MultiGeiger

Release V1.16.0

15.08.2021

Inhaltsverzeichnis

1 Übersicht **1**
1.1 Was ist MultiGeiger? 1

KAPITEL 1

Übersicht

1.1 Was ist MultiGeiger?

Der MultiGeiger ist ein Messgerät für Radioaktivität.

1.1.1 Haupt-Eigenschaften

unterstützt mehrere Einsatz-Szenarien

- Stationen mit fixem Standort (Daten-Transfer über Netzwerk)
- mobile Anwendung (Benutzung des OLED-Displays und des Lautsprechers)
- (weitere sind geplant)

mehrere 400V-Geiger-Müller-Zählrohre werden unterstützt

- große, empfindliche Zählrohre, wie das Si21g oder Si22g (für Mess-Stationen)
- kleinere, weniger empfindliche Zählrohre, wie das SBM21 (gut, um Hotspots zu finden)

Hardware

- geringe Zahl an Bauteilen, einfach zusammen zu bauen
- preisgünstig, aber trotzdem gute Bauteile
- benutzt den ESP32, einen modernen und schnellen 32bit-Micro-Controller
- mit WLAN (WPA2) oder mit WLAN + LoRA
- mit OLED-Display

Firmware

- implementiert in C
- benutzt die beliebte Arduino-API

- over-the-air Firmware-Updates

Netzwerk und Community

- dies ist ein Ecocurious-„Citizen Science“-Projekt
- web-basierte Karte für das Veröffentlichen von Messungen
- web-basiertes Archiv von historischen Strahlungs-Werten

Frei und Offen

- GPL v3 lizensiert
- offene Entwicklung und Issue-Tracking auf GitHub

Über

Der **MultiGeiger** ist ein Projekt von Ecocurious, Deiner Umwelt-, Natur- und Technik-Community, siehe https://ecocurious.de/projekte/multigeiger/.

Ziel ist es, ein Bürger-Messnetz für Radioaktivität in Deutschland aufzubauen.

Die **MultiGeiger** Hard- und Software hat Jürgen Böhringer konzipiert (http://www.boehri.de).

Reinhard/rexfue hat die Software und Platine weitergedacht und kümmert sich um die Einbindung der Sensoren in unsere Map https://ecocurious.de/multigeiger-karte/.

Wir haben die ersten Workshops gestartet, in denen Du die Bauteile und das Gehäuse mit unserer Unterstützung zusammenbauen kannst.

Workshop-Termine findest Du hier:

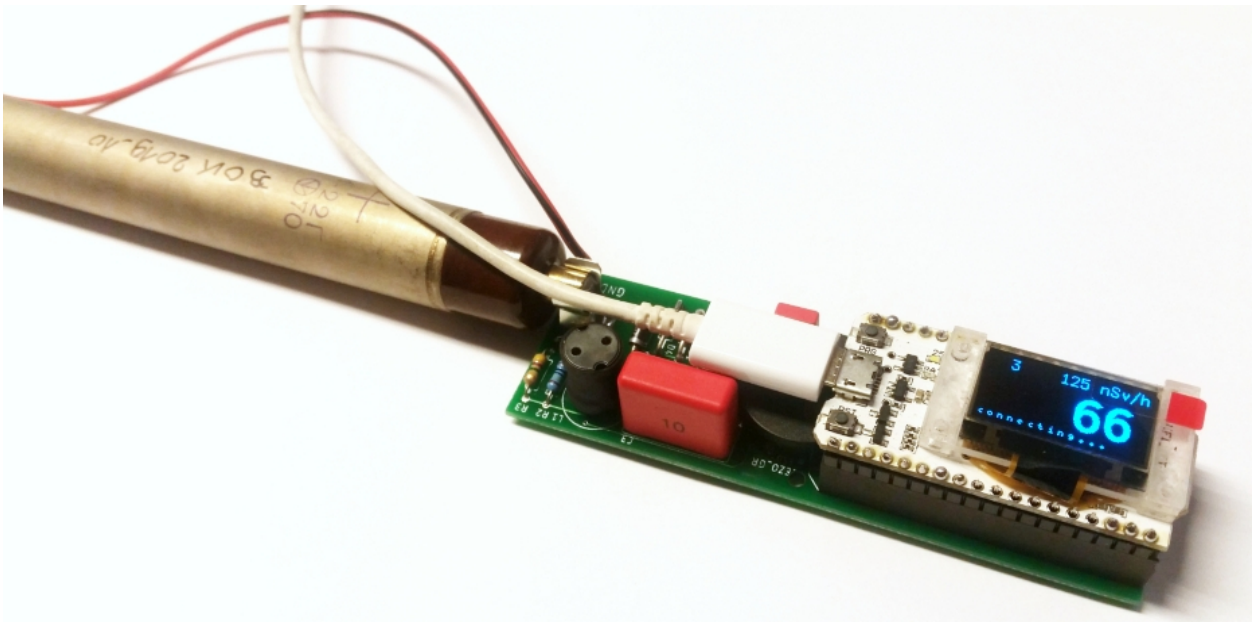
- https://www.meetup.com/de-DE/Ecocurious-deine-Umwelt-Natur-und-Technik-Community/
- https://ecocurious.de/events/

Klingt das interessant für Dich? Dann mach mit, herzliche Einladung!

Assembly

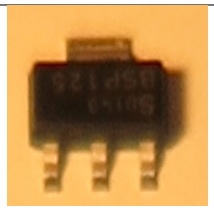
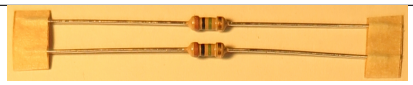
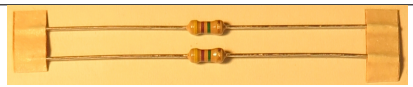
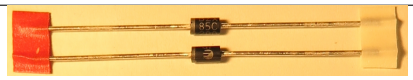
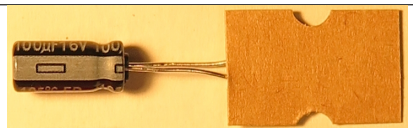
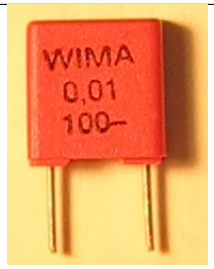
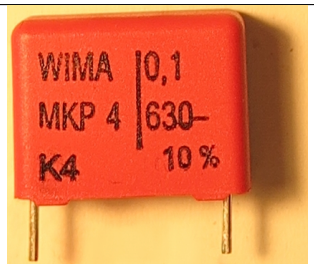
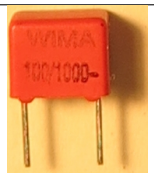
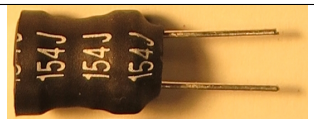
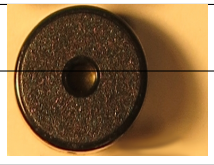
Ergebnis

Das Multigeigerprojekt ist eine Idee der Ecocurious-Initiative. Das dazugehörige Open-Source Multigeigermessgerät zum Messen radioaktiver Gammastrahlung wird von der Community entwickelt. Die Echtzeitdaten der stationären Geräte werden per LoRaWAN oder WiFi/WLAN an einen Server gefunkt und auf einer Karte visualisiert. So entsteht ein Radioaktivitätsmessnetz in Bürgerhand, das wir hiermit aufbauen wollen. Herzliche Einladung zum Mitmachen!



Bauteile

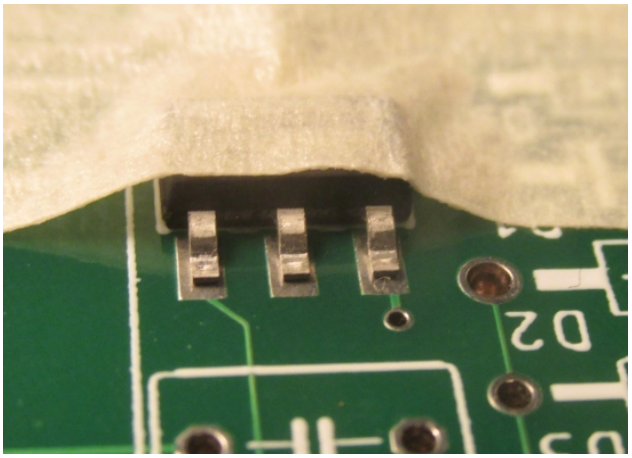
Elektronikbauteile aus (s. Stückliste) zunächst sortieren und auf Vollständigkeit prüfen:

ID(s)	#	Bild	Typ
T1	1		N-MOS-Transistor BSP125
R1	1		10K Widerstand braun-schwarz-orange-gold
R2 R5	2		1,0M Widerstand braun-schwarz-grün-gold
R3 R4	2		4,7M Widerstand gelb-violett-grün-gold
D1 D2	2		Z-Diode ZY 200
D3	1		Z-Diode BZX55C3V3
D4	1		Gleichrichterdiode BYV 26E
D6	1		Gleichrichterdiode GP10Y
C1	1		ELKO Elektrolytkondensator 100 uF, 16 V
C2	1		Folienkondensator 10nF, 100V 0.01 100-
C3	1		Folienkondensator 100nF, 630V 0.1 630-
C4	1		Folienkondensator 100pF, 1000V 100/1000-
L1	1		Spule 150uH 154J
Pz1	1		Piezo-Lautsprecher

Lötarbeiten

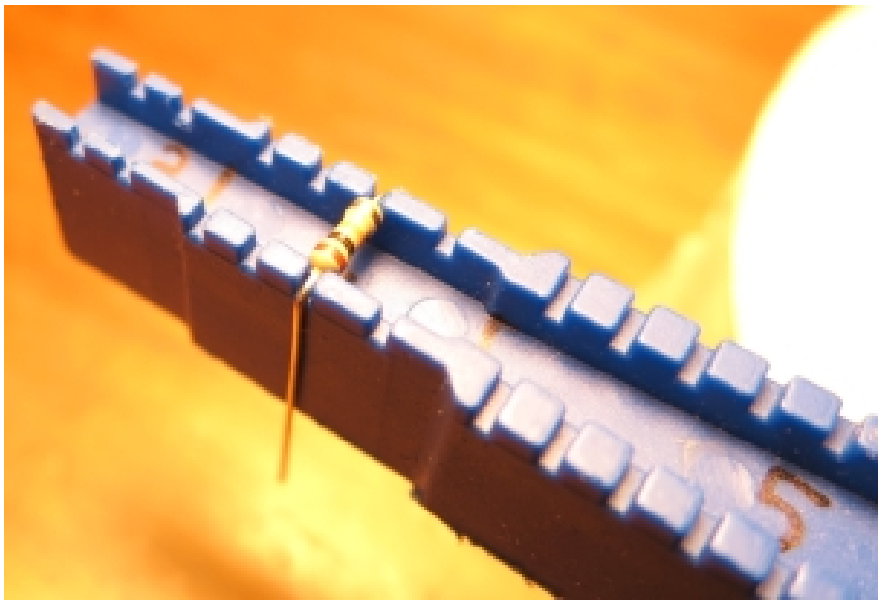
Feldeffekt-Transistor T1

Als erstes Bauteil wird das einzige Bauteil im SMD-Bauform montiert. Mit Klebeband (Kreppband hat sich bewährt) wird er so auf die Platine geklebt, so dass die drei Pins sichtbar sind. Dann werden die vorderen Anschlüsse verlötet, das Klebeband vorsichtig entfernt und der hintere Anschluss verlötet.

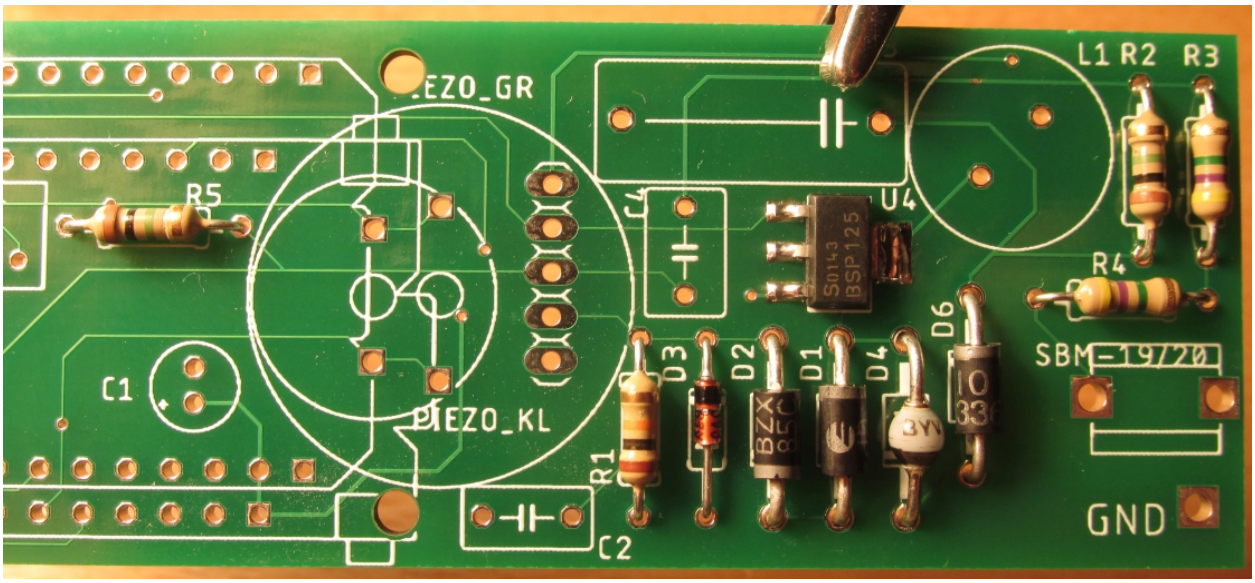


Widerstände und Dioden

Die Widerstände und Dioden werden alle in einer Abbiegevorrichtung (oder entsprechend mit der Hand oder einer kleinen Zange) gebogen. Die Breite ist immer vier Einheiten – sprich ca. 10 mm.

