



OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

Fakultät Informatik und Mathematik



Prof. Dr. Waas

Praktikum zum Fach Kommunikationssysteme/ Rechnernetze

Tutorial

IoT mit MQTT

(Message Queuing Telemetry Transport)

Mit microPython

Linux

(Version 06.02.2023 KVM)

VORWORT

Als angehende Ingenieure erwarten wir von Ihnen einen sorgfältigen und umsichtigen Umgang mit den Übungsgeräten. Bei dieser Übung kann durch falsche Bedienung, bzw. falsche Einstellungen durchaus etwas kaputtgehen. Also lesen Sie die Übungsanleitung genau und überlegen Sie jeden Schritt vorher, den sie durchführen. Dann kann nichts schiefgehen. Wir hoffen, dass Ihnen diese Übung viel Spaß und vor allem viel Erkenntnis bringt.

Der Micro-USB-Anschluss am Sensor ist ein Schwachpunkt und sehr empfindlich. Bitte stecken Sie dort nicht um, damit der Sensor ein paar Semester hält!

1 EINFÜHRUNG

Die folgende Übung für IoT kann direkt auf dem Labor-PC durchgeführt werden. An jedem Arbeitsplatz ist ein Sensor an einem USB-Port angeschlossen und wird darüber mit Strom versorgt. Die Sensoren kommunizieren über WLAN in einem privaten IP-Netzwerk.

Der Labor-PC ist über LAN mit dem privaten IP-Netzwerk der Sensoren verbunden.

Die Sensoren übertragen die Messdaten mittels des Netzwerkprotokolls MQTT zu einem sog. MQTT-Broker (siehe Diag. 1). Anwendungen oder andere Sensoren können die Messdaten vom Broker abholen. IoT-Cloud Anbieter wie Microsoft oder IBM verwenden ebenfalls dieses Konzept.

2 FUNKTIONSWEISE VON MQTT (MESSAGE QUEUING TELEMETRY TRANSPORT)

Sensoren, Aktoren und Anwendungen kommunizieren über einen sog. **Broker** miteinander. Das Konzept ähnelt einem Diskussionsforum. Die Daten werden am Broker in sog. **Topics** gespeichert.

Topics: Die Namen der Topics sind frei wählbare Strings mit beliebigen Zeichen. Topics können strukturiert werden, ähnlich einem Forum. Die Strukturebenen werden mit „/“ getrennt.

Bsp.:

| Topic | Daten |
|--|--------|
| /FakultaetIM/K142/Sensor18/Status | online |
| /FakultaetIM/K142/Sensor18/Temperatur | 23 °C |
| /FakultaetIM/K142/Sensor18/Luftfeuchte | 45 % |
| /Berlin/Schaltwerk2/Relais102 | On |

Publish: Der Produzent (Sensor oder App) sendet Daten an ein Topic am Broker und umgekehrt.

Subscribe: Der Abonnent (Sensor oder App) abonniert ein Topic, d.h. er holt die Daten vom Broker in regelmäßigen Abständen, sobald diese verfügbar sind.

Retain: Der Broker speichert jeweils das letzte Datum (=Wert) eines Topics, den er mittels Publish erhalten hat. Wenn ein Sensor oder eine App ein Topic neu abonniert, kann der Broker

sofort den letzten Stand übermitteln. Andernfalls müsste der Abonnent warten bis ein neues Datum (=Wert) in das Topic geschrieben wurde.

Last Will/Testament: Jeder Sensor/Aktor oder jede App kann auf dem Broker ein sog. **Last Will**, oder auch **Testament** genannt, hinterlegen. Dabei wird ein vorbereiteter Wert in ein vorgegebenes Topic geschrieben, wenn ein Sensor, Aktor oder eine App offline geht. Zum Beispiel kann der Broker im Topic /FakultaetIM/K142/Sensor18/Status den Wert offline eintragen, wenn die Verbindung zum Sensor abreißt.

Das folgende Diagramm verdeutlicht die MQTT Kommunikation:

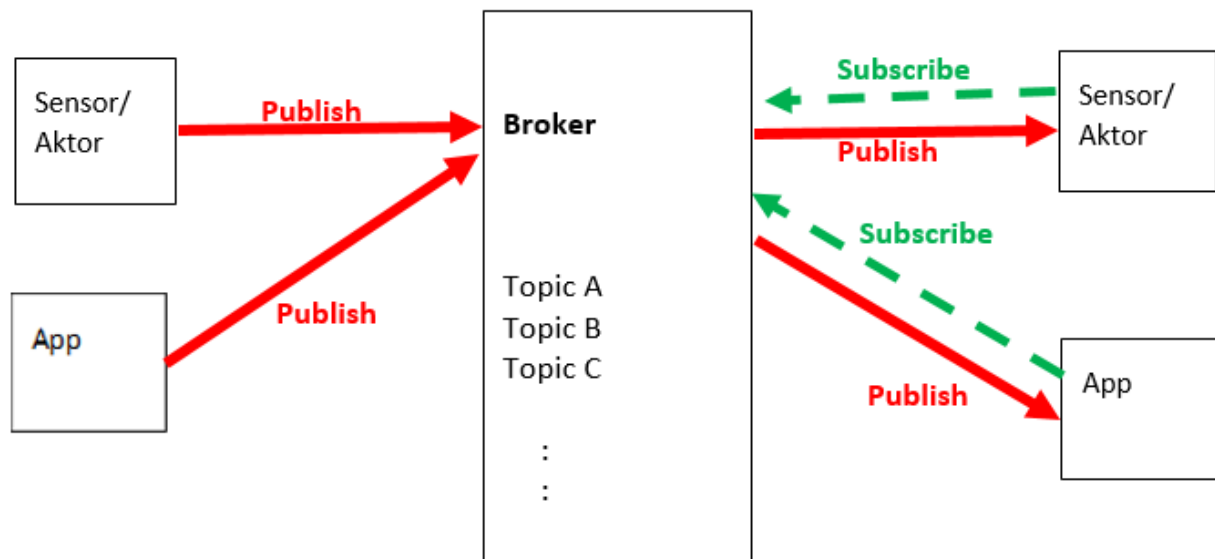
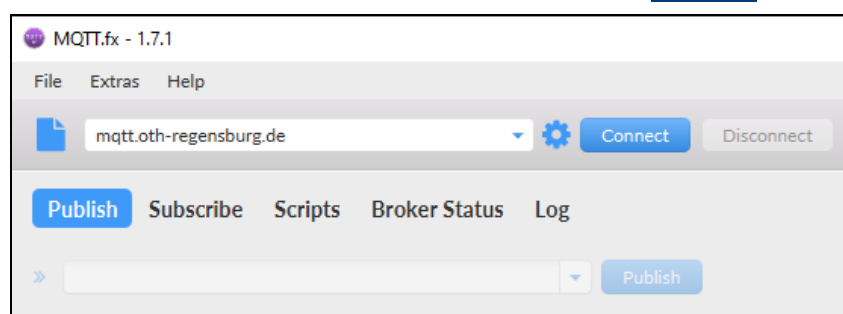


Diagramm 1: MQTT Kommunikation

3 PUBLISH UND SUBSCRIBE MIT MQTT.fx

Im Labor ist mit der IP-Adresse **mqtt.oth-regensburg.de** ein MQTT Broker (vom Typ Mosquitto unter Debian) installiert. Verwenden Sie jetzt diesen Broker um Sensordaten zu simulieren (**zum** Broker=Publish; **vom** Broker=Subscribe, die Daten sendet der Broker dann mittels Publish). Später soll der echte Sensor am Arbeitsplatz die Topics senden und empfangen.

- Öffnen Sie das Programm MQTT.fx auf dem **Desktop**.



- Wählen Sie den Broker: **mqtt.oth-regensburg.de** und klicken Sie auf **Connect**. Das Symbol rechts oben wird grün, wenn die Verbindng besteht.



Hinweis: Falls die Verbindung zum Broker immer wieder abbricht, sollten Sie die Client-ID im MQTT.fx ändern. Evtl. gibt's die Client-ID mehrfach. Klicken Sie auf das Zahnrad, wählen Sie links den Broker **mqtt.oth-regensburg.de** und klicken sie rechts in der Zeile von Client-ID auf **Generate**.

Client ID: 9b3f85e4e85a4c1386edb9c4fc5dc1b5 Generate

Dann unten auf den **Apply** Button und mit **Cancel** kommen Sie zurück ins Programm.

- Klicken Sie auf **Subscribe** und fügen Sie eine Subscription mit dem **Topic** `/#` hinzu. Klicken Sie dann daneben auf **Subscribe**. Das Topic wird auf der linken Seite in die Liste der aktiven Subscriptions aufgenommen. Auf der rechten Seite des Fensters werden die erhaltenen Topics aufgeführt und darunter deren Inhalte angezeigt.

The interface shows the 'Subscribe' tab selected. A subscription for `/#` has been added. The right pane displays a list of received topics: `/PC10/Temperature/Humidity`, `/PC10/Lum`, `/PC10/Switch`, `/PC10/LED`, and `/PC10/LED`. Each topic has a corresponding QoS level (QoS 0) and a timestamp. The bottom pane shows the message content for the selected topic, which is `0`.

- Erstellen Sie nun selbst ein Topic. Grundsätzlich ist fast jede Zeichenfolge erlaubt. Zur Übersicht verwenden Sie bitte folgende Struktur: `/Raum/Arbeitsplatz/Messwerttyp` (Bsp: `/K142/PC3/Temperatur`). Denken Sie sich selbst etwas aus.

The interface shows the 'Publish' tab selected. A new topic `/K142/PC3/Temperatur` is entered in the text field. The 'Publish' button is visible next to it. The message content field below shows `23.5`.

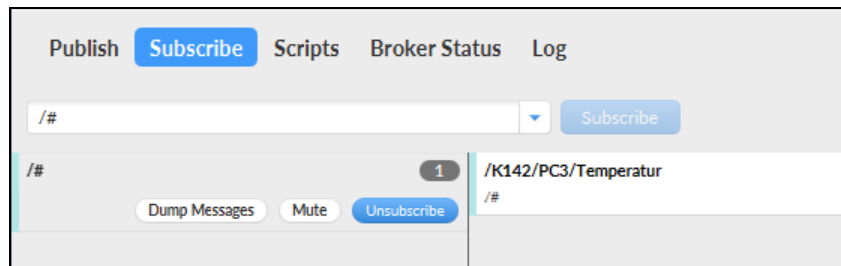
- Wählen Sie **QoS 0** aus. Das bedeutet, dass das Topic einmal gesendet wird ohne Garantie, dass es auch empfangen wird.

