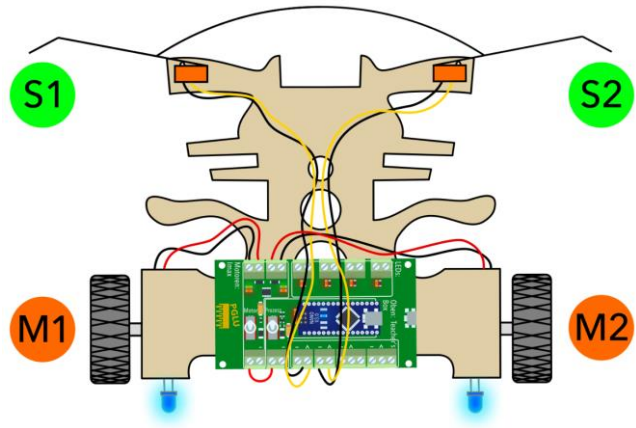


# 1

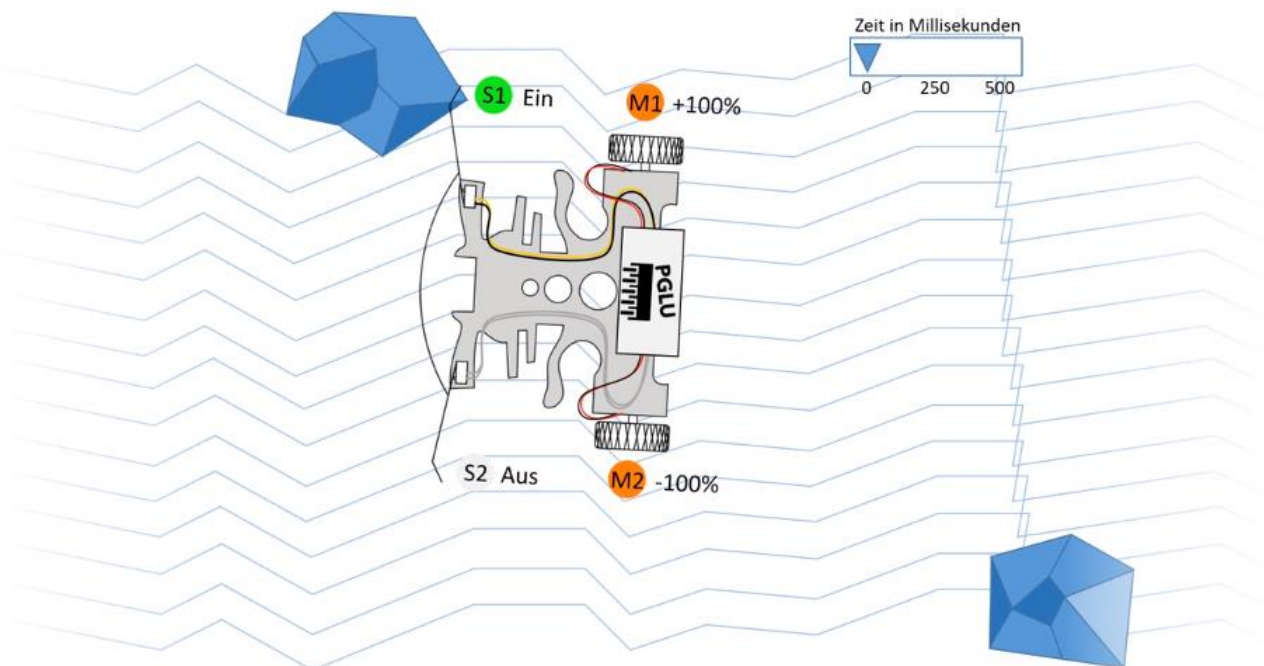


**Aufgabe** Beschreibe das Auto und das was es tut!

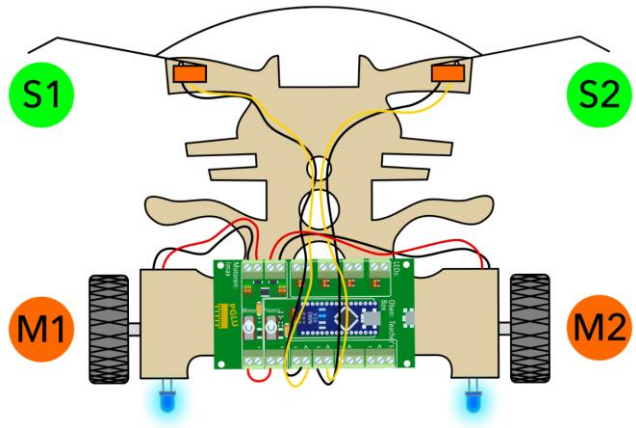
Schaue die Animation und beobachte was geschieht. Stelle dir vor, du müsstest das selbstfahrende Auto einer Person am Telefon erklären, um es ihr zu verkaufen.

Schreibe einen Text dazu!

- Aus welchen Komponenten besteht das Selbstfahrende Auto?
- Welche Einzelteile sind elektronisch, welche mechanisch?
- Wie werden die elektronischen Komponenten gesteuert, was ist der Auslöser für eine Steuerung?
- Wie bewegt sich das Auto, wenn es auf ein Hindernis trifft?
- Was macht der Timer?



# 2



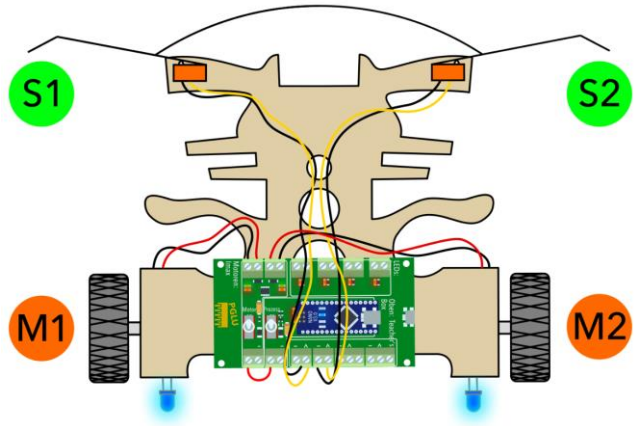
**Aufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung rückwärts. Beachte: der andere Motor läuft standardmässig rückwärts.

Setze diese Blöcke richtig in den Hauptloop ein!

- Wenn Sensor 2 ein Hindernis berührt (S2=Ein), dann lasse Motor 1 Rückwärts laufen (M1=-100%)
- Sonst (wenn Sensor 2 nichts berührt und S2=Aus), lasse Motor 1 vorwärts laufen (M1=100%)

The screenshot shows the PGLU programming environment with the 'Simulator' tab selected. The main workspace contains a 'Hauptloop: 100'000 mal pro Sekunde' block. Inside the loop, there is a 'Sensor 2' block with 'EIN' selected and 'prüfe standard'. This is followed by a 'wenn' (if) block. The 'wenn' block has two branches: one for 'Motor 1' set to 'auf -100 %' and another for 'Motor 1' set to 'auf 100 %'. The 'sonst' (else) branch is also visible. The left sidebar shows various action blocks like Sensor, Motor, LED, Pixel, and logic blocks like Frage, Ablauf, Zahl, Variable, Mathe, Struktur, Funktion, Loop, and Spezial.

