Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

Projektbericht zum Thema:

Funksteckdosen per Telegrambot, mithilfe eines ESP8266-Moduls steuern

Vorgelegt von:

Mario Teklic
Am Tierpark 7
10315 Berlin

E-Mail: mario.teklic@htw-berlin.de
Matrikelnummer: 566426
Modul: Drahtlose Netzwerke
Fachsemester: 5
Studiengang: Angewandte Informatik

Abgabe: 17.07.2020

Lehrkraft: Prof. Dr. Alexander Huhn

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Durch die Automatisierung von elektronischen Geräten die alltäglich von uns benutzt werden, kann das Leben einfacher gestaltet werden. Im Zuge dessen, treten aber auch weitere nützliche Nebeneffekte auf. Beispielsweise kann Strom durch die Automatisierung von Stromverbrauchern in den eigenen vier Wänden eingespart und somit Geld gespart werden.

In diesem Projekt geht es darum, elektronische Geräte von unterwegs aus steuerbar zu machen. Realisiert werden, kann das von kleinen Mikrocontrollern. Ein Mikrocontroller ist der Grundbaustein für ein automatisierte Umgebung. Durch Module werden dem Mikrocontrollern weitere Fähigkeiten hinzugefügt. Sie können dann beispielweise per Funk, mit anderen Geräten im Haus oder in der Wohnung kommunizieren. Einige Mikrocontroller, haben außerdem ein WLAN-Modul verbaut. Durch dieses Modul sind dem Projekt schlussendlich, keine Grenzen mehr gesetzt. Durch das WLAN-Modul kann sich der Mikrocontroller mit dem Internet verbinden und der Mikrocontroller ist von unterwegs aus Beliebig steuerbar.

Ziel dieses Projekts ist es, Funksteckdosen per Telegrambot, von unterwegs aus steuern zu können.

2 Grundlagen

2.1 Komponenten

Es werden die einzelnen Komponenten aufgelistet, welche in diesem Projekt Anwendung finden.

2.1.1 ESP8266

Die nebenstehenden Abbildung 1, zeigt ein sogenanntes ESP8266 Modul. Dieses Modul wurde in diesem Projekt verwendet, da es günstig ist und die Rechenleistung für diese Zwecke genügt. Es ist ein kleiner Mikrocontroller, auf welchem die Firmware NodeMCU läuft.[1]

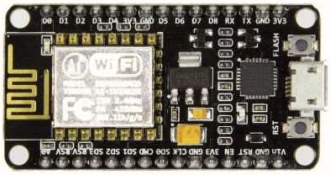

Abbildung 1 ESP8266

Das ESP8266 Modul ist mit einer WLAN Modul ausgestattet, was ermöglicht, dass eine Verbindung zum Internet aufgebaut werden kann. In der Tabelle 1 werden die wichtigsten Eckdaten des ESP8266 festgehalten.

Daten ESP8266	
Model	NodeMCU ESP8266
Wireless Standard	802.11 b/g/n
Arbeitsspeicher	64 kB
Datenspeicher	96 kB
Taktfrequenz	80 – 160 MHz

Tabelle 1 ESP8266 Daten

[1] nodemcu.

2.1.2 433 Mhz Empfänger- und Sendemodul

Abbildung 2 433 MHz Sender und Empfänger

Die Abbildung 2 zeigt auf der linken Seite den Empfänger und auf der rechten Seite den Sender.

Mit dem Empfängermodul lassen sich die Codes der Funksteckdosen auslesen. Mit dem Sendemodul können die vorher ausgelesenen Codes dann an die Funksteckdosen zum An- und Ausschalten gesendet werden. Sie senden und empfangen auf einer 433Mhz Frequenz.

