Upgrading ESP32 Firmware

Created by Brent Rubell

https://learn.adafruit.com/upgrading-esp32-firmware

Last updated on 2021-11-15 07:50:03 PM EST
# Table of Contents

## Overview 3
- Why would I update my ESP32’s firmware? 3
- Parts 4

## Upgrade All-in-One ESP32 AirLift Firmware 6
- Uploading Serial Passthrough Code 6
- Install esptool.py 8
- Uploading nina-fw with esptool 9
- Verifying the Upgraded Firmware Version 10
- Arduino 10
- CircuitPython 10

## Upgrade External ESP32 Airlift Firmware 11
- External AirLift FeatherWing, Shield, or ItsyWing 11
- Upload Serial Passthrough code for Feather or ItsyBitsy 12
- External AirLift Breakout 14
- Code Usage 16
- Install esptool.py 17
- Burning nina-fw with esptool 18
- Verifying the Upgraded Firmware Version 19
- Arduino 19
- CircuitPython 20

## Upgrade RP2040 AirLift Firmware 20
- Upload Serial Passthrough Code 20
- Download NINA Firmware 22
- Flash AirLift using Web Serial ESPTool 22
- Enable Chrome’s Web Serial API 23
- Flash AirLift using esptool.py 25
- Burning nina-fw with esptool 26
- Verifying the Upgraded Firmware Version 27
Overview

If you want to keep the firmware on your ESP32 WiFi-BLE co-processor up-to-date, you'll need to update the firmware on the ESP32.

You're going to turn your board into a USB-to-Serial converter to flash new firmware to your ESP32 - no extra hardware required!

This process is mostly setup and should take from 10 to 20 minutes.

This guide is not for when you are running Arduino/MicroPython/FreeRTOS/etc *directly* on the ESP32, this is only for using the ESP32 as an AirLift/WiFi co-processor!

To support BLE on the ESP32 AirLift, you'll need NINA_W102-1.7.1.bin or later.

Why would I update my ESP32's firmware?

Using an ESP32 as a WiFi-BLE co-processor is a way to connect your CircuitPython and Arduino projects to the internet. Having WiFi managed by a separate chip means your code is simpler, you don't have to cache socket data, or compile in & debug an SSL library.
Adafruit ships a variety of products which use the ESP32 as a WiFi-BLE co-processor with a variant of the Arduino nina-fw core (https://adafruit.com/FWj). This firmware is programmed to the ESP32 at the Adafruit factory. If you wish to update to a newer version of nina-fw, you'll need to program it to the ESP32.

BLE is supported on the ESP32 co-processor only with version NINA_W102-1.7.1.bin or later of the firmware (released in October 2020). If you want BLE support, it is quite likely you'll need to upgrade.

**Parts**

**External ESP32 Co-Processors**

If you already have a project which uses a popular microcontroller (like the ATMega328 or ATSAMD51), you can easily add WiFi by using an externally connected ESP32 module.

![Adafruit AirLift – ESP32 WiFi Co-Processor Breakout Board](https://www.adafruit.com/product/4201)

Give your plain ol' microcontroller project a lift with the Adafruit AirLift - a breakout board that lets you use the powerful ESP32 as a WiFi co-processor. You probably...

https://www.adafruit.com/product/4201

![Adafruit AirLift FeatherWing – ESP32 WiFi Co-Processor](https://www.adafruit.com/product/4264)

Give your Feather project a lift with the Adafruit AirLift FeatherWing - a FeatherWing that lets you use the powerful ESP32 as a WiFi co-processor. You probably have your...

https://www.adafruit.com/product/4264
Adafruit AirLift Shield - ESP32 WiFi Co-Processor
Give your Arduino project a lift with the Adafruit AirLift Shield - a shield that lets you use the powerful ESP32 as a WiFi co-processor. You probably have your favorite...
https://www.adafruit.com/product/4285

Adafruit AirLift Bitsy Add-On – ESP32 WiFi Co-Processor
Give your ItsyBitsy project a lift with the Adafruit AirLift Bitsy Add-On - a daughterboard that lets you use the powerful ESP32 as a WiFi co-processor. You probably have your...
https://www.adafruit.com/product/4363

ESP32 Co-Processor All-in-One Boards

Don't want to add extra hardware to your project? Consider grabbing a board which has an ESP32 WiFi co-processor built-in.

