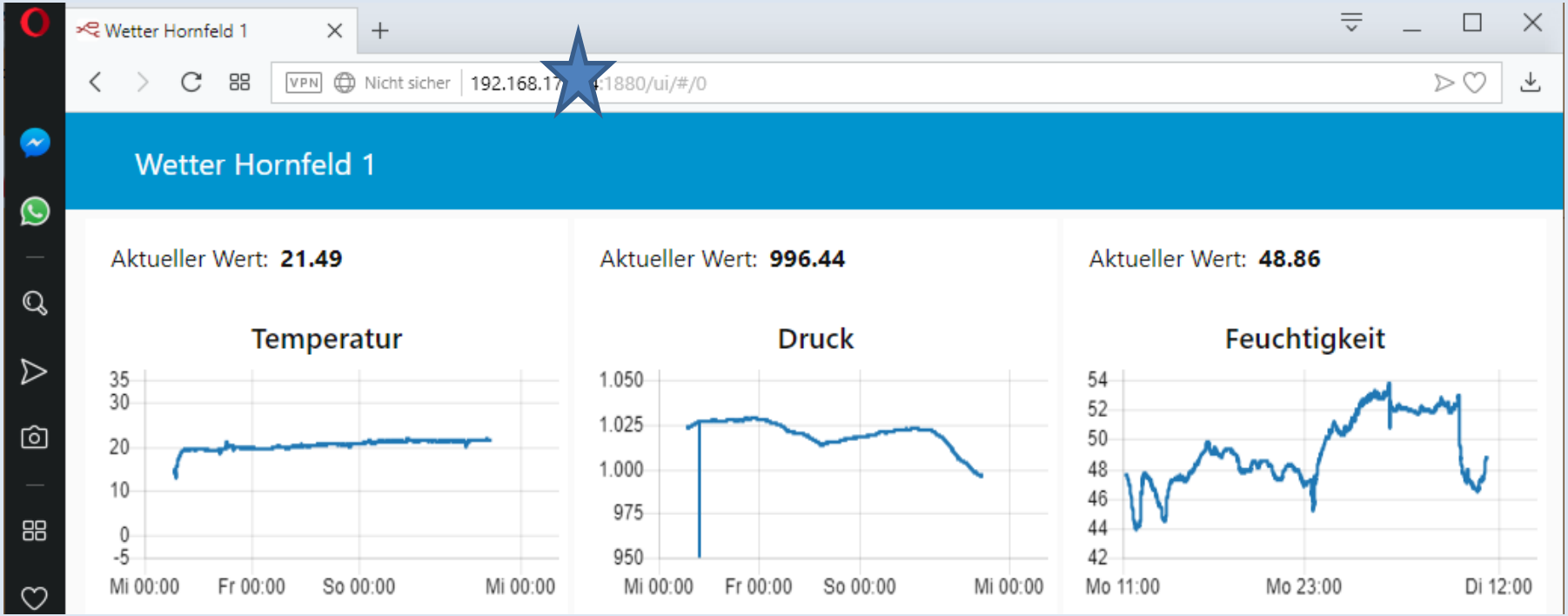
IoT & Node-RED & RPI

H39@email.de

Wetterdaten im PC-Browser



Raspberry Pi

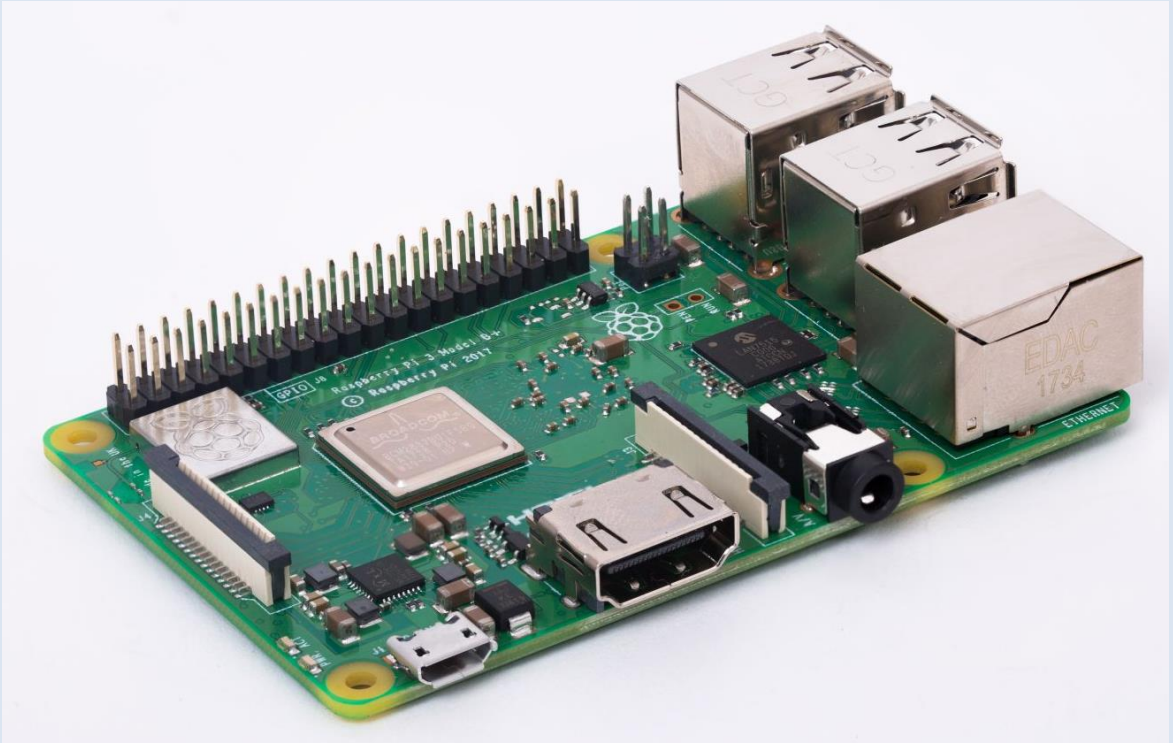
Raspberry Pi 3 Model B+

oder

Raspberry Pi 3 Model B

