CanSat Raspberry Pi Pico Arbeitsbuch

Inhaltsübersicht


CANSAT

Einführung

Die folgenden Übungen wurden entwickelt, um Sie in einige der Elektronik- und Programmierkenntnisse einzuführen, die für die Durchführung der CanSat-Hauptmission erforderlich sind.

Dieses Dokument ist auf den Punkt gebracht, ohne ein vollständiger technischer Leitfaden zu sein.
Wo nötig, werden Verweise auf vollständige technische Leitfäden gegeben.

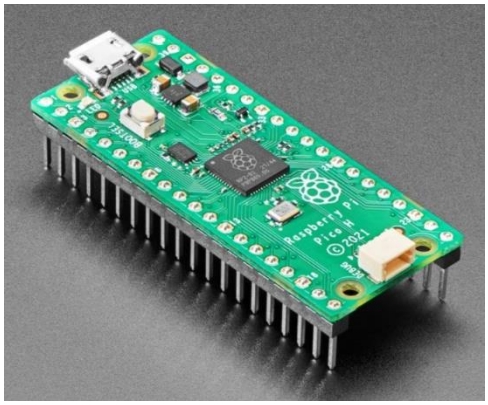
Der Schwierigkeitsgrad der Übungen ist progressiv und beginnt mit der Verdrahtung und Programmierung ohne Lötarbeiten.

Wenn Sie noch nie in Python programmiert haben, empfehlen wir Ihnen, sich den Abschnitt Python lernen anzusehen

Der CanSat-Bausatz

Das Kit besteht aus Standard-Hardware, die günstig und einfach online zu kaufen ist. Dadurch können die Teams defekte Komponenten leicht ersetzen und auch Unterstützung und Ideen aus der Fülle von Online-Tutorials und technischen Ressourcen im Zusammenhang mit Raspberry Pi Pico und MicroPython finden. Das Kit, das wir in den Labors verwenden, enthält die folgenden Teile:

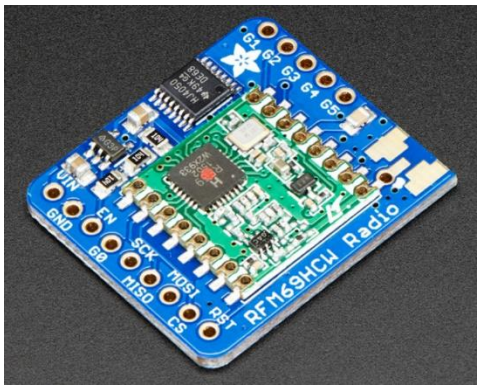
2x Raspberry Pi Pico mit Headern



Der Pico wird mit oder ohne vorgelötete Header verkauft. Wir geben Ihnen 2 Pico mit Headern, um Ihnen den Einstieg zu erleichtern und sich frühzeitig auf die Programmierung zu konzentrieren.

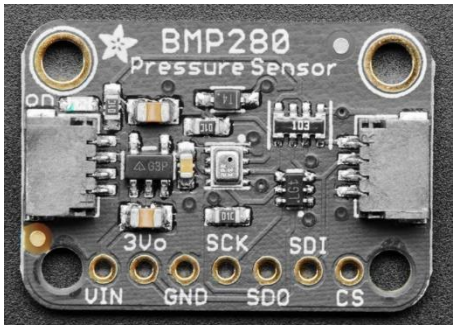
https://www.adafruit.com/product/5525

2x Funksprechgeräte RFM69HCW



https://www.adafruit.com/product/3071

1x BMP280 Druck-/Temperatursensor



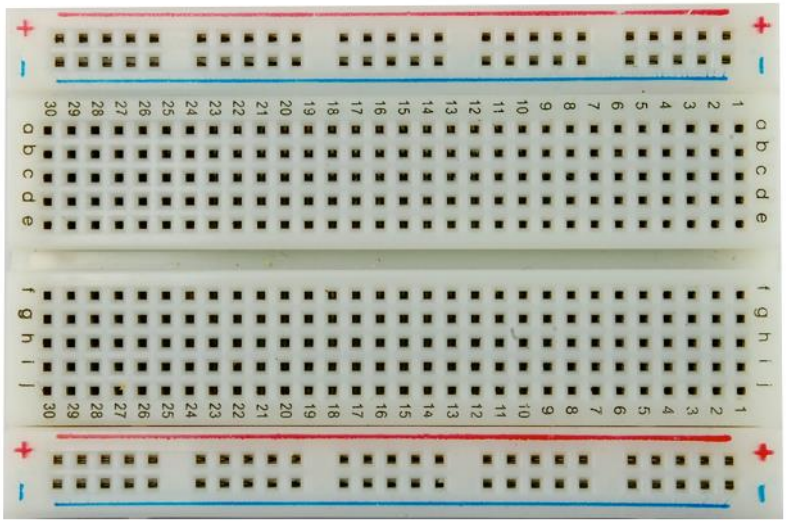
https://www.adafruit.com/product/2651

1x TMP36-Sensor

Alternativ zum BMP280 ist der TMP36 ein einfacher analoger Temperatursensor, der eine Spannung ausgibt, die von der Umgebungstemperatur um den Sensor abhängt.

https://learn.adafruit.com/tmp36-temperature-sensor

1x Breadboard für Prototyping-Schaltungen



Ein Breadboard ist eine Konstruktionsunterlage, mit der semipermanente Prototypen elektronischer Schaltungen ohne Löten gebaut werden können.

1x USB-Kabel

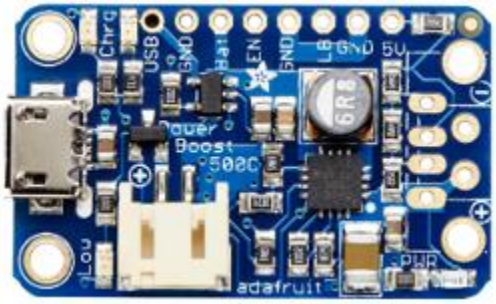
Kabel zum Anschluss des Raspberry Pico an einen Computer, um ihn zu programmieren oder den Lipo-Lithium-Akku zu laden.

1x Lithium-Batterie

3.7V 1300mAh Batterie zur Stromversorgung des Raspberry Pico im CanSat, ohne USB-Kabel.

https://cdn-shop.adafruit.com/datasheets/Li-poly+603562-1300mAh.pdf

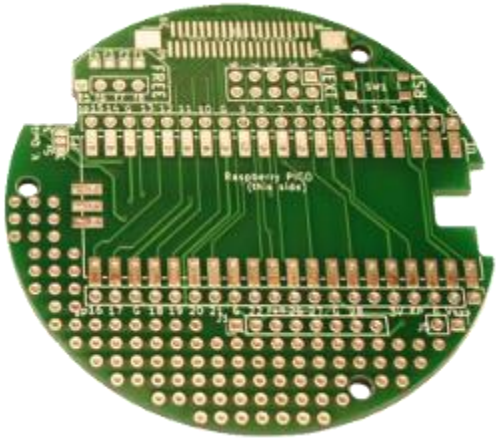
1x 5-Volt-Konverter und Lithium-Batterieladegerät



Platine, die den Strom der Lithiumbatterie in eine für den Raspberry Pico geeignete 5-Volt-Stromquelle umwandelt. Es kann die Lithium-Batterie aufladen, wenn das Kit über USB mit Strom versorgt wird.

https://www.adafruit.com/product/1944

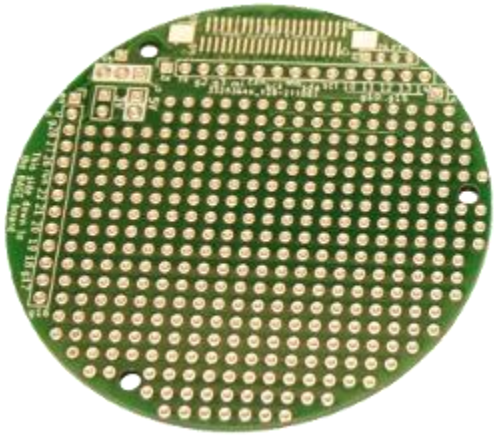
1x Cansat-Basisplatine für Pico



Nachdem Sie die Verdrahtung Ihrer Komponenten und der zugehörigen Software auf dem Breadboard getestet haben, können Sie den Raspberry Pico, die 5V-Wandlerplatine und den Funksender auf diese Platine löten. Der Durchmesser der Platine entspricht dem maximalen CanSat-Durchmesser. Die BMP280-Platine kann mit dem mitgelieferten JST-Kabel eingesteckt werden.

https://shop.mchobby.be/fr/pico-rp2040/2275-carte-de-base-cansat-pour-pico-3232100022751.html

1x Cansat-Verlängerungsplatine



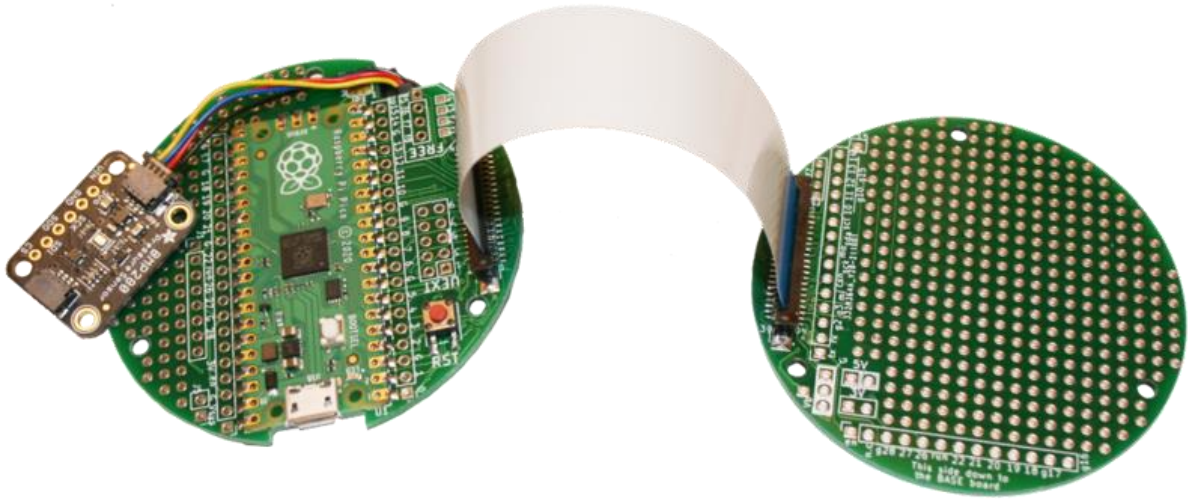
Diese CanSat-Erweiterungsplatine ist nützlich, um Komponenten hinzuzufügen, die für die sekundäre Mission benötigt werden, wie ein GPS oder andere Sensoren.

https://shop.mchobby.be/fr/pico-rp2040/2272-carte-de-prototypage-cansat-pour-pico-3232100022720.html

1x FPC-Farbband

Dieses Band wird verwendet, um die Basisplatine mit der Erweiterungsplatine zu verbinden.

https://shop.mchobby.be/en/wire-cables/2278-fpc-ribbon-40-pos-p05-1016mm-3232100022782.html



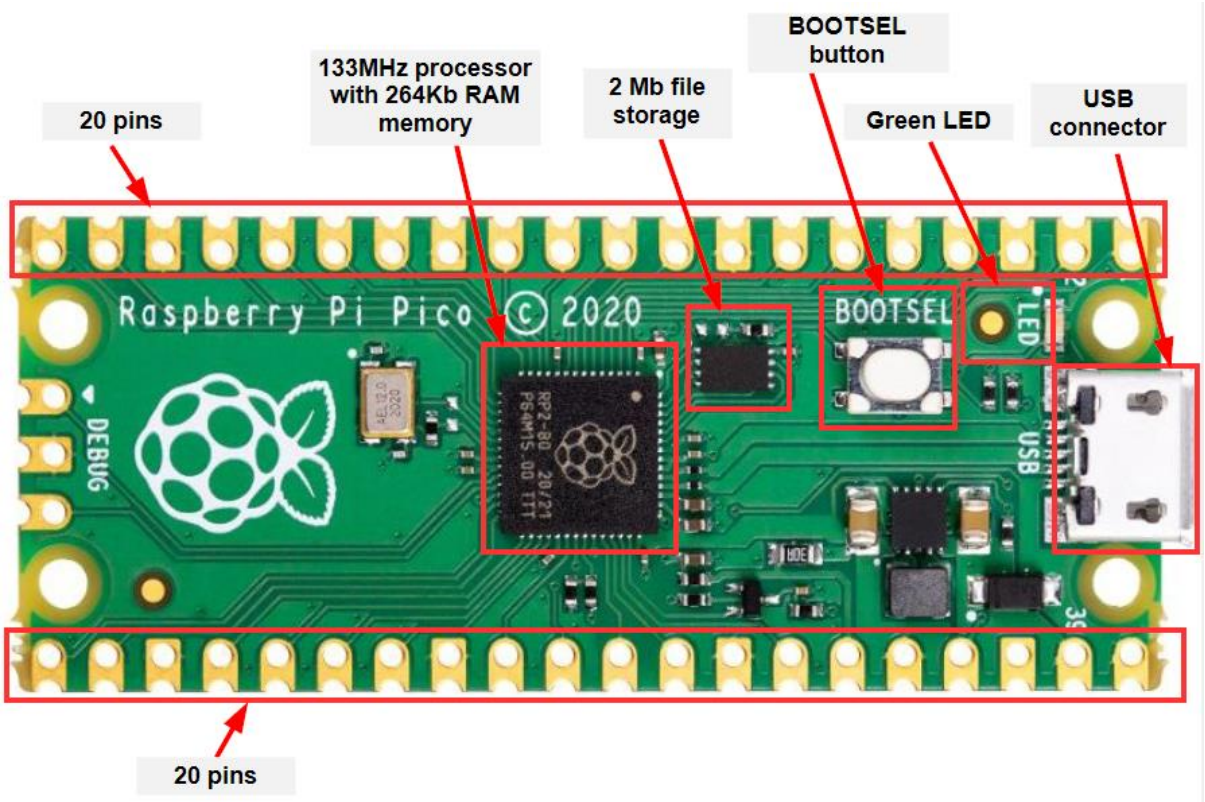
ESERO stellt Ihnen einen kompletten Bausatz kostenlos zur Verfügung, aber die verschiedenen Komponenten können separat bei McHobby gekauft werden

Treffen Sie Raspberry Pi Pico

Ein Raspberry Pi Pico ist ein kostengünstiges Mikrocontroller-Gerät. Mikrocontroller sind winzige Computer, aber sie haben nur einen kleinen Dateispeicher (im Gegensatz zu einer Festplatte in einem typischen Computer) und keine Peripheriegeräte, die Sie anschließen können (z. B. Tastaturen oder Monitore).

