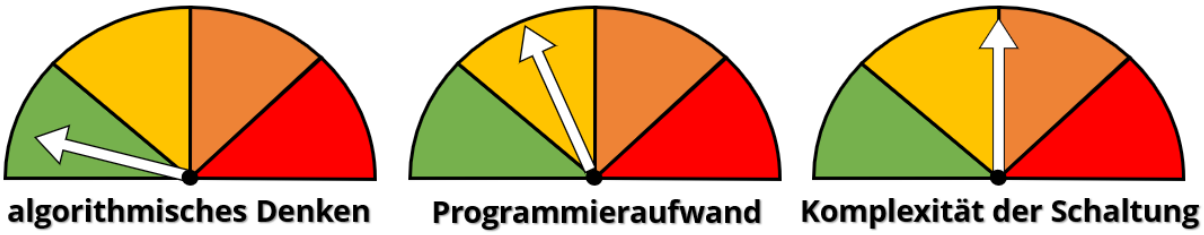
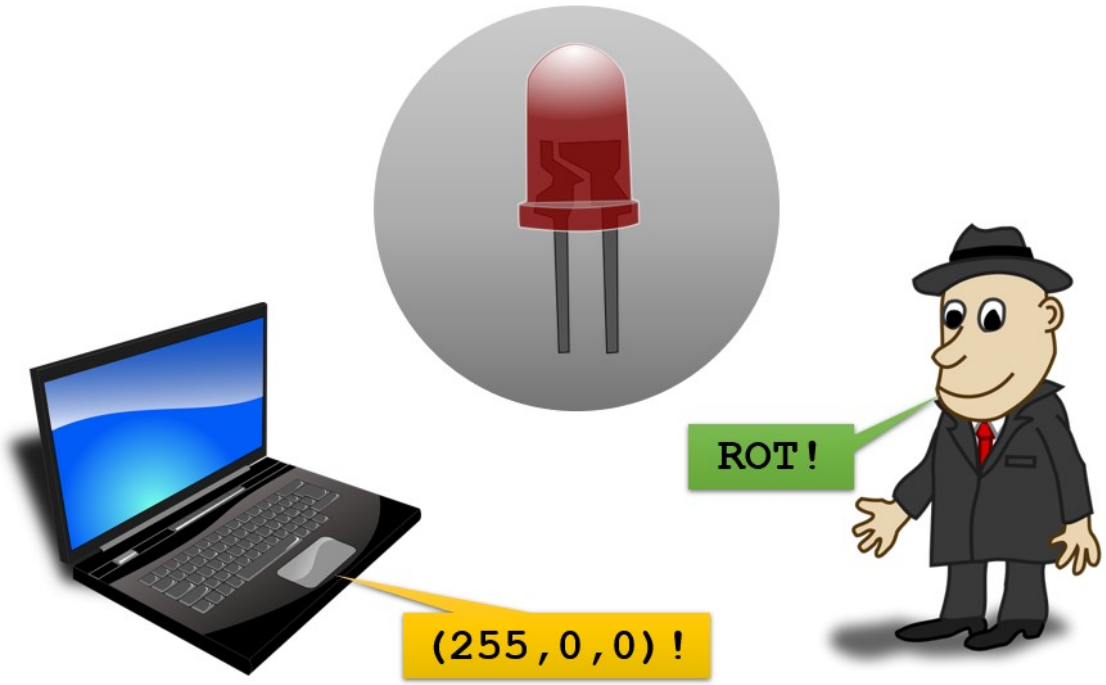
TECHNISCHE UNIVERSITÄT DRESDEN

Arduino Uno

Station 6a | Farbsensor und RGB-LED





Professur für Didaktik der Informatik

SYLBER Synergetische Lehrerbildung im exzellenten Rahmen

Das Maßnahmenpaket „TUD-Sylber² – Synergetische Lehrerbildung im exzellenten Rahmen“ wird im Rahmen der gemeinsamen „Qualitätsoffensive Lehrerbildung“ von Bund und Ländern aus Mitteln des Bundesministeriums für Bildung und Forschung gefördert.

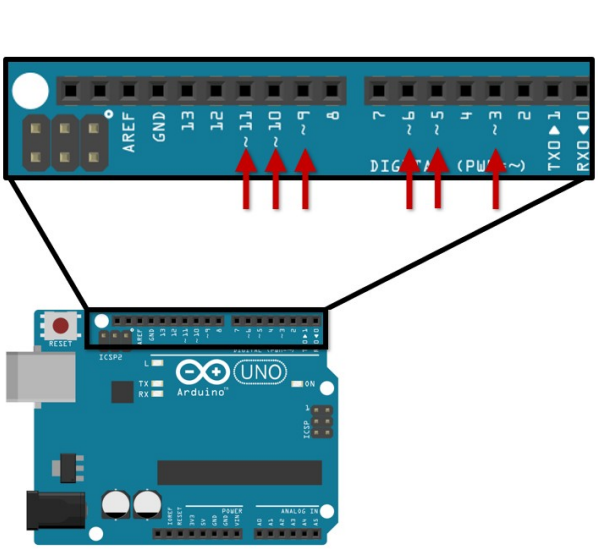
GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

