Daily Cheer Automaton

Created by Dano Wall

https://learn.adafruit.com/daily-cheer-automaton

Last updated on 2024-06-03 03:07:09 PM EDT

Table of Contents

Overview

The Collective Cheer

> The collective cheering ritual started in Wuhan, China, in January and spread across the globe in the virus's wake. Here, it is sometimes accompanied by the song "New York, New York."
>
> - What NYC Sounds Like Every Night at 7 (https://adafru.it/KZc)

Sometimes you can't be around at exactly the right time every day to participate in the collective cheer for healthcare workers. This device will remember for you and participate on your behalf, even if you can't. Sometimes we can all use a helping hand.

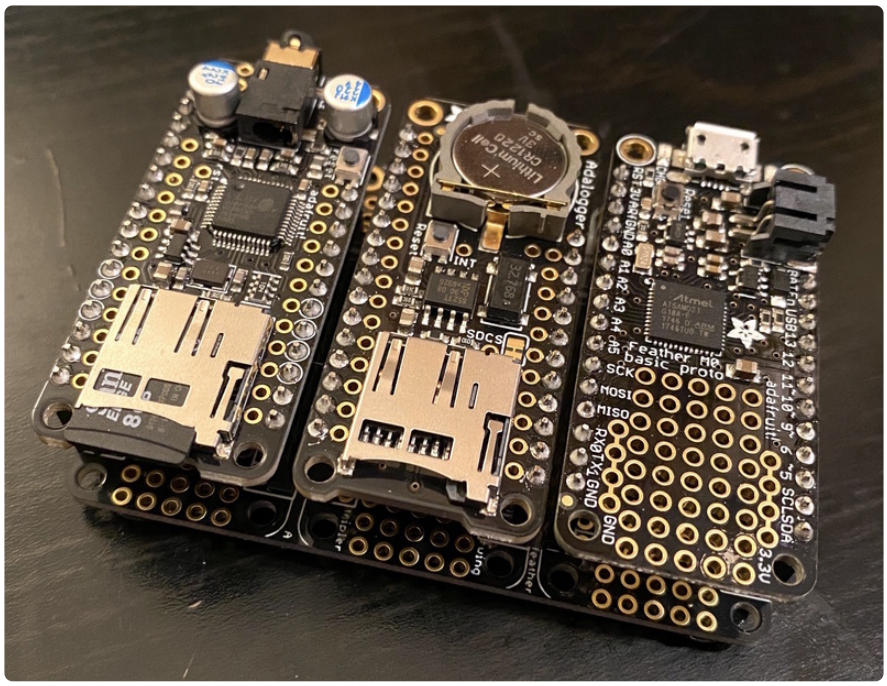
This project creates an automatic cheering device that will play an MP3 of a crowd cheering every day starting at 7 PM.

We've designed this project around the Feather system of boards (https://adafru.it/l7B), which makes the components modular and easy to put together and expand upon. It can be easily modified to play any MP3 of your choice, and to go off at any time of your choice.

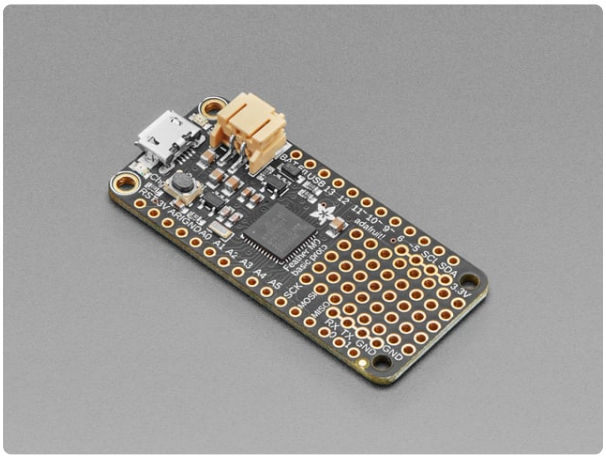
Arduino version

There is an Arduino version and CircuitPython version of this project, and they each use different hardware.

The Arduino version uses a Music Maker FeatherWing for MP3 playback and an Adalogger FeatherWing to keep track of the time.



Parts List

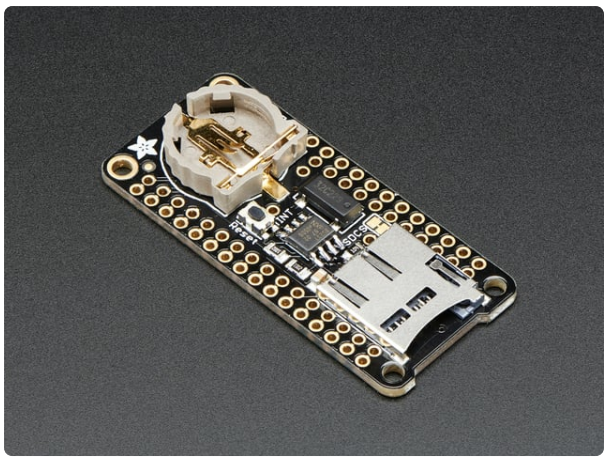


Adafruit Feather M0 Basic Proto - ATSAMD21 Cortex M0

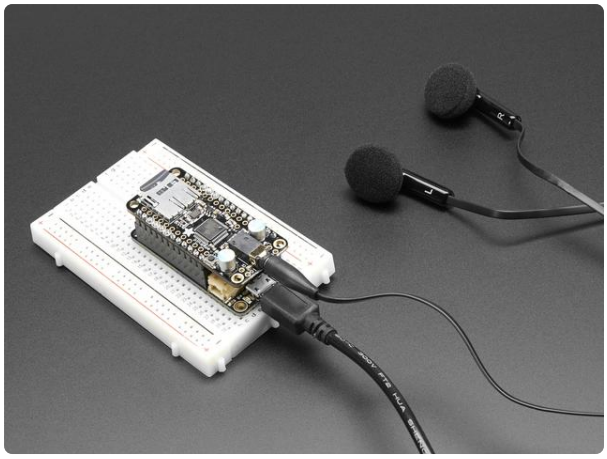
Feather is the new development board from Adafruit, and like its namesake it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller...

https://www.adafruit.com/product/2772

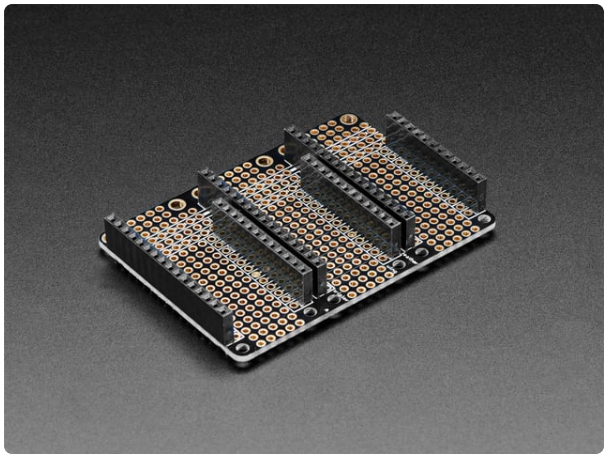
The Feather M0 Basic Proto is recommended here, but nearly any fast Feather board (https://adafru.it/l7B) should work!



[Adalogger FeatherWing - RTC + SD Add-on For All Feather Boards]
A Feather board without ambition is a Feather board without FeatherWings! This is the Adalogger FeatherWing: it adds both a battery-backed Real Time Clock and micro SD...
https://www.adafruit.com/product/2922



[Adafruit Music Maker FeatherWing - MP3 OGG WAV MIDI Synth Player]
Bend all audio files to your will with the Adafruit Music Maker FeatherWing! It's a fun-size version of our Music...
https://www.adafruit.com/product/3357



[FeatherWing Tripler Mini Kit - Prototyping Add-on For Feathers]
This is the FeatherWing Tripler - a prototyping add-on and more for all Feather boards. This is similar to our
https://www.adafruit.com/product/3417

[SD/MicroSD Memory Card (8 GB SDHC)]
Add mega-storage in a jiffy using this 8 GB class 4 micro-SD card. It comes with a SD adapter so you can use it with any of our shields or adapters. Preformatted to FAT so it works out...
https://www.adafruit.com/product/1294

CR1220 12mm Diameter - 3V Lithium Coin Cell Battery

These are the highest quality & capacity batteries, the same as shipped with the iCufflinks, iNecklace, Datalogging and GPS Shields, GPS HAT, etc. One battery per order...

