Bedienungsanleitung für das

IKHDS – Power Shield–A Rev. 1.1

zur Verwendung mit einem *Arduino* Mikrocontrollersystem *UNO* oder *MEGA*

Alle Angaben in dieser Bedienungsanleitung beziehen sich auf das *IKHDS – Power Shield Arduino* mit dem Ausgabenstand **Rev. 1.1.** Der Ausgabenstand des Shields ist auf der Unterseite angegeben.

Inhalt

1 Sicherheitshinweise

Vorsicht:

An das *IKHDS – Power Shield Arduino* darf unter keinen Umständen ~230 V – Netzspannung angeschlossen werden! **Es besteht Lebensgefahr !!**

- Diese Bedienungsanleitung ist Bestandteil des Produktes. Sie enthält wichtige Hinweise zur Inbetriebnahme und Bedienung! Achten Sie hierauf, auch wenn Sie das Produkt an Dritte weitergeben! Bewahren Sie deshalb diese Bedienungsanleitung zum Nachlesen auf!
- Das *IKHDS – Power Shield Arduino* darf nicht fallen gelassen oder starkem mechanischem Druck ausgesetzt werden, da es durch die Auswirkungen beschädigt werden kann.
- Benutzen Sie das *IKHDS – Power Shield Arduino* nicht weiter, wenn es beschädigt ist.
- Das *IKHDS – Power Shield Arduino* muss vor Feuchtigkeit, Spritzwasser und Hitzeeinwirkung geschützt werden.
- Betreiben Sie das *IKHDS – Power Shield Arduino* nicht in der Umgebung von brennbaren Gasen, Dämpfen oder Staub.
- In Schulen, Ausbildungseinrichtungen, Hobby- und Selbsthilfewerkstätten ist das Betreiben durch geschultes Personal verantwortlich zu überwachen.

- In gewerblichen Einrichtungen sind die Unfallverhütungsvorschriften des Verbandes der gewerblichen Berufsgenossenschaften für elektrische Anlagen und Betriebsmittel zu beachten.
- Dieses Gerät ist nicht dafür bestimmt, durch Kinder oder Personen mit eingeschränkten physischen, sensorischen oder geistigen Fähigkeiten oder mangels Erfahrung und / oder mangels Wissen benutzt zu werden.
- Das Produkt ist kein Spielzeug! Halten Sie das *IKHDS – Power Shield Arduino* von Kindern fern.
- Entfernen Sie keine Aufkleber vom Produkt. Diese können wichtige sicherheitsrelevante Hinweise enthalten.

2 Bestimmungsgemäße Verwendung

Das *IKHDS – Power Shield Arduino* wurde entwickelt zur Adaption von in der Industrie weit verbreiteten Geräten mit einer Betriebsspannung von U_B = +24 V an ein Arduino Board des Typs UNO oder MEGA, das mit einer Betriebsspannung von Vcc = +5 V betrieben wird.

Vorsicht:

Alle Ein- und Ausgangs-Pins der Buchsenleisten IOL, IOH, AD und PW des *Arduino* –Boards (siehe Abbildung 1 und Abbildung 4) sind durch das *IKHDS – Power Shield Arduino* belegt und können im Sketch des Anwenders nicht mehr anderweitig verwendet werden!

Der Zugriff auf die Ein- und Ausgänge des *IKHDS – Power Shield Arduino* muss ausschließlich über die Functions der Library *PowerShieldArduino* erfolgen, die Sie mittels `#include` –Anweisung in Ihren Sketch einbinden müssen (siehe Abschnitt 6).

Vorsicht:

Das *IKHDS – Power Shield Arduino* ist nur zur Verwendung mit Arduino Boards des Typs UNO oder MEGA bestimmt. Diese *Arduino* Boards müssen mit einer Betriebsspannung von Vcc = +5 V arbeiten. Andernfalls droht die Gefahr einer irreversiblen Beschädigung oder der Zerstörung des *Arduino* Boards.

Vorsicht:

Bitte beachten Sie, dass Bedien- und / oder Anschlussfehler außerhalb unseres Einflussbereiches liegen.

Für die korrekte Verbindung mit dem *Arduino* – Board und dessen Programmierung, sowie für den ordnungsgemäßen Betrieb ist alleine der Anwender verantwortlich!

Für alle aus falschem Anschluss, falscher Ansteuerung, falscher Programmierung und / oder falschem Betrieb resultierende Schäden trägt der Benutzer die alleinige Verantwortung!

Verständlicherweise können wir für Schäden, die daraus entstehen, keinerlei Haftung übernehmen.

Eine andere Verwendung als angegeben ist nicht zulässig! Änderungen können zur Beschädigung dieses Produktes führen, darüber hinaus ist dies mit Gefahren, wie z.B. Kurzschluss, Brand, elektrischem Schlag etc. verbunden. Für alle Personen- und Sachschäden, die aus nicht bestimmungsgemäßer Verwendung entstehen, ist nicht der Hersteller, sondern der Betreiber verantwortlich.

3 Einführung

Das *IKHDS – Power Shield Arduino* ermöglicht die Kommunikation eines *Arduino* - Boards mit der industriellen Welt, in der mit einer Betriebsspannung von U_B = +24 V gearbeitet wird.

- Die Klemmleiste des *IKHDS – Power Shield Arduino* hat sechs digitale Eingänge DI1 … DI6 (siehe Abbildung 1). Eine an einen dieser Eingänge angelegte Spannung im Bereich 0 … +24 V wird in ein Logiksignal im Bereich 0 … +5V umgesetzt, das der Mikrocontroller auf dem Arduino Board verarbeiten kann.

- Das *IKHDS – Power Shield Arduino* setzt sechs Ausgangssignale des Mikrocontrollers auf dem Arduino Board in den Bereich 0 … +24 V um. Diese bilden die sechs digitalen Ausgangssignale DO1 … DO6 auf der Klemmleiste des Shields.
- Das *IKHDS – Power Shield Arduino* verfügt zur Messung analoger Gleichspannungen im Bereich 0 … +10 V über zwei Eingänge AI1 und AI2 auf seiner Klemmleiste (siehe Abbildung 1). Auf dem *Power Shield* werden diese Spannungen in den Bereich 0 … +5 V umgesetzt, den der ADC (Analog / Digital – Converter) des Mikrocontrollers auf dem Arduino Board verarbeiten kann. Die Analog / Digital – Wandlung selbst wird nicht durch das Shield, sondern durch den Mikrocontroller des Arduinos durchgeführt (siehe Abschnitt 9).
- Das *IKHDS – Power Shield Arduino* kann an seinem Ausgang AO1 auf der Klemmleiste des Shields (siehe Abbildung 1) eine analoge Gleichspannung im Bereich 0 … +10 V ausgeben. Das *IKHDS – Power Shield Arduino* erzeugt diese Gleichspannung durch Tiefpassfilterung eines PWM-Signals, das der Mikrocontroller auf dem Arduino Board erzeugt (siehe Abschnitt 10). Das PWM-Signal muss eine genügend hohe Frequenz ($f_{PWM} \geq 10$ kHz) aufweisen, um eine gute Glättung mit einer geringen Restwelligkeit der Ausgangsgleichspannung durch das Tiefpassfilter sicherzustellen.

