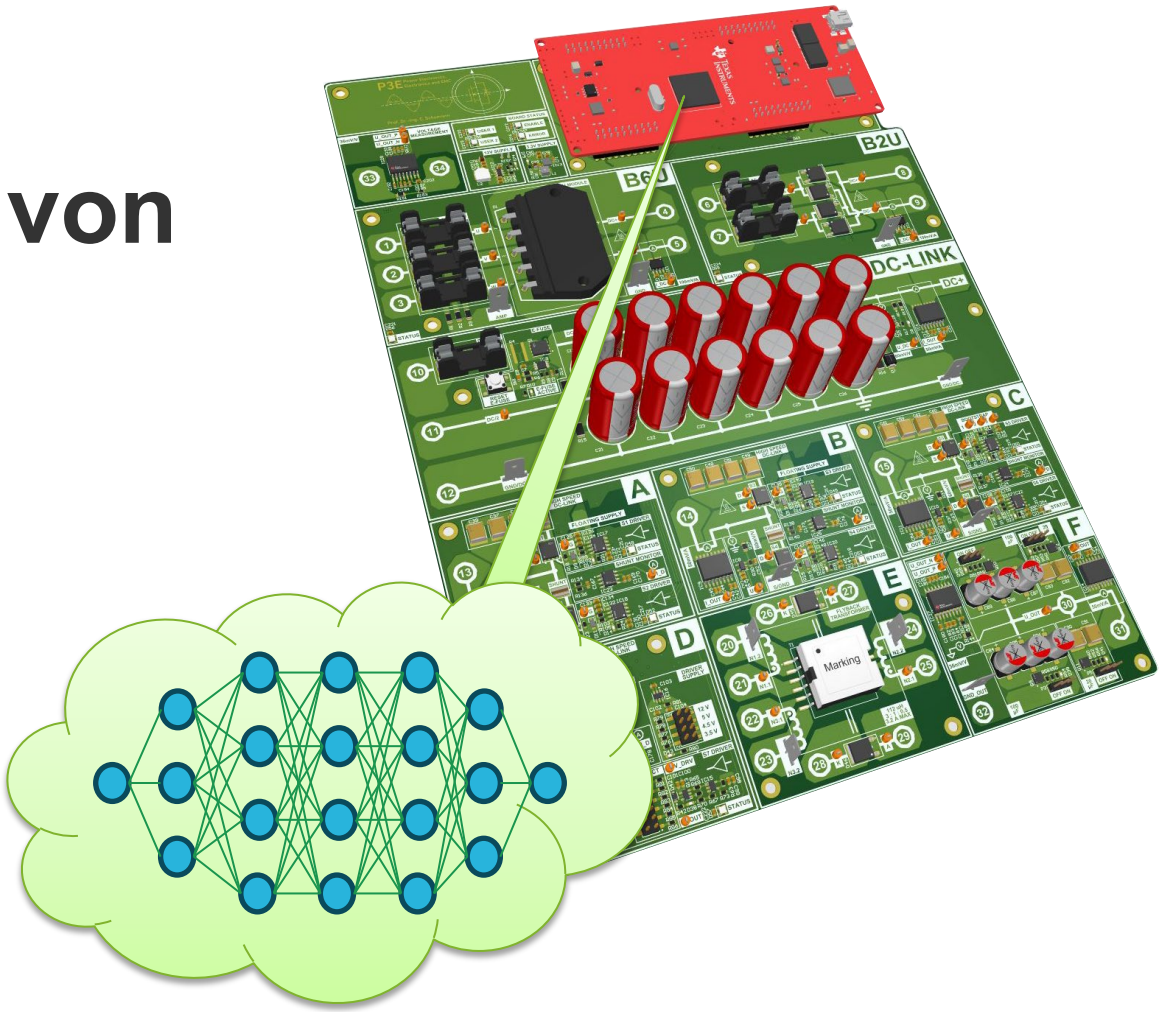
Hochschule Kaiserslautern
University of Applied Sciences

Angewandte Ingenieurwissenschaften Kaiserslautern

P3E Power Electronics, Electronics and EMC

Prof. Dr.-Ing. C. Schumann

KI-basierte Ansteuerung von leistungselektronischen Schaltungen als Laborpraktikum

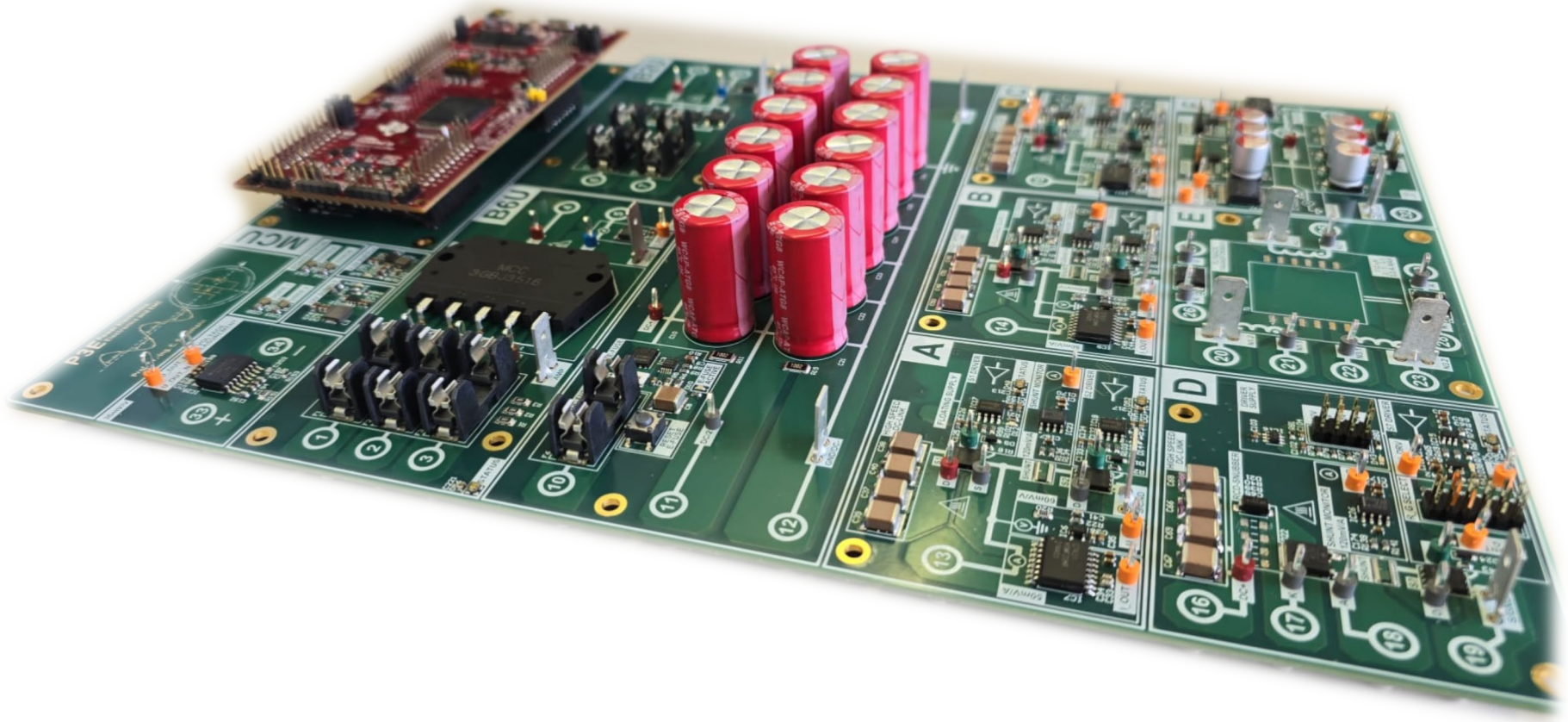


Jennifer Piela, B. Eng., Prof. Dr.-Ing. C. Schumann

03.06.2025

Unsere Multiversuchshardware

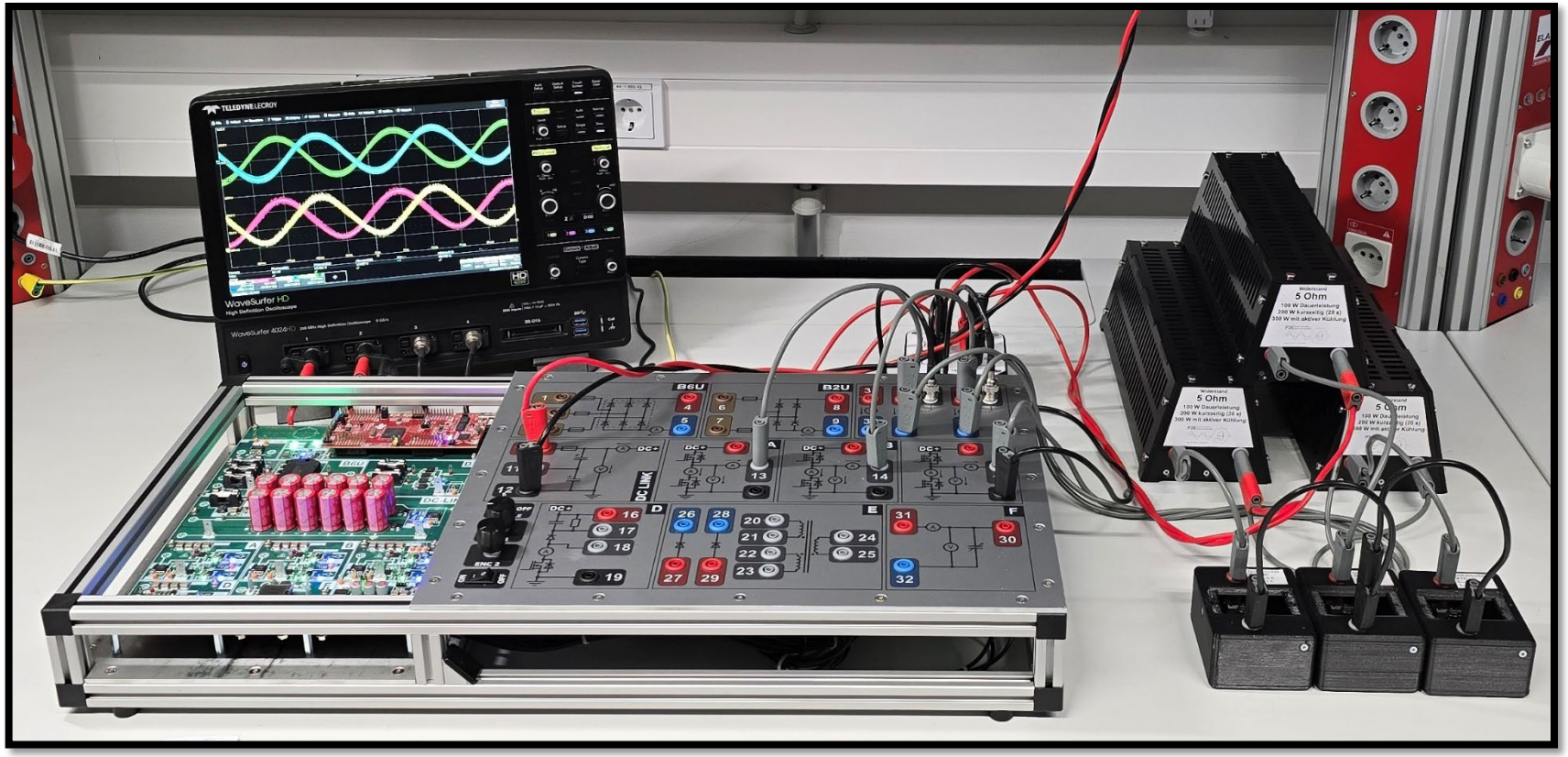
- Vorangehendes SQL-Projekt
- Selbst entworfenes Design
- Einzigartige Hardware
- Modulare Auslegung



