Dwight Look College of
ENGINEERING
TEXAS A&M UNIVERSITY

ECEN 403 Final Presentation
Team 28: Guitar Entertainment System

Rishabh Ruikar, Rawan Ibraheem, Monte Martin III

Problem Overview

- Problem statement: Current guitar amplification and sound modification systems lack the integration of modern technology and user-friendly control methods for those with limited experience.

Solution Proposal

- Solution proposal: Develop a high-tech guitar sound system with an amp, pedals, and a bluetooth app, to allow to the user to more seamlessly integrate and customize the sound system to their liking.

Bluetooth™

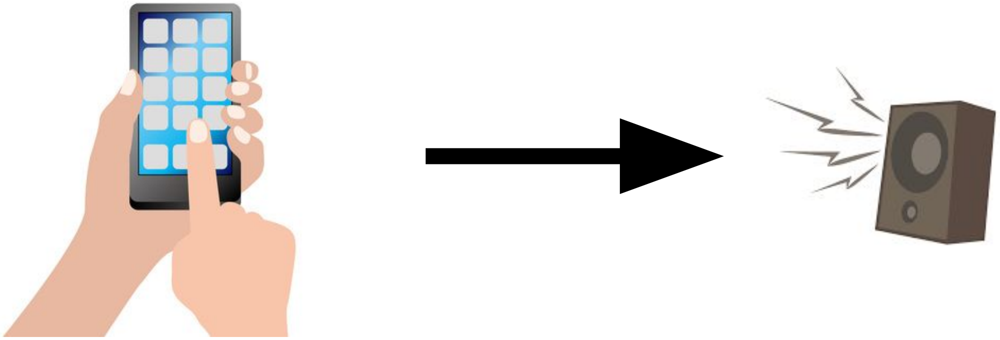


Diagram of subsystems and interface

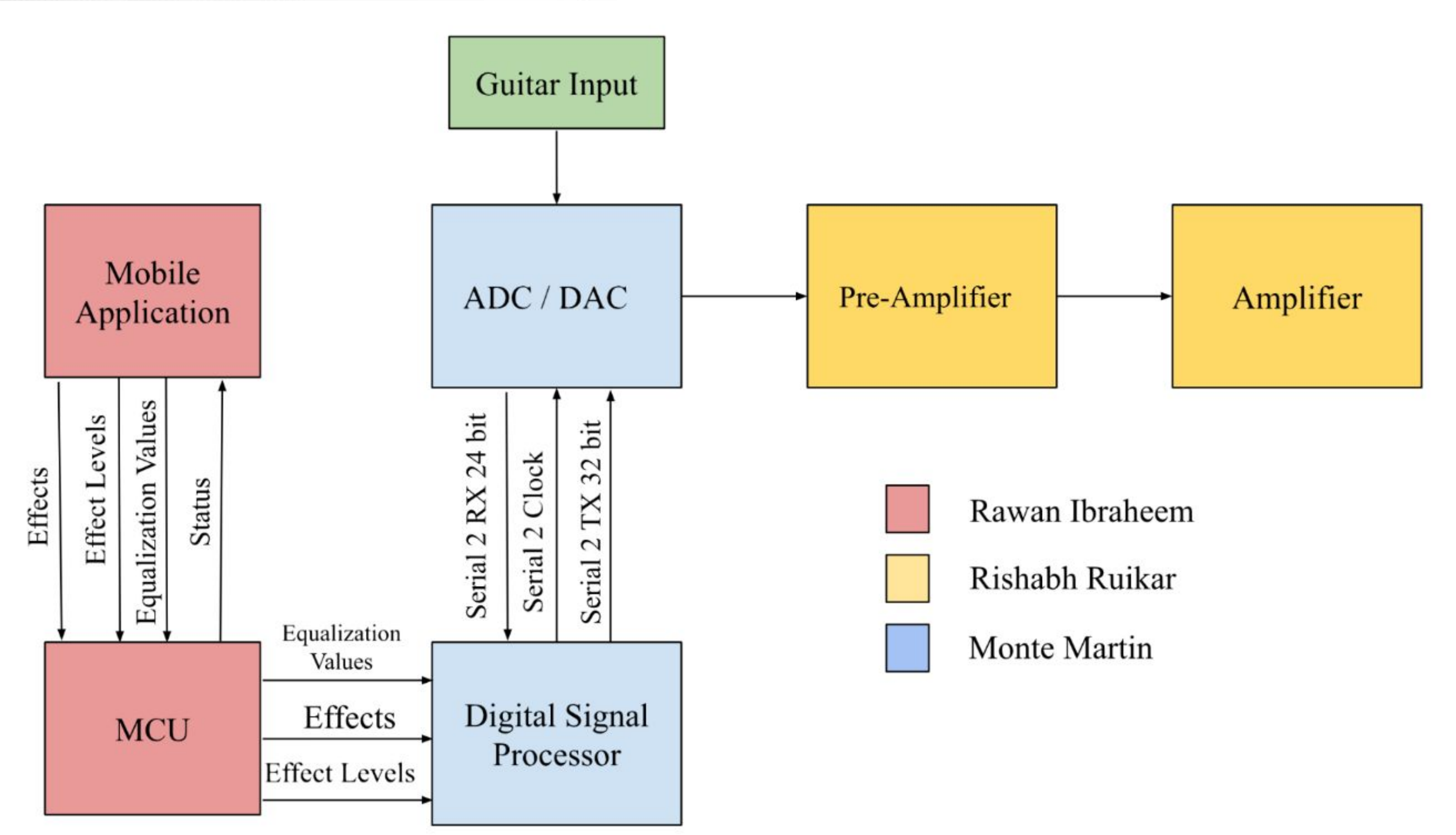
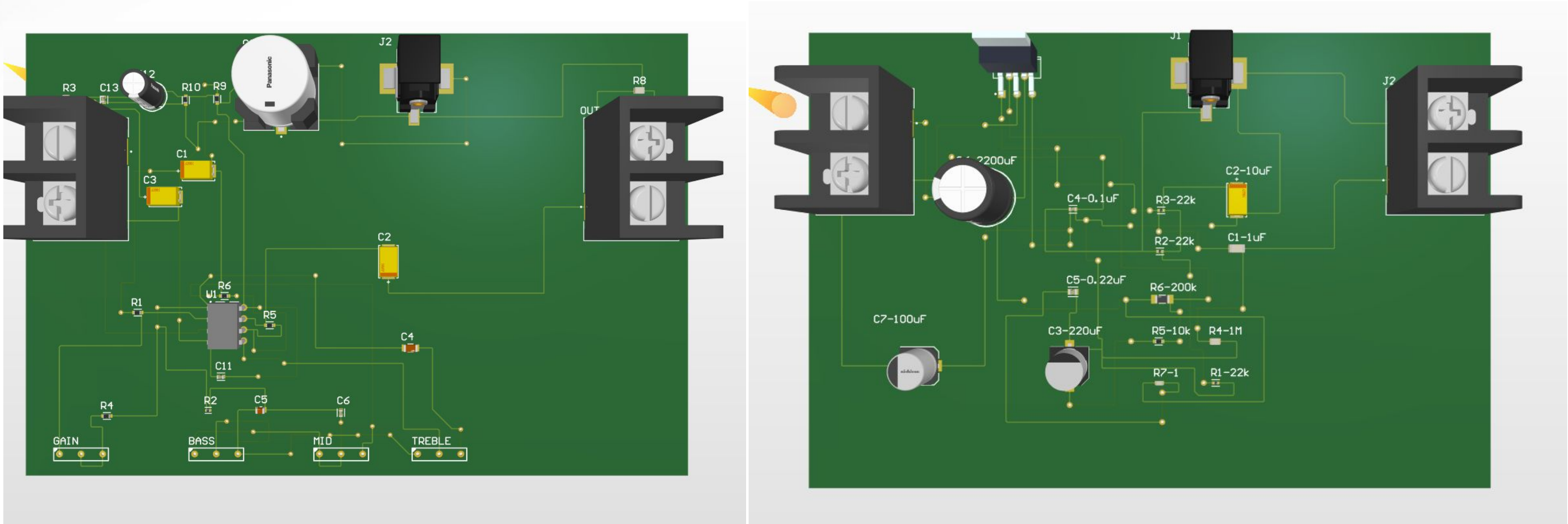


Figure 1: Diagram of subsystems in the Guitar Entertainment System.

Amplifier Subsystem

- Accomplishments
  - Built the circuit and PCB for the preamplifier and added potentiometers for equalization effects (Mid, Treble, Bass, and Gain)
    - Volume control will be done purely through the app
  - Built the circuit and PCB for the amplifier
    - Amplifier will output to a Jensen C8R 8” 8-ohm speaker

Amplifier Subsystem (Cont.)