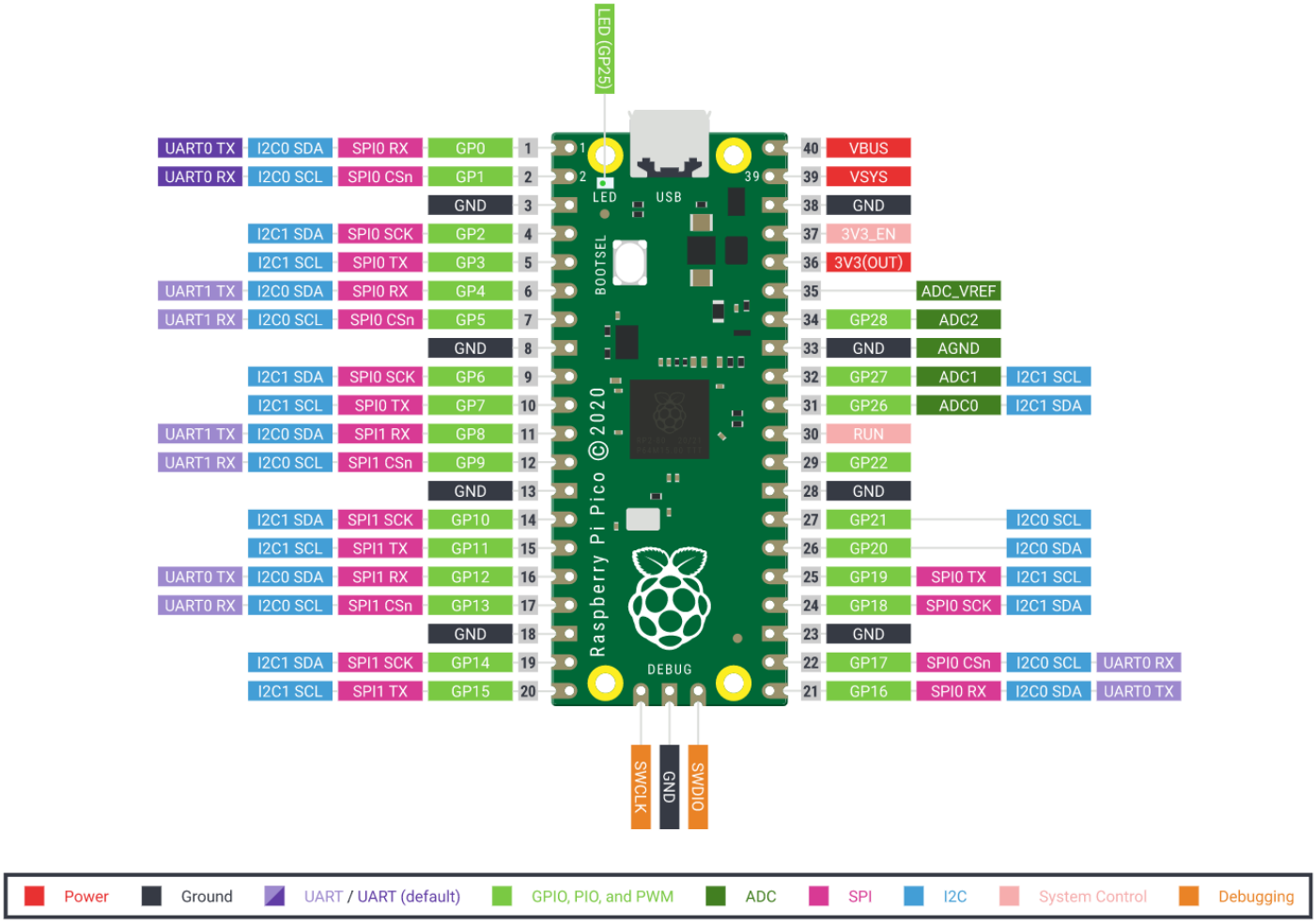
Ein Raspberry Pi Pico hat

1- Ein 133-MHz-Prozessor mit 264 Kilobytes RAM-Speicher
2- 2 Megabyte Dateispeicher
3- Eine BOOTSEL-Taste zur Installation von MycroPython auf dem Pico
4- Eine grüne LED
5- Ein USB-Anschluss zur Stromversorgung des Pico und zur Übertragung von Software oder Daten.
6- 2 x 20 Pins für die Stromversorgung des Pico sowie für die Steuerung und den Empfang von Eingaben von einer Vielzahl elektronischer Geräte.



Jeder der 40 Stifte hat eine eigene Funktion.

Wenn Sie die Pin-Nummern für einen Raspberry Pi Pico wissen müssen, können Sie das folgende Diagramm oder diese interaktive Website verwenden



Wenn Sie mit dem Pico arbeiten, brauchen Sie nur mit zu arbeiten:

1- Rote und schwarze Stifte = Stifte für die Strom- und Erdungsverbindung
2- Violette Pins = Pins, die mit der UART-Kommunikation zusammenhängen
3- Rosa Pins = Pins, die sich auf das SPI-Kommunikationsprotokoll beziehen
4- Blaue Pins = Pins im Zusammenhang mit dem I2C-Kommunikationsprotokoll

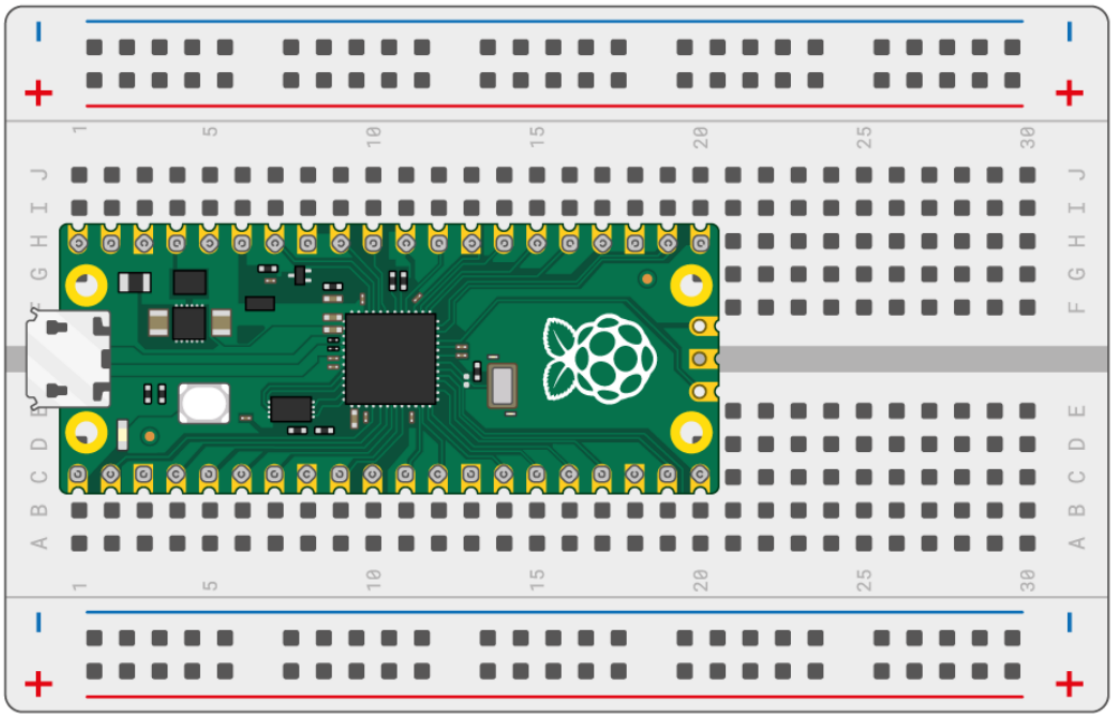
Es gibt mehrere Möglichkeiten, den Pico mit Strom zu versorgen, entweder über

1- das Micro-USB-Kabel (in der Entwicklungsphase)
2- den mitgelieferten 5-Volt-Wandler und die Lithium-Batterie (nach Abschluss der Entwicklung)

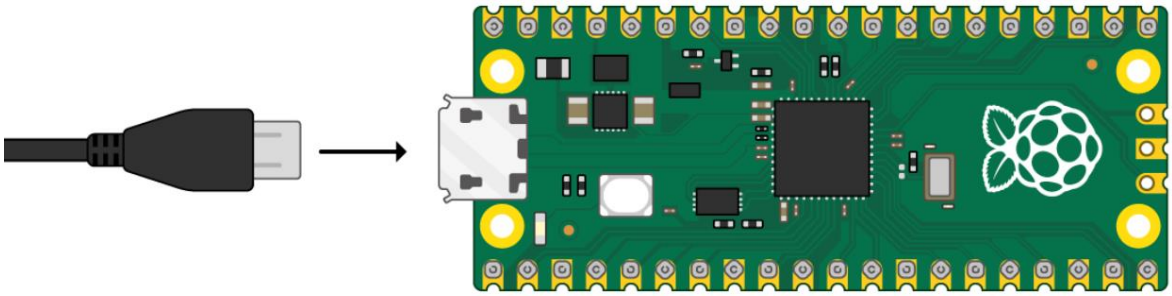
Verbinden Sie Ihren Raspberry Pi Pico mit Ihrem Computer

In diesem Abschnitt werden Sie einen Raspberry Pi Pico an einen anderen Computer anschließen und lernen, wie man ihn mit MicroPython programmiert.

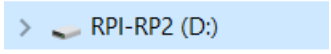
Stecken Sie Ihren Raspberry Pi Pico fest auf das mitgelieferte Breadboard, wie in der folgenden Abbildung gezeigt. Platzieren Sie ihn so, dass er durch die Schlucht in der Mitte des Breadboards getrennt ist.



Stecken Sie das mitgelieferte Micro-USB-Kabel in den Anschluss an der linken Seite des Pico.



Ihr Pico sollte wie ein USB-Speicher in Ihrem Dateisystem erscheinen.

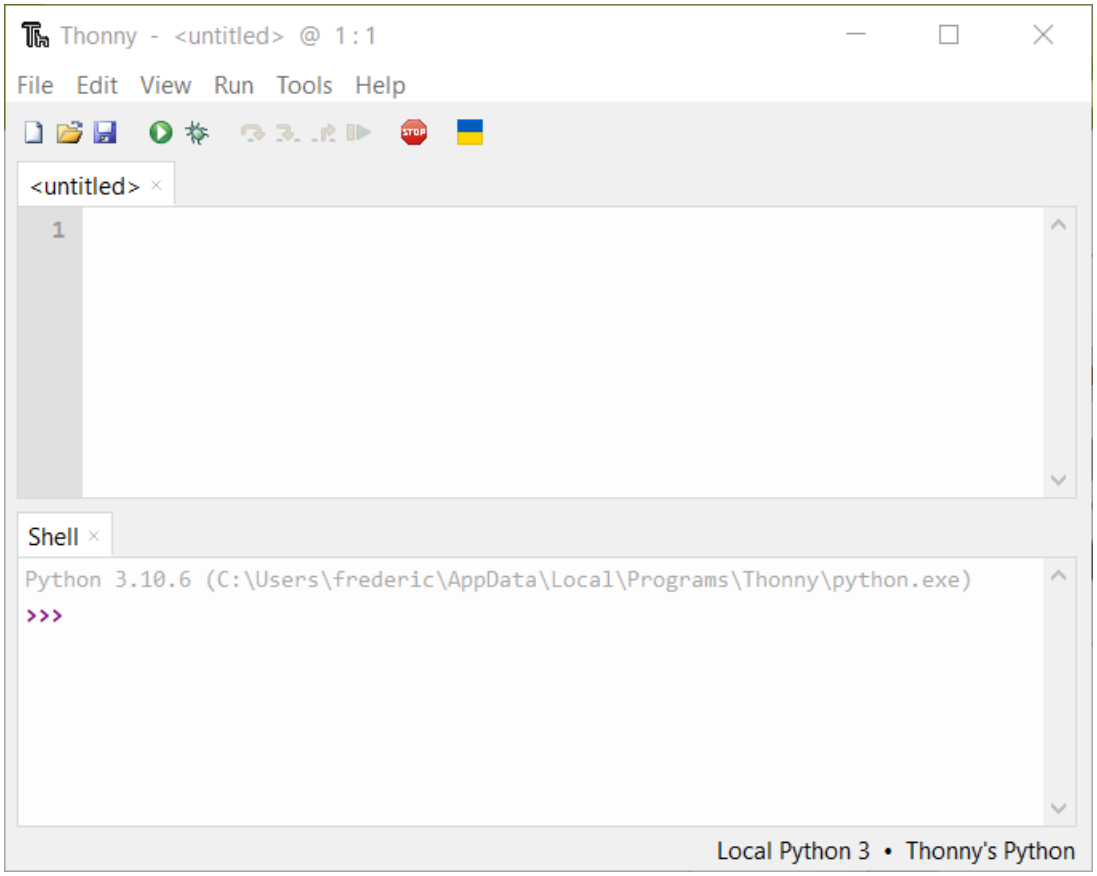


Installieren Sie Thonny auf Ihrem Computer

Python ist eine Allzwecksprache, die in einer Vielzahl von Anwendungen eingesetzt wird (Datenwissenschaften, künstliche Intelligenz, Statistik, ...), während MicroPython speziell für Mikrocontroller wie den in unserem Projekt verwendeten Raspberry Pi Pico entwickelt wurde.

Um unseren Code in der Sprache MicroPython zu bearbeiten, auszuführen und zu debuggen, werden wir eine integrierte Entwicklungsumgebung (IDE) namens Thonny installieren, die unter https://thonny.org erhältlich ist.

Öffnen Sie Thonny von Ihrem Anwendungsstartprogramm aus. Es sollte in etwa so aussehen:



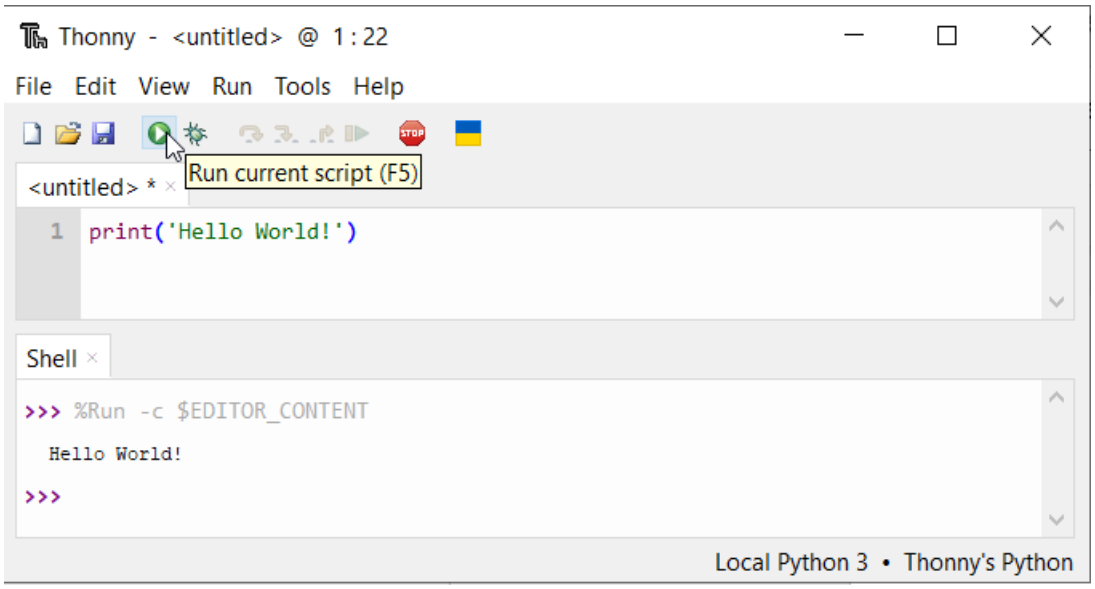
Irgendwann wird Thonny zweimal geöffnet werden müssen.

