Station 3 – Farbthermometer

Temperatur zum Leuchten bringen

Ihr habt euch also ein Projekt aus der Kategorie „Die Welt ein bisschen besser machen“ entschieden. Warum dieses Projekt die Welt besser macht? Wie viele Menschen haben sich schon die Finger verbrannt, weil die Herdplatte noch heiß war? Oder haben angewidert das Gesicht verzogen, weil der Tee schon kalt war? Das alles passiert nur, weil der Mensch Temperatur nicht sehen kann. Doch das könnt ihr jetzt ändern!

Abb. 1: LEDs

Abb. 2: Brunnen mit verbauten LEDs

Eure Aufgabe heute ist es, ein Thermometer zu konstruieren, das mittels verschiedener Farben anzeigt, welche Temperatur gerade herrscht.

AUFBAU DER SCHALTUNG

Eure Schaltung beinhaltet folgende Elemente:

- drei 220 Ω Widerstände
- Temperatursensor TMP36
- RGB-LED
- ein grünes, vier blaue, drei rote, ein gelbes

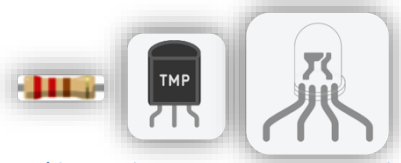


Abb. 3: Widerstand, Temperatursensor und

Farbmischung

RGB-LEDs

RGB-LEDs sehen zwar aus wie normale LEDs (die ihr ja aus dem Einstiegsprojekt kennt), aber in Wirklichkeit verstecken sich darin drei verschiedene LEDs! Nämlich eine rote, eine grüne und eine blaue LED, daher auch der Name RGB-LED. Das wirft natürlich die Frage auf, warum sie keine sechs Beinchen, sondern nur vier haben. Das liegt daran, dass der Hersteller die Minuspole schon im Gehäuse verbunden hat. Hier muss das Beinchen zwischen den Beinchen für rot und blau mit dem Minuspol verbunden werden. Habt ihr die RGB-LED auf eure Arbeitsfläche gezogen, könnt ihr mit eurer Maus über die Enden der Beinchen gehen und bekommt dann angezeigt, welches Beinchen zu welcher Farbe gehört. Normalerweise sind die Beinchen aber so angeordnet, wie ihr es in Abbildung 4 sehen könnt. In Tinkercad wird das Beinchen für den Minuspol mit Kathode bezeichnet; lasst euch davon nicht verwirren. Diese drei Farben reichen, um alle sichtbaren Farben darzustellen. Doch dazu später mehr.

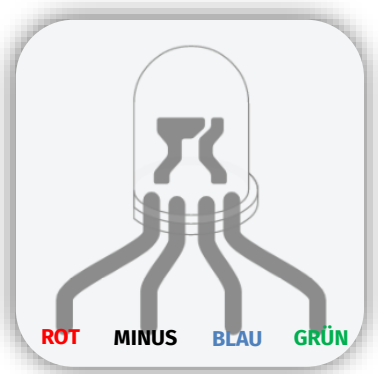


Abb. 4: RGB-LED

1. Als Erstes verbindet ihr die Plus- und Minusleiste des Steckbretts mit 5V bzw. GND auf dem Arduino, so wie ihr es aus dem Einstiegsprojekt kennt.
2. Steckt die RGB-LED jetzt in das Steckbrett, und verbindet den Minus-Pin mit der Minusleiste. Die anderen Pins kommen jeweils in eine eigene Reihe des Steckbretts. Anschließend werden diese Reihen jeweils mit einem Widerstand verbunden. Dahinter könnt ihr auch schon die drei Steckkabel in der jeweiligen Farbe einstecken, das obere Ende hängt aber erstmal in der Luft.
3. Verbindet nacheinander die Steckkabel jeweils mit der Plusleiste, und notiert eure Beobachtungen in der Tabelle auf der nächsten Seite.