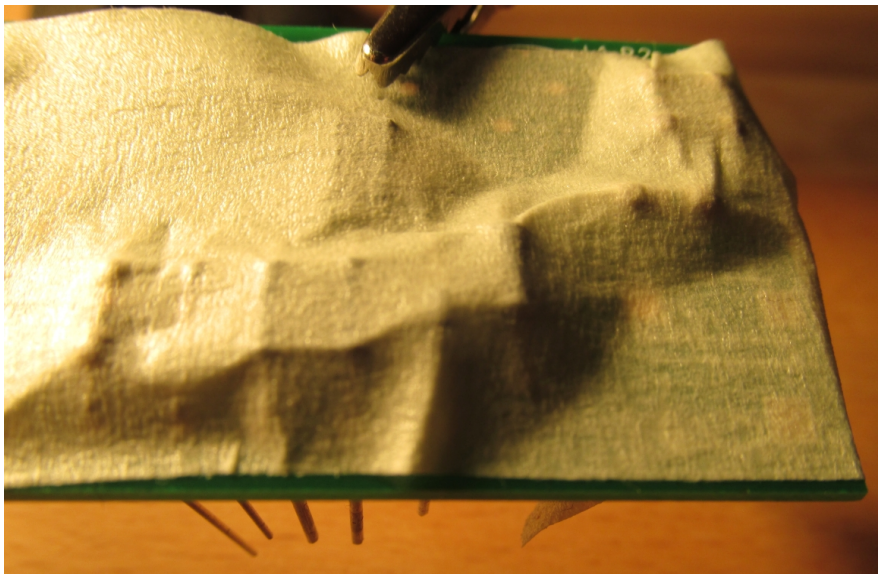


Anschließend wird die Platine mit den Widerständen und Dioden bestückt. Bei den Dioden unbedingt auf die Polarität achten! Die Kathode (Minus-Pol) ist mit einem Strich auf Platine und Bauteil markiert.



Vor dem Löten der Unterseite werden die Bauteile auf der Oberseite mit Klebeband fixiert, oder die Beinchen der Bauteile durch leichtes Auseinanderbiegen verklemmt. Dann Bauteile anlöten, Klebeband entfernen und überschüssigen Draht abschneiden.

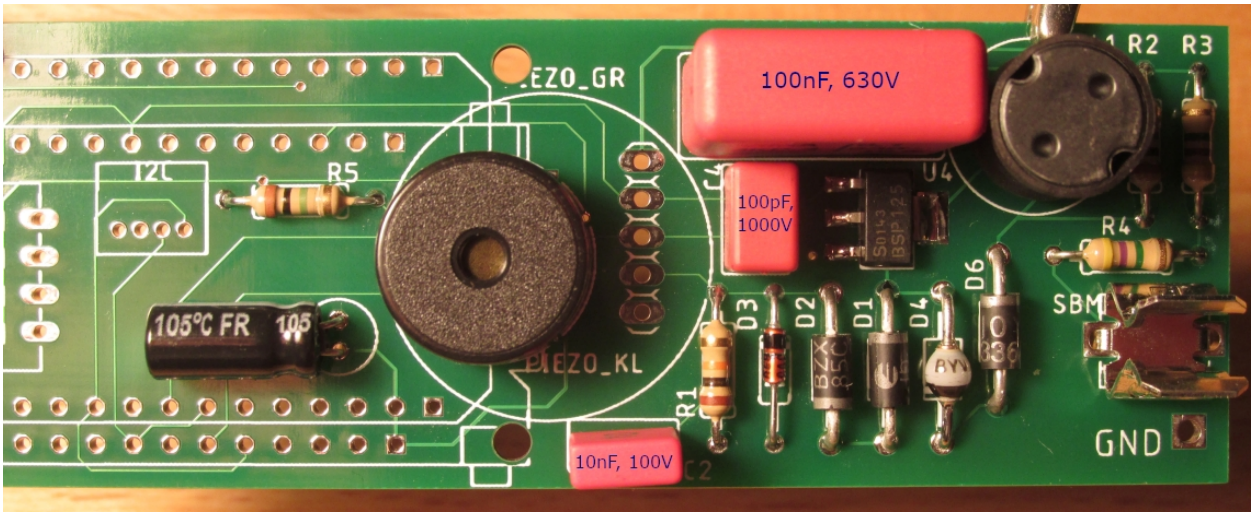
Bemerkung: Verletzungsgefahr: Beim Abschneiden darauf achten, dass der davonfliegende Draht niemanden verletzten kann. Möglichst festhalten beim Abschneiden!



Kondensatoren und restliche Bauteile

Der Elektrolytkondensator (Elko) wird vorher mit einer Flachzange angewinkelt. Beim Elko unbedingt auf die Polung achten: Auf dem Elko ist der „–“-Pol markiert, auf der Platine der „+“-Pol. Die Beschriftungen müssen sich gegenüber liegen. Der Zählrohrhalter muss so herum eingesetzt werden, dass das Zählrohr nach außen steht kann – am besten testen! Dann die Bauteile mit Klebeband fixieren, anlöten und den überschüssigen Draht abschneiden.

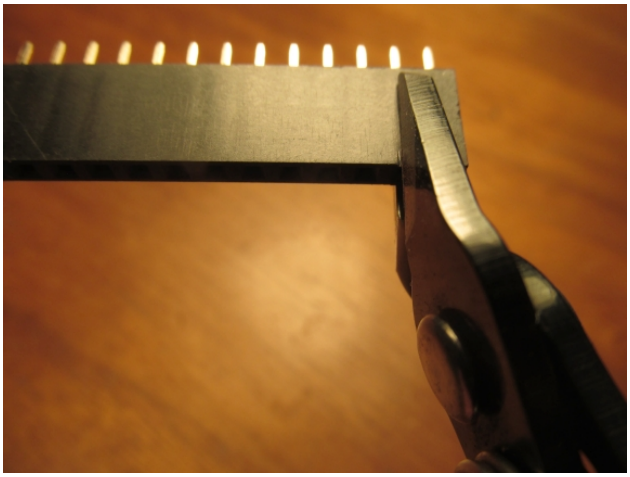
Kondensatoren, Zählrohrhalter, Spule, Piezo-Lautsprecher sind bestückt:



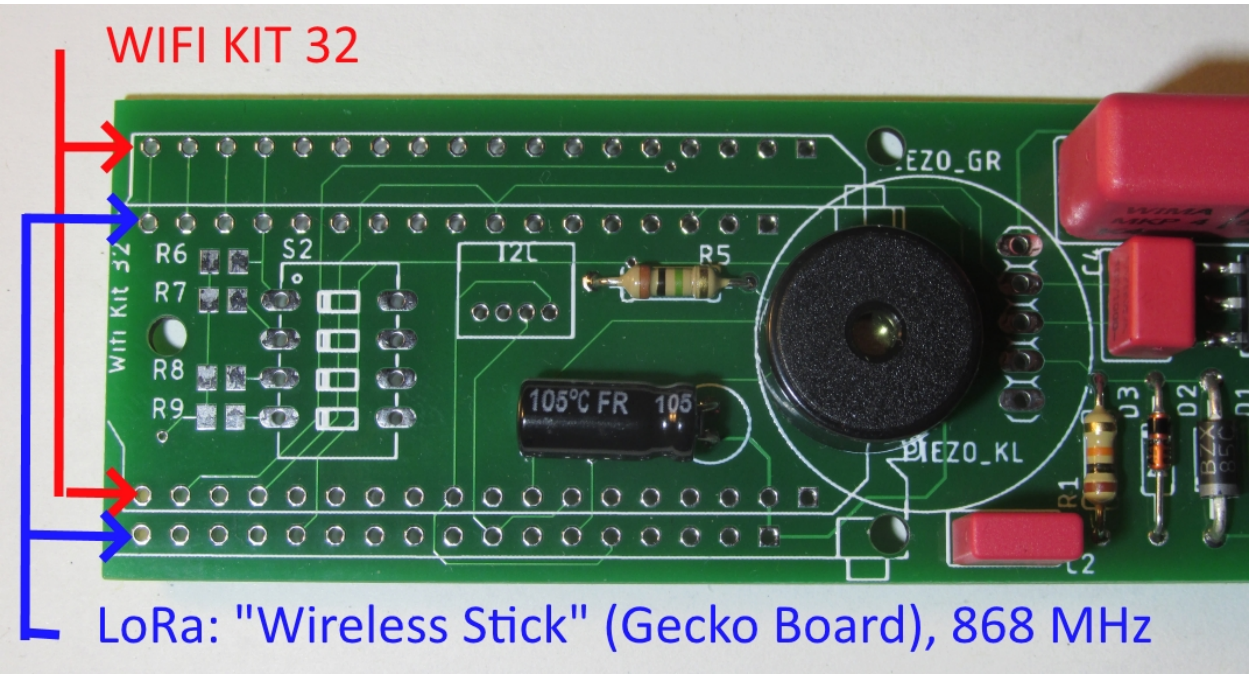
Buchsenleisten und Mikrocontroller-Modul

Die Buchsenleiste muss vorher evtl. noch gekürzt werden.

Bemerkung: Dabei muss der Seitenschneider genau auf dem **letzten nicht mehr benötigten Kontakt** angesetzt werden – **nicht dort wo man ihn eigentlich kürzen will** – ansonsten kann sie splittern. Beim Abschneiden wegfliegende Teile möglichst festhalten!



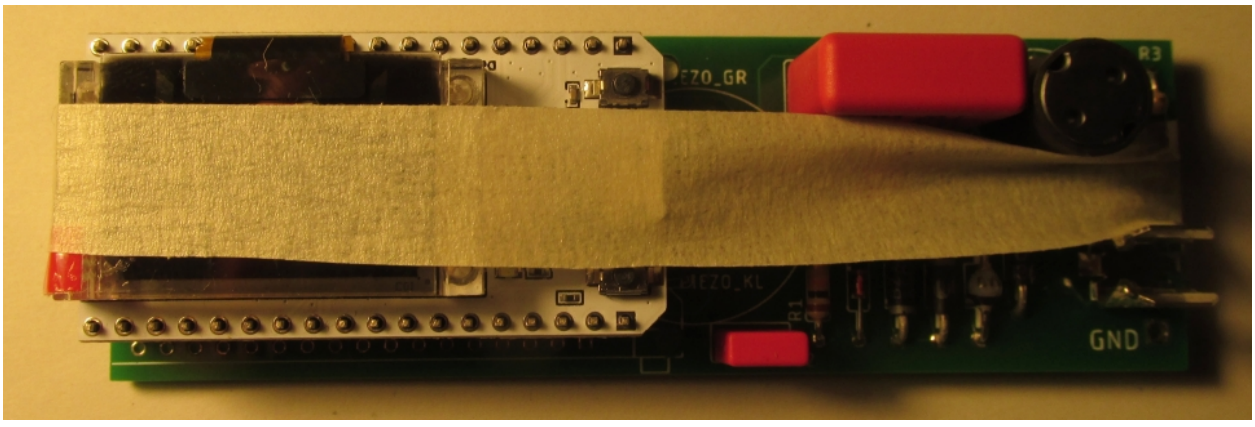
Wird der große Piezo-Lautsprecher verwendet, so sollte die Buchsenleiste mit einer Feile etwas abgefeilt werden (vorher ausprobieren). Am einfachsten geht das an der originalen, nicht bearbeiteten Seite der Buchsenleiste. Je nachdem, welches Mikrocontroller-Board verwendet wird, müssen die Buchsenleisten an den entsprechenden Stellen bestückt werden.



Die Winkel der Buchsenleiste sollten genau stimmen. Daher am besten so vorgehen:

- Die erste Buchsenleiste einstecken.
- Von unten her eine Lötstelle in der Mitte löten, die Lötstelle nochmals heiß machen und gleichzeitig die Buchsenleiste justieren (rechter Winkel, alle Pins richtig).
- Die zweite Buchsenleiste einstecken.
- In beide Buchsenleisten die Stiftleiste aus dem Mikrocontroller-Modul-Set einstecken, so dass deren längerer Teil der Stiftleiste in der Buchsenleiste steckt.
- Das Mikrocontroller-Modul aufstecken, so dass das Display sichtbar ist und die Mikro-USB-Buchse über dem Piezo-Lautsprecher liegt.
- Alles mit einem schmalen Klebeband fixieren (siehe Bild unten), so dass alle Lötstellen zugänglich sind.
- Nun können alle Kontakte gelötet werden.

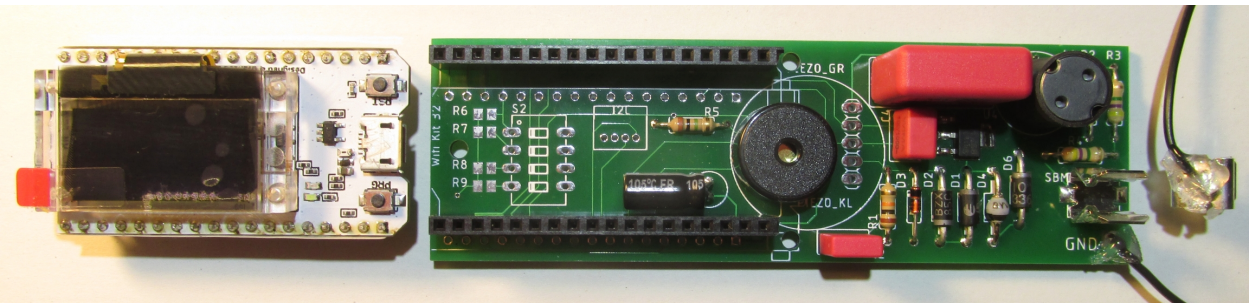
Bemerkung: Das OLED-Grafikdisplay ist über eine dünne Flex-Leitung angeschlossen, die nicht zu heiß werden darf. Im Zweifel lötet man die Pins in der Nähe der Flex-Leitung nicht an – sie werden nicht benötigt.



Nach dem Löten kann das Mikrocontroller-Modul mit wippenden Bewegungen vorsichtig abgezogen werden.

