Daynight Panel

Group 2

Drew Haas	Computer Engineer
Jordan Larsen	Electrical Engineer
Ornella Ngalamulume	Electrical Engineer
Ricardo Rivera-Matos	Electrical Engineer

1. Executive Summary.....3 Pages
2. Design Motivations and Marketing .....10 Pages
2.1. Motivations
2.2. Trade-Off Table (House of Quality)
2.3. Block Diagrams
2.3.1. Hardware Block Diagram
2.3.2. Software Block Diagram
3. Research and Parts Specification .....46 Pages
3.1. Lighting
3.1.1. Typical Conditions
3.1.2. Lighting Technology
3.1.3. Potential Products
3.1.4. Light Diffusion
3.2. LED Driver
3.2.1. Why Use an LED Driver?
3.2.2. LED Driver Selection Process
3.2.3. TLC5955
3.2.4. STP16CPC05
3.2.5. TLC59116
3.3. Power Supply
3.3.1. Connector Selection
3.3.2. Internal versus External Power Supply Unit
3.3.3. Off-the-shelf versus Custom Designed Power Supply Unit
3.4. Voltage Regulators
3.4.1. Linear Regulators
3.4.2. Switching Regulators
3.5. Communication
3.5.1. Microcontroller Master
3.5.2. Modes of Communicating
3.6. Peripheral Device Requirements and Capabilities
3.6.1. TLC59116 – LED Driver

3.6.2. Bluetooth LE Module
3.7. STP16CP05 – 16 Bit Shift Register
3.8. Wireless Networking
3.8.1. Bluetooth's Role in Networking
3.8.2. Wi-Fi's Role in Networking
3.9. Finding the Right Microcontroller Hardware
3.10. Finding the Right Software
3.11. Programming Languages for iOS
3.11.1. Core Bluetooth
3.11.2. CBCentral and CBCentralManager
3.11.3. CBPeripheral and CBPeripheralManager
3.11.4. Scanning for Advertising Peripherals
3.11.5. Transmitting Data Over Bluetooth
3.12. Model-View-Controller Software Architecture Pattern
3.13. Lighting Configurations in Software
3.13.1. Memory-Based Configuration
3.13.2. Calculation Based Lighting Configuration
3.13.3. Memory Based vs. Calculation Based
4. Project Standards .....9 Pages
4.1. Lights Standards
4.1.1. Minimum Lighting Standard
4.1.2. LED Footprint
4.2. Communication Standards
4.2.1. Serial Peripheral Interface (SPI)
4.2.2. Inter-Integrated Circuit (I2C)
4.2.3. UART
4.2.4. Bluetooth Low Energy (LE)
4.3. Power Standards
4.3.1. National Electrical Manufacturers Association Connectors
4.4. Printed Circuit Board Standards
4.4.1. Institute for Printed Circuits Standards

4.4.2. IPC-2221 Conductor Width Standard
4.5. Electromagnetic Interference Standards
4.5.1. Federal Communications Commission Part 15
4.5.2. ANSI C63.4
4.6. Environmental Safety Standards
4.6.1. Restriction of Hazardous Substances (ROHS) Compliance.
4.7. Design Standards
4.7.1. iOS Client Software
4.8. Microcontroller Software
4.8.1. Standard Semiconductor Packaging and Footprints
5. Project Design .....19 Pages
5.1. Specifications and Features of Note for ATmega328P
5.2. TLC59116 Functional Diagram
5.2.1. TLC59116 Features – Brightness Control
5.2.2. TLC59116 Features – Current Source
5.2.3. TLC59116 Features – Error Monitoring
5.2.4. TLC59116 Communication
5.3. iOS and Bluetooth
5.3.1. The Client-Server Model in Software
5.3.2. An Introduction to GATT Profiles, Services, and Characteristics
5.4. ATmega328P and C
6. Panel Component Integration.....27 Pages
6.1. Compatibility Testing
6.1.1. LED Testing
6.1.2. Warm (2700K) LED
6.1.3. Cool (6500K) LED
6.1.4. Blue LED
6.1.5. Red LED
6.1.6. Green LED
6.1.7. LED Driver Functionality Test
6.1.8. Power Supply Functionality Test

6.1.9. Voltage Regulator Functionality Test
6.2. Project Prototyping
6.2.1. LED Color Balancing
6.2.2. Prototype Daynight App for iOS
6.2.3. Prototype Software Flowchart Diagrams
6.2.4. Using the UART GATT Service and GATT Characteristics
6.2.1. Prototype Microcontroller Code
6.2.2. Electronic Design Automation
7. Project Operation.....4 Pages
7.1. Getting Started with the Daynight Panel
7.2. Daynight Panel Operation
7.2.1. Select a New Color
7.2.2. Select a New Creative Vista
7.3. Safety Warnings
8. Vendors and Personnel .....2 Pages
8.1. LED Vendors
8.2. Integrated Circuit Sampling
8.3. Diffuser Panel
8.4. Printed Circuit Board Manufacturing
8.5. Adafruit
8.6. Arduino
9. Administrative .....4 Pages
9.1. Budget and Financing
9.2. Project Milestones
9.2.1. Semester 1 Milestones
9.2.2. Semester 2 Milestones
10. Final Remarks .....1 Page
11. Appendix A – Copyright Permissions
12. Appendix B – Data Sheets
13. Appendix C – Software
14. Appendix D – Other

15. Works Cited

Total Estimated: 125 Pages

1. Executive Summary

The Daynight Panel is a panel designed to improve the internal ambient conditions of a room by illuminating it with artificial daylight or moonlight. The panel is designed to provide a relief for the eyes and provide the inhabitants of a room with the warmth and relief of being connected to the outdoors while they are inside. There are many devices that provide this service, but the aim of this project is to make the panel affordable, configurable, scalable and relatively easy to install.

The Daynight Panel is a system that simulates outdoor lighting by using a grid of LEDs behind a light diffuser. The LEDs provide a source of light by emitting exact, calculated measurements of color, composed by a mix of color intensity and color temperature. The light diffuser will mask the individual LEDs and appear as a blur effect or frosted effect. These together are intended to create the illusion of looking out into a window or of light coming from a window. By pairing the Daynight Panel to the Daynight app for iOS, data can be collected by utilizing location services, location-specific weather, and seasonal conditions. This information will be used to change color intensity and color temperature of the LEDs in Daynight Panel to mimic local outdoor conditions. Users will also have the freedom to choose to simulate custom sunlight schedules. The changes in simulated light intend to follow the rhythm of actual sunlight. Providing a sense of outside daylight can “ground” a human’s circadian rhythm in instances where access to daylight is sparse, infrequent, or unavailable. All these features will be performed and controlled by a microcontroller, which will be at the heart of the panel controlling the LEDs, sending information to or receiving instructions from the Daynight app.

The Daynight Panel will be optimized to provide the best outdoor atmosphere. It relieves homeowners, architects and designers of the burden of comfortably illuminating office space and it creates opportunities for saving time and money in efforts to achieve this otherwise. For example, this expands the opportunity to accommodate more indoor-locked rooms or basements while providing a pleasant atmosphere. A daylight simulation system may also be used to benefit people living in gloomy climates, people that suffer from isolated work environments, closed off or walled-in classrooms or housing quarters, or even those who experience jet lag.

Benefits of a system that simulate daylight reach beyond Earth; astronauts that spend time on the space station are often exposed to limited amount of light during the daytime. It is common for astronauts on the ISS to only receive 2 hours of sleep per night [1]. Providing users with a customizable simulation of daylight will provide benefits to people who cannot access true daylight and can be done in a way that is modern and exciting.

Giving the Daynight Panel and Daynight app a modern appeal is key in making it seem as a viable, affordable option when compared to the competing panels on the market. These qualitative features are implemented through the app and the

assembly of the app. The Daynight app places the ability to alter the state of the panel in the owner's pocket. Control is always comfortably within reach whether it is for maintenance or personal preference. This feature taps in to the modern desire for convenience and advanced technology. The panel can easily go from a simple portal to the outdoors, to a dynamic art piece, to a notifier and anything in between. This multi-capability allows the panel to appeal to markets even the engineers cannot fathom. The Daynight app paired with a simple, sleek modern design for the panel diffuser and frame make it attractive and easily adaptable to any of the rooms in which it will be placed.

The panels design requirements are based on the following goals, simplicity, modernity, improved wellbeing, affordability, environmental safety and manufacturability. The panel is mainly composed, as shown in the image below, of an array of LEDs controlled by a microcontroller behind a diffuser panel. The major operating blocks of the panel include an iOS application, a LED driver, a controller, a diffusing panel, LEDs and a power supply. These major components along with how they meet appropriate design goals are briefly summarized in the following sections.

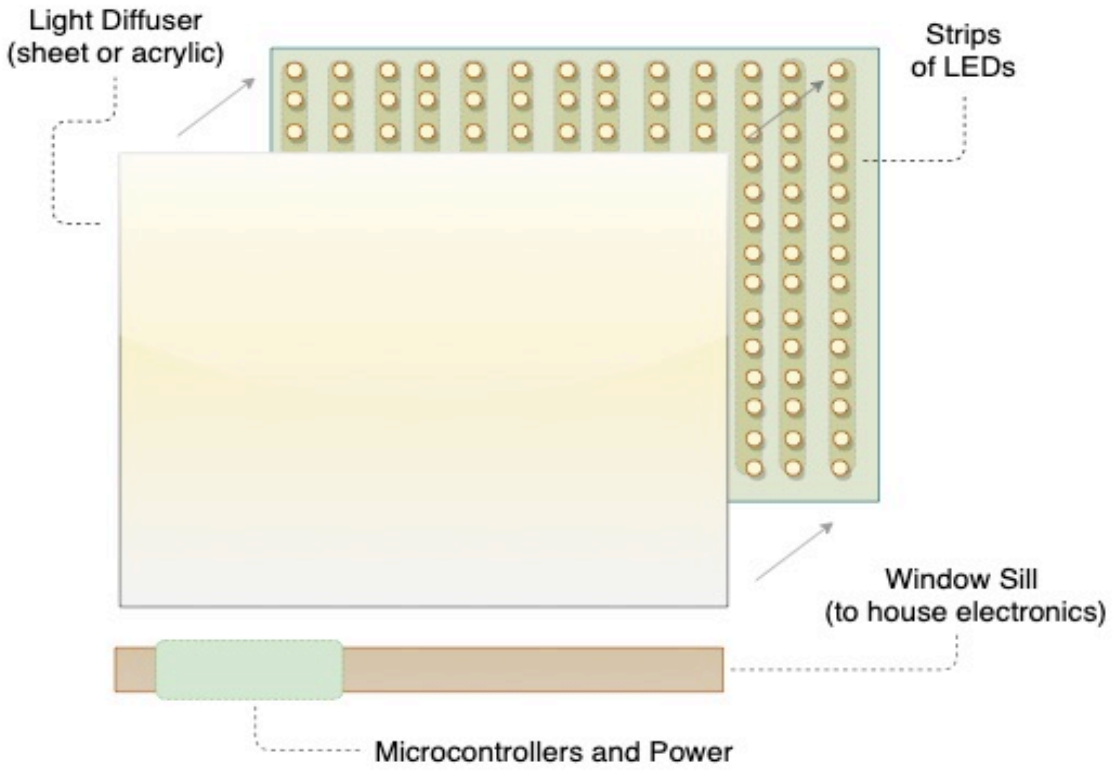


Figure 1. Rough Diagram of Daynight Panel

The LEDs are key to this panel's success because success is dependent on the scenes the panel can mimic and this ability is dependent on the amount and quality of the LEDs placed in the panel. They need to be bright enough to illuminate a

room alone or as supplement depending on the rooms size, despite the loss in brightness caused by the diffuser. Since there will be a lot of LEDS they need to be efficient to limit overall energy consumption and to lower design costs. The LEDs need to come in a set that allows color mixing. The primary colors are necessary for defining components of a scene, but a scene is composed of color and color temperature. Therefore, these LEDs need to be able to provide a range of color temperatures. This range would be somewhere between 2700K and 6000K. The LEDs are in effect like the paint the controller will use to create a scene and maximizing their capabilities improves the quality of the image, but this desire is kept in check by the requirement to make the panel affordable.

Right behind the LEDs are the LED drivers. There needs to be some sort of extension from the controller to the LEDs because there are no controllers with enough I/O pins to control all the LEDs that will be placed on the panel. Even if the controllers I/O pins were treated as a bus from which some LEDs could be controlled this configuration would limit resolution and no controller could provide the current to power several high-power LEDs from a single pin. The driver acts as an extension of the microcontroller and allows the design to become more modular and easily scaled or reconfigured. Ideally the driver would provide higher output current and additional control features for the microcontroller.

Due to integrating multiple types LEDs and how sharp of a light they are, LEDs needs to be effectively mixed and softened. This will be carried out by a light diffuser. Light diffusers will soften and mix the light that comes from the panel. It will also act as a canvas for painting color.

The image displayed on the diffuser will be determined by the controller. The controller chosen for this application needs to be able to handle pulse width modulation for several devices, it should be able to handle popular communication protocol so the options for peripheries on the panel are not limited. The controller will be the brains of the panel and it was makes what would otherwise be a static cold display seem more authentic and warmer. It will hold the code and calculations required to configure peripheral devices to set a scene for display or ambiance for a room.

Springing from the microcontroller are the relevant peripheral devices, namely the Bluetooth module and the software application. The Bluetooth module enables communication from the Daynight app to the controller for panel configuration and it allows control from anywhere within a 30ft radius. The Daynight app for iOS will provide users with a wide range of controls and customization options. Any configurations performed in the app will be transmitted wirelessly to the microcontroller on board. This abstracts the need to have hardware switches or controls on the Daynight panel.

Power for the Daynight Panel must come from the standard home or commercial wall outlet and meet the requirements set by the system design. To meet the high

lumen output required to produce a visual experience similar to daylight, more power will be required by the LEDs. This power must be also be smooth and noiseless to allow for glitch free communication on the interface busses. To fulfill the power requirements described a power supply unit will be used to rectify the alternating current to direct current and provide basic regulation. Additionally, a network of regulator will step down the voltage from the power supply unit and create appropriate power rails for each of the devices.

2. Design Motivations and Marketing

This section is focused on elaborating project motivations, goals and objectives as it pertains to marketing, the customer, and how these requirements are quantified and met through engineering specifications. These motivations are based on the following marketing requirements.

1. The panel and accompanying software applications should have a modern appeal and design.
2. The panel needs to be ‘smart’ by acting on data gathered through sensors or an iOS app to reproduce daylight.
3. The panel need to be safe and easy to install.
4. The panel should have a sustainable lifespan.
5. The panel should be affordable when compared to similar light panels currently on the market.

The modern appeal is necessary to compete with what is currently on the market. There are currently many panels and fixtures on the market that mimic daylight or daylight color temperature. These systems usually come in one of three forms, an individual panel or set of fixtures which the user controls by an application, a panel that can be switched or dialed into different settings, or an extravagant statement piece.

The individual set controlled by an application is usually the most affordable and configurable. One of the current market leaders in this product is Phillips with their HUE lightbulbs. These bulbs are designed with a variety of settings to operate without the constant need for Wi-Fi. They provide a collection of smarts lights loaded with features dealing with different modes of control, improved home automation, entertainment and notifications [2]. Their product comes as a collection which can be easily configured for any location. This form is the closest parallel to what the panel will achieve. The main difference boils down to the end shape of each device. The panels rectangular shape is designed to make it seem like a window or portal designed to improve the ambiance of a room.

The panels with dials or switches come in all shapes and sizes. Some of these resemble frosted or diffused windows reflecting a warm or cold outdoor scene depending on the setting. These panels, beautiful as they are, seem outdated because their controls are usually hardwired inside the panel. This leaves the

panel to be controlled by controls are hardwired behind the wall accessed by a switch at the entrance of a room. These panels do come with option to change the color temperature and dim the brightness. The Daynight Panel brings ease of operation from anywhere within a home or a room and an infinite amount of display options.

