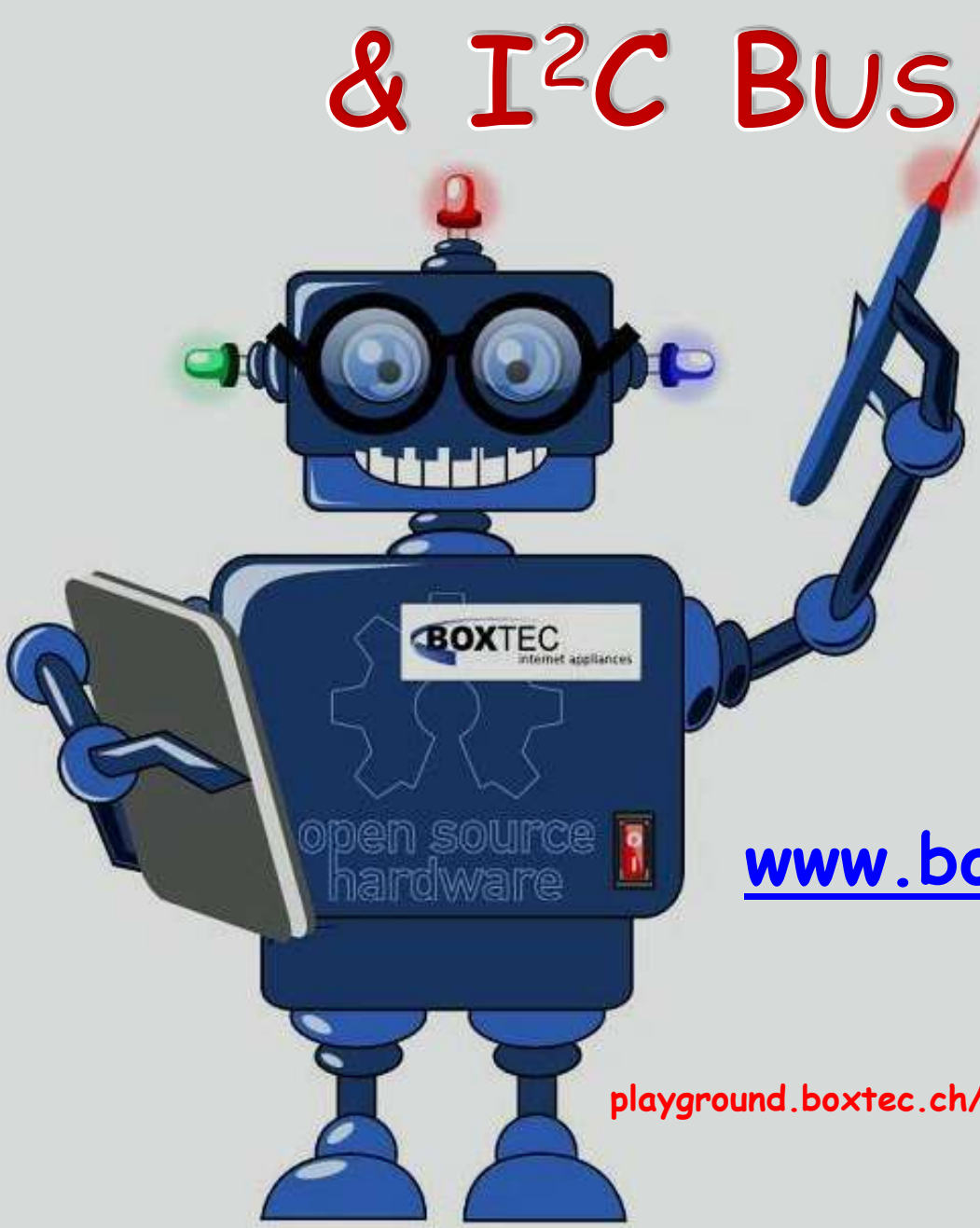
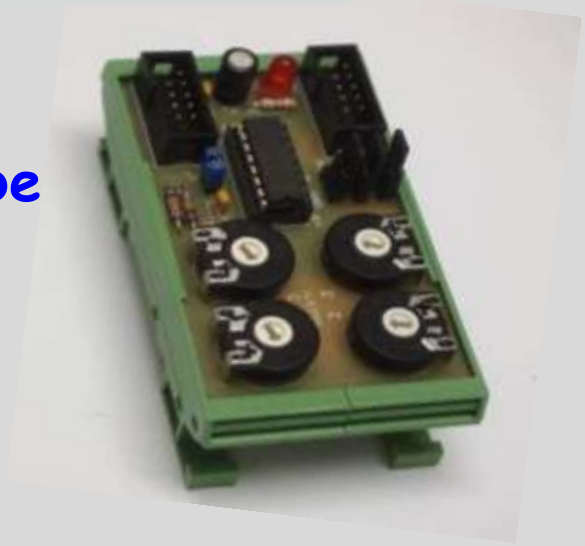
Mikrokontroller & I²C Bus



