Adafruit Music Maker FeatherWing

Created by lady ada
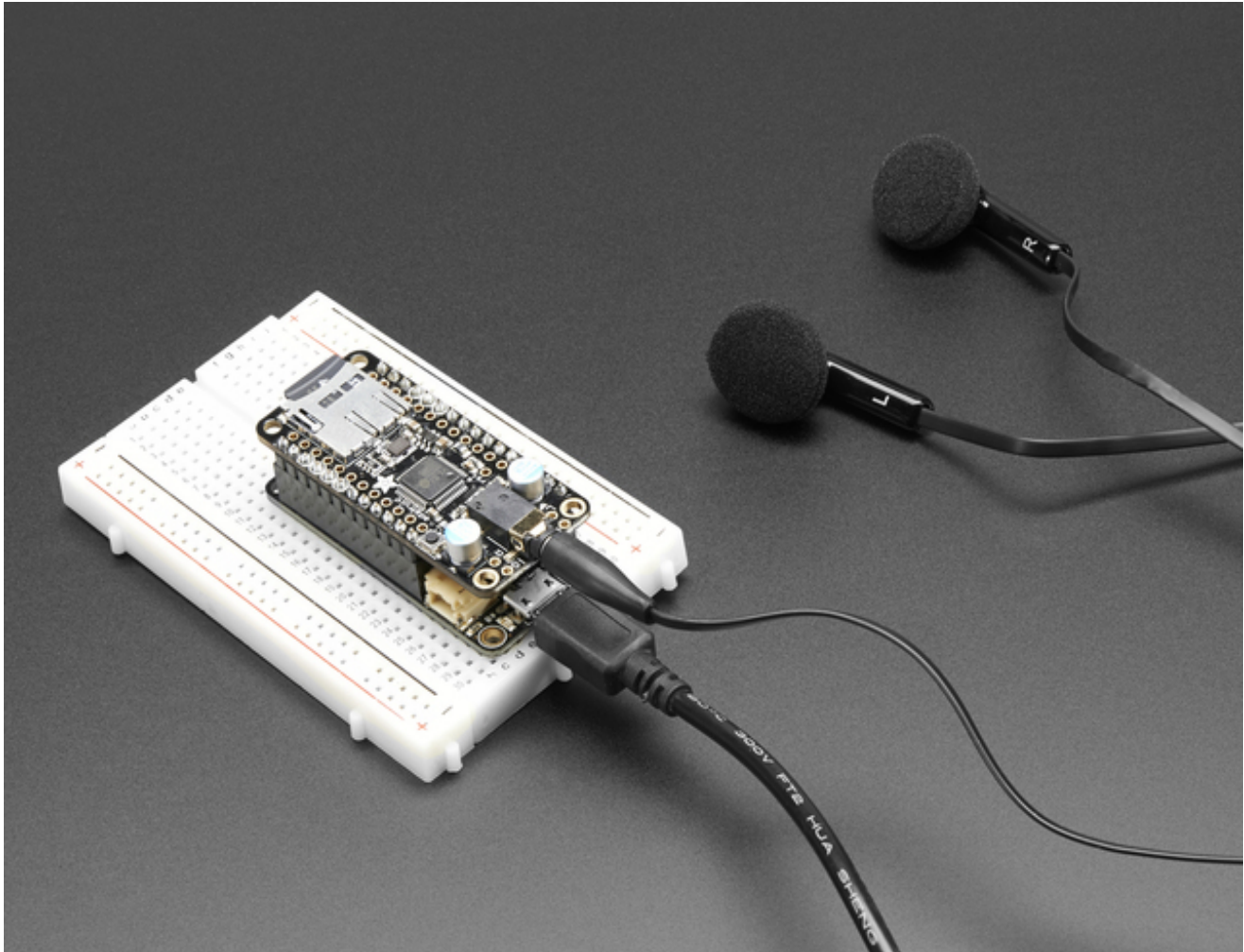
Guide Contents

Overview

Bend all audio files to your will with the**Adafruit Music Maker FeatherWing**! It's a fun-size version of our [Music Maker shield for Arduino](http://adafru.it/tbM) (http://adafru.it/tbM)! This powerful shield features the VS1053, an encoding/decoding (codec) chip that can decode a wide variety of audio formats such as MP3, AAC, Ogg Vorbis, WMA, MIDI, FLAC, WAV (PCM and ADPCM). You can do all sorts of stuff with the audio as well such as adjusting bass, treble, and volume digitally.
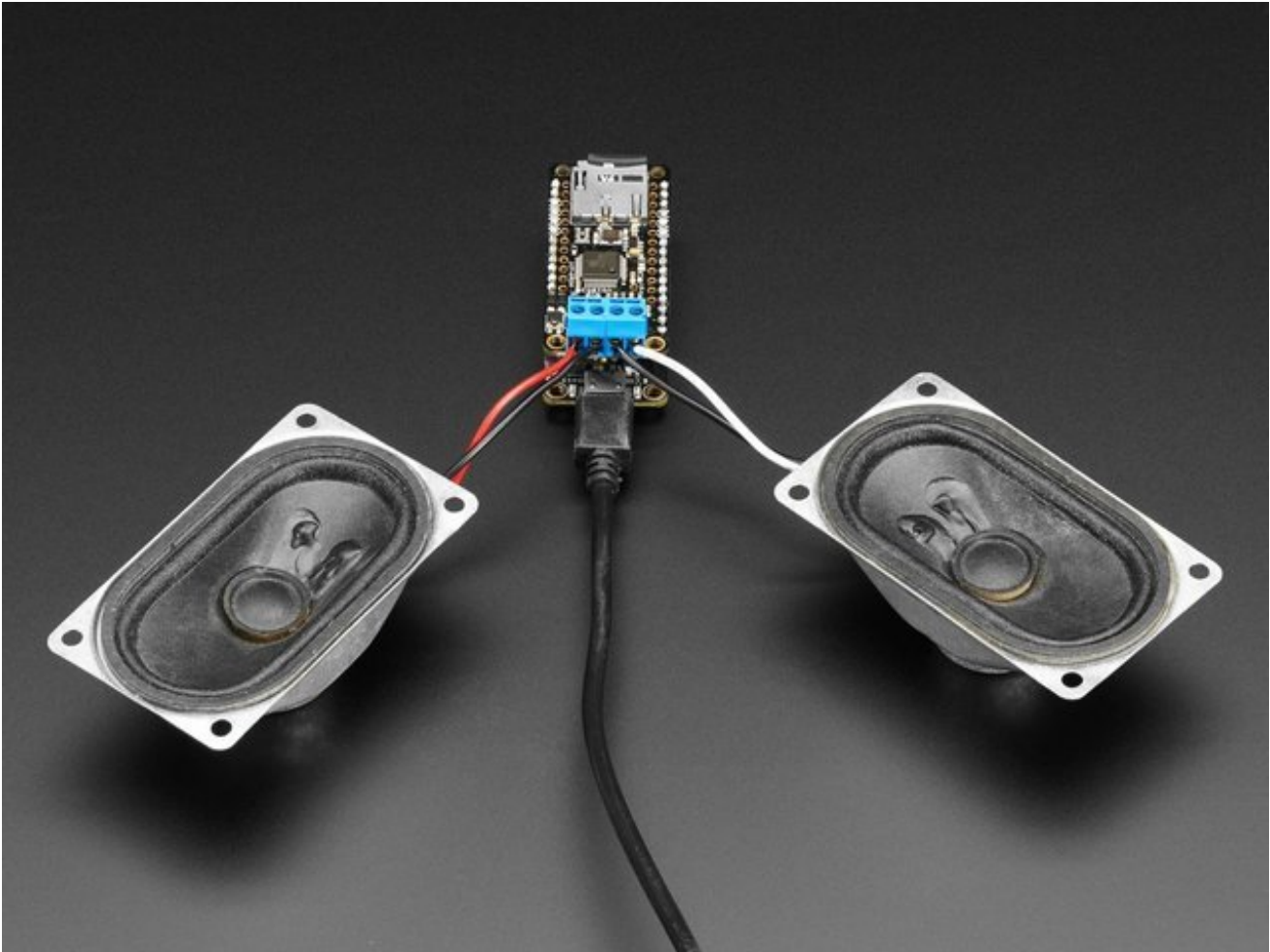
All this functionality is implemented in a light-weight SPI interface so that any Feather Board can play audio from an SD card. There's also a special MIDI mode that you can boot the chip into that will read 'classic' 31250Kbaud MIDI data from the UART TX pin and act like a synth/drum machine - there are dozens of built-in drum and sample effects!
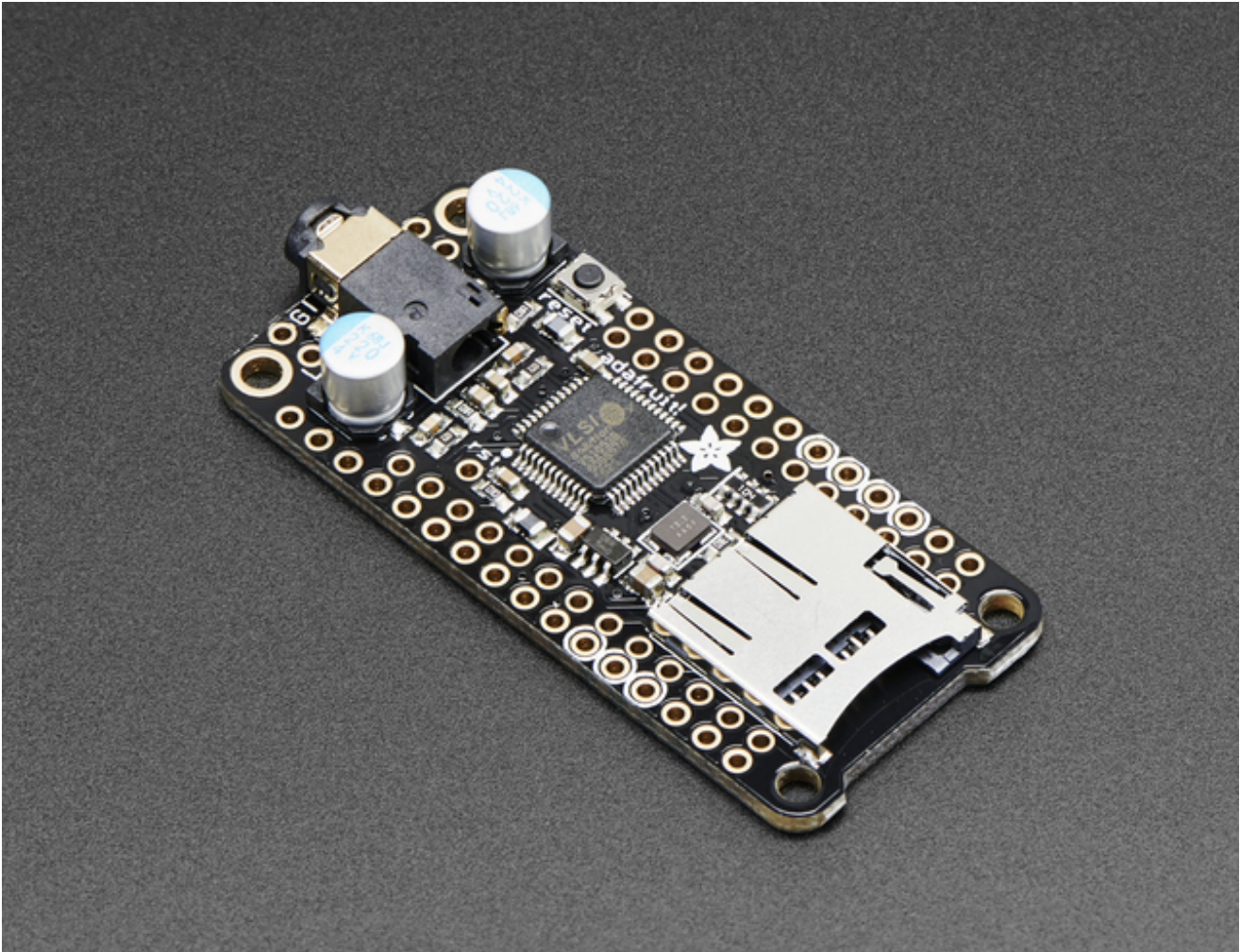
What a great musical add-on to your Feather! That's why we spun up this super FeatherWing, perfect for use with any of our Feather Boards! **Music (MP3) playback testing working with all Feathers**. MIDI playback works with all Feathers but you may need to remove the 'Wing during programming and also for USB-Serial Feathers like the ESP8266 and nRF52 you'll end up re-using the main Serial console for MIDI which can be confusing.