Preamplifier

Amplifier

Figure 2: Printed circuit board designs for the preamplifier and amplifier.

Pedal System

- Accomplishments
  - Designed schematics for the ESP32, STM32, PCM4222, and PCM5142. Designed supplementary circuits necessary for proper functionality and operation
  - Placed all parts on the PCB and routed each individual chip on the board

Pedal System(Cont.)

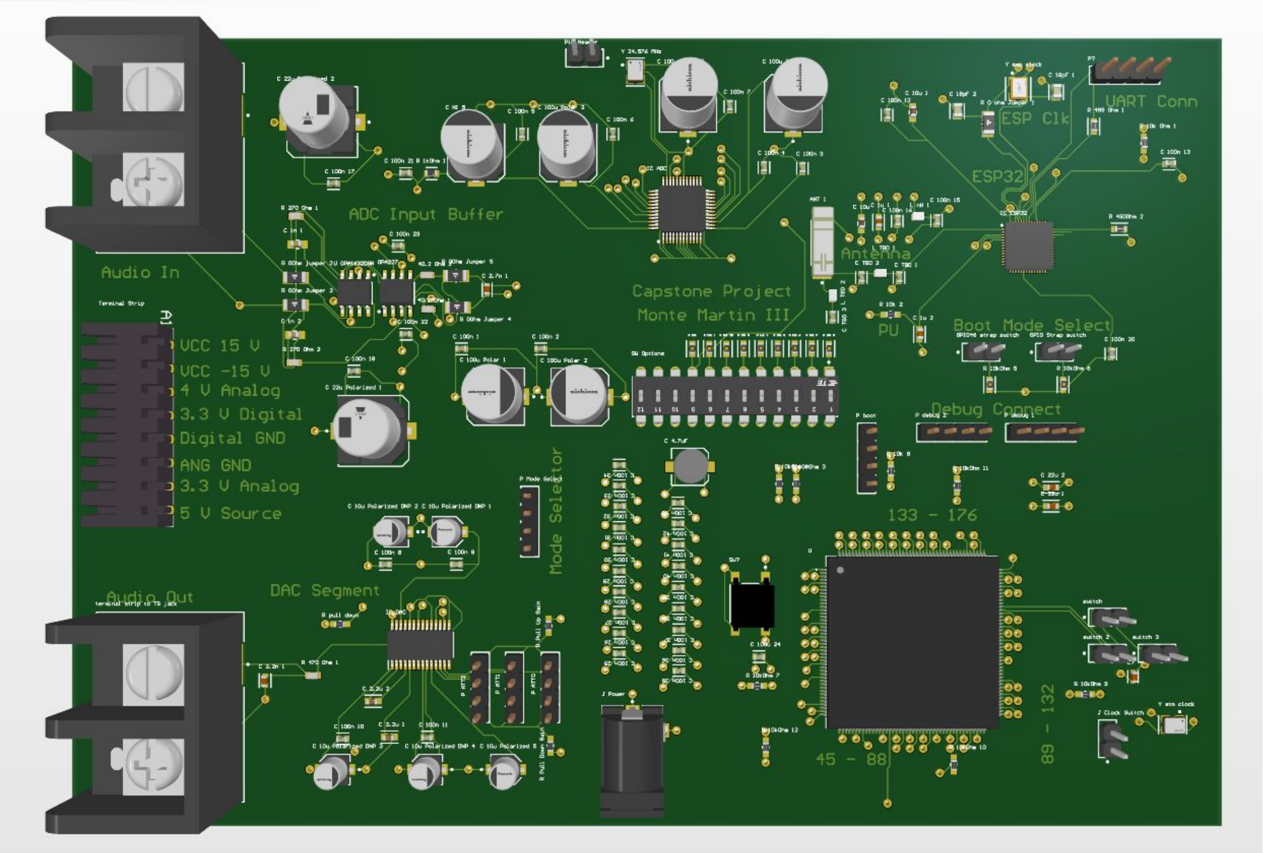


Figure 3: Pedal subsystem PCB 3D view.

Pedal System(Cont.)

