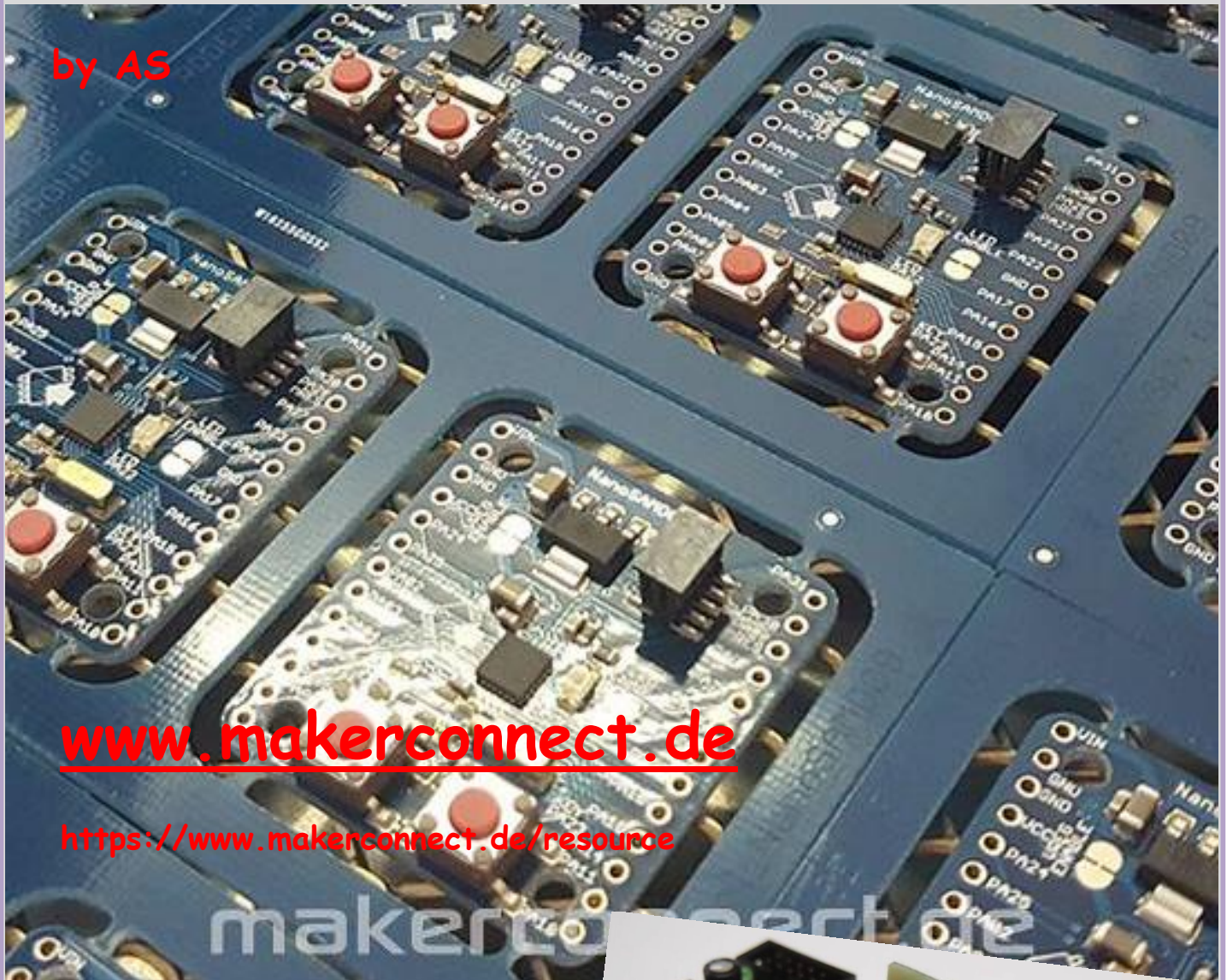


MIKROKONTROLLER & I²C BUS

by AS



www.makerconnect.de

<https://www.makerconnect.de/resource>

makerconnect.de

Temperaturmessung
mit dem I²C - Bus



I2C-Bus - Temp 1

Copyright

Sofern nicht anders angegeben, stehen die Inhalte dieser Dokumentation unter einer „Creative Commons - Namensnennung-NichtKommerziell-Weitergabe unter gleichen Bedingungen 3.0 DE Lizenz“



Sicherheitshinweise

Lesen Sie diese Gebrauchsanleitung, bevor Sie diesen Bausatz in Betrieb nehmen und bewahren Sie diese an einem für alle Benutzer jederzeit zugänglichen Platz auf. Bei Schäden, die durch Nichtbeachtung dieser Bedienungsanleitung verursacht werden, erlischt die Gewährleistung/Garantie. Für Folgeschäden übernehmen wir keine Haftung! Bei allen Geräten, die zu ihrem Betrieb eine elektrische Spannung benötigen, müssen die gültigen VDE-Vorschriften beachtet werden. Besonders relevant sind für diesen Bausatz die VDE-Richtlinien VDE 0100, VDE 0550/0551, VDE 0700, VDE 0711 und VDE 0860. Bitte beachten Sie auch nachfolgende Sicherheitshinweise:

- Nehmen Sie diesen Bausatz nur dann in Betrieb, wenn er zuvor berührungssicher in ein Gehäuse eingebaut wurde. Erst danach darf dieser an eine Spannungsversorgung angeschlossen werden.
- Lassen Sie Geräte, die mit einer Versorgungsspannung größer als 24 V- betrieben werden, nur durch eine fachkundige Person anschließen.
- In Schulen, Ausbildungseinrichtungen, Hobby- und Selbsthilfewerkstätten ist das Betreiben dieser Baugruppe durch geschultes Personal verantwortlich zu überwachen.
- In einer Umgebung in der brennbare Gase, Dämpfe oder Stäube vorhanden sind oder vorhanden sein können, darf diese Baugruppe nicht betrieben werden.
- Im Falle einer Reparatur dieser Baugruppe, dürfen nur Original-Ersatzteile verwendet werden! Die Verwendung abweichender Ersatzteile kann zu ernsthaften Sach- und Personenschäden führen. Eine Reparatur des Gerätes darf nur von fachkundigen Personen durchgeführt werden.
- Spannungsführende Teile an dieser Baugruppe dürfen nur dann berührt werden (gilt auch für Werkzeuge, Messinstrumente o.ä.), wenn sichergestellt ist, dass die Baugruppe von der Versorgungsspannung getrennt wurde und elektrische Ladungen, die in den in der Baugruppe befindlichen Bauteilen gespeichert sind, vorher entladen wurden.
- Sind Messungen bei geöffnetem Gehäuse unumgänglich, muss ein Trenntrafo zur Spannungsversorgung verwendet werden
- Spannungsführende Kabel oder Leitungen, mit denen die Baugruppe verbunden ist, müssen immer auf Isolationsfehler oder Bruchstellen kontrolliert werden. Bei einem Fehlers muss das Gerät unverzüglich ausser Betrieb genommen werden, bis die defekte Leitung ausgewechselt worden ist.
- Es ist auf die genaue Einhaltung der genannten Kenndaten der Baugruppe und der in der Baugruppe verwendeten Bauteile zu achten. Gehen diese aus der beiliegenden Beschreibung nicht hervor, so ist eine fachkundige Person hinzuzuziehen

Bestimmungsgemäße Verwendung

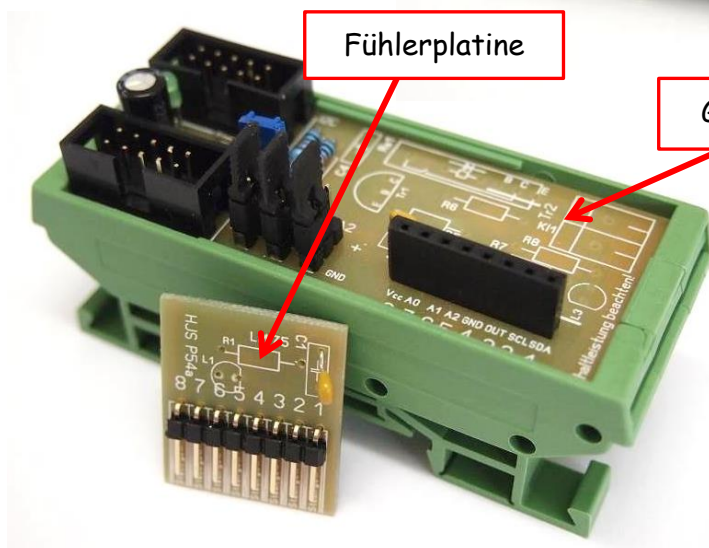
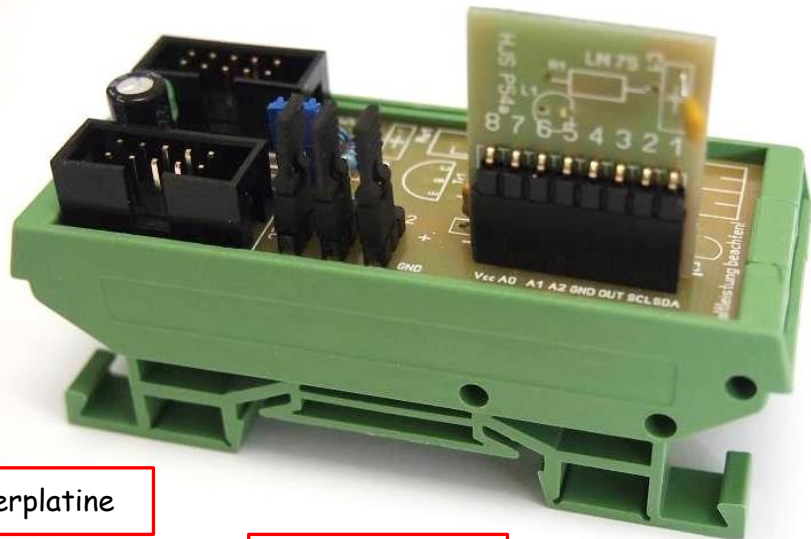
- Auf keinen Fall darf 230 V~ Netzspannung angeschlossen werden. Es besteht dann Lebensgefahr!
- Dieser Bausatz ist nur zum Einsatz unter Lern- und Laborbedingungen konzipiert worden. Er ist nicht geeignet, reale Steuerungsaufgaben jeglicher Art zu übernehmen. Ein anderer Einsatz als angegeben ist nicht zulässig!
- Der Bausatz ist nur für den Gebrauch in trockenen und sauberen Räumen bestimmt.
- Wird dieser Bausatz nicht bestimmungsgemäß eingesetzt kann er beschädigt werden, was mit Gefahren, wie z.B. Kurzschluss, Brand, elektrischer Schlag etc. verbunden ist. Der Bausatz darf nicht geändert bzw. umgebaut werden!
- Für alle Personen- und Sachschäden, die aus nicht bestimmungsgemäßer Verwendung entstehen, ist nicht der Hersteller, sondern der Betreiber verantwortlich. Bitte beachten Sie, dass Bedien- und /oder Anschlussfehler außerhalb unseres Einflussbereiches liegen. Verständlicherweise können wir für Schäden, die daraus entstehen, keinerlei Haftung übernehmen.
- Der Autor dieses Tutorials übernimmt keine Haftung für Schäden. Die Nutzung der Hard- und Software erfolgt auf eigenes Risiko.

Temperatur 1

Ein sehr beliebtes Selbstbauprojekt ist so ein Temperaturfühler. Eigentlich ist er nicht schwer und besteht nur aus wenigen Teilen.

