Abstract

This document reviews the previous six weeks of work done on the Smarter Thunder Lamp. This includes not only a breakdown of problems and solutions the team encountered, but a detailed account of what each group member has worked on so far. The project is broken down into three sections. Each of these sections discusses the progress that has been made during the development that has been executed this term. It then details each of the problems that has been encountered during the different phases of development, and the solutions, or lack thereof, for each of these problems. The future plans for this project along with a rough timeline are also included in each section.
I. PROJECT PURPOSE AND GOALS

The Smarter Thunder Lamp is a thunder cloud shaped lamp with lightning like functionality. Inspired by a popular art piece by Richard Clarkson, the lamp will need to simulate a thundercloud via the lights inside and a speaker. Richard Clarkson’s lamp is extremely expensive, and thus inaccessible to many people, however, this project will aim to change that, by creating a Smarter Thunder Lamp for a much lesser cost. The Smarter Thunder Lamp also functions as a Bluetooth speaker and light source with a mobile application that can determine these configurations. In the end, the project will produce a novelty product for the home that many people can enjoy.

II. ROBERT ROSENBERGER: SOUND PROCESSING AND BLUETOOTH

The Bluetooth elements to be discussed in this section deal with the lamp-side hardware and firmware. The Bluetooth on the physical lamp is what will allow a users Bluetooth device and the lamps companion application to interact and control the lamp. The lamp will receive commands for how to operate sent to it that it must react to while music is being streamed to it. The applications Bluetooth interaction will be discussed in a later section.

The sound processing on the Smarter Thunder Lamp has to do with the music reactivity mode that is required. The lights on the lamp will need to have the ability to change based on what music is being played through the speakers. The processing must not have a noticeable impact on the music itself. The process must be able to take small samples of sound to look at both the volume and beat. Based on what is found, the lights will receive corresponding commands to change.

A. Current State

The initial implementation of the Bluetooth hardware for the lamp is feature complete. A Bluetooth module has been connected to the Arduino that is able communicate with both the Arduino and the speakers at the same time. An RN-52 Bluetooth module, see Fig 2 was chosen to be this device (further details discussed in section II-C). This module supports both the SPP and A2DP Bluetooth protocols required by the lamp. A serial connection has been established between the RN-52 and Arduino to allow commands to be sent to the Arduino. At the same time, music is able to be streamed from another Bluetooth device and output from the module via speaker-out pins. The supporting lamp-side software is feature complete as well. The serial connection is able to successfully read commands sent to the module and react based on what the command received was.

The Bluetooth features have been successfully tested. A testing circuit consisting of basic LEDs (Not the addressable LEDs required by the project but individual LEDs) was connected to the Arduino and testing software was written. The circuit can be seen in Fig 2. Based on a given commands, a specific pattern on the LEDs is shown. The companion application was not used to the test the reception of commands over Bluetooth. Instead a smartphone with a Bluetooth terminal application was used. This circuit and software will be useful for when integration between the companion application and lamp hardware begins.

The sound processing hardware is not feature complete. To be discussed in greater detail in section II-C a new plan is currently being implemented to accomplish the sampling of the sound. An audio reader (A fancy microphone) module will be connected to the Arduino so that it can take the readings. The sound processing software is in a similar state as no true success was ever had in analyzing the sound readings. Currently there is code written that may work if the audio reader works as expected. It is assumed that this code will need to be refactored.
B. Future Work

Refinement and the flushing out of bugs is all that is required for the lamp-side Bluetooth. Currently there is a known issue that, sometimes, when no music is being streamed to the RN-52, the speakers emit a horrible screeching sound. The sound starts roughly one second after music is paused if it happens at all. The root cause of this is unknown and is currently under investigation. The other future work expected is the refactoring for any compatibility issues that may arise when integration with the companion application begins. This is expected to mostly have to do with the command strings that are being sent and received respectively.

The sound processing requires a bit of work. The audio reader module needs to be integrated into the circuitry (As explained later, it has yet to arrive). Once this is done, the supporting code will need to be refactored and tested. This will involve modifying the current test circuit as well as the testing code. The test circuit will need to have more LEDs to accommodate more patterns; reflective of the multiple patterns that the lamp will express in its lights. The testing code will need to have the sound processing code added to it. The goal of this will be to see the sound be processed, music streamed, and commands given to the Arduino all happening at once.

