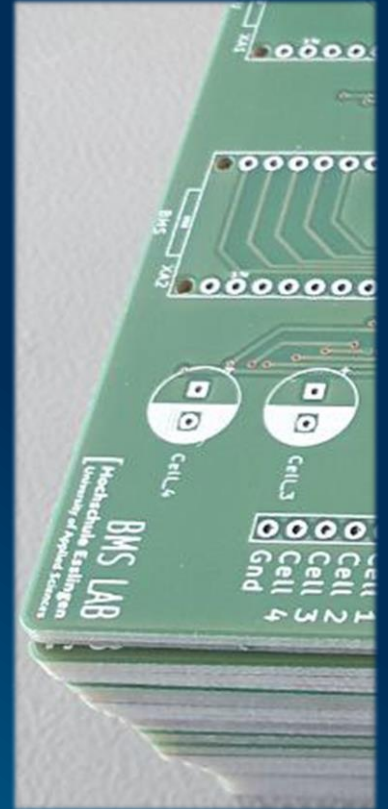


HOCHSCHULE ESSLINGEN  
**BMSLAB**

PROF. DR.-ING. KAI ANDRÉ BÖHM



# INHALT

1. Übersicht Projekt **BMSLAB**
2. Funktionale Anforderungen
3. Teamzusammensetzung
4. Rollen und Verantwortlichkeiten
5. Manuelle Testfunktionen
6. Beschreibung der Software
7. Manuelle Testfunktionen
8. Softwaregesteuerte Testfunktionen
9. Arduino Installation und Einrichtung

# SYSTEMÜBERSICHT BMSLAB



# SYSTEMÜBERSICHT BMSLAB



# ÜBERSICHT BMSLAB

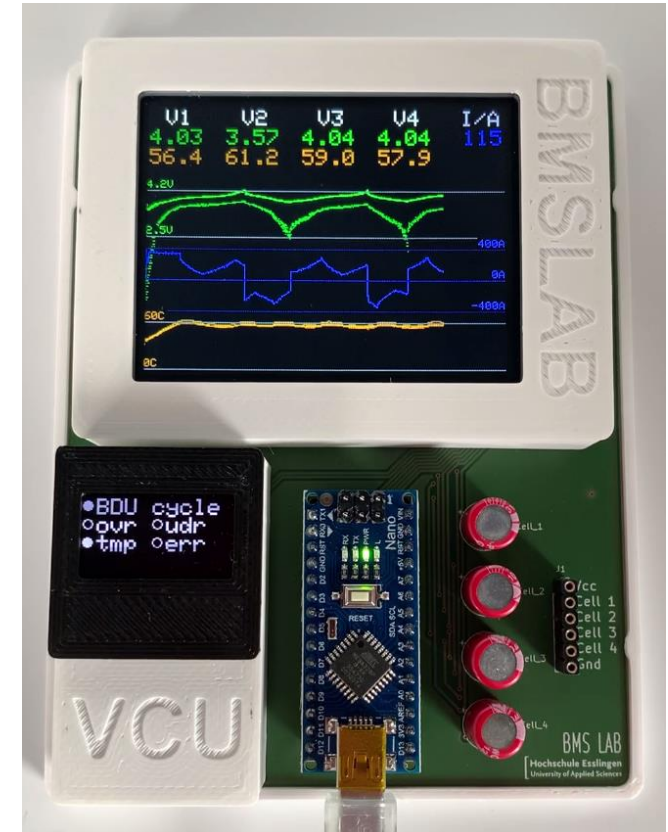
## Simulation eines Battery-Management-Systems

