5-2017

Project Pradio

Trigg T. La Tour

Project Pradio

An Honors Thesis

Daniel Trigg La Tour

University of Arkansas

Department of Computer Science and Computer Engineering

Spring 2017

Abstract

This paper examines the design and manufacturing of a device that allows two or more users to share a wireless audio stream. Effectively, this allows a group of people to listen to the same audio in a synchronized manner. The product was unable to be completed in the allotted time. Regardless, significant progress was made and valuable insight into the circuit board design process was gained.

Nomenclature

1. *A2DP:* Advanced Audio Distribution Profile. A Bluetooth profile used for streaming audio.
2. *Audio Source:* A device capable of streaming audio via a standard interface, such as Bluetooth A2DP or a 3.5mm headphone connection. Examples of *audio sources* include smartphones, tablets, MP3 players, laptop computers, etc.
3. *Host User:* The user who is sharing an audio stream from an *audio source*.
4. *Sink User:* A user who is listening to an audio stream being broadcast by a *host user*.
5. *Audio Player:* A device used to convert audio signals into audible signals. E.g. a speaker, headphones, etc.
6. *Controlling Device:* A device used to configure the proposed device and modify the operating parameters. E.g. used pair the device with an *audio player*, change the device name, etc. Note that some *audio sources* may also be used as *controlling devices*. Examples of *controlling devices* include smartphones, tablets, laptop computers, etc. The proposed device may be operated without the presence of a *controlling device*, but some operational parameters will be off-limits.

I. Introduction

Described in this thesis is the creation of a device from idea to reality through the project titled *Pradio*. The purpose of the device is as follows: to allow a user to stream the contents of an *audio source* to additional local users—in other words, to enable a person to privately share what they're listening to with a select group of other people. The audio stream is synchronized across all devices so that all users hear the same audio at the same time. At first glance, there is another device that seems to accomplish this same task using sound waves: namely, a speaker. However, one of the defining features of the device described here is that it allows the user to *privately* share the contents of an audio stream with others. A group of people can listen to audio in a public setting without disturbing anyone else in the area. This goal can be accomplished using headphones connected to the device that allow the transmission of audio to the user. Additionally, this method allows the quality of the audio stream to be far superior to that of a speaker, particularly in certain environments. A device with these features has many unique applications—several of which are described here. The primary application area for this device is the outdoor sports sector: outdoor running, snow sports, biking, etc. Many people, while perfectly capable of pursuing these sports as individuals, choose to participate as a part of a group or team. The experiences of individuals within participating groups can be greatly enhanced by the addition of music during active participation. Many recreational athletes already actively use some form of personal music playback for just this reason. If this use of audio as an experience-enhancer is expanded to the team or group level, additional enhancing qualities can be added to the user's experience. When a running team reaches a daunting hill, they can experience a second wind together as the perfect song plays to spur them on towards the top. Two friends out for a jog can enjoy the fun of listening to a song together as they go, rather than being locked in their own worlds, separated by two sets of headphones playing different songs. A group of friends snowboarding on the slopes of the Colorado Rockies can enjoy the same up-beat tempo as they charge through the ramps and half-pipes of a terrain park. They'll be able to watch as their friends tackle these features perfectly in step with the beat of the music they're listening to. In addition to being used as a unifying tool, it can also be used in areas where a group wishes to be isolated from the environment around them. Working out in a public gym has become almost commonplace in western culture today, and there are many benefits: access to specialized, often expensive, equipment; encouragement from seeing others reach their goals; training staff; etc. But, there are some aspects that one may wish to change. Most gyms will have some sort of music playing in the background. As is often the case, it is difficult to find music that every person in the gym will enjoy. Many people remedy this by simply using headphones to listen to their own carefully curated music collections; however, when working out as a pair, the proposed device would eliminate this annoying situation and allow a group to listen to their own music without giving up the personal interactions that occur amongst sweaty gym buddies. Finally, adding to the seemingly contradictory applications of this device, it serves as a conduit for discovery and a uniquely human connection. A device such as the one described here would allow individuals who are drinking coffee in the same cafe, working in the same office, or hanging out in the same space but are still strangers to one another to connect with each other using music. With this device, a user could tune into a channel that another user is hosting and, with their permission, listen to the same thing that they're listening to. Experiencing a small piece of life through their eyes, or ears. This set of diverse applications and broad potential appeal make Project Pradio an ideal candidate for a product that could carve out a new niche in the electronics industry. A niche based on unification, personal freedom, and discovery.

II. High-Level Functionality

From the list of this device's use cases described above, a set of necessary functions can be derived. As stated previously, the primary purpose of the device is to allow users to share an audio stream with the other users around them. To accomplish this goal, an *audio source* is needed. The audio stream must originate and be sourced from a separate device, as the device proposed here will merely act as a link. Thus, the proposed device must provide an interface to allow a user to connect an *audio source*. Once an audio source is connected, an interface to allow the *host user* to listen to the stream is needed. The user playing the audio will, in all likelihood, also want to listen to the audio stream. So far, we have an *audio source* creating an audio stream and sending it to the device. This audio stream then plays for the *host user* through some method and is broadcast out to other potential listeners. Next, some additional user may want to join the *host user* and listen to the audio stream. When this new listener joins the audio stream through some wireless pairing procedure, the same interface used by the *host user* to listen to the audio can be utilized to on the *sink user*'s device to play the audio for them to hear. Additional users should be able to join and listen in an identical manner. Detailed requirements for the implementation of this functionality are discussed in the sections below.

