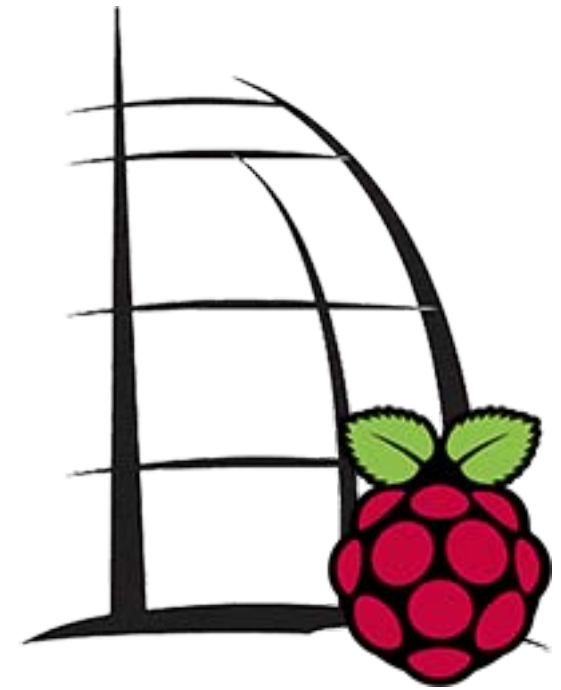


RasperryMatic (1)



Wieso/Weshalb/Warum?

Vor-/Nachteile

Entwicklungsstand

Alternativen

Roadmap

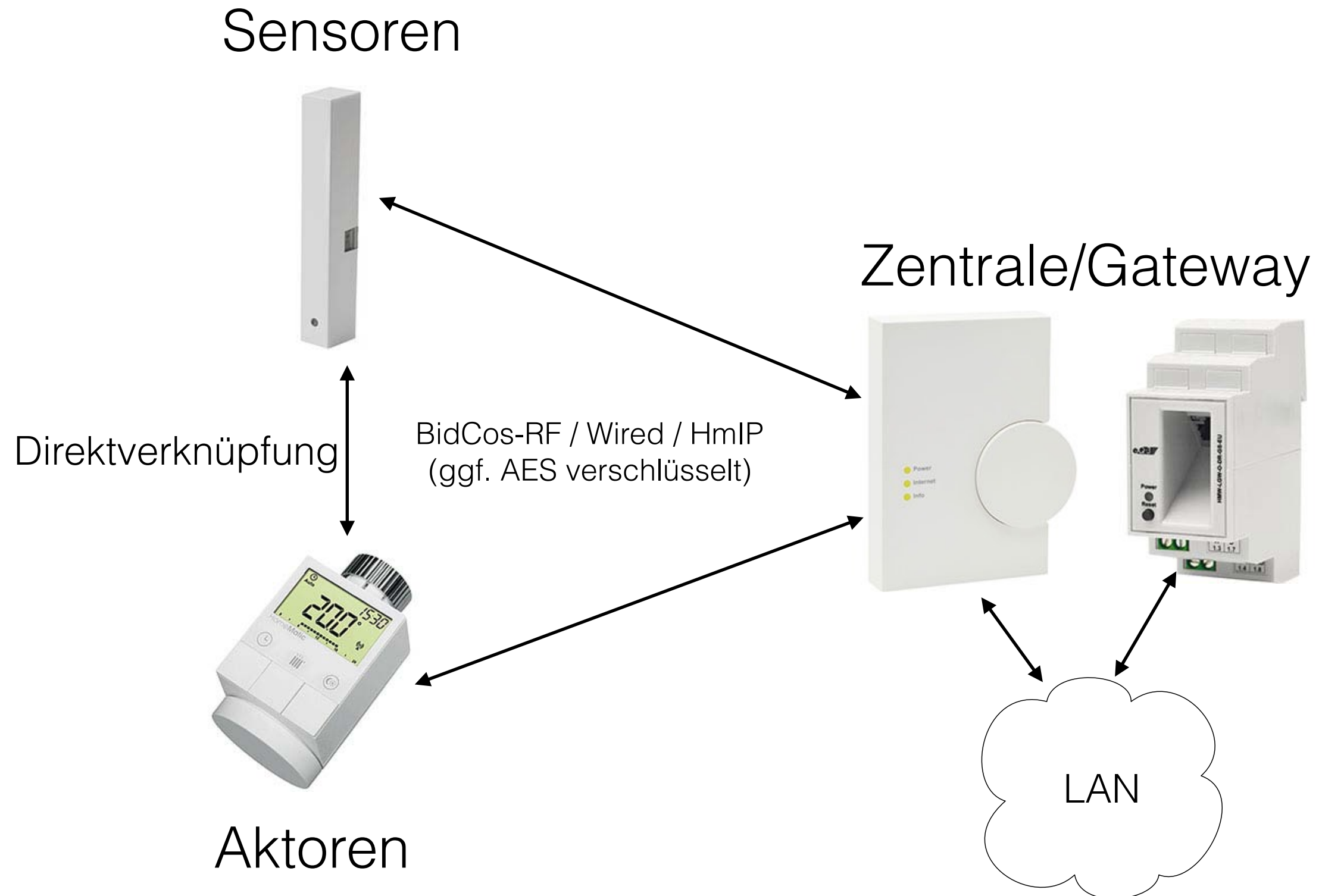
Jens Maus – mail@jens-maus.de

HomeMatic User-Treffen 2017, Kassel, 06. Mai 2017

Disclose Statement

- keinerlei Garantie auf Korrektheit bzw. technischer Unversehrtheit der eigenen Hardware/Software
- Lediglich Äußerung privater Meinungen – nicht die Meinung von eQ3, ELV oder anderer genannter Firmen und deren Produkte
- Erwähnte Hardwaremodifikationen können die Garantie aber auch die Zulassung dieser Hardware gefährden