oder

Raspberry Pi 3 Model A+



Quelle: *https://www.raspberrypi.org/products/*

Wetterstation

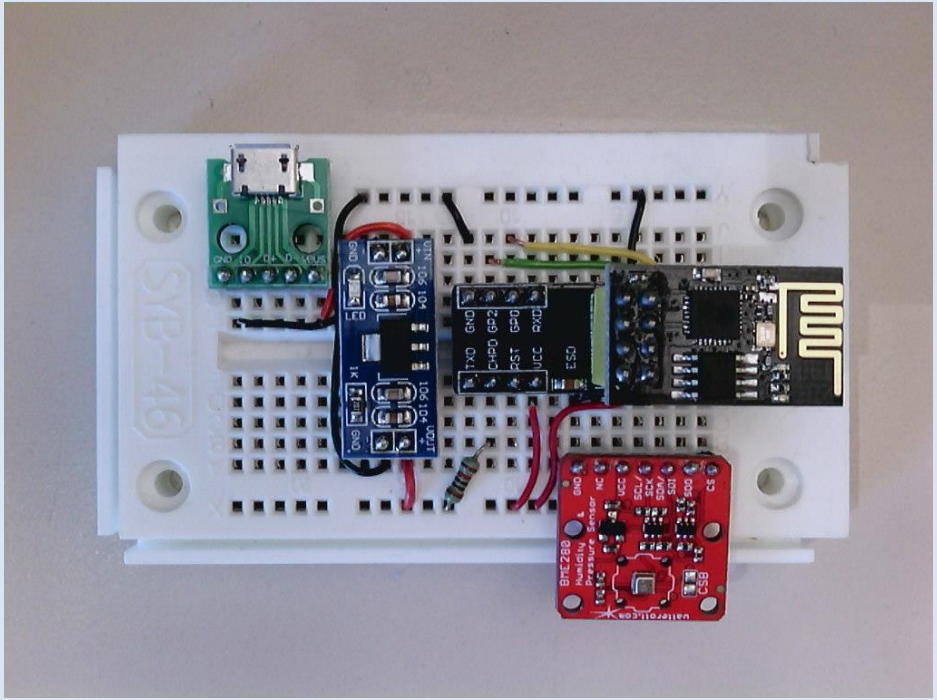
Mikrokontroller:
ESP8266 ESP-01 Serial Port WIFI Transceiver Wireless Modul

