Keyboard DJ Set

ECE 445 Design Document - Fall 2024

Project #45

Manas Gandhi (manaspg2)

Jack Prokop (jprokop2)

Milind Sagaram (milinds2)

TA: Sainath Barbhai

I. Introduction

I.I 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*. However, despite their popularity, traditional DJ boards pose significant barriers to entry for aspiring DJs. Three key issues stand out with these boards: the lack of portability, complexity for beginners, and high costs.

1. **Lack of Portability:** DJ boards are usually bulky, heavy, and cumbersome to transport. In fact, DJs typically need full suitcases to be able to carry them around. This poses difficulties for DJs who wish to practice in outdoor settings or perform at smaller, more intimate events. The weight and size of these boards make them impractical for travel, limiting their use to stationary setups like studios or large venues as well.
2. **Complexity for Beginners:** Standard DJ boards feature an overwhelming array of controls, including turntables and lots of knobs, sliders, and buttons. This complexity creates a steep learning curve for those new to DJing.
3. **High Cost:** Traditional DJ boards are expensive, often priced at $300 or more [1]. For many individuals who are simply exploring the hobby or wanting to learn the basics, this price point is prohibitive. The cost of entry is a significant barrier, especially for younger audiences or those with limited disposable income.

I.II Solution

To overcome these limitations of traditional DJ boards, we want to create a portable, user-friendly, and affordable DJ board. Our solution aims to make DJing accessible to beginners and enthusiasts by addressing the core issues of portability, ease of use, and cost:

1. **Portability**: The DJ board will be lightweight and compact, allowing users to easily transport it to different locations, whether it's a park, a friend's house, or a small event.
2. **Simplified Interface**: Designed with beginners in mind, our DJ board will feature a minimalistic control scheme. Instead of overwhelming users with too many knobs and buttons, it will provide just the essential controls.
3. **Affordability**: By leveraging cost effective hardware and focusing on essential features, the DJ board will be priced significantly lower than traditional models.

I.III Visual Aid

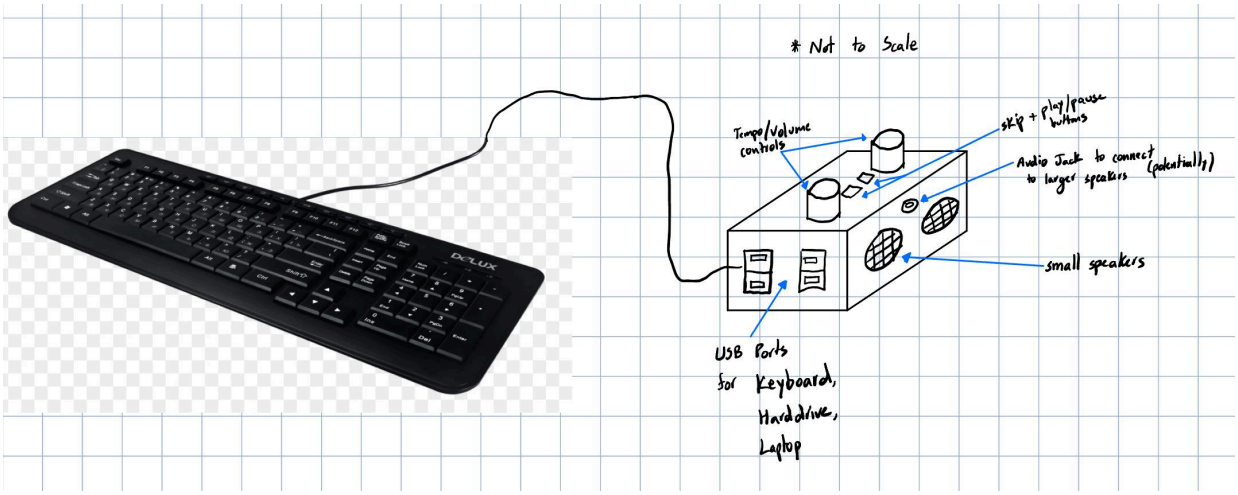

Figure 1: Visual Aid of DJ Set/Box

In this board, we will have a “DJ box” that contains the PCB with all the buttons and knobs. The user will be able to turn the knobs, which are displayed at the top of the DJ box. The speakers will be at the front of the box to output music that is being played, and there are USB ports on the side to connect up the keyboard and other needed components. The buttons will be placed in the middle of the knobs to provide easy access. Overall, it is pretty clear from this design how portable the DJ board will be - it just requires transportation of the DJ Box, a keyboard, and a laptop, all of which can fit in a backpack instead of a large suitcase.