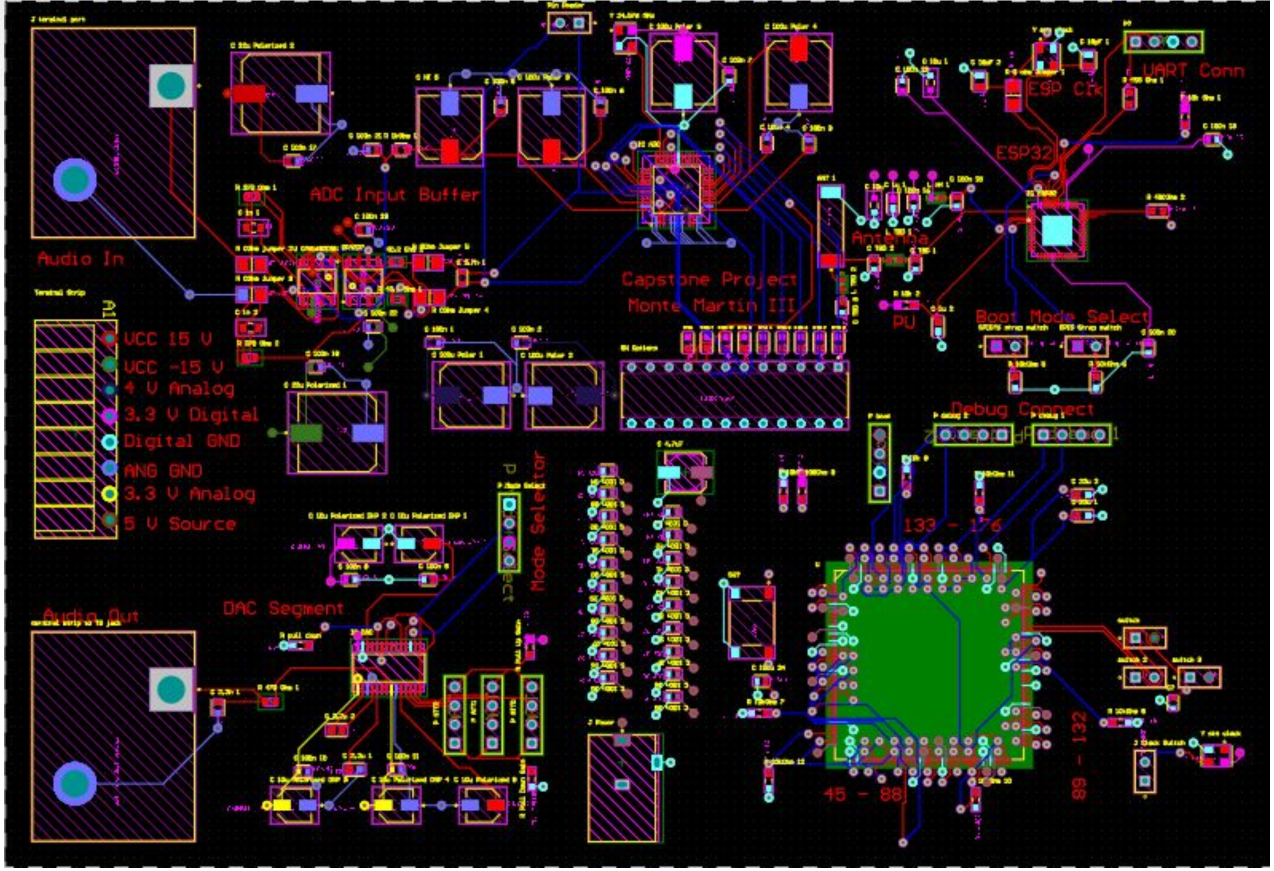


Figure 4: Pedal subsystem PCB 2D view.

Application and MCU Subsystem

- Accomplishments
  - Developed Android application that can communicate with MCU through Bluetooth Low Energy
  - Application can send signals to MCU to adjust volume, delay, reverb, chorus, distortion, and wah-wah effects.
  - Uses 149.4 MB of memory and a maximum of 18% CPU
  - Takes 2.88 seconds to connect from mobile to MCU and confirm connection in monitor
  - Takes 2.18 seconds to send and receive effect adjustment