ESP8266 Breadboard-Adapter:
ESP-01 Adapter Breakout Board Platine ESP01

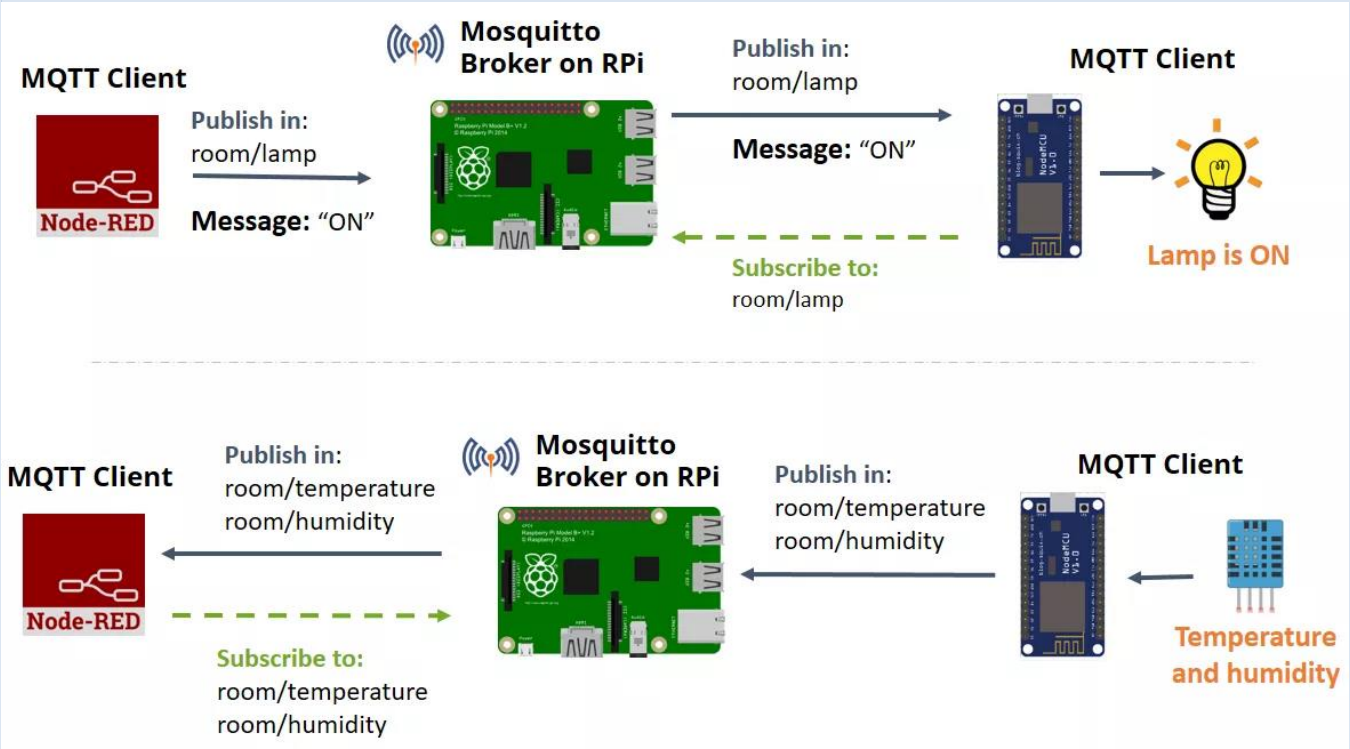
Spannungsregler Step Down Power Modul:
AMS1117 3,3 V

Mikro-USB Anschluss:
Micro USB Breakout Modul Board Platine

Wettersensor:
BME280-BREAKOUT (watterott.com)



MQTT, Node-RED, ESP8266



Quelle *https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2017/08/MQTT-ESP8266-publish-and-subscribe-Node-RED.png?ssl=1*

Projektbausteine

	Das Projekt hat als Basis die Informationen von: *https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/*
Raspberry Pi (RPI)	Raspberry Pi mit Betriebssystem Raspbian einrichten.
Node-RED	Raspberry Pi mit Node-RED konfigurieren. Als Dienst einrichten.
Broker Mosquitto	Raspberry Pi mit Mosquitto konfigurieren. Als Dienst einrichten.
Betriebsart	Node-RED und Mosquitto laufen 24h auf dem RPI.
ESP8266 ESP-01	ESP8266 Sketch anpassen und flashen.
Wetterstation	BME280-Sensor & Zusatzkomponenten aufbauen.
Node-RED flow	Einen Flow, der die Messdaten erfasst, und auf einem Dashboard (Webinterface) anzeigt, erstellen.