Daten Empfängermodul	
Empfangsfrequenz	433 Mhz
Empfindlichkeit	-105dB

Tabelle 2 Daten Empfängermodul

Daten Sendemodul	
Sendefrequenz	433 Mhz
Sendeleistung	10mW
Übertragungsrate	4KB/s

Tabelle 3 Daten Sendemodul

2.1.3 Telegrambot

Telegrambots sind Applikationen, welche innerhalb eines Telegramchats laufen. Dem Bot werden Anweisungen mitgeteilt, indem Ihnen bestimmte Nachrichten geschrieben werden.[2]

Dabei sind die Möglichkeiten vielfältig. Sie können als eine Art Postbote agieren, mit anderen Services wie bspw. von IMDB, Youtube oder GitHub Bot kommunizieren oder die Wettervorhersage machen.

Für dieses Projekt reicht es aber, dass der Bot anschreibbar ist und dass es eine Bibliothek für das ESP8266 Modul gibt, die es ermöglicht eine Verbindung zum Bot aufzubauen.[3]

2.1.4 Funksteckdose

Abbildung 3 Renkforce Funksteckdosenset

Dies sind für dieses Projekt ideale Funksteckdosen, da diese nicht selbstlernend sind. Der Code, welcher zum An- und Ausschalten genutzt wird, ist an der Fernbedienung und an den Steckdosen einstellbar. Somit ändert sich der Code nicht, sobald die Steckdosen vom Strom entfernt sind. Ansonsten müssten jedes Mal die korrekten Codes ausgelesen und im Programmcode ersetzt werden.

Das Steckdosenset sendet und empfängt bei 433 Mhz die Ein- und Ausschaltsignale.[4]

[2] Telegram.
[3] Gianbacchio.
[4] renkforce.

3 Durchführung

3.1 Erstellung des Telegrambots[5]

Wie eingangs bereits erwähnt, kann ein Telegrambot zahlreiche Sachen erledigen. Für die Zwecke dieses Projekts reicht es aber, wenn der Bot dauerhaft verfügbar und anschreibbar ist.

Zunächst wird der User namens „BotFather“ in Telegram benötigt.

Der BotFather ist zuständig für die Bot-Erstellung. Mithilfe des Kommandos */newbot,* wird ein neuer Bot erstellt.

Als nächstes wird der Name des Bots ausgewählt. Es kann irgendein Name gewählt werden. Er wird in diesem Beispiel SteckdosenBot genannt.

Nun muss ein Username für den Bot ausgesucht werden. Dieser Username muss einzigartig sein und mit „bot“ enden. Des Weiteren darf kein Bindestrich benutzt werden. Der Username des Bots ist in diesem Projekt ESP8266_SB_bot.

Der BotFather schickt nun zwei wichtige Sachen zu:

- Den Link, unter welchem der Bot anschreibbar ist, welcher in etwa so aussehen wird: t.me/ESP8266_SB_bot

- Und den Bot Token, welcher später benötigt wird, um das ESP8266 Modul mit dem Bot zu verbinden. Der Token sieht in etwas folgendermaßen aus: 1151053176:AAFYv3IdiwcIZXOofGsPHeaTCgjh_GdnHso

Diese Informationen sollten in einer Textdatei festgehalten werden, da sie später benötigt werden.

3.2 Vorbereiten der Arduino IDE

Um externe Bibliotheken zu finden, die für das ESP8266 Modul programmiert wurden, wird zunächst eine Voreinstellung benötigt.

3.2.1 Bibliotheken

In der Arduino IDE: Datei > Voreinstellungen.

Unter Zusätzliche Boardverwalter URLs wird folgende URL hinzugefügt:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

OK um die Einstellung zu speichern.

Nun können die zusätzlichen Bibliotheken über die Arduino IDE hinzugefügt werden.

[5] Telegram.

Sketch > Bibliothek einbinden > Bibliotheken verwalten….

Benutzt bzw. benötigt werden folgende Bibliotheken:

- TelegramBot[6] - Verbindung zum Bot mit dem ESP8266 herstellen
- ESP8266[7] - WLAN-Verbindung
- rc-switch[8] - Senden und Empfangen der Funksteckdosencodes

Der IDE sollten diese drei Bibliotheken hinzugefügt werden. Falls eine nicht gefunden wird, kann diese ggf. auch per Zip-Datei installiert werden. Für weitere Informationen, in dem jeweiligen GitHub-Repository nachschauen. Die GitHub-Links sind im Quellenverzeichnis aufgeführt.

