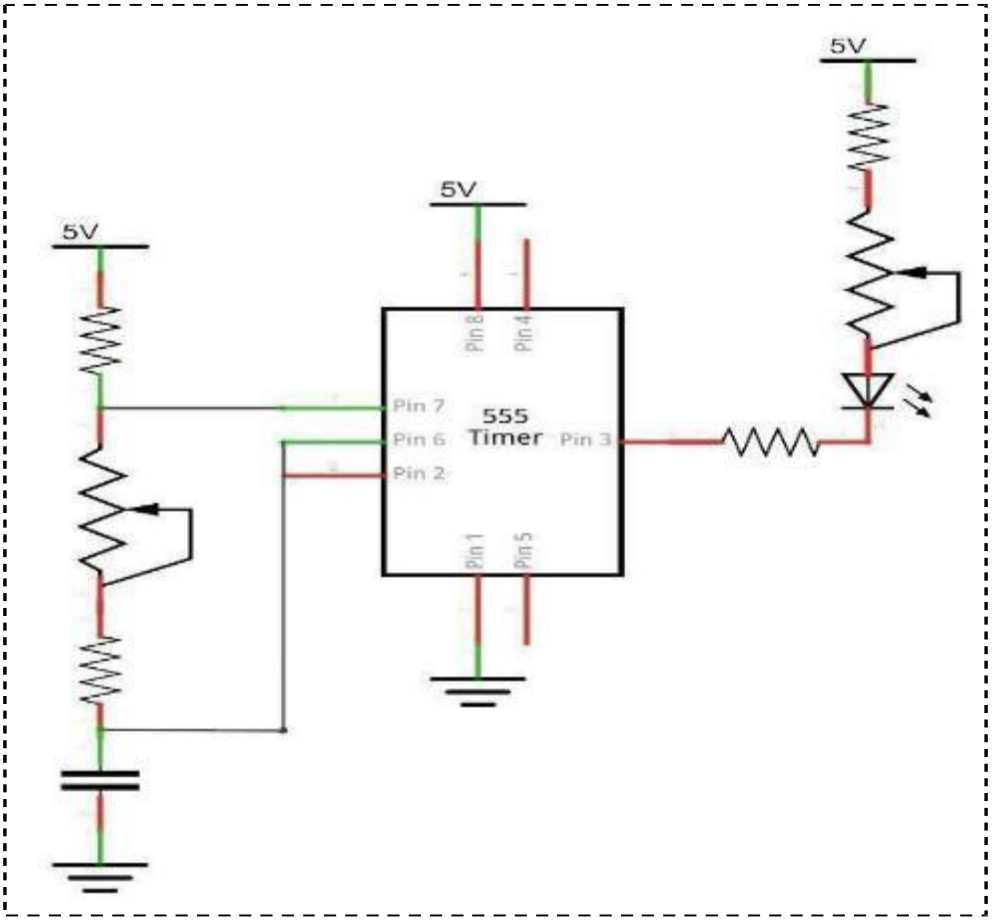


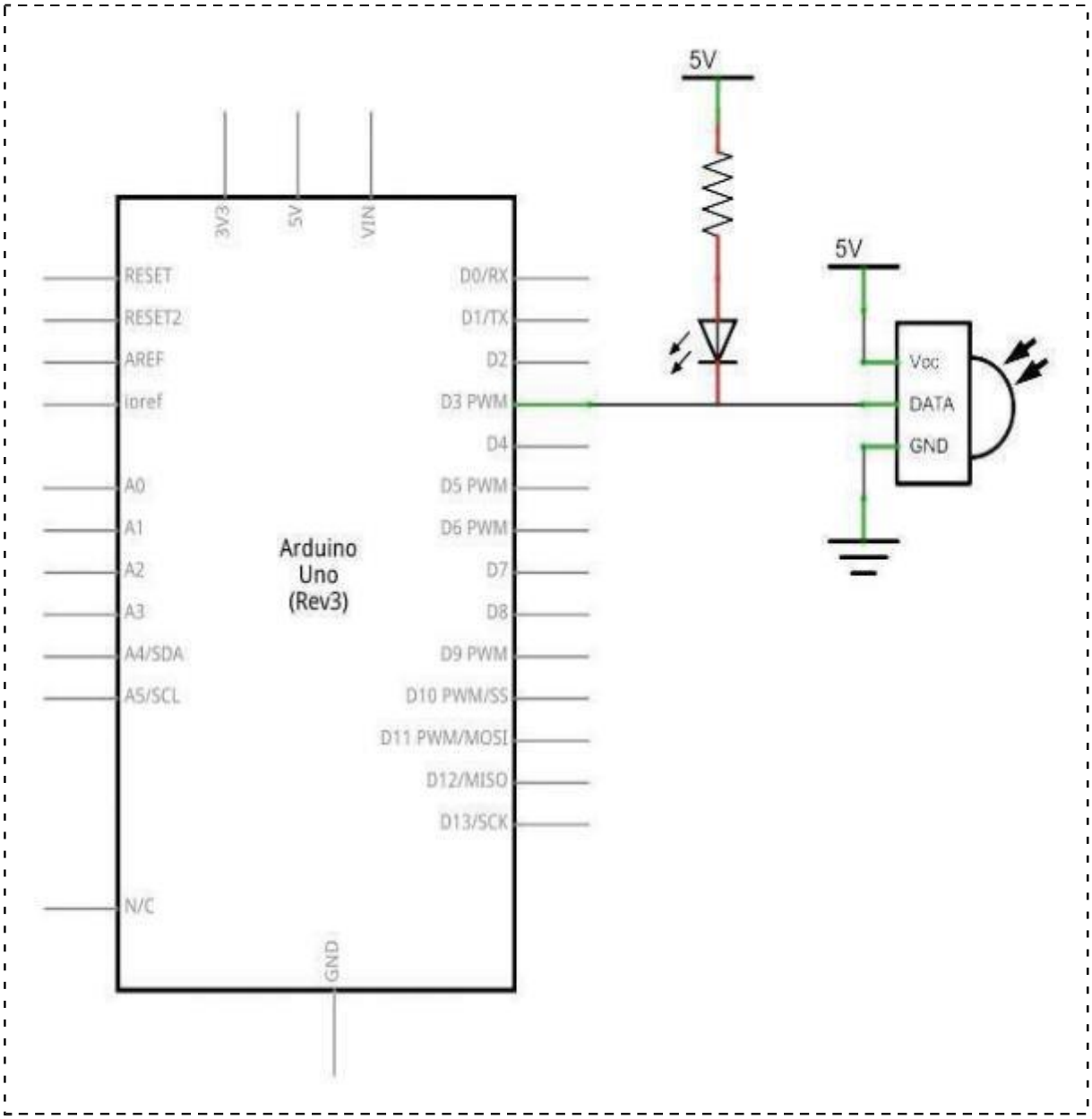
Auf dem Modul befinden sich zwei Potentiometer, von denen eines die Arbeitsfrequenz steuert (Mittenlage bei 38 kHz) und das andere die Empfindlichkeit. Normalerweise gibt es keinen Grund die Arbeitsfrequenz zu justieren.



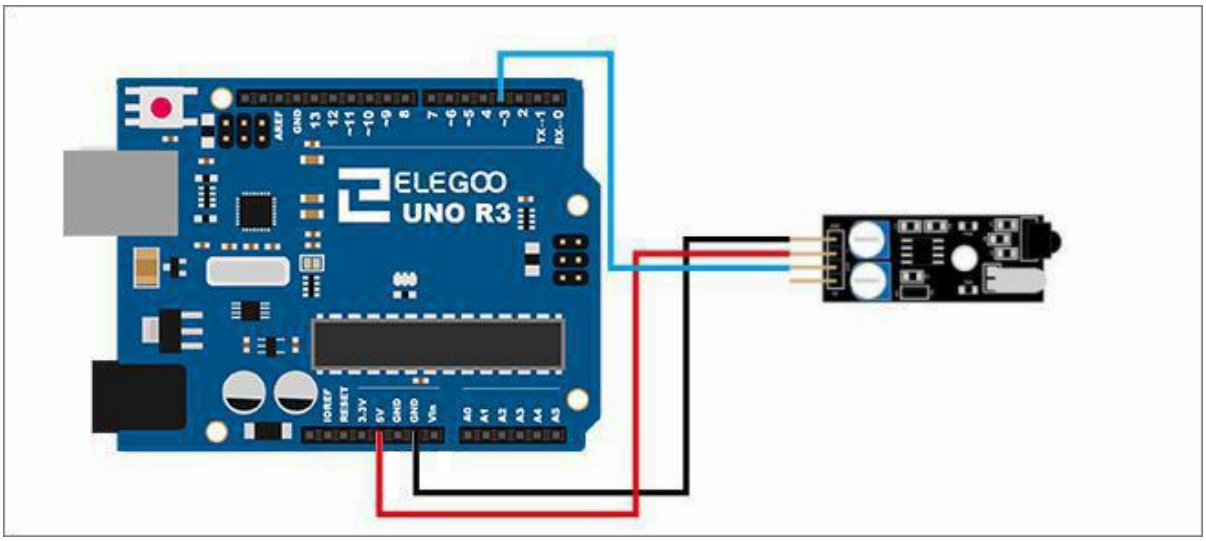
Verbindung

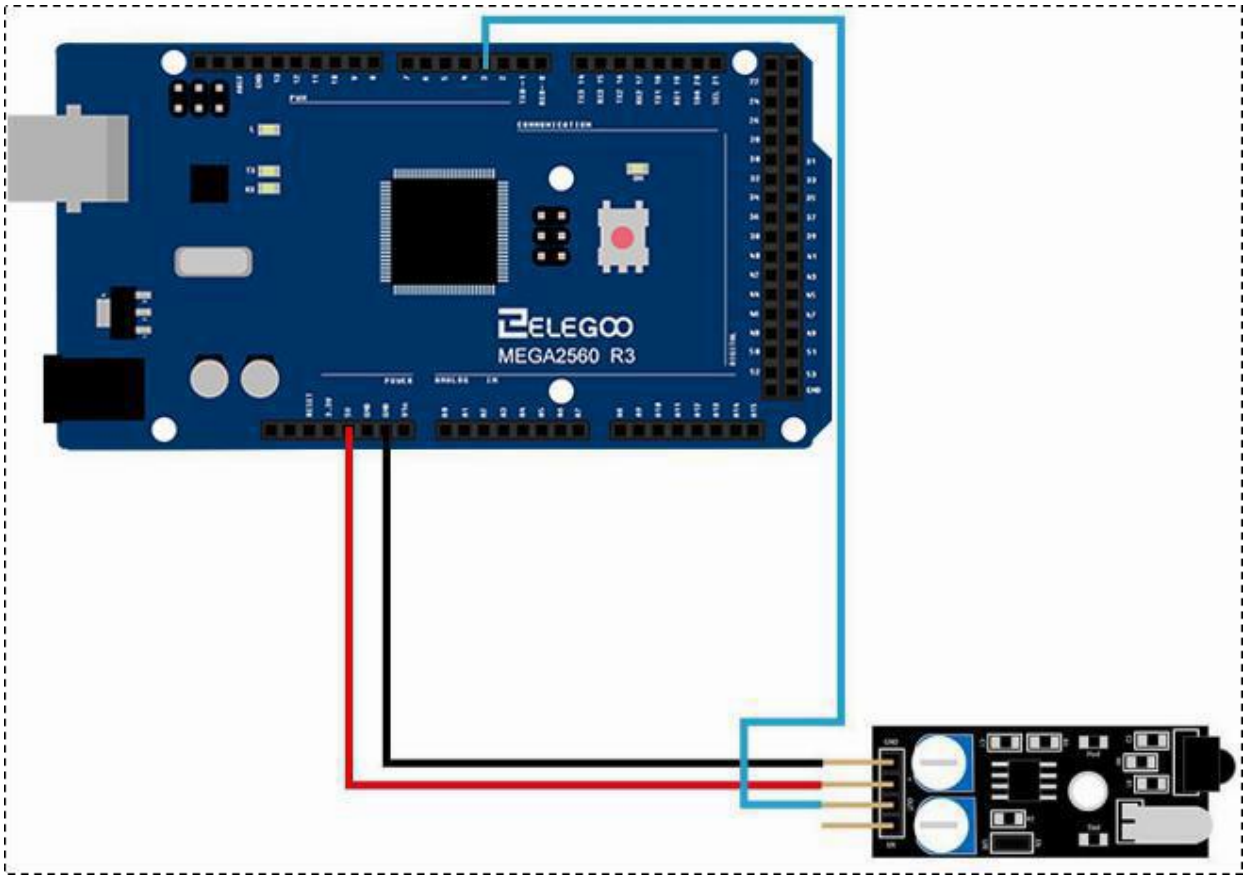
Schaltplan




5V
3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
5V
Vcc
DATA
GND

Verdrahtungsplan





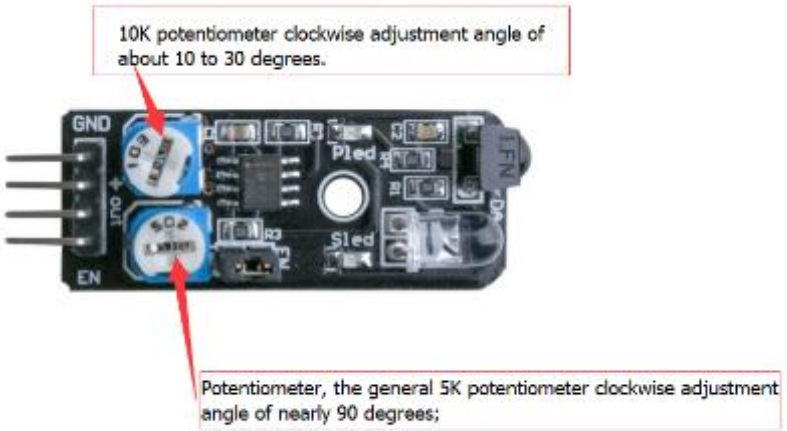
Code

Nachdem Sie die Schaltung verdrahtet haben, laden Sie bitte das Programm "Lesson 25 Infrared 38KHz obstacle avoidance module" auf den Arduino.

Beispielbild

Der Code funktioniert so, dass wenn ein Hindernis erkannt wird vom Modul, ist die eingebaute LED auf dem Arduino ausgeschalten. Wird kein Hindernis erkannt (wenn also freie Bahn ist), leuchtet die LED 13.

Sie müssen zuerst die beiden Potentiometer einstellen, bevor Sie das Hindernisvermeidungsmodul verwenden. Im Allgemeinen müssen Sie das 5K-Potentiometer im Uhrzeigersinn auf etwa 90° und das 10K-Potentiometer auf 10-30° einstellen.



Im Folgenden finden Sie den Code und ein paar Erklärungen:

```
int Led=13;
int buttonpin=3;
int val;
void setup()
{
    pinMode(Led,OUTPUT);
    pinMode(buttonpin,INPUT);
}
void loop()
{
    //Auslesen des Ausgangs des Hindernisvermeidungsmoduls
    val=digitalRead(buttonpin);
    //Eingebaute LED leuchtet, wenn der Ausgang des Moduls auf high liegt
    if(val==HIGH)
    {
        digitalWrite(Led,HIGH);
    }
    else
    {
        digitalWrite(Led,LOW);
    }
}
```

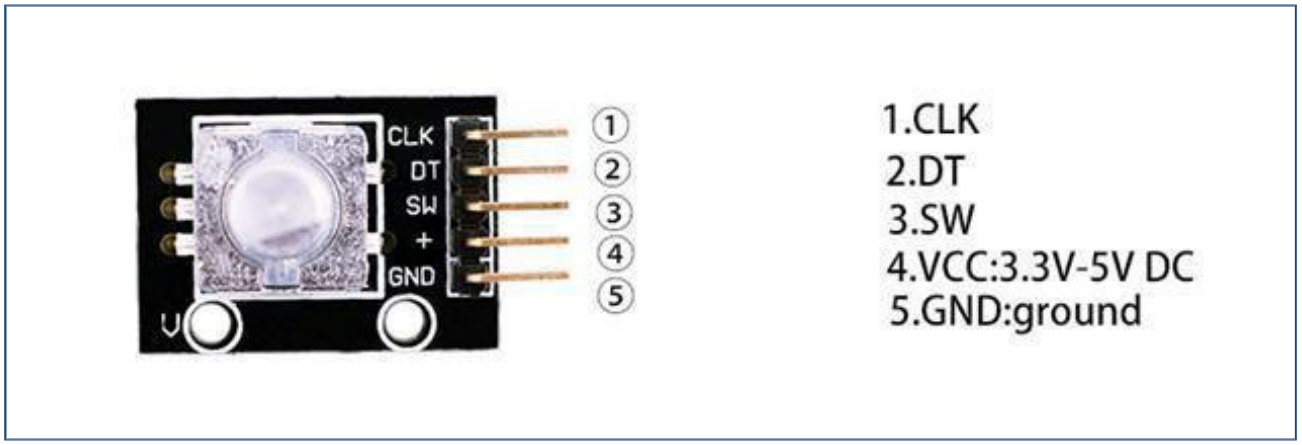
Lektion 26 DREHGEBERMODUL

Überblick

In diesem Versuch lernen wir, wie man ein Drehgebermodul benutzt.

Drehgeber

Ein Drehgeber ist ein Sensor für Drehwinkel, der Ausgangssignale liefert, die am anderen Ende der Sensorleitung im Auswertegerät decodiert werden müssen. Deshalb wird er auch als Encoder bezeichnet.



Benötigte Komponenten:

1 x Elegoo Uno R3

1 x USB Kabel

1 x Drehgebermodul

5 x F-M Drähte

Komponenteneinführung

Datenblattauszug (nur englisch):

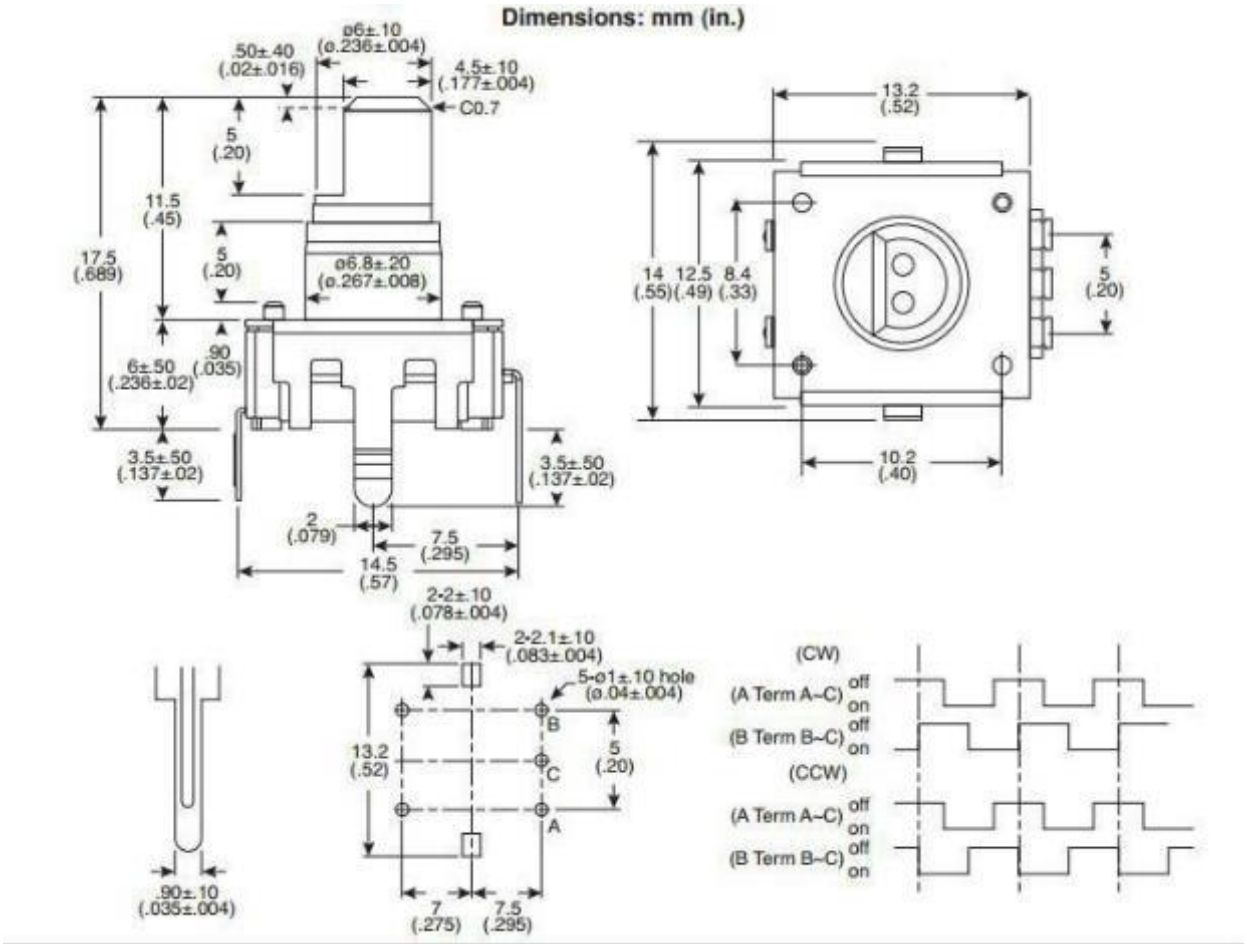
Mechanical Specifications:
- Operating Temp: -10°C to 70°C
- Storage Temp: -40°C to 85°C
- Rotational Torque: 50gf.cm max.
- No. and Pos. of detents: 12 detents (Step angle 30°±3°)
- Terminal Strength: A static load for 300gf.cm shall be applied to the tip of the terminals for 10 sec. in any direction
- Shaft push-pull strength: 5.1kgf
- Rotational life: 30,000 cycles

Note:
- RoHS Compliant

Electrical Specifications:
- Rating: 1mA/10VDC
- Insulation Resistance: 50VDC 10MΩ Min.
- Dielectric Strength: 50VAC for 1 min.
- Resolution: 12 pulses/360° for each phase

Soldering Specifications:
- Soldering: To be performed in 5 seconds within 260±5°C
- Manual Soldering: To be performed in 3 seconds within 350±5°C
- Preheating: The entire flow duration should not exceed 2 min., and soldering surface temperature (undersurface of PCB) shall be settled within 100°C

Push-on Switch Specifications:
- Type: Single Pole Single Throw (Push on)
- Rating: 10mA/5VDC
- Switch Travel (mm): 0.5±0.4
- Operating Force: 200~460gf
- Operating Life: 20,000 times



Prinzip

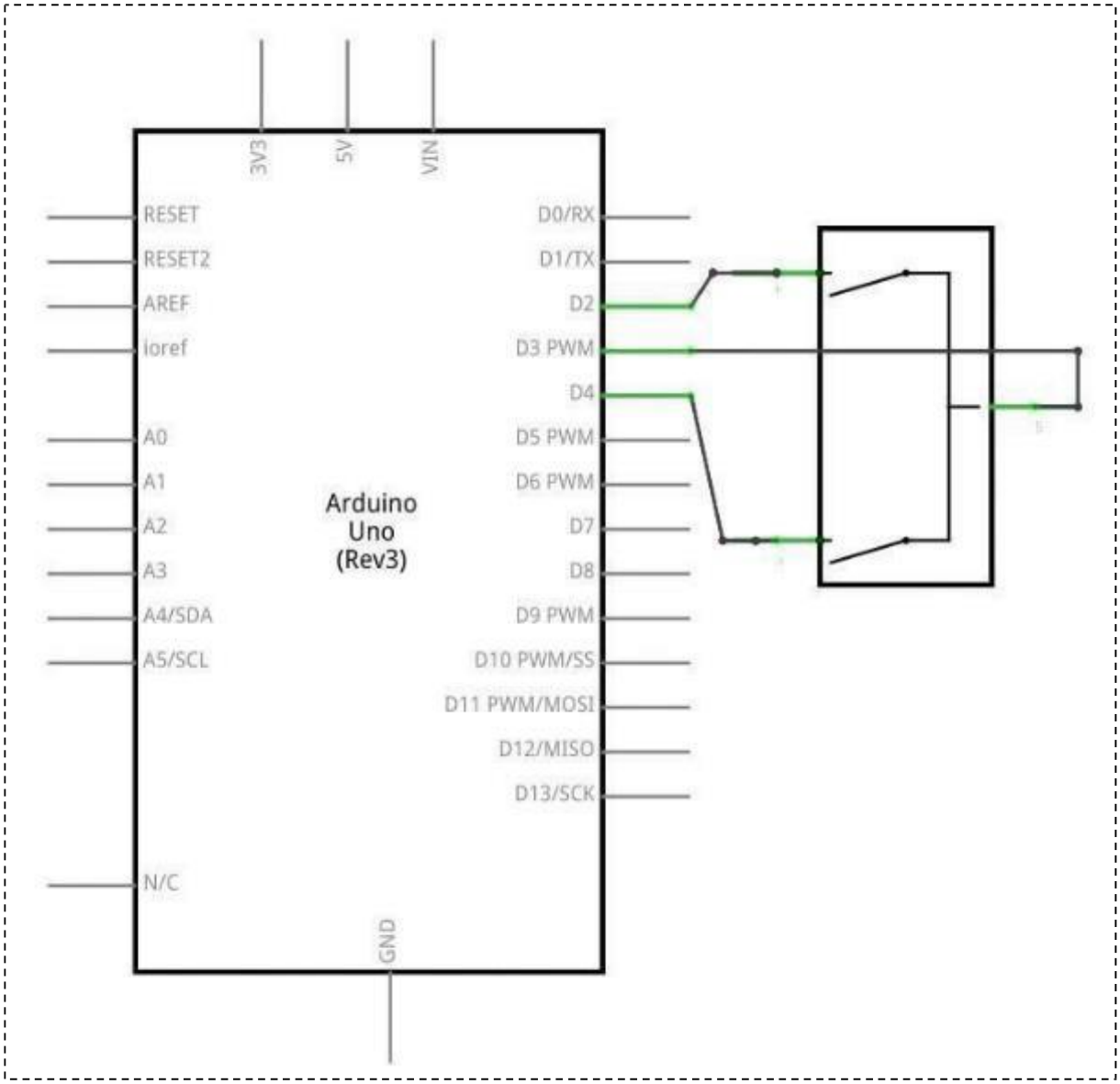
Inkrementalgeber

Der Ausgang von Inkrementalgebern liefert Informationen über die Bewegung, die typischerweise an anderer Stelle zu Informationen wie Geschwindigkeit, Weg und Position weiterverarbeitet werden. Ein Inkrementalgeber ist ein Drehgeber, der die Drehbewegung in eine Reihe von digitalen Impulssignalen umwandelt, die dann zur Steuerung der Winkelbewegung verwendet werden. Er erzeugt zweiphasige Rechtecksignale, deren Phasendifferenz 90° beträgt. Normalerweise werden die zweiphasigen Rechtecksignale als Kanal A (bzw. bei unserem Modul CLK genannt) und Kanal B (bzw. bei unserem Modul DT genannt) bezeichnet.

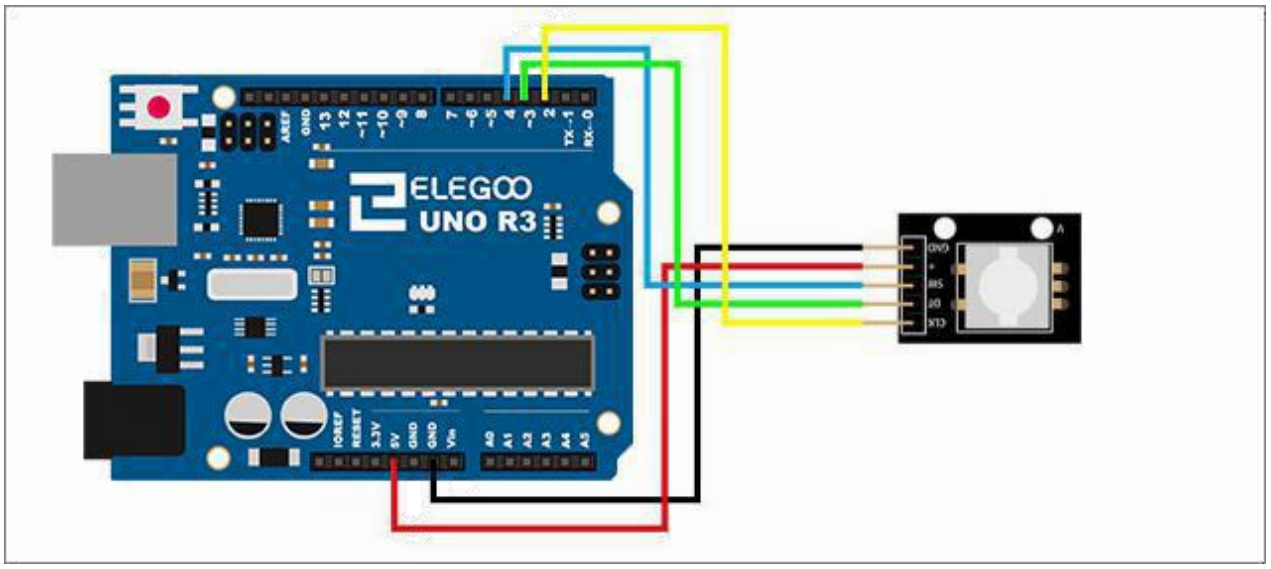
Dem obigen Datenblatt kann entnommen werden, dass wenn sowohl Kanal A als auch Kanal B ‚high' sind, wird der Schalter im Uhrzeigersinn gedreht; wenn Kanal A ‚high' und Kanal B ‚low' ist, wird der Schalter gegen den Uhrzeigersinn gedreht. Wenn also Kanal A ‚high' ist, können Sie beurteilen, ob sich der Drehgeber nach links oder rechts dreht, solange Sie den Zustand von Kanal B kennen.

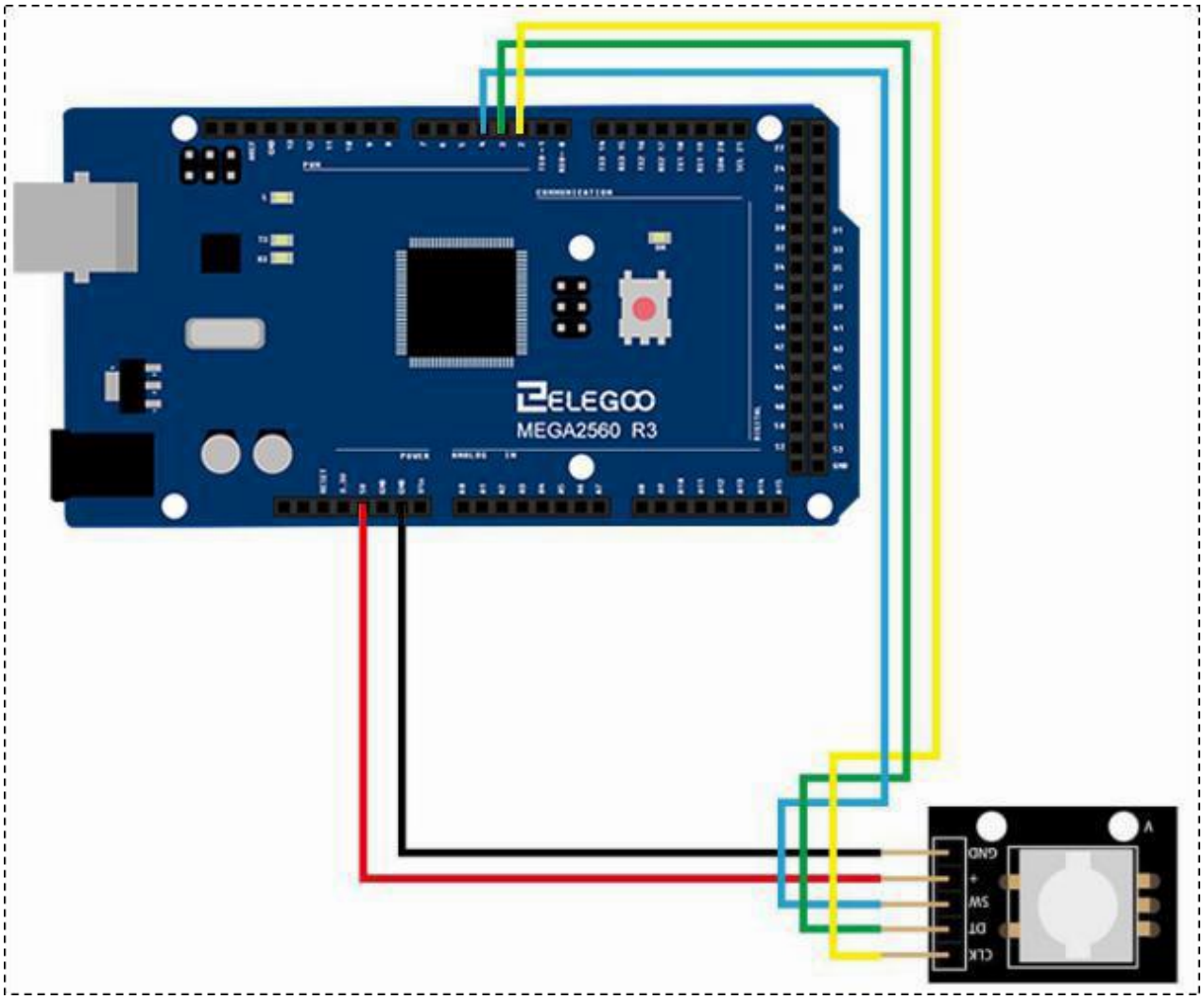
Verbindung

Schaltplan



Verdrahtungsplan

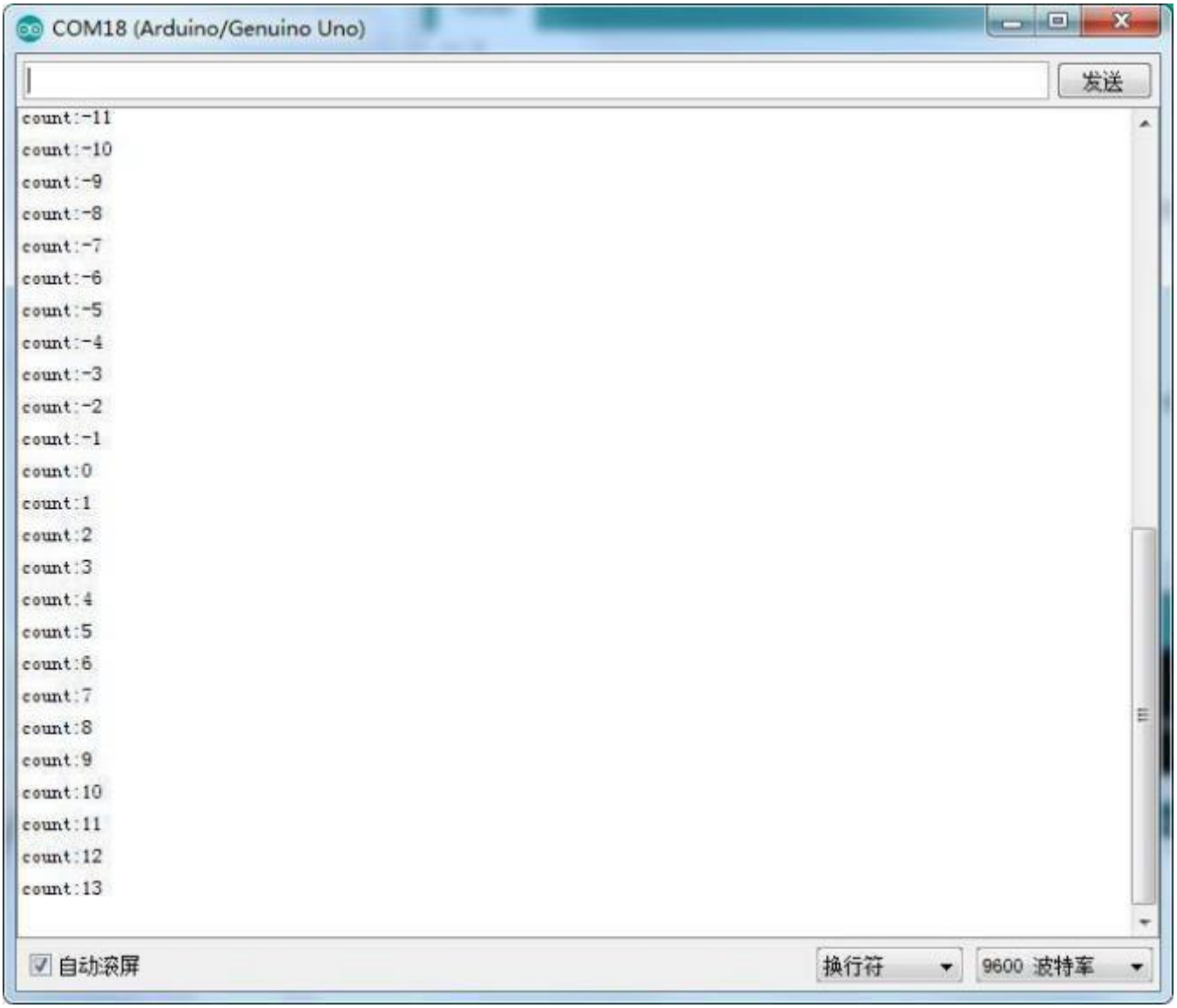




Code

Nach der Verdrahtung laden Sie bitte das Programm ‚Lesson 26 ROTARY ENCODERS MODULE' auf Ihren Arduino. Öffnen Sie danach den seriellen Monitor und drehen Sie am Drehgeber. Die Ausgabe sollte in etwa wie auf dem folgenden Screenshot aussehen. Beim Drehen im Uhrzeigersinn steigt der angezeigte Wert, beim Drehen entgegen dem Uhrzeigersinn sinkt er.

