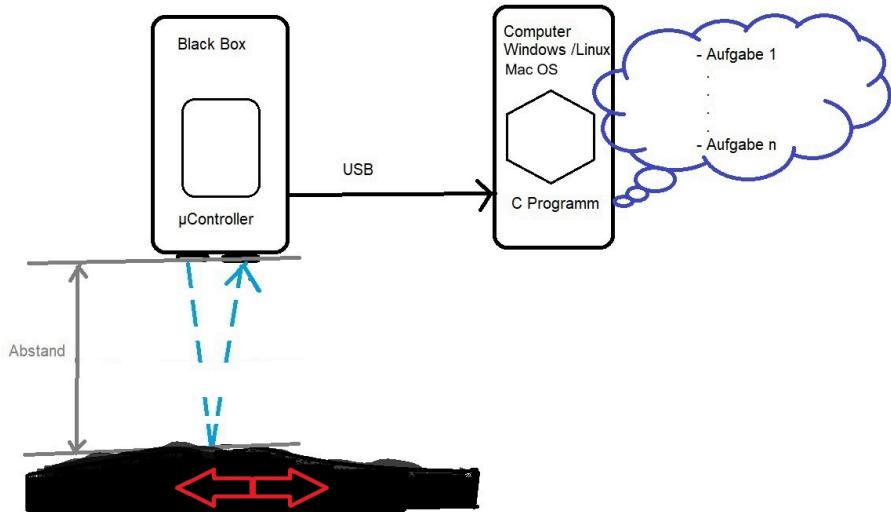


Laboraufgabe.

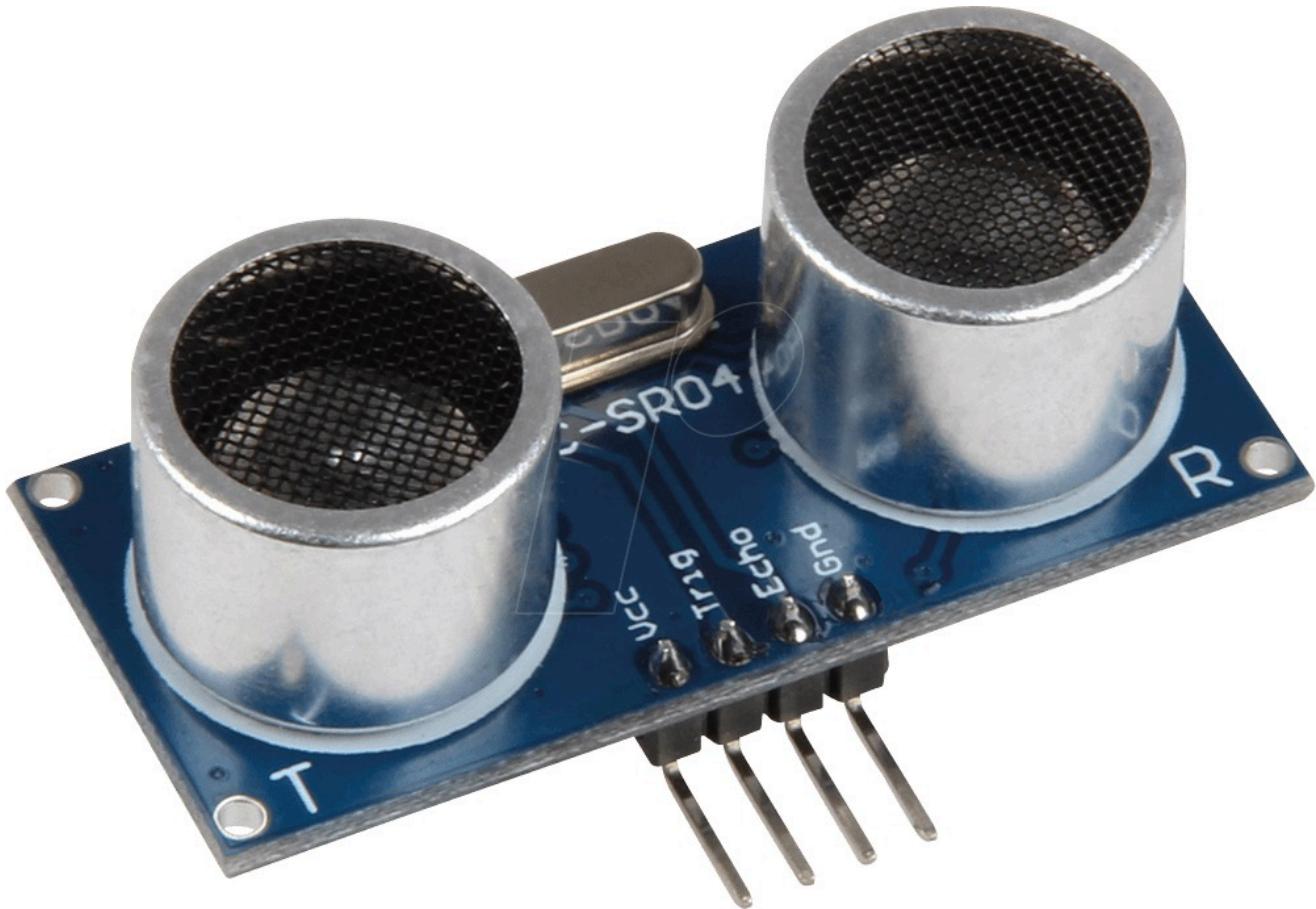
Es soll eine Höhenmesser entwickelt werden, der ein Höhenprofil aufnehmen kann. Dazu wird in diesem Labor die Informatik - Komponente entwickelt.



