Aus der Vortragsreihe des OV Meschede :
„Arduino und Co.“

Hier : Raspberry Pi

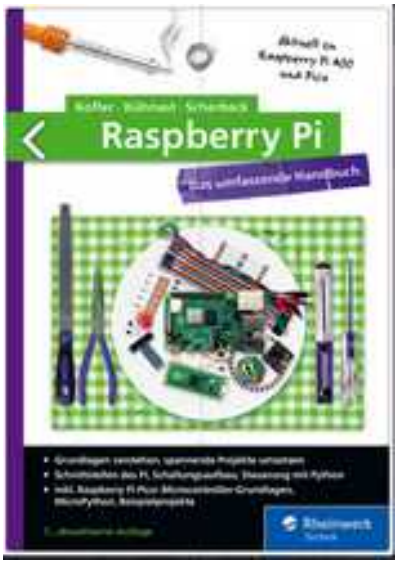
© Heribert, DK2JK und Josef, DL8DBN

RaspberyPi

Dieses Board ist praktisch ein kleiner PC; es fehlen nur noch Tastatur und Bildschirm; ansonsten verhält er sich ähnlich wie ein PC. Das Betriebssystem ist Linux. Gegenüber einem PC hat er den Vorteil, dass weitere Pins zum Steuern zur Verfügung stehen, also Ein- und Ausgänge, über die man z.b Relais oder Leuchtdioden ansteuern kann .

Für den Bastler sind natürlich die digitalen Ein- und Ausgänge von großem Interesse.

