Dokumentation

Arduino-Projekt

Elektronisches Info Management für einen Citroen 2CV-Elektro

Dipl.Ing Michael Strnad, 8160 Weiz, Austria
November 2012

Inhaltsverzeichnis

Danksagung:
Dass dieses Projekt zustande gekommen ist, dafür möchte ich mich bei allen ganz herzlich bedanken, die mir mit Rat und Tat weitergeholfen, die meine Abwesenheit geduldig ertragen und mich mit manchen wichtigen Tipps versorgt haben. Ohne sie hätte ich nicht die Ausdauer und Energie aufgebracht.

1. Vorbemerkung:

2007 hat es an der Höheren Technischen Bundeslehranstalt (HTBLA) in Weiz ein Maturaprojekt gegeben, das zum Ziel hatte, einen Citroen 2CV, die legendäre „Ente“, auf Elektroantrieb umzubauen. Um den Maturanten die Möglichkeit zu geben, an einem realitätsnahen Projekt ihre praktischen und theoretischen Kenntnisse zu vertiefen und unter Beweis zu stellen.

Nach Vorstellung des Projektes durch Prof. DI Karl Haar und kurzer Suche haben sich auch sofort 6 Maturanten gefunden, die dieses Projekt unter Mithilfe des Lehrkörpers und der Recourcen der HTBLA in wenigen Monaten durchgeführt haben.

Dazu gibt es reichliche Dokumentation, auf die ich hier auch verweise. Gleichzeitig möchte ich aber auch die Gelegenheit wahrnehmen, der Schule, den Professoren und nicht zuletzt den 6 Maturanten für die hervorragende Leistung und die Professionalität zu gratulieren und mich auch herzlich dafür zu bedanken. Bin ich doch jetzt Nutznießer dieses Einsatzes!!

Danach ist die Ente einige Zeit, erst in der Schule, dann bei Prof. Haar, gestanden, bis ich sie dann im Februar 2011 kaufen konnte. Optimierungen, die in der Zeit der Maturarbeit nicht vorgenommen werden konnten, haben bei den folgenden Maturajahrgängen keine Interessen geweckt.

Da es aber von Anfang an mein erklärtes Ziel war, dieses tolle Fahrzeug nicht nur in der Garage zu haben, sondern es auch im täglichen Leben gebührend einzusetzen, stand ich vor der Aufgabe, mich in die Materie zu vertiefen und zu lernen. Hatte ich doch seit der Studentenzeit, meine Wegbegleiter waren damals ein Puchroller und später eine Vespa, eigentlich mit „Autoschrauben“ nichts am Hut!

Natürlich, damals kannte ich jeden Teil meiner Fahrzeuge, doch inzwischen sind diese mechatronischen Wunderdinge so komplex geworden, dass auch ein einfacher Eingriff für mich nicht mehr sinnvoll war.

Also lernen lernen lernen: Bücher kaufen, im Internet suchen, Freunde und Spezialisten fragen und letztlich probieren, studieren, probieren, studieren!

Die mechanische Arbeit an der Ente war von einigen Rückschlägen begleitet, letztlich aber doch von Erfolg gekrönt. Im September 2011 haben wir dann die Enzelgenehmigungsprüfung für die Ente bestanden und Anfang Oktober 2011 ein „Entedankfest“ mit der Ente und mit vielen Freunden gefeiert.

Nach der Winterruhe, die Blei-Säure-Akkus sind alt und verlieren bei Kälte viel Kapazität, hat mich der Wunsch, auch noch über die elektrische Seite des Autos Kenntnis zu erlangen, auf die Spur des Arduino-Projektes geführt. Mein Wunsch, die Werte des Verbrauches und des Zustandes sichtbar zu machen, haben mich das ganze Jahr beschäftigt. Nicht ununterbrochen, man hat ja auch noch andere Sachen zu tun, innerlich aber beständig!

