

# PointhiBoard Dokumentation

Thomas Pointhuber

## Contents

<b>1 Einleitung</b>	<b>2</b>
1.1 Überblick der Funktionen . . . . .	2
<b>2 Technische Spezifikationen</b>	<b>2</b>
2.1 Mechanische Spezifikationen . . . . .	2
2.2 Elektrischer Spezifikationen . . . . .	3
2.2.1 Steckverbinder . . . . .	3
<b>3 Bestückung</b>	<b>5</b>
3.1 Top-Layer . . . . .	5
3.2 Bottom-Layer . . . . .	5
<b>4 Inbetriebnahme der Hardware</b>	<b>6</b>
4.1 Flashen des Steuer-PIC . . . . .	6
4.2 Testen des Step-Down-Wandler . . . . .	6
4.3 Fehlerbehebung . . . . .	6
4.4 Testen der Hardware . . . . .	6
<b>5 Inbetriebnahme der Software</b>	<b>7</b>
5.1 Aktivieren von I2C und SPI . . . . .	7
5.2 Aktivieren der I2C-Echtzeituhr . . . . .	7
5.2.1 Fehlerbehebung . . . . .	8
5.3 Aktivieren des LCD-Touch-Displays (MI0283QT-9A) . . . . .	8
<b>6 PIC-Firmware</b>	<b>9</b>
6.1 I2C . . . . .	9
6.1.1 Schreiben . . . . .	9
6.1.2 Lesen . . . . .	9

# 1 Einleitung

Der Raspberry Pi ist ein einfacher Einplatinencomputer, welcher ideal für komplexe Steuerungs-/Regelungsaufgaben im Bereich der Robotik, Hausautomation, und vielen anderen Bereichen ist.

Er zeichnet sich durch einen niedrigen Stromverbrauch in Verbindung mit einer großen Anzahl von Schnittstellen, angefangen von USB und LAN über Bussysteme wie SPI und I2C bis zu einzelnen Digitalen Ausgängen, welche mithilfe von Linux angesprochen werden können.

Die Nutzung der Low-Level Schnittstellen die an einer Stifteleiste herausgeführt sind gestaltet sich aber nicht immer trivial, auch ist die Spannungsversorgung in Robotern nicht für eine zusätzliche Last mit bis zu 1A ausgelegt. Aus diesem und anderen Gründen soll daher ein Erweiterungsboard für den Raspberry Pi entstehen welches grundlegende Aufgaben übernimmt, die für Systeme in Steuerungssystemen notwendig erscheinen.

## 1.1 Überblick der Funktionen

Die Grundlegenden Ausstattung, die die Erweiterungsplatine besitzen soll, bzw. bereits besitzt.

- Effizienter Spannungsregler der Spannungen zwischen 7V-25V auf saubere 5V regelt
- Pegelwandler für I2C, um sich mit normaler 5V-Hardware zu verbinden
- Schnittstelle für LCD-Touch-Display auf SPI-Basis, welche später auch für SPI→CAN Konverter benutzt werden kann.
- Echtzeituhr (RTC), weil der Raspberry Pi keine besitzt, diese aber für diverse Anwendungen benötigt werden
- Spannungüberwachung und Überstromschutz, um auch mit LiPo-Versorgung sicher zu arbeiten (Tiefentladeschutz)
- Digitale Ein/Ausgänge und 8 Analoge Eingänge um ohne zusätzlicher Erweiterungsboard bereits einfache Steuerungen zu realisieren.
- RS232-TTL Schnittstelle
- RN-Standard Steckverbinder mit Verriegelung, für einfache Erweiterbarkeit und Betriebssicherheit.

# 2 Technische Spezifikationen

## 2.1 Mechanische Spezifikationen

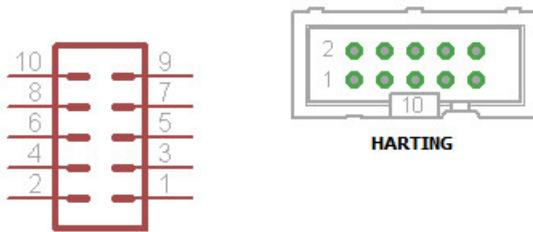
Characteristic	Min	Typ.	Max	Units
Breite	-	60	-	mm
Länge	-	110	-	mm

## 2.2 Elektrischer Spezifikationen

Characteristic	Symbol	Test Conditions	Min	Typ.	Max	Units
Versorgungsspannung	$U_{IN}$		8	12	20	V
Eingangsstrom	$I_{IN}$		-	-	8	A
Leerlaufstrom	$I_{INO}$	$U_{IN} = 12V$	-	20	-	mA
Ausgangsspannung, 5V	$U_{5V}$		4.95	5	5.05	V
Ausgangsstrom, 5V	$I_{5V}$		-	-	2.5	A
Umgebungstemperatur			-10	20	50	$^{\circ}C$

### 2.2.1 Steckverbinder

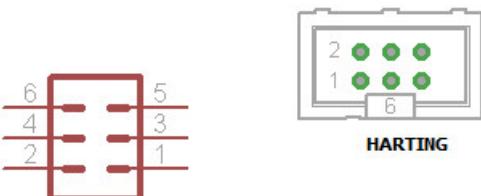
#### I2C



Der I2C-Stecker ist gemäß den RN-Definitionen<sup>1</sup> belegt.

