Christmas Soundboard with NeoTrellis M4

Created by Dano Wall

https://learn.adafruit.com/xmas-sound-board

Last updated on 2023-08-29 03:55:20 PM EDT
# Table of Contents

## Overview
- About the NeoTrellis M4
- Parts

## Enclosure Assembly
- Prep
- Elastomer Pads
- Place the Trellis M4
- Frame Layer
- Penultimate Layer
- Back Layer
- Fasteners
- Customization

## Trellis M4 Setup
- CircuitPython
- Libraries
- Troubleshooting
- Next

## Code and Audio Files

## Happy Holidays
- Custom Colors

## Sound Customization
- No Mix & Match on mono and stereo files, please.

## Troubleshooting
Overview

This guide shows you how to build your own soundboard. Soundboards are commonly used for sound effects or to select various pieces of dialog on command, though this one features a set of open-source holiday audio clips for bountiful fun and mischief this holiday season.

This project uses the NeoTrellis M4 to create a soundboard capable of playing any of 32 audio clips when the corresponding button is pressed. Sounds are able to seamlessly interrupt one another, so you can press as messily as you like and this soundboard will keep up with you.

About the NeoTrellis M4

The NeoTrellis M4 is an all-in-one Audio board, ready to become your next synth, soundboard, drum machine, keyboard, or any other invention you'd like to adapt it for. It's powered by the SAMD51, a Cortex M4 core running at 120 MHz, featuring a roomy 512KB of flash and 192KB of SRAM. A separate flash chip provides a full 8MB of space for files and audio clips.

On the front side is a 4x8 grid of elastomer button pads with a NeoPixel nestled in the center of each one. You can read any/all button presses simultaneously thanks to the fully diode'd matrix, and also set each button color to any of 24-bit colors.
Parts

Adafruit NeoTrellis M4 with Enclosure and Buttons Kit Pack
So you've got a cool/witty name for your band, a Soundcloud account, a 3D-printed Daft Punk...
https://www.adafruit.com/product/4020

USB cable - USB A to Micro-B
This here is your standard A to micro-B USB cable, for USB 1.1 or 2.0. Perfect for connecting a PC to your Metro, Feather, Raspberry Pi or other dev-board or...
https://www.adafruit.com/product/592

Pick how you would like your sound output:

USB Powered Speakers
Add some extra boom to your audio project with these powered loudspeakers. We sampled half a dozen different models to find ones with a good frequency response, so you'll get...
https://www.adafruit.com/product/1363
Analog Potentiometer Volume Adjustable TRRS Headset
Most modern headphone sets are purely digital - with three volume control buttons in-line with the cable. These headphones are interesting in that they have an analog volume...
https://www.adafruit.com/product/3959

Or if you have something that takes audio in, you might consider this cable:

3.5mm Male/Male Stereo Cable
Seamlessly transmit high-quality stereo audio with this 3.5mm Male/Male Stereo Cable. Ideal for "passing the AUX cord," or connecting...
https://www.adafruit.com/product/2698

For on the go power with headphones:

1 x USB Battery Pack - 2200 mAh
USB Battery Pack - 2200 mAh Capacity - 5V 1A Output
https://www.adafruit.com/product/1959

or for use with speakers:

1 x USB Battery Pack - 10000mAh
USB Battery Pack - 10000mAh - 2 x 5V outputs
https://www.adafruit.com/product/1566

Be sure you have adequate power as the NeoTrellis (with all those LEDs) draws a decent amount of current and will need power, especially with the LED brightness on higher settings.
Enclosure Assembly

The acrylic enclosure is quick and easy to assemble. Each part is horizontally symmetrical, so there is very little worry about placing them in a "wrong" direction.

Prep

First, you have the satisfying task of peeling off the protective paper from the acrylic pieces. If you peel off all of the paper without tearing them you will have seven weeks of good luck. Do not squander them.
Elastomer Pads

Place both of the button pads into the top piece as shown. Note, there are two pairs of rubber registration nubs which must align with their associated holes on the Trellis M4 board, so match the orientation as seen in the photos.
Place the Trellis M4

Place the Trellis M4 board down onto the elastomer pads with the NeoPixels and button contacts aligned with the buttons as shown. Make sure all of the sixteen registration nubs fit into their associated holes on the PCB.
Frame Layer

Now, place the frame layer which helps hold the board in place.
Penultimate Layer

This next layer has some large cutouts in it to allow for parts clearance.
Back Layer

It's the final layer! Lay it down and get ready for some fastening action.
Fasteners

Place the five nylon M3 screws through their holes from the front of the case toward the back, then thread the nuts onto them and finger tighten.
Your Trellis M4 is ready for use!
Customization

Are you the sort of tinkerer who feels the urge to customize things? We knew it! In case you want to try your hand at building a custom enclosure for your Trellis M4, and make all your other DJ/Producer friends jealous, [here are the CAD files to get you started](#). Use them on a laser cutter to create a unique wooden faceplate. Or, print the top piece out with ill graphics on a color printer, trim the button holes and screw holes, and make a cool skin. You might even want to use the templates as a reference for 3D modelling a case!

