

# ROBOTER DESIGN CHALLENGE

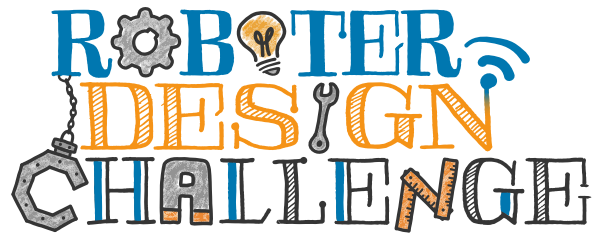


Regelwerk

2026



# Regelwerk 2026



## Inhalt

### Allgemeine Regeln

|                                                    |   |
|----------------------------------------------------|---|
| Teams                                              | 2 |
| Roboter                                            | 2 |
| Spielfeld                                          | 3 |
| Allgemeine Wettkampfregeln                         | 3 |
| Roboterabnahme                                     | 4 |
| Konstruktionswettbewerb und Anerkennung im Studium | 5 |

### Ablauf des Wettkampfes

|              |    |
|--------------|----|
| Vorrunde     | 6  |
| Hauptspiele  |    |
| Gruppenphase | 7  |
| Halbfinale   | 8  |
| Finale       | 10 |

### Sonstiges

|                          |    |
|--------------------------|----|
| Unterstützende Maßnahmen | 11 |
| Timelines                | 11 |
| Ansprechpartner          | 11 |

### Anlagen

|                                                                             |    |
|-----------------------------------------------------------------------------|----|
| <b>Anlage 1</b> Gliederungsvorschlag für den Projektbericht                 | 12 |
| <b>Anlage 2</b> Anleitung zum Aufbau der Steuerplatine                      | 13 |
| <b>Anlage 3</b> Dokumentation zur RobotKit - Bibliothek für den RDC-Roboter | 18 |



## Teams

- 2-3 Teilnehmer pro Team, Schüler-Teams sind über den betreuenden Lehrer anzumelden.
- Pro Runde darf nur ein Teammitglied den Roboter steuern.

## Roboter

- Für den Bau des Roboters sind **ausschließlich** die im **Bausatz** enthaltenen Komponenten zu verwenden:
  - viele (Elektronik-)Teile des Bausatzes werden nach dem Wettbewerb erneut verwendet und sind dementsprechend vorsichtig zu behandeln und wieder abnehmbar im Fahrzeug zu verbauen;
  - der Roboter soll stabil aufgebaut werden. Abgefallene Teile sind umgehend vom Spielfeld zu entfernen.
- Die **Maße des Fahrzeugs** müssen so sein, dass das Fahrzeug im eingeklappten Zustand von ggf. vorhandenen ausfahrbaren Teilen in einen Karton mit Innenmaße (B x T x H) 200mm x 290mm x 230mm gedeckelt passt.
- Jedes Team bekommt zusammen mit dem Bausatz einen Halter mit der **Team-Nummer**, die auf dem Fahrzeug gut sichtbar anzubringen ist. Dieser Halter muss sich nicht innerhalb der vorgegebenen Maße des Fahrzeugs befinden.
- Das **Fahrzeug** darf keine scharfen Kanten besitzen, die bei Kontakt zur Verletzung der Teilnehmer und / oder Beschädigung des Spielfelds oder des gegnerischen Fahrzeugs führen kann.
- Zugelassene **Eigeninvestitionen**
  - max. 10 €
  - Zur Abnahme am Wettkampftag muss der Roboter inklusive nachgekaufter Teile vorgezeigt werden. Der Wert der Eigeninvestitionen ist durch Kaufbelege nachzuweisen. Es wird dringend empfohlen, vor der Anschaffung, eine Rücksprache mit dem Orga-Team zu halten. Regelmissbrauch wird als Verstoß gewertet und kann zur Disqualifizierung des Teams führen.
  - Kleinteile wie z.B. Schrauben können anteilig des Kaufpreises angerechnet werden.
- **3D-Druck**
  - Das Gesamtgewicht von 3D gedruckten Komponenten darf **max. 150 g** betragen. Sofern der Druck nicht über das Orga-Team erfolgt, ist der Nachweis über die Vorlage des Screenshots der Slicing-Software inkl. Gewichtsangaben zu erbringen.



- Der Roboter soll folgende **funktionale Grundanforderungen** erfüllen:
  - auf dem Spielfeld fortbewegen, auch auf einer schiefen Ebene (max. Steigung 10°, verschiedener Untergründe), und ggf. eine Schwelle (Höhe etwa 7 mm) überwinden;
  - gegebene (Tischtennis-)Bälle aufnehmen, transportieren, lagern und in die dafür vorgesehenen Vorrichtungen ablegen. Die konstruktive Umsetzung dieser Funktionen ist dem Team überlassen;
  - verschiedene Mechanismen bedienen, z.B. Drehen von Rädern;
  - einer auf dem Untergrund eingezeichneten Linie autonom folgen, ohne dabei den Ball zu verlieren;
  - innerhalb „**ferngesteuerter**“ Spielbereiche darf der Roboter ausschließlich mit der mitgelieferten Fernbedingung gesteuert werden. Diese darf nicht demontiert oder zerstört werden;
  - innerhalb „**autonomer**“ Bereiche soll der Roboter dem Weg der eingezeichneten Wegelinien automatisch folgen. Dieser Modus, kann wahlweise per Taster auf dem Roboter oder der **Fernbedienung** aktiviert werden. Die Fernbedingung darf währenddessen nicht genutzt werden und **ist sichtbar hochzuhalten**.
- Für den **Bausatz** wird bei der Ausgabe ein **Pfand in Höhe von 50€** eingezogen. Sämtliche Fahrzeuge und Fernbedingungen müssen (unmittelbar) nach dem Wettkampfe bei dem Orga – Team abgegeben werden. **Das Pfand wird bei der Fahrzeugabgabe zurückgegeben.**

## Spielfeld

- Das Spielfeld besteht insgesamt aus zwei Tischtennisplatten je ca. 2,74 m x 1,52 m. Es wird parallel an zwei Tischen gespielt.
- Das Spielfeld sowie die darauf befindende Aufbauten dürfen weder umgestellt noch beschädigt werden. Sämtliche Beschädigungen des Spielfeldes (bei Testfahrten) oder der Bausätze vor dem Wettbewerb sind unverzüglich dem Orga-Team zu melden. Mutwillige Beschädigungen während des Spiels am Wettkampfrang können als Regelverstoß gewertet werden und ggf. zur Disqualifizierung führen.

## Allgemeine Wettkampfregele

- Je nach Spielrunde und Aufgabenstellung können sich zwei (in der Vorrunde max. vier) Teams gleichzeitig auf dem Feld befinden.
- Die Runden beginnen und enden pünktlich zu den im **Spielplan** vorgegebenen Zeiten. **Die Teams haben selbst auf die Zeit zu achten.** Es wird auf das sich verspätende Team nicht gewartet, das Team darf aber verspätet in das Spiel einsteigen und die Runde bis zum regulären Ende absolvieren. Es wird keine Zeitverlängerung gewährt.