[***Hinweis:*** *Habt ihr Probleme mit der Schaltung, könnt ihr in Abbildung 5 auf der nächsten Seite sehen, wie sie aussehen sollte.]*

CC BY SA

Station 3 – Farbthermometer

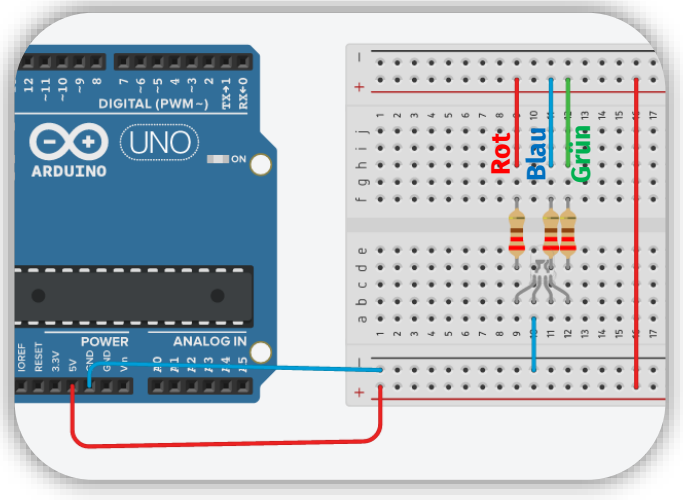


Abb. 5: Aufbau der RGB-LED-Schaltung

Verbundene Kabel	Farbe der LED
Rot	
Blau	
Grün	
Rot & Blau	
Rot & Grün	
Blau & Grün	
Rot & Blau & Grün	

Habt ihr die Tabelle schon ausgefüllt? Super! Nun stellt sich euch sicher die Frage, wie es sein kann, dass das Ergebnis überhaupt nichts mit dem zu tun hat, was ihr im Kunstunterricht über Farbmischung gelernt habt.

ADDITIVE FARBMISCHUNG

Aus dem Kunstunterricht kennt ihr ja den guten alten Malkasten. Wenn ihr dort die Farben Blau und Gelb mischt, erhaltet ihr Grün. Das funktioniert, weil sich das weiße Licht aus allen Farben zusammensetzt. Die Farben aus dem Malkasten filtern bestimmte Anteile aus dem Licht, sodass beim Mischen mehrere Anteile wegfallen. Mischt man alle Farben, wird jeder Anteil gefiltert, und das Resultat ist schwarz.

Abb. 6: Farbmischung

Im Gegensatz zu diesem sogenannten subtraktiven Farbmodell, nehmt ihr bei RGB-LEDs kein Licht weg, sondern mischt neues dazu. Mischt man alle Anteile, so erhält man hier weißes Licht. Die Grundfarben sind hier Rot, Grün und Blau. Mit diesen lassen sich alle Farben des sichtbaren Lichts erzeugen.

Dieses Prinzip machen sich auch Fernseher und Monitore zu nutze. Betrachtet ihr den Bildschirm aus der Nähe, so seht ihr, dass auch dieser sich aus winzigen roten, grünen und blaue Bildpunkten zusammensetzt.

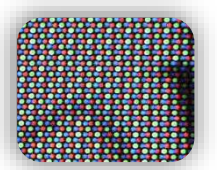
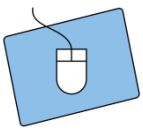


Abb. 7: Nahaufnahme Monitor

Natürlich ergibt es Sinn, die RGB-LED mit dem Arduino zu steuern, statt immer Kabel umstecken zu müssen.



1. Öffnet das Code-Fenster in eurer Schaltung, und stellt es auf „Text“.
2. Erstellt im Bereich Einstellungen für die drei LEDs Variablen vom Typ int, und speichert in ihnen die Pins 9 (Grün), 10 (Blau) und 11 (Rot) ab.
3. Die LEDs sind an Ausgängen des Arduino angeschlossen. Legt den richtigen Anschlusstypen mit **pinMode** für die drei Variablen fest.
4. Schaltet die LEDs nacheinander und in verschiedenen Kombinationen mit **digitalWrite** an und aus.
5. Benutzt **delay()** zwischen den Schaltvorgängen, um die Unterschiede zu sehen.
6. Ausprobieren. :)

Falls euch das Prinzip der RGB-LED noch nicht ganz klar ist, hilft dieses Bild vielleicht.
Stellt euch die RGB-LED als eine blaue, grüne und rote LED in einer einzigen vor. Das (in diesem Bild) linke Beinchen steuert die blaue, das mittlere die grüne und das rechte die rote LED. Das vierte Beinchen ist die Verbindung aller drei Farben zum Minuspol.

