Christmas Lights Animation

System Design Document | Current Version 1.1.0

Prepared By:

Austin Wentz

Jordan Doell

Revision History

Date	*Author*	*Version*	*Comments*
9/12	*Austin Wentz*	*1.0.0*	*Initial version*
12/11/12	*Austin Wentz*	*1.1.0*	*Fleshed out content*

Table of Contents

1.0 Overview

The overview should take the form of an executive summary. Give the reader a feel for the purpose of the document, what is contained in the document, and an idea of the purpose for the system or product.

1.1 Scope

This document covers the design specifications of the XMASLA project.

1.2 Purpose

To make an interactive Christmas lights animation product. It will be controllable from an iOS device.

1.2.1 Hardware

- Raspberry Pi
- Renard 64XC
- SSRez solid state relays

1.2.2 iOS App and Device

Describe briefly the role this major component plays in this system.

1.2.3 Miscillaneous

- Christmas lights
- Extension cords
- Display case

1.3 Systems Goals

- Choose a song from the iPhone app
- Play music and have the Christmas lights synced to the music
- Make sequences on the iPhone app to have a fully interactive experience

1.4 System Overview and Diagram

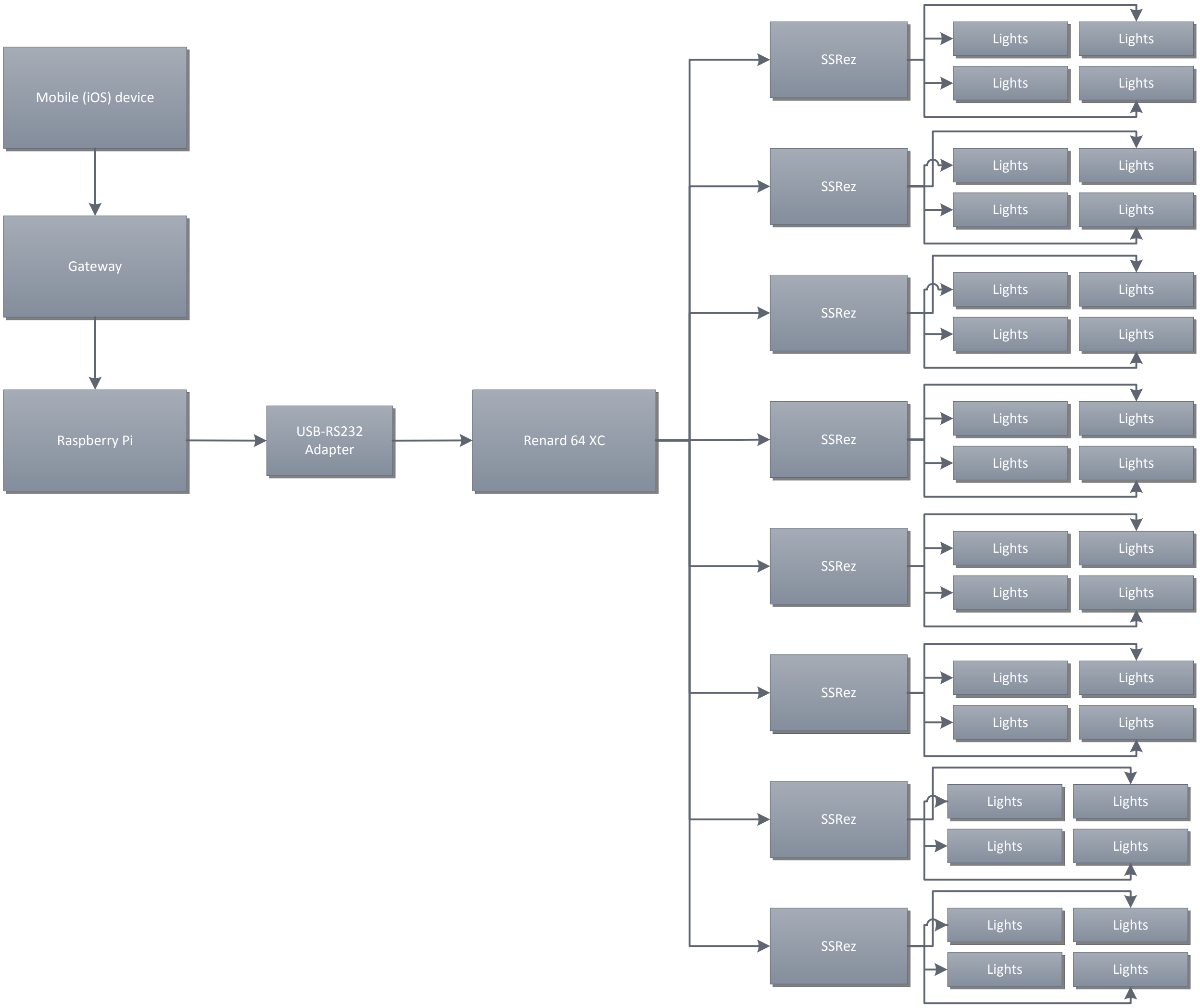

Figure 1 System Diagram

1.5 Technologies Overview

2.0 Project Overview

2.1 Team Members and Roles

- Jordan Doell – iOS development / front end
- Austin Wentz – Hardware / back end

