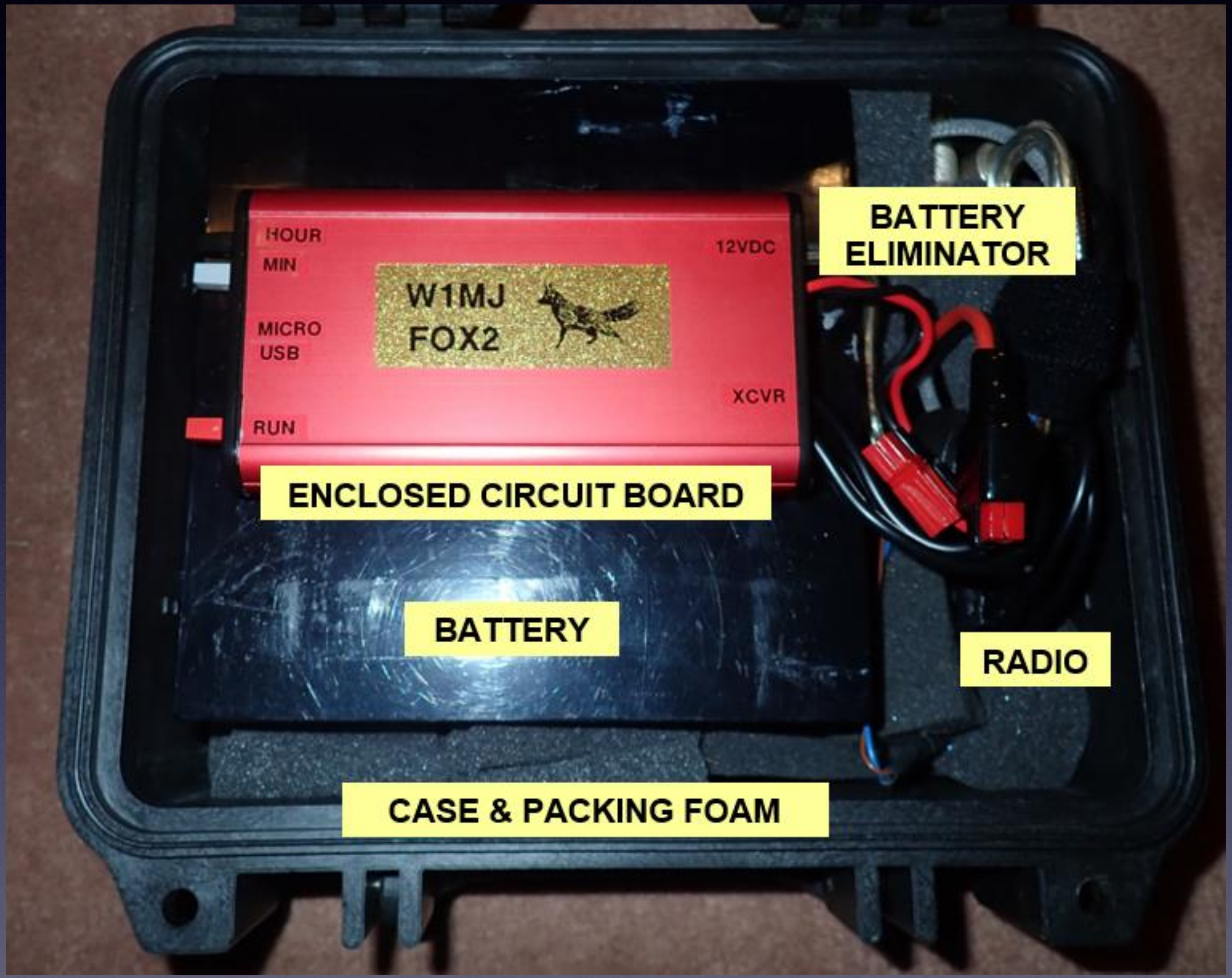
W1MJ Fox Design

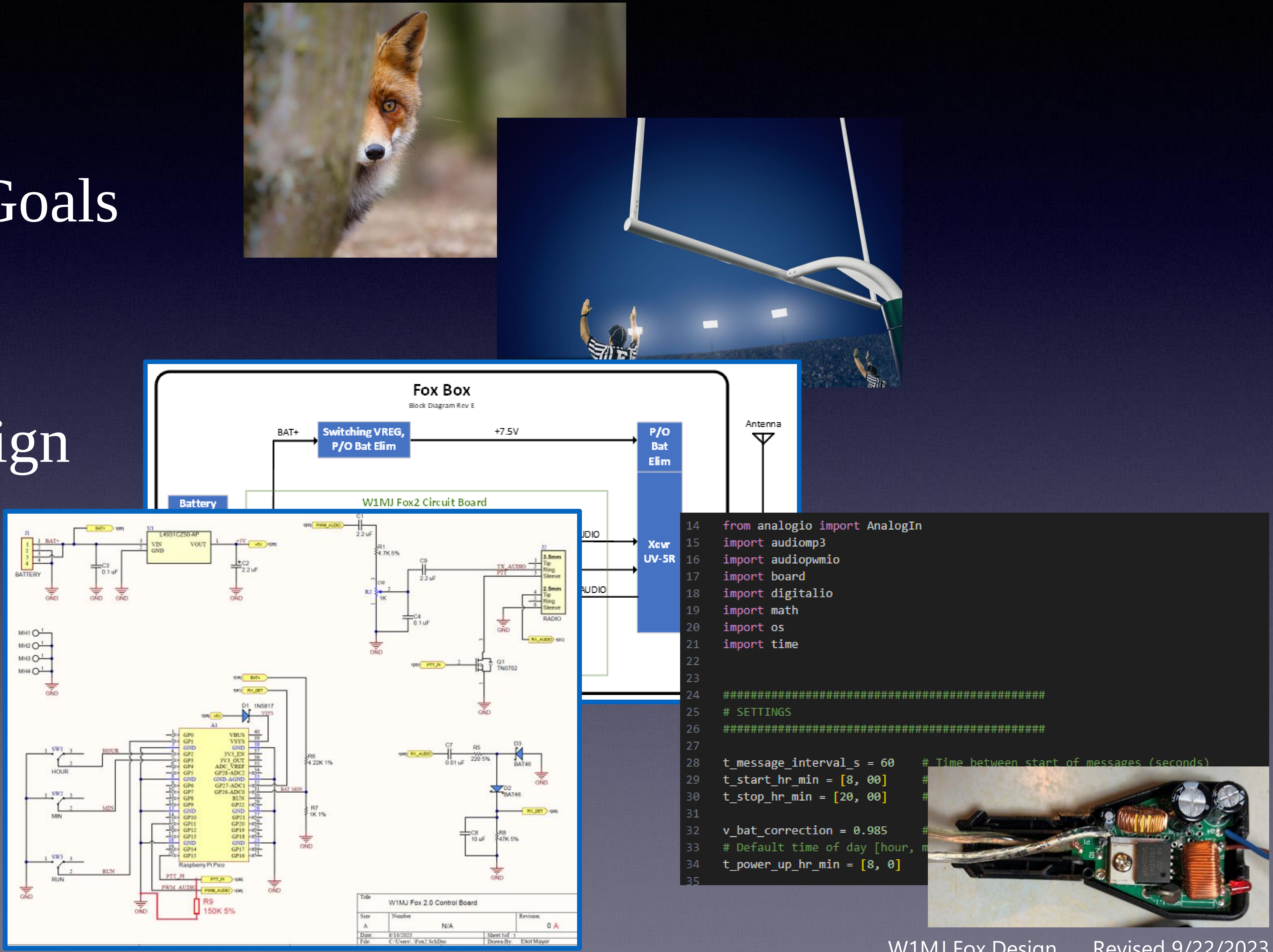
Presented by Eliot Mayer, W1MJ
Waltham Amateur Radio Association
September 27, 2023



W1MJ Second-Generation Fox

Workshop Outline

- What is a Fox?
- W1MJ FOX2 Design Goals
- Design Overview
- Fox Circuit Board Design
- Software Design
- Lessons Learned



```
from analogio import AnalogIn
import audiomp3
import audiopwmio
import board
import digitalio
import math
import os
import time


###############################################
# SETTINGS
###############################################

t_message_interval_s = 60    # Time between start of messages (seconds)
t_start_hr_min = [8, 00]     #
t_stop_hr_min = [20, 00]     #

v_bat_correction = 0.985     #
# Default time of day [hour, m
t_power_up_hr_min = [8, 0]
```

What is a Fox?

Some radio amateurs participate in hidden transmitter hunts for fun or for training, preferably both.

Hidden transmitter hunts are sometimes called fox hunts. The fox is the hidden transmitter. It could be a radio ham hiding in the woods, but these days is more frequently an automated transmitter.

