

# Bewässerungssystem mit ESP32 / Arduino

PC-Treff-BB VHS Aidlingen

# Inhaltsverzeichnis

▶	Motivation	
▶	aktueller Status – Beschreibung	
▶	Bewässerungsanlage mit 50L Reservoir	
▶	Recherche – Beispiele	
▶	Systemüberblick mit Kurzbeschreibung	
▶	Fundamente legen	
▶	Rahmenbedingungen schaffen	
▶	Geräte auswählen	
▶	Fehlerhinweise im Compiler-Fenster	
▶	erster Test mit Beispielprogramm „Blink“	
▶	Systemaufbau – Hardware	Baustelle
▶	Systemaufbau – Programm	Baustelle
▶	elektronische Hilfen	
▶	Literatur	

# Motivation

- Ein Bewässerungssystem für Balkonpflanzen ist im Einsatz, welches mittels Zeitschaltuhr nur zeitgesteuert wurde.
- Auch bei Regenwetter wird bewässert = schlecht für die Pflanzen und überlaufende Untersetzer.
- Der Wasservorrat (ca. 50L) soll effektiver genutzt werden und länger halten
- Einfacherer Betreuungsaufwand für eine Urlaubsvertretung

# Aktueller Status - Beschreibung

- mit Wasseranschluss

- Gardena Bewässerungscomputer C1060 solar; Feuchtmessmesser; 1 Kunststoffverteilerrohr; Einzelzuleitungen mit Ausflussregler
- Gießzeit und -mengen-Steuerung

