Abgabe

Geben Sie die Lösung bis **Montag, 15. Mai** um 23:59 als PDF-Dokument in einem ZIP, inklusive aller Quelldateien, per E-Mail an Ihren Tutor ab. Die Abgabeformalitäten sind die Gleichen wie auf Blatt 1.

1 Automatische Ampelsteuerung (3 Punkte)

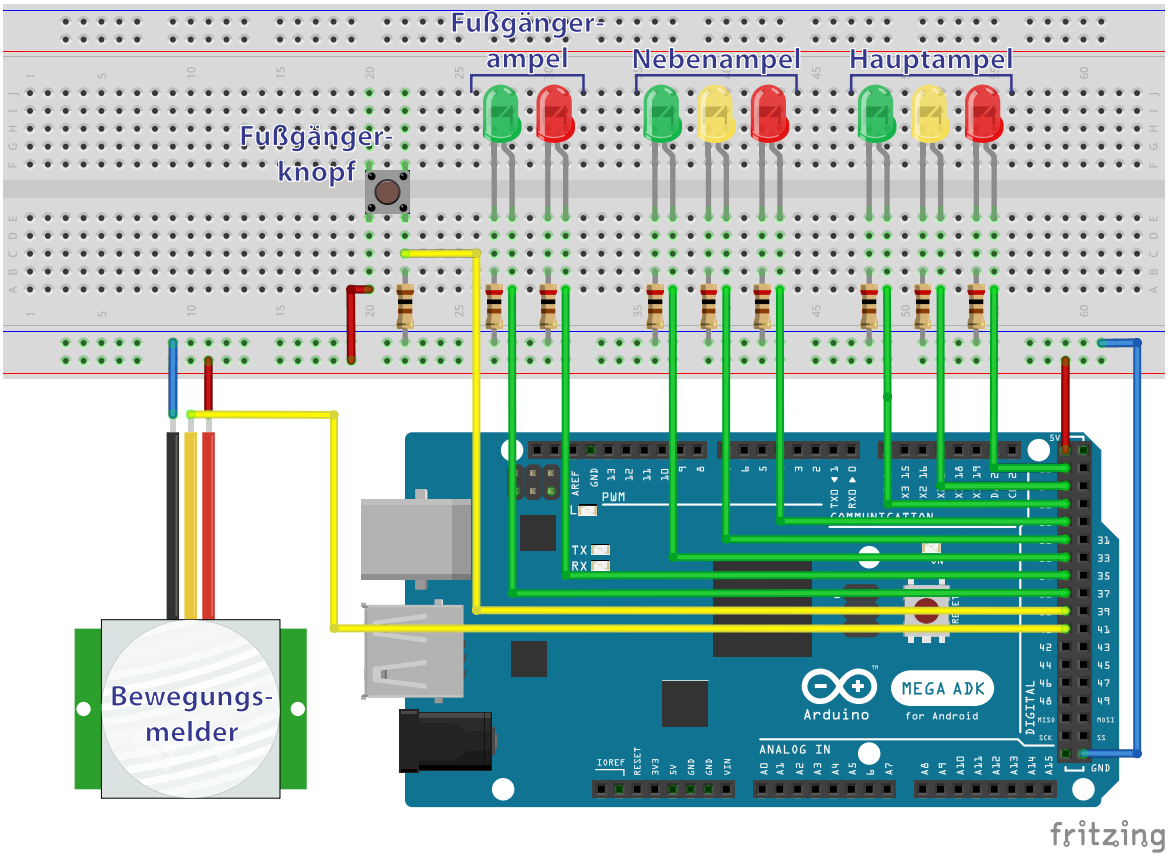
Auf diesem Übungsblatt geht es darum, den Einsatz von `delay(ms)`, `millis()` und `if () {}` an einem praktischen Projekt zu erproben.

Es geht um die Realisierung einer Ampel, die Automobilen anzeigt, ob sie fahren dürfen oder nicht.

