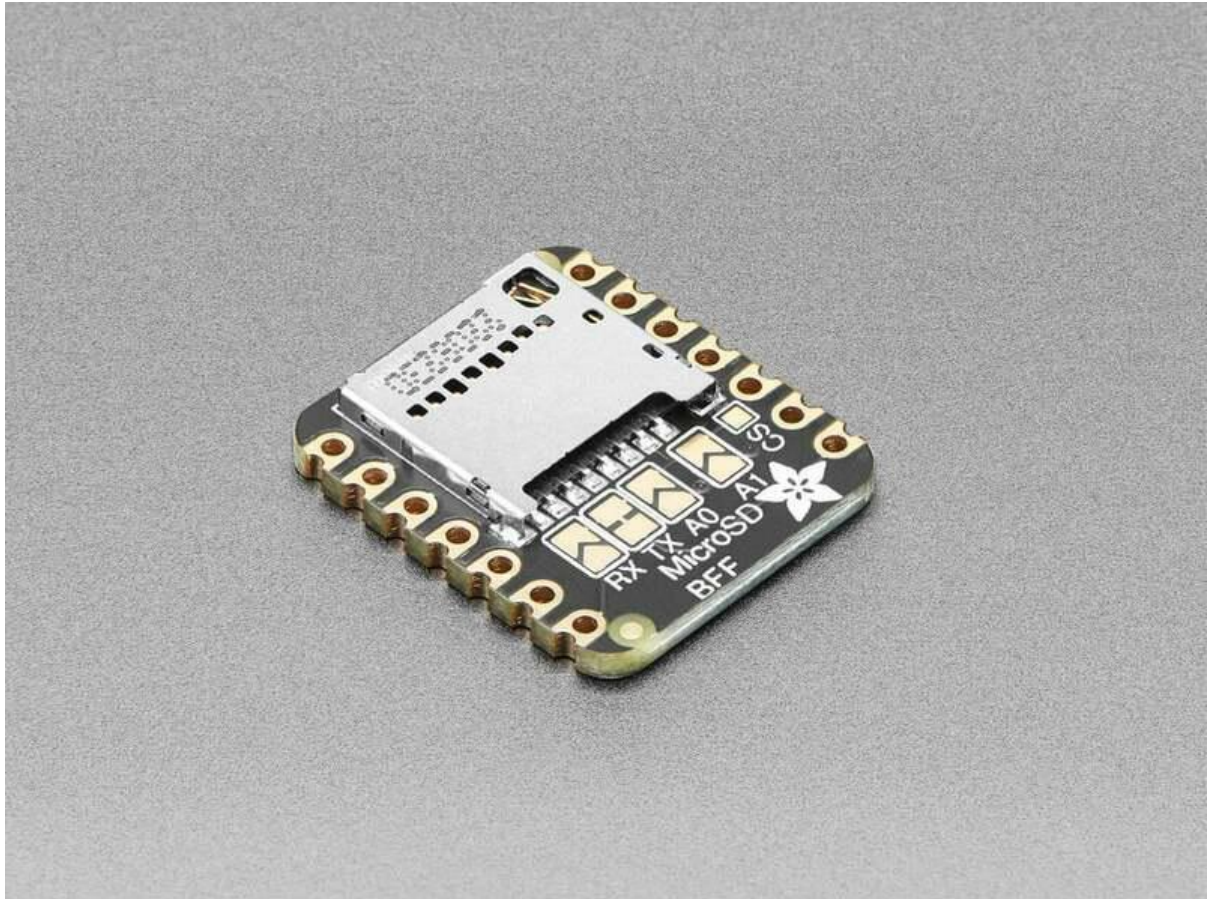




# Adafruit microSD Card BFF

Created by Liz Clark



<https://learn.adafruit.com/adafruit-microsd-card-bff>

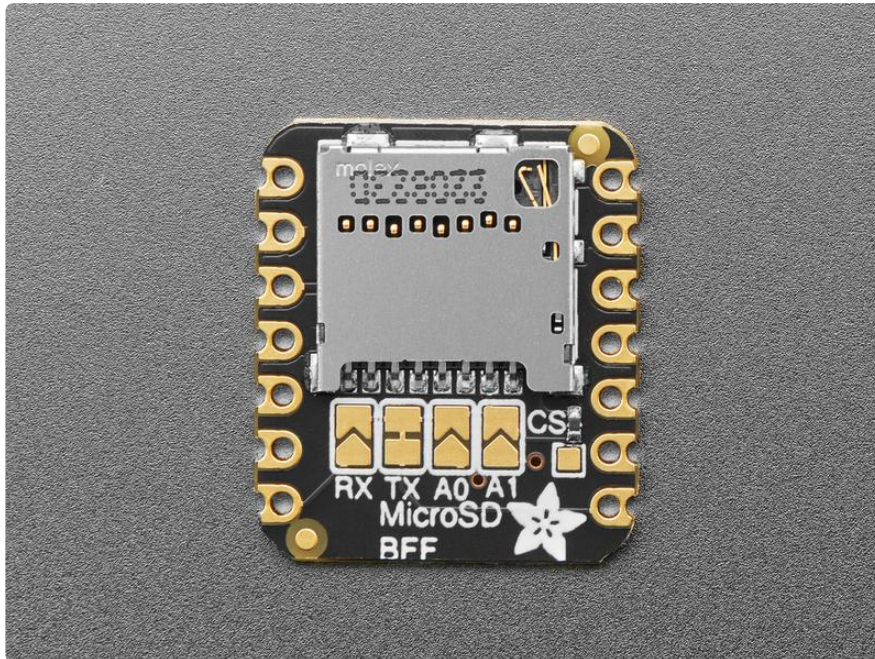
Last updated on 2023-08-29 04:53:52 PM EDT

# Table of Contents

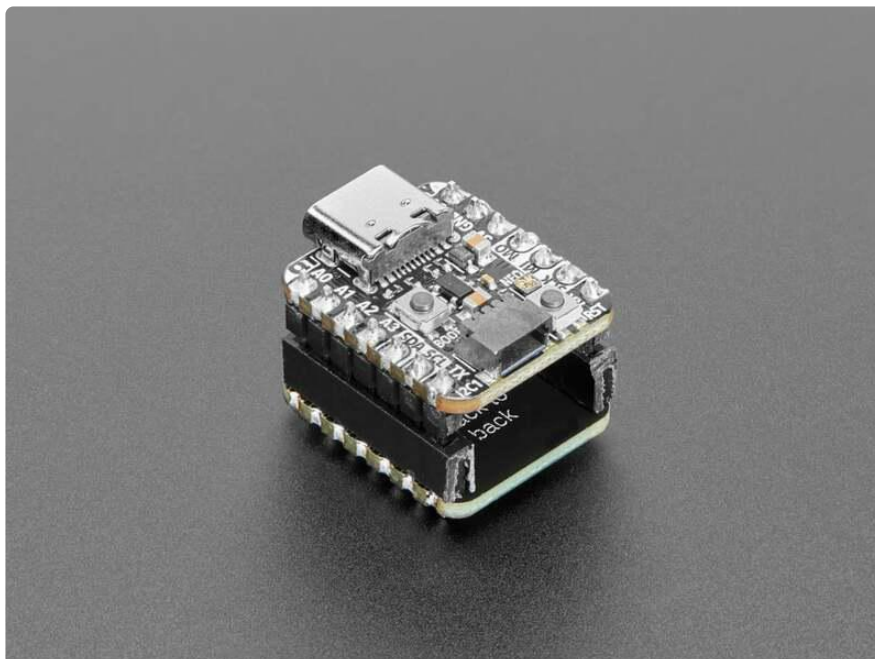
<a href="#">Overview</a>	3
<a href="#">Pinouts</a>	5
<ul style="list-style-type: none"><li>• <a href="#">TX Jumper</a></li><li>• <a href="#">Pin Select Solder-jumpers</a></li><li>• <a href="#">CS Pad</a></li></ul>	
<a href="#">CircuitPython</a>	6
<ul style="list-style-type: none"><li>• <a href="#">CircuitPython Microcontroller Wiring</a></li><li>• <a href="#">CircuitPython Usage</a></li><li>• <a href="#">SD Card Read Test</a></li><li>• <a href="#">SD Card Write Test</a></li></ul>	
<a href="#">Python Docs</a>	9
<a href="#">Arduino</a>	10
<ul style="list-style-type: none"><li>• <a href="#">Wiring</a></li><li>• <a href="#">Library Installation</a></li><li>• <a href="#">Read/Write Example</a></li></ul>	
<a href="#">Arduino Docs</a>	13
<a href="#">Downloads</a>	13
<ul style="list-style-type: none"><li>• <a href="#">Files</a></li><li>• <a href="#">Schematic and Fab Print</a></li></ul>	