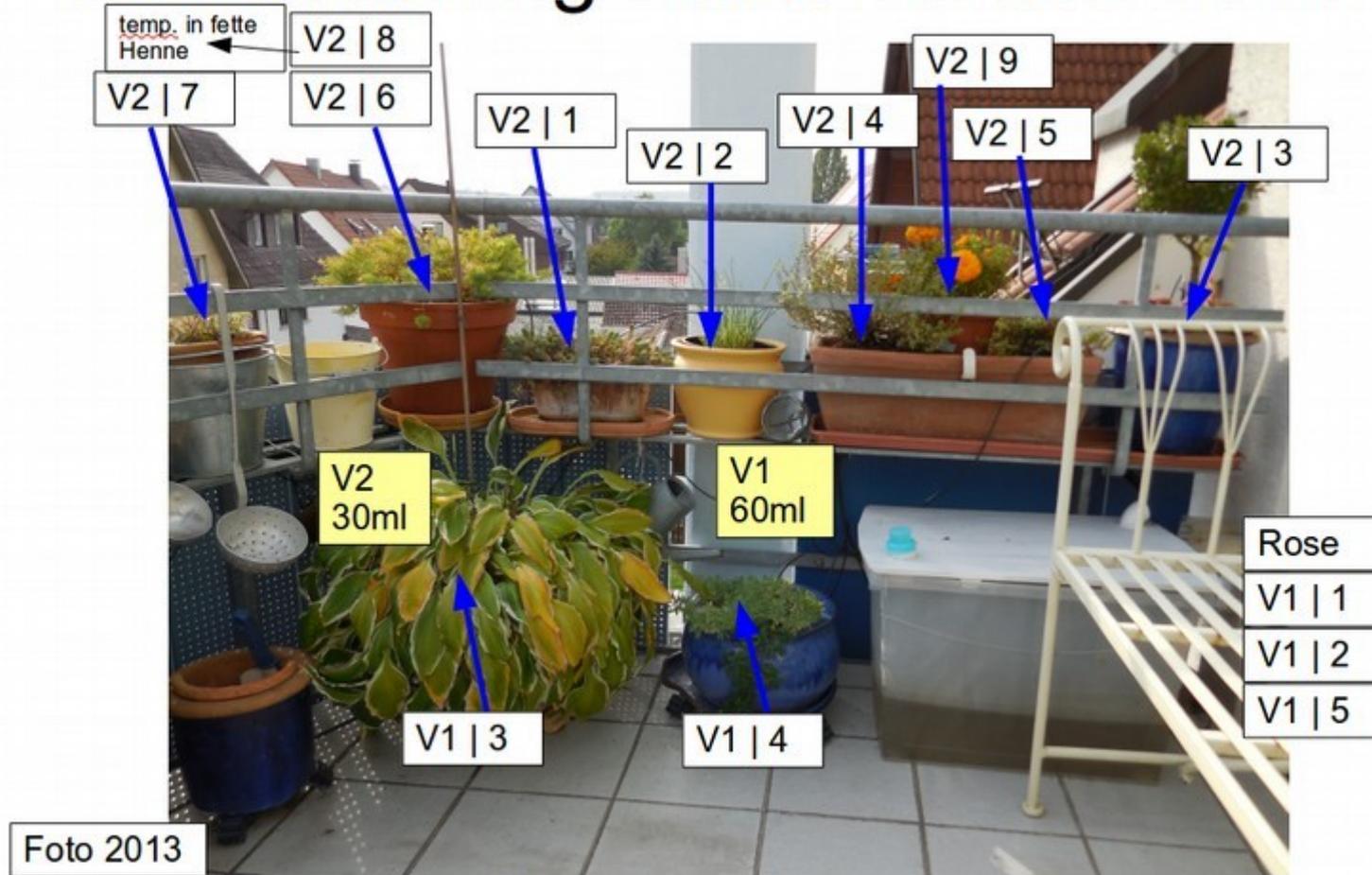


- ohne Wasseranschluss

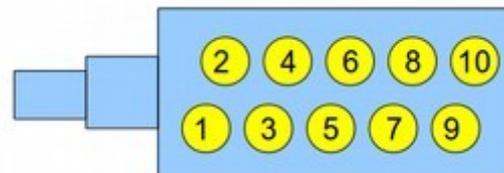
- Gardena Urlaubsbewässerung; Netzteil (1min/24h); Tauchpumpe (180L/h; 1bar); 3 Kunststoffverteilerrohre; 5 Mengenverteiler; Einzelzuleitungen ohne Ausflussregler
- Gießzeitsteuerung durch 220V Zeitschaltuhr  
3x morgens im Abstand 15 min (Hochsommer)  
3x abends im Abstand 15 min (Hochsommer)

# Bewässerungsanlage mit 50L Reservoir

## Bewässerung Balkon unten 2014



160505\_Balkone\_Bewaesserung.odp



- Verteiler V1 = 60ml
- Verteiler V2 = 30ml
- Verteiler V3 = 60ml
- Verteiler V4 = 60ml
- Verteiler V5 = 15ml