C. Blocking Issues

The largest problem has been inexperience with the subject matter. It hasnt been the root of all the problems in these two portions of the project, but it contributed to enough that a specific note is being made. Problems arose that were never considered because it was unknown that it could be an issue in the first place. Specific issues and solutions will be described next.

1) Problems:

- The original Bluetooth module that was purchased for use was the HC-05. This module functioned perfectly for the commands portion of the Bluetooth. This module is not able to support the streaming of music. A different Bluetooth module was needed. The HC-05 only supports the SPP Bluetooth protocol. Streaming of audio, or any other complex and large data, requires A2DP.
- The original method of getting sound samples to the Arduino was to have the Bluetooth module receive the data, parse out commands from the music, then send the music to the speakers. Once the issue described above was found, a fallback method was implemented. The wires carrying the sound signal themselves were to be hijacked and read straight into the Arduino board. Sound waves have a positive and negative amplitude. The Arduino can only read positive amplitude; negative can even damage the board. A circuit to modify the voltage for reading was needed. The circuit required was more involved than the team wished, and previous attempts did not work.

2) Solutions:

- A new Bluetooth module was ordered to replace the HC-05. The board chosen was the RN-52. This module supports both SPP and A2DP Bluetooth protocols as described above. The reason this board was chosen extends beyond having the proper protocol capabilities. This board has a native audio amplifier with speaker channel ports. This means the Module can handle music streaming all on its own without needing assistance from the Arduino. In summary, it is a much better chip in every way, shape, and form.
- Dedicated hardware, not designed by the team, is being used to solve the audio sampling problem. A chip that is an audio reader is going to be used to take sound samples for the Arduino the process. The specific board is a DAOKI
High Sensitivity Sound Detection module as seen in Fig 3. It is Arduino compatible and will handle all collection and modification of the sound. It will provide the Arduino with the precise information and format it needs to process the sound data.

D. Code Samples

```cpp
// Check to see if a command was received from the app
if(Serial.available() > 0){
  cmd = Serial.read();
}
switch(cmd){
  case L_OFF: // 1
    // CHANGE LIGHTS
    ALL_OFF
    digitalWrite(RED, HIGH);
    break;
  case L_STATIC: // 2
    // CHANGE LIGHTS
    ALL_OFF
    digitalWrite(GREEN, HIGH);
    break;
  case L_RAND: // 3
    // CHANGE LIGHTS
    ALL_OFF
    digitalWrite(BLUE, HIGH);
    break;
  case L_REACT: // 4
    // CHANGE LIGHTS
    ALL_OFF
    break;
  case L_STORM: // 5
    // CHANGE LIGHTS
    ALL_ON
    break;
}
```

Fig. 1. Sample code from the testing firmware to modify the test circuit LEDs based on the received Bluetooth command.

E. Photos

Fig. 2. RN-52 Bluetooth module on a break-out board.
III. Supriya Kapur: Lighting and Aesthetic Design

The lighting elements in this project consist of writing code that allows the LED lights to mimic four lighting settings that the user can set, which include being statically light, randomly lighting up, emulating a thunder cloud, and lighting up to the beat of music being played, along with settings to adjust the lamps brightness and colors. The light strip will also need to be wired to the Arduino board correctly so that it can be controlled with Arduino code and have enough voltage to power the lights without frying the board.

The aesthetic portion of the lamp mainly concerns the materials used for creating the cloud itself, its shape, and the way the lights are arranged within the cloud shell in order to make it look as realistic as possible. This portion of the project is key, as it is what makes the speaker unique and desirable, but this will be focused on once the technical aspects of the project are complete.