Raspberry Pi Raspbian

NOOBS	Raspbian Betriebssystem herunterladen. Empfolener Download als zip-Datei.
Version 3.0.0 Release 16.11.2018	*https://downloads.raspberrypi.org/NOOBS_latest*
	zip-Datei entpacken. Ordner „NOOBS_v3_0_0“ bereit stellen.
SD-Karte vorbereiten	SD Card Formatter herunterladen und installieren.
	https://www.sdcard.org/downloads/formatter_4/eula_windows/index.html
	SD-Karte mit „SD Card Formatter“ formatieren.
NOOBS auf SD Karte	Alle Ordner & Dateien aus „NOOBS_v3_0_0“ auf die SD-Karte kopieren.
RPI	Die SD-Karte in den RPI einstecken, Schritte der Installation durchführen.
	Raspbian einrichten (…)

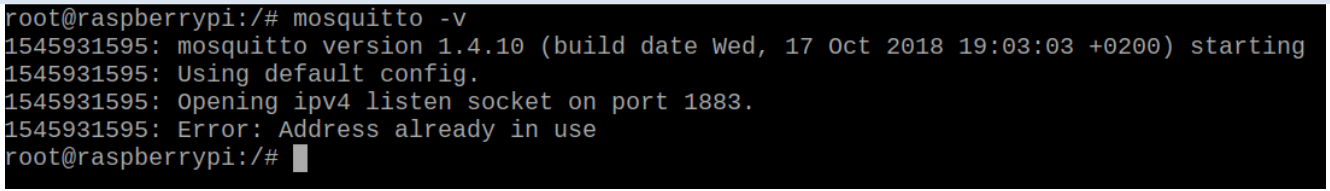
Node-RED

Raspberry Pi mit Raspbian GNU/Linux 9	*Die NOOBS-Version vom November 2018 enthält bereits ein vorinstalliertes Node-RED. Leider ohne „Manage palette“ und „Dashbord Nodes“.*
„Manage palette“ hinzufügen	Terminal: $*sudo apt-get install npm && sudo npm i -g npm* **Achtung, das $ oder #-Zeichen nicht miteingeben!**
Automatischen Start einrichten	Terminal: $*sudo systemctl enable nodered.service*
Node-RED starten	Terminal: $*node-red-star*t (bei automatischem Start nicht erforderlich)
Node-RED beenden	Terminal: $*node-red-stop*
Node-RED auf RPI	*Browser: http://127.0.0.1:1880*
IP vom RPI?	Terminal*: $hostname –I; ergibt bei mir „192.168.178.xxx“*
Node-RED von PC	*Browser: http://192.168.178.xxx:1880*
Node-RED Dashboard nachinstallieren	*Manage Palette -> Install -> node-red-dashboard*

MQTT Broker

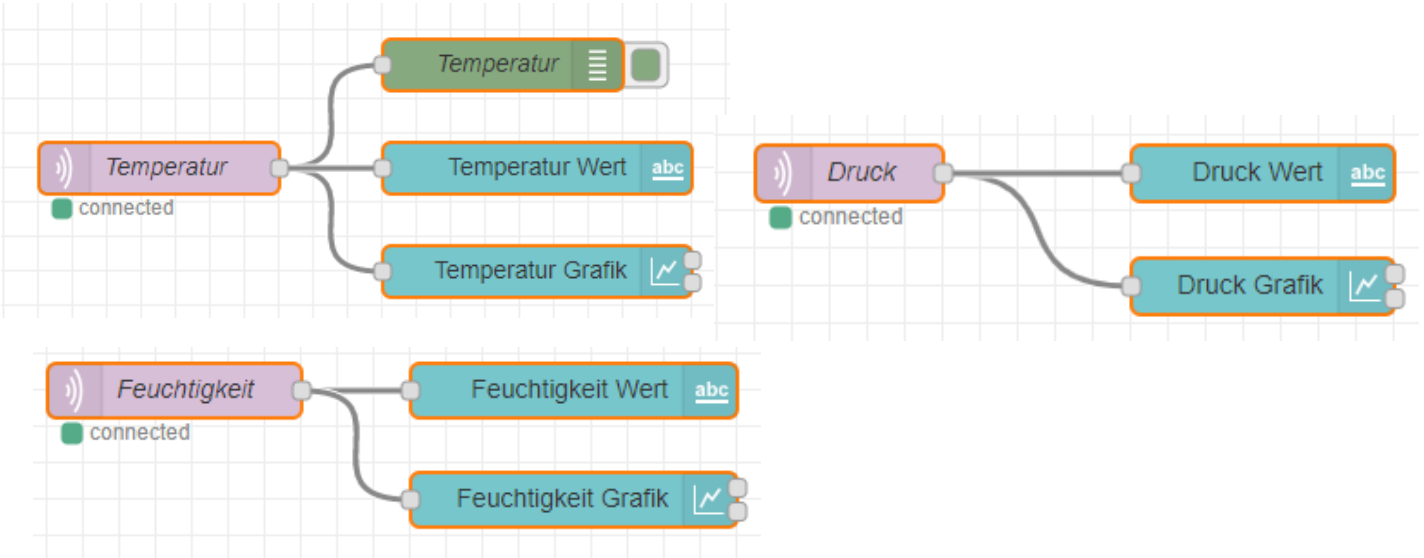
MQTT	**M**essage **Q**ueue **T**elemetry **T**ransport
	Das Nachrichten-Protokoll MQTT wurde für die Kommunikation von Geräten zu Geräten entwickelt.
	Grundlange für die Kommunikation bildet ein sogenannter Broker.
	Ein Broker ist eine Server-Anwendung (z.B. auf RPI).
	Ein Gerät kann sich nun mit dem Broker verbinden und Nachrichten Publischen (veröffentlichen), also an den MQTT-Broker, unter Nennung von Topics (Bezeichner), senden.
	Ein anderes Gerät (hier: node-RED) kann dann Topics Subscriben (abonnieren), und bekommt dann eine Nachricht vom MQTT-Broker zugestellt.
Information	*https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/*

