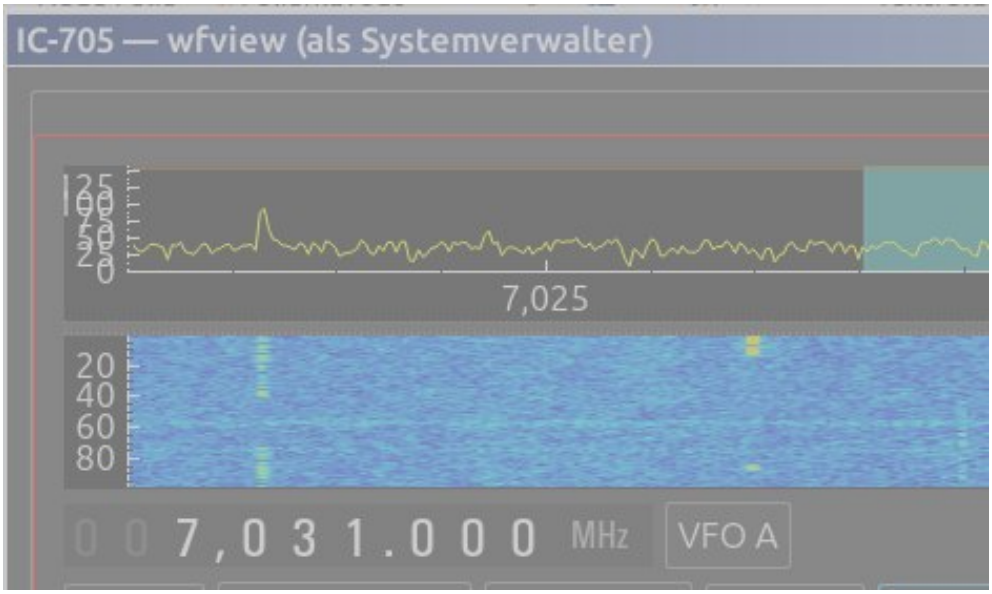
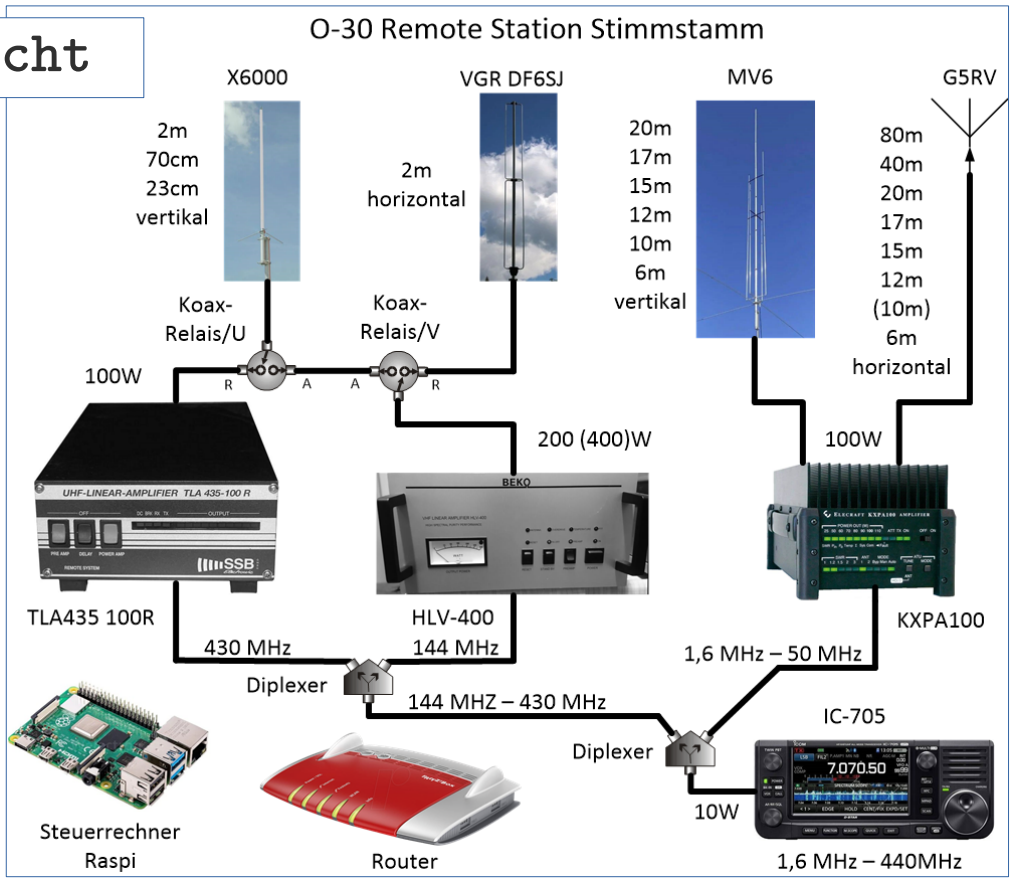
Endstufen- und Antennen- Fernsteuerung für einen ICOM IC-705



