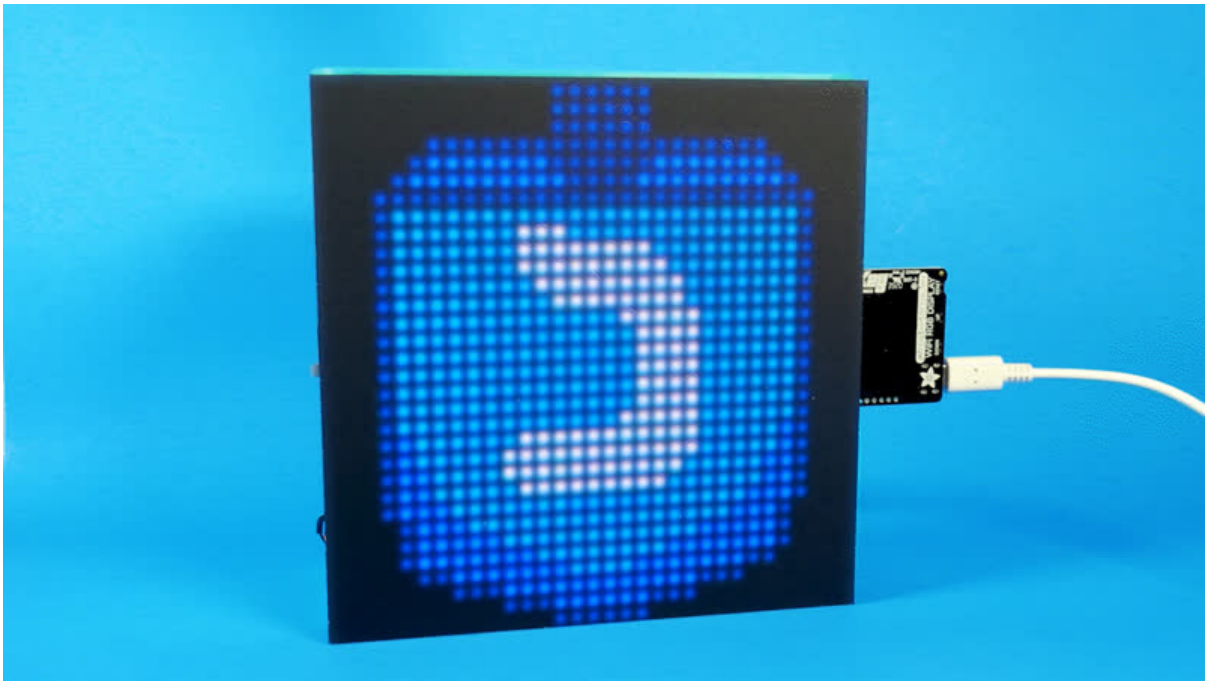
RGB Matrix Dreidel Game

Created by Liz Clark



https://learn.adafruit.com/rgb-matrix-dreidel-game

Last updated on 2024-06-03 03:30:17 PM EDT

Table of Contents

Overview 3

- Parts

Circuit Diagram 8

- Audio Amplifier Circuit
- I2S Audio Circuit

3D Printing 11

Install CircuitPython 12

- Set up CircuitPython Quick Start!
- Further Information

Coding the RGB Matrix Dreidel Game 14

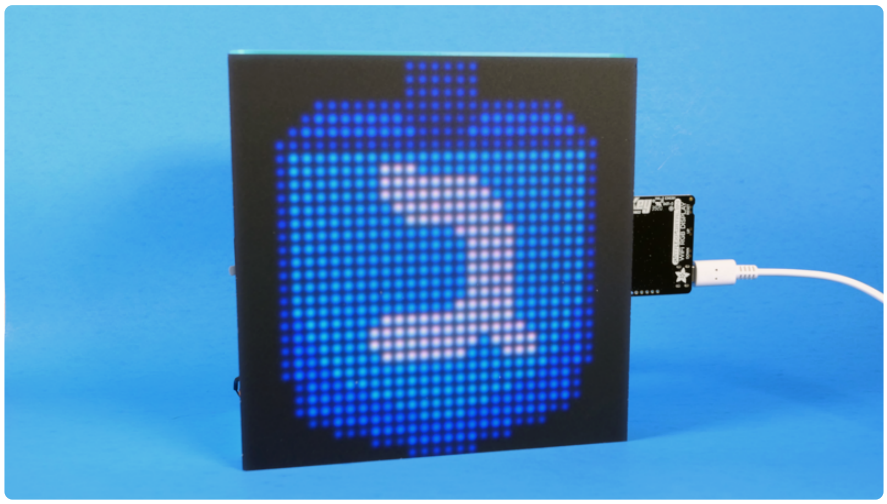
- I2S Audio Version Changes
- Upload the Code, Audio File, Bitmap and Libraries to the MatrixPortal M4
- How the CircuitPython Code Works

Wiring 21

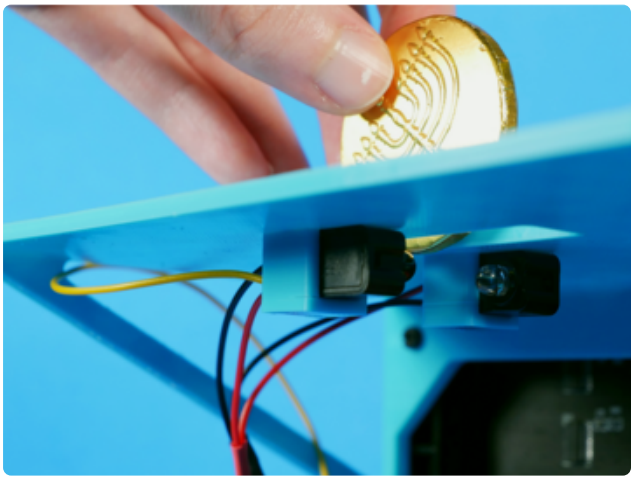
Assembly 27

Play Dreidel! 33

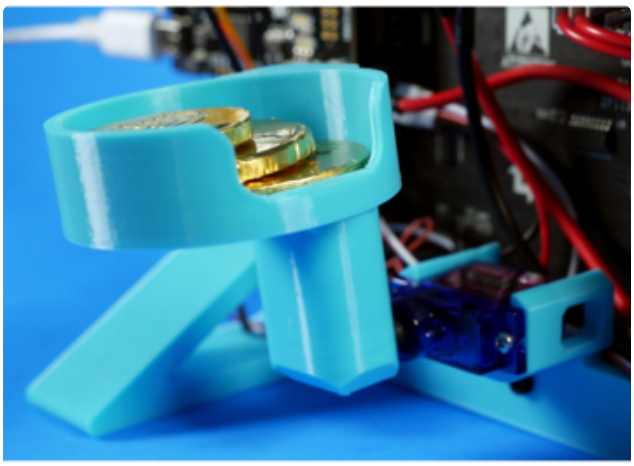
Overview



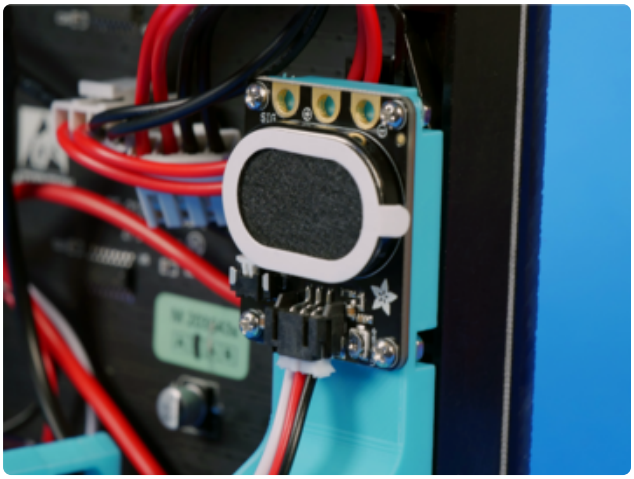
In this project, you can build your own electronic dreidel game, complete with a servo motor and festive music, to celebrate Hanukkah. To play, drop chocolate coins into the slot at the top of the matrix. The RGB matrix will spin the dreidel while playing the classic dreidel song. If you roll gimel, you win! All of the chocolate coins will spill out for you.



An IR breakbeam sensor is located at the top of the matrix to detect when a chocolate coin is dropped. This triggers the dreidel to spin.



A servo motor is attached to the chocolate coin cup. The servo tilts the cup when the dreidel sprite sheet lands on gimel.

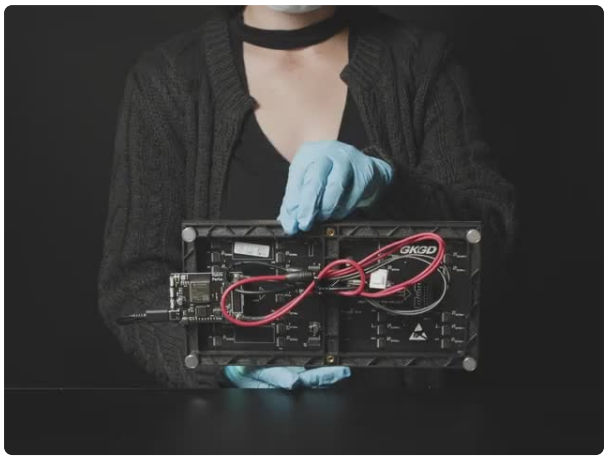


A STEMMA speaker plays the dreidel song while the RGB matrix spins the dreidel.

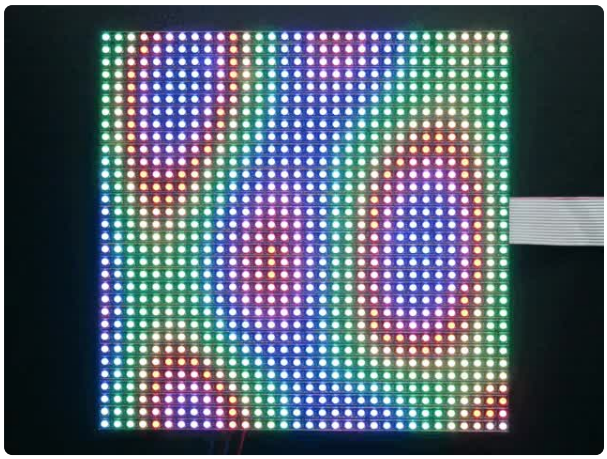
If you've ever played dreidel before, you'll note that this version is slightly modified since it doesn't account for rolling hei to get half of the pot. A servo dumping out the entire pot when you roll gimel should hopefully make up for that though.

Parts

This project has a couple of different way to wire up the audio depending on the microcontroller that you are using.