HomeMatic CCU Architektur

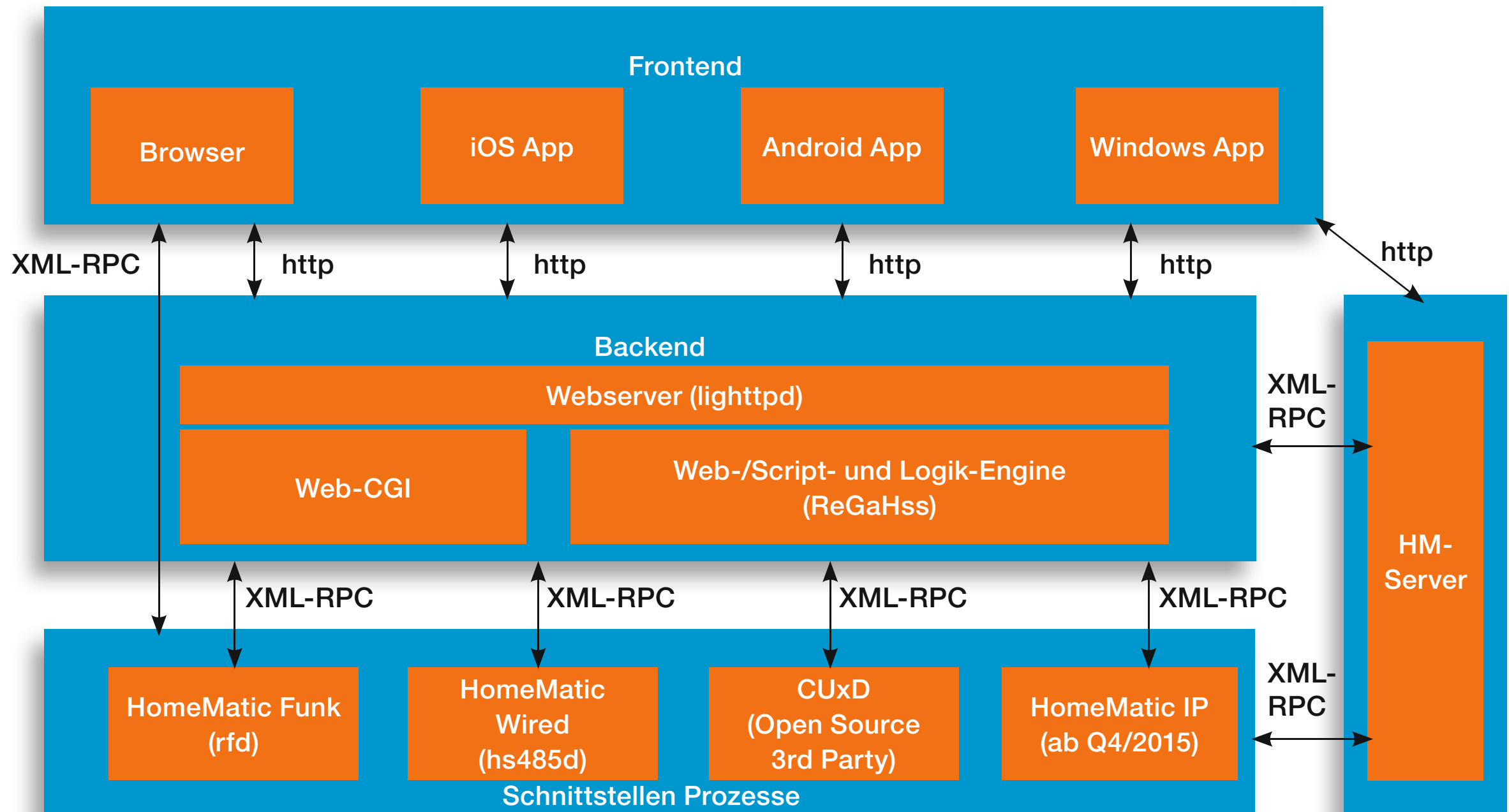


HomeMatic CCU

- CCU = **C**entral **C**ontrol/Communication **U**nit
- Markteinführung 2008 (CCU1) bzw. 2013 (CCU2)
- Erlaubt die zentrale Steuerung/Konfiguration von HomeMatic Komponenten mittels WebInterface (WebUI)
- Rudimentäre/grafische Programmierumgebung (inkl. proprietärer Skriptsprachumgebung)
- API Schnittstellen inkl. Remote Procedure Call Funktionalität (XML-RPC, BIN-RPC) für Anbindung externer Applikationen (z.B. CUxD, ioBroker, CCU.io, Smartphone Apps, etc.)
- Addon Schnittstelle für die Erweiterung von Funktionalität durch Drittanbieter



HomeMatic CCU Softwarearchitektur



Schnittstellenbeschreibung [1]

[1] ELV Journal 05/2015

CCU-Softwarearchitektur – Vorteile

- zentralisierte Steuerung aller HomeMatic Geräte
- rudimentäre grafische Programmierumgebung für die Implementierung einfachster Abläufe/Beziehungen zwischen HomeMatic-Komponenten (Anfängertauglich)
- einfachste Konfiguration / Einbindung in existierendes LAN
- Potentiell hohe Stabilität:
 - Betriebssystem basierend auf spezialisiertem Embedded Linux-System (buildroot [1])
 - Hardware-Watchdog