Multiversuchshardware im Labor

P3E Power Electronics, Electronics and EMC
Prof. Dr.-Ing. C. Schumann
Hochschule Kaiserslautern University of Applied Sciences

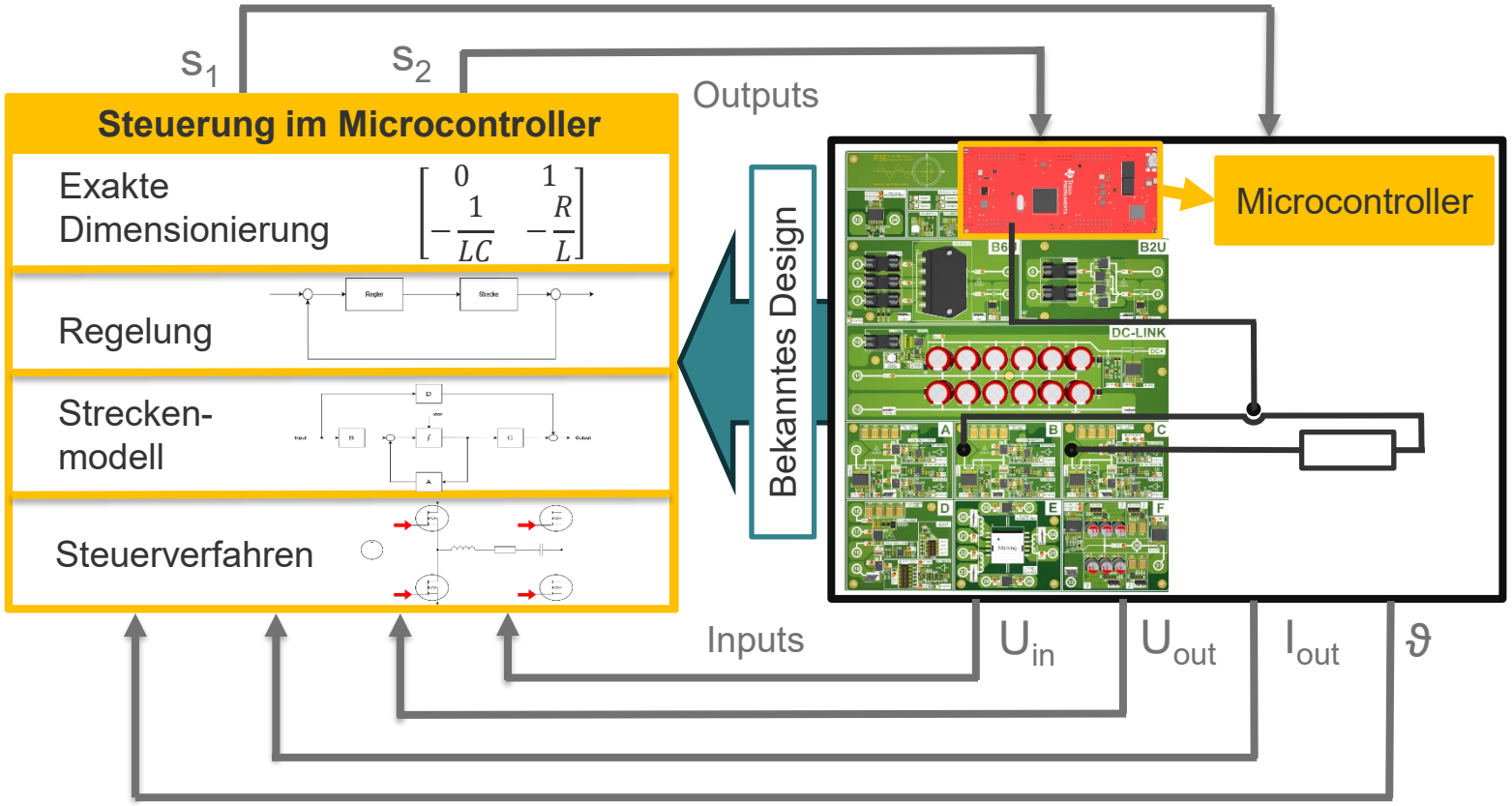
- Einsatz im Leistungselektronik-Labor
- Vier Versuche – eine Hardware
- Gutes Feedback von Studierenden



Leistungselektronische Anwendung

Klassische Regelanwendung :

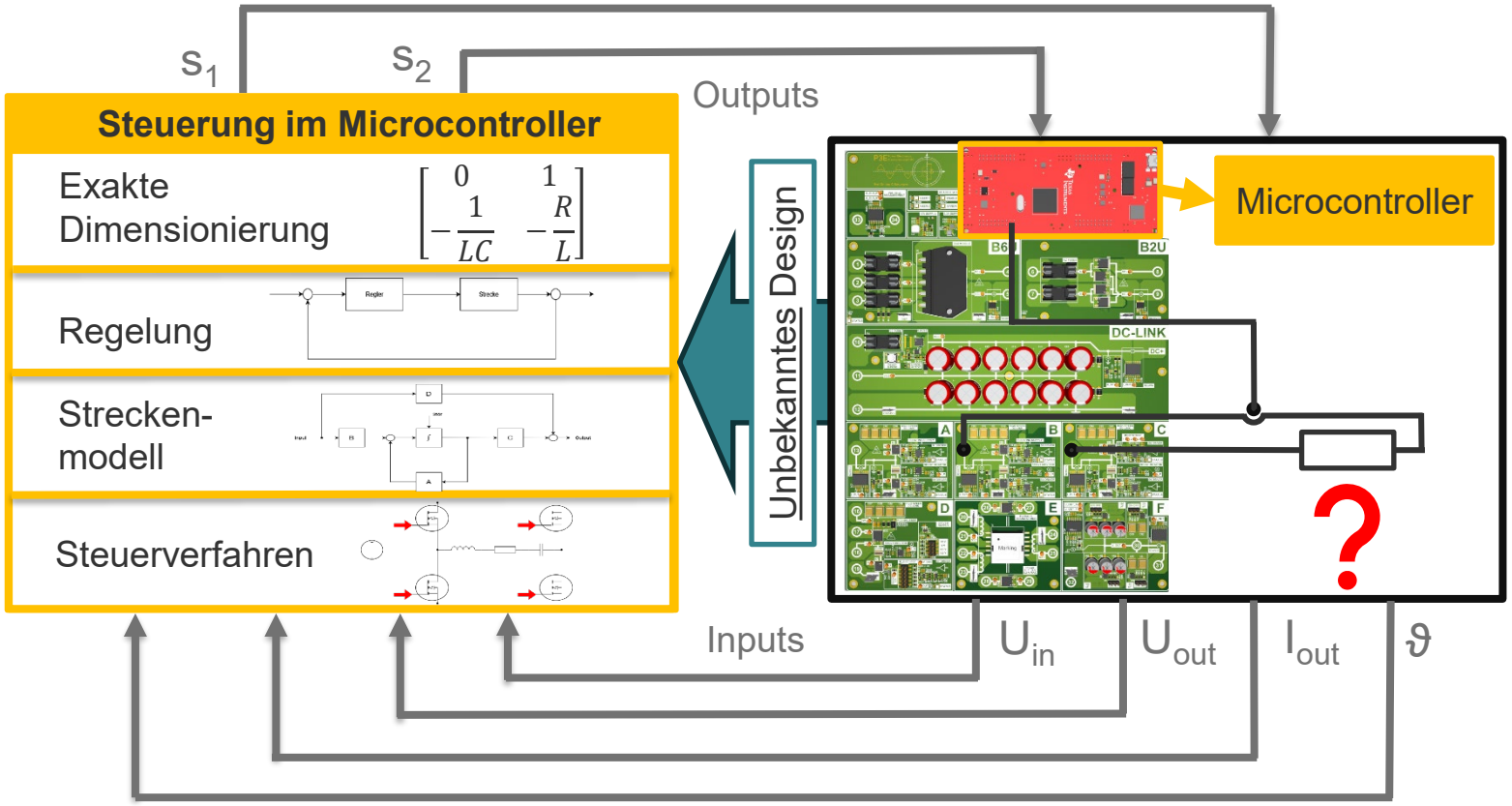
- Geeignet für bekannte Systeme
- Für einen Arbeitspunkt optimal
- Schnelle Reaktionszeit



Leistungselektronische Anwendung

Klassische Regelanwendung :

