

Das LILYGO® TTGO T-Beam V1.1 ESP32 LORA 433/868/915/923MHZ WiFi Wireless Bluetooth Module GPS NEO-M8N IPEX 18650 Battery Holder module kommt schon mit einem GPS-Modul auf der Basisplatine.

Der Stromverbrauch liegt bei 10...14mA bei normalem Betrieb und im Sendemodus steigt der Stromverbrauch auf ca. 120mA

Der Temperaturbereich wird von -40Grad Celsius bis +85Grad Celsius angegeben.

**Allerdings bietet dieses Modul kein I2C Interface an, nur SPI**

### 2.1.5.1 GPS Empfangsfrequenz

GPS-Systeme senden ihre Signale auf drei verschiedenen Frequenzen: auf 1.575,42 MHz werden GPS-Signale für zivile Nutzung gesendet, auf 1.57542 GHz werden Präzisionssignale für militärische Nutzung gesendet und auf 1.176,45 MHz Signale für die Luftfahrt.

### 2.1.5.2 GPS Empfangsdaten - Erfahrungen

Im Innenraum ist nahezu kein GPS-Empfang mit diesem Modul zu bekommen. Unmittelbar am Fenster funktioniert dann der GPS-Empfang einigermaßen.

Im Auto hat der GPS-Empfang mit dem Modul auch gut funktioniert.

Die Daten vom GPS-Modul werden über die serielle Schnittstelle gelesen.

Die empfangenen Daten habe eine NMEA Kennung

<https://www.rfwireless-world.com/Terminology/GPS-sentences-or-NMEA-sentences.html>

NMEA Sentence	Meaning
PGGA	Global positioning system fix data (time, position, fix type data)
PGLL	Geographic position, latitude, longitude
GPVTG	Course and speed information relative to the ground
GPRMC	Time, date, position, course and speed data
PGPSA	GPS receiver operating mode, satellites used in the position solution, and DOP values.
PGPSV	The number of GPS satellites in view satellite ID numbers, elevation, azimuth and SNR values.
PGMSS	Signal to noise ratio, signal strength, frequency, and bit rate from a radio beacon receiver.
GPTRF	Transit fix data

GPSTN	Multiple data ID
GPXTE	cross track error, measured
GPZDA	Date and time (PPS timing message, synchronized to PPS).
150	OK to send message.

Eigentlich sind die Daten mit der \$GPGGA Kennung die relevanten Daten für die GPS Ortung, allerdings kommt es bei schlechtem GPS Empfang vor, dass diese Daten nur sehr selten übertragen werden, so war zumindest meine Erfahrung.

Die \$GPRMC Kennung liefert auch die Longitude und Latitude Werte allerdings keine altitude Werte und der Daten stream von der seriellen Schnittstelle war immer wieder zerstückelt. Die Kennung \$GPRMC wurde aber häufiger übertragen als \$GPGGA Kennung.

Im esp32 Programm habe ich dann einen einfachen Check eingebaut, um zu erkennen, ob es ein brauchbarer Datenstream ist oder nicht. Da gibt es sicherlich ausgereiftere Methoden. Ich schaue einfach ob an der vorgesehenen Stelle ein N oder S und ein W oder E (North South und West East Kennung steht) wenn das der Fall ist, verwende ich die Daten.

Hier Daten die ich vom GPS Modul im esp32 empfangen habe:

```
read-sentence: $GPGGA,145046.00,,,,,0,03,2$GPRMC,145053.00,V,,,1,32,12,24,,,,,,27.84,27.82,1.00*03
read-sentence: $GPGSV,3,1,12,06,20,079,17,10,0$GPRMC,145106.00,A,48049,15,22,14,322,10*7B
read-sentence: $GPGSV,3,3,12,24,60,134,26,25,49,261,26,29,08,199,,32,36,299,29*7C
read-sentence: $$GPRMC,145119.00,A,48$GPRMC,145126.00,A,4840.75970,N,00854.18731,E,0.212,,301222,,,A*75
read-sentence: $GPVTG,,T,,M,0.212,N,0F
read-sentence: $GPGSV,3,1,12,06,20,079,16,10,01,262,,11,11,115,19,12,81,289,27*72
read-sentence: $GPG$GPRMC,145139.00,A,4840.75946,N,00854.18915,E,0.227,,301222,,,A*70
read-sentence: 1,049,19,22,14,322,15*72
read-sentence:
$GPGSV,3,3,12,24,60,134,26,25,49,261,24,29,08,199,,32,36,298,29*$GPRMC,145152.00,A,4840.7$GPRMC,145159.00,A,48
40.75857,N,00854.19029,E,0.219,,301222,,,A*7D
read-sentence: 0F
read-sentence: $GPGSV,3,1,12,06,20,079,15,10,01,262,,11,11,115,07,12,81,289,26*7F
read-sentence: $GPRMC,145212.00,A,4840.7500854.19071,E,145218.00,A,A*6A
read-sentence: $GPRMC,145219.00,A,4840.75937,N,00854.19002,E,0.219,,301222,,,A*74
read-sentence: $GPVTG,,T,,M,0.219,N,0.406,K,A*2B
read-sentence: $GPGGA,145219.00,4840.$GPR$GPRMC,145232.00,A,4840.75640,N,00854.18642,E,0.507,,301222,,,A*79
```

read-sentence: 09

read-sentence:

\$GPGSV,3,1,12,06,21,078,12,10,00,261,,11,12,115,21,12,\$GPRMC,145245.00,A,4840.75147,N,00854.18115,E,1.998,,30122,  
2,,,A\*77

read-sentence: \$GPRMC,145252.00,A,4840.75116,N,0085258.00,4840.75307,N,00854.18284,E,1,04,3.61,435.8,M,47.5,M,,\*5F

read-sentence: \$GPGSA,A,3,32,25,12,24,,,,\$GPR03

read-sentence: \$GPGSV,3,1,12,06,21,078,,10,00,261,,11,12,115,21,12,82,292,28\*79

read-sentence:

\$GPGSV,3,2,12,15,01,177,,17,03,035,,19,30,048,18,22,\$GPRMC,145318.00,A,4840.75508,N,00854.18629,E,0.201,,301222,,  
A\*73

read-sentence:

\$GPVTG,,T,,M,0.201,N,0.371,K,A\*25\$GPRMC,145325.00,A,4840.75546,N,00855331.00,4840.75620,N,00854.18856,E,1,04,3.  
64,428.1,M,47.5,M,,\*54

read-sentence: \$GPRMC,145338.00,A,4840.75695,N,00854.18943,E,0.042,,301222,,,A\*70

read-sentence:

\$GPGSV,3,1,11,06,21,077,17,11,13,114,,12,\$GPRMC,145517.00,A,4840.75767,N,00854.19229,E,0.445,,301222,,,A\*72

read-sentence: \$GPVTG,,T5,321,23,24,58,135,20\*76

read-sentence:

\$GPGGA,145623.00,4840.75457,N,00854.19142,E,1,04,3.79,420\$GPR\$GPRMC,145636.00,A,4840.75410,N,00854.19084,E,  
0.136,,301222,,A\*75

read-sentence: \$GPRMC,145643.00,A,4840.75415,N,00854.18925,E,0.305,,301222,,,A\*73

Die Daten müssen so aufbereitet werden, dass die Position sich daraus ableiten lassen kann.