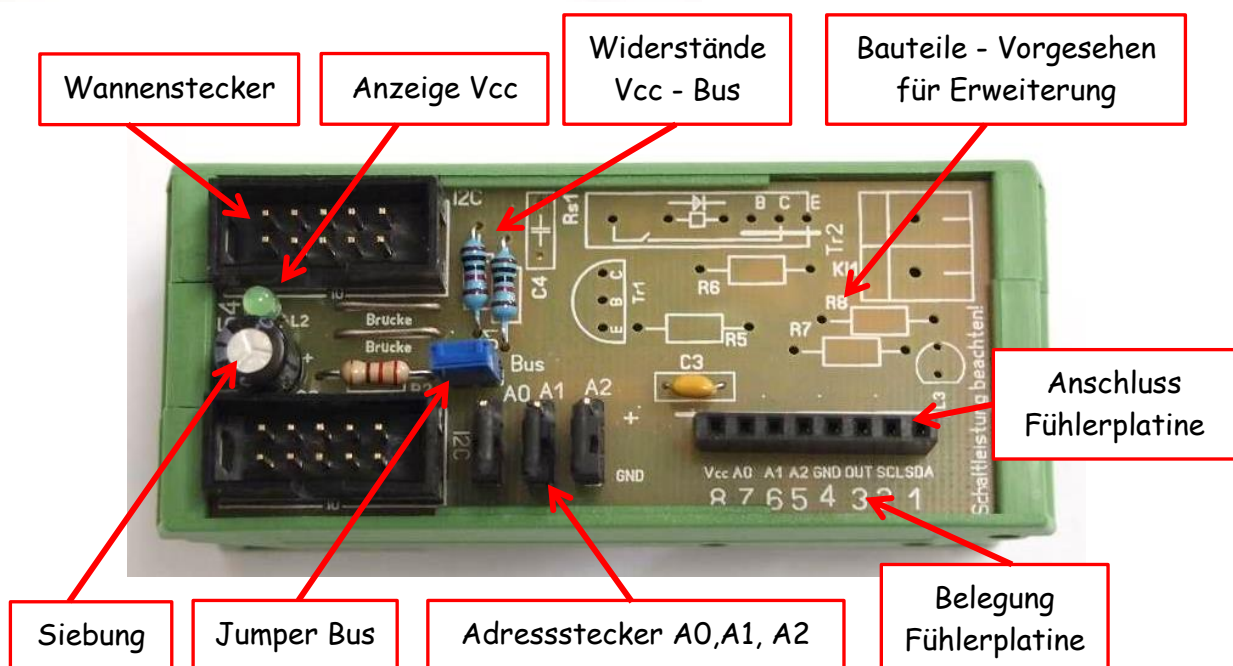
BM - Busmodul Temp 1

Bei mir muss er natürlich wieder zum System 72 passen.

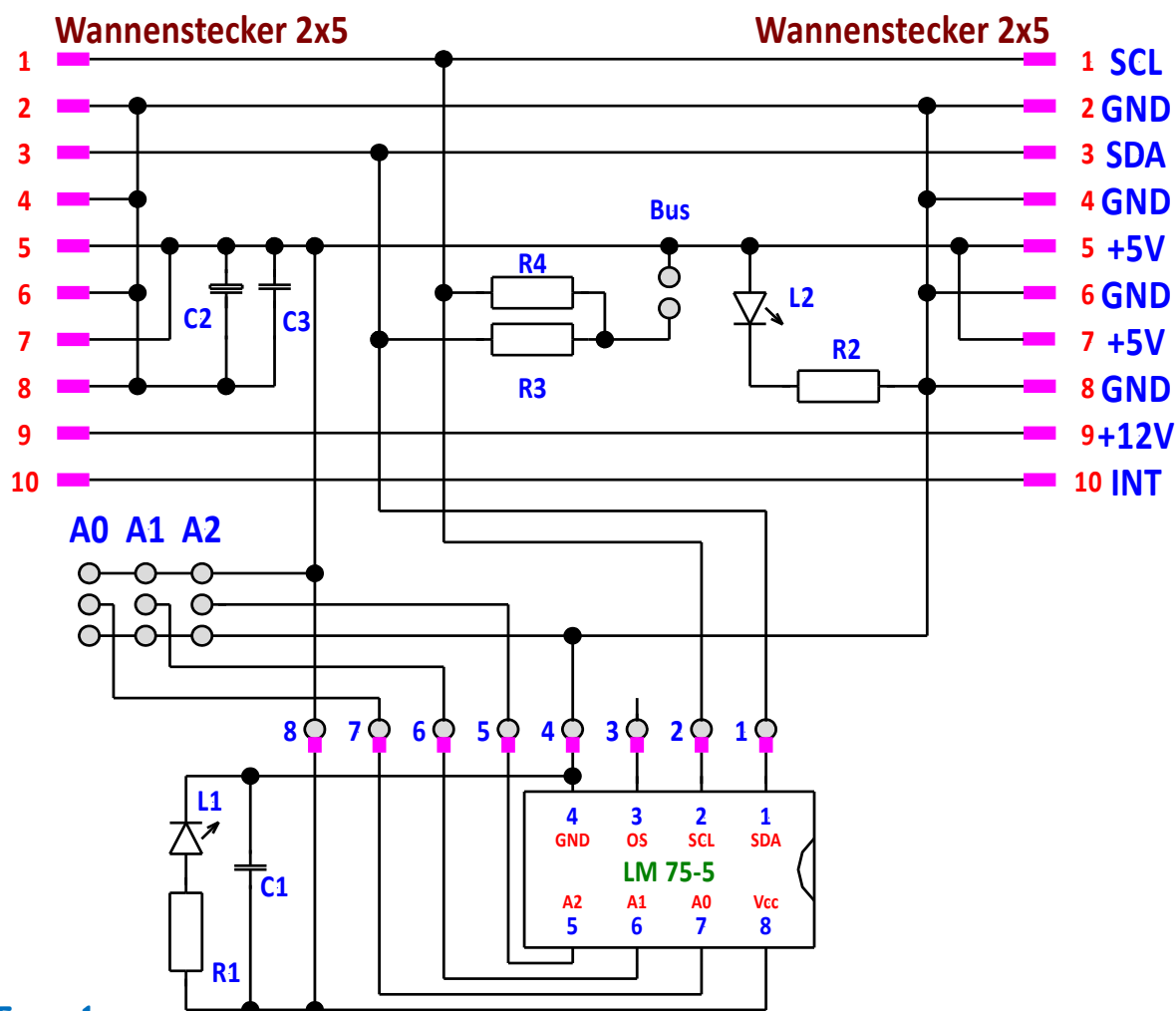


Eigentlich besteht das BM - Temp 1 aus 2 Teilen.