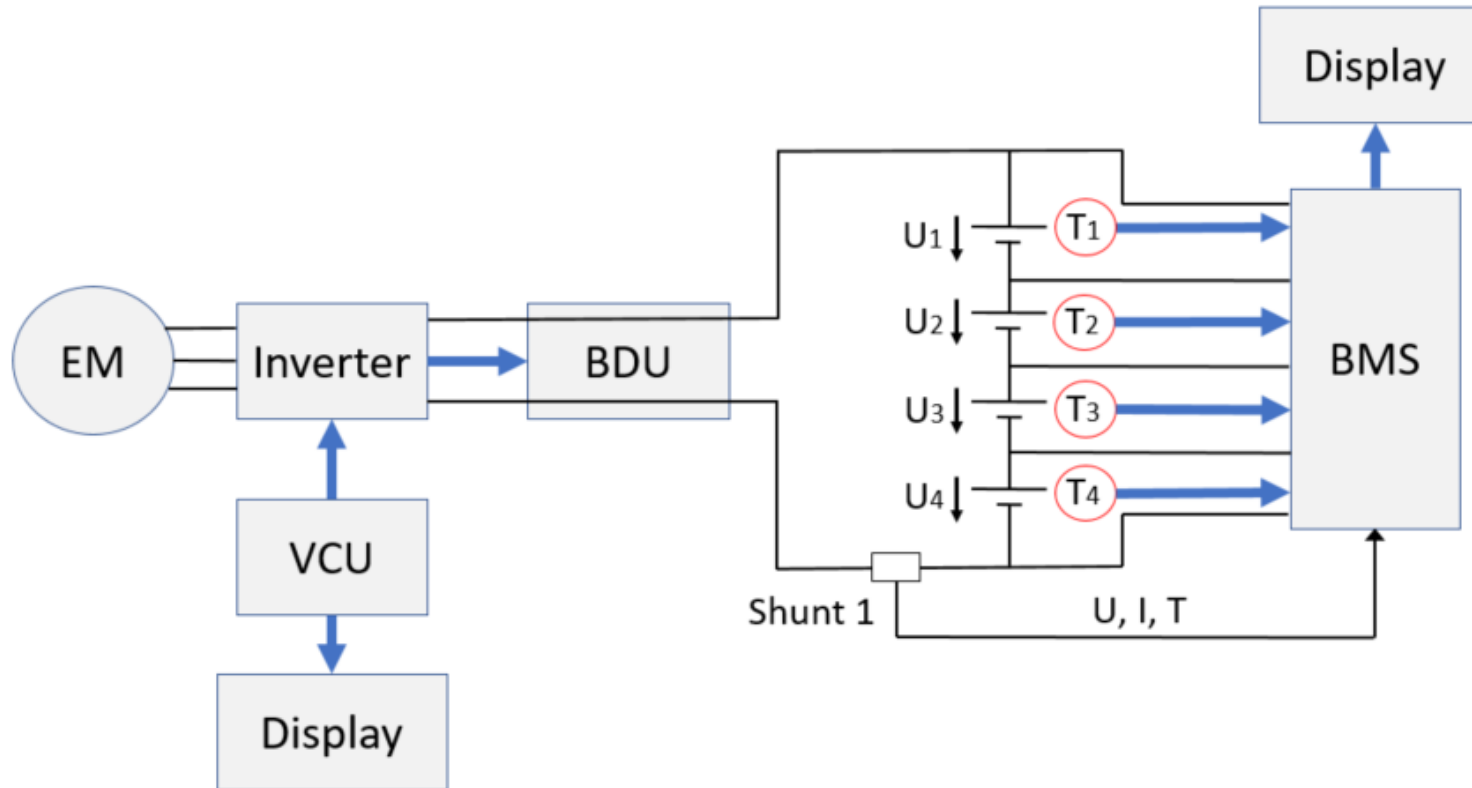
### Funktionen

- Simulation der Batterie eines EVs
- Gekapselte VCU simuliert Fahrzeug
- Frei programmierbares BMS zur Überwachung der Batterie
- Basissoftware bereitgestellt

### BMS Basisfunktionen

- Schutz der Zellen vor Überspannung, Überstrom und Übertemperatur
- Cell-Balancing
- Fehlerdiagnose
- Optional grafische Ausgabe
- Optional Batteriezustandserkennung