- o Bei **Beginn** einer Spielrunde befinden sich die Fahrzeuge **auf dem** ihnen zugewiesenen **markierten Startpunkt** auf dem Spielfeld.
- o Sofern die Aufgabenstellung in der Runde es erlaubt bzw. erfordert, dürfen sich die Fahrzeuge über die gesamte Spielfeldfläche frei bewegen. In diesem Fall ist **kein Kontakt zwischen den gegnerischen Fahrzeugen erlaubt**. Die Fahrzeuge dürfen sich nicht gegenseitig berühren, abschießen oder beschädigen. Gegenstände bzw. Bälle, die während des Spiels (versehentlich) die Spielfläche verlassen (z.B. vom Tisch runterfallen) werden nicht mehr zurückgestellt.
- o Die Teams dürfen den gegnerischen Roboter in keiner Weise manipulieren, dies würde zu einer sofortigen Disqualifizierung des manipulierenden Teams führen.
- o Während der Spielrunde ist das **Eingreifen** in das Spielgeschehen nur dann erlaubt, wenn ein Team seinen eigenen Roboter nach einem Umkippen oder Hängenbleiben (neu) positionieren muss. Kleinere Reparaturen des Roboters dürfen am Spielfeldrand erfolgen, die Spielzeit wird nicht angehalten und läuft dabei weiter. Das Fahrzeug ist nach einem Eingriff stets auf dem Team zugewiesenen, markierten Startpunkt zu positionieren.
- o Sämtliche **Wartungsarbeiten und Reparaturen** am Fahrzeug sind zwischen den Spielen durchzuführen. Es dürfen keine signifikanten technischen Veränderungen am Fahrzeug vorgenommen werden, die sich auf die Gesamtfunktionalität auswirken. Es dürfen lediglich defekte Bauteile ersetzt werden.
- o **Während einer laufenden Runde** dürfen nur Reparaturarbeiten erledigt werden. Wenn ein Defekt auftritt, darf das Team notwendige Reparaturen vornehmen und das Spiel fortsetzen. Der Roboter ist nach der Reparatur auf dem Team zugewiesenen markierten Startpunkt zu positionieren. Die Zeit wird dabei nicht angehalten, die Runde wird regulär beendet.
- o **Zwischen den Runden** ist es den Teams erlaubt, die Roboterwerkzeuge zu tauschen. Diese müssen bei der Roboterabnahme bekanntgegeben werden. Die Regeln bzgl. Roboteraufbau sind dabei einzuhalten.
- o **Disqualifizierung**
  - Bei Verstößen gegen die allgemeinen Regeln führt dies zur Disqualifizierung des Teams.
  - Bei Verstößen gegen die technischen Regeln führt dies zur Disqualifizierung des Teams.
  - Bei unsportlichem Verhalten oder Sabotage von Teams oder einzelnen Teammitgliedern gegenüber anderen Teams oder dem Organisations-Team kann dies zur Disqualifikation führen. Die Beurteilung erfolgt durch das Organisations-Team.

## Roboterabnahme

Die Abnahme der Roboter erfolgt am Morgen des Wettkampftages.



## Konstruktionswettbewerb und Anerkennung im Studium

Im Rahmen des Wettbewerbs findet auch ein **Konstruktionswettbewerb** statt. Es wird ein Konstruktionspreis verliehen.

Die Jury bewertet den Roboter, dessen Präsentation durch das Team und die Funktionsfähigkeit auf dem Feld. Zur Bewertung werden folgende Kategorien herangezogen:

- o **Konstruktion** („Kreativität“, Qualität und Begründung des (mechanischen) Konzepts)
- o **Fertigung** (praktische Umsetzung des Konzeptes und Qualität der Fertigung, z.B. ob „sauber“ gearbeitet wurde, welche Materialien wurden verwendet etc.)
- o **Elektrotechnik/Elektronik** (z.B. Dimensionierung und Auslegung elektrischer Komponenten wie Sensoren, Motoren, Leitungsführung etc.)
- o **Kreativität des Gesamtsystems** (äußerliches Erscheinungsbild, Kombination der Funktionalität und des (optischen) Designs, etc.)
- o **Präsentation des Roboters durch das Team** (Präsentationsmaterial, Gesamteindruck von dem Team)

Die Jury schlägt ein System zur Verleihung des **Preises „Beste Konstruktion“** vor.

Spätestens drei Wochen nach dem Wettbewerb, bis zum 14.12.2026, ist ein **Projektbericht** mit einem Umfang von maximal 20 Seiten ohne Anhang bei den Wettbewerbskoordinatoren einzureichen. In dem Bericht sind insbesondere der systematische Ansatz zur Findung der konstruktiven Lösung für das Fahrzeug nachzuweisen und die konkrete Ausführung zu dokumentieren.

Der Projektbericht ist mit einem Deckblatt zu versehen, aus dem der Teamname und die Teammitglieder, gegebenenfalls mit Matrikelnummern, ersichtlich sind. Ein Gliederungsvorschlag für den Bericht ist in der Anlage zum Regelwerk zu finden.

Die **Teilnahme am Wettbewerb und Konstruktionsprojekt** kann als **TWP-Modul „Design Challenge Teilnahme“** anerkannt werden.

Die Bewertung für die Leistung im Modul erfolgt auf Grundlage der Jury-Bewertung, der erreichten Wettbewerbsplatzierung (Bonuspunkte) sowie Bewertung des Projektberichts.

Die an der Hochschule bereits immatrikulierten Studierenden melden sich im eCampus innerhalb regulärer Prüfungsanmeldeperiode zur Prüfungsleistung „Design Challenge Teilnahme“ an (Studierende nach BPO2018 unter Prüfungsnummer „6000“, Studierende nach BPO2025 unter Prüfungsnummer „1302“, Erstprüfer Prof. Dr. Olena Kuzmicheva). Die Note wird im eCampus eingetragen.

Teilnehmende Schüler:innen bekommen für die Teilnahme am Wettbewerb eine Bescheinigung über die Bewertung und können später (sofern sie ein Studium im Fachbereich aufnehmen) die erfolgreiche Teilnahme am Wettbewerb für das oben genannte Modul anrechnen lassen.



## Ablauf des Wettbewerbs

Der genaue Spielplan wird am Wettbewerbstag nach der Roboter-abnahme zeitig bekannt gegeben. Der Wettbewerb wird in mehrere Phasen mit steigender Komplexität der Aufgaben aufgeteilt.

### Vorrunde

Die Vorrunde dient der Vorstellung der Teams vor der Jury und der Zuschauer sowie Testung diverser Funktionalitäten des Roboters.

Jedes Team bekommt zusammen mit dem Bausatz eine Nummer, die auf dem Fahrzeug sichtbar anzubringen ist. Zu Beginn des Rennens hat jedes Team **zwei Minuten Zeit den Roboter und das Team in wenigen Sätzen vorzustellen**. Die Vorstellung des Teams ist für die Bewertung im Rahmen des TWP – Moduls „Design Challenge Teilnahme“ obligatorisch.

Jedes Team bekommt einen Startplatz zugewiesen. Das Fahrzeug ist auf der entsprechenden Markierung zu platzieren.

Der Aufbau des Tisches in der Vorrunde ist in der Abbildung 1 schematisch dargestellt.

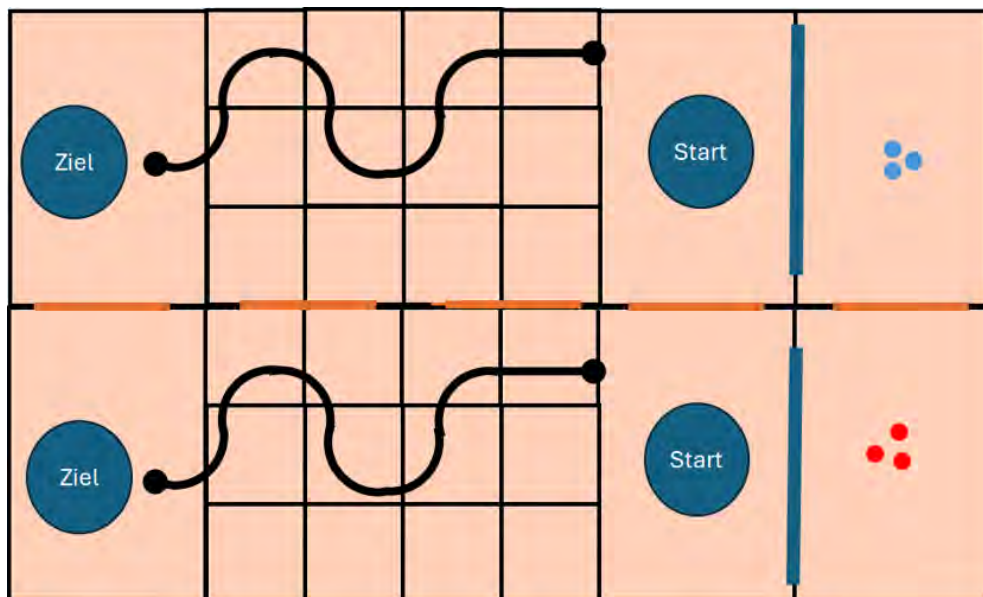


Abbildung 1: Schematischer Aufbau des Tisches in der Vorrunde

Die **Aufgabe der Teams** besteht darin, einen Ball ferngesteuert aufzunehmen und zum Anfang des autonom zu befahrenden Bereichs zu bringen. Sobald der Roboter einen Startpunkt für das autonome Fahren erreicht hat, ist in den Modus „Linienfolge“ umzuschalten und der Ball autonom zu transportieren (die Fernbedienung ist gut sichtbar hochzuhalten). Am Ende des autonomen Weges kann wieder auf Fernsteuerung umgeschaltet werden, um den Ball auf einer freien Ablageposition abzulegen.

