

# Programowanie mikrokontrolerów w MicroPython na przykładzie Raspberry Pi Pico

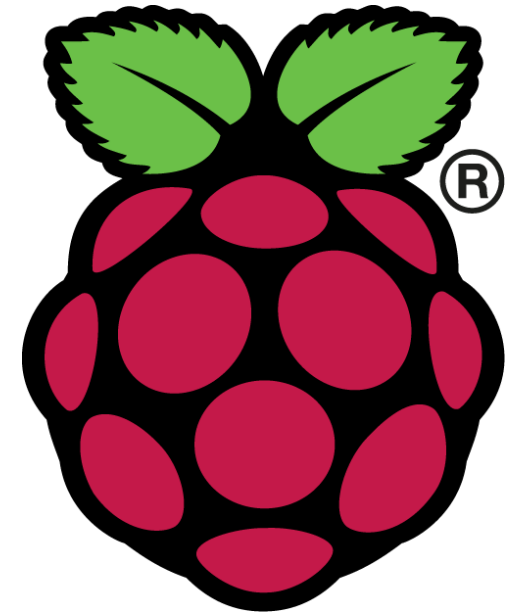
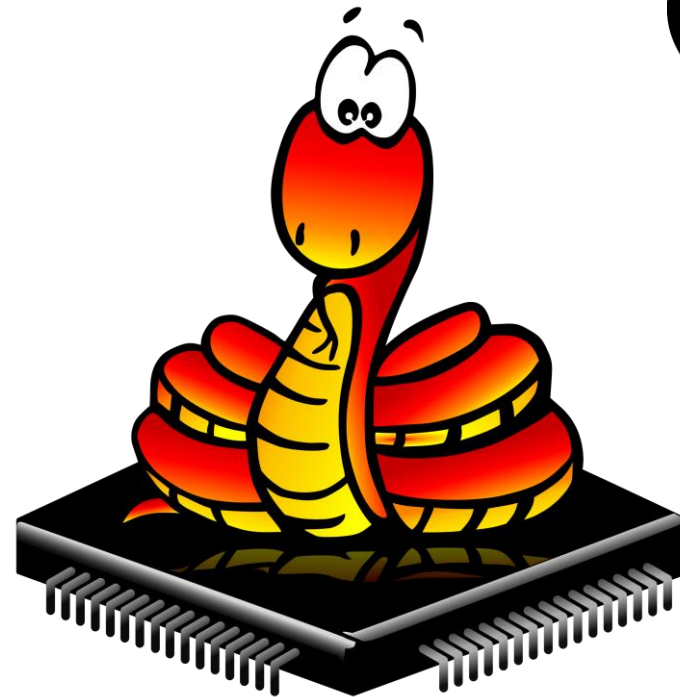
**Filip Marciniak**

Politechnika Poznańska

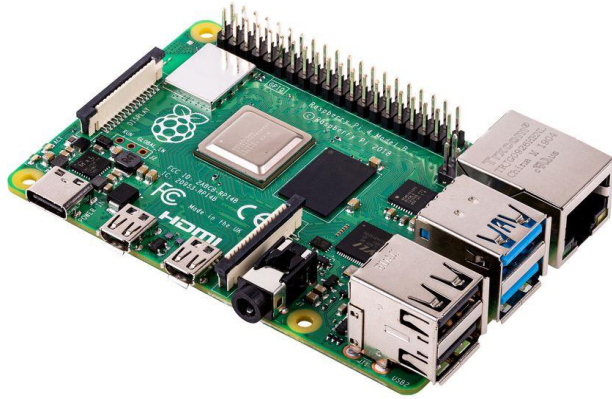
2022

# Plan prezentacji

1. Minikomputery Raspberry Pi
2. Budowa Raspberry Pi Pico
3. MicroPython
4. Programowanie Pico
5. Proste programy w MicroPython
6. Moduły
7. Komunikacja zdalna
8. Podsumowanie



# Minikomputery Raspberry Pi



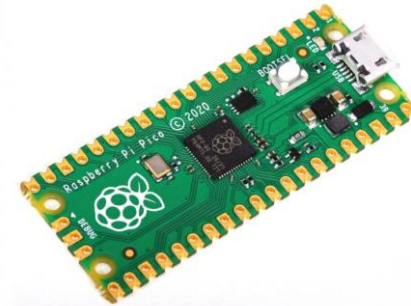
Raspberry Pi 4



Raspberry Pi 3



Raspberry Pi Zero



Raspberry Pi Pico

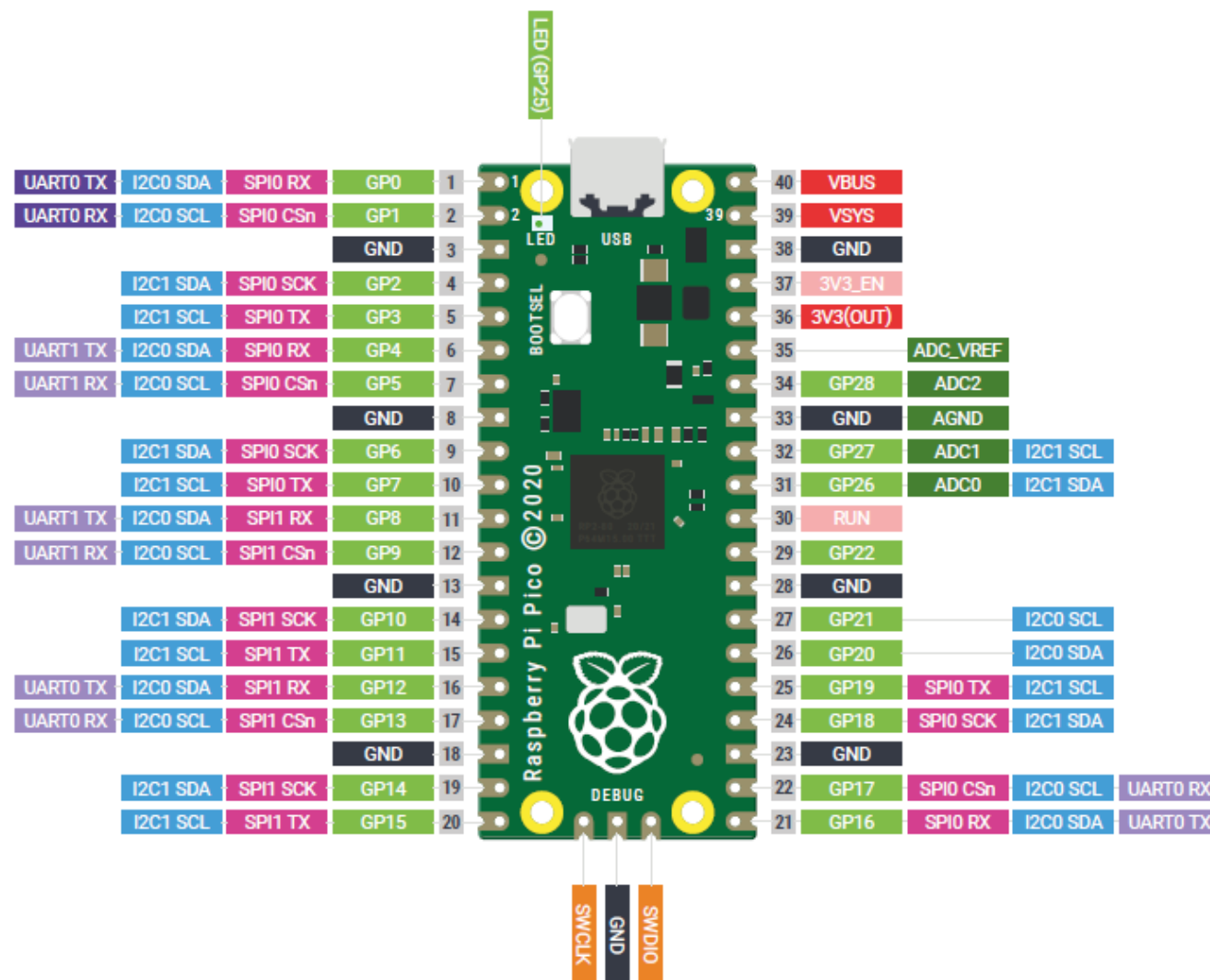
# Dlaczego Raspberry Pi Pico?

Raspberry Pi Pico wyposażony jest w:

1. Autorski mikrokontroler **RP2040**
2. 264kB pamięci SRAM oraz 2 MB pamięci Flash (układ zewnętrzny)
3. 26 pinów GPIO, które pracują z napięciem 3.3V
4. SPI, I2C, UART, ADC, PWM.

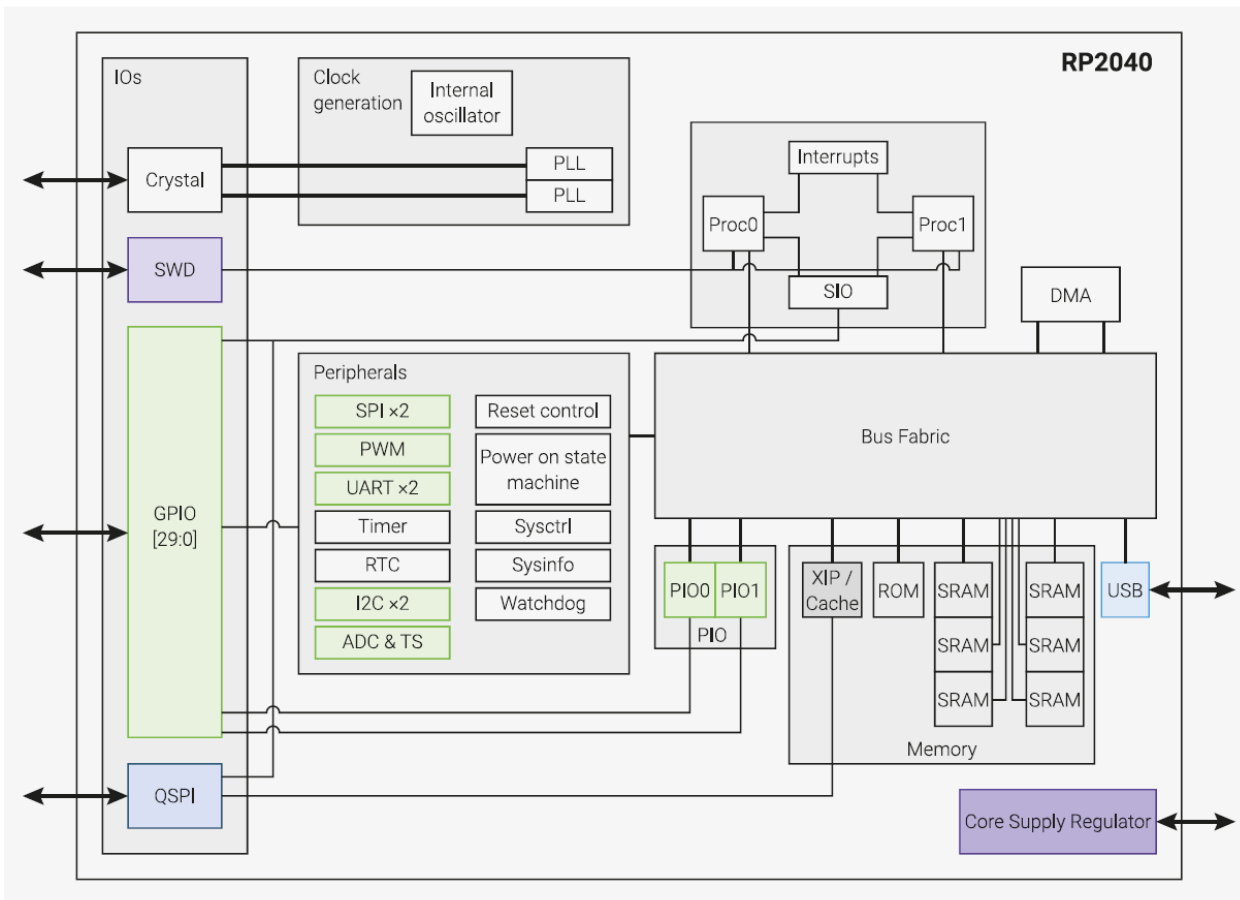
Raspberry Pi Pico programowane jest w języku C/C++ oraz MicroPython poprzez łącze microUSB.

Zasilanie Raspberry Pi Pico odbywa się przy użyciu złącza microUSB przy zasilaniu napięciem 5V.



# Mikrokontroler Raspberry Pi Pico

Figure 2. A system overview of the RP2040 chip

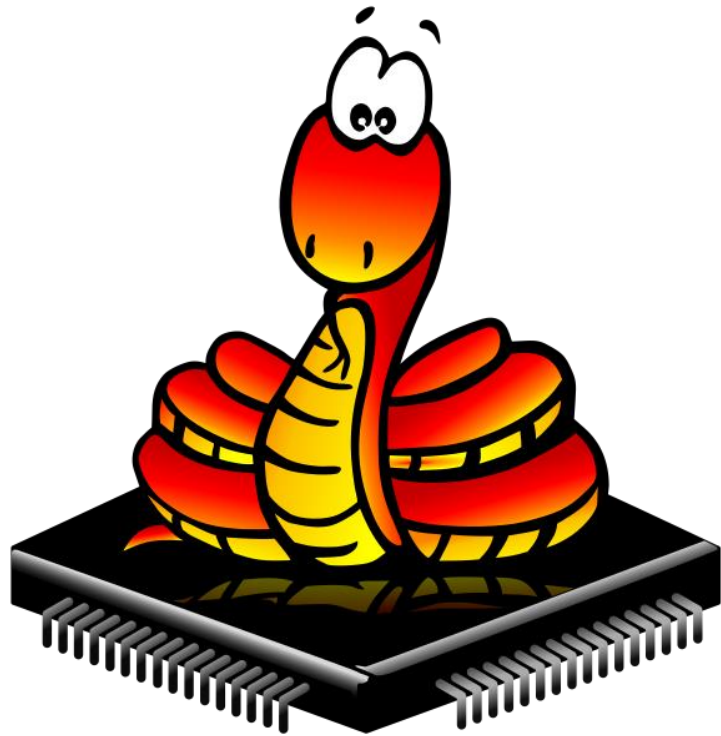


Dwurdzeniowy procesor RP2040 jest oparty na 32-bitowym ARM Cortex M0+ o taktowaniu 133MHz.

MicroPython to wydajna i odchudzona implementacja Pythona 3, przygotowana dla mikrokontrolerów. Zawiera ona niewielki podzbiór bibliotek, które dostępne są w standardowym Pythonie 3. Dzięki temu nie obciąża mikrokontrolera oraz nie zajmuje wiele pamięci.







MicroPython został opracowany z myślą o jak największym podobieństwie do Pythona. Dzięki temu znajomość Pythona gwarantuje umiejętność posługiwania się MicroPythonem.

# Programowanie Raspberry Pi Pico

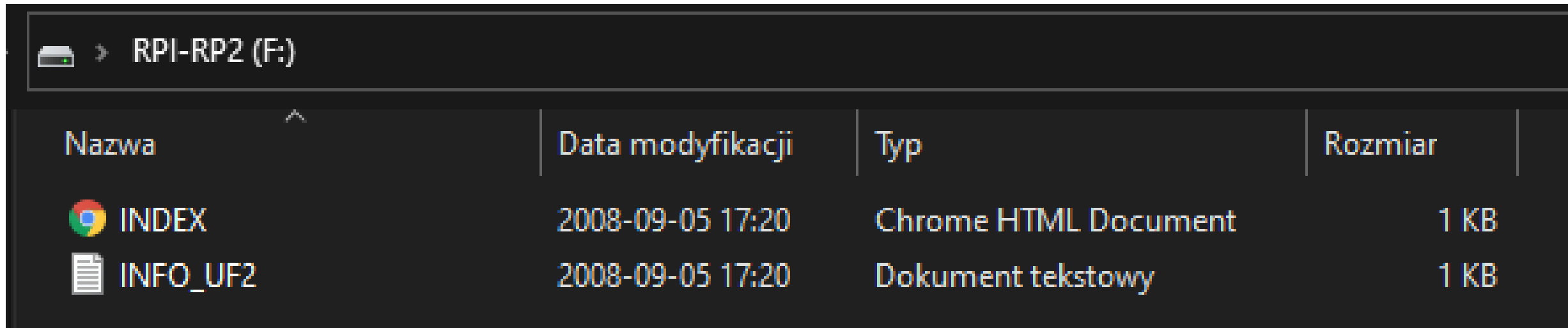
1. Przytrzymanie przycisku BOOTSEL i podłączenie płytki do komputera.
2. Zwolnienie przycisku BOOTSEL po 3 sekundach.







# Programowanie Raspberry Pi Pico

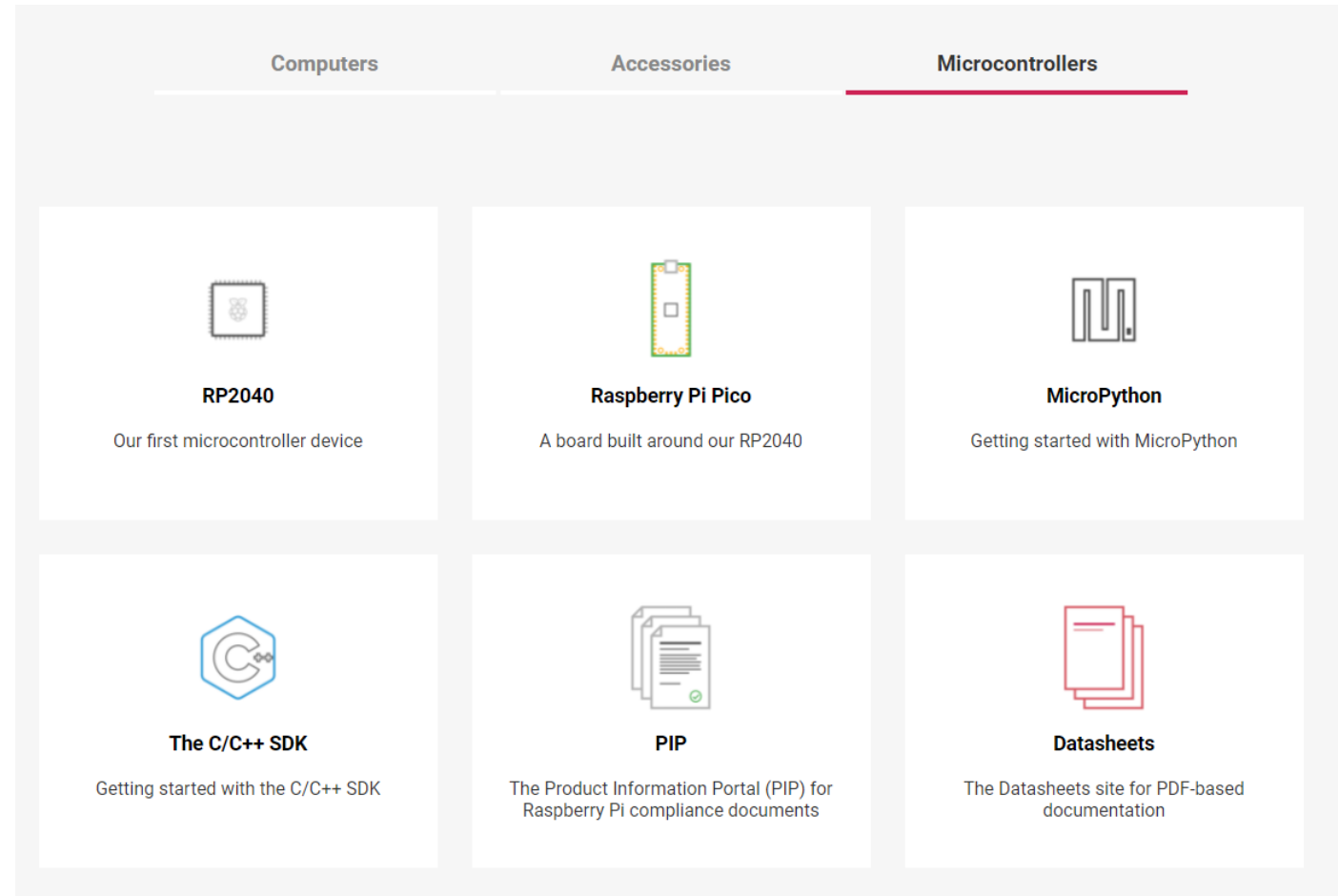
3. Raspberry Pi Pico zostanie wykryte jako urządzenie pamięci masowej RPI-RP2.
4. Otworzenie pliku INDEX i pobranie pliku UF2.



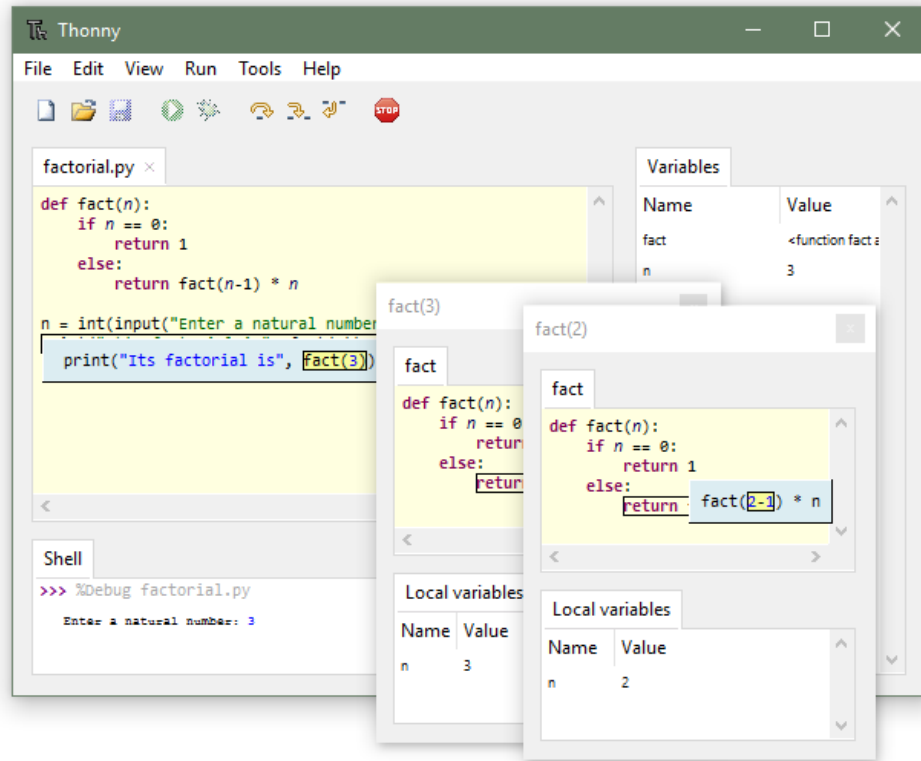
RPI-RP2 (F:)				
Nazwa	Data modyfikacji	Typ	Rozmiar	
 INDEX	2008-09-05 17:20	Chrome HTML Document	1 KB	
 INFO_UF2	2008-09-05 17:20	Dokument tekstowy	1 KB	

# Programowanie Raspberry Pi Pico

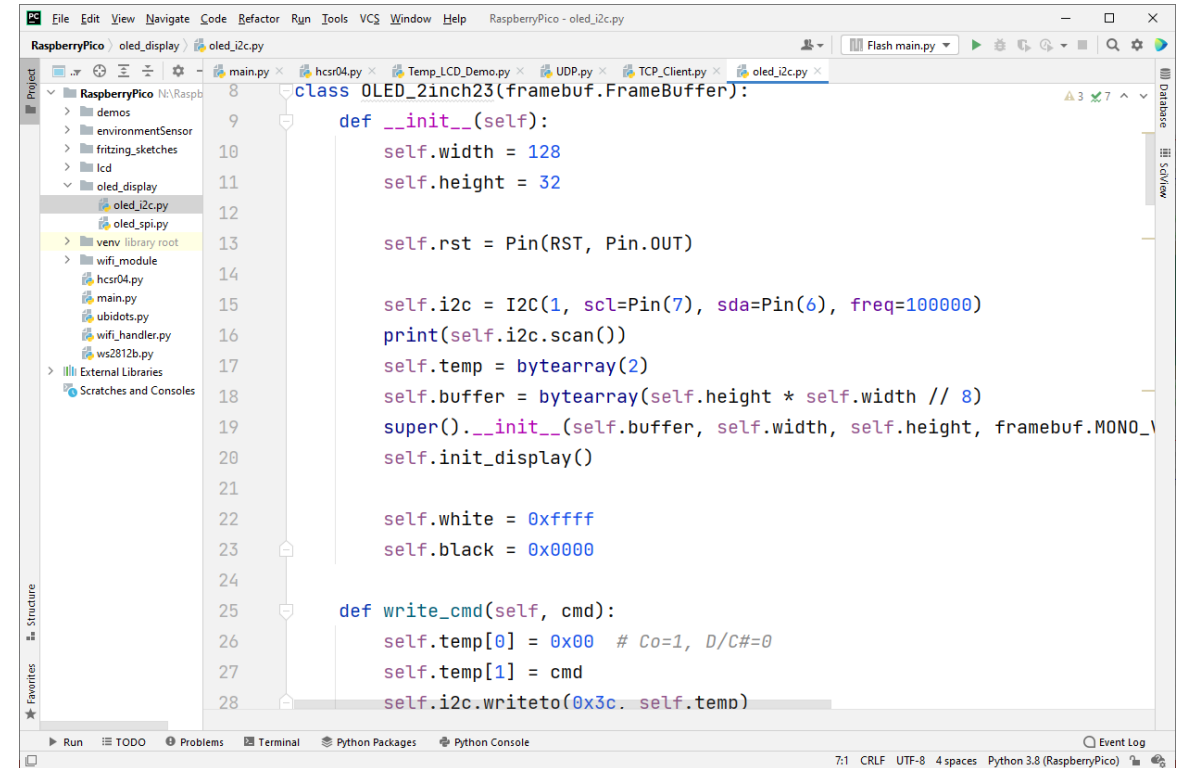
5. Wgranie pliku UF2 na wolumin RPI-RP2.
6. Raspberry uruchomi się ponownie.
7. Dostęp do MicroPython możliwy poprzez port USB.



# Środowisko do pracy z MicroPython

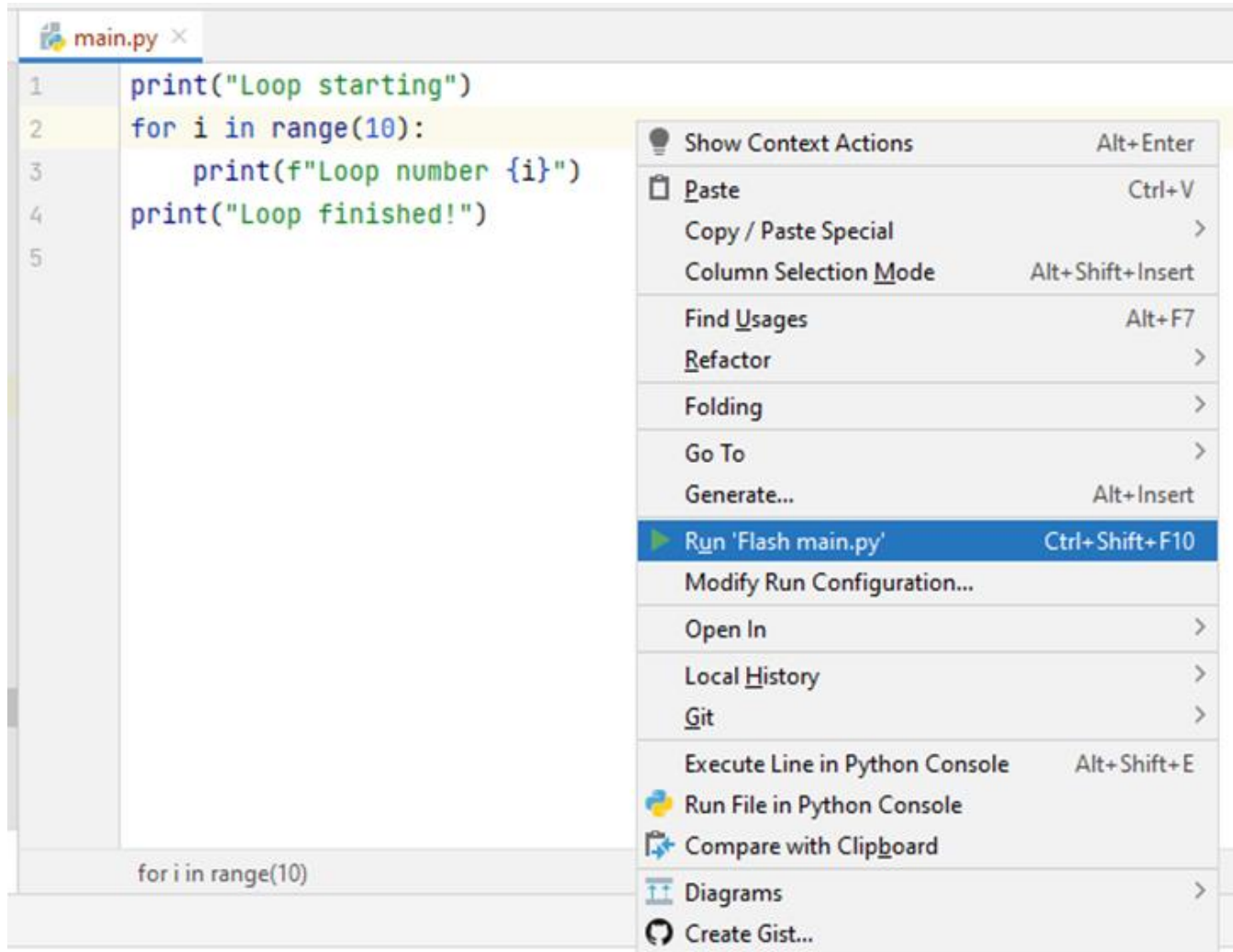


Thonny IDE



JetBrains PyCharm

# Przykład nr 1 w MicroPython



The screenshot shows a MicroPython IDE window with a file named `main.py`. The code in the editor is as follows:

```
1 print("Loop starting")
2 for i in range(10):
3     print(f"Loop number {i}")
4 print("Loop finished!")
5
```

A right-click context menu is open over the code. The menu items and their shortcuts are:

- Show Context Actions (Alt+Enter)
- Paste (Ctrl+V)
- Copy / Paste Special (>)
- Column Selection Mode (Alt+Shift+Insert)
- Find Usages (Alt+F7)
- Refactor (>)
- Folding (>)
- Go To (>)
- Generate... (Alt+Insert)
- Run 'Flash main.py' (Ctrl+Shift+F10)**
- Modify Run Configuration...
- Open In (>)
- Local History (>)
- Git (>)
- Execute Line in Python Console (Alt+Shift+E)
- Run File in Python Console
- Compare with Clipboard
- Diagrams (>)
- Create Gist...

At the bottom of the editor, a snippet of the code `for i in range(10)` is visible.

MicroPython v1.17 on 2021-09-02; Raspberry Pi Pico with RP2040  
Type "help()" for more information.

>>>

MPY: soft reboot

Loop starting

Loop number 0

Loop number 1

Loop number 2

Loop number 3

Loop number 4

Loop number 5

Loop number 6

Loop number 7

Loop number 8

Loop number 9

Loop finished!

# Przykład nr 2 w MicroPython

```
1 import utime
2 print("Loop starting")
3 time_1 = utime.time()
4 while True:
5     if (utime.time() - time_1) > 5:
6         break
7     else:
8         print(f"Time: {utime.time()}")
9         utime.sleep(1)
10 print("Loop finished!")
11
```

```
MicroPython v1.17 on 2021-09-02; Raspberry Pi Pico with RP2040
Type "help()" for more information.
```

```
>>>
```

```
MPY: soft reboot
```

```
Loop starting
```

```
Time: 1609459451
```

```
Time: 1609459452
```

```
Time: 1609459453
```

```
Time: 1609459454
```

```
Time: 1609459455
```

```
Time: 1609459456
```

```
Loop finished!
```



# Blink – dioda wewnętrzna

Program „Blink” pozwalający na mruganie diodą wewnętrzną.

Elementy potrzebne do zbudowania układu:

- Raspberry Pi Pico

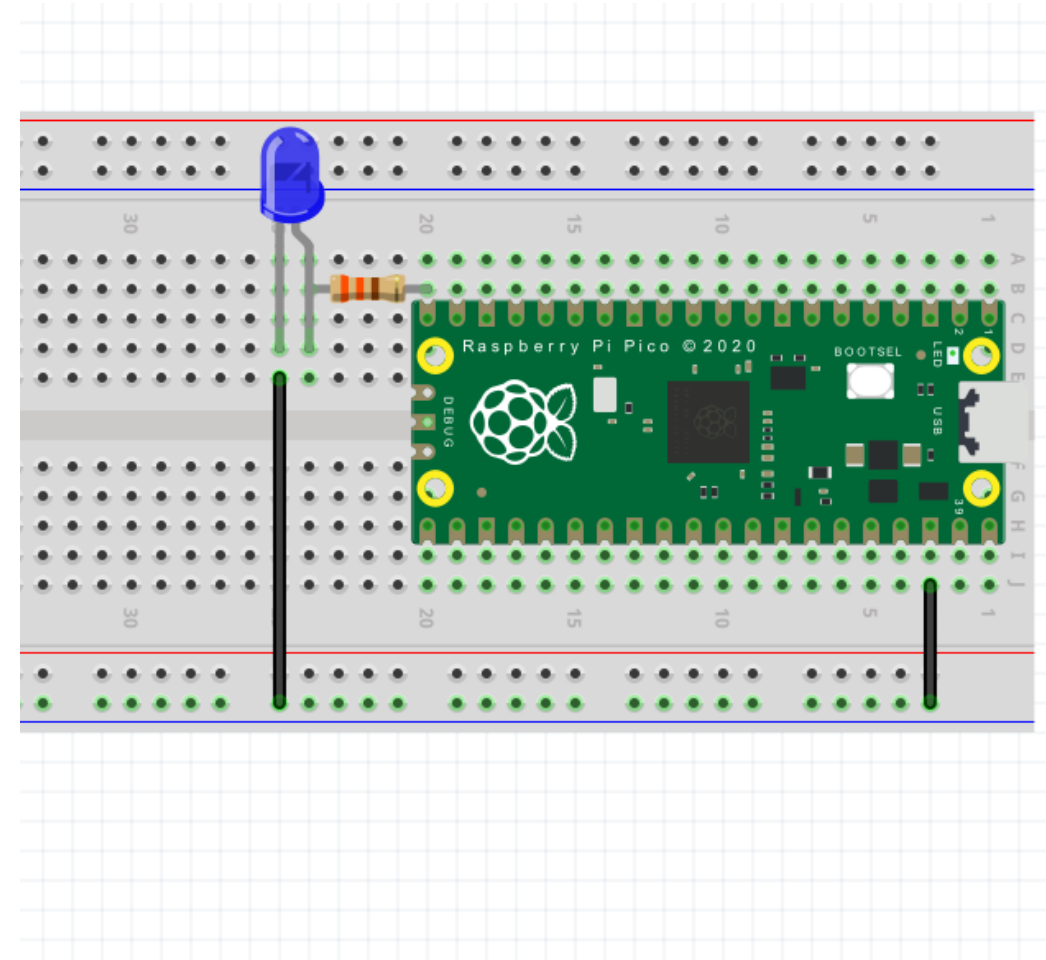
```
1  import machine
2  import utime
3
4  led_internal = machine.Pin(25, machine.Pin.OUT)
5
6  while True:
7      led_internal.toggle()
8      utime.sleep_ms(200)
9  |
```

# Przykład Blink – dioda zewnętrzna

Program „Blink” pozwalający na mruganie diodą.

Elementy potrzebne do zbudowania układu:

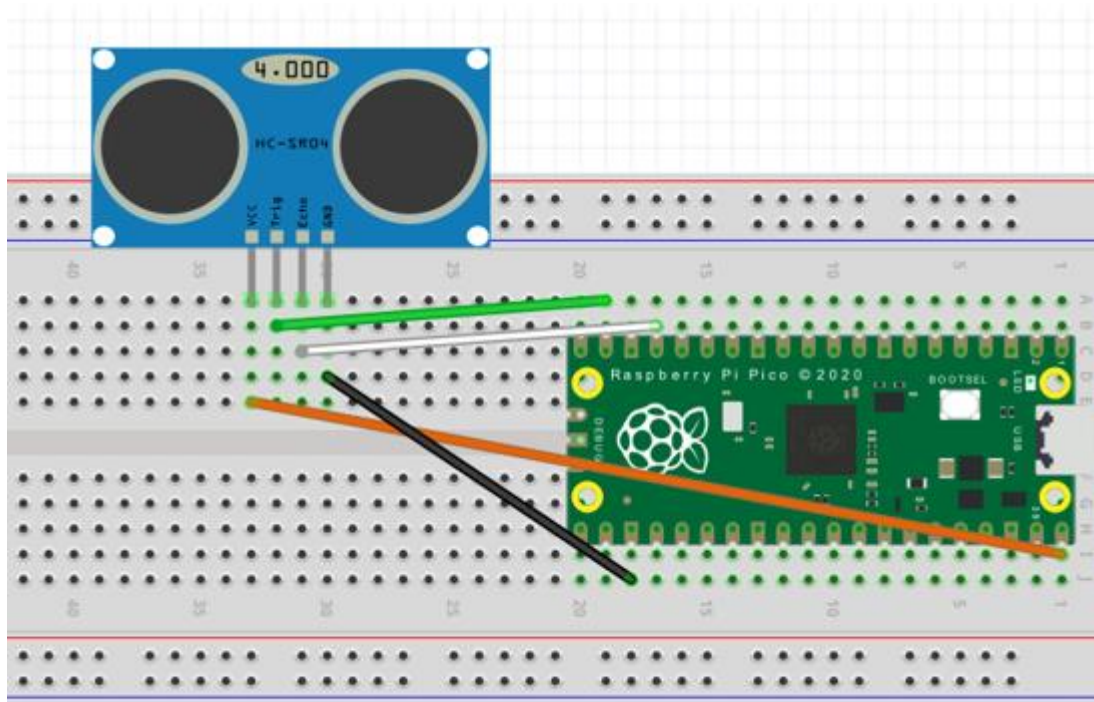
- Raspberry Pi Pico
- Płytki stykowa
- Rezystor 330  $\Omega$
- Dioda świecąca
- 2 przewody M-M



# Blink – dioda zewnętrzna – kod programu

```
1  import machine
2  import utime
3
4  led_external = machine.Pin(15, machine.Pin.OUT)
5
6  while True:
7      led_external.toggle()
8      utime.sleep_ms(200)
9
```

# Czujnik odległości



```
1  from machine import Pin
2  import utime
3
4  trigger = Pin(14, Pin.OUT)
5  echo = Pin(13, Pin.IN)
6
7  while True:
8      trigger.low()
9      utime.sleep_us(5)
10     trigger.high()
11     utime.sleep_us(5)
12     trigger.low()
13     while echo.value() == 0:
14         signal_off = utime.ticks_us()
15     while echo.value() == 1:
16         signal_on = utime.ticks_us()
17     distance = ((signal_on - signal_off) * 0.0343) / 2
18     print(f"The distance from object is {distance}cm")
19
```

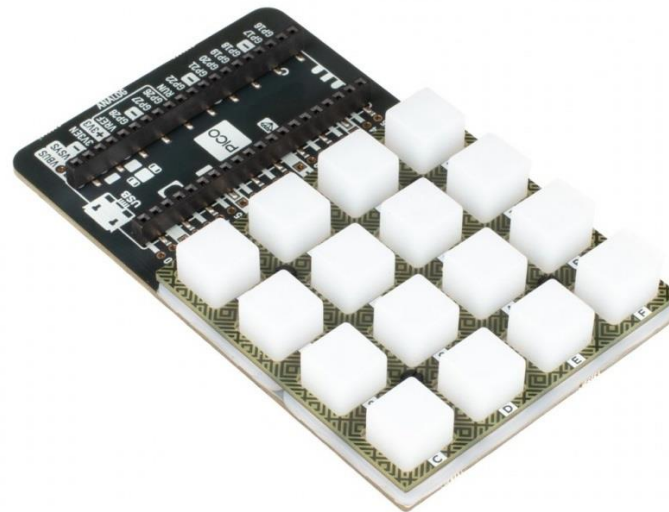
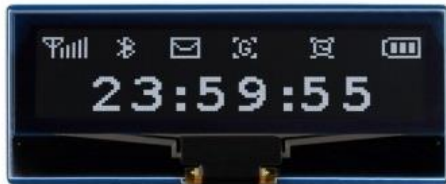
# Czujnik odległości – rezultat programu

```
MicroPython v1.17 on 2021-09-02; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MPY: soft reboot
The distance from object is 5.1107cm
The distance from object is 5.2136cm
The distance from object is 5.5223cm
The distance from object is 9.8784cm
The distance from object is 9.89555cm
The distance from object is 9.55255cm
The distance from object is 12.5538cm
The distance from object is 12.5538cm
The distance from object is 12.53665cm
The distance from object is 15.5722cm
The distance from object is 16.24105cm
The distance from object is 17.23575cm
The distance from object is 16.5669cm
The distance from object is 12.91395cm
The distance from object is 11.91925cm
```



# Rozszerzenia oraz nakładki

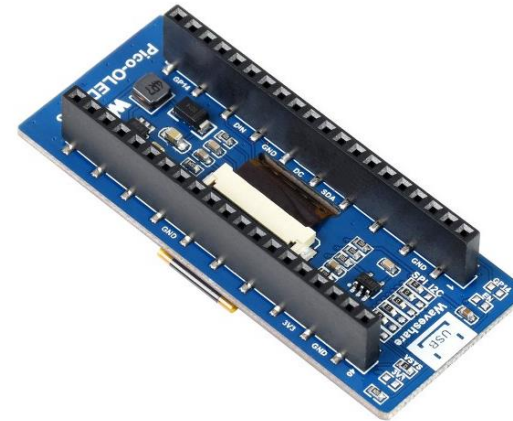
Budowanie projektów z Pico jest łatwe dzięki zaprojektowanym specjalnie modułom dostępnym na rynku. Są to wyświetlacze, czujniki środowiskowe, klawiatury czy nawet moduły do komunikacji zdalnej.



# Wyświetlacz OLED

Wyświetlacz OLED o rozmiarze 2.23" firmy Waveshare został wyposażony w 4-przewodową komunikację SPI oraz interfejs I2C (do wyboru).

Wyświetlacz jest rozdzielczości 128x32 i wyświetla 2 różne kolory – czarny albo biały.



# Montaż wyświetlacza na Raspberry Pi Pico

Aby skorzystać z możliwości modułu, należy nałożyć go na płytke.

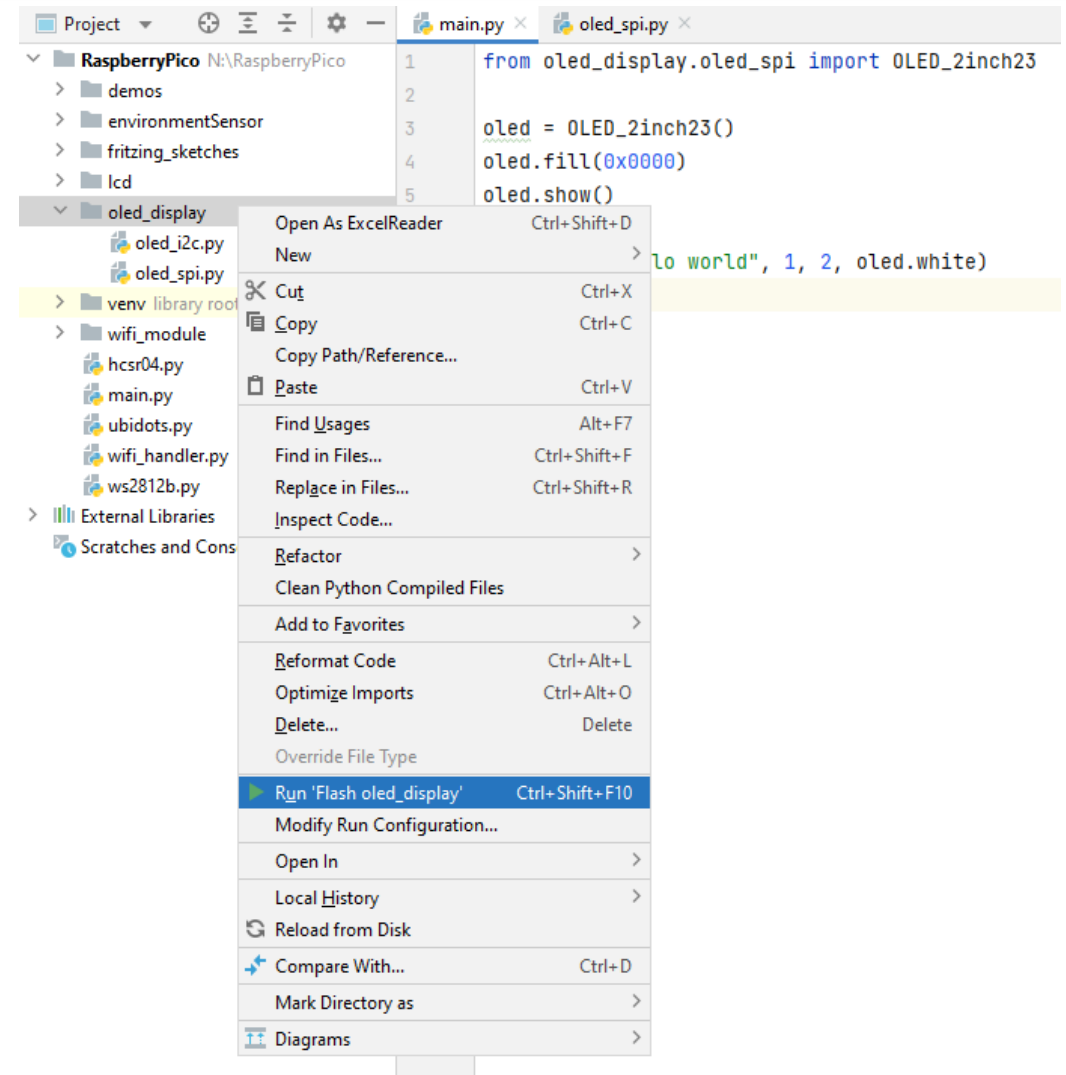


# Wyświetlanie napisu „Hello World”



```
1 from machine import Pin, SPI
2 import framebuffer
3 import time
4
5 DC = 8
6 RST = 12
7 MOSI = 11
8 SCK = 10
9 CS = 9
10
11
12 class OLED_2inch23(framebuffer.FrameBuffer): ...
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
```

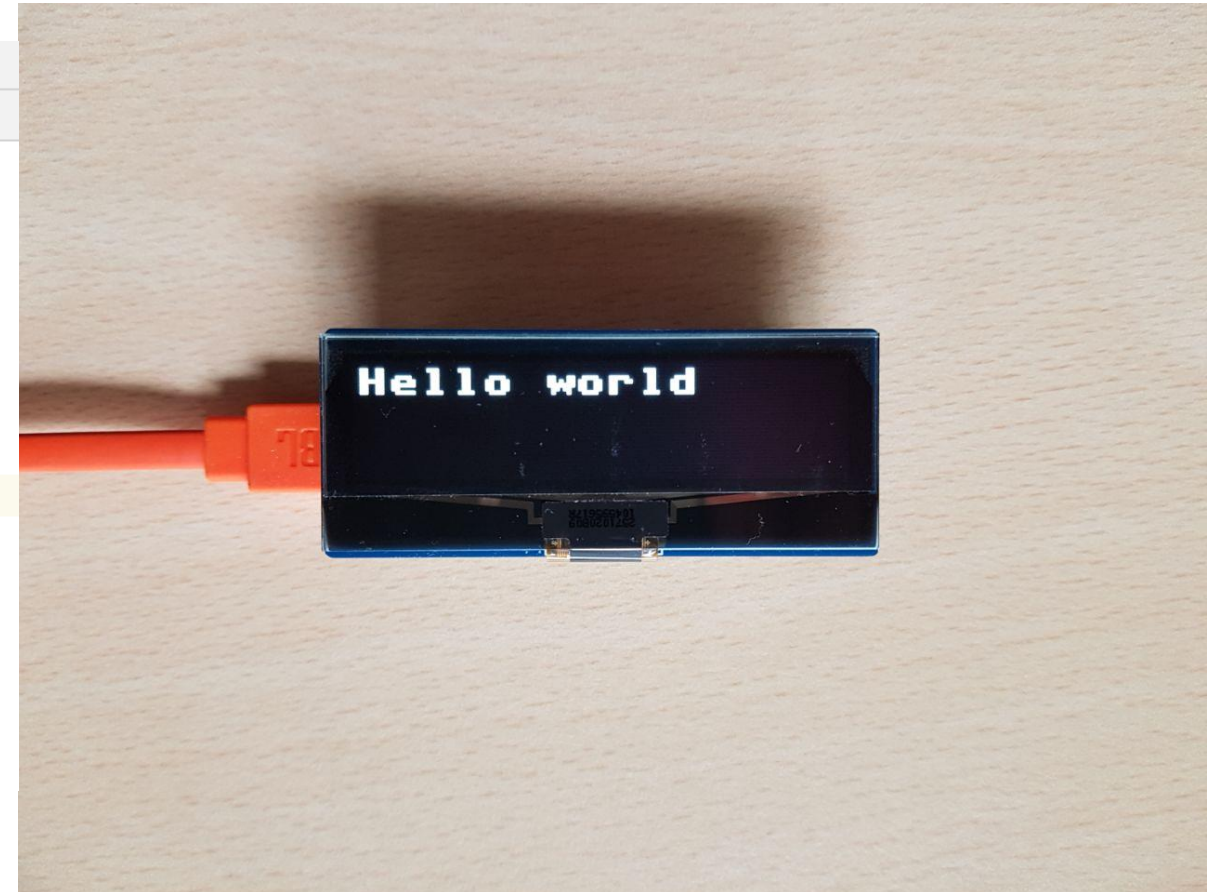
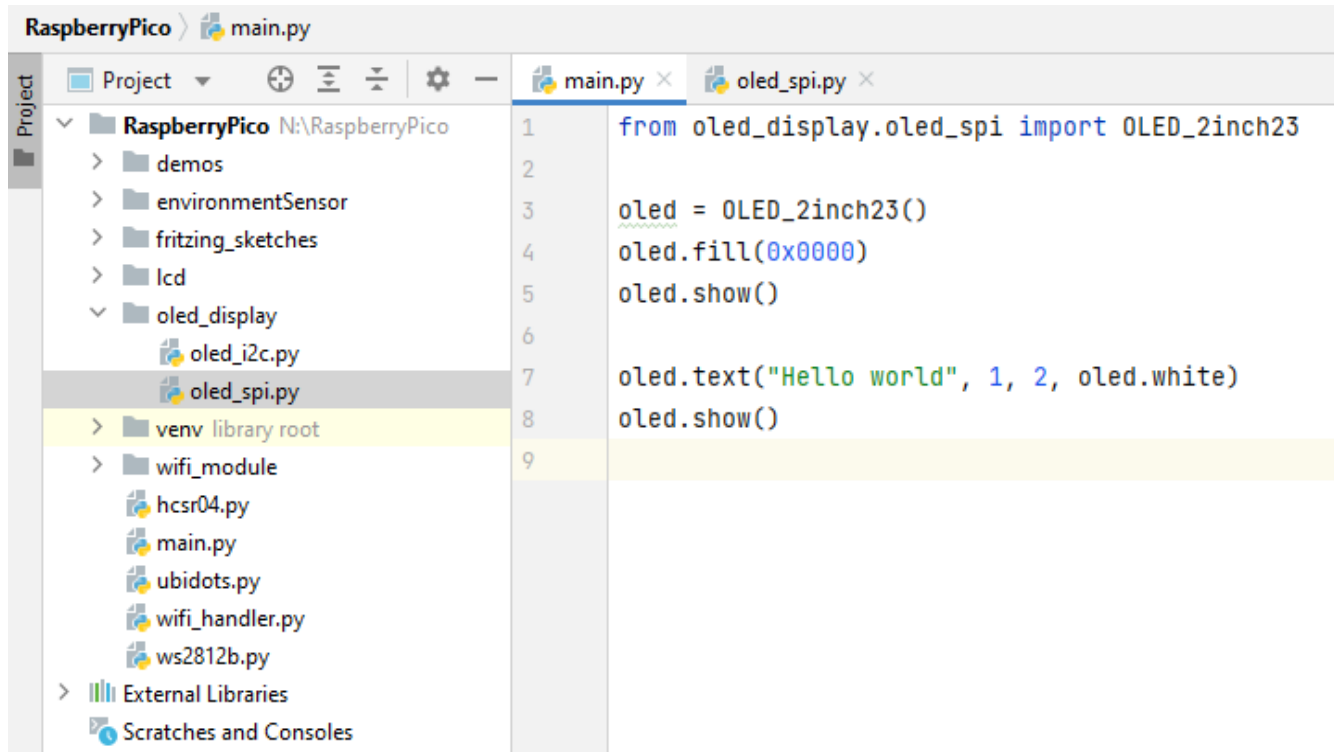
Program do obsługi wyświetlacza ze strony producenta



```
1 from oled_display.oled_spi import OLED_2inch23
2
3 oled = OLED_2inch23()
4 oled.fill(0x0000)
5 oled.show()
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
```

Wgrywanie całego folderu na Pico

# Wyświetlanie napisu „Hello World”



Rezultat „Hello world” (po Run main)



# Wyświetlenie pomiaru odległości na wyświetlaczu

Cel miniprojektu:

Wyświetlenie przy pomocy **wyświetlacza OLED** odległości od przedmiotu zmierzonej przy pomocy czujnika ultradźwiękowego **HC-SR04**.

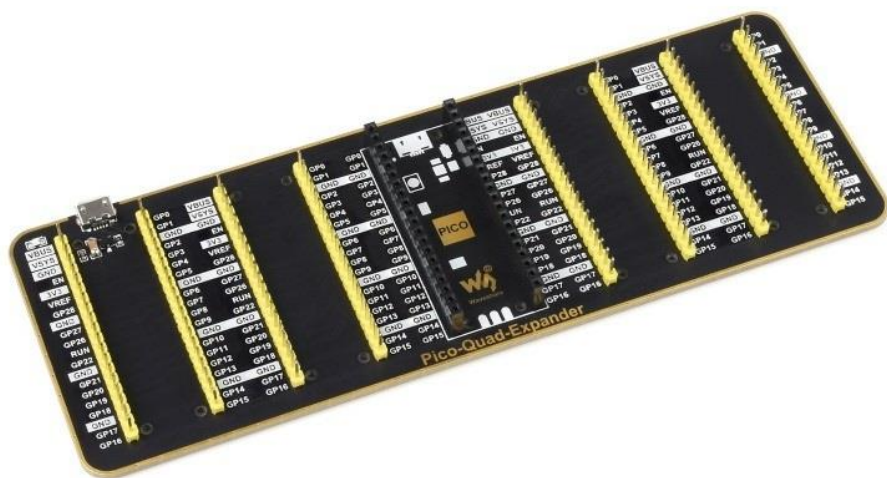
Napotkany problem:

Nakładka OLED połączona bezpośrednio z Pico uniemożliwia podłączenie innych urządzeń.

Rozwiązanie:

Płytki rozszerzeń

# Płytki rozszerzeń

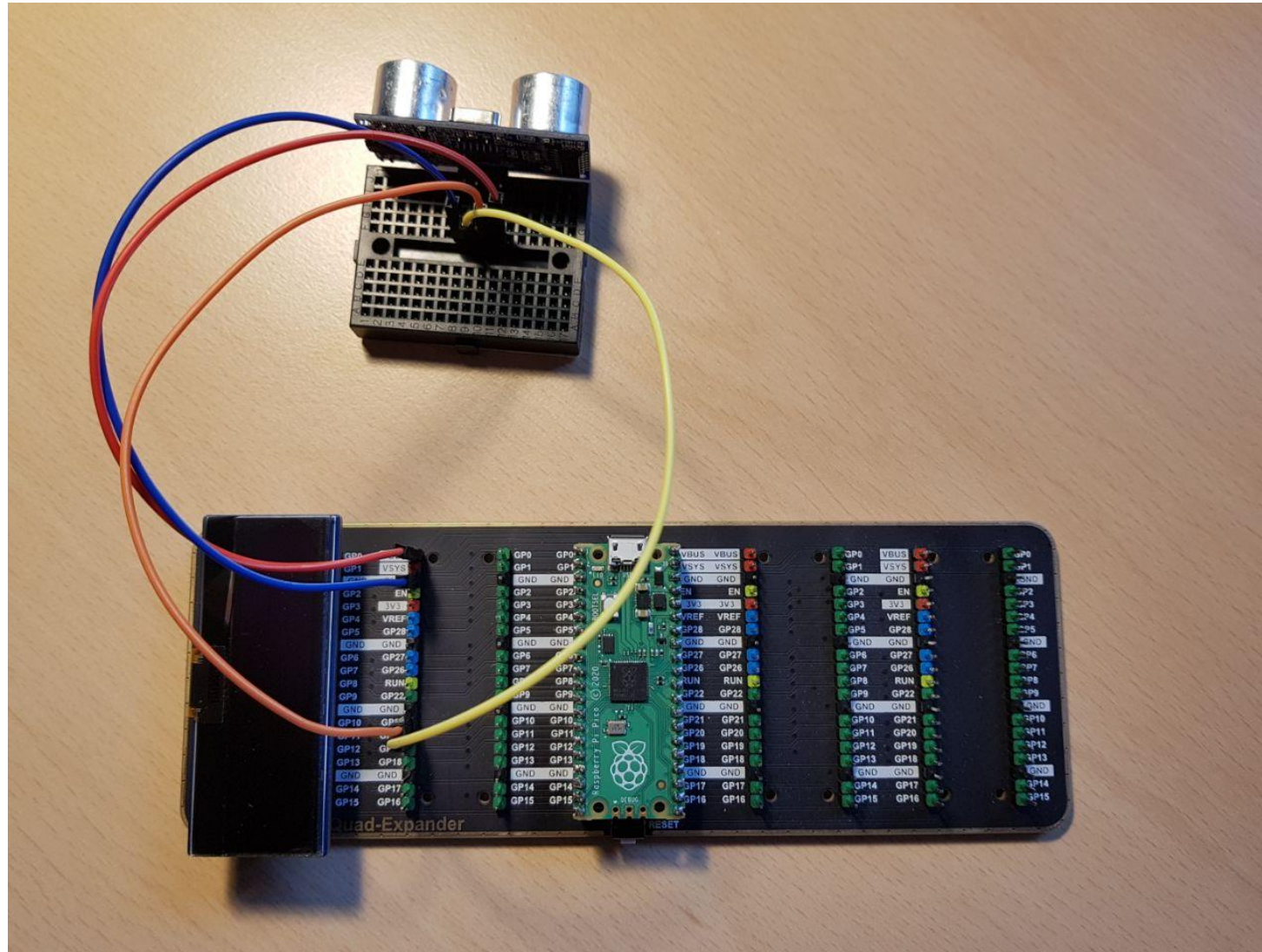


Pusta płytki rozszerzeń



Płytki rozszerzeń z nałożonymi elementami

# Wyświetlanie odległości od przedmiotu



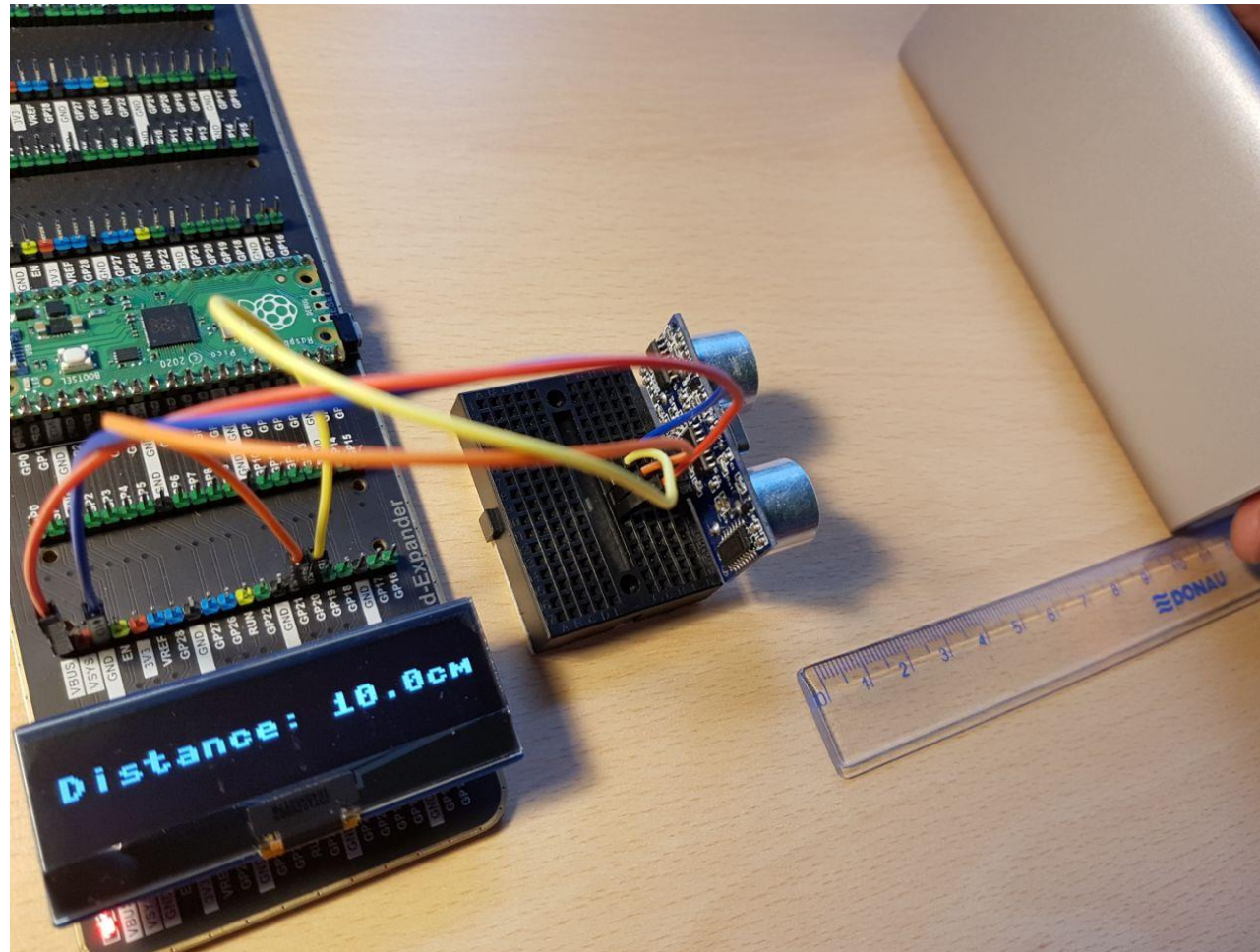
Połączenie Raspberry Pi Pico, wyświetlacza oraz czujnika przy pomocy płytki rozszerzeń.

# Wyświetlanie odległości od przedmiotu

```
main.py × hcsr04.py × oled_spi.py ×
1  from oled_display.oled_spi import OLED_2inch23
2  from hcsr04 import HCSR04
3  import utime
4
5  oled = OLED_2inch23()
6
7  sensor = HCSR04(trigger_pin=21, echo_pin=20)
8
9  while True:
10     distance = sensor.distance_cm()
11     oled.fill(oled.black)
12     oled.text(f"Distance: {distance:.1f}cm", 0, 12, oled.white)
13     oled.show()
14     utime.sleep_ms(300)
15
```



# Wyświetlanie odległości od przedmiotu



Przykładowe działanie programu



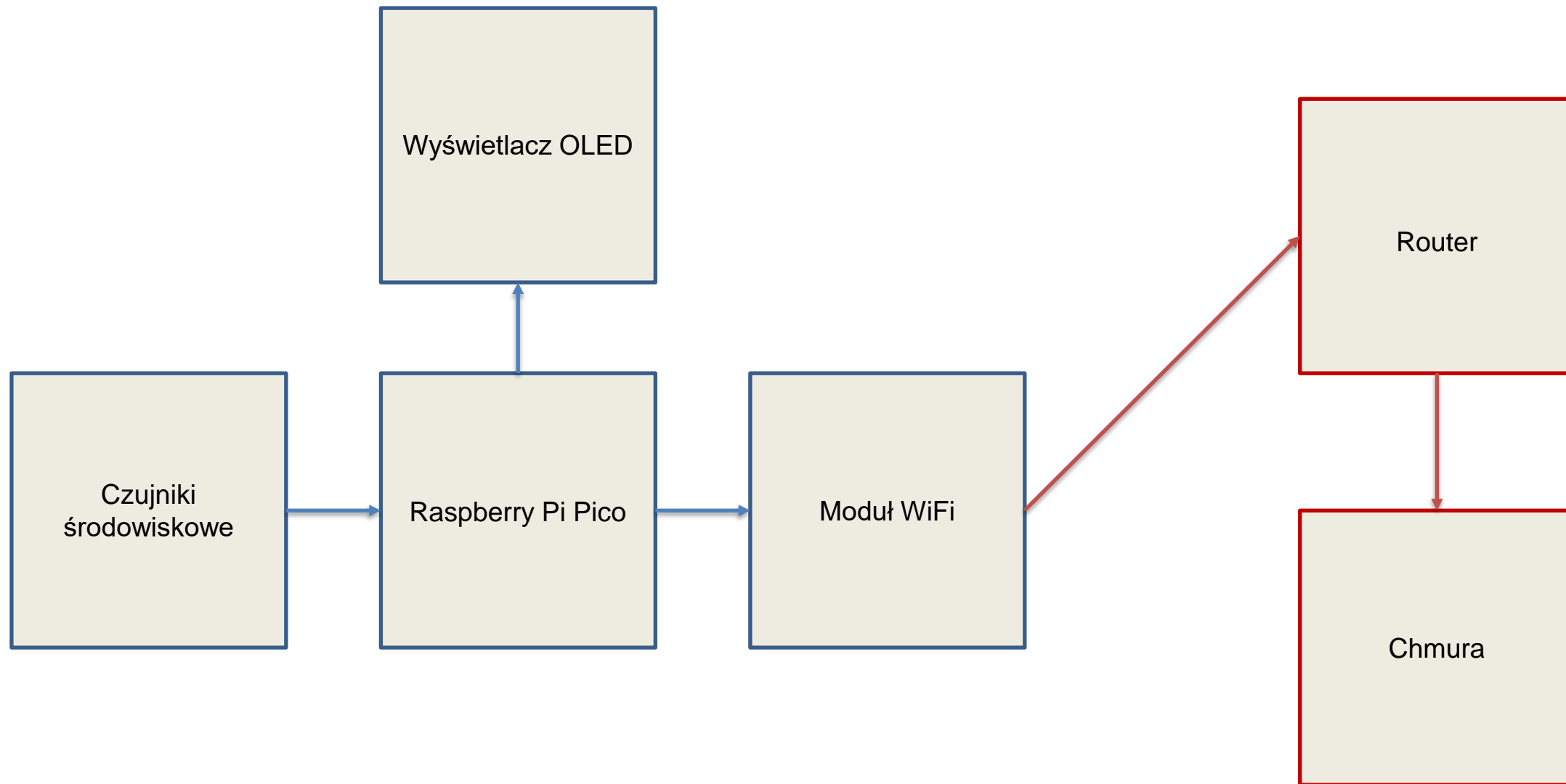
# Komunikacja zdalna

Do komunikacji przez Wi-Fi zastosowano moduł ESP8266 zamontowany na rozszerzeniu Pico Wi-Fi HAT producenta Sb-Components.

Moduł posiada również własny wyświetlacz o wielkości 1.14''.



# Stacja pogodowa - schemat

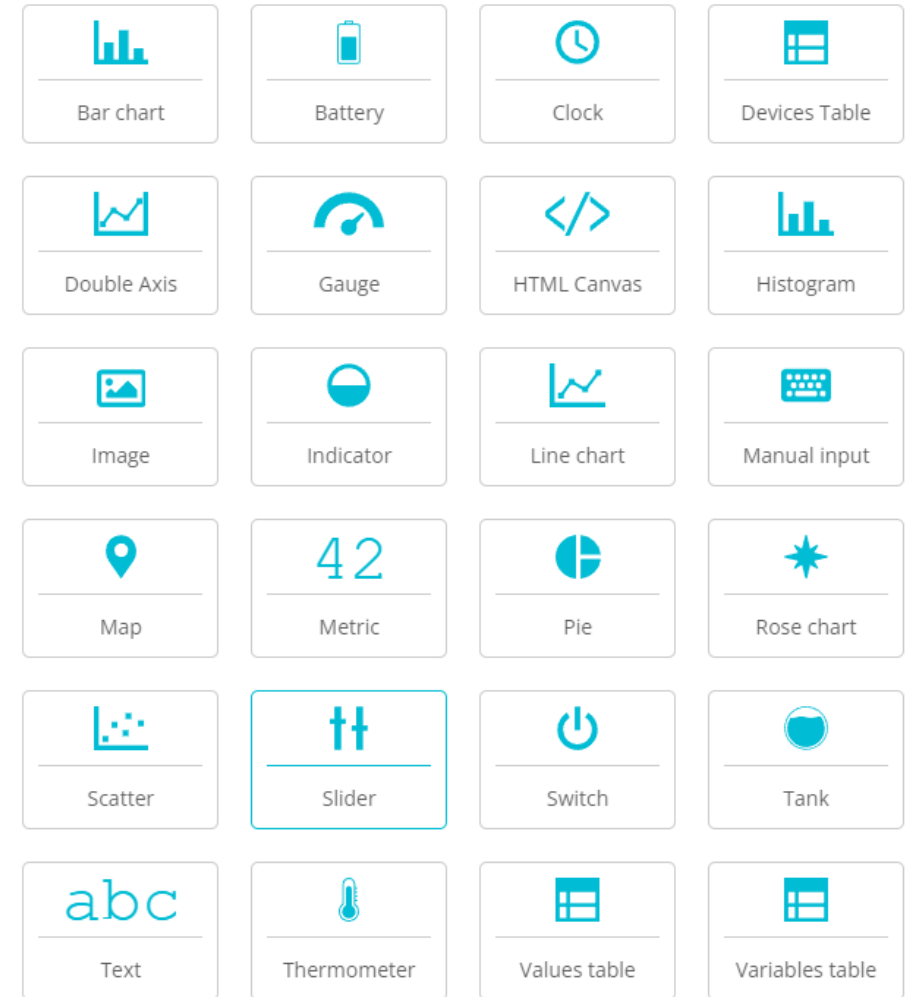




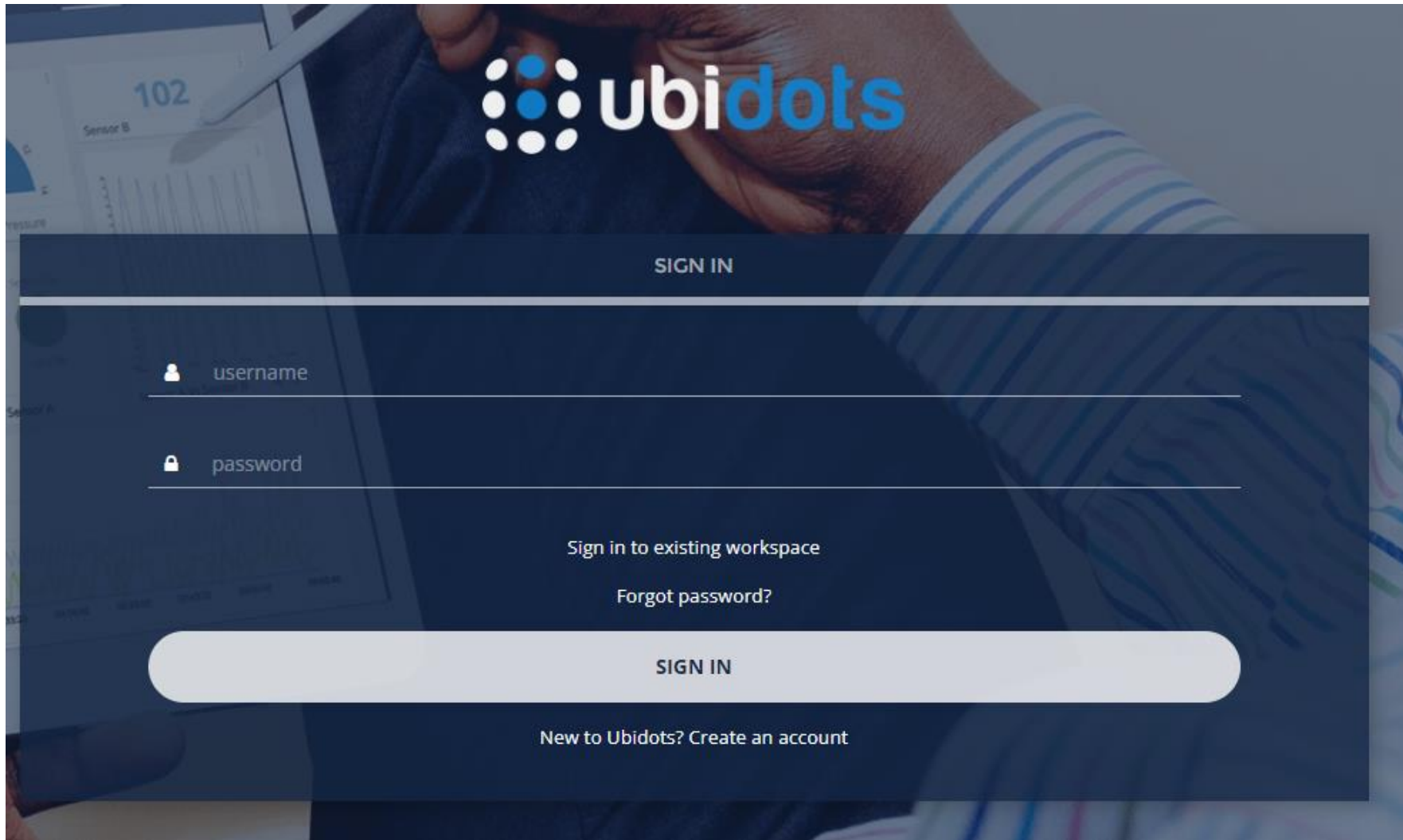
Dzięki ESP8266 można wysyłać dane na zewnętrzny serwer. Przykładem takiego serwera jest strona Ubidots.

# Ubidots


- Ubidots pozwala na zbieranie danych z wielu urządzeń IoT.
- Dane wysyłane są na serwer przez TCP albo HTTP przy pomocy API.
- Strona umożliwia również tworzenie Dashboardów, na których zebrane dane może przedstawiać w ciekawy sposób, np. na wykresach i diagramach.





# Połączenie z Ubidots


The image shows the Ubidots login page. At the top, the Ubidots logo is displayed in white and blue. Below the logo, there is a dark blue rectangular box containing the login form. The form has a 'SIGN IN' header, followed by two input fields: 'username' with a person icon and 'password' with a lock icon. Below these fields are two links: 'Sign in to existing workspace' and 'Forgot password?'. A large, light blue 'SIGN IN' button is positioned below the links. At the bottom of the box, there is a link that says 'New to Ubidots? Create an account'. The background of the slide features a person's hand holding a pen over a document with a line graph and the number '102'.

# Połączenie z Ubidots

 My account



 **API Credentials**

 Plans and Billing

 Usage


## API Key






Use your API key to create temporary Tokens through the [Auth endpoint](#). This API key can't be used to authenticate API requests; its only purpose is to generate Tokens.

API Key 1 .....  

## Tokens

Use Tokens to authenticate API requests. You can create Tokens from this interface, or [through our API](#).



<u>NAME</u>	<u>TOKENS</u>	<u>RATE LIMIT</u>	<u>ACTIONS</u>
Default token	.....  	1 req/1s	  

ROWS PER PAGE 10 ▾ < >



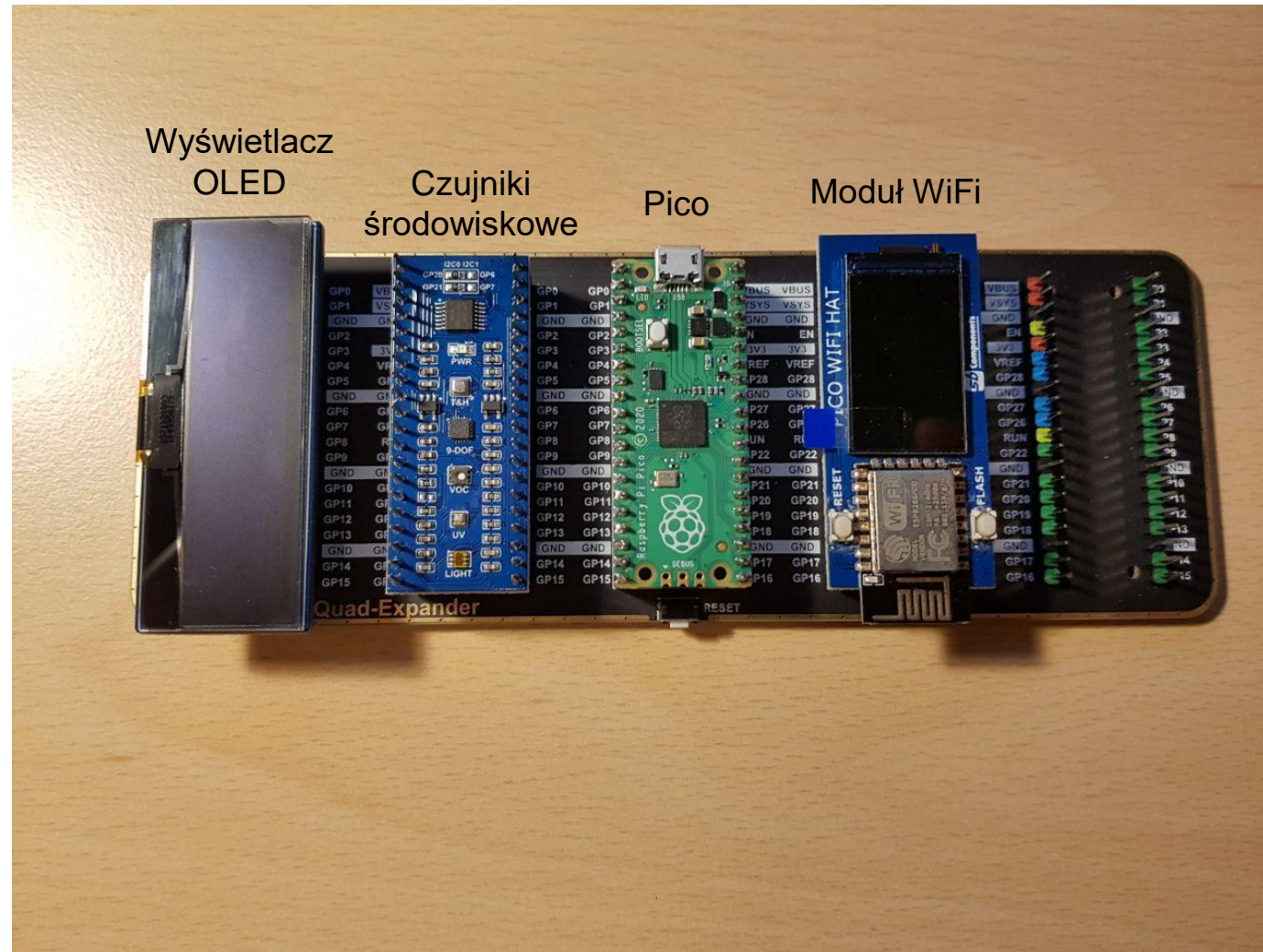
# Pico Environment Sensor

Moduł z czujnikami środowiskowymi firmy Waveshare posiada aż 5 czujników:

- BME280 – ciśnienie atmosferyczne, temperatura i wilgotność
- ICM20948 – żyroskop, przyspieszenie oraz magnetometr
- LTR390 – czujnik UV
- TSL2591 – czujnik natężenia światła
- SGP40 – czujnik jakości powietrza



# Stacja pogodowa – budowa układu



Płytki rozszerzeń z zamontowanymi modułami i Raspberry Pi Pico

# Stacja pogodowa - program

Program stacji pogodowej składa się z:

1. Programu głównego `main.py`
2. Programu `wifi_handler.py`
3. Programu `BME280.py`
4. Programu `oled_spi.py`

# Stacja pogodowa – główna pętla programu

```
36 try:
37     while True:
38         utime.sleep(3)
39         pressure = round(bme280.readData()[0], 2)
40         temperature = round(bme280.readData()[1], 2)
41         humidity = round(bme280.readData()[2], 2)
42
43         screen.fill(screen.black)
44         screen.text(f'Pressure: {pressure}', 1, 2, screen.white)
45         screen.text(f'Temp: {temperature}', 1, 12, screen.white)
46         screen.text(f'Humidity: {humidity}', 1, 22, screen.white)
47         screen.text(f'Humidity:', 1, 22, screen.white)
48
49         screen.show()
50
51         variables = {
52             'pressure': pressure,
53             'temperature': temperature,
54             'humidity': humidity
55         }
56
57         data_sender.send_tcp(variables)
58 except KeyboardInterrupt:
59     exit()
```

# Odczyt danych w Ubidots

**raspberry-pi-pico-bme**

**Description**  
Change description

**API Label** ⓘ  
raspberry-pi-pico-bme

**ID** ⓘ  
61cf229c1d84725ac1ff7613

**Token**  
.....

**Tags**  
Add new tag

**Last activity**  
a few seconds ago

**Device type** ★  
Set Device Type

**Location** ⓘ

**Mode** ⓘ  
Auto

**46.38**  
humidity  
Last activity: a few seconds ago

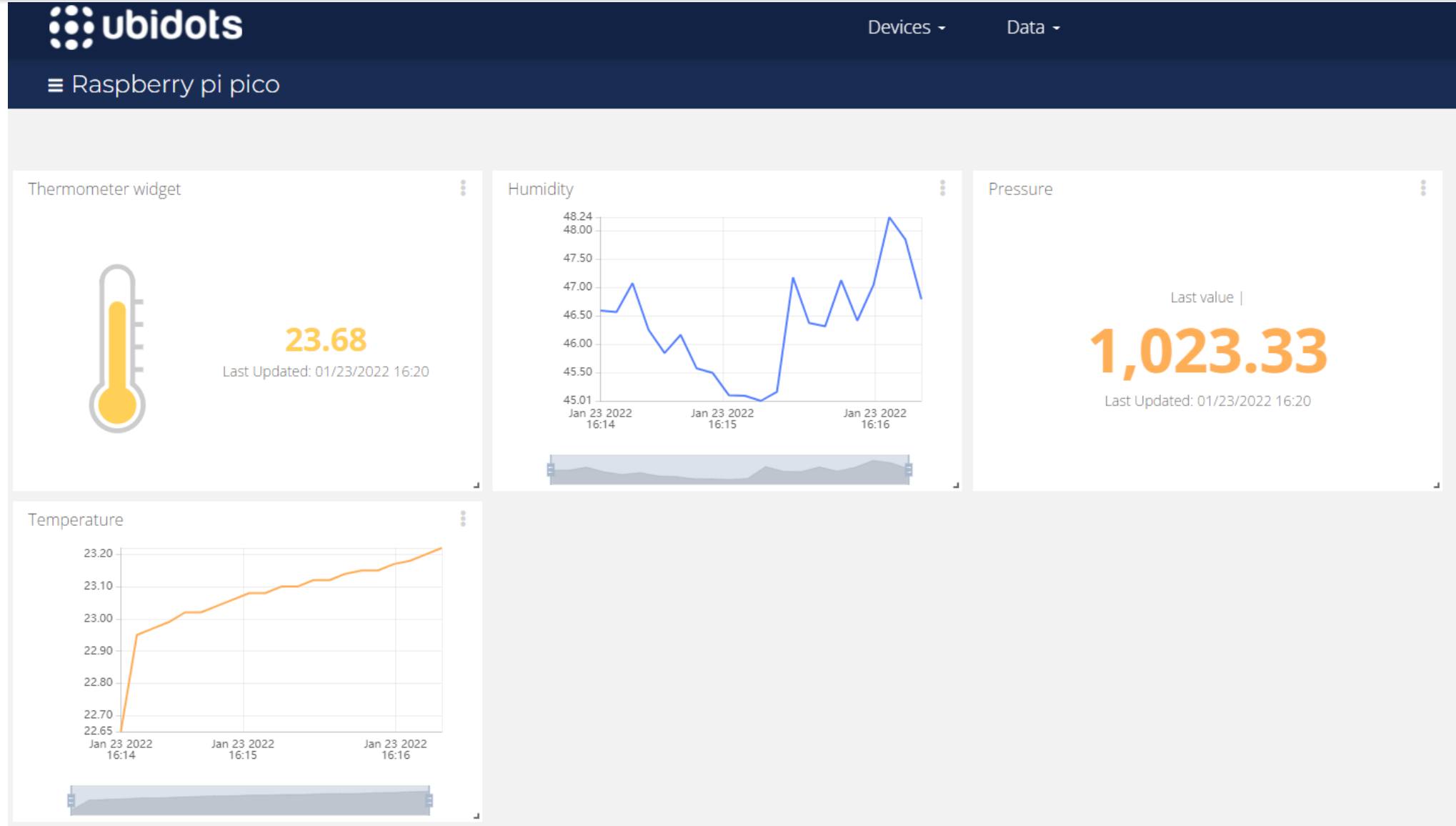
**1,023.46**  
pressure  
Last activity: a few seconds ago

**23.12**  
temperature  
Last activity: a few seconds ago

**Add Variable**

**VARIABLES PER PAGE** 30

# Dashboard w Ubidots



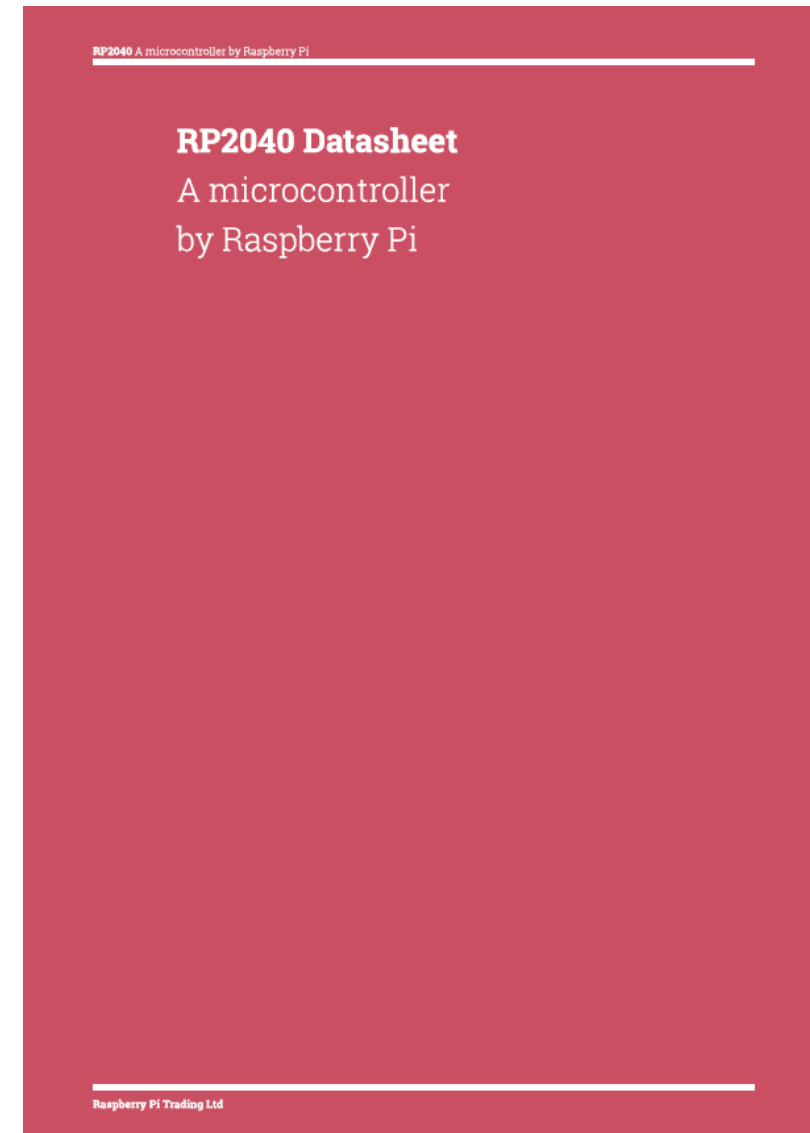
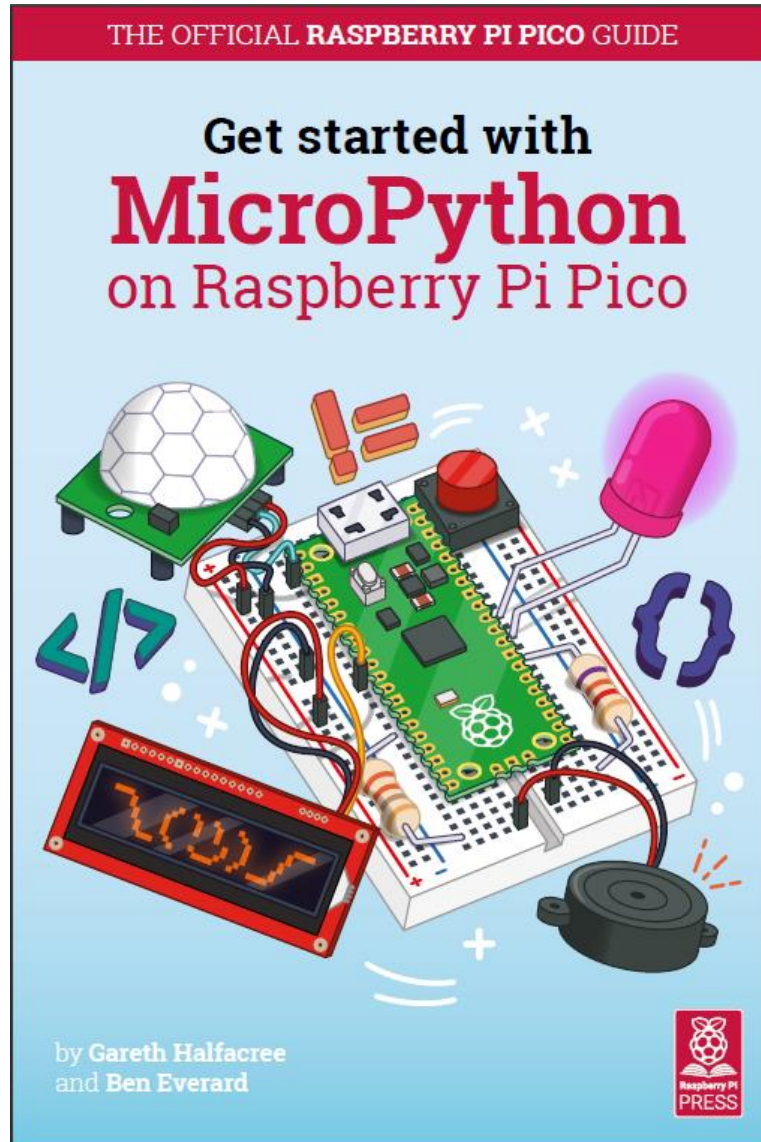


- Układ Raspberry to alternatywa dla Arduino UNO:
  - SRAM: 264kB pamięci vs 2kB
  - Flash: 2 MB (układ zewnętrzny) vs 32 kB
  - Cena: 19,90 zł vs 90,00zł (wersja oryginalna)
- Programowanie w MicroPython
- Duża liczba modułów rozszerzeń – alternatywa dla połączeń na płytce stykowej



Repozytorium z programami omawianymi podczas prezentacji:

<https://github.com/anras5/Raspberry-Pi-Pico>



- <https://www.raspberrypi.com/products/raspberry-pi-pico/>
- <https://botland.com.pl/moduly-i-zestawy-do-raspberry-pi-pico/18767-raspberry-pi-pico-rp2040-arm-cortex-m0-0617588405587.html>