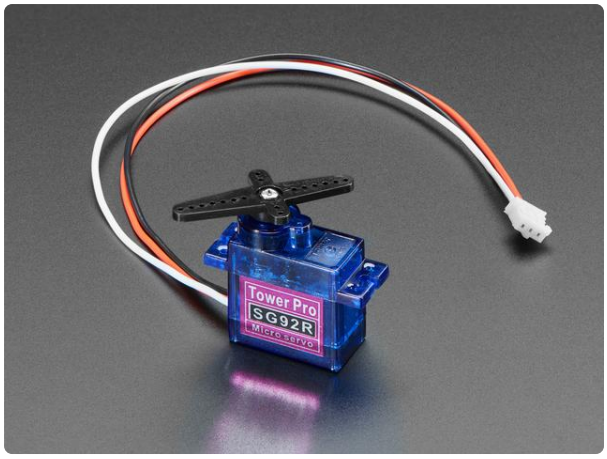


Adafruit Matrix Portal - CircuitPython Powered Internet Display
Folks love our wide selection of RGB matrices and accessories, for making custom colorful LED displays... and our RGB Matrix Shields...
https://www.adafruit.com/product/4745

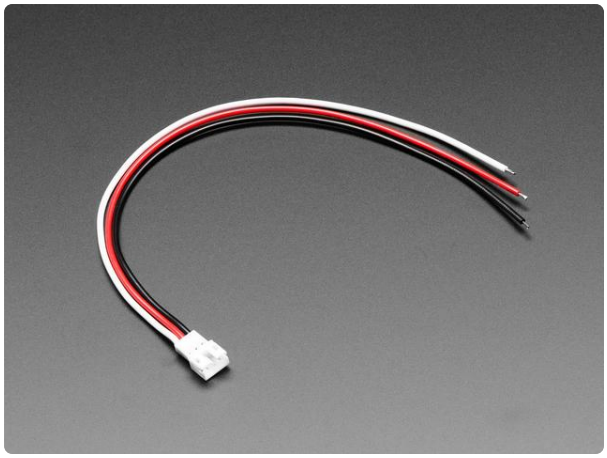


32x32 RGB LED Matrix Panel - 5mm Pitch
Bring a little bit of Times Square into your home with this sweet 32 x 32 square RGB LED matrix panel. These panels are normally used to make video walls, here in New York we see them...
https://www.adafruit.com/product/2026

[IR Break Beam Sensor with Premium Wire Header Ends - 5mm LEDs]
Infrared (IR) break-beam sensors are a simple way to detect motion. They work by having an emitter side that sends out a beam of human-invisible IR light, then a receiver across the...
https://www.adafruit.com/product/2168

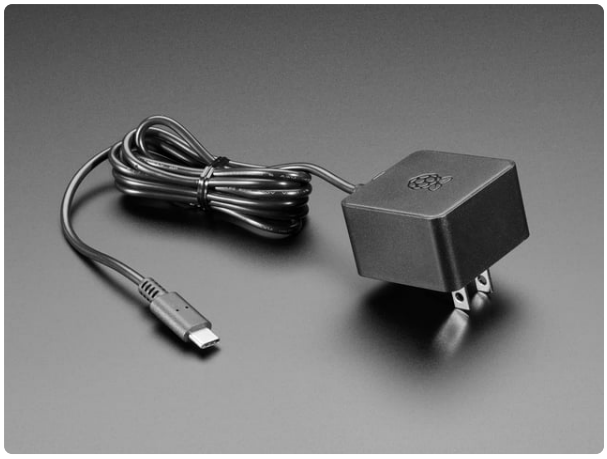


[Micro Servo with 3-pin JST PH 2mm Cable - TowerPro SG92R]
This tiny little servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds you're used to but smaller. You can use any...
https://www.adafruit.com/product/4326



[JST PH 2mm 3-Pin Socket to Color Coded Cable - 200mm]
This cable will let you turn a JST PH 3-pin cable socket into 3 individual tinned wires. These are great to match up with our JST 3-PH cables, for extending and connecting...
https://www.adafruit.com/product/4046

[Black LED Diffusion Acrylic Panel - 10.2" x 5.1"]
nice whoppin' rectangular slab of some lovely black acrylic to add some extra diffusion to your LED Matrix project. This material is 2.6mm (0.1") thick and is made of...
https://www.adafruit.com/product/4749



Official Raspberry Pi Power Supply 5.1V 3A with USB C

The official Raspberry Pi USB-C power supply is here! And of course, we have 'em in classic Adafruit black! Superfast with just the right amount of cable length to get your Pi 4...
https://www.adafruit.com/product/4298

Black Nylon Machine Screw and Stand-off Set – M3 Thread

Totaling 420 pieces, this M3 Screw Set is a must-have for your workstation. You'll have enough screws, nuts, and hex standoffs to fuel...
https://www.adafruit.com/product/4685

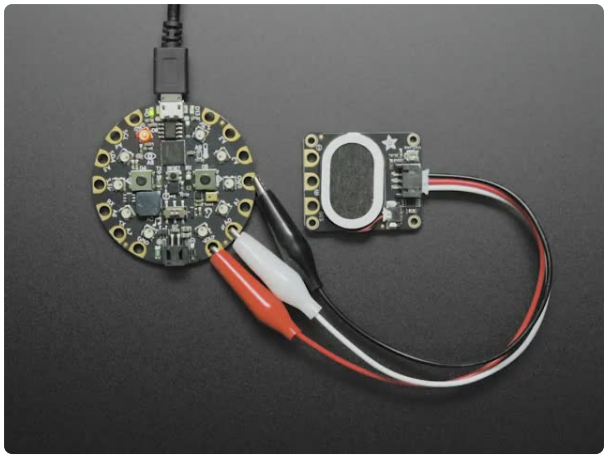
1 x M2 Screws
M2 hardware
https://www.amazon.com/Hilitchi-300-Piece-Phillips-Assortment-stainless/dp/B01NBOD98K/

---

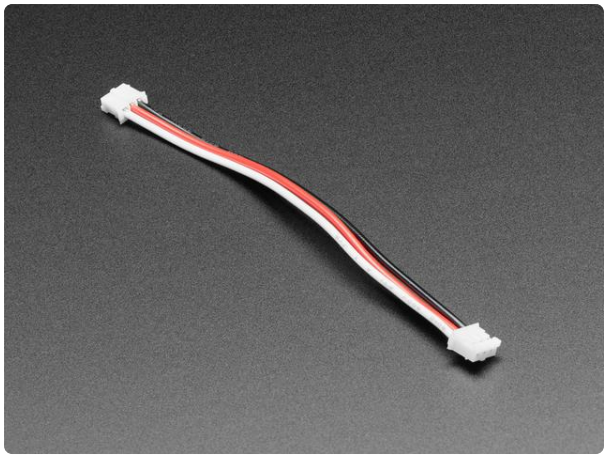
1 x Package of chocolate coins
Chocolate coins aka gelt to play dreidel
https://www.target.com/p/palmer-gifts-of-hanukkah-milk-chocolate-coins-1-5oz/-/A-80021401

Audio Amplifier Version

This version of the project uses the STEMMA Speaker, making it easy to hook up and is suitable for use with the MatrixPortal M4.



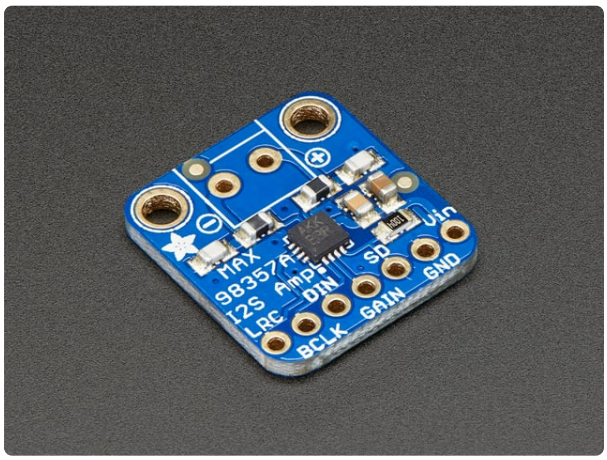
[Adafruit STEMMA Speaker - Plug and Play Audio Amplifier]
Hey, have you heard the good news? With Adafruit STEMMA boards you can easily and safely plug sensors and devices together, like this Adafruit STEMMA Speaker - Plug and Play...
https://www.adafruit.com/product/3885



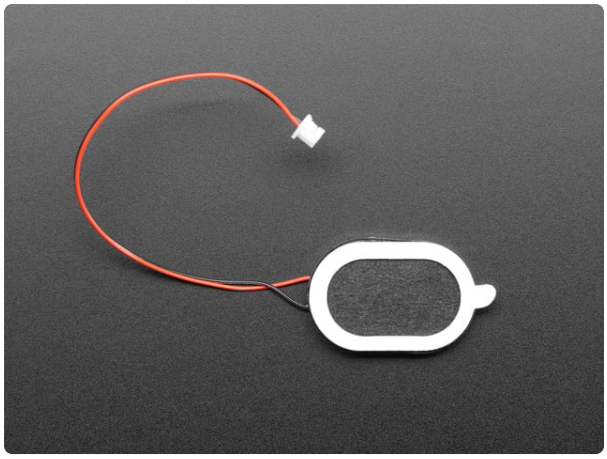
[JST PH 2mm 3-pin Plug-Plug Cable - 100mm long]
This cable is a little over 100mm / 4" long and fitted with JST-PH 3-pin connectors on either end. We dig the solid and compact nature of these connectors and the...
https://www.adafruit.com/product/4336

I2S Audio Version

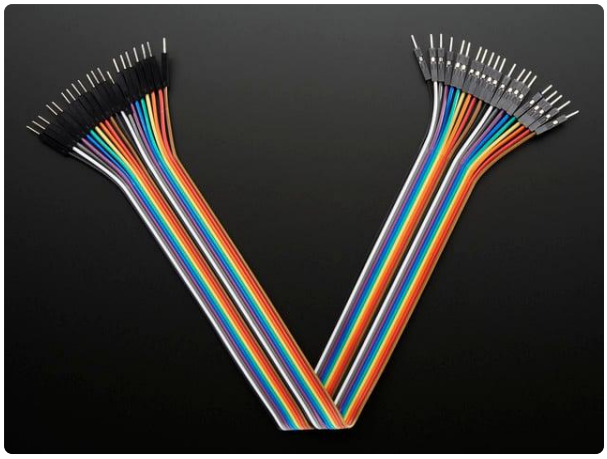
This version of the project uses the MAX98357 and a speaker and is suitable for use with the microcontrollers with only I2S audio such as the ESP32-based boards.



[Adafruit I2S 3W Class D Amplifier Breakout - MAX98357A]
Listen to this good news - we now have an all in one digital audio amp breakout board that works incredibly well with the
https://www.adafruit.com/product/3006



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

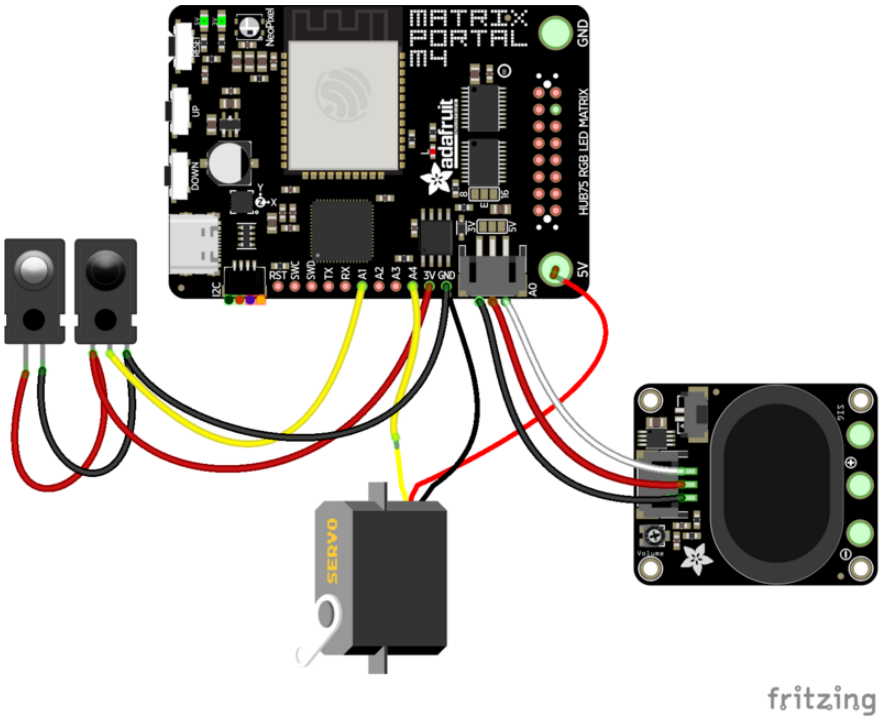


Premium Male/Male Jumper Wires - 20 x 12" (300mm)
These Male/Male Jumper Wires are handy for making wire harnesses or jumpering between headers on PCB's. These premium jumper wires are 12" (300mm) long and come in a...
https://www.adafruit.com/product/1955

Circuit Diagram

Audio Amplifier Circuit

The Amplifier Circuit is the original version of the Dreidel Game and works well with the MatrixPortal M4.