Adafruit PyPortal - CircuitPython Powered Internet Display
PyPortal, our easy-to-use IoT device that allows you to create all the things for the “Internet of Things” in minutes. Make custom touch screen interface...
https://www.adafruit.com/product/4116
**Adafruit Metro M4 Express AirLift (WiFi) - Lite**

Give your next project a lift with AirLift - our witty name for the ESP32 co-processor that graces this Metro M4. You already know about the Adafruit Metro...

https://www.adafruit.com/product/4000

---

**Materials**

1 x **USB Cable**

USB cable - USB A to Micro-B - 3 foot long

https://www.adafruit.com/product/592

---

**Upgrade All-in-One ESP32 AirLift Firmware**

To support BLE on the ESP32 AirLift, you'll need NINA_W102-1.7.1.bin or later.

---

**Uploading Serial Passthrough Code**

First, back up any code and files you have on your CIRCUITPY drive. It will be overwritten by the code you're going to upload to your board. You should not end up losing any files on the QSPI flash, but it's a good idea to back them up anyways.

Download the UF2 for your board to your Desktop.

- **PyPortal_ESP32_Passthru.UF2**
  - https://adafru.it/Ep1

- **Metro_M4_WiFi_ESP32_Passthru.UF2**
  - https://adafru.it/END

- **MatrixPortal_ESP32_Passthru.UF2**
  - https://adafru.it/Obx

- **PyBadge_M4_with_AirLift_Wing_Passthru.UF2**
Find the reset button on your board. It's a small, black button, and on most of the boards, it will be the only button available.

[Image of a circuit board with a reset button]

Tap this button twice to enter the bootloader. If it doesn't work on the first try, don't be discouraged. The rhythm of the taps needs to be correct and sometimes it takes a few tries.

Once successful, the RGB LED on the board will flash red and then stay green. A new drive will show up on your computer. The drive will be called `boardnameBOOT` where `boardname` is a reference to your specific board. For example, a Feather will have `FEATHERBOOT` and a Trinket will have `TRINKETBOOT` etc. Going forward we'll just call the boot drive `BOOT`.

[Image of a PC file explorer with Boot drive]

The board is now in bootloader mode. Now find the UF2 file you downloaded. Drag that file to the `BOOT` drive on your computer.
The lights should flash again, **BOOT** will disappear. It should re-enumerate USB and appear as a COM or Serial port on your computer. Make a note of the serial port by checking the Device Manager (windows) or typing `ls /dev/cu*` or `/dev/tty*` (Mac or Linux) in a terminal.

If you see your board listed in the terminal: Proceed to the Uploading nina-fw with esptool section of this page.

```
$ ls /dev/cu.*
/dev/cu.Bluetooth-Incoming-Port /dev/cu.usbmodem1432201
```

This guide assumes you have Python3 installed. If you have not installed it, navigate to the Python downloads page (https://www.python.org/downloads) and install the latest release.

**Install esptool.py**

Esptool is an application which can communicate with the ROM **bootloader** (https://adafruit.it/LKe) in Espressif chips.

To install esptool, run the following in your terminal:

```
pip3 install esptool
```
Uploading nina-fw with esptool

Click the link below to download the latest nina-fw .bin file. Unzip it and save the .bin file to your desktop.

Latest nina-fw .bin file

https://adafru.it/G3D

If you're using macOS or Linux - run the following command, replacing /dev/ttyACM0 with the serial port of your board and NINA_W102-1.6.0 with the binary file you're flashing to the ESP32.

```bash
esptool.py --port /dev/ttyACM0 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

If you're using Windows - run the following command, replacing COM7 with the serial port of your board and NINA_W102-1.6.0 with the binary file you're flashing to the ESP32.

```bash
esptool.py --port COM7 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

The command should detect the ESP32 and will take a minute or two to upload the firmware. The NeoPixel on your board will flicker and flash as the firmware uploads.

If ESPTool doesn't detect the ESP32, make sure you've uploaded the correct .UF2 file to the bootloader.
Once the firmware is fully uploaded, the ESP32 will reset and the NeoPixel will glow blue.

Verifying the Upgraded Firmware Version

Arduino

To verify everything is working correctly, we'll load up either an Arduino sketch or CircuitPython code.

If you were previously using your ESP32 with Arduino, you should load up an Arduino sketch to verify everything is working properly and the version of the nina-fw correlates with the version the sketch reads.

Open up File->Examples->WiFiNINA->ScanNetworks and upload the sketch. Then, open the Serial Monitor. You should see the firmware version printed out to the serial monitor.