The third form is the extravagant statement piece. These are usually architectural projects designed for a specific location and because it is so specialized it becomes unaffordable for most of for those who could afford such a piece might not like the design intent behind these show pieces. The panel on one can be a simple fixture but on the other it can be a dynamic piece of wall art that can be reconfigured and hung in just about any location.

The Daynight Panel is not a novel idea, but it does bring new functionality and viable options to the market. Filling this gap constitutes the motivation for the project and inspired the marketing requirements.

The requirements and specifications of the Daynight Panel are defined in the table below. Six marketing requirements were determined to be the primary needs of the user that the Daynight Panel must fulfill. These requirements define how the Daynight Panel must look, operate, and interact with the user and the surrounding environment. To fulfill the marketing requirements engineering requirements are defined to quantitatively define parameters that must be set in order to fulfill the marketing requirements. Each engineering requirement has a quantitative value with appropriate units that must be met to fulfil the engineering requirement. A justification for engineering requirements is also provided by defining how the respective engineering requirement meets various marketing requirements.

The requirements and specifications of the Daynight Panel are defined in the table below. Six marketing requirements were determined to the needs of the user that must be fulfilled by the Daynight Panel. These requirements define how the Daynight Panel must look, operate, and interact with the user and the surrounding environment. To fulfill the marketing requirements engineering requirements are defined to quantitatively define parameters that must be met in order to fulfill the marketing requirements. Each engineering requirement has quantitative value with appropriate units that must be met to fulfil the engineering requirement. A justification for engineering requirement is also provide defining how the respective engineering requirement meets various marketing requirements

Table 1. Requirements Table

Marketing Requirements	Engineering Requirements	Justification
1, 3, 4, 6	1. Should have an iOS app that supports Bluetooth Low Energy at a range of *>= 10 m*.	To deliver the best user experience with Bluetooth technology, should have a range reasonable for controls within a home.
2, 5	2. Should have an integrated power supply that occupies *<=20% area*.	For aesthetic reasons the power supply required for the AC/DC conversion must be integrated into the frame and have a form factor.
1, 3	3. Reproduce the effect of daylight with *> 12 hours* of autonomous run time.	The device must use external data and/or sensors without requiring constant interaction.
2, 4, 6	4. Should weigh *<= 50 lbs*	Keeping the panel under 50 lbs makes sure that no more than two people are required to install.
5, 6	5. Weakest component must have at least *4,500 hours* lifespan	This will ensure the panel can stay powered for up to 12 hours of daylight use.
4, 6	6. Must cost *< $2,000*	Keep potential market prices low to make the technology accessible.

Market Requirements

1. The panel must have a modern appeal and design
2. The power brick cannot detract from the aesthetic appeal of the device
3. The device must be able to use sensors or data to reproduce the daylight
4. The panel must be easy and safe to install
5. The panel should have a sustainable lifespan
6. The panel should be affordable

2.1. Motivations

The circadian rhythm is kept on track and maintained by observing the color temperature of natural light. Therefore, exposure to blue light (naturally observed during the daytime) when the circadian rhythm is expecting orange light (naturally overused during or after sunset) can be disruptive. The production of melatonin is kept on schedule with circadian rhythms and can be disrupted by extension.

Natural light and its effects on the human circadian rhythm are important to understand for the scope of this project. As outside light enters a building, it provides a lower level of inside light. These two measurements of light are used to create a daylight factor, seen in architecture. A ratio is created between outside light to inside light and is used to create environment lighting requirements inside buildings to support the best human well-being [3]. The daylight factor in a building can be increased by placing windows to have a better “view” of the sky, by increasing the reflection of light of exterior surfaces outside of the window, or by choosing furniture inside of a room with lighter colors to increase the illumination of the room. When simulating daylight, the first two approaches to adjust the daylight factor of a building will be abstracted from the user.

2.2. Trade-Off Table (House of Quality)

The House of Quality summarizes the relationships between various marketing and engineering requirements for a product. It sets up a framework for easily seeing how changing one aspect of a project can affect other aspects.

The relationships will show if the aspects have a strong or mild positive correlation strong or mild negative correlation, or no real correlation. Marketing requirements have their correlations with engineering requirements. These make up the base of the house.

The roof the House of Quality is made up of the relationships of the different engineering requirements. Each correlation diamond on the head of the house break off to the two engineering requirements that is describes.

A house of quality is necessary because it gives an easy documentation of the different relationships between various requirements. It speeds up the design process and prevents miscommunication. It also helps prevent people from forgetting how changing one part can affect many others.

The end goal of the House of Quality is to help open communication between the marketing world and the engineering worlds. These two sides typically have trouble communicating their constraints and requirements. The House of Quality ties the two together by giving basic constraints that they need and graphically showing these constraints to the marketing side. The flipside is also true, showing how marketing requirements affect the engineering constraints.

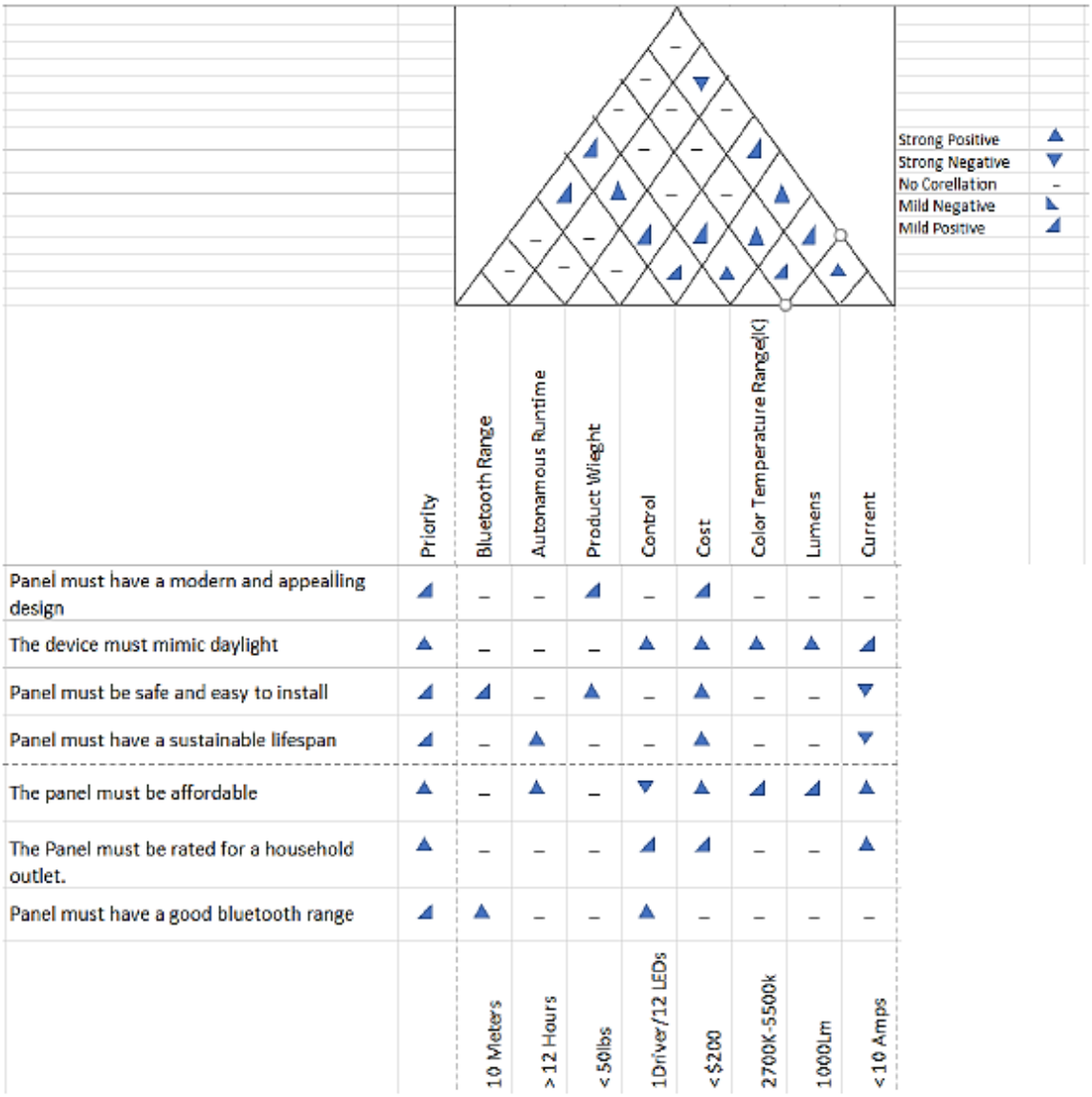


Figure 2. House of Quality

The House of Quality table visualizes the relationships between various market requirements and engineering solutions. These aspects can have weak, moderate, or strong relationships with each other. These relationships can then be directly or inversely related. The block of requirements on the far-left details important customer ideals for the panel. The bottom block or “basement” of the house defines the quantitative parameters for measuring how we will meet the customers’ demands. The block right beneath the roof of the house defines technical requirements that have to be met for the panel to meet customer, market, and engineering constraints.

The customer requirements that affect the most engineering specifications are the requirements for daylight mimicking and affordability. Since mimicking daylight is the ultimate purpose of the panel, all parameters are going to be strained to meet this requirement. Looking at how the different engineering parameters meets these critical customer requirements will help to define some design constraints.

One of the parameters is control. Control is necessary because it is what will make the simulation seem realistic. Fine control of the LEDs touches several aspects of the design. It requires good resolution, high LED Density, the number of steps in pulse width modulation, and the controller. The resolution is directly linked to the LED density the more LEDs there are in an area of the panel the brighter it will be. The resolution will also help to enable other modes on the panel for future designer. It can become a dynamic art piece on the wall perhaps mimic vague scenes one would see outside, or it could be used in a party disco mode. All of these modes require pulse width modulation and a microcontroller to send the appropriate instructions to enact this mode.

Another parameter is cost. There are many daylight mimicking lighting devices on the market but what we are designing aims to be affordable for mass consumption. This requirement offsets all the others because increase anything increases cost when the goal is to decrease it. The star of the Daylight Panel is the LED. It has to be bright enough to light up a room, like the sun would, despite the diffuser that will cover and house these LEDs. These LEDs are not cheap and typically LEDs this powerful are not through hole. There will be an assembly cost in addition to purchasing the surface mounted LED. In this arena cost also affects appearance. The panel should be slim and sleek yet the cheap powerful fixtures capable of emitting the light needed are usually bulky. Therefore, to meet the customer requirement of being sleek the more expensive surface mounted LEDs may be used.

The next parameter that affects the panels ability to mimic daylight is color temperature. Although all colors can be described with color temperatures, we are most interested in the color temperature description of white light, as to closely match daylight simulation. Bounding our attention to white light also abstracts information that might be confusing since lights of different colors may be quantified as having the same color temperature. Normalizing color temperatures to a white color will provide the best way to clearly understand and simulate daylight.

It is very easy to produce a bright light but these kinds of lights are often cold and unsettling. Since they also can be painful to the eye, the LEDs that are chosen need to be able to mix light color to create the warmth that is missing in a lot of fluorescent lights popularly used today.

In the following table the color temperature in kelvin is listed along with a reference example of a fixture that could output something in this color temperature along with the corresponding color. This table shows the difference between how color is perceived indoors and outdoors. Part of designing the panel will be defining a formula that can provide the outdoor color will being inside.

Degrees Kelvin	Type of Light Source	Indoor (3200k) Color Balance	Outdoor (5500k) Color Balance
1700-1800K	Match Flame		
1850-1930K	Candle Flame		
2000-3000K	Sun: At Sunrise or Sunset		
2500-2900K	Household Tungsten Bulbs		
3000K	Tungsten lamp 500W-1k		
3200-3500K	Quartz Lights		
3200-7500K	Fluorescent Lights		
3275K	Tungsten Lamp 2k		
3380K	Tungsten Lamp 5k, 10k		
5000-5400K	Sun: Direct at Noon		
5500-6500K	Daylight (Sun + Sky)		
5500-6500K	Sun: through clouds/haze		
6000-7500K	Sky: Overcast		
6500K	RGB Monitor (White Pt.)		
7000-8000K	Outdoor Shade Areas		
8000-10000K	Sky: Partly Cloudy		

Figure 3. Color Temperature Chart [4] Permission Granted by William Kelvin. Appendix A-7

The next parameter that affects the panels ability mimic daylight is the Lumen output. It is easy to mix colors and get the right temperature but they panel needs to be bright as well and this is measure with Lumens. To increase lumens a high lumen output LED will need to be found. A high density grouping of high lumen LEDs will help to meet the brightness requirement. It has to be understood that brightness is relative and that it would be inefficient to make a panel that could even get close to outputting solar lumen levels. Instead the brightness would be relative to indoor lighting. The places where the panel would be installed are typically dark or dreary and blue. Placing the panel in these environments would make a great difference and make the atmosphere in these rooms more inviting, warm or natural.

The final parameter that affects the panels ability to mimic daylight is the current. The LEDs that are going to be doing the color mixing, providing the high lumen output and color temperature all require a lot of current to run and while individually

it is not much collectively, they will be drawing a lot of current. Enough that heat may become an issue with the panel. Therefore, while the current enables all of the panels capabilities it also limits the size based on the internal components. The panel is limited to 120V from an outlet and most household breaker trip at 15A. To ensure safe installation and operating conditions the panel would not reach those extremes, so the current becomes a limitation to take into consideration.

2.3. Block Diagrams

The Daynight Panels main subsection can be described using two block diagrams. The first block diagram will represent the physical hardware systems found in the Daynight Panel. This includes the power subsystem, microcontroller, communications subsystems, and LED drivers. The second block diagram will include the architecture for the software design of the Daynight app.

2.3.1. Hardware Block Diagram

The Daynight Panel will be a system composed of a central controller powered by an integrated power supply and possibly additional batteries. This integrated power supply will connect to a typical home or office wall power outlet. The controller will receive software instructions and sensor data to decide how to tune the LED's to mimic current daylight conditions. These software instructions will come from the Daynight app will and be transmitted to the microcontroller via wireless communication. The tuning will be done by the LED driver which will receive instructions from the controller and change the color or intensity of the appropriate group of LEDs. The tuning of the LEDs of various colors and color temperatures creates an overall image that is reminiscent of the view out of a window. The view produced will be set by the user or set to vary automatically based on real world data.

