

The Circuit Navigation System for hobby projects

This document describes how a simulation program that makes it much easier for us to successfully implement hobby projects. Or at least of all projects with own electrics.
It takes the construction of a simple robot as an example project for getting started with hobby electrics.

Content

- 1: "Why simulate?"2
 - 1.1 Mechanics.....2
 - 1.2 Informatics.....3
 - 1.3 Elektrics.....4
- 2. The Circuit Navigation System.....5
 - 2.1 a simple example: light emitting diode.....5
 - 2.2 an advanced example: switching controller.....7
- 3. Conclusion.....8



Image 1: The logo of the Circuit Navigation System

1: "Why simulate?" Example of a robot project

First of all, we should ask ourselves: Do we really need a circuit simulation for a hobby project? After all, we often just want to get started, build something for real, and not spend so much time on computer simulations. Let's make an example, a typical hobby project for the start into robotics is the cleaning robot.



Abbildung 2: selbstgebauter Putzroboter in Aktion.

1.1 Mechanics

The mechanics of this cleaning robot consists of a wooden board, to which one has screwed a wheel with an electric motor on the left and right side, a third wheel for stability and two feather dusters in front, which sweep the dirt up. The mechanics are quickly built here:

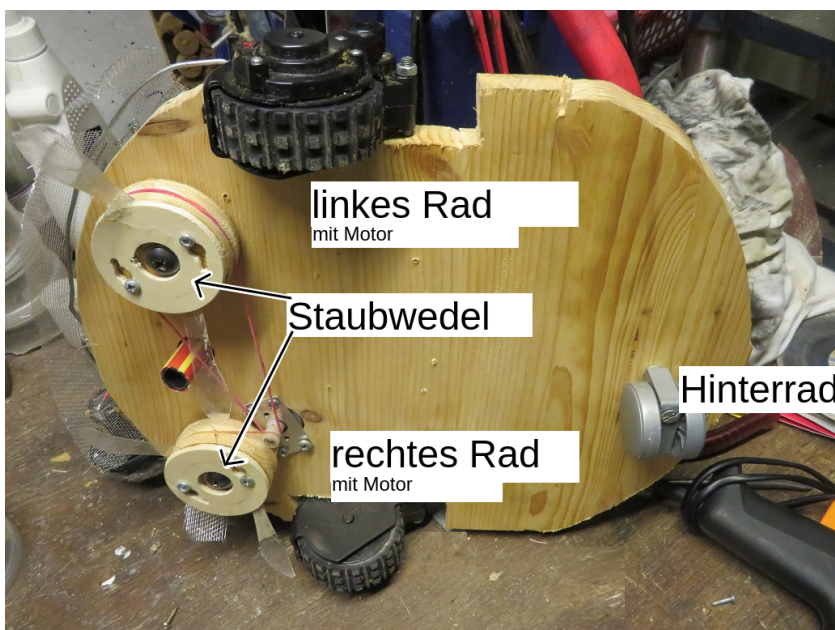
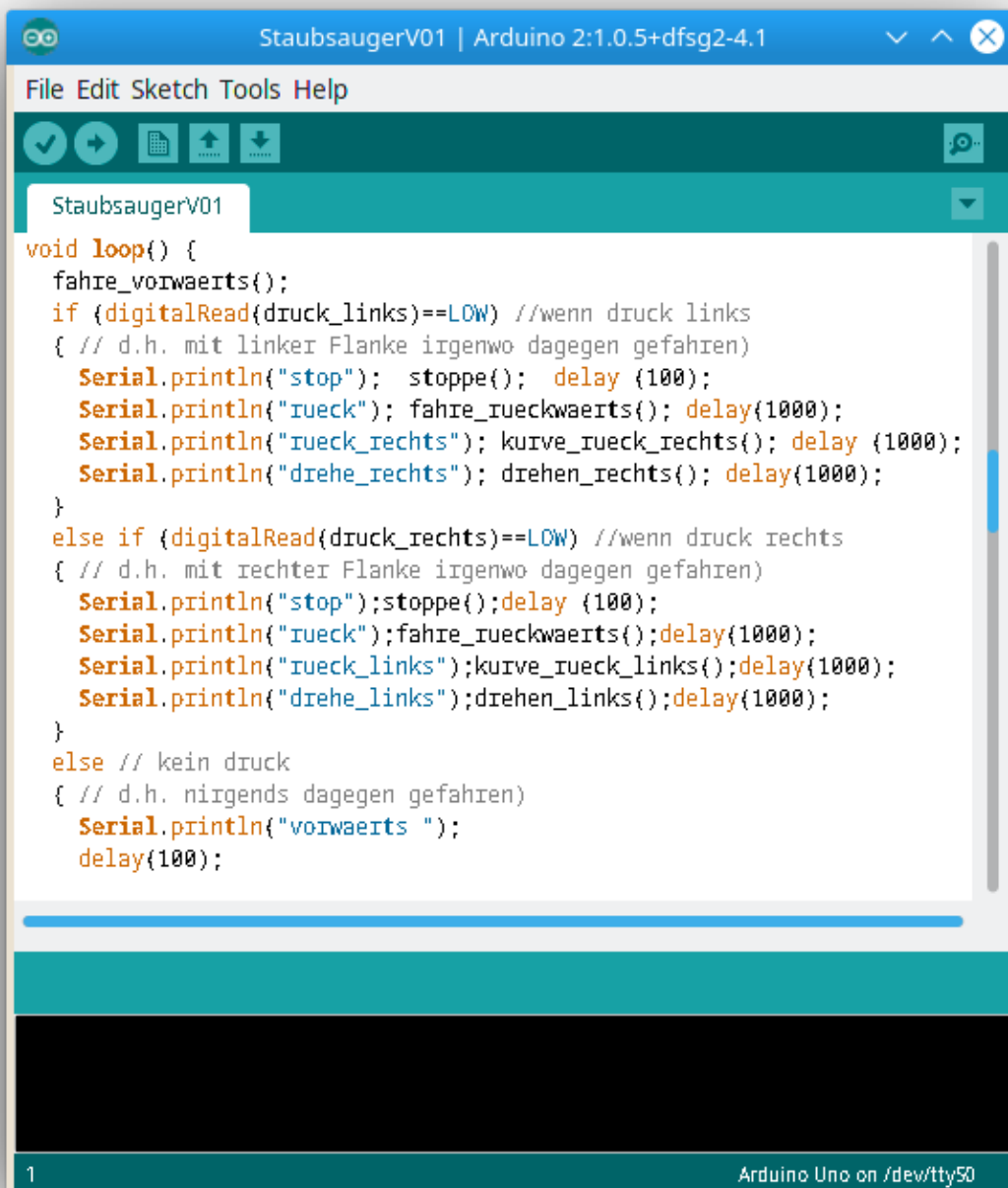


