CONFIG.H

Folgende Parameter sollten/muessen eingestellt werden

```
#ifndef F_CPU
#define F_CPU                8000000L
#endif
```

Hier wird die Frequenz eingestellt, mit der der uC laut Fuses laeuft.

```
#define GLCD_WR_PIN          2
#define GLCD_WR_PORT         PORTD

#define GLCD_RD_PIN          3
#define GLCD_RD_PORT         PORTD

#define GLCD_CE_PIN          4
#define GLCD_CE_PORT         PORTD

#define GLCD_CD_PIN          5
#define GLCD_CD_PORT         PORTD

#define GLCD_RST_PIN         6
#define GLCD_RST_PORT        PORTD

#define GLCD_FS_PIN          7
#define GLCD_FS_PORT         PORTD

#define GLCD_DATA_PORT       PORTB
#define GLCD_DATA_PIN        PINB
#define GLCD_DATADIR         DDRB
```

Die Pins fuer den Datenport liegen hintereinander auf einem Port.
Die Pins WR, RD, CE, CD, RST, FS koennen frei gewaehlt werden und muessen nicht auf dem gleichen Port liegen.

```
#define GLCD_XMAX 128
#define GLCD_XMIN 0
#define GLCD_YMAX 128
#define GLCD_YMIN 0
```

Hier werden die Begrenzungen fuer das Display eingestellt.

```
#define GLCD_VRAM_SIZE                  8192
```

Die Groesse des Videorams auf dem Display. Aus diesem Wert errechnet sich, wieviele Seiten fuer eventuelles #Mapping zur Verfuegung stehen.

```
#define KEY_DDR         DDRC
#define KEY_PORT        PORTC
#define KEY_PIN         PINC
#define KEY_ENTER       0
#define KEY_MENU        1
#define KEY_DOWN        2
#define KEY_UP          3
#define ALL_KEYS    (1<<KEY_ENTER | 1<<KEY_MENU | 1<<KEY_UP |1<<KEY_DOWN)
```

Hier wird der Port fuer die 4 verwendeten Tasten (Enter, Menue-Tab, Up, Down) festgelegt. Der Wert fuer ALL_KEYS sollte in keinem Fall geaendert werden. Ebenso wenig die Namen der Tasten.

```
#define PHASE_A     (PIND & 1<<PD0)
#define PHASE_B     (PIND & 1<<PD1)
```

Diese beiden Parameter sind fuer den Drehregler noetig.

```
#define MAX_GUI_OBJECTS         20
#define MAX_GUI_LABEL_SIZE      20
#define MAX_GUI_TEXT_SIZE       40
#define GUI_MAX_TIMER           10
```

Je nach Groesse SRAM des verwendeten uC koennen diese Werte erhoeht werden. Die Standard Werte sind passend fuer einen Atmega328 mit 4kb SRAM.

INITIALISIEREN

Inititalisieren des Grafik Treibers.

```
GLCD_INIT();
GLCD_CLEAR();
```

Initialisieren der Tasten und des Drehmomentgebers.

```
INIT_INPUTS();
```

Initialisieren des Timer Systems.

```
GUI_TIMER_INIT();
```

Zusaetzlich zu den Initialisierungs Funktionen muss am Ende sei(); aufgerufen werden, da die Abfrage der Tasten und des Drehreglers ueber Timer erfolgen. Ausserdem braucht das Timer System selber Interrupts.

Was bei der Initialisierung nicht vergessen werden darf ist die Zuweisung der Fonts zu erledigen.

```
GUI_SET_NORMAL_FONT(&Arial__8);
GUI_SET_SMALL_FONT(&Verdana__6);
GUI_SET_BIG_FONT(&Fully);
```

Eine vollstaendige Init Funktion koennte so aussehen.

```
void INIT_ALL(void)
{
  // Set the fonts for the gui system
  GUI_SET_NORMAL_FONT(&Arial__8);
  GUI_SET_SMALL_FONT(&Verdana__6);
  GUI_SET_BIG_FONT(&Fully);

  // Init the low level graphic driver
  GLCD_INIT();
  GLCD_CLEAR();

  // Init the key and rotary switch system
  INIT_INPUTS();
  GUI_TIMER_INIT();
  // finaly enable interrupts
  sei();
}
```