Wiring Connections

STEMMA Speaker

- Speaker **GND** to MatrixPortal M4 **GND**
- Speaker **Power** to MatrixPortal M4 **Power**
- Speaker **SIG** to MatrixPortal M4 **A0**

IR Breakbeam Sensor

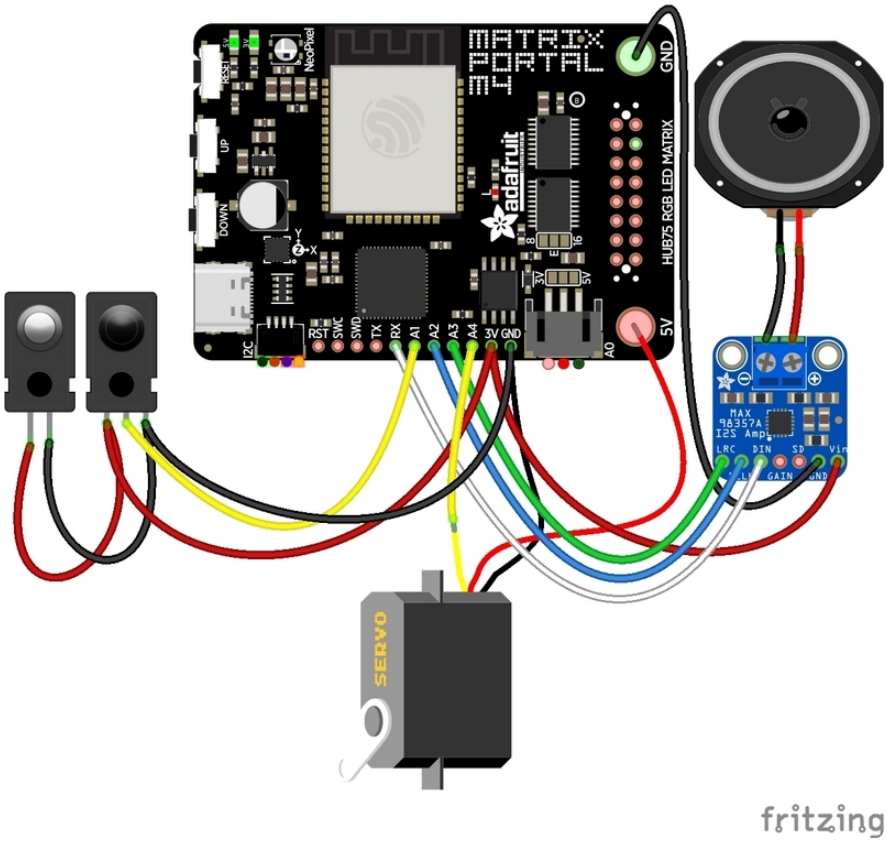
- Receiver **black wire** to MatrixPortal M4 **GND**
- Receiver **red wire** to MatrixPortal M4 **3V**
- Receiver **yellow wire** to MatrixPortal M4 **A1**
- Transmitter **black wire** to MatrixPortal M4 **GND**
- Transmitter **red wire** to MatrixPortal M4 **3V**

Servo

- Servo **black wire** to MatrixPortal M4 **GND**
- Servo **red wire** to MatrixPortal M4 **5V**
- Servo **yellow wire** to MatrixPortal M4 **A4**

I2S Audio Circuit

If you need I2S Audio, you can build this version of the circuit.



MAX98357 I2S Amplifier

- Speaker **black wire** to MAX98357 **output negative**
- Speaker **red wire** to MAX98357 **output positive**
- MAX98357 **GND** to MatrixPortal M4 **GND**
- MAX98357 **Vin** to MatrixPortal M4 **3V**
- MAX98357 **DIN** to MatrixPortal M4 **TX**
- MAX98357 **BCLK** to MatrixPortal M4 **A2**
- MAX98357 **LRC** to MatrixPortal M4 **A3**

IR Breakbeam Sensor

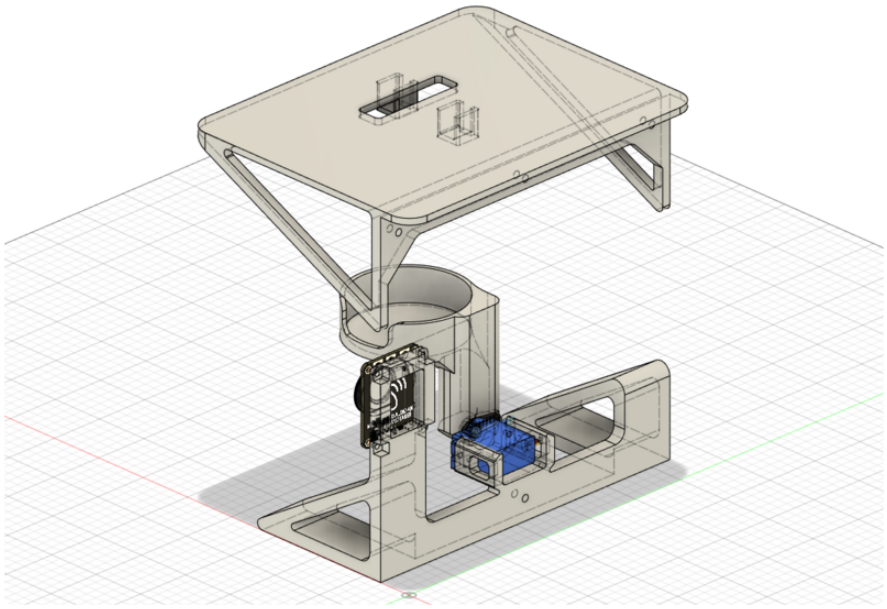
- Receiver **black wire** to MatrixPortal M4 **GND**
- Receiver **red wire** to MatrixPortal M4 **3V**
- Receiver **yellow wire** to MatrixPortal M4 **A1**
- Transmitter **black wire** to MatrixPortal M4 **GND**
- Transmitter **red wire** to MatrixPortal M4 **3V**

Servo

- Servo **black wire** to MatrixPortal M4 **GND**
- Servo **red wire** to MatrixPortal M4 **5V**
- Servo **yellow wire** to MatrixPortal M4 **A4**

3D Printing

The RGB Matrix Dreidel Game uses some 3D printed parts to complete the build. All of the parts can print without supports.



It consists of three parts:

- **matrixDreidel_roof**
- **matrixDreidel_legs**
- **matrixDreidel_bucket**

The files can be downloaded directly here or through Thingiverse.

matrixDreidel-STLs.zip

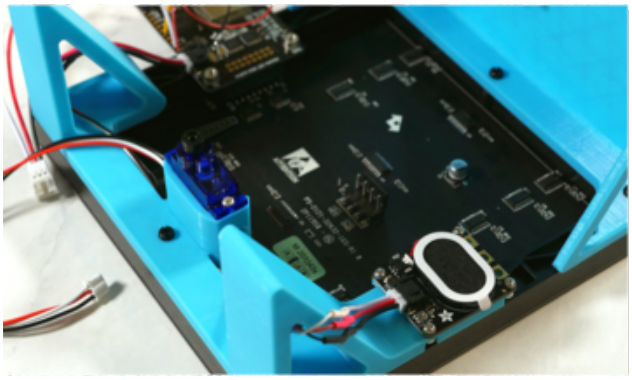
https://adafru.it/19EZ

Thingiverse download

https://adafru.it/WRe

Download CAD Source

https://adafru.it/19F0



The roof mounts to the top of the RGB matrix and has mounts for the breakbeam sensor. The sensors are placed on either side of the chocolate coin slot.

The legs mount to the bottom of the RGB matrix and allow it to stand up freely. Additionally, it has mounts for the servo motor and the STEMMA speaker.

The bucket attaches to the servo motor horn and will catch the chocolate coins as they're dropped in through the slot at the top.

Install CircuitPython

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.

Set up CircuitPython Quick Start!

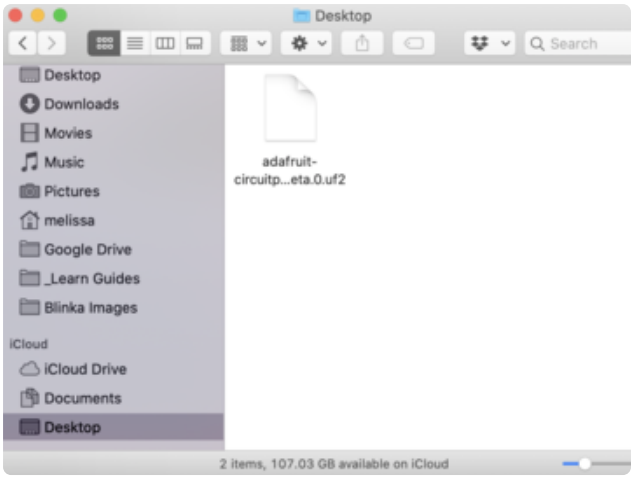
Follow this quick step-by-step for super-fast Python power :)

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

https://adafru.it/Nte

Further Information

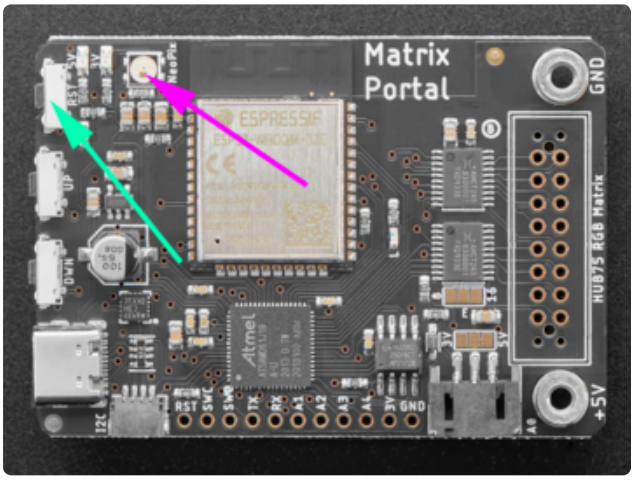
For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafru.it/Amd).



Click the link above and download the latest UF2 file.

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

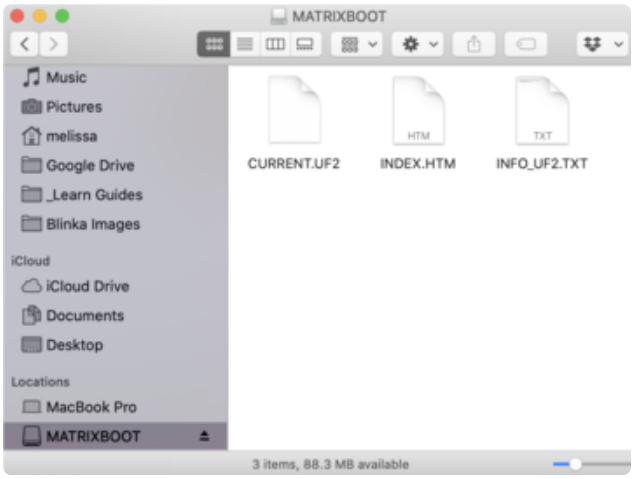
Plug your MatrixPortal M4 into your computer using a known-good USB cable.