- Grundplatine mit I2C Bus Verbindung
- Fühlerplatine mit LM 75 - 5



Sehen wir erst einmal die Schaltung an:



BM - Temp 1

Belegung Steckverbinder Fühlerplatine:

(Reihenfolge angepasst)

8 - Vcc - +5V	7 - Adresse A0
6 - Adresse A1	5 - Adresse A2
4 - GND (Masse)	3 - frei
2 - SCL	1 - SDA

Funktionsbeschreibung:

Im oberen Teil befindet sich unser Anschluss zum I²C Bus. Zwischen den beiden Wannensteckern befindet sich die Abblockung der Betriebsspannung, die beiden Widerstände mit dem Jumper 1 um den Bus auf Vcc legen zu können und die L2 mit Vorwiderstand zur Anzeige der Betriebsspannung. An der linken Seite befinden sich die Adressstecker A0, A1 und A2. Im unteren Bereich befindet sich der Stecker mit Buchse zum Anschluss der Fühlerplatine. Auf der Fühlerplatine befindet sich der IC LM75 - 5. Daneben habe ich einen Kondensator zur Abblockung eingebaut und eine LED mit Vorwiderstand zur Anzeige der Betriebsspannung. Durch die Steckverbindung kann die Fühlerplatine auch mit einem Kabel angeschlossen werden und extra montiert werden.

Im Betrieb können sich der R1 und die LED L1 geringfügig erwärmen. Das reicht aber aus, um das Messergebnis des LM75-5 zu verfälschen.

Wenn die Anzeige nicht unbedingt gebraucht wird, kann sie einfach weggelassen werden.

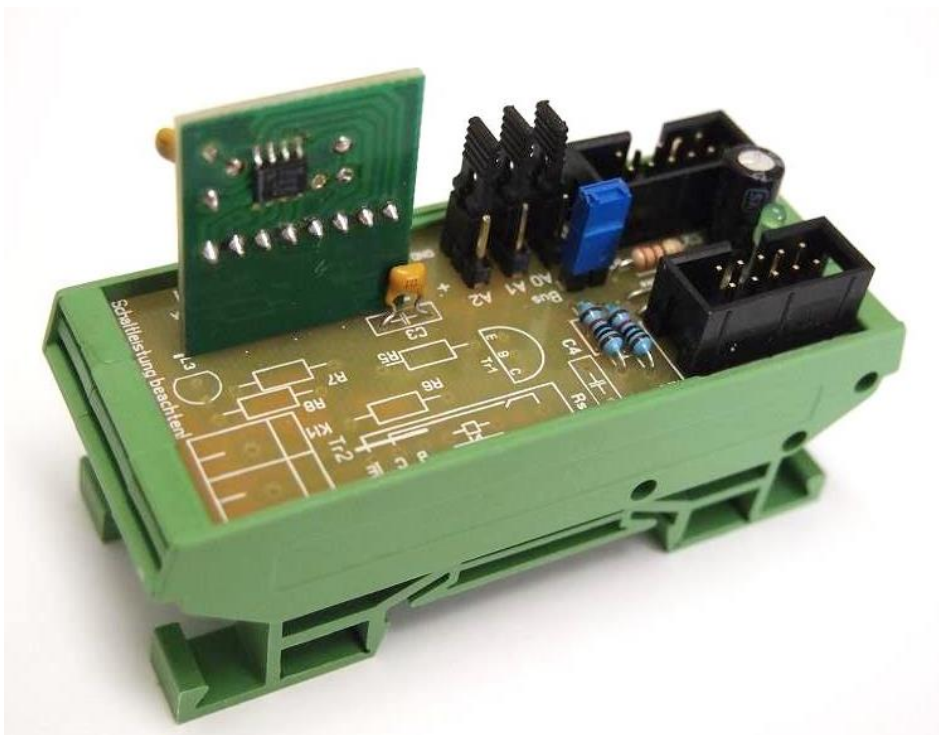
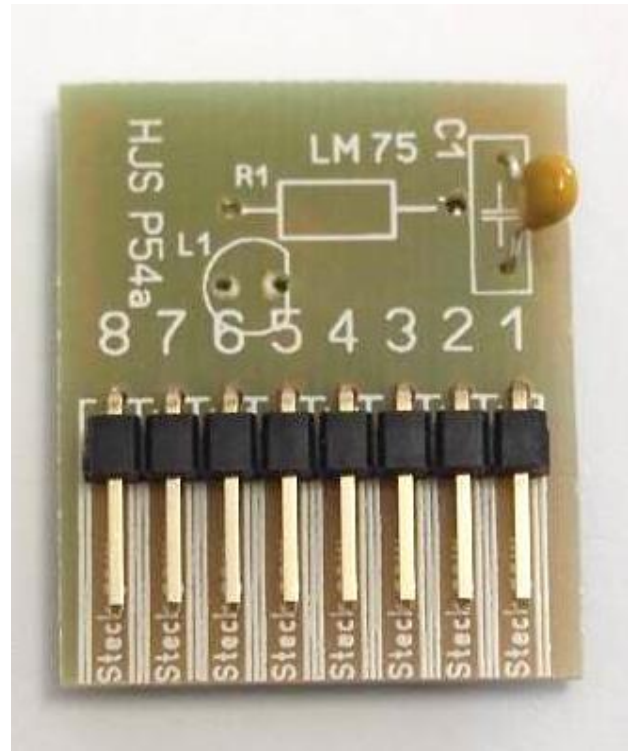
Fühlerplatine

Auf der Fühlerplatine befindet sich der Kondensator C1 und die LED L1 mit Vorwiderstand. An der Unterkante befindet sich der Steckverbinder mit den Kontakten 8 bis 1.