- Beim Mikrocontroller-Modul sind Pin-Beschriftungsaufkleber mit dabei. Diese können jetzt seitlich aufgeklebt werden. Welcher Pin wo ist, steht auf der Unterseite des Mikrocontroller-Moduls.
- Das Kabel für den Anschluss der Zählrohr-Kathode (Minus-Anschluss) an Klammer und Platine anlöten.
- Kabel mit jeweils einem Tropfen Heißkleber sichern (Zugentlastung).
- Sichtkontrolle (hast du Lötstellen vergessen, sind Lötbrücken entstanden, . . .).
- Mikrocontroller-Modul wieder aufstecken.
- Zählrohr anklemmen: der Plus-Pol (Anode) ist markiert und muss in Richtung Platine zeigen.
- Eine erste Funktionskontrolle ist nun möglich, s. *Inbetriebnahme*.

Fertige Geigerzähler- und Mikrocontrollerplatine:



Bemerkung: Das dünne Glas des OLED-Grafikdisplays ist empfindlich. Bitte keine Kraft auf das Display ausüben, sondern nur auf die Mikrocontrollerplatine!

Warnung: Verletzungsgefahr: Auf der Geigerzählerplatine wird eine **Spannung von 400 Volt** erzeugt. Diese hält sich auch noch längere Zeit **nach dem Ausschalten des Geräts**. Bei Berührung kann es zu kleineren Stromschlägen kommen, die normalerweise harmlos sind. Dennoch sollten sie vermieden werden!

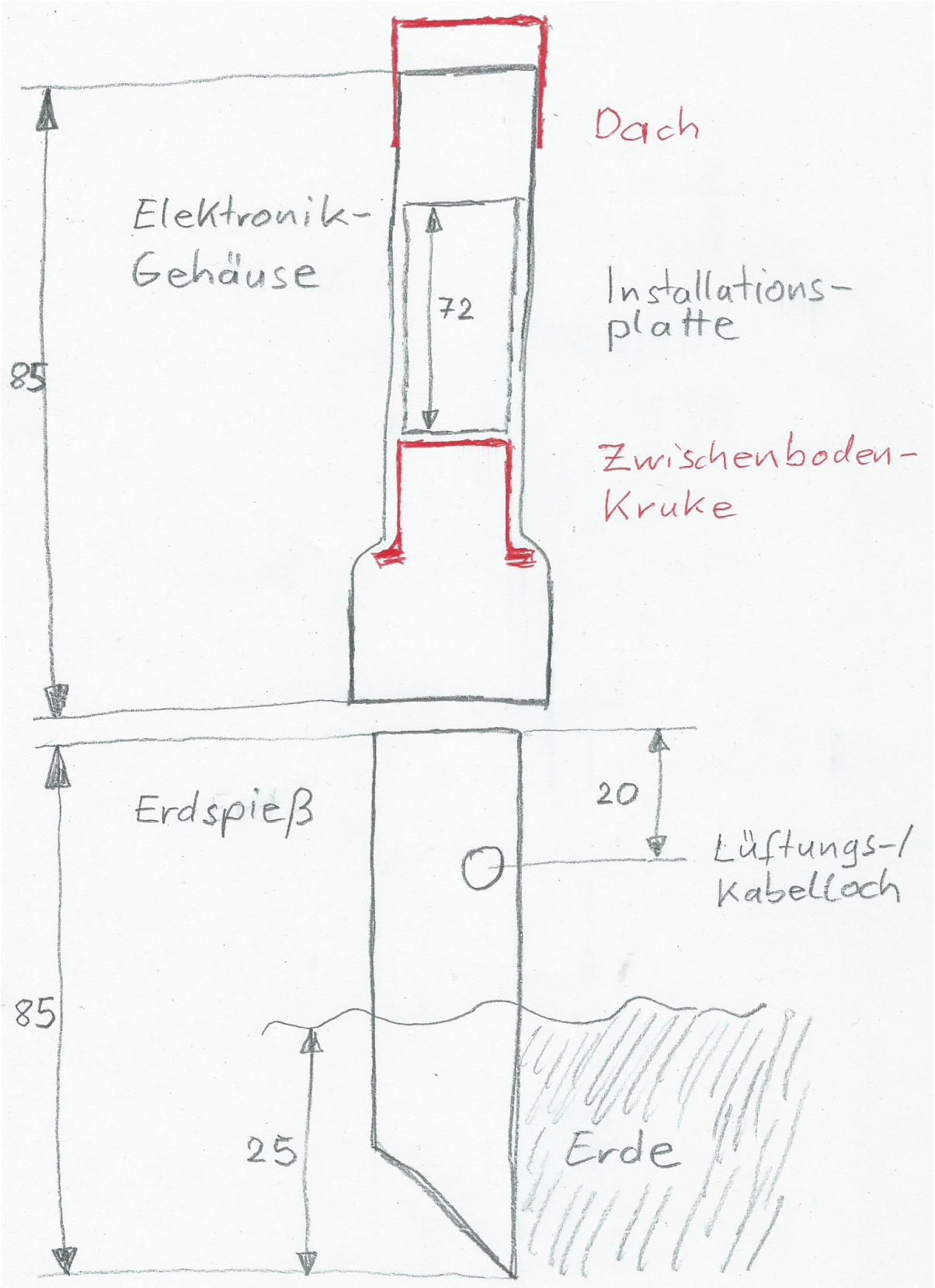
Man sollte vor dem Arbeiten an der Schaltung den Kondensator entladen. Am einfachsten macht man dies mit einer zweiten, ca. 30 cm langen Leitung, mit der man Anode und Kathode des Zählrohrs für etwa eine Sekunde kurzschließt. Dies schützt nicht nur vor unangenehmen Stromschlägen, sondern schützt auch die empfindliche elektronische Schaltung.

Gehäuse

Funktionen des Gehäuses

- Es schützt die Elektronik vor Umwelteinflüssen (z. B. Regen) und Insekten (daher sollte kein Zugang von außen zur Elektronik größer sein als 0,5 mm).
- Es hält das Zählrohr in einer Höhe von 1 m über der Wiese – somit sind unsere Messwerte (bestmöglich) vergleichbar zu denen des amtlichen ODL-Messnetzes.

Rohre


Dach
Elektronik-
Gehäuse
72
Installations-
platte
85
Zwischenboden-
Kruke
Erdspieß
20
Lüftungs-/
Kabelloch
85
25
Erde

- Rohr für das Elektronikgehäuse: Zunächst wird das Rohr mit einer Säge auf entsprechende Länge abgesägt. Die Muffe (Aufweitung) zeigt nach unten.
- Rohr für den Erdspieß: Der noch umgesägte Teil des Restes ist das obere Ende des Erdspießes. Unten wird der Erdspieß auf entsprechende Länge abgesägt im 45-Grad-Winkel. Mit diesem schrägen Anschnitt lässt sich der Erdspieß später mit leichten Drehbewegungen gut in die Erde bohren. Alternativ kann man diesen Teil des Rohrs kurzhalten (ca. 15 cm) und in einem verzinkten Sonnenschirmhalter mit Erdspieß (Ø40mm) festklemmen.
- Lüftungs-/Kabelloch bohren: Mit einem Reißnagel o. ä. wird das Loch seitlich im Erdspieß vorgestochen. Anschließend wird es aufgebohrt. Zuletzt wird es mit einem großen Bohrer / Schälbohrer auf etwa 17 mm Durchmesser aufgebohrt.

Bemerkung: Verletzungsgefahr: Mit der Klinge/dem Bohrer immer nach außen, d. h. vom Körper weg, schneiden oder bohren!

Dach und Zwischenboden

Für das Dach kann man einen Rohrabschlussstopfen oben vorsichtig über das Rohr schieben, s. Bild rechts.

Als Zwischenboden verwenden wir eine Kruke, also eine Plastikdose, die in Apotheken zum Abfüllen von Salben verwendet wird. Der rote Deckel kann als Material für eine kabeldurchführungsverkleinernde Zwischenbodenauflage dienen.

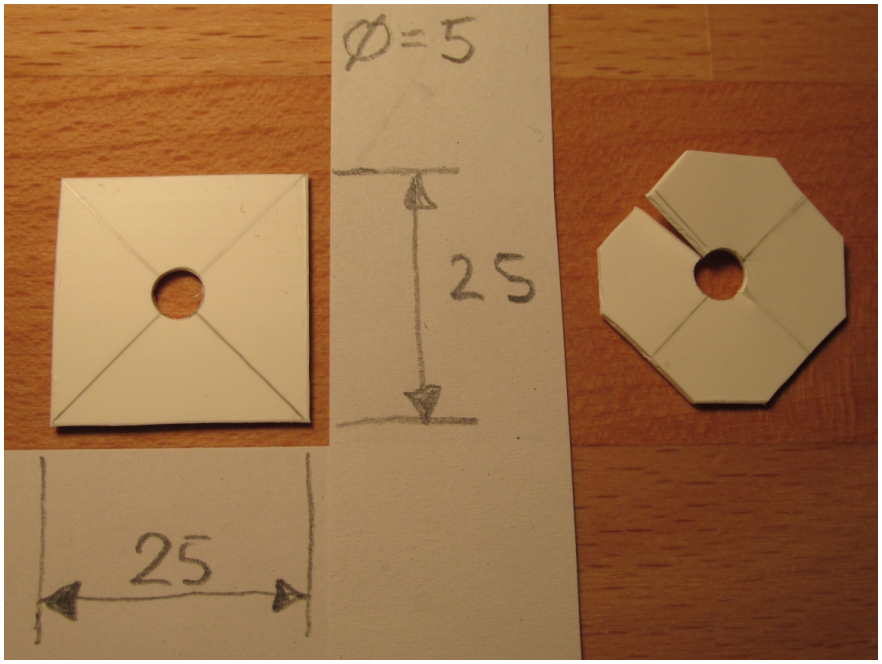
Für den Zwischenboden wird lediglich das Gewinde vorsichtig mit der Säge abgesägt, s. Bild links. Der „Kragen“ muss bestehen bleiben, da er als Anschlag dient.

In den Zwischenboden wird in der Mitte ein Loch zur Kabeldurchführung mit 10 mm gebohrt. Der Stecker des Kabels muss durch das Loch passen.

Eine Zwischenbodenauflage dient zum Schutz vor Insekten, da sie das Loch der Kabeldurchführung verkleinert. Man kann dafür z. B. den roten Deckel der Kruke oder ein ähnliches Stück Kunststoff verwenden.

Es wird die Mitte markiert und dort ein Loch mit dem Durchmesser des Stromkabels (ca. 5 mm) gebohrt. Anschließend werden die Ecken abgeschnitten. Zusätzlich wird ein Verbindungsschlitz von außen zum Loch geschnitten.

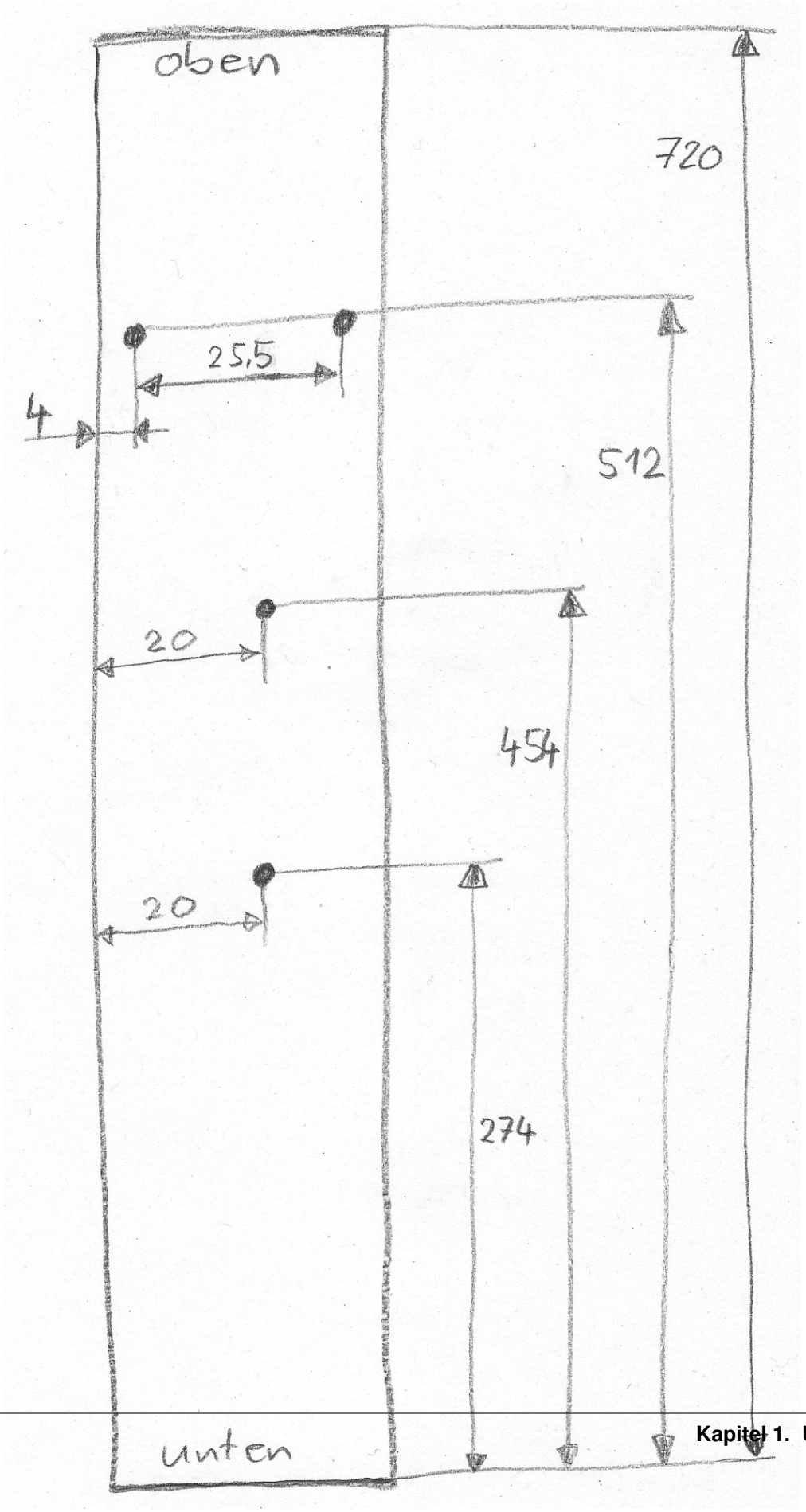
Aufbau der Zwischenbodenauflage (in zwei Arbeitsschritten):



