CSE 237D Final Project Report

BlueRave - Light Shows for Everyone

Professor: Ryan Kastner

Team 1: Chris Taylor, Muhsin Gurel, Mohsen Imani,
Team 2: Jorge Garza Guardado, Shishuo Ding

Abstract

Most light shows that complement music require expensive equipment, long setup times, and trained professionals. Consumer grade hardware that is capable of syncing lights to music typically does not employ very advanced techniques and cannot complement the song very well. Instead of using tempo or beat onset information, most consumer hardware simply responds to large peaks in volume. Our project attempts to bridge the gap between low-cost consumer light show hardware with professional grade equipment. To achieve this, we plan use commodity RGB lightstrip hardware with a host PC that is capable of providing detailed musical timing information. Simplified setup will be achieved by the use of custom Bluetooth Low Energy hardware, which enables wireless connectivity and eliminates the need for pairing.

Introduction

Existing solutions

Currently, there is no consumer-grade light show solution. The best option for light shows uses a standard called DMX (Digital Multiplex), which is often used by professionals. Although DMX is extremely powerful and versatile, it comes with a high cost, both in terms of setup times and equipment cost. For instance, a simple DMX-capable RGB light strip controller costs $26 on Amazon. To connect it to a host machine, you would need a USB DMX converter dongle which costs upwards of $35 (according to Amazon). This still does not include any lights, power supplies, or control software.

As most control software is geared toward professionals, it is often expensive and difficult to learn. This is by virtue of the plethora of features that professional software is required so that it will be useful is a variety of situations. For ordinary consumers who just want to set up a light show for a house party, the complexity of features is often prohibitive. For instance, LightJams, a fairly well-featured DMX control program, costs $200.

Currently, consumer grade music hardware consists of purely hardware solutions which simply respond to ambient sound (example). These generally do not complement multilayered audio tracks very well. For instance, an EDM (electronic dance music) track might consist of bass drums, bass leads, higher pitch synthesizers, vocals, and other percussion. Hardware might simply respond to whichever onset it hears first even though one track might be more prominent than the next. In addition, the lights should respond to the mood of the track. A track might be in a calm period in which the lights should display very smooth patterns and not respond to every single beat. This is way too subjective for hardware.

Our Target

Currently, consumers are offered expensive and difficult solutions that sync to music incredibly well or cheap pure-hardware solutions that barely sync at all. BlueRave targets the middle ground and tries to get the best of both worlds. We aimed for the following constraints:

- **Ease of setup.** Other than lights, everything about the system should reside on a user's PC, which people are likely to already have. The only thing users should have to worry about is placing the lights. To achieve this, we settled on Bluetooth Low Energy. This allows for lights to be placed out of reach of the host PC, does not require running wires to the host, and allows for really fast setup times. Bluetooth LE is an incredibly lightweight protocol and host machines can connect to peripherals extremely fast without the pairing rituals required by Bluetooth 3.0 and WiFi. Bluetooth LE, unlike other ad-hoc protocols like Zigbee, is available on most modern mobile devices and has a very permissive license.
- **Complementing music well.** The processing power of the host machine should do the heavy lifting of deciding what color to send to the lights. It should know the BPM of the song and where the first bar of music information starts. Users should be able to customize how the lights sync by using a simple GUI.

- **Low cost.** The receiving end of the host PC should be very low cost. We achieved this by doing writing custom firmware for a Bluetooth LE chip so that everything is handled by this one chip.

Software

Team 1 handled the software. We created a GUI using WPF (Windows Presentation Foundation), an API created by Microsoft for creating modern looking interfaces for the PC. All of the core code is in C#. The GUI does all the work of playing the music and deciding what color the lights should be at any given time, with some help from the user.

Below is a screenshot of our GUI. The user can edit patterns ahead-of-time for the song or attempt to create them in real-time.

