```
#include <string.h>
#include <avr/eeprom.h>
#include <stdlib.h>
#include <stdio.h>
#include <avr/io.h>
#include <stdint.h>
#include <avr/interrupt.h>
#include <avr/signal.h>
#include <util/delay.h>
#include <inttypes.h>
#include <avr/wdt.h>

#include "lcd.h"

// ----------------------------------------
// Definitionen

#ifndef F_CPU
#define F_CPU           4000000                 // Festlegen des CPU-Takts
#endif

#define TRUE 1
#define FALSE 0

#define MODUS 1
#define FENSTERAUSWAHL 2
#define STEUERUNG 3

#define LINKS 1
#define RECHTS 2
#define OBEN 3
#define UNTEN 4
#define ENTER 5
#define KEINETASTE 6

//Definitionen zur Taster-Entprellung

#define KEY_PORT        PORTD
#define KEY_PIN         PIND
#define TASTE0            0
#define TASTE1            1
#define TASTE2            2
#define TASTE3            3
#define TASTE4            4
#define ALL_KEYS        (1<<TASTE0 | 1<<TASTE1 | 1<<TASTE2 | 1<<TASTE3 | 1<<TASTE4)

#define REPEAT_MASK     (1<<TASTE0 | 1<<TASTE1 | 1<<TASTE2 | 1<<TASTE3 | 1<<TASTE4)
   // repeat: key1, key2
#define REPEAT_START    50                              // after 500ms
#define REPEAT_NEXT     20                              // every 200ms

#define VERZUGSZEIT     200
#define TIPPZEIT        500                             // TIPPZEIT * 100

// ----------------------------------------
// Festlegung der globalen Variablen:

uint8_t key_state;                                      // debounced and inverted key s
tate:
                                                        // bit = 1: key pressed
uint8_t key_press;                                      // key press detect

uint8_t key_rpt;                                        // key long press and repeat

uint8 t messwert, vergleichsWert, ausschaltWertAlle ,ausschaltWertEinzel;       // Wert
e für den AD-Wandler                                  // Messwert vom AD-Wandler

int fensterwahl, jalousiewahl;                          // Auswahl für das Menue MODUS
int fenster1, fenster2, fenster3, fenster4;             // Auswahl für das Menue FENSTE
RWAHL

int spalteModus, spalteFensterwahl, spalteSteuerung;
```

```
int aktivesMenue, aktiveTaste, stopTaste, zaehler, i, tippen, automatik, adWandler;

// init(): Startwerte usw. setzen

void init()
{
    // Globale Variablen setzen:
    fensterwahl     = FALSE;
    jalousiewahl    = FALSE;
    fenster1        = FALSE;
    fenster2        = FALSE;
    fenster3        = FALSE;
    fenster4        = FALSE;


    //setze beim Start auf Menue MODUS
    aktivesMenue = MODUS;
    aktiveTaste = KEINETASTE;
    stopTaste = FALSE;
    tippen =  FALSE;

    //Alle Spalten auf 0 setzen (linke Auswahl)
    spalteModus= 0;
    spalteFensterwahl = 0;
    spalteSteuerung = 0;

    zaehler = 0;

    //initalisierung des Displays
    lcd_init(LCD_DISP_ON_CURSOR);

    //LCD auf Spalte 0 und Zeile 1 setzen!
