Lightsaber Prop-Maker RP2040
Created by Ruiz Brothers

https://learn.adafruit.com/lightsaber-rp2040

Last updated on 2023-08-29 04:56:33 PM EDT
# Table of Contents

## Overview
- An All-in-one Lightsaber
- Color Cycle
- Streamlined Assembly
- Open Design
- Parts
- Non-Adafruit Parts

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

## CAD Files
- 3D Printed Parts
- Parts List
- 3D Printing Service
- Resin Parts
- Build Volume
- Design Source Files

## CircuitPython
- CircuitPython Quickstart
- Safe Mode
- Flash Resetting UF2

## Code the Lightsaber
- Upload the Code and Libraries to the RP2040 Prop-Maker Feather
- How the CircuitPython Code Works
- Sound FX
- Button, NeoPixels, and Accelerometer
- RGB LED
- The Loop
- Default Mode
- Lightsaber Battle
- Power Down
- Choose Your Color

## Wiring
- Button Wires
- Wiring Button Switch
- Wiring RGB LED
- Wired Button
- Slide Switch Wire
- Wiring Slide Switch
- Wired Slide Switch
- Speaker Wires
- Install Speaker
- Connect Slide Switch
- Connected Slide Switch
- Connect RGB Cable
- Solder RGB Cable
Blade Assembly

- Blade Tips
- Blade Parts
- Polycarbonate Tube
- Corrugated Plastic Setup
- NeoPixel Strip Setup
- Remove Sheathing
- JST Cable
- Solder JST Cable
- NeoPixel Cable
- Cut NeoPixel Strip
- Setup NeoPixel Strip
- Attaching NeoPixel Strip
- Secured NeoPixel Strip
- Install NeoPixel

Assembly

- Battery Holder
- Battery Holder
- Feather Holder
- Speaker Mount
- Install Slide Switch
- Cable Setup
- Install Feather
- Feather Wires
- Connect Speaker
- Blade Holder
- Install Blade Holder
- NeoPixel Cable Extension
- Install Button
- Hilt Wires
- Connect RGB LED
- Connect Wires
- Install Battery Holder
- Install PCB Holder
- Installed PCB Holder
- Secure Button
- Test Circuit
- Install Blade Emitter
- Turn On and Off
- Install Pommel
- Final Build
Overview

An All-in-one Lightsaber

You can build a lightsaber with the Adafruit RP2040 PropMaker Feather. The all-in-one board has everything you need for a fully featured prop. The on-board I2S audio amplifier features digital audio for excellent sound quality. Screw terminals make it easy to connect components like buttons, speakers, and NeoPixels.

Color Cycle

Change the color of the blade by pressing and holding the push button. Press the button again to cycle through different colors. A looping sound effect lets you know when you are in color changing mode.

Streamlined Assembly

Building a lightsaber just got a whole lot better, thanks to the RP2040 PropMaker Feather. Less soldering and better access to the USB port makes for a much better building experience. No hardware screws necessary, parts snap fit and fasten together.
Open Design

The design is open-source and the CAD files are available to download so you can make your own modifications.

If you choose to use an FDM-based machine, no support material is needed. Resin-based printers create exceptional parts that look great.

Parts

Adafruit RP2040 Prop-Maker Feather with I2S Audio Amplifier
The Adafruit Feather series gives you lots of options for a small, portable, rechargeable microcontroller board. By picking a feather and stacking on a FeatherWing you can create...
https://www.adafruit.com/product/5768

Rugged Metal Pushbutton - 16mm 6V RGB Momentary
By popular demand, we now have these buttons with a full color RGB LED ring light! These chrome-plated metal buttons are rugged, but certainly not lacking in flair. Simply drill a...
https://www.adafruit.com/product/3350
**Adafruit Mini Skinny NeoPixel Digital RGB LED Strip - 144 LED/m**
So thin. So mini. So teeeeeeny-tiny. It's the 'skinny' version of our classic NeoPixel strips! These NeoPixel strips have 144 digitally-addressable pixel Mini LEDs...
[https://www.adafruit.com/product/2969](https://www.adafruit.com/product/2969)