Das Arduino-Projekt an sich vorzustellen, wird hier nicht nötig sein, für näher Interessierte hier ein paar Links: www.arduino.cc arduino.org de.wikipedia.org/wiki/Arduino-Plattform www.arduino-tutorial.de/ www.golem.de/specials/arduino/ arduinoprojects101.com/ www.arduino.de www.erik-bartmann.de ohne Anspruch auf Vollständigkeit!

Soweit nur etwas allgemein: Arduino ist ein Open Source Projekt und arbeitet mit Microkontrollern. Microkontroller sind eine Art kleine, programmierbare Computer, die sich vor allem durch ihre Vielzahl an digitalen und analogen Ein- und Ausgängen und an der leichten Verbindbarkeit mit PC etc. auszeichnen.

In meinem Fall war das so, dass ich, nach intensiver Suche, auf dem breiten Feld der Microkontroller, bald auf eben dieses Arduino-Projekt aufmerksam wurde und mir einen „Arduino Mega“ kaufte. Und es begann wieder das Lernen: Programmiert hab ich in meinem Leben schon viel, allerdings fachspezifisch und haupsächlich in Basic. Hier war C++ gefragt. Na ja, heute würde ich sagen, das war nicht die größte Hürde, vielmehr machte mir mein Nichtwissen über Elektrotechnik und Elektronik zu schaffen!

In der Skizze weiter unten hab ich die ungefähre Lage der plazierten Sensoren dargestellt. Diese leiten ihre Messwerte über drei- bis vieradrige Kabel zum Arduino, der am Amaturenbrett, neben einer Halterung fürs Handy befestigt ist. Die analogen Sensoren, wie Temperatur, Stromstärke und Spannung, liefern ihre Messspannung an den analogen Eingängen des Arduino ab und werden dort, entsprechend ihrer Anforderung, auf einem kleinen Farb-Touch-Screen dargestellt. Die beiden Hall-Sensoren für die Drehzahl des Motors und der linken Antriebsachse (für die Geschwindigkeit) liefern ihre Signale, Low and High, auf digitalen Eingängen ab, die dort einen Interrupt erzeugen. Die Interrupts werden gezählt und durch die Anzahl der Schrauben dividiert, die an dieser Achse liegen und deren Durchgänge eben vom Hallsensor registriert werden. Beim Motor gibt das dann sofort die Drehzahl in Umdrehungen pro Minute. Bei der Achse wird das dann noch mit dem Umfang des Rades multipliziert und gibt so die Kilometer pro Stunde als Geschwindigkeit des Fahrzeuges.