Installationsplatte

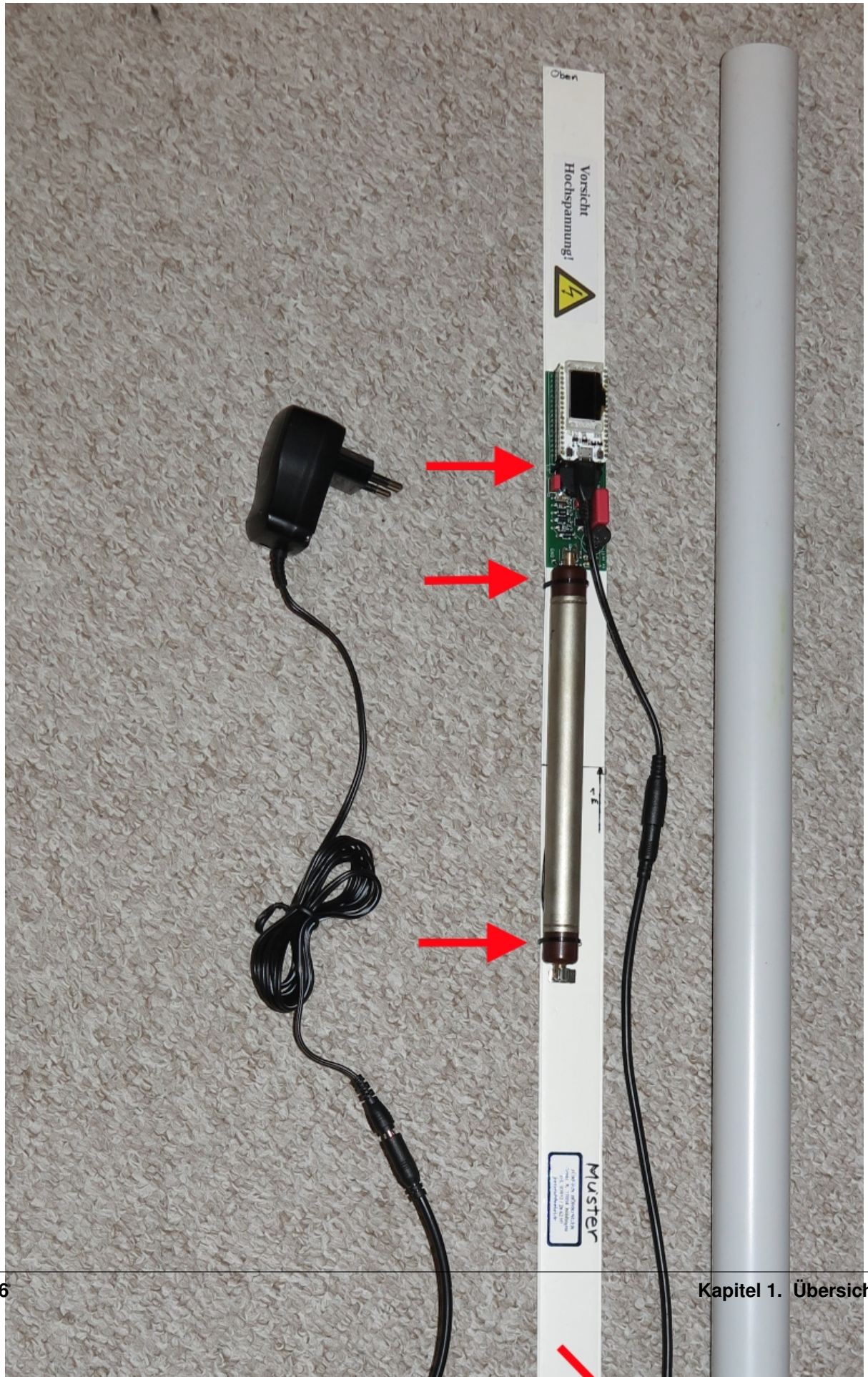
- Die Installationsplatte kann aus dem Ober- oder Unterteil eines Kabelkanals gefertigt werden. Dazu werden die Seitenteile vorsichtig mit einem Messer abgeschnitten. Danach wird der dadurch entstandene, biegsame Kunststoffstreifen auf die richtige Länge gebracht. Hierzu kann eine starke Schere verwendet werden.
- Löcher durch Installationsplatte bohren: Elektronik und Zählrohr werden so mit Kabelbindern auf Installationsplatte fixiert, dass das Zählrohr sich auf 1 m Höhe befindet, wenn der Erdspieß 25 cm in der Erde steckt. Das Zählrohr zeigt nach unten. Die Löcher werden mit dem Reißnagel vorgestochen und dann auf 3 mm aufgebohrt, s. Bild „Bohrplan Installationsplatte“.
- Die Platine wird lediglich mit einem Kabelbinder auf der Platte montiert, der durch beide Löcher gezogen wird. Er verläuft unterhalb des USB-Steckers.

Bohrplan Installationsplatte:


oben
720
25,5
4
512
20
454
20
274
unten

Einbau

Einbau des Geräts (die roten Pfeile markieren Kabelbinder):


Oben
Vorsicht
Hochspannung!
Muster

- Platine und Zählrohr werden mit Kabelbinder auf die Installationsplatte montiert.
- In die Micro-USB-Buchse wird der passende Adapter gesteckt.
- Das „10 m Verlängerungskabel“ wird richtig herum zunächst durch den Erdspieß und dann durch den Zwischenboden gefädelt. Anschließend wird er mit dem Adapter verbunden.
- Danach wird die Zwischenbodenauflage oberhalb des Zwischenbodens vorsichtig auf das Kabel geschoben.
- Oberhalb der Zwischenbodenauflage wird ein Kabelbinder auf das Kabel gezurrt, so dass eine Zugentlastung entsteht.
- Erster Test des Gehäuses: Wir schieben erst vorsichtig die Installationsplatte in das Elektronikgehäuse. Anschließend folgen Zwischenboden und der Erdspieß.
- Wenn alles passt wird noch die Zwischenbodenauflage mit Silikon-Kleber auf den Zwischenboden geklebt und mit Kreppband (bis zum Aushärten) fixiert.
- Der Warn-Aufkleber „Vorsicht Hochspannung“ wird auf die Installationsplatte geklebt.

Inbetriebnahme

WLAN-Variante: Das Gerät baut einen eigenen WLAN-Accesspoint (AP) auf. Die SSID des AP lautet ESP32-xxxxxxxx, wobei die xxx die Chip-ID des WLAN-Chips sind (Beispiel: ESP32-51564452). Bitte diese Nummer notieren, sie wird später für die Anmeldung benötigt, z. B. bei https://devices.sensor.community/. Der Access-Point bleibt für 30 s aktiv. Danach versucht das Gerät, sich mit dem (früher) eingestellten WLAN zu verbinden. Dieser Verbindungsversuch dauert ebenfalls 30 s. Kommt keine Verbindung zustande, wird wieder der eigene AP für erzeugt. Das standardmäßig vergebene WLAN-Kennwort ist „*ESP32Geiger*“ und sollte zeitnah geändert werden.

Weitere Informationen: *Aufstellung* und *Benutzung*.

Wir wünschen viel Freude mit dem neu gebauten Gerät und hoffen dass es niemals unnatürliche bzw. gesundheitsschädliche Ausschläge messen wird!

Setup

Download and unpack the latest release from GitHub (https://github.com/ecocurious/MultiGeiger/releases) as source code (zip) or source code (tar.gz). In the new directory, open the directory *multigeiger* and load the file *multigeiger.ino* with the Arduino IDE.

Die Platine unterstützt zwei verschiedene Heltec-Bausteine, verschiedene Zählrohre und optional einen Temperatur/Luftdruck/Luftfeuchtigkeits-Sensor (BME280 oder BME680). Die Software kann via Netzwerk Daten zu verschiedenen Services senden.

- **Heltec WiFi Kit 32** Diese MCU hat ein großes Display und WiFi. Auf dem Board wird dieser Baustein in die längeren Buchsenleisten gesteckt.
- **Heltec Wireless Stick** Diese MCU hat ein sehr kleines Display, dafür aber zusätzlich zu WiFi noch LoRa. Es wird in die kürzeren Buchsenleisten gesteckt.

Um die Heltec-Boards in der Arduino IDE auswählen zu können, müssen die folgenden Schritte unternommen werden:

- Die Datei: https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json muss in Preferences->Zusätzliche Boardverwalter-URLs hinzugefügt werden.
- Danach können unter Werkzeuge->Board:->Boardverwalter... die ESP32 Boards (Name „esp32 by Espressif Systems“) installiert und anschließend unter Werkzeuge->Board das richtige Heltec Board ausgewählt werden.
- Für **beide** Boards muss hier der **Heltec wireless Stick** ausgewählt werden.

- Wenn Du arduino-esp32 >= 1.0.5 benutzt, musst Du ggf. einen Patch anwenden, um das „partition scheme“-Menü, das im nächsten Schritt gebraucht wird, wieder zurück zu bekommen - siehe die Dateien `misc/arduino-esp32/paritition-menu-*.*` in unserem git-Repo.
- Unter Tools muss noch **Flash size: „4MB(32Mb)“** und **Partition Scheme: „Minimal SPIFFS (Large APPS with OTA)“** eingestellt werden. Die Software erkennt selbstständig, welches Board bestückt ist.

Diverse Software-Einstellungen kann man über folgende Dateien machen (siehe Kommentare dort):

- **./multigeiger/userdefines.h** (immer notwendig, ein Beispiel hierzu wird in userdefines-example.h mitgeliefert)
- **./platformio.ini** (nur bei platformio, ein Beispiel hierzu wird in platformio-example.ini mitgeliefert)

Es werden alle externen Libraries benötigt, die aufgelistet sind in der Datei

```
platformio-example.ini
```

unter dem Abschnitt

```
lib-deps =
```

Bitte die aktuellste Version über platform.io / Libraries installieren.

Achtung: Wenn die Arduino-IDE verwendet wird, dann bitte prüfen, dass in der Datei project_config/lmic_project_config.h (in der obersten Ebene in dieser Library) die richtigen Einstellungen gesetzt sind. Die Datei muss so aussehen:

```
// project-specific definitions
#define CFG_eu868 1
//#define CFG_us915 1
//#define CFG_au921 1
//#define CFG_as923 1
// #define LMIC_COUNTRY_CODE LMIC_COUNTRY_CODE_JP /* for as923-JP */
//#define CFG_in866 1
#define CFG_sx1276_radio 1
//#define LMIC_USE_INTERRUPTS
```

Die angegebenen Versionen der Libraries sind die Mindestanforderungen. Wir testen immer auch mit den neuesten Versionen, deshalb bitte immer die aktuellen Versionen installieren und verwenden. Falls der Compiler andere Libraries anmahnt, diese bitte in der Arduino IDE per Sketch -> Include Library -> Manage Libraries .. installieren.

Ablauf nach dem Start

Das Gerät baut einen eigenen WLAN-Accesspoint (AP) auf. Die SSID des AP lautet **ESP32-xxxxxxxx**, wobei die xxx die Chip-ID (bzw. die MAC-Adresse) des WLAN-Chips sind (Beispiel: **ESP32-51564452**). **Bitte diese Nummer notieren, sie wird später noch gebraucht.** Dieser Access-Point bleibt für 30sec aktiv. Danach versucht das Gerät, sich mit dem (früher) eingestellten WLAN zu verbinden. Dieser Verbindungsversuch dauert ebenfalls 30sec. Kommt keine Verbindung zu Stande, wird wieder der eigene AP erzeugt und der Vorgang fängt wieder und wieder von vorne an.

Einstellung des WLAN

Nachdem der WLAN-AP des Geräts auf Deinem Handy oder PC erscheint, verbinde Dich mit diesem. Die Verbindung fragt nach einem Passwort, es ist **ESP32Geiger**. Normalerweise öffnet sich automatisch die Startseite des Gerätes. Falls die Startseite nicht erscheint, so muss mit dem Browser die Adresse **192.168.4.1** aufgerufen werden. Die Startseite erscheint und dort findet man einen Link zur **configure page** - dort drauf klicken und man kommt zur Einstellungsseite.

Die Einstellungs-Seite hat die folgenden Zeilen:

- Geiger accesspoint SSID Dies ist die SSID des eingebauten APs und kann geändert werden. Wenn der Sensor bereits mit dieser Nummer bei sensor.community registriert war, muss eine neue Registrierung erfolgen.
- Geiger accesspoint password Dies ist das Passwort für den eingebauten AP. Dieses **MUSS** beim ersten Mal geändert werden. Wenn gewünscht, kann natürlich auch das gleiche Passwort **ESP32Geiger** wieder verwendet werden. Das Eingabefeld darf nicht leer gelassen werden. Sichere das Passwort in Deinem bevorzugten Passwort-Manager.
- WiFi client SSID Hier muss die SSID des WLANs für den Netzwerk/Internet-Zugang eingegeben werden.
- WiFi client password Und hier das zugehörige Passwort.

Für bessere Sicherheit wird empfohlen, ein separates WLAN-Netzwerk zu verwenden (z.B. das Gast-Netzwerk), um die Kommunikation vom normalen Netzwerk zu abzutrennen.

Ist alles eingegeben, auf **Apply** klicken. Dann werden die eingestellten Daten übernommen und in das interne EEPROM gespeichert. Dann bitte diese Seite über **Cancel** verlassen, weil nur dann das Programm den Config-Mode schliesst und sich mit dem heimischen WLAN verbindet. Wenn es kein **Cancel**-Knopf gibt, dann gehe zurück zu den WLAN-Einstellungen des Geräts und tippe dort die normalen Heim-Netzwerk-Parameter wieder ein.