Beispielbild


COM18 (Arduino/Genuino Uno)
发送
count:-11
count:-10
count:-9
count:-8
count:-7
count:-6
count:-5
count:-4
count:-3
count:-2
count:-1
count:0
count:1
count:2
count:3
count:4
count:5
count:6
count:7
count:8
count:9
count:10
count:11
count:12
count:13
自动滚屏
换行符
9600 波特率

Im Folgenden finden Sie den Code und einige Erklärungen.

```
int CLK = 2;//CLK->Arduiono D2
int DT = 3;//DT-> Arduiono D3
int SW = 4;//SW-> Arduiono D4
// Interrupt 0 an pin 2
const int interrupt0 = 0;
int count = 0;
int lastCLK = 0;
void setup()
{
   pinMode(SW, INPUT);
   digitalWrite(SW, HIGH);
   pinMode(CLK, INPUT);
   pinMode(DT, INPUT);
   //Interrupt handler aktivieren und auf „Changed-Events“ anmelden
   // → Methode ClockChanged wird aufgerufen, wann auch immer sich der Wert am Interrupt Pin
//ändert
   attachInterrupt(interrupt0, ClockChanged, CHANGE);

   Serial.begin(9600);
}

void loop()
{//Beim Drücken des Schalters/Stiftes den Zähler auf 0 setzen
  if (!digitalRead(SW) && count != 0)
  {
      count = 0;
      Serial.print("count:");
      Serial.println(count);
    }
}
```

```
//Interrupt handler
void ClockChanged()
{
  //Wert des CLK pin lesen
  int clkValue = digitalRead(CLK);
  //Wert des DT pin lesen
  int dtValue = digitalRead(DT);
  if (lastCLK != clkValue) //Falls sich der Wert des CLK Pins geändert hat …
  {
    lastCLK = clkValue;
    //Falls CLK und DT unterschiedliche Werte haben, wurde im Uhrzeigersinn gedreht, ansonsten
    //gegen den Uhrzeigersinn
    count += (clkValue != dtValue ? 1 : -1);
    Serial.print("count:");
    Serial.println(count);
  }
}
```

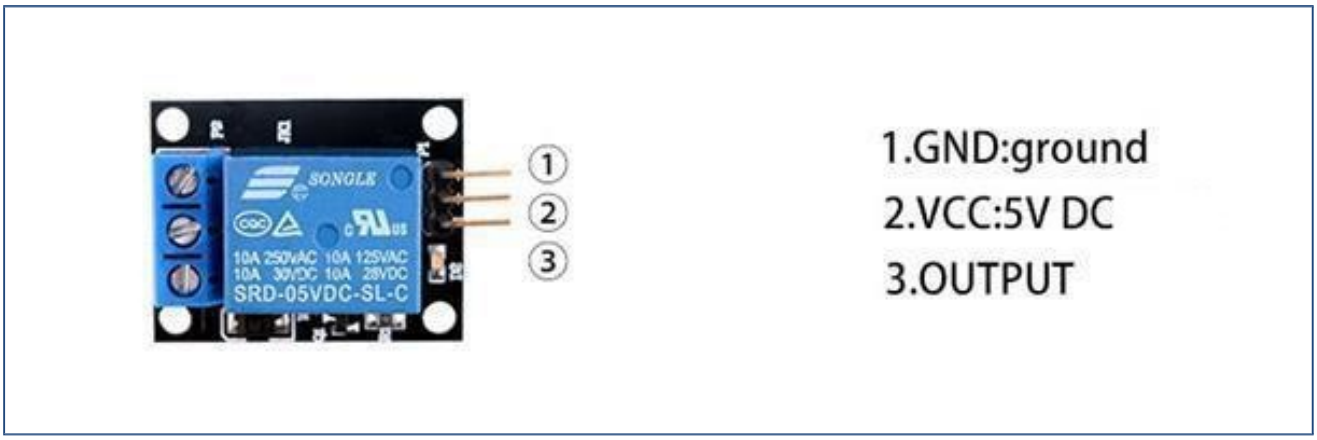
Lektion 27 EINKANALRELAISMODUL

Überblick

Ein Relais ist ein durch elektrischen Strom betriebener, fernbetätigter Schalter mit in der Regel zwei Schaltstellungen. Das Relais wird über einen Steuerstromkreis aktiviert und kann weitere Stromkreise schalten. Unser Unternehmen produziert Relaismodule von 28V bis 240V, sowohl für Wechselstrom als auch für Gleichstrom. Sie können in Kombination mit einem Mikrocontroller verwendet werden, um einen Zeitschalter zu erstellen. Relais werden in vielen Bereichen eingesetzt, von Alarmanlagen über Spielzeug bis hin zu Baumaschinen.

Relais

Ein Relaismodul für den direkten Anschluss an ein Arduino-Board. Das Modul benötigt eine Spannungsversorgung von 5 V. Das Eingangssteuersignal wird mit einem 'S' gekennzeichnet (Im Bild unten fälschlicherweise als „Output“ bezeichnet). Das Relais hat einen Wechselkontakt. Er kann ohmsche Lasten bis zu 10 A bei 250 VAC und bis zu 10 A bei bei maximal 30 VDC schalten.



Benötigte Komponenten:

1x Elegoo Uno R3

1x USB Kabel

1x 1-Kanal-Relais-Modul

3x F-M Drähte

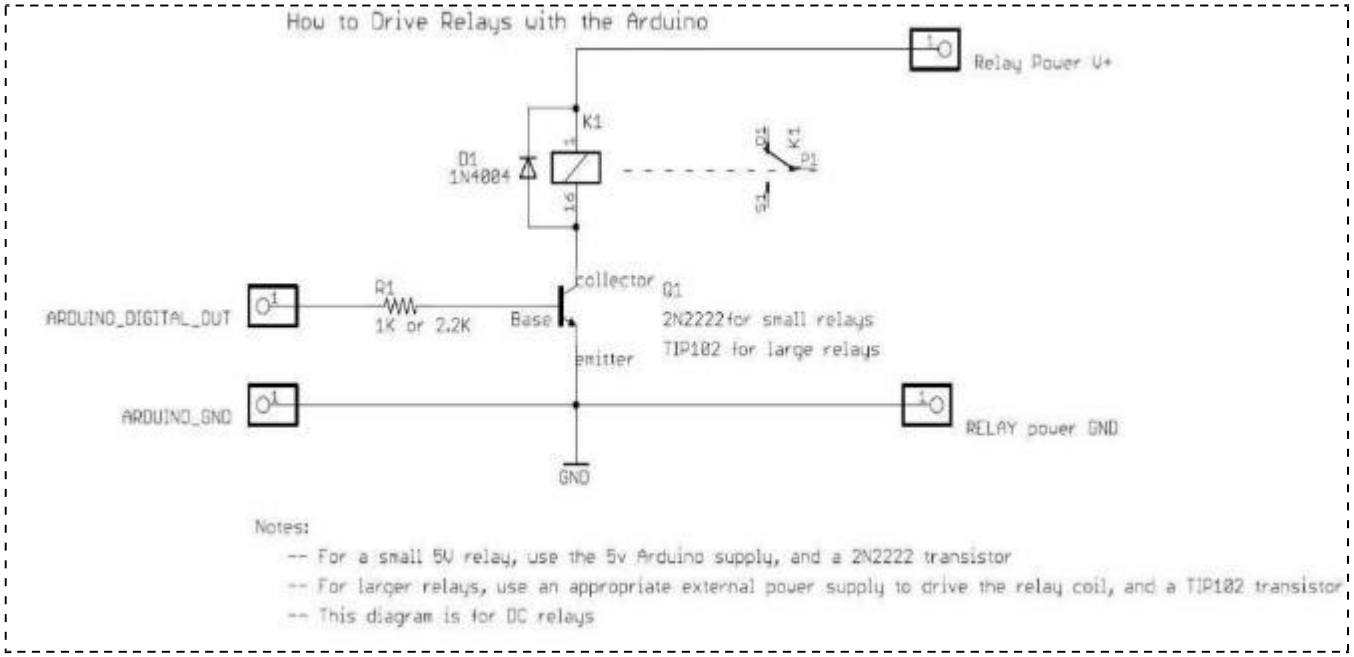
Komponenteneinführung

Relais:

Ein Relais ist ein elektrisch betätigter Schalter. Viele Relais verwenden einen Elektromagneten zur mechanischen Betätigung eines Schalters, aber auch andere Funktionsprinzipien, wie z.B. Solid-State-Relais, werden verwendet. Relais werden dort eingesetzt wo es notwendig ist einen Stromkreis durch ein schwaches Signal (z.B. 5V) zu steuern (mit vollständiger galvanischer Trennung zwischen Steuerkreis und Lastkreis), oder wo mehrere Stromkreise durch ein Signal gesteuert werden müssen. Die ersten Relais wurden in Langstrecken-Telegraphenschaltungen als Verstärker eingesetzt: Sie wiederholten das von einer Schaltung kommende Signal und übertrugen es auf eine andere Schaltung. Relais wurden ausgiebig in Telefonzentralen und frühen Computern eingesetzt, um logische Operationen durchzuführen.

Ein Relaistyp, der die hohe Leistung bewältigen kann, die zur direkten Ansteuerung eines Elektromotors oder anderer Verbraucher erforderlich ist, wird als Schütz bezeichnet. Solid-State-Relais steuern Stromkreise ohne bewegliche Teile und verwenden stattdessen ein Halbleiterbauelement zum Schalten. Relais mit kalibrierten Betriebseigenschaften und teilweise mehreren Betriebsspulen werden eingesetzt, um elektrische Schaltungen vor Überlastung oder Störungen zu schützen; in modernen Stromversorgungssystemen werden diese Funktionen von digitalen Geräten übernommen, die immer noch "Schutzrelais" genannt werden.

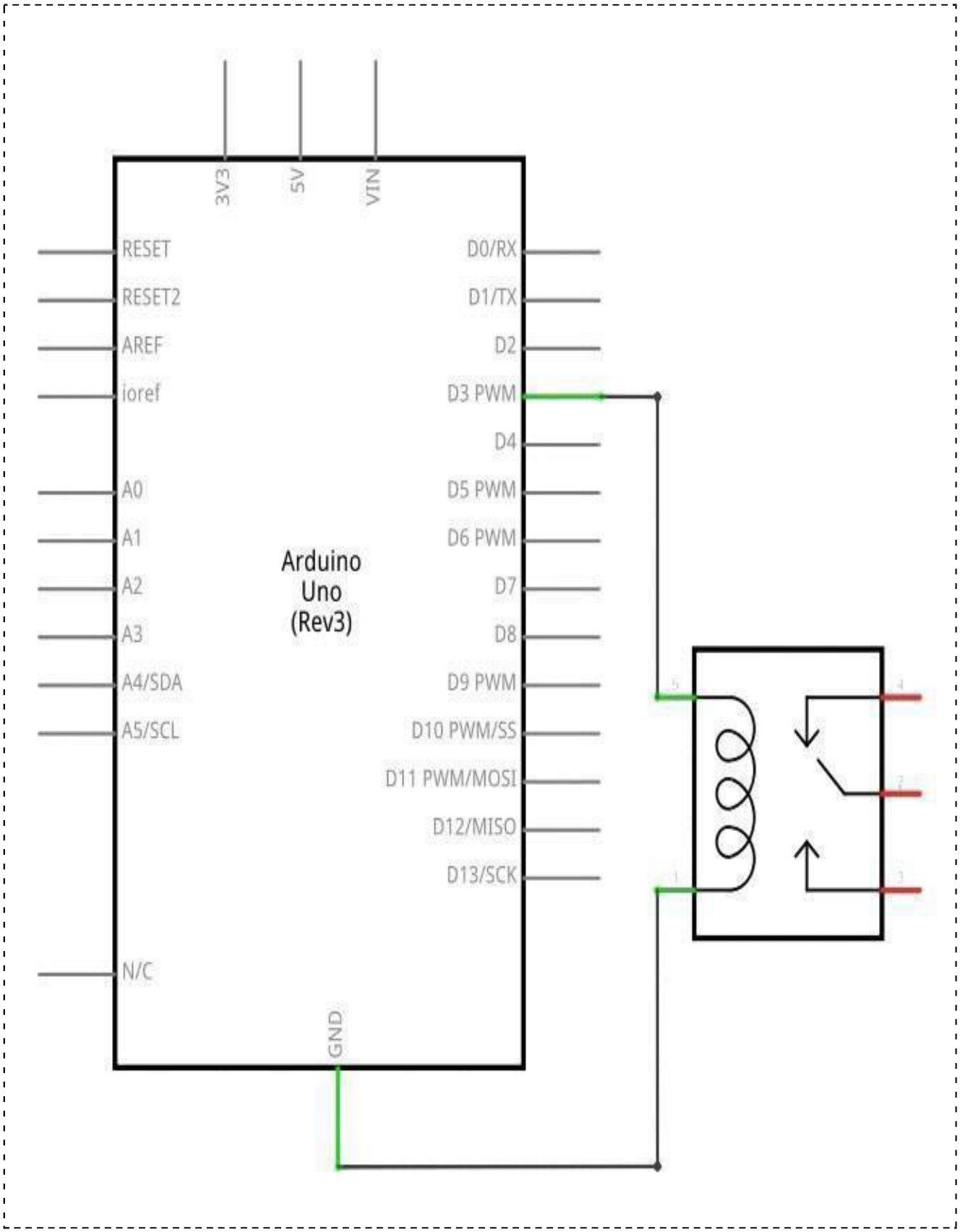
Nachfolgend ist die schematische Darstellung der Ansteuerung von Relais mit Arduino (von arduino.cc)



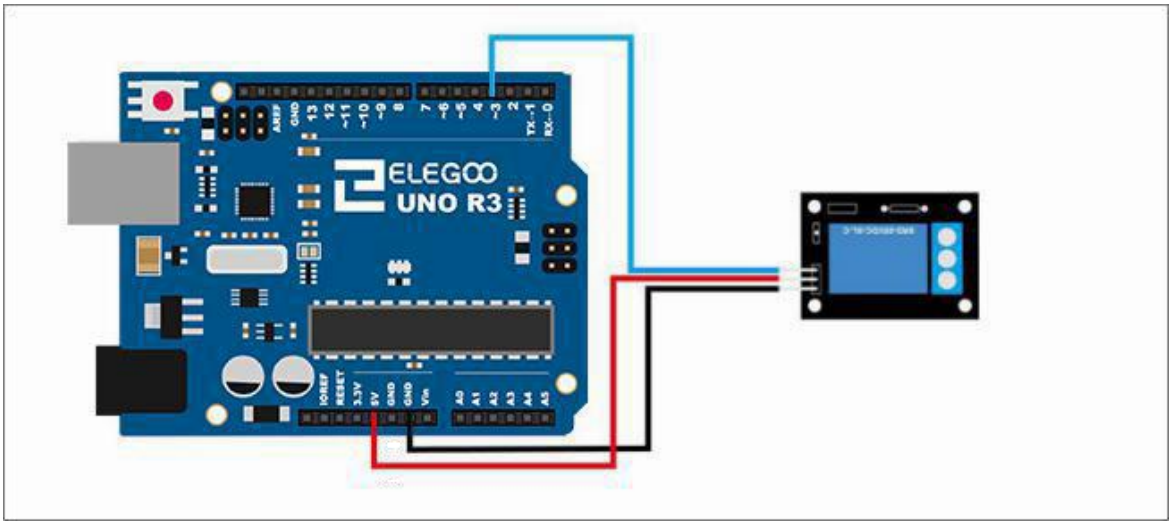
Wir sehen also, dass ein Relais normalerweise nicht direkt von einem Arduino Port aus angesteuert werden kann, weil die Arduino Ports nicht genug Strom liefern können für das Relais. Unser Relaismodul hat allerdings Transistor und Widerstände schon an Board. Im Gegensatz zu einem nackten Relais können wir das Relaismodul direkt von einem Arduino Port aus ansteuern.

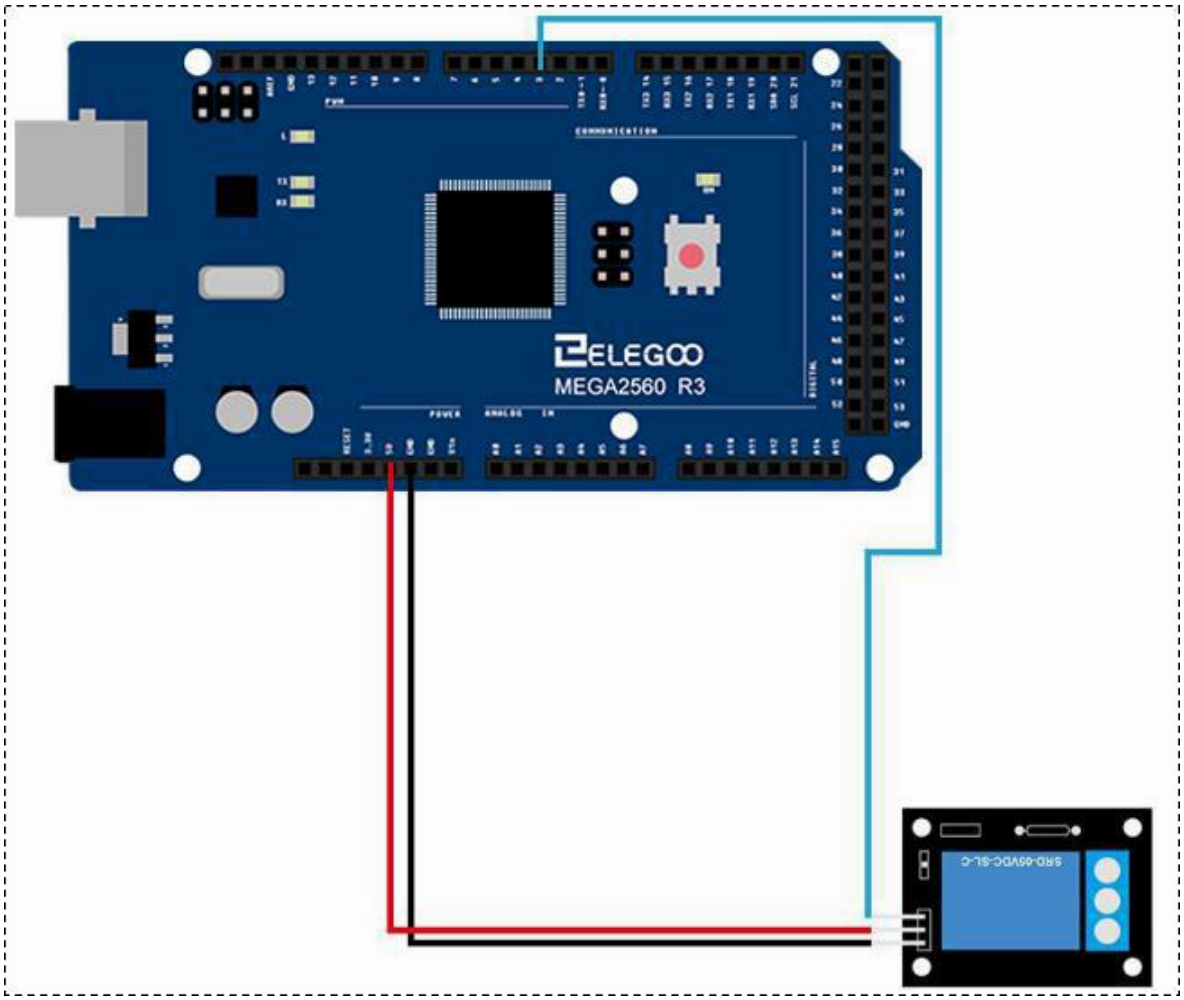
Unser Relaismodul hat auf der Lastseite drei Klemmen mit Schlitzschrauben. Wie im nachfolgenden Bild zu sehen ist, handelt es sich bei dem Relais um einen Wechsler. D.h. der mittlere Anschluss ist je nachdem ob das Relais angezogen ist mit dem oberen oder mit dem unteren Anschluss verbunden.

Verbindung
Schaltplan



Verdrahtungsplan

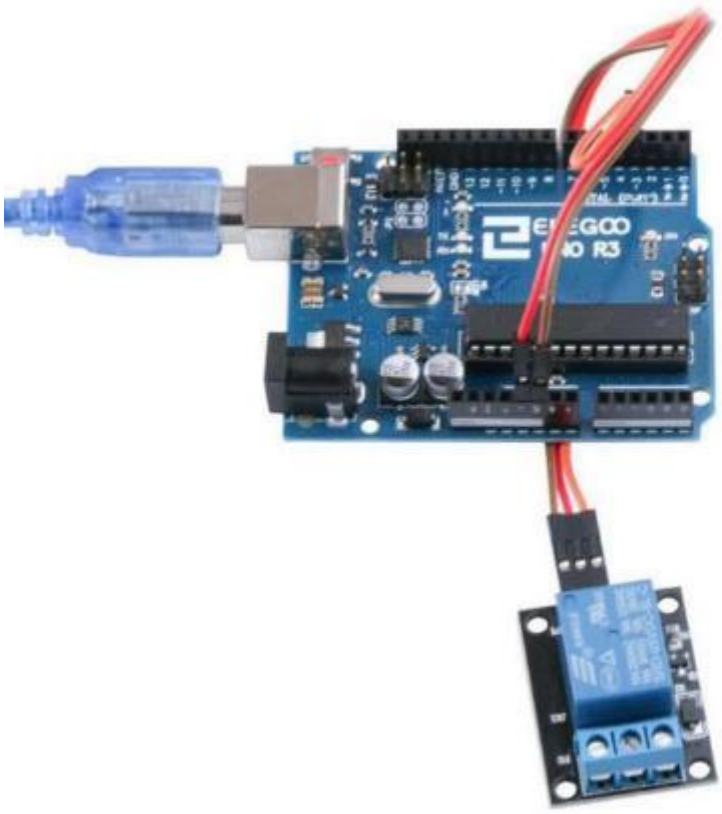




Code

Nachdem die Schaltung verbunden ist, laden Sie bitte das Programm "Lesson 27 1 CHANNEL RELAY MODULE" auf den Arduino. Wir hören das Klacken des Relais, wenn das Relais anzieht bzw. abfällt. Wenn Sie wollen, können sie auch die Lastseite (die Klemmen mit den Schlitzschrauben) beschalten. Zum Beispiel könnten sie eine LED mit Vorwiderstand auf der einen Seite mit 5V verbinden und auf der anderen Seite mit dem oberen Kontakt des Relaisausgangs. Der mittlere Ausgang des Relaismoduls könnte dann mit Masse verbunden werden. In dieser Beschaltung würde die LED bei unserem Code immer 2s leuchten und wieder 2s aus sein.

Beispielbild



Im Folgenden finden Sie den Code und einige Erklärungen

```
int relayPin = 3;
void setup()
{
    pinMode(relayPin, OUTPUT);
}

void loop()
{
     // Relais zieht an, d.h. der Ausgang des Relais ist durchgeschalten
     digitalWrite(relayPin, HIGH);
     // 2s warten
    delay(2000);
    // Relais fällt, d.h. der Ausgang des Relais ist getrennt
    digitalWrite(relayPin, LOW);
    // 2s warten
    delay(2000);
}
```

Lektion 28 LCD-Display

Übersicht

In dieser Lektion lernen Sie, wie man ein alphanumerisches LCD-Display verdrahtet und verwendet. Das Display hat eine LED-Hintergrundbeleuchtung und kann zwei Zeilen mit bis zu 16 Zeichen pro Zeile anzeigen. Sie sehen die Rechtecke für jedes Zeichen auf dem Display und die Pixel, aus denen sich jedes Zeichen zusammensetzt. Das Display zeigt nur weiß auf blau und dient zur Anzeige von Text.

In dieser Lektion führen wir nur ein Arduino-Beispielprogramm aus der LCD-Bibliothek aus, aber in der nächsten Lektion wird unser Display zur Anzeige der Temperatur benutzt.

Benötigte Komponenten:

1x Elegoo Uno R3

1x LCD1602 Modul

1 x Potentiometer(10k)

1 x 830 Breadboard

16 x M-M Drähte

Komponenteneinführung

LCD1602

Pinbelegung:

VSS: Masse (Ground)

VDD: 5V Spannungsversorgung

VO: Kontrasteinstellung

RS: Ein Registerauswahlpin, der steuert, wohin im Speicher der LCD-Anzeige Daten geschrieben werden sollen. Sie können entweder das Datenregister wählen, in dem sich das, was auf dem Bildschirm erscheint, befindet, oder ein Befehlsregister, in dem der LCD-Controller nach Anweisungen sucht, was als nächstes zu tun ist.

R/W: Read/Write (Lesen/Schreiben). Wählt aus ob gelesen oder geschrieben wird

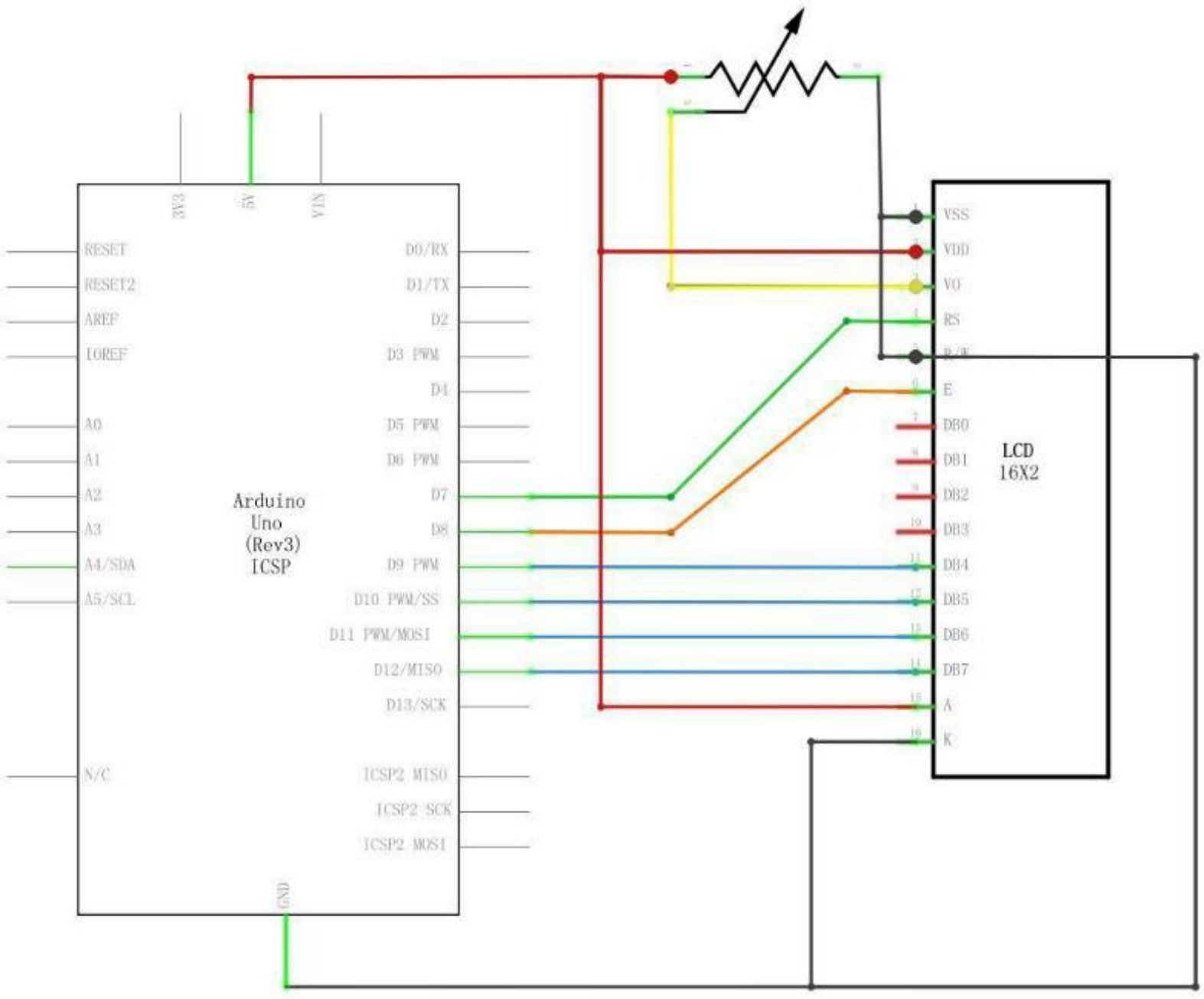
E: Der Pin startet das Lesen/Schreiben des Displays