www.boxtec.ch

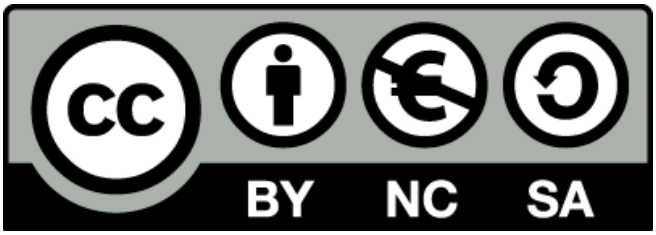
playground.boxtec.ch/doku.php/tutorial



I²C Bus und analoge Eingabe
= Teil 1 – Software =

Analog 1

Copyright



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 Nichtbe-achtung 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 die-ser 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 eine 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 Versor-gungsspannung 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 Spannungsver-sorgung 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 un-verzüglich 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 Baugrup-pe 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 Ge-fahren, 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.

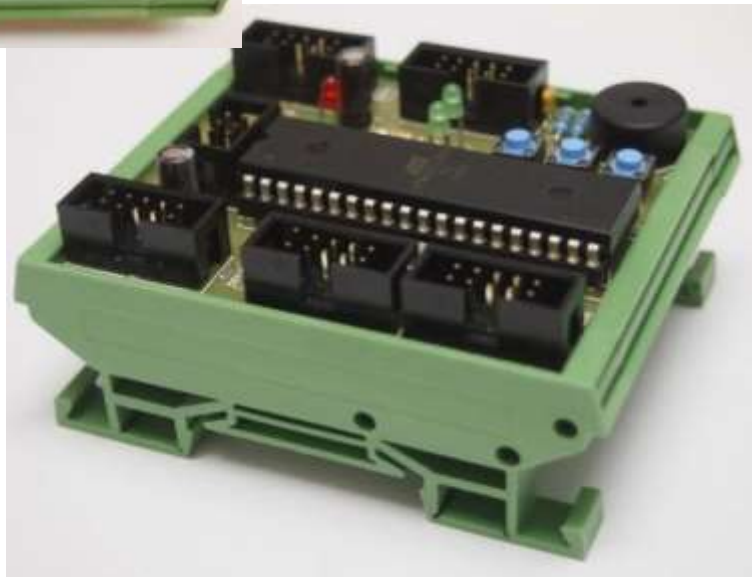
I²C Bus und analoge Eingabe
= Teil 1 – Software =

Ein „Klassiker" beim I²C Bus ist der PCF 8591. Mit diesem IC lassen sich bis zu 4 analoge Spannungen messen und eine analoge Spannung ausgeben.

Platine mit dem PCF 8591 und
4 x Trimmer (Einstellregler)

Wir benutzen wieder diese Platine um 4 analoge Spannungen einzustellen und über den I²C Bus auszugeben.
Als Anzeige nutze ich ein Display mit 4 x 16 Stellen, das ich ebenfalls im I²C Bus System betreibe.

Platine D2 (Display) mit 4 x 16
zur Anzeige im I²C Bus



Board 1 mit dem ATmega 1284p

Die Verbindung dieser 3 Module erfolgt mit dem I²C Bus

Als Stromversorgung benutze ich das NT2.