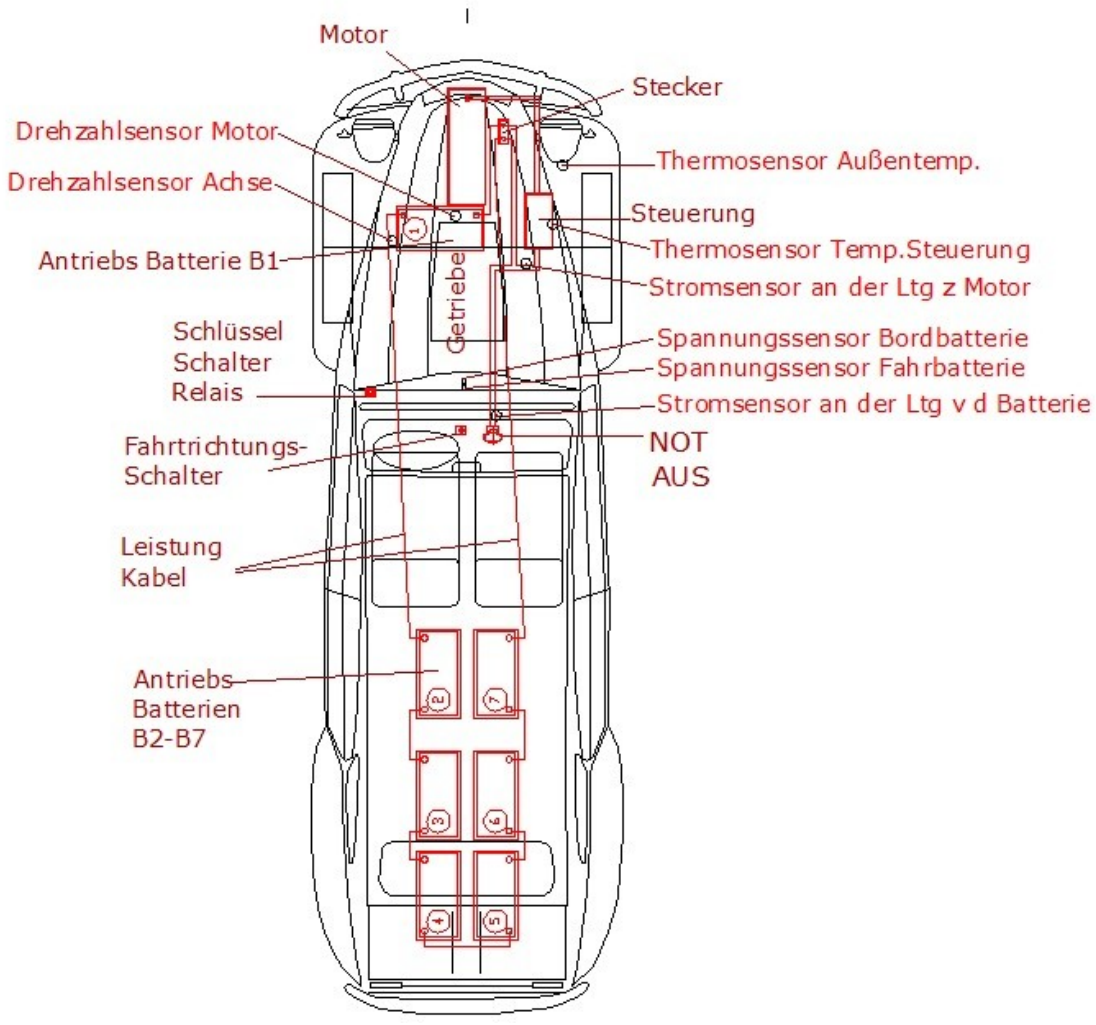


Abb. 1 Lage der Sensoren

2. Bauteile

Thermosensoren

LM35 DZ, www.national.com/pf/LM/LM35.html

Versorgungsspannung:	5V
Bereich:	– 55 bis 150 Grad C
Signalausgang:	10mV / °C

Der Sensor ist ein temperaturempfindlicher Widerstand und liefert eine Ausgangsspannung, die linear proportional zur Temperatur ist.

Ein Sensor misst die Außentemperatur, der andere ist an der Steuerung angebracht. Diese ist intern so geregelt, dass sie die Leistung reduziert, wenn die Steuerung 80 °C erreicht. Es ist also wichtig zu wissen, wann die kritische Temperatur erreicht wird.

Spannungssensoren

CYVT02-32S1-0.5-U90V www.chenyang-gmbh.com

Versorgungsspannung:	12 V
Bereich	0 – 90 V
Signalausgang:	0 – 5 V

Ein Sensor liegt an der Leitung von der Fahrbatterie (84 V), misst also die Spannung der Antriebsbatterie, der zweite Sensor liegt an der Bordbatterie (12 V). Sie sollen ein wenig Hinweise geben, wie viel Kapazität die Batterien noch haben. Das ist nicht ganz exakt, denn mit stärkerer Stromentnahme sinkt die Spannung und auch mit der Temperatur, doch hoffe ich, mit der Zeit ein wenig das Gefühl zu bekommen, wie viel noch da ist.