Serious Fun!
K1PJW finds joy in locating W1MJ's Fox

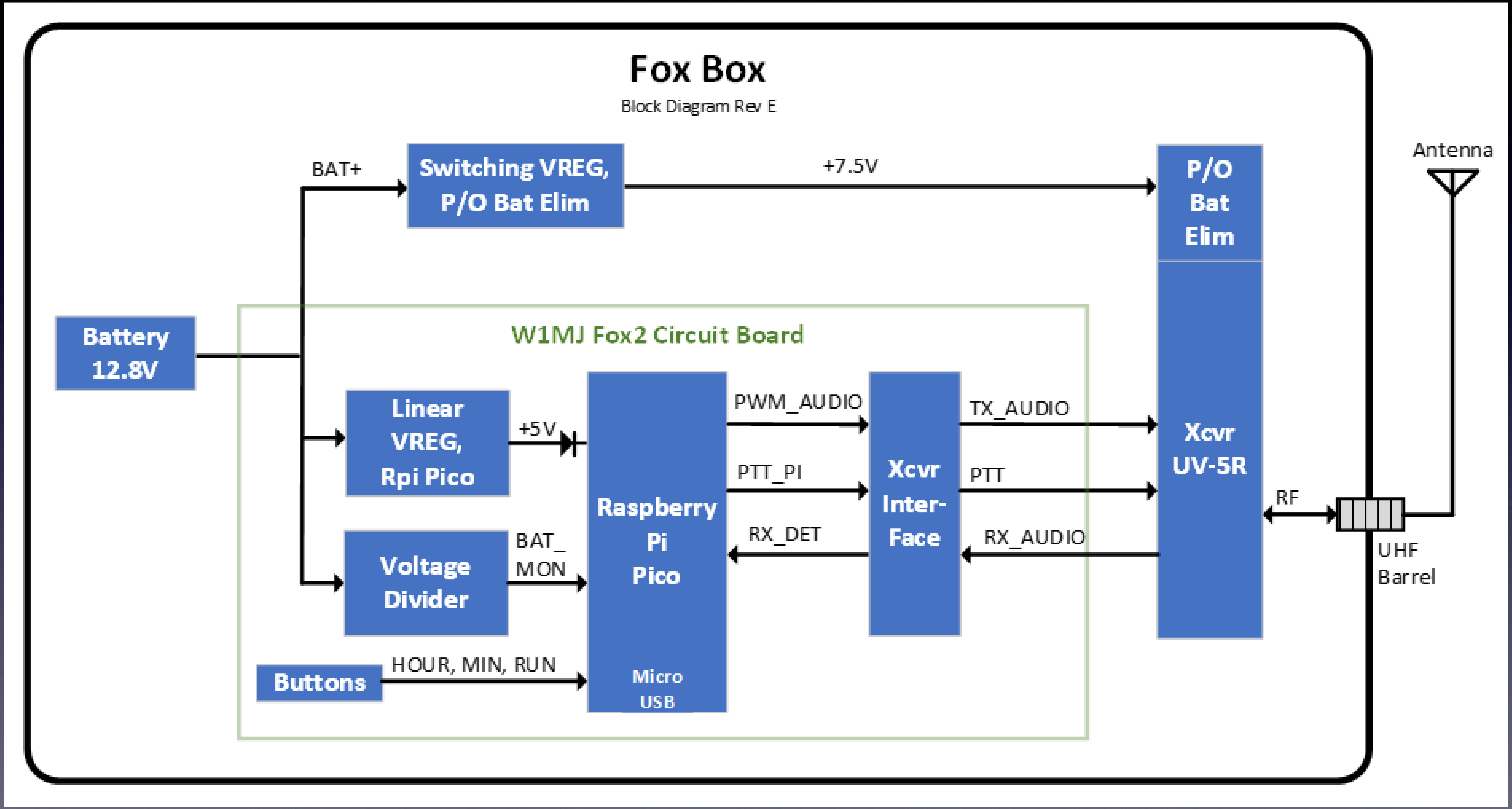
Design Goals, W1MJ FOX2

First-generation fox used an MP3 player with a very simple interface to a Baofeng radio.

Goals for second-generation fox:

- Longer deployments by going to sleep at night or transmitting on demand
- Announce battery voltage
- Stop transmitting when battery is low

