1. Vorwort

Jedes Jahr müssen die Schüler der 13. Klasse auf dem Gebiet Elektrotechnik, sowohl auf der Sektion Energie, wie auch auf der Sektion Kommunikation, ein Projekt bestehen, um an den Abschlussarbeiten der 13. Klasse teilnehmen zu können.

Was versteht man unter einem Projekt?

Ein Projekt ist, wie der Name schon sagt, eine Arbeit, die man von einem Vorstand zugeteilt bekommt und in einer Gruppe von 2 bis zu 5 Schülern, der Sektion Energie oder Kommunikation, bewältigen sollte. Man muss ein solches Projekt durchführen und bestehen, um die 13. Klasse ze bestehen.

Anfang des Jahres bekamen wir 15 verschiedene Projekte präsentiert. Anschließend haben wir eine Tabelle mit denjenigen angelegt, die wir gerne oder weniger gerne durchführen möchten . Diese haben wir eine Woche danach unsere Entscheidung mitgeteilt.

Wir, Einsweiler Tom, Man Chun Ling, Mangers Charel, Weyland Mich und Weyrich Sascha wurden vom Vorstand einander zugeteilt und bekamen das Projekt „Mess-Stationen im Musée des Mines, Rumelange“. Unser Betreuer von dem auch der Projektvorschlag stammt, ist Herr Jean-Claude Feltes.

In den ersten drei Stunden waren wir in Rumelange, um uns die Vorarbeit von Herr Feltes anzusehen. Herr Feltes hat bereits eine komplette Mess-Station gebaut und diese in Rumelange installiert. In den darauffolgenden Stunden machten wir uns Gedanken, wie wir dies realisieren sollten, verteilten die einzelnen Aufgaben, zeichneten erste Schaltpläne und bauten diese auf. Wir arbeiteten den größten Teil der Zeit alleine, tauschten jedoch immer wieder Meinungen aus. Natürlich hatten wir gelegentlich Meinungsunterschiede. Jedoch arbeiteten wir in einem angenehmen Klima und verstanden uns recht gut. Als die Schaltungen endlich nach ein paar Wochen / Monaten fast funktionierten, machten wir uns Gedanken darüber, wie das ganze optisch aussehen sollte, entworfen Layouts und ätzten diese auf Platinen. Wir kamen gut voran, jedoch hatten wir ein großes Problem. Keiner von uns war gut im programmieren und so waren wir bis zum Schluss nur noch am programmieren.

2. Aufgabenstellung

In diesem Projekt geht es darum, im Bergwerksmuseum in Rumelange mehrere Mess-Stationen zu installieren, deren Daten an einem zentralen Ort gesammelt und auf einem PC gespeichert werden. In einem späteren Projekt (vielleicht nächstes Jahr) sollen diese dann über Internet zugänglich gemacht werden.
Gemessen werden sollen erst einmal CO2-Konzentration und Temperatur. Hierzu werden fertige Sensoren mit Microcontroller-Schnittstellen benutzt.

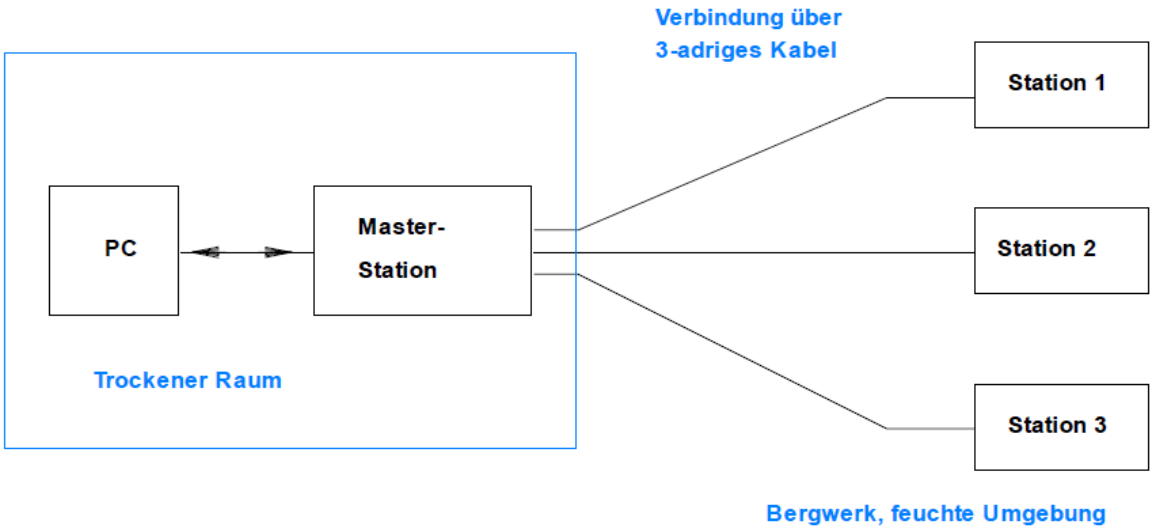

Abb. 1: Blockschaltbild der Aufgabenstellung.

Mess-Stationen

- Erfassen der CO2-Konzentration mit einem fertigen Sensor, Datenübertragung über I2C-Bus.
- Erfassen der Temperatur im Stollen.
- Temperatur-Regelung auf eine einstellbare Temperatur für das Innere der Mess-Station, um Korrosion zu vermeiden.
- Erweiterung möglich für andere Sensoren. Je nach Interesse der Schüler eventuell noch Sensoren für Luftfeuchtigkeit und Windgeschwindigkeit.
- Datenübertragung: Empfang von Kommandos und Senden der Messdaten zur Master-Station.

Kabel

Aus Kostengründen wird ein normales 3-adriges Elektrokabel benutzt.
Das bedeutet: 2 Adern für die Betriebsspannung (+UB und Masse) und eine Ader für die bidirektionale Kommunikation.

Master-Station

- Verwaltung der Datenübertragung:
  Schicken von Kommandos an die Mess-Stationen, Empfangen der Daten.
- Anzeige der Messwerte auf einem LCD-Display.
- Echtzeit-Uhr + Kalender (fertiger Chip).
- Zusammenfassung aller Daten und Übertragung an einen PC (über die serielle Schnittstelle).

Das Projekt richtet sich an Schüler der Energie- oder Kommunikationsklasse. Interesse an Mikrocontrollern ist von Vorteil. Je nach Schülerzahl und Interessen ist auch (in Grenzen) eine Anpassung der Aufgabenstellung möglich.

Originale Aufgabenstellung von Herr Feltes, gefunden unter:
http://staff.ltam.lu/feljc/school/projets13/2010_Projet_MMR_Aufgabenstellung.pdf

3. Blockschaltbild

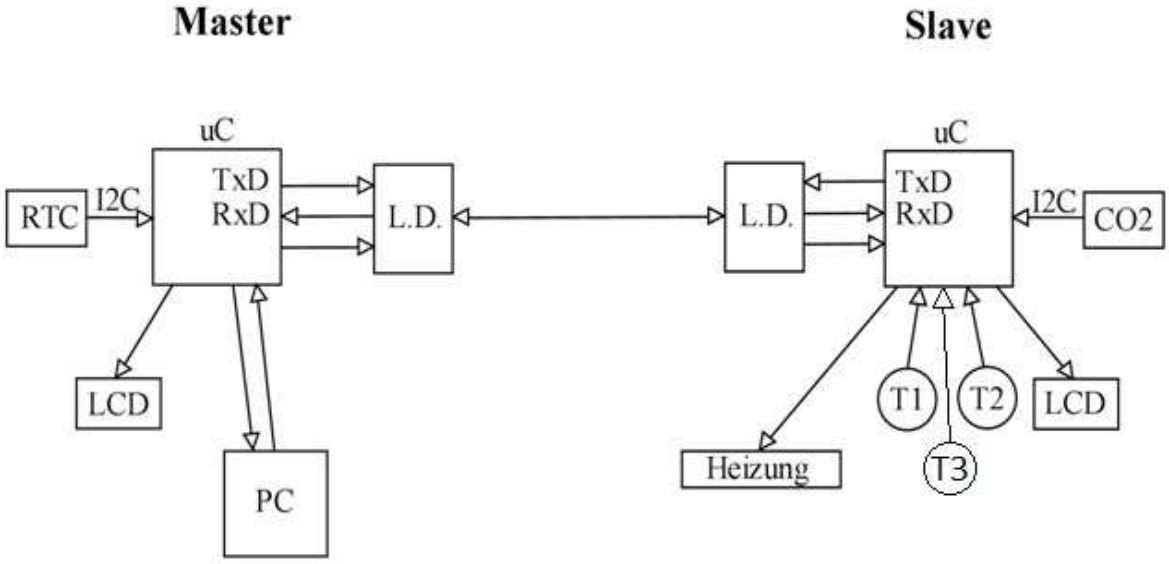

Abb. 2: gesamtes Blockschaltbild.

Bezeichnung	Erklärung	
Masterstation	Empfangen der Daten von der Messstation und weiterleiten an PC und LCD.	Charel /Tom
RTC	Real Time Clock (Echtzeituhr).	Tom
LCD	LCD-Display vom Master.	Charel
PC	Kommunikation zwischen PC und uC (RS 232).	Charel / Tom
Kommunikation	Kommunikation zwischen Master- und Messstation.	Charel / Tom
Messstation	Erfassen des CO2-Gehaltes und den Temperaturwerte. Weiterleiten der Daten an die Masterstation.	Chun Ling / Mich / Sascha
L.D.	Datendriver, senden/empfangen der Daten über eine Leitung.	Sascha
Heizung	Heizung falls die Messstation unterkühlt ist.	Mich
T1-3	Temperatursensoren.	Mich
CO2	Erfassen des CO2 in der Mine.	Chun Ling
LCD	LCD-Display von der Messstation.	Sascha

4. Masterstation

Abb. 3+4: Fotos der Masterstation.

