Sync-In

Levis Agaba, CSE, Ajwad Alam, EE, Joseph Bellve, EE, and Carl Senecal, CSE

***Abstract* — Our world is becoming increasingly more connected by technology. The creation of ever smaller and cheaper embedded electronics has allowed us to add computing and networking capabilities to a wide array of everyday devices. Sync-In takes advantage of these advances in technology to create a set of headphones that communicate wirelessly with other headphones in a local area. Each user can choose to broadcast the input audio signal on their own headset or to scan the area for other broadcasters and listen to what they are sharing. This device has both social and practical applications. The Sync-In headset makes use of an ultra-low power audio codec and single-chip wireless MCU to convert input analog audio signals to digital, transmit those signals over a wireless local area network, and convert the signals back to an analog form on the receiving end. An audio amplifier then controls the output volume of the received signal. Headset users may choose to listen to their own audio signal privately, or transmit it to others within the range of the local area network, or switch between audio signals being broadcasted by other users in the area.**

I. Introduction

IMAGINING oneself on public transportation conjures up images of strangers sparsely seated, staring at screens, headphones in their ears, ignoring the physical world around them. Alternatively, consider a tour guide in a crowded museum, trying to explain the historical significance of American Revolutionary War artifacts to a group of elderly tourists over the din of screaming elementary school students on a field trip. Or, consider a multinational conference, where a keynote speaker needs to address an audience that does not all share knowledge of a single language, requiring the use of multiple interpreters. All of these situations would benefit from the use of wireless headphones able to communicate audio signals to one another in a wireless group independent of the Internet in general.

In the public transportation scenario, social interaction could be encouraged via the use of wireless headphones able to broadcast music signals to one another. Users of the Sync-In headset could choose to broadcast the audio signal that they are listening to, scan the area around them for other audio signals being transmitted, or to bypass the system and use the headphones directly. The Sync-In headset will have indicator LEDs that will light up in the same color for all users who would be listening to each other on a given channel. This feature will allow people on a bus, for example; to share their music locally without an Internet connection, to locate strangers who are listening to the same music, and talk to people who share their musical interests.

There currently exist **Android** and **iOs** apps that allow streaming music to other listeners, but these apps require a data plan or internet connection. Sync-In devices create their own Wi-Fi access points, negating the need for a connection to the broader Internet. Since Sync-In aims to connect as many people as possible in as many places as possible-locally, removing this dependence on the Internet, with its potentially significant monetary access costs, is important for the realization of our goals.

In the museum guide scenario, the Sync-In system would allow members of the tour group to listen to only the signal being broadcast by their particular tour guide who would also

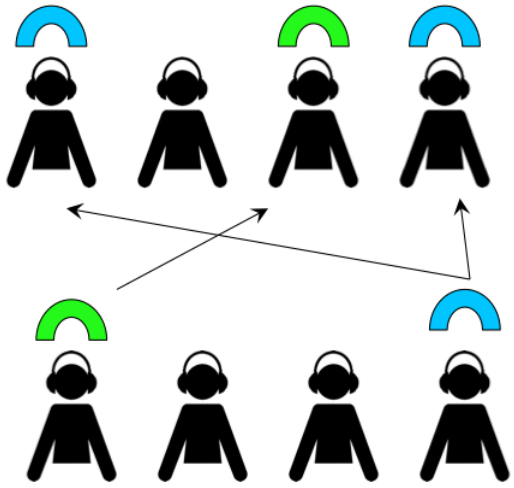

Fig. 1. Demonstration of the functionality of Sync-In headphones. Users may broadcast the signal they are listening to one or many other users. All users listening to a particular channel will display the same indicator color. Users may also choose to neither broadcast nor receive.

be able to adjust the volume level based on the surroundings. Indicator lights would allow multiple tour groups to occupy a single area and would enable listeners to easily identify with the correct group.

In the conference scenario, translations of a speaker's address could be broadcast simultaneously in multiple languages. Attendees could scan the channels until discovery of a language that they understand or chose from a host of speakers broadcasting simultaneously.