Übersicht

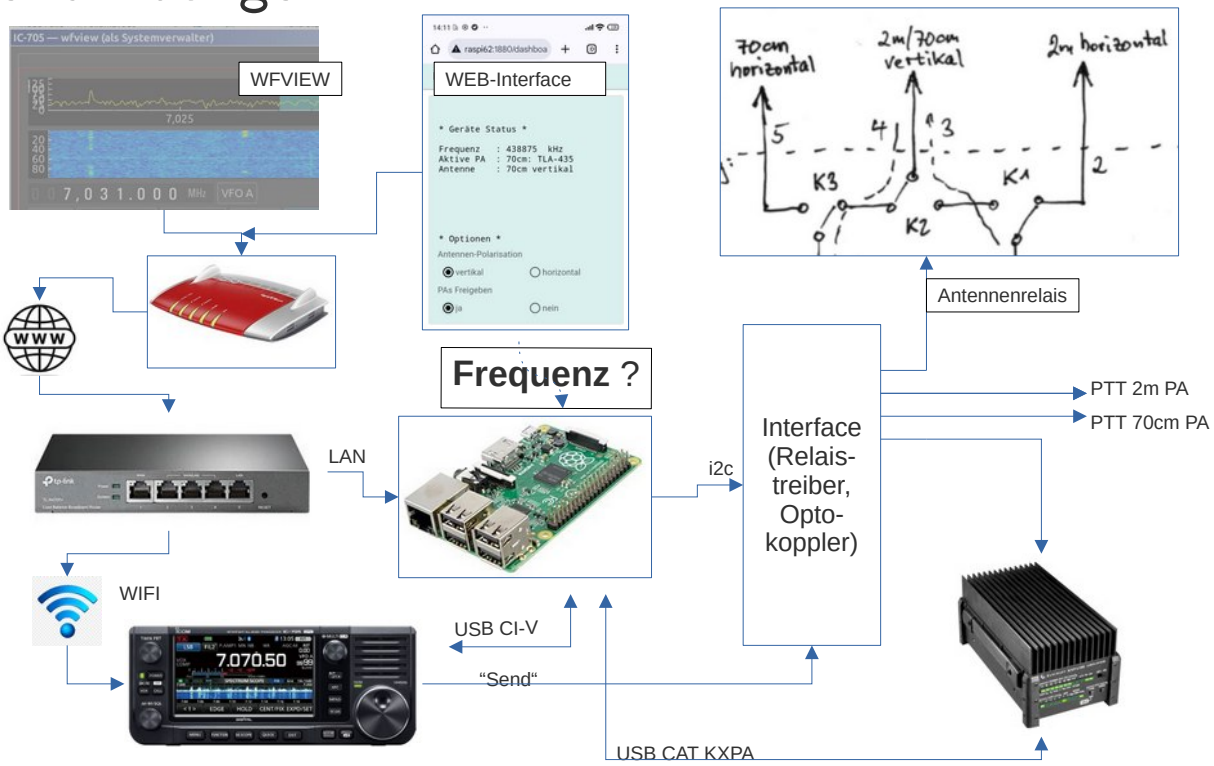


Die Übersicht zeigt die Gerätezusammenstellung.