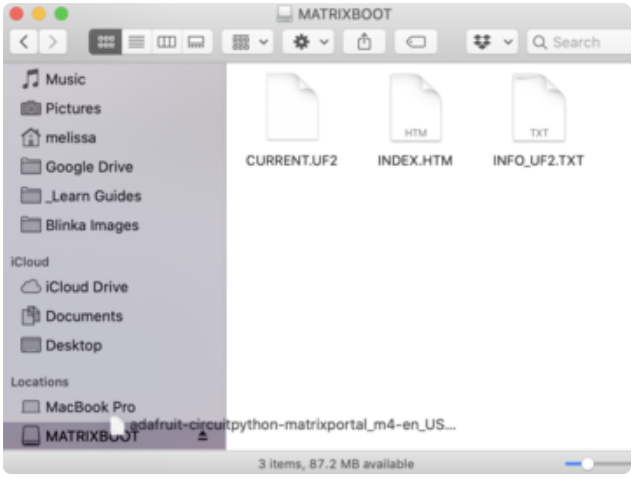
A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Double-click the **Reset** button (indicated by the green arrow) on your board, and you will see the NeoPixel RGB LED (indicated by the magenta arrow) turn green. If it turns red, check the USB cable, try another USB port, etc.

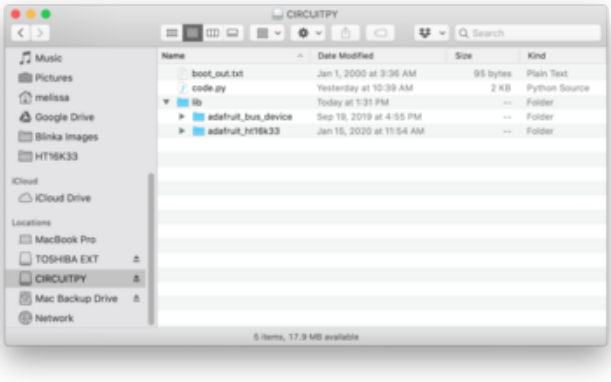
If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!



You will see a new disk drive appear called **MATRIXBOOT**.



Drag the **adafruit_circuitpython_etc.uf2** file to **MATRIXBOOT.**



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

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

Coding the RGB Matrix Dreidel Game

Once you've finished setting up your MatrixPortal M4 with CircuitPython, you can access the code, audio file, bitmap 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 as a zipped folder.

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

import time
import random
import board
import pwmio
import displayio
import adafruit_imageload
from audiocore import WaveFile
from adafruit_motor import servo
from digitalio import DigitalInOut, Direction, Pull
from adafruit_matrixportal.matrix import Matrix

I2S_VERSION = False #  set to True if using I2S audio out

#  import the appropriate audio module
if I2S_VERSION:
    from audiobusio import I2SOut
else:
    from audioio import AudioOut

#  setup for down button on matrixportal
switch = DigitalInOut(board.BUTTON_DOWN)
switch.direction = Direction.INPUT
switch.pull = Pull.UP

#  setup for break beam sensor
break_beam = DigitalInOut(board.A1)
break_beam.direction = Direction.INPUT
break_beam.pull = Pull.UP

#  pwm for servo
servo_pwm = pwmio.PWMOut(board.A4, duty_cycle=2 ** 15, frequency=50)

#  servo setup
servo = servo.Servo(servo_pwm)
servo.angle = 90

#  import dreidel song audio file
wave_file = open("dreidel_song.wav", "rb")
wave = WaveFile(wave_file)

#  setup for audio out
if I2S_VERSION:
    audio = I2SOut(board.A2, board.A3, board.TX)
else:
    audio = AudioOut(board.A0)

#  setup for matrix display
matrix = Matrix(width=32, height=32)
display = matrix.display

group = displayio.Group()

#  import dreidel bitmap
dreidel_bit, dreidel_pal = adafruit_imageload.load("/dreidel.bmp",
                                                   bitmap=displayio.Bitmap,
                                                   palette=displayio.Palette)

dreidel_grid = displayio.TileGrid(dreidel_bit, pixel_shader=dreidel_pal,
                                  width=1, height=1,
                                  tile_height=32, tile_width=32,
                                  default_tile=0,
                                  x=0, y=0)

group.append(dreidel_grid)
```

```
#  show dreidel bitmap
display.root_group = group

timer = 0 #  time.monotonic() holder
spin = 0 #  index for tilegrid
speed = 0.1 #  rate that bitmap updates
clock = 0 #  initial time.monotonic() holder to act as time keeper
gimel = 3 #  bitmap index for gimel, the winning character
countdown = 5 #  countdown for length of game. default is 5 seconds
beam_state = False #  state machine for break beam
reset = False #  state for reset of game
dreidel = False #  state to track if dreidel game is running

clock = time.monotonic() #  initial time.monotonic()

while True:
    #  debouncing for break beam sensor
    if not break_beam.value and not beam_state:
        beam_state = True

    #  if the break beam sensor is triggered or the down button is pressed...
    if (not break_beam.value and beam_state) or not switch.value:
        #  update break beam state
        beam_state = False
        #  begin reset for game states
        reset = True
        print("pressed")
        #  quick delay
        time.sleep(0.1)

    #  if reset state...
    if reset:
        #  hold time.monotonic() value
        clock = time.monotonic()
        #  reset countdown
        countdown = 5
        #  choose random side of dreidel to begin spinning on
        spin = random.randint(0, 3)
        #  choose random speed spin the dreidel
        speed = random.uniform(0.05, 0.1)
        #  set game state to True
        dreidel = True
        #  turn off reset state
        reset = False

    #  if the game is running...
    if dreidel:
        #  play the dreidel song
        audio.play(wave)

        #  while the dreidel song is playing...
        while audio.playing:
            #  if more time has passed than the random delay setup in reset...
            if (timer + speed) < time.monotonic():
                #  dreidel grid index is set to spin value
                dreidel_grid[0] = spin
                #  spin is increased by 1
                spin += 1
                #  timer is updated to current time
                timer = time.monotonic()

                #  if a second has passed...
                if time.monotonic() > (clock + 1):
                    print(clock)
                    print(spin)
                    #  the delay is increased to slow down the dreidel
                    speed += 0.05
                    #  clock is set to current time
```

```
                    clock = time.monotonic()
                    #  countdown value is decreased by 1
                    countdown -= 1

                #  if countdown is 0 aka 5 seconds has passed since the start of
game...
                if countdown == 0:
                    #  if the bitmap is showing gimel...
                    #  you win!
                    if spin is gimel:
                        #  the servo turns 90 degrees to dump out chocolate coins
                        servo.angle = 0
                        #  2 second delay
                        time.sleep(2)
                        #  servo turns back to default position
                        for i in range(0, 90, 2):
                            servo.angle = i
                            time.sleep(0.1)
                        #  ensures servo is in default position
                        servo.angle = 90
                        #  stop playing the dreidel song
                        audio.stop()
                        #  game state is turned off
                        dreidel = False

                    #  if you didn't win...
                    else:
                        #  the dreidel song stops
                        audio.stop()
                        #  game state is turned off
                        dreidel = False

                #  if you are at the end of the sprite sheet...
                if spin > 3:
                    #  index is reset to 0
                    spin = 0
```

I2S Audio Version Changes

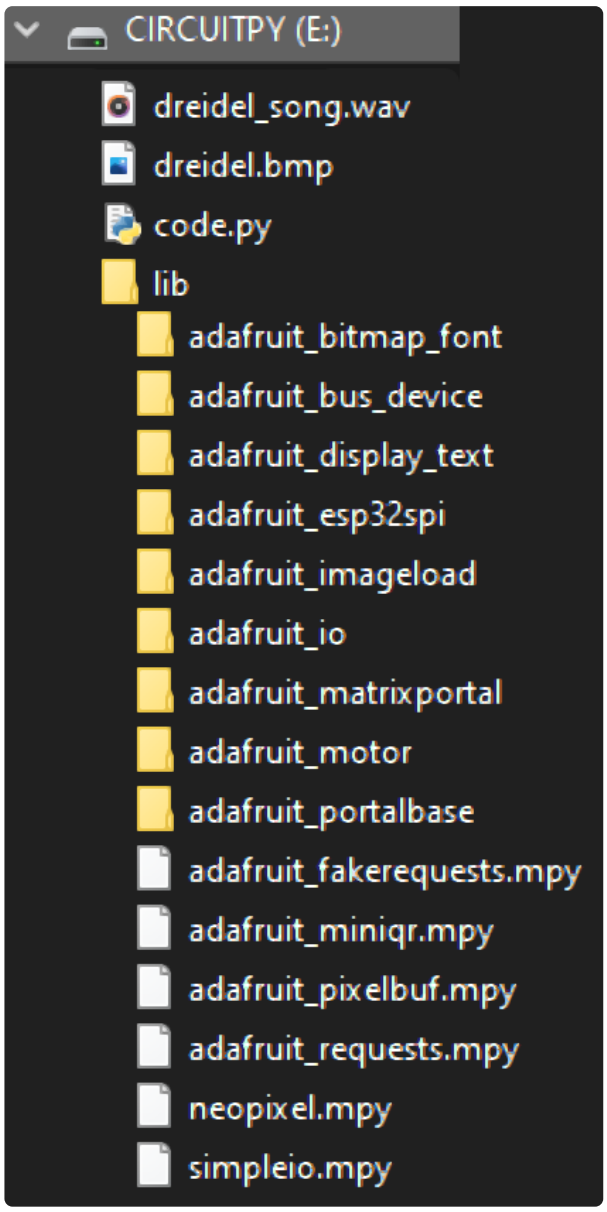
If you have built the I2S Audio version of the circuit, you will need to change `I2S_VERSION` to **True** before uploading your code.

Upload the Code, Audio File, Bitmap and Libraries to the MatrixPortal M4

After downloading the Project Bundle, plug your MatrixPortal M4 into the computer's USB port. 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 MatrixPortal M4's **CIRCUITPY** drive.

- **lib** folder
- **code.py**
- **dreidel_song.wav**
- **dreidel.bmp**

Your MatrixPortal M4 **CIRCUITPY** drive should look like this after copying the **lib** folder and the **code.py**, **dreidel_song.wav** and **dreidel.bmp** files.



How the CircuitPython Code Works

"Spin" the Dreidel

The MatrixPortal M4's onboard down button and an IR breakbeam sensor are setup as digital inputs that can be read to start up the dreidel game sequence. If either input is triggered in the loop, then the `reset` state is set to `True`. This resets all of the game's values to launch a new sequence.

```
#  if the break beam sensor is triggered or the down button is pressed...
    if (not break_beam.value and beam_state) or not switch.value:
        #  update break beam state
```

```
        beam_state = False
        #  begin reset for game states
        reset = True
        print("pressed")
        #  quick delay
        time.sleep(0.1)
```

`reset` State

The `reset` state acts as a way to make sure everything is ready for a new round of dreidel. Most importantly, the values of `spin` and `speed` are set to random values. This allows for some variation in how the dreidel spins. `spin` affects the first index that is shown in the tilegrid and `speed` affects the speed at which the indexes change on the matrix.

