Interrupt - Licht ein- ausschalten mit Taster

Der Taster soll als Lichtein- und ausschalter funktionieren.

Der Interrupt soll jetzt bei FALLING ausgelöst werden, weil der Zustand des Taster von HIGH auf LOW „fällt“:

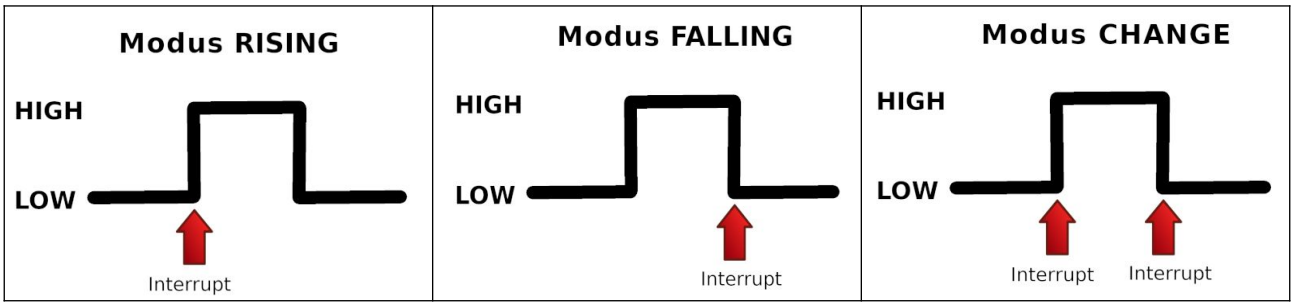
```
  attachInterrupt(digitalPinToInterrupt(TASTER), LEDSchalten, FALLING);
```

Der Taster wird dem Interrupt zugeordnet (`attachInterrupt`). Wenn der Taster betätigt wird, löst er den Interrupt aus. Der normale Programmablauf wird unterbrochen und die festgelegte Methode (Interrupt-Service-Routine) wird ausgeführt. Anschließend wird das Programm normal fortgesetzt.

```
attachInterrupt(digitalPinToInterrupt(TASTER), LEDSchalten, FALLING);
```

Der Taster löst den Interrupt aus, die Interrupt-Service-Routine `LEDSchalten` wird aufgerufen. Der Interrupt soll auf einen Wechsel des Tasterzustands (LOW oder HIGH) reagieren.
Es gibt verschiedene Ereignisse, die den Interrupt auslösen können:

RISING	der Interrupt wird ausgelöst wenn sich der Status von LOW zu HIGH ändert
FALLING	der Interrupt wird ausgelöst wenn sich der Status von HIGH zu LOW ändert
CHANGE	der Interrupt wird ausgelöst wenn sich der Status ändert



Der Taster muss zwingend am Pin 2 oder Pin 3 angeschlossen werden.

Zusätzlich muss die Variable, die eine Statusänderung anzeigt, als `volatile` definiert werden. Variable werden im RAM und temporär im Speicherregister gespeichert, bearbeitet oder verändert.

Es kann vorkommen, dass der Zustand der Variablen im Flashspeicher und im SRAM durch eine neue Wertzuweisung für kurze Zeit nicht übereinstimmen.
Das Schlüsselwort `volatile` weist das Programm an, die Variable immer aus dem Flashspeicher und nicht vom SRAM zu laden. Damit wird garantiert, dass der jeweils aktuelle Wert geladen wird.

