„Vertrauen ist gut Kontrolle ist besser“: Die solarbetriebene, autonome Messstation für Umwelt-Radioaktivität von 4N-Galaxy

Bernd Laquai
3.1.2015

Abb. 1, Fotomontage: Die autonome Messstation R10 von 4N-Galaxy vor dem KKW Neckarwestheim

„Vertrauen ist gut, Kontrolle ist besser“, das ist ein altes Sprichwort, das seine Gültigkeit vor allem dann hat, wenn es um hohe Risiken geht. Im versicherungstechnischen Sinne versteht man unter einem Risiko das Produkt der Eintrittswahrscheinlichkeit eines zufälligen Ereignisses und der Schwere des Schadens. Ein hohes Risiko besteht also auch dann, wenn die Eintrittswahrscheinlichkeit zwar sehr gering ist, der Schaden aber umso höher ist. Im Gesundheitswesen schlagen entsprechende Versicherungen ihren Kunden Vorsorgeuntersuchungen zur Krebsfrüherkennung vor und führen eine intensive Aufklärung durch um solche Risiken zu minimieren. Eine solche regelmäßige Kontrolle kann, wenn man schnell genug reagiert, in vielen Fällen helfen Schlimmeres zu vermeiden.

Ob eine vergleichbare Betrachtung von Risiken auch auf kerntechnische Anlagen sinnvoll anwendbar ist, ist bekanntlich sehr umstritten. Aber zumindest geht es um kleine Eintrittswahrscheinlichkeiten von Schadensfällen mit unüberschaubar großen Schäden, die nicht nur die Gesundheit ganz fundamental betreffen, sondern auch den Verlust der Heimat eines großen Teils der Bevölkerung. So gesehen haben laufende Kontrollen solcher Anlagen schon eine gewisse Bedeutung. Mit etwas Kenntnis kapitalistisch ausgeprägter Wirtschaftssysteme wird schnell klar, dass die reine Selbst-Kontrolle der Anlagenbetreiber alleine äußerst fahrlässig wäre, gerade wenn es um die Gesundheit der Bevölkerung in den typischerweise großen, von Schadensfällen betroffenen Gebieten geht. Aus diesem Grund macht es völlig Sinn, dass die Regierung Behörden beauftragt, gleichermaßen Kontrollen durchzuführen. Leider aber legen die bisher aufgetretenen Schadensfälle sehr deutlich offen,

wie problematisch selbst die Kontrolle der kerntechnischen Anlagen durch die Behörden ist. In Fukushima in Japan, einem Hoch-Technologie-Land der vordersten Front mit demokratischer Regierung, konnte man jedenfalls sehr deutlich erkennen, wie gleichzeitig mit dem Betreiber auch die Behörden völlig versagten. Das betraf einerseits die Erfassung der Messwerte für die eingetretene radioaktive Kontamination der Umwelt, die plötzlich nicht mehr verfügbar waren, wie auch die richtigen Anweisungen an die Bevölkerung und deren Unterstützung und Versorgung im Laufe der Katastrophe. So wurde z.B. eine kreisrunde 20km Evakuierungszone angeordnet, obwohl später sichtbar wurde, dass die Kontaminationsfahne sehr langgestreckt war und eine kritische Ausdehnung von deutlich mehr als 50km erreicht hatte (siehe auch Abb. 7)

Es stellt sich also die Frage, ob nicht genau wie bei gesundheitlichen Vorsorge-Kontrollen, auch eine generelle Mitwirkungsmöglichkeit der Bevölkerung bei der Kontrolle der Umwelt-Radioaktivität nicht nur der Nähe von kerntechnischen Anlagen vorgesehen werden sollte um die Aufgabe der Messwerterfassung auch in Katastrophenfällen bewältigen zu können, denn schließlich geht es wie bei der Krebs-Vorsorge um fundamentale Gesundheitsinteressen eines jeden Einzelnen und um sehr hohe individuelle Risiken, für die jeder Mensch das Recht auf eigene Beurteilung und Entscheidung haben sollte.

Auf welchen wackeligen Füßen die „offizielle“ Beurteilung der angeblich „geringen Risiken“ solcher Anlagen steht, kann man an einem einfachen statistischen Experiment erkennen. Würfelt man beispielsweise mit 3 Würfeln gleichzeitig und ordnet dem Auftreten des Ereignisses „drei 6er“ den Schadensfall (z.B. einen GAU) in einem Beobachtungs-Jahr zu, dann kann man leicht feststellen, dass er mit der Wahrscheinlichkeit von $1/(6^3) = 0.0046$ auftritt. Das bedeutet auch, dass im Mittel alle 6^3 Jahre = 21600 Jahre ein Schadensfall zu verzeichnen ist, was nicht allzu häufig wäre. Wenn man nun wie bei kerntechnischen Anlagen und ganz im Gegensatz zu den 3 Spielwürfeln, die Größe der Wahrscheinlichkeit des Schadensfalles gar nicht kennt, sondern diese erst statistisch schätzen muss, dann merkt man wie in dem Beispiel, dass man dazu das Ereignis „drei 6er“ mindestens 100 mal abwarten müsste um die tatsächliche Eintrittswahrscheinlichkeit aus der relativen Häufigkeit des Auftretens mit brauchbarer statistischer Sicherheit abschätzen zu können. Man bräuchte also mehr als 2 Millionen Jahre um eine statistisch sichere Aussage treffen zu können. Das heißt aber, der Beobachtungszeitraum bis heute reicht bei weitem nicht aus um statistisch gesicherte Aussagen treffen zu können. Genauso fehlt bei der Beurteilung des Schadessausmaßes in ganz vergleichbarer Weise die Grundlage für eine sinnvolle Schätzung.

Der andere gravierend fragwürdige Punkt der Risikoabschätzung, der bei diesem einfachen Experiment schnell klar wird, ist die Tatsache, dass beim gleichzeitigen Würfeln mit 3 Würfeln der Fall der „drei 6er“ zu jeder Zeit eintreten kann. Das ist nämlich genau das Wesen der Statistik, die ja nur die Grundgesamtheit beschreibt und nicht den Einzelfall. Während bei versicherungstechnischen Betrachtungen der Einzelfall von der Summe aller Versicherungsteilnehmer aufgefangen wird, ist ein solches Konzept bei einer nuklearen Katastrophe kaum anwendbar. Bei der Risiko-Beurteilung einer nuklearen Katastrophe müsste davon ausgegangen werden, dass der Schadensfall im nächsten Jahr genauso wahrscheinlich ist (in dem Gedankenexperiment beim nächsten „Wurf“) wie in allen anderen Jahren auch (den übrigen Würfel-Würfen). Und man müsste dabei auch davon ausgehen, dass kein Geschädigter davon profitieren kann, wenn in den folgenden 21599 Jahren (Würfel-Würfe) kein Schadensfall eintritt. Das aber würde das Risiko sofort untragbar

machen, so dass auch schnell einleuchtet, dass man mit purer Statistik hier nicht rechnen darf.

Ungeachtet der Sinnlosigkeit die wahre Größe der Risiken genau bestimmen zu wollen, wäre es aber sehr wohl sinnvoll und hilfreich, wenn eine autonome und individuelle Kontrolle durch die Bevölkerung selbst erfolgen würde und so die behördlichen und vom Anlagenbetreiber vorgenommenen Kontrollmaßnahmen komplementiert werden würden ohne sie gleich in Frage zu stellen. Die dafür notwendige Aufklärung sowie Ausstattung der Bevölkerung würde sich wie im Fall der Krebs-Früherkennung, bei der die Risken ebenfalls schwer oder gar nicht beziffert werden können, genauso bezahlt machen. Sie würden dann auch für deutlich mehr Vertrauen in die staatlichen Kontroll-Instanzen sorgen und würde helfen den Ausstieg aus der Sackgasse der nuklearen Energieerzeugung besser abzusichern und der Bevölkerung das Gefühl zu vermitteln, dass er wirklich ernsthaft betrieben und nicht wieder einmal umgekehrt wird.