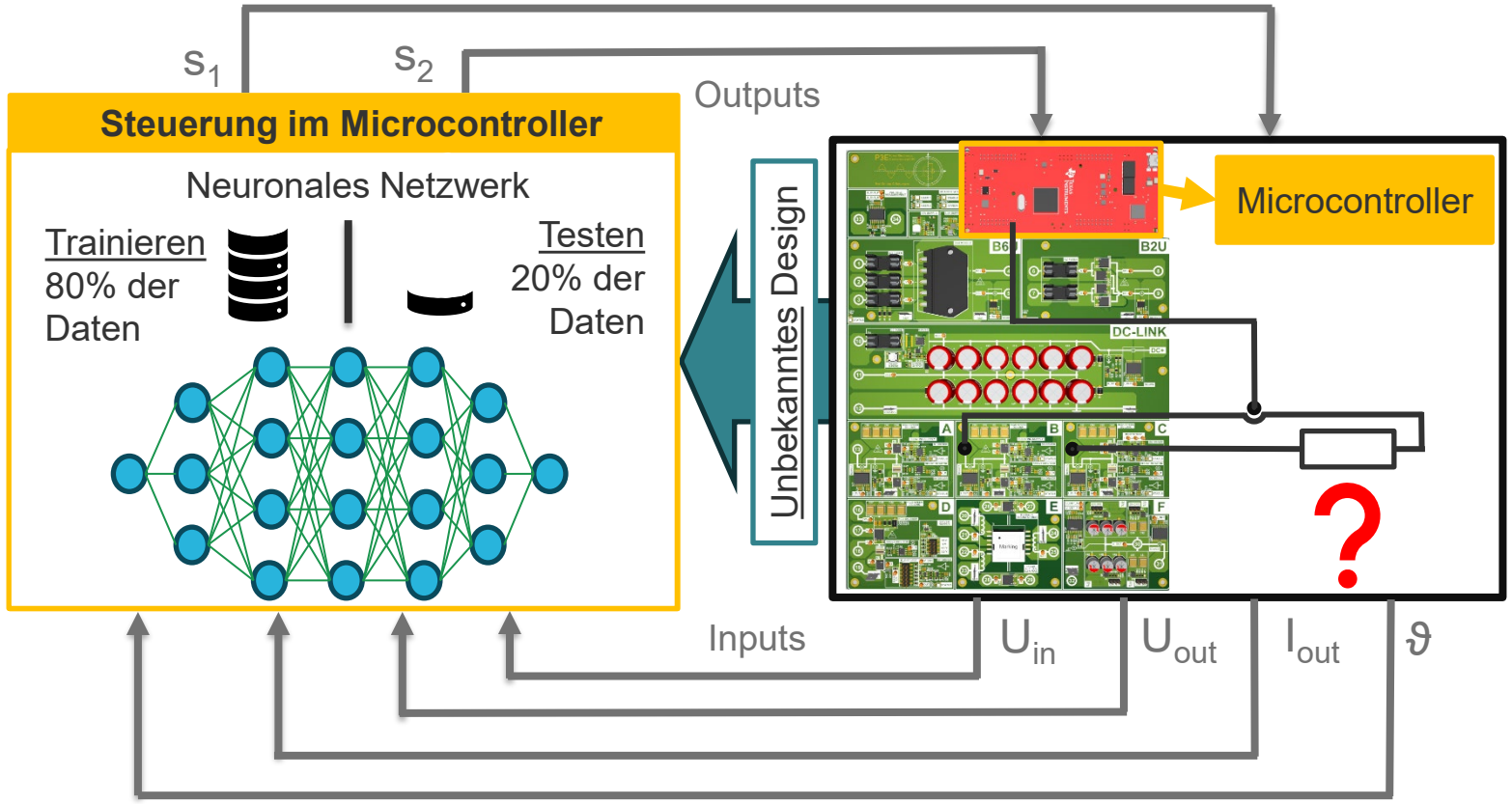
- Suboptimal für neue Systeme
- Neue Auslegung notwendig



Leistungselektronische Anwendung

KI Regelanwendung :

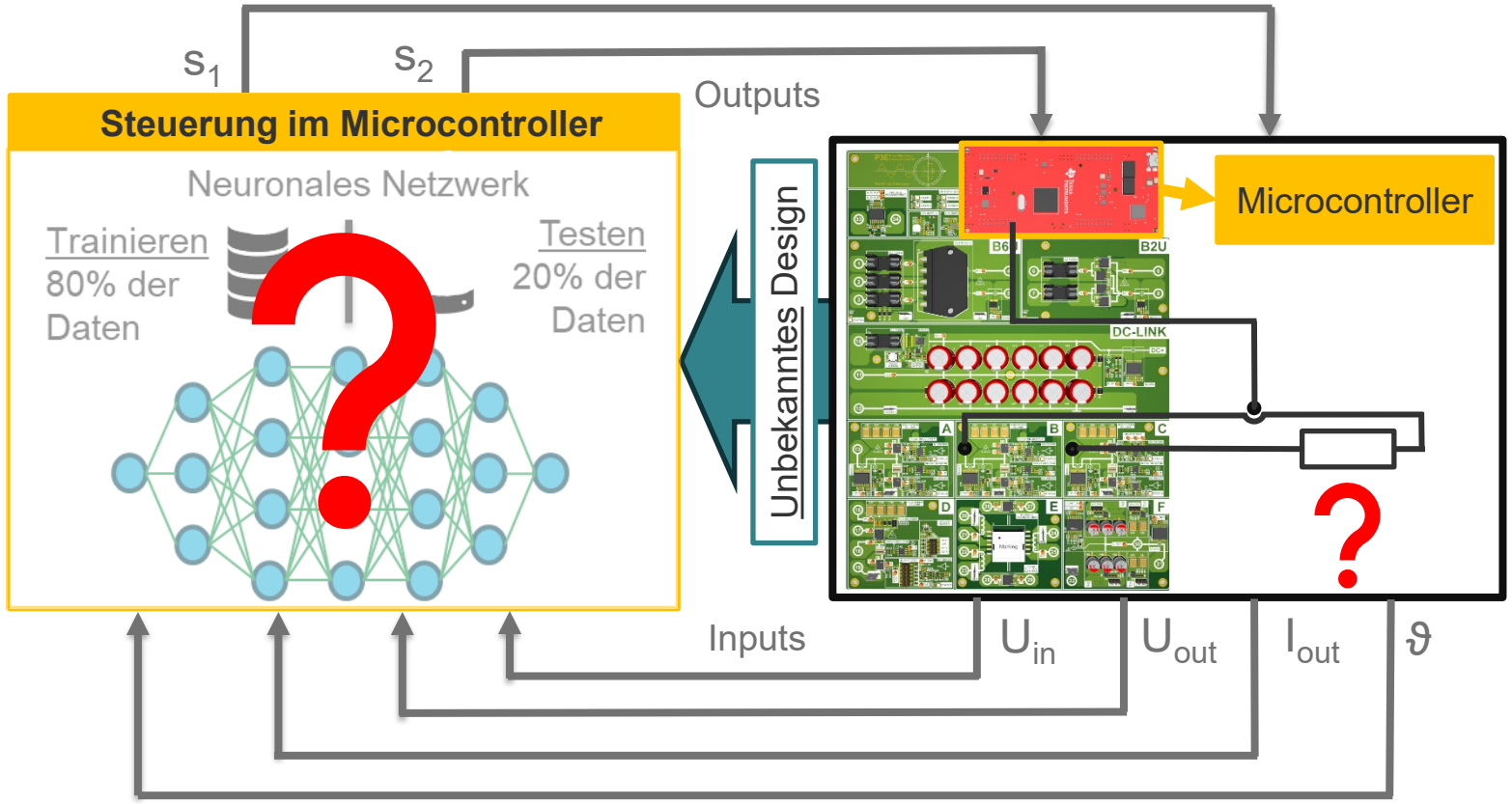
- Selbstlernendes Steuerverfahren
- Optimale Regelung für jeden Arbeitspunkt
- verbesserungsfähig



Leistungselektronische Anwendung

Voraussetzungen an Studierende:

- Grundkenntnisse zu KI
- Grundkenntnisse in der Programmierung
- Grundkenntnisse von leistungselektronischen Schaltungen

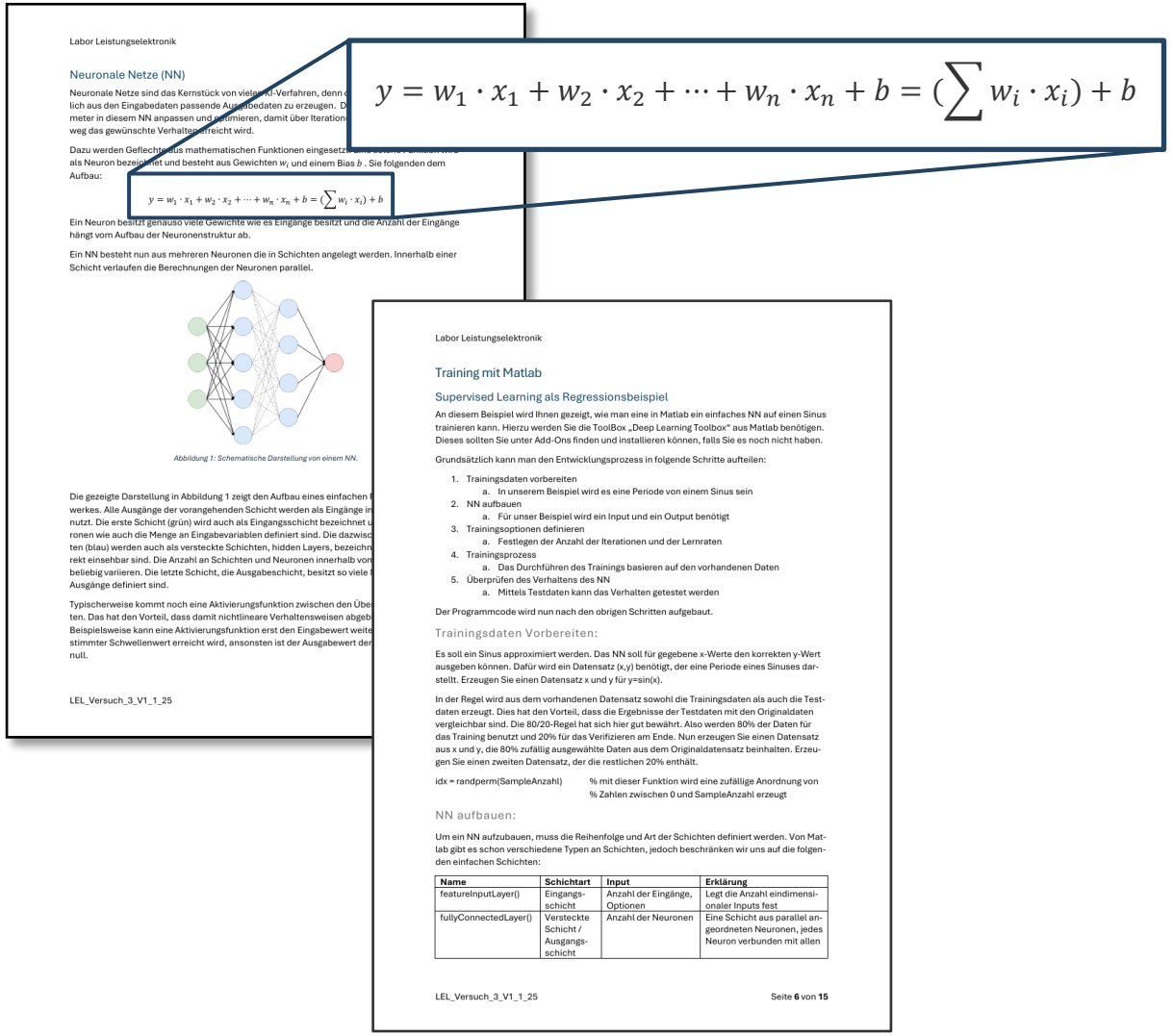


Grundkenntnisse von KI

P3E Power Electronics, Electronics and EMC
Prof. Dr.-Ing. C. Schumann
Hochschule Kaiserslautern University of Applied Sciences

<u>Wiederholung von KI-Grundkenntnissen</u>:

- Begriffserklärung
- Eigenrecherche zu KI-Lerntypen
- Schritt-für-Schritt-Anleitung:
  - Einfache NN trainieren
  - Einfluss der Trainingsdaten auf Lernprozess
  - Einfluss von Trainingsparameter



Labor Leistungselektronik

Neuronale Netze (NN)

Neuronale Netze sind das Kernstück von vielen KI-Verfahren, denn d… lich aus den Eingabedaten passende Ausgabedaten zu erzeugen. D… meter in diesem NN anpassen und optimieren, damit über Iteration… weg das gewünschte Verhalten erreicht wird.