III. History and Background

In the past, there have been several attempts to create devices or software to accomplish essentially the same function. In this paper, we'll look at three such examples. The first example is that of headphone-tycoon corporation Beats by Dre. Several Beats by Dre headphones implement a feature that Beats calls “Music sharing [1].” This feature allows two pairs of Beats headphones to be connected by wire to let the audio flow from the source to both sets of headphones. While this system no doubt provides excellent audio quality and minimal latency, there are obvious disadvantages. The tethering of the two sets of headphones severely limits the use cases in which this feature can be used. Outdoor sports or any activity that requires even a minimal amount of bodily movement are out of the question. The proposed device, while operating in a similar fashion, will eliminate the cable connecting the two users and allow for a broad range of use case activities, including activates that involve high amounts of physical movement. The second example follows a similar path forwards from the Beats solution. The headphones developed by Wearhaus, known as the Wearhaus Arc, also seek to eliminate the physical link between the users [2]. The Arc headphones allow users to stream music from an audio device via Bluetooth and then broadcast this audio stream out to other Arc users wirelessly. Again, this solution boasts premium audio performance and synchronized playback for multiple users; however, a more optimal and affordable solution is available. The Wearhaus Arc headphones cost $200. While this price falls somewhat on the low side of the price range for high-end headphones, it can be a major deterrent for a user who wants the music sharing functionality. Additionally, many audiophiles already own a set of expensive, high-end headphones, and this additional investment seems to put their previous investment to waste, especially if the first pair of headphones boast superior audio performance. The creators of the Wearhaus Arc have widened the use case scenarios for their device by eliminating the physical link, but they have limited themselves by binding the audio sharing functionality to the headphones. Wearhaus currently only offers one model of the Arc and it is not suited for physically-active or outdoor use. This fact, coupled with the price barrier makes the Arc headphones a sub-optimal solution for a user who would like to share the audio they're listening to. By separating the audio sharing functionality from the headphone, a broader market may be approached at a lower price point, making the proposed device a more attractive and affordable option. Finally, a software solution is offered by an application named MyStream for the iOS platform [3]. The implementation of this application is quite advanced, allowing users to connect via either a Wi-Fi connection, an ad-hoc network, or even over a Bluetooth net. It supports as many as twenty users over Wi-Fi and up to six users via Bluetooth. The Bluetooth connection can be unreliable, but the presence of the ad-hoc network still allows the device to operate anywhere, free from the bonds of internet access [4]. Impressive as the groundwork development for this application may be, it is not without fault. The primary negative aspects of this solution are that it requires all users to carry their phones with them while using the audio sharing feature and the limited application platform. Requiring a user to carry a phone is becoming less and less of an issue in modern culture, as more individuals voluntarily carry their phones with them at all times; however, there are still several use cases in which a user may not want to carry a large device such as a phone. Examples include running, working out, etc.. In these instances, this application falls short of providing the optimal solution. Additionally, the application is currently only offered on the iOS platform. Updates to the application have been non-existent for the past five years, and there are no indications that development is currently underway for any additional platforms [5]. This severely limits both the target market as well as the application use cases. Because this solution is software-based, it is inherently platform dependent. A new solution must be developed for each platform the developer wishes to support. An optimal, scalable solution would be platform independent, making use of the audio interfaces that already exist in many of today's *audio source* systems. That is exactly what the proposed device is designed to accomplish. One advantage of the software solution is affordability. The MyStream app is currently available on the Apple App Store for free, using ads and other sideline sources for generating revenue. While this application may fill the need of some potential users, the device proposed in this paper remains the optimal solution for the user who desires portability and freedom in his or her audio sharing solution.

IV. Design Requirements and Constraints

To meet the needs of the aforementioned use cases as well as provide the optimal audio sharing solution, there are several constraints that design must adhere to.

A. Size

Size is arguably the most pressing constraint for this design. Because of the broad range of application use cases, many of them being in the active sports sector, the size of the device must be such that it does not inhibit the user in participating in the activity in which they are involved. For the purposes of this design, the final product dimensions should be no more than $(25.4mm \times 50.8\text{mm} \times 7.62\text{mm})$.

B. Battery Life

The battery life for this device is determined in terms of number of typical use scenarios. Per the use case scenarios mentioned in the introduction portion of this document, the typical use scenario can be assumed to be between 2 and 3 hours. For this project, a minimum goal of two typical use cycles will be set. This places the overall battery life between 4 and 6 hours. The hardware and software must be optimized with a goal towards power efficiency to meet this requirement.

C. Connectivity

As was also mentioned in the introduction, wireless connectivity for the user is essential for many of the perceived use case scenarios. This wireless connectivity must act in several directions as shown in Figure 1 below.