I.IV High-Level Requirements

To consider our project successful, it must meet the following requirements:

1. **Simultaneous Track Playback:** Our DJ board must be capable of playing two tracks simultaneously, allowing users to transition between songs or mix them together. This feature is essential for replicating the core functionality of traditional DJ equipment, enabling smooth transitions and creative layering of audio tracks.
2. **Precision Tempo Control:** The DJ board must support the ability to increase or decrease the tempo of each track by at least 3 beats per minute (BPM). This level of precision will give users the flexibility to fine-tune the speed of the music, allowing for smooth beat-matching.
3. **Dynamic Volume Adjustment:** The DJ board must allow users to adjust the volume of each track by at least 10 decibels (dB) in both directions (increase or decrease). This ensures that users can balance audio levels during live performances or practice sessions.

II. Design

II.I Block Diagram

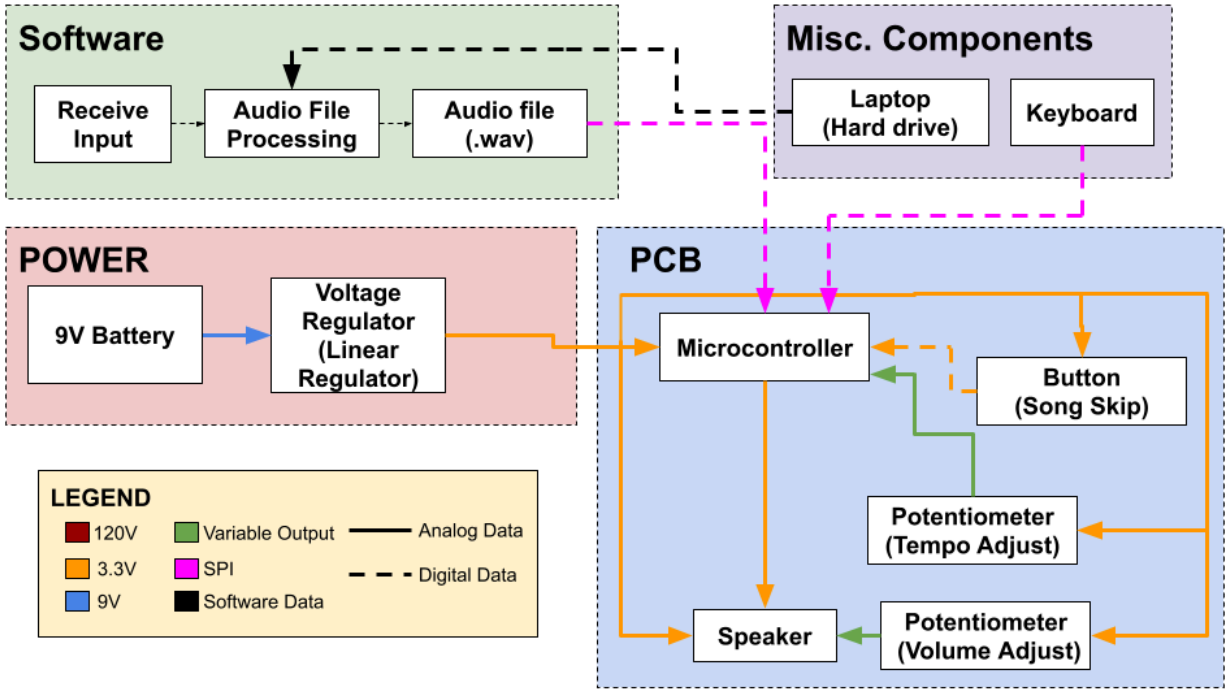

Figure 2: Keyboard DJ Set Block Diagram

II.II Physical Design

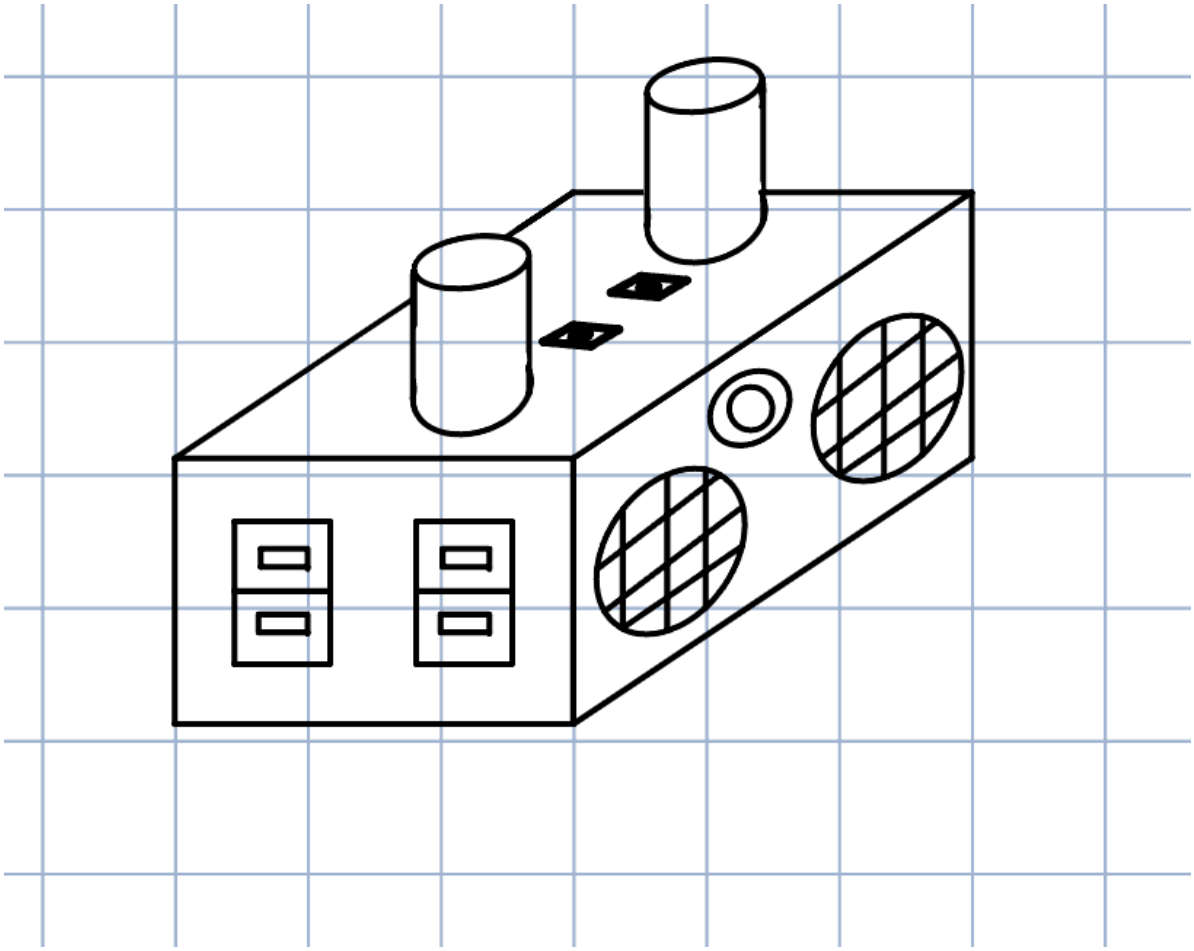

Figure 3: Physical Desing of DJ Set/Box

Above is the physical design for the DJ box that will contain the DJ board. We can see the two knobs for volume and tempo at the top, the buttons for play/pause and skipping a song. The USB connection ports are on the side of the box for the keyboard/laptop connections, and the speakers are in the front to play music. The overall idea of the physical design is to make it look almost like a boombox or stereo, as this kind of "retro" style appeals to today's generation.

II.III Subsystem 1: Microcontroller

The microcontroller will be mounted on a custom-designed printed circuit board (PCB), serving as the "brains" of the system. It plays a central role in managing and controlling all the inputs and outputs necessary for the system to function. On the input side, the microcontroller will receive signals from multiple sources. These include commands from the keyboard when the user presses buttons to select notes or actions, as well as audio signals from the laptop, such as music files or live streams that need processing. These inputs are crucial in determining the sound that will be played through the speaker.

