PROFE1:

SEISMOGRAF

Name: PAQUET Charel
Klasse: T3EE
Datum: 30.1.15

1)Inhaltsverzeichnis

Table of Contents

2)Einleitung

In diesem Bericht geht es um mein Abschlussprojet der Klasse T3EE, ein Seismometer. Ein Seismometer ist ein Sensor der Bodenschwingungen erfasst und sie analysiert und Messergebnisse liefert. Die Messergebnisse haben die Einheit g oder . Ein Seismometer kann Erdbeben wahrnehmen. Ein Erdbeben meldet sich erst mit einer Welle vor, die P-Welle. Die P-Welle hat eine Geschwindigkeit von bis zu 8km/s. Mein Modul erfasst die Beschleunigung mit dem Sensor BMA180 von Siemens. Dieser Sensor erfasst 3-dimensionale Beschleunigungen auf der X;Y;Z-Achse.

3)Aufgabenstellung

Für das Projekt 2014/2015 soll eine neue Messstation entwickelt werden. Mein Modul ist ein Seismometer. Das Modul wird an den Bus des Interface 2 angeschlossen. Die Steuerung erfolgt über eine serielle Schnittstelle über die Signale RxD und TxD. Zur Kontrolle der Übertragung wird ein Signal (Write/Read) benutzt. Dieses kontrolliert die Datenrichtung. Außerdem sollen einige Spezifikationen getroffen werden für jedes Modul. Diese wären:

-Über einen Jumper (JP1) wird die Baudrate eingestellt.

Wenn Jumper 1 gesetzt ist, dann wäre die Baudrate 2400 . Wenn nicht, dann wäre die Baudrate 9600.

- Jedes Modul hat eine Adresse, diese kann über DIP-Switches eingestellt werden. Die Adresse soll ein druckbares ASCII-Zeichen sein. Zum Beispiel Hex (80) = P. So kann man auch per Tastatur im Terminal Befehle eingeben.

-Das Interface 2 kann mit einem Modul kommunizieren und Befehle erteilen über der serielle Schnittstelle, die bestehen aus drei Bytes:

Das erste Byte ist ein Synchronisierbyte “!“ = 33 = 0x21.

Das zweite Byte ist die Adresse des Moduls.

Das dritte Byte ist ein Kommando. Es gibt drei Kommandos:

“i“ = sende Modulinfo

“s“ = sende Daten

“r“ = Reset des Moduls

-Die Messwerte oder andere Daten des Moduls werden normalerweise nur per Kommando geschickt, aber durch einen zweiten Jumper (JP2) kann entschieden werden ob das Modul per Kommando schickt oder jede Sekunde. Dies erleichtert das Testen der Schaltung. Im normalen Betrieb schickt das Modul die Daten nur auf Kommando.

Wenn Jumper 2 gesetzt ist, wird kontinuierlich schicken

Wenn Jumper 2 nicht gesetzt ist, wird nur auf Kommando geschickt.

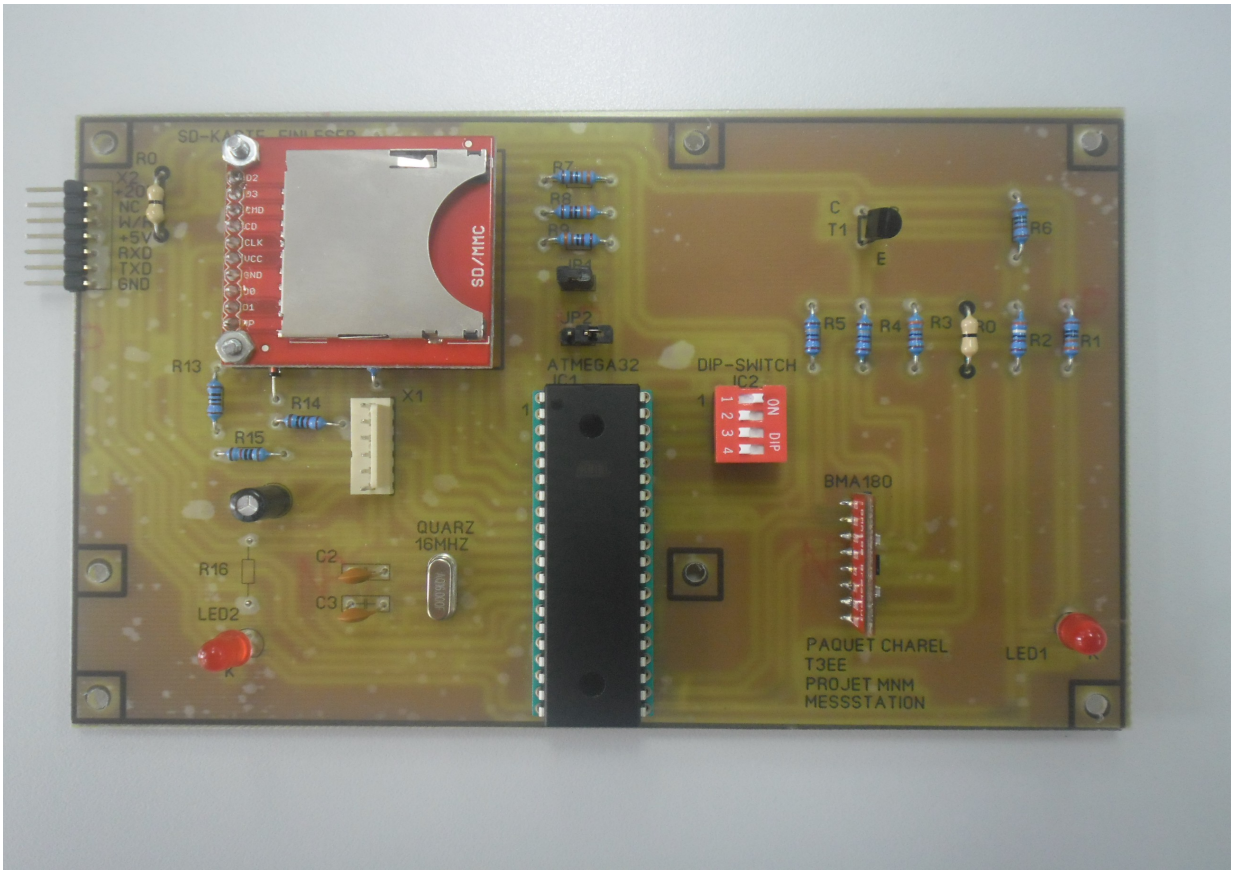
-Die Aktivität des Mikroocontrollers wird über den Watchdog überwacht.

-Das Modul hat einen Alive-Ticker, der jede Sekunde inkrementiert wird. So kann man sehen ob das Modul aktiv oder abgestürtzt ist.

-Die Daten werden in Textform gesendet und durch Tabulatorzeichen (0x09) getrennt. Am Ende wird “EOD“ (End of Data) und ein Zeilenvorschub gesendet (0x0D). Zu Beginn wird zusätzlich eine Information über die Daten im Klartext gesendet, sie ist durch Vorsetzen einer “Raute“ (#) als Kommentar gekennzeichnet.

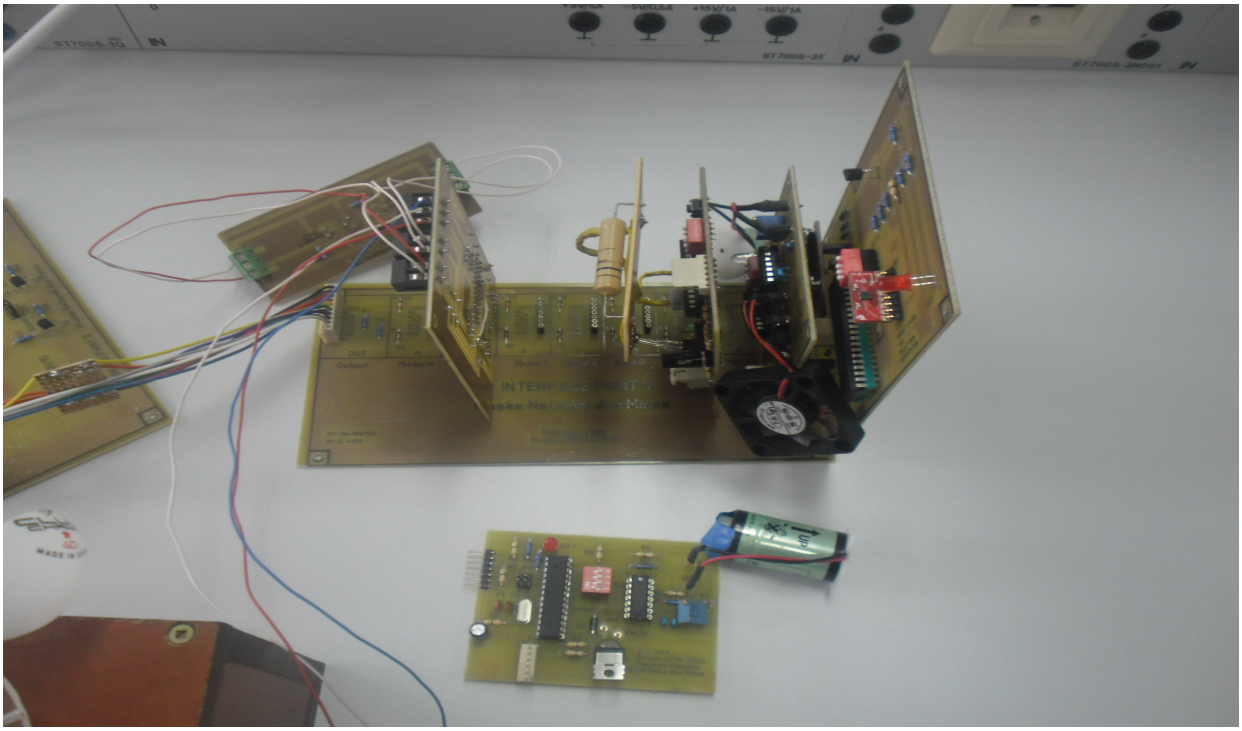
-Zur Kontrolle der Übertragung soll das Modul einen Steuerausgang W/R (Write/Receive) haben, dieser wird vor dem Übetragen auf High gesetzt, solange bis Daten geschickt werden. Genauer gesagt soll vor dem Senden, der W/R- Ausgang 10 ms auf High liegen und 10 ms nach der Übertragungauf High bleiben.

