Teil 2

Lektion 3

Digitale Eingänge

Übersicht

In dieser Lektion werden Sie lernen Drucktaster in Verbindung mit den digitalen Eingängen des UNO Boards zu benutzen, um eine LED an- und auszuschalten.

Durch Drücken des einen Schalters wird die LED eingeschaltet, drückt man den anderen Schalter, wird sie wieder ausgeschaltet.

Benötigte Bauteile:

(1) x Elegoo Uno R3

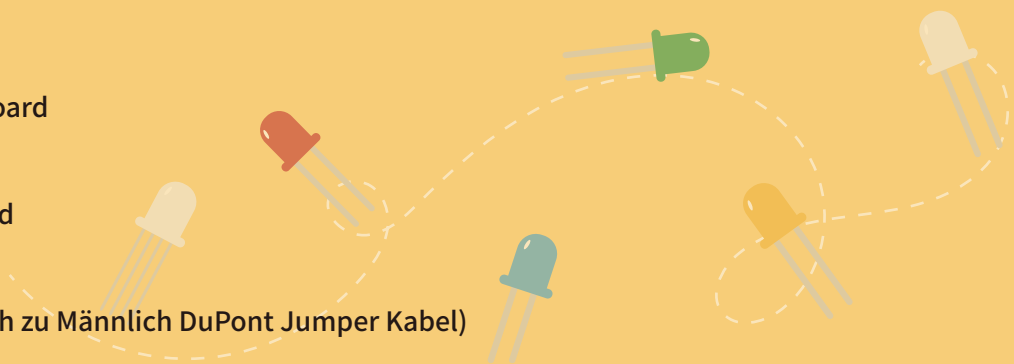
(1) x 830 Punkte Breadboard

(1) x 5mm rote LED

(1) x 220 Ohm Widerstand

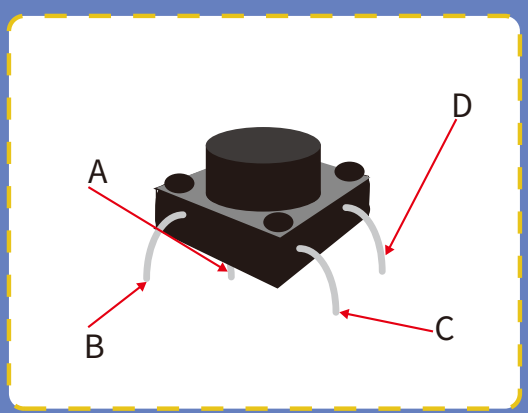
(2) x Drucktaster

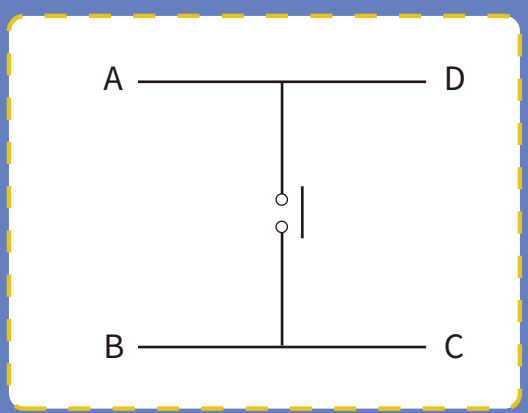
(7) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)



Einführung in die Komponenten
Drucktaster:

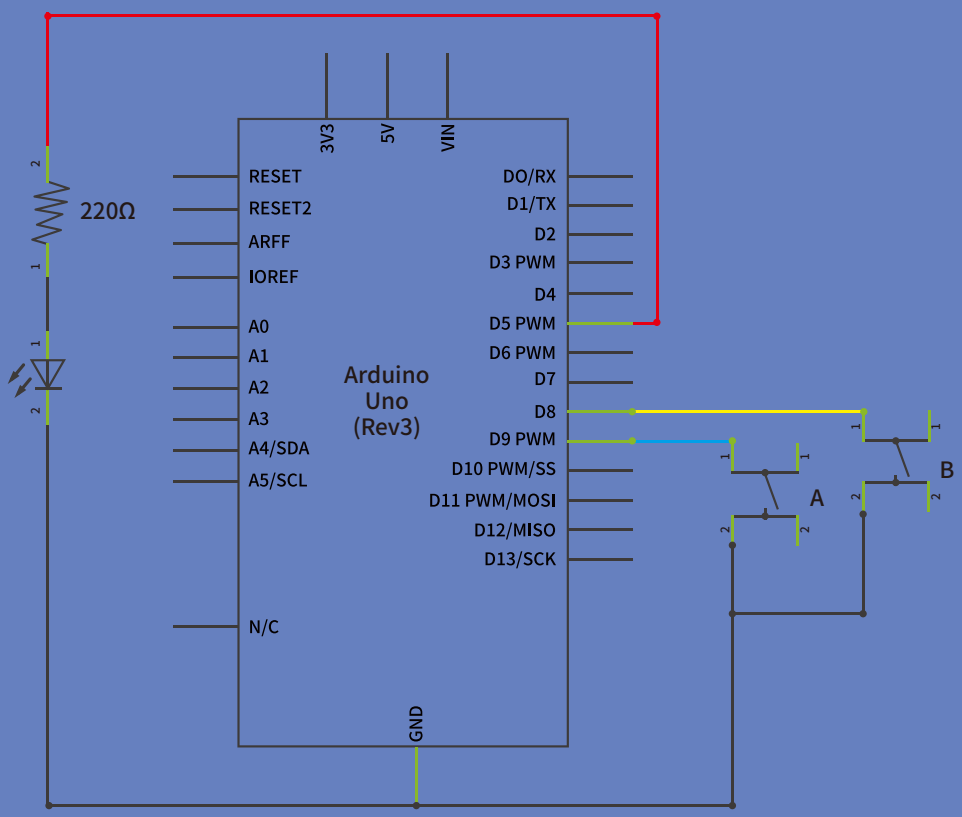
Schalter sind sehr einfache Bauteile. Wenn Sie einen Knopf drücken oder einen Hebel umlegen, verbinden sich zwei Kontakte miteinander, sodass elektrischer Strom fließen kann. Die kleinen Drucktaster, die wir in dieser Lektion benutzen werden, haben vier Kontakte, die zuerst kompliziert erscheinen.

