elektor

Webinars

Embedded Development

APRIL 14 2022

16:00 CEST

elektor design > share > earn

Development with Visual Studio Code and PlatformIO

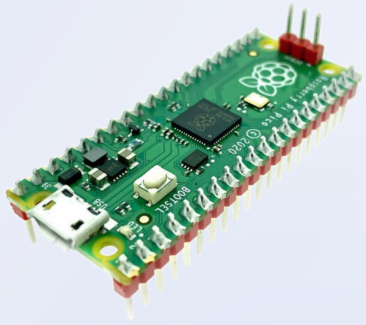
Topics covered today:

- What is Visual Studio Code
- What is PlatformIO
- How to start a new Project
- Compile and Upload Code
- Debugging

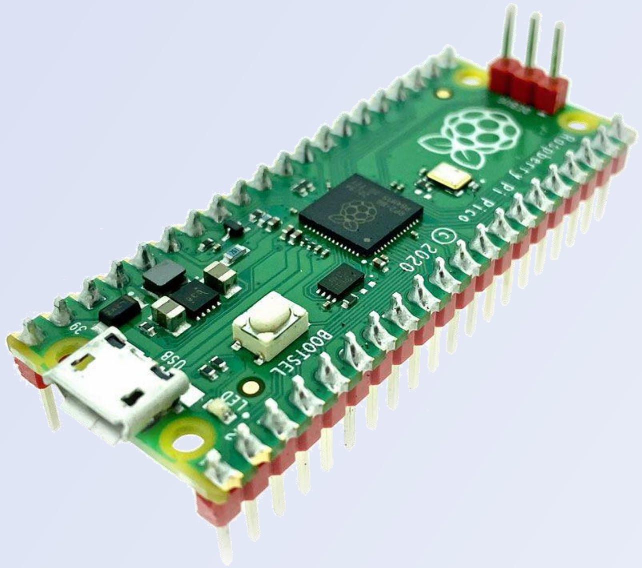
Development with Visual Studio Code and PlatformIO

Hardware used today:

- M5Stack Grey (ESP32)
- Raspberry Pi Pico and Pico Experimenting Kit
- Some jumper wires and USB cables



Chance to Win a Raspberry Pi Pico (with pre-soldered Headers)

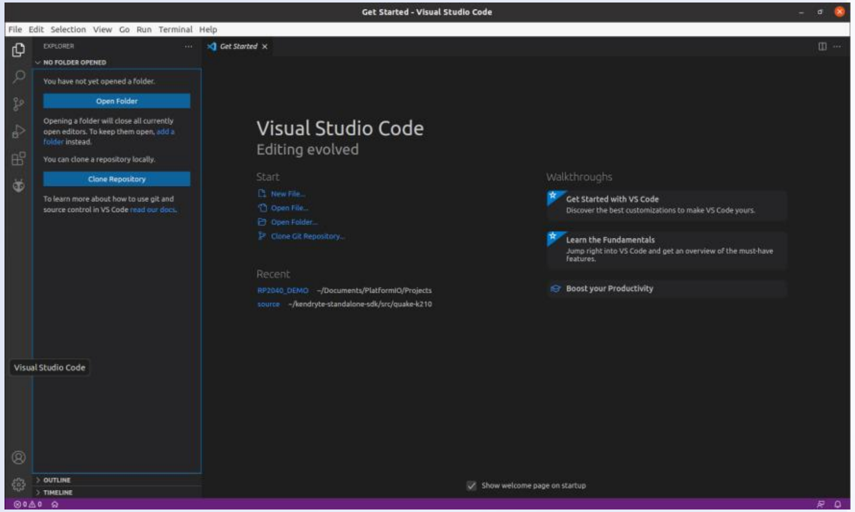


Raspberry Pi Pico with pinheaders
Incl. BONUS coupons
Soldered with industry precision by Eurocircuits
With Würth Elektronik Pinheaders
elektor

Visual Studio Code & PlatformIO

What is Visual Studio Code?

- Atom-based IDE by Microsoft
- Editor for various Languages (such as C, Python, HTML, JavaScript)
- Syntax highlighting / autocomplete
- Extensions through Marketplace
- Runs on Windows, Mac, Linux (x86 and ARM)
- IDE used for Rasperry Pi Pico



You have not yet opened a folder.

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

You can clone a repository locally.

To learn more about how to use git and source control in VS Code read our docs.

