

Projekt

Entwicklung der Steuerung einer Cocktailmaschine

vorgelegt von:

Steffen Pfiffner

am 13.03.2013

Einleitung

In dieser Projektarbeit wurde ein funktionsfähiger Prototyp einer Cocktailmaschine (Abb. 1) realisiert, der mit einem Android-Gerät kabellos bedient werden kann. Die Bediensoftware bietet unter anderem die Möglichkeit in einer Datenbank vorhandene Cocktails auszuschenken, eigene Cocktails zu erstellen und Benutzerstatistiken anzuzeigen. Die Cocktailmaschine kann mit Hilfe von Magnetventilen flüssige Zutaten dosieren. Ein horizontal beweglicher Schlitten ermöglicht das automatische Befüllen von mehreren Gläsern nacheinander. In Zukunft kann die Maschine einfach um einen Rührer oder eine Crusheis-Maschine erweitert werden, der/die dann oberhalb im Wirkungsbereichs des Schlittens montiert wird.

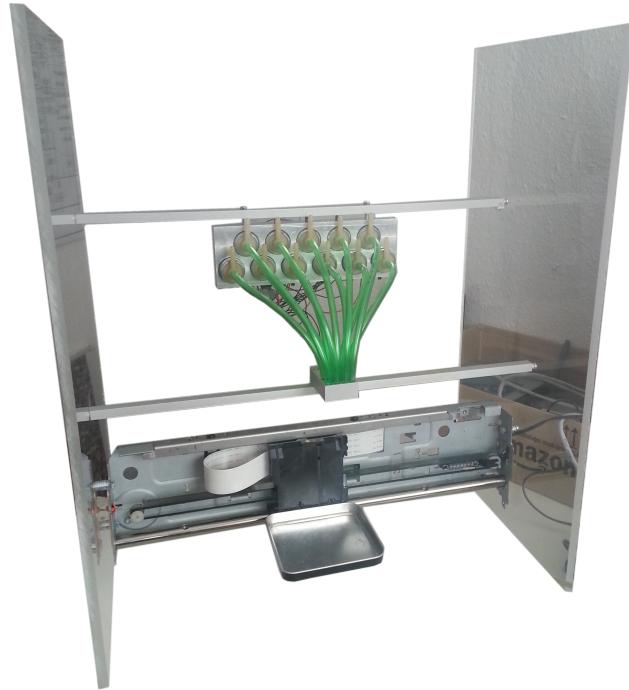


Abbildung 1: Cocktailmaschine

1 Funktionsprinzip

Oberhalb der Maschine werden Flaschen mit dem Deckel nach unten aufgehängt, die durch Schläuchen im Deckel mit den Ventilen verbunden sind. Je höher die Flaschen über der Maschine hängen, desto mehr Druck wird erzielt, da sich der Druck in einer Wassersäule pro Meter um 0.1 Bar erhöht. Wird ein Ventil geöffnet, fließt der Inhalt der am entsprechenden Ventil angeschlossenen Flasche durch das Ventil in ein Glas, welches auf dem dafür vorgesehenen Glashalter steht.

2 Systemübersicht

Eine Systemübersicht soll anhand des Ablaufs eines Bestellvorgangs gegeben werden (siehe Abb. 2):

Der Benutzer wählt in der **Drinkmixer Control Software**, die auf einem Android-Gerät läuft, den gewünschten Cocktail. Die Software schickt Befehle wie z.B. „Ventil 1 und 2 öffnen“ an die **Drinkmixer Embedded Software**, die auf dem Atmega32 eines AVR-NET-IO-Entwicklungsboards läuft. Beide Geräte befinden sich im selben Netzwerk und kommunizieren über TCP/IP. Die **Drinkmixer Embedded Software** sendet Schaltbefehle an das angeschlossene **Schieberegister** auf der **Interface Platine**, welches die gewünschten **Ventile** öffnet. Die an den entsprechenden Ventilen angeschlossenen Zutaten fließen durch die Ventile in das Glas. Die Dosierung der Zutaten funktioniert zeitlich. Nachdem ein zuvor berechneter Zeitraum abgelaufen ist, sendet die **Drinkmixer Control Software** einen Befehl der z.B. lautet „Ventil 1 und 2 schließen“, der zum Schließen der zuvor geöffneten Ventilen führt.

Darüber hinaus werden von der **Drinkmixer Embedded Software** zwei Motoren über einen auf der **Interface Platine** befindlichen **Motortreiber-Baustein** angesteuert. Ein Motor bewegt den horizontalen Schlitten. Der zweite Motor ist für die vertikale Steuerung eines Rührgeräts vorgesehen. Die vertikale Achse für das Rührgerät ist jedoch noch nicht an der Maschine vorhanden.

Ein **Quadratur Encoder** gibt Auskunft über die aktuelle Position des horizontalen Schlittens wird als Feedback für den PID-Regler verwendet der die Position des Schlittens steuert. Der **End-Schalter** wird zur Bestimmung der absoluten Nullposition des Schlittens benötigt.

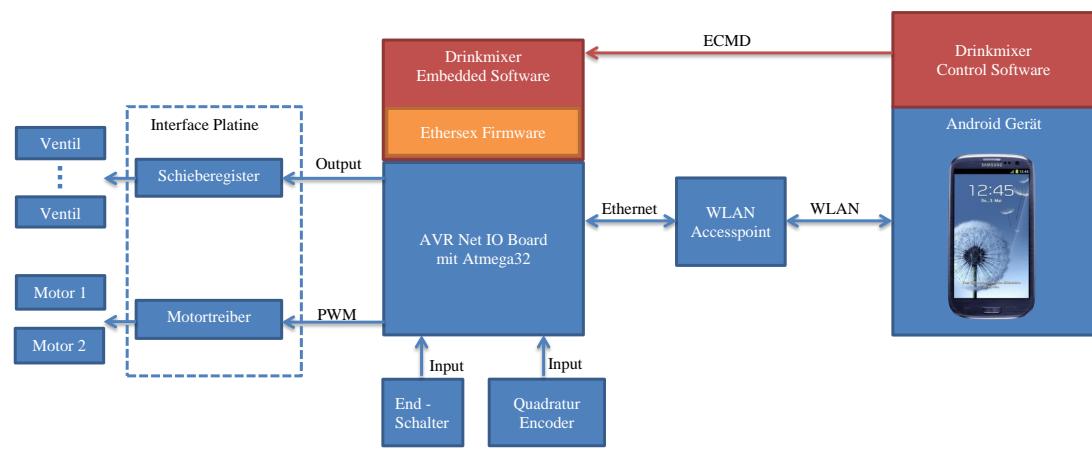


Abbildung 2: Gesamtübersicht. (Blau = Hardware, Rot = Software)

Inhaltsverzeichnis

1 Funktionsprinzip	3
2 Systemübersicht	3
3 Hardware	6
3.1 AVR-NET-IO-Entwicklerboard	6
3.2 Drinkmixer Interface Platine	7
3.3 Endschalter	8
3.4 Schieberegister	8
3.5 Motortreiber	9
3.6 Drucker Hardware	10
3.7 Gehäuse	13
3.8 Ventile	13
4 Drinkmixer Embedded Software	14
4.1 Ethersex Firmware	14
4.2 Pinning	16
4.3 ECMD	16
4.4 Port Erweiterungsmodul (HC595 output expansion)	16
4.5 Motor Ansteuerungs Modul (H-Bridge)	17
5 Drinkmixer Control Software	22
5.1 Programmierumgebung	22
5.2 Verwendete Bibliotheken	22
5.3 Funktionsweise	23
6 Probleme	32
7 Ergebnis	32
8 Erweiterungsmöglichkeiten	33
9 Anhang	35
9.1 Schaltplan Interface Platine	35
Literatur	36

3 Hardware

Im Folgenden werden die für die Cocktailmaschine benötigten Hardwarekomponenten beschrieben.

3.1 AVR-NET-IO-Entwicklerboard

Das AVR-NET-IO-Board (siehe Abb. 3) ist eine vollständig unabhängig arbeitende Ethernetplatine die mit dem Netzwerkcontroller ENC28J60 und einem ATmega32 Mikrocontroller ausgestattet ist.

Es wird zur Steuerung aller Hardwarekomponenten verwendet und erhält über die Netzwerkschnittstelle Steuerbefehle von der **Drinkmixer Control Software**.

Features:

- Netzwerkcontroller ENC28J60 RJ 45 Port
- Mikrocontroller ATMega32
- 8 digitale Ausgänge die über den 25-pol. Sub-D-Stecker abgegriffen werden
- 4 digitale Eingänge die ebenfalls den 25-pol. Sub-D-Stecker zugeführt werden
- 4 ADC-Eingänge (10 Bit) die direkt über die Anschlussklemmen eingespeist werden
- ISP-Steckerleiste (Atmel-Standardbelegung) für die Programmierung des Microcontroller ATMega32 mit geeignetem Schnittstellenadapter
- RS232-Schnittstelle (Tx, Rx) mit Pegelanpassung über MAX232
- separate AVRef-Referenzspannungszuführung
- 10-polige Pfostenleiste (Extension-Port) für weitere Mikrocontroller-Ports
- Betriebsspannung 9 V
- max. Stromaufnahme ca. 190 mA
- Maße (LxBxH): 108x76x22 mm

Das Entwicklungsboard ist bei [Pol12a] als Bausatz (19,95 Euro) oder also Fertigmodul (27,95 Euro) erhältlich. Dort ist auch das Handbuch des Moduls zu finden.

