MIDI Solenoid Drum Kit

Created by Ruiz Brothers

https://learn.adafruit.com/midi-solenoid-drum-kit

Last updated on 2021-12-08 04:58:08 PM EST
# Table of Contents

## Overview
- CircuitPython MIDI
- Plug and Play
- DIY Drum Kit
- Modular Design
- Parts
- Hardware List

## Circuit Diagram
- Adafruit Library for Fritzing
- Wired Connections
- Powering

## 3D Printing
- Parts List
- Mallet Assembly
- Kick Drum Assembly
- Motion Linkage
- Slicing Parts
- Design Source Files

## CircuitPython on Feather M4 Express
- Set up CircuitPython Quick Start!
- Further Information

## CircuitPython Libraries
- The Adafruit CircuitPython Library Bundle
- Downloading the Adafruit CircuitPython Library Bundle
- The CircuitPython Community Library Bundle
- Downloading the CircuitPython Community Library Bundle
- Understanding the Bundle
- Example Files
- Copying Libraries to Your Board
- Understanding Which Libraries to Install
- Example: ImportError Due to Missing Library
- Library Install on Non-Express Boards
- Updating CircuitPython Libraries and Examples

## Coding the MIDI Solenoid Drum Kit

## CircuitPython Code Walkthrough
- Importing the Libraries
- Digital Output Pins
- MIDI Setup
- Time Keeping
- The Loop
- Solenoid and MIDI Note Array Indexes
- Play the Drum!

## Feather Header Setup
- Feather Headers
- Install Headers to Feather 30
- Breadboard Helper 30
- Female Headers for Perma-Proto 31
- Solder Perma-Proto 31
- Check Feather Headers 31

### ULN2803A Setup

- DIP Socket 32
- Installing Dip Socket 32
- Solder Socket to Perma-Proto 32
- Double Check 33

### PermaProto Wiring

- Solder Connections to Perma-Proto 33
- JST–PH Cables 33
- Cable Lengths 34

### DC Jacks and Switches

- Solenoid DC Jacks 34
- DC Jack Polarity 34
- Wire Switch and DC Jack 35

### Solenoid Wiring

- Wire Solenoids 35
- Solder Wires 35
- Soldered 2.1mm Barrel Cable 36

### Case Assembly

- Install Perma-Proto to Covers 36
- Panel Mount DC Jacks 36
- Panel Mount Switch and DC Jack 37
- Connect Switch and DC Jack 37
- Connect DC Jacks for Solenoids 37
- Install Case to Cover 37
- Install Top Cover 38

### Mallet Assembly

- Solenoid Holder 38
- Install Solenoid 38
- Install Linkage Plate 38
- Secure Plate to Holder 39
- Hardware for Base 39
- Install Standoffs 39
- Secure Standoffs 39
- Install Solenoid Holder to Base 40
- Secure Solenoid Holder to Base 40
- Mallet Hardware 41
- Secure Mallet Parts 41
- Assembled Mallet and Linkage 42
- Installing Mallet to Solenoid Plate 42
- Install Linkage to Solenoid 42
- Secure Linkage 43
- Install Linkage to Plate 43
- Secure Linkage to Plate 43
- Test Motion 44
- Degree of Rotation 44
2020 Assembly

- Assembled Solenoid Mallet
- Solenoid Mounting Parts
- Install Slim T-Nuts to Brackets
- Secure Brackets to Mounting Plate
- Installing Mounting Plate to Solenoid
- Install Mounting Plate
- Secure Mounting Plate to Solenoid
- Assembled Solenoid Mount
- Hardware for 2020 Base
- Setup Legs for 2020 Base
- Install T-Nuts onto 2020 Base
- Install Hex Nuts to 2020 Base
- Install Legs to 2020 Base
- Tighten Hex Nuts
- Pair of 2020 Base Mounts
- Install Solenoid Mount to 2020 Extrusion
- Solenoid Wiring
- Height Adjustment
- Secure 2020 Mounting Brackets
- Installing DC Cable
- Install Base Mount
- Assembled 2020 Solenoid

Usage

- Setup DAW Software Instruments
- Mac OS – Audio MIDI Setup
- Logic Pro X – External MIDI Track
- MIDI Track Port
- Keyboard Playing
- Create MIDI Notes
Overview

CircuitPython MIDI
Build a MIDI drum kit using solenoids and CircuitPython! 3D print parts to create a solenoid driven mallet to trigger snare drums, cymbals and much more! Use the Adafruit Feather M4 and ULN2803A darlington driver to create your own custom USB MIDI percussion ensemble.

Plug and Play
The Feather M4 and ULN2803A darlington driver are fitted onto a Perma-Proto board housed inside a snap fit case. DC jacks on the side of the case allow for plug and play of the solenoids.

DIY Drum Kit
DIY your own cymbals, tom toms, snare and kick drum. Use the mallets to hit any surface to make programmable percussion instruments.
Modular Design
To make the kick drum, a solenoid is mounted to a piece of 2020 extrusion that hovers over a practice drum pad. This creates a direct hit and makes a nice 'thud'.