Abbildung 1 zeigt das *IKHDS – Power Shield Arduino* von oben auf die Bauteileseite gesehen.

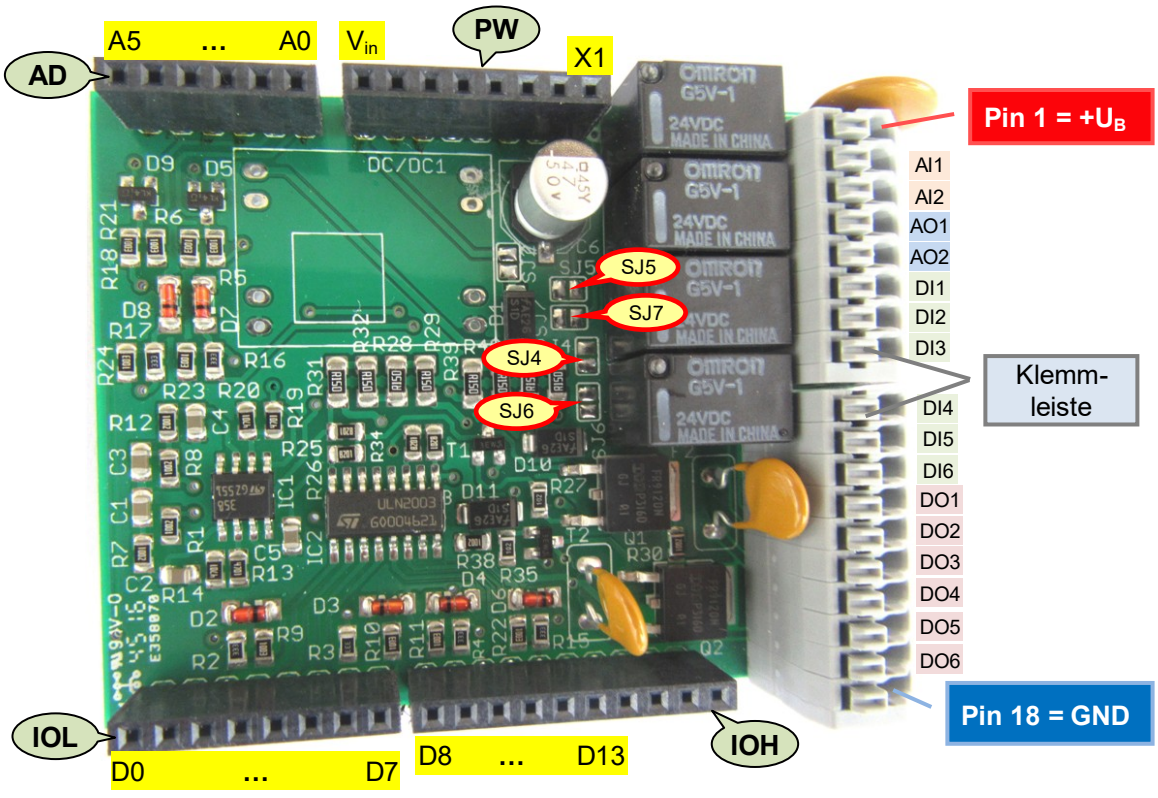

Abbildung 1: Blick von oben auf die Bestückungsseite des *IKHDS – Power Shield Arduino, Rev. 1.1*

4 Verwendung und Eigenschaften des *IKHDS – Power Shield Arduino*

- Das *IKHDS – Power Shield Arduino* ist geeignet zur Verwendung mit den weitverbreiteten Mikrocontroller – Boards *Arduino UNO* (siehe Abbildung 4) oder *Arduino MEGA* (siehe Abbildung 2), die mit einer Betriebsspannung von Vcc = +5 V betrieben werden.

- Das *IKHDS – Power Shield Arduino* wird über Abstands-Buchsenleisten auf die Buchsenleisten IOL, IOH, AD und PW des *Arduino* – Boards aufgesteckt, wie in Abbildung 2 gezeigt. Die Ein- / Ausgangspins der

Buchsenleisten IOL, IOH, AD und PW können daher von Anwender in seinem Sketch nicht mehr anderweitig genutzt werden.

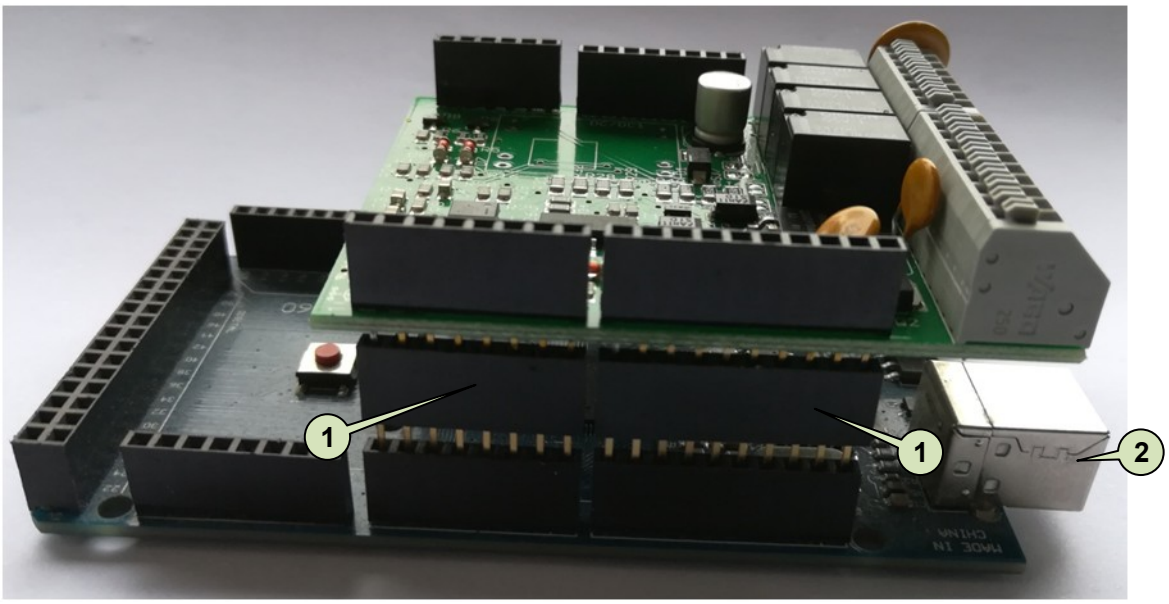

Abbildung 2: Das *Power Shield Arduino* wird über Abstands-Buchsenleisten (①) auf den *Arduino MEGA* oder *Arduino UNO* aufgesteckt

