

# Dokumentation zum ioBroker.ems-esp Adapter – August 2023

ioBroker.ems-esp Adapter - Einleitung.....	2
API-Calls .....	2
Unterschiede Original Bosch Gateways (km200) vs. Ems-Esp Gateway.....	3
Adapter Ablauf-Logik .....	3
Einstellungen EMS-ESP: .....	4
Einstellungen KM200 .....	5
Energieverbrauch-Statistiken für EMS-ESP .....	6
Energieverbrauch-Statistiken für KM200 (Recordings) .....	7
Statistiken .....	8
Brennwert-Nutzen – Brennereffizienz .....	8
Veränderungen in der States-Struktur.....	9
Wärmebedarfs-Steuerung im ems-esp Adapter .....	10
Funktionen des eingebauten Syslog Servers.....	12

## ioBroker.ems-esp Adapter - Einleitung

Der Adapter unterstützt eine Schnittstelle zu den Heizsystemen der Bosch-Gruppe mit EMS- oder EMS+-Bus. (Buderus/Junkers/Netfit usw.) Der Adapter kann sich dabei sowohl

- mit den originären LAN-Schnittstellen der Heizungen der Boschgruppe (IP-inside, km200, km100, km50, MB-LAN2 etc),  
Die neueren Bosch interfaces (z.B. MX200 etc) unterstützen keinen lokalen LAN-Zugriff mehr und werden vom Adapter nicht unterstützt.
- als auch mit dem EMS-Bus Gateway mit ESP32 Chip verbinden. (<https://github.com/emsesp/EMS-ESP32>).  
Das EMS Bus Gateway kann von BBQKees bestellt werden:  
[BBQKees Electronics – EMS bus to Home Automation interfaces \(bbqkees-electronics.nl\)](https://bbqkees-electronics.nl)

Das ems-esp Gateway ist eine kleine Box, das an den Serviceanschluss oder direkt am EMS-Bus der Heizanlage / Wärmepumpe angeschlossen wird und die dann über WLAN/LAN und MQTT/WEB-API die Verbindung zwischen dem Heizsystem und dem Hausautomatisierungssystem herstellt.

Dieses EMS-BUS Gateway ist in der Lage eine Vielzahl von älteren Heizungsanlagen ohne Internet-Anschluss mit Homeassistant zu verbinden. Zusammen mit den Softwareentwicklern der EMS-ESP Firmware wurde die WEB-API so angepasst, dass dieser ems-esp ioBroker Adapter nahtlos integriert werden kann.

## API-Calls

Der Adapter kann Daten auf beiden Gateways per WEB-API lesen und schreiben, um alle Heizungskomponenten zu steuern. Es kann entweder für die Original-Gateways der Bosch-Gruppe oder das ems-esp Gateway oder Beides parallel verwendet werden.

Die WEB-API Kommunikation zum km200 Gateway erfolgt verschlüsselt, die zum EMS-ESP Gateway nicht. Schreiboperationen werden beim ems-esp Gateway durch einen generierten Access Token sichergestellt.

ioBroker Adapter				
regular polling	< ----- web API ----- > LAN en- / decrypted	<b>km200 Gateway</b>	< ----- >	<b>E</b>
regular polling	< ----- web API /V3 ----- > LAN / WLAN	<b>ems-esp Gateway</b>	< ----- >	<b>M      BUS</b>
Energy polling & Calculation additional functions	< ----- Adapter Logic ----- >	<b>ems-esp/km200</b>	< ----- >	<b>S</b>

## Unterschiede Original Bosch Gateways (km200) vs. Ems-Esp Gateway

Die original Bosch Gateways stellen die Internet-Bedienbarkeit der Heizungsanlage über die entsprechenden App's der Boschgruppe sicher. Der Funktionsumfang ist dabei sehr eingeschränkt. Die Kommunikation erfolgt dabei über die Bosch-Cloud. Im Wesentlichen können die Schaltzeiten, Urlaubszeiten und die Temperaturvorgaben eingestellt werden. Eine wenige Anlagendaten sind lesbar.

Eine dokumentierte API existiert bei Bosch nicht. Die Kommunikation mit den Original-Gateways erfolgt verschlüsselt – auch im lokalen LAN (s.u.). Das km200 Gateway unterstützt dabei im LAN keine Regelungs- bzw. Anlagenparameter. Die API-Zugriffe müssen separat für jedes Datenfeld erfolgen. Damit ist der Lesevorgang (Polling) langsam und der Polling-Zyklus wird vom Adapter auf minimal 90 Sekunden begrenzt.

Das Ems-ESP Gateway ist eine separate Hardware, welche direkt auf den Telegramverkehr im EMS-Bus zugreift. Da diese Telegramme nicht dokumentiert sind, ist es sehr viel Detektivarbeit gewesen diese für die unterschiedlichen Heizsysteme zu entschlüsseln und zu implementieren. Viele Heizungsparameter und Einstellungsmöglichkeiten werden unterstützt, aber andere Benutzereinstellungen sind (noch) nicht vorhanden. Es wird eine Vielzahl unterschiedlicher (auch älterer) Heizungssysteme ohne Internetgateway und Marken unterstützt.

Das Auslesen der Daten erfolgt sehr schnell, damit sind auch kurze Polling-Zyklen von 15 Sekunden möglich.

