A handheld hall probe based on an Arduino Micro

TECHNISCHE UNIVERSITÄT DARMSTADT

Application for *Arduino Uno WiFi XMAS Contest 2015*
by Henning Janssen

This is a short documentation about a mobile handheld hall probe, which is based on an Arduino Micro. A battery powered Arduino evaluates via the analog input a hall effect sensor. The measured value will be shown on a display.

With this device it is possible to measure magnetic fields, i.e. the magnetic flux density as a constant and as a time-depending value. It is portable and independent from a power grid.

1 Hardware

The total system consists of a handheld (includes the Arduino, energy supply and display) and a set of external sensors. A picture of the whole measuring system is presented in **Figure 1**.

Figure 1: The handheld hall probe measuring the magnetic flux density of a permanent magnet

1.1 Set of probes

The magnetic flux density can be measured with different sensitivities. Therefor different hall effect sensors are used. The probes are equipped with ratiometric linear hall effect sensor ICs from *Honeywell* and *Allegro*. They need a supply voltage of $V_{CC} = 5\,V$ and the output voltage depends linearly from the magnetic flux density. **Figure 2** shows the three different sensor in their housings and a rod to hold the device into the measuring area.

Figure 2: Three types of sensor chips with different sensitivities can be used

1.2 Connection between measuring probes and handheld device

The different probe are connected with the handheld device with a 6-wire-ribbon cable. Two wires are used for the power supply. A third wire connects the voltage output of the hall sensor ICs to the handheld. Two of the remaining three wires are used to identify the different hall sensor ICs and thereby the different sensitivities.

1.3 The handheld in itself

The handheld is based on a Arduino Micro and a custom shield. An inside is given by **Figure 3**. The

Figure 3: View inside the handheld with the Arduino

device is powered by a nine-volt battery. The voltage for the microcontroller is directly regulated with

a 7805 (fixed positive voltage regulator). Thus the Arduino Micro is powered via the port "+5V". The output of the 7805 is also used for the sensor chip, the display and the display background LED, too. The LCD is connected to the Arduino Micro in 4-bit mode and without the "rw contol" to minimise the wiring effort. As already mentioned in **Chapter 1.2** the Arduino Micro can check automatically which sensor is linked. Therefor two digital ports are used to detect whether the corresponding channel is pulled to ground or 5 V. The measured signal is connected to the analog port "A0". The schematic of the developed shield can be found in **Appendix II**. The PCB layout was designed with EAGLE. It is shown in **Appendix III**.

2 Software

The Arduino Micro is flashed by the USB port with the Arduino IDE and Arduino code. The complete code is attached to **Appendix I**. The LCD is controled by the Arduino Micro using the `LiquidCrystal Library`. For timing purposes the `TimerNull Library` is used. It provides the functionality of the timer interrupts of the ATmega 32U4. To access the measured data the variables are initialised as `volatile`.

2.1 The `setup()` function

Within the `setup()` function the LCD will be started and static data will be written.

The measurement should be performed with an averaging. Every single point of measurement shall be taken every 50 µs. For this reason the timer interrupt will be set to this time and be attached as interrupt.

2.2 The `loop()` function

Nothing will be perfomed. The Arduino Micro is just waiting for instructions from the timer interrupt.

2.3 The `readHall()` function and the sub-functions

Every times the timer interrupts the magnetic flux density will be measured and stored in a collection of all current measuring values. The number of all measuring values is increased by one. When 5000 measurements are performed the average will be calculated and shown on the display. To be able to measure also alternating fields the rms-value will be used as averaging method:

$$B_{\mathrm{rms}} = \sqrt{\frac{\sum_{i=1}^{n=5000} \left(\frac{\frac{U_{\mathrm{in},i}}{1023} \cdot 5000 - U_{\mathrm{ref}}}{k_{\mathrm{divider}}} \right)^2}{5000}}$$

$U_{\mathrm{in},i}$: value returned by `analogRead()`

U_{ref}: reference voltage which is returned by sensor at $B = 0\,\mathrm{T}$

k_{divider}: divider to convert measured voltage into a flux density, unit: $\frac{\mathrm{mV}}{\mathrm{mT}}$

The values U_{ref} and k_{divider} depend on the type of hall sensor IC and will be set by the function `setSensorTyp()`. This functions evaluates the digital status (`HIGH` or `LOW`) on the inputs "A2"' and "A3" and sets the sensor specific values to the volatile variables. If no sensor is identified an error is returned to the customer: `Sensor?`.

Finally the measured magnetic flux density will be shown on the display.

