Dipl.-Ing. Michael Zimmermann
Buchenstr. 15
42699 Solingen
☎ 0212 46267
https://kruemelsoft.hier-im-netz.de
BwMichelstadt@t-online.de

Michelstadt (Bw)

Der in diesem Dokument beschriebene Workflow hat dem Autor geholfen, seinen RaspBerry Pi als Access-Point für das Handregler-WLAN einzurichten. Das Dokument ist als Rezept zu betrachten, aufgrund neuerer kommender Software-Versionen (Updates) können sich Änderungen oder Ergänzungen ergeben.

Der Autor dieser Anleitung übernimmt keine Garantie oder Haftung für Schäden oder Mängel, die sich durch die Verwendung dieser Anleitung ergeben.

Für Hinweise auf Fehler oder Ergänzungen ist der Autor dankbar.

Die Nennung von Marken- und Firmennamen geschieht in rein privater und nichtgewerblicher Nutzung und ohne Rücksicht auf bestehende Schutzrechte.

Diese Zusammenstellung wurde nach bestem Wissen
und ohne Funktionsgarantie in der Hoffnung erstellt, dass sie nützlich ist.
Wenn sie nicht nützlich ist – dann eben nicht.

---

21. September 2024

Übersicht

RaspBerryPi – System erstellen 3
Vorbereitende Maßnahmen 3
SD-Card erstellen 3
64Bit-Kernel aktivieren (optional) 4
WiFi-Name und Passwort ändern 4
Hostname ändern 4
DHCP/DNS-Adressbereich ändern 5
Passwort ändern 5
WiFi überprüfen 5
WiFi-Teilnehmer auflisten *(‚Poll WiFi-Clients')* 5
JMRI 6
wiThrottle-Server - Einstellungen 6
wiThrottle-Server starten 6
LocoNET® - Einstellungen 7
Angeschlossene Modellbahnkomponenten 8
USB-Namen vergeben 8
USB-Schnittstellen prüfen 8
weitere Software installieren 10
Tool für die Adressvergabe von Fahrzeugen für den wiThrottle(A.Heckt) 10
Software updaten 11
Raspberry-Pi 11
Überprüfen der OS-Version 11
Update/Upgrade 11
Java 11
JMRI 11
Anhang A: JMRI – wiThrottle-Server 12
Status: Unbekannt 12
Status: Ausgeschaltet 12
Status: Eingeschaltet 12
Anzeige: WLAN-Handregler 13
Anmerkungen 13
Anhang B: Michaels WLAN-Handregler 13
Anhang C: Verbindung der Geräte 14
Anhang D: Raspberry von USB-Stick starten 15
Raspberry Pi für das Booten von einem USB-Laufwerk aktivieren 15
USB-Laufwerk vorbereiten 15
Raspberry Pi mit USB-Laufwerk in Betrieb nehmen 15
Troubleshooting 16
Ergänzungen 16

RaspBerryPi – System erstellen

Verwendete Hardware:

```
Pi3 Model B Rev 1.2, 1GB, 4*USB2.0
RASPBERRY PI 7TD (7" [17,8cm], 800x480 Pixel)
SD-Card: 32GB (min: 16GB, max:64GB)
```

Vorbereitende Maßnahmen

- Auf dem PC:
  - Download der Softwarepakete:
    - Aktuelles Image-File
      https://mstevetodd.com/jmri-raspberrypi-access-point
      [12.9.2023, ca. 1,5GB]
      beinhaltet:
      - OS „Bullseye“
      - JMRI 5.5.4
      - Java 11.0.18
    - „balenaEtcher“ (https://etcher.io/) zur Erstellung der SD-Card

SD-Card erstellen

- Auf dem PC:
  - balenaEtcher-Portable-1.8.11.exe aufrufen
    und das Image RPi-JMRI.20230912.xz auf SD-Card übertragen

- Am RaspPi:
  - Micro-SD-Card einsetzen
  - TFT, Tastatur und Maus sind am PI angeschlossen
    GANZ WICHTIG: LocoNET und DCC-Zentrale müssen vor jedem Einschalten des RaspPi angeschlossen und eingeschaltet sein!
  - RaspPi starten
  - Für die Verwendung einer Remoteverbindung (VNC-Viewer, SSH o.ä.) ist eine zusätzliche Verbindung mit dem heimischen LAN erforderlich.
  - Einstellungen > Raspberry-Pi-Konfiguration:
    - Lokalisierung = alles auf Deutsch einstellen
    - Zeitzone = GMT, Berlin
    - Auflösung[1] = 720*480
  - Einstellungen > Screen Configuration[1]:
    - Bildschirm (hier: DSI-1) auswählen
      - Rechtsklick mit der Maus, Menüpunkt Drehung auswählen
      - Inverted anwählen
      - Mit Apply bestätigen

[1] Wenn erforderlich, z.B. beim Einsatz des TFT-Displays. Als Anzeige kann anstelle des TFT-Displays auch ein Monitor über den HDMI-Port angeschlossen werden.

