Functional Specification

Year: 2017 Semester: Spring Team: 12 Project: Guitutar
Creation Date: January 18, 2017 Last Modified: May 2, 2017
Authors:
Member 1: Austin Peterson Email: peter174@purdue.edu
Member 2: Brian Rieder Email: brieder@purdue.edu
Member 3: Cole Giannotti Email: agiannot@purdue.edu
Member 4: Jen Isaza Email: jisaza@purdue.edu

Assignment Evaluation:

Item	Score (0-5)	Weight	Points	Notes
Assignment-Specific Items				
Functional Description	5	x3	15	
Theory of Operation	5	x3	15	
Expected Usage Case	5	x3	15	
Design Constraints	4.5	x3	13.5	
Writing-Specific Items				
Spelling and Grammar	4.5	x2	9	Some spelling mistakes
Formatting and Citations	5	x1	5	
Figures and Graphs	5	x2	10	
Technical Writing Style	5	x3	15	
Total Score	97.5/100			

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

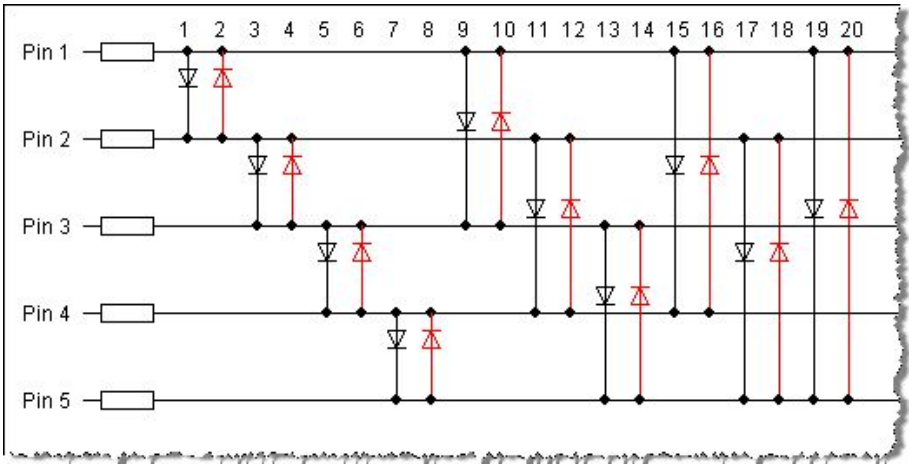
This is an excellent analysis of your project! I had a few cautions about individual constraints, but overall great work.

1.0 Functional Description

The function of Guitutar is to guide the user through the song of their choice with the help of light-up fingerings and chords. The user interface will allow the user to select a song and choose either a learning mode or a "play along" mode. With the help of the learning mode, the LEDs on the fret board will light up along with the chords of the song and will not move on until the user correctly strums the first chord. The "play along" mode will play through the song at its actual speed while LEDs light-up the chords without stopping if the user plays the wrong note.

2.0 Theory of Operation

The primary form of teaching users of the Guitutar how to play the guitar is by lighting up where their hands should be through the use of LEDs. Given that there are 22 frets on a 6-string electric guitar, there needs to be 132 LEDs on the guitar neck. The typical method of powering LEDs on a microcontroller connects the anode to an output pin and the cathode to ground. This allows only one LED to be controlled per pin. In order to remove the requirement to have 132 output pins on a microcontroller, charlieplexing will be used to allow for multiple LEDs to be controlled by fewer pins. The concept behind charlieplexing comes from the fact that microcontroller pins can output high and low voltages. By taking two output pins of a microcontroller, two LEDs can be attached uniquely by having the direction of anode to cathode opposite for both LEDs. By setting pin 1 to high and pin 2 to low, one LED will light up. Reversely if pin 1 is set to low and pin 2 is set to high, then the other LED will light up. When this is applied to sets of pins larger than 2, connections can be made from every unique pair of pins given. This quickly grows in scale to where a large number of LEDs can be controlled over a small number of pins. This can be shown better visually with an example here [5]:



Charlieplexing Example

The figure shown has 5 pins that are able to control 20 LEDs. The exact pin to LED equation can be shown as the following [2]:

$$\boldsymbol{P = ceil[\frac{1 + \sqrt{1 + 4 * L}}{2}]}$$

Where P is the number of pins required to control L number of LEDs. When applied to LEDs on a guitar, 132 LEDs can be controlled with only 12 pins.

3.0 Expected Usage Case

The environmental conditions will be continuous with the normal environmental conditions that an electric guitar can sustain. Guitutar can be used indoors or outdoors in fair weather without rain. Guitutar will be attached to the guitar's neck and will rely upon being as portable as the electric guitar itself. The nature of our users would focus on beginner-level guitar players, but could range up to expert guitar players as well. Guitutar will not add any additional user restrictions compared to a typical electric guitar. Anyone who is able to meet the physical/mental requirements in order to play an electric guitar is expected to be able to use a Guitutar embedded guitar.

4.0 Design Constraints

Placement of the rechargeable battery charger through USB must be convenient for the user so that the guitar can still play without the charger getting in the way. One consideration is to place the USB outlet near the amplifier outlet on the guitar's front face. Another issue is by using charlieplexing for the LEDs, it might be catastrophic if one LED goes out. The reason is that when one LED goes out, then multiple other LEDs may all come on together. It would then be very difficult to troubleshoot which LED is open, short, or leaky. This also presents the difficulty of reaching the electronics after the final product, since we do not want to re-glue the fretboard to the guitar every time we need to look inside the guitar neck. This is also a competitive constraint, as competing models seem to have either a method of removing the LEDs and PCB or no methods at all.