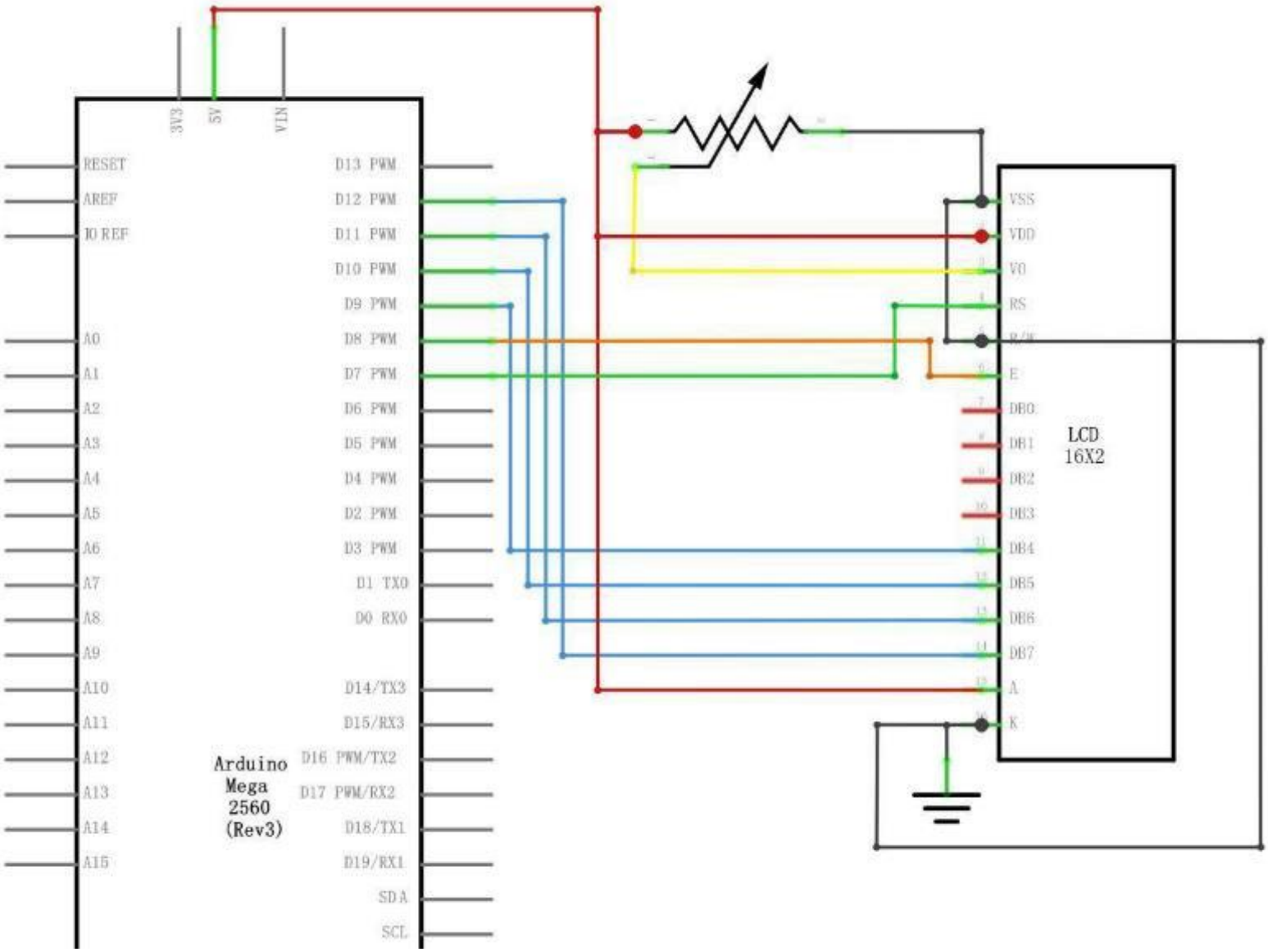
D0-D7： Datenpins

A and K: Stromversorgung für die Hintergrundbeleuchtung

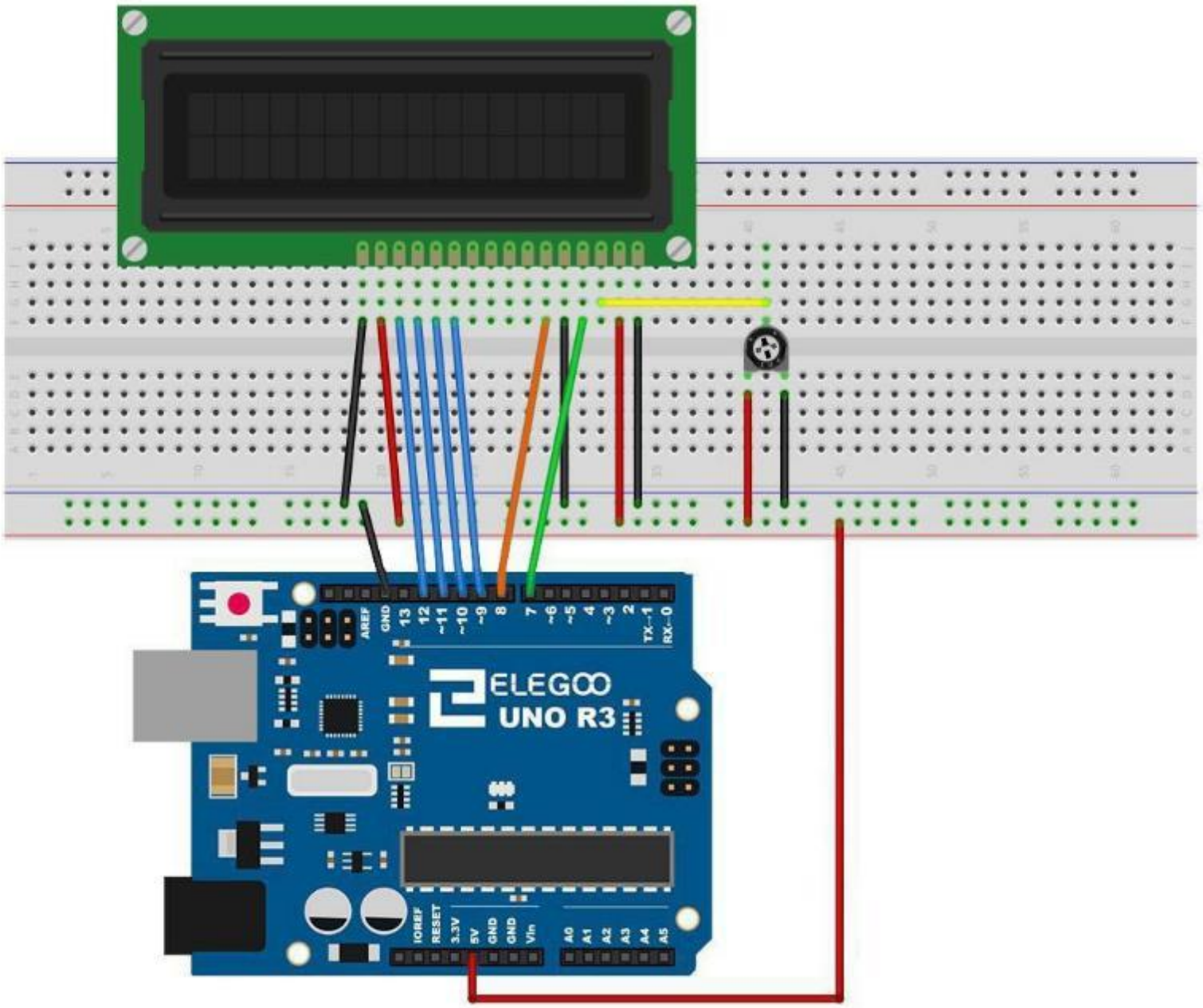
Verbindung

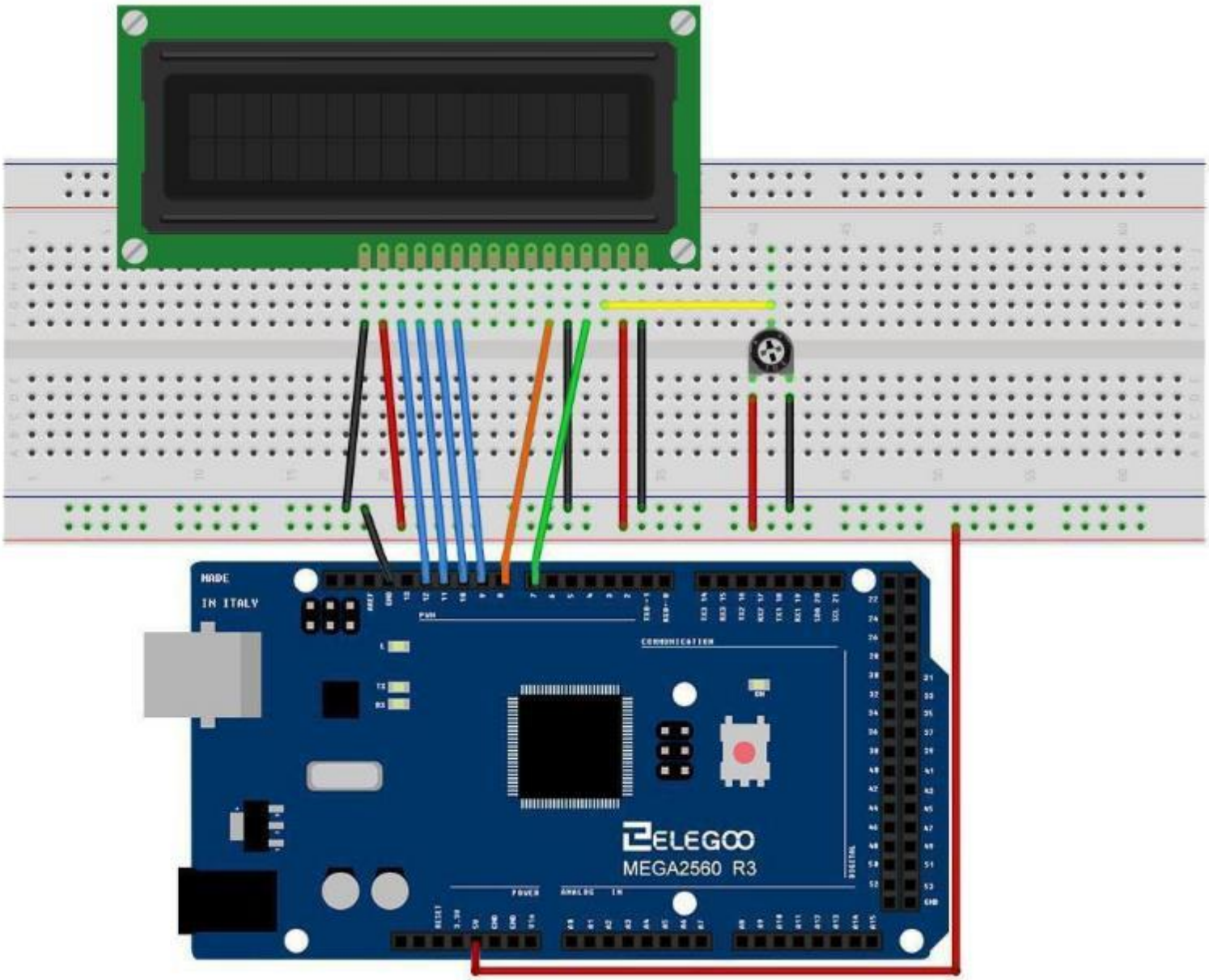
Schaltplan




3V3
5V
VIN
RESET
AREF
IO REF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
Arduino
Mega
2560
(Rev3)
VSS
VDD
V0
RS
E
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
A
K
LCD
16X2

Verdrahtungsplan




MADE
IN ITALY
ELEGOO
MEGA2560 R3

Das LCD-Display benötigt sechs Arduino-Pins, die alle digitale Ausgänge sind. Außerdem benötigt er 5V und GND Anschlüsse.

Es gibt eine Reihe von Verbindungen, die hergestellt werden müssen. Die Ausrichtung des Displays mit der Oberseite des Breadboards hilft, seine Pins ohne zu viel Zählen zu identifizieren, besonders wenn die Zeilen des Breadboards nummeriert sind. Vergessen Sie nicht das lange gelbe Kabel, das den Schleifer des Potentiometers mit Pin 3 des Displays verbindet. Das Potentiometer dient zur Steuerung des Kontrastes des Displays. Ohne passenden Kontrast ist auf dem Display nichts erkennbar.

Möglicherweise wird Ihr Display ohne Stiftleisten geliefert. Falls dem so ist, folgen Sie den Anweisungen im nächsten Abschnitt.

Code

Nach der Verdrahtung öffnen Sie bitte das Programm im Code-Ordner - Lesson 28 LCD Display und klicken Sie auf „Hochladen“, um das Programm hochzuladen. Falls Fehler auftreten, gehen Sie bitte zu Lektion 2 für Details über das Hochladen von Programmen.

Stellen Sie sicher, dass Sie die < LiquidCrystal > Bibliothek installiert haben und installieren Sie sie, falls sie noch nicht installiert ist. Andernfalls wird Ihr Code nicht funktionieren (es werden Kompilierfehler auftreten).

Einzelheiten zum Installieren von Bibliotheken finden Sie in Lektion 1.

Laden Sie den Code auf Ihr Arduino-Board und Sie sollten die Meldung 'Hello World' sehen, gefolgt von einer Zahl, die von Null aufwärts zählt.

Die erste Zeile im Sketch ist:

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

Diese Zeile bindet die LiquidCrystal Bibliothek ein

Die nächste Zeile definiert welche Arduino Pins mit dem Display verbunden werden.

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

Nach dem Hochladen des Codes, drehen Sie das Potentiometer, so dass sie den Text auf dem Display gut lesen können.

In der Setup Funktion gibt es zwei Befehle:

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

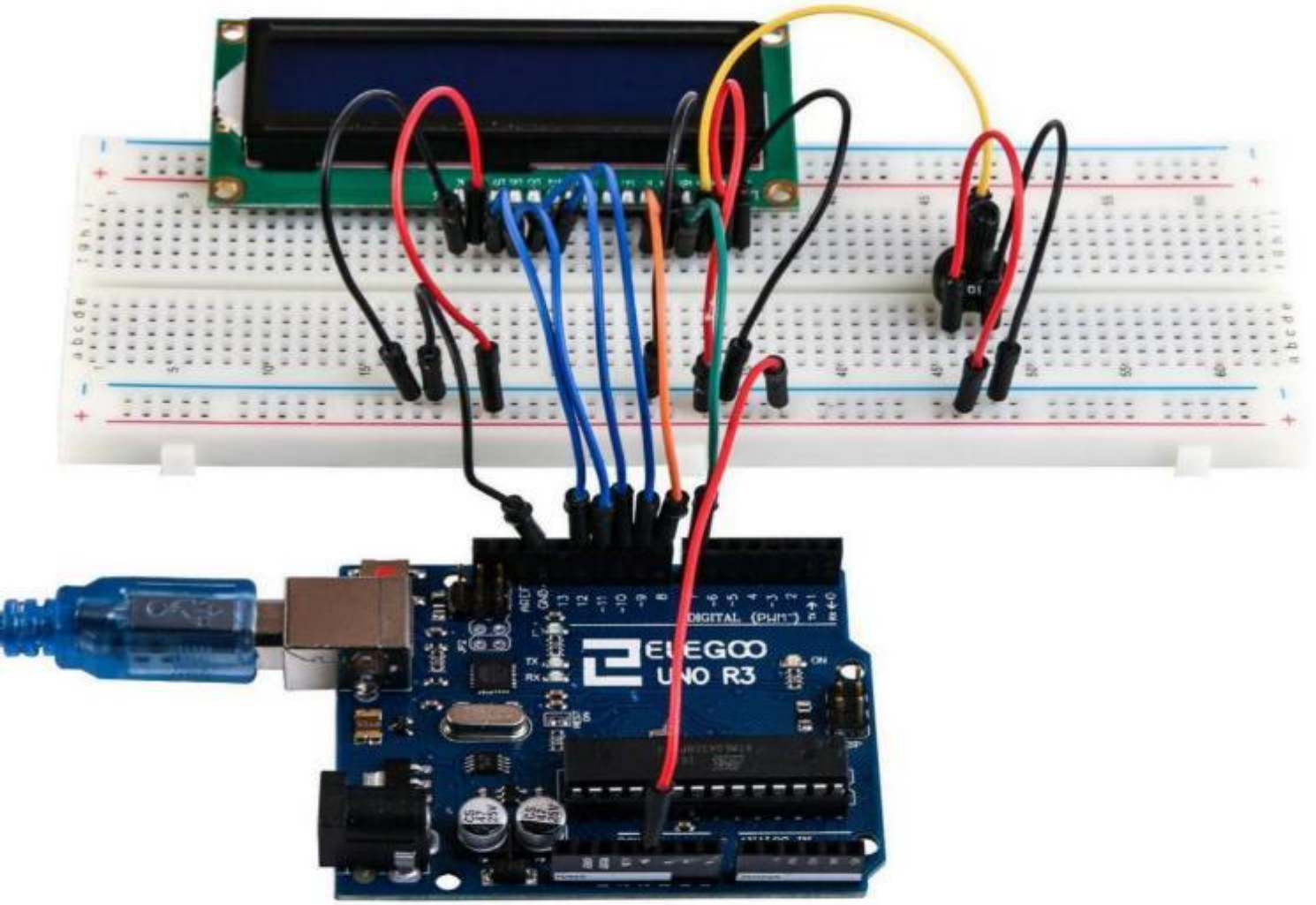
```
lcd.print("Hello, World!");
```

Die erste sagt der Liquid Crystal Bibliothek, wie viele Spalten und Zeilen die Anzeige hat. Die zweite Zeile zeigt die Meldung, die wir in der ersten Zeile des Bildschirms sehen.

In der Loop Funktion haben wir ebenfalls zwei Befehle:

Der erste setzt den Cursor in Spalte 1 der zweiten Zeile (die Nummerierung beginnt bei 0). Die zweite Anweisung schreibt die Anzahl der Sekunden seit dem Arduino Neustart an die Cursorposition. Auf der nächsten Seite sehen Sie, dass die Zahl in der zweiten Zeile steht.

```
lcd.setCursor(0, 1);
lcd.print(millis()/1000);
```


ELEGOO
UNO R3


Hello, World!
1

Lektion 29 Ultraschallmodul

Überblick

Der Ultraschallsensor ist ideal für alle Arten von Projekten, bei denen Abstandsmessungen erforderlich sind, z.B. zur Vermeidung von Hindernissen bei Robotern.

Der HC-SR04 ist preiswert und einfach zu bedienen, da wir eine speziell für diese Sensoren entwickelte Bibliothek verwenden werden.

Benötigte Komponenten:

1 x Elegoo Uno R3

1 x Ultraschallmodul

4 x F-M Drähte

Komponenteneinführung

Ultraschallsensor:

Das Ultraschallmodul HC-SR04 kann zwischen 2cm-400cm berührungslos Abstände messen. Die Messgenauigkeit beträgt bis zu 3mm. Die Module beinhalten Ultraschallsender, Empfänger und eine Steuereinheit. Das Grundprinzip des Moduls ist folgendermaßen:

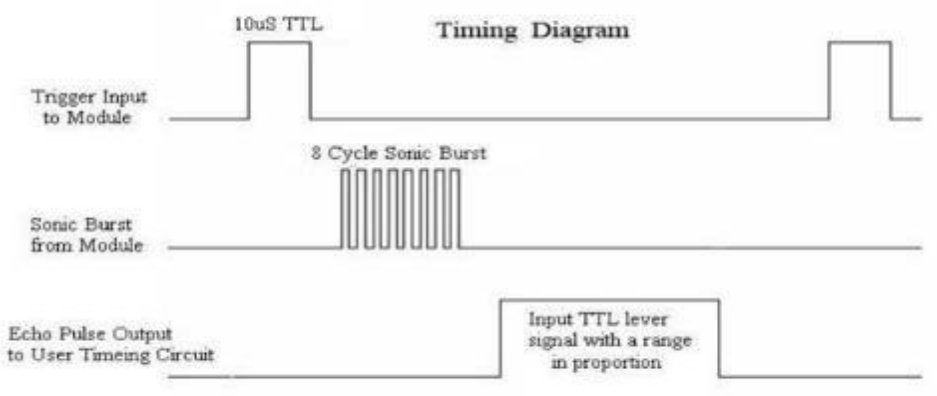
Durch einen Trigger Impuls (z.B. vom Arduino), der mindestens 10 Mikrosekunden lang sein muss, wird die Messung gestartet.

Das Modul sendet automatisch acht Signale mit 40kHz aus und empfängt die reflektierten Signale. Dabei misst es die Zeit, die vergeht, bis die Signale zurückkommen.

Abstand zum Hindernis = (Zeitdauer bis der Impuls zurückkommt × Schallgeschwindigkeit (340m/s) /2

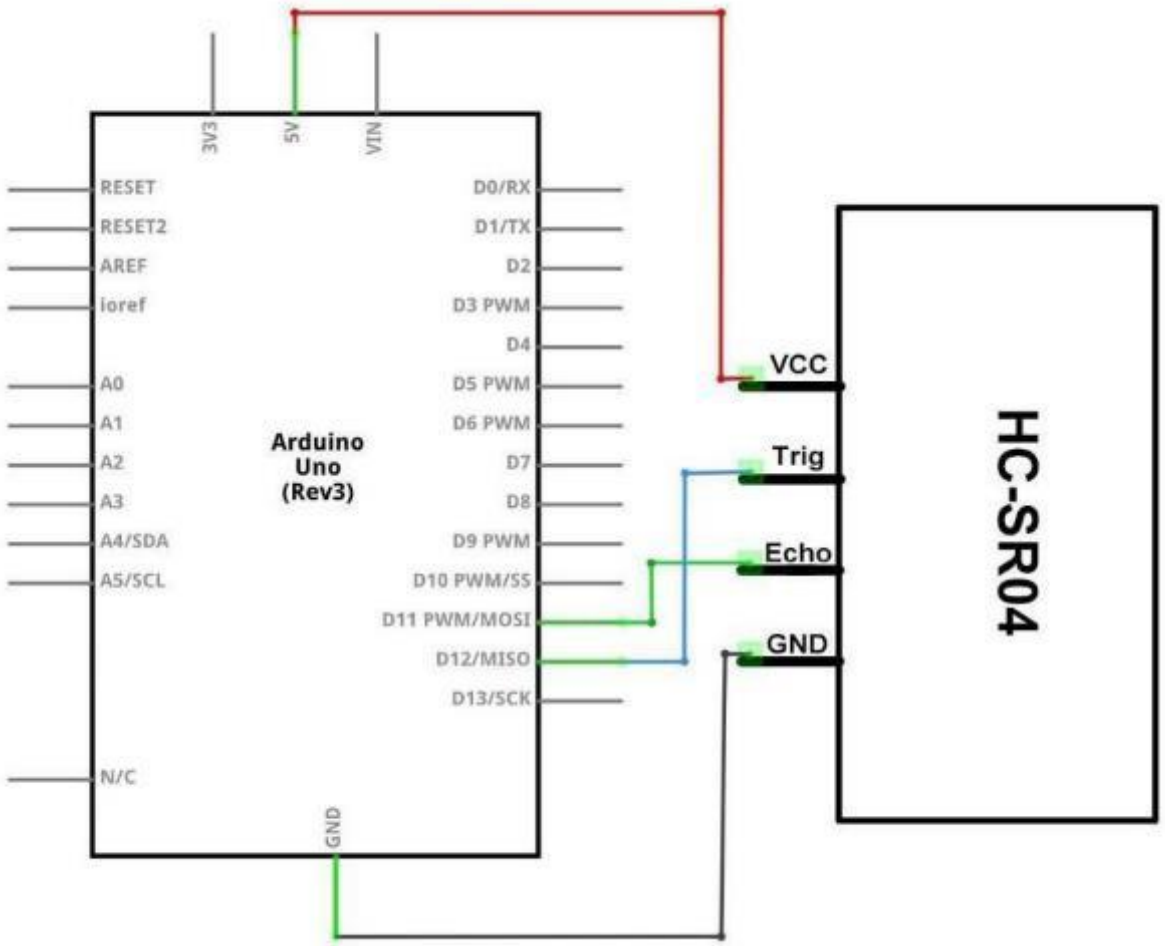
Das Timing-Diagramm ist unten dargestellt.

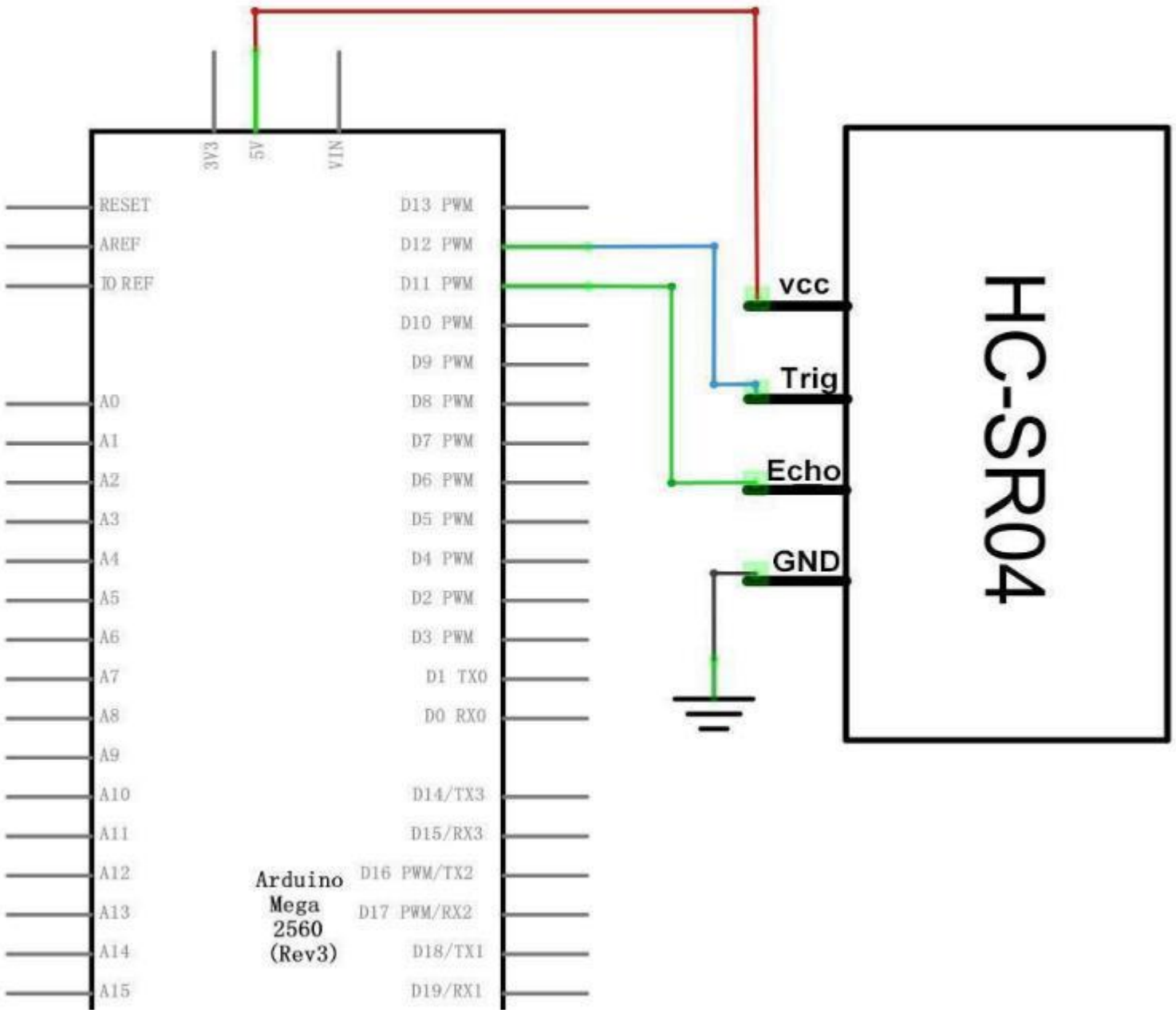
Am Echo-Pin wird ein Signal ausgegeben, das proportional zum Abstand zum Hindernis ist. Die Dauer dieses Signals werden wir am Arduino messen und so den Abstand zum Hindernis berechnen. Dabei hilft uns eine Bibliothek.



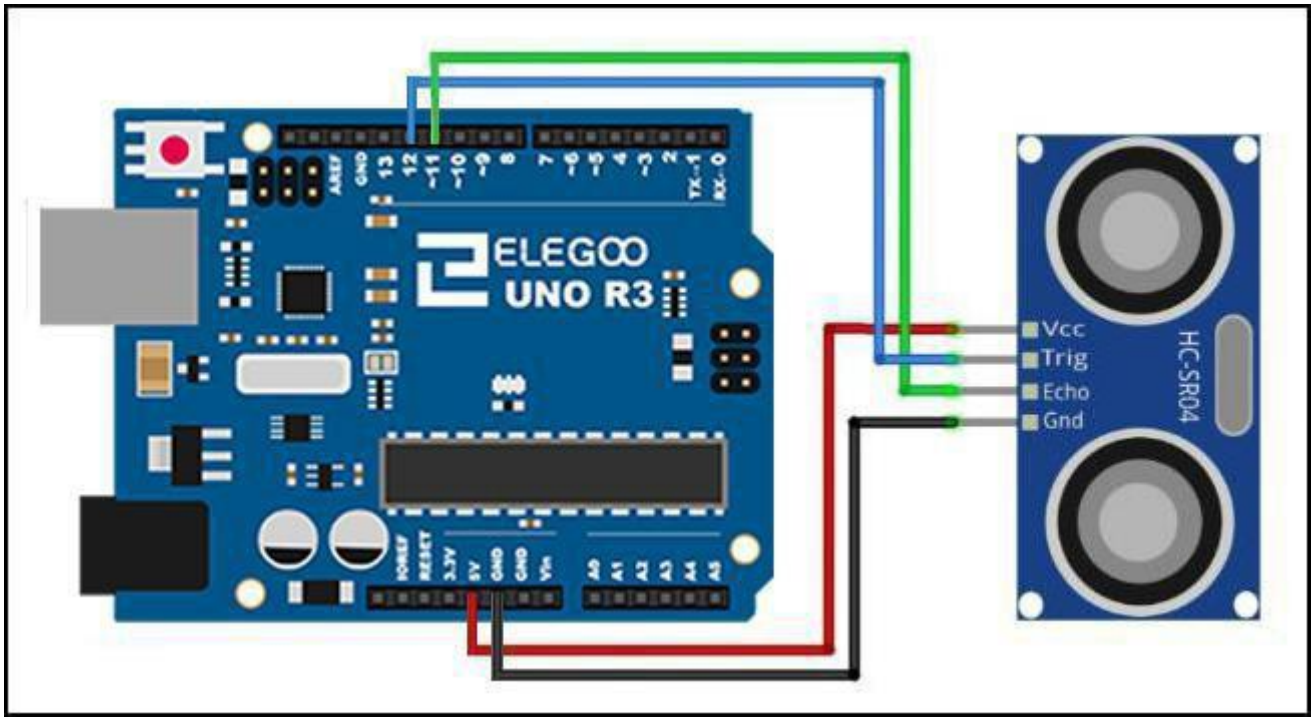
Verbindung

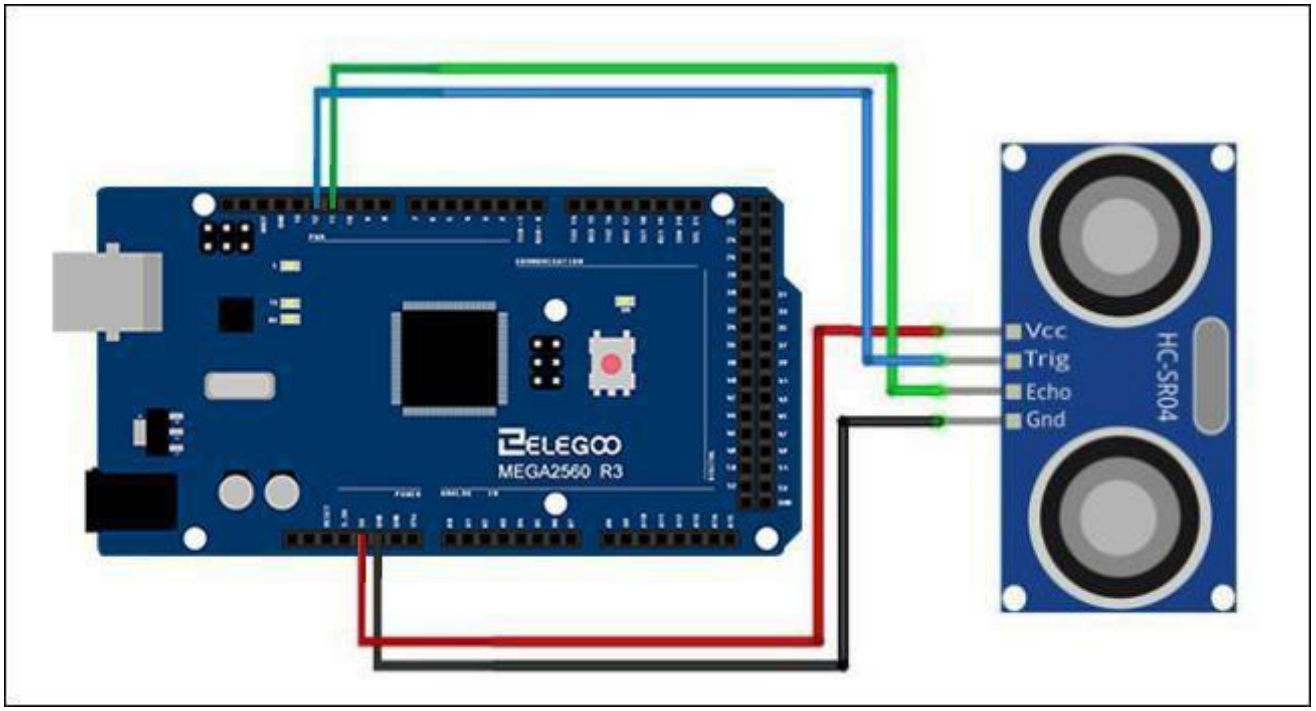
Schaltplan


3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
Arduino
Uno
(Rev3)
VCC
Trig
Echo
GND
HC-SR04


3V3
5V
VIN
RESET
AREF
IO REF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
Arduino
Mega
2560
(Rev3)
vcc
Trig
Echo
GND
HC-SR04

Verdrahtungsplan

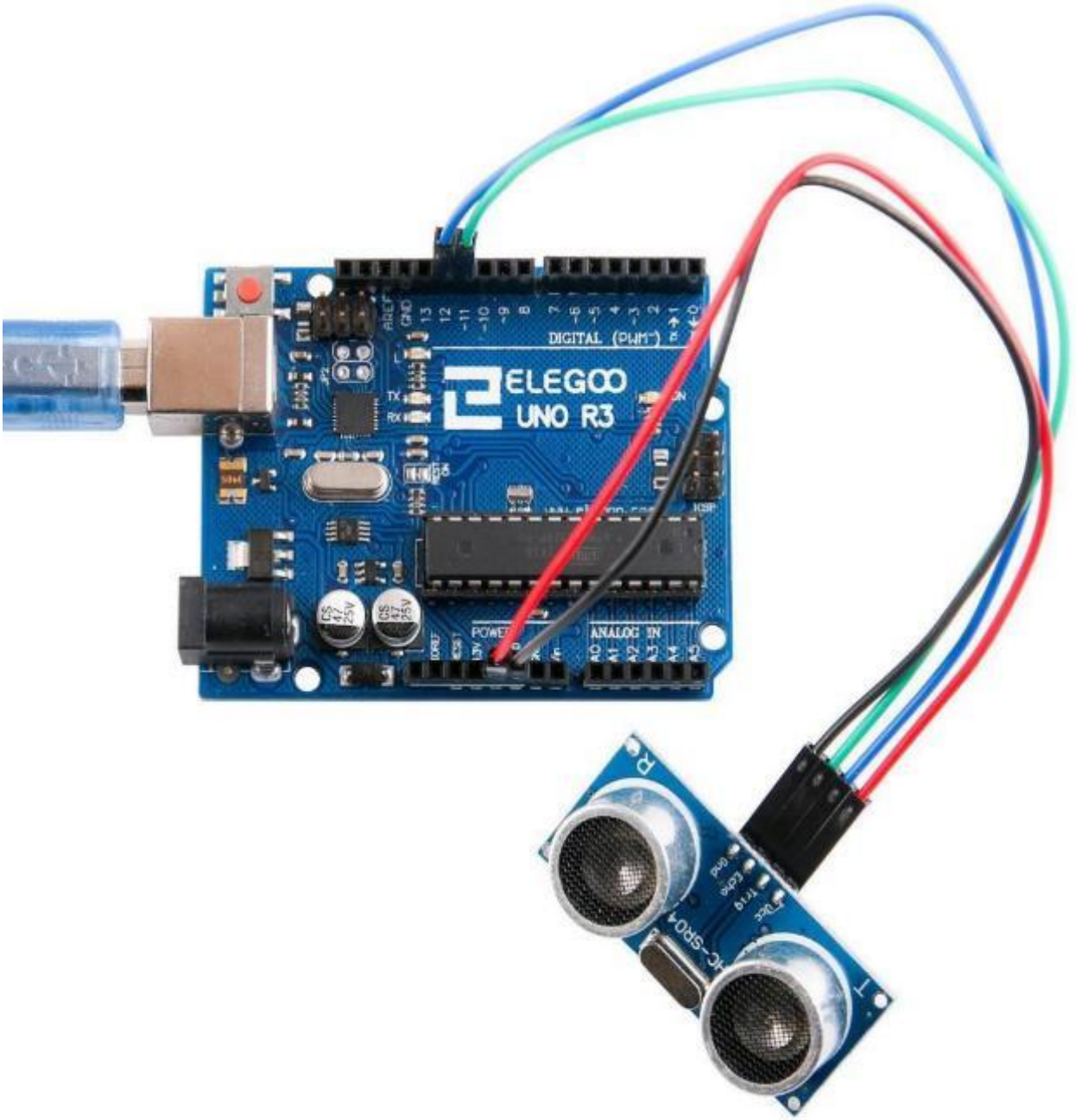


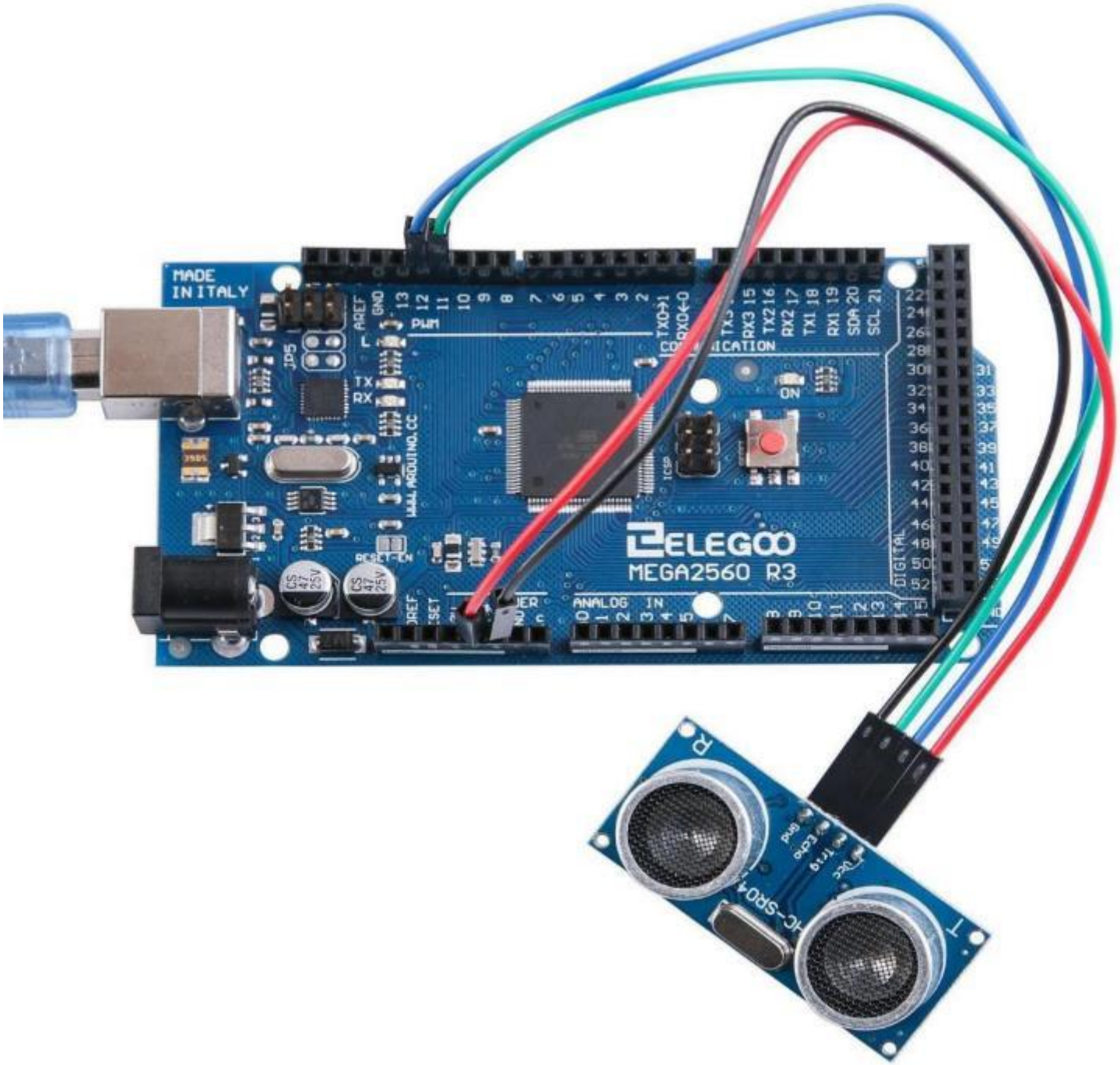


Code

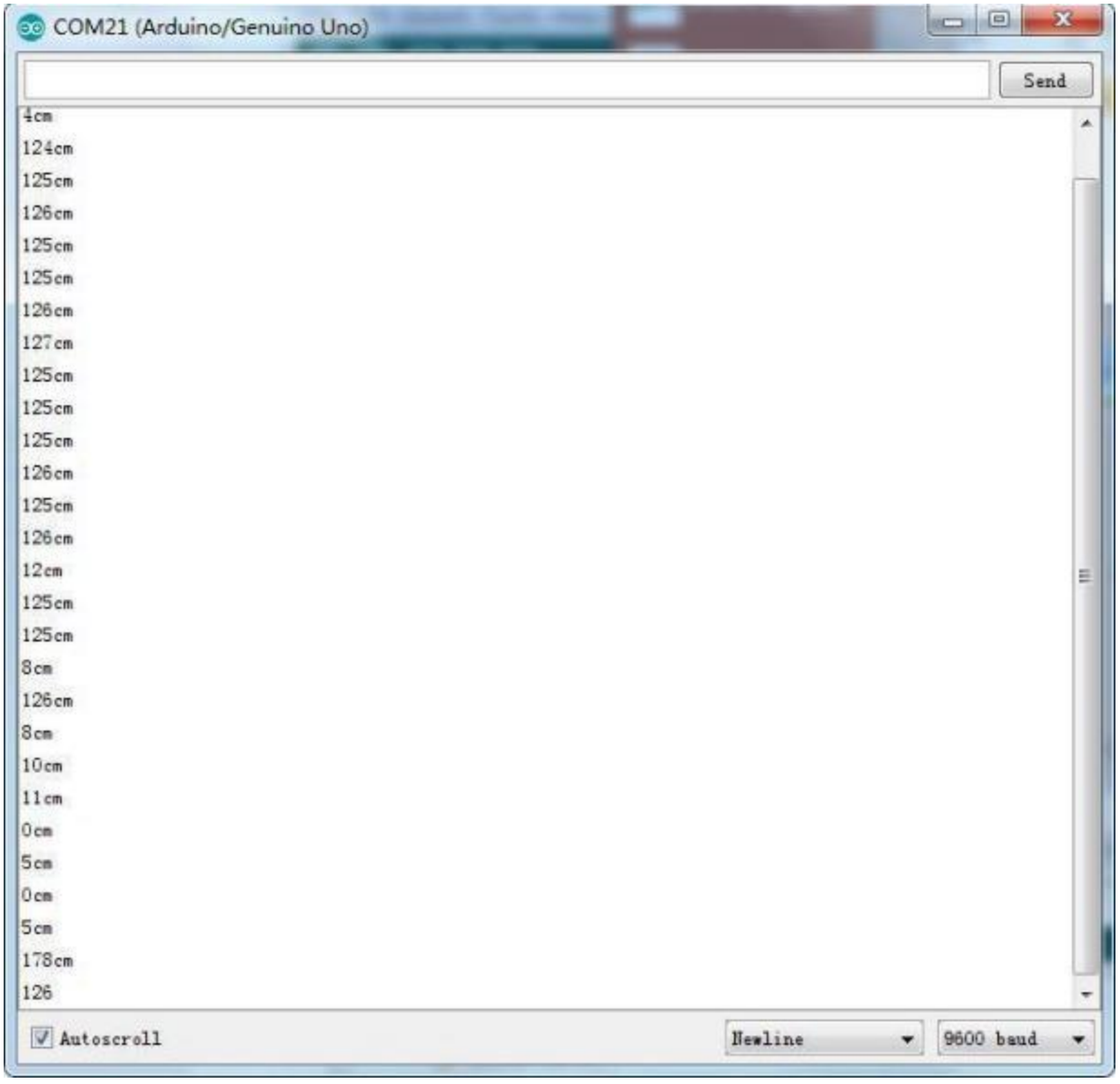
Die Verwendung einer Bibliothek, die für diese Sensoren entwickelt wurde, macht unseren Code kurz und einfach. Wir binden die Bibliothek am Anfang unseres Codes ein, und dann können wir mit einfachen Befehlen die Sensoren ansteuern.
Nachdem Sie die Verdrahtung abgeschlossen haben, öffnen Sie bitte das Programm im Code-Ordner - Lektion 29 Ultraschall-Sensormodul und klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen, falls es irgendwelche Fehlermeldungen gibt. Stellen Sie sicher, dass Sie die < NewPing>-Bibliothek installiert haben oder installieren Sie sie bei Bedarf neu. Andernfalls wird Ihr Code nicht funktionieren.
Einzelheiten zum Laden von Bibliotheken finden Sie in Lektion 1.