A. Alam from Dhaka, Bangladesh (e-mail: aalam@umass.edu).
A. Agaba from Worcester, Ma (e-mail: lagaba@umass.edu).
J. Bellve from Hubbardston, Ma (e-mail: jbellve@umass.edu).
C. Senecal from Hyde Park, MA (e-mail: cjsenecal@umass.edu).

Prior solutions for the tour guide and conference scenarios required specialized radio broadcasting equipment. With Sync-In headphones, users are able to bring their own Sync-In enabled device, potentially lowering operating costs for the event organizers, while the indicator lights provide additional functionality not found in traditional radio receivers.

Our project has a number of requirements and specifications outlined in Table 1. To keep the headphones portable, all of the components of the Sync-In system, including control and transmission units and the amplifier, must fit within the headset itself. The controls should be simple and intuitive with clear indicators. For use on long trips or at lengthy events, the headphones should be battery powered with a charge that lasts several hours, and should be easily rechargeable. To ensure a quality listening experience, the audio signal have a minimum bit rate of a 192kbps. For use in conference rooms or public transportation compartments, the headphones should have around a 100-foot transmission radius. Additionally, to keep the headphones accessible to as many people as possible, the cost per unit should be kept at or below $100, a cost similar to that of other high-quality consumer level headphones.

TABLE I
SPECIFICATIONS

Specification	Value
Streaming Quality	192 kbps, 48 kHz audio quality sampling rate
Concurrent Users	At least 3, ideally 10-30
Range	100 feet
Battery Life	> 4 hours
Cost	< $100 per unit
Network Operation	Wi-Fi, No need for Internet connection
Portability	Complete containment within headset

II. DESIGN

A. *Overview*

Our solution to the problem of an economically viable, power efficient, and intuitive system for wireless broadcast of individual listeners' audio signals to groups of other users is to develop a system based on a single chip wireless microcontroller, an ultra-low power audio codec, and high quality dedicated audio amplifier.

We selected the TI CC3200 as our single chip wireless MCU. The CC3200 provides both signal processing and Wi-Fi capabilities in a small package. The TI TLV320AIC3254 audio codec provides efficient analog to digital and digital to analog conversion. Our amplifier system is based around the OPA1662 operational amplifier and the LM2662 switched capacitor voltage converter.

The use of Wi-Fi allows for transmission characteristics that meet or exceed our requirements for transmission range and quality. The use of a consolidated chip also keeps our costs low, as a number of necessary components that would have required individual sourcing are already contained in the single package.

We considered Bluetooth as an alternative to Wi-Fi, but Bluetooth does not provide the necessary bandwidth for high quality audio transmission, and does not provide the necessary range with common, inexpensive Bluetooth modules. While Bluetooth offers a convenient connection protocol and provides low power transmission, the Bluetooth solution would not scale well to large numbers of listeners.

The other reason we rejected Bluetooth was because it does not provide a broadcasting type of option, only one-to-one transmission. This would reduce the effective available transmission bandwidth for each additional user that attempts to connect to a broadcaster. Wi-Fi, on the other hand, can broadcast information to any number of users on a given subnet with no loss of bandwidth.

The dedicated low-power, low-voltage codec on the Audio

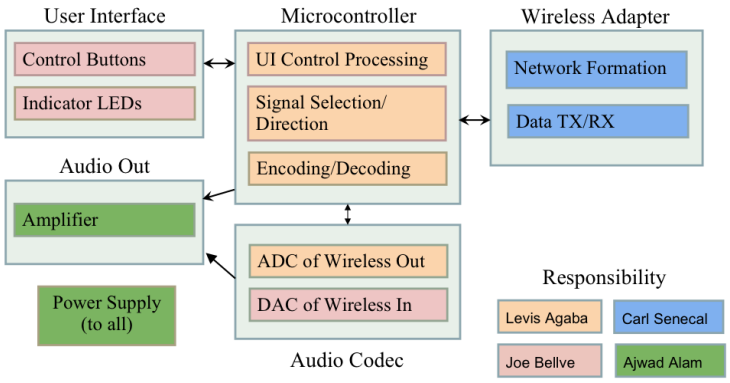

Fig. 2. Block Diagram

booster pack has programmable inputs and outputs as well as a fully programmable miniDSP. The codec's ADC's has a weighted SNR = 93dB which is close to the theoretical value of 98.08dB for 16-bit quantization. Since it was flexible and modular, we decided to use the codec's ADC as opposed to using one on the MCU.