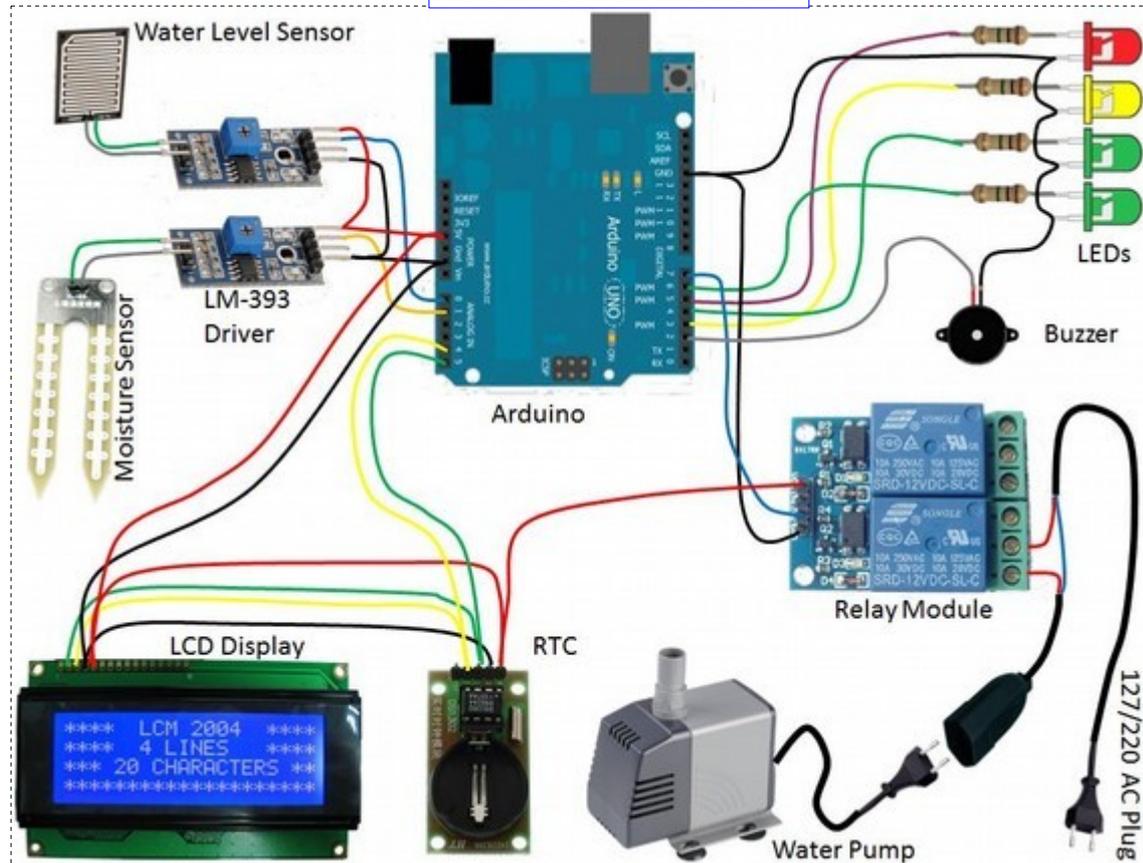
# Systemüberblick

Wasserstands-  
sensor, so  
eingelant

Feuchtigkeits-  
sensor, so  
eingelant

Display als  
OLED  
eingelant

Arduino Uno wird  
ersetzt durch ESP32



LEDs, nicht  
eingelant

Buzzer, nicht  
eingelant

Relais zur  
Ansteuerung  
der Pumpe, so  
eingelant

RTC-Uhr, so  
eingelant

Pumpe mit 12V eingelant  
Tauchpumpe (180L/h; 1bar)

## Kurzbeschreibung

ESP32 wird 2x am Tag aufgeweckt; Uhrzeit mit RTC prüfen; Feuchtemessung im „Master-Topf“ zeigt bewässern ja/nein an; Pumpe fördert Wasser; Wasser-Level Sensor überwacht Wasserstand im Reservoir.

# Fundamente legen

- <https://github.com/espressif/arduino-esp32>
  - Download der neusten Arduino IDE
  - <https://www.heise.de/download/product/arduino-ide-84057/download>
  - Kommandos in Konsolen-Fenster

```
tar xpvf arduino-1.8.5-linux64.tar.xz
```

```
cd arduino-1.8.5/
```

```
arduino-1.8.5/# ./install.sh
```

```
sudo usermod -a -G dialout $USER
```

- Programmierumgebung Java muß installiert sein

```
sudo apt-get install openjdk-8-jre
```

→ **An- und Abmelden oder Rechnerneustart !!!**

*\$USER steht für den angemeldeten user*

*in einer Konsole kann man herausfinden als welcher user man gerade angemeldet ist*

```
>whoami
```

*in welchen Gruppen der user zugehörig ist erfährt man mit*

```
>groups       z.B. ... adm dialout cdrom sudo dip plugdev lpadmin sambashare
```

```
>sudo -s       so wird man root permanent, wenn nötig; mit >exit geht's wieder heraus
```

# Rahmenbedingungen schaffen

- siehe ct-Artikel weiter über link [ct.de/y73t](http://ct.de/y73t) dann „Arduino Core für ESP32“
- dort laut Hinweise zur Debian/Ubuntu Installation alles in ein Konsolenfenster hineinkopieren, mit „Enter“ bestätigen, Passwort eingeben und starten

```
sudo apt-get install git && \
```

```
wget https://bootstrap.pypa.io/get-pip.py && \
```

```
sudo python get-pip.py && \
```

```
sudo pip install pyserial && \
```

```
mkdir -p ~/Arduino/hardware/espressif && \
```

```
cd ~/Arduino/hardware/espressif && \
```

```
git clone https://github.com/espressif/arduino-esp32.git esp32 && \
```

```
cd esp32 && \
```

```
git submodule update --init --recursive && \
```

```
cd tools && \
```

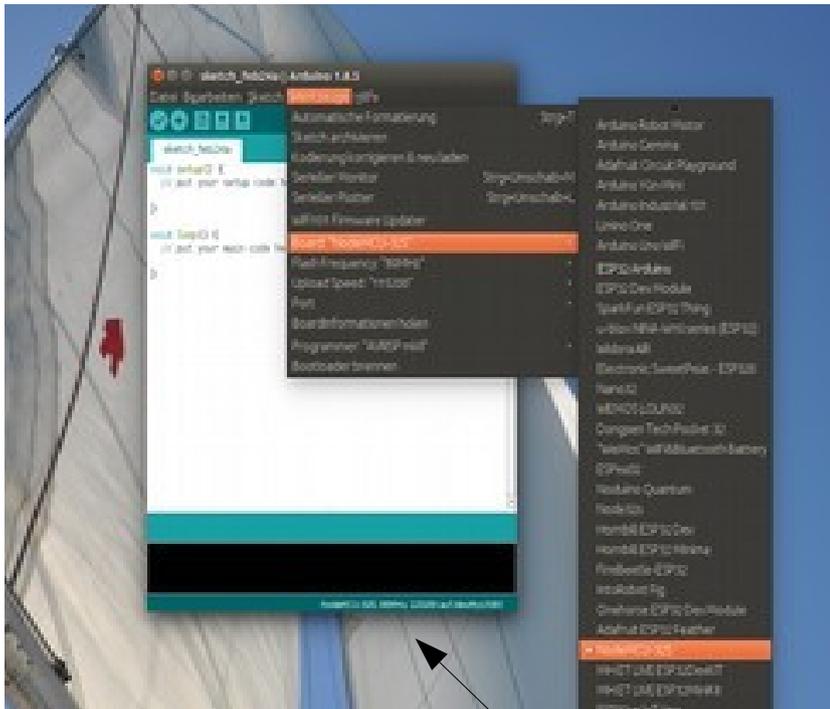
```
python get.py
```

# Geräte auswählen

- Arduino IDE starten



- Werkzeuge/Boards
  - Auswahl ESP32-Modell (NodeMCU-32S od. WEMOS LOLIN32) vornehmen oder ESP32 Dev Module
- Werkzeuge/Upload-Speed
  - Auf 115200 reduzieren, sonst kommt es manchmal zu Übertragungsfehlern

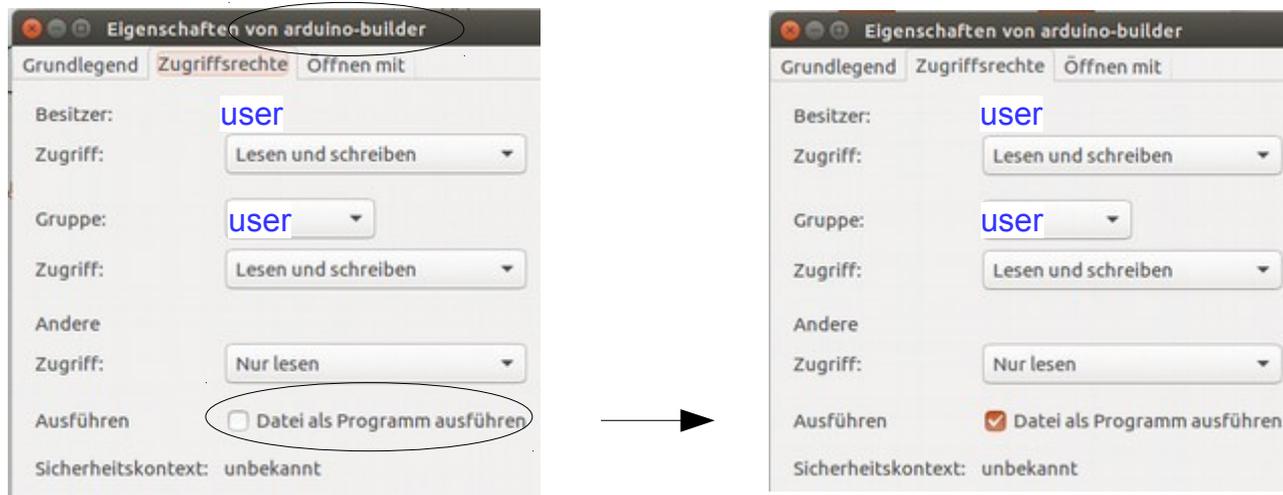


NodeMCU-32S, 80MHz, 115200 auf /dev/ttyUSB0

# Fehlerhinweis im Compiler-Fenster

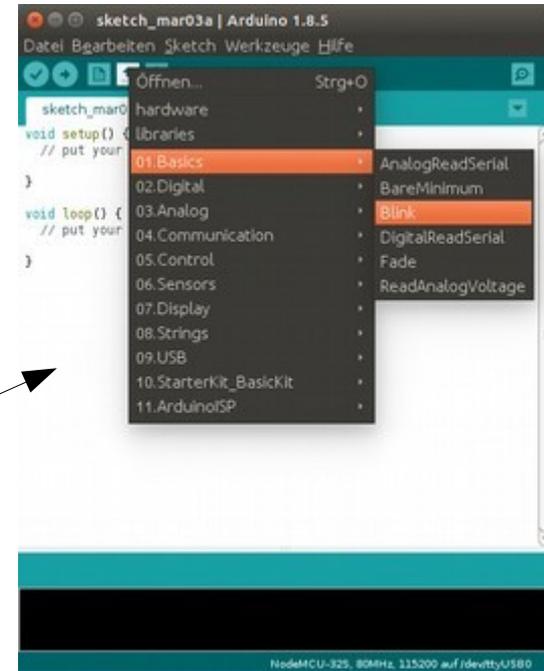
- Wenn beim Kompilieren eine Fehlermeldung erscheint, dann handelt es sich evtl. um die fehlende Kennzeichnung „Datei als Programm ausführen“
  - a) Caused by: java.io.IOException: Cannot run program "/home/user/0\_Kopie\_Arduino/ESP32/ArduinoIDE/arduino-1.8.5/arduino-builder": error=13, Keine Berechtigung
  - b) fork/exec /home/user/0\_Kopie\_Arduino/ESP32/ArduinoIDE/arduino-1.8.5/tools-builder/ctags/5.8-arduino11/ctags: permission denied

Fehler beim Kompilieren für das Board WEMOS LOLIN32.



# erster Test mit Beispielprogramm „Blink“

- Start Arduino IDE
- ESP32 über USB an den PC anstecken
- Beispielprogramm „Blink“ laden
  - Programmcode in Arduino IDE laden



- Beispielprogramm „Blink“ Hochladen
  - Programmcode wird compiliert und zum ESP32 übertragen

Bei Erfolg!



# Systemaufbau - Hardware

hier wird an der Umsetzung der Hardware gearbeitet

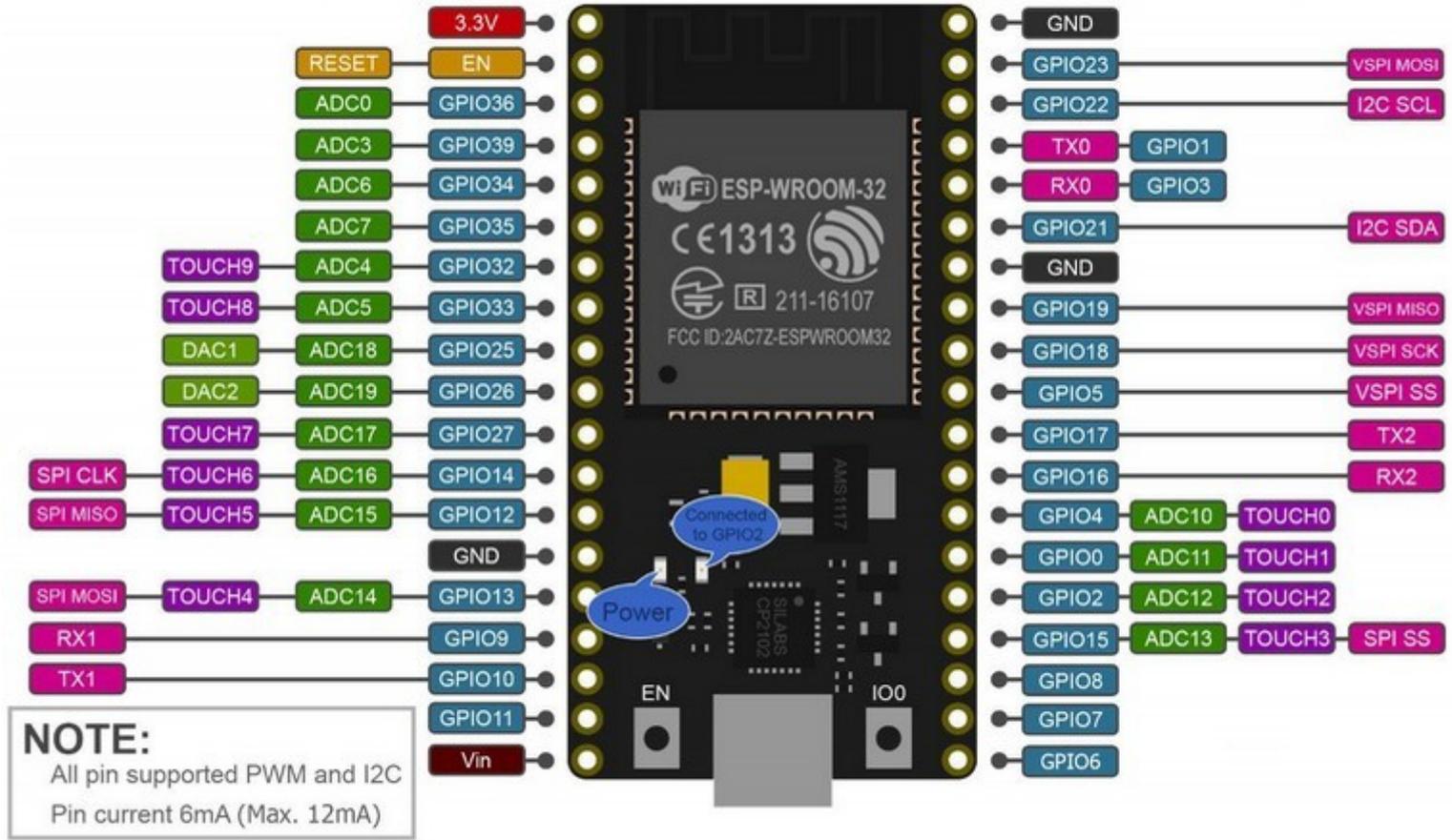
- Espressif ESP32 NodeMCU WROOM32 Dev Board WiFi Bluetooth Development Stück=9,35€
- 4 MB Flash WEMOS Lolin32 V1.0.0 WIFI & Bluetooth Card Based ESP-32 ESP-WROOM-32 mit Anschluss für Akku Stück=5,85€
- Pegelwandler 4 Kanal I2C IIC Logic Level Converter BiDirektional 5V~3.3V Arduino zum Anschluss von 3,3V Komponenten an 5V-Mikrocontroller Stück=2,06€
- WeMos DHT22 Shield Temperatursensor Luftfeuchte AM2302 ESP8266 WiFi D1 mini Stück=6,48€
- Luft- und Bodenfeuchtigkeits Sensor-Modul LM393 Arduino kompatibel Stück=3,77€
- Regentropfensensor Regensensor Wassersensor LM393 Arduino Raspberry PI (0040) Stück=3,39€
- WeMos Relais Shield Relay IoT ESP8266 Stück=3,53€
- 0,96" OLED Display Weiß I2C SDD1306 128x64 Modul Arduino Raspberry Pi Stück=5,59€
- DC-DC Step Down Buck Spannungswandler Modul mit Spannungsregler LED Voltmeter (In 4-40V; Out 1.25V – 37V; 2A kurz 3A; max. 15W) Stück=2,61€
- AMS1117 5V Spannungsregler Modul Voltage Regulator Arduino Raspberry Pi Stück=2,34€
- 3x DS1307 Real Time Clock RTC Modul Echtzeituhr Arduino Raspberry Pi Stück=1,70€
- Breadboard MB-102 830 Kontakte lötfreies Experimentierboard Arduino Raspberry Pi Stück=4,22€
- 60 IoT Kabel Set 20 x male→male + 20 x female→male + 20 x female→female arduino Set=3,87€

# Systemaufbau - Programm

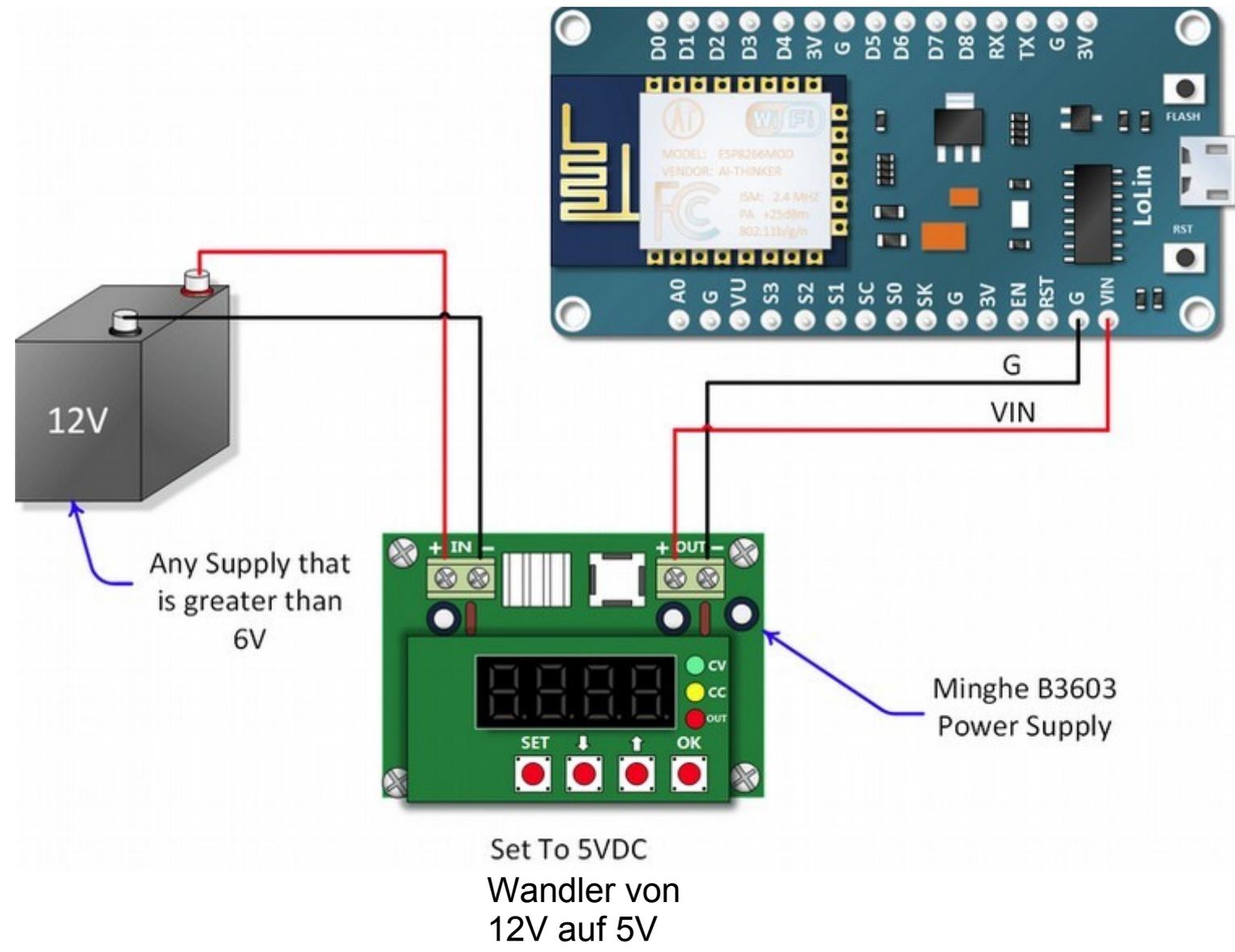
- hier wird an der Umsetzung des Programmes gearbeitet

# NodeMCU-32S Pinout

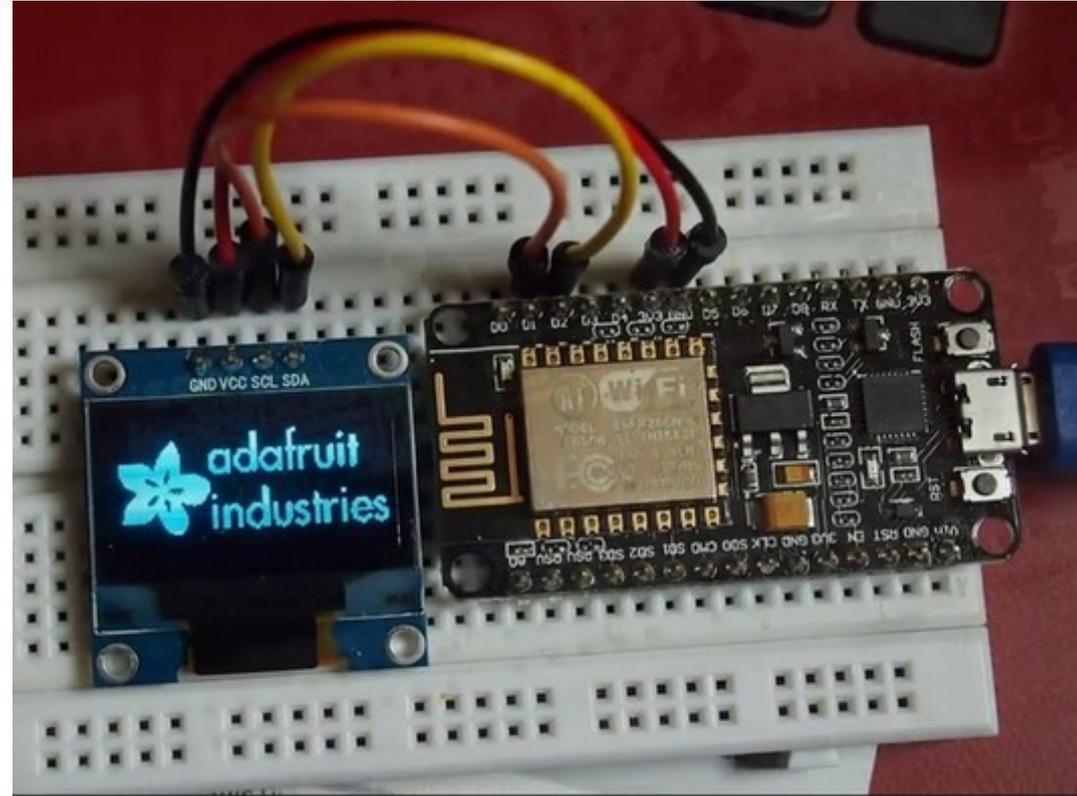
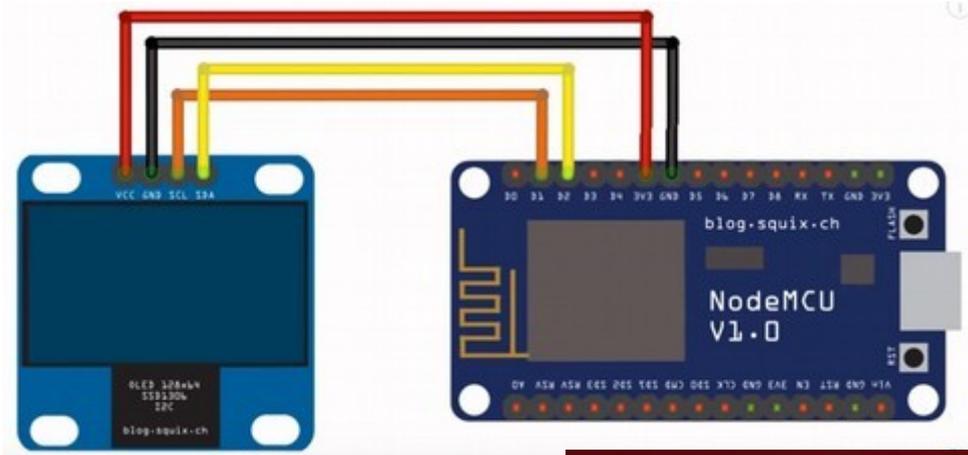
## NodeMCU-32S PINOUT



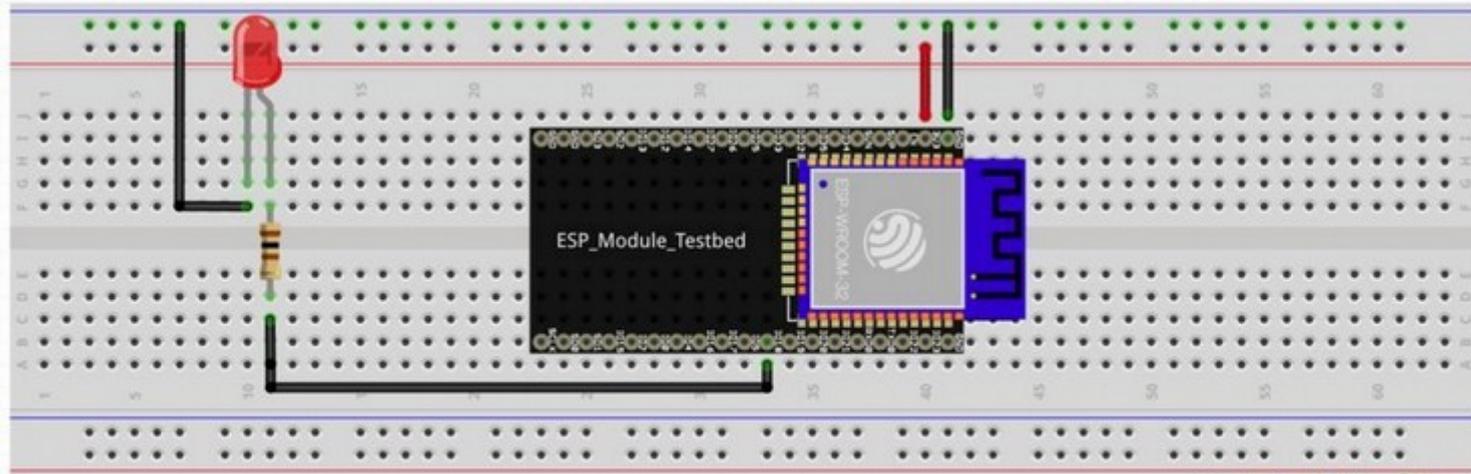
# ESP32 Beispiel Stromversorgung



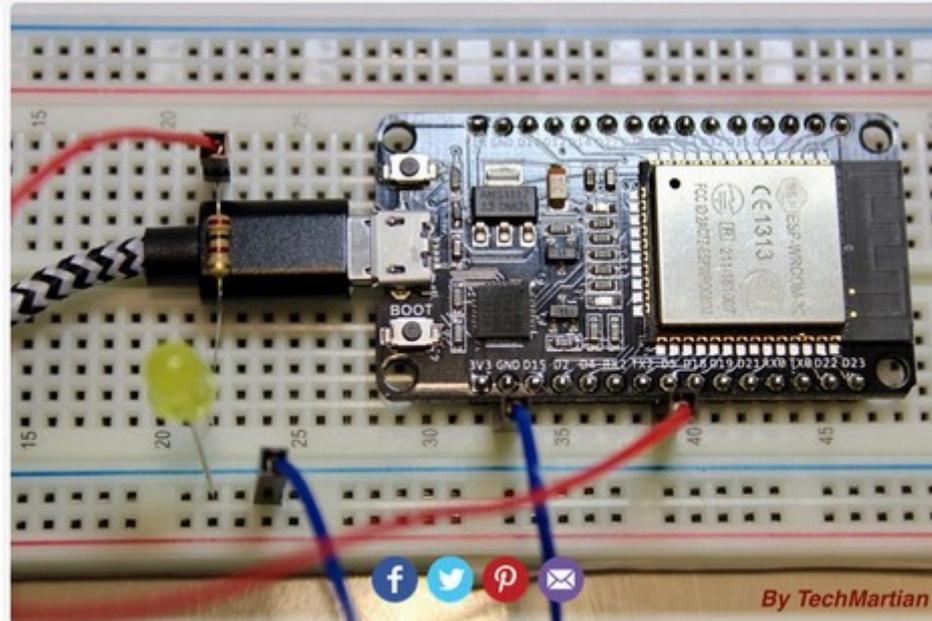
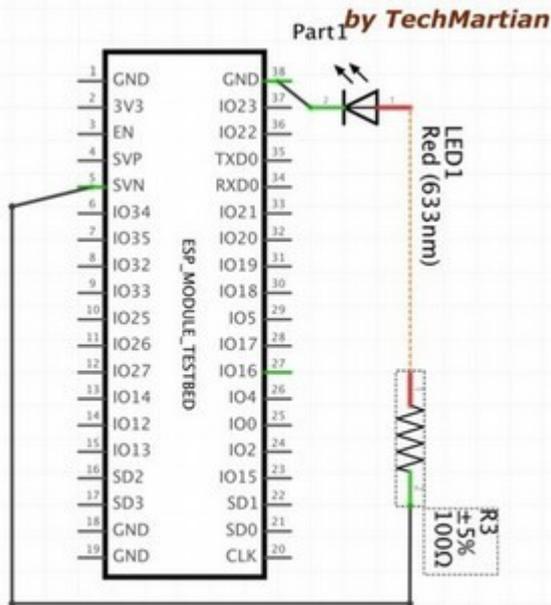
# ESP32 Beispiel OLED Display



# ESP32 Beispiel LED



by TechMartian



By TechMartian

# Farbcodes von Widerständen

- Farbcodes von Widerständen

<http://www.umrechnung.org/elektrischer-widerstand-farbcode/widerstand-farbcode-rechner-tabelle.htm>

**Widerstand-Farbcode Rechner**

Rot Rot Schwarz Gold



Gold  
Silber  
ohne

+/- 5%  
+/- 10%  
+/- 20%

22 Ω ±5%

The image shows a digital resistor color code calculator interface. At the top, it is titled 'Widerstand-Farbcode Rechner'. Below the title are four dropdown menus for the color bands: 'Rot', 'Rot', 'Schwarz', and 'Gold'. In the center, there is a 3D rendering of a resistor with four color bands: red, red, black, and gold. To the right of the resistor is a legend for the tolerance bands: 'Gold' (pointing to +/- 5%), 'Silber' (pointing to +/- 10%), and 'ohne' (pointing to +/- 20%). At the bottom, there are three input fields: a text box containing '22', a dropdown menu for the unit symbol 'Ω', and a dropdown menu for the tolerance '±5%'.

# Formeln und Umrechnungen

- Spannungsabfall

Pin am ESP32 3,3V – Spannungsabfall einer LED 0,7V = 2,6V

- Ohmsches Gesetz

$R = U / I$        $2,6V / 0,00012A = 21,6\text{Ohm}$  → welcher Farbcode?

- Elektrische Leistung

$P = U * I$        $3,3V * 0,00012A = 0,000396W = 3,96\text{mW}$

$P = I^2 * R$

- $0,12\text{mA} = 0,00012A$
- $21,6\text{Ohm} = 0,0216\text{kOhm}$
- $1\text{mWatt} = 0,001W$

# Weitere Infos

- Spannungsregler verheizt Wärme
- Step-down od. Step-up-Regler setzen Spannung effizient/verlustfrei um
  - 24v/12V → 5V od. 3,3V
- Gute Technikanleitung für Elektronikbauteile bei [www.exp-tech.de](http://www.exp-tech.de)
- Gute Elektronikbauteile mit Anleitungen von Firma Adafruit
- Sensoren benötigen Betriebsspannung und geben dann Signale ab
- Logig-Eingang ist direkt anschließbar
- Leistungs-Eingang (z.B. Relais) nur mit Treiberbaustein anschließen
  - UTC njcg/ ULN2004L
    - Spannung für ULN2004L anschließen
    - Pin eine Seite rein z.B. von ESP32 → andere Seite raus zum Relais
- <https://github.com/espressif/arduino-esp32>
  - Installation instructions for Debian / Ubuntu OS

# Literatur

- ct 02/2018 „Smarte Helfer selbst gebaut“ von Merlin Schumacher
- 2017 „PomodoPi“ Heise Archivseite  
<https://shop.heise.de/katalog/pomodopi-bf3c22>
- TechMartian <http://www.instructables.com/id/Blinking-an-LED-With-ESP32/>

धन्यवाद  
Hindi

多謝  
Traditional Chinese

ขอบพระคุณ  
Thai

Спасибо  
Russian

Graci  
Spanish

شكراً  
Arabic

Thank  
English  
You

as  
Obrigado  
Brazilian Portuguese

Grazie  
Italian

多谢  
Simplified Chinese

Danke  
German

Köszönöm  
Hungarian

Merci  
French

நன்றி  
Tamil

ありがとうございました  
Japanese