Es ergeben sich folgende Aufgaben:

- Aufteilung der HF auf 3 Zweige
- Ansteuerung der passenden PA („Send“ - Signal)
- Ansteuerung der Antennenrelais
- Tuner Voreinstellung

Verbindungen



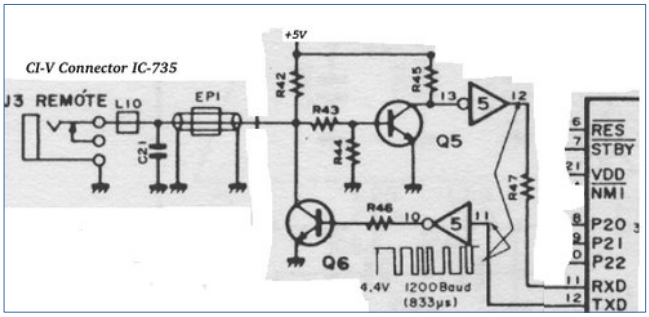
Wie kommst man an die Frequenz ?

---> Dazu gibt es bei ICOM die CI-V Schnittstelle alias CAT.

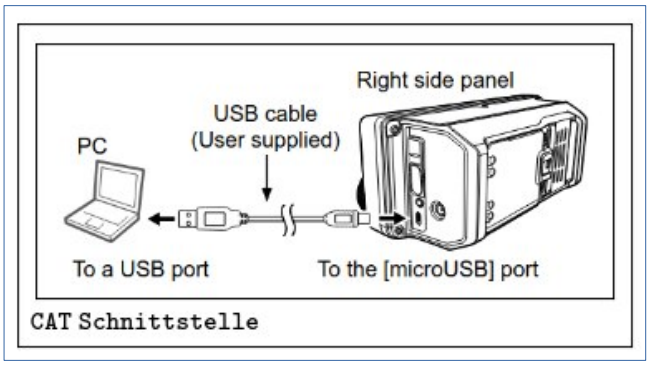
Beim IC-705 ist die CAT-Schnittstelle über USB erreichbar.

Hardware CI-V

CAT Steuerung = „Computer Aided Transceiver""



CI-V „ Computer Interface V (Fünf)
TTL -Level (Ruhezustand ca +5V, Aktivzustand ca. 0V)
Die Sende- und Empfangsleitung (RXD und TXD) wird auf eine Leitung kombiniert.



CAT Schnittstelle

Beim IC-705 ist stattdessen eine virtuelle serielle Schnittstelle als **USB-Schnittstelle** vorhanden

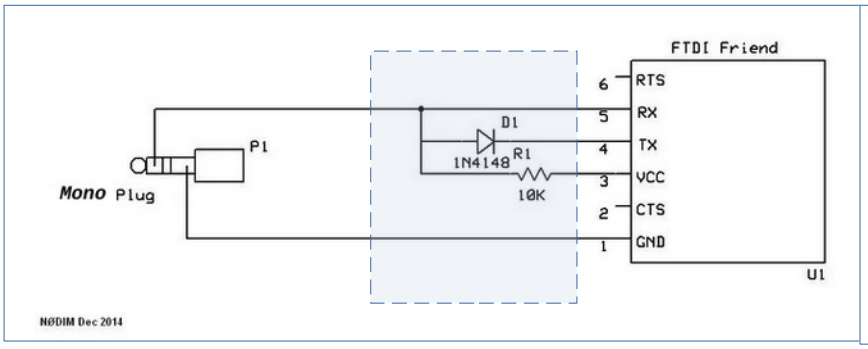
Bei älteren Icom- Geräten (z.B. IC-735) ist die CAT-Verbindung über nur eine Leitung für Senden und Empfangen von Daten realisiert (bidirektional); Bild oben.

Daher ‚hört' das angeschlossene Gerät oder die Steuerung seine gesendeten Daten auf der Empfangsseite mit (ist ja der gleiche Draht);

Beim IC-705 ist das nicht so, da RXD - und TxD - Kanal getrennt sind.

Beim IC-705 kann man jedoch dieses Echo ebenfalls einstellen durch „CI-V USB Echo Back = ON", wenn die Anwendung es erfordert.

CI-V – Adapter selbstgebaut



Adapter	---	*CI-V Adapter*
GND ,schwarz	---	G
TXD ,grün	---	rx
RXD ,weiß	---	tx
+5v	---	:

Solch einen CI-V-Adabter benötigt man , um einen normalen Ein-Draht-CI-V-Anschluss mit einem PC zu verbinden.