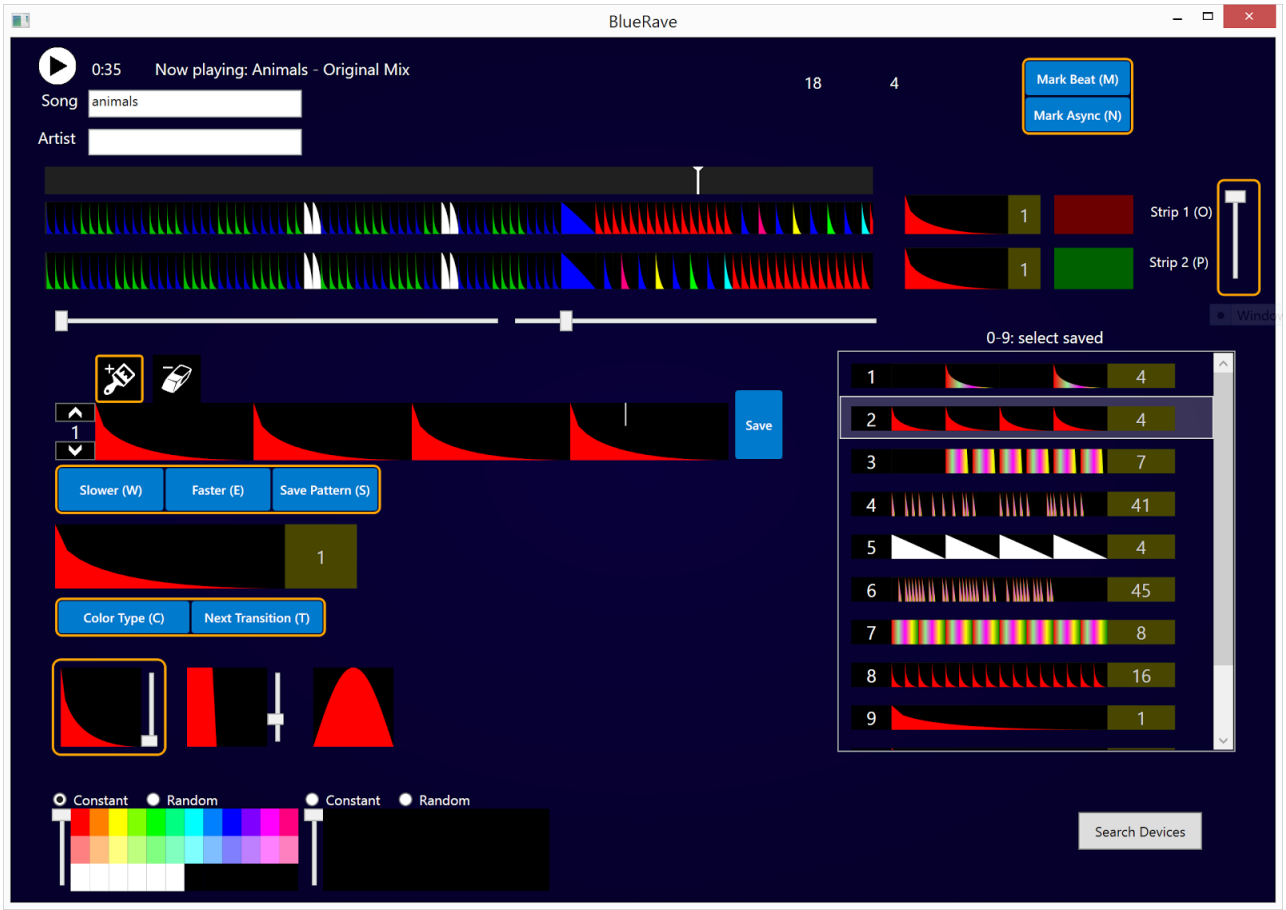

Figure 1. GUI for User Inputs

From the top, the features shown include:

- **Spotify song search**. The results are songs which are on both Echonest and Spotify.
- **Beat marking.** Since Echonest isn't perfect, you sometimes need to hit a key when you hear bar one of a song to get the syncing right.
- **Timeline.** A timeline of patterns with a playhead showing the song progress.

- **Light strip indicators.** The rectangles next to “strip 1” and “strip 2” give a real-time indicator of what color is currently being sent to the strips.
- **Pattern editor.** Here you can chose the transition type (currently 3 choices) and the colors.
- **Saved patterns.** On the right is a list of saved patterns. You paint these on to the timeline.

Hardware

For the Hardware part of the project we developed a custom PCB that is shown below, together with our own firmware code for the CC2541 Bluetooth Low Energy Integrated Circuit from Texas Instruments.