    lcd_gotoxy(0,1);

    //LCD MenueModus zeichnen
    lcdMenue();

    //Definiton der Ein-/Ausgänge (0x00 = Eingang, 0xff = Ausgang)
    DDRA = 0xfe;                    // Setzt PA auf Ausgang bis aufPA0 auf Eingang gese
tzt (AD-Wandler)
    //DDRA &= ~((1<<PA0));          // PA0 auf Eingang gesetzt (AD-Wandler)
    //DDRB = 0x00;
    //DDRC = 0xff;
    DDRD = 0x00;


    // Timer-Interrupt
    TCCR0 = (1 << CS02)| (1 << CS00);       // Systemtakt/1024
    TIMSK = (1 << TOIE0) | (1 << TOIE1);;   // Timer0 und Timer1 Interrupt frei
    TCNT1 = 0x0000;                         // Startwert auf 0

    // AD-Wandler
    ADMUX = 0x60;                                   // VREF=VCC; ADLAR schreibt die 8 hö
herwertigen Bits in das ADHC
    ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS0) | (1<<ADATE) | (1<<ADSC); // Frequenzt
eiler (ADPS0,ADPS2) setzen: hier 32 cpu takt/32=125 kHz
    vergleichsWert = 0x0A;                          // Schwelle bei ca. 0,2V
    ausschaltWertAlle = 0xC8;                       // Max. Wert wegen 800/1024*5V

    ausschaltWertEinzel = 0x32;                 // Achtung: 8-bit und nicht 10-bit
                                                // ADATE: Messung begint bei prositiver
 Flanke

    sei();                                          // Freischaltung von Interrupts

    //Alle Ausgänge auf LOW setzen
    PORTA = 0x00;
    PIND = 0xff;

}

// ----------------------------------------
// SPEZIELLE FUNKTIONEN
```

```
void lcdMenue()
{
    lcd_clrscr();                           // löscht das LCD

    lcd gotoxy(0,0);                        // setzt Cursor an Spalte = 0, Zeile = 1
    lcd_puts_p(PSTR("Bitte waehlen:\n"));   // schreibt ersten String

    lcd_puts_p(PSTR("Fenster Jalousie"));   // schreibt zweiten String

    if ( spalteModus == 0 )
        lcd gotoxy(0,1);                    // setzt Cursor an Spalte = 0, Zeile = 1

    else
        lcd_gotoxy(8,1);

    lcd_command(LCD_DISP_ON_BLINK);         // Cursor blinkt
}

void lcdFenster()
{
    lcd_clrscr();

    lcd_puts_p(PSTR("Fenster waehlen:\n"));

    lcd_puts_p(PSTR("Alle F1 F2 F3 F4"));

    switch (spalteFensterwahl)
    {
    case 0: lcd gotoxy(0,1);
            break;
    case 1: lcd gotoxy(5,1);
            break;
    case 2: lcd gotoxy(8,1);
            break;
    case 3: lcd gotoxy(11,1);
            break;
    case 4: lcd gotoxy(14,1);
            break;
    }

    lcd_command(LCD_DISP_ON_BLINK);

}

void lcdJalousie()
{
    lcd_clrscr();

    lcd_puts_p(PSTR("Jalousie wahl:\n"));

    lcd_puts_p(PSTR("Alle J1 J2 J3 J4"));

    switch (spalteFensterwahl)
    {
    case 0: lcd gotoxy(0,1);
            break;
    case 1: lcd gotoxy(5,1);
            break;
    case 2: lcd gotoxy(8,1);
            break;
    case 3: lcd gotoxy(11,1);
            break;
    case 4: lcd gotoxy(14,1);
            break;
    }
    lcd_command(LCD_DISP_ON_BLINK);

}

void lcdSteuerung()
{
    lcd_clrscr();
```

```
    lcd_puts_p(PSTR("Aktion waehlen:\n"));

    lcd_puts_p(PSTR(" AUF       ZU "));

    switch (spalteSteuerung)
    {
    case 0: lcd gotoxy(1,1);
            break;
    case 1: lcd gotoxy(11,1);
            break;
    }

    lcd_command(LCD_DISP_ON_BLINK);

}

void setzeFensterBits()
{
    if ( fenster1 == TRUE )
        {
             delay ms(VERZUGSZEIT);
            PORTA |= (1 << PA3);
        }
    if ( fenster2 == TRUE )
        {
             delay ms(VERZUGSZEIT);                  // Einschaltverrögerung
            PORTA |= (1 << PA4);
        }
    if ( fenster3 == TRUE )
        {
             delay ms(VERZUGSZEIT);                  // Einschaltverrögerung
            PORTA |= (1 << PA5);
        }
    if ( fenster4 == TRUE )
        {
             delay ms(VERZUGSZEIT);                  // Einschaltverrögerung
            PORTA |= (1 << PA6);
        }
}

void loescheFensterBits()
{
    if ( fenster1 == TRUE )
        {
             delay ms(VERZUGSZEIT);
            PORTA &= ~(1 << PA3);
        }

    if ( fenster2 == TRUE )
        {
             delay ms(VERZUGSZEIT);
            PORTA &= ~(1 << PA4);
        }

    if ( fenster3 == TRUE )
        {
             delay ms(VERZUGSZEIT);
            PORTA &= ~(1 << PA5);
        }

    if ( fenster4 == TRUE )
        {
             delay ms(VERZUGSZEIT);
            PORTA &= ~(1 << PA6);
