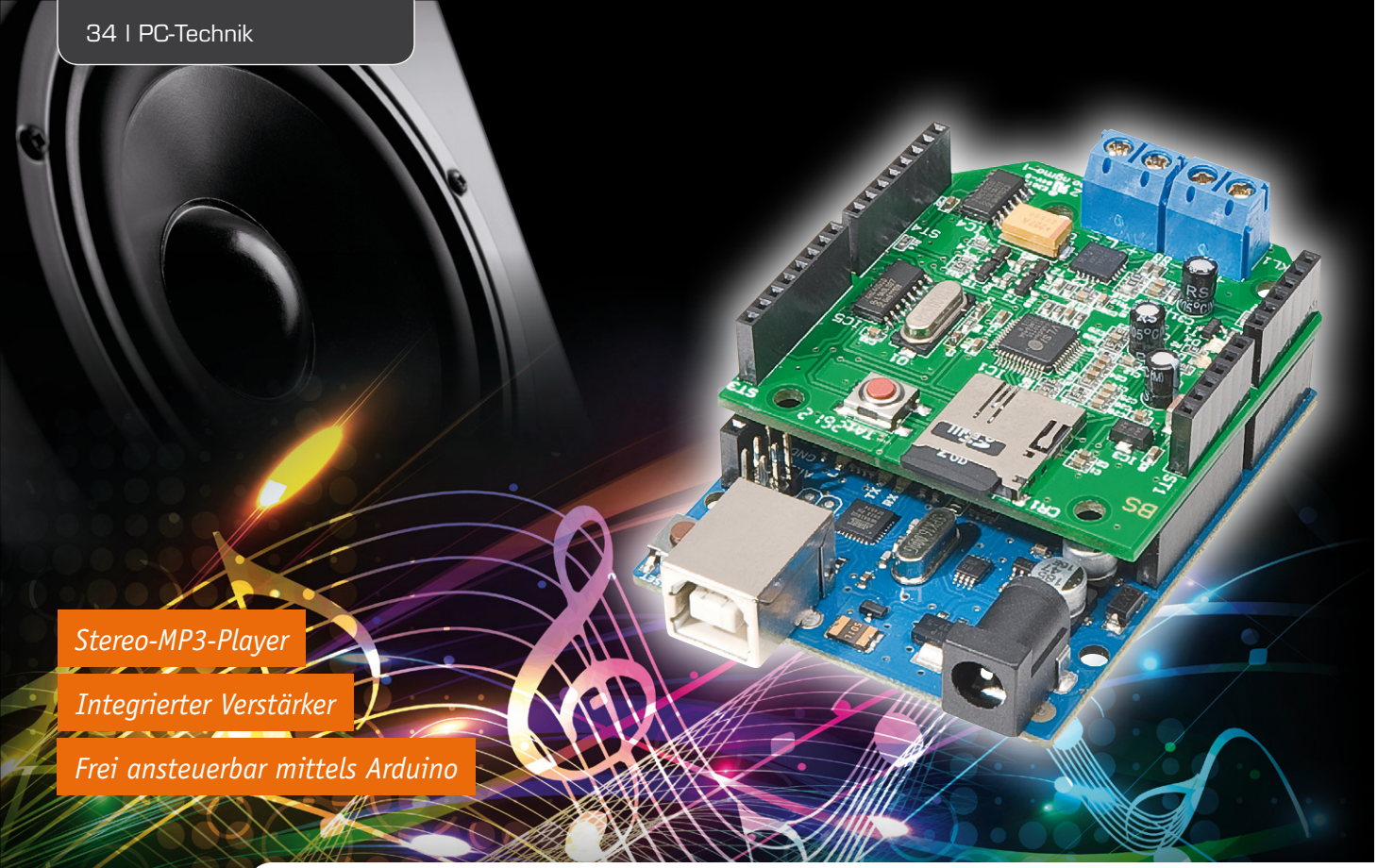


Stereo-MP3-Player

Integrierter Verstärker

Frei ansteuerbar mittels Arduino

AudioShield for Arduino

Mit dem „AudioShield for Arduino“ kann ein Arduino-Board in einen MP3-Player verwandelt werden. Dabei steht – neben der Fähigkeit, MP3-Dateien direkt auf Stereolautsprecher ausgeben zu können – zusätzlich ein microSD-Kartenschacht bereit, so dass ein geeignetes Speichermedium unmittelbar eingebunden werden kann. Die mit dem AudioShield bereitgestellte Arduino-Library enthält bereits einfache Beispiele zur Soundwiedergabe von SD-Karte.

