Piano Visualizer Final Report

By

Jay Park

Nuwan Singhal

Sarayu Suresh

Design Document for ECE 445, Senior Design, Spring 2026

TA: Wenjing Song

6th May 2026

Project No. 98

Abstract

The Piano Visualizer is a standalone piano learning system designed to help beginner musicians identify correct keys, timing, and note duration without relying on phone-based applications or proprietary light-up keyboards. The system reads Standard MIDI files from a microSD card, processes note and timing data using an STM32F446RET6 microcontroller, and displays the expected notes on a 64×32 RGB LED matrix. The final design uses physical buttons for song selection, a 12 V wall adapter with a buck converter for power regulation, and timer-based bit-banging to drive the LED matrix. The microphone subsystem from the original design was removed to focus on reliable MIDI-file-based playback and display.

The completed system successfully displayed MIDI-based note guidance at a measured refresh rate of 62.5 Hz, operated without visible flicker or ghosting, allowed users to select multiple songs from the microSD card, and powered the LED matrix, microcontroller, and supporting subsystems without overheating. The microSD subsystem used 1-bit SDIO communication and supported efficient reading and parsing of MIDI files. However, the MIDI input subsystem was not fully functional because the optocoupler circuit did not provide a valid signal to the microcontroller. As a result, the final system met the display, file-reading, user interface, and power requirements, but did not verify the real-time MIDI correctness feedback requirement.

Contents

1. Introduction

1.1 Problem

Learning piano is difficult for beginners because the feedback loop is slow. A new player must translate sheet music into correct keys, timing, and note duration while also trying to stay on tempo. When notes occur quickly or simultaneously, it can be hard for beginners to recognize mistakes in real time. This often leads to repeated errors, inefficient practice, and frustration.

Existing piano learning tools partially address this problem but introduce other barriers. App-based tools often depend on a phone, internet access, a camera, or a microphone, which may not be reliable or convenient in every practice environment. Commercial light-up keyboards and proprietary learning systems can also be expensive or tied to specific hardware. Our project addresses this gap by creating a standalone visual guidance system that reads songs stored locally and displays what the user should play on an LED matrix.

1.2 Solution

Our project is a real-time piano visualizer that reads Standard MIDI files from a microSD card and displays note guidance using a 64×32 RGB LED matrix. The system parses note number, velocity, and timing data from the selected MIDI file and displays which piano keys should be pressed and for how long. Users select songs through physical buttons, and the selected song is displayed through the LED matrix interface.

The system is built around the STM32F446RET6 microcontroller, which coordinates communication between the microSD card, LED matrix, user interface, power subsystem, and MIDI input circuitry. The final design provides a standalone learning tool that visually guides users through stored songs without requiring a phone, internet connection, or proprietary keyboard system.

1.3 Visual Aid:

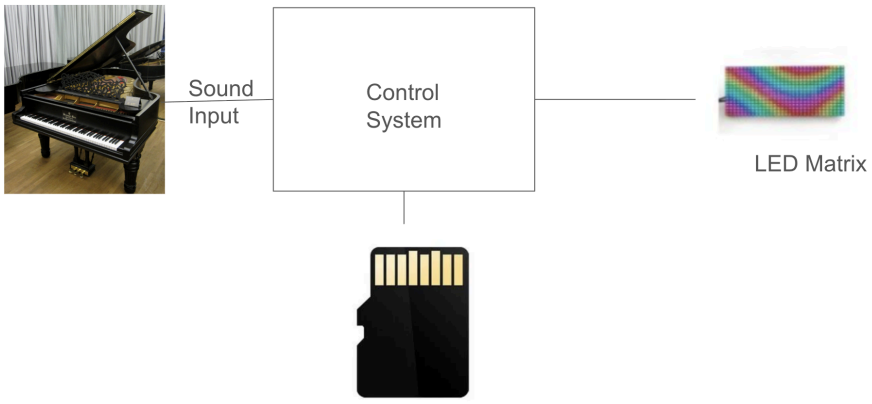

Figure 1 Visual Aid for our project

Figure 1 shows the high-level concept of the Piano Visualizer. The microSD card stores the MIDI songs that the user wants to practice. After the user selects a song, the control system reads the MIDI file, processes the note and timing data, and sends display information to the LED matrix. The LED matrix then shows the piano notes and note durations so the user can follow along visually.

1.4 High-level Requirements

1. The system shall read, parse, and display Standard MIDI files from a microSD card on the 64×32 RGB LED matrix while maintaining a display refresh rate of at least 60 Hz.
2. The system shall support MIDI input from a keyboard, detect Note On and Note Off events, and update the LED matrix with an end-to-end latency of 30 ms or less while supporting up to 10 simultaneous notes.
3. The system shall allow users to select songs of their choice through a microSD card using the physical user interface.

1.5 Changes

Several major changes were made from the original design. First, the microphone subsystem was removed. The original design included microphone input as a secondary option for acoustic piano note detection, but this was removed so the team could focus on reliable MIDI-file-based playback and visual guidance.

Second, the power subsystem changed from a specialized 5 V wall adapter to a standard 12 V wall adapter with a buck converter. This made the power input easier to source and allowed the board to regulate the voltage needed by the LED matrix and logic circuitry. The reason behind this change was to make the product more accessible as 12V wall adapters are significantly more common and cheaper than the specialized one we had originally planned.

Third, the LED control strategy changed from Direct Memory Access to timer-based bit-banging. The final timer-based approach successfully drove the LED matrix at 62.5 Hz, which exceeded the 60 Hz requirement and produced a stable display without visible flicker. This change was mainly due to the complexity required to implement DMA, which was originally thought to be necessary to have enough CPU time but during development we saw that bit-banging was sufficient.