Mosquitto RPI-broker

	Mosquitto ist ein MQTT Broker für den Raspberry Pi
Anleitungen von	https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/ https://randomnerdtutorials.com/testing-mosquitto-broker-and-client-on-raspbbery-pi/
Terminal	*$sudo su* *#sudo apt update* *#sudo apt install -y mosquitto mosquitto-clients* *#sudo systemctl enable mosquitto.service* *#mosquitto –v*
	
	Erwartete Ausgabe, d. h. Mosquitto läuft
IP?	Terminal*: $hostname –I; ergibt hier „192.168.178.xxx“.* Wird für den ESP8266-Sketch gebraucht, daher notieren.

Node-RED Flow

Vorhanden	Raspberry Pi mit Node-RED und Mosquitto konfiguriert.
Aufgabe von Node-RED	Node-RED & Mosquitto laufen 24h auf einem Raspberry Pi.
Node-RED öffnen	RPI-Browser: http://127.0.0.1:1880 PC-Browser: http://192.168.178.xxx:1880
Flow ESP8266	



ESP8266 I

Sketch	Der ESP8266 muss mit einem Sketch geflasht werden, der WLAN , MQTT und die Sensordaten vom BME280 verarbeitet.
PubSubClient Library	Erzeugt einen Client auf dem ESP8266, um Publish/Subscribe Messages mit einem MQTT Server zu ermöglichen.
Download	*https://github.com/knolleary/pubsubclient/archive/master.zip*
BME280 Sensor Libraries	Weiterhin die Libraries zum Einbinden des BME280-Sensors.
Download	*https://github.com/adafruit/Adafruit_BME280_Library* *https://github.com/adafruit/Adafruit_Sensor*
	Alle Libraries entpacken und in den Arduino „libraries“ Ordner kopieren
Information	*https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/*

ESP8266 II

Sketch	Ich habe einen Sketch vom „randomnerdtutorials“ angepasst. Den angepassten Sketch hier unterladen:
Sketch herunterladen	https://c.web.de/@3343227392989625 15/UHEzho6uQIixymcOjNzr3A
Sketch editieren	„ssid“ und „password“ eintragen. „IP“ vom MQTT-Server eintragen.

```
24  // Change the credentials below, so your ESP8266 connects to your router
25  const char* ssid = "Insert your WLAN ssid";
26  const char* password = "Insert your WLAN password";
27
28  // Change the variable to your Raspberry Pi IP address, so it connects to your MQTT broker
29  const char* mqtt_server = "Insert the IP of your MQTT-Server";
```

Abschnitt Setup()	Wire.begin(0, 2) einfügen (bei ESP8266 ESP-01).
	Sketch auf „ESP826“ flashen.
Information	*https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/*