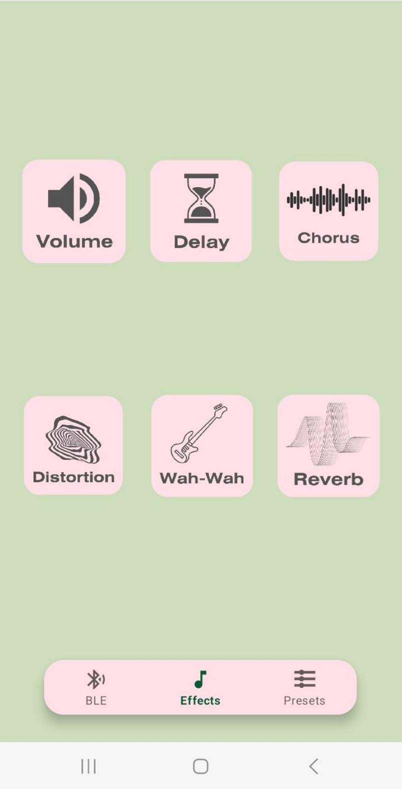
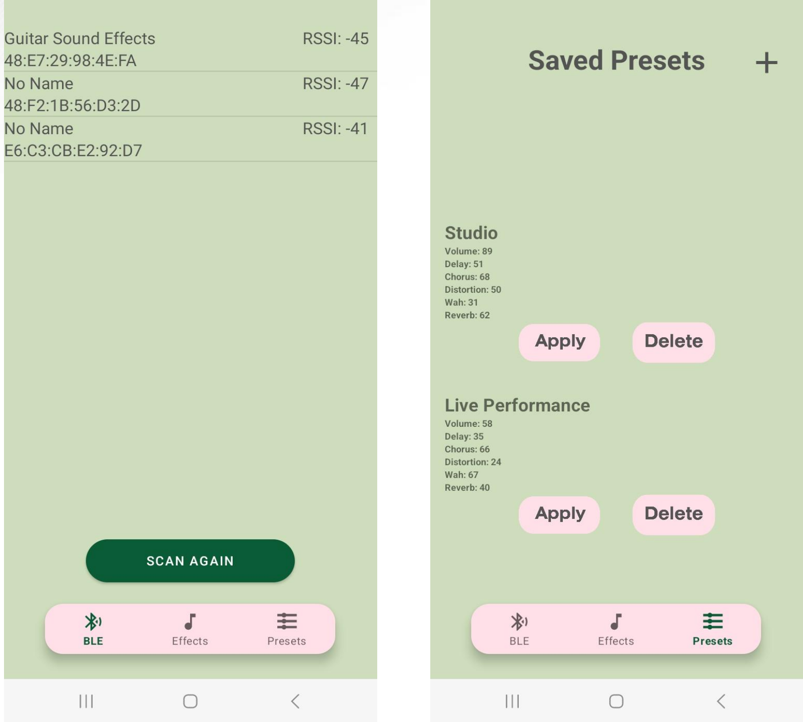


Figure 5: Application Effects screen.

Application and MCU (Cont.)



Guitar Sound Effects RSSI: -45
48:E7:29:98:4E:FA

No Name RSSI: -47
48:F2:1B:56:D3:2D

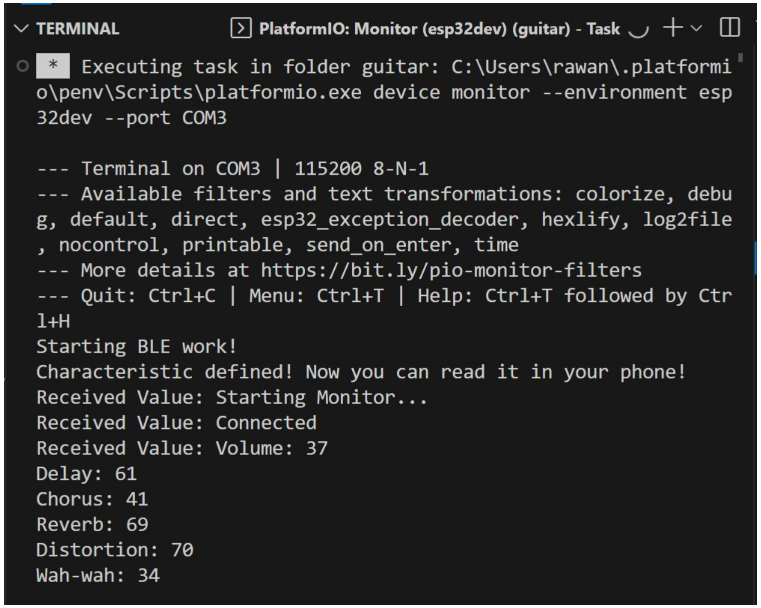
No Name RSSI: -41
E6:C3:CB:E2:92:D7

Saved Presets

Studio
Volume: 89
Delay: 51
Chorus: 68
Distortion: 50
Wah: 31
Reverb: 62

Live Performance
Volume: 58
Delay: 35
Chorus: 66
Distortion: 24
Wah: 67
Reverb: 40

Figure 6. BLE and Saved Presets screens on the Android Application.

