Paul Fischer

PHYS 580

Dr. Kwon

12/10/20

<p align="center">Final Project Presentation File</p>

Title and Objective of the project:

Photoresistor Theremin.vi reads the voltage across a photoresistor (the schematic diagram and photographs of the circuit are on the Front Panel) and converts it to an audio frequency played through the connected speakers. The VI begins with the calibration of the instrument for a key inputted by the user (major scale), then gives the user the option to change the volume, add a harmony (3rd above), transpose up an octave, record their playing, and playback any of their saved recordings.

Schematic Diagram(s) showing three components and their connections (circuits, connection to DAQ/Arduino):
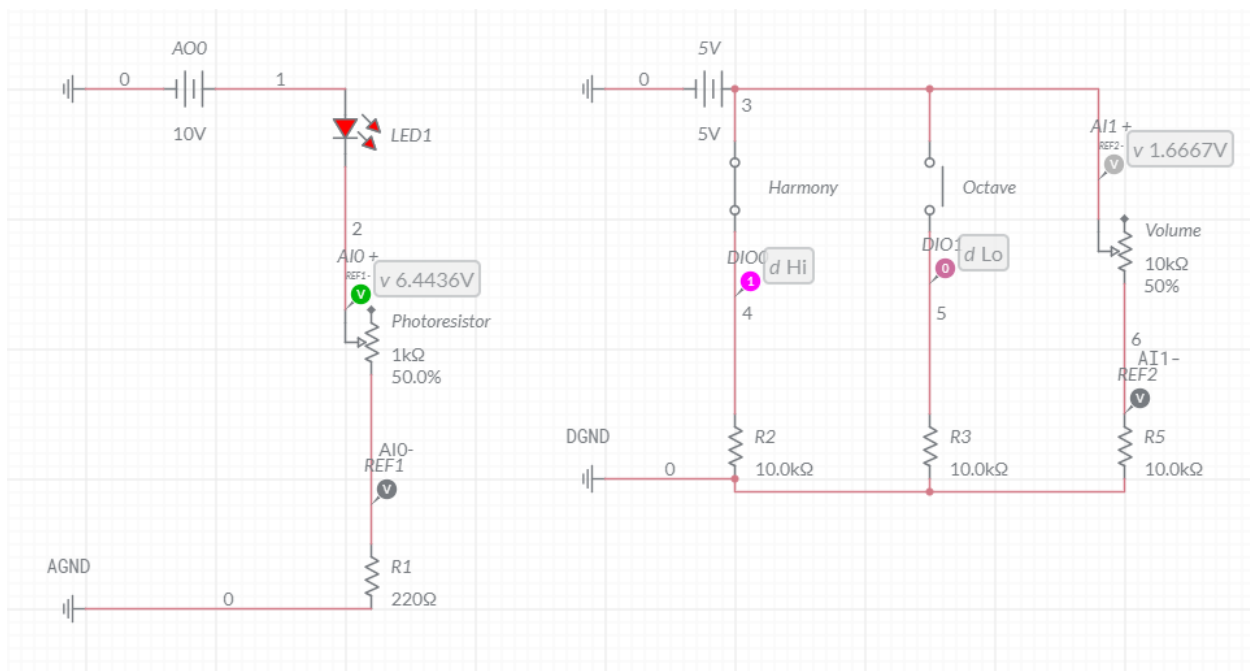
<u>Description of the LabVIEW program:</u>

<u>Functionality in terms of control/communicate, acquire, visualize, analyze, and save data:</u>

Control/communicate:

The VI sends 10 V as analog output out of the myDAQ terminal AO0 to one of the rails of the breadboard. The voltage across the photoresistor is measured by the AI0 analog input terminal and the voltage across the potentiometer is measured by the AI1 analog input terminal. The NI myDAQ also sends a digital output of 5V to the other rail of the breadboard, where the Boolean state of two buttons connected to digital input terminals DIO0 and DIO1 are measured.

Acquire:

The photoresistor voltage (AI0) controls the audio output frequency, the potentiometer voltage (AI1) controls the volume, the left button (DI1) transposes the audio output frequency up an octave, and the right button (DI0) adds as a harmony the corresponding frequency one 3$^{rd}$ above the audio output frequency in the major scale.