ACHTUNG, **Beim Update auf die Version 1.13 müssen die WLAN-Daten noch einmal neu eingegeben werden**. Bei zukünftigen Versionen wird das nicht mehr nötig sein.

Weiter können über die Einstell-Seite einige verschiedene Definitionen festgelegt werden:

- Start-Melodie, Lautsprecher-Tick, LED-Tick und Anzeige jeweils an oder aus
- Daten senden zu sensor.community oder/und zu madavi.de
- bei LoRa-Hardware können hier auch die LoRa-Parameter (DEVEUI, APPEUI und APPKEY) eingegeben werden.

Am Ende der Einstellungsseite gibt es einen Link **Firmware update** - hiermit kann man die Firmware auf dem MultiGeiger aktualisieren. Die .bin-Datei runterladen, dann diese über **Browse..** auswählen und **Update** klicken. Es dauert ca. 30sec für das Hochladen und Flashen der Firmware. Wenn Du dann **Update Success! Rebooting…** siehst, startet der MultiGeiger neu und die neue Firmware ist dann aktiv.

Erscheint **Update error: …**, dann hat das Update nicht geklappt - es ist dann die seitherige Firmware weiter aktiv.

Die Einstellseite kann zu jeder Zeit aus dem eigenen WLAN heraus aufgerufen werden. Dazu wird in der Adresszeile des Browsers http://esp32-xxxxxxx eingegeben (xxxxxxx ist die Chip-ID - siehe oben). Sollte es mit diesem Hostnamen nicht klappen, dann muss die IP-Adresse des Geigerzählers verwendet werden. Diese kann aus dem Router ausgelesen werden. Bei Erfolg erscheint zunächt eine Login-Seite. Hier ist als Username **admin** und als Passwort das ausgewählte Passwort (siehe oben) einzugeben. Dann erscheint die gleiche Einstellseite wie beschrieben.

Server für die Messdaten

Es werden jeweils einen Messzyklus lang die Impulse gezählt und dann die „Counts per Minute“ (cpm) berechnet. Jeweils nach diesem Zyklus werden die Daten zu den Servern bei *sensor.community* und bei *madavi.de* gesendet.

Bei *sensor.community* werden die Daten gespeichert und stehen am nächsten Tag zum Abruf als CSV-Datei bereit. Diese Datei findet man unter http://archive.sensor.community/DATE/DATE_radiation_si22g_sensor_SID.csv) wobei DATE = Datum im Format YYYY-MM-DD ist (beides mal gleich) und SID die Sensornummer des Sensors (**nicht** die ChipID). Für andere Sensoren ist der Zählrohr-Name **si22g** durch den entsprechenden Namen zu ersetzen (z.B.: sbm-20 oder sbm-19).

Bei *madavi* werden die Daten in einer RRD-Datenbank abgelegt und können direkt als Grafik über diesen Link aufgerufen werden: https://www.madavi.de/sensor/graph.php?sensor=esp32-CHIPID-si22g Hier ist dann CHIPID die ChipId (also die Ziffern der SSID des internen Accesspoints).

Während der Übertragung der Daten zu den Servern wird in der Statuszeile (unterste Zeile) des Displays kurz der Name des Servers eingeblendet.

Anmelden bei sensor.community

Um Mess-Daten an sensor.community zu senden, muss man dort ein gültiges Konto haben und der Sensor muss registriert sein. Beides kann man auf https://devices.sensor.community erledigen. Wenn man noch kein Konto hat, über „Register“ eins anlegen und sich einloggen. Um einen neuen Sensor zu registrieren, auf „Register new sensor“ klicken und das Formular ausfüllen:

- Erste Zeile, Sensor ID: Hier die Nummer (nur die Zahlen) der SSID des Sensors eingeben (z.B. bei dem Sensor ESP-51564452 dann nur 51564452 eingeben).
- Zweite Zeile, Sensor Board: *esp32* auswählen (über die kleinen Pfeile rechts)
- Basisinformation: Hier die Adresse mit Land eingeben. Der interne Name des Sensors kann beliebig vergeben werden, muss aber eingegeben werden. Bitte den Haken bei **Indoor-Sensor** setzen, solange der Sensor nicht draussen aufgestellt ist.
- Zusätzliche Informationen: Kann leer gelassen werden, es ist aber nett, um zusätzliche Informationen anzugeben.
- Hardware-Konfiguration: Hier als Sensor-Typ den Eintrag **Radiation Si22G** (oder ggf. entsprechend) auswählen. Für den zweiten Sensor kann DHT22 stehen bleiben, das ist für uns irrelevant.
- Position: Hier bitte so genau wie möglich die Koordinaten eingeben. Man kann den rechten Knopf benutzen, um die Koordinaten berechnen lassen. Diese werden benötigt, um den Sensor auf der Karte anzuzeigen.

Nun mit „Save settings“ die Einstellungen beenden. Auf der Übersichts-Seite für diesen Sensor auf *Data* klicken. Hier kann man u.a. die Sensor ID sehen. Immer dran denken: die ID wird für Abfragen auf sensor.community oder multigeiger.citysensor.de zwingend benötigt.

Setup (LoRA)

Der Multigeiger kann mit den folgenden Schritten an TTN („The Things Network“) angebunden werden:

- Anlegen des TTN-Devices in Deinem Profil bei *The Things Network*
- Übernehmen der Parameter in den Multigeiger
- Anmelden bei sensor.community (ehemals luftdaten.info)
- HTTP-Integration

Anlegen eines TTN-Devices

Das Gerät muss bei TTN registriert werden. Dazu muss zuerst (falls noch nicht vorhanden) ein Konto bei TTN angelegt werden.

TTN-Konto anlegen

Bei https://account.thethingsnetwork.org/register einen **USERNAME**, die **EMAIL ADRESSE** und ein PASSWORD eingeben. Dann rechts unten über **Create account** das Konto anlegen. Danach kann man sich mit den neuen Daten an der Console anmelden (https://account.thethingsnetwork.org/users/login).

Applikation anlegen

Nachdem man sich eingelogged hat, über **APPLICATIONS** und **add applications“ eine neue Applikation anlegen. Folgende Felder müssen ausgefüllt werden:

- **Application ID:** Ein beliebiger Name für diese Applikation, die darf es aber in dem Netzwerk noch nicht geben (z.B.: geiger_20200205)
- **Description:** Any description of the application can be entered here.
- **Application EUI:** Bleibt frei, die Nummer wird vom TTN-System erzeugt.
- **Handler registration:** The pre-filled value (ttn-handler-eu) is already correct and remains.

Nun mit **Add application** rechts unten die Applikation hinzufügen.

Gerät anlegen

Zuletzt muss noch das Gerät angelegt werden. Dazu in der Übersicht der Applikationen die gerade neu angelegte Applikation auswählen (klicken). Hier dann im mittleren Bereich bei **DEVICES** das Anlegen eines neuen Geräts über **register device** starten. Folgende Felder müssen ausgefüllt werden:

- **Device ID:** Ein beliebiger Name für das Gerät. Er muss innerhalb der Applikation eindeutig sein (also z.B.: geiger_01) und nur aus Kleinbuchstaben bestehen.
- **Device EUI:** Einmal auf das Symbol ganz links an der Zeile klicken, dann erscheint der Text, dass diese Nummer vom System erzeugt wird. Wir müssen sonst nichts weiter eingeben.
- **App Key:** Keine Eingabe nötig
- **App EUI:** Bleibt so stehen

Nun rechts unten auf **Register** klicken. Nun ist auch das Gerät angelegt.

Übernahme der LoRa-Parameter

- Nach erfolgter Registrierung, können die LoRa-Parameter in das Programm übernommen werden.
- Diese können auf der Konfigurations-Seite des Geigerzählers eingestellt werden (siehe oben).
- Nun auf der Konfigurations-Seite durch die Konfiguration gehen bis die Einstellung der LoRa-Parameter angezeigt wird. Nun die 3 Werte (APPEUI, DEVEUI, APPKEY) aus der TTN-Console (siehe oben) entnehmen und hier dann eintragen. In der TTN-Console auf das Device gehen, das oben angelegt wurde. Hier sind die 3 Werte zu finden. Es müssen die HEX-Werte ohne Leerzeichen eingegeben werden, und zwar so, wie sie in der TTN-Console erscheinen.

Beispiel: In der TTN-Console steht

```
Device EUI 00 D0 C0 00 C3 19 7C E8
```

Dann muss Folgendes eingeben werden:

```
00D0C000C3197CE8
```

Dies gilt genauso ebenso für die **APPEUI** und den **APPKEY**.

Daten loggen zu sensor.community (ehem. luftdaten.info)

Wenn der Multigeiger seine Daten über TTN auch an *sensor.community* weitergeben soll, muss er dort angemeldet werden. Die Anmeldung ist ähnlich wie oben für TTN beschrieben. Hier werden nur die Änderungen erläutert:

- Sensor ID: Die letzten 4 bytes der DEVEUI von links nach rechts eingeben (z. B. wenn die DEVEUI *00 D0 C0 00 C3 19 7C E8* ist, dann *C3197CE8* eingeben), aber in dezimal umgerechnet, nicht in HEX (letztendlich: 3273227496).
- Sensor Board: **TTN** auswählen über die kleinen Pfeile rechts.

HTTP-Integration

Um die Daten über TTN zu **sensor.community** zu bekommen, muss noch die HTTP-Integration bei TTN aktiviert werden. In der TTN-Console **Applications** anklicken und dann die Applikation des Geigerzählers (z. B. *geiger_20200205*) anklicken. Dann rechts oben in der Leiste mit *Overview*, *Devices*, *Payload Formats*, *Integrations*, *Data*, *Settings* das **Integrations** anklicken. Hier dann über **add integration** die **HTTP Integration** auswählen.

Nun die angezeigten Felder ausfüllen:

- **Process ID:** Hier einen beliebigen Namen für diese Integration vergeben **Access Key:** Hier einmal rein klicken und den *default key* auswählen
- **URL:** Hier die URL zum ttn2luft-Programm eingeben: https://ttn2luft.citysensor.de
- **Method:** Wenn dort schon *POST* steht, nicht ändern
- **Authorization:** bleibt leer
- **Custom Header Name:** hier kommt der Text **X-SSID** rein
- **Custom Header value:** Die SSID des Sensors eingeben (also die Nummer, die man bei der Anmeldung bei sensor.community erhalten hat, **NICHT** die Chip-ID).
- Dann rechts unten auf **Add integration** klicken, um die Änderungen zu bestätigen.

Siehe dieses Beispiel, wie das Formular aussehen sollte:

TTN-Payload (Beispiel)

Um in der TTN-Console nicht nur die Datenbytes sondern lesbare Werte zu sehen, kann ein kleines Script als Payload-Decoder eingefügt werden. Auf der TTN-Webseite einloggen, über **Applications** die oben angelegte Application suchen. Hier in der Menü-Zeile den Tab **Payload Formats** auswählen und den folgenden Code in das Feld einfügen. Vorhandener Code wird überschreiben):

```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var decoded = {};
  if(port == 1) {
  decoded.counts = ((bytes[0]*256 + bytes[1]) * 256 + bytes[2]) * 256 + bytes[3];
  decoded.sample_time = (bytes[4] * 256 + bytes[5]) * 256 + bytes[6];
  decoded.tube = bytes[9];
  var minor = (bytes[7]&0xF)+(bytes[8]>>4) ;
  decoded.sw_version="" + (bytes[7]>>4) + "." + minor + "." + (bytes[8]&0xF);
  }
  if (port === 2) {
    decoded.temp = ((bytes[0] * 256 + bytes[1]) / 10) + "°C";
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    decoded.humi = bytes[2] / 2 + "%";
    decoded.press = ((bytes[3] * 256 + bytes[4]) / 10) + "hPa";
  }
  return decoded;
}
```

Benutzung

OLED-Display

Oberste Zeile

Links: Zeit seit Einschalten (nicht auf kleinen Displays angezeigt).

Rechts: Gesamte mittlere Strahlung seit Einschalten.

Mittlerer Bereich

Aktueller CPM (Zählungen pro Minute) angezeigt in ziemlich großer Schrift

Untere Zeile

Das ist eine Statusanzeige mit 8 Positionen, numeriert von 0..7:

Faustregeln:

- `.` bedeutet meist „aus“ oder „nicht in Gebrauch“.
- eine *Ziffer* (`0` .. `7`) im Status-Display bedeutet, dass etwas falsch ist.

Positionen:

- 0: WiFi
  - `A`: AccessPoint aktiv
  - `w`: WiFi-Client versucht, sich zu verbinden
  - `W`: WiFi-Client ist verbunden
  - `0`: ein Fehler ist aufgetreten
- 1: sensor.community-Übertragung
  - `.`: aus (nicht konfiguriert / nicht aktiviert)
  - `?`: Initialisierung (aktiviert, vor der 1. Übertragung)
  - `S`: Senden
  - `s`: Leerlauf (wird nach erfolgreichem Senden angezeigt)
  - `1`: Senden fehlgeschlagen (nach Sendeversuch angezeigt)
- 2: madavi-Übertragung
  - `.`: aus (nicht konfiguriert / nicht aktiviert)
  - `?`: Initialisierung (aktiviert, vor der 1. Übertragung)

  - `M`: Senden
  - `m`: Leerlauf (wird nach erfolgreichem Senden angezeigt)
  - `2`: Senden fehlgeschlagen (nach Sendeversuch angezeigt)
- 3: TTN („The Things Network“)
  - `.`: aus (nicht konfiguriert, nicht aktiviert, keine LoRa Hardware)
  - `?`: Initialisierung (aktiviert, vor der 1. Übertragung)
  - `T`: Senden
  - `t`: Leerlauf (wird nach erfolgreichem Senden angezeigt)
  - `3`: Senden fehlgeschlagen (nach Sendeversuch angezeigt)
- 4: BLE (Bluetooth® Low Energy)
  - `.`: aus (nicht aktiviert)
  - `?`: Initialisierung (aktiviert, vor Start des BLE-Dienstes)
  - `B`: verbunden und sendet Benachrichtigungen, wenn es von einem verbundenen Gerät angefordert wird
  - `b`: verbindbar (sichtbar und Bereit zur Verbindung)
  - `4`: BLE-Fehler
- 5: ungenutzt
- 6: ungenutzt
- 7: Hochspannungskondensator laden
  - `H`: OK
  - `7`: Fehler beim Laden des Hochspannungskondensators

ESP32-Taster

Das Multigeiger-ESP32-Microcontroller-Board hat 2 Taster:

- RST: Reset (startet Gerät neu)
- PRG: Program (PRG drücken und halten, dann kurz RST drücken, Firmware hochladen)

Oft ist es nicht notwendig, PRG zum Flashen von Firmware zu benutzen, weil es auch „einfach so“ funktioniert, wenn man die richtigen Werkzeuge nutzt.