Parts

Adafruit Feather M4 Express - Featuring ATSAMD51
It's what you’ve been waiting for, the Feather M4 Express featuring ATSAMD51. This Feather is fast like a swift, smart like an owl, strong like a ox-bird (it's half ox,...
https://www.adafruit.com/product/3857

Large push-pull solenoid
Solenoids are basically electromagnets: they are made of a big coil of copper wire with an armature (a slug of metal) in the middle. When the coil is energized, the slug is pulled into...
https://www.adafruit.com/product/413
<table>
<thead>
<tr>
<th>Quantity</th>
<th>Description</th>
<th>Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 x</td>
<td><strong>10-Wire Cable</strong></td>
<td><a href="https://www.adafruit.com/product/3890">https://www.adafruit.com/product/3890</a></td>
</tr>
<tr>
<td></td>
<td>Silicone stranded-core 28AWG</td>
<td></td>
</tr>
<tr>
<td>5 x</td>
<td><strong>DC barrel jack</strong></td>
<td><a href="https://www.adafruit.com/product/610">https://www.adafruit.com/product/610</a></td>
</tr>
<tr>
<td></td>
<td>Panel Mount 2.1mm DC barrel jack</td>
<td></td>
</tr>
<tr>
<td>4 x</td>
<td><strong>2.1mm DC Plug</strong></td>
<td><a href="https://www.adafruit.com/product/2788">https://www.adafruit.com/product/2788</a></td>
</tr>
<tr>
<td></td>
<td>DC Jack Adapter Cable</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td><strong>12V 5A Power supply</strong></td>
<td><a href="https://www.adafruit.com/product/352">https://www.adafruit.com/product/352</a></td>
</tr>
<tr>
<td></td>
<td>12V 5A switching power supply</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td><strong>Feather Header Kit</strong></td>
<td><a href="https://www.adafruit.com/product/2886">https://www.adafruit.com/product/2886</a></td>
</tr>
<tr>
<td></td>
<td>Header Kit for Feather - 12-pin and 16-pin Female Header Set</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td><strong>USB Cable</strong></td>
<td><a href="https://www.adafruit.com/product/4111">https://www.adafruit.com/product/4111</a></td>
</tr>
<tr>
<td></td>
<td>Fully Reversible Pink/Purple USB A to micro B Cable - 1m long</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td><strong>10-wire silicone ribbon cable</strong></td>
<td><a href="https://www.adafruit.com/product/3890">https://www.adafruit.com/product/3890</a></td>
</tr>
<tr>
<td></td>
<td>10-wire silicone ribbon cable</td>
<td></td>
</tr>
</tbody>
</table>

---

**Adafruit Perma-Proto Half-sized Breadboard PCB - Single**
Customers have asked us to carry basic perf-board, but we never liked the look of most basic perf: it’s always crummy quality, with pads that flake off and no labeling. Then we...
[https://www.adafruit.com/product/1609](https://www.adafruit.com/product/1609)

**ULN2803: 8 Channel Darlington Driver (Solenoid/Unipolar Stepper)**
Bring in some muscle to your output pins with 8 mighty Darlington drivers! This DIP chip contains 8 drivers that can sink 500mA from a 50V supply and has kickback diodes included inside for...
[https://www.adafruit.com/product/970](https://www.adafruit.com/product/970)
6 x JST PH 2-Pin Cable
Female Connector 100mm
https://www.adafruit.com/product/261

6 x JST PH 2-Pin Cable
Male Header 200mm
https://www.adafruit.com/product/3814

1 x Toggle Switch
Mini Panel Mount SPDT Toggle Switch
https://www.adafruit.com/product/3221

1 x 18 Pin DIP Socket
18 (2 x 9) Pos DIP, 0.3" (7.62mm) Row Spacing Socket Tin Through Hole

1 x 2020 Extrusion
Slotted Aluminum Extrusion - 20mm x 20mm - 610mm long
https://www.adafruit.com/product/1221

1 x Slim T-Nuts
Aluminum Extrusion Slim T-Nut for 20x20 - M4 Thread - pack of 50
https://www.adafruit.com/product/1157

1 x M4 Machine Screw
Button Hex - 8mm long - pack of 50
https://www.adafruit.com/product/1160

1 x Remo Drum Pad
Gray Tunable Practice Pad with Ambassador Coated Drumhead
https://amzn.to/2XptKed

1 x Small Cymbals
2pc First Note FN240 Firstnote Cymbals
https://amzn.to/2Pmjjna

Hardware List
Screws, standoffs hex nuts and locknuts used to assemble and fasten parts.

Mallet Assembly (Single Set)

- 12x M3x6mm screws
- 4x M3x10mm standoffs
- 2x M3x16mm screws
- 3x M3 locknuts
- 1x M3x14mm screw
Kick Drum Assembly

- 1x 2020 Aluminum Extrusion – 305mm long
- 4x M4x8mm screws
- 4x M4 slim t-nuts
- 2x M3x10mm screws
- 2x M3 hex nuts
- 4x M3x6mm screws
- 4x M5x30mm screws
- 6x M5 hex nuts

Cymbal Assembly

- 1/4-20 D-ring tripod screw (https://adafru.it/MBJ)
- 3/8 to 1/4 screw adapter (https://adafru.it/toF)
- 2x M5x10mm screw
- 2x M5 hex nuts

Circuit Diagram

The diagram below provides a visual reference for wiring of the components. This diagram was created using the software package Fritzing (https://adafru.it/oEP).

