

Plugin-Daten	
Autor	Michael Schlenstedt
Logo	
Status	UNSTABLE
Version	0.1.0
Min. LB Version	3.0.0
Pre-Release Download	https://github.com/mschlenstedt/LoxBerry-Plugin-Debmatic/archive/refs/tags/LoxBerry-Plugin-Debmatic-V0.1.0.zip
Beschreibung	Mit dem DebMatic Plugin können mit Hilfe der Aufsteck-Funkmodule von Homematic sämtliche Homematic Sensoren und Aktoren in das Loxonesystem integriert werden.
Sprachen	DE, EN
Diskussion	https://www.loxforum.com/forum/projektforen/loxberry/plugins/124452-plugin-homematic-auf-dem-loxberry

DebMatic Plugin

[Version History...](#)

<https://github.com/mschlenstedt/LoxBerry-Plugin-Debmatic/releases>

Funktion des Plugins

Das Plugin installiert [DebMatic von Alexander Reinert](#) auf dem LoxBerry. Damit steht eine vollwertige Homematic CCU3 auf dem LoxBerry bereit, die über Node Red und MQTT direkt mit dem LoxBerry bzw. Miniserver kommuniziert. Sämtliche Homematic Sensoren und Aktoren können ausgelesen und gesteuert werden. Zur Nutzung des Plugins ist ein Homematic Funkadapter notwendig, den es als Modul für den Raspberry Pi in verschiedenen Ausführungen gibt (siehe unten unter Hardware). Die Anbindung der Funkmodule kann anstelle über die GPIO Leiste auch über die USB- oder Ethernet-Platine von Alexander Reinert erfolgen. Damit ist ein Betreiben des Plugins auch in einer VM möglich. Eine separate CCU2/CCU3 ist nicht mehr nötig. Die Sensoren/Aktoren stehen als vollwertige Geräte in Loxone Config zur Verfügung.

Download

- Das Plugin-Archiv (ZIP) kann auf GIT-Hub heruntergeladen werden:
<https://github.com/mschlenstedt/LoxBerry-Plugin-Debmatic/releases>
- Der Sourcecode ist auf GitHub verfügbar:
<https://github.com/mschlenstedt/LoxBerry-Plugin-Debmatic>

Installation

Das Plugin wird über die Pluginschnittstelle installiert. Es werden spezielle Kernelmodule während der

Installation aus den Quelltexten kompiliert. Wird der Kernel bei einem LoxBerry update erneuert, ist ein Update des Plugins notwendig, damit für den neuen Kernel die entsprechenden Kernelmodule neu kompiliert werden.

Da verschiedene Komponenten während der Installation aus den Quelltexten kompiliert werden, kann die Installation etwas Zeit in Anspruch nehmen. Unterstützt werden aktuell folgende Hardware-Architekturen: **Arm64, Arm7l, x64**.

