Was ist "LoRa"(TM) ?

oder: Spread-Spectrum-Funkverbindung zur Übertragung von (GPS-) Telemetriedaten

In Bearbeitung

Stand: 18. März 2020

Automatic translation by GOOGLE:
https://translate.google.de/translate?hl=de&sl=de&tl=en&u=http%3A%2F%2Fwww.kh-gps.de%2Flora.htm

---

NEU im März 2020: http://www.kh-gps.de/lora_aprs2.htm

WAS IST NEU:

*** Von mir gibt es einen neuen Platinenentwurf, bei dem RFM-9x-LoRa-Bausteine und Pro-Mini-Mikrocontrollerbausteine auf einer Karte vereint wurden [30]. Dabei wurde kein besonderer Wert auf Miniaturisierung, dafür aber einfache Bestückbarkeit gelegt. Interessenten können Platinen über Dirk Ruffing, DH4YM [35] erwerben.**

*** Hinzugekommen ist eine Seite, auf der es um u.a. LoRa-Textkommunikation zwischen Android-Smartphones oder Tablets geht [33].**

*** Vorliegende Seite wurde um Informationen zur Nutzung der "DRAGINO" ARDUINO-Shields zur LoRa-Übertragung von Daten des BOSCH-Sensors "BME280" erweitert (siehe dazu weiter unten).**

*** Erfolgreiche Versuche erfolgten mit einer über LoRa gesteuerten Schaltfunktion incl. Quittierung. Mehr dazu ist hier [28] (neue Version im Dez. 2017) zu lesen.**

*** Getestet wurde ein "LoRa-zu-APRS-Gate". Im Gegensatz zu ähnlichen Projekten, bei denen ausgehende Daten direkt in das Internet übermittelt werden, habe ich dabei allerdings einen etwas anderen Ansatz verfolgt, indem ich via LoRa empfangene Navigationsdaten in das Standard-APRS-Format wandele und zur Aussendung auf z.B. der Frequenz 144.8 MHz bereitstelle.**

*** Interessant ist auch die realisierte LoRa-Repeaterfunktion. Auf einfache Weise ermöglicht sie eine Wiederaussendung empfangener Datenpakete und kann damit bei entsprechender Standortwahl die Reichweite von LoRa-Funksystemen erheblich erweitern [29].**

*** Für das weiter unten auf dieser Seite bereits beschriebene GPS-Datenübertragungssystem gibt es inzwischen Softwareversionen mit zusätzlicher Einfügung von Ident-Kennungen (z.B. Amateurfunkrufzeichen) und Bereitstellung serieller APRS-Empfangsprotokolle im KISS-Format. Damit wird z.B. die Auswertung und Kartenvisualisierung empfangener Positionsdaten mithilfe von ANDROID-Software wie z.B "APRSDROID" möglich.**

*** In Österreich gibt es ein von Wiener-Funkamateuren in Zusammenarbeit mit dem Versuchssenderverband "ÖVSV" entwickeltes Projekt, welches z.B. ein (nachbaubares) "LoRa-APRS-iGate" beinhaltet. Dieses ermöglicht die APRS-Netzwerkintegration von LoRa-Daten, die z.B. auf einer Frequenz im 70cm-Band gesendet wurden [31],[32].**
Entsprechende Übertragungsversuche hat es mit beachtlichen Ergebnissen bereits in Wien und Tirol gegeben.
Seit kurzer Zeit gibt es vom ÖVSV auch ein (noch auszubauendes) WIKI [34] zum Thema "Ham-IOT" (Internet of Things für Funkamateuere)

"LoRa" (TM) steht für "Long Range". Dabei handelt es sich um ein Funkübertragungsverfahren, mit dessen Hilfe sich beispielsweise Telemetriedaten im Rahmen des IoT (Internet of Things) über Distanzen übertragen lassen, die im Vergleich zur Nutzung von z.B. FSK- oder GMSK-Modulation bei üblicherweise mehr als dem Zehnfachen liegen. Erreicht wird das durch die Verwendung von Spread Spectrum Modulation. Der sich hierbei ergebende Systemgewinn von 20dB und mehr würde ansonsten nur durch Erhöhung der Sendeleistungen um den Faktor 100 erreichbar sein. Bei "LoRa" handelt es sich um eine ursprünglich von der Firma SEMTECH eingeführte Technologie. Als weiterer Systemvorteil sei auch noch der selbst längere Batteriebetriebszeiten erlaubende geringe Energiebedarf genannt.
Mehr zu "LoRa" findet man beispielsweise hier: [15], [16], [17], [18]. Wenn es um nichtkommerzielle Hobbyanwendungen geht, so gibt es einige Einsatzbeispiele z.B. auch im englischsprachigen Forum www.rcgroups.com [1], [2], [8].

Abb.1 Logo der LoRa ALLIANCE (TM)

Durch Verwendung sehr preiswert erhältlicher Transceiverbausteine [4] lassen sich in der genannten Betriebsart Distanzen von (natürlich geländeabhängig) bis zu 20 Km und mehr überbrücken. In Verbindung mit Ballonprojekten konnten es mehrfach aber auch schon einige Hundert Kilometer sein.
Sendeleistungen von nur wenigen Milliwatt (die üblichen Bausteine erlauben Leistungswerte zwischen 2 und 100mW) gestatten den Einsatz durch Jedermann. Hierbei sind lediglich die gesetztlichen Bestimmungen im Rahmen von SRD-Allgemeinzulassungen (SRD = Short Range Devices) einzuhalten [13]. Infrage kommende Frequenzbereiche liegen bei 433 MHz und 868 MHz.
Nicht geeignet ist "LoRA" allerdings in Verbindung mit hohen Datenraten. Die Vorzüge des Systems kommen vor allem dort zum tragen, wo es um die Übertragung nur relativ geringer Datenmengen in nicht allzu häufigen Zeitfolgen geht. Klassische Anwendungen sind allgemeine Telemetriedatenübertragungen z.B. im Rahmen des "Internet of Things" oder auch Trackingsysteme für den Nahbereich. GPS-Standortdaten lassen sich dabei beispielsweise zur Verfolgung von Tieren oder auch allgemein zum Wiederauffinden von Objekten nutzen.

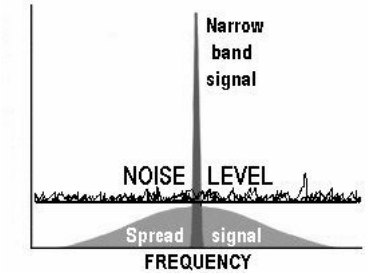

Abb.2 Vergleich zwischen "normalem" Schmalband- und einem unter dem Rauschpegel liegenden Spread Spectrum-Signal

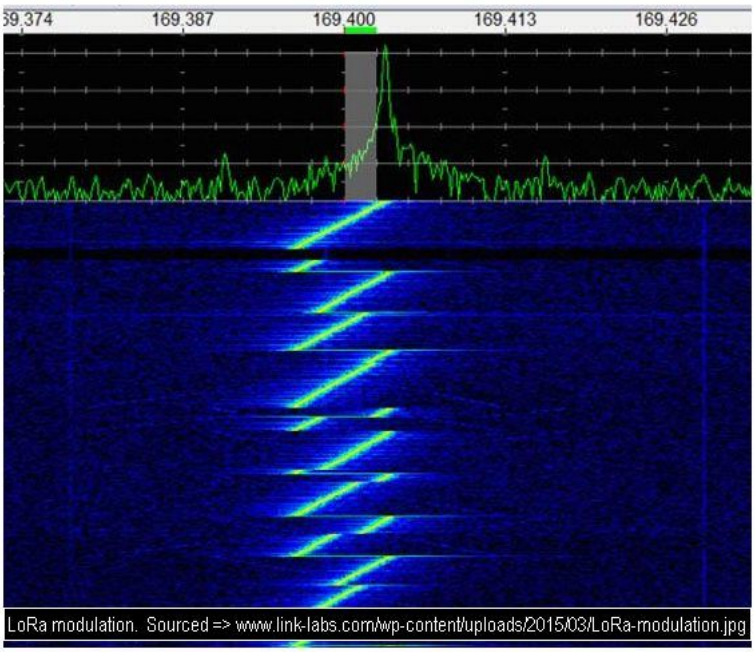

Abb.3 Wasserfalldiagramm mit Beispiel für das belegte Frequenzspectrum eines LoRa-Signals