2.2 Project Management Approach

The project is managed using the Agile methodology Scrum. The Scrum Master is Dr. Jeff McGough. Sprints are 3 weeks in length and weekly meetings are held. Trello is used for managing the backlog.

2.3 Phase Overview

2.4 Terminology and Acronyms

- SSR – Solid State Relay
- Renard – PIC-based dimming controller used to animate Christmas lights

3.0 Requirements

List, describe, and define the requirements for the system. These requirements are mostly provided by the stakeholders. However, this section should contain details about each of the requirements and how the requirements are or will be satisfied in the design and implementation of the system.

There could be any number of sub-sections to help provide the necessary level of detail.

4.0 Design and Implementation

This section is used to describe the design details for each of the major components in the system. This section is not brief and requires the necessary detail that can be used by the reader to truly understand the architecture and implementation details without having to dig into the code.

4.1 Hardware

4.1.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.1.2 Component Overview

This section can take the form of a list of features.

4.1.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.1.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.1.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.1.6 Design Details

This is where the details are presented and may contain subsections.

4.2 iOS app and Device

4.2.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.2.2 Component Overview

This section can take the form of a list of features.

4.2.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.2.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.2.5 Data | Logic Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.2.6 Design Details

This is where the details are presented and may contain subsections.

4.3 Miscellaneous

4.3.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.3.2 Component Overview

This section can take the form of a list of features.

4.3.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.3.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.3.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.3.6 Design Details

This is where the details are presented and may contain subsections.

5.0 System and Unit Testing

This section describes the approach taken with regard to system and unit testing.

5.1 Overview

Provides a brief overview of the testing approach, testing frameworks, and general how testing is/will be done to provide a measure of success for the system.

5.2 Dependencies

Describe the basic dependencies which should include unit testing frameworks and reference material.

5.3 Test Setup and Execution

Describe how test cases were developed, setup, and executed. This section can be extremely involved if a complete list of test cases was warranted for the system.

6.0 Development Environment

The basic purpose for this section is to give a developer all of the necessary information to setup their development environment to run, test, and/or develop.

6.1 Development IDE and Tools

Describe which IDE and provide links to installs and/or reference material.

6.2 Source Control

Github is used for source control.

6.3 Dependencies

Describe all dependencies associated with developing the system.

6.4 Build Environment

How are the packages built? Are there build scripts?

6.5 Development Machine Setup

If warranted, provide a list of steps and details associated with setting up a machine for use by a developer.

7.0 Release | Setup | Deployment

This section should contain any specific subsection regarding specifics in releasing, setup, and/or deployment of the system.

7.1 Deployment Information and Dependencies

Are there dependencies that are not embedded into the system install?

7.2 Setup Information

How is a setup/install built?

7.3 System Versioning Information

How is the system versioned?

8.0 End User Documentation

This section should contain the basis for any end user documentation for the system. End user documentation would cover the basic steps for setup and use of the system. It is likely that the majority of this section would be present in its own document to be delivered to the end user. However, it is recommended the origin is contained and maintained in this document.

Appendix I: List of Figures

Appendix II: Supporting Information and Details

This document will contain several appendices used as a way to separate out major component details, logic details, or tables of information. Use of this structure will help keep the document clean, readable, and organized.

Appendix III: Progress | Sprint Reports

This section will contain a complete list of all of the period progress and/or sprint reports which are deliverables for the phases and versions of the system.

III.1 Sprint 1 Progress Report

Sprint Report 1

Team Members:	Austin Wentz and Jordan Doell
Date:	October 5, 2012
Class:	Senior Design
Subject:	Sprint 1 Report
Sponsor:	L-3: June Alexander-Knight

Sponsor Description:

L-3 Communications is a world class defense contractor. They play a huge role in the defense industry for the United States government. June Alexander-Knight graduated from SDSMT and since then, works for L-3. She has also been a strong supporter of SDSMT students and graduates.