Design Overview

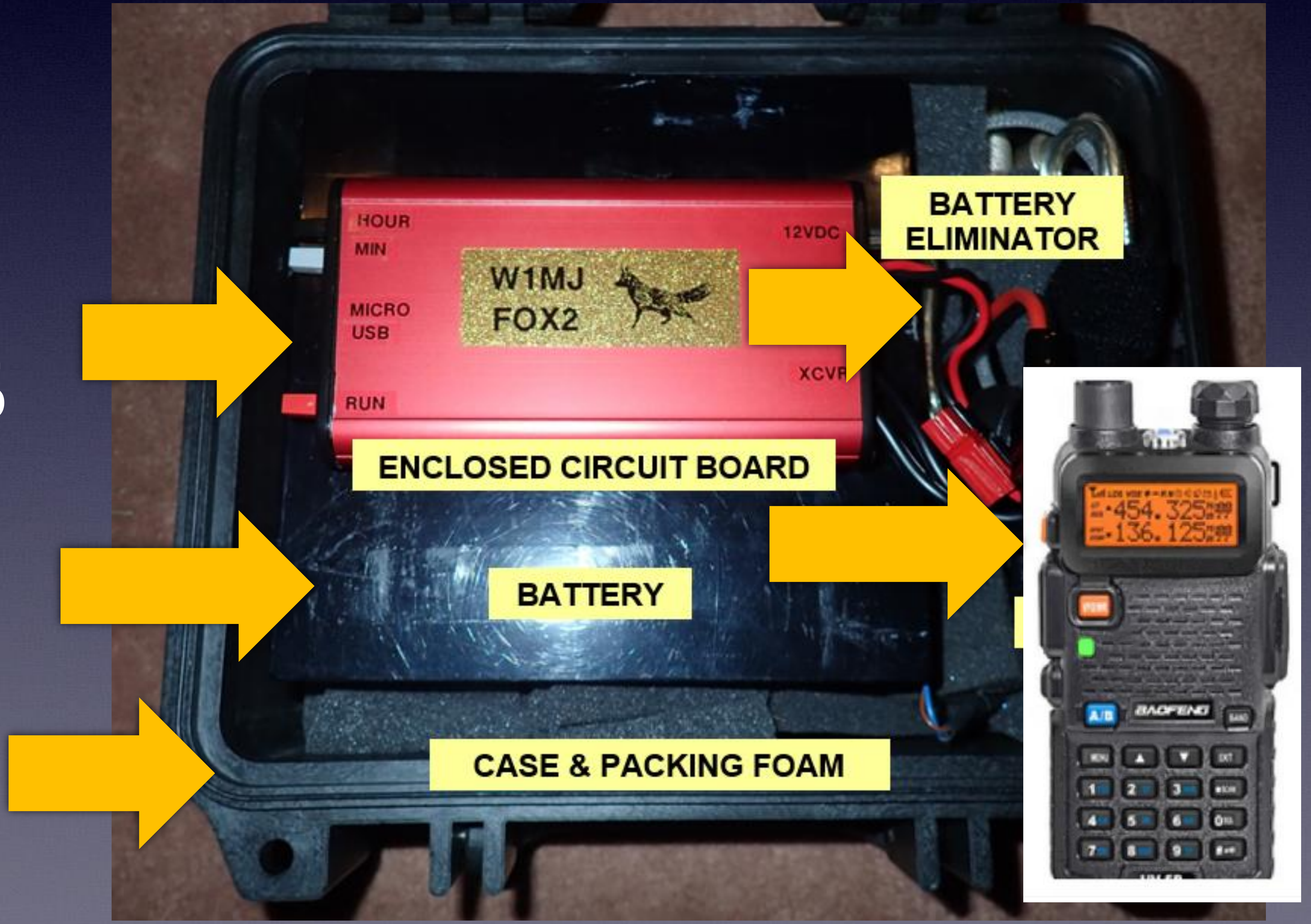


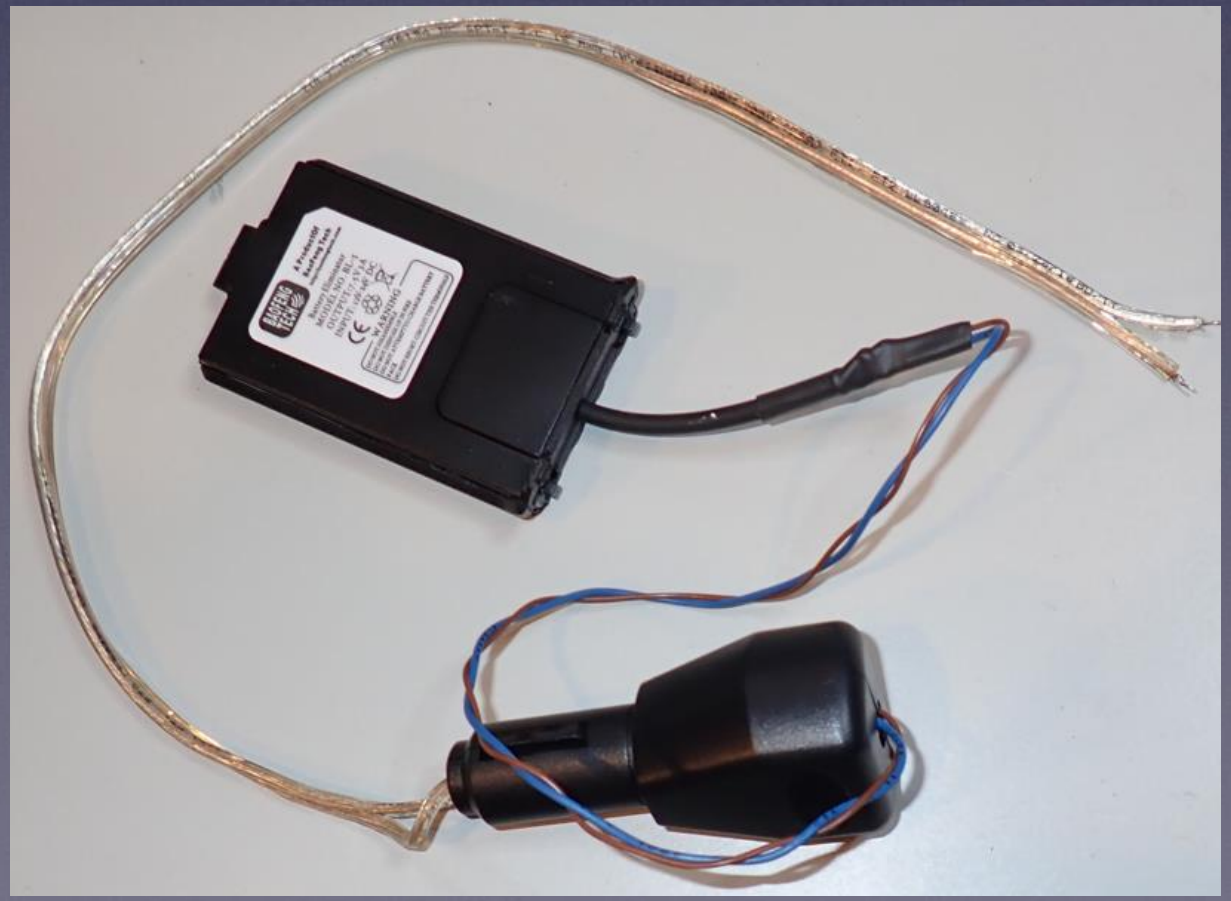
Block Diagram, W1MJ FOX2

Design Overview

Main Components

- Case: Pelican 1200 Black
- Battery: 12.8V 30AH LiFePO4
- Radio: Baofeng UV-5R
- Battery Eliminator: Baofeng Tech BL-5
- Circuit Board

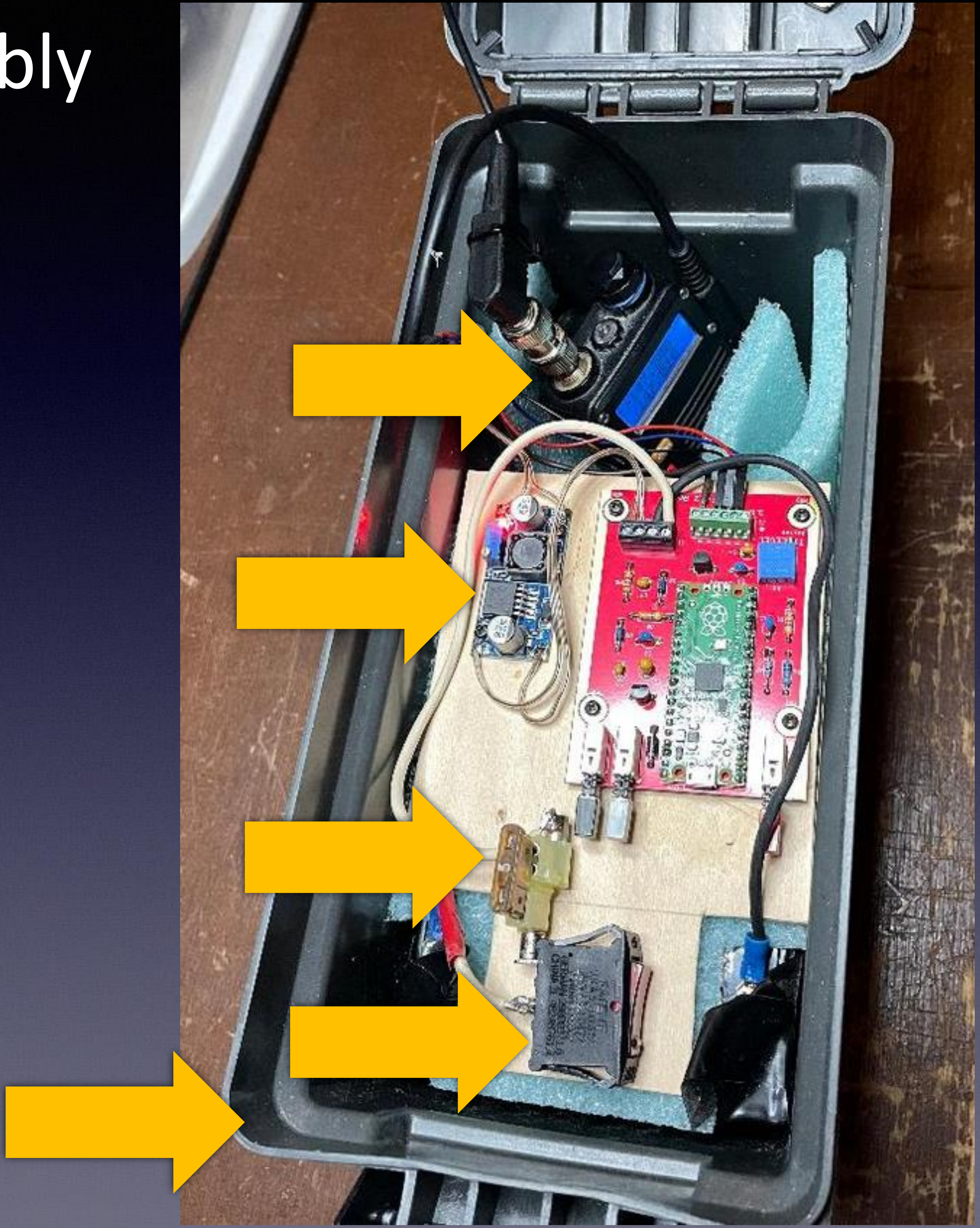




Design Overview

K1MJC’s Fox Assembly

- Case: Harbor Freight 0.30 Cal Ammo Box
- Has Power Switch
- Has Fuse
- Baofeng F8HP Radio
- 12V 15AH Lead Acid Battery
- DZS Elec Voltage Regulator



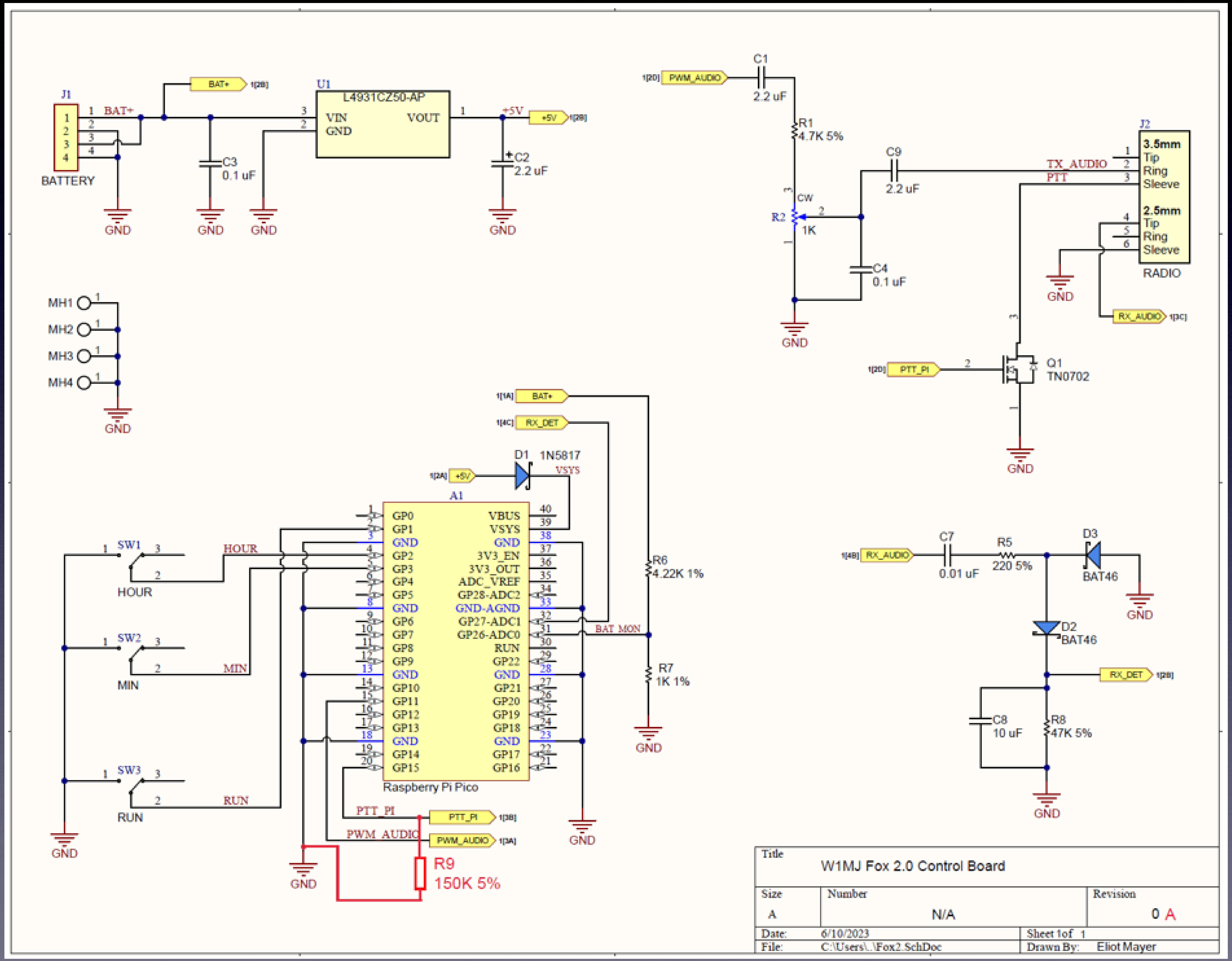
Design Overview

Why Raspberry Pi Pico?

