CEC Continuing Education Center

DesignNews

Raspberry Pi Pico/Pico2 Development Using Visual Studio Code

Day 3:

Raspberry Pi Pico 2 C Programming with VS Code

Sponsored by

DigiKey

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

Fred Eady

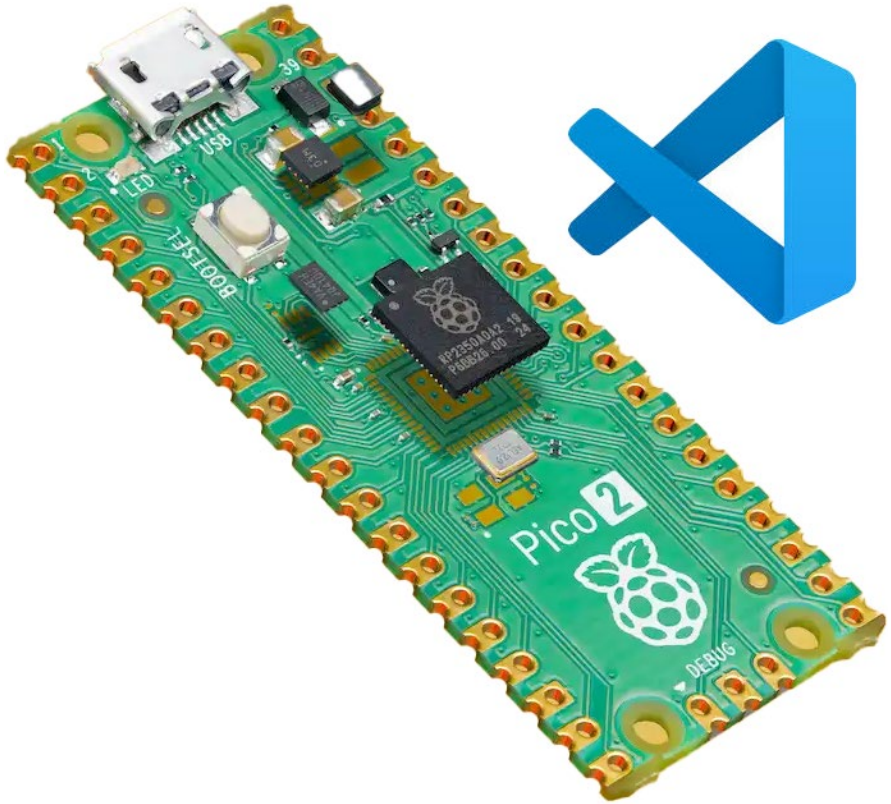
Visit ‘Lecturer Profile’ in your console for more details.

CEC Continuing Education Center

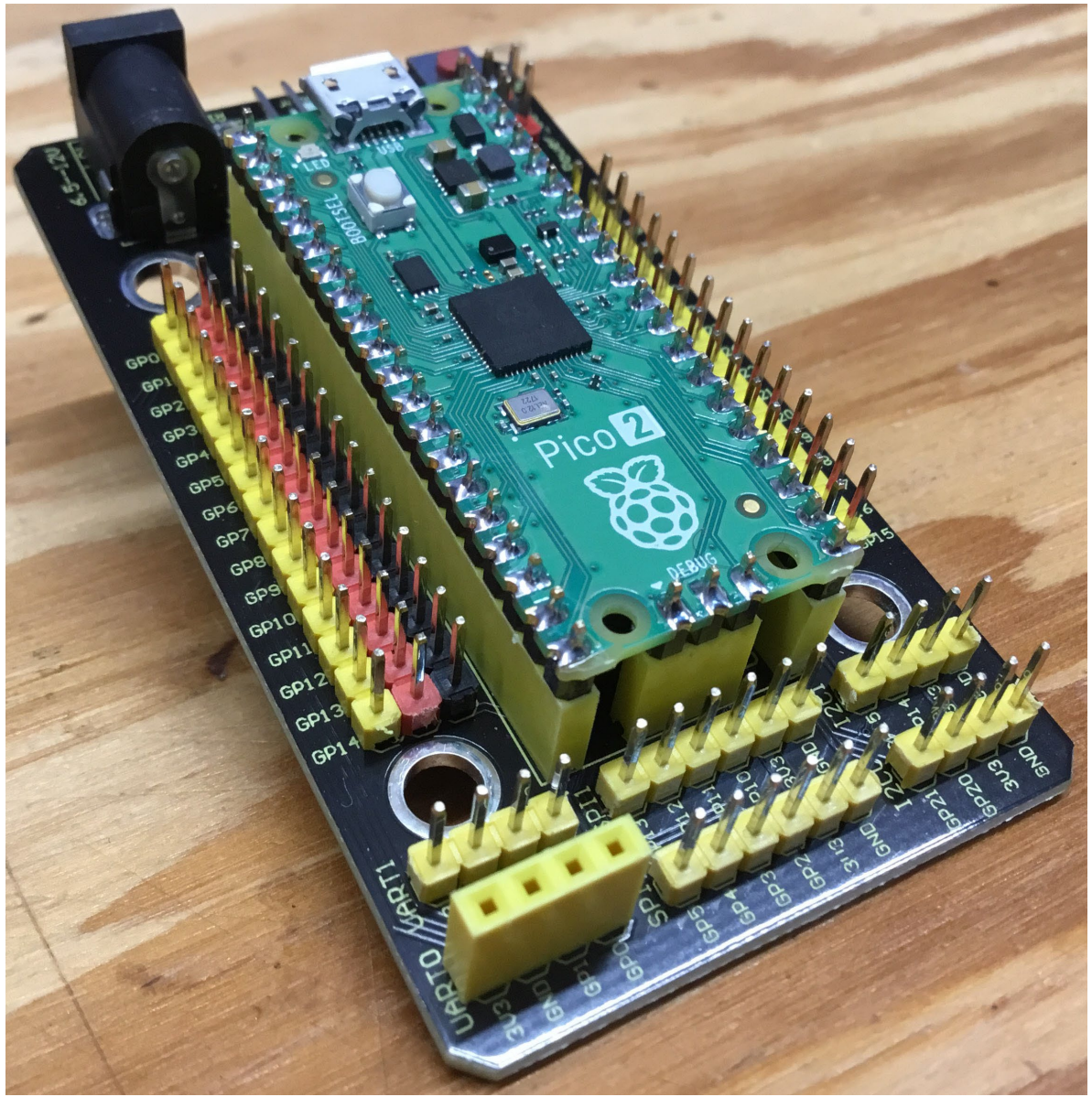
AGENDA

DigiKey

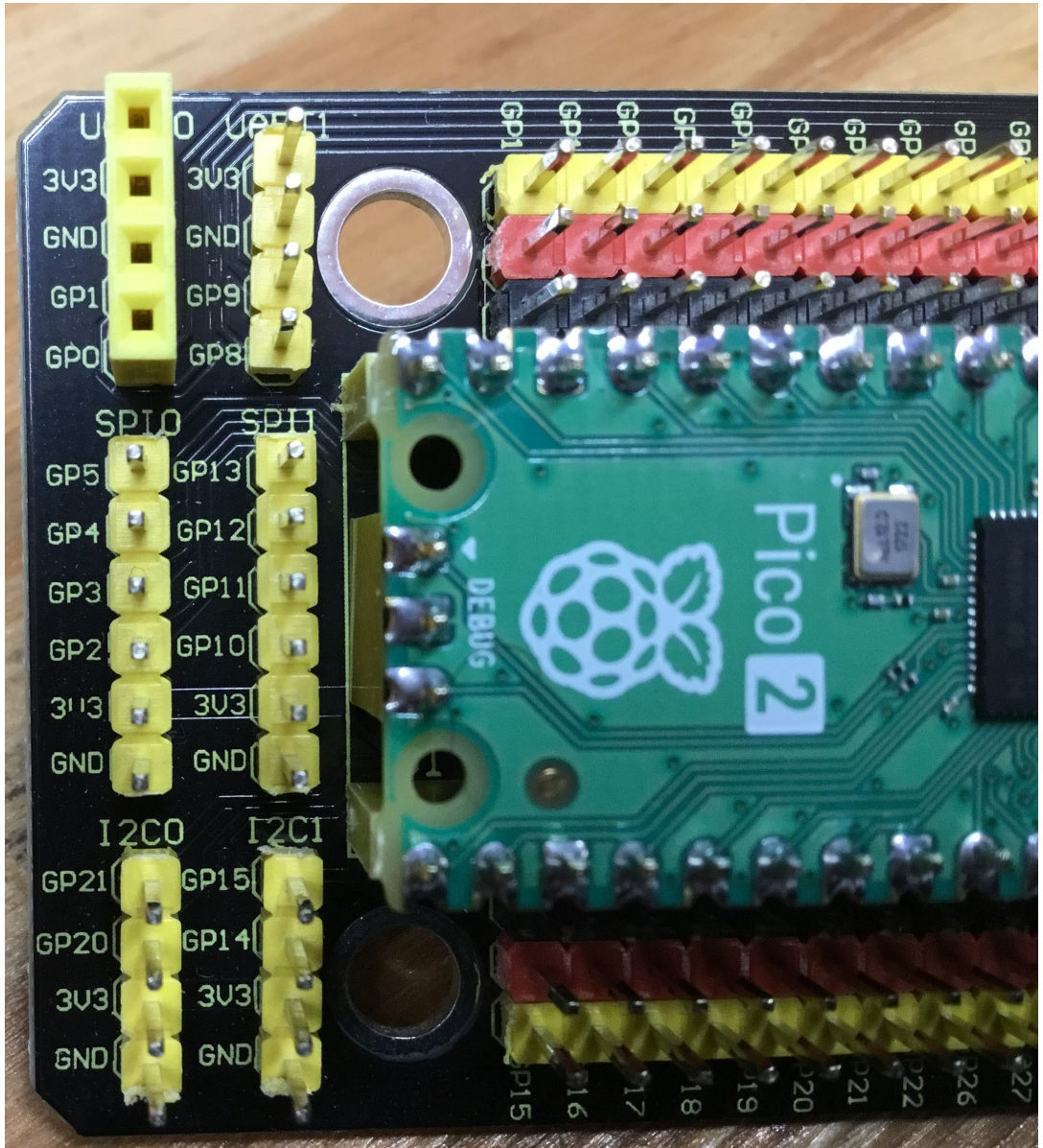
- Pico 2 Development Hardware
- Raspberry Pi Debug Probe
  - The Hook Up
- Code a Pico 2 Ring Buffer Application