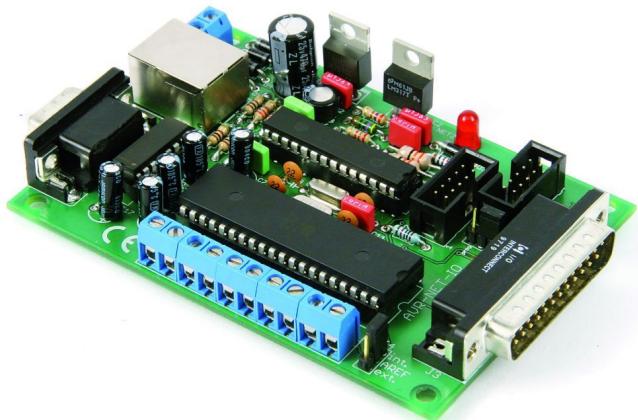


Abbildung 3: Das AVR-NET-IO-Board

3.2 Drinkmixer Interface Platine

Die **Drinkmixer Interface Platine** stellt die Verbindung zwischen AVR-NET-IO-Board und den anderen Hardware Komponenten her. Sie ist mit zwei Schieberegistern und einem Motortreiber bestückt. Per Netzteil wird die Platine mit 12V für die Ventile und die Motoren versorgt. Sie ist mit dem **EXT.** Anschluss des AVR-NET-IOs mit einem Flachbandkabel verbunden. Tabelle 1 und Abbildung 4 zeigen die Belegung des Flachbandkabels (die rote Ader des Kabels ist Pin 10).

EXT. Pin	μ Controller Pin	Name	Funktion
1	PD2	HC595_CLOCK	SCK (siehe Kap. 3.4)
2	PD3	HC595_DATA	SER (siehe Kap. 3.4)
3	PD4	HC595_STORE	RCK (siehe Kap. 3.4)
4	PD5 (OC1A)	HBRIDGE_1_ENABLE	PWM Motor 1
5	PD6	HBRIDGE_I1	Richtung Motor 1
6	PD7 (OC2)	HBRIDGE_2_ENABLE	PWM Motor 2
7	PB0	HBRIDGE_I2	Richtung Motor 2
8	PB3	Encoder Phase A	siehe Kap. 3.6.2
9	GND	GND	Masse
10	VCC	5V	Stromversorgung für Logik

Tabelle 1: Belegung des Flachbandkabels bzw. des Ext. Anschlusses des AVR-NET-IOs

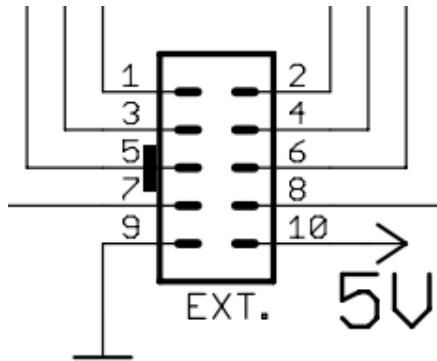


Abbildung 4: Belegung des Ext. Anschlusses des AVR-NET-IOs

3.3 Endschalter

Der Endschalter dient zur Bestimmung des Nullpunkts der horizontalen Achse. Er ist auf der linken Seite des Wirkungsbereichs des Schlittens angebracht und mit der Masse und Pin PB3 des AVR-NET-IOs verbunden.

3.4 Schieberegister

Zur Ansteuerung der Ventile werden TPIC6B595 8-Bit Schieberegister (Datenblatt siehe [Tex12]) verwendet. Diese erlauben es durch Kaskadierung beliebig viele Ventile anzusteuern und sind speziell für das Schalten induktiver Lasten ausgelegt. Die Ventile können dadurch direkt an das Bauteil angeschlossen werden. Da die Schieberegister Masseschaltend sind, können nachträglich auch Bauteile mit Spannungen bis zu 50V und einer Stromaufnahme von bis zu 500mA, z.B. LEDs, daran angeschlossen werden.

Schieberegister sind im Prinzip seriell -> parallel Wandler, die seriell Bits entgegen nehmen und ihre Ausgänge dann entsprechend schalten.

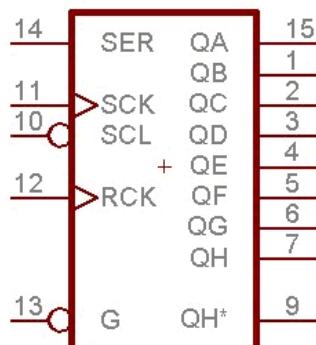


Abbildung 5: Anschlüsse eines Schieberegisters (Quelle: [Mik12])

Ein Bauteil enthält ein Schieberegister und ein Ausgangsregister. Durch eine steigende Taktflanke (LOW->HIGH) an **SCK** wird das am Eingang **SER** anliegende Bit ins Schieberegister übernommen. Alle anderen Bits werden weitergeschoben. Dieser Vorgang wird wiederholt bis alle Bit Positionen des Registers die gewünschten Werte haben. Durch einen Puls an **RCK** wird der aktuelle Inhalt des Schieberegisters in das Ausgangsregister übernommen und die Ausgänge entsprechend geschaltet (siehe Abb. 5). Mehrere Bauteile können kaskadiert werden indem Pin **QH*** mit dem **SER** Pin eines zweiten Bauteils verbunden wird.

Die Interface Platine ist mit zwei dieser Register bestückt und ermöglicht so die Ansteuerung von maximal 16 Ventilen.

Um Prozessor Pins einzusparen wird Pin **G** mit Masse verbunden. So sind die Ausgänge immer aktiviert. Außerdem wird **SCL** mit 5V verbunden. **SCL** auf 0 würde das Schieberegister komplett löschen. Dies kann aber auch durch Einschieben von 8 Nullen erreicht werden.

Weiter Informationen zu Schieberegistern gibt es unter [Mik12].

3.5 Motortreiber

Zur Ansteuerung der DC Motoren wird ein L293D vier Kanal Motortreiber verwendet (Datenblatt [STM12]). Er erlaubt es zwei Motoren in jeweils zwei Drehrichtungen zu betreiben. Die bereits integrierten Freilaufdioden ermöglichen den direkten Anschluss der Motoren.

Um einen Motor in zwei Richtungen zu betreiben sind zwei Kanäle nötig. Im Folgenden wird die linke Seite von Abb. 6 beschrieben. Diese ist für den ersten Motor zuständig. Die rechte Seite funktioniert analog.

An **ENABLE1** wird das vom Mikrocontroller generierte PWM Signal angeschlossen. Dieses ermöglicht es die Geschwindigkeit des Motors mit 8 Bit Auflösung (255 Stufen) zu steuern. PWM wird in Kapitel 4.5 näher beschrieben. Die Anschlüsse des Motors sind mit **OUT1** und **OUT2** verbunden. Die Eingänge **IN1** und **IN2** werden zur Richtungs-Steuerung verwendet. Tabelle 2 zeigt die möglichen Zustände mit den zugehörigen Funktionen. **Motor stoppen** wird hier nicht benötigt. Bei Drehrichtung links bzw. rechts sind IN1 und IN2 jeweils invertiert. Mit Hilfe eines vorgesetzten Inverter ICs kann die Richtung deshalb mit nur einem Mikrocontroller Pin gesteuert werden. Nähere Informationen zur Schaltung sind dem Schaltplan in Kapitel 9.1 zu entnehmen.

IN1	IN2	FUNKTION
0	1	Drehrichtung links
1	0	Drehrichtung rechts
1	1	Motor stoppen
0	0	Motor stoppen

Tabelle 2: IN1 und IN2 Zustandstabelle

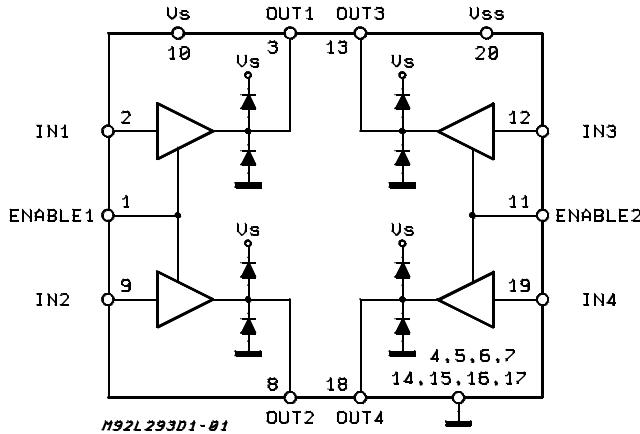


Abbildung 6: DL293D interne Schaltung (siehe [STM12])

Auf der Interface Platine befindet sich ein Motortreiber mit vorgeschaltetem Inverter IC. Der Inverter IC ist ein 74HC/HCT14 Hex inverting Schmitt trigger (siehe [Phi93]).

3.6 Drucker Hardware

Für den horizontalen Schlitten wird die Hardware und Mechanik eines HP Deskjet 940C Druckers verwendet. Diese kann leicht für die Cocktailmaschine angepasst werden und bietet zuverlässige Mechanik inklusive Motor und einem Quadratur Encoder.

