



Projekt Peer Review (PPR)

Physics for IoT (BTI2001) 21

Studiengang:	Informatik
Autor:	Mac Müller
Betreuer:	Prof. Florian Löwenthal
Datum:	04.06.2021

¹ Bild Quelle: <https://www.warema.de/referenzen/hochzwei-allmend.php>

Inhaltsverzeichnis

1	Projektidee.....	3
1.1	Einleitung.....	3
1.2	Idee.....	3
2	Studium	4
3	Projektplanung.....	5
3.1	Material	5
3.1.1	ESP32 Board mit einem TMP36 Sensor und einem Luxmeter	5
3.1.2	Raspberry Pi 4b als MQTT Broker.....	5
3.1.3	W&T Web-IO Digital 12x IN, 6xRelais Out.....	5
3.2	Etappierung / Zeitplan.....	6
4	Umsetzung.....	7
4.1	Entwurf der Daten via MQTT.....	7
4.2	W&T 6xRelais Out.....	8
4.3	MQTT Broker	11
4.3.1	Ein MQTT Broker auf dem Raspberry Pi installieren	11
4.3.2	Ein Zugang für MQTT Broker einrichten.....	11
4.4	ESP32 Board.....	11
4.4.1	Business Logic.....	11
4.4.2	Mit WLAN verbinden	12
4.4.3	Mit MQTT verbinden - Last will definieren	13
4.4.4	Zeit Server einrichten	13
4.4.5	Status des ESP32.....	14
5	Meilenstein und Zeitplan	16
6	Testphase.....	17
6.1	Funktion Test.....	17
6.1.1	Broker Test	17
6.1.2	W&T I/O Test.....	17
6.1.3	ESP32 und Hauptfunktionstest.....	18
6.2	Last Test.....	18
6.2.1	Alle Geräte über 24 Stunden laufen lassen.....	18
6.2.2	Die Lamellen 5 Mals hintereinander automatisch schauen.....	18

1 Projektidee

1.1 Einleitung

Im Rahmen des Moduls "BTI2001: Physics for IoT" soll ein Projekt mit MAKE KIT für ESP32 ausgeführt werden. Ziel bei dieser vorliegenden Arbeit ist den Überblick über das Projektkonzept von Lamellen Steuerung mit ESP32 bis zur Realisierung zu zeigen.

1.2 Idee

Ein Student von der Berner Fachhochschule wohnt zurzeit in einem Hochhaus bei der Allmend Messe in Luzern. Die Fassade besteht hauptsächlich aus Glas-Fenster. Die Wohnungen wurden zum Schutz von der Sonne mit den elektrischen Lamellen ausgerüstet. Im normalen Fall werden die Lamellen von einer automatischen Zentralsteuerung gefahren. Der Mieter hat die Möglichkeit mit Tasten die Lamellenstellung anzupassen beziehungsweise zu übersteuern. Nach einem manuellen Eingriff werden die Lamellen nicht mehr automatisch gesteuert, bis der Bewohner sie wieder auf automatisch stellt. Ausgenommen sind Ereignisse wie Wind, Frost und die Vogelschwarm Erkennung. In solchen Fällen übernimmt die Zentralsteuerung die Kontrolle der Lamellen, um den weiter Schäden zu vermeiden.

Sehr oft am Abend will der Bewohner die Lamellen manuell anpassen und vergisst sie anschliessend wieder auf automatisch zu stellen. Dies bedeutet das am nächsten Tag, selbst wenn die Sonnen scheint, die Lamellen nicht runterfahren werden. Wenn der Bewohner dann am Abend zurück kommt - ist es in der Wohnung bis 40 Grad warm.

Bei diesem Problem kann der Einsatz vom ESP32 mit einem Schalter eine Lösung sein.

Das ESP32 soll die Temperatur und die Helligkeit von der Wohnung überwachen. Ein Schalter wird an der Lamellen Steuerung angeschlossen. Bei einem bestimmten Wert soll das ESP32 den Schalter über das MQTT/REST informieren, dass die Lamellen die Automatisierung der zentralen Steuerung übernehmen muss.