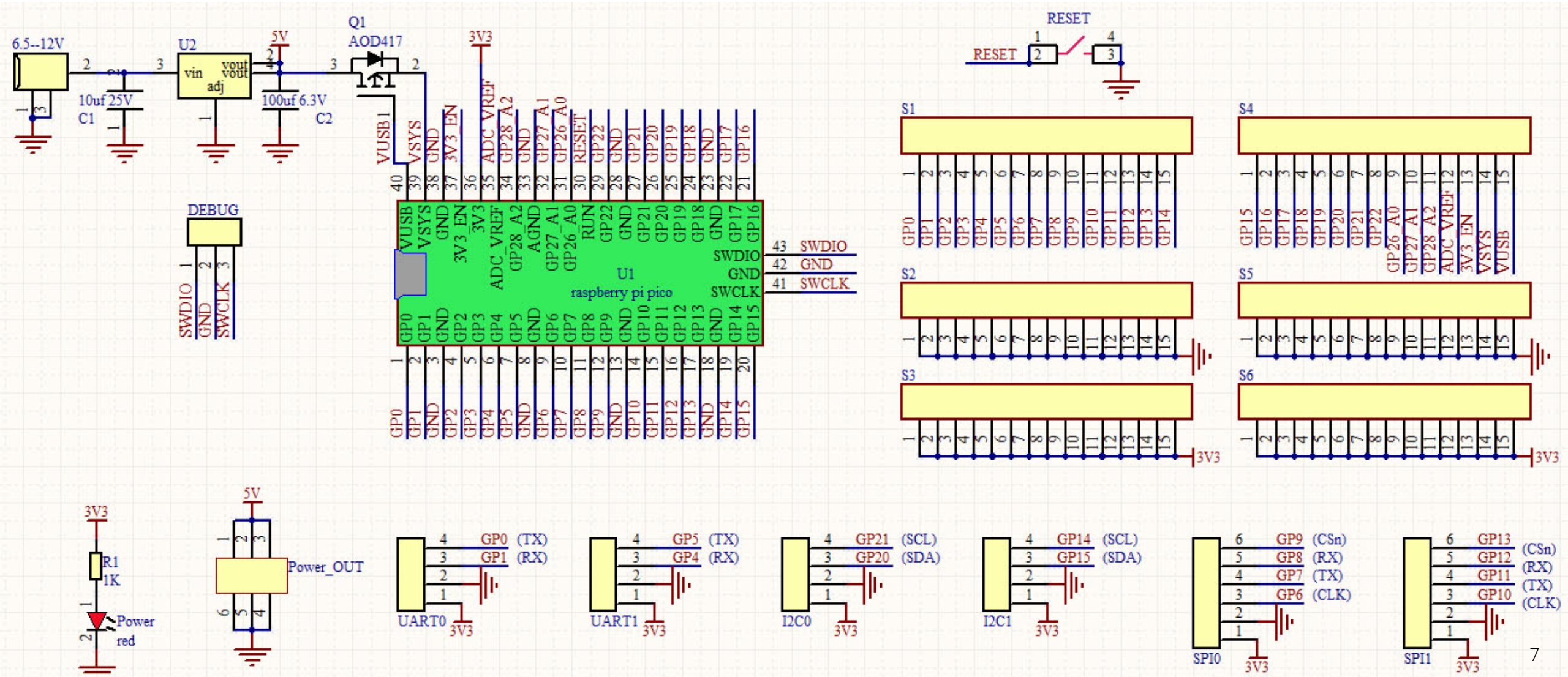
Pico 2 Carrier Board



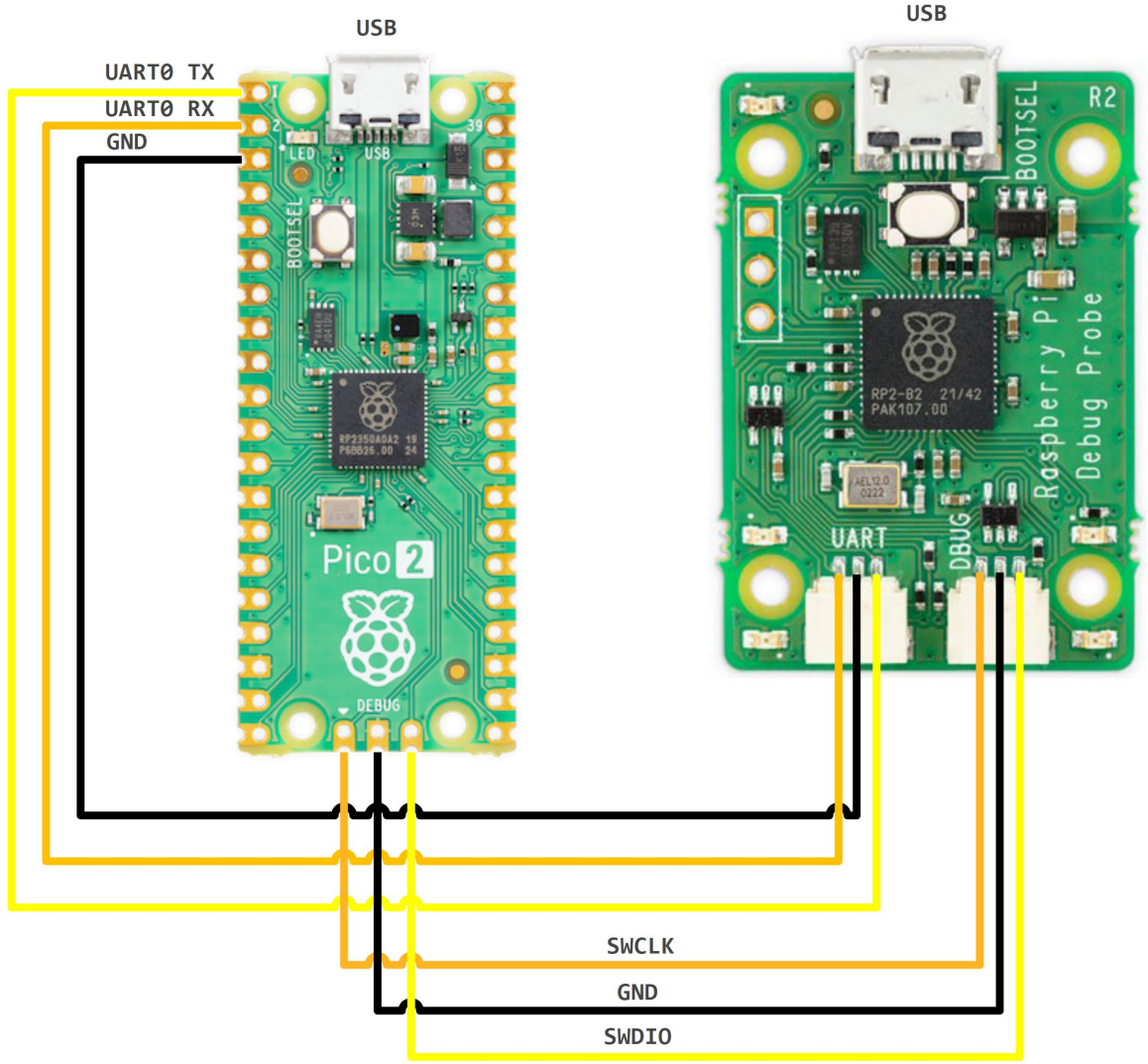
Pico 2 Carrier Board

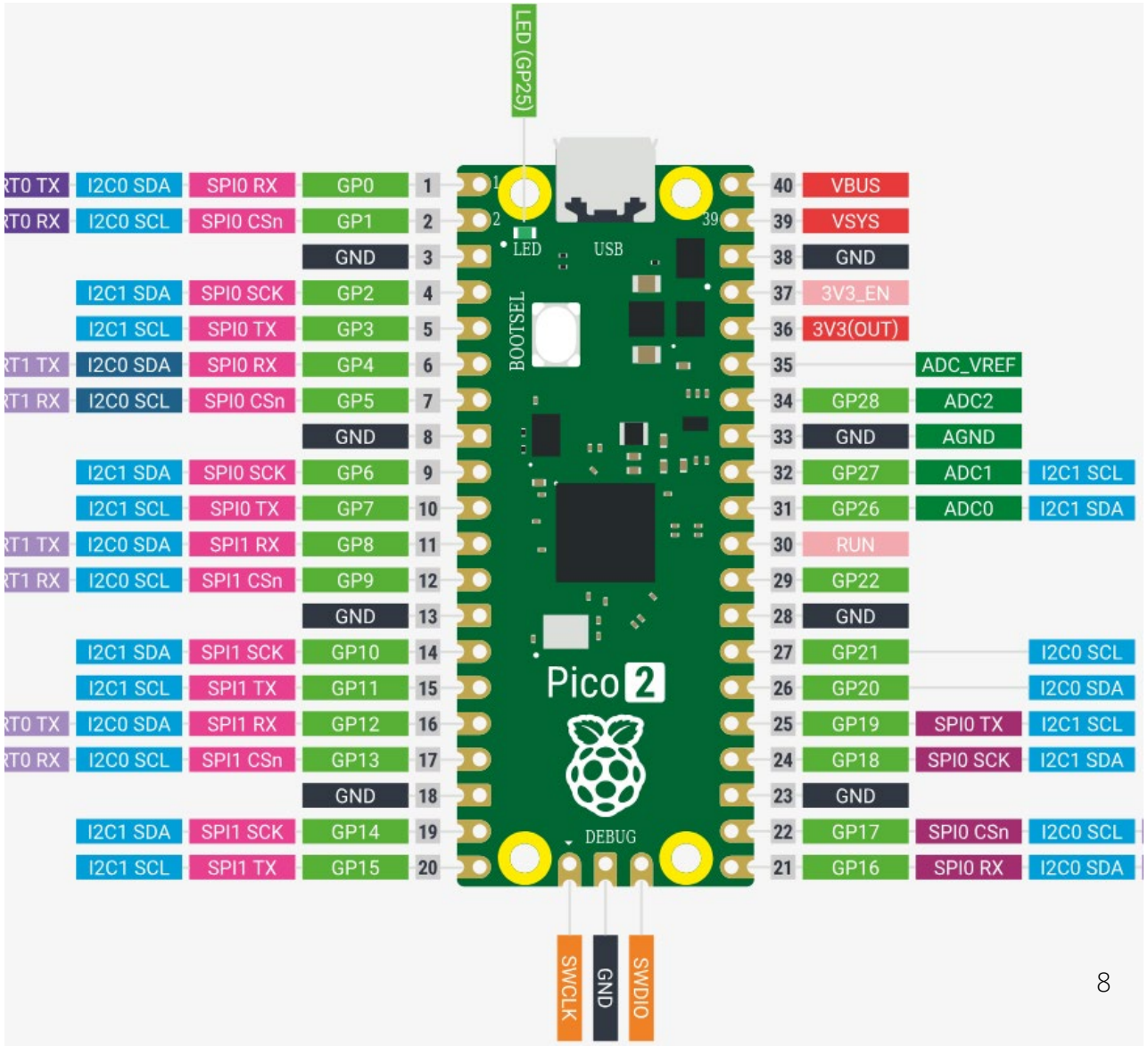


Pico 2 Carrier Board



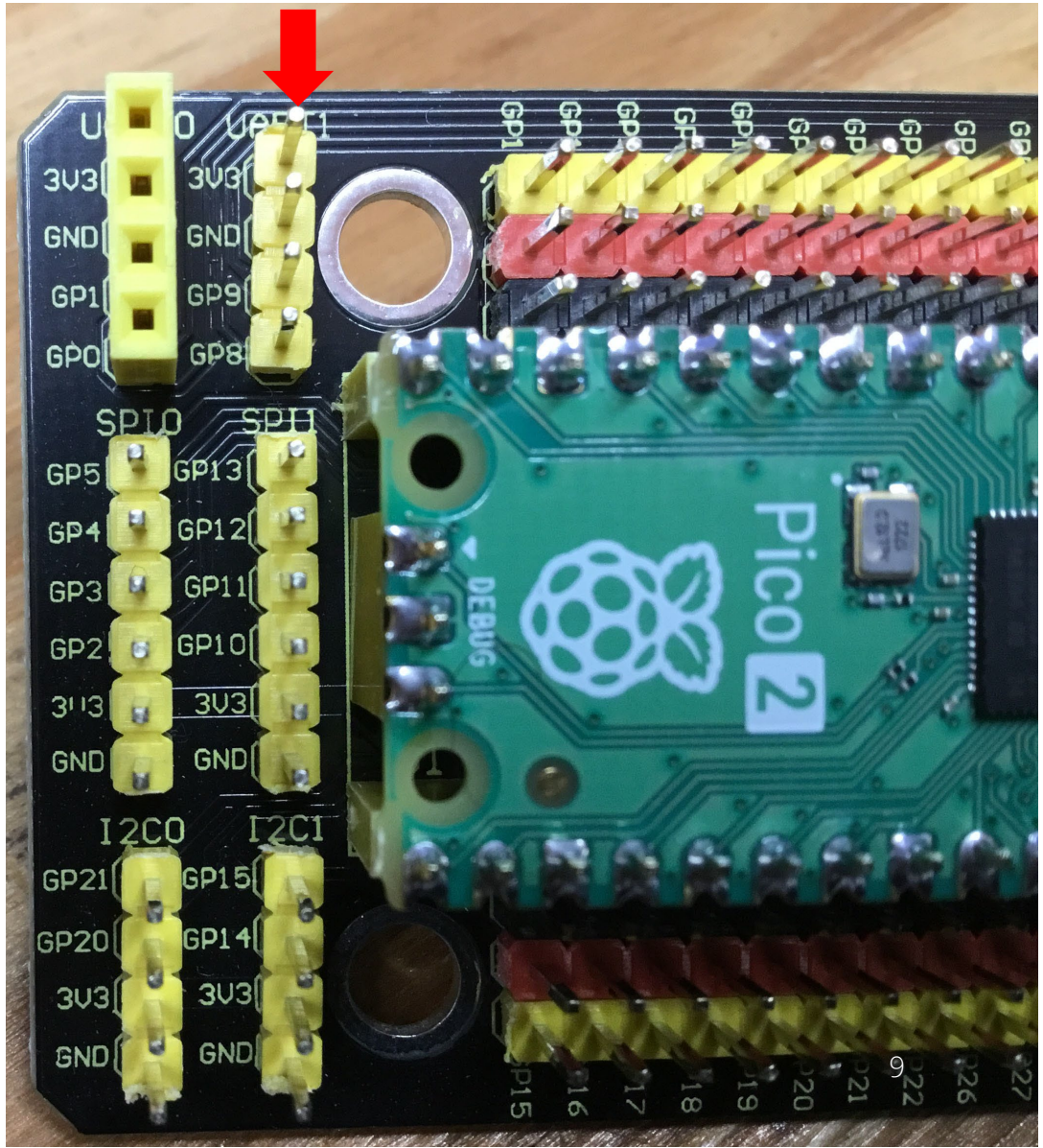
The Hook Up


USB
USB
UART0 TX
UART0 RX
GND
SWCLK
GND
SWDIO
Pico 2
Raspberry Pi Debug Probe
R2
BOOTSEL
LED
USB
DEBUG
UART
DBUG

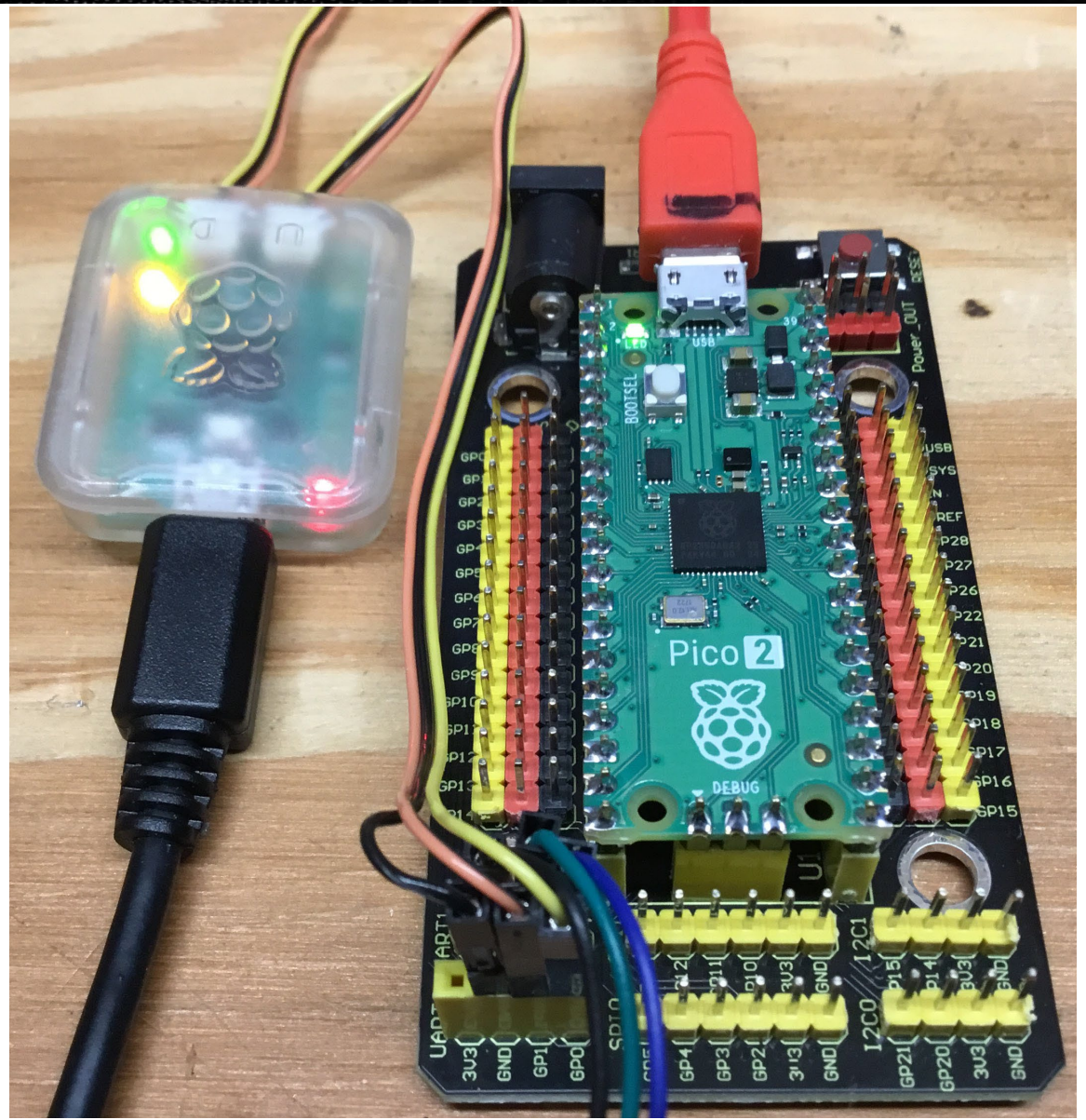

LED (GP25)
GP0 1 I2C0 SDA SPI0 RX
GP1 2 I2C0 SCL SPI0 CSn
GND 3
GP2 4 I2C1 SDA SPI0 SCK
GP3 5 I2C1 SCL SPI0 TX
GP4 6 I2C0 SDA SPI0 RX
GP5 7 I2C0 SCL SPI0 CSn
GND 8
GP6 9 I2C1 SDA SPI0 SCK
GP7 10 I2C1 SCL SPI0 TX
GP8 11 I2C0 SDA SPI1 RX
GP9 12 I2C0 SCL SPI1 CSn
GND 13
GP10 14 I2C1 SDA SPI1 SCK
GP11 15 I2C1 SCL SPI1 TX
GP12 16 I2C0 SDA SPI1 RX
GP13 17 I2C0 SCL SPI1 CSn
GND 18
GP14 19 I2C1 SDA SPI1 SCK
GP15 20 I2C1 SCL SPI1 TX
40 VBUS
39 VSYS
38 GND
37 3V3_EN
36 3V3(OUT)
35 ADC_VREF
34 GP28 ADC2
33 GND AGND
32 GP27 ADC1 I2C1 SCL
31 GP26 ADC0 I2C1 SDA
30 RUN
29 GP22
28 GND
27 GP21 I2C0 SCL
26 GP20 I2C0 SDA
25 GP19 SPI0 TX I2C1 SCL
24 GP18 SPI0 SCK I2C1 SDA
23 GND
22 GP17 SPI0 CSn I2C0 SCL
21 GP16 SPI0 RX I2C0 SDA
LED
USB
BOOTSEL
Pico 2
DEBUG
SWCLK
GND
SWDIO