Visualize:

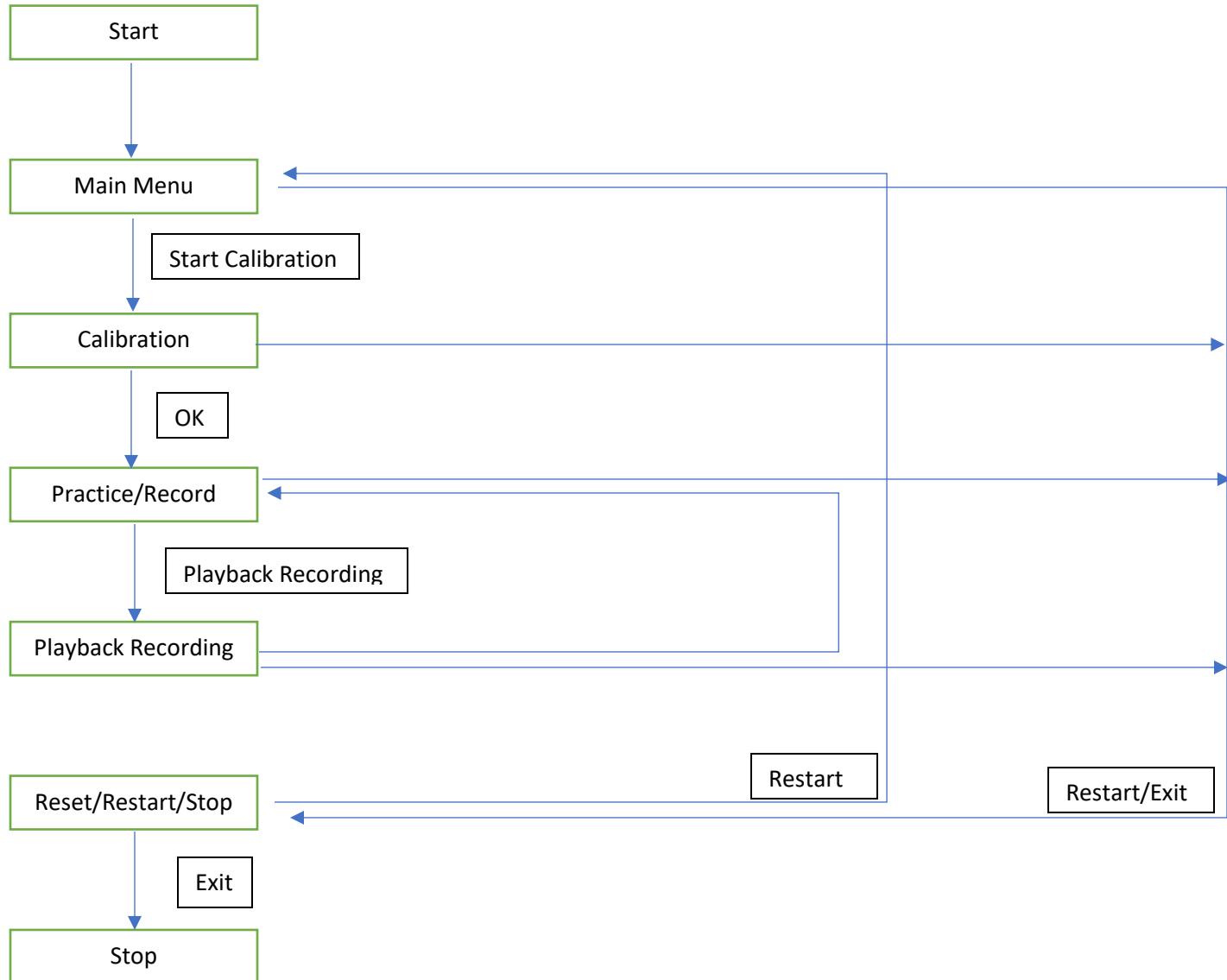The audio frequency outputted to the speakers is displayed on a waveform graph.

Analyze:

The frequency is divided by the sample rate to play the correct frequency through the speakers according to the audio output API. Recordings that are saved can be played back through the speakers.

Save data:

All of the VI's indicators are saved to a .csv file with the path and filename inputted by the user when the user chooses to record data. The data file contains the absolute date and time as well as a header labeling the iteration indicator columns.

Algorithm (state transition diagram):

```
        ┌─────────────────┐
        │      Start      │
        └─────────────────┘
                 │
                 ▼
        ┌─────────────────┐
        │   Main Menu     │◄──────────────────────────────────────┐
        └─────────────────┘                                       │
                 │     ┌──────────────────┐                       │
                 │     │ Start Calibration│                       │
                 │     └──────────────────┘                       │
                 ▼                                                 │
        ┌─────────────────┐                                       │
        │   Calibration   │──────────────────────────────────────►│
        └─────────────────┘                                       │
                 │   ┌──────┐                                      │
                 │   │  OK  │                                      │
                 │   └──────┘                                      │
                 ▼                                                 │
        ┌─────────────────┐                                       │
        │ Practice/Record │◄─────────────────────────────────────►│
        └─────────────────┘                                       │
                 │    ┌──────────────────┐                        │
                 │    │ Playback Recording│                       │
                 │    └──────────────────┘                        │
                 ▼                                                 │
        ┌─────────────────┐                                       │
        │Playback Recording│──────────────────────────────────────►│
        └─────────────────┘                                       │
                        ┌──────────┐         ┌──────────────┐
                        │ Restart  │         │ Restart/Exit │
        ┌─────────────────┐─────────┘         └──────────────┘
        │Reset/Restart/Stop│◄──────────────────────────────────────┘
        └─────────────────┘
                 │   ┌──────┐
                 │   │ Exit │
                 │   └──────┘
                 ▼
        ┌─────────────────┐
        │      Stop       │
        └─────────────────┘
```