- Rufen Sie das Menü "Extras -> Optionen...->Allgemein" auf.
- Deaktivieren Sie das Kontrollkästchen "Nur eine Thonny-Instanz zulassen".
- Drücken Sie "OK".

Sie können Thonny verwenden, um Standard-Python-Code zu schreiben. Geben Sie Folgendes in das obere Fenster ein und klicken Sie dann auf die Schaltfläche Ausführen.

```
print('Hallo Welt!')
```

Das Ergebnis wird im Fenster "Shell" angezeigt



Installieren Sie MicroPython auf Ihrem Pico

Ihr neuer Raspberry Pi Pico benötigt MicroPython, um Ihre Software auszuführen.

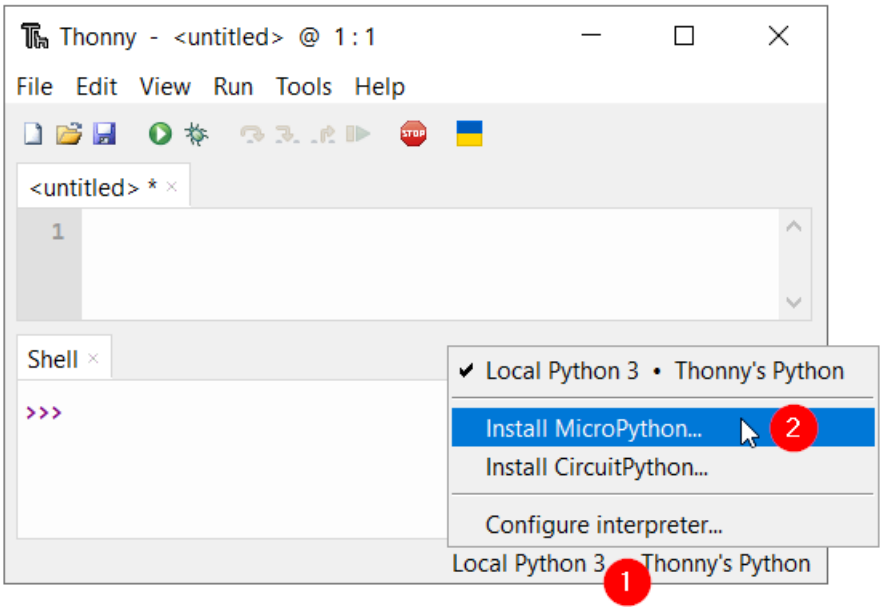
MicroPython ist eine weitgehend mit Python kompatible Programmiersprache, die für die Ausführung auf einem Mikrocontroller wie dem Raspberry Pico optimiert ist.
Die Dokumentation ist auf der offiziellen MicroPython-Website zu finden.

Ziehen Sie zunächst das Micro-USB-Kabel von Ihrem Computer ab, lassen Sie es aber an Ihrem Pico angeschlossen.

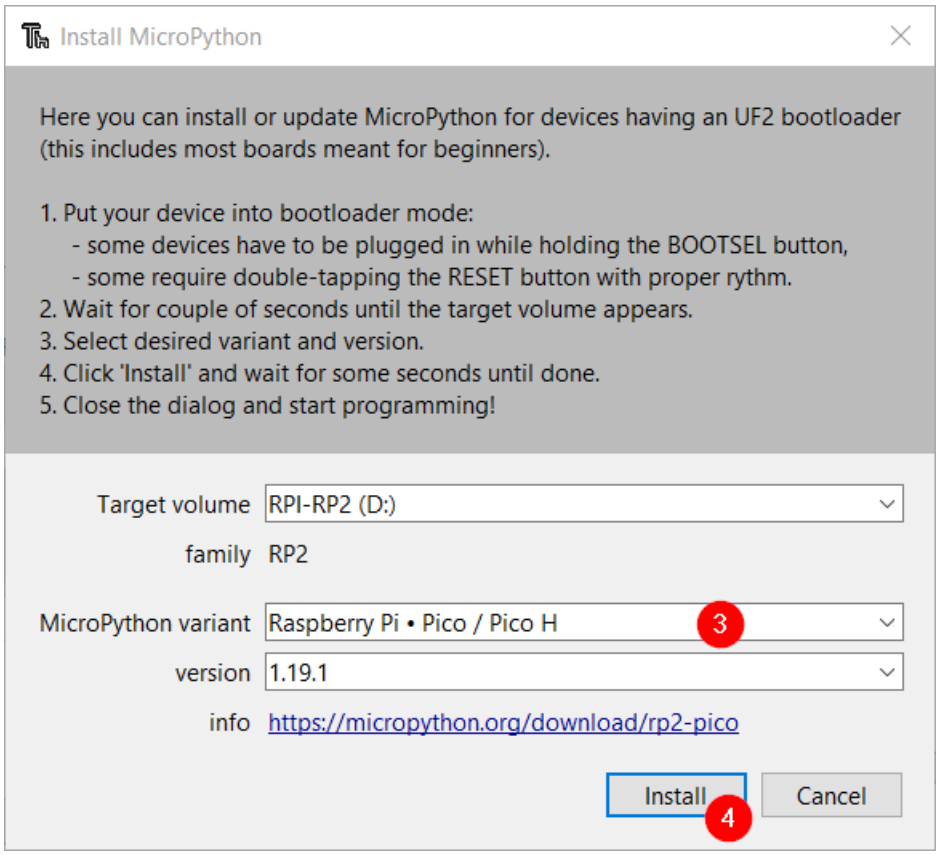
Drücken Sie die BOOTSEL-Taste und halten Sie sie gedrückt, während Sie das andere Ende des Micro-USB-Kabels an Ihren Computer anschließen.

In der rechten unteren Ecke des Thonny-Fensters sehen Sie die Version von Python, die Sie gerade verwenden.

Klicken Sie mit der linken Maustaste auf die Python-Version und wählen Sie "MicroPython installieren...".



Ein Dialogfeld wird angezeigt, um die MicroPython-Firmware auf Ihrem Pico zu installieren. Wählen Sie die richtige MicroPython-Variante und klicken Sie auf die Schaltfläche Installieren.



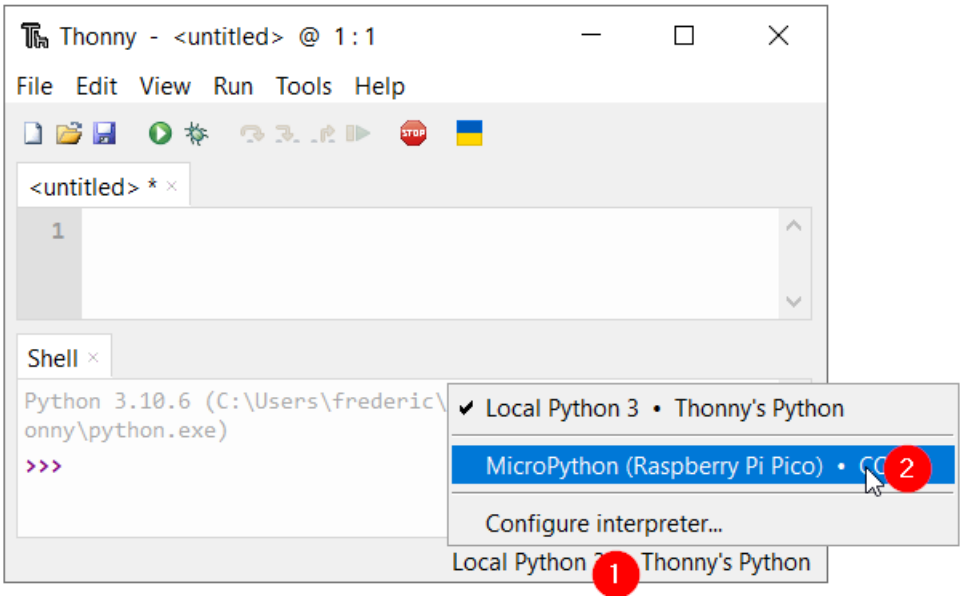
Warten Sie, bis die Installation abgeschlossen ist, und klicken Sie auf die Schaltfläche Schließen.

Sie müssen die Firmware nicht jedes Mal aktualisieren, wenn Sie Ihren Pico verwenden.

Beim nächsten Mal können Sie das Gerät einfach an Ihren Computer anschließen, ohne die BOOTSEL-Taste zu drücken.

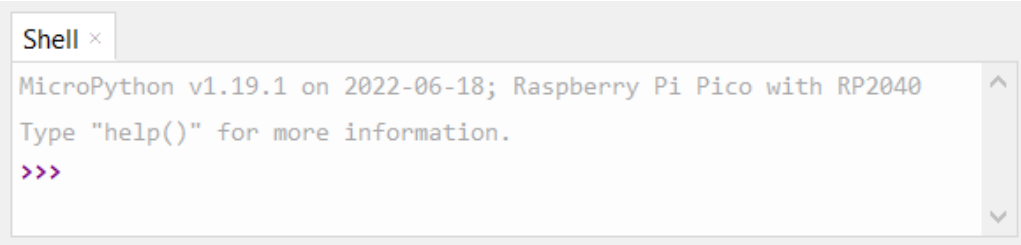
Führen Sie Ihren ersten MicroPython-Code auf Ihrem Pico aus

Stellen Sie sicher, dass Ihr Raspberry Pi Pico noch an Ihren Computer angeschlossen ist. Wählen Sie den MicroPython (Raspberry Pi Pico) Interpreter unten rechts aus.



Sehen Sie sich das Shell-Panel am unteren Rand des Thonny-Editors an.

Sie sollten etwa so aussehen:



Thonny ist nun in der Lage, mit dem Pico über die REPL (read-eval-print loop) zu kommunizieren, was es Ihnen ermöglicht, Python-Code in die Shell einzugeben und das Ergebnis direkt zu sehen.

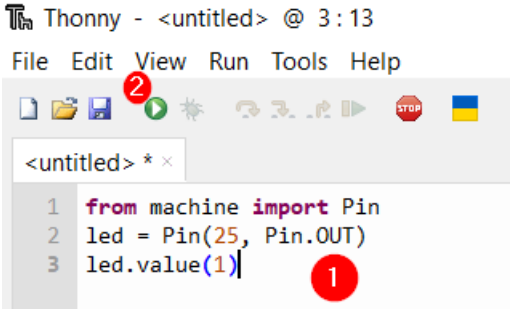
MicroPython fügt hardwarespezifische Module hinzu, wie z.B. `Machine`, die Sie zur Programmierung Ihres Raspberry Pi Pico verwenden können.

Erstellen wir ein `machine.`Pin-Objekt, um mit der Onboard-LED zu spielen, auf die über GPIO-Pin 25 zugegriffen werden kann.

Wenn Sie den Wert der LED auf `1` setzen, schaltet sich die integrierte LED ein.

Geben Sie den folgenden Code ein und tippen Sie nach jeder Zeile auf Enter.

```
from machine import Pin
led = Pin(25, Pin.OUT)
led.value(1)
```



Nachdem Sie die grüne "Run"-Taste gedrückt haben, sollte die Onboard-LED aufleuchten.

Ändern Sie den Code und setzen Sie den LED-Wert auf 0, um die LED auszuschalten.

```
led.value(0)
```

Schalten Sie die LED so oft ein und aus, wie Sie möchten.

Tipp: Sie können den Pfeil nach oben auf der Tastatur verwenden, um schnell auf die vorherigen Zeilen zuzugreifen.

Wir haben gesagt, dass der Onboard-LEP an GPIO-Pin 25 angeschlossen ist, aber was ist GPIO?

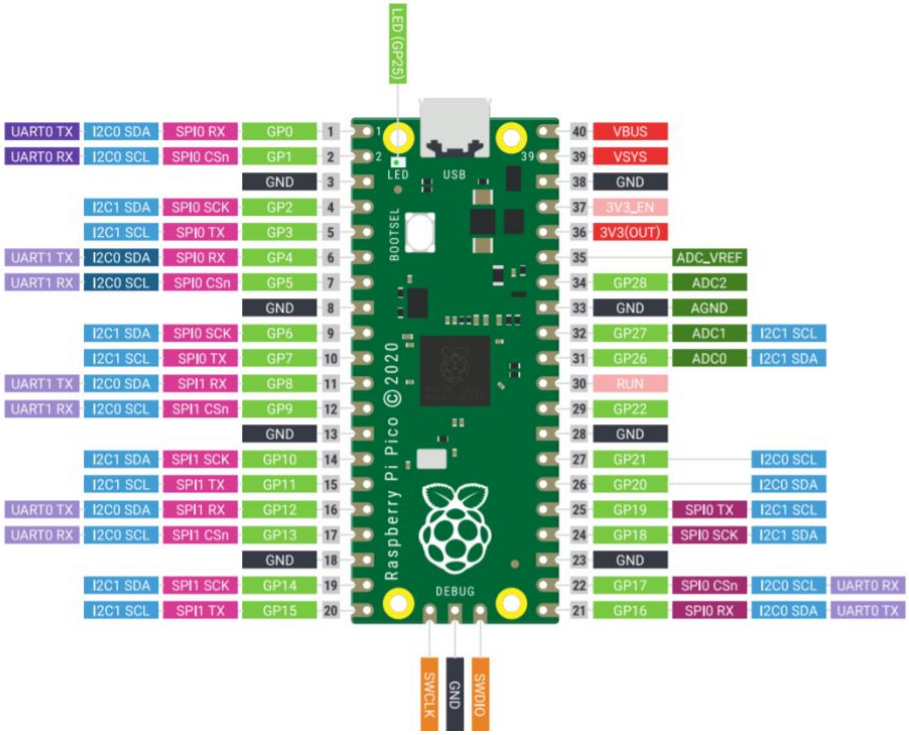
Allzweck-Eingang/Ausgang (GPIO)

Mit den GPIO-Pins des Raspberry Pi können externe Spannungen von der Software ausgelesen werden, und sie ermöglichen auch die Einstellung externer Spannungen durch die Software. GPIOs können entweder digital oder analog sein. Analoge Pins können als digitale Pins eingestellt werden, aber digitale Pins bleiben digital. Analoge Pins sind nur Eingangspins.
Bei unserem 3,3-V-Raspberry Pi wird jede Spannung unter 2,5 V als "Falsch" und umgekehrt jede Spannung über 2,5 V als "Wahr" interpretiert (bis zu 3,3 V). Ähnlich verhält es sich mit den Ausgangssignalen. Ein "True"-Ausgang setzt die Spannung des Pins auf 3,3V und der "False"-Ausgang setzt die Spannung des Pins auf 0V.

GPIO-Pins können entweder als Eingangs- oder als Ausgangs-Ports verwendet werden und werden per Software eingestellt.