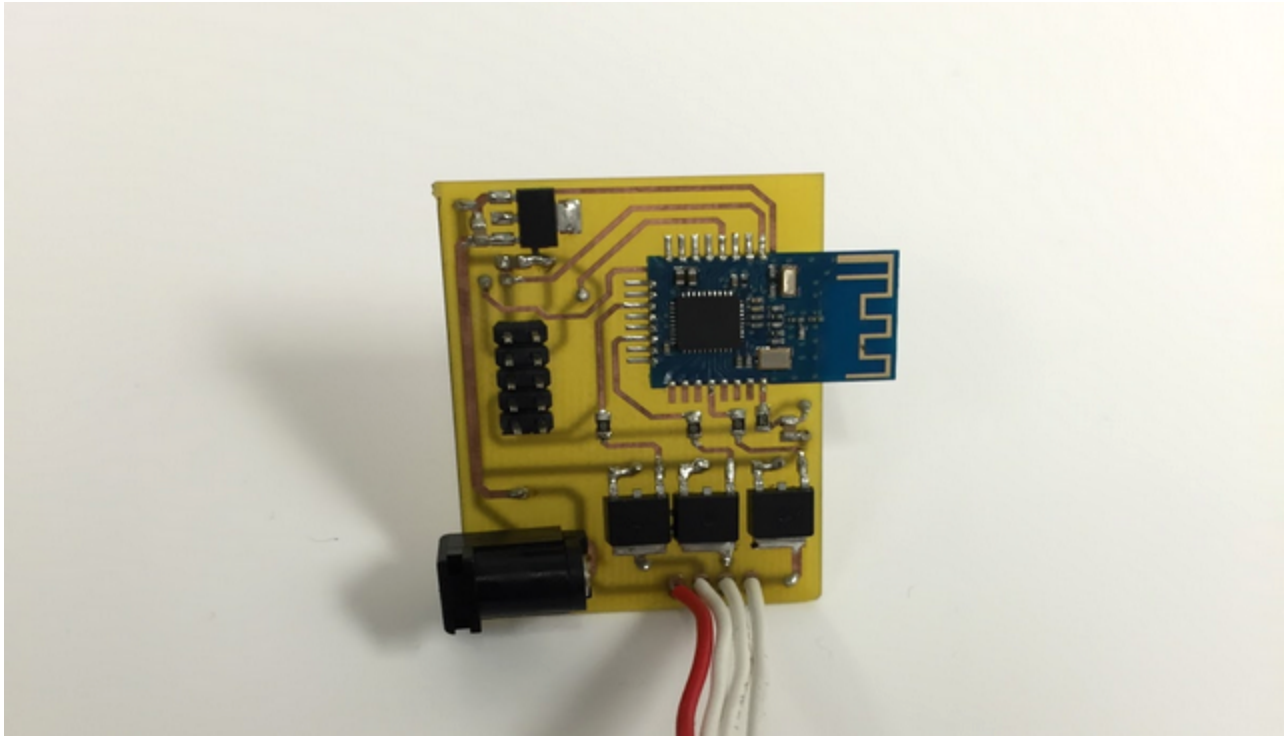

Figure 2. BLE RGB Controller Hardware

For now, communication from the host to the hardware requires a USB to BLE dongle (BLED112) from Bluegigia. We hope to use the machine’s native BLE hardware in the future.

The hardware development then can be divided as follows:

- BLE Receiver/Control Module with CC2541 (Firmware)
- PCB Schematic and Layout
- GUI for Testing communication and Connection

BLE Receiver/Control Module with CC2541

For the purposes of our project, we needed to provide a hardware service to enable Team 1’s GUI to communicate with the RGB light strips in a fast and reliable manner. Initially, researched

off-the-shelf components but we eventually settled on the CC2541 chip from Texas Instruments that uses Bluetooth Low Energy (BLE) 4.0 because it of cost.
The firmware was programmed using an IDE from IAR Systems because it is the only IDE that can program this chip. The firmware runs a RTOS (Real Time Operating System) and we added a task to this RTOS that adds a Descriptor to the BLE Services. When sending the RGB values to this descriptor the code inside captures the data and writes the values to PWM (Pulse Width Modulation) for each color Red, Green and Blue. The following image show the descriptor of the BLE module where the selected one is the one that we added.