Adafruit Library for Fritzing

Use Adafruit's Fritzing parts library to create circuit diagrams for your projects. Download the library or just grab individual parts. Get the library and parts from GitHub - Adafruit Fritzing Parts (https://adafru.it/AYZ).
Wired Connections

Feather M4 Express

- D5 from Feather to Pin #5 on ULN2803A
- D6 from Feather to Pin #6 on ULN2803A
- D9 from Feather to Pin #7 on ULN2803A
- D10 from Feather to Pin #8 on ULN2803A
- 3V from Feather to VCC (upper) rails on Perma-Proto
- GND from Feather to Ground (upper) rails on Perma-Proto

Switch

- EN from Feather to Switch
- GND from Feather to Switch

DC Jack

- Ground to Ground (lower) rails on Perma-Proto
- VCC to VCC (lower) rails on Perma-Proto

ULN2803A

- Pin #10 (VCC) from ULN2803A to VCC (lower) on Perma-Proto
Pin #9 (GND) from ULN2803A to GND (lower) rails on Perma-Proto

Powering

The Adafruit board can be powered via USB or JST using a 3.7v lipo battery. In this project, a 400mAh lipo battery is used. The lipo battery is rechargeable via the USB port on the board. The switch is wired to the enable and ground pins on the board.

12V Power

The 12V 5A power supply is plugged into the DC jack on the lower power and ground rails on the Perma-Proto.

The power and ground rails connected to the 12V 5A power supply must only be used to power the solenoids NOT the Feather!
3D Printing

Parts List
STL files for 3D printing are oriented to print "as-is" on FDM style machines. Parts are designed to 3D print without any support material. Original design source may be downloaded using the links below.

File names & Quantity

- 1x snare-noid-mount.stl
- 2x snare-extrusion-mount.stl
- 2x snare-extrusion-bracket.stl
- 3x mallet-stick.stl
- 3x mallet-noid-plate.stl
- 3x mallet-noid-linkage.stl
- 3x mallet-noid-holder.stl
- 3x mallet-noid-bottom-base.stl
- 6x mallet-ball.stl
- 1x feather-case-top-cover.stl
- 1x feather-case-frame.stl
- 1x feather-case-bottom-cover.stl

Download CAD files from PrusaPrinters
https://adafru.it/MBK

Download CAD files from Thingiverse
https://adafru.it/MBL
Mallet Assembly
The solenoid is inserted into the holder. The solenoid holder is attached to the standoffs on the base with M3x6mm screws. The linkage connector is pinned to the solenoids plunger with an M3x16mm screw and locknut. The linkage mounting plate is secured to the solenoid holder with M3x6mm screws. The linkage is then secured to the mounting plate with an M3x16mm screw and locknut. M3x10mm long standoffs are secured to the base for elevation.

Kick Drum Assembly
The solenoid is secured to the mounting plate. Two mounting brackets are secured to the slotted holes on the mounting plate. The brackets slide onto the profile of the 2020 extrusion and secured with M4x8mm screws and slim t-nuts. The base mount is secured to the 2020 extrusion with an M4x8mm long screw and slim t-nut.

Motion Linkage
This selection analysis shows the linear motion of the plunger pushing the linkage bar and pivoting at the joint to make the mallet strike. This was created in Fusion 360 using joints and contact sets. Revolute joint type is used in the pivoting point in the linkage bar. Slider joint type is used in the plunger of the solenoid. Contact sets are created between the linkage bar and M3 screw secured to the plunger.
Slicing Parts
No supports are required. Slice with settings for PLA material.

The parts were sliced using CURA using the slice settings below.

- PLA filament 220c extruder
- 0.2 layer height
- 10% gyroid infill
- 90mm/s print speed
- 60c heated bed

Design Source Files
The project assembly was designed in Fusion 360. This can be downloaded in different formats like STEP, STL and more. Electronic components like Adafruit's board, displays, connectors and more can be downloaded from the Adafruit CAD parts GitHub Repo (https://adafru.it/AW8).

CircuitPython on Feather M4 Express

CircuitPython (https://adafru.it/tB7) is a derivative of MicroPython (https://adafru.it/BeZ) designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the CIRCUITPY drive to iterate.

The following instructions will show you how to install CircuitPython. If you've already installed CircuitPython but are looking to update it or reinstall it, the same steps work for that as well!

Set up CircuitPython Quick Start!

Follow this quick step-by-step for super-fast Python power :)

©Adafruit Industries
Download the latest version of CircuitPython for this board via CircuitPython.org
https://adafruit.it/Emh

Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).

Plug your Feather M4 into your computer using a known-good USB cable.

A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Double-click the Reset button next to the USB connector on your board, and you will see the NeoPixel RGB LED turn green. If it turns red, check the USB cable, try another USB port, etc. Note: The little red LED next to the USB connector will pulse red. That's ok!

If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!
You will see a new disk drive appear called FEATHERBOOT.

Drag the adafruit_circuitpython_etc.uf2 file to FEATHERBOOT.

The LED will flash. Then, the FEATHERBOOT drive will disappear and a new disk drive called CIRCUITPY will appear.

That's it, you're done! :)

Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafru.it/Amd).
CircuitPython Libraries

As CircuitPython development continues and there are new releases, Adafruit will stop supporting older releases. Visit https://circuitpython.org/downloads to download the latest version of CircuitPython for your board. You must download the CircuitPython Library Bundle that matches your version of CircuitPython. Please update CircuitPython and then visit https://circuitpython.org/libraries to download the latest Library Bundle.

Each CircuitPython program you run needs to have a lot of information to work. The reason CircuitPython is so simple to use is that most of that information is stored in other files and works in the background. These files are called libraries. Some of them are built into CircuitPython. Others are stored on your CIRCUITPY drive in a folder called lib. Part of what makes CircuitPython so great is its ability to store code separately from the firmware itself. Storing code separately from the firmware makes it easier to update both the code you write and the libraries you depend.