On the output side, the microcontroller processes these inputs and sends the resulting audio signals to the speaker for playback. It ensures the quality and timing of the audio output are precise, maintaining the overall performance of the system. The software running on the microcontroller includes essential algorithms for audio processing, such as converting sound signals into digital formats or generating .wav files. It also manages port I/O, facilitating communication between the keyboard, speaker, and other components. Overall, the microcontroller is vital for running the software, executing audio processing tasks, and delivering the final sound output in a seamless and efficient manner.

Components:

Component	Part Number	Quantity	Unit Cost
Microcontroller	ESP32-S3 module	1	$3.48
Resistor	RMCF0805JG10K0	8	$0.10
Capacitor	CL21B105KBFNNNG	4	$0.10

II.IV Subsystem 2: Buttons

The buttons will be part of a keyboard interface, allowing the user to perform various operations on a song, such as skipping tracks, pausing, or adjusting playback. Each time a button is pressed, it sends an input signal indicating the desired action. These inputs represent the user's commands and are essential for controlling how the audio behaves. The keyboard essentially provides a direct way for the user to interact with the system, ensuring they can easily manage song playback or other operations as needed.

The buttons circuit has a pull up resistor connected to a switch in parallel with a capacitor, both going to ground. The voltage across the switch (same as the voltage across the capacitor) is input into the microcontroller. When the switch is open, the capacitor gets charged up and holds the bus voltage, in our case 3.3 V, corresponding to a digital high (1). When this happens, the current going through both the resistor and capacitor are 0. When the switch is pressed (when the switch is closed), the capacitor discharges, and all of the current that goes through the pull up resistor goes through the 0 resistance switch path. When this happens, the voltage across the switch (and therefore the capacitor) is 0, corresponding to a digital low (0).

Once the input is received from the keyboard, it is transmitted to the microcontroller, which acts as the central processor. The microcontroller runs the software and audio processing algorithms that interpret these inputs and execute the corresponding actions. For example, when a skip button is pressed, the microcontroller processes the command and skips the current song to the next one in the playlist. Other operations, like rewinding, pausing, or adjusting volume, are similarly handled. The microcontroller ensures that all inputs are correctly processed, and the system responds promptly to the user's commands.

Components:

Components	Part Number	Quantity	Unit Cost
Keyboard	Rii RK907	1	$9.99

II.V Subsystem 3: Volume and Tempo Control

To control the tempo and volume of the audio, we will incorporate potentiometers into the design. These are adjustable dials that allow the user to fine-tune these aspects based on their preferences. The position of the dial will determine the level of volume or the speed of the tempo, giving the user direct control over these parameters in real-time.

The output from each potentiometer is sent as a signal to the microcontroller, which interprets the input and adjusts the audio accordingly. Whether the user is increasing the volume or slowing down the tempo, the microcontroller processes this information and modifies the audio file before transmitting it to the speaker. This ensures that the system responds accurately to the user's adjustments, providing a seamless listening experience.

Components:

Component	Part Number	Quantity	Unit Cost
Potentiometer	P160KN2-0QA25B10K	2	$1.83

II.VI Subsystem 4: Power

The power subsystem is composed of two key components: a battery and a voltage regulator. Together, these elements ensure that each part of the system receives the appropriate power it

needs for smooth operation. The battery serves as the main power source, and will be 9V. It will handle the charging and discharging to maintain reliable power output.

The voltage regulator plays a crucial role in distributing the correct voltage to different subsystems. For instance, the microcontroller requires a 3.3V power rail to function properly, so the power subsystem must supply exactly 3.3V to it. The rest of our components only require 3.3V, but if other components in the system require different voltage levels, such as 5V, the power subsystem must provide the correct voltage for those components as well. This ensures that each subsystem operates efficiently with the power it requires.

Components:

Components	Part Number	Quantity	Unit Cost
9V Battery	3046-9V-ND	1	$4.45
Voltage Regulator	AZ1117CD-3.3TRG1	1	$0.44

II.VII Subsystem 5: Software

The software subsystem is divided into two essential components: drivers and audio processing algorithms. The drivers are responsible for controlling the input and output ports of the microcontroller. They ensure that the microcontroller's pins are correctly configured to receive input signals, such as those from the keyboard or potentiometers, and to send output signals to devices like the speaker. This allows for smooth communication between the microcontroller and the hardware components, ensuring the system operates as intended.

