### 1. Einführende Informationen

Nun wollen wir einen Taster als Schalter verwenden, um LEDs zu steuern.

**Info: Der Taster**

Mit dem Arduino sind kleine Taster mitgeliefert werden. Normalerweise haben Taster nur zwei Anschlüsse, bei uns sind es vier! Jeweils zwei Anschlüsse sind miteinander verbunden! Die Anschlüsse, die verbunden werden, wenn du den Taster drückst, liegen jeweils auf derselben Seite.

Die LED ist an PIN 11, der Taster an PIN 7angeschlossen. PIN 11 musst du – wie bisher – als OUTPUT schalten. PIN 7 für den Taster ist aber keine Ausgabe, sondern eine Eingabe. Für einen Taster musst du den PIN als INPUT\_PULLUP deklarieren. Der Pin wird also als ein Eingang verwendet, der nach oben gezogen ist. Der Quellcode könnte z. B. so lauten:   
pinMode (Schalter, INPUT\_PULLUP);

Zur Abfrage, ob ein Taster gedrückt ist, wird der Funktion digitalRead verwendet.

digitalRead(Schalter) liefert LOW zurück, wenn der Taster gedrückt wird. Bei HIGH ist der Taster nicht gedrückt. (Das ist genau andersherum wie man vermuten könnte, liegt aber daran, dass bei gedrücktem Taster der Minuspol mit dem Eingang verbunden ist)

Den Zustand von digitalRead kannst du z. B. mit dem if-Befehl abfragen. Die Syntax für den if-Befehl lautet:

if (Bedingung)

{

/\* hier stehen die Befehle, die ausgeführt werden sollen, wenn die if-Bedingung erfüllt ist. \*/

}

Wenn bei gedrücktem Taster die Lampe leuchten soll, müssten unsere Befehle so lauten:

if (digitalRead(Schalter)==LOW)

{

digitalWrite(LED,HIGH);

}

Beachte, dass in der Programmiersprache C für Vergleiche == (also zwei Gleichheitszeichen verwendet werden müssen!)

**Achtung**

Wenn man den Taster einfach nur mit dem Mikrocontroller verbindet, dann liegt an dem Pin des Mikrocontrollers eine Spannung an, sobald der Taster gedrückt wird. Man kann sich das so vorstellen, als würden an dem besagten Pin ganz viele Elektronen herumschwirren. Wenn der Taster dann losgelassen wird, kommen keine neuen Elektronen mehr hinzu. Jetzt kommt der Knackpunkt. Die Elektronen sind dann immer noch da und entweichen nur ganz langsam. Der Mikrocontroller „denkt“ dann, dass der Taster nicht nur kurz gedrückt wird sondern dass er ganz lange gedrückt wird. Nämlich so lange, bis sich keine Elektronen mehr auf dem Pin aufhalten.

Dieses Problem kann man dadurch beheben, dass man den Pin über einen Widerstand mit ca. 1000 Ohm (1 K Ohm) erdet. Die Elektronen können dadurch recht schnell vom Pin abfließen und der Mikrocontroller erkennt, dass der Taster nur kurz „angetastet“ wurde.

### 2. Aufgabe

Schreibe ein Programm, das die LED an PIN 11 einschaltet, wenn du den Taster an PIN 7 drückst. Baue die Schaltung nach.

Erfasse den Sketch mit Variablen und speichere ihn ab.

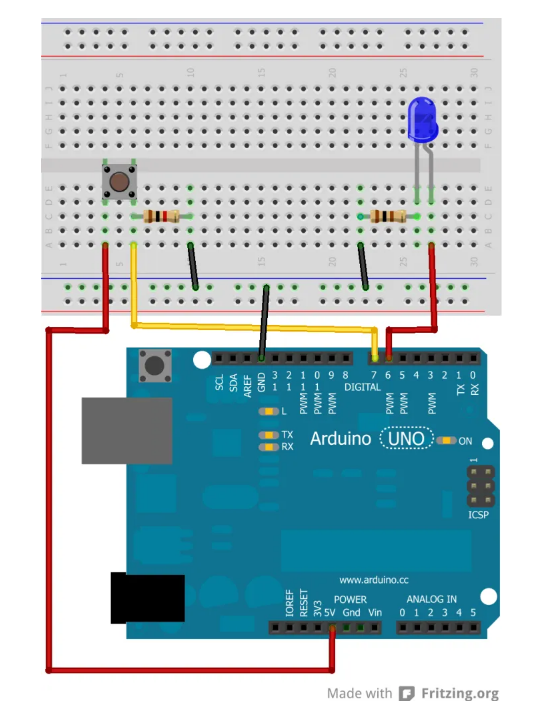
Notiere in der Tabelle entsprechende Kommentare zum Sketch.

Färbe alle Befehle rot, die den Taster programmieren.

### 3. Notwendiges Material

Breadboard, Arduino, eine LED mit Widerstand 220 Ohm, ein Widerstand mit 1 K Ohm, ein Taster, Kabel

### 4. Tipp: Schaltung



### 5. Tipp: Sketch

|  |  |
| --- | --- |
| int LEDblau=6; |  |
| int taster=7; |  |
| int tasterstatus=0; |  |
|  |  |
| void setup() |  |
| {pinMode(LEDblau, OUTPUT); |  |
| pinMode(taster, INPUT\_PULLUP);} |  |
|  |  |
| void loop() |  |
| {tasterstatus=digitalRead(taster); |  |
| if (tasterstatus == HIGH) |  |
| {digitalWrite(LEDblau, HIGH); |  |
| delay (5000); |  |
| digitalWrite(LEDblau, LOW);} |  |
| else |  |
| {digitalWrite(LEDblau, LOW); |  |
| } |  |
| } |  |