## Adapter Ablauf-Logik

Im Adapter ist auswählbar, ob ein KM200 und/oder ein EMS-ESP Gateway ausgelesen werden soll. Sollte beides vorhanden sein, empfiehlt es sich die EMS-ESP Datenstruktur in der KM200-Logik darzustellen.

Darüber hinaus sind über den Reiter „Parameter“ ausgewählt werden, ob Statistiken erstellt werden sollen, bzw. der Kesselwirkungsgrad in Abhängigkeit der Temperaturen ermittelt werden soll. (Brennwerteffekt). Darüber hinaus gibt es eine Heizungsbedarfs-Steuerung (s.u.)

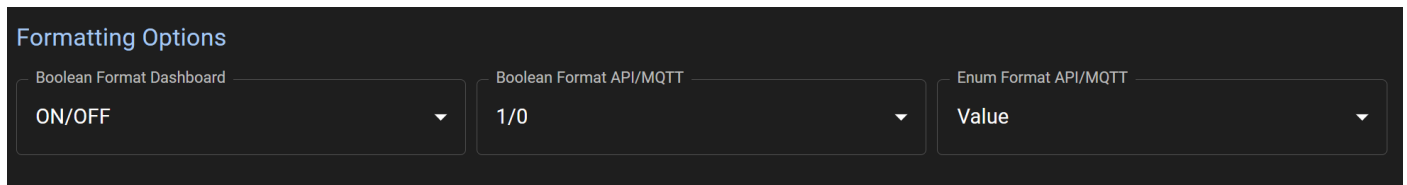
Bei Start des Adapters können alle bisherigen States gelöscht werden (siehe Parameter). Das empfiehlt sich nur, wenn Datenstrukturen verändert werden.

Nach dem Start werden einmalig alle Details zu den States per API gelesen und die entsprechenden ioBroker Objekte erzeugt. Danach erfolgt im regelmäßigen Polling-Zyklus das Lesen der aktuellen Werte und ein Update der States.

Alle anderen Verarbeitungen erfolgen davon unabhängig in jeweils eigenen Verarbeitungs-Zyklen.

## Einstellungen EMS-ESP:

Im EMS-ESP Gateway müssen unter Settings die „Formatting Options“ für das API-Interface eingestellt werden. Formatierungsoptionen für das boolesche Format muss 1/0 und für das Enum-Format Index oder Value sein.