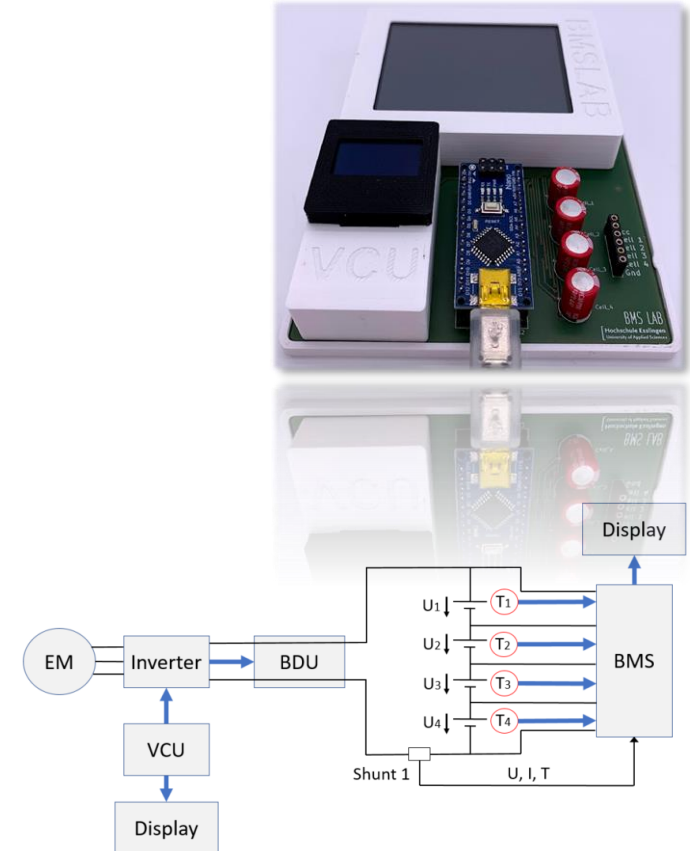




# ÜBERSICHT BMSLAB

## Systemskalierung

	Simuliertes EV	Faktor	BMS-LAB physikalisch
Batteriekapazität	100 kWh	$1,8 \cdot 10^{11}$	$5,5 \cdot 10^{-10}$ kWh
Systemspannung	420 V	100	0 .. 4,2 V
Batteriestrom	400 A	$4,4 \cdot 10^7$	9 $\mu$ A
Batterieleistung	168 kW	$4,4 \cdot 10^9$	37,8 $\mu$ W
Zellzahl	100	25	4
Zellspannung	2.5V .. 4.2V	4	0V .. 1,05V
Zellkapazität	238 Ah	$6,25 \cdot 10^9$	$3,8 \cdot 10^{-8}$ Ah
Balancing-Strom	50 mA	5000	1 $\mu$ A



## Funktionale Systemanforderungen

ID	Typ	Anforderungs - Beschreibung	Relevanz	Reifegrad
BMS-10	Überschrift	Safety Management		akzeptiert
BMS-11	Anforderung	Das BMS muss die Spannung aller Zellen überwachen und mindestens alle 200 ms messen.	obligatorisch	akzeptiert
BMS-12	Anforderung	Das BMS muss bei Überschreitung einer oder mehrerer Zellspannungen über 4.2 V ein geeignetes Signal an die VCU senden.	obligatorisch	akzeptiert
BMS-13	Anforderung	Das BMS muss bei Unterschreitung einer oder mehrerer Zellspannungen unter 2.5 V ein geeignetes Signal an die VCU senden.	obligatorisch	akzeptiert
BMS-14	Anforderung	Das BMS muss die Temperatur aller Zellen überwachen.	obligatorisch	akzeptiert
BMS-15	Anforderung	Das BMS muss bei Überschreitung einer oder mehrerer Zelltemperaturen über 60°C ein geeignetes Signal an die VCU senden.	obligatorisch	akzeptiert
BMS-16	Anforderung	Das BMS muss den Strom der Traktionsbatterie messen.	obligatorisch	akzeptiert
BMS-17	Anforderung	Das BMS bei andauernder Überschreitung eines oder mehrerer der Grenzwerte die Batterie abschalten, d.h. die Schütze öffnen (BDU).	obligatorisch	akzeptiert
BMS-18	Anforderung	Auf ein Überschreiten der genannten Grenzen muss spätestens innerhalb 50 ms reagiert werden.	obligatorisch	akzeptiert
BMS-19	Information	In diesem Projekt ist keine Betrachtung der ISO 26262 und damit keine ASIL Klassifizierung notwendig.	obligatorisch	akzeptiert



