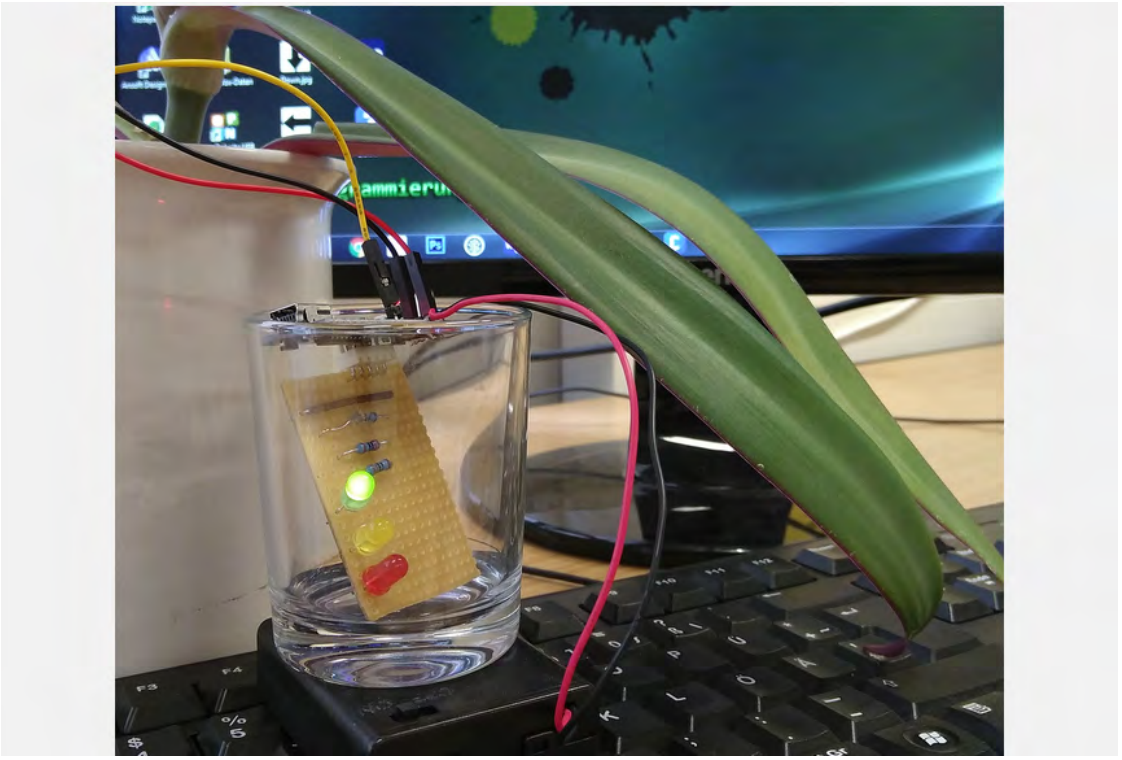
Projekt:
Blumenwächter

Leuchtender Gießalarm für Zimmerpflanzen



das Elektrotechnik- und Informatik-Labor der Fakultät IV
http://www.dein-labor.tu-berlin.de

Handout zum Projekt:

Blumenwächter

Leuchtender Gießalarm für Zimmerpflanzen

Inhaltsverzeichnis

1 Einleitung

Pflanzen sind Lebewesen und brauchen wie wir Menschen Wasser zum Überleben. Wenn du zu Hause eine Pflanze im Zimmer hast, weißt du, dass du sie auch gießen musst. Aber wann genau braucht die Pflanze Wasser? Reicht es, sie ein Mal in der Woche zu gießen? Oder solltest du dich doch lieber täglich um sie kümmern? Dieses Problem lösen wir in diesem Workshop mit unserem Blumenwächter. Er schlägt Gießalarm, wenn die Blumenerde zu trocken wird.

Im Workshop erfährst du zunächst, welche Bauteile wir brauchen. Unser „Gießalarm“ist eher eine „Gießampel“: Wir wollen eine rote LED leuchten lassen, wenn Gießbedarf ist, eine gelbe, wenn noch alles halbwegs ok ist, und eine grüne, wenn die Erde richtig gut feucht ist.

Außer drei bunten LEDs brauchen wir also noch einen Feuchtigkeitssensor, eine Batterie, eine elektronische Schaltung, die alles verbindet, und ein Programm, mit dem wir die Steuerung umsetzen.

In diesem Handout erfährst du, wie das geht!

2 Elektrotechnik

Die Elektrotechnik findet man an jeder Straßenecke. Überall blinkt es, große Leuchttafeln erzählen uns, was wir als nächstes kaufen sollen, und zu Hause flimmern im Fernseher die neuesten Nachrichten vor sich hin. Die Elektrotechnik hat sich zur Aufgabe gemacht, Elektrizität näher zu erforschen und sie in technischen Geräten einzusetzen.

