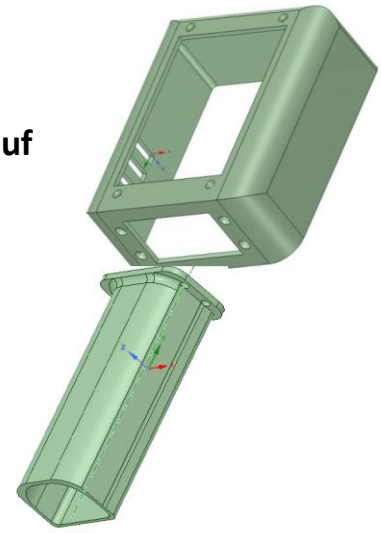


Bauanleitung zum Bau eines Feinstaubhand- Messgerätes auf Basis des SDS011 Sensormoduls

Dipl. –Ing. R. Kufferath Okt. 2020
Hochschule Niederrhein

Diese Anleitung soll dazu dienen es interessierten Personen zu ermöglichen die Luftqualität der Umgebung zu messen.

Das OK-Lab Stuttgart „entwickelt und baut bezahlbare open source DIY Feinstaub-Sensoren. Anschließend wurden diese über die Stadt Stuttgart (und darüber hinaus Europa- und Weltweit) verteilt um ein genaueres Bild der Feinstaubverteilung zu bekommen.“ www.codefor.de/stuttgart

Die bereits erzielten Ergebnisse dieses Projektes sind auf der Webseite www.luftdaten.info einzusehen. Es sind bereits beachtliche Daten erfasst worden, sodass für Deutschland ein nahezu flächendeckendes Sensornetzwerk mit minutenaktuellen Messwerten entstanden ist.
Die Messstellen sind fest montiert und über W-LAN angebunden.
Um auch an anderen Stellen aktuelle Messdaten der Feinstaubkonzentration PM10 und PM2.5 zu erfassen, soll dieses Messgerät dienen.

Bauteileliste:

- 3D-Druckteile des Gehäuses
- SDS011 Feinstaubsensor
- Arduino UNO Mikrocontroller
- Nextion-Display Typ:
- Micro USB Wemos ESP32 Plug 18650 Battery Shield
- Schalter
- Verbindungskabel

Allgemeine Hinweise:

Neben der Erfassung der Feinstaubdaten des SDS011 Sensors muss die Energieversorgung sichergestellt werden und eine Ausgabe der Messwerte möglich sein.
Optional wäre eine Eingabe von Messparametern und Speicherung der Messwerte wünschenswert. Daher fiel die Wahl auf ein Display mit Touchfunktion der Firma Nextion, welches sich leicht per Windows App programmieren lässt und dem leichten Empfang von seriellen Daten erlaubt.

So ist es möglich die aktuellen Messwerte für PM10 und PM2.5 sowie Temperatur/Feuchte darzustellen und daneben Minimal- und Maximalwert, sowie Mittelwert auszugeben. Außerdem ist auch eine grafische Darstellung der Messwerte über die Zeit realisiert.

Die Energieversorgung erfolgt über eine Standard LIPO Zelle 18650 mit ca 2500 mAh und einem Batterie Shield, welches eine konstante Spannung von 5 Volt oder 3,3 Volt abgibt, sowie über eine MikroUSB Buchse incl. Laderegler verfügt, welche ein komfortables Laden ermöglicht. Die Betriebszeit des Gerätes ist mit einer Ladung mehr als 12 Stunden.

Erhältlich hier:
https://www.ebay.de/itm/Micro-USB-Wemos-ESP32-Plug-18650-Battery-Shield-V3-ESP-32-f-Arduino-Raspberry-Pi/273807522795?ssPageName=STRK%3AMEBIDX%3AIT&_trksid=p2057872.m2749.l2649

Das Gehäuse ist aus 4 Teilen konstruiert um eine einfache Montage zu ermöglichen und sich mit relativ kleinen 3D Druckern wie zum Beispiel dem Monoprice Mini V2 Modell mit einer Druckfläche von 120 x 120mm herstellen lässt. (die entsprechenden stl Dateien sind im Downloadbereich zu finden)

Programmierung des Arduino UNO:

Der Arduino Sketch beinhaltet zunächst folgende Bibliotheken:

SdsDustSensor.h	für die Erfassung der Feinstaubwerte des SDS011
SoftwareSerila.h	für den zweiten Seriellen Port für das Display
Nextion.h	UM das Display bequem ansteuern zu können

Er lässt sich in folgende Hauptteile gliedern

Erfassung der Feinstaubwerte
Ausgabe der Werte per Serial.Print zur Kontrolle am PC
Ausgabe der aktuellen Werte auf dem Display
Speichern der Werte in einem Array zur Berechnung von min/max/mean
Ausgabe von min/max/mean sowie der Anzahl der Messwert
Bei Umschalten auf dem Display in den Graph-Modus erfolgt ein Plot der PM10 und PM2.5 Werte bis maximal 60µg/m^3

Zur Ausgabe der Messwerte grafisch über das Display kann eine Eingabe am Touchdisplay erfolgen, die eine Umschaltung des Programms ermöglicht, indem ein serieller Befehl an den Arduino gesendet wird.

<u>Auszug aus dem Arduino Programms:</u>

```
#include "SdsDustSensor.h"  // Bibliothek einbinden
#include <SoftwareSerial.h>
#include <Nextion.h>

Variablendeklaration ……..

void setup() {
  Serial.begin(9600);
myNextion.init();
  sds.begin();
  Serial.println(sds.setCustomWorkingPeriod(0).toString()); // Sensor sendet alle (x) Minuten einen Wert. Bei (0) kontinuierlich
 // Serial.println(sds.setCustomWorkingPeriod(10).toString()); //  HIer Sensor sendet alle (10) Minuten
}
```

```
void loop() {

  PmResult pm = sds.readPm();
  if (xx>warte_in_sec){
     xx=0;
     myNextion.setComponentText("t19", String (loopcount));
  if (pm.isOk()) {          // Wenn Daten anliegen
   Serial.print(", PM10 = ");
   Serial.println(pm.pm10); // hier wird PM10 µ  ausgegeben

  myNextion.setComponentText("t1", String(pm.pm10));  // PM10µ Konzentration ausgeben
  myNextion.setComponentText("t2", String(pm.pm25));  // PM2,5µ Konzentration ausgeben

    myNextion.setComponentText("t5", "20");  //Temperature
    myNextion.setComponentText("t7", "50");  //Humidity
//**********************PM10 calculate ************************
    PM10=pm.pm10;
    PM10f=pm.pm10;
    PM10f=PM10f*4;   // Display hat die Maße 240x240 obere Grenze soll 60µg sein deshalb
240/60=4 Faktor
    PM10i=PM10f; // Umwandeln in Integer;
    if (PM10i >240)
         {PM10i=236;
         myNextion.setComponentText("t3","Over");
         state =1;  }
     if ((PM10i <235)&& state ==1){
                 myNextion.setComponentText("t3", "60ug");
                  state =0;}
  //**********************PM25 calculate ************************

    PM25f=pm.pm25;
    PM25f=PM25f*4;   // Display hat die Maße 240x240 obere Grenze soll 60µg sein deshalb
240/60=4 Faktor
    PM25i=PM25f; // Umwandeln in Integer;
      if (PM25i >240)
         PM25i=233;

  //*************   wave ausgeben  ************************
  nextion.print("add 1,0,");
  nextion.print(PM10i);  //PM10 Value multiply by 4 for better display the Wave
  nextion.write(0xff);
  nextion.write(0xff);
  nextion.write(0xff);
   delay (30);
//*********************************************************
    nextion.print("add 1,1,");
```