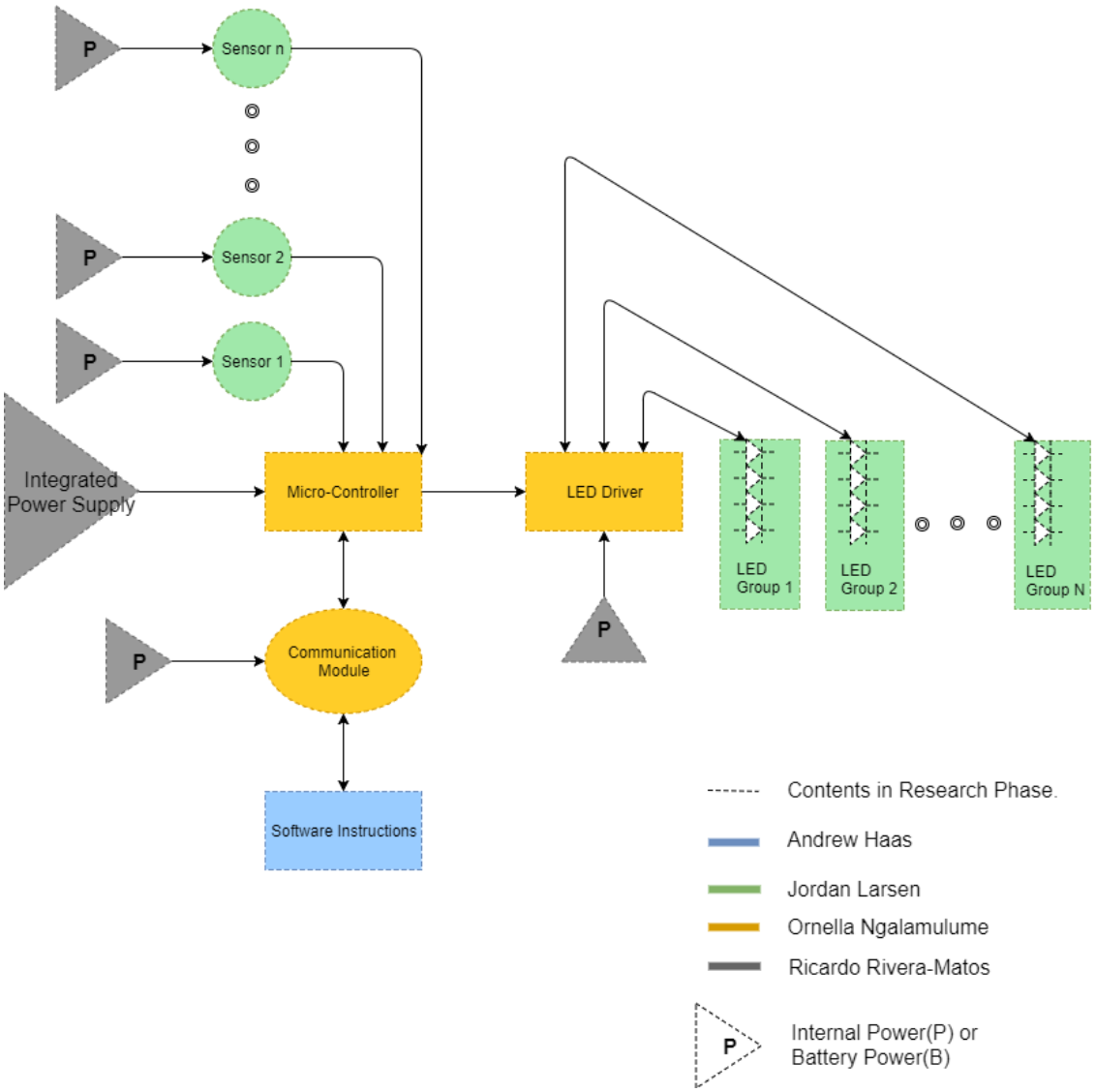


Figure 4. Hardware Block Diagram

2.3.2. Software Block Diagram

The Daynight app will be the center for all user interaction with the Daynight Panel. After the Daynight Panel is paired with the Daynight app, the user will have a full range of controls to customize the appearance. Preset colors and creative vistas will be available for selection and can be adjusted to the user's current location. If the user prefers to disable location services, the sunlight cycle can be set by using approximated location based on time zone.

All iOS software will be designed and implemented using the Swift API provided by Apple, Inc. Core Bluetooth is a framework that provides classes and functions to create a seamless experience between a software application on iOS and hardware equipped to support Bluetooth 4.0 low energy technology. UIKit is a

framework that supplies classes, functions, media, objects, and more to assist building a UI. UIKit-designed UIs are easy to use and quick to understand. This will be beneficial in creating an app that has a modern appeal and can neatly display all relevant information and controls. Core Location is a framework that provides information about the geographical location of a device by using all available on-device hardware such as Wi-Fi, GPS, Bluetooth, magnetometer, barometer, and cellular hardware.

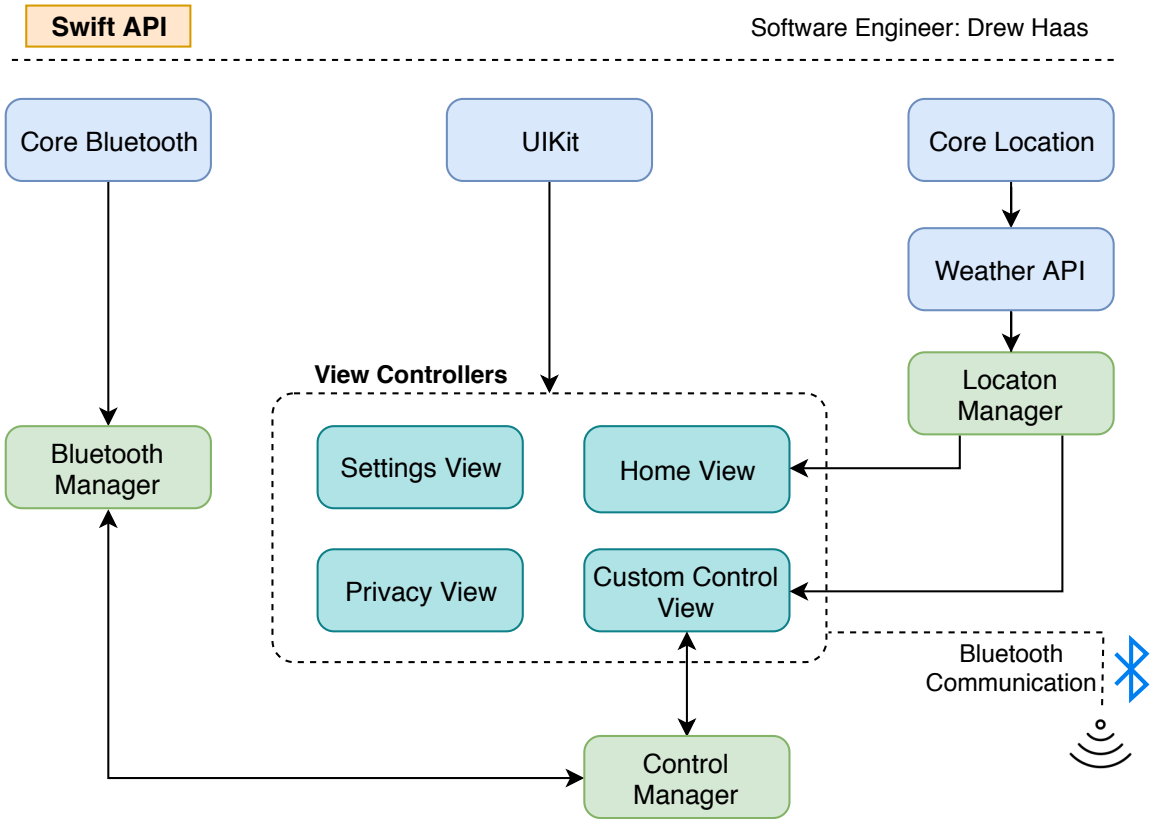


Figure 5. Software Block Diagram

View Controllers represent separate pages or views in the app. Laid out above is an outline of what views would be most beneficial for an app of this design. These are where the user has direct control of how to interact with the Daynight Panel. Any actions performed on these view controllers will be communicated over Bluetooth.

All classes outside of the View Controllers will be working in the background to update the data to be presented. This falls in line with the Model-View-Controller software architecture pattern.

Managers are software classes that utilize information provided from APIs to interface with the view controllers. Manager classes should be the only classes editing data, and view controller classes should be the only classes editing the UI.

3. Research and Parts Specification

The panel is composed of 6 major components, the LEDs, LED drivers, power supply, voltage regulators, a controller and instructions provided by code. These components all need to work together in an environment that ensures communication and safe operation for all components. This section will discuss the importance of each component to the panel as an ideal then look for real world options that can meet the ideal criteria. Of course, there will be some discrepancy between the expected performance of a device, what the market has available and what the budget will allow. These components will be analyzed with these discrepancies in mind to choose the most viable option.

3.1. Lighting

Lighting is one of the most important aspects for the project. LEDs need to be specially considered. Sun and skylight both need to be simulated to a well enough degree that people would be interested in it while also being cost effective. The color of light will be an important factor in this.

3.1.1. Typical Conditions

Typical conditions are looking at how the panel interacts with its environment. Usually typical indoor conditions are constant and, in a state, requiring aid from the panels light. The state of a room is therefore simple to account for. Usually what is needed will be defined by the user through an application.

In the grand scheme of things typical daylight conditions may be considered seasonally for Daylight but on a daily scale the minute changes can be considered infinite. An important fact to consider for typical conditions is the state of daylight and how much light is needed in a room. The goal is for the Daynight panel to be a light source that can provide the same amount of light that would be ideal for an office environment. How ideal is defined can change every second or every and a scope for how the panel will respond and keep up need to be defined.

Granted, while the panel is smart it is not an independent responsive source. While this capability can be introduced in the future, the panels response will change with the preferences of whoever is operating the application. In the following sections the effect of typical indoor and outdoor conditions on the panel operation will be investigated.

3.1.1.1. Daylight

Daylight has a wide spectrum of light produced and the color of said light. The color of the sky can also be different from the color of light produced by the sun. These conditions must be able to be simulated by the Daynight system for it to be successful.

The table below shows a range of color temperatures. The desire is to simulate the range of color temperatures that is produced by daylight. This will be used as a reference on the desired range of color to create daylight conditions.

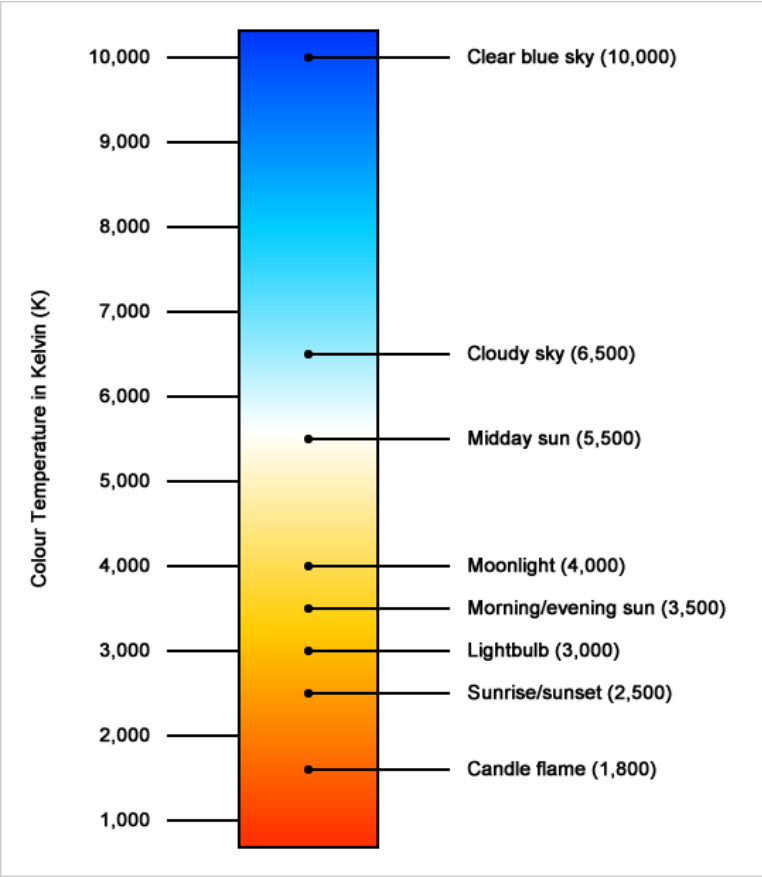


Figure 6. Color Temperature Table [5] Permission Requested from PhotographyMad

Daylight has a color temperature range of 2500K for sunrise/sunset, to ~3500K for early morning and late afternoon, to as high as 6500K for high noon. Lights will be needed to provide for this spectrum of color. A positive aspect is that two lights that produce two different temperatures can be used in tandem to create a different color temperature. For example, a 3000K light can be used with a 6000K light to produce a 4500K temperature. This can go further though. By reducing the power of one light, whole spectrums of color temperature can be simulated using only two lights.

The next aspect that must be simulated is the various colors of the sky. Due to the wide spectrum of color that the sky can take on, RGB would be the best option to use. RGB can produce a similar range of color making it an optimal option.

An important feature for simulating light is a lights color rendering index. This number represents a scale of 0 to 100% that describes how accurately at light source can render it's given reference color. For example, if a bulb was described

to be a 2700k temperature, a higher number would mean it more accurately is able to create that specified color temperature. [6]

3.1.1.2. Working Light Levels

Daynight must also produce enough light to be usable for work conditions. The amount of produced light must be in the range of 500-1000 Lux. This range is typical office conditions [7]. Lumens describe a volume of light that is produced by a bulb. To get lux from this, one finds the total surface area reached by the bulb, and then divides lumens by this surface area.

The image shows a few examples. The LED has a volume of light that it places onto a surface. The lux is determined by the viewing angle of the LED and the distance from the LED to the surface. This spreads in a cone like fashion and creates a circular surface. Assume that all the LED sources have the same number of lumens. The first two have the same angle, so the different distances to surfaces will result in the second having a much lower lux. The third has a smaller viewing angle, but the same surface area as the first. Therefore, the first and third will result in the same lux.

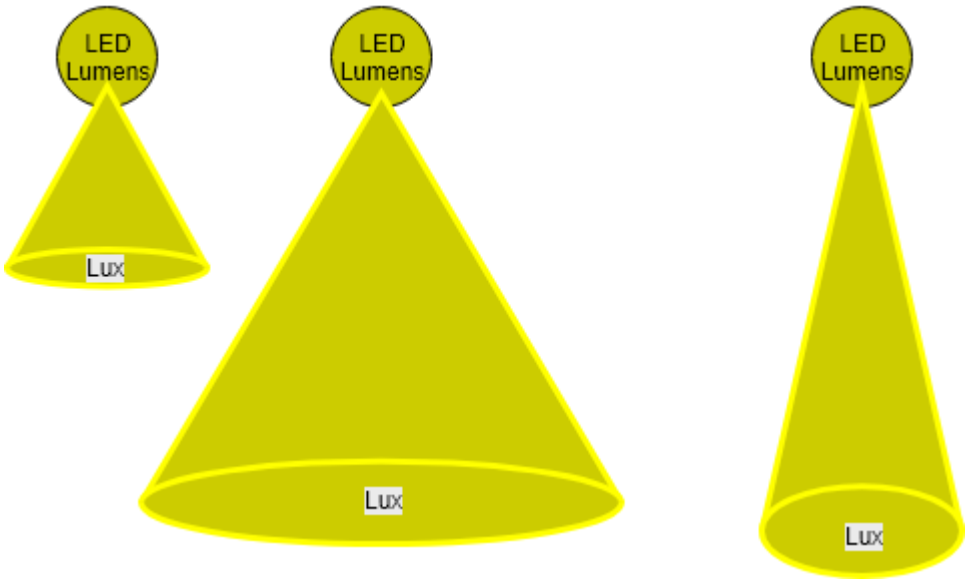


Figure 7. Lumens to Lux

A calculator was made in Excel for calculating how much lux was produce by a LEDs in a given spacing for a given surface area. The viewing angle and the lumens of the light were inputted into the workbook and this output the lux was received at an input distance. This tool can be used to determine what products to choose for a given lighting technology, with considerations to a bulb's viewing angle and lumens.

The graph below shows the inverse square relationship between lumens, lux and distance. The graph used a 1,000-lumen source with a 120-degree viewing angle. It ranges from 1-10 meters distance from the source.