WICHTIG
Die Elektrotechnik beschäftigt sich mit der Änderung von Strom und Spannung.

2.1 Strom und Spannung

Wenn man von einer „Strömung im Fluss" spricht, weiß jeder, was damit gemeint ist. Wenn ein Fluss schnell fließt, dann hat er eine hohe Strömung. Das gleiche gibt es in der Elektrotechnik. Statt Wasser haben wir eine Art Wolke aus ganz kleinen elektrisch geladenen Teilchen, die sich in einem Draht bewegen. Wir nennen diese Teilchen Ladungsträger.

WICHTIG
Wenn sich Ladungsträger bewegen, spricht man von einem Strom.
Formelzeichen: I für die Stromstärke
Einheit: A, gesprochen „Ampehr" (zum Andenken an den Physiker *André Marie Ampère*)

Wie fließt ein Fluss? Natürlich nur bergab, denn an dem Wasser zieht eine Kraft. Die Kraft ist die Anziehungskraft der Erde. Alles will nach unten, ein Apfel, den ich fallen lasse oder das Wasser, welches bergab fließt. In der Elektrotechnik gibt es ebenfalls so eine Kraft, diese nennen wir Spannung. Und wenn wir eine Spannung anlegen, dann bewegen sich Ladungsträger, also fließt ein Strom.

WICHTIG
Eine Spannung ist eine Kraft, die Ladungsträger in Bewegung setzt.
Formelzeichen: U
Einheit: V, gesprochen „Wolt" (zum Andenken an den Physiker *Alessandro Volta*)

2.2 Widerstand

Widerstand ist, wenn man sich gegen etwas wehrt! Das Bauelement *Widerstand* macht dies auch, es "wehrt sich"gegen den Stromfluss. Um es sich bildlich vorzustellen, nehmen wir einen Wasserschlauch. Das Wasser ist wieder unsere Wolke aus Ladungsträgern, und die Pumpe erzeugt eine Kraft, welche an dem Fluss zieht, also eine Spannung. Das Wasser sprudelt ungehindert aus dem Schlauch. Jetzt stellen wir uns vor, wir stellen uns auf den Schlauch.

Das Wasser hört sofort auf zu fließen. Nehmen wir den Fuß langsam herunter, läuft das Wasser immer schneller. Der Fuß stellt also einen Widerstand für den Wasserfluss dar.

WICHTIG

Je höher der Widerstand ist, desto geringer ist der Strom, der bei einer bestimmten Spannung fließen kann.
Formelzeichen: R
Einheit: Ω, gesprochen „Ohm"(zum Andenken an den Physiker *Georg Simon Ohm*)

Zwischen Strom, Spannung und Widerstand gibt es einen Zusammenhang: das *Ohm'sche Gesetz*. Das besagt, dass Strom und Spannung immer in einem bestimmten Verhältnis zu einander stehen. Dieses Verhältnis ist der Widerstand R: $\boxed{R = \frac{U}{I}}$

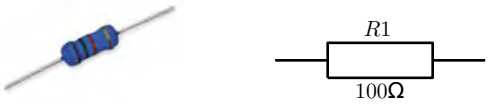


Abbildung 1: Widerstand: Bauelement (links) und Schaltungssymbol (rechts)

2.3 Leuchtdiode

Dioden sind vergleichbar mit einem Ventil. Sie lassen den Strom nur in eine bestimmte Richtung durch (genannt Durchlassrichtung), in die andere Richtung sperren sie. Wir verwenden farbige **Leuchtdioden** (Licht emittierende Dioden, LEDs). Fließt durch die Dioden elektrischer Strom in Durchlassrichtung, so strahlen sie Licht in verschiedenen Farben ab.

Abbildung 2: Leuchtdiode: Bauelement (links) und Schaltungssymbol (rechts)

3 Aufbau und Verdrahtung des Blumenwächters

Der Blumenwächter besteht aus vier Komponenten: Das Herzstück ist ein *Mikrocontroller*, das ist ein kleiner programmierbarer Computer. Er heißt *Arduino Nano*. Über Kabel wird er mit den anderen drei Komponenten verbunden. Da ist erstens der Feuchtigkeitssensor, der misst, ob die Erde noch feucht genug ist, und die Werte an den Mikrocontroller schickt. Zweitens haben wir dann die LED-Schaltung, also eine Platine, auf die wir drei farbige LEDs löten. Und drittens gibt es noch die Stromversorgung, also ein Kästchen mit drei Batterien, die den Arduino und unsere Schaltung mit Energie versorgt. In diesem Kapitel stellen wir dir alle Komponenten genauer vor.