Beispielbild




MADE
IN ITALY
ELEGOO
MEGA2560 R3
ANALOG IN
DIGITAL
ON
PWM
TX
RX
HC-SR04

Klicken Sie auf die Schaltfläche „Serieller Monitor“, um den seriellen Monitor einzuschalten. Die Grundlagen des seriellen Monitors werden in Lektion 1 ausführlich erläutert. Der Monitor sollte ungefähr so aussehen wie im Screenshot. Natürlich sind die Zahlen abhängig davon, wie weit das nächste Objekt vom Sensor entfernt ist.



Nachfolgend finden Sie den für das Experiment benötigten Code：

```
#include <NewPing.h>
#define TRIGGER_PIN     12    //Pin, der den Messvorgang startet
#define ECHO_PIN        11    //Pin, der den vom Modul gemessenen Wert empfängt
#define MAX_DISTANCE 200 //Maximal können 200cm gemessen werden laut Spezifikation des Sensors
//Neues NewPing Objekt erzeugen und initialisieren
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
        Serial.begin(9600); //Initialisieren des seriellen Monitors
}

void loop() {
// Die Anzahl der Messungen auf 2 pro Sekunde begrenzen
delay(500);
// Messung starten, zurück kommt die Dauer, bis der Impuls zurückkam. Diesen Wert müssen wir noch
umrechnen in einen Abstand
unsigned int uS = sonar.ping();
Serial.print("Ping: ");
// Umrechnen der Zeit in den Abstand. US_ROUNDTRIP_CM ist eine Konstante aus der NewPing Bibliothek
Serial.print(uS / US_ROUNDTRIP_CM);
Serial.println("cm");
}
```

Lektion 30 MPU6050 Modul

Überblick

In dieser Lektion lernen wir wie man das MPU6050-Modul verwendet, einen der besten IMU (Inertia Measurement Unit) Sensoren, kompatibel mit Arduino. IMU-Sensoren wie der MPU 6050 werden unter anderem in selbstbalancierenden Robotern, Smartphones, etc. eingesetzt.

Benötigte Komponenten:

1 x Elegoo Uno R3

1 x MPU6050 Modul

4 x F-M Drähte

Komponenteneinführung

MPU6050 SENSOR

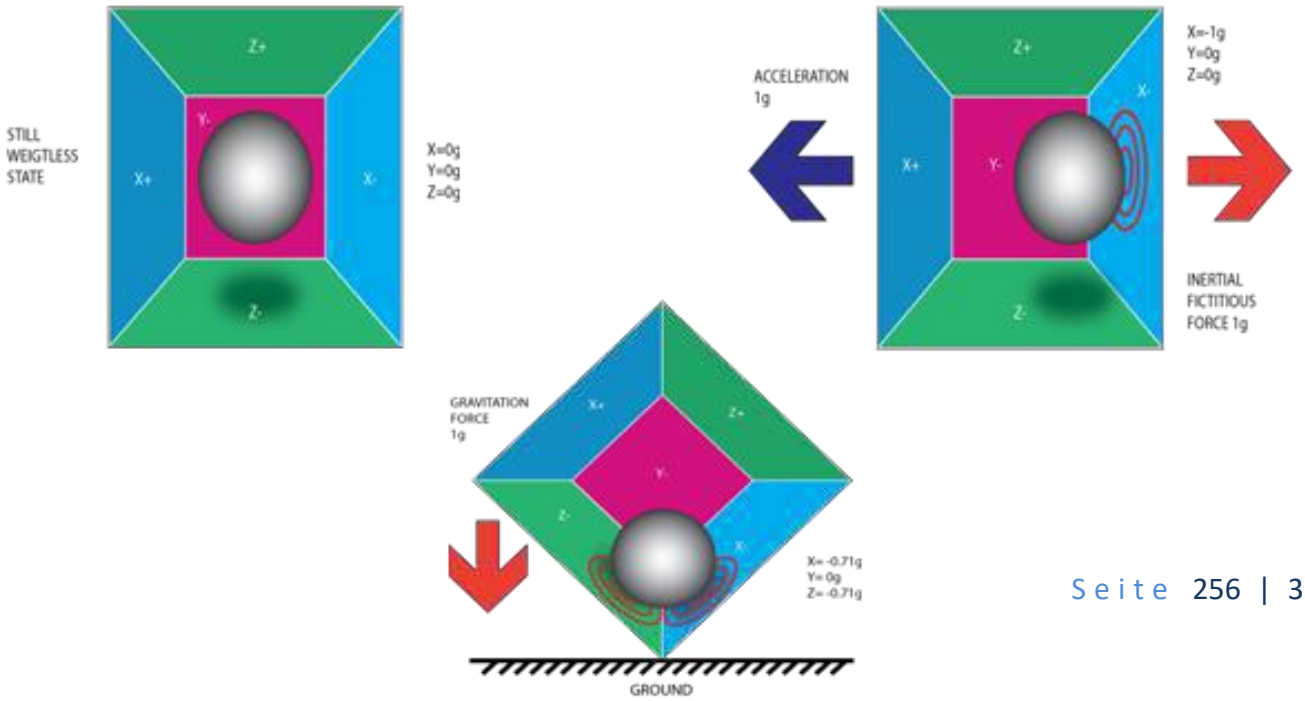
Der InvenSense MPU-6050 Sensor enthält einen MEMS-Beschleunigungssensor und einen MEMS-Kreisel auf einem einzelnen Chip. Er ist sehr genau, da er für jeden Kanal einen 16-Bit Analog-Digital-Wandler enthält. Deshalb erfasst er gleichzeitig den x-, y- und z-Kanal. Der Sensor verwendet den I2C-Bus zur Anbindung an den Arduino. Die MPU-6050 ist nicht teuer insbesondere, wenn man bedenkt, dass er sowohl einen Beschleunigungsmesser und einen Kreisel enthält.

IMU-Sensoren sind Sensoren, die heute in allen Arten von elektronischen Geräten verwendet werden. Sie sind in Smartphones, Wearables, Gamecontrollern usw. zu finden. IMU-Sensoren helfen uns dabei, die Lage eines Objekts – an dem der Sensor befestigt ist - in einem dreidimensionalen Raum zu bestimmen. Diese Werte sind in der Regel Winkel, aus denen die Lage bestimmt wird. So werden sie in Smartphones verwendet, um deren Lage zu erkennen. Und auch in tragbaren Gadgets wie dem Fitbit, verfolgen IMU-Sensoren die Bewegungen.

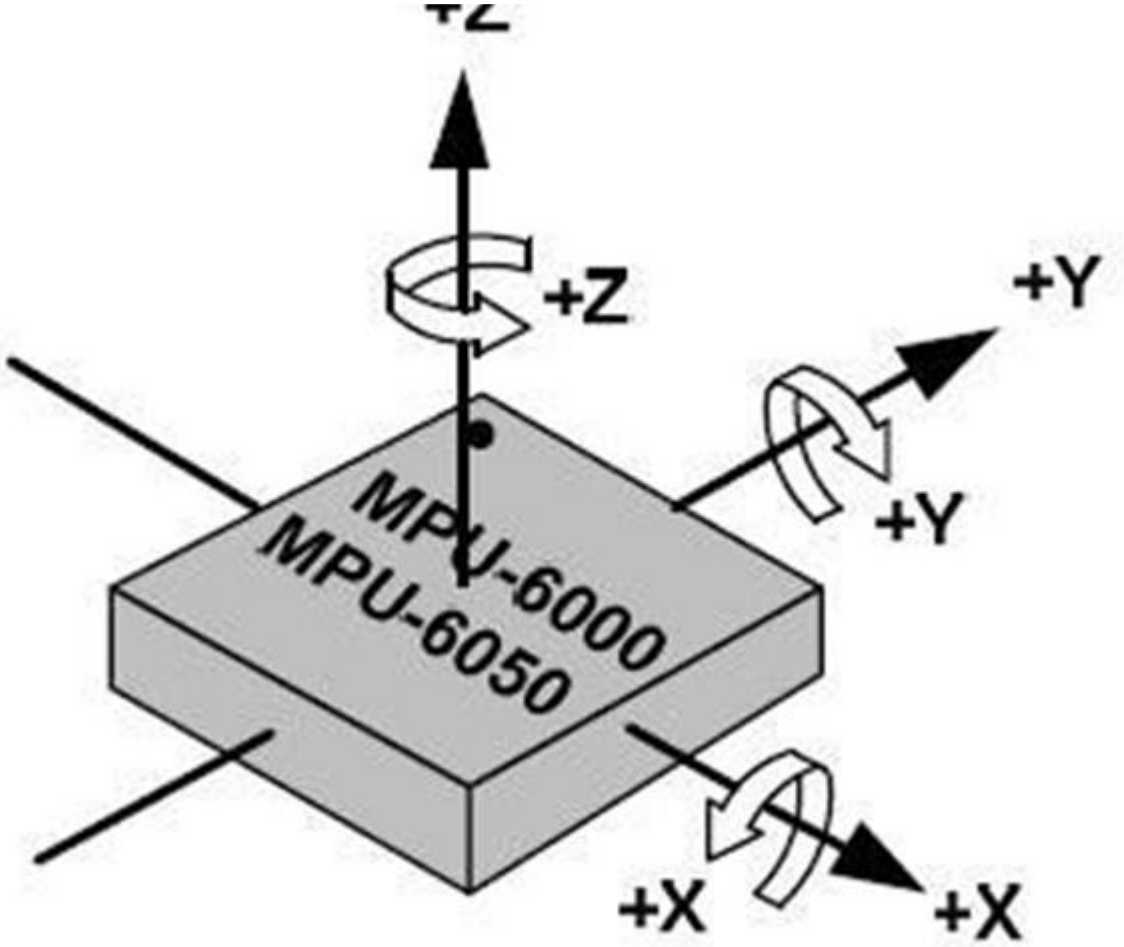
Wie funktionieren diese Sensoren nun?

IMU-Sensoren bestehen in der Regel aus zwei oder mehr Teilen. Aufgelistet nach Wichtigkeit sind das: Beschleunigungsmesser, Gyroskop, Magnetometer und Höhenmesser. Der MPU 6050 ist ein 6 DOF (Degrees of Freedom = Freiheitsgrade) oder ein 6-achsiger IMU-Sensor, was bedeutet, dass er sechs Werte als Ausgang liefert. Drei Werte vom Beschleunigungssensor und drei vom Gyroskop. Der MPU 6050 ist ein Sensor auf Basis der MEMS-Technologie (Micro Electro Mechanical Systems). Sowohl der Beschleunigungssensor als auch das Gyroskop sind in einem einzigen Chip untergebracht. Dieser Chip verwendet das I2C-Protokoll für die Kommunikation mit der Außenwelt.

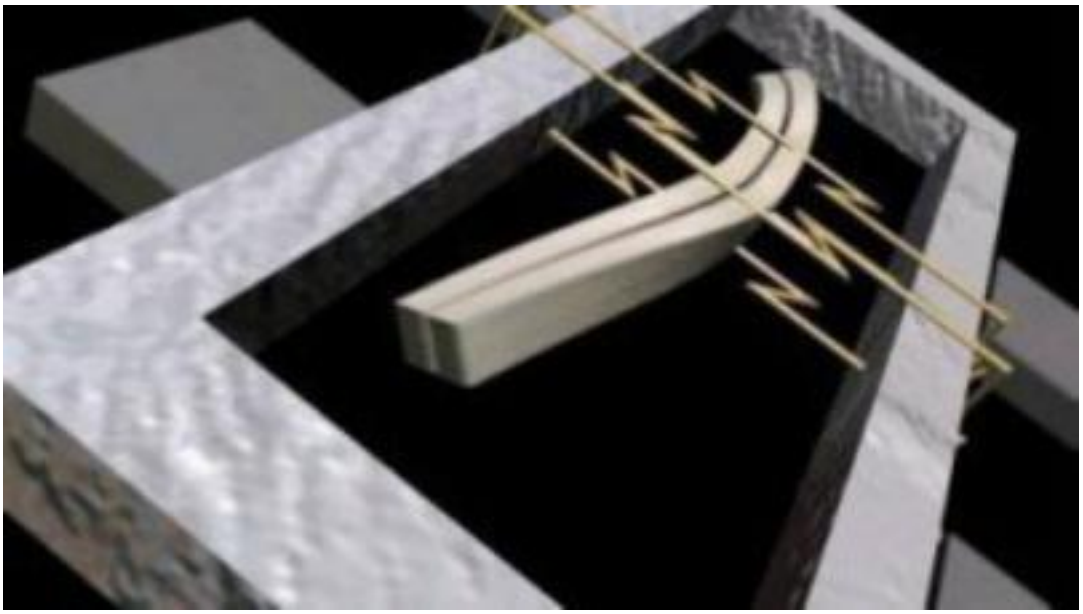
Wie funktioniert ein Beschleunigungsmesser?



Ein Beschleunigungssensor arbeitet nach dem Prinzip des piezoelektrischen Effekts. Stellen Sie sich vor Sie haben eine kleine Kugel im Inneren, wie auf dem Bild oben. Die Wände dieser Box sind aus piezoelektrischen Kristallen gefertigt. Immer wenn Sie die Box kippen, wird der Ball durch die Schwerkraft in Richtung der Neigung bewegt. Die Wand, mit der die Kugel kollidiert, erzeugt winzige piezoelektrische Ströme. Es gibt insgesamt drei Paare von gegenüberliegenden Wänden in einem Quader. Jedes Paar entspricht einer Achse im 3D-Raum: X-, Y- und Z-Achse. Abhängig vom Strom, der von den piezoelektrischen Wänden erzeugt wird, können wir die Richtung der Neigung und deren Größe bestimmen.



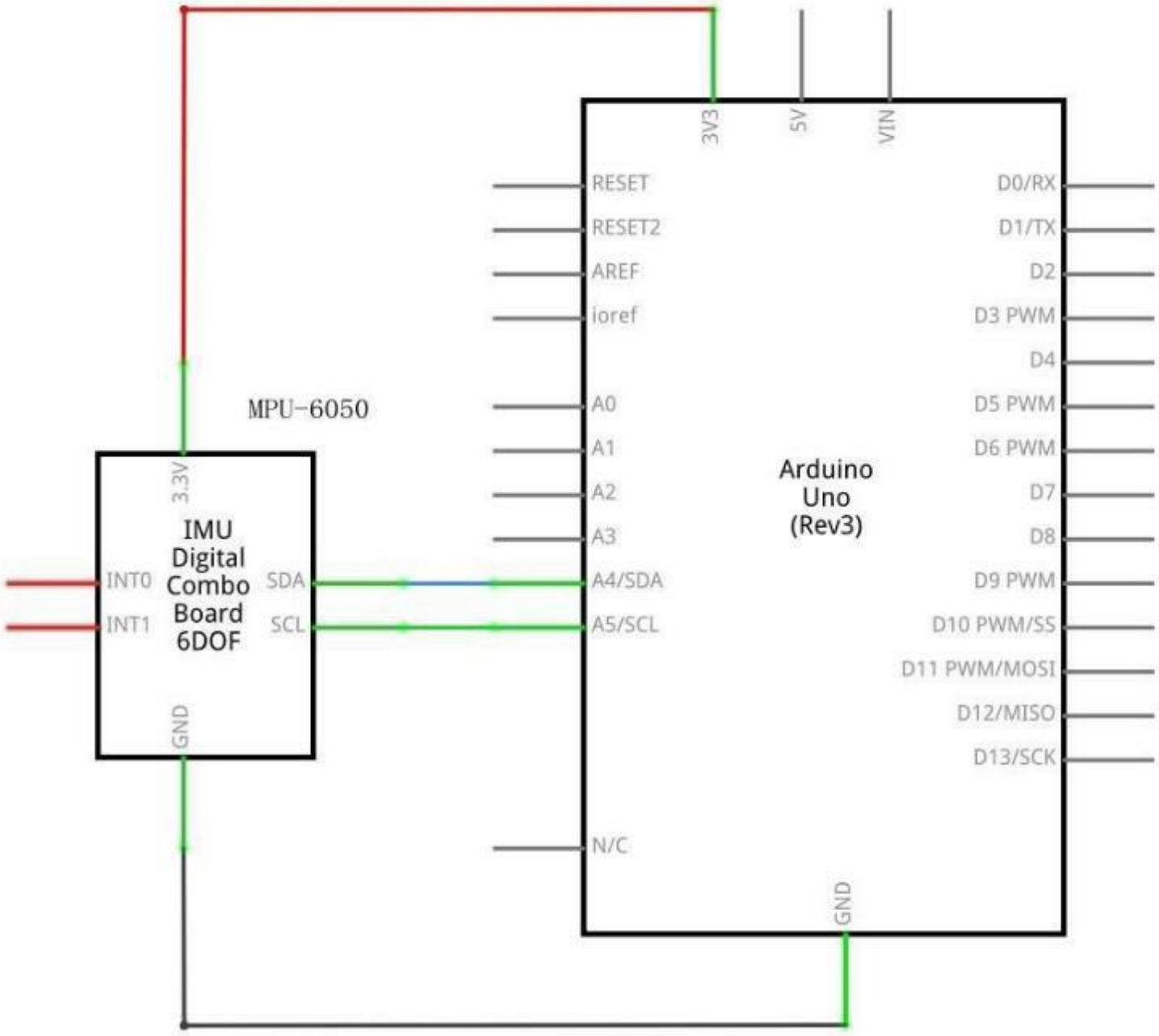
Wie funktioniert ein Gyroskop?

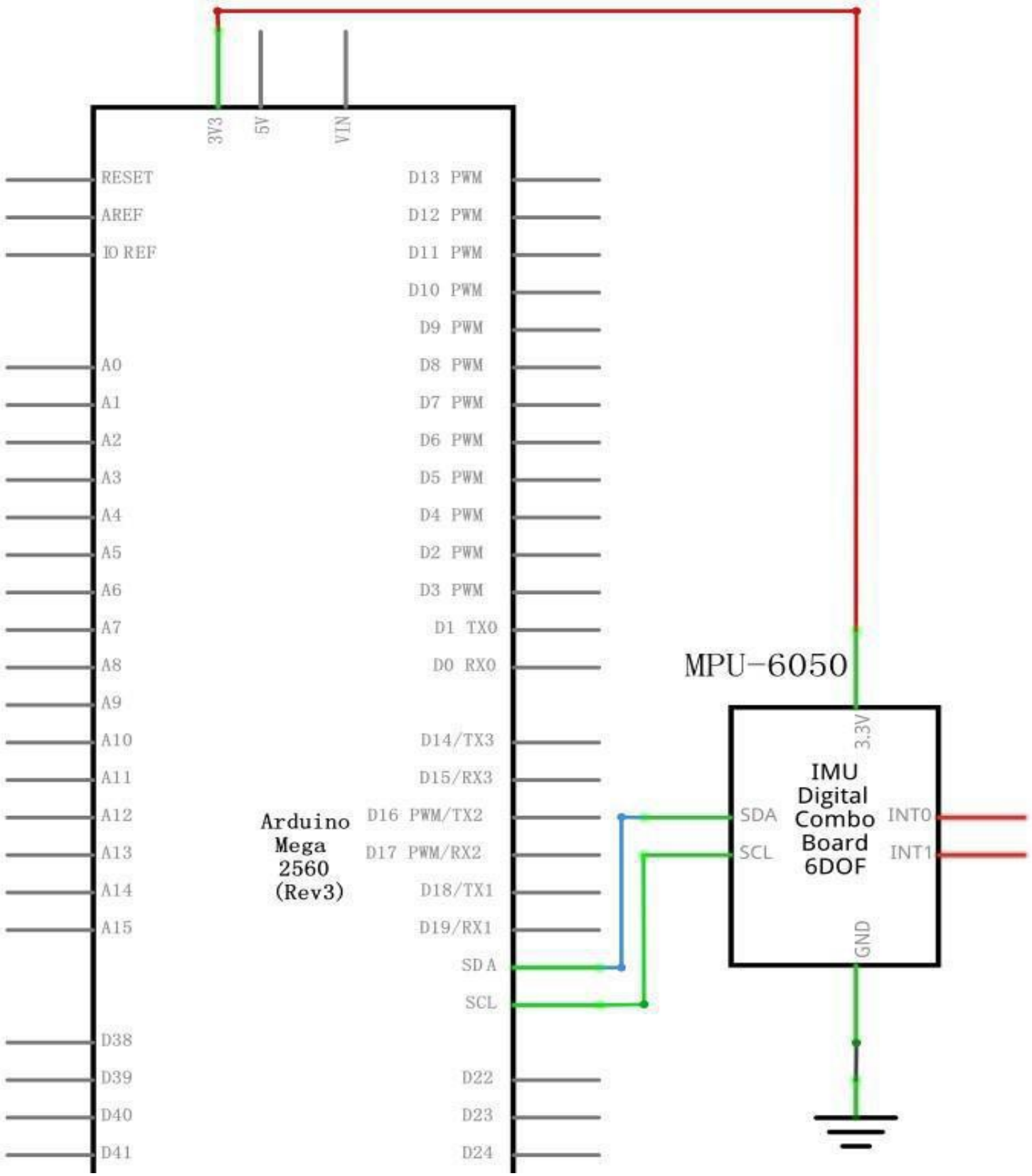


Gyroskope arbeiten nach dem Prinzip der Coriolis-Beschleunigung. Stellen Sie sich vor, dass es eine gabelartige Struktur gibt, die sich in ständiger Hin- und Herbewegung befindet. Es wird durch piezoelektrische Kristalle gehalten. Wann immer Sie versuchen diese Anordnung zu kippen, erfahren die Kristalle eine Kraft in Richtung der Neigung. Diese wird durch die Trägheit der beweglichen Gabel verursacht. Die Kristalle erzeugen also im Konsens mit dem piezoelektrischen Effekt einen Strom, der verstärkt wird. Die Werte werden dann von einem-Mikrocontroller verarbeitet.

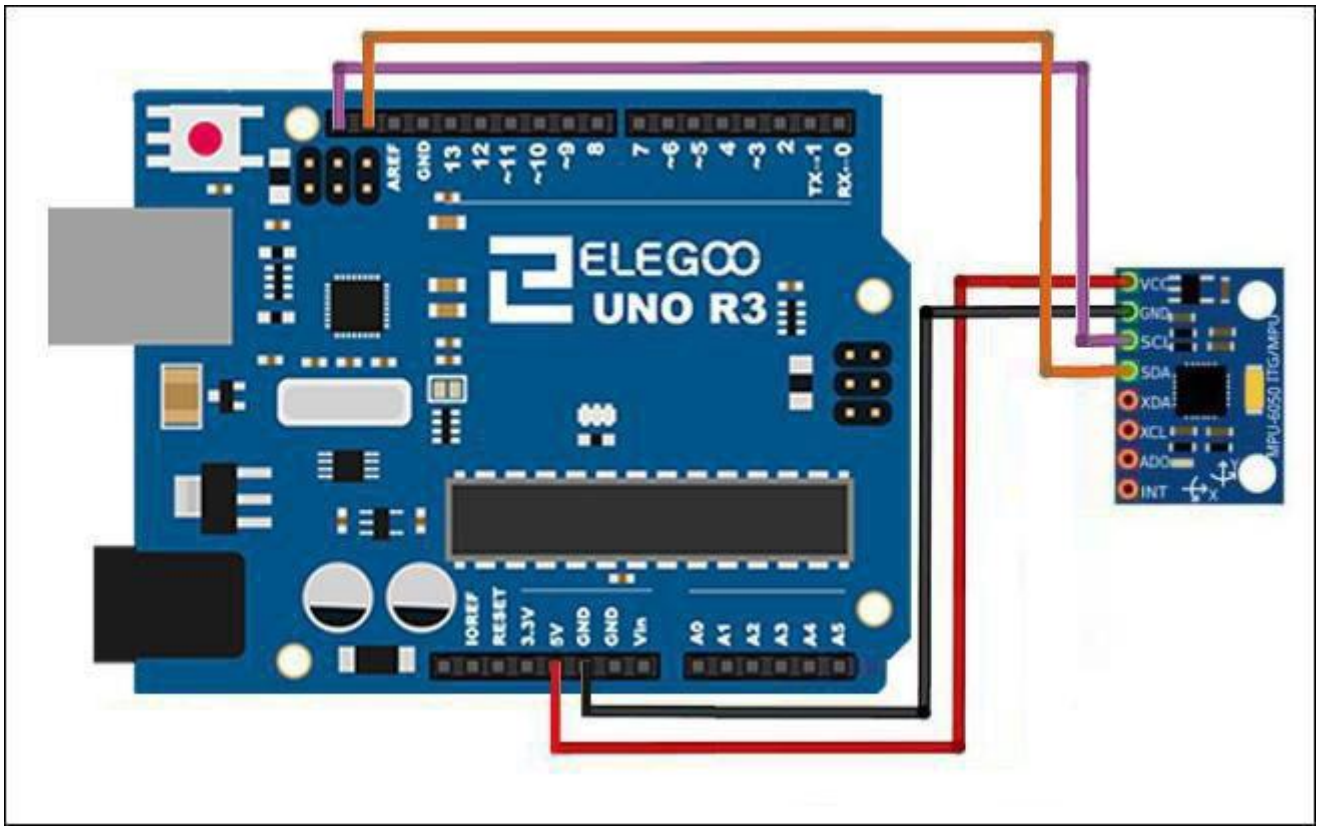
Verbindung

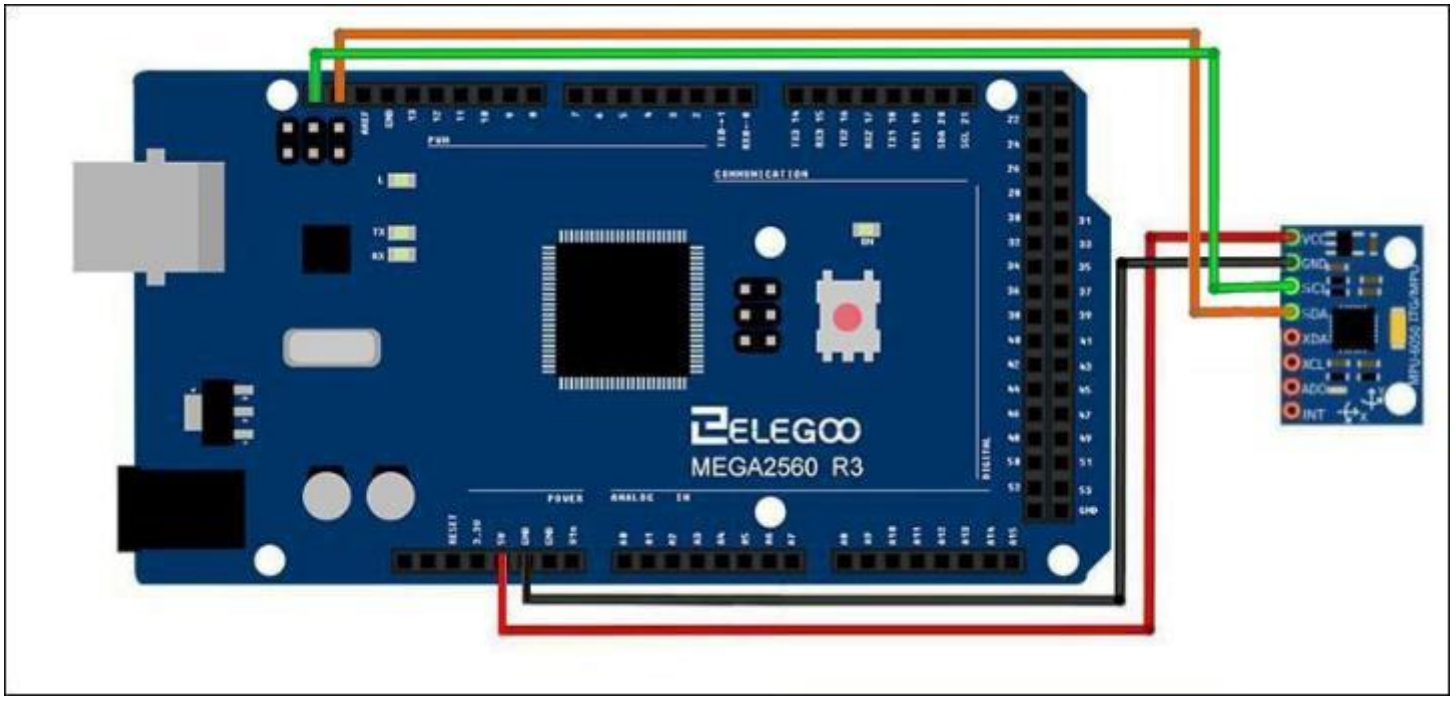
Schaltplan




3V3
5V
VIN
RESET
AREF
IO REF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D38
D39
D40
D41
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
D22
D23
D24
Arduino
Mega
2560
(Rev3)
MPU-6050
3.3V
IMU
Digital
Combo
Board
6DOF
SDA
SCL
INT0
INT1
GND

Verdrahtungsplan





Als nächstes müssen wir die I2C-Leitungen einrichten. Verbinden Sie dazu den als SDA gekennzeichneten Pin der MPU 6050 mit dem analogen Pin 4 (SDA) des Arduino. Und den als SCL bezeichnete Pin an der MPU 6050 an den analogen Pin 5 (SCL) des Arduino. Und das war's, Sie haben die Verdrahtung der Arduino MPU 6050 abgeschlossen.

