Eclipse Projekt BlinkLED

LED blinkt jede 0,5 Sekunden und gibt dies auf einen LC-Display aus.
Zusätzlich wird ein Analogwert auf diesem Display ausgegeben, welcher im A/D-Wandler 0 eingelesen wird.

Headerdatei (LCD):

```
#ifndef LCD_H
#define LCD_H

//LCD-Anschlüsse; Es werden P0.5 - P0.15 zur Ansteuerung des LCD benutzt
#define RS  1<<5
#define RW  1<<6
#define E 1<<7
#define DB0 1<<8
#define DB1 1<<9
#define DB2 1<<10
#define DB3 1<<11
#define DB4 1<<12
#define DB5 1<<13
#define DB6 1<<14
#define DB7 1<<15

#endif
```

Programmcode:

```
/* ********************************************************
               Function declarations
  ******************************************************** */

void initialize(void);
void timer0Init(void);
void pllInit(void);
void feed(void);
void lcdInit(void);
void wait5ms(void);
void enable(void);
void writeChar(char c);
void writeString(char s[]);
void returnHome(void);
void displayClear(void);
void switchLED(void);
void floatToString(float f, char voltage[5]);
void setDDAdress(void);
void output(char spannungLED[]);
void canInit(void);

void IRQ_Routine (void)    __attribute__ ((interrupt("IRQ")));
void FIQ_Routine (void)    __attribute__ ((interrupt("FIQ")));
void SWI_Routine (void)    __attribute__ ((interrupt("SWI")));
```

```
void UNDEF_Routine (void) __attribute__ ((interrupt("UNDEF")));

/*****************************************************
                  Header files
 *****************************************************/

#include "LPC22xx.h"
#include "LCD.h"

/*****************************************************
                       MAIN
*****************************************************/

#define LED 1<<30

int       main (void) {

          float spannung;
          char spannungLED[12]; //Spannungsausgabe auf LED

          initialize();

          while (1) {
                    if(T0IR==1){
                    //Falls MR0 Interrupt gesetzt ist

                              spannung = (float)((ADDR & 0x000FFC00) >>
6)*3.3/1024.0;
                              //Bits 6-15 von ADDR enthalten gewandelte Spannung

                              floatToString(spannung, spannungLED);
                              switchLED();
                              output(spannungLED);
                              T0IR=1;
                              //MR0 Interrupt wird rückgesetzt
                    }
          }
}

/*****************************************************
                      initialize
*****************************************************/

void initialize(void){

          pllInit();
          //PLL-Konfiguration (Phase Locked Loop)

          timer0Init();

          PINSEL0=0x00000000;
          //P0.0  - P0.15 bekommen 1. Funktionalität(GeneralPurposeInputOutput)

          PINSEL1=0x00414000;
          //P0.16 - P0.22, P0.25 und P0.28 bis P0.31 bekommen 1.
