Getting the hardware up and running

Install toolchain and environment

We will use the official guide "Getting started with Raspberry Pi Pico-series", which can be found here: getting-started-with-pico.pdf

Install VS Code editor and plugins

Follow the instructions in "Getting started" chapter 2 and 3.

Blink an LED

Follow the instructions in chapter 4 until (but not including) chapter 4.3.

Hello world on serial console

Follow the instructions in chapter 5. This will get you to the point, where "Hello world" is printed on the USB serial device and shown in the serial monitor in VS Code.

Configure debugging from VS Code

Download the "debugprobe_on_pico2.uf2" file from https://github.com/raspberrypi/debugprobe/releases/latest and follow the instructions from https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html#software-utilities to install it on your second RPi Pico 2W.

Connect the RPi with the debugprobe SW to the target RPi.

Connect the pins between the debugger and the target as described in section "Debug with a second Pico or Pico 2" in getting-started-with-pico.pdf . Note, that on the target Pico 2W, the SWDIO and SWCLK connnections are located in the middle of the board, next to the blank shield.

Set optimization level

The compiler sometimes opmizes the code in a way, where code is removed or restructured. This can make the debugging session a bit wierd. To disable this, open the CMakeLists.txt file in the project and add the following section, which will disable optimization (e.g. set the compiler flag -O0 and prevent inlining of code):

```
target_compile_options(blink PRIVATE -O0 -g -fno-inline -fno-inline-functions)
```

after the `target_link_libraries(blink pico_stdlib)` section.

Install a terminal program to talk to serial console

You can use the built-in Serial Monitor in VS Code. Select the "USB Serial Device" in the Port dropdown and press "Start Monitoring".

Add serial output to the blink demo

Make the blink.c project write to the serial port. Add a counter variable and turn on the LED when the counter value is an equal number and off when it is an odd number. Remenber to #include <stdio.h> and initialize stdio.

```
int main() {
    uint16_t cnt = 0;

    int rc = pico_led_init();
    stdio_init_all();

    printf("init_return_value: %d\r\n", rc);
    hard_assert(rc == PICO_OK);
    while (cnt < 50) {
        if (cnt%2) {
            pico_set_led(true);
            printf("On  - cnt: %d\r\n", cnt);
        }
        else
        {
            pico_set_led(false);
            printf("Off - cnt: %d\r\n", cnt);
        }
        sleep_ms(LED_DELAY_MS);
        cnt++;
    }
}
```

Start the program in a debug session and experiment with breakpoints and stepping through the code.