2 Design

2.1 Block Diagram

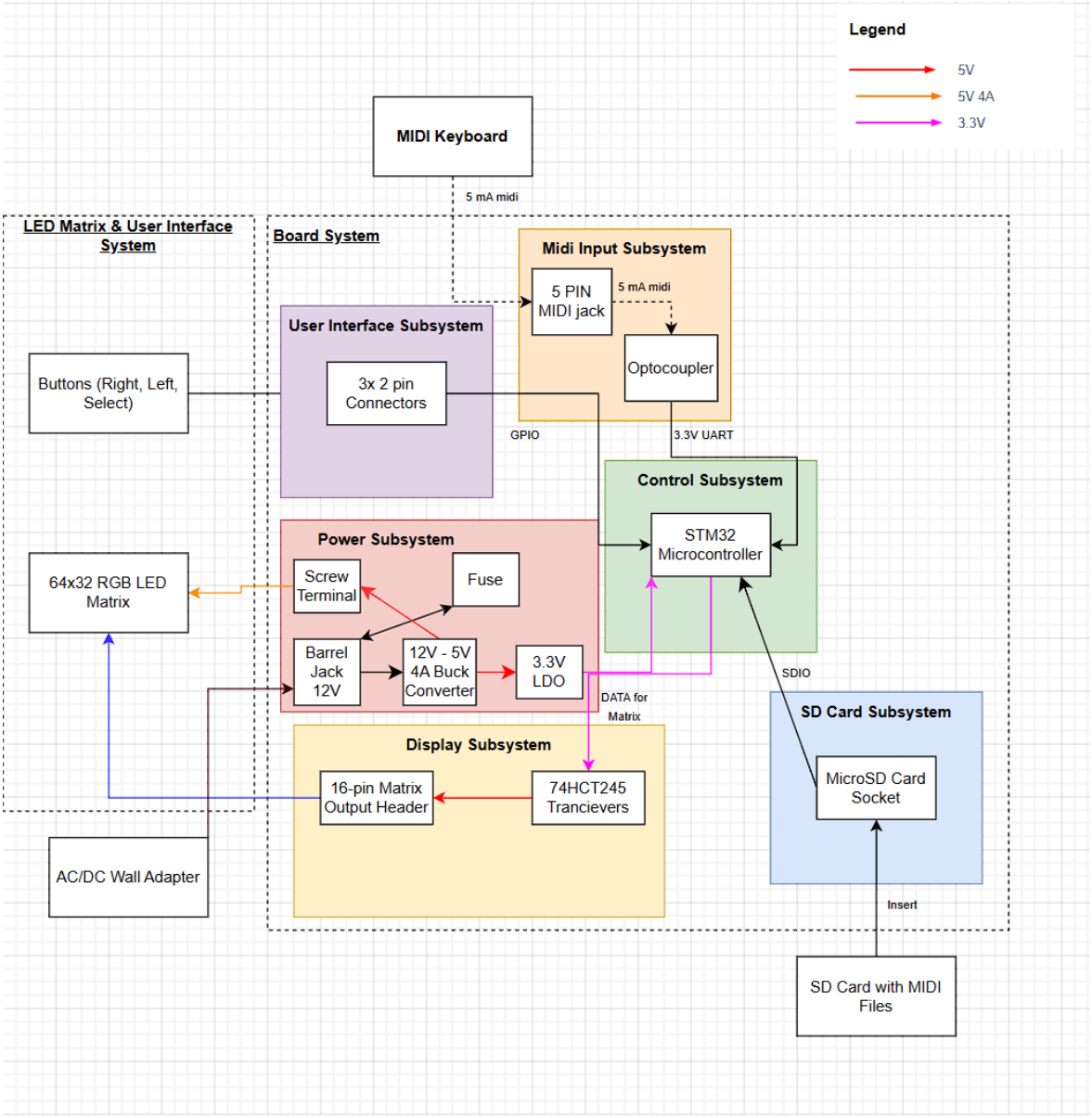

Figure 2 Block Diagram of the entire system.

2.2 Subsystem Overview

2.2.1 RGB LED Matrix Subsystem

The RGB LED Matrix Subsystem provides real-time visual output to the user and serves as the primary feedback interface of the system. It displays the piano keys that should be pressed according to the selected MIDI file and represents note duration as the song progresses. This subsystem directly supports

the first high-level requirement by ensuring that note position and timing information are presented clearly without perceptible flicker.

The selected display is a 64×32 HUB75 RGB LED matrix operating at 5 V with a maximum current draw of approximately 4 A under high-brightness conditions. The panel uses a 1/16 scan multiplexing scheme, which requires continuous row updates to maintain a stable image. To achieve a minimum refresh rate of 60 Hz, the microcontroller must update all 16 scan rows at a rate of at least 960 row updates per second.

Because the STM32F446RET6 microcontroller operates at 3.3 V logic levels and the LED matrix expects 5 V logic inputs, 74HCT245 transceivers are used for level shifting. These transceivers allow the microcontroller to reliably drive the display inputs and reduce signal integrity issues.

The final design uses timer-based bit-banging to update the LED matrix. The subsystem interfaces electrically with the Power Management Subsystem for the high-current 5 V supply and logically with the Control Subsystem, which provides the note and timing data required for display generation. The subsystem was verified by measuring the display refresh rate and visually inspecting the matrix during playback. The final implementation achieved a measured refresh rate of 62.5 Hz, exceeding the 60 Hz requirement, with no apparent flickering or ghosting.

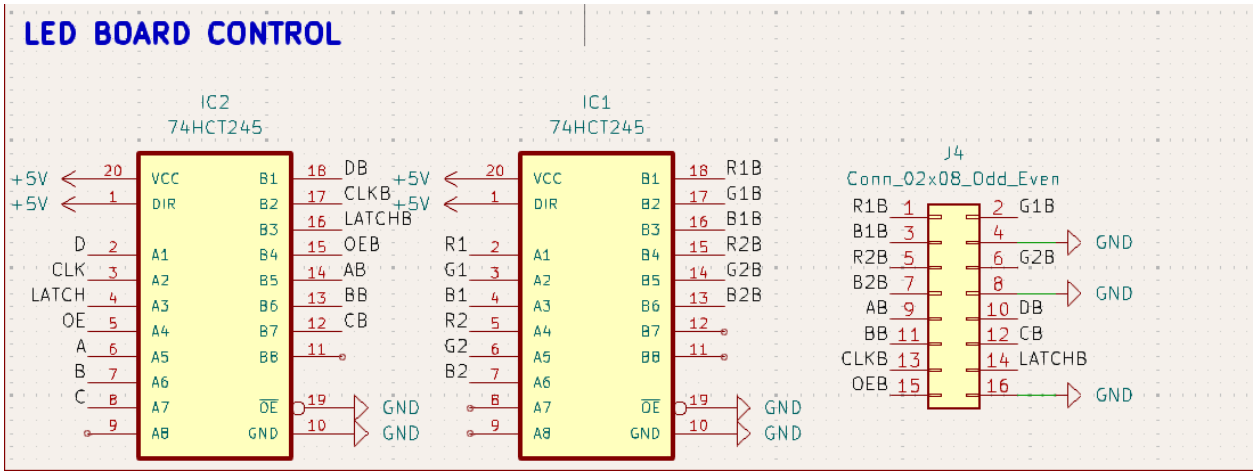

Figure 3 RGB LED Matrix Subsystem Schematic

2.2.2 SD Card Storage Subsystem

The SD Card Storage Subsystem stores the Standard MIDI files used by the Piano Visualizer for visual note guidance. This subsystem supports the first high-level requirement by allowing the system to retrieve note number, velocity, and timing information from locally stored MIDI files and use that information to generate the LED matrix display.

A microSD card socket is mounted directly on the PCB and interfaces with the STM32F446RET6 microcontroller using SDIO communication. The final implementation uses 1-bit SDIO rather than 4-bit SDIO because the wider bus did not operate reliably during testing, likely due to a soldering or signal connection issue. Even with 1-bit SDIO, the subsystem achieved a read speed of approximately 12.5 MB/s, which was sufficient for loading and parsing MIDI files without noticeable delay during normal operation.

When the user selects a song, the microcontroller reads the selected MIDI file from the microSD card and extracts the note events, timing deltas, and velocity information. The parsed MIDI data is then used by the Control Subsystem to determine which notes should be displayed on the LED matrix and how long each note should remain visible.

The subsystem supports multiple MIDI files stored on the microSD card, allowing the user to select between songs without changing the firmware. During testing, multiple file selection worked successfully through the user interface. One MIDI file failed to load correctly, which suggests that future versions should include stronger file validation, clearer error messages, and more robust handling of unsupported or corrupted MIDI files.

