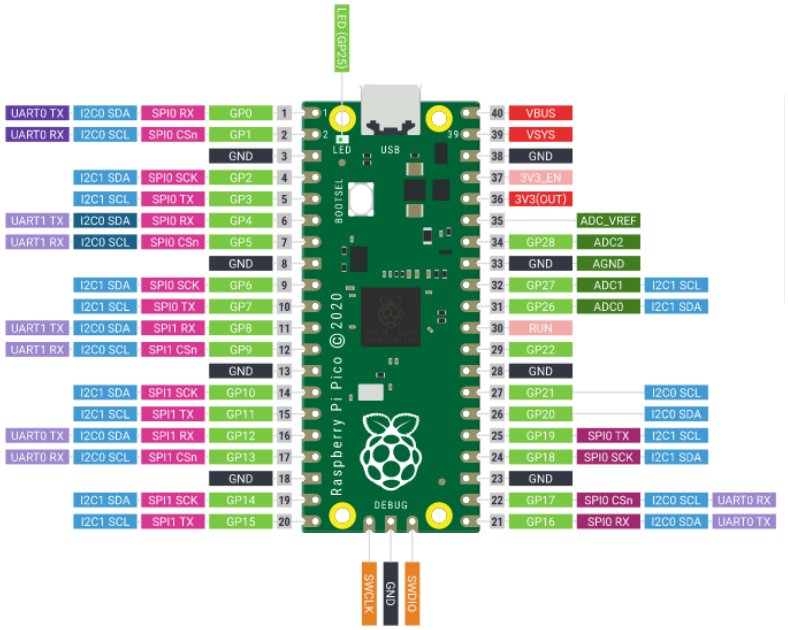
Introduction. Raspberry Pi Pico W adds on-board single-band 2.4GHz wireless interfaces (802.11n) using the Infineon CYW43439 while retaining the Pico form factor. The onboard 2.4GHz wireless interface has the following features:

- Wireless (802.11n), single-band (2.4 GHz).
- WPA3.
- A 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.

Due to pin limitations, some of the wireless interface pins are shared. The CLK is shared with the VSYS monitor, so only when there isn't an 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.

Task 1. Set up Your Raspberry Pi Pico W board. Read the manual available at the following link:

projects.raspberrypi.org/en/projects/get-started-pico-w/1

All the codes are available in the GitHub repository - `ex05_wifi.zip`.

Task 2. Connect a Raspberry Pi Pico W board to the `Laboratorium-IoT` wireless network.

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

ssid = "Laboratorium-IoT"
password = "enter the password here"

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()
```

Connection status codes are available on page 16 of *Connecting to the Internet with Raspberry Pi Pico W*:

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

Task 3. Create a `wifi_connect()` function.

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

gc.collect()
ssid = "Laboratorium-IoT"
password = "enter the password here"

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()
```

Task 4. Asynchronous web server implementation. Connect an LED to 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 = "enter the password here"

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()
```

Task.5. Implement the following code.

```
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 = "enter the password here" #"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'))
```

Task.6. (1 point to the final score) Connect Pico-LCD-0.96 and BMP280 modules.

Write a script to display the temperature obtained from the sensor and the *OpenWeather* service (openweathermap.org).

Task.7. (1.5 points to the final score) Extract, in real-time, at least 3 weather parameters from the *OpenWeatherMap* webpage. Present the parameters on the webpage. Add an icon/drawing for every weather parameter. At least one of the parameters should be presented on the gauge and chart.

Task.8. (1 point to the final score) Connect Pico-LCD-0.96 and BMP280 modules.

Write a script to display the temperature obtained from the sensor and the current time. Present the current time and the temperature on the web page.

For those interested:

1. Raspberry Pi Pico W vs. Pico. What's the difference?
   core-electronics.com.au/guides/raspberry-pi-pico-w-vs-pico-whats-the-difference/

2. MicroPython web page:
   micropython.org/download/rp2-pico/

3. Set up your Raspberry Pi Pico W:
   projects.raspberrypi.org/en/projects/get-started-pico-w/

4. Raspberry Pi Pico and Pico W:
   www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html

5. Waveshare Pico LCD 0.96
   www.waveshare.com/wiki/Pico-LCD-0.96