Abbildung 3: Mechanik des Putzroboters

1.2 Informatics

The second part is a control system for the robot. The program of a simple cleaning robot is not that complicated. It just drives straight ahead until it bumps into something. Then it turns around and drives in some other direction. And it does this until it has been everywhere and cleaned all the places. And that's why a cleaning robot like that one is a good project for getting started in robotics: You can write a simple program with little effort, and it works to some extent. Or - if you feel like it - you can improve the program so that the robot cleans all parts of the room in the shortest possible time. We can simply try out which programs work, and learn programming along the way.



```
void loop() {
  fahre_vorwaerts();
  if (digitalRead(druck_links)==LOW) //wenn druck links
  { // d.h. mit linker Flanke irgendwo dagegen gefahren)
    Serial.println("stop"); stoppe(); delay (100);
    Serial.println("rueck"); fahre_rueckwaerts(); delay(1000);
    Serial.println("rueck_rechts"); kurve_rueck_rechts(); delay (1000);
    Serial.println("drehe_rechts"); drehen_rechts(); delay(1000);
  }
  else if (digitalRead(druck_rechts)==LOW) //wenn druck rechts
  { // d.h. mit rechter Flanke irgendwo dagegen gefahren)
    Serial.println("stop");stoppe();delay (100);
    Serial.println("rueck");fahre_rueckwaerts();delay(1000);
    Serial.println("rueck_links");kurve_rueck_links();delay(1000);
    Serial.println("drehe_links");drehen_links();delay(1000);
  }
  else // kein druck
  { // d.h. nirgends dagegen gefahren)
    Serial.println("vorwaerts ");
    delay(100);
  }
}
```

Abbildung 4: Die Programmschleife zum Steuern des Putzroboters

1.3 Electrics

But for the robot to move at all, it also has to be wired together.