https://www.adafruit.com/product/380

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

Assembly

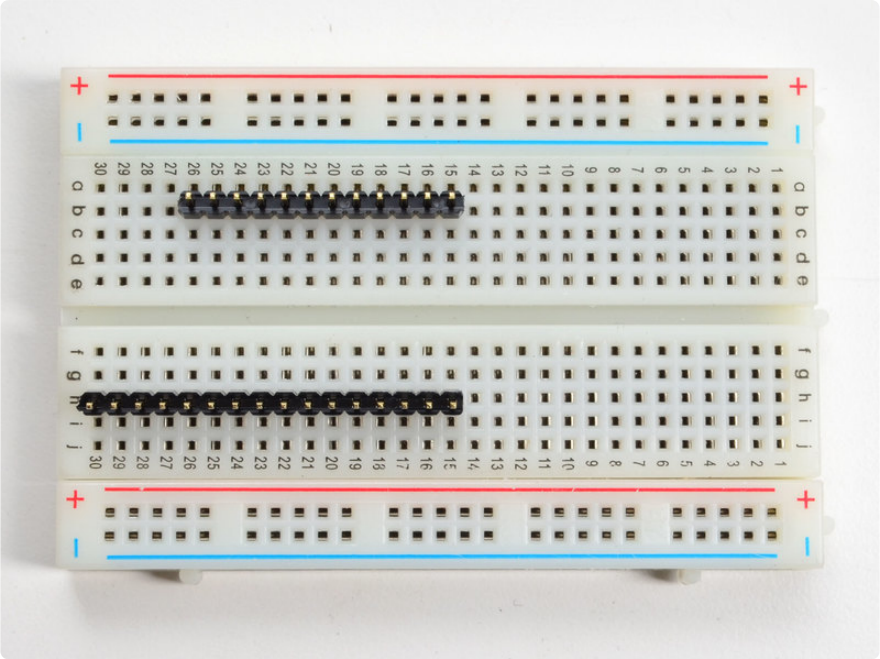
You'll need a soldering iron and solder (https://adafru.it/drl).

We are building this project around the FeatherWing Tripler (https://adafru.it/L1a) to connect the Feather and two FeatherWings.

You could also use Feather Stacking Headers (http://adafru.it/2830) for a different form factor, to have the boards stacked on top of each other rather than next to each other. Decide what you'd like to do, then plan ahead accordingly!

In this example, we'll solder male headers onto the Feather and two FeatherWings, and female headers onto the FeatherWing Tripler.

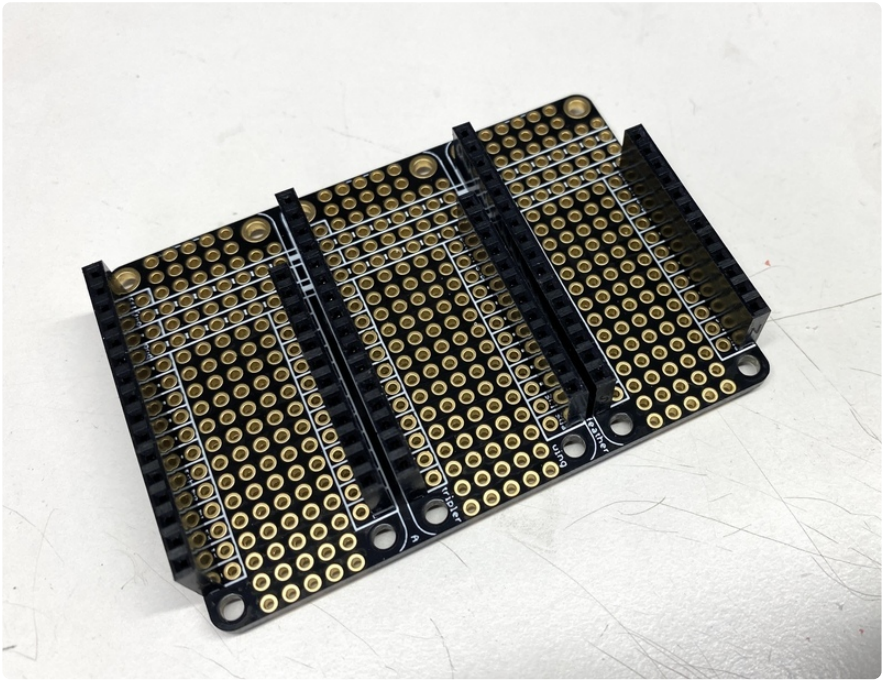
Follow the instructions for attaching male headers to your Feather board (https://adafru.it/L1b). It helps to use a solderless breadboard (http://adafru.it/64) to get the headers on straight.



Do the same for the Music Maker FeatherWing (https://adafru.it/xcm), attaching **male headers**.

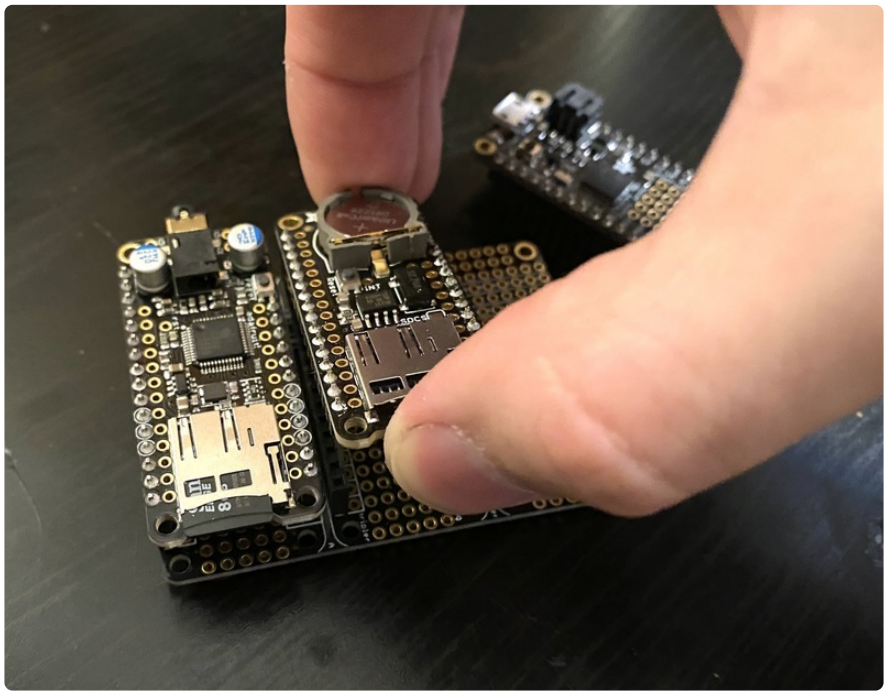
Then attach **male headers** to the Adalogger FeatherWing (https://adafru.it/LBF).

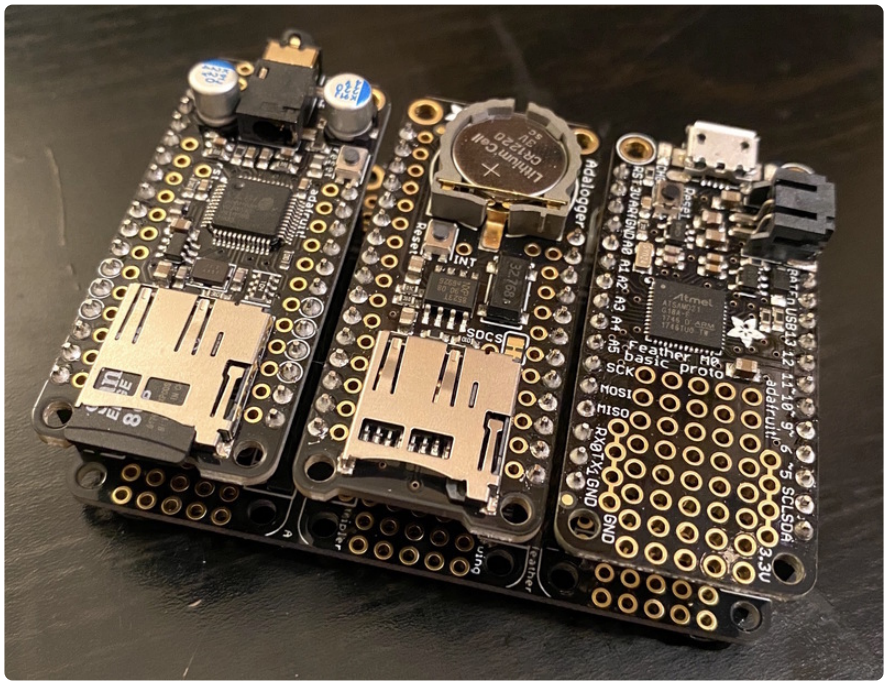
Attach **female headers** to the FeatherWing Tripler. You can use one your soldered Feather boards to help keep the female headers straight while you solder.