[1] buildroot – <https://buildroot.org>

CCU-Softwarearchitektur – Nachteile

- Nutzung sehr heterogener Softwareumgebungen
 - rfd / hs485d / ReGaHss – C-Programme
 - HMServer / HMIPServer – Java-Programme
 - Skripting – TCL / proprietäre Skriptsprache
- Kein vollumfängliches Multithreading (z.B. Nutzerskripte laufen sequentiell ab)
- Nur teilweise Offenlegung als „Open Source“ – OCCU [1] (Build-Umgebung Open Source, rfd / hs485d / ReGaHss / HMServer, etc. nur als Binaries)
- *Funktionserweiterungen rein über Addon-Schnittstelle
→ kein „Standard“-Linux (z.B. Debian, Raspbian)*

[1] OCCU – <https://github.com/eq-3/occu>

CCU-Softwarearchitektur – Nachteile

- Stabilitätsprobleme bei größeren Installationen, z.B.
 - kein Sandboxing von Addons
(Bugs in Addons können Stabilität der CCU gefährden)
 - Alle Prozesse laufen mit „root“-Rechten
 - keine Realtime Kernel Nutzung (PREEMPT_RT)
 - Bugs in der Programmierschicht (ReGaHss) bringen WebUI mitunter zum Absturz
- teils gravierende Bugs/Limitationen in der Programmierumgebung (ReGaHss)

CCU2 Hardware – Vorteile

- sehr gut für HomeMatic Einsteiger
- immer noch für kleine bis mittlere HomeMatic-Umgebungen ausreichend (~ max. 50 Geräte)
- Konstanter Support / Updates durch eQ3 (auch über 2017 hinaus)



CCU2 Hardware – Nachteile

- veralteter Hardwarestand
 - 32bit ARM926EJ-S (Single Core)
(ARMv5 454 MHz – ~2001/2 Markteinführung)
 - 256 MB RAM
 - dadurch gerade bei großen Installation (> 50 Geräte) sehr langsame WebUI bzw. Verarbeitung von Ereignissen, zunehmende Stabilitätsprobleme
- ➔ **Motivation: Bei größeren Installation / Wunsch nach mehr Flexibilität stößt man schnell an Grenzen der CCU2-HW**



Potentielle Vorteile von CCU2 Alternativen

- moderne Hardware

RaspberryPi3 – 1,2GHz QuadCore CPU
1 GB RAM
WLAN / Bluetooth



- Einsatz moderner Linux-Umgebungen für mehr Stabilität / Erweiterbarkeit
(aktualisiertes buildroot, aktueller Linux-Kernel, USB WLAN etc.)
- Unabhängigkeit + mehr Flexibilität
(vielfältigere Anwendungsszenarien)
- große Community mit möglichen Software/Hardwareerweiterungen

Alternative CCU-Software Umgebungen

(LXCCU)
(2014)

OCCU(-SDK)
(2015)

RaspberryMatic
(2015)

YAHM
(2016)

oder:



CCU2 Alternativen: Funkanbindung

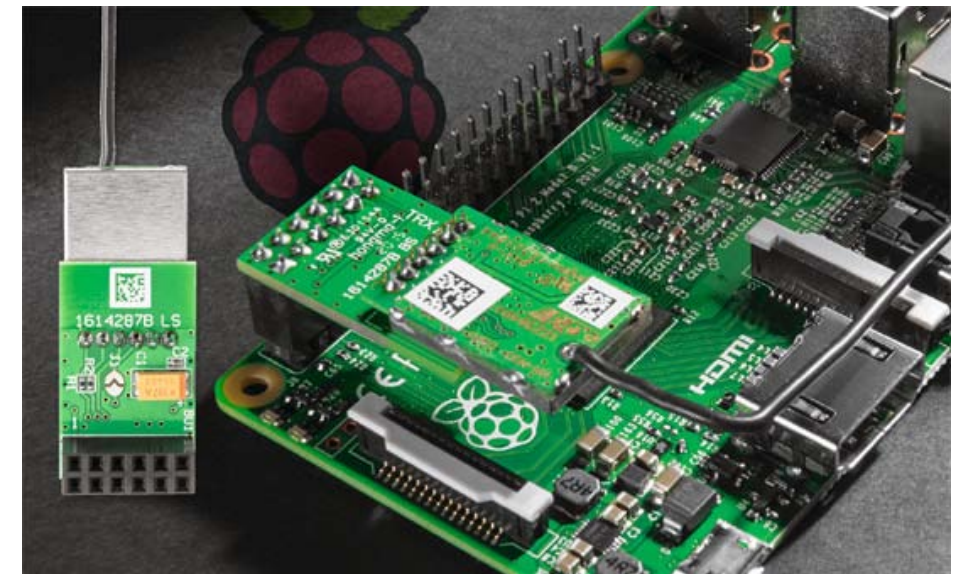
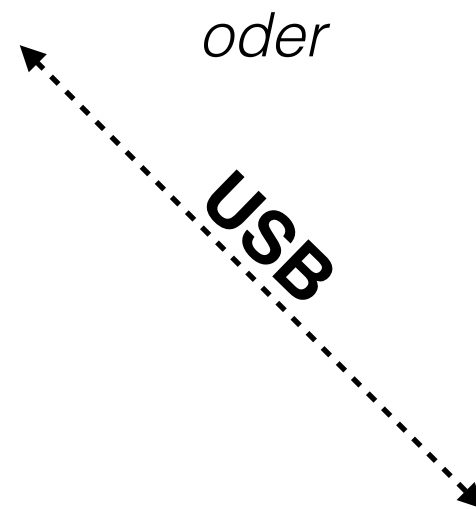
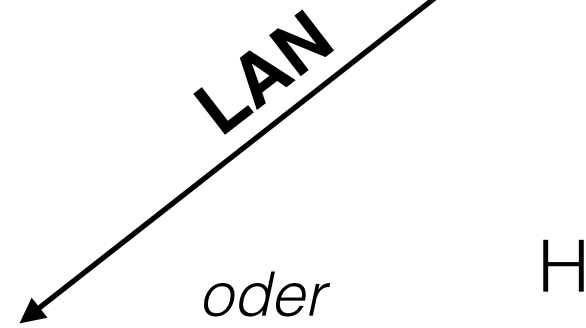