---

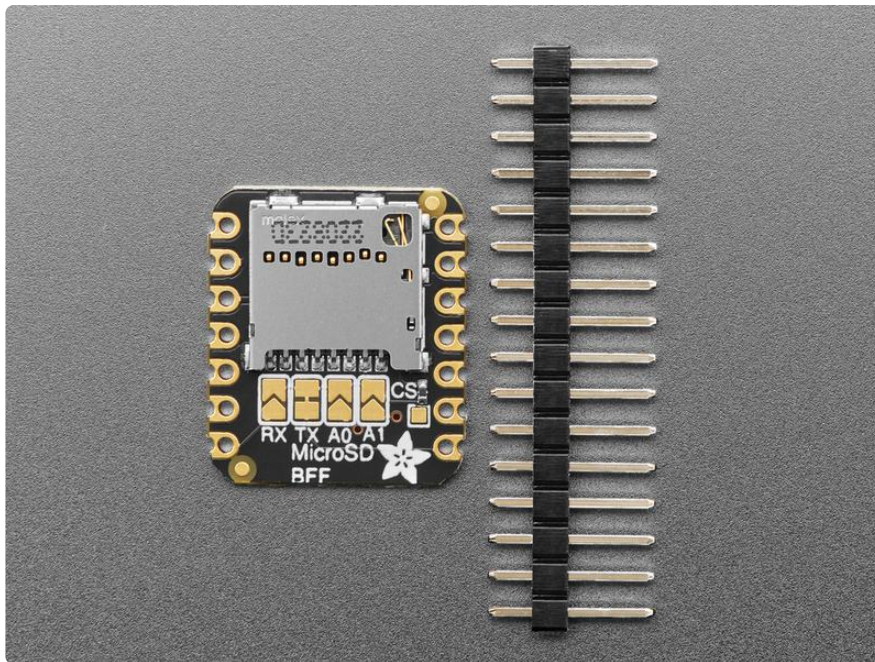
# Overview



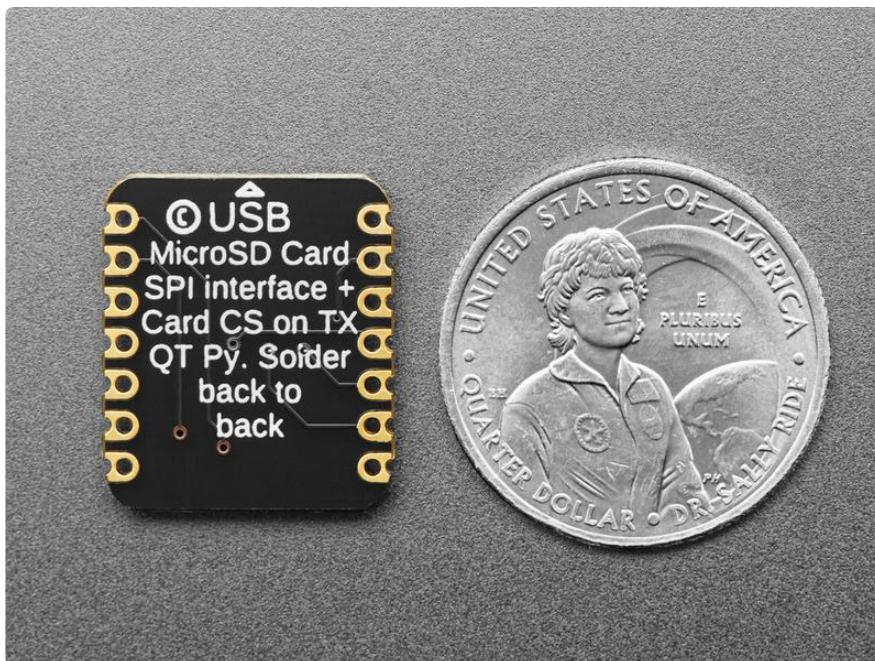
Adafruit QT Py boards are a great way to make very small microcontroller projects that pack a ton of power - and now there is a way for you to add a ton of storage, for reading and writing, with a micro SD card slot that can fit on the back of your miniature dev board. It uses the three SPI pins plus one chip select pin to access megs or gigs of data.