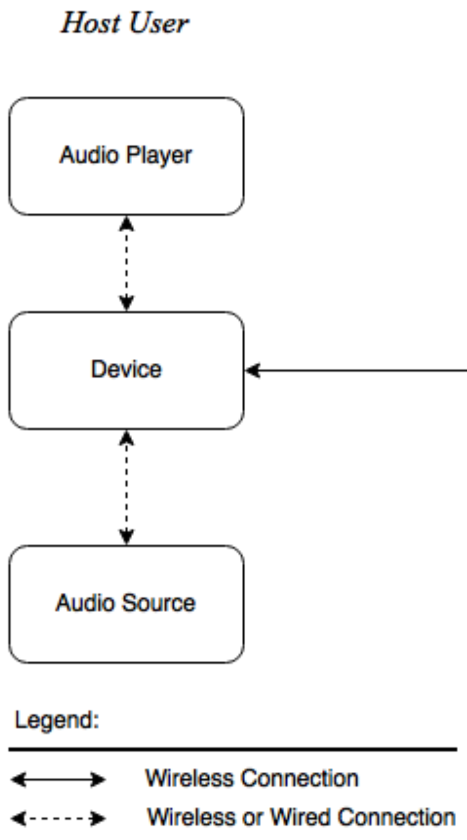


Figure 1. Device Connections Required

Note that there may be more than one *sink user* that the *host user* must be able to support. Also, note that many of the connections shown above indicate that the link may be wireless or wired. For optimal mobility, every link would be wireless. However, due to power constraints and limited hardware resources, an optimal compromise between mobility and power should be used. The link between the *audio source* and the device is used to transmit the audio stream to the device. The device must support both wired and wireless interfaces. This allows the user to determine what type of power and mobility compromise he or she desires. The link from the device to the *audio player* is used to transmit the audio stream to a device that plays the audio for the user. It also can be wired or wireless, again allowing the user to choose the desired mobility and power efficiency tradeoff. The link from device to device is used to share the audio stream to accomplish the primary function of the device. This link must be wireless. The link with the *controlling device* on the *sink user* side of Figure 1 is used to set various operating parameters of the device. This link is defined to be wireless only, but because of the low amount of data to be sent across this link, it can use a low-power wireless protocol and thus have minimal impact on the battery life of the device.

D. User Interface

A user interface is an essential element of almost every consumer electronics device. A user must be able to determine what state the system is in, determine what its status is, and alter the operating parameters. In the case of the proposed device, this interface will be used to allow the user to put the device in pairing mode, as well as tell the user which peer device the user's device is connected to, what the computed remaining battery capacity is, and many other parameters.

V. System Design

The system design consists of two primary parts and one secondary part. The two primary parts are the device hardware and firmware, while the secondary part is the *audio source* and *controlling device* application software. The scope of this project has been limited to focus on the two primary components. As such the development of the *audio source* and *controlling device* application software is not discussed in this paper.

A. Hardware Design

The hardware design process for this project was broken into three main steps detailed below:

1. Block diagram development and part collection
2. Schematic design
3. Printed circuit board (PCB) layout

During the block diagram and part collection phase, the input output (IO) requirements were drawn up for the central controller and the wireless link protocols and hardware requirements were determined. The output of this phase was a high-level block diagram and associated parts list for the primary system components. The block diagram is shown below in Figure 2.

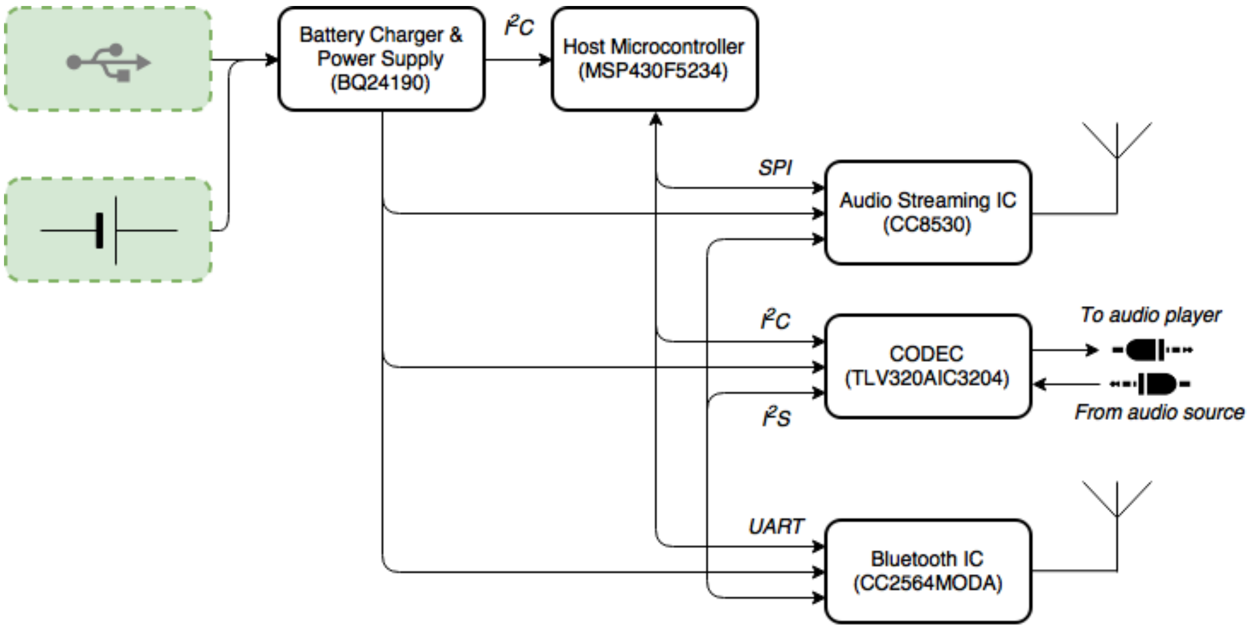

Battery Charger & Power Supply (BQ24190)
I²C
Host Microcontroller (MSP430F5234)
SPI
Audio Streaming IC (CC8530)
I²C
CODEC (TLV320AIC3204)
I²S
To audio player
From audio source
UART
Bluetooth IC (CC2564MODA)