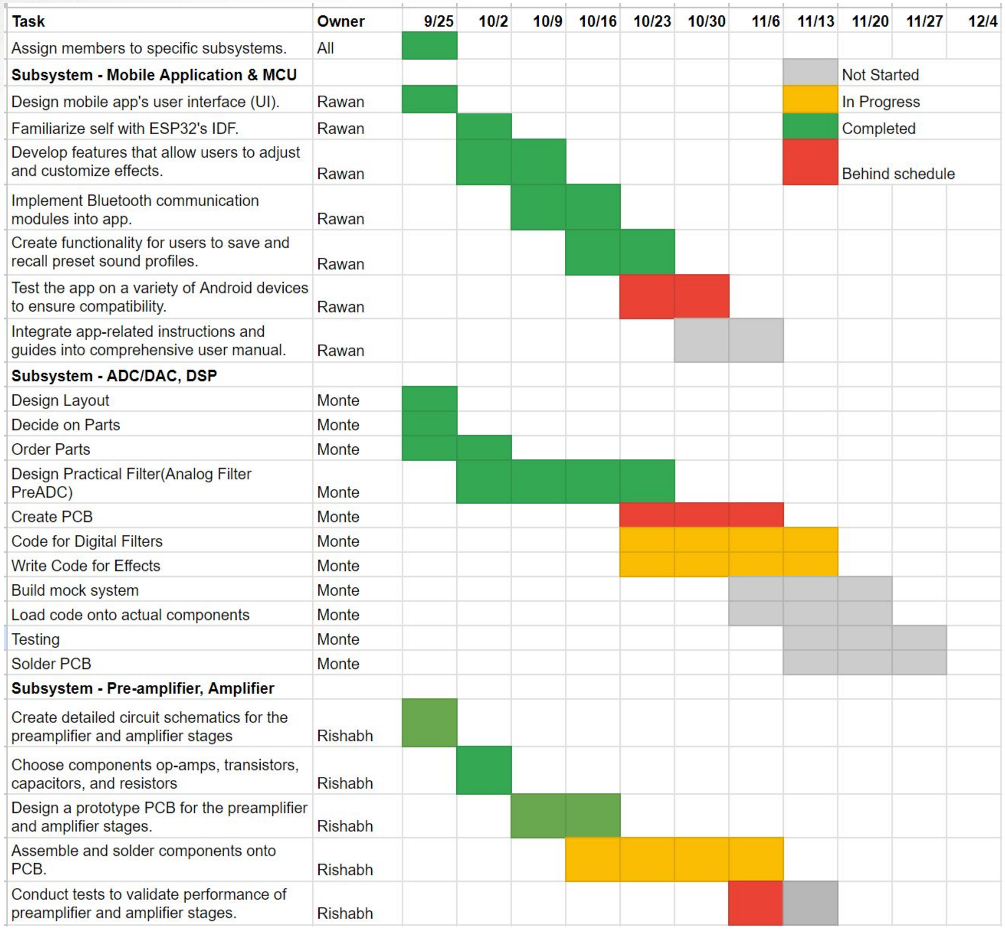


```
 *  Executing task in folder guitar: C:\Users\rawan\.platformio\penv\Scripts\platformio.exe device monitor --environment esp32dev --port COM3

--- Terminal on COM3 | 115200 8-N-1
--- Available filters and text transformations: colorize, debug, default, direct, esp32_exception_decoder, hexlify, log2file, nocontrol, printable, send_on_enter, time
--- More details at https://bit.ly/pio-monitor-filters
--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H
Starting BLE work!
Characteristic defined! Now you can read it in your phone!
Received Value: Starting Monitor...
Received Value: Connected
Received Value: Volume: 37
Delay: 61
Chorus: 41
Reverb: 69
Distortion: 70
Wah-wah: 34
```

Figure 7. PlatformIO Monitor displaying signal changes from the application.

