FUNCTIONAL OVERVIEW

My project will be a device that allows users to create music. The device can be used by musicians and music novices alike. It will essentially allow users to select a sequence of notes, and it will then play them back in a loop. It will also have a Theremin-style tone generator, which will function separately from the sequencer. The device will be able to run on its own, hopefully with its own power source, and in a professional-looking enclosure.

DESIGN OVERVIEW

The device will have two modes: Sequencer and Theremin. A switch selects between the two.

In the sequencer mode, the user will use a potentiometer to select a frequency, and will then push a button to add a note of that frequency to the playback queue. The device continually cycles through the playback queue, playing all the notes, leaving a constant interval between each note. By changing the position of the potentiometer, and pushing the button at various different times in the device’s playback queue, the user can effectively create a sequence of different notes which will be played over and over again. Adding a note to a “beat” in the playback queue that already has a note will simply replace the old note. The frequencies available for selection using the potentiometer won’t just be every frequency within the defined range but will actually be rounded to certain frequencies of the same scale, to ensure that the sounds produced by the machine match up. A separate potentiometer will be used to select between different scales/keys. A flashing red LED will indicate each beat in the playback queue, whether or not the user has added a note to that beat. A flashing blue LED will indicate when the playback queue has reached the end of a cycle and is starting over from the beginning. Notes can be deleted from the playback queue by turning the potentiometer all the way down and then pushing the button, which places an “empty” note, or by pushing a separate button, which will clear the entire playback queue. The speed of
playback can be adjusted by using a second potentiometer. Additional beats can be added by pushing another button (not pictured.)

In Theremin mode, the user will start by calibrating the Theremin, and will then wave their hands above the device to fluctuate the amount of light being received by the device’s photovoltaic resistor. To calibrate, the user will push the calibration button, at which point the device will start rapidly recording the amount of light it receives. During this time, approximately 5 seconds, the green power/status LED on the device will flash to indicate to the user that calibration is taking place. During calibration the user should alternate between completely covering the device and completely uncovering the device, in order to expose it to the entire range of light values, including the maximum and minimum possible light readings. At the end of the calibration period, the device will map the maximum and minimum light readings to a preset range of frequencies. Depending on how much light the device gets during normal operation, it will play a corresponding mapped frequency, allowing the user to “play music” by changing the light readings. A flashing red LED will indicate the current frequency being played. Quicker flashing = more light input = higher frequency. Slower flashing = less light input = lower frequency.

![FIGURE 2: SIDE PANEL](image)

The device will consist of the following components:

- A project box, a plastic casing that will contain all the electronics and provide the user with a clutter-free surface/interface. This will also improve the portability of the device.
- Arduino board
- Circuit board with components soldered on. All initial testing will be done on breadboards and components will be permanently soldered on later.
- Battery clip. Will give the device an external source of power. This gives it portability.

- DC power cord

- 1/8” Stereo phone jack. Will allow the user to plug in external speakers or headphones to listen to the sounds produced.

- 1/8” Stereo phone jack. Will allow the user to preview the notes they have selected and are about to place.

- Pushbutton on/off to turn on and off the device.

- Green LED to indicate whether the device is on or off. Will also flicker every time a button is pushed, for “feedback”

- Momentary pushbutton to place a note.

- Potentiometer to select frequency of note.

- Potentiometer to adjust speed of playback.

- Momentary pushbutton to clear all the stored notes

- Momentary pushbutton to add more beats to the sequence.

- Potentiometer to cycle through scales/keys

- Red LED to indicate each beat

- Blue LED to indicate when the sequence is repeating

- Dual-position switch to select between sequencer and Theremin modes

- Photovoltaic resistor to measure light input for the Theremin

- Red LED to indicate the current frequency being played by the Theremin

- Momentary pushbutton to calibrate the Theremin
The components will be soldered/connected as shown here: (resistors and additional buttons not pictured)

**DESIGN DETAILS/CODING**

The code for everything should be straightforward. Most of the required code will be basic-level functions and syntax. Here are two examples of things I will use:

Sequencer: The program will essentially loop through an array of frequencies, which will make up the playback queue. When the user pushes the “Add note” button, the currently selected frequency will be added to the playback array at the index currently being accessed.

Arduino array example: ```int myPins[] = {2, 4, 8, 3, 6};```  
Individual records are accessed by code via: ```int myPins[6] = 4;```  

Theremin: Light input from the photovoltaic cell will be mapped to a range of frequencies.

Map function example: ```int thisPitch = map(sensorReading, 250, 900, 100, 1000);``` Maps the sensor readings between 250 and 900 to a frequency scale of 100 to 1000.
**TESTING**

Before soldering anything onto the printed circuit board or encasing anything in the project box, I will rigorously test the device on a bread board, which will allow me to make changes easily when they will be needed. I will also hand the device to random people to see if they can figure out how it works. This will give me insight onto the intuitiveness of the interface. I want people to be able to use the device with minimal instruction. In addition to these random interface tests (which will happen at the end of the project, when I am creating labels for the device,) I will also conduct focused testing, where I ask users to carry out specific tasks, and look for bugs.

**PROPOSED IMPLEMENTATION SCHEDULE**

December 16<sup>th</sup>: Design review

December 17<sup>th</sup>-25<sup>th</sup>: Breadboard assembling/coding

December 25<sup>th</sup>-31<sup>st</sup>: Breadboard testing/functional testing

January 1<sup>st</sup>-6<sup>th</sup>: Soldering/Mounting in project box

January 7<sup>th</sup>: Code Review

January 7<sup>th</sup>-14<sup>th</sup>: Presentation/interface testing/interface tweaking

January 14<sup>th</sup>: Project due

**POTENTIAL SHOWSTOPPERS**

- Components won’t fit in/on the project box: Not a huge issue, I can simply buy a larger box.
- No more showstoppers foreseen: Everything in this project should be within the scope of the Arduino