3.2.2 Weitere Einstellungen

Des weiterem muss unter Werkzeuge > Board, das richtige Board ausgewählt werden und unter Werkzeug > Port, der korrekte COM Port.

Zunächst wird das ESP8266 Modul per Mini-USB an den PC an.

Wird genau das ESP8266 Modul aus der Abbildung 1 verwendet, so muss nun unter Board „NodeMCU 1.0 (ESP-12e Module)“ ausgewählt werden.

3.3 Konstruktion der Hardware

3.3.1 Empfänger anschließen[9]

Es sind 3 Pins des Empfängermoduls zu belegen, bzw. mit dem ESP8266 zu verbinden.

GND: Zunächst wird der Ground Pin mit dem Grund (GND) Pin verbunden.

DATA: Dann wird der rechte DATA Pin mit dem D3 Pin des ESP8266 verbunden.

VCC: Zum Schluss wird der VCC Pin wird mit dem VIN Pin des ESP8266 Moduls verbunden. Hier wird das Empfängermodul mit der Spannung versorgt.

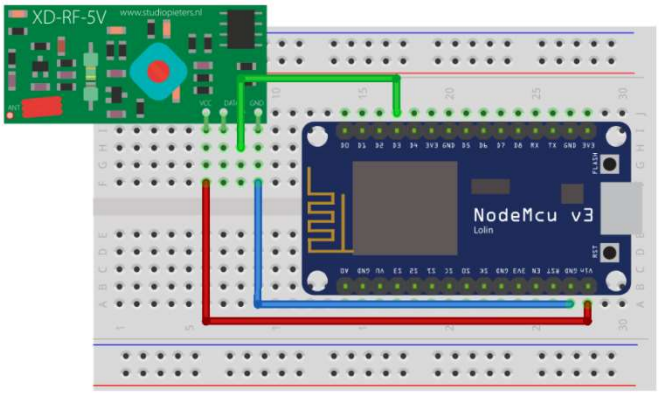

Abbildung 4 433 Mhz Empfängeranschlussplan

[6] Casa Jasmina.
[7] esp8266.
[8] sui77.
[9] Daniel Ziegler.

3.3.2 Funksteckdosencodes auslesen

Dadurch, dass im Schritt 3.2 die nötigen Bibliotheken installiert wurden, stehen nun auch einige Beispiele für diese Bibliotheken direkt zur Verfügung.
Zum Auslesen der Funksteckdosencodes, wird ein Beispielcode der rc-switch Bibliothek verwendet.
In der Arduino IDE: Datei > Beispiele > rc-switch > ReceiveDemo_Advanced
Das einzige was an dem Code noch verändert werden muss, ist:
mySwitch.enableReceive(0); -> mySwitch.enableReceive(D3);
Da der Data Pin des 433 Mhz Empfängermoduls, an den D3 Pin des ESP8266 verbunden wurde, muss eben diese Einstellung hier noch verändert werden.

Der Sketch wird kompiliert und auf das ESP8266 Modul hochgeladen.

Nun wird eine der Tasten von der Funksteckdosenfernbedienung gedrückt.
Als Beispiel werden nur die An- und Ausknöpfe für die 1. Steckdose (siehe Abbildung 5) genutzt.
Zum Auslesen wird der Serielle Monitor benutzt.
Unter Werkzeuge > Serieller Monitor, lässt sich dieser öffnen.
Zum korrekten empfangen der Codes sollte die Fernbedienung direkt auf den Empfänger (mit sehr kurzem Abstand) gerichtet sein.
Zunächst wird die linke Taste gedrückt.

Abbildung 5 Funksteckdosenfernbedienung

Auf dem Seriellen Monitor erscheinen nun die Codes, die der Empfänger empfangen hat.
Es erscheinen: Decimal, Tri-State, PulseLength, Protocol und Raw data.

