Streaming Audio over Visual Light Communication

The Coffee Boys
Rushi Patel – ruship Kiran Vishal - vishal94

Abstract — *The transmission of audio from a .wav file to a speaker wirelessly using visual light communication. Audio should be streamed in real time and allow speakers to be connected and disconnected on-the-fly.*

1. Introduction

Our project began as a way to get some first hand experience with "Li-Fi" or visual light communication. The idea of having one way transmission of information with the use of light has been an interesting advancement in communication methods possible from server to client. The current Wi-Fi spectrum is limited to the highest radio frequency which currently is 5Ghz (802.11ac). However the use of light as a means of communication can avoid this bottle neck and instead reach frequencies of 100x that of Wi-Fi and still remain in the visual light spectrum.

This problem is important because eventually communication channels will need to find an alternative to the currently crowded Wi-Fi bandwidth. This alternative method of data transfer can help with special applications where unrestricted one way communication will aid with broadcasting information to all devices in a given area. An idea would be the use of transmitting information to all individuals in a stadium or ball park with the lights already installed in the stadium. This would make it easy to send information about the game directly to anyone sitting under the lights. Since light cannot pass through walls like radio waves, visual light communication has an inherent security benefit by only communicating with those inside a room that is broadcasting the current message. For example a warehouse could pass information around to all autonomous devices without the worry of outsiders interfering with the signal.

Our first steps in understanding how communication over light works we decided to begin by studying the current methods used today to perform data transfer over light, as well as what new methods are used with VLC. Eventually we decided on communicating audio because of the one way flow of data from source to speaker. This application lends itself well to the basic idea of communicating with light since it is a one way data transfer from server to client.

The final product consists of a Gumstix board which sends a .wav file to a receiver circuit which is connected to a set of speakers. The receiver circuit does not contain any proprietary boards or information. This makes connecting to another Gumstix board outputting audio data as simple as moving it in front of the light source. Our implementation also contains no delay and can be connected and disconnected while the audio source is being played.

2. Design Flow

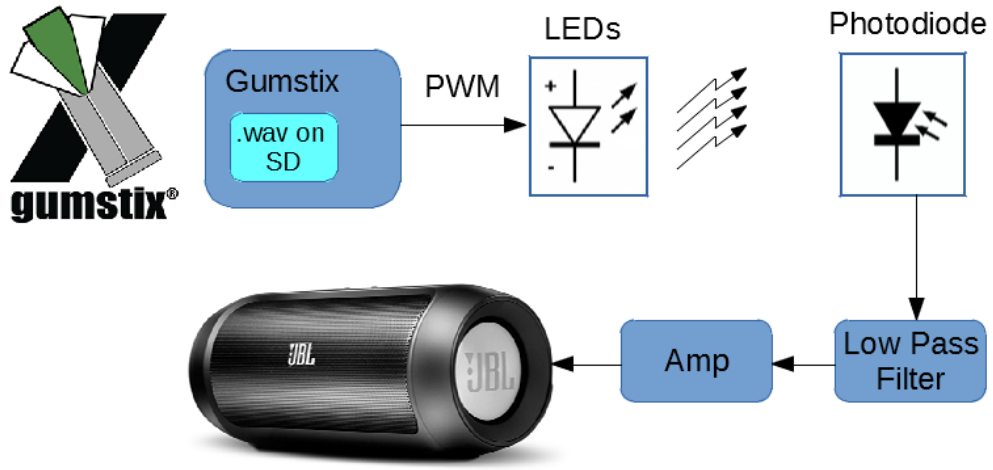

Figure 1. Diagram of communication from Gumstix to speaker

The hardware consisted of an LCD screen and LED panel connected to a Gumstix board. The LCD display was used to operate our audio controller (media player) which had the ability to play three tracks. Each track was stored on an sd card mounted on the linux operating system found on the Gumstix board. The connected LEDs were used to transmit the light data. All of these components acted as our sender circuit. The receiver circuit consisted of a photo-diode which collected the light wave data and translated it into an analog signal which was then sent through a low pass filter and amplifier before transmitting it to a speaker for playback. The speaker was connected with a simple 3.5mm audio jack to the amplifier circuit. Since the overall system was separated into a sender and receiver system. The sender code and circuit was lead and configured by Rushi and the receiver circuit was designed and developed by Kiran. A major portion of the project was spent during integration of the two systems and was done together in room 115. Since both of us had knowledge of our own subsystem and we required the use of power supplies and oscilloscopes to diagnose and debug any problems all integration was done together.