2 Studium

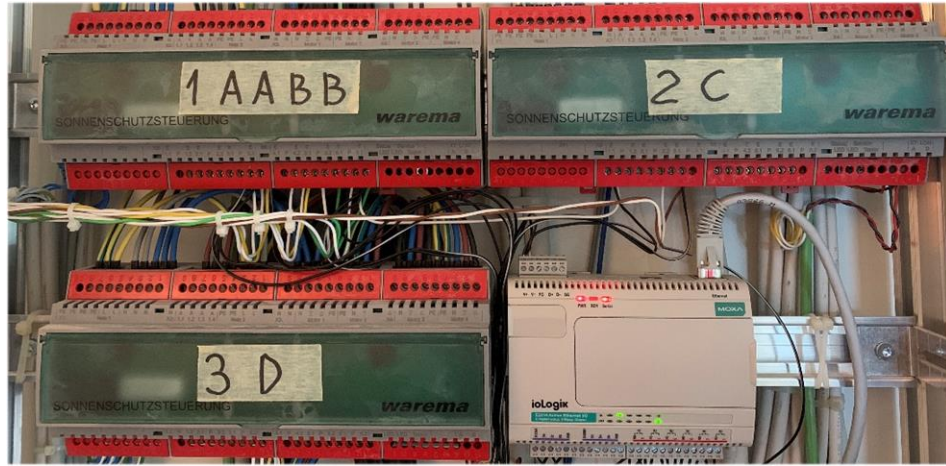


Bild: Die Steuerungen der Lamellen

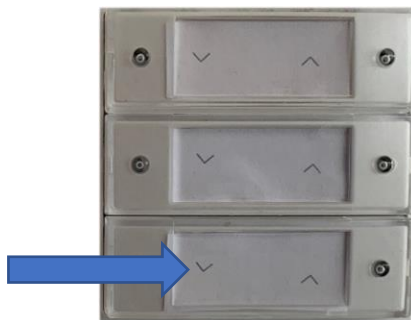


Bild: Die Taster

Der Bewohner hat keinen Zugriff auf die Programmierung der Steuerung.

Die Steuerungen in der Wohnung sind mit den Taster und die automatischen Zentralsteuerung verbunden. Die Taster senden die 30V Signalen zur Steuerung.

Um die Lamellen auf automatisch einzustellen, muss der Bewohner sie unten fahren, und für 10 Sekunden den Taster (Nach Unten) gedrückt halten.

Anleitung:

1. 5 Sekunden lang den Taster nach unten drücken
2. 75 Sekunden warten bis die Lamellen komplett ausgerollt
3. 15 Sekunden lang den Taster nach unten drücken, um die Lamellen automatisch einzustellen.

Signale Vorgänge: 0V → 5s = 30V → 75s = 0V → 15s = 30V

3 Projektplanung

3.1 Material

3.1.1 ESP32 Board mit einem TMP36 Sensor und einem Luxmeter



Ein "Maker Kit für ESP32" ist ein 32-Bit-Mikrocontroller mit integriertem Wi-Fi und Bluetooth. Der TMP36 Sensor erfasst die Temperaturwerte. Der Luxmeter erfasst die Helligkeit der Wohnung, um sicherzustellen, ob die Sonne direkt in die Wohnung scheint.

3.1.2 Raspberry Pi 4b als MQTT Broker



Der MQTT Broker ist der Vermittler zwischen das ESP32 und den zu simulierenden Tastern.

3.1.3 W&T Web-IO Digital 12x IN, 6xRelais Out



Das Web-IO von der Firma W&T wird die Taster zur Steuerung der Lamellen simuliert. In der Schweiz wird das Web-IO im professionellen Bereich eingesetzt. Das MQTT- und REST-Protokoll werden unterstützt.

3.2 Etappierung / Zeitplan

Das Projekt sollte in etwa 10 Stunden fertig gestellt werden können.

Beschreibung	Minuten	Komplexität
1. Projekt Einrichten		
1.1 ESP32-Board mit Sensoren einrichten	30	niedrig
1.2 Raspberry Pi mit MQTT Broker installieren	15	niedrig
1.3 Web-IO mit der Steuerung anbinden. (Verkabelung)	45	normal
2. Projekt Design und Modelling		
2.1 Kommunikation der Daten entwerfen	30	normal
3. Projekt Umsetzung		
3.1 Alle Geräten zusammen verbinden	420	hoch
4. Test		
4.1 Funktionstest	30	normal
4.2 Last-Test	30	normal
SUMME	600	-