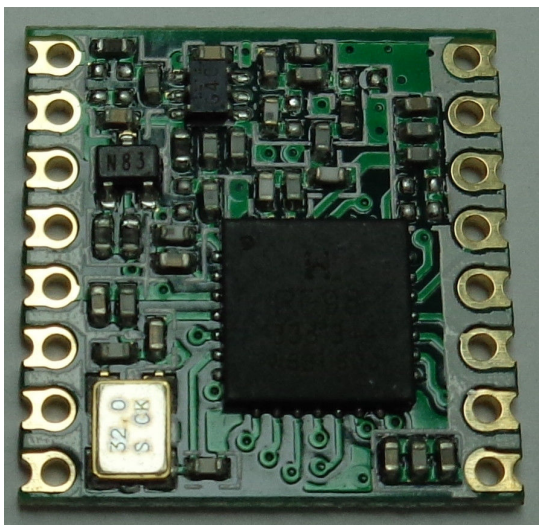

Abb.4 LoRa-Transceiver-Baustein "RFM98W" für den 433 MHz-Bereich [4]

SENDESEITE

Aus Abb.5 ist das Beispiel einer realisierten Sendeanordnung zur Übertragung von GPS-Navigationsdaten ersichtlich. Benutzt wird dabei ein Prozessor-Board des Typs ARDUINO "Pro Mini". Es gibt sie in einer 5V- und einer 3,3V-Version. Die Verwendung der 3,3V-Version hat dabei den Vorteil, dass sich die Anschaltung peripherer Baugruppen sehr einfach gestaltet, da diese mit identischen Signalpegelwerten arbeiten und es somit keine zusätzlichen Anpassungsprobleme gibt.
Zusätzlich zu den GPS-Daten erfolgt auch noch eine Übertragung der Bordspannungswerte. Nachdem die hierfür benutzbaren Analogeingänge des Prozessors nur eine Maximalspannung in Höhe der Versorgungsspannung (hier also 3,3V) erlauben, wird die Bordspannung mithilfe eines Spannungsteilers auf den halben Wert heruntergeteilt und dem Analogeingang "A1" zugeführt. Über derzeit noch unbenutzte Prozessorports sind auch noch Erweiterungen denkbar.

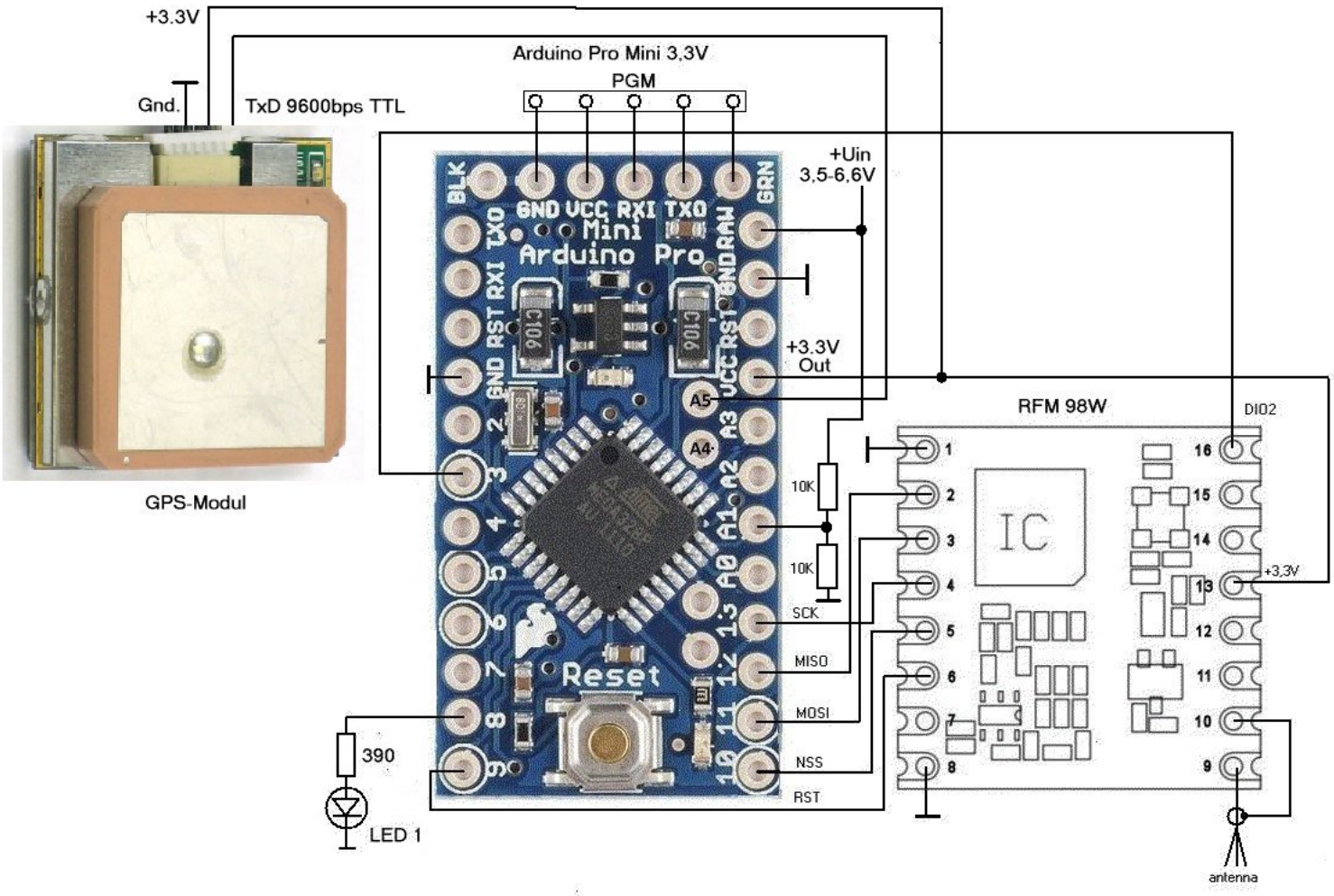

Abb.5 Schaltbild einer Anordnung zum Senden von GPS-Navigationsdaten und des Wertes der Versorgungsspanung

Funktion	ca. Strombedarf RFM98W
TX +20dbm (100mW)	120mA
TX +17dBm (50mW)	90mA
TX +13dbm (20mW)	30mA
TX + 7dbm (5mW)	20mA
RX	12mA

Tabelle1 ca. Strombedarf des RFM98W in Abhängigkeit vom Betriebsmodus

Tabelle1 listet die zu erwartenden Stromaufnahmewerte der benutzten Transceivereinheit. Wie erkennbar ist, ergeben sich dabei abhängig von der gewählten Sendeleistung erhebliche Unterschiede.
Bei Versorgung des RFM98W über den 3.3V-Ausgang des "ProMini's" ist auch die an dieser Stelle maximal verfügbare Belastbarkeit von ca. 135mA zu beachten. Ist zusätzlich auch noch ein GPS-Modul zu speisen und erfolgt die Gesamtversorgung z.B. aus einem 3.7V-Akku , so ist es ggf. sinnvoller, dieses Modul direkt mit dem Akku zu verbinden.

EMPFANGSSEITE

Zum Empfang via "LoRa" übertragener Navigationsdaten wurde eine Anordnung gem. Abb.6 aufgebaut. Wir erkennen wieder unser Prozessorboard "ProMini" in der 3,3V-Version und die mit ihm verbundene Transceivereinkeit "RFM98W". Zur Ausgabe empfangener Daten wurde ein über I2C-Bus angeschlossenes 0.96" OLED-Display verwendet. Es zeichnet sich durch seine relativ geringen Erstehungspreise bei einfacher Einsetzbarkeit aus. Es ist im Spannungsbereich von 3-5V verwendbar und sein Strombedarf wurde zu 8mA gemessen.
Folgendes gilt es an dieser Stelle allerdings noch zu beachten: Wer die Beschaltung der OLED-Displays in den Abb.6 und weiter unten auf dieser Seite in Abb.14 aufmerksam betrachtet, der wird die unterschiedliche Anordnung der Pins für Masse und Versorgungsspannug (+Vcc) bemerken, was offenbar herstellerabhängig ist. Dazu kann aus eigener Erfahrung noch gesagt werden, dass eine falsche Beschaltung zur sofortigen Zerstörung des Displays führen dürfte und es danach nur noch für den Mülleimer taugt (bei mir war es zumindest so !).
Nach jeder erfolgreichen Paketeinlesung wird die im Schaltbild sichtbare LED1 einmal kurz aufleuchten und ein ggf. angeschlossener Buzzer auch einen lauten Ton abgeben (deswegen ich ihm bald auch noch einen Ausschalter verpasst habe!). Hinsichtlich der Verwendung des in Abb.6 noch sichtbaren BLUETOOTH-Moduls siehe den folgenden Abschnitt.