3.1 Der Mikrocontroller: Arduino Nano

Der Mikrocontroller, den wir zur Steuerung des Blumenwächters verwenden, heißt *Arduino Nano*. Er hat einen Speicher für ein Programm und an den Rändern *Pins*, also Anschlüsse für Kabel, um elektronische Bauteile mit dem Arduino zu verbinden. Die Programmierung beschreiben wir später. Hier geht es zunächst nur um den Aufbau des Arduino-Nano-Mikrocontrollers. In Abbildung 3 siehst du, wie ein Arduino Nano aussieht.

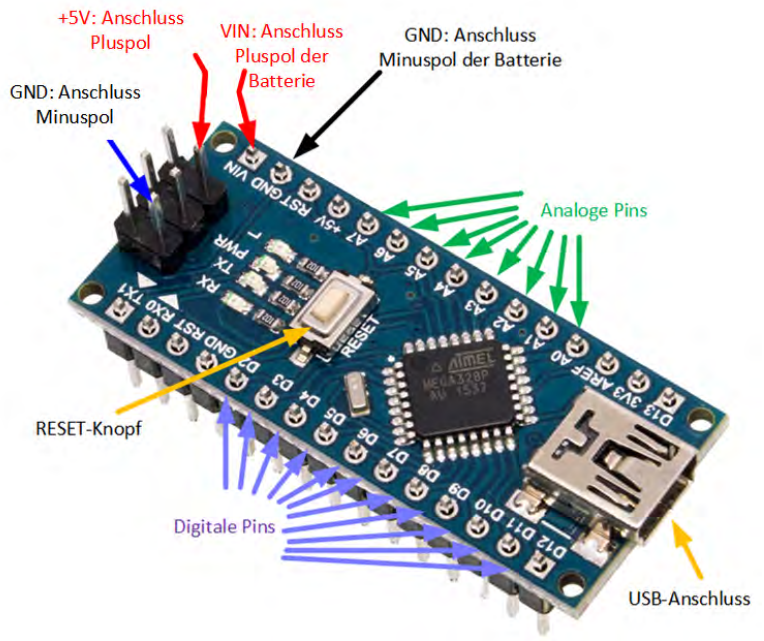


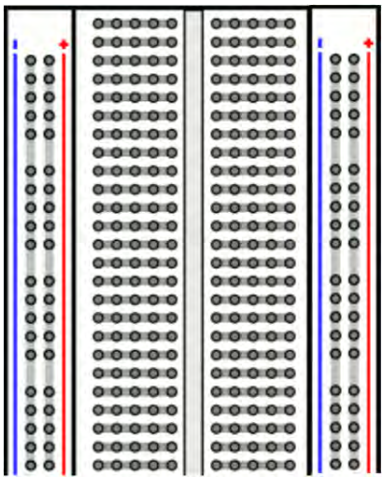
Abbildung 3: Aufbau des Arduino-Nano-Mikrocontrollers

An den Arduino lässt sich eine Batterie mit einer Spannung zwischen 5V und 12 V anschließen. Den Pluspol der Batterie muss man mit dem *VIN*-Pin verbinden und den Minuspol mit einem *GND*-Pin. Hast du den Arduino über die USB-Mini-Buchse mit dem PC oder Laptop verbunden, dann brauchst du gar keine Batterie, weil der Arduino dann über das USB-Kabel Strom und Spannung vom Computer bekommt. An die anderen Pins kann man nun elektro-

nische Bauteile anschließen, wie zum Beispiel Sensoren oder unsere drei farbigen LEDs. Die mit A beschrifteten Pins A0 bis A7 heißen *Analoge Pins*, weil hier Sensoren verbunden werden können, die Werte aus der realen (analogen) Welt messen können, wie zum Beispiel ein Feuchtigkeitsmesser. Auf der gegenüberliegenden Seite gibt es mit D beschriftete Pins D2 bis D12. Diese heißen *Digitale Pins*. Hier können Bauteile wie LEDs angeschlossen werden, die nur zwei Zustände (*An* oder *Aus*) haben können.

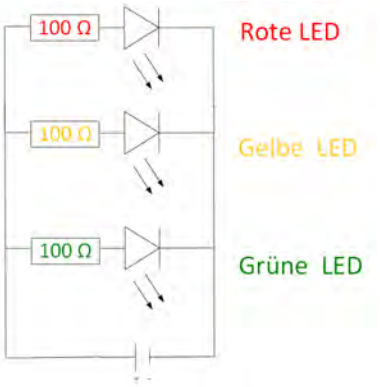
3.2 Drei LEDs leuchten lassen

Um drei LEDs leuchten zu lassen, musst du kein Programm schreiben. Du nutzt den Arduino als Spannungsquelle, indem du ihn über den USB-Port mit dem Computer verbindest. Nun fehlt noch der Stromkreis mit drei LEDs. Wir verwenden dafür ein Steckbrett (engl. Breadboard).