It's easy to press the Trellis M4's reset button through the case using a small object such as a headphone plug, chopstick, or 5mm LED. But, if you're doing a lot of development in Arduino and resetting a lot, you may want to make a button extender. Here's a file you can 3D print for that!

![resetbuttonTrellisM4.stl.zip](resetbuttonTrellisM4.stl.zip)

Trellis M4 Setup

First, we'll prep the Trellis M4 for running CircuitPython.

After assembling the case, plug your NeoTrellis into a computer (mac/PC/Linux) and look for a new flash drive in the operating system file explorer/finder program. It should be named CIRCUITPY. If you browse the files on this new drive, you'll see a file named boot_out.txt. If you see this, then you should be set. If not, go to troubleshooting below.
CircuitPython

The NeoTrellis M4 Express will be programmed using CircuitPython, the easy to use language on Adafruit's Express products. No coding is needed if you will just be copying the files in this guide.

If you'd like to know more about CircuitPython, [there is a full getting started guide here](#).

To get the latest version of CircuitPython, please see the [instructions in the NeoTrellis Guide](#).

The code is in text files so you can easily make changes with your favorite text editor. Adafruit suggests using the Mu editor to edit your code and have an interactive REPL in CircuitPython. [You can learn about Mu and its installation in this tutorial](#).

This project requires CircuitPython version 4.0.0 alpha or later which does ship with NeoTrellis. Please do not use an earlier version of CircuitPython or libraries as they will not provide the functionality needed for this project.

Libraries

The code uses several libraries for helper code. Please go to the [CircuitPython libraries releases page](#) and download the zip file for CircuitPython 4.x to your computer hard drive. It will have a name similar to adafruit-circuitpython-bundle-4.x-mpy-20181211.zip but the date will most likely be different (as Adafruit adds more features),

Go into the library zip file and copy the following files to your NeoTrellis CIRCUITPY drive into a subdirectory named /lib:

- Directory adafruit_bus_device
- Directory adafruit_fancyled
- adafruit_adxl34x.mpy
- adafruit_matrixkeypad.mpy
- adafruit_trellism4.mpy
- neopixel.mpy
Troubleshooting

You can use the serial capability in Mu to connect to the NeoTrellis M4 to interact with the CircuitPython prompt. You can see error messages, restart the program, etc.

Problem: I don't see the CIRCUITPY drive when I plug the NeoTrellis M4 into my computer or I see a drive named TRELM4BOOT instead.

Solution: Ensure CircuitPython is installed via this guide.

Problem: The code does not run.

Solution: Check the files on the CIRCUITPY drive. /lib should have the latest CircuitPython 4.x libraries and you have placed code.py in the main (root) directory.

Problem: Not enough disk space for all the files.

Solution: If you have files from another NeoTrellis project, you may have to remove some or all of the other project-specific files to load files specific to this project.

If you use your own phrases, you cannot use long stereo clips or songs for each button, there is not enough flash disk space. You can reduce the number of files below 32. You can trim your clips to make them shorter. You should ensure all your clips are mono instead of stereo.

Problem: My board isn't booting up!

Solution: Make sure your /lib folder is set up with the libraries in the 4.0 latest release.

Problem: I'm not hearing any sounds!

Solution: Ensure you have copied the sound files to the appropriate directories: /xmas holds 32 sound files. The files are in GitHub in a ZIP when you download the code.
Ensure you have headphones or powered speakers plugged into the sound output jack and that the volume is set appropriately. If you have put custom sounds on the NeoTrellis, check that all your files are formatted as Mono 16-bit, 22,050Hz, PCM.

Next

Your Trellis M4 is now prepped to run CircuitPython code. Next, we'll add the sound and code files.

Code and Audio Files

Time to download the software and sounds.

Connect your NeoTrellis M4 to your computer via a known good power+data USB cable. Your computer (PC/Mac/Linux/Chromebook) should indicate in the file explorer/finder a new flash disk drive named CIRCUITPY.