CircuitPython

If you were previously using your ESP32 with CircuitPython, you'll need to first reinstall CircuitPython firmware (UF2) for your board. The QSPI flash should have
retained its contents. If you don't see anything on the CIRCUITPY volume, copy files from the backup you made earlier over to CIRCUITPY.

To verify the new ESP32 WiFi firmware version is correct, follow the Connect to WiFi step in this guide (https://adafruit.it/Eao) and come back here when you've successfully ran the code. The REPL output should display the firmware version you uploaded.

code.py output:
ESP32 SPI webclient test
ESP32 found and in idle mode
Firmware vers. bytearray(b'1.3.0\x00')
MAC addr: ['0x19', '0x5c', '0xd', '0x33', '0x4f', '0xc4']

Upgrade External ESP32 Airlift Firmware

To support BLE on the ESP32 AirLift, you'll need NINA_W102-1.7.1.bin or later.

External AirLift FeatherWing, Shield, or ItsyWing

External AirLift boards have three optional ESP32 control pins which are not connected by default:

- ESPGPIO0
- ESPRX
- ESPTX
Make sure to solder each of these pads together. You will not be able to upload firmware to your ESP32 if they are not connected.

Upload Serial Passthrough code for Feather or ItsyBitsy

First, back up any code and files you have on your CIRCUITPY drive. It will be overwritten by the code you're going to upload to your board. You should not end up losing any files on the QSPI flash, but it's a good idea to back them up anyways.

This section is only for an AirLift FeatherWing with a Feather M4, or an AirLift BitsyWing with an ItsyBitsy M4. If you are using a different hardware combination - scroll down to the "External AirLift Breakout" section.
Download the UF2 for your board to your Desktop.

**Feather M0/M4 Passthrough UF2**
[https://adafru.it/OYF](https://adafru.it/OYF)

**Feather NRF52840 Passthrough UF2**
[https://adafru.it/PTE](https://adafru.it/PTE)

**Itsybitsy Passthrough UF2**
[https://adafru.it/IEK](https://adafru.it/IEK)

Find the reset button on your board. It's a small, black button, and on most of the boards, it will be the only button available.

![Image of a Feather board with reset button highlighted](image)

Tap this button twice to enter the bootloader. If it doesn't work on the first try, don't be discouraged. The rhythm of the taps needs to be correct and sometimes it takes a few tries.

Once successful, the RGB LED on the board will flash red and then stay green. A new drive will show up on your computer. The drive will be called `boardnameBOOT` where `boardname` is a reference to your specific board. For example, a Feather will have `FEATHERBOOT` and a Trinket will have `TRINKETBOOT` etc. Going forward we'll just call the boot drive `BOOT`
The board is now in bootloader mode. Now find the UF2 file you downloaded. Drag that file to the **BOOT** drive on your computer in your operating system file manager/finder.

The lights should flash again, **BOOT** will disappear. Your board should re-enumerate USB and appear as a COM or Serial port on your computer. Make a note of the serial port by checking the Device Manager (Windows) or typing `ls /dev/cu*` or `/dev/tty*` (Mac or Linux) in a terminal.

If your board is listed in the terminal, proceed to the Uploading nina-fw with esptool section of this guide.

---

**External AirLift Breakout**

You'll be turning your Arduino board into a USB to Serial converter. To do this, you'll need a special Arduino sketch named SerialESPPassthrough.ino and an Arduino-compatible board with Native USB support such as the Adafruit Metro M4.

You will also need to make the following connections between the board and the AirLift Breakout:

- Board Pin 12 to ESP32_ResetN
- Board Pin 10 to ESP32_GPIO0
- Board TX to RXI
- Board RX to TX0

Click Download: Project ZIP to download the code below.

```cpp
// SPDX-FileCopyrightText: 2018 Arduino SA

©Adafruit Industries
```
SerialNINAPassthrough - Use esptool to flash the ESP32 module
For use with PyPortal, Metro M4 WiFi...