We call this the Adafruit microSD BFF - a "Best Friend Forever". When you were a kid you may have learned about the "buddy" system, well this product is kinda like that! A board that will watch your QT Py's back and give it more capabilities.



This PCB is designed to fit onto the back of any QT Py or Xiao board, it can be soldered into place or use pin and socket headers to make it removable. Onboard is a slim, high quality Molex push-pull micro SD card socket. Since the QT Py is already 3V, no level shifter or regulator is required. We're using SPI mode to interface, use the M OSI/MISO/SCK pins plus one other chip select pin. The default is TX but there are some solder jumpers you can use to select RX, A0 or A1.



We include some header that you can solder to your QT Py. [You can also pick up an Itsy Bitsy short female header kit to make it removable but compact \(\)](#), you'll just need to trim down the headers to 7 pins long.

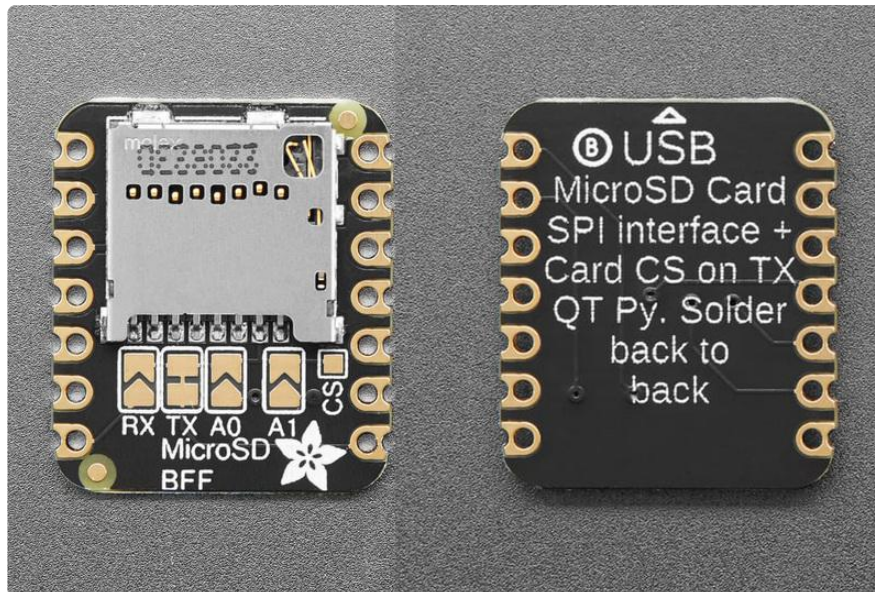
- Comes as an assembled and tested PCB

- For any QT Py or Xiao boards
- Use any micro SD card that supports SPI mode with one CS pin.
- Any Arduino / CircuitPython / MicroPython library can be used to talk to the card just like a normal SD card breakout!

microSD card and QT Py are not included.

---

## Pinouts



The default chip select pin (CS) is TX.

## TX Jumper

- TX jumper - This jumper is located on the front of the board, towards the center directly above the MicroSD board text and is labeled TX. If cut, the chip select pin (CS) for the SD card is no longer connected to TX and you can solder one of the solder-jumpers closed to change the CS pin.

One of the pins must be used as CS for the microSD card to work

## Pin Select Solder-jumpers

- Three solder-jumpers are available to change the CS pin for the microSD BFF. They are located on the front of the board, directly below the microSD card slot.

Their corresponding pin names can be found directly below the jumpers. You can choose to solder one of the solder-jumpers for one of the following pins:

- RX
- A0
- A1

## CS Pad

- CS - The CS pad is a single square pad, located on the right side of the board between the microSD slot and the Adafruit logo on the silk. You can use this pad to connect the chip select pin to another digital pin.