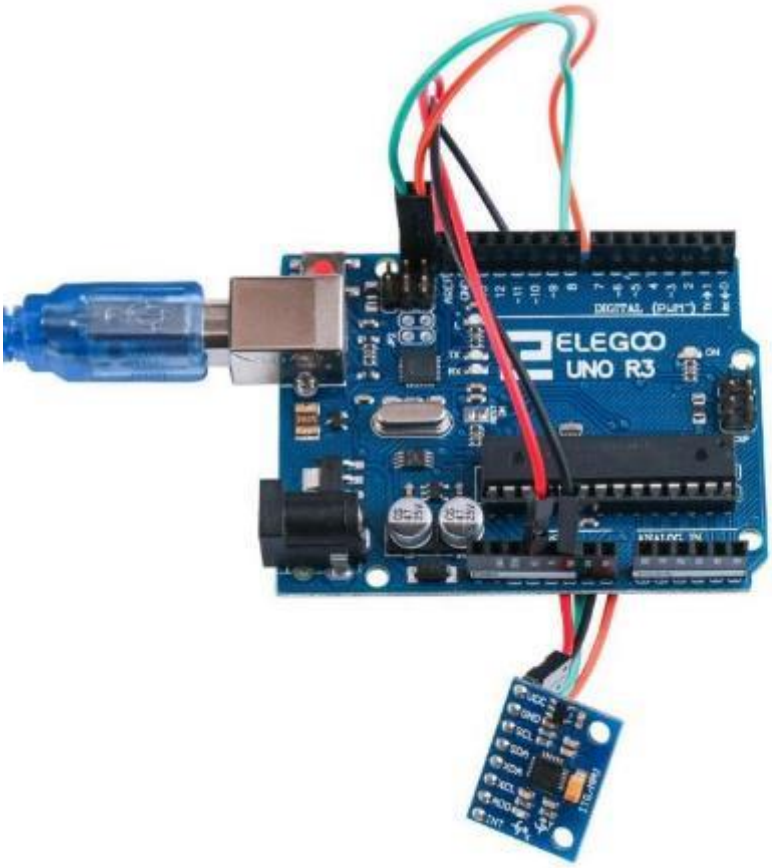
Benötigte Bibliotheken: MPU6050

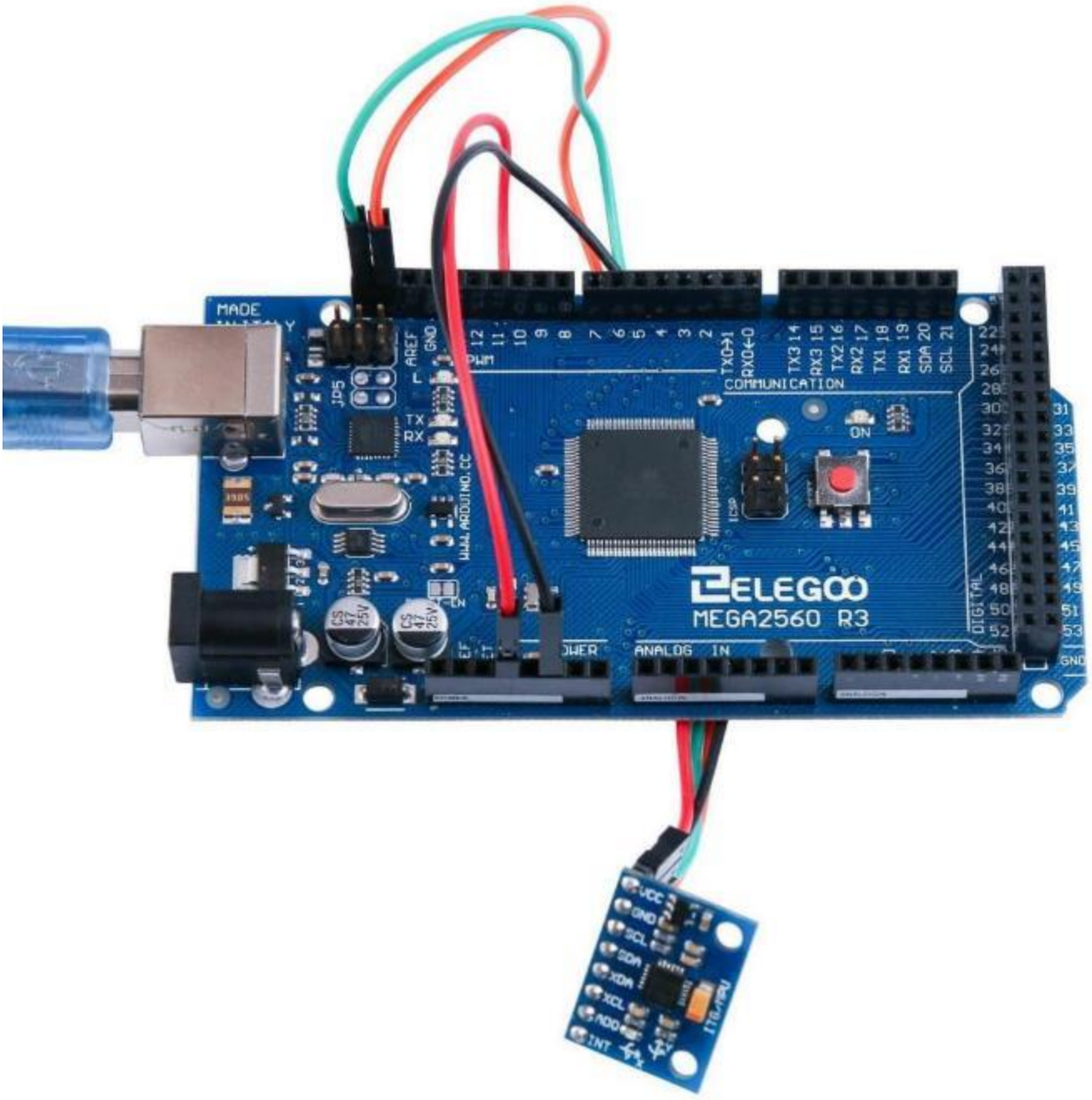
Der Code

Das kurze Beispielprogramm ist sehr rudimentär und zeigt nur alle Rohdaten an (Beschleunigungsmesser, Kreisel und Temperatur).

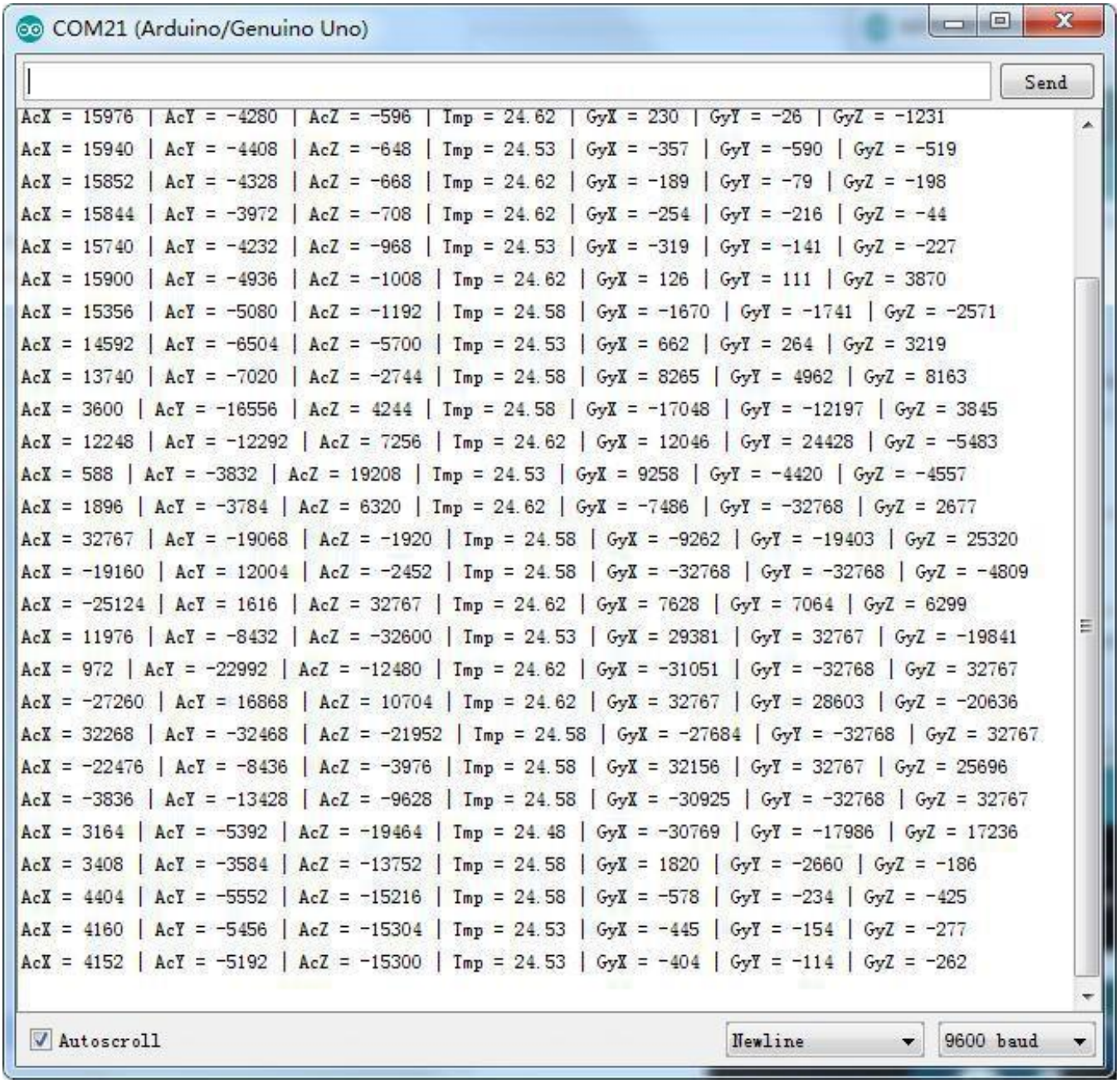
Nach der Verdrahtung laden Sie bitte das Programm „Lesson 30 MPU-6050 Module“ auf den Arduino.

Beispielbild




ELEGOO
MEGA2560 R3
COMMUNICATION
ANALOG IN
DIGITAL
ON
VCC
GND
SCL
SDA
XDA
XCL
ADO
INT

Öffnen Sie bitte den seriellen Monitor und die Ausgabe sollte in etwa so wie unten dargestellt aussehen.



Im Folgenden finden Sie den Code und einige Erklärungen

```
#include<Wire.h>
// I2C Adresse des MPU-6050
const int MPU_addr=0x68;
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);
  // 0 weckt den MPU-6050 auf
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600);
}
void loop(){
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);
  Wire.endTransmission(false);
  // Abfrage von 14 Registern
  Wire.requestFrom(MPU_addr,14,true);
  // Reihenfolge der Werte laut Datenblatt…jeder Wert hat 16 Bit, also 2 Byte
  AcX=Wire.read()<<8|Wire.read();
  AcY=Wire.read()<<8|Wire.read();
  AcZ=Wire.read()<<8|Wire.read();
  Tmp=Wire.read()<<8|Wire.read();
  GyX=Wire.read()<<8|Wire.read();
  GyY=Wire.read()<<8|Wire.read();
  GyZ=Wire.read()<<8|Wire.read();
  Serial.print("AcX = "); Serial.print(AcX);
  Serial.print(" | AcY = "); Serial.print(AcY);
  Serial.print(" | AcZ = "); Serial.print(AcZ);
```

```
    //Umrechnung der Temperatur in Grad Celsius laut Datenblatt
    Serial.print(" | Tmp = "); Serial.print(Tmp/340.00+36.53);
    Serial.print(" | GyX = "); Serial.print(GyX);
    Serial.print(" | GyY = "); Serial.print(GyY);
    Serial.print(" | GyZ = "); Serial.println(GyZ);
    delay(333);
}
```

Lektion 31 HC-SR501 PIR Sensor

Überblick

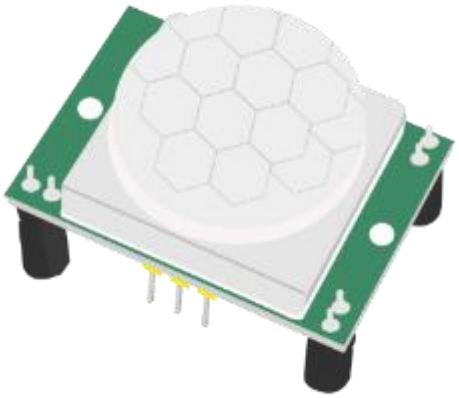
In dieser Lektion lernen Sie, wie man einen PIR-Bewegungsmelder mit einem Arduino UNO verwendet. Der UNO ist das Herzstück dieses Projekts. Er "hört" auf den PIR-Sensor und schaltet eine LED an oder aus, wenn eine Bewegung erkannt wird.

Benötigte Komponenten:

1 x Elegoo Uno R3

1 x HC-SR501 PIR Bewegungsmelder

3 x F-M Drähte

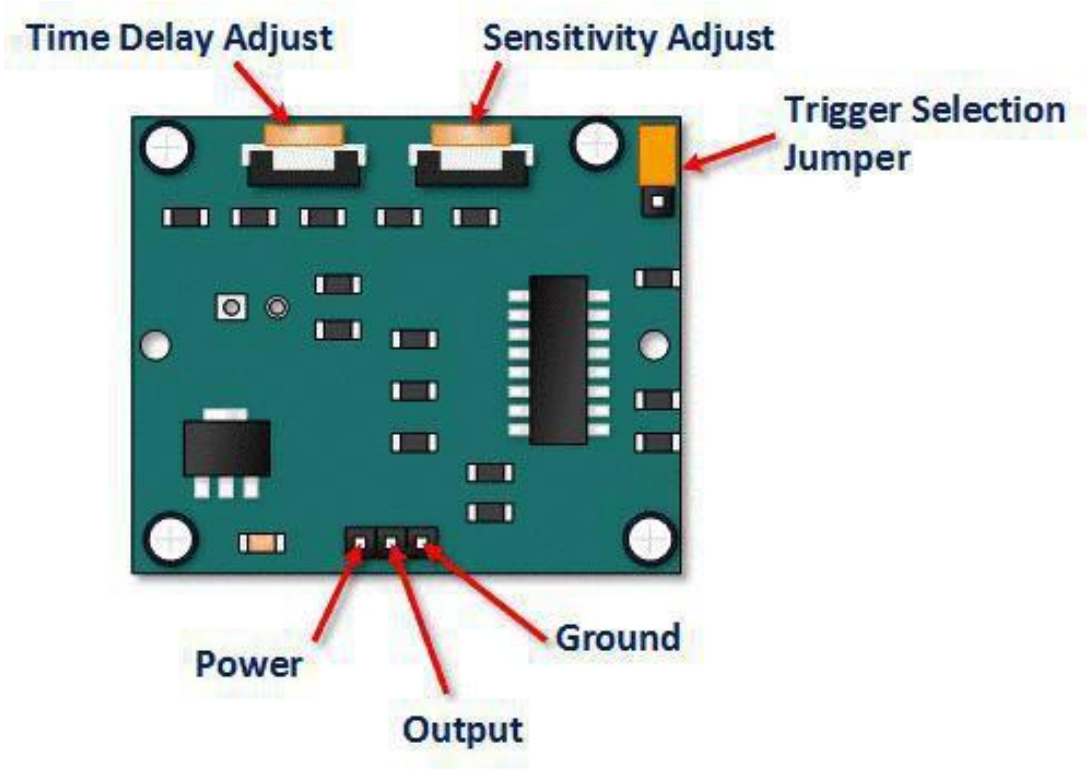


Komponenteneinführung

PIR Sensor:

PIR-Sensoren sind komplizierter als viele der anderen in diesem Dokument erläuterten Sensoren, da es mehrere Variablen gibt, die die Funktion der Sensoren beeinflussen.

Der PIR-Sensor selbst hat zwei Sensoren. Jeder IR-Empfänger besteht aus einem speziellen, IR-empfindlichen Material. Wenn der Sensor im Leerlauf ist, erfassen beide IR-Empfänger die gleiche Menge an IR (infrarotem Licht), die vom Raum oder von Wänden abgestrahlt wird. Wenn ein warmer Körper wie ein Mensch oder ein Tier vorbeikommt, deckt er zunächst eine Hälfte des PIR-Sensors ab, was zu einer positiven Differenzveränderung zwischen den beiden Hälften führt. Verlässt der warme Körper den Erfassungsbereich, erfolgt die Umkehrung, wobei der Sensor eine negative Differenzveränderung erzeugt. Diese Änderungsimpulse werden erkannt.



Pin or Control	Function
Time Delay Adjust	Legt fest, wie lange der Ausgang nach der Bewegungserkennung ‚high' bleibt. Einstellbar von 5 Sekunden bis 5 Minuten.
Sensitivity Adjust	Stellt den Erfassungsbereich ein. Einstellbar von 3 Meter bis 7 Meter
Trigger Selection Jumper	Einstellung für einzelnen oder wiederholbaren Trigger
Ground pin	Masseanschluss
Output Pin	Low, wenn keine Bewegung erkannt wird. High, wenn Bewegung erkannt wird. High ist bei dem Modul 3.3V

HC SR501 PIR Funktionsbeschreibung

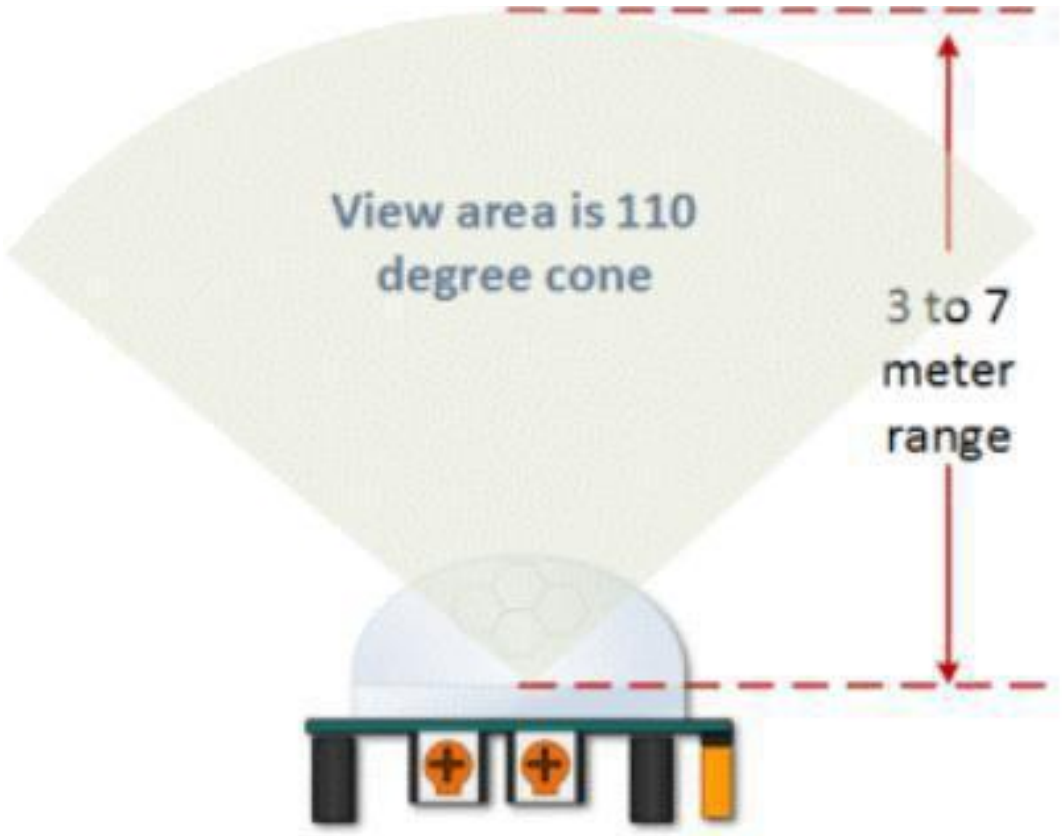
Das SR501 erkennt Infrarotlichtveränderungen und setzt, wenn er diese als Bewegung interpretiert, seinen Ausgang auf High. Was als Bewegung interpretiert wird, hängt stark von den Einstellungen des Benutzers ab.

Geräte-Initialisierung

Das Gerät benötigt fast eine Minute zur Initialisierung. Während dieser Zeit kann und wird es oft falsche Signale ausgeben. Diese Initialisierungszeit muss bei der Schaltung bzw. Steuerungslogik berücksichtigt werden.

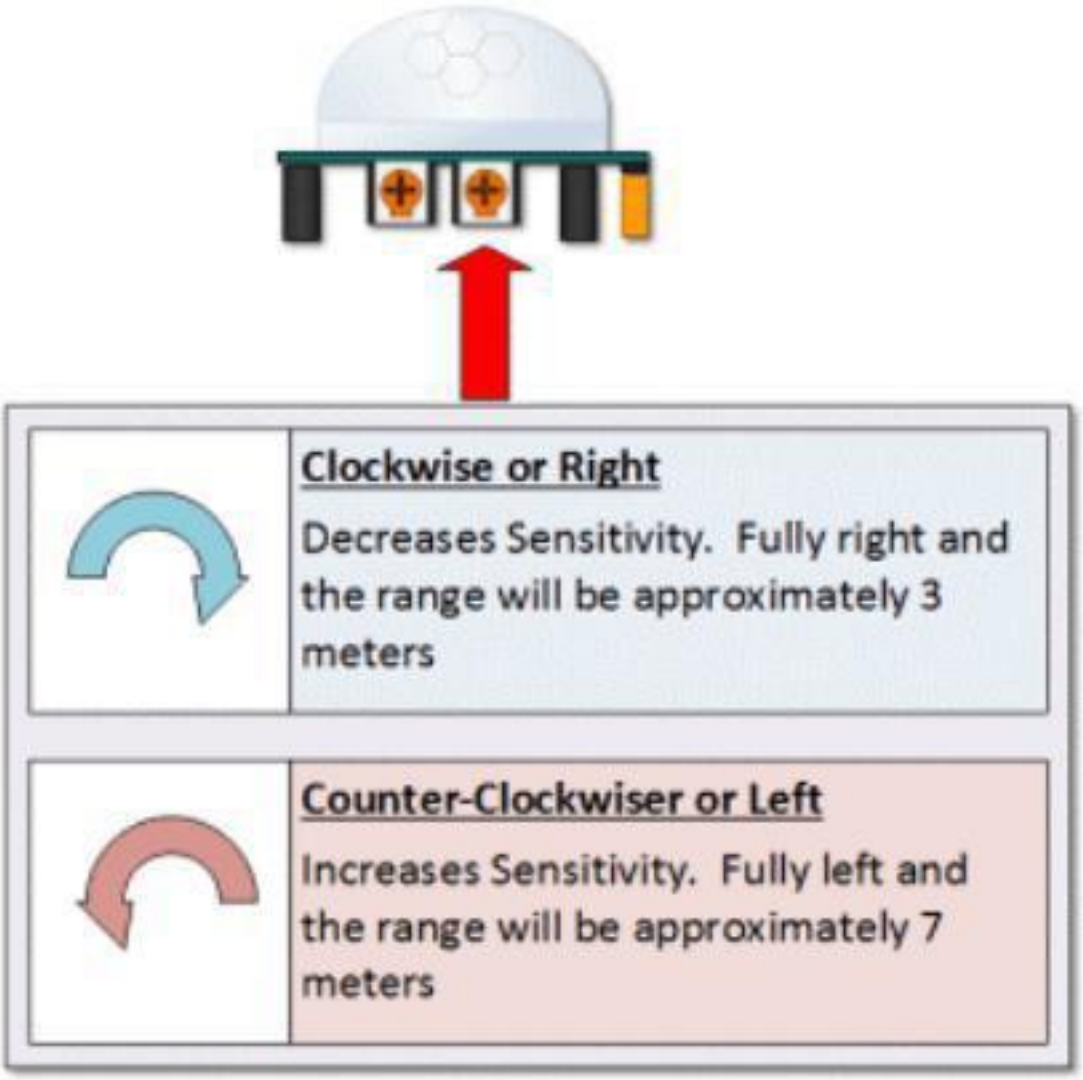
Erkennungsbereich

Das Gerät erkennt Bewegungen innerhalb eines 110 Grad Bereichs mit einer Reichweite von 3 bis 7 Metern.



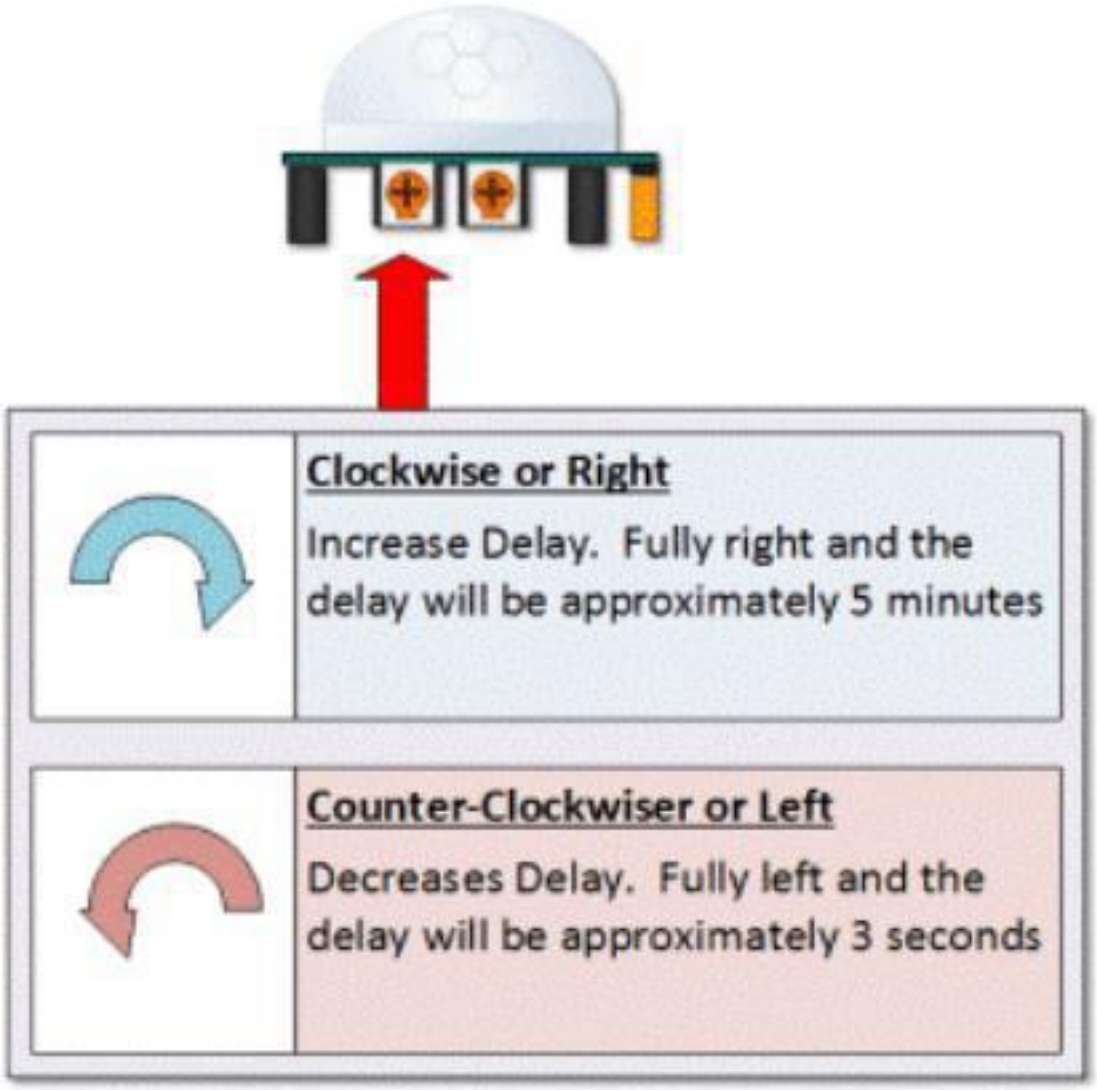
HC SR501 Empfindlichkeitseinstellung

Die Bereichseinstellung (Empfindlichkeit) ist wie erwähnt zwischen ca. 3 und 7 Metern. Die folgende Abbildung zeigt diese Einstellung. Drehen im Uhrzeigersinn verringert die Empfindlichkeit, Drehen entgegen des Uhrzeigersinns erhöht die Empfindlichkeit.



HC SR501 Sensitivity Adjust Time Delay Adjustment

Die Zeitverzögerungseinstellung bestimmt, wie lange der Ausgang des PIR-Sensormoduls nach der Bewegungserkennung ‚high' bleibt. Der Bereich reicht von ca. 5 Sekunden bis zu 5 Minuten. Drehen im Uhrzeigersinn erhöht die Zeitdauer, drehen entgegen des Uhrzeigersinns verringert sie.



HC SR501 Totzeit nach Bewegungserkennung

Der Ausgang dieses Gerätes geht nach Ablauf der Zeitverzögerung für ca. 3 Sekunden auf ‚low'. Mit anderen Worten, die gesamte Bewegungserkennung wird während dieser drei Sekunden blockiert.

Beispiel:

Stellen Sie sich vor Sie befinden sich im Single-Trigger-Modus und Ihre Zeitverzögerung ist auf 5 Sekunden eingestellt. Der Sensor erkennt die Bewegung und legt den Ausgang für 5 Sekunden auf ‚high'. Nach fünf Sekunden legt der PIR seinen Ausgang für ca. 3 Sekunden auf ‚low'.

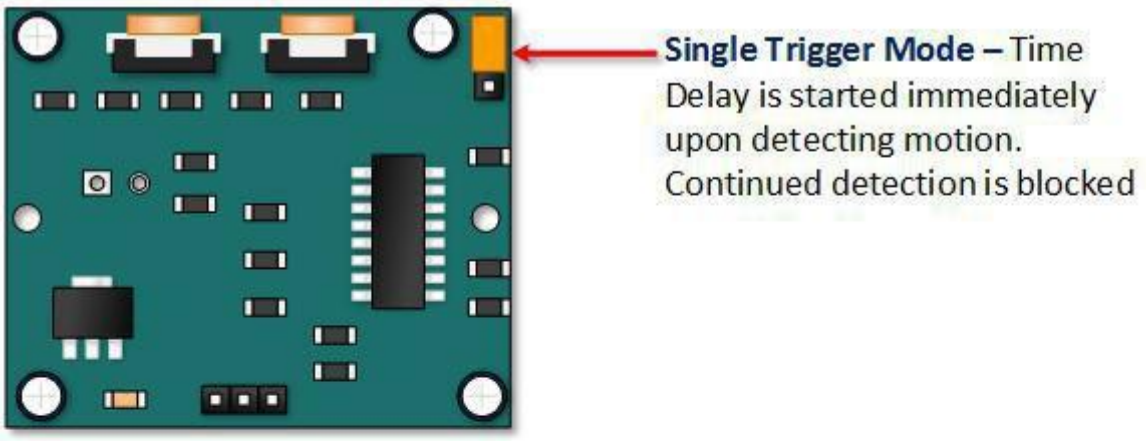
Während der 3 Sekunden erkennt der Sensor keine Bewegung. Nach den 3 Sekunden erkennt der Sensor wieder Bewegungen.

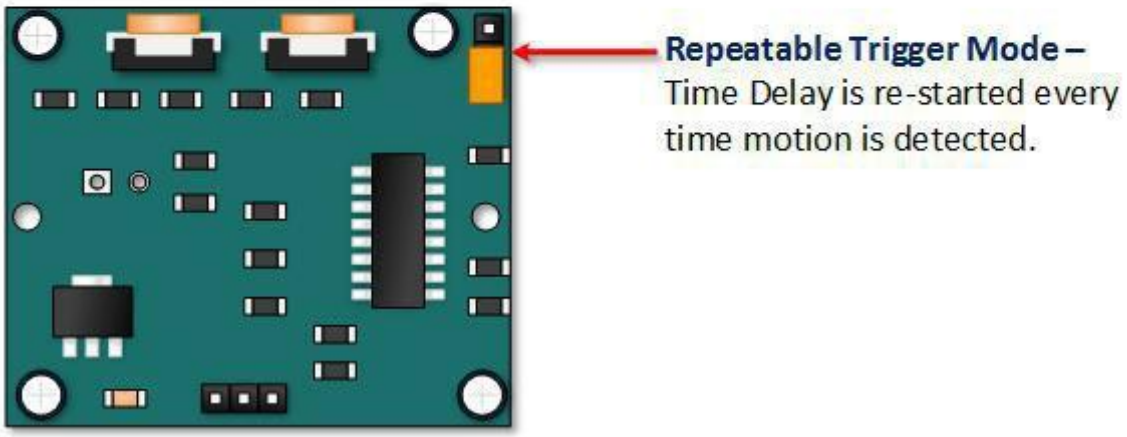
Triggermodus Auswahl Jumper

Mit dem Triggermodus-Auswahl-Jumper können Sie zwischen einzelnen und wiederholbaren Triggern wählen. Der Sinn dieser Jumper-Einstellung ist festzulegen wann die Zeitverzögerung beginnt.

SINGLE TRIGGER – Die Zeitverzögerung beginnt sofort, wenn die Bewegung zum ersten Mal erkannt wird.

REPEATABLE TRIGGER – Jede erkannte Bewegung setzt die Zeitverzögerung zurück. Die Zeitverzögerung beginnt also mit der letzten erkannten Bewegung.



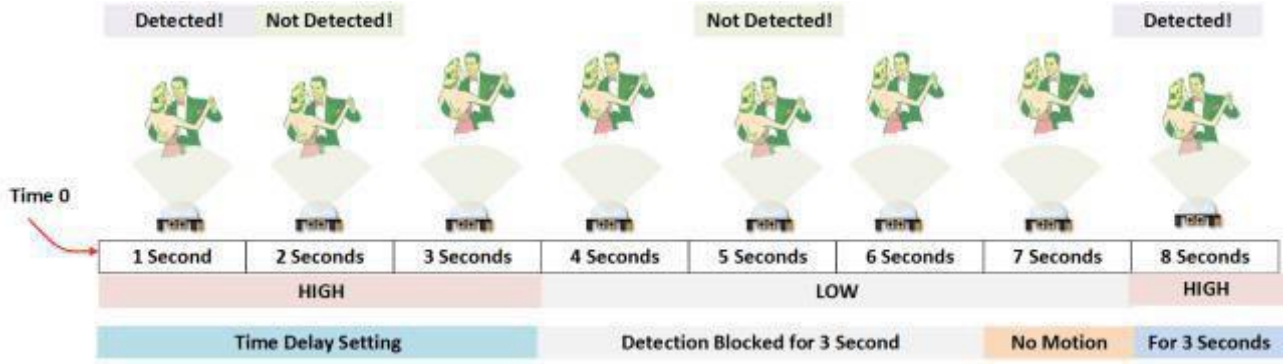


HC-SR501 Tanzflächenbeispiel zur Erklärung der Triggermodi

Stellen Sie sich vor, Sie wollen das Licht auf einer Tanzfläche steuern je nachdem wo die Tänzer tanzen. Das Verständnis, wie die Zeitverzögerung und der Triggermodus zusammenwirken ist notwendig um diese Beleuchtung in der von Ihnen gewünschten Weise zu steuern.

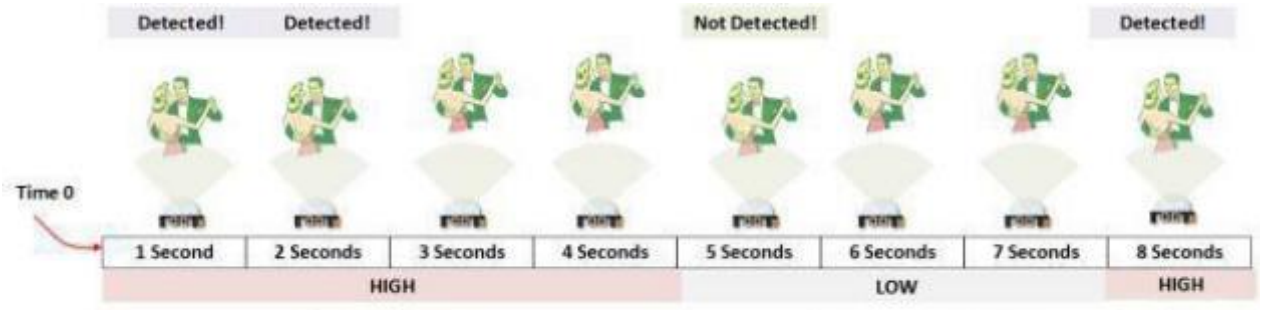
Beispiel 1

In diesem ersten Beispiel ist die Zeitverzögerung auf drei Sekunden und der Triggermodus auf Single eingestellt. Wie Sie in der Abbildung unten sehen können, wird die Bewegung nicht immer erkannt. Tatsächlich gibt es eine Zeitspanne von etwa sechs Sekunden, in der Bewegungen nicht erkannt werden können. Gleich am Anfang wird die Bewegung erkannt und der Ausgang bleibt 3s auf ‚high'. Danach folgen die 3s Totzeit während derer ebenfalls nichts erkannt wird. In Sekunde 7 findet keine Bewegung statt und in Sekunde 8 gibt es wieder eine Bewegung, die auch erkannt wird.