Leider sind wir noch ein ganzes Stück weg davon, dass jeder Einwohner selbst die Radioaktivität der Umwelt mit einer eigenen Low-Cost Wetterstation genau wie Temperatur und Luftfeuchte messen kann und so einen Beitrag zur Überwachung und Vorsorge leisten könnte. Entsprechende professionelle Geräte kosten mehrere 10000 Euro und für die Installation und Konfiguration sind Service Techniker des Herstellers und eine spezielle Schulung durch den Hersteller nötig. Allerdings rückt diese Möglichkeit nun wenigstens für Leute, die über Kenntnisse der Mikrocontroller-Technik verfügen in den Bereich des Machbaren. Die kleine Firma 4N-Galaxy aus Hamburg hat nämlich die Idee aufgegriffen und eine autonome, photovoltaisch betriebene Messstation für Umwelt-Radioaktivität entwickelt. Sie ermöglicht in Verbindung mit einfachen und viel-verwendeten Mikrocontroller-Plattformen wie Arduino oder Raspberry Pi mit wenig Aufwand eine solche private Kontrolle durch die Bevölkerung selbst.

Die autonome Messstation R10 liefert die Strahlungsdaten eines SBM-20 Geiger-Müller Zählers im 30 Sekunden-Takt als ASCII-String entweder über Funk oder über Kabel mit 4800Baud Datenrate an den Mikrocontroller. Dabei werden die ASCII Daten entweder über eine 433MHz Funkschnittstelle wie bei einem Funk-Thermometer übertragen oder per Kabel über eine differentiellen RS485 Schnittstelle. Im Falle der Funk-Übetrtragung sorgt ein Empfänger im USB-Stick Format für eine Umsetzung über USB auf einen virtuellen COM-Port oder über einen 2.54mm Pfostenverbinder direkt auf ein serielles RS232 Protokoll. Nur die Auswertung der ASCII Daten und die Anzeige bzw. den Weitertransport ins Internet muss noch per Anwendungsprogramm auf dem Mikrocontroller selbst bewerkstelligt werden. Im Falle eines Arduino Mikrocontroller kann dazu das in diesem Dokument vorgestellte Programm (bzw. Sketch) verwendet werden. Um die Stromversorgung des Sensors (photovoltaisch versorgte NiMH Zellen), die Erzeugung der Hochspannung für das Geiger-Müller Zählrohr, sowie die Erfassung der Zählrate mit Umsetzung auf die Funk- bzw. Kabelschnittstelle muss sich der Anwender genauso wenig kümmern, wie um Feuchtigkeit und den Druckausgleich im wasserdichten Gehäuse. So gesehen ist die autonome Messstation R10 ein idealer digitaler Outdoor-Sensor für einfache und komplexere Mikrocontroller Plattformen, deren Betrieb „nur noch“ einfache informatische Kenntnisse erfordert. Sie liegt mit Kosten unter 400Euro auch durchaus im Bereich dessen, was man sich für dieses Hobby und für diesen Zweck an Kosten tragen kann und will. Am Ende leistet man mit der fertigen Lösung, welche die Daten im Internet für jedermann zugänglich macht,

einen gesellschaftlich äußerst wertvollen und von vielen sicher hoch geschätzten ehrenamtlichen Beitrag.

Die Messstation R10 enthält einen Geiger-Müller Zähler, der dem SBM-20 Driver Interface R02 von 4N-Galaxy weitestgehend entspricht und der durch einen sehr niederen Stromverbrauch auffällt. Die Elektronik der Messstation ist in einem wasserdichten Gehäuse (IP 65) untergebracht, das auch für einen Druckausgleich über ein Ventil mit Goretex-Membran sorgt und so einer Kondenswasserbildung wirksam entgegenwirkt. Der Geigerzähler liefert die für das SBM-20 Zählrohr typischen 175 Counts Per Minute pro uSv/h Gammadosisleistung (bezogen auf Cs137). Das Zählrohr hat zudem eine gewisse Empfindlichkeit für Betastrahlung sofern sie von höherer Energie ist. Des weiteren enthält die Station ein Solarpanel und vier AAA-Size NiMH Akkus, welche über einen Mikrocontroller gesteuerten Solarregler geladen und überwacht werden. Der Stromverbrauch der Station ist so gering, dass sie auch im Winter bei geringer Sonneneinstrahlung und temporärer Schneebedeckung des Solarpanels gut einige Wochen übersteht. Man sollte allerdings NiMH-Zellen mit niedriger Selbstentladung (z.B. von Eneloop) verwenden, die auch bis –20°C noch zuverlässig arbeiten. Das Solarpannel ist hinter dem transparenten Kunststoffdeckel des Gehäuses untergebracht, so dass bei Funkbetrieb keine weitere Öffnung im Gehäuse bzw. kein Steckverbinder mehr nötig ist.

Abb. 2: das Innere der Messtation R10

Darüber hinaus enthält die Station eine RS485 Kabelschnittstelle und ein Funkmodul für 433.92MHz. Das Funkmodul hat eine Sendeleistung von 10mW, sowie Jumper für zwei Boost Modi, deren Betrieb in Deutschland jedoch nicht zugelassen sind. Allerdings reichen die

10mW Sendeleistung gut um eine Distanz von 50m auch in bebautem Gebiet sicher zu überbrücken. Für die Funkübertragung wird ein störsicheres Frequency-Shift-Keying (FSK) Modulationsverfahren eingesetzt. Die differentielle Kabelübertragung per RS485 erfordert ein Twisted-Pair Kabel mit Masseschirmung (z.B. Telefonkabel) und kann gemäß Spezifikationen der RS485 Schnittstelle durchaus mehrere 100m Distanz überbrücken. Der Signalpegel der RS485 beträgt +/-5V und wird im Gegensatz zur normalen RS232 Schnittstelle mit zwei differentiellen Signalen übertragen aber ansonsten gleichem seriellen Protokoll.

Abb 3a: Die Messstation im wasserdichten Gehäuse mit Gore-Membran für den Druckausgleich (gelber Punkt) und Solarpanel unter dem transparenten Kunststoffdeckel