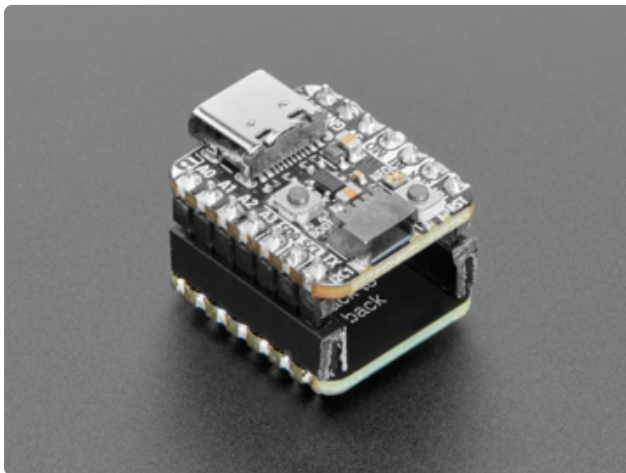
---

## CircuitPython

It's easy to use the microSD Card BFF with CircuitPython and the [Adafruit\\_CircuitPython\\_SD \(\)](#) module. This module allows you to easily write Python code that lets you read and write to an attached SD card.

## CircuitPython Microcontroller Wiring

Plug a microSD Card BFF into your QT Py or Xiao form factor board exactly as shown below. Here's an example of connecting a QT Py RP2040 to the BFF.



Connect the QT Py RP2040 with plug headers into the microSD Card BFF with socket headers. They should be plugged in with the backs of the boards facing each other.

For more information on soldering socket headers, [check out this Learn Guide \(\)](#).

[How to Solder Headers Learn Guide](#)

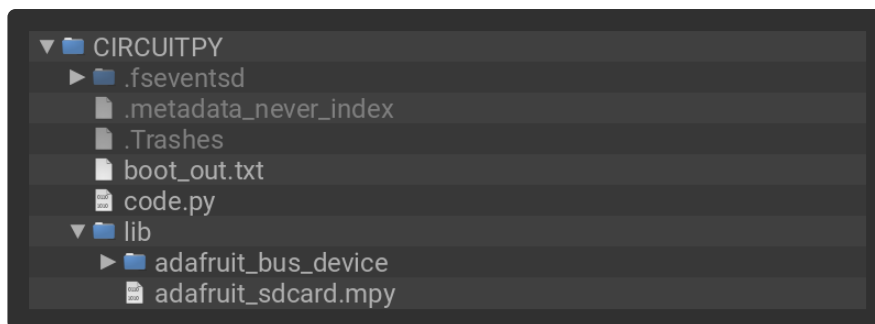
## CircuitPython Usage

To use with CircuitPython, you need to first install the `adafruit_sdcard` library, and its dependencies, into the `lib` folder on your CIRCUITPY drive. Then you need to update `code.py` with the example script.

Thankfully, we can do this in one go. In the example below, click the [Download Project Bundle](#) button below to download the necessary libraries and the `code.py` file in a zip file. Extract the contents of the zip file, and copy the entire `lib` folder and the `code.py` file to your CIRCUITPY drive.

Your CIRCUITPY/`lib` folder should contain the following folders and files:

- `/adafruit_bus_device`
- `adafruit_sdcard.mpy`



## SD Card Read Test

Once everything is saved to the CIRCUITPY drive, [connect to the serial console \(\)](#) to see the data printed out!

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import os
import busio
import digitalio
import board
import storage
import adafruit_sdcard

# The SD_CS pin is the chip select line.

SD_CS = board.TX # setup for microSD BFF

# Connect to the card and mount the filesystem.
spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
cs = digitalio.DigitalInOut(SD_CS)
sdcard = adafruit_sdcard.SDCard(spi, cs)
vfs = storage.VfsFat(sdcard)
```

```

storage.mount(vfs, "/sd")