Der Pi Pico hat 28 GPIO-Ports, wie in den grünen Kästen im folgenden Diagramm zu sehen ist. Viele Pins sind multifunktional und können auch für andere Schnittstellen (UART, SPI,

I2C) verwendet werden. Diese werden durch die mehrfarbigen Kästchen neben den grünen Kästchen im Diagramm dargestellt. Der folgende Link enthält die Pinbelegung: https://datasheets.raspberrypi.org/pico/Pico-R3-A4-Pinout.pdf



https://datasheets.raspberrypi.org/pico/Pico-R3-A4-Pinout.pdf

Typische GPIO CanSat Verwendungen:
Eingänge: Ein/Aus-basierte Sensoren, Schalter, Tasten, Einsatzsensoren
Ausgänge: Status-LEDs, einfache Servos (besser PWM), Einschalten von Sensoren, Zurücksetzen von Sensoren, die durch andere Signale verbunden sind

Wenn Sie ein längeres Programm schreiben wollen, ist es am besten, es in einer Datei zu speichern. Dies werden Sie im nächsten Abschnitt tun.

Schreiben Sie Ihre erste microPython-Datei zur Steuerung der Onboard-LED

Die Shell ist nützlich, um schnelle Befehle auszuprobieren. Es ist jedoch besser, längere Programme in einer Datei abzulegen.

Thonny kann MicroPython-Programme direkt auf Ihrem Raspberry Pi Pico speichern und ausführen.

In diesem Schritt werden Sie ein MicroPython-Programm erstellen, um die Onboard-LED in einer Schleife über GPIO-Pins ein- und auszublinken

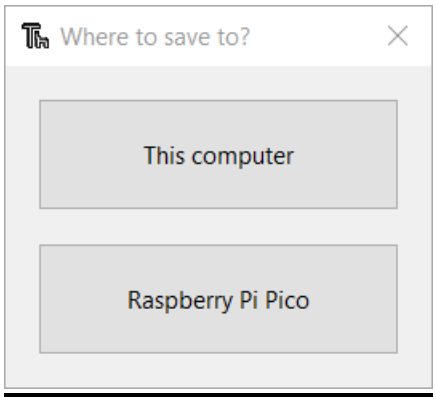
Gehen Sie zurück zu Thonny und klicken Sie im Hauptfenster des Editors auf.

Geben Sie den folgenden Code ein, um die LED zu schalten.

```
# importiere die notwendigen, bereits existierenden Bibliotheken
from machine import Pin
# Deklarieren Sie eine Variable namens "led", verknüpfen Sie sie mit Pin Nummer 25
und definieren Sie sie als Ausgang
led = Pin(25, Pin. OUT)
# Ändern Sie den Zustand der LED von led.value(0) zu led.value(1) und umgekehrt.
led.toggle()
```

Die vollständige Dokumentation der "Pin"-Bibliothek finden Sie in der offiziellen MicroPython-Dokumentation

Klicken Sie auf die Schaltfläche **Speichern**  , um Ihren Code zu speichern. Daraufhin wird der folgende Bildschirm angezeigt:



Wählen Sie "Raspberry Pi Pico" und nennen Sie die Datei "blink.py".

Tipp: Sie müssen die Dateierweiterung . py eingeben, damit Thonny die Datei als Python-Datei erkennt. Thonny kann Ihr Programm auf Ihrem Raspberry Pi Pico speichern und es ausführen.

Sie sollten sehen, dass die integrierte LED jedes Mal, wenn Sie auf die Schaltfläche Ausführen klicken, zwischen an und aus wechselt.

Aber was ist, wenn Sie die LED blinken sehen wollen, ohne immer wieder auf die Schaltfläche Ausführen klicken zu müssen?

Um dies zu erreichen, verwenden wir eine "while"-Schleife und die "sleep"-Funktion

Achten Sie darauf, den Code mit 4 Leerzeichen innerhalb der while-Schleife einzurücken, damit MicroPython weiß, dass diese Zeilen Teil der while-Schleife sind.

Aktualisieren Sie Ihren Code so, dass er wie folgt aussieht:

```
# importiere die notwendigen, bereits existierenden Bibliotheken
from machine import Pin
from time import sleep
# Deklarieren Sie eine Variable namens "led", verknüpfen Sie sie mit Pin Nummer 25
und definieren Sie sie als Ausgang
led = Pin(25, Pin. OUT)

# Achten Sie darauf, den Code innerhalb der while-Schleife mit 4 Leerzeichen
einzurücken, damit MicroPython weiß, welche Zeilen Teil der while-Schleife sind.
```

```
while True:
    # Ändern Sie den Zustand der LED von led.value(0) zu led.value(1) und
umgekehrt.
    led. toggle()
    # Anhalten der Programmausführung für eine halbe Sekunde
    sleep(0,5)
```

Sie können auch das Timer-Modul (erste Zeile unten) verwenden, um einen Timer zu setzen, der eine Funktion in regelmäßigen Abständen ausführt. Aktualisieren Sie Ihren Code so, dass er wie folgt aussieht:

```
# importiere notwendige, bereits existierende Bibliotheken
from machine import Pin, Timer

# Deklarieren Sie eine Variable namens "led", verknüpfen Sie sie mit Pin Nummer 25
und definieren Sie sie als Ausgang
led = Pin(25, Pin. OUT)
# Deklarieren Sie eine Timer-Variable, um das Timing von Perioden und Ereignissen
zu behandeln
timer = Timer()

# Deklariere eine Funktion namens "blink", die den Zustand der LED umschaltet
def blink(timer):
    # # Ändern des LED-Status von led.value(0) zu led.value(1) und umgekehrt
    led.toggle()

# Konfigurieren Sie den Timer so, dass er die vordefinierte Blinkfunktion alle 2,5
Sekunden aufruft.
timer.init(freq=2.5, mode=Timer. PERIODIC, callback=blink)
```

Die vollständige Dokumentation der "Timer"-Bibliothek finden Sie in der offiziellen MicroPython-Dokumentation

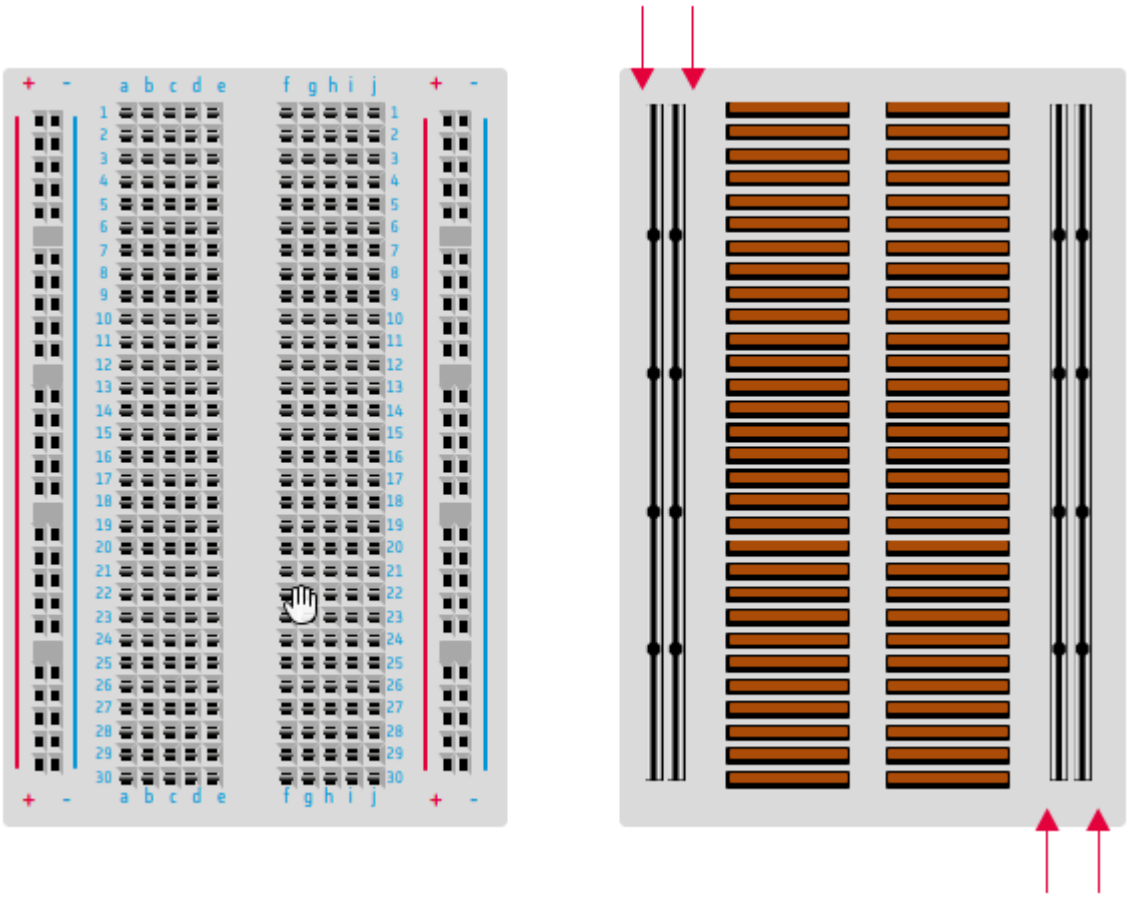
Klicken Sie auf **Ausführen** und Ihr Programm blinkt die LED an und aus, bis Sie auf die Schaltfläche **Stopp** klicken.

Tipp: Wenn Sie Ihren Pico aus- und wieder einstecken, müssen Sie erneut die Schaltfläche **Ausführen** drücken, um das Programm zu starten. Wenn du aber dein Programm "**main.py**" auf deinem Pico nennst, wird es automatisch ausgeführt, wenn der Pico eingeschaltet wird.

Software-Entwicklung & Verdrahtungstest

Treffen Sie ein Breadboard

Während du die Grundlagen von Pico und Sensoren lernst, ist es am besten, ein lötfreies Breadboard zu verwenden, da Fehler, die du beim Aufbau deiner Schaltung machst, leicht geändert werden können.
Ein Breadboard ist ein einfaches Werkzeug, mit dem man elektrische Komponenten miteinander verdrahten kann.



Die Stifte der elektrischen Bauteile können in die Klemmen auf der Platine gesteckt werden. In der Mitte sind die Reihen horizontal verbunden. Das bedeutet zum Beispiel, dass die beiden Stifte eines Widerstands in verschiedenen Reihen platziert werden sollten, da er sonst mit sich selbst einen geschlossenen Stromkreis bildet.

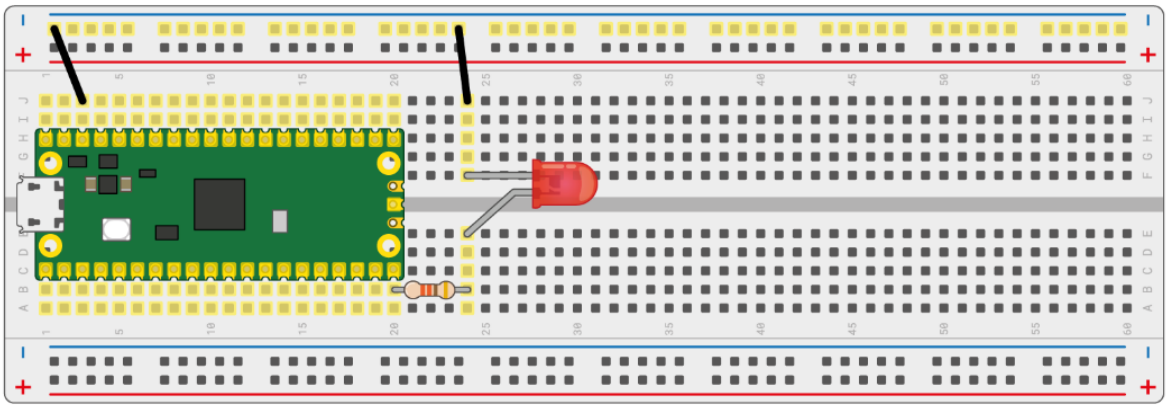
Es ist sehr wichtig, dass Sie eine Skizze Ihrer Schaltung anfertigen, bevor Sie die Schaltung anschließen und mit Strom versorgen, da Sie sonst Gefahr laufen, die Bauteile zu zerstören. Die äußeren Spalten der Platine sind nicht in Reihen, sondern in Spalten verbunden. In der Regel werden sie für die Erdungs- und Spannungsanschlüsse verwendet.

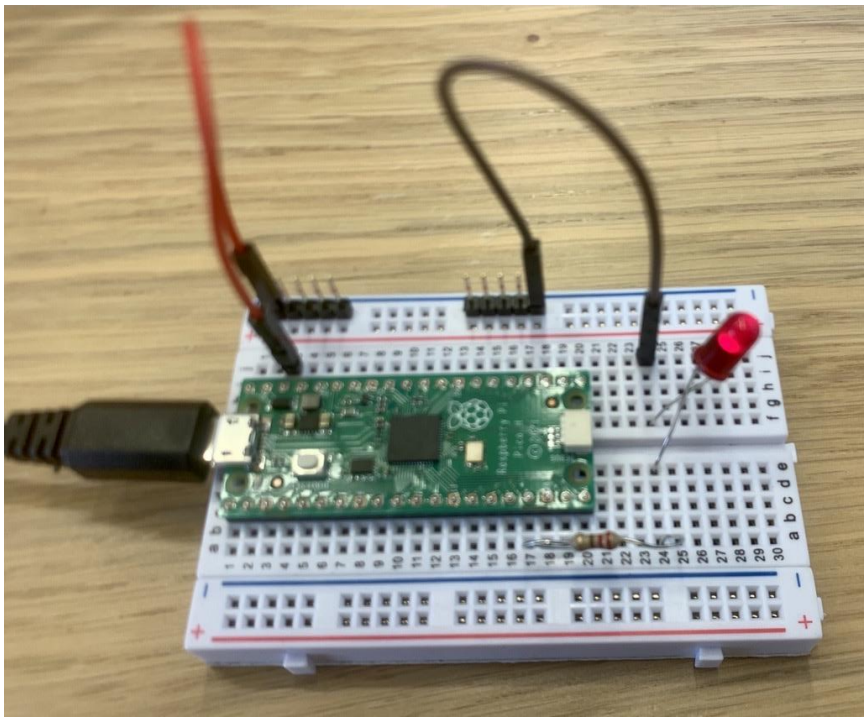
Blinken einer LED auf dem Breadboard

Sie werden lernen, wie man eine externe LED steuert.

Verwenden Sie einen 220-Ohm-Widerstand, eine LED, einige Steckbrücken und ein paar Stiftleisten, um Ihren Raspberry Pi Pico auf Ihrem Breadboard anzuschließen, wie in der Abbildung unten gezeigt.

Beachten Sie, wie die LED auf der einen Seite an GPIO 15 angeschlossen ist (der letzte links unten, wie Sie im Pico-Belegungsplan sehen können) und auf der anderen Seite an den Masse-Pin des Pico.





Verwenden Sie in Thonny den Code aus dem vorherigen Abschnitt, aber statt GPIO 25 verwenden Sie GPIO 15

```
....
# Deklarieren Sie eine Variable namens "led", verknüpfen Sie sie mit Pin Nummer 15 und definieren Sie sie als Ausgang
led = Pin(15, Pin. OUT)
....
```

In diesem Beispiel haben wir die LED an GPIO 15 angeschlossen, aber Sie können auch einen anderen GPIO verwenden, wenn Sie möchten.