**Mini Oval Speaker - 8 Ohm 1 Watt**
Hear the good news! This wee speaker is a great addition to any audio project where you need 8 ohm impedance and 1W or less of power. We particularly like...
[https://www.adafruit.com/product/3923](https://www.adafruit.com/product/3923)

**Lithium Ion Cylindrical Battery - 3.7v 2200mAh**
Need a big battery for your project? This lithium-ion battery contains a 2200mAh and a protection circuit that provides over-voltage, under-voltage, and over-current protection. Yet,...
[https://www.adafruit.com/product/1781](https://www.adafruit.com/product/1781)

**Breadboard-friendly SPDT Slide Switch**
These nice switches are perfect for use with breadboard and perfboard projects. They have 0.1" spacing and snap in nicely into a solderless breadboard. They're easy to switch...
1 x 4-pin Cable Set
1.25mm Pitch 4-pin Cable Matching Pair - 40cm long - Molex PicoBlade Compatible
https://www.adafruit.com/product/4722

1 x 3-pin JST Cable - Socket
ST PH 2mm 3-Pin Socket to Color Coded Cable - 200mm
https://www.adafruit.com/product/4046

1 x 3-pin JST Cable - Plug
STEMMA JST PH 2mm 3-Pin to Male Header Cable - 200mm
https://www.adafruit.com/product/3893

1 x USB Type A to Type C Cable
Approximately 1 meter / 3 ft long
https://www.adafruit.com/product/4474

Non-Adafruit Parts

- Polycarbonate Tubes – 1” OD
  - Ultrasaber Blades ()
  - 24” Ultra Edge Heavy Grade ()
  - The Custom Saber Shop blades ()
  - 40” PolyC TransWhite (thick walled) ()

- Coroplast / Plastic Corrugated Sheet – 20in x 30in – 4mm
  - Nitto Double-sided Tape ()
  - Clear Scotch Tape

Circuit Diagram

The diagram below provides a general visual reference for wiring of the components once you get to the Assembly page. This diagram was created using the software package Fritzing ()..

Adafruit Library for Fritzing

Adafruit uses the Adafruit's Fritzing parts library to create circuit diagrams for projects. You can download the library or just grab individual parts. Get the library and parts from GitHub - Adafruit Fritzing Parts ().
Wired Connections

• The NeoPixel strip, button switch, and speaker are each connected to pins on the screw block terminal.
• The slide switch is connected to the EN and GND pins on the Feather.
• A 2200mAh battery is connected to the battery port on the Feather.
• The RGB LED in the button switch has the following connections:
  ◦ R pin to D10 on Feather
  ◦ G pin to D11 on Feather
  ◦ B pin to D12 on Feather
  ◦ + pin to 3V on Feather
3D Printed Parts
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 using PLA filament. Original design source may be downloaded using the links below.

Parts List
A list of the parts need to build the lightsaber.

1x Battery Holder
1x Blade Holder
1x Emitter
6x Grip
1x Hilt
1x PCB Holder
1x Pommel
Optional Parts

6x Grip
1x Blade Tip

Download CAD source
STLs.zip
3D Printing Service
These parts were made using JLCPCB's 3D printing service.

Use the following options for matte black resin parts.

3D Technology: SLA(resin)
Material: Black Resin
Color: Grayish Black
Sanding: Yes

Resin Parts
The blade holder should be printed in translucent resin. Use the following options for the blade holder.

3D Technology: SLA(resin)
Material: 8001 (Translucent)
Color: Translucent

Build Volume
The parts require a 3D printer with a minimum build volume.

80mm (X) x 80mm (Y) x 190mm (Z)
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 boards, displays, connectors and more can be downloaded from the Adafruit CAD parts GitHub Repo.

CircuitPython

CircuitPython is a derivative of MicroPython 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.

CircuitPython Quickstart

Follow this step-by-step to quickly get CircuitPython running on your board.

Download the latest version of CircuitPython for this board via circuitpython.org

Click the link above to download the latest CircuitPython UF2 file.