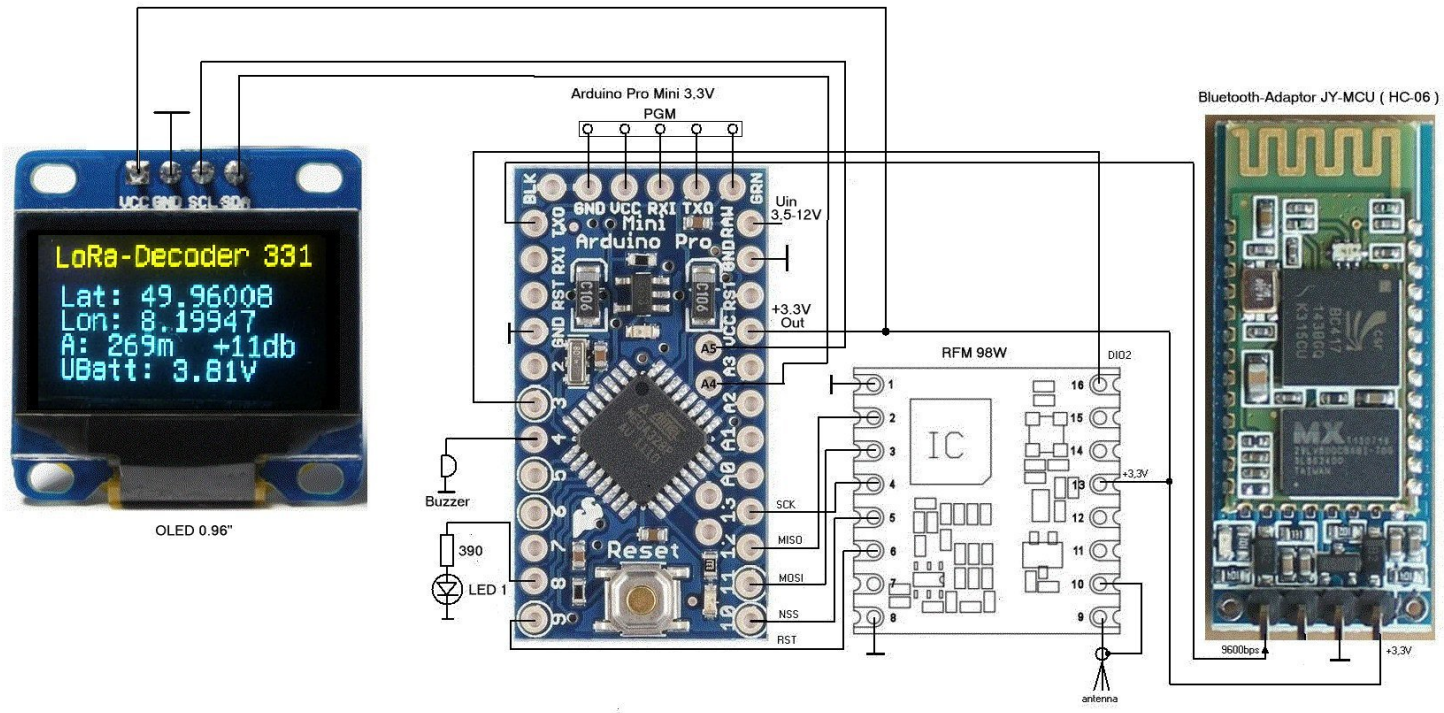

Abb.6 Schaltbild der Anordnung zum Empfang und zur Auswertung von GPS-Navigationsdaten. Zusätzlich erfolgt die Anzeige des Zählerstandes eines nach jeder erfolgreichen Datenpaketeinlesung inkremitierenden Upcounters (hier: Zählerstand 331) sowie dem SNR-Wert des Empfangssignals und der von der Sendeeinheit übertragenen Bordspannung.

NEU: Das LoRa-Empfangsprogramm wurde inzwischen auch noch in einer Version erstellt, mit der sich die via Funk von der Gegenseite empfangenen und die von einem auf der Empfängerseite angeschlossenen GPS-Modul stammenden Positionsdaten nutzen lassen, um daraus laufend Distanz- und Richtungswerte berechnen und anzeigen zu können. Zusätzlich werden auch wieder die Zählerstände eines Upcounters, SNR-Werte (Empfängerstörabstand), sowie Höhenwerte und Batteriespannung von der Sendeseite angezeigt (Abb.11a).

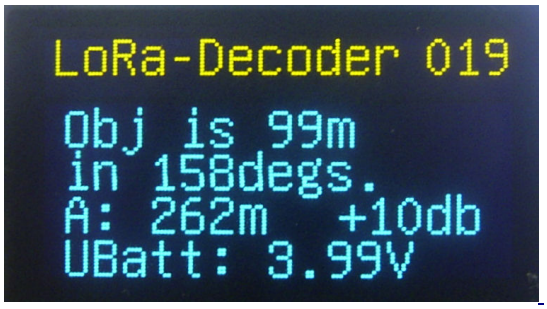

Abb.11a

EXTERNE AUSWERTUNG EMPFANGENER NAVIGATIONSDATEN MITHILFE VON (z.B.) ANDROID-SMARTPHONES

Zusätzlich zur Anzeige empfangener Navigationsdaten mithilfe des OLED-Displays (Abb.6), werden die Daten zudem noch als serieller ASCII-Datenstring aufbereitet und stehen (mit 9600bps) am seriellen UART-Ausgang des Prozessors zur Verfügung. Hierdurch ergibt sich die Möglichkeit, die darin enthaltenen Navigationsdaten auch noch mithilfe geeigneter Smartphone-APP's weiterzuverarbeiten, um sie bei Bedarf z.B. in Form von Positionsmarken auf Karten visualisieren zu können.
Um eine Datenverbindung zwischen LoRa-Empfangseinheit und Smartphone herstellen zu können, eignen sich preiswert erhältliche BLUETOOTH-Moduln, wie sie z.B. in Abb.6 zu sehen sind. Da im vorliegenden Fall nur das vom Prozessor abgehende serielle Datensignal übertragen werden muss, ergibt sich hierfür ein äußerst einfacher Anschluss mit nur drei Verbindungsadern.

Was die zur Kartenauswertung geeigneten APP's betrifft, so gibt es hierfür leider kein einheitlich akzeptiertes Eingangs-Datenformat. Daher musste ich die Empfängersoftware in mehreren Versionen und mit unterschiedlichem seriellem Ausgangs-Datenformaten erstellen.

Hier sei zuerst diejenige genannt, die das klassische APRS-Datenformat verwendet und z.B. eine Auswertung mithilfe der ANDROID-APP mit Namen "W2APRS" [11] zulässt. Die hierfür von der LoRa-Empfängersoftware aufbereiteten Ausgangsdaten sind (mit roter Umrandung) in Abb.7 zu sehen.
Was "W2APRS" betrifft, so sei noch darauf hingewiesen, dass es diese Software auch in einer OSM-Karten benutzenden Version gibt. Damit lassen sich vorher heruntergeladene interessierende Kartenausschnitte später auch wieder ohne noch bestehende Datenverbindung nutzen.

```
lora_Setup()
lora.ResetDev()
lora_Tone()
lora_PrintModem() 82 199 8

lora_RXtoReady()
Packet Ready
lora_RXPKTInfo() RXtype,60,Destination,255,Source,1,Length,26,RXCount,1,CR
RSSI -64dBm
Noise -137dBm
PacketSNR +9dB
lora_RXBuffPrint() 49.96012,8.20004,275,*
UNIT1>APRS:!4957.61N\00812.00Ec /A=275m SNR+9db
lora_RXtoReady()
```

Abb.7 Ausgangssdaten im APRS-Format; Beispiel für serielles Empfangsprotokoll am Monitor-Ausgang der ARDUINO-IDE

Abb.8 zeigt ein zugehöriges Screenprint, wobei wir erkennen können, dass neben den Kartenpositionen auch noch als Textzeile ankommende Rohdaten sichtbar sind, so dass auch mitübertragene Zusatzdaten wie Höhen- und SNR-Werte auswertbar sind.
Der benutzte Ident-Name "UNIT1" wird übrigens vorläufig nur als Festwert von der LoRa-Empfängersoftware eingefügt (und ist hier ebenso wie das gewählte Icon nach Belieben veränderbar). Für spätere Versionen ist aber auch die Übertragung und Auswertung unterschiedlicher Ident-Kennungen ausgehend bereits von der Sendeseite denkbar.