        }
}

void ausschaltReset()
{
    ADCSRA = 0x00;                      // ADC auschalten
    TCCR1B = 0x00;                      // Timer1 auschalten
    TCNT1H = 0x00;                      // Timerregister setzt H-Reg auf 0
```

```
    TCNT1L = 0x00;                      // Timerregister setzt L-Reg auf 0
    adWandler = FALSE;
    automatik = FALSE;
    stopTaste = FALSE;
}
// ----------------------------------------
// ZEITFUNKTIONEN

void timerTipp()                        // Tippfunktion
{
     delay ms( TIPPZEIT );
    loescheFensterBits();
}


ISR ( TIMER1 OVERFLOW_vec)              // nach einer Zeit t=120s sollen die Ausgänge z
urückgesetzt werden
{
    zaehler++;
    if(zaehler == 6)
        {
            loescheFensterBits();
            ausschaltReset();
            zaehler = 0;
        }
}


// ----------------------------------------
// STROM-/SPANNUNGSMESSUNG
ISR ( SIG_ADC )
{
    messwert = ADCL;
    messwert = ADCH;
    if (messwert >= vergleichsWert)
        {
        adWandler = TRUE;
        }
    else if ( (adWandler == TRUE) && (messwert == 0x0000) )
                {
                loescheFensterBits();
                ausschaltReset();
                }
    else if ( (messwert > ausschaltWertAlle)  && spalteFensterwahl == 0 )
                {
                loescheFensterBits();
                if (spalteSteuerung == 0)
                    {
                    PORTA |= (1 << PA2);
                    setzeFensterBits();
                    timerTipp();
                    loescheFensterBits();
                    PORTA &= ~(1 << PA2);
                    }
                else
                    {
                    PORTA &= ~(1 << PA2);
                    setzeFensterBits();
                    timerTipp();
                    loescheFensterBits();
                    PORTA |= (1 << PA2);
                    }
                ausschaltReset();
                }
    else if ( messwert > ausschaltWertEinzel )
                {
                loescheFensterBits();
                if (spalteSteuerung == 0)
                    {
                    PORTA |= (1 << PA2);
                    setzeFensterBits();
                    timerTipp();
                    loescheFensterBits();
```

```
                    PORTA &= ~(1 << PA2);
                    }
                else
                    {
                    PORTA &= ~(1 << PA2);
                    setzeFensterBits();
                    timerTipp();
                    loescheFensterBits();
                    PORTA |= (1 << PA2);
                    }
                ausschaltReset();
                }

}

// ----------------------------------------
// TASTERENTPRELLEN


ISR( TIMER0_OVF_vect )                          // every 10ms
{
  static uint8_t ct0, ct1, rpt;
  uint8_t i;

  TCNT0 = (uint8_t)(int16_t)-(F_CPU / 1024 * 10e-3 + 0.5);  // preload for 10ms

  i = key state ^ ~KEY PIN;                     // key changed ?
  ct0 = ~( ct0 & i );                           // reset or count ct0
  ct1 = ct0 ^ (ct1 & i);                        // reset or count ct1
  i &= ct0 & ct1;                               // count until roll over ?
  key state ^= i;                               // then toggle debounced state
  key_press |= key_state & i;                   // 0->1: key press detect

  if( (key state & REPEAT MASK) == 0 )          // check repeat function
     rpt = REPEAT START;                        // start delay
  if( --rpt == 0 ){
    rpt = REPEAT NEXT;                          // repeat delay
    key_rpt |= key_state & REPEAT_MASK;
  }
}


uint8_t get_key_press( uint8_t key_mask )
{
  cli();                                        // read and clear atomic !
  key mask &= key press;                        // read key(s)
  key press ^= key_mask;                        // clear key(s)
  sei();
  return key_mask;
}

uint8_t get_key_rpt( uint8_t key_mask )
{
  cli();                                        // read and clear atomic !
  key mask &= key rpt;                          // read key(s)
  key rpt ^= key_mask;                          // clear key(s)
  sei();
  return key_mask;
}


uint8_t get_key_short( uint8_t key_mask )
{
  cli();                                        // read key state and key press atomi
c !
