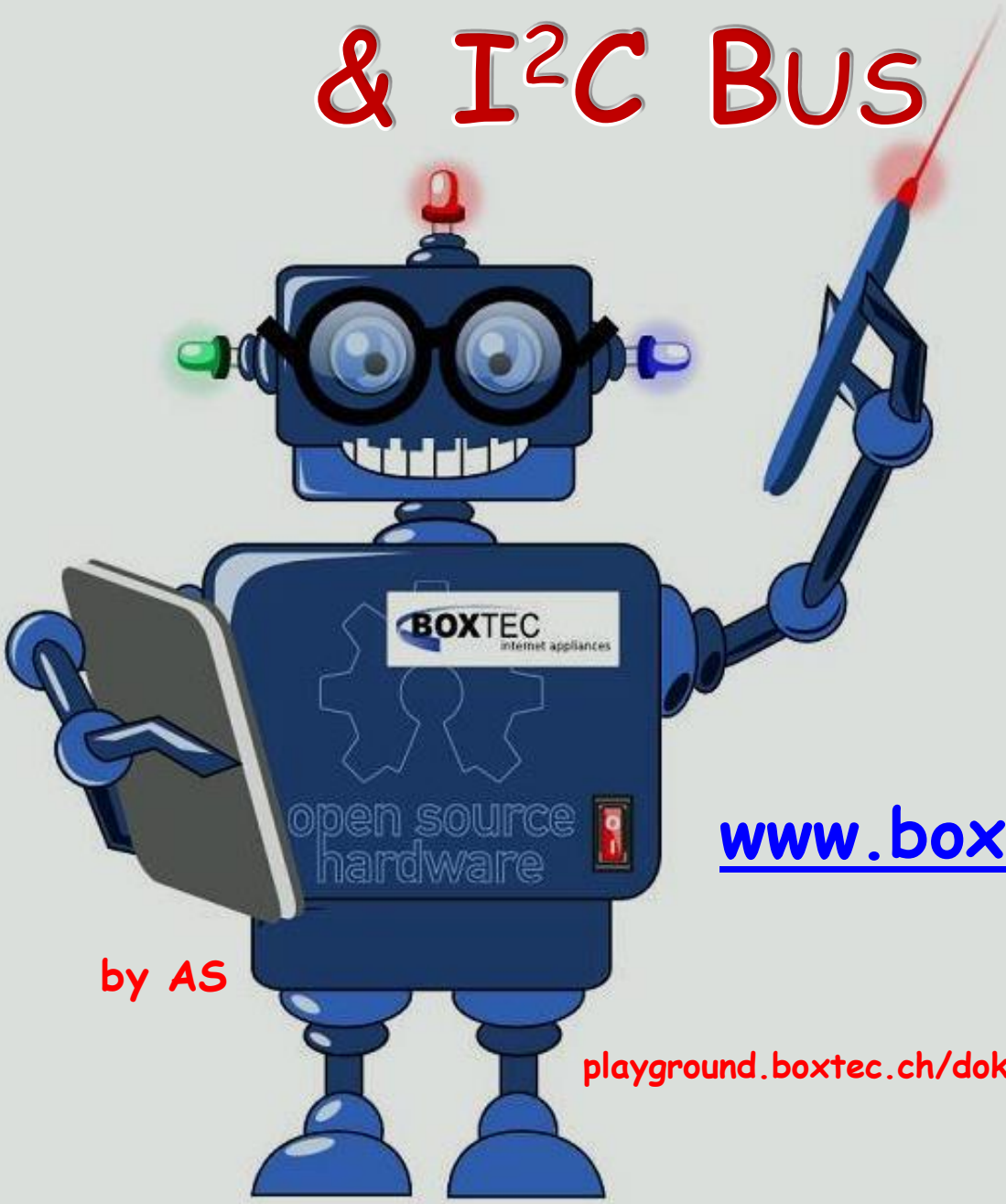
Mikrokontroller & I²C Bus

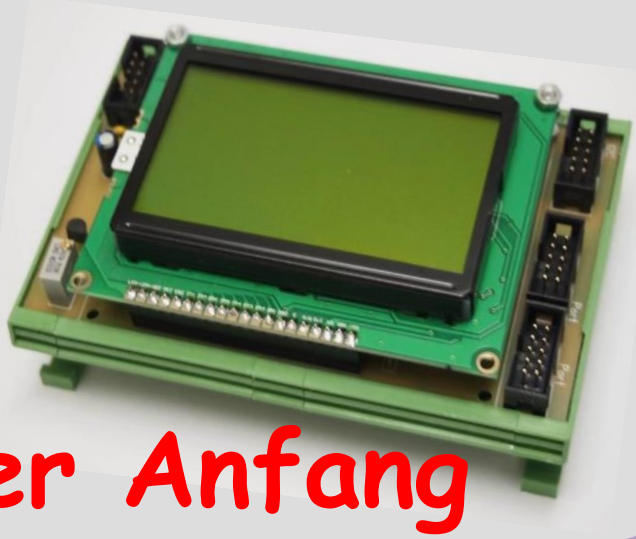


www.boxtec.ch

by AS

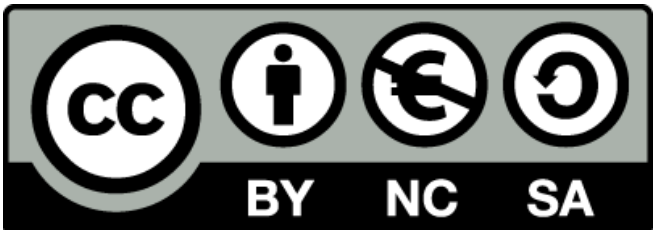
playground.boxtec.ch/doku.php/tutorial

Graphik Display (128x64)
mit dem KS0108, 2 Ports
und 2 x I²C Verbindung



Graphik 2 – Der Anfang

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 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 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 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

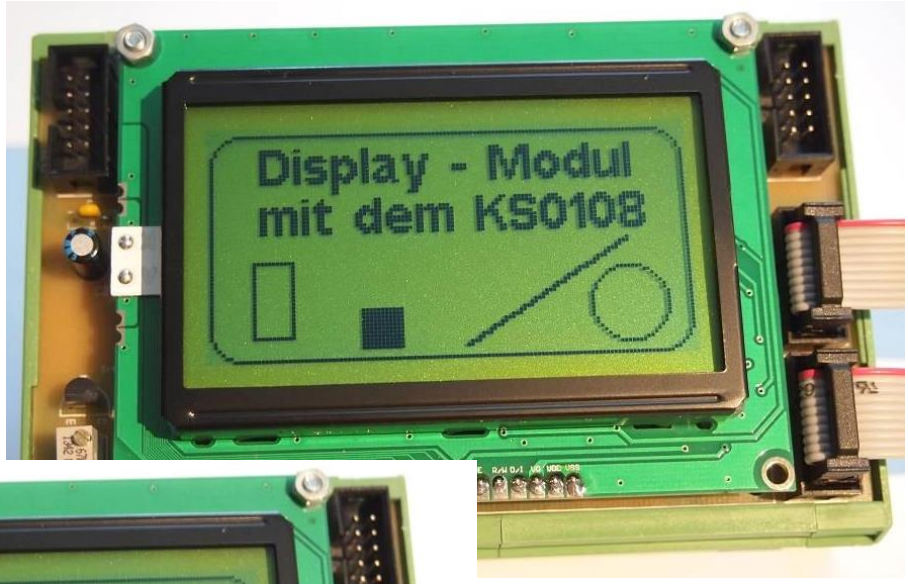