Auf der Zeichnung vom Steckbrett siehst du, welche Löcher unten durch Kupferkabel verbunden sind (die Verbindungen werden auch „Busse“genannt): Alle Löcher an der roten senkrechten Linie außen sind miteinander verbunden; genauso alle Löcher entlang der blauen Linie. In die rote Leiste wird üblicherweise ein (rotes) Verbindungskabel zum Pluspol der Batterie gesteckt (bei uns: zum 5V-Anschluss auf dem Arduino). In die blaue Leiste kommt das (meist schwarze) Kabel zum Minuspol der Batterie (bei uns: zum GND-Anschluss auf dem Arduino). Die Pluspol- und Minuspol-Leisten gibt es zweimal, links und rechts außen. Im inneren Teil sind immer fünf Löcher miteinander quer verbunden. In der Mitte gibt es einen großen Zwischenraum.

In unserem Stromkreis sollen drei LEDs parallel geschaltet werden und gleichzeitig leuchten. Eine LED kann aber nicht unmittelbar an eine 5V-Spannungsquelle angeschlossen werden. Wir müssen eine Strombegrenzung in unsere Schaltung einbauen.



Die einfachste Möglichkeit ist, in Reihe zu jeder LED einen Vorwiderstand zu schalten. Wie aber kommen wir darauf, dass die Vorwiderstände einen Wert von $R = 100\Omega$ haben sollten? Zur Berechnung des Vorwiderstands R gilt die Formel

$$R = \frac{U_{gesamt} - U_{LED}}{I_{LED}}$$

Bei uns ist $U_{gesamt} = 5V$, die Durchlassspannung der LED ist $U_{LED} = 2V$, und der Betriebsstrom der LED ist $I_{LED} = 30mA$. Es ergibt sich also:

$$R = \frac{5V - 2V}{0,03A} = 100\Omega$$

Diesen Schaltplan überträgst du nun auf das Steckbrett: Zunächst steckst du die LEDs (rot, gelb und grün) so, dass die kurzen Beinchen (Minuspole) jeder LED in der Minusleiste (blau) stecken, und die langen Beinchen jedes in einer extra Reihe in der Mitte. Genau in diese Reihe kommt dann zu jeder LED ein Beinchen eines 100Ω-Widerstands. Das jeweils andere Beinchen von jedem Widerstand steckst du in die Plusleiste (rot) vom Steckbrett. Nun müssen nur noch die Plusleiste und die Minusleiste selbst mit einem 5V- und einem GND-Pin am Arduino verbunden werden. Wo du die findest, zeigt dir Abbildung 4.

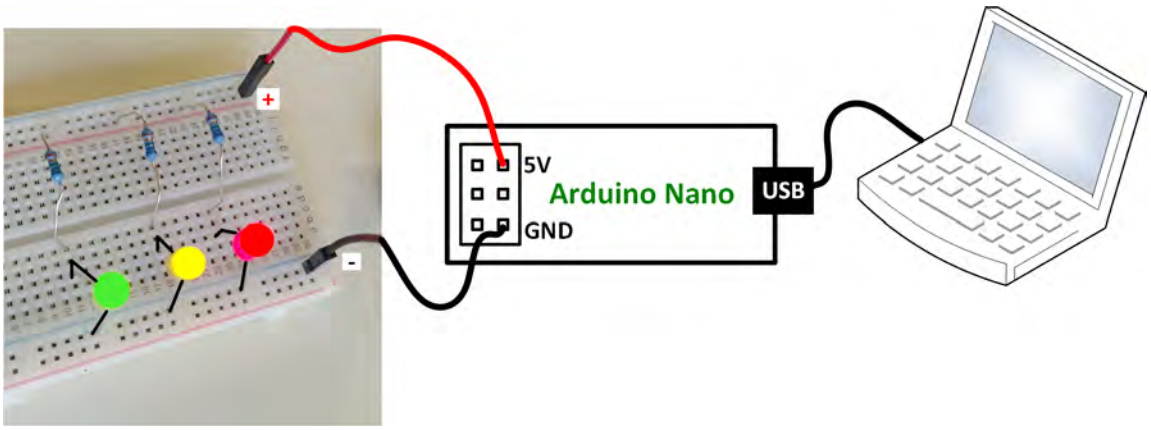


Abbildung 4: Die LED-Schaltung „Drei LEDs leuchten“

3.3 Drei LEDs blinken lassen

Nun wollen wir die LEDs aber nicht nur einfach leuchten lassen. Sie sollen blinken, und zwar in einer Geschwindigkeit, die wir vorgeben. Wir wollen die LEDs also nun steuern, und zwar über den Arduino, mit einem Programm, was wir selber schreiben.