Wie aus dem Trace zu erkennen ist, wird die Kennung \$GPRMC häufiger gesendet als \$GPGGA

Von ca 600 records wurden ca. 300 mal die Kennung \$GPRMC empfangen und ca. 60 mal die Kennng  
\$GPGAA

Ein brauchbarer \$GPGAA Datenstream sieht so aus:

\$GPGGA, 161229.487, 3723.2475, N, 12158.3416, W, 1, 07, 1.0, 9.0, M, , , 0000\*18

Name or Field	Example	Description
Message ID	\$GPGLL	GLL protocol header
Latitude	3723.2475	ddmm.mmffff

N/S indicator	N	N =North or S = south
Longitude	12158.3416	dddmm.mmmm
E/W indicator	W	E =East or W = West
UTC time	161229.487	hhmmss.sss
Status	A	A = data valid or V = data not valid
Mode	A	A =Autonomous , D =DGPS, E =DR (This field is only present in NMEA version 3.0)
Checksum	*41	
<CR><LF>		End of message termination

**Ein brauchbarer \$GPRMC**

\$GPRMC, 161229.487, A, 3723.2475, N, 12158.3416, W, 0.13, 309.62, 120598, , \*10

Name or Field	Example	Description
Message ID	\$GPRMC	RMC Protocol Header
UTC time	161229.487	hhmmss.sss
Status	A	A = data valid or V = data not valid
Latitude	3723.2475	ddmm.mmmm
N/S indicator	N	N = North or S = South
Longitude	12158.3416	dddmm.mmmm
E/W indicator	W	E = East or W = West
Speed over ground	0.13	knots
Course over ground	309.62	degrees
Date	120598	ddmmyy
Magnetic Variation		Degrees (E= East or W = West)
Mode	A	A = Autonomous, D = DGPS, E =DR
Checksum	*10	

&lt;CR&gt;&lt;LF&gt;

End of message termination

\$GPGLL Kennung enthält auch die Daten für Longitude und Latitude

### 2.1.5.3 Genauigkeit der GPS Ortung mit dem TTGO T-Beam

Ich hatte das TTGO Board am Fenster (Büro) liegen und da ist der GPS Empfang schwach und da habe ich die Koordinaten mal auf der Karte dargestellt

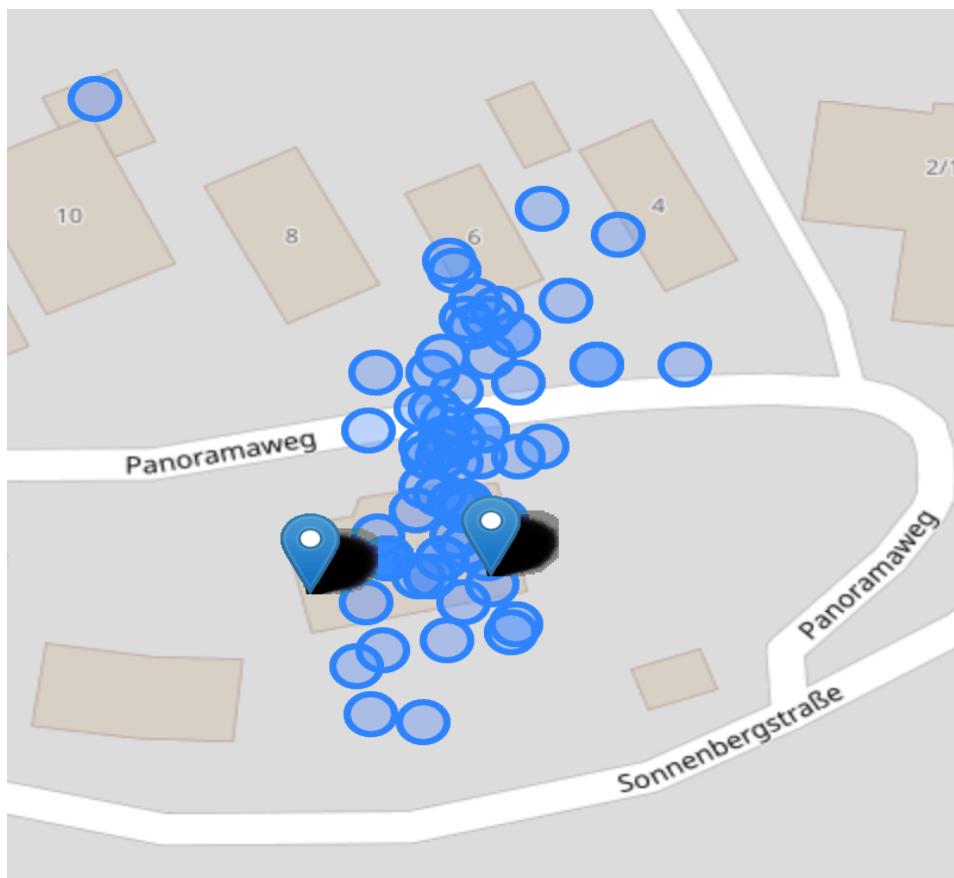


Figure 8: GPS Daten -Genauigkeit bei mäßigem GPS Empfang

Wie hier zu erkennen ist, ist die Abweichung GPS Ortung schon zig Meter! (Das abgestellte Haus ist ca. 12m lang, als Maßstabsvergleich.