4)Hardware



Mein Modul

Messtation mit alle Modulen



a) Schaltplan

Meine Schaltung besteht aus 5 Teile:

-Kollektorschaltung

-SD-Karte

-Programmieradapter

-2 Jumper

-1 Adresseeingabe

Gesamtschaltung:

Kollektorschaltung:

$$\Re = \frac{UCE}{IE} = \frac{3{,}3\,V}{1\text{mA}} = 3{,}3\,k\,\Omega$$

$$B = \frac{Ie}{IB} = IB = \frac{1\text{mA}}{50} = 0{,}002\,mA$$

$$URB2 = UBE + URE = 0{,}7\,V + 3{,}3\,V = 4\text{V}$$

$$URB1 = UB - URB2 = 5\text{V} - 4\text{V} = 1\text{V}$$

$$RB2 = \frac{URB2}{IRB2} = \frac{4\text{V}}{1\text{mA}} = 4\text{k}\,\Omega$$

$$RB1 = \frac{URB1}{IRB1} = \frac{1\text{V}}{1{,}002\,mA} = 980\,\Omega$$

Die Kollektorschaltung wandelt die Spannung von 5V auf 3,3V. Ich musste eine Kollektorschaltung entwickeln, weil der BMA180 eine Belastung darstellt. Deswegen konnte ich kein Spannungteiler benutzen.

SD-Karte:

Diese Schaltung hab ich von einer PDF-Datei von Herr Feltes auf seiner Seite gefunden.

Programmieradapter:

Das benutzte Programmieradapter ist der AVR ISP mkII. Dies hat den Vorteil, dass man den Microcontroller nicht raus nehmen muss, sondern sofort programmieren kann.

Jumper:

Der eine Jumper ist für die Einstellung der Baudrate und der andere Jumper ist für dauerhaftes senden oder warten auf Kommando.

Adresseingabe:

Die Adresseingabe erfolgt über einen Switch. Die Adresse des Moduls ist der Buchstabe A.

b) Layout und Bestückungsplan und Bauteilliste

Bauteilliste

Bezeichnung	Anzahl und Info	Bauteil
R0	2X ; 0 Ohm	Metallschichtwiderstand
R1/R16	2X ; 330 Ohm	Metallschichtwiderstand
R2	1X; 4k Ohm	Metallschichtwiderstand
R3	1X ; 3,3 k Ohm	Metallschichtwiderstand
R4/R5/R10/R11/R12	5X ; 1,8 k Ohm	Metallschichtwiderstand
R6/R13/R4	3X ; 1 k Ohm	Metallschichtwiderstand
R7/R8/R9	3X ; 3,3 k Ohm	Metallschichtwiderstand
R15	1X ; 10 k Ohm	Metallschichtwiderstand
X1	1X ; 6 Stifte	Adaptersockel
X2	1X ; 7 schräg	Stiftleisten
C1	1X ;	Elcokondensator
C2/C3	2X ; 22p	Keramikkondensator
IC1	1 X ; DIP40	Sockel
Master	1 X ; Atmega 32	Mikrocontroller
Switch	1 X ; 4 Schalter	
IC2	1 X ; DIP8	Sockel
LED1/LED2	2 X ; rot 5mm	Leuchtdiode
T1	1X ; BC547C	NPN-Transistor
D1/D2/D3	3 X ; 1N4148	Diode
ZD1	1 X ; 3,3V	Zenerdiode
JP1/JP2	2 X ; 2 Stifte	Jumper
BMA	1 X ; BMA180 Breakout	Beschleunigungssensor

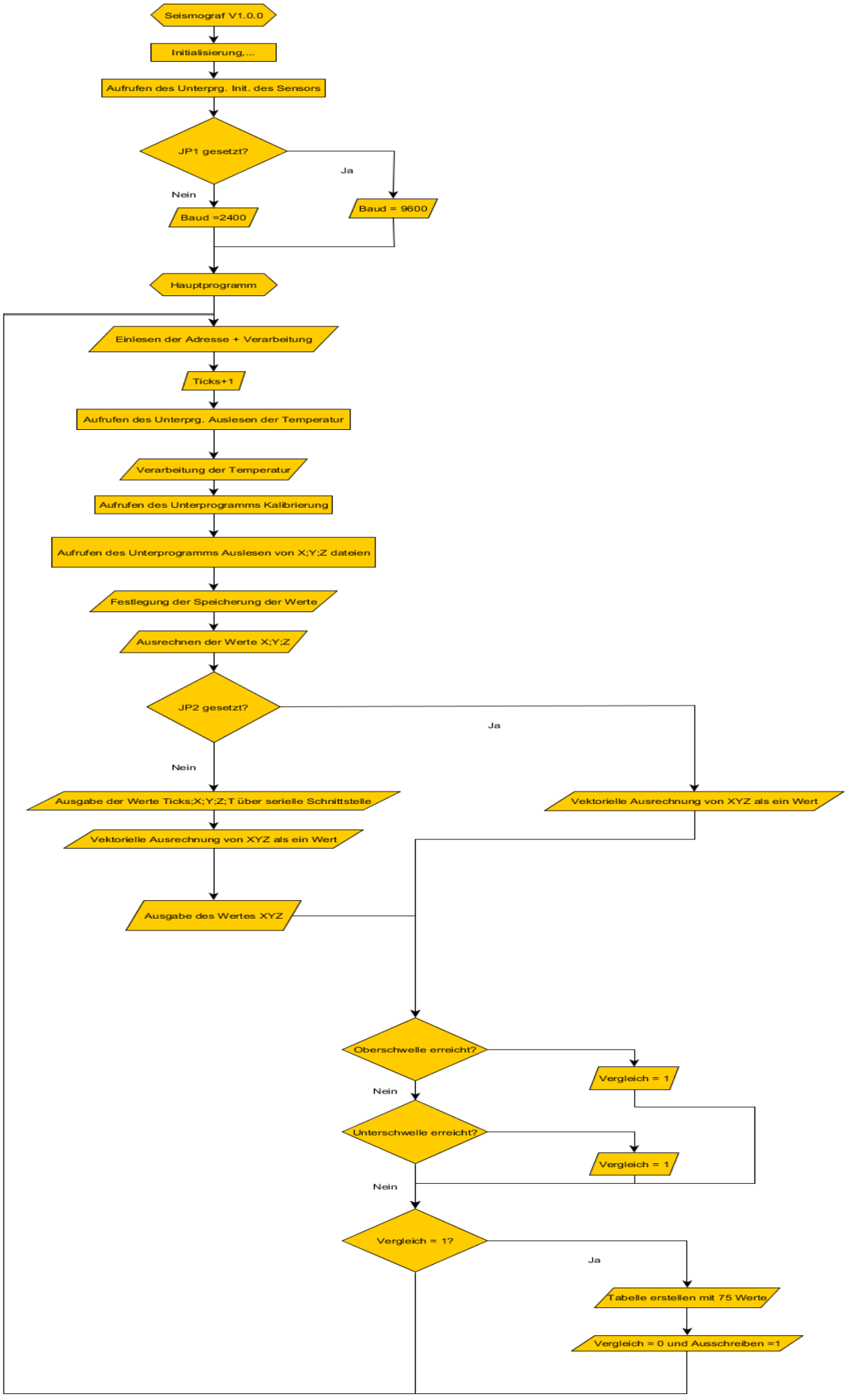
SD	1 X ; Halter	SD-Kartehalter
M3	9 X ; Schraube	
	1 X ; 16 Mhz	Quarz
	1 X ; 160X100	Platine

5)Software (Firmware)

Die Software ist kommentiert und in verschiedene Unterteilungen unterteilt. Die grünen Bemerkungen auf den Bildern sind Kommentare. Für dieses Programm zu schreiben, brauchte ich einige Zeit dafür. Ich fing damit an, erstmals die Version des Sensors herauszulesen, danach schrieb ich das Programm, um den Sensor zu initialisieren. Daraufhin kam das Auslesen der Daten. Nach dem Datenaustausch der Interruptroutine, programmierte ich den Mikrocontroller so, dass er 75 Messwerte aufnehmen kann. Diese 75 Messwerte werden erst gespeichert wenn die Werte über oder unter der Schwelle liegen. Ohne Teilaufgaben des Programms wäre es mir nicht möglich gewesen dies alles zu tun, wie es jetzt ist.

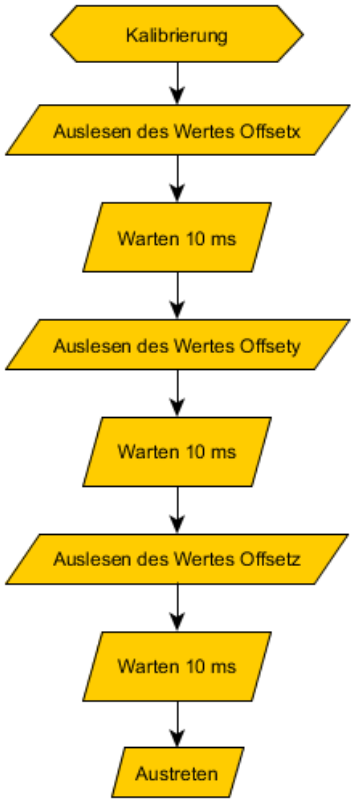
a) Flussdiagramme

Das erste Flussdiagramm ist die Darstellung des Hauptprogramms. Das erste Baustein stellt den Namen des Programms dar. Daraufhin kommt die Initialisierung. Danach wird der Jumper 1 abgefragt und der Interrupt wird aktiviert. Danach erfolgt das Hauptprogramm. Die einzelne Schritte werden dargestellt. Die Flussdiagramme der Unterprogramme und der Interruptroutine erfolgen nach dem ersten Flussdiagramm.

