Arduino IDE with Olimex ESP32-S2-DevKit-Lipo-USB

This document applies for Olimex-made ESP32-S2-DevKit-Lipo-USB board and all its variants (ESP32-S2-DevKit-Lipo-USB-EA, ESP32-S2-WROVER-DevKit-Lipo-USB-EA, ESP32-S2-WROVER-DevKit-Lipo-USB).

Notice that ESP32-S2-DevKit-Lipo-USB and ESP32-S2-DevKit-Lipo are different boards. The ESP32-S2-DevKit-Lipo uses CH340T to handle the USB communication while ESP32-S2-DevKit-Lipo-USB uses the ESP32-S2 chip to handle USB communication. This guide does not apply to ESP32-S2-DevKit-Lipo but can be used as a basis.

This document will guide you on how to install and set the Arduino IDE software support for ESP32-S2-DevKit-Lipo-USB. By end of this document you should be able to program ESP32-S2-DevKit-Lipo-USB (or any of its variants) from your computer by using Arduino IDE over the USB connection.

The guide is based on the Windows installation of the software but the guide can be used as a basis by users of other operating systems. This guide requires active Internet connection (to visit download locations and follow links).

Before begin ensure you have the required hardware:

- A computer running operating system supported by Arduino IDE (Windows, macOS, Linux);
- Olimex ESP32-S2-DevKit-Lipo-USB (or any of its variants);
- fitting USB cable between the computer and the ESP32-S2 board; the board comes with micro USB connector.

1. Download and install Arduino IDE

If you still don’t have Arduino IDE, navigate to the following web-address to download it:

https://www.arduino.cc/en/software/

Download the version suitable for your operating system and install it (or extract it if you use the stand-alone version).

After the installation is complete, launch the Arduino IDE application.

2. Install the ESP32-S2 support for Arduino IDE

2.1. Install Espressif Arduino-ESP32 stable release. To do so follow the detailed instructions here:

https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html

Short summary of what needs to be done:

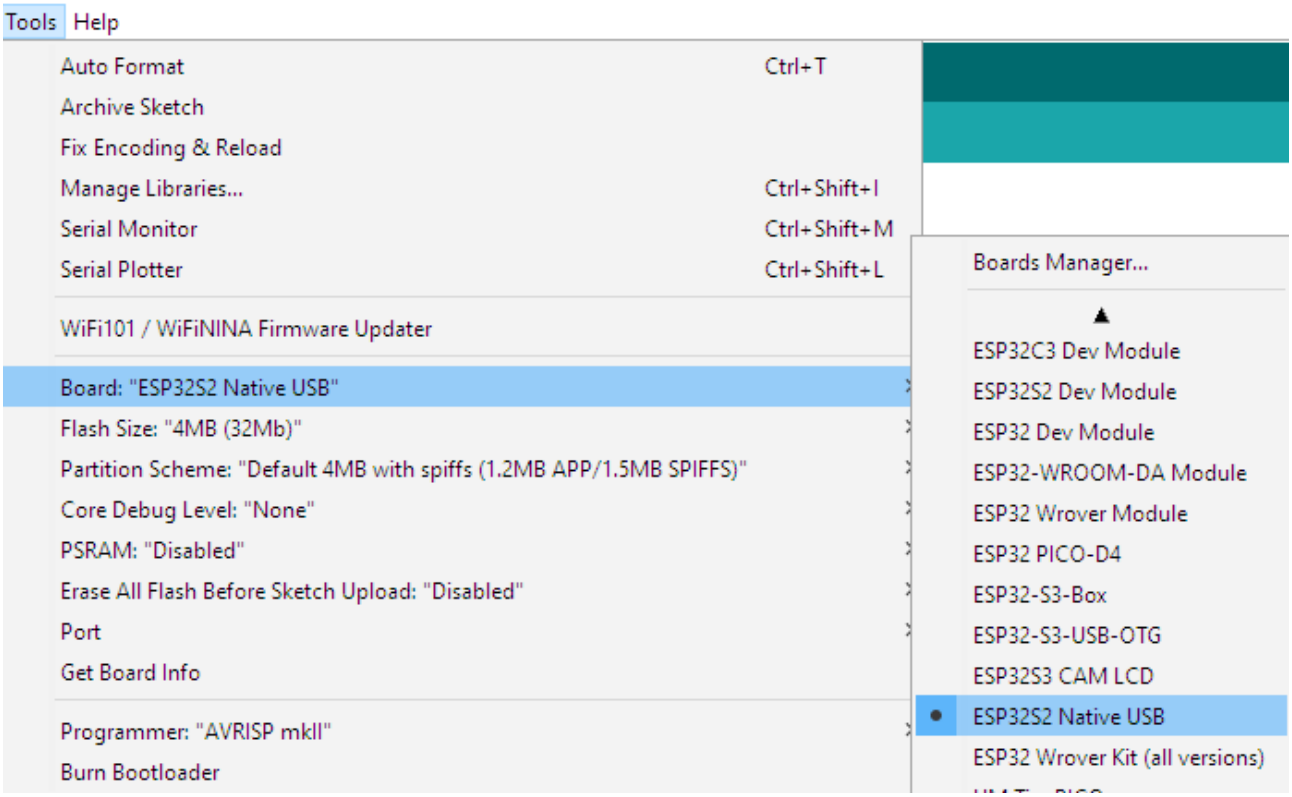
- In Arduino IDE navigate to “File” → “Preferences” and in the filed that says “Additional Boards Manager URLs” append the following json link (copy-paste):

https://espressif.github.io/arduino-esp32/package_esp32_index.json

- Navigate to “Tools” → “Boards” → “Board Manager” and search for the *esp32* platform. Install the latest version.