Formatting Options

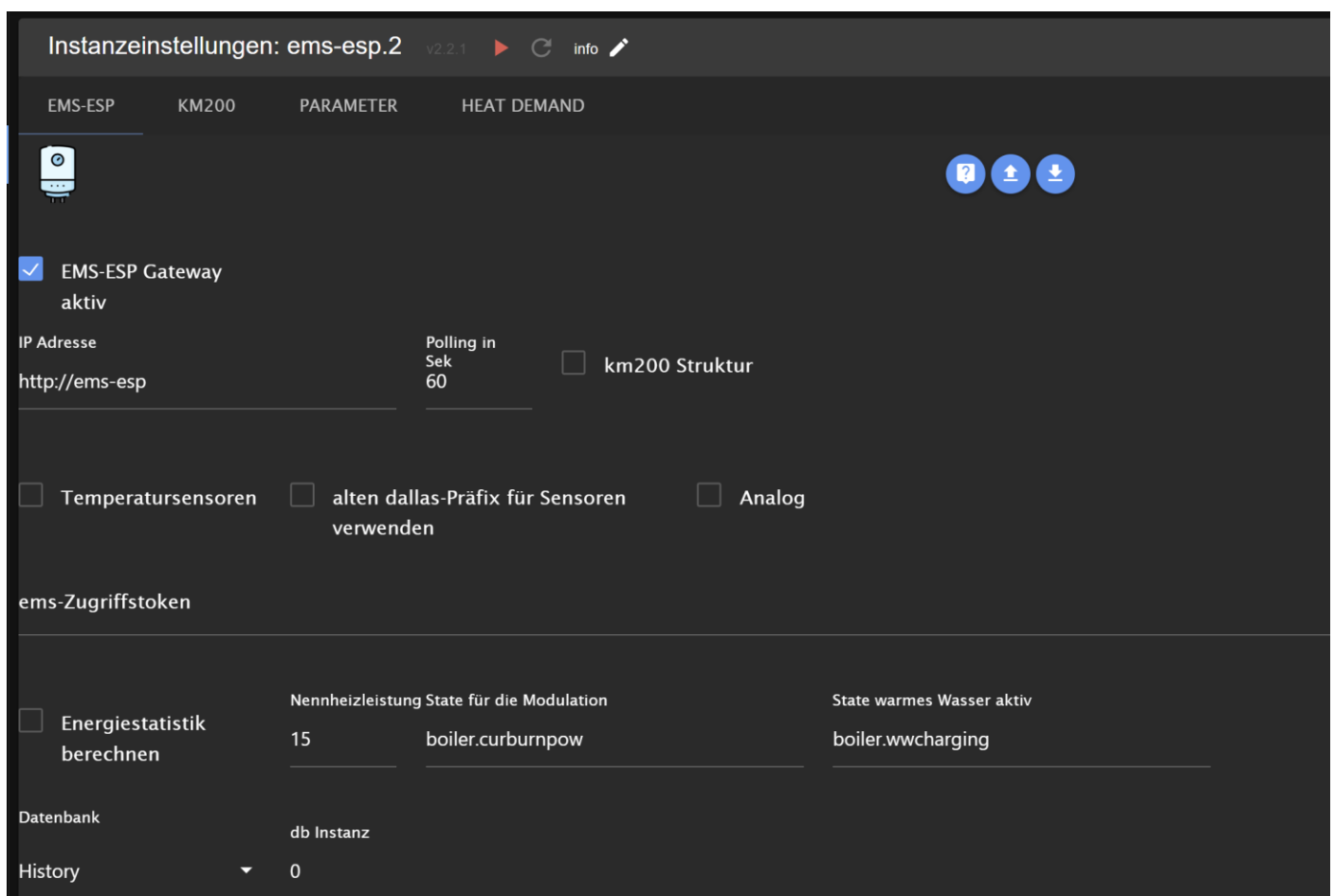
Boolean Format Dashboard: ON/OFF

Boolean Format API/MQTT: 1/0

Enum Format API/MQTT: Value

Damit der ioBroker-Adapter auch Werte per API schreiben kann, muss dies in den Settings erlaubt werden. Ich empfehle Anfängern erst einmal „**Bypass Access Token authorization on API calls**“ zu setzen und später zum Zugriffsschutz das Token zu generieren und in der Adapterkonfiguration einzugeben.

ioBroker Instanzenereinstellungen:



Instanzeinstellungen: ems-esp.2 v2.2.1

EMS-ESP KM200 PARAMETER HEAT DEMAND

☒ EMS-ESP Gateway aktiv

IP Adresse: http://ems-esp

Polling in Sek: 60

☐ km200 Struktur

☐ Temperatursensoren ☐ alten dallas-Präfix für Sensoren verwenden ☐ Analog

ems-Zugriffstoken

☐ Energiestatistik berechnen

Nennheizleistung State für die Modulation: 15

boiler.curburnpow

State warmes Wasser aktiv: boiler.wwcharging

Datenbank: db Instanz

History: 0

EMS-ESP Einstellungen:

Mit dem km-Struktur Kontrollkästchen wird entweder die km200-ähnliche Gerätestruktur für ems-ESP-Datenfelder verwendet oder die ursprüngliche EMS-ESP-Geräteansicht beibehalten: Boiler, Thermostat, Mischer usw.

Ansonsten sind die IP-Adresse des Gateways, die Polling-Zeit und der Zugriffstoken einzugeben. Es kann darüber hinaus das Lesen angeschlossener Dallas- und Analogsensoren aktiviert werden.

NEU: Es kann nun das Erstellen von Energieverbrauchs-Statistiken aktiviert werden (Sie Kapitel Energie).

## Einstellungen KM200

Die Web-API-Aufrufe zum/vom km200-Gateway sind verschlüsselt. Für die Ver-/Entschlüsselung werden zwei Passwörter benötigt:

- Das Gateway-Passwort auf einem Label auf dem Gateway in der Form: xxxx-xxxx-xxxx-xxxx (Groß-/Kleinschreibung beachten und die Bindezeichen mit eingeben)
- Das private Passwort ist das, welches mit der Buderus **MyDevice** App festgelegt wurde!
- **(nicht myBuderus oder ähnliche Cloud-Apps verwenden!)**

ioBroker Instanzeinstellungen:

The screenshot shows the 'Instanzeinstellungen: ems-esp.2' window with the 'KM200' tab selected. The interface includes a header bar with 'EMS-ESP', 'KM200', 'PARAMETER', and 'HEAT DEMAND' tabs. Below the header, there is a section for 'KM200 Gateway' with a checkbox labeled 'aktiv'. The 'IP Adresse' field contains 'http://192.168.178.x'. The 'Polling in Sek' field is set to '300'. The 'csv-Datei für km200' field has a note: 'leer: keine km200-Daten \*: alle km200-Felder lesen'. Below this, there are two password fields: 'Gateway-Passwort' and 'privates Passwort'. At the bottom, there is a section for 'Energieaufzeichnungen (nur km200)' with a checkbox. The 'Recordings State-Format ohne Datenbank' dropdown is set to 'Array von Werten'. The 'Datenbank' dropdown is set to 'no database'. The 'db Instanz' field is set to '0'. The 'Multiplikator' field is set to '1'. The 'Samples-Interpolation' checkbox is checked.

Es sind die IP-Adresse des Gateways, die Polling-Zeit und die beiden Passwörter einzugeben.

Beim 1. Adapterstart wird empfohlen alle km200 Datenfelder mit einem "\*" auszuwählen. Der Adapter erstellt dann eine km200.csv-Datei im Verzeichnis ../iobroker-data/ems-esp/{instance}.

Diese Datei kann beim nächsten Start der Adapter-Instanz verwendet werden. Nicht benötigte Zeilen (Felder) können gelöscht werden, um die Anzahl der auszulesenden km200-Felder zu reduzieren. (Kopie erstellen und Datei umbenennen).

Die km200 Web-API erfordert das Abfragen jedes einzelnen Wertes mit einem eigenen http-get Befehl. Das können schon bei Anlagen mit 2 Heizkreisen ca. 150 einzelne Abfragen sein. Entsprechend lange dauert dann ein einzelner Abfragezyklus (15-40 Sekunden).