Raspberry Pi2 oder Pi3

HomeMatic LAN-Gateway



HomeMatic RPi Funkmodul



CUL-Stick (868 MHz)

OCCU-SDK [1]

Open Central Control Unit SDK



CCU1



CCU2



RPi
ARM



ARM/x86

Virtualisierungsansatz



YAHM [2]

➔ 100% Community

Distributionsansatz

RaspberryMatic [3]

➔ ~40% eQ3
~60% Community

Integrationsansatz

OCCU [4]

➔ 100%
Community



[1] <https://github.com/eq-3/occu>

[2] <https://homematic-forum.de/forum/viewforum.php?f=67> / <https://github.com/leonsio/YAHM>

[3] <https://homematic-forum.de/forum/viewforum.php?f=65> / <https://github.com/jens-maus/RaspberryMatic>

[4] <https://homematic-forum.de/forum/viewforum.php?f=55>



Integrationsansatz: **OCCU** – open central control unit [1]

- Methode: Direkte Integration/Installation aller CCU Dienste (ReGaHss/WebUI, rfd, hs485d, etc.) in bestehendes Linux System (z.B. unter Raspbian/Debian, z.B. als *.deb Pakete)
- Vorteile: Direkte Interaktion zwischen CCU-Architektur und Raspbian/Debian Betriebssystem bzw. Drittsoftware möglich
- Nachteile: Hohes Linux-KnowHow notwendig, Aufwendiges extrahieren aller notwendigen Programme/Dienste aus OCCU-SDK; Updatefähigkeit unklar; Host-OS Updates/Stabilität gefährden CCU Funktionalität
- Installation: Schwierig/Manuell, noch kein separates Projekt verfügbar

[1] <https://homematic-forum.de/forum/viewforum.php?f=55>

Virtualisierungsansatz: **YAHM** – Yet Another Homematic Management [1]

- Methode: Virtualisierung (mittels Linux Container – LXC) der kompletten CCU Firmware innerhalb einer existierenden Linux-Umgebung (z.B. unter Raspbian/Debian) mit Schnittstellenanpassungen an das Host-OS
- Vorteile: Updatefreundlich (basiert auf offiziellen eQ3 CCU2 Firmware Releases); klare Trennung zwischen CCU Umgebung und Host-OS; potentielle Unterstützung anderer Linux-Derivate; Nutzung aller CCU2-Addons ohne Anpassungen möglich
- Nachteile: Mittleres Linux-KnowHow (Kommandozeile) notwendig; Raspbian/Debian Updates mit Vorsicht zu genießen; direkte Nutzung von CCU2 Firmware-Updates problematisch; Bei zu hoher Auslastung des Host-OS Gefährdung der CCU Funktionalität / Stabilität; Host-OS Updates können Einfluss auf CCU Funktionalität haben; Nutzung der Soft-Float Binaries aus CCU2
- Installation: Mittel bis Schwer (Kommandozeile)

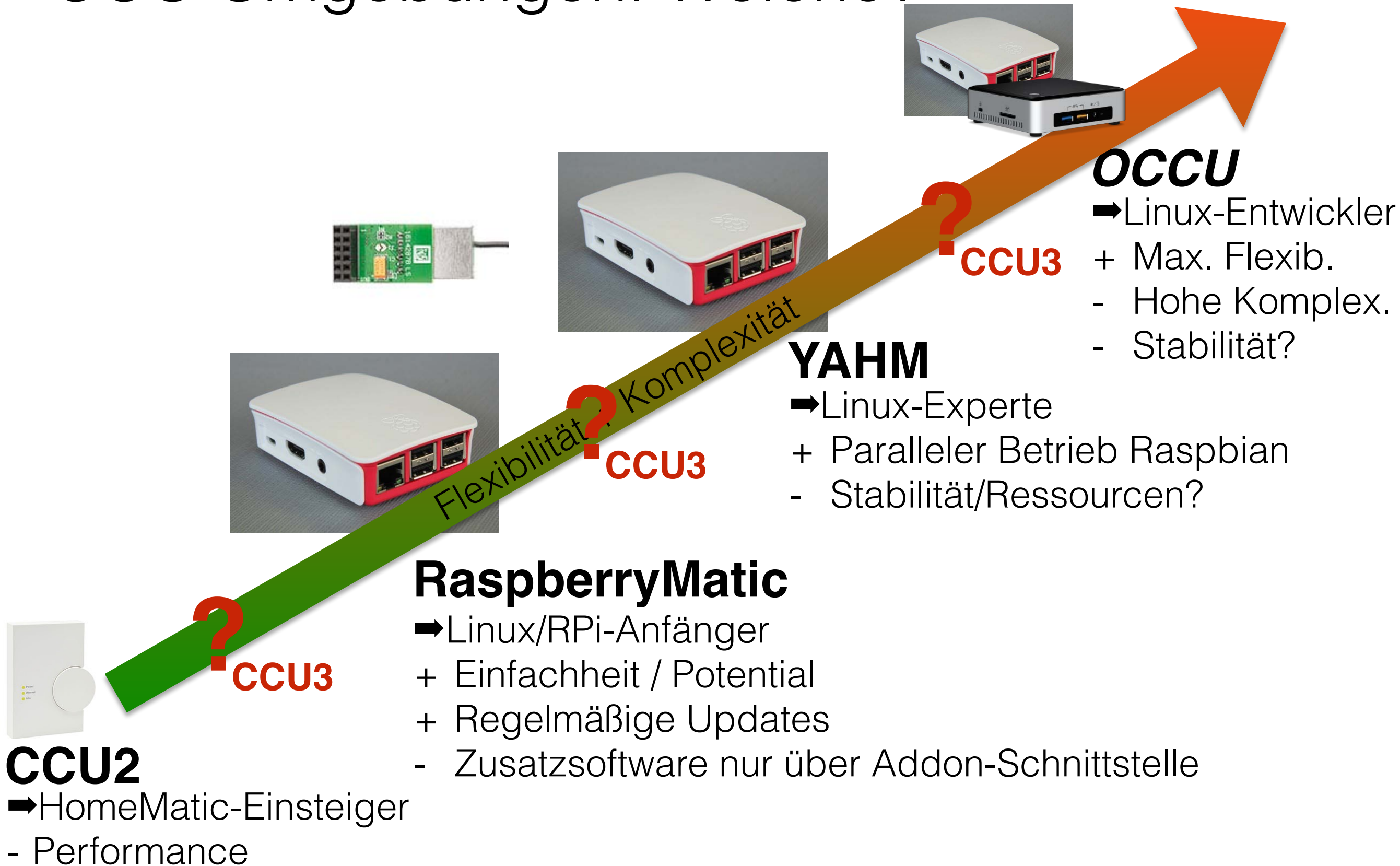
[1] <https://homematic-forum.de/forum/viewforum.php?f=67> / <https://github.com/leonsio/YAHM>