DECIMAL: 1381717 (24BIT) BINARY: 000101010000101010101010101 TRI-STATE: 0FFF0FFFFFFF PULSELENGTH: 401 MICROSECONDS PROTOCOL: 1
RAW DATA:
12434,393,1208,396,1209,394,1209,1194,409,394,1208,1195,409,392,1212,1194,409,392,1212,1194,409,391,1213,1194,410,390,1213,1194,410,390,1213,1194,410,390,1214,1192,411,390,1214,1192,412,389,1214,1190,414,1190,413,1190,414,

Zum An- und Ausschalten werden lediglich die Dezimalzahl und die Bitanzahl benötigt:
Decimal: 1381717 (24Bit)
Das ist der Code zum Anschalten der 1. Funksteckdose.

Dieser Schritt muss für alle weiteren Tasten bzw. Steckdosen wiederholt werden. Es sind alle Dezimalwerte zu notieren.

Die Tabelle 4 ist eine Beispielliste aller Codes, welche später in der Implementierung des Arduino Sketches verwendet wird.

Steckdose	An	Aus	Bit
A	1381717	1381716	24
B	1394005	1394004	24
C	1397077	1397076	24

Tabelle 4 Beispielliste der Funksteckdosencodes

Hieraus ist abzulesen, dass sich der Code jeweils nur um 1 verändert. Bei Selbstlernenden Steckdosen wird heutzutage meist ein Rolling-Code verwendet, welcher diesen Teil des Projekts sehr erschweren würde, da dieser eine Art Authentifizierung zwischen den beiden Komponenten darstellt.[10]

Der Empfänger kann nun vom Breadboard bzw. ESP8266 abgenommen werden. Er wird nicht mehr benötigt.

3.3.3 Sender anschließen

Auch beim Sender sind wieder 3 Pins zum belegen.

GND: Zunächst wird der Ground Pin mit dem Grund (GND) Pin verbunden.

DATA: Dann wird der DATA Pin mit dem D3 Pin des ESP8266 verbunden.

VCC: Zum Schluss wird der VCC Pin wird mit dem VIN Pin des ESP8266 Moduls verbunden. Hier wird das Sendemodul mit der Spannung versorgt.

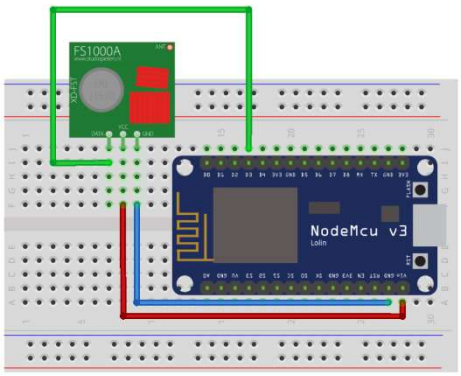

Abbildung 6 433 Mhz Senderanschlussplan

3.4 Programmierung des ESP8266 Moduls

Nun kommt es zur Programmierung des ESP8266 Moduls inklusive WLAN-Verbindung, Telegrambot-Verbindung und Funkcodesendung.

Zunächst wird ein neuer Arduino Sketch benötigt.

3.4.1 WLAN [Programmierung]

Um die Zugangsdaten zum WLAN nicht direkt im Programmcode stehen zu haben, wird ein neuer Header angelegt: arduino_secrets.h.

In dem Header werden die Zugangsdaten des WLAN-Netzwerkes definiert.

```
#define SECRET_SSID "WIFI-NAME"
#define SECRET_PASS "WIFI-PW"
```

Entsprechend müssen der Namen und das Passwort des WLAN-Netzwerkes ersetzt werden.

Nun kommt es zum eigentlichen Sketch.

```
#include <Arduino.h>
#include "X:Pfad\zur\arduino_secrets.h"
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
```

```
//WIFI
const char* ssid  = SECRET_SSID;
const char* password = SECRET_PASS;
WiFiClientSecure net_ssl;
```

[10] itwissen.info.