Folgende Komponenten werden verwendet (siehe Abb. 7):

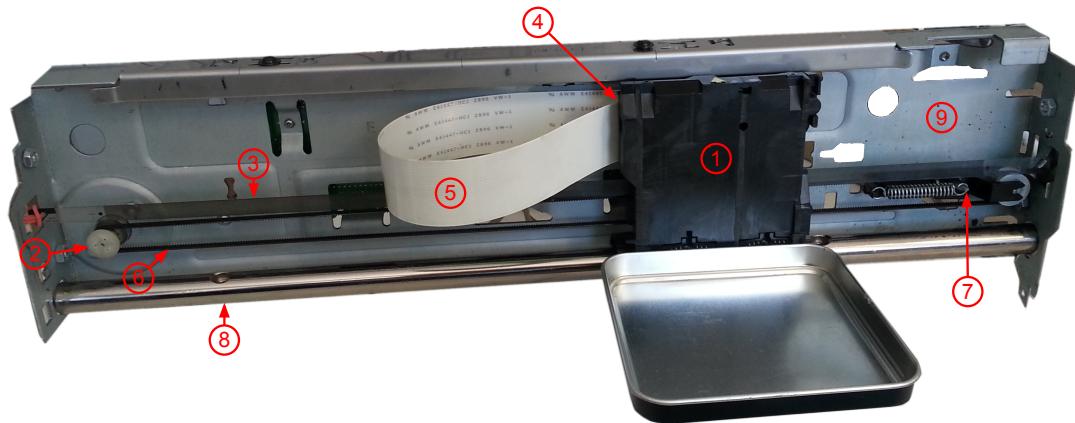


Abbildung 7: Die Druckerkomponenten

1. Zum Glashalter umfunktionierter Druckerschlitten.
2. Einen DC Motor mit unbekannten Eigenschaften inklusive Antiebswelle mit angebrachtem Zahnrad als Antrieb.
3. Optischer Encoder-Streifen mit 60 Linien pro cm zur Positionsbestimmung des Schlittens.
4. Auf der Rückseite des Schlittens befindet sich eine mit einem Quadratur Encoder von Agilent bestückte Platine, die mit 8 verbunden ist (siehe Abb. 9).
5. Flexibles Flachbandkabel, das für Verbindung zum Quadratur Encoder verwendet wird.
6. Zahnriemen der hinten am Schlitten befestigt ist und zur Kraftübertragung vom Motor auf den Schlitten dient.
7. Spannvorrichtung die den Zahnriemen auf Spannung hält.
8. Führungsschiene für den Schlitten.
9. Metallgehäuse, an dem alle Komponenten befestigt sind.

3.6.1 Motor

Der Motor ist ein 12 oder 18V Gleichstrommotor. Leider konnte zu diesem Bauteil kein Datenblatt gefunden werden. Er wird hier mit 12V betrieben.

3.6.2 Quadrature Encoder

Der Quadrature Encoder in Verbindung mit dem Encoder-Streifen wird zur Positionsbestimmung des Schlittens verwendet.

Das Datenblatt dieses Encoders konnte leider nicht gefunden werden, unter [Agi02] ist jedoch das Datenblatt eines vergleichbaren Moduls zu finden.

Ein Encoder besteht aus einer LED auf der einen und zwei Phototransistoren auf der anderen Seite. Die LED leuchtet durch einen Encoder-Streifen mit dünnen aufgedruckten Linien. Die Phototransistoren sind genau die halbe Linienbreite von einander entfernt angeordnet. Wenn sich der Encoder relativ zum Encoder-Streifen bewegt, produzieren die Phototransistoren zwei Sinus Signale (hier 0...5V), die der Frequenz der vorbeifahrenden Linien des Encoder-Streifens entspricht. Das Signal ist aber durch die spezielle Anordnung der Phototransistoren um 90 Grad phasenverschoben. Dies ermöglicht es neben dem Zählen der Impulse auch die Bewegungsrichtung festzustellen. (siehe Abb. 8)

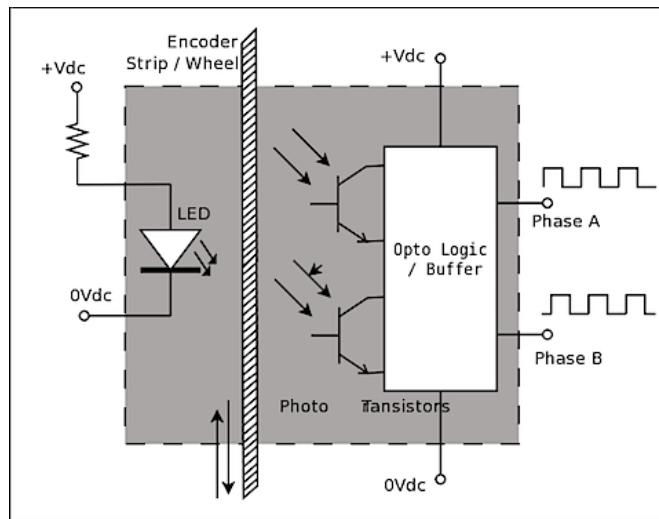


Abbildung 8: Schematische Darstellung der Funktionsweise eines Quadrature Encoders (siehe [Mad11])

Der Encoder ist an die 5V Stromversorgung und Masse des Atmega32 angeschlossen. Phase A ist mit Pin PB3 (AIN1: Analog Comparator Eingang) und Phase B mit Pin PD1 des Atmega32 verbunden. Diese Verbindungen werden über die im Drucker vorhandene Encoder Platine (siehe App. 9) die auf der Rückseite des Drucker Schlittens sitzt über das flexible Flachbandkabel zum AVR-NET-IO-Board geführt. Die Verarbeitung dieser Signale wird in Kapitel 4.5.3 beschrieben.

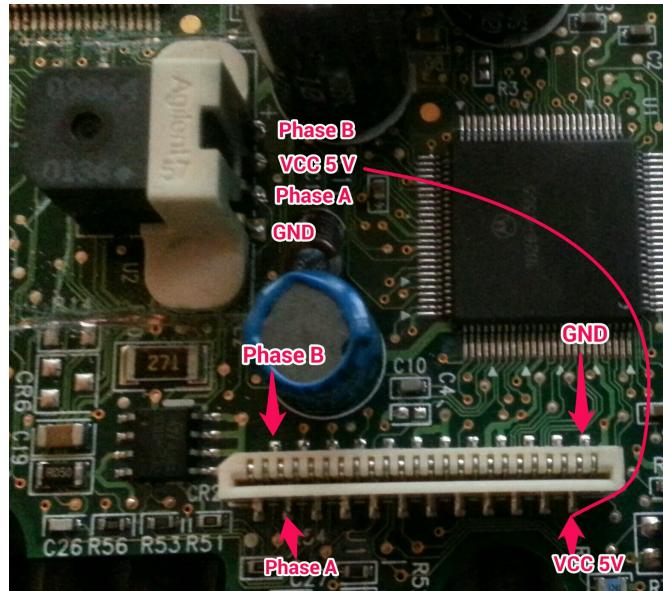


Abbildung 9: Belegung des Steckers bzw. des Flachbandkabels der Encoder Platine. Die rote Linie ist eine zusätzlich hinzugefügte Drahtbrücke.

3.7 Gehäuse

Das Gehäuse (siehe Abb. 1) besteht aus zwei 8mm starken Plexiglas Scheiben (250mmx500mm). Dazwischen sind die Druckermechanik (siehe Kap. 3.6), ein Ventilhalter für die Ventile (siehe 3.8) und ein Schlauchhalter, an dem die Schläuche zusammengeführt werden, befestigt. Als Befestigung dienen Alu U-Profile (10mmx8mmx405mm).

3.8 Ventile

Zur Dosierung der Getränke werden Druckluft-Magnetventile des Typs SH-V0829BC-R (siehe Abb. 10) verwendet. Diese sind bei z.B. bei [Pol12b] günstig erhältlich. Die Ventile sind im stromlosen Zustand geschlossen. Die Betriebsspannung beträgt 12 V und die Stromaufnahme 100mA. Als Zu- und Ableitung werden PVC Schläuche mit 4 mm Innendruckmesser und einer Wandstärke von 1 mm verwendet (4 x 1). Die 11 verwendeten Ventile werden in einem Ventilhalter fixiert.



Abbildung 10: Druckluft-Magnetventil des Typs SH-V0829BC-R

4 Drinkmixer Embedded Software

Die **Drinkmixer Embedded Software** besteht aus der Firmware **Ethersex** mit den Modulen **HC595 output expansion** zur direkten Ansteuerung der Schieberegister und dem Modul **H-Bridge** zur Ansteuerung der Motoren. Sie wird auf dem Atmega32 des AVR-NET-IO-Boards ausgeführt.

4.1 Ethersex Firmware

Ethersex ist eine unter GPLv3 lizenzierte Firmware mit Netzwerk Unterstützung für 8-bit AVR Mikrocontroller. Viele AVR Entwickler Boards werden mit allen Hardware Features unterstützt, z.B. auch das hier verwendete AVR-NET-IO-Board von Pollin.de. Alle gängigen Netzwerk Protokolle sind implementiert und fertige Module zur Ansteuerung von Hardware wie z.B. USB Geräten, RFM12 Funk Modulen, Schieberegistern oder LCD Displays sind vorhanden. Eine Liste alle Features ist unter [Eth12a] zu finden.

4.1.1 Programmierumgebung einrichten

Als Programmierumgebung für die Embedded Software wird Ubuntu 11.04 verwendet.

Zum Kompilieren der Ethersex Firmware sind folgende Voraussetzungen nötig:

- AVR GCC-Compiler (Version 4.1 oder höher)
- AVR LIBC (Version 1.6.8, für 128er ATMegas 1.7)
- GNU-Tools (Bash, Make, m4, awk)
- AVR-Programmer (hier wird avrdude verwendet)

Alle Programme können mit einem Befehl installiert werden:

```
apt-get install gcc-avr avr-libc binutils-avr m4 gawk  
libncurses5-dev make dialog git-core avrdude
```

Der komplette Quellcode kann mit dem Befehl

```
git clone https://github.com/fivef/ethersex.git
```

von GitHub heruntergeladen werden (siehe [Eth12b]).