The QoS selection buttons are shown: **QoS 0** (selected), **QoS 1**, **QoS 2**, and **Retained**.

Hinweis:

- **QoS 0:** Topic wird einmal gesendet, ohne Empfangsbestätigung. Kommt an oder nicht.
- **QoS 1:** Topic wird gesendet mit Empfangsbestätigung. Falls keine Empfangsbestätigung kommt, wird die Übertragung wiederholt.
- **QoS 2:** Topic wird gesendet mit Empfangsbestätigung. Falls keine Empfangsbestätigung kommt, wird nachgefragt. Es wird sicher gestellt, dass das Topic nur einmal empfangen wurde.

- Klicken Sie dann rechts auf **Publish**, damit das Topic gesendet wird.
- Schalten Sie wieder auf **Subscription** um damit Sie Ihr gesendetes Topic sehen können.

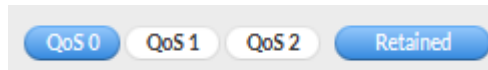


Hinweis: Wenn zu viele Topics zur selben Zeit gesendet werden, wird die Anzeige unübersichtlich. Besser ist es dann, die Subscription von „/#“ bzw: „/#“ genauer zu spezifizieren. Zum Beispiel auf den Beginn Ihres Topics „/K142/#“ oder noch genauer Topic + Arbeitsplatz „/K142/PC3/#“. Wichtig ist, dass am Ende das „/#“ Zeichen steht.

- Senden Sie nun dasselbe Topic noch ein paar Mal aber mit anderen Werten und prüfen Sie, dass alle Meldungen erscheinen. Merken Sie sich die letzte Meldung.
- **Beenden** Sie nun den MQTT.fx Browser und **starten** Sie ihn wieder neu. Gehen Sie auf **Subscribe** und führen Sie das Subscribe (/#) von vorher aus und versuchen Sie Ihr letztes Topic zu finden.

Ergebnis: Sie können es nicht finden.

- Gehen Sie nun wieder auf **Publish** und senden Sie das letzte Topic nochmal mit dem Unterschied, dass Sie jetzt zusätzlich das **Retain** einschalten.

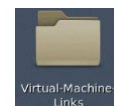


- **Beenden** Sie nun den MQTT.fx Browser und **starten** Sie ihn wieder neu. Gehen Sie auf **Subscribe** und führen Sie das Subscribe (/#) von vorher aus und versuchen Sie Ihr letztes Topic zu finden.

Ergebnis: Der Broker hat sich das letzte Topic gemerkt und kann es sofort senden, wenn es mit Subscribe angefordert wird.

4 STARTEN DER ÜBUNGSUMGEBUNG

- Öffnen Sie den Ordner **Virtual Machines Links** auf dem Desktop.
- Starten Sie die folgende virtuelle Maschine durch Doppelklick:
iot

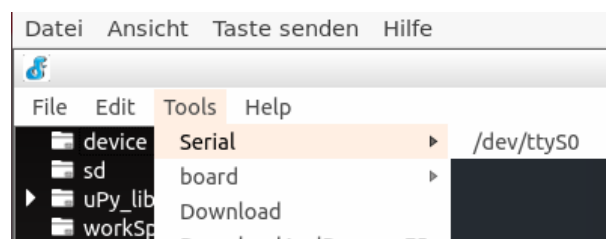


5 SENSOR KONFIGURIEREN

Der Name des Sensors muss mit dem Namen Ihres Arbeitsplatzes übereinstimmen!



- Auf dem Sensor ist microPython für ESP8266 installiert.
- Starten Sie auf der virtuellen Maschine iot das Programm **uPyCraft** auf dem Desktop. Damit können Sie microPython Programme auf den Sensor laden, verändern, herunterladen und ausführen. Es ist auch möglich, interaktiv Python Befehle auszuführen (REPL=Read Eval Print Loop). Das Terminalfenster im Hintergrund können Sie minimieren.
- Öffnen Sie das **Tools Menü** und wählen Sie **Serial**. Wählen Sie dann die serielle Schnittstelle für den Sensor aus (meist **/dev/ttyS0**)

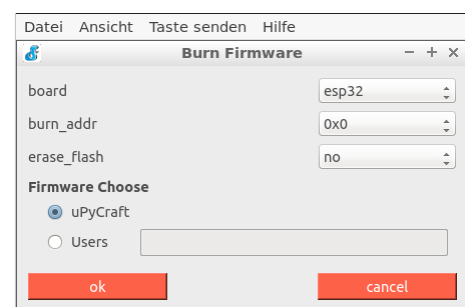


Hinweis: Es muss unten das **Python Eingabesymbol** (REPL=Read-Evaluate-Print-Loop) **>>>** angezeigt werden, dann ist die Verbindung aufgebaut. Falls dies nicht so ist, wählen Sie im **Tools-Menü** den Punkt **Stop** und sofort danach **Serial** und **/dev/ttyS0**. Wiederholen Sie die beiden Schritte bei Bedarf noch ein paar Mal, bis das Python Eingabesymbol angezeigt wird.