Die Funktion ist folgende:
Die CI-V Leitung ist im Ruhezustand HIGH (ca. 5 V) .
Die Seite , die die Leitung auf LOW zieht ‚gewinnt'.
Frage und Antwort wechseln sich ab.

CI-V Datenformat

alle Bytes in Hexadezimal-Schreibweise

```
''' Frequenzabfrage '''
kommando = "FE FE A4 E0 03 FD"
                    CI-V Adr.   Kommando: Frequenz?
#  => CI-V =>

antwort  = 'FE FE E0 A4 03 00 30 65 03 00 FD'
nutzdate =                "00 30 65 03 00"

''' Bytes von hinten nach vorne gelesen; '''
revers   =                "00 03 65 30 00"

frequenz = 3.65300  #MHz
```


Hier das CI-V - Kommando zum Abfragen der Frequenz .

Es besteht aus einem Header ‚FE FE', der Geräteadresse ‚A4', der Zieladresse ‚E0' und einem Abschlussbyte ‚FD'.

Die Daten hier sind Hexadezimal dargestellt ; Beispiel: FD (Hex) ist gleich ‚11111101' (binär).

Über die Datentypen braucht man sich jedoch wenig Gedanken machen, da die gängigen Programmiersprachen diese unterstützen. Man muss wissen , in welchen ‚Zahlenwelt' man sich befindet.

Test Frequenzabfrage

alle Bytes in Hexadezimal-Schreibweise

```
Python 3.11.2 (/usr/bin/python3) (/usr/bin/python3 on 80.149.48.126)
>>> from icom import Icom
>>> icom= Icom(port='/dev/ttyACM0', adr= 0xa4)
>>> icom.frequenz()
7015
>>> icom= Icom(port='/dev/ttyACM0', adr= 0xa4, debug=1)
>>> icom.frequenz()
 gesendet: FEFEA4E003FD (Frequenzabfrage ic-705)
 von ICOM: FEFEE0A4030050010700FD
    Daten: 0007015000
    [kHz]: 7015
```


Hier ein Beispiel für einen interaktiven Python-Script-Aufruf.
Interaktiv heisst: Python Aufrufe am Terminal.

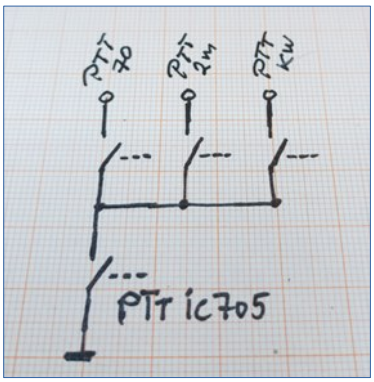
Basis ist ein Script ‚icom.py‘.
Einzelheiten werden hier nicht erläutert [6].

1. Class ‚Icom‘ wir importiert als ‚icom‘ (kleingeschrieben).
2. Die ser. Schnittstelle wird geöffnet
3. Die Funktion icom.frequenz() liefert die Frequenz.

Anschliessend das gleiche nochmals mit erweiterten Ausgaben zum Test.

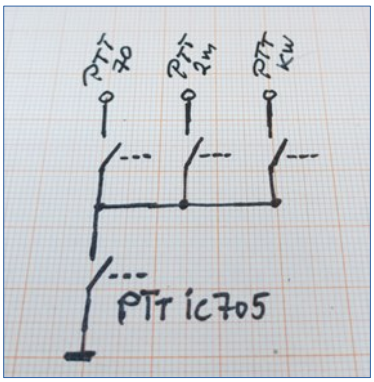
RaspberryPi schaltet PTT-Relais

PTT Freigabe der Endstufen erfolgt automatisch durch Frequenzabfrage über CI-V



Die PTT des Ic-705 (‚SEND'- Signal) schaltet beim Senden.
Die Raspi-Steuerung schaltet abhängig von der Frequenz vorher das PTT- Signal der passenden PA.