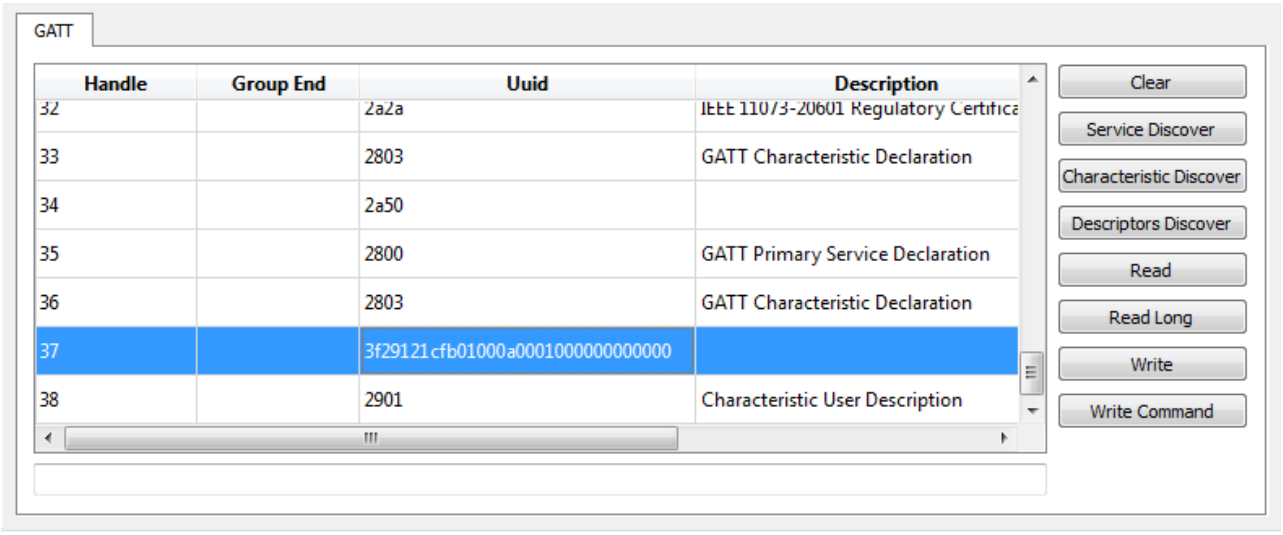

Figure 3. Descriptor of BLE Modul

As we made further progress in the project we came to realize that we needed to make our own custom PCB aside from the commercial components. The custom PCB is described in subsequent sections.

PCB Schematic and Layout

For the schematic of this design we used the following parts:

Part Number in Schematic and Layout	**Description**
J1	CONNECTION POWER JACK 2.1MM
IC1	IC REG LDO 3.3V 0.8A SOT223
U$1	Bluetooth Low Energy Module with the CC2541 IC
Q1,Q2, Q3	AOD4130, MOSFET N-CH 60V 30A TO252
C1, C2, C3	1nF, 0.1uF, 10uF and 1nF Capacitors
R1, R2, R3, R4	2.7K, 100 Ohms Resistors

JP3, Con	10 pin and 4 pin connectors for the programmer and the RGB Output

Table 1. Schematic and Layout Components

Figure 4 shows the schematic for the custom PCB, created using Eagle Design Suite. We had to include a 10-pin connector so that we could program the CC2541 chip initially, as the physical programmer module uses a 10-pin connector. Also, in addition to the CC2541 module, we included a power jack for the external power supply and a power regulator that convert the external 12V voltage to the 3.3V voltage that the RGB lightstrips use.

Figure 5 shows the layout we made in Eagle Design Suite. The version depicted here is our 2nd revision, as our first design didn't conform to the capabilities of the milling machine that our group used to manufacture the board ourselves. The second version functions correctly.