Aufbau eines Minimal-Dialogs

```
void  button,callback_cancel_button(void)
{
        return 0;
}



void DIALOG_SHOW(void)
{
  GUI_DIALOG dialog;
  GUI_OBJECT label;
  GUI_OBJECT button;

  dialog=GUI_DIALOG_NEW();
    label=GUI_OBJECT_NEW(GUI_OBJECT_TYPE_LABEL, 5, 5, 120, 14, text);
    button=GUI_OBJECT_NEW(GUI_OBJECT_TYPE_BUTTON, 75, 80, 30, 14, "QUIT");

    GUI_OBJECT_SET_FRAME(&label,TRUE);
    GUI_OBJECT_SET_CALLBACK(&button,callback_cancel_button,GUI_CB_RETURN);

    GUI_OBJECT_SET_HELP_P(&button,"Click to exit");

    GUI_DIALOG_INSERT_OBJECT(&dialog_drive,&label);
    GUI_DIALOG_INSERT_OBJECT(&dialog_drive,&button);

    // Show the dialog
    GUI_DIALOG_DISPLAY(&dialog_drive);
}
```

Aufbau von Main und aufruf des GUI Systems

```
int main(void)
{
  INIT_ALL();
  GUI_MAIN_MENU_SHOW();
  GUI_RUN();
  return 0;
}
```

INIT_ALL(); ist die Funktion, die fuer das Initialisieren aufgebaut wurde.
Der erste Dialog wird mit der Aufbau Funktion aufgerufen und anschliessend wird GUI_RUN(); ausgefuehrt. Den Rest uebernimmt das GUI-System. Diese Funktion wird niemals beendet. Das Handling der Aufrufe wird komplett in Callbacks ausgefuehrt.

Ein vollstaendiges Beispiel!

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/eeprom.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/wdt.h>
#include <math.h>

#include "include/config.h"
#include "include/general.h"
#include "include/t6963c.h"
#include "include/graphics.h"
#include "include/inputs.h"
#include "include/mem_check.h"
#include "include/font.h"
#include "data/fonts/font_def.h"
#include "include/gui.h"
#include "include/timer.h"

void INIT_ALL(void)
{
  // Set the fonts for the gui system
  GUI_SET_NORMAL_FONT(&Arial__8);
  GUI_SET_SMALL_FONT(&Verdana__6);
  GUI_SET_BIG_FONT(&Fully);
```

```
  // Init the low level graphic driver
  GLCD_INIT();
  GLCD_CLEAR();

  // Init the key and rotary switch system
  INIT_INPUTS();
  GUI_TIMER_INIT();
  // finaly enable interrupts
  sei();
}


void  button,callback_cancel_button(void)
{
          return 0;
}


void DIALOG_SHOW(void)
{
  GUI_DIALOG dialog;
  GUI_OBJECT label;
  GUI_OBJECT button;

  dialog=GUI_DIALOG_NEW();
    label=GUI_OBJECT_NEW(GUI_OBJECT_TYPE_LABEL, 5, 5, 120, 14, text);
    button=GUI_OBJECT_NEW(GUI_OBJECT_TYPE_BUTTON, 75, 80, 30, 14, "QUIT");

    GUI_OBJECT_SET_FRAME(&label,TRUE);
    GUI_OBJECT_SET_CALLBACK(&button,callback_cancel_button,GUI_CB_RETURN);

    GUI_OBJECT_SET_HELP_P(&button,"Click to exit");

    GUI_DIALOG_INSERT_OBJECT(&dialog_drive,&label);
    GUI_DIALOG_INSERT_OBJECT(&dialog_drive,&button);