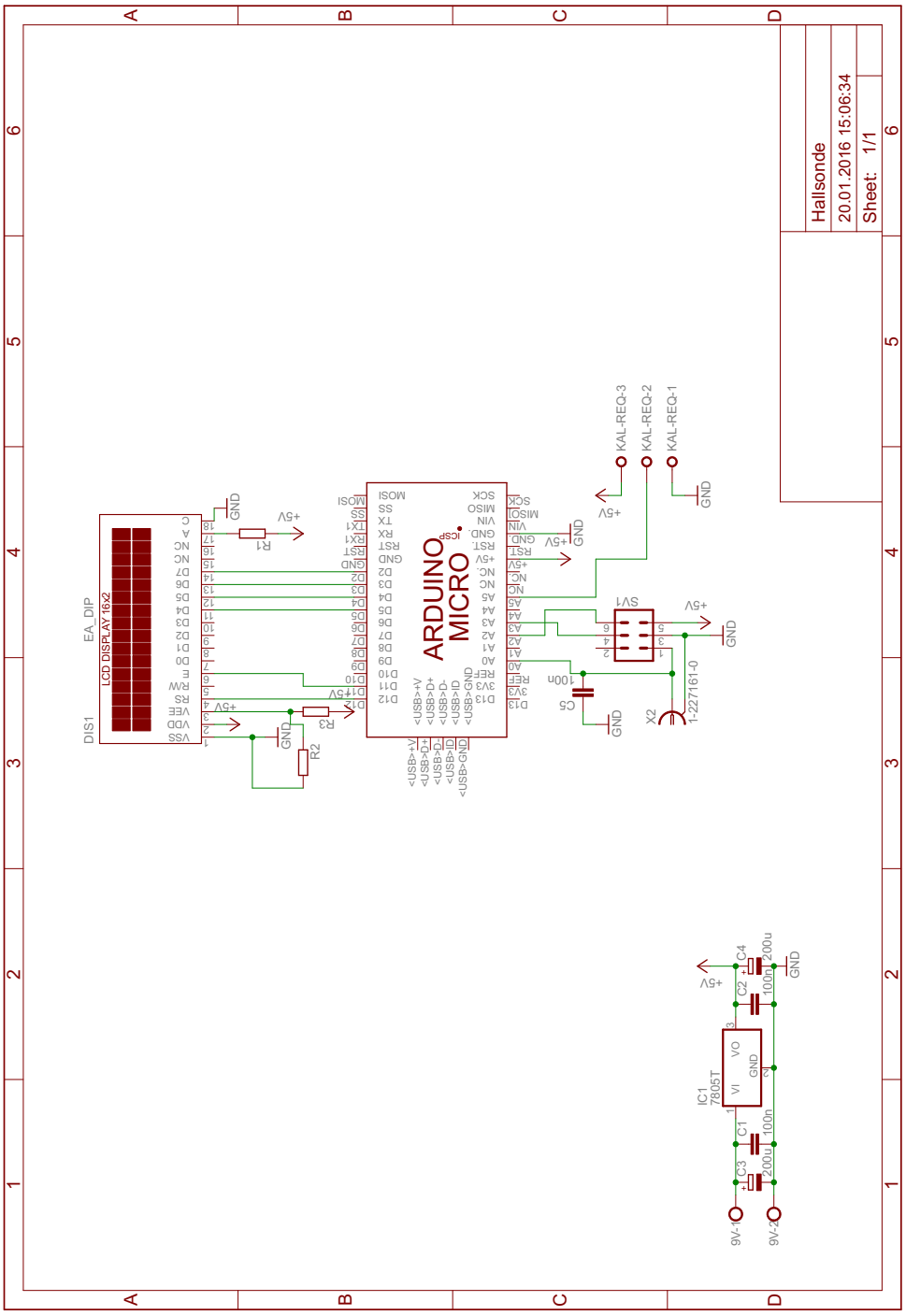
3 Appendix

I The Arduino code

```
/*
 Hallsonde

 Misst mit verschiedenen Hallsensoren eine Magnetische Flussdichte und stellt
    diese auf einem Display dar.

  The circuit:
 * LCD RS pin to digital pin 12
 * LCD Enable pin to digital pin 11
 * LCD D4 pin to digital pin 5
 * LCD D5 pin to digital pin 4
 * LCD D6 pin to digital pin 3
 * LCD D7 pin to digital pin 2

 von Henning Janssen
 */

// include the library code:
#include <LiquidCrystal.h>
#include <TimerNull.h>
//#include <TimerOne.h>


// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// initialize input
volatile int hallPin = A0; //Eingang A0
volatile float hallRef = 2500; //Spannung bei B = 0 mT
volatile double messwertsammlung = 0; //Speicher fuer Messwerte
volatile float messwert = 0;
volatile int messwertAnz = 0;  //Zaehler fuer Messwertanzahl
volatile float sensordivider = 25; //Sensorverhaeltnis in mT/mV


void setup()
{
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  lcd.clear();
  lcd.print("Hall-Sonde aktiv");

  setSensorTyp();

  Timer0.initialize(50);
//  Timer1.initialize(500000);
  Timer0.attachInterrupt(readHall);
}

void loop(){
//Warte und tue nichts
```

```
}

void readHall()
{
  if (5000 == messwertAnz) showRMS();

  messwert = (((float) analogRead(hallPin))/1023*5000 - hallRef)/sensordivider;
  messwert = messwert*messwert;
  messwertsammlung += messwert;
  messwertAnz++;
}

void showRMS()
{
  messwert = sqrt(messwertsammlung/messwertAnz);
//  messwert = messwertsammlung/messwertAnz;
  messwertsammlung = 0;
  messwertAnz = 0;

  setSensorTyp();

  lcd.setCursor(7,1);
  lcd.print("␣␣␣␣␣␣␣␣␣␣␣␣␣");
  if (messwert < 10) lcd.setCursor(9,1);
  else lcd.setCursor(8,1);
  lcd.print(messwert);
  lcd.setCursor(13,1);
  lcd.print("␣mT");
}

void setSensorTyp()
{
  String sensorTyp;
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
  int typPin4 = digitalRead(A2);
  int typPin5 = digitalRead(A3);
  if (LOW == typPin4 & LOW == typPin5){         //Sensor 1 = SS495A
    hallRef = 2480;
    sensorTyp = "SS495A␣";
    sensordivider = 31.25;
  }else if (HIGH == typPin4 & LOW == typPin5){  //Sensor 2 = SS496A
    hallRef = 2470;
    sensorTyp = "SS496A␣";
    sensordivider = 25;
  }else if (HIGH == typPin4 & HIGH == typPin5){  //Sensor 3 = Allegro A1302K
    hallRef = 2499;
    sensorTyp = "A1302K␣";
    sensordivider = 13;
  }else sensorTyp = "Sensor?";

  lcd.setCursor(0,1);
  lcd.print(sensorTyp);

}
```

II Schematic of the shield



III Layout of the PCB