Your board may ship with a lib folder already, it's in the base directory of the drive. If not, simply create the folder yourself. When you first install CircuitPython, an empty lib directory will be created for you.

CircuitPython libraries work in the same way as regular Python modules so the Python docs (https://adafruit.it/rar) are an excellent reference for how it all should work. In Python terms, you can place our library files in the lib directory because it's part of the Python path by default.

One downside of this approach of separate libraries is that they are not built in. To use them, one needs to copy them to the CIRCUITPY drive before they can be used. Fortunately, there is a library bundle.
The bundle and the library releases on GitHub also feature optimized versions of the libraries with the .mpy file extension. These files take less space on the drive and have a smaller memory footprint as they are loaded.

Due to the regular updates and space constraints, Adafruit does not ship boards with the entire bundle. Therefore, you will need to load the libraries you need when you begin working with your board. You can find example code in the guides for your board that depends on external libraries.

Either way, as you start to explore CircuitPython, you'll want to know how to get libraries on board.

The Adafruit CircuitPython Library Bundle

Adafruit provides CircuitPython libraries for much of the hardware they provide, including sensors, breakouts and more. To eliminate the need for searching for each library individually, the libraries are available together in the Adafruit CircuitPython Library Bundle. The bundle contains all the files needed to use each library.

Downloading the Adafruit CircuitPython Library Bundle

You can download the latest Adafruit CircuitPython Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

Match up the bundle version with the version of CircuitPython you are running. For example, you would download the 6.x library bundle if you're running any version of CircuitPython 6, or the 7.x library bundle if you're running any version of CircuitPython 7, etc. If you mix libraries with major CircuitPython versions, you will get incompatible .mpy errors due to changes in library interfaces possible during major version changes.

Click to visit circuitpython.org for the latest Adafruit CircuitPython Library Bundle

https://adafruit.it/ENC
Download the bundle version that matches your CircuitPython firmware version. If you don't know the version, check the version info in boot_out.txt file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.

There's also a py bundle which contains the uncompressed python files, you probably don't want that unless you are doing advanced work on libraries.

The CircuitPython Community Library Bundle

The CircuitPython Community Library Bundle is made up of libraries written and provided by members of the CircuitPython community. These libraries are often written when community members encountered hardware not supported in the Adafruit Bundle, or to support a personal project. The authors all chose to submit these libraries to the Community Bundle make them available to the community.

These libraries are maintained by their authors and are not supported by Adafruit. As you would with any library, if you run into problems, feel free to file an issue on the GitHub repo for the library. Bear in mind, though, that most of these libraries are supported by a single person and you should be patient about receiving a response. Remember, these folks are not paid by Adafruit, and are volunteering their personal time when possible to provide support.

Downloading the CircuitPython Community Library Bundle

You can download the latest CircuitPython Community Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

Click for the latest CircuitPython Community Library Bundle release
https://adafruit.it/VCn

The link takes you to the latest release of the CircuitPython Community Library Bundle on GitHub. There are multiple versions of the bundle available. Download the bundle version that matches your CircuitPython firmware version. If you don't know the version, check the version info in boot_out.txt file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.
Understanding the Bundle

After downloading the zip, extract its contents. This is usually done by double clicking on the zip. On Mac OSX, it places the file in the same directory as the zip.

Open the bundle folder. Inside you'll find two information files, and two folders. One folder is the lib bundle, and the other folder is the examples bundle.

Now open the lib folder. When you open the folder, you'll see a large number of .mpy files, and folders.

Example Files

All example files from each library are now included in the bundles in an examples directory (as seen above), as well as an examples-only bundle. These are included for two main reasons:

- Allow for quick testing of devices.
- Provide an example base of code, that is easily built upon for individualized purposes.
Copying Libraries to Your Board

First open the lib folder on your CIRCUITPY drive. Then, open the lib folder you extracted from the downloaded zip. Inside you'll find a number of folders and .mpy files. Find the library you'd like to use, and copy it to the lib folder on CIRCUITPY.

If the library is a directory with multiple .mpy files in it, be sure to copy the entire folder to CIRCUITPY/lib.

This also applies to example files. Open the examples folder you extracted from the downloaded zip, and copy the applicable file to your CIRCUITPY drive. Then, rename it to code.py to run it.

If a library has multiple .mpy files contained in a folder, be sure to copy the entire folder to CIRCUITPY/lib.

Understanding Which Libraries to Install

You now know how to load libraries on to your CircuitPython-compatible microcontroller board. You may now be wondering, how do you know which libraries you need to install? Unfortunately, it's not always straightforward. Fortunately, there is an obvious place to start, and a relatively simple way to figure out the rest. First up: the best place to start.

When you look at most CircuitPython examples, you'll see they begin with one or more `import` statements. These typically look like the following:

- `import library_or_module`

However, `import` statements can also sometimes look like the following:

- `from library_or_module import name`
They can also have more complicated formats, such as including a `try / except` block, etc.

The important thing to know is that an `import` statement will always include the name of the module or library that you're importing.

Therefore, the best place to start is by reading through the `import` statements.

Here is an example import list for you to work with in this section. There is no setup or other code shown here, as the purpose of this section involves only the import list.

```python
import time
import board
import neopixel
import adafruit_lis3dh
import usb_hid
from adafruit_hid.consumer_control import ConsumerControl
from adafruit_hid.consumer_control_code import ConsumerControlCode
```

Keep in mind, not all imported items are libraries. Some of them are almost always built-in CircuitPython modules. How do you know the difference? Time to visit the REPL.