64Bit-Kernel aktivieren (optional)

Version prüfen:

```
uname -a
```

Erscheint im Ausgabetext die Angabe `aarch64`, so ist der 64Bit-Kernel bereits aktiviert, ansonsten:

```
sudo nano /boot/config.txt
```

Parameter am Ende hinzufügen:

```
arm_64bit=1
```

Änderungen speichern, RaspPi wird später neugestartet

WiFi-Name und Passwort ändern

```
sudo nano /etc/hostapd/hostapd.conf
```

Parameter anpassen (Password wird auch für den SSH-Zugriff benötigt):

```
ssid=RPi-JMRI_MZ[2]
wpa-passphrase=rpI-jmri_mz
country_code=DE
```

Änderungen speichern, RaspPi wird später neugestartet

Hostname ändern

```
sudo nano /etc/hostname
```

Parameter anpassen:

```
RPi-JMRI_MZ
```

Änderungen speichern,

```
sudo nano /etc/hosts
```

Parameter anpassen:

```
RPi-JMRI_MZ
```

Änderungen speichern, RaspPi wird später neugestartet

[2] ***_MZ*** bezeichnet meine Komponenten, hier sollte jeder sein eigenes Kürzel verwenden (oder ein völlig anderes WLAN-Netz aufspannen). Dies ist vor allem wichtig, wenn mehrere WLAN-Netze auf (Modul-)Treffen im Einsatz sind…

DHCP/DNS-Adressbereich ändern

Unschwer zu erkennen: mein Adressbereich verwendet den Bereich *192.168.**60***.

```
sudo nano /etc/dhcpcd.conf
```

Parameter anpassen:

```
interface wlan0
    static ip_address=192.168.60.1/24
    nonhook wpa_supplicant
```

Änderungen speichern,

```
sudo nano /etc/dnsmasq.conf
```

Parameter anpassen:

```
interface=wlan0
dhcp-range=192.168.60.50,192.168.60.99,255.255.255.0,12h
```

Änderungen speichern, RaspPi wird später neugestartet

Passwort ändern

```
passwd                                    (initial: rpI-jmri)
sudo vncpasswd -service
sudo smbpasswd -a pi
```

WiFi überprüfen

```
iwconfig
```

WiFi-Teilnehmer auflisten *(‚Poll WiFi-Clients')*

show_wifi_clients.sh erstellen (sofern noch nicht im Verzeichnis /home/pi/ vorhanden):

```
#!/bin/sh
echo    "# All connected wifi devices [ip hostname mac]:"
for interface in `iw dev | grep Interface | cut -f 2 -s -d" "`
do
  maclist=`iw dev $interface station dump | grep Station | cut -f 2 - s -d"
"`
  for mac in $maclist
  do
    ip="UNKN"
    host=""
    ip=`cat /var/lib/misc/dnsmasq.leases | cut -f 2,3,4 -s -d" " ↵
         | grep $mac | cut -f 2 -s -d" "`
    host=`cat /var/lib/misc/dnsmasq.leases | cut -f 2,3,4 -s -d" " ↵
         | grep $mac | cut -f 3 -s -d" "`
    echo "$ip\t$host\t$mac"
  done
donepi
```

und Teilnehmer prüfen:

```
bash show_wifi_clients.sh
```

oder

```
iwlist wlan0 scan
```

verwenden.