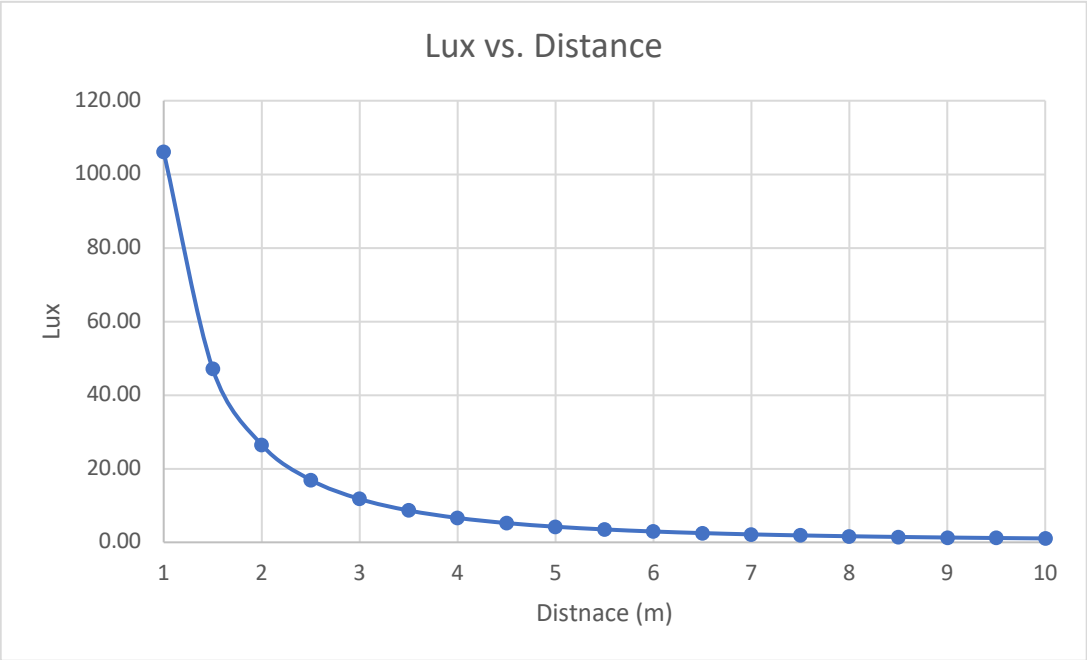


Figure 8: Lux vs. Distance Table at 1000 lumens

Real life examples were also studied, and it was determined that at least 1000 lumens would be needed to properly light up a room. This was done by observing various light sources and panels at home depot.

3.1.2. Lighting Technology

The lighting system of the Daynight panel must be able to last for at least 2 years of use, effectively light a room, and not consume an excessive amount of power. The important considerations to take when choosing a type of lighting is the amount of light produce, the power consumed, the total runtime hours of the bulb, and how much space the lights take up.

A goal would be to minimize power consumed for light that is produced. Each lighting technology had general trends of how much power it consumes for a typical light level. Another factor of this is how much heat is produced by each type of technology. If a technology produces a large amount of heat, then heat sinks would be required to disperse that heat to prevent the system from burning up.

Another consideration is how long a type of technology runs. This system needs to last an extended period without failing to preserve the value of Daynight. Daynight cannot afford to have a short lifespan.

The size of the light source must be minimized for light that it produces. While the source can be relatively large, it must also produce a corresponding amount of light. Smaller sizes would allow for greater resolutions of light to be produce, making the Daynight system more attractive.

Finally, Daynight must be able to simulate sunrise, sunset, high noon, and the color of the sky. A wide spectrum of colors is needed to be able to fulfill this.

Table 2. Lighting Technology Goals

Consideration	Goal
Light Source Size	Minimize
Heat Produced	Minimize
Light Produced	Maximize
Light Resolution	Maximize
Color Range	Maximize

3.1.2.1. Incandescent

Incandescent lights consume a large amount of power and produce large amount of waste heat. Due to how much energy is transformed into heat instead of light, this bulb consumes a relatively large amount power for the light it produces. This heat needs to be dispersed away from Daynight panel to prevent heat damage to the system.

These lights tend to be large. This would cause the resolution of the produced light to be reduced. This would also force the system to take up more space, which would make Daynight not look as good, as well as violating size specifications. The need for a larger heat sink would work further against using incandescent lights as an option. Another drawback worth mentioning is that integrating this into a board could be difficult since it would likely need a fixture to lock the bulb in.

The incandescent lights also only have an average run time of 1200 hours [8] which fails the required run time for the system. This makes incandescent lights extremely unviable for the Daynight panel.

Incandescent lights have the benefit of spreading light in all directions. This reduces the need for light diffusion since the light is effectively diffusing from the bub itself. This gives incandescent lights a point for use.

Incandescent lights tend to have high Color Rendering Index (CRI) scores, making them good at creating their claimed color temperature.

3.1.2.2. CFL

Compact Fluorescent Lights (CFL) lights are highly efficient in terms of producing light for how much power it consumes. It still produces waste heat, though not nearly to the degree that incandescent lights produce.

CLF lights share incandescent lights problem of being relatively large. leading to a less bulbs in the system to make up the lighting resolution as well as forcibly increasing the size of the system. This size increase would violate required parameters for the design making it weak as an option.

These lights typically last from 8,000-12,000 hours [8]. This satisfies the minimum runtime for the system. However, it only just makes this requirement.

CFL lights also have high efficiency for light produced vs energy consumed. This leads to less heat being created which allows for smaller heat sinks to be used. This helps in reducing the size of the system.

CFL lights tend to have weaker CRI scores compared to other light sources. This makes them less viable as they wouldn't as reliably produce daylight colors.

3.1.2.3. LED

Light emitting diodes (LED) consume little power for the light they can produce. They are slightly more efficient that CFL lights. This means that less heat is produced, which reduces the heat dispersion required for the system.

LEDs have runtimes of approximately 25,000 hours [8]. This far exceeds the minimum runtime for the system. LEDs are less likely to die which can prevent quality reductions or failure of the system.

LEDs tend to be relatively small, which allows for a high resolution of lighting to be created. However, LEDs tend to disperse their light in a single direction. This requires greater diffusion to even disperse the light produced by the LEDs. This issue can be lessened by since LEDs are designed with a viewing angle. This viewing angle is the angle that light is distributed from. The effect of increasing the viewing angle is that more light is distributed over a given area.

Although LEDs are small, they will have their own challenge for integrating them. More LEDs will be needed to produce a given number of lumens, and these will have to be surface mounted into a PCB board. It will take a decent effort to integrate all the LEDs into the PCB.

LED lights tend to have very good CRI scores. This makes them idea for creating reliable daylight colors. This also helps with the RGB aspect since the colors that are created will be able to be more faithful to typical sky conditions.

3.1.2.4. Lighting Summary

The table below gives a summary of the important considerations for the different types of lighting technologies. Based off these considerations, the best choice of lighting technologies was LEDs. LEDs hold the best qualities in almost every category that was deemed important. Areas of particular focus are heat production, color range, lumens per light, and run time. These aspects are particularly important because the system must run for a long time, not produce too much heat, and be bright for the number of LEDs on the Daynight Panel.

Table 3. Lighting Technology Comparison

	Incandescent	CLF	LED
Heat Produced	High	Low	Very Low
Size	Large	Mid-Sized	Small
Integration Effort	Med-High	Med-High	Medium
Color Range	Low	Low	All color ranges
Run Time	Low	High	Very High
Lumens per Light	Low	Medium	Medium

Based off this information, the best option to go with is LED lights because of their small size, color range, and high runtime.

3.1.3. Potential Products

There is a wide range of product available to choose on for the Daynight panel. Special consideration needs to be made for a wide range of factors to ensure the maximum success of the project. These considerations and their consequences are discussed in the following sections.

3.1.3.1. General Considerations

Important aspects for LED products are the lumen output, that header angle that the LED outputs, and how much power it draws on. The list below shows the benefits and negatives of different aspects.

- Lumens output
    - Higher lumens can be higher cost per LED, but less LEDs placed

    - More LEDs mean harder to PCB
- LED angle
    - Wider angles disperse their light to greater degree
    - Can be advantage or disadvantage based on design
- Power
    - Forward Voltage
    - Lumens/Watt
    - Current Draw

For Daynight, there are 3 types of LEDs that are needed for the system to optimally run. A cool temperature LED, a war temperature LED, and a RGB LED. Cool temperatures in the range of 6500 Kelvin simulate light colors seen in broad daylight. Warm LEDs in the range of 1500-2700 Kelvin (K) can simulate light levels seen in sunrise/sunset light conditions. RGB lights will be used to simulate the color of sky for various light conditions.

To simulate the light conditions between 1500K and 6500K, these two LEDs can be mixed together with varying intensities to create those color temperatures. These two LEDs will be the heavyweight in representing the various day conditions of light.

With regards to power, forward voltage is currently not a heavy concern, though the drivers cannot handle a forward voltage greater than 20 V. Current draw was a much heavier concern, with the LED driver only being able to handle 100 mA of current. A workaround to this is connecting an LED to multiple lines on the driver.

This leads to two options. Either multiple LEDs can be assigned to a single channel, and the resolution of the system goes down, or use more LED drivers. This will be a heavy consideration when choosing LEDs.

The second heavy consideration will be cost. Daynight will implement over a hundred LEDs of each type, so price needs to be relatively low, so the system does not cost too much to build. More powerful LEDs can reduce the number of LEDs needed.

The final heavy consideration will be how many lumens they produce. The LEDs need to be able to sufficiently lighten up a room so that it can be used as a light source. Increasing the viewing angle of the LED will cause the LED to diffuse across a greater area for a given distance from the LED. This can be both positive in that a weaker diffuser is needed, and a negative in that more light will be needed.

A small list of manufacturers that makes the LEDs that are needed is Samsung Group, Nichia Corporation, and Philips LumiLeds.

3.1.3.2. Cool LED

The cool LED needs to have a color temperature of approximately 6000-7000K. Due to LED standardization, the only temperature LED that is produce that ends up satisfying this requirement is 6500K.

Distributers that were looked at tended not to have LEDs of this temperature in stock, and to order them would require ordering in quantities of over 1000. This led to a reduced amount of choices for what LEDs to use. Using the tool described in the lighting section, several products were chosen based on their ability to produce the required amount of light, max current draw, and cost.

Table 4. Cool LED Comparisons

	Product A	Product B	Product C
Manufacturer	Lumileds	Cree	Samsung
Lumens (lm)	97	126	175
Viewing Angle ($\boldsymbol{\theta}$)		110	110
Forward Voltage ($\boldsymbol{V_f}$)	6.1	2.9	2.9
Max Current Draw (mA)	240	1000	1400
Max Power Draw (W)	1.46	2.9	4
PCB Mount Type	3030	1414	2-SMD
Cost	$0.36	$0.89	$0.45

The final decision for the cool LED was to use the Lumiled option. This decision was made since it was one of the extremely few cool LEDs that had a relatively low current draw. Other options existed that had a low current draw, but they were above a dollar a LED.

Although the current draw for a single LED exceeds the limit for a single channel on the LED driver, this will be alleviated by each LED to multiple channels. This will require more drivers overall, but this was found to be acceptable.

The footprint of the selected product is a standard 1212 package or 3030 in metric (3.0 mm x 3.0 mm). It has 2 pads, one for cathode and one for the anode. The pad for the cathode is 2.2 mm x 1.43 mm and the anode side are 2.2 mm x 0.48 mm.

3.1.3.3. Warm LED

The warm LED needs to have a color temperature range of 1500K-2700K. LED Standardization had options for either end of these two temperatures, but it was decided to use 2700k lights.

Once again, lights that matched our requirements were not common as they either required to be ordered in extreme quantities or were out of stock. This reduced the number of available options that could be applied to the project. Using the tool described in the lighting section, several products were chosen based on their ability to produce the required amount of light, max current draw, and cost.

Table 5. Warm LED Comparisons

	Product A	Product B	Product C
Manufacturer	Bridelux	Samsung	Samsung
Lumens (lm)	101	90	103
Viewing Angle ($\boldsymbol{\theta}$)	116	120	145
Forward Voltage ($\boldsymbol{V_f}$)	9.2	9.4	5.9
Max Current Draw (mA)	120	110	250
Max Power Draw (W)	1.1	1	1.475
PCB Mount Type	2835	2835	1313
Cost	$0.11	$0.21	$0.07

The final decision was to use Product B. A strong factor for the decision to use this one was that its max current draw was within the parameters of the drivers being used. This option although the most expensive of the three top options would ultimately save down the line since extra driver space wouldn't be needed to split the current.

The selected product has a standard footprint of 2835 (2.8 mm x 3.5mm). The LED has two pads, one for anode and one for cathode. The anode pad is 2.18 mm x 1.9 mm while the cathode pad is 2.18 mm x 1.1 mm.

3.1.3.4. RGB LED

The RGB lights needed to be bright enough to paint the diffuser panel while also not filling the room with too much blue light. Cost effective options were extremely difficult to find.

Table 6. RBG Comparisons

	Product A	Product B	Product C
Manufacturer	Cree	Kingbright	Lumex
Lumens (lm)	8.2-30	7.2-28	18.5-79.2
Viewing Angle ($\boldsymbol{\theta}$)	110	120	120
Forward Voltage ($\boldsymbol{V_f}$)	2.1-3.1	2.3-3.3	2.5-3.5
Max Current Draw (mA)	100	150	150
Max Power Draw (W)	0.52-0.72	1.56	0.5
PCB Mount Type	SMD	5050	5050
Cost	$0.72	$3.36	$4.79

The chosen product also includes a white light that will not be used. Due to limited product option, this one was chosen despite the presence of an extra light that will not be used.

The footprint of the chosen product is described by the image in Appendix D-1. It has eight pads with red, green, blue, and white each taking up two pads. The white light is not being implemented in this project.

3.1.4. Light Diffusion

Light diffusion is key to the panel's function and appearance. This is a feature that will be set and invariable once in place. Therefore, it is within our goal to find a light diffuser that provides the greatest viewing experience. Having a light diffuser that is sleek in design, simple in appearance and handling, and inhibits the least amount of light from being emitted are all key features to keep in mind while considering which kind of light diffusion will be best to use in the Daynight Panel.

3.1.4.1. Why Use Light Diffusers

Light diffusers are useful for spreading out light across an area more evenly. It also softens the light, making it less harsh for those who want to use the Daynight Panel as a source of light [9]. Light diffusers are especially helpful for LEDs since LEDs by nature are highly directional. The light diffuser will soften the edges of the LEDs creating a cleaner light surface.

Another important aspect for the diffuser is it creates a surface that can be 'painted on'. This helps to simulate blue sky on the Daynight panel. To do this, a clouded diffuser will be needed. This will absorb some of the light and reduce lux on ground, but it achieves the goal of creating a sky-like surface.

3.1.4.2. How Light Diffusers Work

Light diffusers are typically a plastic that act similarly to prisms. Diffusers soften light by scattering it into an increased amount of area. Unlike prisms though, they are designed so that the various wavelengths of light don't separate. Light can be spread out into several different orientations and shapes. These primarily can be circular, elliptical, and extreme elliptical [10]. Diffusing light in to different orientations and shapes allows the project to give off a different appearance. During the considerations for which light diffuser works best for the Daynight Panel, this is something important to keep in mind.