**In dieser Runde werden keine Punkte gesammelt. Die Runde ist beendet, wenn alle Teams ihren Ball zum Ziel gebracht haben bzw. wenn max. Rundedauer erreicht ist.**



## Hauptspiele

Alle Hauptspiele haben zum Ziel, möglichst viele Punkte zu sammeln. Je nach Runde kommen N beste Teams weiter. Bei Punktgleichheit wird eine Entscheidungsrunde gefahren.

## Gruppenphase

In dieser Phase stellen die Roboter ihre Transportfähigkeiten und die Fahrer ihre Reaktionsgeschwindigkeit und Steuergeschicklichkeit unter den Beweis. Die Teams werden in vier Gruppen eingeteilt, um Spielreihenfolge zu bestimmen. Eine Spielrunde hat eine fixierte Dauer, in der Punkte gesammelt werden. Je nach Anzahl angetretener Teams wird über die Anzahl Runden entschieden. Anschließend wird anhand der Gesamtpunktzahl ein Rating erstellt, um die Teams für die Halbfinale zu bestimmen.

Gespielt wird parallel an zwei Tischen mit je zwei Teams an einem Tisch. Die Roboter werden ferngesteuert. Die Abbildung 2 stellt den Tischaufbau schematisch dar.

Der Tisch ist ähnlich zu einem Billardtisch aufgebaut, ist aber durch eine Schwelle (oder mehreren, beispielhaft durch dicke blaue Linien im Bild gekennzeichnet) „geteilt“. Zum Versenken der Bälle sind mehrere farbige Körbe vorgesehen (in der Abbildung exemplarisch vier). Jedes teilnehmende Team ist einer Farbe zugeordnet und hat als Aufgabe, Körbe entsprechender Farbe mit Bällen zu füllen.

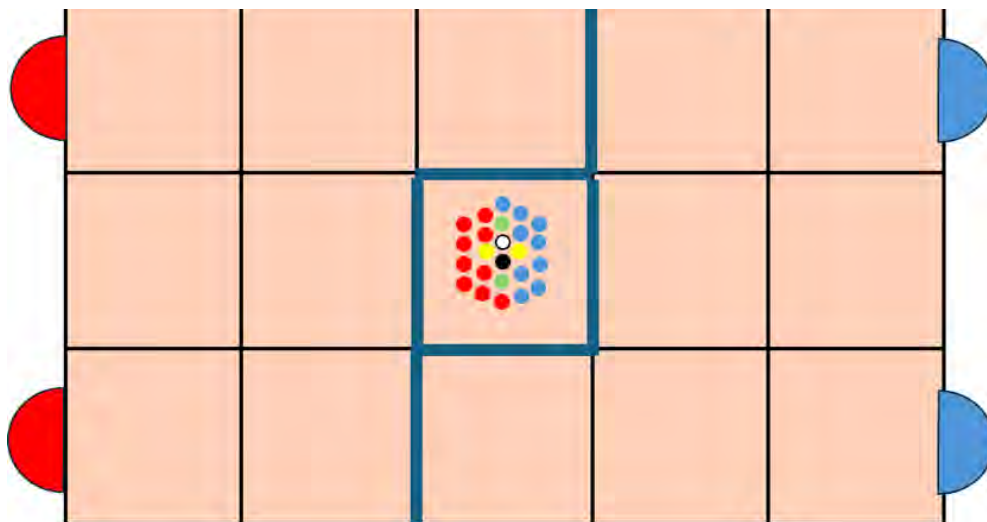


**Abbildung 2: Schematischer Aufbau des Tisches in der Gruppenphase**

Am Anfang des Spiels ist die Anordnung in der Mitte des Tisches mit farbigen Bällen bestückt, die verschiedene Wertigkeit haben:

- o **Teamfarbe 1 (blau)** – 2 Punkte im Netz des Teams 1 // 1 Punkt im Netz des Teams 2 – 7 Bälle
- o **Teamfarbe 2 (rot)** – 2 Punkt im Netz des Teams 2 // 1 Punkt im Netz des Teams 1 – 7 Bälle
- o **Gelb** – 3 Punkte – 2 Bälle

- o **Grün** – 5 Punkte – 2 Bälle
- o **Schwarz** - verdoppelt Punkte der Bälle im Netz – 1 Ball
- o **Weiß** - annulliert Punkte der Bälle im Netz – 1 Ball
- o **Die Wirkung des schwarzen Balls wird durch den weißen Ball „kompensiert“, genauso umgekehrt.**

Nach dem Startsignal gilt es für die Teams die **Bälle in Körbe der Teamfarbe zu versenken**. Es ist dabei darauf zu achten, dass **alle Körbe** mit Bällen **zu füllen sind**. Bleibt am Ende des Spiels ein Korb leer, ergeben sich für das zugehörige Team **5 Strafpunkte**, die von dem erreichten Punktestand abgezogen werden.

**Die Runde ist beendet, wenn alle Bälle in Körben abgelegt sind bzw. max. Rundedauer erreicht ist.**

#### Weiterkommen

- o bei mehreren Runden werden die Punkte über alle Runden „gesammelt“;
- o **acht Teams** mit dem **besten Punktestand** „global“ **nach allen Runden** der Gruppenphase kommen in das Halbfinale.

## Halbfinale

Die acht besten Teams aus der Gruppenphase treten paarweise gegeneinander an.

In dieser Runde werden höhere Anforderungen an die Manipulationswerkzeuge der Roboter sowie Geschick und strategische Fähigkeiten der Fahrer gestellt. Die Bälle müssen nicht nur von A nach B transportiert werden. Die Bälle müssen erst „befreit“ werden, um die anschließend zu sammeln, zu transportieren und gemäß Aufgabe zu lagern.

Wie in Gruppenphase wird eine feste Dauer definiert, in der Punkte gesammelt werden können. Anschließend wird anhand der Gesamtpunktezahl ein Rating erstellt, um die Teams für das Finale zu bestimmen.

Auch in dieser Runde wird parallel an zwei Tischen mit je zwei Teams an einem Tisch gespielt. Der Aufbau des Tisches ist im Vergleich zur Gruppenphase modifiziert (siehe schematische Darstellung in der Abbildung 3).

Dicke blaue Linien symbolisieren Trennwände, die variabel auf dem Feld verteilt sind und Hindernisse darstellen, um die herumgefahren werden soll. Hindernisse dürfen nicht (absichtlich) gekippt, verschoben oder umgefahren werden.