Stromsensoren

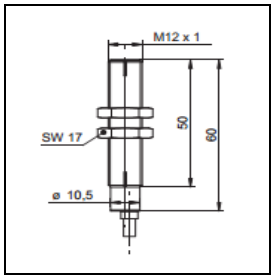
CYHCT-C2TV-U150A-32 www.chenyang-gmbh.com

Versorgungsspannung:	12 V
Bereich	0 – 150 A
Signalausgang:	0 – 5 V

Ein Sensor liegt an der Leitung der Fahrbatterie, mit dem anderen wollte ich die Rekuperation messen, doch da blicke ich noch nicht ganz durch, wo und wie (und ob überhaupt) welcher Strom zurück zur Batterie fließt.

Hallsensoren für die Drehzahl

MHRM 12G2501 www.chenyang-gmbh.com



Versorgungsspannung:	12 V
Schaltfrequenz	0 – 15 kHz
Stromaufnahme	20 mA
Signal	LOW and HIGH; 0 / 5V

Hallsensoren erzeugen einen LOW- oder HIGH- Pegel auf dem Signaldraht, je nach Annäherung eines Magnetfeldes. Das heißt, der Pegel änder sich jedes Mal, wenn sich eine Schraube des Flansches vorbeidreht. Dieses Signal wird auf einen digitalen Port des Arduino geleitet und ruft dort einen Interrupt hervor, wenn der Pegel von High auf Low abfällt. Diese Interrupts werden in einer eigenen, kurzen Routine gezählt und die Anzahl jede volle Sekunde abgefragt. Diese ergeben dann, dividiert durch die Anzahl der Schrauben, die Umdrehungen pro Sekunde.
Ein Sensor liegt am Motor – Motordrehzahl, der andere liegt an der linken Vorderachse – Geschwindigkeit des Fahrzeuges. Die hätte ich auch am Tacho, hier bekomme ich sie aber digital.

Spannungsversorgung für den Arduino

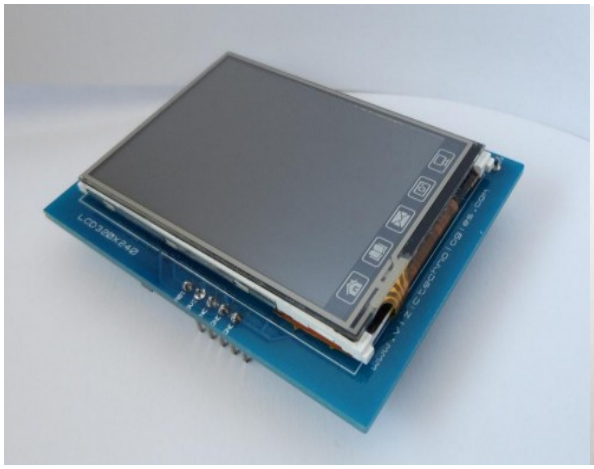
DC/DC Converter THL 10-2412WI

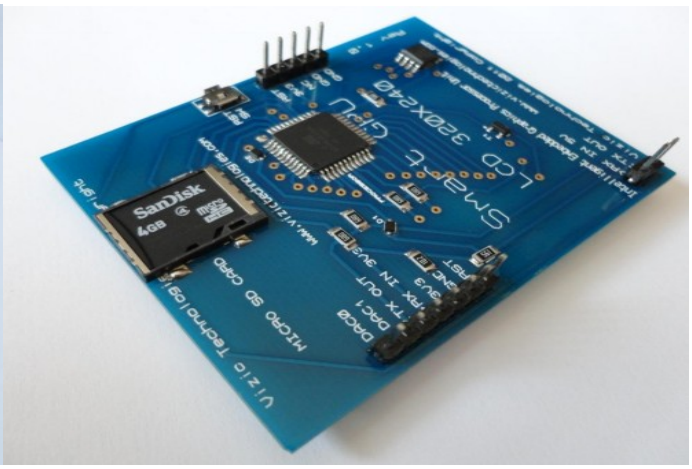
Eingang	9 – 36 V
Ausgang	12 V
Spannungsglättung, Überspannungsschutz	

Die Versorgungsspannung des Arduinos soll durch diesen DC/DC Wandler von der Bordbatterie gespeist, auf 12 V gehalten und von Spannungsspitzen geschützt werden.