Save it wherever is convenient for you.
To enter the bootloader, hold down the BOOT/BOOTSEL button (highlighted in red above), and while continuing to hold it (don't let go!), press and release the reset button (highlighted in blue above). Continue to hold the BOOT/BOOTSEL button until the RPI-RP2 drive appears!

If the drive does not appear, release all the buttons, and then repeat the process above.

You can also start with your board unplugged from USB, press and hold the BOOTSEL button (highlighted in red above), continue to hold it while plugging it into USB, and wait for the drive to appear before releasing the button.

A lot of people end up using charge-only USB cables and it is very frustrating! Make sure you have a USB cable you know is good for data sync.
You will see a new disk drive appear called RPI-RP2.

Drag the adafruit_circuitpython_etc.uf2 file to RPI-RP2.

The RPI-RP2 drive will disappear and a new disk drive called CIRCUITPY will appear.

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

Safe Mode

You want to edit your code.py or modify the files on your CIRCUITPY drive, but find that you can't. Perhaps your board has gotten into a state where CIRCUITPY is read-only. You may have turned off the CIRCUITPY drive altogether. Whatever the reason, safe mode can help.
Safe mode in CircuitPython does not run any user code on startup, and disables auto-reload. This means a few things. First, safe mode bypasses any code in boot.py (where you can set CIRCUITPY read-only or turn it off completely). Second, it does not run the code in code.py. And finally, it does not automatically soft-reload when data is written to the CIRCUITPY drive.

Therefore, whatever you may have done to put your board in a non-interactive state, safe mode gives you the opportunity to correct it without losing all of the data on the CIRCUITPY drive.

**Entering Safe Mode**

To enter safe mode when using CircuitPython, plug in your board or hit reset (highlighted in red above). Immediately after the board starts up or resets, it waits 1000ms. On some boards, the onboard status LED (highlighted in green above) will blink yellow during that time. If you press reset during that 1000ms, the board will start up in safe mode. It can be difficult to react to the yellow LED, so you may want to think of it simply as a slow double click of the reset button. (Remember, a fast double click of reset enters the bootloader.)

**In Safe Mode**

If you successfully enter safe mode on CircuitPython, the LED will intermittently blink yellow three times.

If you connect to the serial console, you'll find the following message.

```plaintext
Auto-reload is off.
Running in safe mode! Not running saved code.
CircuitPython is in safe mode because you pressed the reset button during boot.
Press again to exit safe mode.
Press any key to enter the REPL. Use CTRL-D to reload.
```

You can now edit the contents of the CIRCUITPY drive. Remember, your code will not run until you press the reset button, or unplug and plug in your board, to get out of safe mode.

**Flash Resetting UF2**

If your board ever gets into a really weird state and doesn't even show up as a disk drive when installing CircuitPython, try loading this 'nuke' UF2 which will do a 'deep
clean' on your Flash Memory. You will lose all the files on the board, but at least you'll be able to revive it! After loading this UF2, follow the steps above to re-install CircuitPython.

Download flash erasing "nuke" UF2

**Code the Lightsaber**

Once you've finished setting up your RP2040 Prop-Maker Feather with CircuitPython, you can access the code and necessary libraries by downloading the Project Bundle.

To do this, click on the Download Project Bundle button in the window below. It will download to your computer as a zipped folder.

```python
# SPDX-FileCopyrightText: 2023 Liz Clark for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time
import os
import random
import board
import pwmio
import audiocore
import audiobusio
from adafruit_debouncer import Button
from digitalio import DigitalInOut, Direction, Pull
import neopixel
import adafruit_lis3dh
import simpleio

# CUSTOMIZE SENSITIVITY HERE: smaller numbers = more sensitive to motion
HIT_THRESHOLD = 120
SWING_THRESHOLD = 130
RED = (255, 0, 0)
YELLOW = (125, 255, 0)
GREEN = (0, 255, 0)
CYAN = (0, 125, 255)
BLUE = (0, 0, 255)
PURPLE = (125, 0, 255)
WHITE = (255, 255, 255)
COLORS = [RED, YELLOW, GREEN, CYAN, BLUE, PURPLE, WHITE]
SABER_COLOR = 3
CLASH_COLOR = 6

# enable external power pin
# provides power to the external components
external_power = DigitalInOut(board.EXTERNAL_POWER)
external_power.direction = Direction.OUTPUT
external_power.value = True

wavs = []
for filename in os.listdir('/sounds'):
    if filename.lower().endswith('.wav') and not filename.startswith('.'):  
        wavs.append('/sounds/' + filename)

wavs.sort()
print(wavs)
print(len(wavs))
```