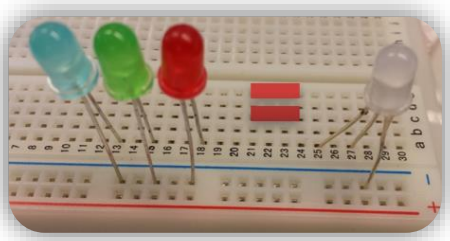


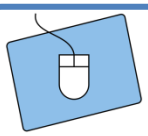
Abb. 8: Funktionsweise RGB-LED

CC BY SA

Station 3 – Farbthermometer

MEHR ALS NUR DREI FARBEN

Wie ihr bereits an der Tabelle auf Seite 2 gesehen habt, können RGB-LEDs noch mehr als nur drei Farben darstellen. Dazu müsst ihr zwei oder drei der eingebauten LEDs anschalten.



1. Erweitert euren bisherigen Code nun so, dass eure RGB-LED ein Farbspiel aus Rot, Grün, Blau, Gelb, Pink, Türkis und Weiß aufführt.
2. Testet euer Programm.

All diese Farben könnt ihr später nutzen, um verschiedene Temperaturen darzustellen. So sieht selbst ein Kind auf den ersten Blick den Unterschied zwischen warm und kalt.

Die Temperatur

Es gibt viele verschiedene elektronische Bauteile, die Temperaturen messen können. Der TMP36 ist eines davon. Auch wenn er nicht danach aussieht, beherbergt er eine komplexe Schaltung, die euch die Arbeit deutlich erleichtert.

TMP36

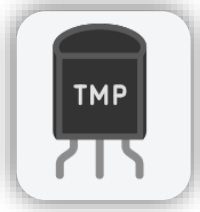


Abb. 9: TMP36

Der TMP36 hat drei Beinchen. Die komplexe Schaltung im Inneren muss mit Strom versorgt werden. Hierzu dienen die beiden äußeren Beinchen. Zeigt die flache Seite zu euch, so muss das linke Beinchen mit dem 5V-Anschluss (also der Plusleiste) verbunden werden, das rechte Beinchen verbindet ihr mit GND (also der Minusleiste).
Der mittlere Pin erfüllt die eigentliche Funktion des Temperaturmessers: Pro 1 °C liegt hier eine Spannung von 10 mV an. Bei 20 °C liegen also 200 mV an.
Verbunden wird der mittlere Pin des TMP36 mit einem **ANALOGEN EINGANG** des Arduino.

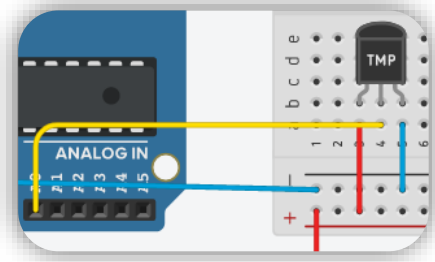


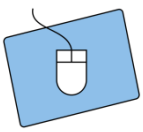
Abb. 10: Schaltung mit TMP36

ANALOGE PINS

Bisher kennt ihr nur **DIGITALE PINS**, über die z. B. eine LED angeschlossen wird. An diesen Pins können nur **BINÄRE WERTE** ein- und ausgelesen werden – also ein/aus, 0/1 oder eben die bekannten Werte **HIGH** und **LOW** für eine hohe bzw. niedrige Spannung.
Ein Temperatursensor wie der TMP36 kann aber nicht nur zwei Werte annehmen, sondern misst im Bereich zwischen dem minimalen und maximalen Wert ganz viele **ZWISCHENWERTE**. Das realisieren am Arduino **ANALOGE PINS** (*A0 bis A5*), an denen ein ganzer **BEREICH** gemessen werden kann.
Die analogen Eingänge des Arduino werden mit der Funktion

variablen-name **= analogRead(**pin-name**)**;

ausgelesen. Diese Funktion liefert Werte zwischen 0 und 1023. 0 bedeutet, dass keine Spannung anliegt; bei 1023 liegen volle 5 Volt an.