If you encounter problems during the package installation make sure to check all details on the official guide from the link on the previous page.

2.2. Restart Arduino IDE and select the configuration for the board. Navigate to “Tools” → “Board” →under “ESP32 Arduino” tab. Find in the list “ESP32S2 Native USB” and select it. You can also change the properties (ESP32-WROOM and ESP32-WROVER require slightly different settings, since ESP32-WROVER has extra 8MB PSRAM).



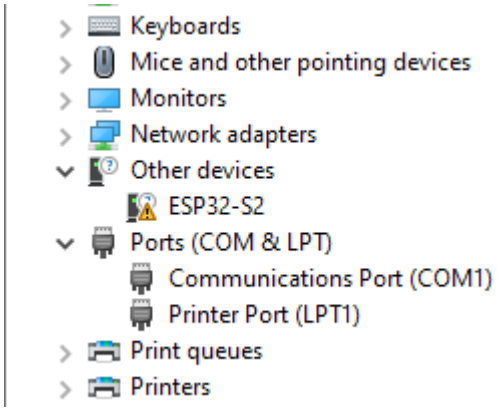
3. Put the board in bootloader mode

First connect the ESP32-S2 board to your computer via the micro USB connector using USB cable. In order to download code to ESP32-S2 you need to enter bootloader mode first. It is done by pressing and holding button BUT1, resetting the board by pressing RST1 button, then releasing button BUT1. The LED should stop blinking and new device would be visible in “Windows Device Manager”. At this point you can select the COM port and download your code. Skip to chapter “5. Upload a sketch”. If you have a problem and the driver doesn’t automatically get installed or you can’t identify the port refer to next chapter:

4. (Optional) Install Driver

Usually a driver installation is not required, despite that the unit gets marked with exclamation mark. If you however experience problems in “5. Upload a sketch”, follow install the drivers manually as explained below.

4.1. In order to start bootloader mode reset the board while holding BUT1. Navigate to “Windows Device Manager” to identify the entry for the board. Your device will most likely be on “Other devices” tab:

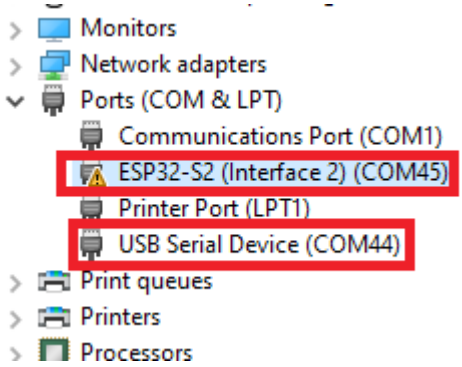


4.2. Go to https://zadig.akeo.ie/ and download Zadig software and run it;

4.3. Navigate to “Options” → “List all devices”. If there are multiple devices in the dropdown menu choose “ESP32-S2 (Interface 2)” option and from the menu below select “USB Serial (CDC)” driver;

4.4. Install Driver and wait a little bit (it takes about a minute);

4.5 Now in the “Windows Device Manager” you should be seeing 2 new devices. The one we need is called “USB Serial Device” (make sure to identify it, it will be different):



5. Upload a sketch

5.1. After you enter the bootloader mode as mentioned above, navigate to “Tools” → “Port…” select the COM of your device (make sure to identify it from “Windows Device Manager”):

5.2. Compile and upload your sketch – there is an example provided for the RGB LED at the end of this document;

5.3. If everything is alright the demo would be uploaded successfully, and error message that the board can't be reset would be thrown. You can disable reset after upload to not see the error. This is normal and expected:

```
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1638.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
ERROR: ESP32-S2 chip was placed into download mode using GPIO0.
esptool.py can not exit the download mode over USB. To run the app, reset the chip manually.
To suppress this error, set --after option to 'no_reset'.
To suppress this error, set --after option to 'no_reset'.
```

5.4. Reset the board with button RST1 (without BUT1 being pressed) to switch from bootloader mode to execution mode – this will make the board run the uploaded sketch. Remember every time you want to re-program the board you need to re-enter bootloader mode!

6. RGB LED demo code

The demo is based on Adafruit demo. It requires to also install the Adafruit NeoPixel library from "Sketch" → "Include Library" → "Manage libraries..." search for Adafruit NeoPixel and install it.

After that copy paste the code below in a new sketch:

```
/*
  Board: ESP32-S2-DevKit-USB
  Requires Adafruit NeoPixel library
 */

#include <Adafruit_NeoPixel.h>
#define PIN 18
#define NUMPIXELS 1
#define PERIOD 10  //ms

Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

int R=255, G=0, B=0;

void setup()
{
  pixels.begin();
}

void loop ()
{
  for (G=0; G<255; G++)
  {
    pixels.setPixelColor(0, pixels.Color(R, G, B));
    pixels.show();
    delay (PERIOD);
  }

  for (R=255; R>0; R--)
  {
```

```
  pixels.setPixelColor(0, pixels.Color(R, G, B));
  pixels.show();
  delay (PERIOD);
 }

 for (B=0; B<255; B++)
 {
  pixels.setPixelColor(0, pixels.Color(R, G, B));
  pixels.show();
  delay (PERIOD);
 }

 for (G=255; G>0; G--)
 {
  pixels.setPixelColor(0, pixels.Color(R, G, B));
  pixels.show();
  delay (PERIOD);
 }

 for (R=0; R<255; R++)
 {
  pixels.setPixelColor(0, pixels.Color(R, G, B));
  pixels.show();
  delay (PERIOD);
 }

 for (B=255; B>0; B--)
 {
  pixels.setPixelColor(0, pixels.Color(R, G, B));
  pixels.show();
  delay (PERIOD);
 }
}
```