The codec's DAC supports data rates ranging from 8 kHz to 192 kHz which is sufficient for high quality audio. Using a standard 24-bit, 48 kHz sampling format will require approximately 1.2 Mbps of bandwidth, which our networking and processing components should be able to support.

The codec has support for both I^2S and I^2C communication protocols and has capabilities for headset detection, interrupt generation and flexible pin multiplexing. The high conversion factor ensures audio quality as per our specifications. The built in ADC in our single chip wireless MCU did not have strong enough specifications to meet our sampling and conversion specifications.

The user of our own external amplifier allows us to include an amplifier well-suited to audio applications with a low power draw. While we could have included a pre-existing, pre-packaged audio amplifier, designing or choosing our own

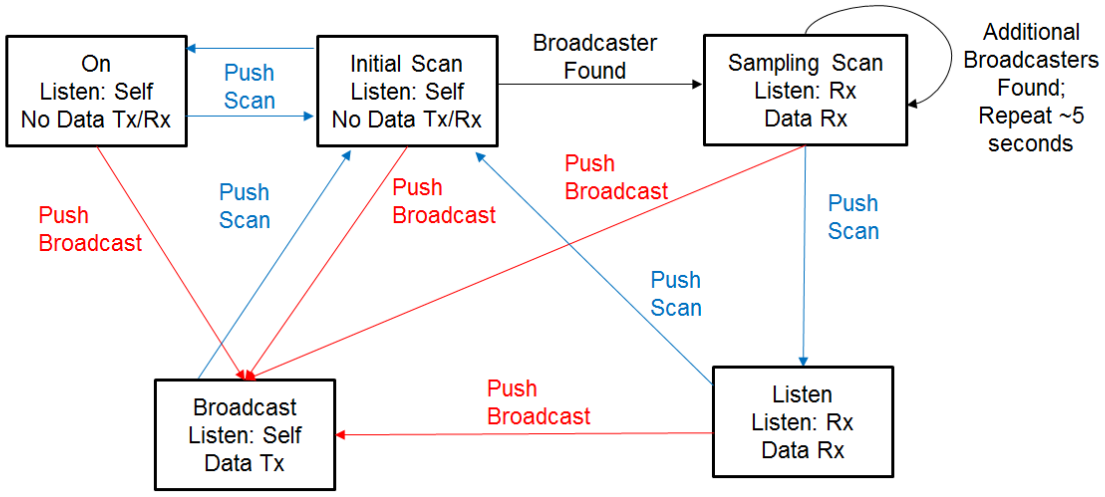

Fig. 3. State Transition Diagram

allows for better control of the distortion and noise characteristics, better control over power draw, and the ability to adjust the output voltages and impedances as best suit the headphones that we chose to contain our amplifier.

B. Block 1: User Interface

Block 1 consists of scan and broadcast buttons, a knob, and an indicator LEDs that allow the user to control and interact with the headphones. A power button turns the system on, the broadcast button allows the user to broadcast whatever they are listening to, and the scan button, once pressed, prompts the system to look for other broadcasters, it then plays a sample of what each broadcaster is transmitting, much like the scan button on a car radio. The system waits until the user presses scan again to settle on the current station. See Figure 3 for a diagram showing the possible inputs and state transitions associated with the user's possible input. The volume knob controls the volume of the audio output when listening to audio received from other users.

As of MDR, the volume knob, broadcast, and scan buttons have been implemented. The volume knob currently uses a potentiometer but will be implemented as a nicer, knurled knob on the actual headset. The buttons use GPIO interrupts to indicate input to the MCU. The interrupts are handled by a dedicated thread that watches for button press interrupts and alters the state of the system appropriately (i.e. changing to access point mode upon press of the broadcast button).

The indicator LEDs have not yet been implemented, but will be a simple addition via GPIO.

