Alexa dimmt und steuert eine Schreibtischlampe

Hallo und willkommen zu einem weiteren Teil der Reihe rund um Alexa. Im ersten Teil haben wir uns dem Thema Sprachassistent genähert und anhand eines einfachen Beispiels die Funktionsweise der Sprachassistentin Alexa erklärt. Im heutigen Teil werden wir statt einer Lavalampe eine LED-Schreibtischlampe steuern. Dies hört sich erst einmal im Prinzip ziemlich gleich und austauschbar an, jedoch unterscheidet sich heutiges Projekt in drei wichtigen hinzugefügten Punkten von Teil 1. Diese sind:

1. Implementierung eines „Captive Portals“ für die Eingabe von WLAN-Verbindungsdaten.
2. Zusätzlicher „Push-Button“ zur manuellen Bedienung der Lampe „An/Aus Funktion“.
3. Lampe ist durch einen Alexa Sprachbefehl zwischen 1-99% dimmbar.

Das Captive Portal ist eine Webseite, die der ESP32 zur Konfiguration der WLAN-Einstellungen bereitstellt, falls diese nicht konfiguriert sind oder die konfigurierten Verbindungsdaten ungültig sind. Damit können die WLAN-Verbindungsdaten zur Laufzeit nachkonfiguriert werden, und müssen nicht mehr fest im Code programmiert sein. Die Schreibtischlampe muss so aufgebaut sein, dass das aktive Leuchtelement bzw. das LED-Panel mit einer maximalen Niedergleichspannung von ca. 20 Volt betrieben wird und NICHT direkt mit Netzspannung. Die Schreibtischlampe selbst hängt also selbst an einem externen, schutzisolierten Niederspannungstransformator, der selbst in keiner Weise für unseren Umbau verändert werden darf!

Auch dieses Projekt ist NUR für Kleinspannun gs LED-Lampen mit einer Betriebsgleichspannung bis zu 20 Volt geeignet! Bei Netzspannung besteht Lebensgefahr!

Da es auch mit den vollen 230 Volt betriebenen LED-Schreibtischlampenvarianten im Handel gibt, (sog. Hoch Volt LED-Lampen) überprüfen Sie bitte sorgfältig, dass Sie eine geeignete LED-Lampe für dieses Projekt besitzen. Auch darf die Stromstärke der geschalteten Lampe nicht die maximale schaltbare Stromstärke des eigesetzten MOSFET Moduls von 5 Ampere nicht übersteigen. Für unser heutiges Projekt benötigen wir folgende Hardware:

Anzahl	Bezeichnung	Anmerkung
2	ESP32 Node MCU	
1	DC-DC – Step Down Converter	
1	MOS Modul	
1	Taster	
1	Kleinspannungs LED - Schreibtischlampe	Mit Kleinspannungstransformator < 20 Volt

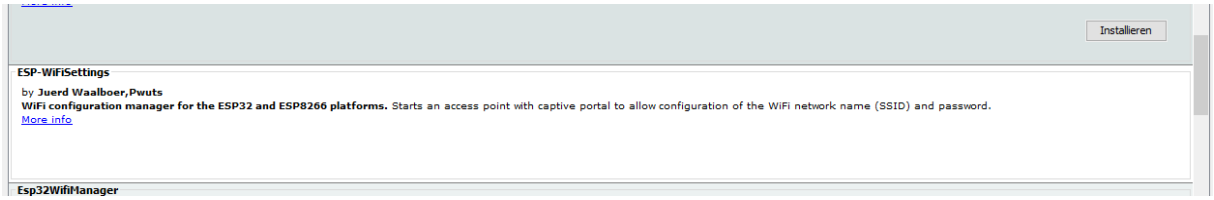
Die Hardware für dieses Projekt kann, bis auf die Kleinspannungs- LED–Schreibtisch Lampe, komplett bei AZ-Delivery bezogen werden. Die Arduino IWE muss für das

Projekt neben den ESP32 Bibliotheken um zwei weitere Bibliotheken erweitert werden. Doch zunächst binden Sie für den eingesetzten ESP32 die Board Bibliothek ein. Sehen Sie hierzu im eBook zum ESP32 nach.

