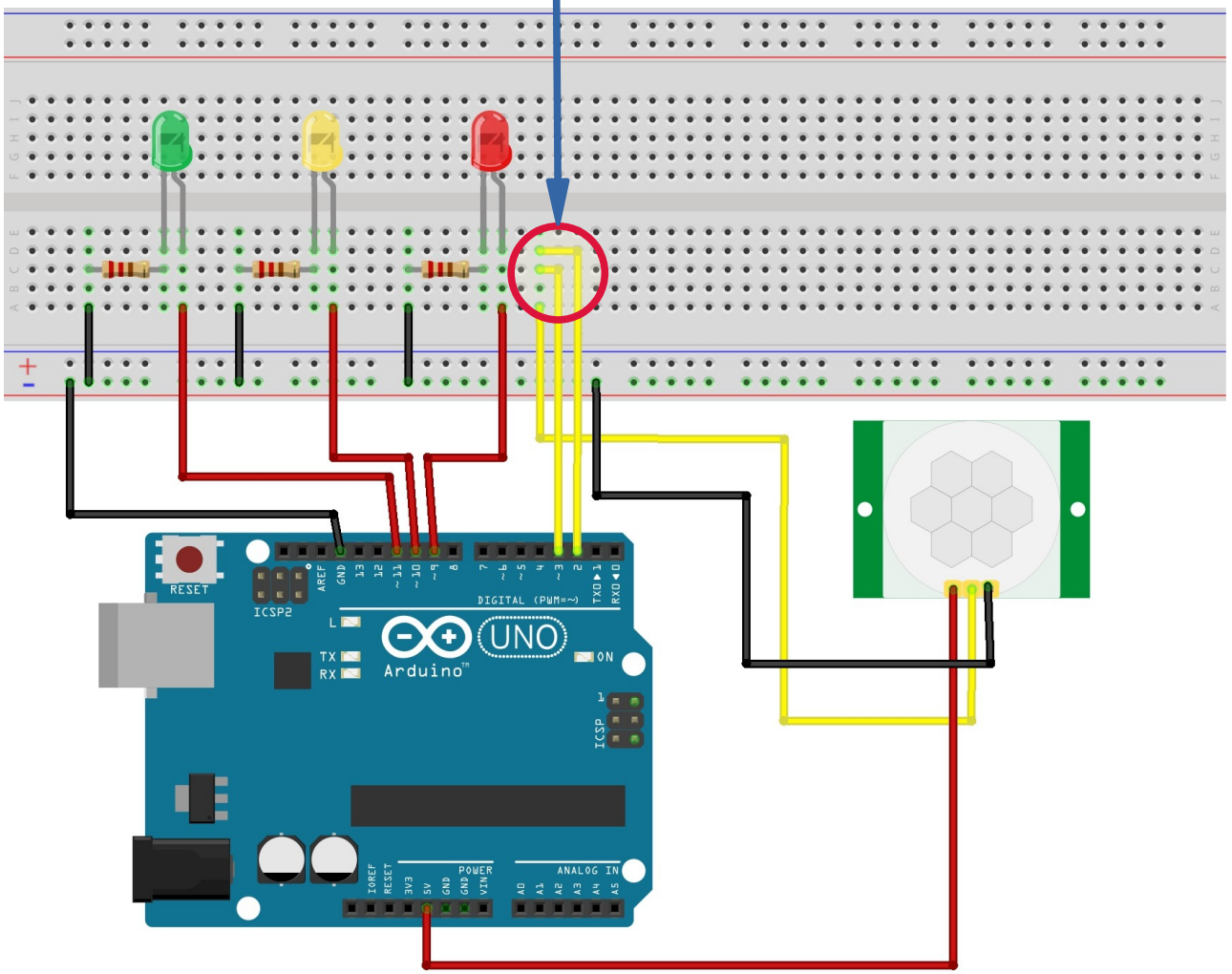
Interrupt - Blinkende LEDs mit Bewegungsmelder

Benötigte Bauteile:

- Bewegungsmelder HC-SR501
- 3 LEDs
- 3 Widerstände 220 Ω
- Leitungsdrähte

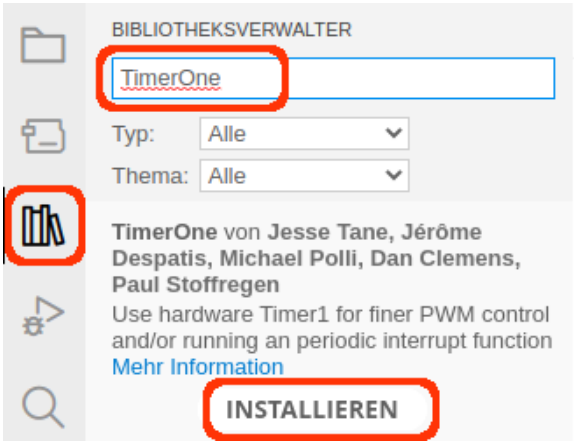
Baue die Schaltung auf.

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



Installiere die Bibliothek TimerOne:

Sketch → Bibliothek einbinden → Bibliotheken verwalten



Die gelbe LED soll dauerhaft leuchten. Die rote und die grüne LED sollen in regelmäßigen aber unterschiedlichen Rhythmus solange blinken, bis der Bewegungsmelder sie wieder ausschaltet.