Click on the Download Project Zip button in the top left of the code listing below to get a zip file of all the needed project files. This includes:

- code.py
- color_names.py
- welcome.wav
- /xmas subdirectory (contains 32 pre-made audio clips)

Open the ZIP file in your computer File Explorer/Finder. Plug in your NeoTrellis and open the CIRCUITPY flash drive that appears when NeoTrellis is plugged in.

Copy code.py, color_names.py and welcome.wav to the root/main directory.

Copy the directory of sounds, /xmas to the root directory.

Files that have a period in the first character of the filename may be made on Mac computers, no worries.
import time
import board
import audiioio
import audiocore
import adatafancyled.adafruit_fancyled as fancy
import adafruit_trellism4
from color_names import * # pylint: disable= wildcard-import, unused-wildcard-import

PLAY_SAMPLES_ON_START = False

SAMPLE_FOLDER = "/xmas/" # the name of the folder containing the samples
# This soundboard can select up to *32* sound clips! each one has a filename
# which will be inside the SAMPLE_FOLDER above, and a *color* in a tuple()
SAMPLES = ["01.wav", RED],
("02.wav", RED),
("03.wav", RED),
("04.wav", GREEN),
("05.wav", GREEN),
("06.wav", RED),
("07.wav", RED),
("08.wav", RED),
("09.wav", RED),
("10.wav", RED),
("11.wav", GREEN),
("12.wav", GREEN),
("13.wav", GREEN),
("14.wav", GREEN),
("15.wav", RED),
("16.wav", RED),
("17.wav", RED),
("18.wav", GREEN),
("19.wav", GREEN),
("20.wav", GREEN),
("21.wav", GREEN),
("22.wav", GREEN),
("23.wav", GREEN),
("24.wav", RED),
("25.wav", RED),
("26.wav", RED),
("27.wav", RED),
("28.wav", ORANGE),
("29.wav", ORANGE),
("30.wav", RED),
("31.wav", RED),
("32.wav", RED)]

# For the intro, pick any number of colors to make a fancy gradient!
INTRO_SWIRL = [RED, GREEN, BLUE]
# the color for the selected sample
SELECTED_COLOR = TEAL

# Our keypad + neopixel driver
trellis = adafruit_trellism4.TrellisM4Express(rotation=0)

# Play the welcome wav (if its there)
with audiioio.AudioOut(board.A1, right_channel=board.A0) as audio:
    try:
        f = open("welcome.wav", "rb")
        wave = audiocore.WaveFile(f)
        audio.play(wave)
        swirl = 0 # we'll swirl through the colors in the gradient
        while audio.playing:
            for i in range(32):
                palette_index = ((swirl+i) % 32) / 32
                color = fancy.palette_lookup(INTRO_SWIRL, palette_index)
                # display it!
                trellis.pixels[(i%8, i//8)] = color.pack()
                swirl += 1
time.sleep(0.005)
f.close()
# Clear all pixels
trellis.pixels.fill(0)
# just hold a moment
time.sleep(0.5)
except OSError:
    # no biggie, they probably deleted it
    pass

# Parse the first file to figure out what format its in
channel_count = None
bits_per_sample = None
sample_rate = None
with open(SAMPLE_FOLDER+SAMPLES[0][0], "rb") as f:
    wav = audiocore.WaveFile(f)
    print("%d channels, %d bits per sample, %d Hz sample rate " %
          (wav.channel_count, wav.bits_per_sample, wav.sample_rate))

    # Audio playback object - we'll go with either mono or stereo depending on
    # what we see in the first file
    if wav.channel_count == 1:
        audio = audioio.AudioOut(board.A1)
    elif wav.channel_count == 2:
        audio = audioio.AudioOut(board.A1, right_channel=board.A0)
    else:
        raise RuntimeError("Must be mono or stereo waves!")

# Clear all pixels

trellis.pixels.fill(0)

# turn on maybe play all of the buttons
for i, v in enumerate(SAMPLES):
    filename = SAMPLE_FOLDER+v[0]
    try:
        with open(filename, "rb") as f:
            wav = audiocore.WaveFile(f)
            print(filename, "%d channels, %d bits per sample, %d Hz sample rate " %
                    (wav.channel_count, wav.bits_per_sample, wav.sample_rate))
            if wav.channel_count != channel_count:
                pass
            if wav.bits_per_sample != bits_per_sample:
                pass
            if wav.sample_rate != sample_rate:
                pass
            trellis.pixels[(i%8, i//8)] = v[1]
            if PLAY_SAMPLES_ON_START:
                audio.play(wav)
                while audio.playing:
                    pass
    except OSError:
        # File not found! skip to next
        pass