DIP-Schalter

Optional kann der Multigeiger auch mit einem 4-poligen DIP-Schalter unterhalb der EPS32-Microcontroller-Platine ausgerüstet werden.

Die Firmware benutzt diese 4 Schalter so wie im „Switches“-Datentyp in „switches.h“ definiert:

- SW0: Lautsprecher an
- SW1: Display an
- SW2: LED an
- SW3: BLE an

BLE - Bluetooth® Low Energy

BLE-Herzfrequenz-Dienst

Der MultiGeiger stellt einen Bluetooth® Low Energy (BLE) Dienst zur Verfügung, um die Geiger-Müller-Zählrate als einen Herzfrequenz Dienst bereitzustellen GATT Heart Rate Service (Service UUID 0x180D). Die folgenden Merkmale werden genutzt:

- 0x2A37 (‚Merkmal Herzfrequenz-Messung'):
  - Das erste Byte ist eine Sammlung von Status-Flags nach Dienststandard
  - ‚Herzfrequenz-Messung' als 16 Bit Wert (little endian), entspricht dem Geiger-Müller-Zählwert pro Minute (CPM)
  - ‚Energieverbrauch' als 16 Bit Wert (little endian), repräsentiert einen rollierenden Paket-Zähler
- 0x2A38 (‚Merkmal Herzfrequenz-Sensorposition')
  - ‚Sensorposition' als 8 Bit Wert, der TUBE-TYPE entspricht, ermöglicht die Umrechnung von CPM zu Strahlungswert
- 0x2A39 (‚Merkmal Herzfrequenz-Kontrollpunkt')
  - Beschreibbares Merkmal, benötigt nach Dienst Standard, um den Energieverbrauch auf 0 zu setzen. Schreiben von 0x01 setzt den rollierenden Paketzähler auf 0 zurück.

Test BLE

Jede brauchbare Bluetooth®-Scanning-App sollte zu dem Gerät verbinden und die Werte aus den Benachrichtigungen anzeigen können.

Die MultiGeiger companion app wurde speziell für den MultiGeiger gemacht.

Es gibt auch einige generische Apps:

- nordic® nRF Connect app
- HRM heart rate monitor app

Jede andere Herz-Überwachungs-App / jedes Herz-Überwachungs-Gerät sollte sich auch mit dem MultiGeiger verbinden können.

- Abbildung 1: Verbinden Sie sich mit dem MultiGeiger mit dem Namen ‚ESP32-<id>', wobei <id> die gleiche ID (7 bis 8 Ziffern) sein sollte, wie beim Access Point, um das Gerät initial einzurichten.
- Abbildung 2: Finden Sie den richtigen Dienst (UUID 0x180D) und, falls erforderlich, laden / öffnen Sie die Einstellungen.
- Abbildung 3: Suchen Sie das Merkmal für die Herzfrequenzmessung (UUID 0x2A37) und klicken Sie auf das Symbol, um die Benachrichtigungen zu starten.

Bitte beachten Sie beim Testen, dass nur alle ~10s ein Update-Paket gesendet wird (zusammen mit einem Display-Update, falls aktiviert). Wenn eine WiFi-Übertragung Display- und BLE-Updates blockiert, kann das Intervall sogar noch länger sein. Es kann also eine Weile dauern, bis eine Reaktion mit aktuellen Daten in der App zu sehen ist.

Abb. 1

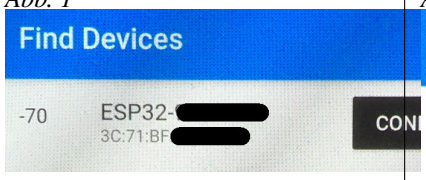


Abb. 2

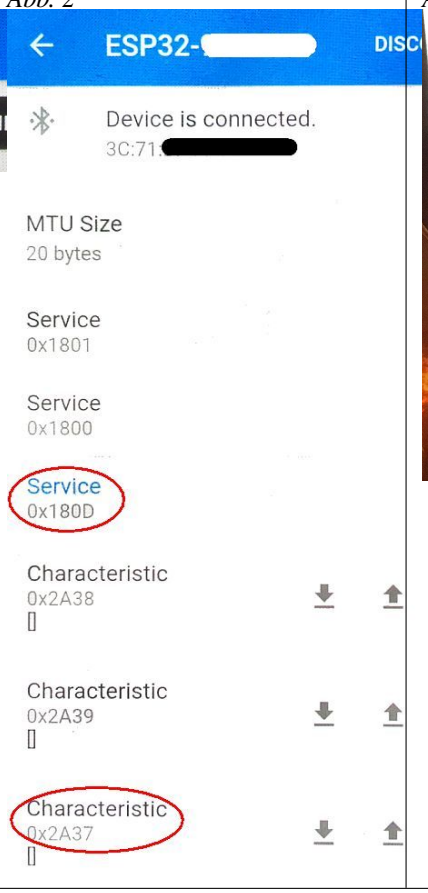
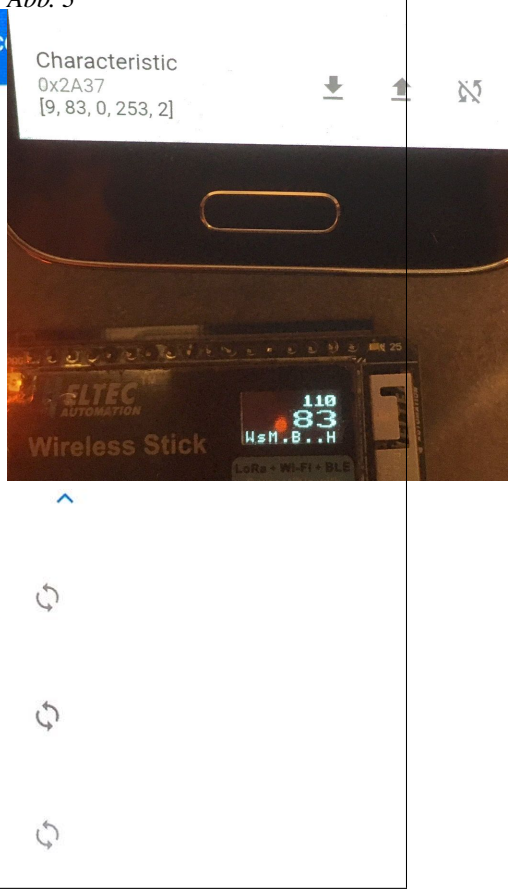


Abb. 3



Aufstellung

Bei der Aufstellung der Multigeigermesstation sind einige Punkte zu beachten.

Natürlich wird es, bei uns Privatleuten, selten einen 100% perfekten Aufstellort geben. Dennoch sollten wir uns bewusst sein welche Parameter negative Einflüsse haben können.

Anforderungen an den Aufstellort

Wahl des Aufstellungsorts

Was wir mit unserem Gerät nachweisen wollen sind radioaktive Stoffe (Stäube), die man auch „Fallout“ nennt. Sie entstehen z. B. bei Atombombenexplosionen und Reaktorkatastrophen.

In sehr schwacher Form entstehen sie natürlicherweise auch in der Atmosphäre auch durch den Zerfall des radioaktiven Edelgases Radon. Auch dieser „Radon-Fallout“ erzeugt bei unseren Geräten einen kleinen, aber messbaren, Ausschlag, der sich in etwa mit einer Halbwertszeit von 45 Minuten wieder abbaut.

Anhand dieser „Radon-Peaks“ können wir sehen ob unser Gerät gut aufgestellt ist und auch für „echten“ Fallout empfindlich ist.

Abregenfläche

Wir brauchen in direkter Umgebung der Messstation eine größere Fläche auf die es regnen kann. Diese Fläche sollte auch die Möglichkeit haben radioaktive Partikel festzuhalten.

Rasen oder Wiese ist hierfür perfekt.

Eine Asphaltfläche dagegen kann dies nicht, da die Partikel direkt nach dem Abregnen sofort abgespült werden.

Windrichtung

Das Gelände sollte von der Hauptwindrichtung her unverbaut sein und auch Büsche oder Bäume sollten sich dort keine befinden.

Sonnenlicht

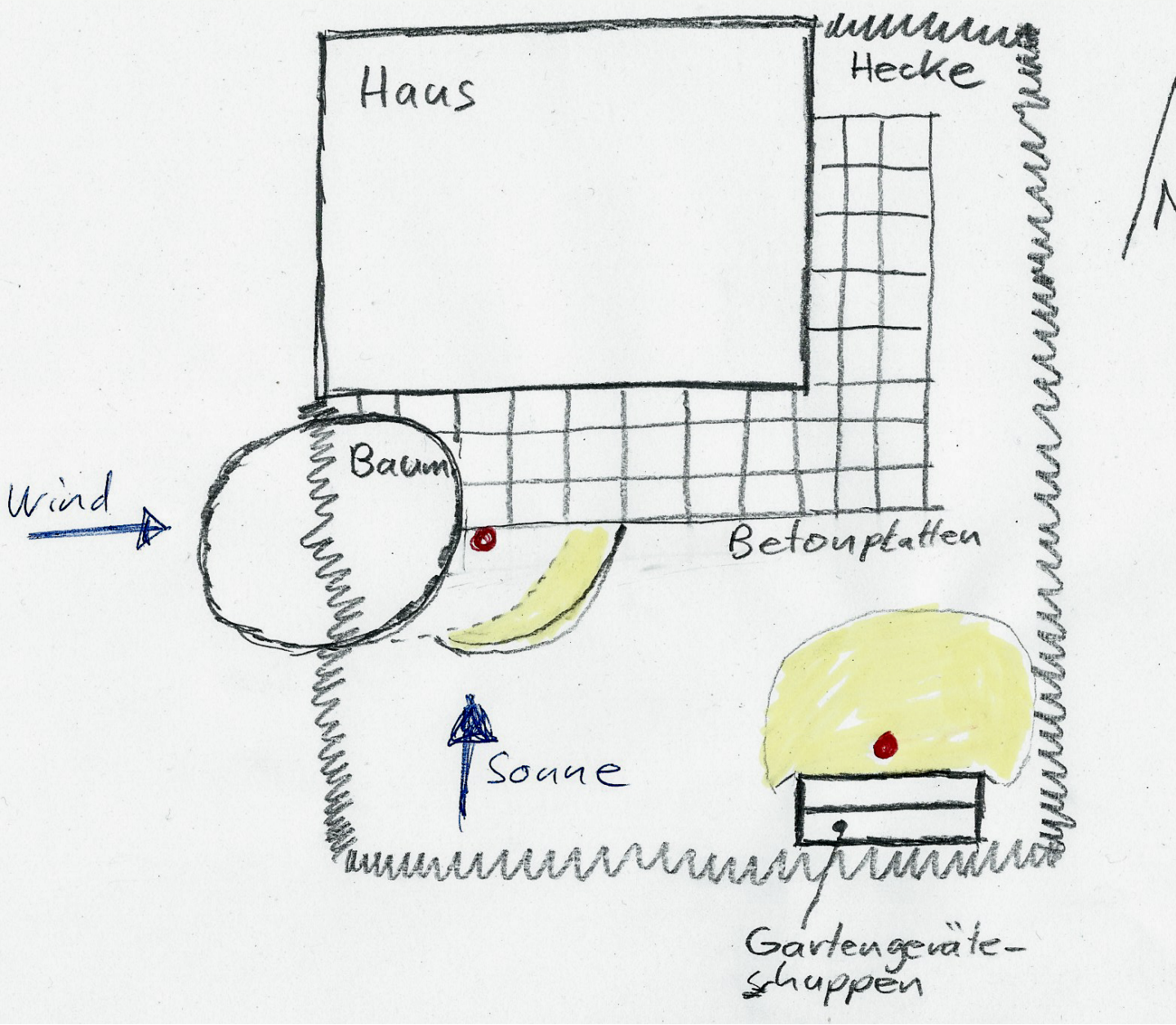
Sowohl das 5 V-Kabel, als auch das Gerät selbst, sollten sich möglichst die meiste Zeit des Tages im Schatten befinden.

Direktes Sonnenlicht kann zweierlei negative Auswirkungen haben:

- Hitze: Die Temperatur im Gerät kann in der Sonne auf bis zu 60 °C ansteigen (bei 45 °C im Schatten). Zwar haben unsere Temperaturtests gezeigt dass das Gerät bei 70 °C noch einwandfrei funktioniert. Jedoch ist das Zählrohr nur bis 50 °C spezifiziert.
- UV-Schäden: Direktes Sonnenlicht kann unser Gehäuse im Laufe der Jahre porös werden lassen und dadurch schädigen.

Beispiele

Beispielhafter Vergleich eines guten und eines schlechten Aufstellungsorts:



Vergleich zweier Aufstellorte. Gelb ist die „Nutzbare Abregenfläche“.

Schlechter Aufstellungsort (linker roter Punkt)

Gründe:

- Wiese: Ein Großteil der Fläche um den Geigerzähler besteht aus Waschbetonplatten. Hier kann kein abgeregneter Fallout gespeichert werden.
- Wind: Der Wind kommt hauptsächlich aus Westen und der Baum erschwert das Abladen des Fallouts in der Nähe des Zählrohrs.
- Sonne: Die Sonne kommt hauptsächlich von Süden und bestrahlt den Multigeiger stark.

Guter Aufstellungsort (rechter roter Punkt)

Gründe:

- Wiese: Ein Großteil der Fläche um den Geigerzähler besteht aus Wiese, die abgeregneten Fallout speichern kann.
- Wind: Keine Hindernisse.
- Sonne: Der Geigerzähler wird gut vom Gartengeräteschuppen verschattet.