4.1.2 Konfiguration

Ethersex bietet ein grafisches Konfigurations-Tool um die gewünschten Software-module auszuwählen und Grundeinstellungen vorzunehmen. Diese Tool wird mit `make menuconfig` gestartet.

Alle für die Cocktailmaschine nötigen Einstellungen werden über **Load an Alternate Configuration File** geladen. In das Eingabefeld wird dazu der Dateiname SHIFT_MOTOR_WORKING eingegeben.

Mit `make` wird Ethersex kompiliert. (Falls beim Kompilieren der Fehler `Assertion failure in obj_elf_init_stab_section at config/obj-elf.c line 1782` auftritt, kann dieser ignoriert werden. Er wird wahrscheinlich von der verwendeten binutils-avr-Version ausgelöst, beeinträchtigt aber die Funktion der Software nicht.)

Make erzeugt eine `ethersex.hex` Datei im Ethersex Sammverzeichnis.

Die so geladenen Einstellungen basieren, mit Ausnahme der folgenden Annahmen, auf den Standard-Einstellungen des AVR-NET-IO-Boards die mit **Load a Default Configuration** geladen werden können.

Hier die Einstellungen im Einzelnen:

- **General Setup->Enable Debugging->(115200) UART Baudrate**
Debugging über die RS232 Schnittstelle aktivieren und die Baudrate auf 115200 stellen. Vom PC aus können so alle Debugausgaben mit einem beliebigen Terminal Programm (hier wurde HTerm verwendet) mit den Einstellungen **Baudrate: 115200, Data: 8, Stop:1, Parity: None, Flow Control: no** angezeigt werden.
- **Network->Ethernet (ENC28J60) support->IP address: 192.168.178.90**
Einstellen der IP Adresse auf 192.168.178.90 und Subnetzmaske auf 255.255.255.0
- **I/O->(Full-featured) I/O abstraction model (Port I/O)**
Alle I/O Einstellungen aktivieren/sichtbar machen
- **I/O->HC595 output expansion->(2) Number fo HC595 registers**
Das Modul zur Ansteuerung des Schieberegisters aktivieren. Anzahl der Schieberegister auf zwei setzen.
- **I/O->H-Bridge (EXPERIMENTAL)->H-Bridge Debug ist aktiviert.**
Das H-Bridge-Modul zur Ansteuerung des Motortreibers aktivieren.
- **Protocols->ECMD (Ethersex Command) support->USART (RS232) und TCP/Telnet sind ausgewählt**
Das Ethersex Command Protokoll (ECMD) (siehe Kap. 4.3) an der RS232 Schnittstelle und im Netzwerk aktivieren.

4.1.3 Flashen

Diese ethersex.hex Datei wird mit Hilfe eines USB-Programmers und einem beliebigen AVR-Chip-Brenn-Programm auf den Mikrocontroller geflasht.

Für dieses Projekt wurde ein USBasp USB-programmer von [Tho12] und das Linux Programm avrdude verwendet. Dieses wurde in Kapitel 5.1 bereits installiert. Der Befehl `sudo avrdude -c usbasp -p m32 -U flash:w:ethersex.hex` lädt

die `ethersex.hex` auf den Atmega32. Sollte ein anderer Programmer verwendet werden muss nur der Parameter `-c usbasp` angepasst werden. Der Programmer wird am ISP Port des AVR-NET-IO-Boards angeschlossen und der Jumper direkt neben dem ISP Port wird auf „prog“ gesetzt.

Die Fusebits des ATMEGA32 bleiben bei den Standardeinstellungen des mit dem AVR-NET-IO-Board gelieferten Mikrocontrollers.

Sie lauten wie folgt:

```
Low Fuse: 0xBF
High Fuse: 0xCB
Lock Fuse: 0xFF
Calibration: 0xB2B2B2B1
```

4.2 Pinning

Die Datei `<Ethersex Verzeichnis>/pinning/hardware/netio.m4` enthält im Abschnitt „conf_HBRIDGE“ die Zuweisung von Mikrocontroller Pins zu den in der Software definierten Konstanten für diese Pins. Siehe Kap. 3.2 für die Zuordnung.

4.3 ECMD

Das Ethersex Command Protokoll (ECMD) ist ein Protokoll das zur Kommunikation mit der Ethersex Firmware über die RS232 Schnittstelle oder über TCP/IP (HTTP oder Telnet) verwendet werden kann. Die **Drinkmixer Control Software** nutzt dieses Protokoll um Befehle an die **Drinkmixer Embedded Software** zu senden. Folgende Befehle wurden für das H-Bridge Ethersex Modul in der Datei `<Ethersex Verzeichnis>/hardware/hbridge/ecmd.c` definiert:

<code>hbridge setpoint <int></code>	Setzt die Stellgröße des PID-Reglers auf den Wert <code><int></code> in Encoderimpulsen (siehe Kap. 4.5.3).
<code>hbridge kp <float></code>	Setzt den kp Anteil des PID-Reglers (siehe Kap. 4.5.1). Standardwert: 0.5
<code>hbridge ki <float></code>	Setzt den ki Anteil des PID-Reglers (siehe Kap. 4.5.1). Standardwert: 0.0007
<code>hbridge acc <float></code>	Setzt die Steilheit der Rampe (siehe Kap. 4.5.1). Standardwert: 1

4.4 Port Erweiterungsmodul (HC595 output expansion)

Zur Ventil Ansteuerung wird das in Ethersex enthaltene **HC595 output expansion** Modul verwendet. Dieses erweitert den Atmega32 um zwei weitere virtuelle Ports E und F, die jeweils 8 Pins enthalten. Diese Pins werden durch die

Drinkmixer Control Software direkt per ECMD zur Steuerung der Ventile geschaltet. Das Modul befindet sich im Ordner <Ethersex Verzeichnis>/hardware/io_expander.

4.5 Motor Ansteuerungs Modul (H-Bridge)

Zur Motor Ansteuerung wird das in Ethersex enthaltene **H-Bridge (EXPERIMENTAL)** Modul verwendet. Dieses Modul ist normalerweise zum Steuern eines Roboters mit zwei Motoren gedacht und befindet sich noch in der Experimentierphase. Für dieses Projekt wurden einige Funktionen und das Gerüst des Moduls als Grundlage für ein eigenes Modul zur Regelung der Motorposition verwendet. Das Modul befindet sich im Ordner <Ethersex Verzeichnis>/hardware/hbridge. Die in Kapitel 3.6 beschriebene Druckerhardware in Kombination mit der hier beschriebenen Regelungssoftware bilden zusammen einen Servomotor.

Als Servomotor werden elektrische Motoren verschiedener Bauart bezeichnet, die mit einem Servoregler (der aus einem Servoverstärker und meistens weiteren Regelkreis-Übertragungsgliedern besteht) einen Servoantrieb bilden. Die Servomotoren werden in einem geschlossenen Regelkreis betrieben. Der Betrieb kann momenten-, geschwindigkeits- oder positionsgeregelt sein.”[Wik12]

Hier wird der Servomotor positionsgeregelt betrieben, da der Schlitten bestimmte Positionen anfahren soll.

4.5.1 PID-Regelung der Position

Zur Regelung der Schlitten-Position wird ein PID-Regler verwendet. Abbildung 11 beschreibt den gesamten Regelkreis.

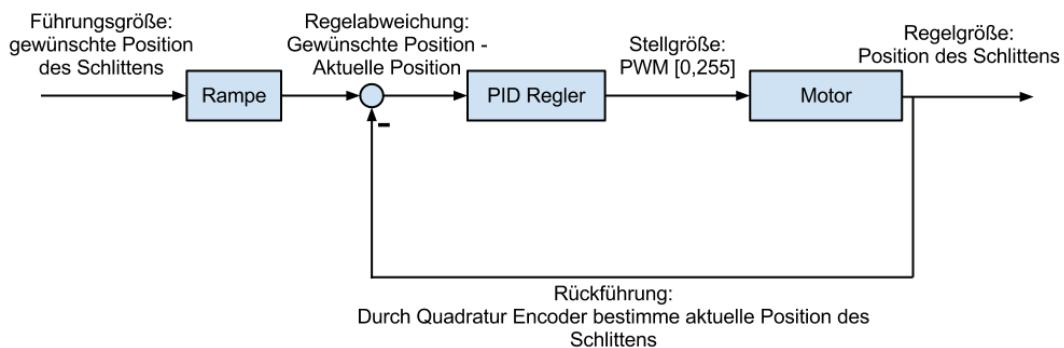


Abbildung 11: Regelkreis zur Positionierung des Schlittens

Der Kasten „Rampe“ ist eine Rampen-Funktion, die große Sprünge der Führungsgröße verhindern soll. Wird die Führungsgröße z.B. von 0 auf 100 geändert, sorgt

die Rampen-Funktion dafür, dass sich die Frührungsgröße des PID-Reglers nur um einen Schritt pro Zeitintervall (hier 20ms) ändert. Dadurch werden zu hohe Beschleunigungen verhindert.

12 zeigt den Kasten „PID-Regler“ genauer. Der PID-Regler erhält die aktuelle Regelabweichung $e(t)$ und gewichtet den proportionalen, integrierenden und differenzierenden Anteil gemäß der Faktoren k_p , k_i und k_d . Das Ergebnis ist die neue Stellgröße.