Sponsor's Problem/Goal:

Sync Christmas lights to music using a Linux board and controller, and control the system using an iPhone app.

Customer Needs:

- Linux board to control lights
- SSR's to power on and off the strands of lights
- iPhone app to do sequences or play music
- Use sequencer software to program light show

Project Environment

Project Boundaries

- The project will have two separate environments: mobile device environment and Christmas lights controller environment

- The project’s mobile environment will be focused on iOS devices
- The project’s controller environment consists of Raspberry Pi, PIC microcontrollers, and additional circuitry to control the lights
- Communication between environments will be done over TCP/IP via JSON
- The mobile environment will be developed in Objective-C
- The controller environment will be developed in Python, and also in Clojure
- No code will need to be written for the PIC microcontrollers

Project Context

Technical Environment

The technical environment can be split into three parts: mobile device, high-level controller, and low-low-level controller.

Mobile Device

The iPhone is used as the mobile device. Development will be done on a Mac mini.

High-Level Controller

The Raspberry Pi is used as a high-level controller. It will receive commands from the mobile device, perform any required processing on command data, and send the commands to the low-level controller. The Raspberry Pi uses a Debian-like flavor of Linux. Development will be done in Linux and Windows.

Low-Level Controller

To directly control the Christmas lights, we are using a popular do-it-yourself light dimmer scheme called Renard. In particular we are using the Renard 64 XC design. No development needs to be done on the low-level controller.