Derselbe Versuch, jedoch liegt dieses Mal das TTGO Board im Freien und der GPS Empfang ist da besser und auch die Ortung ist besser

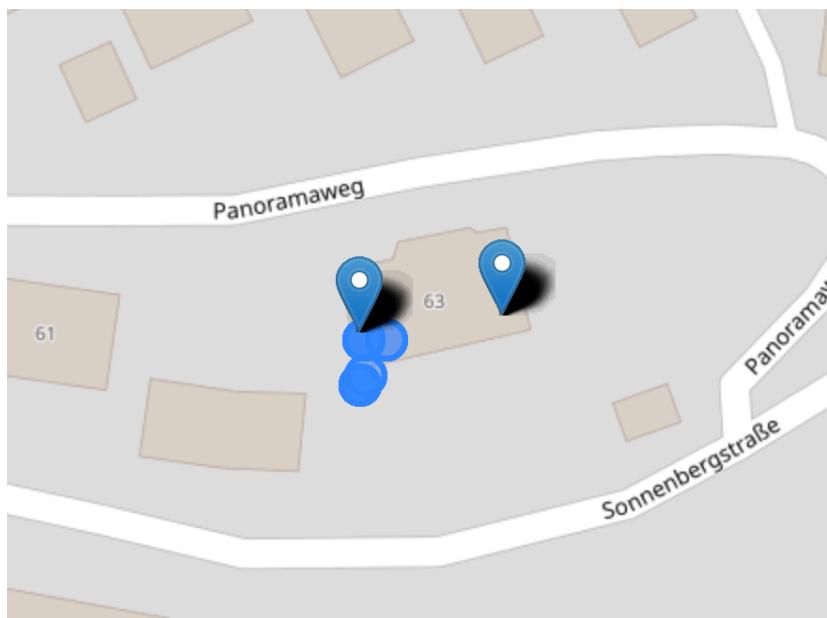


Figure 9: GPS Daten -Genauigkeit bei besserem GPS Empfang

#### 2.1.5.4 Adafruit Ultimate GPS Breakout board PA16116S

The breakout is built around the MTK3339 chipset, a no-nonsense, high-quality GPS module that can track up to 22 satellites on 66 channels, has an excellent high-sensitivity receiver (-165 dBm tracking!), and a built-in antenna. It can do up to 10 location updates a second for high speed, high sensitivity logging, or tracking. Power usage is incredibly low, only 20 mA during navigation.

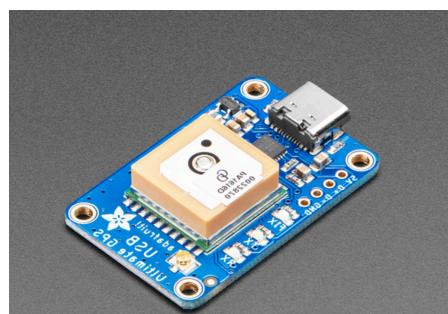


Figure 10: Adafruit GPS Modul

It can be powered with 3,3V...5V

An advantage of this board is the possibility to add an active antenna to the GPS Module

- GPS + GLONASS supported
- Satellites: 33 tracking, 99 searching
- Patch Antenna Size: 15mm x 15mm x 4mm
- Update rate: 1 to 10 Hz

- Position Accuracy: < 3 meters (all GPS technology has about 3m accuracy)
- Velocity Accuracy: 0.1 meters/s
- Warm/cold start: 34 seconds
- Acquisition sensitivity: -145 dBm
- Tracking sensitivity: -165 dBm
- Maximum Velocity: 515m/s
- Vin range: 3.0–5.5VDC
- MTK3333 Operating current: 34mA acquisition, 29 mA tracking
- Output: NMEA 0183, 9600 baud default, 3V logic level out, 5V-safe input
- DGPS/WAAS/EGNOS supported
- AGPS support (Offline mode : EPO valid up to 14 days )
- Up to 210 PRN channels
- Jammer detection and reduction
- Multi-path detection and compensation

Um die Daten vom GPS breakout board zum Controller (ESP32) zu übertragen steht das Serielle Interface zur Verfügung und die entsprechenden Libraries von adafruit.

Sollten die GPS Daten sowohl über LoRaWAN übertragen werden als auch über Amateurfunk, so würde es sich anbieten zwei GPS Module zu verwenden.

## 2.2 Umweltdaten - Sensoren zu messende Daten

### 2.2.1 Temperatur

Der Messbereich der Temperatursensor sollte auch Temperaturen um die -50Grad Celsius abdecken. Die Temperatur sollte auch außerhalb des Gehäuses gemessen werden.

Wenn der Temperatursensor eine I2C Schnittstelle bietet, würde das den Anschluss an einen Controller vereinfachen.

#### 2.2.1.1 Bosch BME680 Humity/Pressure/Temperature

Der Bosch BME680 Sensor beinhaltet 3 Sensoren und liefert folgende Messwerte

##### Technische Daten

- digitale Schnittstellen: I2C, SPI
- Betriebsspannung: 3 ... 5 V
- kompatibel mit Arduino, Raspberry Pi, etc.
- Maße: 30 x 14 x 10 mm

##### Luftfeuchtigkeitssensor

- Reaktionsgeschwindigkeit: 8 s
- Toleranz: ± 3%
- Hysterese: 1.5%

##### Drucksensor

- Druckbereich: 300 ... 1100 hPa

- relative Genauigkeit:  $\pm 0.12 \text{ hPa}$
- absolute Genauigkeit:  $\pm 1 \text{ hPa}$

### **Temperatursensor**

- Arbeitsbereich: -40 ... +85 °C
- vollständige Genauigkeit: 0 ... 65 °C

### **Luftgütesensor**

- Reaktionsgeschwindigkeit: 1 s

Alternativ kann auch der DS18S20 eingesetzt werden (One Wire Interface) allerdings sind meine Erfahrungen im Zusammenhang mit ESP32 und LoraWAN nicht so gut, da das One Wire Interface nicht mehr zuverlässig funktioniert, wenn LoraWAN verwendet wird. Irgendwie kommt dann das Taktsignal für das one wire interface nicht mehr konstant.

Der DS18S20 soll Temperaturen von -55 bis +125Grad messen.

## **2.2.2 Luftdruck/Pressure**

Der zu erwartende Luftdruck wird sich im Bereich von ca.

- 1000hPa (Meereshöhe)
- 280hPa in 10km Höhe
- 100hPa in 17km Höhe
- 78 hPa in 20km Höhe
- 10hP in 35km Höhe und
- 0,16hPa in 50km Höhe

bewegen.

Quelle: <https://de.wikipedia.org/wiki/Luftdruck>

D.h. der BME680 wird nur einen Bereich bis zu 10km Höhe abdecken können.