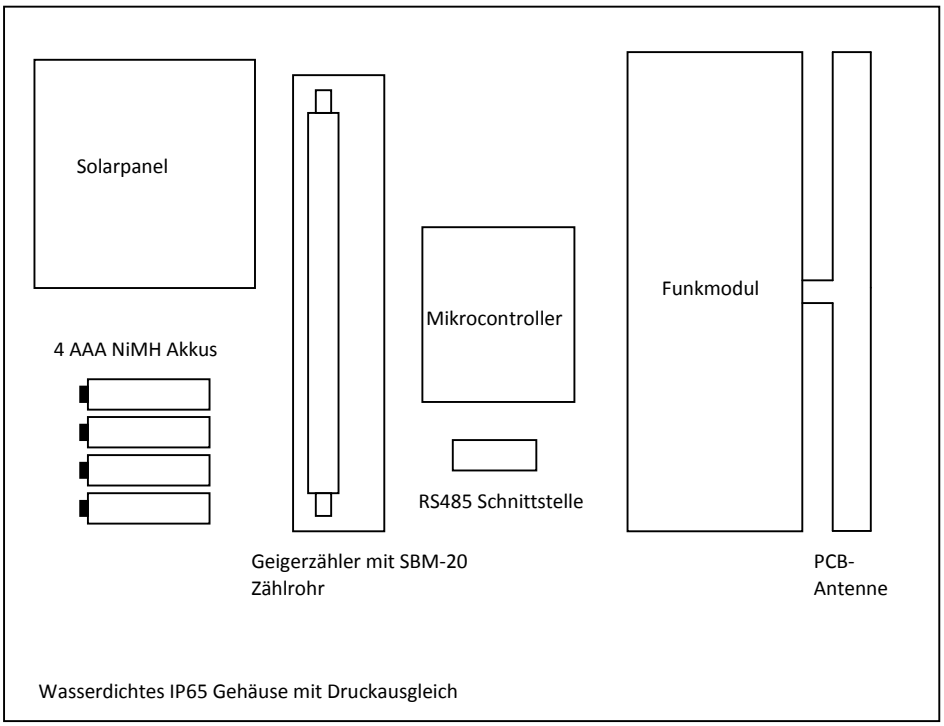

Abb. 3b: Funktionskomponenten der autonomen Messstation

Der Mikrocontroller erfasst die Zählimpulse und berechnet alle 30 Sekunden eine Zählrate. Außerdem wird die Temperatur im Gehäuse erfasst, sowie die Spannung der Akkus. Aus diesen Daten wird eine ASCII Datennachricht zusammengesetzt und über eine Checksum gegen Übertragungsfehler abgesichert. Außerdem erzeugt der Mikrocontroller eine laufende Nummern für die Nachrichten, die vom verarbeitenden Mikrocontroller auf Lücken kontrolliert werden können. Außerdem wird außer dem laufenden Strahlungs-Messwert immer auch noch der vorige Wert übertragen, so dass eine gewisse Redundanz besteht falls eine Nachricht verloren geht. Jede Nachricht wird 3 Mal übertragen. Im Empfänger wird auf 3 Nachrichten gewartet und für jede empfangene Nachricht wird die Checksum geprüft. Aus der Zahl der empfangenen Nachrichten und den Checksum Fehlern wird im Empfänger ein Signalgüte Parameter berechnet, der vom Empfänger an eine empfangene Nachricht angehängt wird. Dieser Parameter ist ein zuverlässiges Maß für die Sicherheit der Funkübertragung, der nicht nur Möglichkeiten der Fehlererkennung bietet, sondern auf Grund der Redundanz auch einer gewisse Fehlerkorrektur ermöglicht. In der Detektorbox zeigen zwei LEDs den Status an: Eine LED signalisiert einen registrierten Zählimpuls, die andere die Übertragung einer Datennachricht. Im Empfänger signalisieren ebenfalls zwei LEDs den Status, eine blinkende, welche die grundsätzliche Aktivität des Empfängers anzeigt und eine zweite, welche den Empfang einer Nachricht sowie gewisse Fehlerbedingungen signalisiert.

Der Empfänger-Stick enthält auch einen Mikrocontroller, der den Empfänger steuert und die empfangene Nachricht dekodiert und auf Fehler prüft. Dieser gibt die Nachricht einmal direkt am zweireihigen 2.54mm Pfostenstecker aus. An diesem Pfostenstecker kann außerdem eine externe Antenne angeschlossen werden sowie eine 5V Stromversorgung bereitgestellt werden. Der Mikrocontroller reicht die Daten aber auch an einen FTDI Konverterchip weiter, der sie auf ein virtuelles COM-Port Format bringt und über eine USB-Buchse an einen USB-Port weitergeben kann. Wird der USB Stecker in einen Windows-PC gesteckt, installiert sich die entsprechende Treibersoftware ähnlich wie bei einem USB-Speicherstick von selbst. Danach kann z.B. über ein Terminalemulationsprogramm auf einen COM-Port zugegriffen werden, der mit dem USB-Port assoziiert ist. Dann erscheinen die Nachrichten im Klartext, wenn das serielle Datenprotokoll auf 4800Baud mit 8Bit, keine Parity und 1 Stopbit (8N1) eingestellt ist.

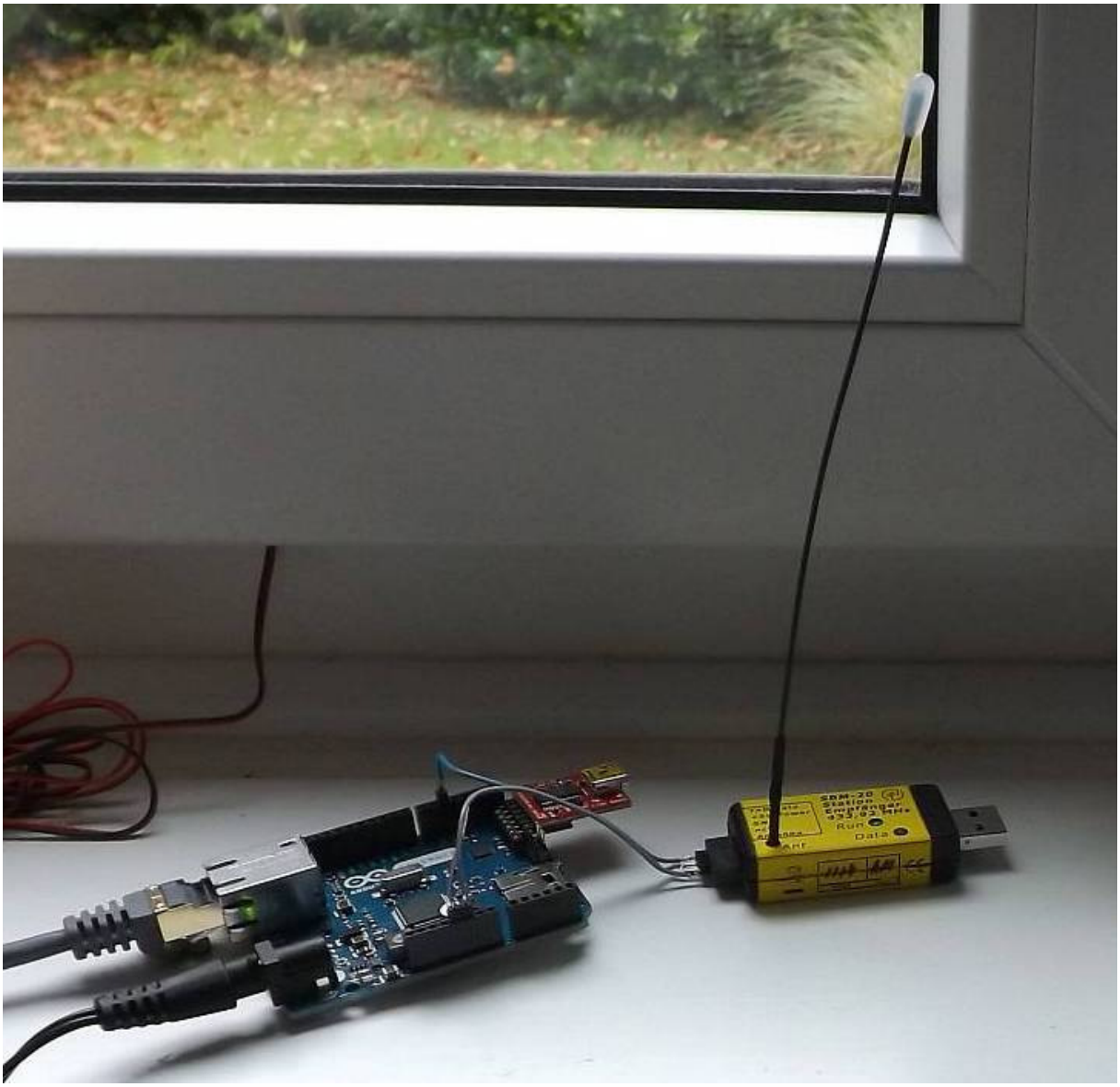

Abb. 3: Lediglich 3 Leitungen sind nötig: Verbindung des Empfänger-Sticks mit dem Arduino Mikrocontroller (provisorische Lösung ohne Berücksichtigung einer Groundplane für die Antenne)

Die beiden Schnittstellen bieten eine äußerst hohe Flexibilität für die Anpassung der gesamten Station an die unterschiedlichen Gegebenheiten am Aufstellungsort. Idealerweise sollte eine Messstation für Umwelt-Radioaktivität im Freien im Zentrum einer Grasfläche von ca. 3x3m in einer Höhe von 1m installiert werden. Das kommt den Bedingungen der amtlichen Messstellen für die Gamma-Ortsdosisleistung (ODL-Stationen des BfS) sehr nahe. Eine leichte Neigung der Box ist vorteilhaft, dazu wird das Solarpanel auf die Frühjahrshöhe der Sonne im Süden ausgerichtet (ca. 45° in Deutschland). Als einfache Lösung für die Aufstellung der Detektorbox bietet sich in der Verwendung einer Gelenk-Teleskopstange aus dem Malerzubehör an (z.B. Leifheit Klick-System 110-190cm), an die eine Kunststoffschale mit Hilfe einer Rundschelle angeschraubt wird. Diese Schale nimmt die Detektorbox mit der optimalen Neigung auf und schütz sie zusätzlich (siehe Bild 4a und 4b).