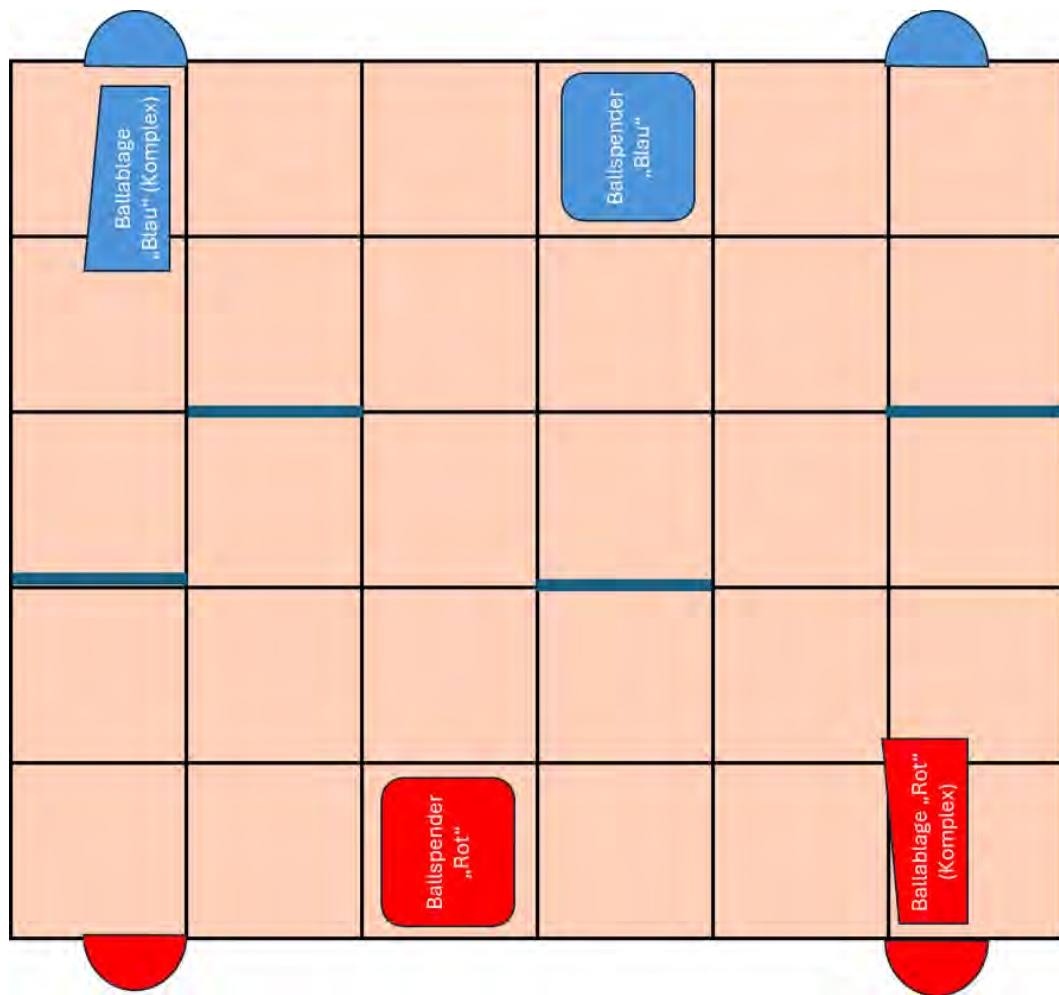


Abbildung 3: Schematischer Aufbau des Tisches im Halbfinale

Die Bälle werden über zwei Spender verteilt. Beide Spender werden mit gleicher Anzahl Bälle „bestückt“. Die Reihenfolge der Bälle im Spender ist nicht einsehbar. Die Wertigkeit der Bälle bleibt wie in der Gruppenphase. Zur Entnahme des Balls aus dem Spender muss an einem Rad gedreht werden. Der Ball muss anschließend aus der Ablage rausgenommen und zu Ablagekörben des Teams transportiert werden.

Die **Aufgabe der Teams** ist, **Bälle zu sammeln und diese in den Aufhängekörben zu sichern**. Dabei sind **die Sammelstellen unterschiedlich komplex zu erreichen**. Die Wertigkeit der gesammelten Bälle richtet sich nach Farbe (siehe Gruppenphase) und nach „Komplexität“ der Ablage. Bälle, die in der komplexeren Ablage gesichert sind, werden doppelt gezählt.

**Die Runde ist beendet, wenn alle Bälle in Körben abgelegt sind bzw. max. Rundedauer erreicht ist.**

#### Weiterkommen

- o bei mehreren Runden werden die Punkte über alle Runden „gesammelt“;
- o **vier Teams** mit dem **besten Punktestand nach allen Runden der Halbfinale** kommen in das Finale.



## Finale

Die Finalrunden werden zwischen den vier Gewinnern der Halbfinale ausgetragen. Zwei Teams treten in einer Runde gegeneinander an. Es gilt jeder gegen jeden.

In dieser Runde müssen die Roboter nicht nur in der Lage sein, Bälle für das Sammeln zu „befreien“, diese zu transportieren und „geschickt“ zu lagern. Das Feld beinhaltet diesmal auch Bereiche, in denen Roboter autonom agieren müssen und die Bälle ohne Fernsteuerung durch den Fahrer zum Ziel bringen müssen.

Der Punktestand nach den Finalrunden definiert die Platzierung.

Im Finale werden beide bisher getrennt bespielte Tische über zwei „Brücken“ miteinander verbunden. Das Spielfeld beinhaltet nun auch Bereiche, die ausschließlich autonom befahrbar sind. Ebenfalls werden Ablagestellen für Bälle vorgesehen, die „frei“ angefahren werden können. Ggf. ist dabei eine Rampe (max. Steigung 10°) zu überwinden, um die Ablagefläche zu erreichen. Das Befahren dieser Ablagebereiche ist sowohl für das eigene als auch für das gegnerische Team erlaubt.

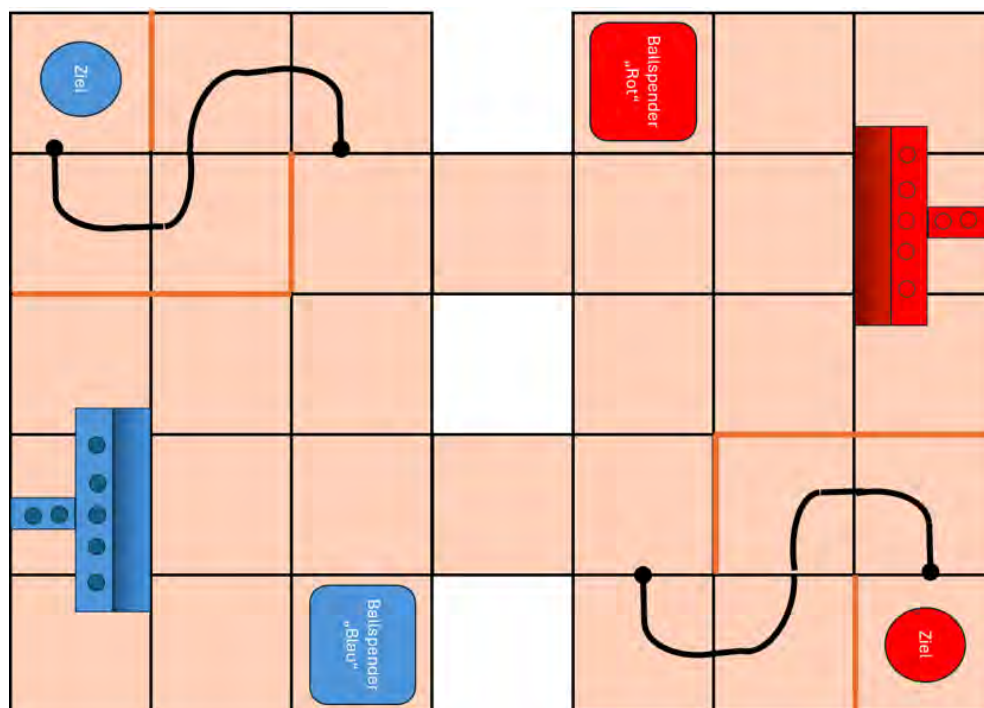


Abbildung 4: Schematischer Aufbau des Tisches im Finale

Die Bälle werden wie im Halbfinale über zwei Spender verteilt. Die **Wertigkeit** der gesammelten Bälle **verdoppelt sich**, wenn die Bälle zu einer **Ablage** gebracht werden, **die nur autonom erreichbar ist**. Hierzu muss der Roboter in der Lage sein, den aufgenommenen Ball über das autonome Folgen der aufgezeichneten Linie (siehe Vorrunde) sicher ins Ziel zu bringen und aus diesem Bereich autonom wieder rauszufahren. **Das gegnerische Team darf diesen Bereich nicht betreten.**

**Die Runde ist beendet, wenn alle Bälle in Körben abgelegt sind bzw. max. Rundedauer erreicht ist.**



## Sonstiges

### Unterstützende Maßnahmen

- o **Technische Fragen** werden für alle Teilnehmer öffentlich gemacht
- o **Druckservice** wird **bis zum 11.11.2026** angeboten unter Absprache mit Selbstabholung. Falls die Möglichkeit besteht, darf auch eigener 3D Drucker verwendet werden. Die Teile müssen bei der technischen Abnahme klar identifizierbar sein. Bei Fragen bitte an die genannten Ansprechpartner:innen wenden.
- o Den Teams wird während der letzten drei Wochen der Bauphase Zugang zu ausgewählten Teilstücken der Spielfeldkomponenten gewährleistet. Dafür bitte an die unten aufgeführten Ansprechpartner:innen wenden, um einen passenden Termin zu finden. Wie oft man sich das Spielfeld anschaut, ist nicht begrenzt.