Alternative MS5803-01BA Temperatur und Luftdrucksensor

### **2.2.2.1 MS5803 Sensor**

The MS5803-01BA is a new generation of high resolution altimeter sensors from TE Connectivity with SPI and I2C bus interface.

<b>Sensor Performances (<math>V_{DD} = 3</math> V)</b>				
<b>Pressure</b>	<b>Min</b>	<b>Typ</b>	<b>Max</b>	<b>Unit</b>
Range	10		1300	mbar
ADC	24			bit
Resolution (1)	0.065 / 0.042 / 0.027 / 0.018 / 0.012			mbar
Accuracy 25°C, 750 to 1100 mbar	-1.5		+1.5	mbar
Accuracy -20°C to + 85°C, 300 to 1100 mbar (2)	-2.5		+2.5	mbar
Response time	0.5 / 1.1 / 2.1 / 4.1 / 8.22			ms
Long term stability		-1		mbar/yr
<b>Temperature</b>	<b>Min</b>	<b>Typ</b>	<b>Max</b>	<b>Unit</b>
Range	-40		+85	°C
Resolution		<0.01		°C
Accuracy	-0.8		+0.8	°C
Notes: (1) Oversampling Ratio: 256 / 512 / 1024 / 2048 / 4096 (2) With autozero at one pressure point				

Figure 11:Schaubild Kommunikationskanal via Amateurfunk

[Datenblatt MS5803-01BA](#)

### 2.2.3 O2 – Sauerstoffgehalt

Sauerstoff Gas Sensor O2 Sensor 70XV-CiTiceL (Aliexpress z.B.)

### 2.2.4 Luftfeuchtigkeit

Hier kann auch der BME680 eingesetzt werden oder *Si7021-A10-GM1*

### 2.2.5 Radioaktivität

tbd

## 2.3 Sensordaten erfassen und übertragen

Das Erfassen und das Übertragen der Sensordaten wir hier mal separat betrachtet, da für hierfür wahrscheinlich auch eine separate Infrastruktur (Controller) bereitgestellt werden sollte. Übertragung via LoraWAN z.B.

## 2.4 Bildinformationen

The watering system should have two modes of operations.

The learning phase, the soil moisture should be monitored by using the soil moisture sensor and a microprocessor. The measured values should be transferred to a server application.

The operating phase, depending on moisture sensor of a plant a watering system should be switch on / off to reach the moisture monitored during the learning phase.

Remark: It should be possible to monitor several soil moisture sensors and each sensor should be controlled independent for the other sensors.

## 3 TTN Daten Auswertungen

Es wurde angeregt, dass man eine Auswertung über die LoraWAN Empfangsdaten (Metadaten) vornimmt. Wie z.B. die über welche Gateways hat die LoraWAN Funkverbindung stattgefunden.

### 3.1 TTN LoreNode und Gateway Tracking

Bei jeder Datenübertragung in das TTN Netzwerk werden die Gateways mit übermittelt, über die das LoraWAN Signal vom Ballon übertragen wurde.

Diese Daten werden auch von TTN über die MQTT Schnittstelle einer Applikation zur Verfügung gestellt. (getestet mit Node\_Red und Node.js Programm)

Ich habe ein Node.js Programm geschrieben, das die Daten von TTN über MQTT empfängt, in eine Datenbank schreibt und dann eine Geo-Karte aufbaut

```

▼ rx_metadata: array[2]
  ▼ 0: object
    ▼ gateway_ids: object
      gateway_id: "hs-raspi-
      gateway-1"
      eui:
      "B827EBFFFECBA8EE"
      timestamp: 1028220
      rssi: -63
      channel_rssi: -63
      snr: 10.5
    ▼ location: object
      latitude:
      48.67932591278685
      longitude:
      8.903032098084648
      altitude: 430
      source:
      "SOURCE_REGISTRY"
    uplink_token:
    "CiAKHgoSaHMtcmFzcGktZ2F
    0ZXdheS0xEgi4J+v//su07hD
    84D4aDAii30KcBhCAoMGXAYD
    gyKXqg7sW"
    channel_index: 6
    received_at: "2022-12-
    13T16:53:54.827330754Z"
  ▼ 1: object
    ▼ gateway_ids: object
      gateway_id: "hs-
      dragion-lsp8"
      eui:
      "A8404120BF144150"
      time: "2022-12-
      13T16:53:54.837400Z"

```

Figure 12: TTN Message Metadata Gateway Info

In den meisten Fällen haben die Gateways auch die Lokationsdaten mit dabei und darüber kann dann eine Internetseite aufgebaut werden mit einer Maps und den Standorten, wo das Signal empfangen wurde.

Die Daten können in einer Datenbank abgelegt werden. Folgende Daten werden im mqtt message Objekt übermittelt:

```

gateway_id: "hs-raspi-gateway-1"
timestamp: 1028220
rssi: -63
channel_rssi: -63
snr: 10.5
location: object
latitude: 48.67932591278685
longitude: 8.903032098084648
altitude: 430
received_at: "2022-12-13T16:53:54.827330754Z"

```

Im settings object der empfangen Nachricht ist noch der interessante Wert der Datenübertragung und des Spreadingfaktors enthalten sowie die Frequenz

```

settings: object
data_rate: object
lora: object
bandwidth: 125000
spreading_factor: 7
coding_rate: "4/5"
frequency: "867700000"