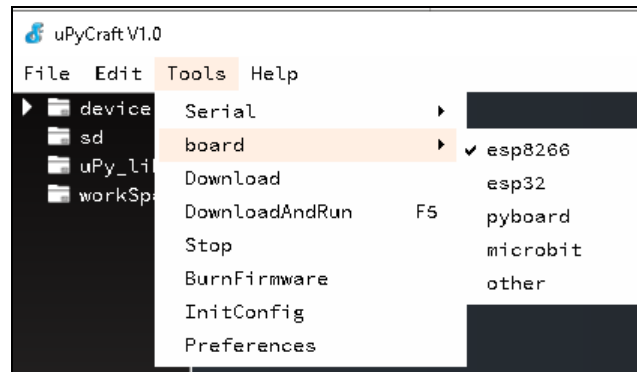
Die serielle Verbindung zum Sensor klappt nicht auf immer Anhieb. Falls das nebenstehende Bild angezeigt wird, klicken Sie es weg und wiederholen Sie die Verbindung zu **/dev/ttyS0** ein paar Mal bis unten **>>>** kommt.

Falls es immer noch nicht klappt, schließen Sie die uPyCraft Anwendung und starten Sie sie neu. Stecken Sie nicht den USB-Port um, weil Sie dann eine andere Portnummer bekommen, die nicht funktionieren wird.

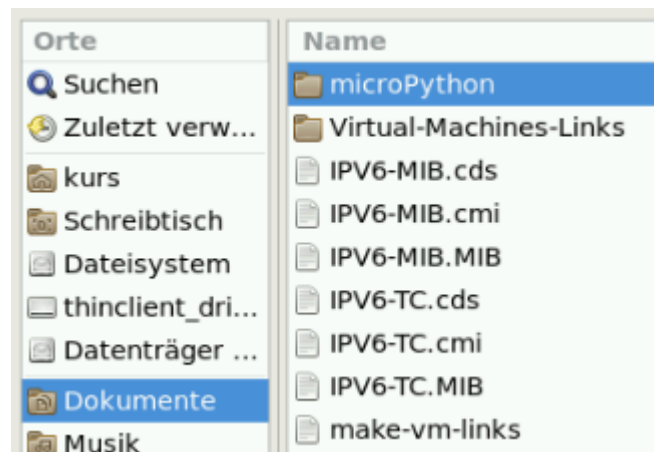
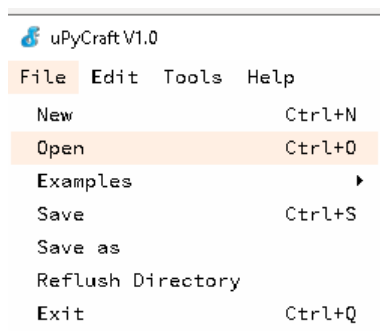
Bitte etwas Geduld haben. Wenn die Verbring mal funktioniert, gibt's kaum mehr Probleme.



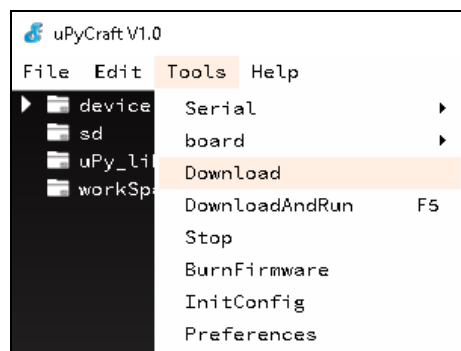
- Öffnen Sie wieder das **Tools Menü** und wählen Sie **board** aus. Wenn der Sensor erkannt wurde, wird als Board **esp8266** angezeigt.



- Öffnen Sie das **File Menü** und klicken Sie auf **Open**. Wählen Sie die Datei **Dokumente/microPython/umqttsimple.py** um sie zu laden.



- Öffnen Sie wieder das **Tools Menü** und wählen Sie **Download**. Damit wird die aktuell selektierte Datei (in diesem Fall **umqttsimple.py**) auf den Sensor gespeichert. Das ist die Bibliothek für das MQTT Protokoll.



Nach dem Download sollten Sie unten die folgende Anzeige sehen:

```
Ready to download this file, please wait!
.....
download ok
```

Hinweis: Sollte der Download mehrfach nicht funktionieren, weil evtl. die folgende Meldung angezeigt wird:
`already in download model, please wait,`
wählen Sie in diesem Fall im Tools-Menü **Stop** (evtl. mehrmals) und/oder drücken Sie die Eingabetaste (evtl. mehrmals) bis das REPL-Prompt „>>>“ kommt.
Falls das immer noch nicht hilft, bleibt nur noch die Option mit BurnFirmware im ToolsMenü die Firmware-Datei **S/micropython/esp8266...bin** auf dem Sensor neu zu flashen. Folgen Sie dazu den Anweisungen des Programms.

- Öffnen Sie wieder das **File Menü** und laden Sie die Datei **Dokumente/microPython/main.py**. Das ist der MQTT Client, der das Senden und Empfangen von Topics abwickelt.
- Öffnen Sie wieder das **Tools Menü** und wählen Sie **Download**. Damit wird die aktuell selektierte Datei (in diesem Fall **main.py**) auf den Sensor gespeichert.
- Öffnen Sie wieder das **File Menü** und laden Sie die Datei **Dokumente/microPython/boot.py**. Das ist das Programm für den Systemstart und die Grundeinstellungen. Es wird auch die WLAN-Verbindung aufgebaut.
- Finden Sie in der Datei **boot.py** den folgenden Text-Block. Passen Sie die Werte für Ihren Sensor an: In den Topics muss der Name des Sensors (steht auf dem Sensor) in Großbuchstaben eingetragen werden. (Beispiele: /PC1/..., /PC2/... /PC12/...). Der Rest des Topics sollte nicht verändert werden. Bitte setzen Sie das message intervall nicht kleiner als 20, damit der Broker und andere nachgeschaltete Systeme nicht überlastet werden. Die IP-Adresse des MQTT Brokers ist voreingestellt.