BME280-Breakout von watterott.com

www.watterott.com	BME280-Breakout (Luftfeuchtigkeits-, Druck & Temperatursensor)
	Der BME280 ist einer der neuesten Luftfeuchtigkeits-, Druck- und Temperatursensoren von Bosch mit einem digitalen I2C und einem SPI Interface. Auf dem Breakout befinden sich ein Spannungsregler und ein Pegelwandler für die I2C/SPI Schnittstelle, daher kann der Sensor von 3V - 5,5V betrieben werden.
Features	 • Humidity sensor • Pressure sensor Pressure range 300 … 1100 hPa • Temperatur Sensor Operating range Operational -40°C - +85°C
Weitere Infos	github.com/watterott/BME280-Breakout
Quelle:	https://www.watterott.com/de/BME280-Breakout-Luftfeuchtigkeits-Druck-Tempertursensor

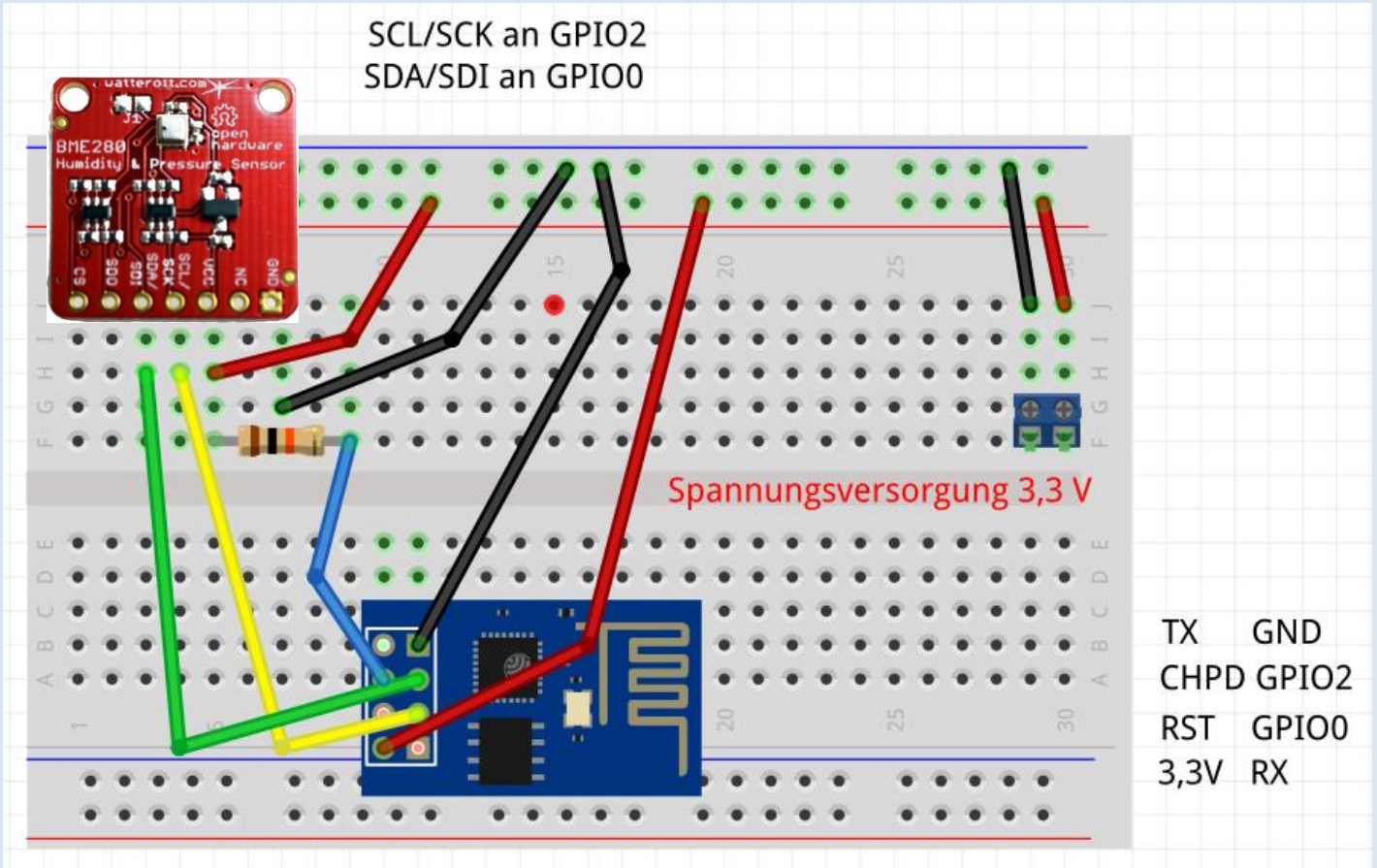
BME Libraries installieren

	Bevor der BME280 mit der Arduino-IDE programmiert werden kann, müssen 2 Libraries heruntergeladen, entpackt und installiert werden.
Arduino Library and Examples	https://github.com/adafruit/Adafruit_BME280_Library https://github.com/adafruit/Adafruit_Sensor
Achtung	Das installieren der „Adafruit_BME280_Library“ reicht nicht aus. Zusätzlich muss noch die „Adafruit_Sensor“ Library installiert werden.