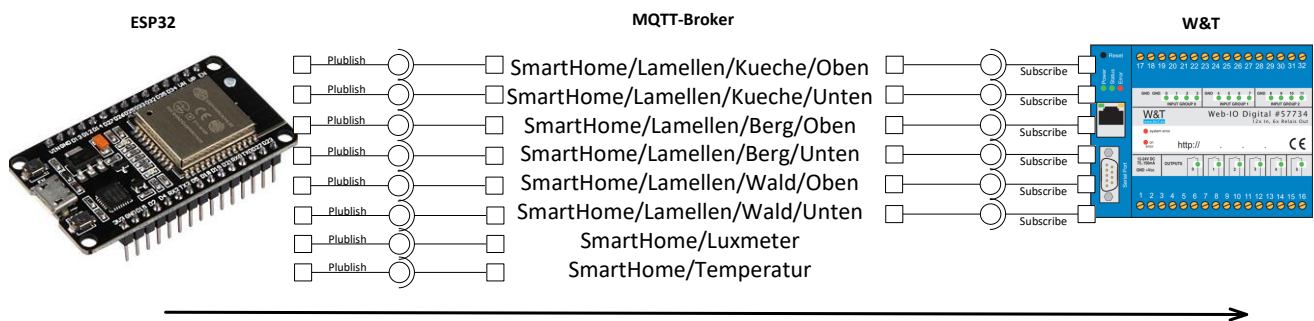
4 Umsetzung

4.1 Entwurf der Daten via MQTT

Der MQTT-Broker ist ein Vermittler.

Das ESP32 publiziert die Daten zum Broker.

Das W&T Gerät hört das Topic des MQTT-Brokers an. Wenn das Topic publiziert ist, führt es eine entsprechende Aktion aus.



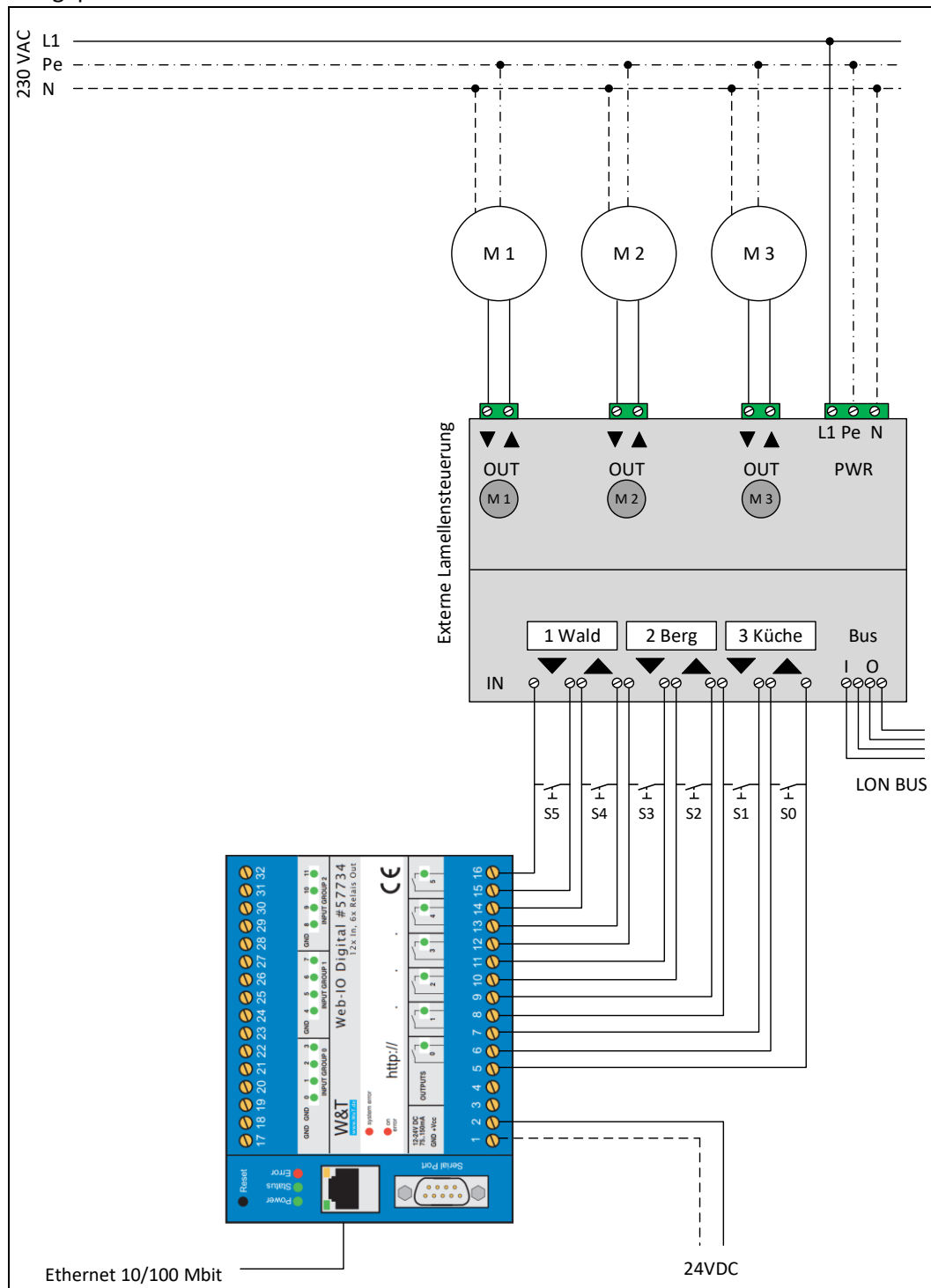
Für diese Projekt die bi-direktionale Datenkommunikation ist nicht notwendig, die Daten fließen von links nach rechts.