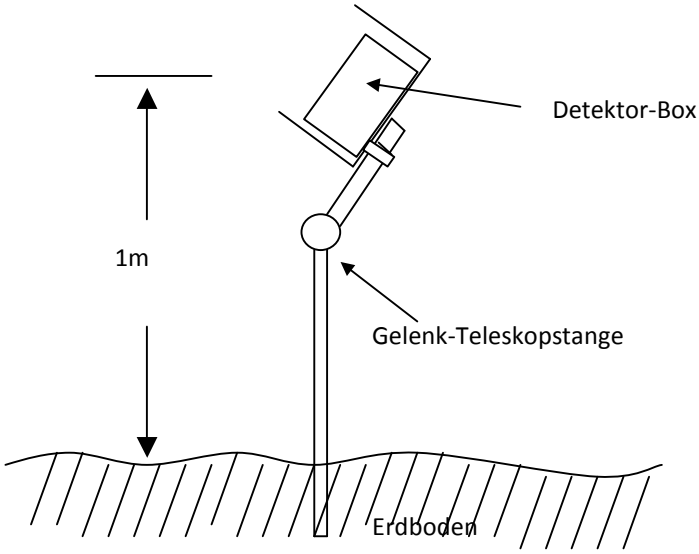

Abb. 4a: Schematische Darstellung der Aufstellung der Detektorbox

Abb. 4b: Benötigtes Material zur Aufstellung (empfehlenswert ist die abwinkelbare Teleskopstange von Leifheit)

Wer kein Problem mit Funkwellen hat (der Sender steht im Garten und hat eine vergleichbare Leistung zu einem Funk-Außenthermometer), der braucht keinerlei Kabel zu verlegen und auch sonst am Gehäuse nichts zu verändern. Im Gebäude kann der Empfänger direkt am PC oder Webserver betrieben werden, was allerdings mit einem hohen Stromverbrauch verbunden ist. Sinn und Zweck der Station ist es ja, eine Überwachung non-stop (24/7) zu ermöglichen.

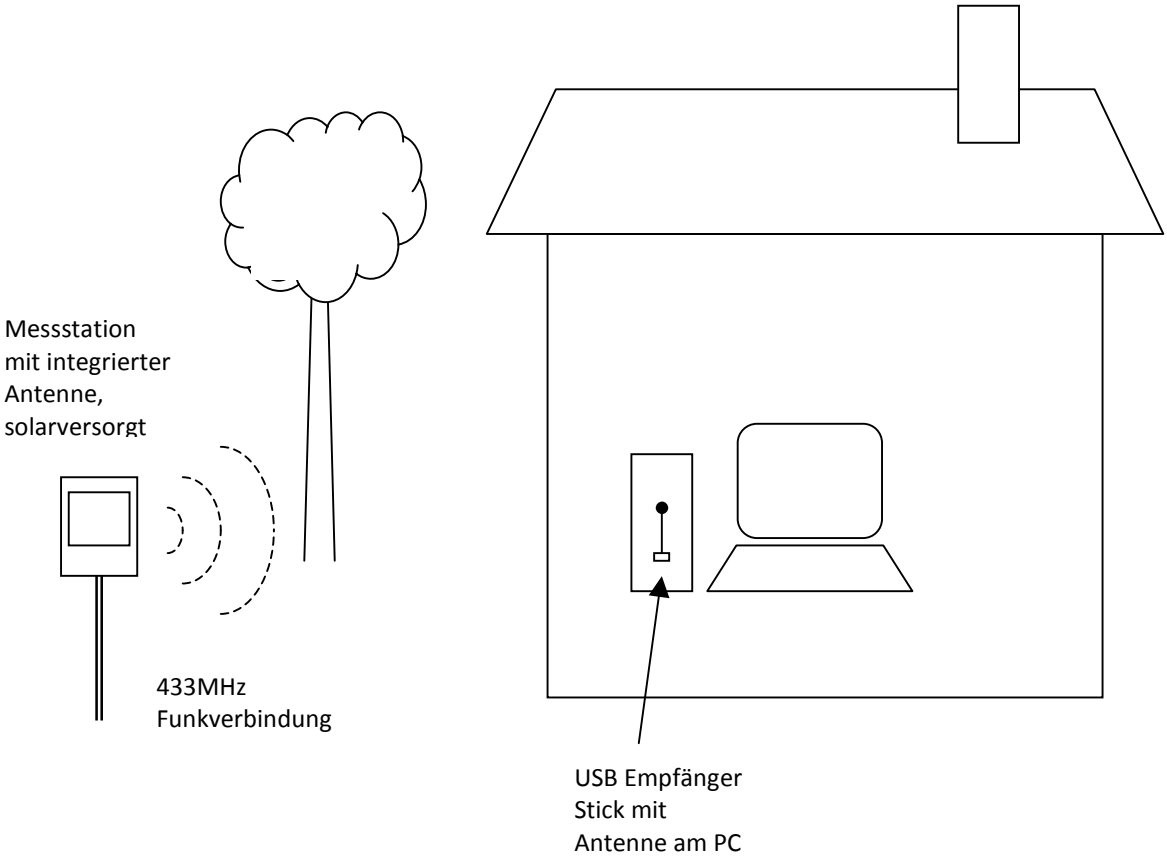

Abb. 5a: Einfacher Betrieb der Station mit Funkverbindung am PC

Eine vom Energiestandpunkt und den Kosten her deutlich bessere Alternative besteht darin, ein Mikrocontroller-Board einzusetzen, das eigenständig eine Verbindung mit dem Internet, entweder über DSL oder über ein Mobilfunknetz herstellt um die Daten auf einen Internetserver zu übertragen. Hierzu ist ein ganz einfaches Mikrocontroller-Board z.B. ein Arduino Uno völlig ausreichend, auf das entweder ein GSM/GPRS Shield oder ein Ethernet Shield aufgesteckt werden kann. Eine solche Lösung im Gebäude erzeugt einen Stromverbrauch von unter 100mA an einem herkömmlichen 9V Steckernetzteil. Alternativ besteht natürlich auch die Möglichkeit den Empfänger und die Datenaufbereitung in einem Gebäude oder in einem Gartenhaus ebenfalls über eine Solaranlage zu versorgen. Da es für die Mikrocontroller-Boards auch Speicherkarten-Aufsätze gibt, ist eine zusätzliche Datensicherung auf einer großen SD-Karte mit 16 oder 32Gbyte ebenfalls leicht möglich. Genauso kann ein einfaches serielles LCD-Display für Debugging-Zwecke angeschlossen werden.