Distributionsansatz: **RaspberryMatic** ^[1]

- Methode: Distribution einer für RaspberryPi 2 und 3 angepassten OCCU Umgebung mit Unterstützung des eQ3-RPi-Funkmoduls bzw. LAN Gateway
- Vorteile: Simple Installation (microSD Kartenimage + Funkmodul); regelmäßige Verteilung von Updates; Zukünftige CCU Referenzplattform; Hohe Stabilität durch Anpassungen an RaspberryPi Hardware; Hohe Performance durch Einsatz von hard-float Binaries + modernem Linux-Kernel, etc.
- Nachteile: Datenübernahme von CCU1+CCU2 semi-automatisch; *Zusatzsoftware nur als CCU-Addon möglich; Updates/Upgrades nur via Backup → Neuinstallation → Restore möglich*
- Installation: Einfache Installation – selbst für Linux/RaspberryPi-Anfänger

[1] <https://homematic-forum.de/forum/viewforum.php?f=65> / <https://github.com/jens-maus/RaspberryMatic>

CCU Umgebungen: Welche?



RaspberryPi-basierte CCU Umgebungen: Limitationen

- Wie bei CCU2: Schlechte Empfangs/Sendequalität mit Funkmodul
 - ➔ Tausch der „Stummelantenne“ gegen leistungsstarken Ersatz [1,2]
- RaspberryPi hat keine RTC-Clock
 - ➔ Einsatz eines zusätzlichen RTC-Modules am GPIO empfehlenswert [3]
- SD Karten Einsatz problematisch (geringe Lebensdauer)
 - ➔ Deshalb Read-Only Root-FS bei RaspberryMatic, Logfiles im RAM, etc.
 - ➔ Regelmäßige Backups (1x pro Woche) empfehlenswert
- RaspberryPi lediglich 1GB RAM
 - ➔ Für CCU-Funktionen ausreichend, aber bei Parallelbetrieb anderer Software (z.B. bei Nutzung YAHM, ioBroker, etc.) problematisch
 - ➔ deshalb bei RaspberryMatic Zusatzsoftware nur via CCU-Addon



[1] <https://www.stall.biz/project/externe-antenne-stabantenne-fuer-raspberrymatic-occu-hm-mod-rpi-pcb>

[2] <https://homematic-forum.de/forum/viewtopic.php?f=27&t=27287>

[3] <https://homematic-forum.de/forum/viewtopic.php?f=56&t=27684> / http://www.piface.org.uk/products/piface_clock/

RaspberryMatic: Entwicklungshistorie

- *Ende 2015:* Erste Betaversion von eQ3 mit rudimentären BidCos-RF Funktionen (kein HomeMatic-IP)
- *August 2016:* Übernahme der Weiterentwicklung durch Community (<https://github.com/jens-maus/RaspberryMatic>)
- *bis April 2017:* 4 Betaversionen mit sukzessiver Anpassung an 100% CCU2 Funktionalität / Kompatibilität inkl. HomeMatic-IP Unterstützung
- *13. April 2017:* Release Candidate (rc1)
- *01. Mai 2017:* Finale Version
- *Zukunft:* ausschließlich „stable“ Versionen (meist synchron mit CCU2 Firmware Releases)

RaspberryMatic: Features – I

- 100% HomeMatic CCU2 kompatibel
- Volle BidCos-RF (HomeMatic), Wired (HomeMatic-Wired) und HmIP-RF (HomeMatic-IP) Unterstützung
- Simple Installation via microSD-Karten Image (*.img)
- Basierend auf gleicher buildroot-Linux Umgebung wie CCU2 (jedoch aktuellere Version – 2017.02.1)
- Einsatz aktueller Linux-Kernel Versionen (v4.4.x)
- Nutzung RealTime Kernel (PREEMPT_RT) für potentiell geringere Latenzzeiten (garantierte Responsezeiten)
- Schreibgeschütztes root-Dateisystem für potentielle Lebenszeitverlängerung der SD Karte

RaspberryMatic: Features – II

- Unterstützung für onboard WLAN (RPi3) oder USB-basierten WLAN-Sticks
- Unterstützung für onboard Bluetooth (RPi3)
- Unterstützung Network UPS Tools (NUT) – Anschluss einer externen unterbrechungsfreien Stromversorgung via USB oder Netzwerk (z.B. USV an Synology NAS)
- Nutzbar als reiner LAN-Gateway – Betrieb mehrerer RaspberryMatic Installationen (z.B. 1xHaupt-RM + 2xRM als LAN-GW)
- Betrieb ohne Funkmodul – nur Anschluss eines LAN-Gateway (z.B. 1xHaupt-RM im Keller ohne Funkmodul + 2 x eQ3 LAN-GW)
- Voller IPv6 Support
- Standardmäßig aktivierter HTTPS Support (kein externes Zertifikat)