We even made two versions. One version has headphone jack output, so you can listen directly or connect to an audio amp. The other version has a built in stereo amplifier that can driver 4-8 ohm speakers, up to 3W per channel!
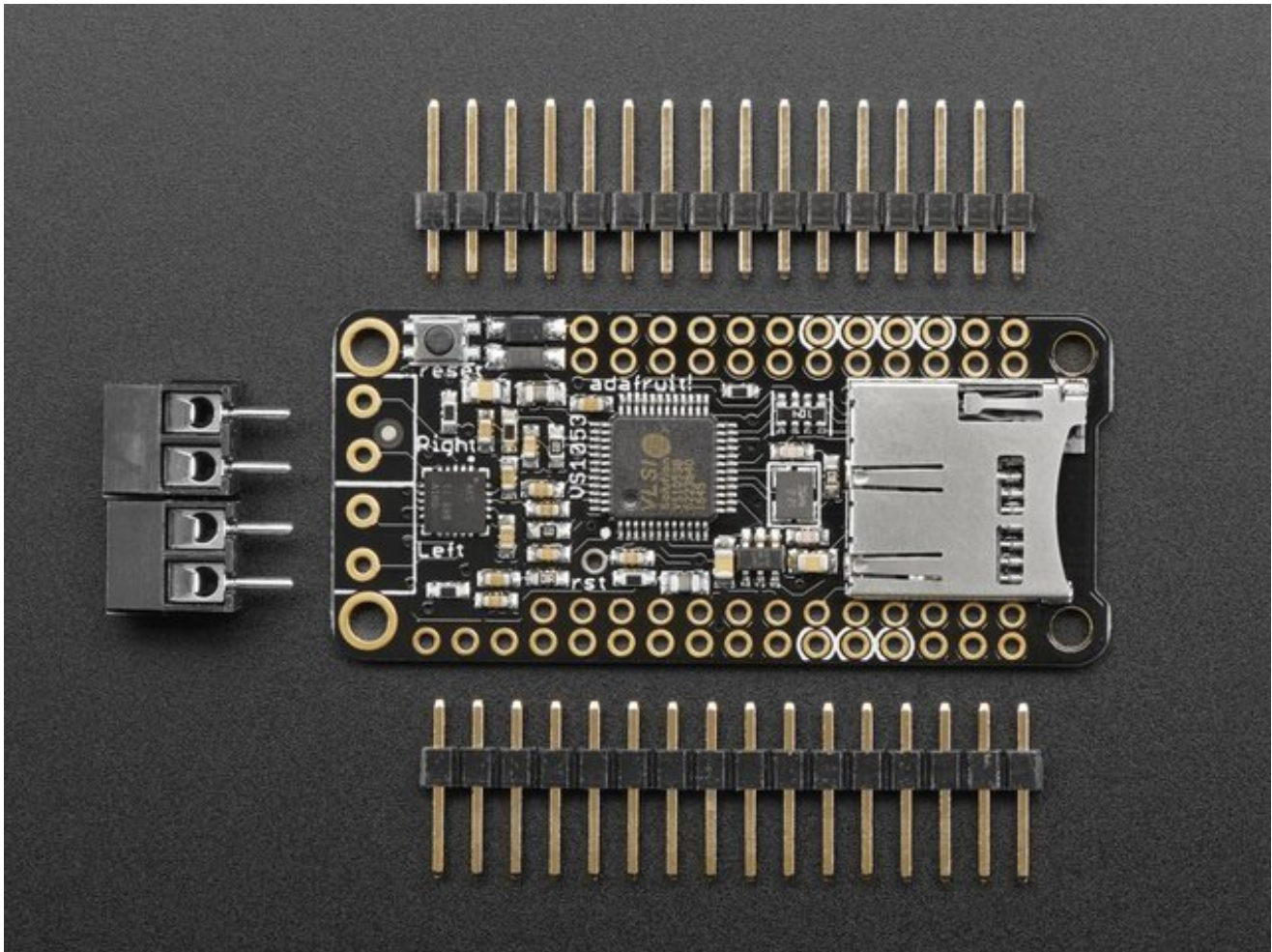
Here are some specs:

- Features the VS1053B codec chip - decodes Ogg Vorbis, MP3/MP2/MP1, MP4, AAC, WMA, FLAC, WAV/PCM, MIDI.
- Stereo audio out with proper audio filter caps and ground reference so it can be safely connected directly to headphones, a stereo system or other powered speakers
- MicroSD card socket, for any FAT16/FAT32 formatted SD card from 64MB or greater
- Built in MIDI synth/drum machine with dozens of instruments
- Headphone 3.5mm stereo line level or stereo audio out to any 4ohm speakers (3W max) or 8 ohm speakers (1W max), class D output
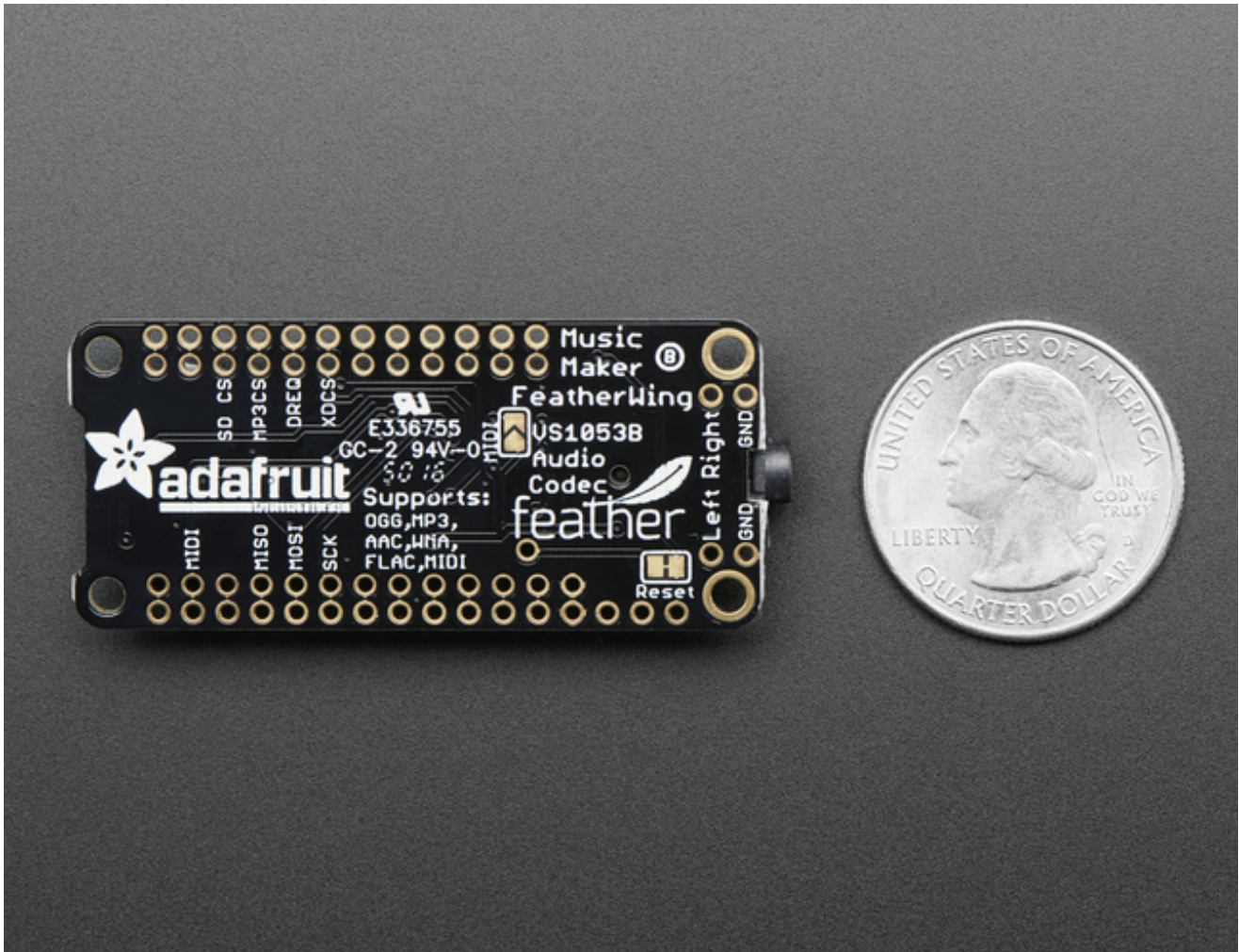
Each order comes with one tested FeatherWing, and two sticks of 0.1" male headers to plug it into your Feather and terminal blocks if you have the Amplifier version. Some light soldering is required to attach the through-hole headers to the PCB for plugging into your Feather Board. **Headphones, speakers, SD card and Feather Board not included!**
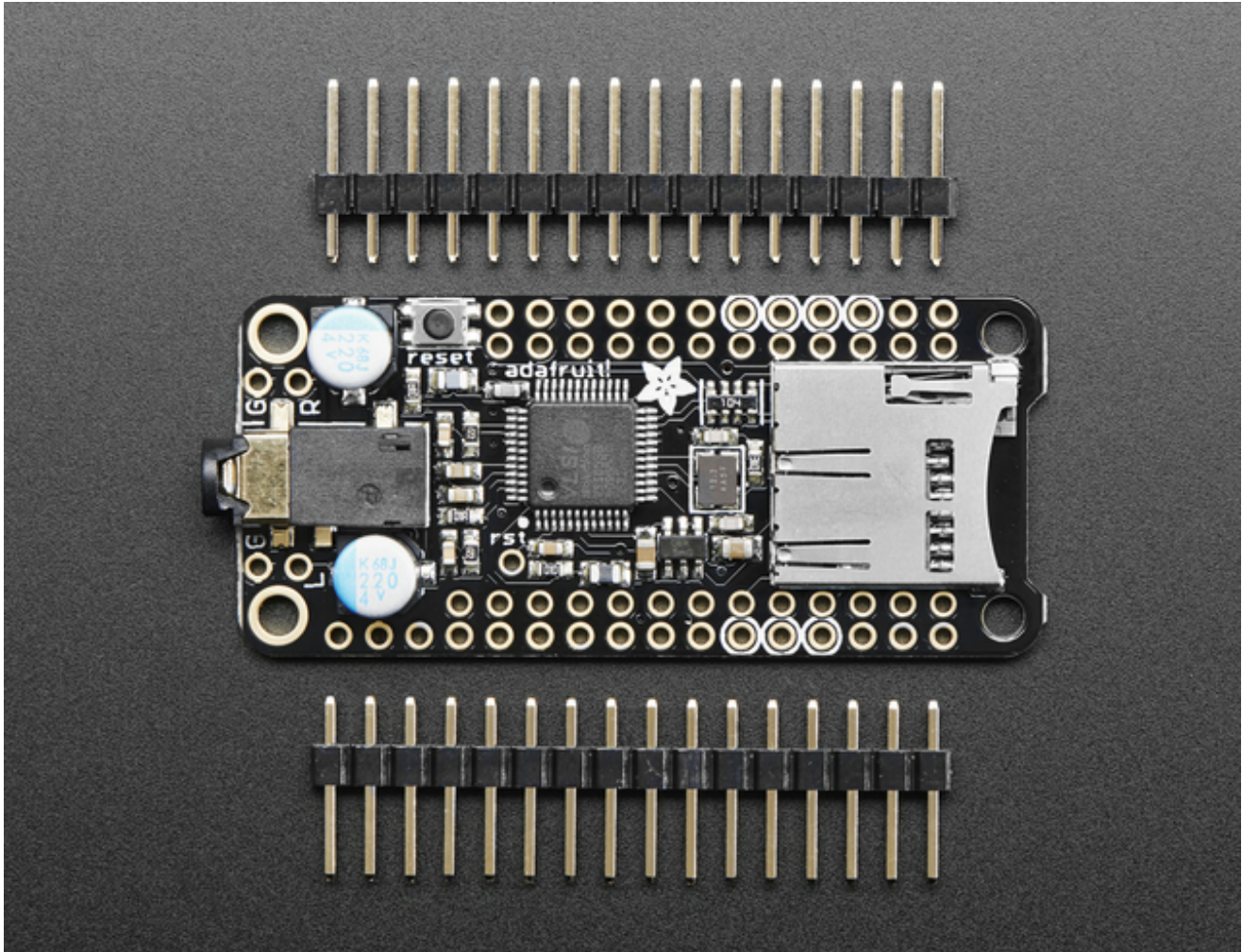
Pinouts
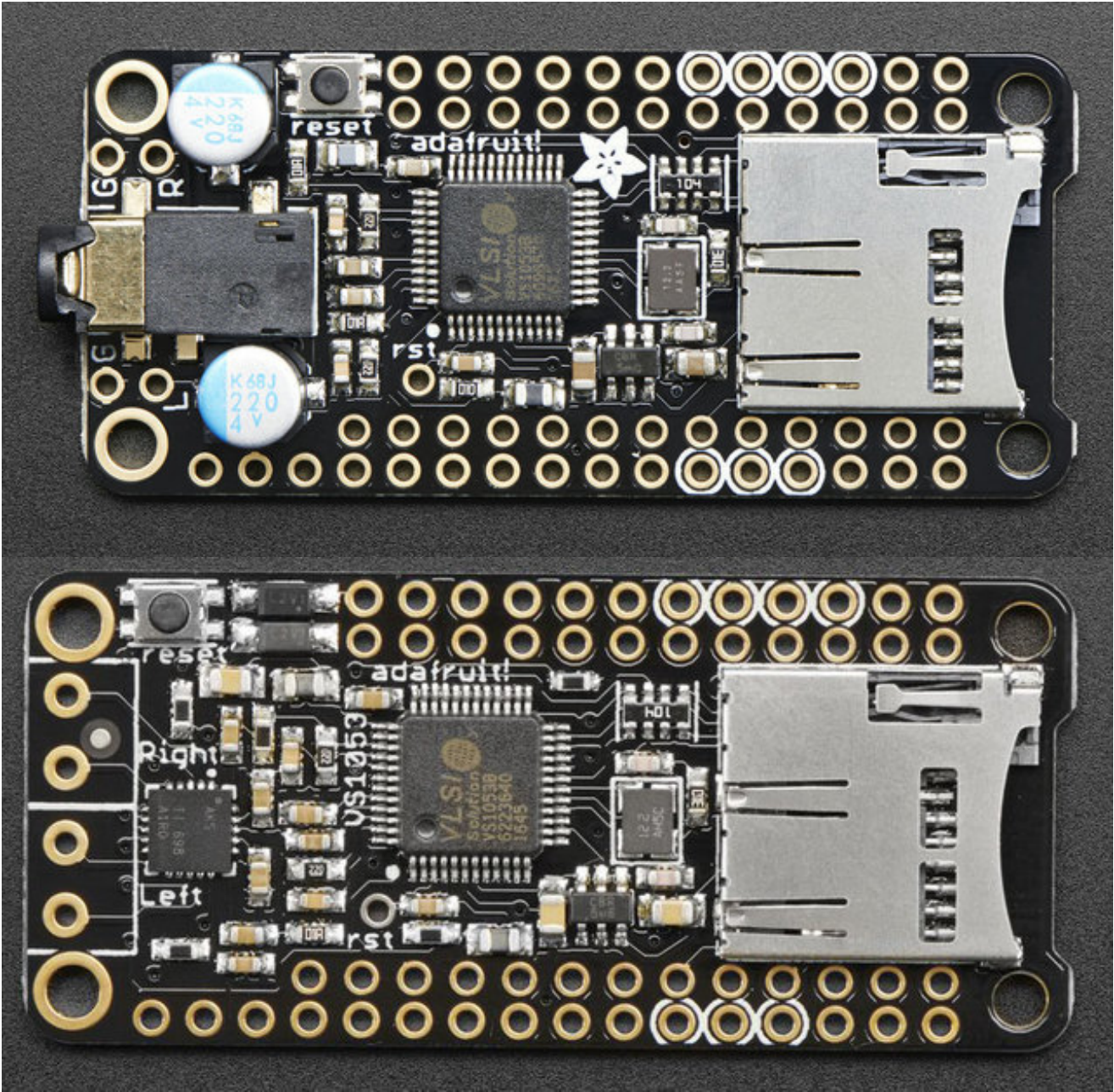
There's a lot of stuff going on in this wing! Lets look at the board and all the pinouts.

Note that overall pin usage and software is the same between both headphone and amp versions.

SPI Pins
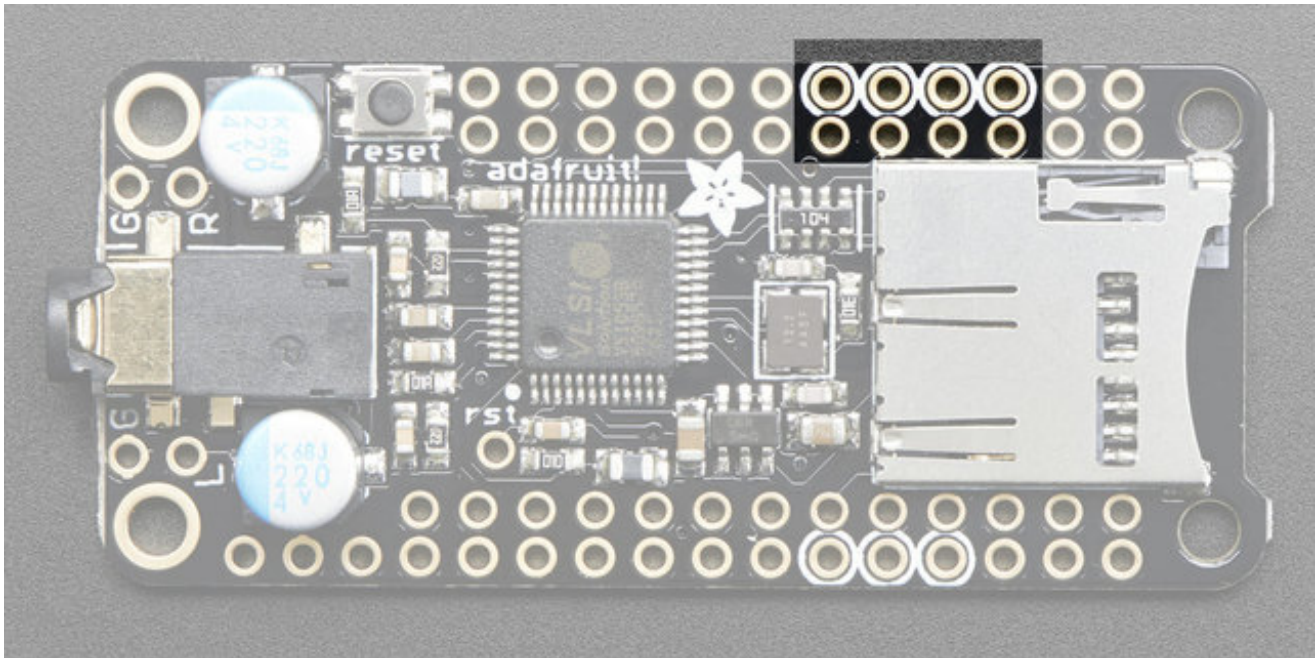
The Music Maker wing has a bunch of pins required for use.

There are three 'totally fixed' pins, the hardware SPI pins:

- **SPI SCK** - connected to **SCK** - used by both the SD card and VS1053
- **SPI MISO** - connected to **MISO** - used by both the SD card and VS1053
- **SPI MOSI** - connected to **MOSI** - used by both the SD card and VS1053

These are needed whenever you are transmitting data from SD card to the decoder chip. If you are using the wing in the special MIDI mode, they're not used.

Control Pins
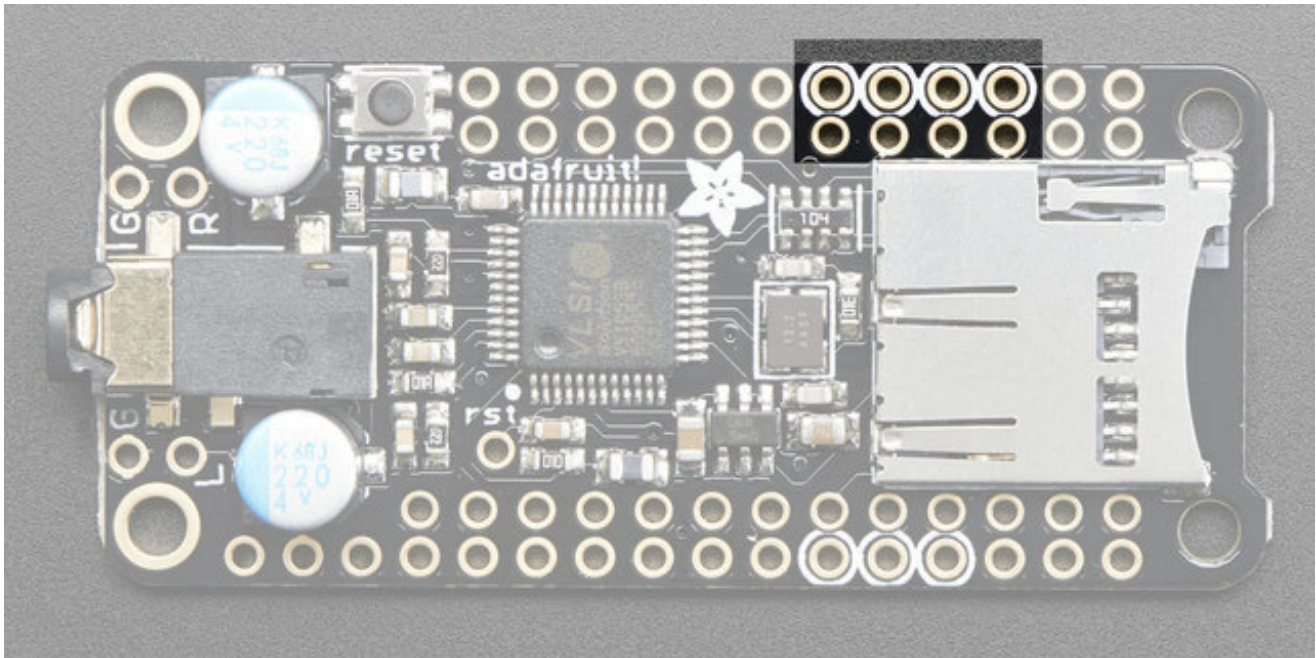
There are a couple other pins that are required for talking to the SD and VS1053 to play MP3s and such. From left to right, above, they are:

- **XDCS** - this is the VS1053 data select pin
- **DREQ** - this is the VS1053 data request interrupt pin
- **MP3CS** - this is the VS1053 chip select pin
- **SD CS** - this is the SD Card chip select pin

These are needed whenever you are transmitting data from SD card to the decoder chip. If you are using the wing in the special MIDI mode, they're not used.