Bauen Sie dazu zunächst die Ampelschaltung wie in der Abbildung gezeigt auf. Achten Sie dabei besonders auf die Polung der LEDs: Der kürzere Pin der LED ist dabei die Kathode und zeigt in Richtung der Erde (GND). Auf der Abbildung ist die Kathode durch den **nicht** abgeknickten Pin dargestellt. Außerdem ist es wichtig die LEDs nur zusammen mit einem Widerstand (ca. 200Ω) anzuschließen. Der Tasterausgang wird mit einem 10$k\Omega$-Widerstand gegen Ground definiert.

Zunächst soll die Hauptampel rein zeitgesteuert sein, die anderen beiden sind inaktiv. Ein Ampelübergang läuft in Zeitschritten von 1 Sekunde wie folgt ab: Rot auf Rot-Gelb, auf Grün, auf Gelb, und zurück auf Rot.

Zur Realisierung der Verzögerung sollen Sie dabei die Funktion `delay(ms)` einsetzen, die die in Millisekunden übergebene Zeit lang wartet.



2 Interaktive Ampelsteuerung (6 Punkte)

Die Ampel steht natürlich nicht an einer geraden Straße, sondern an einer Kreuzung. Die Nebenampel gehört zu einer Nebenstraße, die auf die Hauptstraße trifft. Die Fußgängerampel wird bei dieser Aufgabe noch nicht beachtet.
Implementieren Sie folgende Steuerung:

- Standardmäßig ist die Hauptampel grün und die Nebenampel rot
- Wird eine Bewegung registriert (der Ausgang des Bewegungsmelders wird HIGH), dann soll die Nebenampel auf grün und die Hauptampel auf rot schalten
- Hierbei gilt, dass niemals beide Ampeln gleichzeitig grün oder gelb sein dürfen. Bevor die zweite Ampel zu schalten beginnt, muss also die erste auf rot sein, nach einer Toleranzsekunde kann nun auch die zweite Ampel schalten
- Die Nebenampel darf nur eine Grünphase innerhalb von 20 Sekunden haben
- Die Grünphase der Nebenampel dauert 5 Sekunden
- wird nach einer Grünphase an der Nebenampel eine Bewegung registriert und der letzte Schaltvorgang ist noch keine 20 Sekunden her, so wird die Bewegung nicht ignoriert, aber erst nach Ablauf der Zeit ausgeführt. Mehrfache Bewegungen in einem Intervall werden jedoch ignoriert.

Ablaufbeispiel zum Verständnis: (Die Zahlen vorne stehen für die Zeit in Sekunden)
00 - Hauptampel grün, Nebenampel rot
01 - Bewegung an Nebenampel registriert, Hauptampel schaltet auf gelb, Zeitpunkt der letzten Bewegung wird in einer Variablen gespeichert
02 - Hauptampel schaltet auf rot
03 - Nebenampel schaltet auf gelb-rot
04 - Nebenampel schaltet auf grün
09 - Nebenampel schaltet auf gelb
10 - Nebenampel schaltet auf rot
11 - Hauptampel schaltet auf gelb-rot, Bewegung an Bewegungsmelder wird registriert, letzter Zyklus ist aber erst 10 Sekunden her, Bewegung wird gespeichert
12 - Hauptampel schaltet auf grün
21 - Hauptampel schaltet auf gelb aufgrund der zur Zeit 11 erkannten Bewegung
22 - ...

Es folgen Hinweise auf der nächsten Seite.

Hinweise:

Diese Steuerung lässt sich nicht mit `delay(ms)` realisieren, nutzen Sie hier die `millis()` Funktion, damit das Programm nicht blockiert wird und Eingaben weiterhin erkannt werden.

Nutzen Sie den Datentyp `unsigned long`, um Zeiten zu speichern.

Der Bewegungsmelder kann genauso wie ein Button gehandhabt werden, es muss jedoch beachtet werden, dass der Ausgang des Bewegungsmelder noch eine Weile HIGH ist, nachdem ein Signal erkannt wurde.

Beim Anschluss des Bewegungsmelders ist die Pinbelegung zu beachten. Unter der Kappe befindet sich eine Beschriftung, die zu befolgen ist. Eventuell ist ein Finetuning an den beiden Potentiometern notwendig.