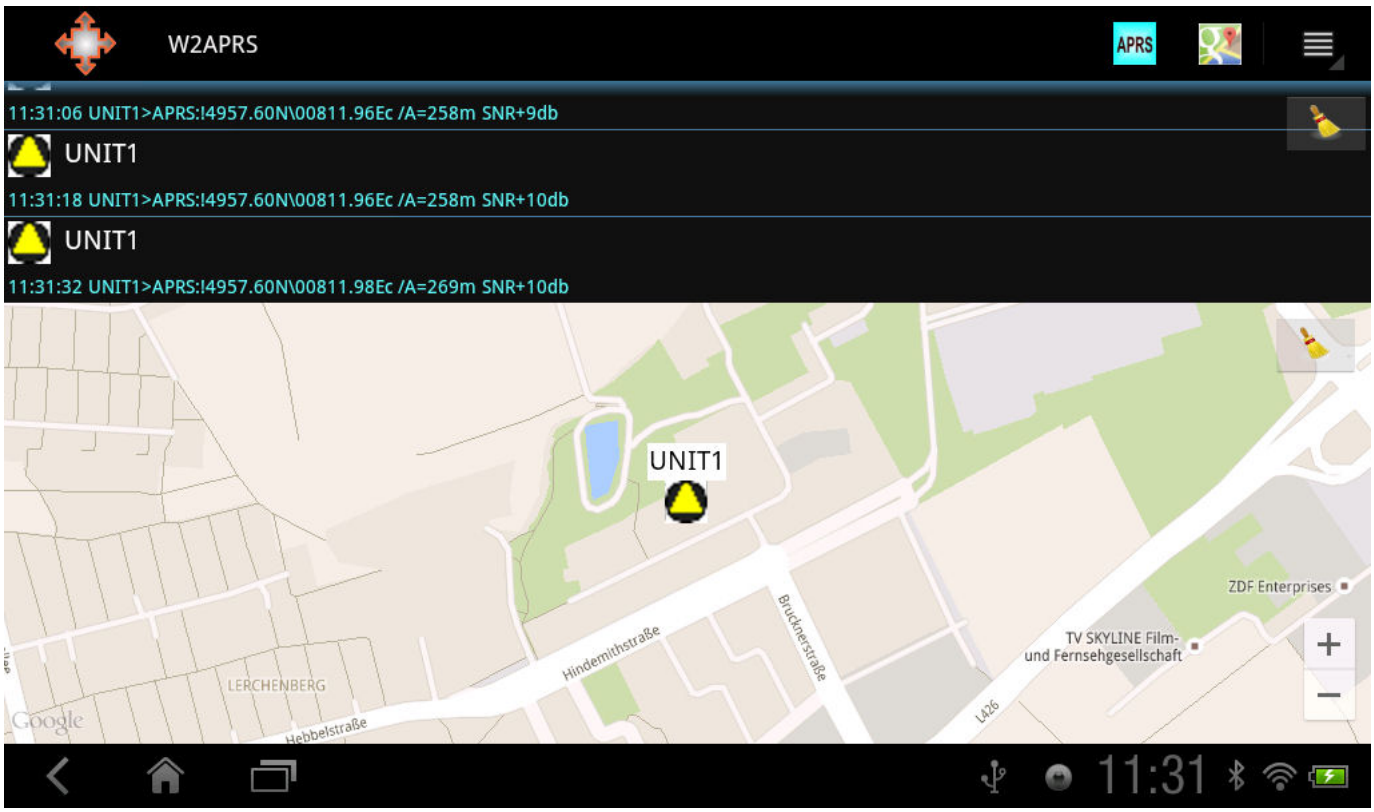

Abb.8 LoRa-Empfängerseite:
Beispiel für Auswertung via BLUETOOTH weiter geleiteter serieller GPS-Daten bei Verwendung eines Smartphones und ANDROID APP "W2APRS"

Eine andere Version der LoRa-Empfängersoftware stellt ihre seriellen Ausgangsdaten im WPL-Format bereit. Ein Protokollbeispiel ist (in roter Umrandung) aus Abb.9 ersichtlich. Es handelt sich hierbei um ein speziell (und nur) der Übertragung von Wegpunkdaten dienendes, sehr einach aufgebautes NMEA-Protokoll.

```
lora_RXtoReady()
NP.NP.NP.NP.NP.NP.NP.NP.NP.NP.NP.NP.Packet Read
lora_RXPKTInfo() RXtype,60,Destination,255,Sour
RSSI -59dBm
Noise -88dBm
PacketSNR +10dB
lora_RXBuffPrint() 49.96002,8.19966,262,4.04,*
49.96002
8.19966
262
4.04
$GPWPL,4957.6011,N,00811.9796,E,UNIT-1*51
lora_RXtoReady()
```

Abb.9 Ausgangsdaten im WPL-Format: Beispiel für serielles Empfangsprotokoll am Monitor-Ausgang der ARDUINO-IDE

Eine die Auswertung von WPL-Daten ermöglichende ANDROID-APP ist "Locus Map Pro" [12]. Während umfangreiche Funktionen auch schon von der kostenlosen Basisversion "Locus Map" abgedeckt sind, ist zur zusätzlichen Nutzung auch der WPL-Daten allerdings m.W. die wenige Euros kostende Pro-Version erforderlich.
Auch "Locus Map [Pro]" zeichnet sich durch die Möglichkeit zur bedarfsweisen Nutzung von Offline-Karten bzw. OSM-Karten aus.
Abb.10 zeigt ein Kartenbeispiel mit Darstellung der GPS-Position des ANDROID-Smartphones (Kreis) und der Senderposition (Flagge).

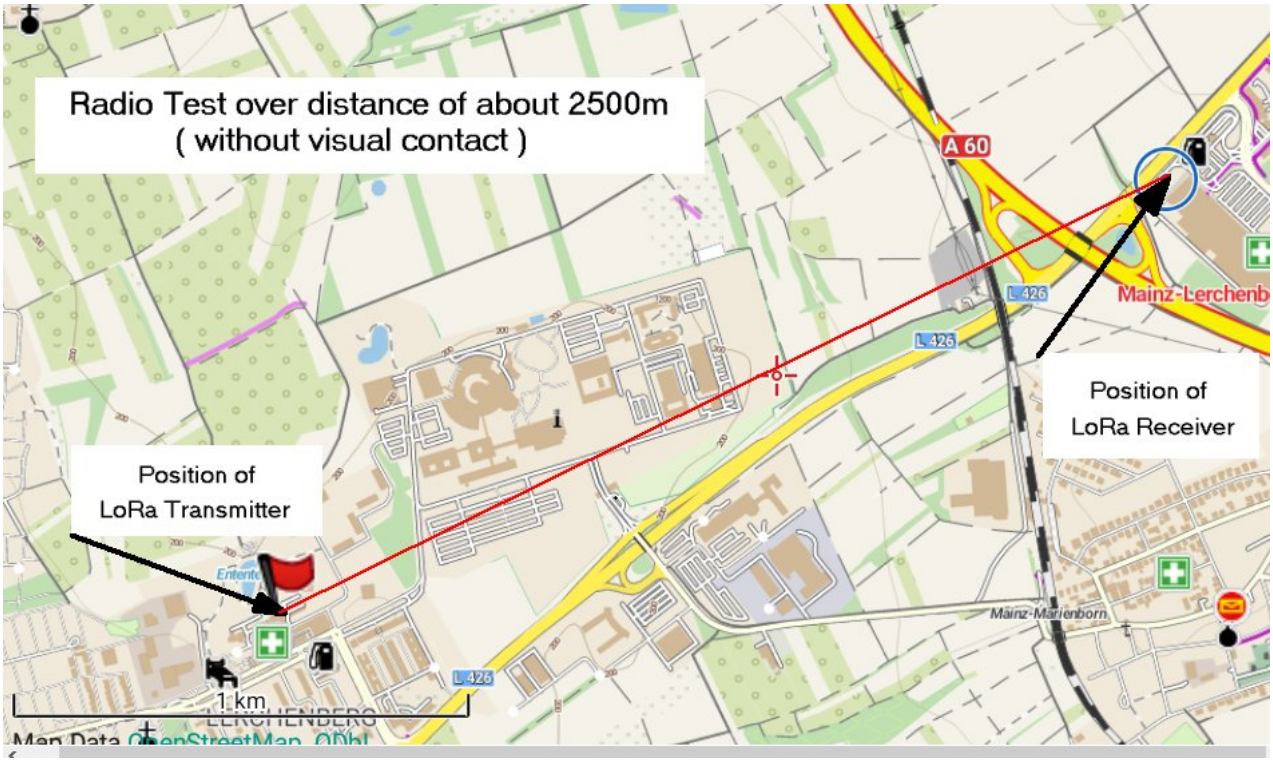

Abb.10 LoRa-Empfängerseite
Beispiel für Auswertung via BLUETOOTH weitergeleiteter serieller GPS-Daten bei Verwendung eines Smartphones und ANDROID APP "LOCUS MAP PRO"