Pin 1	SCL (Taktleitung)
Pin 3	SDA (Datenleitung)
Pin 5,7	+5V
Pin 9	Batteriespannung
Pin 2,4,6,8	GND
Pin 10	INT

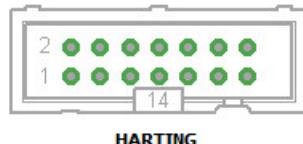
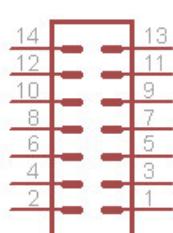
#### ICSP



Pin 1	PGD
Pin 2,4	VSS (GND)
Pin 3	PGC
Pin 5	VDD (+5V)
Pin 6	MCLRE/VPP

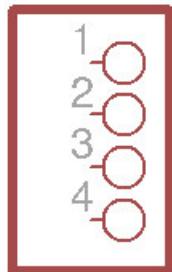
<sup>1</sup>[http://www.rn-wissen.de/index.php/RN-Definitionen#I2C-Bus\\_Stecker](http://www.rn-wissen.de/index.php/RN-Definitionen#I2C-Bus_Stecker)

## 8-Bit



Pin 1...8	PIN 0...7
Pin 9,11	+5V
Pin 10,12,14	GND
Pin 13	Batteriespannung

## RS232-TTL



**PIN HEADER**

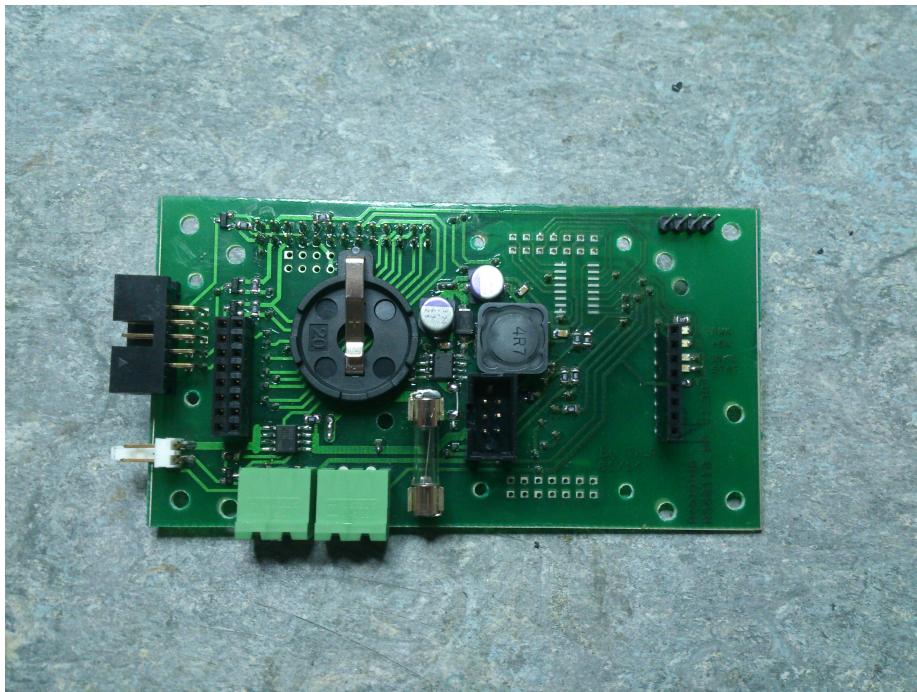
Der RS232-TTL-Stecker ist gemäß den RN-Definitionen<sup>2</sup> belegt.