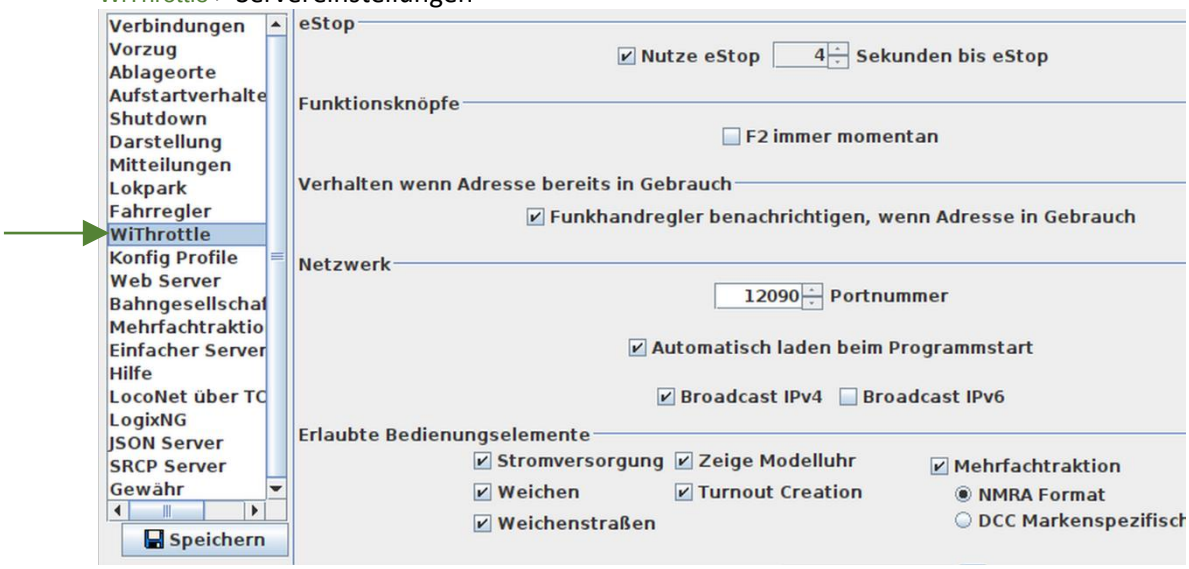
JMRI

- Icon auf den Desktop bringen (sofern noch nicht geschehen):
  - Einstellungen > Main Menu Editor
    - Unter „Sonstiges“ hinzufügen: „DecoderPro®“ aus /home/pi/JMRI/
    - Unter „Sonstiges“ hinzufügen: „PanelPro™“ aus /home/pi/JMRI/
  - Sonstiges > DekoderPro: rechte Maustaste: Der Arbeitsfläche hinzufügen
  - Sonstiges > PanelPro: rechte Maustaste: Der Arbeitsfläche hinzufügen

- DecoderPro – Voreinstellung für Verbindungen
  je nachdem, wo LocoBuffer und OpenDCC-Zentrale angeschlossen sind, ändert sich:
  *→ **ttyUSB0** in **ttyUSB1** bzw. **ttyUSB2** und umgekehrt (siehe oben)*

wiThrottle-Server - Einstellungen

- Bearbeiten > Voreinstellungen…
  WiThrottle > Servereinstellungen



wiThrottle-Server starten

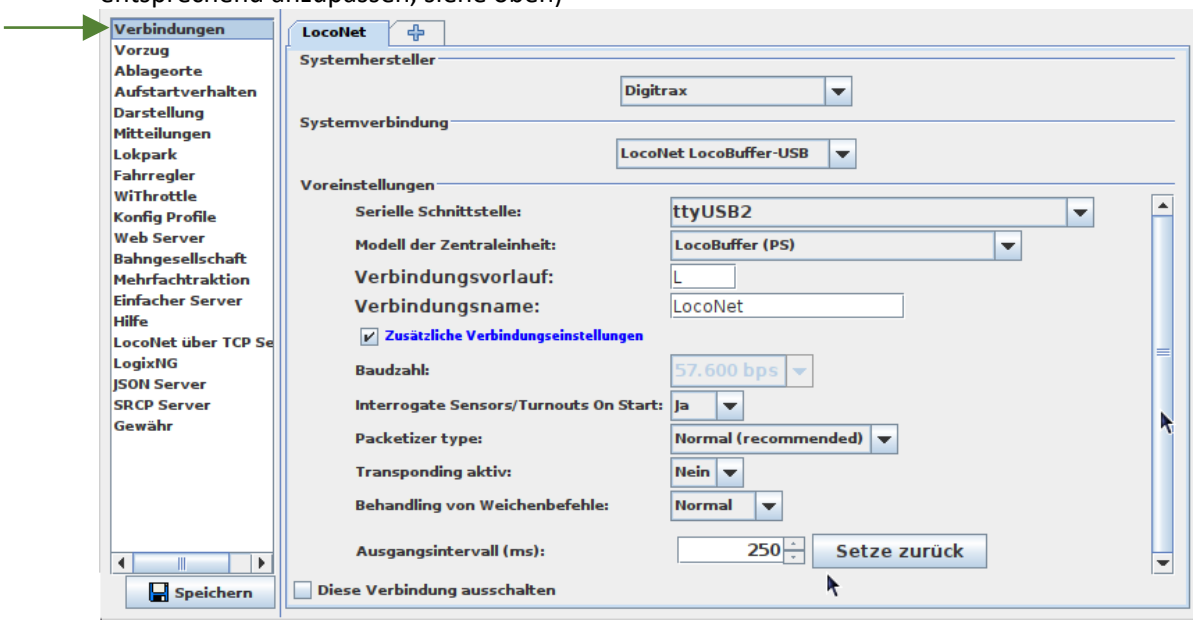
Durch die Verwendung und Installation des Softwarepakets von Steve Todd startet der wiThrottle-Server bei jedem Starten des RaspBerry automatisch, hier sind also keinerlei Bedienhandlungen erforderlich.

