Sprint 1 Report

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

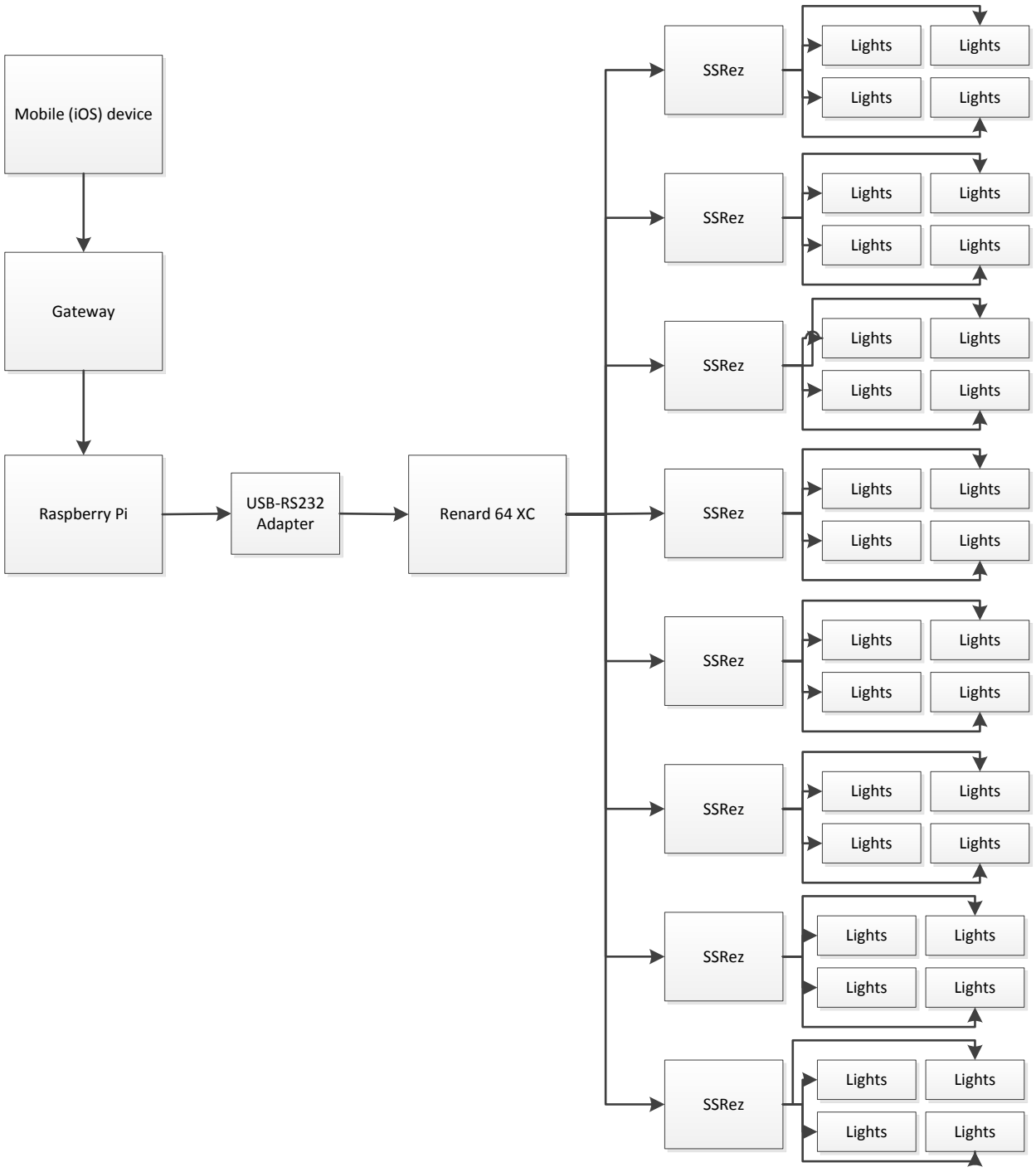

Figure 1: System Overview

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