©Adafruit Industries
audio = audiobusio.I2SOut(board.I2S_BIT_CLOCK, board.I2S_WORD_SELECT, board.I2S_DATA)

def play_wav(num, loop=False):
    """
    Play a WAV file in the 'sounds' directory.
    :param name: partial file name string, complete name will be built around
      this, e.g. passing 'foo' will play file 'sounds/foo.wav'.
    :param loop: if True, sound will repeat indefinitely (until interrupted
      by another sound).
    """
    try:
        n = wavs[num]
        wave_file = open(n, "rb")
        wave = audiocore.WaveFile(wave_file)
        audio.play(wave, loop=loop)
    except:  # pylint: disable=bare-except
        return

# external button
pin = DigitalInOut(board.EXTERNAL_BUTTON)
pin.direction = Direction.INPUT
pin.pull = Pull.UP
switch = Button(pin, long_duration_ms = 1000)
switch_state = False

# external neopixels
num_pixels = 100
pixels = neopixel.NeoPixel(board.EXTERNAL_NEOPIXELS, num_pixels, auto_write=True)
pixels.brightness = 0.8

# onboard LIS3DH
i2c = board.I2C()
int1 = DigitalInOut(board.ACCELEROMETER_INTERRUPT)
lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, int1=int1)
lis3dh.range = adafruit_lis3dh.RANGE_2_G
lis3dh.set_tap(1, HIT_THRESHOLD)

red_led = pwmio.PWMOut(board.D10)
green_led = pwmio.PWMOut(board.D11)
blue_led = pwmio.PWMOut(board.D12)

def set_rgb_led(color):
    # convert from 0-255 (neopixel range) to 65535-0 (pwm range)
    redLed.duty_cycle = int(simpleio.map_range(color[0], 0, 255, 65535, 0))
    greenLed.duty_cycle = int(simpleio.map_range(color[1], 0, 255, 65535, 0))
    blueLed.duty_cycle = int(simpleio.map_range(color[2], 0, 255, 65535, 0))

set_rgb_led(COLORS[SABER_COLOR])

mode = 0
swing = False
hit = False

while True:
    switch.update()
    # startup
    if mode == 0:
        print(mode)
        play_wav(0, loop=False)
        for i in range(num_pixels):
            pixels[i] = COLORS[SABER_COLOR]
            pixels.show()
            time.sleep(1)
        play_wav(1, loop=True)
        mode = 1
        # default
        elif mode == 1:
            x, y, z = lis3dh.acceleration
accel_total = x * x + z * z
if lis3dh.tapped:
    print("tapped")
    mode = "hit"
elif accel_total >= SWING_THRESHOLD:
    print("swing")
    mode = "swing"
if switch.short_count == 1:
    mode = 3
if switch.long_press:
    audio.stop()
    play_wav(19, loop=True)
    print("change color")
    mode = 5
    # clash or move
elif mode == "hit":
    audio.stop()
    play_wav(random.randint(3, 10), loop=False)
    while audio.playing:
        pixels.fill(WHITE)
        pixels.show()
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        play_wav(1, loop=True)
        mode = 1
elif mode == "swing":
    audio.stop()
    play_wav(random.randint(11, 18), loop=False)
    while audio.playing:
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        play_wav(1, loop=True)
        mode = 1
    # turn off
elif mode == 3:
    audio.stop()
    play_wav(2, loop=False)
    for i in range(99, 0, -1):
        pixels[i] = (0, 0, 0)
        pixels.show()
        time.sleep(1)
    external_power.value = False
    mode = 4
    # go to startup from off
elif mode == 4:
    if switch.short_count == 1:
        external_power.value = True
        mode = 0
    # change color
elif mode == 5:
    if switch.short_count == 1:
        SABER_COLOR = (SABER_COLOR + 1) % 6
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        set_rgb_led(COLORS[SABER_COLOR])
    if switch.long_press:
        play_wav(1, loop=True)
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        set_rgb_led(COLORS[SABER_COLOR])
        mode = 1
Upload the Code and Libraries to the RP2040 Prop-Maker Feather