Vorsicht:

Das *IKHDS – Power Shield Arduino* **muss** über **Abstands-Buchsenleisten** (① in Abbildung 2 und Abbildung 3) auf das *Arduino* – Board aufgesteckt werden, da andernfalls das Metallgehäuse der USB-A – Buchse (② in Abbildung 2 und Abbildung 3) des *Arduino UNO* oder *Arduino MEGA* zu einem Kurzschluss der Klemmleisten-Anschlusspins auf der Unterseite des *Power Shields* (③ in Abbildung 3) führen würde!

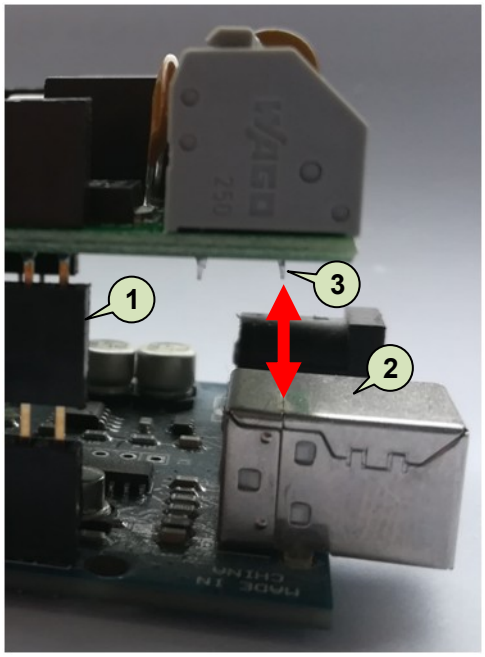

Abbildung 3: Gefahr eines Kurzschlusses zwischen den Pins der Klemmleiste und dem Metallgehäuse der USB-A – Buchse des Arduinos

- Das *IKHDS – Power Shield Arduino* muss an Anschlusspin 1 der Klemmleiste mit einer externen Betriebsspannung U_B von +24 V versorgt werden. Die Masse der Betriebsspannung (GND, 0 V) wird an Pin 18 der Klemmleiste angeschlossen (Abbildung 1).
- Die Zuführung der Betriebsspannung des *Arduino* – Boards kann wahlweise über seine USB – Buchse (rechts unten in Abbildung 4) oder durch das Einspeisen einer externen Gleichspannung U_{EXT} = +9 **…** +12 V, über die Buchse Bu1 (rechts oben in Abbildung 4) erfolgen.

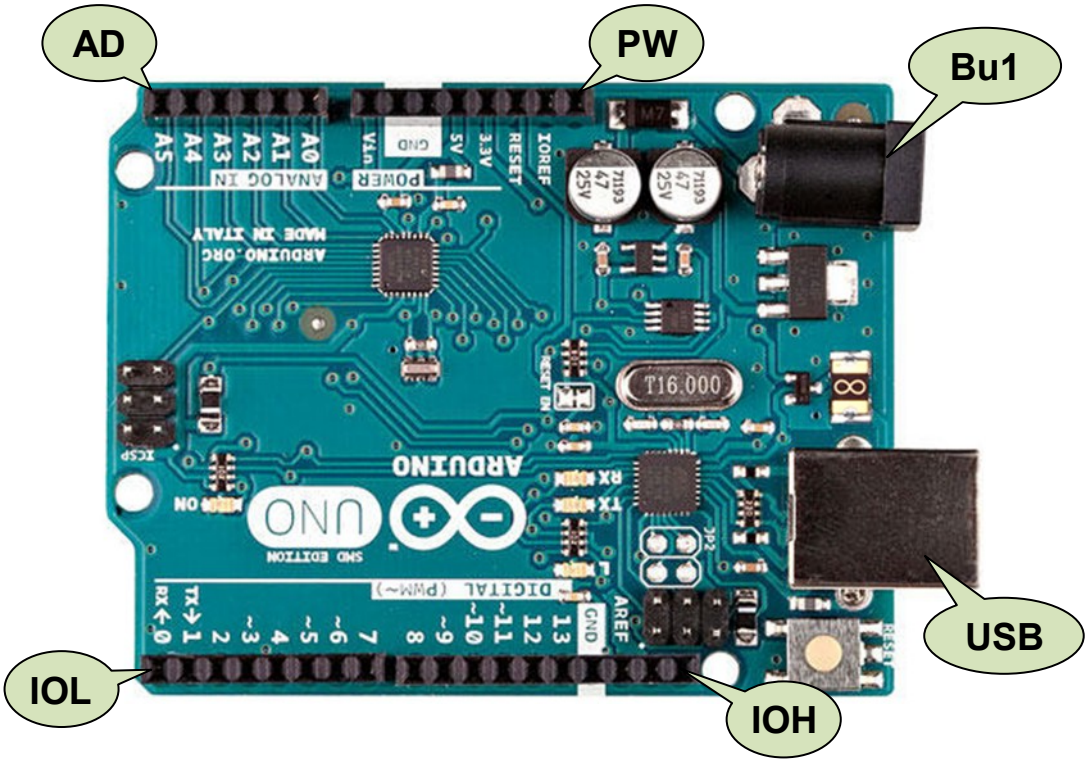

Abbildung 4: Arduino UNO mit Mikrocontroller ATmega328 (Quelle: http://www.arduino.org)

- Das *IKHDS – Power Shield Arduino* hat **...**

  6 digitale Eingänge DI1 … DI6, Eingangsspannungsbereich 0 … U_B = +24 V (Beschreibung siehe Abschnitt 7 auf Seite 9).