Visual Studio Code
Editing evolved

RP2040_DEMO ~/Documents/PlatformIO/Projects
source ~/kendryte-standalone-sdk/src/quake-k210

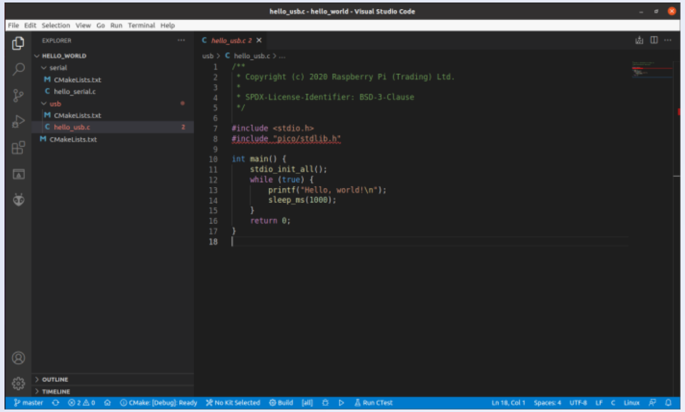
Get Started with VS Code
Discover the best customizations to make VS Code yours.

Learn the Fundamentals
Jump right into VS Code and get an overview of the must-have features.

Show welcome page on startup

What is Visual Studio Code?

- Code is open-source (MIT-License)
- Binary is not FLOSS
  (Free-Libre /Open Source Software)
- Binary has telemetry / tracking
  (can be disabled / opt-out)
- VSCodium has no telemetry / tracking
- VSCodium built from open-source code



```c
/**
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

#include <stdio.h>
#include "pico/stdlib.h"

int main() {
    stdio_init_all();
    while (true) {
        printf("Hello, world!\n");
        sleep_ms(1000);
    }
    return 0;
}
```

What is PlatformIO?

- Vendor independent Environment
- Cross-platform
- Open-source
- Extension e.g. for Visual Studio Code
- Platform and Library management
- Enables Debugging (multi platform)

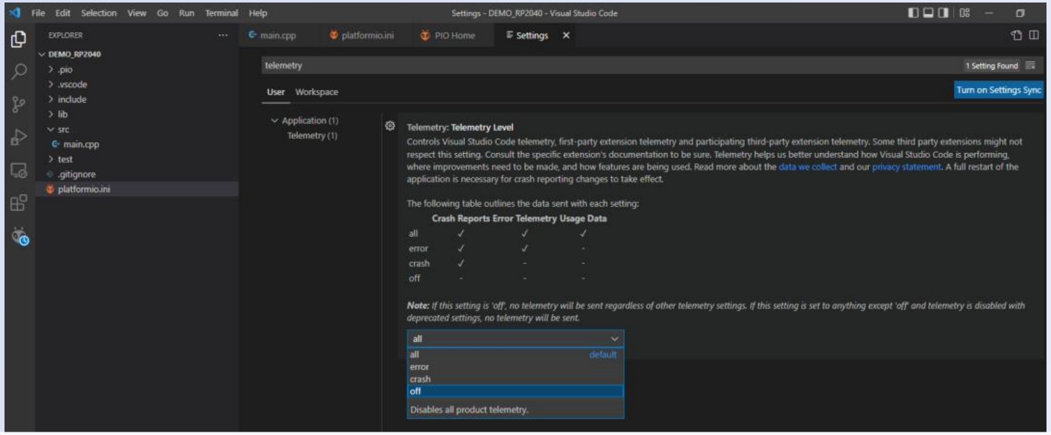
What is PlatformIO

- Multi Vendor support
- Multi-Framework support
  (Arduino, ESP-IDF, PicoSDK, Raspberry Pi…)
- Build-in Serial Monitor

ARMmbed ASR Atmel ESPRESSIF freescale GREENWAVES
infineon intel LATTICE SEMICONDUCTOR maxim integrated MICROCHIP NORDIC SEMICONDUCTOR 芯来科技 NUCLEI
NXP PULP Parallel Ultra Low Power RISC-V SiFive SAMSUNG ARTIK SILICON LABS
ST life.augmented Teensy TEXAS INSTRUMENTS WIZnet ARDUINO Energia libOpenCM3

Disable / Opt-out Visual Studio Code telemetry

- From File > Preferences > Settings (macOS: Code > Preferences > Settings), search for ***telemetry***,
- Set ***Telemetry: Telemetry Level*** setting to ***off***



Telemetry: Telemetry Level

Controls Visual Studio Code telemetry, first-party extension telemetry and participating third-party extension telemetry. Some third party extensions might not respect this setting. Consult the specific extension's documentation to be sure. Telemetry helps us better understand how Visual Studio Code is performing, where improvements need to be made, and how features are being used. Read more about the data we collect and our privacy statement. A full restart of the application is necessary for crash reporting changes to take effect.