Das Display

SMART GPU SOFT V2 Colour Touch Screen www.vizictechnologies.com





The **SMART GPU** is a powerful easy to use embedded graphics processor mounted on a board with a touch color LCD. It's aimed to help developers to create advanced Graphical User Interfaces (GUIs) in a very easy way. The SMART GPU is a low power/very high performance graphics processor, with a microSD card slot supporting up to 32 GB of storage (read/write), and FAT/FAT12/FAT16 or FAT32 universal file System that is compatible with any PC, no special format is needed.

Features:

- 2.4“ LCD capable of displaying 262,144 colors.
- Easy 5 pin interface to any host device: VCC, TX, RX, GND, RESET.
- On-board uSD/uSDHC memory card adaptor compatible with FAT(windows PC), Support up to 32GB for storing images and text, New Data Logger functions (read-write).
- Integrated Touch screen driver.
- PWM controlled display brightness.
- 5 general purpose Icons on touch
- Sleep mode.
- 10 bit accuracy touch.
- 2 General purpose Digital Output pins on board.
- Baud Rate speed up to 2000000 bps, 8 bits, no parity, 1 stop bit.
- 5V and 3V3 I/O compatible.
- 3V3 power supply.
- External reset switch.

- Und ganz wichtig: passt direkt drauf auf den Arduino, kann einfach aufgesteckt werden, ohne zusätzlliche Verdrahtung und lässt beim Arduino Mega noch genügend Ports frei für die Sensoren.

Das tolle Color Touch Display hab ich selbstverständlich noch lange nicht ausgereitzt, da gäbe es noch einige Ideen, die Darstellung der Sensorwerte User-freundlicher zu gestalten – kommt vielleicht noch.

Weiters ist ein SD-Kartenslot eingebaut, der Speicherung und Auslesen von Daten erlaubt.

Anmerken möchte ich noch, dass die Menge an neuen elektrischen Verbrauchern, eine Reihe von Änderungen im Bereich Sicherungen nach sich gezogen hat. Ursprünglich im 2CV vorhanden waren 3 Glasröhrchen-Sicherungen, jetzt hab ich 4 Bänke à 6 Sicherungen. Noch nicht alles voll, doch es wird schon!

2. Der Einbau

Die Sensoren wurden im Auto montiert und die Sensor-Leitungen durch vorhandene Bohrungen in den Autoinnenraum geführt. Dort sind sie in einem 44-poligen Stecker, mit drei Reihen von Stiften zusammengeführt und verlötet. Das Gegenstück dazu befindet sich auf der Rückseite des Arduino-Gehäuses. Von dort sind sie auf einer Steckplatine verlötet, welche in die Steckerbuchsen des Arduino passt.

Der Stecker

44-polig, 3 Reihen. 8 Sensorkabel sind hier zusammengeführt und verlötet.

Die Buchse am Arduino-Gehäuse

Um sicherzustellen, dass man den Arduino jederzeit aus dem Fahrzeug nehmen kann, ist ein solides Gehäuse mit einem gut gängigen Steckerpaar verwendet worden.

Die Steckplatine

Die Sensorkabeln vom 44-poligen Stecker, die Stromversorgung, 5 V und 12 V, sind auf der Platine verlötet und zu Steckern geführt, die in den Arduino passen. Kann als Shield direkt auf den Teil des Arduino Mega gesteckt werden, den das Display freilässt. Hier ohne das Display.