- KM200 Polling ist ebenfalls ein Parameter (Standard 300 Sekunden) und der minimal einstellbare Wert beträgt 90 Sekunden. (s.o. Dauer des Abfragezyklus)
- km200-recordings (Energieverbrauchs- und Temperaturstatistiken) werden stündlich aktualisiert

## Energieverbrauch-Statistiken für EMS-ESP

Das EMS-ESP Gateway berechnet die Verbrauchswerte Werte nicht in der Firmware. Neu ist diese Berechnung im ioBroker Adapter realisiert. Vergangenheitswerte sind nicht lesbar. Für das EMS-ESP Gateway werden Werte alle 15 Sekunden ermittelt und fortgeschrieben.

Die Energiestatistiken können in der Instanz aktiviert werden und setzen zwingend eine aktive Datenbank-Instanz voraus. (History, mySQL, Influxdb). Für InfluxDB V1 muss die Aufbewahrungsrichtlinie auf mindestens 170 Wochen eingestellt werden. (Aufbewahrungsrichtlinie global ändern für ioBroker-Dauer 170w;)

EMS-ESP:

☒ Energiestatistik berechnen

Nennheizleistung

22

State für die Modulation

boiler.curburnpow

State warmes Wasser aktiv

boiler.wwcharging

State-Format ohne Datenbank

Array von Werten

Datenbank

▼  
mySQL

db Instanz

▼  
0

Es ist dabei die Nennheizleistung des Kessels einzugeben und die Statenamen für die aktuelle Modulation und für WW-Aktiv. Auf dieser Basis werden dann die aktuelle Brenner-, Heizungs- (CH) und WW-Leistung (DHW) ermittelt.

In der Objektstruktur werden dann unter energy drei Unterstrukturen angelegt: actualCHPower (Heizung), actualDHWPower (Warmwasser) und actualPower (Gesamt).

Die power-States werden alle 15 Sekunden aktualisiert und dann daraus alle 10 Minuten die Energieverbrauchswerte ermittelt. Die Verbrauchswerte können dann graphisch mit z.B. Flot dargestellt werden.

- stündlichen Werte unter \_Hours
- die Täglichen Werte unter \_Days
- die monatlichen Werte unter \_Months

mit Datenbank Direktzugriff gespeichert / aktualisiert. (Deswegen null). Die JSON-Array Werte werden je nach Auswahl des Formates fortgeschrieben (aktuelle Werte als Erstes):

- stündlichen Werte unter Hours
- die Täglichen Werte unter Days
- die monatlichen Werte unter Months

## Energieverbrauch-Statistiken für KM200 (Recordings)

Die meisten modernen Heizsysteme verfügen über ein IP-Inside- oder KMxxx Gateway und unterstützen Energie- und Temperaturstatistiken. Diese werden im Gateway berechnet. In der Regel sind damit Verbrauchs- und Temperaturwerte der letzten 12 Monate abrufbar. Die Bosch Gateways berechnen die Verbrauchswerte durch „Samples“ alle 60 Sekunden. Es sind stündliche, tägliche und monatliche Werte auslesbar. Der Adapter liest diese Werte stündlich aus.

KM200:

<input checked="" type="checkbox"/> Energieaufzeichnungen (nur km200)	Recordings State-Format ohne Datenbank Array von Werten	Datenbank History	db Instanz 0	Multiplikator 1	<input checked="" type="checkbox"/> Samples-Interpolation
--	--	----------------------	-----------------	--------------------	---

Die Checkbox „Energieaufzeichnungen“ muss aktiviert und die Datenbankinstanz (History, MySQL oder InfluxDB) definiert sein. Der History, SQL oder InfluxDB Adapter muss installiert und aktiv sein, um diese Option zu verwenden. Für InfluxDB V1 muss die Aufbewahrungsrichtlinie auf mindestens 170 Wochen eingestellt werden. (Aufbewahrungsrichtlinie global ändern für ioBroker-Dauer 170w;)

Es kann ein Multiplikator eingegeben werden, falls Werte nicht in kWh zur Verfügung stehen. Es gibt Monate in denen Samples fehlen. Die Cloud-Apps rechnen dann die Werte hoch (Wenn z.B. 10% der Werte fehlen, dann wird der Wert um 10% erhöht). Das kann im Adapter aktiviert werden, damit die Werte mit der App übereinstimmen.










recordings									
dhwCircuits									
dhw1									
actualTemp	km200:recordings.dhwCircu...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
heatSources									
actualCHPower	km200:recordings.heatSour...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
actualDHWPower	km200:recordings.heatSour...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
actualPower	km200:recordings.heatSour...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
hs1									
actualPower	km200:recordings.heatSour...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
km200				Heizungsraum	Heizung				
Days	recordings days	<input type="radio"/> state		Heizungsraum	Heizung	[4.5,11,3.8,7.8,2.4....	664		
Hours	recordings hours	<input type="radio"/> state		Heizungsraum	Heizung	[0,0,0,0,0,0,2,2,2,...	664		
Months	recordings months	<input type="radio"/> state		Heizungsraum	Heizung	[0,0,0,0,0,0,0,0,0,...	664		
_Days	db daily recordings	<input type="radio"/> state		Heizungsraum	Heizung	(null) kWh	664		
_Hours	db hourly recordings	<input type="radio"/> state		Heizungsraum	Heizung	(null) kWh	664		
_Months	db monthly recordings	<input type="radio"/> state		Heizungsraum	Heizung	(null) kWh	664		
last12m	kWh total for last 12 months	<input type="radio"/> state	value	Heizungsraum	Heizung	17786 kWh	664		
heatingCircuits									
hc1									
roomtemperature	km200:recordings.heatingC...	<input type="radio"/> state	value	Heizungsraum	Heizung		664		
km200				Heizungsraum	Heizung				
Days	recordings days	<input type="radio"/> state		Heizungsraum	Heizung	[23.9,24.1,24.5,23....	664		