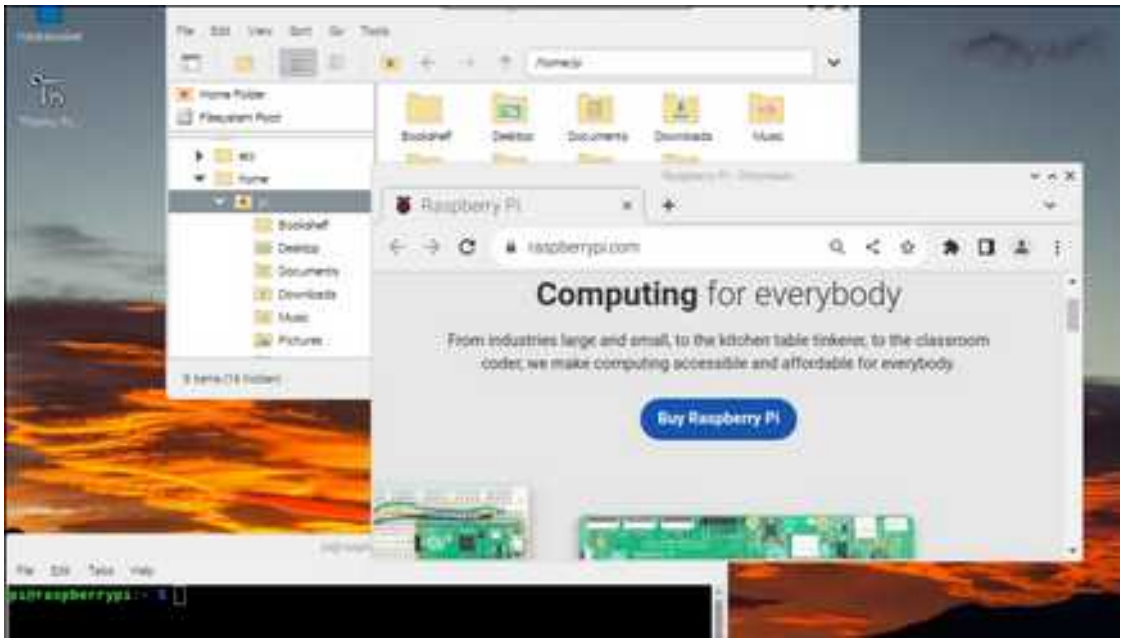
Nachschlagewerk



1088 Seiten, 7., aktualisierte Auflage 2021, gebunden, in Farbe
Rheinwerk Computing, ISBN 978-3-8362-8351-9

Ein zu empfehlendes Nachschlagewerk

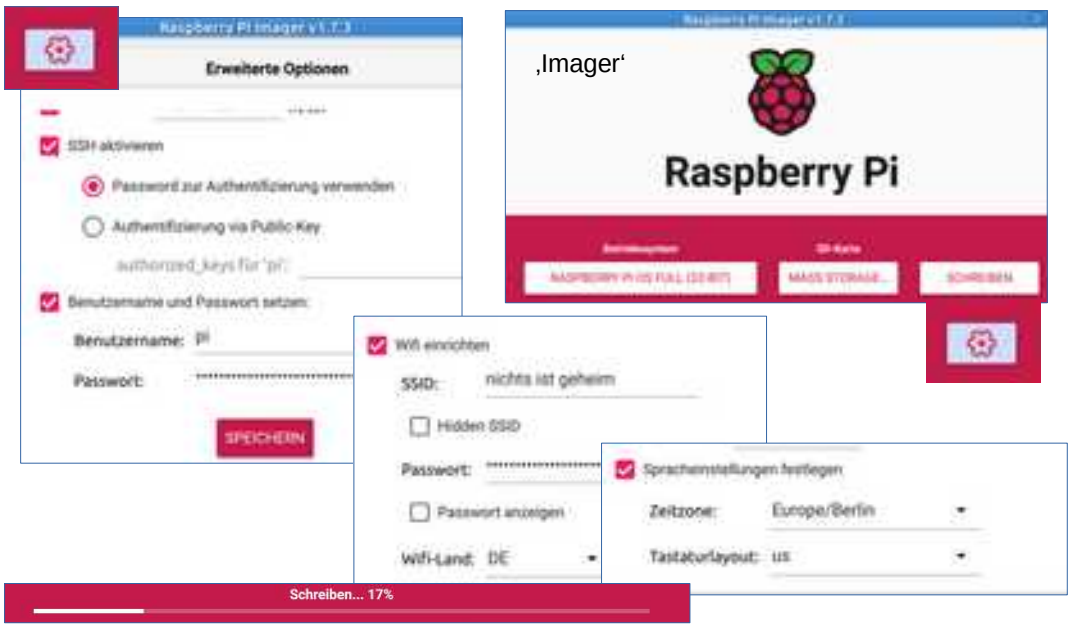
Linux Oberfläche



Dies ist die Desktop -Ansicht des Raspberry Pi; einige Apps kann man schon wie bei Windows verwenden (Dateimanager, Browser, Terminal, etc).

Die Fernbedienung geschieht über VNC Viewer, wodurch die Oberfläche des ‚Raspi' mit exakt demselben Bildschirm auf dem PC angezeigt wird. Dies muss in der Konfiguration unter Punkt ‚I3' VNC enable eingestellt werden.

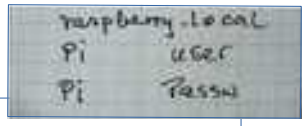
Installation mit ‚Imager‘



In den meisten Fällen will man den Raspberry Pi aber nicht als PC benutzen, sondern als eine freie programmierebare Platine, die auch Pins zum Steuern hat und WLAN als Netzwerkverbindung.

Man gibt schon beim Schreiben des Systems auf SD-Karte mitteles ‚Imager‘ die Zugangsdaten für das heimische WLAN an und dass man den ‚Raspi‘ fernbedienen möchte.

Erster Kontakt




```
dk2jk@samsung-laptop:~$ ping raspberrypi.local -c 1
PING raspberrypi.local (192.168.178.21) 56(84) Bytes Daten.
64 Bytes von raspi21.fritz.box (192.168.178.21): icmp_seq=1 ttl=64 Zeit=5.01 ms

--- raspberrypi.local ping statistics ---
1 Pakete übertragen, 1 empfangen, 0% Paketverlust, Zeit 0ms
rtt min/avg/max/mdev = 5.011/5.011/5.011/0.000 ms
```

```
dk2jk@samsung-laptop:~$ ssh 192.168.178.21 -l pi
pi@192.168.178.21's password: **
Linux raspberrypi 5.15.61-v7+ #1579 SMP Fri Aug 26 11:10:59 BST 2022 armv7l
pi@raspberrypi:~ $
```

```
pi@raspberrypi:~ $ sudo raspi-config
```