Figure 2. High-Level Hardware System Block Diagram

As shown in Figure 2, the system is powered by a Universal Serial Bus (USB) connection. Power is stored in a battery for use when the system does not have a USB connection. The selected battery charger provides the host controller with an Inter Integrated Circuit (I^2C) serial interface that allows the processor to poll the charger for charging status, and a battery fuel gauge integrated circuit (IC) allows the processor to determine the predicted remaining battery capacity. This battery charger was selected because it is designed to charge a single-cell, Lithium-ion battery and provides a well-documented interface for setting the charge parameters. A single-cell battery was chosen to reduce system complexity and decrease the difference in stored voltage and system voltage with a goal of maximizing battery life.

The host processor oversees the configuration of hardware peripherals, the storage of system operating parameters, the facilitation of the user interface, and the controlling of dataflow throughout the system. The 16-bit MSP430 microcontroller was chosen because of its low-power design and a readily available Bluetooth interface provided by Texas Instruments for use with the CC2564MODA Bluetooth controller. The controller provides generous IO and serial interface capabilities coupled with 128 KB of flash memory and 8 KB of Random Access Memory (RAM) [6]. The micro controller will also be responsible for implementing the device-side user interface. For the purposes of the first design iteration, the user interface will consist of three simple push-buttons and three indicator light emitting diodes (LEDs).

The CC2564MODA Bluetooth chip was added to this design in order to provide Bluetooth connectivity from the device to *audio sources*, *controlling devices*, and Bluetooth-enabled *audio players*. This chip was chosen specifically because it has advanced capabilities in audio streaming using Bluetooth. The CC2564MODA supports an audio streaming protocol named A3DP, which means Assisted A2DP or Assisted Advanced Audio Distribution Protocol.

The A3DP protocol moves the audio processing logic from the host controller to the Bluetooth controller. This offloads much of the work typically required of the host controller in audio-centric systems and frees it up to handle other system tasks. The Bluetooth firmware stack included with this module enables the module to join multiple Bluetooth piconets simultaneously, one as the master and two as the slave [7]. Additionally, it allows for up to 10 simultaneous Bluetooth Low Energy (BLE) connections. For more information on piconets and the various Bluetooth profiles, see the Bluetooth specification [8]. Using the resources of this powerful device should allow the design to fulfill all the wireless link requirements for both the *host user* and *sink user* scenarios as shown in Figure 1 above. The high-speed, high-power piconets can be used to transmit the audio streams from the *audio source* in the slave role and to the *audio player* in the master role. A single BLE connection is used to connect a *controlling device* and transmit any necessary operational parameters. Any of these connections can be disabled if the corresponding link is replaced with a wired connection in order to conserve energy.

The CC8530 was chosen as the audio streaming solution in order to solve the problem of streaming stereo, digital audio in a one-to-many network topography. There is no standard Bluetooth protocol for accomplishing this at this time, so an additional hardware interface is required. The CC8530 uses a star network topology to stream stereo, 48 KHz audio feeds with support for up to four slave devices [9]. This will allow up to five devices to listen to a single, high-quality audio stream. The IC provides a Serial Peripheral Interface (SPI) that is used by the host controller to configure the network and audio stream parameters. Figure 3 below shows the overall wireless topology of the device.