In der setup()-Methode wird folgendes geschrieben:

```
1.  Serial.begin(115200);
2.
3.  while (!Serial); //Warten, bis der Monitor bereit ist
4.
5.  WiFi.begin(ssid, password);
6.
7.  while (WiFi.status() != WL_CONNECTED) {
8.    delay(500);
9.  }
10.
11. delay(1500);
12.
13. net_ssl.setInsecure(); //Um später den Telegrambot korrekt nutzen zu können
```

Da nun die Internetverbindung steht, kann mit dem Telegrambot fortgefahren werden.

3.4.2 Telegrambot [Programmierung]

Zunächst muss der arduino_secrets.h noch der Telegrambot-Token zugefügt werden.

Folgende Zeile wird dem Header zugefügt:

```
1.  #define SECRET_BOT_TOKEN "1151053176:AAFYv3IdiwcIZXOofGsPHeaTCgjh_GdnHso"
```

Der Token ist der in 3.1 festgehaltene Schlüssel, welcher vom BotFather geschickt wurde.

Begonnen wird mit dem Importieren des Headers und der Erstellung der Telegrambot-Objekts.

```
1.  #include <TelegramBot.h>
2.
3.  //TELEGRAM
4.  const char BotToken[] = SECRET_BOT_TOKEN;
5.  TelegramBot bot (BotToken, net_ssl);
```

Der Bottoken kommt aus der ardunio_secrets.h Datei.

Anschließend wird der Bot in der setup()-Methode gestartet, nachdem sich das ESP8266 Modul mit dem WLAN erfolgreich verbunden hat.

```
1.    bot.begin();
```

Der Bot ist nun Einsatzbereit.

In der loop()-Methode wird ständig geprüft, ob dem Bot eine neue Nachricht zugeschickt wurde.

```
1.    message m = bot.getUpdates();
2.    if (m.chat_id != 0) {
3.      bot.sendMessage(m.chat_id, "Ok.");
4.      commands(m.text);
5.    }
```

Sobald eine Nachricht reingekommen ist, wird dem Benutzer ein „Ok.“ zugesendet, um Bescheid zu geben, dass die Nachricht ankam.

Die commands()-Methode wird im Punkt 3.4.3 erläutert.

3.4.3 Funkcodesendung [Programmierung]

Nun steht die Internetverbindung und Nachrichten, die dem Bot geschrieben wurden, können auslesen werden.

Es fehlen noch, die Benutzung des 433 Mhz Senders und wie mit den Nachrichten umgegangen wird. In diesem Unterpunkt wird ersteres programmiert.

Um den Sender benutzen zu können, muss die Bibliothek (rc-switch) importiert sein (siehe 3.2.1).

```
1.  #include <RCSwitch.h>
```

Erstellung des RCSwitch-Objekt:

```
1.  //RCSWITCH
2.  RCSwitch mySwitch = RCSwitch();
```

Um den Überblick zu behalten, wird für jede Steckdose eine Konstante angelegt:

```
1.  //Socket Consts
2.  const int A = 1;
3.  const int B = 2;
4.  const int C = 3;
```

Nun werden die in 3.3.2 festgehaltenen Funksteckdosencodes (siehe Tabelle 4) verwendet.

```
1.  //[0] == on, [1] == off
2.  int codes_A[] = {1381717, 1381716};
3.  int codes_B[] = {1394005, 1394004};
4.  int codes_C[] = {1397077, 1397076};
```

Der erste Eintrag ist jeweils der Code zum Einschalten, der zweite der zum Ausschalten.
A, B, C steht jeweils für eine Steckdose.