Power Pins
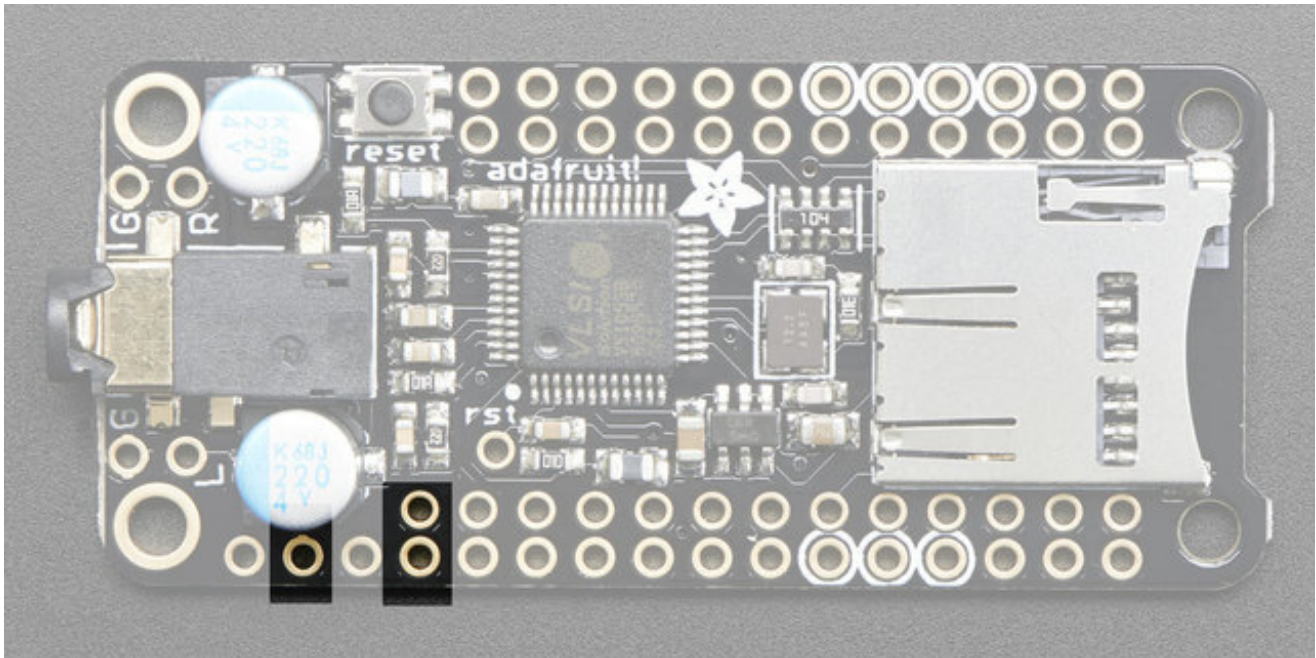
For powering the SD card and MP3 chip, we need power from the **3V** (2nd pin) and **GND** (4th pair of pins).

MIDI Data Pin
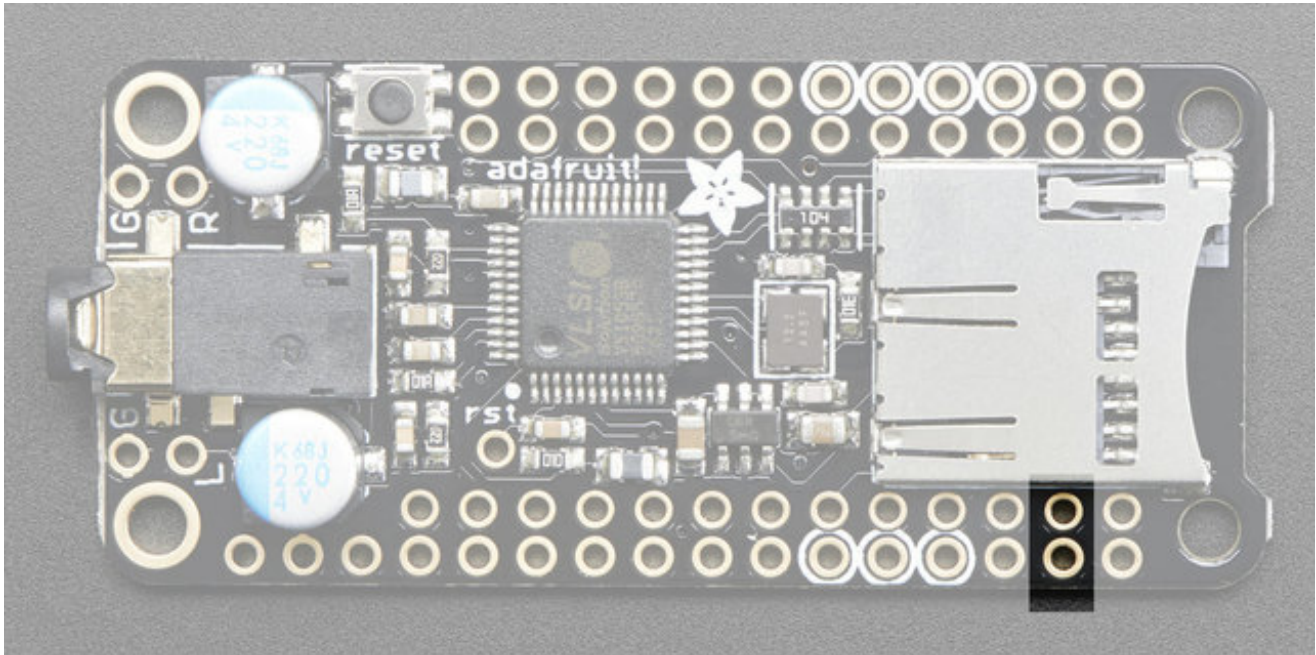
This is the UART data pin that is connected to the MIDI RX on the VS1053. If you short the jumper on the back to put it in MIDI mode, you can send data from the **TX** pin.

This pin is always connected.

Line Level Audio Breakouts (Headphone out version)
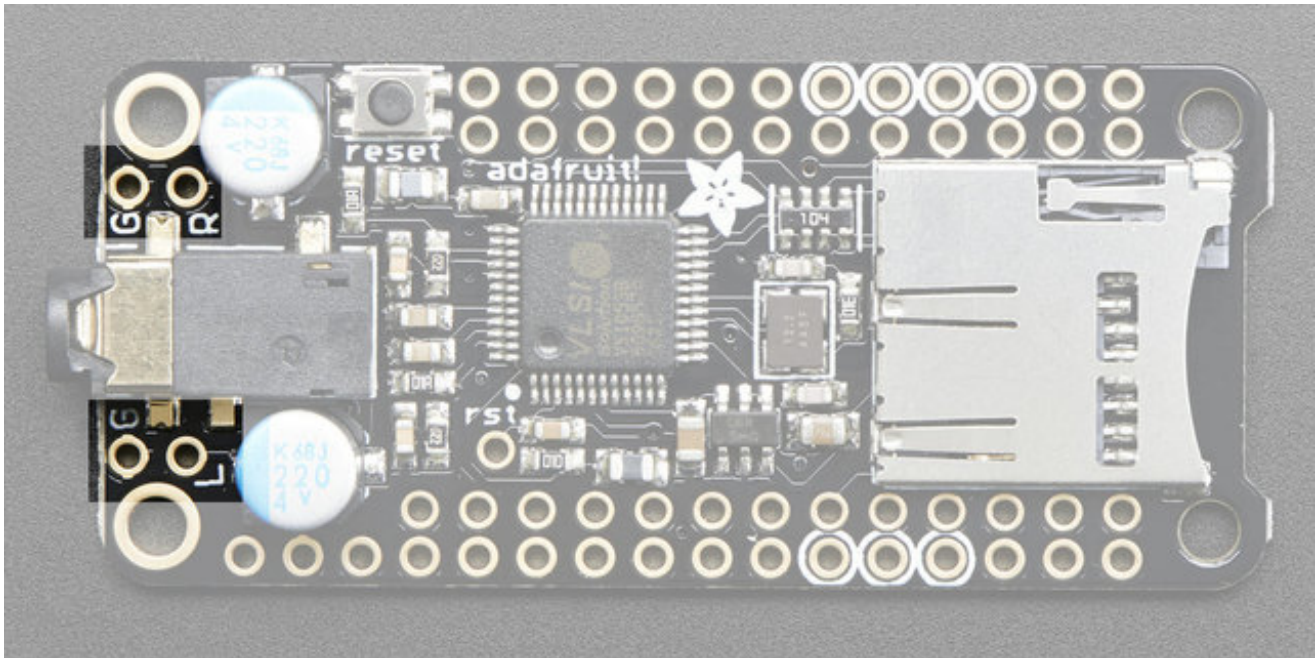
If you don't want to use a 3.5mm jack, you can always connect directly to the **Ground, Right** and **Left** pins next to the headphone jack. They are line level, AC coupled outputs - so its safe to connect them to an amplifier or other device input.