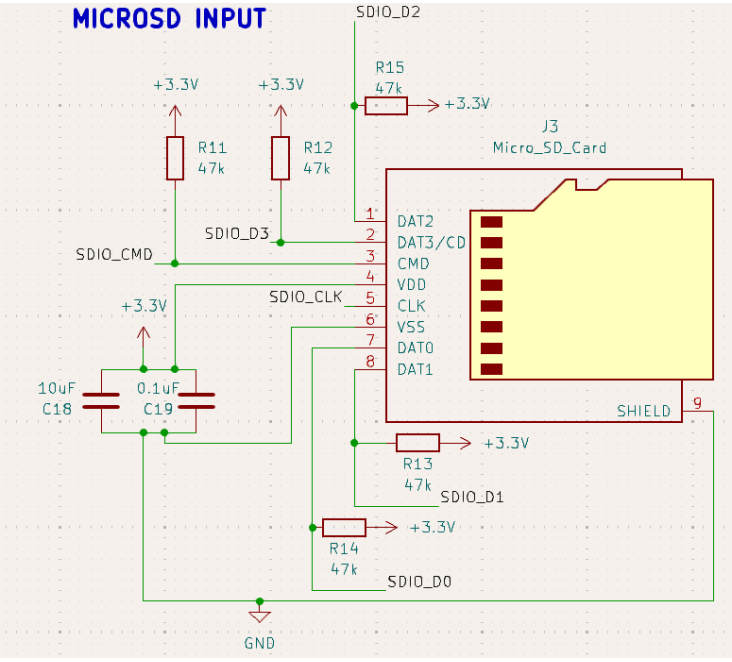

Figure 4 SD Card Storage Subsystem Schematic

2.2.3 MIDI Input Subsystem

The MIDI Input Subsystem is responsible for receiving real time note data from an external digital piano and providing low latency input to the system for correctness evaluation. This subsystem directly satisfies the second high level requirement by ensuring that Note On and Note Off events are detected and processed within the required 30 millisecond latency window.

A standard 5 pin DIN MIDI connector is used to receive input from the keyboard. MIDI communication follows the MIDI 1.0 electrical specification and operates as a 5 mA current loop at a baud rate of 31,250 bits per second using asynchronous serial transmission. To ensure electrical isolation between the external instrument and the embedded system, a 6N138 optocoupler is used in accordance with the MIDI standard. This prevents ground loops and protects the microcontroller from external voltage transients.

The output of the optocoupler is conditioned to produce a clean 3.3 volt logic signal compatible with the STM32F446RET6 UART peripheral. The microcontroller UART is configured to operate at 31,250 baud with 8 data bits, no parity, and one stop bit. Incoming MIDI bytes are buffered and parsed in firmware to detect Note On and Note Off messages, extract velocity values, and compute note duration based on timestamp differences.

The parsed MIDI events are compared against the expected note sequence extracted from the SD Card Subsystem. Correctness and timing accuracy are then passed to the RGB LED Matrix Subsystem for immediate visual feedback. Verification of this subsystem will include confirming correct baud rate operation using a logic analyzer, validating electrical isolation through continuity testing, and measuring end to end input latency to ensure it does not exceed 30 milliseconds.

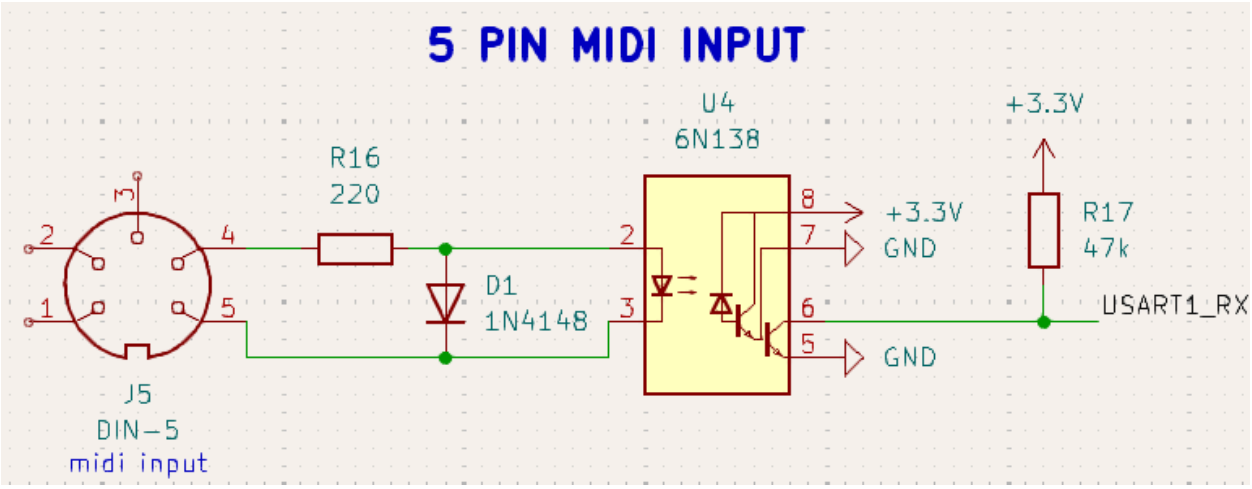

Figure 5 MIDI Input Subsystem Schematic

2.2.4 User Interface Subsystem

The User Interface Subsystem is responsible for allowing the user to select songs, adjust playback parameters, and control system operation. This subsystem enables interaction with the SD Card Storage Subsystem for song selection and configures operational modes that affect the MIDI Input and RGB LED Matrix subsystems.

User input is provided through physical push buttons connected to general purpose input pins on the STM32F446RET6 microcontroller. These buttons allow navigation through stored MIDI files, selection of playback modes such as normal, wait mode, and single hand practice, adjustment of playback speed, and control of start and stop functionality. Each button input is configured with appropriate pull-up or pull-down resistors and is debounced in firmware to prevent false triggering due to mechanical contact bounce.

System feedback related to menu navigation and operational state is displayed directly on the RGB LED Matrix. Simple text indicators and status symbols are rendered using predefined font patterns stored in memory. This approach eliminates the need for a separate LCD display while maintaining clear user feedback.

The subsystem interfaces electrically with the microcontroller GPIO pins and logically with the SD Card and MIDI subsystems by modifying configuration variables that affect playback behavior. Verification of this subsystem will include confirming correct detection of button presses under repeated operation, validating proper debounce behavior, and ensuring that menu selections result in the intended system configuration changes.

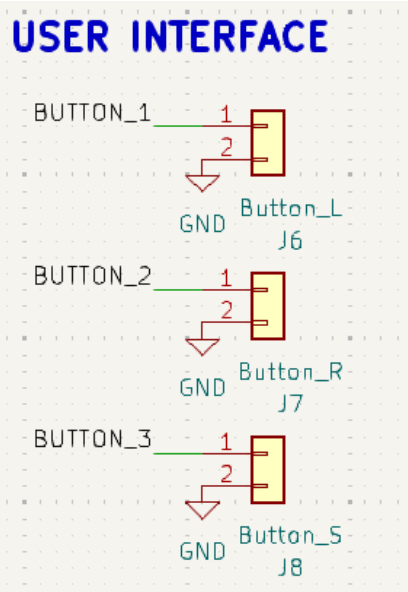

Figure 6 User Interface Subsystem Schematic

2.2.5 Power Management Subsystem

The Power Management Subsystem provides stable and protected power to the LED matrix, microcontroller, microSD card, and supporting circuitry. This subsystem is important because the LED matrix requires a high-current 5 V supply, while the STM32F446RET6 microcontroller and logic components require 3.3 V.