### Timelines

- o **Ausgabe der Bausätze:** 12.10.2026, 9:45 - 16:00 im Raum L112 oder nach Absprache
- o **3D-Druckservice:** bis zum 11.11.2026
- o **Wettbewerb:** 18.11.2026

### Ansprechpartner

- o **3D-Druck, Spielfeld/Tisch**
  - Tim Sirek (tim.sirek@jade-hs.de)
  - Max Strehlke (max.strehlke@student.jade-hs.de)
  - Julian Liebetrau (julian.liebetrau@student.jade-hs.de)
- o **Roboterbau / Bausatz**
  - Jan Dierks (jan.dierks@jade-hs.de)
- o **Allgemeine Fragen**
  - Anna-Marie Dörfler (anna-marie.doerfler@student.jade-hs.de)
- o **Regelwerk, Konstruktionsprojekt und Anerkennung als TWP**
  - Prof. Dr. Olena Kuzmicheva (olena.kuzmicheva@jade-hs.de)



## Anlage 1

### Gliederungsvorschlag Projektbericht

#### 1. Team und Motivation

#### 2. Bausatz

#### 3. Mechanisches Konzept und Roboterkomponenten

##### 3.1 Gesamtkonzept

##### 3.2 Fahrwerk

##### 3.3 Greifer bzw. weitere Tools

##### 3.4 Gesamtaufbau und mechanische Integration aller Komponenten

##### 3.5 Konstruktionsunterlagen

#### 4. Elektronik und Steuerung

##### 4.1 Elektronische Komponenten und deren Integration in das Gesamtsystem

##### 4.2 Steuerungskonzept

##### 4.3 Programmbeschreibung

#### 5. Zusätzliche Komponenten aus Eigenentwicklung

##### 5.1 ...

#### 6. Zusammenfassung und persönliches Fazit der Gruppenmitglieder

#### 7. Anhänge (wenn im Hauptdokument nicht integriert)

##### A Technische Zeichnungen

##### B Schaltbilder, Verdrahtungspläne, etc.

##### C Programm: Flussdiagramme und Quelltext



## Anlage 2

### Anleitung zum Aufbau der Steuerplatine

Die Platine (siehe Abbildung unten) bildet das Herzstück eines Roboters, der im Rahmen der Roboter Design Challenge mehrere Aufgaben bewältigen soll.

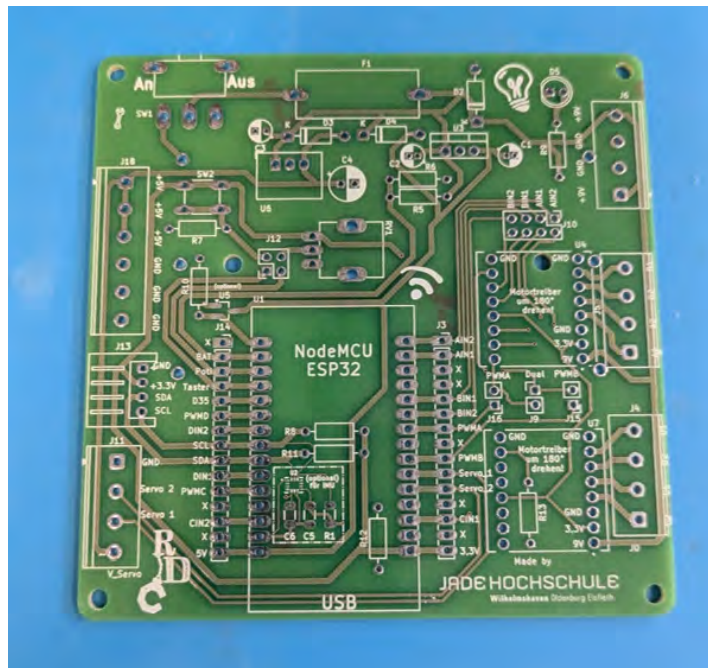


**Abbildung A1: Die unbestückte Platine**

Gesteuert wird der Roboter über einen ESP32-Mikrocontroller, der sowohl die Motoransteuerung übernimmt als auch die Steuersignale eines Gamingcontrollers verarbeitet. Für die Motoransteuerung sind zwei Motortreiber (TB6612FNG) verbaut, sodass mehrere Motoren gleichzeitig und unabhängig angesteuert werden können.

Die Spannungsversorgung ist bewusst getrennt aufgebaut: Ein Spannungsregler versorgt ausschließlich den Mikrocontroller, ein zweiter unabhängig davon die Servomotoren. Dadurch werden Störungen durch Lastspitzen der Motoren vom empfindlicheren Steuerteil ferngehalten.

Für Erweiterungen und Debugging bietet die Platine zusätzlich:

- einen frei belegbaren Taster und ein Potentiometer zur Einstellung von Parametern,
- eine I<sup>2</sup>C-Schnittstelle mit einfacher Steckverbindung für weitere Sensoren oder Module,

Damit vereint die Platine alle nötigen Funktionen – Steuerung, Motoransteuerung, Sensorik und Energieversorgung – auf einer kompakten Baugruppe, die als Basis für den Roboter Design Challenge-Roboter dient.



Als Erstes werden die Widerstände eingelötet. Sie sind die flachsten Bauteile auf der Platine und sollten deshalb zuerst platziert werden.

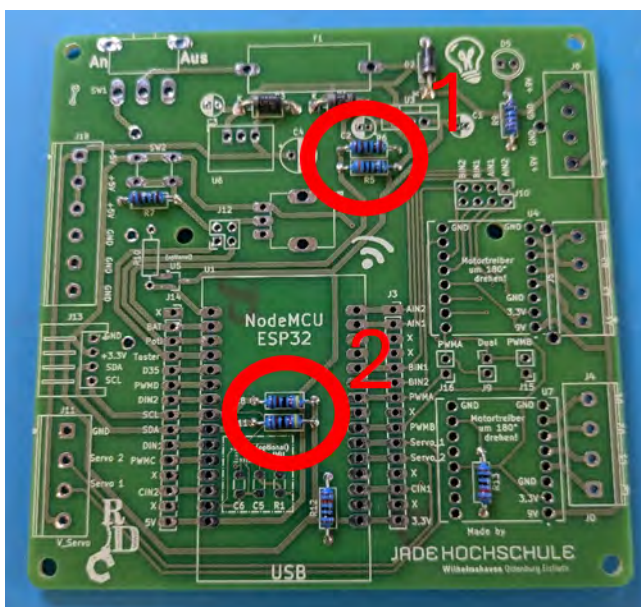


Abbildung A2: Eingelötete Widerstände

1: Spannungsteiler, um die Akkuspannung auf ein verträgliches Maß für die ADC des ESP32 zu bringen. So lässt sich mit dem ESP32 direkt die Akkuspannung messen, um Rückschlüsse auf die Restladung der Akkus zu ziehen. rst platziert werden.

2: Pull Up Widerstände der i<sup>2</sup>c-Schnittstelle. Nur nötig, wenn der i<sup>2</sup>c-Bus genutzt werden soll.

Für den Motortreiber und den ESP32 werden Buchsenleisten statt direkter Lötverbindungen eingesetzt. So lassen sich diese Bauteile im Fehlerfall einfach austauschen, ohne erneut löten zu müssen. **Achte auf korrekte Polarität, besonders bei den Kondensatoren.**

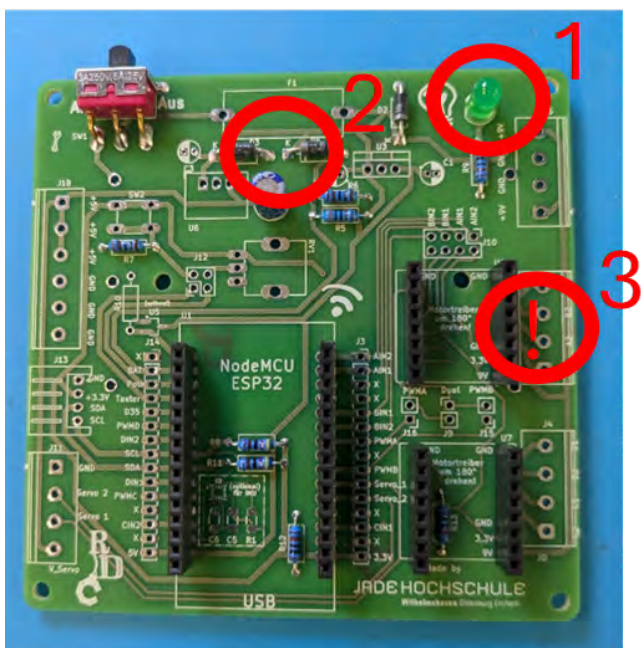


Abbildung A3: Eingelötete Buchsenleisten

1: Spannungsanzeige

2: Spannungsregler reagieren empfindlich darauf, wenn die Spannung am Ausgang höher ist als am Eingang. Beim Ausschalten der Schaltung oder wenn der Elektromotor abrupt stoppt, kann der Motor als Generator wirken und eine Spannung zurückspeisen.

