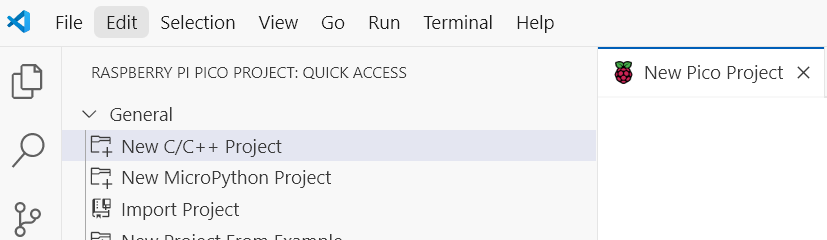
# Visual Studio Code: pierwszy projekt PiPico

Rozszerzenie Pico w Visual Studio Code ułatwia zakładanie pierwszego projektu:



Aby ułatwić sobie życie zwróć uwagę na te opcje:

|  | 1. nazwa projektu 2. wybór modułu MCU 3. opcjonalnie, np. drugi UART do obsługi RS485 4. printf będzie przekierowane na UART0 5. inicjalizacja CYW43 (WiFi), co jest niezbędne, aby używać on-board LED. Pico1 bez ‘W’ miało LED połączony z GPIO, więc niektóre przykłady nie zadziałają! 6. kod programu ładowany do RAM a nie do Flash - trochę przyspiesza programowanie, ale po zaniku zasilania program zniknie. Możesz wybrać gdy projekty są niewielkie oraz nie potrzebujesz całej pamięci RAM. 7. Wygenerowany zostanie podstawowy projekt C++ gotowy do dalszego rozwijania. 8. Utwórz projekt wg wybranych ustawień. |
| --- | --- |

Taki powinien być efekt działania kreatora projektu C++:

#include <stdio.h>

#include "pico/stdlib.h"

#include "pico/cyw43\_arch.h"

#include "hardware/uart.h"

// UART defines

// By default the stdout UART is `uart0`, so we will use the second one

#define UART\_ID uart1

#define BAUD\_RATE 115200

// Use pins 4 and 5 for UART1

// Pins can be changed, see the GPIO function select table in the datasheet for information on GPIO assignments

#define UART\_TX\_PIN 4

#define UART\_RX\_PIN 5

int main()

{

stdio\_init\_all();

// Initialise the Wi-Fi chip

if (cyw43\_arch\_init()) {

printf("Wi-Fi init failed\n");

return -1;

}

// Example to turn on the Pico W LED

cyw43\_arch\_gpio\_put(CYW43\_WL\_GPIO\_LED\_PIN, 1);

// Set up our UART

uart\_init(UART\_ID, BAUD\_RATE);

// Set the TX and RX pins by using the function select on the GPIO

// Set datasheet for more information on function select

gpio\_set\_function(UART\_TX\_PIN, GPIO\_FUNC\_UART);

gpio\_set\_function(UART\_RX\_PIN, GPIO\_FUNC\_UART);

// Use some the various UART functions to send out data

// In a default system, printf will also output via the default UART

// Send out a string, with CR/LF conversions

uart\_puts(UART\_ID, " Hello, UART!\n");

// For more examples of UART use see https://github.com/raspberrypi/pico-examples/tree/master/uart

while (true) {

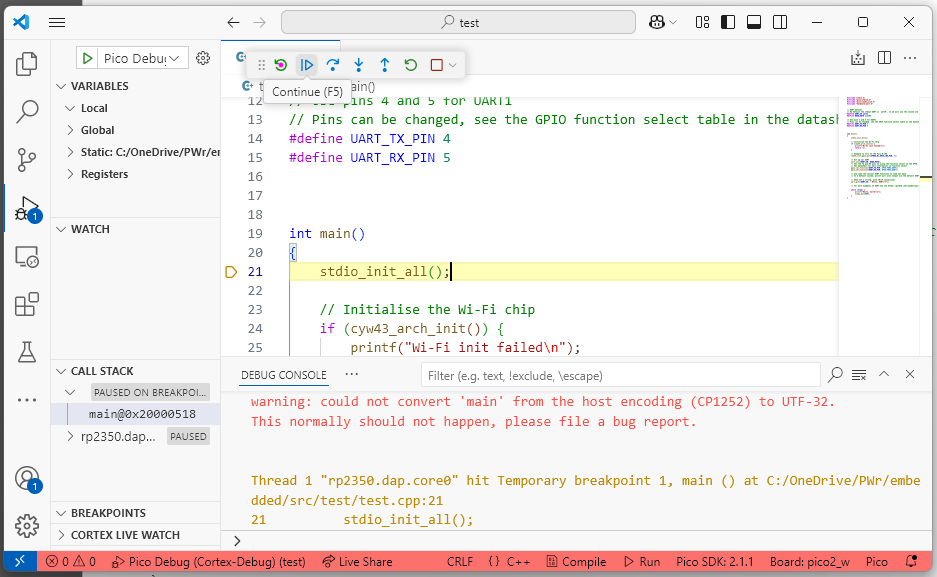
printf("Hello, world!\n");

sleep\_ms(1000);

}

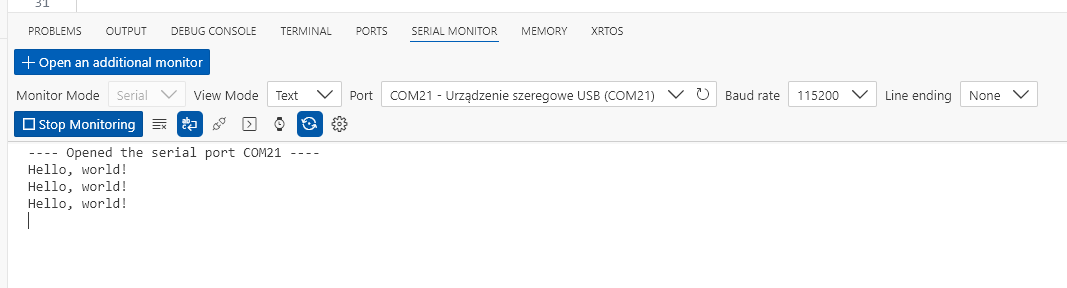
}

Budowa projektu i zaprogramowanie MCU: F5



Kolejne F5 uruchomi program.

Wyniki podejrzysz w zakładce SERIAL MONITOR pod oknem z kodem źródłowym. Po wybraniu portu szeregowego (w Linux będzie to np. /dev/ttyACM0) kliknij [Start monitoring]:



Gotowe!