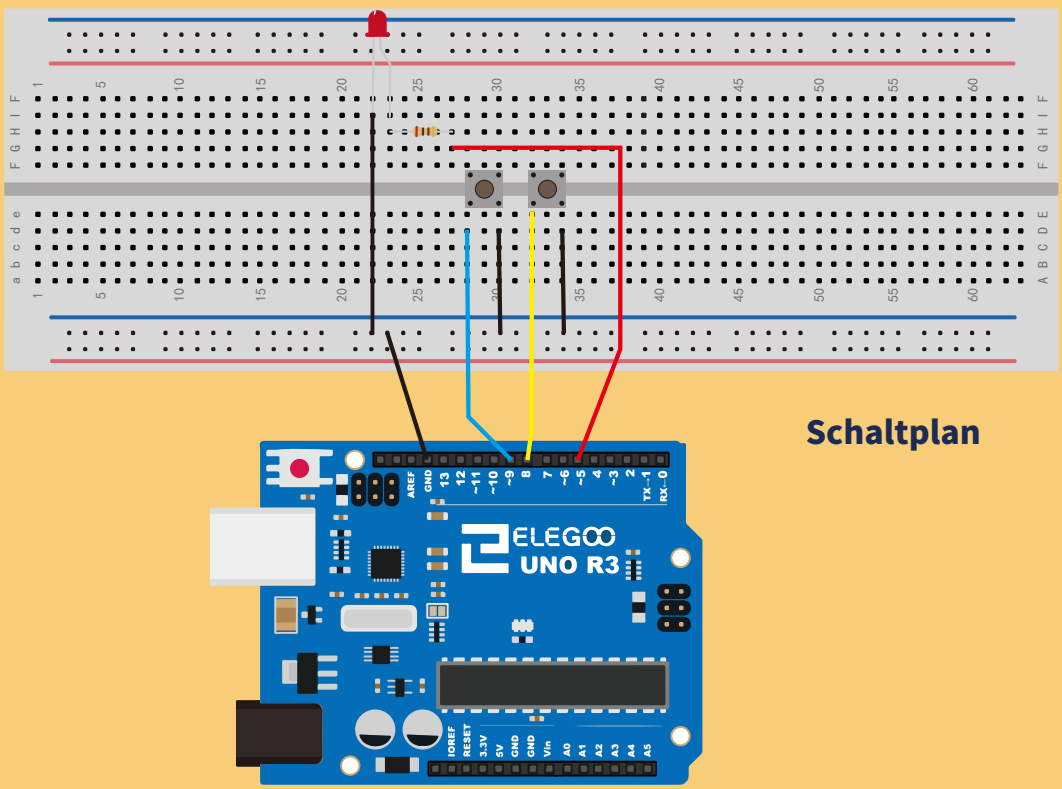




Eigentlich gibt es nur zwei richtige elektrische Verbindungen. Innerhalb des Schalters sind die Pins A und D sowie die Pins B und C miteinander verbunden.

Verbindungsschema:



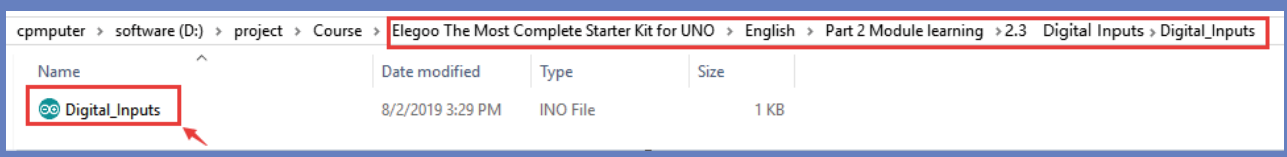


Schaltplan

- Obwohl die Schalter quadratisch sind, können sie nicht in jede Richtung aufgesteckt werden. Das bedeutet, dass die Pins richtig auf dem Breadboard platziert werden müssen, damit sie in die Mitte des Breadboards passen.
- Denken Sie daran, dass die LED den positiven Anschluss auf der links Seite hat.