Bestimmungsgemäße Verwendung

- 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.

Graphik 2 – Der Anfang

Im ersten Teil haben wir die Hardware aufgebaut. Eine Anzeige haben wir bekommen. Es wird auch das dargestellt, was wir haben wollen.
Müssen wir „nur noch" angeben, was angezeigt werden soll.

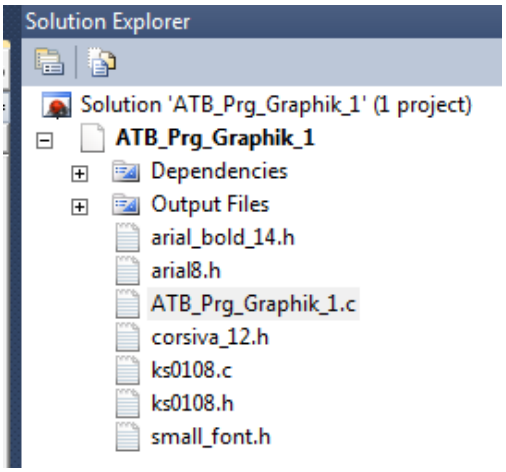
Auf dem ersten Bild ist der Startbildschirm zu sehen. Es wird eine Schrift und einige geometrische Figuren angezeigt.



Auf dem zweiten Bild werden verschiedene Schriftarten in verschiedenen Grössen angezeigt.

Welche Dateien brauchen wir dazu?