After downloading the Project Bundle, plug your RP2040 Prop-Maker Feather into the computer's USB port with a known good USB data+power cable. You should see a new flash drive appear in the computer's File Explorer or Finder (depending on your operating system) called CIRCUITPY. Unzip the folder and copy the following items to the RP2040 Prop-Maker Feather's CIRCUITPY drive.

- lib folder
- sounds folder
- code.py

Your RP2040 Prop-Maker Feather CIRCUITPY drive should look like this after copying the lib folder, sounds folder and the code.py file.
How the CircuitPython Code Works

At the top of the code, you can customize a few attributes for your lightsaber. `HIT_THRESHOLD` and `SWING_THRESHOLD` affect the force needed to trigger a hit or swing for the lightsaber. The `COLORS` array has all of the available colors for the NeoPixels. You can change the value of `SABER_COLOR` to match the index for the color that you want your lightsaber to be. `CLASH_COLOR` is the color that the lightsaber turns when a hit is detected.

```python
# CUSTOMIZE SENSITIVITY HERE: smaller numbers = more sensitive to motion
HIT_THRESHOLD = 120
SWING_THRESHOLD = 130
RED = (255, 0, 0)
YELLOW = (125, 255, 0)
GREEN = (0, 255, 0)
CYAN = (0, 125, 255)
BLUE = (0, 0, 255)
PURPLE = (125, 0, 255)
WHITE = (255, 255, 255)
COLORS = [RED, YELLOW, GREEN, CYAN, BLUE, PURPLE, WHITE]
SABER_COLOR = 3
CLASH_COLOR = 6
```

Sound FX

Audio is played back through the onboard I2S amp on the Feather. The `play_wav()` function opens a WAV file from the sounds folder and then plays it.

```python
wavs = []
for filename in os.listdir('/sounds'):
    if filename.lower().endswith('.wav') and not filename.startswith('. '):
        wavs.append('/sounds/' + filename)
wavs.sort()
print(wavs)
print(len(wavs))

audio = audiobusio.I2SOut(board.I2S_BIT_CLOCK, board.I2S_WORD_SELECT, board.I2S_DATA)

def play_wav(num, loop=False):
    ""
    Play a WAV file in the 'sounds' directory.
    :param name: partial file name string, complete name will be built around
                 this, e.g. passing 'foo' will play file 'sounds/foo.wav'.
    :param loop: if True, sound will repeat indefinitely (until interrupted
                 by another sound).
    ""
    try:
        n = wavs[num]
        wave_file = open(n, "rb")
        wave = audiocore.WaveFile(wave_file)
        audio.play(wave, loop=loop)
    except:  # pylint: disable=bare-except
        # pylint: disable=unnecessary-pass
        pass
        return
```
Button, NeoPixels, and Accelerometer

The button and NeoPixels are connected to the external pins in the terminal block. The button uses the Debounce library so that long press and short press can be monitored in the loop. The onboard LIS3DH accelerometer is used to read movement and tap detection for the swing and hit functionality.

```python
# external button
pin = DigitalInOut(board.EXTERNAL_BUTTON)
pin.direction = Direction.INPUT
pin.pull = Pull.UP
switch = Button(pin, long_duration_ms = 1000)
switch_state = False

# external neopixels
num_pixels = 100
pixels = neopixel.NeoPixel(board.EXTERNAL_NEOPIXELS, num_pixels, auto_write=True)
pixels.brightness = 0.8

# onboard LIS3DH
i2c = board.I2C()
int1 = DigitalInOut(board.ACCELEROMETER_INTERRUPT)
lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, int1=int1)
lis3dh.range = adafruit_lis3dh.RANGE_2_G
lis3dh.set_tap(1, HIT_THRESHOLD)
```