  6 digitale Ausgänge DO1 … DO6, davon DO1 … DO4 mit Relais- und DO5 … DO6 mit MOSFET-Ausgang (siehe Abbildung 6 und Abbildung 7). DO5 und DO6 können wahlweise als „normale“ digitale Ausgänge oder als PWM-Ausgänge betrieben werden (Beschreibung siehe Abschnitt 8 auf Seite 10).

  2 analoge Eingänge AI1 und AI2, Eingangsspannungsbereich 0 … 10 V, siehe Abschnitt 9 auf Seite 12).

  1 analogen Ausgang AO1, Ausgangsspannungsbereich 0 … 10 V, siehe Abschnitt 10 auf Seite 12).

5 Technische Daten des *IKHDS – Power Shield Arduino*

Nominale Betriebsspannung U_B (empfohlen)	U_B = +24 V
Zulässiger Bereich der Betriebs-spannung U_B	U_B = +22 … +25 V
Zulässige HIGH-Eingangsspannung U_{DIH} eines digitalen Eingangs	+4,5 V < U_{DIH} ≤ +U_B
Zulässige LOW-Eingangsspannung U_{DIL} eines digitalen Eingangs	0 ≤ U_{DIL} < +2,0 V
Eingangsstrom I_{DI} eines digitalen Eingangs @ U_{DIH} = +24 V	I_{DI} < 1 mA
HIGH-Ausgangsspannung U_{DOH} eines digitalen Ausgangs	DO1 … DO4: U_{DOH} = U_B DO5, DO6: U_{DOH} = (U_B – 0,8 V) … U_B [1)]
Maximaler Ausgangsstrom I_{DOmax} der Relais-Ausgänge DO1 … DO4	I_{DOmax} = 1 A (siehe hierzu nachfolgenden Hinweis ⚠)
Maximaler Ausgangsstrom I_{DOmax} der MOSFET-Ausgänge DO5 und DO6	I_{DOmax} = 250 mA (siehe hierzu nachfolgenden Hinweis ⚠)
Zulässige Eingangs-Gleichspannung U_{AI} eines analogen Eingangs	0 ≤ U_{AI} < 2· U_{AREF} [2)]
Eingangswiderstand R_{AI} eines analogen Eingangs	R_{AI} ≈ 200 kΩ
Ausgangsspannung U_{AO} eines analogen Ausgangs	U_{AO} = 0 … +10 V
Maximaler Ausgangsstrom I_{AOmax} eines analogen Ausgangs	I_{AOmax} = 10 mA

Tabelle 1: Charakteristische Kenngrößen und Eigenschaften des *IKHDS – Power Shield Arduino*

Achtung:
Die **Summe** der Ausgangsströme **aller** Ausgangs-Pins des *IKHDS – Power Shields* darf 2 A nicht überschreiten !

6 Die Library zur Steuerung des *IKHDS – Power Shield Arduino*

Zur Steuerung des *IKHDS – Power Shield Arduino* steht die Library *PowerShieldArduino* zur Verfügung, die Sie als *ZIP*-Datei *PowerShieldArduino.zip* aus dem Download-Bereich
http://www.kaftan-media.com/epages/63190602.sf/de_DE/?ObjectPath=/Shops/63190602/Categories/Downloads der der KAFTAN media UG auf Ihren PC herunterladen können.

Die *ZIP*-Datei enthält den Ordner *PowerShieldArduino*, in dem sich die Dateien *PowerShieldArduino.h, PowerShieldArduino.cpp* und *keywords.txt* sowie der Ordner *example* mit einem Anwendungsbeispiel befindet.

[1)] Die Ausgangsspannung der digitalen Ausgänge DO5 und DO6 ist durch den Spannungsabfall an Sensorwiderstand R1 (Abbildung 7) in geringem Maße abhängig von der Belastung des Ausgangspins

[2)] U_{AREF} ist die Referenzspannung des Analog / Digitalwandlers des Arduinos

6.1 Installation der Library für das *IKHDS – Power Shield Arduino*

Laden Sie die Datei *PowerShieldArduino.zip* von der oben angegebenen Website auf Ihren PC herunter. Zur Installation der Library klicken Sie in der Arduino IDE auf *Sketch* und danach auf *Include Library.* Wählen Sie in der sich öffnenden Dropdown-Liste die Option »*Add .ZIP Library*…«, wie in Abbildung 5 gezeigt.

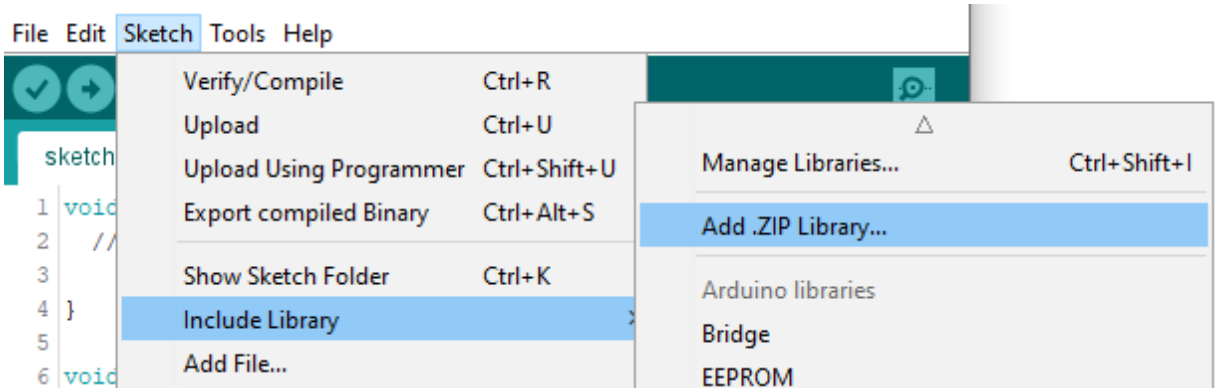

Abbildung 5: Installation der "gezippten" Library

Anschließend werden Sie aufgefordert, die Library auszuwählen, die Sie installieren möchten. Navigieren Sie zu dem Verzeichnis, in das Sie die *ZIP*-Datei heruntergeladen haben (in der Regel das Verzeichnis *Downloads* auf Ihrem PC), wählen Sie die Datei *PowerShieldArduino.zip* durch anklicken aus und klicken Sie danach den Button »Open«. Die *ZIP*-Datei wird entpackt und in den *Library*-Ordner in Ihrem *Arduino-Sketchbook* übertragen.

Die Library kann jetzt mittels Anweisung

```
#include <PowerShieldArduino.h>
```

in Ihren Arduino-Sketch eingebunden werden.