4.2 W&T 6xRelais Out

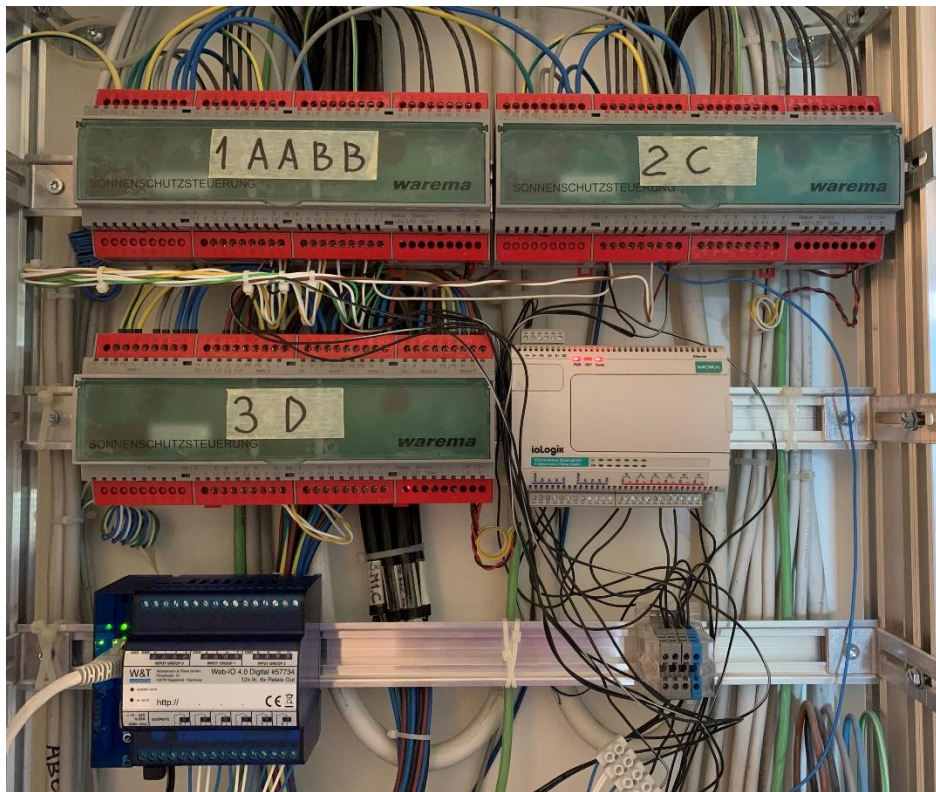
Auf die bereits installierten Lamellen-Steuerung hat der Bewohner keinen Zugriff. (Zentrale LON Steuerung).

Mit der Hilfe eines Elektrotechnik-Experten konnte ich mit dem W&T Web I/O die Wohnungs-Taster S0 bis S5 simulieren. Um die verschiedenen Stromkreise komplett zu entkoppeln, wurden Potentialfreie Kontakte verwendet.

Der geplante Aufbau sieht wie das untenstehende Bild:



Das Bild nach der fertigen Installation:



Im Bild oben befindet sich die originale Steuerung, unten Links das W&T Web I/O welches wir für die Taster Simulation einsetzen.

Die Liste der Aktionsfällen bei W&T Web I/O:

Die Erklärung folgt mit dem nächsten Bild.

0.0 Küche oben	Trigger	Löschen
0.1 Küche oben	Trigger	Löschen
1.0 Küche unten	Trigger	Löschen
1.1 Küche unten	Trigger	Löschen
2.0 Berg oben	Trigger	Löschen
2.1 Berg oben	Trigger	Löschen
3.0 Berg unten	Trigger	Löschen
3.1 Berg unten	Trigger	Löschen
4.0 Wald oben	Trigger	Löschen
4.1 Wald oben	Trigger	Löschen
5.0 Wald unten	Trigger	Löschen
5.1 Wald unten	Trigger	Löschen

Beispiel:

"Aktion 1.1 Küche unten"

Die erste Nummer bedeutet die Output Nr. 1.