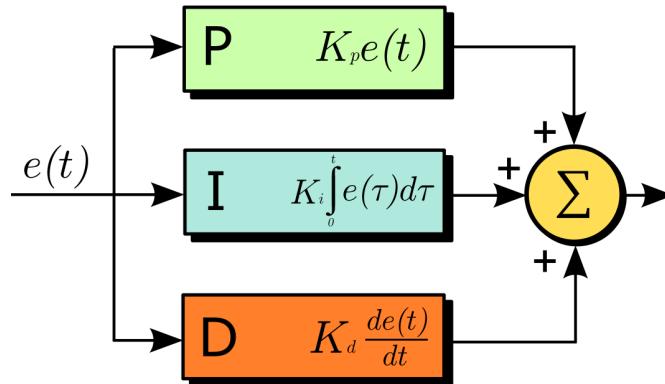


Abbildung 12: Aufbau eines PID-Reglers (siehe [CoA12])

Die Berechnung der P,I und D Anteilen ist zeitdiskret einfach möglich. Als Abtastzeit (T_a), die Zeit zwischen zwei Aufrufen der PID-Regler-Update-Funktion, wurden 20ms gewählt. Dies entspricht der kleinsten Zeitspanne, die mit der in Ethersex eingebauten timer() Meta-Funktion möglich ist. Der Code des verwendeten PID-Regler stammt von [CoA12]. Dort ist auch eine kurze Einführung zu PID-Reglern zu finden.

Zur Einstellung des PID-Regler wurde ein eigenes Verfahren entwickelt, da die gängigen Verfahren für Online-Tuning wie Ziegler-Nichols, Tyreus-Luyben und Cohen-Coon immer nur auf hohe Geschwindigkeiten ausgelegt sind. Hier spielt die Geschwindigkeit jedoch eine untergeordnete Rolle. Viel wichtiger ist es, dass keine großen Beschleunigungen wirken, da auf dem Schlitten später Getränke stehen. Deshalb wird der PID-Regler so eingestellt, dass er langsam beschleunigt und abbremst. Der D-Anteil des Reglers wird nicht verwendet und auf 0 gesetzt, da dieser Anteil nur für hohe Regelgeschwindigkeiten zuständig ist. Um das gewünschte Verhalten zu erreichen, wurde folgendes Verfahren zur Bestimmung der Parameter k_p , k_i entwickelt:

1. k_p und k_i werden auf 0 gesetzt.
2. k_i wird soweit erhöht bis das System anfängt zu schwingen.
3. k_i wird verringert bis kein Schwingen mehr auftritt.

4. k_p wird erhöht bis die gewünschte Anregelzeit erreicht wird.
5. Durch Feintuning von k_p und k_i kann die Beschleunigung beim Anfahren und Abbremsen eingestellt werden.

Grundsätzlich gilt:

- Vergrößern von k_p verkleinert die Anregelzeit aber vergrößert stationäre Regelabweichung.
- Vergrößern von k_i vermindert stationäre Regelabweichung aber vergrößert die Anregelzeit.

Folgende Werte wurden für ein gutes Regelverhalten bei mäßigen Beschleunigungen ermittelt:

$$k_p = 0.5$$

$$k_i = 0.0007$$

4.5.2 PWM

Die Geschwindigkeit des Motors wird per Pulse Width Modulation (PWM) gesteuert. Für diese Anwendung wird 8-Bit Fast-PWM verwendet. Das bedeutet, die Geschwindigkeit des Motors kann in 256 Stufen gesteuert werden. Dieser PWM Wert ist die vom PID-Regler manipulierte Stellgröße.

Die PWM Frequenz wird normalerweise oberhalb des für den Menschen hörbaren Bereichs gewählt ($> 20\text{kHz}$). Hier wird jedoch eine Frequenz von nur 60Hz verwendet, da es nur diese niedrige Frequenz erglaubt, den verwendeten Motor mit niedrigen Drehzahlen zu betreiben. Alle möglichen „prescale factor“ Einstellungen des „Timer/Counter2“ des Atmega32 wurden getestet. Der Faktor 1024 führte zum besten Ergebnis bei niedrigen Drehzahlen. Die PWM Frequenz wird durch Setzen des „prescale factors“ auf 1024 auf 61Hz festgelegt (siehe Gleichung 1).

$$f_{OCnPWM} = \frac{f_{clk_I/O}}{\text{prescale_factor} \cdot 256} = \frac{16\text{MHz}}{1024 \cdot 256} \approx 61\text{Hz} \quad (1)$$

Die für diese Einstellungen benötigten Register sind dem Atmega32 Handbuch ([Atm11]) entnommen.

4.5.3 Encoder Signal verarbeiten

In diesem Kapitel wird beschrieben wie das Signal des Quadrature Encoders verarbeitet wird. Abbildung 13 zeigt die beiden Signal-Phasen A und B. Ist A die führende Phase, bewegt sich der Encoder in eine Richtung (hier links) relativ zum Encoder-Streifen, sonst in die andere Richtung (hier rechts).

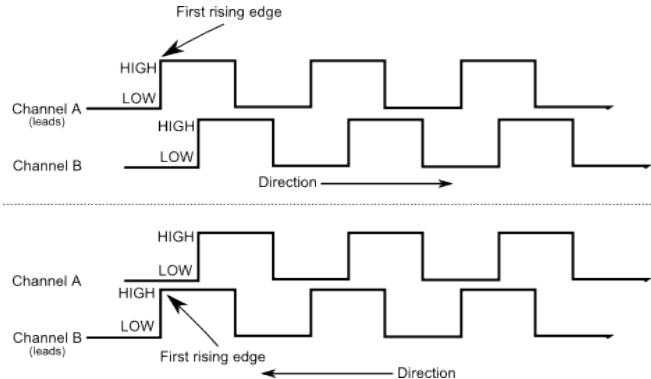


Abbildung 13: Auswertung der beiden Phasen des Quadrature Encoders (Quelle: [Gor11])

Phase A ist am Analogkomparator des Atmega32 angeschlossen (Atmega32 Pin PB3). Dieser erzeugt für jede steigende und fallende Taktflanke einen Interrupt. Die Interrupt Service Routine (ISR) enthält den in Abb. 14 gezeigten Zähl-Algorithmus, der auch die Richtung beachtet. Phase B ist am Pin PA1 des Atmega32 angeschlossen.

Dieser Zähler enthält die aktuelle Position des Schlittens relativ zum Nullpunkt in der Einheit Encoderimpulse. Der mit dem Encoder verwendete Encoderstreifen hat eine Auflösung von 60 Strichen/cm. Da bei steigender und fallender Taktflanke ein Interrupt ausgelöst wird, ist die effektive Auflösung 120 Impulse/cm. Ein Impuls entspricht somit $\frac{10\text{mm}}{120} = 0,083\text{mm}$. Die Auflösung könnte nochmals verdoppelt werden indem auch bei Phase B bei steigender und fallender Flanke ein Interrupt ausgelöst wird. Die nur mit Phase A erreichte Auflösung ist aber für diese Anwendung völlig ausreichend.

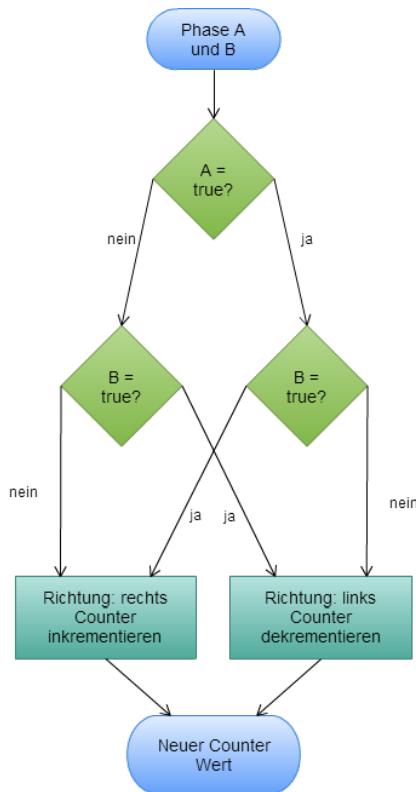


Abbildung 14: Zähl ISR

5 Drinkmixer Control Software

Die **Drinkmixer Control Software** wird auf einem beliebigen Gerät mit Android-Betriebssystem und WLAN-Unterstützung ausgeführt. Sie ermöglicht die Fernsteuerung der Cocktailmaschine. Die Software ist unter der GPLv3 Lizenz auf Github.com veröffentlicht.

```
git clone https://github.com/fivef/Drinkmixer.git
```

5.1 Programmierumgebung

Als Programmierumgebung für die **Drinkmixer Control Software** wird Windows 8 x64 und die IDE Eclipse in der Version Indigo verwendet. Folgende Voraussetzungen sind zum Kompilieren des Codes notwendig:

- Das „ADT Bundle for Windows“ enthält alle Voraussetzung die zur Entwicklung von Android Apps erforderlich sind. Erhältlich unter [Goo13]. Hier wurden folgende Versionen der Komponenten verwendet:
 - Android SDK Tools Version 20.0.3
 - Android SDK Platform-tools Version 14
 - Eclipse Plugins (u.a. Android Development Tools) Version 21
- Java SE Development Kit Version 7u7 von Oracle

Als Target Plattform wird Android Version 4.0 (API 14) verwendet. D.h. die Software ist nur mit Geräten kompatibel die mindestens Android Version 4.0 Ice Cream Sandwich installiert haben. Falls die Software für Android Versionen kleiner API 14 kompiliert werden soll, wird von Google ein Kompatibilitäts-Paket zur Verfügung gestellt, welches aber hier nicht verwendet wird. Für die Inkompatibilität mit älteren Versionen sind hauptsächlich die verwendete Fragment Struktur und die dazugehörenden Menüs verantwortlich.

5.2 Verwendete Bibliotheken

Folgende Bibliotheken werden von der Drinkmixer Control Software verwendet:

- **JEthernetControl 0.1.1** ist eine Java API zur Ansteuerung von Ethersex basierten Geräten per ECMD Protokoll [Rud13].
- **db4o 9.1.249.16099** ist eine open-source objektorientierte Datenbank [VER13].