The control signals from the microcomputer have to become stronger currents to drive the motors. For that, we need driver circuits. And a status display with LEDs, whether the dust bag is full, would also be nice. But for the driver circuits as well as for the status display we have to select components and wire them correctly to each other.

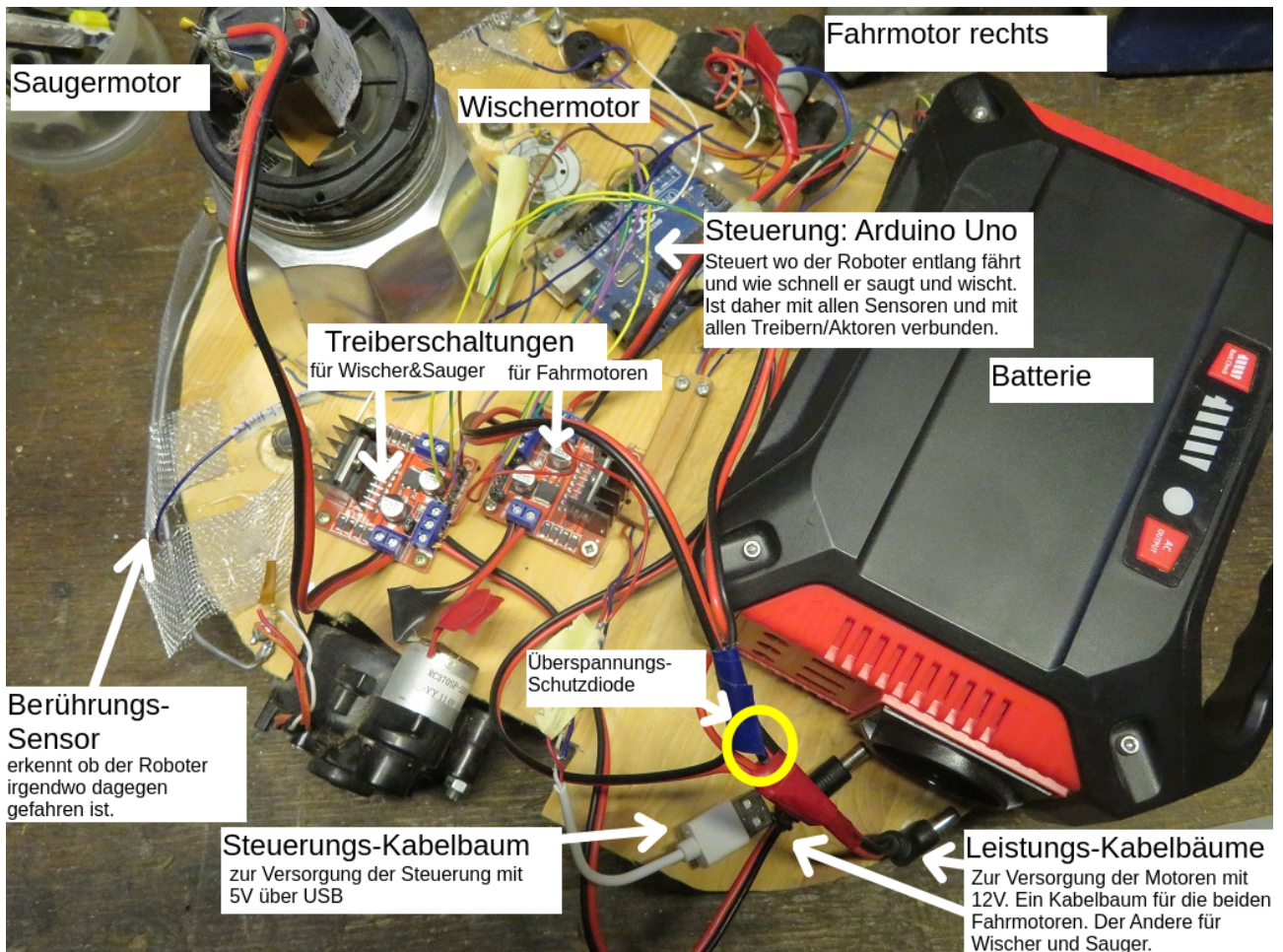


Abbildung 5: Elektrischer Aufbau des Putzroboters. Mit Beschreibung der meisten Bauteile.