In the Interacting with the REPL section (https://adafru.it/Awz) on The REPL page (http://adafru.it/Awz) in this guide, the `help("modules")` command is discussed. This command provides a list of all of the built-in modules available in CircuitPython for your board. So, if you connect to the serial console on your board, and enter the REPL, you can run `help("modules")` to see what modules are available for your board. Then, as you read through the `import` statements, you can, for the purposes of figuring out which libraries to load, ignore the statement that import modules.

The following is the list of modules built into CircuitPython for the Feather RP2040. Your list may look similar or be anything down to a significant subset of this list for smaller boards.
Now that you know what you’re looking for, it’s time to read through the import statements. The first two, `time` and `board`, are on the modules list above, so they’re built-in.

The next one, `neopixel`, is not on the module list. That means it’s your first library! So, you would head over to the bundle zip you downloaded, and search for neopixel. There is a neopixel.mpy file in the bundle zip. Copy it over to the lib folder on your CI RUCITPY drive. The following one, `adafruit_lis3dh`, is also not on the module list. Follow the same process for adafruit_lis3dh, where you’ll find adafruit_lis3dh.mpy, and copy that over.

The fifth one is `usb_hid`, and it is in the modules list, so it is built in. Often all of the built-in modules come first in the import list, but sometimes they don’t! Don’t assume that everything after the first library is also a library, and verify each import with the modules list to be sure. Otherwise, you’ll search the bundle and come up empty!

The final two imports are not as clear. Remember, when `import` statements are formatted like this, the first thing after the `from` is the library name. In this case, the library name is `adafruit_hid`. A search of the bundle will find an adafruit_hid folder. When a library is a folder, you must copy the entire folder and its contents as it is in the bundle to the lib folder on your CIRCUITPY drive. In this case, you would copy the entire adafruit_hid folder to your CIRCUITPY/lib folder.

Notice that there are two imports that begin with `adafruit_hid`. Sometimes you will need to import more than one thing from the same library. Regardless of how many times you import the same library, you only need to load the library by copying over the adafruit_hid folder once.

That is how you can use your example code to figure out what libraries to load on your CircuitPython-compatible board!
There are cases, however, where libraries require other libraries internally. The internally required library is called a dependency. In the event of library dependencies, the easiest way to figure out what other libraries are required is to connect to the serial console and follow along with the *ImportError* printed there. The following is a very simple example of an *ImportError*, but the concept is the same for any missing library.

**Example: *ImportError* Due to Missing Library**

If you choose to load libraries as you need them, or you're starting fresh with an existing example, you may end up with code that tries to use a library you haven't yet loaded. This section will demonstrate what happens when you try to utilise a library that you don't have loaded on your board, and cover the steps required to resolve the issue.

This demonstration will only return an error if you do not have the required library loaded into the lib folder on your CIRCUITPY drive.

Let's use a modified version of the Blink example.

```python
import board
import time
import simpleio

led = simpleio.DigitalOut(board.LED)

while True:
    led.value = True
    time.sleep(0.5)
    led.value = False
    time.sleep(0.5)
```

Save this file. Nothing happens to your board. Let's check the serial console to see what's going on.

You have an *ImportError*. It says there is *no module named 'simpleio'*. That's the one you just included in your code!
Click the link above to download the correct bundle. Extract the lib folder from the downloaded bundle file. Scroll down to find simpleio.mpy. This is the library file you're looking for! Follow the steps above to load an individual library file.

The LED starts blinking again! Let's check the serial console.

![Serial Console](image)

No errors! Excellent. You've successfully resolved an `ImportError`!

If you run into this error in the future, follow along with the steps above and choose the library that matches the one you're missing.

**Library Install on Non-Express Boards**

If you have an M0 non-Express board such as Trinket M0, Gemma M0, QT Py M0, or one of the M0 Trinkeys, you'll want to follow the same steps in the example above to install libraries as you need them. Remember, you don't need to wait for an `ImportError` if you know what library you added to your code. Open the library bundle you downloaded, find the library you need, and drag it to the lib folder on your CIRCUITPY drive.

You can still end up running out of space on your M0 non-Express board even if you only load libraries as you need them. There are a number of steps you can use to try to resolve this issue. You'll find suggestions on the [Troubleshooting page](https://adafruit.it/Den).

**Updating CircuitPython Libraries and Examples**

Libraries and examples are updated from time to time, and it's important to update the files you have on your CIRCUITPY drive.

To update a single library or example, follow the same steps above. When you drag the library file to your lib folder, it will ask if you want to replace it. Say yes. That's it!
A new library bundle is released every time there's an update to a library. Updates include things like bug fixes and new features. It's important to check in every so often to see if the libraries you're using have been updated.