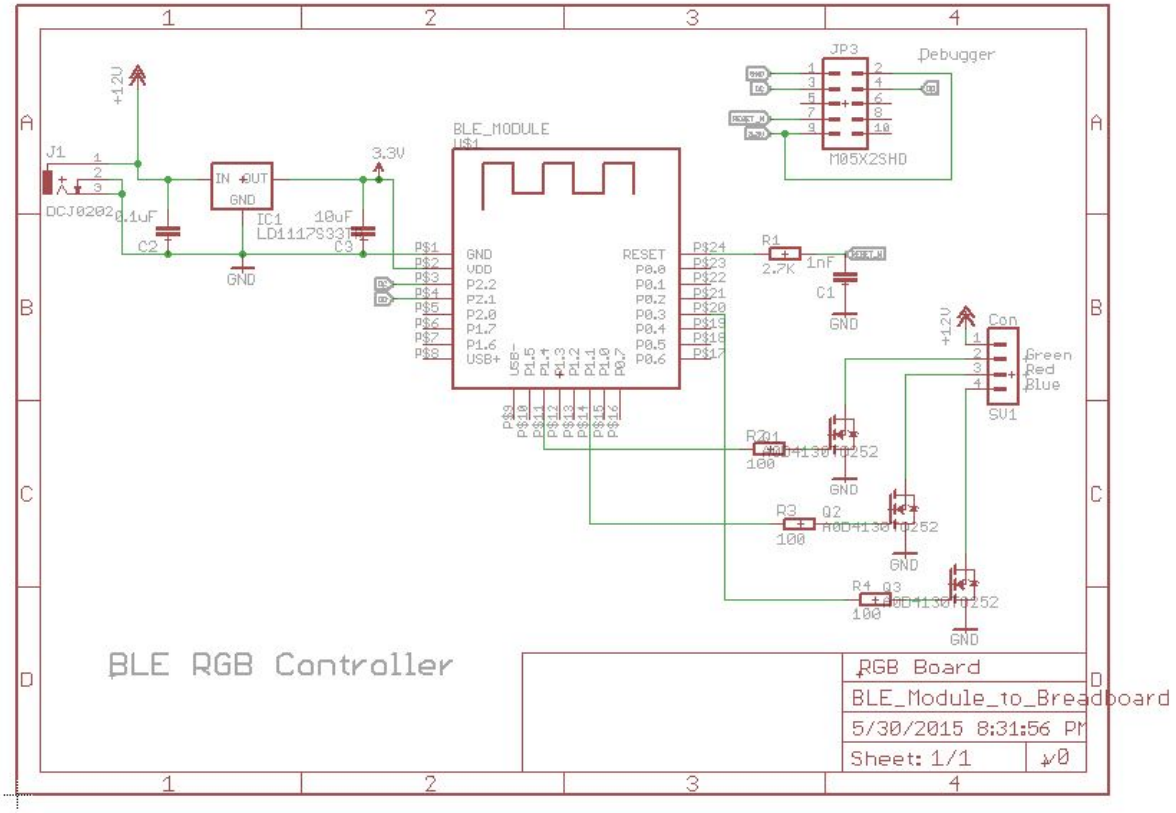

Figure 4. Schematic for the Custom PCB

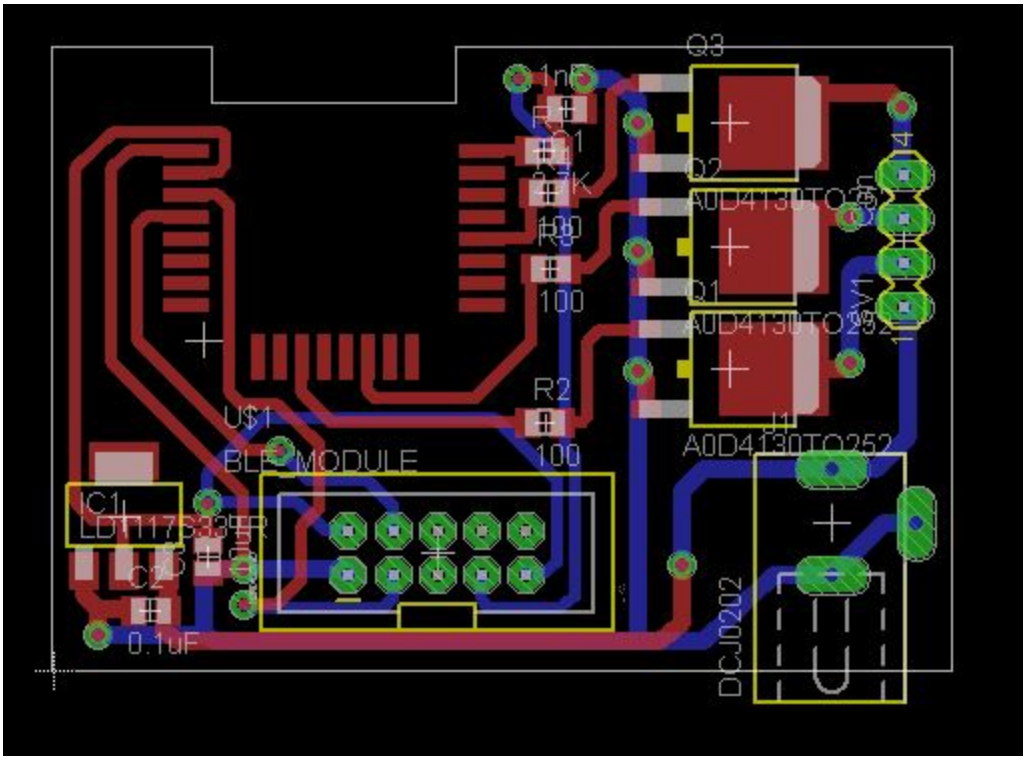

Figure 5. Layout for the Custom PCB

Custom GUI for Testing

The final goal for Team 2 was to provide a communication service to Team 1's software. Before we attempted to integrate the BLE hardware with the GUI software from Team 1, we did some testing on our custom PCB hardware. We created a simple GUI to test the basic communication over BLE and the control of a simple RGB LED (instead of the actual lightstrips).

To test communication over BLE, the software uses the Bluetooth dongle plugged into the host PC and searches for Bluetooth devices. We assigned the specific names "LedControllerP" to our Bluetooth receiver devices. Figure 6 shows our tests for establishing communication with two BLE devices.