RaspberryMatic: Features – III

- Nutzer Partition (/usr/local) automatisch so groß wie max. SD-Kartengröße
- Unterstützung für verschiedene gängige RealTime-Clock (RTC) Module (PiFace Shim RTC, DS3231, DS1307)
- Unterstützung des RaspberryPi-Hardware WatchDog (automatischer Neustart bei hängender Hardware)
- Unterstützung für RaspberryPi-Hardware Random Number Generator
- Direkt integrierter CloudMatic / meine-homematic.de Support
- Für Entwickler: Erzeugung eines ARM-basierten Cross-Compilers um Drittsoftware speziell für RaspberryMatic zu kompilieren (z.B. um ein Addon zu erzeugen)
- Frei verfügbar und Open Source (Apache 2.0 Lizenz)

RaspberryMatic: Limitationen / Bugs

- Geräte-Updates für verschiedene Geräte (Rolladen, Dimmer) funktionieren zur Zeit nicht
(eQ3 arbeitet bereits an Funkmodul-Firmware Update)
- Möglicherweise inkorrekte DutyCycle Berechnung
(eQ3 analysiert das Problem)
- Keine Möglichkeit WLAN / Bluetooth via WebUI zu konfigurieren
- Keine Möglichkeit den LAN-Gateway Modus via WebUI zu konfigurieren

RaspberryMatic vs. CCU2

- Kosten: kein Unterschied
CCU2: 99,95 €
RaspberryPi3 Starter Kit + Funkmodul: 79,95 + 19,95 = 99,90 €
- Aktuellere buildroot Umgebung (2017.02.1)
- Neuere Linux-Kernel (v4.4.x) Version
- ~ 10 – 20x schnellerer WebUI Aufbau
- Unterstützung für WLAN + Bluetooth
- Potentiell höhere Sicherheit (Passwörter mit SHA256 Hashes, Standardmäßig aktiviertes HTTPS)
- **Aktuell – Neue ReGaHss/WebUI BugFix Version**
<https://github.com/eq-3/occu/tree/master/arm-gnueabi/f/packages-eQ-3/WebUI-Beta>