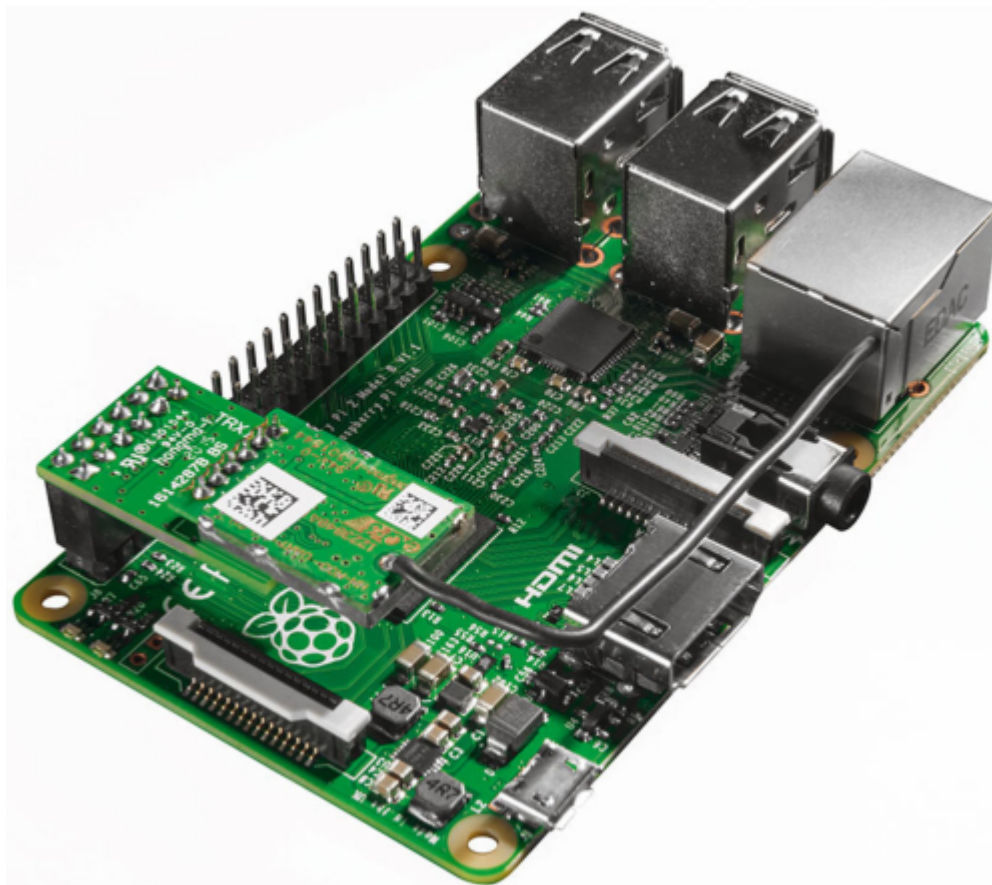
Unterstützte Hardware

Architekturen

Das Plugin läuft auf den folgenden Hardware Architekturen:

- **Arm64**, z. B. Raspberry Pi, Odroid, Rock Pi
- **Arm7l**, z. B. Raspberry Pi 2
- **x64**, z. B. VMs (getestet: Virtual Box und Proxmox, andere sollten ebenfalls laufen)

Funkmodul HM-MOD-RPI-PCB (Älteres Modul)

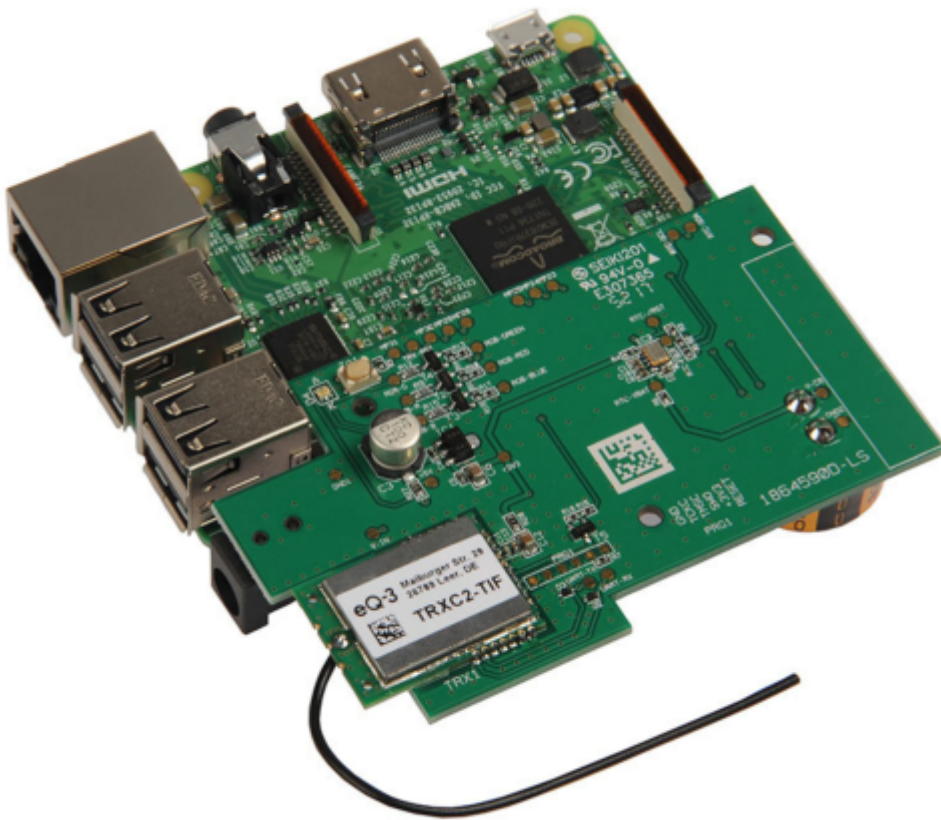


Aufgestecktes HM-MOD-RPI-PCB auf einem Raspberry Pi, Quelle: [ELV Webshop](#)