Es werden folgende zwei Bibliotheken für das Projekt benötigt:

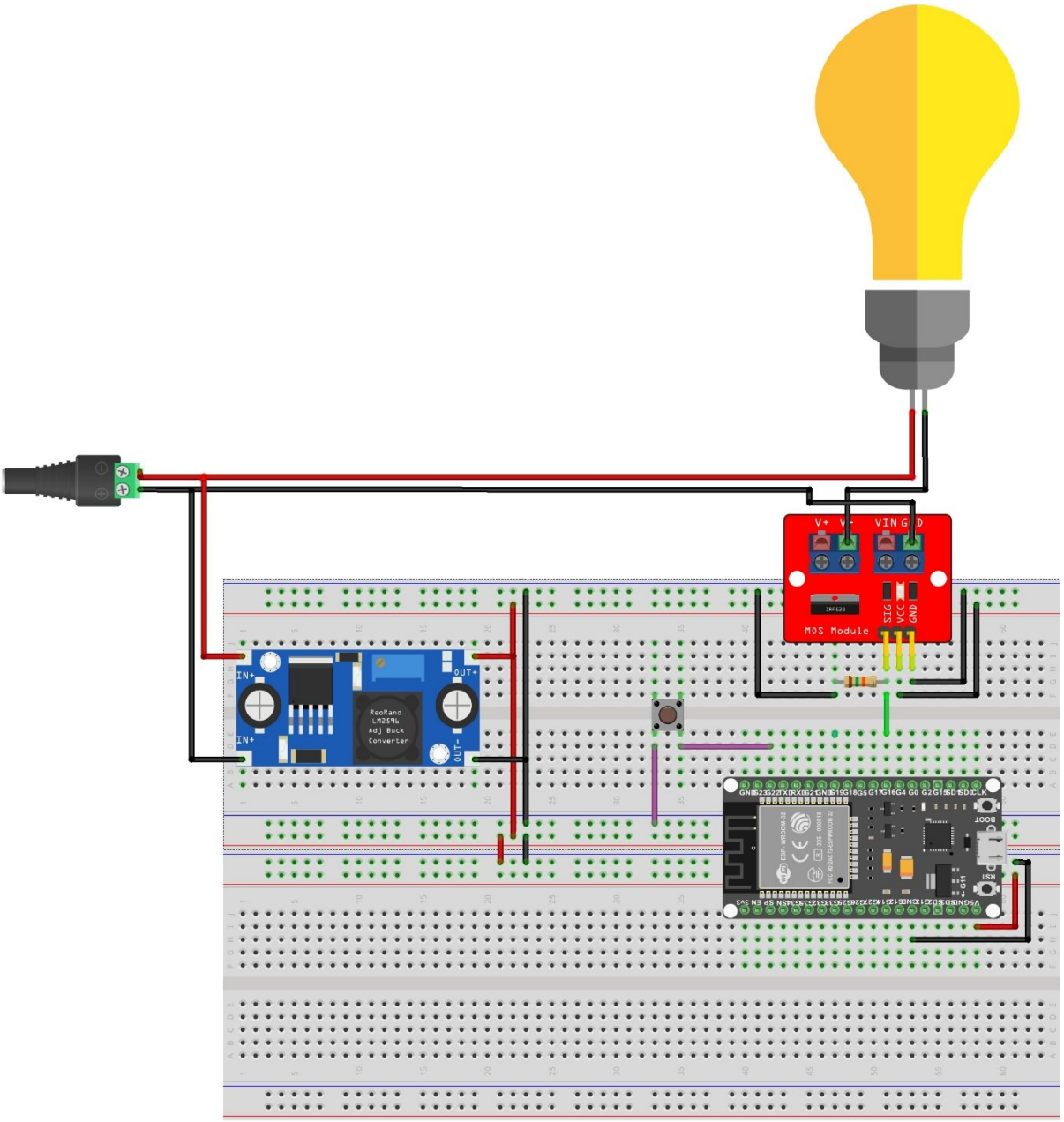
1. ESPAlexa
2. ESP-WIFISettings

Die Bibliotheken beiden können über die o.g. bereitgestellten Links heruntergeladen werden. Die im Projekt genutzte Bibliothek ESP-WIFISettings kann alternativ auch über den Bibliotheksverwalter eingebunden werden. Diese kann unter dem Namen ESP-WiFiSettings gefunden werden