### Coding the MIDI Solenoid Drum Kit

Once you've finished setting up your Feather M4 Express with CircuitPython, you can add this library to the lib folder:

- adafruit_midi

Then, you can click on the Download: Project Zip link below to download the code.

```python
import time
import board
import digitalio
import usb_midi
import adafruit_midi
from adafruit_midi.note_on import NoteOn

# pins for the solenoid output signals
noid_pins = [board.D5, board.D6, board.D9, board.D10]

# array for the solenoids
noids = []

# setup for the solenoid pins to be outputs
for pin in noid_pins:
    noid = digitalio.DigitalInOut(pin)
    noid.direction = digitalio.Direction.OUTPUT
    noids.append(noid)

# MIDI note array
notes = [60, 61, 62, 63]

# MIDI in setup
midi = adafruit_midi.MIDI(midi_in=usb_midi.ports[0], in_channel=0)

# delay for solenoids
speed = 0.03
retract = 0

while True:
    # msg holds MIDI messages
    msg = midi.receive()
    for i in range(4):
        # states for solenoid on/off
        noid_output = noids[i]

    # states for MIDI note recieved
    notes_played = notes[i]

    # if NoteOn msg comes in and the MIDI note # matches with predefined notes:
    if isinstance(msg, NoteOn) and msg.note is notes_played:
        print(time.monotonic(), msg.note)
```
# solenoid is triggered
noid_output.value = True

# quick delay
retract = time.monotonic()

# retracts solenoid using time.monotonic() to avoid delays between notes activating
if (retract + speed) < time.monotonic():
    noid_output.value = False

Your Feather M4 Express CIRCUITPY drive should look like this after you load the libraries and code.py file:

![CIRCUITPY drive](image)

## CircuitPython Code Walkthrough

### Importing the Libraries

The code begins by importing the CircuitPython libraries.

```python
import time
import board
import digitalio
import usb_midi
import adafruit_midi
from adafruit_midi.note_on import NoteOn
```

### Digital Output Pins

Next, the Feather's digital pins are setup to be outputs to send on and off signals to the ULN2803, which will activate the solenoids.

```python
# pins for the solenoid output signals
noid_pins = [board.D5, board.D6, board.D9, board.D10]

# array for the solenoids
noids = []

# setup for the solenoid pins to be outputs
```
for pin in noid_pins:
    noid = digitalio.DigitalInOut(pin)
    noid.direction = digitalio.Direction.OUTPUT
    noids.append(noid)

MIDI Setup

The MIDI notes are setup next. If you need to change the MIDI notes that will activate the solenoids, you can edit this array with the note numbers that you need.

The MIDI object is setup after the notes. The Feather is setup to be a MIDI-in device, meaning that it is receiving MIDI data.

# MIDI note array
notes = [60, 61, 62, 63]

# MIDI in setup
midi = adafruit_midi.MIDI(midi_in=usb_midi.ports[0], in_channel=0)

Time Keeping

Finally, two variables are setup. speed will act as a delay for how long the solenoids will remain activated before retracting. retract will be a time.monotonic() device.

# delay for solenoids
speed = 0.03
retract = 0

The Loop

The loop begins by setting up msg to receive any incoming MIDI data.

while True:
    # msg holds MIDI messages
    msg = midi.receive()

Solenoid and MIDI Note Array Indexes

Next, the solenoids' array index position is setup the be held in noid_output. The same is done for the MIDI note numbers with notes_played.
for i in range(4):
    # states for solenoid on/off
    noid_output = noids[i]
    # states for MIDI note received
    notes_played = notes[i]

Play the Drum!

Then the real action of the code takes place. If a `NoteOn` MIDI message is received that matches one of the MIDI note numbers listed in `notesPlayed`, then the matching solenoids are activated. `retract` is also updated to hold the current value of `time.monotonic()`.

```python
# if NoteOn msg comes in and the MIDI note # matches with predefined notes:
if isinstance(msg, NoteOn) and msg.note is notes_played:
    print(time.monotonic(), msg.note)
    # solenoid is triggered
    noid_output.value = True
    # quick delay
    retract = time.monotonic()
```

Finally, the solenoids retract when the sum of `retract` and `speed (0.03)` is less than the current `time.monotonic()` value.

By doing this, you can activate your solenoids at the same time to stay on beat. If you used the more traditional `time.sleep(value)` to delay the solenoids' retractions, you would run into delays in getting the solenoids to hit the drums.

```python
# retracts solenoid using time.monotonic() to avoid delays between notes activating
if (retract + speed) < time.monotonic():
    noid_output.value = False
```
Feather Header Setup

**Feather Headers**
The Feather M4 Express and ULN2803A will be fitted onto a half size perma-proto PCB. In order to make them removable, the Feather will use female headers. The ULN2803A will snap onto a DIP socket.

**Install Headers to Feather**
A 12-pin and 16-pin strip of male headers are soldered to the pins on the Feather M4 Express.

**Breadboard Helper**
A breadboard can help assist by keeping the header pins in place while soldering.
Female Headers for Perma-Proto
Female headers are fitted onto male headers on the Feather M4 Express then fitted onto the Perma-Proto half size PCB. Note the placement of the Feather shows the USB port close to the edge.

Mounting tack can be used to temporarily secure the female headers to the Perma-Proto PCB.

Solder Perma-Proto
Apply solder to all of the pins on the bottom of the Perma-Proto PCB. A panavise can help assist by holding the PCB in place while soldering.

Check Feather Headers
Carefully remove the Feather M4 Express by slowly pulling it out from the female headers. Check to ensure all of the pins have solid solder joints.
ULN2803A Setup

DIP Socket
The ULN2803A will snap onto an 18 pin DIP socket. This allows the IC to be removable if it ever needs to be replaced.

Installing Dip Socket
The DIP socket is placed onto the Perma-Proto half size PCB. Reference the photo for best placement. Use pieces of mounting tack to temporarily secure the socket to the PCB.

Solder Socket to Perma-Proto
Apply solder to all 18 pins from the bottom the Perma-Proto.
Double Check
Check to ensure all of the pins are properly soldered and have solid solder joints.