Das ältere Funkmodul HM-MOD-RPI-PCB als Aufsteckvariante für die GPIO-Schnittstelle des Raspberry Pi wird von DebMatic voll unterstützt. Das Modul besteht aus zwei Einzel-Platinen, die von [eQ-3 über den Webshop von ELV](#) (als Bausatz ab etwa 20€) verkauft werden. Der Zusammenbau gelingt leicht. Das Funkmodul funktioniert auch auf anderen Single Board Computern, deren GPIO-Leiste kompatibel mit der des Raspberry Pi sind:

<https://github.com/alexreinert/debmatic#voraussetzung-f%C3%BCr-hm-mod-rpi-pcb-und-rpi-rf-mod>

Funkmodul RPI-RF-MOD (Neues Modul)



Aufgestecktes RPI-RF-MOD auf einem Raspberry Pi, Quelle: [ELV Internetshop](#)

Das neue Funkmodul RPI-RF-MOD als Aufsteckvariante für die GPIO-Schnittstelle des Raspberry Pi wird von DebMatic voll unterstützt. Gegenüber dem älteren Modul soll das neue Modul eine bessere Reichweite besitzen und bringt zudem eine Real Time Clock mit. Zudem hat das Modul eine Status RGB-LED sowie eine Reset-Taste. Das Modul wird von [eQ-3 über den Webshop von ELV](#) (als Bausatz ab etwa 40€) verkauft. Der Zusammenbau gelingt leicht. Das Funkmodul funktioniert auch auf anderen Single Board Computern, deren GPIO-Leiste kompatibel mit der des Raspberry Pi sind:

<https://github.com/alexreinert/debmatic#voraussetzung-f%C3%BCr-hm-mod-rpi-pcb-und-rpi-rf-mod>



Raspberry 4 PROBLEM

Das neue Funkmodul RPI-RF-MOD passt aufgrund der veränderten Position der Netzwerkbuchse nicht mehr direkt auf den Pi4. Mit einem einfachen Stacking Header kann man das **RPI-RF-MOD** aber einwandfrei nutzen.

Zusatz-Platine HB-RF-USB-TK von Alexander Reinert



HB-RF-USB-TK Platine, Quelle: [Smartkram Webshop](#)

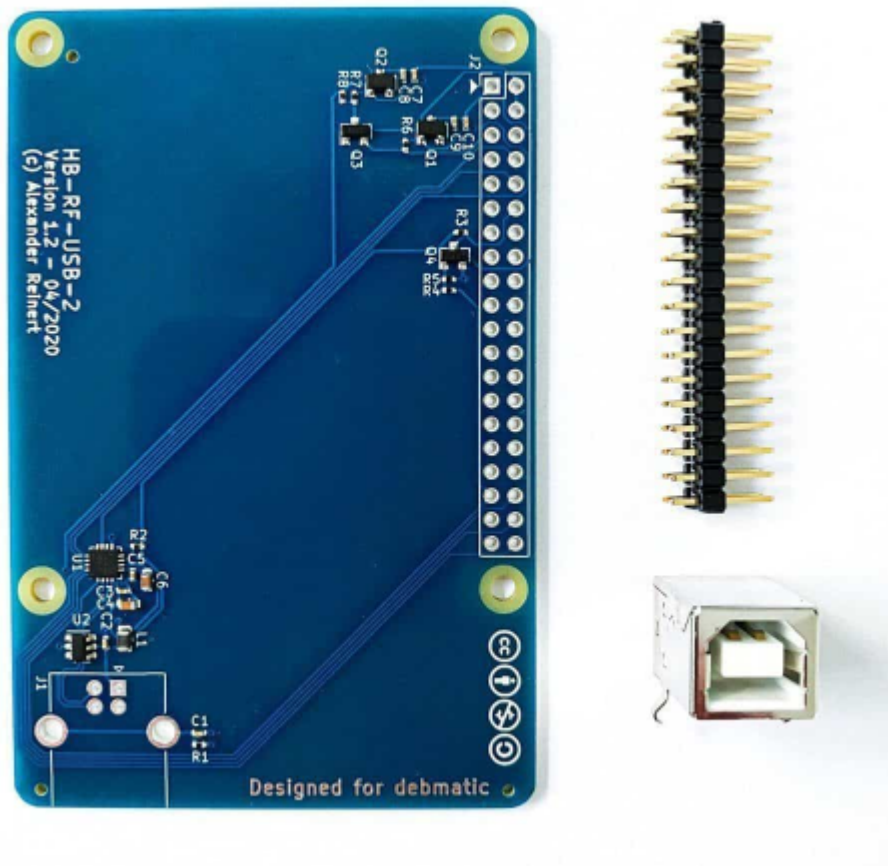
Diese USB-Platine ist eine Eigenentwicklung von Alexander Reinert! Sie bietet eine USB-Schnittstelle zu den Original-GPIO-Aufsteckmodulen von eQ3. Es ist also zusätzlich zur Platine auch noch die Aufsteckplatine RPI-RF-MOD bzw. HM-MOD-RPI-PCB notwendig (siehe oben). Der große Vorteil liegt darin, dass damit die GPIO-Leiste des Raspberry für andere Erweiterungen frei bleibt und auf Grund der freien Platzierung des Aufsteckmoduls kann dieses außerhalb vom Schaltschrank installiert werden. Passende Gehäuse [gibt es auf Thingiverse](#).

