Keyboard DJ Set

Project Proposal

Introduction

Team Members:

- Manas Gandhi (manaspg2)
- Jack Prokop (jprokop2)
- Milind Sagaram (milinds2)

Problem

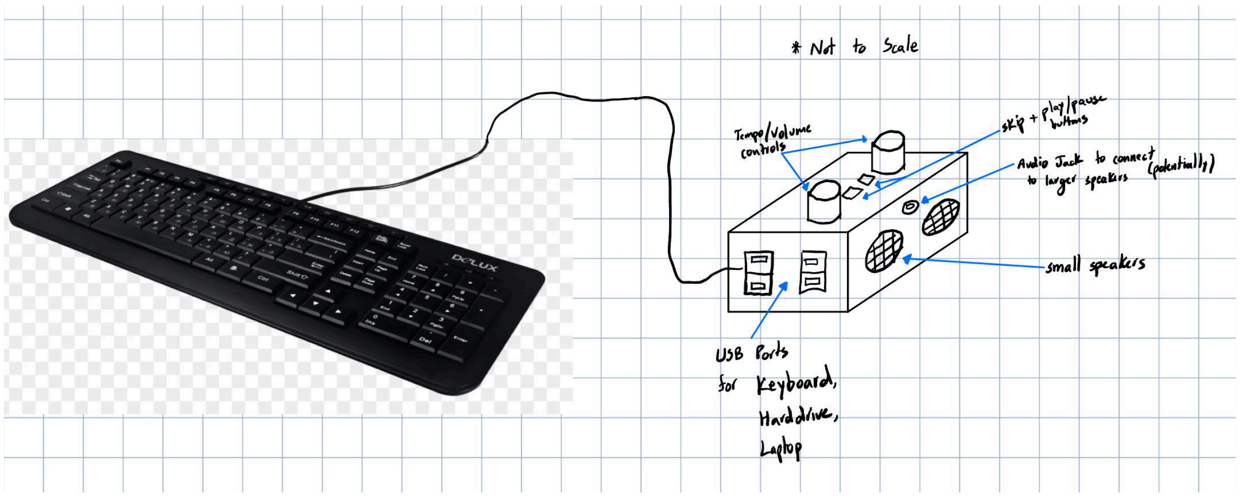
DJ boards have become the "hot topic" of today's music industry, with the tool giving way to many of the greatest artists of our generation, including *John Summit* and *Twinsick*. While we have seen many great EDM artists shine due to the traditional DJ set, it has some drawbacks as well - namely the lack of portability, ease of use for new users, and high prices. First off, DJ boards tend to be large, heavy, and tough to transport. For DJs who want to go practice outside or DJ parties, this setup is suboptimal due the size and weight of the board. DJ boards are also incredibly complicated: there are two wheels, nearly a dozen knobs, a couple sliders, and many buttons on a board. For enthusiasts who want to learn the basics of DJ-ing, this creates a high learning barrier. Lastly, the cost of a DJ board is very high. Boards can be north of $300, which is out of budget for most people who are interested and want to learn the fundamentals of mixing music together. Hence, we propose a portable, easy to use, and cost effective DJ board to help young and interested DJs learn.

Solution

To address these challenges, we propose the DJ keyboard, a DJ board that simply uses the keys on an external keyboard, connected to a computer. This makes use of a microcontroller on a PCB for processing the inputs of the keyboard and converting that into commands for the software. We also will create software for the songs and audio processing, as well as the speaker technology. This approach simplifies the complicated DJ board, reduces the cost and size of getting a DJ board, and makes it easy to take anywhere, making the DJ experience for everyone much easier.

The specific DJ board elements we want to incorporate are volume control, tempo control, music slicing and looping, song skipping functionality, and if we have time, auto-crossfade capabilities

Visual Aid



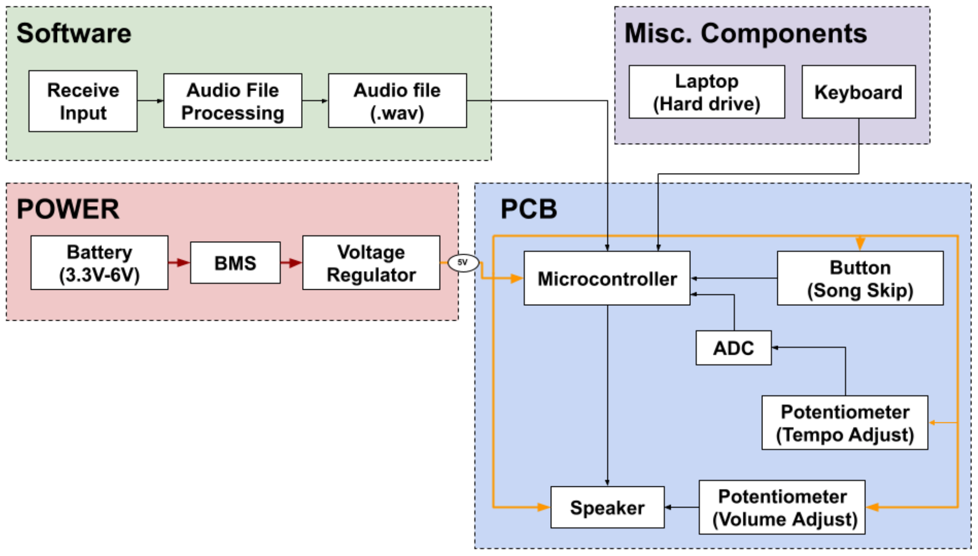
* Not to Scale
Tempo/Volume controls
skip + play/pause buttons
Audio Jack to connect to larger speakers (potentially)
small speakers
USB Ports for Keyboard, Harddrive, Laptop

High Level Requirement List

1. The DJ set can play one song at a time.
2. The DJ set can increase and decrease the tempo by 3 bpm.
3. The DJ set can increase and decrease volume by at least 10 db.

