



Aufgabenblatt 1 (Arduino Basics)

Aufgabe 1

Starten Sie die Entwicklungsumgebung und erstellen Sie eine neue Arduino-Datei. Übernehmen Sie den Code für die Definitionen und die Setup-Datei.

Erstellen Sie für die folgenden Aufgaben jeweils eine eigene (lokale) Funktion:

- Vorwärtsfahren
- Rückwärtsfahren
- Rechts drehen
- Links drehen
- Anhalten

Übersetzen Sie den Code und laden Sie den Code auf den Arduino. Dazu wird zunächst der Code kompiliert (Überprüfen) und dann hochgeladen. Um den Code auf den Arduino zu laden, muss er mit einem USB-Kabel mit dem Rechner verbunden werden.

Aufgabe 2

Mit dem Befehl

```
delay(1000);
```

kann eine Zeitverzögerung in Millisekunden eingebaut werden. Im obigen Beispiel ist es eine Sekunde, die der Arduino wartet, bevor er den nächsten Befehl ausführt.

Erstellen Sie in der loop()-Funktion folgenden Ablauf:

- 1 Sekunde lang vorwärtsfahren
- 1 Sekunde lang rückwärtsfahren
- 1 Sekunde lang rechts drehen
- 1 Sekunde lang links drehen

Wiederholen Sie die obige Sequenz dreimal und stoppen Sie danach das Fahrzeug. Mit dem Befehl

```
exit(0);
```

kann die loop()-Dauerschleife abgebrochen werden (return reicht nicht, da die loop()-Funktion immer wieder neu aufgerufen wird).

Beobachten Sie das Fahrzeug bei der Ausführung der Befehle. Was kannst über die Wiederholgenauigkeit festgestellt werden?

Aufgabe 3

Die beiden Enable-Kontroll-Pins können aber auch verwendet werden, um den Spannungspiegel einzustellen. Damit lässt sich die Geschwindigkeit der Motoren einstellen. Dazu wird der Pin analog beschrieben. Da der zugehörige Speicher eine Größe von einem Byte ohne Vorzeichen hat, sind Werte von 0 bis 255 möglich. Definieren Sie dazu einen eigenen Datentypen u8:



Robotik

Einstieg in das Arbeiten mit dem Arduino Aufgabenblatt L1

```
typedef unsigned char u8;
```

Damit unterschiedliche Geschwindigkeiten des Motors eingestellt werden können, müssen die ENA- und ENB-Pins analog beschrieben werden.

```
analogWrite(ENA, <u8-wert>)
```

Schreiben Sie dafür eine neue Funktion

```
setSpeed(u8 speed)
```

in der nur die beiden Befehle analogWrite mit dem übergebenen Parameterwert aufgerufen werden. Um Doppelungen zu vermeiden, müssen die digitalWrite-Befehle für ENA und ENB in den bereits erstellten Funktionen jeweils gelöscht werden.

Ändern Sie die Funktionen so ab, dass sie mit einem Byte-Wert parametrisiert werden können.

Ändern Sie die Aufgabe 2 von oben so ab, dass das Fahrzeug unter Verwendung der umgearbeiteten Funktionen mit langsamem Tempo geradeaus fährt und anschließend langsam dreht.

Beispiel: analogWrite(ENA, 100); Dabei muss beachtet werden, dass der Motor nicht mit jedem kleinen Wert anläuft und je geringer der eingestellte Wert ist, desto größer wirkt sich die Trägheit des Motors aus.

Aufgabe 4

Speichern Sie die Aufgabe 3 als Aufgabe 4 (Kopie erstellen) und erstellen Sie dann eine eigene Bibliothek für die Funktionen der Motoransteuerung (die Funktionen von Aufgabe 3). Löschen Sie die Funktionen der Motorsteuerung in Aufgabe4.ino und importieren Sie sie in die aktuelle Programmdatei (Aufgabe4.ino). Testen Sie die Anwendung erneut.

Hinweise: eigene Bibliotheken müssen als C++-Dateien zusammen mit ihrer zugehörigen Headerdatei erstellt werden, zur Definition der Pins und der Funktion Serial müssen die folgenden Standard-Arduino-Bibliotheken eingebunden werden: <Arduino.h> und <SoftwareSerial.h> in die Header-Datei eingebunden werden. Diese beiden Dateien müssen in den libraries-Ordner im lokalen Sketchbook in einem Ordner erstellt werden, der ebenso heißt wie die Bibliothek. Beispiel hier:

C:\Users\<username>\Documents\Arduino\libraries\motormovement\motormovement.h

C:\Users\<username>\Documents\Arduino\libraries\motormovement\motormovement.cpp

Zusatzaufgabe

Lassen Sie das Auto dreimal ein Rechteck fahren mit einer Dauer von 1 Sekunde für die Geradeausfahrt.

Hinweis: dazu müssen Sie zunächst ermitteln, wie lange die Verzögerung sein muss, damit das Fahrzeug genau eine 90-Grad-Drehung macht.