MP3-Player selbst gebaut

Wer schon immer einen MP3-Player selbst bauen wollte, für den ist das „AudioShield for Arduino“ genau richtig, denn der Arduino bietet mit der bereits fertig aufgebauten Hardware-Umgebung und der einfach beherrschbaren Entwicklungsplattform ideale Voraussetzungen für die Implementierung individueller Funktionen und Wünsche in ein solches Gerät, das dann vielfältig einsetzbar ist.

Das im ELVjournal 6/2012 vorgestellte MP3-Soundmodul MSM3 bietet bereits die Möglichkeit, MP3-Dateien auf einer SD-Karte über eine UART- oder I²C-Schnittstelle zu steuern. Beim AudioShield wird hingegen der MP3-Decoder direkt gesteuert, was dem Anwender weitaus mehr Möglichkeiten bietet.

Daten

Kurzbezeichnung:	ASA1
Versorgungsspannung:	4,5–5,5 VDC
Stromaufnahme:	500 mA max.
Leistungsaufnahme Ruhebetrieb:	0,1 W
Ausgangsleistung pro Kanal:	max. 0,15 W (Sinus 1 kHz, 8 Ω)
Umgebungstemperatur:	+5 °C bis +35 °C
Abmessungen (B x H x T):	56 x 53 x 21 mm
Gewicht:	20 g

Neben der SD-Karte sind auch andere Quellen für das Audio-Signal denkbar, zum Beispiel I²C oder UART (dabei Übertragungsraten und Bitraten der Soundfiles beachten), oder mit zusätzlichen Erweiterungsboards (Shields) z. B. Ethernet oder Bluetooth.

Das AudioShield bietet neben einem MP3-Decoder einen Slot für eine microSD-Karte, von der Soundfiles abgespielt und zur Ausgabe auf einem Lautsprecher mittels eines integrierten Stereoverstärkers ausgegeben werden können. Die Ansteuerung erfolgt dabei über das Arduino-System, so dass der Funktionsumfang vom Anwender selbst zu bestimmen ist. Mit Hilfe der Libraries aus der Arduino-Welt lassen sich so z. B. auch ein Display und Bedienelemente einbinden oder das Abspielen lässt sich z. B. über einen Netzwerkanschluss steuern.

Die Arduino-Umgebung liefert bereits eine Library zur Ansteuerung einer SD-Karte mit. Zusammen mit der bereitgestellten Library für das AudioShield lässt sich so nach eigenen Wünschen ein MP3-Player erstellen.

Je nach Funktionsumfang sollte ein Arduino-Board mit mindestens 32 KB Flash verwendet werden (z. B. Arduino Uno [1]). Denn schon mit der einfachsten Anwendung – eine MP3-Datei von SD-Karte abzuspielen – ist ein Arduino mit einem Flash-Speicher von nur 16 KB bereits an seinen Grenzen angelangt.

Dies ist auf die Größe der SD-Bibliothek des Arduino zurückzuführen, so dass für Anwendungen mit SD-Karte ein größeres Board mit mehr Speicher genutzt werden sollte.

Bedienung

Zu der bereitgestellten Library existieren einige einfache Beispiele, welche die Verwendung des AudioShield erklären sollen. Die grundlegenden Funktionen und den Ablauf wollen wir hier kurz auflisten.

Dabei wird z. B. eine Datei von der SD-Karte über den MP3-Decoder abgespielt, aber ebenso auch ein im Flash fest hinterlegtes Audio-File. Weiterhin werden die LEDs initialisiert und angesprochen und die Überprüfung der SD-Karte und der Spannungsversorgung vorgenommen.

Der MP3-Decoder wird entsprechend über die Library VS10xx angesprochen, dabei muss diese am Anfang mit *VS1011.begin()* initialisiert werden, und über *VS1011.Reset()* wird der MP3-Decoder zurückgesetzt, dann können die Daten in 32-Byte-Paketen übertragen werden.

Um die Audio-Signale hörbar zu machen, muss entsprechend die Mute-Schaltung deaktiviert werden, damit wird auch gleichzeitig der Verstärker aktiviert.