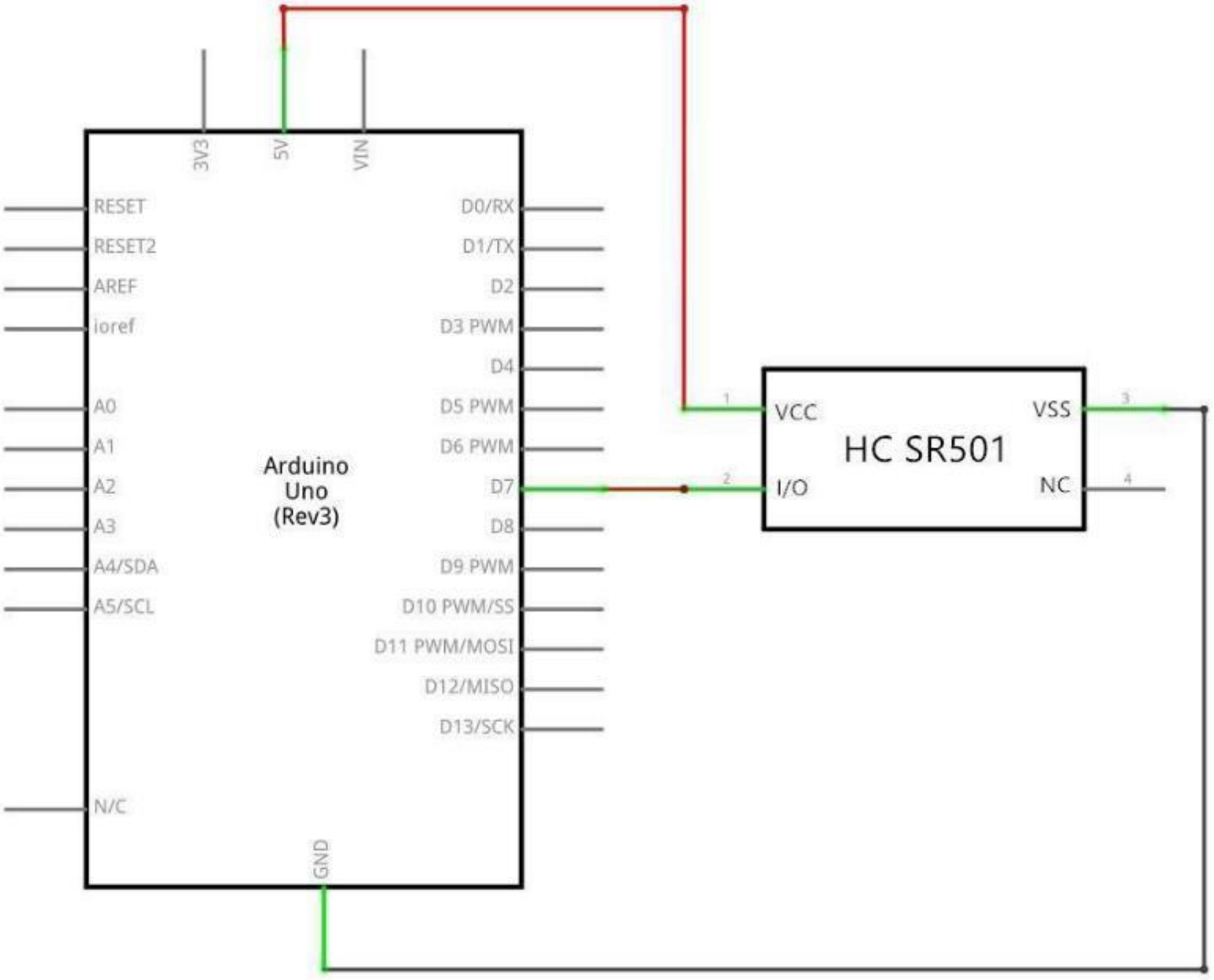
Beispiel 2

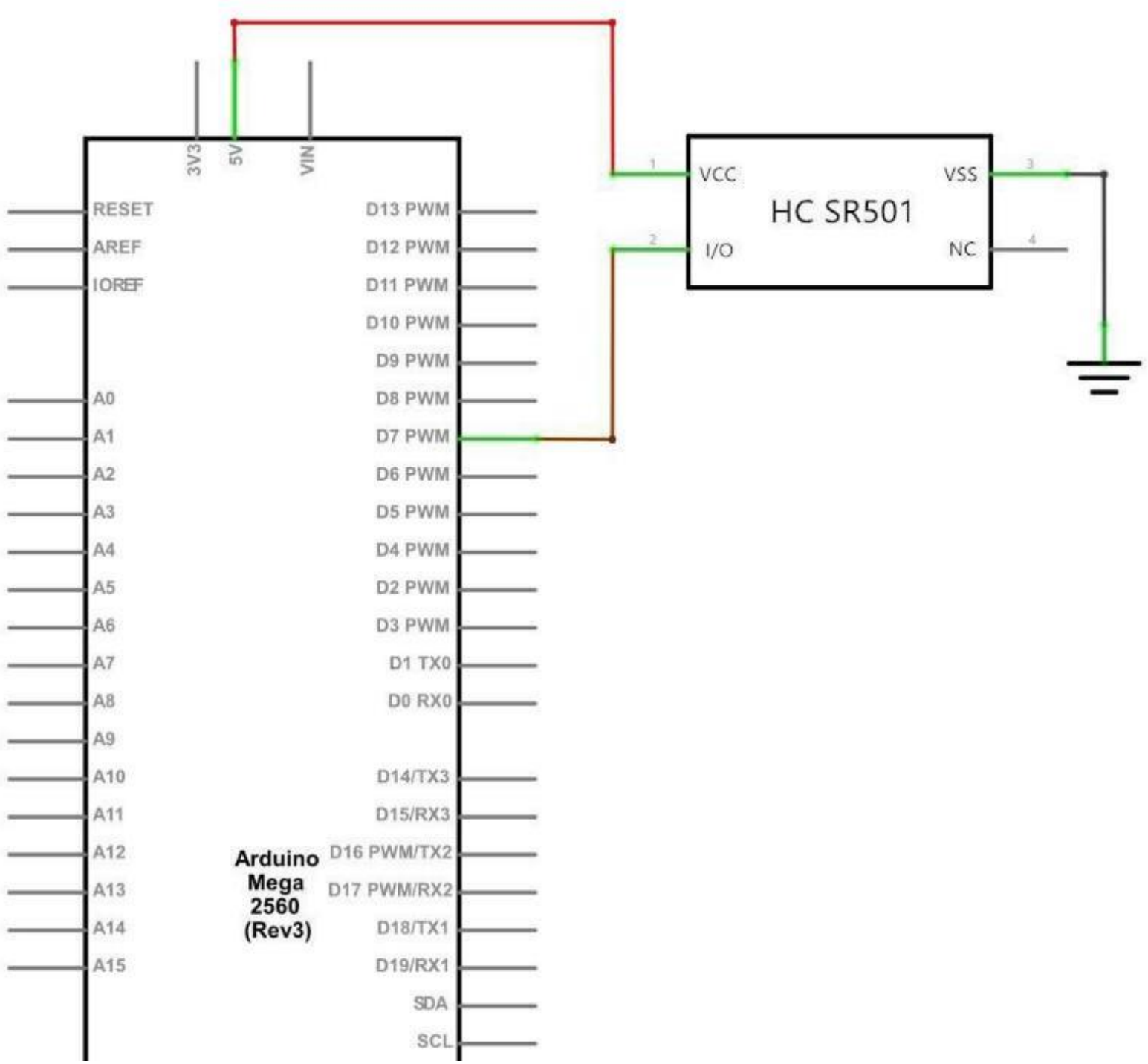
Im nächsten Beispiel liegt die Zeitverzögerung auch bei drei Sekunden und der Trigger ist auf Repeatable eingestellt. In der folgenden Abbildung sehen Sie, dass die Zeitverzögerung neu gestartet wird. Nach diesen drei Sekunden wird die Erkennung jedoch immer noch für drei Sekunden blockiert. Gleich am Anfang wird die Bewegung erkannt und der Ausgang geht auf ‚high'. Nach der ersten Sekunde wird wieder eine Bewegung erkannt und die 3s ‚high-Zeit' starten von vorne. Danach folgen die 3s Totzeit während derer nichts erkannt wird. In Sekunde 8 gibt es wieder eine Bewegung, die auch erkannt wird.



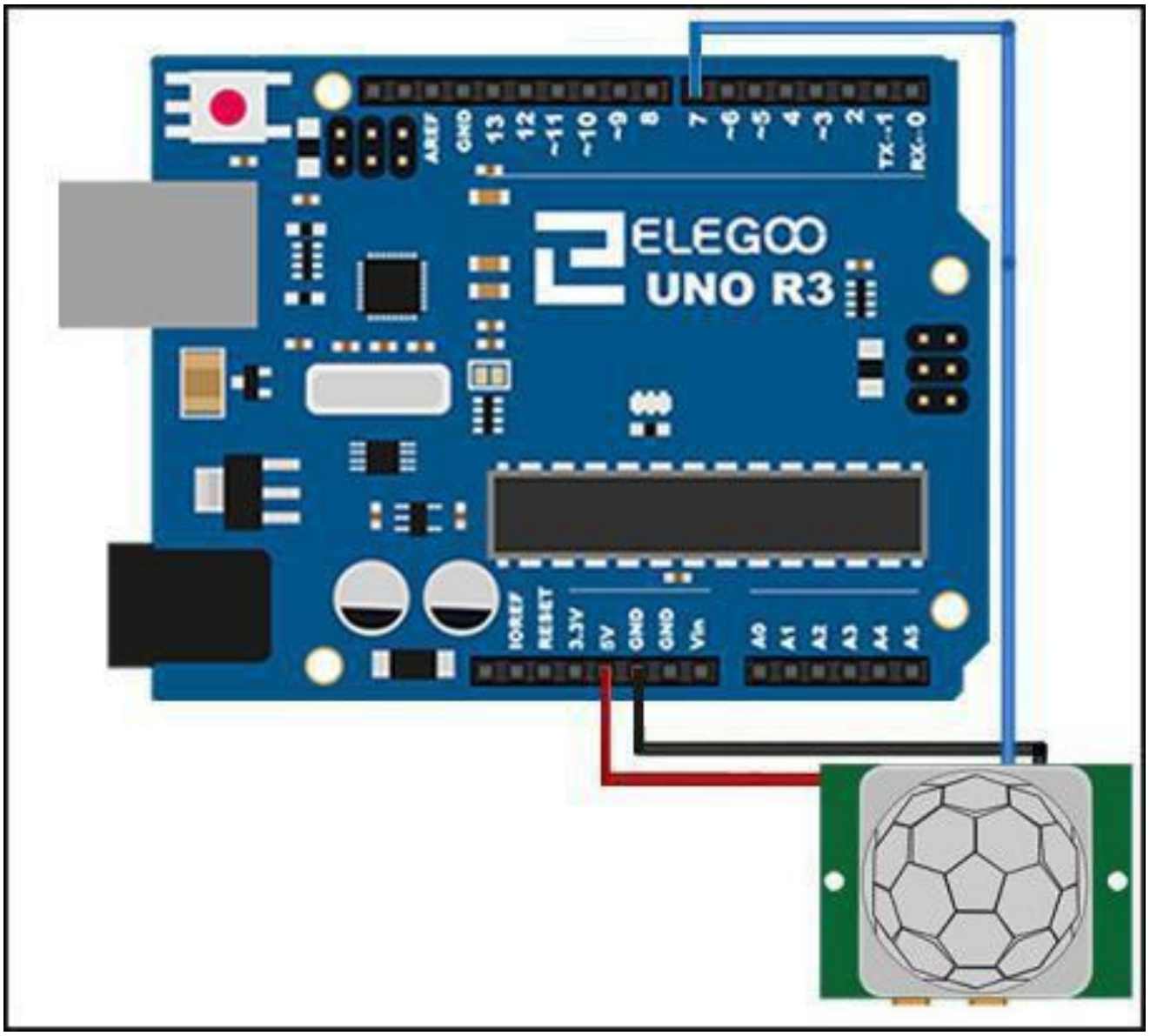
Verbindung

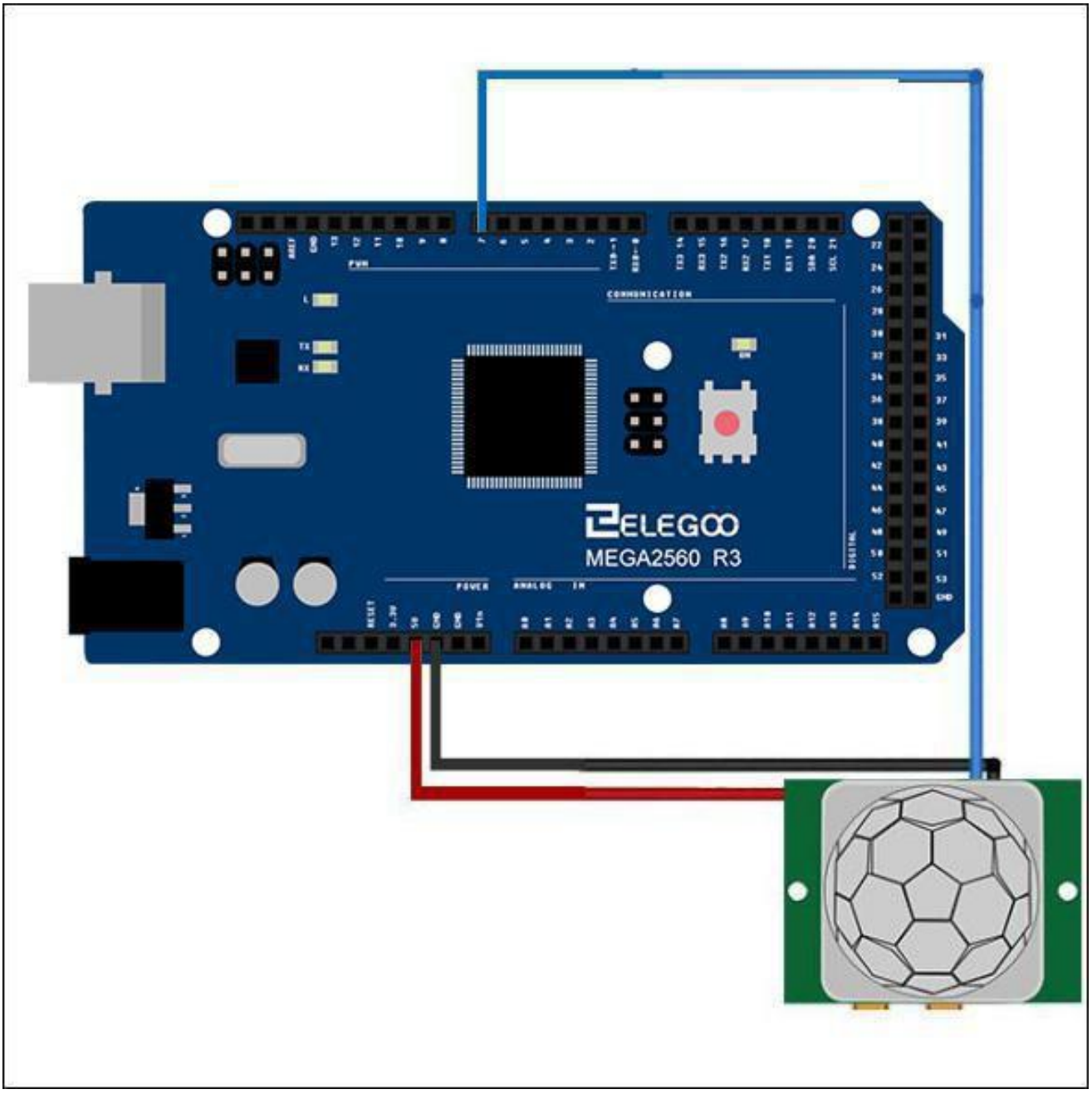
Schaltplan




3V3
5V
VIN
RESET
AREF
IOREF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
Arduino
Mega
2560
(Rev3)
1
2
VCC
I/O
HC SR501
VSS
NC
3
4

Verdrahtungsplan



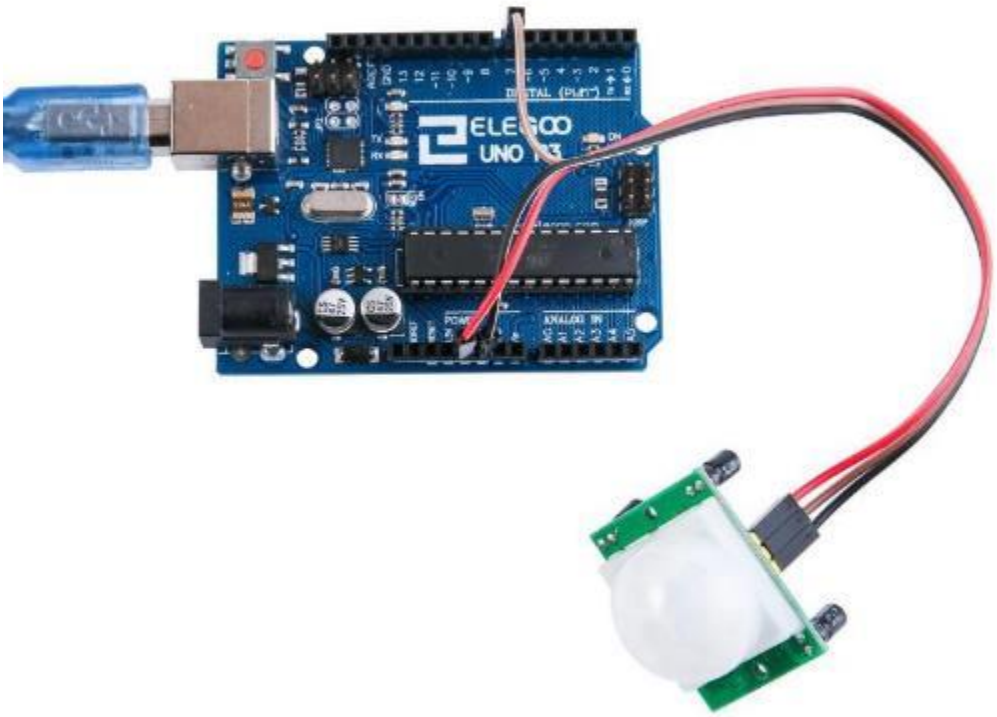

ELEGOO
MEGA2560 R3

Der Anschluss von PIR-Sensoren an einen Mikrocontroller ist denkbar einfach. Der PIR hat einen digitalen Ausgang, so dass Sie nur ‚high' oder ‚low' auslesen müssen. Es ist wahrscheinlich, dass Sie eine Nachtriggerung wünschen, also stellen Sie den Jumper in die H-Position! Den PIR bitte mit 5V versorgen und Masse an Masse anschließen. Dann verbinden Sie den Ausgang mit einem digitalen Pin. In diesem Beispiel verwenden wir Pin 7.

Code

Nach der Verdrahtung laden Sie bitte das Programm Lesson 31 HC-SR501 PIR Sensor auf den Arduino. Das Programm schaltet einfach die eingebaute Arduino-LED ein - die an Pin 13 angeschlossen ist - wenn eine Bewegung erkannt wird. Achten Sie darauf, dass der Sensor erst nach einer Minute startbereit ist. Dies müssen Sie beachten, wenn sie den Sensor in eigenen Applikationen verwenden.

Beispielbild




MADE
IN ITALY
ELEGOO
MEGA2560 R3

Im Folgenden finden Sie den Code und einige Erklärungen

```
// Die eingebaute LED ist Pin 13
int ledPin = 13;
// der PIR Sensor wird an Pin 7 angeschlossen
int inputPin = 7;
// speichert den vorherigen Zustand
int pirState = LOW;
int val = 0;

void setup() {
   pinMode(ledPin, OUTPUT);
   pinMode(inputPin, INPUT);
   Serial.begin(9600);
}

void loop(){
   // Einlesen des Wertes am Ausgang des PIR Sensors
   val = digitalRead(inputPin);
   if (val == HIGH) {
      // Falls der Sensor ‚high' ausgibt …
      // …LED einschalten
      digitalWrite(ledPin, HIGH);
      if (pirState == LOW) {
         // Wenn der letzte Status „Low" war…Meldung ausgeben
         Serial.println("Motion detected!");
         pirState = HIGH;
      }
   }
else
{
      // Falls der Sensor ‚low' ausgibt…LED ausschalten
```

```
        digitalWrite(ledPin, LOW);
        if (pirState == HIGH){
          // Wenn der letzte Status „High“ war…Meldung ausgeben
          Serial.println("Motion ended!");
          pirState = LOW;
        }
    }
}
```

Lektion 32 Wasserstandssensormodul

Überblick

In dieser Lektion lernen Sie, wie Sie ein Wasserstandssensormodul verwenden. Dieses Modul kann die Wassertiefe wahrnehmen und die Kernkomponente ist eine Verstärkerschaltung, die aus einem Transistor und mehreren Leiterbahnen besteht. Wenn sie ins Wasser gelegt werden, stellen diese Leitungen einen Widerstand dar, der sich mit der Änderung der Wassertiefe ändert. Dann wird der Widerstand in ein elektrisches Signal umgewandelt, und wir können die Änderung der Wassertiefe durch die ADC-Funktion des Arduino auswerten.

Benötigte Komponenten:

1x Elegoo Uno R3

1x Wasserstandssensormodul

3x F-M Drähte

Komponenteneinführung

Wasserstandssensor:

Ein Wasserstandssensor ist für die Wasserdetektion konzipiert und kann für die Erfassung von Niederschlägen, des Wasserspiegels und sogar von Lecks verwendet werden. Der Baustein besteht im Wesentlichen aus drei Teilen: einem elektronischen Bausteinverbinder, einem 1 MΩ Widerstand und mehreren Leitungen aus blanken, leitenden Drähten.

Dieser Sensor arbeitet mit einer Reihe von freiliegenden Leiterbahnen, die mit Masse verbunden sind. Zwischen diesen sind die Sensorleitungen verschachtelt.

Die Sensorleitungen haben einen schwachen Pull-Up-Widerstand von 1 MΩ. Der Widerstand zieht den

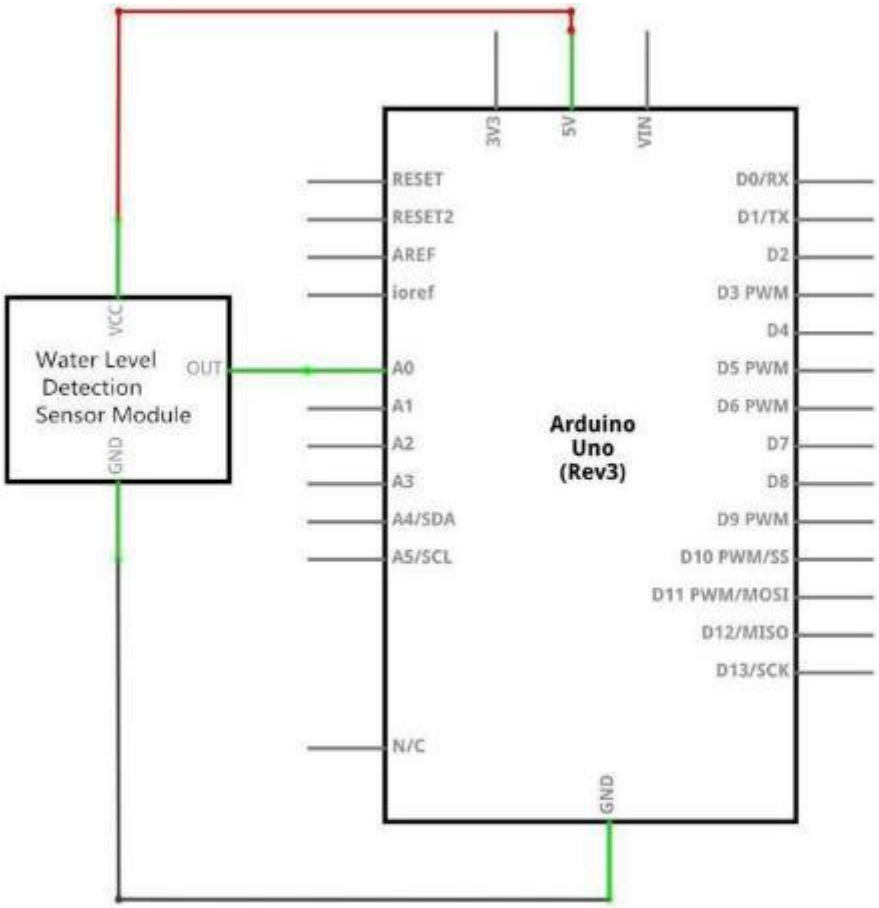
Spannungswert hoch, bis ein Wassertropfen die Sensorleitung mit einer Leitung auf Masse kurzschließt. Sie können die analogen Pins des Arduino verwenden, um die Menge an wasserinduziertem Kontakt zu erkennen. Je tiefer der Sensor in Wasser eingetaucht ist, desto geringer ist der Widerstand. Natürlich ist die Leitfähigkeit des Wassers von mehreren Faktoren abhängig. Der absolute Wert, den der Sensor liefert hat also wenig Aussagekraft. Allerdings geben die relativen Unterschiede im selben Wasserbehälter durchaus genaue Rückschlüsse auf die Wassertiefe.

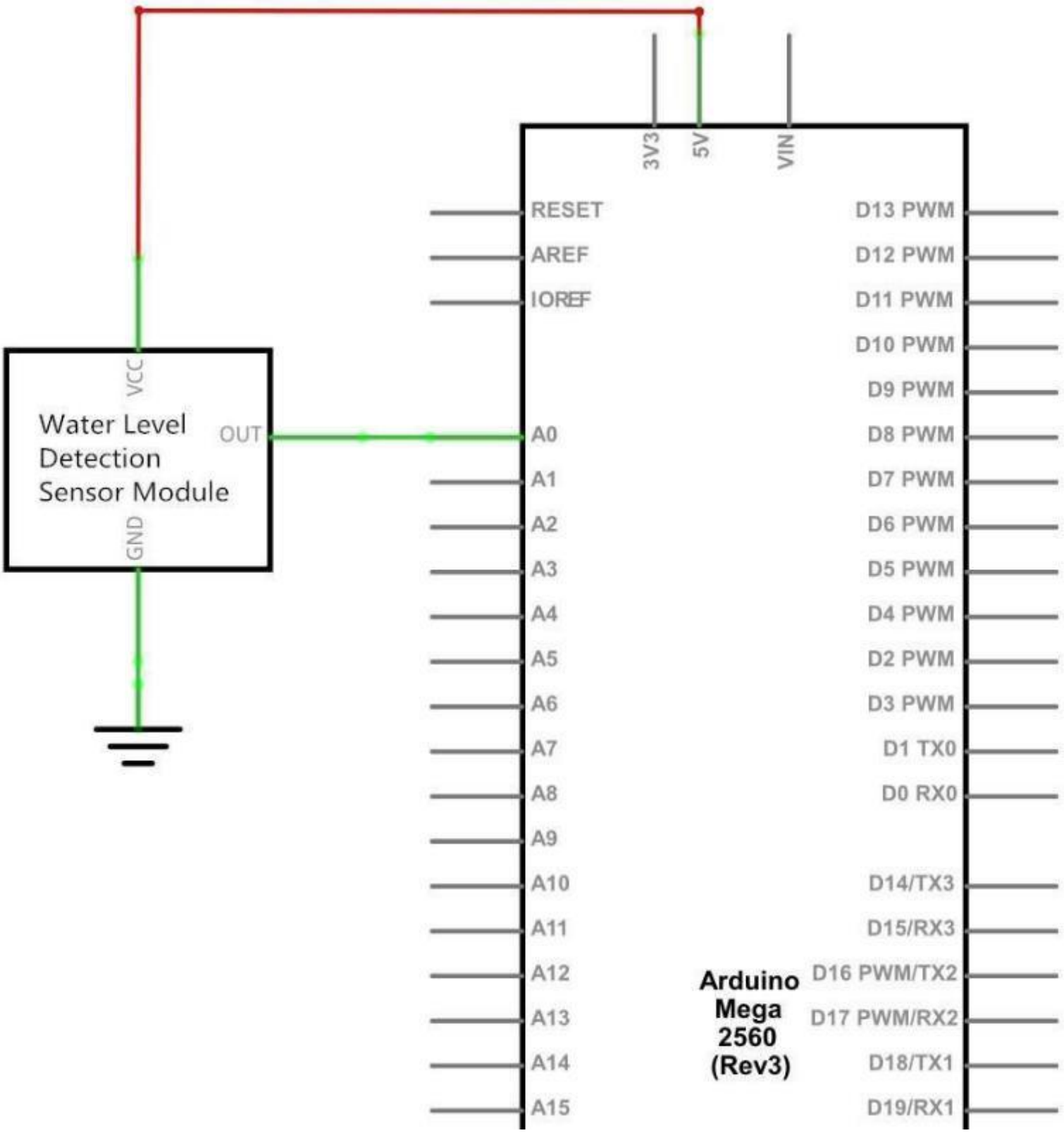
Es hat einen niedrigen Stromverbrauch und eine hohe Empfindlichkeit.

1. Spannungsversorung: 5V
2. Stromverbrauchim Betrieb: <20mA
3. Schnittstelle: Analog
4. Detektionsfläche: 40mm×16mm
5. Arbeitstemperatur: 10℃ bis 30℃
6. Ausgangsspannungssignal: 0 bis 4.2V

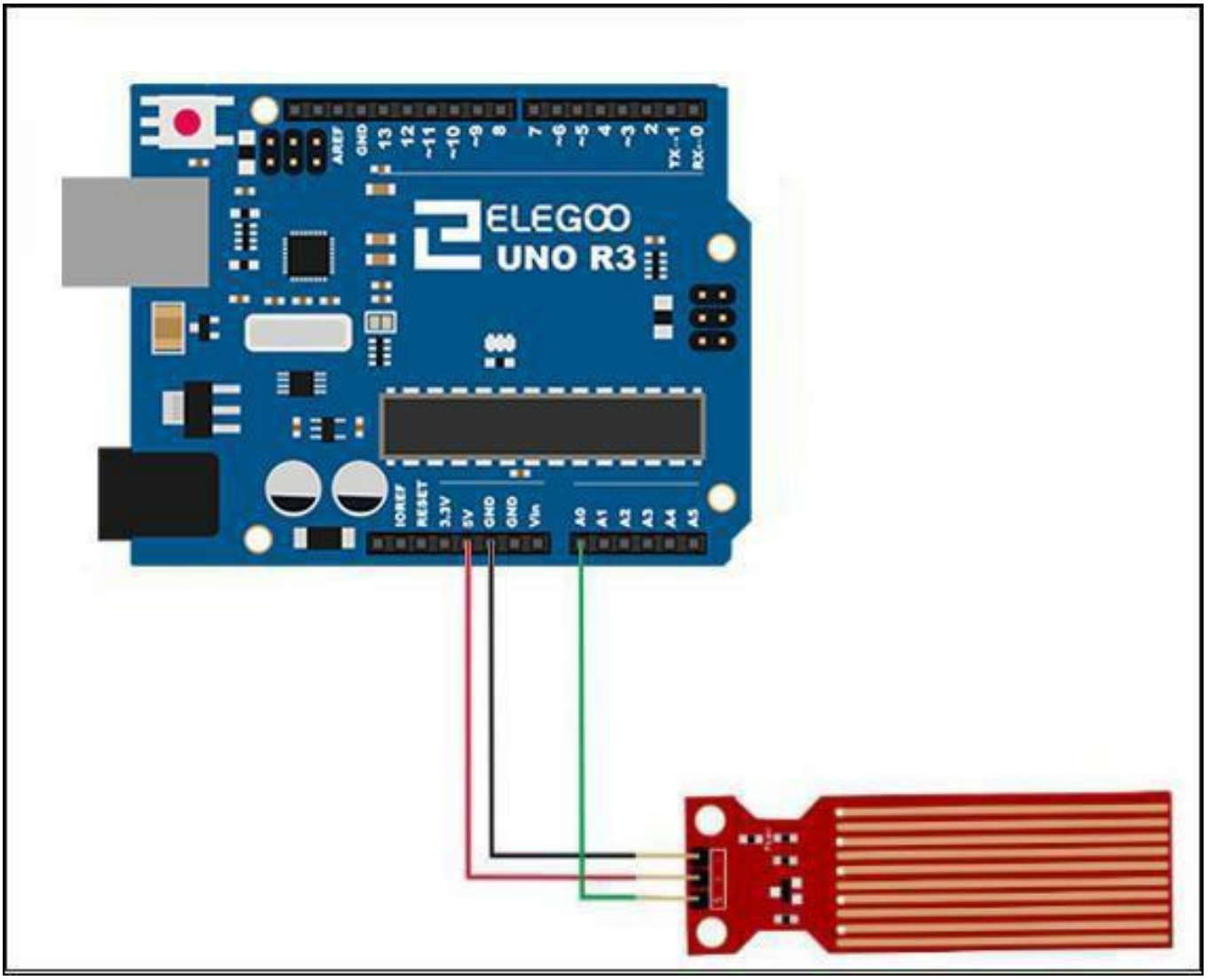
Verbindung

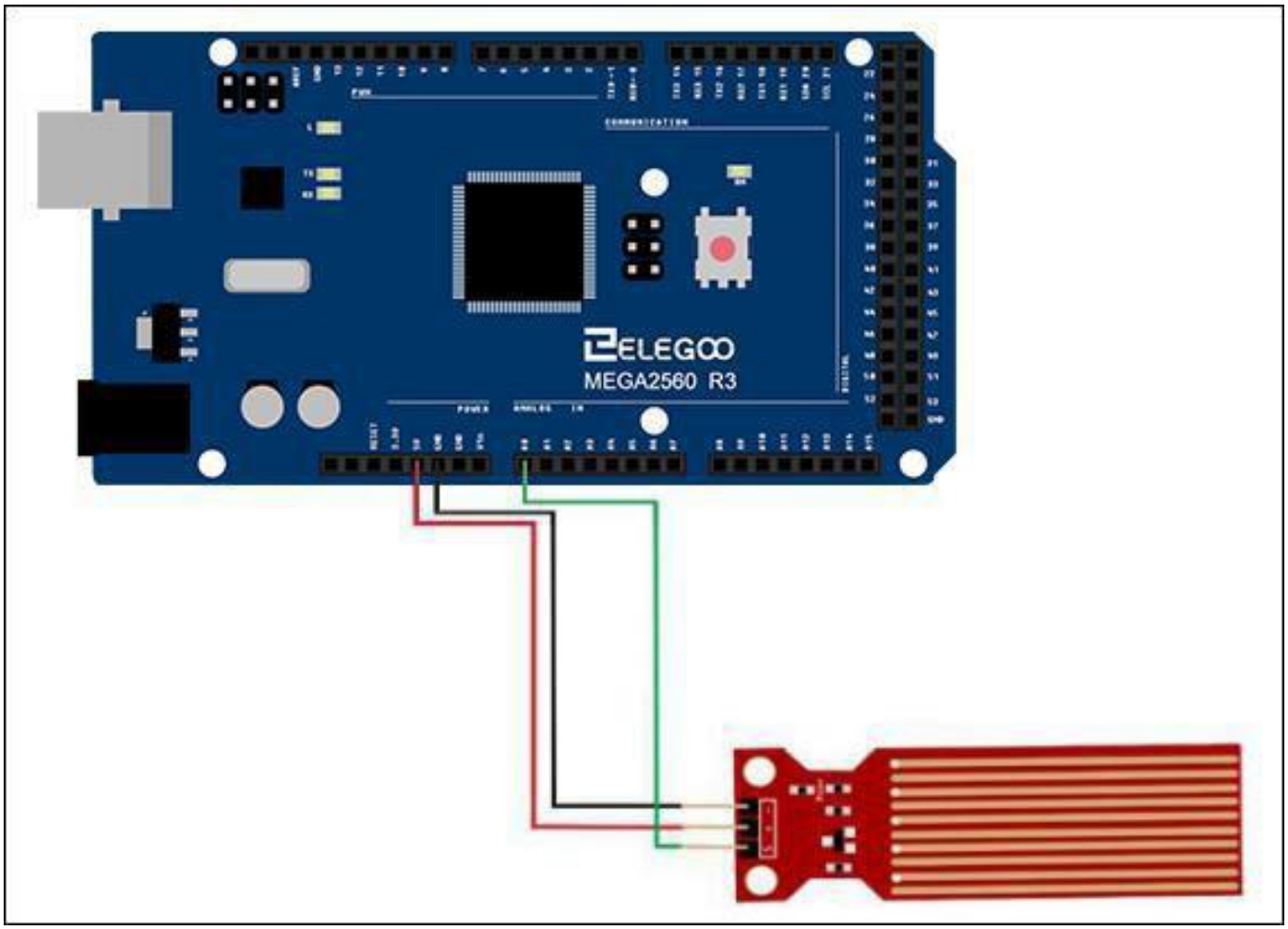
Schaltplan




3V3
5V
VIN
RESET
AREF
IOREF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
Arduino Mega 2560 (Rev3)
VCC
Water Level Detection Sensor Module
OUT
GND