Testen Sie Ihre Jumper

Der Bausatz enthält mehrere Steckbrücken, mit denen Sie die Verdrahtung Ihrer Bauteile auf Ihrem Breadboard testen können, bevor Sie die Bauteile auf die CanSat-Basisplatine löten.

Manchmal kann es vorkommen, dass einige dieser Jumper aufgrund von Problemen im Werk defekt sind.

Um Ihnen einige Kopfschmerzen und Zeit zu ersparen, raten wir Ihnen dringend, alle Jumper mit der vorherigen Übung "Blinken einer externen LED" zu testen, indem Sie die 2 Jumper durch die anderen im Kit enthaltenen Jumper ersetzen.

Alternativ können Sie die Steckbrücken auch mit dem Durchgangsprüfer eines Multimeters testen

Messung von Temperatur und Druck

Der mitgelieferte BMP280-Chip misst sowohl den atmosphärischen Druck als auch die Temperatur.

Installieren Sie die BMP280 Python-Bibliothek

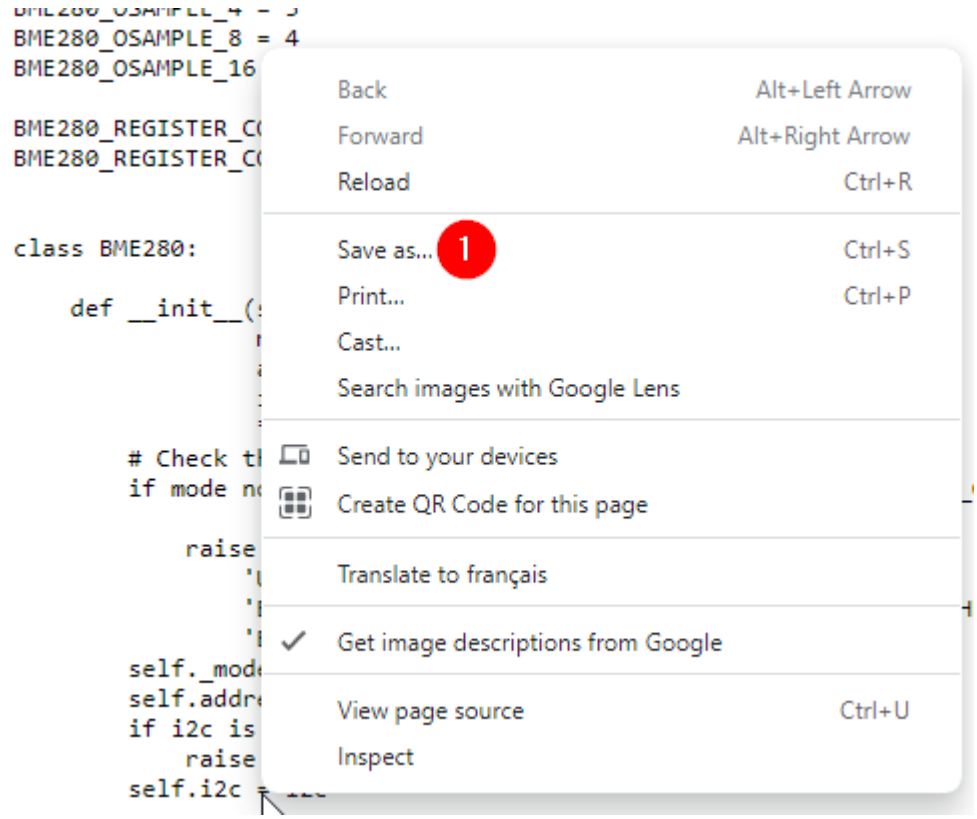
Micro Python bietet einige integrierte Funktionen für die Verwaltung des Pi Pico; diese können jedoch durch die Verwendung von Bibliotheken von Drittanbietern erweitert werden. Dabei handelt es sich um Bibliotheken, die von Herstellern, Zulieferern und der Micro Python-Community für die Verwendung zusätzlicher Geräte mit dem Pi Pico erstellt wurden. Diese Bibliotheken reduzieren die Komplexität der Verwendung externer Geräte, indem sie High-Level-Funktionen zur Interaktion mit den von ihnen unterstützten Geräten bereitstellen

Das Kit besteht aus mehreren Sensoren, für die wir spezielle Micro-Python-Bibliotheken verwenden werden, die wir im Internet finden können.

Um mit dem BMP280 zu interagieren, benötigen wir eine spezielle Python-Bibliothek, die über diesen Link heruntergeladen werden kann:

https://raw.githubusercontent.com/mchobby/esp8266-upy/master/bme280-bmp280/bme280.py

Klicken Sie mit der rechten Maustaste auf die Datei und wählen Sie "Speichern unter...", wie in der Abbildung unten gezeigt:

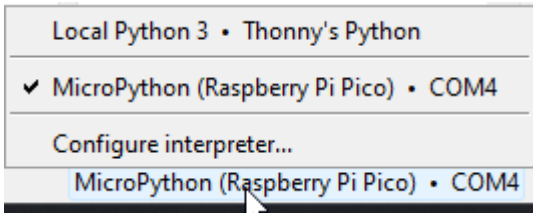


Speichern Sie die Datei auf Ihrem PC und behalten Sie den Namen "bme280.py" in Kleinbuchstaben bei.

Mit Thonny muss die Bibliothek dann von Ihrem PC in das Verzeichnis /lib des Raspberry Pi Pico übertragen werden.

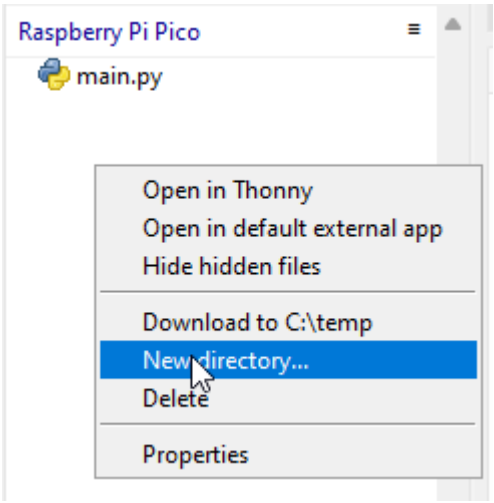
Stellen Sie sicher, dass der Pico an Ihren PC angeschlossen und eingeschaltet ist.

Stellen Sie in Thonny sicher, dass Sie mit dem Pico verbunden sind, indem Sie auf das Menü unten rechts klicken

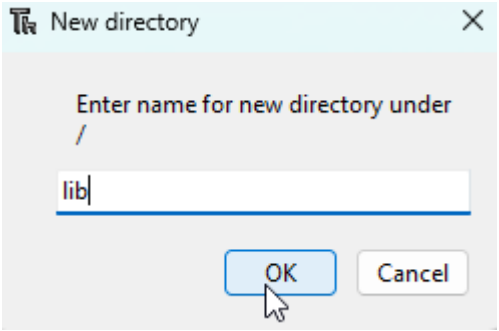


Vergewissern Sie sich im Menü "Ansicht", dass die Option "Dateien" aktiviert ist.

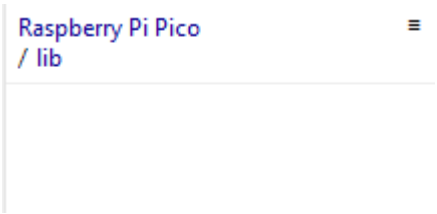
Klicken Sie dann im Fenster "Raspberry Pi Pico" unten rechts mit der rechten Maustaste und wählen Sie "Neues Verzeichnis...".



Geben Sie "lib" (**in Kleinbuchstaben**) ein und drücken Sie ok

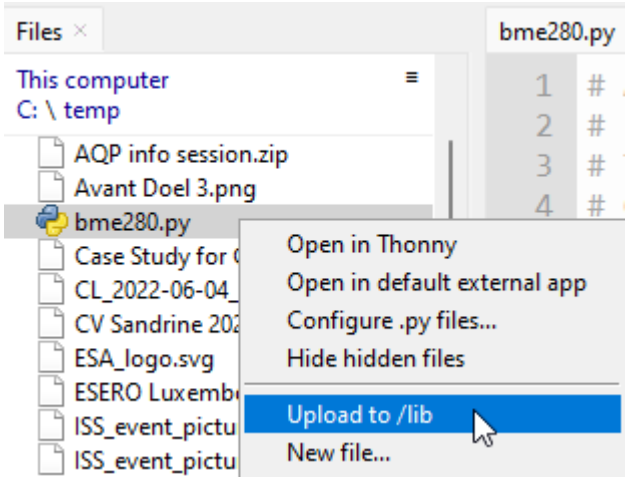


Doppelklicken Sie dann auf das Verzeichnis "lib", das im Moment noch leer ist



Navigieren Sie im Fenster "Dateien" oben links zu dem Ort, an dem Sie die Datei bme280.py gespeichert haben.

Klicken Sie mit der rechten Maustaste auf die Datei bme280.py und drücken Sie "upload to /lib", um die Bibliothek auf dem Pico zu installieren.

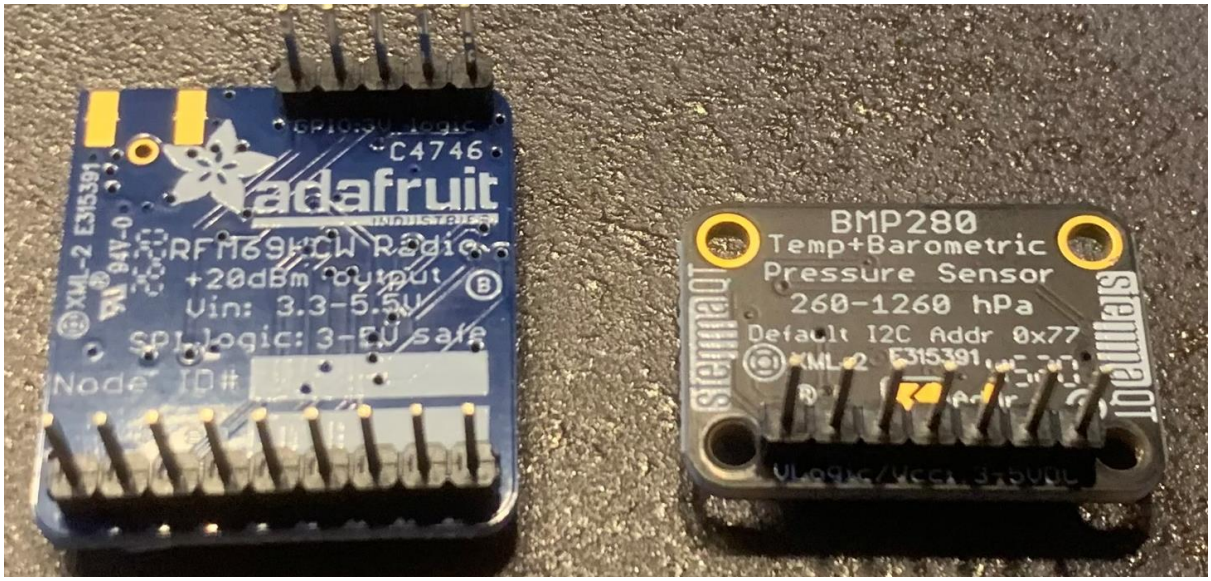


Löten der Komponentenstifte

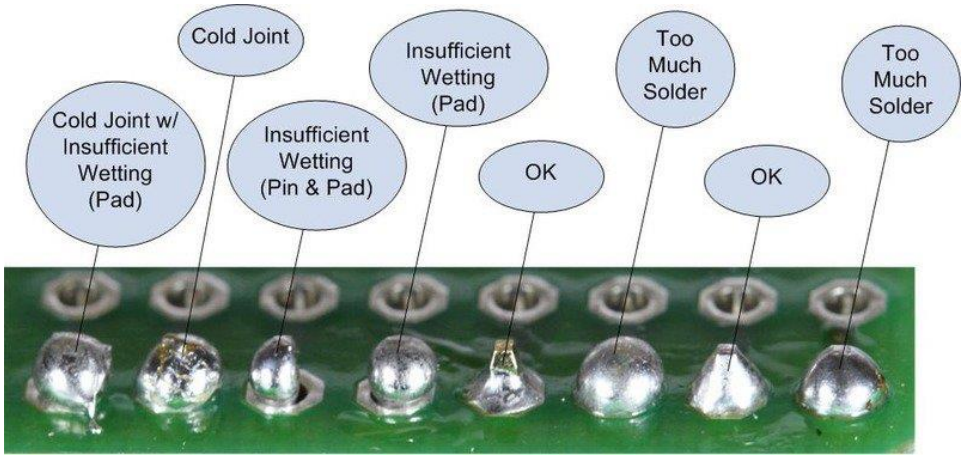
WARNUNG: Bitte lesen Sie zuerst sorgfältig die Anleitung zum exzellenten Löten von Adafruit

Sowohl das BMP280 als auch das Funkmodul RFM96W für die Bodenstation müssen mit Stiftleisten versehen werden, damit sie an die Lochrasterplatinen angeschlossen werden können.

Die Pins der BMP280-Platine und des RFM96W-Funkmoduls, das für die Bodenstation verwendet werden soll, müssen mit Stiftleisten verlötet werden.



Für das zweite RFM96W-Funkmodul sollten Sie an die Pins **Buchsenbrücken** und **keine Stiftleisten** anlöten, damit Sie das Modul später sowohl auf der Lochrasterplatine als auch auf der CanSat-Basisplatine leicht anschließen können.



https://learn.adafruit.com/adafruit-guide-excellent-soldering/common-problems

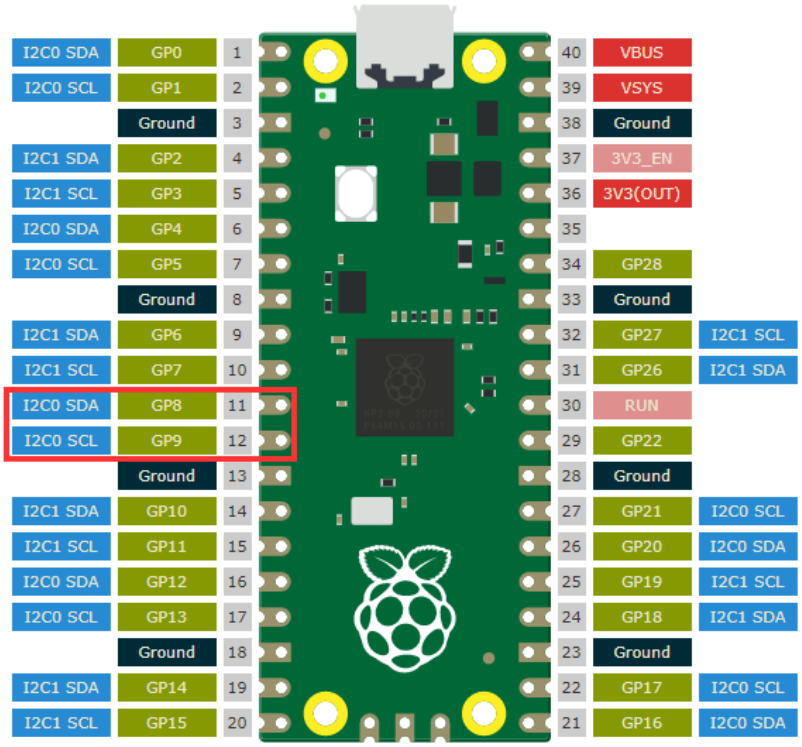
Anschluss des Druck-/Temperatursensors BMP280 über I2C

Bevor wir Daten vom Sensor lesen können, müssen wir ihn mit dem Pi verbinden. Dazu müssen wir zunächst Stiftleisten an seine Pins löten, wie auf Seite 8 der BMP280-Dokumentation beschrieben.