```
#  if reset state...
    if reset:
        #  hold time.monotonic() value
        clock = time.monotonic()
        #  reset countdown
        countdown = 5
        #  choose random side of dreidel to begin spinning on
        spin = random.randint(0, 3)
        #  choose random speed spin the dreidel
        speed = random.uniform(0.05, 0.1)
        #  set game state to True
        dreidel = True
        #  turn off reset state
        reset = False
```

The Dreidel is Rolling

When the `dreidel` state is `True`, then the dreidel song will play through the STEMMA speaker and you'll see the dreidel sprite sheet iterate through the four sides with the different characters.

`speed` is used as the delay, with `timer` being reset to the current time with each iteration. This lets you avoid using `time.sleep()`, which would pause the entire loop.

```
#  if the game is running...
    if dreidel:
        #  play the dreidel song
        audio.play(wave)

        #  while the dreidel song is playing...
        while audio.playing:
            #  if more time has passed than the random delay setup in reset...
            if (timer + speed) &lt; time.monotonic():
                #  dreidel grid index is set to spin value
                dreidel_grid[0] = spin
                #  spin is increased by 1
                spin += 1
```

```
                #  timer is updated to current time
                timer = time.monotonic()
```

The game is setup to "spin" the dreidel for 5 seconds. With each passing second, the value of `speed` is increased by `0.05` seconds. This slows the dreidel down gradually to mimic how it would spin in real life. `countdown` keeps track of how many seconds are left in the game.

```
#  if a second has passed...
                if time.monotonic() &gt; (clock + 1):
                    print(clock)
                    print(spin)
                    #  the delay is increased to slow down the dreidel
                    speed += 0.05
                    #  clock is set to current time
                    clock = time.monotonic()
                    #  countdown value is decreased by 1
                    countdown -= 1
```

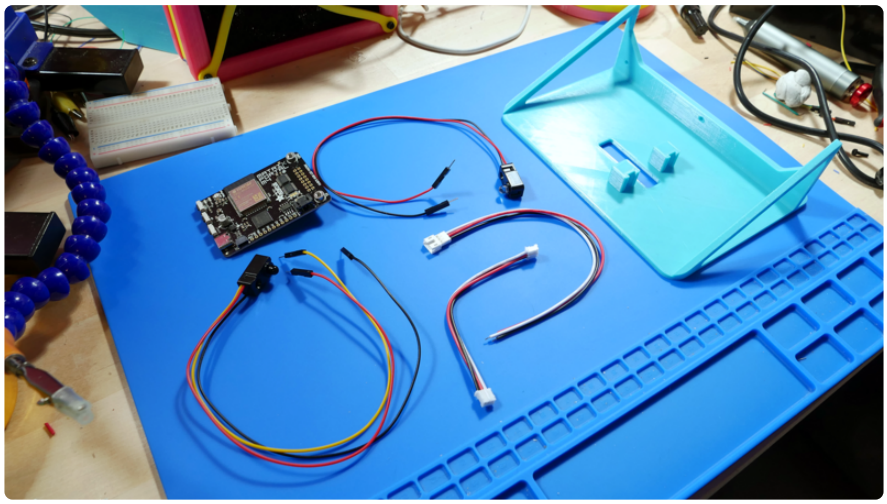
Once the game is over, it's determined whether or not you've won the pot of chocolate coins. If the matrix is showing gimel, which is the third index of the sprite sheet, then you've won! As a result, the servo will turn to dump out the chocolate coins for you and then turn back to its default position. The dreidel song will also stop playing and `dreidel` is set to `False`, stopping the game sequence.

If you didn't win, the song stops playing and the `dreidel` state is set to `False`, but there isn't any servo action.

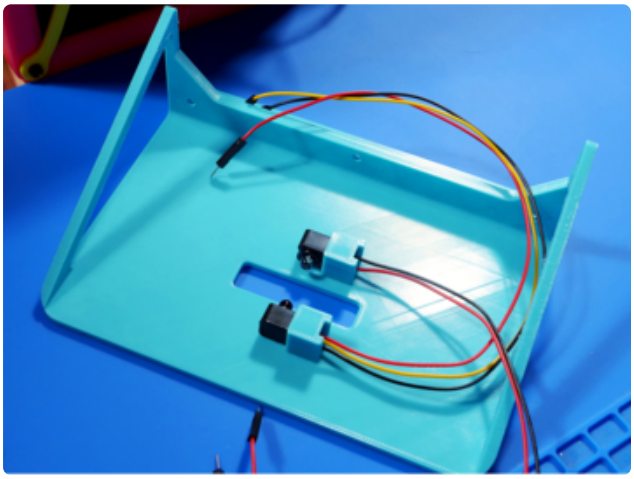
```
#  if countdown is 0 aka 5 seconds has passed since the start of game...
                if countdown == 0:
                    #  if the bitmap is showing gimel...
                    #  you win!
                    if spin is gimel:
                        #  the servo turns 90 degrees to dump out chocolate coins
                        servo.angle = 0
                        #  2 second delay
                        time.sleep(2)
                        #  servo turns back to default position
                        for i in range(0, 90, 2):
                            servo.angle = i
                            time.sleep(0.1)
                        #  ensures servo is in default position
                        servo.angle = 90
                        #  stop playing the dreidel song
                        audio.stop()
                        #  game state is turned off
                        dreidel = False