Die Belegung bitte der Schaltung entnehmen.

Der IC LM 75 - 5 befindet sich auf der Rückseite und ist direkt auf die Leiterzüge gelötet, da es ein SMD Typ ist.

In dieser Version habe ich die LED und den Widerstand wieder entfernt, da durch die geringe Eigenerwärmung der Messwert verfälscht wurde.



Ansicht der
Grundplatine mit
aufgesteckter
Fühlerplatine und
LM75

Auf der Rückseite der Fühlerplatine ist der LM 75-5 zu sehen. Den Widerstand und die LED habe ich wieder demontiert.

Stückliste:

2 x Wannenstecker 2 x 5 RM 2,54

4 x Jumper

R1, R2 - Widerstand 220 Ohm

C2 - Elko 100/16

L1, L2 - LED 3 mm grün 20 mA

3 x Stecker 3 polig

Platine (Grund) ca. 72 x 31 mm

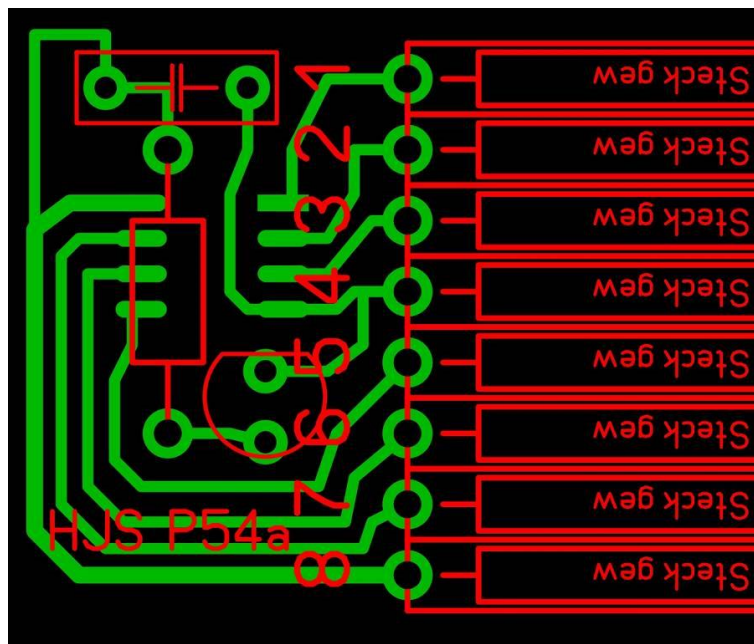
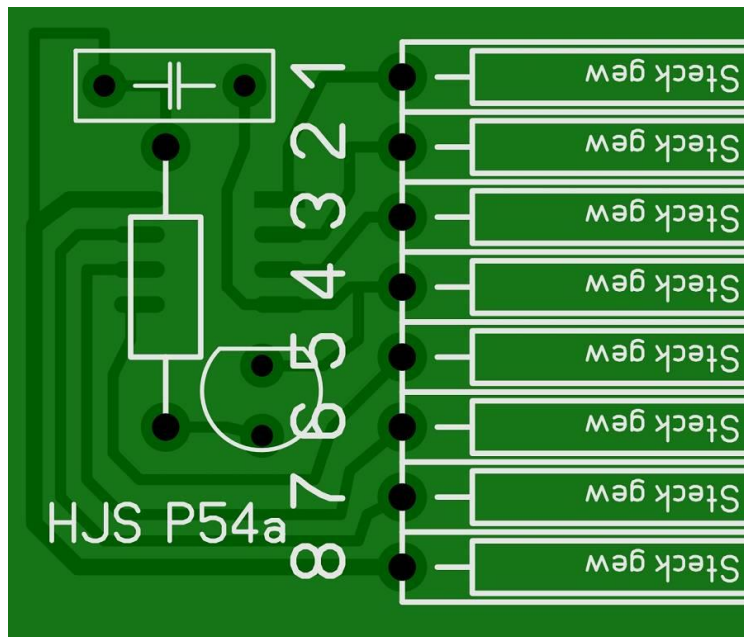
Platine (Fühler) ca. 23 x 27 mm