Weitere Informationen

Beschreibung des BFS-ODL-Netzes:

- Messnetz
- ODL-Sonde

Videointerview über das BFS-ODL-Netz:

- Interview über das BFS-ODL-Netzwerk

Oft gestellte Fragen (FAQ)

Allgemein

Strahlungs-Dosis-Werte, gibt es da praktische Beispiele?

Es gibt dazu ein großartiges XKCD, siehe dort:

https://xkcd.com/radiation/

Ressourcen

Dies ist eine Sammlung von zusätzlichen Ressourcen für MultiGeiger.

Hardware / Einzelteile

Von Multigeiger unterstützte Zählrohre:

- Si22G, SBM-20, SBM-19

MultiGeiger-unterstützte ESP32-Boards:

- Heltec WiFi Kit 32 (WiFi-only, Infos vom Boardhersteller) https://heltec.org/project/wifi-kit-32/
- Heltec Wireless Stick (WiFi+LoRa, Infos vom Board-Hersteller) https://heltec.org/project/wireless-stick/

Software

- Windows-Treiber für den USB -> UART-Chip CP2102: http://esp32.net/usb-uart/#SiLabs

Changelog

V1.16.0 2021-08-15

New features:

- local radiation alarm sound, #427

Fixes:

- audio isr fixes, #436
- fix occasional reboots (hopefully), #314

Other changes:

- logging related improvements / code cleanups:
  - emit log header now and then for better readability
  - add UTC timestamp to log output
  - unify radiation and t/h/p log output into single line, use „DATA“ to mark these lines.
- BLE support now implemented via NimBLE-Arduino lib
- upgrade libs to current versions
- use arduino-esp32 1.0.6

V1.15.1 2021-05-13

New features:

- None

Fixes:

- initialize sound and LED to „off“, #398

Other changes:

- explicitly turn ticking off before melody/init
- cosmetic: don’t touch LED in pauses between melody notes
- upgrade to Adafruit BME680 Library >=2.0.0
- add root cert „Amazon CA 1“
- docs:
  - bluetooth: fix app links, fix reST formatting
  - fix typo in sensor.community domain
  - fixed switch numbering, #349
  - add supported counter tubes
  - create multigeiger-bill-of-material.txt
  - developer docs: bump master branch version to -dev after release
  - developer docs: update pulling translations from transifex
  - update translations

V1.15.0 2021-03-21

New features:

- add bluetooth (BLE) support, #78

Fixes:

- improve LoRaWAN stability (work around LMIC bug #677, add LMIC polling from loop()), #373
- do async NTP/clock setup, #316
- speaker: init „duty_mode“ member in MCPWM config
- avoid using IotWebConf 3.0.0 for now, #357, PR #370

Other changes:

- patch: restore partition scheme menu for arduino-esp32 1.0.5
- move CI from travis CI to github workflow
- start screen cleanups, #335
- code / naming style fixes
- remove dates in file names, commit relevant versions to git
- add drill files, #354
- docs:
  - use transifex / sphinx / readthedocs.org for translatons (en/de for now)
  - document docs/translation workflow in development docs
  - added assembly and deployment guide
  - document esp32 board buttons, #129
  - document dip switch usage, #128
  - move README-{de,en}.* contents into the .rst docs
  - BLE usage documentation update with some images, #338
  - added links to map, ecocurious, assembly room
  - markup, rendering, spelling fixes, cleanups
  - fix unclear version / date in Aufbauanleitung, #110
  - moved links to docs -> resources, #223
  - add xkcd about radiation doses to FAQ, #310

V1.14.0 2020-05-16

New features:

- implement status line on OLED display (see docs), #257
- also support BME680 sensor for temperature, humidity, pressure
- display time up to 60s / 60m / 24h / 99d, then roll over
- speaker/LED: timer-driven sequencer, hw PWM sound, #35

- TLS support
  - add clock module, use NTP to set the clock
  - use persistent per-server HTTPClient instances
  - use connection: keep-alive for web requests
  - add https capability (can be used for sending data)
  - note: transmission to sensor.community and madavi is still using http!

Fixes:

- fixed GM pulse debouncing, #248
- pulse counting: deal with microseconds uint32 overflow, #273
- check WiFi status before trying to transmit
- fix race condition, #286

Other changes:

- dip switches: only read once at boot time, #207
- new font (u8x8 uses 8px width anyway)
- slow down main loop
- toilet -> custom server, add comments about toilet usage, #214
- refactor/simplify pulse counting ISR, bookkeeping in main loop, #220
- refactor big main loop into smaller functions with local bookkeeping.
- misc. other code cleanups
- loraWan: removed unused/not needed code, #212, #234
- removed meeting notes, #294
- docs:
  - README improvements (board name, flash size, partition scheme, passwords, LoRa)
  - update development/release docs (create/test binaries, IDE settings, . . .)

V1.13.0 2020-04-14

- auto-detect hardware (STICK vs. WIFI) by hardware pin
- use config web page for more values (userdefines.h has the defaults), #140
- try both adresses of BME280
- LoRa payload changes, e.g. to fulfill ‚TTN Fair Access Policy‘
- send additional data to servers
- send to MADAVI in one single request both geiger and thp data
- new logging with DEFAULT_LOG_LEVEL configuration
- integrated travis-ci:
  - for compile checks (platformio, wifi and stick build)
  - for style checks (using the „astyle“ CPP checker)

- source: modularization, cleanups, less globals, ... (quite huge internal changes, please help testing!)
- building:
  - platformio-based build: suppress lmic_project_config.h usage
  - arduino-ide-based build: you still need to edit that file
- use bump2version tool for project version bumps, #169
- docs:
  - added upgrade hints for 1.13 in README on github
  - https://multigeiger.readthedocs.io/ == the beginning of new (sphinx / reST-markup based) online docs, #163
  - add a basic, short README in English (also for online docs)
  - include infos about project name, #121
  - moved changelog.md to docs/source/changes.rst
  - updated/fixed development docs, #46
  - update docs about new 5V power supply / cabling, #122
  - description of LoRa Payload updated
  - other docs improvements / fixes

V1.12.0 2020-01-18

- simple OTA (Over-The-Air) updates via web browser based upload, #120
- use less charge pulses in loop() for timing, more in setup() for initial charging, #134
- output error msg on Serial if HV charging fails
- tag log output with „GEIGER: „, #85
- add TUBE_UNKNOWN 0 to have a specific value for experimenting
- adapted platformio.ini to pull all dependencies
- send CR and LF on serial
- changed default tube from sbm-20 to si22g
- semantic versioning, version numbers now like x.y.z
- changed building of revString and lora_version
- docs updated / improved
- explain SBM-19/SBM-20 conversion factor
- removed IotWebconf bundled&patched code, used as a lib now.

V1.11.1 2019-12-16 rxf

- change luftdaten.info to sensor.community

V1.11.0 2019-12-16 rxf

- defaults in userdefines-example.h changed
- Software version for LoRa now 2 Bytes
- Display start screen for Wireless stick fixed
- changed to semantic versioning

V1.10 2019-12-13

- conversion factor for Si22G tube fixed
- char variables changed to int
- isr routines shielded with portMUX
- debug serial out formatting improved
- sequence of counting and dispaying and hv charging improved
- speaker and led tick fixed
- many calls to millis() consolidated

V1.9 2019-11-12

- structure for different counter tubes
- LoRa payload changed again
- hv pulse every second
- calculate and display cpm value every 10 seconds
- fixed div by 0 if there's no tube
- Readme corrected

V1.8 2019-11-04

- indentation/spacing, refactor OLED functions, fix conversion factor
- MEASUREMENT_INTERVAL 150sec
- changed LoRa payload

V1.7 2019-10-21

- PINs rearranged, so we can use new Wifi-Kit-32 and WiFi Stick Light
- Hardware-Layout V1.4 and up
- use switch for speaker tick and display off

V1.61 2019-09-30

- default measuring interval is now 2.5min

V1.6 2019-09-13

- some rearrangement of files
- userdefine.h for user changable #defines
- test with dip-switch (needs pullup resistors!)
- Hardware layout V1.3 and lower - OLD Wifi-Kit-32!

V1.5 2019-09-11

- added BME280 (uses same I2C as display)
- Support for display on Wireless Stick
- For LoRa-Devices added LoRa functionality

V1.4 2019-09-03

- default configuration with measurement interval of 10min

V1.3 2019-09-03

- building of ESP-ID out of MAC address is now identical to ‚Feinstaubsensor‘

V1.2 2019-09-02

- sending to madavi corrected

V1.1 2019-09-01

- Library IoTWebConfig changed -> function ‚setThingName‘ added
- Move this (IoTWebConfig) library to source path
- building the SSID from the MAC corrected: first 3 Bytes of MAC build SSID
- LoRa autodetection removed

V1.0 2019-08-19 rxf

- added detection of LoRa device
- WiFiManager to enter WLAN data and other configs
- send to luftdaten.info every 2.5 min

V0.3 2019-05-12 jb

- added bug fix for the „Double-Trigger-Problem“. This was caused by the rising edge falsly triggering an other pulse recording. The Problem is that there is no Schmitt-Trigger available in the controller.
- simplified serial printing modes
- made seconds in Display as inverse to be able to separate it from minutes
- cleaned up the code
- Fixed overflow bug in Minute-Count+

V0.2 2019-04-26 jb

- added 1 Minute RS232 (USB) logging mode

V0.1 2019-03-25 jb

- first version for ESP32 board

Entwicklung

In diesem Kapitel werden Sie in die Entwicklung von MultiGeiger eingeführt.

MultiGeiger ist in C geschrieben und benutzt Arduino-ESP32.

Beiträge

. . . sind willkommen!

Einige Hinweise für Beitragende:

- diskutieren Sie über Änderungen auf dem Github-Issue-Tracker
- machen Sie Ihre PRs gegen den `master`-Branch
- machen Sie saubere Changesets:
  - konzentrieren Sie sich auf ein Thema, ändern Sie nichts anderes.
  - machen Sie keine Stiländerungen gemischt mit funktionalen Änderungen.
  - lassen Sie den automatischen Code-Formatierer laufen, bevor Sie committen
  - versuchen Sie, Refactoring gemischt mit funktionalen Änderungen zu vermeiden.
  - wenn Sie nach dem Commit/Push etwas korrigieren müssen:
    - wenn es laufende Überprüfungen gibt: machen Sie einen Fixup-Commit, den Sie später mit dem schlechten Commit zusammenführen können.
    - Wenn es keine laufenden Überprüfungen gibt oder Sie den schlechten Commit noch nicht gepusht haben: Bearbeiten Sie den Commit, um Ihre Korrektur einzuschließen oder führen Sie ihn mit dem Fixup-Commit vor dem Pushen zusammen.
  - schreiben Sie einen schönen, klaren, tippfehlerfreien Commit-Kommentar
  - wenn Sie ein Problem behoben haben, verweisen Sie darauf in Ihrem Commit-Kommentar

- Machen Sie einen Pull-Request auf Github und überprüfen Sie auf der PR-Seite, was das CI-System über den Code in Ihrem PR sagt
- Warten Sie auf die Überprüfung durch andere Entwickler

Eine Entwicklungsumgebung aufbauen

TODO

Automatischer Code-Formatierer

Wir verwenden astyle für automatische Code-Formatierung / Formatierungsprüfungen.

Führen Sie es wie folgt aus:

```
astyle --options=.astylerc 'multigeiger/\*'
```

Dokumentation

Erstellen der Dokumentation mit Sphinx

Die Dokumentation wird in Englisch geschrieben und dann von dieser Quelle aus in andere Sprachen übersetzt (anfangs Deutsch).

Die Dokumentation (im reStructuredText-Format, .rst) befindet sich in `docs/source/`, `index.rst` ist der Startpunkt dort.

Um die Dokumentation zu erstellen, müssen Sie Sphinx installiert haben und dies ausführen:

```
cd docs/
make html
```

Rufen Sie dann einen Webbrowser für `docs/build/html/index.html` auf.

Die Website wird von ReadTheDocs automatisch über GitHub-Web-Hooks auf dem Haupt-Repository aktualisiert.

Nach Änderungen an den (englischen) Master-Docs müssen die Übersetzungs-Masterdateien (`*.pot`) aktualisiert werden (Hinzufügen/Entfernen/Aktualisieren von Strings darin):

```
cd docs/build/gettext
sphinx-build -b gettext ../../source .
```

Dann müssen diese Änderungen nach transifex gepusht werden, damit die Übersetzer bequem im Web übersetzen können:

Die Übersetzung ist auf [transifex](https://www.transifex.com/thomaswaldmann/multigeiger/) organisiert, Du brauchst dort ein Konto oder musst zumindest Dich dort einloggen und einen „Team beitreten“-Antrag absetzen. Dann die fehlenden Teile übersetzen und den Entwicklern Bescheid sagen (z. B. über den Issue Tracker).

```
tx push --source
```

Später, nachdem die Übersetzer ihren Teil erledigt haben, müssen die aktualisierten Übersetzungen von transifex gepulled werden:

```
tx pull --all --force
```

Jetzt haben wir Änderungen in unserem Git-Workdir und müssen sie committen:

```
git add locales/
git commit -m "updated translations"
git push
```

Dies wird einen Build der Dokumentation und ihrer Übersetzung(en) auf readthedocs.io auslösen.

Geräte flashen / Binärdateien erzeugen

Arduino IDE:

- Führen Sie ein Git Checkout der gewünschten Version durch, z.B. `git checkout V1.13.0`
- verwenden Sie die Standard userdefines.h (verfügbar als userdefines-example.h)
- IDE-Einstellungen:
  - Gerät: Heltec WiFi Stick (diesen immer verwenden, auch wenn Sie ein WiFi Kit 32 haben)
  - Flash-Größe: 4MB (32Mb)
  - Partitionsschema: minimal SPIFFS (große APPS mit OTA) - das passt auf 4MB-Geräte.
- `Arduino IDE -> Sketch -> Hochladen`

  Dies ist zum Testen, ob der kompilierte Code nach dem USB-Flashen auf Ihr Gerät tatsächlich funktioniert.
- `Arduino IDE -> Sketch -> Kompilierte Binärdatei exportieren`

  Dies erzeugt eine .bin-Datei für die OTA-Aktualisierung. Testen Sie, ob die OTA-Aktualisierung mit dieser Datei funktioniert.