3: Diesen Lötplatz für Schraubklemmen im Gegensatz zur folgenden Anleitung zunächst freilassen. Dann lässt sich der Akkuhalter deutlich einfacher verlöten!



Die Platine ist nun mit fast allen Bauteilen bestückt. Die noch fehlenden Bauteile sind optional und für den Betrieb nicht notwendig. Zu sehen sind außerdem bereits gesetzte Jumper, mit denen ein zweiter Motortreiber zugeschaltet werden kann. Werden diese Jumper entfernt, lassen sich die entsprechenden Pins anderweitig nutzen – dann steht allerdings kein zweiter Motortreiber zur Verfügung.

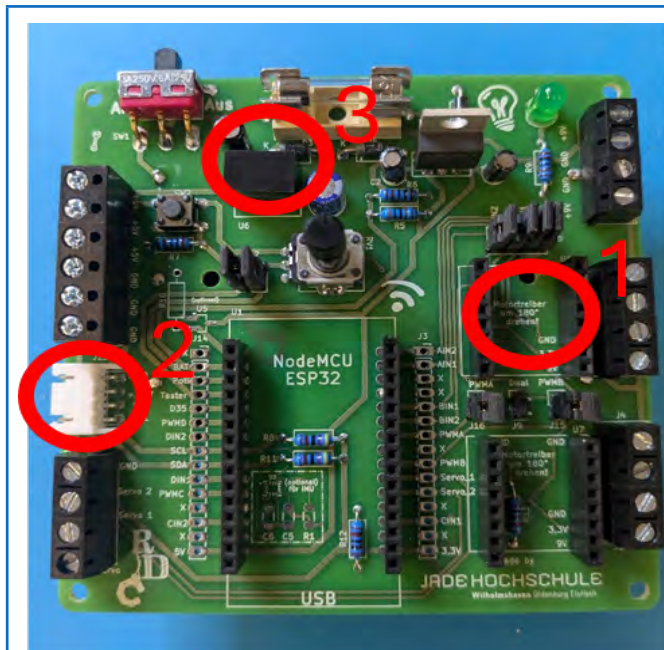


Abbildung A4: Fast fertig bestückte Platine

- 1: optionaler Motortreiber Steckplatz. Hier einen Motortreiber einzubauen ist nur sinnvoll, wenn neben den Antriebsmotoren noch weitere Motoren (Abgesehen von Servos oder Schrittmotoren zum Einsatz kommen sollen.
- 2: Steckplatz für i<sup>2</sup>c fähige Sensoren oder Aktoren
- 3: Schaltregler für die Spannung der Servos. Beim Einlöten müssen die Beinchen vorsichtig etwas angepasst werden. Das Layout passt nicht zum Bauteil.

Zuletzt werden ESP32, Sicherung und Motortreiber eingesetzt.

**Wichtig!** Der Motortreiber muss **um 180 Grad gedreht** eingesetzt werden.

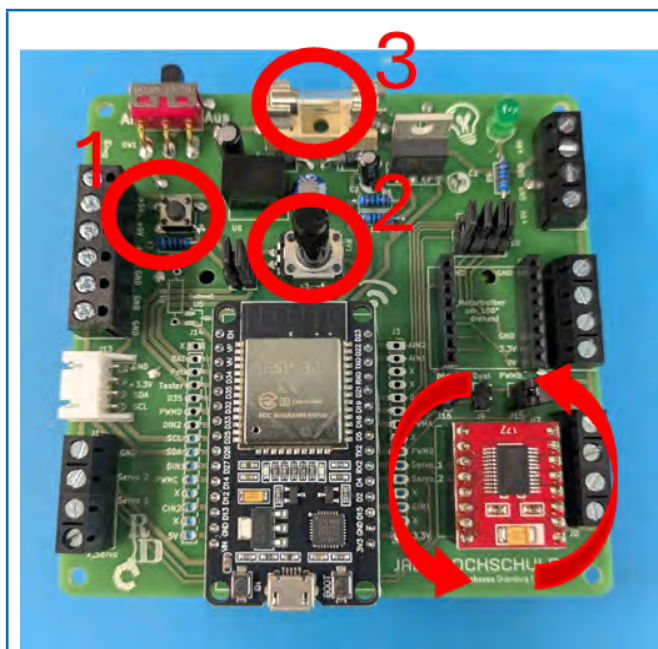


Abbildung A5: ESP32, Sicherung und Motortreiber installiert

- 1: Taster frei benutzbar in der Programmierung. Beispielsweise zum Umschalten zwischen manueller und autonomer Fahrweise.
- 2: Potentiometer ebenfalls als Analogwert frei nutzbar (beispielsweise um die Motoren so zu trimmen, dass sie gleich schnell laufen).
- 3: Feinsicherung 20x5mm 2A (flink oder mittelträge)