Place the Feather, Adalogger FeatherWing, and Music Maker FeatherWing into each of the Tripler's slots. The order is up to you, but you may want to consider things which will connect to the device, such as a possible LiPo battery, which would mean

placing the Feather on the far right side. Also consider cables that will be connecting to the device, such as a USB cable to the Feather to provide power, or the audio jack output on the Music Maker FeatherWing.





Software

Installing Boards and Libraries

Make sure to use Arduino IDE 1.6.4 or higher (https://adafru.it/fvm) and follow this tutorial (https://adafru.it/f7X) to install the Adafruit boards.

Download the code (https://adafru.it/L1d) from github by clicking **Download ZIP**. Uncompress the file and copy the contents of the **Arduino** folder to your Arduino sketchbook folder.

```
// SPDX-FileCopyrightText: 2020 Andy Doro for Adafruit Industries
//
// SPDX-License-Identifier: MIT

/*
    AUTOCHEER DEVICE
    ---
    code by Andy Doro https://andydoro.com/

    using Adafruit Feather hardware
    will play an MP3 at a specified time each day

    sketch can easily be modified to perform some other functions at the specified
time, such as operate physical/analog noisemakers


    HARDWARE
    ---
    - any Feather board, e.g. Feather M0 Basic Proto https://www.adafruit.com/
product/2772

    - Adalogger FeatherWing https://www.adafruit.com/product/2922
    - CR1220 coin cell https://www.adafruit.com/product/380

    - Music Maker FeatherWing https://www.adafruit.com/product/3357 for 1/8" audio
jack output
    or Music Maker FeatherWing w/Amp https://www.adafruit.com/product/3436 for
speaker wire output
    - MicroSD card https://www.adafruit.com/product/1294 FAT formatted with
"cheer.mp3"

    - FeatherWing Tripler https://www.adafruit.com/product/3417
    or Feather Stacking Headers https://www.adafruit.com/product/2830 for a
different form factor



    SOFTWARE
    ---
    libraries
    - VS1053 https://github.com/adafruit/Adafruit_VS1053_Library for Music Maker
    - RCTlib https://github.com/adafruit/RTClib for RTC
    - DST_RTC https://github.com/andydoro/DST_RTC for Daylight Saving Time
adjustments

    cheer.mp3, place on FAT formatted SD card and insert into Music Maker
    http://www.orangefreesounds.com/street-crowd-cheering-and-applauding/
```

```
*/

// Specifically for use with the Adafruit Feather, the pins are pre-set here!

// include SPI, MP3 and SD libraries
#include <SPI.h>
#include <SD.h>
#include <Adafruit_VS1053.h> // https://github.com/adafruit/Adafruit_VS1053_Library

// Date and time functions using a DS3231 RTC connected via I2C and Wire lib
#include <Wire.h>
#include "RTClib.h" // https://github.com/adafruit/RTClib
#include "DST_RTC.h" // download from https://github.com/andydoro/DST_RTC


// These are the pins used
#define VS1053_RESET   -1     // VS1053 reset pin (not used!)

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

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

// Feather Teensy3
#elif defined(TEENSYDUINO)
#define VS1053_CS       3     // VS1053 chip select pin (output)
#define VS1053_DCS     10     // VS1053 Data/command select pin (output)
#define CARDCS          8     // Card chip select pin
#define VS1053_DREQ     4     // VS1053 Data request, ideally an Interrupt pin

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

#elif defined(ARDUINO_NRF52832_FEATHER )
#define VS1053_CS       30     // VS1053 chip select pin (output)
#define VS1053_DCS      11     // VS1053 Data/command select pin (output)
#define CARDCS          27     // Card chip select pin
#define VS1053_DREQ     31     // VS1053 Data request, ideally an Interrupt pin

// Feather M4, M0, 328, nRF52840 or 32u4
#else
#define VS1053_CS       6     // VS1053 chip select pin (output)
#define VS1053_DCS     10     // VS1053 Data/command select pin (output)
#define CARDCS          5     // Card chip select pin
// DREQ should be an Int pin *if possible* (not possible on 32u4)
#define VS1053_DREQ     9     // VS1053 Data request, ideally an Interrupt pin

#endif

Adafruit_VS1053_FilePlayer musicPlayer =
  Adafruit_VS1053_FilePlayer(VS1053_RESET, VS1053_CS, VS1053_DCS, VS1053_DREQ,
CARDCS);


//RTC_DS3231 rtc;
RTC_PCF8523 rtc; // RTC object
```

```
DST_RTC dst_rtc; // DST object

// Do you live in a country or territory that observes Daylight Saving Time?
// https://en.wikipedia.org/wiki/Daylight_saving_time_by_country
// Use 1 if you observe DST, 0 if you don't. This is programmed for DST in the US /
Canada. If your territory's DST operates differently,
// you'll need to modify the code in the DST_RTC library to make this work properly.
#define OBSERVE_DST 1

// Define US or EU rules for DST comment out as required. More countries could be
added with different rules in DST_RTC.cpp
const char rulesDST[] = "US"; // US DST rules
// const char rulesDST[] = "EU";   // EU DST rules

// the hour and minute you'd like MP3 to start playing
const int PLAYHOUR = 19; // 24 hour time
const int PLAYMIN = 0;

const int VOLUME = 0; // lower means louder!


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  // if you're using Bluefruit or LoRa/RFM Feather, disable the radio module
  //pinMode(8, INPUT_PULLUP);

  // Wait for serial port to be opened, remove this line for 'standalone' operation
  /*while (!Serial) {
    delay(1);
    }
  */
  delay(500);
  Serial.println("\n\nAdafruit VS1053 Feather Test");

  if (! musicPlayer.begin()) { // initialise the music player
    Serial.println(F("Couldn't find VS1053, do you have the right pins defined?"));
    while (1);
  }

  Serial.println(F("VS1053 found"));


  // Set volume for left, right channels. lower numbers == louder volume!
  musicPlayer.setVolume(VOLUME, VOLUME);

  musicPlayer.sineTest(0x44, 1000);    // Make a tone to indicate VS1053 is working

  if (!SD.begin(CARDCS)) {
    Serial.println(F("SD failed, or not present"));
    while (1);  // don't do anything more
  }
  Serial.println("SD OK!");

  // list files
  printDirectory(SD.open("/"), 0);



#if defined(__AVR_ATmega32U4__)
  // Timer interrupts are not suggested, better to use DREQ interrupt!
  // but we don't have them on the 32u4 feather...
  musicPlayer.useInterrupt(VS1053_FILEPLAYER_TIMER0_INT); // timer int
#else
  // If DREQ is on an interrupt pin we can do background
  // audio playing
  musicPlayer.useInterrupt(VS1053_FILEPLAYER_PIN_INT);  // DREQ int
#endif
```

```
  // Play a file in the background, REQUIRES interrupts!
  /*
    Serial.println(F("Playing full track 001"));
    musicPlayer.playFullFile("/track001.mp3");

    Serial.println(F("Playing track 002"));
    musicPlayer.startPlayingFile("/track002.mp3");
  */

  // start RTC
  if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    while (1);
  }

  // set RTC time if needed
  //if (rtc.lostPower()) { // if using DS3231
  if (! rtc.initialized()) {
    Serial.println("RTC lost power, lets set the time!");
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // DST? If we're in it, let's subtract an hour from the RTC time to keep our
DST calculation correct. This gives us
    // Standard Time which our DST check will add an hour back to if we're in DST.
    if (OBSERVE_DST == 1) {
      DateTime standardTime = rtc.now();
      if (dst_rtc.checkDST(standardTime) == true) { // check whether we're in DST
right now. If we are, subtract an hour.
        standardTime = standardTime.unixtime() - 3600;
      }
      rtc.adjust(standardTime);
    }
  }
}

void loop() {
  // put your main code here, to run repeatedly:

  DateTime theTime;
  // check time
  if (OBSERVE_DST == 1) {
    theTime = dst_rtc.calculateTime(rtc.now()); // takes into account DST
  } else {
    theTime = rtc.now(); // use if you don't need DST
  }

  printTheTime(theTime);

  byte theHour = theTime.hour();
  byte theMinute = theTime.minute();

  //check whether it's time to play mp3
  if (theHour == PLAYHOUR && theMinute == PLAYMIN) {
    Serial.println(F("Playing full track"));
    musicPlayer.playFullFile("/cheer.mp3");
  }

  // only check every second
  delay(1000);

}


// File listing helper
void printDirectory(File dir, int numTabs) {
  while (true) {

    File entry =  dir.openNextFile();
    if (! entry) {
```

```
      // no more files
      //Serial.println("**nomorefiles**");
      break;
    }
    for (uint8_t i = 0; i < numTabs; i++) {
      Serial.print('\t');
    }
    Serial.print(entry.name());
    if (entry.isDirectory()) {
      Serial.println("/");
      printDirectory(entry, numTabs + 1);
    } else {
      // files have sizes, directories do not
      Serial.print("\t\t");
      Serial.println(entry.size(), DEC);
    }
    entry.close();
  }
}

// print time to serial
void printTheTime(DateTime theTimeP) {
  Serial.print(theTimeP.year(), DEC);
  Serial.print('/');
  Serial.print(theTimeP.month(), DEC);
  Serial.print('/');
  Serial.print(theTimeP.day(), DEC);
  Serial.print(' ');
  Serial.print(theTimeP.hour(), DEC);
  Serial.print(':');
  Serial.print(theTimeP.minute(), DEC);
  Serial.print(':');
  Serial.print(theTimeP.second(), DEC);
  Serial.println();
}
```