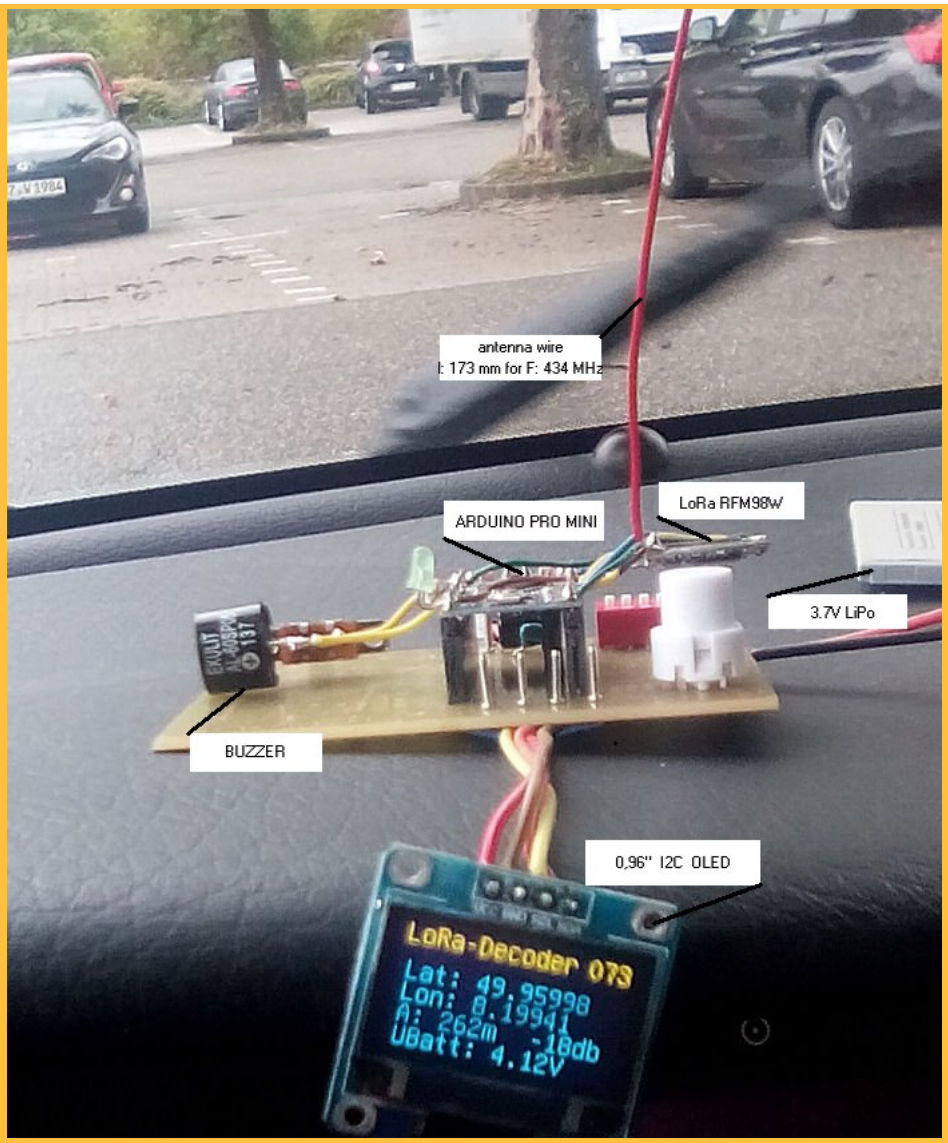

Abb.11 Empfangstest über ca. 2500m (ohne Sichtverbindung)
(erkennbar ist der untere Teil des als Empfangsantenne dienenden roten Drahtes mit einer Länge von ca. 17cm [etwa Lambda/4 für 434 MHz])

MUSTERAUFBAUTEN

Im deutschsprachigen Forum der fpv-community hatte ich noch einen Thread zum Thema "LoRa" gestartet [14]. Teilnehmer "QuadMax" hat darin freundlicherweise ein Layout für Platinen entworfen, die eine einfache Zusammenschaltung von ARDUINO ProMini's und RFM9xW-LoRa-Moduln ermöglichen. Mit entsprechenden Musterplatinen habe ich daraufhin je eine Sende- und Empfangseinheit (Abb.12, Abb.13) aufgebaut und inzwischen auch schon erste praktische Tests durchgeführt. Die dabei benutze Softwareversion dient vorerst nur der Übertragung von 4 Analogkanälen. An den Analogports "A0-A3" des auf der Sendeseite eingesetzten Prozessors anliegende Gleichspannungen im Bereich 0V bis +3,3V werden hierbei jeweils als Werte zwischen 0 und 1023 (10-Bit-AD-Wandlerauflösung) übertragen. Die Ergebnisse werden via OLED-Display angezeigt (Abb.14). Denkbar sind an dieser Stelle natürlich auch noch Programmerweiterungen mit Umrechnung empfangener Wandlerdaten entsprechend tatsächlicher Messwerte mit Hinzufügung von z.B. Namensbezeichnungen, Masseinheiten usw.
Ebenfalls angezeigt werden auch noch der Zählerstand eines Upcounters (000-999) und der SNR-Wert. Eine weiterhin realisierte Softwarevariante diente der Übertragung von beliebigen Textdaten (ASCII-Strings).