Die Library des AudioShield stellt für die Ansteuerung des MP3-Decoders das Objekt „VS1011“ bereit, welches die folgend aufgeführten Funktionen enthält. Dabei sind den Funktionen die Objektnamen mit einem Punkt voranzustellen, beispielsweise zur Initialisierung der Library: *„VS1011.begin();“*

Die Funktionen:

Mute/Verstärker aktivieren und deaktivieren:
void UnsetMute(void);
void SetMute(void);

Initialisierung des AudioShield:
void begin(void);

Hardware-Reset des MP3-Decoders:
void Reset(void);

Software-Reset des MP3-Decoders, sollte zwischen 2 Dateien ausgeführt werden:
void SoftReset(void);

Lautstärke einstellen:
0 = maximale und 254 minimale Lautstärke,
255 deaktiviert die Ausgangsstufe des MP3-Decoders:
void SetVolume(unsigned char leftchannel, unsigned char rightchannel);

32-Byte-Datenblock an MP3-Decoder übertragen:
unsigned char Send32(unsigned char* pBuffer);*

Puffer des MP3-Decoders ausleeren:
void Send2048Zeros(void);

Register des MP3-Decoders auslesen oder schreiben, nähere Informationen dazu bitte aus dem Datenblatt des VS1011 entnehmen:
void WriteRegister(unsigned char addressbyte, unsigned char highbyte, unsigned char lowbyte);
unsigned int ReadRegister(unsigned char addressbyte);

1-kHz-Sinus-Testton ausgeben:
void SineTest(void);

Zusätzlich zu der Ansteuerung des MP3-Decoders verfügt das AudioShield, wie bereits erwähnt, über zusätzlich aktivierbare Funktionen wie die Ansteuerung zweier LEDs, die Steuerung der Spannungsversorgung der SD-Karte oder Kontrolle eines Kontakts zur Überprüfung auf das Vorhandensein einer SD-Karte im Kartenschacht. Für diese Funktionen stehen folgende Makros aus der Library zum AudioShield bereit:

Ausgänge:
LED_BLUE_ON
LED_RED_ON
LED_BLUE_OFF
LED_RED_OFF
SD_POWER_ON
SD_POWER_OFF

Eingang:
CHECK_SD_DETECT
liefert „true“ zurück, wenn eine SD-Karte vorhanden ist, oder „false“, falls keine Karte eingesetzt ist.

Während des Setups sind die entsprechenden I/O-Ports wie folgt zu initialisieren:

Einrichten der LEDs:
```
pinMode(LED_BLUE, OUTPUT);
digitalWrite(LED_BLUE, LOW); //low-aktiv
pinMode(LED_RED, OUTPUT);
digitalWrite(LED_RED, LOW); //low-aktiv
```

Einrichten SD_POWER switch:
```
pinMode(SD_OFF, OUTPUT);
digitalWrite(SD_OFF, HIGH);  //SD-Karte low-aktiv
```

SD_Detect-Eingang:
```
pinMode(SD_DETECT, INPUT); //externer Pullup vorhanden
```

Um die SD-Karte nutzen zu können, muss die SD-Library von Arduino hinzugefügt werden, dabei ist zu beachten, dass der ChipSelect der SD-Karte beim AudioShield von der SD-Library abweicht und deshalb beim Initialisieren dies als Parameter zu übergeben ist.
SD_CS als Parameter der SD-Library übergeben:
SD.begin(SD_CS);

Die SD-Library enthält keine Funktionen zur Überprüfung auf Vorhandensein einer SD-Karte oder zum Schalten der Spannungszufuhr, so dass, falls diese Funktionalität gewünscht ist, die Ansteuerung aus dem Hauptprogramm heraus erfolgen muss.

Folgende Beispiele existieren zu der Library:

Sinetest:

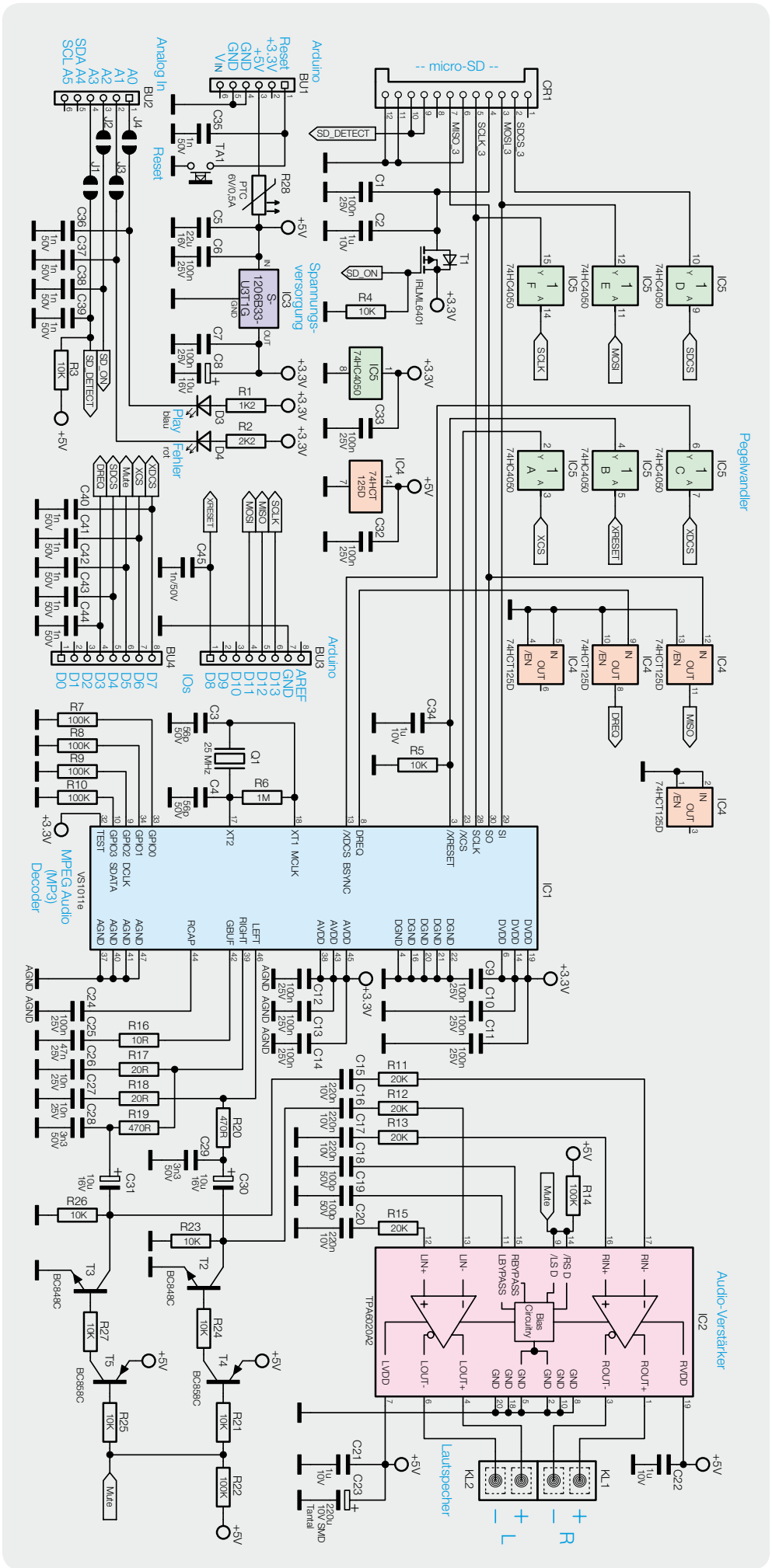
- Ausgabe eines Sinus-Testtons (1 kHz), generiert vom MP3-Decoder zum Test des MP3-Decoders und des Stereoverstärkers

Simple:

- Die Datei mit Namen „001.MP3“ wird von SD-Karte abgespielt.
- Rote LED leuchtet auf, falls die Datei nicht gefunden wird.
- Blaue LED leuchtet während des Abspielvorgangs.

Flash:

- Ein fest im Flash abgelegter Sound wird abgespielt.
- Blaue LED leuchtet während des Abspielvorgangs.



Extended:

- Aus der Variablen „Filenumber“ wird der Dateiname im Format „XXX.MP3“ generiert, wobei jedes „X“ ein Zeichen von „0“ bis „9“ sein kann.
- „Filenumber“ wird dabei hochgezählt und beginnt bei Überschreiten von 100 wieder mit 1.
- Rote LED leuchtet auf, falls Datei nicht gefunden wird.
- Blaue LED leuchtet während des Abspielvorgangs.

In den Beispielen erfolgen zur einfacheren Kontrolle und zum Verständnis der Funktion teilweise zusätzlich Rückmeldungen über die serielle Verbindung.

Schaltung

Bild 1 zeigt die Schaltung des „AudioShield for Arduino“. Der lineare Spannungsregler IC3 erzeugt aus der 5-V-Spannung des Arduino-Boards eine 3,3-V-Spannung zur Versorgung des MP3-Decoders und der SD-Karte, C5 bis C8 dienen dabei zur Spannungsstabilisierung. Der Verstärker wird direkt aus der 5-V-Spannung des Arduino versorgt. Der Taster TA1 bewirkt einen Reset des Arduino-Boards.