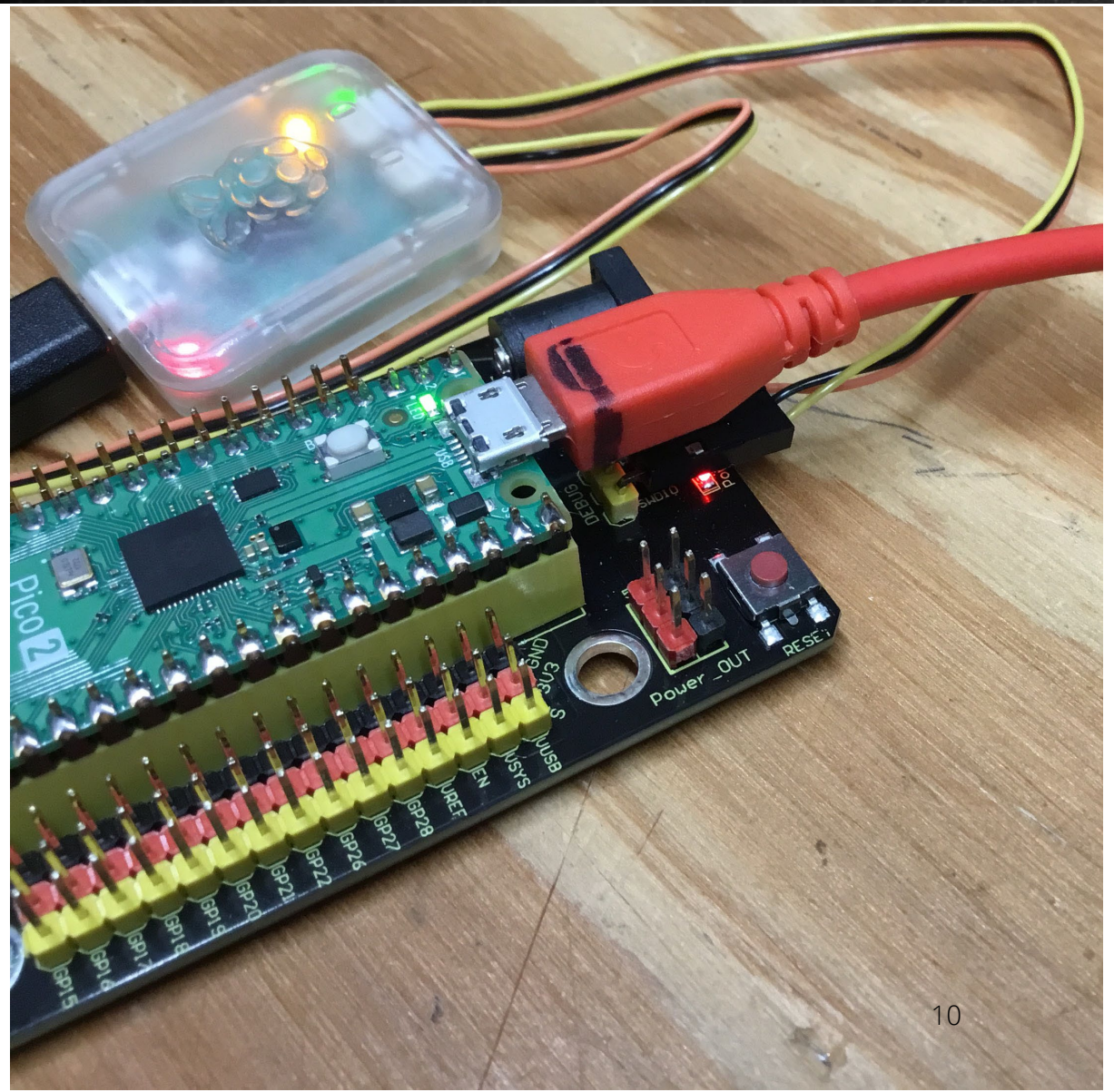
The Hook Up


USB TO TTL (D)
PWR
TXD
RXD
USB
5V
3V3
VCC
GND
TXD
GND
RXD

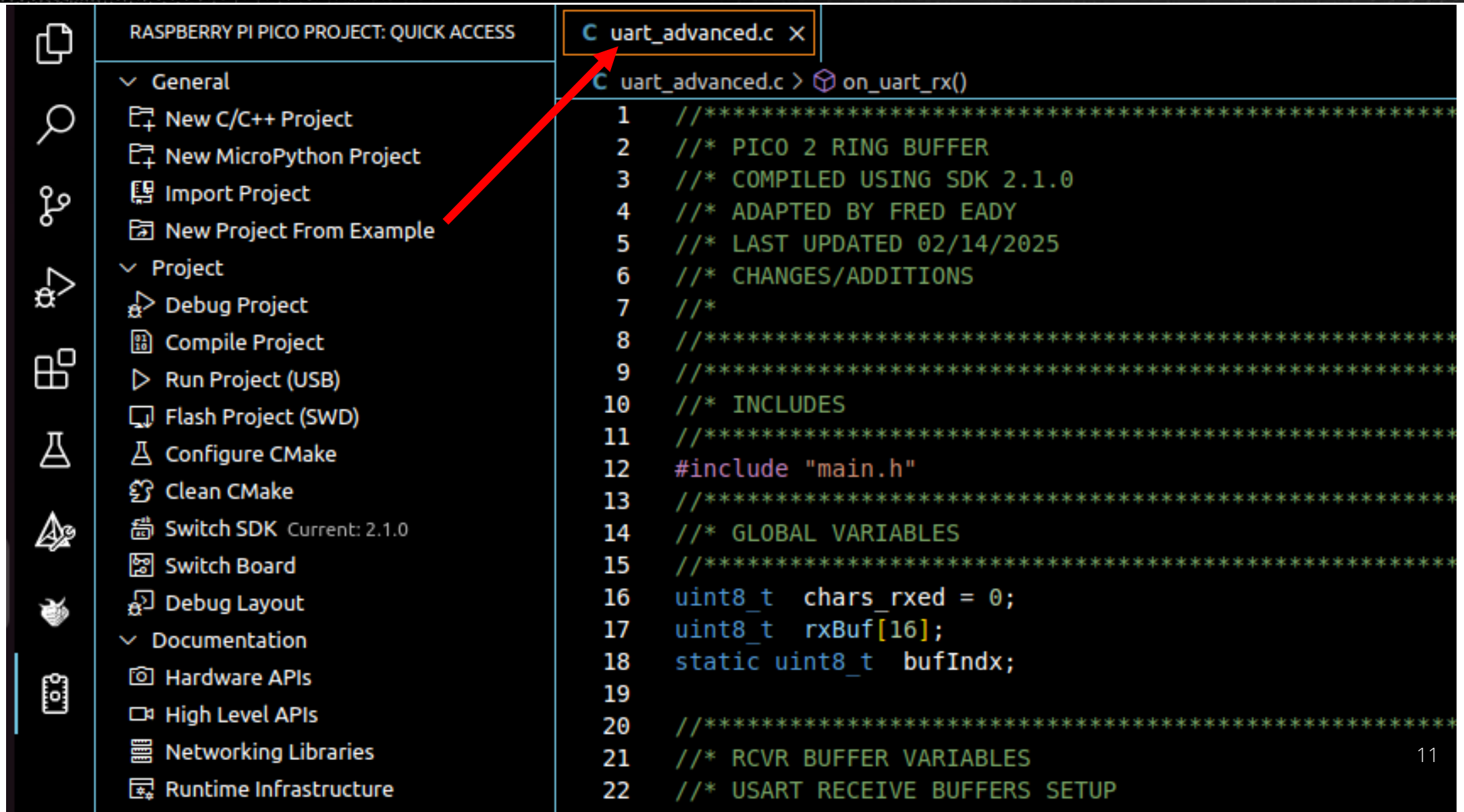

3V3
GND
GP1
GP0
GP9
GP8
SPI0
SPI1
GP5
GP4
GP3
GP2
GP13
GP12
GP11
GP10
3V3
GND
I2C0
I2C1
GP21
GP20
GP15
GP14
Pico 2
DEBUG

The Hook Up



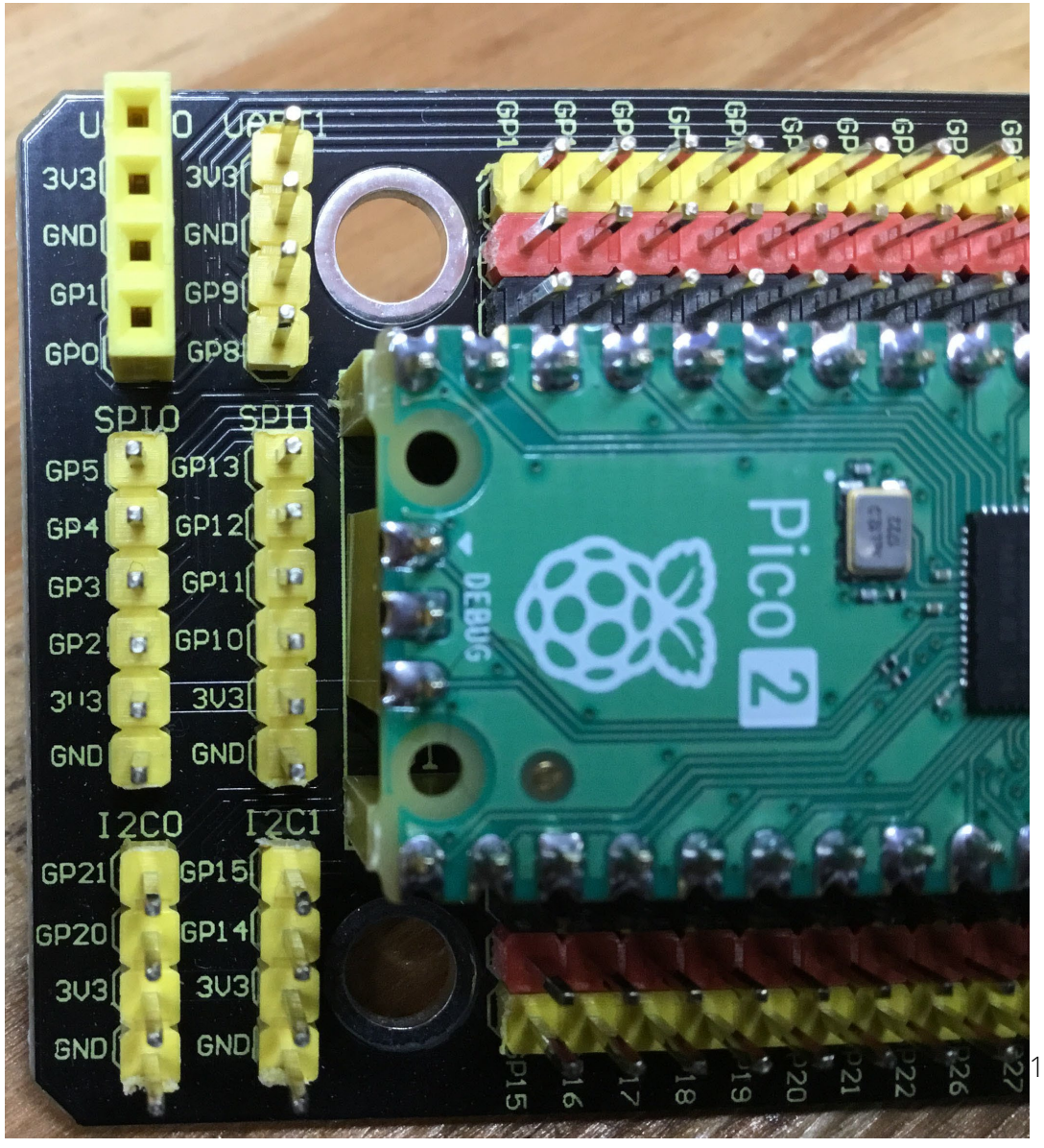


Modify the *uart_advanced.c* Example



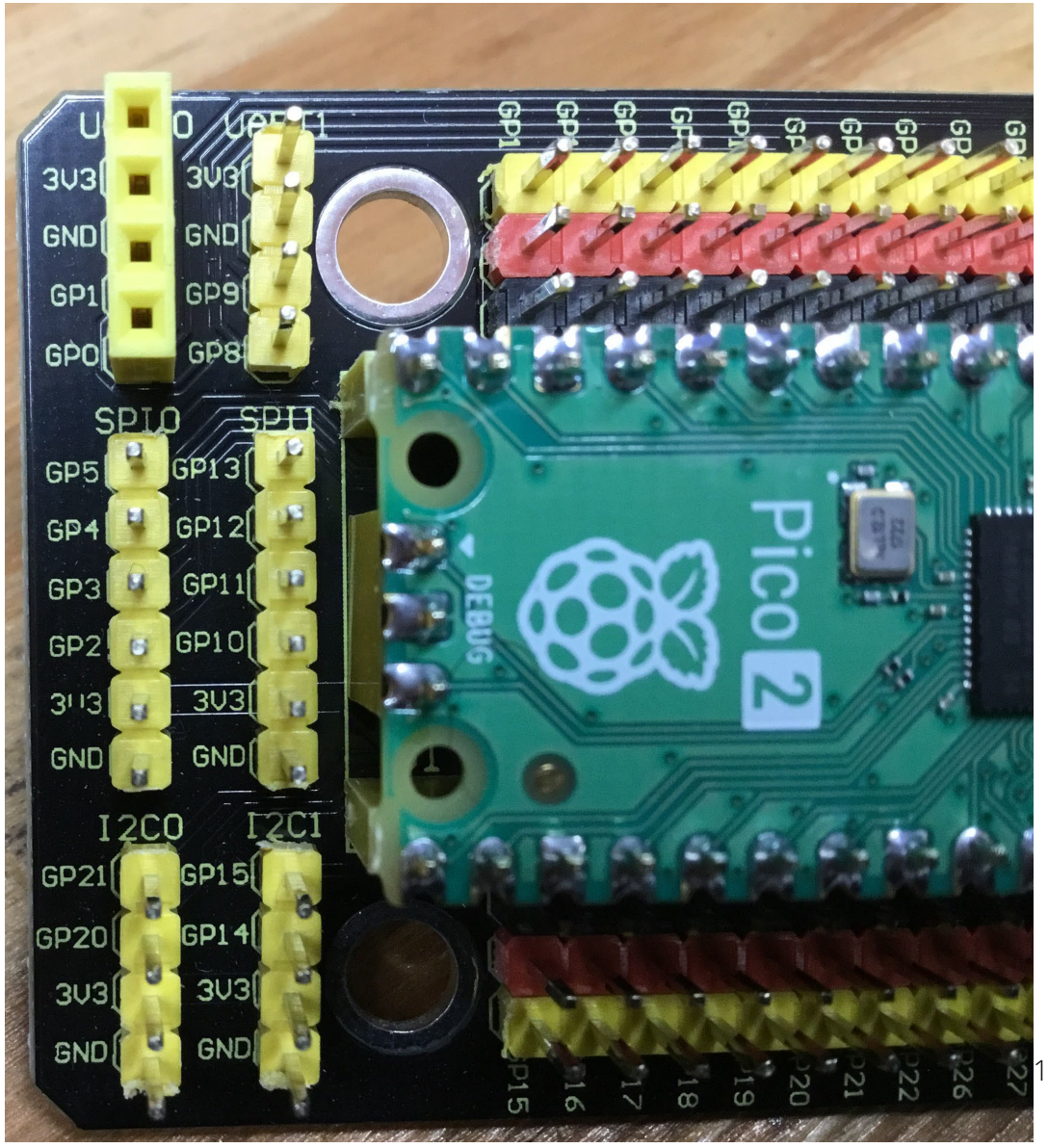
Modify the *uart_advanced.c* Example

```
//*****************************************************
//* PICO 2 RING BUFFER
//* COMPILED USING SDK 2.1.0
//* WRITTEN BY FRED EADY
//* LAST UPDATED 02/14/2025
//* CHANGES/ADDITIONS
//*
//*****************************************************
//*****************************************************
//* INCLUDES
//*****************************************************
#include "pico/stdlib.h"
#include "hardware/uart.h"
#include "hardware/irq.h"
//*****************************************************
//* FUNCTION PROTOTYPES
//*****************************************************
uint8_t readring(void);
void on_uart_rx(void);
//*****************************************************
//* UART DEFINITIONS
//*****************************************************