From this picture we can see that setting up the electrics correctly in this project is a bit more difficult than the other two sections before.

Here, all of these components need to be wired together. And they have to be wired correctly. Because a mistake in the electrical system often leads to some component being overloaded and burning out, and then the project doesn't work, but just causes frustration. So we cannot just try out what works, because that usually goes wrong. Unless we are very lucky. If we want to avoid unnecessary frustration here, then we should simulate beforehand:

- What series resistor do I need in front of an LED so that it shines brightly, but not "too brightly" that it burns out?
- Does the driver circuit work with this transistor, or should I use a different one?

2. The Circuit Navigation System

Such questions as above decide whether a project works.

For this we should be able to see inside the circuit. See what is happening at each component.

The Circuit Navigation System shows us exactly that: It shows us as an animation video how much current flows through each component and what voltage is applied to each component.

2.1 a simple example: light emitting diode

Let's start with a status display with an LED. For such an LED to light up, a current must flow through the diode. The current must not be too small, otherwise it will not light up or only so weakly that it cannot be seen at all. But it must also not be too large, otherwise the LED will burn out. Most LEDs have a maximum limit of about 20mA.¹

In the following image we see that at 9V and a series resistor of 100Ω a current of 82mA would flow through the LED. So the diode would burn out:

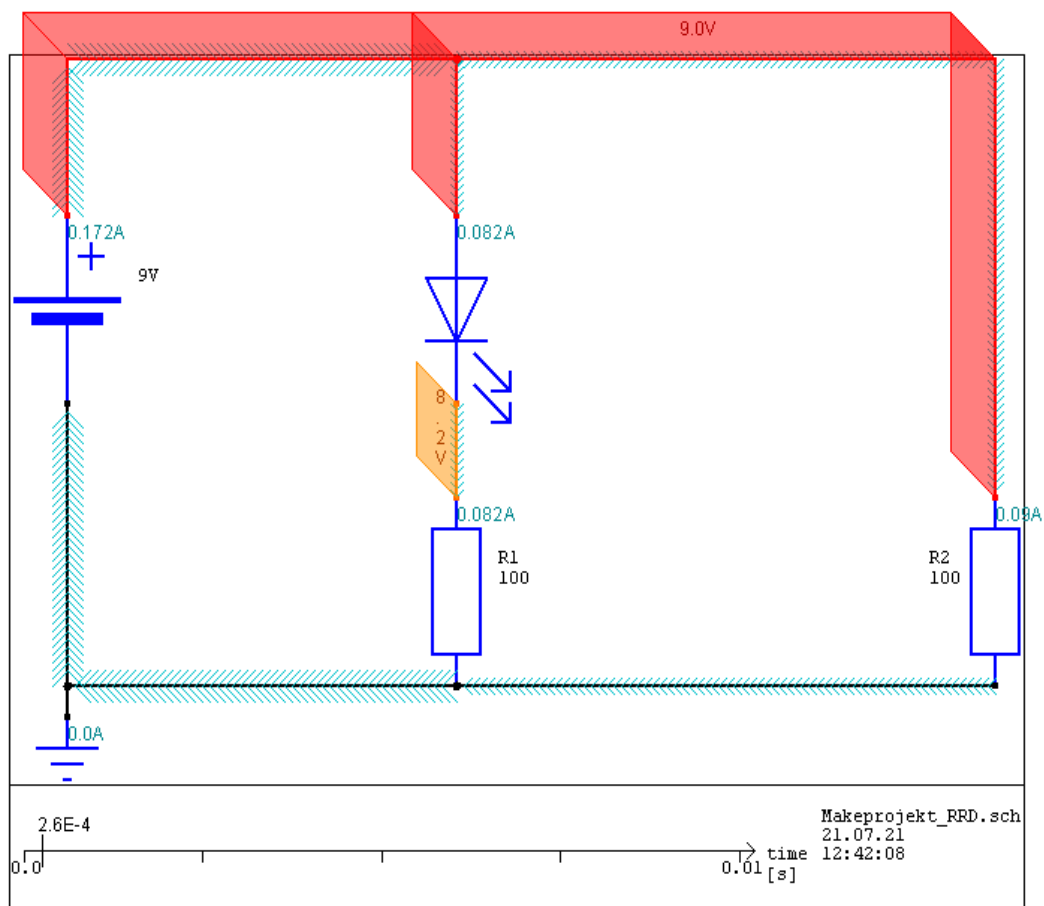


Abbildung 6: Eine einfache Schaltung mit einer Batterie, zwei Widerständen und einer LED als Ausgabe des Schaltplan-Navigations-Systems

Therefore we adjust the resistor R1 in the schematic editor (here in KiCad EESchema) and change it to 500Ω :

¹ For example, in the case of LEDs from an online seller of electronic components called "Conrad", we see that most LEDs have "20mA" written on them, see at <https://www.conrad.de/de/o/leds-bedrahtet-0212029.html>

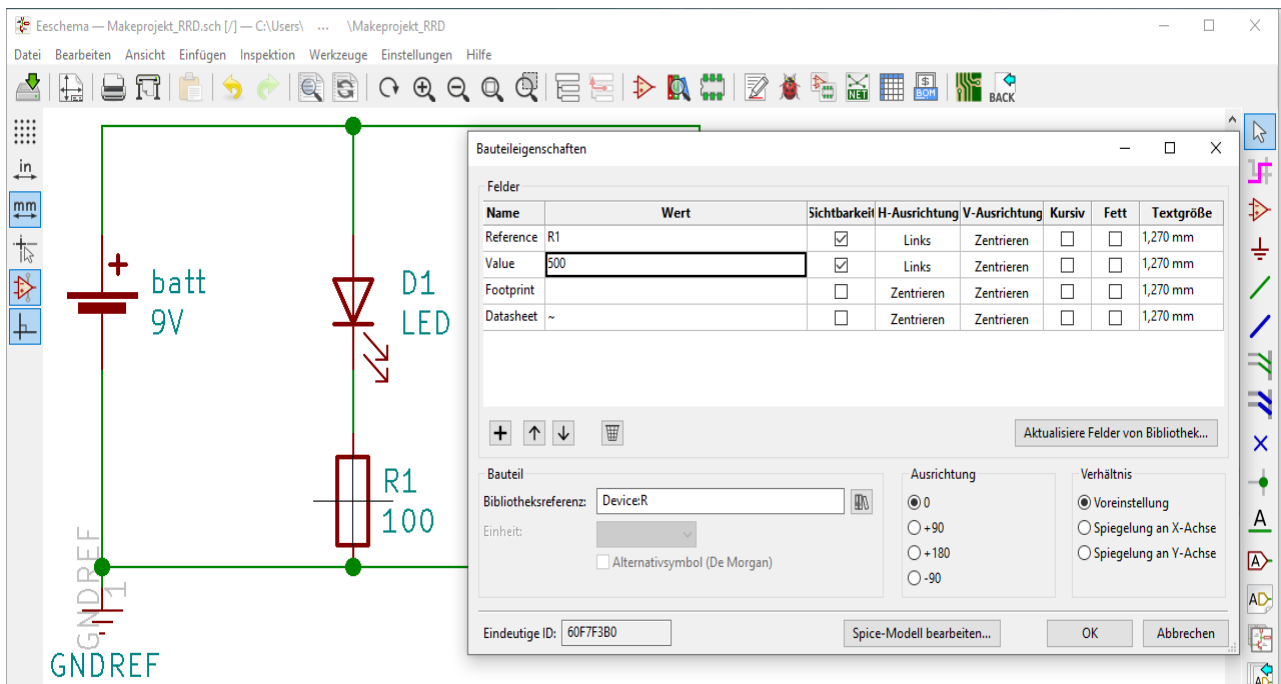


Abbildung 7: Ändern des Wertes von R1 von 100 Ohm zu 500 Ohm

We simulate the circuit again and see that the current through the LED is only 16.5mA. So it stays a bit below the maximum value of 20mA. Thus, the LED will shine well visible, but will not burn out:

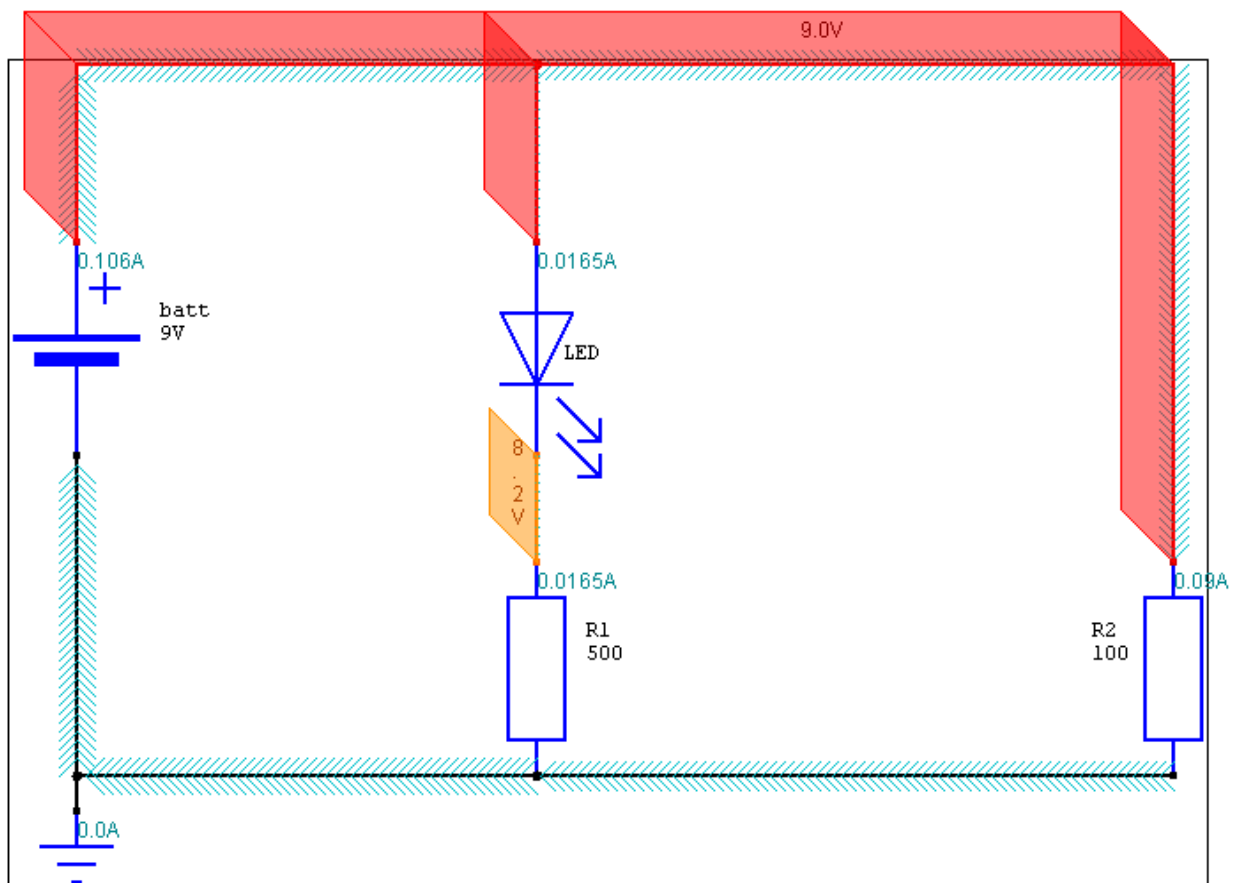


Abbildung 8: Erneute Simulation nach der Anpassung von R1 auf 500 Ohm

Let's try it afterwards in real life:

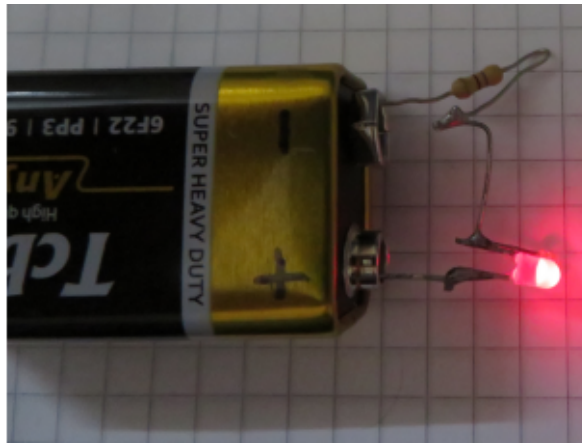


Image 9: 9V block battery, a resistor with approx. 500 Ohm and a LED. The LED shines.