- ✓ **Hinweis bei Remotezugang:** Starten Sie die Eingabeaufforderung/Terminal.
Aus der Kopfleiste des Terminal Fensters können Sie entnehmen an welchem PC Sie arbeiten (siehe rfhpci...): Die folgende Tabelle gibt an, welcher Sensor dann an Ihrem PC angeschlossen ist.



kurs@rfhpci143: ~

| | | | |
|-----------------|-----------------|-----------------|------------------|
| rfhpci131 : PC1 | rfhpci134 : PC4 | rfhpci137 : PC7 | rfhpci140 : PC10 |
| rfhpci132 : PC2 | rfhpci135 : PC5 | rfhpci138 : PC8 | rfhpci141 : PC11 |
| rfhpci133 : PC3 | rfhpci136 : PC6 | rfhpci139 : PC9 | rfhpci142 : PC12 |

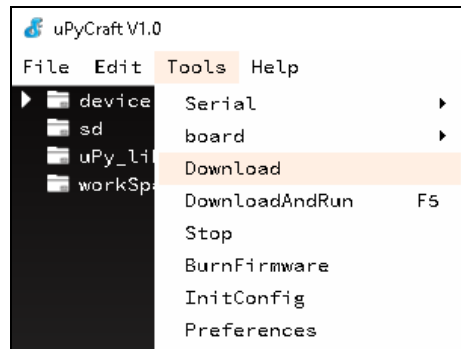
```
# START setup the correct values for your project

mqtt_server = '194.95.109.89'
topic_Temperature = b'/PC13/Temperature'
topic_Humidity = b'/PC13/Humidity'
topic_Switch = b'/PC13/Switch'
topic_LED = b'/PC13/LED'
topic_Lum = b'/PC13/Lum'
Lum_Factor = 0.1 # scale light values . . .
message_interval = 30 # time to send topics in seconds

# END setup the correct values for your project
```

Anmerkung: Sie dürfen sich gerne den Programmcode ansehen 😊.

- Öffnen Sie wieder das **Tools Menü** und wählen Sie **Download**. Damit wird die aktuell selektierte Datei (in diesem Fall **boot.py**) auf den Sensor gespeichert



- Geben Sie nun im unteren Teil des Fensters einmal die Eingabetaste ein, damit der für Python typische Eingabeprompt (REPL) `>>>` angezeigt wird. Sie können nun den folgenden Befehl eingeben. Bitte die Eingabe mit der **Eingabetaste** abschließen.

machine.reset()

```
Ready to download this file, please wait!
.....
download ok

>>> machine.reset()
```

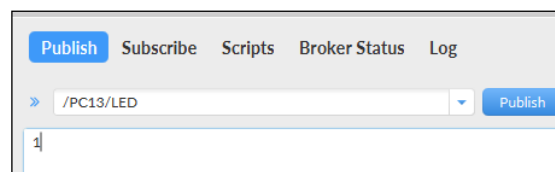
Anmerkung:

Sollte der obige Befehl nicht funktionieren, geben Sie ein: **import machine** und wieder holen Sie den obigen Befehl. Bitte die Eingabe immer mit der **Eingabetaste** abschließen.

Der Sensor wird gebootet, stellt eine WLAN-Verbindung her, verbindet sich zum Broker und beginnt die Topics mit den Messwerten für Temperatur, Luftfeuchte, Licht und Schaltereingabe an den Broker zu senden. Mittels Subscribe bekommt der Sensor die Topics zur Steuerung der LED. Die kryptischen Zeichen beim Start sind normal.

- Überprüfen Sie mit dem MQTT-Browser, ob die Topics Ihres Sensors korrekt gesendet werden. Abonnieren Sie dazu alle Topics Ihres Sensors. (Bsp.: /PC13/#)
- Prüfen Sie, ob Sie die LED auf dem Sensor schalten können. Verwenden Sie dazu **Publish** auf dem MQTT-Browser. Geben Sie das korrekte Topic (Beispiel: /PC13/LED) ein. mit den Werten 0 für OFF bzw. 1 für ON. (Geht nicht bei Remote Zugang)