Handskizze

Für die Entwicklung ist schon ein Hardware-Aufbau (Black-Box) vorhanden. Diese bietet verschiedene Grundfunktionen an, die anhand der Aufgaben erweitert werden sollen.

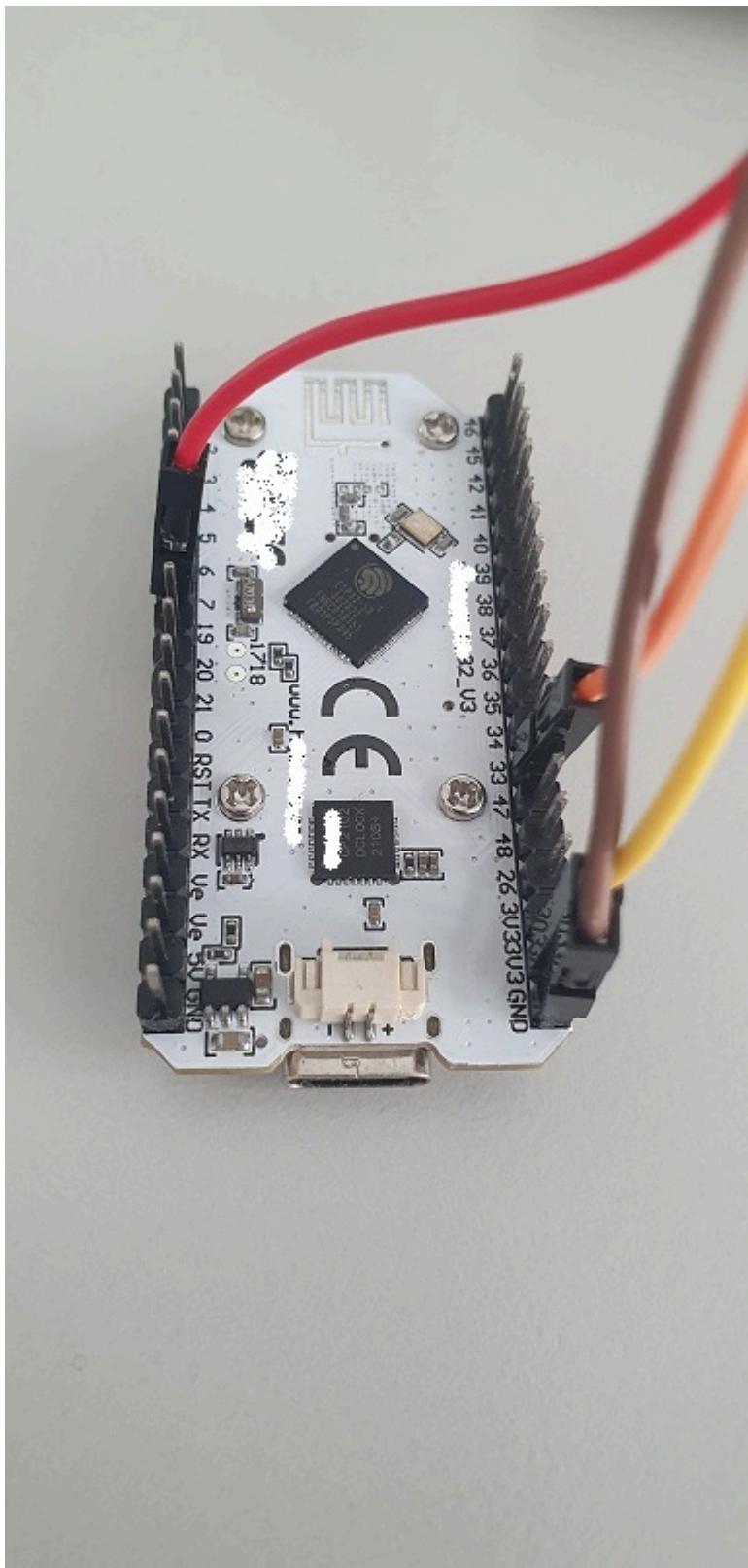
Beschreibung Hardware-Blackbox:



Ultraschal Sensor



μController Vorderseite



μController Rückseite

Figure 1: Komponenten der Black-Box.



Ultraschal Sensor

Die Blackbox ist ein Gehäuse in dem ein µ Controller Signale von einem Ultraschall Sensor (US-Sensor) erfasst diese verarbeitet und über die USB Schnittstelle ausgibt. Die gesendeten Daten sind im INT (Integer) Format und werden mit einer Frequenz von 1Hz ausgegeben. Die Spannungsversorgung erfolgt auch über den USB Anschluss (5V/0,500mA).

Rahmenbedienung

- Es muss in C programmiert werden.
- Verwendung einer sqlite-Datenbank
- Verwendung von `structs`
- Aufteilen des Codes in Header (.h) und Source (.c) Dateien
- Verwendung von CMake als Buildsystem
- Jede Laboraufgabe wird auf einem eigenen Branch in github oder gitlab entwickelt.
- Die Implementierung erfolgt in dem während der Vorlesung erstellten C-Projekt-Template.

Laborsetup

Die Daten von der Black Box werden über die serielle Schnittstelle (USB) gesendet und können sowohl unter Windows, Linux oder MacOS Betriebssystem bearbeitet werden.

Zum einrichten des jeweiligen Systems siehe Beschreibungen am Ende der Laboranleitung.

Implementierung erfolgt auf den Jetsons aus dem Digitalisierungslabor. Diese wird durch die Dozenten bereitgestellt und supportet. Ein Austausch des Codes erfolgt über git (github oder gitlab).

Das C-Project kann auf den Jetsons ohne devcontainer verwendet werden, da es schon ein Linux als host System ist.

Um die Entwicklung auf Windows-PCs zu ermöglichen und in dem bekannten Setup mit Visual Studio Code und cmake zu arbeiten gibt es folgende Möglichkeiten:

1. Mingw Installation auf Windows mit cmake und gcc ()
2. Windows Subsystem for Linux (WSL) mit Ubuntu installieren und dort cmake und gcc installieren.
3. Serielle Kommunikation zur Blackbox in devcontainer einrichten. Siehe Anhang

Abnahme

- Die Laboraufgaben werden einzeln abgenommen.
- Der Code wie auch die erzeugten Visualisierungen müssen erklärt werden.
- Der Code muss sauber mit Methoden, structs und unterschiedlichen .h und .c Dateien strukturiert sein. (vgl. Vorlesungsprojekt)
- Die weiteren Rahmenbedingungen müssen eingehalten werden.