Speaker outputs (Amplifier version)
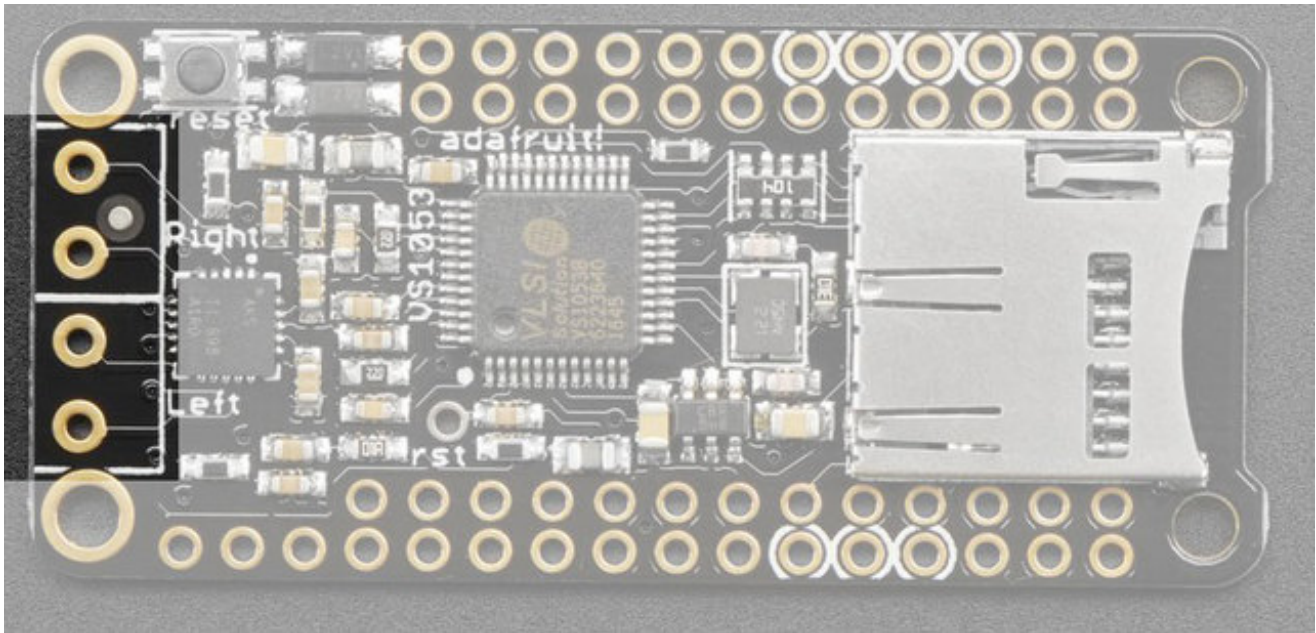
This is the bridge-tied-load output from the VS1053 through the amplifier chip. Connect to 4 to 8 ohm speakers. If you don't need both outputs, just keep one disconnected. Don't connect the outputs together! Nor can you connect the outputs to another amplifier - they are for direct connection to a speaker only!

Once you've soldered the terminal blocks in you can connect any speaker you like, such as our 4Ω 3W (http://adafru.it/1314) or 8Ω 1W (http://adafru.it/1313). Hey you can even use a surface transducer (http://adafru.it/1785)! Don't forget to make sure you have a good strong 5V power supply - especially if you're using the 3W 4 ohm speakers! Our 2.4A power plug is recommended (http://adafru.it/1995).

Amplifier Gain

The default gain is 2 Volts/Volts (6 dB). You can change the decoding gain/volume through the software library. On the off chance if you absolutey have to change the gain, you can short the jumpers on the bottom of the Wing. This is really really not recommended because you can easily blow out your speakers or damage the amp!
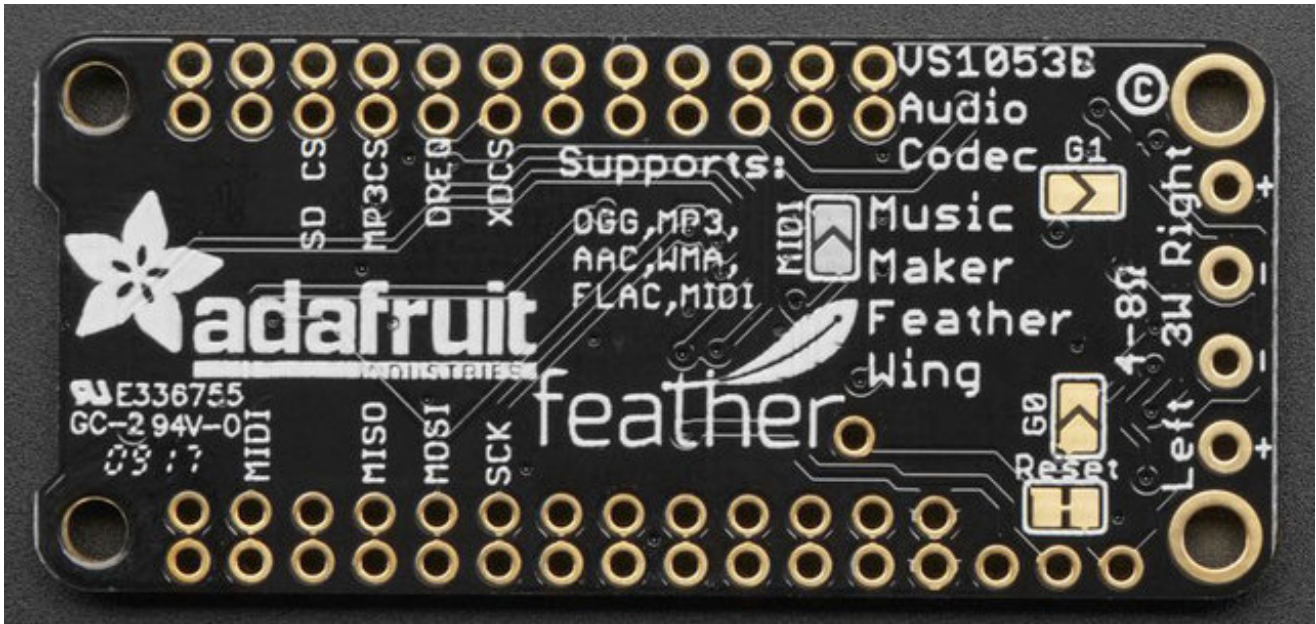
Installing software

To get started with the Music Maker, you'll need to control the built-in VS1053 chip.

So download the Adafruit VS1053 Library:

[Download latest VS1053 library](http://adafru.it/cDQ)
http://adafru.it/cDQ
Uncompress the zip file and remove the folder inide. Rename it **Adafruit_VS1053** and make sure you see **Adafruit_VS1053.cpp** and **Adafruit_VS1053.h** inside. Copy the folder to the **Libraries** folder inside your Arduino Sketchbook folder. For more details on how to install Arduino libraries, check out our detailed tutorial using the link below:
[Click to go to our Library Installation tutorial](http://adafru.it/aYM)
http://adafru.it/aYM

Play Music

Load some MP3 files

Copy 2 MP3 files to a micro SD card and name them **track001.mp3** and **track002.mp3** (this is just for the test, you can re-name them later). Then push the uSD card into the slot on the shield
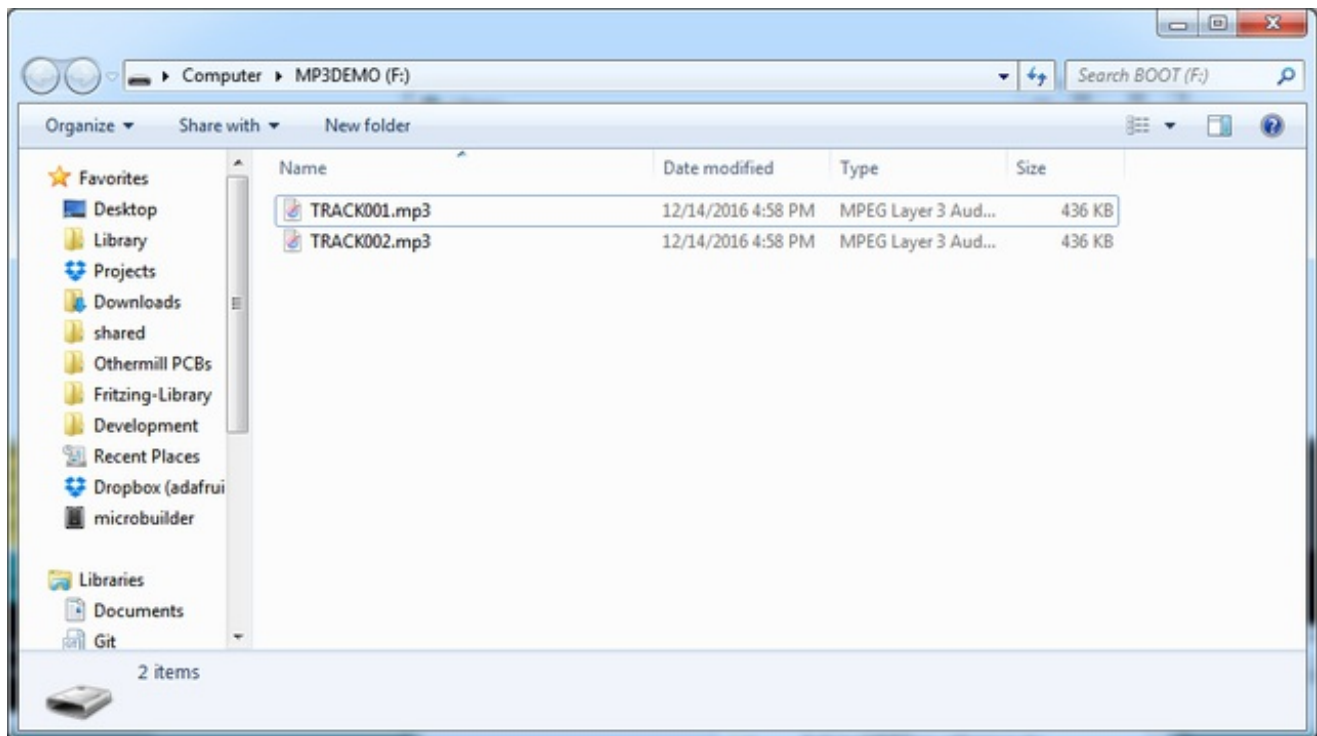
The SD library for Arduino can only handle 8.3 names, that means you can name your file track001.mp3 (8 letters dot 3 letters) but not MyFavoriteMusic.mp3

Make sure you have a good quality SD card, some cheap SD cards won't work, causing confusion! Especially 'non-brand' knockoffs.

Simple Audio Player Sketch

Connect the Feather to your computer with a USB cable and plug your headphones into the headphone jack. Since it may be loud at first, don't put them full on your ears yet.

Select **File->Examples->Adafruit_VS1053->feather_player** to load the example code.