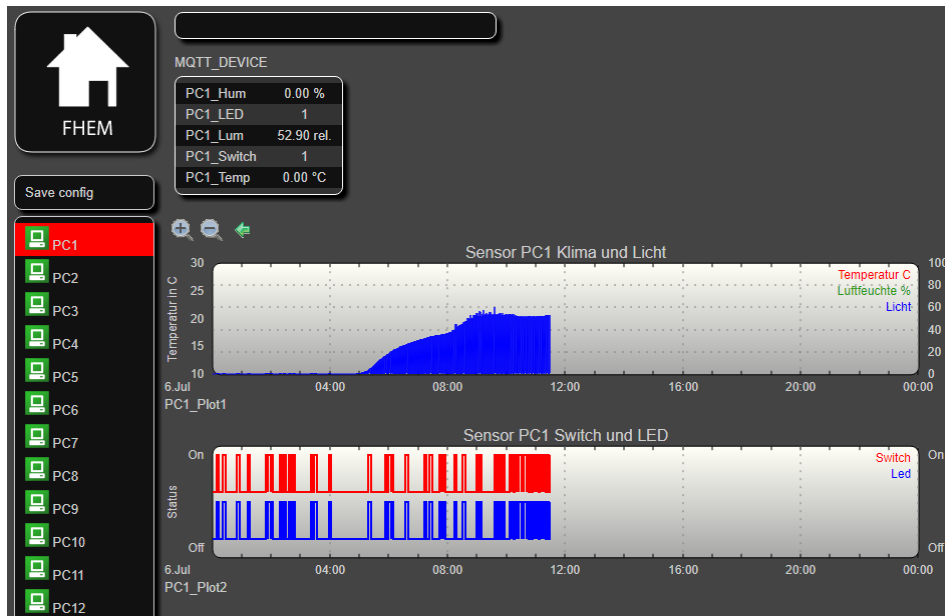
Beispiel für PC13:



6 SENSORWERTE AN APP (FHEM) WEITERREICHEN

Zur Aufbereitung und Verarbeitung von Sensordaten ist eine Applikation nötig. Zu Testzwecken ist im Labor dafür eine Home Automation Anwendung vom Typ FHEM (www.fhem.de) installiert. Es ist allerdings nötig, dass Sie die Topics Ihrer Sensoren so anpassen, wie Sie die Applikation benötigt.

- Verbinden Sie sich mit einem Web-Browser auf die folgende URL.
<https://mqtt.oth-regensburg.de:8083/fhem>
Benutzer: **comlab** und Passwort: **comlab**




- In der **Navigationsleiste links** finden Sie Ihren Sensor (PC1 bis PC15). Öffnen Sie dort Ihren Sensor und betrachten Sie die angelegten **Sensor-Objekte** (PCn_Hum, PCn_LED, etc.). Öffnen Sie jedes Sensorobjekt mit Doppelklick und entnehmen Sie in der Zeile **subscribeReading_...** das jeweils eingestellte Topic (siehe folgendes Bild).

| Attributes | | |
|-------------------------|----------------------|------------|
| IODev | mqtt | deleteattr |
| event-on-update-reading | H | deleteattr |
| room | PC1 | deleteattr |
| stateFormat | H % | deleteattr |
| subscribeReading_H | /PC1/Humidity | deleteattr |

Hinweis: Ihr Sensor muss seine Daten in die passenden Topics schreiben. Die Topics haben Sie in der Datei boot.py auf dem Sensor eingestellt. Falls keine Sensordaten kommen, überprüfen Sie die Topics auf dem Sensor.

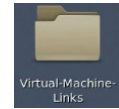
Beispiel: Wenn der Systemname des Sensors also **PC1** lautet, und der Valuenname **Humidity**, dann wird daraus das Topic **/PC1/Humidity** gebildet und die Messdaten in dieses Topic geschrieben.

- Prüfen Sie, dass Ihre Sensorwerte auf der App aktualisiert werden (Zeitstempel rechts in den Anzeigen prüfen)
- Lassen Sie etwas Zeit vergehen, sodass die Sensordaten auch in den Grafiken zu sehen sind.
Durch Klicken auf diese Symbole  über der Grafik können die Grafen gedehnt oder gestaucht werden oder die Zeitachse verschoben werden.
- Der Schalter Ihres Sensors ist in der App mit der LED verlinkt. Die LED sollte der Schalterstellung folgen. Probieren Sie das aus.

7 MQTT MELDUNGEN ANSEHEN

Hinweis: Sollte diese Übung nicht funktionieren, so finden Sie im Anhang eine kurze Hilfe.

● Öffnen Sie den Ordner **Virtual Machines Links** auf dem Desktop am PC.



● Starten Sie die folgende virtuelle Maschine durch Doppelklick:
mqtt

Die virtuelle Maschine führt einen mqtt Broker aus. Sie können die Anzeige der VM auf die Taskleiste **minimieren**. Eingaben sind hier nicht nötig.

● Auf dem **Sensor** müssen Sie nun in der Datei **boot.py** die IP-Adresse der MQTT Brokers der VM einstellen. Die IP-Adresse dafür ist die **IP-Adresse Ihres PC** (steht angeschrieben am PC: 194.95.109...) oder sie können Sie über das folgende Kommando finden:

- Öffnen Sie eine Eingabeaufforderung/Terminal.
- Geben Sie ein: **ifconfig br0**
- Die IP-Adresse steht bei inet und beginnt mit 194.95.109...

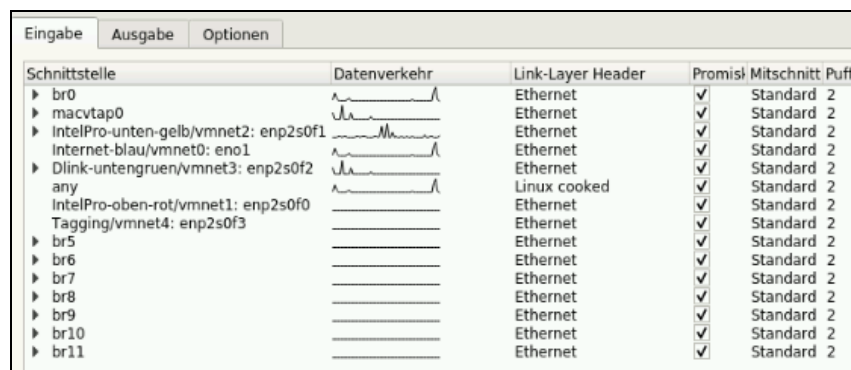


● Speichern Sie die geänderte Datei **boot.py** auf den Sensor und führen Sie unten in der Python Kommandozeile das Kommando **machine.reset()** aus. Der Sensor startet neu müsste nun mit dem Broker in der VM eine Verbindung aufbauen, die Sie auf Ihrem PC mit dem Programm Wireshark beobachten können.