Verdrahtungsplan

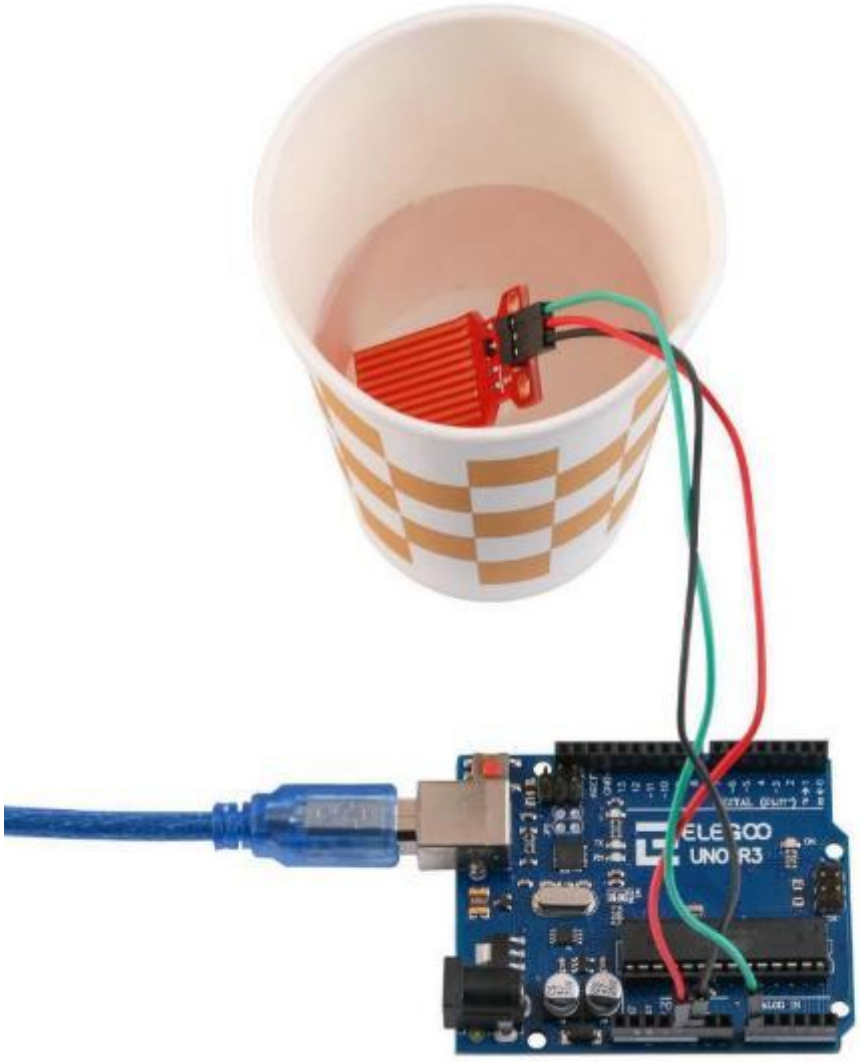


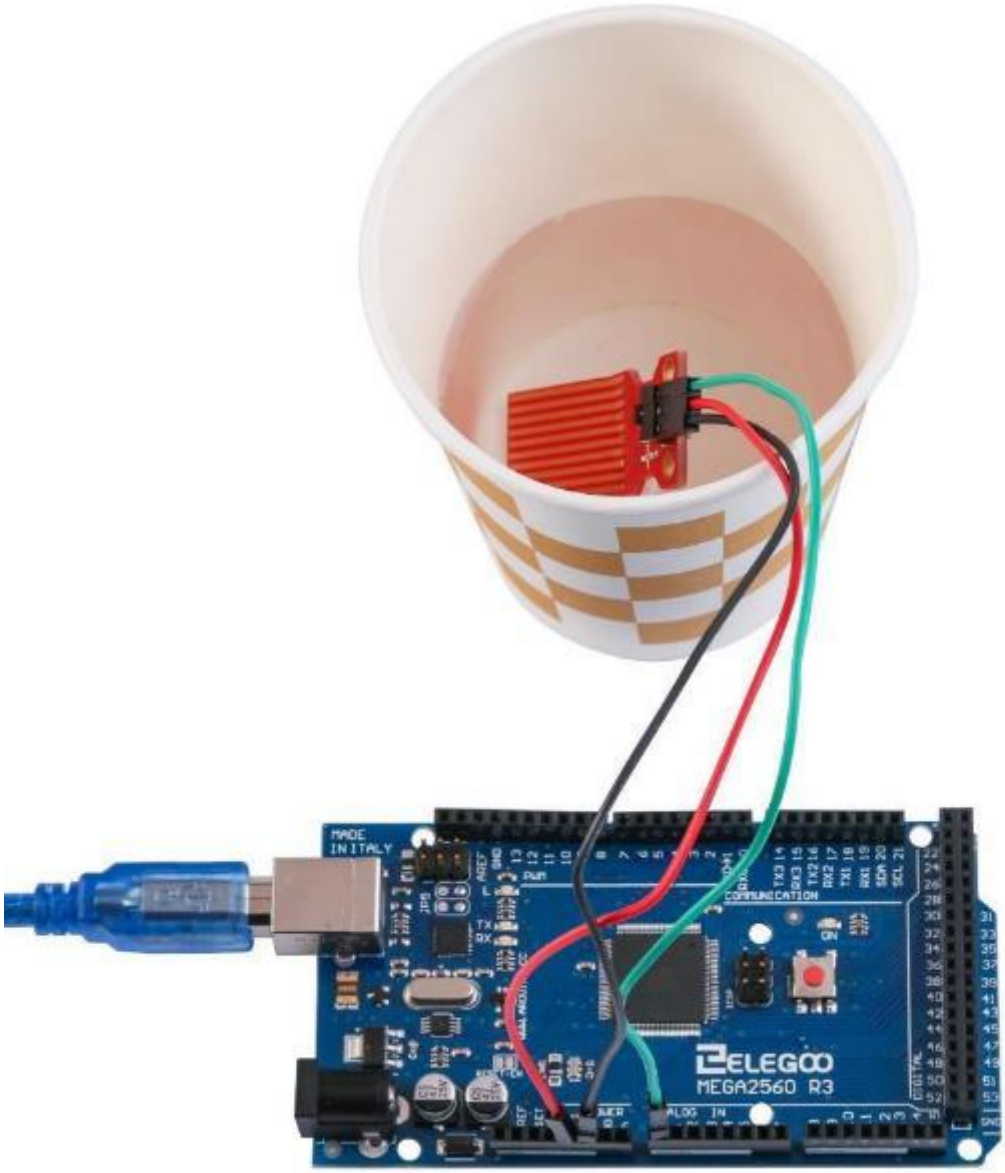

ELEGOO
MEGA2560 R3

Code

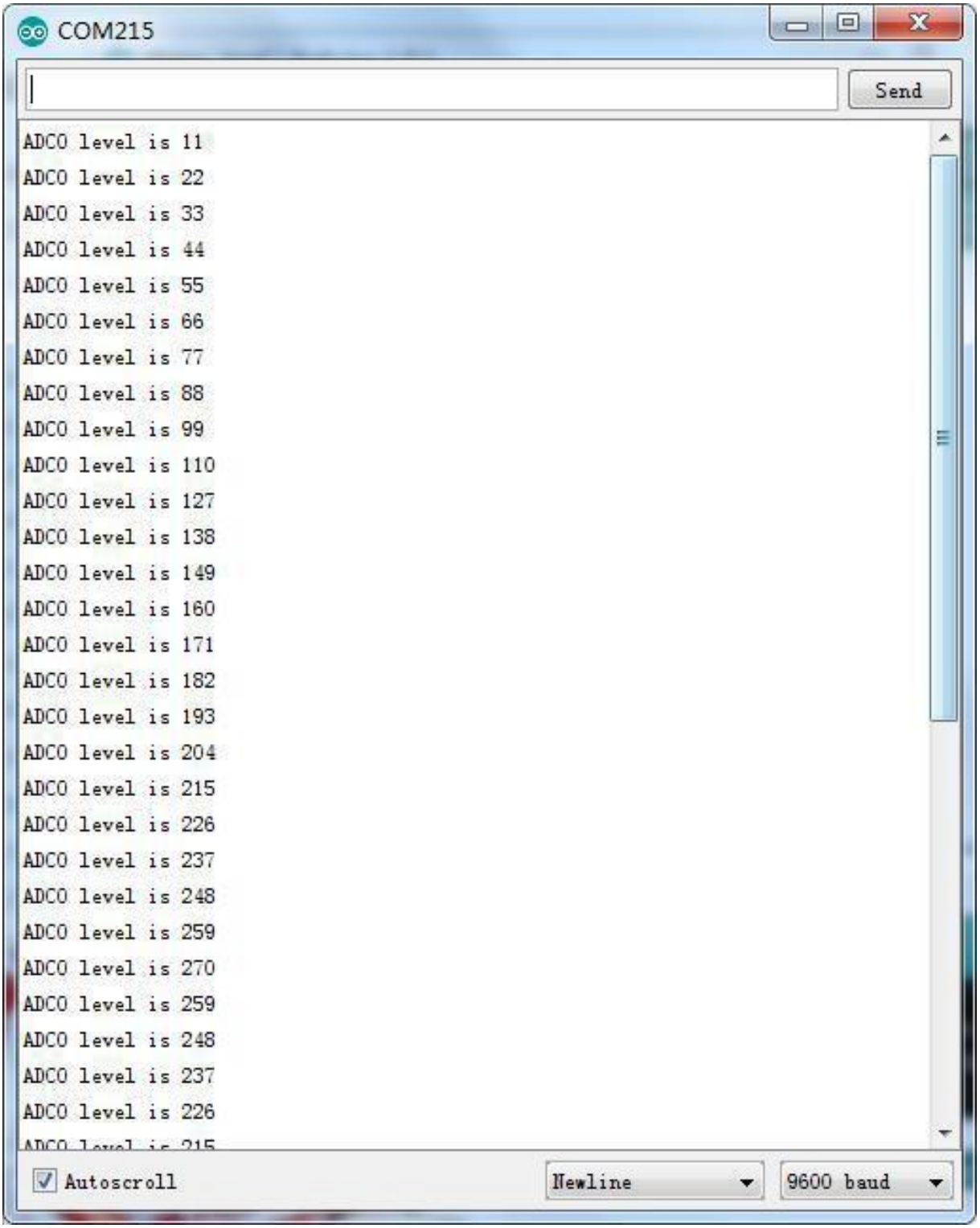
Nach der Verdrahtung laden Sie bitte das Programm „Lesson 32 Water Level Detection Sensor Module auf den Arduino.“

Beispielbild




MADE
IN ITALY
ELEGOO
MEGA2560 R3

Öffnen Sie nun den seriellen Monitor. Die Ausgabe sollte in etwa wie unten aussehen.



Im Folgenden finden Sie den Code und einige Erklärungen

```
int adc_id = 0;
int HistoryValue = 0;
char printBuffer[128];
void setup()
{
Serial.begin(9600);
}

void loop()
{
    int value = analogRead(adc_id); // auslesen des Wertes vom Sensormodul
    if(((HistoryValue>=value) && ((HistoryValue - value) > 10)) || ((HistoryValue<value) &&
        ((value - HistoryValue) > 10)))
        //Der Wert wird nur ausgegeben, wenn er sich mindestens um 10 unterscheidet vom vorherigen Wert. Bei
        //kleinsten Wertänderungen kann es sich auch um Messungenauigkeiten handeln
    {
        sprintf(printBuffer,"ADC%d level is %d\n",adc_id, value);
        //Die Funktion sprintf formatiert einen String. Sie braucht als ersten Parameter einen freien
        //Speicherbereich. In unserem Fall hat besteht er aus 128 Bytes (siehe oben)
        Serial.print(printBuffer);
        HistoryValue = value;
    }
}
```

Lektion 33 Echtzeituhrmodul

Überblick

In dieser Lektion lernen Sie, wie Sie das DS3231-- Echtzeituhrmodul verwenden, das Jahr, Monat, Tag, Stunde, Minute, Sekunde und Woche anzeigt. Das Modul hat eine Backupbatterie und kann auch ohne Verbindung zum Arduino weiterlaufen.

Benötigte Komponenten:

1x Elegoo Uno R3

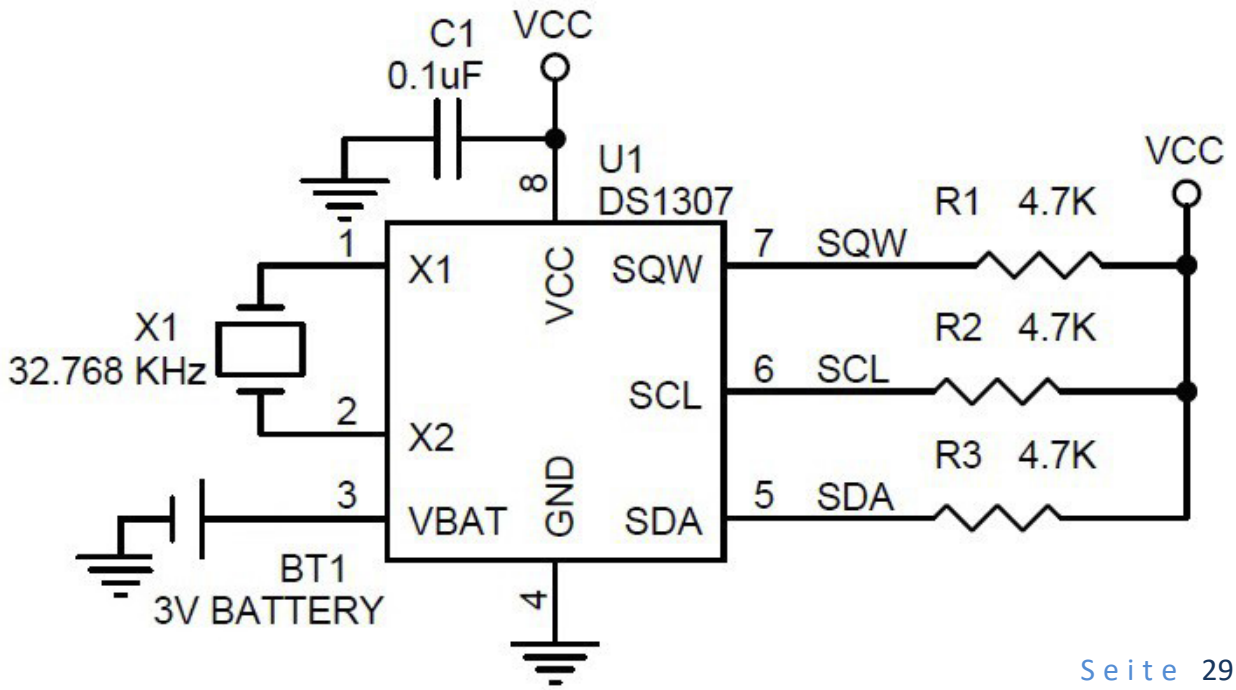
1x DS3231 RTC Modul

4x F-M Drähte

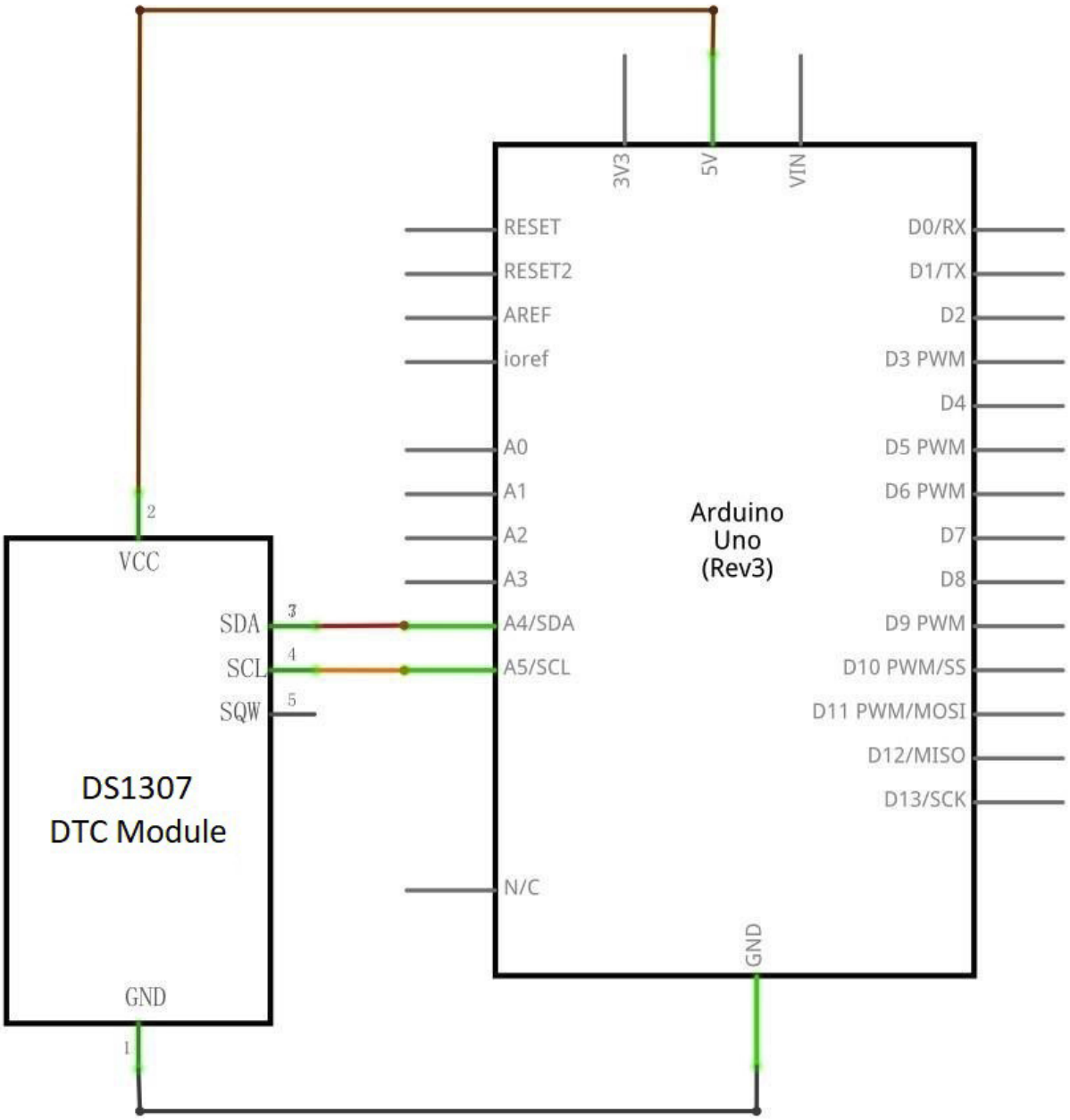
Komponenteneinführung

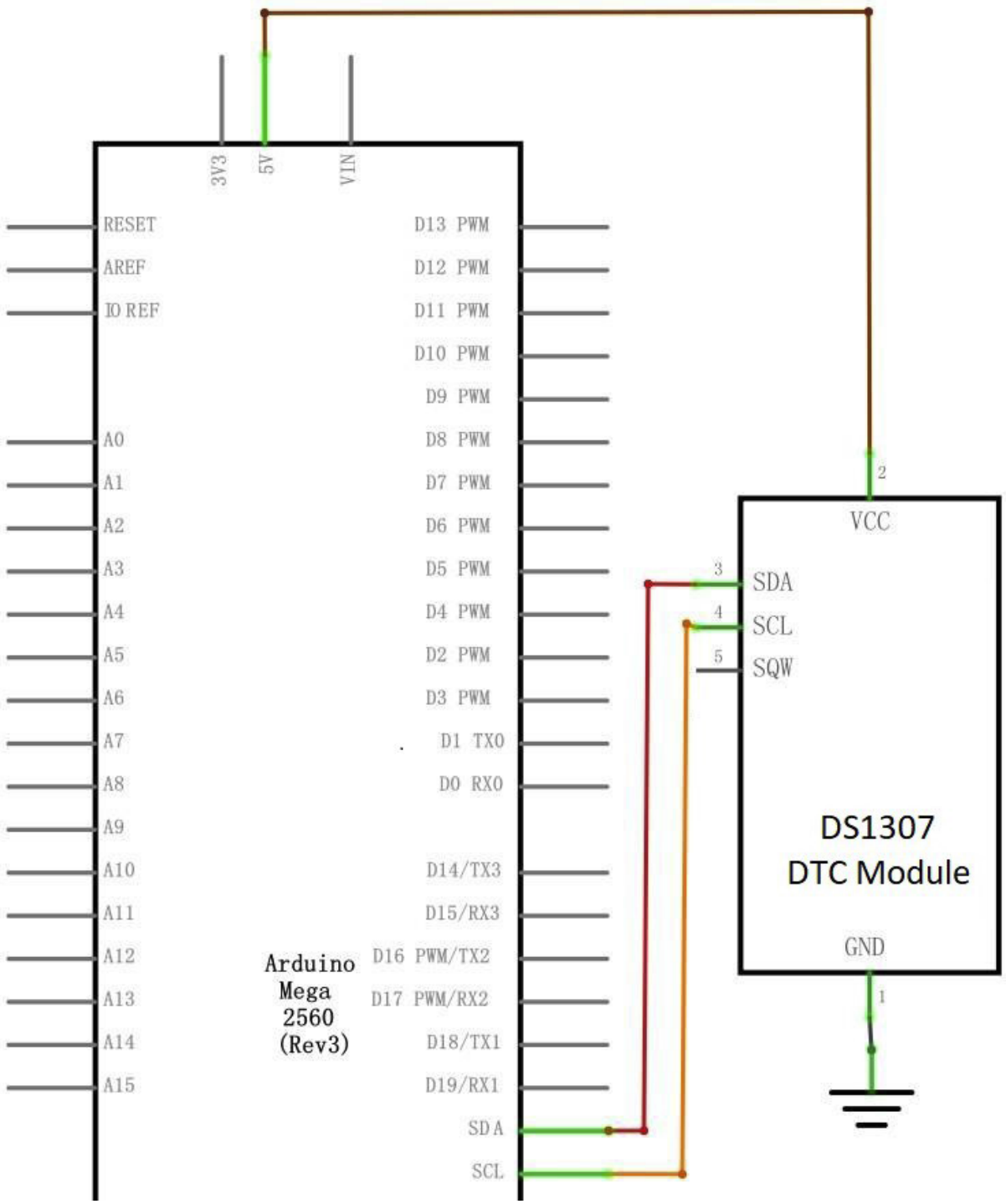
DS3231

Der DS3231 ist ein einfacher Uhrzeit-Chip. Er hat eine integrierte Batterie, so dass die Uhr auch im ausgeschalteten Zustand die Zeit beibehalten kann. Nachfolgend finden Sie den Schaltplan des Moduls.

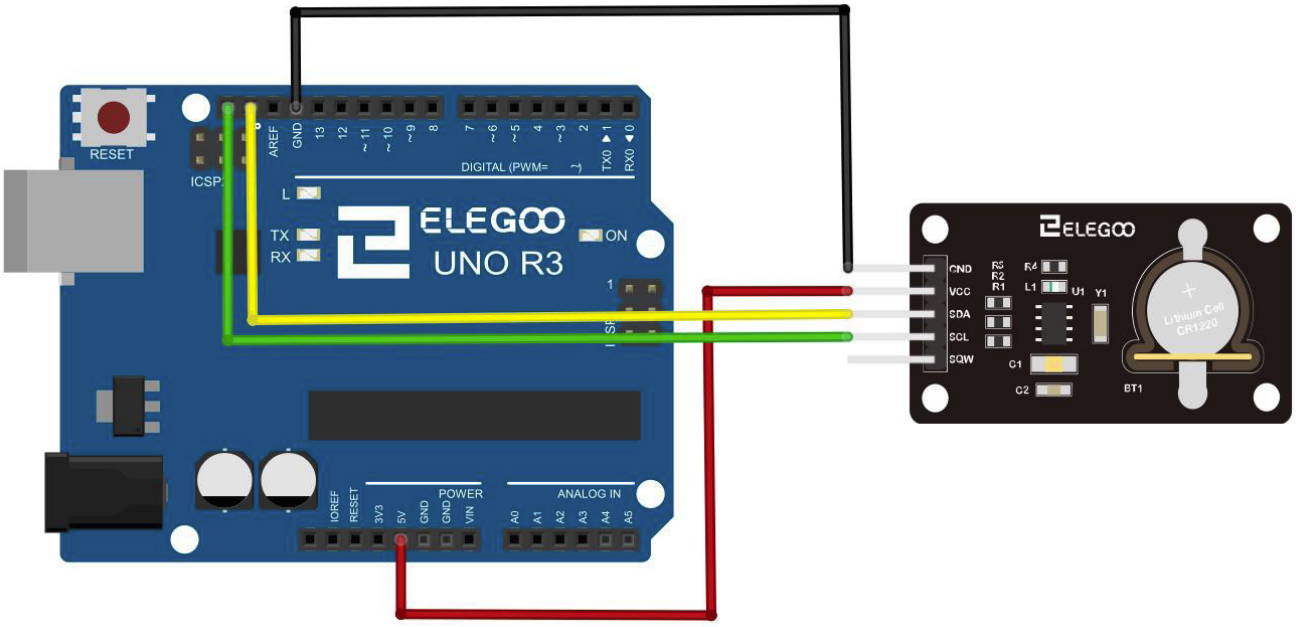


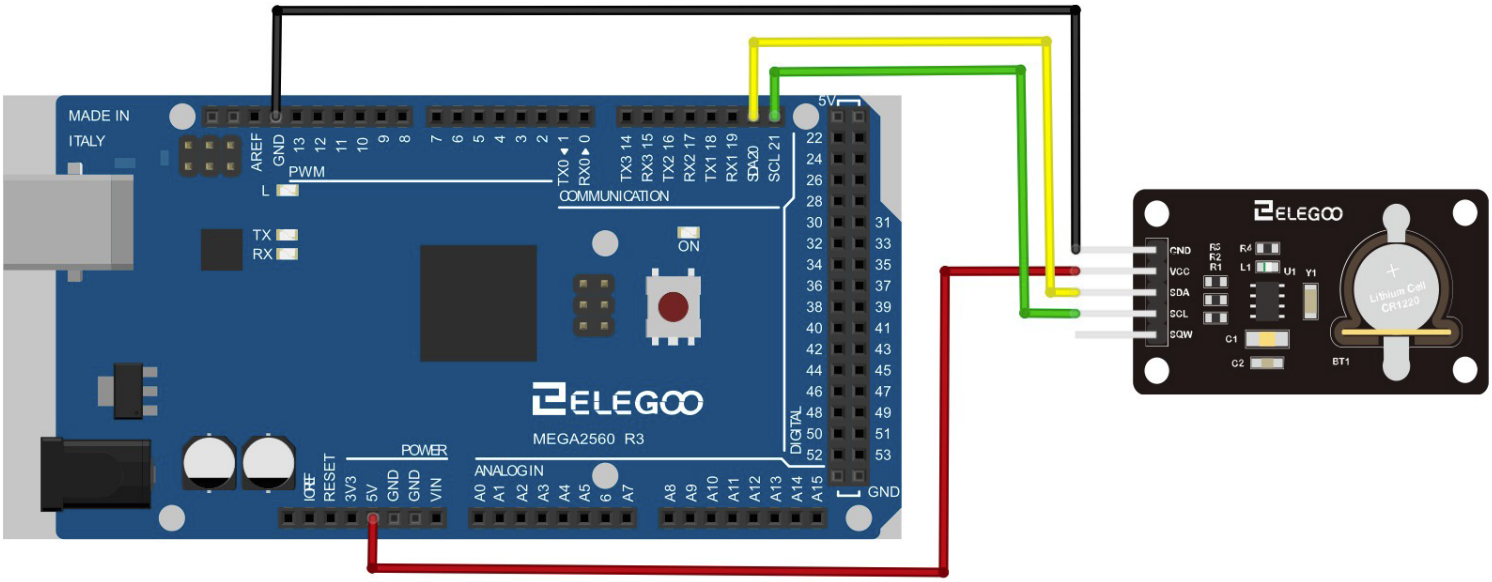
Verbindung
Schaltplan




3V3
5V
VIN
RESET
AREF
IO REF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
SCL
Arduino
Mega
2560
(Rev3)
2
VCC
3
SDA
4
SCL
5
SQW
DS1307
DTC Module
GND
1

Verdrahtungsplan




MADE IN
ITALY
AREF
GND
PWM
COMMUNICATION
TX
RX
ON
ELEGOO
MEGA2560 R3
POWER
ANALOG IN
DIGITAL
GND
5V
SCL 21
SDA 20
GND
VCC
SDA
SCL
SQW
Lithium Cell
CR1220
BT1

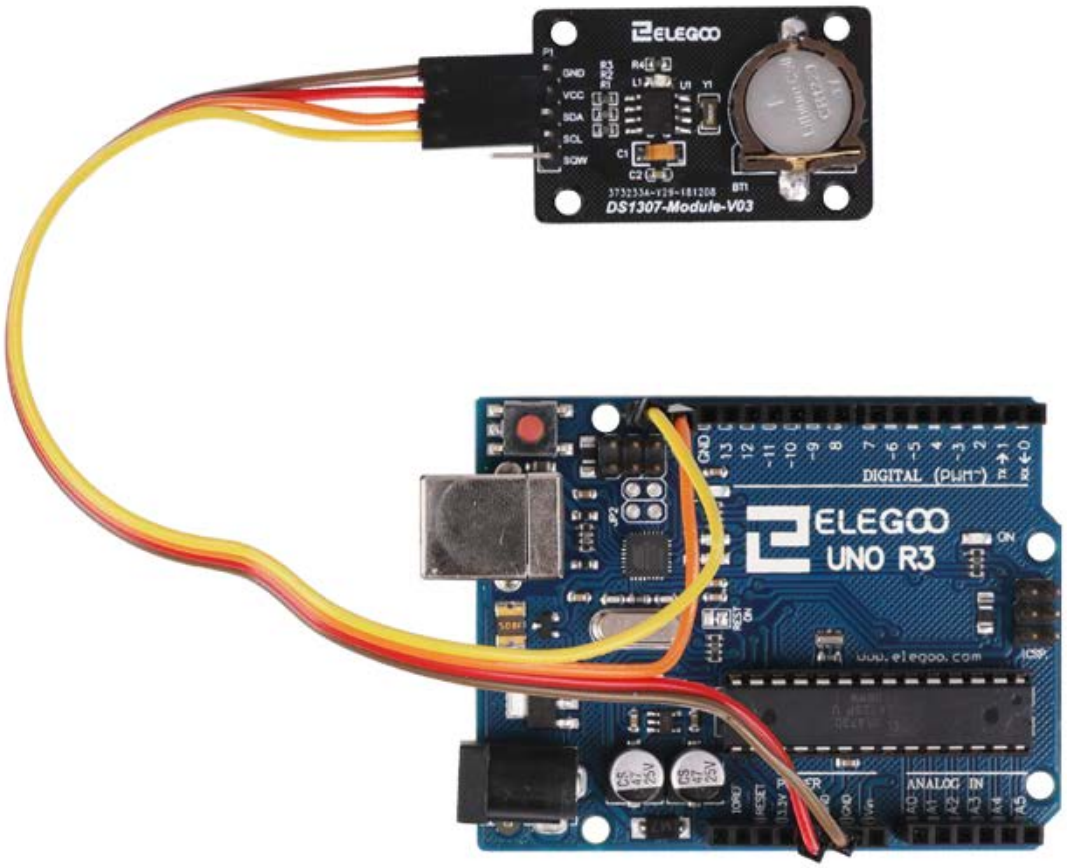
Erstellen Sie die Verbindung nach dem vorherigen Bild.

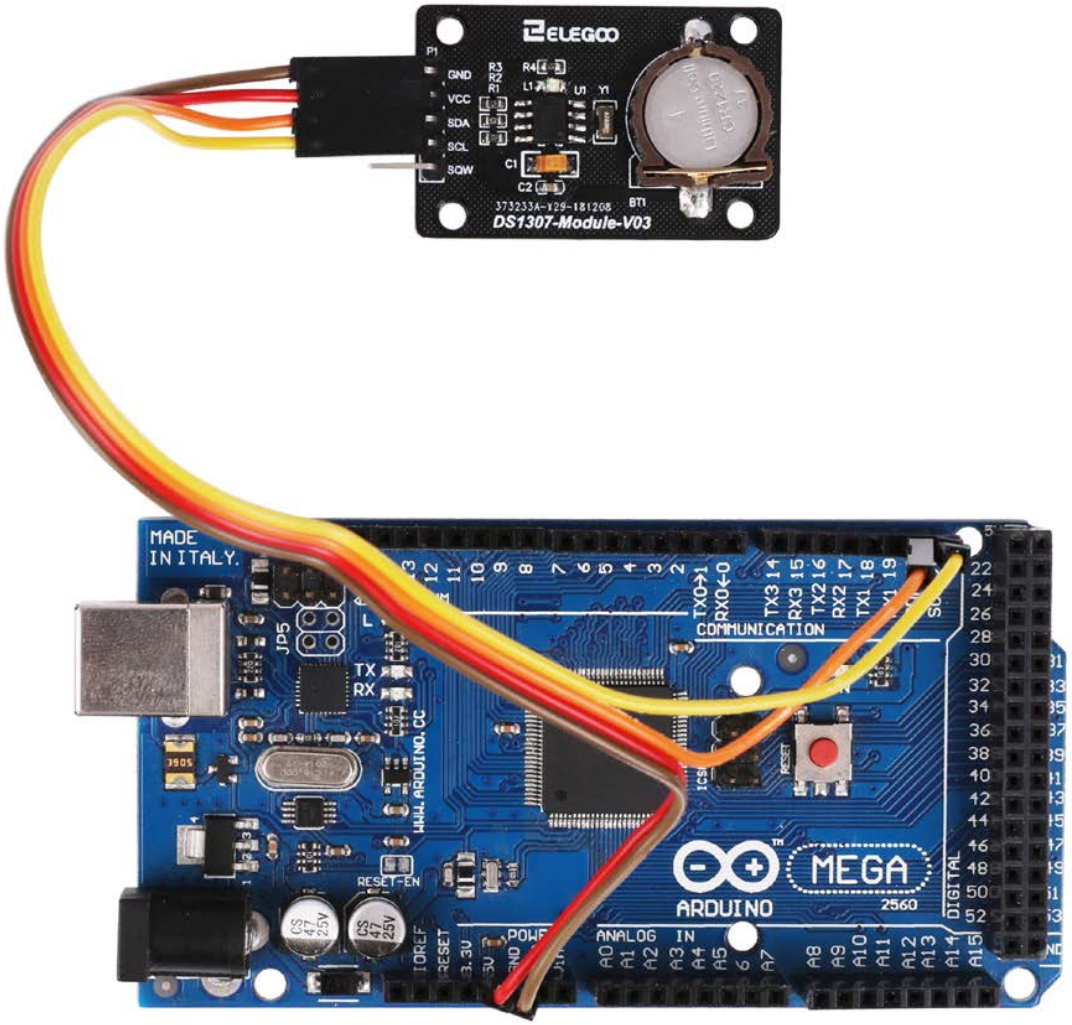
Achtung: Ignorieren Sie die 32K und SQW pins; sie werden in dieser Lektion nicht benutzt.