File Edit Sketch Tools Help

New	Ctrl+N	
Open...	Ctrl+O	
Open Recent	▶	
Sketchbook	▶	
Examples	▶	
Close	Ctrl+W	
Save	Ctrl+S	
Save As...	Ctrl+Shift+S	
Page Setup	Ctrl+Shift+P	
Print	Ctrl+P	
Preferences	Ctrl+Comma	
Quit	Ctrl+Q	

Adafruit VS1053 Library ▶   feather_player
Adafruit WS2801 Library ▶   player_gpiotest
Adafruit Zero I2S Library ▶   player_interrupts
Adafruit_9DOF ▶   player_miditest
Adafruit_ADS1015 ▶   player_simple
Adafruit_ADXL345_U ▶   playmp3file
Adafruit_AHRS ▶   record_ogg
Adafruit_AS3935 ▶
Adafruit_BMP085 ▶   index number,
Adafruit_BMP183 ▶
Adafruit_BMP183_Unified ▶

For all the current Feathers, the correct pins are layed out for you in the beginning. When you upload the correct pins will automagically be used!

```
// Feather M0 or 32u4
#if defined(__AVR__) || defined(ARDUINO_SAMD_FEATHER_M0)
  #define VS1053_CS      6     // VS1053 chip select pin (output)
  #define VS1053_DCS    10     // VS1053 Data/command select pin (output)
  #define CARDCS         5     // Card chip select pin
  // DREQ should be an Int pin *if possible* (not possible on 32u4)
  #define VS1053_DREQ    9     // VS1053 Data request, ideally an Interrupt pin

// Feather ESP8266
#elif defined(ESP8266)
  #define VS1053_CS     16     // VS1053 chip select pin (output)
  #define VS1053_DCS    15     // VS1053 Data/command select pin (output)
  #define CARDCS         2     // Card chip select pin
  #define VS1053_DREQ    0     // VS1053 Data request, ideally an Interrupt pin

// Feather ESP32
#elif defined(ESP32)
  #define VS1053_CS     32     // VS1053 chip select pin (output)
  #define VS1053_DCS    33     // VS1053 Data/command select pin (output)
  #define CARDCS        14     // Card chip select pin
  #define VS1053_DREQ   15     // VS1053 Data request, ideally an Interrupt pin

// Feather Teensy3
#elif defined(TEENSYDUINO)
  #define VS1053_CS      3     // VS1053 chip select pin (output)
```

```
  #define VS1053_DCS    10    // VS1053 Data/command select pin (output)
  #define CARDCS        8     // Card chip select pin
  #define VS1053_DREQ   4     // VS1053 Data request, ideally an Interrupt pin

// WICED feather
#elif defined(ARDUINO_STM32_FEATHER)
  #define VS1053_CS     PC7   // VS1053 chip select pin (output)
  #define VS1053_DCS    PB4   // VS1053 Data/command select pin (output)
  #define CARDCS        PC5   // Card chip select pin
  #define VS1053_DREQ   PA15  // VS1053 Data request, ideally an Interrupt pin

#elif defined(ARDUINO_FEATHER52)
  #define VS1053_CS     30    // VS1053 chip select pin (output)
  #define VS1053_DCS    11    // VS1053 Data/command select pin (output)
  #define CARDCS        27    // Card chip select pin
  #define VS1053_DREQ   31    // VS1053 Data request, ideally an Interrupt pin
#endif
```
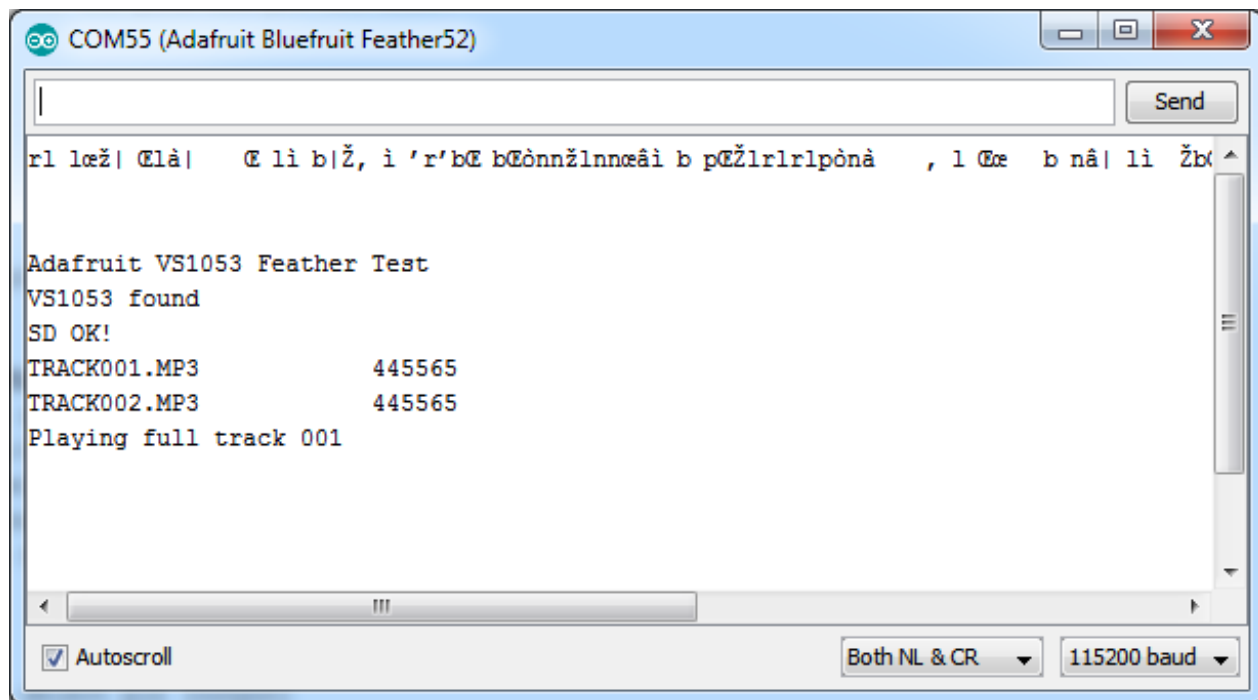
Upload & Test!

Upload the sketch to your Feather, make sure the SD card with the correct mp3 files is inserted. Open up the serial console once upload is complete

You should see the following:

And audio playing from the headphone jack.
If you get

Adafruit VS1053 Simple Test
Couldn't find VS1053, do you have the right pins defined?

Check that you've soldered all the pins of the FeatherWing and if you have soldered closed the MIDI jumper on the bottom, clear that jumper with some solder wick! (You cannot use MIDI mode and SPI control mode at once)

If you don't see **TRACK001.MP3** in the file listing, make sure you copied over the files!

Also, if you're using Bluefruit 32u4/M0 or LoRa or RFM69 Feather, make sure to disable the radio with a pinMode(8, INPUT_PULLUP); in setup()

Note that for all but the 32u4-based Feathers, playing occurs in 'the background' with an interrupt that handles reading data from the SD card and piping it to the MP3 chip!

MIDI Synth

With a simple solder jumper connection, the board will boot up in MIDI mode that will read 'classic' 31250Kbaud MIDI data on the UART pin and act like a synth/drum machine - there are dozens of built-in drum and sample effects.

On the Feathers that have a separate UART for USB communicating (e.g. 32u4, M0, WICED, Teensy and any others with native USB) you'll want to use **Serial1** as the serial port. On the ones that use the UART (e.g. nRF52, ESP8266, etc) use **Serial** and note that this will make the serial console not work right because the baud rate is now 31250 KBaud!

You may also need to remove the FeatherWing while progamming

Solder Closed MIDI Jumper

By default, MIDI mode is not 'activated' - but its very easy to turn on. Start by soldering the jumper on the bottom of the Feather.

Once that is done, you can no longer play an MP3, you will only be able to send MIDI commands via the hardware UART (TX) pin. That pin is automatically connected.
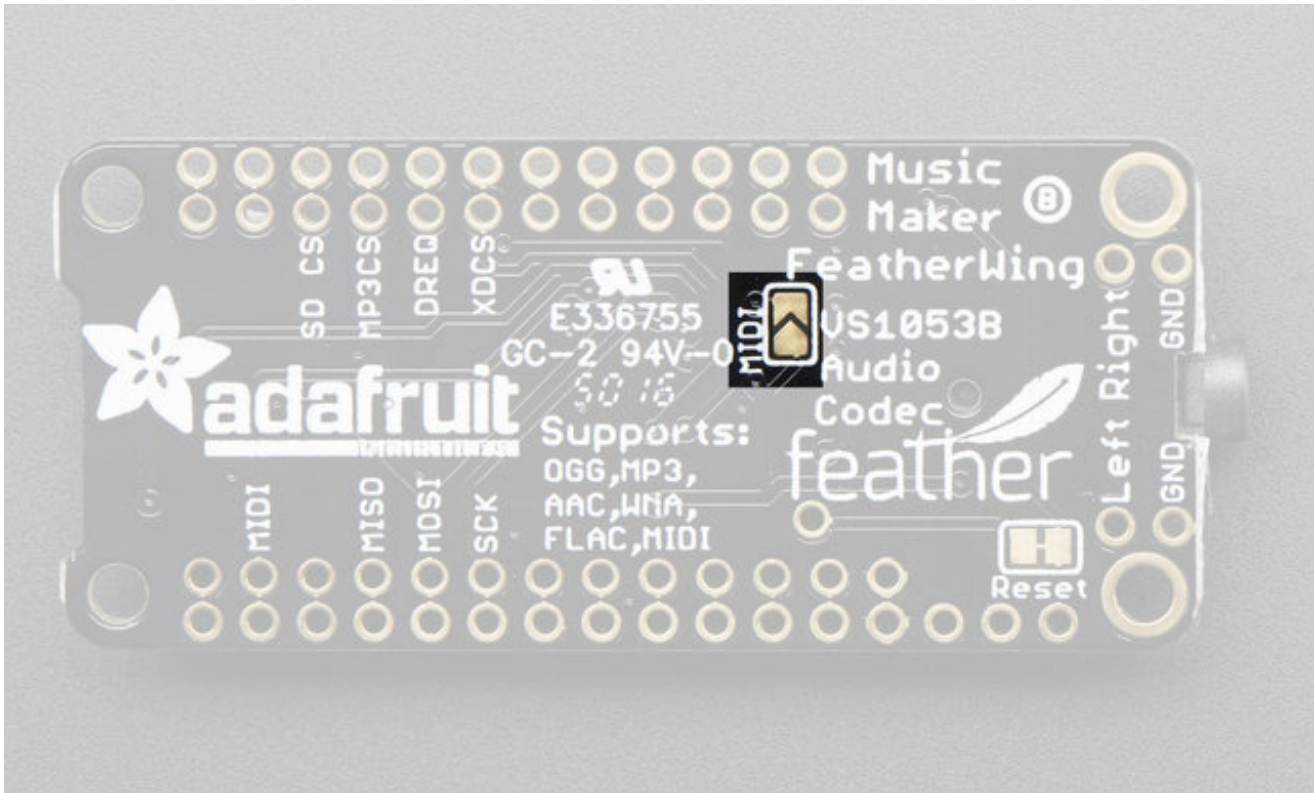
Now load up the MIDI demo **File->Examples->Adafruit_VS1053->player_miditest**
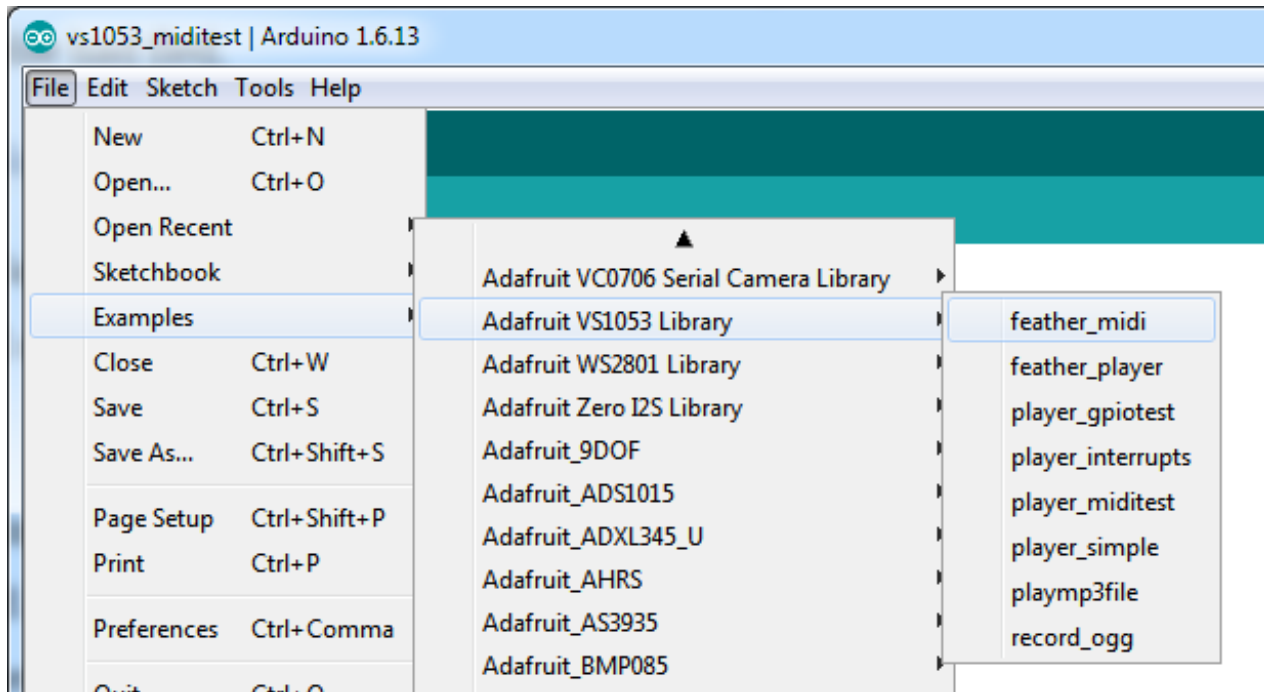
Upload to the Arduino + Shield and listen on the headphone jack for the Ocarina scale being played. You can check the datasheet for a list of all the instruments (there's a lot!)