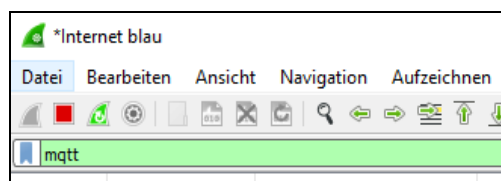
● Starten Sie nun auf Ihrem PC das Programm **Wireshark**.



● Klicken Sie im Bereich Aufzeichnen doppelt auf **Internet-blau**.



● Geben Sie links oben in der Filterzeile **mqtt** ein und drücken Sie die **Eingabe-Taste** oder klicken Sie in derselben Zeile, ganz rechts den **Pfeil nach rechts** an. Wenn der Filter richtig eingegeben wurde, wird das Feld grün.



● Nun können Sie die mqtt Meldungen sehen, die der Sensor an den Broker auf Ihrem Labor-PC sendet.

*Internet blau

Datei Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe

mqtt

| No. | Source | Destination | Protocol | Info |
|------|-----------------|-----------------|----------|--|
| 1014 | 192.168.109.133 | 194.95.109.102 | MQTT | Publish Message [/PC3/Klima/Temperature] |
| 1015 | 194.95.109.102 | 192.168.109.133 | MQTT | Publish Message [/PC3/Klima/Temperature] |
| 1027 | 192.168.109.133 | 194.95.109.102 | MQTT | Publish Message [/PC3/Klima/Humidity] |
| 1028 | 194.95.109.102 | 192.168.109.133 | MQTT | Publish Message [/PC3/Klima/Humidity] |

<

> Frame 1014: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface 0

> Ethernet II, Src: Cisco_e5:a9:80 (70:ca:9b:e5:a9:80), Dst: HewlettP_31:2a:9a (10:e7:c6:31:2a:9a)

> Internet Protocol Version 4, Src: 192.168.109.133, Dst: 194.95.109.102

> Transmission Control Protocol, Src Port: 10275, Dst Port: 1883, Seq: 121, Ack: 200, Len: 31

MQ Telemetry Transport Protocol, Publish Message

> Header Flags: 0x31, Message Type: Publish Message, QoS Level: At most once delivery (Fire and Msg Len: 29

Topic Length: 22

Topic: /PC3/Klima/Temperature

Message: 27.00

- Klicken Sie im mittleren Fenster von Wireshark auf die Zeilen mit **MQ**. Dann klappen Detailinformationen aus der Nachricht auf, z.B. die genauen Messwerte.
- Wenn Sie auf ein Topik klicken, können Sie im ganz unteren Fenster sehen, wie es zusammen mit dem Messwert als ASCII-Text in die Meldung eingebaut wurde.

MQ Telemetry Transport Protocol, Publish Message

> Header Flags: 0x31, Message Type: Publish Message, QoS Level: At most once delivery (Fire and Msg Len: 29

Topic Length: 22

Topic: /PC3/Klima/Temperature

Message: 27.00

| | | |
|------|---|-------------------|
| 0000 | 10 e7 c6 31 2a 9a 70 ca 9b e5 a9 80 08 00 45 00 | ...1*.p.E. |
| 0010 | 00 47 03 a4 00 00 7f 06 da 19 c0 a8 6d 85 c2 5f | .G.....m.. |
| 0020 | 6d 66 28 23 07 5b 00 03 be 5b c2 55 51 88 50 18 | mf(#.[...[.UQ.P. |
| 0030 | 16 09 9e 5e 00 00 31 1d 00 16 2f 50 43 33 2f 4b | ...^..1..../PC3/K |
| 0040 | 6c 69 6d 61 2f 54 65 6d 70 65 72 61 74 75 72 65 | lima/Tem perature |
| 0050 | 32 37 2e 30 30 | 27.00 |

7 ENDE DER ÜBUNG

- Beenden Sie alle **Programme** und **virtuelle Maschinen**! Im Rahmen der Übung an Ihrem Arbeitsplatz erzielte Messergebnisse können Sie im Labor auf Ihren Memorystick zur späteren Nachbearbeitung abspeichern. Gewonnene sicherheitsrelevante Informationen insbesondere Passwörter, dürfen nicht weitergegeben oder unbefugt verwendet werden. Geht leider nicht im Remotebetrieb.
- Loggen** Sie sich aus dem Labor-PC aus!
- Lassen Sie den PC weiterlaufen. Er wird automatisch ausgeschaltet.