Dazu änderst du zunächst deine Schaltung: Da jede LED jetzt nur noch leuchten darf, wenn sie ein Signal vom Arduino erhält, musst du jeden LED-Vorwiderstand nun anstatt mit der Plusleiste auf dem Steckbrett mit einem digitalen Pin auf dem Arduino verbinden. Verschiebe die Beinchen der Widerstände aus der Plusleiste in den Mittelbereich vom Steckbrett. Das rote Kabel verwendest du nun für die Verbindung der Plusseite des roten LED-Vorwiderstands mit einem digitalen Pin, z.B. Pin D8. Nimm ein gelbes und ein grünes Kabel und verbinde die Plusseiten der gelben und der grünen LED-Vorwiderstände jeweils mit den digitalen Pins D7 und D6, wie in Abbildung 5.

Nun zur Programmierung: Auf dem Computer vor euch findest du ein Programm, das genauso heißt wie der Mikrocontroller, nämlich Arduino. Wenn du es startest, siehst du das Fenster in Abbildung 6.

Ganz oben, vor den Abschnitt **setup{ ...}** (auf Deutsch *Einrichtung*), gehören Variablen. Das sind Container für Zahlen oder andere Informationen, die wir im Laufe des Programms verwenden wollen. In den Abschnitt **setup{ ...}** gehören Anweisungen, die beim Start des Programms ein einziges Mal ausgeführt werden. Hier wird zum Beispiel eingestellt, mit welchen