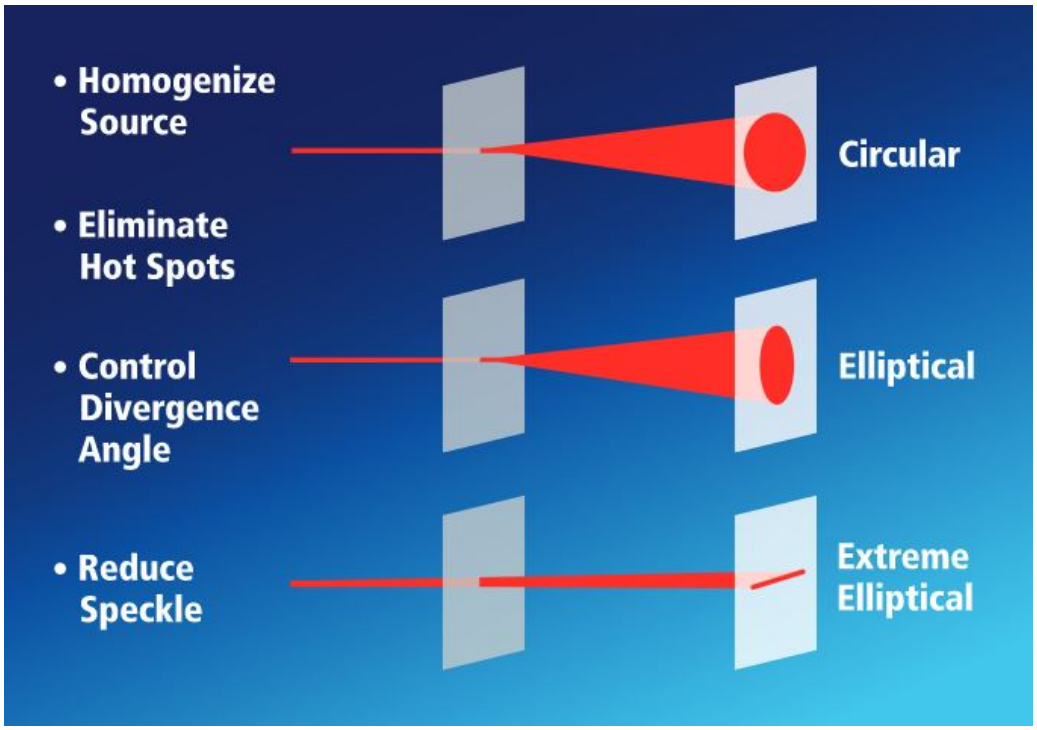


Figure 9. Different diffused Light [10] Permission granted from Luminit

3.1.4.3. Diffusion Options

To find options for light diffusers, companies that produce them were contacted on the best options for the project's specifications. These were being able to mix color temperatures and diffuse the light when the LED is less than 3 inches from the diffuser. Brightview was contacted for this.

The LED luminaire efficiency represents the amount of light that makes it through. For the product line given by Brightview, this efficiency is inversely proportional with how well the diffuser hides the LEDs. The diffuser strength is directly correlated with how well it diffuses the light. The ratio in distance for hiding the LED represents distance between LEDs to the distance to the diffuser.

Out of intentions to expand our scope of light diffusion and to offer ourselves the most amount of options, two other manufacturers—Luminit and Condale Plastics Ltd.—have also been contacted about diffuser options, but no response has been received at this time.

Table 7. Diffuser Comparison

	Product A	Product B	Product C
Manufacturer	Brightview	Brightview	Brightview
Item Name	C-HE80	E-1560	C-HH90
LED Luminaire Efficiency	88-82%	90-95%	88-93%
Distance complete LED hiding	1:1.5		1:0.85
Diffuser Strength		15x60	

A final decision will be selected after samples can be tested and compared across one another. This provides us the opportunity to find the right fit for the look and feel of the Daynight Panel. A goal is to make the light from the LEDs homogenize across one another. Product A and C are as described Brightview high-efficiency diffusers while product B is an elliptical diffuser.

The differences between these two types of diffusers are in the way they spread, mix, and disguise discrete light sources. High-efficiency diffusers are described as being best for, “General purpose for smoothing, lamp hiding, and color mixing.” Elliptical diffusers “spread light elliptically. Mixes lines of LEDs, and homogenizes fluorescent lamps. Also excellent for wall wash, cove, linear, specialty.” Considering the description between the two types, we feel that high-efficiency diffusers will suit our goals the best [11].

3.2. LED Driver

The LED driver acts as the “brush” that will be used to create the scenes in the Daynight Panel. It is an extension of the microcontroller and should be either directly transfer controller functionality or add to the controller’s abilities. The following sections look at the importance of the driver to the panel and investigates different types of drivers that could be used in an effort to select the best one for our application.

3.2.1. Why Use an LED Driver?

To drive the large number of LEDs on the Daynight Panel individually would require an MCU with M by N number of pins as shown in the image below. Rather than using several of the individual general-purpose input/output (GPIO) pins on the microcontroller to drive multiple LEDs, an LED Driver can be used. These drivers can drive upwards of 48 LEDs using only a few GPIO pins on the microcontroller. This is accomplished by using a few GPIO pins to establish serial communication between the microcontroller and the LED driver chip. Through this serial communication link, instructions for what LEDs to “drive” are fed from the microcontroller to the driver IC. Additionally, some driver ICs allow for special instructions such as brightness control and image correction that would not be easily implemented by driving the LEDs directly through the microcontroller.

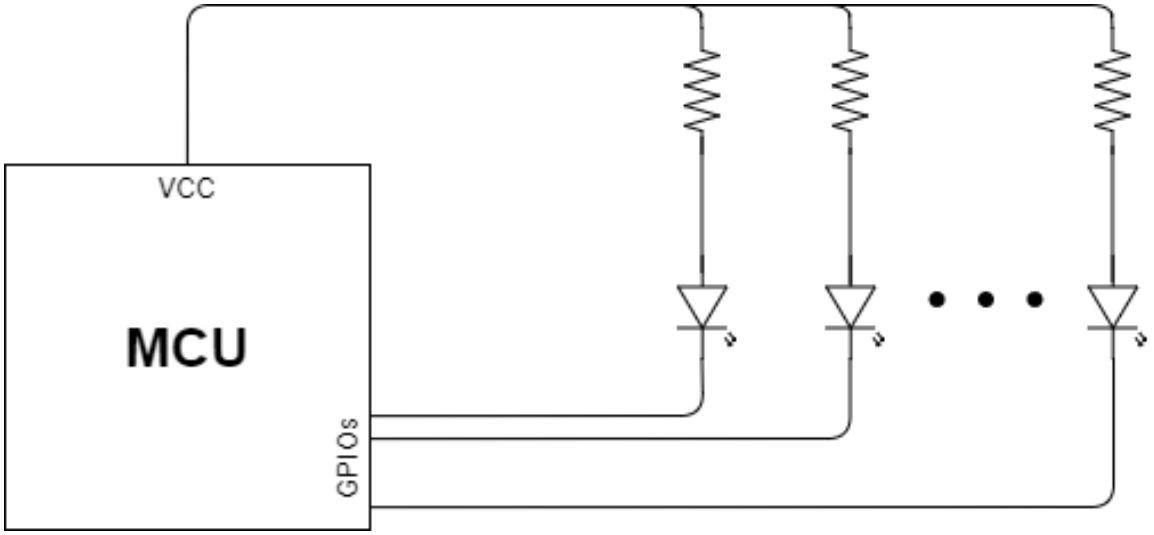


Figure 10. LEDs driven from MCU

The figure above shows how several LEDs are driven using only a microcontroller. This configuration is simple but makes inefficient use of the microcontroller’s GPIO pins. The GPIO pins are a limited resource on the MCU that must be preserved for interfacing with the Bluetooth module and other peripheral devices that may be added to the Daynight panel. Or several MCU’s would have to use at the same time but this configuration is not cost effective.

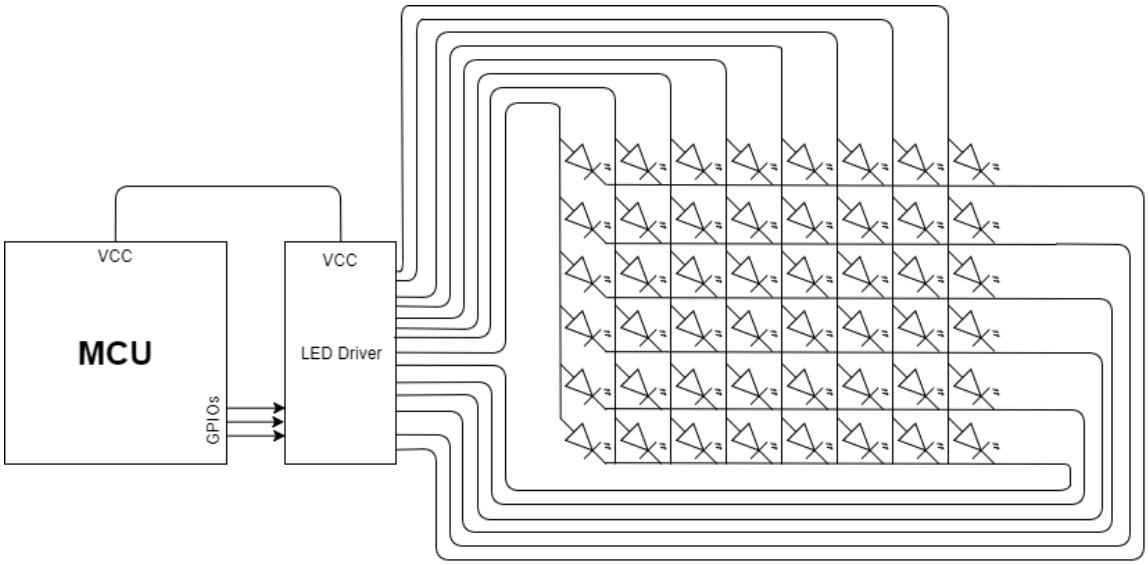


Figure 11. LEDs driven using a Generic MCU and LED driver

Instead, the figure above shows how several LEDs are driven using LED driver chip and a microcontroller. In this configuration the GPIO pins on the microcontroller are used more efficiently as the LED array is interfaced to the LED driver. The configuration above is an example of how a microcontroller, LED driver, and LEDs can be arranged, but architecture may vary.

To efficiently drive the various arrays of LEDs on the Daynight Panel, an LED Driver will be used. The selection process of this driver chip will be described in detail in the next section.

3.2.2. LED Driver Selection Process

LED drivers are manufactured by various semiconductor manufacturers. For simplicity, the search for an LED Driver will focus on the industry leaders: Texas Instruments, ST Microelectronics, and Analog Devices. These manufacturers offer a variety of products easily available through distributors. Additionally, the support provided by these manufacturers in the form of samples, evaluation kits, and extensive user guides will be useful to our troubleshooting.

The basic requirements for an LED Driver for the Daynight Panel are to be able to drive our higher lumen output LEDs while maintaining board space and power efficiency, support serial communication to be able drive each LED individually to support high resolution color mixing, and to be easily scalable by being able to daisy-chain the drivers.

Initially, LED drivers with a high number of output channels were considered due to their perceived ability to save on board space and power consumption of the device. Two potential devices from Analog Device and Texas Instruments are described below.

Table 8. High Output Channel LED Driver Table Comparison [12] [13]

GPN	Manufacturer	Output Channels	Output Current	V_{in} [V]	1ku Price	Packaging
LT8500	Analog Devices	48 (16, RGB)	30 mA	3.3 - 5	2.95	QFN - 56
TLC5955	Texas Instruments	48 (16, RGB)	31.9 mA	3.0 - 5	2.45	QFN - 56

Based on the high number of output channels discussed in the above, the TLC5955 was selected as the LED driver for this project over the LT8500 because of its lower 1ku price.

3.2.3. TLC5955

The TLC5955 is a 48 channel LED driver integrated circuit with built in pulse width modulation [13]. The integrated pulse width modulators on each individual channel is a key feature of this LED driver in our Daynight Panel application. These modulators allow for brightness control of the individual colors of the red, green, and blue diodes that make up the RGB LEDs.

Each output channel is individually controlled via serial communication link from the microcontroller to the driver. The instructions sent to the driver through the serial communication protocol will detail what channels output and at what brightness level is desired on each channel.

Additionally, the TLC5955 is scalable through the use of its serial output pin. The serial output of the initial TLC5955 is connected to the serial input of the next TLC5955. This allows for the driver chips to be daisy chained allowing for output channels. The figure below shows how the TLC5955 would be configured in the Daynight panel to drive our array of LEDs.

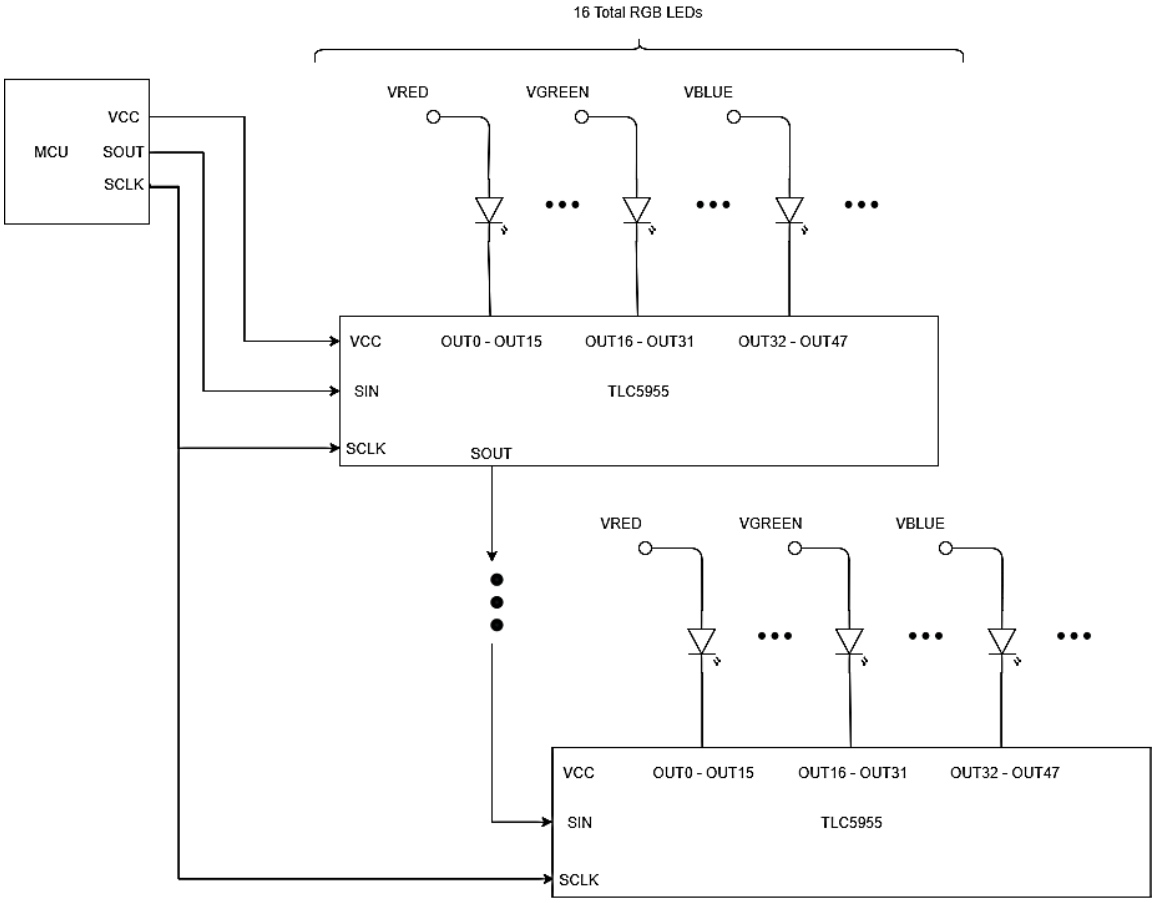


Figure 12. TLC5955 RGB LED Driving Configuration

However, it became apparent that due to the current limitations on the output channels. Our LED forward currents range from 100 mA to 240 mA per LED. A solution was devised to be able to use the TLC5955 with the desired higher current LEDs by dividing the LED forward current among as many pins as necessary.