Die Objektstruktur wird von der Bosch API vorgegeben. Es gibt Verbrauchswerte und Temperaturen. Wie beim EMS-ESP unterteilt nach Stunden (Hours), Tagen (Days) und Monaten (Months) als Arrays von Werten und in den States \_Hours, \_Days und \_Months direkt als Datenbankeinträge zur graphischen Darstellung.

**WICHTIG:** Bei Datenbanken werden die Daten direkt mit SQL-Befehlen in die „\_“-States geschrieben. Unter Objekte wird dann der Wert (null) angezeigt. (Das ist richtig !!!). Siehe vorheriges Kapitel.

## Statistiken

Brennerstatistiken können aktiviert werden und zeigen:

statistics			
 boiler-on-1h	percentage boiler on per hour	<input type="checkbox"/> statevalue	0 %
 boiler-starts-1h	boiler starts per hour	<input type="checkbox"/> statevalue	0
 boiler-starts-24h	boiler starts per 24 hours	<input type="checkbox"/> statevalue	1
 created	Database (mySQL/InfluxDB) enabled for fiel...	<input type="checkbox"/> statevalue	true
 efficiency	boiler efficiency	<input type="checkbox"/> statevalue	0 %
 ems-read	ems read time for polling	<input type="checkbox"/> statevalue	1,181 seconds
 km200-read	km200 read time for polling	<input type="checkbox"/> statevalue	33,225 seconds
 ww-starts-1h	ww starts per hour (EMS-ESP only)	<input type="checkbox"/> statevalue	0
 ww-starts-24h	ww starts per 24 hours (EMS-ESP only)	<input type="checkbox"/> statevalue	1

- boiler-on-1h: Wieviel Prozent (0-100%) war der Kessel aktiv während der letzten Stunde
- boiler-starts-1h und boiler-starts-24h: Anzahl der Kesselstarts im Zeitraum (1 / 24 Stunden)
- created: Indikator, dass die Statistik-Struktur angelegt wurde
- efficiency: aktuelle Kessel-Effizienz wenn aktiviert (Brennwertnutzen bei Gas- und Ölkessel)
- ems-read: Die Abfragezyklus-Verarbeitungszeit für EMS-ESP -Gateway-Lesevorgänge
- km200-read: ... analog für KM200
- ww-starts-1h und ww-starts-24h (nur bei aktiven EMS-ESP Gateway) – Kesselstarts zur WW-Bereitung

Zur Berechnung der Statistiken wird eine aktive Datenbankinstanz (siehe oben) benötigt.

## Brennwert-Nutzen – Brennereffizienz

Der Kesselwirkungsgrad kann berechnet werden, wenn die Parameter ausgefüllt sind.  
(nur Gas- und Ölkessel)

Die Effizienz (Brennwertnutzen) wird basierend auf der durchschnittlichen Kesseltemperatur berechnet:  
 $(\text{Kesseltemperatur} + \text{Rücklauftemperatur}) / 2$ .

Sehen Sie im Datenblatt Ihres Heizkessels nach, um die Effizienztabelle entsprechend anzupassen.

Die States zur Modulation, Vor- und Rücklauftemperatur sind einzugeben.

<input checked="" type="checkbox"/> Kesselwirkungsgrad berechnen (Gas und Öl)									
State Modulation			State Vorlauftemp			State Rücklauftemp			
< 20°C	< 25°C	< 30°C	< 35°C	< 40°C	< 45°C	< 50°C	< 55°C	< 60°C	< 70°C
109,5	109,3	108,3	108	106,5	105,2	103	100	98	97



## Veränderungen in der States-Struktur

Wann immer eine neue EMS-ESP-Firmware neue Datenfelder hinzufügt und/oder Datenfeldnamen ändert, werden sie während des Adapterlaufs verarbeitet.

Trotzdem werden veraltete Datenfelder nicht automatisch vom Adapter gelöscht. Es besteht die Möglichkeit, die Zustandsstruktur neu aufzubauen, indem Zustände beim Neustart des Adapters gelöscht werden (Zustände mit Historie / DB-Einträgen bleiben erhalten und müssen ggfs. manuell gelöscht werden).

Nach einem Ems-Esp Firmwareupdate wird zu einem Neustart des Adapters mit Löschen der Felder geraten. Dieser kann aber auch später nachgeholt werden.

## Wärmebedarfs-Steuerung im ems-esp Adapter

In der aktuellen Version des ems-esp Adapters ist eine Steuerung der Heizung in Abhängigkeit eines berechneten Wärmebedarfs realisiert.