2.2 an advanced example: switching controller

A switching regulator can be used to drive motors: you put a PWM signal into a transistor, and the transistor amplifies the signal into a larger current flow that is strong enough to drive motors. So they are a type of amplifier circuit.

This switching regulator here is a step-up converter that has a voltage of 15V at the input (red), and it turns that into a voltage at the output (green) that goes from 15V to 30V in about 3ms. So it increases the voltage. It does this in two switching states, which are shown in the following two images:

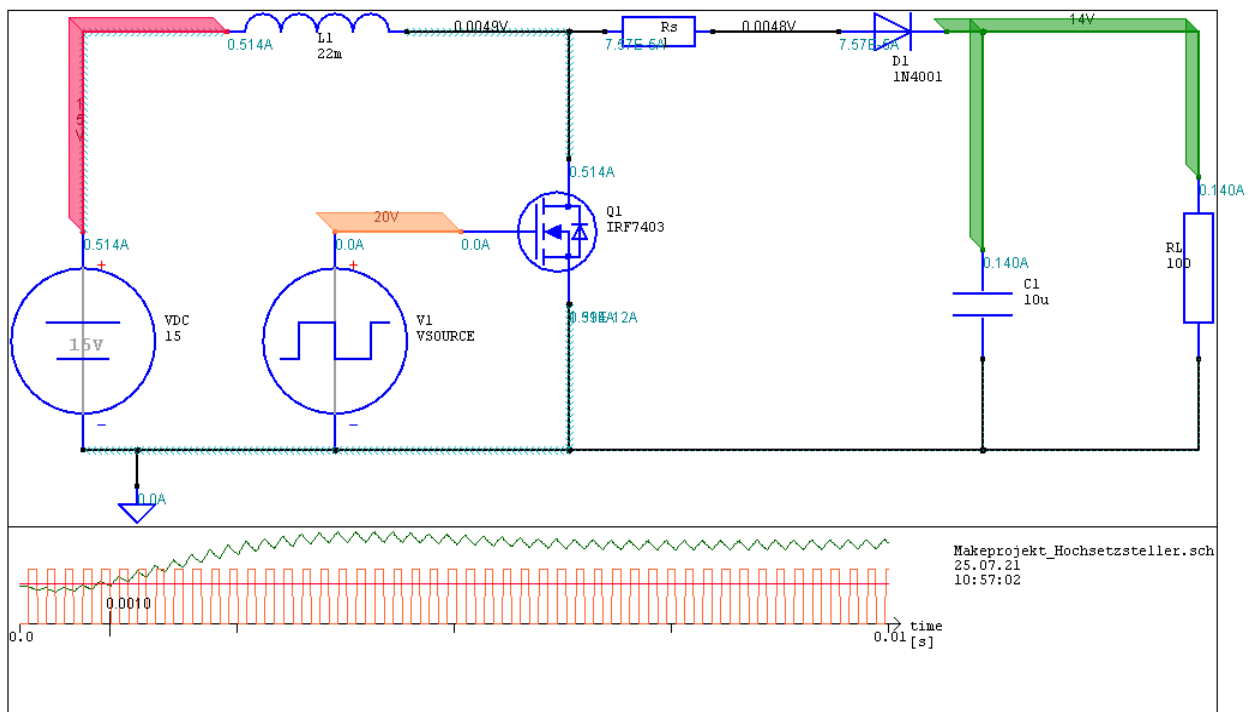


Image 10: Step-up converter with conductive transistor: The current through the coil is built up. The transistor conducts at the first switching state because 20V are applied to the gate (orange). Therefore, the current through the coil increases.