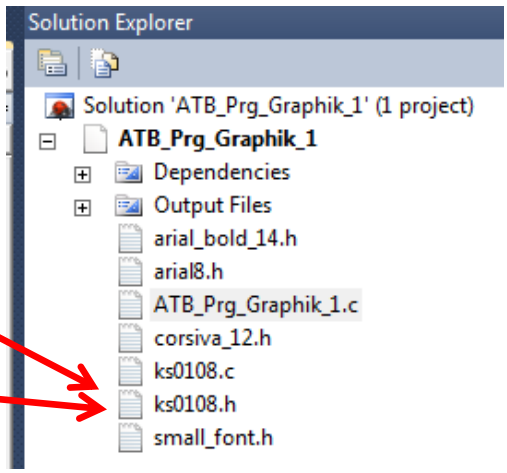
- ATB_Prg_Graphik_1.c
- ks0108.c
- ks0108.h
- arial_bold_14.h
- arial8.h
- corsiva_12.h
- small_font.h



In der Datei **ATB_Prg_Graphik_1** steht unser eigentliches Programm. **ks108.c** und **ks108.h** sind unsere Libs zum Betrieb des Displays.
In **arial_bold_14, arial8, corsiva_12** und **small_font** stehen unsere verschiedenen Schriftarten.
Diese Dateien müssen in unseren Solution Explorer eingebunden werden. Den Ablauf habe ich in einem anderen Teil bereits beschrieben.

```
#define F_CPU 16000000UL
#include <inttypes.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include "ks0108.h"
#include "arial_bold_14.h"
#include "corsiva_12.h"
#include "arial8.h"
#include "small_font.h"
```

Diese Dateien müssen in unserem Programm eingebunden werden.

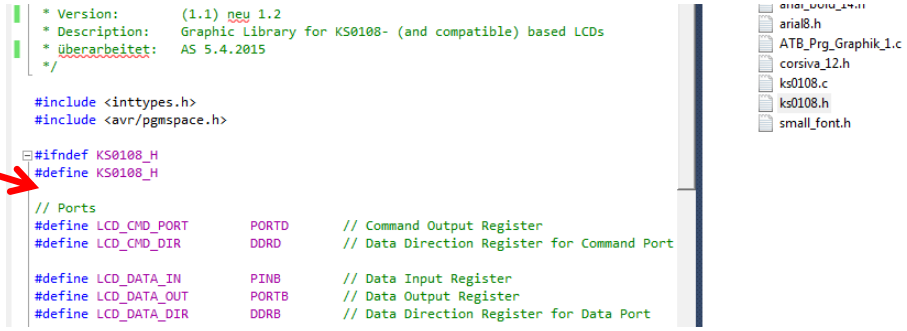


Die Datei ks0108.h müssen wir an unsere Hardware anpassen.

Die Datei wird durch ein Doppelklick auf den Namen aufgerufen

Anschliessen erscheint diese Bild:

Inhalt der Datei



Das ori wurde von Maximilian Thiele erstellt. Leider ist er nicht mehr erreichbar.

Als nächste müssen wir in der Datei ks0108.h unsere verwendeten Ports angegeben werden.

```
// Ports
#define LCD_CMD_PORT        PORTD       // Command Output Register
#define LCD_CMD_DIR         DDRD        // Data Direction Register for Command Port

#define LCD_DATA_IN         PINB        // Data Input Register
#define LCD_DATA_OUT        PORTB       // Data Output Register
#define LCD_DATA_DIR        DDRB        // Data Direction Register for Data Port
```

Mit der Angabe **Port B** gebe ich die Datenausgänge am AT1284p an. Diese werden auf die Dateneingänge des Graphikdisplays geführt. Die genaue Zuordnung habe ich bereits im ersten Teil beschrieben. Die Zuordnung von **0** bis **7** ist vorgegeben.
Mit der Angabe **Port D** gebe ich die Steuerausgänge am AT1284p.

An dieser Stelle gebe ich die Belegung meiner Hardware an. An der rechten Seite habe ich die ori Bezeichnung als Kommentar notiert.

```
// Command Port Bits
#define D_I             6       // D/I Bit Number 0x00
#define R_W             5       // R/W Bit Number 0x01
#define EN              2       // EN Bit Number 0x02
#define CSEL1           0       // CS1 Bit Number 0x03
#define CSEL2           1       // CS2 Bit Number 0x04
```

Die **Command Port Bits** müssen an die verwendete Hardware angepasst werden.
Die Zählung der Pins beginnt bei **0** !