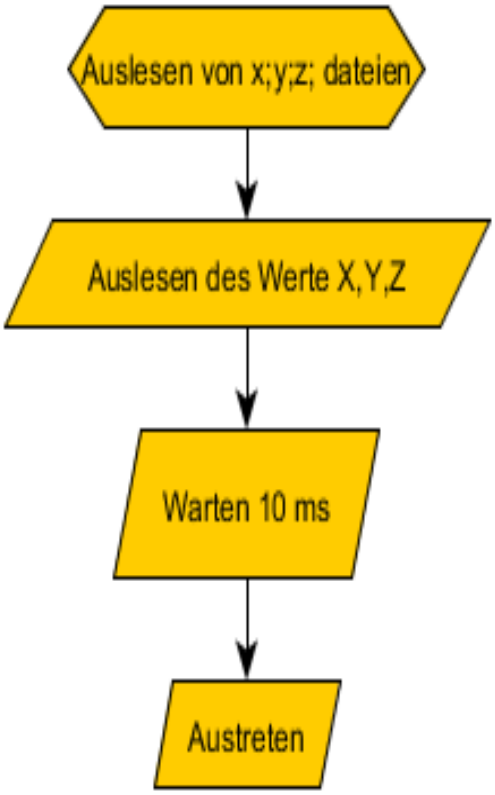

Seismograf V1.0.0
Initialisierung,...
Aufrufen des Unterprg. Init. des Sensors
JP1 gesetzt?
Ja
Nein
Baud =2400
Baud = 9600
Hauptprogramm
Einlesen der Adresse + Verarbeitung
Ticks+1
Aufrufen des Unterprg. Auslesen der Temperatur
Verarbeitung der Temperatur
Aufrufen des Unterprogramms Kalibrierung
Aufrufen des Unterprogramms Auslesen von X;Y;Z dateien
Festlegung der Speicherung der Werte
Ausrechnen der Werte X;Y;Z
JP2 gesetzt?
Ja
Nein
Ausgabe der Werte Ticks;X;Y;Z;T über serielle Schnittstelle
Vektorielle Ausrechnung von XYZ als ein Wert
Vektorielle Ausrechnung von XYZ als ein Wert
Ausgabe des Wertes XYZ
Oberschwelle erreicht?
Vergleich = 1
Nein
Unterschwelle erreicht?
Vergleich = 1
Nein
Vergleich = 1?
Ja
Tabelle erstellen mit 75 Werte
Vergleich = 0 und Ausschreiben =1

In dem ersten Unterprogramm wird der Sensor initialisiert. Die Chipid und die Version werden zuerst ausgelesen, danach wird der Sensor quittiert. Erst dann wird die Range und die Bandbreite bestimmt. Mehrere Details sind im unteren Flussdiagramm zu sehen.

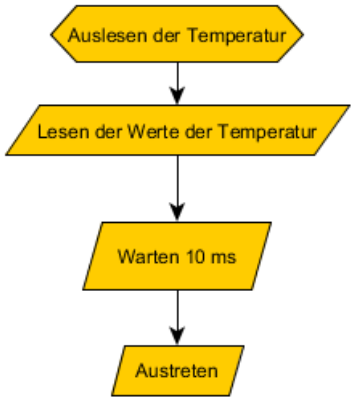
In dem nächsten Unterprogramm, werden die Werte für die Kalibrierung ausgelesen. Das Flussdiagramm ist das nächste.



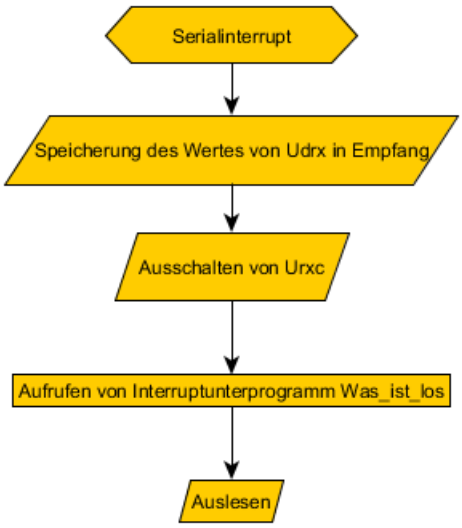
In dem nächsten Unterprogramm werden die Werte XYZ ausgelesen.



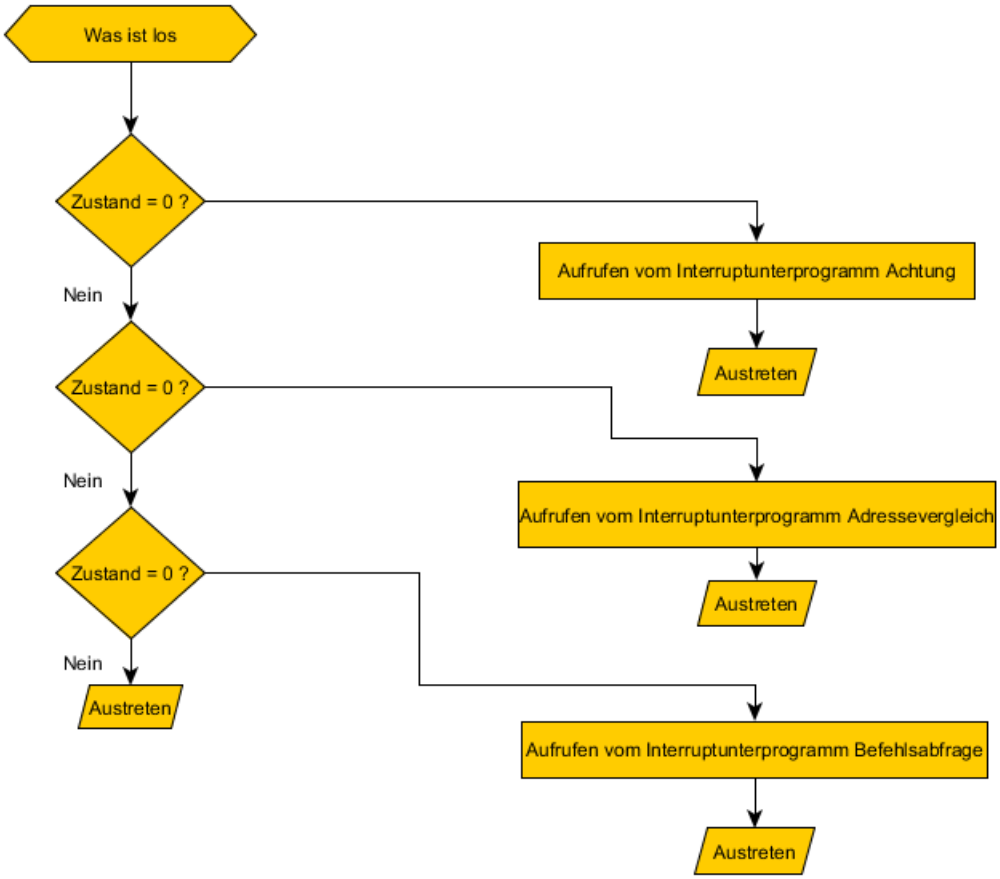
In diesem Unterprogramm wird de Temperatur herausgelesen und in Variablen gespeichert.



In den nächsten Flussdiagramme wird der Verlauf der Interruptroutine dargestellt. Dies erleichtert das Verständnis des Prinzip als purer Programmcode.



Das nächste Flussdiagramm ist der erste Unterprogramm in der Interruptroutine.



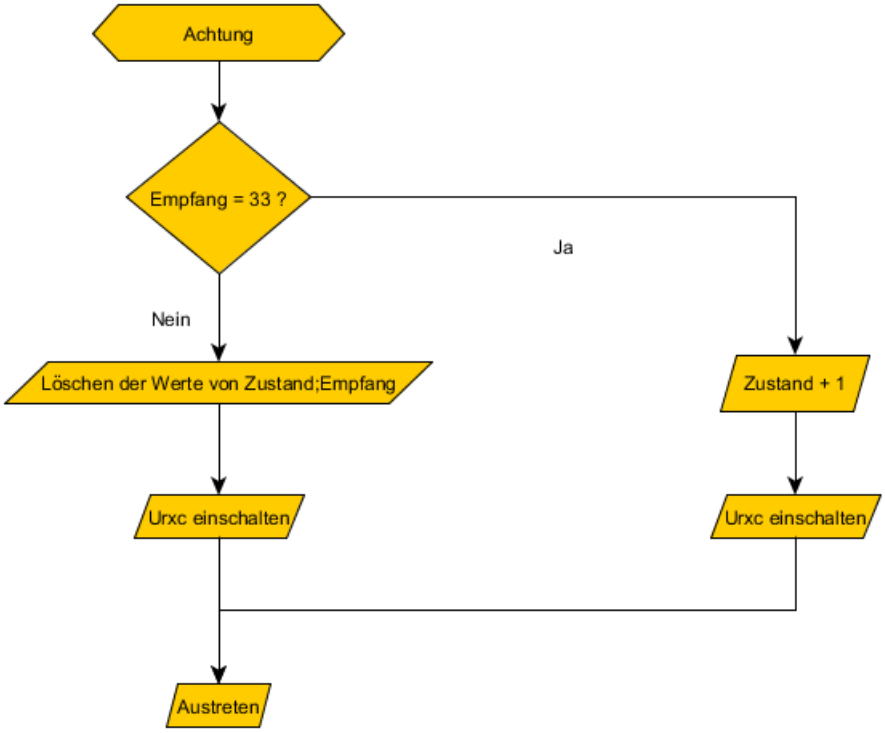
Die drei folgende Flussdiagramme erklären die Routine, wenn ein PC ein Modul anschreibt. Zuerst wird kontrolliert ob ein ! Ankommt, danach ob die Adresse stimmt. Zum Schluss wird geschaut welcher Befehl der PC geschickt hat und demnach wird dieser Befehl auch ausgeführt. Die Befehle lauten:

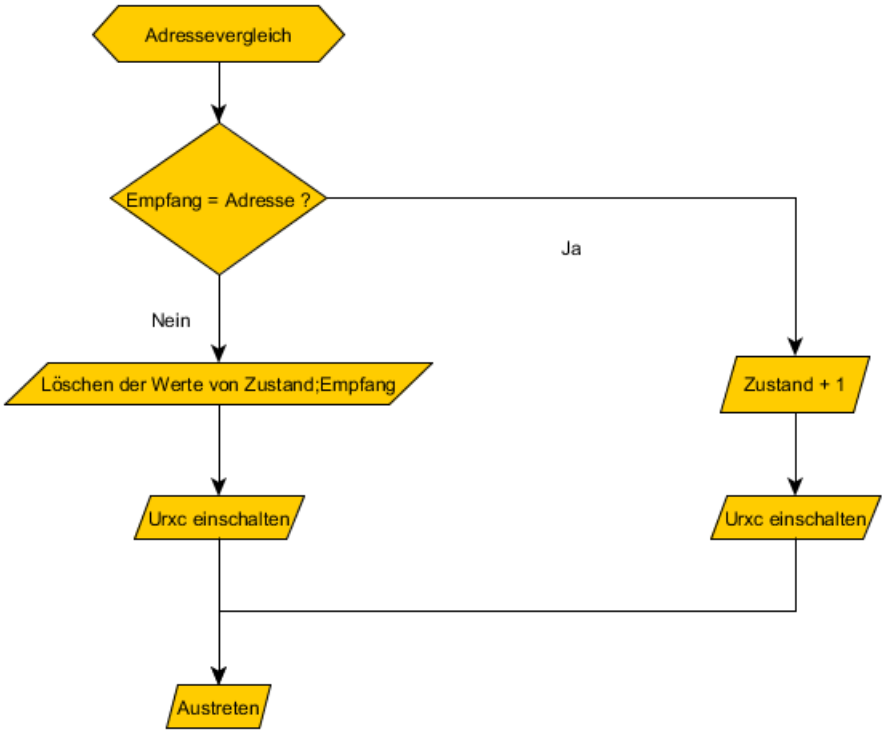
s für sende Daten

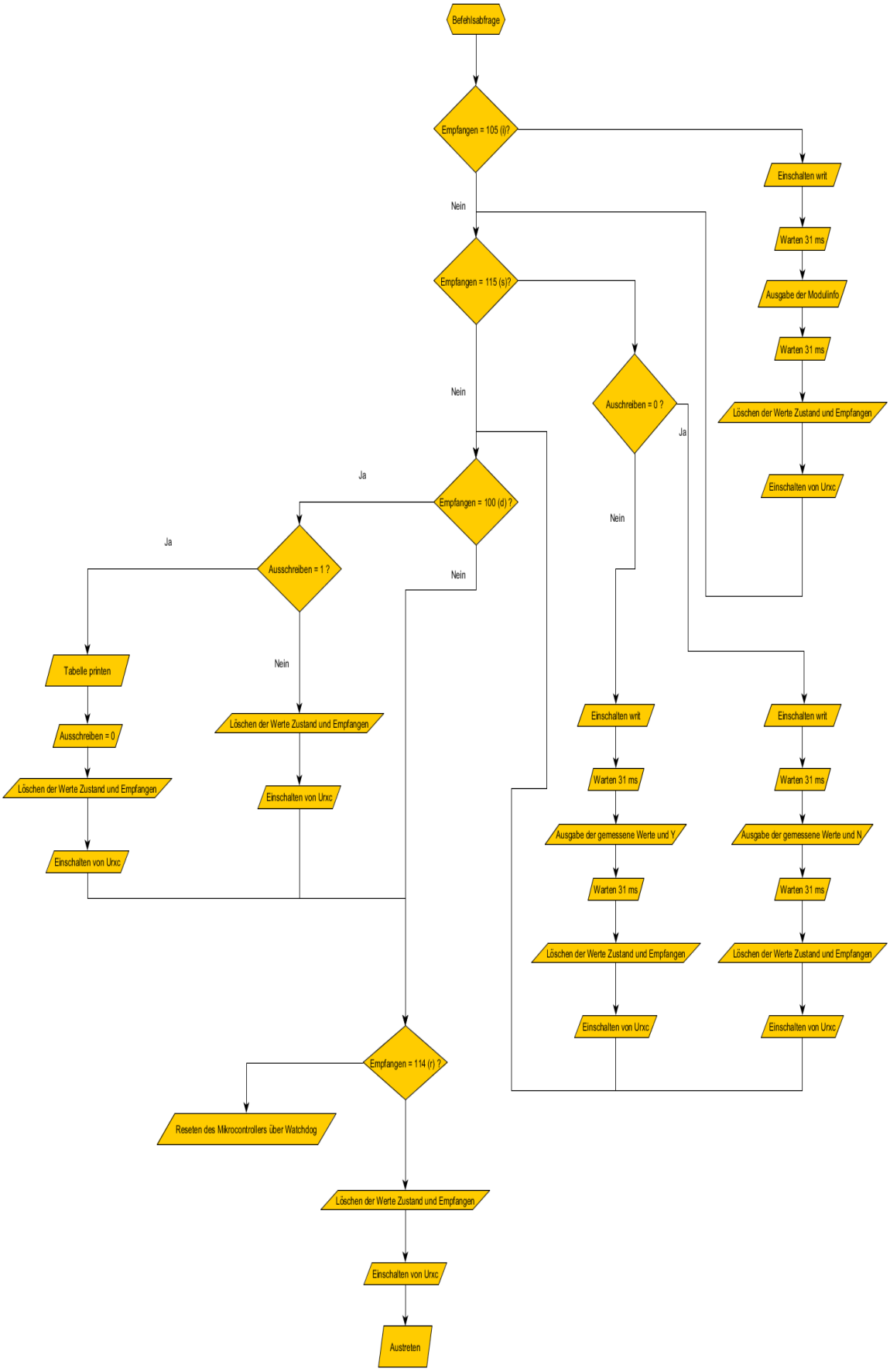
d für Tabellebfrage

i für Modulinfo und

r für quittieren des Mikrocontrollers


Achtung
Empfang = 33 ?
Ja
Nein
Löschen der Werte von Zustand;Empfang
Zustand + 1
Urxc einschalten
Urxc einschalten
Austreten


Adressevergleich
Empfang = Adresse ?
Ja
Nein
Löschen der Werte von Zustand;Empfang
Zustand + 1
Urxc einschalten
Urxc einschalten
Austreten


Befehlsabfrage
Empfangen = 105 (i)?
Einschalten writ
Warten 31 ms
Ausgabe der Modulinfo
Warten 31 ms
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Nein
Empfangen = 115 (s)?
Nein
Auschreiben = 0 ?
Ja
Nein
Empfangen = 100 (d) ?
Ja
Nein
Ausschreiben = 1 ?
Ja
Nein
Tabelle printen
Ausschreiben = 0
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Einschalten writ
Warten 31 ms
Ausgabe der gemessene Werte und Y
Warten 31 ms
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Einschalten writ
Warten 31 ms
Ausgabe der gemessene Werte und N
Warten 31 ms
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Empfangen = 114 (r) ?
Reseten des Mikrocontrollers über Watchdog
Löschen der Werte Zustand und Empfangen
Einschalten von Urxc
Austreten

b) Initialisierungen

```
'*********************************************************************************

                          'BMA 180 Beschleunigungssensor

'*********************************************************************************

'Vorstellung
'"Projet  T3EE Beschleunigungssensor"
'"Name:   Paquet Charel"
'"Klasse: T3EE"

$crystal = 16000000                                          ' Externer Quarz mit
16 MHz
$regfile = "m32def.dat"                                      ' ATmega 32
$lib "i2c_twi.lbx"                                           'Hardware I2c-Bus
$hwstack = 100                                               'reserviert 64 für
Hardwarestack
$swstack = 100                                               'reserviert 64 für
Swstack
$framesize = 100
                                          'reserviert 64 für Framespace

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

'Initialisierung der Ein- und Ausgänge
Config Porta.6 = Output                                      ' Port A als Ausgang
Config Portb = Input
Writ Alias Porta.6                                           'Write/Recieve
Config Porta.3 = Input                                       ' Switch von PORTA.0
bis 3
Config Porta.2 = Input
Config Porta.1 = Input
Config Porta.0 = Input
Porta = 15                                                   'Pull-down
Widerstände eingeschaltet
'---------------------------------------------------------------------------------

'Initialisierung des Watchdog zur überwachung des Programms
Config Watchdog = 2048

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

'Initialisierung des I2C-Bus
Config Sda = Portc.1                                         ' SDA am Pin C.1
Config Scl = Portc.0                                         ' SCK am Pin C.0
Config Twi = 400000                                          ' Taktrate 400kHz
I2cinit                                                      'Initialisiert den
I2C-Bus

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

'Initialiserung der Slaveadresse sowie Lesen und Schreiben
Const Slave = &H80                                           'Ansprechadresse bzw
slaveadresse um BMA 180 anzusprechen
Const Schreiben = &H80                                       'Ansprechadresse um
zu Schreiben
Const Lesen = &H81                                           'Ansprechadresse um
zu lesen von BMA180
```

Als erstes wird der Mikrocontroller definiert, danach werden die verschiedenen Speicherbereichen reserviert, daraufhin werden die Eingänge und Ausgänge bestimmt. Danach wird der Watchdog auf 2 Sekunden gesetzt. Nach dem Watchdog wird der I²C-Bus initialisiert. SCL und SDA werden an ein Pin bestimmt. SDA ist die Leitung für die Daten und SCL steht für Clock. Dann werden 3 Konstanten für den I²C-Bus bestimmt. Als erstes die Slaveadresse, danach Schreiben und zum Schluss Lesen. Diese Konstanten werden benötigt um den Sensor anzusprechen.

c) Konstanten

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

'Variablendeklaration für Register und ihre Konstanten des Sensor

Const Resetreg = &H10                                   'Resetregister
Const Reset1 = &HB6                                     'Um den Sensor zu
reseten

Const Chip_idreg = &H00                                 'Register um die
Chip_Id zu lesen

Const Rangereg = &H35                                   'Register um den
Range auszuwählen

Const Bwreg = &H20                                      'Register um die
Bandbreite auszuwählen

Const Offset_x = &H38                                   'Kalibrierung der X-
Datei
Const Offset_y = &H39                                   'Kalibrierung der Y-
Datei
Const Offset_z = &H3A                                   'Kalibrierung der Z-
Datei
Const Offset_t = &H37                                   'Kalibrierung der
Temperatur-Datei

Const Powermodereg = &H0D                               'Register um den
Sensor einsatzbereit zu machen
Const Wakeup = &B00010000                               'Code damit der
Sensor erweckt

Const Temperaturreg = &H08                              'Temperaturregister
um die Temp zu lesen

Const Datareg = &H02                                    'Aus diesem Register
werden die Dateien X_Y_Z gelesen

Const Versireg = &H01                                   'Aus diesem Register
wird die Version gelesen

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

Hier werden die Register des Sensors definiert. Diese Register sind notwendig ,denn

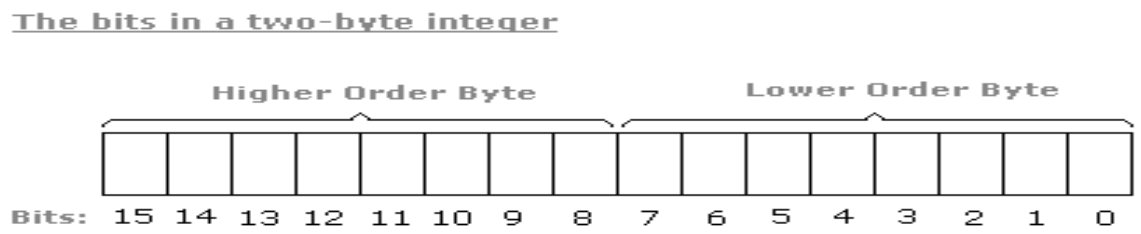
die Register haben eine wichtige Aufgabe. Mit dem Microcontroller, hat man den Zugriff auf diese Register des Sensors über I²C-Bus. In den Register sind auch die Messwerte gespeichert. Durch bestimmte Register hat man eine Vielfalt von Einstellungsmöglichkeiten des Sensors. Die Register sind auch im Datasheet zu finden. Bei den Register handelt es sich nicht nur um die Range und die Bandbreite, sondern auch zum Lesen der XYZ- Werte oder Temperatur, Version, Kalibrierung , und so weiter...