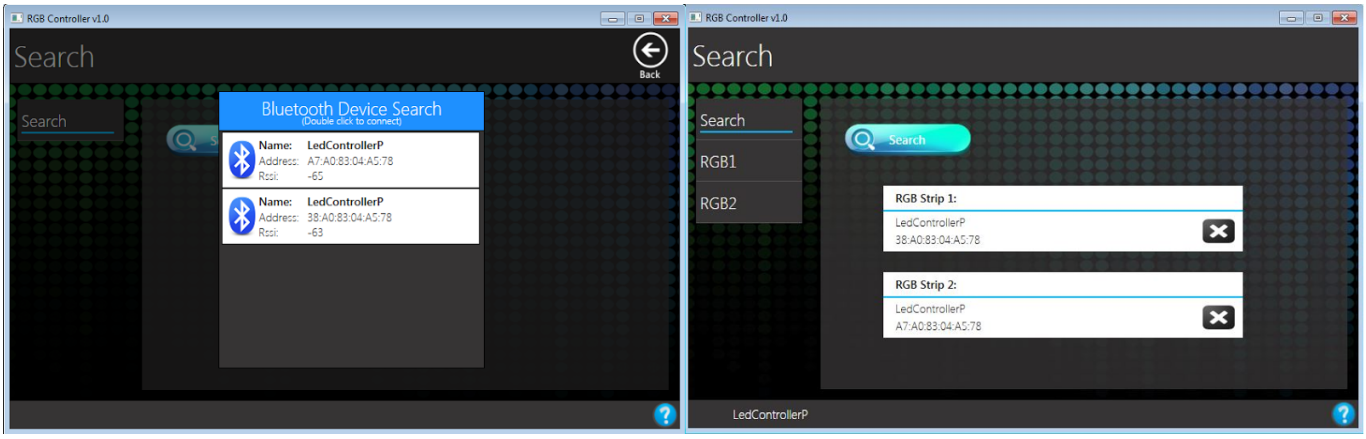

Figure 6. Test GUI Establishing Connection with 2 BLE Devices

After we establish communication with the 2 BLE receivers (CC2541 modules) on our custom PCB, we connected a simple RGB LED to our PCB and performed tests of changing the intended color in the test GUI and send the control signals over BLE to the LED. Figure 7 and 8 depicts our tests that changed the first LED to color Red and the second to color Blue.