Sehen wir uns ein Beispiel dazu an:

```
/* ATB_Prg_Graphik_1.c Created: 04.04.2015 12:17:16 Author: AS */

#define F_CPU 16000000UL
#include <inttypes.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include "ks0108.h"
#include "arial_bold_14.h"
#include "corsiva_12.h"
#include "arial8.h"
#include "small_font.h"

int main(void)
 {
   DDRD=0b00001000;                          // Angabe Pin am AT1284p
   //PORTD &=~(1<<PD3);                      // LED aus
   PORTD |=(1<<PD3);                         // LED Licht an
   ks0108Init(0);                            // Initialize GLCD
   _delay_ms(5);                             // Pause
   // Ausgabe Rand
   ks0108DrawRoundRect(1, 2, 125, 61, 8, BLACK);
   //Ausgabe Linie
   ks0108DrawLine(110, 32, 67, 60, BLACK);
   //Ausgabe Viereck leer
   ks0108DrawRect(12, 37, 10, 20, BLACK);
   //Ausgabe Viereck gefüllt
   ks0108FillRect(40, 50, 10, 10, BLACK);
   // Ausgabe Kreis
   ks0108Circle(110, 50, 10, BLACK);
   // Auswahl Schrift
   ks0108SelectFont(Arial_Bold_14, ks0108ReadFontData, BLACK);
   ks0108GotoXY(14,5);                       // Angabe Position
   // Ausgabe Text
   ks0108Puts_P(PSTR("Display - Modul\nmit dem KS0108"));
   _delay_ms(5000);                          // Pause
   ks0108SelectFont(small_font, ks0108ReadFontData, BLACK);
   ks0108GotoXY(35,35);                      //Angabe Position
   ks0108Puts_P(PSTR("AS 2015"));            // Ausgabe Text
   _delay_ms(5000);                          // Pause
   ks0108ClearScreen();                      // Display löschen
   _delay_ms(2000);                          // Pause
   // Ausgabe Rand
   ks0108DrawRoundRect(1, 2, 125, 61, 8, BLACK);
   // Text 1
   ks0108SelectFont(Corsiva_12, ks0108ReadFontData, BLACK);
   ks0108GotoXY(10,8);                       // Angabe Position
   ks0108Puts_P(PSTR("Text mit Corsiva 12"));// Ausgabe Text
   // Text 2
   ks0108SelectFont(Arial8, ks0108ReadFontData, BLACK);
   ks0108GotoXY(12,22);                      // Angabe Position
```

```
  ks0108Puts_P(PSTR("Text mit Arial 8"));          // Ausgabe Text
  // Text 3
  ks0108SelectFont(small_font, ks0108ReadFontData, BLACK);
  ks0108GotoXY(15,35);                             //Angabe Position
  ks0108Puts_P(PSTR("Text mit Small Font"));       // Ausgabe Text
  // Text 4
  ks0108SelectFont(Arial_Bold_14, ks0108ReadFontData, BLACK);
  ks0108GotoXY(10,45);                             // Angabe Position
  ks0108Puts_P(PSTR("Arial Bold 14"));             // Ausgabe Text
  while(1);
 }
```

Das Programm in kleinen Stücken:

```
#define F_CPU 16000000UL
#include <inttypes.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include "ks0108.h"
#include "arial_bold_14.h"
#include "corsiva_12.h"
#include "arial8.h"
#include "small_font.h"
```

Angabe der Quarzfrequenz und der notwendigen Dateien

```
int main(void)
 {
  DDRD=0b00001000;                                 // Angabe Pin am AT1284p
  //PORTD &=~(1<<PD3);                             // LED aus
  PORTD |=(1<<PD3);                                // LED Licht an
  ks0108Init(0);                                   // Initialize GLCD
  _delay_ms(500);                                  // Pause
```

Angabe des Port/Pins am AT184p, Angabe des Pins zum LED ein/ausschalten, ks0108 wird initiiert und gewartet bis er fertig ist. Zeit bitte anpassen.

```
  // Ausgabe Rand
  ks0108DrawRoundRect(1, 2, 125, 61, 8, BLACK);
```

Mit dieser Anweisung kann ich z.B. einen Rand mit runden Ecken ausgeben.