- Heard about it at a QSO Today Expo tech forum
- Capable of implementing the design goals
- Low power consumption: 5V x 26 mA = 130 mW
- Small: 51 x 21 mm
- Inexpensive: $4

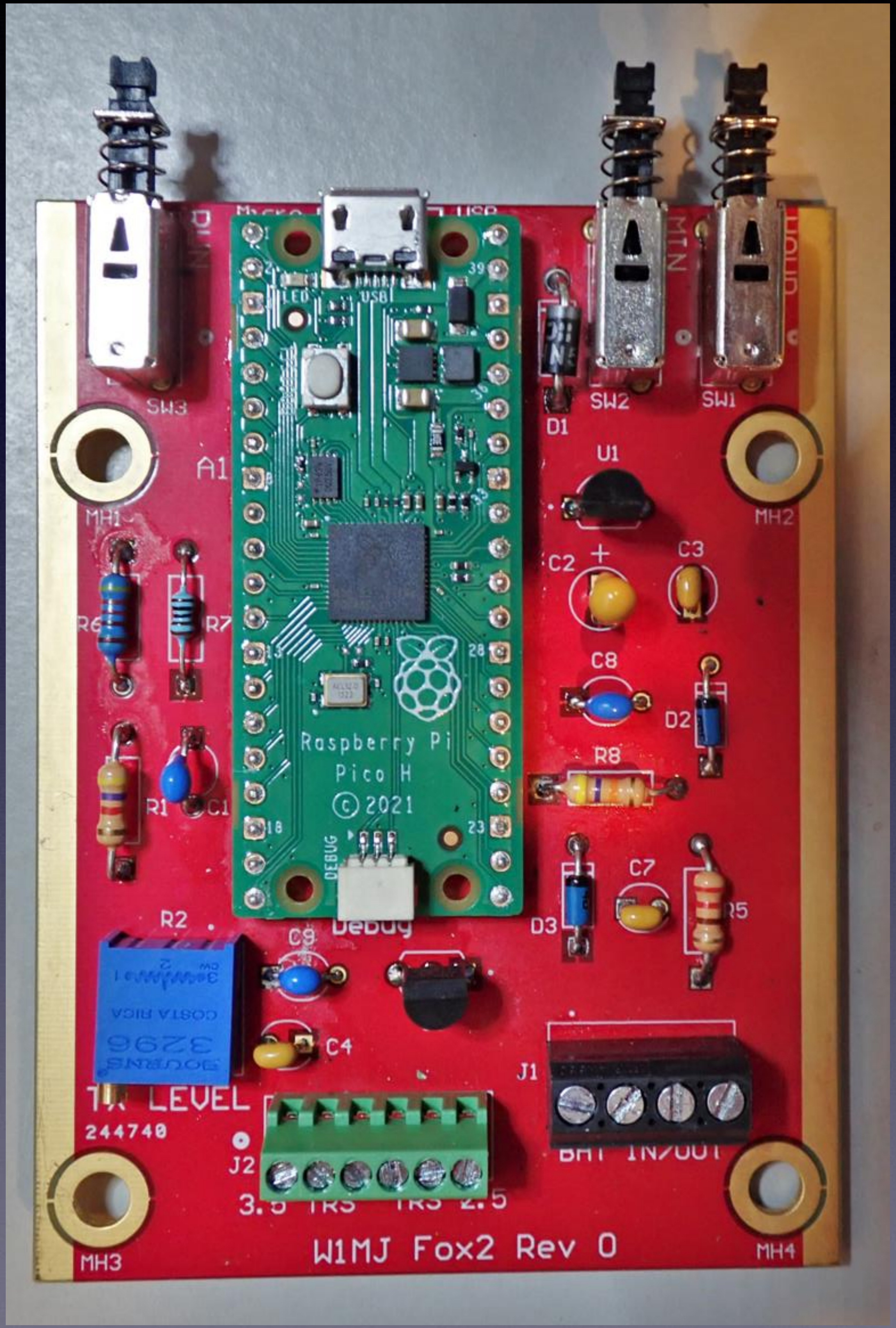
Waltham Amateur Radio Association Est. 1938

Fox Circuit Board Design

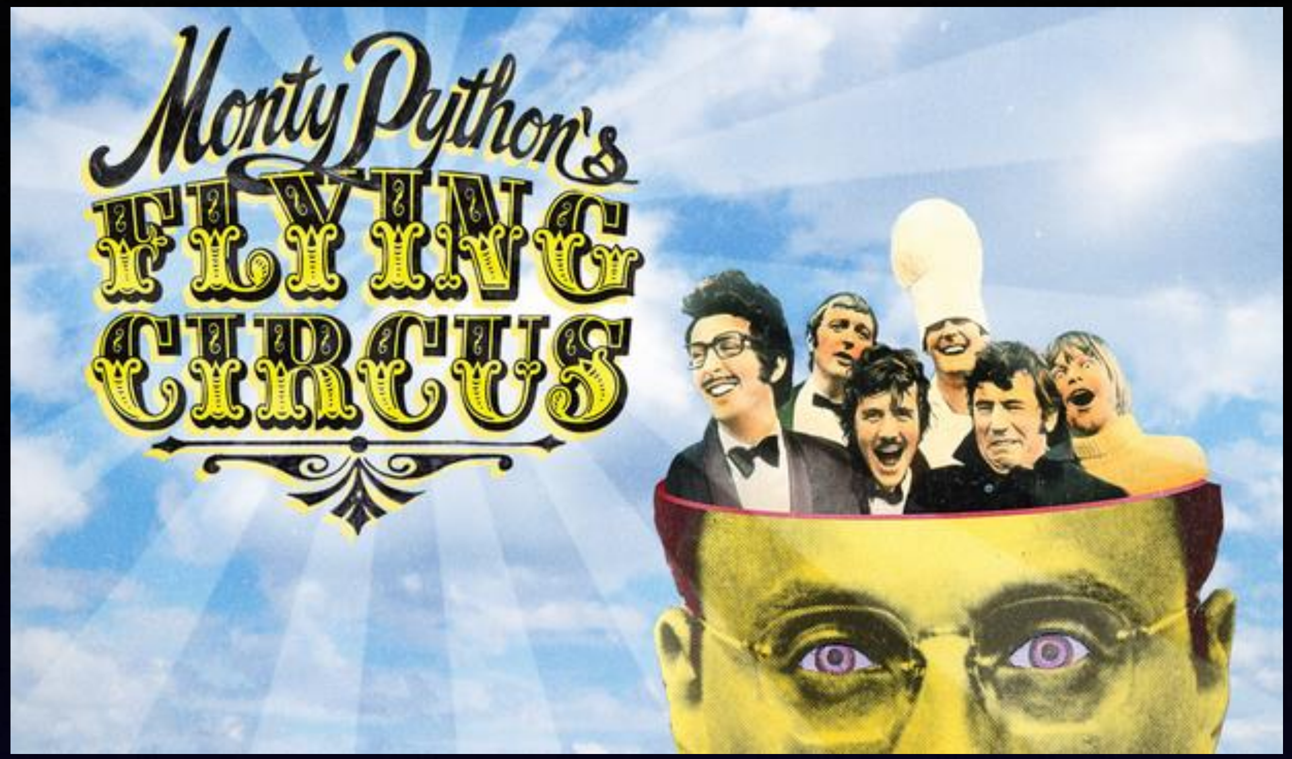


Fox Circuit Board Design

- Schematic: Altium Designer
- PCB Layout: Altium Designer
- Fabrication: Digi-Key DKRed