Funktionalität(GPIO)
          //P0.23 wird CAN-RD2
          //P0.24 wird CAN-TD2
          //P0.27 wird AIN0 (A/D Converter)


          IO0DIR = LED;
          //P0.30 wird zum Ausgang, P0.0-P0.26, P0.28, P0.29 und P0.31 werden
Eingang
```

```
        IO0CLR = LED;
        //P 0.30 wird Low gesetzt => LED leuchtet

        ADCR = 0x01210D01;
        //A/D-Converter-Initialisierung:
        //AIN0 als Eingang ausgewählt (P0.27)
        //A/DC-Clock = pclk/(13+1) => ist knapp unter geforderten 4.5MHz
        //alle 11 Clocks wird eine neue 10-Bit-A/D-Wandlung gestartet

        lcdInit();
}

/*********************************************************
                    timer0Init
*********************************************************/

void timer0Init(void){

        T0PR = 0x0;
        //Prescale Register (Überlaufwert) 0 => pclk = 58.9824MHz = Timer-
                Clock

        T0TCR=0x2;
        //Prescale-Counter und Timer-Counter werden resettet

        T0TCR=0x1;
        //Prescale-Counter und Timer-Counter werden gestartet

        T0MCR = 0x3;
        //TimerCounter = MatchRegister0 => Interrupt 0 + Timer Counter Reset

        T0MR0=58982400;
        //MatchRegister0 entspricht 0.5s
}

/*********************************************************
                    pllInit
*********************************************************/

#define PLOCK 1<<10

void pllInit(void)  {

        //                Setting the Phased Lock Loop (PLL)
        //              ---------------------------------
        //
        // Olimex LPC-H2294 has a 14.7456 MHz crystal
        //
        // We'd like the LPC-H2294 to run at 58.9824 MHz (has to be an even
                multiple of crystal)
        //
        // According to the Philips LPC2294 manual:   M = cclk / Fosc
        where:    M    = PLL multiplier (bits 0-4 of PLLCFG)
        //

                        cclk = 58982400 Hz
        //

                        Fosc = 14745600 Hz
        //
        // Solving:       M = 58982400 / 14745600 = 4
        //
        //                          Note: M - 1 must be entered into bits 0-4
```

```
                of PLLCFG (assign 3 to these bits)
        //
        //
        // The Current Controlled Oscilator (CCO) must operate in the range
                156 mhz to 320 mhz
        //
        // According to the Philips LPC2106 manual:    Fcco = cclk * 2 * P
                where:    Fcco = CCO frequency
        //

                                    cclk = 58982400 Hz
        //

                                    P = PLL divisor (bits 5-6 of PLLCFG)
        //
        // Solving:        Fcco = 58982400 * 2 * P
        //                         P = 2  (trial value)
        //                         Fcco = 58982400 * 2 * 2
        //                         Fcc0 = 235929600 hz    (good choice for P
                since it's within the 156 mhz to 320 mhz range
        //
        // From Table 19 (page 48) of Philips LPC2106 manual    P = 2, PLLCFG
                bits 5-6 = 1  (assign 1 to these bits)
        //
        // Finally:      PLLCFG = 0  01  00011  =  0x23
        //
        // Final note: to load PLLCFG register, we must use the 0xAA followed
                0x55 write sequence to the PLLFEED register
        //              this is done in the short function feed() below
        //

        // Setting Multiplier and Divider values
        PLLCFG=0x23;
        feed();

        // Enabling the PLL */
        PLLCON=0x1;
        feed();

        // Wait for the PLL to lock to set frequency
        while(!(PLLSTAT & PLOCK)) ;

        // Connect the PLL as the clock source
        PLLCON=0x3;
        feed();

        // Enabling MAM and setting number of clocks used for Flash memory
fetch (4 cclks in this case)
        MAMCR=0x2;
        MAMTIM=0x4;

        // Setting peripheral Clock (pclk) to System Clock (cclk)
        VPBDIV=0x1;
}

/*********************************************************
                    feed
*********************************************************/

void feed(void)
{
  PLLFEED=0xAA;
  PLLFEED=0x55;
}
```

```
/*********************************************************
                        lcdInit
*********************************************************/

void lcdInit(void){
          //Initialisierung siehe LCD-Datenblatt

          IO0DIR |= RS|RW|E|DB0|DB1|DB2|DB3|DB4|DB5|DB6|DB7;
          IO0CLR = E;

          IO0CLR = RS|RW|DB7|DB6;
          IO0SET = DB5|DB4;
          enable();

          IO0CLR = RS|RW|DB7|DB6;
          IO0SET = DB5|DB4;
          enable();

          IO0CLR = RS|RW|DB7|DB6;
          IO0SET = DB5|DB4;
          enable();
          //Dreimal Interface auf 8-Bit setzen

          IO0CLR = RS|RW|DB7|DB6|DB2;
          IO0SET = DB5|DB4|DB3;
          enable();
          //Set Function:
          //DB4 = DL = High => 8-Bit-Interface
          //DB3 = N  = High => 2-Line-Display