RGB LED

Inside the button is an RGB LED. The color of the LED matches the color of the NeoPixels with the help of the `set_rgb_led()` function.

```python
red_led = pwmio.PWMOut(board.D10)
green_led = pwmio.PWMOut(board.D11)
blue_led = pwmio.PWMOut(board.D12)

def set_rgb_led(color):
    # convert from 0-255 (neopixel range) to 65535-0 (pwm range)
    red_led.duty_cycle = int(simpleio.map_range(color[0], 0, 255, 65535, 0))
green_led.duty_cycle = int(simpleio.map_range(color[1], 0, 255, 65535, 0))
blue_led.duty_cycle = int(simpleio.map_range(color[2], 0, 255, 65535, 0))

set_rgb_led(COLORS[SABER_COLOR])
```

The Loop

At the top of the loop, the button is monitored for any change in state with `switch.update()`. The rest of the loop is a series of states defined by the value of `mode`. When `mode` is 0, the lightsaber boots up by playing the start-up sound and having the NeoPixels light-up one by one.
switch.update()
# startup
if mode == 0:
    print(mode)
    play_wav(0, loop=False)
    for i in range(num_pixels):
        pixels[i] = COLORS[SABER_COLOR]
pixels.show()
time.sleep(1)
    play_wav(1, loop=True)
    mode = 1

Default Mode

When **mode** is **1**, the lightsaber is idle, playing the idling sound on a loop. The LIS3DH is monitored for any changes that match or exceed the thresholds defined at the top of the code for a hit or swing.

# default
elif mode == 1:
    x, y, z = lis3dh.acceleration
    accel_total = x * x + z * z
    if lis3dh.tapped:
        print("tapped")
        mode = "hit"
    elif accel_total &gt;= SWING_THRESHOLD:
        print("swing")
        mode = "swing"

The button is also monitored for a short press or long press. When a short press is detected, the lightsaber goes into **mode 3**, which is the shutdown mode. When a long press is detected, the lightsaber goes into **mode 5**, which lets you change the color of the lightsaber.

if switch.short_count == 1:
    mode = 3
if switch.long_press:
    audio.stop()
    play_wav(19, loop=True)
    print("change color")
    mode = 5

Lightsaber Battle

If a swing or hit is detected, a randomized matching sound effect is played. In the case of a hit, the NeoPixels change in color to white. After the swing or hit has finished, the idle sound begins playing again on a loop.

# clash or move
elif mode == "hit":
    audio.stop()
    play_wav(random.randint(3, 10), loop=False)
while audio.playing:
    pixels.fill(WHITE)
    pixels.show()
    pixels.fill(COLORS[SABER_COLOR])
    pixels.show()
    play_wav(1, loop=True)
    mode = 1
elif mode == "swing":
    audio.stop()
    play_wav(random.randint(11, 18), loop=False)
    while audio.playing:
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        pixels.fill(COLORS[SABER_COLOR])
        pixels.show()
        play_wav(1, loop=True)
        mode = 1

---

### Power Down

If a short press is detected in idle mode, the lightsaber powers down by playing the shutdown sound and turning off the NeoPixels one by one. The external power pin is also turned off to conserve battery power.

If a short press is detected in this mode, `mode` is set to 4 and the external power pin is turned back on. Then, `mode` is set to 0 to return to the start-up mode.

```python
# turn off
elif mode == 3:
    audio.stop()
    play_wav(2, loop=False)
    for i in range(99, 0, -1):
        pixels[i] = (0, 0, 0)
    pixels.show()
    time.sleep(1)
    external_power.value = False
    mode = 4

# go to startup from off
elif mode == 4:
    if switch.short_count == 1:
        external_power.value = True
        mode = 0
```

---

### Choose Your Color

A long press in idle mode changes the `mode` to 5, which lets you change the color of the NeoPixels. With every short press, the selected index in the `COLORS` array advances by 1. The RGB LED also changes its color to match. You'll use a long press to exit color change mode with your new color and return to idle mode.

```python
# change color
elif mode == 5:
    if switch.short_count == 1:
        SABER_COLOR = (SABER_COLOR + 1) % 6
```
Wiring

Button Wires
Use the 4-pin cable for the RGB LED inside the button.