You will also need to install some Adafruit Arduino libraries. Follow this tutorial (https://adafru.it/dit) if you are unfamiliar with how to do this. You need to install the following Adafruit Arduino libraries:

- VS1053 (https://adafru.it/clE) for the Music Maker
- RTClib (https://adafru.it/c7r) for the Real Time Clock (RTC)
- DST_RTC (https://adafru.it/BR3) for automatic Daylight Saving Time adjustments

All of the libraries should be available through Arduino's Library Manager (https://adafru.it/FXK), or you can install them manually (https://adafru.it/m3e).

Once you have installed the libraries and have the correct boards installed, choose your board and check if the code compiles. If the libraries aren't installed correctly you will see errors.

Audio File & SD Card

You'll also need an audio file for the Music Maker to play! We like this sample of a crowd cheering for front-line medical staff (https://adafru.it/L1f).

Rename the audio file **cheer.mp3** and make sure your SD card is FAT32 formatted (https://adafru.it/L1A). Copy the file onto the SD card and insert into the Music Maker FeatherWing.

Coin Cell

The Adalogger RTC FeatherWing needs a CR1220 coin cell installed in order to work properly and keep time when the device is unpowered.

Understanding the Code

Conceptually the daily cheering device is simple: the microcontroller is attached to a clock and an MP3 player. Every second, the microcontroller checks the time. If the hour and minute match the appointed time, the microcontroller will have the MP3 player play an MP3.

The selected time is defined in the code here:

```
// the hour and minute you'd like MP3 to start playing
const int PLAYHOUR = 19; // 24 hour time
const int PLAYMIN = 0;
```

The RTC gives time in 24 hour time (https://adafru.it/L1B), so 7 PM is 19:00.

Daylight Saving Time

Do you live in a territory that observes daylight saving time (https://adafru.it/ixB) (DST)? If you do, you usually have to reprogram your clocks twice a year! This clock includes some code so that the adjustments are made automatically. The code follows the current rules for DST in the USA (https://adafru.it/ixC) and Canada. If you live somewhere that follows different DST rules you may be able to modify the code to suit your rules— just look in the DST_RTC library functions (https://adafru.it/BR3). Wikipedia has a great reference on daylight saving time rules (https://adafru.it/ixD).

If you live in a territory that doesn't observe daylight saving time, just alter the following line by changing the `1` to `0`.

```
#define OBSERVE_DST 1
```

The daylight saving time code works by keeping the real time clock on "standard time" and checking to see if the current date falls within daylight saving time. If the date falls within daylight saving time, an hour is added to the displayed time to convert from standard time to daylight saving time.

Troubleshooting

The serial console will be important for debugging. To see all of the serial output from startup, make sure to uncomment:

```
// Wait for serial port to be opened, remove this line for 'standalone' operation
  /*while (!Serial) {
    delay(1);
    }
  */
```

However, this may prevent the device from working until you open the serial console.

You should see some startup messages, and then the date and time from the RTC being printed every second. Make sure the time is set correctly!

No audio? MP3 not playing?

You should hear a 1 second tone on startup, to let you know that the Music Maker audio chip (VS1053) is working and able to produce sounds.

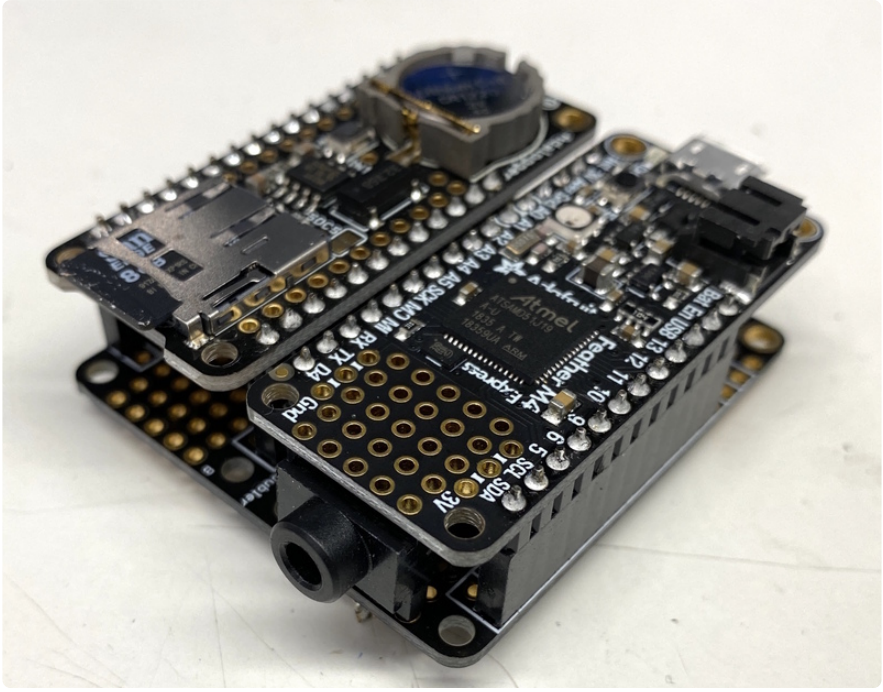
Otherwise, check the serial console for error messages. Make sure both the VS1053 and SD card are found. The serial console should indicate both `"VS1053 found"` and `"SD OK!"`. The serial console should also print the contents of the SD card, so you can check that to make sure the SD card is being read correctly.

You can always test that your mp3 will play by using the VS1053 example sketch (https://adafru.it/x9d). If you're having trouble, try this tutorial to make sure your audio file is microcontroller compatible (https://adafru.it/BvU).

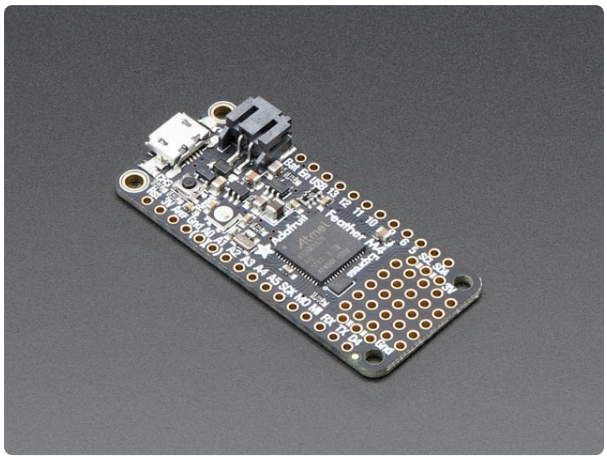
CircuitPython version

There is an Arduino version and CircuitPython version of this project, and they each use different hardware.

The CircuitPython version uses and M4 Processor running CircuitPython v5.0+ for native MP3 playback, and a Adalogger FeatherWing for its Real Time Clock and SD card.