Ein manueller Start ist in PanelPro™ über die Menüpunkte Werkzeuge → Servers → Starte WiThrotttle Server möglich.

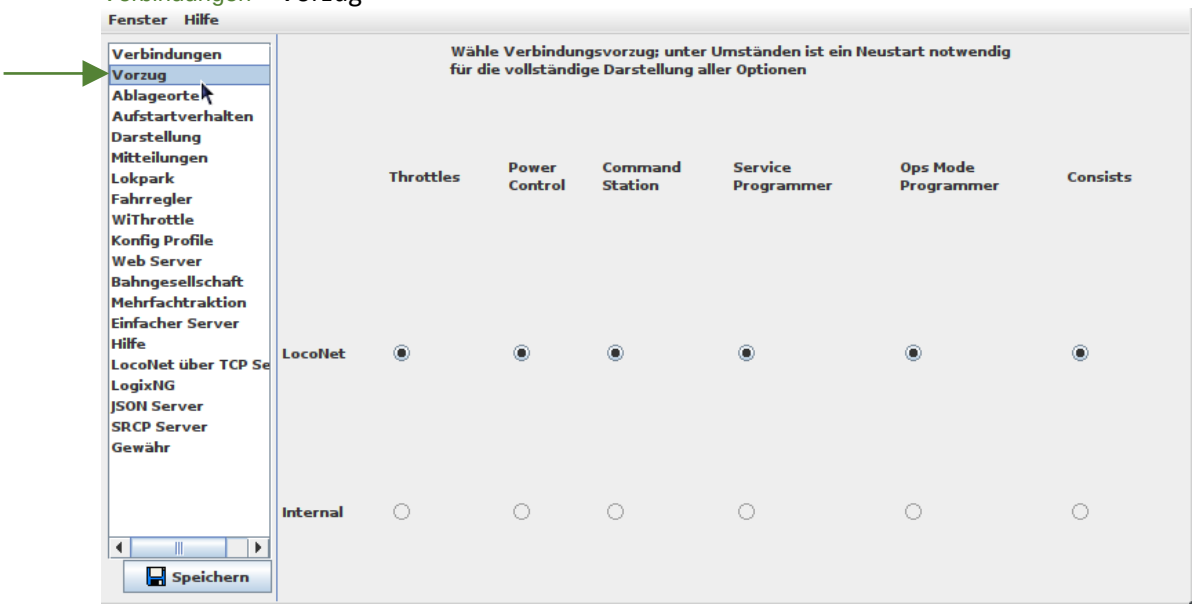
LocoNET® - Einstellungen

Als Beispiel wird hier ein LocoBuffer als Zentrale konfiguriert, da in meinem Aufbau der wiThrottle-Server seine Befehle über das LocoNET® sendet. Diese Befehle werden dann von der ans LocoNET® angeschlossenen Zentrale in DCC-Befehle übersetzt, diese werden dann ans Gleis gesendet.
Tatsächlich kann man hier natürlich auch direkt über den USB-Anschluss eine (DCC-)Zentrale anschließen und sich den Umweg über den LocoBuffer sparen. Das wurde jedoch mit dem TwinCenter nicht getestet.

Verbindungen > LocoNET® erstellen (die verwendete Schnittstelle – hier: ***ttyUSB2*** - ist entsprechend anzupassen, siehe oben)



Verbindungen > Vorzug



Änderungen speichern und RaspPi neu starten:

```
sudo reboot
```

Um Problemen mit der Speicherkarte frühzeitig aus dem Weg zu gehen, empfiehlt es sich, den Raspberry von einem USB-Stick zu booten (siehe Anhang D).