Trimmer P1

Adressen A0, A1 und A2

Trimmer P2

Trimmer P3

Trimmer P4

Zuordnung der 4 Trimmer (Einstellregler) zu den 4 analogen Eingängen und Anschluss der Referenzspannung

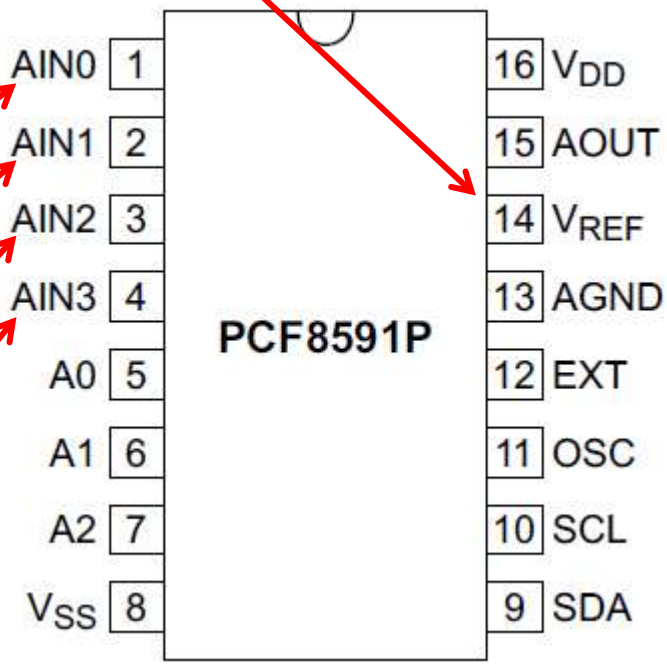
Referenzspannung - Vcc

Analog 0 – P1 – AIN0

Analog 1 – P2 – AIN1

Analog 2 – P3 – AIN2

Analog 3 – P4 – AIN3



Ansicht und PIN Belegung des PCF 8591

In den beiden oberen Bildern seht ihr die Zuordnung der 4 Trimmer zu den 4 analogen Eingängen des PCF 8591 und den Anschluss der Referenzspannung. Die Referenzspannung ist der Bezugswert zur Messspannung.
In dieser Schaltung nutze ich die Vcc (+5V) als Referenzspannung. Die komplette Schaltung habe ich bereits im Teil 1 dargestellt. Dort könnt ihr auch die genauen Verbindungen und Anschlüsse entnehmen.

Table 5. I²C slave address byte

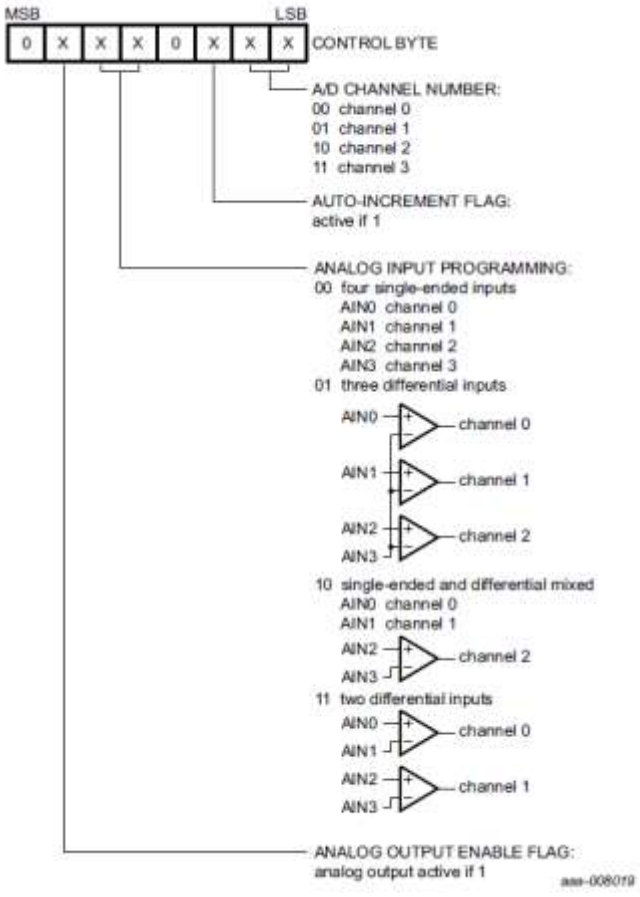
Bit	Slave address							
	7 MSB	6	5	4	3	2	1	0 LSB
slave address	1	0	0	1	A2	A1	A0	R/$\overline{W}$