Design

Block Diagram



Subsystem Overview + Requirements

PCB

This subsystem will hold all the electrical components. The parts of this subsystem are shown below.

1. Microcontroller

This subsystem is the core of the whole system, providing the communication between the input (keyboard) and the software. This is where the processing of the keys will happen, as well as sending signals to the hardware.

The microcontroller will be responsible for receiving the keystroke/signals from hardware (in the form of button presses/potentiometer changes) and sending that information to the software. It will also be responsible for receiving audio files from the software and sending that data to the speaker to output the volume.

- **Communication between keyboard/hardware and microcontroller:** This is the first part of the communication that the microcontroller does. It receives inputs from the keyboard and hardware and translates that into information that is readable and usable by software. It will then send this information to the software to use.

- **Communication between software and microcontroller:** this is the second part of the communication that the microcontroller does. After the software does its processing, the software will send the microcontroller the audio files (.wav files) to send to the speaker to play.

3. Volume Control

This subsystem is where we control the volume using the PCB. This will be in the form of a potentiometer, which will send a signal to the microcontroller.

- **Potentiometer:** We will use a potentiometer (which we might connect to a slider) to increase and decrease the volume (increased resistance = lower volume). This will be directly connected in series with the speaker, controlling the volume output of the speaker.
- **Speaker:** We will put two small speakers on the PCB, and connect to output of the potentiometer to the input of the speaker to allow audio control.

4. Tempo Control

This subsystem is where we control the tempo of the song (in BPM) that is currently playing. This will also use a potentiometer, but it will send a signal to the microcontroller through an ADC that will let the microcontroller read voltage levels.

- **Tempo controller:** We will use a potentiometer (which we might connect to a slider) to send a signal to the microcontroller to increase or decrease the tempo. It will be connected to ground and voltage, of course, as well as an ADC that is connected to the microcontroller, so that the microcontroller can get the voltage output of the potentiometer and translate that to a tempo increase or decrease. We will have to set a default value as well (the potentiometer will be set to the middle to begin, and that will be +0 bpm).

5. Skip song

This subsystem is where we skip the song that is currently playing, allowing us to switch to the next song. This will use a button, connected to the microcontroller.

- **Skip button:** We will use a button that sends a high signal to the microcontroller, which will tell the software to send the next song into the speaker, effectively skipping this song and replacing it with the next song.
- **Inputs:** The keyboard should take inputs for all keys on the keyboard, and maybe combinations of keys (if we have time).

Power

This subsystem where the power is supplied to the system, including the microcontroller and speakers.

- **Battery:** We will have a battery on our PCB that will supply the power to the rest of the system, including the microcontroller and all other hardware components that require it.
- **BMS:** The purpose of this component is to monitor the battery to make sure that it is working properly.
- **Voltage Regulator:** This component is used to convert higher voltages to lower voltages, if needed. Our system should mostly require 5V to all hardware components, so we should only need this if we cannot find a battery that is 5V.

Software

This subsystem where the control and processing of the system happens. The inputs should be processed into some functionality based on a DJ board here.

- **Receiving Input:** Inputs will be processed and translated into functions here, from keyboard/hardware to the microcontroller to the software.
- **Audio processing:** audio processing will occur here, where the audio files will be processed and manipulated based on the inputs of the keyboard.
  - Will have to incorporate signal processing libraries.
- **Audio file (.wav):** The .wav file will be the processed audio file that will be sent to the microcontroller.
- **Communication with Keyboard:** embedded software will be written to communicate with the microcontroller through a bidirectional USB using SPI protocol. This will be how the software will receive input data and send audio files to the microcontroller.

Misc. Components

1. Keyboard

This subsystem has the inputs that our system will be receiving. This will be in the form of keystrokes, which the microcontroller will go on to translate into different functions.

2. Laptop

This subsystem will be used for all the components we cannot buy, specifically for a hard drive and a speaker.

- **Hard drive:** hard drive is just for storage purposes, real DJ boards have hard drives that they plug into the boards typically

Tolerance Analysis

The primary concern for the success of this project will be the speed of the software for audio processing. To achieve smooth transitions in tempo, slicing, looping, and crossfades, the audio processing software will need to execute with very low latency. For example, Spotify has a minimum sample rate of 44.1KHz, meaning there are only ~22 microseconds between samples and any audio processing will need to execute ideally between two samples. This means that the microcontroller must have a high clock speed and DSP-specific hardware to execute processing algorithms in real-time allowing for high-quality effects. Microcontrollers do exist with these capabilities, such as the arm-based Cortex-M7, which has a maximum clock speed of 216MHz and supports DSP instructions.

Ethics and Safety

Privacy and Data Security (ACM Code 1.6):

- **Issue:** The software might need to interact with personal files on a user's device, including music libraries. Our design must prevent unauthorized access to the user's music or other personal files.
- **Solution:** We will not collect or store any personal data from users unless explicitly required (such as music) and will follow strict data security measures (ex. consent).

Intellectual Property (ACM Code 1.5):

- **Issue:** Users could potentially use the software to play/mix copyrighted music without authorization, violating intellectual property laws.
- **Solution:** We will include disclaimers encouraging users to comply with copyright regulations and offer placed to legally obtained music files that can be used.

Honesty and Integrity (IEEE Code 7.8.I):

- **Issue:** Ethical guidelines also mandate transparency in communication. We must avoid misleading claims about the functionality or features of the DJ set.
- **Solution:** All documentation, including advertising or promotional materials, will clearly represent the capabilities of the system.