4.1. Schaltplan

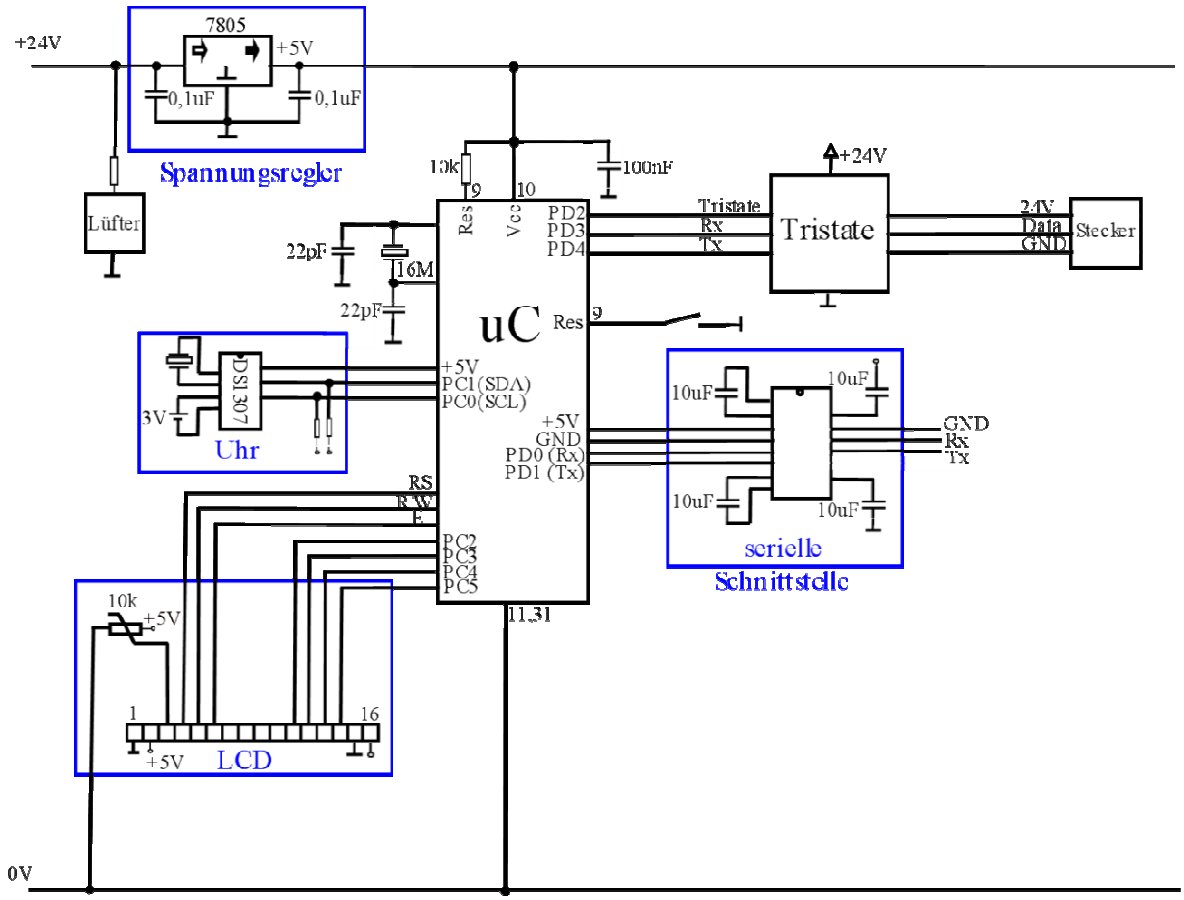

Abb. 5: Schaltplan der Masterstation (von Mangers Charel).

4.2. Netzteil

Als Speisung für unsere Master- sowie Messstation, haben wir ein 230V AC auf 24V DC Netzteil der Marke „Digital“ (heute: Compaq) genommen.

Eingang: 230V / 50Hz

Ausgang: 24V

← Auf dem Foto sehen wir nicht das originale Netzteil!

Abb. 6: Netzteil für die Versorgungsspannung.

4.3. Spannungsregler

4.4. Einleitung

Da auf unserer Master- sowie Messstation, mehrere Bauteile mit 5V gespeist werden und wir von unserem Netzteil 24V bekommen, mussten wir einen Spannungsregler vorsehen. Dafür nahmen wir den Festspannungsregler “7805” für positive Spannungen.

4.4.1. Schaltung

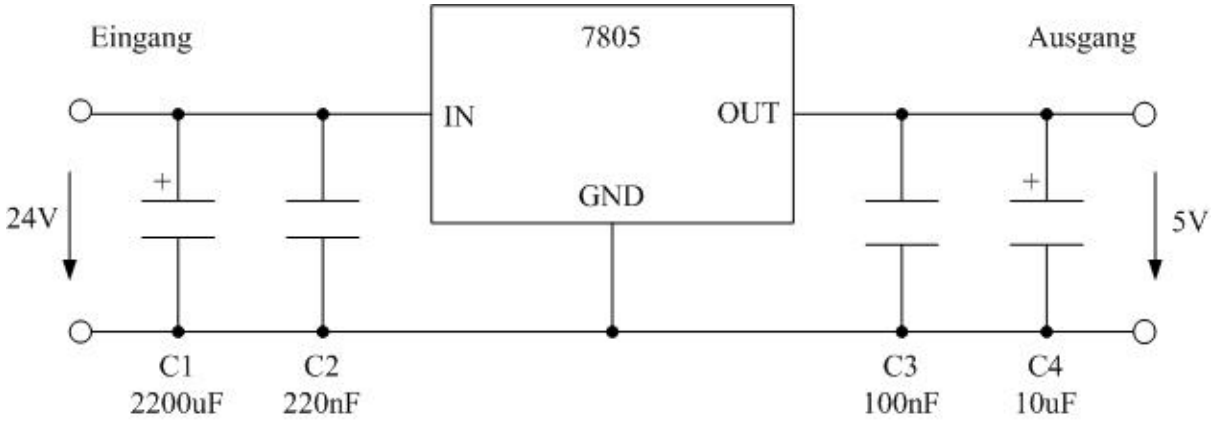

Abb. 7: Schaltplan (Spannungsregler).

4.4.2. Platine

Das Layout wurde im Eagle Layout Editor 5.10.0 der Firma Cadsoft gezeichnet.

Layout:

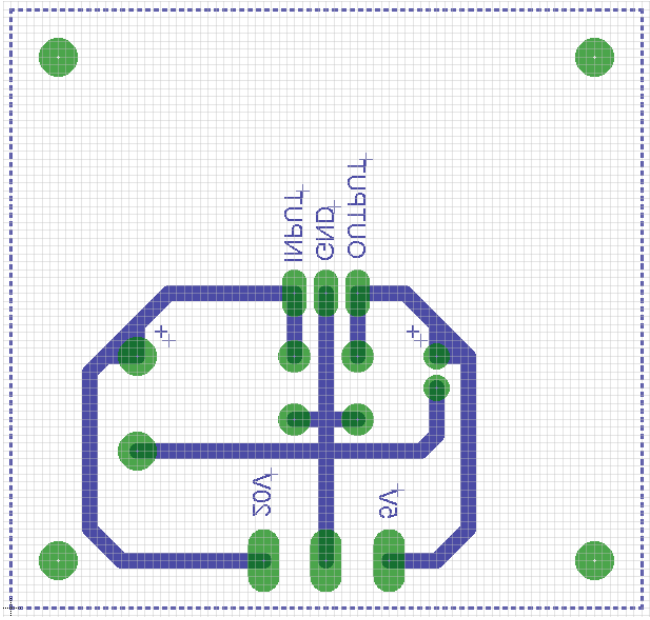

Abb. 8: Layout (Spannungsregler).

Bestückungsfolie:

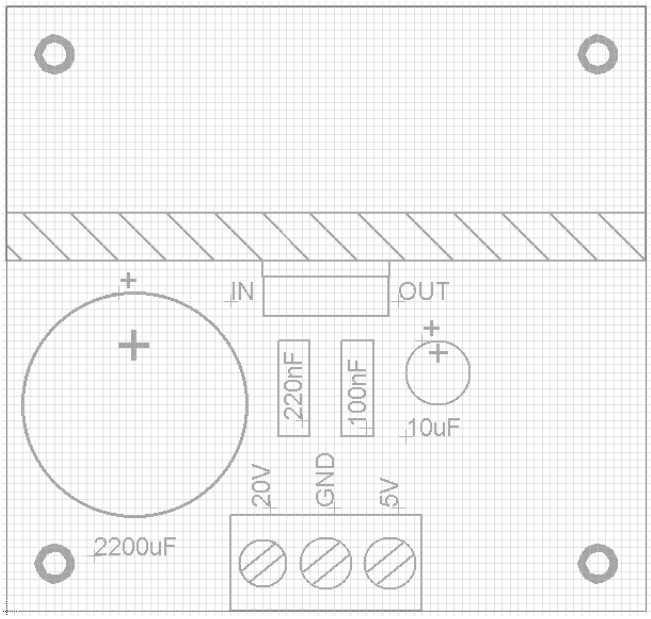

Abb. 9: Bestückungsfolie (Spannungsregler).

Platine:

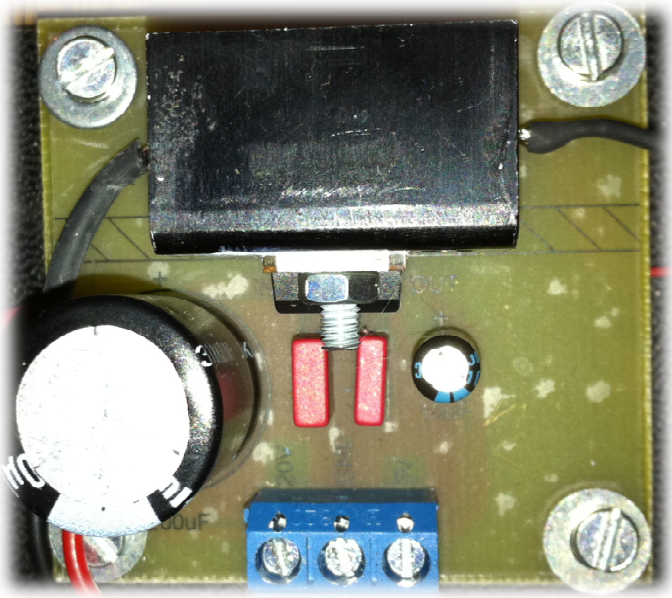

Abb. 10: Fertige Platine (Spannungsregler).

4.4.3. Bauteilliste

- Platine (50mm x 50mm)
- 1x Lüsterklemme Anreihe-3 (Blau)
- 1x Elektrolyt Kondensator von 2200uF
- 1x MKS Kondensator von 220nF
- 1x MKS Kondensator von 100nF
- 1x Elektrolyt Kondensator von 10uF

- 1x Festspannungsregler 7805
- 1x Kühlkörper

4.5. Lüfter

4.5.1. Einleitung

Beim Spannungsregler haben wir eine Eingangsspanung von 24V und eine Ausgangsspannung von 5V. Dies verursacht einen Spannungsabfall von 24V - 5V = 19V, was natürlich viel zu hoch und nicht gerade energiesparend ist. Diese 19V fallen ausschließlich am Festspannungsregler ab und somit wird dieser und dessen Kühlkörper extrem heiß, weswegen wir auch zu einem Lüfter zurückgriffen. Dieser kühlt den Festspannungsregler und den Kühlkörper auf eine angenehme Temperatur herab.

Der Lüfter gehört der Marke SUNOM an und ist vom Typ MegLev B1245PFV1-8A und wird mit 12V gespeist.

Diese 12V Betriebsspannung sind ein weiteres Problem, da der Lüfter vor den Festspannungsregler angebreacht werden muss, und wir dort wiederum nur 24V zur Verfügung haben. Es musste also ein Leistungswiderstand her, der unsere 24V auf 12V herabsetzt. Dafür wurde ein 82 Ohm Leistungwiderstand benutzt, da wir Strom von 146mA haben: Rv = (24V-12V) / 146mA = 83 Ohm.

Da dieser Leistungswiderstand auch wieder extrem heiß wird, haben wir ihn unter unseren Kühlkörper auf unsere Spannungsregler-Platine angebracht, so dass dieser gleichzeitig mit gekühlt wird.