R3, R4 - Widerstand 10 kOhm

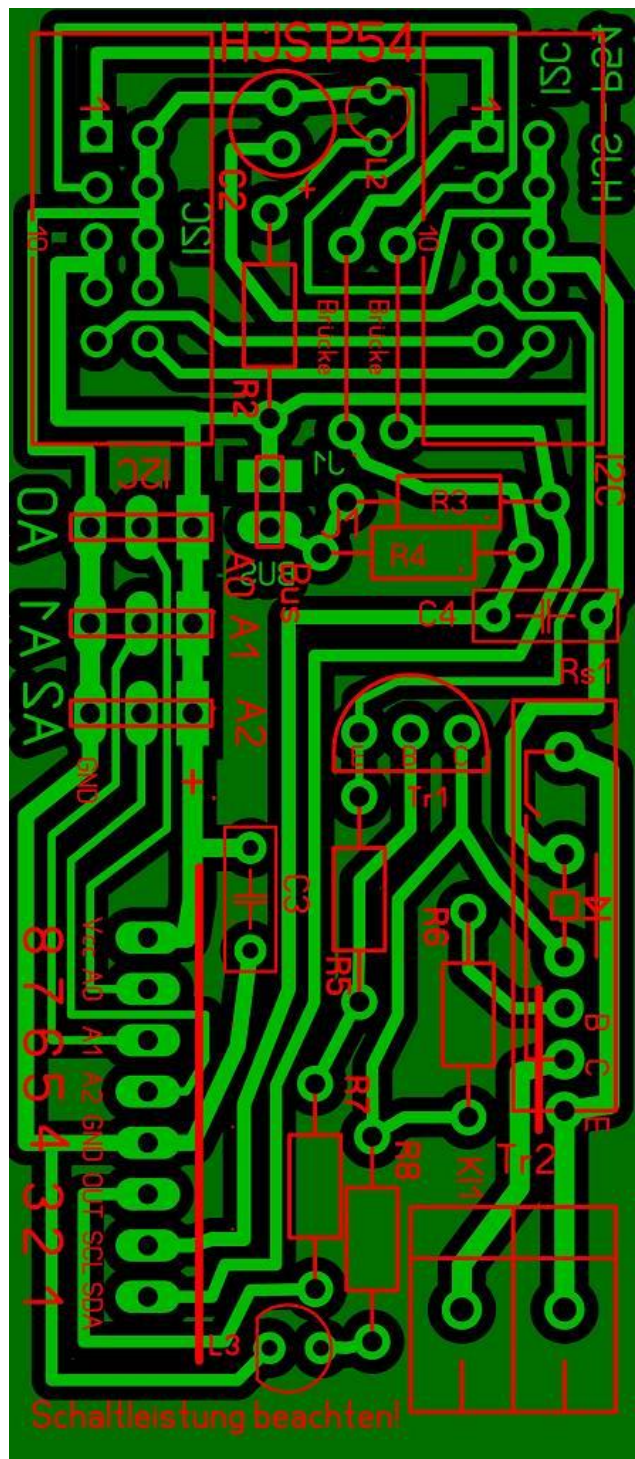
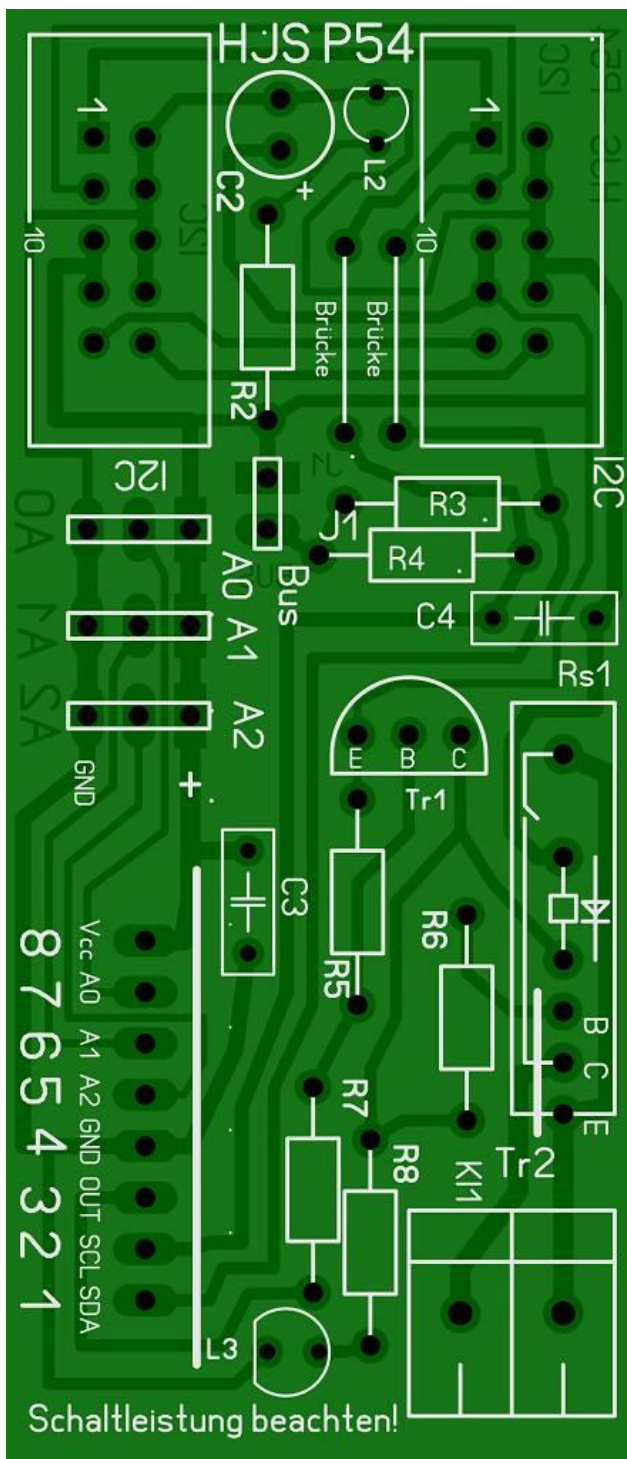
C1, C3 - Kondensator 100nF

IC1 - LM 75 - 5 (5V) **SMD !!!**

1 x Stecker 2 polig



Ansicht der Fühlerplatine - Fotoansicht mit Bestückungsdruck und in der Durchsicht



Ansicht der Grundplatine - Fotoansicht mit Bestückungsdruck und in der Durchsicht

Ein Teil der Platine ist zur Bestückung vorgesehen, aber noch nicht ausgeführt.