```

### 3.2 TTN LoreNode und Gateway Tracking Application

### **3.2.1 System Context**

# System context diagram

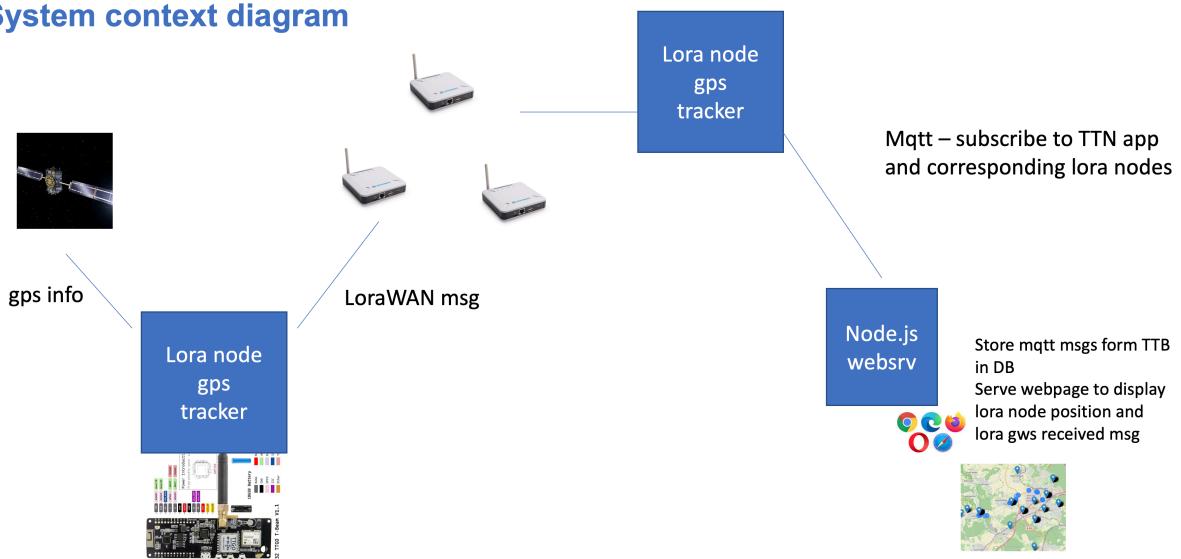


Figure 13: System Context Diagram

### 3.2.2 Introduction WebSrv – node.js

## The webserver app based on node.js

#### Main function blocks:

- Subscribe to TTN MQTT broker and get all the messages from an TTN APP
  - All LoraTracking nodes should be defined in this app (makes the MQTT topic subscription easier)
  - Save all received MQTT messages in a database (for the prototype dev. I used nedb – it is very handy and saves all date in a flat file which can be edited)
  - Get /gpsdata request form browser -> fetch data
  - Read data from database (query string has to be finally defined)
  - Send it as an array to browser

### 3.2.3 Deployment consideration

The webserver should be deployed in as a docker image which e.g. runs on a raspi server (Intranet) (I host such server at home which run 24h so I can capture all data)

If this app will be used during the stratosphere flight it should be deployed to cloud env. to access it via Internet

Raspi is an ARM processor and building a docker image for ARM using buildx did not work (on my MacOS – I don't know why)

A separate raspi was used to build the docker image

Main development env. is MacOS

To build the docker image the development Artefacts has to be copied to the raspi  
 Due to the fact a database is used the docker image / container should use a volume so that the data  
 is also available after restart of the docker image  
 IP addressing (name of the server and the port setting/mapping must be considered – where is to  
 node.js program running, where is the browser running ....)  
 Clarification necessary where to select the relevant data (on client side or on server side)  
 Design of database query necessary in conjunction with the webpage design

## Deployment model

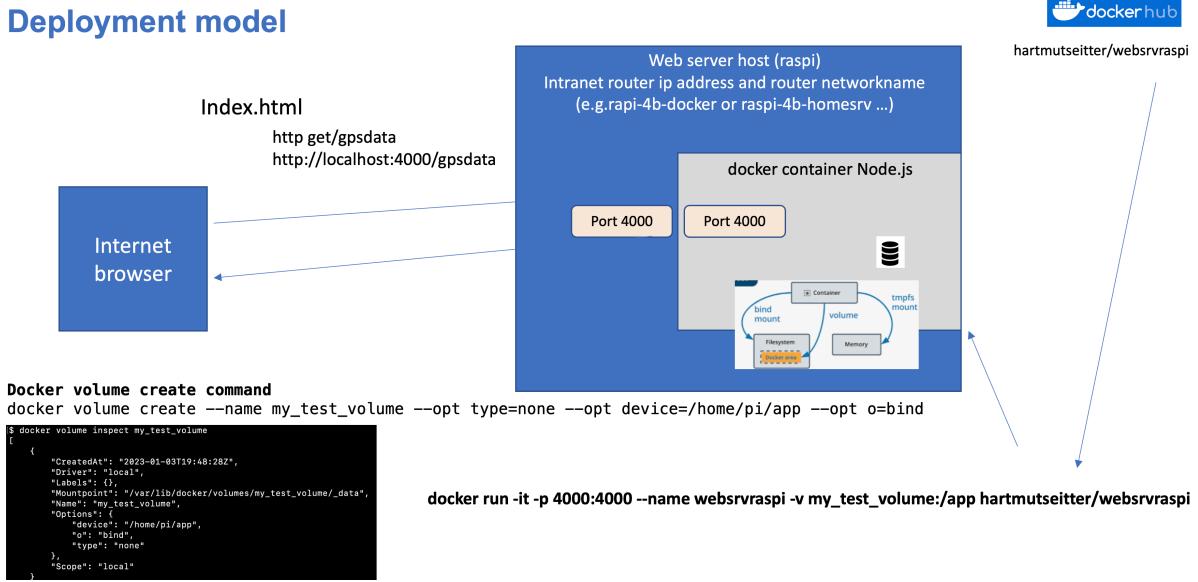


Figure 14: Deployment model Webserver

### 3.2.4 GPS Tracking Result

Für Testzwecke habe ich den LoraGPS Tracker mit auf die Fahrt von Aidlingen nach Sindelfingen mitgenommen und hier das Ergebnis.

Die Kreise sind die GPS Daten der LoraWAN end node und die Marker sind die LoraWan Gateways die die Lora Message empfangen haben.

Die LoraWan Funkabdeckung ist ziemlich vollständig, soweit ich das aus diesem Bild beurteilen kann, d.h. alle oder nahezu alle LoraMsg wurden von einem Gateway empfangen. Die GPS Daten den Liliygo T-Beam sind allerdings nicht so zuverlässig.

Das Sendeintervall lag bei 20 Sekunden was bei einer Geschwindigkeit von 70km/h ungefähr 400m entspricht.