Die Designvorlagen für die Platine sind frei zugänglich: <https://github.com/alexreinert/PCB/tree/master>

Die Platine kann auch als Bausatz fertig bestellt werden:

<https://smarkram.de/hardware-shop/produkte/bauteile-zentralen/diy-bausaetze/platine-hb-rf-usb-tk/>

Zusatz-Platine HB-RF-USB-2 von Alexander Reinert



HB-RF-USB-2 Platine, Quelle: [Smartkram Webshop](#)

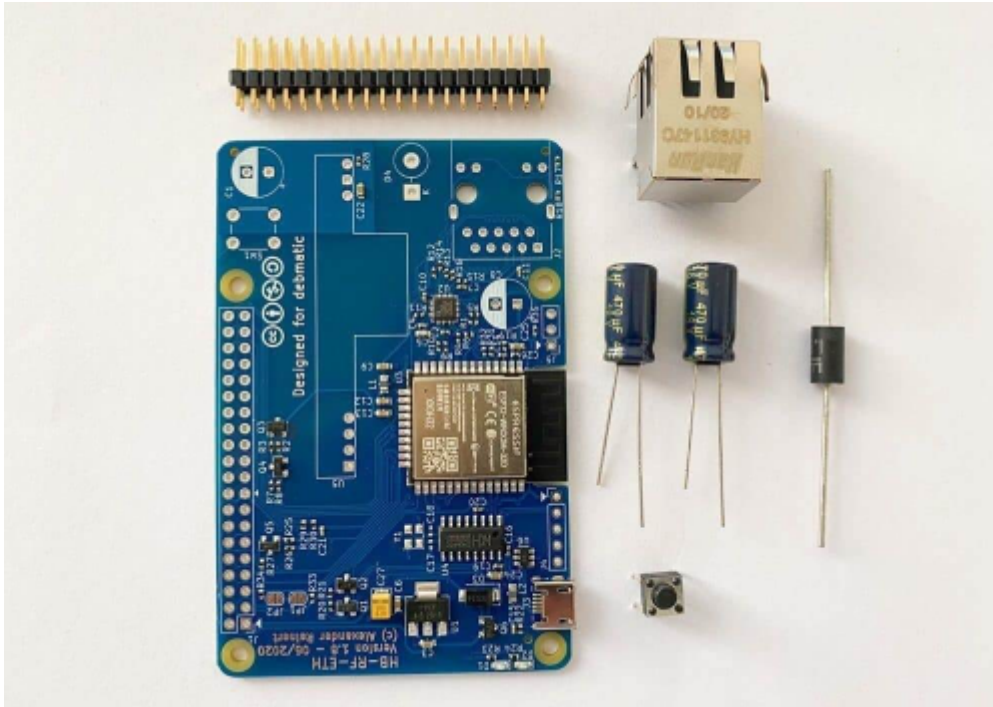
Diese USB-Platine ist eine Eigenentwicklung von Alexander Reinert! Sie bietet eine USB-Schnittstelle zu den Original-GPIO-Aufsteckmodulen von eQ3. Es ist also zusätzlich zur Platine auch noch die Aufsteckplatine RPI-RF-MOD bzw. HM-MOD-RPI-PCB notwendig (siehe oben). Der große Vorteil liegt darin, dass damit die GPIO-Leiste des Raspberry für andere Erweiterungen frei bleibt und auf Grund der freien Platzierung des Aufsteckmoduls kann dieses außerhalb vom Schaltschrank installiert werden. Passende Gehäuse [gibt es auf Thingiverse](#)

Die Designvorlagen für die Platine sind frei zugänglich: <https://github.com/alexreinert/PCB/tree/master>

Die Platine kann auch als Bausatz fertig bestellt werden:

<https://smarkram.de/hardware-shop/produkte/bauteile-zentralen/diy-bausaetze/platine-hb-rf-usb-2/>