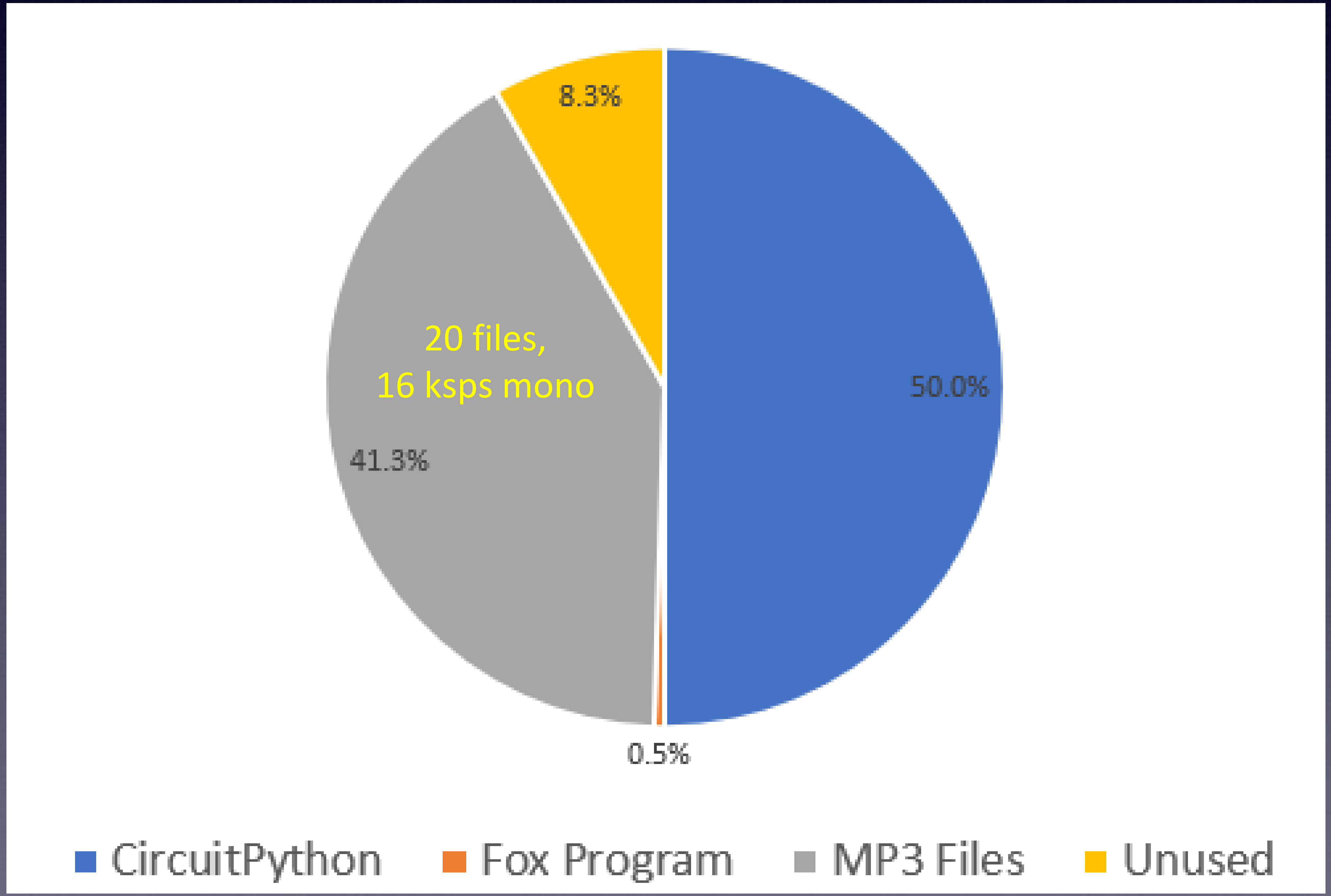


Software Design



Why CircuitPython?

- Python is popular, easy to use, and free. I had taken a course.
- CircuitPython has everything needed for this fox, including an MP3 decoding function.
- Python is actually named after Monte Python’s Flying Circus!



Usage of Pico’s 2MB Storage

Software Design

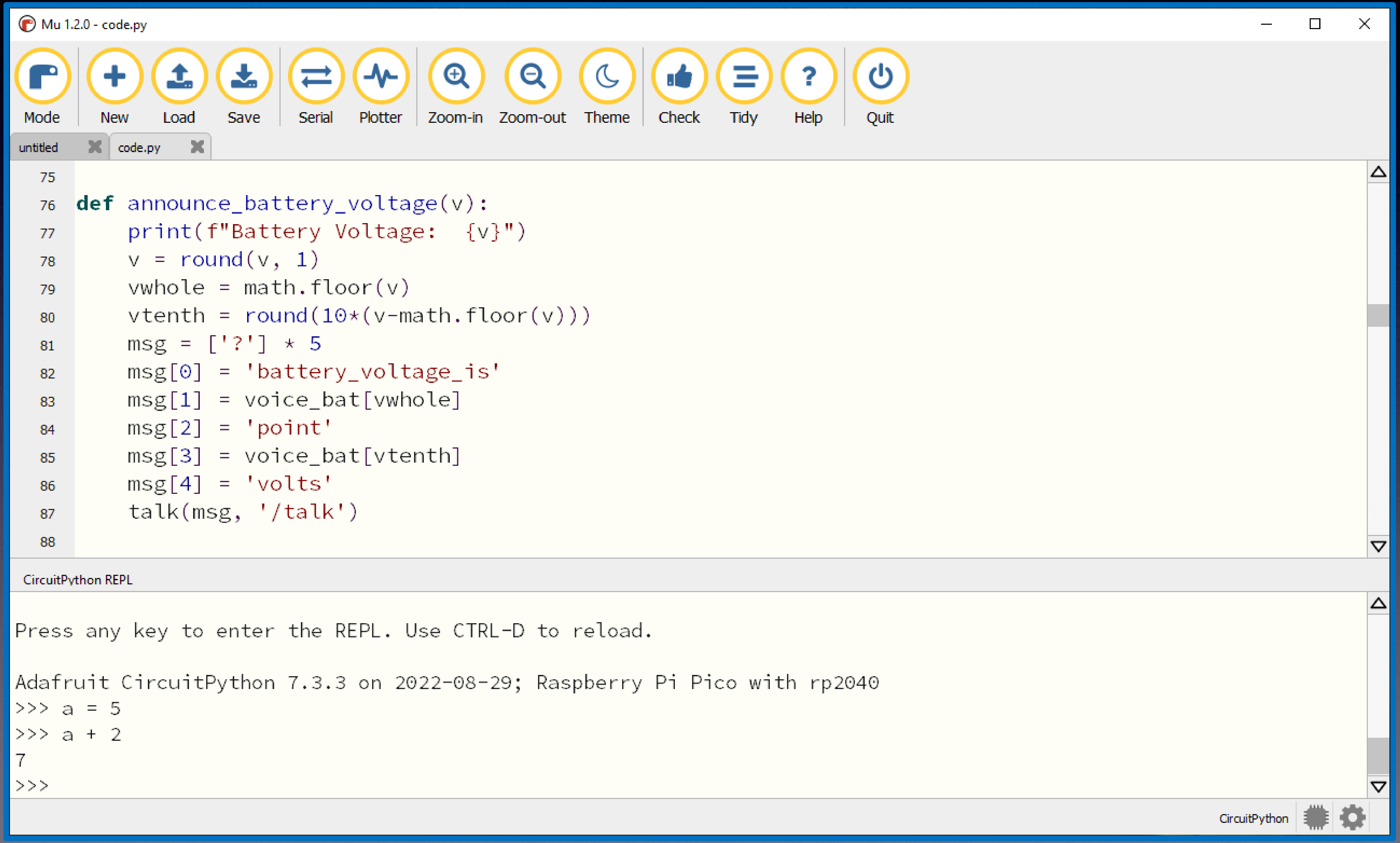
Program Outline

1. Import modules
2. Settings (message interval, start & stop times, etc)
3. Define Functions
4. Initialization
   a. IO Port assignments
   b. Miscellaneous initialization tasks
5. Set time (buttons, voice feedback via radio)

   Buttons: Hour, Min (5-min increments), Run (long push for On Demand Mode)
6. Message Loop
   a. During scheduled time, with OK battery voltage
      i. Transmit all messages sequentially
      ii. Measure and transmit battery voltage
      iii. Repeat
   b. In On Demand Mode, wait for new request