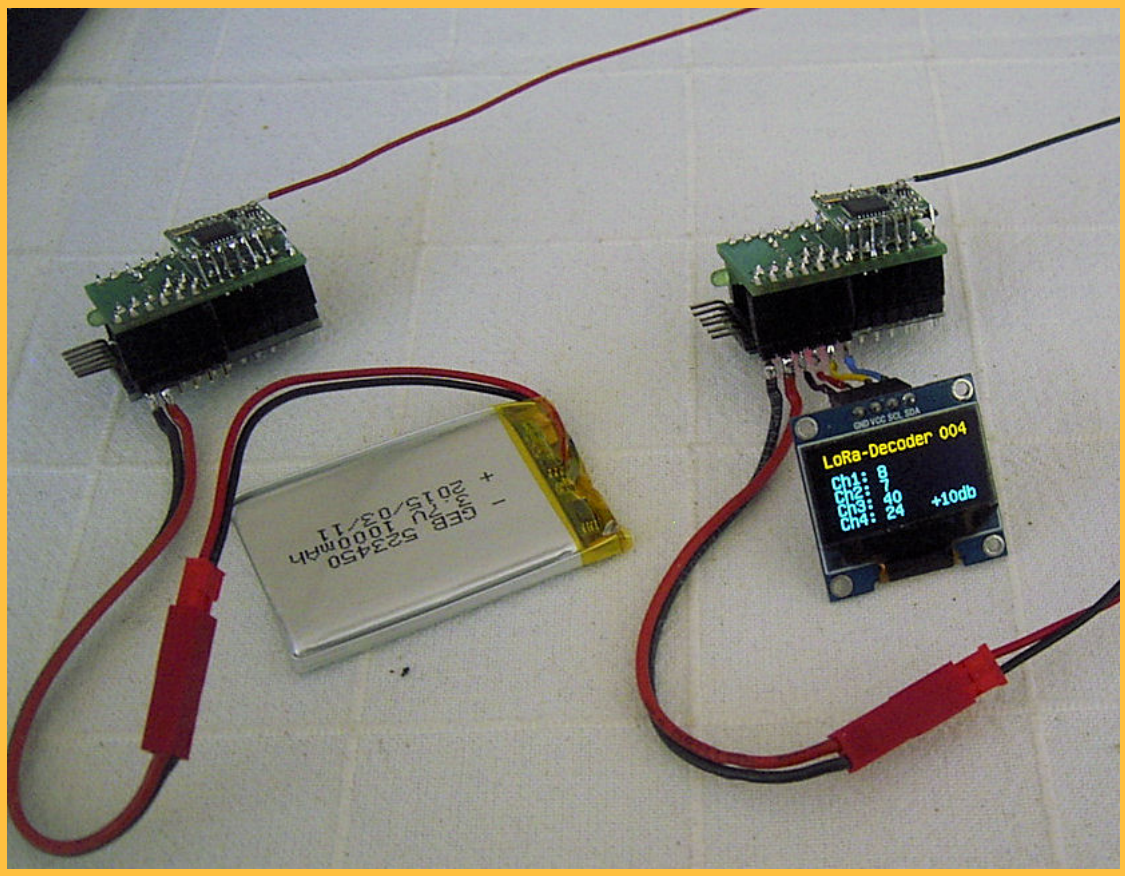

Abb.12 LoRa-Sende- und Empfangseinheit

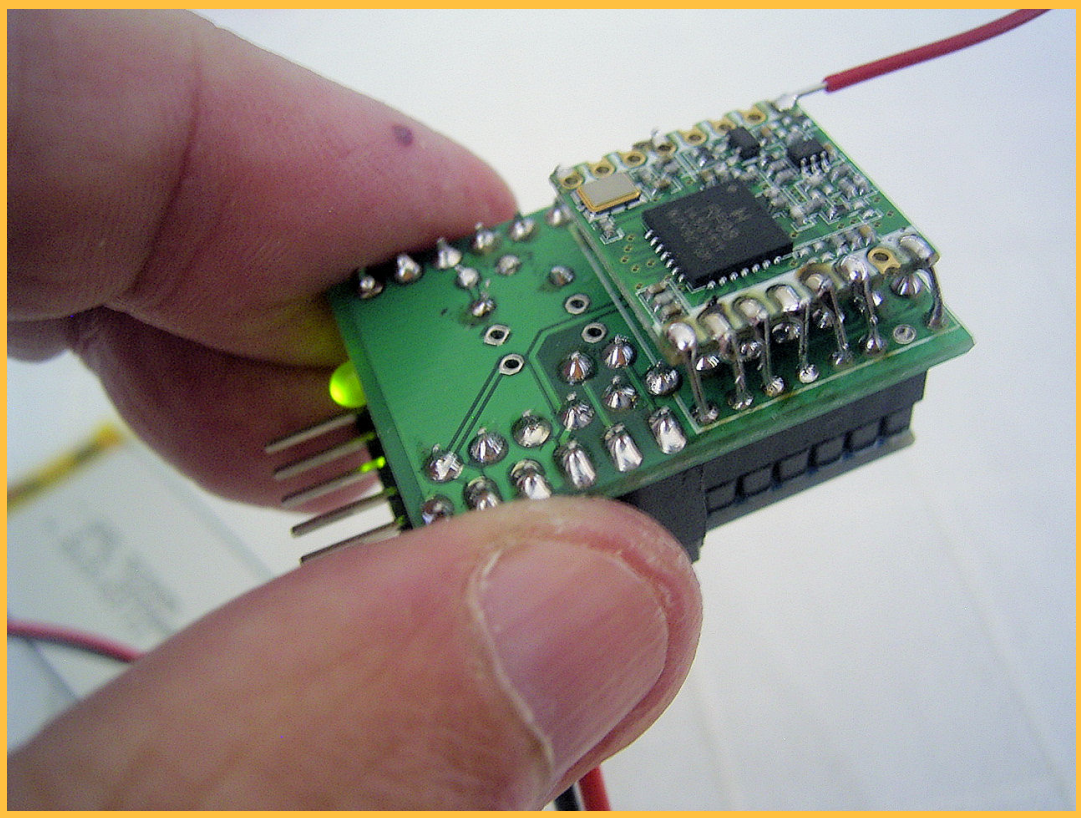

Abb.13

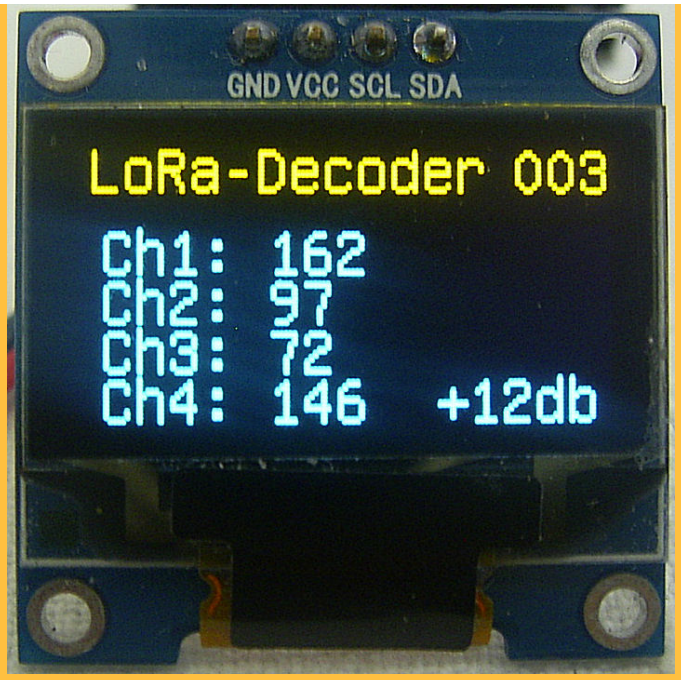

Abb. 14

AUSWERTUNG EMPFANGENER DATEN MITHILFE VON (ANDROID-) SMARTPHONES/TABLETS

In dem oben gezeigten Anwendungsbeispiel wurden via Funk empfangene Daten durch Darstellung am Display ausgewertet. Darüber hinaus werden sie aber zusätzlich auch noch über den seriellen Datenausgang (#D1) des Prozessors bereitgestellt um erlauben damit eine externe Verarbeitungsmöglichkeit. Soll hierzu z.B. ein Smartphone benutzt werden, so erfolgt eine Zusammenschaltung auf einfachste Weise via BLUETOOTH. Dabei lassen sich preiswert erhältliche Moduln (Reichweite bis zu ca. 15m) verwenden, so wie sie z.B. in Abb.15 zu sehen sind. Defaultmäßig (im Lieferzustand) sind sie bereits zur Nutzung einer Datenrate von 9600bps konfiguriert. Nachdem die gleiche Datenrate auch vom seriellen Ausgangssignal des benutzten ARDUINO-Prozessor benutzt wird, ist die Zusammenschaltung sehr einfach realisierbar und erfordert nur drei Verbindungsadern (Masse, Versorgungsspannung und Verbindung von Prozessorausgang "#D1" zum Dateneingang "RxIn" des Bluetoothmoduls.
Der Aufbau einer Bluetoothverbindung (Pairing) erfolgt auf die übliche Weise ausgehend vom verwendeten Smartphone/Tablet.

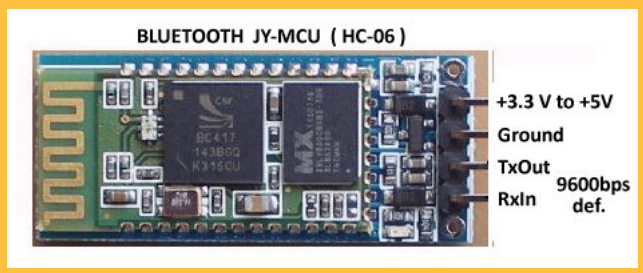

Abb.15. Bluetoothmodul

Für die Datenauswertung am Smartphone bzw. Tablet könnte man sich natürlich speziell hierfür erstellte Programme vorstellen. Im einfachen Fall genügen aber auch herkömmliche Terminalprogramme, so wie sie als APP's, z.B. aus dem GOOGLE "Play Store" herunterladbar sind. Die Abb.16 und Abb.17 zeigen entsprechende Einsatzbeispiele. Erkennbar ist dabei, dass zusätzlich zu den übertragenen Analogwerten auch noch der Zählerstand eines Upcounters und ein SNR-Wert ausgegeben werden. "SNR" steht dabei für "Signal to Noise" und ist ein Kriterium für die jeweilige Empfangsqualität. Bei Lora-Übertragungen liegen diese Werte zwischen etwa +14db (max. Wert) und -20db (min. Wert).

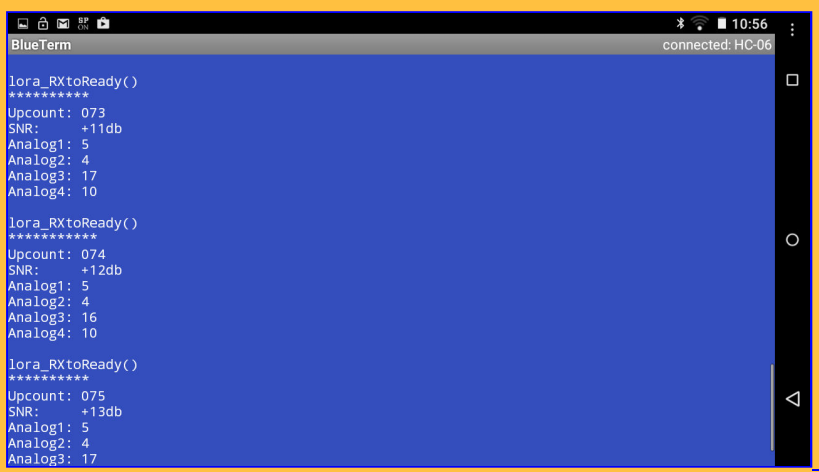

Abb.16 Screenprint von ANDROID-Smartphone/Tablet
Beispiel für Datenauswertung mithilfe von "BlueTerm" (zur Grossdarstellung anklicken)
Interessant ist dabei auch die bei Bedarf aktivierbare LOG-Funktion. Ankommende Daten werden danach auch in Dateien auf dem Smartphone abgelegt und stehen somit auch noch zur nachträglichen Auswertung zur Verfügung.