The audio processing algorithms handle various functions related to manipulating the audio files (.wav files). These algorithms are designed to perform tasks such as speeding up the song,

skipping tracks, adjusting the volume, and executing other audio-related operations. By integrating these algorithms, the software subsystem ensures that the microcontroller can process user inputs and modify the audio playback accordingly, delivering a seamless and responsive audio experience.

II.VIII Subsystem 6: Laptop

The laptop subsystem is solely responsible for hosting the music files, and act as a hard drive. This will store all the songs the user has to select from, and the user will be able to select the song that they want to be played from the DJ board. The laptop (hard drive) will send the music files to the software, which will handle the rest of audio editing.

III. Requirements and Verification

II.II Requirements & Verification

II.II.I Microcontroller

Requirements	Verification
I. Must be able to handle Keyboard, Potentiometer, and Button Inputs II. Must send information from PCB components to software	I. The microcontroller will receive the correct voltage from the components, which we can measure within the microcontroller software. In order to verify these voltage measurements, we have added specific reference points that route to ground and different signals. When turning the potentiometer dial, for example, we will place one probe of the multimeter on the ground test reference and the other on the reference point representing the voltage across the potentiometer. As the knob is turned, we should see this voltage change. For the buttons, we will place a multimeter probe on the ground reference point

	and the other on the reference point corresponding to the voltage across the button. We will ensure that the voltages when the button is depressed is the opposite from when it is not. For the keyboard, there should be no signal through the USB port when no buttons are being pressed. When they are pressed, however, there should be a signal corresponding to the button pressed, so we will measure this voltage to see that there are changes using the ground reference and data reference points on the board. II. Software will go to the appropriate state when respective components are activated. To do this, we will have three output test points to verify that components are working: one for the keyboard, one for the potentiometer, and one for the button. When a keyboard button is depressed the software will indicate that by setting

	the voltage of the keyboard microcontroller output test point high. For the button, it will do the same with the button microcontroller output test point. Finally, for the potentiometer, when the dial is moved, the voltage at the potentiometer microcontroller output test point will be set high. By using this, we will be able to tell that the software is able to pick up on physical tampering with the components.

II.II.III Volume and Tempo Control

Requirements	Verification
I. When the potentiometers are adjusted, the DJ board must respond appropriately, by adjusting the volume and tempo accordingly	I. The system must increase and decrease the volume by 10 decibels (dB), measured by a decibel meter II. The system must increase and decrease the tempo by 3 beats per minute (bpm), measured by a

	metronome

II.II.IV Power

Requirements	Verification
<u>Battery:</u> I. Supply +9V ± 5% Voltage with 500mA current draw II. Must last for up to 2 hours <u>Linear Regulator:</u> I. Voltage must be regulated to +3.3 V ± 5% at 500mA draw	<u>Battery:</u> I. Use an oscilloscope and multimeter to verify a consistent voltage at the specified level II. We will operate the DJ Box for 2 hours and measure the voltage provided with a multimeter and oscilloscope during operating time <u>Linear Regulator:</u> I. Voltage must be regulated to +3.3 V ± 5% at 500mA draw

II.II.V Software

Requirements	Verification
I. Must be able to adjust the tempo of the song with a latency of less than .5 seconds	I. We will timestamp the input tempo change and execution of the tempo-adjusting software and verify the latency over a serial terminal on a laptop connected to the microcontroller. II. Alternatively, we can use a metronome to detect the live tempo of a song and measure the delay of the tempo change.

II.II Tolerance Analysis

II.II.I Microcontroller (ESP32-S3 Module)

The ESP32-S3 operates at 3.3V with a maximum current of around 500 mA. To ensure proper functionality, we need to make sure the power supply is constantly delivering 3.3V ± 5%. As such, as the power supply's voltage tolerance is ±5% of 3.3V = 3.135V to 3.465V. This tolerance is okay because the microcontroller does not need a specific to power itself, but rather just needs a high enough voltage to supply it power consistently so that it works properly.