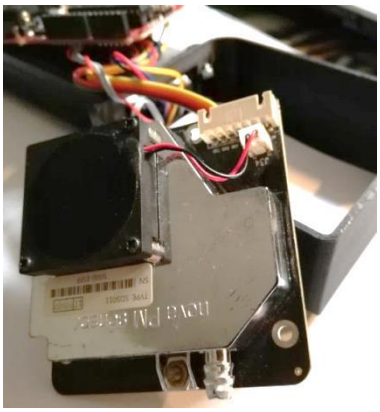
```
 nextion.print(PM25i);  //PM10 Value multiply by 4 for better display the Wave
 nextion.write(0xff);
 nextion.write(0xff);
 nextion.write(0xff);
  delay (30);

 PMx[y]=PM10;
 y++;
       } else {  }

  x++; // Counter
  if (x==20)x=0;
//---------------MEAN ----------------------
if (y==10){
 y=0;
 for (z=0;z<10;z++)
      {mean+= PMx[z];
       }
       mean=mean/10;
  myNextion.setComponentText("t14", String (mean));
  mean=0;
//-----------------max --------
   for ( z = 0; z < 10; z++ )
      {
       if ( PMx[z] > max_v )
          { max_v = PMx[z];
            max_i = z;  }
      }
 myNextion.setComponentText("t16", String (max_v));
 max_v=0;
 //-------------- min ----------------
  for ( z = 0; z < 10; z++ )
      {
       if ( PMx[z] < mini )
          mini = PMx[z];  }

       myNextion.setComponentText("t18", String (mini));
       mini=1000;
       }
 loopcount++;
if (y==10) y=0;}
xx++;
   delay(1000);

}
```



Anschluss des SDS011 Feinstaubsensors.

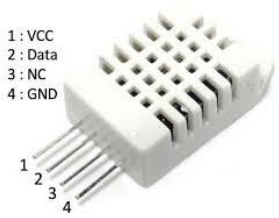
Es werden benötigt:

Ground	0V
Vss/Ubb	5V
RX(SdS011)	-> PIN 7 Arduino (Definitionen über rxPin in Software)
TX(SdS011)	-> PIN 6 Arduino

Wenn der Arduino UNO nun per USB Kabel verbunden ist, und die Verkabelung korrekt ist, müssten im Seriellen Terminal bereits Werte ausgegeben werden.
Sollte dies nicht der Fall sein bitte überprüfen.

Anschluss des DHT22 Temperatur/Feuchtesensors an Arduino.

1 Vcc	5V
2 Data	-> PIN 9 Arduino (Definitionen über DHTPIN in Software)
3 NC	--
4 GND	0V Ground



Hinweis: Zur Zeit laufen die beiden Bibliotheken für den DHT22 Feuchtesnsor und den SDS011 Feinstaubsensor nicht gleichzeitig. Wahrscheinlich kommen sich Interruptservice Routinen in die Quere. Wir arbeiten an einer Lösung. Bis dahin wird ein Dummy Wert von 20°C und 50% angezeigt.
Wer eine Lösung des Problems gefunden hat kann sie gern teilen.

Programmieren des Nextion Displays

Das Nextion TFT Display mit Touch-funktion erlaubt eine einfache Darstellung von Werten die über die serielle Schnittstelle empfangen werden. Sie können in einem Zahlenfeld oder auch grafisch, z.B. durch einen Balken oder Zeiger dargestellt werden.
Auch eine Y/t Darstellung ist möglich.

Dazu wird das Display zunächst mit einem Windows Programm programmiert und VarialbelnNamen definiert und dann die Firmware für das Display überspielt.
Entweder über ein Serielles Kabel , oder über eine SD Karte auf die zuvor die neue Firmware kopiert wurde und die dann einfach in den SD Kartenslot gesteckt wird und anschließend das Display mit Spannung versorgt wird.
Die SD Karte kann dann wieder entfernt werden und für andere Zwecke genutzt werden.

Hier wird bereits die fertige Firmware bereit gestellt, sowie der Quelltext des Nextion Editors, um eigene Variationen vornehmen zu können.

Zu beachten ist, dass hier ein 2,8‘‘ Display verwendet wurde. Das Gehäuse ist so ausgelegt, dass dieses genau passt und festgeschraubt werden kann.
Itead Studio Nextion NX3224T028 - Generic 2.8" HMI LCD Touch Display

Anschluss des Displays an Arduino GND , 5V, RX (Display) -> PIN 3 Arduino, TX (Display) -> PIN2

Nähere Infos zur Programmierung findet man hier:
https://www.boecker-systemelektronik.de/Seite-/-Kategorie-1/Grafische-Oberflaechen-fuer-Mikrocontroller-Anwendungen-mit-Nextion-Displays

Das Display ist erhältlich hier:
https://www.exp-tech.de/displays/lcd/7375/itead-studio-nextion-nx3224t028-generic-2.8-hmi-lcd-touch-display?c=1074
(ca. 20 Euro)