Angeschlossene Modellbahnkomponenten

An den RaspPi sind über USB angeschlossen:

- das Arduino-Nano-Interface (= Arduino-Nano mit *LocoLinx.ino*, für den wiThrottle-Server relevant, https://github.com/Kruemelbahn/LocoBuffer-Nano)
- die OpenDCC-Zentrale Z1 von W.Kufer (*optional*)
- der LocoBuffer von H.Deloof (*optional*)

Die nachfolgenden Beschreibungen zu den Einstellungen sind auf diese Geräte abgestimmt.

USB-Namen vergeben

```
sudo nano /etc/udev/rules.d/10-opendcc.rules
```

einfügen:

```
KERNEL=="ttyUSB*" ATTRS{idVendor}=="1a86", ATTRS{idProduct}=="7523",↵
  SYMLINK+="loconet/NANO-Interface"
```

optional auch:

```
KERNEL=="ttyUSB*" ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001",↵
  SYMLINK+="opendcc/LocoBuffer"
KERNEL=="ttyUSB*" ATTRS{idVendor}=="0403", ATTRS{idProduct}=="bfd8",↵
  SYMLINK+="opendcc/opendcc"
```

(als neue Datei) speichern.
Diese Datei kann mit

```
sudo cp <quelle> /etc/udev/rules.d/10-opendcc.rules
```

kopiert bzw. mit

```
sudo mv <quelle> /etc/udev/rules.d/10-opendcc.rules
```

verschoben werden.

Nach allen Änderungen wird jetzt der RaspPi **neugestartet**:

```
sudo reboot
```

USB-Schnittstellen prüfen

je nachdem, wo NANO-Interface, LocoBuffer und OpenDCC-Zentrale angeschlossen sind, ändert sich:
*→ **ttyUSB0** in **ttyUSB1** bzw. **ttyUSB2** und umgekehrt*

usb.sh erstellen (sofern noch nicht im Verzeichnis /home/pi/ vorhanden):

```
#!/bin/bash
for sysdevpath in $(find /sys/bus/usb/devices/usb*/ -name dev); do
(
 syspath="${sysdevpath%/dev}"
 devname="$(udevadm info -q name -p $syspath)"
 [[ "$devname" == "bus/"* ]] && exit
 eval "$(udevadm info -q property --export -p $syspath)"
 [[ -z "$ID_SERIAL" ]] && exit
 echo "/dev/$devname - $ID_SERIAL"
)
done
```

und Schnittstellen prüfen:

```
bash usb.sh
```

oder

```
lsusb
```

verwenden.

Ein Ergebnis kann sein:

Für den WLAN-Handregler von A.Heckt:

```
/dev/ttyUSB1 - Silicon_Labs_CP2102N_USB_to_UART_Bridge_Controller_↵
      828e08137410e9118891597387f8ef3e
```

Für den LocoBuffer:

```
/dev/ttyUSB0 - 1a86_USB2.0-Serial
```

Für den AVR-Programmer DIAMAX Prog-S2:

```
/dev/ttyACM0 - ERFOS_PROG-S2_19379-35100-255
```

weitere Software installieren

Tool für die Adressvergabe von Fahrzeugen für den wiThrottle(A.Heckt)

Die Adressvergabe von Triebfahrzeugen für den wiThrottle von A.Heckt erfolgt über eine USB-Schnittstelle mit einem Terminal-Programm. Am einfachsten geht das mit der Arduino-IDE. Dafür ist aber auf einer 16GB-Micro-SD-Karte kein Platz mehr.
Eine gute Alternative – es wird ja nur ein Terminalprogramm und keine komplette IDE benötigt - ist das frei verfügbare Programm `cutecom`.

```
sudo apt update
sudo apt-get install cutecom
```

Aufruf von `cutecom`

```
sudo cutecom
```

Im Konfigurationsfenster sind folgende Einstellungen vorzunehmen (sofern nicht bereits gesetzt):

```
Serieller Anschluss: /dev/ttyUSBx
Baudrate und Datenformat: 115200 8N1
```

Für `cutecom` gilt:

- das Programm hat eine grafische Oberfläche (kann also nicht z.B. über SSH gestartet werden)
- ***x*** ist durch die entsprechende USB-Nummer zu ersetzen ist (siehe auch weiter oben)
- das Programm sendet in der Eingabezeile stehende Daten erst, wenn die Zeile mit `Return` beendet wird – entspricht also dem Verhalten des seriellen Monitors der Arduino-IDE.