Current Systems Overview

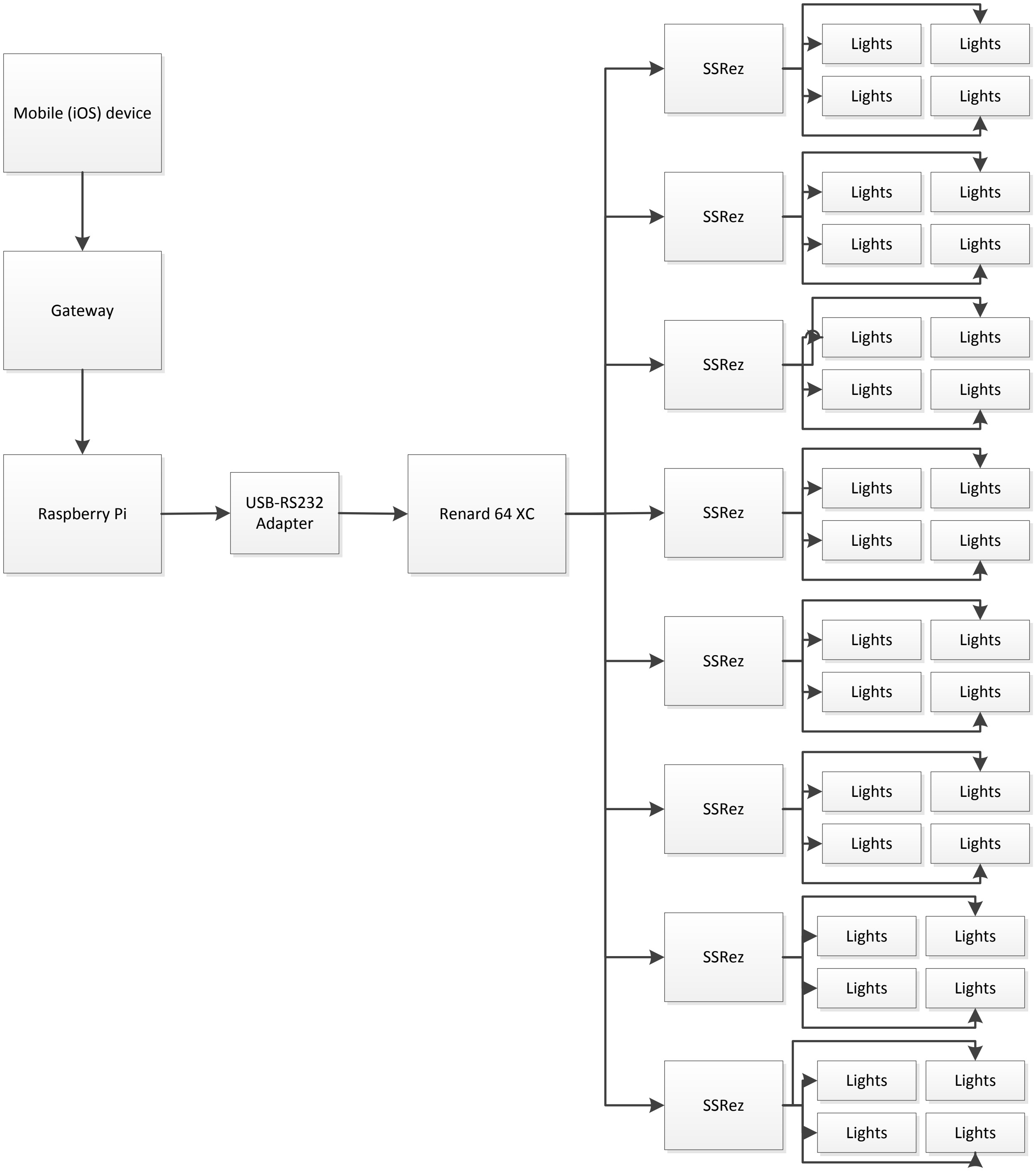

Figure 2: System Overview

Product Deliverables

No product deliverables at this point.

Future Product Deliverables

- Functional prototype
- Source code
- User manual / documentation
- Requirements document
- Design document

Backlog

Completed

- Purchase and configure single board computer (SBC) to act as high –level controller
- Purchase SSR pcb kit, SSR heat sinks, and Renard microcontroller pcb kit
- Analysis and research for design and requirements for project
- Start learning iOS development

Remaining

- Develop interface between Raspberry Pi and Renard 64XC
- Implement Renard serial protocol
- Develop prototype which switches lights on and off using predefined sequence
- Assemble additional circuitry (SSR and Renard kits)
- Purchase Christmas lights
- Purchase extension cords
- Program and configure Raspberry Pi to act as midi sequencer for lights
- Develop and implement iPhone app which controls the Christmas lights

Potential Issues

- Difficulty in assembling additional circuitry correctly
- Troubleshooting issues with SSRs and Renard microcontroller
- Safety issues when dealing with high voltage power sources
- Possible issues with iOS development

III.2 Sprint 2 Progress Report

Sprint 2 Report

Team Members:	Austin Wentz and Jordan Doell
Date:	November 1, 2012
Class:	Senior Design
Subject:	Sprint 2 Report
Sponsor:	L-3: June Alexander-Knight

Backlog

Completed

- Purchase and configure single board computer (SBC) to act as high –level controller
- Purchase SSR pcb kit, SSR heat sinks, and Renard microcontroller pcb kit
- Analysis and research for design and requirements for project
- Start learning iOS development
- Assemble additional circuitry (SSR and Renard kits)
- Implement Renard serial protocol
- Purchase Christmas lights
- Purchase extension cords
- Develop prototype which switches lights on and off using predefined sequence
- iPhone app prototype

Remaining

- Design display case for electronic components
- Have the display case made and assembled.
- Program and configure Raspberry Pi to act as midi sequencer for lights
- Develop and implement iPhone app which controls the Christmas lights

iOS Application progress:

Jordan Doell

During Sprint 2, I have been continuing to learn Objective-C and iOS application development. I found and have been watching a podcast that covers iOS development and Objective-C. Also, James has been lecturing to me and Josh about iOS and some of the components we will need for the project. We still have a few more lectures to go, but we are making progress.

App Prototype:

I have gained enough knowledge of iOS so far to make a simple prototype. It is nonfunctional so far, but gives a little direction to where we are headed with the app. Below are some screenshots of the different views in the app.

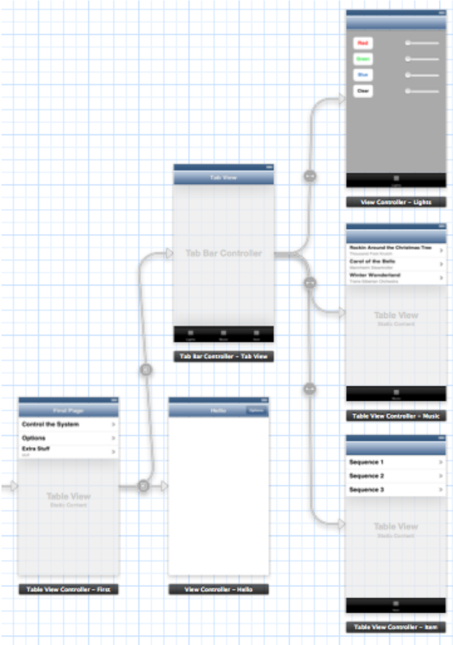
Tab View
Tab Bar Controller
Tab Bar Controller – Tab View
View Controller – Lights
Table View Controller – Music
Table View Controller – Item
Table View Controller – First
View Controller – Hello
Table View