Aufruf

```
ks0108DrawRoundRect(uint8_t x, uint8_t y, uint8_t width, uint8_t height, uint8_t radius, Black);
```

X – Position Beginn waagerecht, Bereich von 0 – 127, Breite beachten
Y – Position Beginn senkrecht, Bereich von 0 – 63, Höhe beachten
width – Länge nach rechts
height – Länge nach unten
radius – Angabe des Radius, Bereich 0 – 10 (wahrscheinlich)

```
  //Ausgabe Linie
  ks0108DrawLine(110, 32, 67, 60, BLACK);
```

Mit dieser Anweisung kann ich eine Linie ausgeben

Aufruf
ks0108DrawLine(uint8_t x1, uint8_t y1, uint8_t x2, uint8_t y2, Black);

X1 – Position Anfang waagerecht, Bereich von 0 - 127
Y1 – Position Anfang senkrecht, Bereich von 0 - 63
X2 – Position Ende waagerecht, Bereich von 0 - 127
Y2 – Position Ende senkrecht, Bereich von 0 – 63

```
//Ausgabe Viereck leer
ks0108DrawRect(12, 37, 10, 20, BLACK);
```

Aufruf
ks0108DrawRect(uint8_t x, uint8_t y, uint8_t width, uint8_t height, Black);

X – Position Beginn waagerecht, Bereich von 0 – 127, Breite beachten
Y – Position Beginn senkrecht, Bereich von 0 – 63, Höhe beachten
width – Länge nach rechts
height – Länge nach unten

```
//Ausgabe Viereck gefüllt
ks0108FillRect(40, 50, 10, 10, BLACK);
```

Aufruf
ks0108FillRect(uint8_t x, uint8_t y, uint8_t width, uint8_t height, Black);

X – Position Beginn waagerecht, Bereich von 0 – 127, Breite beachten
Y – Position Beginn senkrecht, Bereich von 0 – 63, Höhe beachten
width – Länge nach rechts
height – Länge nach unten

```
// Ausgabe Kreis
ks0108Circle(110, 50, 10, BLACK);
```

Aufruf
ks0108Circle(uint8_t cx, uint8_t cy , uint8_t radius, Black);

CX – Position Mittelpunkt waagerecht, Bereich von 0 – 127, Breite beachten
CY – Position Mittelpunkt senkrecht, Bereich von 0 – 63, Höhe beachten
radius – Radius Kreis

```
// Text 1 – Ausgabe Schrift
ks0108SelectFont(Corsiva_12, ks0108ReadFontData, BLACK);
ks0108GotoXY(10,8);                          // Angabe Position
ks0108Puts_P(PSTR("Text mit Corsiva 12"));  // Ausgabe Text
```

Aufruf
ks0108SelectFont(const char* font, ks0108FontCallback callback, Black)

const char* font – Angabe der Schrift

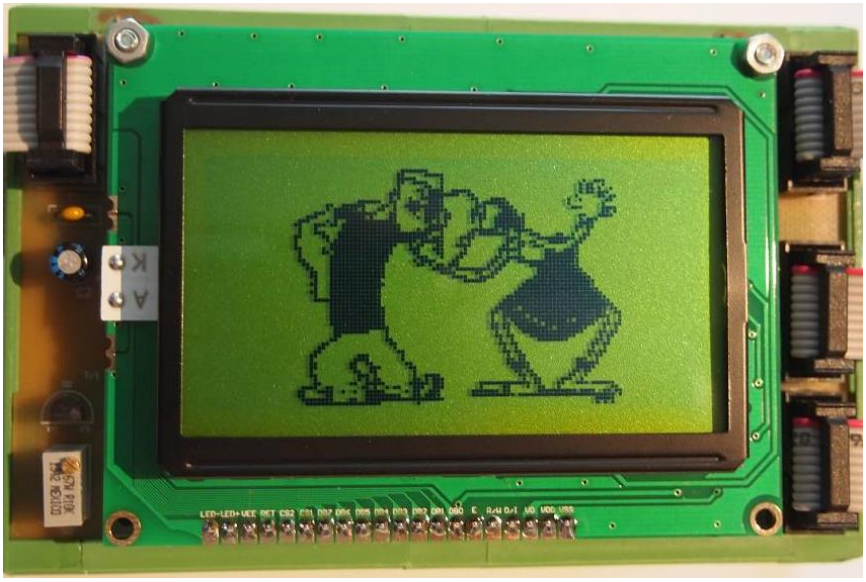
```
ks0108GotoXY(10,8);                          // Angabe Position
```

X – Position waagerecht, Bereich von 0 – 127, Breite beachten
Y – Position senkrecht, Bereich von 0 – 63, Höhe beachten

```
ks0108ClearScreen();                    // Display löschen
```

Mit dieser Anweisung löschen wir das gesamte Display

Im Programm sind noch andere Funktionen. Wer will kann diese testen und mich darüber informieren.



Neben Schrift und relativ einfachen Zeichen, können auch Bilder dargestellt werden.
Die Funktionen sind im Programm enthalten.
Diese müssen aber anders aufgerufen werden.

Im nächsten Teil mehr dazu.

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