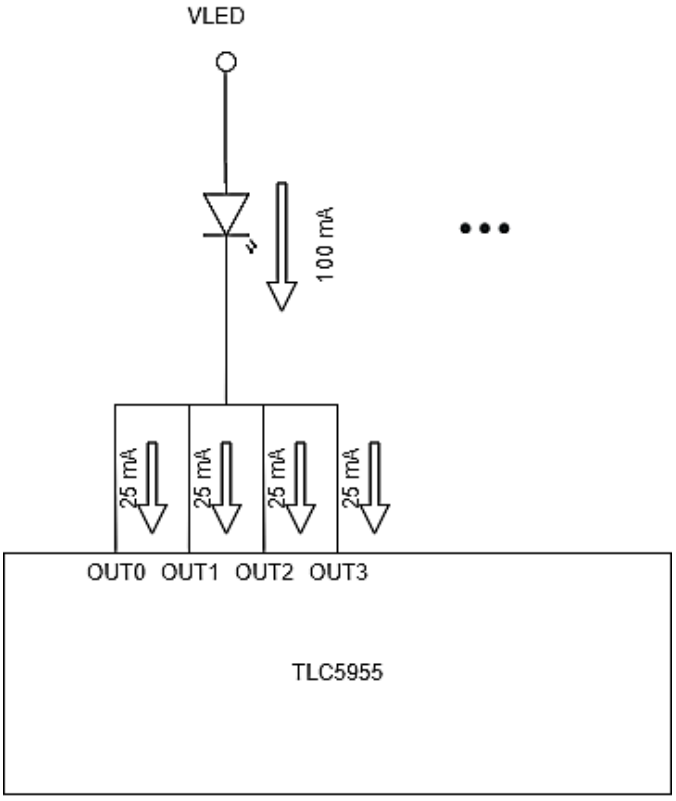


Figure 13. TLC5955 Current Divider Solution

Although this solution would work, it is not practical to make efficient use of board space on the PCB. Additionally, this configuration significantly increases the total LED driver costs for the project. The table below shows a breakdown of how the current division solution would affect the cost and space budget. The calculations seen in this table indicate that using the TLC5955 would require a total of seven LED driver chips at a total cost of $17.10 and a total area of 448 mm^2. This solution is too costly implement as several of these high output channel LED drivers will be required to operate the number LEDs required.

Table 9. TLC5955 Current Divider Calculations

	Number of LEDs	Forward Current [mA]	Number of TLC5955 Channels Required	Number of TLC5955 Drivers Required	Area [mm^2]	Cost @1ku Price
2700K White	12	100	48	1	64	$2.45
6700K White	12	240	96	2	128	$4.90
RGBW	48	100	192	4	256	$9.80
Total	72	N/A	336	7	448	$17.10

By compromising the need for a high number of output channels drivers capable of higher output current can be found. For this search, shift registers designed for LED driving were considered alongside dedicated LED drivers. Shift registers behave similarly to LED drivers and can handle the higher power consumption of the white LEDs. This tradeoff comes at the expense of shift registers not having some of the dedicated LED driver features such as integrated pulse width modulators. The table below compares several LED drivers considered.

Table 10. High Current LED Driver Table Comparison [14] [15] [16]

	Sink Current, Max [mA]	Output Voltage [V]	Max number of LEDs driven
MAX16815/MAX16828	100/200	6.5 - 40	4
STP16CPC05	100	20	16
TPS61169	1800	38	4

Based on the parameters discussed above, the STP16CPC05 shift register was selected as the LED driver for this project. The STP16CPC05 shift register was chosen over the two other higher current LED drivers because of its ability to drive up to 16 LEDs using a single chip with minimal external circuitry required.

3.2.4. STP16CPC05

The STP16CP05 is a 16-bit shift register capable of handling higher voltages and input currents than most multichannel LED drivers [15]. For this reason, it is a

perfect fit for our application of driving our higher current bright, white LEDs. However, as a simple shift register some of the features tailored to LED display applications seen in the RGB LED driver are lost.

The most significant loss is the ability to control the brightness of each individual output channel. This is because the STP16CPC05 does not contain an integrated pulse width modulator. Instead the pulse width modulator on the microcontroller will be used to drive the output enable pin and drive all 16 LEDs at one time. The figure below illustrates this configuration and how it will be used in the Daynight panel.

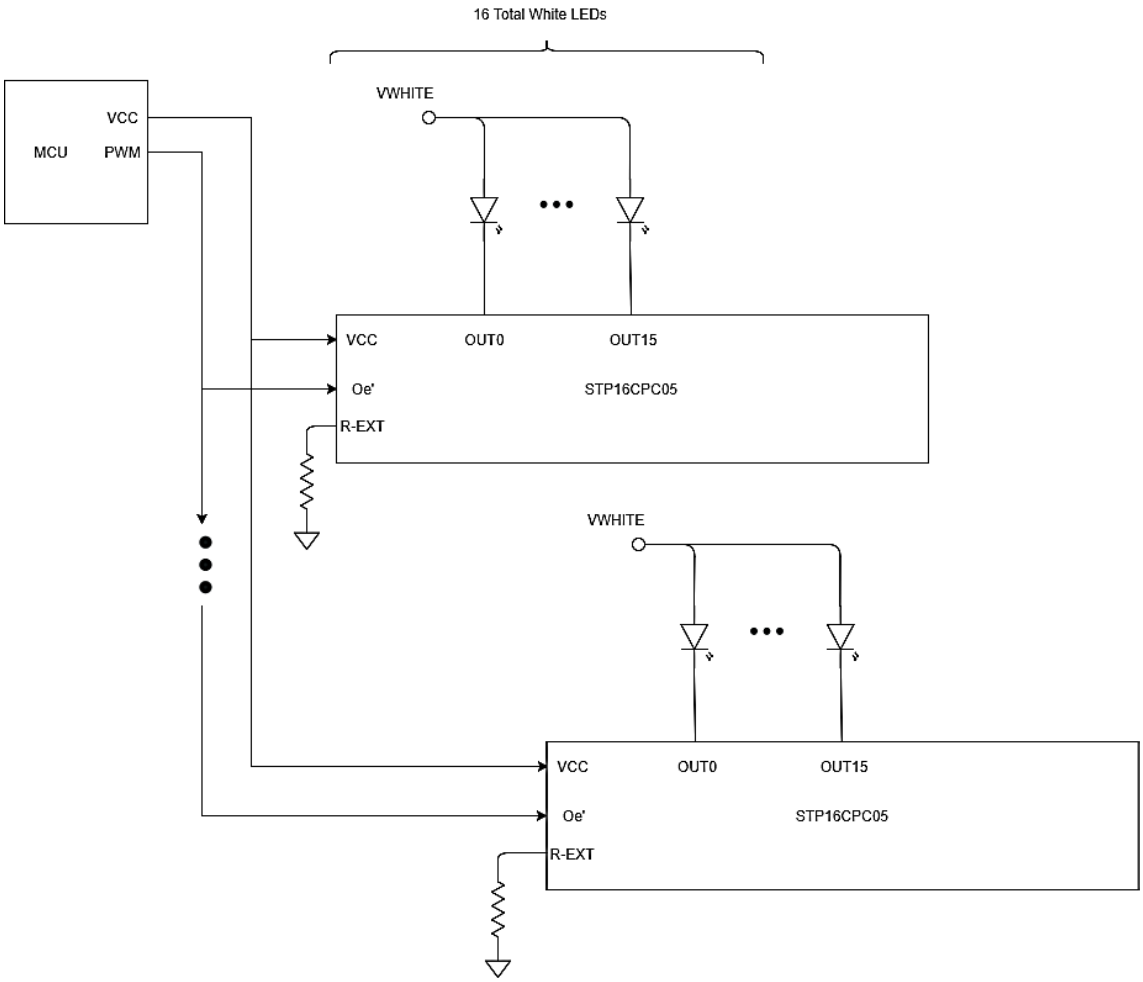


Figure 14. STP16CPC05 White LED Driving Configuration

However, a major tradeoff of the LED drivers described above is that they all lack the ability to individually control brightness of the individual LEDs due to a lack of integrated pulse width modulators. The figure above shows how the brightness would be controlled on the STP16CPC05 by toggling the output enable pin rapidly using the pulse width modulation pins on the microcontroller to control the brightness of all 16 LEDs at the same time.

Ideally, the LED driver needs to be like the STP16CPC05 in terms of output current while maintaining as many output channels as possible and allowing for brightness control of the individual output channels. The table below shows the parameters of an LED driver that met all three of the desired specifications.

Table 11. TLC59116 Parametric Data [16]

	Output Current [mA]	Number of Output Channels	Area [mm^2]	1ku Price
TLC59116	120	16	25	$1.00

The TLC59116 meets all the desired specifications for an LED driver in the Daynight Panel application while maintaining practical board space and efficient costs. The table below details the total impact selecting the TLC59116 will have on the cost and space budget.

Table 12.TLC59116 Calculations

	Number of LEDs	Forward Current [mA]	Number of TLC59116 Channels Required	Number of TLC59116 Drivers Required	Area [mm^2]	Cost @1ku Price
2700K White	12	100	12	1	25	$1.00
6700K White	12	240	24	2	50	$2.00
RGBW	48	100	48	3	75	$3.00
Total	72	N/A	84	6	150	$6.00

Based on these calculations above the TLC59116 is the selected LED driver to be used on the Daynight Panel. The main point illustrated by the table of calculations written above is that this solution can meet the design specifications at a much lower area and cost. These calculations can be directly compared to the table of TLC5955 calculations. The TLC59116 solution requires 150 mm^2 of printed circuit board space for a total cost of $6.00. Comparatively, The TLC5955 requires 448 mm^2 of printed circuit board space at a cost of $17.10. Ultimately because the financial cost and printed circuit board space of the TLC59116 is less than that of the TLC5955, the TLC 59116 was selected to be the LED driver for the Daynight Panel.

3.2.5. TLC59116

The TLC59116 is a 16-channel current sink LED driver capable of driving 16 LEDs using a constant current source rated for an absolute maximum current of 120 mA per channel [16]. To achieve brightness control, the TLC59116 employs analog dimming using an external current resistor and digital brightness control by use of integrated pulse width modulators. The brightness resolution is 8 bits or 256 steps for the digital brightness control. The individual output channels are controlled by sending instructions from the microcontroller to the TLC59116 over an I^2C bus. The bus is composed of three IO pins and a single power pin coming from the microcontroller to establish a bus. The drivers would each connect to this bus and receive instructions from the controller on how to drive the LEDs. The usefulness of this bus and the different configurations are discussed in the Design section of this document.

3.3. Power Supply

The Daynight panel will be powered by connecting its power supply to standard mains power outlet. In the United States, the power outlet outputs a standard 120 Volt, alternating current at a rate of 60 Hertz through a NEMA connector [17]. This alternating current be sent to a power supply unit that will rectify and step down the voltage to an appropriate, usable DC voltage. This simple process has several degrees of freedom in its design and constraints imposed by standards and the Daynight panel that will be discussed below.

The Daynight panel will use a 120W AC-DC power supply. This number comes from the rounding up of the highest voltage needed in the circuitry. The greatest voltage needed on the Daynight panel is the 9.4 V forward voltage of the warm white LED, so the output voltage of the power supply is rounded to 12 V. The input current is determined by finding the sum of the major sources of current draw. The major sources of current draw are the LEDs. The table below shows the how the maximum current draw is calculated.

Table 13. Total Current Draw Calculation

Manufacturer Part Number	Color	Current per Diode [mA]	Number of Diodes	Total Current [A]
L130-6580003000W21	6500K – White	120	12	1.44
SPMWH1229AD7SGWMSA	2700K – White	100	12	1.20
CLQ6A-TKW-C1L1R1H1QBB7935AA3	RGBW	100	48	4.80
			Total	7.44

The current draw of the driver electronics and the microcontroller can be neglected as it will never exceed more than 1 A. To ensure safety, the total current is rounded up to 10 A. The resulting desired power supply is a constant voltage power supply of 12 volts DC with varying current draw not exceeding 10 A.

3.3.1. Connector Selection

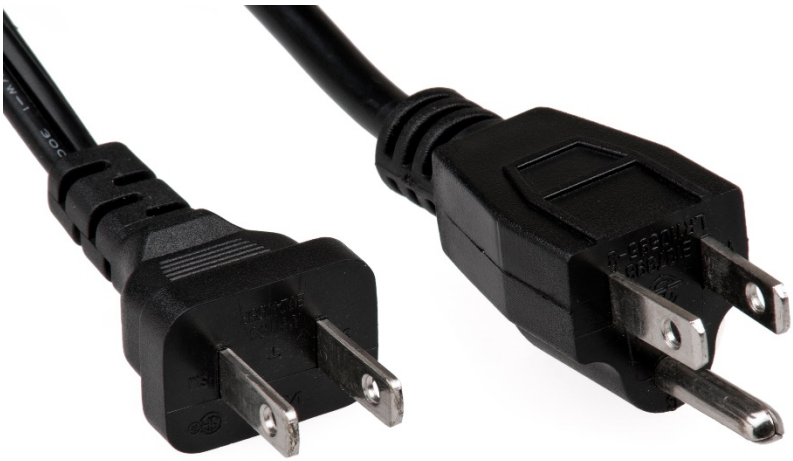


Figure 15. NEMA 1-15 (left) versus NEMA 5-15 (right) Plug Comparison [17] [18]

The NEMA 1-15 is a two-prong power connector that serves as one of the two standard power connectors in North America and Japan. This plug is ungrounded and rated for a maximum current of 15 Amperes. The NEMA 5-15 is a three-prong power connector that serves as the other standard power connector. Similarly, this plug is rated for 15 amperes. The third prong on the NEMA 5-15 plug serves a

grounding pin connected to earth ground. The grounding pin is seen in high current applications such as household appliances or personal computers. In these high current applications any leakage current from the circuitry of the devices passes through the grounding pin safely to earth ground.

The Daynight panel will be consumer electronic device meant for home or office use. The panel will run for several hours or potentially days at a time while being mounted to a wall. In a high current application, these operating conditions would merit a grounding pin to prevent user shock or a fire hazard caused by leakage current. The Daynight panel has an expected maximum operating current of no more than 10 A at 12 V and will be in the home operating for several hours at a time, so a NEMA 5-15 plug will be used.

3.3.2. Internal versus External Power Supply Unit

Figure 16. An example of a typical external power supply unit

To maintain the neat, aesthetic appeal of the Daynight panel as a piece of home or office décor, the Daynight panel will have an internal power supply. Choosing to integrate the power supply into the body of the Daynight panel comes with some tradeoffs such as increased weight and thermal heating concerns as well as constraints to the overall size of the power supply unit.

Additionally, choosing to use an exterior power supply like the one shown in the figure above may become a safety hazard if the user attempts to dangle the power “brick.” The power brick could fall and damage property or persons. By engineering the power supply into the Daynight panel this potential hazard can be eliminated completely.

3.3.3. Off-the-shelf versus Custom Designed Power Supply Unit

The power supply unit for the Daynight Panel could either be custom designed using schematics or an “off the shelf” power supply unit can be purchased. Both options for obtaining a regulated DC voltage have tradeoffs financially, technically, and physically.

Buying an off-the-shelf power supply unit has the advantages of speed and reliability. Quick online research reveals countless 120W power supplies with an output of 12 VDC capable of 10 A current draws. These power supplies typically come with a warranty and marks of certification ensuring safety and emissions performance. Some of these safety features are reverse polarity protection, short circuit protection, and proper electromagnetic filtering components. Additionally, an off-the-shelf power supply can be delivered and tested easily and quickly ensuring its performance. The table below compares a few 120W power supply units that meet our design specifications.

Table 14. Off-the-shelf Power Supply Units [19] [20] [21]

	Output Voltage [VDC]	Maximum Current Output [A]	Dimensions	Cost
Aiposen S-120-12	12	10	6.25 inch x 3.85 inch x 1.65 inch	$17.99
Mean Well DR-120-12	12	10	2.6 inch x 3.9 inch x 4.9 inch	$37.89
Padaresy S-120-12	12	10	7.87 inch x 3.93 inch x 1.77 inch	$13.99

These power supply units have multi-voltage inputs with binding posts for line, neutral, and ground. A NEMA plug and cord are crimped and soldered to the proper posts seen in the figure below to input AC voltage, and similarly connectors crimped and soldered to the output posts to output voltage to the PCB. The power supplies above include similar protection features. The Aiposen is the smallest power supply in total volume and comes at a reasonable price. For these reasons it is the preferred off-the-shelf power supply for this project.

