

Arbeitsblatt 2: Licht manuell steuern

Beschreibung: Schalter und Taster verwenden

Großartig, das wäre geschafft! Unser LED-Ausgang „13“ kann nun mit einem Namen angesprochen werden. Statt das Licht nur wahllos blinken zu lassen, würden wir es gerne durch einen Taster steuern können.

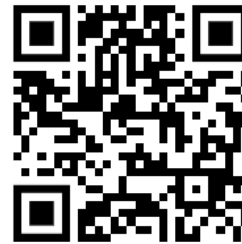
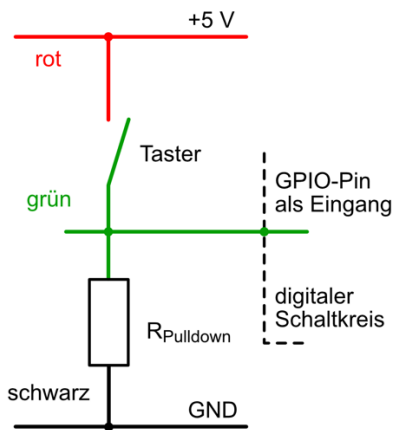


Fotos: R. P. Dröge 2020



a) PULLDOWN-Widerstand

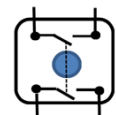
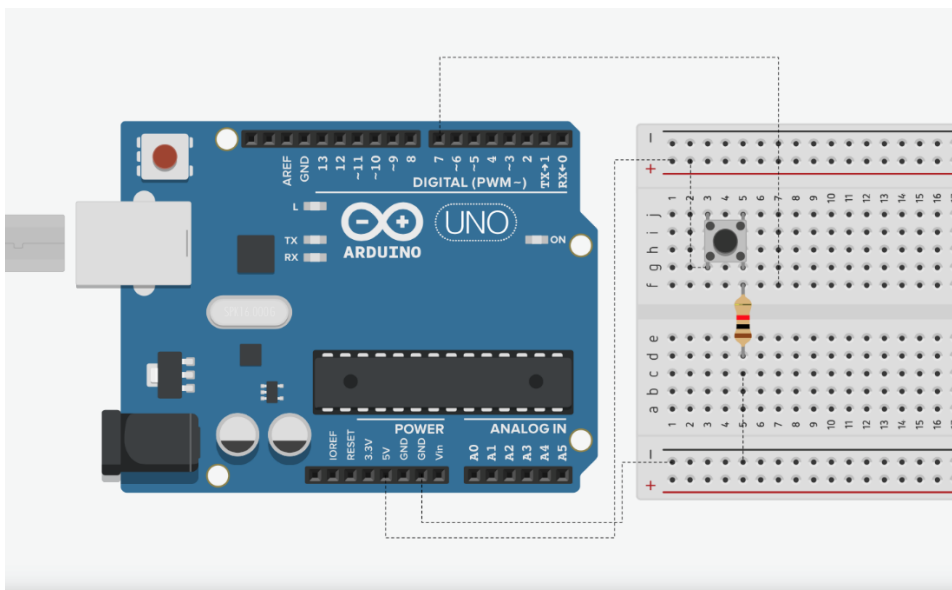
Durch den inneren elektrischen Aufbau eines Mikrokontrollers (= Arduino) muss zur Erkennung eines Tasters oder Schalters ein „PULLDOWN-Widerstand“ verwendet werden. Die nachstehende Schaltung zeigt den elektrischen Schaltplan für die Taster-Widerstand-Kombination.



Homepage:
„PULLDOWN
_Widerstand
erklärt“



In TinkerCAD sieht die Umsetzung einer „Taster-Erkennung“ mit „PULLDOWN-Widerstand“ wie folgt aus. Male die Verbindungen mit den gleichen Farben wie im Schaltplan oben aus.



Quelle: BSZ Bietigheim-Bissingen

b) Umsetzung

Nachdem nun die Hardware zur Erfassung eines Taster-Signals umgesetzt ist, fehlt nun nur noch die Software. Für das Einlesen eines Tasters muss der Befehl „digitalRead()“ verwendet werden. Das Ergebnis bekommt natürlich wieder einen Namen (eine Variable) und zwar den Namen „buttonState1“.



```
bool buttonState1=0;           //Variable vom Typ „boolean“ wird angelegt mit Wert „0“

buttonState1=digitalRead(7);  // Der Taster wird eingelesen und der Wert der Variable
                               zugewiesen.
```



Wenn der Taster gedrückt wird, wird die Variable buttonState auf „HIGH“ („1“) gesetzt.

```
void setup()
{
  pinMode(LED1, OUTPUT);
  pinMode( , INPUT);
}

void loop()
{
  buttonState1 =  // Befehl zum Einlesen des Taster-Wertes
  digitalWrite(LED1, buttonState1);  // die Lampe leuchtet bei Betätigung
}
```



Für Profis:

Füge einen zweiten Taster („buttonPin2“ auf Port 6 mit dem Status „buttonState2“) und eine zweite LED hinzu (LED2 an Port 12). Ist der Taster gedrückt, soll die LED2 im Sekundentakt blinken.