# 3

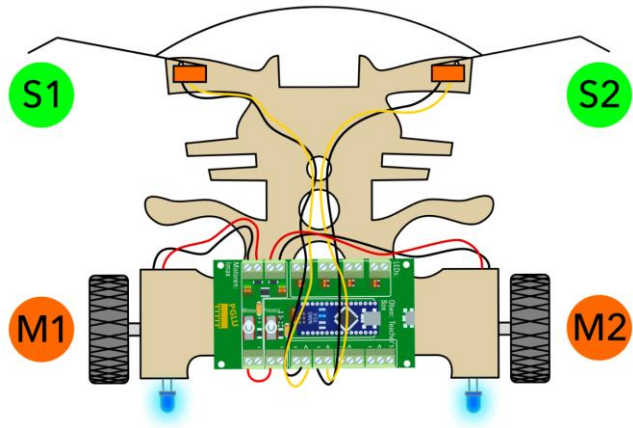


**Aufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung eine halbe Sekunde rückwärts

- Wenn Sensor 2 ein Hindernis berührt (S2=Ein), dann lasse Motor 1 rückwärts drehen (M1=-100%). **Tue dies während einer halben Sekunde, auch wenn die Berührung nur sehr kurz war!**
- Wenn Sensor 2 nichts berührt (S2=Aus), dann lasse Motor 1 vorwärts laufen (M1=100%)

The screenshot shows the PGLU programming environment with the 'Code' tab selected. The main workspace contains a 'Hauptloop: 100'000 mal pro Sekunde' block with a 'Blinkcode: kurz 1 lang 1' dropdown. Below it is a logic block: 'Sensor 2 = EIN prüfe standard'. This is followed by two 'setze Motor 1 auf' blocks: one set to '-100 %' and another set to '100 %'. A 'wenn/sonst' structure is used to connect the sensor check to the motor speed settings. A 'pausiere 500 ms' block is placed after the 'wenn' branch. The left sidebar shows various tool categories: Aktion, Logik, Frage/Ablauf, Zahl, Variable, Mathe, Struktur, Funktion, and Spezial. Yellow lines connect the 'Sensor 2' block to the 'Sensor 2' dropdown in the code, and the 'wenn/sonst' block to the 'setze Motor 1 auf' blocks.

# 4



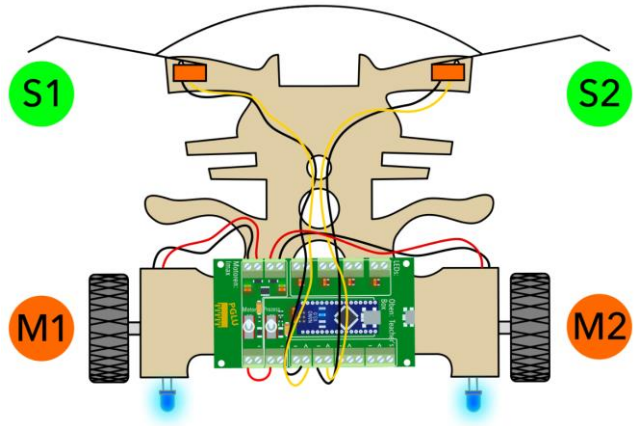
**Erweiterungsaufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung eine halbe Sekunde Rückwärts, mit LED-Blinker

- Wenn Sensor 2 nichts berührt (S2=Aus), soll M1 mit voller Geschwindigkeit vorwärts drehen (M1=100%).
- Wenn Sensor 2 auf ein Hindernis trifft (S2=Ein), soll M1 mit vollem Tempo während einer halben Sekunde rückwärts drehen (M1=-100%). Während dies geschieht soll LED1 3x aufblinken.

The screenshot shows the PGLU programming environment with the following components:

- Header:** PGLU logo, tabs for Grafik, Code, Simulator, and Programme.
- Left Panel (Action/Logik):**
  - Aktion:** Sensor, Motor, LED, Pixel.
  - Logik:** Frage, Ablauf, Zahl, Variable, Mathe, Struktur, Funktion, Loop, Spezial.
- Main Code Area:**
  - Loop:** "Hauptloop: 100'000 mal pro Sekunde" with "Blinkcode: kurz 1" and "lang 1".
  - Sensor Check:** "Sensor 2" block with "EIN" and "prüfe standard".
  - Motor Control:** "setze Motor 1 auf -100 %" and "setze Motor 1 auf 100 %".
  - LED Control:** "setze LED 1 auf EIN", "pausiere 100 ms", "setze LED 1 auf AUS", "pausiere 100 ms".
  - Logic:** "wenn" and "sonst" blocks.