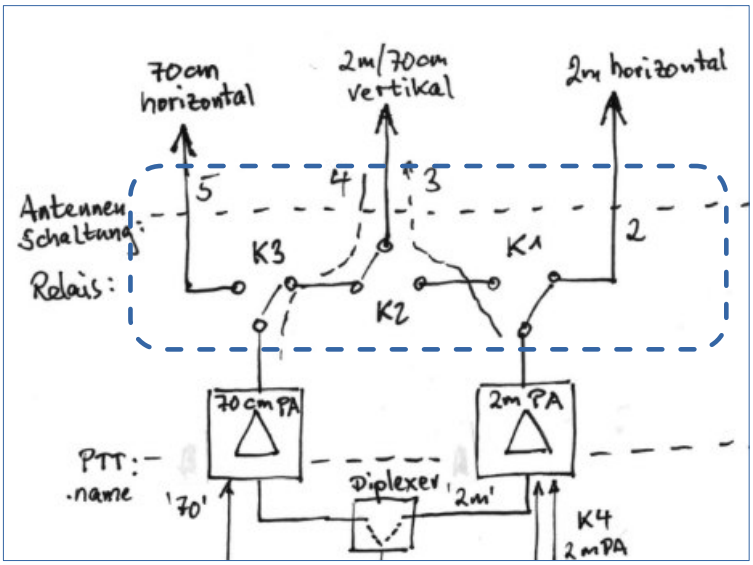
Dashboard blockiert PTT-Relais



Bei Bedarf kann man durch ein WEB-Interface (Dashboards) die **PA sperren .**

Bei Bedarf kann man zusätzlich durch ein WEB-Interface (Dashboard) die **PA sperren .**

RaspberryPi schaltet Antennenrelais



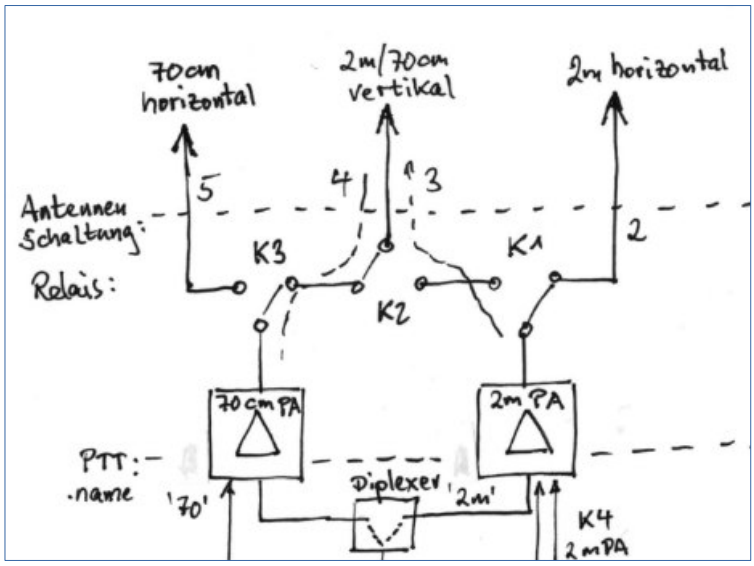
Die Steuerung wählt mit der Frequenz eine passende Antenne.

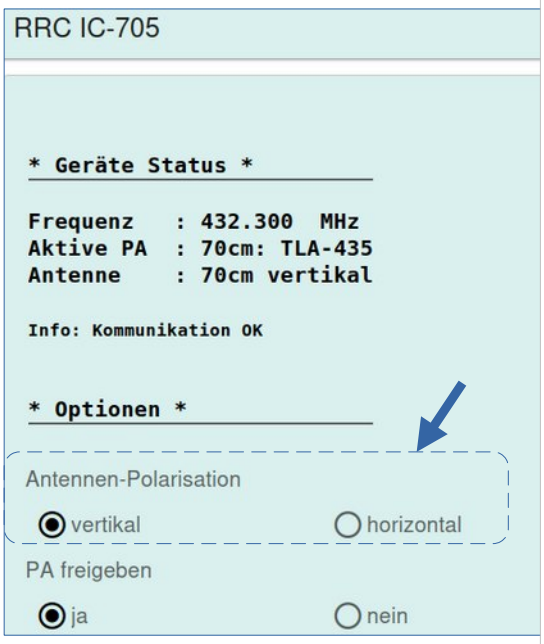
Standard: Vertikalantenne

Die Steuerung wählt mit der Frequenz eine passende Antenne.