In der Dokumentation wird erklärt, dass wir eine 3,3-V-Stromversorgung benötigen, um das Gerät zu betreiben. Wir können die Pico 3V3 und GND (Masse) Pins dafür verwenden. In der Dokumentation wird auch erklärt, dass wir eine Verbindung über den I2C-Bus herstellen können. I2C erfordert den Anschluss von zwei Datenkabeln, so dass wir insgesamt vier Kabel anschließen müssen.

1. Suchen Sie mithilfe der interaktiven Pinout-Website den 3V3-Stromversorgungs-Pin am Pico

2. Verwenden Sie weiterhin die interaktive Pinout-Website, um die I2C (0)-Pins auf dem Pico zu finden (siehe unten)



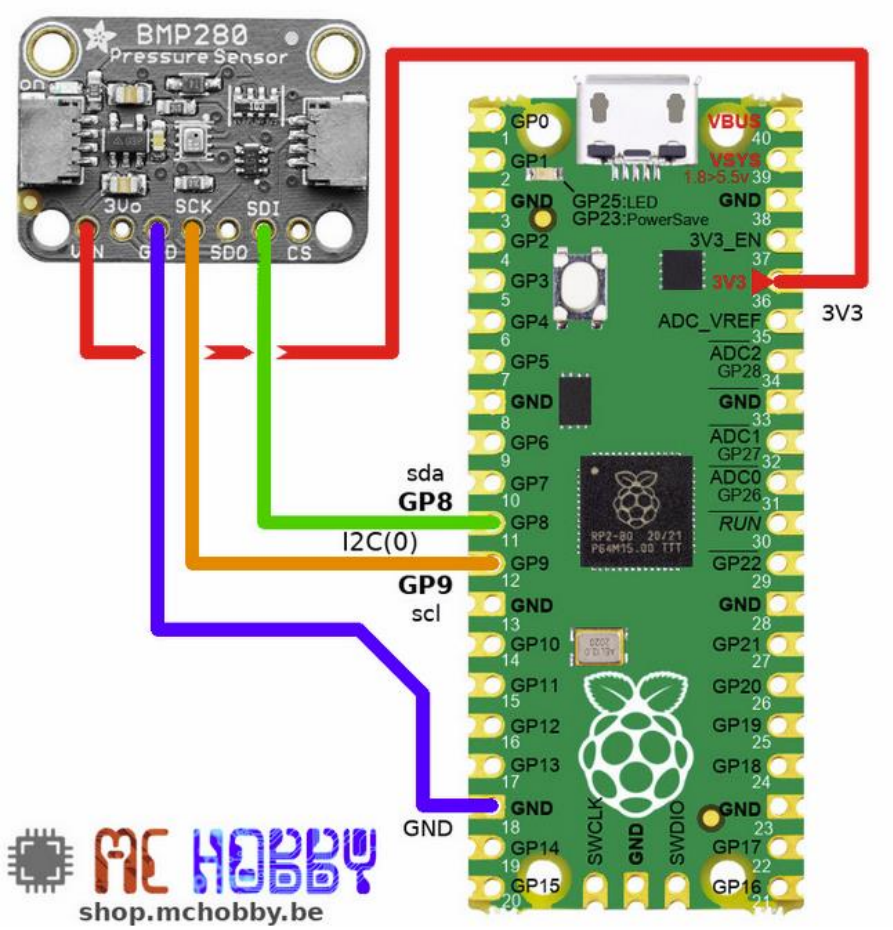
3. Verbinden Sie den 2-6V Eingangspin des BMP280 mit dem Pin 3V3 des Pi (3,3 Volt 300 mA Ausgang)

4. Verbinden Sie den GND-Pin des BMP280 mit einem GND-Pin des Pi.

5. Verbinden Sie den I2C 0 SDA (Pin 11) mit dem SDI-Pin des BMP280.

6. Verbinden Sie den I2C 0 SCL (Pin 11) mit dem SCK-Pin des BMP280.

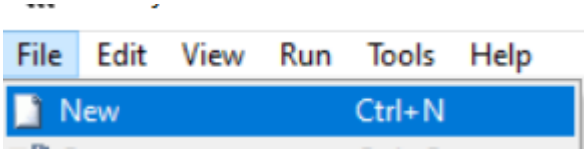
Siehe das nachstehende Anschlussschema:



Ablesen von Temperatur und Druck

Nun, da der BMP280 angeschlossen und eingerichtet ist, können wir Daten von ihm lesen.

1. Eine neue Datei erstellen



2. Speichern Sie diese Datei unter dem von Ihnen gewünschten Namen mit **der Endung . py**

3. Füllen Sie die Datei mit folgendem Code

```
from machine import I2C
# Die BME280-Bibliothek funktioniert auch für den BMP280-Sensor.
from bme280 import BME280, BMP280_I2CADDR
from time import sleep
# Initiieren eines neuen I2C-Anschlusses auf Bus 0, sda=GP8, scl=GP9 @ 400
KHz (Standard)
i2c = I2C(0)
# Erstellen Sie eine neue BME280-Variable, die an den i2c-Bus 0
angeschlossen ist und mit der Adresse BMP280_I2CADDR kommuniziert.
bmp = BME280(i2c=i2c, Adresse=BMP280_I2CADDR)
while True:
    # Druck eines Tupels mit (Temperatur, Druck und Feuchtigkeit)
    print( bmp. raw_values )
    sleep(1)
```

4. Durch Drücken der Taste "Run" werden die Temperatur, der Druck und die Luftfeuchtigkeit im Sekundentakt ausgedruckt.

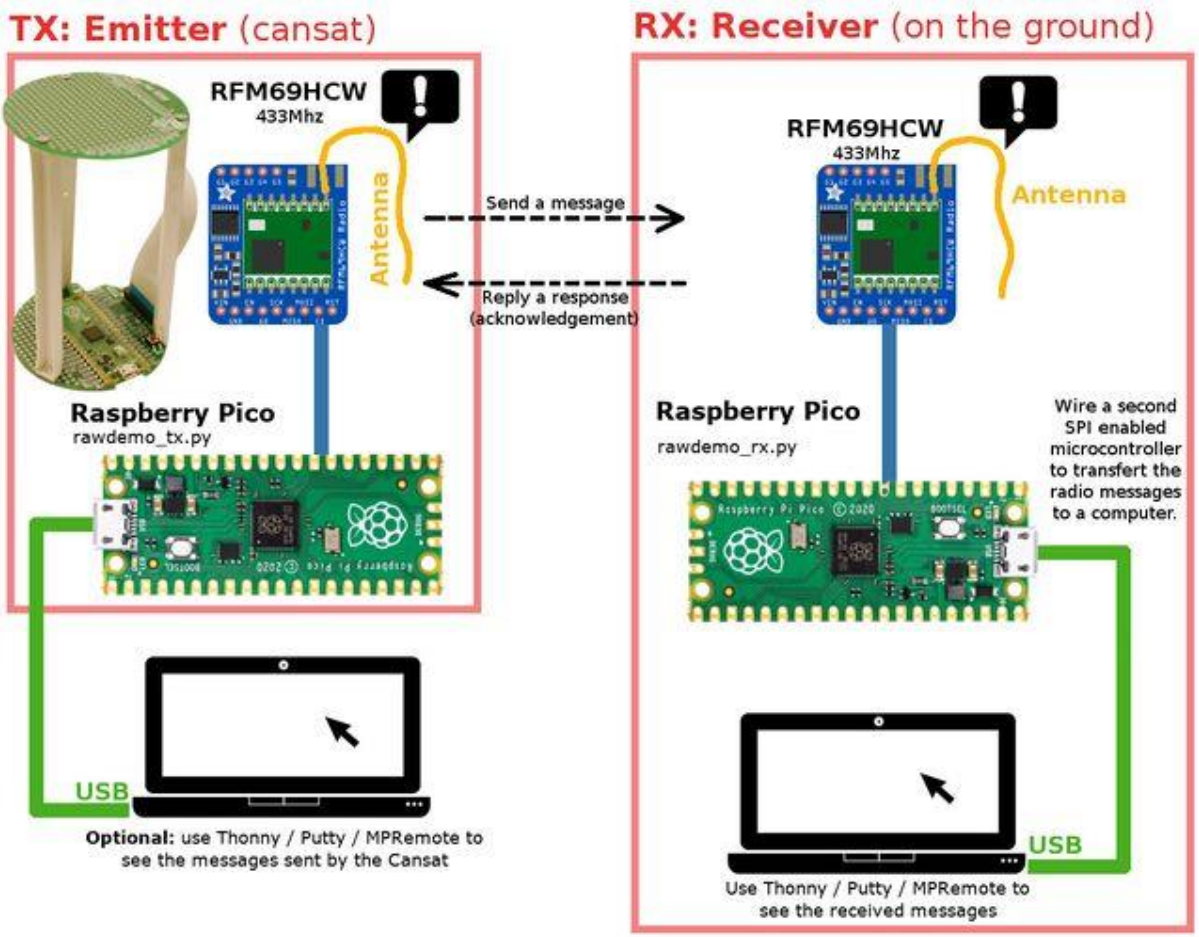
```
(22.28, 1017.68, 0.0)
(22.27, 1017.66, 0.0)
(21.87, 1017.67, 0.0)
(21.83, 1017.73, 0.0)
(21.83, 1017.68, 0.0)
(21.81, 1017.68, 0.0)
(21.81, 1017.68, 0.0)
```

Die Höhe kann anhand des Luftdrucks berechnet werden, wie in diesem Dokument erläutert.

Empfang von Funkdaten

Der Bausatz enthält 2 Pico-Mikrocontroller und 2 RFM69HCW-Module, um sowohl einen "Datensender" (den CanSat) als auch einen "Datenempfänger" (die Bodenstation) zu bauen.

In diesem Abschnitt werden wir die Bodenstation einrichten (rechts unten)



Eine erfolgreiche Kommunikation erfordert beide Seiten:

- Identische Frequenzen (z. B. 433,1 MHz).
- Identische Verschlüsselungsschlüssel.
- Gut gestaltete Antennen, außer bei Tests auf Breadboards, wo die 2 RFM69HCW sehr nahe beieinander liegen.

Installieren Sie die RFM69HCW-Python-Bibliothek

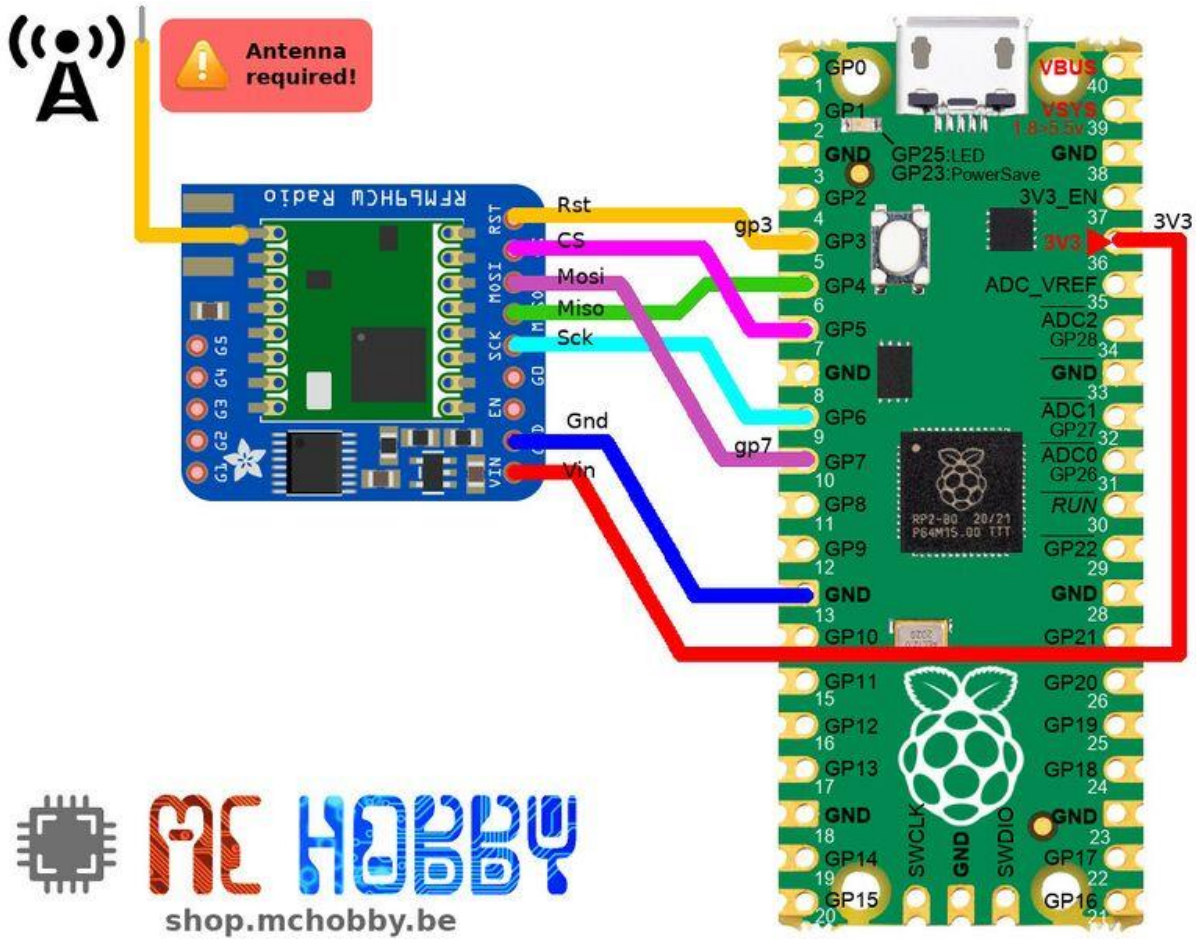
Gehen Sie mit Thonny genauso vor wie bei der Installation der BMP280-Bibliothek, um die folgende Datei rfm69.py in den Ordner /lib Ihres 2 Pico zu laden.