Dies ist nicht gerade die passende Lösung in Punkto Spannungsregler, Lüfter und Leistungswiderstand, jedoch rannte uns die Zeit weg und wir mussten voran kommen!

4.5.2. Schaltung

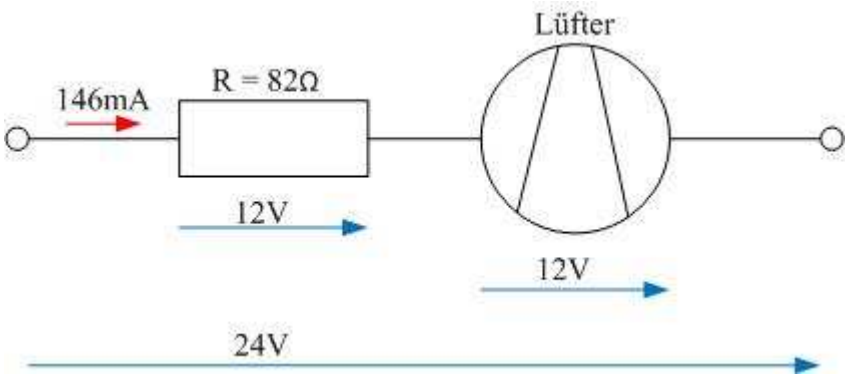

Abb. 11: Schaltplan (Lüfter).

4.5.3. Foto

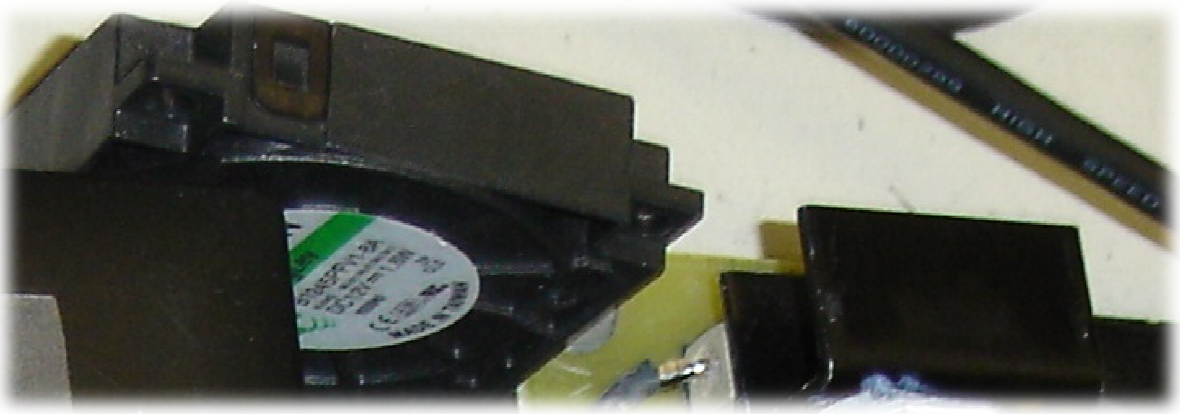

Abb. 12: Foto vom Lüfter.

4.5.4. Bauteilliste

- Lüfter
- Leitungen
- 82Ω Leistungswiderstand
- Schrumpfschlauch

4.6. RTC

4.6.1. Einleitung

Die Echtzeituhr, auch noch RTC genannt, liefert dem Mikrokontroller auf der Masterstation das Datum und die aktuelle Uhrzeit. Das Datum und die Uhrzeit werden im Terminal mit dem Buchstaben “s” eingestellt und werden am LCD und am Terminal mit den dazugehöhrigen Daten angezeigt.

Die Echtzeituhr wird aus einem Taktgeber von 32,768kHz gewonnen und wird über eine Lithium Batterie von 3V gepuffert, damit sie auch dann weiterläuft, wenn sie von der Versorgungsspannung, die 5V beträgt, getrennt wird.

Wir bekammen von Herrn Weiler den “DS1307”- Baustein der Marke Dallas mit dem wir unsere Echtzeituhr realisisierten. Dieser Baustein verfügt über eine interne Kalenderfunktion, so dass es leichter war, das Datum zu integrieren. Die Kommunikation zwischen RTC und Mikrokontroller erfolgt über die I2C Schnittstelle (SDA und SCL). SDA und SCL werden parallel zum Mikrokontroller auch noch an 4,7kΩ Widerstände angeschlossen und dienen als Pullup-Widerstände.

4.6.2. Schaltung

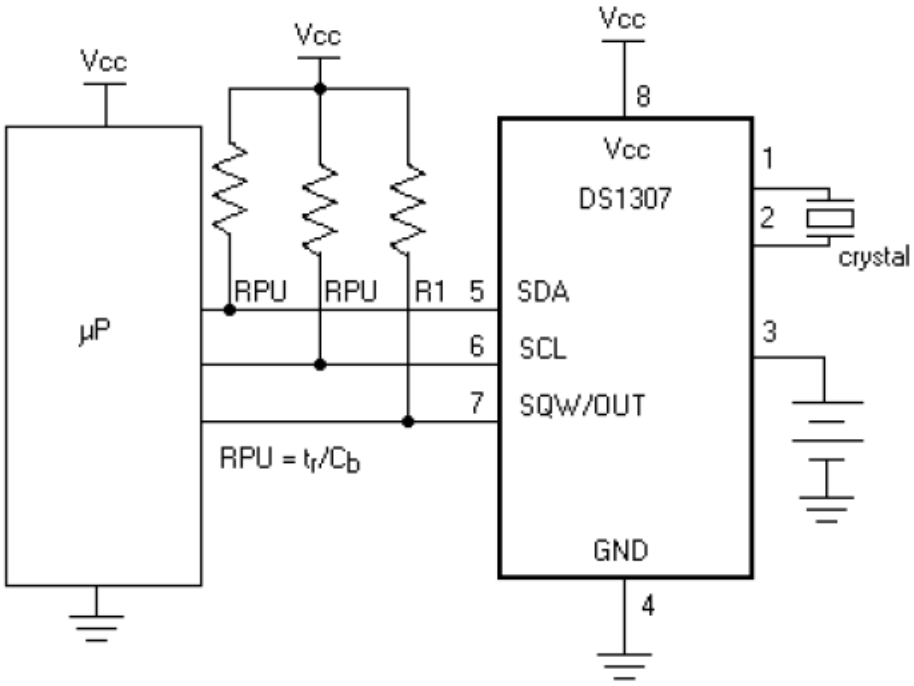

Abb. 13: Schaltplan (RTC).

4.6.3. Platine

Das Layout wurde im Eagle Layout Editor 5.10.0 der Firma Cadsoft gezeichnet.

Layout:

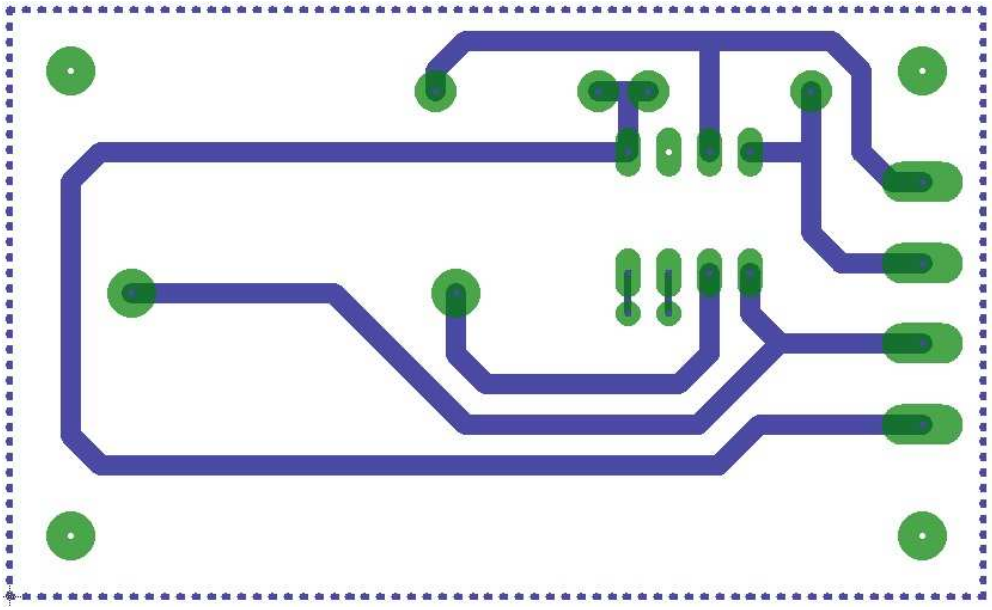

Abb. 14: Layout (RTC).

Bestückungsfolie:

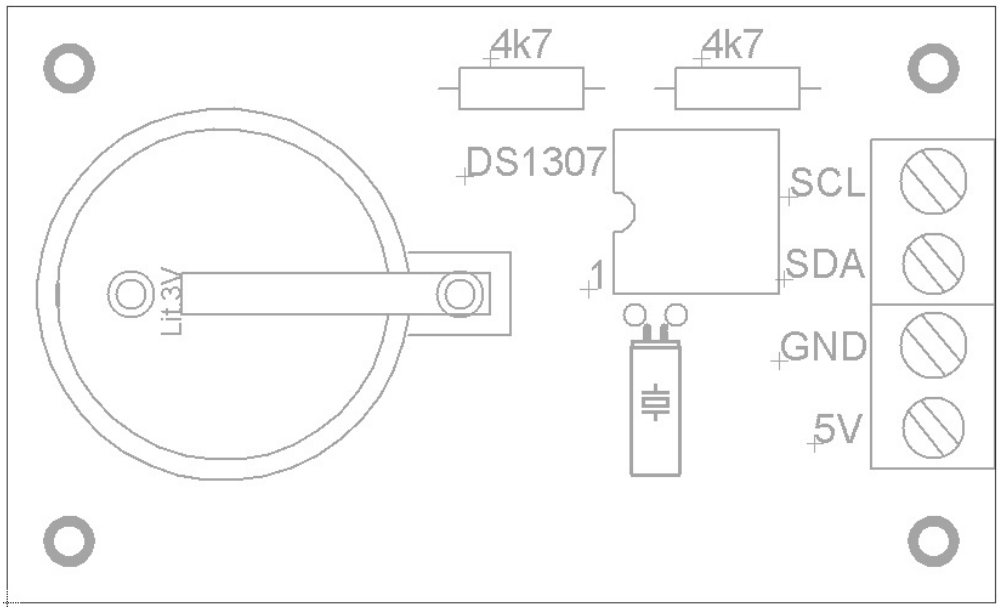

Abb. 15: Bestückungsfolie (RTC).

Platine:

Abb. 16: Fertige Platine (RTC).