Zusatz-Platine HB-RF-ETH von Alexander Reinert



HB-RF-ETH Platine, Quelle: [Smartkram Webshop](#)

Diese Ethernet-Platine ist eine Eigenentwicklung von Alexander Reinert! Sie bietet eine Ethernet-Schnittstelle zu den Original-GPIO-Aufsteckmodulen von eQ3. Es ist also zusätzlich zur Platine auch noch die Aufsteckplatine RPI-RF-MOD bzw. HM-MOD-RPI-PCB notwendig (siehe oben). Der große Vorteil liegt darin, dass damit die GPIO-Leiste des Raspberry für andere Erweiterungen frei bleibt und auf Grund der freien Platzierung des Aufsteckmoduls kann dieses außerhalb vom Schaltschrank oder irgendwo im Gebäude installiert werden. Passende Gehäuse [gibt es auf Thingiverse](#)

Die Designvorlagen für die Platine sind frei zugänglich: <https://github.com/alexreinert/PCB/tree/master>

Die Platine kann auch als Bausatz fertig bestellt werden:

<https://smarkram.de/hardware-shop/produkte/bauteile-zentralen/diy-bausaetze/platine-hb-rf-eth/>

Konfigurationsoptionen

Das Plugin benutzt das Übertragungsprotokoll [MQTT](#). Daher muss das [MQTT Widget](#) korrekt konfiguriert sein. Bei der Konfiguration des [MQTT Widget](#) setzt bitte unter *MQTT Gateway* → *Gateway Settings* die folgenden 2 Optionen, damit die Daten vom DebMatic Plugin im Gateway korrekt aufbereitet werden:

Data transformations

- ☒ Convert booleans to 1 and 0
- ☒ Expand JSON data

Ihr solltet zudem einen Usernamen und ein Passwort für den MQTT Broker vergeben. Ihr benötigt beide Angaben später zur Konfiguration im Plugin.

MQTT Broker username
Default is **loxberry** for the local Mosquitto installation. Emptying user and password will disable authentication.

MQTT Broker password
Use a safe password! Click the dice to open a funny password generator: 

Im Plugin können die Ports der Homematic WebUI und von Node Red konfiguriert werden. Ändert diese Werte nur, wenn ihr wisst was ihr tut und wenn dieses bei Konflikten mit anderen Diensten unbedingt notwendig ist. Wenn ihr die Ethernet-Platine von Alexander Reinert nutzt, dann könnt ihr hier die IP-Adresse der Platine eingeben oder auch automatisch danach suchen lassen. Wenn das Plugin das Ethernet-Modul automatisch gefunden und aktiviert hat, ihr es aber z. B. in einer anderen Installation im Haus nutzen wollt, könnt ihr es hier wieder deaktivieren.

Settings

Homematic WebUI Port

Node Red WebUI Port

Enable HB-RF-ETH Module ☐ Off

IP Address HB-RF-ETH

Einrichtung Homematic CCU

Die Einrichtung der Homematic CCU wird hier nicht weiter beschrieben. Alles funktioniert wie auf einer Original-CCU von Homematic. Erweiterungen lassen sich auf DebMatic allerdings nicht installieren bzw. dieses muss über apt -get auf der Kommandozeile geschehen. Alle verfügbaren Erweiterungen (wie z. B. CuxD) sind aber bereits vorinstalliert. Näheres auf der [Debmatic Homepage](#).

Wer auf die XML-API zugreifen möchte, nutzt folgenden Link:

`https://<LOXBERRY IP>:8081/addons/xmlapi/info.html`



Vergebt für eure Geräte und für jeden einzelnen Ein- oder Ausgang entsprechende Namen. Diese Namen erscheinen später in der MQTT Anbindung und damit auch im Miniserver. Das vereinfacht hier die weitere Konfiguration und ihr könnt zudem später (z.



B. bei einem Defekt) einen Sensor/Aktor einfach austauschen, ohne dass ihr die weitere Konfiguration in der LoxoneConfig anpassen müsst.

Einrichtung Node Red / MQTT Anbindung

Für die Anbindung der Homematic CCU an das [MQTT Gateway](#) des LoxBerry und damit später an die Loxone Config nutzen wir Red Matic. Hier ist nur ein sehr kurzer Flow für eine komplette MQTT-Anbindung in beide Richtungen notwendig. Startet die Red Matic WebUI über das Plugin.