Die Zweite Nummer ist den Zustand des Outputs "ON" (1 = "ON", 0 = "OFF").

Der Name sagt, wo die Lamellen sich befindet.

"unten" = Die Lamellen nach unten fahren, "oben" = Die Lamellen nach darauf öffnen.

Diese Aktion empfängt den Wert des Topics "SmartHome/Lamellen/Kueche/Unten".

Wenn beim MQTT-Broker ein MQTT-Topic "SmartHome/Lamellen/Kueche/Unten" den Wert "ON" publiziert wurde, soll es den Output 1 aktiviert werden (Taster drücken: Lamellen nach unten fahren). Der Output 1 wird deaktiviert nur dann von der Aktion "1.0 Küche unten", dort erwartet ein MQTT-Topic "SmartHome/Lamellen/Kueche/Unten" den Wert "OFF"(Taster loslassen).

1.1 Küche unten

Konfigurieren Sie hier die geplante Aktion.

Einstellungen	
Aktion:	<input checked="" type="checkbox"/> aktiviert i
Aktions Name:	1.1 Küche unten i
Auslöser:	<div>MQTT-Subscribe i</div> <div>Topic-Pfad: i</div> <div>SmartHome/Lamellen/Kueche/Unten</div> <div>Topic-Text / Payload:</div> <div>ON</div>
Aktion:	<div>Output schalten i</div> <div> <input checked="" type="radio"/> Output dieses Web-IO schalten <div> <div>mehrere Outputs i</div> <div>in Zustand:</div> <div> <input type="checkbox"/> Output 0 <input checked="" type="checkbox"/> Output 1 <div> <input type="radio"/> OFF <input checked="" type="radio"/> ON </div> <input type="checkbox"/> Output 2 <input type="checkbox"/> Output 3 <input type="checkbox"/> Output 4 <input type="checkbox"/> Output 5 </div> </div> <div> <input type="radio"/> Outputs eines anderen Web-IO schalten </div> </div>

4.3 MQTT Broker

4.3.1 Ein MQTT Broker auf dem Raspberry Pi installieren

```
sudo apt update  
sudo apt-get install -y mosquitto mosquitto-clients
```

-y = Der Broker soll automatisch nach dem Neustart wiederstarten.

4.3.2 Ein Zugang für MQTT Broker einrichten

```
sudo systemctl stop mosquitto.service  
sudo mosquitto_passwd -c /etc/mosquitto/passwd <user_name>
```

Danach das Password zwei Mals eingeben und die Konfigurationsdatei des Brokers ändern.

```
sudo nano /etc/mosquitto/mosquitto.conf
```

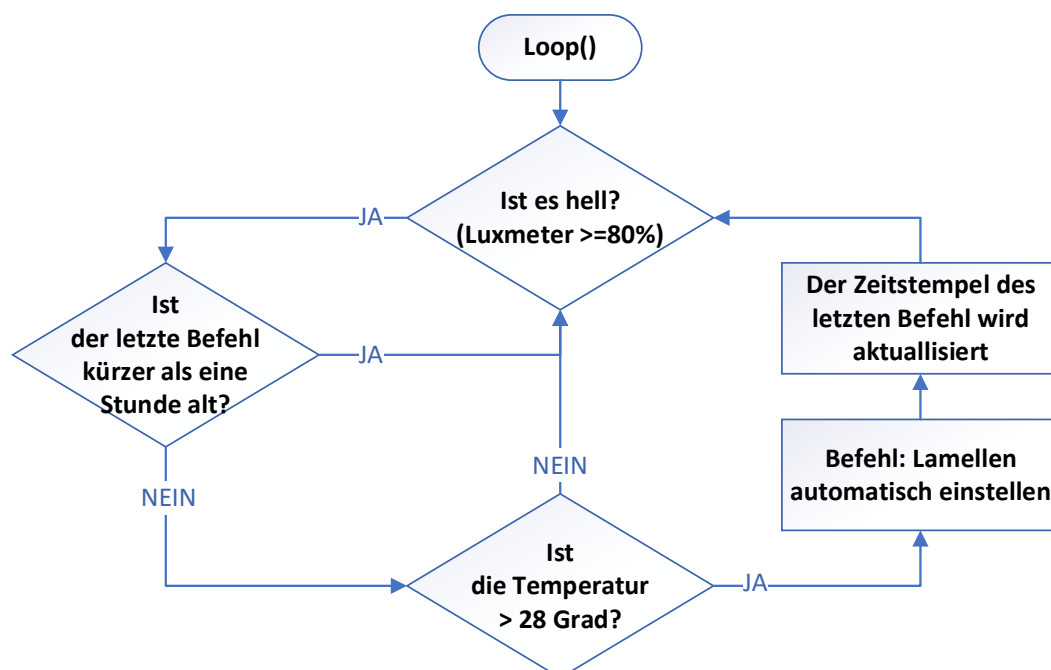
Füge folgenden Texte zum "mosquitto.conf" File hinzu.

```
password_file /etc/mosquitto/passwd  
allow_anonymous false
```