                    #  if you didn't win...
                    else:
                        #  the dreidel song stops
                        audio.stop()
                        #  game state is turned off
                        dreidel = False
```

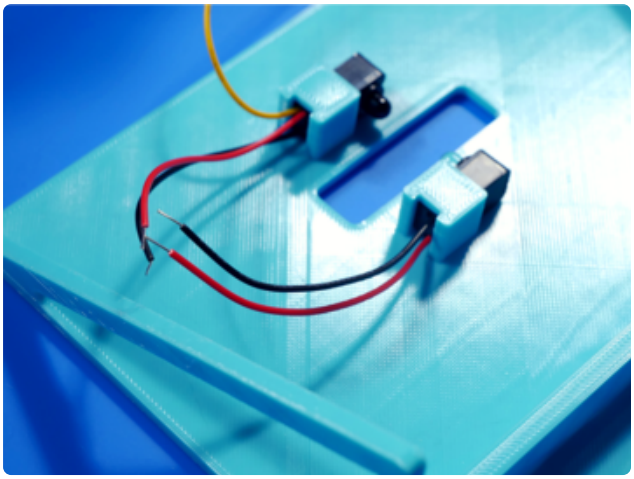
Wiring

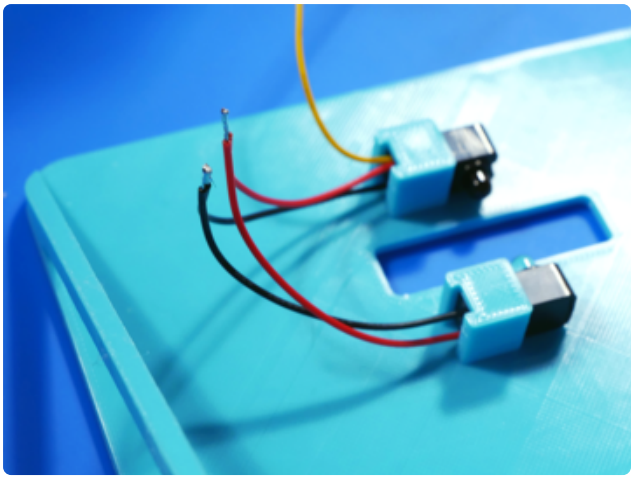


Make sure to use heat shrink for any exposed wire connections!

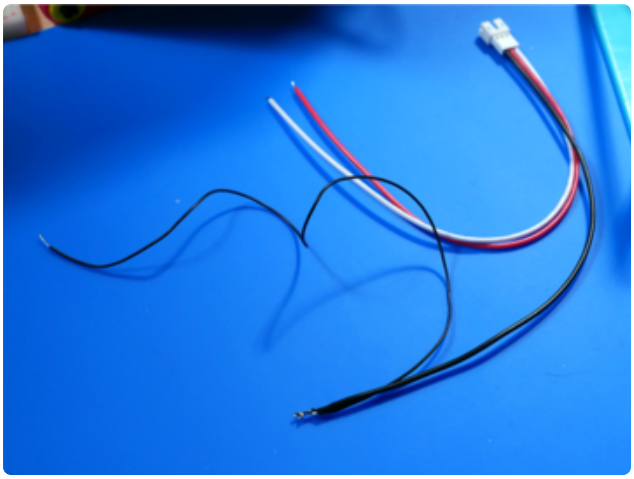


Insert the two breakbeam sensors into their slots at the top of the 3D printed lid.

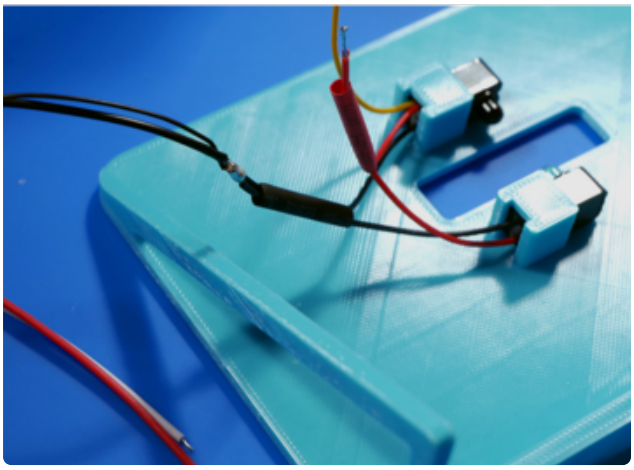




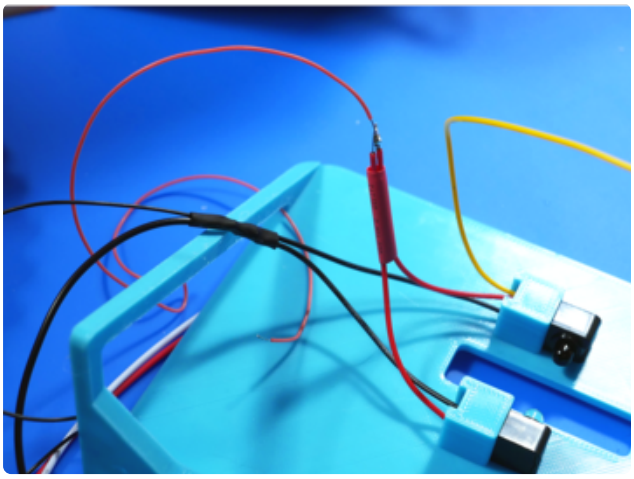
Strip and tin the breakbeam sensors' ground and power connections. Solder their ground connections together. Then, solder their power connections together.



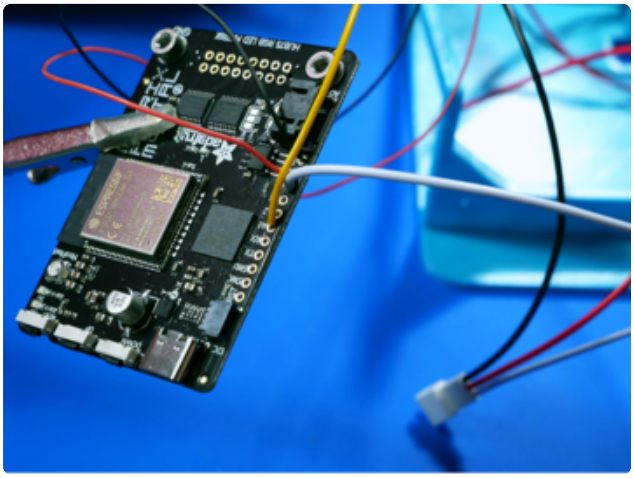
The JST 3-pin socket cable will plug into the servo motor. Solder the cable's ground connection to an additional wire.



Then, solder the breakbeam's shared ground connections to this ground junction. Cover the connection with heat shrink.



Solder an additional wire to the breakbeam sensors' power wires. Cover the connection with heat shrink.

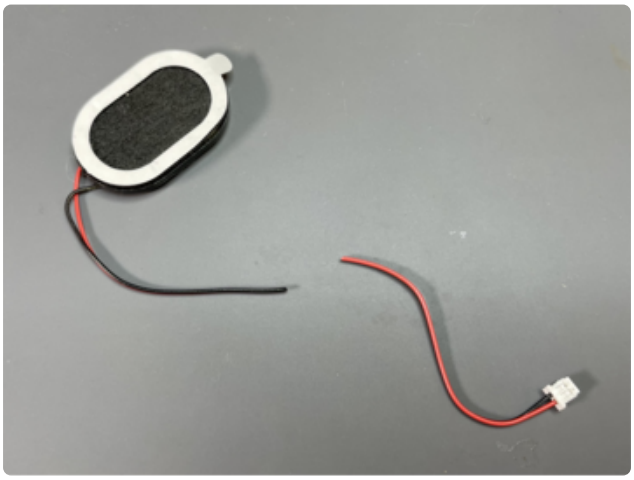


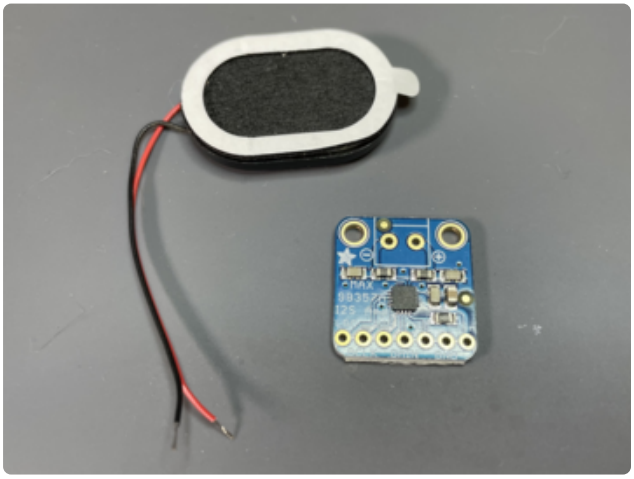
Solder the following connections to the MatrixPortal M4's pins.

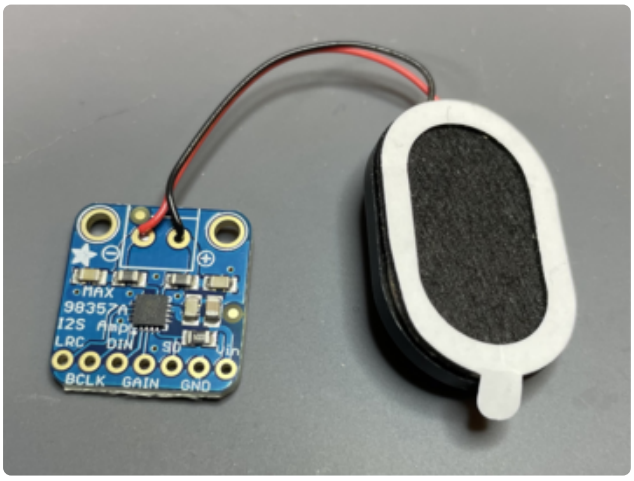
The group's **ground** connection to the MatrixPortal M4's **ground** pin
The breakbeam sensors' **data** pin to **A1**
The JST socket's **data** pin to **A4**
For the Audio Amplifier Version, also solder:

The breakbeam sensors' **power** connection to the MatrixPortal M4's **3V** pin

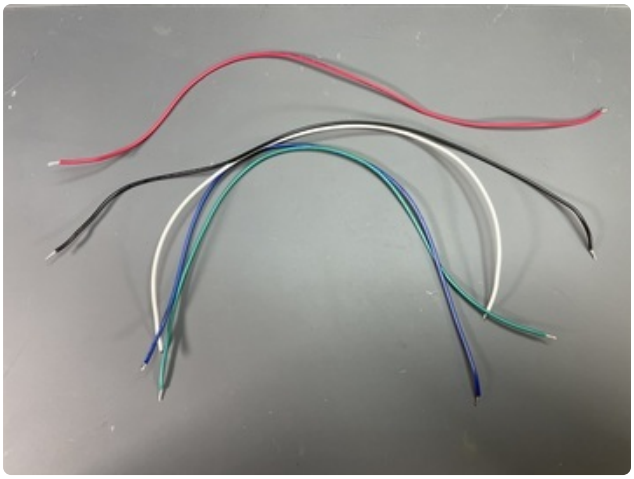
Additional I2S Audio Version Wiring



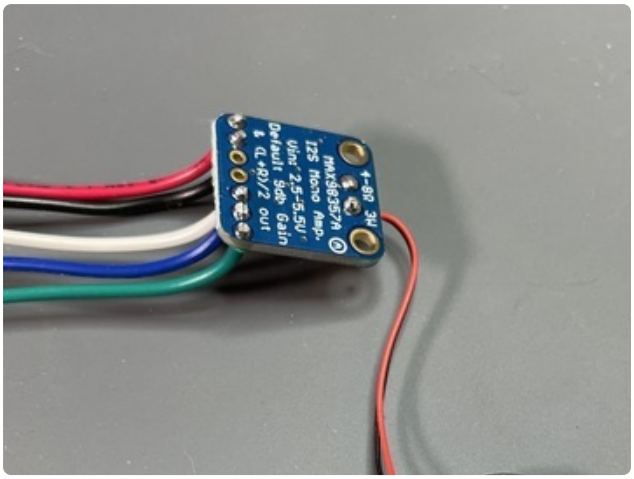




Prepare the speaker by cutting and stripping the wire. Solder the speaker wires to the MAX98357.



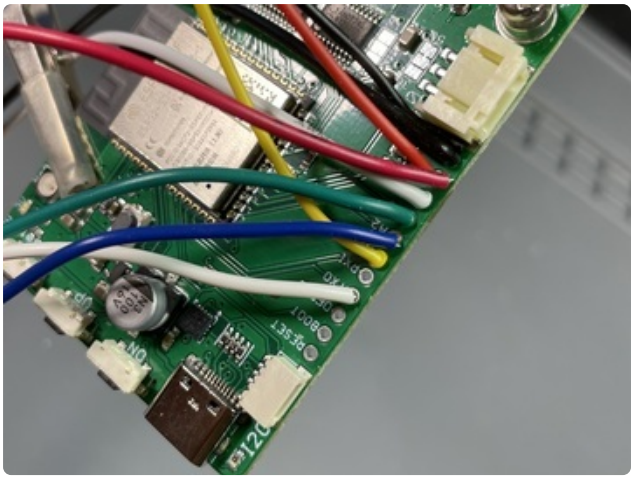
Prepare 5 wires about 12" long each by stripping the insulation. If you are using jumpers, cut off the ends first.



Solder the following connections to the MAX98357:

Red wire to the **Vin**
Black wire to **Gnd**
White wire to **DIN**
Blue wire to **BCLK**
Green wire to **LRC**

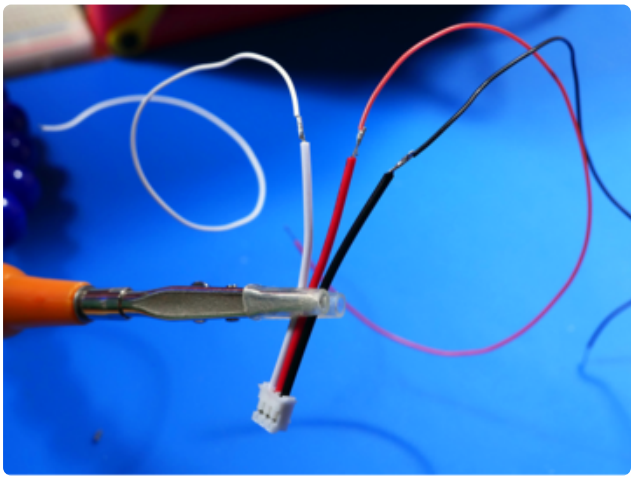
If you are having difficulty keeping the 3V wires and Ground Wires from touching, you may consider adding some hot glue as insulation.

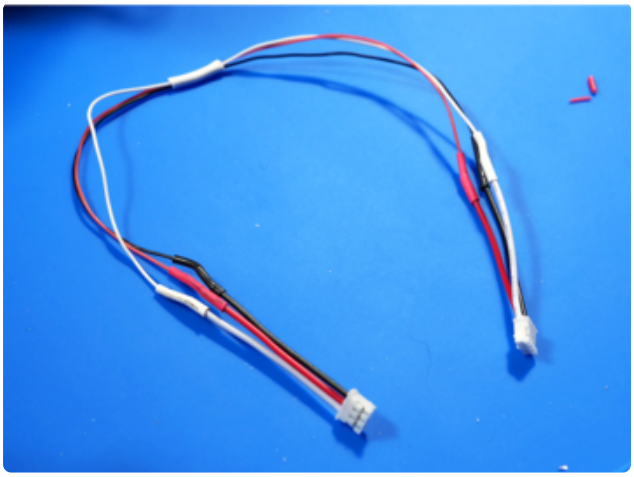