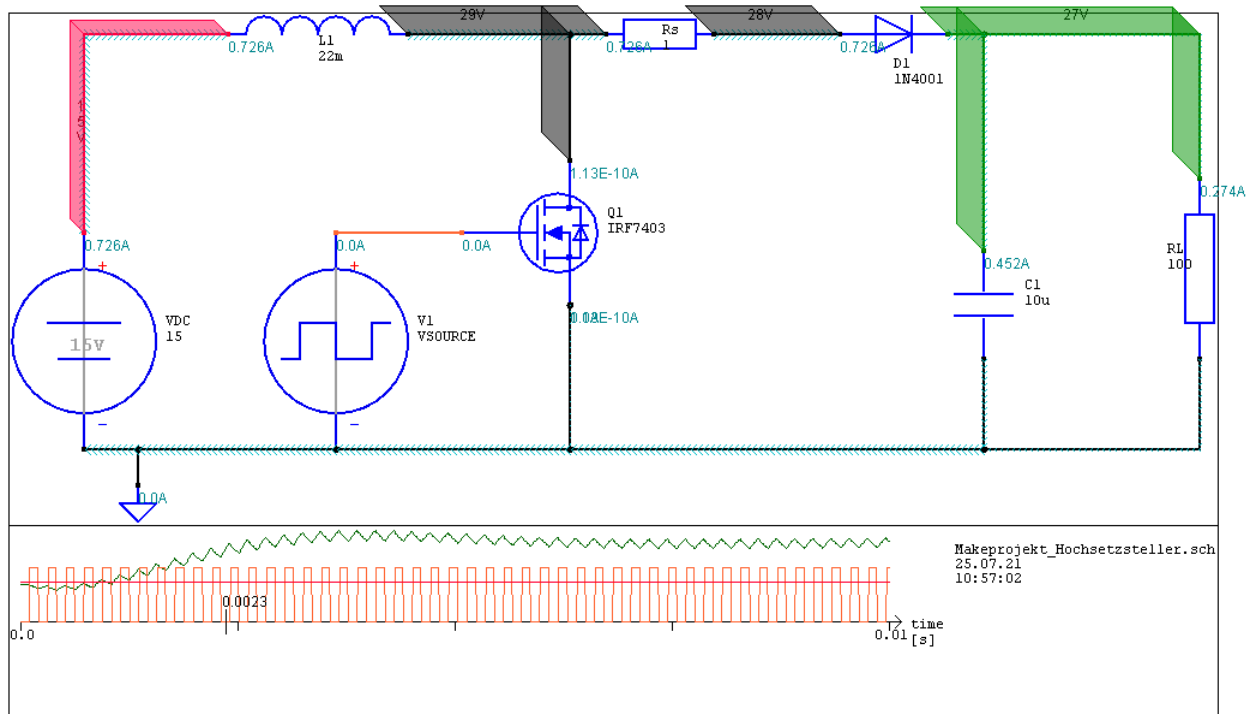


Abbildung 11: Step-up converter with blocking transistor. The current is directed to the output.

As soon as the transistor is switched off and blocks, the current from the coil must find another way. But there is only one way: to the output (green). No matter how high the voltage at the output is, the current still flows there. It flows into the capacitor C1 and increases the voltage there. Therefore, the voltage at C1 keeps increasing until the current inflow from the coil is equal to the current outflow across the 100 ohms of RL. In this example, this equilibrium "current inflow equals current outflow" is reached at about 30V output voltage. It takes about 3ms to reach this equilibrium in this example.

For such a circuit to work, the frequency from the PWM of V1 must match the to coil value of L1 and to C1. For this there are either complicated calculations, described in electrical engineering manuals or you do a simulation. And if the values do not fit, then you try it with other values and a new simulation. Until you have a circuit that works the way you want it to.

3. Conclusion

In all projects that contain any electrics, it is very useful to have an exact overview of what must be wired how, and which components are suitable in which place, and where not. Because otherwise you need a lot of luck to make the project work.

The fastest way to get such an overview is a schematic simulation.

Another good option is to study electrical engineering, but that takes longer.

True, getting started with simulation is never easy either, because you have to become familiar with a new program and deal with the sometimes strange messages from the simulation program. But compared to the hassle of troubleshooting an incorrectly built or even burnt out circuit, simulation is a breeze.