Dazu werden Geflechte aus mathematischen Funktionen eingesetzt… als Neuron bezeichnet und besteht aus Gewichten w_i und einem Bias b. Sie folgenden dem Aufbau:

$$y = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + b = (\sum w_i \cdot x_i) + b$$

Ein Neuron besitzt genauso viele Gewichte wie es Eingänge besitzt und die Anzahl der Eingänge hängt vom Aufbau der Neuronenstruktur ab.

Ein NN besteht nun aus mehreren Neuronen die in Schichten angelegt werden. Innerhalb einer Schicht verlaufen die Berechnungen der Neuronen parallel.

Abbildung 1: Schematische Darstellung von einem NN.

Die gezeigte Darstellung in Abbildung 1 zeigt den Aufbau eines einfachen … werkes. Alle Ausgänge der vorangehenden Schicht werden als Eingänge in… nutzt. Die erste Schicht (grün) wird auch als Eingangsschicht bezeichnet u… ronen wie auch die Menge an Eingabevariablen definiert sind. Die dazwisc… ten (blau) werden auch als versteckte Schichten, hidden Layers, bezeichn… rekt einsehbar sind. Die Anzahl an Schichten und Neuronen innerhalb von… beliebig variieren. Die letzte Schicht, die Ausgabeschicht, besitzt so viele … Ausgänge definiert sind.

Typischerweise kommt noch eine Aktivierungsfunktion zwischen den Übe… ten. Das hat den Vorteil, dass damit nichtlineare Verhaltensweisen abgeb… Beispielsweise kann eine Aktivierungsfunktion erst den Eingabewert weite… stimmter Schwellenwert erreicht wird, ansonsten ist der Ausgabewert der… null.

LEL_Versuch_3_V1_1_25

Labor Leistungselektronik

Training mit Matlab

Supervised Learning als Regressionsbeispiel

An diesem Beispiel wird Ihnen gezeigt, wie man eine in Matlab ein einfaches NN auf einen Sinus trainieren kann. Hierzu werden Sie die ToolBox „Deep Learning Toolbox" aus Matlab benötigen. Dieses sollten Sie unter Add-Ons finden und installieren können, falls Sie es noch nicht haben.

Grundsätzlich kann man den Entwicklungsprozess in folgende Schritte aufteilen:

1. Trainingsdaten vorbereiten
   a. In unserem Beispiel wird es eine Periode von einem Sinus sein
2. NN aufbauen
   a. Für unser Beispiel wird ein Input und ein Output benötigt
3. Trainingsoptionen definieren
   a. Festlegen der Anzahl der Iterationen und der Lernraten
4. Trainingsprozess
   a. Das Durchführen des Trainings basieren auf den vorhandenen Daten
5. Überprüfen des Verhaltens des NN
   a. Mittels Testdaten kann das Verhalten getestet werden

Der Programmcode wird nun nach den obigen Schritten aufgebaut.

Trainingsdaten Vorbereiten:

Es soll ein Sinus approximiert werden. Das NN soll für gegebene x-Werte den korrekten y-Wert ausgeben können. Dafür wird ein Datensatz (x,y) benötigt, der eine Periode eines Sinuses darstellt. Erzeugen Sie einen Datensatz x und y für y=sin(x).

In der Regel wird aus dem vorhandenen Datensatz sowohl die Trainingsdaten als auch die Testdaten erzeugt. Dies hat den Vorteil, dass die Ergebnisse der Testdaten mit den Originaldaten vergleichbar sind. Die 80/20-Regel hat sich hier gut bewährt. Also werden 80% der Daten für das Training benutzt und 20% für das Verifizieren am Ende. Nun erzeugen Sie einen Datensatz aus x und y, die 80% zufällig ausgewählte Daten aus dem Originaldatensatz beinhalten. Erzeugen Sie einen zweiten Datensatz, der die restlichen 20% enthält.

```
idx = randperm(SampleAnzahl)        % mit dieser Funktion wird eine zufällige Anordnung von
                                    % Zahlen zwischen 0 und SampleAnzahl erzeugt
```

NN aufbauen:

Um ein NN aufzubauen, muss die Reihenfolge und Art der Schichten definiert werden. Von Matlab gibt es schon verschiedene Typen an Schichten, jedoch beschränken wir uns auf die folgenden einfachen Schichten:

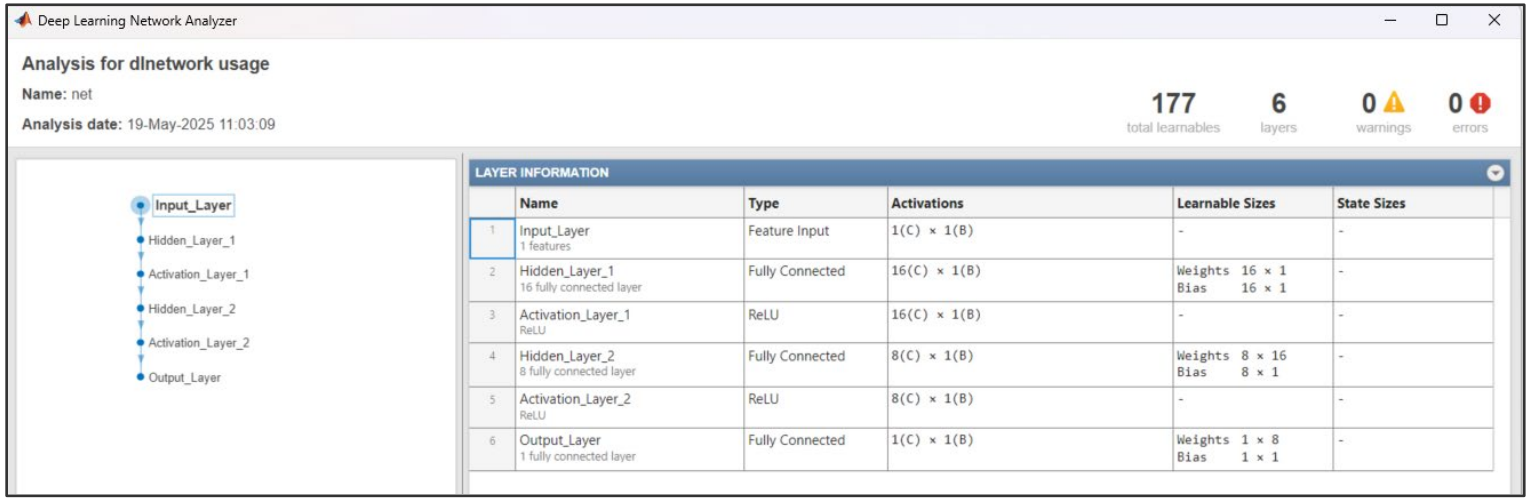
Name	Schichtart	Input	Erklärung
featureInputLayer()	Eingangs-schicht	Anzahl der Eingänge, Optionen	Legt die Anzahl eindimensionaler Inputs fest
fullyConnectedLayer()	Versteckte Schicht / Ausgangs-schicht	Anzahl der Neuronen	Eine Schicht aus parallel angeordneten Neuronen, jedes Neuron verbunden mit allen

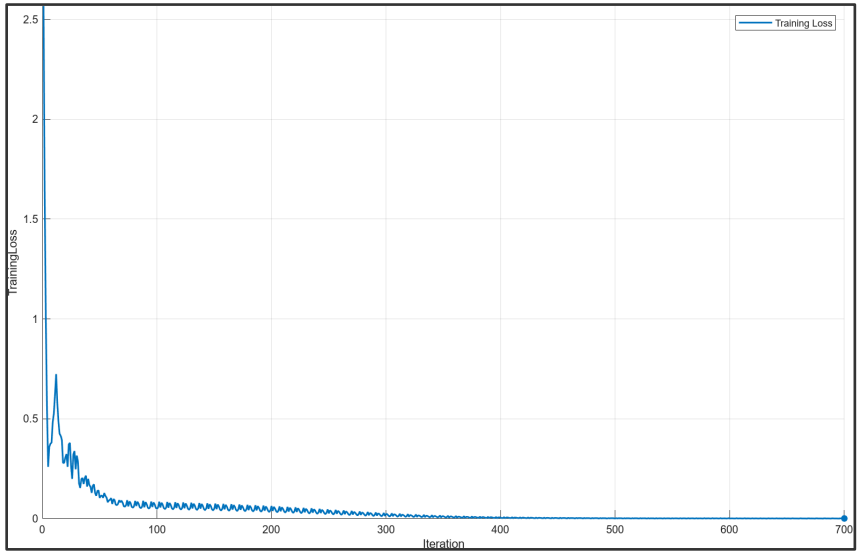
LEL_Versuch_3_V1_1_25 Seite 6 von 15

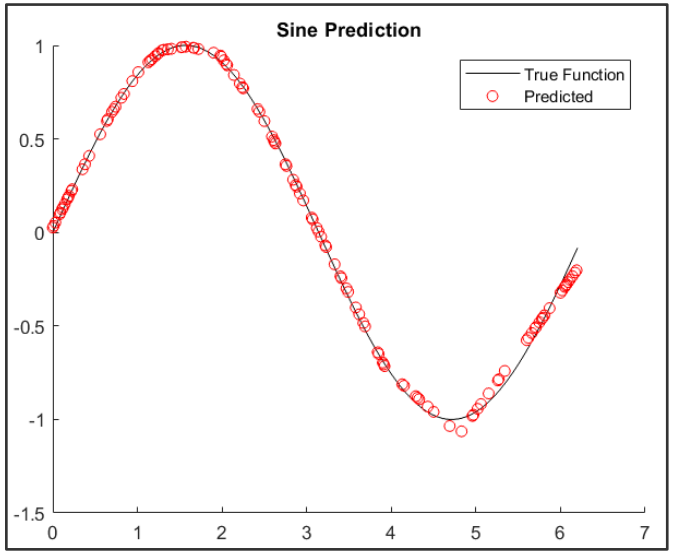
Grundkenntnisse von KI

Visualisierung durch MatLab möglich:

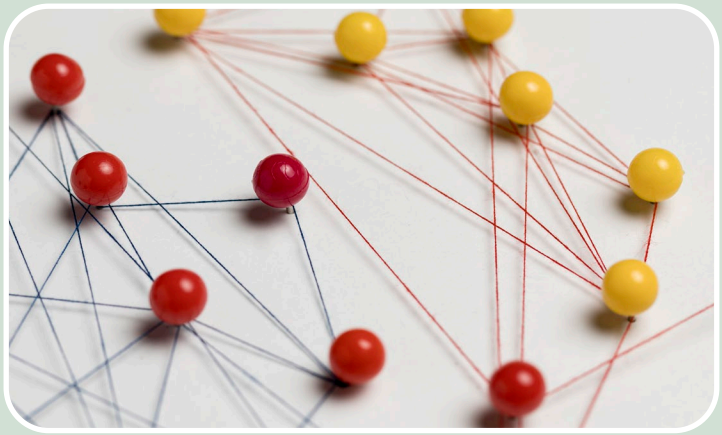
- Netzstruktur mit Parametern einsehbar
- Trainingsablauf verfolgbar
- In wenigen Sekunden bis Minuten trainiert







Welche Arten an KI gibt es?



Supervised Learning

- überwachtes Lernen

Unsupervised Learning

- unbeaufsichtigtes Lernen

Reinforcement Learning

- verstärktes Lernen

Wie funktioniert Supervised Learning?

Training:

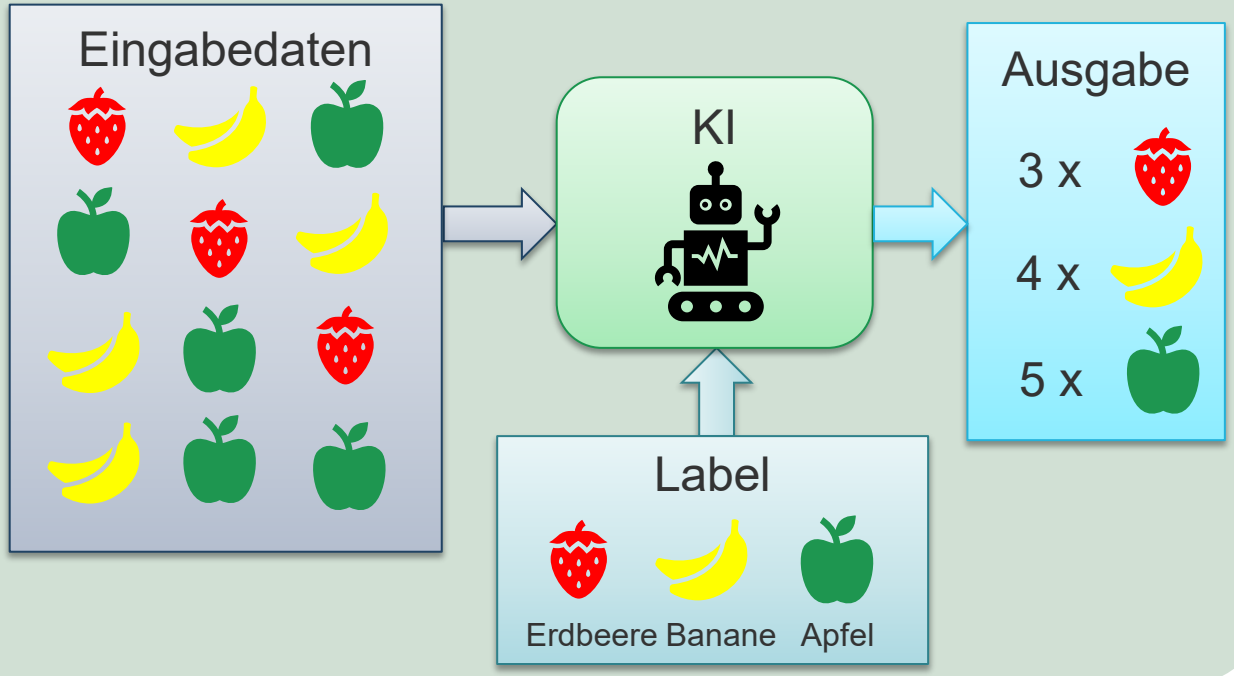
- Eingabedaten (Features) und passende Ausgaben (Labels)
- lernt Zusammenhänge zwischen Eingabe und Ausgabe

Ziel:

- Vorhersage Ausgabe von unbekannten Daten

Anwendung:

- Einsatz für Klassifikations- oder Regressionsaufgaben



Supervised Learning
- überwachtes Lernen

Unsupervised Learning
- unbeaufsichtigtes Lernen

Reinforcement Learning
- verstärktes Lernen

Wie funktioniert Unsupervised Learning?

Training:

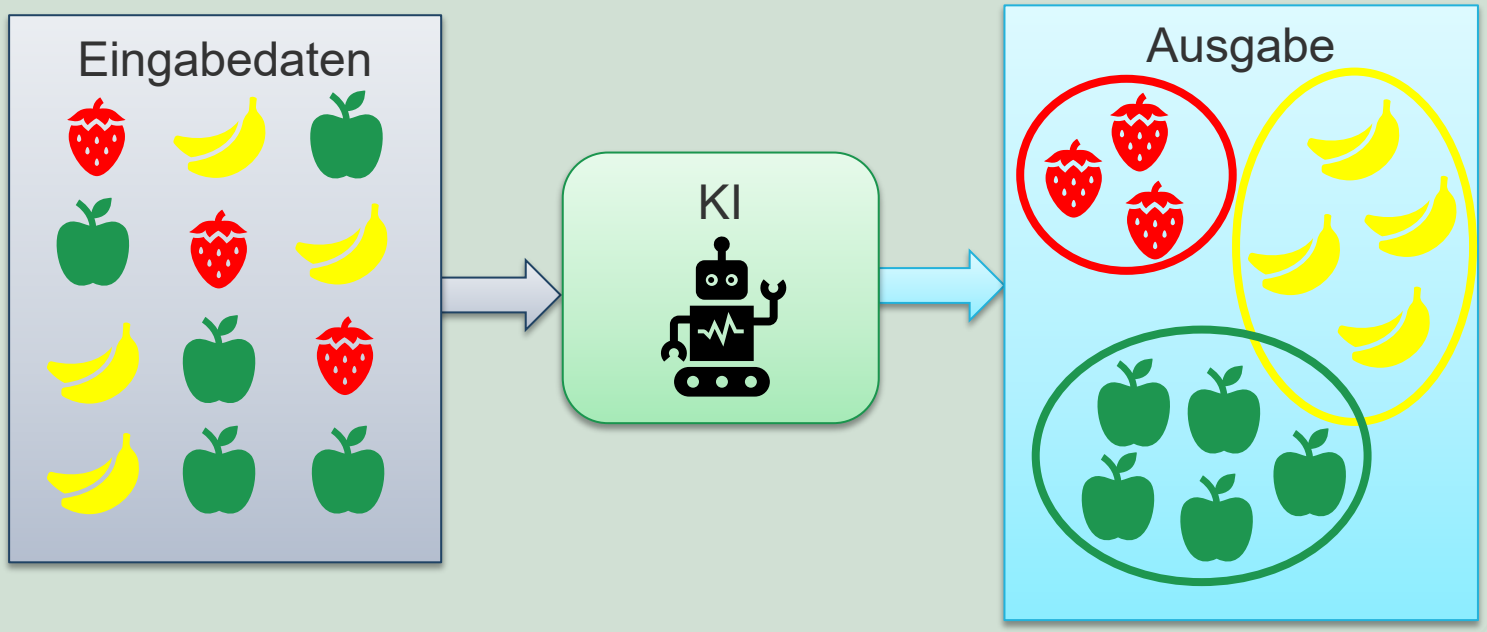
- nur die Eingabedaten, keine Zielwerte

Ziel:

- Mustererkennung
- Gruppieren oder Clustern ähnlicher Daten

Anwendungen:

- Gruppierung, Anomalieerkennung



Supervised Learning

- überwachtes Lernen

Unsupervised Learning

- unbeaufsichtigtes Lernen

Reinforcement Learning

- verstärktes Lernen

Wie funktioniert Reinforcement Learning?