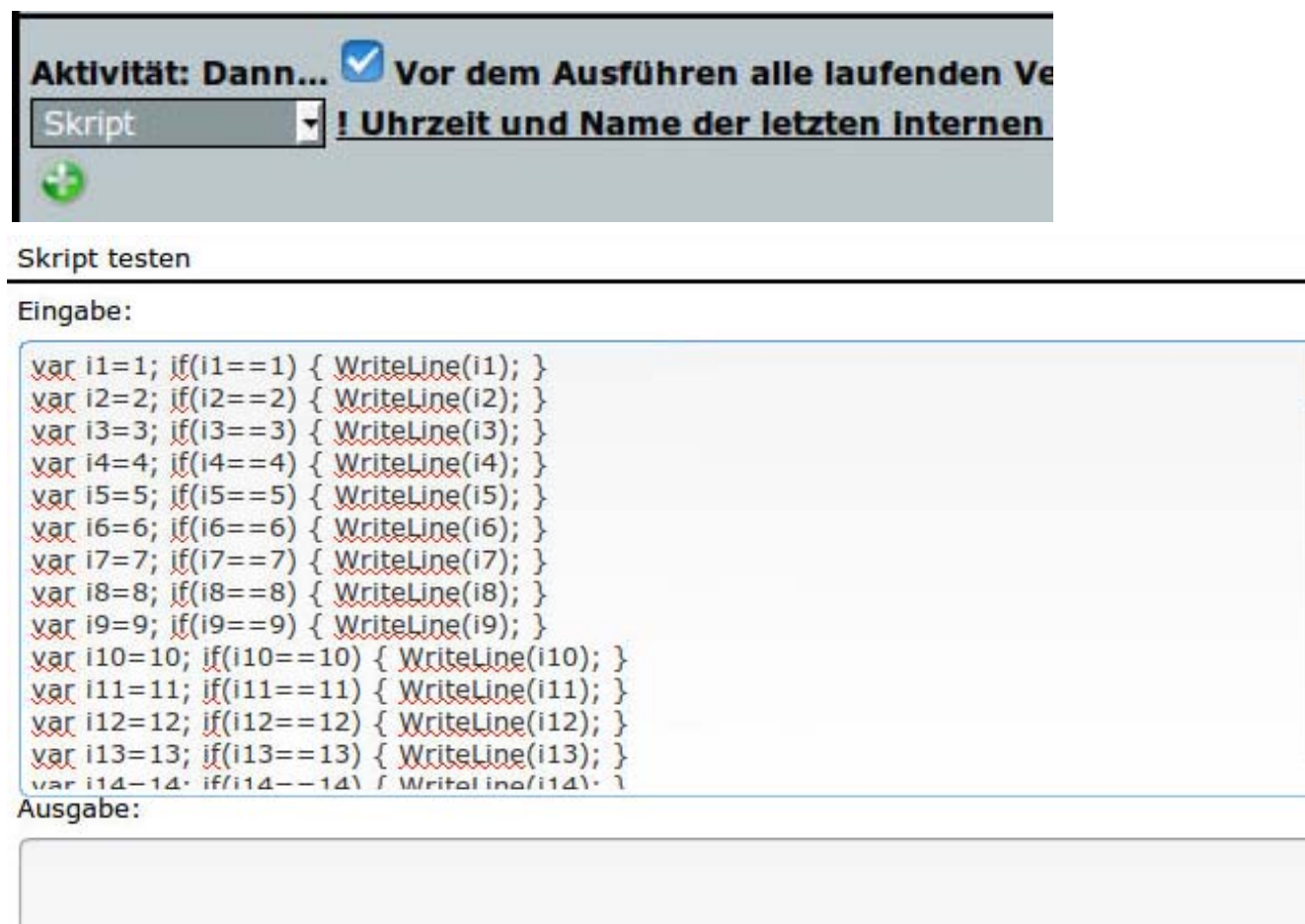
Bugfixes ReGaHss/WebUI – I

- Via Non-Disclosure-Agreement (NDA) Zugriff auf ReGaHss Quellcode erhalten
- Beseitigung teils langjährig limitierender Bugs / Probleme:
 - (1) Bugs in Timer-Modul (Zeitsteuerung)
(z.B. Timer < 15 Minuten stoppten abrupt)

The screenshot shows a web interface for configuring a timer. At the top, a grey box titled "Bedingung: Wenn..." contains a dropdown menu set to "Zeitsteuerung", a button labeled "Zeitmodul" with a red error icon, and a green plus icon next to a dropdown menu set to "UND". Below this is a section titled "Zeitmodul einstellen" which contains a warning message: "Wir empfehlen bei der Benutzung des Zeitmoduls, keine Zeitabstände kleiner 15 Min. zu verwenden. So ist sichergestellt, dass alle konfigurierten Programme zuverlässig ausgeführt werden können." At the bottom, under the heading "Zeit", there are two radio buttons: "Zeitspanne" (selected) and "Ganztägig". The "Zeitspanne" option has two time input fields: "Beginn: 11:29" and "Ende: 11:59", each with a small "v" button next to it.

Bugfixes ReGaHss/WebUI – II

- Beseitigung teils stark limitierender Bugs / Probleme:
 - (2) max. 200 Variablen-Problem
(in HM-Skripten konnten max. 200 verschiedene Skriptvariablennamen systemweit genutzt werden)



Bugfixes ReGaHss/WebUI – III

- Beseitigung teils stark limitierender Bugs / Probleme:
 - (3) Beseitigung max. 100kB HM-Skript Limitation
 - (4) Reparatur Zeitkonvertierung in HM-Skripten
(z.B. falsche Sommer/Winterzeit-Ausgabe in Format())
 - (5) Abfangen/Ignorieren von „PONG“ events
(falsche Fehlermeldungen im Logfile)
 - (6) Beseitigung verschiedener potentieller Absturzprobleme
durch inkorrekte Speicherallokierung/freigaben

Roadmap ReGaHss/WebAPI

- Veröffentlichung auch für CCU2 in kommenden FW-Updates geplant
- Potentielle Beseitigung weiterer Bugs/Probleme:
 - „11622“-like Fehlercode-Problem:
`ReGaHss: Error: IseESP::ExecError= Execution failed: [-1] 0 0x00 [0] 144 0x90 [1] 0 0x00 [2] 97 0x61 [3] 0 0x00 [4] 49 0x31 [../Platform/DOM/iseESPexec.cpp (11622)]`
 - weiteres Entfernen von statischen Limitation (max. Objektanzahl, etc.)
 - weitere HM-Skript Probleme (z.B. kein ,^‘ erlaubt, kein leeres Kommentar via ,!‘, etc.)
 - WebUI Probleme (z.B. kein ,\‘ oder ,\n‘ in WebUI erlaubt)
- ggf. Möglichkeit Geräte ,inaktiv‘ zu schalten damit keine Servicemeldungen für diese inaktiven Geräte mehr eingehen

➔ **Weitere Fehler/Wünsche? Bitte! ➔ Umfrage im HomeMatic-Forum geplant**

Roadmap RaspberryMatic

- RaspberryPi Zero / ZeroW Unterstützung
- Linux Kernel v4.9+ Unterstützung
- Update/Upgrade Mechanismus via WebUI
 - ➔ Momentan muss Update/Upgrade via Backup → Neuinstallation → Restore erfolgen
- WebUI-Konfiguration für WLAN, Bluetooth und LAN Gateway-Modus
- Support für RPi USV HATs (z.B. S.USV, PicoUPS)
- Unterstützung für andere Embedded Boards mit RaspberryPi kompatiblen GPIO (z.B. BananaPi M3, OrangePi Plus 2e, etc.)



Ideen Weiterentwicklung

- RaspberryMatic wird HomeMatic-IP Gateway?
- Erhöhte Sicherheit: Prozesse (rfd, hs485d, ReGaHss, etc.) nicht als „root“ ausführen lassen sondern als Nutzer; Sicherheitskonzept für XML-RPC/BIN-RPC?
- CCU2 Laborfirmwares basierend auf RaspberryMatic/OCCCU?
- ioBroker CCU-Addon sobald RaspberryPi \geq 2GB RAM (<https://github.com/jens-maus/hm-iobroker>)
- YAHM könnte RaspberryMatic als zukünftige Basis nutzen!?!

Rückblick Usertreffen 2016 / Danke eQ3 !