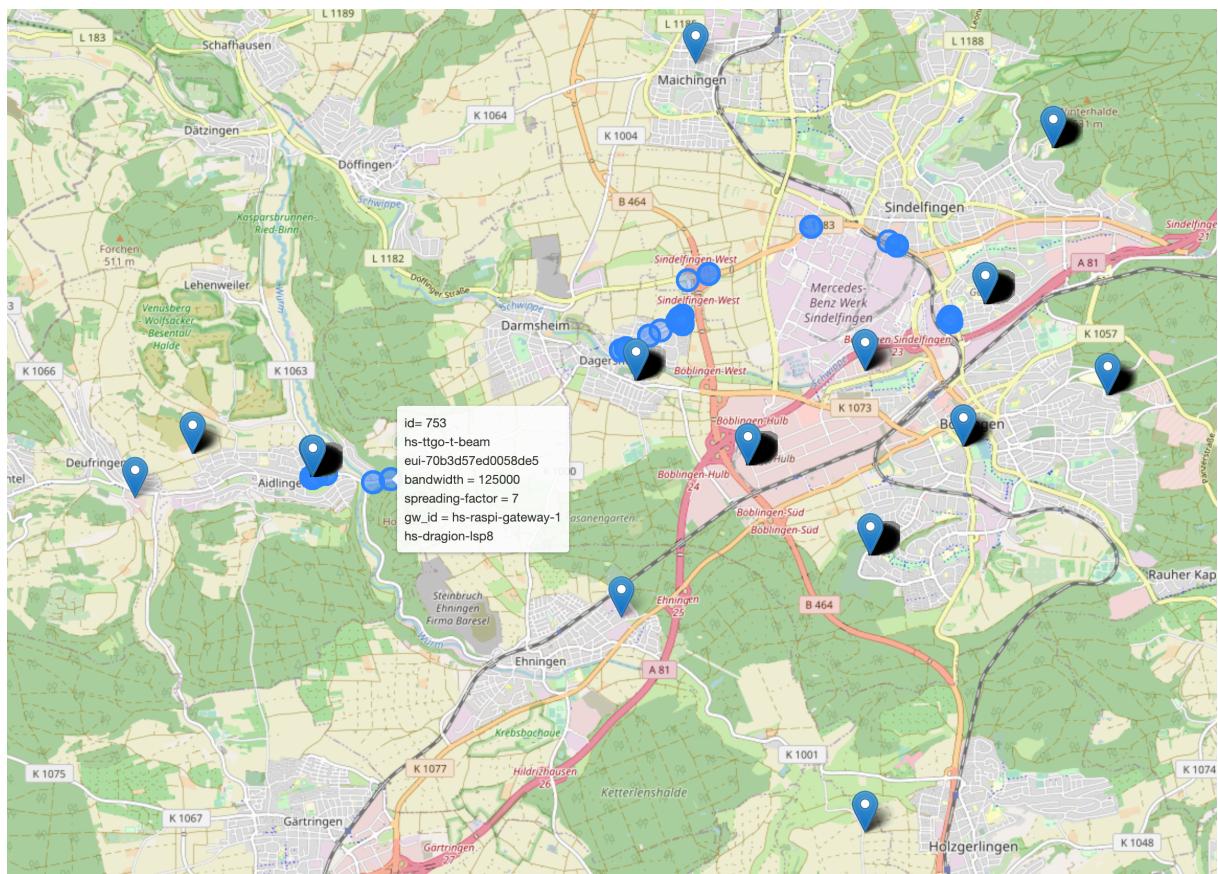


Figure 15: Geolocation map Gateways und LoraWAN end node

### 3.2.5 Entfernungs berechnung node – gateway

Aus den GPS Koordinaten lässt sich relativ einfach die Entfernung berechnen. Das habe ich mal für einige Daten gemacht und als Tabelle dargestellt.

index	app_id	dev_id	node_lat	node_lon	gw_id	gw_lat	gw_lon	distance	gw_rssi
0	hs-ttgo-t-beam	eui-70b3d57ed0058de5	0	0	hs-dragon-lsp8 hs-raspi-gateway-1	48.6792063554118 48.6791900511128	8.90313130163122 8.90292763710022	0	-75 -75
1	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.679191	8.902942	hs-dragon-lsp8 hs-raspi-gateway-1	48.6792063554118 48.6791900511128	8.90313130163122 8.90292763710022	0.0006568538963368586	-78 -61
2	hs-ttgo-t-beam	eui-70b3d57ed004ed36	0	0	hs-raspi-gateway-1 hs-dragon-lsp8	48.6791900511128 48.6792063554118	8.90292763710022 8.90313130163122	0	-75 -61
3	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.695972	9.007773	qv-ttn-gateway-1 packetbroker oli-home-01	48.67069568196863 48.68044553 48.69783	8.994565308094026 8.97439957 9.01346	0.28937191283495467	-114 -123 -126
4	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.679141	8.902929	hs-raspi-gateway-1 hs-dragon-lsp8	48.6791900511128 48.6792063554118	8.90292763710022 8.90313130163122	0.01027428033501132	-60 -73
5	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.695987	9.007825	qv-ttn-gateway-1 packetbroker	48.67069568196863 48.68044553	8.994565308094026 8.97439957	1.8647207734988152	-115 -123
6	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.696002	9.007836	qv-ttn-gateway-1	48.67069568196863	8.994565308094026	1.8502386325995104	-124
7	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.679191	8.902952	hs-dragon-lsp8 hs-raspi-gateway-1	48.6792063554118 48.6791900511128	8.90313130163122 8.90292763710022	0.001129686640348544	-72 -63
8	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.679191	8.902952	hs-raspi-gateway-1 hs-dragon-lsp8	48.6791900511128 48.6792063554118	8.90292763710022 8.90313130163122	0.008247881747216638	-77 -70
9	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.695972	9.007778	qv-ttn-gateway-1 oli-home-01	48.67069568196863 48.69783	8.994565308094026 9.01346	0.2891675840535546	-120 -119
10	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.695995	9.007857	qv-ttn-gateway-1 packetbroker	48.67069568196863 48.68044553	8.994565308094026 8.97439957	1.8662322538090184	-115 -123
11	hs-ttgo-t-beam	eui-70b3d57ed0058de5	48.679183	8.902945	hs-dragon-lsp8 hs-raspi-gateway-1	48.6792063554118 48.6791900511128	8.90313130163122 8.90292763710022	0.0009289316886971975	-72 -65

Figure 16: Geolocation Entfernungsberechnung Node zu Gateway

## 4 SSTV – Slow Scan Television

Um Bilddaten vom Stratosphärenflug online zu übermitteln, würde sich das SSTV vom Amateurfunk anbieten.

SSTV gibt es schon eine lange Zeit und damit werden überwiegend Bilddaten von Amateurfunkern übertragen. Aber auch die ISS sendet auf bestimmten Frequenzen Bilder vom Weltraum und der ISS via SSTV.

Damit man Bilddaten senden darf, ist eine Amateurfunklizenz notwendig, dies nur als Randbemerkung.