Solder the following additional connections to the MatrixPortal's pins.

The breakbeam sensors' **power** connection and the MAX98357's **power** connection to the MatrixPortal M4's **3V** pin
The **blue wire** from the MAX98357's **BCLK** pin to **A2**
The **green wire** from the MAX98357's **LRC** pin to **A3**
The **white wire** from the MAX98357's **DIN** pin to **TX**

Make a longer Audio Amplifier JST 3-pin cable

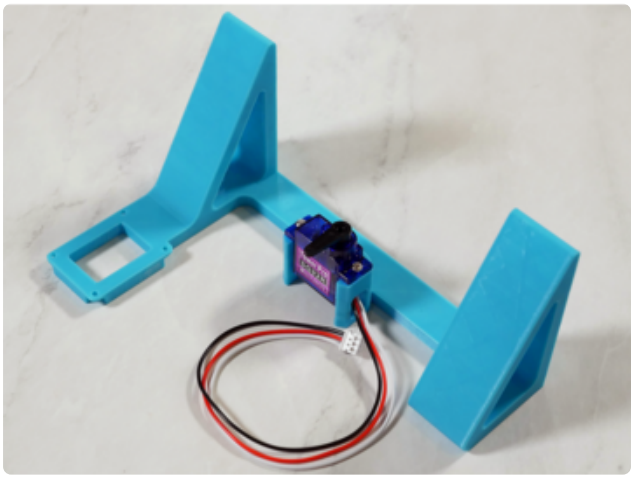




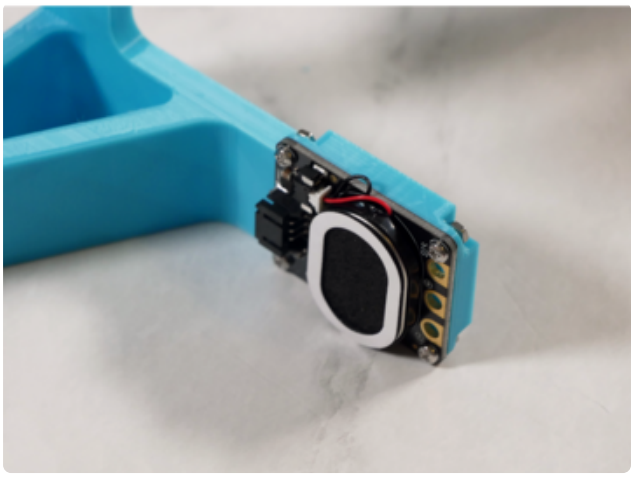
If you are building the Audio Amplifier version, cut a JST 3-pin cable in half. Then, strip and tin its wires.

Solder wires to each JST end to create a longer cable. Cover all connections with heat shrink. This cable will be used for the STEMMA speaker.

Assembly

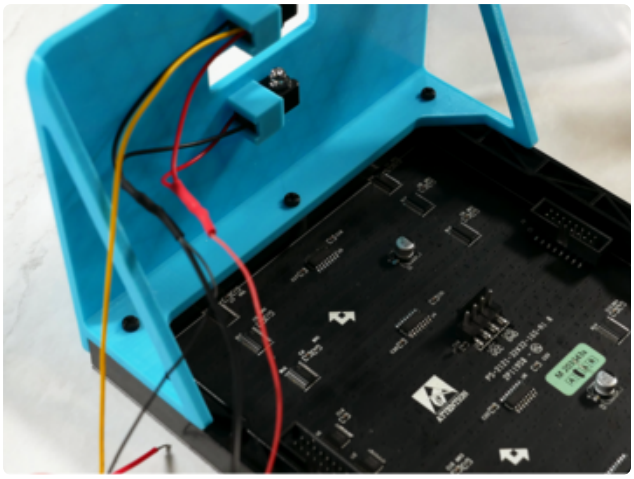


Slide the servo motor into the servo motor slot on top of the 3D printed stand. Make sure to pass its wires through the cut-out on the side. Secure it with two M2 screws.

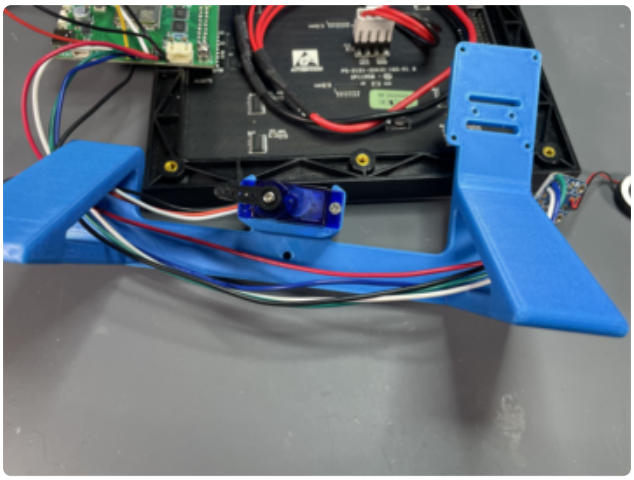


If building the audio amplifier version, attach the STEMMA Speaker to the side of the 3D printed stand with four M2 screws and four M2 nuts.

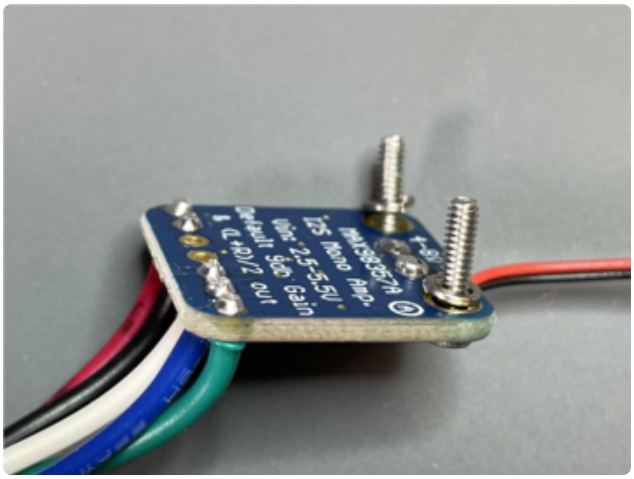
If you are building the I2S version, the speaker will be attached in a later step.

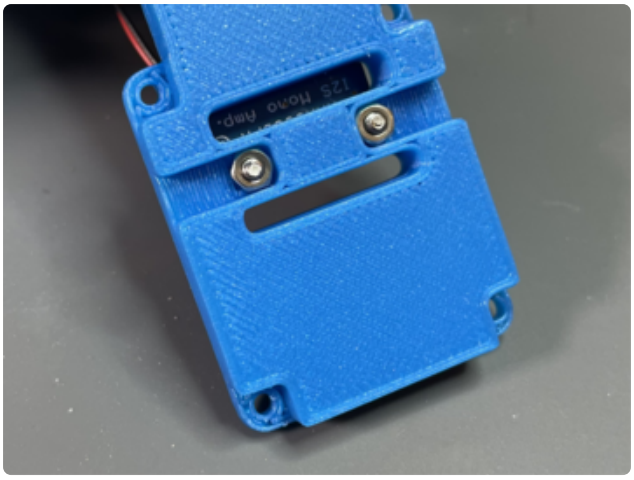


Attach the 3D printed lid to the top of the RGB matrix using three M3 screws.



If you are building the I2S Audio version, feed the wiring through the legs of the printed stand before attaching.

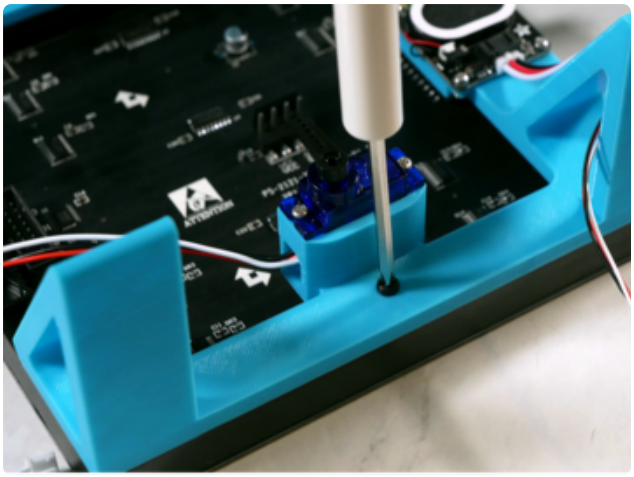




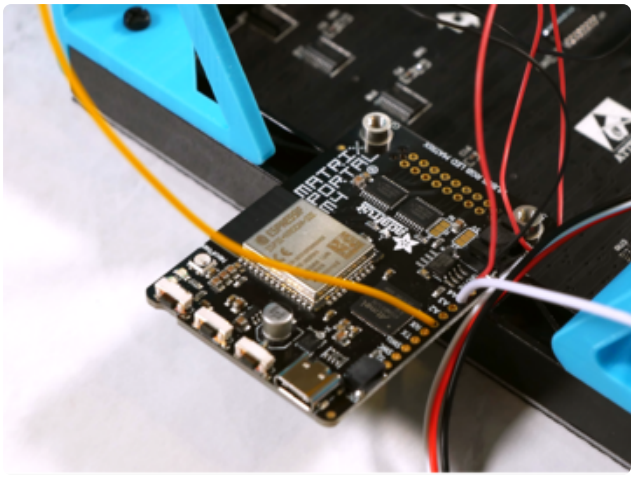
Place 2 of the 8mm M2 screws into the MAX98357 and then slide on a couple locknuts to act as spacers on those screws.

Insert the screws into the printed stand and secure with nuts.

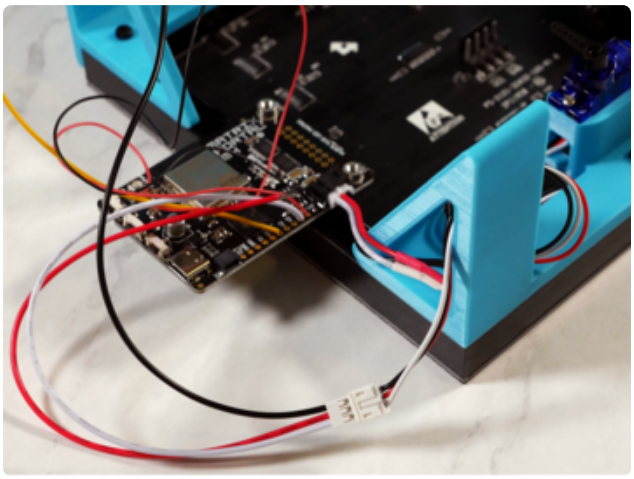
Use the adhesive on the back of the speaker to attach it right above the MAX98357.



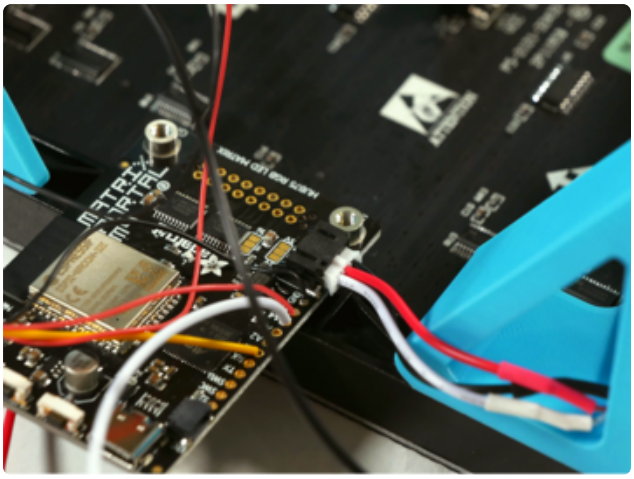
Attach the 3D printed stand to the bottom of the RGB matrix using one M3 screw.

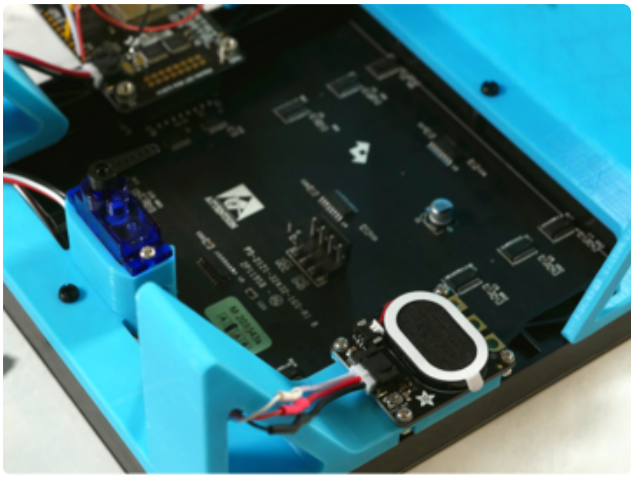


Plug the MatrixPortal M4 into the back of the RGB matrix.

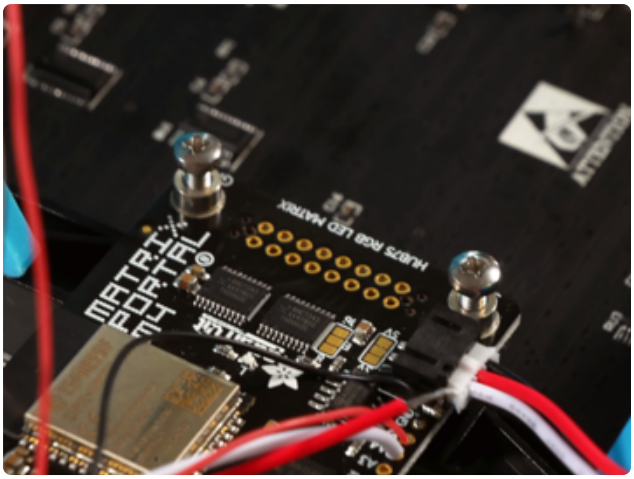


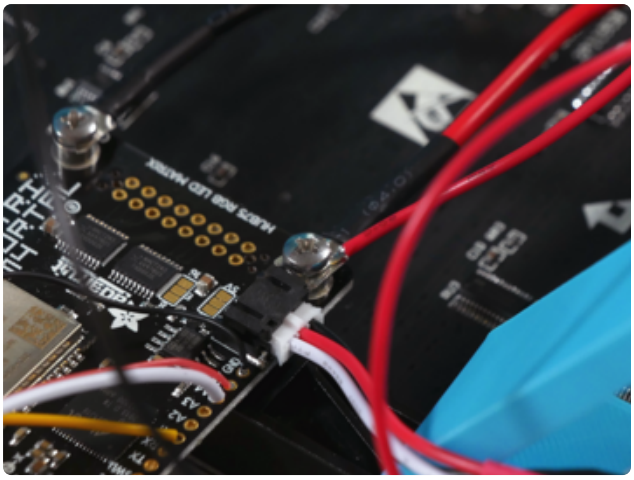
Plug the servo motor into the soldered 3-pin JST socket cable. This connects the servo to pin A4.

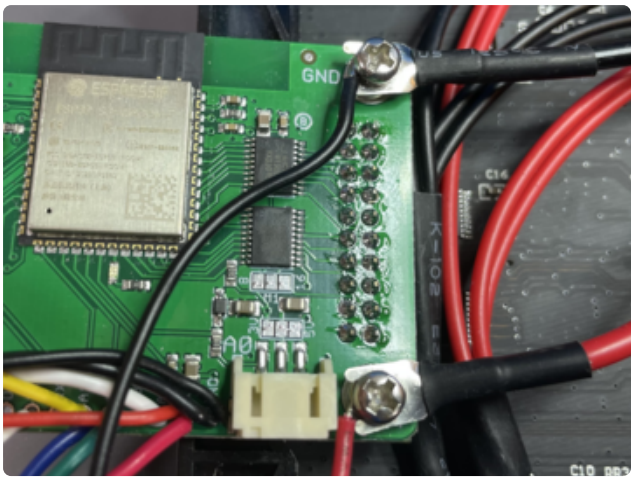