Software updaten

Raspberry-Pi

Überprüfen der OS-Version

```
cat /etc/os-release
```
(=> *BULLSEYE*)

Update/Upgrade

```
sudo apt update
sudo apt upgrade
```

Java

```
sudo apt update
sudo apt install openjdk-11-jdk
java -version
```
(=> *11.0.21*)

JMRI

→ zu Hinweisen, welche JAVA-Version für welche JMRI-Version erforderlich ist: ←
→ siehe https://www.jmri.org/←

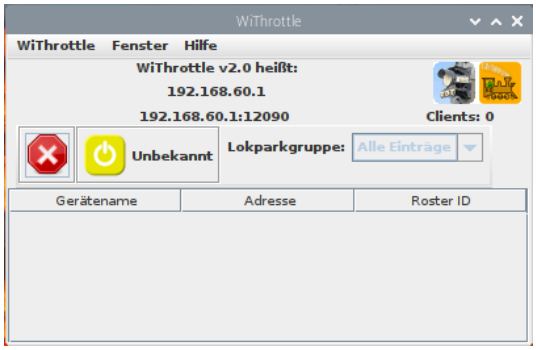
- Download der aktuellen JMRI-Linux-Version von https://www.jmri.org/ durch klicken auf die tgz-Datei
- Archiv entpacken nach: /home/pi/

Anhang A: JMRI – wiThrottle-Server

Der wiThrottle-Server hat je nach Betriebszustand verschiedene Darstellungen. Eine ausführliche Hilfe gibt es auf der Seite https://www.jmri.org.

Status: Unbekannt

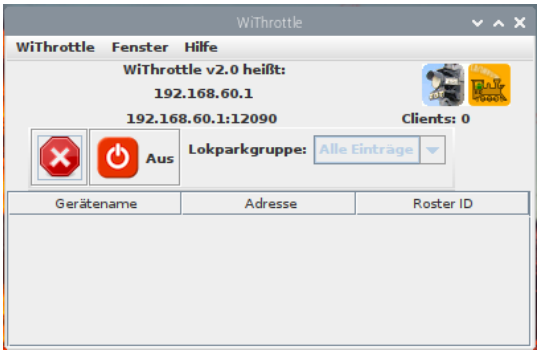
Ist der Zustand der angeschlossenen Zentrale „unbekannt“ (oder die Zentrale nicht angeschlossen / eingeschaltet), sieht der wiThrottle-Server so aus:



Status: Ausgeschaltet

Schaltet man die Zentrale auf **stop** (am TwinCenter ist dafür die **stop**-Taste oben links zu betätigen 

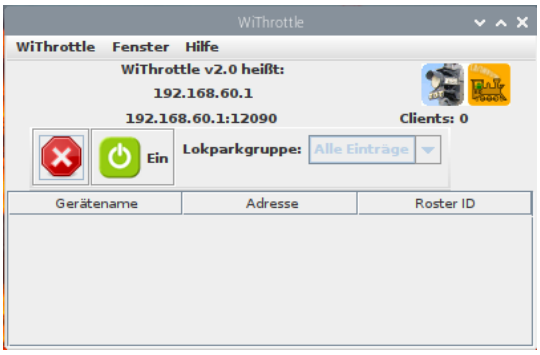
), sieht der wiThrottle-Server so aus:



Status: Eingeschaltet

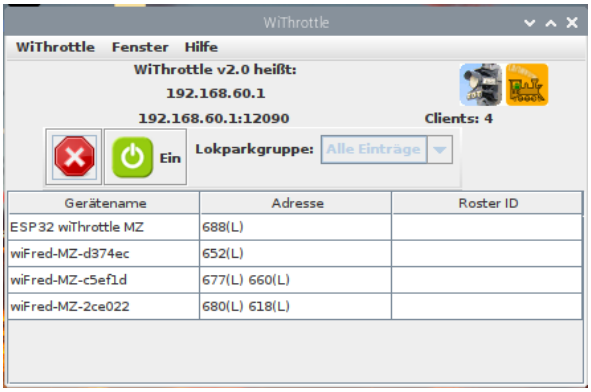
Schaltet man die Zentrale auf **go** (am TwinCenter ist dafür die **go**-Taste oben links zu betätigen 

), sieht der wiThrottle-Server so aus:



Anzeige: WLAN-Handregler

Haben sich WLAN-Handregler mit dem wiThrottle-Server erfolgreich verbunden, so werden diese in der Liste angezeigt, im Bild unten sind es vier Geräte (≙ vier Clients), die insgesamt sechs Fahrzeuge kontrollieren:



Anmerkungen

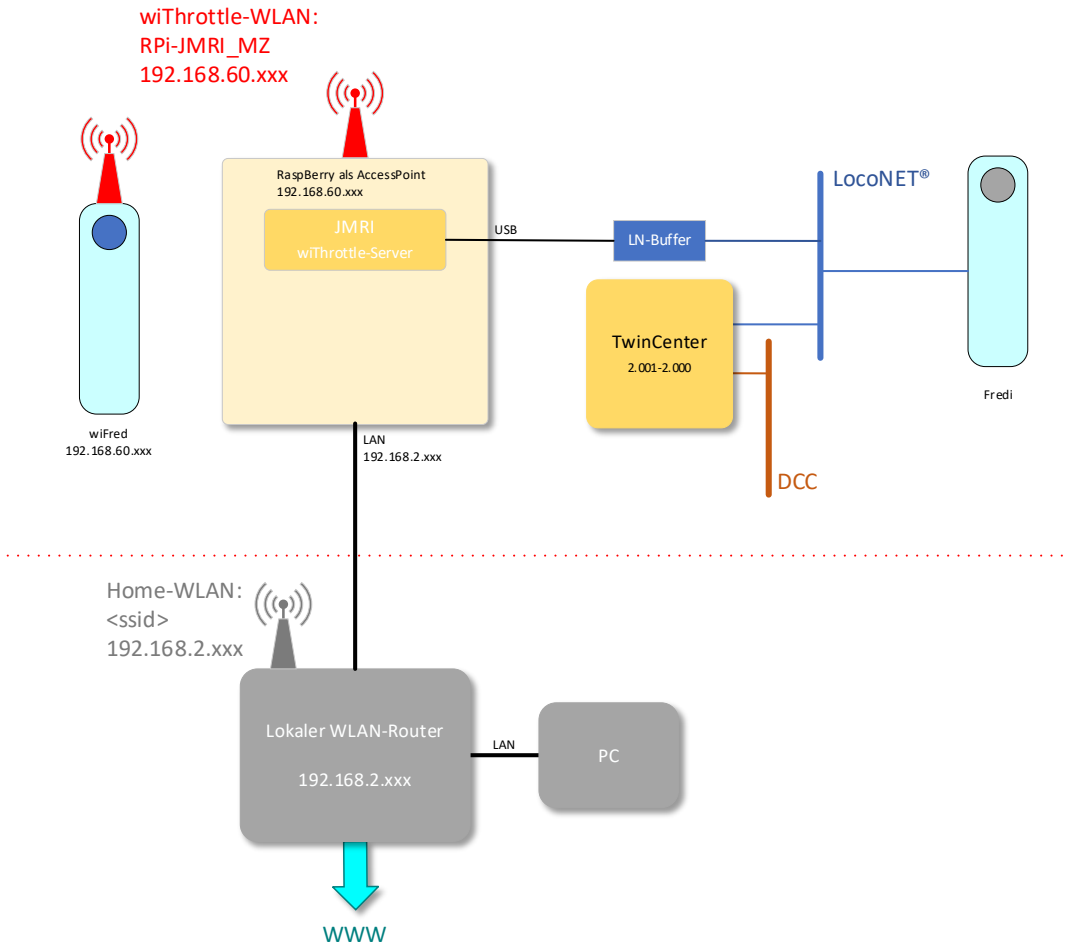
- **GANZ WICHTIG**: LocoNET® und DCC-Zentrale müssen vor jedem Einschalten des RaspPi angeschlossen und eingeschaltet sein!
- Der wiThrottle-Server startet nur, wenn eine Verbindung zu einer Zentrale hergestellt ist.
- Durch Betätigen der **go** / **stop** – Taste muss sich die Server-Anzeige der Betriebsart der Zentrale entsprechend ändern (siehe oben).
- Der RaspBerry kann Hardware bedingt maximal acht Clients verwalten; ein Client entspricht einem Gerät – auch wenn ein Gerät mehrere Fahrzeuge kontrollieren kann.

Anhang B: Michaels WLAN-Handregler

Modell	MAC	Name	IP
wiThrottle (A.Heckt)	98:F4:AB:13:41:94	ESP32 WiThrottle MZ	192.168.60.71
wiFred rev0.51 LiPo	3C:61:05:D3:74:EC	wiFred-MZ-d374ec	192.168.60.91
wiFred rev0.51 LiPo	E8:DB:84:C5:EF:1D	wiFred-MZ-c5ef1d	192.168.60.84
wiFred rev0.62	68:67:25:2C:E0:22	wiFred-MZ-2ce022	192.168.60.56

Anhang C: Verbindung der Geräte

Die Verbindung zu einer (LocoNET®-fähigen) DCC-Zentrale erfolgt über das LocoNET®:



Die Verbindung vom RaspBerry zum lokalen WLAN-Router und zum PC ist optional und erforderlich, wenn eine SSH-Verbindung mit dem PPC erfolgen soll.