          //DB2 = F  = Low  => 5x7 dots

          IO0CLR = RS|RW|DB7|DB6|DB5|DB4|DB1|DB0;
          IO0SET = DB3|DB2;
          enable();
          //Display On/Off:
          //DB2 = D = High => Display On
          //DB1 = C = Low  => Cursor Off
          //DB0 = B = Low  => Blinking Off

          IO0CLR = RS|RW|DB7|DB6|DB5|DB4|DB3|DB2|DB1;
          IO0SET = DB0;
          enable();
          //Display Clear

          IO0CLR = RS|RW|DB7|DB6|DB5|DB4|DB3|DB0;
          IO0SET = DB2|DB1;
          enable();
          //Entry Mode Set:
          //DB1 = I/D = Cursor bewegt sich nach rechts (Cursor increases)
          //DB0 = SH  = Diplay Shift
}

/*********************************************************
                        wait5ms
*********************************************************/

void wait5ms(){
          int time = T0TC;
          while((T0TC - time)< 294912){}
}

/*********************************************************
                        enable
```

```
*********************************************************/

void enable(void){
          IO0SET = E;
          wait5ms();
          IO0CLR = E;
          wait5ms();
}

/*********************************************************
                    writeChar
*********************************************************/

void writeChar(char c){
          IO0SET = RS;
          //Datenregister ausgewählt
          IO0CLR = RW;
          //Schreibzugriff ausgewählt
          int i;
          for(i=0;i<8;i++){
                    if(c&(1<<i))
                              IO0SET = (DB0<<i);
                    else
                              IO0CLR = (DB0<<i);
          }
          //c wird in DB0-7 geschrieben
          enable();
}

/*********************************************************
                    writeString
*********************************************************/

void writeString(char s[]){
          int i=0;
          while(s[i]!='\0'){
                    writeChar(s[i]);
                    i++;
          }
}

/*********************************************************
                    returnHome
*********************************************************/

void returnHome(void){
          //returnHome siehe LCD-Datenblatt
          IO0CLR = RS|RW|DB7|DB6|DB5|DB4|DB3|DB2;
          IO0SET = DB1;
          enable();
}

/*********************************************************
                    displayClear
*********************************************************/

void displayClear(void){
          //displayClear siehe LCD-Datenblatt
          IO0CLR = RS|RW|DB7|DB6|DB5|DB4|DB3|DB2|DB1;
          IO0SET = DB0;
          enable();
}

/*********************************************************
```

```
                        switchLED
*********************************************************/

void switchLED(void){
          if(IO0PIN&LED)
                    IO0CLR = LED;
          else
                    IO0SET = LED;
}

/*********************************************************
                        floatToString
*********************************************************/

void floatToString(float f, char voltage[12]){
          voltage[0] = 'A';
          voltage[1] = '/';
          voltage[2] = 'D';
          voltage[3] = '0';
          voltage[4] = ':';
          voltage[5] = ' ';
          voltage[6] = (char)f+48;
          voltage[7] = '.';
          voltage[8] = (char)((f-(float)(voltage[6]-48))*10.0)+48;
          voltage[9] = (char)((f-(float)(voltage[6]-48)-((float)
(voltage[8]-48))/10.0)*100.0)+48;
          voltage[10]= 'V';
          voltage[11]= '\0';
}

/*********************************************************
                        setDDAdress
*********************************************************/

void setDDAdress(void){
          //sedDDAdress siehe LCD-Datenblatt
          IO0CLR = RS|RW|DB5|DB4|DB3|DB2|DB1|DB0;
          IO0SET = DB7|DB6;
          enable();
}

/*********************************************************
                        output
*********************************************************/

void output(char spannungLED[]){
          displayClear();
          setDDAdress();
          if(IO0PIN&LED)
                    writeString("LED aus!");
          else
                    writeString("LED an!");
          returnHome();
          writeString(spannungLED);

}

/*********************************************************
                        canInit
*********************************************************/

void canInit(void){
          C2MOD = 0x0;
```

```
}

/*  Stubs for various interrupts (may be replaced later)  */
/*  ----------------------------------------------------  */

void IRQ_Routine (void) {
          while (1) ;
}

void FIQ_Routine (void)  {
          while (1) ;
}

void SWI_Routine (void)  {
          while (1) ;
}

void UNDEF_Routine (void) {
          while (1) ;
}
```