Das Display

Das Display kann als Shield direkt aufgesteckt werden, samt Kommunikation und Stromversorgung. Keine zusätzliche Lötung etc. mehr nötig. Beim Arduino Mega bleiben unten noch einige Anschlüsse frei.
Die Abdeckung drauf und zugeschraubt!

Halterung im Auto

Im 2CV musste noch eine Halterung für das Handy (nicht nötig, aber praktisch) und den Arduino gemacht werden.

Anmerkung:

Bei der Durchführung der Sensorkabel aus dem Motorraum ins Innere des Fahrzeuges, sind die Kabel der Temperaturfühler durch die selbe Öffnung gezogen worden, durch die auch die dicken Kabel zum Not-Aus-Schalter verlegt sind. Da fließt der ganze Fahr-Strom durch und der beeinflussen den Strom, der in den Sensorkabeln fließt! (Geb ich „Gas“, „steigt“ die Temperatur um bis zu 4 °C). Das muss ich noch ändern!

Vision

Weil so ein Projekt ja NIE fertig und abgeschlossen ist, gibt's immer was zu tun dabei. Ich habe also schon noch gewisse Ziele, die ich verwirklichen möchte.

Da aber die Batterien des Fahrzeuges schon über 5 Jahre alt sind und kaum noch lange halten werden, will ich vorerst nichts Wesentliches mehr investieren. Weder an Zeit noch an Geld. Alles weitere erst mit neuen Batterien, denn ich weiß heute noch nicht, was neue Akkus an Änderungen im elektrischen Verhalten, beim Verbrauch und bei der Ladung bedingen werden.

Wunsch jedoch ist: Einen Logfile der Messwerte anlegen, der auf der SD-Karte gespeichert wird. Daraus ein mittleres Verbrauchsverhalten rechnen und dieses dann dem jeweiligen beim Fahren gegenüberzustellen. Soll exaktere Rückschlüsse auf die noch vorhandenen elektrischen Reserven geben, Reichweiten etc. Weiters die grafische Aufbereitung der Messwerte in Balkendiagrammen und/oder Farbe Rot bei Erreichen kritischer Werte etc. Da gibt's noch viel Spiel-Raum.

3. Das Programm

Damit man auch sehen kann, was die Sensoren so alles herausgefunden haben, braucht es nicht nur das Equipment, wie oben beschrieben, sondern auch noch ein Programm, das dem Microkontroller sagt, was er mit den Werten anfangen soll. Geschrieben in C++, eine objektorientierte Sprache, hier mit einer Umgebung, die direkt an den Arduino angepasst ist.

Kann man alles frei bei den oben angeführten Links herunterladen, ist Open Source.

Für das Display gibt es eine eigene Library und Beispiele. Ist recht einfach zu verwenden.