C. Block 2: Microcontroller

The CC3200 contains an ARM Cortex M4 as the main microcontroller. The microcontroller serves as the intermediary between all the other components of the system. It runs a button handling thread, a *SimpleLink* thread, an audio playback thread, and a network transmission/receiving thread.

The button handling thread monitors for button press interrupts and configures the networking aspects of the system appropriately upon discovering a button press.

The *SimpleLink* thread handles all of the lower-level operating system tasks and networking events. The CC3200 offers an API by which networking configuration and network transmission/receiving events are done through calls that are handled by the *SimpleLink* thread. Configuring the station in access point mode or setting IP addresses, for example, are done through calls that are handled by this thread.

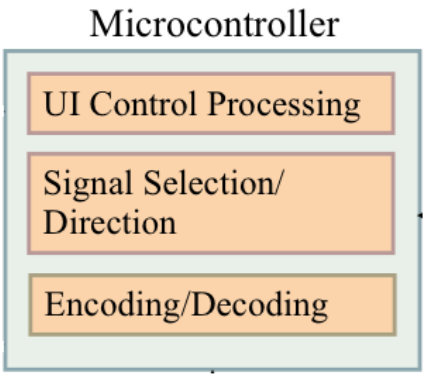

Figure 4: Microcontroller subsystem.

The audio playback thread either passes through the user's current input audio if the user is broadcasting or choosing to neither broadcast nor receive. If the user is broadcasting, the input audio will be sent through the analog to digital converter, encoded, and placed in buffers for use by the network receive/transmit thread. If the user is in a receiving mode, the audio playback thread will monitor the buffers for audio data, send the audio through the digital to analog converter, and play back the converted audio data through the user's headset via the amplifier while data remains in the buffer.

The network receiving/transmission thread handles sending and receiving of data with other Sync-In headsets. This thread will be described in more detail in the networking block.

As of MDR, the button event handling thread, *SimpleLink* thread, and networking thread have been implemented. The audio playback thread is only in the design phase, as

ADC/DAC was not part of our MDR goals.

D. Block 3: Wireless Adapter

Network formation and data transmission and receiving go through this block. The CC3200 contains a separate Wi-Fi processor that monitors for networking events in the background, buffering received packets and raising an interrupt upon networking events. These events are then handled by event handlers within software.

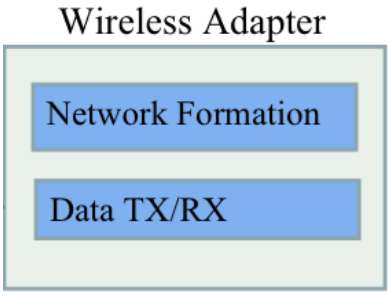

Figure 5: Wireless Adapter Subsystem

The networking thread establishes the CC3200 in access point mode or station mode depending on whether the device is in a state that should include broadcasting or receiving audio. The access point acts a broadcaster, with an SSID matching a particular pattern known to listener devices as a possible broadcaster. The broadcaster access point allows any device to connect and transmits its audio signal via UDP. The listener monitors for UDP packets once it has connected to an access point.

We wasted a significant amount of time attempting to use the Wi-Fi Direct protocol to develop an ad-hoc network, which theoretically would have been easier and more secure for our purposes. However, we had difficulty interpreting the API for Wi-Fi Direct functionality, never got two boards to fully negotiate a connection, and eventually discovered that, while the Wi-Fi Direct protocol supports group formation, the CC3200's API does not. Switching to a traditional station and access point Wi-Fi network model resulted in rapid progress.

As of MDR, the broadcaster access point and listener station modes have been implemented in software. We have tested by debugging over UART. One board is set to access point mode by pressing button A, while another board is set to station mode by pressing button B. Entering a command via UART causes the access point broadcaster to transmit 1000 packets to the station mode receiver, which are displayed on the receiver's UART terminal.

In the future, the network formation procedure will be updated to allow multiple client listeners for a single broadcaster. In our pre-MDR testing, we only owned two CC3200 boards; we have ordered more for expanded testing. The network formation procedure will remain largely the same for stations, with the addition of cycling through access points in scan mode. The broadcasters will have to handle network events slightly differently to allow multiple clients to connect at once. The transmission of packets can still use UDP as a transmission layer protocol, but will now involve multicast rather than direct-addressed packets.