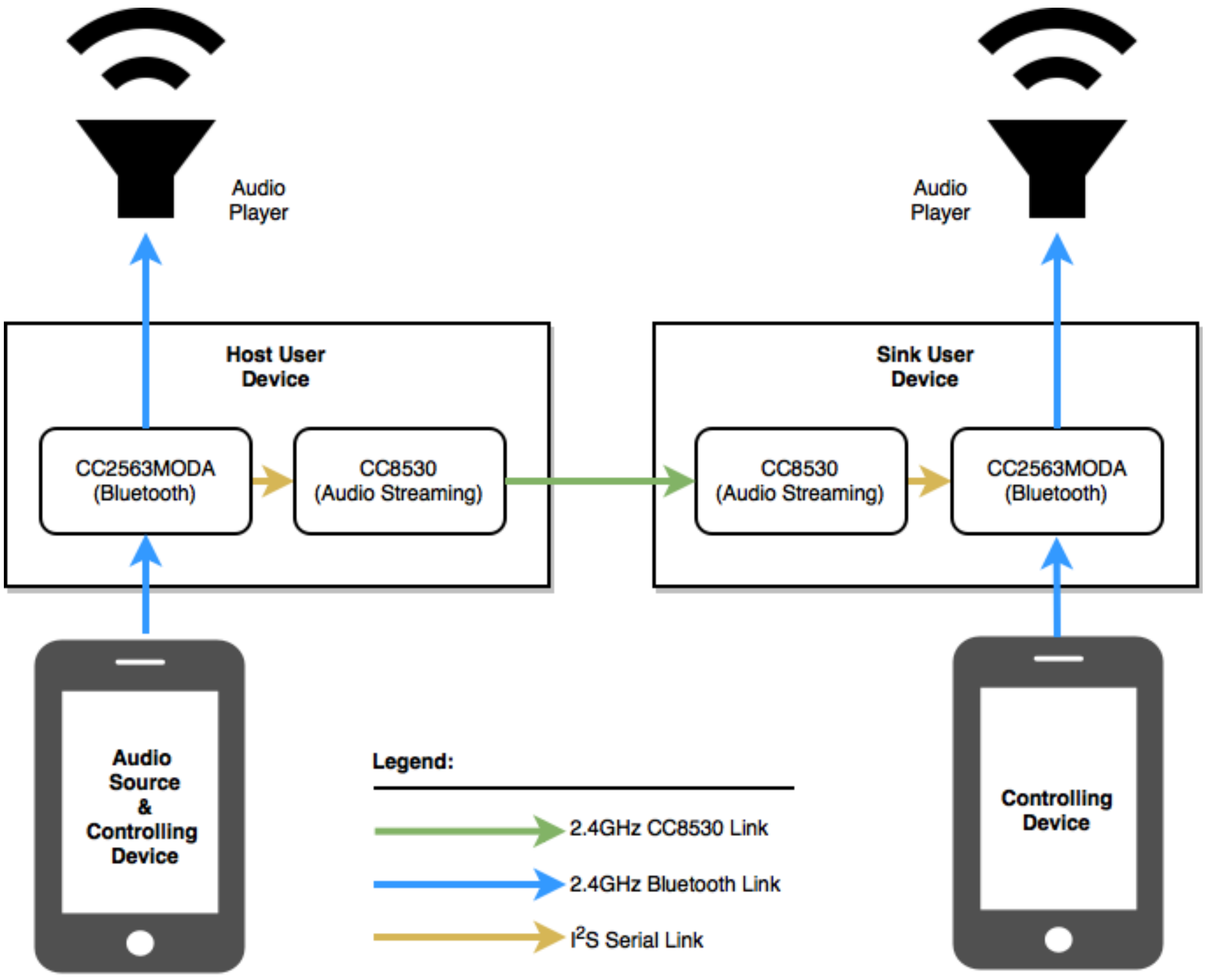


Figure 3. Wireless Network Topology

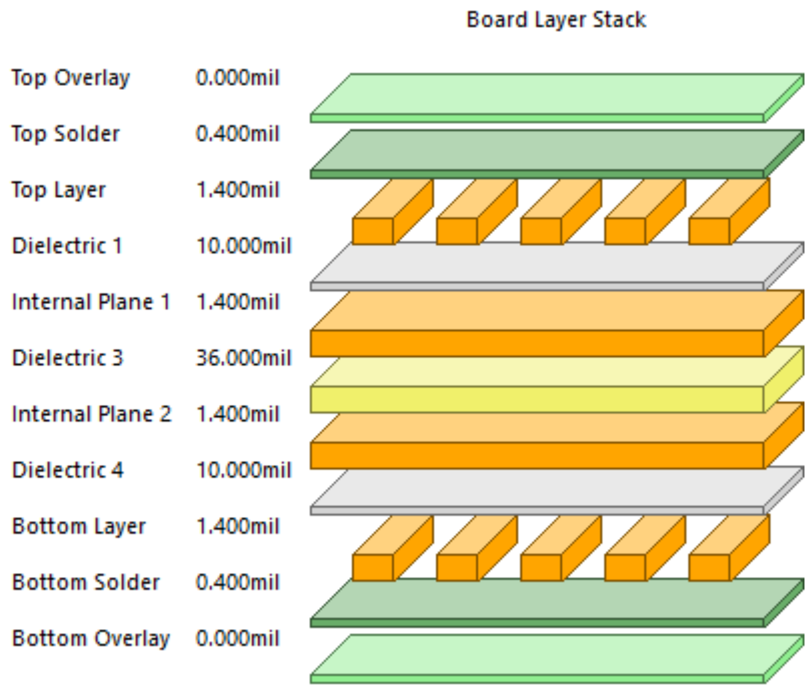
In the case that the user is the *host user*, the Bluetooth connection is used to receive the audio stream from the *audio source*. Additionally, the Bluetooth connection can be used to send and receive control data from a *controlling device* as well as to transmit audio to the *audio player*. In parallel, the CC8530 receives and transmits audio from other devices in the audio network. It is worth noting that both the proprietary CC8530 wireless protocol as well as the Bluetooth protocol stack are operating in the 2.4GHz bandwidth. While this may lead to some radio interference, both components implement frequency as well as time transmission and reception hopping. This should help keep interference to a minimum.

The audio Coder Decoder (CODEC) is used to convert analog audio signals coming from an *audio source* into digital signals that can be broadcast over the network. Conversely, it also converts digital audio data being received across the network into analog audio signals that can be sent to a wired *audio player*. The TLV320AIC3204 was chosen because it is a combined CODEC solution that is packaged with internal low-noise analog power supplies, a stereo class-D amplifier, and a very flexible clock and phase-locked loop (PLL) circuit [10]. The low-noise power supplies ensure that the audible audio signals are clean and crisp, while the class-D amplifier allows the device to drive analog external *audio players* without additional components.