Weitere Hinweise zur Installation vom Libraries in der Arduino-IDE finden Sie z.B. unter https://www.arduino.cc/en/Guide/Libraries.

6.2 Die Functions der Library *PowerShieldArduino*

Die Library *PowerShieldArduino* ist in der Sprache C++ geschrieben und definiert die Klasse »*powerShieldArduino*«. Um mit der Library zu arbeiten, erzeugen Sie zu Beginn Ihres Sketches eine Instanz der Klasse *powerShieldArduino* mit einem von Ihnen gewählten Namen. Als Beispiel erzeugt die Anweisung

```
powerShieldArduino  powerShield;
```

die Objekt-Variable »powerShield«. "Schreibfaule" können auch kürzere Namen wählen (z.B. »ps«). Anschließend können die Functions der Klasse im Sketch verwendet werden.

Die Bezeichner `DI1 … DI6`, `DO1 … DO6`, `AI1` und `AI2` sowie `A01` für die Anschlüsse der Klemmleiste des Shields (Abbildung 1) sind in dem Header-File *PowerShieldArduino.h* der Library definiert und müssen vom Anwender in seinem Arduino Sketch für die Zugriffe auf die Ein- / Ausgänge des Shields verwendet werden.

Der Zugriff auf das *Power Shield Arduino* erfolgt mithilfe der folgenden Methoden der Library:

Function	Argumente	Beschreibung
`begin()`	–	**Muss** in der SETUP()-Function des Sketches zur Initialisierung der PWM-Timer aufgerufen werden.

getArduinoType()	–		Liefert den Typ UNO, MEGA oder UNSUPPORTED des angeschlossenen Arduino-Boards zurück (Datentyp *arduinoType_t*)
digRead(*digitaler Eingang***)**	*digitaler Eingang*:	DI1 … DI6	Liest den logischen Zustand an dem betreffenden Eingang in eine *bool*-Variable ein (siehe Abschnitt 7)
digWrite(*digitaler Ausgang, Level***)**	*digitaler Ausgang*: *Level*:	DO1 … DO6 LOW oder HIGH bzw. 0 oder 1	Setzt den betreffenden Ausgang auf logisch 0 (LOW) bzw. logisch 1 (HIGH = +24 V), siehe Abschnitt 8
digToggle(*digitaler Ausgang***)**	*digitaler Ausgang*:	DO1 … DO6	Invertiert den logischen Zustand an diesem Ausgang: Aus 0 wird 1 und umgekehrt (siehe Abschnitt 8)
anaReadInt(*analoger Eingang***)**	*analoger Eingang*:	AI1 … AI2	Liest den Wert (U_{IN} · 1024) / 10 V an dem betreffenden Eingang in eine 16 Bit - *int*-Variable ein (siehe Abschnitt 9)
anaReadVoltage(*analoger Eingang***)**	*analoger Eingang*:	AI1 … AI2	Liest den Wert der Gleichspannung im Bereich 0.0 … 10.0 V an dem betreffenden Eingang in eine *float*-Variable ein (siehe Abschnitt 9)
anaWriteInt(*analoger Ausgang, Wert***)**	*analoger Ausgang*: *Wert*:	AO1 0 … 1023	Gibt *Wert* (Datentyp *int*) als analoge Gleichspannung (*Wert* · 10 V) / 1024 an Ausgang AO1 aus (siehe Abschnitt 10)
anaWriteVoltage(*analoger Ausgang, Wert***)**	*analoger Ausgang*: *Wert*:	AO1 0.0 … 10.0	Gibt den *float*-*Wert* als analoge Gleichspannung im Bereich 0.0 … 10.0 V an Ausgang AO1 aus (siehe Abschnitt 10)
pwmWriteInt(*PWM-Ausgang, Wert***)**	*PWM- Ausgang*: *Wert*:	DO5 … DO6 0 … 255	Gibt ein PWM-Signal mit dem Tastgrad (*Wert* / 256) · 100 % an dem betreffenden Ausgang aus (siehe Abschnitt 8.2)
pwmWritePercent(*PWM-Ausgang, Wert***)**	*PWM- Ausgang*: *Wert*:	DO5 … DO6 0.0 … 100.0	Gibt ein PWM-Signal mit dem Tastgrad *Wert* (Datentyp *float*) in Prozent an dem betreffenden Ausgang aus (siehe Abschnitt 8.2)
setPwmFrequency(*Wert***)**	*Wert*:	FREQ31Hz, FREQ122Hz, FREQ488Hz	Die Frequenz des PWM-Signals an DO5 bzw. DO6 wird auf *Wert* (Datentyp *pwmFreq_t*) eingestellt (siehe Abschnitt 8.2)

Tabelle 2: Die Functions der Library *PowerShieldArduino* zur Steuerung der Ein- / Ausgänge des *IKHDS – Power Shield Arduino*

6.3 Anwendungsbeispiele für die Library *PowerShieldArduino*

In dem folgenden Sketch finden Sie Anwendungsbeispiele für die Functions der Library *PowerShieldArduino*.

```
#include "PowerShieldArduino.h"  // die Library wird in den Sketch eingebunden

powerShieldArduino pwrShieldA;   // Deklaration einer Instanz der Klasse powerShieldArduino

void setup() {
  // Die Function .begin() muss in setup() aufgerufen werden, um das Shield verwenden zu können!
  pwrShieldA.begin();  // das Power Schield Arduino wird initialisiert, PWM-Frequenz = 488 Hz

  arduinoType_t type = pwrShieldA.getArduinoType();  // einlesen des angeschlossenen Arduino Boards
  if(type == UNO) ...  // entsprechende Aktionen ausführen
  else if(type == MEGA) ...  // entsprechende Aktionen ausführen
  else if(type == UNSUPPORTED) ...  // entsprechende Aktionen ausführen

  bool level = pwrShieldA.digRead(DI3);  // Eingangspegel an DI3 in Variable »level« einlesen

  pwrShieldA.digWrite(DO1, HIGH);  // setzt den Ausgang DO1 auf HIGH (+24 V)
  pwrShieldA.digWrite(DO5, LOW);   // setzt den Ausgang DO5 auf LOW (0 V)

  // Bei den Ausgängen DO5 und DO6 ist die Function »digToggle« nur anwendbar, wenn sich diese
  // nicht im PWM-Modus befinden. Im PWM-Modus wird der Aufruf der Function »digToggle« ignoriert.
  pwrShieldA.digToggle(DO6);   // invertiert das Ausgangssignal an DO6 (aus 0 wird 1 und umgekehrt)

  // An AI1 liege die Spannung U_AI1 = 4.5 V. Dann wird mit der folgenden Anweisung der Wert
  // ((4.5 V · 1024) / 10 V) = 460 in die Variable »analogInt« eingelesen
  int analogInt = pwrShieldA.anaReadInt(AI1);  // Integerwert ((U_AI1 · 1024) / 10 V) einlesen

  // An AI2 liege die Spannung U_AI2 = 0.3 V. Dann wird mit der folgenden Anweisung der Wert 0.3 V
  // in die Variable »analogFloat« eingelesen
  float analogFloat = pwrShieldA.anaReadVoltage(AI2);  // Fließkommazahl 0.3 V einlesen

  pwrShieldA.anaWriteInt(AO1, 102);  // setzt die Spannung U_AO1 auf 102/1024 * 10 V = 0.996 V
  pwrShieldA.anaWriteVoltage(AO1, 2.75);  // setzt die Spannung an Ausgang U_AO1 auf 2.75 V

  pwrShieldA.pwmWriteInt(DO5, 85);  // setzt den PWM-Tastgrad an DO5 auf g = 85/256 = 33 %
  pwrShieldA.pwmWritePercent(DO6, 66.7);  // setzt den PWM-Tastgrad an DO6 auf 66,7 %
  pwrShieldA.setPwmFrequency(FREQ122Hz);  // setzt die PWM-Frequenz auf 122 Hz
}

void loop() { }  // hier nicht verwendet
```

