# West Lab Hydrophone Data Logger

The West Lab Hydrophone Data Logger uses a Teensy 4.0 and the Teensy Audio shield to periodically record audio data. The Teensy is powered using an Adafruit Powerboost-1A and a 10Ah battery. The 4 dip switches on the data logger allow for 4 gain/level* settings, input selection, and mode selection.

* Gain applies to the amplified microphone input. Level refers to the peak to peak voltage of the line-in signal.
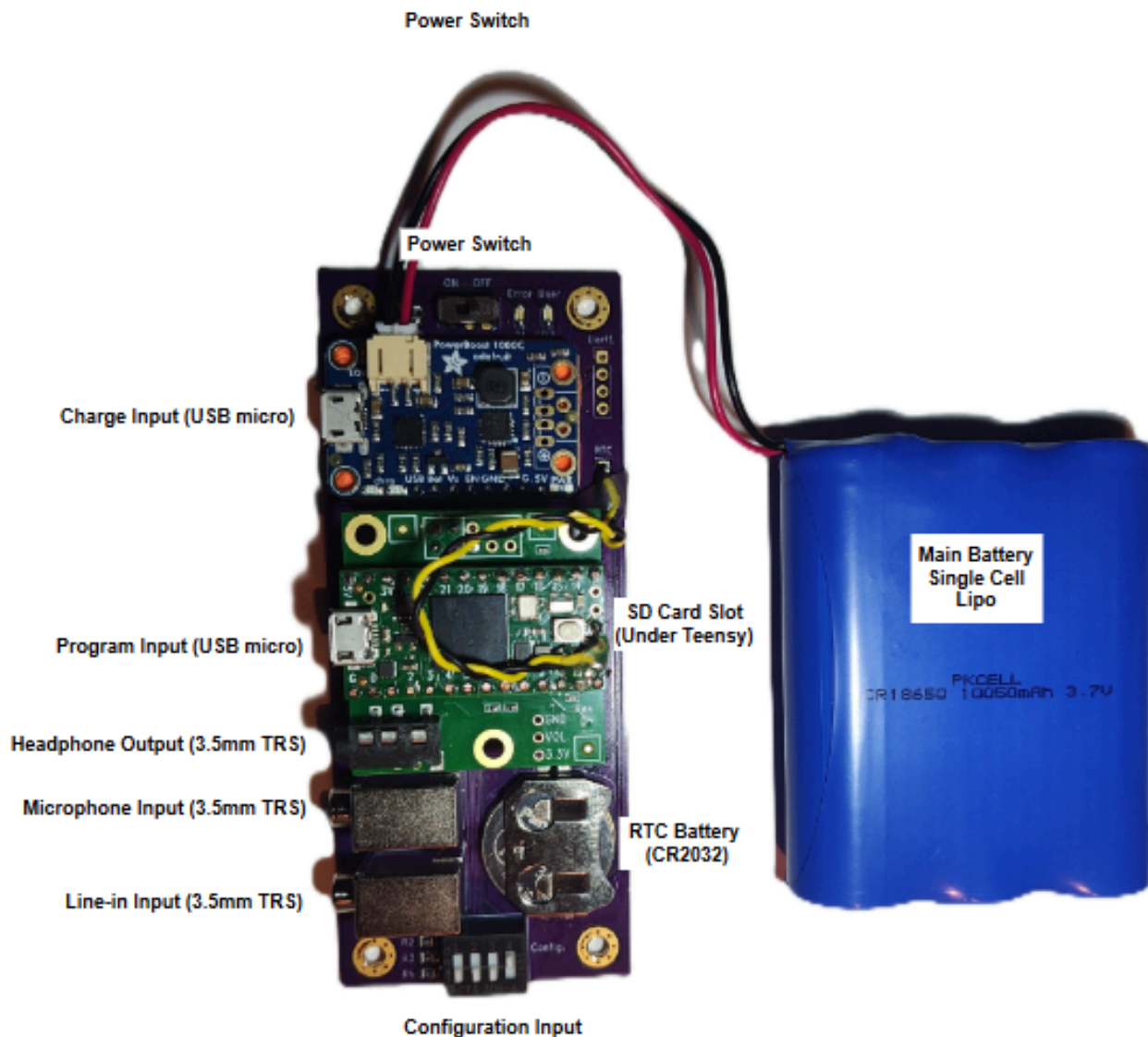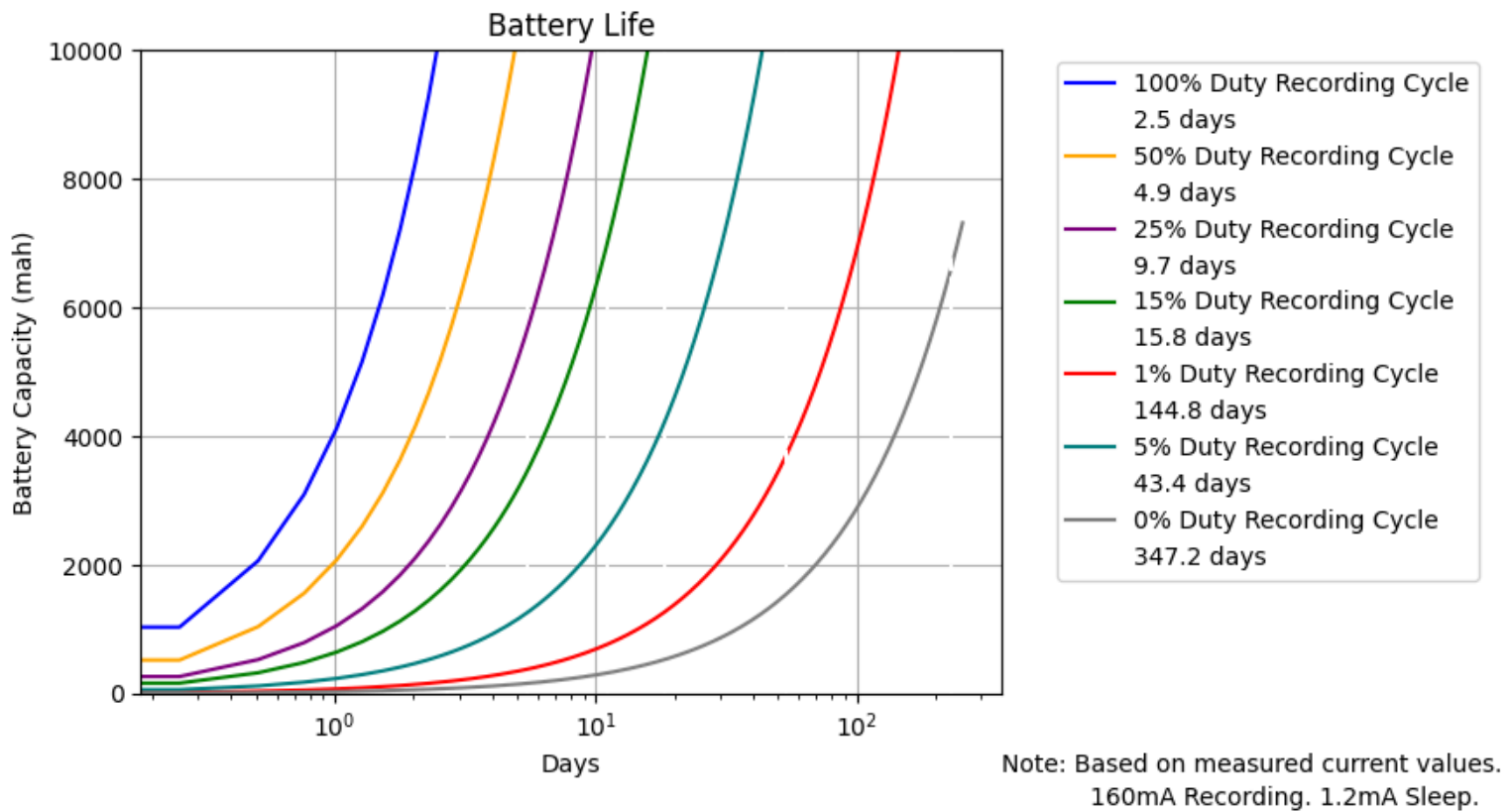