Utilizing the hardware discussed in this section allows the device to operate in an efficient and functional manner. The essential step of high-level architecture complete, the detailed schematic design was completed next.

In the schematic design phase, the logical connection of IC devices and their supporting components are drawn out in a manner that is easy to read and that clearly and concisely conveys how the devices are to be physically connected once they are placed on the circuit board. In general, it consists of a constant cyclical movement from datasheet to design to datasheet, etc. to gather the information required to support the system components. Each completed schematic underwent multiple design iterations and optimizations. Because the details of this process are irrelevant to the purpose of this paper and rather tedious to discuss, they will not be covered here.

The circuit board layout process is the process of converting the logical connections from the schematic into physical connections on the printed circuit board (PCB). During this process, rather than trying to connect components in a clear manner as in schematic design, the components are laid out in a way that minimizes the space required on the circuit board and maximizes signal integrity and ease of manufacturing. The PCB design process begins with the selection of the PCB stack, that is, which materials are used to build the circuit board. This design uses a four-layered PCB. This means that there are four conductive layers and three insulating layers. The electronic connections are made on the conductive layers with circuit board vias connecting the signals across various layers. The layer stack up for this design is shown below in Figure 4.



Top Overlay	0.000mil
Top Solder	0.400mil
Top Layer	1.400mil
Dielectric 1	10.000mil
Internal Plane 1	1.400mil
Dielectric 3	36.000mil
Internal Plane 2	1.400mil
Dielectric 4	10.000mil
Bottom Layer	1.400mil
Bottom Solder	0.400mil
Bottom Overlay	0.000mil

Figure 4. PCB Layer Stack

A four-layer stack was chosen for use in this design for two primary reasons: to provide ample physical space in which to route the PCB traces and to keep inadvertent radio frequency (RF) noise to a minimum. The internal plane 1 shown in Figure 4 is connected to ground, while the internal plane 2 shown in Figure 4 is divided up into several voltage-specific power planes that are used to power the system’s components. Having these solid ground and power reference planes built into the PCB allows a low-resistance return path for any high frequency noise that may be generated by one of the system components and allows the top and bottom layers to be routed without concern for creating a floating or isolated ground or power connection. This in turn allows for the top and bottom layer trace routing to be optimized to fit within tighter spatial constraints.

Once the board layers are defined, component placement may commence. Components should be placed in such a way that minimizes the length of the traces that are needed to connect them and maximizes signal integrity. Figure 5 below shows a blueprint of the circuit board and gives several examples of how these principles were taken into consideration during the component placement phase.

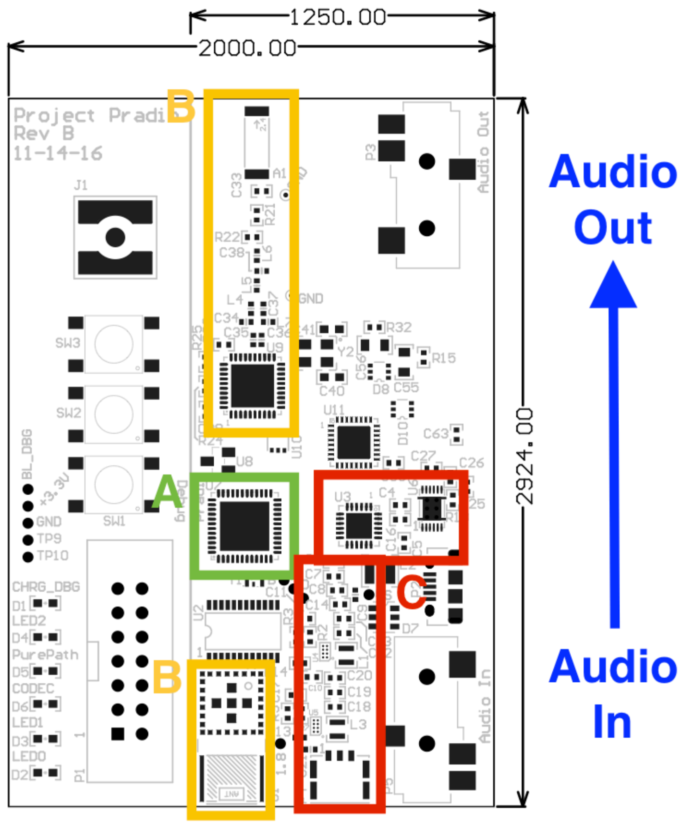


Figure 5. Component Placement

Section *A*, highlighted in green, surrounds the host controller. As mentioned previously, this is the controller that oversees dataflow throughout the system. As such, it is connected to every other peripheral on the circuit board. Thus, it is placed in a central location in order to minimize the trace length required to make the component connections. In turn, this placement leads to less clutter throughout both routing layers and more physical space to route additional traces. Section *C*, surrounded in red, encapsulates that power supply circuit which includes the battery charger, the battery fuel gauge, and two power regulators. By placing each of these components adjacent to each other, not only are the trace lengths shortened, but the number of high-current paths through the circuit board are reduced, leading to less noise in both the RF circuits and the analog circuits of the audio CODEC. Section *B*, shown in orange, highlights the Bluetooth IC and CC8530 audio streaming IC. These are the two RF ICs in the system. As they both operate in the 2.4GHz bandwidth, it is necessary to take advantage of every opportunity to separate the devices within the system. In this case, the ICs have been placed at opposite ends of the board in order to minimize their near-field interference and the cross-talk generated by their respective high frequency circuits. These design choices among many others like them allow the system to operate under smaller physical constraints with increased reliability.