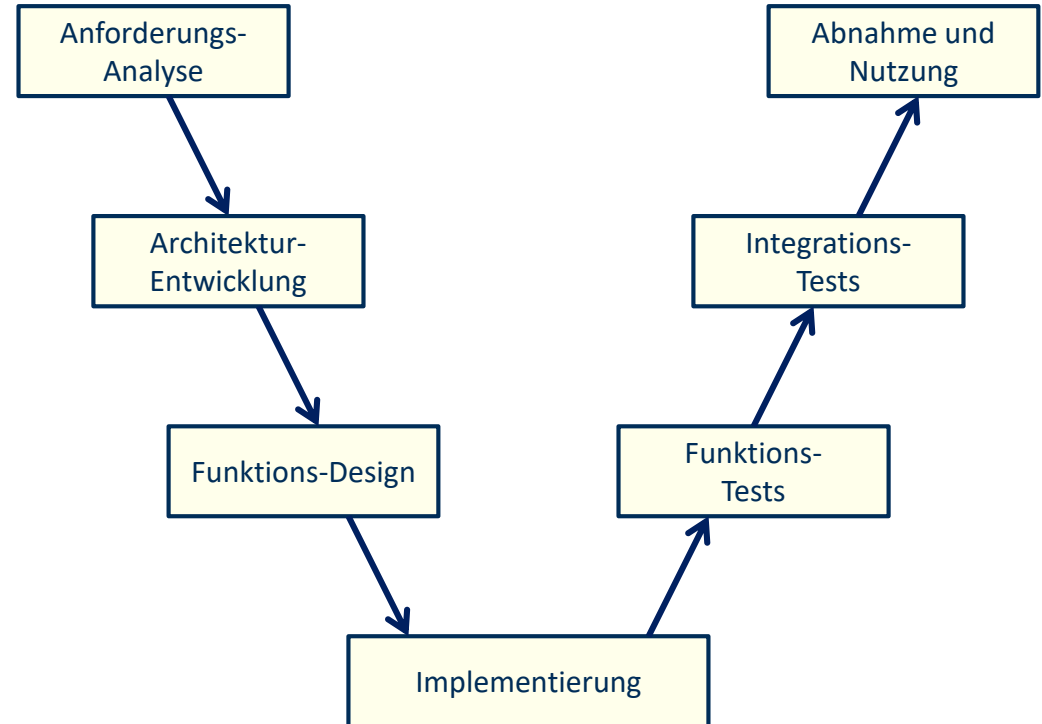
## Funktionale Systemanforderungen

ID	Typ	Anforderungs - Beschreibung	Relevanz	Reifegrad
BMS-20	Überschrift	Cell Balancing		akzeptiert
BMS-21	Anforderung	Das BMS soll einem auseinanderdriften der Zellspannungen entgegen wirken, d.h. die Zellen müssen gebalanced werden.	obligatorisch	akzeptiert
BMS-22	Anforderung	Die Anforderung des Balancings soll einmal pro Sekunde an die BSW gesendet werden.	obligatorisch	akzeptiert
BMS-30	Überschrift	Fehlerdiagnose		akzeptiert
BMS-31	Anforderung	Das BMS soll die Zellspannungen und Temperaturen auf Fehler wie Kabelbrüche und Kurzschlüsse hin untersuchen. Dazu ist ein Konzept auszuarbeiten und vorzuschlagen	erwünscht	akzeptiert
BMS-40	Überschrift	Informationsdarstellung auf Display		akzeptiert
BMS-41	Anforderung	Das BMS soll alle notwendigen Informationen auf dem Display darstellen. Dazu ist ein Konzept auszuarbeiten und vorzuschlagen	erwünscht	akzeptiert

# PROJEKT BMSLAB

## Teamzusammensetzung

1. Projektleitung / Koordination	1-3
2. Requirements-Ingenieure	2-4
3. Software-Architekten	2-4
4. Funktionsdesigner / Implementierer	2-6
5. Funktionstester	2-6
6. Integratoren	<u>2-4</u>
	11-27



## Rollen und Verantwortlichkeiten

### Projektleitung / Koordination

- Koordinierung des Projektes
- Optimierung der Teamzusammensetzung
- Definition der Randbedingungen / Arbeitsprozesse
  - Vorlagen Arbeitsdokumente
  - Datenformate und Datenaustausch

### Requirements-Ingenieure

- Erhebung und Verfeinerung der Requirements
- Change Management

### Software - Architekten

- Entwicklung der Software – Architektur
- Definition der Komponenten, deren Funktionalitäten und Schnittstellen
- Spezifikation des zeitlichen Ablaufs, von Betriebszuständen und Ausführungsdaten
- Verlinkung gegen Requirements

### Funktionsdesigner / Implementierer

- Entwicklung der Komponenten – Spezifikation
- Jeder Entwickler programmiert und spezifiziert eine oder mehrere der Komponenten – auf Basis der Architektur und unabhängig voneinander
- Verlinkung gegen Requirements

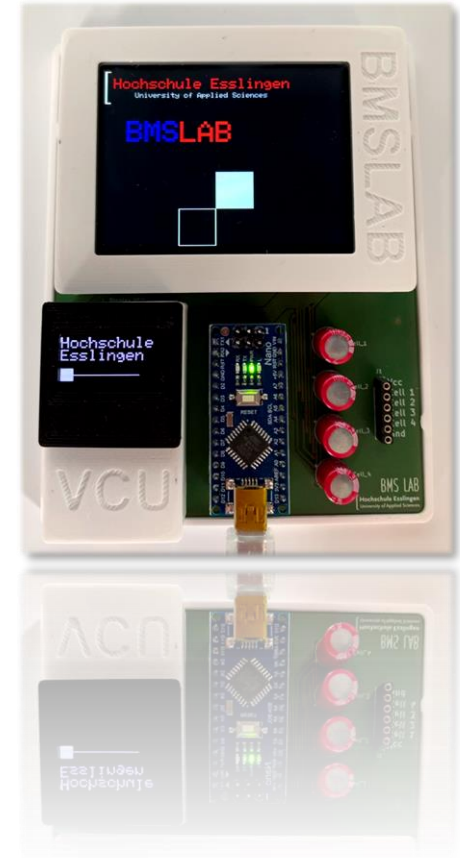
### Funktionstester

- Entwicklung der Test – Spezifikation auf Basis der Komponenten Requirements
- Nutzung von Tests und Reviews
- Dokumentation der Testergebnisse und Verlinkung gegen Requirements
- Jeder Tester testet eine oder mehrere der Komponenten – auf Basis der Spezifikation und unabhängig voneinander

### Integratoren


- Integration aller Software – Komponenten zu einer Gesamtsoftware
- Integrationstest / Reviews gegenüber der Architektur
- Test gegenüber Systemrequirements
- Arbeit im Team

Reporte bestehen aus einer kurzen Präsentation, Länge 10 - 15 Minuten.