  return get_key_press( ~key_state & key_mask );
}


uint8_t get_key_long( uint8_t key_mask )
{
  return get_key_press( get_key_rpt( key_mask ));
}
```

```
// ----------------------------------------
// MENUEUNTERFUNKTIONEN

void MenueModus(int Taste)
{
    switch(Taste)
    {
        case LINKS  :
                    if (spalteModus == 1)
                        {
                            spalteModus = 0;
                            lcd_gotoxy(0,1);                    // LCD Cursor vor. Pos
                        }
                    else if (spalteModus == 0)
                    {
                        spalteModus = 1;
                        lcd_gotoxy(8,1);                        // LCD Cursor nä. Pos
                    }
                    else break;

                    break;
        case RECHTS :
                    if (spalteModus == 0)
                        {
                            spalteModus = 1;
                            lcd_gotoxy(8,1);                        // LCD Cursor nä. P
os
                        }
                    else if (spalteModus == 1)
                    {
                        spalteModus = 0;
                        lcd_gotoxy(0,1);                        // LCD Cursor nä. P
os
                    }
                    else break;

                    break;
        case OBEN   :
                    break;
        case UNTEN  :
                    if (fensterwahl || jalousiewahl)
                        {
                            aktivesMenue = FENSTERAUSWAHL;
                            if ( fensterwahl == TRUE )
                                {
                                    PORTA |= ( 1<<PA1 );            // Bit A1 setzen
                                    lcdFenster();                   // Fenster aufrufen
 (LCD)
                                }
                            else
                                {
                                    PORTA &= ~( 1<<PA1 );           // Bit A1 löschen
                                    lcdJalousie();
                                }
                        }
                    break;
        case ENTER  :
                    switch(spalteModus)
                    {
                        case 0 :
                                fensterwahl  = TRUE;
                                jalousiewahl = FALSE;
                                break;
                        case 1 :
                                fensterwahl  = FALSE;
                                jalousiewahl = TRUE;
                    }

                    fenster1 = FALSE;
                    fenster2 = FALSE;
                    fenster3 = FALSE;
                    fenster4 = FALSE;
```

```
                    aktivesMenue = FENSTERAUSWAHL;
                    spalteFensterwahl = 0;

                    if ( fensterwahl == TRUE )
                        {
                            PORTA |= ( 1<<PA1 );      // Bit PA0 setzen
                            lcdFenster();             // Fenster aufrufen (LCD)
                        }
                    else
                        {
                            PORTA &= ~ ( 1<<PA1 );    // Bit PA0 löschen
                            lcdJalousie();
                        }
                        break;
    }
}

void MenueFensterauswahl(int Taste)
{
    switch(Taste)
    {
        case LINKS  :
                    if ( spalteFensterwahl == 0 )
                        {
                            spalteFensterwahl = 4;
                            lcd_gotoxy(14,1);
                        }
                    else if ( spalteFensterwahl == 1 )
                        {
                            spalteFensterwahl = 0;
                            lcd_gotoxy(0,1);
                        }

                    else if ( spalteFensterwahl == 2 )
                        {
                            spalteFensterwahl = 1;
                            lcd_gotoxy(5,1);
                        }

                    else if ( spalteFensterwahl == 3 )
                        {
                            spalteFensterwahl = 2;
                            lcd_gotoxy(8,1);
                        }

                    else if ( spalteFensterwahl == 4 )
                        {
                            spalteFensterwahl = 3;
                            lcd_gotoxy(11,1);
                        }
                    else break;

                    break;
        case RECHTS :
                    if ( spalteFensterwahl == 0 )
                        {
                            spalteFensterwahl = 1;
                            lcd_gotoxy(5,1);
                        }

                    else if ( spalteFensterwahl == 1 )
                        {
                            spalteFensterwahl = 2;
                            lcd_gotoxy(8,1);
                        }

                    else if ( spalteFensterwahl == 2 )
                        {
                            spalteFensterwahl = 3;
                            lcd_gotoxy(11,1);
                        }

                    else if ( spalteFensterwahl == 3 )
```

```
                        {
                            spalteFensterwahl = 4;
                            lcd_gotoxy(14,1);
                        }
                    else if ( spalteFensterwahl == 4 )
                        {
                            spalteFensterwahl = 0;
                            lcd_gotoxy(0,1);
                        }
                    else break;

                    break;
        case OBEN   :
                    aktivesMenue = MODUS;
                    lcdMenue();
                    break;
        case UNTEN  :
                    if ( spalteSteuerung >= 0 && spalteSteuerung <5 );
                    {
                    aktivesMenue = STEUERUNG;
                    lcdSteuerung();
                    }
                    break;
        case ENTER  :
                    switch(spalteFensterwahl)
                    {
                        case 0 :
                                fenster1 = TRUE;
                                fenster2 = TRUE;
                                fenster3 = TRUE;
                                fenster4 = TRUE;


                                break;
                        case 1 :
                                fenster1 = TRUE;
                                fenster2 = FALSE;