In diesem Fall war der Bitwert 24. Ebenfalls aus der Tabelle 4 entnommen.

```
1.  //Bit
2.  byte bit_mode = 24;
```

Nun wird eine Methode implementiert, in der eine Steckdose an- oder ausgeschaltet wird.

```
1.  /*Sockets:
2.  1 == A
3.  2 == B
4.  3 == C
5.  */
6.  void doSwitch(bool on, int socket){
7.    int on_off_code;
```

```
8.
9.    if(on){
10.     on_off_code = 0;
11.   }else{
12.     on_off_code = 1;
13.   }
14.
15.   if(socket == A){
16.     mySwitch.send(codes_A[on_off_code], bit_mode);
17.     delay(1000);
18.   }else if(socket == B){
19.     mySwitch.send(codes_B[on_off_code], bit_mode);
20.     delay(1000);
21.   }else if(socket == C){
22.     mySwitch.send(codes_C[on_off_code], bit_mode);
23.     delay(1000);
24.   }else{
25.     Serial.println("No socket found.");
26.   }
27. }
```

Die Booleanvariable „on“ gibt an, ob der Nutzer die Steckdose an oder ausschalten möchte.
True = Steckdose an, False = Steckdose aus

Mit der Integervariale „socket“ wird übermittelt, welche der Steckdosen angesteuert werden soll.

In der Integervariable on_off_code wird der Dezimalwert gespeichert, welcher gesendet werden soll. Falls die Steckdose angeschaltet werden soll, d.h. „on“ war true, wird der erste Wert (0) aus den Steckdosencodesarray genommen. Falls „on“ false war, wird der zweite Wert (1) aus dem Array genommen.

In dem nachfolgenden if-else Statement wird nun geschaut, welche Steckdose angefunkt werden soll.

Der eigentliche Befehl des Sendens besteht aus dem Dezimalcode und dem Bitwert.
Den Bitwert (das zweite Argument beim Senden), wird aus der Klassenvariable „bit_mode“ genommen, welche wir zuvor aus der Tabelle 4 entnommen haben.

3.4.4 Kommandosteuerung [Programmierung]

In 3.4.2, bzw. im loop des Sketches wird, falls eine Nachricht eingegangen ist, die commands()-Methode aufgerufen und die eingehende Nachricht als Argument übergeben.

```
1.  //Commands
2.  void commands(String s) {
3.
4.    if (s.equals("A on")) {
5.     doSwitch(true, A);
6.    } else if (s.equals("B on")) {
7.      doSwitch(true, B);
8.    } else if (s.equals("C on")) {
9.      doSwitch(true, C);
10.   } else if (s.equals("A off")) {
11.     doSwitch(false, A);
12.   } else if (s.equals("B off")) {
13.     doSwitch(false, B);
14.   } else if (s.equals("C off")) {
15.     doSwitch(false, C);
16.   } else if (s.equals("All on")) {
```

```
17.     doSwitchAll(true);
18.   } else if (s.equals("All off")) {
19.     doSwitchAll(false);
20.   } else {
21.     printer("Command not known.");
22.   }
23. }
```

Hier können die Kommandos ausgewertet werden und falls bekannt, weitergeleitet werden.

Auffällig ist, dass eine weitere Methode beim Kommando „All on“ bzw. „All off“ aufgerufen wird.

```
1.  void doSwitchAll(bool on){
2.    doSwitch(on, A);
3.    delay(1000);
4.    doSwitch(on, B);
5.    delay(1000);
6.    doSwitch(on, C);
7.    delay(1000);
8.  }
```

Die doSwitchAll()-Methode ist da, um die Steuerung zu vereinfachen, falls alle Steckdosen an- oder ausgeschaltet werden sollen. Da alle und nicht eine Steckdose speziell umgestellt werden sollen, reicht hier die Information, ob die Steckdose an- oder ausgeschaltet werden soll.

3.4.5 Weiteres [Programmierung]

```
1.   void printer(String text){
2.     Serial.println(text);
3.     //oled.display.println(text)..
4.   }
```

Beispielsweise wird in der commands()-Methode die printer()-Methode aufgerufen.

Um im Sketch unnötigen Code zu vermeiden, kann in der Methode die Nachricht auf mehreren Medien ausgeben werden. Beispielsweise im Seriellen Monitor und, falls vorhanden und angebracht, auf einem OLED Display am ESP8266 Modul.

3.5 Steuerung der Funksteckdosen per Telegrambot

Der in 3.4 programmierte Sketch wird nun auf das ESP8266 Modul geladen und hat der Sender wird gemäß 3.3.3 angeschlossen.