4.6.4. Bauteilliste

- Platine (37mm x 61mm)
- 2x Lüsterklemme Anreihe-2 (Blau)
- 1x IC →DS1307 (8Pin)
- 1x IC Sockel (8Pin)
- 1x 32,768kHz Quarz
- 2x Widerstand 4,7kΩ
- 1x Lithium Batterie von 3V
- 1x Batteriehalterung

4.6.5. Flussdiagramm

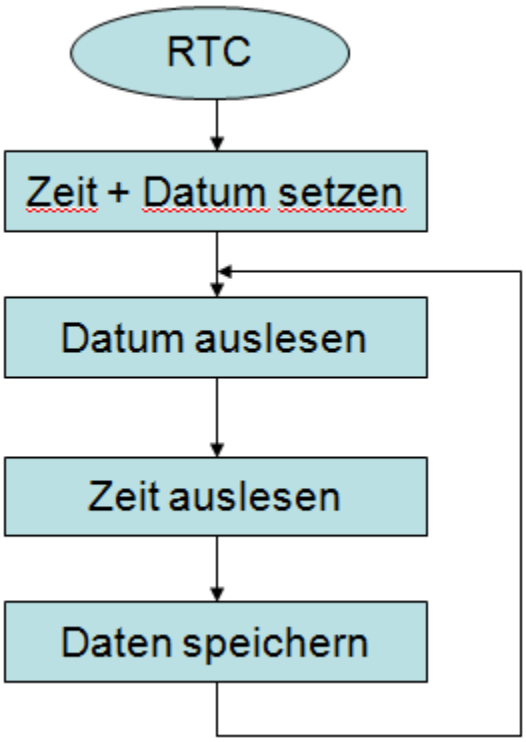

Abb. 17: Flussdiagramm (RTC).

4.6.6. Programm

```
'*********************************************************************************
'MUSEE DES MINES RUMMELANGE
'PROJET T3 EC+EE 2010/2011
'BETREUER: JEAN-CLAUDE FELTES
'*********************************************************************************
'MASTER STATION     MASTER-RTC mit Timer 1s und setzen der Uhr über Terminal
'*********************************************************************************

'uC-INIT-----------------------------------------------------------------------
$regfile = "m16def.dat"
$crystal = 16000000
$baud = 9600
$hwstack = 100
$swstack = 100
$framesize = 100

'DISPLAY-----------------------------------------------------------------------
Config Lcd = 16 * 4
Config Lcdpin = Pin , Db4 = Portc.3 , Db5 = Portc.2 , Db6 = Portd.7 , Db7 = Portd.6
, E = Portc.4 , Rs = Portc.5
Cursor Noblink
Cursor Off

'I2C---------------------------------------------------------------------------
Config Sda = Portc.1
Config Scl = PortC.0

'RTC INIT----------------------------------------------------------------------
Dim S$ As String * 8
Dim Weekday As Byte
Config Clock = User
```

```
Config Date = Dmy , Separator = .

'Time$ = "22:14:00" 'wird nicht gebraucht, da die Uhr per Terminal gesetzt wird
'Date$ = "02.02.11"

'Timer 1s-------------------------------------------------------------------
Config Pina.7 = Output                                      'LED 1s
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1 , Compare A = Disconnect
Ocr1ah = High(15625)
Ocr1al = Low(15625)
Tccr1a = 0
On Compare1a Tim1_1s
Enable Compare1a
Enable Interrupts

'HAUPTPROGRAMM--------------------------------------------------------------
Cls
Do
   If Inkey() = "s" Then
    Disable Interrupts   'Interrupts wegen Terminal auschalten!!
      'mit Variable arbeiten, da sonst Time$ / Date$ nicht gesetzt wird
      Input "hh:mm:ss" , S$
         Time$ = S$
      Input "dd.mm.yy" , S$
         Date$ = S$
    Enable Interrupts
   End If
   Waitms 10
Loop
End

'UNTERPROGRAMME------------------------------------------------------------
Tim1_1s:
Gosub Lcdclock
Toggle Porta.7
Return

'--------------------------------------------------------------------------------
Getdatetime:
   I2cstart
   I2cwbyte &HD0
   I2cwbyte 0
   I2cstop
   I2cstart
   I2cwbyte &HD1
   I2crbyte _sec , Ack
   I2crbyte _min , Ack
   I2crbyte _hour , Ack
   I2crbyte Weekday , Ack
   I2crbyte _day , Ack
   I2crbyte _month , Ack
   I2crbyte _year , Nack
   I2cstop
   _sec = Makedec(_sec)
   _min = Makedec(_min)
   _hour = Makedec(_hour)
   _day = Makedec(_day)
   _month = Makedec(_month)
   _year = Makedec(_year)
Return

'--------------------------------------------------------------------------------
Setdate:
   _day = Makebcd(_day)
   _month = Makebcd(_month)
   _year = Makebcd(_year)
   I2cstart
```

```
   I2cwbyte &HD0
   I2cwbyte 4
   I2cwbyte _day
   I2cwbyte _month
   I2cwbyte _year
   I2cstop
Return

'-------------------------------------------------------------------------------
Settime:
   _sec = Makebcd(_sec)
   _min = Makebcd(_min)
   _hour = Makebcd(_hour)
   I2cstart
   I2cwbyte &HD0
   I2cwbyte 0
   I2cwbyte _sec
   I2cwbyte _min
   I2cwbyte _hour
   I2cstop
Return

'-------------------------------------------------------------------------------
Lcdclock:
   Locate 1 , 1 : Lcd "Datum: " ; Date$
   Locate 2 , 1 : Lcd "Zeit:  " ; Time$
Return

'-------------------------------------------------------------------------------
Terminal:
   Print "Date: " ; "    Time: " ; "    CO2: " ; "    Temperature IN: " ; "
Temperature OUT: " ; Chr(13) ; Date$ ; "  " ; Time$
Return

'-------------------------------------------------------------------------------
```

4.7. Mikrokontroller

4.7.1. Einleitung

Ein Mikrokontroller ist ein Baustein, in dem man fast alle Komponenten eines Mikrocomputers wiederfindet. Das heißt man spricht von einem Mikrokontroller, auch noch µC geschrieben, wenn man die CPU, den Speicher und einige Ein- und Ausgabebausteine in einen einzigen Chip integriert hat.

Sie können vielfältige Aufgaben mit unterschiedlicher Komplexität übernehmen und sind heute in fast allen elektronischen Geräten, oft sogar mehrfach, zu finden. Es existieren Mikrocontroller in vielen Varianten. Wir benutzen den ATmega 16 von Atmel, mit dem wir bereits seit der 12. Klasse arbeiten und somit auch kennen, bei der Masterstation.

Wir verwenden den µC um eine RTC (Real Time Clock, Echtzeituhr) anzusteuern, die Kommunikation zwischen Messstation und Masterstation herzustellen und um die empfangenen Werte mittels der seriellen Schnittstelle an den PC zu senden sowie an einen LCD.

Mikrokontroller kann man mit vielen Programmiersprachen programmieren. Einige davon sind beispielsweise Assembler, C oder AVR-Bascom. Es gibt jedoch noch viele mehr.

Wir haben mit AVR-Bascom programmiert, da wir in dieser Sprache eher zu Recht finden als in Assembler.

4.7.2. Schaltung

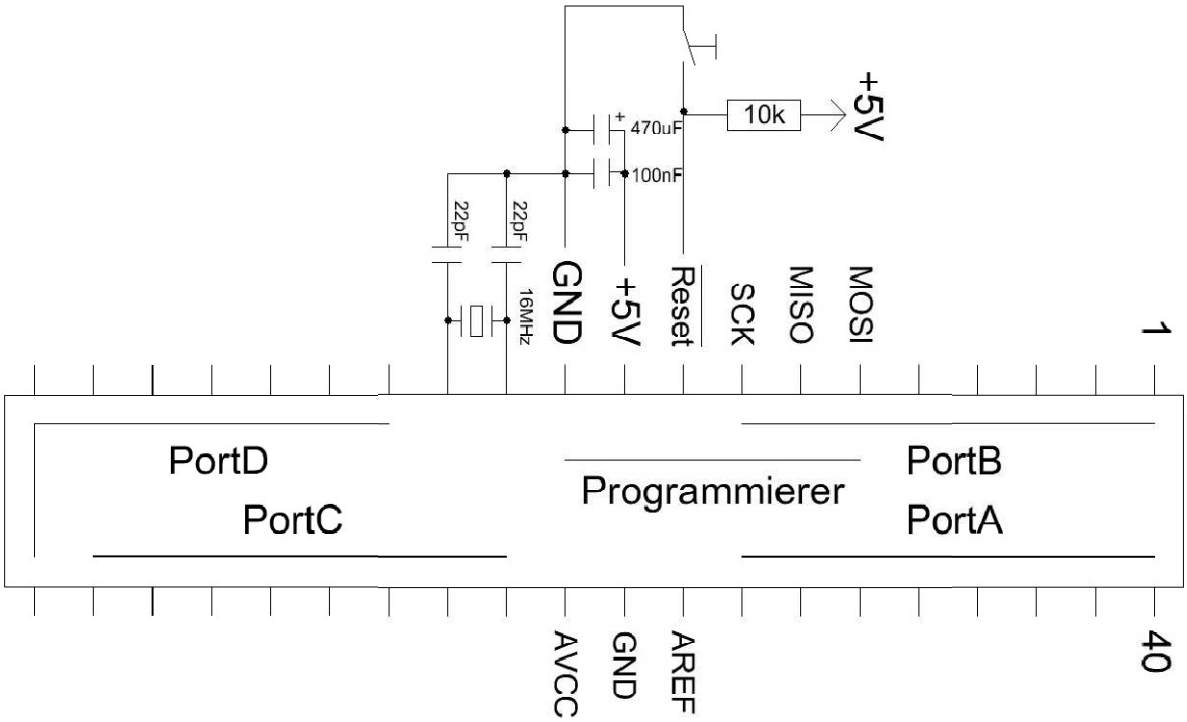

Abb. 18: Schaltplan (Mikrokontroller).

4.7.3. Platine

Das Layout wurde im Eagle Layout Editor 5.10.0 der Firma Cadsoft gezeichnet.

Layout:

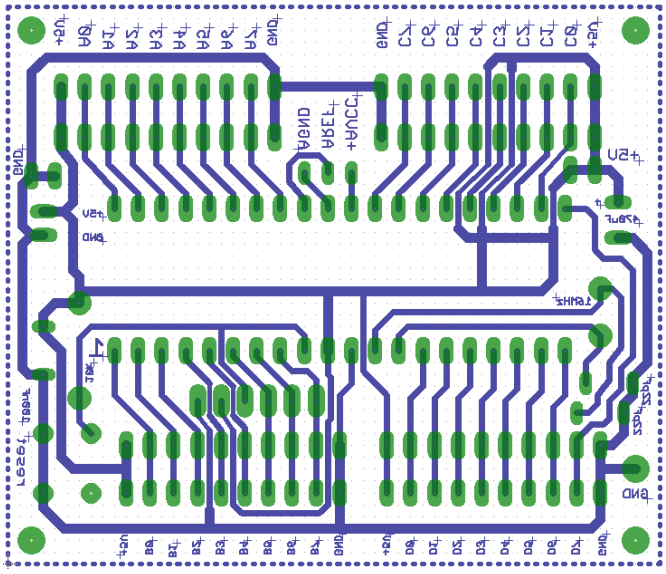

Abb. 19: Layout (Mikrokontroller).