Copyright (c) 2018 Arduino SA. All rights reserved.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

#include <Adafruit_NeoPixel.h>

unsigned long baud = 115200;

#if defined(ADAFRUIT_FEATHER_M4_EXPRESS) ||
    defined(ADAFRUIT_FEATHER_M0_EXPRESS) ||
    defined(ARDUINO_AVR_FEATHER32U4) ||
    defined(ARDUINO_NRF52840_FEATHER) ||
    defined(ADAFRUIT_ITSYBITSY_M0) ||
    defined(ADAFRUIT_ITSYBITSY_M4_EXPRESS) ||
    defined(ARDUINO_AVR_ITSYBITSY32U4_3V) ||
    defined(ARDUINO_NRF52_ITSYBITSY)
    // Configure the pins used for the ESP32 connection
#define SerialESP32 Serial1
#define SPIWIFI SPI // The SPI port
#define SPIWIFI_SS 13 // Chip select pin
#define ESP32_RESETN 12 // Reset pin
#define SPIWIFI_ACK 11 // a.k.a BUSY or READY pin
#define ESP32_GPIO0 10
#define NEPIXEL_PIN 8
#elif defined(ARDUINO_AVR_FEATHER328P)
#define SerialESP32 Serial1
#define SPIWIFI SPI // The SPI port
#define SPIWIFI_SS 4 // Chip select pin
#define ESP32_RESETN 3 // Reset pin
#define SPIWIFI_ACK 2 // a.k.a BUSY or READY pin
#define ESP32_GPIO0 -1
#define NEPIXEL_PIN 8
#elif defined(TEENSYDUINO)
#define SerialESP32 Serial1
#define SPIWIFI SPI // The SPI port
#define SPIWIFI_SS 5 // Chip select pin
#define ESP32_RESETN 6 // Reset pin
#define SPIWIFI_ACK 9 // a.k.a BUSY or READY pin
#define ESP32_GPIO0 -1
#define NEPIXEL_PIN 8
#elif defined(ARDUINO_NRF52832_FEATHER)
#define SerialESP32 Serial1
#define SPIWIFI SPI // The SPI port
#define SPIWIFI_SS 16 // Chip select pin
#define ESP32_RESETN 15 // Reset pin
#define SPIWIFI_ACK 7 // a.k.a BUSY or READY pin
#define ESP32_GPIO0 -1

©Adafruit Industries
#define NEOPIXEL_PIN  8
#elif !defined(SPIWIFI_SS)  // if the wifi definition isn't in the board variant
// Don't change the names of these #define's! they match the variant ones
#define SerialESP32 Serial1
#define SPIWIFI SPI
#define SPIWIFI_SS 10  // Chip select pin
#define SPIWIFI_ACK 7  // a.k.a BUSY or READY pin
#define ESP32_RESETN 5  // Reset pin
#define ESP32_GPIO0 -1  // Not connected
#define NEOPIXEL_PIN  8
#endif

#if defined(ADAFRUIT_PYPORTAL)
#define PIN_NEOPIXEL  2
#elif defined(ADAFRUIT_METRO_M4_AIRLIFT_LITE)
#define PIN_NEOPIXEL  40
#endif

Adafruit_NeoPixel pixel = Adafruit_NeoPixel(1, PIN_NEOPIXEL, NEO_GRB + NEO_KHZ800);

void setup() {
  Serial.begin(baud);
  pixel.begin();
  pixel.setPixelColor(0, 10, 10, 10); pixel.show();
  while (!Serial);
  pixel.setPixelColor(0, 50, 50, 50); pixel.show();
  delay(100);
  SerialESP32.begin(baud);
  pinMode(SPIWIFI_SS, OUTPUT);
  pinMode(ESP32_GPIO0, OUTPUT);
  pinMode(ESP32_RESETN, OUTPUT);
  // manually put the ESP32 in upload mode
  digitalWrite(ESP32_GPIO0, LOW);
  digitalWrite(ESP32_RESETN, LOW);
  delay(100);
  digitalWrite(ESP32_RESETN, HIGH);
  pixel.setPixelColor(0, 20, 20, 0); pixel.show();
  delay(100);
}

void loop() {
  while (Serial.available()) {
    pixel.setPixelColor(0, 10, 0, 0); pixel.show();
    SerialESP32.write(Serial.read());
  }
  while (SerialESP32.available()) {
    pixel.setPixelColor(0, 0, 0, 10); pixel.show();
    Serial.write(SerialESP32.read());
  }
}

Code Usage

Unzip the file, and open the SerialESPPassthrough.ino file in the Arduino IDE.
If you’re using the AirLift FeatherWing, AirLift Shield or AirLift Bitsy Add-On, use the PassThrough UF2 instructions above.

If you have an AirLift Breakout (or are manually wiring up any of the boards above), change the following pin definitions in the sketch to match your wiring:

```c
//elif !defined(SPIWIFI_SS)  // if the wifi definition isn't in the board variant
// Don't change the names of these #define's! they match the variant ones
#define SerialESP32 Serial1
#define SPIWIFI SPI
#define SPIWIFI_SS 10  // Chip select pin
#define SPIWIFI_ACK 7  // a.k.a BUSY or READY pin
#define ESP32_RESETN 5  // Reset pin
#define ESP32_GPIO0 -1  // Not connected
#define NEOPIXEL_PIN 8
#endif
```

Using the Arduino IDE, upload the code to your board (Sketch->Upload).