After the components are placed, trace routing may proceed. During trace routing, preference is given to high-speed and high-current traces with designs to again maximize signal integrity. Because this design includes several RF components, special care had to be taken in routing the RF antenna feed lines. These lines were routed with matched 50 ohm impedance in order to minimize signal reflections within the RF circuits. During the routing process, many of the high-current power traces were converted into internal power planes to provide low resistance paths in areas of high electric load. This minimizes power dissipation and voltage drop during transient current spikes. The board underwent multiple layout design iterations before the final form was accepted.

From layout verification, the board went through the manufacturing process. Once the physical board was complete and the circuits were proven, the design was ready to accept programming.

B. Firmware Design

The firmware created in this design is loaded onto the host controller within the system. The host controller can then, in turn, program the various operating parameters of all the peripherals using various serial protocols. Accordingly, the only external programming interface provided by the circuit board is the host controller programming interface. The overall firmware architecture that will be loaded into the host controller is a layered design. The lower layers of the design deal directly with the hardware, while the upper layers tend to deal specifically with system functionality tasks. Each hardware module is initialized and managed by unique hardware abstraction layer (HAL) firmware modules. On top of each HAL layer module is a module that handles the upper-level functionality of the module and implements the methods necessary to abstract the module's interfaces. These upper-layer modules should not be specific to the hardware, but rather should call the HAL layer functions to handle hardware-specific tasks. In designing with this type of modularity, the hardware may be completely replaced, and the upper-level functionality will remain the same. This design strategy is especially important when designing a prototype, such as this board, where the underlying hardware is subject to change. This layered approach has the additional advantage of defining clear lines between software modules as defined by the hardware. The modules defined for this project as well as the connections between them are shown below in Figure 6.

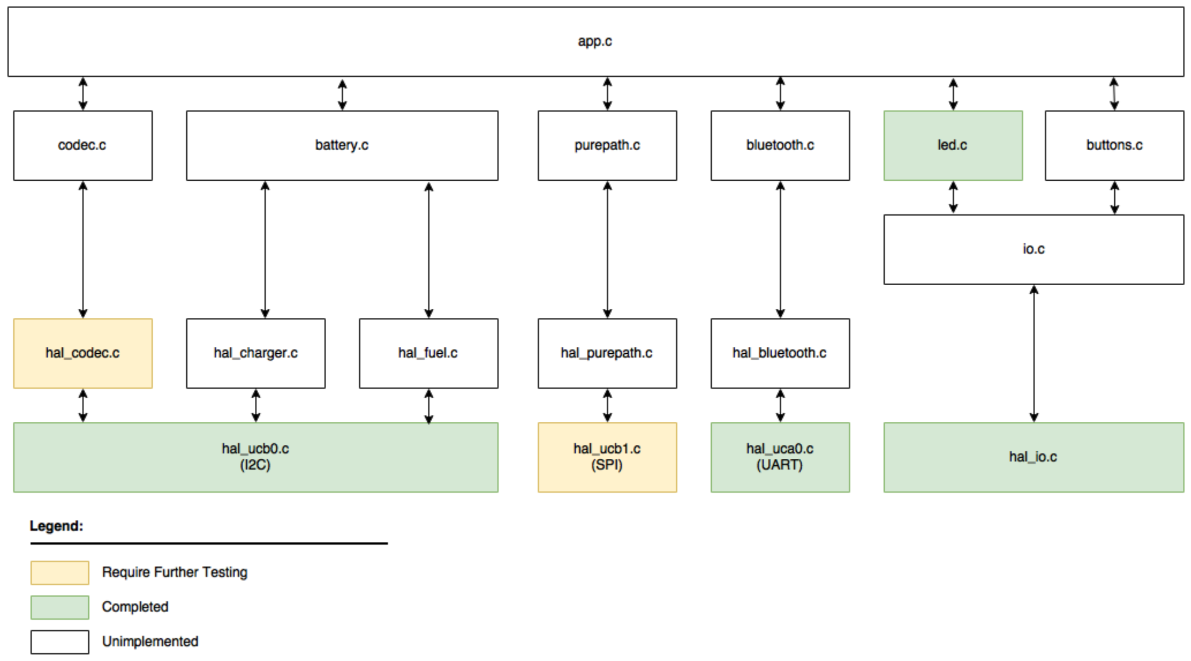


Figure 6. Firmware Architecture

As Figure 6 above shows, many of the proposed firmware modules were left unimplemented. This was due to a misappropriation of time in the schedule, as the amount of time required to complete the low-level hardware abstraction layer modules was greatly underestimated. A brief description of each module that was implemented is given here.