Bestückungsfolie:

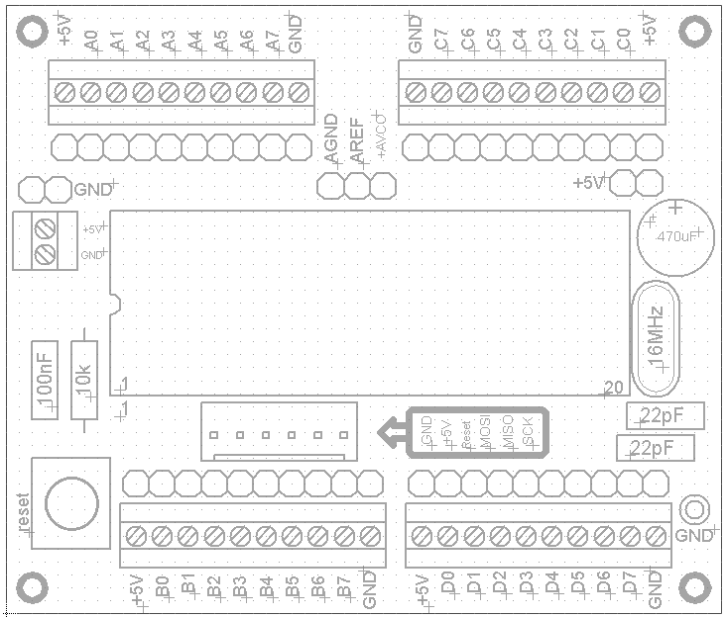

Abb. 20: Bestückungsfolie (Mikrokontroller).

Fertige Platine:

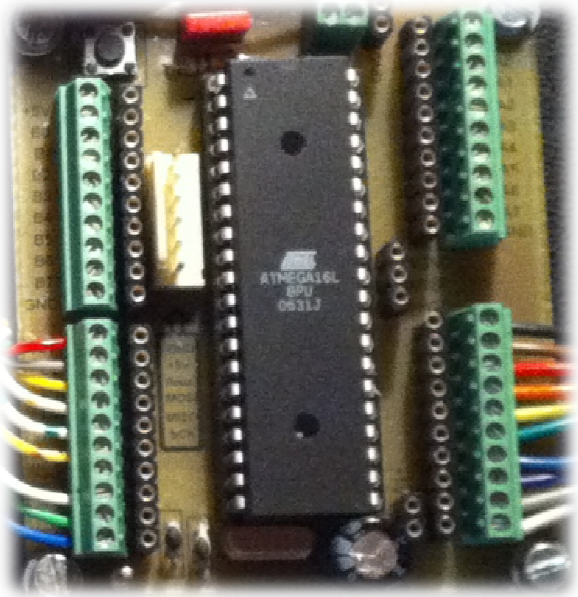

Abb. 21: Fertige Platine (Mikrokontroller).

4.7.4. Bauteilliste

- Platine (80mm x 80mm)
- 4x Lüsterklemme Anreihe-10 (Grün)
- 1x Lüsterklemme Anreihe-10 (Grün)
- 1x IC →Atmega 16 (40Pin)

- 1x IC Sockel (40Pin)
- 1x 16MHz Quarz
- 2x Kondensator 22pF
- 1x Kondensator 470uF
- 1x Kondensator 100nF
- 1x Widerstand 10kΩ
- 1x mini Taster
- 1x Lötnägel

4.8. LCD-Display

Das LCD-Display DEM 16481 SYH-LY der Marke Display Elektronik GmbH wird auf der Masterstation zum Anzeigen von Datum, Uhrzeit, Aussentemperatur und des CO2 Wertes benutzt. Dieses Display besitzt 16 Pins und kann 4x16 Karakteren anzeigen. Ausserdem besitzt das Display noch eine Hinterbeleuchtung.

4.8.1. Pinbelegung

1	VSS	GND
2	VDD	+5V
3	V0	+5V für das Flüssigkristall-Laufwerk
4	RS	Register wählen: 0 → Befehlsregister 1 → Datenregister
5	R/W	Read / Write: 0 → Write 1 → Read
6	E	R/W Freigabesignal
7	DB0	/
8	DB1	/
9	DB2	/
10	DB3	/
11	DB4	PC2
12	DB5	PC 3
13	DB6	PC 4
14	DB7	PC 5
15	LED – (K)	GND für die Hinterbeleuchtung
16	LED + (A)	+5V für die Hinterbeleuchtung

4.9. RS232

4.9.1. Einleitung

Die serielle Schnittstelle dient dem Datenaustausch zwischen Computern und Peripheriegeräten (hier: Mikrokontroller). Bei einer seriellen Datenübertragung werden die

Bits nacheinander (seriell) über eine Leitung übertragen. Wenn ohne nähere Kennzeichnung von einer „seriellen Schnittstelle“ gesprochen wird, ist damit fast immer die CCITT-V.24 bzw. RS-232 (heute EIA-232) Schnittstelle gemeint. Die Bedeutung und Anordnung der Bits wird bei der RS-232 mit einem sogenannten UART (Motorola-Bezeichnung) oder USART (Intel, Zilog, …) vorgegeben.

Die Datenübertragung erfolgt mit 9600Baud!

4.9.2. Schaltung

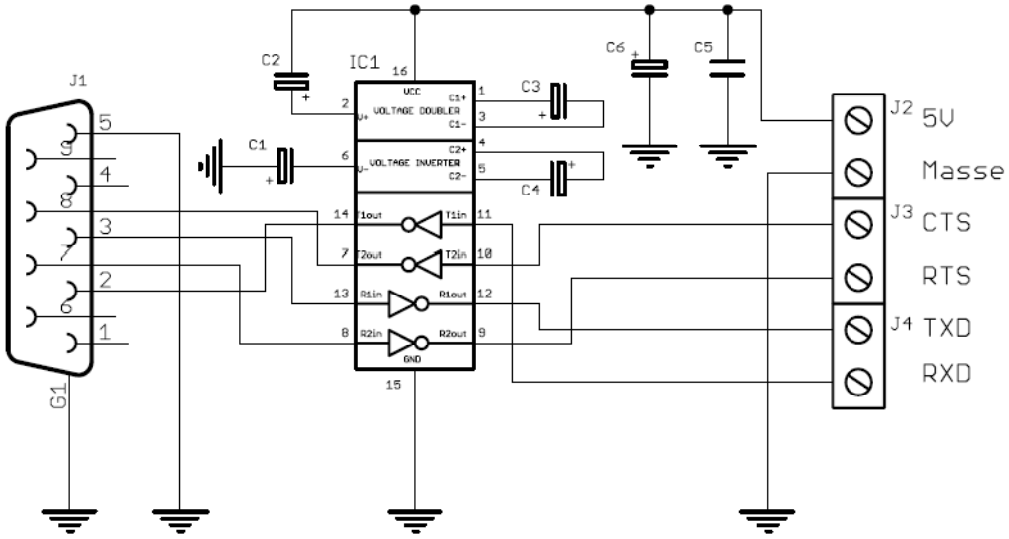

Abb. 22: Schalplan (RS232).
Quelle: http://www.pollin.de/shop/downloads/D810036B.PDF

4.9.3. Platine

Das Layout wurde im Eagle Layout Editor 5.10.0 der Firma Cadsoft gezeichnet.

Layout:

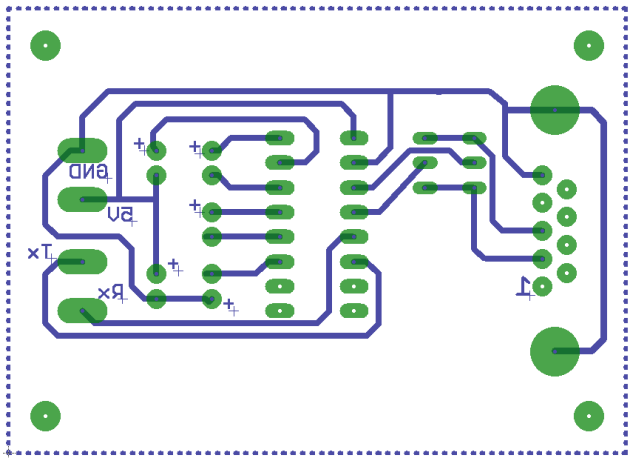

Abb. 23: Layout (RS232).

Bestückungsfolie:

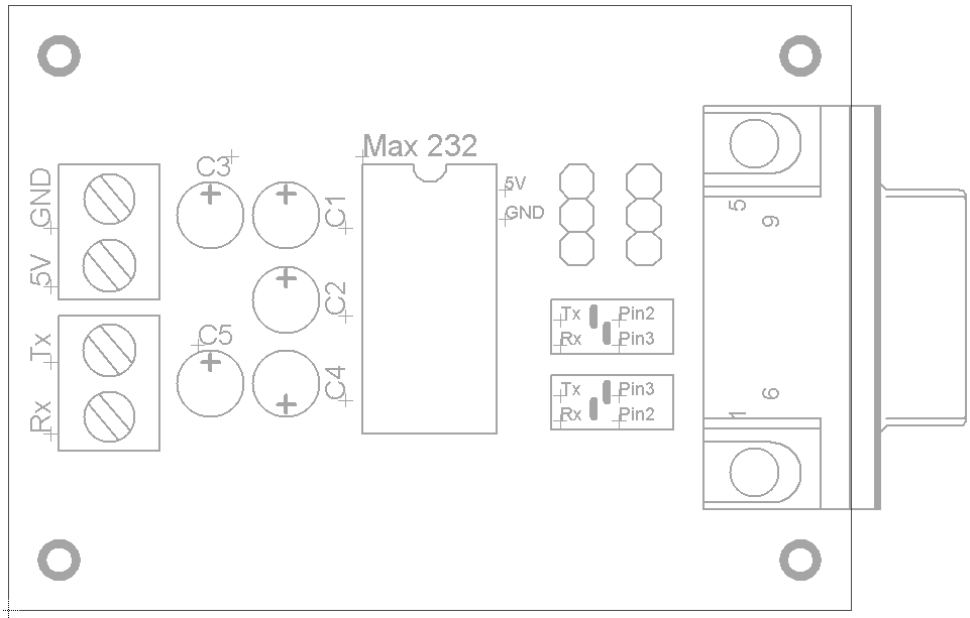

Abb. 24: Bestückungsfolie (RS232).

Platine:

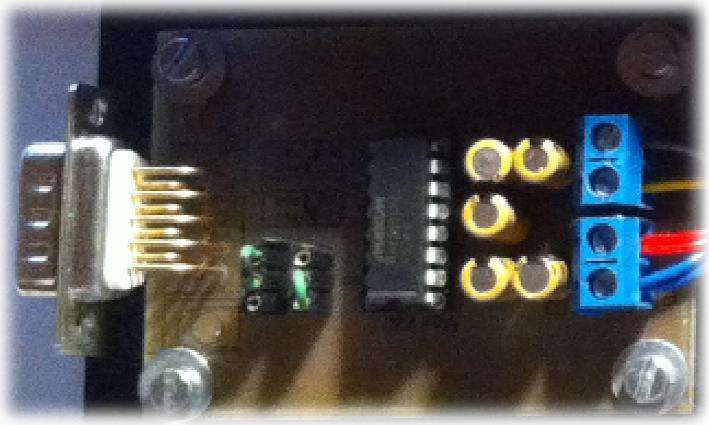

Abb. 25: Fertige Platine (RS232).

4.9.4. Bauteilliste

- Platine (60mm x 68mm)
- 1x IC → MAX 232 (16Pin)
- 1x IC Sockel (16Pin)
- 2x Lüsterklemme Anreihe-2 (Blau)
- 2x Jumper (3Pin)
- 1x 9Poliger SUB-D Stecker
- 5x Kondensator

4.9 Computer

Die Schule stellte uns einen PC zur Verfügung, den wir in einen trocknen Raum im Musée des Mines installieren konnten. Herr Feltes hatte diesen bereits im Gebrauch um seine Messstation zu betreiben.
Mit dem PC werden die vom Master erhaltene Daten in einen Terminal angezeigt und nach und nach gespeichert, damit man diese beispielsweise in Excel auswerten, und dementsprechende Kennlinien zeichnen kann.

Als Terminal kann man den vom Bascom integrierten Terminal benutzen, oder den „wxterm“, mit dem man auch hervorragend arbeiten kann.
Man muss nur beachten, dass man den Terminal auf 9600 Baud, die Datenbits auf 8, die Partität auf „None“ und die Stoppbits auf 1 einstellt. Des weiteren muss man bei Input „EOS is: none“ und „Input is: raw“ einstellen.

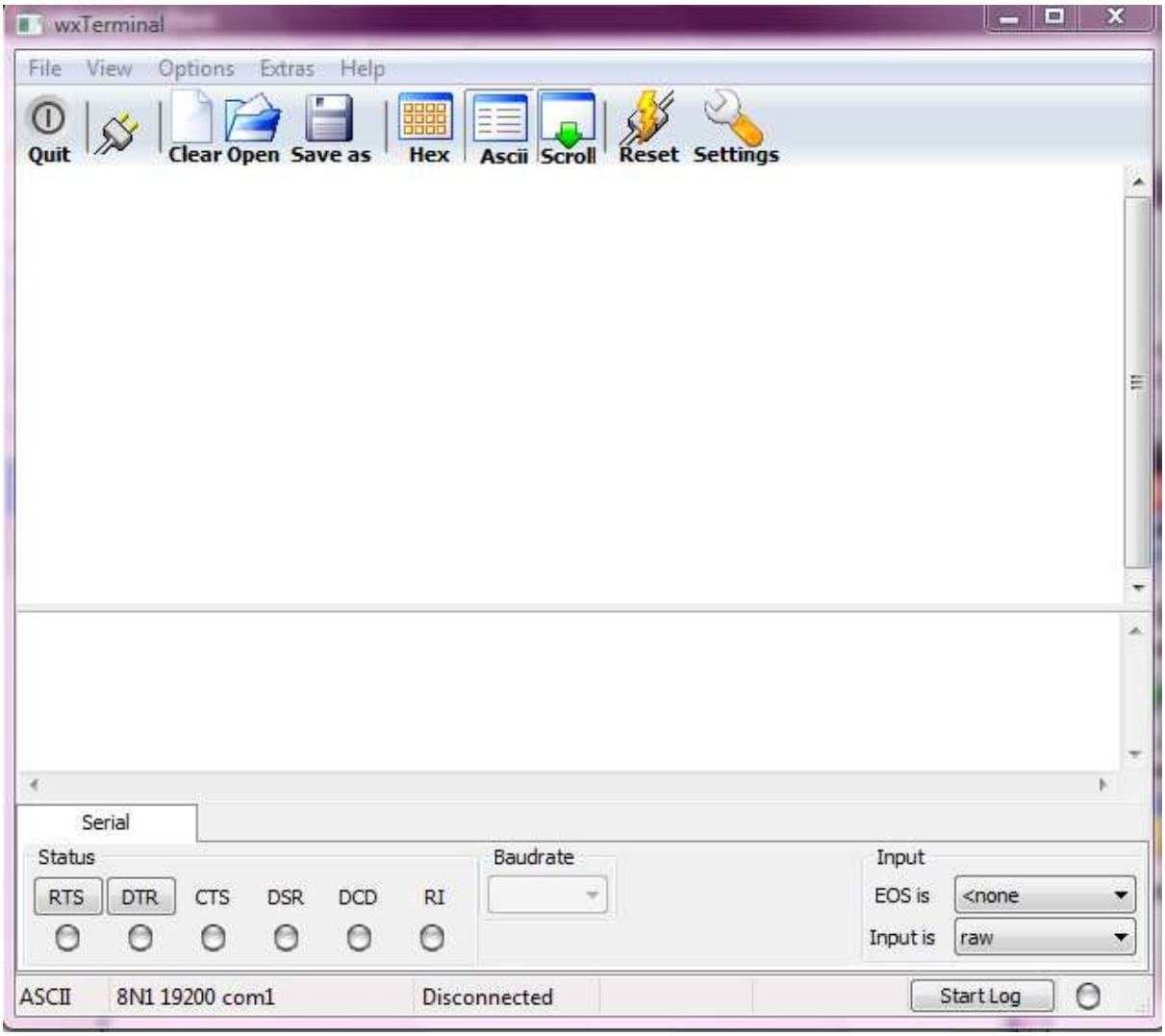

Abb. 26: „wxterm“-Terminal.

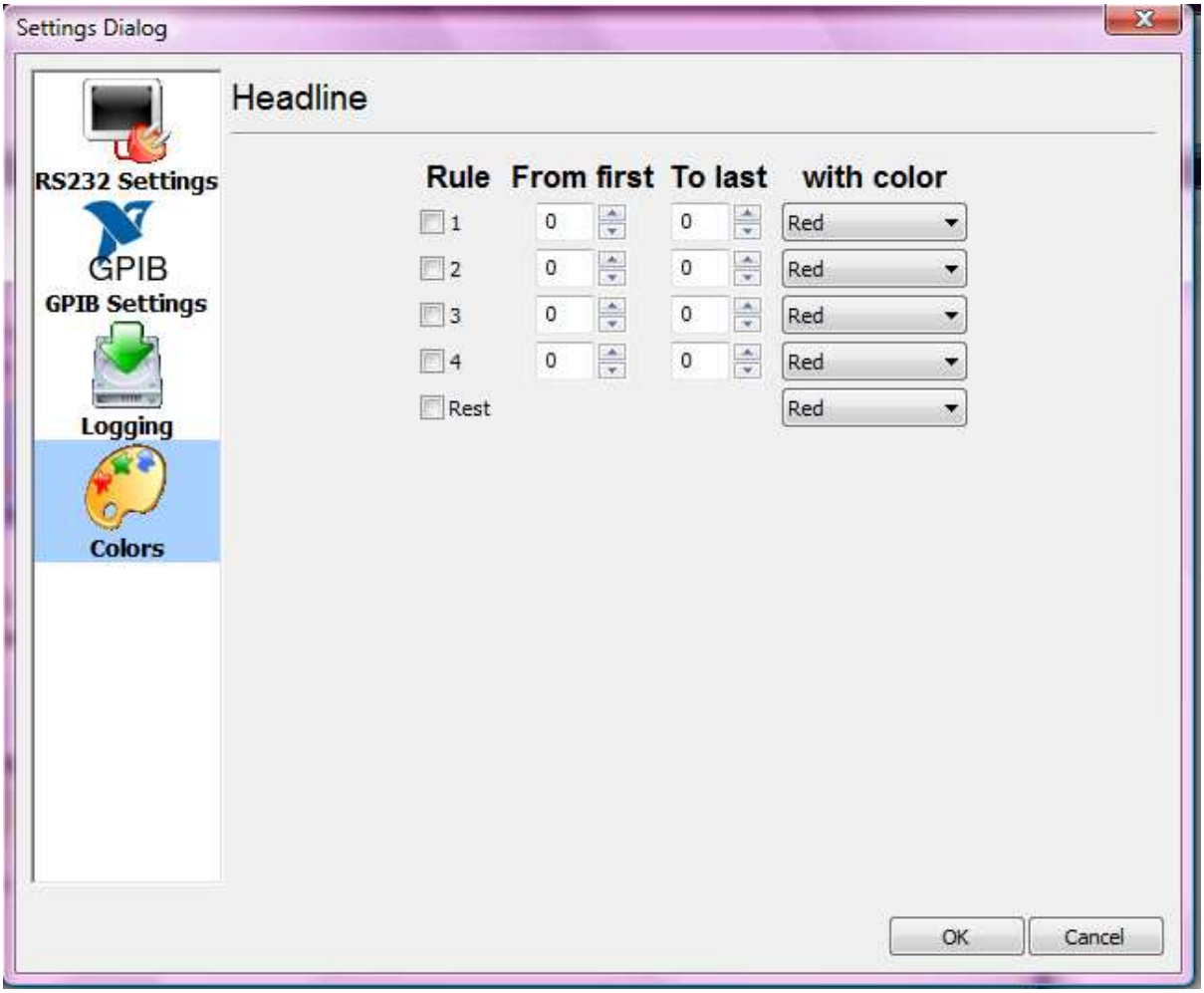

Abb. 27: „wxterm“-Settings.

4.10 Kommunikation

Damit die Masterstation mit der Messstation kommunizieren kann, und der Mikrokontroller der Masterstation jedoch nur eine serielle Hardware-Schnittstelle zur Verfügung hat, mussten wir eine serielle Software-Schnittstelle programmieren. Dies geschieht mit folgenden zwei Zeilen:

```
Open "comd.4:9600,8,n,1" For Output As #1
Open "comd.3:9600,8,n,1" For Input As #2
```

Output ist für das Senden zuständig und Input für das Empfangen, d.h. Output ist gleichgestellt mit TX (Transceive) und Input mir RX (Receive).

Damit man nun auch Senden und Empfangen kann, dienen diese zwei Zeilen:

```
Print #1, “*“
Input #2, In(i)
```

Da wir nun im Rhythmus von z.B. 15 Sekunden immer wieder an der Messstation die Daten anfragen wollen, mussten wir im Programm der Masterstation einen Timer einbauen

der jede Sekunde einen Flag inkrementiert. Ausserdem brauchen wir den Timer noch um jede Sekunde die Uhrzeit und das Datum auf dem LCD anzuzeigen und eine LED im Sekundentakt zu togglen.

Um nun den Timer auf exakt eine Sekunde festlegen zu können, mussten wir folgende Formel anwenden:

$$f = \frac{\text{Systemtakt}}{\text{Vorteiler} \cdot \text{Zählschritte}}$$

Zählschritte = 1Sekunde = 1Hz.
Vorteiler zb 1024.
Der Systemtakt beträgt 16Mhz.

Das Resultat beträgt 15625Hz.