E. Block 4: Audio Codec

This block converts input analog audio signals to digital for transmission over Wi-Fi and converts received digital audio signals back into analog for amplification by the amplifier subsystem.

In order to allow users to share music, we must be able to convert the analog signal coming from their devices into a digital signal, which we are going to send over Wi-Fi. The way we are going to accomplish this is through an Analog to Digital Converter which is included in the TLV320AIC3254 audio CODEC by Texas Instruments.

When looking at an ADC, there are different metrics you judge the quality by: resolution, sampling rate, Total Harmonic distortion and noise. Resolution is the number of discrete values it can produce over a range of analog values [9]. This affects the SNR, Signal to Noise Ratio, of the ADC without oversampling. Sampling rate is the frequency at which the analog values are recorded.

In order to get a measure of what sampling rate we need, we use the Nyquist frequency, which is twice the frequency we care about. The maximum audible frequency for humans is about 20 kHz. So, the rate at which we need to sample is 40 kHz. The ADC we have is able to sample at a frequency up to 192 kHz, which is plenty to retrieve all the information.

Oversampling is when you sample well above the Nyquist frequency and this is useful in order to provide a higher resolution than at the Nyquist rate. The problem that arises is that it comes with noise across the whole spectrum. It is solved using digital filters instead of analog to give sharper roll off, which results in better anti-aliasing.

Resolution is important to meet the quality of music we are striving to achieve. The ADC that we picked has a resolution of 32 bits. So it can take on 232 different numbers to resolve music between the analog volumes of around ±.5 volts. We can define our voltage resolution as Q. With the formula voltage rang divide by the resolution.

Q= 2.3283064365386962890625E-10 Volts. The total harmonic distortion of the ADC is -85 dB.

In order to play another user's music, we have to convert the original analog signal to digital to send it via Wi-Fi and then convert the received digital signal back to analog again on reception for playback. In order to achieve this, we are going to use the same chip, TLV320AIC3254, but use its built-in DAC, digital to analog converter, instead of ADC.

Like the ADC, the DAC has a couple of metrics used to define its performance. These metrics are the resolution, sampling rate, the total harmonic distortion and noise.

The sampling rate and the resolution are matching the ADC with a resolution of 32 bits and a sample rate of up to 198 kHz if over sampled. The total harmonic distortion is more than that of the ADC and it is - 83 dB. DACs also have a metric of

dynamic range which is based off of the resolution. It is the range from the smallest to the largest analog signal we can represent. Since our resolution is high, we started with a low value of -0.5V and high of 0.5V for our ADC.

F. Block 5: Audio Amplifier

This block contains the amplifier that will be used to drive the headphones. A major requirement of the amplifier is that it must have low noise present at the output. Power is also a major constraint, as the amplifier will be running off of a battery source. Keeping considerations in mind, a suitable op-amp was chosen that met the power and noise requirements.

The op-amp chosen for the amplifier design is the OPA1662 chip, designed by Texas Instruments. This chip has been specifically designed for audio applications, as it possesses low-power and low-noise properties. Another important factor is the slew rate of this op-amp. Slew rate is defined as the maximum rate of change of output voltage per unit of time, and is measured in units of V/us. The formula used to calculate the slew rate is as follows:

$$Slew\ rate = 2\pi f V_{pk}$$

For humans, the highest audible frequency is 20 kHz. Taking V_{pk} as 0.447V (line level for consumer audio), the slew rate turns out to be 0.06 V/us. The op-amp's slew rate must exceed this value, and it does, by a large factor, as the slew rate of the OPA1662 op-amp is 17 V/us.

As the audio signal that requires amplification is an AC signal, the op-amp requires both positive voltage and negative voltage supplies. While the positive voltage is supplied by the battery source (+5V), the negative voltage (-5V) is generated with the help of a switched capacitor voltage converter.