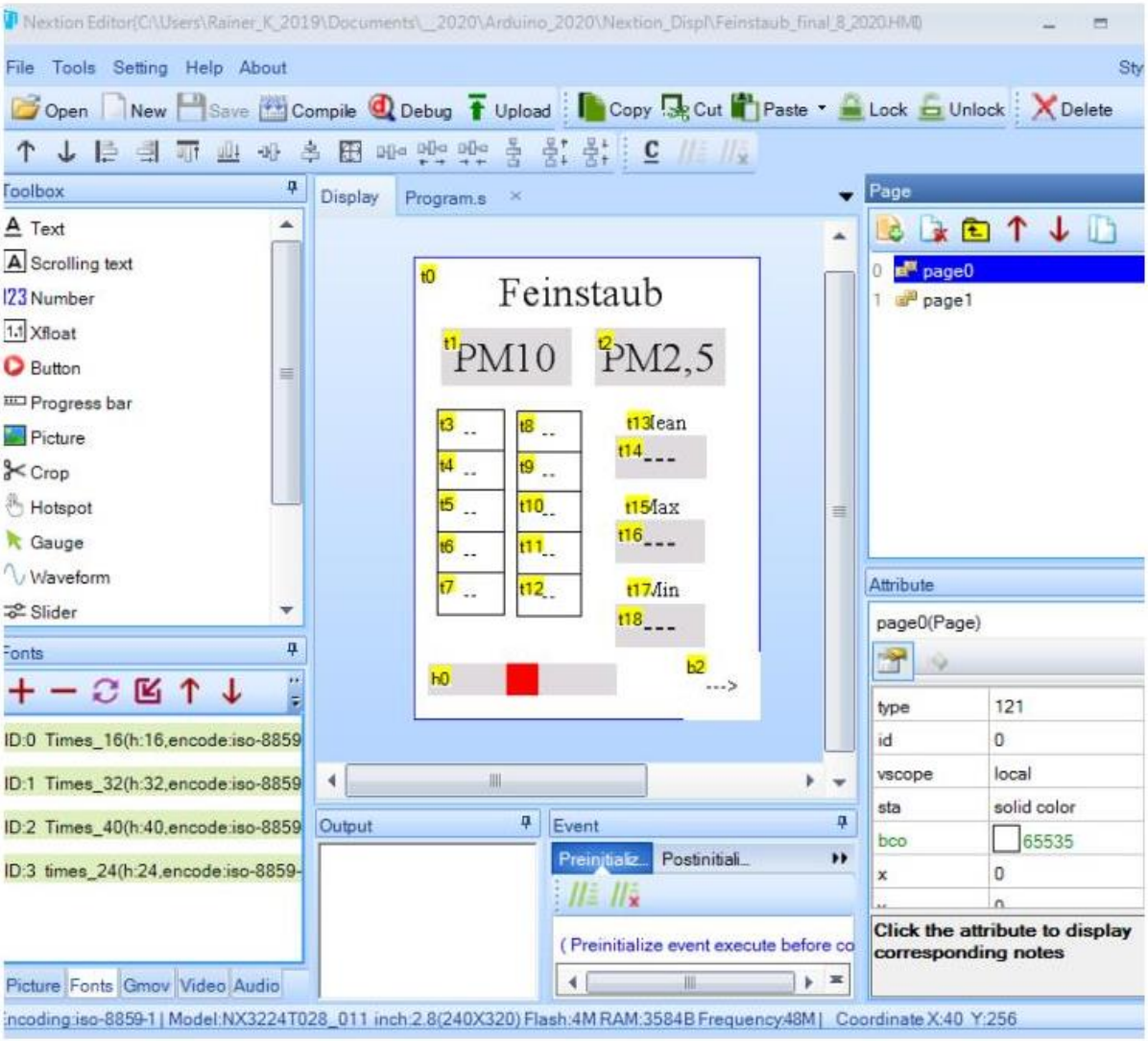