Parts List

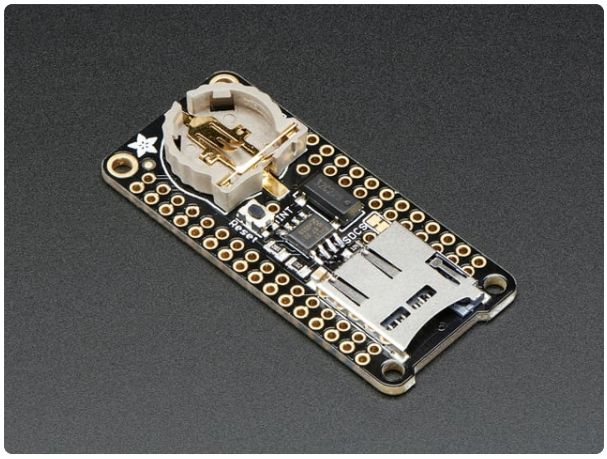


[Adafruit Feather M4 Express - Featuring ATSAMD51]

It's what you've been waiting for, the Feather M4 Express featuring ATSAMD51. This Feather is fast like a swift, smart like an owl, strong like a ox-bird (it's half ox,...

https://www.adafruit.com/product/3857

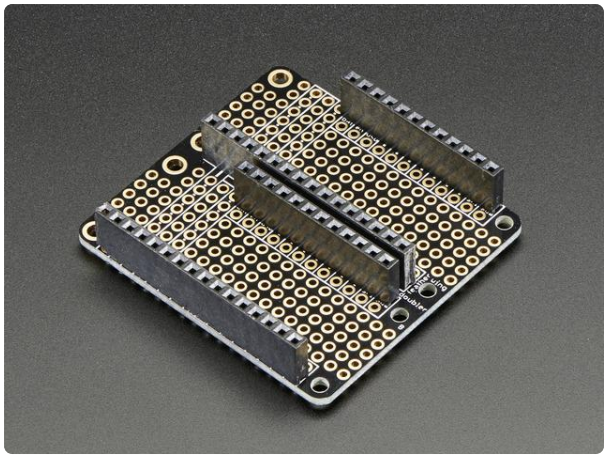
An M4 processor is required to be fast enough to play an MP3.



Adalogger FeatherWing - RTC + SD Add-on For All Feather Boards

A Feather board without ambition is a Feather board without FeatherWings! This is the Adalogger FeatherWing: it adds both a battery-backed Real Time Clock and micro SD...

https://www.adafruit.com/product/2922



FeatherWing Doubler - Prototyping Add-on For All Feather Boards

This is the FeatherWing Doubler - a prototyping add-on and more for all Feather boards. This is similar to our

https://www.adafruit.com/product/2890

SD/MicroSD Memory Card (8 GB SDHC)

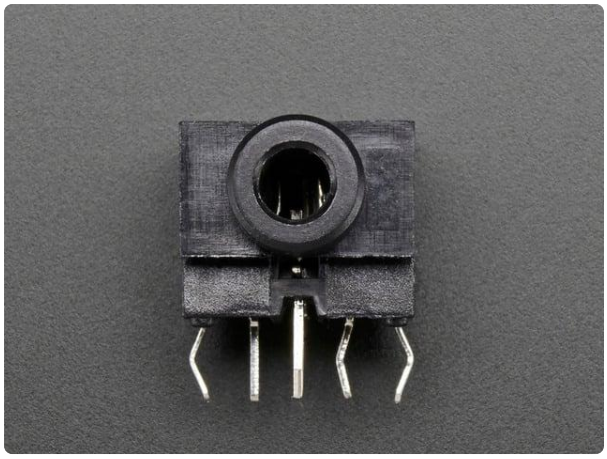
Add mega-storage in a jiffy using this 8 GB class 4 micro-SD card. It comes with a SD adapter so you can use it with any of our shields or adapters. Preformatted to FAT so it works out...

https://www.adafruit.com/product/1294

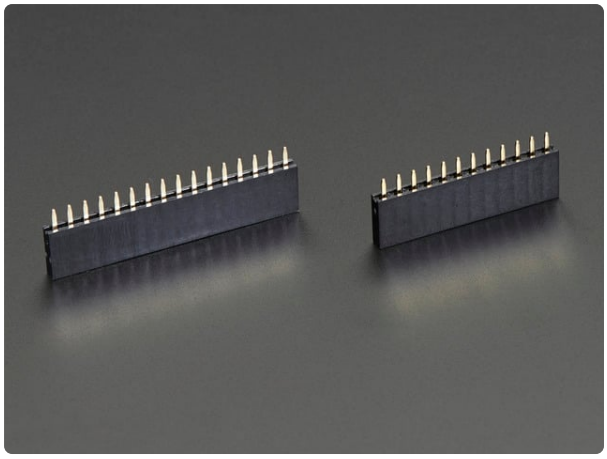
CR1220 12mm Diameter - 3V Lithium Coin Cell Battery

These are the highest quality & capacity batteries, the same as shipped with the iCufflinks, iNecklace, Datalogging and GPS Shields, GPS HAT, etc. One battery per order...

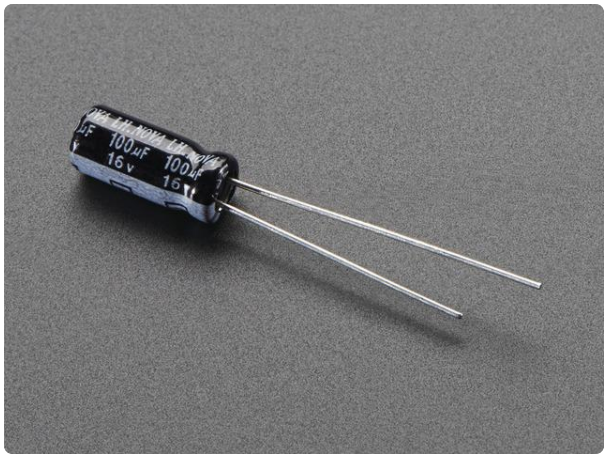
https://www.adafruit.com/product/380



[Breadboard-Friendly 3.5mm Stereo Headphone Jack]
Pipe audio in or out of your project with this very handy breadboard-friendly audio jack. It's a stereo jack with disconnect-switches on Left and Right channels as well as a center...
https://www.adafruit.com/product/1699

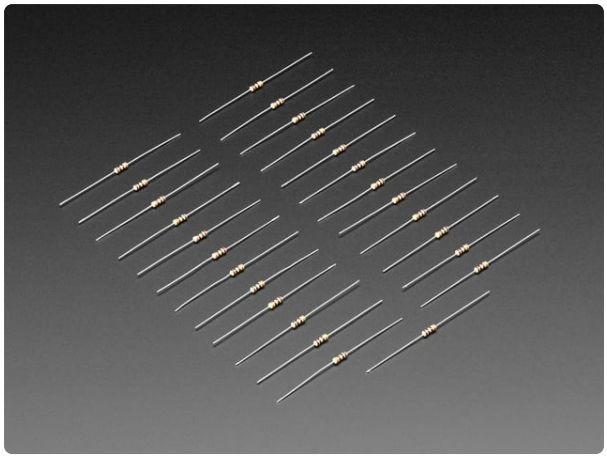


[Header Kit for Feather - 12-pin and 16-pin Female Header Set]
These two Female Headers alone are, well, lonely. But pair them with any of our
https://www.adafruit.com/product/2886



[100uF 16V Electrolytic Capacitors - Pack of 10]
We like capacitors so much we made a kids' show about them. ...
https://www.adafruit.com/product/2193

we need two 100uF capacitors



Through-Hole Resistors - 100 ohm 5% 1/4W - Pack of 25
ΩMG! You're not going to be able to resist these handy resistor packs! Well, axially, they do all of the resisting for you!This is a 25 Pack of...
https://www.adafruit.com/product/4293

we need two 100 ohm resistors

Assembly

You'll need a soldering iron and solder (https://adafru.it/drl), as well as some wire.

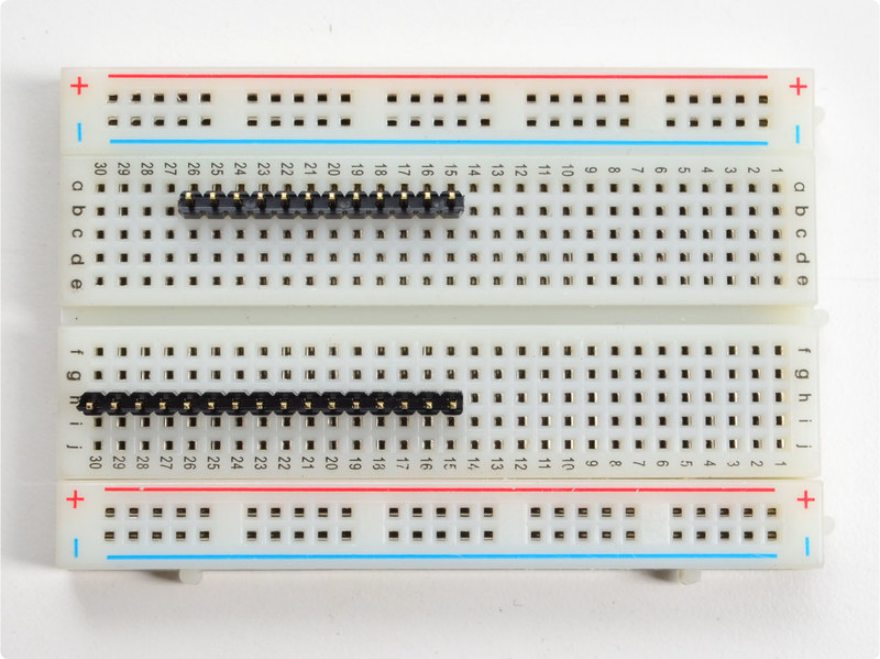
Feather Assembly

We are building this project around the FeatherWing Doubler (http://adafru.it/2890) to connect the Feather M4 Express and Adalogger FeatherWing.