Erstellen eines neuen Releases

Checkliste:

- Sicherstellen, dass alle Probleme für diesen Meilenstein geschlossen oder zum nächsten Meilenstein verschoben worden sind
- Prüfen Sie, ob es noch ausstehende Korrekturen für schwerwiegende Probleme gibt
- prüfen, ob ein CA-Zertifikat (siehe `ca_certs.h`) bald abläuft und ob wir bereits deren nächstes gültiges Zertifikat hinzufügen können.
- alle einfach zu lösenden Probleme im Issue Tracker finden und beheben
- Release-Meilenstein auf Github schließen
- Aktualisieren von `docs/source/changes.rst`, basierend auf `git log $PREVIOUS_RELEASE..`
- `bump2version --new-version 1.23.0 release` - dies wird:
  - die Versionen überall aktualisieren
  - einen Git-Tag automatisch erstellen
  - einen Git-Commit automatisch erstellen
- Überprüfen Sie das automatisch erzeugte Changeset

- einen Github-Release für dieses Tag erstellen:
  - Erstellen Sie ein Binary (siehe oben) und hängen Sie es an den Github-Release an
  - Fügen Sie einen Link zum relevanten `changes.rst`-Abschnitt zum Github-Release hinzu
- `bump2version --no-tag --current-version 1.23.0 minor` - dies wird:
  - die Versionen überall aktualisieren (jetzt auf: 1.24.0-dev)
  - leider nicht ganz korrekt die changes.rst aktualisieren, so dass manuelle Korrekturen danach notwendig sind
  - nach der Korrektur: git commit –amend

Autoren

- Juergen Boehringer (see www.boehri.de)
- Reinhard X. Fuerst (see feinstaub.rexfue.de)
- Thomas Waldmann <twaldmann@thinkmo.de>
- und andere

Lizenz

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                       TERMS AND CONDITIONS

  0. Definitions.

  "This License" refers to version 3 of the GNU General Public License.

  "Copyright" also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.

  "The Program" refers to any copyrightable work licensed under this
License.  Each licensee is addressed as "you".  "Licensees" and
"recipients" may be individuals or organizations.

  To "modify" a work means to copy from or adapt all or part of the work
in a fashion requiring copyright permission, other than the making of an
exact copy.  The resulting work is called a "modified version" of the
earlier work or a work "based on" the earlier work.

  A "covered work" means either the unmodified Program or a work based
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
on the Program.

  To "propagate" a work means to do anything with it that, without
permission, would make you directly or secondarily liable for
infringement under applicable copyright law, except executing it on a
computer or modifying a private copy.  Propagation includes copying,
distribution (with or without modification), making available to the
public, and in some countries other activities as well.

  To "convey" a work means any kind of propagation that enables other
parties to make or receive copies.  Mere interaction with a user through
a computer network, with no transfer of a copy, is not conveying.

  An interactive user interface displays "Appropriate Legal Notices"
to the extent that it includes a convenient and prominently visible
feature that (1) displays an appropriate copyright notice, and (2)
tells the user that there is no warranty for the work (except to the
extent that warranties are provided), that licensees may convey the
work under this License, and how to view a copy of this License.  If
the interface presents a list of user commands or options, such as a
menu, a prominent item in the list meets this criterion.

  1. Source Code.

  The "source code" for a work means the preferred form of the work
for making modifications to it.  "Object code" means any non-source
form of a work.

  A "Standard Interface" means an interface that either is an official
standard defined by a recognized standards body, or, in the case of
interfaces specified for a particular programming language, one that
is widely used among developers working in that language.

  The "System Libraries" of an executable work include anything, other
than the work as a whole, that (a) is included in the normal form of
packaging a Major Component, but which is not part of that Major
Component, and (b) serves only to enable use of the work with that
Major Component, or to implement a Standard Interface for which an
implementation is available to the public in source code form.  A
"Major Component", in this context, means a major essential component
(kernel, window system, and so on) of the specific operating system
(if any) on which the executable work runs, or a compiler used to
produce the work, or an object code interpreter used to run it.

  The "Corresponding Source" for a work in object code form means all
the source code needed to generate, install, and (for an executable
work) run the object code and to modify the work, including scripts to
control those activities.  However, it does not include the work's
System Libraries, or general-purpose tools or generally available free
programs which are used unmodified in performing those activities but
which are not part of the work.  For example, Corresponding Source
includes interface definition files associated with source files for
the work, and the source code for shared libraries and dynamically
linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those
subprograms and other parts of the work.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
  The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding
Source.

  The Corresponding Source for a work in source code form is that
same work.

  2. Basic Permissions.

  All rights granted under this License are granted for the term of
copyright on the Program, and are irrevocable provided the stated
conditions are met.  This License explicitly affirms your unlimited
permission to run the unmodified Program.  The output from running a
covered work is covered by this License only if the output, given its
content, constitutes a covered work.  This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

  You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains
in force.  You may convey covered works to others for the sole purpose
of having them make modifications exclusively for you, or provide you
with facilities for running those works, provided that you comply with
the terms of this License in conveying all material for which you do
not control copyright.  Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction
and control, on terms that prohibit them from making any copies of
your copyrighted material outside their relationship with you.

  Conveying under any other circumstances is permitted solely under
the conditions stated below.  Sublicensing is not allowed; section 10
makes it unnecessary.

  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

  No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article
11 of the WIPO copyright treaty adopted on 20 December 1996, or
similar laws prohibiting or restricting circumvention of such
measures.

  When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention
is effected by exercising rights under this License with respect to
the covered work, and you disclaim any intention to limit operation or
modification of the work as a means of enforcing, against the work's
users, your or third parties' legal rights to forbid circumvention of
technological measures.

  4. Conveying Verbatim Copies.

  You may convey verbatim copies of the Program's source code as you
receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice;
keep intact all notices stating that this License and any
non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all
recipients a copy of this License along with the Program.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
  You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

  5. Conveying Modified Source Versions.

  You may convey a work based on the Program, or the modifications to
produce it from the Program, in the form of source code under the
terms of section 4, provided that you also meet all of these conditions:

    a) The work must carry prominent notices stating that you modified
    it, and giving a relevant date.

    b) The work must carry prominent notices stating that it is
    released under this License and any conditions added under section
    7.  This requirement modifies the requirement in section 4 to
    "keep intact all notices".

    c) You must license the entire work, as a whole, under this
    License to anyone who comes into possession of a copy.  This
    License will therefore apply, along with any applicable section 7
    additional terms, to the whole of the work, and all its parts,
    regardless of how they are packaged.  This License gives no
    permission to license the work in any other way, but it does not
    invalidate such permission if you have separately received it.

    d) If the work has interactive user interfaces, each must display
    Appropriate Legal Notices; however, if the Program has interactive
    interfaces that do not display Appropriate Legal Notices, your
    work need not make them do so.

  A compilation of a covered work with other separate and independent
works, which are not by their nature extensions of the covered work,
and which are not combined with it such as to form a larger program,
in or on a volume of a storage or distribution medium, is called an
"aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users
beyond what the individual works permit.  Inclusion of a covered work
in an aggregate does not cause this License to apply to the other
parts of the aggregate.

  6. Conveying Non-Source Forms.

  You may convey a covered work in object code form under the terms
of sections 4 and 5, provided that you also convey the
machine-readable Corresponding Source under the terms of this License,
in one of these ways:

    a) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by the
    Corresponding Source fixed on a durable physical medium
    customarily used for software interchange.

    b) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by a
    written offer, valid for at least three years and valid for as
    long as you offer spare parts or customer support for that product
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    model, to give anyone who possesses the object code either (1) a
    copy of the Corresponding Source for all the software in the
    product that is covered by this License, on a durable physical
    medium customarily used for software interchange, for a price no
    more than your reasonable cost of physically performing this
    conveying of source, or (2) access to copy the
    Corresponding Source from a network server at no charge.

    c) Convey individual copies of the object code with a copy of the
    written offer to provide the Corresponding Source.  This
    alternative is allowed only occasionally and noncommercially, and
    only if you received the object code with such an offer, in accord
    with subsection 6b.

    d) Convey the object code by offering access from a designated
    place (gratis or for a charge), and offer equivalent access to the
    Corresponding Source in the same way through the same place at no
    further charge.  You need not require recipients to copy the
    Corresponding Source along with the object code.  If the place to
    copy the object code is a network server, the Corresponding Source
    may be on a different server (operated by you or a third party)
    that supports equivalent copying facilities, provided you maintain
    clear directions next to the object code saying where to find the
    Corresponding Source.  Regardless of what server hosts the
    Corresponding Source, you remain obligated to ensure that it is
    available for as long as needed to satisfy these requirements.

    e) Convey the object code using peer-to-peer transmission, provided
    you inform other peers where the object code and Corresponding
    Source of the work are being offered to the general public at no
    charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  "Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  "Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
    reasonable ways as different from the original version; or
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    d) Limiting the use for publicity purposes of names of licensors or
    authors of the material; or

    e) Declining to grant rights under trademark law for use of some
    trade names, trademarks, or service marks; or

    f) Requiring indemnification of licensors and authors of that
    material by anyone who conveys the material (or modified versions of
    it) with contractual assumptions of liability to the recipient, for
    any liability that these contractual assumptions directly impose on
    those licensors and authors.

  All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10.  If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further
restriction, you may remove that term.  If a license document contains
a further restriction but permits relicensing or conveying under this
License, you may add to a covered work material governed by the terms
of that license document, provided that the further restriction does
not survive such relicensing or conveying.

  If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the
additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

  Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions;
the above requirements apply either way.

  8. Termination.

  You may not propagate or modify a covered work except as expressly
provided under this License.  Any attempt otherwise to propagate or
modify it is void, and will automatically terminate your rights under
this License (including any patent licenses granted under the third
paragraph of section 11).

  However, if you cease all violation of this License, then your
license from a particular copyright holder is reinstated (a)
provisionally, unless and until the copyright holder explicitly and
finally terminates your license, and (b) permanently, if the copyright
holder fails to notify you of the violation by some reasonable means
prior to 60 days after the cessation.

  Moreover, your license from a particular copyright holder is
reinstated permanently if the copyright holder notifies you of the
violation by some reasonable means, this is the first time you have
received notice of violation of this License (for any work) from that
copyright holder, and you cure the violation prior to 30 days after
your receipt of the notice.

  Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under
this License.  If your rights have been terminated and not permanently
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
reinstated, you do not qualify to receive new licenses for the same
material under section 10.

  9. Acceptance Not Required for Having Copies.

  You are not required to accept this License in order to receive or
run a copy of the Program.  Ancillary propagation of a covered work
occurring solely as a consequence of using peer-to-peer transmission
to receive a copy likewise does not require acceptance.  However,
nothing other than this License grants you permission to propagate or
modify any covered work.  These actions infringe copyright if you do
not accept this License.  Therefore, by modifying or propagating a
covered work, you indicate your acceptance of this License to do so.

  10. Automatic Licensing of Downstream Recipients.

  Each time you convey a covered work, the recipient automatically
receives a license from the original licensors, to run, modify and
propagate that work, subject to this License.  You are not responsible
for enforcing compliance by third parties with this License.

  An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered
work results from an entity transaction, each party to that
transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could
give under the previous paragraph, plus a right to possession of the
Corresponding Source of the work from the predecessor in interest, if
the predecessor has it or can get it with reasonable efforts.

  You may not impose any further restrictions on the exercise of the
rights granted or affirmed under this License.  For example, you may
not impose a license fee, royalty, or other charge for exercise of
rights granted under this License, and you may not initiate litigation
(including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for
sale, or importing the Program or any portion of it.

  11. Patents.

  A "contributor" is a copyright holder who authorizes use under this
License of the Program or a work on which the Program is based.  The
work thus licensed is called the contributor's "contributor version".

  A contributor's "essential patent claims" are all patent claims
owned or controlled by the contributor, whether already acquired or
hereafter acquired, that would be infringed by some manner, permitted
by this License, of making, using, or selling its contributor version,
but do not include claims that would be infringed only as a
consequence of further modification of the contributor version.  For
purposes of this definition, "control" includes the right to grant
patent sublicenses in a manner consistent with the requirements of
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
patent license under the contributor's essential patent claims, to
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.

  12. No Surrender of Others' Freedom.

  If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
covered work so as to satisfy simultaneously your obligations under this
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
to collect a royalty for further conveying from those to whom you convey
the Program, the only way you could satisfy both those terms and this
License would be to refrain entirely from conveying the Program.

  13. Use with the GNU Affero General Public License.

  Notwithstanding any other provision of this License, you have
permission to link or combine any covered work with a work licensed
under version 3 of the GNU Affero General Public License into a single
combined work, and to convey the resulting work.  The terms of this
License will continue to apply to the part which is the covered work,
but the special requirements of the GNU Affero General Public License,
section 13, concerning interaction through a network will apply to the
combination as such.

  14. Revised Versions of this License.

  The Free Software Foundation may publish revised and/or new versions of
the GNU General Public License from time to time.  Such new versions will
be similar in spirit to the present version, but may differ in detail to
address new problems or concerns.

  Each version is given a distinguishing version number.  If the
Program specifies that a certain numbered version of the GNU General
Public License "or any later version" applies to it, you have the
option of following the terms and conditions either of that numbered
version or of any later version published by the Free Software
Foundation.  If the Program does not specify a version number of the
GNU General Public License, you may choose any version ever published
by the Free Software Foundation.

  If the Program specifies that a proxy can decide which future
versions of the GNU General Public License can be used, that proxy's
public statement of acceptance of a version permanently authorizes you
to choose that version for the Program.

  Later license versions may give you additional or different
permissions.  However, no additional obligations are imposed on any
author or copyright holder as a result of your choosing to follow a
later version.

  15. Disclaimer of Warranty.

  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  16. Limitation of Liability.

  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```