https://raw.githubusercontent.com/mchobby/esp8266-upy/master/rfm69/lib/rfm69.py

Verbinden des RFM69HCW über SPI

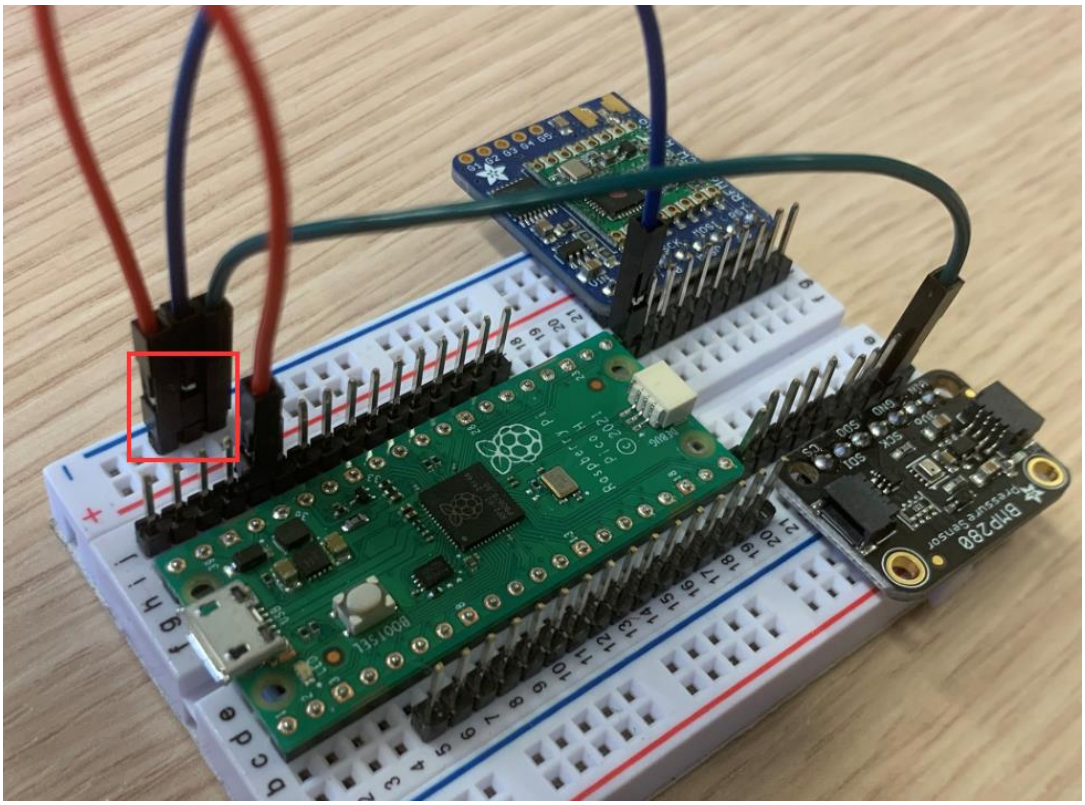
Mit Hilfe von Stiftleisten und Überbrückungskabeln müssen wir die Bodenstation Pico und den RFM69HCW auf einem Breadboard miteinander verbinden

In der Dokumentation des RFM69HCW wird erklärt, dass wir ihn über den SPI-Bus anschließen können.
SPI erfordert den Anschluss von 4 Datenkabeln, einer Reset-Verbindung und 2 Stromkabeln, so dass wir insgesamt sieben Kabel anschließen müssen.

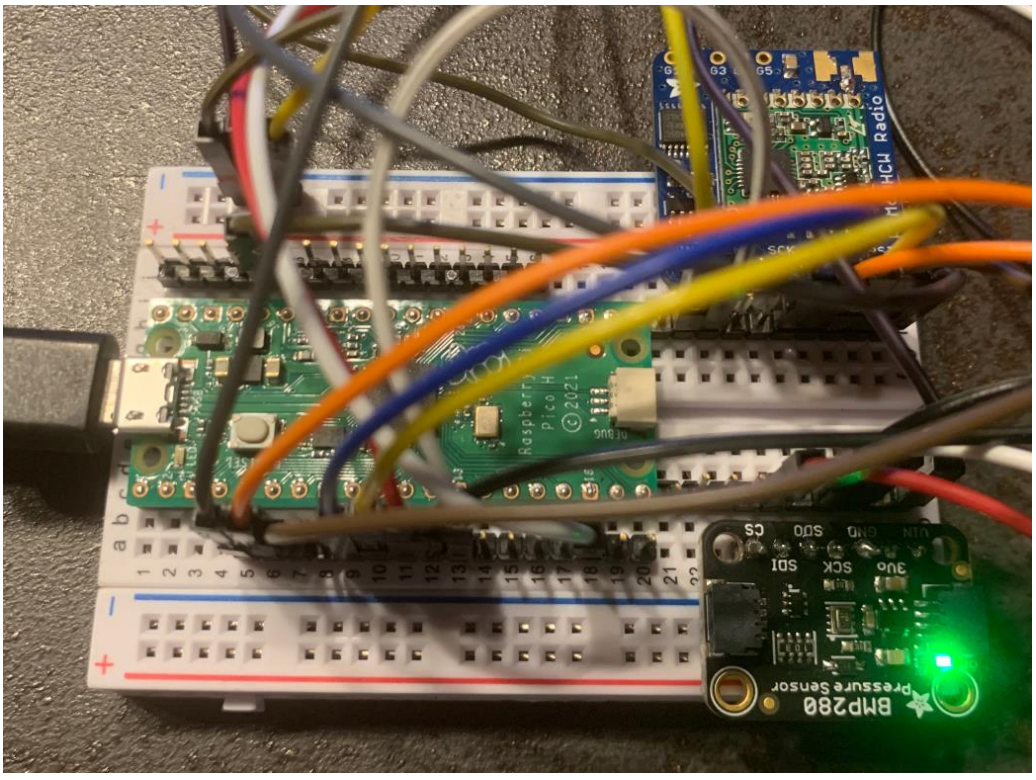


RFM69HCW	PICO
RST	GP3
CS	GP5 (Slave Select)
MOSI	GP7 (Miso)
MISO	GP4 (Mosi)
SCK	GP6 (Clock)
GND	GND
VIN	3V3

Da der 3V3-Power-Pin bereits für die Stromversorgung des BMP280 verwendet wird, können Sie ihn auch für die Stromversorgung des Funkmoduls verwenden, wie unten in rot dargestellt



Nach der Verkabelung des Radios sollte das Breadboard wie folgt aussehen



Warten auf Funkdaten

1- Laden Sie das Empfängerskript herunter und öffnen Sie es in Thonny

2- Aktualisieren Sie die Frequenz und den Verschlüsselungscode in den Zeilen 17 und 30

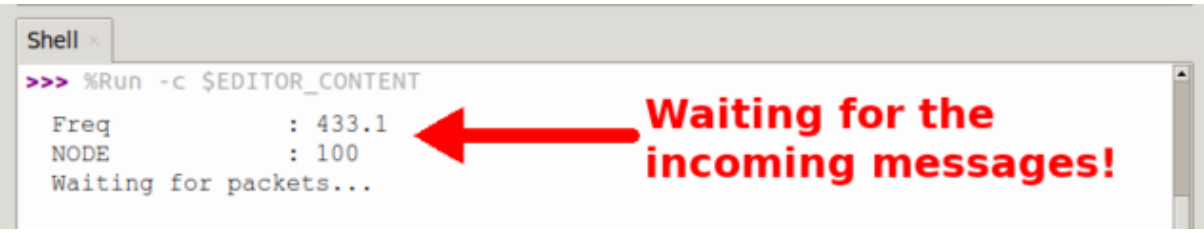
   a. Die Frequenz wird in Zeile 17 festgelegt und liegt zwischen 430 und 440Mhz,

   b. Der Verschlüsselungscode kann in Zeile 30 mit folgendem Code definiert werden

```
rfm.encryption_key = bytes( [1,2,3,4,5,6,7,8,1,2,3,4,5,6,7,8] )
```

   Jedes Team erhält vor dem Start der Rakete seine eigene Frequenz und seinen eigenen Verschlüsselungscode.

3- Das Skript ausführen



Senden von Funkdaten

In diesem Abschnitt werden wir den Datenemitter bauen, der sich an Bord Ihres CanSat befinden wird.

Lassen Sie Thonny geöffnet, während das Empfängerskript läuft, und öffnen Sie ein zweites Mal Thonny.

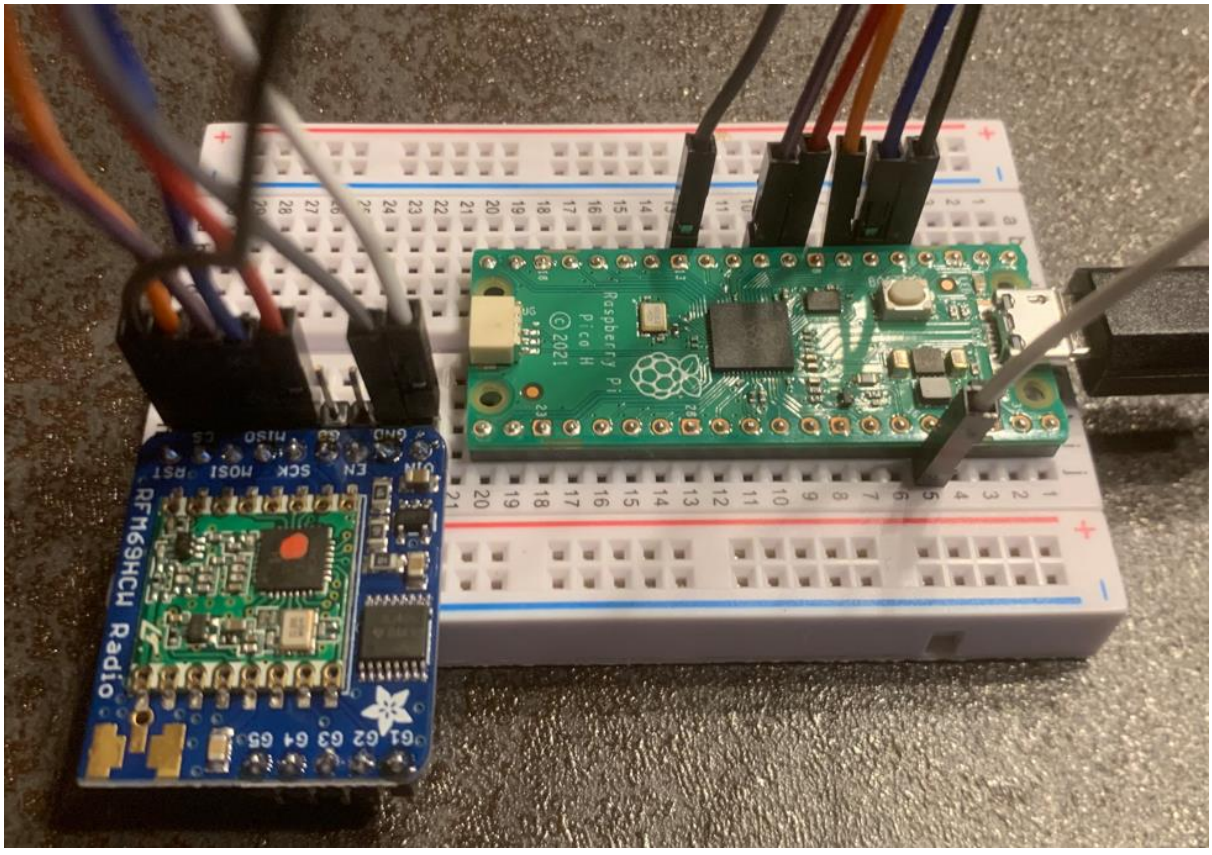
Mit Hilfe von Stiftleisten und Überbrückungskabeln müssen wir den CanSat Pico und den RFM69HCW auf einem anderen Breadboard miteinander verbinden

Auf dem zweiten Pico:

1- MicroPython installieren

2- Installieren Sie die RFM69HCW-Python-Bibliothek, falls noch nicht geschehen (siehe vorheriger Abschnitt)

3- Verbinden Sie den RFM69HCW über SPI (siehe vorherigen Abschnitt)