The final design is powered by a standard 12 V wall adapter. A buck converter steps the 12 V input down to 5 V for the 64×32 RGB LED matrix. The LED matrix can draw up to approximately 4 A under high-brightness conditions, so the power system was designed to provide a stable 5 V rail with enough current capacity for reliable display operation. The measured LED matrix voltage stayed close to the expected 5 V value during testing, ranging from approximately 4.98 V to 5.01 V across five trials.

The STM32F446RET6 microcontroller and supporting logic operate at 3.3 V. A 3.3 V regulator is used to generate the logic supply from the available power rail. Decoupling capacitors are included near the regulator and microcontroller to reduce voltage ripple and improve stability during switching and display updates.

Overcurrent protection is included using a fuse in series with the input power path. This protects the system from excessive current draw or short-circuit conditions, especially because the LED matrix can draw significant current. Screw terminals are used for secure power connections to reduce the risk of loose or intermittent wiring during operation.

The subsystem was verified during full-system operation by powering the LED matrix, microcontroller, microSD card, and supporting logic. The system successfully supplied the required power rails, operated without overheating, and maintained a stable LED matrix voltage during testing.

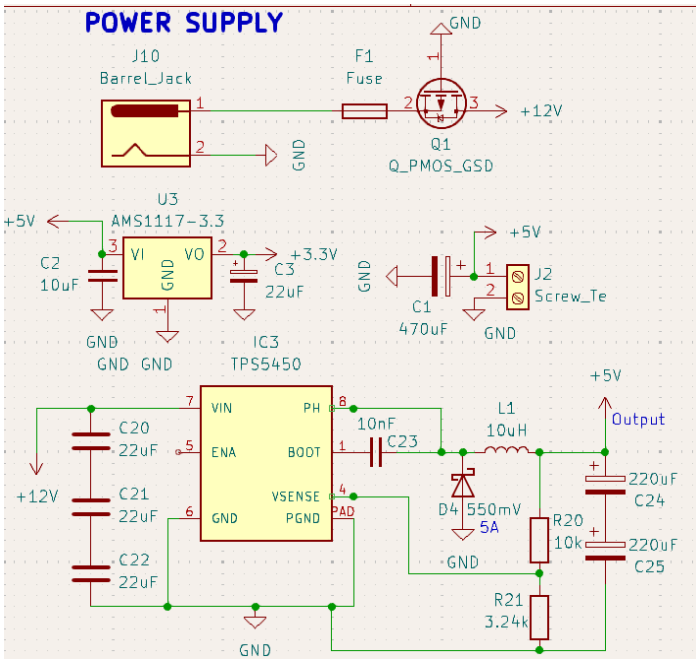

Figure 7 Power Management Subsystem Schematic

2.2.6 Control Subsystem

The Control Subsystem is centered around the STM32F446RET6 microcontroller and serves as the main processing and coordination unit of the Piano Visualizer. It manages communication between the microSD card, LED matrix, user interface, power subsystem, and MIDI input circuitry.

The microcontroller reads Standard MIDI files from the microSD card using 1-bit SDIO communication. After a song is selected, the firmware parses the MIDI file to extract note number, velocity, and timing information. This parsed data is stored in memory and used to determine the visual note sequence displayed on the LED matrix.

The Control Subsystem receives user input through GPIO-connected physical buttons. These inputs allow the user to navigate between stored songs, select a song, and begin playback. The firmware prioritizes user input over background tasks so the system can respond quickly when the user changes options or returns to the main menu.

The microcontroller drives the 64×32 RGB LED matrix using timer-based bit-banging. This replaced the original Direct Memory Access approach and allowed the team to debug the display timing more directly. The final implementation achieved a measured refresh rate of 62.5 Hz, exceeding the 60 Hz display requirement and producing a stable image without visible flicker.

The MIDI input circuitry was included in the final hardware design and was intended to send Note On and Note Off events to the microcontroller through a UART input. However, the MIDI signal did not successfully pass through the optocoupler to the microcontroller during final testing, so the real-time correctness feedback portion of the control logic could not be fully verified.

The Control Subsystem was verified by confirming that the STM32F446RET6 could coordinate SD card reading, MIDI file parsing, LED matrix updates, and button input. The final system had sufficient processing time to read and parse MIDI files, update the display, and respond to user commands with minimal delay.