II.II.II Resistors (RMCF0805JG10K0)

These resistors will be used for pull-up/down functions and setting reference voltages.
A 10kΩ resistor with a tolerance of ±5% means the actual resistance could vary between 9.5kΩ and 10.5kΩ. This will be fine for the design, as the functionality of pull-up/down of the resistor will not be affected, as the high voltage and low voltage will still have a clear difference.

The resistors can tolerate this change in resistance because they are primarily used as pull up resistors for the potentiometer and for the capacitor. For the pull-up resistors, those are simply to provide the circuit a basis for what a high voltage looks like. So a tolerance of just 5% is okay, because we still have a basis for high voltage and low voltage. Additionally, for the capacitors, the resistors are used for current control. This ensures that there is not a current overload in the capacitor so that it can charge appropriately. As such, we can tolerate a 5% change in value, as we are still preventing current overload. Additionally, a 5% tolerance is industry standard[5].

II.II.III Capacitors (CL21B105KBFNNNG)

The capacitors will be used for the switch circuits. With a nominal value of 1μF and ±10% tolerance, the capacitance will range between 0.90μF and 1.10μF. This tolerance for the capacitor is okay because over infinite time in the buttons circuit (explained above), it will still reach the high voltage (3.3 V), as that is a property of capacitance. As such, if the capacitor is fully charged, it will be high, and when it is discharged, it will have a value of 0 V, and its value will be low. So a 10% tolerance, which is fairly common for capacitors in the industry, is okay for our system.

II.II.V Potentiometers (P160KN2-0QA25B10K)

With a 10kΩ potentiometer, the tolerance will be ±5%, meaning the range is from 9.5kΩ to 10.5kΩ. This range will not critically affect the system, as the volume and tempo will only change slightly, to an unnoticeable amount. This tolerance is okay because the potentiometer output is used for tempo and volume control. The way that the tempo/volume control works is based on the change in voltage that is inputted into the microcontroller, where it takes the difference between the current voltage and the new voltage. So as long as we monitor how much the tempo and volume are changing, there is no issue.

II.II.VI Power System (5 V Battery and Voltage Regulator AZ1117CD-3.3TRG1)

The 5 V battery supplies power and the voltage regulator steps it down to 3.3V for the microcontroller and other components. These components will require the most tolerance, as they tend to vary in output. A typical 5 V battery will range from 5.2 V to around 4.8 V when depleted. The voltage regulator should be able to provide a steady 3.3 V output across this input range. The AZ1117 voltage regulator has a dropout voltage of about 1.15V [2], so it requires at least 4.45 V input to maintain 3.3V output, which is lower than the 4.8 to 5.2 V range.

II.II.VIII Software Tolerance Analysis

Another perspective to examine, tolerance wise, is the software, and the tolerance we need to be able to accomplish our goals with the software. Our software will adjust the tempo of a .wav file sampled at 44.1 kHz on an ESP32-S3 microcontroller. This must be completed within 0.5 seconds from the time the tempo adjustment request is initiated. The software will be run on an

ESP32-S3 microcontroller, which has a 240 MHz clock speed, 384 KB ROM, and 512 KB SRAM. To quantify the analysis, we need to first define the parameters associated with the microcontroller and other data that the software will interact with.

Key Parameters for ESP32-S3

I. **Clock Speed:** 240 MHz (0.24 GHz)

II. **Memory Constraints:** 512 KB SRAM and 384 KB ROM

III. **WAV File Parameters:**

III.I **Sampling Rate**: 44.1 kHz (44,100 samples per second).

III.II **Bit Depth:** 16 bits (2 bytes) per sample.

IV. **Tempo Adjustment Requirement**: The system must be able to change the tempo of the audio and process it in real-time with a response time of less than 0.5 seconds.

We need to first consider the processing and memory part of the software. In terms of the processing, we can measure this by the complexity of the algorithm. The complexity of an FFT-based phase vocoder is O(n*log(n)), where n is the number of samples processed per frame. The FFT phase vocoder is how we process audio data. In terms of memory, we only have 512 KB SRAM in the microprocessor. This limits the number of samples that can be stored and processed in RAM at one time, and also impacts how large each audio processing frame can be. We also have 384 KB ROM that we can use to store program code and static data.