4.4 ESP32 Board

4.4.1 Business Logic

- Das ESP32 Bord soll in der Nähe der Fenster-Fron hingestellt werden.
- Es wird die Helligkeit vom Luxmeter gemessen (Sonneneinstrahlung in die Wohnung)
- Falls ja: Es wird weiter geprüft, ob die Business-Logik bereits innerhalb von einer Stunde die Lamellen automatisch eingeschalt hat oder nicht.
- Falls nein: Es wird weiter geprüft, ob die Temperatur (Temperatur Sensor) über 28 Grad misst.
- Nur dann wird ein Befehl zur automatischen Steuerung der Lamellen via MQTT gesendet.



Code: Business Logic

```
void loop() {  
    mqttClient.loop();  
    luxValue = readLUX();  
    tempValue = readTemp();  
  
    if(luxValue >= 80 && checkTime() && tempValue >= 28){  
        Serial.println("Set automatic ..");  
        if(!publishAuto()){  
            Serial.println("ERROR!! Fail by setting the horizontal blinds to automatic");  
        } else {  
            Serial.println("Set automatic succeeded");  
        }  
    }  
    printStatus();  
    delay(1000);  
}
```

4.4.2 Mit WLAN verbinden

Die Verbindung zu WLAN wurde die Bibliothek "*WiFi.h*" von dem Handbuch eingerichtet:
Ein statische IP-Adresse wurde deklariert.

Code: WLAN-Verbindung mit einer statischen IP-Adresse

```
#include <WiFi.h>  
void setup() {  
    IPAddress ip(192, 168, 0, 202);  
    IPAddress dns(192, 168, 0, 1);  
    IPAddress gateway(192, 168, 0, 1);  
    IPAddress subnet(255, 255, 255, 0);  
    WiFi.config(ip, dns, gateway, subnet);  
    WiFi.begin(SECRET_SSID, SECRET_PASS);  
}
```

4.4.3 Mit MQTT verbinden - Last will definieren

Für die Verbindung zum MQTT-Broker wurde eine MQTT-Bibliothek von Joel Gaehwiler heruntergeladen und verwendet. <https://github.com/256dpi/arduino-mqtt>

MQTT
by **Joel Gaehwiler**
MQTT library for Arduino This library bundles the lwmqtt client and adds a thin wrapper to get an Arduino like API.
[More info](#)
Version 2.5.0

Zum Schutz der Fehler während laufenden Befehles wurde die MQTT-Testamente (Will) vordefiniert. Ein Testament ist ein spezielles MQTT-Publish. Es wird publiziert, wenn das ESP32 die Verbindung zum Broker verliert.

Falls ESP32 keine Verbindung zum Broker hat, werden alle Relais-Kontakte getrennt (OFF).

Code: MQTT Verbindung und die Lastwille

```
#include <MQTT.h>
//MQTT
WiFiClient net;
MQTTClient mqttClient;

void setup() {
  mqttClient.begin(MQTT_SERVER, net);
  mqttClient.setKeepAlive(120);
  mqttClient.connect(MQTT_ID, MQTT_USERNAME, MQTT_PASSWORD, false);
  mqttClient.setCleanSession(true);
  mqttClient.setWill(KUECHE_OBEN, OFF, true, 1);
  mqttClient.setWill(KUECHE_UNTEN, OFF, true, 1);
  mqttClient.setWill(BERG_OBEN, OFF, true, 1);
  mqttClient.setWill(BERG_UNTEN, OFF, true, 1);
  mqttClient.setWill(WALD_OBEN, OFF, true, 1);
  mqttClient.setWill(WALD_UNTEN, OFF, true, 1);
}
```

4.4.4 Zeit Server einrichten

Es soll geprüft werden, ob die Lamellen bereits in innerhalb eine Stunde auf automatisch eingestellt geworden sind. Um das messen zu können, benötigt man einen Zeit-Server (NTP – **Net time Protocol**)

Zwei weitere "WiFiUdp.h" und "NTPClient.h" Bibliotheken für die Verbindung sind notwendig.