Es gibt eine separate Konfigurationsseite „Wärmebedarf“ in der 2 Eingabelisten existieren:  
(Neue Einträge mit dem +-Symbol)

Im ersten Block werden für jeden Raum (Name frei wählbar) die folgenden Einträge definiert:

- *Settemp*: State für die Solltemperatur des Heizkörpers / Raumes
- *Actual temp*: State für die Ist-Temperatur des Raumes
- *Minimum delta* – Differenz zwischen settemp – actualtemp ab der Heizbedarf besteht:  
Beispiel: Soll 21° - Ist 20° → delta 1°. Wenn delta >= minimum delta → dann Heizbedarf.  
Minimum Delta = 0 bedeutet Heizbedarf, wenn die aktuelle Temperatur gleich oder kleiner der Solltemperatur ist.
- *Hc* : Zuordnung zum Heizkreis (hc1 ... hc4)
- *Weight*: Gewichtung des Heizkörpers / Raumes (Welche Heizleistung hat der Radiator bzw. die Fußbodenheizung?).

Im zweiten Block werden für jeden Heizkreis festgelegt:

- *Weighton*: **HK an** bei Summe der Gewichtungs-Werte des Heizkreise >= weighton
- *Weightoff*: **HK aus** bei Summe der Gewichtungs-Werte des Heizkreise <= weightoff
- *State*: zu schaltender State
- *On*: Wert des States für HK an.
- *Off*: Wert des States für HK aus.
- *Savetemp*: Wenn eingeschaltet, dann wird der aktuelle Sollwert gespeichert und dieser Wert bei ausgeschaltetem Heizkreis als Referenz genommen. Das ist notwendig, da bei ausgeschaltetem Fußboden-Heizkreis der Sollwert auf 0 gesetzt wird.

Der Schalter *heatdemand* schaltet die automatische Wärmebedarfssteuerung bei Adapterstart ein oder aus.

Hier eine Beispiel-Konfiguration mit Homematic Thermostaten (hc1) und km200 Mischer-gesteuertem Fußboden-Heizkreis (hc2).

Instanzeinstellungen: ems-esp.0

HAUPT-EINSTELLUNGEN WÄRMEBEDARF

☐ heatdemand

room	set temp	actual temp	minimum delta	hc	weight	
Wintergarten	hm-rpc.0.MEQ0479199.2.SET_TEMPERATURE	hm-rpc.0.MEQ0479199.2.ACTUAL_TEMPERATURE	1.5	hc1	2	<div><div></div><div></div><div></div></div>
Wohnzimmer	hm-rpc.0.MEQ0478977.2.SET_TEMPERATURE	hm-rpc.0.MEQ0478977.2.ACTUAL_TEMPERATURE	1.0	hc1	3	<div><div></div><div></div><div></div></div>
Badezimmer	hm-rpc.0.MEQ0447040.4.SET_TEMPERATURE	hm-rpc.0.MEQ0447040.4.ACTUAL_TEMPERATURE	2.0	hc1	2	<div><div></div><div></div><div></div></div>
Arbeitszimmer	hm-rpc.0.MEQ0450531.4.SET_TEMPERATURE	hm-rpc.0.MEQ0450531.4.ACTUAL_TEMPERATURE	1.0	hc1	2	<div><div></div><div></div><div></div></div>
Wohnzimmer FB	ems-esp.0.heatingCircuits.hc2.currentRoomSetpoint	ems-esp.0.heatingCircuits.hc2.roomtemperature	0.5	hc2	5	<div><div></div><div></div><div></div></div>

hc	weighton	weightoff	state	on	off	savetemp	
hc1	3	2	ems-esp.0.heatingCircuits.hc1.temporaryRoomSetpoint	-1	0	<input type="checkbox"/>	<div><div></div><div></div><div></div></div>
hc2	5	0	ems-esp.0.heatingCircuits.hc2.temporaryRoomSetpoint	-1	0	<input checked="" type="checkbox"/>	<div><div></div><div></div><div></div></div>

SPERICHERN

SPERICHERN UND SCHLIESSEN

ABBRECHEN

In der Objekt-Struktur des ems-Adapters werden dann nach dem die Instanz gestartet wurde folgende Objekt-States unter **controls** angelegt:

ID	Name	Typ	Rolle	Raum	Funktion	Wert	Einstellun...
ems-esp							---
0							---
controls							---
hc1							---
Arbeitszimmer							---
actualtemp	actual temperature	state	value			22.9	664
deltam	minimum room delta temperature for swit...	state	value			1	664
settemp	set temperature	state	value			20	664
weight	room weight for switching off	state	value			2	664
Badezimmer							---
Wintergarten							---
Wohnzimmer							---
off	state value off	state	value			0	664
on	state value on	state	value			-1	664
state	state for heating control	state	value			ems-esp.0.heatin...	664
status	hc control status	state	value			(null)	664
weight	hc weight actual	state	value			0	664
weightoff	hc weight for switching off	state	value			2	664
weighton	hc weight for switching on	state	value			3	664
hc2							---
Wohnzimmer FB							---
off	state value off	state	value			0	664
on	state value on	state	value			-1	664
savesettemp	saved settemp when switching off	state	value			-1	664
state	state for heating control	state	value			ems-esp.0.heatin...	664
status	hc control status	state	value			(null)	664
weight	hc weight actual	state	value			0	664
weightoff	hc weight for switching off	state	value			0	664
weighton	hc weight for switching on	state	value			5	664
active	hc control active	state	value			false	664

Der letzte State *active* ist bei Adapterstart mit dem Wert von *heatdemand* vorbelegt und steuert, ob die wärmebedarfsabhängige Regelung aktiv ist (true) oder inaktiv ist (false). Der Wert kann dann z.B. über VIS gesetzt werden. Bei Adapterstart ist z.B. ohne gesetztem Wert *heatdemand* die Regelung erst einmal inaktiv und kann später in VIS aktiv gesetzt werden.