Training:

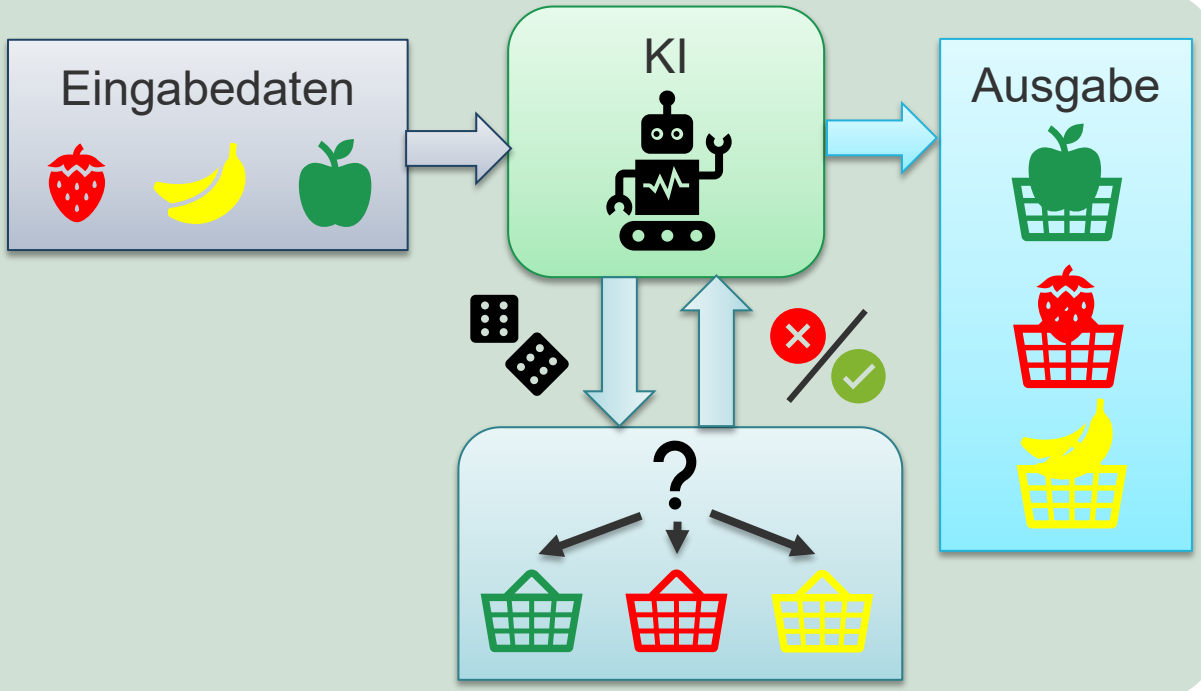
- Lernverfahren durch Ausprobieren von Aktionen in Umgebung
- Feedback in Form von Belohnung oder Strafe (Reward/Penalty)

Ziel:

- Strategie (Policy) entwickeln für langfristig maximale Belohnung

Anwendung:

- Optimierungsprobleme, z.B.: komplexer Steuerung, Kartenspielen



Supervised Learning

- überwachtes Lernen

Unsupervised Learning

- unbeaufsichtigtes Lernen

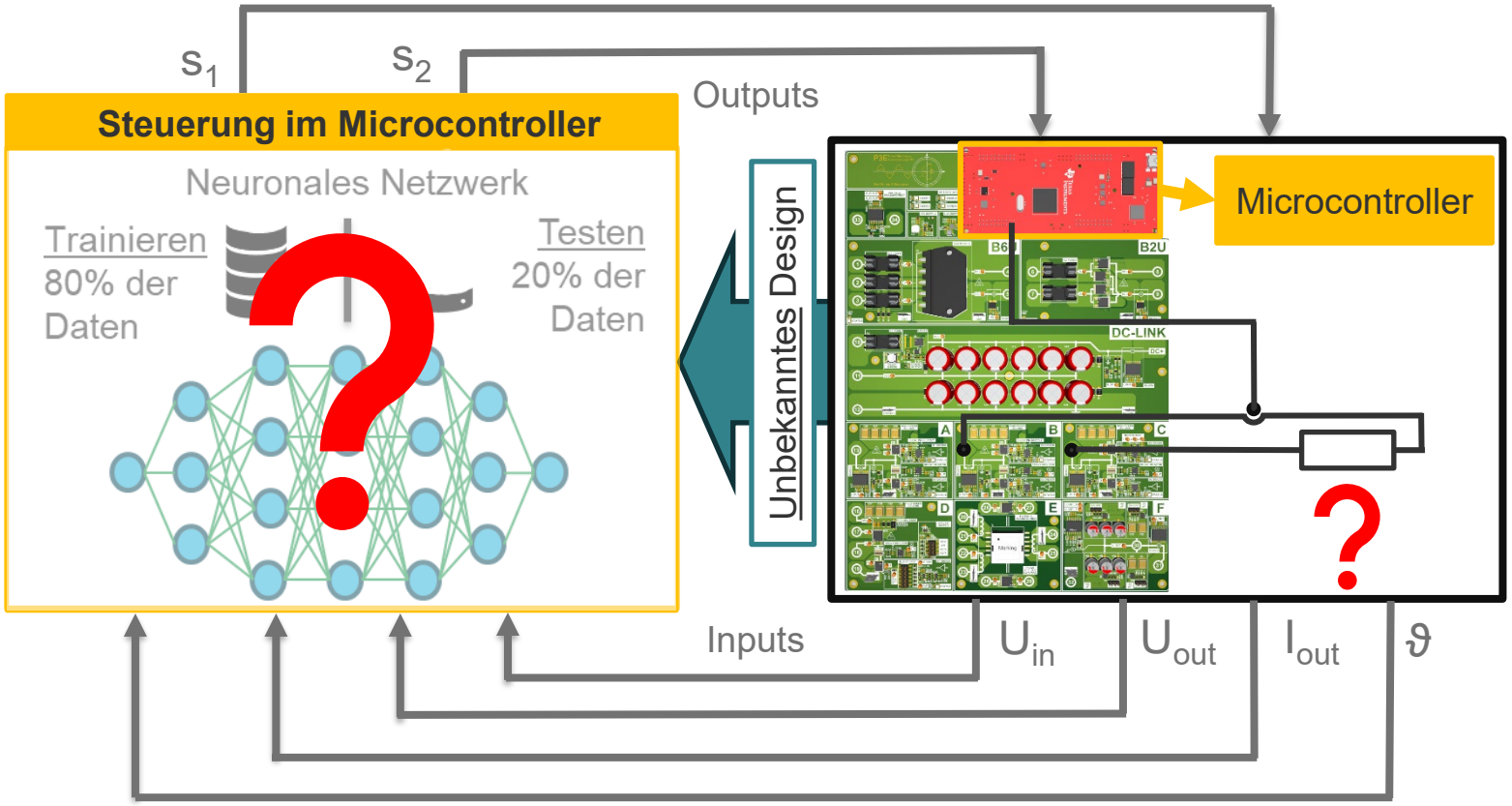
Reinforcement Learning

- verstärktes Lernen

Leistungselektronische Anwendung

Simple Anwendung zu Einstieg:

- Reduzieren der Komplexität

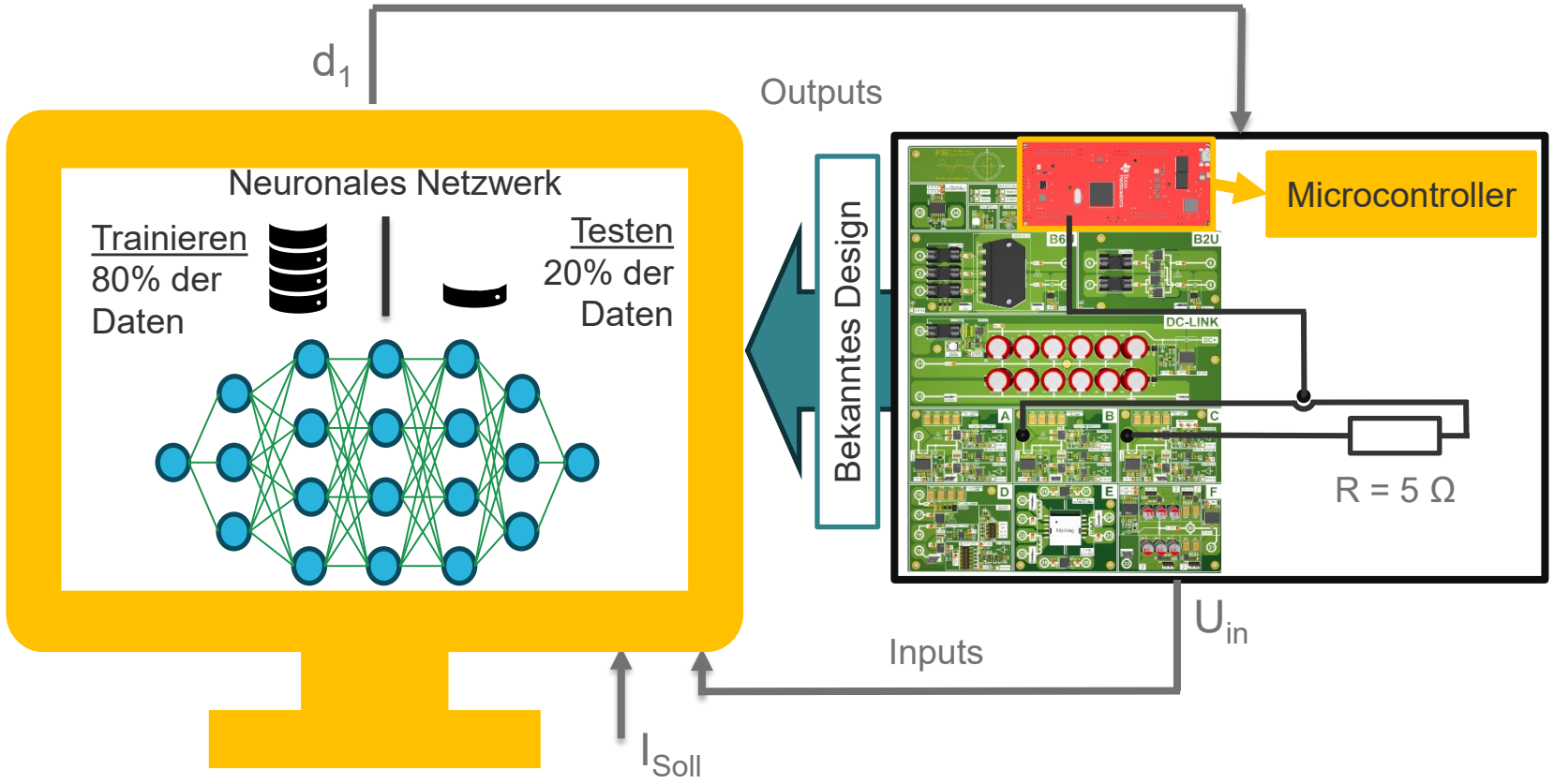


Leistungselektronische Anwendung

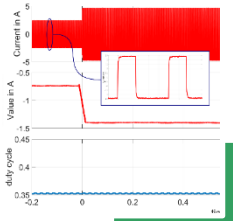
Simple Anwendung zu Einstieg:

- Reduzieren der Komplexität
- Training in Simulationssoftware
- Einfache Steuerung entwerfen

Steuerung im Microcontroller

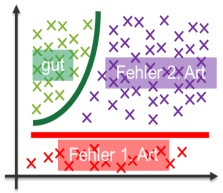


Versuchsaufbauten



Supervised Learning

- Trainierte KI-Steuerung auf Microcontroller
- Operationsgrenzen vom trainiertem NN kennenlernen



