```
/*
*
*Description:                Empfaengt von USART
*                          und gibt Daten auf TWI aus.
*
*                          Datum   29.03.2006
                             abgespeckt nur noch Watchdog
*****************************************************************************/
#include <ioavr.h>
#include <inavr.h>

               //Einstellungen für MUX und Quelle
#define VOLT_LADE  ((0 << MUX0) | (0 << MUX1) | (0 << MUX2) | (0 << MUX3) | \
                  (0 << REFS1) | (1 << REFS0) | (1 << ADLAR)) //ADC0
#define VOLT_EINGANG  ((1 << MUX0) | (0 << MUX1) | (0 << MUX2) | (0 << MUX3) | \
                  (0 << REFS1) | (1 << REFS0) | (1 << ADLAR)) //ADC1

unsigned int ADC_WERT;            //für Spannungsüberwachung
unsigned int x = 0;               //für Schleifen
unsigned int i,j;                 //für Zählaufgaben
unsigned int blink = 0;           //Als Zähler für Blinklampe
unsigned int blink_count = 0;     //Zähler für Blinkfrequenz
unsigned char LADESPANNUNG = 0;   //Netzteil oder Booster liefert Ladespannung


/* ACCU_LOAD line macros. */
#define ACCU_LOAD PD2 //!< The pin used as  ACCU_LOAD ON.
#define DDIR DDRD //!< The data direction register for the ACCU_LOAD ON line.
#define DPORT PORTD //!< The port register for the ACCU_LOAD ON.
#define SETDDIR DDIR |= (1<<ACCU_LOAD); //!< Set output direction for ACCU_LOAD.
#define SETACCU_LOAD DPORT |= (1<<ACCU_LOAD); //!< Set ACCU_LOAD ON high.
#define CLRACCU_LOAD DPORT &= ~(1<<ACCU_LOAD); //!< Set ACCU_LOAD ON low.

// Hier die Voraussetzung für ein Synchronsignal NUR FUER TESTZWECKE!!
#define SYNC PB3 //Signal MOSI
#define BDIR DDRB //Datenrichtungsregister B
#define BPORT PORTB //Portregister B
#define SETBDIR BDIR |= (1<<SYNC); //Set direction fuer PB3
#define SETSYNC BPORT |= (1<<SYNC); //Set Pin PB3
#define CLRSYNC BPORT &= ~(1<<SYNC); //Clesr Pin PB3

#define LADE PB4 //Signal MISO *******LED 1************später PIN PB2
#define LADEDIR BDIR |= (1<<LADE); //Set direction fuer PB4 Ausgang
#define SETLADE_LED BPORT |= (1<<LADE); //Set Pin PB4
#define CLRLADE_LED BPORT &= ~(1<<LADE); //Clesr Pin PB4


//Initialize USART, TWI, ADC and PORTC
    void init_usart (void)
    {
       //! Initialize ADC module.
        ADCSRA = (1 << ADEN) | (1 << ADPS1); // Enable ADC, clock prescaler = 4
    }


   void get_adc ()
        {
             ADCSRA |= (1 <<ADSC);   //lese Spannung/10
            //Wait for ADC conversion
                do {} while (ADCSRA & (1 << ADSC));
        }
```

```
   void wdt_init ()
  {
            //Watchdog - Einstellungen

            WDTCSR |= (1<<WDCE) | (1<<WDE);
            WDTCSR = (1<<WDE) | (1<<WDP1) | (1<<WDP2); // 1 Sekunde

  }

   void wdt_off ()
   {
           __disable_interrupt ();
           __watchdog_reset ();
             MCUSR = 0x00;
             WDTCSR |= (1<<WDCE) | (1<<WDE);
             WDTCSR = 0x00;
           __enable_interrupt ();
   }

      void main()
{
       wdt_init ();   //Initialisiere Watchdog

       unsigned char ACCU_LEER = 52;   // diese Zuweisungen sind noch nicht
       unsigned char ACCU_VOLL = 66;   // endgültig. Sie beziehen sich auf
       unsigned char ACCU_HALB = 58;   // den im Test verwendeten NIMH-Accu

       SETDDIR;       //Set PD2 at output für ACCU_LOAD
       SETBDIR;       //Set PB3 to output für Syncronsignal nur für Oszilloskop
       LADEDIR;       //Set PB4 für Ladekontrolle


  // Loop forever. Hauptschleife
       for(;;)

  {
        ADMUX = VOLT_EINGANG;    //Die Spannung vor dem FET messen
        get_adc ();              //um festzustellen, ob ein Ladegerät
        LADESPANNUNG = ADCH;     //angeschlossen ist

        ADMUX = VOLT_LADE;      //Muxeinstellung für ACCU Spannungsmessen
        get_adc ();             //ADC lesen der Accuspannung

        if (LADESPANNUNG > ADCH) LADESPANNUNG = 1; else LADESPANNUNG = 0;
        if (LADESPANNUNG == 0)
   {                   //Wenn kein Ladegerät angeschlossen ist
     CLRACCU_LOAD;     //hat laden keinen Sinn, also aus

     if (ADCH < ACCU_LEER) // Accu ist ganz leer
    {
       //hier schaltet der Prozessor alles aus und legt sich schlafen
       wdt_off ();
       SMCR |= (1<<SE) | (1<<SM1);    //Vorbereitung für power down
      __sleep ();           // Gute Nacht

    }
       //Wenn der Accu noch nicht ganz leer ist, aber warnen
      if (ADCH < ACCU_LEER +4) blink = 1300; else blink = 0;   //(26)
}
else        //Hier geht's weiter, wenn Ladespannung vorhanden ist
{
     if (ADCH <= ACCU_HALB) //Wenn der Accu weniger als halb voll ist
```

```
     {                      //sollte die Ladung immer beginnen
     SETACCU_LOAD;                  //Laden einschalten
     blink = 10000;                 //LED blinkt langsam    (200)
     } else if (blink == 10000)     //wenn Accu schon geladen wird
       { if (ADCH >= ACCU_VOLL)     //Erst wenn ACCU voll ist
        {                           //kann abgeschaltet werden
        blink = 0;                  //nicht mehr blinken
        CLRACCU_LOAD;               //auch nicht mehr laden
        }
       }
}
             //hier wird geblinkt, schnell, langsam oder gar nicht
    if (blink_count >= blink) blink_count = 0;    //Zähler reset
    if (blink_count > (blink / 2)) SETLADE_LED;   //LED ein
    if (blink_count <= (blink / 2)) CLRLADE_LED;  //LED aus

        blink_count++;   //Zähler der Schleifen Durchläufe


         //Der watchdogzähler muß gelöscht werden, wenn Programm OK
      __watchdog_reset();




}
  }


 //end of file
```