Pin 1	RX
Pin 2	TX
Pin 3	GND
Pin 4	+5V

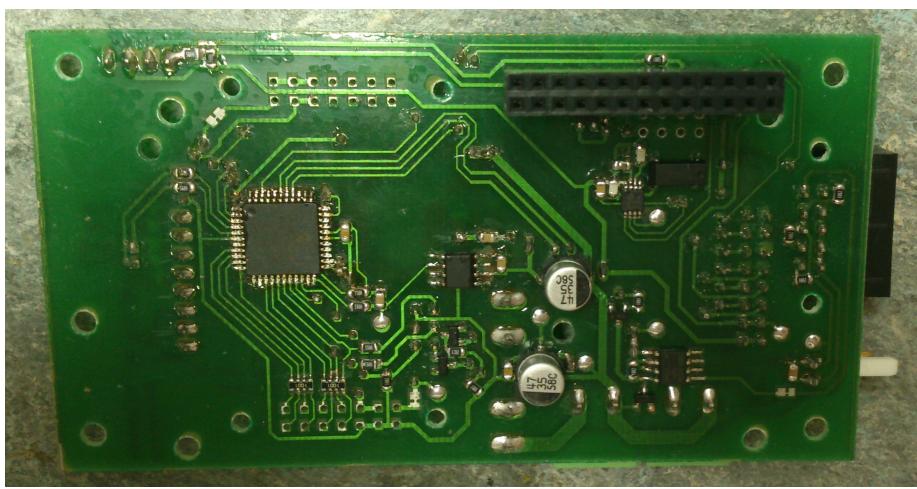
<sup>2</sup>[http://www.rn-wissen.de/index.php/RN-Definitionen#RS232\\_TTL\\_Stecker](http://www.rn-wissen.de/index.php/RN-Definitionen#RS232_TTL_Stecker)

### 3 Bestückung

#### 3.1 Top-Layer



#### 3.2 Bottom-Layer



## 4 Inbetriebnahme der Hardware

### 4.1 Flashen des Steuer-PIC

Es wird ein ICSP-Programmiergerät, welches einen PIC18F4420 flashen kann am board angeschlossen (ICSP-Stecker). Der PIC sollte korrekt erkannt werden, und kann dann mit der Firmware beschrieben werden.

### 4.2 Testen des Step-Down-Wandler

1. Es wird am Eingang des Board ein Netzteil mit folgenden Einstellungen angeschlossen:  $I_{max} = 100mA, U_a = 5V$
2. Am Spannungs-Ausgang für den Raspberry Pi wird ein Multimeter angeschlossen, welches eine Spannung von ungefähr 5V anzeigen sollte.
3. Jetzt wird die Spannung des Netzteiles langsam auf etwa 10-12V gesteigert, die Spannung am Multimeter darf dabei eine Spannung von 5,1-5,2V nicht überschreiten! Falls die Spannung höher als etwa 5,2V steigen sollte funktioniert der Step-Down-Wandler nicht ordnungsgemäß. Der Step-Down-Wandler auf der Platine muss überprüft und wenn möglich repariert werden. Eine zu hohe Spannung am 5V-Spannungskreis könnte ansonsten zu einer Beschädigung diverser Bauteile auf dieser, und anderen angeschlossenen Platinen führen.

### 4.3 Fehlerbehebung

Problem	Fehlerursache
Keine der Status-LED leuchtet auf	Board nicht korrekt angeschlossen Sicherung durchgebrannt PIC nicht/falsch geflasht
+5V, PWR-Led leuchtet nicht	Tiefentladeschutz aktiv (kurz aufleuchtende STAT-LED)
h	MOSFET schaltet nicht durc

### 4.4 Testen der Hardware

Der PIC wird mit der speziellen TEST-Firmware beschrieben<sup>3</sup>. Dann werden die beiden 14-Poligen Stiftleisten 1:1 miteinander verbunden. Sobald das Board mit Spannung versorgt wird startet die Testroutine. Solange die Testroutine aktiv ist blinken beide Status-LEDs schnell. Wenn diese dann erfolgreich war, Leuchtet STAT dauerhaft. Falls die Testroutine fängt INFO zu blinken an.

---

<sup>3</sup>siehe Flashen des Steuer-PIC

## 5 Inbetriebnahme der Software

Hier ist die Konfiguration einen Raspberry Pi der 2. Revision beschrieben. Wenn ein Raspberry Pi der 1. Revision verwendet wird muss i2c-1 durch i2c-0 ersetzt werden! Das verwendete Distribution ist das oft verwende Wheezy, bei Arch und anderen Linux Derivaten kann die Konfiguration abweichen!

### 5.1 Aktivieren von I2C und SPI

1. In der Datei /etc/modprobe.d/raspi-blacklist.conf werden die beiden Schnittstellen aktiviert, indem die beiden Zeilen wie folgt auskommentiert werden:

```
#blacklist spi-bcm2708  
#blacklist i2c-bcm2708
```

2. In der Datei /etc/modules werden dann folgende Module hinzugefügt:

```
snd-bcm2835  
i2c-bcm2708  
i2c-dev
```

### 5.2 Aktivieren der I2C-Echtzeituhr

1. Hinzufügen der folgenden Zeile in der Datei /etc/modules:

```
i2c:mcp7941x
```

2. Es werden 2. Zeilen in der Datei /etc/rc.local hinzugefügt (vor dem Befehl "exit 0"):

```
echo mcp7941x 0x6f > /sys/class/i2c-dev/i2c-1/  
device/new_device  
hwclock -s
```

3. Jetzt kann man die fake-hwclock deaktivieren

```
sudo update-rc.d fake-hwclock remove
```

4. Jetzt wird der Raspberry Pi neu gestartet, und dann die aktuelle zeit wie folgt in UTC gesetzt (natürlich das Datum durch das aktuelle ersetzen):

```
sudo date -s "9 JAN 2014 12:00:00"
```

5. Beim Herunterfahren wird dann ab sofort die Zeit in die Echtzeituhr geschrieben, und beim nächsten Starten daraus ausgelesen.