Als Programmcode sieht dies dann folgendermassen aus:

```
'Timer 1s-------------------------------------------------------------------

Config Pina.7 = Output                                          'LED 1s
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1 , Compare A = Disconnect
Ocr1ah = High(15625)
Ocr1al = Low(15625)
Tccr1a = 0
On Compare1a Tim1_1s
Enable Compare1a
Enable Interrupts

'UNTERPROGRAMME-------------------------------------------------------------
Tim1_1s:

   Onesecflag = 1                           'Flag für Clock setzen
   Incr Masterseconds

   Toggle Porta.7

   Incr J
   If J = 20 Then                            'jede 15 Sekunden Commandflag setzen
      Commandflag = 1
      J = 0
   End If

Return
```

Damit nun auch die Messstation mitbekommt, dass die Masterstation die Daten senden will, arbeiteten wir hier mit einem seriellen Interrupt, was heißt, dass die Messstation einen Interrupt (Unterbrechung) auslöst (das Programm erledigt den letzen angefangenen Befehl und springt dann in die Interruptroutine) und dem Master die Daten sendet. Wenn dies erledigt ist, springt das Programm wieder dorthin zurück wo es stehen geblieben ist. Auf dieser Station wird nicht mit der softwaremässigen seriellen Schnittstelle gearbeitet, sondern wieder mit der hardwaremässigen.

Damit der serielle Interrupt auch auftreten kann müssen zuerst folgende drei Zeilen geschrieben werden:

```
'SERIALINTERRUPT-----------------------------------------------------------

On Urxc Serialinterrupt
Enable Urxc ‘serieller Interrupt freischalten
Enable Interrupts ‘globale Interrupts freischalten
```

Die Interruptroutine sieht dann folgendermassen aus:

```
'--------------------------------------------------------------------------------
Serialinterrupt:

   Ok = Udr
   Select Case Ok
       Case "*"
       Sendeflag = 1
   End Select

Return
```

Da die Interruptroutinen (Timer, serieller Interrupt, …) immer so kurz wie möglich sein sollen, arbeiteten wir sehr viel mit Flag’s. Das heisst, wenn ein Interrupt auftritt, wird meistens ein Flag auf logisch „1“ gesetzt und erst im Hauptprogramm wird mit diesen Flags weitergearbeitet. Das Hauptprogramm sieht dann beispielsweise wie folgt aus:

```
'HAUPTPROGRAMM-------------------------------------------------------------

Cls
Commandflag = 1

Do

   If Commandflag = 1 Then
      Gosub Co2_messen
      Gosub Temperaturlesung
      Gosub Lcd_anzeige
      Commandflag = 0
   End If

   If Sendeflag = 1 Then
      Waitms 10
      'write
      Portd.2 = 0
      Waitms 10

      Gosub Terminal

      Sendeflag = 0
      Waitms 10
      'read
      Portd.2 = 1
   End If

Loop

End
```

4.11 Master-Programm

```
'*********************************************************************************
'MUSEE DES MINES RUMMELANGE
'PROJET T3 EC+EE 2010/2011
'BETREUER: JEAN-CLAUDE FELTES
'*********************************************************************************
'MASTER-STATION (RTC, LCD, Terminal, Tristate R/W)
'*********************************************************************************

'uC-INIT-----------------------------------------------------------------------

$regfile = "m16def.dat"
$crystal = 16000000
$baud = 9600
$hwstack = 100
$swstack = 100
$framesize = 100

'DISPLAY-----------------------------------------------------------------------

Config Lcd = 16 * 4
Config Lcdpin = Pin , Db4 = Portc.3 , Db5 = Portc.2 , Db6 = Portd.7 , Db7 = Portd.6
, E = Portc.4 , Rs = Portc.5
Cursor Noblink
Cursor Off

'I2C---------------------------------------------------------------------------

Config Sda = Portc.1
Config Scl = Portc.0

'RTC INIT----------------------------------------------------------------------

Dim S$ As String * 8
Dim Weekday As Byte
Config Clock = User
Config Date = Dmy , Separator = .

'VARIABLEN DECLARIEREN---------------------------------------------------------

Dim In(7) As String * 10
Dim I As Byte
Dim J As Byte

Dim Slaveseconds As Long
Dim Temperature_outside As String * 10
Dim Temperature_inside As String * 10
Dim Temperature_inside_near As String * 10
Dim Onflag As Byte
Dim Unknow As Byte
Dim Co2value As Word

Dim Commandflag As Bit
Dim Onesecflag As Bit
Dim Masterseconds As Long
```

```
Masterseconds = 0
Dim Ok As Byte

'SOFTWARE-UART-------------------------------------------------------------------

Open "comd.4:9600,8,n,1" For Output As #1
Open "comd.3:9600,8,n,1" For Input As #2

'TRISTATE------------------------------------------------------------------------

Config Portd.2 = Output
Portd.2 = 1                                                 '1 = read

'Timer 1s------------------------------------------------------------------------

Config Pina.7 = Output                                      'LED 1s
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1 , Compare A = Disconnect
Ocr1ah = High(15625)
Ocr1al = Low(15625)
Tccr1a = 0
On Compare1a Tim1_1s
Enable Compare1a
Enable Interrupts

'HAUPTPROGRAMM-------------------------------------------------------------------

Cls
Commandflag = 1

Print "Date" ;
Print "  " ;
Print "Time" ;
Print "  " ;
Print "Master_Sec" ;
Print "  " ;
Print "Slave_Sec" ;
Print "  " ;
Print "Temp_Out" ;
Print "  " ;
Print "Temp_In";
Print "  " ;
Print "Temp_In_near";
Print "  " ;
Print "Heizung" ;
Print "  " ;
Print "Unknow";
Print "  " ;
Print "Co2"

Do

   If Commandflag = 1 Then
      Gosub Empfangen_slave
   End If

   If Onesecflag = 1 Then                'Zeit anzeigen wenn 1s vorbei
      Gosub Lcdclock
      Onesecflag = 0
```

```
   End If

   Ok = Inkey()

   If Ok <> "" Then Gosub Tasten

   Waitms 10

Loop

End

'UNTERPROGRAMME-----------------------------------------------------------------
Tim1_1s:

   Onesecflag = 1                                  'Flag für Clock setzen
   Incr Masterseconds

   Toggle Porta.7

   Incr J
   If J = 20 Then                                  'jede 15 Sekunden Commandflag
setzen
      Commandflag = 1
      J = 0
   End If

Return

'-------------------------------------------------------------------------------
Empfangen_slave:

   'write
   Portd.2 = 0
   Waitms 5

   Print #1 , "*"
   Waitms 10

   'Read
   Portd.2 = 1

   For I = 1 To 7
      Input #2 , In(i)
   Next I

   For I = 1 To 7
      If Left(in(i) , 1) = Chr(10) Then In(i) = Mid(in(i) , 2)
   Next I

   Slaveseconds = Val(in(1))
   Temperature_outside = In(2)
   Temperature_inside = In(3)
   Temperature_inside_near = In(4)
   Onflag = Val(in(5))
   Unknow = Val(in(6))
   Co2value = Val(in(7))
   Gosub Lcd_display
```

```
   Gosub Terminal

   Commandflag = 0

Return

'--------------------------------------------------------------------------------
Tasten:

Select Case Ok

       Case "s"
         Input "hh:mm:ss" , S$                              'mit Variable arbeiten,
da sonst Time$ / Date$ nicht gesetzt wird
            Time$ = S$
         Input "dd.mm.yy" , S$
            Date$ = S$

       Case "v"
         Commandflag = 1

End Select
Return

'--------------------------------------------------------------------------------
Lcd_display:

   Locate 3 , 1 : Lcd "                "
   Locate 3 , 1 : Lcd "Temp_Out: " ; Temperature_outside ; "C"
   Locate 4 , 1 : Lcd "                "
   Locate 4 , 1 : Lcd "Co2: " ; Co2value ; "ppm"

Return

'--------------------------------------------------------------------------------

Lcdclock:

   Locate 1 , 1 : Lcd "Datum: " ; Date$
   Locate 2 , 1 : Lcd "Zeit:  " ; Time$

Return

'--------------------------------------------------------------------------------

Terminal:

Print Date$ ;
Print "  " ;
Print Time$ ;
Print "  " ;
Print Masterseconds ;
Print "  " ;
Print Slaveseconds ;
Print "  " ;
Print Temperature_outside ;
Print "  " ;
Print Temperature_inside ;
```

```
Print "  " ;
Print Temperature_inside_near ;
Print "  " ;
Print Onflag ;
Print "  " ;
Print Unknow;
Print "  " ;
Print Co2value

Return

'--------------------------------------------------------------------------------
Getdatetime:

   I2cstart
   I2cwbyte &HD0
   I2cwbyte 0
   I2cstop

   I2cstart
   I2cwbyte &HD1
   I2crbyte _sec , Ack
   I2crbyte _min , Ack
   I2crbyte _hour , Ack
   I2crbyte Weekday , Ack
   I2crbyte _day , Ack
   I2crbyte _month , Ack
   I2crbyte _year , Nack
   I2cstop

   _sec = Makedec(_sec)
   _min = Makedec(_min)
   _hour = Makedec(_hour)
   _day = Makedec(_day)
   _month = Makedec(_month)
   _year = Makedec(_year)

Return

'--------------------------------------------------------------------------------
Setdate:

   _day = Makebcd(_day)
   _month = Makebcd(_month)
   _year = Makebcd(_year)

   I2cstart
   I2cwbyte &HD0
   I2cwbyte 4
   I2cwbyte _day
   I2cwbyte _month
   I2cwbyte _year
   I2cstop

Return

'--------------------------------------------------------------------------------
Settime:
```

```
   _sec = Makebcd(_sec)
   _min = Makebcd(_min)
   _hour = Makebcd(_hour)

   I2cstart
   I2cwbyte &HD0
   I2cwbyte 0
   I2cwbyte _sec
   I2cwbyte _min
   I2cwbyte _hour
   I2cstop

Return

'--------------------------------------------------------------------------------
```