You could also use Feather Stacking Headers (http://adafru.it/2830) for a different form factor, to have the boards stacked on top of each other rather than next to each other. Decide what you'd like to do, then plan ahead accordingly!

In this example, we'll solder male headers onto the Feather M4 Express and FeatherWing, and female headers onto the FeatherWing Doubler.

Follow the instructions for attaching male headers to your Feather board (https://adafru.it/L1b). It helps to use a solderless breadboard (http://adafru.it/64) to get the headers on straight.

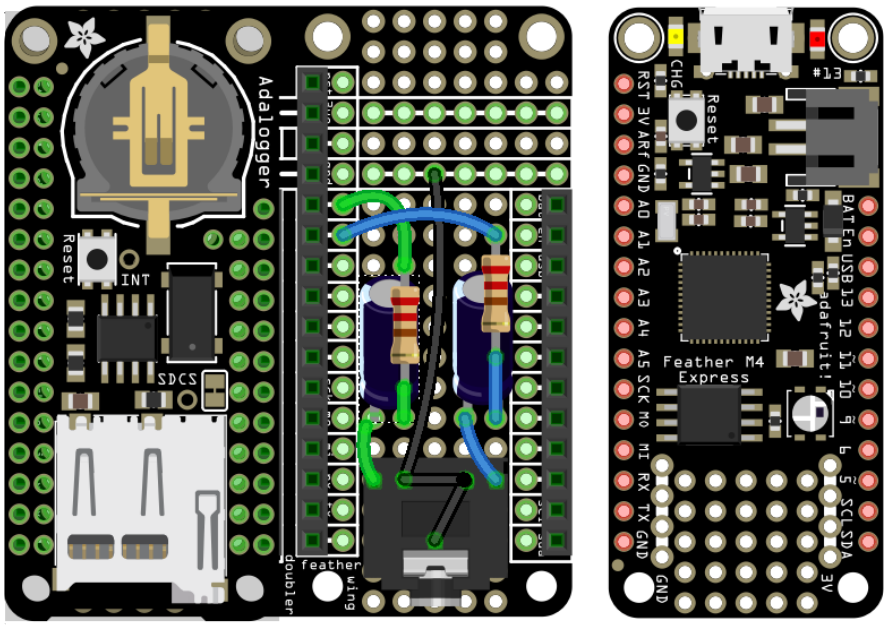


Do the same for the Adalogger FeatherWing (https://adafru.it/LBF), attaching **male headers**.

Attach **female headers** to the FeatherWing Doubler. You can use one your soldered feathers to help keep the female headers straight while you solder.

Headphone Jack

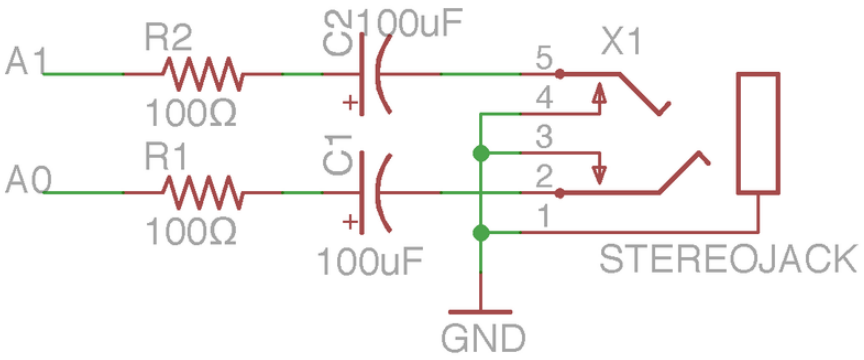
We'll be playing the audio from the Feather M4 Express through this headphone jack (http://adafru.it/1699). We'll solder the headphone jack and some other components to the FeatherWing Doubler, under where the Feather M4 Express will go. The Feather M4 Express should go into the right Feather slot, in case you'd like to attach a LiPo battery.



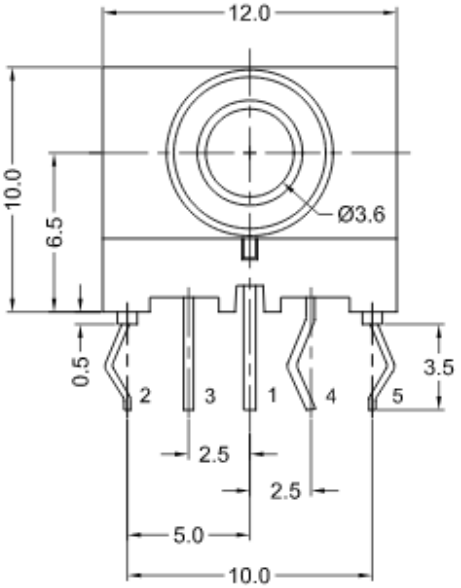
The headphone jack can fit under the Feather M4. I've moved the Feather M4 to the side so we can see where the headphone jack and other components go.

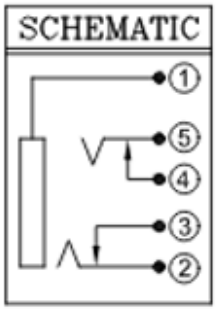
The middle three pins of the headphone will be connected to **ground**. You can connect them to the Doubler ground rail.

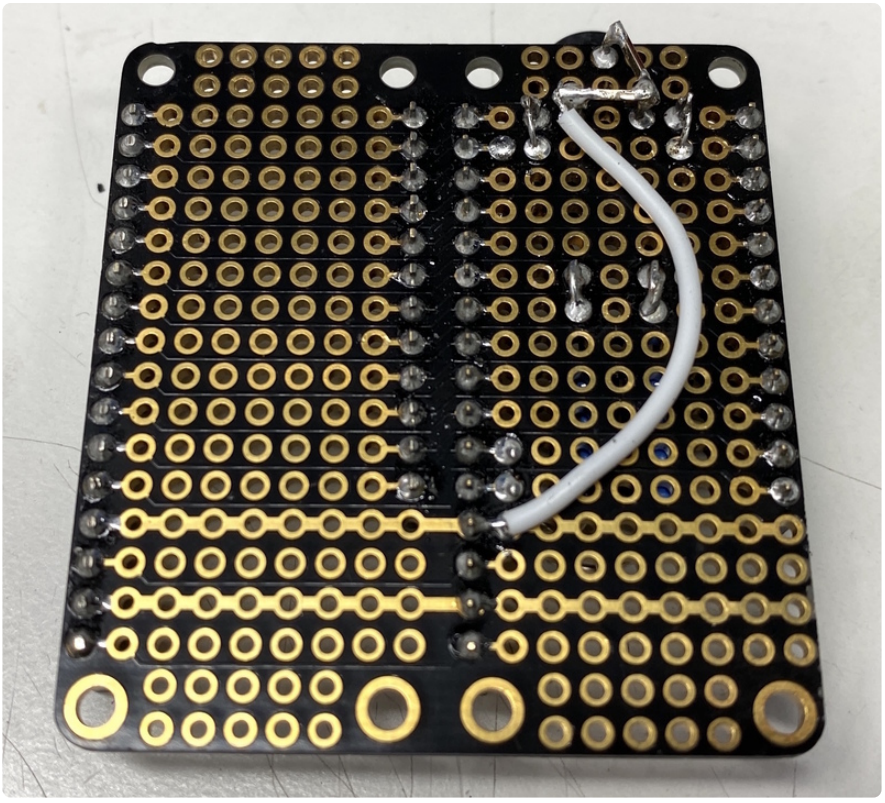
The outer pins of the headphone jack should be connected in series through a 100uF capacitor and a 100 ohm resistor to the Feather M4's A0 and A1 pins. Make sure the cathode sides of the capacitors are connected to the headphone jack. The A0 output (in green) will be the audio left channel and the A1 output (in blue) will be the audio right channel.