A. Current State

Thus far, a basic light library has been created in the Arduino language that represents the basic firmware for the lamp. This library houses functions such as turning all of the lights on, turning all of the lights off, adjusting brightness and changing the lights color. The purpose of creating this library is so that when the user picks one of the four lighting settings, the simple functions from the light library can be called in order to make the needed events happen. Excerpts from this library can be seen in the Code Sample section below. The library will help to make the code more readable as well as more concise, as there is limited memory on the board being used.
In addition to creating the basic functions that will be used, a basic structure for the wiring of the light strip to the board has been successfully tested. An LED RGB strip with 50 lights was connected to the Arduino board with one ground pin and one data pin, and the positive and negative wires were attached to a separate 12-volt power source. This set up represents a smaller scale version of the final LED and Arduino board set up will look light. This set up can be seen in Fig ??.

All the functions in the light library have been successfully tested on this set up as well.

B. Future Work

Right now, the functions in the light library uniformly change the entire LED strip, so the next step will be adding functions to the light library that access smaller groups of lights at a time. The purpose of this is so that when the light strips are arranged within the cloud exterior, the lights will look like a thunder cloud, with different sections lighting up at different times, rather than just a cloud that flashing light uniformly. In addition to adding more to the light library, four functions will need to be added to the main control file for the lamp that represent the four lighting settings the user can set.

For creating the lamp itself, the group plans on making a blueprint of the size of the lamp so that parts, such as the LED strips, wiring, frame, and cotton cloud material, can be ordered. By the end of this term, the plan is to have a working model of the cloud that represents the size and shape of the cloud, with the LEDs appropriately distributed about the wire frame. This model however will not include the cotton that represents the actual cloud. The work around creating the cloud itself will be done at the very beginning of spring term once all of the technical aspects of the project have been completed can successfully work in tandem.

C. Blocking Issues

1) Problems: One of the main problems with the lights has been figuring out how to correctly wire them to the Arduino board. It was unclear at first which wires represented the ground and which were the data pins, as the LEDs ordered didn’t have specifications as to what was what.

Another problem was getting the correct amount of voltage to the LEDs. The strip that was used had 50 RGB LEDs and needed more voltage that the Arduino board could handle, so for a while, the lights were only dimly light and flickering because they didn’t have enough power.

2) Solutions: Eventually, after trial and error, the correct wiring was figured out. There turned out to be two data pins, when we only need one, so one wire is unused. Once this was figured out, the LEDs were able to be controlled by the Arduino board.

In order to get the proper amount of voltage to the LEDs, a separate 12-volt power source was used to power the LEDs and a much smaller power source was used to power the board. This solved the problem of not getting enough power to the entire strip, while also making sure that the board was not getting fried.
D. Code Samples

```c
/* Turn all lights on */
void allLightsOn()
{
    analogWrite(REDPIN, HIGH);
    analogWrite(GREENPIN, HIGH);
    analogWrite(BLUENP, HIGH);
}

/* Adjust Brightness of all lights (value between 0 and 255)*/
void AdjustBrightness(int LedBrightness);
{
    analogWrite(REDPIN, LedBrightness);
    analogWrite(BLUENP, LedBrightness);
    analogWrite(GREENPIN, LedBrightness);
}

/* Change Color of all LEDs*/
void allLightsChangeColor(int red, int green, int blue);
{
    red = 255 - red;
    green = 255 - green;
    blue = 255 - blue;

    analogWrite(REDPIN, red);
    analogWrite(GREENPIN, green);
    analogWrite(BLUENP, blue);
}
```

Fig. 5. Sample code from the LED library that changes the colors of the RGB LED strip.

E. Photos

Fig. 6. Set up for testing the LED lights.

IV. COURNEY MILLER: MOBILE APPLICATION

A. Current State

The first thing that was completed for the mobile application portion of this project was the design for the user interface. The general layout of what the app will look like at completion was put together in order to give the developers an idea about how the application might function at a high level. This design was implemented on paper as well as in software; this was meant to aid in user interface design.
Once the user interface was designed, the development of the application began. The current state of the application is very simple. There are four buttons on the first page of the application. Each of these buttons has a unique event that is triggered when they are toggled by the user. The behavior of these events is not the expected behavior, but it is a placeholder for when the application is able to be integrated with the other parts of the project. This will be discussed in more detail later.