Eine gute Beschreibung, wie Bibliotheken in die Arduino IDE eigebunden werden können, finden Sie im Blog „Arduino IDE - Programmieren für Einsteiger - Teil 1“ im Abschnitt Bibliotheken Verwaltung. Zur Funktion und Funktionsmöglichkeiten der Bibliothek "EspAlexa" finden sich Informationen in Englisch hier. Bevor wir uns an das Aufbauen der eigentlichen Schaltung machen können, müssen wir den DC-DC Konverter noch auf 5 Volt Ausgangsspannung einstellen. Dazu benötigen wir neben dem bereits vorhandenem Kleinschutzspannungstransformators der Schreibtischlampe ein Voltmeter. Dieses schließen wir an den Ausgang des DC-DC Konverters an das Voltmeter an und stellen den Spannungsmessbereich auf Gleichspannung. Wir drehen an dem blauen Potentiometer mit einem Plastikschraubendreher so lange, bis die Ausgangsspannung exakt 5.00 Volt beträgt.

Jetzt ist auch zu erkennen, warum wir KEINESFALLS eine Lavalampe mit Netzspannung verwenden dürfen. Wir beziehen die Betriebsspannung von 5 Volt der kompletten Elektronik über einen nicht galvanisch getrennten DC-DC Konverter, den wir von der Kleinschutzspannungstransformator versorgen lassen. Eine Versorgung mit Netzspannung würde diesen zerstören und die gesamte Schaltung unter Netzspannung setzen! Sobald wir den DC-DC auf 5 Volt Ausgangsspannung eingestellt haben, und die korrekte Funktion mit dem Voltmeter überprüft haben, kann mit dem Verdrahten der Komponenten begonnen werden. Diese müssen wie folgt verdrahtet werden:



Nach korrekter Verdrahtung kann nun folgender Code auf den ESP hochgeladen werden:

```
/*  Alexa LED Desk LAMP
 *
 *  Zweck:
 *  Steuerung einer LED Niederspannungsschreibtischlampe bis 12 Volt mit Hilfe von Alexa
 *
 *  Tobias Kuch 2020 -- tobias.kuch@googlemail.com
 *
 * Lizensiert under GPL 3.0
 *
 *
 */
```

```
#include <Espalexa.h>
#include <SPIFFS.h>
#include <WiFi.h>
#include <WebServer.h> //if you get an error here please update to ESP32 Arduino core 1.0.0
#include <WiFiSettings.h>

const uint8_t LED_PIN = 23;  // Status LED
const uint8_t LIGHT_MANUAL_SW = 22; // Switch
const String Alexa_DeviceName = "Schreibtischlampe";

// PWM Configuration Stuff for ESP 32
const int freq = 5000;
const int PWMChannel = 0;
const int resolution = 8;
const int PWMOutputPin = 16;  // 16 corresponds to GPIO16


#define LED_ON  LOW
#define LED_OFF HIGH


//callback functions
void firstLightChanged(uint8_t brightness);

//boolean wifiConnected = false;

EspalexaDevice* lamp1;

Espalexa espalexa;
#ifdef ARDUINO_ARCH_ESP32
WebServer server(80);
#else
ESP8266WebServer server(80);
#endif

bool Sw_old_State = true;

// Bugfix: Every second time, no Success Connection is Made with Access Point
WiFiEventId_t eventID = WiFi.onEvent([](WiFiEvent_t event, WiFiEventInfo_t info) {
   Serial.print("WiFi lost connection. Reason: ");
   Serial.println(info.disconnected.reason);
   if (info.disconnected.reason == 202) {
   Serial.println("Connection failed, REBOOT/SLEEP!");
    esp_sleep_enable_timer_wakeup(10);
    esp_deep_sleep_start();
   delay(100);
   }
  }, WiFiEvent_t::SYSTEM_EVENT_STA_DISCONNECTED);

void setup() {
Serial.begin(115200);
SPIFFS.begin(true);  // Will format on the first run after failing to mount
```

```
pinMode(LIGHT_MANUAL_SW, INPUT_PULLUP);
pinMode(LED_PIN, OUTPUT);
pinMode(PWMOutputPin, OUTPUT);
ledcSetup(PWMChannel, freq, resolution);
ledcAttachPin(PWMOutputPin, PWMChannel);
// Set custom callback functions
WiFiSettings.onSuccess  = []() {
digitalWrite(LED_PIN, LED_ON); // Turn LED on
};
WiFiSettings.onFailure  = []() {
digitalWrite(LED_PIN, LED_OFF); // Turn LED off
};
WiFiSettings.onWaitLoop = []() {
digitalWrite(LED_PIN, !digitalRead(LED_PIN)); // Toggle LED
return 500; // Delay next function call by 500ms
};
// Callback functions do not have to be lambda's, e.g.
// WiFiSettings.onPortalWaitLoop = blink;
// Define custom settings saved by WifiSettings
// These will return the default if nothing was set before
String host = WiFiSettings.string( "server_host", "default.example.org");
int    port = WiFiSettings.integer("server_port", 80);
// Connect to WiFi with a timeout of 30 seconds
// Launches the portal if the connection failed
WiFiSettings.connect(true, 40);
Serial.println("Initializing Alexa API..");
server.on("/", HTTP_GET, []()
 {
 server.send(200, "text/plain", "Alexa Schreibtischlampe . Tobias Kuch 2020
tobias.kuch@googlemail.com");
 });
server.onNotFound([]()
 {
 if (!espalexa.handleAlexaApiCall(server.uri(),server.arg(0))) //if you don't know the URI, ask
espalexa whether it is an Alexa control request
   {
   //whatever you want to do with 404s
   server.send(404, "text/plain", "404 - Not found");
   }
 });

// Define Alexa Devices
lamp1 = new EspalexaDevice(Alexa_DeviceName, firstLightChanged,
EspalexaDeviceType::dimmable); //Dimmable device, optional 4th parameter is beginning state
(here fully off)
espalexa.addDevice(lamp1);
espalexa.begin(&server); //give espalexa a pointer to your server object so it can use your server
instead of creating its own
Serial.println("Done.");
}


void Query_Manual_Controls()
```

```
{
bool Sw = digitalRead(LIGHT_MANUAL_SW);
if ((Sw == LOW) and (Sw_old_State == false))
 {
 Sw_old_State = true;
 delay(100);
 if (lamp1->getValue())
  {
  Serial.println("Manual OFF");
  lamp1->setValue(0);
  ledcWrite(PWMChannel, lamp1->getValue());
  }
  else
  {
  Serial.println("Manual ON");
  lamp1->setValue(255);
  ledcWrite(PWMChannel, lamp1->getValue());
  }
 } else if (Sw == HIGH)
 {
  Sw_old_State = false;
 }
}

void loop() {
 espalexa.loop();
 Query_Manual_Controls();
 delay(1);
}

// Callback functions
void firstLightChanged(EspalexaDevice* d) {
   Serial.print("Lamp changed to ");
   if (d->getValue() == 255) {
    Serial.println("ON");
   }
   else if (d->getValue() == 0) {
    Serial.println("OFF");
   }
   else {
    Serial.print("DIM "); Serial.println(d->getValue());
   }
   ledcWrite(PWMChannel, lamp1->getValue());
}
```

Die Funktion ist folgende: Über den DC-DC Konverter versorgt sich der ESP32mit Strom. Er ist somit dauernd aktiv und verbindet sich nach hochladen des Codes in das WLAN-Netz, dessen Verbindungsdaten über das Captive Portal hinterlegt wurden. Nach erfolgreicher Anmeldung des ESP's an das WLAN bitten Sie Alexa nun, mit dem Befehl: "Alexa, suche Smarthomegeräte" nach neuen Smarthomegeräten zu suchen.

Alexa wird nun nach ca. 1 Minute vermelden das ein neues Smarthomegerät gefunden wurde. Dies ist unser ESP! Nun kann der Pin D0 des ESPs und damit unsere Schreibtischlampe mit dem Befehl "Alexa, Schreibtischlampe ein" eingeschaltet und mit dem Befehl "Alexa, Schreibtischlampe aus" ausgeschaltet werden. Zusätzlich kann die Helligkeit der Lampe im Bereich von 1-99% gesteuert werden. Der zugehörige Sprachbefehl zum dimmen lautet: “Alexa, Schreibtischlampe 50 Prozent ein" oder auch verkürzt “Alexa, Schreibtischlampe 50 Prozent"

Ich wünsche Ihnen viel Spaß beim Bauen und beim Verwenden der Schreibtischlampe mit Alexa.