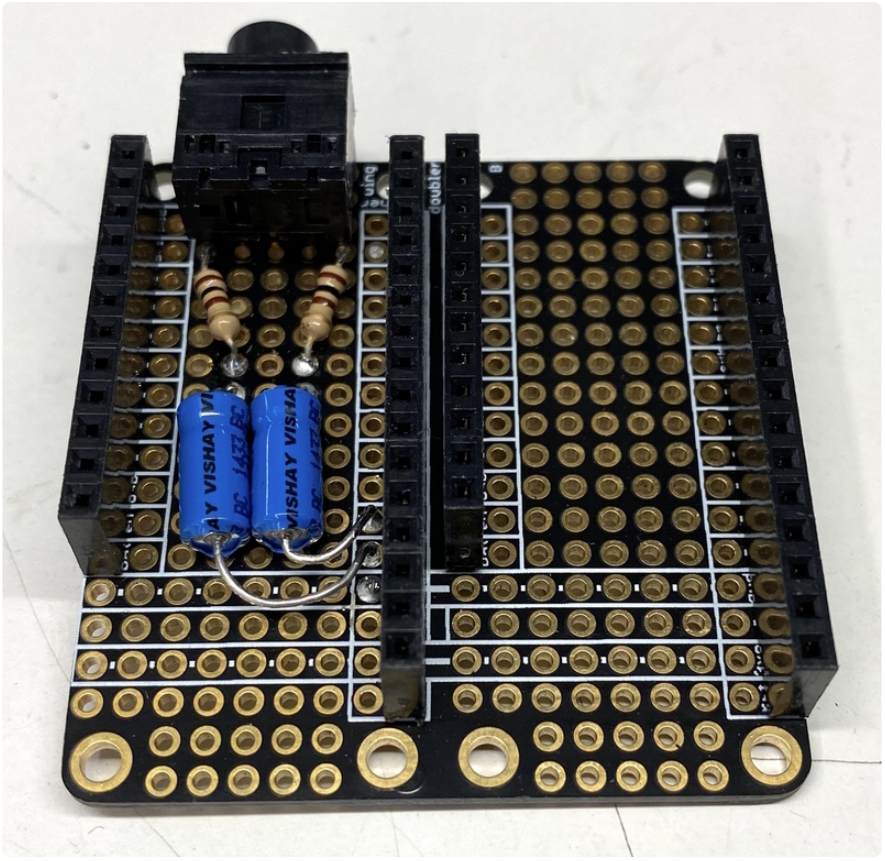


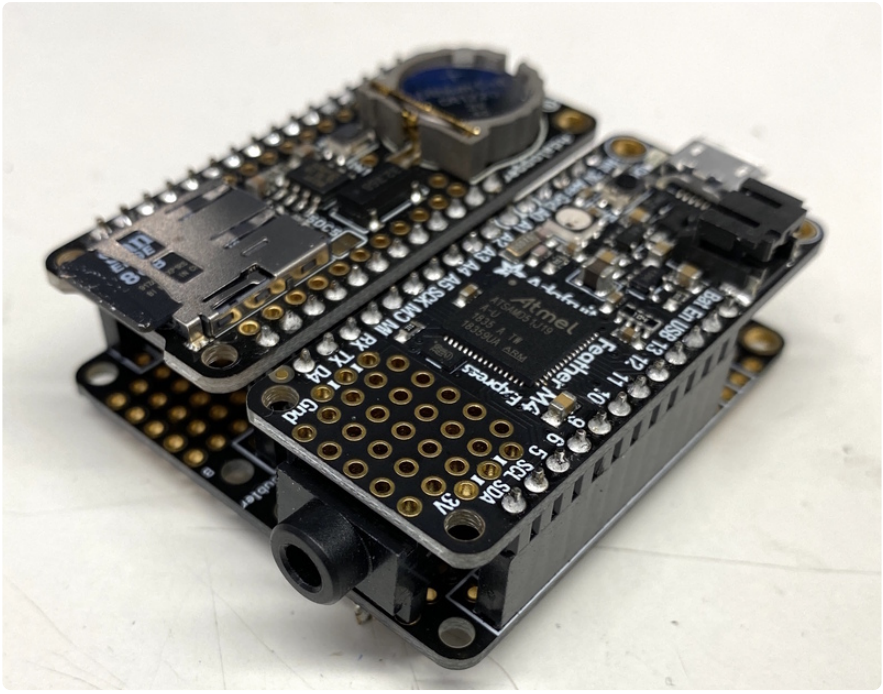
Here is a schematic of the circuit. Note that pin 1 of the stereo jack is the middle physical pin. Pins 3 and 4 are the next two inner physical pins. Pins 1, 3, and 4 are all electrically connected to ground. Note that they are **not** connected to pin 2.


12.0
10.0
6.5
Ø3.6
0.5
2
3
1
4
5
3.5
2.5
2.5
5.0
10.0


SCHEMATIC
1
5
4
3
2







Software

Hardware & Software Requirements

We've written software in CircuitPython for the Feather M4 Express. This tutorial will get you started (https://adafru.it/CVs). We need to use an M4 (ATSAMD51 32-bit Cortex M4 core) because it's fast enough to play back MP3s. We also need to use CircuitPython version 5.3.0+. Follow this tutorial to install a newer version of CircuitPython (https://adafru.it/Amd).

CircuitPython Libraries

The code requires a few libraries, which we need to use with the Adalogger FeatherWing. Download The latest bundle (https://adafru.it/ENC) and install the libraries listed below into the **lib** folder on your Feather M4 Express:

- **adafruit_bus_device** folder
- **adafruit_register** folder
- **adafruit_pcf8523.mpy**

CircuitPython Code

Download the code (https://adafru.it/L1d) from github by clicking **Download ZIP**. Uncompress the file and copy **code.py** from the **CircuitPython** folder to your Feather M4 Express **CIRCUITPY** drive.

```
# SPDX-FileCopyrightText: 2020 Andy Doro for Adafruit Industries
#
# SPDX-License-Identifier: MIT

# AUTOCHEER DEVICE
# code by Andy Doro
#
# plays an MP3 at a specific time.
#
# uses native CircuitPython mp3 playback
#
# REQUIREMENTS:
# should use M4 (or higher)
# use CircuitPython 6.0.0+
#
#
# HARDWARE:
# Feather M4 Express https://www.adafruit.com/product/3857
# Adalogger https://www.adafruit.com/product/2922
#
```

```
#
# TO DO
# ---
# - daylight saving time
# - use built-in NeoPixel as indicator
#

import os
import time
import board
import audiomp3
import audioio
import digitalio


# For hardware I2C (M0 boards) use this line:
import busio as io

# Or for software I2C (ESP8266) use this line instead:
# import bitbangio as io

# import adafruit_ds3231
import adafruit_pcf8523

# SD card
import sdcardio
import storage

# NeoPixel
# import neopixel


# Use any pin that is not taken by SPI
# For Adalogger FeatherWing: https://learn.adafruit.com/adafruit-adalogger-
featherwing/pinouts
# The SDCS pin is the chip select line:
#    On ESP8266, the SD CS pin is on GPIO 15
#    On ESP32 it's GPIO 33
#    On WICED it's GPIO PB5
#    On the nRF52832 it's GPIO 11
#    On Atmel M0, M4, 328p or 32u4 it's on GPIO 10
#    On Teensy 3.x it's on GPIO 10

SD_CS = board.D10  # for M4

# Connect to the card and mount the filesystem.
spi = io.SPI(board.SCK, board.MOSI, board.MISO)
sdcard = sdcardio.SDCard(spi, SD_CS)
vfs = storage.VfsFat(sdcard)
storage.mount(vfs, "/sd")


# Use the filesystem as normal! Our files are under /sd
# This helper function will print the contents of the SD


def print_directory(path, tabs=0):
    for file in os.listdir(path):
        stats = os.stat(path + "/" + file)
        filesize = stats[6]
        isdir = stats[0] & 0x4000

        if filesize < 1000:
            sizestr = str(filesize) + " by"
        elif filesize < 1000000:
            sizestr = "%0.1f KB" % (filesize / 1000)
        else:
            sizestr = "%0.1f MB" % (filesize / 1000000)
```

```
        prettyprintname = ""
        for _ in range(tabs):
            prettyprintname += "   "
        prettyprintname += file
        if isdir:
            prettyprintname += "/"
        print("{0:<40} Size: {1:>10}".format(prettyprintname, sizestr))

        # recursively print directory contents
        if isdir:
            print_directory(path + "/" + file, tabs + 1)


print("Files on filesystem:")
print("====================")
print_directory("/sd")


data = open("/sd/cheer.mp3", "rb")
mp3 = audiomp3.MP3Decoder(data)
# a = audioio.AudioOut(board.A0) # mono
a = audioio.AudioOut(board.A0, right_channel=board.A1)  # stereo sound through A0 &
A1


i2c = io.I2C(board.SCL, board.SDA)  # Change to the appropriate I2C clock & data
# pins here!

# Create the RTC instance:
# rtc = adafruit_ds3231.DS3231(i2c)
rtc = adafruit_pcf8523.PCF8523(i2c)

# Lookup table for names of days (nicer printing).
days = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
"Sunday")


# selected time
# 24 hour time
playhour = 19
playmin = 0

# pylint: disable-msg=bad-whitespace
# pylint: disable-msg=using-constant-test
# no DST adjustment yet!
if False:  # change to True if you want to set the time!
    #                     year, mon, date, hour, min, sec, wday, yday, isdst
    t = time.struct_time((2020, 5, 13, 15, 15, 15, 0, -1, -1))
    # you must set year, mon, date, hour, min, sec and weekday
    # yearday is not supported, isdst can be set but we don't do anything with it
at this time
    print("Setting time to:", t)  # uncomment for debugging
    rtc.datetime = t
    print()
# pylint: enable-msg=using-constant-test
# pylint: enable-msg=bad-whitespace

# setup NeoPixel
# pixel = neopixel.NeoPixel(board.NEOPIXEL, 1)


# Main loop:
while True:
    t = rtc.datetime

    # print(t)     # uncomment for debugging
    print(
        "The date is {} {}/{}/{}".format(
            days[int(t.tm_wday)], t.tm_mday, t.tm_mon, t.tm_year
```

```
        )
    )
    print("The time is {}:{:02}:{:02}".format(t.tm_hour, t.tm_min, t.tm_sec))

    if t.tm_hour == playhour and t.tm_min == playmin:
        print("it is time!")
        # turn NeoPixel green
        # pixel[0] = (0, 255, 0)
        # play the file
        print("playing")
        a.play(mp3)
        while a.playing:
            pass
        print("stopped")
        # turn NeoPixel off
        # pixel[0] = (0, 0, 0)

    time.sleep(1)  # wait a second
```

