Mit Hilfe des EMS-Collectors kann die Heizung verhältnismäßig einfach in OpenHAB eingebunden werden. Das dafür verwendete Protokoll heißt MQTT, wofür schon ein OpenHAB-Binding existiert.

Grundvoraussetzungen

- Eine aktive OpenHAB-Installation mit installiertem MQTT-Addon (im OpenHAB-APT-Repo ist das das Package 'openhab-addon-binding-mqtt')
- Ein MQTT-Broker-Dienst (z.B. Mosquitto - in Debian ist das das Package 'mosquitto' - dieser kann in der Standardkonfiguration belassen werden und muss nur gestartet werden)
- Eine funktionierende Datenquelle für den Collector, z.B. das NET-IO (siehe diese Anleitung)

Collector compilieren

- Wer das noch nie gemacht hat, dem sei zunächst die obige Anleitung empfohlen :)
- Collector auschecken: git clone git:*github.com/maniac103/ems-collector.git * MQTT-Bibliothek auschecken: git:*github.com/redboltz/mqtt_client_cpp
- In Collector-Verzeichnis wechseln: cd ems-collector/collector
- Makefile bearbeiten, dabei den Block 'Uncomment the following lines to build the collector with MQTT support...' suchen und die beiden Zeilen am Ende dieses Blocks auskommentieren (Raute am Anfang der Zeile entfernen)
- Compilieren: make

Collector installieren

Wenn alles gut ging, sollte jetzt eine Datei namens 'collectord' im Verzeichnis liegen. Der muss jetzt installiert und als Service gestartet werden - siehe hierfür wieder obige Anleitung :) Als Inhalt der ems-collector.conf empfiehlt sich hierbei folgendes:

```
mqtt-broker = MQTT_IP:1883
command-port = 7777
data-port = 7778
```

MQTT_IP ist die IP-Adresse (oder der Hostname) des Servers, auf dem Mosquitto läuft. command-port und data-port sind streng genommen optional, aber nützlich für's Debugging

MQTT testen

Man kann diesen Schritt auch überspringen, aber es ist immer hilfreich, mal zu schauen, ob alles geht 😉 Um zu überprüfen, ob der installierte Collector die EMS-Daten richtig weiterleitet, installieren wir den Mosquitto-Client (Debian-Paket 'mosquitto-clients') und warten darauf, EMS-Daten zu bekommen:

mosquitto_sub -v -t '/ems/#'

Nach ein paar Sekunden sollten die Namen der Werte und die eigentlichen Werte vorbeiscrollen.

EMS in OpenHAB einbinden

Jetzt fehlt nur noch ein Schritt: Das Anlegen von OpenHAB-Items für die EMS-Daten. Zunächst muss das MQTT-Binding konfiguriert werden, damit sich OpenHAB mit Mosquitto verbindet - siehe hierfür das OpenHAB-Wiki. Wenn dieses getan ist, fehlt nur noch eine .items-Datei, bei der jeder Eintrag in etwa wie folgt aussieht:

```
Number HeizungVorlaufHK1Soll "Vorlauf HK1 Soll [%.1f °C]"
{mqtt="<[mosquitto:/ems/sensor/hk1/targettemperature/value:state:default]"}
```