3. Project Details

The project contained numerous different parts which all worked together to help make our end product a success. Each part is described in detail below and will start from the source .wav file and complete at the speaker. This order will help with any future implementation efforts and recreation of our project.

a. **QT User level application** (See /QT/main.cpp, /QT/update.cpp, and /QT/update.h)
The QT application contains the code which was used to read a given .wav file and convert the data into an appropriate value used in the kernel module. The basic functionality consisted of 4 user interface buttons and a Qtimer. Each UI button is connected to an individual callback function when pressed. The 4 functions consists of playing Track1, Track2, Track3, and stopping all playback. Once a track is selected for playback, a POSIX thread is launched and set to run looping playback function "thread_play" (found in /QT/update.cpp). This function begins by opening the kernel device "/dev/mygpio" and opening a file pointer for the specific .wav file associated with the button that was pressed. If both files are opened successfully the main loop is started. In this main loop each sample data is read from the .wav file, with the help of the libsndfile c library, and sent to the kernel module at an 8 kHz rate or after 125 microseconds between samples. We initially had setup a nanosleep(125000) as our delay between loop iterations. However the Gumstix that we were using did not correctly use the nanosleep function and was returning control back to the user level application after 4-10 milliseconds. We resolved this problem with the help of Carlos and instead implemented a spinlock while monitoring the monotonic clock using the function clock_gettime(). This was our final challenge which helped to enable proper playback of the sampled data. The libsndfile library was found online as a simple to use resource to read different audio files on a linux operating system [1]. I quickly cross-compiled the library and incorporated it into the QT code.

b. **Kernel Module and LEDs** (See /km/mygpio.c)
Using the code from the previous lab which involved PWM control. The kernel module was very straight forward. During initialization the PWM registers were enabled and control was set to fast mode to avoid any period disturbances when changing between samples. The simple write function was then used to translate a sample which was in the range of 0-255 into a valid PWM duty cycle and would maintain this configuration until the next time the kernel module is ordered to change the sample again. On exit the PWM is disabled and all kernel memory is cleaned up to avoid any kernel panics. The set of LEDs are directly connected to the PWM0 output pin on the Gumstix board. The main purpose of PWMing LEDs was to alter the brightness of the LED output which when read by the photo-diode would translate the sample into a voltage to control the speaker.