7 Die digitalen Eingänge DI1 … DI6

Die Belegung der Klemmleiste des *IKHDS – Power Shield Arduino* geht aus Abbildung 1 hervor. Die digitalen Eingänge DI1 … DI6 sind die Pins 6 … 11 der Klemmleiste.

Damit der Eingangspegel an DI1 … DI6 sicher als HIGH erkannt werden kann, muss die HIGH-Eingangsspannung U_{DIH} eines digitalen Eingangs im Bereich $+4{,}5\ V < U_{DIH} \leq +U_B$ liegen.

Damit der Eingangspegel an DI1 … DI6 sicher als LOW erkannt werden kann, muss die LOW-Eingangsspannung U_{DIL} eines digitalen Eingangs im Bereich $0 \leq U_{DIL} < +2{,}0\ V$ liegen.

Um einen Digitalwert einzulesen, führt man im Arduino Sketch die Function »digRead« aus der *Power Shield* - Library aus (siehe Abschnitt 6), der man als Argument den Namen DI1 … DI6 des gewünschten Digitaleingangs übergibt:

```
pwrShieldA.digRead(DI4);
```

8 Die digitalen Ausgänge DO1 … DO6

Die digitalen Ausgänge DO1 … DO6 liegen an den Pins 12 … 17 der Klemmleiste (Abbildung 1).

8.1 Die Relais-Ausgänge DO1 … DO4

Bei den digitalen Ausgängen DO1 … DO4 schaltet im EIN-Zustand (logisches HIGH) ein Relaiskontakt K die Betriebsspannung U_B = +24 V an den zugehörigen Pin DO1 … DO4 der Klemmleiste (Abbildung 6). Im AUS-Zustand ist der Kontakt offen, wenn die beiden Pads des jeweiligen Löt-Jumpers SJ4… SJ7 nicht durch eine Lotbrücke verbunden sind. Die Lage der Löt-Jumper SJ4… SJ7 kann Abbildung 1 entnommen werden. Soll auch im AUS-Zustand ein definiertes Potenzial am Ausgang anliegen, kann durch Kurzschließen des jeweiligen Löt-Jumpers SJ4… SJ7 durch eine Lötzinnbrücke im AUS-Zustand Massepotenzial an den zugehörigen Pin der Klemmleiste gelegt werden.

Im Auslieferungszustand sind die Löt-Jumper SJ4… SJ7 des *IKHDS – Power Shield Arduino* offen.

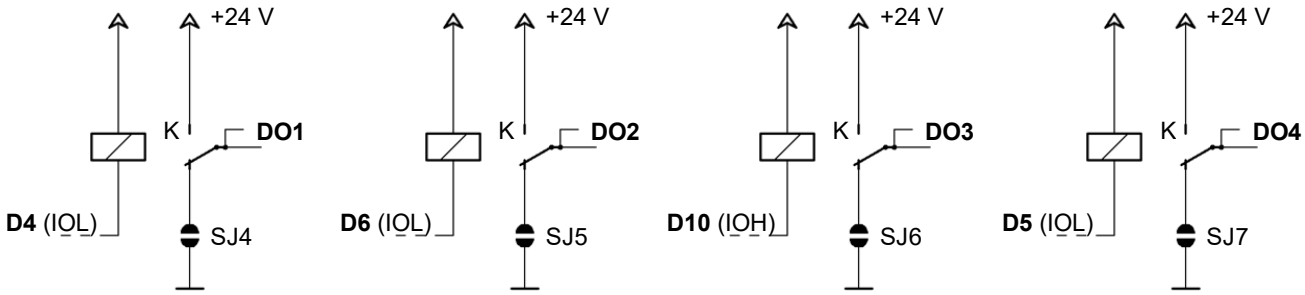

Abbildung 6: Die vier Relais-Ausgangsstufen DO1 … DO4 des *IKHDS – Power Shield Arduino*

Löt-Jumper	Relais-Ausgang
SJ4	DO1
SJ5	D02
SJ6	DO3
SJ7	DO4

Tabelle 3: Zuordnung der Löt-Jumper SJ4 … SJ7 zu den Relais-Ausgängen DO1 … DO4

Steuerung der Relais-Ausgänge DO1 … DO4

Um einen Digitalwert an DO1 … DO4 auszugeben, führt man im Arduino Sketch die Function »digWrite« aus der *Power Shield* - Library (siehe Abschnitt 6) aus, der man den Namen DO1 … DO4 des gewünschten Digital-ausgangs und den gewünschten Logikpegel LOW bzw. HIGH als Argument des Datentyps *bool* übergibt:

```
pwrShieldA.digWrite(DO1, HIGH);
```

Wird in obigem Beispiel ein HIGH-Pegel ausgegeben, zieht das Relais an und über den Relaiskontakt K liegt an Ausgang DO1 die Betriebsspannung U_B = +24 V. Wird LOW-Pegel ausgegeben, fällt das Relais ab und der Ausgang DO1 ist offen, sofern nicht der Löt-Jumper SJ4 durch eine Lotbrücke kurzgeschlossen ist.