26.10.22

Im Terminal, bei Windows: CMD

1. Test:

ping raspberrypi.local

  Hurra! → Raspi ist im Netz

2. Test:

ssh 192.168.178.21 -l pi

Der Prompt hat sich in pi@raspberrypi:~ $

Geändert: → wir sind auf dem Linux Terminal

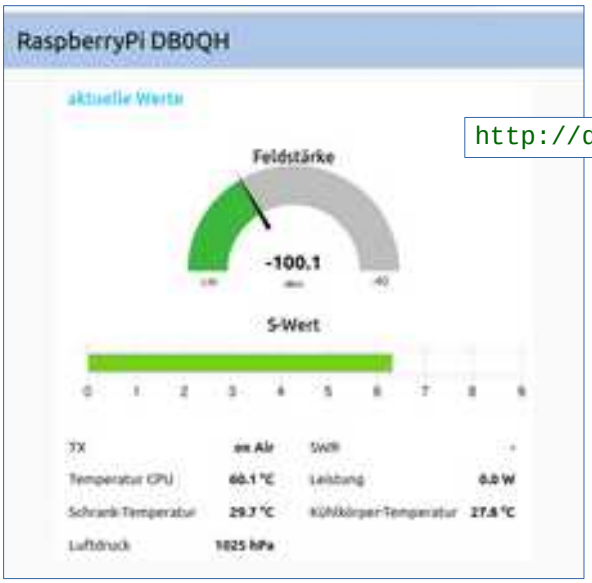
  des Raspi gelandet

3. Schritt:

sudo raspi-config

--> weitere Einstellungen

Raspi-Anwendung: DB0QH Anzeige



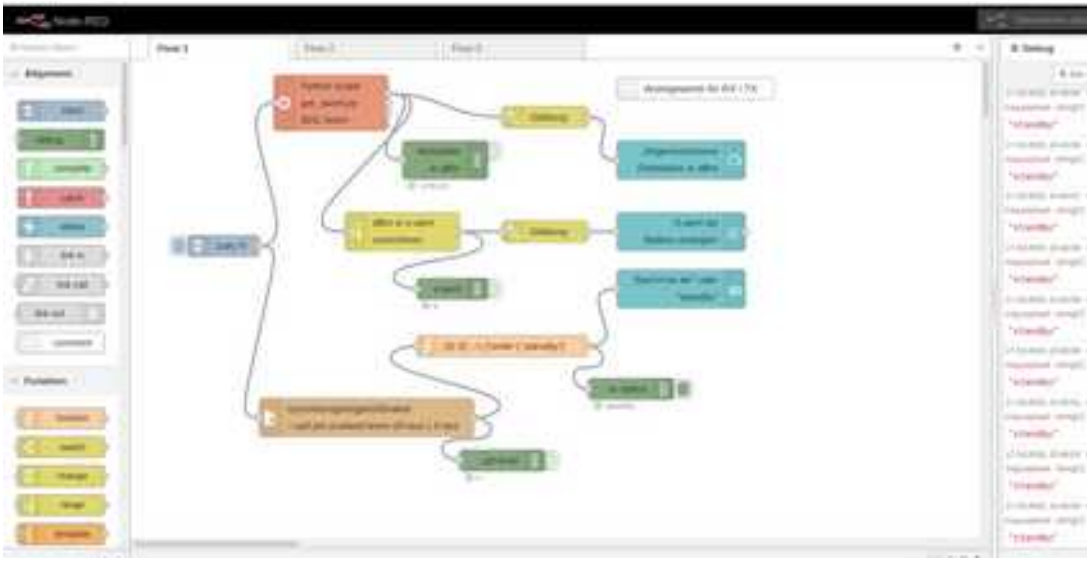
http://dmrhsk.spdns.de:59876/ui/

So sieht die DB0QH- Status- Anzeige aus.

Die Messwerte werden durch Python-Programme geholt.

Die Anzeige-Elemente wurden mit NodeRed - Funktionsblöcken programmiert

Beispiel NodeRed DB0QH

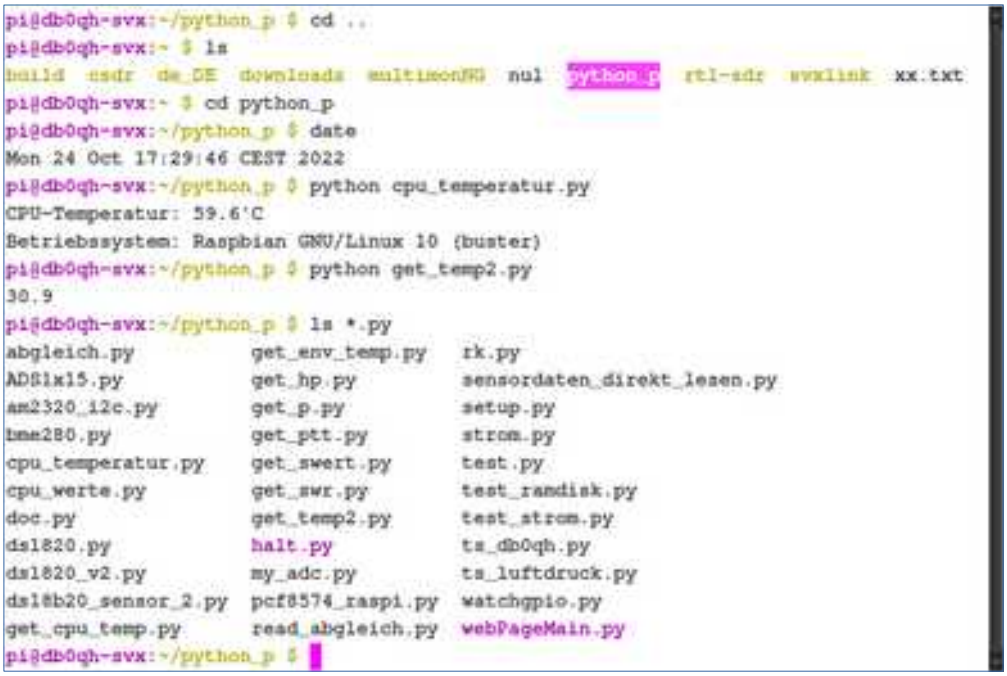


Node-RED ist ein flussbasiertes Entwicklungstool für die visuelle Programmierung.

Node-RED bietet einen Webbrowser- basierten Flow-Editor, der auf dem Raspi läuft.

Man hat Bausteine, die zu verbinden sind.
Links ein Sekundentakt. In der Mitte: Daten holen und anpassen. Rechts in Blau : Anzeige-Elemente.

Linux Terminal



cd ….change directory
ls …. list directory
date…. Zeit und Datum
python xxx.py … python Script starten

Beispiele; Lesen der CPU-Temperatur und
Lesen der Schrank-Temperatur

Python: Die „Muttersprache“ des Raspi

Python -Script:

```
pi@raspberrypi:~ $ cat app/countdown.py
#countdown.py

from time import sleep

start=4
ende=-1
for n in range (start,ende,-1):
    sleep(.5)
    print(n)
print( 'lift off !')
```

Das kommt heraus:

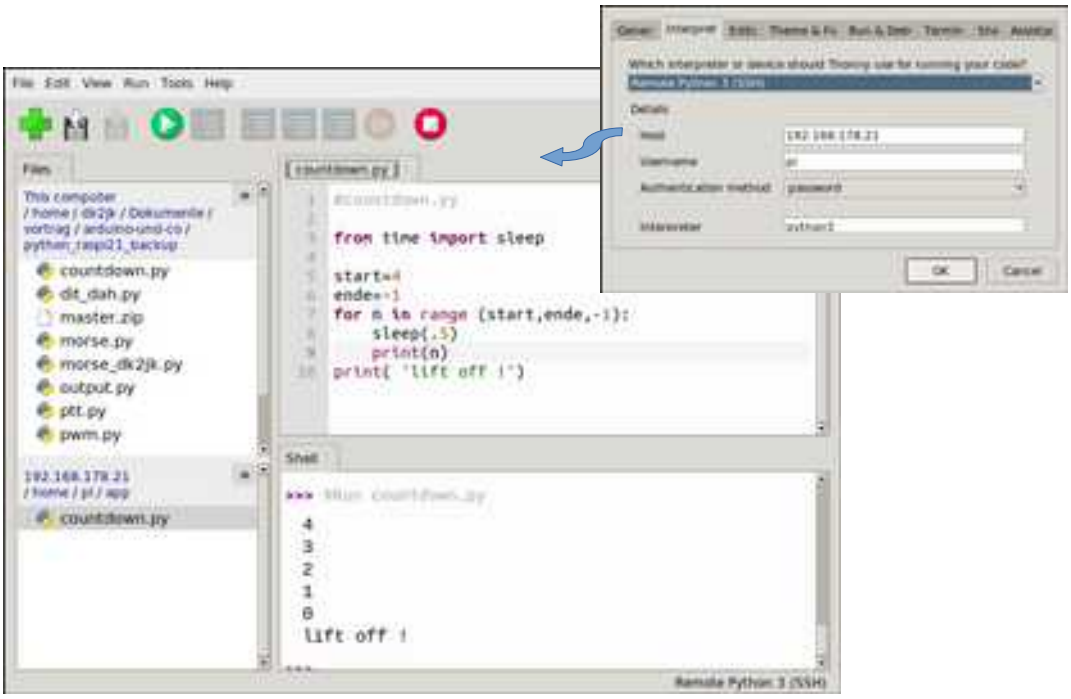
```
pi@raspberrypi:~ $ python
app/countdown.py
4
3
2
1
0
lift off !
```

Hier ein einfaches Python- Script, im Terminal angezeigt.

Start des Python- Scripts im Terminal.

(Dieses Python- Script würde auch auf einem Windows-PC laufen)

Programmieren mit Thonny



Das gleiche Python-Script wie auf der letzten Seite.

Jedoch: jetzt alles in einem Fenster und per Mausklick bedienbar.

Man beachte: Thonny läuft hier nicht auf dem Raspi, sondern auf dem PC; der Raspi ist fernbedient via SSH.

GPS entschlüsselt

Die üblichen GPS-Empfänger geben ein leicht lesbares ASCII-Protokoll aus.

Hier werden die Daten auseinander gepflückt.

Python Script: GPS Decoder

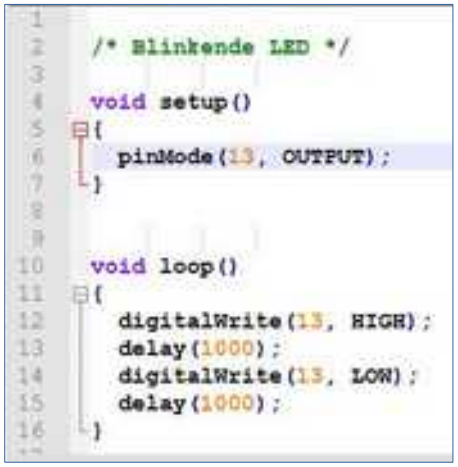
```
"nmea.txt"....
$abcde,xxx566
$GPRMC,222615.000,A,5128.9775,N,00819.3163,E,4.10,,060619,,,A*7D
$xyz...
```

gps_decoder2.py

```
#gps-decoder
with open("nmea.txt") as f:
    data = f.read()
print( data)
fundstelle=(data.find("$GPRMC")) # suche schlüsselwort
RMC_str= data[fundstelle : -1]   # fundstelle bis ende
listen_werte= RMC_str.split(',') # komma separierte werte in liste schreiben
for i in range (0, 10):          # eine Anzahl Listenwerte anzeigen
    print( i, "...",listen_werte[i])
zeit= listen_werte[1]            # Listenwert Nr. 1 ist die Zeit
zeit_formatiert= zeit[0:2]+':'+zeit[2:4]+':'+zeit[4:6]
print("Uhrzeit:",zeit_formatiert)
knoten= float(listen_werte[7])   # Nr. 7, String nach Fliesskomma umwandeln
kmh= knoten * 1.852              # Umrechnung Knoten in km/h
kmh0 = format(kmh, '.2f')        # Rundung x Stellen hinterm komma
print("   km/h:",kmh0)
```

Hier das Python Script zum Herausfiltern der GPS-Daten Zeit und Geschwindigkeit

Vergleich: ‚C' mit Python



```
/* Blinkende LED */

void setup()
{
  pinMode(13, OUTPUT);
}


void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

```
from machine import Pin
from time    import sleep

led= Pin(2,Pin.OUT)