# SOFTWARE BMSLAB

## Basisprogramm BMS



```

// Include necessary libraries
#include <SPI.h>           // SPI communication library
#include <Adafruit_ILI9341.h> // Display library
#include <TouchScreen.h>    // Display touch functionality

extern volatile uint32_t pwmDuration;

void setup()
{
    setupBSW();
}

void loop()
{
    //setBDU_Activation(true);    // schaltet BDU ein
    //setDriveMode(1);           // 1-Cycle Test 2-Slow Driver 3-Fast Driver 4-Power Mode
    receiveAndParseCommands();    // Empfängt Befehle über den Serial Monitor und führt diese aus

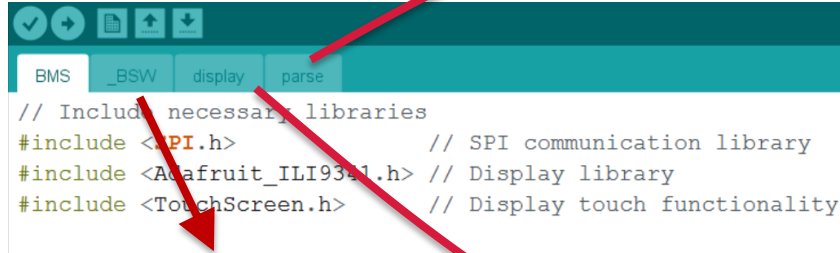
    showMeasurementValues();      // Stellt Messwerte numerisch dar
    drawMeasurementCurves(10);   // Messkurven - Parameter defines Minutes for full scale of X-Axis
    //showOCVcurve();             // Stellt OCV Kurve der Li-Ionen Zellen dar
}

```

Stellt lauffähige Umgebung  
zur Verfügung und  
Messwerte sowie zeitliche  
Verläufe auf Display dar.

# SOFTWARE BMSLAB

## Basisprogramm BMS



```
// Include necessary libraries
#include <SPI.h> // SPI communication library
#include <Adafruit_ILI9341.h> // Display library
#include <TouchScreen.h> // Display touch functionality
```

### BSW

Beinhaltet die Basis-Software-Funktionen und stellt Kommunikation mit VCU zur Verfügung.

Kann gerne eingesehen werden aber  
**HIER KEINE ÄNDERUNGEN VORNEHMEN!**

Beinhaltet die Funktion `receiveAndParseCommands()`. Sollte auch nicht geändert werden.

Beinhaltet diese drei Funktionen, kann benutzt oder modifiziert werden. Alternativ auch als Vorlage für eigene Funktionen und Darstellungen.

```
//setBDU_Activation(true); // schaltet BDU ein
//setDriveMode(1); // 1-Cycle Test 2-Slow Drive 3-Fast Drive
receiveAndParseCommands(); // Empfängt Befehle über den Serial Monitor und

showMeasurementValues(); // Stellt Messwerte numerisch dar
drawMeasurementCurves(10); // Messkurven - Parameter defines Minutes for full scale of X-Axis
//showOCVcurve(); // Stellt OCV Kurve der Li-Ionen Zellen dar
}
```

# BASISSOFTWARE BMSLAB

## Verfügbare Basissoftware-Funktionen

### Messwerterfassung und Balancing

<code>float getCellVoltage(int cell)</code>	Liefert Zellspannung von Zelle [1-4] in Volt zurück.
<code>float getCellTemp(int cell)</code>	Liefert Zelltemperatur von Zelle [1-4] in Grad Celsius zurück.
<code>float getPackCurrent()</code>	Liefert Batteriestrom in Ampere zurück. Positive Ströme sind Lade- bzw. Rekuperationsströme, negative Ströme sind Entladeströme.
<code>float getPackVoltage()</code>	Liefert Batteriegesamtspannung in Volt zurück. Dabei wird auf eine Zellanzahl von 100 Zellen extrapoliert, um eine EV – Gesamtspannung zu simulieren.
<code>void setBalancing(int cell)</code>	Aktiviert das Cell-Balancing (Entladung) für eine Zelle (1-4. Der Parameter cell = 0 deaktiviert das Balancing.