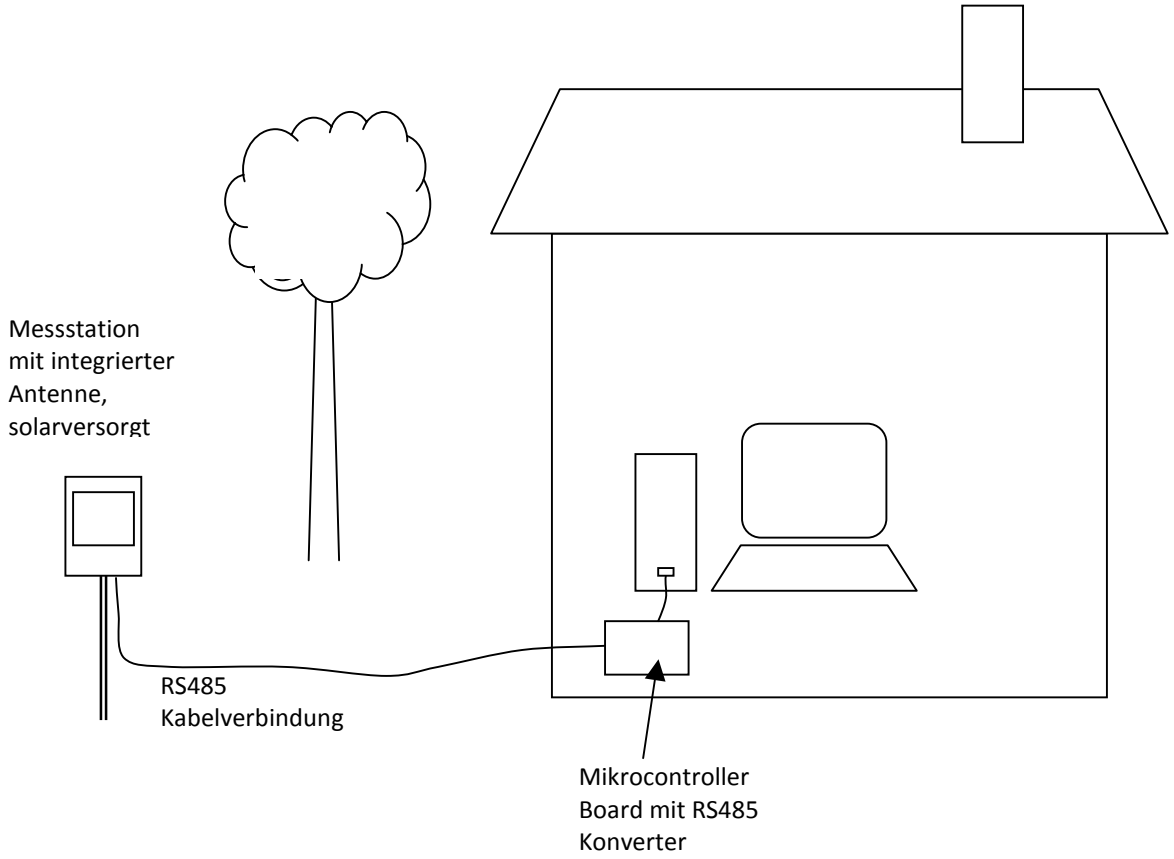

Abb. 5b: Betrieb der Station mit einer reinen Kabelverbindung ohne Funk

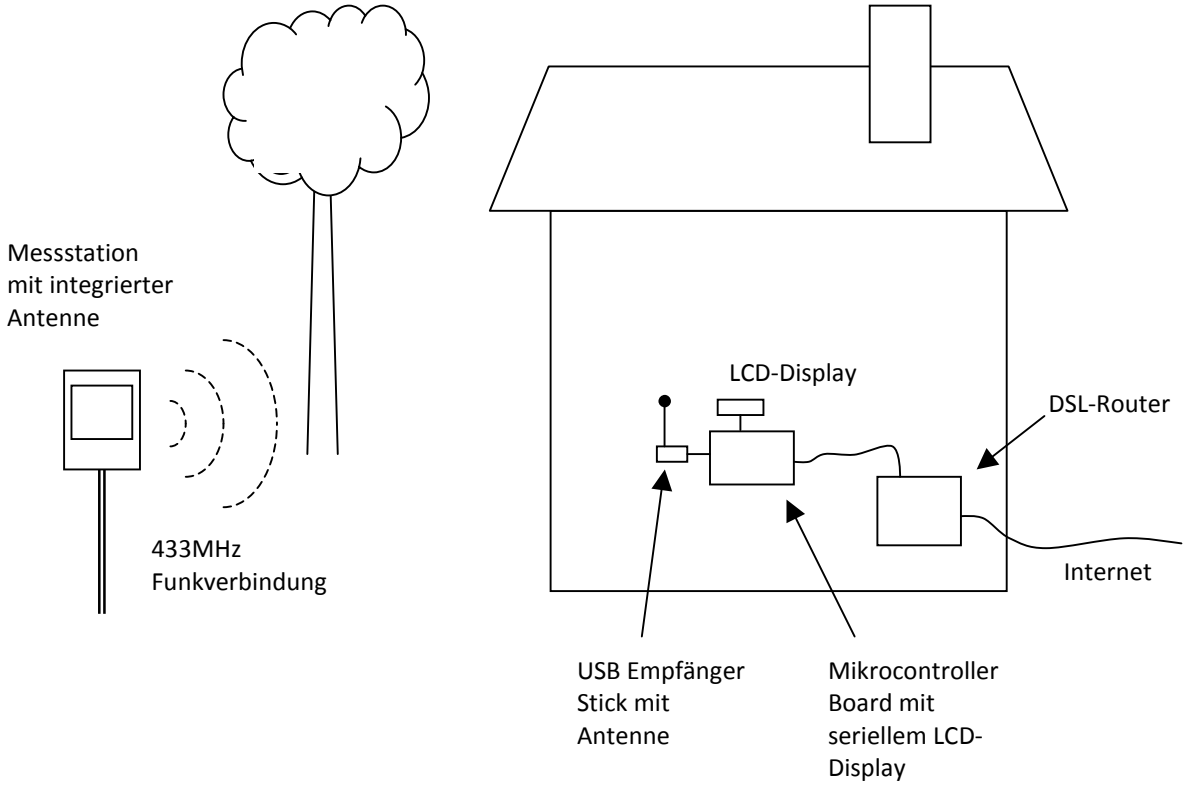

Abb. 5c: Betrieb der Station mit stromsparendem Mikrocontroller und Dateneinspeisung ins Internet

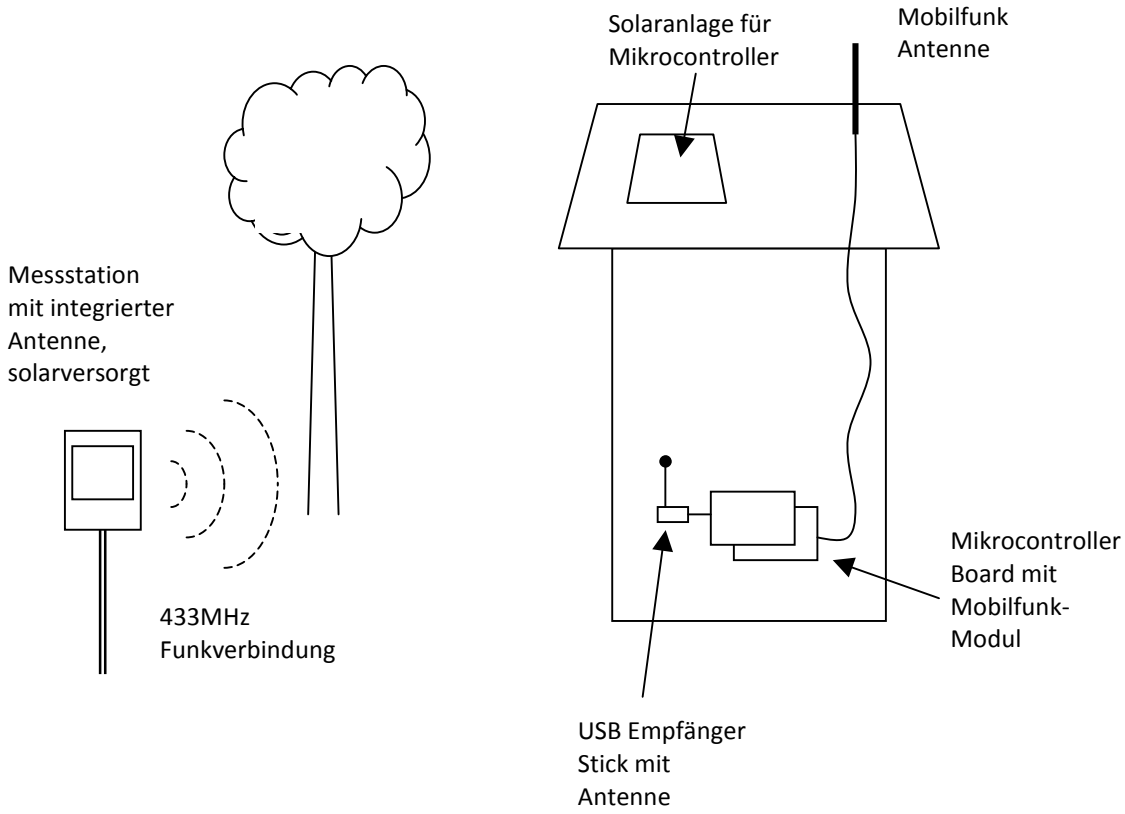

Abb. 5d: Betrieb der Messstation in einem Gartenhaus mit Mobilfunk-Anbindung ans Internet

Auf der Oberseite des Empfänger-Sticks befindet sich ein kleiner Stift auf den die mitgelieferte Stabantenne gesteckt werden kann. Um eine sichere Funkverbindung zu gewährleisten ist es aber absolut entscheidend, dass die Empfangsantenne (λ/4-Stabstrahler) auch eine entsprechende Groundplane „sieht". Solange der Empfängerstick mit dem USB Stecker in einem PC-Gehäuse aus Metall gesteckt wird ist das gewährleistet. Betreibt man den Stick aber über den Pfostenstecker und die entsprechenden für die Stromversorgung vorgesehenen Stifte, dann muss gesondert für eine Groundplane gesorgt werden. Die einfachste Lösung ist, den Empfänger mit dem Mikrocontroller Board auf ein isoliertes Kupfer- oder Alublech zu stellen, das etwa 25x50cm groß sein sollte. Man kann jedoch auch eine selbstgebaute externe 50Ω Groundplane Antenne anschließen, die aus einem Stabstrahler mit 4 Radials aufgebaut ist, welche die Groundplane ersetzen.