Abgabe Ausarbeitung

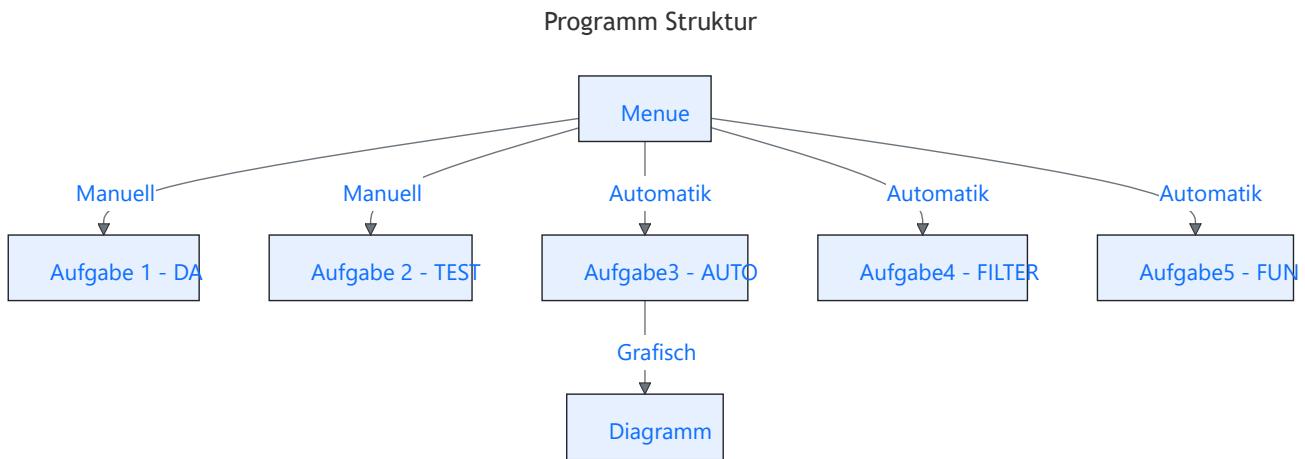
- Laden Sie auf Moodle folgende Dateien hoch
- 4x .csv Dateien (messung_1.csv, messung_2.csv, messung_3.csv, messung_4.csv)
- C Programm
- Screenshot der ausgegebenen grafik in .png oder .jpeg Format

Abgabefrist wird im Labor bekannt gegeben

Spielregeln

- Labortermine sind Pflichttermine.
- Ausarbeitung daheim ist erlaubt.
- Team Work, max 3 Teilnehmer ist erlaubt. Allerdings ist ein gesamt Kenntnisstand vom jedem Teilnehmer erfragt.

AUFGABEN:



Aufgabe 1

Plausibilität der eingelesen Werte. Modus "DA" - DatenAufnahme

Ziel: Es ist ein C Programm zu entwickeln, mit dem der von Hand gemessene Abstand und der Blackbox gelieferter Wert auf Tastendruck in einer csv-Datei **messung_x.csv** speichert.

Labortask Beschreibung

1. US-Sensor in einer festen Position zu einem Objekt abstellen
2. Den Abstand vom Sensor zur Fläche mit einem Lineal messen und über die Kommandozeile (in Zentimeter) eingeben und mit Enter bestätigen. Dieser Wert soll als realer Abstandswert in der csv-Datei gespeichert werden. (siehe nachfolgende Tabelle)
3. In der nachfolgenden Spalte den vom Sensor gelieferte Wert speichern (auch Wert in Zentimeter)
4. In der nachfolgenden Spalte die berechnete Differenz der beiden Werte eintragen
5. 20 Messungen Anfang = 10 cm, Schrittweite = 2 cm durchführen.
6. Nach 20 Punkten soll die Eingabe nicht mehr möglich sein und das Programm springt automatisch zum Menue Auswahl über.
7. Speichern der Daten in einer Datenbank (sqlite). Das Datenbank Layout ist passend gewählt. Aus der Datenbank kann eine Tabelle als csv Name: **messung_1.csv** exportiert werden.
8. Option: Plot mit reeller und den vom Sensor gelieferte Werte in Excel programmieren und graphisch ausgeben.