1. Erweitert euren Code um eine weitere int-Variable für den analogen Pin des Temperatursensors.
2. Bevor es weitergeht, speichert ihr den Sensorwert am besten in einer weiteren Variable. Der Sensor liefert nur ganze Zahlen, also braucht ihr eine Variable vom Typ int. Weist dieser in **loop()** den Sensorwert zu.

CC BY SA

Station 3 – Farbthermometer

DIE RECHNUNG

Euch stehen also folgende Fakten zur Verfügung:

- ein Sensorwert zwischen 0 und 1023
- eine Skala der Spannung von 0 bis 5 V
- ein Sensor, der 10 mV, also 0,01 V pro 1 °C liefert
- Es gibt eine Abweichung von 0.5 °C bei der Spannung.

Ihr ahnt es vielleicht schon: Da steht eine Menge Umrechnen an! Aber keine Sorge, das wird jetzt Schritt für Schritt erledigt.

Um die Temperatur aus dem Sensorwert zu errechnen, müsst ihr die Spannung kennen, die am TMP36 anliegt. Dafür müsst ihr den Sensorwert in die Spannung umrechnen. Dies gelingt mit folgender Formel:

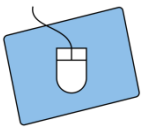
SPANNUNG = (5.0 / 1023) * SENSORWERT

5.0 statt 5 sagt dem Arduino, dass er mit Kommazahlen rechnen soll.

Der Sensor liefert 0,01 V pro °C, und da ihr die Spannung jetzt kennt, könnt ihr diese durch den Umrechnungsfaktor 0,01 teilen und habt die Temperatur. Denkt daran, die 0.5 °C Abweichung von der Spannung abzuziehen.
Wer gut in Mathe aufgepasst hat, der weiß, dass geteilt durch 0,01 das Gleiche ist wie mal 100. Daher sieht die Formel jetzt folgendermaßen aus:

TEMPERATUR = 100 * (SPANNUNG – 0.5)

So, und jetzt zur Programmierung!



1. Für die Spannung benutzt ihr eine Division mit **KOMMAZAHLEN**. Dafür braucht ihr einen neuen Variablentyp: float. Erstellt Variablen für Spannung und Temperatur im Bereich Einstellung vom Typen float:
   float temperatur;
   float spannung;
2. Errechnet die Spannung, und speichert sie in eurer Variable ab.
3. Berechnet die Temperatur, und speichert auch sie in der angelegten Variable ab.
4. Startet den **SERIEN MONITOR** in **setup()**, und gebt in **loop()** den Sensorwert, die Spannung und auch die Temperatur aus.
5. Fügt noch ein **delay()** ein, sodass der Arduino die Temperatur nur einmal pro Sekunde überprüft.
6. Ändert die Temperatur, indem ihr während der Simulation auf den Sensor klickt und den Schieberegler bewegt.
7. Schaut euch die Werte auf dem Serien Monitor an, und macht euch ein paar Notizen zu einzelnen Beispielwerten.

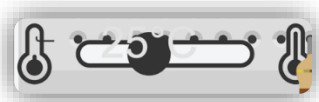


Abb. 11: Schieberegler

CC BY SA

Das Farbthermometer

Funktioniert alles? Gut, jetzt habt ihr schon alles beisammen, um Temperaturen zu messen! Zudem habt ihr eine RGB-LED, mit der ihr mehrere Farben darstellen könnt. Ihr habt also alles, um die Temperatur anzuzeigen.

Was haltet ihr davon, die LED bei Raumtemperatur blau leuchten zu lassen? Wenn es wärmer wird, könntet ihr über gelb nach rot wechseln. Nutzt die folgende Tabelle, um zu notieren, bei welcher Temperatur ihr welche Farbe anzeigen wollt! Und natürlich auch welche der drei LEDs dazu leuchten muss.

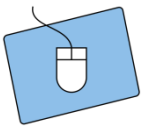
Temperatur	Farbe	Rot	Grün	Blau

Das klingt nach einem Plan! Allerdings muss euer Code dafür noch ein wenig erweitert werden.