Aber auch derjenige, der Funkwellen ganz grundsätzlich scheut, kann mit einem Jumper das Sendemodul in der Detektorbox komplett abschalten und die Daten mit der eingebauten RS485 Schnittstelle über ein Kabel übertragen. Dazu ist ein dünnes Telefonkabel ausreichend, es muss zwei Adern für die differentielle Übertragung mit 5V Pegeln haben, sowie eine Schirmung oder eine zusätzliche Ader, die sowohl als Schirmung als auch zur Übertragung eines Referenz-Ground dient. Auf der Mikrocontroller-Seite muss dann ein Receiver oder Transceiver Chip vorgesehen werden, wie z.B. der SN65176 von Texas Instruments, der die RS485 Signale wieder in RS232 Signale umsetzt. Für kurze Distanzen (einige Meter) kann aber auch in der Box das digitale Zählimpulssignal abgegriffen und über

Kabel direkt auf den Inerrupt-Eingang des Mikrocontroller geführt werden um z.B. eine eigene Zählraten-Berechnung zu implementieren.

Die ASCII-Nachrichten, welche der Empfänger abliefert haben folgende Bauart:

000016 000009 5.4V +04°C 69% 26NR(CR)(LF)

Aktuelle Zählrate | Vorige Zählrate | Akkuspannung | Temperatur | Signalgüte | Laufende Nummer | Zeilenende

Folgende Details dazu sind noch wichtig:

- Die Nachrichten werden zeilenweise gesendet und haben einen zeitlichen Abstand von 30*1,024 Sekunden
- Die Temperatur wird im Prozessor gemessen, unter 0°C werden die Akkus nicht geladen um sie bezüglich der Lebensdauer zu schonen
- Der Signalgüteparameter wird aus der Anzahl der korrekt empfangenen Nachrichten wie folgt ermittelt:

Die erste Ziffer kann 0, 3, 6 oder 9 sein.
0 - keine Nachricht korrekt empfangen
3 - nur eine Nachricht korrekt empfangen
6 - zwei Nachrichten korrekt empfangen
9 - alle drei Nachrichten korrekt empfangen

Die zweite Ziffer kann 0, 3, 6 oder 9 sein.
0 - keine Nachricht empfangen
3 - nur eine Nachricht empfangen
6 - zwei Nachrichten empfangen
9 - alle drei Nachrichten empfangen

z.B. bedeutet 39% - drei Nachrichten empfangen davon wurde aber nur eine korrekt und vollständig mit der richtigen Checksumme empfangen.

- Die laufende Nummer wird Modulo 100 berechnet, d.h. nach 99 kommt wieder die 00.

- Wenn die Checksummenprüfung bzw. der Vergleich der 3 mal wiederholt gesendeten Nachricht einen Fehler ergibt, dann wird nach ca. 40 Sek. eine Fehlernachricht ausgegeben. Dabei wird der Signalgüte-Parameter auf 00% gesetzt. Eine Nachricht sieht dann wie folgt aus:

EEEEEE EEEEEE E.EV +EE°C 00% 26NR(CR)(LF)

Auf Grund der umfangreichen Fehlersicherung zeigt sich die Funkverbindung als äußerst robust. Versuche haben auch gezeigt, das Distanzen bis 50m mit guten Signalgüte-Werten überbrückt werden können. Obwohl außerhalb Deutschlands auch ein Boost Modus möglich wäre, der für höhere Reichweiten sorgt, ist ein Betrieb im Normalmoduls in jedem Fall wegen des geringeren Stromverbrauchs vorzuziehen. Falls die Detektor-Box an einem verschatteten Ort aufgestellt werden muss, kann ein externes Solarpanel problemlos an dem entsprechenden Pfostenstecker angeschlossen werden. Ein zusätzlicher wasserdichter Steckverbinder muss dann natürlich genau wie im Fall des RS485 Datenkabels nachträglich in das Gehäuse eingebaut werden.

Der Anschluss an einen Mikrocontroller ist wegen des seriellen Protokolls und dem ASCII Nachrichtenformat denkbar einfach. Im Falle eines Arduino Boards kann einfach die Software Serial Bibliothek eingebunden werden um zwei Digital-Pins als entsprechende serielle Datenübertragungs-Pins zu definieren. Der Rest ist reine String-Behandlung und Datenauswertung.

Besonders interessant ist natürlich ein Mikrocontroller mit Ethernetanschluss und Internetzugang. Für den Arduino kann dafür entweder ein Ethernet-Shield oder der Ethernet Arduino in Verbindung mit einem LAN-Kabel benutzt werden, welches auf der anderen Seite in den DSL-Router eingesteckt wird, bzw. ein Wireless-Shield, das mit einem Wireless-Router kommunizieren kann.

Auf diese Weise können die Daten an einen Webserver geschickt werden, der die Daten im Internet abrufbar macht. Dazu kann entweder ein Internetportal wie Xively (früher Pachube) benutzt werden, oder aber man nutzt die Dienste eines Webspace-Providers und macht von der Verfügbarkeit von PHP oder andern Web-Sprachen in Verbindung mit SQL-Datenbanken Gebrauch um die Daten zu speichern und zu visualisieren. Ein Beispiel dazu ist für einen Ethernet Arduino im Folgenden gezeigt. Das Flussdiagramm zeigt die prinzipielle Programmstruktur, der detaillierte Programmcode findet sich im Anhang.

Was gegenüber den bisher vorgestellten Arduino Programmen zur Erfassung von Zählraten und ihrer Übertragung ins Internet hier von spezieller Bedeutung ist, ist die Tatsache, dass gemittelt wird um die Menge der Daten überschaubar zu halten. Das Einlesen der Daten vom Empfänger ist recht trivial gestaltet. Im Prinzip muss der ASCII String, der gepuffert wird, zunächst zerlegt werden. Dies erfolgt durch sukzessives Einfügen von ‚\0' Zeichen. Dadurch entstehen die entsprechenden Substrings, welche als Zahlen interpretiert werden. Wenn nun aber ein Empfangsfehler auftritt, stellt ein Substring keine Zahl dar und kann daher nicht ins Fleißkommaformat konvertiert werden. Die Signalgüte erscheint aber dennoch als eine Zahl mit dem Wert Null und kann daher konvertiert werden. Deswegen wird im Fehlerfall nur die Signalgüte weitergemittelt während die Mittelwertbildung der übrigen Parameter wie Zählrate, Akkuspannung und Temperatur ausgesetzt wird. Damit wird bei einem Empfangsfehler nur die Signalgüte schlechter, was durchaus so gewollt ist. Die Mittelwerte werden nach Erreichen einer Fenstergröße von 20 alle 10.24 Minuten auf den Server übertragen.

Auf dem Server wird ein Skript angestoßen, das die übertragenen Werte in eine SQL-Datenbank einträgt. Gleichzeitig wird ein Skript bereitgestellt, mit dem die Messdaten der letzten 7 Tage graphisch dargestellt werden können.

Das Ergebnis der Skriptbearbeitung ist unter: http://opengeiger.de/graph_sensornwh.php für die derzeit betriebene Station im Internet abrufbar.