4.12 Mess/Slave-Programm

```
'********************************************************************************
'MUSEE DES MINES RUMMELANGE
'PROJET T3 EC+EE 2010/2011
'BETREUER: JEAN-CLAUDE FELTES
'********************************************************************************
'MESS-STATION (CO2, Temperatur IN/IN_Near/OUT + Regelung, Tristate R/W, LCD)
'********************************************************************************

'uC-INIT---------------------------------------------------------------------

$regfile = "m8def.dat"
$crystal = 8000000
$baud = 9600
$hwstack = 100
$swstack = 100
$framesize = 100

'DISPLAY---------------------------------------------------------------------

'Config Lcd = 20 * 2
'Config Lcdpin = Pin , Db4 = Portb.2 , Db5 = Portb.3 , Db6 = Portb.4 , Db7 =
Portb.5 , Rs = Portb.0 , E = Portb.1
'Cursor Noblink
'Cursor Off

'CO2-------------------------------------------------------------------------

Config Scl = Portc.4                                        'Pin0 am PortD als
Taktleitung deklarieren
Config Sda = Portc.5                                        'Pin1 am PortD als
Dateneingang deklarieren
                                                             '!!! Niemals die Pins
am selben LCD-Pins benutzen sonst wird die Werte verfälscht !!!
Config Pinc.3 = Input                                       'Definiert Pin4 am
PortD als Eingang
Set Portc.3                                                 'Pull-Up Widerstand
einschalten
```

```
Dim Operation_status As Byte                                'Variable für den
Operation Status Byte
Dim Highbyte As Integer                                     'Variable für den
Highbyte
Dim Lowbyte As Byte                                         'Variable für den
Lowbyte
Dim Check_sum As Byte                                       'Variable für den Check
Sum Byte
Dim Check_sumcompare As Byte                                'Variable für den
Vergleichwert Check SumCompare
Dim Co2 As Word                                             'Variable für den Co2
Wert
Dim I As Integer                                            'Variable für den
Zähler Korrektur I
Dim Multiplicator As Integer                                'Variable für den
Multiplicator

'TEMPERATURLESUNG-----------------------------------------------------------

Config 1wire = Portd.5
Config 1wire = Portd.6
Config 1wire = Portd.7

Dim Sci(9) As Byte
Dim Ti As Integer
Dim Temp As Single
Dim Thpinnr As Byte
Dim Temperatur_out As Single
Dim Temperatur_inmc As Single
Dim Temperatur_inheiz As Single
Dim Temperatur_out_rund As String * 5
Dim Temperatur_inmc_rund As String * 5
Dim Temperatur_inheiz_rund As String * 5                    'Pull-Up Widerstand
einschalten

'TEMPERATURREGLUNG----------------------------------------------------------

Config Pinc.0 = Output
Dim Onflag As Bit

'VARIABLEN DECLARIEREN------------------------------------------------------

Dim Sendeflag As Bit
Dim Commandflag As Bit
Dim Ok As Byte
Dim Slaveseconds As Long
Slaveseconds = 1

'TRISTATE-------------------------------------------------------------------

Config Portd.2 = Output
Portd.2 = 1                                                 '1 = read

'Timer 1s-------------------------------------------------------------------

Config Pind.4 = Output                                      'LED 1s
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1 , Compare A = Disconnect
Ocr1ah = High(7812.5)
```

```
Ocr1al = Low(7812.5)
Tccr1a = 0
On Compare1a Tim1_1s
Enable Compare1a
Enable Interrupts

'SERIALINTERRUPT-----------------------------------------------------------------

On Urxc Serialinterrupt
Enable Urxc

'HAUPTPROGRAMM-------------------------------------------------------------------

Cls
Commandflag = 1

Do

   If Commandflag = 1 Then
      Gosub Co2_messen
      Gosub Temperaturlesung
      Gosub Lcd_anzeige
      Commandflag = 0
   End If

   If Sendeflag = 1 Then
      Waitms 10
      'write
      Portd.2 = 0
      Waitms 10

      Gosub Terminal

      Sendeflag = 0
      Waitms 10
      'read
      Portd.2 = 1
   End If

Loop

End

'UNTERPROGRAMME------------------------------------------------------------------
Tim1_1s:

   Toggle Portd.4
   Commandflag = 1

   Incr Slaveseconds

   If Temperatur_inheiz < 20 Then
      Set Portc.0
      Onflag = 1
   Else
      Reset Portc.0
      Onflag = 0
   End If
```

```
Return

'--------------------------------------------------------------------------------
Serialinterrupt:

   Ok = Udr
   Select Case Ok
       Case "*"
       Sendeflag = 1
   End Select

Return

'--------------------------------------------------------------------------------
Co2_messen:

   Gosub Choose_k30k33_sr                                  'Wähle Modus für K30
oder K33
   Gosub Check_ops_chs_sr                                  'Ermittle Operation
Byte und Check Sum Byte
   Gosub Correction_sr                                     'Korrekturverfahren mit
Operation Byte und Check Sum Byte
   Gosub Co2_wert_sr                                       'Berechne Co2 Wert
                                                           'Ausgabe an der LCD
Display
   Waitms 50

Return

'--------------------------------------------------------------------------------
Temperaturlesung:

   Gosub Temperatur_out_sr
   Gosub Temperatur_inmc_sr
   Gosub Temperatur_inheiz_sr
   Gosub Runden_sr

Return

'--------------------------------------------------------------------------------
'Unterprogramme CO2-------------------------------------------------------------
'--------------------------------------------------------------------------------
Choose_k30k33_sr:

   If Pinc.3 = 0 Then                                      'Falls Pin4 am PortD am
Masse liegt (Jumper)
      Multiplicator = 10
   Else
      Multiplicator = 1
   End If

Return

'--------------------------------------------------------------------------------
Check_ops_chs_sr:

   I2cstart                                                'Start Bedingung
   I2cwbyte &B11010000                                     'schicke Adressbit
```

```
I2cwbyte &B00100010                                    '1. Nibbel :schicke Befehl
2: lese aus Ram 2. Nibbel : Inhaltspeicherplatz betragt 2 Byte
   I2cwbyte &B00000000
   I2cwbyte &B00001000                                    'Lese im Adress im Ram
:0x08
   I2cwbyte &B00101010                                    'Checksumme
   I2cstop                                                'Stop Bedingung

   Waitms 50                                              'Warte 50ms
   I2cstart                                               'Start Bedingung
   I2cwbyte &B11010001                                    'spreche Slave an und
schicke Adressbit mit Lese Befehl
   I2crbyte Operation_status , Ack                        'lese aus variable
Operation_status
   I2crbyte Highbyte , Ack                                'lese Highbyte
   I2crbyte Lowbyte , Ack                                 'lese Lowbyte Der CO2-
Wert besteht aus den Highbyte & Lowbyte
   I2crbyte Check_sum , Nack                              'lese Check_sum
   I2cstop                                                'Stop Bedingung

   Check_sumcompare = Operation_status + Highbyte         'Check_sumcompare
besteht aus den Addition von 1. - 3. Byte
   Check_sumcompare = Check_sumcompare + Lowbyte          'Check_sumcompare wird
berechnen für eine späteren Vergleich mit Check_sum

Return

'--------------------------------------------------------------------------
Correction_sr:

   If Operation_status = 1 Or Check_sum <> Check_sumcompare Then       'Bedingung:
Falls Operation Status Byte = 1 ODER Check Sum Byte ist
      For I = 1 To 10                                     'nicht gleich wie der
berechnete Check Sumcompare dann Schleife bis 10
      Gosub Check_ops_chs_sr
   If Operation_status = 0 And Check_sum = Check_sumcompare Then Exit For
'Ausnahme für die Schleife: Operation Status Byte = 0 UND
      Next I                                              'Check Sum Byte = Check
Sumcompare
   Else
      I = 0                                               'Zähler wird zurück
gesetzt
   End If

Return

'--------------------------------------------------------------------------
Co2_wert_sr:

   Highbyte = Highbyte * 256                              'Highbyte berechnen
   Co2 = Highbyte + Lowbyte                               'Co2-Werte berechnen
   Co2 = Co2 * Multiplicator

Return

'--------------------------------------------------------------------------
'Unterprogramme Temperaturlesung-------------------------------------------
'--------------------------------------------------------------------------
```

```
Temperatur_out_sr:

    Thpinnr = 6
    Gosub Temperaturmessung_sr
    Temperatur_out = Temp

Return

'-------------------------------------------------------------------------------
Temperatur_inmc_sr:

    Thpinnr = 7
    Gosub Temperaturmessung_sr
    Temperatur_inmc = Temp

Return

'-------------------------------------------------------------------------------
Temperatur_inheiz_sr:

    Thpinnr = 5
    Gosub Temperaturmessung_sr
    Temperatur_inheiz = Temp

Return

'-------------------------------------------------------------------------------
Temperaturmessung_sr:

   'Mißt Temperatur
   'Variablen benutzt: "Thpinnr" Bit des Portes des Sensors, muss vorher gesetzt
werden
                       '"Sc1" Ausgelesener Wert
                       '"Ti"  Integer der Temperatur
   'Ausgabe in Variable "Temp" Temperatur in dezimaler Zahl


   'Messen
   1wreset Pind , Thpinnr                                  'Reset 1Wire
   1wwrite &HCC , 1 , Pind , Thpinnr                       'skip rom alle
Temperatursensoren
   1wwrite &H44 , 1 , Pind , Thpinnr                       'Convert T alle
Temperatursensoren
   Waitus 200                                              'Warte 200 us

   'Auslesen
   1wreset Pind , Thpinnr                                  'reset 1Wire
   1wwrite &HCC , 1 , Pind , Thpinnr                       'skip rom
Temperatursensor innen
   1wwrite &HBE , 1 , Pind , Thpinnr                       'read scratchpad
Temperatursensor innen

   'Auswertung
   Sci(1) = 1wread(9 , Pind , Thpinnr)
   Ti = Makeint(sci(1) , Sci(2))                           'interger aus sci(1)
und sci(2)
   Temp = Ti                                               'aus dem interger wird
ein single gemacht
```

```
Temp = Temp / 16                                          'der wert wird durch 16
geteilt um den richtigen dezimal wert anzuzeigen

Return

'--------------------------------------------------------------------------------
Runden_sr:

  Temperatur_out_rund = Fusing(temperatur_out , "#.##")
  Temperatur_inmc_rund = Fusing(temperatur_inmc , "#.##")
  Temperatur_inheiz_rund = Fusing(temperatur_inheiz , "#.##")

Return

'--------------------------------------------------------------------------------
Lcd_anzeige:

Config Lcd = 20 * 2
Config Lcdpin = Pin , Db4 = Portb.2 , Db5 = Portb.3 , Db6 = Portb.4 , Db7 = Portb.5
, Rs = Portb.0 , E = Portb.1
Cursor Noblink
Cursor Off
Cls

   Locate 1 , 1 : Lcd "                    "
   Locate 1 , 1 : Lcd "O:" ; Temperatur_out_rund ; "C"
   Locate 1 , 11 : Lcd "I:" ; Temperatur_inmc_rund ; "C"
   Locate 2 , 1 : Lcd "                    "
   Locate 2 , 1 : Lcd "H:" ; Temperatur_inheiz_rund ; "C"
   Locate 2 , 11 : Lcd Co2 ; "ppm"

Return

'--------------------------------------------------------------------------------
Terminal:

   Print Slaveseconds
   Print Temperatur_out_rund
   Print Temperatur_inmc_rund
   Print Temperatur_inheiz_rund
   Print Onflag
   Print "none"
   Print Co2

Return

'--------------------------------------------------------------------------------
```

5 Verwendeter Kabel

Als Kommunikations Kabel zwischen Master- und Messstation verwenden wir aus Kostengründen ein 3-adriges Installationskabel. Die braune Ader wird für +24V, die blaue für die Masse und die gelb/grüne wird für den Datenaustausch verwendet. Die beiden Enden wurden jeweils mit einem 3 poligem XLR-Stecker versehen.

Abb. 28: Kabeltrommel (Vorarbeiten im Musée des mines).