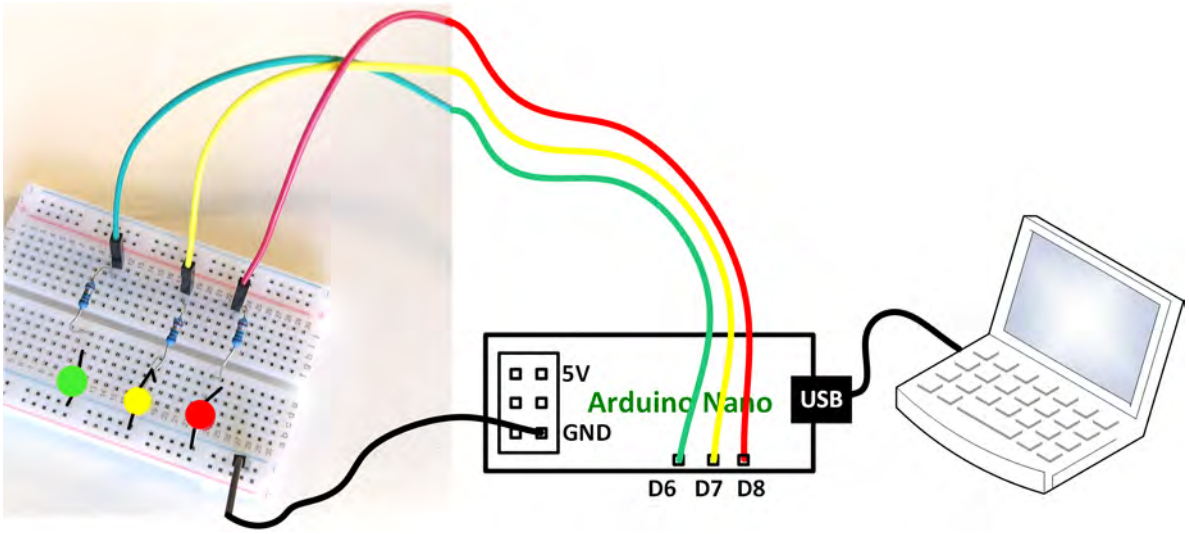


Abbildung 5: Die LED-Schaltung „Drei LEDs blinken“

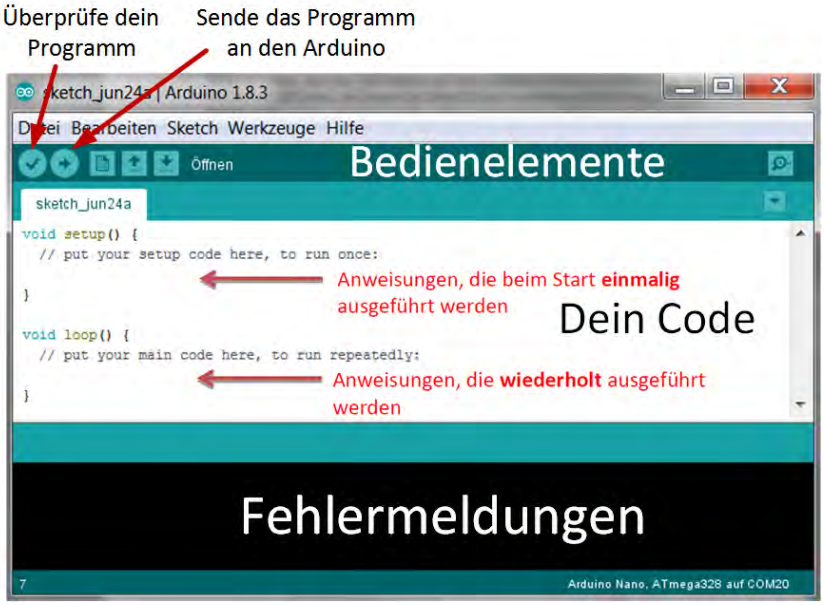


Abbildung 6: Die Arduino-Entwicklungsumgebung

Pins unsere LEDs verbunden sind. In den Abschnitt **loop{ ... }** (auf Deutsch *Schleife*) gehören Anweisungen, die nach dem Start des Programms nicht nur einmal durchlaufen werden, sondern immer wieder neu, in der Reihenfolge, in der sie in **loop{ ... }** stehen. Wenn nun eine LED blinken soll, dann schreibst du hier den Befehl für das Anschalten der LED hinein, und darunter den Befehl für das Ausschalten der LED. Fertig. Da beide Befehle immer wieder wiederholt werden, sehen wir ein ständiges Wechseln von An- und Ausschalten der LED: sie blinkt. Ergänze

das Programm in Abbildung 7, so dass nicht nur eine sondern alle drei LEDs blinken.

```
//--- Variablen: Namen statt Zahlen

int ledrotPin   = 8;
int ledgelbPin  = 7;
int ledgruenPin = 6;

//--- setup: Alles wird nur einmal durchlaufen

void setup() {  // Alle unsere LEDs sind Signale nach außen
  pinMode(ledrotPin, OUTPUT);
  pinMode(ledgelbPin, OUTPUT);
  pinMode(ledgruenPin, OUTPUT);
}

//--- loop: Am Ende beginnt es wieder von vorne

void loop() {
  digitalWrite(ledrotPin, HIGH);   // LED, gehe an
  delay(100);                      // Bleib so für 100 ms
  digitalWrite(ledrotPin, LOW);    // LED, gehe aus
  delay(100);                      // Bleib so für 100 ms
}
```

Abbildung 7: Das Programm „Drei LEDs blinken“

4 Den Blumenwächter zum Leben erwecken

Damit unser Blumenwächter weiß, welche LEDs er leuchten lassen soll, muss er wissen, wie feucht die Blumenerde ist. Dafür braucht er einen Sensor: den Feuchtigkeitssensor. Außerdem müssen wir unser Programm erweitern, damit die LEDs nur noch dann leuchten, wenn eine bestimmte Feuchtigkeit herrscht. Dazu brauchen wir „*Wenn-Dann-Sonst*-“Anweisungen (auf Englisch *if . . . else . . .*).

4.1 Der Feuchtigkeitssensor

In Abbildung 8 siehst du einen Feuchtigkeitssensor (engl. *moisture sensor*). Er sieht aus wie eine Gabel mit nur zwei Zinken. An diesen beiden Zinken, auch *Kontakte* genannt, liegt eine Spannung an. Die beiden Zinken werden in die Blumenerde gesteckt. Über zwei Kabel (rot und schwarz) wird der Sensor mit Strom versorgt, das dritte (gelb) dient zur Übertragung der Messwerte an den Arduino.

Je höher die Feuchtigkeit zwischen den beiden Kontakten ist, desto besser kann der Strom zwischen ihnen fließen. Die Information über den gemessenen Stromfluss wird dann als eine Zahl an den Mikrocontroller geschickt. Ein Feuchtigkeitssensor kann Zahlen im Bereich von 0 (Sensor ist trocken wie an der Luft) bis ca. 800 (Sensor ist komplett in Wasser getaucht) an den Arduino schicken. Die genaue Kalibrierung ist jedoch abhängig vom Sensor und von der Art der Flüssigkeit, die gemessen wird (bspw. hat Salzwasser eine bessere Leitfähigkeit und der

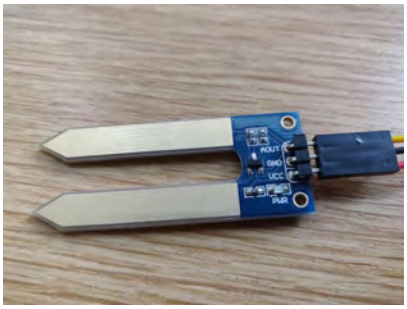


Abbildung 8: Der Feuchtigkeitssensor

Messwert wäre entsprechend höher). Mit den erhaltenen Zahlen können wir dann im Programm bestimmen, wann welche LED leuchten soll (z.B. bei „0“ die rote LED und bei „800“ die grüne.