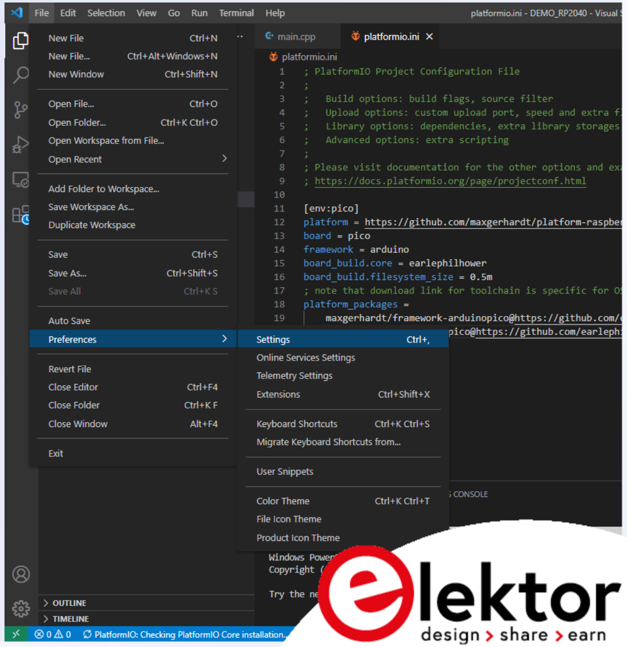
The following table outlines the data sent with each setting:

	Crash Reports	Error Telemetry	Usage Data
all	✓	✓	✓
error	✓	✓	-
crash	✓	-	-
off	-	-	-

***Note:** If this setting is 'off', no telemetry will be sent regardless of other telemetry settings. If this setting is set to anything except 'off' and telemetry is disabled with deprecated settings, no telemetry will be sent.*

all
all default
error
crash
off

Disables all product telemetry.



```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra f
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and ex
; https://docs.platformio.org/page/projectconf.html

[env:pico]
platform = https://github.com/maxgerhardt/platform-raspber
board = pico
framework = arduino
board_build.core = earlephilhower
board_build.filesystem_size = 0.5m
; note that download link for toolchain is specific for O
platform_packages =
    maxgerhardt/framework-arduinopico@https://github.com/
    pico@https://github.com/earleph
```

```
Windows Power
Copyright (

Try the ne
```

Download Links and Demo projects

- Webinar material on Github
  https://github.com/ElektorLabs/Elektor_Webinars
- Visual Studio Code
  https://code.visualstudio.com/
- PlatformIO
  https://platformio.org/

Prerequisite for ESP32

- Linux, MacOS or Windows
- Git
  (https://git-scm.com/download)
- Visual Studio Code installed
  (https://code.visualstudio.com/)
- PlatformIO installed (as Visual Studio Code extension)
  (https://platformio.org/install/ide?install=vscode)
- M5Stack Development Kit
  (https://www.elektor.de/catalogsearch/result/?q=m5stack)

  or ESP32 based board (e.g. JOY-iT NodeMCU ESP32)
  (https://www.elektor.com/19973 @ 11.66€)

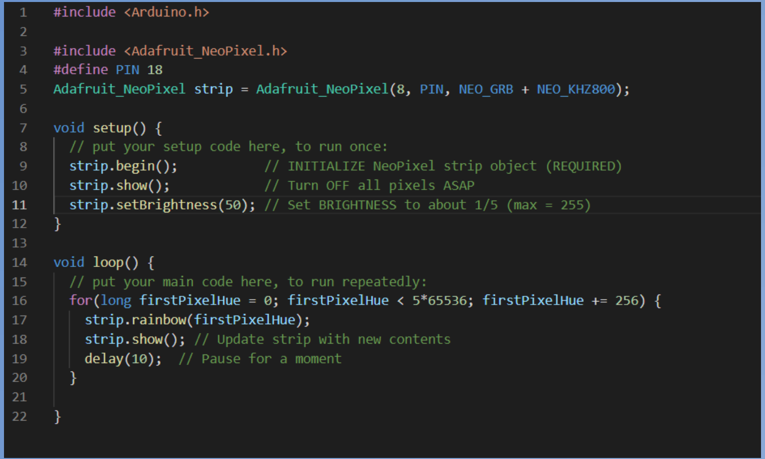
Setup an ESP32 Project

What we are going to do:

- Generate a new PlatformIO Project
- Initialize it as Git Repository
- Add a few new libraries
- Add some Code to it
- Upload Project
- Upload Filesystem (SPIFFS)

Time for a Demo

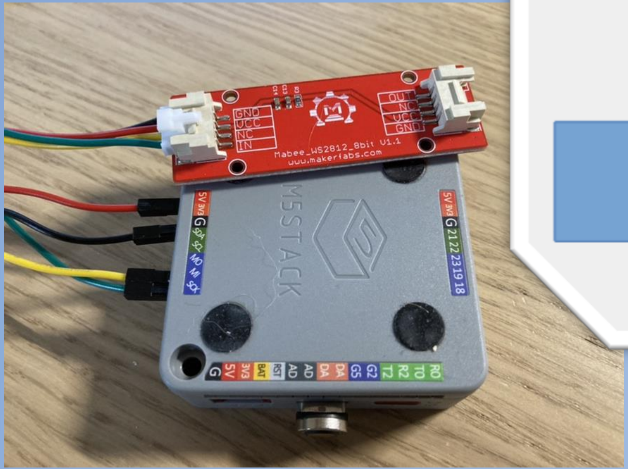
“Simple” ESP32 Project

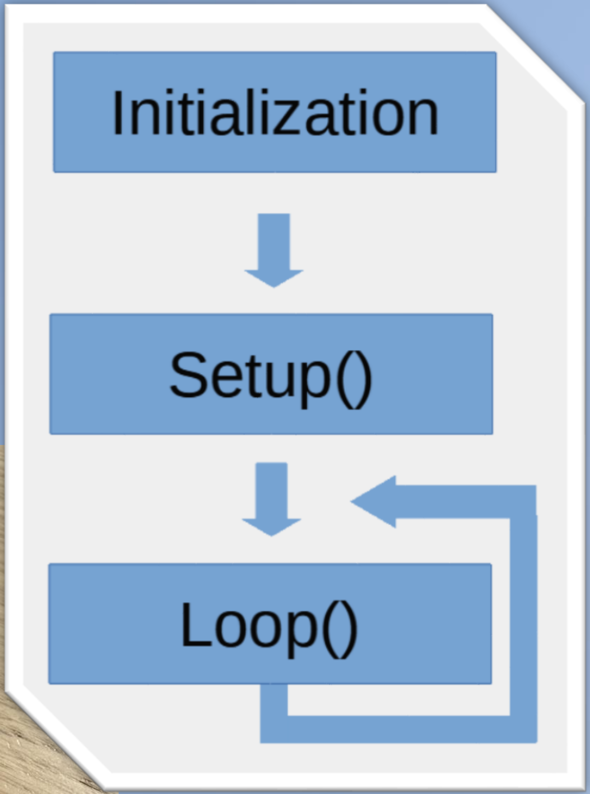


```cpp
#include <Arduino.h>

#include <Adafruit_NeoPixel.h>
#define PIN 18
Adafruit_NeoPixel strip = Adafruit_NeoPixel(8, PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  // put your setup code here, to run once:
  strip.begin();           // INITIALIZE NeoPixel strip object (REQUIRED)
  strip.show();            // Turn OFF all pixels ASAP
  strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
}

void loop() {
  // put your main code here, to run repeatedly:
  for(long firstPixelHue = 0; firstPixelHue < 5*65536; firstPixelHue += 256) {
    strip.rainbow(firstPixelHue);
    strip.show(); // Update strip with new contents
    delay(10);  // Pause for a moment
  }

}
```

WS2812 Connection:

- GND to GND
- VCC to 5V
- IN to GPIO18 / SCK
- NC to GPIO23 /MI





PlatformIO and ESP32-C3

- Only official support for ESP32-C3-DevKitM1
- Only supports Espressif IDF

- Unofficial support for Arduino Framework is worked on (https://github.com/platformio/platform-espressif32/issues/619)

Official and Unofficial Frameworks

Raspberry Pi RP2040

Raspberry Pi Pico and Frameworks

- PlatformIO supports only Arduino Mbed Framework
- No bare metal Pico SDK support (officially)
- No official support for Earle F. Philhower Arduino-Pico