Standard bei VHF/ UHF ist Vertikalantenne

Dashboard wählt Antenne / Polarisation





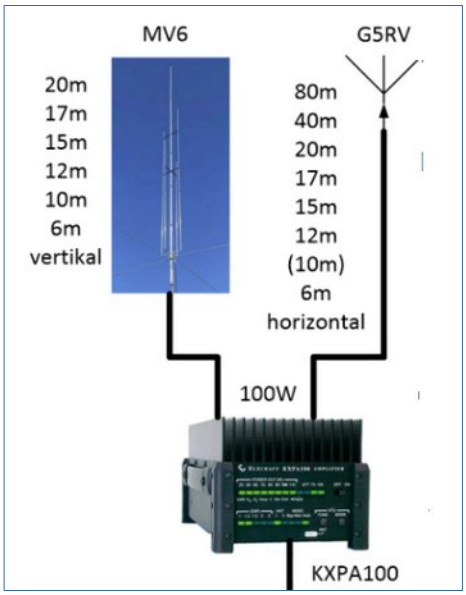
Zusätzlich kann man über das Dashboard Antennen umschalten.
Das trifft hier nur für die UHF- und VHF- Antennen zu; z.B. zum Umschalten von horizontaler auf vertikale Polarisation.

Voreinstellung der KXPA100

Der Tuner wird auf die Frequenz geschaltet noch bevor er überhaupt HF sieht.
über CAT-Schnittstelle
(einfaches ASCII-Format)

Befehl: ^MT7032
== Memmory Tune 7032 kHz

Die Abtennenumschaltung
Ant1 (< 10Mhz) und
Ant2 >=10 MHz
erfolgt von der KXPA-100 durch Setup automatisch.



Der Tuner wird auf die Frequenz geschaltet noch bevor er überhaupt HF sieht.

Der Tuner speichert alle 100 kHz Einstellungen, die schon einmal angewählt wurden.

Die 2 Antennenausgänge sind fest zugeordnet durch Parametrierung der KXPA.

Demo Python Script

Shell

```
Python 3.11.2 (/usr/bin/python3) (/usr/bin/python3 on 192.168.178.62)
>>> import check_serial_ports
  Serial Ports verfügbar: ['/dev/ttyUSB0', '/dev/ttyUSB1']
>>> from icom import Icom
>>> from kxpa import Kxpa
>>> icom=Icom(port='/dev/ttyUSB0', adr=0x58, debug=True)
>>> kxpa=Kxpa(port='/dev/ttyUSB1')
>>> f= icom.frequenz()
  gesendet: fefe58e003fd
  *** Test mit Icom 706, adr = 0x58 ***
  von ICOM: fefee058034121030700fd
>>> f
7032
>>> kxpa.set_frequenz(f)
b'^MT07032;'
```


Hier nochmals Programmteile im Python-Interaktiv-Mode:

- Frequenz lesen

- Frequenz an KXPA senden;
  Befehle sind in ASCII: hier ‚^‘ zur Einleitung ,
  dann MT für ‚Memory recall Tune‘,
  dann F in kHz und
  ‚;‘ als Abschluss.