After uploading, the board should enumerate USB and appear as a COM or Serial port on your computer.

Make a note of the serial port by checking the Device Manager (Windows) or typing in `ls /dev/cu*` or `/dev/tty*` (Mac or Linux) in a terminal.

```
$ ls /dev/cu.*
/dev/cu.Bluetooth-Incoming-Port /dev/cu.usbmodem1432201
```

This guide assumes you have Python3 installed. If you have not installed it, navigate to the Python downloads page (https://www.python.org/downloads) and install the latest release.

Install esptool.py

Esptool is an application which can communicate with the ROM bootloader (https://adafruit.it/LKe) in Espressif chips.

To install esptool, run the following in your terminal:

```
pip3 install esptool
```
Burning nina-fw with esptool

Click the link below to download the latest nina-fw .bin file. Unzip it and save the .bin file to your desktop.

Latest nina-fw .bin file
https://adafru.it/G3D

If you’re using macOS or Linux - run the following command, replacing /dev/ttyACM0 with the serial port of your board and NINA_W102-1.6.0 with the binary file you’re flashing to the ESP32.

```bash
esptool.py --port /dev/ttyACM0 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

If you’re using Windows - run the following command, replacing COM7 with the serial port of your board and NINA_W102-1.6.0 with the binary file you’re flashing to the ESP32.

```bash
esptool.py --port COM7 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

The command should detect the ESP32 and will take a minute or two to upload the firmware.

If ESPTool doesn’t detect the ESP32, make sure you’ve uploaded the correct .UF2 file to the bootloader and are using the correct serial port.
Once the firmware is fully uploaded, the ESP32 will reset.

Verifying the Upgraded Firmware Version

To verify everything is working correctly, we'll load up either an Arduino sketch or CircuitPython code. At this point, you must desolder the connections between the Optional ESP32 control pins you made earlier using a [solder sucker](https://adafru.it/FWk) or a bit of [solder wick](https://adafru.it/yrC).

Arduino

If you were previously using your ESP32 with Arduino, you should load up an Arduino sketch to verify everything is working properly and the version of the nina-fw correlates with the version the sketch reads.

Open up File->Examples->WiFiNINA->ScanNetworks and upload the sketch. Then, open the Serial Monitor. You should see the firmware version printed out to the serial monitor.
CircuitPython

If you were previously using your ESP32 project with CircuitPython, you'll need to first reinstall CircuitPython firmware (UF2) for your board. The QSPI flash should have retained its contents. If you don't see anything on the CIRCUITPY volume, copy files from the backup you made earlier to CIRCUITPY.

To verify the new ESP32 WiFi firmware version is correct, follow the Connect to WiFi step in this guide (https://adafru.it/Eao) and come back here when you've successfully ran the code. The REPL output should display the firmware version you flashed.

```
code.py output:
ESP32 SPI webclient test
ESP32 found and in idle mode
Firmmare vers. bytearray(b'1.3.0\x00')
MAC addr: ['0x19', '0x5c', '0xd', '0x33', '0x4f', '0xc4']
```

Upgrade RP2040 AirLift Firmware

Upload Serial Passthrough Code

If you have already installed CircuitPython, back up any code and files you have on your CIRCUITPY drive. It will be overwritten by the code you're going to upload to your board. You should not end up losing any files on the QSPI flash, but it's a good idea to back them up anyways.

Download the UF2 for your board to your Desktop:

- Feather RP2040 Passthrough UF2
  [https://adafru.it/TE0](https://adafru.it/TE0)
- Raspberry Pi Pico RP2040 Passthrough UF2
  [https://adafru.it/TE1](https://adafru.it/TE1)
To enter bootloader mode, start with your RP2040 board unplugged from USB.

Press and hold the BOOTSEL button (highlighted in red in the image of the Feather RP2040, but all RP2040 boards should include this button), continue to hold it while plugging it into USB, and wait for the RPI-RP2 drive to appear before releasing the button.

You will see a new disk drive appear called RPI-RP2. The board is now in bootloader mode.

Find the UF2 file you downloaded. Drag that file to the RPI-RP2 drive on your computer in your operating system file manager/finder.