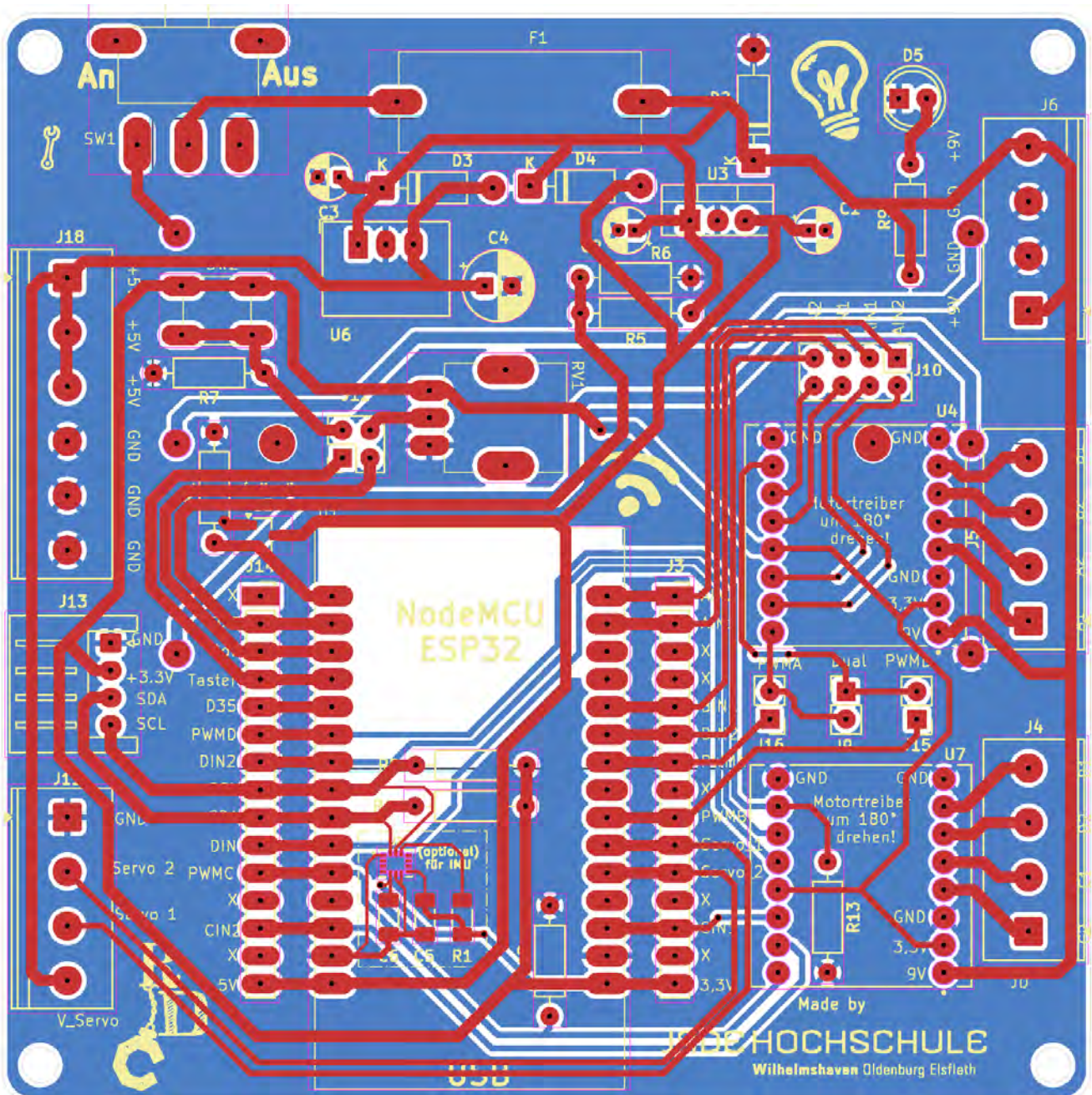


Abbildung A6: Das Board





| Reference       | Value                | Footprint                                                | Qty |
|-----------------|----------------------|----------------------------------------------------------|-----|
| BT1             | BK-18650-PC6         | Batteriehalter_3x:BAT_BK-18650-PC6                       | 1   |
| C1              | 220 nF               | Capacitor_THT:CP_Radial_D4.0mm_P1.50mm                   | 1   |
| C2              | 330 nF               | Capacitor_THT:CP_Radial_D4.0mm_P1.50mm                   | 1   |
| C3              | 2,2 uF               | Capacitor_THT:CP_Radial_D4.0mm_P2.00mm                   | 1   |
| C4              | 100uF                | Capacitor_THT:CP_Radial_D6.3mm_P2.50mm                   | 1   |
| C5, C6          | 100n                 | Capacitor_SMD:C_1206_3216Metric_Pad1.33x1.80mm_Han       | 2   |
| D2, D3, D4      | 1N 4004              | Diode_THT:D_DO-41_SOD81_P10.16mm_Horizontal              | 3   |
| D5              | grün                 | LED_THT:LED_D5.0mm_Clear                                 | 1   |
| F1              | ESKA 503400          | Sicherungshalter:ESKA 503.400                            | 1   |
| J3, J14         | MPE 115-1-015        | Connector_PinSocket_2.54mm:PinSocket_1x15_P2.54mm_v      | 2   |
| J4, J5, J6, J11 | Screw_Terminal_01x04 | TerminalBlock_Phoenix:TerminalBlock_Phoenix_PT-1,5-4-5.0 | 4   |
| J9, J15, J16    | MPE 087-1-002        | Connector_PinHeader_2.54mm:PinHeader_1x02_P2.54mm_       | 3   |
| J10             | MPE 087-2-008        | Connector_PinHeader_2.54mm:PinHeader_2x04_P2.54mm_       | 1   |
| J12             | MPE 087-2-004        | Connector_PinHeader_2.54mm:PinHeader_2x02_P2.54mm_       | 1   |
| J13             | Conn_01x04           | Connector_JST:JST_XH_S4B-XH-A_1x04_P2.50mm_Horizonta     | 1   |
| J18             | Screw_Terminal_01x06 | TerminalBlock_Phoenix:TerminalBlock_Phoenix_PT-1,5-6-5.0 | 1   |
| R1, R10         | 100k                 | Resistor_SMD:R_1206_3216Metric_Pad1.30x1.75mm_Handl      | 2   |
| R5              | 3k                   | Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm      | 1   |
| R6, R7, R9      | 1k                   | Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm      | 3   |
| R8, R11         | 4,7k                 | Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm      | 2   |
| R12, R13        | 10k                  | Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm      | 2   |
| RV1             | 10k                  | Potentiometer:RK09K113,Resistor_THT:R_Axial_DIN0207_L6   | 1   |
| SW1             | T80-ST5-13P-A12R-EH  | Schalter:T80-ST5-13P-A12R-EH                             | 1   |
| SW2             | Taster 9302          | Taster:JTP-1130                                          | 1   |
| U1              | NODEMCU-ESP32        | Breakout_Boards:NODEMCU_ESP32                            | 1   |
| U2              | BMI270               | BMI270:XDCR_BMI270                                       | 1   |
| U3              | uA 78S05             | Package_TO_SOT_THT:TO-220-3_Vertical                     | 1   |
| U4, U7          | ROB-14450            | TB6612FNG:MODULE_ROB-14450                               | 2   |
| U5              | SGM809B-JXN3LG       | Package_TO_SOT_SMD:SOT-23_Handsoldering                  | 1   |
| U6              | R-78B5.0-1.0         | Converter_DCDC:Converter_DCDC_RECOM_R-78B-2.0_THT        | 1   |

Abbildung A8: Schtückliste



## Anlage 3

### RobotKit - Bibliothek für den RDC-Roboter

#### 1. Was ist die RobotKit-Bibliothek?

Die RobotKit-Bibliothek macht es (hoffentlich) einfach, euren Roboter in Arduino C++ zu programmieren. Ihr müsst nichts über Pins, PWM, Bluetooth oder Interrupts wissen – das erledigt die Bibliothek. Ihr schreibt nur noch das, was euer Roboter tun soll. Natürlich steht es euch frei aus diesem Dokument z.B. nur die Pinbelegung zu entnehmen und euer eigenes Ding zu machen.

##### Was die Bibliothek für euch übernimmt

| Modul           | Was es macht                                    | Zugriff im Sketch      |
|-----------------|-------------------------------------------------|------------------------|
| RobotKit        | Motoren, Servos, Sensoren, Batterieschutz       | robot.xxx()            |
| RobotController | PS4-Controller via Bluetooth (Bluepad32)        | controller.xxx()       |
| Motor           | Einzelmotor direkt ansteuern (Fortgeschrittene) | robot.motorC.setze(80) |
| RobotServo      | Servo 1 und 2 (direkt nach begin() nutzbar)     | robot.servo1.xxx()     |

Ihr braucht nur zwei Includes: RobotKit.h und RobotController.h – alles andere ist automatisch dabei.

#### 2. Einrichtung (einmalig)

##### 2.1 Arduino IDE vorbereiten

###### o Schritt 1: Boardverwalter-URL hinzufügen

**Arduino IDE** --> Einstellungen --> Zusätzliche Boardverwalter-URLs:

[https://raw.githubusercontent.com/ricardoquesada/esp32-arduino-lib-builder/master/bluepad32\\_files/package\\_esp32\\_bluepad32\\_index.json](https://raw.githubusercontent.com/ricardoquesada/esp32-arduino-lib-builder/master/bluepad32_files/package_esp32_bluepad32_index.json)

###### o Schritt 2: Board installieren und auswählen

**Boardverwalter** --> ‚ESP32 + Bluepad32‘ installieren

**Board:** ESP32 + Bluepad32 --> ESP32 Dev Module

###### o Schritt 3: Bibliothek ESP32Servo installieren

**Bibliotheksverwalter** --> ‚ESP32Servo‘ suchen --> installieren

###### o Schritt 4: RobotKit installieren

**Sketch** --> Bibliothek einbinden --> ZIP-Datei hinzufügen --> RobotKit.zip



## 2.2 Controller koppeln

Euer Roboter und euer PS4-Controller müssen einmalig fest miteinander verknüpft werden. Danach verbinden sie sich immer automatisch.

| Schritt | Was zu tun ist                                                  |
|---------|-----------------------------------------------------------------|
| 1       | 01_Grundfahrt hochladen --> Serial Monitor öffnen (115200 Baud) |
| 2       | BT-MAC des ESP32 ablesen (wird automatisch angezeigt)           |
| 3       | sixaxispaier.exe starten (sixaxispaier.sourceforge.net)         |
| 4       | Controller per USB an PC anschließen                            |
| 5       | ESP32-MAC in sixaxispaier eintragen--> ‚Update‘ klicken         |
| 6       | USB abziehen --> PS-Taste am Controller 3 Sek. halten           |
| 7       | Serial Monitor zeigt ‚Controller verbunden‘                     |
| 8       | Angezeigte Controller-MAC in RobotKit_Config.h eintragen        |

**!!! Achtung !!!**

Jedes Team hat einen eigenen Roboter und Controller. Tragt die MAC eures Roboters in RobotKit\_Config.h ein – sonst verbindet sich euer Controller mit dem falschen Roboter!