Hier soll die Bibliothek TimerOne verwendet werden.

Sie kennt die Schlüsselwörter:

Timer1.initialize(Mikrosekunden);	Timer initialisieren, Mikrosekunden ist die Zeitspanne, nach der der Interrupt ausgelöst wird
Timer1.start();	Timer starten
Timer1.resume();	Timer fortsetzen
Timer1.stop();	Timer stoppen
Timer1.attachInterrupt(Methode);	Methode definieren, die nach Ablauf der Zeitspanne durch den Interrupt ausgelöst wird
Timer1.detachInterrupt();	Interrupt ausschalten
Timer1.pwm(Pin, Zeit);	Pin 9 oder 10 werden aktiviert 0 = aus, 1023 = immer an, Werte dazwischen erzeugen eine blinkende LED
Timer1.disablePwm(Pin);	Pin deaktivieren

Das Blinken der roten LED soll mit `Timer1.pwm()` realisiert werden, `Timer1.attachInterrupt()` sorgt für das Blinken der grünen LED.

Binde als Erstes die Bibliothek TimerOne ein und definiere die Variablen:

```
# include <TimerOne.h>

// Status der günen LED festlegen
// eine Änderung bewirkt das Blinken
bool Status = true;

# define ROT 9
# define GELB 10
# define GRUEN 11
# define BEWEGUNG_EIN 2
# define BEWEGUNG_AUS 3
```

Im setup-Teil wird zusätzlich zum oinMode der Bauteile der Timer initialisiert:

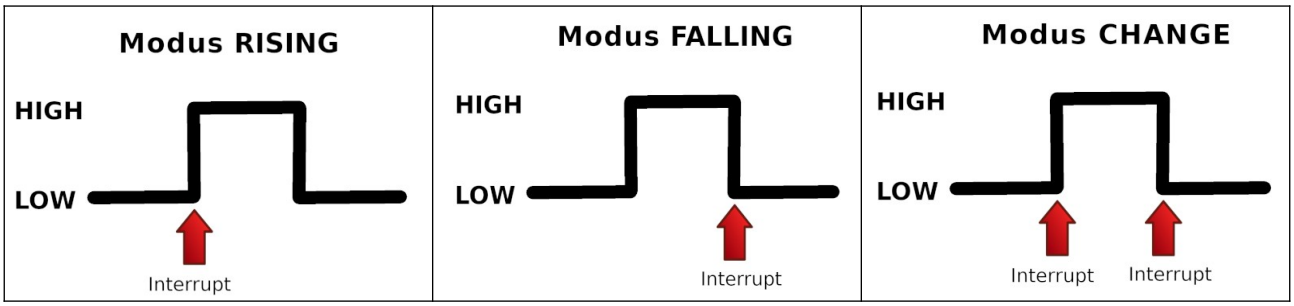
```
void setup()
{
  pinMode(ROT, OUTPUT);
  pinMode(GELB, OUTPUT);
  pinMode(GRUEN, OUTPUT);
  pinMode(BEWEGUNG_EIN, INPUT);
  pinMode(BEWEGUNG_AUS, INPUT);

  /*
    Zeit in Mikrosekunden
    1000000 = 1 Sekunde
    500000 = 0,5 Sekunden
  */
  Timer1.initialize(500000);

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

Der Bewegungsmelder wird beiden Interrupts zugeordnet (`attachInterrupt`). Wenn eine Bewegung registriert wird, löst sie den Interrupt aus. Der normale Programmablauf wird unterbrochen und die festgelegten Methoden (Interrupt-Service-Routine) werden ausgeführt. Anschließend wird das Programm normal fortgesetzt.

```
  attachInterrupt(digitalPinToInterrupt(BEWEGUNG_EIN), Start, RISING);
  attachInterrupt(digitalPinToInterrupt(BEWEGUNG_AUS), Stopp, FALLING);
```

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



Die Methode `Start()` ordnet den Interrupt mit `attachInterrupt()` zu und deaktiviert die rote LED mit `Timer1.disablepwm()`:

```
void Start()
{
  Timer1.attachInterrupt(BlinkenGruen);
  Timer1.start();
  Timer1.pwm(ROT, 500);
}
```

Die Methode `Stopp()` hält den `Timer` an, deaktiviert die rote LED mit `Timer1.disablePwm()` und entfernt den Interrupt mit `detachInterrupt()`:

```
void Stopp()
{
  Timer1.stop();
  Timer1.disablePwm(ROT);

  // möglicherweise bleibt die grüne LED an,
  // wenn das Signal FALLING gesendet wird
  // -> sicherheitshalber ausschalten
  digitalWrite(GRUEN, LOW);
  Timer1.detachInterrupt();
}
```

Jetzt musst du noch die Methode `BlinkenGruen()` erstellen, die über den Interrupt gesteuert wird:

```
void BlinkenGruen()
{
  Status = !Status;
  digitalWrite(GRUEN, Status);
}
```

Der loop-Teil:

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

Verändere die Werte für `Timer1.pwm()` oder die Werte für `Timer1.initialize()`.