Audio File & SD Card

You'll also need an audio file for the Feather M4 to play! We like this sample of a crowd cheering for front-line medical staff (https://adafru.it/L1f).

The Feather M4 doesn't have enough storage space for this audio file, so we'll need to use an SD card. Rename the audio file "cheer.mp3" and make sure your SD card is FAT32 formatted (https://adafru.it/L1A). Copy the file onto the SD card and insert it into the Adalogger FeatherWing.

Coin Cell

The Adalogger RTC FeatherWing needs a CR1220 coin cell installed in order to work properly and keep time when the device is unpowered.

Understanding the Code

Conceptually, the daily cheering device is simple: the Feather M4 Express is attached to a clock. Every second, the Feather M4 Express checks the time. If the hour and minute match the appointed time, the Feather M4 Express will play an MP3 off the SD card.

The selected time is defined in the code here:

```
# selected time
# 24 hour time
playhour = 19
playmin = 0
```

The RTC uses 24 hour time (https://adafru.it/L1B), so 7 PM is 19:00.

Serial Monitor

You can use Mu Editor (https://adafru.it/ANO) or another program to examine the serial output from the Feather M4.

SD Card & Audio File

On startup, the program will mount the SD card and print its contents (https://adafru.it/L1A). This is a good way to troubleshoot the SD card.

It will find the specified audio file and set the file to play stereo audio on the A0 and A1 pins.

```
data = open("/sd/cheer.mp3", "rb")
mp3 = audiomp3.MP3Decoder(data)
#a = audioio.AudioOut(board.A0) # mono
a = audioio.AudioOut(board.A0, right_channel=board.A1) # stereo sound through A0
&amp;amp; A1
```

RTC (Real Time Clock) & Setting The Time

Afterwards, you should see the date and time being printed every second.

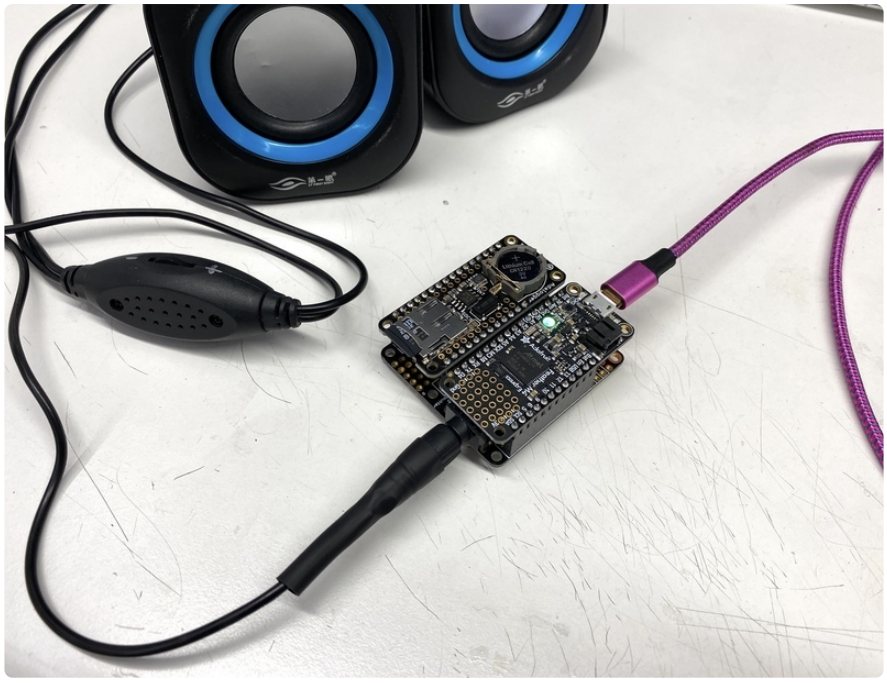
```
# Main loop:
while True:
    t = rtc.datetime

    # print(t)     # uncomment for debugging
    print(
        "The date is {} {}/{}/{}".format(
            days[int(t.tm_wday)], t.tm_mday, t.tm_mon, t.tm_year
        )
    )
    print("The time is {}:{:02}:{:02}".format(t.tm_hour, t.tm_min, t.tm_sec))
```

If the time is incorrect, you can set it in the following section of code by changing `False` to `True`:

```
if False:  # change to True if you want to set the time!
    #                     year, mon, date, hour, min, sec, wday, yday, isdst
    t = time.struct_time((2020, 5, 13, 15, 15, 15, 0, -1, -1))
    # you must set year, mon, date, hour, min, sec and weekday
    # yearday is not supported, isdst can be set but we don't do anything with it
at this time
    print("Setting time to:", t)  # uncomment for debugging
    rtc.datetime = t
    print()
```

This will change the time on the Adalogger FeatherWing's RTC. The RTC runs off the CR1220 coin cell battery when it isn't attached to power, which is how it continues to keep track of the time when the Feather is unpowered. Set `True` back to `False` once the time is set correctly. You can read more about the RTC and setting the time in the Adalogger FeatherWing Learn Guide (https://adafru.it/Lbl).

Usage



Power the automaton through the Feather's microUSB port. These 5V power supplies with microUSB (http://adafru.it/1995) work well.

With the audio jack output you can easily connect the device to a stereo amplifier or computer speakers for a louder volume.

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

[3.5mm Stereo Male/Male Cable in Various Metal Colors - 1 meter]
Here is a gorgeous metal-covered audio cable straight from Blade Runner/ cyberpunk heaven. And we have them in a couple different colors! They're a step up from plain rubber/ABS...
https://www.adafruit.com/product/4070

[5V 2.5A Switching Power Supply with 20AWG MicroUSB Cable]
Our all-in-one 5V 2.5 Amp + MicroUSB cable power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's...
https://www.adafruit.com/product/1995

Going Further

- You can modify this project to perform different tasks at different times throughout the day, such as participate in Pride Cheer (https://adafru.it/MeO) at 1pm, play howling noises at 8pm (https://adafru.it/L1E), or even play "Happy Birthday" on a specific date.
- Using a Motor FeatherWing (http://adafru.it/2927) you could expand this project to have the device operate physical noisemakers, such as banging pots and pans together, or clapping animatronic hands. Or you could create a cuckoo clock!