Bild 1: Schaltung des „AudioShield for Arduino“

Da SD-Karte und MP3-Decoder mit 3,3 V versorgt werden und entsprechend auch mit 3,3-V-Signalen arbeiten, muss eine Pegelwandlung von den 5-V-Signalen des Arduino auf 3,3-V-Signale und umgekehrt stattfinden. Dies erledigen IC5 und IC4, dabei handelt es sich bei IC5 um einen Levelshifter, den 74HC4050, und bei IC4 um einen Tristate-Buffer vom Typ 75HCT125, welche jeweils die Pegelanpassung in eine Richtung (5 V -> 3,3 V; 3,3 V -> 5 V) durchführen (siehe „Elektronikwissen“).

Die Spannungsversorgung der SD-Karte kann vom Arduino gesteuert werden, dazu ist J2 zu schließen und der I/O-Port A2 des Arduino mit High-Pegel anzusteuern, damit wird die Spannungsversorgung zur SD-Karte unterbrochen.

Am I/O-Port A3 schließt ein Kontakt gegen Masse, wenn eine SD-Karte im Kartenschacht sitzt. Dieser Kontakt besitzt einen Pull-up-Widerstand (R3) und kann so am Arduino direkt als Eingang abgefragt werden.

Auch für diese Funktion muss erst ein Lötjumper, in diesem Fall J1, geschlossen werden.

An den I/O-Ports A0 und A1 können eine rote und eine blaue LED angesprochen werden, sofern die Lötjumper J3 für Rot bzw. J4 für Blau geschlossen sind. Somit können Statusrückmeldungen – z. B. rote LED bei Fehlern, blaue LED beim Abspielen etc. – ausgegeben werden.

Über die Widerstände R11 bis R13 + R15 wird die Verstärkung des Verstärkers eingestellt. Dabei wird die Verstärkung aus den internen Widerständen (40 kΩ) des IC2 und den externen Widerständen (R11 bis R13 + R15) bestimmt:

$$Ri/Re = Gain \Rightarrow 40\ k\Omega/20\ k\Omega = 2$$

Die Mute-Beschaltung aus T2 bis T5 und den Widerständen R21, R22, R24, R25 und R27 dient zur Geräusch-Unterdrückung beim Einschalten des MP3-Decoders und gleichzeitig zum Deaktivieren des Verstärkers.

Der Oszillator Q1 stellt einen Takt von 25 MHz für den MP3-Decoder bereit.

Hinweis:
Das Arduino-Board ist bei Verwendung des AudioShields immer mit externer Stromversorgung zu betreiben, da über USB dem Arduino maximal 200 mA zur Verfügung stehen, diese aber zum Betrieb des AudioShield nicht ausreichen und es zu einer Überlastung des USB-Ports kommen könnte.

Nachbau

Da bei dem AudioShield vorwiegend SMD-Komponenten verwendet wurden, gestaltet sich der Aufbau recht einfach, denn die Baugruppe wird bereits weitgehend bestückt geliefert (Bild 2).

Es sind lediglich die Klemmen zum Anschluss der Lautsprecher, die Buchsenleisten zum Verbinden mit dem Arduino und die Elektrolyt-Kondensatoren C8, C30, C31 auf der Platinenoberseite zu bestücken. Dies erfolgt unter Zuhilfenahme von Bestückungsplan, Platinenfoto (Bild 3), Stückliste und Bestückungsdruck.

Bei den Elkos ist auf die Polarität zu achten (Minuspol am Elko markiert, auf der Platine Plus-Markierung).

Bei den Buchsenleisten und den Klemmen ist darauf zu achten, dass diese plan auf der Platine aufliegen (Bild 4), um die Kräfte beim Aufstecken weiterer Shields auf die Platine und nicht auf die Lötstellen zu über-

Bild 2: Die Platine des ASA1 wird bereits mit SMD-Bauteilen bestückt geliefert.

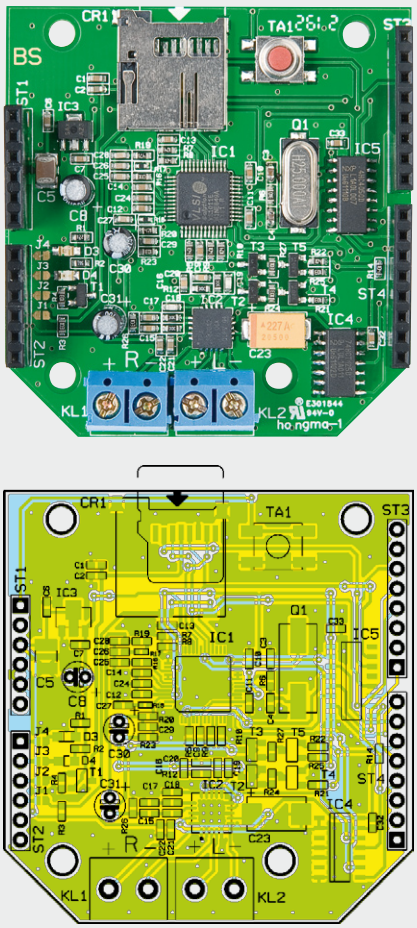


Bild 3: Ansicht der bestückten Platine des ASA1 mit zugehörigem Bestückungsplan

Achtung:

Störaussendung
Beim Einsatz des Shields mit Arduino ist im Hinblick auf Störaussendung noch etwas zu beachten. Da die kommerziell erhältlichen Arduino-Boards beim Anschluss von externen Komponenten dazu neigen, Störsignale über die Zuleitungen auszusenden, muss ein Ferritring in die Zuleitung der Versorgungsspannung eingebracht werden. Hierzu wird die Zuleitung viermal durch den Ferritring geführt.

Pegelwandler für SPI

Häufiges Problem in der Schaltungsentwicklung sind die unterschiedlichen Versorgungsspannungen digitaler Schaltkreise.

Beispielsweise wird ein Mikrocontroller mit einer Betriebsspannung von 5 V versorgt, aber über SPI sollen Komponenten angeschlossen werden, welche mit einer Betriebsspannung von 3,3 V arbeiten.

Beim SPI handelt es sich um eine unidirektionale Verbindung. In einer Richtung ist die Wandlung der 5-V-Ausgangssignale des Mikrocontrollers auf 3,3-V-Signale für die Eingänge der Komponenten (MOSI, SCK und CS) umzusetzen. Die zweite Richtung ist die Datenrichtung von den Komponenten zum Mikrocontroller (MISO), dort werden Signale mit maximal 3,3-V-Pegel zu einem mit 5 V versorgten Mikrocontroller gesendet.

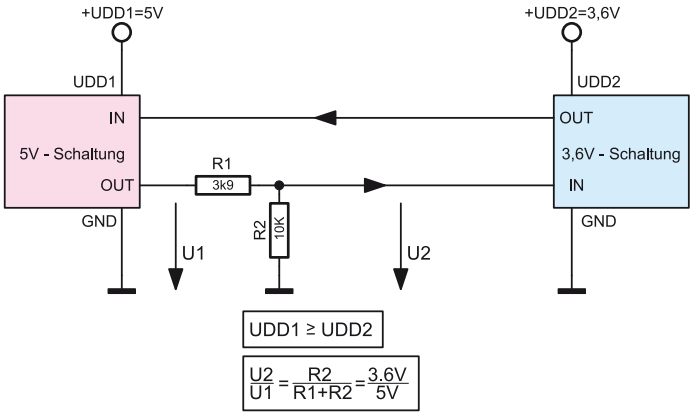
Lösung 1: Spannungsteiler

Oft werden für die Wandlung von 5-V-Ausgangssignalen zu 3,3-V-Eingängen Spannungsteiler aufgebaut. Damit ist jedoch häufig kein zu 100 % sicherer Betrieb möglich, denn es sollten möglichst hochohmige Widerstände verwendet werden, um die Ströme gering zu halten, und je nach Eingangskapazitäten der Komponenten kann es zu Verschleifungen bei schnellerer Signalübertragung kommen.

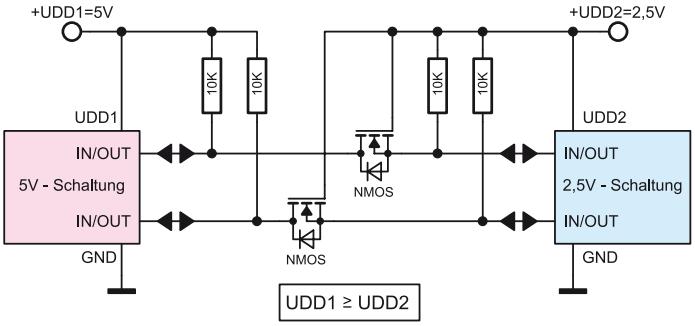
In die andere Richtung, also von der 3,3-V-Komponente zum 5-V-Controller, werden die Signale direkt verbunden.

Für einen ATmega wie auf dem Arduino-Board ist die minimale Spannung am Eingang, welche sicher als High-Pegel erkannt wird, mit VIH = 0,6 x VCC definiert.