Given that each sample in the .wav file is 2 bytes (16-bit audio), the available 512 KB SRAM can hold:

$$(512 \times 1024)/2 = 262144 \text{ samples}$$

At a sampling rate of 44.1 kHz, this gives us:

$$262144 / 44100 \approx 5.944 \text{ seconds of audio that we can store at a time}$$

This means that the ESP32-S3 can hold about 6 seconds of uncompressed audio in memory at once, allowing room for buffering and overlap processing.

Now let's focus on the latency and processing speed. The key parameters to consider for this are:

I. **Frame Size (F):** the number of samples processed in one frame

II. **Algorithm Time Complexity (C(n)):** The complexity of the time-stretching algorithm. For the phase vocoder, it's O(n*log(n)), as we calculated before.

III. **Processing Speed (S):** The ESP32-S3 runs at a speed of 240 MHz.

IV. **Memory Bandwidth (B):** The speed at which the ESP32-S3 can access and process memory.

V. **I/O Latency (L_{IO}):** The time delay caused by reading audio from external flash memory and writing it back to the audio output device.

Let's first break down the processing time needed per frame, which we can call $T_{process}$. The processing time will be based on the time complexity and clock speed, giving us:

$$T_{process} = C(n) \times S$$

where n is the number of samples per frame. This is thus the tolerance for the processing time needed per frame. In terms of memory and I/O latency, the latency is based on the speed at which data can be read from or written to memory. For tolerance, we want to minimize this latency. For

example, with real-time response time, the total time to process one frame (processing time + I/O latency) must allow for real-time tempo changes within 0.5 seconds. So:

$$T_{total} = T_{process} + L_{IO} \text{ and}$$

$$n \times T_{total} \leq 0.5 \text{ seconds}$$

So if we put all this together, we can get the total time taken per frame, to check our tolerance. Our processing time will be, as mentioned before, will be based on algorithmic complexity and processing speed of our microcontroller.

$$T_{process} = 240 \times 1{,}061{,}024 * \log_2(1024) = 240 \times 1{,}061{,}024 \times 10 \approx 4.26 \times 10^{-5}\ _{seconds} = 42.6\ \mu s$$

Our I/O latency per frame can be calculated assuming we have an I/O bandwidth of 20 MB/s. We also know that each frame has 1024 samples, and we have 2 bytes of sampling depth, so we get a data size of 2048 (2 KB) per frame. Hence:

$$L_{IO} = 2048\ _{bytes}/(20 \times 106_{(bytes/second)}) \approx 1.024 \times 10^{-4}\ _{seconds} = 102.4\ \mu s$$

Putting this all together, we can get the total time per frame:

$$T_{total} = T_{process} + L_{IO} = (4.26 \times 10^{-5}) + (1.024 \times 10^{-4}) \approx 1.45 \times 10^{-4}\ _{seconds} = 145\ \mu s$$

So based on all this, we can figure out that the frames that we can process within the 0.5 seconds we have are:

$$FPS = 0.5\ _{seconds}/T_{total} = 0.5\ _{seconds} / 1.45 \times 10^{-4}\ _{seconds/frame} \approx 3448\ _{frames}$$

Each frame contains 1024 samples, so the system can process:

$$Samples = FPS \times samples = 3448 \times 1024 \approx 3{,}529{,}152 \text{ samples}$$

Given that our sample rate is 44.1 kHz, the total time of audio processed is:

T_{net} = Samples / sample rate = 3,529,152 / 44,100 ≈ 80 seconds worth of audio data processed in 0.5 seconds.

This means that within 0.5 seconds, our system can only process 80 seconds worth of audio data from the .wav file, so we need to make sure the software has enough time to process all the audio files accordingly to different changes.