#define UART_ID uart1
#define BAUD_RATE 115200
#define DATA_BITS 8
#define STOP_BITS 1
#define PARITY    UART_PARITY_NONE

#define UART_TX_PIN 8
#define UART_RX_PIN 9
#define UART_RX_BUFFER_SIZE 8
#define UART_RX_BUFFER_MASK ( UART_RX_BUFFER_SIZE - 1 )
```



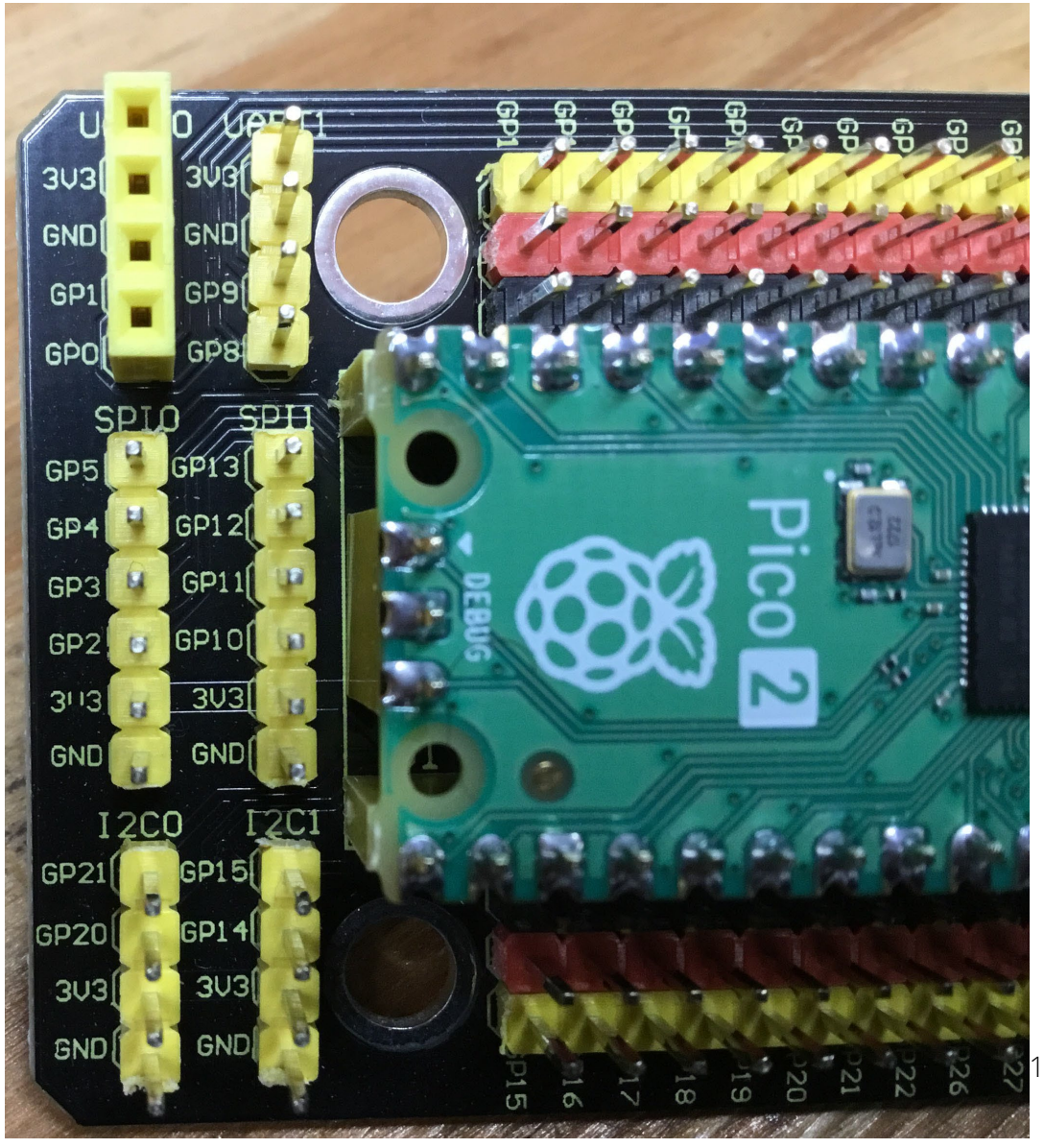
Modify the *uart_advanced.c* Example

```
//*****************************************************************
//* RCVR BUFFER VARIABLES
//* USART RECEIVE BUFFERS SETUP
//* 1,2,4,8,16,32,64,128 or 256 BYTES
//*****************************************************************
uint8_t    UART_RxHead;
uint8_t    UART_RxTail;
uint8_t    UART_RxBuf[UART_RX_BUFFER_SIZE];
uint8_t    data;
uint8_t    tmphead;
uint8_t    tmptail;

void on_uart_rx()
{
    while (uart_is_readable(UART_ID))
    {
        data = uart_getc(UART_ID);
        // calculate buffer index
        tmphead = ( UART_RxHead + 1 ) & UART_RX_BUFFER_MASK;
        // store new index
        UART_RxHead = tmphead;
        // store received data in ring buffer
        UART_RxBuf[tmphead] = data;//(uint8_t)(data & 0x000000FF);
        chars_rxed++;
        //uart_putc(UART_ID, UART_RxBuf[tmphead]);
    }
}
```