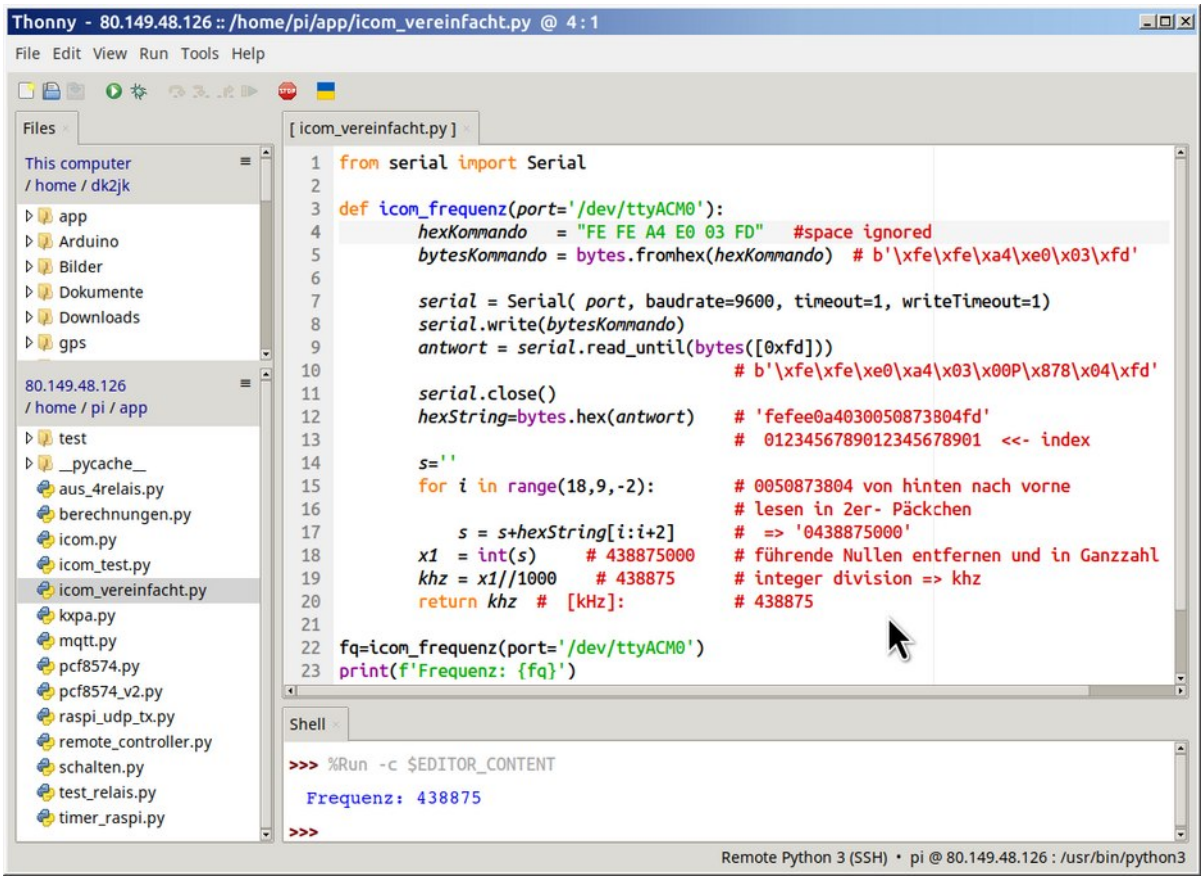
Dashboard

- von jeden Netzwerk- Rechner oder Smartphone aufrufbar:

  <IP-ADR>:<port>/dashboard

- Das Dashboard wurde als „Node Red Flow“ programmiert.

Das Dashboard ergänzt Funktionen, die von der Remote-Software WFVIEW nicht zur Verfügung stehen.
Zusätzlich wird angezeigt: Frequenz, gerade eingeschaltete Antenne und aktive PA.



Hier soll gezeigt werden, wie die Programmieroberfäche Thonny aussieht.

-Links unten : eine Liste mit Dateien auf dem RaspBerryPi
-Links obenen : eine Liste mit Dateien auf dem PC
-Rechts oben : das Editor-Fenster zum Programm-Schreiben.
-Rechts unten : die Ausgaben des Programms (‚print').

Mit dieser Oberfäche wird über die Interpreter-Option Verbindung „remote Python3 SSH" das Raspi-Programm

80.149.48.126.home/pi/app/icom_vereinfacht.py

bearbeitet.