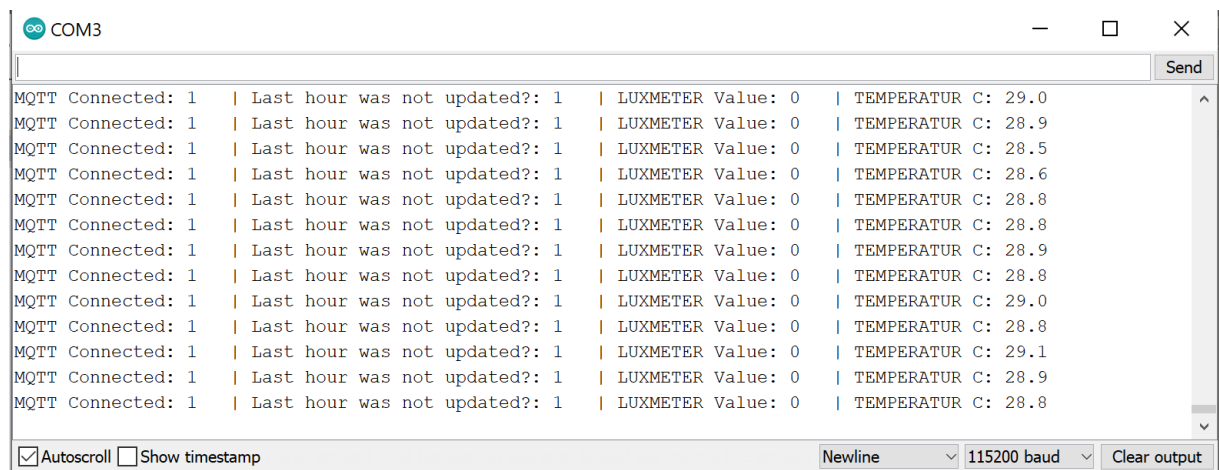
Code: NTP-Verbindung und Prüfen die Zeit des letzten Befehles

```
#include <NTPClient.h>
#include <WiFiUdp.h>
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "ch.pool.ntp.org");
int lastUpdate;
void setup() {
    timeClient.setTimeOffset(3600);
    timeClient.begin();
}
boolean checkTime() {
    if (timeClient.getHours() == 0, lastUpdate != 0) lastUpdate = 0; //reset
    if (lastUpdate < timeClient.getHours()) return true;
}
return false;
}
```

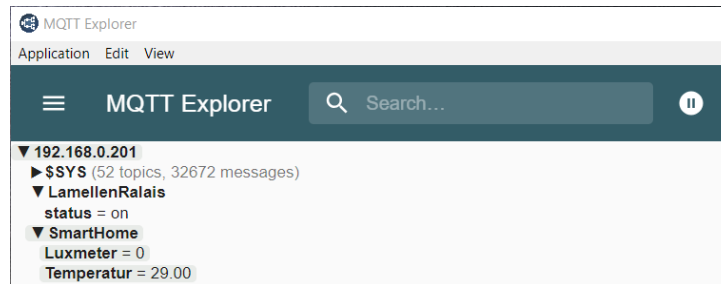
4.4.5 Status des ESP32

Für den Test sollen die Werte und der Status von dem ESP32 einfach abgelesen werden können. Eine Methode ist alle Minute zuständig für die Ausgabe sowohl via "Serial Monitor" als auch via MQTT-Broker.

Bei Serial Monitor werden die Status der MQTT-Verbindung, der Status des letzten Updates, der Luxmeter-Wert, und die Temperatur in Grad ausgegeben.



Bei MQTT-Broker werden nur der Luxmeter-Wert, und die Temperatur in Grad publiziert.