4.1 Computational Constraints

In order to detect if the user is properly playing a note, being able to detect pitch through digital signal processing is important. There are many algorithms for pitch recognition: Autocorrelation, Zero-Cross Detection, and Fast Fourier Transform (FFT). All digital signal processing algorithms would require a minimum frequency of sampling the real signal in order to discretize the signal to make it compatible with any and all computer calculations. The minimum sampling frequency must be twice the frequencies of the highest pitch note that would be played. Assuming standard tuning, the highest note on a guitar that can be played is roughly 1400 Hz [4]. Therefore, the algorithm must be able to take in readings from the live signal at a minimum of 2800 Hz.

Another constraint for digital signal processing is the requirement that an entire period of a signal needs to be processed as a whole in order to determine the frequency. That means that enough memory to store one period of the signal with largest period is required, which would be the lowest note. Continuing the assumption of standard tuning, the lowest note a guitar can play is 82 Hz [4]. Given that the signal must be sampled at a minimum of 2800 Hz, it would require 35 samples to complete a single period of an 82 Hz signal. Using floats to store the data at 4 bytes/float, each signal processing algorithm that we want to run/store simultaneously will require 140 bytes of memory. It would be beneficial to be able to have signal processing on each string individually, which having 6 strings bumps up the memory requirement to 840 bytes.

4.2 Electronics Constraints

A major component of Guitutar is going to be turning the guitar neck, strings, and frets into a switch matrix. The plan is to power the strings and turn the strings and frets into a switch when the two touch. The switch matrix needs to be powered which would need a good battery to power the matrix, LEDs, and microcontroller. For the microcontroller, it has to be able to send Bluetooth signals to an external device, which would be either an app on a phone or a controller. The microcontroller will also get inputs from switch matrix and read the input as a note or chord, and then will get another input from a strum indicating a played note/chord. A STM32 microcontroller (STM32F030C6) seems to be a good choice as it will have plenty of clock speed, flash memory, power consumption, and I/O pins. For the LEDs, we have decided to charlieplex the LEDs to highlight which notes to press on the fret [2]. This maximizes the amount of LEDs that we want (132 individual LEDs) with the least amount of required input pins (12 input pins total). We also would want to use a sensor to determine if the user strum the string. If the strings are powered, we might need to implement a filter to prevent pickup feedback from the other strings. Since we plan to use a battery, the ideal solution to make Guitutar mobile is to use a USB to charge the battery. This would lead us to having the microcontroller also be able to use USB as a peripheral.

4.3 Thermal/Power Constraints

Since Guitutar is intended to be mobile and embedded within the guitar, we would have to power the device with a lithium polymer battery. With this battery, we would also need it to be able to be rechargeable through USB or a similar method. To be able to make the switch matrix work, we would need a constant voltage to the strings and frets. This could result in a lot of power consumption from a battery source. Since the main portion of the board is going to be on the guitar, the device should not exceed the maximum operating temperature of the microcontroller.

4.4 Mechanical Constraints

The switch matrix, LEDs, and PCB(s) are limited to the size of the guitar neck (26” long, 1 ⅝” wide at top, 2 ⅛” wide at bottom, 1” thick at bottom, ¾” thick everywhere else). The weight of the extra components would be negligible since it would not affect the structure of the guitar. To embed the device into the guitar neck, we would route the neck material using a CNC machine. The issue of this is that we would have to be careful of not breaking the neck or cutting the truss rod. We would also have to remove the neck faceplate by heating the neck with a hot iron to melt the glue and separate the two pieces. A total of 132 holes (22 frets, 6 holes per fret) will need to be drilled into the neck for the LEDs. Because of the limited size of the guitar neck, we would need to be careful about drilling holes to avoid splitting the wood of the neck. To protect the LEDs and to protect the device from weathering, we would add a plastic shield to the neck. Since we are powering the frets and strings, we would need to add safety measures to ensure that the user would not be susceptible to electric shock from touching any of the frets or strings. Powering the strings might also create some feedback when an amplifier is connected to the guitar. Using the strings and frets as a switch matrix runs into the issue of a string lower on the

neck touching multiple frets. This means that we would need to filter out extraneous fret touches to ensure that the correct fret is pressed.

4.5 Economic Constraints

There are two routes for building the PCB: building multiple small PCBs for the LED array or using discrete LEDs and guiding the wire down the neck. Using a single PCB is not recommended because the total cost would be too great. Therefore, we decided to modularize the PCBs. However, the price would still be high since we would have to print multiple boards with different sizes and layout. Using discrete LEDs would be more economical than printing specific PCBs, but then we run into the issue of having too many wires running down the neck of the guitar. Any issues with fabricating the guitar neck could also affect the total cost of the project (break strings or neck and have to buy new ones).

4.6 Other Constraints

No other constraints for the project.

5.0 Sources Cited:

[1] (2011, June 17). Guitar Fret Sensing [Online forum]. Available: http://forum.arduino.cc/index.php?topic=64245.0

[2] M. E. Rule. (2013, March 19). Charlieplexing with LED dot matrix modules [Online blog]. Available: http://crawlingrobotfortress.blogspot.com/2013/03/charlieplexing-with-led-dot-matrix.html

[3] C. Bonanno, “Guitar controller for a music synthesizer,” U.S. Patent 4630520 A. Dec, 23, 1986.

[4] ZyTrax. (2016, July 01). Tech Stuff - Frequency Ranges [Online]. Available: http://www.zytrax.com/tech/audio/audio.html

[5] R. Cornelius. (2011). Step 2: Charlieplexing - The theory [Online] Available: http://www.instructables.com/id/Controlling-20-Leds-from-5-Arduino-pins-using-Cha/step2/Charlieplexing-The-theory/