4.2 Verdrahtung von Feuchtigkeitssensor und Arduino

Den Pluspol des Sensors verbindest du über das rote Kabel mit der Plusleiste vom Steckbrett, seinen Minuspol über das schwarze Kabel mit der Minusleiste. Nicht vergessen, die Plusleiste wieder mit dem 5V-Pin auf dem Arduino zu verbinden! Das gelbe Kabel am Sensor überträgt die Messwerte. Dieses verbindest du über den Umweg des Steckbretts mit dem analogen Pin *A0* auf dem Arduino (siehe Abbildung 9). Zur besseren Übersicht haben wir im Bild die drei LED-Kabel zum Arduino entfernt. Lasse sie aber ruhig drin.

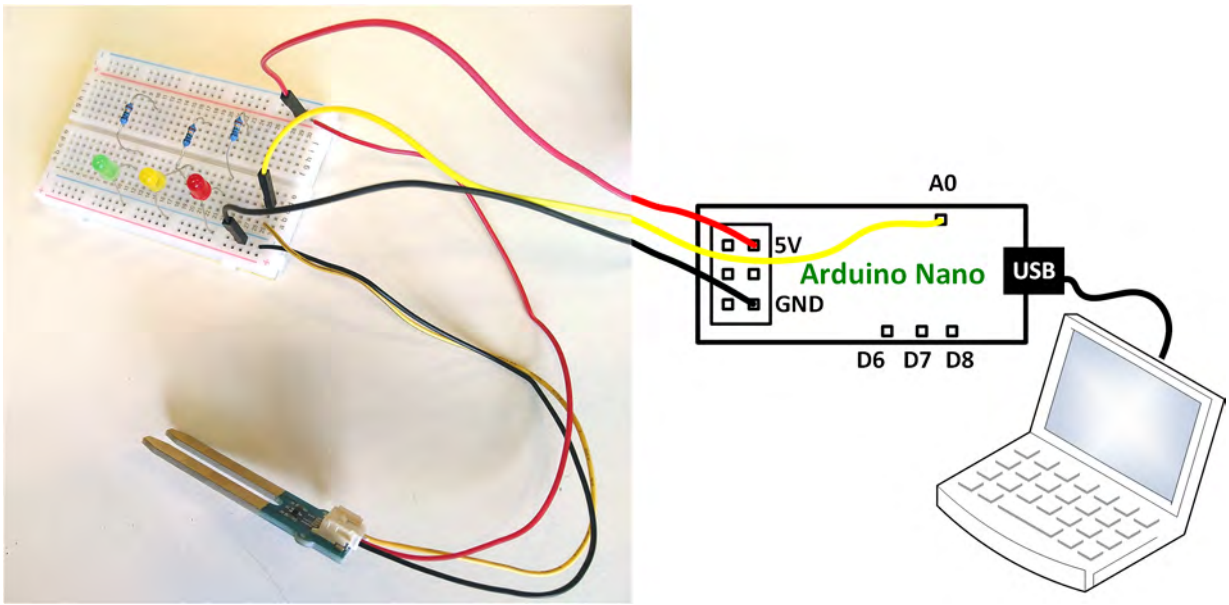


Abbildung 9: Schaltung mit Feuchtigkeitssensor

4.3 Programmierung des Blumenwächters

Die wichtigsten Befehle, die wir nun zusätzlich im **loop**-Bereich verwenden wollen, sind die Anweisungen zum Lesen der Zahlen vom Feuchtigkeitssensor und zum Ansteuern einer der drei LEDs: Rot soll leuchten, wenn die Sensor-Zahl sehr klein ist (die Erde ist trocken). Gelb soll so im mittleren Bereich leuchten, und Grün leuchtet, wenn die Zahl recht hoch ist (die Erde ist nass).