Code: Print Methode

```
void setup(){
    Serial.begin(115200);
}
void printStatus() {
    Serial.print("MQTT Connected: ");
    Serial.print(mqttClient.connected(),1);
    Serial.print(" | Last hour was not updated?: ");
    Serial.print(checkTime(),1);
    Serial.print(" | LUXMETER Value: ");
    Serial.print(luxValue,1);
    Serial.print(" | TEMPERATUR C: ");
    Serial.println(tempValue,1);
    mqttClient.publish(MQTT_LUX, String(luxValue));
    mqttClient.publish(MQTT_TEMP, String(tempValue));
}
```

5 Meilenstein und Zeitplan

Meilenstein

1. Projekt Einrichten

- 1.1 ESP32-Board mit Sensoren einrichten
- 1.2 Raspberry Pi mit MQTT Broker installieren
- 1.3 Web-IO mit der Steuerung anbinden. (Verkabelung)

2. Projekt Design und Modelling

- 2.1 Kommunikation der Daten entwerfen

3. Projekt Umsetzung

- 3.1 W&T 6xRelais Out
- 3.2 MQTT Broker
- 3.3 ESP32 Board

4. Test

- 4.1 Funktionstest
- 4.2 Last-Test

Zeitplan

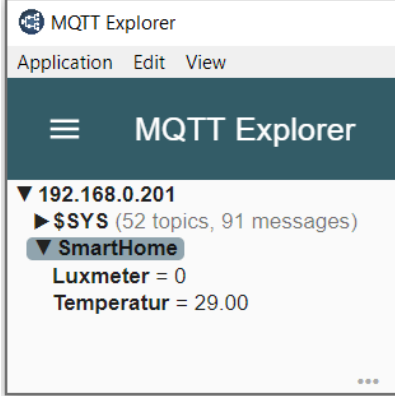
Meilenstein Nr.	Zeit (15 Minuten)																																															
	0	15	30	45	60	75	90	105	120	135	150	165	180	195	210	225	240	255	270	285	300	315	330	345	360	375	390	405	420	435	450	465	480	495	510	525	540	555	570	585	600							
1.1																																																
1.2																																																
1.3																																																
2.1																																																
3.1																																																
3.2																																																
3.3																																																
4.1																																																
4.2																																																

6 Testphase

6.1 Funktion Test

6.1.1 Broker Test

Send MQTT-Publish zum Broker

Vorbedingung	LAN-Verbindung MQTT-Broker ist eingeschaltet	
Vorgänge	1. Publiziert MQTT zum Topic "SmartHome/Luxmeter" mit dem Wert 0	
Akzeptierung	MQTT Explorer zeigt den Wert an.	
Bild	 <p>The screenshot shows the MQTT Explorer application window. It displays a tree view of MQTT topics. Under the 'SmartHome' topic, the 'Luxmeter' value is shown as 0 and the 'Temperatur' value is shown as 29.00.</p>	Mittels des Programm MQTT-Explorer kann man alle Werte im MQTT zeigen lassen. Ebenso auch ein MQTT zu dem bestimmten Topic publizieren.

6.1.2 W&T I/O Test

Send MQTT-Publish zum Broker

Vorbedingung	LAN-Verbindung MQTT-Broker ist eingeschaltet	
Vorgänge	1. Publiziert MQTT zum Topic " SmartHome/Lamellen/Kueche/Unten " mit dem Wert "ON"	
Akzeptierung	Die Lamellen wurden nach Unten gefahren.	

6.1.3 ESP32 und Hauptfunktionstest

Simulation der Temperatur: Mit einem Feuerzeug
Simulation der Helligkeit: Mit einer Taschenlampe

Vorbedingung	LAN-Verbindung MQTT-Broker ist eingeschaltet W&T I/O ist eingeschaltet Der Luxmeter-Wert grösser als 79% Die Temperatur ist grösser als 27 °C Grad
Vorgänge	1. Die Taschenlampe soll direkt auf dem Luxmeter scheinen 2. Das Feuerzeug anzünden und in der Näher den TMP36-Sensor halten 3. All Vorbedingungen sind erfüllt.
Akzeptierung	Die Lamellen müssen automatisch eingestellt werden.

6.2 Last Test

6.2.1 Alle Geräte über 24 Stunden laufen lassen.

Vorbedingung	LAN-Verbindung
Vorgänge	1. MQTT-Broker einschalten 2. ESP-32 am Fenster hinstellen und einschalten 3. W&T 6xRelais Out einschalten 4. 24 Stunden überwachen
Akzeptierung	Alle Geräte sollen fortlaufen. Keinen Software- oder Hardware-Fehler vorhanden.

6.2.2 Die Lamellen 5 Mals hintereinander automatisch schauen.

Vorbedingung	LAN-Verbindung Loop 5 Mals im Code für die automatische Schaltung der Lamellen.
Vorgänge	1. MQTT-Broker einschalten 3. W&T 6xRelais Out einschalten 4. Code ins ESP32 hochladen 5. ESP32 einschalten
Akzeptierung	Die Lamellen sollen 5 Mals automatisch eingestellt. Alle Geräte sollen fortlaufen. Keinen Software- oder Hardware-Fehler vorhanden.