If you are building the audio amplifier version, plug the STEMMA speaker into the MatrixPortal M4's onboard JST socket with the elongated 3-pin JST cable. This will connect the speaker to pin A0.





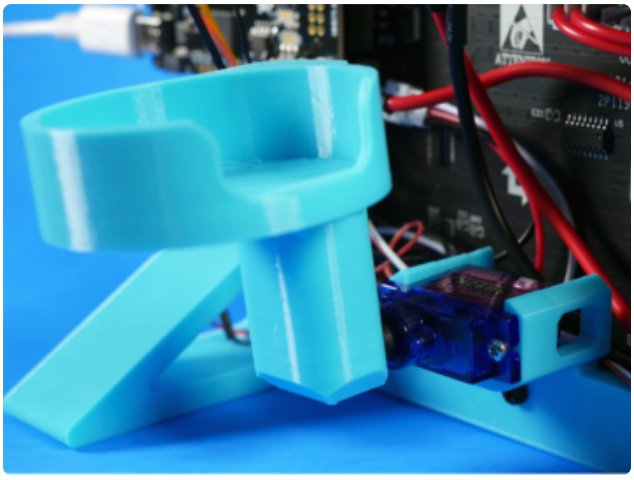


Attach the RGB matrix's power cable and the JST 3-pin socket cable's power pin to 5V on the MatrixPortal M4 by securing them with an M3 screw.

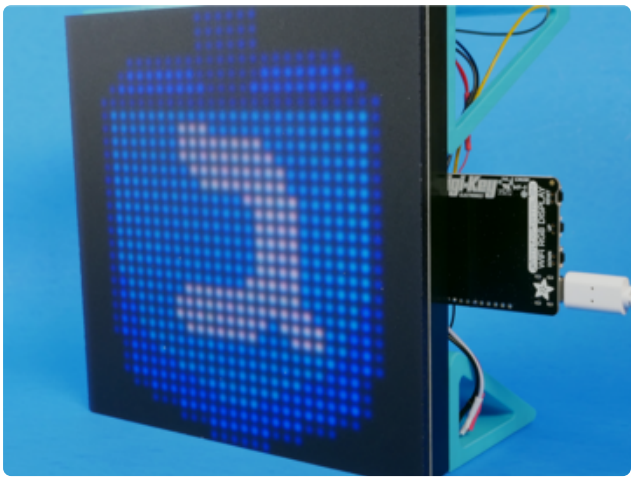
Attach the RGB matrix's ground cable to the MatrixPortal M4 with an M3 screw.

Plug the matrix power cable into the back of the matrix.

If you are building the I2S Audio version, attach the ground wire from the MAX98357 to the matrix's ground cable in the same way you attached the JST's power pin.



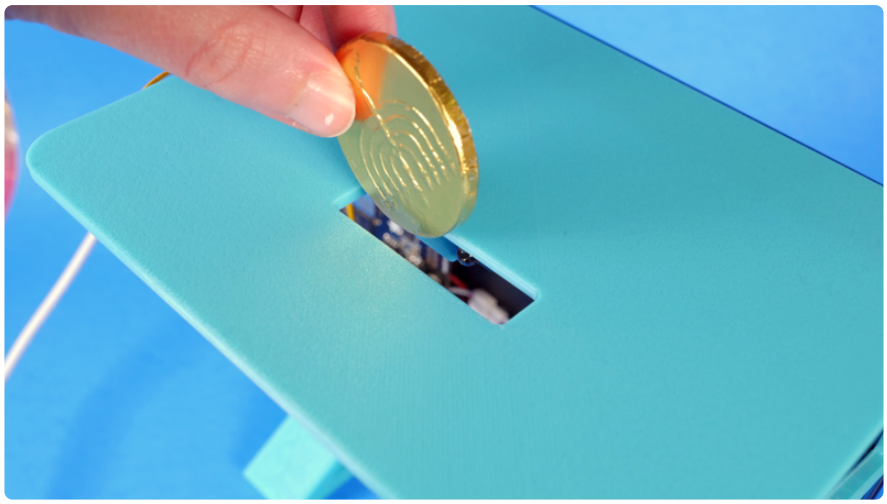
Super glue the servo horn to the 3D printed chocolate coin cup. Then, attach the servo horn to the servo motor.



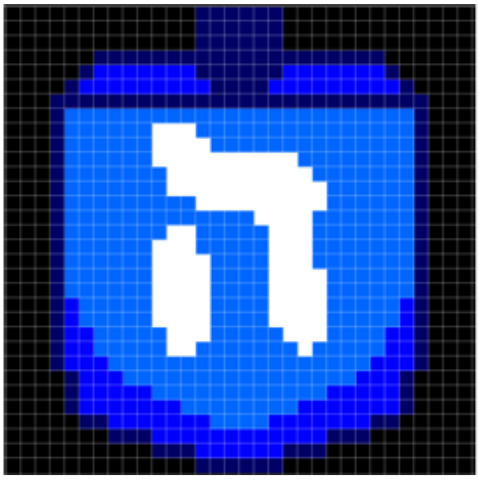
Optional: cut a piece of LED acrylic to fit the 32x32 matrix and attach it to the front of the matrix with four squares of clear adhesive

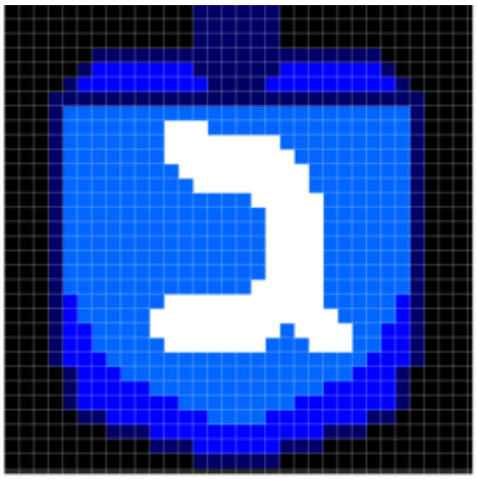
Play Dreidel!

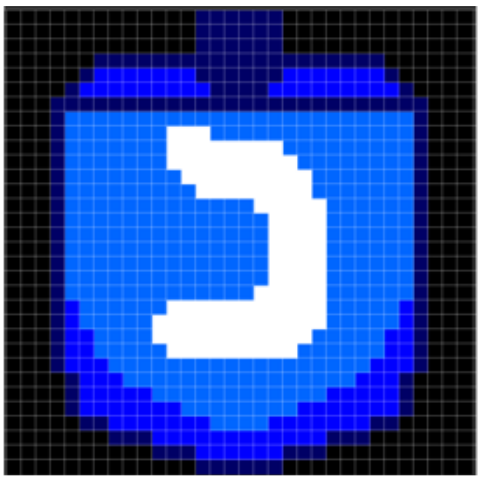
After powering up the MatrixPortal M4 and the RGB matrix, you can begin your game of dreidel two ways: either by dropping a chocolate coin into the coin slot to trigger the break beam sensor or by pressing the down button on the side of the MatrixPortal M4.



The classic dreidel song will begin playing through the STEMMA speaker and the dreidel will "spin" on the RGB matrix, iterating through the dreidel sprite sheet.





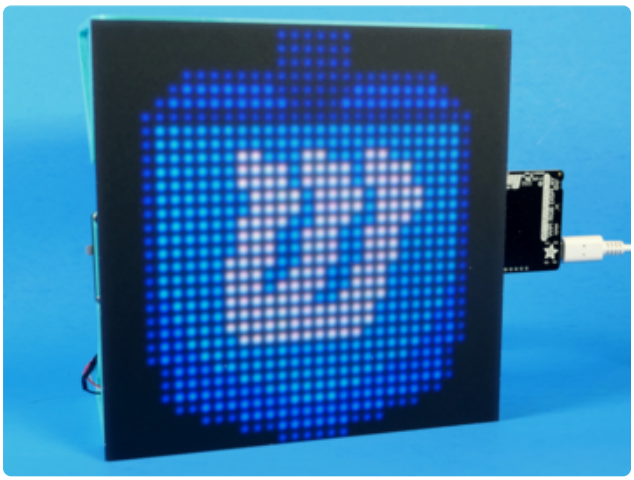


A dreidel has four characters, one on each side. The characters are:

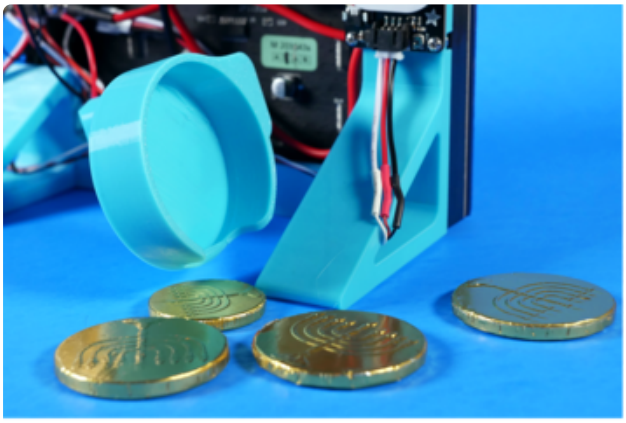
Shin (ש)
He (ה)
Gimel (ג)
Nun (נ)
Traditionally when you are playing dreidel, each side has a different meaning in the game. For the purposes of this version though, you will keep adding coins to the pot until you roll gimel; winning the game.

For more information on the traditional dreidel game rules, check out this page in the Circuit Playground TFT Gizmo Dreidel Learn Guide

https://adafru.it/WRf



If you don't roll gimel to win the game, don't worry! You can keep dropping in coins or pressing the down button to launch a new cycle.



If you do roll gimel though, congratulations! You've won the pot! The servo motor will tip the cup holding all of the chocolate coins, spilling them out for you to feast on.