The board's LED should flash, RPI-RPI2 will disappear. Your board should re-enumerate USB and appear as a COM or Serial port on your computer. Make a note of the serial port by checking the Device Manager (Windows) or typing `ls /dev/cu*` or `/dev/tty*` (Mac or Linux) in a terminal.
Download NINA Firmware

Click the link below to download the latest version of the NINA firmware. Unzip it and save the .bin file to your desktop.

To support BLE on the ESP32 AirLift, you'll need NINA_W102-1.7.1.bin or later.

Download the latest nina-fw .bin file
https://adafru.it/G3D

Next, you'll need to flash the firmware to your ESP32 AirLift module. If you're using the Google Chrome browser, you may follow the instructions below for programming using our web-based ESPTool.

Otherwise, scroll down and follow the instructions for flashing using the Python esptool.py program.

Flash AirLift using Web Serial ESPTool

If you have a computer with the Google Chrome browser, we've created the WebSerial ESPTool as an option for programming ESP32 boards.

Safari and Firefox, etc are not supported because we need Web Serial and only Chrome is supporting it to the level needed. If you're using an unsupported browser, you'll need to upgrade using the Python esptool.py program on your computer (Scroll down to Upgrade with esptool.py,)
Enable Chrome's Web Serial API

At the time of this tutorial, you'll need to enable the Serial API, which is really easy.

Visit `chrome://flags` from within Chrome.
Find and enable the Experimental Web Platform features

Restart Chrome

Next, navigate to [https://adafruit.github.io/Adafruit_WebSerial_ESPTool](https://adafruit.github.io/Adafruit_WebSerial_ESPTool) (https://adafruit.github.io/Adafruit_WebSerial_ESPTool). In the top-right corner of your browser, select 115200 as the baud rate and click the Connect button.

![Connect Button](Image)

You will get a pop-up asking you to select the Pico's COM or Serial port. You may want to remove all other USB devices so only the ESP32-S2 board is attached, that way there's no confusion over multiple ports!

Click Connect.
It may timeout for a bit until it succeeds. On success, you will see that it is Connected and will print out a unique MAC address identifying the board.

Once you have successfully connected, a command toolbar will appear.

Verify that the offset is 0x0 and choose the nina-fw .bin file you downloaded above.

Click the program button to flash the firmware to your ESP32.
ESPTool will take a few minutes to write firmware to your device. After it's complete, the progress bar will disappear and the console will print "To run the new firmware,..."

Press the Reset button (or, on the RP2040 Pico, unplug your device from USB power) to get out of the ROM bootloader.

---

Flash AirLift using esptool.py

Esptool is an application that can communicate with the ROM bootloader (https://adafru.it/LKe) in Espressif chips.

To install esptool, run the following in your terminal:

```bash
pip3 install esptool
```

This guide assumes you have Python3 installed. If you have not installed it, navigate to the Python downloads page (https://www.python.org/downloads) and install the latest release.
Burning nina-fw with esptool

Click the link below to download the latest nina-fw .bin file. Unzip it and save the .bin file to your desktop.

[Latest nina-fw .bin file: https://adafru.it/G3D](https://adafru.it/G3D)

If you're using macOS or Linux - run the following command, replacing `/dev/ttyACM0` with the serial port of your board and `NINA_W102-1.6.0` with the binary file you're flashing to the ESP32.

```
esptool.py --port /dev/ttyACM0 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

If you're using Windows - run the following command, replacing `COM7` with the serial port of your board and `NINA_W102-1.6.0` with the binary file you're flashing to the ESP32.

```
esptool.py --port COM7 --before no_reset --baud 115200 write_flash 0 NINA_W102-1.6.0.bin
```

The command should detect the ESP32 and will take a minute or two to upload the firmware.

If ESPTool doesn't detect the ESP32, make sure you've uploaded the correct .UF2 file to the bootloader and are using the correct serial port.
Once the firmware is fully uploaded, press the Reset button (or, on the RP2040 Pico, unplug your device from USB power) to get out of the ROM bootloader.

### Verifying the Upgraded Firmware Version

To verify everything is working correctly, we'll load up some CircuitPython code.

If you were previously using your ESP32 project with CircuitPython, you'll need to first reinstall CircuitPython firmware for your board. The QSPI flash should have retained its contents. If you don't see anything on the CIRCUITPY volume, copy files from the backup you made earlier to CIRCUITPY.

To verify the new ESP32 WiFi firmware version is correct, follow the Connect to WiFi step in this guide (https://adafru.it/Eao) and come back here when you've successfully run the code. The REPL output should display the firmware version you flashed.