Algorithm:

1.  Main Menu

    a.  Prompt user if they want to start calibration

    b.  If yes, go to "Calibrate" state, else go to "Reset/Restart/Stop" state

2.  Calibrate

    a.  Prompt user for # of Major Scale

    b.  Use user input to calculate frequencies of all notes in the scale

     c.   Prompt user for when they want to record each of the 6 V_PR readings to calibrate the theremin

     d.   Go to "Practice/Record" State

3.  Practice/Record

     a.   Read V_PR and output the corresponding audio frequency

     b.   If Harmony Button is pressed, add the frequency one third above the current frequency in the inputted scale

     c.   If the Octave Button is pressed, double the frequency

     d.   If the Record Button is pressed, start writing the data to a log file with a user-inputted path and filename until the Stop Recording Button is pressed.

     e.   If the Playback Recoding button is pressed, go to "Playback Recording" state

     f.   If the Restart/Exit Button is pressed, go to the "Reset/Restart/Stop" state

4.  Playback Recording

     a.   Prompt the user for the path and filename of the recording to be played

     b.   Play audio file

     c.   Go to "Practice/Record" state

5.  Reset/Restart/Stop

     a.   Prompt user if they want to restart or stop VI.  If they choose restart, go to "Main Menu" state, else stop the while loop, send an Output Voltage of 0V to the physical circuit, reset variables, and clear tasks

Implementation:  Program Structure/Design Pattern:

Please find a brief description of each state of the VI's state machine below:

Main Menu:

Prompt user if they are ready to calibrate the instrument or exit the VI.

Calibration:

Prompt user for key (major scale), calculate array of frequencies, send 10 V to the circuit,

calibrate instrument, and prompt the user when they are ready to begin using the instrument.

Practice/Record:

Read photoresistor voltage, convert to frequency, read and set audio output Volume, incorporate

harmony or octave in waveform if Harmony or Octave are TRUE, display waveform on graph,

output audio frequency to speakers, record if Start/Stop Recording is TRUE, go to Playback

Recording if Playback Recording is TRUE.

Playback Recording:

Prompt user which file they want to play back, read specified columns from formatted .csv file,

and output filename, audio, and waveform graph.

Reset/Restart/Stop:

Prompt user if they want to restart or exit and reset controls and indicators to default values.

<u>List of subVIs created (Name and Functionality):</u>

- Calculate Major Scale Frequency Array.vi takes in the number of the major scale (0->C,

  etc.) the user wants to play in and outputs an array of the frequencies in that scale relative

  to middle X.

- Calibrate Photoresistor Theremin.vi prompts the user for when they would like to record

  7 different photoresistor voltages and outputs the logged voltages as an array.

- Format Data.vi takes the current iteration's Indicators Cluster as input and outputs a string

  that is formatted for the log file contained in the Record (save to Log Data File) subVI.

- Generate Waveform.vi takes in the Frequency/Sample Rate quotient that depends on the voltage read across the photoresistor and turns it into a waveform that can be played audially through speakers as well as a waveform that can be plotted on a waveform graph. It also adds a harmony and/or octave transposition to the waveform depending on whether either control is TRUE.

- Read Controls.vi reads the voltage across the potentiometer and the state of the two digital buttons to determine the Volume double-precision floating-point value and the digital Harmony and Octave Boolean values.

- Read Voltage and Output Frequency.vi reads the voltage across the photoresistor and converts it into a frequency from the Scale Frequency Array that corresponds to which two voltages contained in the V_PR Array it falls between.

- Record (Save to Log Data File).vi prompts the user for a file path, name, and type when recording starts, adds a header to the file, and logs the formatted data string to the file each iteration until Start/Stop Recording is FALSE.

Saved Data file (a view including the header):



Self Evaluation/Reflection/Possible Improvement:

I am thrilled that I was able to build a VI with full functionality of all of the features I put in my original development plan. However, there are MANY features I would love to add to this VI. I will list just a few of them below:

- Play in any mode (instead of just the major scale)

- Transpose to any octave

- Add more complicated harmonies than just the third above

- Replicate the sound of other instruments through the ratios of their harmonics

- Change key/timing of playback files (slow down/speed up)

- Metronome

- Play over playback audio

- Import songs to play over

- Export recordings as audio file

- Use other scales – i.e. microtonal scales

- Traditional theremin mode – no cutoff for notes in a scale