5.3 Funktionsweise

Die Software läuft komplett innerhalb einer Android „Activity“. Bei Bedarf werden die verschiedenen „Fragments“ der Benutzeroberfläche angezeigt. Im Folgenden wird die Funktionsweise der Software anhand der entsprechenden Fragmente erklärt. Die Navigation erfolgt, Android üblich, mit den **Menü** und **Zurück** Hardware bzw. Software Tasten.

5.3.1 Benutzer Auswahl Fragment

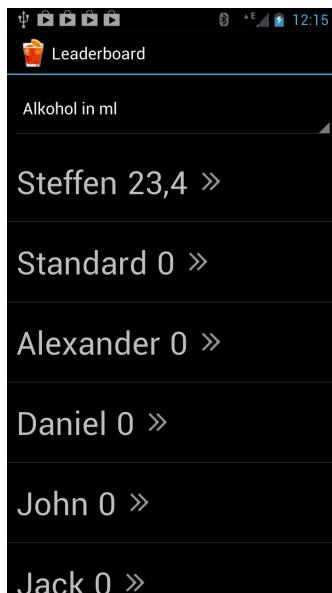


Abbildung 15: Benutzer Auswahl Fragment

Das **Benutzerauswahl Fragment** (Abb. 15) wird beim Start der Software angezeigt. Es dient zur Auswahl des Benutzers der sich einen Cocktail zubereiten lassen will. Mit dem Dropdown Menü oben (Abb. 16) kann das Sortierkriterium für die Liste geändert werden. Die Benutzerauswahl dient so gleichzeitig als eine Rangliste. Alle Werte werden anhand der bereits durch den jeweiligen Benutzer zubereiteten Cocktails berechnet. Die Auswahl eines Benutzers geschieht durch eine Wisch-Geste von links nach rechts. Dadurch wird eine unbeabsichtigte Auswahl erschwert. Wurde ein Benutzer ausgewählt, wird das **Cocktail Auswahl Fragment** geladen.

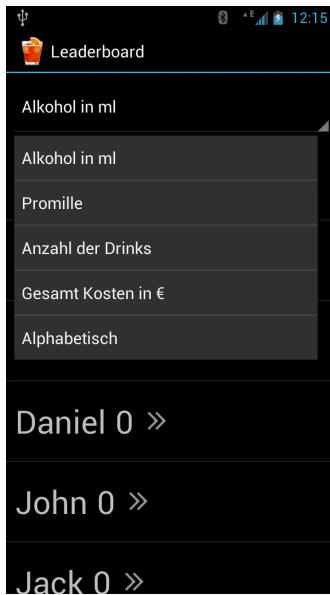


Abbildung 16: Benutzer Auswahl Dropdown Menü

5.3.2 Cocktail Auswahl Fragment

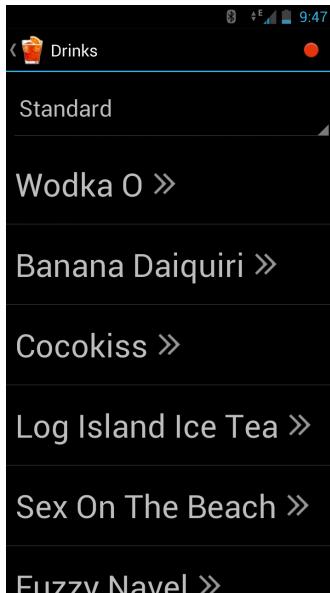


Abbildung 17: Cocktail Auswahl Fragment

Das **Cocktail Auswahl Fragment** (Abb. 17) bietet dem Benutzer alle Cocktails an, die mit der aktuellen Ventilbelegung der Cocktailmaschine hergestellt werden können (siehe 5.3.6). Durch eine Wisch-Geste von links nach rechts wird der gewünschte Cocktail zubereitet. Das **Cocktail zubereiten Fragment** wird angezeigt.

Über die **Menü**-Taste werden folgende Aktionen angeboten:

- **Alle anzeigen** bzw. **Mögliche anzeigen** zeigt wahlweise alle sich in der Datenbank befindlichen Cocktails oder nur die mit der aktuellen Belegung möglichen an.
- **Drinks bearbeiten** wechselt in die Bearbeiten-Ansicht des **Cocktail Auswahl Fragments**. Dort kann ein Cocktail ausgewählt und dann im Menü die gewünschte Aktion gewählt werden. Mögliche Aktionen sind:
 - **Neuer Cocktail hinzufügen** öffnet das **Cocktail bearbeiten Fragment** für einen neuen Cocktail.
 - **Ausgewählten Cocktail bearbeiten** öffnet das **Cocktail bearbeiten Fragment** des ausgewählten Cocktails.
 - **Ausgewählten Cocktail duplizieren** erstellt eine Kopie des ausgewählten Cocktails.
 - **Ausgewählten Cocktail löschen** löscht den ausgewählten Cocktail.

Je nach Bildschirmausrichtung und -größe werden die Optionen als Symbol oder als Symbol mit Text angezeigt. Ein Tooltip mit einer textuellen Beschreibung des Symbols kann unter Android durch Gedrückthalten des jeweiligen Symbols eingeblendet werden.

- **Belegung** öffnet das **Ventilbelegung bearbeiten Fragment**
- **Zutaten** öffnet das **Zutaten bearbeiten Fragment**
- **Benutzer** öffnet das **Benutzerverwaltung Fragment**
- **Einstellungen** öffnet das **Sonstige Einstellungen Fragment**
- **Neue Session** setzt die Benutzerstatistiken zurück, addiert die aktuellen Werte aber zu den Werten in einer „All-Time“-Datenbank, die alle Werte seit der Erstellung der Datenbank bzw. des Benutzer aufsummiert.

Der Indikator rechts oben zeigt den Verbindungsstatus zur Cocktailmachine an. Grün bedeutet, dass eine Verbindung hergestellt ist. Rot bedeutet, dass keine Verbindung vorhanden. Durch Antippen des Indikators kann manuell ein Verbindungsversuch gestartet werden. Beim Verbindungsauftbau bucht sich das Android Gerät zunächst in das WLAN-Netz mit der SSID „WLAN“ des zur Cocktailmachine gehörenden Routers ein. Die Verbindung wird dann zur IP-Adresse 192.168.178.90 des AVR-NET-IO-Boards aufgebaut. Wird keine WLAN mit der SSID „WLAN“ gefunden oder kann die IP Adresse nicht erreicht werden, schlägt der Verbindungsauftbau fehl.

5.3.3 Cocktail zubereiten Fragment

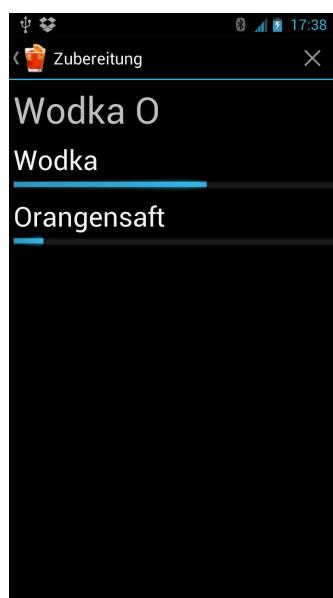


Abbildung 18: Cocktail zubereiten Fragment

Dieses Fragment zeigt den Fortschritt des aktuellen Cocktails an (App. 18). Zunächst werden alle Ventile der entsprechenden Zutaten gleichzeitig geöffnet. Je nach berechneter Dosierung schließen sich die Ventile dann nacheinander. Dabei wird für jede Zutat ein Ladebalken angezeigt der genau so lange läuft wie das zugehörige Ventil geöffnet bleibt. Die Fertigstellung des Cocktails wird durch eine Sprachausgabe der Form „Johns Wodka O ist fertig“ angezeigt. Danach wird automatisch das **Benutzer Auswahl Fragment** angezeigt.

Mit dem Kreuz oben rechts kann der Vorgang abgebrochen werden.

5.3.4 Cocktail bearbeiten Fragment

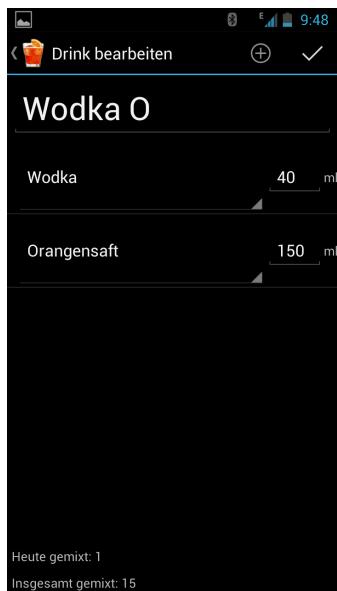


Abbildung 19: Cocktail bearbeiten Fragment

Das **Cocktail bearbeiten Fragment** (Abb. 19) erlaubt es dem Benutzer Cocktails zu bearbeiten. Der Name des Cocktails, die enthaltenen Zutaten und deren Dosierung können hier verändert werden. Das + Symbol im Menü oben rechts fügt dem Cocktail eine neue Zutat hinzu. Durch Gedrückthalten einer bereits vorhandenen Zutat und Auswahl von **Löschen** im sich öffnenden Menü, kann eine Zutat gelöscht werden. Im Dropdown Menü der Zutaten kann eine beliebige Zutat aus der Datenbank gewählt werden. Im Feld rechts wird die gewünschte Dosierung in ml angegeben. Durch Auswahl des Symbols V im Menü oder Be-tätigen der **Zurück**-Taste werden die Änderungen übernommen. Das **Cocktail Auswahl Fragment** wird geladen.