Code

Bitte öffnen Sie das Programm:



Laden Sie den Sketch auf Ihr UNO Board. Durch Drücken des linken Schalters wird die LED eingeschaltet, ein Druck auf den rechten Schalter schaltet sie wieder aus.

Der erste Teil des Sketches definiert drei Variablen für die drei Pins, die benutzt werden. Der „ledPin“ ist der Ausgang für die LED, der „buttonApin“ verweist auf den Schalter, der näher oben am Breadboard liegt und der „buttonBpin“ steht für den unteren Schalter.

```
pinMode(buttonApin,INPUT_PULLUP);
pinMode(buttonApin,INPUT_PULLUP)
```

Die setup-Funktion definiert den ledPin als einen gewöhnlichen Ausgang (OUTPUT), wohingegen die beiden Schalterpins als Eingänge behandelt werden müssen. In diesem Fall setzen wir den pin-Modus auf „INPUT_PULLUP“:

Der pin-Modus INPUT_PULLUP bedeutet, dass der Pin als Eingang behandelt werden soll. Gleichzeitig bestimmt er, dass wenn nichts an den Eingang angeschlossen ist, der Standardwert für den Eingang „HIGH“ ist. In anderen Worten: Der Standardwert für den Pin ist HIGH, außer er wurde durch einen betätigten Schalter auf „LOW“ geschaltet.

Das ist der Grund, warum die Schalter mit GND verbunden sind. Wenn ein Taster gedrückt wurde, verbindet er den Eingangs-Pin mit GND, sodass der Eingang nicht länger HIGH geschaltet ist.

Da der Eingang im Grundzustand HIGH geschaltet und bei gedrücktem Schalter LOW ist, ist die Logik umgekehrt als eigentlich üblich. Darum werden wir uns in der loop-Funktion kümmern.

```
void loop()
{
if (digitalRead(buttonApin) == LOW)
{
digitalWrite(ledPin, HIGH);
}
if (digitalRead(buttonBpin) == LOW)
{
digitalWrite(ledPin, LOW);
}
}
```

In der loop-Funktion befinden sich zwei „if“-Statements (Bedingungsanweisungen). Für jeden Schalter eins. Beide if-Statements haben als Bedingung ein „digitalRead“, das den Zustand des jeweiligen Pins beobachtet.

Bedenken Sie, dass bei jedem Druck eines Tasters der Wert dessen Pins LOW sein wird. Wenn der Taster A gedrückt wird, wird buttonApin LOW sein. Das erste if- Statement (digitalRead(buttonApin) == LOW) wird damit erfüllt sein und die LED zum leuchten bringen (digitalWrite(ledpin, HIGH);).

Ähnlich funktioniert es beim zweiten Taster. Wenn Taster B gedrückt wird, erkennt dies das zweite If-Statement(digitalRead(buttonBpin) == LOW) und schaltet die LED auf LOW (= aus).

①If {} :①Die if-Anweisung sucht nach einer Bedingung und führt die fortlaufende Anweisung oder einen Satz von Anweisungen aus, wenn die Bedingung 'true' ist.

```
Syntax
if (Bedingung) {
  //Anweisung (en)
}
```

Parameter

Bedingung: ein boolescher Ausdruck (d. h. dieser wahr oder falsch sein).

②digitalRead():Liest den Wert von einem angegebenen digitalen Pin, entweder HIGH oder LOW.

Syntax

digitalRead(pin)

Parameter

pin: Die Arduino-PIN-Nummer, welche abgefragt werden soll

Ergebnis

HIGH oder LOW

==

[Vergleichsoperatoren]

Beschreibung

Vergleicht die Variable links mit dem Wert oder der Variablen rechts vom Operator.Gibt true zurück, wenn die beiden Operanden gleich sind.Bitte beachten Sie, dass Sie möglicherweise Variablen verschiedener Datentypen vergleichen können, dies jedoch zu unvorhersehbaren Ergebnissen führen kann. Es wird daher empfohlen, Variablen desselben Datentyps einschließlich des signed/unsigned Datentyps zu vergleichen.

Syntax

```
x == y; // ist true (wahr), wenn x gleich y ist und es ist false (falsch), wenn x nicht gleich y ist
```

Parameter

x: Variable. Zulässige Datentypen: int, float, double, byte, short, long.

y: Variable oder Konstante. Zulässige Datentypen: int, float, double, byte, short, long.