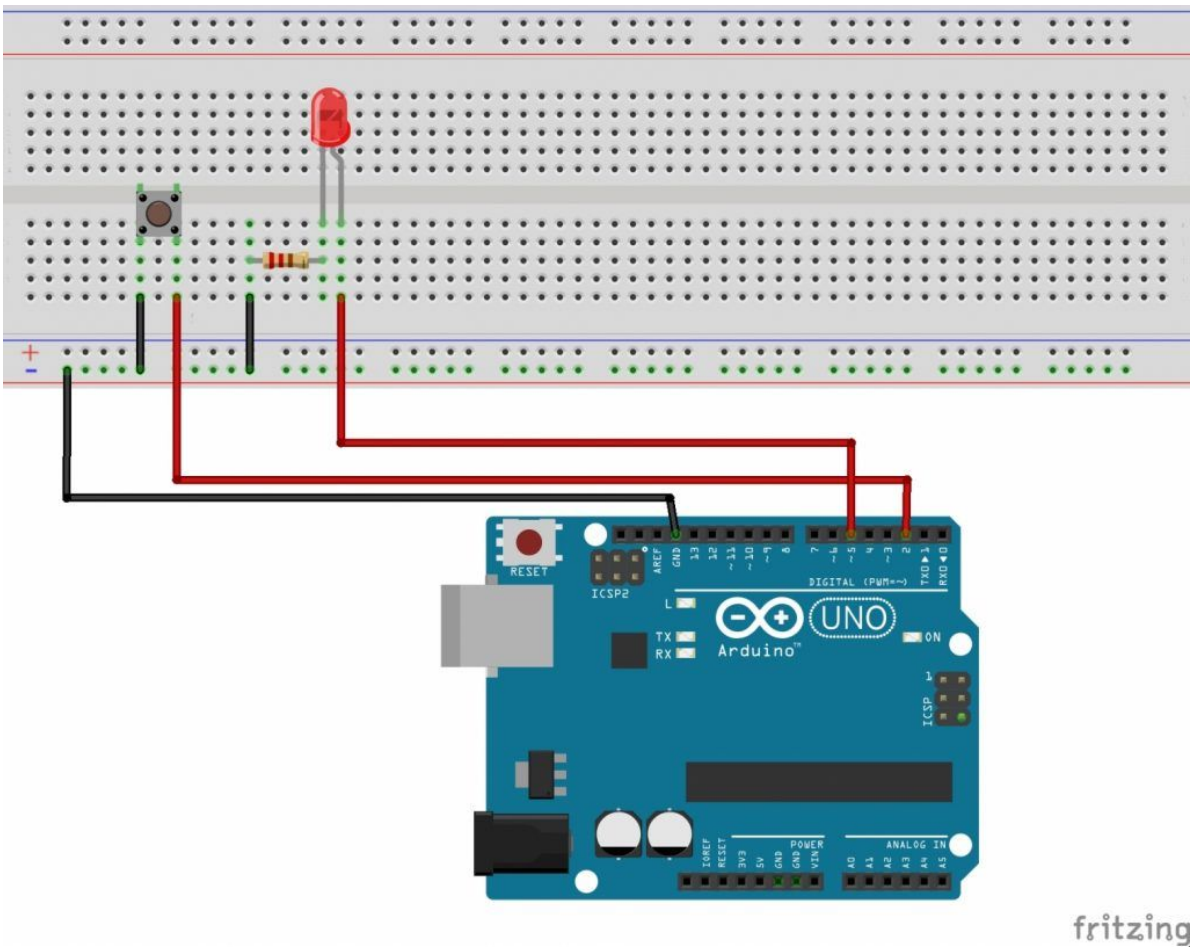
Außerdem sind bei der Programmierung einer Interrupt-Methode einige Regeln zu beachten:

- Der Programmteil muss so kurz wie möglich sein.
- Verwende kein `delay()`.
- Benutze auch kein `Serial.print()`.

Benötigte Bauteile:

- rote LED
- Widerstand > 100 Ω
- Leitungsdrähte
-

Baue die Schaltung auf.




```
define ROT 5
// nur Port 2 und 3 können mit
// attachInterrupt angesprochen werden
define TASTER 2

/*
  je nach Zustand der Variable TasterStatus ist die LED ein-
  oder ausgeschaltet
  beim Start des Programms ist sie ausgeschaltet
*/
 volatile bool TasterStatus = LOW;
```

```
void setup()
{
  pinMode(ROT, OUTPUT);

  // Vorwiderstand einschalten
  pinMode(TASTER, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(TASTER), LEDSchalten, FALLING);
}

void loop()
{
  // nichts zu tun
  // das Programm reagiert nur auf den Interrupt
}
```

In der Methode LEDSchalten muss der TasterStatus „umgedreht“ werden:

```
void LEDSchalten()
{
  TasterStatus = !TasterStatus;
  digitalWrite(ROT, TasterStatus);
}
```