Ausschnitt aus dem Datenblatt

Mit den Steckern/Jumpern A0, A1 und A2 kann ich die Adressen einstellen. Dazu muss ich entweder die Verbindung zu Vcc oder GND stecken. Damit ergeben sich die folgenden möglichen Adressen:

1001 00 00 – 144 – 0 x 90 (Bitte Schreib- und Leseadresse berücksichtigen)
1001 00 10 – 146 – 0 x 92
1001 01 00 – 148 – 0 x 94
1001 01 10 – 144 – 0 x 96
1001 10 00 – 144 – 0 x 98
1001 10 10 – 144 – 0 x 9A
1001 11 00 – 144 – 0 x 9C
1001 11 10 – 144 – 0 x 9E

Im nächsten Bild (Ausschnitt aus dem Datenblatt) wird das „Controlbyt" dargestellt. Mit diesem „Controlbyt" erfolgt die Einstellung der „Betriebsart" des PCF 8591. Das Controlbyt verfügt über 8 Bit: Beginnen wir mit dem LSB (rechts)



- Bit **0** und **1** – Auswahl der Kanal Nummer, die Werte bitte der Tabelle entnehmen
- Bit **2** – Wandlung A/D einschalten
  0 – ausschalten, **1** – einschalten
- Bit **3** – Einstellung **0** (Vorgabe)
- Bit **4** und **5** – Auswahl der Eingangskanäle. Die Zuordnung der einzelnen Kanäle bitte der Tabellen entnehmen
- Bit **6** – 1 – Wandlung AD einschalten
  0 – ausschalten, **1** – einschalten
- Bit **7** – Einstellung **0** (Vorgabe)

Beispiel

0100 0100 → 0 x 44 → Ausgangsspannung ein, Betriebsart Eingangsspannung auf 00, Kanal Nr. auf 00

0000 0100 → **0 x 04** → Ausgangsspannung aus, Betriebsart Eingangsspannung auf 00, Kanal Nr. auf 00

Im Programm nutze ich die Einstellung **0x04**.

Sehen wir uns als nächste das Programm in kleinen Schritten an:

```
#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 "avr/interrupt.h"
#include <stdlib.h>
```

Es werden die einzelnen Dateien/Bibliotheken geladen. In „main.h" gebe ich die Quarzfrequenz und die Adresse der Anzeige D2 an.

```
#define PCF8591w   0x90                    // Adressen PCF 8591
#define PCF8591r   0x91
```

Angabe der Bus Adressen für den PCF 8591

```
int8_t i;
float results[5],adc1 ,adc2 ,adc3 ,adc4;
```

Angabe der notwendigen Variablen

```
void startanzeige()                        // Titelbild
 {
   lcd_command(LCD_CLEAR);
   _delay_ms(2);
   lcd_printlc(1,2,"Analog 1 - I2C");
   lcd_printlc(2,1,"Modul P58 - 4xP");
   lcd_printlc(3,2,"AT1284 und LCD");
   lcd_printlc(4,4,"bei achim S");
   _delay_ms(5000);
 }
```

Startbildschirm mit kurzer Angabe der Anwendung

```
int main(void)
 {
   char Buffer[20];
   i2c_init();                             // Starte I2C Bus
   lcd_init();                             // Starte I2C LCD
   lcd_light(0);                           // 0=Licht an, 1=Licht aus
   // Display Befehle
   lcd_command(LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKINGOFF);
   // Display ON/OFF / Cursor ON/OFF / Blinken ON/OFF
   startanzeige();
   lcd_command(LCD_CLEAR);                 // Leere Display
   _delay_ms(2);
```

Start des Programmes, Buffer definieren, Bus und LCD initiieren, LCD-Licht an, LCD Befehle senden, Aufruf Startbildschirm, Display löschen und 2ms Pause

```
while(1)
   {
     i2c_start(PCF8591w);
     i2c_write(0x04);                    // Analog out enable, auto increment, channel 0
     i2c_stop();                         // Analogausgang aktivieren, Auto-Inkrement, Kanal 0
     i2c_start(PCF8591r);
     i2c_readAck();                      // first data is old (ersten Daten alt)
     for (i=0; i<4; i++)
      {
        results[i]=i2c_readAck();
      }
          i2c_readNak();
          i2c_stop();
          _delay_ms(2);
```