PermaProto Wiring

Solder Connections to Perma-Proto
Reference the circuit diagram for the wired connections. 10-wire silicone cover stranded core ribbon cable is used to keep connections bundled together.

JST–PH Cables
Various JST-PH cables are used for the 4x solenoid DC jacks, toggle switch and 12V DC jack power input. Pieces of heat shrink tubing is used to insulate exposed wire connections.
Cable Lengths
The length of wire is relatively short in order to fit the enclosure. Male JST-PH cables are used for the solenoids. Female JST-PH cables are used for the 12V DC power input and the toggle switch.

DC Jacks and Switches

Solenoid DC Jacks
Four DC jacks are used to connect the solenoids. These will be panel mounted to the enclosure. JST-PH female cables are soldered to each DC jack. The cables are approximately 30mm in length (1.2in).

DC Jack Polarity
Reference the photo for the correct polarity. The terminal with the large contact is VCC / voltage. The middle pin is not used. The remaining pin is ground.
**Wire Switch and DC Jack**
The toggle switch and 12VDC jack are wired to male JST-PH cables. These are also about 30mm in length (1.2in).

---

**Solenoid Wiring**

**Wire Solenoids**
The 12V solenoids come with long pre-tinned wires. These will be soldered to a cable with a 2.1mm barrel plug. This makes it easy to plug it into the DC jacks.

**Solder Wires**
Using wire strippers, remove a bit of insulation for each wire. Tin the wires by adding a bit of solder. This helps prevent the strands of wire from fraying. Solder the wires from the solenoid to the cable with the 2.1mm barrel plug. The wire with the lined dashes is the VCC / voltage wire. Use pieces of heat shrink tubing to insulation the exposed connections.
Soldered 2.1mm Barrel Cable
Check the cabling is properly soldered and has a solid connections. Repeat this process for the other solenoids.

Case Assembly

Install Perma-Proto to Covers
The wired Perma-Proto snap fits onto the bottom cover of the enclosure. Place the PCB over the standoffs and fit edge of the PCB under one of the tabs on the side. Press down to snap fit onto other tab. Adjust wires so the cables are not being kinked or pinched.

Panel Mount DC Jacks
Install the four DC jacks into the case by fitting them through the holes on the side of the enclosure frame.
Panel Mount Switch and DC Jack
Insert the hex nuts onto the DC jacks and tightly fasten. Proceed to panel mount the toggle switch and the 12VDC jack for power input.

Connect Switch and DC Jack
Plug in the toggle switch to the JST-PH connect that is connected to the EN and GND pins on the Feather. Plug in the 12VDC power input to the JST-PH connector that is wired to the power and ground rails on the Perma-Proto PCB.

Connect DC Jacks for Solenoids
Plug in the DC jacks for the solenoids to the various JST-PH connectors on the Perma-Proto PCB.

Install Case to Cover
Fit the enclosure frame over the bottom cover with the notch lined up with the USB port on the Feather. Check to ensure all of the wires are fitted inside the case. Firmly press to snap fit together.
Install Top Cover
Place the top cover over the enclosure frame. Check to ensure all of the wires are fitted inside the case. Firmly press to snap fit close.

Mallet Assembly

Solenoid Holder
The solenoids are fitted into these holders and secured with M3 screws.

Install Solenoid
Install the solenoid into a holder by sliding it in with the piston fitting through the center hole. The mounting holes should be lined up with the holes on the side of the holder.
Install Linkage Plate
Place the plate over the solenoid holder with the mounting holes lined up.

Secure Plate to Holder
Insert and fasten 4x M3 x 6mm long screws to secure the plate to the solenoid holder.

Hardware for Base
M3 standoffs are used to elevate the solenoid. Taller standoffs allow for more height clearance. Choose the length of standoff you'll want to use for your setup. In this project, 10mm long M3 standoffs are used.
Install Standoffs
Use M3 screws to install four M3 standoffs to the slots on the side of the base.

Secure Standoffs
Tightly fasten M3 screws to the standoffs.

Install Solenoid Holder to Base
Place the solenoid holder over the standoffs on the base. Line up the slot with the M3 standoffs.
Secure Solenoid Holder to Base
Install and fasten M3 screws over the slots on the solenoid holder.

Mallet Hardware
Super glue the two dome halves to make the head of the mallet. Glue the head to the mallet stick. The mallet is secured to the linkage using an M3 x 16mm long screw and lock nut (with nylon insert).

Secure Mallet Parts
Place the mallet stick over the linkage with the mounting holes line up. Reference the photo for best placement. Insert the M3 x 16mm long screws through both mallet stick and linkage. Insert and tightly fasten the M3 locknut.
Assembled Mallet and Linkage
Inspect the mallet and linkage to ensure installation is correct.

Installing Mallet to Solenoid Plate
The assembled mallet and linkage will be secured to the linkage plate on the solenoid holder using an M3 x 16mm long screw and locknut.

Install Linkage to Solenoid
Fit the linkage through the slit on the end of the solenoid with the large spring. Line up the mounting holes.
Secure Linkage
Insert the M3 x 16mm long screw through the mounting holes. The screw will go through both the solenoid and linkage. Install and tightly fasten the M3 locknut onto the thread of the screw.

Install Linkage to Plate
Line up the center hole on the linkage with the mounting holes on the linkage plate.