„IMHO“ / Wünsche an eQ3

- ✓ regelmäßige Firmware Updates synchron mit CCU2 (unified build system?)
- ✓ Update auf aktuellste Buildroot-Umgebung (2016.02) ggf. sogar mit RealTime Support (PREEMPT_RT)
- ✓ In Zukunft als Referenzplattform nutzen / propagieren
- ✓ Veröffentlichung komplettes Buildsystem (z.B. eigene Generierung von RaspberryMatic Images bzw. sogar mitunter komplette CCU2 Firmware Image)
- ✓ Veröffentlichung weiterer Quellcodes (z.B. ReGaHss) – Lizenzproblematik müsste geklärt werden (Crowdfunding?)
- ✓ Bekannte Bugs/Limitationen in ReGaHss beseitigen (ggf. auf die OpenSource Community zurückgreifen)

„IMHO“ / Wünsche an eQ3 für 2017

- Weiter regelmäßig zur CCU2-Firmware synchronisierte OCCU Updates
- Mehr direkte Entwicklungsarbeit unter GitHub (nicht nur Snapshots, Nutzung Bug/Issue-Tracker + Pull Requests)
- Noch mehr „OpenSource“ Engagement – Freigabe potentiell unkritischer Softwarekomponenten (z.B. eq3configd, ssdpd, multimacd, rfd, hs485d, etc.)
- Weiterführen der CCU Platform („NoCloud“ liegt mehr und mehr – gerade in der Hausautomation– im Trend)
- Offenlegung der HomeMatic-IP Cloud API um z.B. eigenen Gateway innerhalb RaspberryMatic zu ermöglichen
- mehr Aktoren / Sensoren!

Was kann/sollte die Community beitragen?

- Weiterhin viel eigene (Software)entwicklungen rund um das Thema HomeMatic
- Mehr Anleitungen/Tutorials/HowTos auf homematic-forum.de um Anfängern den Einstieg zu erleichtern
- Mehr OpenSource Engagement (z.B. via GitHub) und Softwareentwicklung die allen zugute kommt
- Direkte Beteiligung an der Weiterentwicklung von OCCU-SDK um eQ3 mehr Motivation für weitere Offenlegungen zu geben
- Weniger eQ3-Bashing im Forum (es wird sehr viel getan!)

Danke für die Aufmerksamkeit

Fragen?

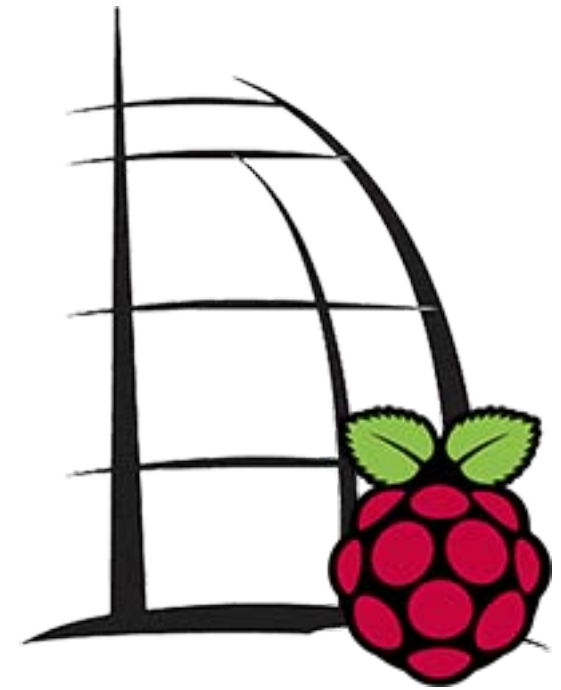
<https://github.com/jens-maus/RaspberryMatic>

Sach-/Geldspenden (PayPal) sehr hilfreich!



Dresden

RaspberryMatic (2)



DEMO

Grundinstallation

CCU2 → RaspberryMatic

Update/Upgrade

Zusatzfunktionen (LAN Gateway, NUT Server)

Jens Maus – mail@jens-maus.de

HomeMatic User-Treffen 2017, Kassel, 06. Mai 2017

Grundinstallation

1. Download *.zip-Datei
(<https://github.com/jens-maus/RaspberryMatic/releases/latest>)
2. Auspacken der zip Datei + Sicherstellen *.img Konsistent
(sha256 Prüfung)
3. „Flashen“ des *.img auf microSD-Karte
4. Installation microSD Karte + initialer Start RaspberryPi
5. Initialer Login WebUI via <http://homematic-raspi/>

CCU2 → RaspberryMatic

1. Erstellen Konfig-Backup CCU2 (*.sbk)
(Einstellungen → Systemsteuerung → Sicherheit → Backup erstellen)
2. Entfernen aller CCU2 Addons (ausser CUxD)
(Einstellungen → Zusatzsoftware)
3. Erneutes erstellen Konfig-Backup CCU2 (*.sbk)
(Einstellungen → Systemsteuerung → Sicherheit → Backup erstellen)
4. Ausschalten CCU2
5. Einloggen via WebUI in RaspberryMatic
6. Wiederherstellen CCU2-Konfigbackup (ohne Addons)
(Einstellungen → Systemsteuerung → Sicherheit → Backup einspielen)

Update/Upgrade RaspberryMatic

1. Erstellen Konfig-Backup (*.sbk)
(Einstellungen → Systemsteuerung → Sicherheit → Backup erstellen)
2. Ausschalten RaspberryMatic
(Einstellungen → Systemsteuerung → Zentralen-Wartung → Shutdown)
3. microSD Karte mit neuer Version beschreiben
(am besten 2. microSD Karte verwenden)
4. Einschalten RaspberryPi
5. Einloggen via WebUI in RaspberryMatic
6. Wiederherstellen Konfigbackup
(Einstellungen → Systemsteuerung → Sicherheit → Backup einspielen)

Zusatzfunktionen RaspberryMatic

- RaspberryMatic als reinen LAN-Gateway betreiben
(Sehr interessant wenn PiZero Unterstützung umgesetzt)
- RaspberryMatic ohne Funkmodul betreiben (nur mit LAN GW)
(Interessant für Betrieb als Zentrale in Keller)
- Einrichtung WLAN Betrieb mit RaspberryMatic
- Einrichtung einer USV via USB bzw. als NUT Client

Danke für die Aufmerksamkeit

Fragen?

<https://github.com/jens-maus/RaspberryMatic>

Sach-/Geldspenden (PayPal) sehr hilfreich!



Dresden