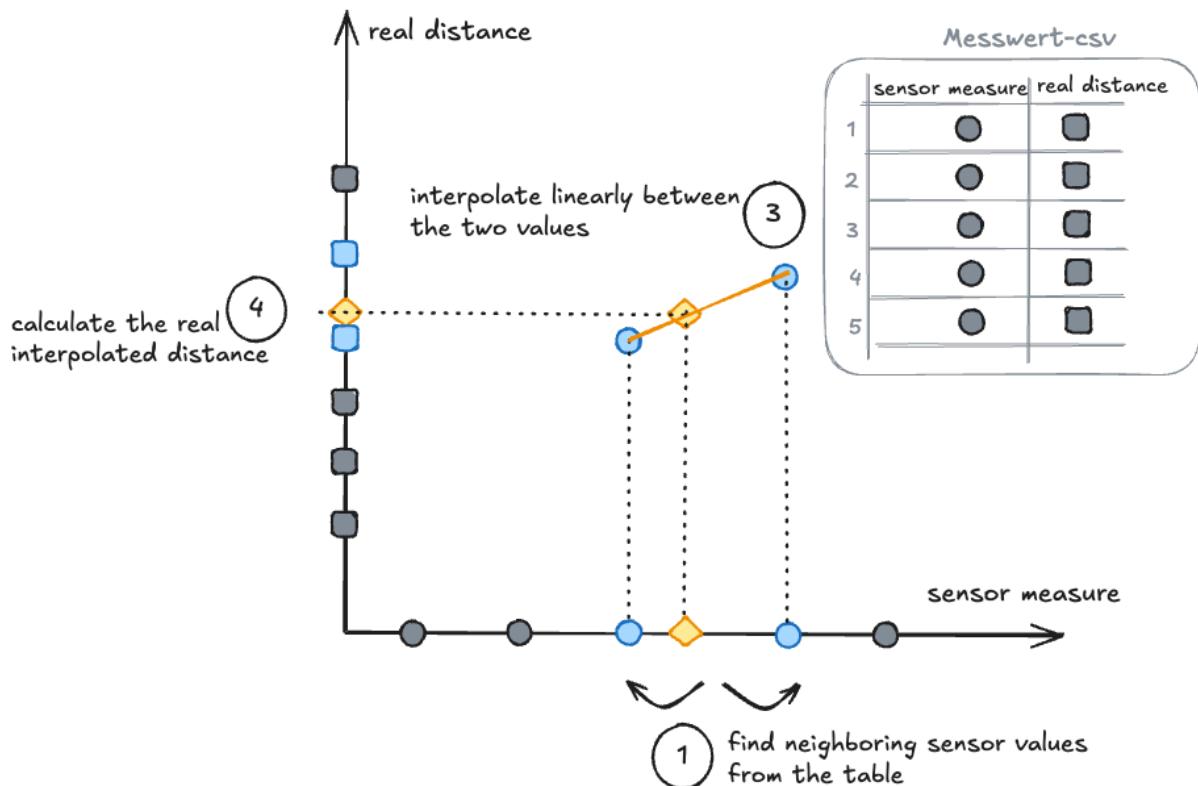
Messung Nr.	ABSTAND	Abweichung
	Reeler Wert	Vom Sensor erfasster Wert
	über Tastatureingabe	über USB eingelesen
1		
2		
...		
n-1		
n		

Aufgabe 2

Kalibrierung des Sensors mit einer Look-Up Tabelle. Modus "TEST"

- Die Ermittlung soll auf Basis der **messung_1.csv** erfolgen.
- Der Sensorwert soll in den realen Abstandswert mit einer Look-Up Tabelle umgerechnet werden.
- Durch den Tausch der Messwert-csv-Datei ist es möglich das Programm auf einen anderen Wertebereich festzulegen oder einen anderen Blackbox zu verwenden..
- Zwischen den Datenpunkten aus der Look-Up Tabelle soll linear interpoliert werden, um aus dem Sensorwert den korrekten Abstand zu berechnen.

Siehe folgende Skizze:



Look-Up Tabelle und Interpolation

Funktion der Look-Up Tabelle:

1. Der Sensorwert wird eingelesen.
2. Die nächsten zwei Sensorwerte aus der Look-Up Tabelle werden gesucht.
3. Zwischen diesen beiden Werten wird linear interpoliert, um den realen Abstand zu berechnen.
4. Der berechnete reale Abstand wird ausgegeben.