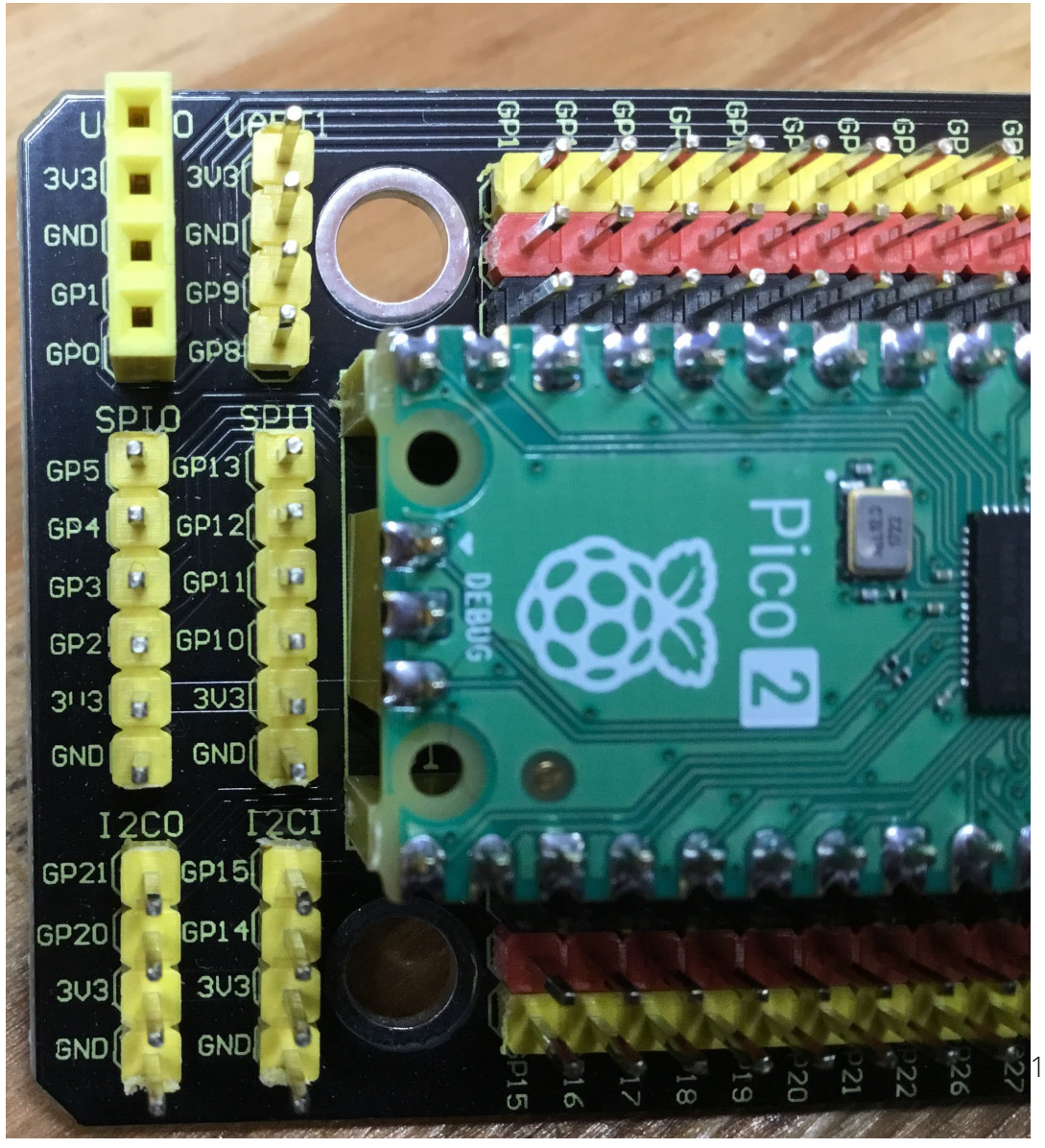
Modify the *uart_advanced.c* Example

```
//*****************************************************
//*    GET BYTE FROM RX RING BUFFER
//*****************************************************
uint8_t readring(void)
{
  // calculate buffer index
  tmptail = ( UART_RxTail + 1 ) & UART_RX_BUFFER_MASK;
  // store new index
  UART_RxTail = tmptail;
  return UART_RxBuf[tmptail];
}
```



Modify the *uart_advanced.c* Example

```c
int main() {
    // Set up our UART with a basic baud rate.
    uart_init(UART_ID, 2400);

    // 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, UART_FUNCSEL_NUM(UART_ID, UART_TX_PIN));
    gpio_set_function(UART_RX_PIN, UART_FUNCSEL_NUM(UART_ID, UART_RX_PIN));

    // Actually, we want a different speed
    // The call will return the actual baud rate selected, which will be as close
    // possible to that requested
    int __unused actual = uart_set_baudrate(UART_ID, BAUD_RATE);

    // Set UART flow control CTS/RTS, we don't want these, so turn them off
    uart_set_hw_flow(UART_ID, false, false);

    // Set our data format
    uart_set_format(UART_ID, DATA_BITS, STOP_BITS, PARITY);

    // Turn off FIFO's - we want to do this character by character
    uart_set_fifo_enabled(UART_ID, false);

    // Set up a RX interrupt
    // We need to set up the handler first
    // Select correct interrupt for the UART we are using
    int UART_IRQ = UART_ID == uart0 ? UART0_IRQ : UART1_IRQ;

    // And set up and enable the interrupt handlers
    irq_set_exclusive_handler(UART_IRQ, on_uart_rx);
    irq_set_enabled(UART_IRQ, true);

    // Now enable the UART to send interrupts - RX only
    uart_set_irq_enables(UART_ID, true, false);

    // OK, all set up.
    uart_puts(UART_ID, "\nREADY\n");
```



Enable *liveWatch*

EXPLORER

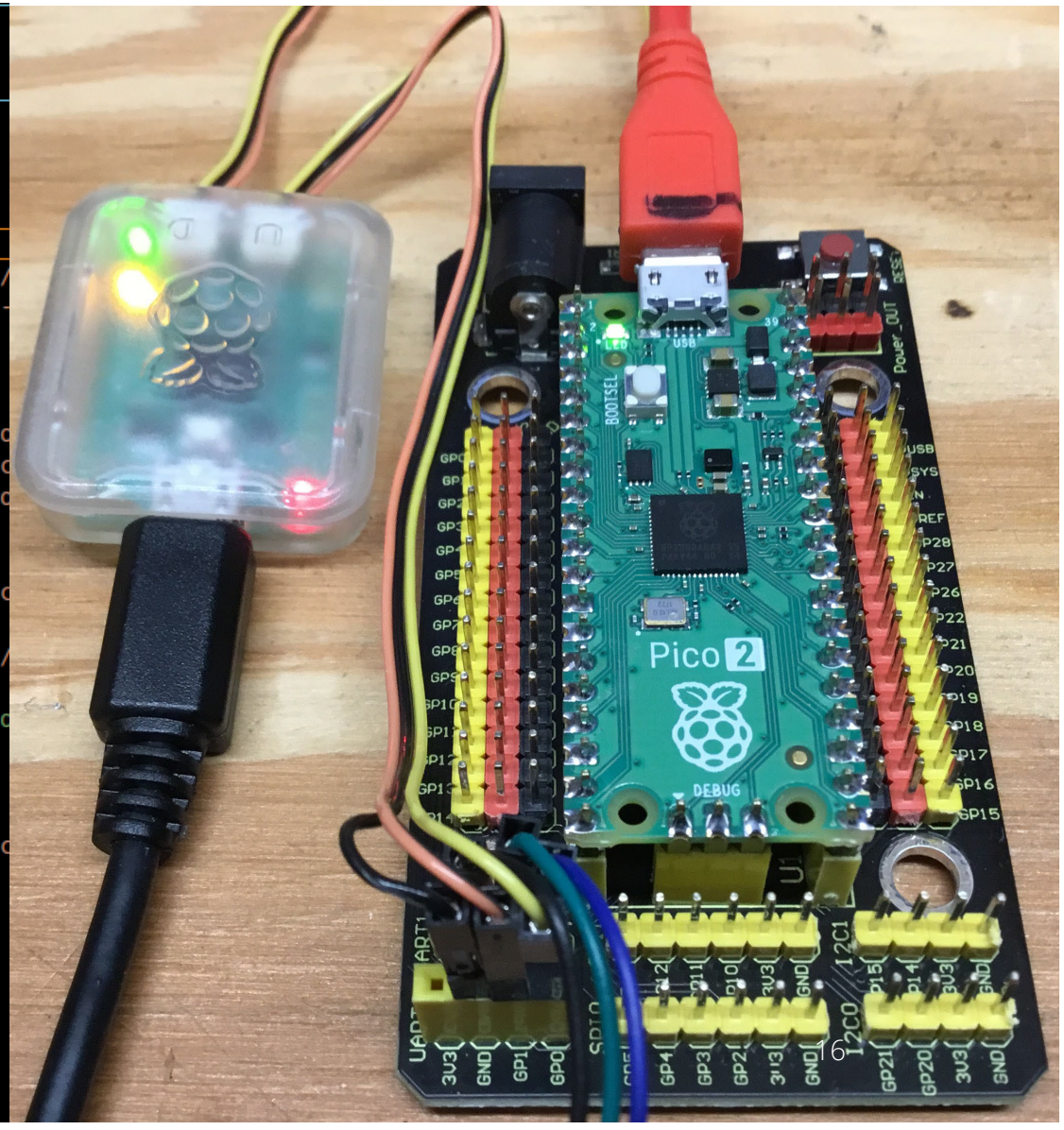
UART_ADVANCED

- .vscode
  - c_cpp_properties.json
  - cmake-kits.json
  - extensions.json
  - launch.json 1
  - settings.json
  - tasks.json
- build
- .gitignore
- CMakeLists.txt
- main.h
- pico_sdk_import.cmake
- uart_advanced.c