Software Design

Coding and Debugging: Mu Editor

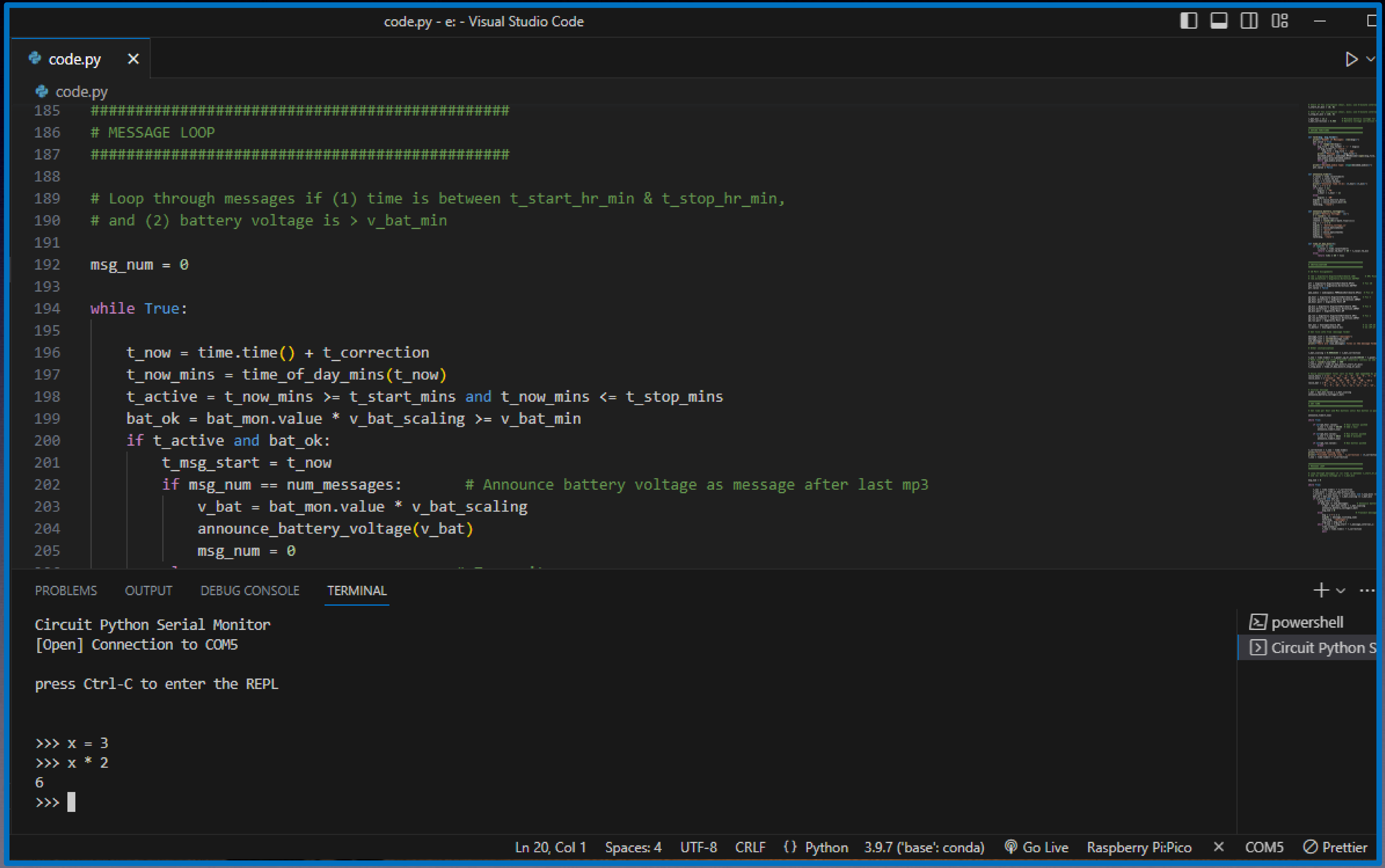


```

def announce_battery_voltage(v):
    print(f"Battery Voltage:  {v}")
    v = round(v, 1)
    vwhole = math.floor(v)
    vtenth = round(10*(v-math.floor(v)))
    msg = ['?'] * 5
    msg[0] = 'battery_voltage_is'
    msg[1] = voice_bat[vwhole]
    msg[2] = 'point'
    msg[3] = voice_bat[vtenth]
    msg[4] = 'volts'
    talk(msg, '/talk')

```

```
Press any key to enter the REPL. Use CTRL-D to reload.

Adafruit CircuitPython 7.3.3 on 2022-08-29; Raspberry Pi Pico with rp2040
>>> a = 5
>>> a + 2
7
>>>
```

Software Design

Coding and Debugging: Visual Studio Code

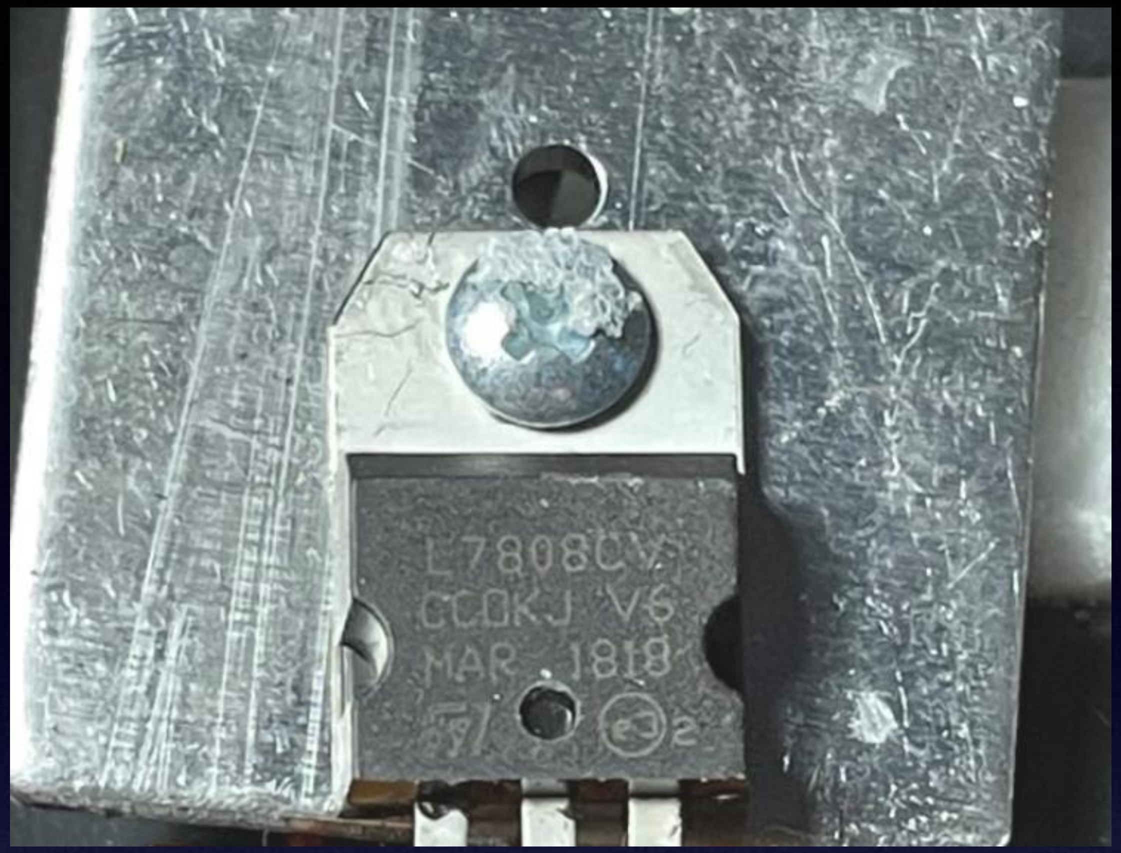


```python
###############################################
# MESSAGE LOOP
###############################################

# Loop through messages if (1) time is between t_start_hr_min & t_stop_hr_min,
# and (2) battery voltage is > v_bat_min

msg_num = 0

while True:

    t_now = time.time() + t_correction
    t_now_mins = time_of_day_mins(t_now)
    t_active = t_now_mins >= t_start_mins and t_now_mins <= t_stop_mins
    bat_ok = bat_mon.value * v_bat_scaling >= v_bat_min
    if t_active and bat_ok:
        t_msg_start = t_now
        if msg_num == num_messages:       # Announce battery voltage as message after last mp3
            v_bat = bat_mon.value * v_bat_scaling
            announce_battery_voltage(v_bat)
            msg_num = 0
```

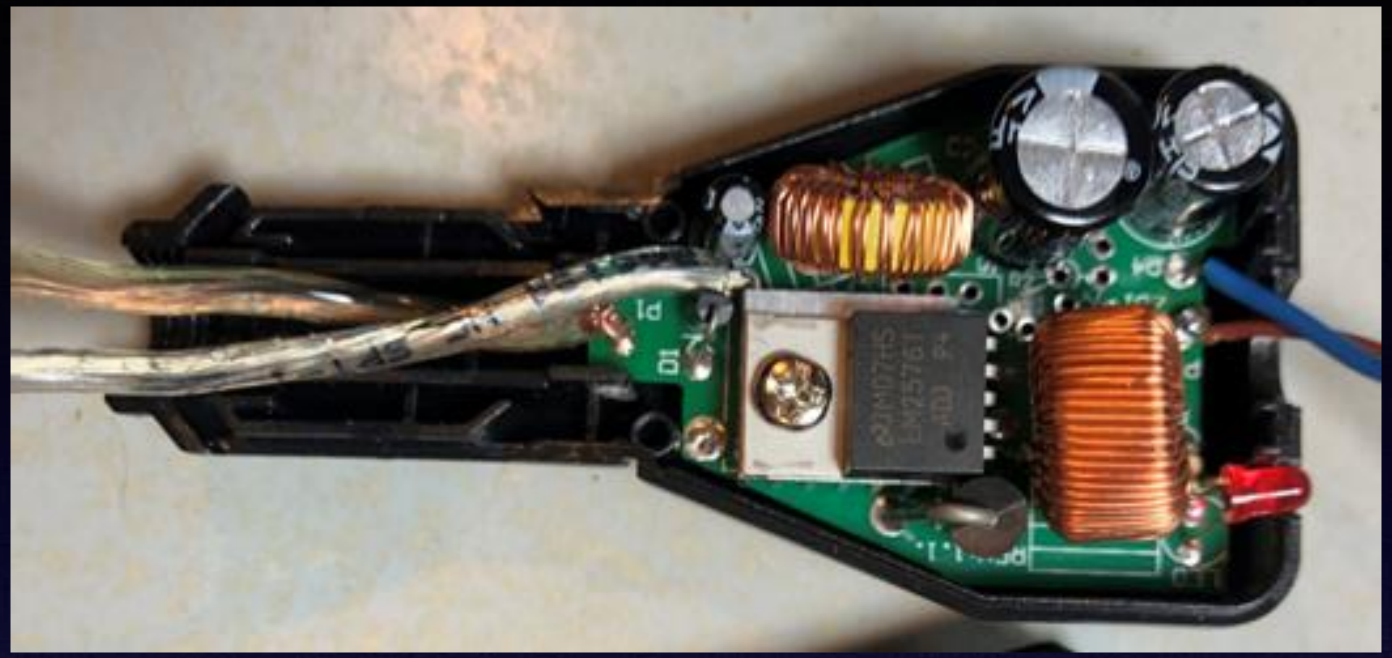
```
Circuit Python Serial Monitor
[Open] Connection to COM5

press Ctrl-C to enter the REPL


>>> x = 3
>>> x * 2
6
>>> 
```