Verschiedene Register brauchen wir, das wäre Reset; Range; Bandbreite; Temperatur; Powermode; Kalibrierung von XYZ und noch Data.

d) Variablen

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

'Tabelle
Dim I As Byte                                           'Tabellevariablen
Dim A(75) As Single

'Definitionen für Speicherung z.B X;Y;Z;Temperatur-Dateien des Sensors
'X;Y;Z; Dateien
Dim X As Integer                                        'X ist als Integer
definiert, Integer besteht aus zwei Byte aus X high und X low
Dim Xl1 As Byte At X + 0 Overlay                        'X low wird in den X
hineingesetzt auf Platz 0
Dim Xh1 As Byte At X + 1 Overlay                        'X high wird in den
X hineingesetzt auf Platz 1

Dim Y As Integer                                        'Y ist als Integer
definiert
Dim Yl1 As Byte At Y + 0 Overlay                        'Y low wird in den Y
hineingesetzt auf Platz 0
Dim Yh1 As Byte At Y + 1 Overlay                        'Y high wird in den
Y hineingesetzt auf Platz 1

Dim Z As Integer                                        'Z ist als Integer
definiert
Dim Zl1 As Byte At Z + 0 Overlay                        'Z low wird in den Z
hineingesetzt auf Platz 0
Dim Zh1 As Byte At Z + 1 Overlay                        'Z high wird in den
Z hineingestzt auf Platz 1

'Kalibrierung Speicherung
Dim Offx As Integer                                     ' ist ein Integer
für Kalibrierung, X, Y, Z
Dim Offsetx As Byte At Offx + 0 Overlay

Dim Offy As Integer
Dim Offsety As Byte At Offy + 0 Overlay

Dim Offz As Integer
Dim Offsetz As Byte At Offz + 0 Overlay
```

Zuerst wird eine Tabelle mit 75 Werte deklariert. Danach wird definiert wie die Beschleunigungswerte gespeichert werden:



Auf diesem Bild kann man sich vorstellen wie die Werte geseichert werden. Der X-Wert besteht jetzt aus Xlow und Xhigh. So Xlow wird jetzt in dem ersten Byte gespeichert und Xhigh wird im zweiten Byte gespeichert. Aus diesen zwei Bytes entsteht jetzt der Integer. Wir haben jetzt ein Wert, nämlich X.

```
'Speicherung der X_Y_Z Dateien zum späteren Berechnungen
Dim Xs As Single                                        'Xs steht für X-
Datei als Single
Dim Ys As Single                                        'Ys steht für Y-
Datei als Single
Dim Zs As Single                                        'Zs steht für Z-
Datei als Single

'Temperaturdateispeicherung
Dim Temperaturd As Byte                                 'Temperaturd als
Dezimahl
Dim Temperatur As Byte                                  'Temperatureinlesung

'Auslesen Speicherung
Dim Wir As Byte                                         'was ist range
Dim Wibw As Byte                                        'was ist bandbreite

'Sollwert Speicherung
Dim Sr As Byte                                          'Sollrange
Dim Sbw As Byte                                         'Sollbandbreite

'Printdefinitonen am Pc
Dim Id As Byte                                          'ID des Sensors
Dim Versi1 As Byte                                      'Version des Sensors

'Rechnung
Dim Xys As Single
Dim Xyzs As Single                                      'Variable zum
speichern des Wertes XYZ

'Adresse
Dim Adresse As Byte

'Schwelle
Dim Vergleich As Byte                                   'Bestimmung der
Ober- und Unterschwelle
Const Xsu = 0.9
Const Xso = 1.2
Dim Ausschreiben As Byte                                'Ausschreiben wenn
über oder unter der Schwelle ein Wert aufgenommen wird.


'--------------------------------------------------------------------------
Start Watchdog
```

```
'Beginn der Initialisierung des Sensor
Gosub Initialisierung_des_sensors

'Jumper1 Auswahl Des Bauds

Portb.1 = 1
Portb.2 = 1

      If Pinb.1 = 1 Then                                    'Jumper 1 für
Baudrate
         Baud = 9600
      Else
         Baud = 2400
      End If

'-------------------------------------------------------------------------
'Interrupt

Dim Ticks As Single
Dim Empfangen As Byte
Dim Zustand As Byte

On Urxc Serialinterrupt
Enable Urxc
Enable Interrupts

Reset Watchdog
```

Bei den darauffolgende Variablen ist nicht spezielles dabei, bei jeder bestimmte Variabel ist ein Kommentar, der aussagt für was diese Variabel gebraucht wird. Die Oberschwelle und Unterchwelle hab ich bewusst so gewählt, denn ich hab über 5 Minuen die Werte aufgenommenwar immer zwischen 1 und 1,1 g also entschied ich mich, für einmal – 0,1 und + 0,1 und bin so auf die Werte von 0,9 und 1,2 gekommen. Dies hat halt den Vorteil, wenn mal was passieren würde, würde der Seismograf auch schwache Beschleunigung mitkriegen. Danach wird ein Unterprogramm aufgerufen das den Sensor initialisiert. Danach wird geschaut ob der Jumper 1 gestzt ist oder nicht. Demnach wird dann eine bestimmte Baudrate eingestellt. Danach wird der serielle Interrupt initialisiert. Wenn jetzt ein Zeichen empfangen wird, wird das Unterprogramm Serialinterrupt aufgerufen, dann geht der Mikrocontroller zum Serialinterrupt und macht dort seine Arbeit. Danach kehrt er wieder zurück wo er vorher war.

e) Hauptprogramm

Im Hauptprogramm wir zuerst die Adresse eingelesen und gespeichert. Danach wird der Ticks um 1 addiert. Dann wird der Unterprogramm Auslesen der Temperatur aufgerufen. In diesem Unterprogramm wird die Temperatur gemessen. Danach wird die Temperatur ausgerechnet. Danach wird die XYZ-Kalibrierung eingelesen. Als nächstes wird geschaut ob Jumer 2 gesetzt ist oder nicht. Wenn Jumper 2 gesetzt ist, werden die XYZ-Werte eingelesen. Diese werden zwei Stellen nach rechts verschoben und behalten ihr Vorzeichnen. Danach werden XYZ-Werte verwertet.

Dazu werden zum Beispiel die X-Werte mit der Kalibrierug addiert und als Single gespeichert. Danach wird X durch 8192 geteilt. Jenachdem die Range ist, muss man unterschiedlich teilen. Danach werden die Werte rausgeprintet über die serielle Schnittstelle.

```
'********************************************************************************
'*                                                                              *
'*                                  'Hauptprogramm                              *
'*                                                                              *
'********************************************************************************

Do

         Adresse = Pina                                        'Einlesen der
Adresse
            Toggle Adresse
         Adresse = Adresse And &H0F
         Adresse = 64 + Adresse

            Start Watchdog
         Ticks = Ticks + 1                                     'Ticks wird + 1
addiert

            Gosub Auslesen_der_temperatur

         Temperaturd = Temperatur / 2                          'Die Temperatur wird
berrechnet
         Temperaturd = Temperatur + 24

            Reset Watchdog
'-------------------------------------------------------------------------------

            Gosub Kalibrierung

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



         If Pinb.2 = 0 Then                                    'Jumper 2 wenn 0,
dann jede Sekunde was schicken sonst nicht

                        Gosub Auslesen_von_x_y_z_dateien

                     'der Wert X wird zwei Stellen nach rechts verschoben und
Vorzeichen wird nicht verändert
                        Shift X , Right , 2 , Signed
                        Shift Y , Right , 2 , Signed
                        Shift Z , Right , 2 , Signed

                     'Rechnungen um X;Y;Z zurechnen
                     X = X + Offx                              'X = X +
Kalibrierung von X
                     Xs = X                                    'X wird als Single
übernommen
                     Xs = Xs / 8192                            'X wird durch 8192
geteilt

                     Y = Y + Offy
                     Ys = Y
                     Ys = Ys / 8192
```

```
                    Z = Z + Offz
                    Zs = Z
                    Zs = Zs / 8192

                    'Ergebnis
                       Print Xs ; Chr(9) ; Ys ; Chr(9) ; Zs ; Chr(9) ;
Temperaturd
```

Nachdem die XYZ-Werte ausgerechnet sind, werden sie als Betrag des Vektors gespeichert. Aus den drei Werte ist jetzt nur noch ein Wert. Dieser Wert wird dann auch geprint. Wenn Jumper 2 jetzt offen ist, werden die XYZ-Werte ausgelesen und verarbeitet. Gleiches Prinzip wie vorher, es wird nur nichts über die serielle Schnittstelle geschickt.

Vektorieller Betrag:

$$XY = \sqrt{X^2 + Y^2}$$

$$XYZ = \sqrt{XY^2 + Z^2}$$


```
                  Xs = Xs ^ 2
                      Ys = Ys ^ 2
                      Zs = Zs ^ 2

                      Xys = Xs + Ys
                      Xys = Sqr(xys)
                      Xys = Xys ^ 2
                      Xyzs = Xys + Zs
                      Xyzs = Sqr(xyzs)

                         Print "
" ; Xyzs

                         Wait 1

            End If


        Reset Watchdog

                         Gosub Auslesen_von_x_y_z_dateien

               'der Wert X wird zwei Stellen nach rechts verschoben und
Vorzeichen wird nicht verändert
                         Shift X , Right , 2 , Signed
                         Shift Y , Right , 2 , Signed
                         Shift Z , Right , 2 , Signed

               'Rechnungen um X;Y;Z zurechnen
```

```
                X = X + Offx                                    'X = X +
Kalibrierung von X
                Xs = X                                          'X wird als Single
übernommen
                Xs = Xs / 8192                                  'X wird durch 8192
geteilt