IF-ANWEISUNG MIT BEDINGUNGEN FÜR FORTGESCHRITTENE

Eine Bedingung muss nicht immer prüfen, ob zwei Dinge den gleichen Wert haben (zur Erinnerung: ==), sondern kann auch vergleichen, ob etwas kleiner (<), größer (>) oder kleiner/größer gleich (<= bzw. >=) ist. Mit if könnt ihr auch mehr als eine Bedingung gleichzeitig abfragen. Verknüpft diese Bedingungen dazu mit einem &&. Dabei müsst ihr aber um die einzelnen Bedingungen Klammern schreiben. So wird die Anweisung nur ausgeführt, wenn beide Bedingungen zutreffen.

if ((bedingung1) && (bedingung2)) {anweisungen}



8. Erstellt für jeden eurer Temperaturbereiche eine if-Anweisung. Der unterste und oberste Bereich haben jeweils nur eine Bedingung, alle Bereiche dazwischen zwei.
9. Lasst je nach Bereich die richtigen LEDs leuchten.
   [***Hinweis:*** *Vergesst nicht, die jeweils anderen (unbenutzten) LEDs auszuschalten.*]

Wenn alles geklappt hat, sollte es bei euch ungefähr so wie in den beiden Bildern unten funktionieren! Falls nicht, checkt noch mal eure if-Bedingungen und ob ihr jeweils die richtigen LEDs ein- und ausgeschaltet habt.

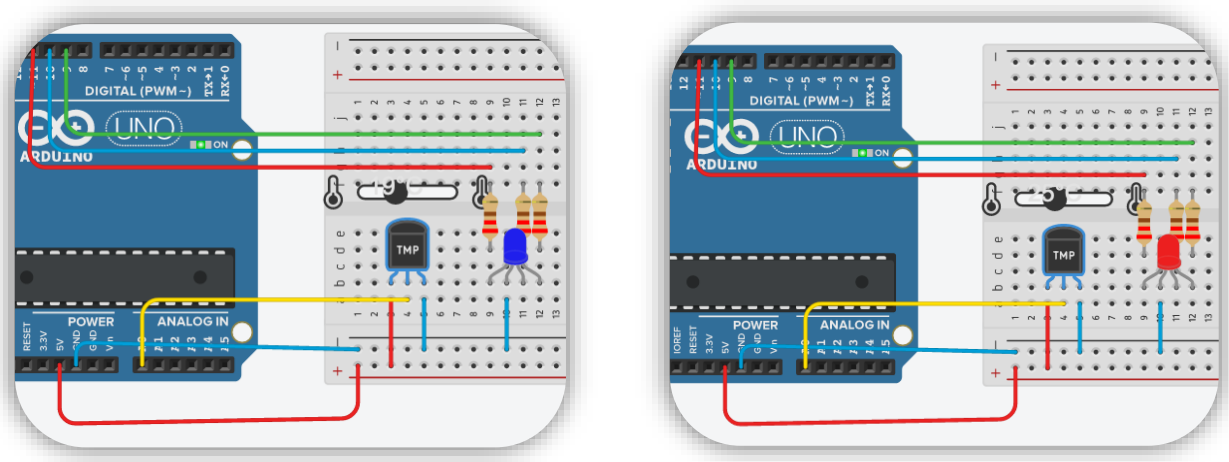


Abb. 12a und b: Komplette Schaltung. Links auf Raumtemperatur (blau), rechts heißer (rot)

CC BY SA

Station 3 – Farbthermometer

Hat alles geklappt? **HERZLICHEN GLÜCKWUNSCH!**

Quellenverzeichnis:
Abb. 1 – *Quelle: wikipedia.org, Autor: Akimbomidget (CC BY-SA 2.5)*
Abb. 2 – *Quelle: wikipedia.org, Autor: Diliff (CC BY-SA 3.0)*
Abb. 3, 4, 5, 9, 10, 11, 12 – *Quelle: Screenshots von Tinkercad https://www.tinkercad.com/*
Abb. 6 – *Quelle: wikipedia.org, Autor: Quark67 (CC SA 3.0)*
Abb. 7 – *Quelle: wikipedia.org, Autor: Ernst Schütte (CC SA 3.0)*
Abb. 8 – *Quelle: InfoSphere*
, , , angefertigt vom InfoSphere-Team

CC BY SA

CC BY SA