Es empfiehlt sich den Status der Ampel in einer Variablen zu speichern. Also beispielsweise Status 1 (Initialstatus, der am Ende immer wieder erreicht wird) steht für Hauptampel grün und Nebenampel rot, Status 2 für den Fall, dass der Bewegungsmelder ein Fahrzeug registriert hat. Wenn ein Statuswechsel auftritt, dann muss ein Phasenwechsel durchlaufen werden (Schaltvorgang der Ampel), welcher in einer zweiten Variable hochgezählt wird.
Eine Phase schaltet also nur dann, wenn eine gewisse Zeit verstrichen ist und eine Phase erwartet wird. Dieser Vorschlag ist natürlich rein optional und Sie können es auf Ihre eigene Art und Weise lösen.

Beispiel:

Listing 1: Beispiel zu dem Schaltvorgang

```
if (phase == 1) {
        //do something ...

        if (millis() - last_switch >= 1000) { //Zeit (1000 ms) der Phase ist
            abgelaufen, nun erfolgt Wechsel
                last_switch = millis();
                phase = 2;
        }
} else if () ...
```

Hier sieht man ein Beispiel eines Phasenwechsels innerhalb eines Status. Am Ende des Phasenwechsels, wird auch der Status in den Initialzustand zurückgesetzt. Es empfiehlt sich hier, den Status 1 als *Hauptampel grün* und den Status 2 als den *kompletten Ampelzyklus beider Ampeln* zu definieren.

Eingaben durch den Bewegungsmelder werden außerhalb dieser If/Else-Struktur (aber in der loop!) abgefragt und in globalen Variablen gespeichert. Machen Sie sich hier Gedanken, wie Eingaben so abgefangen werden, dass sie sich nicht sofort auf einen Statuswechsel auswirken müssen, allerdings auch nicht ignoriert werden.

3 Fußgängerampel (3 Punkte)

Wenn Sie Teil 2 dieser Übung verstanden haben (daher auch die ausführliche Erklärung), werden Sie diese Erweiterung recht einfach umsetzen können.

1. Zwei weitere LEDs simulieren eine Fußgängerampel, der Taster dient für die Fußgänger zur Signalanfrage.
2. Die Fußgängerampel soll 5 Sekunden grün sein.
3. Tastereingaben werden jederzeit registriert, es darf jedoch nur einmal innerhalb von 20 Sekunden geschaltet werden. Doe Neben- und die Fußgängerampel nutzen beide die selbe Sperrzeit. Wenn also eine der beiden Ampeln auslöst, dann dürfen in den kommenden 20 Sekunden beide nicht mehr schalten
4. Es kann vorkommen, dass die Fußgängerampel nie schaltet, wenn ständig ein Signal von dem Bewegungsmelder kommt.

Ein paar Tipps:

Die Aufgabe basiert auf Nummer zwei, das heißt, der Aufbau der Steuerung ist hier sehr ähnlich.

Sie brauchen einen weiteren Status (gleiche Statusvariable wie in Aufgabe 2, für Fußgängerampel jedoch beispielsweise mit Wert 3) mit ihrem eigenen Phasendurchlauf für die Fußgängerampel.

Da nach Nebenampel, sowie Fußgängerampel eine Schaltsperre von 20 Sekunden verhängt wird, ist es garantiert, dass jeder Schaltzyklus immer komplett abgearbeitet werden kann und nicht von einem weiteren unterbrochen wird. Somit brauchen Sie sich bei strukturierter Implementierung der Aufgabe darüber keine Gedanken zu machen.

4 Anhang: Pin-Liste

Listing 2: Pindefinierung

```
int tl_1_red    = 22;
int tl_1_yellow = 24;
int tl_1_green  = 26;
int tl_2_red    = 28;
int tl_2_yellow = 30;
int tl_2_green  = 32;
int tl_3_red    = 34;
int tl_3_green  = 36;

int motionsensor = 40;
int button       = 38;

void setup { ... }

void loop { ... }
```