The voltage converter that is used to generate -5V from an input of 5V is the LM2662 chip, also made by Texas Instruments. This chip requires two external capacitors, each of 100 µF capacitance. A block diagram of the implementation of the amplifier is presented as follows:

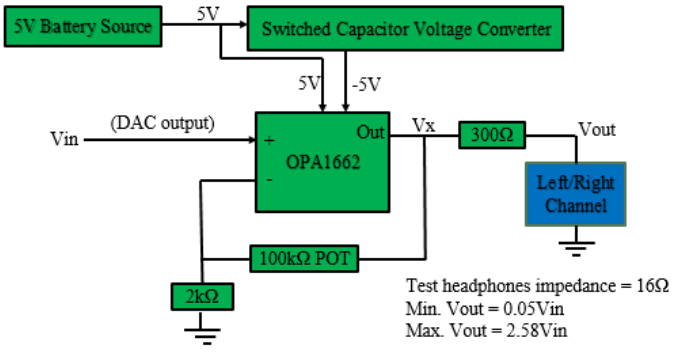

Fig. 6. Block diagram for amplifier

As the block diagram shows, the OPA1662 op-amp takes in the output from the DAC as its input. The objective of the amplifier is to boost up the signal from the DAC to suitable levels; it essentially works like a volume control. As a result, the gain of the network is also variable. In order to achieve this, a 100 kΩ potentiometer was used in series with a 2 kΩ resistor in the feedback network. The maximum value of Vx can be calculated using the following formula:

$$Vx = Vin(1 + \frac{100k\Omega}{2k\Omega})$$

So, at the output of the op-amp, *Vx = 51Vin*. This output voltage is then distributed to the headphones by means of a voltage divider. *Vout* is calculated as follows:

$$Vout = 51Vin(\frac{16\Omega}{16\Omega + 300\Omega})$$

Thus, the voltage that drives the headphone drivers, *Vout = 2.58Vin* at max. Note that these calculations are based off of our test headphones, which had an impedance of 16Ω. The preferred headphone drivers for this project are yet to be chosen, and so the amplifier design may change depending on the impedance of the drivers that we choose. The 300 Ω resistor at the output of the op-amp is used to increase the load that the output of the op-amp sees. It was realized that adding this resistor knocked down the noise at the output and the high frequency hiss a lot.

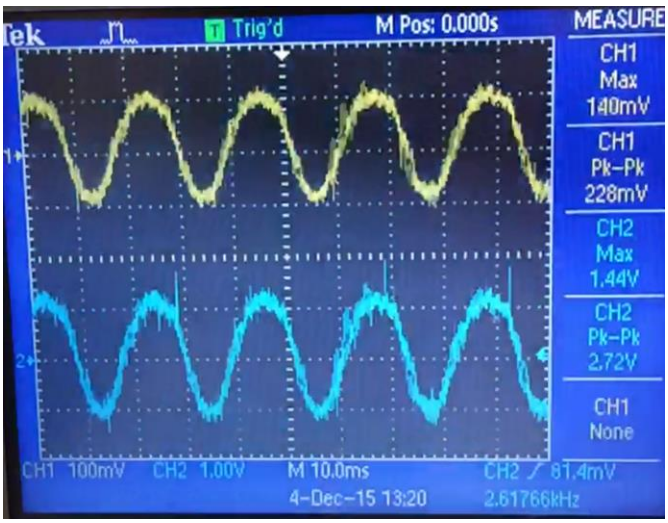

Fig. 7. Amplifier test results

The figure above shows the amplifier test results, with a test audio signal on channel 1 and the output on channel 2. It can be seen that a good replication of the audio signal is achieved at the output, as there is not much distortion.

G. Block 6: Power Supply

This block is yet to be implemented practically, but an overview of how it will be implemented is described as follows. The power supply, which has to be rechargeable, will power 3 major subsystems of the project:

i. The MCU
ii. Booster pack (ADC/DAC)
iii. Amplifier (with switched capacitor voltage converter).

A 5V rechargeable battery will be chosen to power the subsystems. There are two voltage requirements for our

project. The MCU and the audio-booster pack require a 3.3V input power supply whereas the amplifier and the switched capacitor voltage converter require a 5V input power supply. As a result, two voltage regulators will be used to get these two output voltages from a single input voltage.