Empfehlung manuelle Installation	Finde den Speicherort der Arduino-Installation heraus. Um Probleme mit Schreibrechten zu vermeiden, ist es ratsam die Arduino-IDE unter „C:\Users\Public“ zu speichern. Hier sind stets Schreibrechte vorhanden. Der Pfad zum Library-Ordner lautet dann: *„C:\Users\Public\Programme\Arduino-1.8.5\libraries“*
Installation	Beide Libraries entpacken und in den „libraries-Ordner“ der Arduino-IDE kopieren.
	Die Arduino-DIE neu starten.
Link „Documentation on learn.watterott.com“:	http://learn.watterott.com/sensors/bme280/

I2C-Betriebsart

Wertvolle Tipps in	https://learn.sparkfun.com/tutorials/i2c		
	https://tronixstuff.com/2010/10/20/tutorial-arduino-and-the-i2c-bus/		
I2C	Die Idee ist, dass über 2 Leitungen, SDA und SCL genannt, mehrere Geräte kommunizieren können. Daher auch I2C-Bus.		
	Dazu wird z.B. ein Temperatursensor in einem „Breakout Board“ integriert, das I2C bereit stellt, erkennbar an den Kontakten SDA und SCL.		
Für I2C sind festgelegt:	Board	SDA (data)	SCL (clock)
	Arduino UNO	A4	A5
	Arduino Mega	DPIN 20	DPIN 21 (nicht getestet)
	ESP-12E nodeMCU	D2	D1
	**** ESP-01**	**GPIO0**	**GPIO2**
** Anderer DPIN mit	Wire.begin(SDA, SCL) im Setup() einfügen und definieren. **Wire.begin(0, 2);**		
Quelle	https://www.arduino.cc/en/Tutorial/MasterReader		