Folgendes Übersichtsbild habe ich entworfen um die verschiedenen Komponenten (bezogen auf den Ballonflug) darzustellen

### 4.1 Übersichtsbild SSTV für Ballonflug

## SSTV - my test – system context / overview

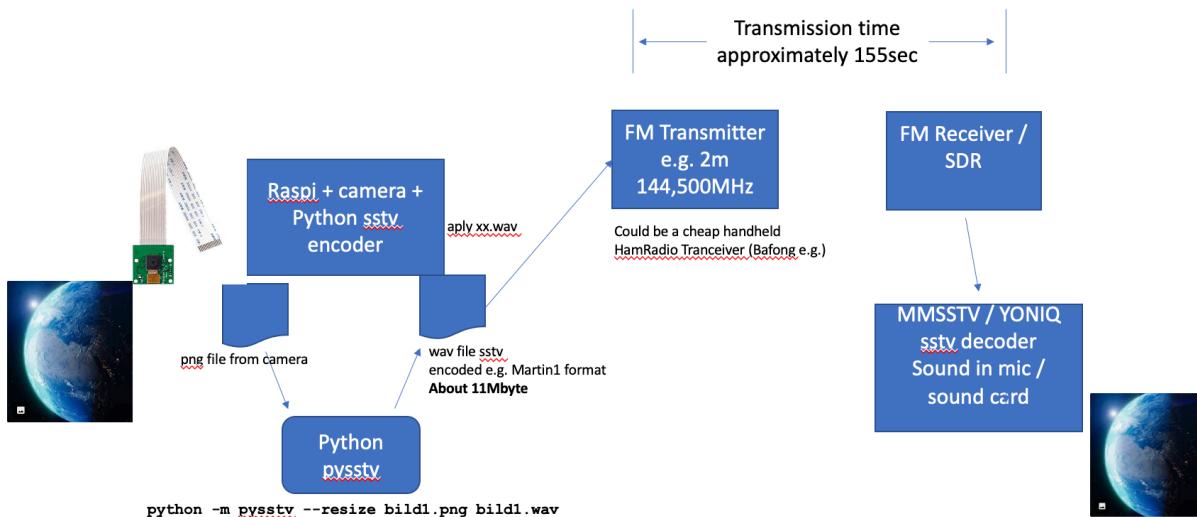


Figure 17: Übersichtsbild SSTV Komponenten

## 4.2 Kamera

Ich habe lediglich eine Versuchsaufbau realisiert in dem ich eine Raspi Kamera verwendet habe. Es würde auch eine ESP32 Cam gehen, allerdings gibt es in Bezug auf das ‚encoding‘ der Bilddatei eine Einschränkung

## 4.3 Raspi als Camera Controller und SSTV encoder

Das Bilder der Kamera (File) von der Übertragung in eine audio (wav) Datei in einem bestimmten Format umgewandelt werden, damit diese Daten dann per Transceiver übertragen werden können.

Für die Umwandlung habe ich ein fertiges Programm gefunden, das auf dem Raspi zu Ausführung kommt und die .png Datei in eine .wav Datei umwandelt.

Diese .wav Datei wird dann abgespielt und an den Ham Transceiver übermittelt werden. Der Ham Transceiver sollte einen Eingang besitzen, der direkt die Audiosignale entgegennehmen kann und dann auf dem 2m Band senden kann.

Die Größe der WAV Datei beträgt ca. 11Mbyte und das ‚Audio‘ für die Übertragung eines Bildes beträgt ungefähr 155sec.

Die WAV Datei kann mit dem Befehl

**aply xxx.wav**

abgespielt werden (Audio muss im raspi-config enabled sein)

An der Stelle gleich mal eine kurze Kalkulation der benötigten Batteriekapazität

### 4.3.1 Leistungsbedarf für diesen Setup RasPi 4

Raspi 4 braucht im Idle Betrieb ca. 540mA

Je nach Belastung des PI steigt die Stromaufnahme auf 1000mA und mehr

Wenn der RasPi ca. 2 Stunde Bilder aufnehmen / konvertieren und als Audiodatei dem Transceiver übergeben soll, dann kommt eine Gesamtleistung von

5V\*1A\*2Std = 10Ah zusammen.

Also dafür könnte z.B. eine 20000mAh Powerbank zu Einsatz kommen.

Die Powerbank wiegt ungefähr 400g

#### 4.3.2 Leistungsbedarf für diesen Setup Zero

Als Alternative würde der Pi Zero noch zur Verfügung stehen.

	<b>Zero</b>	<b>Zero W</b>	<b>A+</b>	<b>A</b>	<b>B+</b>	<b>B</b>	<b>Pi2B</b>	<b>Pi3B</b>
<b>Idling</b>	100	120	100	140	200	360	230	230
<b>Loading LXDE</b>	140	160	130	190	230	400	310	310
<b>Watch 1080p Video</b>	140	170	140	200	240	420	290	290
<b>Shoot 1080p Video</b>	240	230	230	320	330	480	350	350