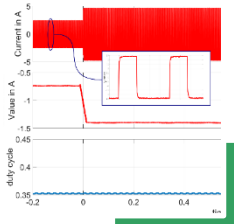
Unsupervised Learning

- Neue Anwendungsfälle kennenlernen: Mustererkennung
- KI soll unterschiedliche Verläufe erkennen

Reinforcement Learning

- Komplexere Lernmethoden kennenlernen
- Optimale Regelung in allen Fällen

KI als Steuerung

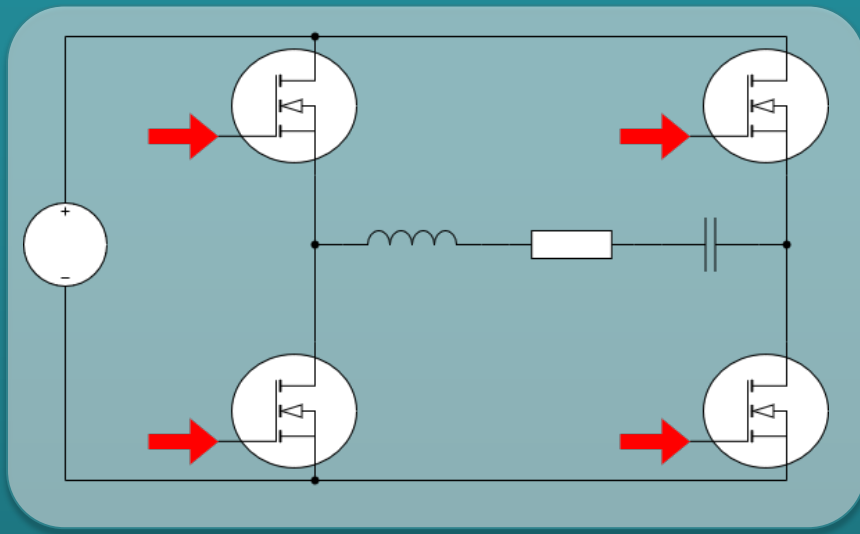


Supervised Learning

- Trainierte KI-Steuerung auf Microcontroller
- Operationsgrenzen vom trainiertem NN kennenlernen

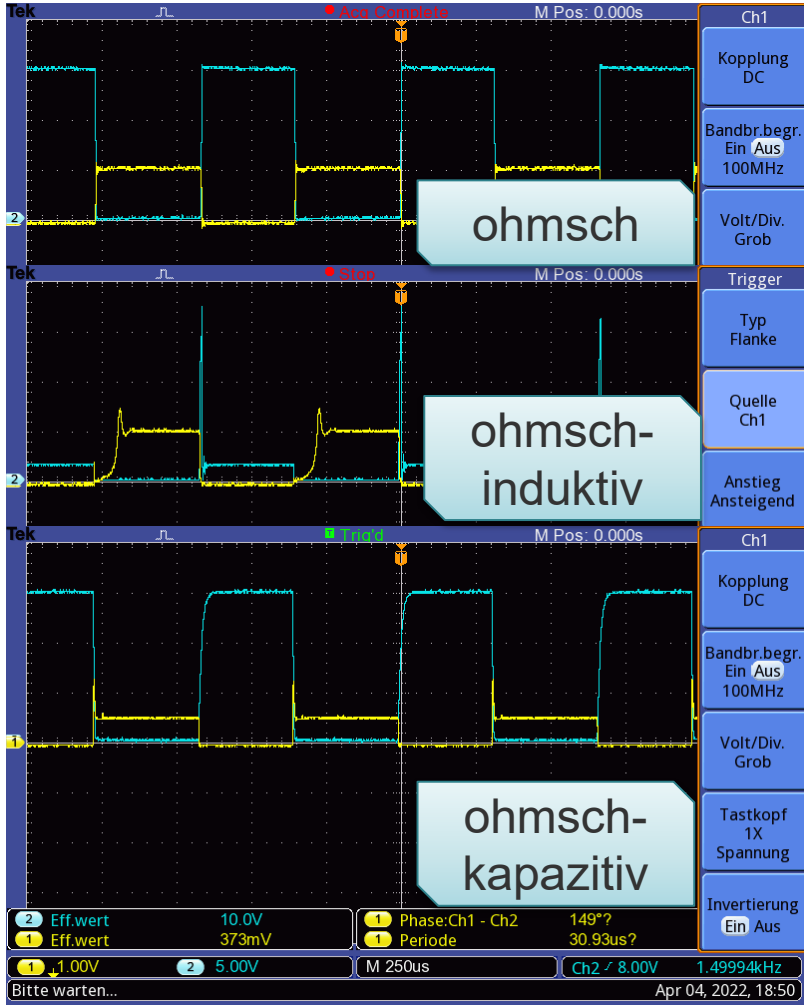
Steuerung der Vollbrücke:

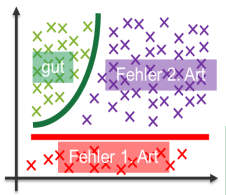
- Lineares Verhalten
- Arbeitsbereich
  - U_{in} = 10 – 25 V
  - I_{Soll} = -2,5 – 2,5 A
  - R = 10 Ω



- Testen unterschiedlich gut trainierter KI-Steuerungen
- Verhalten innerhalb und außerhalb des Arbeitsbereichs
- Verändern der Last

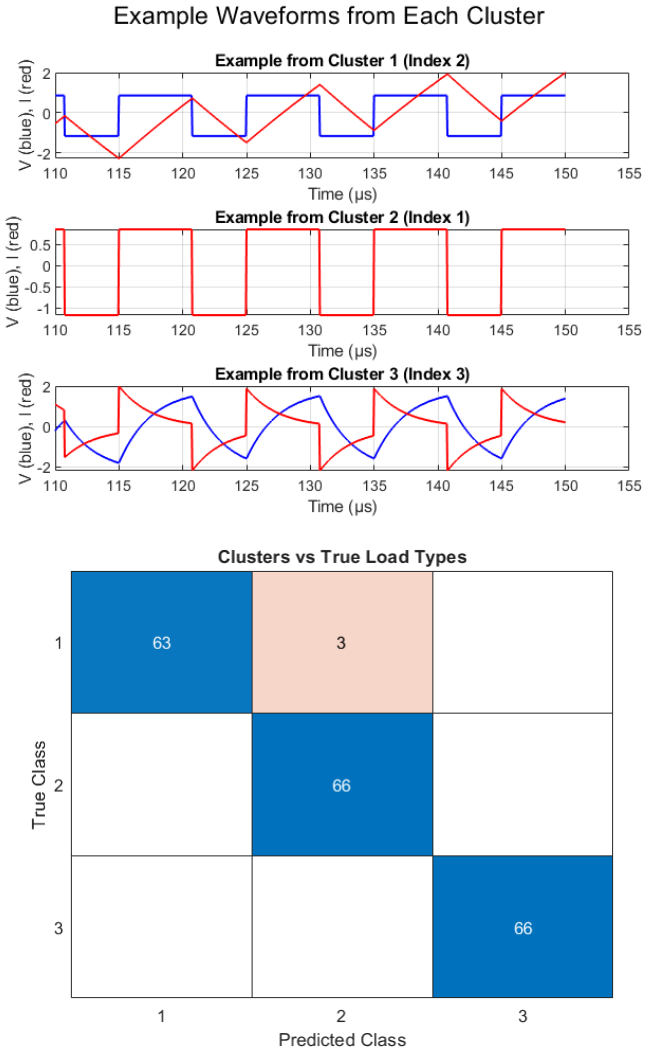
KI als Datenanalyse (Feedback)



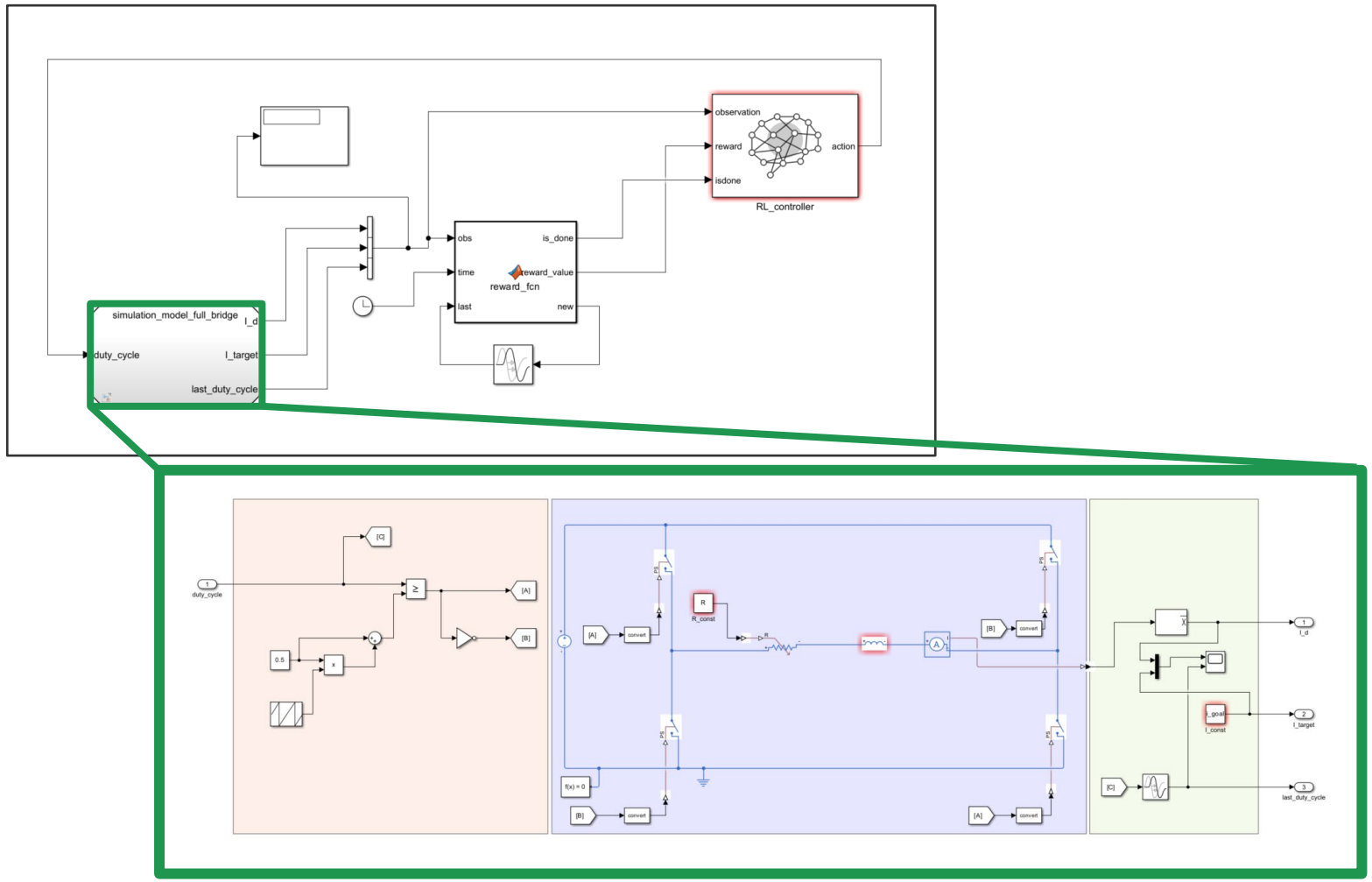


Unsupervised Learning

- Neue Anwendungsfälle kennenlernen: Mustererkennung
- KI soll unterschiedliche Verläufe erkennen



KI als Regelung



Reinforcement Learning

- Komplexere Lernmethoden kennenlernen
- Optimale Regelung in allen Fällen

Was lernen die Studierenden?