Create a 2-wire cable using the silicone ribbon cable. Measure and cut to create a wire that is 23cm in length.

Wiring Button Switch
Solder the 2-wire cable to the two unlabeled pins on the button switch.
Wiring RGB LED
Solder the 4-pin cable to the labeled pins on the button. Follow the RGB LED wiring:

Red Wire - R Pin (Red LED)
White Wire - G Pin (Green LED)
Black Wire - B Pin (Blue LED)
Yellow Wire - C+ Pin (Voltage)

Wired Button
Double check the wires have been properly soldered to the pins on the button.

Slide Switch Wire
Use the silicone ribbon cable to create a 2-pin cable. Measure and cut the wire to be 7.5cm in length.
Wiring Slide Switch
Cut one of the three pins on the slide switch, either the far left or right but not the middle. Then, cut the two remaining pins half their length.

Solder the 2-pin cable to the pins on the slide switch.

Wired Slide Switch
Double check the wires have been properly soldered to the pins on the slide switch.

Speaker Wires
Cut the cable from the speaker so the wires are 8cm in length.

Remove a bit of insulation from the two wires and tin them using a bit of solder.

Optionally save the 2-pin connector for another project.
Install Speaker
Peel away the protective backing from the speaker.
Attach the speaker to the bottom of the PCB holder.
Reference the photo for the correct orientation.

Connect Slide Switch
Get the slide switch ready to connect to the Feather.
**Connected Slide Switch**
The two wires are soldered to the pins on the bottom of the Feather.

Solder one wire to the ground pin (G) and the other to the enable pin (EN). It doesn't matter which wire goes to which pin.

**Connect RGB Cable**
Get the 4-pin cable ready to solder to the Feather.
Solder RGB Cable
The 4-pin cable is soldered to the pins on the bottom of the Feather.

Cut the wires from the cable so they’re 10cm in length.

Make the following connections:

Red Wire – Pin #10
White Wire – Pin #11
Black Wire – Pin #12
Yellow Wire – Pin 3V
Blade Assembly

Blade Tips
The polycarbonate tubes may need a separate piece for the tip. Each blade can have a different tip depending on the supplier.

Polycarbonate tubes from Ultrasaber() have blade tip options that come preinstalled.

Optionally 3D print your own tip with a translucent filament or resin using the provided files in the CAD files page.

Blade Parts
Get the reel of NeoPixels, corrugated poster board and polycarbonate tube ready.
Polycarbonate Tube
Check the polycarbonate tube has the inner lining and is ready for use.

Corrugated Plastic Setup
Measure and cut two strips of corrugated plastic so they're the length matches the polycarbonate tube and the width matches the NeoPixel strip.

If it's too short, extend the length with additional strips and attach using scotch tape.
NeoPixel Strip Setup
Remove the NeoPixel strip from the packaging and unspool it from the reel.

Use a hobby knife to remove the hot glue from the ends of the NeoPixel Strip.

Use a soldering iron to desolder the stock cable from the NeoPixel strip.

Remove Sheathing
Carefully remove the sheathing from the NeoPixel Strip.

Use caution when cutting or soldering to avoid injury. Please use appropriate protective equipment.
JST Cable
Get the 3-pin JST cable ready to solder to the NeoPixel strip.

Cut and strip the ends of the wires using wire strippers. Tin the ends using a bit of solder to prevent the strands of wire from fraying.

Solder JST Cable
Attach the wires to the pads on the end of the NeoPixel strip with the Data In pads.

Solder the red wire to (+), white wire to (D) and black wire to (–).

NeoPixel Cable
Double check the wires have been properly soldered to the correct end of the NeoPixel strip.
Cut NeoPixel Strip
Measure the NeoPixel strip so it matches the length of the polycarbonate tube.

Cut the strip using snips across the solder pads.

The strip should have 85 NeoPixel LEDs.

Setup NeoPixel Strip
The NeoPixel strip will be sandwiched in between the two strips of corrugated plastic.
Attaching NeoPixel Strip

Use double-sided tape to attach the NeoPixel strip to one of the strips of corrugated plastic.
Secured NeoPixel Strip
Place the other strip of corrugated plastic over the NeoPixel strip.