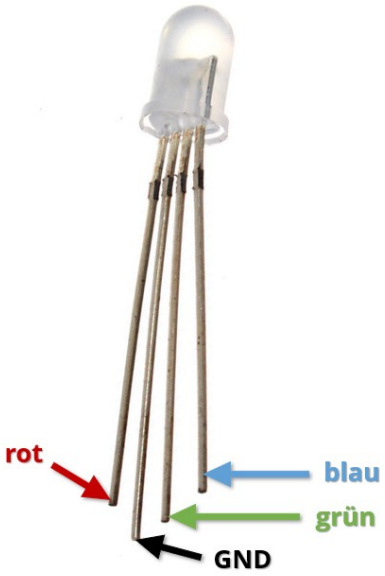
ZIEL DER STATION

Die Darstellung der eingescannten Farben soll nun nicht mehr über den seriellen Monitor ablaufen, sondern optisch mit Hilfe der RGB-LED dargestellt werden. Dazu wiederholst du nochmal kurz die wichtigsten Grundlagen, ergänzt deine Schaltung um die LED und fügst den einfachen Verzweigungen die jeweiligen Befehle hinzu, damit die LED je nach gescannter Farbe leuchtet.

WIEDERHOLUNG RGB-LED

Die RGB-LED hat im Gegensatz zu einer normalen LED vier anstatt zwei Beinchen. Diese sind wie in der unten stehenden Grafik zu verbinden. Achte hierbei darauf, dass du an `GND` einen zusätzlichen Widerstand (230 Ω) einbaust und die Pins für die Pulsweitenmodulation benutzt. Diese sind auf dem Board durch eine Tilde ~ gekennzeichnet.





Nach dem Verbinden der LED solltest du zunächst die notwendigen Variablen initialisieren, beispielsweise für den roten Pin:

```
int LEDrot = 3
```

Und die Pins für die jeweiligen drei Farben den Pin-Modus auf Output festlegen. Nutze dazu den folgenden Befehl.

```
pinMode(LEDrot, OUTPUT)
```

Weiterhin hast du bereits schon mal eine Funktion implementiert, mit Hilfe derer wir die unterschiedlichen Farben gemäß des Farbmodells darstellen können. Diese sah wie folgt aus:

```
void rgbfarbe (int rot, int gruen, int blau){
     analogWrite(LEDrot, rot);
     analogWrite(LEDgruen, gruen);
     analogWrite(LEDblau, blau);
}
```

Der Aufruf der Funktion könnte beispielhaft für Farbe **rot** so aussehen:

```
rgbfarbe(255,0,0);
```

AUFGABE 1 – RGB-LED VERBINDEN UND FUNKTION IMPLEMENTIEREN

Füge deiner Schaltung nun zusätzlich zum Farbsensor die RGB-LED hinzu. Initialisiere die dazugehörigen Variablen und implementierte die oben beschriebene Funktion. Vergiss nicht, die jeweiligen Pin-Modi für die LED zu ergänzen.

Teste die Funktionsweise in dem du Funktion aufrufst und Farben deiner Wahl anzeigen lässt. Dies kannst du tun ohne den vorherigen Code zu löschen, füge einfach innerhalb der `loop()`-Methode den Funktionsaufruf gefolgt von einem `delay()`-Befehl ein.

Nun wollen wir die Anzeige des gescannten Farbwertes realisieren. Dazu kannst du den Funktionsaufruf für die RGB-LED innerhalb der jeweiligen Verzweigungen mit den korrekten Farbwerten hinzufügen. Ich empfehle dir hierbei die Anzeige im seriellen Monitor nicht zu löschen, damit du die Möglichkeit hast die Ausgabe zu kontrollieren.

```
if (rot < blau && rot < gruen){
     Serial.println("Farbe rot erkannt!");
     rgbfarbe(255,0,0);
     delay(1000);
} else if(Bedingung für grüne Farbe){
     ...
} else if(Bedingung für blaue Farbe){
     ...
} else {
     Serial.println("Keine Farbe erkannt! :( ");
}
```

AUFGABE 2 – DARSTELLUNG MIT DER RGB-LED

Ergänze nun jeweils die Fallunterscheidungen mit dem korrekten Funktionsaufruf, für die jeweilige Farbe. Teste deinen Aufbau im Anschluss mit den zur Verfügung gestellten farbigen Materialien.

Grafik auf dem Deckblatt: *Rote LED-Lampe aus, Openclipart, CC0, https://creativecommons.org/publicdomain/zero/1.0/, https://publicdomainvectors.org/de/kostenlose-vektorgrafiken/Rote-LED-Lampe-aus/61716.html*

und

Kaufmann-comic-Figur-Vektor-Bild, CC0, https://creativecommons.org/publicdomain/zero/1.0/, https://publicdomainvectors.org/de/kostenlose-vektorgrafiken/Kaufmann-comic-Figur-Vektor-Bild/6840.html

Screenshots: *fritzing electronics made by easy und Arduino IDE 1.8.12 (windows)*

Abbildung 1: *5mm RGB LED, oomlout, CC BY-SA 2.0, https://creativecommons.org/licenses/by-sa/2.0/deed.en, https://creativecommons.org/licenses/by-sa/2.0/deed.en*

Alle weiteren Grafiken: *Patrick Binkert, EduInf@TUD*