(Anmerkung:
Fehler und Mängel im Programm sind nicht ausgeschlossen, mehr noch: sogar sehr wahrscheinlich)

```
#include <Arduino.h>      // the commmon Arduino library
#include <SMARTGPU.h>     // include the SMARTGPU library!


SMARTGPU lcd;             // create our object called lcd



 int SensPin[] = {15, 14, 12, 11, 9, 8 };     // Sens[6] : analogPin:0=Temp1, 1=Temp2,
                                              // 2=Spann1,  3=Spann2, 4=Strom1,  5=Strom2
 const int AnzSens = 6;                       // Anzahl der AnalogSensoren
 int hw[] = {500, 500, 90, 90, 150, 150 };    // Die Höchstwerte der Analog-Sensoren
 float Mess[AnzSens] ={0, 0, 0, 0, 0, 0};     // Feld der Messwerte von Temp1 bis Drehz2
 float Temp1 = Mess[0];
 float Temp2 = Mess[1];


 float Messg = 0;                             // einzelner Messwert der analog Sensoren
 float Messwert = 0;
 const int cycles = 30;                       // Anzahl der Messungen zur Mittelung
                                              // streuender AnalogWerte
```

```
 char Bez1[] = {'S', 'A', 'F', 'B','S', 'M'};
 char Bez2[] = {'t', 'u', 'B', 'B','t', 'o'};
 char Bez3[] = {'G', 'G', 'V', 'V','A', 'A'};
 char Bez4[] = {' ', ' ', ' ', ' ',' ', ' '};

 float Fakt = 1.1121238;          // Radumfang / AnzSchrauben * 3600 / 1000
                                  //  0,59 x 3,14159 / 6 x 3600 / 1000 = 1,11212379937079
 int pesoT = 10;                  // Gewichtung für Erneuerung der Werte,  20 zu groß,
                                  // Änderung geht langsam

 //_____________________________________________________________________

 //Drehzahl
  unsigned long TaM = millis();
  unsigned long TaA = millis();
  volatile int drehzA = 0;
  volatile int drehzM = 0;

  void zaehlM()   {    // Interruptbehandlungsroutine für Motordrehzahl
    drehzM++;       // Durchgänge mitzählen, muss noch div werden durch Anzahl Schrauben!
 }

  void zaehlA()   {   // Interruptbehandlungsroutine für Raddrehzahl
   drehzA++;        // Durchgänge mitzählen, muss noch div werden durch Anzahl Schrauben!
 }
//_____________________________________________________________________

void pZeil(int Z, char Bez1, char Bez2, char Bez3, char Bez4){
    char Zeile[16]={Bez1,Bez2, ' ',' ',' ',' ',' ',' ',' ',' ',Bez3,Bez4,' ',0x00};
    int Messw = Messwert;

  switch(Z){
    case 3:      // Spann 1   Fahr Batterie
      Messw *= 0.94;    // Korrekturfaktor weil Wert nicht genau ist?? woher kommt die
                        // Ungenauigkeit?
      if(Messw > 999)  Zeile[3]=((Messw/1000%10)+0x30);
      if(Messw > 99)   Zeile[4]=((Messw/100%10)+0x30);
      if(Messw > 9)    Zeile[5]=((Messw/10%10)+0x30);    // convert Messw to ascii format
      Zeile[6]=(Messw%10)+0x30;                          // convert Messw to ascii format
      Zeile[7]=',';
      Messw = Messwert*10;                    // hier müsste jetzt eigentlich die erste
                                              // Nachkommastelle VOR dem Komma stehen
      Zeile[8]= ((Messw%10)+0x30);
      break;

    case 4:    // Spann 2   Bord Batterie
```

```
      Messw *= 0.64;    // Korrekturfaktor weil Wert nicht genau ist?? woher kommt die
                        // Ungenauigkeit?
      if(Messw > 999)  Zeile[3]=((Messw/1000%10)+0x30);
      if(Messw > 99)   Zeile[4]=((Messw/100%10)+0x30);
      if(Messw > 9)    Zeile[5]=((Messw/10%10)+0x30);    // convert Messw to ascii format
      Zeile[6]=(Messw%10)+0x30;                          // convert Messw to ascii format
      Zeile[7]=',';
      Messw = Messwert*10;
      Zeile[8]= ((Messw%10)+0x30);
      break;

    default:
      if(Messw > 999)  Zeile[3]=((Messw/1000%10)+0x30);
      if(Messw > 99)   Zeile[4]=((Messw/100%10)+0x30);
      if(Messw > 9)    Zeile[5]=((Messw/10%10)+0x30);    // convert Messw to ascii format
      Zeile[6]=(Messw%10)+0x30;                          // convert Messw to ascii format

   }   // END switch

    int Ze = (Z * 35)-20;