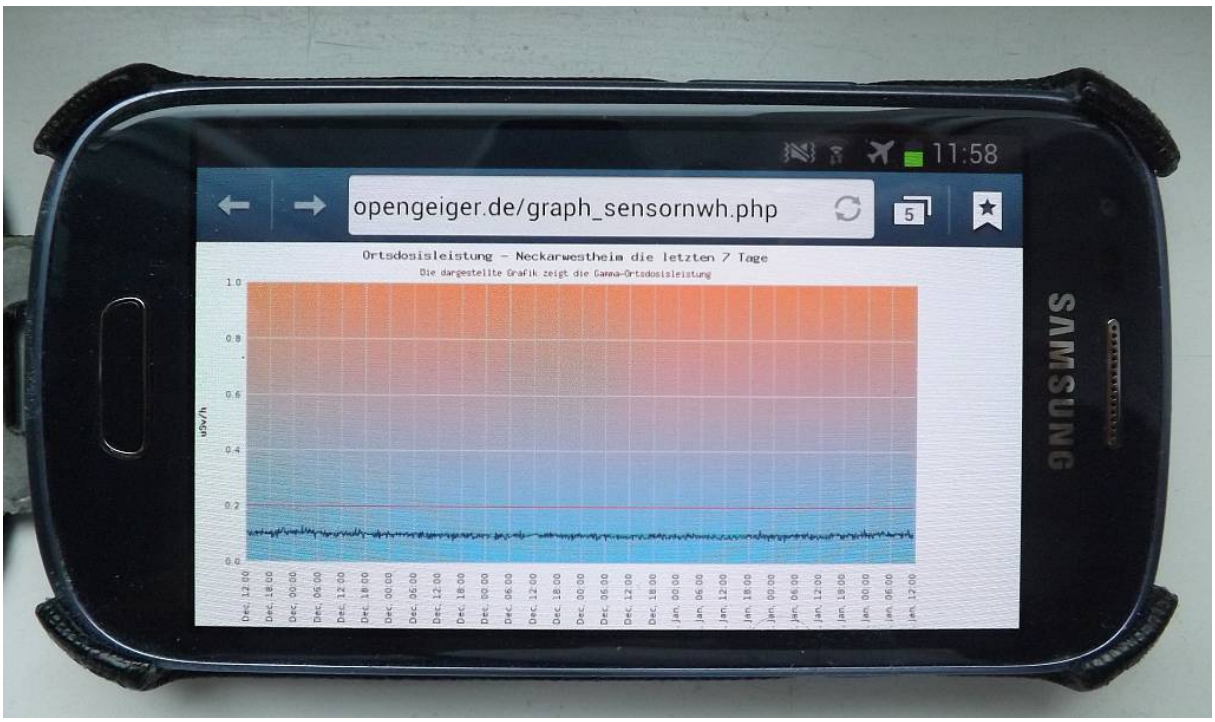

Abb. 7: Die Radioaktivitäts-Messdaten können auch per Mobiltelefon im Internet abgerufen werden

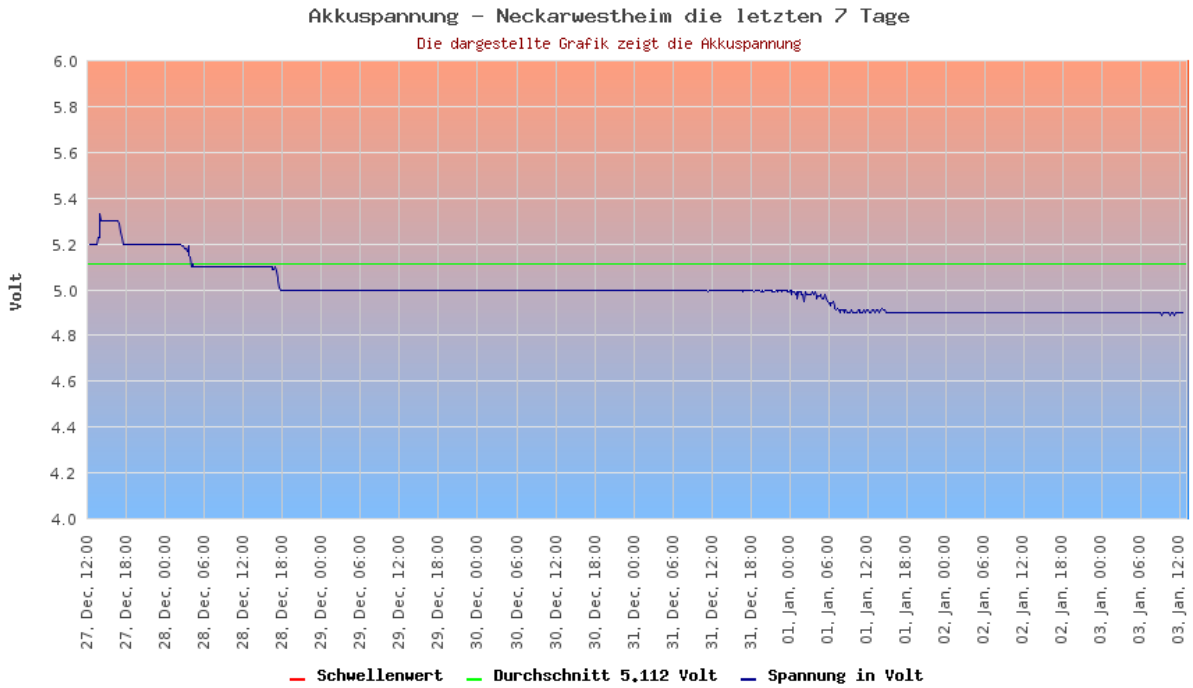

Abb. 8: Auch die Akkuspannung kann sogar per Internet überwacht werden

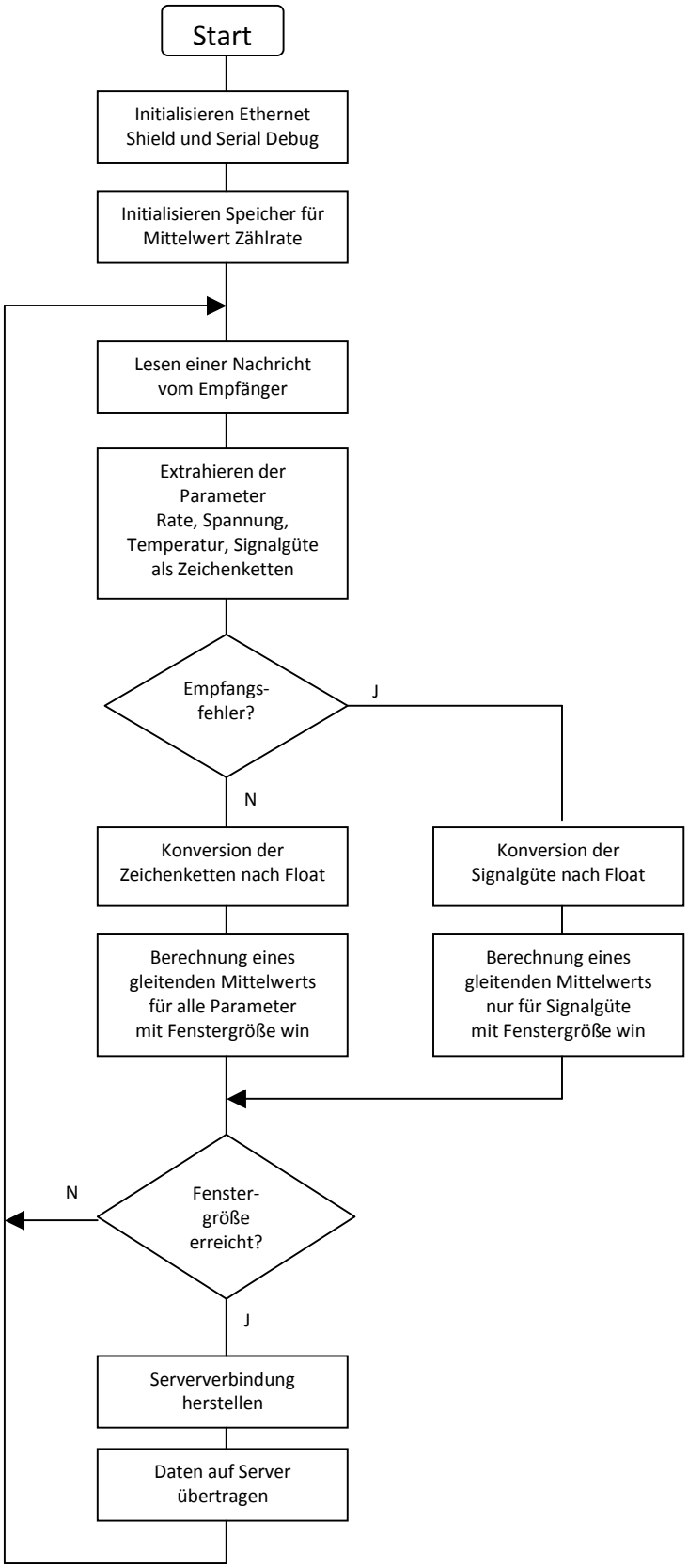

Abb. 6: Programmstruktur der Datenverarbeitung auf dem Arduino

Literatur

Produkt Webseite von 4N-Galaxy mit Beschreibung und Spezifikationen:
http://www.4n-galaxy.de/R10_de.html