To test the power supply, all the major subsystems will be run at max load. The power supply should be able to supply power for at least 4 hours, which is the minimum specification desired for our project.

III. Project Management

Table A: MDR goals

	MDR Goals	Status	Remaining work
1	User Interface block – MCU communication	95% complete	Include UI LEDs
2	MCU – Network Adapter communication	100% complete	3 way communication
3	MCU – Amplifier block communication	Not complete	Direct signals from DAC to Amplifier
4	Working Amplifier	100% complete	Reduce noise + distortion, reduce power consumption

Our MDR goals were not transparent enough when presented during PDR. Some of them were just too ambitious. As a result, our entire block diagram had to be restructured around the feedback we received from evaluators. MDR goals and responsibilities had to be redefined to reflect changes in overall system design.

With the noise cancellation block gone, Joseph's responsibilities had to change as he had previously been responsible for noise cancellation and UI design. We thought it made sense to have one person handle both UI design and UI processing done by the MCU and Joseph was comfortable doing that.

As was demonstrated during the MDR presentation, the scan and broadcast button are able to communicate with the MCU effectively. They were shown to configure the MCUs on two identical boards in one of two modes, AP or STA which determine broadcaster and receiver respectively.

Although Joseph was primarily responsible for UI design and processing, Levis and Carl chipped in whenever things fell apart as they often did.

As the only CSEs on the team, Levis and Carl concentrated mostly on the networking aspects of the project which included network formation, as well as the transmission and receiving of data between two MCUs. Since our goals were more closely aligned we often encountered similar problems. In such cases whenever a solution was found by one team member, it was often shared with the other team member so that other things could be worked on to push the project further. Sometimes were worked individually sometimes we worked together but we were always up to date and often on the same page.

Like Joseph, Ajwad, an EE senior, had been assigned to work on the noise cancellation block in addition to the amplifier subsystem but in this case, his tasks remained challenging so little changed with regards to his MDR responsibilities.

As was demonstrated during the MDR presentation, Ajwad delivered on a working sound amplifier and although it performs well, some work remains to be done. Both noise and distortion need to be reduced. Since the DAC and ADC were not required components for MDR, it was not possible to establish communication between the MCU and the amplifier. Ideally we would have the DAC process received digital signals forwarded to it via the MCU and have it forward the reconstructed signal to the amplifier.

Despite working on the hardware side of things, Ajwad and Joseph have contributed to the overall design of the project by offering suggestions for low-power components and in discussing the performance issues that arise (in software and hardware) from the inclusion of such hardware components. So team communication has been vital to the implementation of all subsystems.

Table B: Minimized Gantt chart

Tasks Lead	Task	Start	End	% Complete
Ajwad. A	Sound Amplifier	9/9/15	12/3/15	100%
Carl. S	Network Formation	9/9/15	12/3/15	100%
Joseph. B	UI Design & UI Processing	9/9/15	12/3/15	100%
Levis. A	Transmitting & Receiving	9/9/15	12/3/15	100 %

Our Plans for the future.

Tasks Lead	Task	Start	End	% Currently Complete
Ajwad. A	Power Supply	1/27/16	3/3/16	0%
Carl. J. S	Multicasting	1/27/16	3/3/16	0%
Joseph. B	ADC implementation	1/27/16	3/3/16	0%
Levis. A	DAC implementation	1/27/16	3/3/16	0 %

IV. Conclusion

As was mention before, our project is where it needs to be as far as meeting our MDR goals is concerned.

We have demonstrated UI functionality and board to board wireless communication as dictated by UI controls. We need to integrate the amplifier block, the ADC/DAC block, and the power block once those blocks have been implemented to a sufficiently developed point.

Evaluator feedback helped direct our progress and it's safe to say we would probably still be working on noise cancellation if it was still part of the project. Despite that, we would like to give it a go in the future if time permits.

We would like to build on our current subsystems and if possible expand some of them to include additional feature and also reduce power consumption as much as possible and have a small form factor.

We expect certain tradeoffs between performance and power-consumption at some point in the future.

PCB integration is another difficulty we will certainly have to work through. Luckily, we are only using a limited number of components on each development board, the MCU inclusive.