# BASISSOFTWARE BMSLAB

## Verfügbare Basissoftware-Funktionen

### Schnittstelle zur VCU

```
void setBDU_Activation(bool state)
```

Aktiviert (true) oder Deaktiviert (false) die Battery Disconnect Unit. Bei true sind die Schütze geschlossen.

```
void setDriveMode(int mode)
```

Auswahl des Drive Modes des Fahrzeuges bzw. der VCU:

- |               |  |
|---------------|--|
| 1-Slow Driver | Simuliert einen langsamen, vorsichtigen Fahrer. Erzeugt zufällige Fahrprofile.   |
| 2-Fast Driver | Simuliert einen sportlichen Fahrer ebenfalls in zufälligen Fahrprofilen  |
| 3-Power Mode  | Simuliert eine Fahrt auf einer Rennstrecke. Es werden zufällige Profile, aber stets mit maximaler Lade- bzw. Entladeleistung gefahren. |
| 4-Cycle Test  | Simuliert Labortest, in dem die Batterie alternierend voll geladen und anschließend mit konstantem Strom entladen wird.                |

# BASISSOFTWARE BMSLAB

## Verfügbare Basissoftware-Funktionen

### Schnittstelle zur VCU

`void setWarningOvervoltage(bool state)`

Sendet bei Übergabe von true eine Überspannungswarnung an die VCU. Die VCU wird versuchen, die Spannung nicht weiter zu erhöhen.

`void setWarningUndervoltage(bool state)`

Sendet bei Übergabe von true eine Unterspannungswarnung an die VCU. Die VCU wird versuchen, die Spannung nicht weiter zu erhöhen.

`void setWarningOvertemp(bool state)`

Sendet bei Übergabe von true eine Übertemperaturwarnung an die VCU. Die VCU wird die Leistung des Fahrprofils reduzieren, um die Wärmeentwicklung zu verringern.

`void setIgnoreLimits(bool state)`

Bei Übergabe von true wird die VCU veranlasst, die drei obigen Warnungen zu ignorieren.



# BASISSOFTWARE BMSLAB

## Verfügbare Basissoftware-Funktionen

### Schnittstelle zur VCU

```
void setModifySignals(bool state, bool randomize)
```

Bei Übergabe von true induziert die VCU zufällige Fehler z.B. in die Temperatur- oder Spannungs-Messung.

randomize = true: bei jedem neuen Übergang für state von false auf true (Mindestdauer ~ 20 ms) wird ein zufällige Fehler ausgewählt.

randomize = false: die Fehler werden durchgezählt.

### Modifications / Errors

Error	Beschreibung
ET01 – ET04	Temperatursensor 1-4 Kurzschluss gegen Masse.
ET11 – ET14	Temperatursensor 1-4 Kurzschluss gegen Versorgungsspannung.
ET21 – ET24	Temperatursensor 1-4 Wackelkontakt.
ET31 – ET34	Temperatursensor 1-4 Drift.
EU01 – EU03	Zellspannungsmessungen Wackelkontakt.
EI01 – EI02	Stromsensor Wackelkontakt (1 = moderat, 2 = stark).
ECDF	Kühlung defekt.

# BASISSOFTWARE BMSLAB

## Verfügbare Basissoftware-Funktionen

### Grafikfunktionen

```
void fillScreen(uint16_t color)
```

```
void drawPixel(uint16_t x, uint16_t y, uint16_t color)
```

```
void writeText(int x, int y, int tSize, String txt, uint16_t color)
```

Schreibt einen Text mit einstellbarer Größe (1-kleinste, 2,3,4,5...)

```
void drawRect(int x, int y, int w, int h, uint16_t color)
```

```
void fillRect(int x, int y, int w, int h, uint16_t color)
```

```
void drawLine(int x0, int y0, int x1, int y1, uint16_t color)
```

```
uint16_t rgb565(uint16_t r, uint16_t g, uint16_t b)
```

```
uint16_t colCell(int i)
```

```
uint16_t colTemp(int i)
```

```
String strLen(String strIn, int len)
```

Löscht den Bildschirm und füllt ihn mit der angegebenen Farbe.

Zeichnet einen Punkt an der Koordinate x/y In der übergebenen Farbe.

Schreibt einen Text mit einstellbarer Größe (1-kleinste, 2,3,4,5...)

Zeichnet ein Rechteck mit Breite w und Höhe h an die Koordinaten x/y.

Zeichnet ein ausgefülltes Rechteck.

Zeichnet eine Linie.

Wandelt einen RGB Wert (0-255 je Farbe) in das Display-Farbformat um.

Liefert eine rgb565 Farbe zur Darstellung der Zellspannung 1-4.

Liefert eine rgb565 Farbe zur Darstellung der Zelltemperatur 1-4.