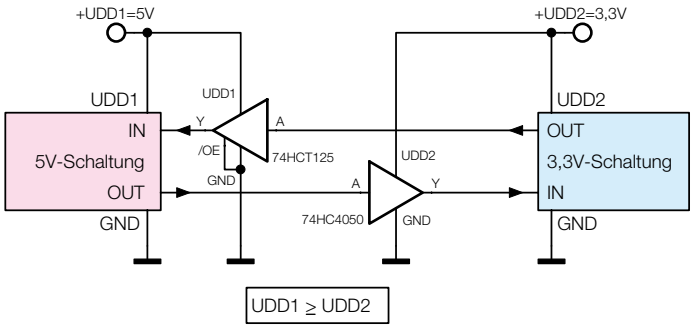
Bei einer Versorgungsspannung von 5 V beträgt VIH also 3 V, für den VS1011 ist ein High-Pegel am Ausgang schon ab 2,3 V definiert, was also unter Umständen nicht ausreicht, um vom Mikrocontroller richtig erkannt zu werden.



Beispiel 1: Pegelwandlung mit Spannungsteiler



Beispiel 2: Pegelwandlung mit MOSFETs



Beispiel 3: Pegelwandlung mit Logikbausteinen

Lösung 2:
Bidirektionaler Pegelwandler aus MOSFETs

Auch eine Lösung mit dem von uns bereits vorgestellten bidirektionalen Pegelwandler kann genutzt werden. Dabei sind jedoch auch die Kapazitäten der MOSFETs zu beachten, diese sollten möglichst gering sein, da es sonst auch dort bei höheren Übertragungsgeschwindigkeiten zu Problemen kommt.

Lösung 3: Logikgatter

Eine sichere Methode zum Umsetzen der Pegel setzt auf die Verwendung von Logikbausteinen der 74er-Reihe. Diese weisen sehr geringe Kapazitäten auf und können somit auch bei höheren Übertragungsgeschwindigkeiten verwendet werden.

Zur Wandlung von hohen Ausgangspegeln auf geringe Pegel kann der Logikbaustein 74HC4050 der 74er-Reihe eingesetzt werden, er kann Eingangssignale bis 15 V auf einen Pegel von bis zu 2 V absenken.

Bei der Umsetzung vom niedrigen zum höheren Pegel kann ein 74HCT125-Baustein abhelfen, dieser arbeitet mit einer Betriebsspannung von 4,5 V bis 5 V und kann Signale von 2,3 V als High-Pegel erkennen sowie mit einem Pegel seiner Betriebsspannung wieder ausgeben.

Im Beispiel-Bild 3 sind jeweils nur einzelne Gatter aus den Logikbausteinen abgebildet.

Anstatt eines 74HCT125 sind auch viele andere Gatter der 74HCT-Familie einsetzbar (245, 244, 240 ...).

Es werden auch spezielle Pegelwandler für SPI angeboten, welche jedoch meist recht teuer sind, hingegen sind 74er-Logikbausteine aufgrund ihrer hohen Verbreitung eine kostengünstige und sichere Alternative.

Stückliste

Widerstände:

10 Ω/SMD/0603	R16
20 Ω/SMD/0402	R17, R18
470 Ω/SMD/0603	R19, R20
1,2 kΩ/SMD/0603	R1
2,2 kΩ/SMD/0603	R2
10 kΩ/SMD/0603	R3, R4, R21, R23–R27
20 kΩ/SMD/0603	R11–R13, R15
100 kΩ/SMD/0603	R5, R7–R10, R14, R22
1 MΩ/SMD/0603	R6
Polyswitch/6 V/0,5 A/SMD/1206	R28

Kondensatoren:

56 pF/SMD/0603	C3, C4
100 pF/SMD/0603	C18, C19
1 nF/SMD/0603	C35–C45
3,3 nF/SMD/0603	C28, C29
10 nF/SMD/0603	C26, C27
47 nF/SMD/0603	C25
100 nF/SMD/0603	C1, C6, C7, C9–C14, C24, C32, C33
220 nF/SMD/0603	C15–C17, C20
1 µF/SMD/0603	C2, C21, C22, C34
10 µF/16 V	C8, C30, C31
22 µF/SMD/1210	C5
220 µF/10 V/Tantal/SMD	C23

Halbleiter:

VS1011e	IC1
TPA6020A2/SMD	IC2
S-1206B33-U3T1G/SMD	IC3
74HCT125/SMD	IC4
74HC4050D/SMD	IC5
IRLML6401/SMD	T1
BC848C	T2, T3
BC858C	T4, T5
LED/blau/SMD/0805	D3
LED/rot/SMD/0805	D4

Sonstiges:

Quarz, 25,000 MHz, SMD	Q1
Schraubklemmleisten, 2-polig, print	KL1, KL2
Mini-Drucktaster, 1x ein, 1,1 mm Tastknopflänge	TA1
microSD-Kartenhalter TFLASH Push/Push	CR1
Buchsenleisten, 1x 6-polig, gerade, print	BU1, BU2
Buchsenleisten, 1x 8-polig, gerade, print	BU3, BU4
1 Ferrit-Ringkern, 25 x 12 mm	

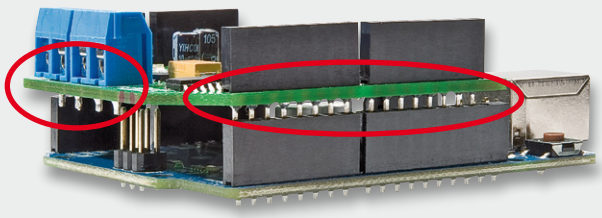


Bild 4: Hier ist, bei bereits auf dem Arduino-Board aufgesteckter Platine, gut die plane Lage der Buchsenleisten sowie der Schraubklemmen auf der Platine zu sehen.

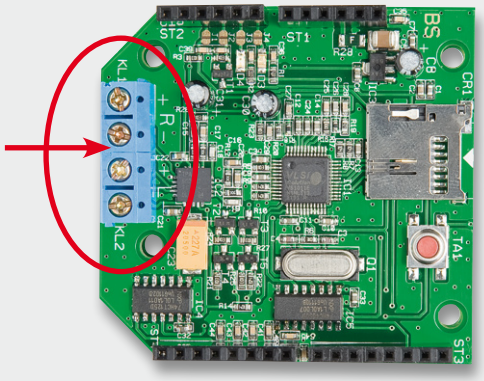


Bild 5: Die Schraubklemmen für den Lautsprecheranschluss sind zusammengesetzt zu bestücken.

Achtung:

Aufgrund der Wärmeentwicklung beim Verstärker und beim Spannungsregler sowie zum Schutz vor ESD ist das Arduino-Board samt AudioShield in ein geeignetes Gehäuse einzubauen und so vor Berührung zu schützen.

tragen. Die Klemmen werden, wie in Bild 5 zu sehen, zusammengesteckt und anschließend eingesetzt.

Die fertig bestückte Baugruppe wird abschließend noch einmal auf Bestückungsfehler, unsaubere Lötstellen usw. kontrolliert und kann dann auf das Arduino-Board gesteckt werden, wie es in Bild 6 aus verschiedenen Perspektiven gezeigt ist. Beim Aufstecken bzw. Herausnehmen ist darauf zu achten, dass keine Steckerkontakte verbogen werden.

[1] Arduino Uno, Best.-Nr. JU-10 29 70

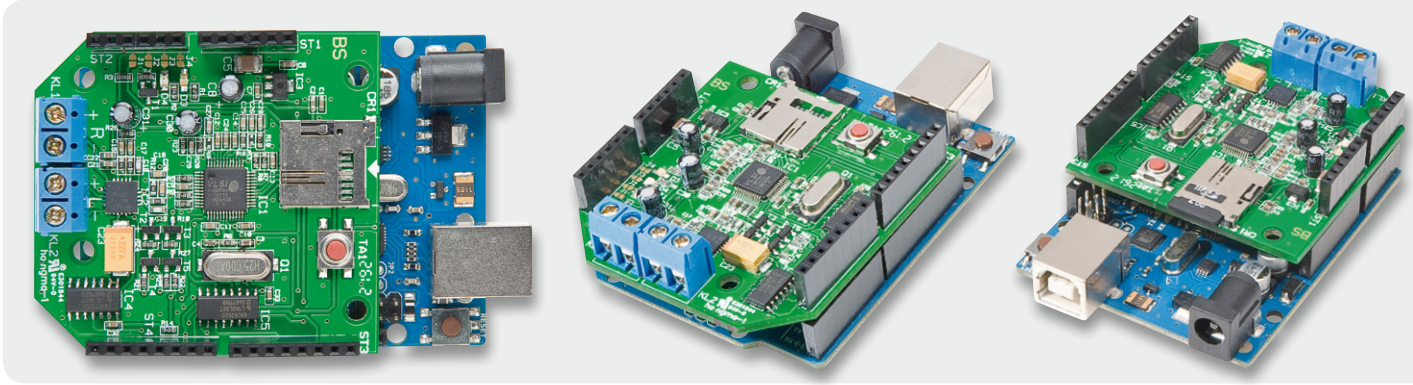


Bild 6: Das fertig bestückte Shield wird wie ein Standard-Arduino-Shield auf das Arduino-Board aufgesteckt, rechts ist bereits eine microSD-Karte eingesteckt.