# Use the filesystem as normal! Our files are under /sd

# This helper function will print the contents of the SD
def print_directory(path, tabs=0):
    for file in os.listdir(path):
        stats = os.stat(path + "/" + file)
        filesize = stats[6]
        isdir = stats[0] & 0x4000

        if filesize < 1000:
            sizestr = str(filesize) + " bytes"
        elif filesize < 1000000:
            sizestr = "%0.1f KB" % (filesize / 1000)
        else:
            sizestr = "%0.1f MB" % (filesize / 1000000)

        prettyprintname = ""
        for _ in range(tabs):
            prettyprintname += "  "
        prettyprintname += file
        if isdir:
            prettyprintname += "/"
        print("{0:<40} Size: {1:>10}".format(prettyprintname, sizestr))

    # recursively print directory contents
    if isdir:
        print_directory(path + "/" + file, tabs + 1)

print("Files on filesystem:")
print("=====")
print_directory("/sd")

```

```

CircuitPython REPL
Auto-reload is on. Simply save files over USB to run them or enter REPL to disable.
code.py output:
Files on filesystem:
=====
System Volume Information/           Size:    0 bytes
  WPSettings.dat                     Size:   12 bytes
  IndexerVolumeGuid                  Size:   76 bytes
icons/                                Size:    0 bytes
  01d.bmp                             Size: 460.9 KB
  01n.bmp                             Size: 460.9 KB
  02d.bmp                             Size: 460.9 KB
  02n.bmp                             Size: 460.9 KB
  03d.bmp                             Size: 460.9 KB
  03n.bmp                             Size: 460.9 KB
  04d.bmp                             Size: 460.9 KB
  04n.bmp                             Size: 460.9 KB
  09d.bmp                             Size: 460.9 KB
  09n.bmp                             Size: 460.9 KB
  10d.bmp                             Size: 460.9 KB
  10n.bmp                             Size: 460.9 KB
  11d.bmp                             Size: 460.9 KB

```

In this read test for the SD card, the filesystem on the SD card is mounted and read. Then, the contents of the filesystem are printed to the REPL.

## SD Card Write Test

```

# SPDX-FileCopyrightText: 2017 Limor Fried for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time

```



```

import adafruit_sdcard
import board
import busio
import digitalio
import microcontroller
import storage

# default CS pin for the microSD bff is TX
SD_CS = board.TX

# Connect to the card and mount the filesystem.
spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
cs = digitalio.DigitalInOut(SD_CS)
sdcard = adafruit_sdcard.SDCard(spi, cs)
vfs = storage.VfsFat(sdcard)
storage.mount(vfs, "/sd")

# Use the filesystem as normal! Our files are under /sd

print("Logging temperature to filesystem")
# append to the file!
while True:
    # open file for append
    with open("/sd/temperature.txt", "a") as f:
        t = microcontroller.cpu.temperature
        print("Temperature = %0.1f" % t)
        f.write("%0.1f\n" % t)
    # file is saved
    time.sleep(1)

```

The image shows two side-by-side windows. The left window is a terminal titled 'code.py output:' displaying the following text:

```

Logging temperature to filesystem
Temperature = 33.2
Temperature = 35.6
Temperature = 35.6
Temperature = 35.1
Temperature = 35.6
Temperature = 36.0
Temperature = 35.6
Temperature = 35.6
Temperature = 35.6
Temperature = 36.0
Temperature = 36.5
Temperature = 34.6
Temperature = 34.6
Temperature = 36.0
Temperature = 36.0

```

The right window is a Notepad application titled 'temperature.txt - Notepad'. It shows the following text:

```

33.2
35.6
35.6
35.1
35.6
36.0
35.6
35.6
35.6
36.0
36.5
34.6
34.6
36.0
36.0

```

In this example, the code is writing data to the SD card. The microcontroller CPU temperature is printed to the REPL and logged to a text file on the SD card. After running the code, you can read the text file from the SD card. You'll see that the text file matches what was printed to the REPL.

## Python Docs

[Python Docs \(\)](#)

---

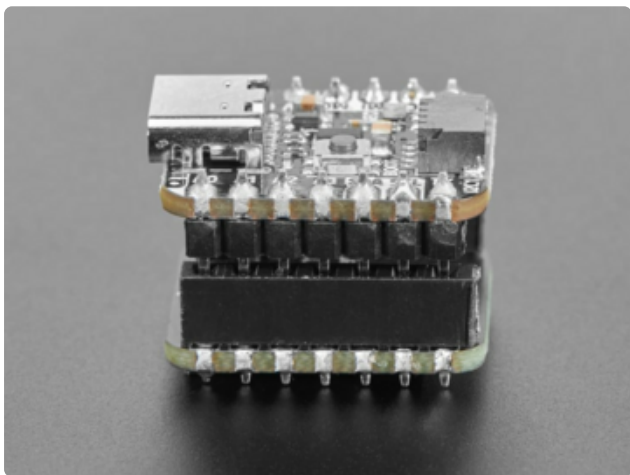
# Arduino

Using the microSD Card BFF with Arduino involves plugging the breakout into your Arduino-compatible QT Py or Xiao form factor board, installing the [Adafruit Fork of the SdFat library](#) (), and running the provided example code.

This code has been tested and confirmed working with a QT Py ESP32-S2. Issues may arise with a QT Py RP2040.

## Wiring

Plug a microSD Card BFF into your QT Py or Xiao form factor board exactly as shown below. Here's an example of connecting a QT Py ESP32-S2 to the BFF.



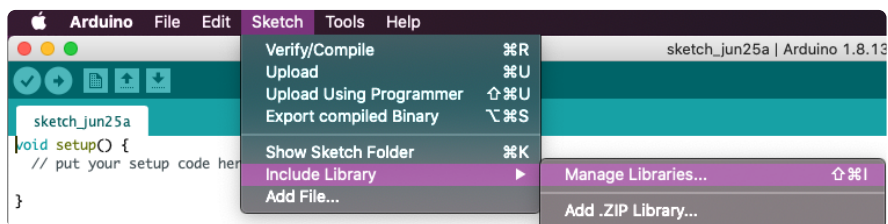
Connect the QT Py ESP32-S2 with pin headers into the microSD Card BFF with socket headers. They should be plugged in with the backs of the boards facing each other.

For more information on soldering socket headers, [check out this Learn Guide](#) ().

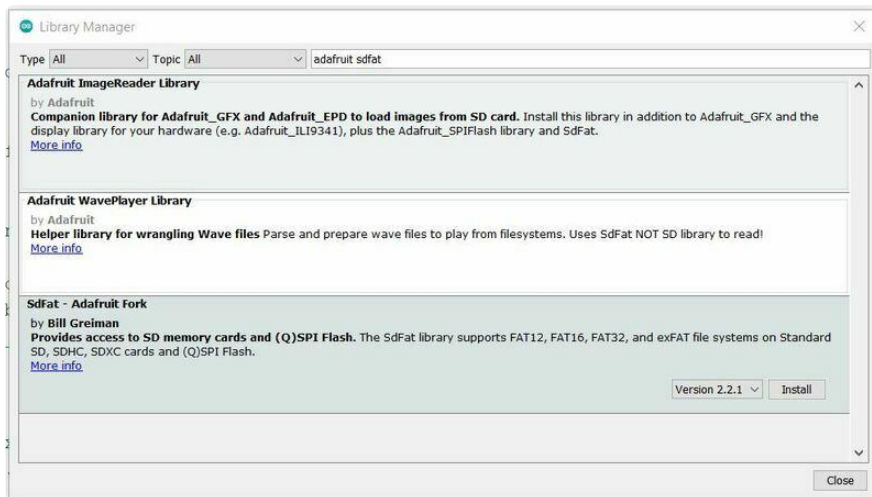
[How to Solder Headers Learn Guide](#)

## Library Installation

You can install the Adafruit Fork of the SDFat library for Arduino using the Library Manager in the Arduino IDE.



Click the Manage Libraries ... menu item, search for Adafruit SDFat and select the SDF at - Adafruit Fork library:



There are no additional dependencies for the SdFat - Adafruit Fork library.

## Read/Write Example

```
// SPDX-FileCopyrightText: 2023 Liz Clark for Adafruit Industries
//
// SPDX-License-Identifier: MIT
/*
  SD card read/write

  This example shows how to read and write data to and from an SD card file
  The circuit:
  * SD card attached to SPI bus as follows:
  ** MOSI - pin 11
  ** MISO - pin 12
  ** CLK - pin 13

  created   Nov 2010
  by David A. Mellis
  modified  9 Apr 2012
  by Tom Igoe
  modified 14 Feb 2023
  by Liz Clark

  This example code is in the public domain.