- Unofficial support for bare metal Pico SDK and Earle F. Philhower Arduino-Pico exist
- Debugging is a bit “hacky” at the moment

Raspberry Pi Pico and Frameworks

To get the Arduino-Pico Framework in PlatformIO:

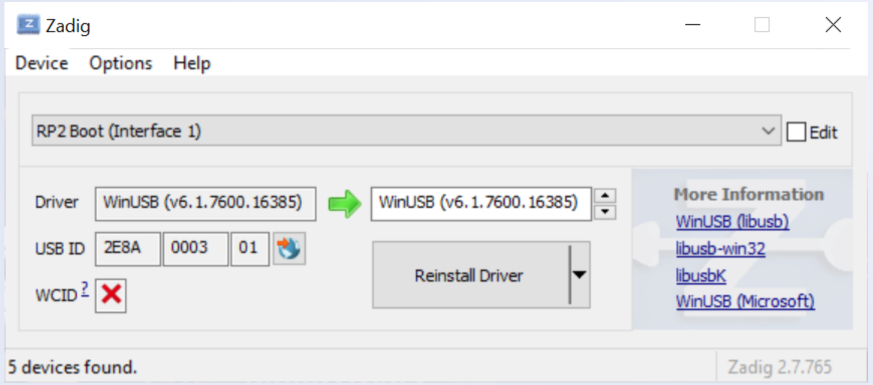
- Edit platformio.ini
- Change [env:pico] to:

```
platform = https://github.com/maxgerhardt/platform-raspberrypi.git
board = pico
framework = arduino
board_build.core = earlephilhower
board_build.filesystem_size = 0.5m
platform_packages =
  maxgerhardt/framework-arduinopico@https://github.com/earlephilhower/arduino-pico.git
  maxgerhardt/toolchain-pico@https://github.com/earlephilhower/pico-quick-toolchain/releases/download/1.3.3-a/x86_64-w64-mingw32.arm-none-eabi-ed6d983.220212.zip
  platformio/tool-openocd-raspberrypi@https://github.com/maxgerhardt/pio-openocd-picoprobe.git
```

Raspberry Pi Pico and Frameworks

To fix uploading problems on Windows:

- Download Zadig (https://zadig.akeo.ie/)
- Put Raspberry Pi Pico into Bootloader
- Open Zadig
- Search for RP2 Boot (Interface 1)
- Hit Install Driver



RP2 Boot (Interface 1)

Driver WinUSB (v6.1.7600.16385) → WinUSB (v6.1.7600.16385)

USB ID 2E8A 0003 01

WCID ? ✗

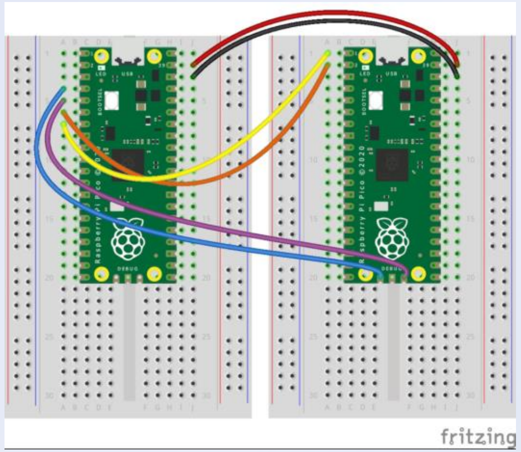
5 devices found.

Zadig 2.7.765

Raspberry Pi Pico and Debugging

- Needs a supported Debugger for SWD (e.g. Pico Probe)
- Firmware for Picoprobe needs to be downloaded

  (https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html#rp2040-device)
- Upload Firmware to Pico Probe (.uf2 file)
- Wire as follows:



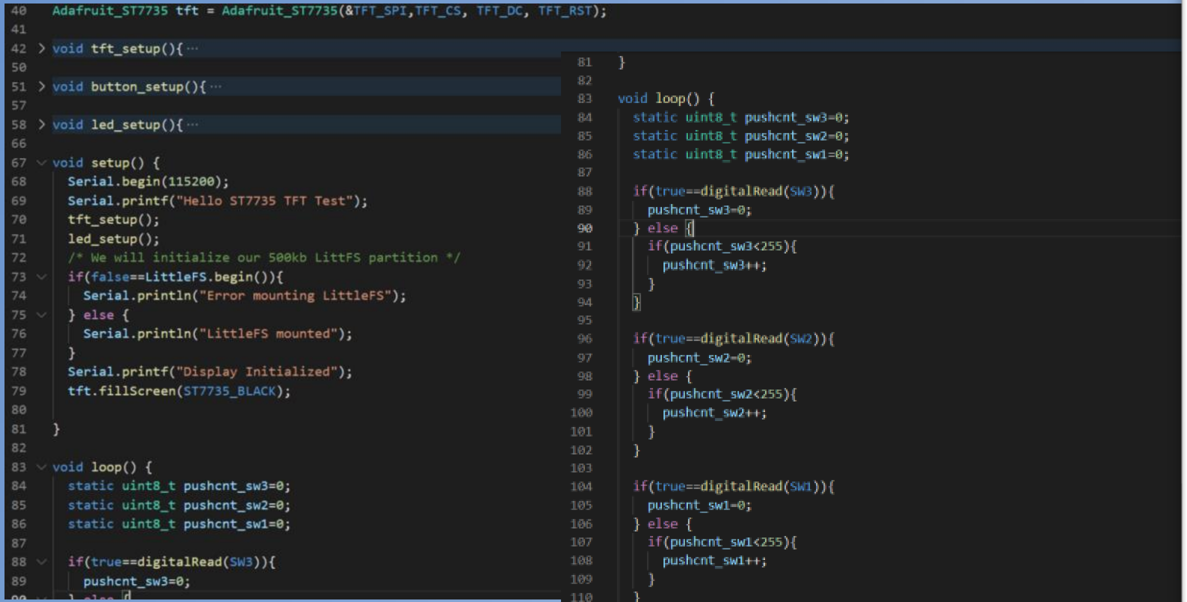
Quelle: https://www.digikey.de/maker-media/bed6afe4-af01-4a6a-b2d0-d768934f6ab5

Raspberry Pi Pico and Debugging

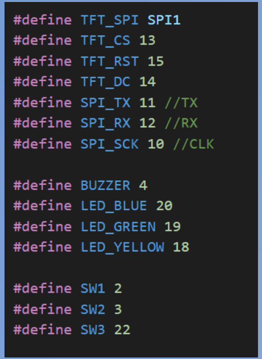
- Has some issues in PlatformIO
- Currently a hacky solution
- Need custom compiled OpenOCD
- PlatformIO may become slow / shows errors
- Allows to debug Arduino-pico projects