8.2 Die MOSFET-Ausgänge DO5 … DO6

Bei den digitalen Ausgängen DO5 und DO6 schaltet ein MOSFET-Leistungstransistor die Betriebsspannung U_B = +24 V an den zugehörigen Pin der Klemmleiste (Abbildung 7). Bei ausgeschaltetem MOSFET liegen die Pins DO5 bzw. DO6 der Klemmleiste über den 10 kΩ - Widerstand R2 auf Massepotenzial.

Der Anwender des Shields kann DO5 und DO6 in seinem *Arduino Sketch* wahlweise als „normale“ logische Ausgänge, die entweder auf LOW- oder auf High-Pegel liegen oder als PWM-Ausgänge verwenden. Im PWM-Modus eignen sie sich z.B. zum Dimmen von Lampen oder zur Steuerung der Drehzahl von Elektromotoren.

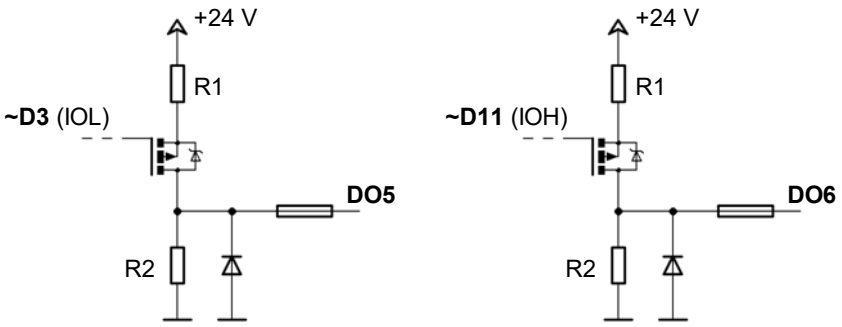

Abbildung 7: Die beiden MOSFET-Ausgangsstufen DO5 und DO6 des *IKHDS – Power Shield Arduino*

Verwendung von DO5 bzw. DO6 als digitalen Ausgang

Um einen Digitalwert an DO5 bzw. DO6 auszugeben, führt man im Arduino Sketch die Function »digWrite« aus der *Power Shield* - Library (siehe Abschnitt 6) aus, der man den Namen DO5 oder DO6 des gewünschten Digitalausgangs und den gewünschten Logikpegel LOW bzw. HIGH als Argument des Datentyps *bool* übergibt:

```
bool level = LOW;
pwrShieldA.digWrite(DO6, level);
```

Verwendung von DO5 bzw. DO6 als PWM-Ausgang

Um ein PWM-Signal an DO5 bzw. DO6 auszugeben, führt man im Arduino Sketch die Function »pwmWriteInt« oder die Function »pwmWritePercent« aus der *Power Shield* - Library (siehe Abschnitt 6) aus, der man den Namen DO5 oder DO6 des gewünschten PWM-Ausgangs und den gewünschten Tastgrad als Argument übergibt:

```
float percent = 66.7;
powerShield.pwmWritePercent(DO6, percent);
```

In diesem Beispiel wird eine Rechteckspannung mit dem in der Variablen *percent* gespeicherten Tastgrad g = 66,7 % an Ausgang DO6 ausgegeben, d.h. für 66,7 % = $^2/_3$ der Periodendauer T ist der Ausgang HIGH und für 33,3 % = $^1/_3$ der Periodendauer ist er LOW.

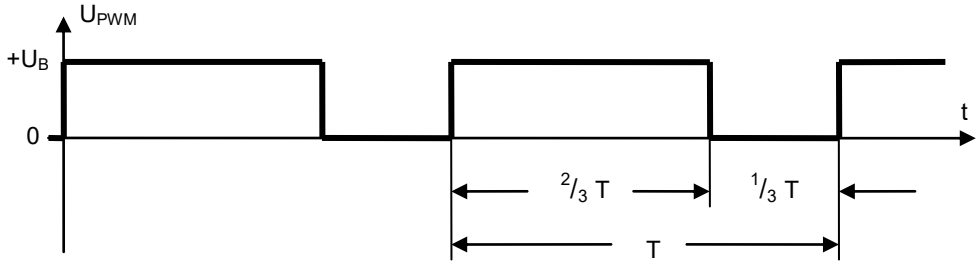

Abbildung 8: PWM-Ausgangsspannung an DO6 mit dem Tastgrad g = 66,7 % gemäß obigem Beispiel

Zur Steuerung der digitalen Ausgänge DO1 … DO6 hält die *Power Shield* - Library (siehe Abschnitt 6) noch die Function »**digToggle**« bereit. Diese Function schaltet den als Argument übergebenen Ausgang DO1 … DO6 um, d.h. hat der Ausgang LOW-Pegel, so wird er auf HIGH gesetzt, hat der Ausgang HIGH-Pegel, so wird er auf LOW gesetzt:

```
powerShield.digToggle(DO5);
```

Anmerkung:

Bei den Ausgängen DO5 und DO6 ist die Function »digToggle« nur anwendbar, wenn sich diese nicht im PWM-Modus befinden. Im PWM-Modus wird der Aufruf der Function »digToggle« ignoriert.

9 Die analogen Eingänge AI1 und AI2

Die analogen Eingänge AI1 und AI2 liegen an den Pins 2 und 3 der Klemmleiste (Abbildung 1).

Die Eingangs-Gleichspannung U_{AI} an den analogen Eingängen AI1 und AI2 muss stets kleiner sein als die doppelte Referenzspannung U_{AREF} des Analog / Digitalwandlers des Arduinos, also

$$U_{AI} < 2 \cdot U_{AREF}.$$

Zur Verwendung mit dem Power *Shield* muss der Analog / Digitalwandler des Arduinos mit der Referenzspannung U_{AREF} = +5 V betrieben werden. Dies ist auch der Standardfall, denn nach dem Einschalten des Arduinos wird automatisch seine Betriebsspannung Vcc = +5 V als Referenzspannung U_{AREF} für den A / D – Wandler eingestellt.

Um einen Spannungswert einzulesen, ruft man im Arduino Sketch die Function »anaReadInt« oder die Function »anaReadVoltage« aus der *Power Shield* - Library (siehe Abschnitt 6) auf, der man als Argument den Namen des gewünschten Analogeingangs AI1 oder AI2 übergibt. Die Function liefert die ermittelte Eingangsspannung zurück:

```
int voltage = powerShield.anaReadInt(AI2);
```

bzw.

```
float voltage = powerShield.anaReadVoltage(AI2);
```

Im obigen Beispiel wird der Gleichspannungswert U_{AI2} an Anschluss AI2 in die Variable *voltage* eingelesen.