Für eine erste Messung habe ich wieder ein Programm geschrieben.

```
/* ATB_B1_Temp_1.c Created: 19.10.2014 20:02:09 Author: AS */
```

```
#include <stdbool.h>
#include <avr/pgmspace.h>
#include "main.h"
#include <util/delay.h>
#include "i2clcd.h"
#include "i2cmaster.h"
#include "avr/io.h"
#include "util/delay.h"
#include "avr/interrupt.h"

#define lm75_r 0x91 // Leseadresse des LM75

uint8_t msb_temp; // Oberes Temperatur-Byte
uint8_t lsb_temp; // Unteres Temperatur-Byte
uint16_t temp_wrd; // Ganzes Temperatur-Wort
uint8_t ret; // Kontrollvariable für I2C Kommunik.
uint8_t x; // X-Position der Kommastelle
signed char temperatur; // Variable m. Vorzeichen für die Temp.berechnung
char Buffer[20]; // Umwandlungs-Variable für LCD Anzeige

void temperfassung(void) // Unterprogramm Temperaturerfassung
{
    ret = i2c_start(lm75_r); // Start Lesen des LM75
    if (ret == 0)
    {
        // Wenn LM75 ein OK sendet...
        msb_temp = i2c_readAck(); // ...speichere oberes Bit
        lsb_temp = i2c_readNak(); // ...speichere unteres Bit
    }
    else // Fehlererkennung
    {
        // Wenn LM75 kein OK sendet
        lcd_command(LCD_CLEAR); // Leere Display
        lcd_printlc(1,13,"READ"); // "Lesevorgang"
        lcd_printlc(2,13,"NOK"); // "Nicht OK (NOK)"
    }
}

int main(void) // Start Hauptprogramm
{
    cli(); // Interrupts deaktiviert
    i2c_init(); // Starte I2C Bus
    lcd_init(); // Starte I2CLCD

    // Display Befehle - Display ein, Cursor aus, Blinken aus
    lcd_command(LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKINGOFF);
    lcd_command(LCD_CLEAR); // Leere Display
    _delay_ms(2); // Warte 2ms
```



```

lcd_printlc(1,6,"Boxtec");           // Zeile 1
lcd_printlc(2,2,"Display Modul 1"); // Zeile 2
lcd_printlc(3,2,"und Temp-Modul");  // Zeile 3
lcd_printlc(4,6,"(achim S.)");      // Zeile 4
_delay_ms(5000);                     // Warte 5000ms
lcd_command(LCD_CLEAR);              // Lösche Display
_delay_ms(2);                        // Warte 2ms
while(1)                             // Start while Schleife
{
    temperfassung();                 // Unterprg Temp.auslesung
    temp_wrd = (msb_temp << 8 | lsb_temp); // Zusammensetzung von o. & u. Byte
    temperatur = msb_temp;            // Umwandlung in ein Signed Char

    // Nachkommastelle erkennen
    if (temperatur <= -100) {x = 5;} // X-Pos. bei <= -100°C
    else if (temperatur <= -10) {x = 4;} // X-Pos. bei <= -10°C
    else if (temperatur <= 9) {x = 3;} // X-Pos. bei <= 9°C
    else if (temperatur <= 99) {x = 4;} // X-Pos. bei <= 99°C
    else {x = 5;} // X-Pos. bei > 99°C

    if (temp_wrd & (1<<7))
    {
        lcd_printlc(3,x,".5");      // Nachkommastelle erkennen
        // ,5 an Pos x schreiben
    }
    else
    {
        lcd_printlc(3,x,".0");      // ,0 an Pos x schreiben
    }
    if (temp_wrd & (1<<15))
    {
        // Wenn Minus-Grade angezeigt werden
        if (temp_wrd & (1<<7))
        {
            // Nachkommastelle erkennen
            temperatur = temperatur +1;
        }
        itoa(temperatur,Buffer, 10 ); // Variable umwandeln...
        lcd_printlc(3,1,Buffer);      // ...und anzeigen
    }
    else
    {
        lcd_printlc(3,1,"+");        // Ausgabe „ + „
        itoa(temperatur, Buffer, 10 );
        lcd_printlc(3,2,Buffer);
    }
    lcd_printlc(1,1,"Temperatur ist:"); // Ausgabe Text
    _delay_ms(100);                  // Pause
}
}

```

Noch ein paar Erklärungen zum Programm:

```
#define lm75_r 0x91 // Leseadresse des LM75
```

Angabe der Adresse des LM75. Der Bereich geht von 0x90 bis 0x9F und wird mit den Adressjumpers A0, A1 und A2 eingestellt. Es sind 8 Adressen möglich.

```
void temperfassung(void) // Unterprogramm Temperaturerfassung
{
    ret = i2c_start(lm75_r); // Start Lesen des LM75
    if (ret == 0)
```

Das Unterprogramm startet „**temperfassung**“ startet die I2C Kommunikation, in dem die Leseadresse des LM75 an den Bus gesendet wird. Erkennt der LM75 seine Adresse, wird eine 0 zurückgesendet und die Kommunikation beginnt

```
    else // Fehlererkennung
    {
        lcd_command(LCD_CLEAR); // Wenn LM75 kein OK sendet
        lcd_printlc(1,13,"READ"); // Leere Display
        lcd_printlc(2,13,"NOK"); // "Lesevorgang"
        lcd_printlc(2,13,"NOK"); // "Nicht OK (NOK)"
    }
```

Gibt der LM75 eine 1 zurück (keine Verbindung) wird der Text auf dem Display ausgegeben.

```
    msb_temp = i2c_readAck(); // ...speichere oberes Bit
    lsb_temp = i2c_readNak(); // ...speichere unteres Bit
```

Der LM75 ist ein Temperatursensor, der digitale Werte mit 16 Bit ausgibt. Der erste Lesebefehl liest das obere Byte aus, welches in der Variablen **msb_temp** abgespeichert wird. Der zweite Lesebefehl liest das untere Byte aus, speichert es in der Variablen **lsb_temp** ab und zeigt an, dass die Kommunikation nach ihm beendet ist.

```
int main(void) // Start Hauptprogramm
{
    cli(); // Interrupts deaktiviert
    i2c_init(); // Starte I2C Bus
    lcd_init(); // Starte I2CLCD

    // Display Befehle - Display ein, Cursor aus, Blinken aus
    lcd_command(LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKINGOFF);
    lcd_command(LCD_CLEAR); // Leere Display
    _delay_ms(2); // Warte 2ms
```

Start des Hauptprogrammes, I²C und LCD initiieren, LCD einstellen, LCD löschen und warte

```
    lcd_printlc(1,6,"Boxtec"); // Zeile 1
    lcd_printlc(2,2,"Display Modul 1"); // Zeile 2
    lcd_printlc(3,2,"und Temp-Modul"); // Zeile 3
    lcd_printlc(4,2,"(achim S.+TG)"); // Zeile 4
    _delay_ms(5000); // Warte 5000ms
    lcd_command(LCD_CLEAR); // Lösche Display
    _delay_ms(2); // Warte 2ms
```