Fig. 1: Overall storyboard for the prototype

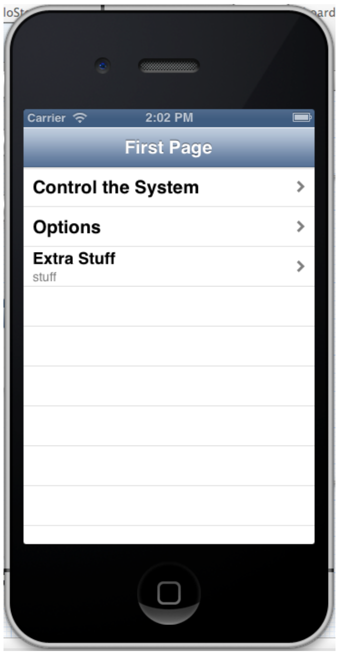

Fig. 2: Main page of the app

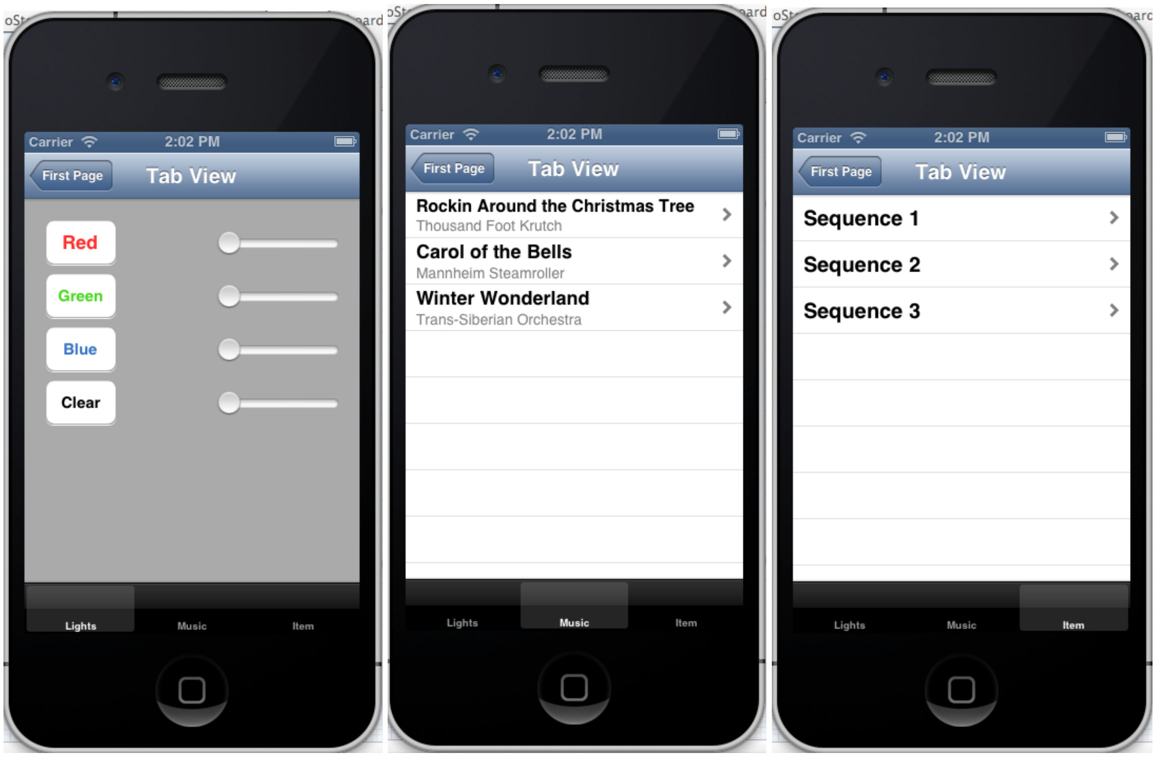

Fig. 3,4,5: Lights tab, Music tab, and Sequences tab

Christmas Light Controller Progress

Austin Wentz

Considerable progress has been made on the hardware front. The Renard 64XC and the 8 SSRez's are now soldered and thoroughly tested. In total, the soldering took 20-25 hours. Testing took another 5 hours to complete. With the hardware assembled, I put together a simple prototype which turns lights on and off using a predefined sequence. Several short videos are available to demo the prototype.

Display Case

I have also been working on a design for a display case which houses the hardware. The dimensions of the case will be 16.5 inches x 16.5 inches x 12 inches. Here are some initial requirements for the case:

- Safety features – Renard 64XC and SSRez's will only be powered when lid is closed.
- Locking mechanism to prevent theft
- Made of acrylic
- Fan for keeping SSR's cool
- Cord management