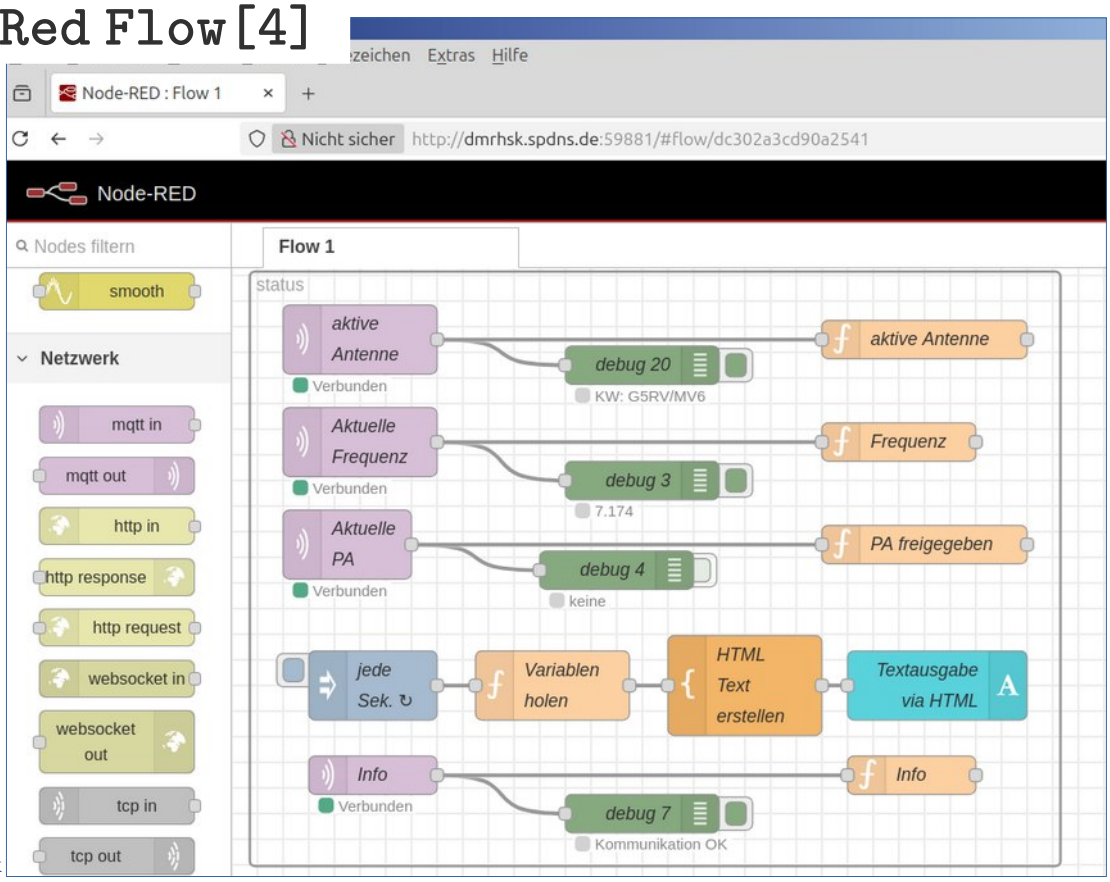
Der Interpreter ist Python 3.11.2

der sich auf dem Raspi bei

80.149.48.126/usr/bin/python3) befindet.

Die Oberfläche ‚Thonny' selbst läuft auf meinem PC .

Node-Red Flow [4]



Okt 2025 dk2jk

Die WEB-Oberfläche „Dashboard' ist in ‚Node-Red' programmiert.

Es wird mit vordefinierten Blöcken gearbeitet, die parametriert werden. Die Linien zwischen den Blöcken sind Signale.

Datenbank Mosquitto [3]

Steuerung und WEB-Interface sind über die Datenbank gekoppelt

publish sendet ein Topic mit Name ‚frequ' und Wert f

subscribe empfängt einen Wert mit dem Namen ‚pa-freigabe'

Anmerkung: die Daten sind zum Testen auch über Smartphone änderbar

Die Datenbank Mosquitto mit MQTT-Protokoll dient zum Koppeln von Steuerung und Dashboard;
Man könnte die Daten auch direkt koppeln, z.b. über FTP. Die Datenbank ist jedoch zum Testen recht praktisch, da sie auch von anderen Quellen gespeist werden kann (z.B. Handy).

Das Prinzip: einer sendet (‚publish') und ein anderer liest (‚subscribe').

Daten bleiben bis zur nächsten Änderung erhalten. Das MQTT-Protokoll wird auch gern bei IOT-Projekten verwendet.

Referenzen

[1] https://www.python.org/

[2] https://thonny.org/

[3] https://mosquitto.org/

[4] https://nodered.org/

[5] https://www.raspberrypi.com/software/

Python-Scripts: [6] https://dk2jk.darc.de/swt/2025/app/

Diese Präsentation:
https://dk2jk.darc.de/swt/2025/praesentation/remote-ic-705-notizen.pdf

Autor:
Heribert Schulte , DK2JK
dk2jk@darc.de