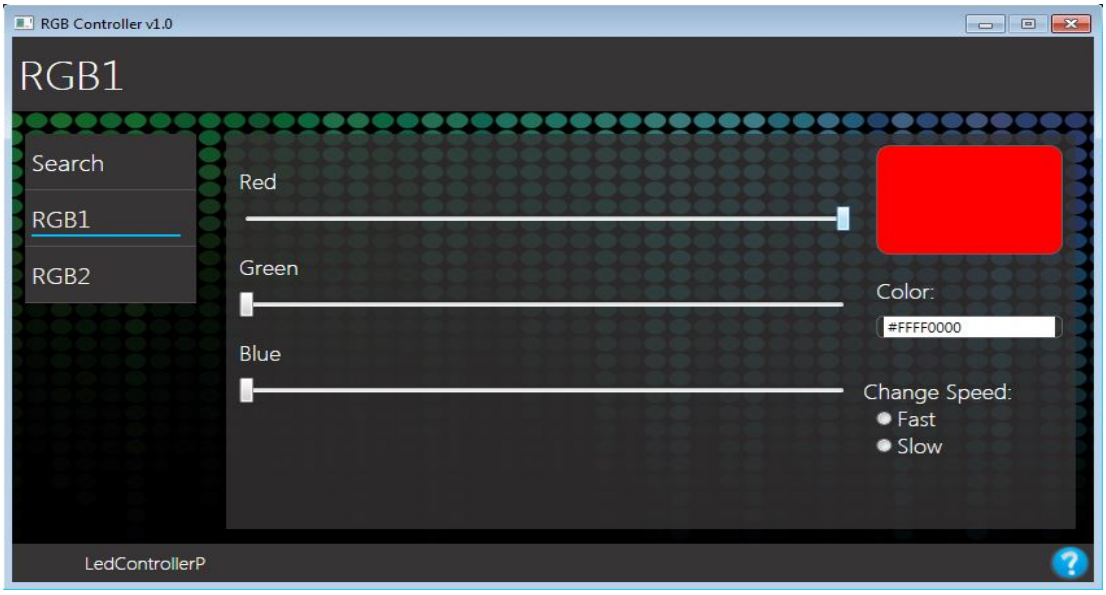

Figure 7. Test GUI Changing First LED to Color Red

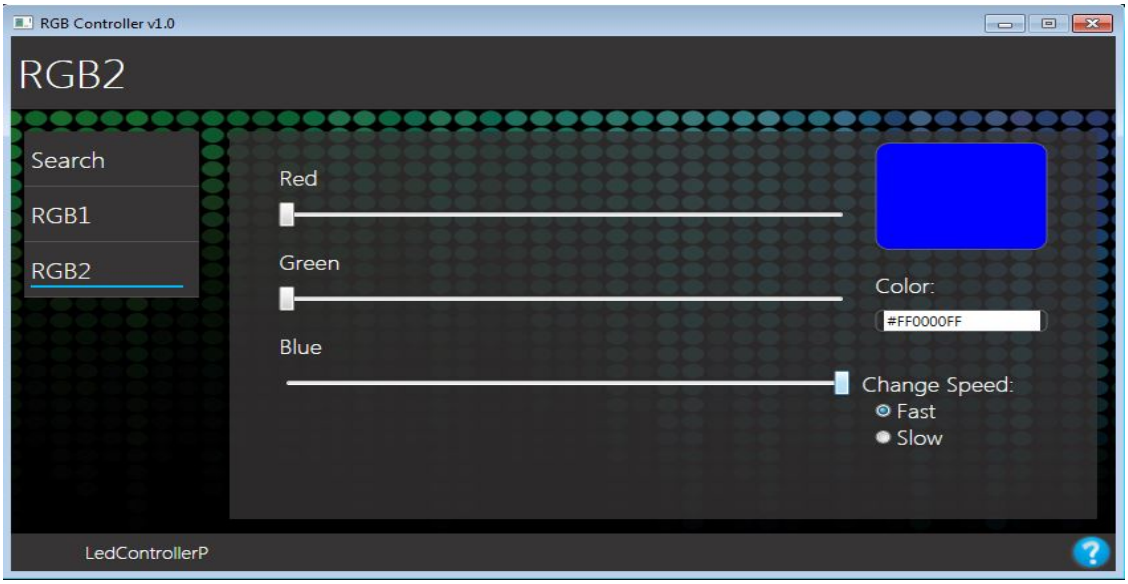

Figure 8. Test GUI Changing Second LED to Color Blue

The test GUI software only takes simple inputs for RGB control signals and only controls a single LED on each receiver, but it successfully tests the essential functions we require of it, which is fast and reliable communication over BLE.

Upon finishing these tests, we were ready to integrate the custom PCB with the GUI for User Control made by Team 1, and also the LED lightstrips. Comparing to the simple Test GUI, the integration presented two complications: faster and much complicated user input signals, and the whole LED lightstrips instead of a single RGB LED for testing. The first one was tested and later proven that the BLE hardware can successfully transmit patterns as fast as up to 32 beats with 2 lightstrips connected at the same time. The second one did not require changing the hardware as the lightstrips are constructed so that all the LEDs in the lightstrips are connected in parallel, and can be controlled in the same manner as the single LED we used in our testing.

Figure 9 depicts the integration of the User Control GUI by Team 1 and BLE communication by Team 2. As can be seen, when the user hits “Search Devices” button, he/she is able to select and establish communication with the BLE receivers on our PCBs that are connected to the lightstrips, before going on to perform control activities such as choosing songs and changing patterns.