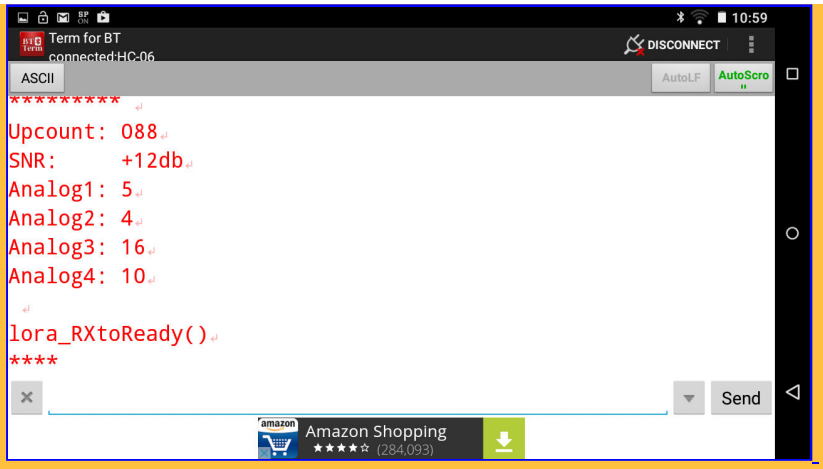

Abb.17 Screenprint von ANDROID-Smartphone/Tablet
Datenauswertung mithilfe von "S2 Terminal für Bluetooth" (zur Grossdarstellung anklicken)

NEU: LoRa mit ARDUINO-Shields von "DRAGINO"

Einen einfachen (wenn auch nicht unbedingt für Freunde der Miniaturisierung geeigneten) Einstieg in die LoRa-Praxis erlauben die neuen ARDUINO kompatiblen Shields von DRAGINO [19], [20]. Die Hardwarelösung besteht im einfachsten Fall nur darin, sie auf Boards wie z.B. "UNO" aufzustecken. Die Shields gibt es in Ausführungen für die Frequenzbänder um 433MHz, 868MHz und 915 MHz, wobei Letztere nicht für den Einsatz in Europa zugelassen sind. Zum Betrieb in Verbindung mit Prozessorboards, die mit 5V-Pegelwerten arbeiten (wie z.B. "UNO" u.ä.), sind die Shields mit entsprechenden Pegelwandlerbausteinen ausgestattet. Auch gibt es sie jeweils in Basisversionen und Varianten mit integrierten GPS-Empfängerbausteinen.

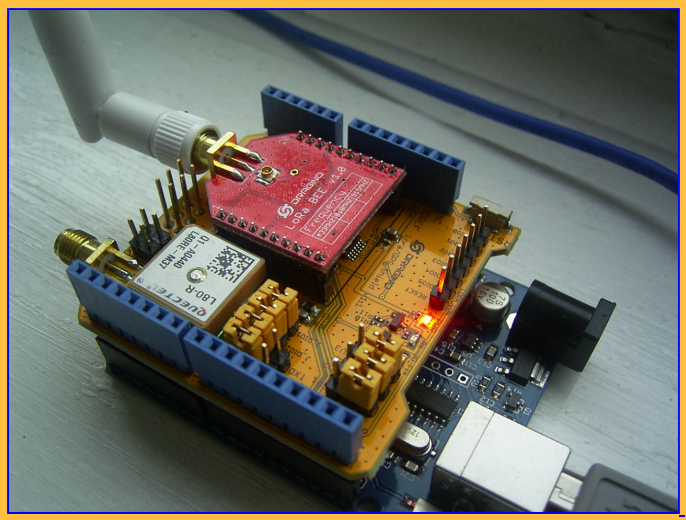

Abb.18 DRAGINO-Version des LoRa-GPS-Shields

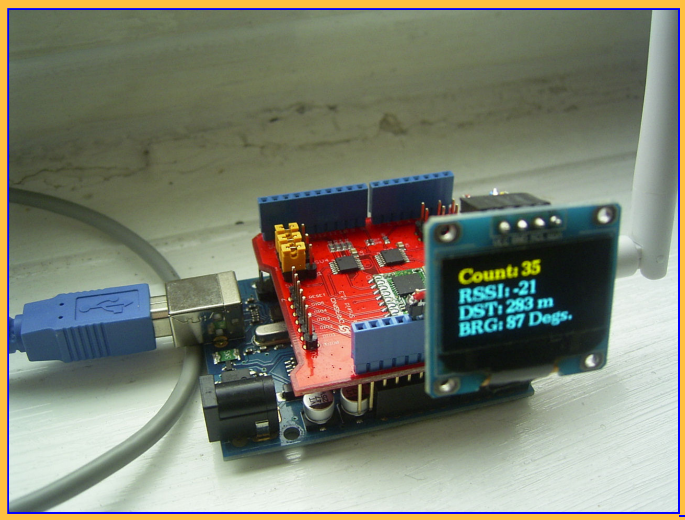

Abb.19 Basisversion des DRAGINO LoRa-Shields (hier mit zusätzlichem I2C-OLED-Display)

Für erste Versuche habe ich auf der Senderseite die DRAGINO-GPS-Version (Abb.18) und für die Empfangsseite die Basisversion verwendet. Dabei wurden die vom GPS-Empfängerbaustein auf der Senderseite stammenden, via LoRa übertragenen Daten von Längen- und Breitengrad benutzt, um daraus zusammen mit im Arduino-Quellcode abgelegten Fixdaten des Empfangsstandortes die Werte von Distanz und Richtung berechnen zu können. Zur Anzeige dient dabei ein (hier erst einmal nur provisorisch installiertes) 0.96"-I2C-OLED-Display, das zusätzlich auch noch Feldstärkewerte (RSSI) und den Stand eines nach jeder Empfangsdateneinlesung incrementierenden Zählers anzeigt. Wie wir dabei erkennen, ist ein solches Übertragungssystem auf einfache Weise realisierbar. Mehr über die beutzten Shields findet man auch in den zugehörigen Wikis [21], [22], während [23] eine mögliche inländische Bezugsquelle listet.

Anm: Obwohl sie für andere Projekte von dieser Seite schon häufig erfolgreich benutzt wurden, gab es an dieser Stelle Probleme beim Einsatz der entsprechenden Softwarebausteine, wobei den Ursachen noch nachzugehen sein wird. In Verbindung mit den DRAGINO-Bausteinen habe ich daher als Ausgangsbasis zumindest vorerst auf die bei ADAFRUIT [24] verfügbaren Beispieldaten incl. der passenden RadioHead-Library zurückgegriffen. Bezüglich Interesse an den von mir benutzten Quellcodes gilt auch hier wieder das im Abschnitt SOFTWARE zu findende.

Ein realisiertes Projekt:
LoRa-Funkübertragung von BME280-Sensordaten mithilfe von DRAGINO-Fertigbausteinen

---

Im Wesentlichen nur durch Zusammenstecken von ARDUINO-Shields wurde ein einfaches LoRa-System zur Funkübertragung von Daten realisiert, die von BOSCH-Sensoren des Typs BME280 stammen. Bei der Auswahl dieser winzigen Bausteine ist besonders die Verwendung der von Fa Watterott lieferbaren sog. Breakouts [25] zu empfehlen. Sie sind nicht nur einfach anschaltbar, sondern erlauben zudem auch ein einfaches Zusammenspiel mit an 5V betriebenen und entsprechenden Signalpegeln arbeitenden Boards. Die Sensoren wurden über eine I2C-Bus-Schnittstelle angeschlossen und liefern Daten von Temperatur, Luftdruck, Höhe und Luftfeuchte. Die Datenabfrage erfolgt mithilfe einer dazu im Prozessor eines ARDUINO-Boards des Typs "UNO" (o.ä.) abgelegten Software. Deren weitere Aufgabe besteht in der Steuerung der Aussendungen mithilfe aufgesteckter DRAGINO-LoRa-Boards [19],[21] .