Ausgabe Text in den Zeilen 1 bis 4, warte 5 Sekunden, lösche das Display, warte 2ms

```
temperfassung();           // Unterprg Temp.auslesung
temp_wrd = (msb_temp << 8 | lsb_temp); // Zusammensetzung von o. & u. Byte
temperatur = msb_temp;     // Umwandlung in ein Signed Char
```

Die Temperatur beim Beginn der Hauptschleife durch den Funktionsaufruf `temperfassung()` ausgelesen. Das obere und untere Byte wird anschließend zu einem 16-Bit-Wort zusammengesetzt. Da die eigentliche Temperatur im oberen Byte steht, schreibe ich nur dieses in das signed char `temperatur`, so dass es ein Vorzeichen "erhält".

```
if (temperatur <= -100) {x = 5;} // X-Pos. bei <= -100°C
else if (temperatur <= -10) {x = 4;} // X-Pos. bei <= -10°C
else if (temperatur <= 9) {x = 3;} // X-Pos. bei <= 9°C
else if (temperatur <= 99) {x = 4;} // X-Pos. bei <= 99°C
else {x = 5;} // X-Pos. bei > 99°C
```

So umgewandelt kann ich die Variable dazu nutzen die Anzahl der Zeichen zu erkennen. Das ist wichtig, um herauszufinden, an welcher Position die Kommastellenangabe platziert werden muss. Bei $\leq -100^{\circ}\text{C}$ werden bereits die ersten vier Zeichen besetzt und ein z.B. ",0" darf erst ab Stelle 5 beginnen. Steigt die Temperatur auf 5°C würden sich drei Leerstellen zwischen 5 und ",0" befinden. Das sieht nicht schön aus! Daher wird `temperatur` ausgelesen und die X-Position der folgenden Kommastelle entsprechend in der Variable `x` abgelegt.

```
if (temp_wrd & (1<<7))
{
    lcd_printlc(3,x,".5"); // Nachkommastelle erkennen
                           // ,5 an Pos x schreiben
}
else
{
    lcd_printlc(3,x,".0"); // ,0 an Pos x schreiben
}
```

Die folgende Abfrage schaut auf Bit 7 um zu erkennen, ob dort eine 1 steht. Ist dies der Fall, dann wird an die zuvor ermittelte X-Position ",5" geschrieben. Ansonsten ",0".

```
if (temp_wrd & (1<<15))
{
    // Wenn Minus-Grade angezeigt werden
    if (temp_wrd & (1<<7))
    {
        // Nachkommastelle erkennen
        temperatur = temperatur +1;
    }
    itoa(temperatur,Buffer, 10 ); // Variable umwandeln...
    lcd_printlc(3,1,Buffer);     // ...und anzeigen
}
```

Wieder wird ein Bit der `temp_wrd` Variable abgefragt um zu erkennen, ob es sich um einen negativen Wert handelt. Bei einer 1 ist dies der Fall und die Variable `temperatur` wird für die Anzeige auf dem LCD-Display umgewandelt und dann angezeigt. Vorher jedoch wird die Temperatur, wie weiter oben beschrieben, um eins erhöht!


```

else
{
  lcd_printlc(3,1,"+");           // Ausgabe „ + „
  itoa(temperatur, Buffer, 10);
  lcd_printlc(3,2,Buffer);
}

```

Handelt es sich um eine positive Zahl sind ein paar mehr Schritte notwendig. Leider beginnt die positive Zahl nicht mit einem "+" (Negative haben das "-" automatisch davor stehen), sodass es manuell gesetzt wird. Anschließend wird auch hier die Temperatur umgewandelt und am LCD-Display ausgegeben. Der Unterschied zur vorherigen "Anzeigeroutine" ist, dass die X-Position bei positiven Werten nicht bei 1, sondern bei 2 liegt. Denn bei positiven Zahlen wird ja das Vorzeichen nicht ausgegeben, sodass diese hinter dem zuvor gesetzten "+" beginnen müssen.

```

lcd_printlc(1,1,"Temperatur ist:"); // Ausgabe Text
_delay_ms(100);                     // Pause

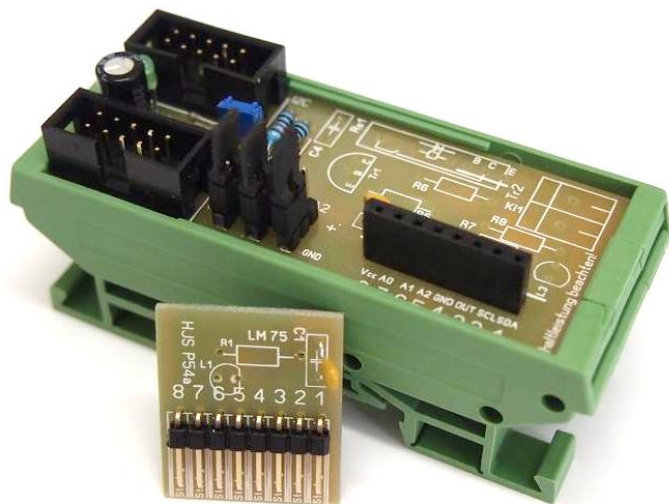
```

Die Temperaturwerte habe ich in die dritte Zeile geschrieben, sodass in der Ersten Platz für eine kleine "Überschrift" bleibt. Abschließend wird gewartet um dann von vorne zu beginnen.

So könnte der Aufbau aussehen.

Die Fühlerplatine wird in die Grundplatine gesteckt.
Bitte die Ausrichtung beachten.

Mit dem Beispielprogramm erfolgen die Messung der Temperatur am LM75, die Übertragung und die Anzeige der gemessenen Temperatur auf dem Display.



Einige Teile des Textes wurden zur besseren Übersicht farblich gestaltet.
Die Nutzung erfolgt auf eigenes Risiko.

Ich wünsche viel Spaß beim Bauen und programmieren
Achim

myroboter@web.de