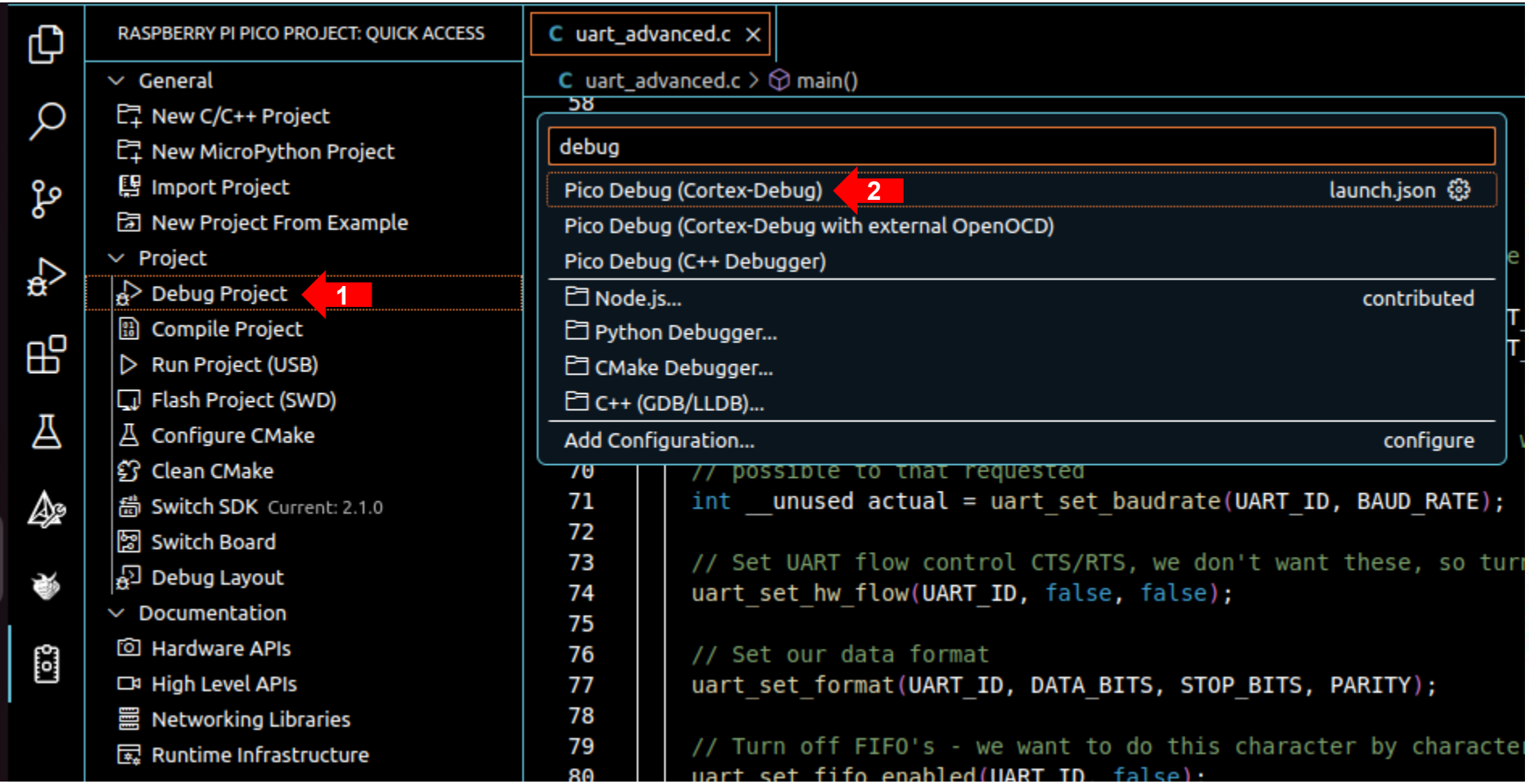
uart_advanced.c | launch.json 1

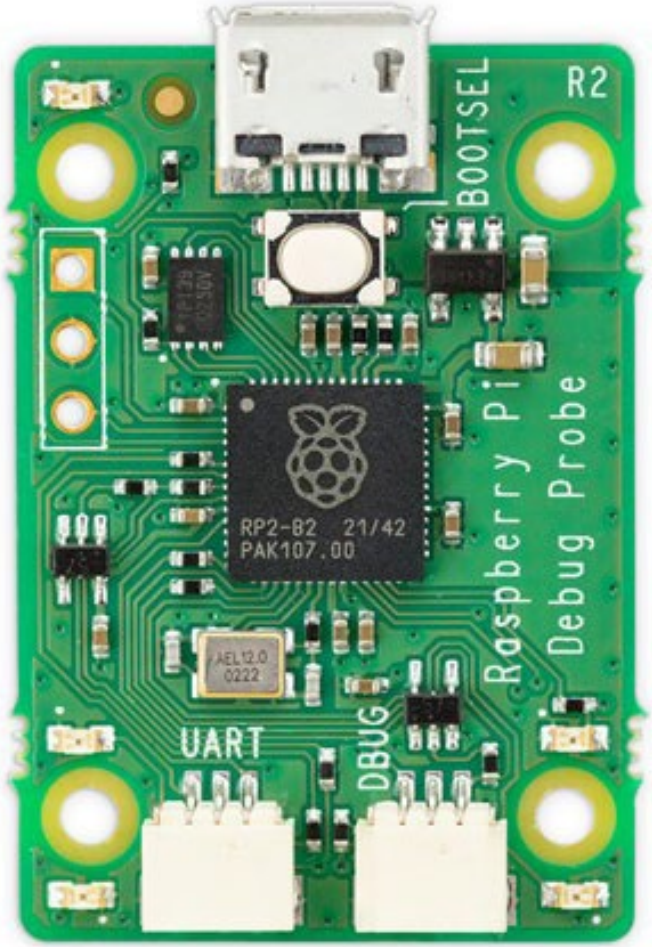
.vscode > {} launch.json > Launch Targets > {} Pico Debug (Cortex-Debug)

```
{
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Pico Debug (Cortex-Debug)",
            "cwd": "${userHome}/.pico-sdk/openocd/
            "executable": "${command:raspberry-pi-
            "request": "launch",
            "type": "cortex-debug",
            "servertype": "openocd",
            "serverpath": "${userHome}/.pico-sdk/o
            "gdbPath": "${command:raspberry-pi-pic
            "device": "${command:raspberry-pi-pico
            "configFiles": [
                "interface/cmsis-dap.cfg",
                "target/${command:raspberry-pi-pic
            ],
            "svdFile": "${userHome}/.pico-sdk/sdk/
            "runToEntryPoint": "main",
            // Fix for no_flash binaries, where mo
            // Also works fine for flash binaries
            "overrideLaunchCommands": [
                "monitor reset init",
                "load \"${command:raspberry-pi-pic
            ],
            "openOCDLaunchCommands": [
                "adapter speed 5000"
            ]
            "liveWatch": {
                "enabled": true,
                "samplesPerSecond": 4
            }
```

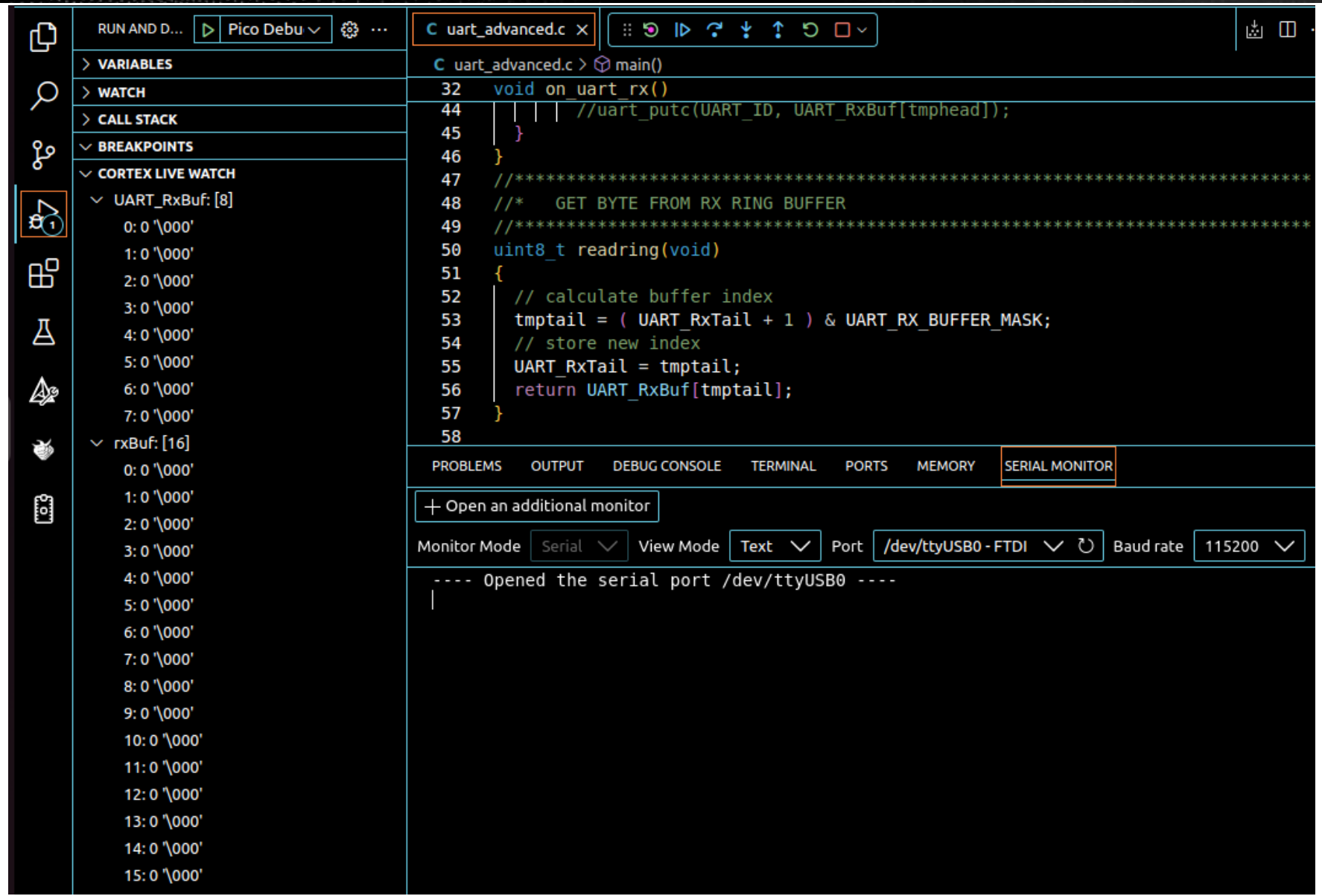


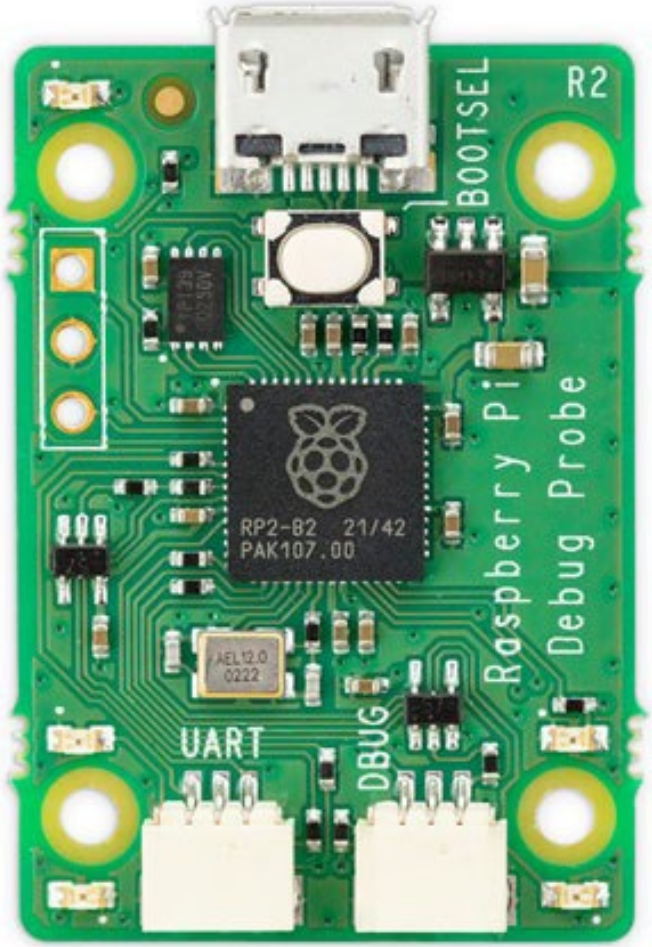
Debug the Pico 2 Ring Buffer Application