Anhang D: Raspberry von USB-Stick starten

(Quelle: https://www.elektronik-kompendium.de/sites/raspberry-pi/2404241.htm)

Manche Anwendungen auf dem Raspberry Pi führen auf Dauer dazu, dass die eingesetzte SD-Karte Schaden nehmen kann. Der Grund ist, dass der darin enthaltene Flash-Memory nicht sehr viele Schreib-Zyklen aushält. Das ist auch klar, weil eine SD-Karte eigentlich für Digitalkameras gemacht sind und dort nur ein paar Fotos und Filme gespeichert werden. In so einem Anwendungsfall hält eine SD-Karte ewig.
Als Speicher für ein Betriebssystem ist eine solche Speicherkarte nicht geeignet. In solchen Fällen macht es Sinn den Inhalt der SD-Karte auf eine USB-Festplatte oder USB-Stick umzuziehen und den Raspberry Pi von dort zu booten.
Ein weiterer Vorteil ist, dass die Geschwindigkeit im Vergleich zu gängigen SD-Karten deutlich besser ist.
Wichtig ist, dass der USB-Port am Raspberry Pi zum Zeitpunkt des Starts nur 600 mA liefert. Alle angeschlossenen USB-Geräte dürfen also beim Einschalten und später im laufenden Betrieb zusammen nicht mehr Strom ziehen. USB-Sticks und SSDs kommen in der Regel mit weniger aus. Anders bei Festplatten. Hier sollte man den Stromverbrauch vorher prüfen.
Folgende Schritte sind durchzuführen:

1. Raspberry Pi für das Booten von einem USB-Laufwerk aktivieren.
2. Inhalt der SD-Karte auf ein USB-Laufwerk umziehen (kopieren).
3. Raspberry Pi mit USB-Laufwerk in Betrieb nehmen.

Raspberry Pi für das Booten von einem USB-Laufwerk aktivieren

Der betreffende Raspberry Pi muss einmalig mit Raspbian von einer SD-Karte gebootet werden. Den USB-Boot-Modus aktiviert man mit einem Parameter in der Konfigurationsdatei „/boot/config.txt“.

```
echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt
```

Wichtig ist jetzt, den Raspberry Pi einmal neu zu starten, damit er diese Einstellung übernimmt.

```
sudo reboot
```

Danach prüft man, ob der Parameter korrekt gesetzt wurde.

```
vcgencmd otp_dump | grep 17:
```

Die Ausgabe sollte „17:3020000a“ sein. Dann kann man den Raspberry Pi herunterfahren und die SD-Karte entfernen.

USB-Laufwerk vorbereiten

In der Regel hat man ein eingerichtetes System auf einer SD-Karte. Den Inhalt muss man in eine Image-Datei sichern und anschließend auf das USB-Laufwerk schreiben.

Raspberry Pi mit USB-Laufwerk in Betrieb nehmen

Wichtig ist, dass die SD-Karte entfernt wurde. Die ist nämlich beim Boot-Vorgang bevorrechtigt. Von einem USB-Laufwerk wird nur dann gebootet, wenn keine SD-Karte steckt. Das USB-Laufwerk mit dem beschriebenen Image steckt man an einen freien USB-Port und nimmt den Raspberry Pi in Betrieb. Er sollte dann vom USB-Laufwerk booten. Wenn das

Laufwerk über eine Aktivitäts-LED verfügt, sollte die nach ein paar Sekunden blinken. Das ist ein gutes Zeichen.

Troubleshooting

Wenn der Raspberry Pi scheinbar nicht in Betrieb geht, sollte man ihn zumindest an einem Monitor in Betrieb nehmen, um den Bootvorgang zu prüfen.
Probleme sollte es nur dann geben, wenn die angeschlossenen USB-Geräte beim Einschalten zu viel Strom ziehen oder der Laufwerks-Controller zu lange braucht, um in Betrieb zu gehen. Für beides gibt es eine Lösung.

- Raspberry Pi: USB-Strombegrenzung aufheben
- Raspberry Pi: Timeout für USB-Laufwerke verlängern

Ergänzungen

- Duplizieren einer SD-Card mit dem Raspberry Pi
- Raspberry Pi: Raspbian auf eine SD-Speicherkarte installieren (Windows)
- Raspberry Pi: Raspbian auf eine SD-Speicherkarte installieren (Linux oder macOS)