Figure 17. Aiposen S-120-12 Connections. Permission requested from Amazon Inc [19]

The figure below shows how a custom linear power supply design for the Daynight panel could be implemented. The power supply is comprised of a step-down transformer feeding into a full bridge rectifier comprised of four diodes specified for low frequency, high current applications. After rectification, the rectified power signal is filtered by a high capacitance capacitor bank. Finally, the filtered DC signal is fed into various low dropout linear regulators each providing enough current to serve their respective power rails on the printed circuit board.

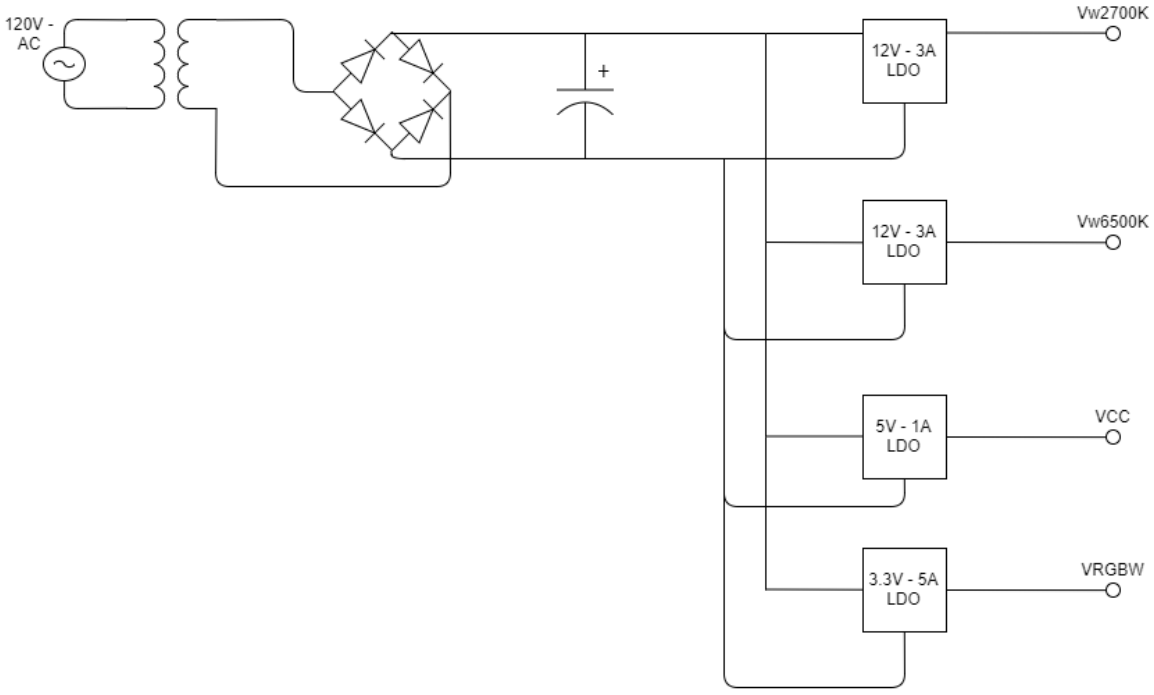


Figure 18. Simplified Power Supply Design

Additionally, several safety mechanisms could be implemented using more external components. To provide overcurrent protection, a consumable fuse or circuit breaker could be put in series with the power supply unit. To provide reverse voltage protection, a protective Zener diode could be placed in a shunt configuration to the power supply unit. Short circuit protection could be implemented using an additional circuit using more external circuitry. However, every protection feature added only further increases component cost and adds more thermal heat. These additional costs may be more than the cost of simply buying an off-the-shelf power supply unit.

3.4. Voltage Regulators

The power supply unit will produce a main 12 volt power rail that will me connected to the Daynight Panel's printed circuit board. In order to reduce and steady the voltage applied to the LEDs, voltage regulators will be used to make power rails for the various LEDs and electronics.

3.4.1. Linear Regulators

Linear regulators provide simplicity of design by only requiring proper input and output filtering capacitors. Additionally, linear regulators produce little electromagnetic interference due to their lack of switching resulting in a less "noisy" output signal. However, the output of a linear regulator is limited by the difference between the input voltage and the dropout voltage. Another concern of supplying the power rails using linear regulators is the significant amount of heat dissipated by the integrated circuit. The table below compares various linear regulators considered for the power rail design.

Table 15. Linear Regulator Table

	LM1084	LT3083	NCP57302
Manufacturer	Texas Instruments	Analog Devices	ON Semiconductor
Input Voltage, Max [V]	29	23	18
Dropout Voltage [mV]	1300	310	300
Output Voltage, Max [V]	27.5	22.5	17.5
Output Current, Max [A]	5	3	3

3.4.1.1. LM1084 Low Dropout Regulator

The LM1084 is a low dropout linear regulator with a wide input voltage range. This large input voltage range is desirable in our design because it is able to connect directly to the power supply unit. Its output current range of up to 5 amperes is the largest of typical low dropout voltage regulators. This output current range should be sufficient to power the RGBW LED power rail and the electronics power rail, but additional LM1084 would be required to split the load for the white LEDs.

3.4.2. Switching Regulators

Switching regulators are able to use external storage elements such as inductors to more efficiently regulate voltage and power rails. Additionally, the switching regulators can be configured to “boost” voltage as well “buck” voltage. Consequently, the proper design of switching regulator power rails is more complicated. To simplify the design process the online Webench Power Architect software is used design a switching regulator solution based on the output of the Aiposen S-120-12 power supply unit. The figure below shows the resulting power rail design produced by the Webench Power Architect.

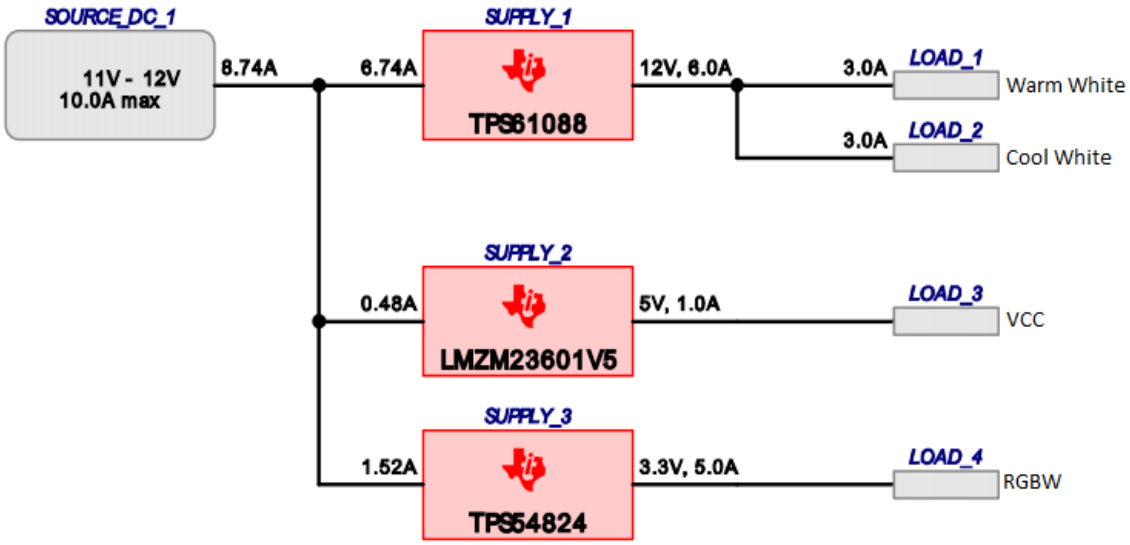


Figure 19. Switching Regulator Power Rail Design

3.4.2.1. TPS61088 Boost Converter

The TPS61088 is a high current boost converter integrated circuit. The converter's wide input voltage range makes it desirable for this design because it is able to connect directly to the power supply unit at 12 volts. It is able provide up to 10 amperes of current. This high output current allows for the TPS61088 to drive the white LED power rail.

In order to configure the TPS61088 as a "boost" converter capable of supplying six amperes of load current at 12 volts from the power supply unit several external components are needed. The total bill of materials for this power rail has a count of 17 discrete components including the regulator at a cost of $3.52.

3.4.2.2. LMZM23601 Step-Down Converter

The LMZM23601 is a step-down or "buck" converter. It has a wide input voltage range that allows for direct connection to the power supply unit at 12 volts. This cost and size effective regulator provide a maximum output current of one ampere. This regulator will be sufficient to handle the load needed to power the LED driver chips and the microcontroller.

To configure the LMZM23601 as a step-down converter capable of supplying one ampere of current at five volts only a few external components are needed. The total bill of materials has a count of four discrete components including the regulator at a cost of $3.41.

3.4.2.3. TPS54824 Step-Down Converter

The TPS54824 is a step-down or “buck” converter. It has a wide input voltage range that allows for direct connection to the power supply unit at 12 volts. It is able provide up to eight amperes of current. This reasonably high output current allows for the TPS54824 to drive the RGBW LED power rail.

To configure the TPS54824 as a step-down converter capable of supplying five amperes of current at 3.3 volts several external components are needed. The total bill of materials has a count of 19 discrete components including the regulator at a cost of $3.28.

3.5. Communication

The Daynight panel is primarily dependent on a driver, microcontroller, Bluetooth module, LEDs and an IOS Application. While all these components need to be correctly circuited to each other they also need to work synchronously. Communication will be looking at all the different types of signals each device can emit to ensure the panel is running in a stable mode and responding correctly to user input. Communication covers all the signals allowing the devices to speak with one another. The communication must be carefully monitored to ensure the correct message is sent and received within the appropriate time span. For the Daynight panel, several protocols can be established for communication between devices. In the following sections the appropriate block diagram configuration, timing diagram, protocol requirements and shortcomings are investigated.

3.5.1. Microcontroller Master

The microcontroller is the brain of the panel and is packed with many uses, but this feature results in fewer pins for all of it applications. Therefore, additional devices are required to manage the LED color and brightness. These devices require different communication protocols and configurations to function. The following section looks at the different modes in which the microcontroller can communicate and how connected devices must be configured to allow the system to function.

3.5.2. Modes of Communicating

To control LEDs three methods are investigated. They each have their own benefits and drawbacks but can successfully drive the LED array. The communication protocols are SPI, I2C and Serial to Parallel shifting.

3.5.2.1. Inter Integrated Circuit (I2C)

Since the I2C protocol is proprietary, some manufactures provide two-wire serial interfaces which generally are the same thing. The microcontrollers under consideration use a two-wire interface and have added features which make

design easier. The microcontroller will be handling a lot of devices and the data can only travel synchronously in one direction along the bus. The communication is powered by an outlet and will be occurring in a high current and voltage environment. This environment opens the door to a lot of noise in the data transmission. Both the ATmega16u4, ATmega32u4 and the ATmega328P provide noise suppression circuitry to keep the data and clock lines stable so that the chances of acknowledgment from the master or slave increases. The two-wire serial interface also provides an address length of 7 bits which allows 128 devices to be controlled by 1 controller [22].

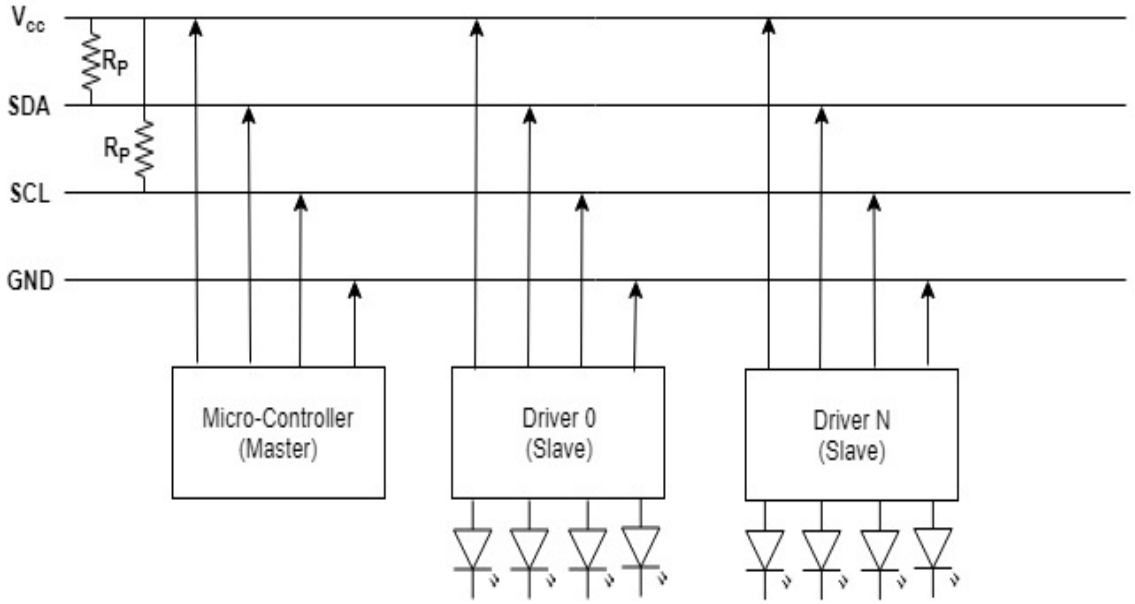


Figure 20. General Master Slave Configuration

The image above shows the general configuration for control in the daylight panel. This circuit can have the 128 drivers connected to the clock and data lines but the requirements for devices on the line to operate can reduce the number of peripheral devices connected to the bus. All these devices can simply be wired and coded to run with simply the addition of the pull up resistors. Despite the great number of devices that can be added to a single bus there are limitations to performance that may occur when the clocks and logic levels of all the different devices vary too much.

3.5.2.2. Serial Peripheral Interface

Serial Peripheral Interface also known as SPI is another viable form of communication for the drivers and microcontroller on the Daynight Panel. It is faster than I2C and also scalable. The main drawback is the number of wires a serial peripheral interface needs to function. Depending on the scale of the system this can become a hindrance instead of an aid.

In Serial Peripheral System a master and slave relationship between devices sending and receiving information. The master provides the SCLK, Master Out Slave In (MOSI), Master In Slave Out (MISO) and Slave Select Pins to each slave that is connected. The slaves send responses back to the master in slave out pin and receive information from the master out slave in port. All data transfer is governed by a single clock.

The ATmega is capable of SPI, but not all drivers are configured to operate as SPI slaves [22].

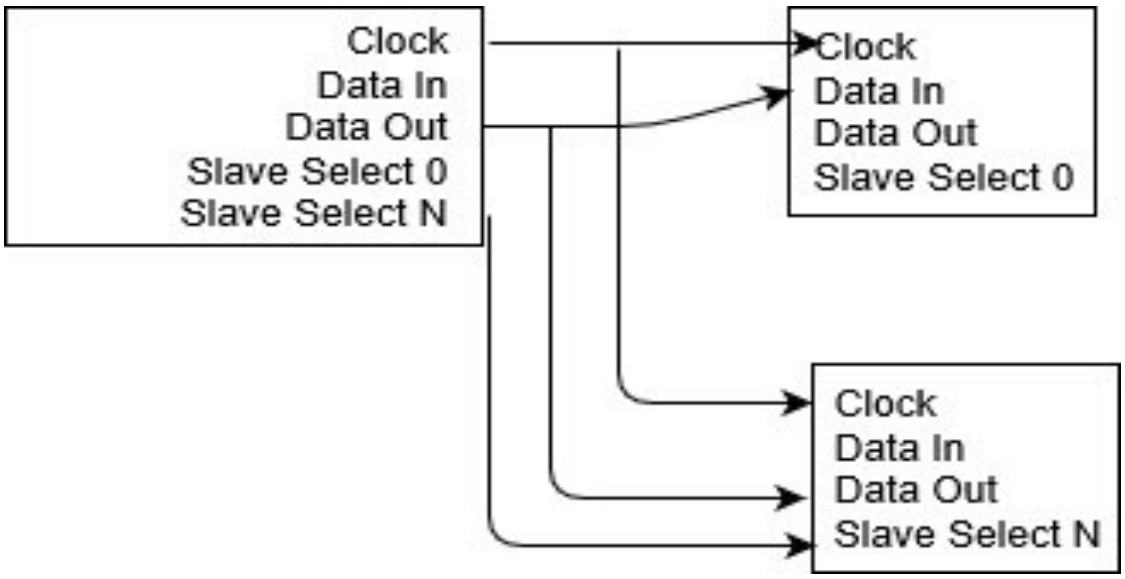


Figure 21. General Function for SPI

3.5.2.3. Pulse Width Modulation

While this is a simple method for generating a waveform, pulse width modulation plays a major role in the most complex driver to the simplest shift register. It is necessary to know how many pulse width modulation pins are available and how frequency is regulated. The Atmega328 has 6 pulse width modulation channels while the Atmega16U4-32U4 has 4 pulse width modulation channels. These microcontrollers offer 3 pulse width modulation modes but the only relevant one is the Fast PWM Mode [22].