Time for a Demo

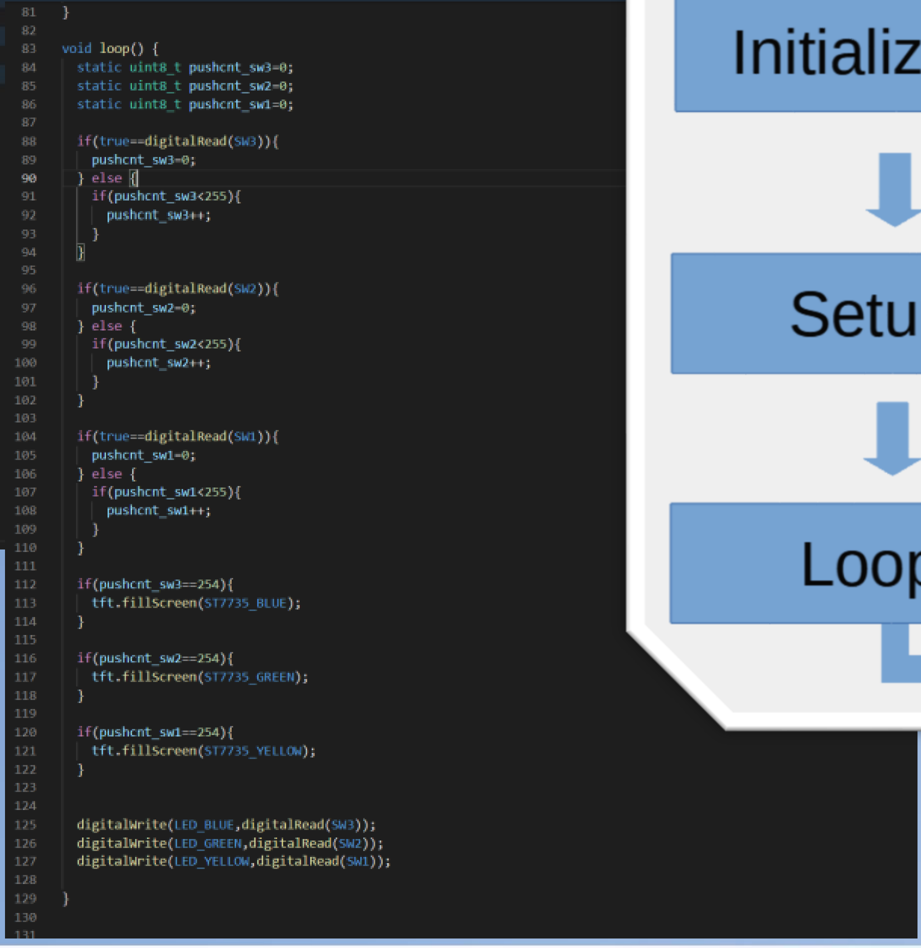
Raspberry Pi Pico and Ardino-Pico



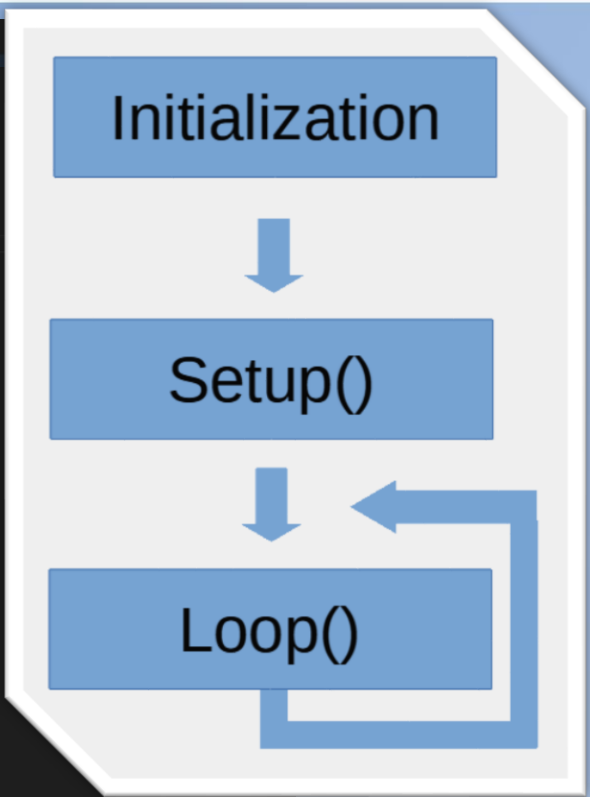
```
Adafruit_ST7735 tft = Adafruit_ST7735(&TFT_SPI,TFT_CS, TFT_DC, TFT_RST);

void tft_setup(){ ···

void button_setup(){ ···

void led_setup(){ ···

void setup() {
  Serial.begin(115200);
  Serial.printf("Hello ST7735 TFT Test");
  tft_setup();
  led_setup();
  /* We will initialize our 500kb LittFS partition */
  if(false==LittleFS.begin()){
    Serial.println("Error mounting LittleFS");
  } else {
    Serial.println("LittleFS mounted");
  }
  Serial.printf("Display Initialized");
  tft.fillScreen(ST7735_BLACK);

}

void loop() {
  static uint8_t pushcnt_sw3=0;
  static uint8_t pushcnt_sw2=0;
  static uint8_t pushcnt_sw1=0;

  if(true==digitalRead(SW3)){
    pushcnt_sw3=0;
```



```
#define TFT_SPI SPI1
#define TFT_CS 13
#define TFT_RST 15
#define TFT_DC 14
#define SPI_TX 11 //TX
#define SPI_RX 12 //RX
#define SPI_SCK 10 //CLK

#define BUZZER 4
#define LED_BLUE 20
#define LED_GREEN 19
#define LED_YELLOW 18

#define SW1 2
#define SW2 3
#define SW3 22
```



```
}

void loop() {
  static uint8_t pushcnt_sw3=0;
  static uint8_t pushcnt_sw2=0;
  static uint8_t pushcnt_sw1=0;

  if(true==digitalRead(SW3)){
    pushcnt_sw3=0;
  } else {
    if(pushcnt_sw3<255){
      pushcnt_sw3++;
    }
  }

  if(true==digitalRead(SW2)){
    pushcnt_sw2=0;
  } else {
    if(pushcnt_sw2<255){
      pushcnt_sw2++;
    }
  }

  if(true==digitalRead(SW1)){
    pushcnt_sw1=0;
  } else {
    if(pushcnt_sw1<255){
      pushcnt_sw1++;
    }
  }

  if(pushcnt_sw3==254){
    tft.fillScreen(ST7735_BLUE);
  }

  if(pushcnt_sw2==254){
    tft.fillScreen(ST7735_GREEN);
  }

  if(pushcnt_sw1==254){
    tft.fillScreen(ST7735_YELLOW);
  }


  digitalWrite(LED_BLUE,digitalRead(SW3));
  digitalWrite(LED_GREEN,digitalRead(SW2));
  digitalWrite(LED_YELLOW,digitalRead(SW1));

}
```