c. Stage 1 – Initial overview + Receiver photo-diode with inverting amplifier

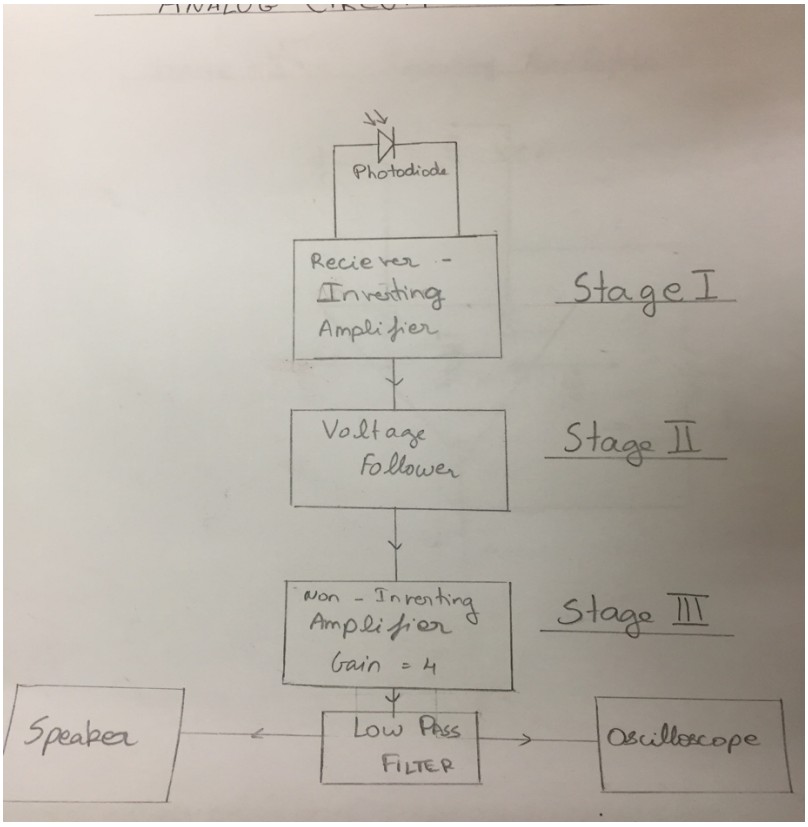

Figure 2: Analog Circuitry stages - block diagram

The above block diagram shows the stages used in the analog circuitry and the oscilloscope in the picture was used to verify if we are receiving the intended signal after each stage before connecting it to the speaker.

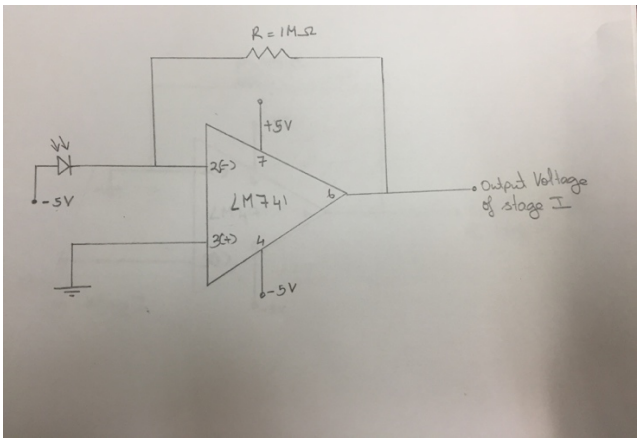

Figure 3: Analog Circuitry stage 1 – receiver photo-diode with inverting amplifier

The main component of the receiver section is the photo-diode. As expected, the photo-diode was extremely sensitive to the changes in the light. One weakness being the current produced

was very small and we tried connecting the speaker (8ohm) directly to it, but failed. So we decided to use some analog circuity to alleviate this issue. Since our skills were a bit rusty on this front we implemented it in stages. Our first attempt was the use of a non inverting amplifier of gain 10 but this design didn't have enough current to drive the speaker even though it had a high voltage output. Thus instead we replaced it with an inverting amplifier to receive the signal and give it to the next stage without any loss or distortion.

d. Stage 2 – Voltage follower

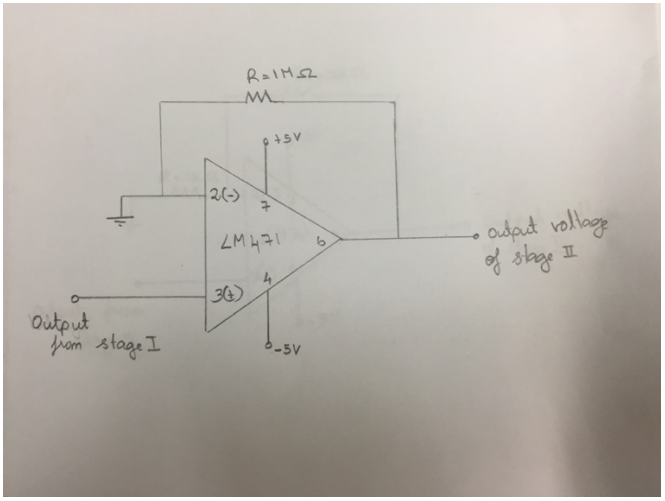

Figure 4: Analog Circuitry stage 2 – Voltage follower

The output from the previous stage was given to the voltage follower. The purpose of the voltage follower is to isolate the input from the output, this removes the load placed on the components of stage 1. The output will be equal to the input to the amplifier but the current drawn for the remaining stages of the amplifier will be directly from the supply and not from the input. This stage solved most of our problems with a lack in current to drive the load (speaker).