Unten wird angezeigt wie oft dieser Cocktail in der aktuellen Session und seit Erstellung der Datenbank zubereitet worden ist.

5.3.5 Zutaten bearbeiten Fragment

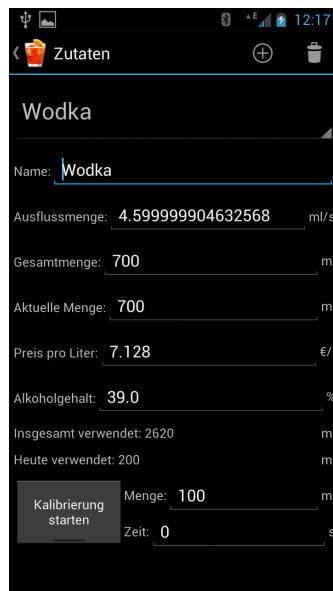


Abbildung 20: Zutaten bearbeiten Fragment

Das **Zutaten bearbeiten Fragment** (Abb. 20) ermöglicht es dem Benutzer die Datenbank um neue Zutaten zu erweitern (Symbol +), Zutaten zu löschen **Mülleimer** oder Eigenschaften bestehender Zutaten zu ändern.

Folgende Eigenschaften können angegeben werden:

- Der **Name** der Zutat.
- Die **Ausflussmenge** der Zutat in ml/s wird für die zeitliche Dosierung der Zutaten benötigt. Die Ausflussmenge kann manuell angegeben werden oder mit Hilfe des integrierten Kalibrierungsprogramms bestimmt werden. Dazu wird die zu kalibrierende Zutat an die Cocktailmachine angeschlossen und ein Messbecher unter den Schlauchhalter gestellt. Unten im **Zutaten bearbeiten Fragment** kann die Kalibrierung, durch Betätigen des **Kalibrierung starten** Buttons, gestartet werden. Das zu dieser Zutat gehörende Ventil öffnet sich. Hat der Füllstand genau 100 ml erreicht, wird die Kalibrierung durch Betätigen des **Kalibrierung stoppen** Buttons beendet. Anhand der gemessenen Zeit und der ausgeflossenen Flüssigkeit wird die Ausflussmenge in ml/s berechnet und als Ausflussmenge übernommen.
- Die **Gesamtmenge** der Zutat ist normalerweise der gesamte Flascheninhalt der Zutat in ml.

- Die **Aktuelle Menge** der Zutat ist der aktuelle Füllstand der Flasche der Zutat in ml. Diese wird berechnet indem alle ausgegebenen Dosen der Zutat von der Gesamtmenge abgezogen werden.
- Der **Preis pro Liter** wird verwendet um die Preise der Cocktails zu berechnen und dem Benutzer so die anfallenden Kosten zu berechnen.
- Der **Alkoholgehalt** in % wird zur Berechnung der Blutalkoholkonzentration der Benutzer verwendet.

Unten wird angezeigt wieviel Milliliter dieser Zutat in der aktuellen Session und seit Erstellung der Datenbank ausgegeben worden sind.

5.3.6 Ventilbelegung bearbeiten Fragment

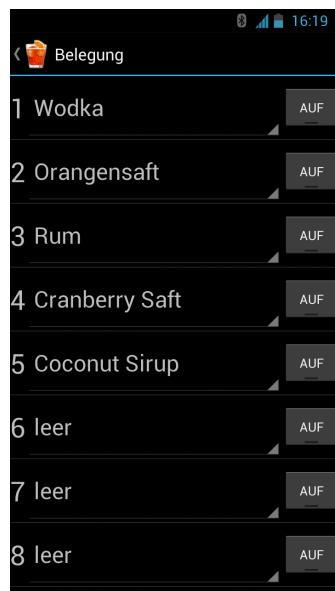


Abbildung 21: Ventilbelegung bereiten Fragment

Das **Ventilbelegung bearbeiten Fragment** (Abb. 21) dient zur Zuordnung der verschiedenen Zutaten zu den Ventilen. Im Dropdown Menü kann die am jeweiligen Ventil angeschlossene Zutat ausgewählt werden. Außerdem können die Ventile mit Hilfe der Buttons **AUF** manuell gesteuert werden.

5.3.7 Benutzer Fragment

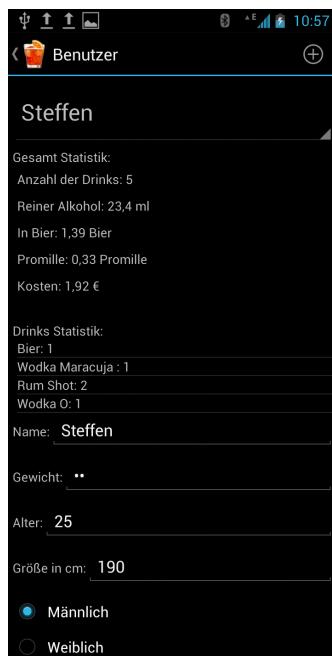


Abbildung 22: Benutzer Fragment

Im **Benutzer Fragment** (Abb. 22) können Benutzerstatistiken angezeigt, Benutzerdaten eingegeben und Benutzer neu angelegt bzw. gelöscht werden. Folgende Statistiken werden angezeigt:

- Anzahl der getrunkenen Cocktails.
- Getrunkener reiner Alkohol in ml.
- Getrunkener reiner Alkohol als Bier-Äquivalent in 0,33 Bier mit 5.1% Alkoholgehalt.
- Die Blutalkoholkonzentration in Promille wird mit Hilfe der Widmark-Formel berechnet, wobei der Verteilungsfaktor im Körper mit der Watson-Formel aus dem Geschlecht, Größe, Gewicht und Alter des Benutzers berechnet wird (siehe [Wik13]). Der Abbau von Alkohol wird momentan nicht beachtet. Die benötigten Daten können vom Benutzer in das Formular unten eingetragen werden.
- Die Gesamtkosten der vom Benutzer konsumierten Getränke.
- Die Drinks Statistik bietet einen Überblick über alle zu sich genommenen Getränke.

5.3.8 Einstellungen Fragment

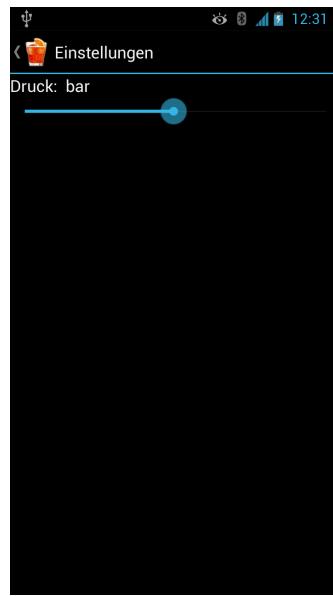


Abbildung 23: Einstellungen Fragment

Das **Einstellungen Fragment** (Abb. 23) wird für erweiterte Statistiken und Einstellungen verwendet. Der **Schieberegler** erlaubt die manuelle absolute Positionierung des Druckerschlittens. Das Feld **Druck** zeigt den aktuellen Druck des experimentell angeschlossenen Drucksensors in bar.

6 Probleme

Folgende Probleme sind bei der Durchführung des Projekts aufgetreten:

- Der mechanische Aufwand für die vertikale Achse, die ein Rührgerät auf und ab fahren lassen sollte, war zu hoch. Deshalb fehlt in diesem Projekt noch das vollautomatische Umrühren der Cocktails.
- Da die üblichen Methoden zur Bestimmung der Konstanten des PID-Reglers nicht verwendet werden konnten, musste ein eigenes Tuning-Verfahren entwickelt werden (siehe Kap. 4.5.1).
- Der verwendete Drucker-Motor scheint nicht für niedrige Geschwindigkeiten ausgelegt zu sein. Durch das Verwenden einer niedrigen PWM Frequenz (60Hz) konnten trotzdem ausreichend langsame Geschwindigkeiten bei ausreichendem Drehmoment erzielt werden.
- Der Aufbau der Interface Platine konnte aus Zeitmangel nur auf Lochrasterplatinen erfolgen. Ein Schaltplan in Eagle ist jedoch bereits vorhanden.
- Unbekannte Störungen haben zu einem zufälligen Schalten der Ventilen geführt. Diese Störungen konnten zunächst durch keine der verwendete Entstörmaßnahmen beseitigt werden.
 - Elektrolytkondensator 100 uF zwischen der 12 Volt Schiene und Masse zur Stabilisierung der Versorgungsspannung.
 - Keramikkondensatoren 100nF an der Stromversorgung der einzelnen ICs zur Stabilisierung der Logik-Versorgungsspannung.
 - 10 kOhm Pull-Down-Widerstände an den Eingängen der Schieberegister für einen definierten Zustand.
 - 10 nF Keramikkondensatoren die beide Pole des Motors mit dem Metallgehäuse des Motors verbinden zur Unterdrückung von durch den Motor ausgelösten Störungen.

Erst durch das wieder Entfernen der 10 nF Kondensatoren am Motor konnten die Störungen beseitigt werden.

7 Ergebnis

Mit diesem Projekt ist ein voll funktionsfähiger Prototyp einer Cocktailmaschine entstanden. Die Cocktailmaschine lässt sich komfortabel über die Drinkmixer Control Software bedienen. Ein Betatest des Gesamtsystems hat großes Potential gezeigt. Die Benutzerstatistiken geben interessante Einblicke in das Konsumverhalten der Benutzer, was mit dieser Genauigkeit nur in Verbindung mit einer

Cocktailmaschine möglich ist. Die verwendete Druckermechanik hat sich als eine solide mechanischen Plattform erwiesen, die viel Entwicklungsaufwand in der Mechanik eingespart hat.