Beispiele für professionelle Messsysteme der Firma Envinet:
http://www.envinet.com/deutsch/

Arduino:
http://www.arduino.cc

ODL Messsonde MWS III des BfS :
http://www.freiburg.linux.de/vortraege/mws3/lt2007.pdf
http://iss.umwelt-campus.de/iss/fileadmin/template/pdf/Fachtagung_2011/bfs_rluff_g.pdf

Dosisleistungskarte der Universität Fukushima
http://www.sss.fukushima-u.ac.jp/FURAD/FURAD/data-map-fukushima.html

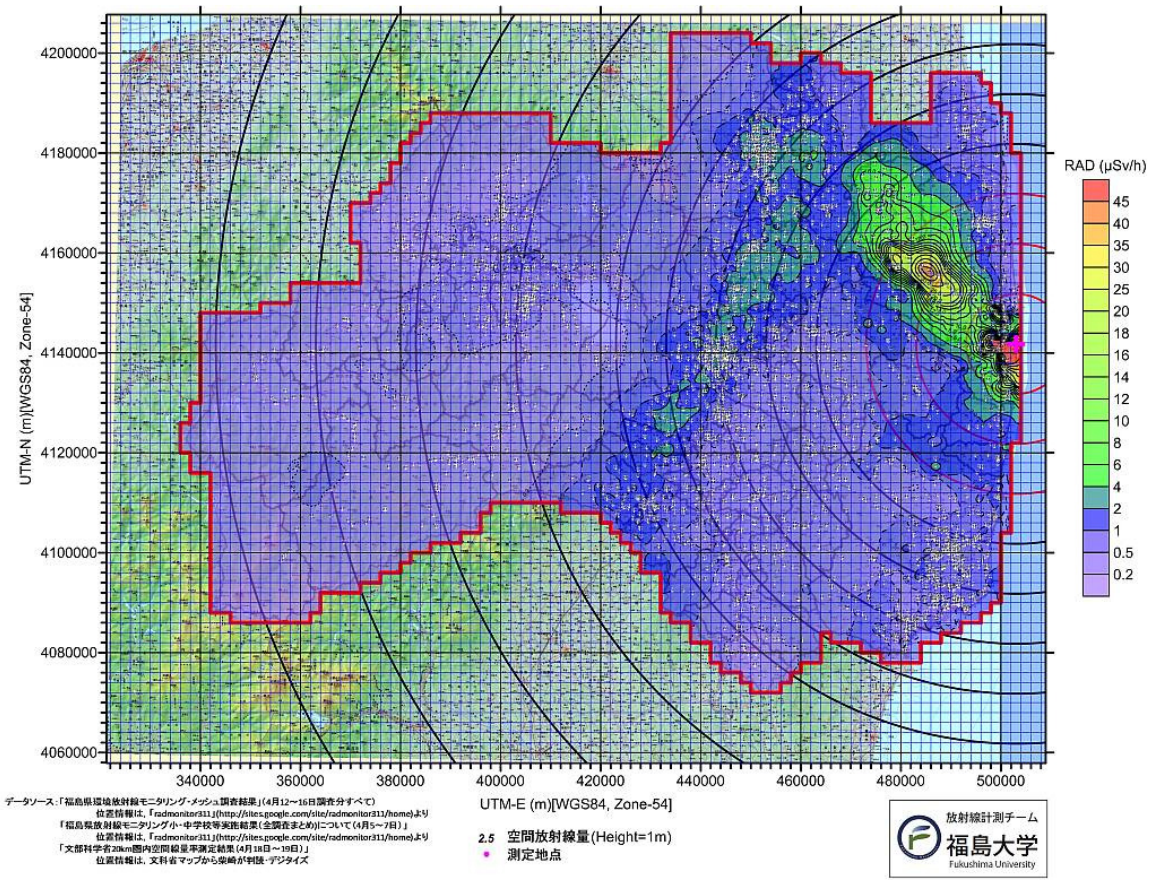

Abb. 9: Verteilung der von der Universität gemessenen Dosisleistung in µSv/h auf dem geodätischen WGS84 UTM-Gitter. Die Entfernungskreise sind alle 10km eingezeichnet, zwangsevakuiert wurden die zwei kleinsten Kreise obwohl auch der dritte Kreis noch ein starkes Nebenmaximum enthält.

Anhang

Listing für den Betrieb der autonomen Messstation für Umweltradioaktivität an einem Ethernet Arduino:

```
#include <SPI.h>
#include <Ethernet.h>
#include <HttpClient.h>
#include <SoftwareSerial.h>
#define win 20 // window of running average

SoftwareSerial mySerial(5, 6); // RX, TX
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0E, 0xC4, 0xE7 };
EthernetClient client;
float arrrate[win], arrvolt[win], arrtemp[win], arrfeld[win];
String mrate, mvolt, mtemp, mfeld; // in case of error these values must survive

int nrval = 1;

float * push(float *arr, float val) {
  int i;
  for (i=win-1; i>0; i--)
    arr[i]=arr[i-1];
  arr[0]=val;
}

float mean(float *arr) {
  float m = 0;
  int i;
  for (i=0; i<win; i++)
    m += arr[i];
  return (m/win);
}

void setup() {
  int i;
  Serial.begin(9600);
  while (Ethernet.begin(mac) != 1)
  {
    Serial.println("Error getting IP address via DHCP, trying again...");
    delay(15000);
  }
  mySerial.begin(4800);
  Serial.println("waiting for data...");
  for (i=0; i<win; i++) arrrate[i]=0;
}

void loop() {
  char buf[40];
  char cc;
  int n = 0;
  char *actval;
  char *prevval;
  char *volt;
  char *temp;
  char *field;
  char *valnr;
  String getstr;
  String strrate, strvolt, strtemp, strfeld;
  float fltrate, fltvolt, flttemp, fltfeld;

  do {
  } while (!mySerial.available());
  cc=mySerial.read();
  while (cc != '\n'){
    buf[n]=cc;
    n++;
    do {
    } while (!mySerial.available());
    cc=mySerial.read();
  }
  buf[n]='\0';
  Serial.println(buf);
  n = 0;

  buf[6]='\0';
  actval=buf;
```

```
  buf[13]='\0';
  prevval=buf+7;
  buf[17]='\0';
  volt=buf+14;
  buf[22]='\0';
  temp=buf+19;
  buf[27]='\0';
  field=buf+25;
  buf[31]='\0';
  valnr=buf+29;

  volt[1] = volt[2]; // remove decimal point;
  volt[2] = '\0';

  Serial.println(actval);
  Serial.println(prevval);
  Serial.println(volt);
  Serial.println(temp);
  Serial.println(field);
  Serial.println(valnr);
  strrate = String(actval);
  strvolt = String(volt);
  strtemp = String(temp);
  strfeld = String(field);
  fltfeld = float(strfeld.toInt());

  if (buf[0] != 'E') { //only calc new mean for these parms if no error
    fltrate = float(strrate.toInt());
    fltvolt = float(strvolt.toInt());
    flttemp = float(strtemp.toInt());
    push(arrrate, fltrate*10);
    mrate = String(int(mean(arrrate)));
    push(arrvolt, fltvolt*10);
    mvolt = String(int(mean(arrvolt)));
    push(arrtemp, flttemp*10);
    mtemp = String(int(mean(arrtemp)));
  }
  push(arrfeld, fltfeld); // in case of error feld = 0, mean is valid
  mfeld = String(int(mean(arrfeld)));
  if ((nrval % win) == 0) { //send only on average pe running average window
    Serial.println("\nStarting connection to server...");
    //if you get a connection, report back via serial:
    if (client.connect("www.opengeiger.de", 80)) {
      Serial.println("connected to server");
      // Make a HTTP request:
      getstr = <Aufruf Serverseitiges php Skript mit Parametern>
      client.println(getstr);
      client.println("Host: www.opengeiger.de");
      client.println("Connection: close");
      client.println();
      Serial.println(getstr);
      Serial.println("Done.");
      Serial.println("Disconnecting.");
      client.stop();
    }
    else { // you didn't get a connection to the server
      Serial.println("No connection to server");
    }
  }
  nrval++;
}
```