                Y = Y + Offy
                Ys = Y
                Ys = Ys / 8192

                Z = Z + Offz
                Zs = Z
                Zs = Zs / 8192

                Xs = Xs ^ 2                                     'Vektorielle
Rechnung als ein Wert
                Ys = Ys ^ 2
                Zs = Zs ^ 2

                Xys = Xs + Ys
                Xys = Sqr(xys)
                Xys = Xys ^ 2
                Xyzs = Xys + Zs
                Xyzs = Sqr(xyzs)

                        Reset Watchdog

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

Als nächstes wird die Oberschwelle kontrolliert. Wenn dies der Fall ist wird Vergleich 1. Danach wird auch die Unterschwelle kontrolliert. Wenn Vergleich 1 ist, werden 75 Werte aufgenommen, sonst nicht. Das Auslesen der Daten und Berechnung ist das gleiche wie vornehin. Wenn 75 Werte aufgenommen wordensind, wird Ausschreiben 1. Wenn dies der Fall ist wird der PC informiert dass Werte aufgenommen worden sind.

```
        If Xyzs > Xso Then                              'Vergleichen der beide
Werte um zu schauen ob der Wert Über die Obereschwelle ist wenn ja, dann
Vergleich = 1

                Vergleich = 1

              Elseif Xyzs < Xsu Then                    ' Kontrolle der
Unterschwelle

                Vergleich = 1

              End If

                        Reset Watchdog                  'Wenn der Wert XYZ
höher oder tiefer als die Ober- oder Unterschwelle war, dann sollen 75 Werte
aufgenommen werden und gespeichert werden

              If Vergleich = 1 Then

                For I = 1 To 75                         'Tabelle, Were
```

```
werden dort drin gespeichert

                                    Reset Watchdog

                                Gosub Kalibrierung          'Auslesen der Werte
X;Y;Z und Verarbeitung

                                  Gosub Auslesen_von_x_y_z_dateien

                                  Gosub Berechnung

                                     Waitms 10

                                A(i) = Xyzs

                        Next

                    Vergleich = 0
                    Ausschreiben = 1                          'Durch das
Ausschreiben, wird der Pc informiert das in der Tabelle was ist.

                End If

                     Reset Watchdog

Loop
```

f) Unterprogramme

Das war das Hauptprogramm jetzt kommen noch Unterprogramme. Hier sehen wir den Unterprogramm Initialisierung des Sensors. Zuerst wird die Chipid herausgelesen, danach die Version und dann wird der Sensor quittiert. Nach dem Quittieren wird der Sensor erweckt in dem wir am Powermoderegister ihn erwecken, indem wir die Konstante Wake up schicken.Dadurch wird der Sensor aktiv in wird in den Beriebsmodus geschaltet. Danach wird die Bandbreite ausgewählt, die Bandbreite beträgt 10 Hz. Wenn wir höher auswählen, könnten digitale Filter eingeschaltet werden. Zuerst lesen wir die Bandbreite heraus, danach maskieren wir sie und setzten demnach einige Bits auf eins mit einer Oder-Verknüpfung. Wir haben das gleiche Prinzip bei bestimmen des Ranges oder der Bandbreite.

```
'*************************************************************************
'*                                                                       *
'*                                  'Unterprogramme                      *
'*                                                                       *
'*************************************************************************
'*                  Erstes Unterprogramm Initialisierung des Sensors     *
'*************************************************************************
Initialisierung_des_sensors:

         'Chipid auslesen
            I2cstart
               I2cwbyte Schreiben
               I2cwbyte Chip_idreg
```

```
            I2cstart
               I2cwbyte Lesen
               I2crbyte Id , Nack
            I2cstop

            Waitms 10

         'Version auslesen
            I2cstart
               I2cwbyte Slave
               I2cwbyte Versireg
            I2cstart
               I2cwbyte Lesen
               I2crbyte Versi1 , Nack
            I2cstop

            Waitms 10

         'Reseten
            I2cstart
               I2cwbyte Slave
               I2cwbyte Resetreg
               I2cwbyte Reset1
            I2cstop

            Waitms 10

         'Erwecken
            I2cstart
               I2cwbyte Slave
               I2cwbyte Powermodereg
               I2cwbyte Wakeup
            I2cstop

            Waitms 10

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

   'Bestimmen der Bandbreite über Maskieren und zurück schreiben

            I2cstart
               I2cwbyte Slave
               I2cwbyte Bwreg
            I2cstart
               I2cwbyte Lesen
               I2crbyte Wibw , Nack                         'was ist bw
            I2cstop

            Sbw = Wibw And &H0F
            Sbw = Sbw Or &B0000

            Waitms 10

            I2cstart
               I2cwbyte Slave
               I2cwbyte Bwreg
               I2cwbyte Sbw
            I2cstop

            Waitms 10

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

    'Bestimmen des Ranges über Maskieren und zurück schreiben
```

```
            I2cstart
               I2cwbyte Slave
               I2cwbyte Rangereg
            I2cstart
               I2cwbyte Lesen
               I2crbyte Wir , Nack                          'was ist range
            I2cstop

            Sr = Wir And &HF1                               'Sollrange
            Sr = Sr Or &H00

            Waitms 10

            I2cstart
               I2cwbyte Slave
               I2cwbyte Rangereg
               I2cwbyte Sr
            I2cstop

            Waitms 10

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

      Return
```

Hier sehen wir wie die Range ausgewählt wird. Danach kommen wir zum nächsten Unterprogramm. In diesem lesen wir jetzt die Temperatur heraus. Das Prinzip beim herauslesen ist überall gleich.

I²C-Bus Start

Slaveadresse

Adresse des Registers das wir schicken wollen

Repeated Start

Lesen

Wert speichern

I²C-Bus Stop

Im nächsten Unterprogramm wird die Temperatur herausgelesen. Dies erfolgt nach dem gleichen Prinzip. In dem nächsten Unterprogramm werden die Werte der Kalibrierug ausgelesen und gespeichert. H39 oder 39 oder 3A sind die Register wo diese erte im Sensor gespeichert sind.

```
'*******************************************************************************
'*                      Zweites Unterprogramm Temperaturauslesung              *
'*******************************************************************************

'Temperatur wird herausgelesen

Auslesen_der_temperatur:
```

```
            I2cstart
               I2cwbyte Slave
               I2cwbyte Temperaturreg
            I2cstart
               I2cwbyte Lesen
               I2crbyte Temperatur , Nack
            I2cstop

            Waitms 10
'-------------------------------------------------------------------------------

      Return

'*******************************************************************************
'*                      Drittes Unterprogramm Kalibrierung von X;Y;Z            *
'*******************************************************************************

'Kalibrierung, die manuelle Kalibrierung erfolgt durch herauslesen der
Offsetdateien

Kalibrierung:

'Herauslesen der Offsetx
            I2cstart
               I2cwbyte Slave
               I2cwbyte &H38
            I2cstart
               I2cwbyte Lesen
               I2crbyte Offsetx , Nack
            I2cstop

            Waitms 10

'Herauslesen der Offsety

            I2cstart
               I2cwbyte Slave
               I2cwbyte &H39
            I2cstart
               I2cwbyte Lesen
               I2crbyte Offsety , Nack
            I2cstop

            Waitms 10

'Herauslesen der Offsetz

            I2cstart
               I2cwbyte Slave
               I2cwbyte &H3A
            I2cstart
               I2cwbyte Lesen
               I2crbyte Offsetz , Nack
            I2cstop

            Waitms 10

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

      Return
```

```
'*****************************************************************************
'*                      Viertes Unterprogramm X_Y_Z Auslesung                *
'*****************************************************************************

'Auslesen der Dateien X_Y_Z
Auslesen_von_x_y_z_dateien:

            I2cstart
               I2cwbyte Slave
               I2cwbyte Datareg
            I2cstart
               I2cwbyte Lesen
               I2crbyte Xl1 , Ack
               I2crbyte Xh1 , Ack
               I2crbyte Yl1 , Ack
               I2crbyte Yh1 , Ack
               I2crbyte Zl1 , Ack
               I2crbyte Zh1 , Nack
            I2cstop

            Waitms 10

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

      Return
'*****************************************************************************
'*                      Viertes Unterprogramm Berechnung                     *
'*****************************************************************************

Berechnung:

 'der Wert X wird zwei Stellen nach rechts verschoben und Vorzeichen wird nicht
verändert
                  Shift X , Right , 2 , Signed
                  Shift Y , Right , 2 , Signed
                  Shift Z , Right , 2 , Signed

                  'Rechnungen um X;Y;Z zurechnen
                      X = X + Offx                          'X = X +
Kalibrierung von X
                      Xs = X                                'X wird als Single
übernommen
                      Xs = Xs / 8192                        'X wird durch 8192
geteilt

                      Y = Y + Offy
                      Ys = Y
                      Ys = Ys / 8192

                      Z = Z + Offz
                      Zs = Z
                      Zs = Zs / 8192

                      Xs = Xs ^ 2
                      Ys = Ys ^ 2
                      Zs = Zs ^ 2

                      Xys = Xs + Ys
                      Xys = Sqr(xys)
                      Xys = Xys ^ 2
                      Xyzs = Xys + Zs
                      Xyzs = Sqr(xyzs)
```

```
   Return
```

Hier ist das nächste Unterprogramm, in diesem werden X_Y_Z herausgelesen. Dies passiert nur auf einem Register. Der Dataregister, seine Registernummer steht oben bei den Variablen. Jeder Wert besteht aus einem Low und einem High-Byte, diese werden nach dem Lesen automatisch in ein Integer gespeichert wegen der Overlay-Definition der Variabel. Danach kommt noch ein Unterprogramm, das XYZ-Werte ausrechnet, gleiches Prini wie vorher.

g) Interrupt-Service-Routine:

```
'*********************************************************************************
'*                                       Interrupt                                *
'*********************************************************************************

'Interface
Serialinterrupt:

      Stop Watchdog
      Empfangen = Udr
      Disable Urxc

      Gosub Was_ist_los

      Start Watchdog

   Return

'*********************************************************************************
'*                     Interrupt Unterprogramm Aufgerufen                        *
'*********************************************************************************

Was_ist_los:
                                          'Zuerst muss ein!, dann eine Adresse
und dann ein Befehl geschickt werden.
                                          'Hier werden die einzelne Schritte
überwacht.

            If Zustand = 0 Then

                  Gosub Achtung

            Elseif Zustand = 1 Then

                  Gosub Adressevergleich

            Elseif Zustand = 2 Then

                  Gosub Befehlsabfrage
```

```
            End If

   Return
```