As stated previously, there are three serial interfaces used by the host controller to communicate with the various peripherals; such as the CODEC, battery charger, audio streaming IC, Bluetooth IC, etc. The HAL modules implemented to utilize these serial interfaces are as follows: *hal_ucb0* for I^{2}C communication*,* *hal_ucb1* for SPI communications, and *hal_uca0* for Universal Asynchronous Receiver Transmitter (UART) communications. For each of these interface modules to operate within the system in an efficient manner, some rather complex management firmware is needed. Each interface operates at a data rate that is limited by the signal integrity of the physical medium. Typically, this data rate is much slower than the clock speed of the host controller. Essentially, this means that if the serial protocols are implemented in software, a dramatic decrease in processor performance is unavoidable. As a solution to this issue, IC designers have provided hardware within the IC that handles the transfer of data on a byte, rather than bit, level. While this does increase the performance of the controller considerably, it also introduces some added firmware complexity. This complexity arises in the form of parallel hardware operation; while the CPU is performing an unrelated operation, the serial hardware interface is transferring data in parallel, and the CPU must be interrupted when the interface is available to accept additional data. Herein lies the complexity; because the data is being transferred in and out of the host controller by an interrupt, the data must be buffered in memory until the hardware is ready to accept it. This means that every function that wishes to read or write data across the serial link must do so with a non-blocking function call. A serial read operation will simply add a read request to the memory buffer that will be emptied at a later time by the interrupt service routine. The data that is returned by read operation will return to the reading module via a callback function, but the entry point of the callback function is not the source of the read operation. There is a delay and a disconnect between when the read operation is requested and when the data that was read is returned. This discontinuity causes the design complexity to increase by forcing the system to maintain some form of state during and after serial read and write operations. As previously stated, this implementation will increase system performance at the cost of firmware complexity. The added complexity did contribute to the misappropriation of development time that threw this project off schedule.

The *hal_io* and *led* firmware modules allow the CPU to write directly to the general-purpose input output (GPIO) pins on the IC package. The *led* module shown in Figure 6 provides the *app* module with a clean interface that can be utilized to control the LEDs that make up a portion of the user interface.

The *hal_codec* module is responsible for setting up and interfacing with the onboard CODEC. As a HAL module, it handles the setting of registers within the codec using an I^{2}C interface. To the upper-layer modules, it provides interfaces that allow the setting of output volume, headphone presence detection, analog audio routing, and several others. This module was nearly completed but is lacking some additional testing and implementation before it can be considered fully functional.

While this firmware design is far from complete, a solid and performance-efficient foundation has been laid. Using the HAL interfaces that have been implemented, it will not be difficult to implement the remaining firmware modules in the upper layers.

VI. Results and Conclusion

Although this project did not produce a finished design at its conclusion due to lack of firmware implementation, valuable experience and knowledge was collected during its duration. The experience of taking an idea and developing it into a reality is one that is not easily replicated in the classroom, and the PCB design and manufacturing process is something that is rarely examined outside of the electronics design industry. By going through the entire design process from start to finish, the concepts of designing hardware for manufacturability and ease of firmware development have been reinforced and enhanced, as was the need for strong project management and a well-defined project schedule. Hopefully in the future these concepts will lead to more valuable, on-time designs and a better engineer.

VII. Future Work

In order to complete the design, the remaining firmware modules detailed in Figure 6 must be completed. Once these are functional, the prototype will be complete.

Additionally, an advanced user interface should be implemented on a *controlling device.* As mentioned previously, this type of interface has potential to drastically improve the user's experience with the device.

Moving past the prototype stage, there are new developments in the industry that could lead to increased system performance at a lower price point. In particular, the area of Bluetooth 5.0 and its audio streaming capabilities should be explored. Perhaps utilizing the increased data throughput of the Bluetooth 5.0 standard would allow the shared audio stream to be consolidated with the Bluetooth connection, leading to a lower bill of materials cost, a probable decrease in power consumption, and reduced RF interference [8].

VIII. References

[1] Beats by Dre. (2017). *Music Sharing* [Online]. Available: https://www.beatsbydre.com/support/how-to/music-sharing

[2] WearHaus. (2017). *Wearhaus Arc* [Online]. Available: https://www.wearhaus.com

[3] MyStream. (2017). *MyStream* [Online]. Available: http://mystreamapp.com

[4] D. Chartier. (2011, March 31). *Share, discover music between iPhones with MyStream* [Online]. Available: http://www.macworld.com/article/1158945/mystream_iphone_music_sharing.html

[5] Apple Inc. (2017, April 26). *MyStream* [Online]. Available: https://itunes.apple.com/us/app/mystream/id423918810?mt=8

[6] Texas Instruments. (2015, November). *SLAS897A* [Online]. Available: http://www.ti.com/lit/ds/symlink/msp430f5234.pdf

[7] Texas Instruments. (2017, January). *SWRS160E* [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2564moda.pdf

[8] Bluetooth SIG. (2017, April 26). *Title NA* [Online]. Available: https://www.bluetooth.com

[9] Texas Instruments. (2013, June). *SWRU250M* [Online]. Available: http://www.ti.com/lit/ug/swru250m/swru250m.pdf

[10] Texas Instruments. (2014, November) *SLOS602C* [Online]. Available: http://www.ti.com/lit/ds/symlink/tlv320aic3204.pdf