Füllt einen String vorne mit Lehrzeichen, bis Die Länge len erreicht ist.

# BMSLAB

## Manuelle Testfunktionen

### Anschlüsse

- Versorgungsspannung (Vcc)
- Zellspannungen 1-4 (mit Vorwiderstand)
- Masse (Gnd)

### Manuelle Manipulationen

- Zellkurzschlüsse
- Debalancing

➔ Kurzschlüsse nicht dauerhaft bestehen lassen!

Weitere Fehler-Induzierungen über Ansteuerung der VCU per Software möglich (siehe BSW-Beschreibung).



# BASISSOFTWARE BMSLAB

## VCU Fernsteuerung zu Testzwecken

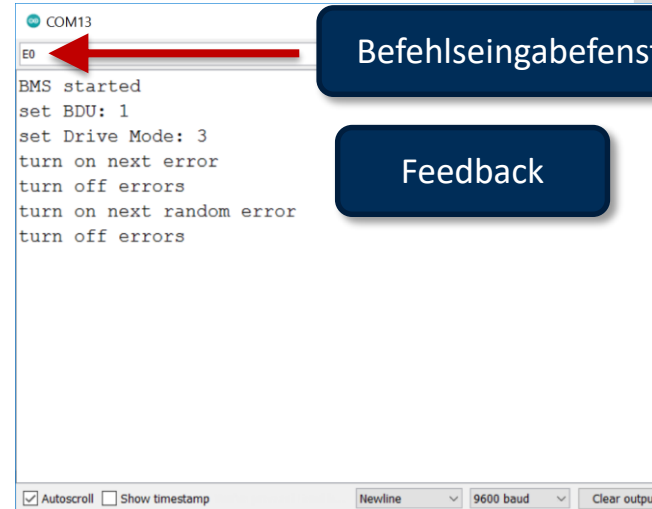
Mit der mitgelieferten BMS-Testsoftware können Sie die VCU zur Erprobung fernsteuern. Dazu öffnen Sie unter Tools den Serial Monitor.

In der oberen Zeile können Sie Steuerungsbefehle nach der unten stehenden Tabelle eingeben.

Wenn der Befehl richtig interpretiert wurde erscheint eine Bestätigung im Serial Monitor und die VCU führt den Befehl aus.

Alle Befehle bestehen aus einem großen Buchstaben gefolgt von einer ganzen Zahl, ohne Leerzeichen dazwischen.

Mit der Eingabetaste wird der Befehl abgeschickt.



## RS232 Befehle

Befehl	Beschreibung	Befehl	Beschreibung
B0	Deaktiviert die BDU.	B1	Aktiviert die BDU.
D1	Setze Drive Mode 1 bis	D4	Setze Drive Mode 4.
E0	Deaktiviere Fehler.	E1	Setze nächsten Fehler.
E2	Setze zufälligen Fehler. Bevor der Fehler mit E1 oder E2 geändert werden kann, muss er immer zuvor mit E0 ausgeschaltet werden.		

# ARDUINO INSTALLATION

## Installation der Arduino IDE



### Downloads



## Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

**SOURCE CODE**

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

### DOWNLOAD OPTIONS

**Windows** Win 7 and newer  
**Windows** ZIP file

**Windows app** Win 8.1 or 10 [Get](#)

**Linux** 32 bits  
**Linux** 64 bits  
**Linux** ARM 32 bits  
**Linux** ARM 64 bits

**Mac OS X** 10.10 or newer

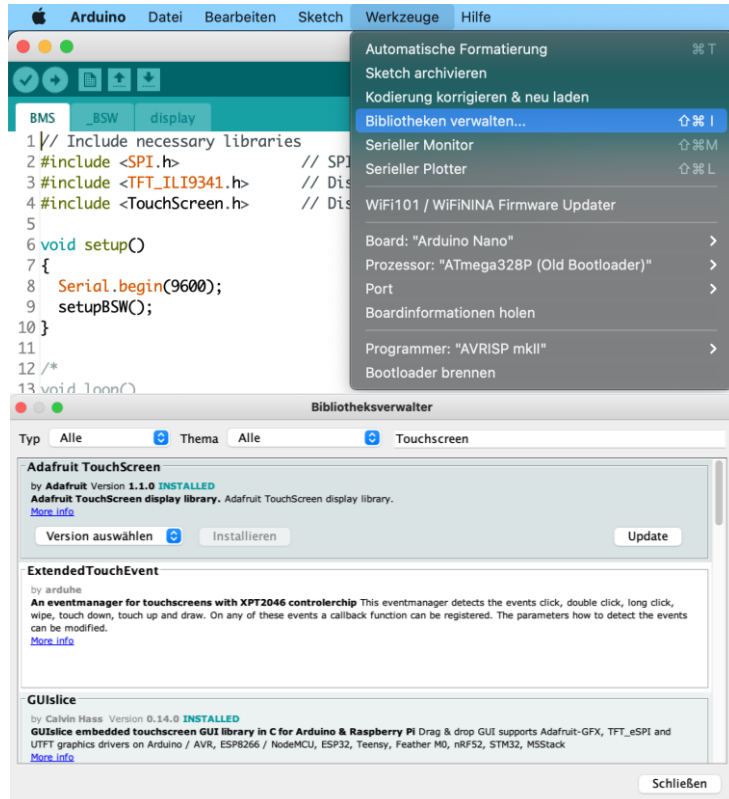
[Release Notes](#) [Checksums \(sha512\)](#)