Wenn der PC etwas zum Modul schickt, erfasst es der serielle Interrupt. In diesem Fall springt der Mikrocontroller ins Unterprogramm des Interrupts. Wenn er dort alles getan hat, macht er mit seiner vorherrigen Arbeit weiter. Zuerst wird das Empfangene gespeichert. Danach wird Urxc ausgeschaltet um Störungen zu vermeiden nd Watchdog wird während ds Interrupts ausgeschaltet. Danach springt das Programm in das Unterprogramm Was ist los. Dort wird analysiert welches Synchronisierbyte es ist. Zuerst kommt !, danach die Adresse und zum Schluss ein Befehl. Bei dem Unterprogramm des Interrupts Achtung, wird nur kontrolliert ob das Empfangene ein ! Ist, wenn ja wird der Zustand um 1 addiert, wenn nicht wird alles zurückgesetzt. Im Unterprogramm Adressevergleich wird die Moduladresse mit der empfangene Adresse verglichen. Wenn dies der Fall ist, wird Zustand um 1 addiert. Wenn nicht wird wieder alles zurück gesetzt.

```
'*******************************************************************************
'*                      Interrupt Unterprogramm Achtung                        *
'*******************************************************************************

Achtung:
                                                                  'Kontrolle
des !
            If Empfangen = 33 Then                          '!
               Zustand = Zustand + 1
               Enable Urxc

            Else
               Zustand = 0
               Empfangen = 0
               Enable Urxc
            End If

   Return

'*******************************************************************************
'*                    Interrupt Unterprogramm Adressevergleich                 *
'*******************************************************************************
Adressevergleich:

             If Empfangen = Adresse Then                    'Kotrolle ob die
Adresse mit dem der Adresse des Mikrocontrollers übereinstimmt
               Zustand = Zustand + 1
               Enable Urxc

            Else
               Zustand = 0
               Empfangen = 0
               Enable Urxc
            End If

   Return

'*******************************************************************************
'*                    Interrupt Unterprogramm Befehlsabfrage                   *
'*******************************************************************************
```

```
Befehlsabfrage:

   If Empfangen = 105 Then                                    'Wenn i vom Pc
geschckt werden, schickt der Microcontrolle die Modulinfo
      'schicke Modulinfo

            Writ = 1

               Waitms 31

                  Print "# Modul Seismograf"
               Print "# PAQUET Charel"
                  Print "# Befehle:"
                  Print "# ! " ; Chr(adresse) ; " s " ; " Daten senden"
                  Print "# ! " ; Chr(adresse) ; " d " ; " Daten der Tabelle
senden"
                  Print "# ! " ; Chr(adresse) ; " i " ; " Sende Modulinfo"
                  Print "# ! " ; Chr(adresse) ; " r " ; " Resetten des Moduls"
                  Print "# Firmware version:" ; Chr(9) ; Version() ; Chr(9) ;
"Version:" ; "V 1.0"
                  Print "# Wenn das Modul nichts besonderes misst dann kommt der
Buchstabe N dort stehen,"
                  Print "# die folgende Wert ist dann ein durchschnitts Wert"
                  Print "# Ticks" ; Chr(9) ; "No or Yes" ; Chr(9) ; "XYZ" ;
Chr(9) ; "X" ; Chr(9) ; "Y" ; Chr(9) ; "Z" ; Chr(9) ; "_EOD_"
                  Print "# Nummer von der tabelle " ; Chr(9) ; "Xyzs" ; Chr(9) ;
"EOD"
                  Print "# Abtastung 50 ms pro Wert der Tabelle"

               Waitms 31

            Writ = 0
            Zustand = 0
            Empfangen = 0

               Enable Urxc

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

           'schicke Daten

   Elseif Empfangen = 115 Then                                'Wenn der PC s
schickt, wird geschaut ob Ausschreiben 1 ist, wenn ja schickt der
Mikrocontroller die Daten und den Buchstaben Y, um das der PC weiss, dass in der
Tabelle Werte sind.
                                                               'Wenn Ausschreiben
0 ist, werden die Werte geschickt und den Buchstabe N, damit der PC weiss, dass
keine Daten in der Tabelle sind
      If Ausschreiben = 0 Then

         'Schwelle nicht erreicht

            Writ = 1

               Waitms 31
                  Print Ticks ; Chr(9) ; "N" ; Chr(9) ; Xyzs ; Chr(9) ; Xs ;
Chr(9) ; Ys ; Chr(9) ; Zs ; Chr(9) ; "EOD"
               Waitms 31

            Writ = 0
            Ausschreiben = 0
            Empfangen = 0
            Zustand = 0
```

```
               Enable Urxc

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

      Else

         'Schwelle erreicht informiert durch Y

            Writ = 1

               Waitms 31
                  Print Ticks ; Chr(9) ; "Y" ; Chr(9) ; Xyzs ; Chr(9) ; Xs ;
Chr(9) ; Ys ; Chr(9) ; Zs ; Chr(9) ; "EOD"
               Waitms 31

            Writ = 0
            Zustand = 0
            Empfangen = 0

             Enable Urxc

      End If

'-------------------------------------------------------------------------------
       'Tabelle schicken mit d

    Elseif Empfangen = 100 Then                              'Wenn der PC den
Buchstaben d schickt, wird geschaut ob Ausschreiben 1 ist. Wenn ja schckt der
Mikrocontroller die Tabelle

      If Ausschreiben = 1 Then

            Writ = 1

               Waitms 31
                  Print Ticks

          For I = 1 To 75
                  Print I ; Chr(9) ; A(i)
          Next I

                  Print "EOD"
               Waitms 31

            Writ = 0
            Ausschreiben = 0
            Empfangen = 0
            Zustand = 0

               Enable Urxc



      Else

            Zustand = 0
            Empfangen = 0

               Enable Urxc

      End If
'-------------------------------------------------------------------------------

        'Reseten des Moduls
```

```
      Elseif Empfangen = 114 Then                          'Wenn der Pc r
schickt, wird der Watchdog ausgelöst und der Mikrocontroller wird quittiert

              Config Watchdog = 512
              Start Watchdog
            Wait 1
              Enable Urxc


      Else

          Zustand = 0
            Empfangen = 0

              Enable Urxc

      End If

Return

'*****************************************************************************
'*                                                                           *
'*                                    Ende                                   *
'*                                                                           *
'*****************************************************************************

End
```

Der letzte entscheidener Punkt ist die Befehsabfrage. Bei meinem Modul gibt es vier Befehle:

s für sende Daten

i für sende Modulinfo

r für reseten des Moduls

d für sende Tabelle

Wenn der Befehl r ankommt, dann wird der Watchdog ausgelöst und der Microcontroller ist quittiert.

Wenn der Befehl i ankommt, dann sendet der Microcontroller die Modulinfo, danach wird wieder alles zurück gesetzt.

Wenn der Befehl s ankommt, schaut der Mikrocontroller ob eine Tabelle vorhanden ist, wenn ja dann wird der Buchstabe Y mit geschickt. Damit weiss der PC dass eine Tabelle da ist. Wenn keine da is, wird der Buchstabe N geschickt.

Wenn der Befehl d ankommt, schaut der Microkontroller ob Ausschreiben 1 ist wenn ja , schick der Mikrocontroller die Tabelle.

Wenn keiner der Befehl übereinstimmt, wird alle zurückgesetzt.

6)Tagesberichte

29.9.14:

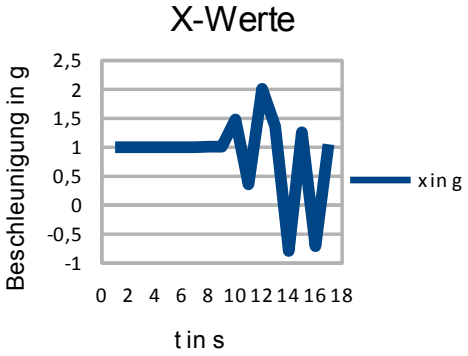
Am 29.9.14 bekamen wir vom Herr Feltes eine Einführung, bzw wie die Module ihre Fähigkeiten benutzen mit spezielle Anwendungen dazu (Anhang). Nachdem Herr Feltes fertig war, begann ich meine Informationssuche über den Sensor. Dies war nicht schwer, denn ich hatte einen schon ausgewählten Beschleunigungssensor, denn BMA 180 Breakout Beschleuinigungssensor. Ich hatte den Datasheet des BMA gefunden. Der Sensor ist von Siemens in SMD-Technik erhältlich. Der Sensor hat einige Rangeauswahlen und Bandbreiteauswahlen. Je nachdem wie und wo man den Sensor benutzt, muss man ihn richtig konfigurieren. Ich hab eine Rangeauswahl von 0-1g und meine Bandbreite beträgt 10Hz. Nebenbei des Datasheet hatte ich auch ein Programm gefunden der die Daten vom Sensor lesen und bearbeiten kann gefunden, aber dies war in Arduinosprache.

6.10.14:

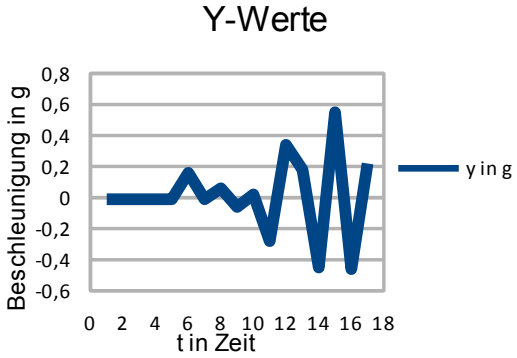
Am 6.10.14 wollte ich mein Sensor testen. Dazu nahm ich mir ein Arduino Uno und schloss mein Senor mit dem Arduino an. Danach schickte ich das gefundene Programm auf den Arduino Uno. Daraufhin testete ich meinen Sensor. Der Sensor erfasste Daten und schickte die Daten auf den Terminal in Arduino. Ich kopierte mir die Werte und erstellte in Excel drei Kennlinien..

Nachdem ich den Beweis hatte, das mein Sensor funktioniert hab ich weiter im Datasheet herumgeblettert um wichtige Infos herauszufinden, wie genau der Sensor funktioniert bzw wie meine Vorschritte sind das mein Sensor Daten erfassen kann. Nebenbei hab ich mir in der Bibliothek ein Buch über Bascom ausgeliehen um einige Befehle anzueignen.