# 5



**Erweiterungsaufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung eine halbe Sekunde rückwärts – Version «nicht blockierend»

Dieses Programm hat genau den gleichen Effekt, wie dasjenige in Aufgabe 3. Der Unterschied ist, dass nie ein Programmblock «pausiere-500-ms» den Programmablauf stoppt. Das ist gut, wenn du eine Lichtanimation parallel zur Fahrfunktion programmieren möchtest.

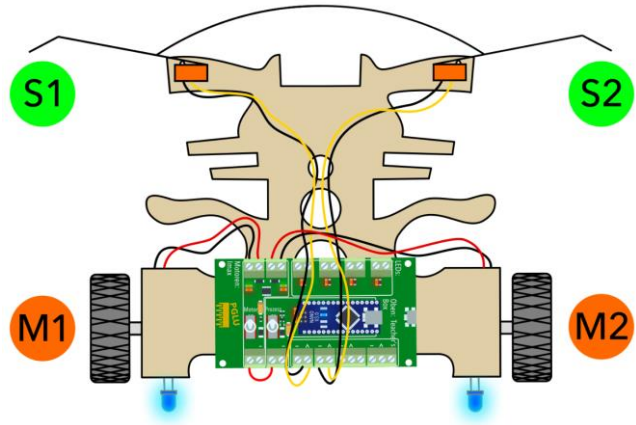
Tip 1: Zeit-in-ms gibt die Anzahl Millisekunden zurück, die seit dem Einschalten des Mikrocontrollers verstrichen sind.

Tip 2: Verstehe [workshop.pglu.ch](http://workshop.pglu.ch) > Sketch > Zeit-in-ms

The screenshot shows the PGLU IDE interface with the following code blocks:

- Vor Hauptloop: 1x**
  - schreibe Zeitstempel Sensor 2 = -501
- Hauptloop: 100'000 mal pro Sekunde**
  - setze Motor 1 auf 100 %
  - wenn Sensor 2 = EIN prüfe standard
    - schreibe Zeitstempel Sensor 2 = Zeit in ms
    - setze Motor 1 auf -100 %
    - wenn Zeit in ms < Zeitstempel Sensor 2 + 500
      - sonst

# 6



## Aufgabe Das Hauptprogramm «Selbstfahrendes Auto»

Programmiere das Selbstfahrende Auto, so dass es sich gleich verhält, wie du es in der Animation beobachtet hast! Verwende deine Erkenntnisse aus den Übungen 1-3. Füge noch blinkende LEDs aus Übung 4 ein!

Falls du die Erweiterungsaufgabe 5 erfolgreich gelöst hast, kannst du die erweiterte Version mit zwei Variablen «Timer 1» und «Timer 2» ausprobieren. Die Lösung dazu findest du in Variante 3.

A screenshot of the PGLU programming environment. The interface is divided into several sections: 'Grafik', 'Code', 'Simulator', and 'Programme'. The 'Code' section is active, showing a block-based program. The program starts with a 'Hauptloop: 100'000 mal pro Sekunde' block. Below this, there are two 'pausiere 500 ms' blocks. The first 'pausiere' block is connected to a 'wenn' block. The 'wenn' block has two paths: one leading to another 'wenn' block and another leading to a 'sonst' block. The second 'pausiere' block is connected to the 'sonst' block. Below the 'wenn' blocks, there are two 'Sensor' blocks. The first 'Sensor' block is connected to the 'wenn' block. The second 'Sensor' block is connected to the 'sonst' block. At the bottom, there are four 'setze' blocks. The first two 'setze' blocks are connected to the 'wenn' block, and the last two 'setze' blocks are connected to the 'sonst' block. The 'setze' blocks are for 'Motor 2' and 'Motor 1', with values of '-100 %' and '100 %' respectively.