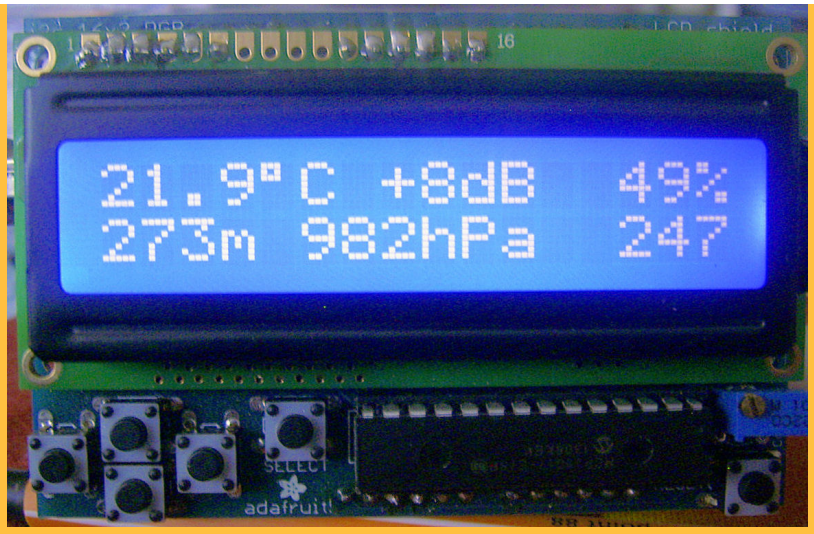

Abb.20 Displayanzeige auf der Auswertseite von Temperatur, Luftfeuchte, Höhe und Luftdruck. Weiterhin werden der Empfänger-Störabstand (SNR) und ein Upcounter angezeigt.

Die auswertende Seite besteht wiederum aus UNO-Boards und aufgesteckten DRAGINO LoRa-Shields. Zur LCD-Anzeige empfangener Daten ist zusätzlich noch ein weiteres Shield aufzustecken. Um Kollisionen mit den vom LoRa-Board benutzen Ports zu vermeiden, wurden an dieser Stelle jedoch keine Standard-LCD-Shields mit Parallelansteuerung benutzt, sondern eine Version mit I2C-Schnittstelle gewählt. Ein geeigneter Typ wird z.B. von Fa. ADAFRUIT unter Verwendung von Steuerbausteinen des Typs MCP23017 angeboten [26], [27]*.

*** Bei dem Kit (Bausatz: Art.715) nach [27] handelt es sich um eine Version OHNE mitgeliefertes Display (Standard-16x2 Zeichen-LCD-Displays waren hier problemlos verwendbar)**

SOFTWARE

Die von mir auf der Sende- und Empfangsseite benutzten ARDUINO-Sketches basieren auf Originalen, die via [8],[1],[2] von Stuart Robinson bereitgestellt wurden. Das schliesst auch die passenden Libraries ein. Seine Softwareversionen wurden von mir entsprechend meiner Wünsche und Vorstellungen geändert bzw. erweitert, wobei zukünftig auch noch weitere Anpassungen möglich sind.
Wer an entsprechenden Dateien interessiert ist, der sollte mir eine E-Mail schicken.

FREQUENZBERECHNUNGSBEISPIEL
FÜR DEN PROGRAMMCODE

*** gewünschte Arbeitsfrequenz sei: 434.400 MHz**
*** 434400000 [Hz] / 61.03515625 = 7117209,6**
*** Nachkommastelle ignorieren und Wert in Hexformat umrechnen:**
*** 7117209d = 6C9999h**
*** aus Hexwert 3x Zweiergruppe bilden;**
Einzelwerte wieder in Dezimalformat umrechnen:
*** 6Ch 99h 99h = 108d 153d 153d**
*** Werte in entsprechende Befehlszeile einfügen:**
*** lora_SetFreq(108, 153, 153);**

Anm.: Die sich mit den errechneten Endwerten ergebende Arbeitsfrequenz beträgt 434399963.4 Hz und liegt damit 36,6 Hz unterhalb des gewünschten Wertes. Für die praktische Nutzung kann diese Abweichung allerdings bedenkenlos ignoriert werden. Generell erlaubt die benutzte Adressierungsmethode Frequenzschritte im Abstand von 61Hz.

SingnalBandWidth	SpreadingFactor	Sensitivity(dbm)	ActualBandRate(pbs)
62.5kHz	SF=7	-126	2169
62.5kHz	SF=8	-129	**1187**
62.5kHz	SF=9	-132	**656**
62.5kHz	SF=10	-135	**296**
62.5kHz	SF=11	-137	**164**
62.5kHz	SF=12	-139	**91**
125kHz	SF=7	-123	4338
125kHz	SF=8	-126	2375
125kHz	SF=9	-129	1312
125kHz	SF=10	-132	733
125kHz	SF=11	-133	328
125kHz	SF=12	-136	183
250kHz	SF=7	-120	8676
250kHz	SF=8	-123	4750
250kHz	SF=9	-125	2624
250kHz	SF=10	-128	1466
250kHz	SF=11	-130	778
250kHz	SF=12	-133	366
500kHz	SF=7	-118	17353
500kHz	SF=8	-121	9501
500kHz	SF=9	-124	5249
500kHz	SF=10	-127	2932
500kHz	SF=11	-129	1557
500kHz	SF=12	-130	830

Die einzelnen LoRa-Parameter und ihr Zusammenhang in der Übersicht.

LINKLISTE

[1] http://www.instructables.com/id/Introducing-LoRa-/
[2] http://www.rcgroups.com/forums/showthread.php?t=2454546
[3] http://www.rcgroups.com/forums/showthread.php?t=2341299
[4] http://www.hoperf.com/upload/rf/RFM95_96_97_98W.pdf
[5] http://www.southgatearc.org/news/2015/january/lora_low_cost_long_distance_telemetry.htm#.Veg29ZakViY
[6] http://hackaday.com/2014/11/16/the-future-of-the-internet-of-things/
[7] http://www.picaxeforum.co.uk/showthread.php?26314-Semtech-s-LoRa-(Long-Range)
[8] https://www.dropbox.com/sh/na2d9yt09nr91f7/AADqSNprExezpBEcqIxewDpBa?dl=0
[9] http://ava.upuaut.net/store/index.php?route=product/product&path=71_63&product_id=110
[10] https://groups.google.com/forum/#!searchin/ukhas/lora%7Csort:date
[11] http://www.kh-gps.de/w2aprs.htm
[12] http://www.kh-gps.de/locus_pro.htm
[13] http://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Allgemeinzuteilungen/2014_69_SRD
[14] http://fpv-community.de/showthread.php?67651-Wer-kennt-quot-LoRa-quot-Long-Range-Telemetrie
[15] http://www.semtech.com/wireless-rf/lora.html
[16] http://www.semtech.com/wireless-rf/lora/LoRa-FAQs.pdf
[17] http://www.wireless-solutions.de/images/imst/pdf/Markt_Technik_WAN-Technology_for_IoT_06-2015.pdf
[18] http://www.funkschau.de/telekommunikation/artikel/117470/
[19] http://www.dragino.com/products/module/item/102-lora-shield.html
[20] http://www.dragino.com/products/module/item/108-lora-gps-shield.html
[21] http://wiki.dragino.com/index.php?title=Lora_Shield
[22] http://wiki.dragino.com/index.php?title=Lora/GPS_Shield
[23] http://www.exp-tech.de/hersteller/dragino
[24] https://learn.adafruit.com/adafruit-rfm69hcw-and-rfm96-rfm95-rfm98-lora-packet-padio-breakouts/rfm9x-test
[25] http://www.watterott.com/de/BME280-Breakout-Luftfeuchtigkeits-Druck-Tempertursensor
[26] https://www.adafruit.com/product/714
[27] https://www.adafruit.com/product/715
[28] http://www.kh-gps.de/lora_schalt2.htm
[29] http://www.kh-gps.de/lora_rel.htm
[30] http://www.kh-gps.de/lora_bt.htm
[31] http://www.iot4pi.com/de/raspberry-pi-projekte-software/
[32] http://www.iot4pi.com/de/hardware/lora-gateway/
[33] http://www.kh-gps.de/loratext.htm
[34] http://wiki.oevsv.at/index.php?title=Was_ist_HAM-IoT
[35] http://www.dh4ym.de/

NACHTRAG:
Hier noch einige m.E. interessante Links zum Thema:
http://syncchannel.blogspot.de/2016/02/lora-featherwing-development-breakout.html
http://thethingsnetwork.org/wiki/Hardware/OverviewNodes
https://oshpark.com/shared_projects (hier nach "LoRa" suchen)
http://cpham.perso.univ-pau.fr/LORA/LoRaDevices.html

E-Mail contact via: dj7oo@t-online.de