Labortask Beschreibung

1. US-Sensor in einer festen Position zu einem Objekt abstellen
2. Das messen des Abstands vom Sensor zur Fläche wird mit einem Tastendruck auf Enter ausgelöst.
Dieser Wert wird als gelesener Abstandswert (in Zentimeter) in einer Datenbank (sqlite) und dann von dort in einer csv-Datei **messung_2.csv** exportiert. (Muster siehe nachfolgende Tabelle)
3. In der nachfolgenden Spalte den interpolierten Wert eintragen (in Zentimeter)
4. Mehrere Messungen durchführen (minimum 20) . Abstand ist diesmal willkürlich ausgewählt.
5. Der Modus soll so lange laufen, bis ein spezieller Tastendruck (z.B. "q") erfolgt. das Programm springt automatisch zum Menue über
6. Option: Plot mit Soll und Ist Wert in Excel programmieren und graphisch anzeigen.

Messung Nr.	ABSTAND	Abweichung
	Vom Sensor Wert	Interpolierter Wert
	über USB eingelesen	Berechnet
1		
2		

Messung Nr.	ABSTAND	Abweichung
...		
n-1		
n		

Aufgabe 3

Automatisch messen. Modus "AUTO"

Labortask Beschreibung

1. für ca. 10 Sekunden eine Messung bzw. ein Bewegungsablauf starten
2. Zeitstempel der erfassten Werte und die Werte in einer Datenbank (sqlite) speichern und dann in einer .csv Tabelle **messung_3.csv** exportieren.
3. Visualisieren der Höhenkarte anhand der Zeitstempel - in **C** programmiert.
4. Ist die Zeit zwischen zwei Zeitstempeln immer gleich?
5. Wenn nein die max und min sowie auch die durchnits Abtast Zeit berechnen.
6. Im Windows/Linux/MacOs System mehrere Task's starten (youtube Video, Maus bewegen usw.) und gleiche Mesusung nochmal durchführen.
7. Wie verändern sich jetzt die Werte der Zeitstempel? Speichern Sie die Daten in einer in einer Datenbank (sqlite) und exportieren sie diese in einer.csv Tabelle - **messung_4.csv**

Aufgabe 4

Daten Filtern. Modus "FILTER"

- Filter Sie die in **messwert3.csv** gespeicherten Abstands Werte.
- Programmieren Sie ein Mittelwert Filter über 3 Werte z.B. $u(m)=(\text{SUMME}(y(m-1):y(m+1))/3)$
- Ploten Sie beide Werte Reihen in einem Diagramm und die dazugehörige Differenz zwischen den beiden Wertereihen in einem neuem Diagramm

Aufgabe 5

Fun: Make it a game. Modus "FUN"

Labortask Beschreibung

1. ASCII Art level durch Höhenkarte. Das C Programm gibt eine Höhenkarte auf der Kommandzeile aus.
2. U-Boot mit W - up, S - down gesteuert durch die Höhenkarte navigieren.
3. von links nach rechts fährt das Uboot automatisch.
4. Beenden Sie das Spiel wenn das U-Boot auf Grund läuft (also mit der Höhenkarte kollidiert).

Beschreibung Betriebssystem anpassen:

Daten im **L I N U X** Betriebs-System empfangen und auslesen/speichern

Vorbereitung jetsons - Falls noch nicht durch Laborpersonal vorbereitet

Mit minicom testen

Mit minicom können Sie testen, ob die Blackbox Werte liefert:

```
sudo minicom -b 115200 -D /dev/ttyUSB0  
#oder  
minicom -b 115200 -D /dev/ttyUSB0
```

Schritt-für-Schritt-Anleitung

1. Black Box an die USB Schnittstelle vom Jetson anschliessen
2. Moodle Beschreibung "ITEC-Lec04.04 Grundlagen C" aus "Einführung in C" ausführen
3. Punkt 4 (mkdir .devcontainer) aus 1.1.1 Schritt 1 "Ordnerstruktur erstellen" nicht ausführen. Kein DevContainer!
4. Kapitel 1.1.2 Schritt 2 "Devcontainer konfigurieren" nicht ausführen
5. Mit Schritt 3 (1.1.) weitermachen. Hinweis: C Code zum auslesen vom Com Port ist am Ende der Beschreibung zu finden.