Das ESP8266 Modul und die Funksteckdosen werden an den Strom angeschlossen.

In 3.1 wurde der Link zum Chat des eigenen Bots, vom BotFather zugeschickt.
Dem Bot können nun die in 3.4.4 festgelegten Kommandos zum Schalten der Steckdosen, per Telegram übermittelt werden.

Hinweis 1: Hindernisse wie bspw. Gegenstände können die Funkreichweite beeinträchtigen.

Hinweis 2: Weiter ist das ESP8266 Modul nicht allzu schnell. Außerdem sind relativ viele delays eingebaut, um den Sender nicht zu überlasten. Nachdem ein Kommando abgesendet wurde, kann es bis zu 10 Sekunden dauern, bis der Funkcode tatsächlich ausgesendet wurden ist.

5 Fazit

Werden die Chancen und Möglichkeiten betrachtet, die diese Konstellation bietet, bietet dieses Projekt viel Spielraum in verschiedenste Weiterentwicklungsrichtungen.

Beispielsweise könnte eine App auf dem Smartphone, erkennen wann sich jemand zuhause befindet und wann nicht. Wenn sich niemand zuhause befindet, sollen automatisch alle Geräte vom Strom genommen werden.

Außerdem könnte getrackt werden, wie viel Strom letztendlich einspart wird.

In einer Zeit, in der Strom immer teurer wird und gleichzeitig der Umwelt etwas Gutes getan wird, in dem Strom spart wird, ist viel Wert auf das eigene Verhalten zu legen. Mittels Automatisierungen wie das in diesem Projekt vorgestellte Zusammenspiel verschiedener Module, wird dies für jeden leichter.

Dieses Projekt bietet eine solide Grundlage für genau diese Aspekte. Es ist ein solides und kostengünstiges Grundgerüst, für verschiedenste Anwendungen.

Literaturverzeichnis

Casa Jasmina: TelegramBot-Library. Host a Telegram Bot on your Arduino, and interact with it from your favourite chat application. Online verfügbar unter https://github.com/CasaJasmina/TelegramBot-Library, zuletzt geprüft am 18.06.2020.

Daniel Ziegler: 433MHz Funksteckdosen aus dem Baumarkt mit dem ESP8266 steuern. Online verfügbar unter https://daniel-ziegler.com/arduino/esp/mikrocontroller/2017/12/30/Funksteckdose-esp8266/, zuletzt geprüft am 18.06.2020.

esp8266: esp. ESP8266 core for Arduino. Online verfügbar unter https://github.com/esp8266/Arduino, zuletzt geprüft am 18.06.2020.

Gianbacchio: ESP8266-TeleramBot. Telegram Bot Library for ESP8266 on Arduino IDE. Online verfügbar unter https://github.com/Gianbacchio/ESP8266-TelegramBot, zuletzt geprüft am 17.06.2020.

itwissen.info: Rolling-Code. Online verfügbar unter https://www.itwissen.info/Rolling-Code-rolling-code.html, zuletzt geprüft am 19.06.2020.

nodemcu: NodeMCU. Online verfügbar unter https://github.com/nodemcu/nodemcu-firmware, zuletzt geprüft am 17.06.2020.

renkforce: Funk-Schalter-Set. Online verfügbar unter https://asset.conrad.com/media10/add/160267/c1/-/gl/001208454ML03/bedienungsanleitung-1208454-renkforce-steckdosen-set-4teilig-innenbereich-2000-w.pdf, zuletzt geprüft am 18.06.2020.

sui77: rc-switch. Online verfügbar unter https://github.com/sui77/rc-switch, zuletzt geprüft am 18.06.2020.

Telegram: Bots: An introduction for developers. Online verfügbar unter https://core.telegram.org/bots, zuletzt geprüft am 17.06.2020.

Telegram: Creating a new bot. Online verfügbar unter https://core.telegram.org/bots#:~:text=Creating%20a%20new%20bot,mentions%20and%20t.me%20links., zuletzt geprüft am 18.06.2020.