Library Reference

class Adafruit_VS1053_FilePlayer

The Adafruit_VS1053_FilePlayer class is derived from the Adafruit_VS1053 class and provides high level functions for playing files stored on the VS1053 breakout SD Card reader.

Public Methods:

Adafruit_VS1053_FilePlayer (int8_t mosi, int8_t miso, int8_t clk, int8_t rst, int8_t cs, int8_t dcs, int8_t dreq, int8_t cardCS) - Software SPI constructor. Uses Software SPI, so you must specify all SPI pins.

Adafruit_VS1053_FilePlayer (int8_t rst, int8_t cs, int8_t dcs, int8_t dreq, int8_t cardCS) - Hardware SPI constructor. Uses Hardware SPI and assumes the default SPI pins. This is what you'll likely use if you're using the shield.

boolean begin(void) - Initialize communication and reset the chip. Returns true if a VS1053 is found

boolean useInterrupt(uint8_t type) - Specifies the interrupt to use for interrupt-driven playback. Valid arguments are:

- VS1053_FILEPLAYER_TIMER0_INT
- VS1053_FILEPLAYER_PIN_INT

boolean startPlayingFile(char *trackname) - Begin playing the specified file from the SD card using interrupt-driven playback. This allows your program to perform other tasks as the file is playing.

boolean playFullFile(char *trackname) - Play the complete file. This function will not return until the playback is complete.

Public Member Variables:

File currentTrack - File currently being played

boolean playingMusic - True if playback in progress

class Adafruit_VS1053

The Adafruit_VS1053 class implements an interface to the basic VS1053 functionality. For more detail on the operation of the VS1053 chip, please refer to the documentation on the Downloads page (see the link to the left). Its a little more powerful but it's also harder to use. We suggest sticking to the FilePlayer class which abstracts a lot of this out for you

public Methods:

Adafruit_VS1053(uint8_t mosi, uint8_t miso, uint8_t clk, uint8_t rst, uint8_t cs, uint8_t dcs, uint8_t dreq) - Software SPI constructor - must specify all pins.

Adafruit_VS1053(uint8_t rst, uint8_t cs, uint8_t dcs, uint8_t dreq)- Hardware SPI constructor - assumes hardware SPI pins.

uint8_t begin(void) - Initialize SPI communication and (hard) reset the chip.

void reset(void) - Performs a hard reset of the chip.

void softReset(void) - Attempts a soft reset of the chip.

uint16_t sciRead(uint8_t addr) - Reads from the specified register on the chip.

void sciWrite(uint8_t addr, uint16_t data) - Writes to the specified register on the chip.

void sineTest(uint8_t n, uint16_t ms) - Generate a sine-wave test signal.

void spiwrite(uint8_t d) - Low-level SPI write operation.

uint8_t spiread(void) - Low-level SPI read operation.

uint16_t decodeTime(void) - Reads the DECODETIME register from the chip.

void setVolume(uint8_t left, uint8_t right) - Set the output volume for the chip.

void dumpRegs(void) - Prints the contents of the MODE, STATUS, CLOCKF and VOLUME registers.

void playData(uint8_t *buffer, uint8_t buffsiz) - Decode and play the contents of the supplied buffer.

boolean readyForData(void) - Test if ready for more data.

void applyPatch(const uint16_t *patch, uint16_t patchsize) - Apply a code patch (See datasheet for details).

uint16_t loadPlugin(char *fn) - Load the specified plug-in.

void GPIO_digitalWrite(uint8_t i, uint8_t val) - Write to a GPIO pin.

void GPIO_digitalWrite(uint8_t i) - Write to all 8 GPIO pins at once.

uint16_t GPIO_digitalRead(void) - Read all 8 GPIO pins at once.

boolean GPIO_digitalRead(uint8_t i) - Read a single GPIO pin.

void GPIO_pinMode(uint8_t i, uint8_t dir) - Set the Pin Mode (INPUT/OUTPUT) for a GPIO pin.

boolean prepareRecordOgg(char *plugin) - Initialize chip for OGG recording.

void startRecordOgg(boolean mic) - Start recording (mic = true for microphone input).

void stopRecordOgg(void) - Stop the recording.

uint16_t recordedWordsWaiting(void) - Returns the number of words recorded.

uint16_t recordedReadWord(void) - Reads the next word from the buffer of recorded words.

uint16_t recordedReadWord(void) - Reads the next word from the buffer of recorded words.
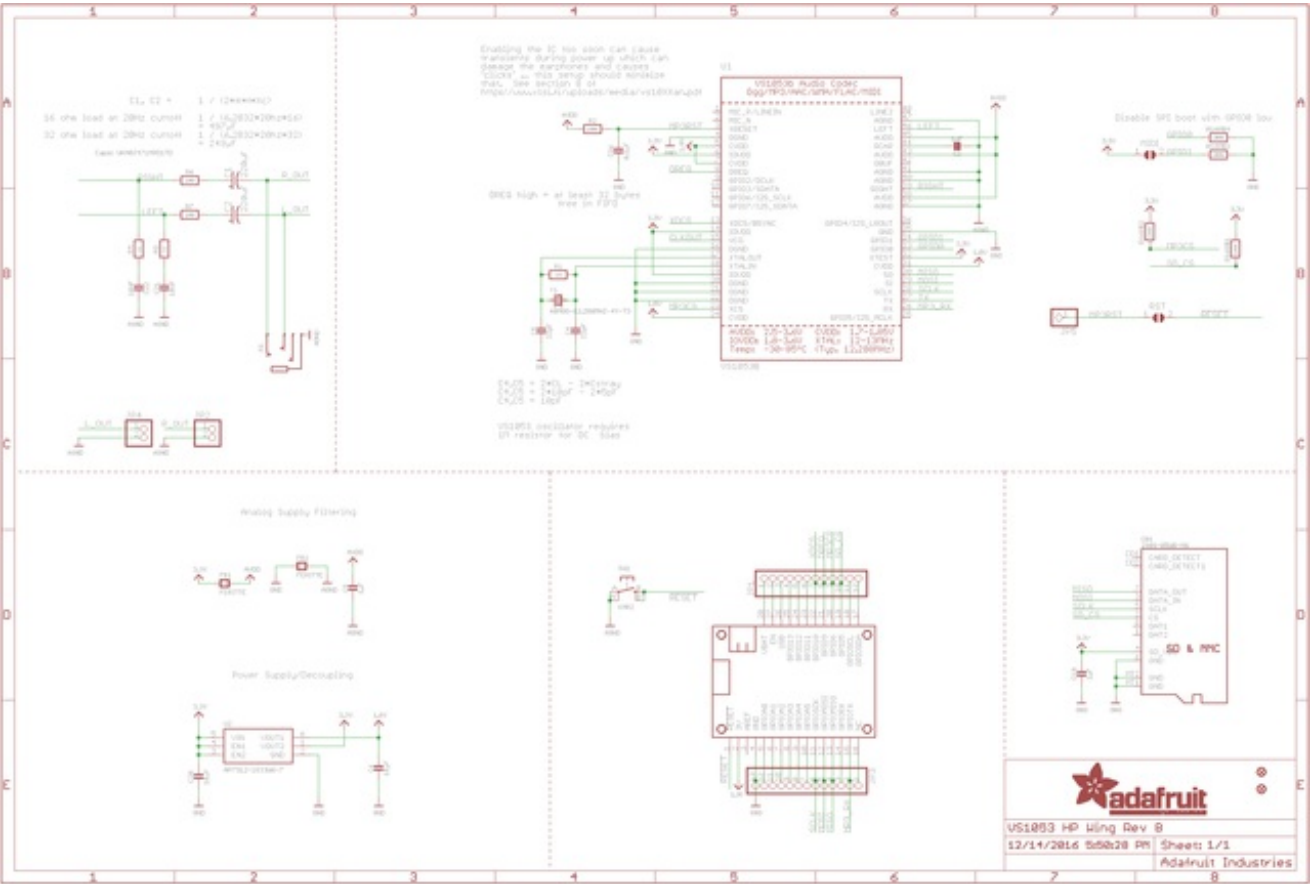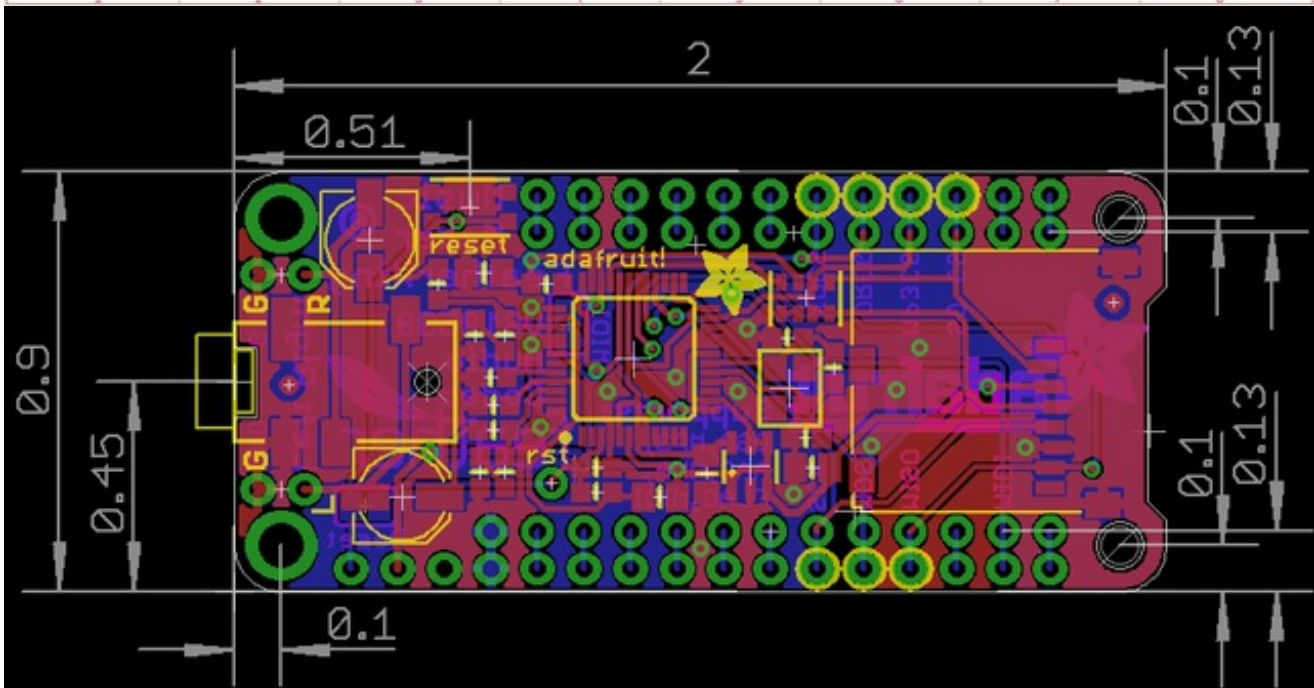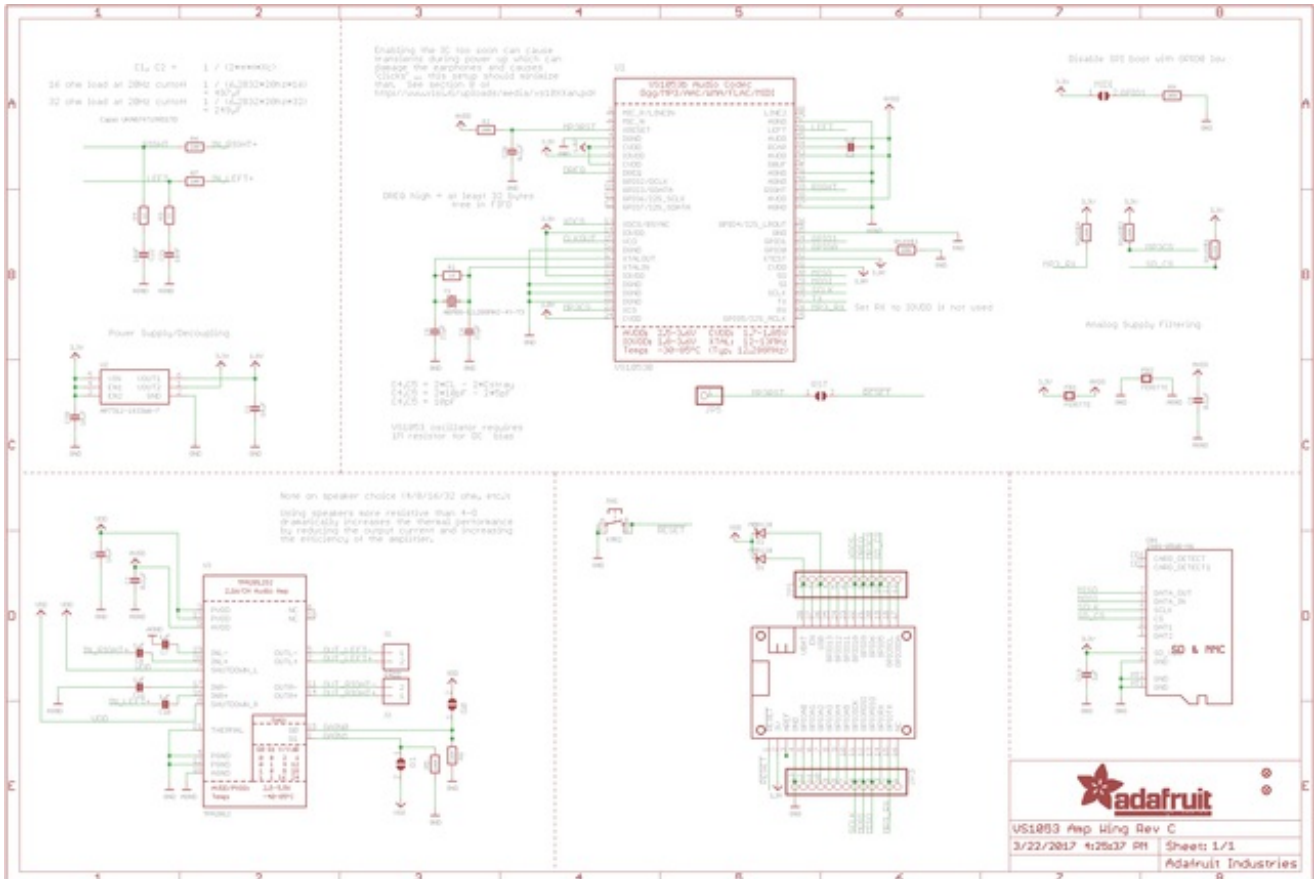
Downloads