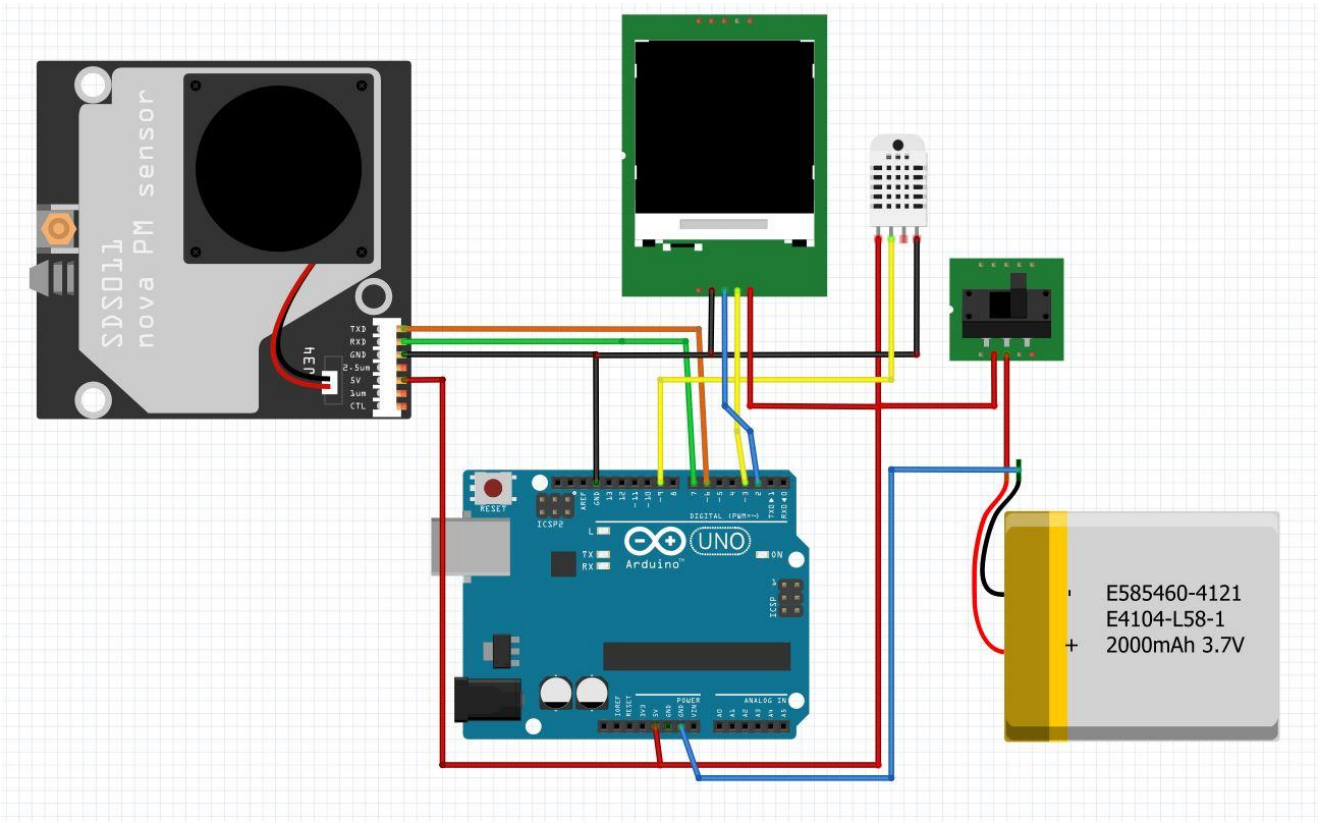