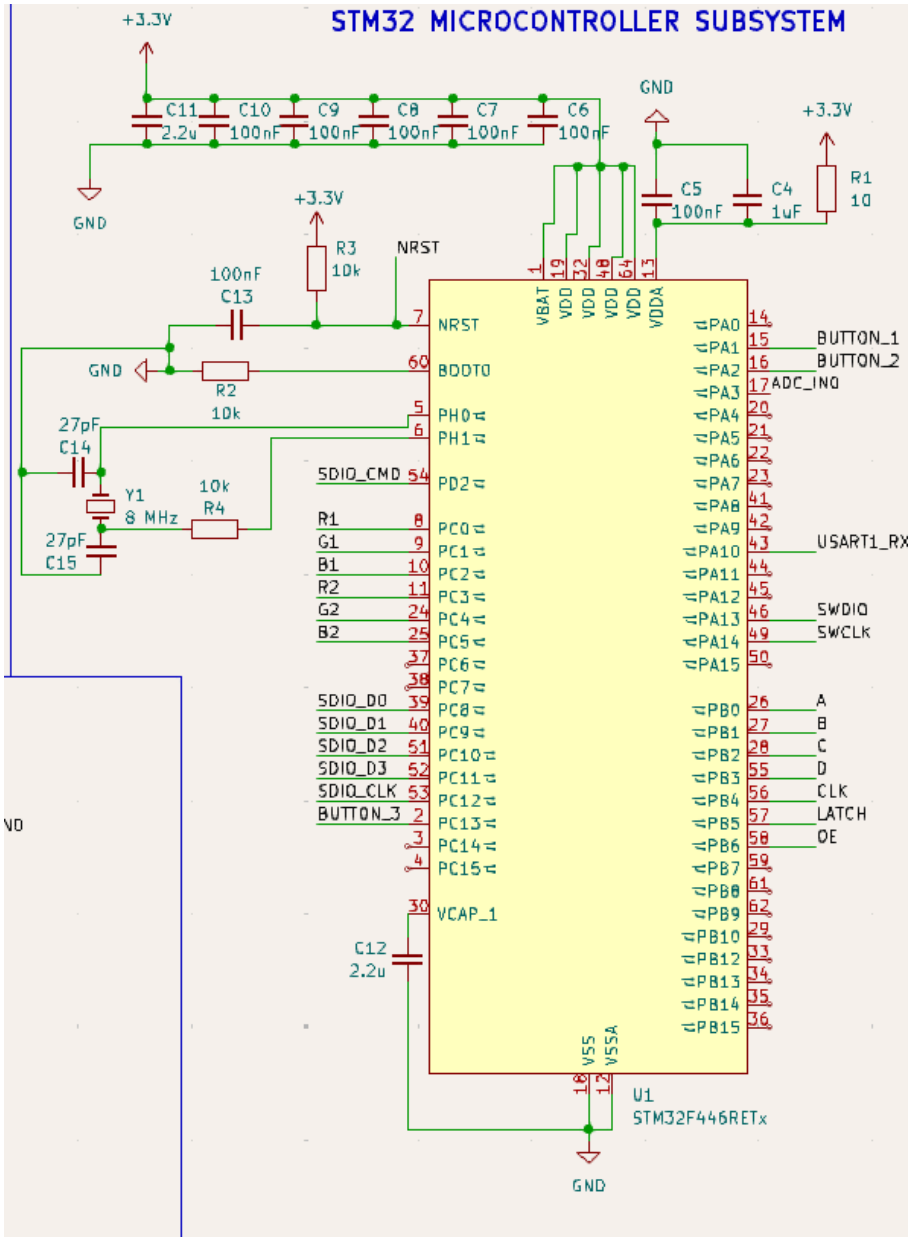

Figure 8 Control Subsystem Schematic

2.3 Verification

The completed Piano Visualizer was tested at both the subsystem level and the full-system level to determine whether the final design met the high-level requirements. The main requirements tested were MIDI file reading from the microSD card, LED matrix display stability, user song selection, power regulation, and control system timing. The full Requirement and Verification tables are included in the appendix.

The RGB LED Matrix Subsystem was tested by displaying parsed MIDI note data on the 64×32 HUB75 matrix and measuring the display behavior during playback. The display achieved a measured refresh rate of 62.5 Hz, which exceeds the 60 Hz high-level requirement. During testing, there was no visible

flickering, ghosting, or noticeable leading or lagging of the note display. The LED matrix supply voltage was also measured across five trials, as shown in Figure 3. The expected voltage was 5 V, and the measured voltage ranged from approximately 4.98 V to 5.01 V. This is well within the acceptable ±5% tolerance range for a 5 V supply, which confirms that the matrix received stable power during operation.

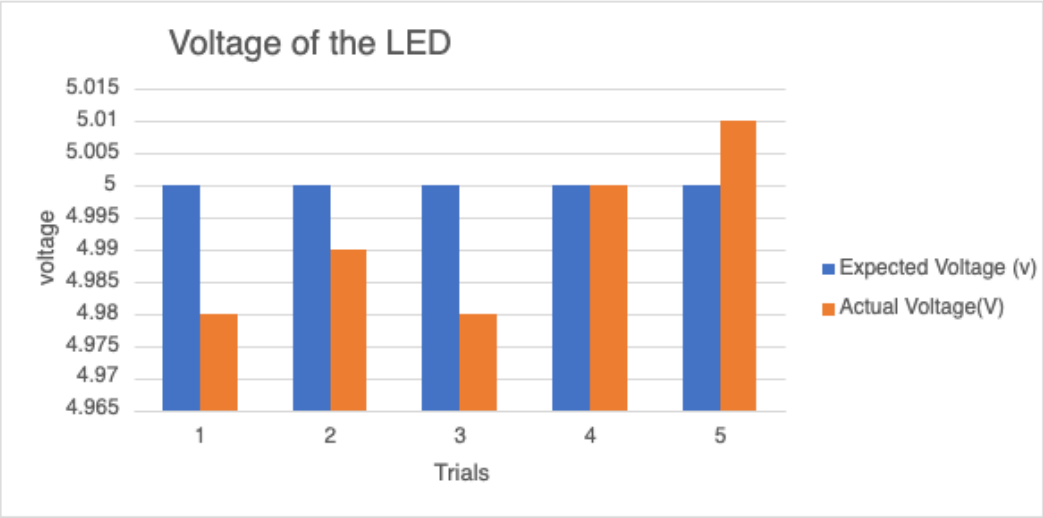

Figure 9 LED Voltage Graph

The MicroSD Storage Subsystem was tested by loading multiple Standard MIDI files from the microSD card and confirming that the selected songs could be parsed and displayed on the LED matrix. The final system used 1-bit SDIO communication and achieved a read speed of approximately 12.5 MB/s. Multiple file selection worked successfully through the user interface. During testing, one MIDI file failed to load correctly, which indicates that the subsystem was functional overall but would benefit from stronger file validation and error handling in future revisions.

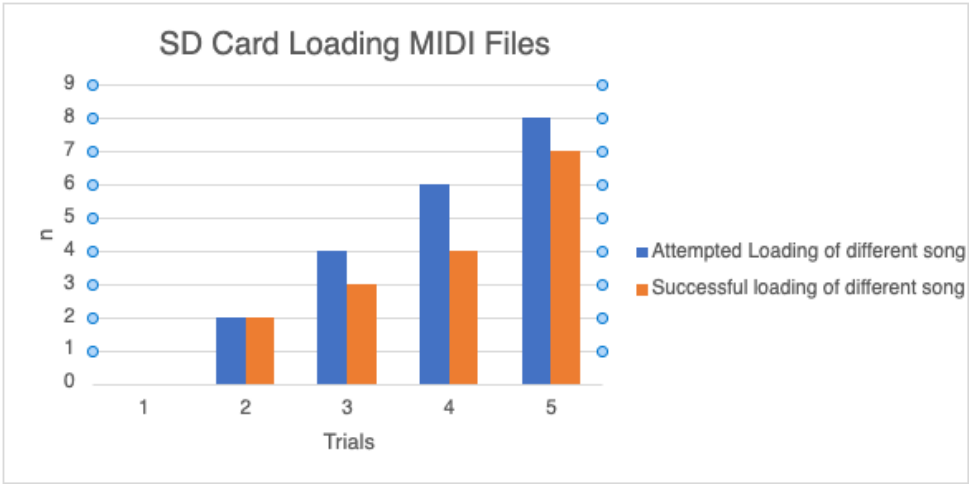

Figure 10 SD Card Loading MIDI Files Graph

The User Interface Subsystem was tested by repeatedly pressing the physical navigation and selection buttons and observing whether the system state updated correctly. The buttons successfully allowed the user to navigate between songs, select a file from the microSD card, and start playback. Button input produced close to no noticeable delay when switching options or beginning playback. This verified that the user interface met the requirement of allowing users to select songs through physical controls.