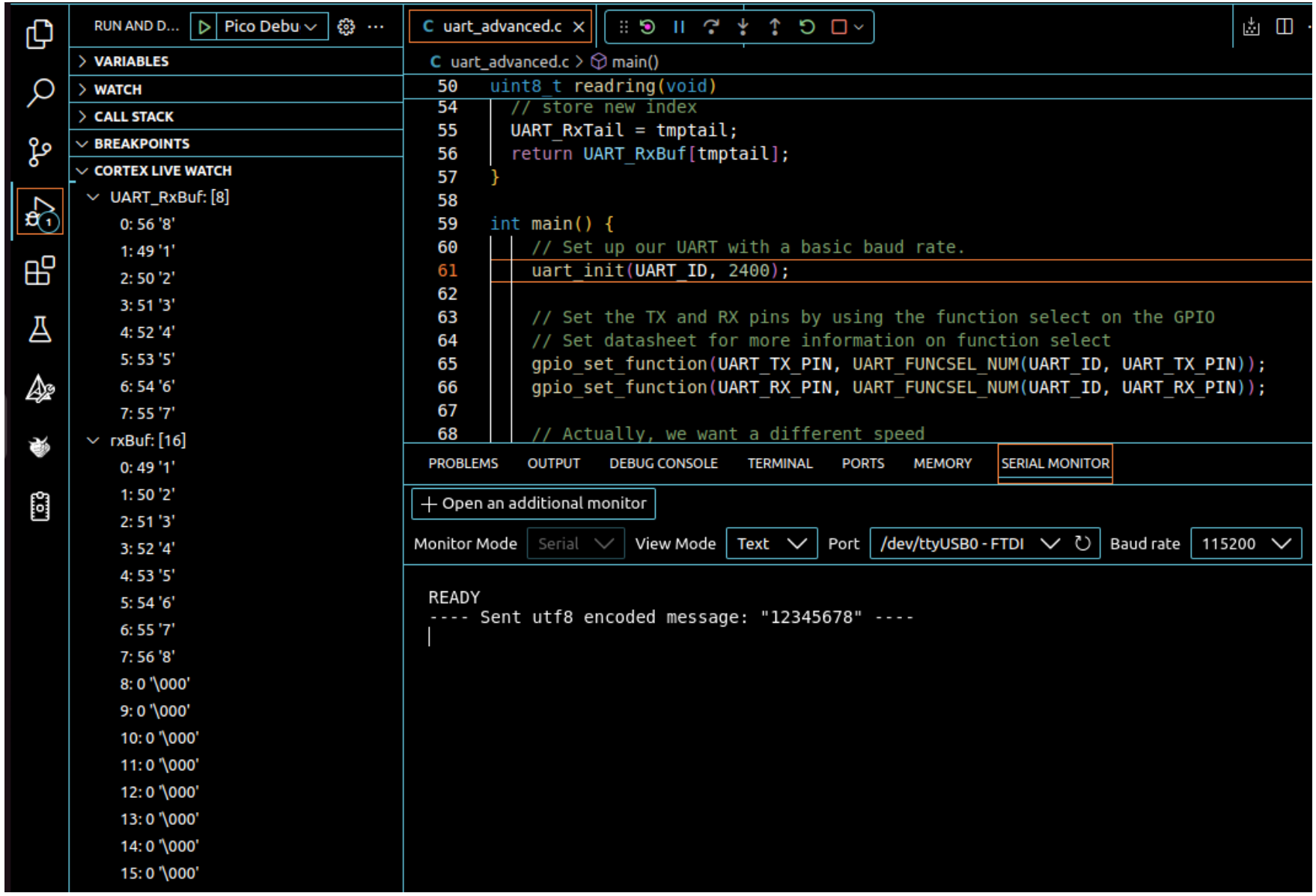


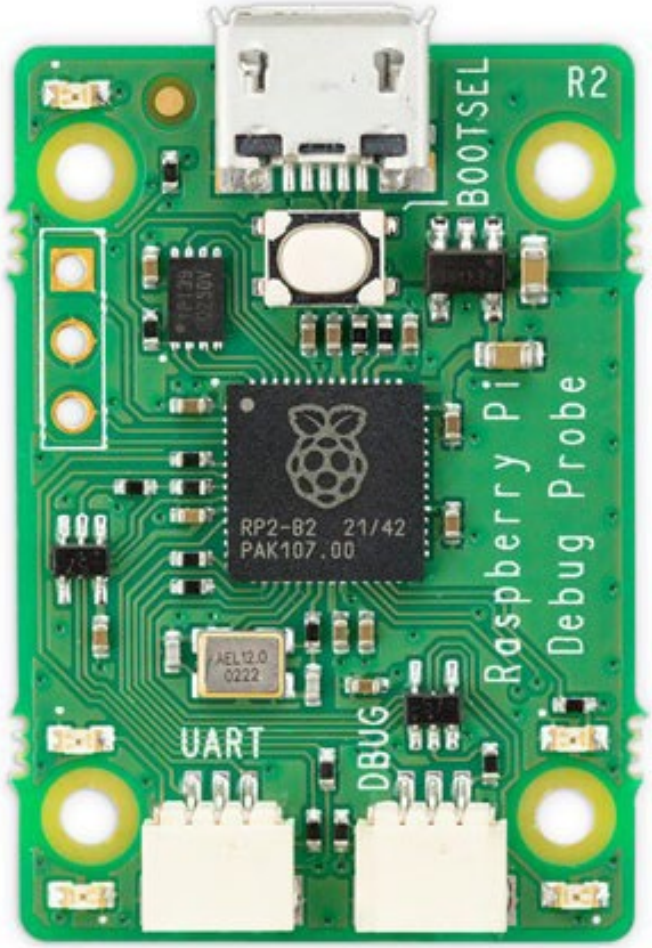
Debug the Pico 2 Ring Buffer Application





Debug the Pico 2 Ring Buffer Application





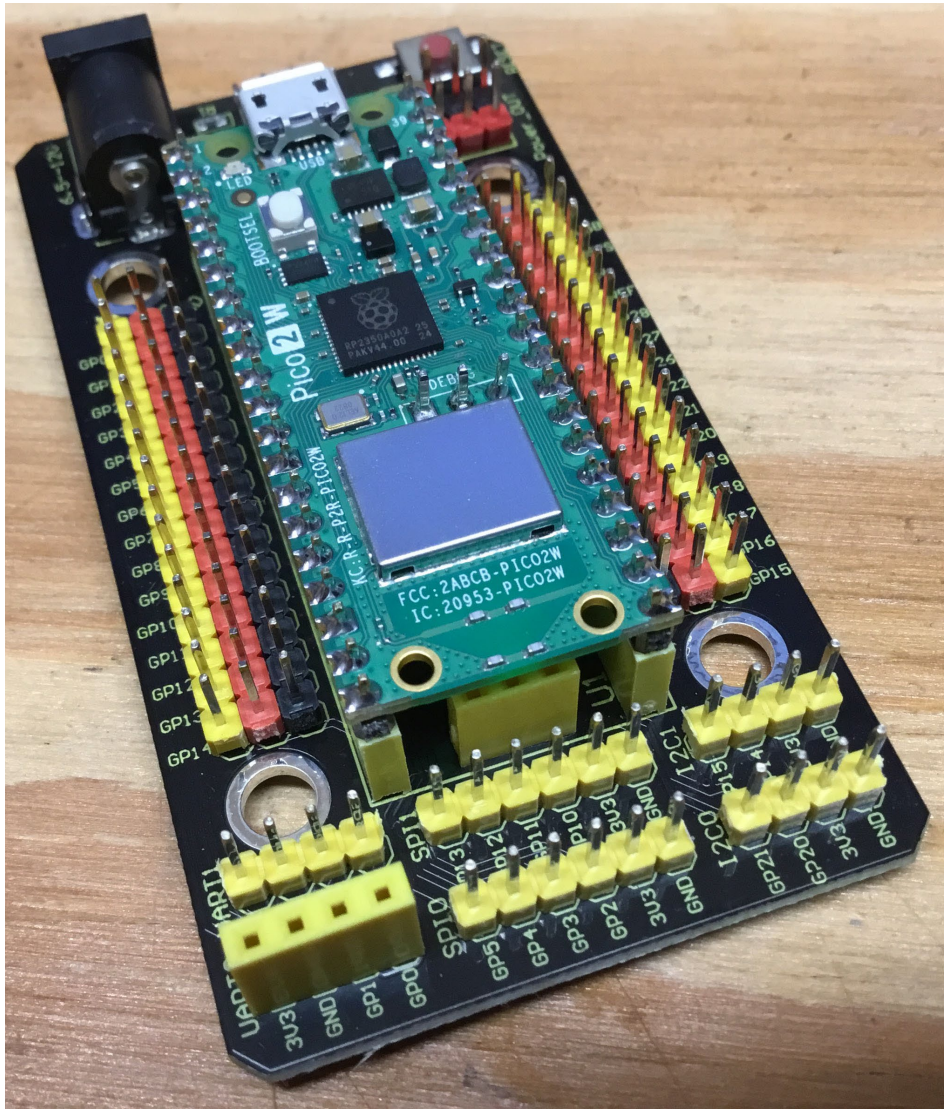
Next Time…

Thank you for attending!!!

Please consider the resources below:

- **Today's Download Package**
- **raspberrypi.org**

MORE TO COME..



CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

informa markets