IV. Cost and Schedule

IV.I Cost Analysis

IV.I.I Labor Cost:

We estimate that each group member will put 10 hours per week into the project in order to get it done. This will be over the next 7 weeks, so each member will put in 70 hours, and in total, our group will put in 210 hours in order to complete the project. A typical entry-level engineer makes around $100,000 a year in salary, which comes out to roughly $50 per hour. The labor cost of this project will be $50 per hour * 3 people * 70 hours per person * 2.5 overhead factor = $26,250 in labor costs. A summary is included below:

Name	**Hourly Rate**	**Hours**	**Total**	**Total x 2.5**
Milind Sagaram	50	70	$3,500	$8750
Manas Gandhi	50	70	$3,500	$8750
Jack Prokop	50	70	$3,500	$8750
Total	150	210	$10,500	$26,250

IV.I.II Parts Cost:

We estimate the cost of our project to be $35.22, which falls well short of the $150 our group will be given. Additionally, many of our parts will be sourced from the Electronic Services Shop at the ECEB, which will be free of cost to us. These parts include the microcontroller, resistors, capacitors, voltage regulators, and other components. With this, we will have north of $115 to spend on other equipment, research and development, or other costs. A summary is included below:

Component	**Part Number**	**Quantity**	**Unit Cost**
Microcontroller	ESP32-S3-WROOM-1-N16	1	$3.48
Resistor	RMCF0805JG10K0	8	$0.10
Capacitor	CL21B105KBFNNNG	4	$0.10
Keyboard	Rii RK907	1	$9.99
Potentiometer	P160KN2-0QA25B10K	2	$1.83
5V Battery	3046-9V-ND	1	$4.45
Voltage Regulator	AZ1117CD-3.3TRG1	1	$0.44
Push Buttons	2223-TS02-66-70-BK-160-LCR-D	2	$0.12

In total, the cost of the project will be $26,285.46 with labor and materials costs included.

IV.II Schedule

Project Component	Estimated Completion Date	Group Member Responsible
Design Review	10/8	Manas
Preliminary PCB Design Review	10/11	Milind
PCB Final Design Completion	10/18	Milind
PCB Components Soldered	11/1	Milind
ESP-32 Drivers Completion	11/1	Jack
Audio Processing Algorithms Completion	11/15	Manas
Integration and Full Test Completion	11/21	Jack

V. Ethics and Safety

V.I IEEE & ACM Code of Ethics

V.I.I Privacy and Data Security (ACM Code 1.6) [3]:

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: The proposed software will prioritize the protection of user privacy and data security. While the software may require access to personal music files, we will implement measures to prevent unauthorized access or disclosure. This includes obtaining explicit user consent for data collection and storage, and adhering to industry-standard data security practices. Such practices might involve encryption, secure data transmission protocols, and regular security audits.

V.I.II Intellectual Property (ACM Code 1.5) [4]:

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.

V.I.III Honesty and Integrity (IEEE Code 7.8.I) [2]:

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.

V.II Safety

Our project does not pose any major security risks, beyond soldering and power supply management.

V.II.I Power Supply and Electrical Safety

The DJ board will be powered by a low-voltage 9V battery to avoid the use of high-voltage power supplies, which pose a significant risk of electric shock or fire. We will make sure to only use manufacturer-approved batteries, store them in cool and dry environments when not in use, and inspect them for damage before ever plugging them in. By following these safety restrictions, we will ensure that we have no issues with electrical safety

V.II.I Soldering

When soldering, we will make sure that we have personal protective equipment including safety glasses and soldering iron stands. We will also make sure that we have completed the required training before engaging in any activity that includes high heat, sharp blades, or other health hazards.

VI. References

[1] Amazon.com Inc. (n.d.). Dj Board [Amazon.com]. Retrieved from https://www.amazon.com/dj-board/s?k=dj+board

[2] DiDiodes Incorporated. (n.d.). document number: DS36736 Rev. 7 - 3 [SNIPPET] AZ1117 1A LOW DROPOUT LINEAR REGULATOR [Data sheet]. https://www.diodes.com/assets/Datasheets/products_inactive_data/AZ1117.pdf

[3] Association for Computing Machinery. (n.d.). *ACM Code of Ethics and Professional Conduct*. ACM. https://ethics.acm.org/

[4] Institute of Electrical and Electronics Engineers. (n.d.). *IEEE Code of Ethics*. IEEE. https://www.ieee.org/about/corporate/governance/p7-8.html

[5] Wilderness Labs. (n.d.). Resistor Tolerance and Preferred Values. Wilderness Labs Developer Portal. Retrieved November 7, 2024, from https://developer.wildernesslabs.co/Hardware/Tutorials/Electronics/Part4/Resistor_Tolerance