Abbildung des Nextion Editors. In der Mitte das Display mit den einzelnen Variablen-fenstern.

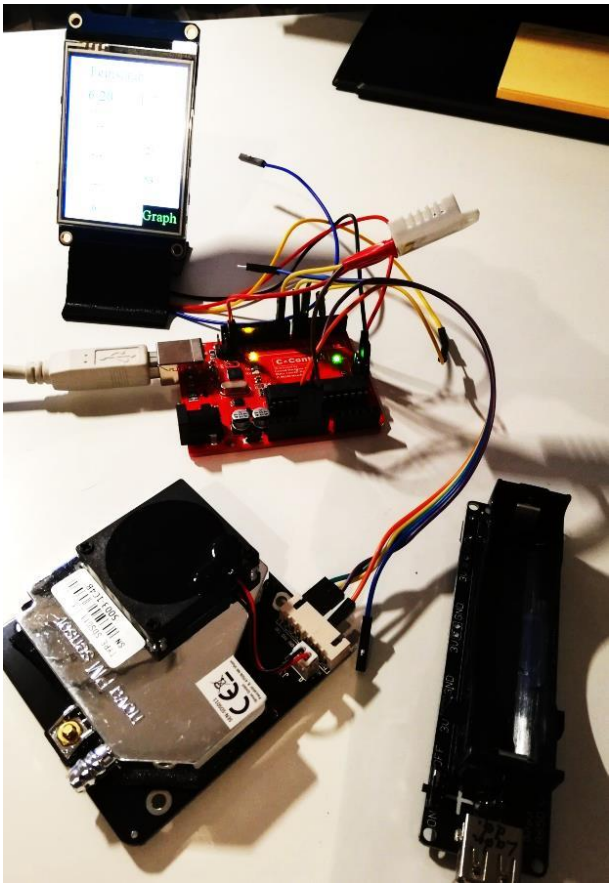
Zusammenbau des Gerätes:

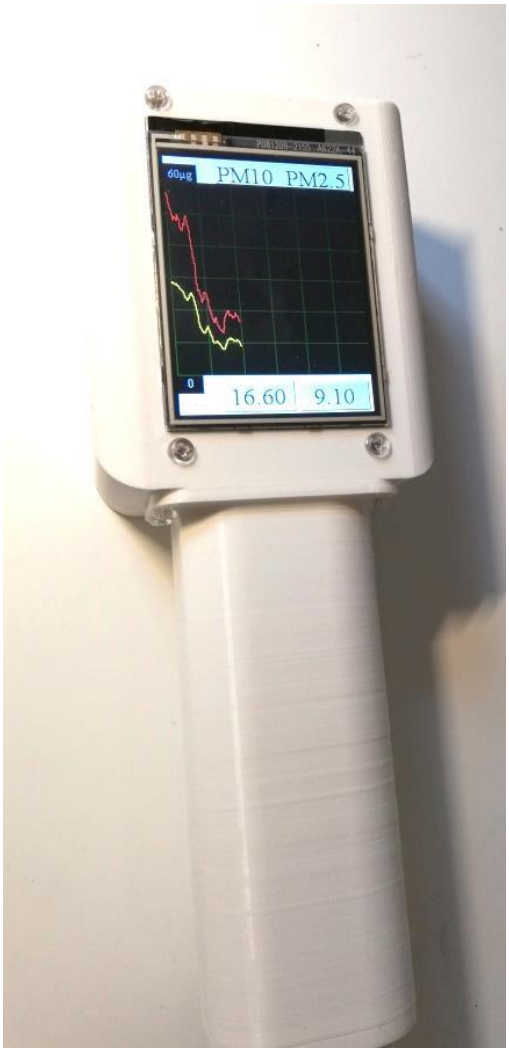
- SDS011 Sensor mit Arduino verbinden
- Arduino 5V PIN über Schalter mit Batterieschild 5V Ausgang verbinden (etwas längeres Kabel verwenden, da der Griff noch festgeschraubt werden muss und das Schild auch im zusammengebauten Zustand herausnehmbar sein soll.
- DHT22 Temperatur/Feuchtesensor anschließen und montieren
- Arduino GND mit GND des Batterieschilds verbinden
- Festschrauben des Displays mit vier 4mm Schrauben am Gehäuse
- Anschluss des Displays an Arduino GND , 5V, RX (Display) -> PIN 3 Arduino, TX (Display) -> PIN2
- Anschluss des SDS011 Sensors RX(SdS011) -> PIN 7 Arduino / TX(SdS011) -> PIN 6 Arduino GND , 5V

- Pappe zur Isolierung innen auf das Display legen, SDS Sensor mit Platinenseite zum Display einlegen, sodass der Ansaugstutzen zum kleinen rechteckigen Fenster zeigt
- Arduino mit Platinenseite zum SDS Sensor ins Gebäuse legen und mit Deckel verschließen. Deckel ist passgenau und hält ohne Verschraubung