## 2.3 Roboter-Nummer einstellen

Öffnet die Datei src/RobotKit\_Config.h und tragt eine zufällige, euch bekannte IP ein:

```
#define BT_MAC { 0x05, 0x00, 0x0A, 0x03, 0x00, 0x01 } // Roboter Beispiel
```

## 3. Fahrbefehle

| Befehl                | Beschreibung                 | Beispiel              |
|-----------------------|------------------------------|-----------------------|
| robot.vorwaerts(70)   | Vorwärts fahren, 0–100%      | robot.vorwaerts(80)   |
| robot.rueckwaerts(50) | Rückwärts fahren, 0–100%     | robot.rueckwaerts(60) |
| robot.dreheLinks(60)  | Auf der Stelle links drehen  | robot.dreheLinks(50)  |
| robot.dreheRechts(60) | Auf der Stelle rechts drehen | robot.dreheRechts(50) |
| robot.kurve(l, r)     | Links/Rechts je -100..+100   | robot.kurve(80, 40)   |
| robot.bremse()        | Aktiv bremsen (hält an)      | robot.bremse()        |
| robot.stopp()         | Freilauf (rollt aus)         | robot.stopp()         |

**kurve()** ist die flexibelste Funktion – sie kann alles:

```
robot.kurve( 70, 70); // geradeaus vorwärts
```

```
robot.kurve(-70, -70); // geradeaus rückwärts
```

```
robot.kurve( 70, -70); // auf der Stelle rechts drehen
```

```
robot.kurve( 80, 40); // Rechtskurve vorwärts
```



## 4. Controller

### 4.1 Joysticks

| Methode                      | Bereich       | Bedeutung                    |
|------------------------------|---------------|------------------------------|
| controller.joystickLinksY()  | -100 bis +100 | vorwärts (+) / rückwärts (-) |
| controller.joystickLinksX()  | -100 bis +100 | links (-) / rechts (+)       |
| controller.joystickRechtsY() | -100 bis +100 | vorwärts (+) / rückwärts (-) |
| controller.joystickRechtsX() | -100 bis +100 | links (-) / rechts (+)       |

### 4.2 Tasten (true = gedrückt)

| Methode                 | PS4-Taste  | Methode                | Taste                     |
|-------------------------|------------|------------------------|---------------------------|
| controller.kreuz()      | X (unten)  | controller.L1()        | Linke Schulter            |
| controller.kreis()      | O (rechts) | controller.R1()        | Rechte Schulter           |
| controller.dreieck()    | Δ (oben)   | controller.L2()        | Linker Trigger (digital)  |
| controller.viereck()    | □ (links)  | controller.R2()        | Rechter Trigger (digital) |
| controller.dpadOben()   | D-Pad ▲    | controller.l2Wert()    | Linker Trigger 0–100      |
| controller.dpadUnten()  | D-Pad ▼    | controller.r2Wert()    | Rechter Trigger 0–100     |
| controller.dpadLinks()  | D-Pad ◀    | controller.options()   | Options-Taste             |
| controller.dpadRechts() | D-Pad ▶    | controller.verbunden() | true = verbunden          |

### 4.3 Callbacks – auf Tastendruck reagieren

Callbacks werden genau einmal ausgelöst, wenn eine Taste gedrückt wird – nicht bei gedrückt halten. Ideal für Aktionen wie Notstopp:

**// In setup() eintragen:**

```
controller.beiKreuz([]) {
    robot.stopp();
    Serial.println(„Notstopp!“);
};
```

```
controller.beiDreieck([]) {
    meineFunktion(); // eigene Funktion aufrufen
};
```

**// Verfügbare Callbacks:**

```
controller.beiKreuz()   controller.beiKreis()   controller.beiDreieck()   controller.beiViereck()
controller.beiL1()     controller.beiR1()   controller.beiOptions()
```



## 5. Servos

Zwei Servos sind direkt nach robot.begin() nutzbar – kein extra Schritt nötig.

Die Servos sind typabhängig nur Servos mit 90° maximalem Drehwinkel.

| Befehl                       | Beschreibung                    |
|------------------------------|---------------------------------|
| robot.servo1.setzeWinkel(90) | Winkel setzen: 0–180 Grad       |
| robot.servo1.drehe(+5)       | Relativ drehen (+/- Grad)       |
| robot.servo1.mitte()         | Mittelstellung: 90°             |
| robot.servo1.links()         | Endstellung links: 0°           |
| robot.servo1.rechts()        | Endstellung rechts: 180°        |
| robot.servo1.getWinkel()     | Aktuellen Winkel abfragen       |
| robot.servo2.xxx()           | Zweiter Servo – gleiche Befehle |

// Beispiel: Servo mit D-Pad steuern

```
if (controller.dpadLinks()) robot.servo1.drehe(-3);
if (controller.dpadRechts()) robot.servo1.drehe(+3);
controller.beiViereck([]) { robot.servo1.mitte(); };
```

## 6. Sensoren auf der Platine

| Sensor        | Befehl                        | Rückgabe          | Verwendung             |
|---------------|-------------------------------|-------------------|------------------------|
| Potentiometer | robot lesePoti()              | 0–100 %           | Geschwindigkeit regeln |
| Taster        | robot.tasterGedrueckt()       | true / false      | Aktionen auslösen      |
| Batterie      | robot leseBatterie()          | Volt (z.B. 11.4)  | Ladestand überwachen   |
| Batterie      | robot.batterieSchwach()       | true / false      | Warnung anzeigen       |
| Batterie      | robot.getBatterieZustand()    | BATT_OK / ...     | Stufe abfragen         |
| Batterie      | robot.getMaxGeschwindigkeit() | 0 / 50 / 80 / 100 | Autom. Begrenzung      |

**Denkt bitte daran die Batterie abzufragen und zum Beispiel das Gamepad rot leuchten zu lassen wenn die Batterie schwächer wird. Wenn die Batterien tiefenentladen werden sind sie kaputt. Bitte informiert euch hier zu Li-Ionen Akkus!**



## 7. Antriebsmodi

| Modus        | Beschreibung                      | Wann nutzen?                            |
|--------------|-----------------------------------|-----------------------------------------|
| ANTRIEB_2RAD | Motor C = links, Motor D = rechts | Standard – 2-Rad Differentialantrieb    |
| ANTRIEB_4RAD | C+A = links, D+B = rechts         | 4-Rad Antrieb                           |
| ANTRIEB_FREI | Alle 4 Motoren einzeln ansteuern  | Fortgeschrittene / Sonderkonstruktionen |

## 8. Beispiel-Sketches

| Beispiel                | Was es zeigt                                                              | Ideal für ...             |
|-------------------------|---------------------------------------------------------------------------|---------------------------|
| 01_Grundfahrt           | Fahren mit Joysticks, Notstopp, Geschwindigkeit per R1/L1, Batterieschutz | Einstieg – hier anfangen! |
| 02_Poti_Geschwindigkeit | Poti auf der Platine regelt die Fahrgeschwindigkeit                       | Analoge Eingabe verstehen |
| 03_Taster_Aktion        | Taster/Dreieck löst autonome Fahrsequenz aus                              | Einfache Autonomie        |
| 04_Servo                | Servo 1 mit D-Pad, Servo 2 mit Oben/Unten                                 | Servo-Ansteuerung         |

## 9. Debugging & häufige Fehler

### 9.1 Debug-Ausgabe

`robot.debugPrint();` // gibt alle Werte im Serial Monitor aus

// Ausgabe:

// === RobotKit Debug ===

// Batterie: 11.40 V

// Poti: 65 %

// Taster: offen

// Motor A: 0

// Motor B: 0

// Servo 1: 90 Grad



## 10. Pinbelegung

| Funktion                | GPIO         | Hinweis                            |
|-------------------------|--------------|------------------------------------|
| Batterie (VP)           | 36           | Nur Input – kein Pull-Up           |
| Poti (VN)               | 39           | Nur Input – kein Pull-Up           |
| Taster                  | 34           | Nur Input – externer Pull-Up nötig |
| Motor C PWM / IN1 / IN2 | 14 / 15 / 13 | Linker Antriebsmotor               |
| Motor D PWM / IN1 / IN2 | 32 / 27 / 33 | Rechter Antriebsmotor              |
| Motor A PWM / IN1 / IN2 | 18 / 22 / 23 | Zusatzmotor / 4-Rad                |
| Motor B PWM / IN1 / IN2 | 17 / 21 / 19 | Zusatzmotor / 4-Rad                |
| I2C SCL / SDA           | 25 / 26      | Für externe Sensoren               |
| Servo 1 / Servo 2       | 16 / 4       | ESP32Servo Bibliothek              |

Alle Pins können in src/RobotKit\_Config.h angepasst werden.