Code

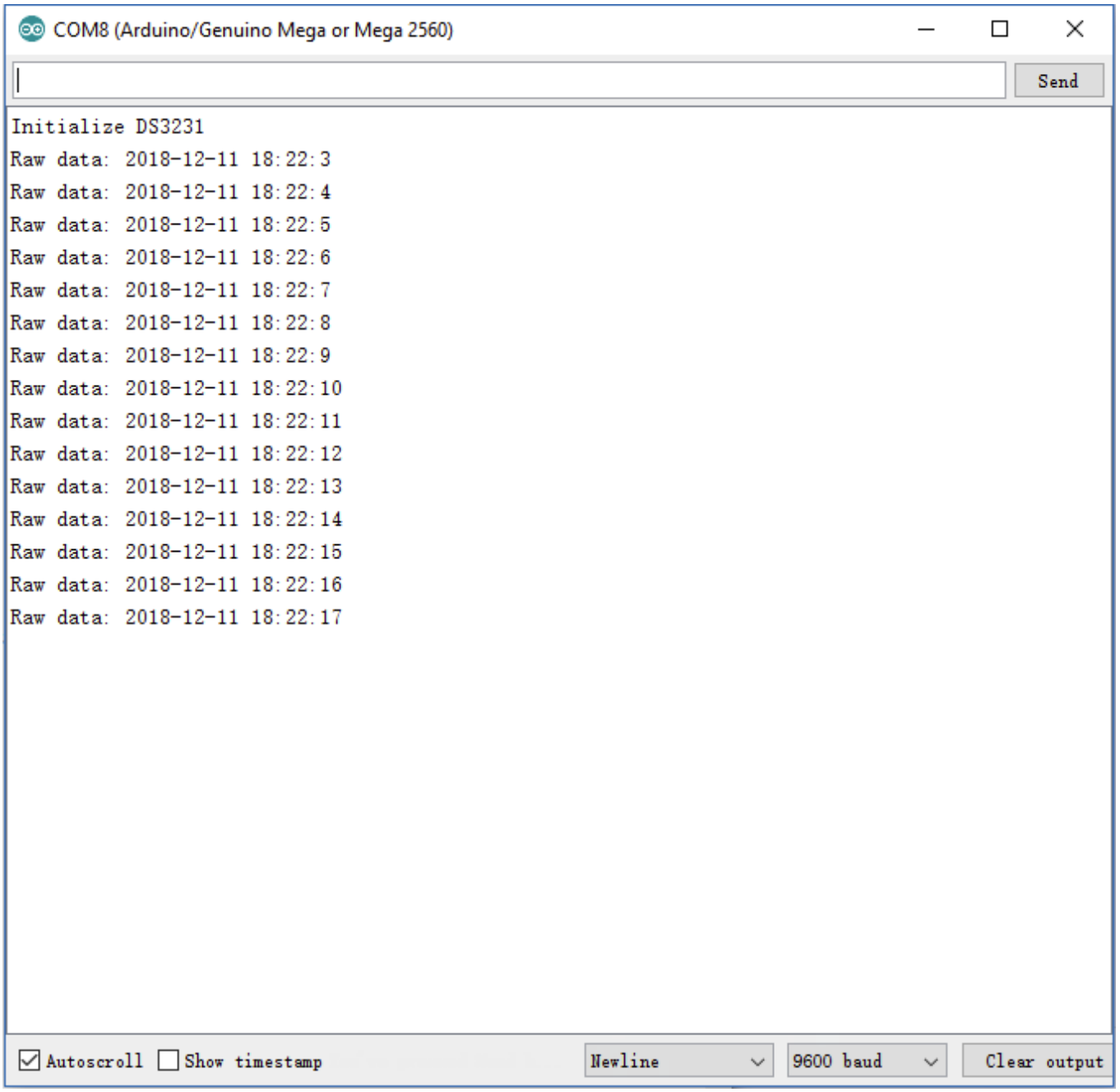
Nach der Verdrahtung laden Sie bitte das Programm „Lesson33 Real Time Clock Module“ auf den Arduino. Der Code benötigt die DS3231 Bibliothek. Stellen Sie sicher, dass diese installiert ist oder installieren Sie sie.

Beispielbild




ELEGOO
GND
VCC
SDA
SCL
SQW
DS1307-Module-V03
MADE IN ITALY.
COMMUNICATION
ARDUINO
MEGA
2560
ANALOG IN
DIGITAL

Öffnen Sie den seriellen Monitor. Er sollte in etwa so aussehen wie auf dem nachfolgenden Bild.



Im Folgenden finden Sie den Code und einige Erklärungen

```
#include <Wire.h>
#include <DS3231.h>
DS3231 clock;
RTCDateTime dt;
void setup()
{ Serial.begin(9600);
   // Initialisieren des DS3231
   Serial.println("Initialize DS3231");
   clock.begin();
   // Setzen der aktuellen Uhrzeit…das Modul braucht eine Startzeit und zählt dann weiter
   clock.setDateTime(__DATE__, __TIME__);
}
void loop()
{    dt = clock.getDateTime();
   // Ausgeben der Zeitwerte
   Serial.print("Raw data: ");
   Serial.print(dt.year);      Serial.print("-");
   Serial.print(dt.month);    Serial.print("-");
   Serial.print(dt.day);       Serial.print(" ");
   Serial.print(dt.hour);      Serial.print(":");
   Serial.print(dt.minute); Serial.print(":");
   Serial.print(dt.second); Serial.println("");
   delay(1000);
}
```

Lektion 34 Tastaturmodul

Überblick

In diesem Projekt geht es darum, eine Tastatur mit einem UNO R3 Board zu verbinden, damit der UNO R3 die vom Benutzer gedrückten Tasten lesen/auswerten kann.
Tastaturen werden in allen Arten von Geräten verwendet, einschließlich Handys, Faxgeräten, Mikrowellen, Öfen, Türschlössern, usw. Sie sind praktisch überall. Millionen von elektronischen Geräten nutzen sie für Benutzereingaben.
Daher ist es sehr nützlich zu wissen, wie man eine Tastatur an einen Mikrocontroller wie einen UNO R3 anschließt, um viele verschiedene Arten von kommerziellen Produkten bauen zu können.

Wenn alles richtig angeschlossen und programmiert ist und eine Taste gedrückt wird, erscheint sie am seriellen Monitor Ihres Computers. Der Einfachheit halber beginnen wir damit, die gedrückte Taste (bzw. das Symbol der Taste) auf dem Computer anzuzeigen.
Für dieses Projekt verwenden wir eine Matrixtastatur. Dies ist eine Tastatur, die einem Kodierungsschema folgt, das es erlaubt, viel weniger Ausgangspins zu haben, als es Tasten gibt. Zum Beispiel hat die Matrix-Tastatur, die wir verwenden, 16 Tasten (0-9, A-D, *, #), aber nur 8 Ausgangspins. Bei einer linearen Tastatur müssten 17 Ausgangspins (einer für jede Taste und ein Masse-Pin) vorhanden sein, um zu funktionieren. Das Matrix-Codierungsschema ermöglicht weniger Ausgangspins und damit wesentlich weniger Verbindungen, die für das Funktionieren der Tastatur notwendig sind. Auf diese Weise sind sie effizienter als lineare Tastaturen, da sie weniger Verbindungen nach außen haben.

Benötigte Komponenten:

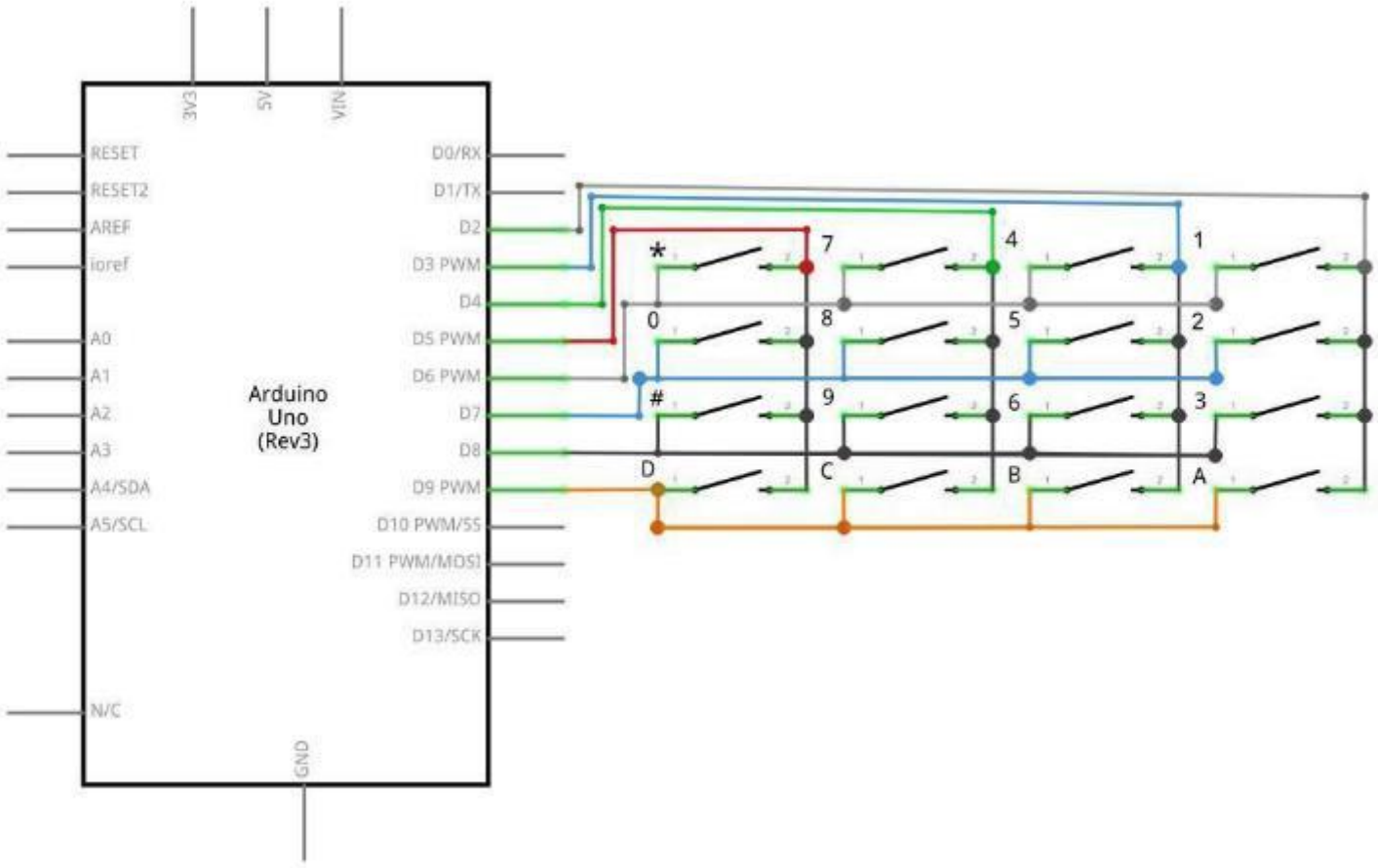
1 x Elegoo Uno R3

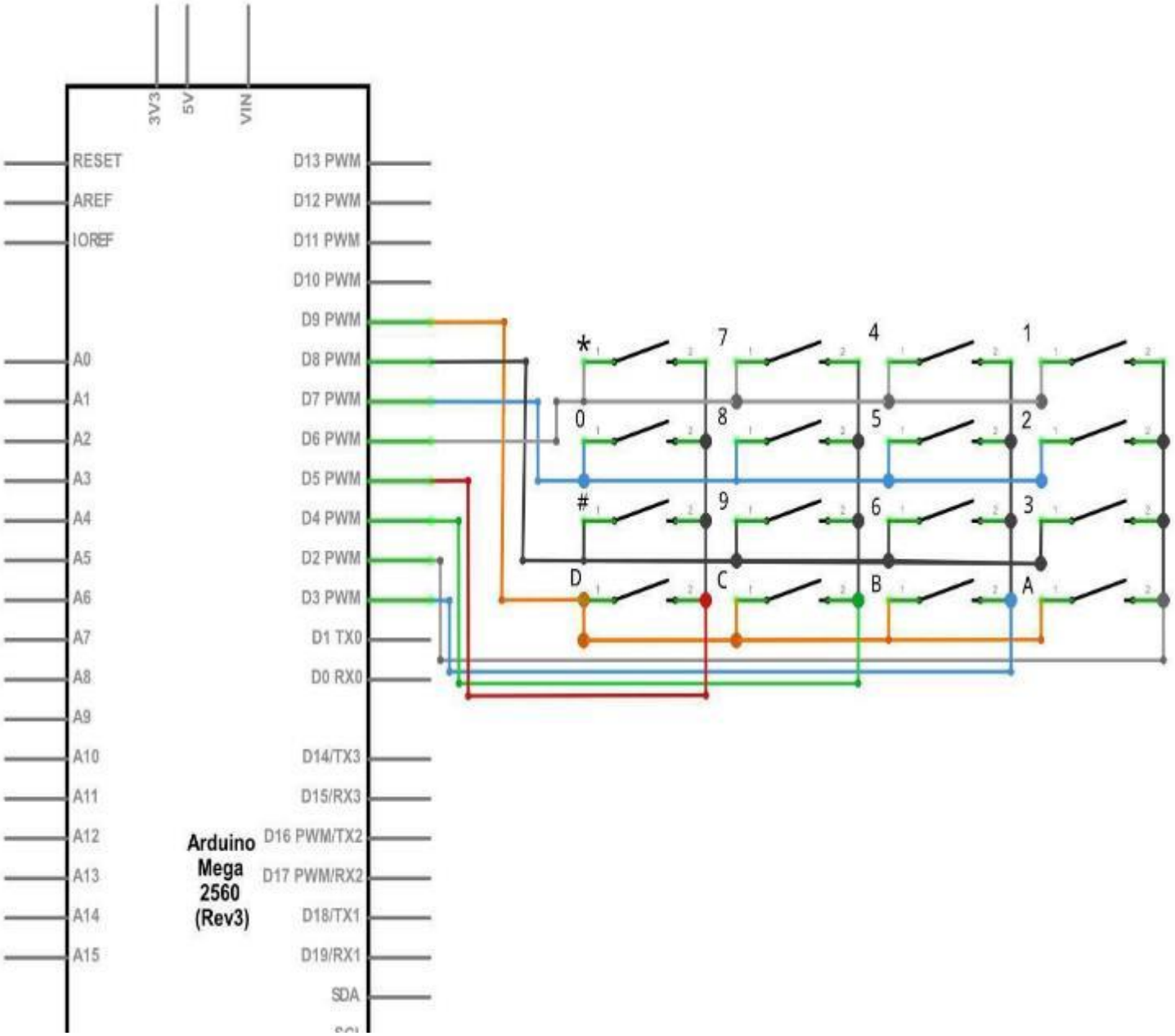
1 x Tastatur

8 x M-M Drähte (männlich zu männlich)

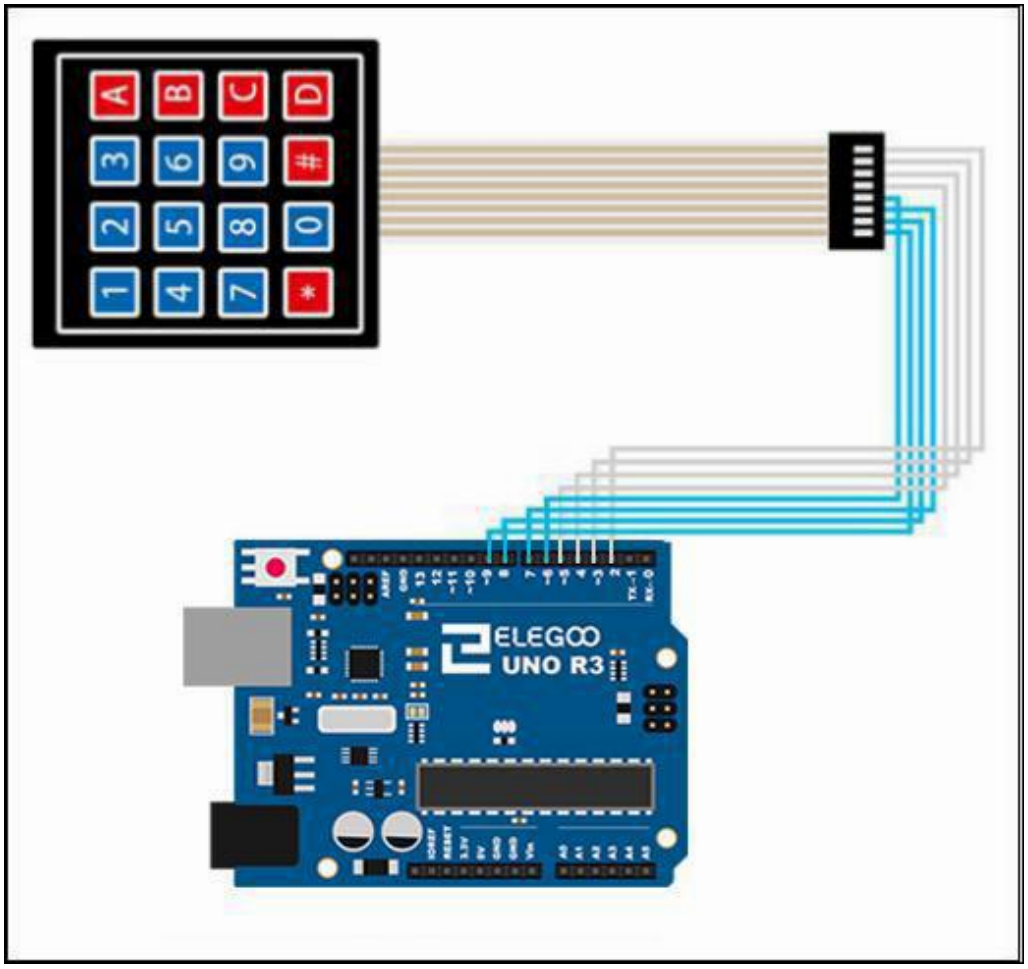
Verbindung

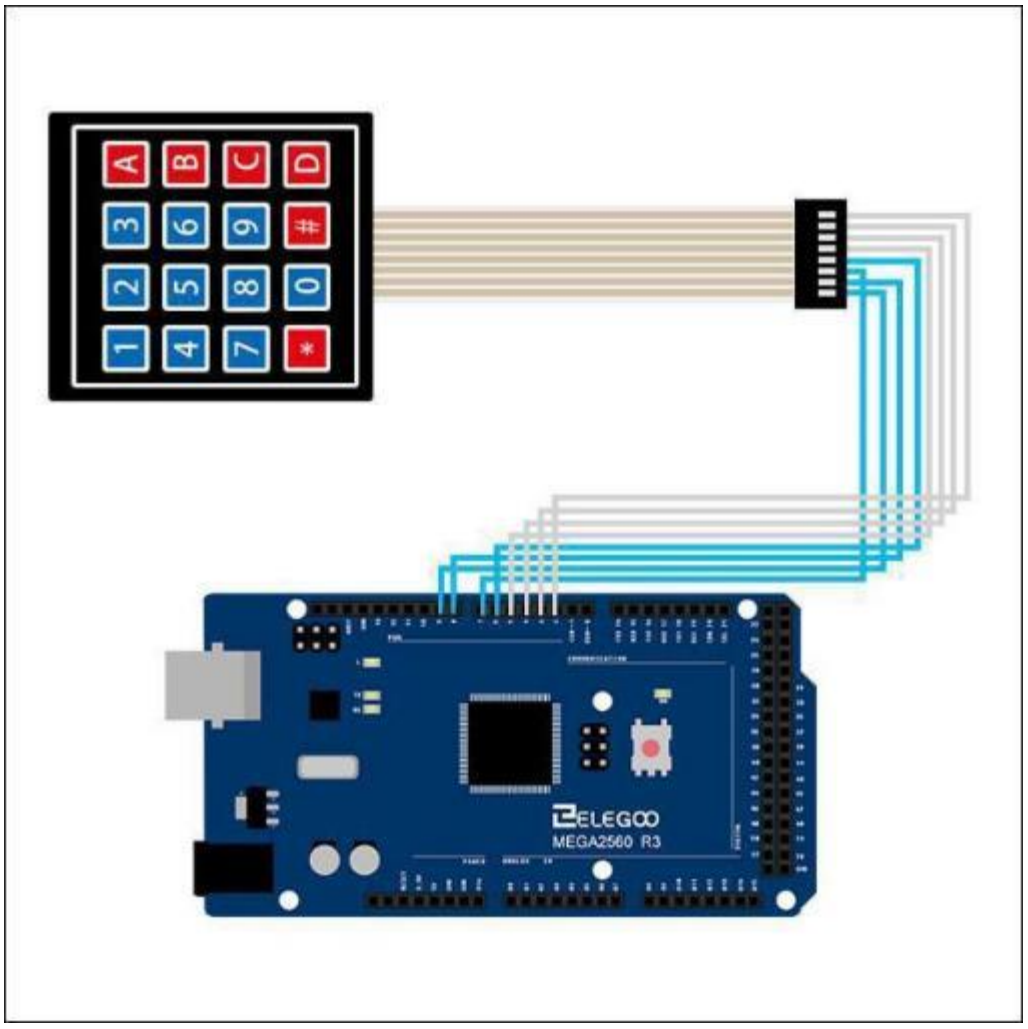
Schaltplan




3V3
5V
VIN
RESET
AREF
IOREF
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D13 PWM
D12 PWM
D11 PWM
D10 PWM
D9 PWM
D8 PWM
D7 PWM
D6 PWM
D5 PWM
D4 PWM
D2 PWM
D3 PWM
D1 TX0
D0 RX0
D14/TX3
D15/RX3
D16 PWM/TX2
D17 PWM/RX2
D18/TX1
D19/RX1
SDA
Arduino
Mega
2560
(Rev3)
*
7
4
1
0
8
5
2
#
9
6
3
D
C
B
A

Verdrahtungsplan




A
B
C
D
3
6
9
#
2
5
8
0
1
4
7
*
ELEGOO
MEGA2560 R3

Wir verbinden die Tastatur PINs mit dem UNO R3 nach folgender Tabelle:

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

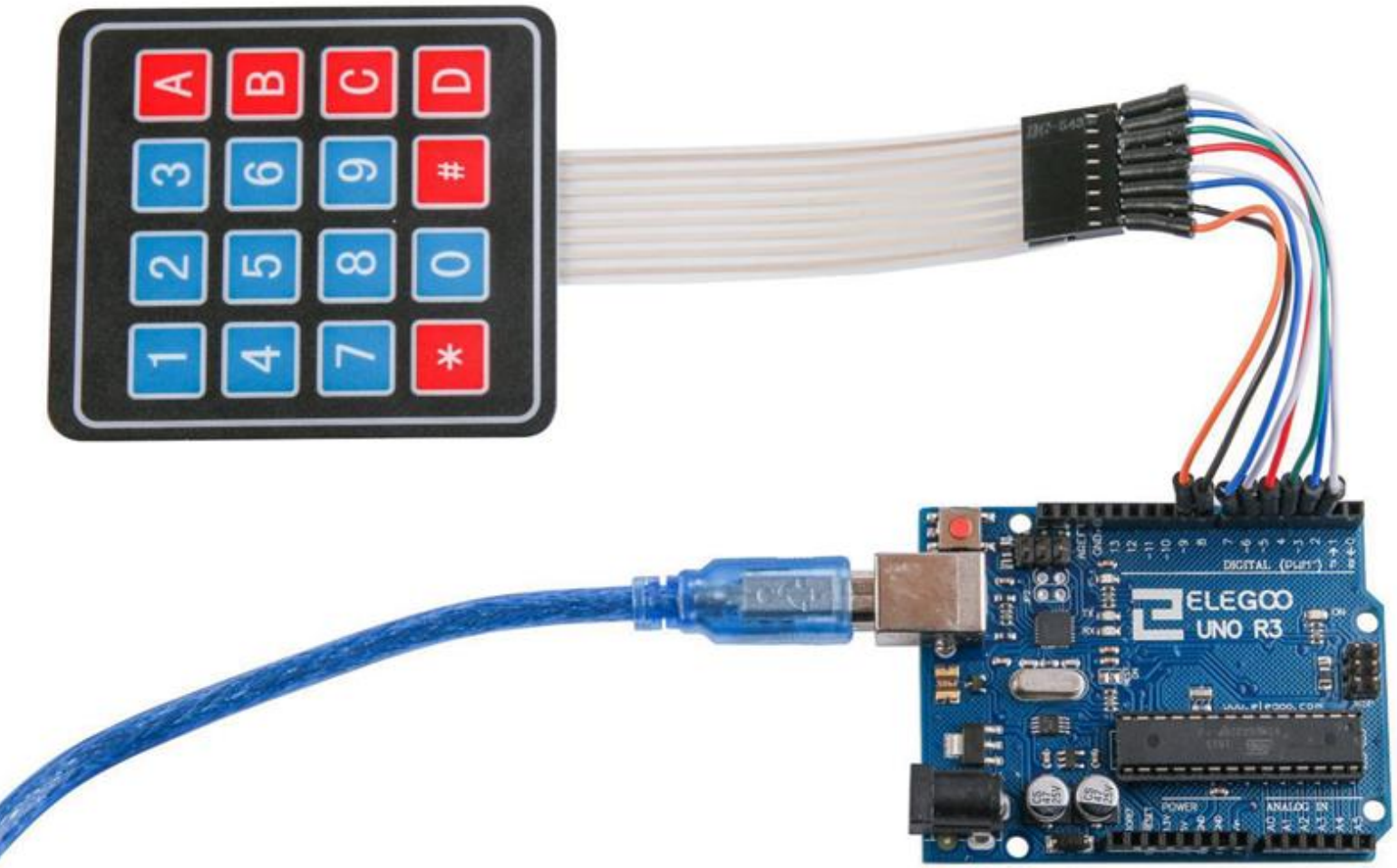
Code

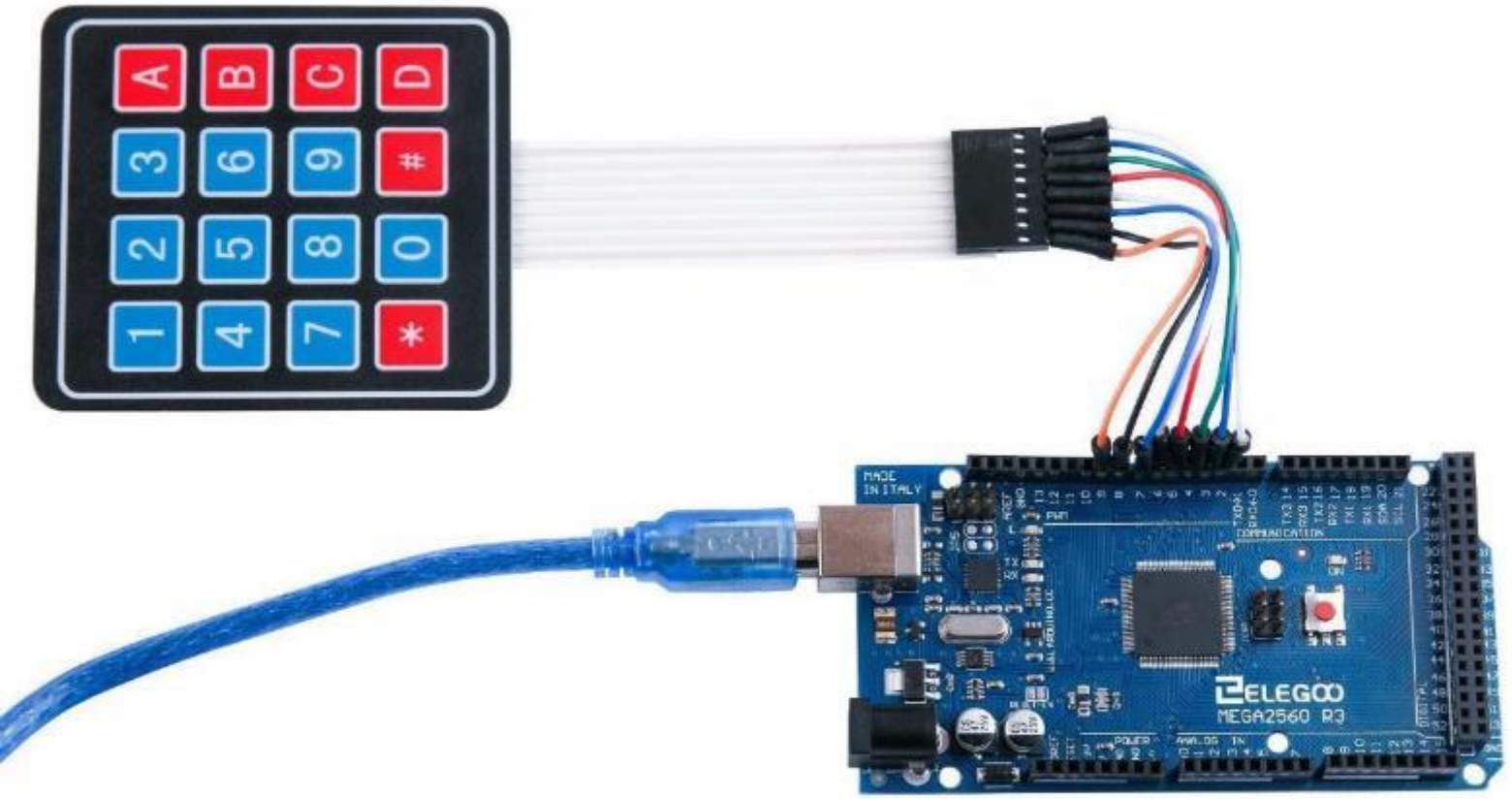
Nach der Verdrahtung öffnen Sie bitte das Programm im Code-Ordner - Lesson 34 Keypad Module und klicken Sie auf „Hochladen“, um das Programm hochzuladen. Siehe Lektion 2 für Details über das Hochladen von Programmen, falls Fehler auftreten.

Bitte stellen Sie sicher, dass Sie die < Keypad> Bibliothek installiert haben. Falls noch nicht geschehen, installieren Sie die Bibliothek. Andernfalls wird Ihr Code nicht funktionieren.

Einzelheiten zum Installieren von Bibliotheksdateien finden Sie in Lektion 1.

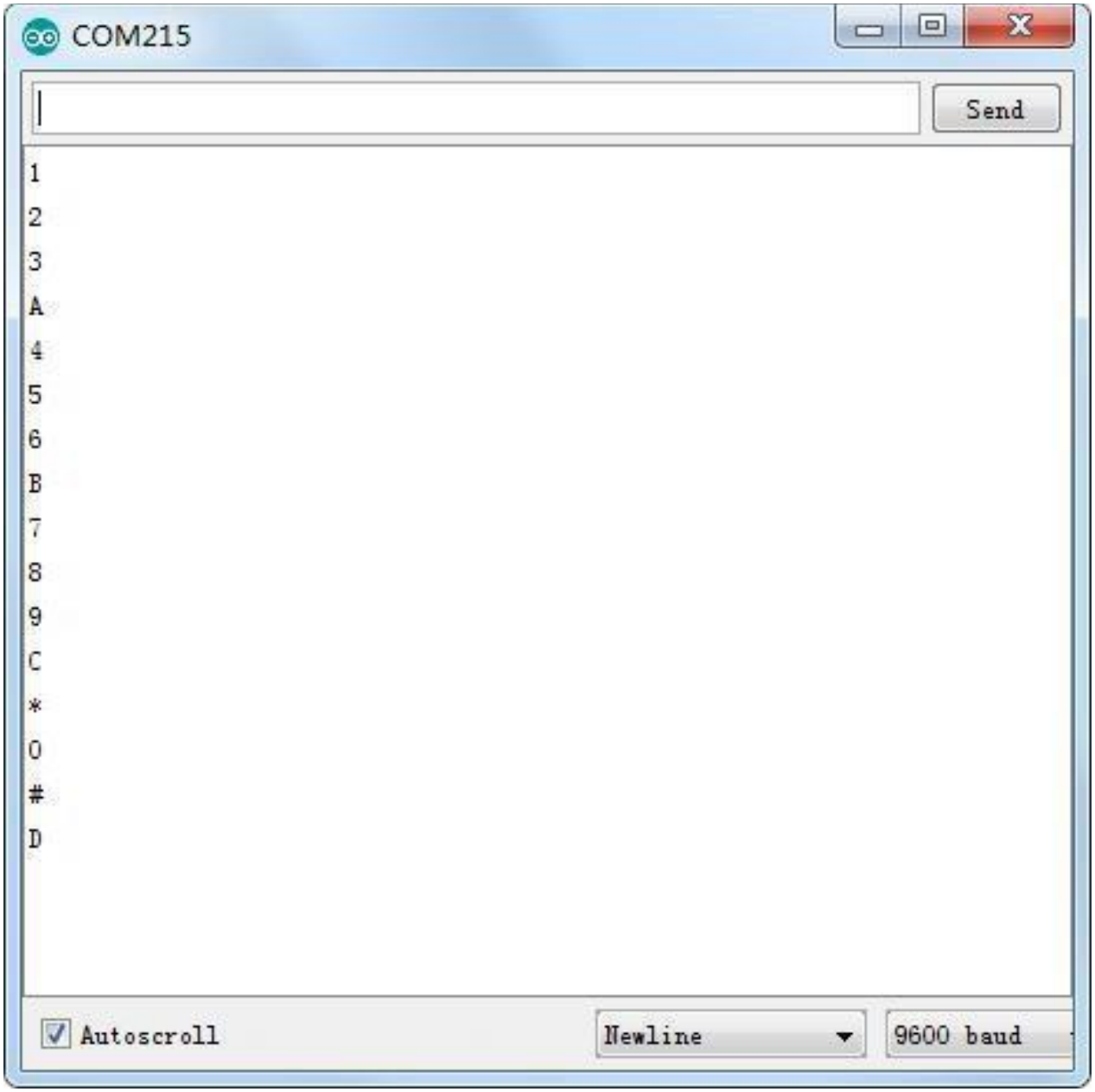
Beispielbild:




A B C D
3 6 9 #
2 5 8 0
1 4 7 *
MADE IN ITALY
ELEGOO
MEGA2560 R3

Immer wenn wir eine Taste auf der Tastatur drücken, sollte sie auf dem seriellen Monitor der Arduino IDE erscheinen. (Nachdem Sie den Code hochgeladen haben)

Klicken Sie auf die Schaltfläche Serieller Monitor, um den seriellen Monitor zu starten. Die Grundlagen des seriellen Monitors werden in Lektion 2 ausführlich erläutert.



Nachfolgend finden Sie den Code

```
#include <Keypad.h>

const byte ROWS = 4; //4 Zeilen auf der Tastatur
const byte COLS = 4; //4 Spalten auf der Tastatur
//Definieren, wie die Tasten auf der Tastatur angeordnet sind
char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {9, 8, 7, 6}; //definieren der Pins für die Zeilen
byte colPins[COLS] = {5, 4, 3, 2}; //definieren der Pins für die Spalten

//Anlegen eines Keypad Objekts, das sich um die Kommunikation mit der Tastatur kümmert
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

void setup(){
  Serial.begin(9600);
}

void loop(){
  char customKey = customKeypad.getKey(); //gedrückte Taste auslesen

  if (customKey){
    Serial.println(customKey); //gedrückte Taste an den seriellen Monitor schicken
  }
}
```

Nachwort

Als Ersteller und Herausgeber dieses Tutorials haben wir versucht Ihnen einiges über die Arduino und Sensorprogrammierung beizubringen, so dass Sie allmählich in der Lage sind, Aufgaben selbst zu lösen. Wenn Sie das Kit oder eines der Projekte modifizieren möchten, können Sie Ihrer Phantasie und Innovation freien Lauf lassen.

Wenn Sie Fragen oder Probleme beim Aufbau der Schaltungen/Programme haben, lassen Sie es uns wissen, aber geben Sie uns bitte auch etwas Zeit um zu antworten. Während Sie auf unsere Antwort warten, können Sie auch zuerst eine Online-Recherche durchführen; es gibt viele Communities, in denen Probleme und Lösungen diskutiert werden. Es ist eine erfüllende Erfahrung Probleme selbst zu lösen, was auch eine großartige Möglichkeit des Selbstlernens ist. Wenn Sie es vorziehen, mit unserem Kundenservice zu sprechen, hinterlassen Sie uns eine Nachricht unter service@elegoo.com oder euservice@elegoo.com, wenn Sie aus Europa und Japan kommen. ;)