TECHNISCHE HOCHSCHULE
OSTWESTFALEN-LIPPE
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS

Messgerät für die Vermessung des Lemgoer Stadtklimas

Dokumentation zu Aufbau und Einrichtung

Im Rahmen des Projektes „Lemgoer Stadtklima“

Mika Stellbrink

Mika.Stellbrink@stud.th-owl.de

19.06.2025

Inhaltsverzeichnis

1. Einführung

1.1. Beschreibung

Diese Dokumentation sowie das dazugehörige Messgerät wurden im Rahmen der Projektarbeit „Lemgoer Stadtklima“ im Sommersemester 2025 an der Technischen Hochschule Ostwestfalen Lippe (TH OWL) von Jonas Möllmann und Mika Stellbrink erstellt.

In diesem Dokument wird der Aufbau sowie die Einrichtung des neu entwickelten Messgerätes auf Basis eines ESP32-Mikrocontrollers beschrieben.

1.2. Benötigte Hardware

Eine vollständige Auflistung und Beschreibung der Komponenten ist der finalen Projektdokumentation beigefügt. Folgende Komponenten werden für den Aufbau des Messgeräts benötigt:

- ESP32-WROOM-32E Mikrocontroller als zentrale Komponente
- GY-BME280 Barometrischer Sensor
- Breakout-Board für MicroSD-Karten
- Grove DS1307 Echtzeituhr (RTC)
- Adafruit monochromes 1.3" Display (128x64 OLED-Pixel)
- GUVA-S12SD UV-Sensor
- Grove Air530 GPS-Sensor
- Steckboard („Breadboard“) – 400 Kontakte
- TP4056 Batterieladungsmodul und Schutzschaltung (USB Typ C)
- Samsung INR18650-30Q Batterie (3,7V; 3000mAh)
- Batteriehalter für 18650 Zelle mit Anschluss (optional)
- 3-Pin-Kipp- oder Schiebeschalter (Zwei Stück)
- Jumper-Kabel (mindestens 36 Stück)
- Gehäuse (3D-gedruckt)

1.3. Pinbelegung des ESP32

Die Pinbelegung kann bei den verschiedenen Varianten des ESP32 variieren. Für dieses Projekt werden folgende Dokumente als Quellen verwendet:

- https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf - Offizielles Datenblatt vom Hersteller (Espressif), abgerufen am 19.06.2025
- https://docs.sunfounder.com/projects/esp-cam-kit/de/latest/component_esp32.html - Dokumentation von der Firma „SunFounder“, abgerufen am 19.06.2025

Aus der zweiten Quelle wurde speziell das folgende Diagramm verwendet, da es einen klaren Überblick über die einzelnen Pins und deren Funktionen bietet:

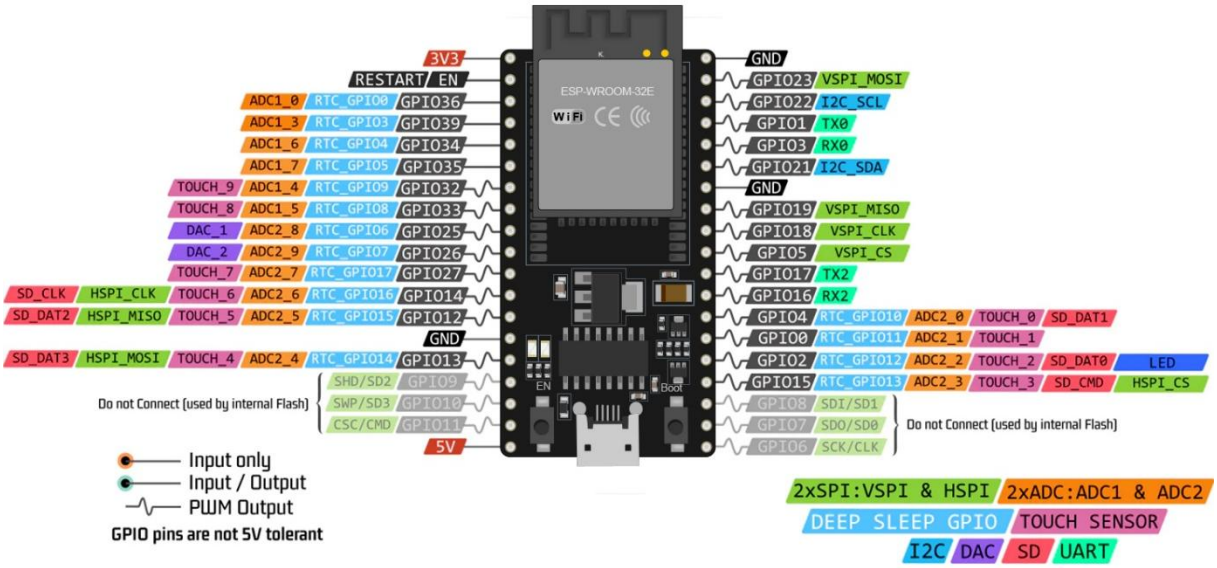

Abbildung 1: Pinbelegung des ESP32-WROOM-32E, Quelle: https://docs.sunfounder.com/projects/esp-cam-kit/de/latest/component_esp32.html

2. Aufbau des Messgeräts

2.1. ESP32 auf dem Steckboard anschließen

Die Pins des ESP32 sind so angeordnet, dass der Mikrocontroller direkt auf ein Standard-Steckboard aufgesteckt werden kann:

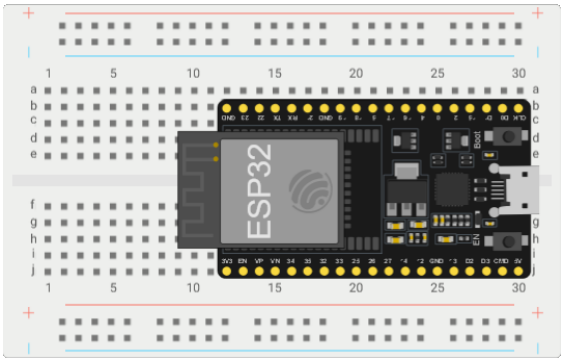

Abbildung 2: ESP32 auf normalem Breadboard, Quelle: https://wokwi.com

Der ESP32 ist allerdings so breit, dass auf einer Seite (in diesem Beispiel unten) keine Möglichkeit mehr besteht, andere Komponenten über das Steckboard mit den Pins des Controllers zu verbinden.

Um dieses Problem zu beheben, wurde das Steckboard in der Mitte durchgesägt und auseinandergezogen, wodurch nun auf jeder Seite des ESP32 drei (bei Bedarf auch vier) Pins zur Verfügung stehen:

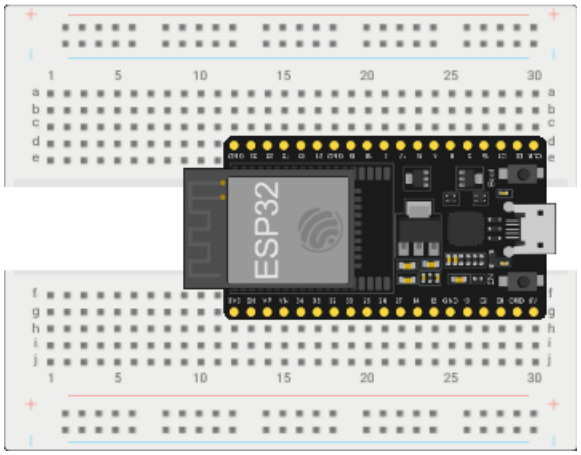

Abbildung 3: ESP32 auf geteiltem Breadboard, Quelle: https://wokwi.com, editiert von Mika Stellbrink

2.2. Interne Stromversorgung

Für die Versorgung der einzelnen Sensoren und Komponenten werden sowohl der 3.3V als auch der 5V Pin des ESP32 benötigt. Das Steckboard bietet hierfür jeweils zwei Versorgungslinien am rechten und linken Rand an, die bereits mit „+“ und „-“ beschriftet sind.

Die 5V Versorgung wird hier auf der linken und die 3.3V Versorgung auf der rechten Seite angeschlossen, die Masse-Pins (GND) können frei gewählt werden:

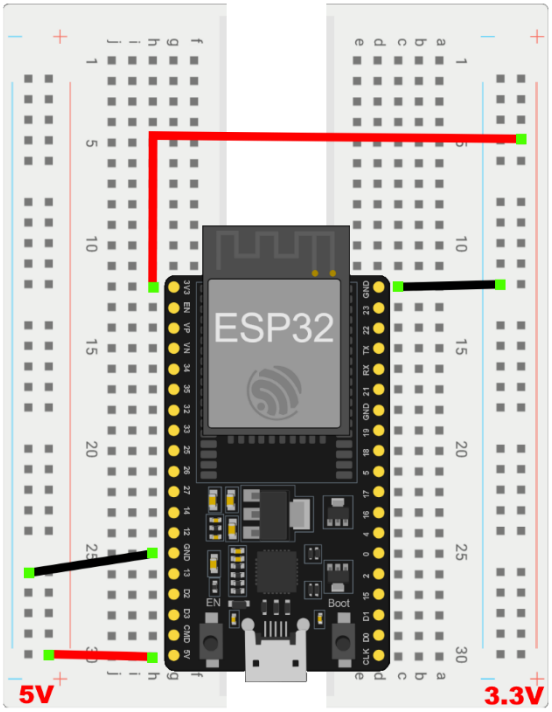

Abbildung 4: Interne Stromversorgung mit dem ESP32, Erstellt von: Mika Stellbrink

2.3. BME280 – Barometrischer Sensor

- SCL → GPIO22 (I2C SCL)
- SDA → GPIO21 (I2C SDA)
- VCC → 3.3V (+, rechts)
- GND → GND (-, rechts)

2.4. Breakout-Board für MicroSD-Karten

- CS → GPIO5 (VSPI CS)
- SCK → GPIO18 (VSPI CLK)
- MOSI → GPIO23 (VSPI MOSI)
- MISO → GPIO19 (VSPI MISO)
- VCC → 5V (+, links)
- GND → GND (-, links)

2.5. Grove DS1307 – Echtzeituhr (RTC)

- SCL → GPIO22 (I2C SCL)
- SDA → GPIO21 (I2C SDA)
- VCC → 5V (+, links)
- GND → GND (-, links)

2.6. Adafruit monochromes 1.3" Display

- Data → GPIO21 (I2C SDA)
- Clk → GPIO22 (I2C SCL)
- A0 / DC → nicht verbunden
- Rst → nicht verbunden
- CS → nicht verbunden
- 3Vo → nicht verbunden
- VIN → 3.3V (+, rechts)
- GND → GND (-, rechts)

2.7. GUVA-S12SD – UV-Sensor

- SIG → GPIO34 (ADC1-6)
- NC → nicht verbunden
- VCC → 3.3V (+, rechts)
- GND → GND (-, rechts)

2.8. Grove Air530 – GPS-Sensor

- TX → GPIO16 (RX2)
- RX → GPIO17 (TX2)
- VCC → 5V (+, links)
- GND → GND (-, links)

2.9. Kippschalter für WiFi-Funktion

Zum Ein- und Ausschalten des internen WiFi-Chips wurde ein Kippschalter installiert. Dieser wird vom Controller ausgelesen, um den WiFi-Adapter zu steuern. Der Adapter wird nicht physisch gesteuert, sondern über die Programmlogik.

Grundsätzlich kann jeder GPIO-Pin verwendet werden, sofern dieser nicht durch ein anderes Gerät belegt oder im Pinout-Diagramm als „Do not Connect“ gekennzeichnet ist.

Der ESP32 hat für die GPIO-Pins einen internen Pull-Up-Widerstand, wodurch der Schalter automatisch entprellt wird und kein zusätzlicher Widerstand zwischen Schalter und Controller verbaut werden muss.

- Schalter-Pin 1 → GPIO13
- Schalter-Pin 2 → GND (-, links)
- Schalter-Pin 3 → nicht verbunden

2.10. TP4056 Batterieladungsmodul und Schutzschaltung

Wichtig: Die Batterie (im eingeschalteten Zustand) und der USB-Anschluss des Mikrocontrollers dürfen nicht gleichzeitig angeschlossen sein, da dies den Mikrocontroller beschädigen oder zerstören kann.

Die 18650-Batterie wird mit Hilfe eines Batteriehalters mit vorgefertigten Anschlüssen angeschlossen. Alternativ kann die Batterie auch direkt am Plus- und Minuspol mit Drähten verlötet werden, was jedoch den späteren Austausch der Batterie erschwert.

Zwischen das Ladungsmodul und den ESP32 wird noch ein Schalter eingebaut, welcher es ermöglicht, das Messgerät vollständig auszuschalten. Die gesamte Energieversorgung des Messgerätes erfolgt über die Batterie und das Ladungsmodul.

Die Komponenten müssen wie folgt verbunden werden:

- TP4056 Lademodul:
  - OUT+ → Pin 1 des Kippschalters
  - B+ → Pluspol der Batterie (+)
  - B- → Minuspol der Batterie (-)
  - OUT- → GND des Mikrocontrollers (-, linke Seite)
- Kipp- / Schiebeschalter
  - Pin 1 → OUT+ des TP4056
  - Pin 2 → 5V des Mikrocontrollers (+, linke Seite)
  - Pin 3 → nicht verbunden

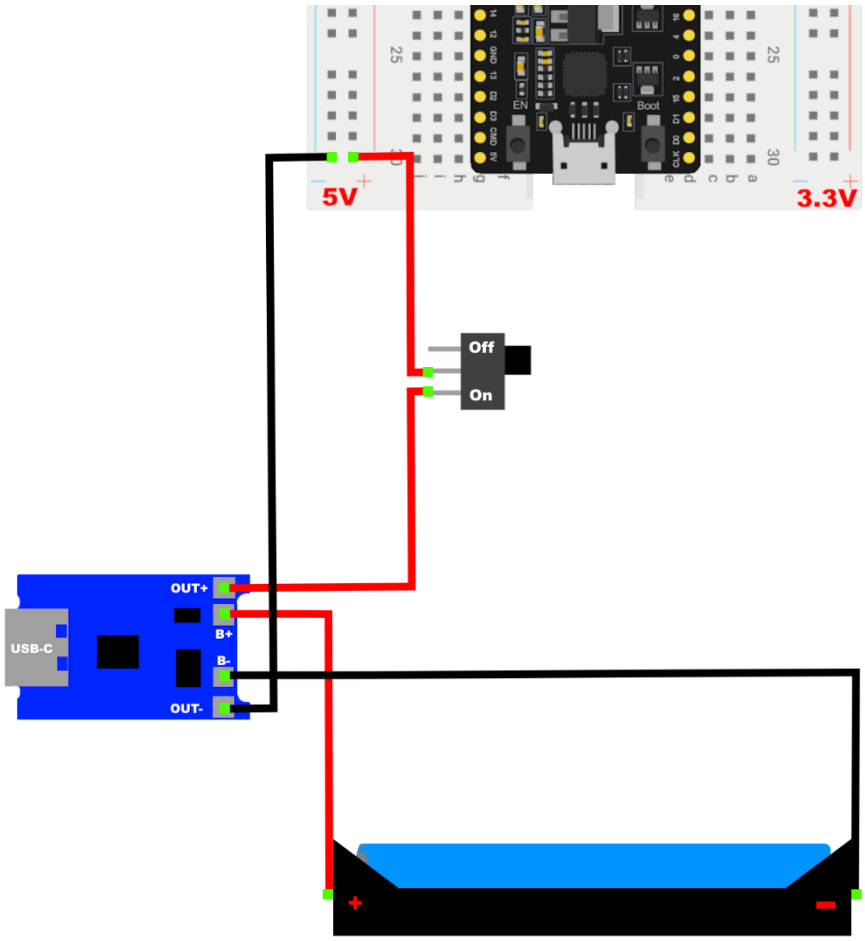

Abbildung 5: Externe Stromversorgung des Messgeräts, Erstellt von: Mika Stellbrink

2.11. Vollständiges Messgerät

Eine Übersicht des vollständigen Messgerätes, ohne das dazugehörige 3D-gedruckte Gehäuse, ist nachfolgend abgebildet.

Die Kabel zur Versorgung der Komponenten (VCC, 5V und 3.3V) sind rot dargestellt, die Masse-Kabel (GND) sind jeweils schwarz dargestellt, die I2C-SDA-Verbindungen in dunkelgrün und die I2C-SCL-Verbindungen in Gelb. Alle anderen Farben dienen nur der besseren visuellen Unterscheidung der Verbindungen.

Einige Verbindungen, z.B. die „Signal“-Verbindung des UV-Sensors oder der Schalter zum Ein- und Ausschalten des WiFi-Adapters können bei Bedarf auch anders belegt werden. Dazu wären jedoch einzelne Anpassungen im Quellcode nötig, um weiterhin die korrekten Pins anzusteuern.

Im Rahmen dieser Projektarbeit wird für alle gebauten Messgeräte der folgende Aufbau verwendet:

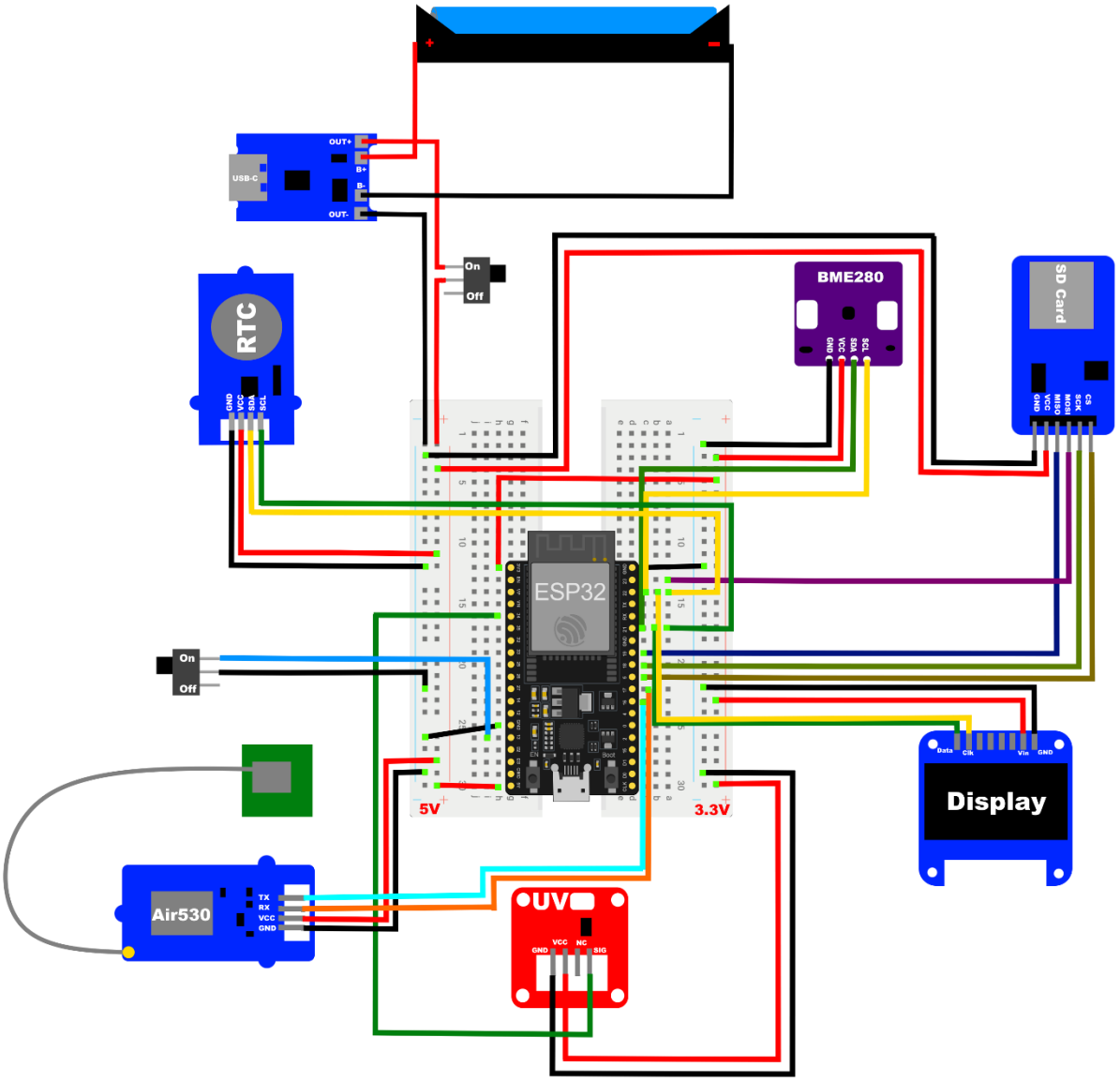

Abbildung 6: Aufbau des vollständigen Messgeräts, Erstellt von: Mika Stellbrink

3. Quellcode laden

3.1. Anbindung an die IDE

Während der Verbindung mit einem Computer über die USB-Schnittstelle darf der ESP32-Mikrocontroller nicht über die Batterie mit Strom versorgt werden, da dies den Controller beschädigen kann! Es empfiehlt sich, sicherheitshalber die Batterie für den nachfolgenden Prozess vollständig vom Messgerät zu trennen.

Für die Entwicklung und das Debugging des Quellcodes wird die vom Mikrocontroller-Hersteller bereitgestellte „Arduino IDE“ eingesetzt, da diese neben den Arduino-Mikrocontrollern auch die ESP32-Geräte von Espressif unterstützt.

Die IDE kann über den folgenden Link heruntergeladen und installiert werden:

- https://www.arduino.cc/en/software/

Zusätzlich muss, damit die serielle UART-Schnittstelle dieses Modells korrekt verwendet werden kann, der „CP210x USB to UART“ Treiber installiert werden. Auf der folgenden Seite muss hierfür das Paket „CP210x Windows Drivers“ heruntergeladen, entpackt und installiert werden:

- https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads

Nach der Installation sollte die USB-Schnittstelle des Controllers beim Anschließen korrekt erkannt werden:

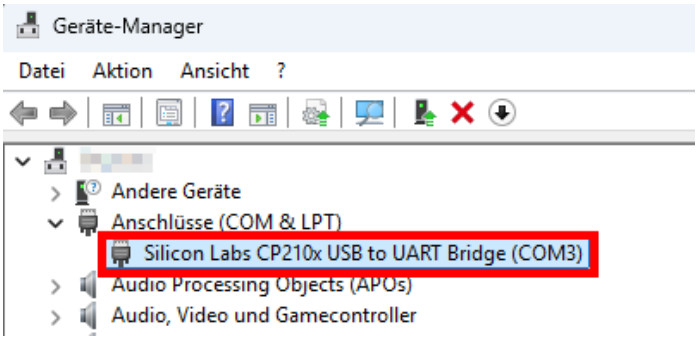

Abbildung 7: ESP32 im Windows Geräte-Manager

In der Arduino IDE muss zunächst die Bibliothek für den ESP32 installiert werden. Über „Tools“ → „Board“ → „Boards Manager“ kann der Boards Manager geöffnet werden. In diesem muss nach dem „ESP32“ gesucht und das dazugehörige Paket von „Espressif Systems“ installiert werden. Die Installation erfordert eine Internetverbindung und kann mehrere Minuten in Anspruch nehmen.

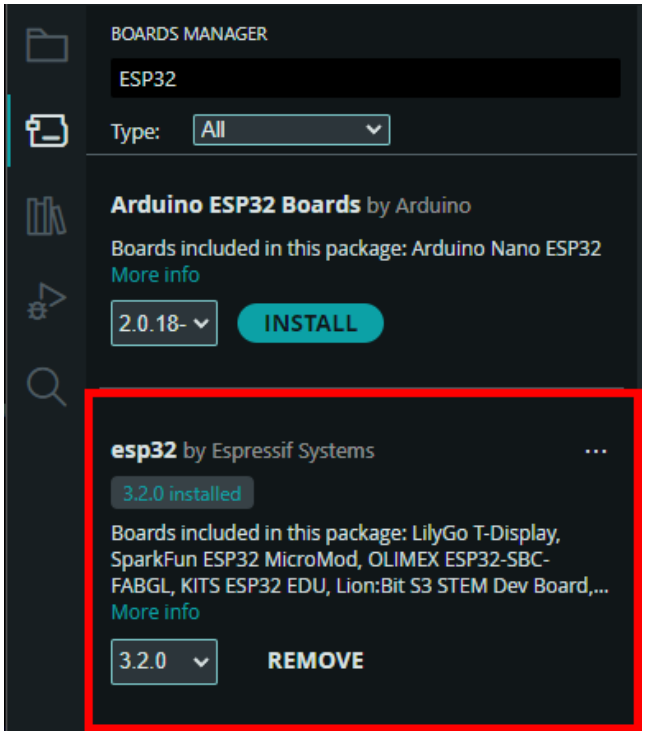

Abbildung 8: Installation des ESP32-Boards in der Arduino IDE

Anschließend muss oben bei der Auswahl des Boards nach dem „ESP32 Dev Module“ gesucht werden. Die zuvor installierte Bibliothek bietet Unterstützung für eine Vielzahl an ESP32-Typen, für dieses Projekt wird die Option „ESP32 Dev Module“ verwendet.

Auf der rechten Seite wird der, zuvor im Geräte-Manager exemplarisch darstellte serielle Port des ESP32 ausgewählt werden. Dieser kann je nach Anschlussart und Computer variieren.

Anschließend kann die Auswahl bestätigt werden und der Mikrocontroller ist einsatzbereit.

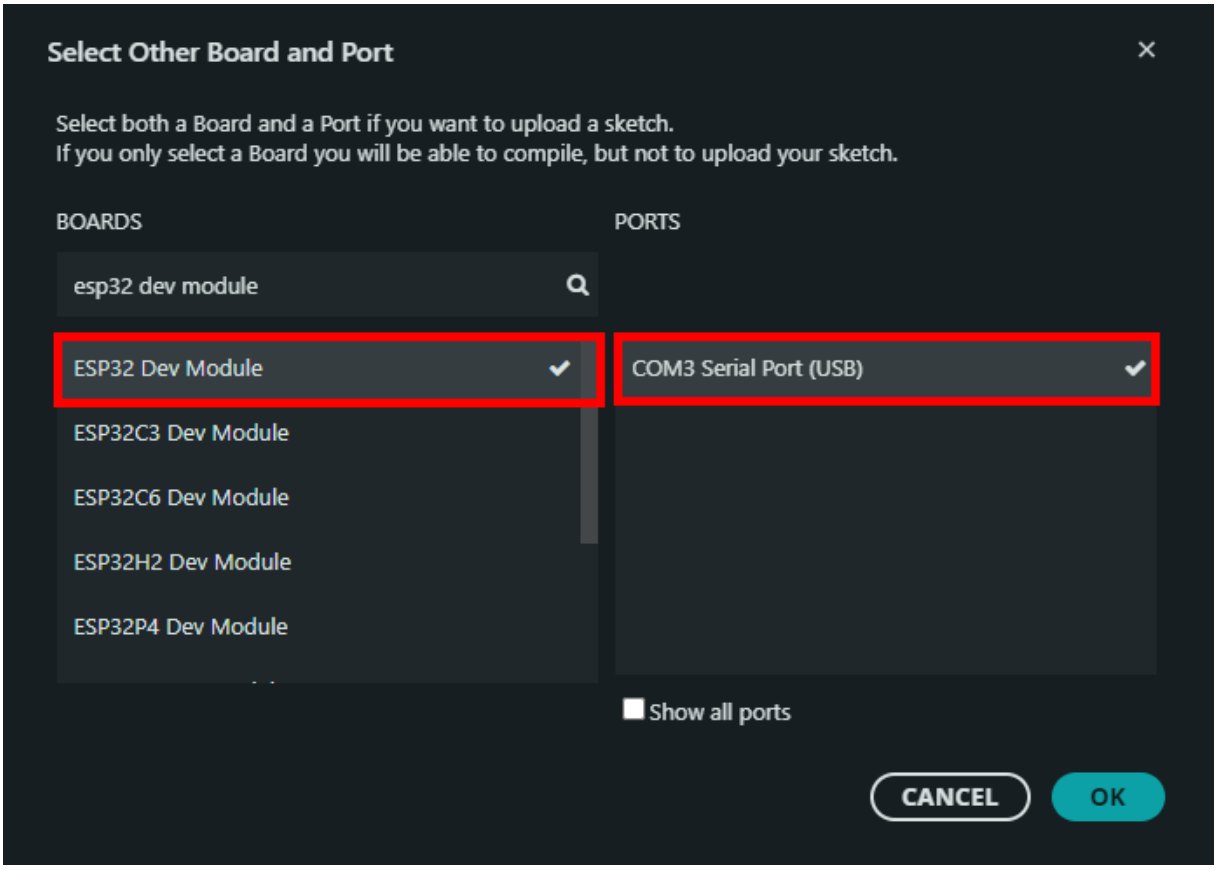

Abbildung 9: Auswahl des Boards und Ports in der Arduino IDE

3.2. Bibliotheken installieren

Für die Nutzung der verschiedenen Komponenten müssen diverse Bibliotheken zum korrekten Auslesen der Messwerte installiert werden. Bis auf die Bibliothek für den GUVA-S12SD UV-Sensor sind alle Bibliotheken direkt über die Arduino DIE installierbar.

Hierzu muss der „Library Manager“ in der linken Seitenleiste aufgerufen und die folgenden Bibliotheken gesucht und installiert werden:

- “SD” by Arduino und SparkFun
- “Soldered BME280 and BME680 EasyC Library” by Adafruit
- “Adafruit SSD1306” by Adafruit
- “Adafruit GFX Library” by Adafruit
- “Grove – RTC DS1307” by Seeed Studio
- “TinyGPSPlus” by Mikal Hart

Die Bibliothek für den GUVA-S12SD UV-Sensor muss manuell über GitHub heruntergeladen werden:

- https://github.com/ma2shita/GUVA-S12SD

Anschließend muss das Verzeichnis mit der Bibliothek in das lokale Bibliotheken-Verzeichnis der Arduino IDE verschoben werden. Unter Windows liegt dieses standardmäßig im Benutzerverzeichnis unter Dokumente → Arduino → libraries. In diesem Verzeichnis sind auch die zuvor über die IDE installierten Bibliotheken abgelegt.

3.3. Quellcode herunterladen

Der im Rahmen dieses Projekts entwickelte Quellcode wird über ein GitHub-Repository bereitgestellt und kann von diesem heruntergeladen werden:

- https://github.com/mstellbrink/klimamessgeraet

Neben dem finalen Quellcode des Messgeräts im Ordner „klimamessgeraet“ befinden sich im Verzeichnis „SensorSamples“ Beispiel-Quellcodes für die Nutzung der einzelnen Komponenten des Messgeräts.

Der Quellcode befindet sich als Arduino-Sketch-Datei („ino“) im Verzeichnis klimamessgeraet → src. Er kann mit der Arduino IDE geöffnet werden.

Nachdem der korrekte Mikrocontroller ausgewählt wurde, kann der Quellcode über die „Upload“-Schaltfläche auf den ESP32 geladen und ausgeführt werden.

Für die mobile Nutzung muss das Messgerät anschließend vom USB-Anschluss getrennt und über die Batterie mit Strom versorgt werden. Durch die angeschlossene Echtzeituhr ist der Controller auch bei Unterbrechungen der Stromversorgung in der Lage, intern die korrekte Zeit beizubehalten.