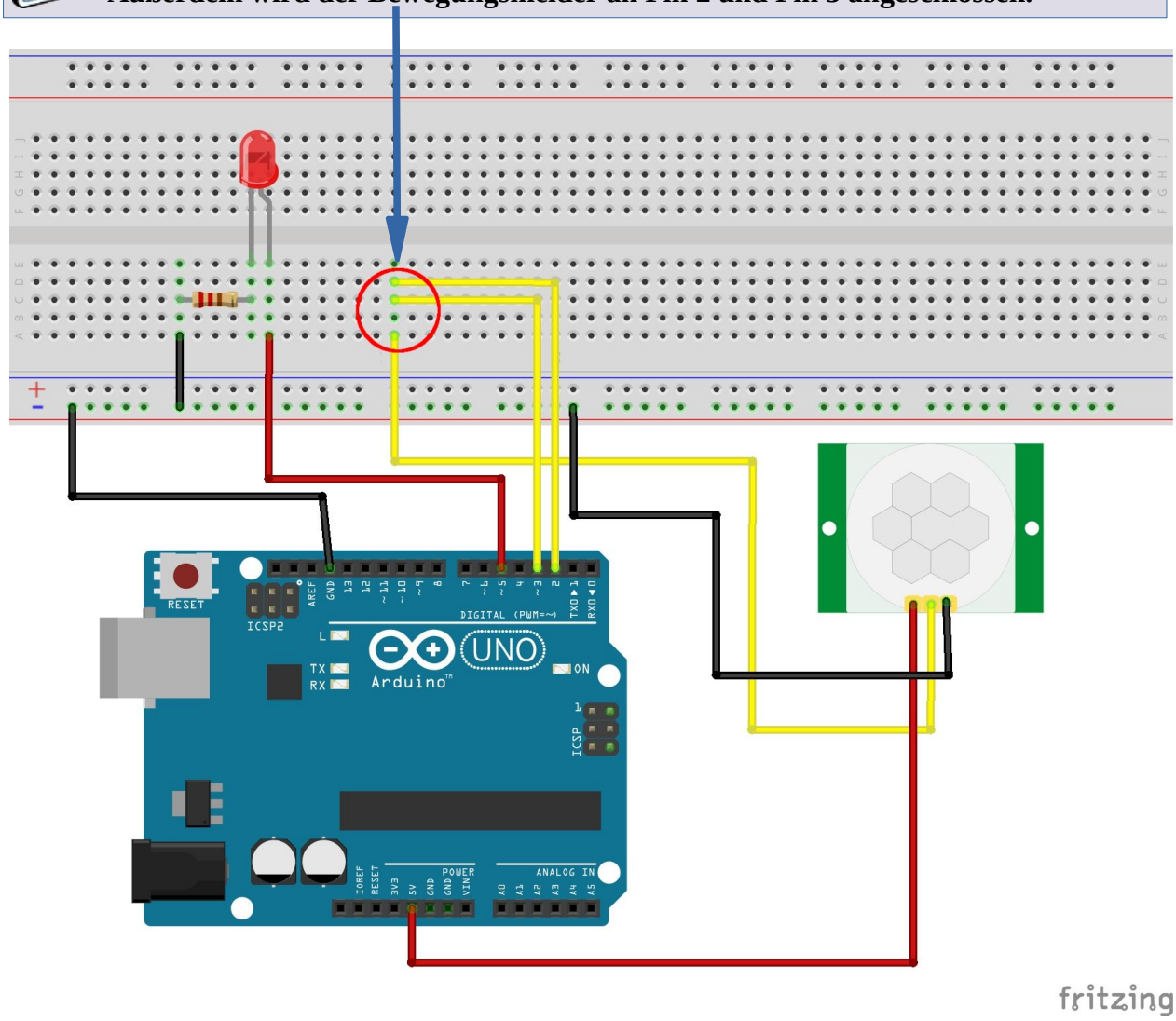
Licht ein- ausschalten mit Bewegungsmelder

Stelle durch Drehen nach rechts des linken Potentiometers ein längeres Ausgangssignal ein.

Benötigte Bauteile:

- Bewegungsmelder HC-SR501
- Widerstand > 100 Ω
- Leitungsdrähte
- LED

Beachte bei der Verkabelung, dass die beiden Potentiometer nach vorn zeigen. Außerdem wird der Bewegungsmelder an Pin 2 und Pin 3 angeschlossen.



Definiere die beiden Pins des Bewegungsmelders und den Pin der LED…

```
# define BEWEGUNG_EIN 2
# define BEWEGUNG_AUS 3
# define ROT 5
```

… und lege im setup-Teil den pinMode für die Bauteile fest.

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

  pinMode(BEWEGUNG_EIN, INPUT);
  pinMode(BEWEGUNG_AUS, INPUT);
```

Außerdem musst du für die Auslöser die Interrupt-Methoden festlegen:

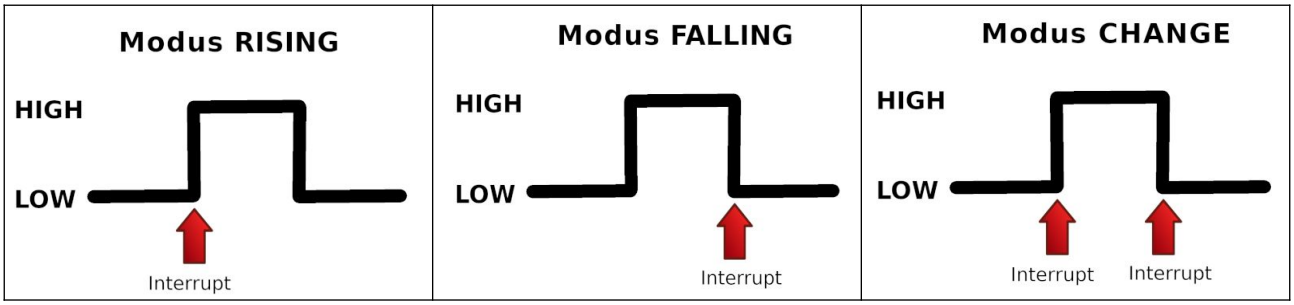
```
 /*
    wenn eine Bewegung registriert wird
    Signal ist HIGH -> RISING -> LEDEin
    wenn die Wartezeit abgelaufen ist
    Signal ist LOW -> FALLING -> LEDAus
  */
  attachInterrupt(digitalPinToInterrupt(BEWEGUNG_EIN), LEDEin, RISING);
  attachInterrupt(digitalPinToInterrupt(BEWEGUNG_AUS), LEDAus, 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()`.

Der loop-Teil bleibt leer, weil das Programm ausschließlich durch die Interrupts gesteuert wird.

Jetzt fehlen nur noch die Methoden `LEDEin()` und `LEDAus()`.

```
void LEDEin()
{
  . . .
}

void LEDAus()
{
  . . .
}
```