  */

#include <SPI.h>
// #include <SD.h>
#include "SdFat.h"
SdFat SD;

#define SD_FAT_TYPE 3

// default CS pin is TX for microSD BFF
#define SD_CS_PIN TX

File myFile;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(115200);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
}
```

```

}

Serial.print("Initializing SD card...");

if (!SD.begin(SD_CS_PIN)) {
  Serial.println("initialization failed!");
  return;
}
Serial.println("initialization done.");

// open the file. note that only one file can be open at a time,
// so you have to close this one before opening another.
myFile = SD.open("test.txt", FILE_WRITE);

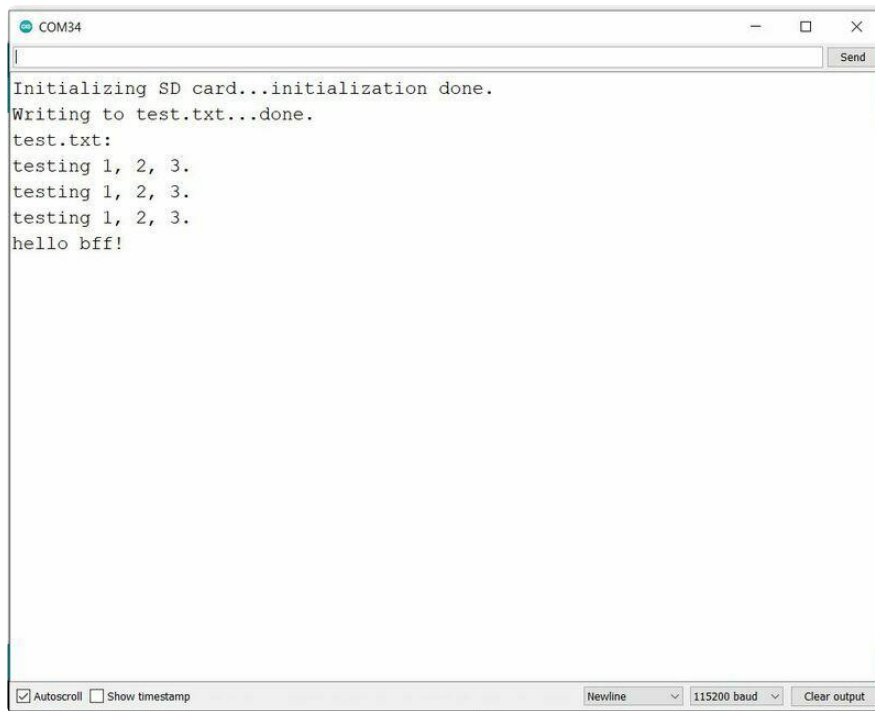
// if the file opened okay, write to it:
if (myFile) {
  Serial.print("Writing to test.txt...");
  myFile.println("testing 1, 2, 3.");
  myFile.println("hello bff!");
  // close the file:
  myFile.close();
  Serial.println("done.");
} else {
  // if the file didn't open, print an error:
  Serial.println("error opening test.txt");
}

// re-open the file for reading:
myFile = SD.open("test.txt");
if (myFile) {
  Serial.println("test.txt:");

  // read from the file until there's nothing else in it:
  while (myFile.available()) {
    Serial.write(myFile.read());
  }
  // close the file:
  myFile.close();
} else {
  // if the file didn't open, print an error:
  Serial.println("error opening test.txt");
}
}

void loop() {
  // nothing happens after setup
}

```



```
COM34
Initializing SD card...initialization done.
Writing to test.txt...done.
test.txt:
testing 1, 2, 3.
testing 1, 2, 3.
testing 1, 2, 3.
hello bff!
```

Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. A test text file will be created and written to on the SD card. Then, the text file will be read back with its contents printed to the Serial Monitor.

---

## Arduino Docs

[Arduino Docs \(\)](#)

---

## Downloads

### Files

- [EagleCAD PCB files on GitHub \(\)](#)
- [Fritzing object in the Adafruit Fritzing Library \(\)](#)

# Schematic and Fab Print