8 Erweiterungsmöglichkeiten

Dieser Prototyp bietet die Basis für viele Verbesserungen und Erweiterungen:

- Aufbau der Interface-Platine als gefräste/geätzte Platine mit SMD Bauteilen, anstatt wie bisher als Lochrasterplatine.
- Bessere Ventile die größere Durchflussmengen erlauben, zuverlässiger und lebensmittelecht sind.
- Eine vertikale Achse für ein Rührgerät zum automatischen Umrühren der Cocktail kann oberhalb im Wirkungsbereich des horizontalen Schlittens angebracht werden.
- Ein Icecrusher kann ebenfalls oberhalb im Wirkungsbereich des horizontalen Schlittens angebracht werden um automatisch Eis in die Gläser einzufüllen.
- Aktive Druckerzeugung in den Flaschen mit Hilfe eines Kompressors für mehr Durchfluss.
- Die Dosierung der Zutaten könnte nicht wie bisher absolut, sondern prozentual eingegeben werden, um verschiedene Glasgrößen verwenden zu können.
- Erweiterte Statistiken mit Diagrammen und automatischen Facebookposts.
- Automatische Bezahlung über Paypal.
- Benutzeridentifikation über Gesichtserkennung oder RFID im Glas.

9 Anhang

9.1 Schaltplan Interface Platine

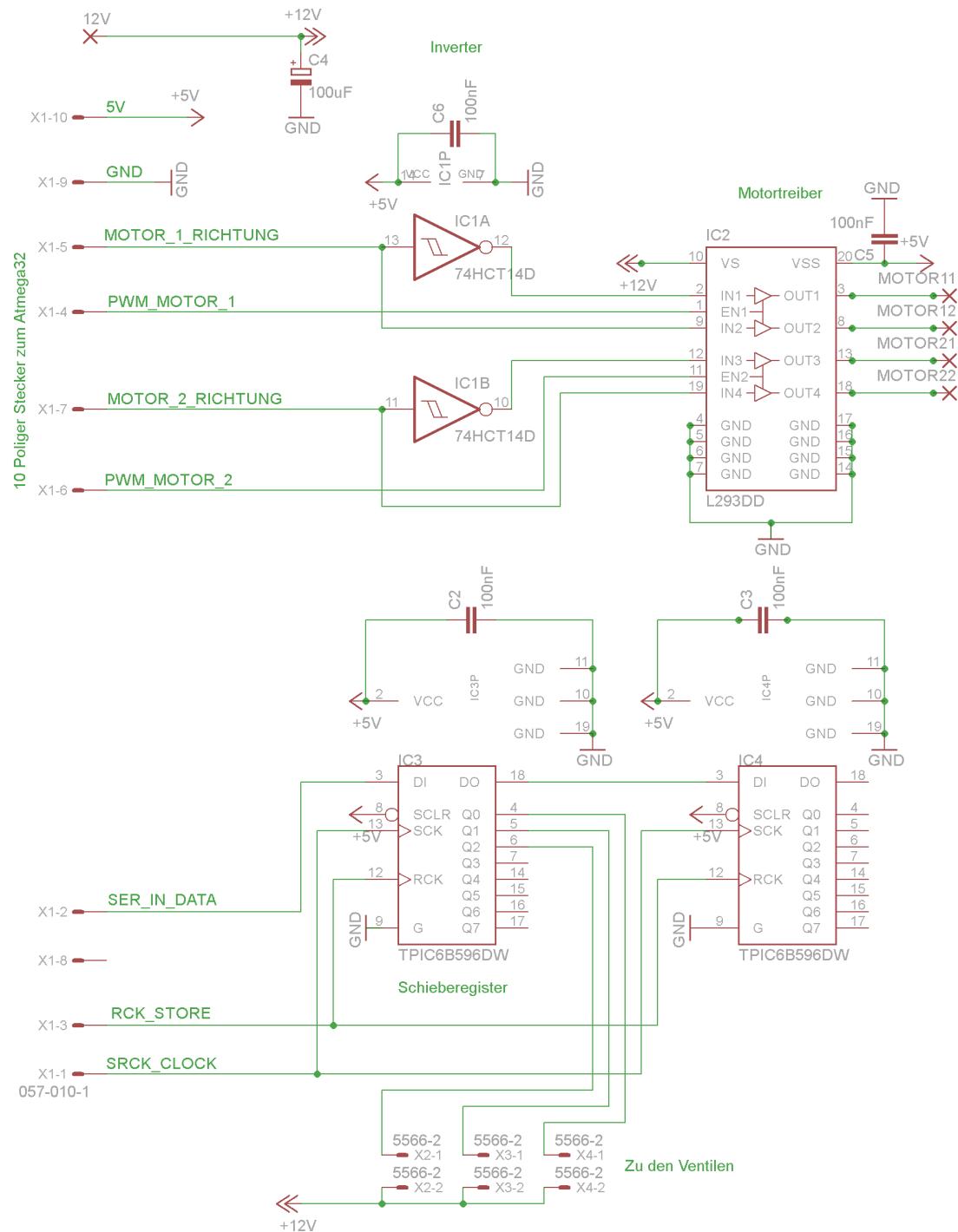


Abbildung 24: Schaltplan der Interface Platine

Literatur

- [Agi02] AGILENT TECHNOLOGIES: *Agilent HEDS-9710, HEDS-9711 200 lpi Analog Output Small Optical Encoder Modules Data Sheet*. Website, 2002. http://mckgyver.pbworks.com/f/Agilent_Optical_Encoders.pdf; Stand: 11.12.2012.
- [Atm11] ATMEL CORPORATION: *Atmega32/L 8-bit AVR Microcontroller with 32KBytes In-System Programmable Flash*. Website, 2011. <http://www.atmel.com/Images/doc2503.pdf>; Stand: 09.01.2013.
- [CoA12] COACTIONOS, INC: *Motor Control using PWM and PID*. Website, 2012. <http://www.coactionos.com/embedded-design/39-motor-control.html>; Stand: 16.12.2012.
- [Eth12a] ETHERSEX: *Ethersex Wiki - Main Page*. Website, 2012. ethersex.de; Stand: 15.12.2012.
- [Eth12b] ETHERSEX: *Quick Start Guide*. Website, 2012. http://ethersex.de/index.php/Quick_Start_Guide; Stand: 15.12.2012.
- [Goo13] GOOGLE, INC: *Get the Android SDK*. Website, 2013. <http://developer.android.com/sdk/index.html>; Stand: 09.01.2013.
- [Gor11] GORDON McCOMB: *Experiments in Quadrature Encoders*. Website, 2011. <http://www.robotoid.com/appnotes/circuits-quad-encoding.html>; Stand: 11.12.2012.
- [Mad11] MAD PENGUIN LABS: *Tutorial: Use an Old Inkjet Printer to Learn Servo Motor Control With EMC2 – Part 2*. Website, 2011. <http://madpenguin.ca/blog/2011/06/14/tutorial-use-an-old-inkjet-printer-to-learn-servo-motor-control-with-emc2>; Stand: 11.12.2012.
- [Mik12] MIKROCONTROLLER.NET: *AVR-Tutorial: Schieberegister*. Website, 2012. http://www.mikrocontroller.net/articles/AVR-Tutorial_Schieberegister; Stand: 06.12.2012.
- [Phi93] PHILIPS SEMICONDUCTORS: *74HC/HCT14 Hex inverting Schmitt trigger*. Website, 1993. <http://docs-europe.electrocomponents.com/webdocs/0037/0900766b80037121.pdf>; Stand: 16.12.2012.
- [Pol12a] POLLIN: *AVR-NET-IO - Fertigmodul*. Website, 2012. http://www.pollin.de/shop/dt/NjI50Tgx0Tk-/Bausaetze_Module/Bausaetze/AVR_NET_IO_Fertigmodul.html; Stand: 06.12.2012.

- [Pol12b] POLLIN: *Druckluft-Magnetventil SH-V0829BC-R.* Website, 2012. http://www.pollin.de/shop/dt/MzQ50TA20Tk-/Bauelemente_Bauteile/Mechanische_Bauelemente/Sonstige_E_Geraete/Druckluft_Magnetventil_SH_V0829BC_R.html; Stand: 12.12.2012.
- [Rud13] RUDIN INFORMATIK: *JEthernetControl.* Website, 2013. <https://github.com/Rudin-Informatik/ch.ri.control/tree/master/ch.ri.control.ethersex>; Stand: 09.01.2013.
- [STM12] STMICROELECTRONICS: *L293D push-pull four channel driver with diodes.* Website, 2012. <http://www.st.com/internet/analog/product/63141.jsp>; Stand: 11.12.2012.
- [Tex12] TEXAS INSTRUMENTS: *TPIC6B595.* Website, 2012. <http://docs-europe.electrocomponents.com/webdocs/0b88/0900766b80b88944.pdf>; Stand: 06.12.2012.
- [Tho12] THOMAS FISCHL: *USBasp - USB programmer for Atmel AVR controllers.* Website, 2012. <http://www.fischl.de/usbasp/>; Stand: 16.12.2012.
- [VER13] VERSANT CORP: *db4objects.* Website, 2013. <http://db4o.com>; Stand: 09.01.2013.
- [Wik12] WIKIPEDIA: *Servomotor.* Website, 2012. <http://de.wikipedia.org/wiki/Servomotor>; Stand: 16.12.2012.
- [Wik13] WIKIPEDIA: *Berechnung der BAK.* Website, 2013. http://de.wikipedia.org/wiki/Blutalkoholkonzentration#Berechnung_der_BAK; Stand: 09.01.2013.