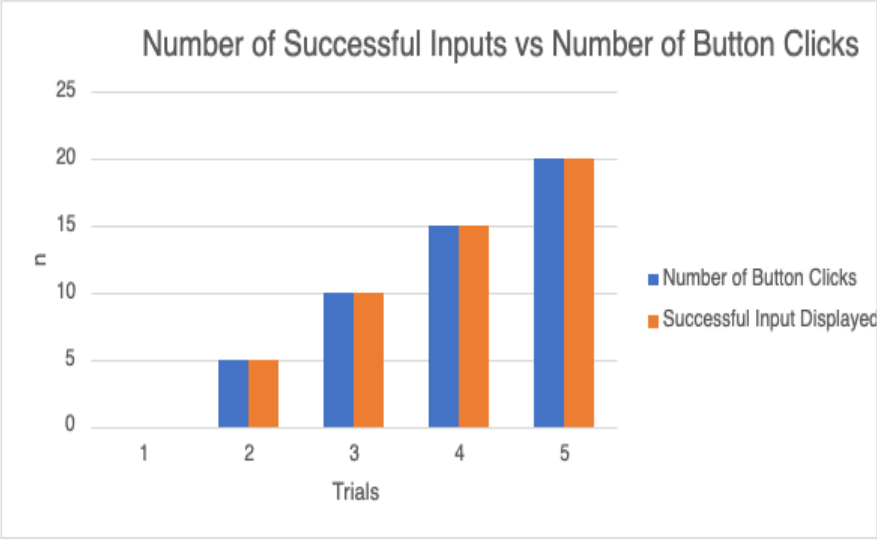

Figure 11 Button Click Successful Input Graph

The Power Management Subsystem was tested by powering the LED matrix, microcontroller, microSD card, and supporting logic during normal system operation. The system successfully supplied 5 V to the LED matrix and powered the remaining subsystems without overheating. The voltage measurements from the LED matrix testing also support that the 5 V rail stayed stable during operation.

The Control Subsystem was verified by confirming that the STM32F446RET6 could coordinate SD card reading, LED matrix output, and user input. The system had enough processing time to read and parse MIDI files, update the LED matrix, and respond to button inputs. User input was prioritized over background tasks, allowing the system to return to the main menu or change states without significant delay.

The MIDI Input Subsystem was tested by checking the signal path from the MIDI adapter to the optocoupler and then to the microcontroller. MIDI data was observed before the optocoupler, but the signal did not successfully reach the microcontroller. Because of this, the final system could not verify the high-level requirement for detecting Note On and Note Off events within 30 ms or supporting up to 10 simultaneous notes. This was the main requirement that was not met in the final design.

Overall, the completed system met the requirements related to MIDI file storage, song selection, LED matrix display, power stability, and control system operation. The main unresolved issue was the MIDI input circuit, which should be redesigned and tested separately in future work before being integrated into the full PCB.

2.4 Design Alternatives

The main design issue we faced was with the MIDI input subsystem, particularly the optocoupler. We found that our midi adapter was pulsing data through a positive voltage difference between pins 4 and 5 of the 5-pin DIN. However, we found that our USB A to MIDI adapter was pulsing a negative voltage difference between these two pins, this is likely due the adapter not following the standard MIDI specifications. To found that that with this adapter the voltage across the internal LED of the optocoupler was -1.7 Volts, when the intended voltage was 1.1 to 1.9V. This issue could have been resolved by either sourcing a MIDI adapter that fit the specifications, which we found would cost around 40 dollars which would push us close to the limit of our budget. The other alternative would have been to rewire our PCB such that the traces going from pin 4 and 5 of the DIN are swapped, unfortunately we encountered this issue very late into the project timeline and were unable to solve it.

3 Costs

Summing up all of the parts we ordered as listed in appendix B, the total cost of the project came out to 124.78 dollars. The majority of this cost was the LED board as well as 2 STM32F4's and does not include the price of shipping or the PCB and stencil.

Had the labour been included in the cost, the labor cost per person should be 30 dollars/hour (USD)*2.5*80 hours = 6,000 dollars per person for this project. In total, the labor cost of the whole project would be 18,000 dollars.

4 Schedule

The schedule for our project can be seen in Appendix C.

5 Ethics, Safety, and Societal Impact

The Piano Visualizer has ethical, safety, and broader-impact considerations because it is an educational electronic device intended for repeated user interaction. In accordance with the IEEE Code of Ethics, the design considers user safety through regulated voltage rails, overcurrent protection, and secure power connections [4]. The LED matrix requires a relatively high-current 5 V supply, so the power subsystem includes a fuse and stable voltage regulation to reduce the risk of overheating, short circuits, or unsafe operation.

The project also follows the IEEE expectation that engineers make honest and realistic claims based on available data [4]. For this reason, the report identifies which requirements were verified and which were not fully verified. The system successfully verified stored MIDI file playback, LED matrix display,

user song selection, power stability, and control-system operation. However, the MIDI input subsystem did not fully verify real-time correctness feedback because the signal did not successfully reach the microcontroller through the optocoupler circuit.

From a societal and economic perspective, the Piano Visualizer supports more accessible music education. Many beginner piano-learning tools require paid applications, internet access, microphones, cameras, or proprietary light-up keyboards. This system uses locally stored MIDI files and a standalone LED matrix display, which can make practice more flexible and potentially lower cost for beginners who benefit from visual guidance. The system also does not require internet access, cloud processing, or user data collection, which reduces privacy concerns.

Environmentally, the project uses standard electronic components and a reusable hardware platform. Future versions should improve repairability, power efficiency, and responsible disposal of electronic parts. Overall, the project's broader impact is that it provides a locally controlled and reusable alternative to app-based piano learning systems while encouraging safe and honest engineering design.

6 Conclusion

The Piano Visualizer successfully demonstrates the core concept of a standalone visual piano learning system. The final system reads MIDI files from a microSD card, allows users to select songs through physical buttons, and displays note position and duration on a 64×32 RGB LED matrix. Based on the completed verification, the design works reliably as a stored-MIDI visual guidance device and provides a strong foundation for a more complete interactive piano learning tool.

The project achieved several major accomplishments. The LED matrix produced a stable and readable display, the microSD subsystem allowed local song storage and selection, the user interface responded with minimal delay, and the control subsystem coordinated file reading, display output, and button input successfully. The final design also became more reliable through several design changes made during testing. The microphone subsystem was removed to reduce signal-processing complexity, the power system was changed to use a standard 12 V adapter with a buck converter, and the LED control method was changed to timer-based bit-banging.

The main remaining uncertainty is the MIDI input subsystem. Since the MIDI signal did not successfully reach the microcontroller through the optocoupler circuit, the final system did not fully verify real-time correctness feedback. A practical adjustment to the final performance specification is to define the completed system as a stored-MIDI visualizer rather than a full MIDI-feedback tutor. To restore the original goal in a future revision, the MIDI input circuit should be rebuilt and tested separately, with the DIN connector, optocoupler, resistor values, diode orientation, and UART signal checked before PCB integration. Additional test points should also be added to make debugging easier.

Overall, the project achieved its primary goal of creating a functional LED-based piano visualizer. While the real-time MIDI input requirement was not fully verified, the completed design proves that stored MIDI files can be translated into clear visual piano guidance using an embedded system and LED matrix. Future work should focus on correcting the MIDI input circuit, adding playback speed control, supporting

one-handed practice, adding sheet music mode, saving progress points, and improving the enclosure for a more polished final product.