Library:

- Adafruit VS1053 Library (http://adafru.it/clE)

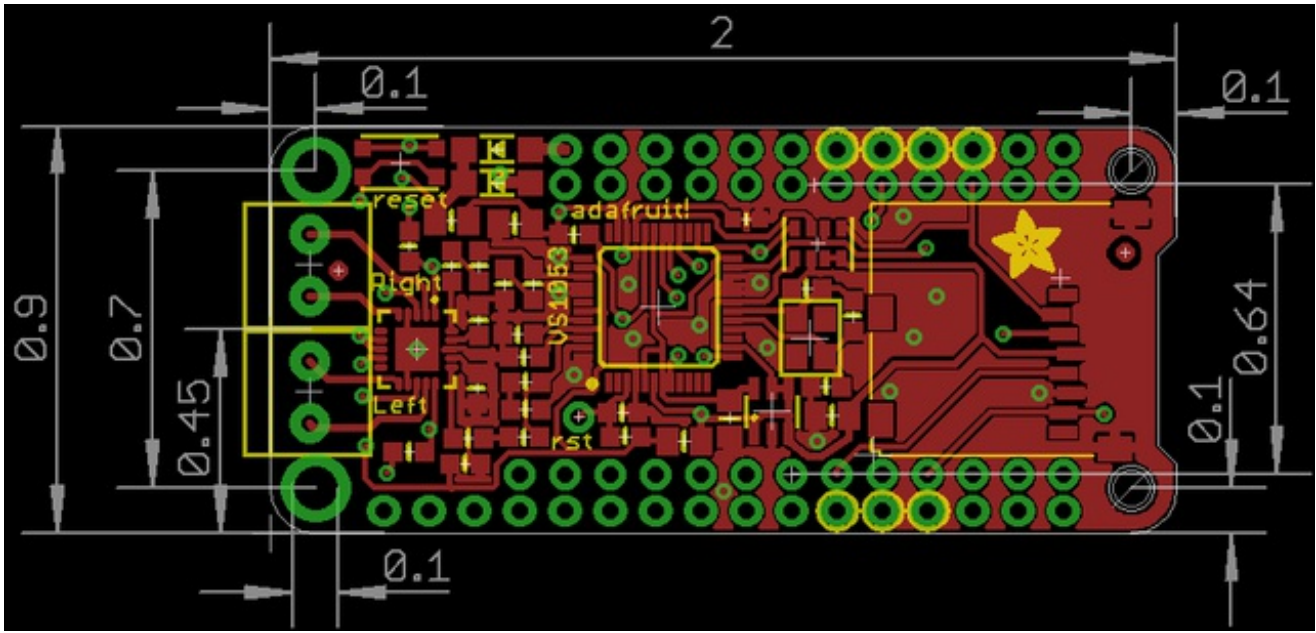# Datasheets & Files

- VS1053B (Codec chip) datasheet (http://adafru.it/cII)
- Fritzing objects in Adafruit Fritzing library (http://adafru.it/aP3)
- EagleCAD PCB files in GitHub (http://adafru.it/tbN)

Schematics

Troubleshooting

In rev B of the Featherwing, we connect the MIDI pin directly to the UART TX on the Feather without a pullup. For unknown reasons (the VS1053 is supposed to ignore the MIDI input when not in MIDI mode), having this pin floating sometimes causes a problem with decoding.

If you're having glitchy hangs, you can fix this intermediately by either:

1) Connecting a 10K - 100K pullup resistor between the Feather TX pin and 3.3V, highlighted here:
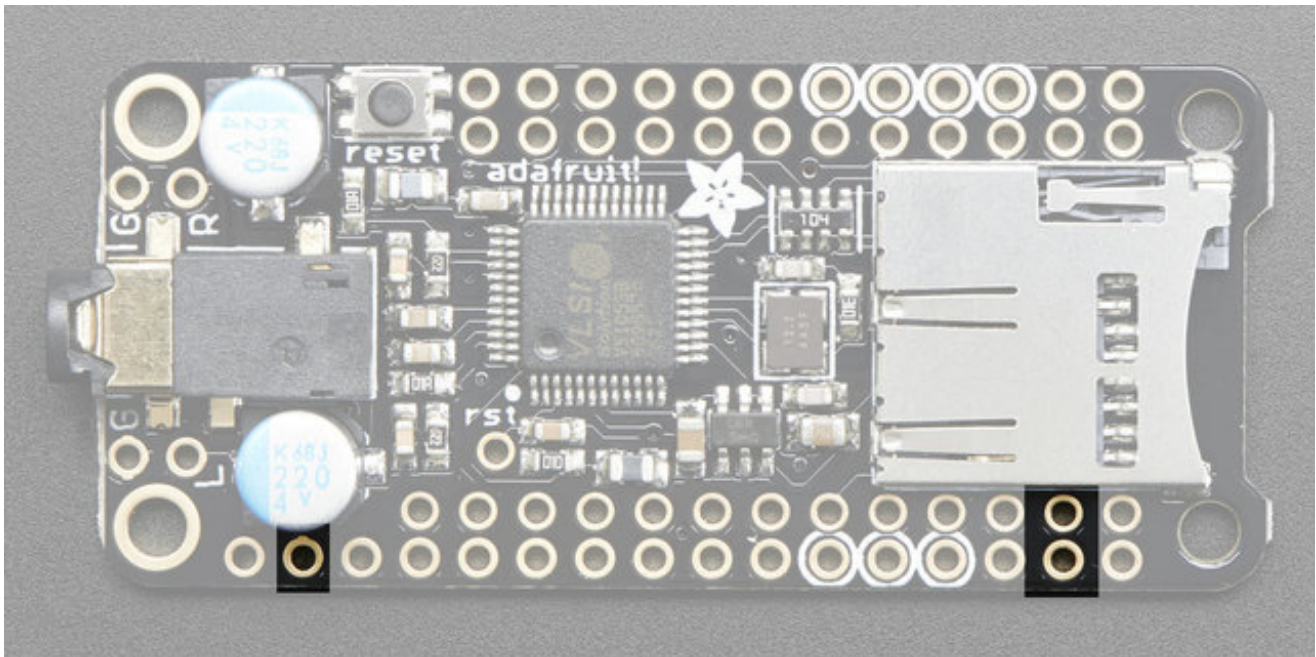
OR

Set the Feather TX pin to be a pullup input *or* output level high.

On a Feather 32u4, Feather M0, or Feather Teensy, add either to setup:
pinMode(1, INPUT_PULLUP);

or

pinMode(1, OUTPUT);
digitalWrite(1, HIGH);

On ESP8266 this is done by default so you can ignore this step (since the UART is always

on). If using another kind of Feather, substitute 1 for whatever pin number is on UART TX.