Lessons Learned

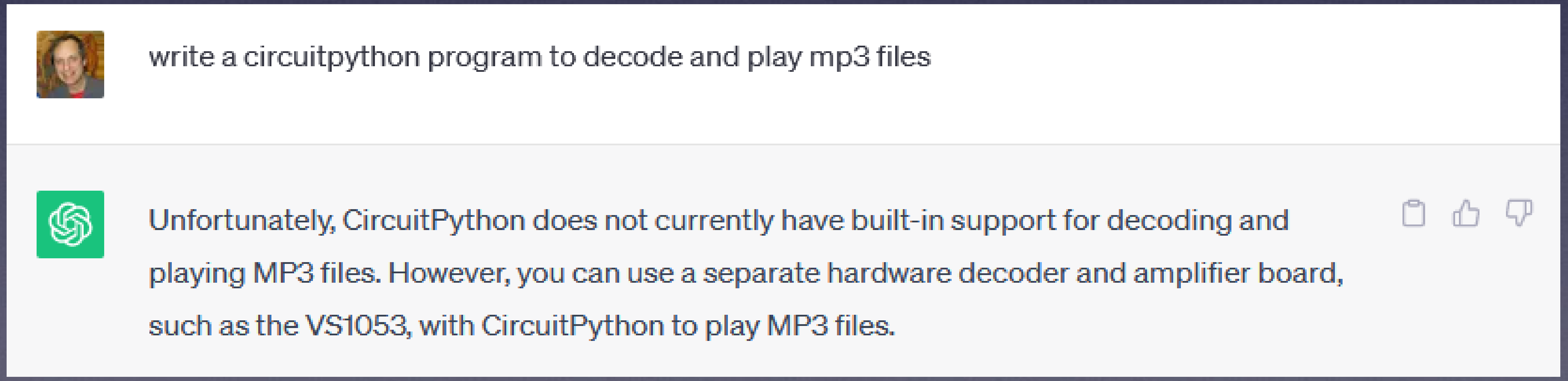
- Don’t Assume
- Use of AI – Caveat Emptor



Linear Regulator in K1MJC’s Battery Eliminator



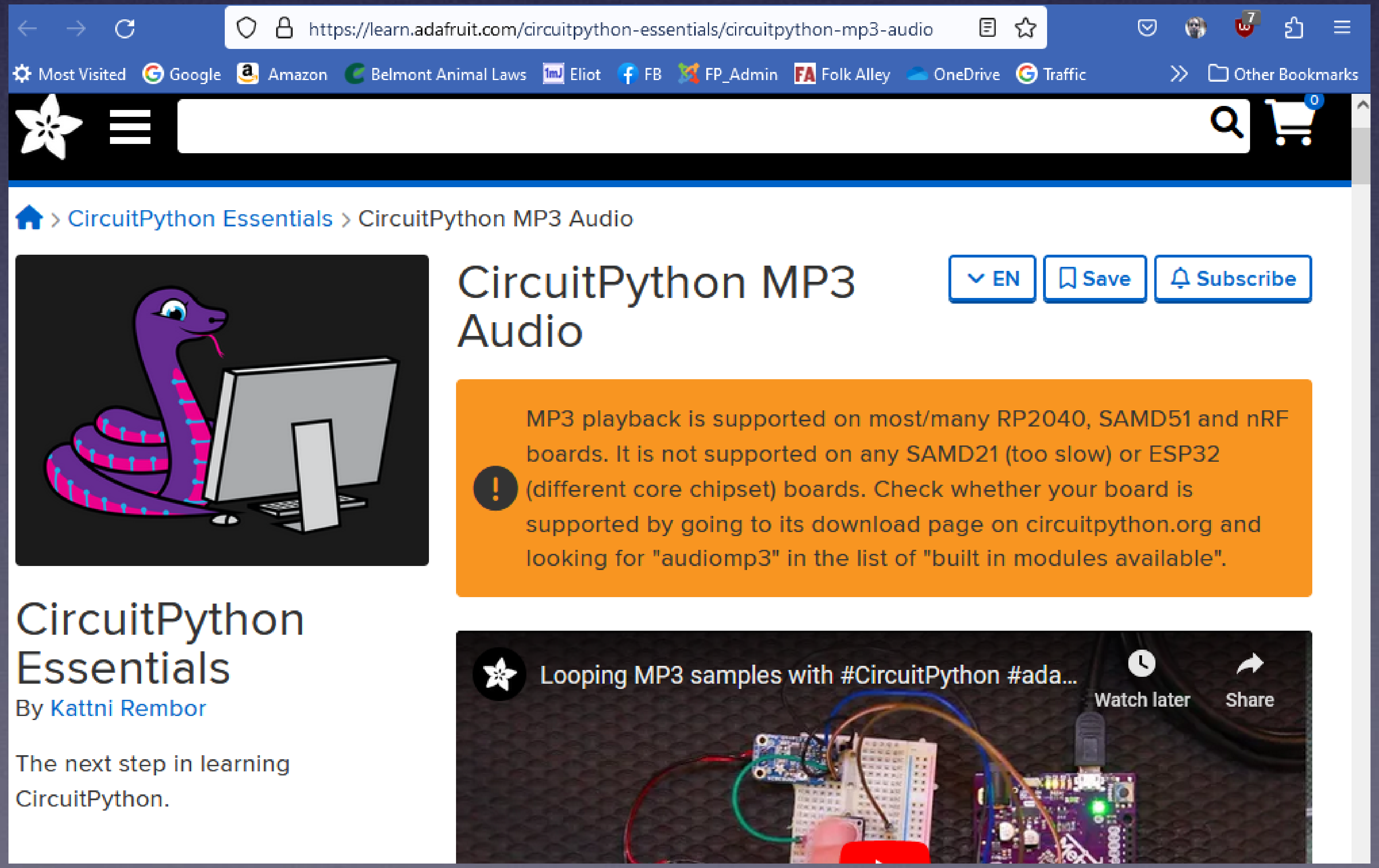
Switching Regulator in Baofeng BL-5 Battery Eliminator



write a circuitpython program to decode and play mp3 files

Unfortunately, CircuitPython does not currently have built-in support for decoding and playing MP3 files. However, you can use a separate hardware decoder and amplifier board, such as the VS1053, with CircuitPython to play MP3 files.

From OpenAI / ChatGPT



CircuitPython Essentials › CircuitPython MP3 Audio

CircuitPython MP3 Audio

MP3 playback is supported on most/many RP2040, SAMD51 and nRF boards. It is not supported on any SAMD21 (too slow) or ESP32 (different core chipset) boards. Check whether your board is supported by going to its download page on circuitpython.org and looking for "audiomp3" in the list of "built in modules available".

CircuitPython Essentials

By Kattni Rembor

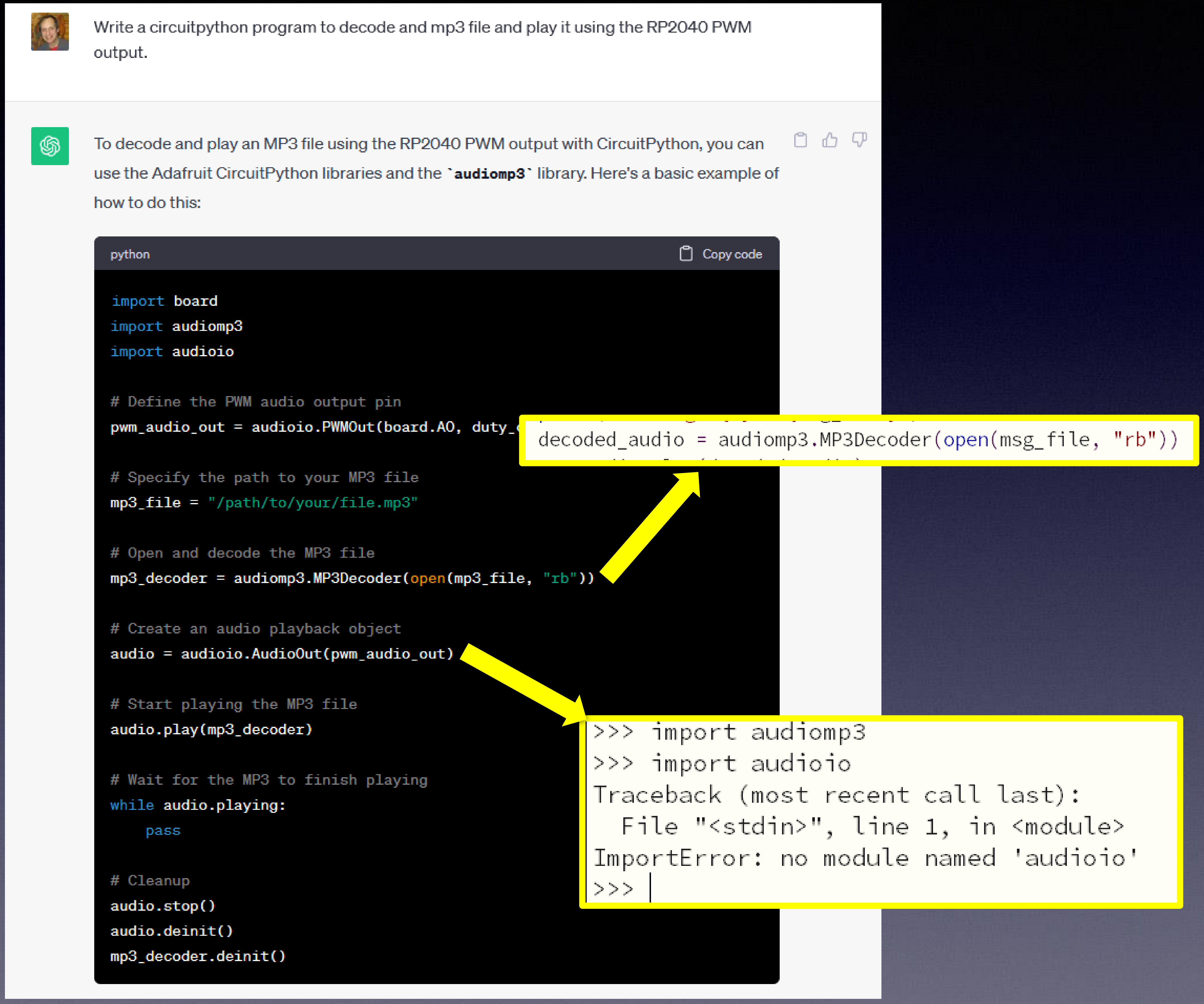
The next step in learning CircuitPython.