[? Help](#)

Laden Sie sich die neueste Arduino DIE Version für Ihr Betriebssystem unter [arduino.cc](https://arduino.cc) im Bereich Software herunter und installieren Sie diese.

# BMSLAB BIBLIOTHEKEN

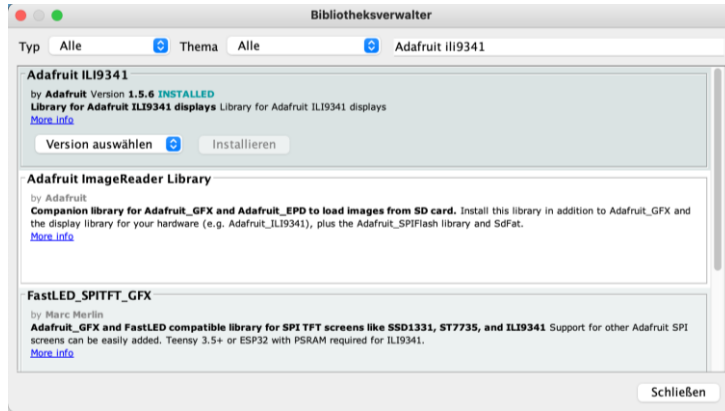
## Einfügen der benötigten Bibliotheken



- Die Bibliothek SPI.h ist bereits in der Arduino IDE enthalten.
- Die Bibliothek für das Display und den Touchscreen können Sie sich über Werkzeuge → Bibliotheken verwalten... herunterladen.
- Suchen Sie wie links gezeigt nach „Adafruit TouchScreen“ und installieren Sie sich die Bibliothek.

# BMSLAB BIBLIOTHEKEN

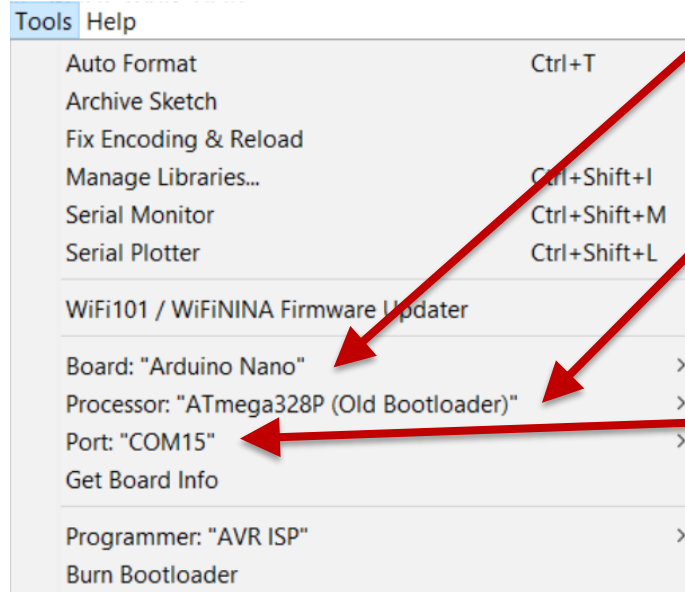
## Einfügen der benötigten Bibliotheken



- Suchen Sie wie links gezeigt nach „Adafruit iLi9341“ und installieren Sie sich die Bibliothek.
- Zusätzlich benötigen Sie „Adafruit GFX“

# BMSLAB ARDUINO EINRICHTUNG

## Einfügen der benötigten Bibliotheken



- Unter „Board“ muss der Arduino Nano eingestellt werden
- Wählen Sie unter „Processor“ den ATmega328P (Old Bootloader) => Upload Geschwindigkeit von 56 kBit/s
- Wählen Sie den richtigen COM Port für den angeschlossenen Arduino aus.  
**Tipp:** wenn Sie sich unsicher sind, stöpseln Sie den Arduino ab. Schauen Sie welche COM Ports verfügbar sind und schließen den Arduino an. Der neu dazugekommene COM Port ist der Richtige.