Es ist wichtig den zu schaltenden State mit Bedacht zu wählen. Es wäre z.B. möglich den Heizkreis über Sommer / Winterbetrieb aus bzw. einzuschalten (z.B. km200: heatingCircuits.hc1.suWiSwitchMode)

Das hat den Nachteil, dass bei Adapterstop oder Netzwerkproblemen (km200 nicht erreichbar) der Heizkreis ggfs. permanent aus oder eingeschaltet bleibt und manuell am Thermostat neu gesetzt werden muss. Nach Murphy's Law passiert das in der Regel während des Urlaubs / Abwesenheit ....

Ich bevorzuge deshalb den „temporary Setpoint“ (z.B. heatingCircuits.hc1.temporaryRoomSetpoint).

Bei meinem RC310 sind diese temporären Einstellmöglichkeiten je Heizkreis vorhanden.

Dieser State hat den Vorteil, dass Werteänderung nur temporär bis zum nächsten Schaltzeitpunkt des Heizprogrammes gelten. Der Wert „0“ schaltet den HK aus, der Wert „-1“ wieder an auf Automatikbetrieb. (Es wäre aber auch möglich eine feste Temperatur festzulegen: z.B. „21“ Grad.)

Ich nehme den Automatikbetrieb, damit eine automatische Heizkreisabschaltung nach Erreichen der Außentemperschwelle des Heizkreises, trotz der aktiven Wärmebedarfs-Regelung weiterhin funktioniert.

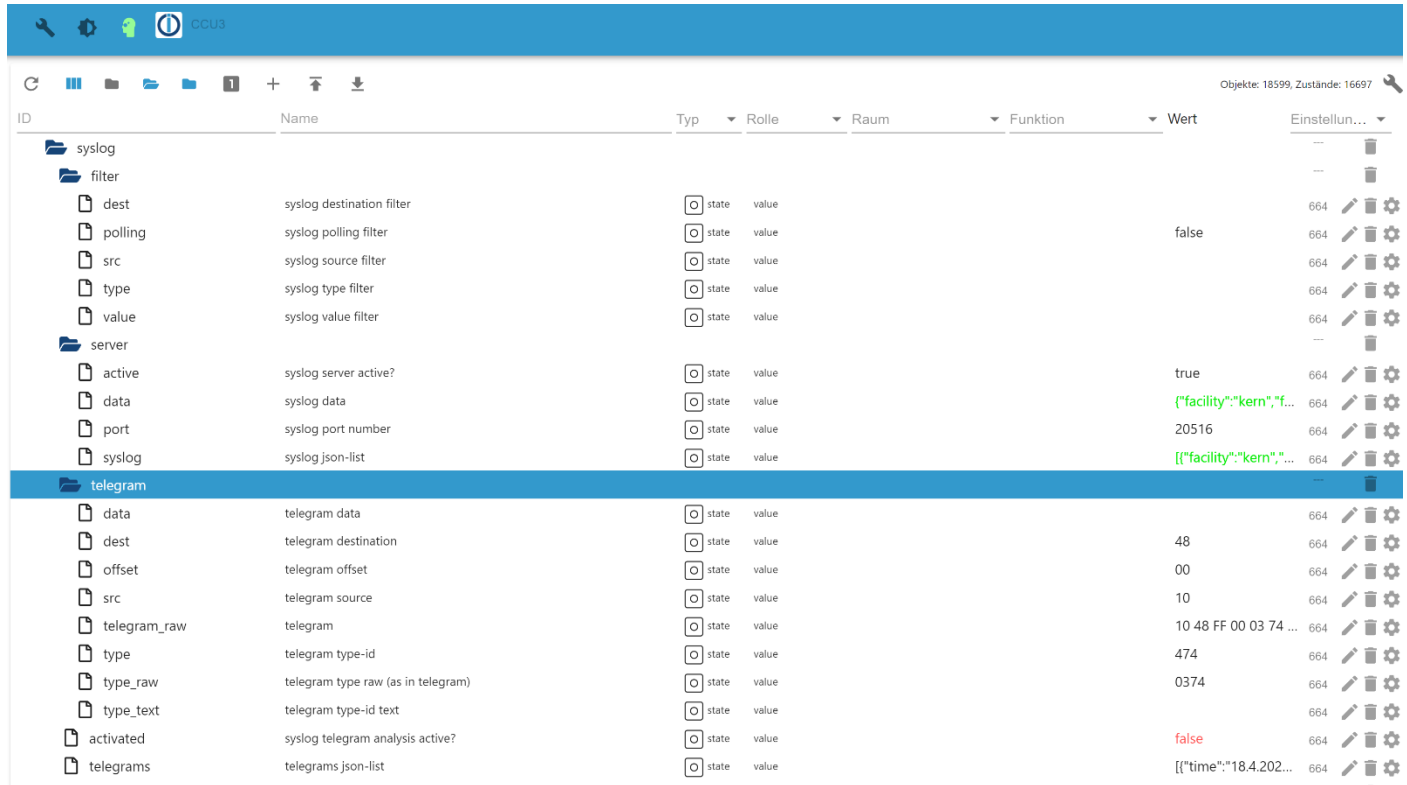
Die Gewichtung der Heizkörper, der Einschaltsschwelle und der „*minimum delta*“ des Heizkreises sollten so festgelegt werden, dass bei kleinster Modulation die Wärmeleistung des Wärmeerzeugers hinreichend lang aufgenommen werden kann.