---

<http://www.100randomtasks.com/raspberry-pi-rtc>

### 5.2.1 Fehlerbehebung

Falls wie bei mir die folgende Fehlermeldung hwclock: ”ioctl( RTC\_RD\_TIME ) to /dev/rtc0 to read the time failed: Invalid argument“ auftritt, werden einfach die folgenden Befehle in die Shell der Reihe nach eingegeben:

```
hwclock -w  
hwclock -s  
hwclock -r
```

## 5.3 Aktivieren des LCD-Touch-Displays (MI0283QT-9A)

1. Download ”FBTFT drivers as loadable modules. See ’Step-by-step’ for loading drivers.” <https://github.com/notro/fbtft/wiki#image-download>
2. In der Datei /etc/modules werden dann folgende Module hinzugefügt:

```
fbtft_device  
ads7846_device
```

3. In der Datei /etc/modprobe.d/pointhiboard.conf werden dann folgende Optionen definiert:

```
options ads7846_device cs=0 speed=2000000  
model=7846 x_min=250 x_max=3780 y_min=160  
y_max=3930 pressure_max=255 x_plate_ohms=60  
gpio_pendown=25 keep_vref_on=1 swap_xy=1  
options fbtft_device cs=1 speed=16000000 fps  
=25 name=mi0283qt-9a gpios=reset:23,led:24  
rotate=90
```

4. In der Datei /etc/X11/xinit/xinitrc wird dann folgendes vor . /etc/X11/Xsession eingefügt:

```
# Touchpanel: Invert X and Y axis  
DISPLAY=:0 xinput --set-prop 'ADS7846  
Touchscreen' 'Evdev Axis Inversion' 1 1
```

5. Damit die Konsole beim booten auf dem Display dargestellt wird, muss folgendes am Ende in /boot/cmdline.txt hinzugefügt werden:

```
fbcon=map:10 fbcon=font:ProFont6x11
```

## 6 PIC-Firmware

### 6.1 I2C

#### 6.1.1 Schreiben

Das erste übertragene Byte setzt den Schreib/Lese-Index fest. Dieser wird dann nach jedem weiteren Schreib/Lesebefehl jeweils um 1 inkrementiert, solange er nicht neu gesetzt wurde.

Index	Aktion	Information
0x01	Setze TRISB	1... Eingang, 0... Ausgang
0x0F	Setze PORTB	1... High, 0... Low

#### 6.1.2 Lesen

Index	Aktion	Information
0x01	Lese TRISB	1... Eingang, 0... Ausgang
0x0F	Lese PORTB	1... High, 0... Low
0x10	ADC - 0 VOLTAGE-HIGH-BYTE	
0x11	ADC - 0 VOLTAGE-LOW-BYTE	
0x12	ADC - 1 VOLTAGE-HIGH-BYTE	
0x13	ADC - 1 VOLTAGE-LOW-BYTE	
0x14	ADC - 2 VOLTAGE-HIGH-BYTE	
0x15	ADC - 2 VOLTAGE-LOW-BYTE	
0x16	ADC - 3 VOLTAGE-HIGH-BYTE	
0x17	ADC - 3 VOLTAGE-LOW-BYTE	
0x18	ADC - 4 VOLTAGE-HIGH-BYTE	
0x19	ADC - 4 VOLTAGE-LOW-BYTE	
0x1A	ADC - 5 VOLTAGE-HIGH-BYTE	
0x1B	ADC - 5 VOLTAGE-LOW-BYTE	
0x1C	ADC - 6 VOLTAGE-HIGH-BYTE	
0x1D	ADC - 6 VOLTAGE-LOW-BYTE	
0x1E	ADC - 7 VOLTAGE-HIGH-BYTE	
0x1F	ADC - 7 VOLTAGE-LOW-BYTE	
0x20	ADC - VCC VOLTAGE-HIGH-BYTE	
0x21	ADC - VCC VOLTAGE-LOW-BYTE	
0x22	ADC - +5V VOLTAGE-HIGH-BYTE	
0x23	ADC - +5V VOLTAGE-LOW-BYTE	