B. Future Work

In the immediate future, the plan for the application is to add the missing components in the application. The four buttons that were talked about earlier will cover some of the functionality that is expected in this project, but they don’t cover all of it. For example, the mobile application needs to be able to turn on and off the lights, control the brightness of the lights, determine if the lights will sync with sound, and many other actions. Each of these actions will need to have an option that the user is able to control from the application, and the current state of the application does not have every available option. Adding these capabilities will require following the design that was discussed earlier.

Once each of the actions is added to the application, the correct behavior will need to be implemented. This will require sending data through Bluetooth. This data will be received and used in order to control the behavior of the Smarter Thunder Lamp.

After the functionality is implemented and able to be tested, the design of the mobile application will be refined. This is happening later in the project in order to ensure that the functionality can be integrated into the rest of the project with enough time to do thorough testing. While the testing is happening, the design can be adjusted in order to make an application that looks better to the average user.

After both of the previous items have been completed, the next phase for the mobile application will be to test that it is correctly communicating through Bluetooth. This will require integrating the mobile application with the other portions of the project. At this phase of the project, testing will begin in order to ensure that expected functionality is behaving correctly. It is expected that there will be adjustments to make during the testing phase of the project.

C. Blocking Issues

1) Problems:

- The most prominent problem that has been encountered while developing the mobile application is the lack of experience in app development. Although the developer has worked on app development briefly in the past, it has taken a lot of learning during development in order to make progress. A result of this has been development that is slower than what was originally expected.

- Another problem that has slowed down the development of the mobile application was finding an Android device to test on. While it is possible to use an emulator to ensure that the code is working during development, it is very helpful to have a device that can be deployed to, and such a device was not available until later than expected. This did not significantly impact the rate of development, but it was still a small issue.

- The Bluetooth communication in an Android app was also new to the developer. This had been researched before development began, but the developer had no previous experience working with Bluetooth connection within app development.
• The last problem that has been faced during this portion of the project has been troubleshooting alone. The other members of the team have been able to help each other make progress since the other parts of the project are so closely related, but the application is very different and almost independent, the application developer has had little feedback about that portion of the project.

2) Solutions:

• The lack of experience in app development has been overcome so far by spending a significant amount of time researching how to implement the kinds of features that are needed for this application. As well as researching online for tutorials and documentation, other peers who have experience in app development have been helpful to talk to about the general process of application development.

• The lack of an Android device for testing was overcome by finding an old device that one of the group members still owned, and using it as a dedicated testing device. It does not have an updated operating system, but it still works to simply test the application.

• The lack of knowledge about how to implement the Bluetooth communication will be overcome similarly to the lack of app development experience. There is documentation online that is helpful and will be used in order to implement this part of the project properly.

• The problem of the application developer working alone on this part of the project has been overcome by regularly checking in with the other group members. This means that a simple update about where the application is in the development process suffices sometimes. Other times, it is helpful to talk to the group members about the difficulties being faced in this part of the project to see if they have any high level ideas for solutions.

D. Code Samples

```java
final ToggleButton allLights = (ToggleButton) findViewById(R.id.all_lights);
allLights.setOnClickListener(new View.OnClickListener(){
    public void onClick(View v){
        if(allLights.isChecked()) {
            String toastText = "Turning off all lights";
            mToast = Toast.makeText(getApplicationContext(), toastText, Toast.LENGTH_LONG);
            mToast.show();
        } else{
            String toastText = "Turning on all lights";
            mToast = Toast.makeText(getApplicationContext(), toastText, Toast.LENGTH_LONG);
            mToast.show();
        }
    }
});
```

Fig. 7. Sample application code that creates a pop-up when a button is pressed.

E. Photos
V. Conclusion

The project as a whole is progressing on schedule. The lamp and the companion application have been able to be successfully worked on concurrently. The two paths are approaching the point where they can be integrated together. The lamp hardware is expected to be feature complete and in beta within two weeks of this document being completed; the firmware following shortly behind. The application is on a similar schedule with initial integration testing occurring within the next two weeks. Once the pieces are all in beta, the streamlining and bug fixing will be able to again occur independently. By the end of this term the team is expecting to have produced an initial beta prototype.