Overview

If you want to keep the firmware on your ESP32 WiFi 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.

Why would I update my ESP32's firmware?

Using an ESP32 as a WiFi 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 co-processor with a variant of the Arduino nina-fw core [https://adafru.it/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.

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

OUT OF STOCK

Out Of Stock

Adafruit AirLift FeatherWing – ESP32 WiFi Co-Processor

OUT OF STOCK

Out Of Stock

Adafruit AirLift Shield - ESP32 WiFi Co-Processor

$15.95
IN STOCK
Add To Cart

Your browser does not support the video tag.

Adafruit AirLift Bitsy Add-On – ESP32 WiFi Co-Processor

$14.95
IN STOCK
Add To Cart
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.

Materials

1 x **USB Cable**
USB cable - USB A to Micro-B - 3 foot long
Upgrade All-in-One ESP32 AirLift Firmware

Uploading Serial Passthrough Code

First, back up any code and files you have on your CIRCUITY 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.

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.

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.

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.

Install esptool.py

Esptool is an application which 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
```
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.

https://adafru.it/G3D

If you're using macOS or Linux - run the following command, replacing /dev/ttyS6 with the serial port of your board and NINA_W102-1.3.0 with the binary file you're flashing to the ESP32.

esptool.py --port /dev/ttyS6 --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. 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.
Upgrade External ESP32 Airlift Firmware

Bridging the ESP32's Optional Control Pins

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

- ESPGPIO0
- ESPRX
- ESPTX

Before continuing the steps on this page - you will need to add solder bridges on the ESPTX, EXPRX and GPIO0 pads on the bottom of breakout.

Make sure you solder all three of these pads together. You will not be able to upload firmware to your ESP32 if they are not connected.

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 - use the "Code - Arduino Passthrough" section instead.

Uploading Serial Passthrough Code for Feather M4 or ItsyBitsy M4

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.

https://adafruit.it/IEK
https://adafruit.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.
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.

Code - Arduino Passthrough

With the ESP32's optional control pins soldered together, you'll be turning your Airlift breakout, shield, or wing into a
USB to Serial converter. To do this, you'll need a special Arduino sketch named SerialESPPassthrough.ino.

Click Download: Project ZIP to download the code below.

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.unconfigured32/0
```

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:

```bash
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.

https://adafru.it/G3D

If you're using macOS or Linux - run the following command, replacing /dev/ttyS6 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/ttyS6 --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, 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 may also want 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.

![ScanNetworks serial output]

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:

```python
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']
```