Wrap strips of scotch tape over the two strips of corrugated plastic to secure the pieces together.

Install NeoPixel
Carefully slide the assembled NeoPixel strip into the polycarbonate tube.
Assembly

Battery Holder
Get the 2200mah battery and 3D printed battery holder ready.

Slide the battery into the holder so it's fitted about half the length of the battery.

Battery Holder
Connect the battery to the Feather.

If the Feather powers on, use the slide switch to turn it off.
Feather Holder
Get the PCB holder ready for installation.

Orient the PCB holder so the end with the flange is aligned with the Feathers USB-C connector.

Speaker Mount
The speaker features a sticker around the speaker frame. Peel the sticker cover off and adhere to the bottom of the Feather holder.

Install Slide Switch
Press fit the slide switch in between the two standoffs.

Cable Setup
Flip the Feather so the bottom side faces up and group the various cables.
Install Feather
Place the Feather into the PCB holder so the PCB is fitted underneath one of the clips. The standoff pegs should fit into the mounting holes.

Then, slightly flex the PCB holder open to fit the other side of the Feather.

Feather Wires
Ensure the various cables are all facing out towards the Feathers screw terminals.

All of the wires should positioned in between the standoffs.
Connect Speaker
Place the two wires from the speaker into the Feathers screw terminals.

Use a small flat head screwdriver to secure the wires to the terminals.

Blade Holder
Get the 3D printed blade holder ready to install onto the polycarbonate tube.
Install Blade Holder
Insert the 3-pin connector from the NeoPixel strip through the hole of the 3d printed blade holder.

Firmly press the holder onto the polycarbonate tube.

It should have a very snug fit. Ensure the wires are not being kinked.

NeoPixel Cable Extension
Connect the 3-pin JST cable to the matching extension cable.
Install Button
Get the button and hilt ready for installation.

Insert included the rubber ring onto the threading of the button.

Insert the two cables from the button through the hole on the side of the hilt.

Do not fully install the button into the hilt just yet.

Hilt Wires
Insert the 3-pin JST extension cable from the NeoPixel strip through the top of the hilt.

Pull the wires from the button and NeoPixel strip so they're coming out of the bottom of the hilt.

Get the Feather ready to secure the various wires to the screw terminals.
Connect RGB LED
Connect the 4-pin cable from the Feather to the 4-pin cable on the button.

Connect Wires
Insert the wires from the button switch and NeoPixel strip to the corresponding pins on the screw terminals.

Use a screwdriver to tighten the screws and secure the wires in place.

Install Battery Holder
Bundle up the various wires inside the hilt to one side.

Then, carefully insert the battery holder into the hilt with space for the wires to pass through.

Push the battery holder into the hilt while keeping the wires inside the hilt taut.
Install PCB Holder
With the battery holder fully inserted into the hilt, begin fitting the PCB holder into the hilt.

Installed PCB Holder
Push the PCB holder into the hilt until the flange is pressed up against the bottom surface of the hilt.
Secure Button
Adjust the various wires inside the hilt and begin to insert the body of the button into the hole.

Firmly press the button into the hole until it's flush with the hilt.

Ensure the various wires in the hilt are not being kinked.
Test Circuit
Use the slide switch to test the circuit. The Feather should power on. The NeoPixels and RGB LED should light up. Audio should also play from the speaker.
Install Blade Emitter
Position the polycarbonate tube so it's aligned with the top of the hilt.

Fit the blade emitter part over the tip of the polycarbonate tube. Slide it all the way down the tube.

Twist the emitter to screw it onto the threading of the hilt.

Fasten the emitter onto the hilt until they're both tight.

Turn On and Off
Use the slide switch to power the Feather on and off. Leave it on when ready for use.
Install Pommel
Get the pommel and begin fastening onto the bottom of the hilt.

Final Build
Congratulations on building your Lightsaber!
Use the hilt’s button to switch between the on and off modes.
Press and hold the button for 2 seconds to go into color changing mode.
Press and hold for another 2 seconds to exit the color changing mode.