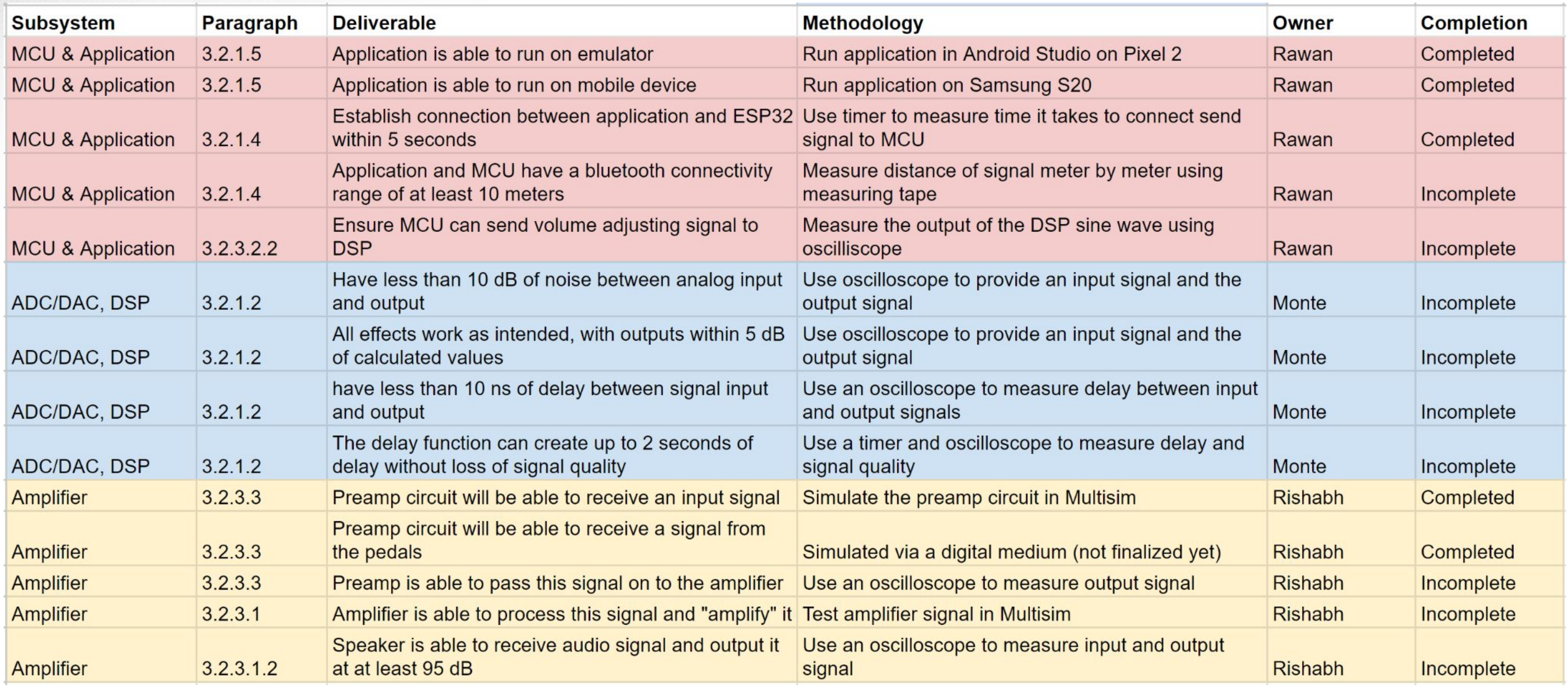
Execution Plan Status



Task	Owner	9/25	10/2	10/9	10/16	10/23	10/30	11/6	11/13	11/20	11/27	12/4
Assign members to specific subsystems.	All	🟩										
Subsystem - Mobile Application & MCU									⬜	Not Started		
Design mobile app's user interface (UI).	Rawan	🟩							🟨	In Progress		
Familiarize self with ESP32's IDF.	Rawan		🟩						🟩	Completed		
Develop features that allow users to adjust and customize effects.	Rawan		🟩	🟩					🟥	Behind schedule		
Implement Bluetooth communication modules into app.	Rawan			🟩	🟩							
Create functionality for users to save and recall preset sound profiles.	Rawan				🟩	🟩						
Test the app on a variety of Android devices to ensure compatibility.	Rawan					🟥	🟥					
Integrate app-related instructions and guides into comprehensive user manual.	Rawan						⬜	⬜				
Subsystem - ADC/DAC, DSP												
Design Layout	Monte	🟩										
Decide on Parts	Monte	🟩										
Order Parts	Monte	🟩	🟩									
Design Practical Filter(Analog Filter PreADC)	Monte		🟩	🟩	🟩	🟩						
Create PCB	Monte					🟥	🟥	🟥				
Code for Digital Filters	Monte					🟨	🟨	🟨	🟨			
Write Code for Effects	Monte					🟨	🟨	🟨	🟨			
Build mock system	Monte							⬜	⬜	⬜		
Load code onto actual components	Monte							⬜	⬜	⬜		
Testing	Monte								⬜	⬜	⬜	
Solder PCB	Monte								⬜	⬜	⬜	
Subsystem - Pre-amplifier, Amplifier												
Create detailed circuit schematics for the preamplifier and amplifier stages	Rishabh	🟩										
Choose components op-amps, transistors, capacitors, and resistors	Rishabh		🟩									
Design a prototype PCB for the preamplifier and amplifier stages.	Rishabh			🟩	🟩							
Assemble and solder components onto PCB.	Rishabh				🟨	🟨	🟨	🟨				
Conduct tests to validate performance of preamplifier and amplifier stages.	Rishabh							🟥	⬜			

Figure 8. Execution plan with updated completion status.

Validation Plan Status



Subsystem	Paragraph	Deliverable	Methodology	Owner	Completion
MCU & Application	3.2.1.5	Application is able to run on emulator	Run application in Android Studio on Pixel 2	Rawan	Completed
MCU & Application	3.2.1.5	Application is able to run on mobile device	Run application on Samsung S20	Rawan	Completed
MCU & Application	3.2.1.4	Establish connection between application and ESP32 within 5 seconds	Use timer to measure time it takes to connect send signal to MCU	Rawan	Completed
MCU & Application	3.2.1.4	Application and MCU have a bluetooth connectivity range of at least 10 meters	Measure distance of signal meter by meter using measuring tape	Rawan	Incomplete