References

[1] Adafruit Industries. (2020). 64x32 RGB LED Matrix - 6mm pitch. Adafruit.com. https://www.adafruit.com/product/2276?gad_source=1&gad_campaignid=21079227318&gbraid=0AAAADx9JvRWYU5tJsMEXvWKXJLSAmWvC&gclid=CjwKCAiAtLvMBhB_EiwA1u6_Pt3TwoBxsc8LnE150b79cn8LxB16iv_X7rbdVaUUnf4V9SRT6fXxnBoCLwgQAvD_BwE

[2] Datasheet - STM32F446xC/E. (2021). https://www.st.com/content/ccc/resource/technical/document/datasheet/65/cb/75/50/53/d6/48/24/DM00141306.pdf/files/DM00141306.pdf/jcr:content/translations/en.DM00141306.pdf

[3] Getting started with STM32F4xxxx MCU hardware development. (2018). https://www.st.com/resource/en/application_note/an4488-getting-started-with-stm32f4xxxx-mcu-hardware-development-stmicroelectronics.pdf

[4] Illinois Information Technology Accessibility Act Standards 1.0. (2026). State.il.us. https://www.dhs.state.il.us/IITAA/IITAAStandards.html

[5] Institute of Electrical and Electronics Engineers. (2020). IEEE Code of Ethics. https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf

[6] Interface SD card with STM32 using SDIO || 4Bit Mode || DMA (2025). Youtube.com. https://www.youtube.com/watch?v=KNuMM7NdgYw&t=81s

[7] Jonathan. (2025, January 19). Interface an SD Card through SDIO on the STM32 BlackPill (Part 1) - Phipps Electronics. Phipps Electronics. https://www.phippselectronics.com/interface-an-sd-card-through-sdio-on-the-stm32-blackpill-part-1/

[8] Labs, L. (2024, March 30). Tang Nano 9K: HUB75 LED Panels. Lushay Labs. https://learn.lushaylabs.com/led-panel-hub75/

[9] MIDI 1.0 Detailed Specification. (2024, January 19). MIDI.org. https://midi.org/midi-1-0-detailed-specification

[10] Power Designer. (2026). Ti.com. https://webench.ti.com/power-designer/switching-regulator/select

[11] riktw. (2020, April 27). Driving a 64*64 RGB LED panel with an FPGA. - jaeblog. Jaeblog. https://justanotherelectronicsblog.com/?p=636

[12] SD-50BV Symbol, Footprint & 3D Model by Same Sky | SnapMagic Search (formerly SnapEDA). (2026). Snapeda.com. https://www.snapeda.com/parts/SD-50BV/Same%20Sky/view-part/

[13] STM32F446RET6 | DigiKey Electronics. (2026). DigiKey Electronics. https://www.digikey.com/en/products/detail/stmicroelectronics/STM32F446RET6/5175962

[14] STM32 PCB Design (2022). Phil's Lab. https://www.youtube.com/watch?v=PMEpQZ90f34&list=WL&index=2&t=1789s

[15] snowsh. (2022, October 16). MIDI 5 pin DIN and TRS swiching circuit. Teensy Forum. https://forum.pjrc.com/index.php?threads/midi-5-pin-din-and-trs-swiching-circuit.71303/

[16] "Piano." Wikipedia, February 23, 2026. https://en.wikipedia.org/wiki/Piano.

[17] Texas Instruments 74HC245: Symbol, Footprint, 3D STEP Model | Ultra Librarian. (2024). Ultralibrarian.com. https://app.ultralibrarian.com/details/01cf4bd4-1075-11e9-ab3a-0a3560a4cccc/Texas-Instruments/74HC245?exports=42&open=exports

[18] Wong, Amanda. "Learn More about SD Card, Mini SD Card & Micro SD Card." iBoysoft, February 26, 2026. https://iboysoft.com/wiki/sd-card.html.

Appendix A Requirement and Verification Tables

Table 8 R&V Table for All Subsystems

LED Matrix Subsystem	
Requirements	Verification
• The RGB LED Matrix Subsystem shall operate at 5 V and maintain a minimum refresh rate of 60 Hz to prevent visible flicker.	• Power the LED matrix using the 5 V supply. • Use an oscilloscope to measure the row scan frequency. • Confirm that the effective display refresh rate is at least 60 Hz. • Visually inspect the display to confirm absence of perceptible flicker.
• The subsystem shall correctly display at least four octaves of piano keys and represent note duration in real time.	• Load a test MIDI file covering four octaves. • Confirm that each MIDI note maps to the correct LED position. • Verify that note duration matches expected timing from the MIDI file.
• The subsystem shall safely interface 3.3 V MCU logic to 5 V display inputs using level shifting.	• Measure MCU output voltage levels. • Measure matrix input voltage levels after the 74HCT245 transceiver. • Confirm proper 5 V logic levels are achieved without signal distortion.
MicroSD Storage Subsystem	
Requirements	**Verification**
• The subsystem shall successfully read and parse Standard MIDI files from a microSD card using SDIO communication.	• Insert a formatted microSD card containing valid MIDI files. • Power the system and select a song. • Confirm that note number, velocity, and timing data are correctly extracted.
• The subsystem shall allow users to select among multiple stored MIDI files.	• Store at least three distinct MIDI files on the SD card. • Navigate through the user interface and confirm correct file selection and playback.
• The subsystem shall operate without data corruption during repeated read	• Repeatedly load and parse the same MIDI file at least 20 times.

operations.	● Confirm consistent and identical note output across trials.
MIDI Input Subsystem	
Requirements	**Verification**
● The subsystem shall accept standard 5 pin DIN MIDI input at 31250 baud.	● Connect a MIDI keyboard transmitting at 31250 baud. ● Use a logic analyzer to confirm correct reception of serial data.
● The subsystem shall detect Note On and Note Off events with latency not exceeding 30 ms.	● Trigger a note on the keyboard. ● Measure the time between MIDI signal transition and LED update using an oscilloscope. ● Confirm latency is ≤ 30 ms.
● The subsystem shall support detection of up to 10 simultaneous notes.	● Play 10 simultaneous keys on the keyboard. ● Confirm all notes are detected and displayed without dropped events.
User Interface Subsystem	
Requirements	**Verification**
● The User Interface Subsystem shall detect physical button presses and update system state accordingly without false triggering.	● Press each button at least 20 times in succession. ● Confirm that each press results in exactly one detected input event. ● Confirm that no additional unintended state transitions occur due to mechanical bounce. ● Observe system logs or debug output to verify correct detection.
● The subsystem shall allow the user to select among multiple MIDI files stored on the SD card.	● Store at least three MIDI files on the microSD card. ● Navigate through the user interface using the push buttons. ● Confirm that each selected file loads correctly and displays the appropriate note data.
● The subsystem shall allow the user to adjust playback speed and operational modes including normal mode, wait	● Select each playback mode individually. ● Confirm that system behavior changes according to mode definition.

mode, and single hand practice mode.	• Adjust playback speed and measure note timing to verify speed scaling is applied correctly.
• The subsystem shall provide visual feedback of menu navigation and system status on the RGB LED matrix.	• Navigate through menu options. • Confirm that displayed text or status indicators change appropriately. • Confirm that operational state changes are visibly reflected on the LED matrix.
Control Subsystem	
Requirements	**Verification**
• The subsystem is able to send correct DATA for the matrix.	• Check if the correct song is displayed on the matrix. • Confirm if the musical input is compared to the actual notes and produce appropriate assessment. • Confirm that note number, velocity, and timing data are correctly extracted.
• The subsystem is able to receive and read MIDI files from the microSD card	• Compare the data contained in the SD card files with those received from the SDIO pins of the STM32 to ensure the data received is correct
• The subsystem is able to receive user input from Push buttons to navigate the menus	• Confirm that button presses are reflected in the menus • Ensure that debouncing is correctly applied by observing that a single button push only results in a single event or change
Power Management Subsystem	
Requirements	**Verification**
• The subsystem shall provide a regulated 5 V supply capable of delivering up to 4 A to the LED matrix.	• Apply maximum LED load. • Measure supply voltage using a multimeter. • Confirm voltage remains within ±5% of 5 V.
• The subsystem shall accept GPIO from the user interface subsystem.	• Measure regulator output under normal operation. • Confirm output remains within ±5% of 3.3 V.