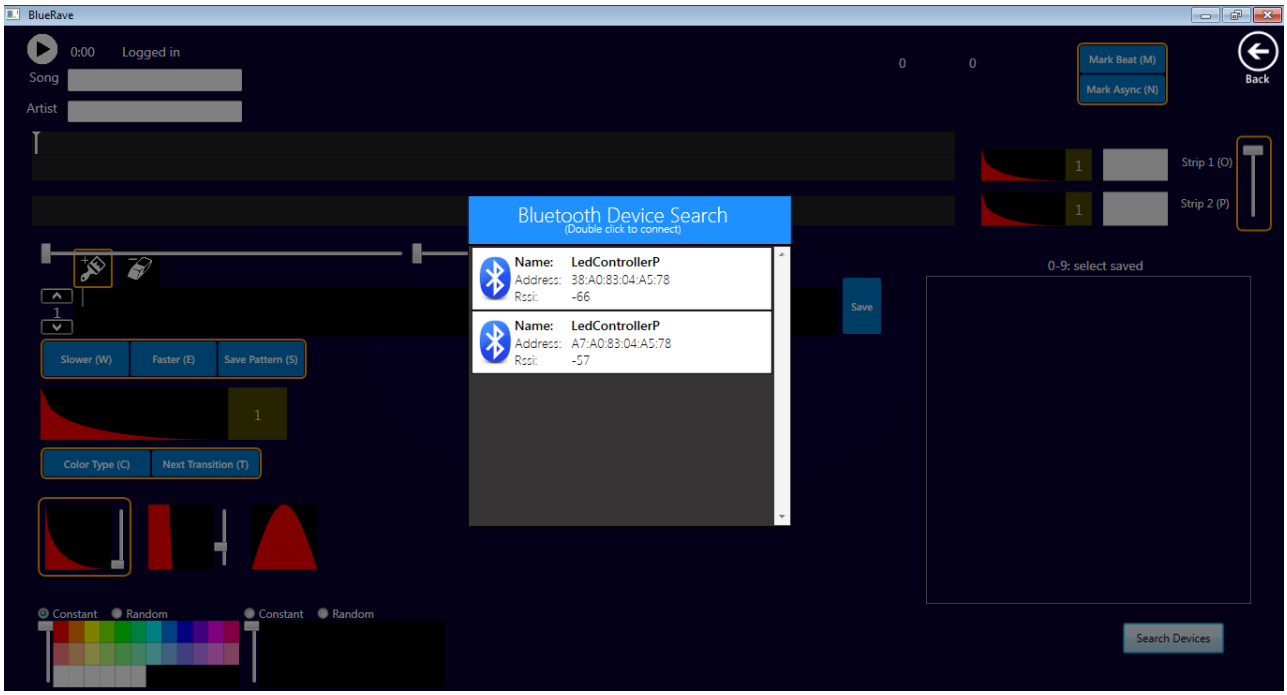

Figure 9. BlueRave GUI Showing BLE Devices Search.

Milestones

- **Accurately find the bpm of a song.** We modified a program called bpm-tools (in our repo) to deliver very accurate bpm information. Originally, it used random sampling to deliver a quick-and-fast estimate of bpm. We made it exact.
- **Learn Windows Presentation Foundation (WPF) to make our UI.** We all read the docs and made some simple hello world apps. None of us have used WPF before but it wasn’t too bad.

- **Display patterns in the UI.** We created a graphical way to see the entire light pattern at once.
- **Sync the UI with a song.** A separate thread in our UI handles all timing related information and we took extra measures to ensure that it would not lose sync if the program lags.
- **Talk to Bluetooth hardware.** We successfully combined our code with Team 2's BLE (Bluetooth Low Energy) code. Our UI can wirelessly communicate with a test board and change the color of an RGB LED in sync with the song. However, it has some issues when the light pattern changes too fast. But for the purposes of our application the communication is fast enough.
- **Spotify.** We can stream any Spotify song through our UI. That means our UI can play any song we want (if it's on Spotify).
- **Echonest.** We can get very accurate BPM and onset information through Echonest, a huge database of songs. The bar information is not very accurate sometimes so the user will still sometimes be required to hit a key to mark beat one.
- **Dynamic Color Changer.** We can define/change the color of the light strips instantly by clicking/touching to the color spectrum on the UI.
- **Save Patterns on a timeline.** We added a timeline on the GUI. This timeline shows us the progress of the song also now we can place the patterns on the timeline and prepare a perfect pattern combination beforehand

Milestones that didn't go so well

- **Bass detection.** Before we started using Echonest, we wanted to have our software automatically make a "bass" pattern which simply flashes a color on every bass note. This was difficult and required solving the problem of "onset detection", the start of a musical note. Echonest provides some onset information about songs so this might be possible in the future.
- **Real time control.** Originally, we wanted the user to be able to control the lights for any song in real-time, without having to set anything up beforehand. Our UI somewhat does this but it turned out to be pretty difficult. Toward the end of the project we shifted our focus toward ahead-of-time lightshow creation.

Conclusion

Figure 9. Parties dog.

Future Work

Here are our ideas for possible features of BlueRave.

- **Songs with odd timing.** Currently, only songs of constant tempo and 4/4 time signature are supported.
- **Wearables.** The receiving hardware is simply Bluetooth LE, so instead of controlling lightstrips it could be RGB lights on an article of clothing.
- **Smart Watches.** In future, people can use their smart watches as a light pattern source during parties, either through the touch screen or motion sensor. In addition, their watches can light up as another light source.
- **MIDI Controllers.** You would be able to record notes in real-time with a MIDI drum pad or adjust lights using knobs.
- **Note onsets.** Echonest does give you some information about note onsets, but they can come from any layer in the audio track. Using a simple linear classifier it might be possible to pick out only onsets that are of a certain type, such as bass notes or claps.

References

[1] BLE Module with CC2541 "http://www.amazon.com/gp/product/B00PI6PZ5G"
[2] BLED112, "Bluetooth Smart Dongle", Bluegiga
"https://www.bluegiga.com/en-US/products/bled112-bluetooth-smart-dongle/"