    lcd.drawRectangle(60,Ze+1,170,Ze+28, BLACK,FILL);     // vorige Ziff löschen, Xmax
                                                          //239,  Ymax 319,  x1,y1, x2,y2
    lcd.string(10,Ze,239,319,WHITE,FONT7,TRANS,Zeile);    // Zeile schreiben

}      // END pZeil

//_____________________________________________________________________

void liesSens (int SensPin, int hw)  {        // Temp1, Temp2,  Spann1,  Spann2,  Strom1,
Strom2
    Messwert = 0;
    float analogValue = analogRead(SensPin);
    Messg = map(analogValue,0,1023,0,hw);

}
//_____________________________________________________________

void setup() { //initial setup
    //Those two functions must always be called for SMARTGPU support
    lcd.init();  //configure the serial and pinout of arduino board for SMARTGPU support
    lcd.start(); //initialize the SMARTGPU processor
    lcd.orientation(PORTRAITL);
    attachInterrupt(2, zaehlM, FALLING);   // Motor
    attachInterrupt(3, zaehlA, FALLING);    // li Achse
                        // Eine Schraube dreht sich vorbei: 8,4 V nix,  0,35 V sie ist da
```

```
                        // numbers 0 (on digital pin 2) and 1 (on digital pin 3)
                        // The Arduino Mega has an additional four: numbers 2 (pin 21),
                        // 3 (pin 20), 4 (pin 19), and 5 (pin 18).
    //Serial.begin(9600);
 }

void loop() { //main loop

    lcd.drawLine(5,9,235,9,YELLOW);           // Trennlinien zwischen den Messwerten
    lcd.drawLine(5,80,235,80,YELLOW);
    lcd.drawLine(5,150,235,150,YELLOW);
    lcd.drawLine(5,220,235,220,YELLOW);

    float Temp1 = Mess[0];
    float Temp2 = Mess[1];

    int i = 0;

    while(i<cycles) {
       i++;
          for(int j = 0; j < AnzSens; j++) {
              liesSens(SensPin[j],hw[j]);      // Sens[6] : 0=Temp1, 1=Temp2,  2=Spann1,
                                               // 3=Spann2,  4=Strom1,  5=Strom2
              Mess[j] += Messg;             // Messwerte aufsummieren für Mittelbildung

          // Drehzah: _______________
              if (millis() >= (TaM + 1000)) {  // weiß nicht, ob das so geht,, ev in
                                               // eig Routine (InterruptBehandlung)
                  TaM = millis();
                  Messwert = drehzM/4*60;      // Durch Anzahl der Schrauben dividieren
                                               // und Umspeichern auf neuen Wert
                  pZeil(7, 'M', 'o', 'U','s');        // Ausdruck Zeile 7 Drehzahl Motor
                  drehzM = 0;
                  //Serial.println(millis());
              }

              if (millis() >= (TaA + 1000)) {
                  TaA = millis();
                  Messwert = drehzA*Fakt;    // Faktor = UmfangRad / AnzSchrauben * 3600
                                             // / 1000  und Umspeichern auf neuen Wert
                  pZeil(8, 'G', 'e', 'k', 'm');  // Ausdruck Zeile 8 Geschwindkt  km/h
                  drehzA = 0;
              }    //_________________

          }     // End For(j=
```

```
  }   // END While


    for(int n = 0; n < AnzSens; n++) {                    //Mittelbildung
               Mess[n] /=  cycles;
               if(n == 0){         // für Temp1 u Temp2 gewichtetes Mittel,,,  jedoch
                                   // kommen anfangs komische Werte daher!?
                   Temp1 *= pesoT;
                   Mess[n] += Temp1;
                   Mess[n] /=pesoT+1;
               }
                if(n == 1){
                   Temp2 *= pesoT;
                   Mess[n] += Temp2;
                   Mess[n] /=pesoT+1;
               }
               Messwert= Mess[n];
      /*         if( n == 4) {           // Am Stomsensor 1 ist dzt nix angeschlossen
                  Messwert = 0;
               }        */
               pZeil(n+1, Bez1[n], Bez2[n], Bez3[n], Bez4[n]);    //Zeil(int Z, char
                                          // Bez1, char Bez2, char Bez3,char Bez4)
          }
}  // end program
```