The transition from one-to-one communication to broadcast/multicast is expected to go smoothly, as associating more users with an access point is a common, fundamental procedure to networking, and multicast protocols already exist and are well-documented. The only difficulties should be in continuing to interpret TI's API for the CC3200 and synchronizing the receive/transmit system with the audio playback system.

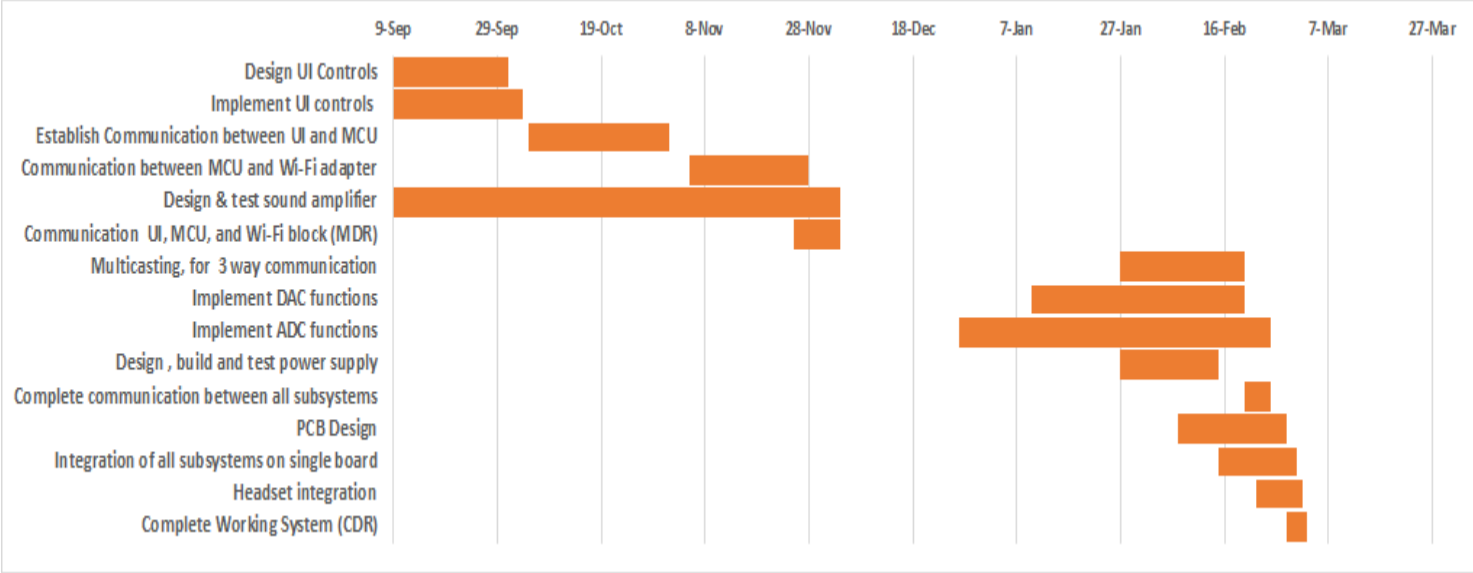

Figure 8: Gantt chart

Acknowledgment

We would like to thank our advisor, Lixin Gao, and our evaluators, Joseph Bardin and Baird Soules, for their invaluable advice and support.

References

[1] Texas Instruments. (2014). *CC3200 SimpleLink Wi-Fi and IoT Solution With MCU LaunchPad Getting Started Guide: User's Guide*. Retrieved Dec. 9, 2015 from http://www.ti.com/lit/ug/swru376c/swru376c.pdf

[2] Texas Instruments. (2014). *CC3200 SimpleLink Wi-Fi and IoT Solution, a Single Chip Wireless MCU: Programmer's Guide*. Retrieved Dec. 9, 2015 from http://www.ti.com/lit/ug/swru369b/swru369b.pdf

[3] Texas Instruments. (2014). *CC3100/CC3200 SimpleLink Wi-Fi Internet-on-a-Chip: User's Guide.* Retrieced Dec. 9, 2015 from http://www.ti.com/lit/ug/swru368a/swru368a.pdf