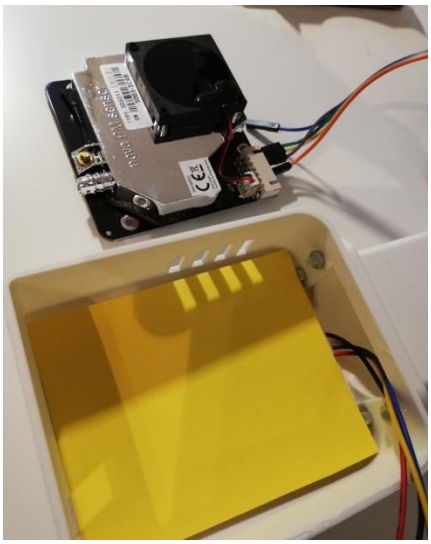
Aufbau des Gerätes in Einzelschritten im Bild dokumentiert:



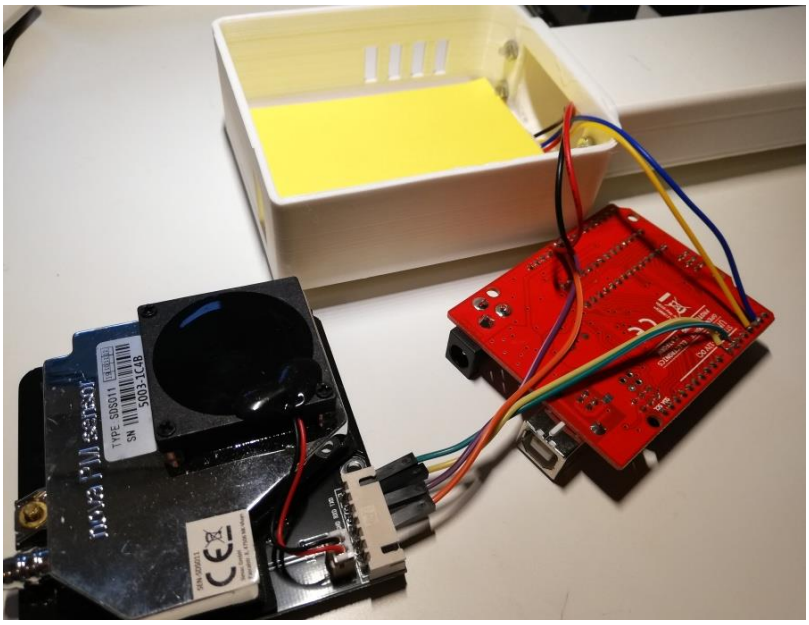


Display Einbau:

Isolation mit Pappe

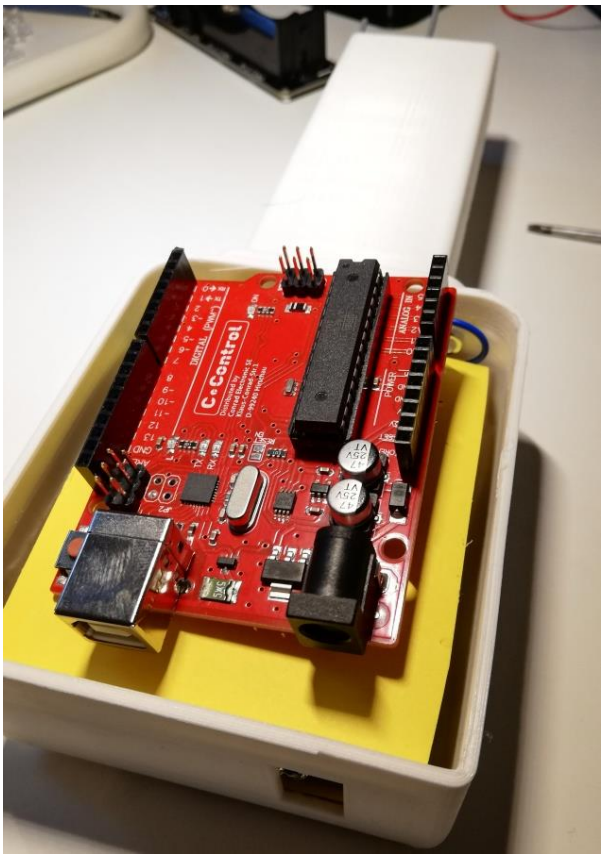


Anschluss des Arduino UNO



Einbau des SDS011 Feinstaubsensors

Einbau des Arduino UNO und Lötanschlüse für Batterie



AKKU Einbau:

Fertiges Gerät: