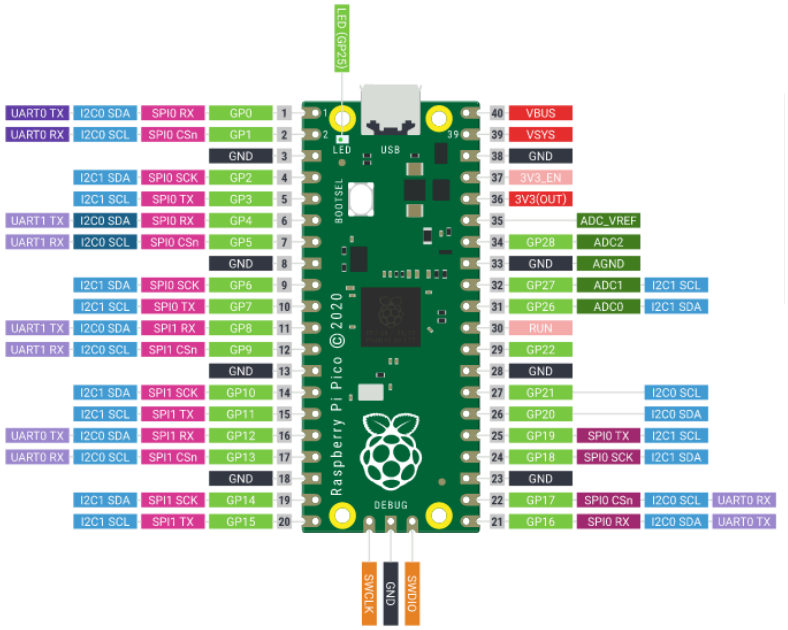
Wprowadzenie. Raspberry Pi Pico W adds on-board single-band 2.4GHz wireless interfaces (802.11n) using the Infineon CYW43439 while retaining the Pico form factor. The onboard 2.4GHz wireless interface has the following features:

- Wireless (802.11n), single-band (2.4 GHz).
- WPA3.
- Soft access point supporting up to four clients.



The antenna is an onboard antenna licensed from ABRACON (formerly ProAnt). The wireless interface is connected via SPI to the RP2040 microcontroller.

Ze względu na ograniczenia w dostępie do wyprowadzeńDue to pin limitations, some of the wireless interface pins are shared. Pin CLK jest współdzielony z VSYS, dlatego komunikacja z wykorzystaniem magistrali SPI transaction in progress can VSYS be read via the ADC. The Infineon CYW43439 DIN/DOUT and IRQ all share one pin on the RP2040. Only when an SPI transaction isn’t in progress is it suitable to check for IRQs. The interface typically runs at 33MHz.

Zadanie 1. Skonfiguruj płytkę Raspberry Pi Pico W. Wykorzystaj informacje dostępne w poniższym linku:

projects.raspberrypi.org/en/projects/get-started-pico-w/

Zadanie 2. Podłącz płytkę Raspberry Pi Pico W do sieci bezprzewodowej Laboratorium-IoT.

```
import time
from network import WLAN, STA_IF
import gc

ssid = "Laboratorium-IoT"
password = "tutaj podaj hasło"

gc.collect()

wlan = WLAN(STA_IF)
wlan.active(True)
wlan.connect(ssid,password)

max_wait = 10
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    print("Connecting to ",ssid)
    time.sleep(1)

if wlan.status() != 3:
    raise RuntimeError("Network connection failed")
else:
    print("Connected")
    status = wlan.ifconfig()
    print("IP adres " + status[0])

wlan.disconnect()
```

Kody określające stan połączenie można znaleźć na 16 stronie w *Connecting to the Internet with Raspberry Pi Pico W*:

www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html

Zadanie 3. Utwórz funkcję `wifi_connect()`.

```
from time import sleep
from network import WLAN,STA_IF
from machine import reset
import gc

gc.collect()
ssid = "Laboratorium-IoT"
password = "tutaj podaj hasło"

def wifi_connect():
    wlan = WLAN(STA_IF)
    wlan.active(True)
    wlan.connect(ssid,password)
    while wlan.isconnected() == False:
        print("Connection to " + ssid)
        sleep(1)
    print(wlan.ifconfig())
    print(f'Connected to {ssid} network, IP {wlan.ifconfig()[0]}')

try:
    wifi_connect()
except KeyboardInterrupt:
    reset()
```

Zadanie 4. Asynchroniczna implementacja serwera. Podłącz LED do GPIO 15.

```
from time import sleep
from network import WLAN,STA_IF
```

```
from machine import reset,Pin
import uasyncio as asyncio
import gc

gc.collect()

def web_page(led_state):
    html = """
    <html>
    <head>
        <meta name="viewport" content="width=device-width,
                    initial-scale=1">
        <link rel="stylesheet"
           href="https://use.fontawesome.com/releases/v5.7.2/css/all.
           css">
        <style>
            html {
                font-family: Arial;
                display: inline-block;
                margin: 0px auto;
                text-align: center;
            }

            .button {
                background-color: #ce1b0e;
                border: none;
                color: white;
                padding: 16px 40px;
                text-align: center;
                text-decoration: none;
                display: inline-block;
                font-size: 16px;
                margin: 4px 2px;
                cursor: pointer;
            }

            .button1 {
                background-color: #000000;
            }
        </style>
    </head>
    <body>
```

```
        <h2>RPi Pico W based web server</h2>
        <p>LED state: <strong>""" + led_state + """</strong></p>
        <p>
            <i class="fas fa-lightbulb fa-3x" style="color:#c81919;">
                </i>
            <a href=\"led_on\"><button class="button">LED ON</button>
                </a>
        </p>
        <p>
            <i class="far fa-lightbulb fa-3x" style="color:#000000;">
                </i>
            <a href=\"led_off\"><button class="button button1">LED
                OFF</button></a>
        </p>
    </body>
    </html>"""
    return html


ssid = "Laboratorium-IoT"
password = "tutaj podaj hasło"

led = Pin(15,Pin.OUT)
led_state = "LED is off"
onboard = Pin("LED",Pin.OUT,value = 0)

def wifi_connect():
    wlan = WLAN(STA_IF)
    wlan.active(True)
    wlan.connect(ssid,password)
    while wlan.isconnected() == False:
        print("Connection to " + ssid)
        sleep(1)
    print(wlan.ifconfig())
    print(f'Connected to {ssid} network, IP {wlan.ifconfig()[0]}')

async def handle_client(reader,writer):
    print("Client connected")
    request_line = await reader.readline()
    print("Request:",request_line)

    #skip header
```

```
    while await reader.readline() != b"\r\n":
        pass

    request = str(request_line)
    led_on = request.find('led_on')
    led_off = request.find('led_off')
    print('led on = ' + str(led_on))
    print('led off = ' + str(led_off))

    if led_on == 7:
        print("led on")
        led.value(1)
        led_state = "LED is ON"

    if led_off == 7:
        print("led off")
        led.value(0)
        led_state = "LED is OFF"

    response = web_page(led_state)
    writer.write('HTTP/1.0 200 OK\r\nContent-type:
text/html\r\n\r\n')
    writer.write(response)

    await writer.drain()
    await writer.wait_closed()
    print("Client disconnected")

async def main():
    try:
        wifi_connect()
    except KeyboardInterrupt:
        reset()

    print("Starting webserver")
    asyncio.create_task(asyncio.start_server(handle_client,
"0.0.0.0", 80))
    while True:
        onboard.on()
        #print("server is still alive")
        await asyncio.sleep(0.25)
        onboard.off()
```

```
        await asyncio.sleep(5)

try:
    asyncio.run(main())
finally:
    asyncio.new_event_loop()
```

Zadanie 5. Sprawdź działanie poniższego kodu.

```
from time import sleep
from network import WLAN,STA_IF
from machine import reset
import urequests as rq
import json

ssid = "Laboratorium-IoT" #"iot_test"
password = "tutaj podaj hasło" #"RPi_pico"

def wifi_connect():
    wlan = WLAN(STA_IF)
    wlan.active(True)
    wlan.connect(ssid,password)
    while wlan.isconnected() == False:
        print("Connection to " + ssid)
        sleep(1)
    print(wlan.ifconfig())
    print(f'Connected to {ssid} network, IP {wlan.ifconfig()[0]}')

try:
    wifi_connect()
except KeyboardInterrupt:
    reset()
```

```
print("\n\n2. Querying the current GMT+0 time:")
date_time = rq.get("http://time.jsontest.com")
print(date_time.json())
today = 'Today is ' + date_time.json().get('date')
print(today)
print('What time is right now? ' + date_time.json().get('time'))
```

Zadanie 6. Podłącz moduły Pico-LCD-0.96 oraz BMP280.

Napisz skrypt wyświetlający informację odnośnie temperatury otoczenia uzyskaną z BMP-280 oraz z serwisu *OpenWeather* (openweathermap.org).

Zadanie 7. Napisz skrypt pobierający w czasie rzeczywistym przynajmniej 3 parametry opisujące pogodę z usługi *OpenWeatherMap*. Wyświetl pobrane informacje na stronie www. Dodaj ikonę/obrazek obrazujące dany parametr.

Zadanie 8. Podłącz moduły Pico-LCD-0.96 oraz BMP280.

Napisz skrypt prezentujący na wyświetlaczu informacje odnośnie temperatury otoczenia, ciśnienia atmosferycznego, oraz bieżącą datę i godzinę (usługi NTP, timejson, itp).

Dla zainteresowanych:

1. Strona MicroPython:
   micropython.org/download/rp2-pico/
2. Moduł Raspberry Pi Pico W:
   projects.raspberrypi.org/en/projects/get-started-pico-w/
3. Raspberry Pi Pico i Pico W:
   www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html
4. Waveshare Pico LCD 0.96:
   www.waveshare.com/wiki/Pico-LCD-0.96