Mit diesem Teil starten wir die Verbindung zum PCF8591 (i2c_start(PCF8591w);- nur schreiben), senden die Betriebsart (i2c_write(0x04);), stoppen den Bus (i2c_stop();), starten den Bus erneut (i2c_start(PCF8591r);- nur lesen) und lesen die Werte (i2c_readAck();) für die 4 Wandler und speichern sie in den (results[i]=i2c_readAck();), stoppen den Bus und warten 2ms. **i** hat den Bereich von **0** bis **3**.

```
itoa( results[0], Buffer, 10 );                    // Zeile 1
lcd_printlc(1,1,"P1 ");
lcd_printlc(1,4,Buffer);
```

results[0] wird durch itoa gewandelt (itoa(results[0], Buffer, 10);) und an Position (lcd_printlc(1,4,Buffer);) ausgegeben. An der Position (lcd_printlc(1,1,"P1 ");) wird P1 ausgegeben
Das wiederholen wir für die nächsten 3 Zeilen

```
lcd_printlc(1,8,"U1 ");          → für die Zeilen 1 bis 4
lcd_printlc(1,16,"V");
lcd_printlc(1,12,".");
```

Ausgabe auf dem Display von U1, V und . (Punkt) an den angebenden Positionen

```
int16_t adc1l, adc1a, adc1r1, adc1r2, adc1b;
```

notwendige Variablen

```
adc1=(results[0]*196);
```

Mit dieser Zeile berechnen wir den adc1 Wert. Dazu wird results[0] mit dem Korrekturfaktor von 196 multipliziert.

Den Korrekturfaktor errechnen wir mit

$$K = \frac{Referensspannung}{2^8} = \frac{5\,V}{256} = 0,0195312 \times 10000 \approx 196$$

Damit haben wir auch die Auflösung der Spannungsmessung berechnet. Diese beträgt bei 2^8 0,0195312 V oder rund 20 mV.

Für die Berechnung wird K mit 10000 multipliziert. Dadurch führen wir nur Festkommaarithmetik durch. Das spart sehr viel Rechenzeit.
Mit dem Korrekturfaktor K kann ein Abgleich der Eingangsspannung vorgenommen werden.

Berechnung der möglichen maximalen Anzeige:

Maximale Ausgabe des PCF 8591 = **255**
Korrekturfaktor = **196**

$$Ausgabe\,\max = Ausgabe\ PCF8591\ max \times Korrekturfaktor = 255 \times 196 = 49980$$

```
if (adc1>50000)
 {
   adc1=50000;
 }
```

Sehen wir uns dazu ein Beispiel an: **adc1 = results[0] x K**
adc1 = 188 x 196 = 36848

Begrenzung der möglichen maximalen Anzeige

```
adc1=adc1/100;                    // 1. Schritt
```

36848 : 100 = 368,48

im ersten Schritt wird **adc1** durch einhundert geteilt
Da **adc1** ein int Wert ist, werden nur ganze Zahlen gespeichert

adc1 = 368

```
adc1a=adc1;                       // 2. Schritt
```

adc1a = 368

im zweiten Schritt speichern wir **adc1** als **adcc1a**

```
adc1r2=adc1a % 10;                // 3. Schritt rechts 2
```

im dritten Schritt wird **adc1a** durch **Modulo 10** geteilt und als **adc1r2** gespeichert
(Modulo (%) bedeutet, Rest einer Division)

adc1r2 = 8

```
adc1b=adc1a/10;                   // 4. Schritt
```

im vierten Schritt wird **adc1a** durch 10 dividiert und als **adc1b** gespeichert

adc1b = 36

```
adc1r1=adc1b % 10;                // 5. Schritt rechts 1
```

im fünften Schritt wird **adc1b** durch **Modulo 10** geteilt und als **adc1r1** gespeichert

adc1r1 = 6

```
adc1l=adc1b/10;                   // 6. Schritt  links
```

im sechsten Schritt wird **adc1b** durch 10 dividiert und als **adc1l** gespeichert

adc1l = 3

```
itoa( adc1l, Buffer, 10 );        // Zeile 1 Zahl 1
lcd_printlc(1,11,Buffer);                                  Anzeige 3
itoa( adc1r1, Buffer, 10 );       // Zeile 1 Zahl 2
lcd_printlc(1,13,Buffer);                                  Anzeige 6
itoa( adc1r2, Buffer, 10 );       // Zeile 1 Zahl 3
lcd_printlc(1,14,Buffer);                                  Anzeige 8
```

in diesem Teil des Programmes wird durch **itoa** die gespeicherten Werte **adc1l, adc1r1** und **adc1r2** gewandelt und an den gewünschten Stellen unserer Anzeige angezeigt.