*Pi Power Usage table adding Zero W (at 5.19V)*

Figure 18: Stromverbrauch Raspberries

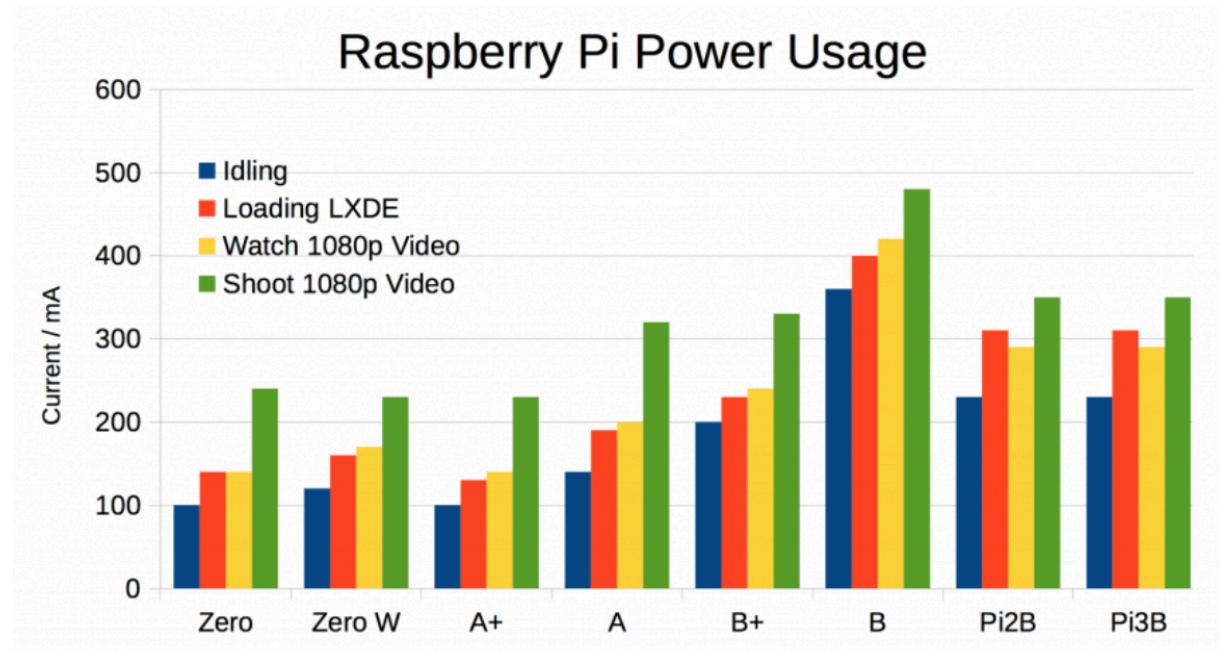


Figure 19: Stromverbrauch Raspberry bei Last

D.h. der Leistungsaufnahme liegt ungefähr bei 200mA

Wenn der RasPi ca. 2 Stunde Bilder aufnehmen / konvertieren und als Audiodatei dem Transceiver übergeben soll, dann kommt eine Gesamtleistung von

$$5V \cdot 0,2A \cdot 2\text{Std} = 2\text{Ah} \text{ zusammen.}$$

Also dafür könnte z.B. eine 2000mAh Powerbank zu Einsatz kommen.

**Die Powerbank wiegt ungefähr 60g**

#### 4.4 Baofeng 2m Tranceiver besitzt eine Akku z.b 1800mAh

Ich habe keine Angaben gefunden, wie lange man damit senden kann, aber die Sendeleistung beträgt ungefähr 5 W.

D.h. grob geschätzt braucht der Baofeng Tranceiver auch 1... 1,5A beim Senden.

Also der dürfte keine 2 Stunden senden – ich schätze mal, die eingebaute Batteriekapazität liegt (weit) unter einer Stunde Sendebetrieb.

Das Gerät wiegt ca. 250g

#### 4.5 Mein Test mit Museums HW

Um die Übertragung zu testen habe ich folgenden Testaufbau gemacht.

- Bild wurde mit RasPi und pysstv in eine WAV Datei umgewandelt
- Die Datei wurde als Audio Datei auf dem Macbook abgespielt.
- Kenwood TR2300 hat auf 145.9 MHz die Daten in FM übertragen

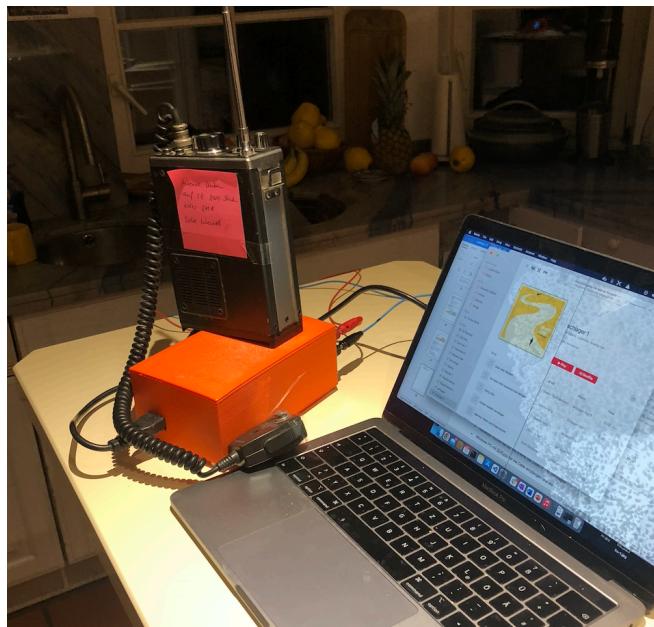


Figure 20: Test SSTV Übertragung Audio Datei zum FM Tranceiver

- Windows PC mit SDRUno empfangen (Bastelantenne – ich weiß die ist überhaupt nicht auf 2m abgestimmt (ist eigentlich ein 70cm Antenne9

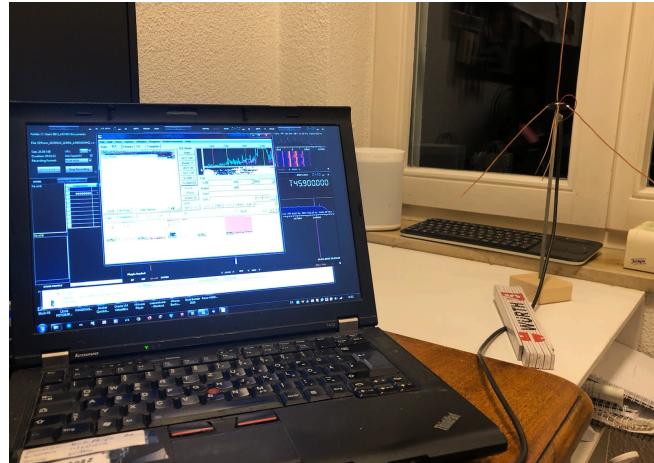


Figure 21: Empfangsstation SDR R1 mit SDRUno software (und auch MMSSTV)

Und die empfangenen Audio Signale werden mit dem MMSSTV über das im PC eingebaute Mikrofon wieder umgewandelt

Und das Bild ist auch erkennbar.

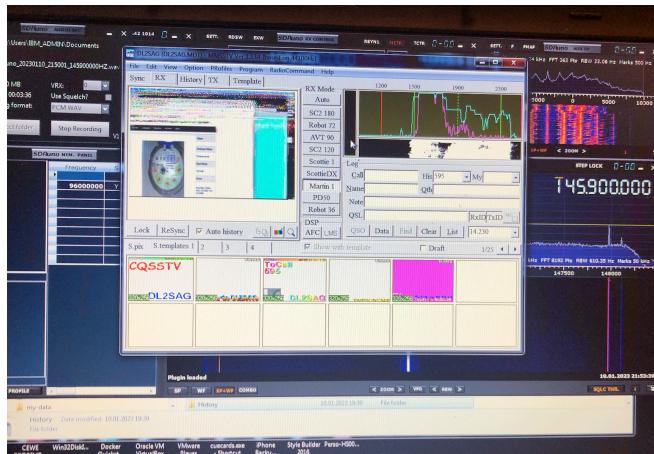


Figure 22: SSTV Empfangsstation Decodierung Audio zu Bild mit MMSSTV