Teensy Recommended SD Card

"Most SD cards are optimized for sequential access, where a camera or camcorder reads or writes a single large file. All SD cards work well for playing a single WAV file at a time.

Cards with "A1" or "A2" rating are likely to have better performance for use with Teensy."
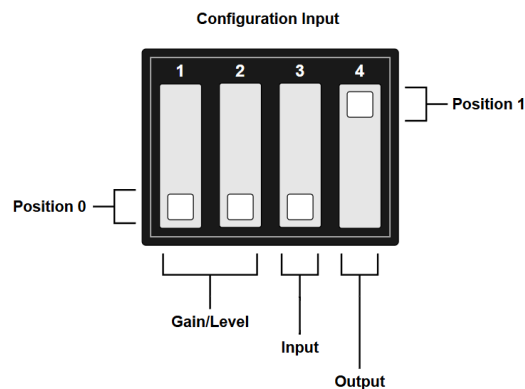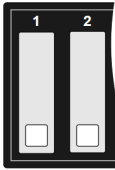
Battery Life

Legend:
- 100% Duty Recording Cycle 2.5 days
- 50% Duty Recording Cycle 4.9 days
- 25% Duty Recording Cycle 9.7 days
- 15% Duty Recording Cycle 15.8 days
- 1% Duty Recording Cycle 144.8 days
- 5% Duty Recording Cycle 43.4 days
- 0% Duty Recording Cycle 347.2 days

Note: Based on measured current values. 160mA Recording. 1.2mA Sleep.
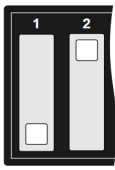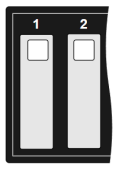
Configuration

Hardware

The device is configured at boot. To configure the device first set the power switch to the off position. Next set the configuration input switches to the desired positions. Last, set the power switch to the on position. The device's user LED will blin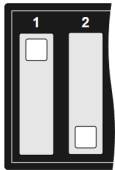k based on the configuration being set. First 1-4 blink pattern is gain/level setting. Followed by a one second delay then the next 1-2 blink pattern in the input selection. Finally the user LED will be off for logging mode or solid on for pass through mode.

Gain/Level			Input			Mode		
Switch Position	User LED	Description	Switch Position	User LED	Description	Switch Position	User LED	Description
	1 blinks	0db / 0.24V p-p		1 blinks	Microphone		none	Logging
	2 blinks	20db / 0.56V p-p						
	3 blinks	30db / 1.33V p-p		2 blinks	Line-in		Solid	Pass Through
	4 blinks	40db / 3.12V p-p						

Software

There are two parameters that control the duty cycle of the data loggers schedule. RECORD_PERIOD and IDLE_PERIOD are defined near the top of the sketch. Both use seconds at their unit. The Arduino IDE is used to recompile when there parameters need changed.

The RTC is also set to the time of the system used to program the Teensy. This time will persist as long as the con cell battery is attached and working.

More information about programming Teensy with the Arduino IDE
(https://www.pjrc.com/teensy/teensyduino.html)

Output Data

The data loggers output if a .RAW file type. The below python script can be used to convert the .RAW files into .wav file types.

```python
# This sketch will take the output files from the West Lab Hydrophone Data Logger and convert each .RAW output
into a .wav file type.
# A path to the folder containing the data files is given(path_data_folder). Each of the output files stored in the folder
at the path will be converted and stored in a new folder.

import wave
import struct
import os

# path_data_folder should be the path folder that contains the data loggers "RECORD.X.X-X-X-X-X-X-X-X.RAW"
files.
# At this path a folder named "Converted Files" will be created and all converted filed will be stored within.
path_data_folder = r'C:\Users\mitch\OneDrive\SchoolFiles\_West\USGS Box\Data_Logger\Testing_Data\Sample
Records'

def convert_raw_to_wav(raw_file, wav_file, channels=1, sample_width=2, sample_rate=44100):
    """Converts a single RAW file to a WAV file."""
    try:
        with open(raw_file, 'rb') as f:
            raw_data = f.read()

        with wave.open(wav_file, 'wb') as wf:
            wf.setnchannels(channels)
            wf.setsampwidth(sample_width)
            wf.setframerate(sample_rate)
            wf.writeframes(raw_data)
        print(f"Converted {raw_file} to {wav_file}")
    except Exception as e:
        print(f"Error converting {raw_file}: {e}")

def convert_all_raw_to_wav(path, file_type='.RAW'):
    """
    Converts all RAW files in the given path to WAV files
    and saves them in a "Converted Files" folder.
    """
    output_folder = os.path.join(path, "Converted Files")  # Create output folder path
    os.makedirs(output_folder, exist_ok=True)  # Create the folder if it doesn't exist

    for root, _, files in os.walk(path):
        for filename in files:
            if filename.endswith(file_type):
                raw_file = os.path.join(root, filename)
                wav_file = os.path.join(output_folder, os.path.splitext(filename)[0] + '.wav')
                convert_raw_to_wav(raw_file, wav_file)

if __name__ == "__main__":
    convert_all_raw_to_wav(path_data_folder)
```

Issues

- When the teensy is in the low power sleep state it can only be brought out of the sleep state in one of two ways. First the normal method of the RTC timer expiring. Second is to short the On/Off pin on the teensy to ground.