Bei einem „*minimum delta*“ von 1° ist bei meinen Homematic Thermostaten sichergestellt, dass die Ventile geöffnet sind. Wenn alle Räume / Thermostate gleich gewichtet sind (z.B. mit 1) und die Einschaltsschwelle des Heizkreises auch 1 ist, dann wird jeder Heizbedarf eines Raumes den Brenner einschalten.

## Funktionen des eingebauten Syslog Servers

Der Syslog-Server läuft - wenn eingeschaltet – permanent mit und analysiert 100% des Telegrammverkehrs auf dem EMS-Bus der Heizungsanlage. Bitte beachten, dass dies durchaus CPU-intensiv ist. Bei meinem Raspberry PI4 mit 4 GB aber kein Problem. Auf einem PI3 würde ich das aber eventuell sein lassen.

Der Syslog Programmteil legt die folgende States-Struktur an:



The screenshot shows the ioBroker interface with the following states structure:

ID	Name	Typ	Rolle	Raum	Funktion	Wert	Einstellun...
<b>syslog</b>							
filter							
dest	syslog destination filter	state	value				664
polling	syslog polling filter	state	value			false	664
src	syslog source filter	state	value				664
type	syslog type filter	state	value				664
value	syslog value filter	state	value				664
<b>server</b>							
active	syslog server active?	state	value			true	664
data	syslog data	state	value			{ "facility": "kern", "f...	664
port	syslog port number	state	value			20516	664
syslog	syslog json-list	state	value			[{"facility": "kern", "...	664
<b>telegram</b>							
data	telegram data	state	value				664
dest	telegram destination	state	value			48	664
offset	telegram offset	state	value			00	664
src	telegram source	state	value			10	664
telegram_raw	telegram	state	value			10 48 FF 00 03 74 ...	664
type	telegram type-id	state	value			474	664
type_raw	telegram type raw (as in telegram)	state	value			0374	664
type_text	telegram type-id text	state	value				664
activated	syslog telegram analysis active?	state	value			false	664
telegrams	telegrams json-list	state	value			[{"time": "18.4.202...	664

Unter „server“ wird der ankommende Syslog Daten-Stream abgelegt.

Unter „telegram“ wird das Telegramm in die Komponenten aufgeteilt:

- src - source
- dest – destination
- offset – aus Telegramm
- type – Telegram-Type (wie im ems-esp Projekt definiert)
- type\_raw: Telegram-Type 1:1 aus Telegram (warum unterschiedlich zu type ???)
- data – Datenteil des Telegramms
- telegram\_raw (gesamtes Telegramm)
- type text – text aus nicht-hex Telegrammen (Umsetzung type in Text)

Unter „filter“ können Filter für die Telegramm-Analyse gesetzt werden. Source, destination, type type und ob polling requests dargestellt werden sollen. Zusätzlich kann ein wert in hex vorgegeben werden, welcher gesucht werden soll (1 Byte).

Es sind dann in „telegrams“ alle Telegramme mit entsprechendem Filter als JSON-Tabelle gespeichert. Es ist dann einfach sich eine einfache VIS-View zur Darstellung zu bauen. So z.B.:

active ☐

src filter 

08: Boiler

dest filter 

ohne

polling ☐

type filter

Port: 20516

telegram: 08 00 FF 00 07 E4 00 00 00 30 48 00 00 00 00 00 AC

src: 08 dest: 00 type raw: 07E4 type: 8E4 offset: 00

data: 00 00 00 30 48 00 00 00 00 00

16.5.2022, 17:52:35 08 00 FF 00 07 E4 00 00 00 30 48 00 00 00 00 00 AC

16.5.2022, 17:52:35 08 00 E9 00 34 01 E2 01 E2 00 00 00 00 4B 3C 00 00 00 01 41 2F 00 07 D1 00 00 00 34 00 34 36

16.5.2022, 17:52:35 08 00 E3 00 04 00 00 00 00 00 00 00 00 00 00 01 B6 00 1E 50 00 00 00 00 00 AB

16.5.2022, 17:52:35 08 00 E4 1B 01 B1 80 00 7F FF 80 00 25

16.5.2022, 17:52:34 08 00 E4 00 10 20 30 48 00 CB 0F 01 B6 00 00 00 00 00 00 00 00 00 01 A2 00 01 10 00 01 B6 80 00 CE

16.5.2022, 17:52:34 Start

VIS view download: <https://github.com/tp1de/ioBroker.ems-esp/blob/main/vis/telegrams.txt>  
 Diese View benutzt den vis-inventwo adapter. Dieser muss installiert sein.

Hier am Beispiel mit scr filter 08 (boiler). Wenn active auf *true* gesetzt wird, dann werden die Telegramme dargestellt. Sobald active auf *false* gesetzt wird, dann wird dieser Zustand „eingefroren“.