Looping MP3 samples with #CircuitPython #ada...

Adafruit page found with Google

Lessons Learned

- Don’t Assume
- Use of AI – Caveat Emptor



Write a circuitpython program to decode and mp3 file and play it using the RP2040 PWM output.

To decode and play an MP3 file using the RP2040 PWM output with CircuitPython, you can use the Adafruit CircuitPython libraries and the `audiomp3` library. Here's a basic example of how to do this:

```python
import board
import audiomp3
import audioio

# Define the PWM audio output pin
pwm_audio_out = audioio.PWMOut(board.A0, duty_

# Specify the path to your MP3 file
mp3_file = "/path/to/your/file.mp3"

# Open and decode the MP3 file
mp3_decoder = audiomp3.MP3Decoder(open(mp3_file, "rb"))

# Create an audio playback object
audio = audioio.AudioOut(pwm_audio_out)

# Start playing the MP3 file
audio.play(mp3_decoder)

# Wait for the MP3 to finish playing
while audio.playing:
    pass

# Cleanup
audio.stop()
audio.deinit()
mp3_decoder.deinit()
```

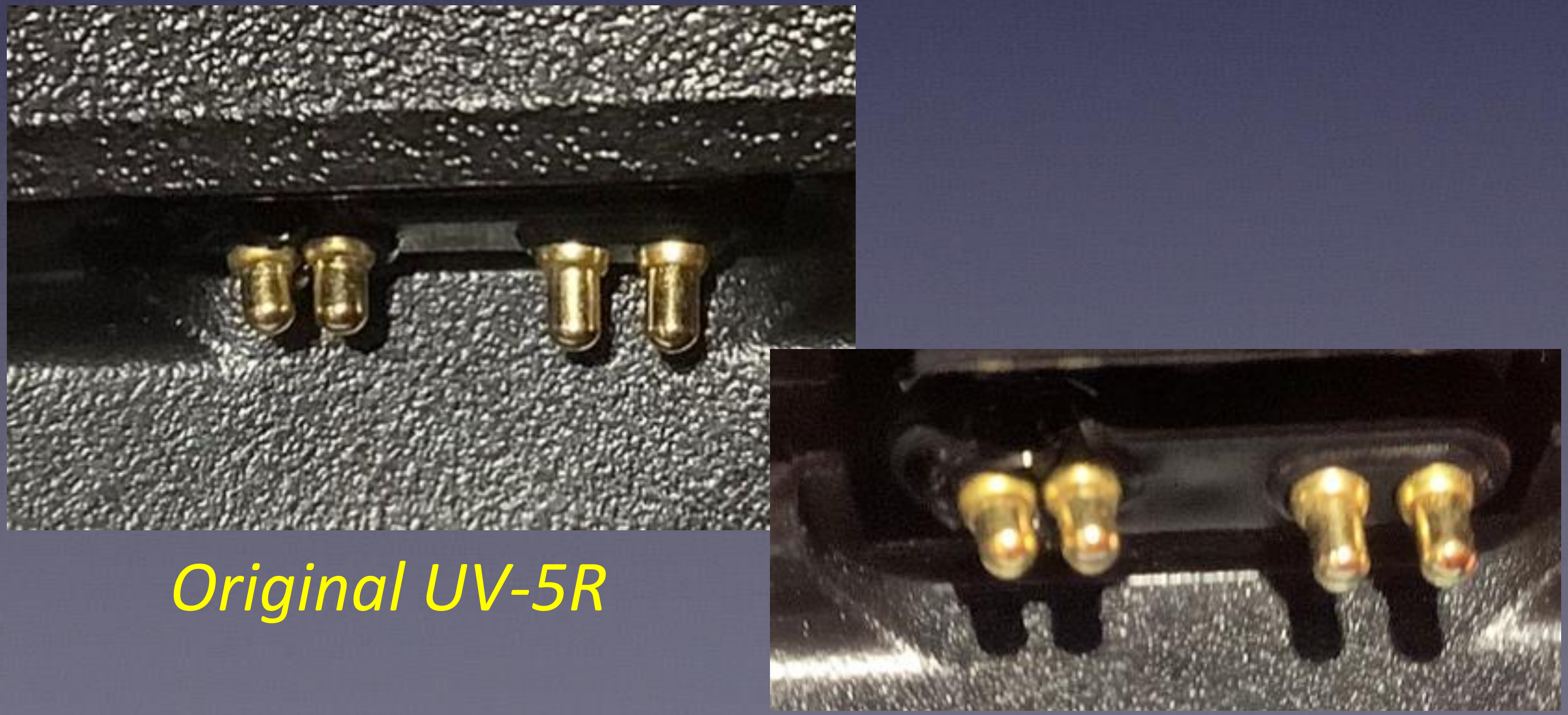
```
decoded_audio = audiomp3.MP3Decoder(open(msg_file, "rb"))
```

```
>>> import audiomp3
>>> import audioio
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: no module named 'audioio'
>>> |
```

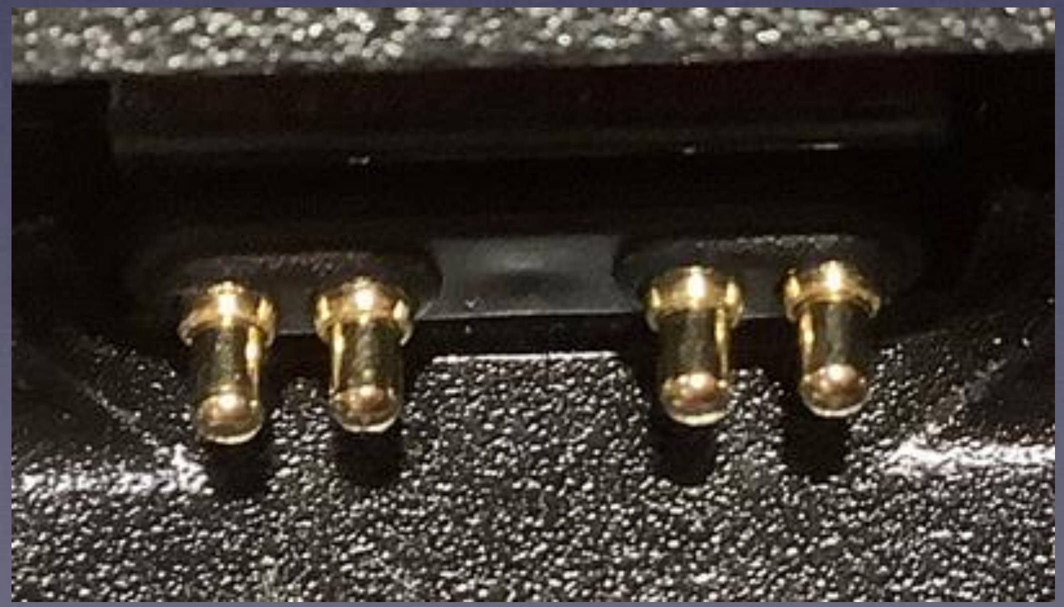
Response to Reworded Question

Lessons Learned

- Don’t Assume
- Use of AI – Caveat Emptor
- It’s always something!



Original UV-5R



New UV-5R

Fox Hunting References

- Northeast Massachusetts Fox Hunters

  https://groups.io/g/NEMassFoxHunters/messages

- Prospect Hill Intermod, Winter 2022

  http://walthamara.org/phi/2022-winter.pdf

- Google

  https://www.google.com/search?q=Hidden+Transmitter+Hunting

Thank You

w1mj.com/fox