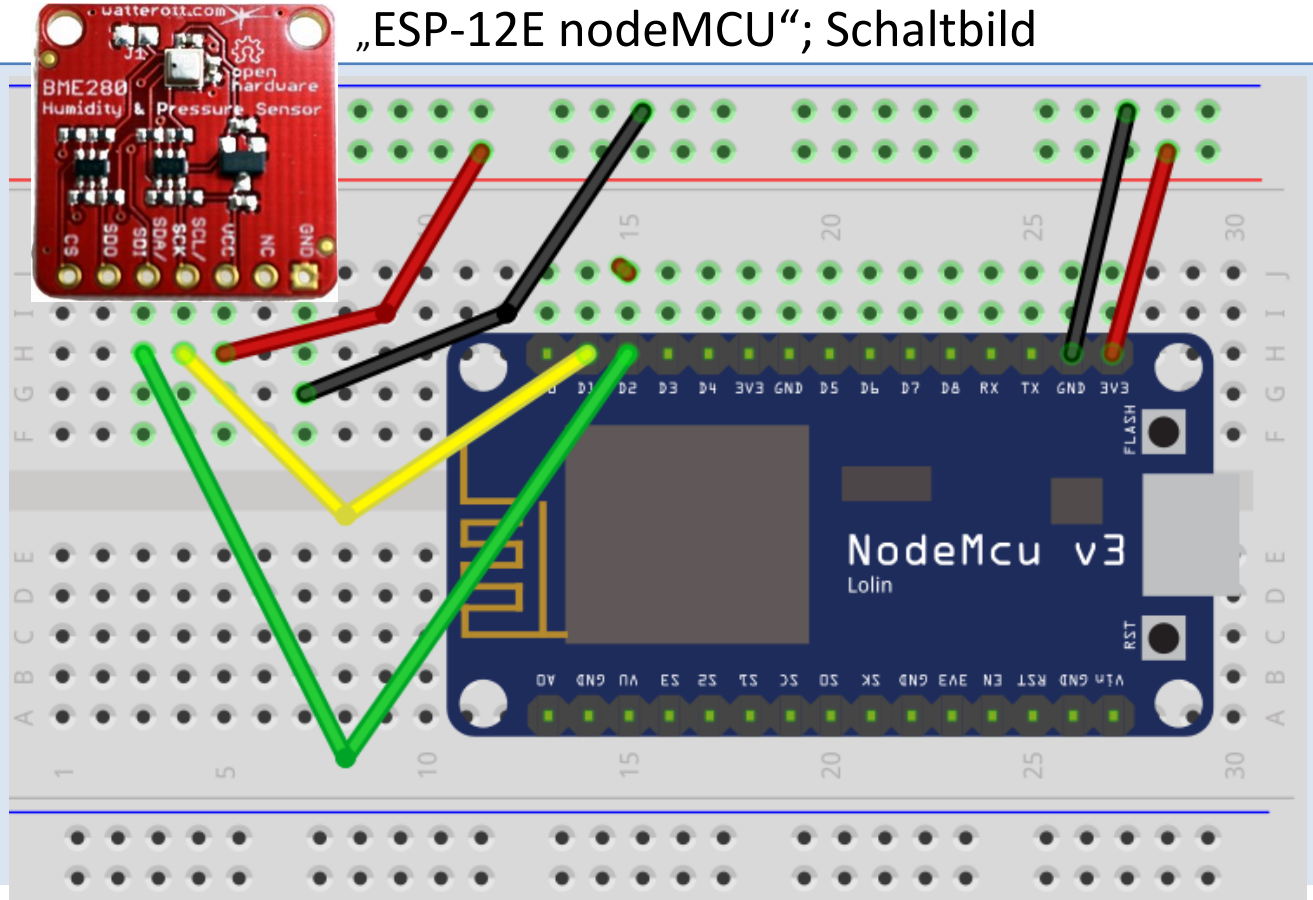
„ESP8266 ESP-01“; Schaltbild



„ESP8266 ESP-01“ & Sketch-Beispiel

Wertvolle Tipps in	http://raphuscucullatus.blogspot.com/2017/07/chinesischer-bme280-sensor-und-esp8266.html		
Die Zeilen 23 bis 26 mit Kommentaren?	Da im I2C-Modus die SPI-DPIN-Zuordnung nicht benötigt wird, kann sie auskommentiert werden.		
I2C in Zeile 30 aktivieren	30 *Adafruit_BME280 bme; // I2C*		
	Board	SDA (data)	SCL (clock)
I2C-Verkabelung „ESP8266 ESP-01 “	ESP-01	GPIO0	GPIO2
Im Codeabschnitt **setup()** noch hinzufügen:	Wire.begin(0, 2);		

„ESP-12E nodeMCU“; Schaltbild



DPIN2 an SDA/SDI und DPIN1 an SCL/SCK

„ESP-12E nodeMCU“ & Sketch-Beispiel

Wertvolle Tipps in	http://raphuscucullatus.blogspot.com/2017/07/chinesischer-bme280-sensor-und-esp8266.html
Sketch öffnen	Datei > Beispiele > „Adafruit BME280 Library“ > „**bme280test.ino**“ öffnen
Die Zeilen 23 bis 26 mit Kommentaren?	Da im I2C-Modus die SPI-DPIN-Zuordnung nicht benötigt wird, kann sie auskommentiert werden.
I2C in Zeile 30 ohne Kommentare	30 *Adafruit_BME280 bme; // I2C*

I2C-Verkabelung „ESP-12E nodeMCU“	`//I2C-wiring nodeMCU`
	`//  BME280  nodeMCU ESP8266 12E`
	`//  SCL/SCK  (Clock)  DPIN 1 GPIO5 Serial-Clock`
	`//  SDA/SDI  (Serial Data In)  DPIN 2 GPIO4 Serial-Data (bi-directed)`

	Der Sketch „bme280test.ino“ sollte ohne Anpassungen funktionieren!

Weitere Infos	github.com/watterott/BME280-Breakout
	Interessante Informationen, die für den Arduino-Programmierer zunächst nicht relevant sind!
Link: Bosch BME280	Produktinformation: https://www.bosch-sensortec.com/en/bst/products/all_products/bme280
	Relevante Informationen finden sich hier:
Link: learn.watterott.com	http://learn.watterott.com/sensors/bme280/
	Für den „BME280 MOD-1022 Weather Multi Sensor“.
	https://github.com/embeddedadventures/BME280
	Für den Bosch BME280 Arduino/Teensy Library
	https://github.com/Protoinfy/BME280_Library