Die Zahl **368** wird angezeigt durch:

1. Zahl - **adc1l** - **3**
2. Zahl - **adc1r1** - **6**
3. Zahl - **adc1r2** - **8**

Die Berechnung und Darstellung habe ich recht ausführlich dargestellt. Es können einige Schritte gekürzt oder zusammengefasst werden.
Die Berechnung der anderen Zeilen erfolgt genauso. Achtet bitte genau auf die Bezeichnungen der Variablen. Auf Grund der Ähnlichkeit kommt man recht schnell durcheinander.

Das ganze Programm:

```
/* ATB_Ana_Prg_1.c Created: 01.12.2015 21:06:48 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 "avr/interrupt.h"
#include <stdlib.h>

#define PCF8591w   0x90                          // Adressen PCF 8591
#define PCF8591r   0x91

int8_t i;
float results[5],adc1 ,adc2 ,adc3 ,adc4;

void startanzeige()                              // Titelbild
 {
   lcd_command(LCD_CLEAR);
   _delay_ms(2);
   lcd_printlc(1,2,"Analog 1 - I2C");
   lcd_printlc(2,1,"Modul P58 - 4xP");
   lcd_printlc(3,2,"AT1284 und LCD");
   lcd_printlc(4,4,"bei achim S");
   _delay_ms(5000);
 }

int main(void)
 {
   char Buffer[20];
   i2c_init();                                   // Starte I2C Bus
   lcd_init();                                   // Starte I2C LCD
   lcd_light(0);                                 // 0=Licht an, 1=Licht aus
   // Display Befehle
   lcd_command(LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKINGOFF);
   // Display ON/OFF / Cursor ON/OFF / Blinken ON/OFF
   startanzeige();
   lcd_command(LCD_CLEAR);                       // Leere Display
   _delay_ms(2);                                 // warte 2ms

   while(1)
    {
      i2c_start(PCF8591w);
      i2c_write(0x04);                 // Analog out enable, auto increment, channel 0
      i2c_stop();                      // Analogausgang aktivieren, Auto-Inkrement, Kanal 0
```

```
i2c_start(PCF8591r);
i2c_readAck();                          // first data is old (ersten Daten alt)
for (i=0; i<4; i++)
 {
   results[i]=i2c_readAck();
 }
      i2c_readNak();
      i2c_stop();
      _delay_ms(2);

  // Anzeige der P und ADC Werte

itoa( results[0], Buffer, 10 );         // Zeile 1
lcd_printlc(1,1,"P1 ");
lcd_printlc(1,4,Buffer);

itoa( results[1], Buffer, 10 );         // Zeile 2
lcd_printlc(2,1,"P2 ");
lcd_printlc(2,4,Buffer);

itoa( results[2], Buffer, 10 );         // Zeile 3
lcd_printlc(3,1,"P3 ");
lcd_printlc(3,4,Buffer);

itoa( results[3], Buffer, 10 );         // Zeile 4
lcd_printlc(4,1,"P4 ");
lcd_printlc(4,4,Buffer);

lcd_printlc(1,8,"U1 ");
lcd_printlc(2,8,"U2 ");
lcd_printlc(3,8,"U3 ");
lcd_printlc(4,8,"U4 ");

lcd_printlc(1,16,"V");
lcd_printlc(2,16,"V");
lcd_printlc(3,16,"V");
lcd_printlc(4,16,"V");

lcd_printlc(1,12,".");                  // Komma 1 Z
lcd_printlc(2,12,".");                  // Komma 2 Z
lcd_printlc(3,12,".");                  // Komma 3 Z
lcd_printlc(4,12,".");                  // Komma 4 Z

  // Zeile 1
int16_t adc1l, adc1a, adc1r1, adc1r2, adc1b;
adc1=(results[0]*196);
if (adc1>50000)
 {
   adc1=50000;
 }

adc1=adc1/100;                          // 1. Schritt