while True:
    sleep(.5)
    led.value(1)
    sleep(.5)
    led.value(0)
```


Wiki:

Python ist eine interpretierte, höhere Programmiersprache.
Blöcke werden nicht durch geschweifte Klammern- wie bei ‚C'- , sondern durch Einrückungen strukturiert.

Raspberry Pico , RP2040 Zero

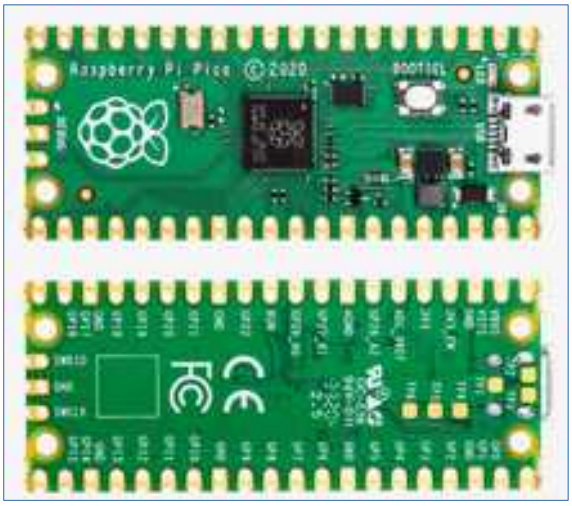
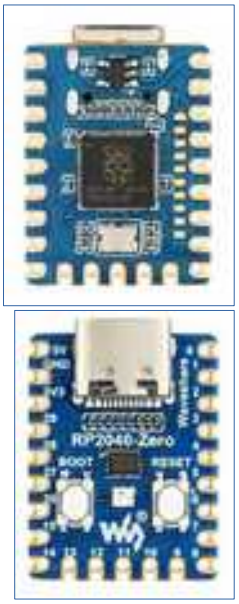


Bild: https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf



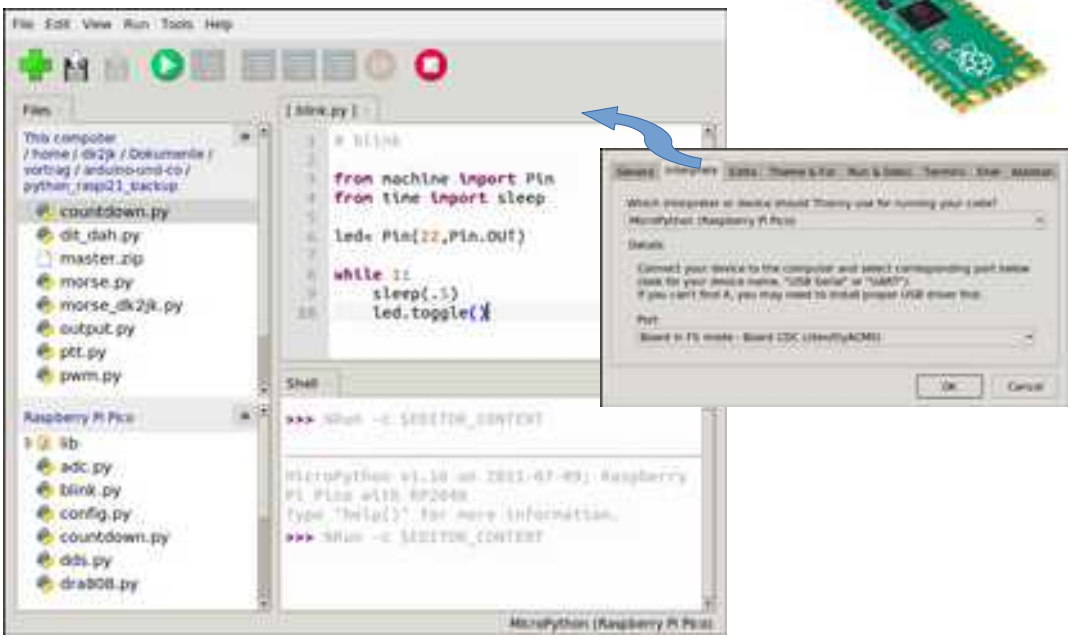
https://www.waveshare.com/rp2040-zero.htm

Raspberry Pico:
Board mit Raspi-CPU, jedoch ohne Betriebssystem, Programmierspache C (z.b. Arduino) oder MicroPython).
Mit Wlan siehe „Raspberry Pi Pico W“
Daten:

- Preiswert (5.95 Euro [Okt 22])
- Viele Pins (26 GPIO)
- 12 Bit ADC
- 2MByte Flash
- 264kB of SRAM,
- 16 × controllable PWM channels
- Micropython

Beschriftung der Pins leider von unten
Kleinere Version : `RP2040 Zero`

Raspberry Pico



Raspberry Pico:
Board mit Raspi-CPU, jedoch ohne Betriebssystem, Programmierspache C (z.b. Arduino) oder MicroPython)

Hier wieder:
Erscheinungsbild in Thonny