e. Stage 3 – Non inverting amplifier

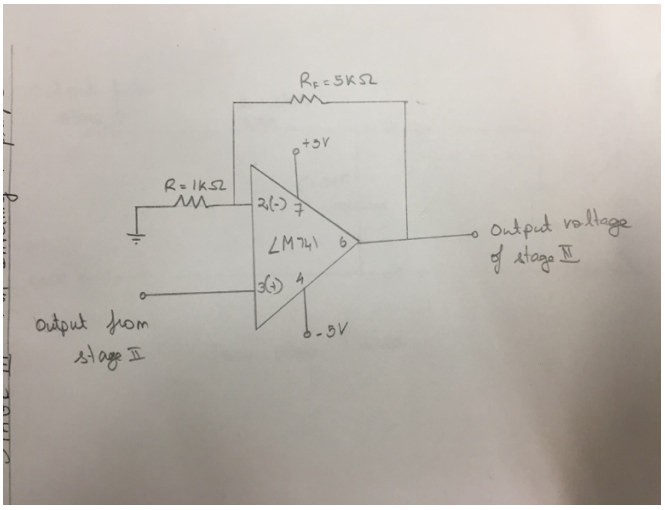

Figure 5: Analog Circuitry stage 3 – Non inverting amplifier

After the first 2 stages the photo-diode signal was received and a stable input to the speaker was produced. The next step was to increase the voltage so that it can be easily heard when played using the speaker. For this we used a non inverting amplifier of gain 4. The output of stage 3 had all the required specifications for the signal to be played by any speaker.

f. Stage 4 – final stage – low pass and speaker

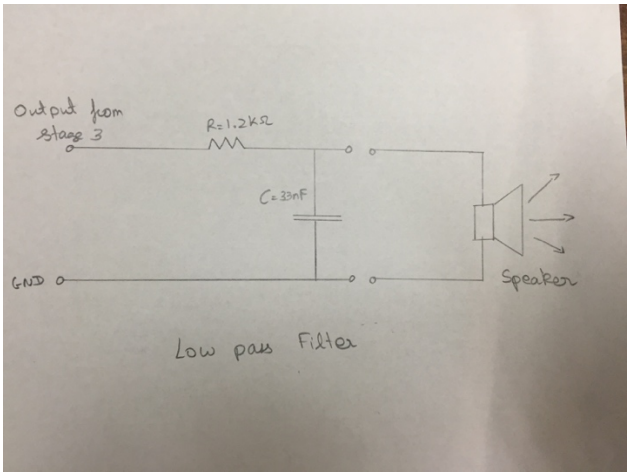

Figure 6: Analog Circuitry Final stage – low pass with speaker

The output signal from stage three was speaker ready. We added a low pass filter because it made our output sound much better. The range of the filter was purely trial and error, we used a resistance of 1.2kohms and a capacitor of 33nF. After this was put in place, the sounds produced was much smoother. This was essentially because the low pass filter removed the high frequency PWM noise. We also replaced the 8ohm speaker with a JBL charge (4ohms), since it has internal amplifiers and a volume control to help with any volume control for the demonstration.

During our testing of all these stages we connected the speaker as well as the oscilloscope to both the transmission end and the receiver end. This was our main testing ritual to make sure we were sending and receiving the same signal at both ends.

Difficulties: The analog circuity in place isn't the most efficient implementation. We used multiple stages so that we can be sure of what we are sending and receiving. The main issue we had was the mismatch in the impedance of our analog circuit and the speaker. Using different implementations of the amplifier and the filters we landed on a design which could give us what we wanted. But there is a definite scope for improvement in the future to make it more compact and sound better.

4. Summary

I believe that our project turned out successful but can still be improved upon in the future. The final output of each track contained a considerable amount of background noise and can be eliminated with a proper extension of time and effort on the entire system. However, with the hard work each of us did, I think the final objective of the project was met and we successfully streamed audio between a device and speaker using visual light. Some future improvements would be to increase the brightness of the LED system to avoid the need to cover the circuitry in a box as we did in the demonstration. A next step in visual light communication would be to stream a video simultaneously with audio or substitute Wi-Fi communication with two VLC systems.

References

[1] libsndfile API. http://www.mega-nerd.com/libsndfile/api.html