Studierende lernen, …

… wie KI-Lernprozesse aufgebaut werden

… welche KI-Typen gibt es

… wie erstellt man einfache NN

… wie man eine KI trainiert und worauf zu achten ist:

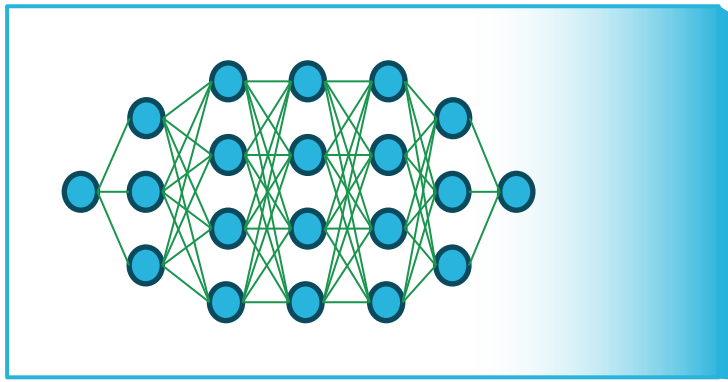
- z.B.: Normalisierung und Verteilung der Trainingsdaten

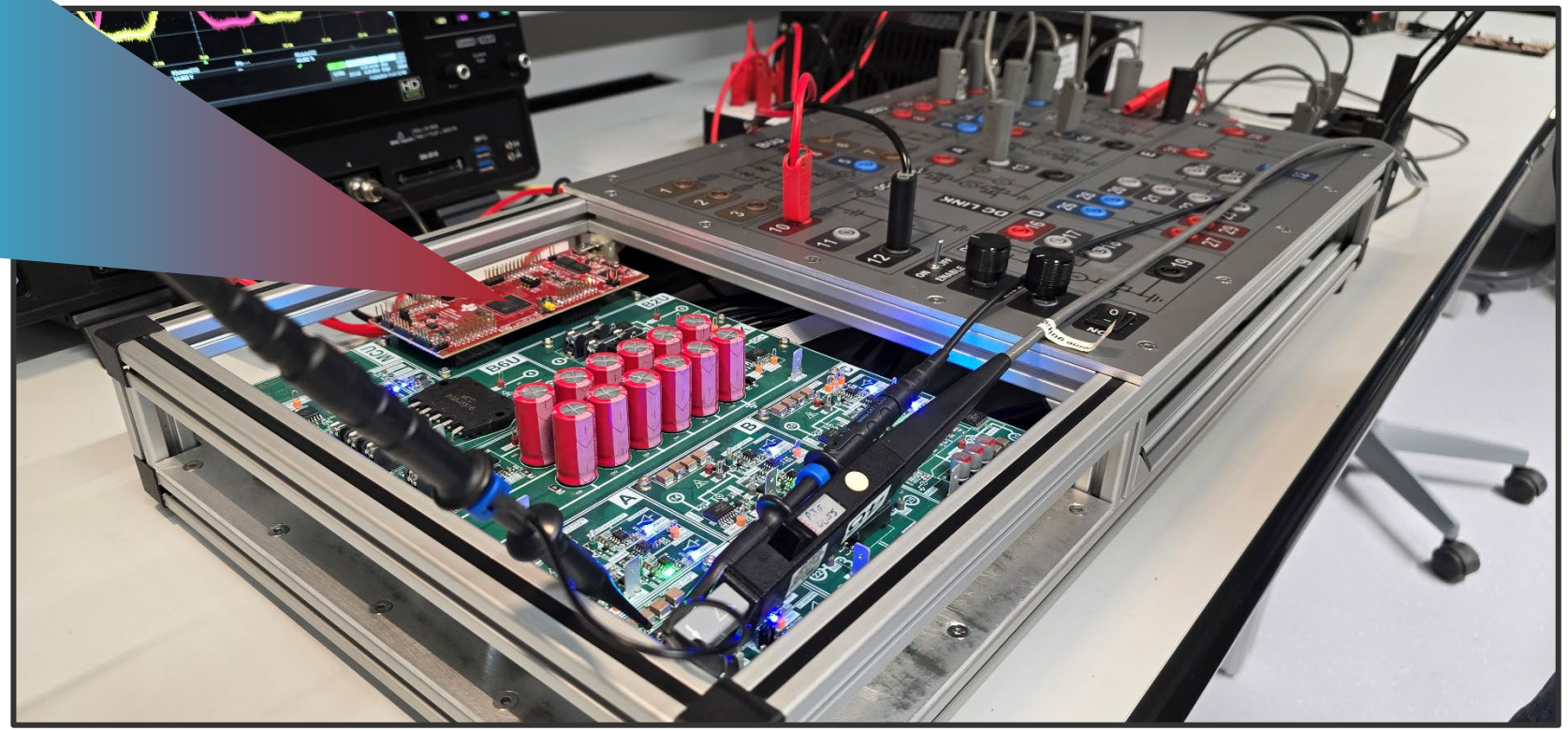
… wie sehen Verwendungsmöglichkeiten aus:

- z.B. komplexe Verhalten nachahmen, Datenanalyse automatisieren

P3E Power Electronics, Electronics and EMC
Prof. Dr.-Ing. C. Schumann

Hochschule Kaiserslautern
University of Applied Sciences



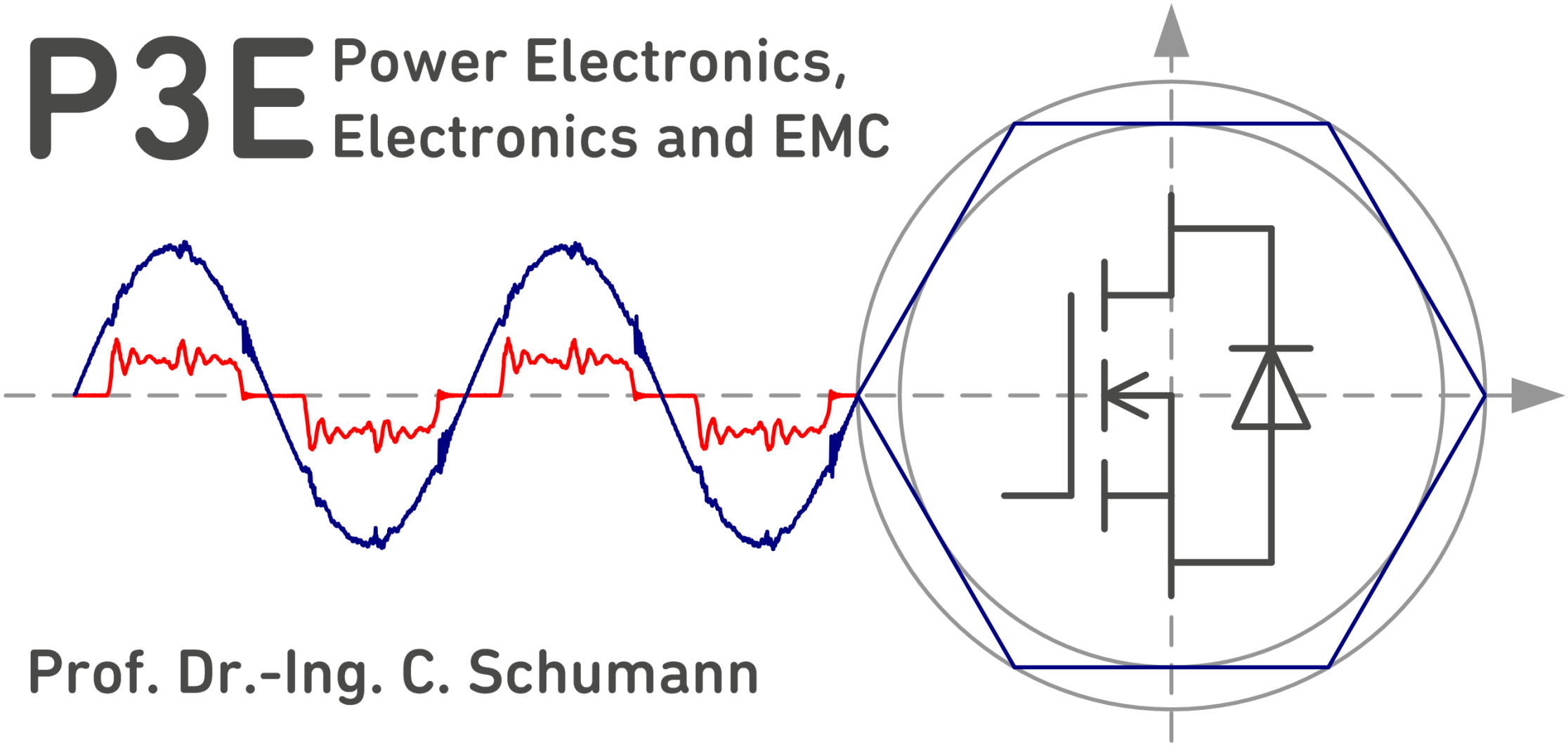


Jennifer Piela, B. Eng.
Power Electronics, Electronics and EMC (P3E)

Schoenstr. 11
67659 Kaiserslautern

+49 631 3724 2464
jennifer.piela@hs-kl.de
http://www.hs-kl.de/p3e

P3E Power Electronics, Electronics and EMC



Prof. Dr.-Ing. C. Schumann