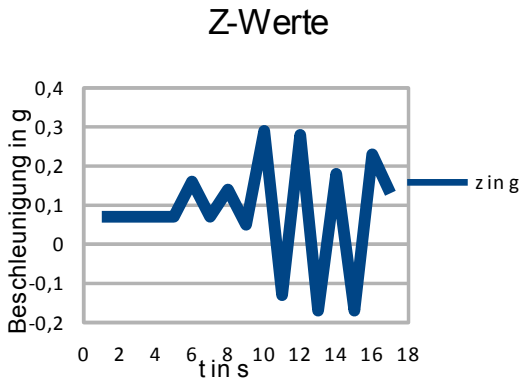
Kennlinien:xf(t) (1); yf(t) (2); zf(t) (3) jeweils in g



(1)



(2)



(3)

20.10.14:

Bei dem Beschleunigungssensor gibt es zwei Möglichkeiten um ihn anzusprechen: den SPI oder den I²C-Bus. Ich hab den I²C-Bus genommen weil ich ihm Buch etwas über den I²C-Bus gelesen habe und eine leichte Ahnung hatte wie er funktioniert. Damit mein Sensor arbeiten kann musste ich ihn mal erst initialisieren. Also fing ich an ein Programm zu schreiben das den Sensor initialisiert. Dabei war der I²C-Bus sehr wichtig. Deswegen überblickte ich nochmal das ganzen Datasheet um wichtige Infos herauszufinden wie Range ; Bandbreite ; Powermode.

3.11.14:

Am 3.11.14, schrieb ich ein Programm, das vom Sensor zuerst die CHIP_ID, danach die Version einliest und ihn dann quittiert. Dann bestimmte ich die Range und die Bandbreite. Nach der Bestimmung der Bandbreite, ist die Initialisierung des Sensors vollendet. Danach wendete ich mich der Auswertung der Daten zu. Dazu liest der Mikrocontoller die Daten heraus und wandelt sie von Binär nach Dezimal um, damit

man sie im Terminal lesen kann.

10.11.14:

Das geschriebene Programm der vorigen Woche kompilierte ich. Ich musste einige Konstanten verändern, sowie das Geteiltzeichen \ anstatt:, weil ich beim kompilieren die entsprechende Fehler angezeigt bekam. Danach begann ich mit der Dimensionierung des I²C-Bus mit zwei Pull-upwiderstände jeweils 1k80 Ohm. Dazu eine Kollektorschaltung die mir 3.3V liefert für den I²C-Bus. Ich gebe 5 V rein und bekomme 3.3V heraus. Warum 3.3V ? Also erstens ist das die Betriebsspannung des Sensors und der Schaltpegel des I²C-Bus, denn es ist höher als 2.5 V. Danach baute ich die Testschaltung auf einer Steckplatine auf.

17.11.14:

Ich nahm die Kollektorschaltung in Betrieb und testete sie. Am Ausgang waren 3.3 V. Also schloss ich alles am Mikrocontroller an und testete mein Programm. Ich bekam im Terminal kein Ergebnis, also habe ich mit einem Oszilloskop die Signale SDA und SCL kontrolliert. Am SDA sah man dass der Slave oder der Master dauernd schrieb.

24.11.14:

Ich untersuchte nochmals das Datasheet und fand einen Hinweis. Beim Lesen des Sensors muss man folgendermaßen vorgehen:

- Start
- Slave
- Register
- repeated Start
- Leseadresse
- Byte lesen und speichern
- Stop

Nach diesem Prinzip hat mein Sensor die Chiptemperatur herausgegeben. Nachdem ich dies herausgefunden hatte, verbesserte ich mein Programm.

1.12.14:

An diesem Tag schrieb ich den Programm um die XYZ-Werte herauszulesen, danach

testete ich das Programm. Es hat leider nicht funktioniert. Als Hilfestellung benutzte ich das Programm in Arduino und verglich es mit meinem Programm. Leider fand ich den Fehler nicht. Ausserdem hab ich die die Daten in ein Word geschrieben, das ich herauslese. Dazu hab ich auch das Programm geschrieben um die XYZ-Werte auszurechnen.

8.12.14:

Am 8. Dezember begann ich meine Fehlersuche erneut. Ich hab den Range und BW im Sensor einfach gesetzt bzw das Register vom Sensor hab ich nur so gesetzt das ich die Range und Bandwith bestimme. Dies war aber falsch. Im Arduino-Programm war die folgende Vorgehensweise:

Zuerst liest man das Register heraus. Danach maskiert man die nicht benutzten Bits und setzt die Bits zum bestimmen des Ranges oder der Bandbreite mit einer UND-Verknüpfung auf eins oder null. Nachdem ich dies analysiert hatte, ging ich diesem Prinzip nach. Nach dem ich dies dann getestet hatte, bekam ich die Werte der Beschleunigung heraus.Die Daten werden geschickt. Auf der X-Achse bekam ich 1 g heraus also 9,81 m/s^2. Die Werte stimmen. Wenn ich den Sensor gedreht hatte, bekam ich auf der Z und Y-Achse auch 1 g heraus.

15.12.14:

An diesem Tag, hab ich mein Programm kommentiert. Außerdem habe ich durch ein Vorgespräch mit Herrn Feltes beschlossen, dass ich meine Werte jetzt als vektorielle Werte herausgebe. Dies hat den Vorteil, dass ich viel Platz spare und es vereinfacht später meine Zusatzaufgabe. Ich wolle nämlich mehrere Werte aufnehmen, wenn in den Stollen der Mine was passiert. Nebenbei hab ich auch den Watchdog hinzugefügt und den Jumper 1.

5.1.15:

Am 5. Januar hab ich den Jumper 2 in mein Programm eingefügt. Ausserdem hab ich mich ein wenig über den Interrupt informiert, den brauche ich damit das Modul und das Interface kommunizieren können.

12.1.15:

Heute habe ich die serielle Schnittstelle in mein Programm hinzugefügt als Interrupt. Wenn ich jetzt im Terminal ein Befehl eingebe, reagiert mein Mikrocontroller und macht seine Arbeit. Ab dem 12.Januar war ich mit meiner Hauptaufgabe im Projekt

fertig. Ich fing jetzt an Zusatzaufgaben zu erledigen.

19.1.15:

Meine Zusatzaufgabe: Ich werde eine Ober- und Unterschwelle programmieren. Danach vergleiche ich ob XYZ höher oder tiefer als die Schwellen sind, wenn ja, dann nehme ich mehrere Werte auf. Dies kann der all sein, wenn in der Mine ein Zusammenfall eines Stollens passieren könnte. Also ich will zuerst die Werte in einer Tabelle speichern und wenn das funktioniert auf einer SD-Karte. Die Werte können durch den Buchstaben d abgefragt werden. Wenn der PC mein Modul abfragt, bekommt der PC 75 Werte über die serielle Schnittstelle geschickt. Aber dies ist nur möglich wenn ich genügend Zeit habe. Das positive an der SD-Karte wäre, ich könnte viele Werte aufnehmen.

3.02.15:

Am 23. Februar plante ich meine Platine. Ich ermittelte alle Bauteile und begann meine Schaltung vorauszuplanen. Ich misste die Rastermassen meiner Bauteile.

2.03.15:

An diesem Tag erstellte ich eine Bauteilliste mit ihren Rastermassen, sowie auch den Layout. Ich ätzte in dieser Woche auch meine Platine und bohrte die Löcher.

9.03.15:

Am 9. März bestückte ich meine Platine und danach testete ich. Aber ich bekam keine Werte.

16.03.15:

Am 16. März kontrollierte ich die Platine und stellte fest, dass am Sensor den Ground fehlte und SDA mit SCL vertauscht waren. Ich behob die Fehler und testete erneut. Diesmal bekam ich wieder Werte.

4.05.15:

An diesem Tag baute ich eine Testschaltung auf, um mit dem Microcontroller und einer SD-Karte zu arbeiten. Dazu benutzte ich eine PDF-Datei von Herr Feltes. Danach testete ich ein Programm um eine SD-Karte auszulesen. Dies funktionierte aber nicht, denn eine Datei aus der Bibliothek von Bascom funktionierte nicht.

11.05.15:

Ich bekam von Herr Feltes einige Bascomprogramme um mit einer SD-Karte zu arbeiteten. Aber irgendwie funktionierte nichts. An diesem Tag entschloss ich, dass ich die SD-Karte auf der Seite lasse und mich nach der Tabelle wendete. Die 75 Werte werden jetzt in der Tabelle gespeichert. Ich kommentierte auch wieder mein Programm.

18.05.15:

An diesem Tag verbesserte ich einige Sachen im Programm, wie das senden der Modulinfo,... .Jetzt bekommt der PC mehr Infos über mein Modul. Nebenbei bekommt der PC mit, ob Werte in der Tabelle sind oder nicht. Später testete ich meine Platine am Interface, aber es funktionierte nicht.

1.06.15:

An diesem letzten Tag fand ich mein Fehler. Ich hatte meine Pulldown-widerstände an den Eingänge des Mikrocontrollers vergessen für die beiden Jumpers und den Ausgang der Variabel Writ zu definieren. Außerdem wurde der Watchdog in der Interrupt-Route ausgelöst. Ich behob alle diese Fehler. Mein Modul funktioniert jetzt einwandfrei.

7)Eigenständigkeitserklärung:

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die den Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quellen kenntlich gemacht.

Paquet Charel

8)Quellen

Buch:

AVR/Bascom

Arduino-Programm von Olaf Meier mit Schaltung:

http://Arduino-Hannover.de

Informationssuche:

http://staff.ltam.lu/feljc/home.html

www.mikrocontroller.net

www.roboternetz.de

www.rn-wissen.de

www.bascom-forum.de

www.google.de mit ihren Suchergebnissen

Bilder:

http://www.google.de/imgres?imgurl=http%3A%2F%2Fwww.indiabix.com%2F_files%2Fimages%2Ftechnical%2Fc%2F23-1.png&imgrefurl=http%3A%2F%2Fwww.indiabix.com%2Ftechnical%2Fc%2Fbits-and-bytes%2F2&h=164&w=430&tbnid=jPcZR6YKxpH0KM%3A&zoom=1&docid=p6y_17CE8C5Z0M&ei=VqRvVYu_Kcf0ULDhgpgO&tbm=isch&iact=rc&uact=3&dur=2080&page=1&start=0&ndsp=8&ved=0CCAQrQMwAA

9)Anhang