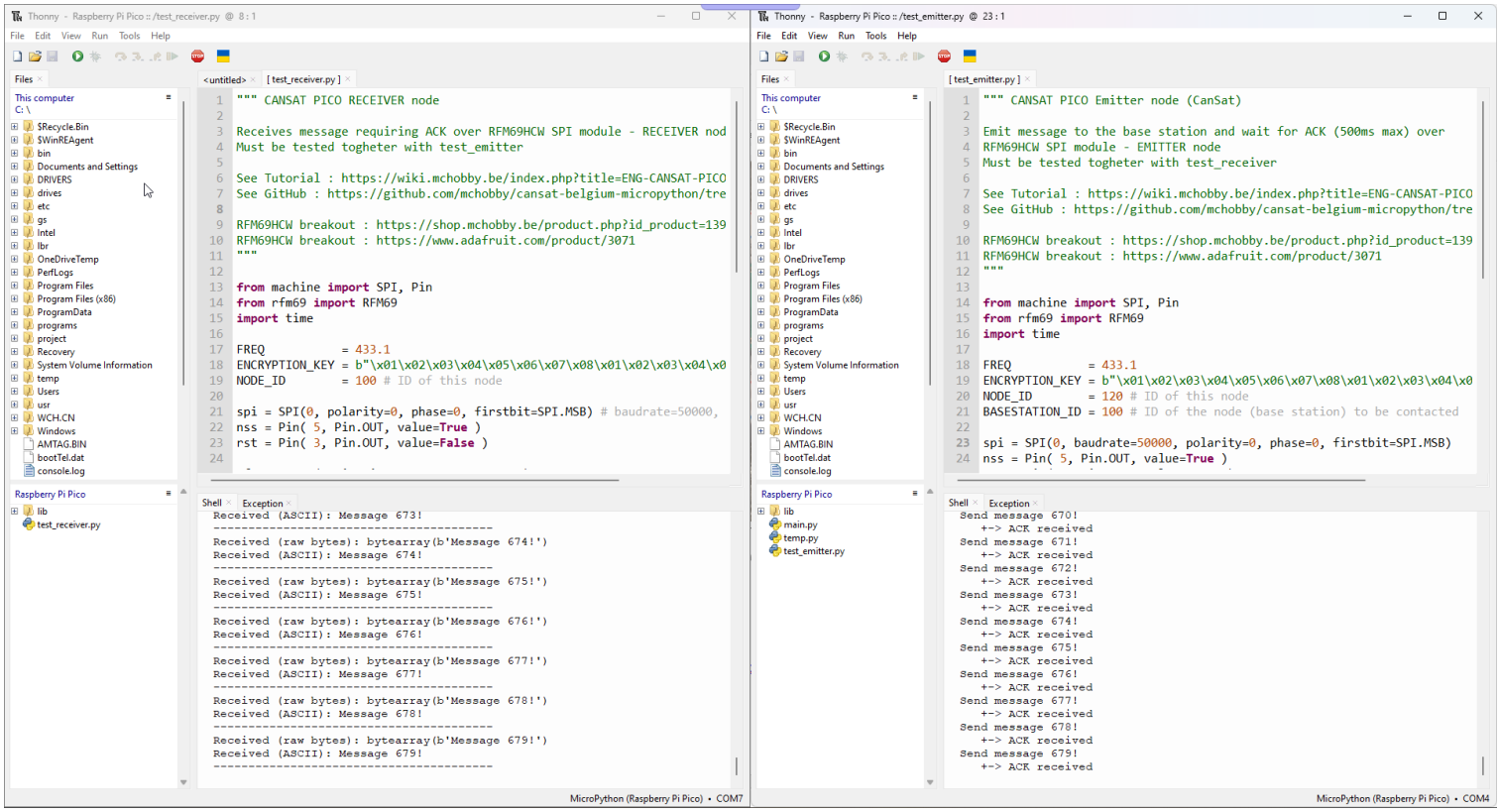


4- Laden Sie das Absender-Skript herunter und öffnen Sie es in Thonny

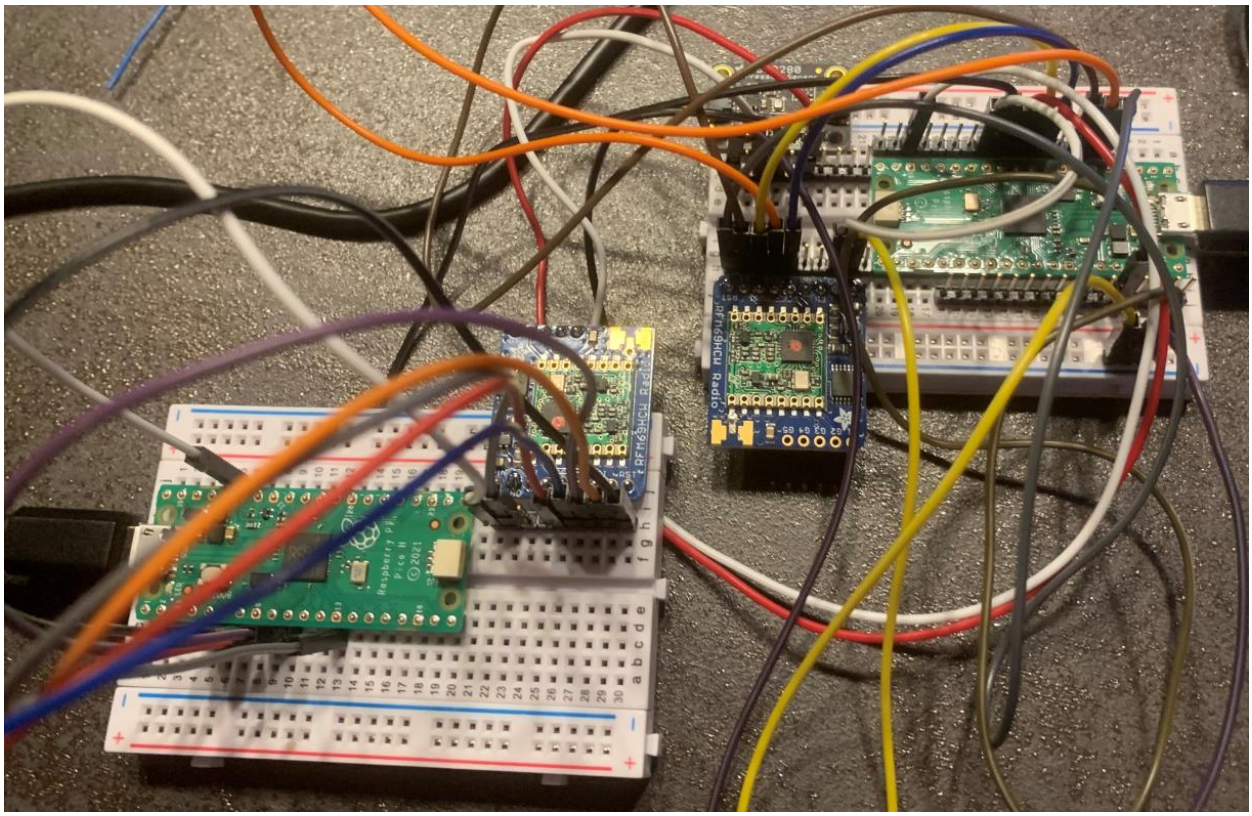
5- Aktualisieren Sie die Frequenz und den Verschlüsselungscode in den Zeilen 18 und 32 auf die gleichen Werte wie bei der Bodenstation.

6- Laden Sie das Skript auf den Pico hoch

7- Das Skript ausführen



Die 2 Funkgeräte übertragen Daten aneinander.
Sie dürfen nicht mehr als 5 cm voneinander entfernt sein.



Ausführlichere Informationen finden Sie auf der [McHobby-Website]

Zusammenbau Ihres CanSat

Wenn Sie alle Ihre Bauteile auf den Lochrasterplatinen getestet haben

Das McHobby CanSat-Wiki enthält alle notwendigen Informationen für den Zusammenbau der Komponenten auf den Cansat-Basis- und Erweiterungsplatinen, die dem Bausatz beiliegen.

Die Montageschritte für die Hauptmission sind die folgenden:

1- Anlöten eines Pico an die CanSat-Basis

   WICHTIG: im Bausatz sind Pico-Platten mit vorgelöteten Steckern enthalten, die Sie auch zum Anlöten der CanSat-Basis verwenden können

2- Löten Sie das PowerBoost 500-Bauteil an die CanSat-Basis

3- Verbinden Sie den BMP280 mit dem Qwiic/StemmaQt-Kabel

4- Anschließen des TMP36 (optional)

5- Anschließen des Funkmoduls

Die CanSat-Erweiterung ist nützlich, wenn Sie Platz benötigen, um weitere Komponenten für die sekundäre Mission hinzuzufügen, wie GPS, Beschleunigungsmesser, Kamera usw.

Was ist als nächstes zu tun?

- Liesen Sie die Ressource zur Gestaltung deines Fallschirms
- Lesen Sie die Ressource zum Entwurf Ihrer Funkantennen

Weiter gehend

- Ziehen Sie ein GPS in Betracht, um Ihre Chancen zu erhöhen, Ihren CanSat nach dem Start zu finden.
  - Tutorial
  - GPS-Modul
  - MicroPython GPS-Bibliothek
- Einige Teams entwerfen ihr eigenes Echtzeit-Dashboard für die Bodenstation

Anhänge

Python lernen

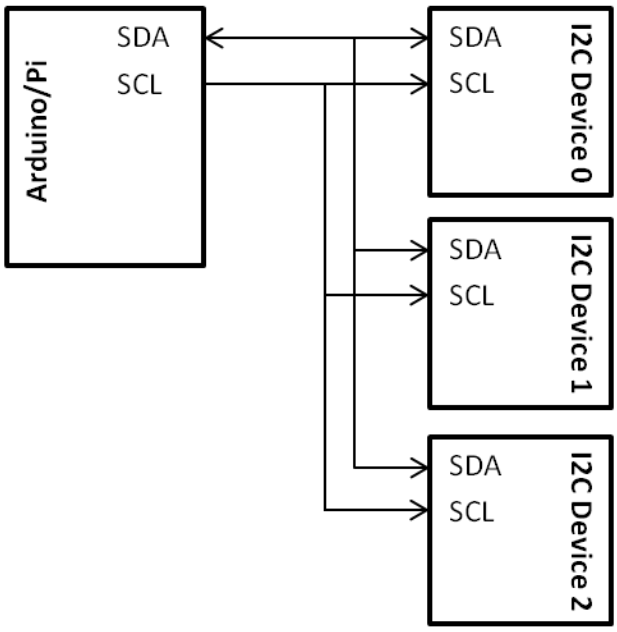
MicroPython ist Python sehr ähnlich. Wenn Sie ein absoluter Anfänger in Python sind, finden Sie hier eine Liste mit nützlichen Ressourcen für den Einstieg:

https://wiki.python.org/moin/BeginnersGuide/NonProgrammers

LearnPython zum Beispiel ist ein großartiges interaktives Lernprogramm, das für absolute Anfänger geeignet ist.

I² C Protokoll

Mit I2C können mehrere Geräte (bis zu 1008) über nur 2 Drähte an dieselbe I2C-Schnittstelle angeschlossen werden. Es ermöglicht auch eine bidirektionale Kommunikation über diese zwei Drähte und ist daher ideal für die Kommunikation mit vielen Sensoren. Ein Beispiel für eine Verdrahtung mit drei Geräten wäre wie folgt:



Die Software, die für die Kommunikation mit I2C-Geräten erforderlich ist, kann sehr komplex sein. Die meisten Geräte verfügen jedoch über eine Software-Bibliothek, die Ihnen Funktionen zur Verfügung stellt, die die Verwendung des Geräts erleichtern. Wir verwenden zum Beispiel die mitgelieferte BMP280-Bibliothek, um den Low-Level-I2C-Code zu verbergen.

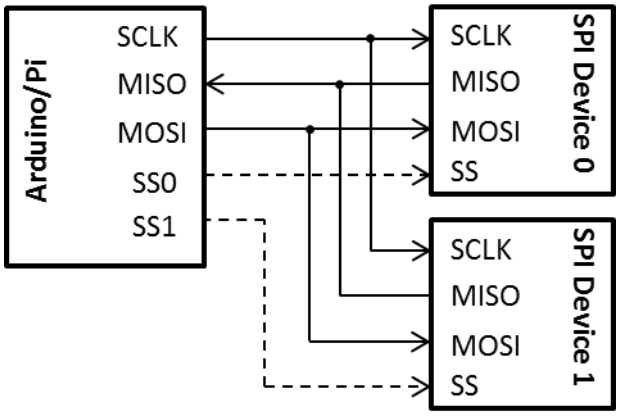
Typische I2C CanSat-Anwendungen:
"intelligentere" Sensoren (z. B. BMP280), Beschleunigungsmesser, Analog-Digital-Wandler, Digital-Analog-Wandler, LCD-Bildschirme, Batterie-Controller

Serielle Peripherieschnittstelle (SPI)

SPI bietet eine Schnittstelle mit leistungsfähigeren Funktionen als I2C, allerdings auf Kosten eines höheren Verdrahtungsaufwands. Wie I2C unterstützt auch SPI die bidirektionale Kommunikation mit mehreren Geräten, bietet aber einen viel höheren Datendurchsatz. Dadurch eignet sie sich für die Kommunikation mit den komplexesten Geräten, die Sie an den CanSat anschließen können. Die Schnittstelle besteht aus mindestens vier Pins:

SCLK: *Serieller Takt*. Ein Strom von 0-1s, an dem die Daten ausgerichtet werden. Die SPI-Taktrate hängt von der Geschwindigkeit dieses Stroms ab. Sie können diesen Strom verlangsamen, wenn Sie Probleme mit der Datenintegrität haben.
MISO: *Master-Eingang / Slave-Ausgang.* Die Daten vom Peripheriegerät zum Pi.
MOSI: *Master-Ausgang / Slave-Eingang.* Die Daten vom Pi zum Peripheriegerät.
SS0/CE0: *Slave-Auswahl / Chip-Freigabe.* Aktiviert ein Peripheriegerät und bedeutet, dass das Gerät an den MISO-Pin ausgeben kann. Für jedes Peripheriegerät wird ein SS/CE-Pin benötigt.

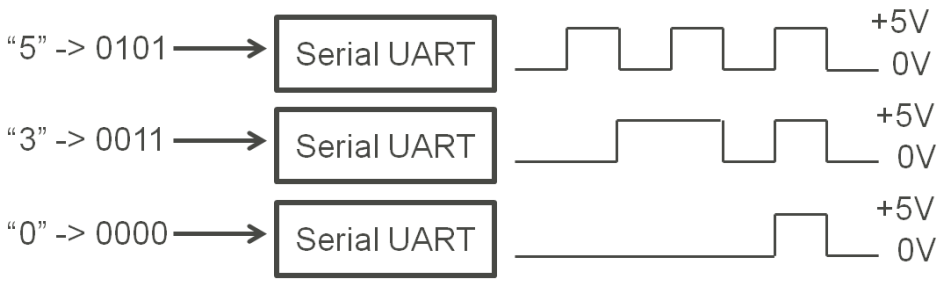
Um SPI zu verwenden, brauchen Sie sich nicht allzu sehr um die Funktion dieser Pins zu kümmern, da die Software-Bibliothek des Geräts den Großteil des Low-Level-SPI-Codes für Sie übernimmt. Es ist jedoch gut, sich ihrer Funktion bewusst zu sein, wenn man mehrere SPI-Geräte kaskadiert, z. B. um zwei Geräte zu verbinden, benötigt man zwei SS/CE-Pins:



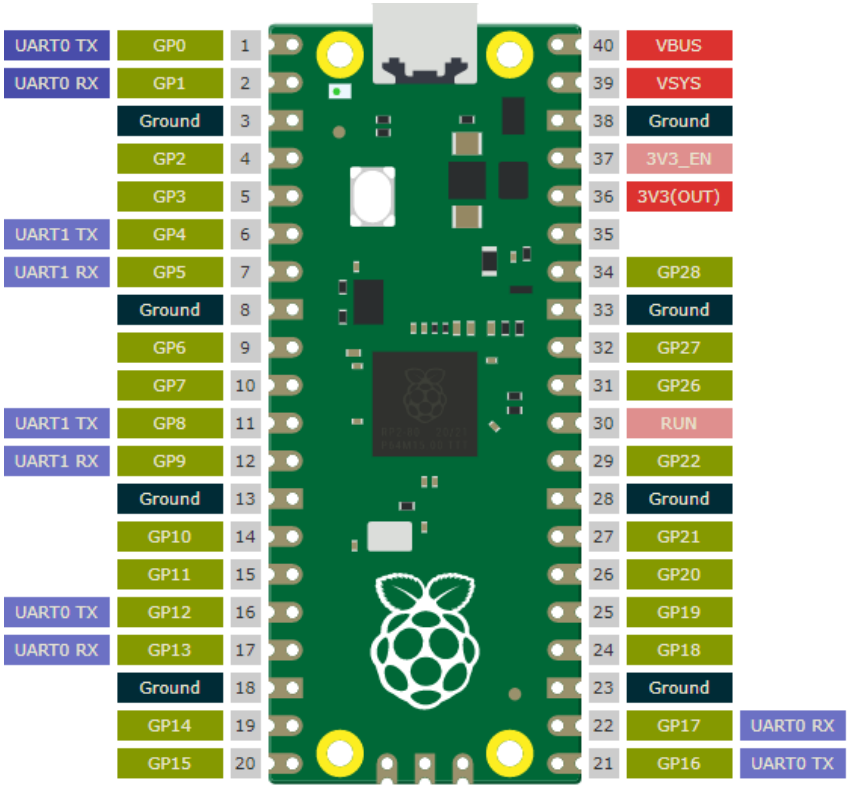
Typische SPI CanSat-Anwendungen:
Kameras, Speicherkarten (z. B. SD-Karten), GPS-Module, WiFi-Modems

Universeller asynchroner Empfänger-Sender (UART)

Trotz des Namens ist UART eine relativ einfache Kommunikationsschnittstelle. Sie funktioniert auf die gleiche Weise wie GPIO mit Wahr/Falsch-Werten, die als 0 V und 5 V dargestellt werden, aber es werden Impulse über die Leitung gesendet, anstatt gleichmäßige Spannungsimpulse. Dadurch kann ein numerischer Wert in eine Reihe von Impulsen umgewandelt und über eine einzige Leitung gesendet werden:



Der Raspberry Pi Pico hat zwei UARTs. Diese können mit vielen Paaren von GP-Pins verbunden werden, wie im Pinout-Diagramm unten in lila dargestellt: TX ist der Transmit (d.h. Daten, die vom Pi gesendet werden) und RX ist der Receive (d.h. Daten, die zum Pi gesendet werden).



Typische UART CanSat Verwendungen:
Senden von Debug- und Entwicklungsmeldungen an einen PC, Kommunikation mit GPS-Sensoren, Kommunikation mit externen WiFi- und GPRS (3G)-Modems.