Wie bei jedem Analog / Digital - Wandler ist auch hier der Wert der eingelesenen Spannung quantisiert, d.h. die Eingangsspannung wird in Stufen mit einer bestimmten Schrittweite eingelesen. Der kleinste auflösbare Spannungswert U_{LSB} beträgt bei einer maximalen Eingangsspannung von +10 V und einer Genauigkeit des Wandlers von *n* Bit:

$$U_{LSB} = \frac{10\,V}{2^n}.$$

Die A / D – Wandler des Arduino UNO und des Arduino MEGA weisen eine Auflösung von 10 Bit auf. Damit ergibt sich für das LSB:

$$U_{LSB} = \frac{10\,V}{2^{10}} = \frac{10\,V}{1024} \approx 9{,}766\,mV.$$

Die Eingangsspannung wird demnach als Vielfaches von 9,766 mV eingelesen.

Beispiel:

Für die Eingangsspannung U_{AI2} = 9,500 V ergäbe sich der Zahlenwert

$$\frac{9{,}500\,V}{0{,}009766\,V} = 972{,}8.$$

Da der Analog / Digital - Wandler des Arduino nur Integerwerte liefern kann, ist der ermittelte Wert abgerundet auf 972. Der Aufruf der Function `anaReadVoltage(AI2)` liefert deshalb statt U_{AI2} = 9,500 V den Wert

U_{AI2} = 972 · 0,009766 = 9,492 V zurück.

Das *IKHDS – Power Shield* hat nur einen geringen Einfluss auf die Genauigkeit der Analog / Digital –Wandlung. Deren Genauigkeit wird in wesentlichem Maß durch den Wandler des Mikrocontrollers auf dem Arduino Board bestimmt. Wird das *IKHDS – Power Shield* auf einen Arduino aufgesteckt, dessen Referenzspannung die Betriebsspannung Vcc = +5 V des Arduinos ist, können an die Messergebnisse keine zu hohen Ansprüche gestellt werden. Mit entsprechenden Toleranzen muss in diesem Fall gerechnet werden. Falls allerdings keine Präzisionsmessungen erforderlich sind, wird die erzielbare Genauigkeit für die meisten Anwendungen sicher genügen.

10 Der analoge Ausgang AO1

Der analoge Ausgang AO1 liegt an Pin 4 der Klemmleiste (Abbildung 1).

Zur Ausgabe eines analogen Gleichspannungswertes führt man im Arduino-Sketch die Function »anaWriteInt« oder die Function »anaWriteVoltage« aus der *Power Shield* - Library (siehe Abschnitt 6) aus, der man den Namen AO1 des Analogausgangs und den Wert der gewünschten Spannung übergibt:

```
powerShield.anaWriteInt(AO1, 1005);
```

bzw.

```
powerShield.anaWriteVoltage(AO1, 9.81);
```

Im Beispiel wird in beiden Fällen eine Gleichspannung mit dem Wert U_{AO1} = 9.81 V an AO1 ausgegeben.

Wie bei jedem Digital / Analog - Wandler ist auch hier die Ausgangsspannung quantisiert, d.h. die Ausgangsspannung kann nur sprungweise mit einer bestimmten Schrittweite eingestellt werden. Der kleinste darstellbare Spannungswert U_{LSB} beträgt bei einer maximalen Ausgangsspannung von +10 V und einer Auflösung des PWM-Signals von *n* Bit:

$$U_{LSB} = \frac{10\,V}{2^n}\,.$$

Die Ausgangsspannung wird somit als Vielfaches von U_{LSB} eingestellt. Der an die Function *anaWriteInt(AO1, wert)* bzw. *anaWriteVoltage(AO1, wert)* übergebene Wert wird von der Function entsprechend angepasst.

Die analoge Ausgangsspannung U_{AO1} wird gebildet durch Tiefpassfilterung eines von dem Arduino erzeugten PWM-Signals. Beim Arduino UNO hat der ausgegebene Wert eine Auflösung von 10 Bit.

Für das LSB ergibt sich somit:

$$U_{LSB-UNO} = \frac{10\,V}{2^{10}} = \frac{10\,V}{1024} \approx 9{,}77\,mV\,.$$

Beim Arduino MEGA liegt an dem entsprechenden I/O-Pin ein Ausgang des PWM-fähigen Timers T/C2 des Mikrocontrollers. Mit T/C2 ist lediglich eine Auflösung von 8 Bit möglich. In diesem Fall entspricht dem LSB ein Spannungswert von

$$U_{LSB-MEGA} = \frac{10\,V}{2^{8}} = \frac{10\,V}{256} \approx 39{,}06\,mV\,.$$

Bei Verwendung der Functions aus der *Power Shield* - Library zur Ausgabe der Analogspannung an AO1 wird der entsprechende Timer durch den Aufruf der Function `.begin()` der Library automatisch als »Fast PWM« konfiguriert. Im Fall des Arduino UNO hat das PWM-Signal eine Auflösung von 10 Bit, dem LSB entspricht ein Spannungswert von ca. 9,8 mV, die Frequenz des PWM-Signals beträgt f_{PWM} = 15625 Hz. Im Fall des Arduino MEGA hat das PWM-Signal eine Auflösung von 8 Bit, dem LSB entspricht ein Spannungswert von ca. 39,1 mV, die Frequenz des PWM-Signals beträgt f_{PWM} = 62500 Hz. Nach der Tiefpassfilterung ergibt sich bei diesen Frequenzen eine gut geglättete Ausgangs-Gleichspannung.

Das PWM-Eingangssignal des Tiefpassfilters zur Erzeugung der Analog-Gleichspannung U_{AO1} besteht aus den HIGH- und LOW-Pegeln des Ausgangssignals der Timer des Mikrocontrollers auf dem Arduino-Board. Die Höhe und Stabilität dieser Spannungspegel wirkt sich daher unmittelbar auf die Genauigkeit der ausgegebenen Spannung aus. Wesentlich beeinflusst wird die Qualität der Spannungspegel von der Stabilität der Versorgungsspannung des Arduinos. Die in der Regel nicht sehr stabile und von einer Restwelligkeit überlagerte +5 V – Spannung, die über die USB-Buchse vom angeschlossenen PC an das Arduino Board geliefert wird, wird schlechtere Ergebnisse bringen als die Ausgangsspannung des +5 V – On Board Spannungsreglers, der von einer externen Spannung an Buchse Bu1 (Abbildung 4) versorgt wird.