Daten im **W I N D O W S** Betriebs-System empfangen und auslesen/speichern

Anhang Installationsvorgehen zur Verwendung von USB serial Kommunikation auf Windows host Systemen mit WSL 2

Important Note

Open the **PowerShell as Administrator!**

Update WSL 2 and install usbipd-win on Windows

Ensure WSL is up-to-date. Open a **PowerShell as Administrator** terminal and run:

```
wsl --update
```

The Microsoft command-line blog notes that WSL needs to be on a 5.10 kernel or newer and suggests updating before using USB/IP.

Install usbipd-win.

Use winget^{OBJ} to install the latest release. In an elevated PowerShell prompt run:

```
winget install --interactive --exact dorssel.usbipd-win
```

[Installing usbipd-win registers a service on Windows that shares local USB devices to WSL.](#)

List available USB devices.

From an elevated command prompt run:

```
usbipd list
```

This command prints each device's **BUSID**, vendor/product IDs and current state. Note the **BUSID** corresponding to the serial adapter you want to share.

Connected:

BUSID	VID:PID	DEVICE	STATE
1-8	5986:2115	Integrated Camera	Not shared
1-9	06cb:00be	Synaptics UWP WBDI SGX	Not shared
1-14	8087:0026	Intel(R) Wireless Bluetooth(R)	Not shared
4-3	046a:0114	USB-Eingabegerät	Not shared
4-4	10c4:ea60	CP2102 USB to UART Bridge Controller	Not shared

Persisted:

GUID

CP2102 USB to UART Bridge Controller is the description of the serial adapter we need. Its **BUSID** is **4-4**.

Bind and attach the device.

First, make the device sharable with `usbipd bind --busid=<BUSID>`.

```
usbipd bind --busid <BUSID>
```

Then attach it to WSL using:

Open WSL terminal

To attach the device to WSL, open a WSL terminal such as Ubuntu from:

Startmenue → Ubuntu xx.xx

```
usbipd attach --busid <BUSID> --wsl
```

The usbipd documentation explains that bind is persistent, whereas attach connects the device to the running WSL VM.

Install USB/IP client tools inside WSL

In order for WSL to communicate with the usbind service, install the Linux USB/IP user-space tools:

```
sudo apt update
# Install the appropriate linux-tools package for your kernel and the hwdatabase database
sudo apt install linux-tools-$(uname -r) hwdatabase usbinodev
# Register the usbind client binary if your distribution does not do so automatically
sudo update-alternatives --install /usr/local/bin/usbind usbind \
    /usr/lib/linux-tools-$(uname -r)/usbind 20
```

Microsoft's guidance demonstrates installing `linux-tools` and `hwdatabase`, then registering `usbind` with update-alternatives [1]. The usbinodev package provides the `lsusb` utility used later.

After attaching the device from Windows, verify that WSL can see it:

```
lsusb
```

You should see an entry corresponding to your serial adapter. Depending on the driver, it will also create a character device such as `/dev/ttyUSB0` or `/dev/ttyACM0`. You can find it with:

```
ls -l /dev/ttyUSB* /dev/ttyACM* 2>/dev/null
```

Im VSCode Terminal folgende Befehle ausführen: RUN apt-get update RUN apt-get install -y sqlite3 libsqlite3-dev ninja-build USER root RUN apt-get install -y -no-install-recommends usbinodev minicom RUN usermod -aG dialout vscode USER vscode WORKDIR /workspaces

This configuration installs `usbinodev` and `minicom` so you can inspect USB buses and open serial. Adding the vscode user to the dialout group avoids running commands as root.

If you know the exact device file (`/dev/ttyUSB0` or similar) you may instead map only that device using “`--device=/dev/ttyUSB0`” for finer control. The example above maps all USB devices.

Anhang: Daten im **Windows** Betriebs-System empfangen und auslesen/speichern

Daten im **Mac OS** Betriebs-System empfangen und auslesen/speichern

Hier ist der gleiche C Code anwendbar.

```
C Code für Linux MacOS und Windows(im WSL2)
```