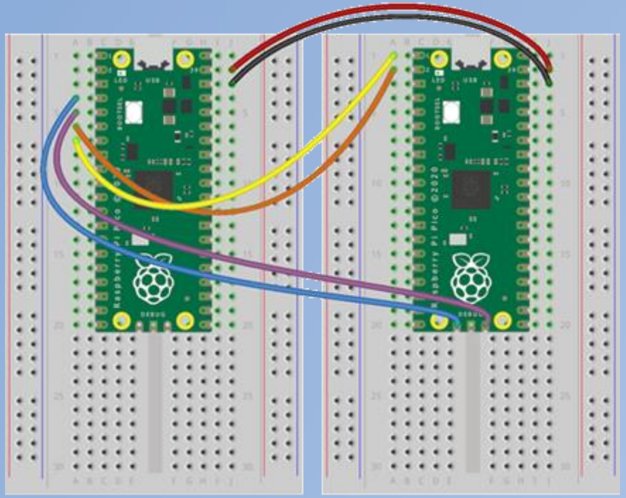


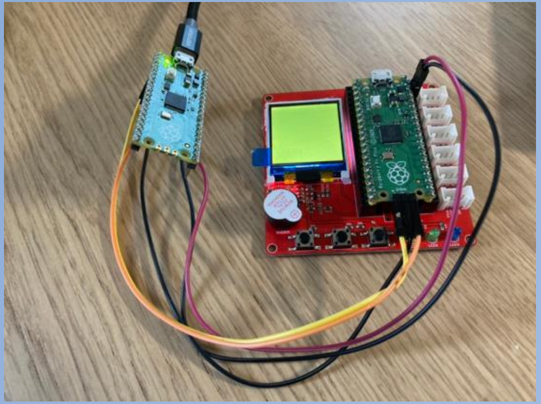
Add a Debugger

The good, the bad, the ugly

Add a Debugger

- Just three wires needed (SWD, SWCLK, GND)
- Allows to have a view inside
- Raspberry Pico as Debugger
- Uses Picoprobe Software
- Open-source
- Has precompiled binary





The next show coming up:

Elektor LabTalk Episode II: Not every board is for surfing

28/Apr/2022 @ 18:00 CEST
Live on YouTube (https://youtu.be/_kyvsGF_m0U)

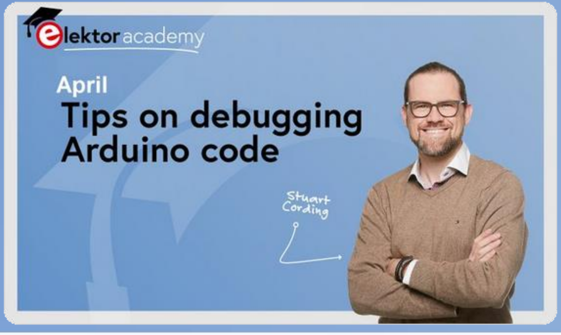
elektor
design > share > earn

Next training coming up:

Debugging Techniques for Arduino

28/4/2022 @ 19:00 CEST

Join on ClickMeeting, don't forget to register.



Next Webinar coming up:

IoT: Connect ESP32 to the cloud

9/Jun/2022 @ 16:00 CEST

Join on ClickMeeting, don't forget to register.

Additional resources:

- Elektor Webinars: https://www.elektormagazine.com/webinars
- Joseph Yiu , "**The Definitive Guide to ARM Cortex-M0 and Cortex M0+**" https://www.elektor.com/the-definitive-guide-to-armr-cortexr-m0-and-cortex
- Elektor, "**Raspberry Pi Pico Essentials**", https://www.elektormagazine.com/news/raspberry-pi-pico-essentials
- Elektor e-zine at https://www.elektormagazine.com/pages/newsletter