def stop_playing_sample(playback_details):
    print("playing: ", playback_details)
    audio.stop()
    trellis.pixels[playback_details['neopixel_location']] = playback_details['neopixel_color']
    playback_details['file'].close()
    playback_details['voice'] = None
    current_press = set()
    currently_playing = {'voice' : None}
    last_samplenum = None
    while True:
        pressed = set(trellis.pressed_keys)
        if pressed:
# print("Pressed:", pressed)

just_pressed = pressed - current_press
just_released = current_press - pressed

# if just_pressed:
#    print("Just pressed", just_pressed)
for down in just_pressed:
    sample_num = down[1]*8 + down[0]
    print(sample_num)
    try:
        filename = SAMPLE_FOLDER+SAMPLES[sample_num][0]
        f = open(filename, "rb")
        wav = audiocore.WaveFile(f)

        # is something else playing? interrupt it!
        if currently_playing['voice'] != None:
            print("Interrupt")
            stop_playing_sample(currently_playing)

        trellis.pixels[down] = SELECTED_COLOR
        audio.play(wav)
        # voice, neopixel tuple, color, and sample, file handle
        currently_playing = {
            'voice': 0,
            'neopixel_location': down,
            'neopixel_color': SAMPLES[sample_num][1],
            'sample_num': sample_num,
            'file': f}
    except OSError:
        pass # File not found! skip to next

# if just_released:
#    print("Just released:", just_released)

# check if any samples are done
if not audio.playing and currently_playing['voice'] != None:
    stop_playing_sample(currently_playing)

time.sleep(0.01) # a little delay here helps avoid debounce annoyances

current_press = pressed

---

Happy Holidays

Your soundboard is now ready to make some noise! Have fun making weird holiday music, or interjecting sound effects into your conversations.
Custom Colors

You can change the soundboard to display any 4x8 scene you want.

Open a file using the Mu editor or a text file editor of your choice.

You can do this by changing the code.py file to assign specific colors to any of the 32 buttons in the array. You can also edit the color_names.py file to add or change colors if you want a new palette to work with.
Sound Customization

Sound clips have been trimmed and edited from the following free audio files:

- 01 "Nov. 24, 2006 Parade" by daveincamas
- 02 "Metal poles" by Percy Duke
- 03 "Ice skating" by inchadney
- 04 "Beads being shaken to the Christmas Bells melody" by monnie101
- 05 "Trenet de Nadal" by paubg_pou
- 06 "Merry Christmas" by davidbain
- 07 "Christmas pagent" jgeralyn
- 08 "Bells ringing at Notre-Dame" by Mxsmanic
- 09 "Pirotecnia navidad" by liban
- 10 "Opening a Christmas present" by escortmarius
- 11 "Spinning a dreidel" by Halleck
- 12 "Merry christmas and a happy new year" by sovy
- 13 "Brass band plays Christmas songs" by dobroide
- 14 "Church bells" by vonfleisch
- 15 "Fast sleigh bells" by jcdecha
- 16 "Jingle bells" by xtrgamr
- 17 "Musical Snow Globe - We Wish You A Merry Christmas" by mhtaylor67
- 18 "The bells of Santa Claus" by BrianJamesLong
- 19 "Merry Christmas" by maurolupo
- 20 "Hanukkah" by tim.kahn
- 21 "Santa Claus is coming to town" by dwightsabeast
- 22 "Ambient Christian bells" by Npeo
- 23 "We Wish You a Merry Christmas music box" by MaestroALF

©Adafruit Industries
If you'd like to add or update the sounds on your soundboard, more audio clips can be downloaded from sites like http://soundbible.com/ or https://freesound.org/.

You will need to convert the files to the appropriate format for microcontroller use. See this guide on how to do that.

When you have a file converted, you can name it like the files in the xmas directory. For example, say you downloaded a clip and you wanted it to be activated on the second row, end key. That is the 16th key counting from upper left. Rename 16.wav to 16-old.wav and name your clip 16.wav and copy it into the xmas folder on the CIRCUITPY drive. Now your sound will play when that button is pressed.

Note for keys 1 to 9, add a leading zero, like 01.wav, 02.wav, etc. so all filenames are two digits.

No Mix & Match on mono and stereo files, please.

Make sure your audio files are exported as 16-bit PCM WAV at 22,050 Hz and they are all Stereo or all Mono -no mix and match!

---

Troubleshooting

Project produces error:

    AttributeError: 'WaveFile' object has no attribute 'channel_count'

You should update the NeoTrellis firmware as described in https://learn.adafruit.com/adafruit-neotrellis-m4/circuitpython.