4.3.1 Anzeigen der Werte vom Feuchtigkeitssensor

Als erstes programmieren wir, dass der Wert, den der Feuchtigkeitssensor an den Arduino schickt, auf den Bildschirm in ein extra Fenster geschrieben wird. Dadurch können wir diese Zahl auch selber lesen und danach entscheiden, bei welcher Zahlen wir welche LED-Farbe leuchten lassen wollen. Wir müssen also im Loop-Block zwei Dinge tun: 1) den Messwert vom Feuchtigkeitssensor lesen und in einer Variable speichern; 2) diesen Wert auf den Bildschirm schreiben. Danach beginnt das Programm ja wieder von vorne, also werden ständig neue Werte gelesen und geschrieben. Bald ist das Fenster voller Zahlen ...

Die Variable, in die wir immer wieder den neuesten Wert des Sensors speichern, heißt *messwert*. Dass dies eine ganze Zahl ist, sagen wir dem Programm ganz oben. Das Fenster, in dem uns die Werte angezeigt werden sollen, heißt „Serieller Monitor“ (serial monitor). Im Setup-Block musst du diesen seriellen Monitor starten. Abbildung 10 zeigt das Programm. Schreibe es ab und teste es mit deinem Sensor. Dieses Programm läuft auch ohne die drei LEDs.

```
int messwert=0;                 //Unter der Variablen „messwert“ wird später der
                                Messwert des Sensors gespeichert.

void setup()
{                               // Hier beginnt das Setup.
Serial.begin(9600);             //Die Kommunikation mit dem seriellen Port wird
                                gestartet. Das benötigt man, um sich den
                                ausgelesenen Wert im serial monitor anzeigen zu
                                lassen.
}

void loop()
{                               // Mit dieser Klammer wird der Loop-Teil geöffnet.
messwert=analogRead(A0);        //Die Spannung an dem Sensor wird ausgelesen und
                                unter der Variable „messwert“ gespeichert.
Serial.print("Feuchtigkeits-Messwert:"); //Ausgabe am Serial-Monitor: Das Wort
                                          „Feuchtigkeits-Messwert:“
Serial.println(messwert);       //und im Anschluss der eigentliche Messwert
delay(500);                     //Zum Schluss noch eine kleine Pause, damit nicht
                                zu viele Zahlenwerte über den Serial-Monitor
                                rauschen.
}
```

Abbildung 10: Das Programm zum Anzeigen der Messwerte

4.3.2 LEDs leuchten abhängig vom Feuchtigkeitswert

Nun fehlen nur noch die Anweisungen, welche LED wann leuchten soll. Die LEDs müssen dafür wieder mit den Pins D6, D7 und D8 verbunden sein.

Im Programm fragst du den Wert (Variable *messwert*) ab und entscheidest je nach aktueller Zahl, was passieren soll. Beispielsweise kannst du die rote LED leuchten lassen, wenn der Messwert zwischen 0 und 300 liegt, die gelbe LED bei einem Messwert zwischen 301 und 500 und die grüne LED bei Werten über 500. Vergiss nicht, die anderen beiden LEDs auszuschalten, wenn nur die rote LED an sein soll. Das Programm dazu findest du in Abbildung 11.

```
//--- Variablen: Namen statt Zahlen

int ledgruenPin = 6;
int ledgelbPin  = 7;
int ledrotPin   = 8;
int sensorPin = 0;
int messwert = 0;

//--- setup: Alles wird nur einmal durchlaufen

void setup() {  // Alle unsere LEDs sind Signale nach außen
  pinMode(ledrotPin, OUTPUT);
  pinMode(ledgelbPin, OUTPUT);
  pinMode(ledgruenPin, OUTPUT);
  Serial.begin(9600);
}

//--- loop: Am Ende beginnt es wieder von vorne

void loop() {
  messwert = analogRead(sensorPin);
  Serial.println(messwert);

  if(messwert < 200){
  digitalWrite(ledrotPin, HIGH);    // Wenn trocken, dann
  digitalWrite(ledgelbPin, LOW);    // leuchte in rot
  digitalWrite(ledgruenPin, LOW);
  }
  if((messwert > 200) && (messwert < 500)){
  digitalWrite(ledrotPin, LOW);     // Wenn mittelfeucht, dann
  digitalWrite(ledgelbPin, HIGH);   // leuchte in gelb
  digitalWrite(ledgruenPin, LOW);
  }
  if(messwert > 500){
  digitalWrite(ledrotPin, LOW);    // Wenn nass, dann
  digitalWrite(ledgelbPin, LOW);   // leuchte in grün
  digitalWrite(ledgruenPin, HIGH);
  }
}
```

Abbildung 11: Das Programm „Blumenwächter“

Fertig ist der Blumenwächter. Du kannst aber auch kreativ sein und die jeweils aktuelle LED rot, gelb oder grün blinken lassen, weil das auffälliger ist. Bekommst du das hin?