MCU & Application	3.2.3.2.2	Ensure MCU can send volume adjusting signal to DSP	Measure the output of the DSP sine wave using oscilliscope	Rawan	Incomplete
ADC/DAC, DSP	3.2.1.2	Have less than 10 dB of noise between analog input and output	Use oscilloscope to provide an input signal and the output signal	Monte	Incomplete
ADC/DAC, DSP	3.2.1.2	All effects work as intended, with outputs within 5 dB of calculated values	Use oscilloscope to provide an input signal and the output signal	Monte	Incomplete
ADC/DAC, DSP	3.2.1.2	have less than 10 ns of delay between signal input and output	Use an oscilloscope to measure delay between input and output signals	Monte	Incomplete
ADC/DAC, DSP	3.2.1.2	The delay function can create up to 2 seconds of delay without loss of signal quality	Use a timer and oscilloscope to measure delay and signal quality	Monte	Incomplete
Amplifier	3.2.3.3	Preamp circuit will be able to receive an input signal	Simulate the preamp circuit in Multisim	Rishabh	Completed
Amplifier	3.2.3.3	Preamp circuit will be able to receive a signal from the pedals	Simulated via a digital medium (not finalized yet)	Rishabh	Completed
Amplifier	3.2.3.3	Preamp is able to pass this signal on to the amplifier	Use an oscilloscope to measure output signal	Rishabh	Incomplete
Amplifier	3.2.3.1	Amplifier is able to process this signal and "amplify" it	Test amplifier signal in Multisim	Rishabh	Incomplete
Amplifier	3.2.3.1.2	Speaker is able to receive audio signal and output it at at least 95 dB	Use an oscilloscope to measure input and output signal	Rishabh	Incomplete

Figure 9. Validation plan with updated completion status.

Remaining Tasks (Amplifier)

- Amplifier + Preamplifier
  - Soldering components onto boards
  - Conduct initial testing (making sure signals are able to pass through TS jacks)
  - Connect speaker to amplifier and see if a signal is received

Remaining Tasks (Pedal System)

- PCB
  - Make power planes and polygons for the PCB
  - Do final checks for the PCB and order it by the 6th/7th
  - Order components and solder them
- STM32
  - Program the microcontroller while waiting for the PCB fabrication, make sure that it can download from the JTAG connection

Remaining Tasks (Android App)

- Refine the user interface, ensuring the proper alignment and centering of all interface elements
- Address application crashes that occur in the absence of Bluetooth connection
- Determine the maximum communication range for establishing Bluetooth connections and transmitting signals

Thank you!
Any questions?