                                fenster3 = FALSE;
                                fenster4 = FALSE;

                                break;
                        case 2 :
                                fenster1 = FALSE;
                                fenster2 = TRUE;
                                fenster3 = FALSE;
                                fenster4 = FALSE;

                                break;
                        case 3 :
                                fenster1 = FALSE;
                                fenster2 = FALSE;
                                fenster3 = TRUE;
                                fenster4 = FALSE;

                                break;
                        case 4 :
                                fenster1 = FALSE;
                                fenster2 = FALSE;
                                fenster3 = FALSE;
                                fenster4 = TRUE;
                    }
                    aktivesMenue = STEUERUNG;
                    spalteSteuerung == 0;
                    lcdSteuerung();
                    break;
    }
}

void MenueSteuerung(int Taste)
{
    switch(Taste)
    {
        case LINKS  :
```

```
                    if (spalteSteuerung == 1)
                        {
                            spalteSteuerung = 0;
                            lcd_gotoxy(1,1);
                        }
                    else if (spalteSteuerung == 0)
                        {
                            spalteSteuerung = 1;
                            lcd_gotoxy(11,1);
                        }
                    break;
        case RECHTS :
                    if (spalteSteuerung == 0)
                        {
                            spalteSteuerung = 1;
                            lcd_gotoxy(11,1);
                        }
                    else if (spalteSteuerung == 1)
                        {
                            spalteSteuerung = 0;
                            lcd_gotoxy(1,1);
                        }
                    break;
        case OBEN   :
                    aktivesMenue = FENSTERAUSWAHL;
                    if ( fensterwahl == TRUE )
                        lcdFenster();
                    else
                        lcdJalousie();
                    break;
        case UNTEN  :
                    break;
        case ENTER  :
                    switch(spalteSteuerung)
                    {
                        case 0 :
                                PORTA &= ~(1 << PA2);               // Richtungbit: AUF

                                if ( tippen == TRUE )               // Tippfunktion
                                {
                                    setzeFensterBits();
                                    timerTipp();
                                }
                                else                                // Autofunktion
                                {
                                    setzeFensterBits();
                                    //TCCR1B = (1 << CS12)| (1 << CS10);        //
                                    ADCSRA = 0xed;                  // ADC einschalten
                                    sei();
                                    stopTaste = TRUE;
                                }

                                break;
                        case 1 :
                                PORTA |= (1 << PA2);                // Richtungsbit: ZU

                                if ( tippen == TRUE )               // Tippfunktion
                                {
                                    setzeFensterBits();
                                    timerTipp();
                                }
                                else                                // Autofunktion
                                {
                                    setzeFensterBits();
                                    //TCCR1B = (1 << CS12)| (1 << CS10);
                                    ADCSRA = 0xed;                  // ADC einschalten
                                    sei();
                                    stopTaste = TRUE;
                                }
                                break;

                    }
```

```
    }
}

// ----------------------------------------

void main(void)
{
    init();     //Achtung LCD-Ports müssen noch umgestellt werden!!!!!

    while(TRUE)
    {

            if ( get key press(1 << TASTE0) ) {aktiveTaste = RECHTS;}
            if ( get key press(1 << TASTE1) ) {aktiveTaste = LINKS; }
            if ( get key press(1 << TASTE2) ) {aktiveTaste = OBEN;  }
            if ( get_key_press(1 << TASTE3) ) {aktiveTaste = UNTEN; }

            if ( (aktivesMenue == STEUERUNG) && ( (fensterwahl == TRUE) || (jalousiewah
l == TRUE) ) )
            {
                switch (automatik)
                {

                case FALSE: if ( get_key_short( 1 << TASTE4) && stopTaste == FALSE )
                            {
                                tippen = TRUE;
                                automatik = FALSE;
                                aktiveTaste = ENTER;
                            }
                            else if ( get_key_long( 1 << TASTE4) && automatik == FALSE)

                            {
                                tippen = FALSE;
                                automatik = TRUE;
                                aktiveTaste = ENTER;
                            }
                            break;

                case TRUE:  if ( get_key_press( 1 << TASTE4) )
                            {
                                loescheFensterBits();
                                ausschaltReset();   // AD-Wandler auschalten
                            }
                            break;
                }
            }
            else if (get key press( 1 << TASTE4))
                     {aktiveTaste = ENTER; }



            //Menuezuweisung
            switch(aktivesMenue)
            {
                case MODUS :
                            MenueModus(aktiveTaste);
                            break;
                case FENSTERAUSWAHL :
                            MenueFensterauswahl(aktiveTaste);
                            break;
                case STEUERUNG:
                            MenueSteuerung(aktiveTaste);
            }


            aktiveTaste = KEINETASTE;

            wdt reset();        //Watchdog reseten


    }
```

```
}
```