adc1a=adc1;                             // 2. Schritt
adc1r2=adc1a % 10;                      // 3. Schritt rechts 2
adc1b=adc1a/10;                         // 4. Schritt
adc1r1=adc1b % 10;                      // 5. Schritt rechts 1
```

```
adc1l=adc1b/10;                          // 6. Schritt  links

itoa( adc1l, Buffer, 10 );               // Zeile 1 Zahl 1
lcd_printlc(1,11,Buffer);
itoa( adc1r1, Buffer, 10 );              // Zeile 1 Zahl 2
lcd_printlc(1,13,Buffer);
itoa( adc1r2, Buffer, 10 );              // Zeile 1 Zahl 3
lcd_printlc(1,14,Buffer);

// Zeile 2
int16_t adc2l, adc2a, adc2r1, adc2r2, adc2b;
adc2=(results[1]*196);
if (adc2>50000)
 {
   adc2=50000;
 }

adc2=adc2/100;                           // 1. Schritt
adc2a=adc2;                              // 2. Schritt
adc2r2=adc2a % 10;                       // 3. Schritt rechts 2
adc2b=adc2a/10;                          // 4. Schritt
adc2r1=adc2b % 10;                       // 5. Schritt rechts 1
adc2l=adc2b/10;                          // 6 Schritt  links

itoa( adc2l, Buffer, 10 );               // Zeile 2 Zahl 1
lcd_printlc(2,11,Buffer);
itoa( adc2r1, Buffer, 10 );              // Zeile 2 Zahl 2
lcd_printlc(2,13,Buffer);
itoa( adc2r2, Buffer, 10 );              // Zeile 2 Zahl 3
lcd_printlc(2,14,Buffer);

// Zeile 3
int16_t adc3l, adc3a, adc3r1, adc3r2, adc3b;
adc3=(results[2]*196);
if (adc3>50000)
 {
   adc3=50000;
 }

adc3=adc3/100;                           // 1. Schritt
adc3a=adc3;                              // 2. Schritt
adc3r2=adc3a % 10;                       // 3. Schritt rechts 2
adc3b=adc3a/10;                          // 4. Schritt
adc3r1=adc3b % 10;                       // 5. Schritt rechts 1
adc3l=adc3b/10;                          // 6 Schritt  links

itoa( adc3l, Buffer, 10 );               // Zeile 3 Zahl 1
lcd_printlc(3,11,Buffer);
itoa( adc3r1, Buffer, 10 );              // Zeile 3 Zahl 2
lcd_printlc(3,13,Buffer);
itoa( adc3r2, Buffer, 10 );              // Zeile 3 Zahl 3
lcd_printlc(3,14,Buffer);

// Zeile 4
int16_t adc4l, adc4a, adc4r1, adc4r2, adc4b;
```

```
      adc4=(results[3]*196);
      if (adc4>50000)
       {
         adc4=50000;
       }

      adc4=adc4/100;                    // 1. Schritt
      adc4a=adc4;                       // 2. Schritt
      adc4r2=adc4a % 10;                // 3. Schritt rechts 2
      adc4b=adc4a/10;                   // 4. Schritt
      adc4r1=adc4b % 10;                // 5. Schritt rechts 1
      adc4l=adc4b/10;                   // 6 Schritt  links

      itoa( adc4l, Buffer, 10 );        // Zeile 4 Zahl 1
      lcd_printlc(4,11,Buffer);
      itoa( adc4r1, Buffer, 10 );       // Zeile 4 Zahl 2
      lcd_printlc(4,13,Buffer);
      itoa( adc4r2, Buffer, 10 );       // Zeile 4 Zahl 3
      lcd_printlc(4,14,Buffer);

      _delay_ms(100);

    }
}
```

So könnte das Ergebnis aussehen

Anzeige von links nach rechts

- Angabe des analogen Kanals (P1 bis P4)
- Werte das PCF 8591
- Angabe der U1 bis U4
- Umgerechnete Spannungen U1 bis U4

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