Secure Linkage to Plate
Insert the M3 x 16mm long screw through the holes in both the linkage plate and the linkage. Insert and tightly fasten the M3 locknut onto the thread of the screw.
Test Motion
Press the piston on the solenoid down to test out the mallet. It should actuate the mallet down and spring back when released. Loosen the locknut if the motion is too tight from the friction.

Degree of Rotation
The 12V solenoid features a push / pull of 10mm throw. The mallet has about 18° degrees of rotation.

Assembled Solenoid Mallet
Proceed to repeat this process for more solenoids. In this project, we created three sets of solenoid mallets.
2020 Assembly

Solenoid Mounting Parts
Get the parts and hardware ready for the mounting the solenoid to a piece of 2020 aluminum extrusion.

- 2x M4 Slim t-nuts
- 2x M4 x 8mm long screws
- 2x M3 x 10mm long screws
- 2x M3 hex nuts

Install Slim T-Nuts to Brackets
Insert the two M4 x 8mm long screws through the mounting holes on the two brackets. Insert and fasten the M4 slim t-nuts onto the threads of the screws. This will make fitting onto the 2020 extrusion easier later in the assembly.

Secure Brackets to Mounting Plate
Place the brackets onto the slotted holes on the mounting plate. Reference the photo for correct placement.
Installing Mounting Plate to Solenoid
The mounting plate is secured to the solenoid using four M3 x 6mm long machine screws.

Install Mounting Plate
Place the assembled mounting plate over the solenoid and line up the mounting holes. Reference the image for correct orientation.

Secure Mounting Plate to Solenoid
Insert and fasten four M3 x 6mm long screws to secure the mounting plate to the solenoid.
Assembled Solenoid Mount
The solenoid mount is ready to fit onto a piece of 2020 aluminum extrusion. Check to ensure the slim t-nuts are preinstalled onto the brackets.

Hardware for 2020 Base
Get the hardware ready for assembling the base for the 2020 extrusion.

- 2x M5 x 30mm long screws
- 6x M5 hex nuts
- 1x M4 x 8mm long screw
- 1x M4 slim t-nut

Setup Legs for 2020 Base
Insert the two M5 x 30mm long screws into the base feet. Insert two M5 hex nuts onto the threads of each screw. Reference the photo for best placement. These rubber bumper feet (https://adafruit.it/dLG) are stuck on the bottom for better grip.
Install T-Nuts onto 2020 Base
Insert the M4 x 8mm long screw through the hole on the 2020 base. Install and fasten the M4 slim t-nut onto the thread of the screw. This will make it easier to fit onto the 2020 extrusion.

Install Hex Nuts to 2020 Base
Press fit two of the M5 hex nuts onto the recessed holes on the 2020 base.

Install Legs to 2020 Base
Fasten the legs onto the M5 hex nuts that were fitted into the 2020 base. Reference photo for best placement.
Tighten Hex Nuts
Use a wrench or vise grips too tightly fasten the hex nuts. The height of the 2020 mount can be adjusted by twisting the thread to a desired length.

Pair of 2020 Base Mounts
Proceed to create a second 2020 base mount. One will go on each side of 2020 aluminum extrusion.

Install Solenoid Mount to 2020 Extrusion
Fit the brackets from the solenoid to the slots on the 2020 aluminum extrusion. Ensure the slim t-nut is properly fitted through the slots.
Solenoid Wiring
The solenoid cable is fitted through the brackets before installing onto the piece of 2020 aluminum extrusion. This helps keep the wiring hidden inside the slots of the extrusion.

Height Adjustment
The solenoid can be adjusted by loosening the two M3 screws that secure the solenoid holder to the brackets. The slotted holes allow for height adjustments.

Secure 2020 Mounting Brackets
The brackets are able to slide along the 2020 aluminum extrusion. Tighten the two M4 screws on the brackets to secure the solenoid in place.
Installing DC Cable
The 2.1mm barrel connector is fitted through the 2020 base. This allows the cable to be routed outside of the extrusion.

Install Base Mount
Fit the base mount over the end of the 2020 aluminum extrusion. Carefully fit the slim t-nut into the slot. Ensure the cable from the solenoid is not being kinked or pinched.

Assembled 2020 Solenoid
Double check all of the screws are nice and tight. The height of the solenoid and legs can be adjusted to fit over a snare drum or practice pad.

Usage

Setup DAW Software Instruments

The Feather M4 Express will show up as a USB MIDI device when connected to a computer. MIDI software instrument with capabilities to create external MIDI out instruments will need to be setup in order to get the solenoids to fire.
Mac OS – Audio MIDI Setup
The Feather M4 Express will show up in the MIDI studio window in the Audio MIDI setup application. Right-click Edit device to open the properties window. There you can change the device name and ports. Nothing needs to be changed here, just FYI.

Logic Pro X – External MIDI Track
In the track list, right-click and choose "New External MIDI Track". This allows you to assign the Feather M4 Express to a MIDI track. Using a MIDI track allows you to create musical MIDI notes.

MIDI Track Port
In the Inspector panel, under the Track section, select Feather M4 Express from the Port dropdown. This will assign any MIDI notes in the track to the Feather M4 Express.
Keyboard Playing
MIDI notes can be played live using your computer's keyboard. In the top menu, select Window > Show Keyboard. Use the interface to trigger music notes, change octaves, velocity, etc. This is a great way to test solenoids and play live.

Create MIDI Notes
In the timeline, right-click and select Create MIDI Region. Write your MIDI notes in the piano roll using the pencil tool. Use the notes that are programmed in CircuitPython code.py file.

- C4 60, C#4 61, D4 62, D#4 63