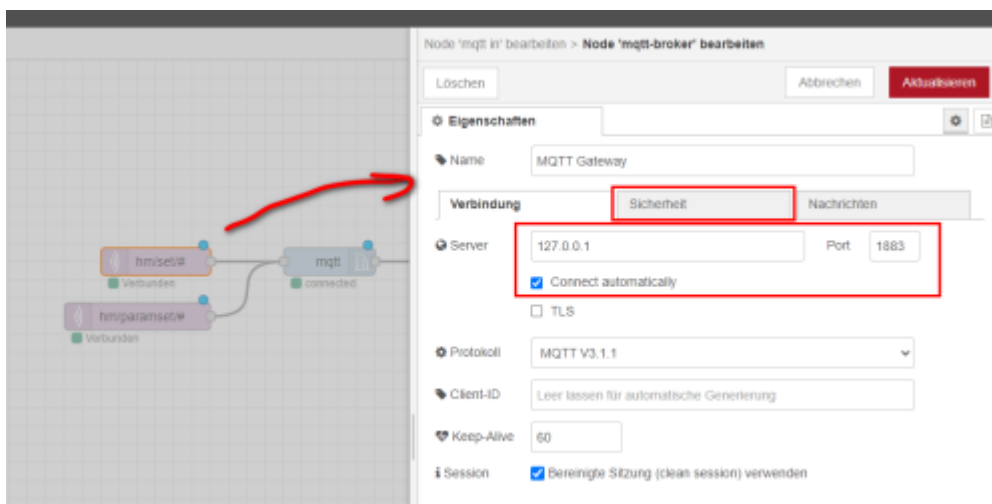
In Node Red seht ihr auf der linken Seite verschiedene Ein- und Ausgänge und Funktionsbausteine. Zunächst zieht ihr 2 Bausteine **Netzwerk → MQTT in** in euren Flow. Mit Doppelklick konfiguriert ihr folgende beiden Topics, an die ihr später entsprechende Befehle senden könnt, die von Homematic dann an die Aktoren weitergegeben werden:

hm/set/#

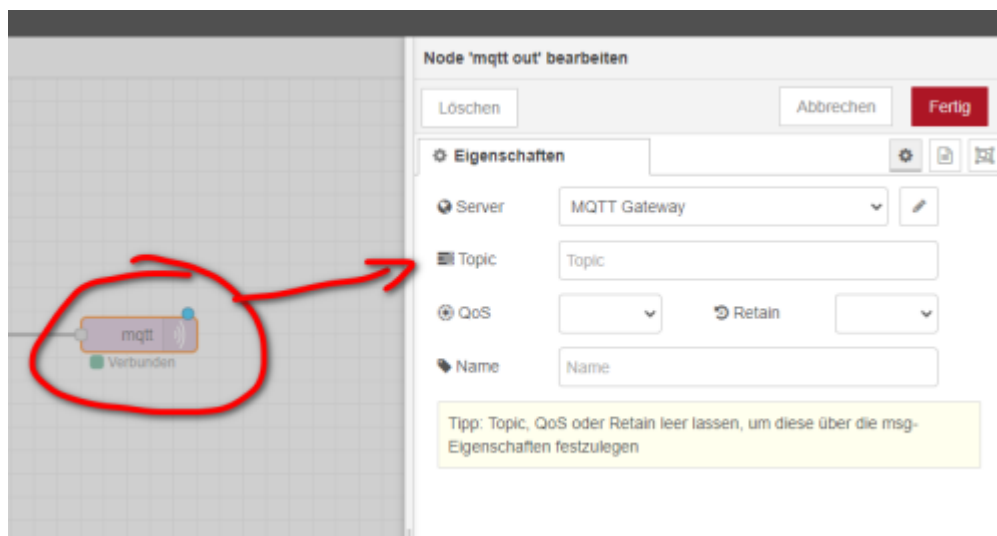
hm/paramset/#



Über den Bleistift neben Server konfiguriert ihr nun einmalig den MQTT Server. Diesen wählt ihr dann bei den folgenden Ein- und Ausgängen einfach aus - er muss also nur einmal eingerichtet werden. Die Serveradresse ist **127.0.0.1** (localhost), unter Sicherheit setzt ihr noch Username und Passwort für den MQTT Server, welche ihr aus dem [MQTT Widget](#) entnehmt:

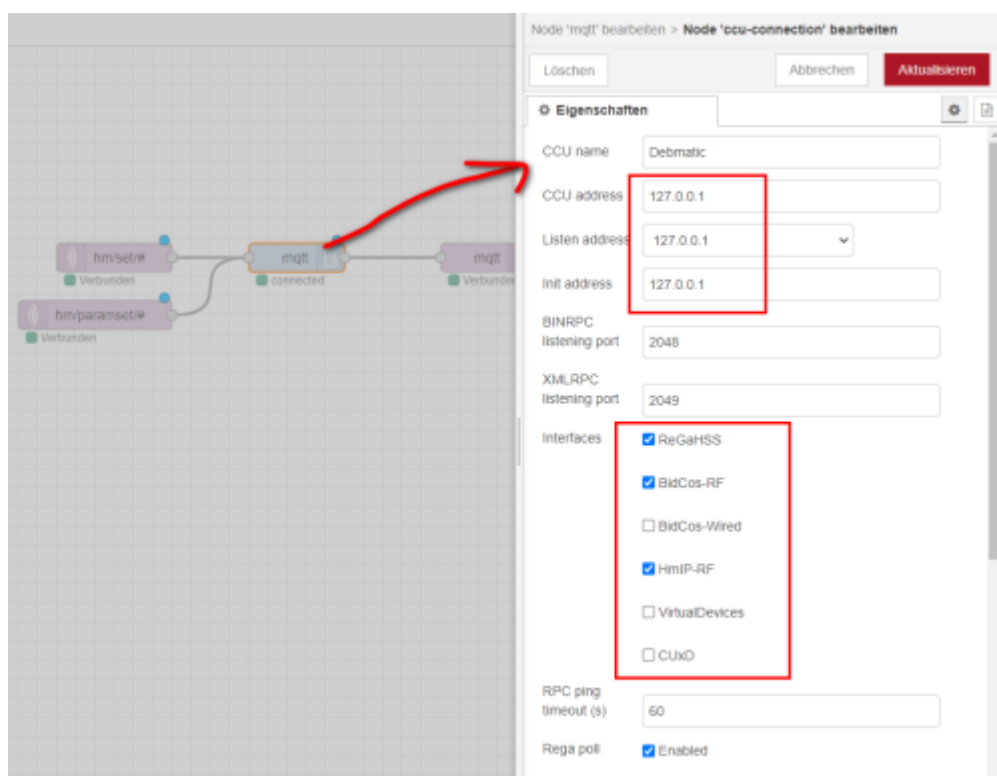


Als nächstes zieht ihr 1 Baustein **Netzwerk → MQTT out** in euren Flow. Dieser wird wie folgt konfiguriert:

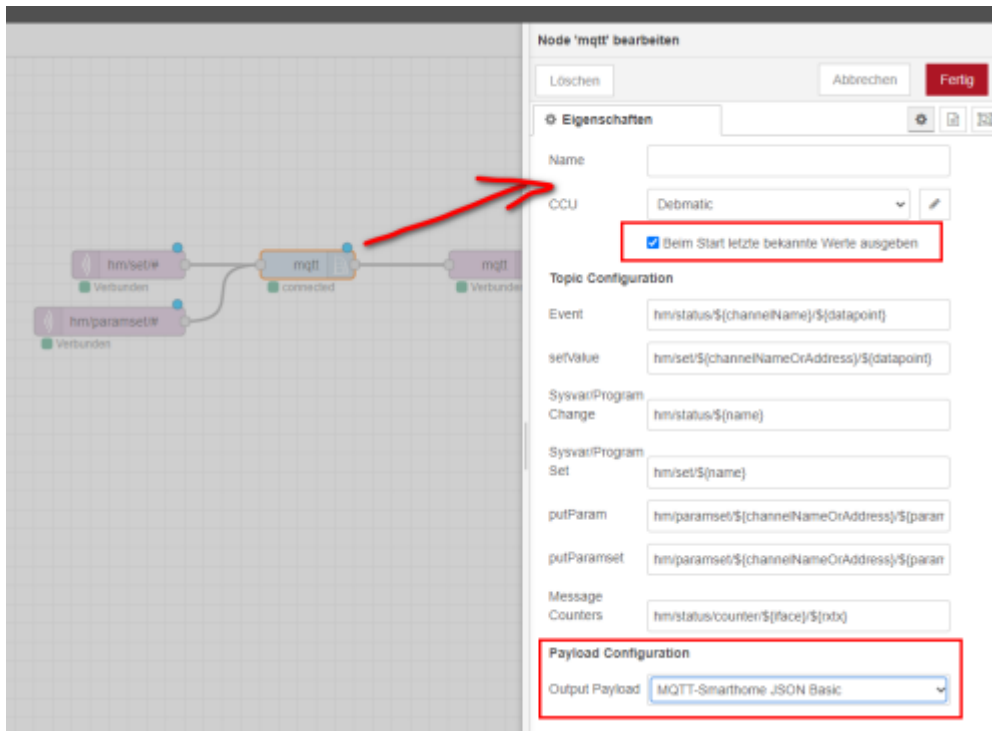


Als letztes zieht ihr noch unter **ccu → mqtt** einen CCU-MQTT-Baustein in die Config. An diesen Baustein verbindet ihr die drei oben erstellten Ein- und Ausgänge. Der MQTT-Baustein wird wie folgt konfiguriert:

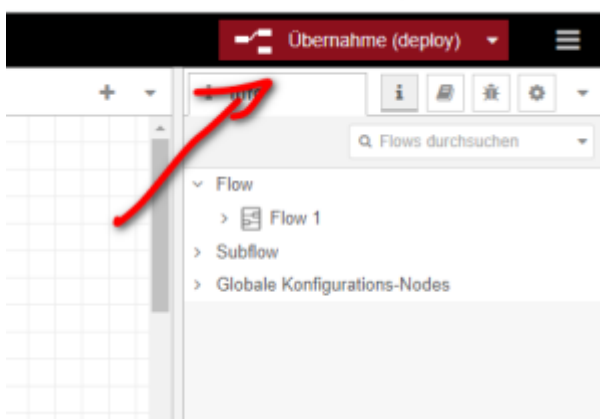
Zunächst fügt ihr über den Bleistift bei CCU eine neue CCU hinzu. Als IP Adresse wird hier **127.0.0.1** (localhost) eingetragen. Hakt weiter unten die Services an, die ihr über das MQTT Gateway übertragen wollt.



Anschließend konfiguriert ihr Euch den Node passend: Die Topics lasst ihr voreingestellt, je nachdem welche Daten ihr später in der Loxone Config weiterwenden wollt, setzt ihr unten JSON auf "Extended" (Achtung! **Sehr** viele Daten) oder "Basic" (empfohlen).



Zum Schluss aktiviert ihr euren neuen Flow über die Schaltfläche rechts oben "**Übernahme (deploy)**":



Keine Namen im MQTT Broker?



Nach der Ersteinrichtung von Sensoren/Aktoren in der CCU hakelt manchmal die Übertragung der selbst vergebenen Namen für Sensoren und Aktoren und die Geräte erscheinen in der Incoming Overview des MQTT Gateway noch mit den kryptischen Standardbezeichnungen oder gar ganz ohne Name. Richtet zunächst alle eure Sensoren fertig in der CCU ein und geht dann wie folgt vor:

Der Haken "Beim Start bekannte Werte ausgeben" muss im Node Red einmal raus-, dann "Node-Red deploy", und dann wieder reingenommen werden, wieder mit anschließendem "Node-Red deploy". Dann MQTT Cache im MQTT Gateway auf dem LoxBerry löschen (in der Incoming Overview) und die neuen Namen sind da!

Einrichtung in der Loxone Config Software

Die Einrichtung in der Loxone Config wird im MQTT Widget näher beschrieben und an dieser Stelle nicht weiter ausgeführt: [Eine SSH-Verbindung mit putty aufbauen / Shell-Zugriff](#)

Anschließend gebt ihr folgenden Befehl ein: `apt-get purge pivccu-modules-raspberrypi`

Nach einem Reboot stehen Euch die GPIOs wieder voll zur Verfügung.

Nutzung der GPIOs mit anderen Anwendungen

Wenn ihr das Homematic-Modul über einen HB-RF-USB oder HB-RF-ETH Adapter an Euren LoxBerry angeschlossen habt, dann könnt ihr die GPIOs natürlich noch anderweitig nutzen (z. B. mit einem 1-Wire-Aufsteckmodul oder ähnlichem). Standardmäßig blockiert aber DebMatic die GPIOs für sich. Damit Homematic die GPIOs freigibt, loggt Euch per SSH auf Eurem LoxBerry ein ([Eine SSH-Verbindung mit putty aufbauen / Shell-Zugriff](#)) und führt anschließend folgenden Befehl aus - danach einmal rebooten.

```
sudo apt-get purge pivccu-modules-raspberrypi
```

Roadmap

- ☐ Update von DebMatic über die Pluginoberfläche

Fragen stellen und Fehler melden

<https://www.loxforum.com/forum/projektforen/loxberry/plugins/124452-plugin-homematic-auf-dem-loxberry>

From:

<https://wiki.loxberry.de/> - **LoxBerry Wiki - BEYOND THE LIMITS**

Permanent link:

<https://wiki.loxberry.de/plugins/debmatic/start?rev=1712135911>

Last update: **2024/04/03 11:18**