● The subsystem shall include overcurrent protection.	● Simulate overcurrent condition. ● Confirm the fuse interrupts excessive current flow.

Appendix B Cost Analysis Table

Table 9 Cost Analysis Table

Part Name	Manufacturer	Quantity	Cost (USD)	Link
STM32F446RET6 microcontroller	STMicroelectronics	1	8.15	Here
64x32 RGB LED Matrix (HUB75)	Adafruit	1	39.95	Here
74HCT245 transceiver	ON Semiconductor	3	2.07	Here
Micro SD Card Socket	Adafruit	1	1.95	Here
Micro SD Card	Solid Run STD	1	12	Here
Barrel Jack 5v	adafruit	1	0.95	Here
Screw Terminal	Amphenol Anytek	1	0.54	Here
AMS1117-3.3 low dropout regulator	UMW	1	0.29	Here
2 Pin Connector	TF Connectivity AMP Connectors	3	2.76	Here
DIN-5, 5 Pin Midi Jack	Adafruit	1	1.75	Here
6N138 Optocoupler	Lite-ON Inc	1	0.82	Here
Fuse	Bourns Inc.	1	0.64	Here
Electret Mic	Same Sky	1	0.79	Here

Amplifier	Microchip Technology	1	0.48	Here
5 pin connector	Tinkerspace	1	1.99	Here
02x08 connector	Sparkfun	1	0.95	Here
1k Ohms	Yageo	2	0.2	Here
220 Ohms	Yageo	1	0.1	Here
1N4148 Diode	Diodes Incorporates	1	0.1	Here
47k Ohms	Yageo	7	0.7	Here
Green LED	Rohm Semiconductors	1	0.51	Here
Red LED	Rohm Semiconductors	1	0.53	Here
0.1uF Capacitor	Yageo	8	0.8	Here
10uF Capacitor	Yageo	2	1.3	Here
10k ohms	Yageo	6	2.4	Here
10 ohms	Yageo	1	0.1	Here
Ceramic Resonator 8MHZ	YIC	1	0.14	Here
2.2u Capacitor	Yageo	2	0.22	Here
27pF Capacitor	Murata Electronics	2	0.2	Here

1uF Capacitor	TDK Corporation	3	0.36	Here
2.2k Ohms	Yageo	1	0.1	Here
100k Ohms	Yageo	1	0.11	Here
PMOS	Adafruit	1	2.95	Here
22uF capacitor	Murata Electronics	4	3.48	Here
470uF capacitor	Nichicon	1	0.8	Here
10nF capacitor	KEMET	1	0.13	Here
10uH Inductor	Abracon LLC	1	0.1	Here
550mV Schottky Diode	Diodes Incorporated	1	0.15	Here
200uF Capacitor	Chemi-Con	2	4.28	Here
3.24k Ohms	Stackpole Electronics Inc	1	0.1	Here
CAP CER 10UF 10V X7R 0805	Murata	4	1.40	Here
CAP TANT 22UF 20% 16V 1206	Kyocera	5	4.20	Here
CAP CER 1UF 16V X7R 0805	KEMET	5	1.20	Here
CAP CER 0.1UF 50V X7R 0805	KEMET	11	0.18	Here
CAP CER 2.2UF 50V X5R 0805	KEMET	3	0.75	Here
CAP CER 27PF 50V C0G/NP0 0805	KEMET	3	0.33	Here

CAP CER 10000PF 500V X7R 0805	YAGEO	2	0.36	Here
LED GREEN CLEAR 0805 SMD	VCC	2	1.66	Here
PTC RESET FUSE 8V 1.6A 1812	Littelfuse Inc.	3	1.68	Here
MOSFET N-CH 500V 350MA TO252	Microchip Technology	2	1.66	Here
CAP TANT 22UF 10% 6.3V 1206	Kyocera AVX	10	2.43	Here
DIODE ARR SCHOTT 40V 5A POWERDI5	Diodes Incorporated	1	1.79	Here
FIXED IND 10UH 7A 29.3 MOHM SMD	Bourns Inc	2	3.18	Here
RES 10 OHM 1% 1/8W 0805	Yageo	2	0.20	Here
RES 10K OHM 1% 1/8W 0805	Yageo	7	0.91	Here
RES 10K OHM 1% 1/8W 0805	Yageo	10	0.09	Here
RES 47K OHM 1% 1/8W 0805	Yageo	10	0.07	Here
RES SMD 2.2K OHM 0.1% 1/8W 0805	Yageo	3	0.69	Here
RES SMD 100K OHM 0.1% 1/8W 0805	Panasonic Industry	2	0.68	Here
RES 200 OHM 1% 1/8W 0805	YAGEO	10	0.09	Here
RES 1K OHM 1% 1/8W 0805	YAGEO	10	0.17	Here
RES 3.24K OHM 1% 1/8W 0805	YAGEO	10	0.08	Here
IC OPAMP GP 2 CIRCUIT 8DFN	Microchip Technology	1	0.48	Here
DIODE SCHOTTKY 30V 5A PMDTM	Rohm Semiconductor	2	1.32	Here
CONN RCPT FMALE DIN 5POS SOLDER	Same Sky	2	3.28	Here

1N4148 DIODE STANDARD 75V 150MA DO204AH	Good-Ark Semiconductor	2	0.20	Here

Appendix C Project Schedule

Table 10 Project Schedule Table

Time	Work	Person
02/23~03/01	Finalize design document and PCB layout. Submit first PCB order. Prepare for Design Review.	Everyone Nuwan (Finalize PCB layout)
03/02~03/08	Complete Design Review. Assemble breadboard prototype. Prepare for Breadboard Demo.	Everyone
03/09~3/15	Breadboard Demo. Begin PCB assembly once boards arrive.	Everyone(Demo)
03/16~03/22	Spring Break	Everyone
03/23~03/29	Continue PCB soldering. Implement SD card interface and MIDI parsing firmware. Begin subsystem testing.	Sarayu, Jay (soldering) Sarayu (Testing subsystem)
03/30~04/05	Integrate LED matrix driver with parsed MIDI data. Begin full system integration. Continue to solder	Nuwan(Integrate LED matrix) Sarayu, Jay(Soldering)
04/06~04/12	Progress Demo. Demonstrate integrated LED	Everyone(Demo)
04/13~04/19	Finish all the soldering. Complete system integration. Debug timing, accuracy, and stability issues. Conduct full verification testing.	Sarayu, Jay (Soldering) Nuwan(full verification testing)
04/20 ~ 04/26	Mock Demo and presentation practice. Finalize enclosure and polish user interface behavior.	Everyone
04/27 ~ 05/03	Final Demo and Final Presentation.	Everyone

05/06	Final paper submission, lab checkout, documentation completion.	Everyone