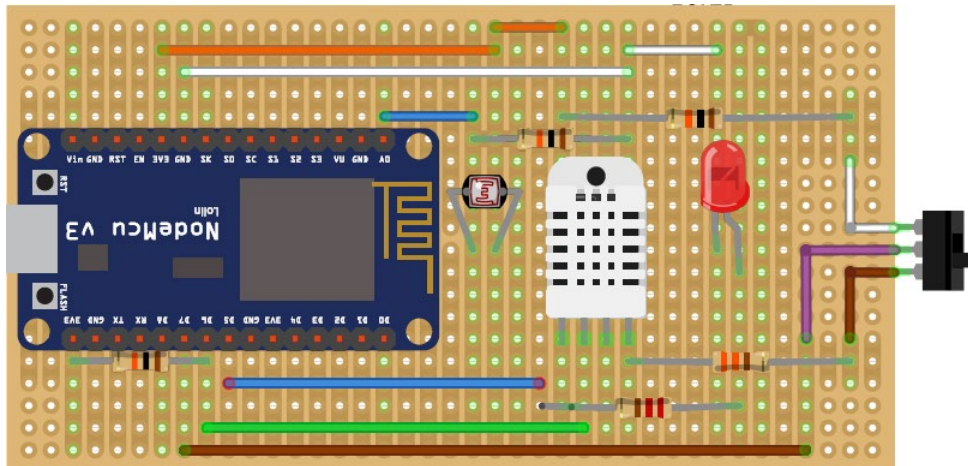
Bitte hinterlassen Sie Ihren Arbeitsplatz in ordentlichem Zustand!

Entsorgen Sie Mitgebrachtes selbst!

Schieben Sie den Stuhl an den Tisch!

8 ANHANG

Schaltung des Sensors



Problemhilfe für Übungsteil 6

Für den Broker wird DNAT via iptables ermöglicht. Der Broker hat die IP-Adresse **192.168.88.128** fest eingestellt.

```
iptables -t nat -A PREROUTING -p tcp --dport 1883 -j DNAT --to-destination 192.168.88.128
```

boot.py

```
import time
from umqttsimple import MQTTClient
import ubinascii
import machine
from machine import Pin
from machine import ADC
from machine import WDT
import micropython
import network
import esp
import os
import dht

# START setup the correct values for your project
mqtt_server = '194.95.109.89'
topic_Temperature = b'/PC10/Temperature'
topic_Humidity = b'/PC10/Temperature/Humidity'
topic_Switch = b'/PC10/Switch'
topic_LED = b'/PC10/LED'
topic_Lum = b'/PC10/Lum'
Lum_Factor = 0.1 # scale light values to interval 0
.. 100
message_interval = 30 # time to send topics in seconds
# END setup the correct values for your project

# It is not necessary to change code beneath this line
```

```

d = dht.DHT22(machine.Pin(12))    # Tem/Hum Sensor an GPIO 12
led = Pin(14, Pin.OUT)            # LED Pin an GPIO 14
adc = ADC(0)                      # Lichtsensor an Pin A0
sw = Pin(13, Pin.IN)              # Switch an GPIO 13
client_id = ubinascii.hexlify(machine.unique_id())
last_message = 0
sw_int = False

def int_switch(p):
    global sw_int
    sw_int = True

sw.irq(trigger=Pin.IRQ_RISING | Pin.IRQ_FALLING, handler=int_switch)

esp.osdebug(None)
import gc
gc.collect()

ssid = 'K142'
password = 'K142-Lzr8-9Fa7'
station = network.WLAN(network.STA_IF)

station.active(True)
station.connect(ssid, password)

while station.isconnected() == False:
    pass

print('Connection successful')
print(station.ifconfig())

```

main.py

```

def sub_cb(topic, msg):
    print('Received: ', topic, msg)
    if topic == topic_LED and msg == b'1':
        led.value(1)
        print('LED ON')
    if topic == topic_LED and msg == b'0':
        led.value(0)
        print('LED OFF')

def connect_and_subscribe():
    global client_id, mqtt_server, topic_LED
    client = MQTTClient(client_id, mqtt_server, user='comlab',
password='comlab')
    client.set_callback(sub_cb)
    client.connect()
    client.subscribe(topic_LED)
    print('Connected to %s MQTT broker, subscribed to %s topic' %
(mqtt_server, topic_LED))
    return client

def restart_and_reconnect():

```

```

    print('Connected to %s MQTT broker, subscribed to %s topic' %
          (mqtt_server, topic_LED))
    time.sleep(10)
    machine.reset()

try:
    client = connect_and_subscribe()
except OSError as e:
    restart_and_reconnect()

while True:
    try:
        client.check_msg()

        if sw_int:
            sw_int = False
            msg=str(sw.value())                # read digital input
            client.publish(topic_Switch, msg)
            print('Sent: ', topic_Switch, msg)

        if (time.time() - last_message) > message_interval:
            d.measure()
            msg = str(d.temperature())        # read Temperature
            client.publish(topic_Temperature, msg)
            print('Sent: ', topic_Temperature, msg)

            msg = str(d.humidity())           # read Humidity
            client.publish(topic_Humidity, msg)
            print('Sent: ', topic_Humidity, msg)

            msg=str adc.read() * Lum_Factor) # read analog value, 0-1024
            client.publish(topic_Lum, msg)
            print('Sent: ', topic_Lum, msg)

            msg=str(sw.value())              # read digital input
            client.publish(topic_Switch, msg)
            print('Sent: ', topic_Switch, msg)
            last_message = time.time()

    except OSError as e:
        restart_and_reconnect()

```