Sensoren:

```
/ems/sensor/3wayonww/value
/ems/sensor/burner/targetmodulation/value
/ems/sensor/burner/currentmodulation/value
/ems/sensor/errorcode/value
/ems/sensor/exhaust/currenttemperature/value
/ems/sensor/flameactive/value
/ems/sensor/flamecurrent/value
/ems/sensor/flamecurrent/value
/ems/sensor/heater/targettemperature/value
/ems/sensor/heater/currenttemperature/value
/ems/sensor/heater/pumpactive/value
/ems/sensor/heater/heaterstarts/value
/ems/sensor/heater/operatingminutes/value
/ems/sensor/heater/operatingminutes2/value
/ems/sensor/heater/heatingminutes/value
/ems/sensor/heater/currenttemperature/value
/ems/sensor/heateractive/value
/ems/sensor/heaterpump/currentmodulation/value
/ems/sensor/heatexchanger/currenttemperature/value
/ems/sensor/hk1/offoptimization/value
/ems/sensor/hk1/onoptimization/value
/ems/sensor/hk1/wwoverride/value
/ems/sensor/hk1/floordrying/value
/ems/sensor/hk1/frostprotectmodeactive/value
/ems/sensor/hk1/summermode/value
/ems/sensor/hk1/daymode/value
/ems/sensor/hk1/opmode/value
/ems/sensor/hk1/roomtargettemperature/value
/ems/sensor/hk1/roomcurrenttemperature/value
/ems/sensor/hk1/onoptimizationminutes/value
/ems/sensor/hk1/offoptimizationminutes/value
/ems/sensor/hk1/characteristic/value
/ems/sensor/hk1/roomtemperaturechange/value
/ems/sensor/hk1/requestedpower/value
/ems/sensor/hk1/partymode/value
/ems/sensor/hk1/pausemode/value
/ems/sensor/hk1/vacationmode/value
/ems/sensor/hk1/holidaymode/value
```

```
/ems/sensor/hk1/switchpointactive/value
/ems/sensor/hk1/targettemperature/value
/ems/sensor/ignitionactive/value
/ems/sensor/operatingminutes/value
/ems/sensor/outdoor/dampedtemperature/value
/ems/sensor/outdoor/currenttemperature/value
/ems/sensor/outdoor/currenttemperature/value
/ems/sensor/pressure/value
/ems/sensor/returnflow/currenttemperature/value
/ems/sensor/servicecode/value
/ems/sensor/systemtime/value
/ems/sensor/warmwaterminutes/value
/ems/sensor/warmwaterpreparationactive/value
/ems/sensor/warmwaterpreparations/value
/ems/sensor/warmwatersystemtype/value
/ems/sensor/warmwatertempok/value
/ems/sensor/ww/targettemperature/value
/ems/sensor/ww/currenttemperature/value
/ems/sensor/ww/daymode/value
/ems/sensor/ww/onetimeload/value
/ems/sensor/ww/desinfectionactive/value
/ems/sensor/ww/boostcharge/value
/ems/sensor/ww/sensor1failure/value
/ems/sensor/ww/sensor2failure/value
/ems/sensor/ww/failure/value
/ems/sensor/ww/desinfectionfailure/value
/ems/sensor/ww/loading/value
/ems/sensor/ww/flowrate/value
/ems/sensor/zirkpump/daymode/value
/ems/sensor/zirkpump/opmode/value
/ems/sensor/zirkpumpactive/value
```

Commands:

Steuerbefehle können mittel eines Commands an den EMS Bus weitergegeben werden. z.B. lautet der Befehl zum Umstellen des Heizkreises 1 der Heizung auf den Tagbetrieb:

- Topic: „/ems/control/hk1/mode“ Message: „day“

Ein MQTT Topic ist nach dem folgenden Pattern aufgebaut: <Präfix><Erste Ebene><Command> (ohne die jeweiligen Anführungszeichen)

Anhand von /ems/control/hk1/mode setzt sich der Topic wie folgt zusammen:

- Präfix: /ems/control/
- Erste Ebene: hk1/
- Command (hier da hk Command): mode

Präfix: „/ems/control/“

```
Erste Ebene
    "hk[1|2|3|4]\n"
```

```
    "ww\n"
    "uba\n"
    "rc\n"
    "raw\n" (optional Flag HAVE_RAW_READWRITE_COMMAND in Makefile must be
uncommented)
    "cache\n"
    "getversion\n"
    "OK"
```

```
hk Commands:
    "mode [day|night|auto]\n"
    "daytemperature <temp>\n"
    "nighttemperature <temp>\n"
    "temperatureoverride <temp>\n"
    "getholiday\n"
    "holidaymode <start:YYYY-MM-DD> <end:YYYY-MM-DD>\n"
    "vacationtemperature <temp>\n"
    "getvacation\n"
    "vacationmode <start:YYYY-MM-DD> <end:YYYY-MM-DD>\n"
    "partymode <hours>\n"
    "pausemode <hours>\n"
    "getactiveschedule\n"
    "selectschedule
[family|morning|early|evening|forenoon|noon|afternoon|single|senior|custom1|
custom2]\n"
    "getcustomschedule [1|2]\n"
    "customschedule [1|2] <index> unset\n"
    "customschedule [1|2] <index> [monday|tuesday|...|sunday] HH:MM
[on|off]\n"
    "scheduleoptimizer [on|off]\n"
    "mintemperature <temp>\n"
    "maxtemperature <temp>\n"
    "reductionmode [offmode|reduced|raumhalt|aussenhalt]\n"
    "heatingsystem [none|heater|floorheater|convection] [outdoor|indoor]\n"
    "vacationreductionmode [outdoor|indoor]\n"
    "maxroomeffect <temp>\n"
    "designtemperature <temp>\n"
    "roomtemperatureoffset <temp>\n"
    "frostprotectmode [off|byoutdoortemp|byindoortemp]\n"
    "frostprotecttemperature <temp>\n"
    "summerwinterthreshold <temp>\n"
    "reducedmodethreshold <temp>\n"
    "vacationreducedmodethreshold <temp>\n"
    "cancelreducedmodethreshold <temp>\n"
    "requestdata\n"
    "OK"
ww Commands:
    "mode [on|off|auto]\n"
    "temperature <temp>\n"
    "limittemperature <temp>\n"
    "loadonce\n"
```

```
    "cancelload\n"
    "getcustomschedule\n"
    "customschedule <index> unset\n"
    "customschedule <index> [monday|tuesday|...|sunday] HH:MM [on|off]\n"
    "selectschedule [custom|hk]\n"
    "showloadindicator [on|off]\n"
    "thermdesinfect mode [on|off]\n"
    "thermdesinfect day [monday|tuesday|...|sunday]\n"
    "thermdesinfect hour <hour>\n"
    "thermdesinfect temperature <temp>\n"
    "zirkpump mode [on|off|auto]\n"
    "zirkpump count [1|2|3|4|5|6|alwayson]\n"
    "zirkpump getcustomschedule\n"
    "zirkpump customschedule <index> unset\n"
    "zirkpump customschedule <index> [monday|tuesday|...|sunday] HH:MM
[on|off]\n"
    "zirkpump selectschedule [custom|hk]\n"
    "requestdata\n"
    "OK"
uba Commands
    "antipendel <minutes>\n"
    "hyst [on|off] <kelvin>\n"
    "burnermodulation <minpercent> <maxpercent>\n"
    "pumpmodulation <minpercent> <maxpercent>\n"
    "pumpdelay <minutes>\n"
    "geterrors\n"
    "schedulemaintenance [off | byhour <hours / 100> | bydate YYYY-MM-DD]\n"
    "checkmaintenanceneeded\n"
    "testmode [on|off] <burnerpercent> <pumppercent> <3wayonww:[0|1]>
<zirkpump:[0|1]>\n"
    "requestdata\n"
    "OK"
rc Commands:
    "mintemperature <temp>\n"
    "buildingtype [light|medium|heavy]\n"
    "outdoortempdamping [on|off]\n"
    "requestdata\n"
    "geterrors\n"
    "getcontactinfo\n"
    "setcontactinfo [1|2] <text>\n"
    "settime YYYY-MM-DD HH:MM:SS\n"
    "OK"
raw Commands:
    "read <target> <type> <offset> <len>\n"
    "write <target> <type> <offset> <data>\n"
    "OK"
cache Commands:
    "fetch <key>\n"
    "OK"

getversion commands
```

```
    "getversion"
```

Fertig

Damit haben wir die EMS-Sensorwerte in OpenHAB übernommen und können sie dort weiterverarbeiten (anzeigen, in Regeln verwenden, über die Persistenzschicht in eine Datenbank schreiben usw. usf.). Prinzipiell ist auch die Steuerung der Heizung über OpenHAB möglich; hiermit wird sich eine spätere Version dieser Anleitung beschäftigen.