    // Show the dialog
    GUI_DIALOG_DISPLAY(&dialog_drive);
}

int main(void)
```

```
{
  INIT_ALL();
  GUI_MAIN_MENU_SHOW();
  GUI_RUN();
  return 0;
}
```

In dem Beispiel wird nur der Dialog mit einem Text Label und einem Button angezeigt. Da in der Callback Funktion noch keine Aktion drin stehet, passiert sonst eigendlich Nichts weiter.

FUNKTIONEN

Grafik Treiber

void GLCD_CLEAR(void);

Loeschen des Displays.

void GLCD_INIT(void);

Initialisieren der Grafik Funktionen.

void GLCD_SET_PAGE(int page);

Legt die aktive Seite fuer's Mapping fest. Das Mapping wird vom Gui nicht benutzt.

void GLCD_CLEAR_PAGE(int page);

Loescht eine einzelnde Seite.

void GLCD_COPY_PAGE(int source, int dest);

Kopiert den Inhalt einer Seite in eine Andere.

Fonts

void GLCD_PRINT_p(unsigned char x,unsigned char y, const char *in, const struct FONT_DEF *struct1, unsigned char invers);

Schreibt einen String aus dem Flash auf das Display.

void GLCD_PRINT(unsigned char x,unsigned char y, const char *in, const struct FONT_DEF *struct1, unsigned char invers);

Schreibt einen String aus dem ram auf's Display.

int GLCD_STRLEN(const char *in, const struct FONT_DEF *struct1);

Giebt die Laenge eines Strings in Pixeln zureuck.

uint8_t GLCD_CHAR(unsigned char x,unsigned char y, char in, const struct FONT_DEF *struct1);

Schreibt einen einzelnden Charakter auf's Display.

int GLCD_GET_FONT_HEIGHT(const struct FONT_DEF *struct1);

Gibt die Hoehe des ausgewaehlten Fonts zureuck.

GUI System

void GUI_SET_SMALL_FONT(const struct FONT_DEF *font);

Dient zum festlegen des kleinen Fonts.

void GUI_SET_BIG_FONT(const struct FONT_DEF *font);

Dient zum festlegen des grossen Fonts.

void GUI_SET_NORMAL_FONT(const struct FONT_DEF *font);

Dient zum festlegen des normalen Fonts.

GUI_OBJECT GUI_OBJECT_NEW(int type, int x, int y, int w, int h, char *label);
Erstellen eines neuen Objects.

GUI_DIALOG GUI_DIALOG_NEW(void);
Erstellen eines neuen Dialogs.

void GUI_OBJECT_SET_HELP(GUI_OBJECT *object, char *text);
Hilfstext fuer die Statuszeile festlegen.

void GUI_OBJECT_SET_HELP_p(GUI_OBJECT *object, char *text);
Das selbe, nur dass der String aus dem Flash kommt.

void GUI_OBJECT_SET_LABEL(GUI_OBJECT *object, char *text);
Aendert den Text des Objects.

void GUI_OBJECT_SET_TEXT(GUI_OBJECT *object, char *text);
Aendert den Text eines Eingabefeldes.

void GUI_OBJECT_SET_POSITION(GUI_OBJECT *object, int x, int y, int w, int h);
Legt die Positionen und Groesse eines Objects neu fest.

void GUI_OBJECT_SET_FRAME(GUI_OBJECT *object, int frame);
Legt fest, ob ein Object bei der Darstellung einen Rahmen haben soll.

void GUI_OBJECT_SET_CALLBACK(GUI_OBJECT *object, uint8_t (*cb)(void), int callback_type);
Setzt einen Pointer auf den Callback eines aktiven Objects.

void GUI_OBJECT_SET_VALUES(GUI_OBJECT *object, int16_t value, int16_t min, int16_t max, int16_t step_size);
Legt die Zahlen fuer Prozentbalken fest.

void GUI_OBJECT_SET_VALUE_UP(GUI_OBJECT *object);
Zaehlt einen Prozentbalken um 1 hoch.

void GUI_OBJECT_SET_VALUE_DOWN(GUI_OBJECT *object);
Zieht 1 von einem Prozentbalken ab.

void GUI_OBJECT_SET_VALUE(GUI_OBJECT *object, uint16_t value);
Setzt den Wert eines Prozentbalkens.

void GUI_OBJECT_SET_XBM(GUI_OBJECT *object, const unsigned char *xbm);
Gibt den Pointer auf ein Bild an.

void GUI_OBJECT_SET_FONT(GUI_OBJECT *object, const struct FONT_DEF *font);
Aendert den Font eines Objects.

char *GUI_OBJECT_GET_TEXT(GUI_OBJECT *object);
Gibt den Text eines Eingabefeldes zureuck.

int GUI_OBJECT_GET_VALUE(GUI_OBJECT *object);

Gibt den Wert eines Prozentbalken zurueck.

void GUI_OBJECT_REFRESH(GUI_OBJECT *object);

Zeichnet ein Object neu auf dem Display

void GUI_OBJECT_INSERT_LIST_P(GUI_OBJECT *object, PGM_P *elements, uint8_t num);

Fuegt eine Liste aus dem Flash ein.

void GUI_OBJECT_INSERT_NLIST(GUI_OBJECT *object, uint8_t min, uint8_t max);

Fuegt eine Numerische Liste ein.

void GUI_OBJECT_LIST_SET_COUNT(GUI_OBJECT *object, uint8_t num);

Fuer eine Liste aus dem ram.

uint8_t GUI_OBJECT_LIST_GET_INDEX(GUI_OBJECT *object);

Gibt den Index des zuletzt ausgewaehlten Eitrags zurueck.

void GUI_OBJECT_INSERT_LIST(GUI_OBJECT *object, uint8_t num);

Fuer eine Liste aus dem ram.

void GUI_DIALOG_INSERT_OBJECT(GUI_DIALOG *dialog, GUI_OBJECT *object);

Fuegt ein Object in einen Dialog ein.

void GUI_DIALOG_DISPLAY(GUI_DIALOG *dialog);

Zeigt einen Dialog an.

void GUI_OBJECT_DISPLAY(GUI_OBJECT *object);

Zeigt ein Object an.

void GUI_DIALOG_SET_ACTIVE_DIALOG(GUI_DIALOG *dialog);

Legt den aktiven Dialog fest.

void GUI_RUN(void);

Startet das Gui system.

Die Funktionen und ihre Benutzung sind gut im Quellcode dokumentiert.