Since the pulse width modulation pins are 8 bit the microcontroller counts to a predefined value and the duty cycle is determined by some value between 0-255. The fast mode is preferred because it can alter the LED values at discrete intervals but faster than the eye can see. The frequency is calculated using the following formula in Appendix D-4.

The formula above provides 3 sets of options for modulation frequency. They are based on the operating logic level since the Clock changes with the logic levels of the system.

There are 3 different operating voltage ranges for the Atmega328P since a range for the system to operate in has not been defined. The following tables look at the pulse width modulation frequencies the controller can output within each range.

The formula detailed in Appendix D-4 is used to fill the following tables. They show that the options for modulation are discrete and depend on the voltage range. Therefore, this parameter will have to be carefully chosen to define logic levels for the system. Fortunately, the microcontroller is capable of working within several ranges, but it is most flexible within the first two ranges shown in the tables below.

Table 16. PWM Frequency Range, 1.8-5.5V

1.8V – 5.5V		
PWM Frequency	Clock Frequency	N
15625.00	4000000	1
1953.13	4000000	8
244.14	4000000	64
61.04	4000000	256
15.26	4000000	1024

Table 17. PWM Frequency Range, 2.7-5.5V

2.7V – 5.5V		
PWM Frequency	Clock Frequency	N
39062.50	10000000	1
4882.81	10000000	8
610.35	10000000	64
152.59	10000000	256
38.15	10000000	1024

Table 18. PWM Frequency Range, 4.5-5.5V

4.5V – 5.5V		
PWM Frequency	Clock Frequency	N
78125.00	20000000	1
9765.63	20000000	8
1220.70	20000000	64
305.18	20000000	256
76.29	20000000	1024

For most of the devices that will be controlled by the microcontroller's clock only the first two options are feasible. All logic levels provide a good frequency range to work with. The pulse width modulation would be used along with shift registers to regulate the brightness of the white LEDs.

3.6. Peripheral Device Requirements and Capabilities

3.6.1. TLC59116 – LED Driver

The driver uses the I2C protocol for controlling LEDs. This method is beneficial mainly because it caters to configurations with several masters and slaves while keeping down the wire size. This protocol can only be used with devices that support it and to qualify for support a device needs an SDA line, SCL line and in most cases they should all refer to the same V_{cc}. The TLC59116 Chip is designed to optimize communication using I2C and is a viable option for LED driving because it can be daisy chained to scale the design up or down [23].

The functional block diagram in the datasheet shows all the major decision-making blocks in the driver chip [16]. Most of them describe parameters that will need to be altered to meet design specifications.

3.6.1.1. TLC59116 Specifications of Note

These are the main parameters which govern basic connections to the microcontroller and they are all within operating range. The controller will be providing a the SCL, SDA, address, V_{cc} and GND as inputs to the driver. The driver takes the inputs and conducts its own calculation to change the LEDs. Controlling the driver requires understanding how to alter the major blocks in its functional diagram shown below.

Table 19. TLC59116 Specifications of Note [23]

Parameter	Min	Max
Supply Voltage	3V	5.5V
High Level Input Voltage	2.1V	5.5V
Low Level Input Voltage	0	1.65V
Supply Voltage to Output Pin	-	17
Output Current per Channel	5mA	120mA

The driver chips provide three operating modes, Standard, Fast and Fast Plus [23]. These modes are based on the SCL clock frequency in which they can operate.

Table 20. TLC59116 - Essential Timing Requirements [23]

	Standard Mode	Fast Mode	Fast Mode Plus
SCL Clock Frequency	0 – 100kHz	0 – 400kHZ	0 – 1000kHz
t_{BUF} (time between stop and start condition)	4.7μs	1.3 μs	0.5 μs
t_{SP} (Pulse width of spikes to be filtered)	50ns	50ns	50ns
t_{HD} for Start	4ns	0.6ns	0.26ns
t_{HD} for Data	0	0	0
$t_{VD,ACK}$ Data Valid Acknowledge Time	0.3-3.45ns	0.1-0.9ns	0.5-0.45ns

The table above shows some essential timing requirements for the driver. The rest will be analyzed in timing diagrams. The values above are compatible with the ATmega chips. But depending on the what is being transferred there might need to be some clock stretching involved. Of all the 3 modes the ATmega chips under consideration can only handle the standard and fast mode from the driver.

3.6.2. Bluetooth LE Module

The Bluetooth module that has been chosen for this project is the Adafruit Bluefruit LE SPI Friend. It is a Low energy Bluetooth module. The Adafruit model is preferred because of the cohesiveness of design and software support. This standalone module is the channel between the application and the controller. It is interfaced to the controller using a serial peripheral interface. This method of communication allows information packets to be sent individually once an interrupt is sent from the controller through the Bluetooth module. Essentially the module simply has to be powered and wired correctly. Once it is turned on it is programmed using AT commands. There will have to be some coordination between the iOS application, the Bluetooth AT Commands and the controllers code.

3.7. STP16CP05 – 16 Bit Shift Register

This shift register has been designed to handle higher currents and voltages than a regular register. It can drive the lights with higher loads, but the degree of control is lessened. It is only capable of altering 16 bits at a time.

The table below looks at the most pertinent parameters immediately concerning the design. These parameters would have to be considered along with the microcontrollers ratings to ensure all devices are operating within a safe range.

Table 21. Maximum Ratings

Parameter	Min	Max
Supply Voltage	3V	5.5V
Output Voltage	-0.5V	20V
Output Current	3mA	100mA
Serial High Output Current	-	1mA
Serial Low Output Current	-	-1mA
Input Voltage	-0.4	V_{DD}+0.4V
GND Terminal Current	-	1600mA
Clock Frequency	-	30MHz

In the overall design this chip would only talk to other shift registers and the controller. For the longevity of the chip it should run at 30 MHz frequency, but this is higher than the controller running at its highest voltage range. The supply and input voltage are also within a range that the controller can communicate with and understand.

The functional diagram shows the limitation of the shift register its ability to control the LEDs is completely dependent on the signal that will be coming in from the microcontroller. It will take in one clock signal to shift serial data in. The shifted data then gets latched and once the microcontroller enables the OE pin the data is displayed via LED. The diagram shows that any changes will have to alter all 16 connected LEDs. The only place to regulate LED behavior is through the external resistor which determines the constant current going to the LEDs and by supplying the OE pin with a modulated signal to bring the connected LEDs to the correct brightness. This means brightness can only be 1 value for each shift register and the number of differing brightness levels is limited by the number of pulse width modulation pins available from the controller. One of the benefits of this shift register is the number of LEDs that can be run on a single output port. The output port voltage is 20V so there is no need to use too many registers to control a lot of LEDs.

3.8. Wireless Networking

Wireless networking plays a crucial role in our project in order to facilitate communication between the Daynight app and the Daynight Panel. Here, different

kinds of wireless networking and communication are considered and discussed to find the best method of transmitting information across the Daynight Panel's peripheral devices.

Wireless networking and wireless communication are ways to establish a medium for communication between two terminals. Telecommunication involves an exchange of signals and messages (usually as data) with the help of technology. Area networks are represented by a network of terminals across a limited space and are supported by using wireless networking techniques. Area networks can range in size to cater to the needs for each application.

Two of the most common area networks are Wide Area Networks (WANs) and Local Area Networks (LANs). WANs cover large regional geographical space and are used mostly by government, education systems, or businesses to exchange data to necessary personnel [24]. LANs are specifically limited in scope to include residential buildings, business offices, or campus buildings. WANs are created by connecting regional networks (usually purchased by the entity aiming to create the WAN). LANs are created and sustained by utilizing Wi-Fi (for wireless connection) and Ethernet (for wired connection) as the most common technologies.

Another type of area network that is of particular interest in the scope of this project is the Personal Area Network (PAN). PANs limit scope to a person's surrounding workspace. Similar to WANs and LANs, a PAN uses technology to exchange data as a means for telecommunication. Wi-Fi and Bluetooth are technologies that support small networks and by efficiently using energy for a small network, for example, by not over-broadcasting or requiring energy to transmit long-distance signals. An advantage of PANs is that they have small range, and thus over a large radius with multiple spaced-out PANs, they can exist without interference to the user.

3.8.1. Bluetooth's Role in Networking

Bluetooth is a wireless telecommunication technology that uses radio waves to transmit data within short distance [25]. Common use for Bluetooth includes phone-to-speaker transmission of audio data, to play music or to take a phone call handsfree. Bluetooth and its organization of standards is maintained and developed by the Bluetooth Special Interest Group (Bluetooth SIG).

The Bluetooth SIG has standards for qualifications that new devices and electronics must pass before declaring compliance with membership agreements laid out by the Bluetooth SIG. Bluetooth technology is backed in support by the Institute of Electrical and Electronics Engineers (IEEE). IEEE initially standardized Bluetooth with the IEEE 802.15.1 standard in 2001, and again in 2005 [26]. As of Bluetooth 3.0, Bluetooth saw support for high-bit traffic in the 802.11 standard, which is a media access control layer used for computer communications (Wi-Fi).

This standard—in particular, 802.11b—restricted energy use and bit exchange which is perfectly suited for use in PANs like Bluetooth.

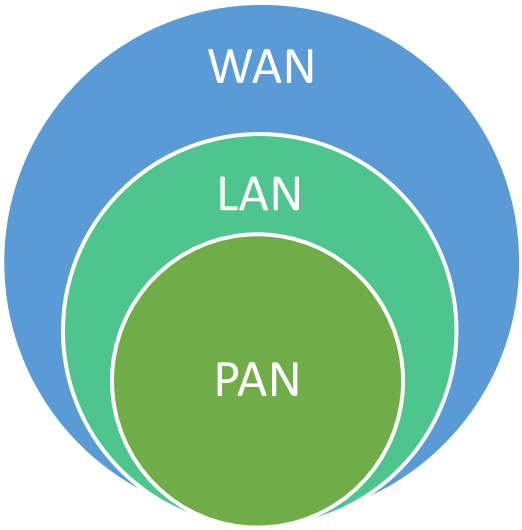


Figure 22. Wireless networking types and their ranges

Bluetooth has two classifications of technology: Bluetooth Classic and Bluetooth Low Energy. Bluetooth Classic was designed with the initial intention of creating a constant stream of data within a personal working area. An example of this kind of application is for exchanging audio from a cell phone to an in-ear device. Throughout the duration of the phone call, audio data is bring exchanged from both devices—or terminals. Such a constant stream of data brings up concerns about long-term energy consumption, which is important to consider when using devices with limited battery capability [27].

Bluetooth Low Energy (also sometimes called Bluetooth Smart, or Bluetooth LE) was introduced alongside Bluetooth 4.0 in 2011 [28]. Bluetooth LE aimed to provide similar functionality to smaller devices that didn't require a continuous stream or exchange of information. A clear benefit to devices that desired this kind of support is that these operations aren't energy-intensive. These devices would only require infrequent and small packages on information. An example of this behavior in relation to the Daynight app and the Daynight Panel is sending a one-time control over Bluetooth LE. During the rest of the day, the Daynight app and Daynight Panel do not require any more interaction since there is no information ready to exchange.

Bluetooth LE devices remain in a sleep-like state until a communication is initiated, even though the connection can be maintained. Since Bluetooth LE has such little overhead, transactions of information are performed very quickly; Bluetooth LE

takes advantage of high data rates that are energy-expensive because they happen infrequently.

3.8.2. Wi-Fi's Role in Networking

Similar to Bluetooth, Wi-Fi is a wireless technology that establishes radio networking. In contrast to aims with Bluetooth, Wi-Fi wavelengths perform optimally in line-of-sight applications. Line-of-sight prorogation describes the propagation of electromagnetic radiation—in our case, radio waves—where the waves travel directly from the source to the destination. Obstructions like buildings, metallic structures, trees, and even the curvature of the Earth can prevent signals from completing their journey at full strength [29]. Wi-Fi is the most common technology used to establish LAN networks and is supported in most personal computer devices [30].

Though there is appeal with a common technology like Wi-Fi, online servers or databases would make the software communication process easier between the Daynight app and the Daynight Panel. Additionally, the operation of the panel would directly rely on availability of Wi-Fi networking in the home and would not be available for users that do not have in-home network.

The Wi-Fi Alliance was established in 1999 to create standards for how the technology could be used. Similar to the way that the Bluetooth SIG manages developments for Bluetooth, the Wi-Fi Alliance—consisting of over 375 companies—passes certifications and maintains the Wi-Fi brand (trademarked by the Wi-Fi Alliance). Approved and certified products are subject to rigorous testing to ensure backward compatibility and interoperability. The Wi-Fi CERTIFIED seal indicates to consumers that a product has been tested and approved by the Wi-Fi Alliance [31]. If the Daynight Panel has intentions to enter the market in this capacity, we would be subject to the described approval and certification processes.

Figure 23. Wi-Fi CERTIFIED Seal [31]

3.9. Finding the Right Microcontroller Hardware

The search for the correct hardware began with keeping a few key things in mind. Here are specific characteristics the hardware must support:

- Bluetooth LE module
- Microcontroller
- Hardware serial, hardware I2C, and hardware SPI support
- USB-to-Serial program capability

For the purposes of testing, having a Bluetooth LE module on-board with a logic chip was acceptable and preferred, but for the final project these cannot be manufactured together. For such an instance, a standalone Bluetooth LE module also became necessary.

There are several microcontrollers on the market capable of handling the processing for the daylight panel. The processors under consideration are listed below along with relevant specifications.

Table 22. Comparison of ATmega Chips

	Raspberry Pi 3B	MSP432	ATmega328P
Manufacturer	Raspberry Pi	Texas Instruments	Atmel
Flash Memory	-	256 Kilobytes	32 Kilobytes
USB Peripheral	Yes	Yes	No
PWM Pins	1	16	6
Speed	1.2GHz	48MHz	20
Timers	2 x 8-bit, 2 x 16-bit	2 x 8-bit, 2 x 16-bit	2 x 8-bit, 1 x 16-bit
Pin Count	44	40	32
Price	$35.00	$1.69	$1.46
Operating Voltage	0-5V	1.65V-3.7V	1.8V-5.5V
Max Current (I/O)	2A	2mA	40mA
I/O Pins	40	40	23

The ATmega chip is considered due to its compatibility with Arduino Software. Flash Memory in a chip describes the amount of physical storage space that is available on the chip for hosting software applications. 16 Kilobytes is a small enough amount of space that it will be avoided for the purpose of this project. Speed in millions of instructions per second (MIPS) is a quantitative measure of how fast a computer can execute one million instructions. Though some believe this measure to be obsolete due to most modern program speeds being heavily controlled by input and output (I/O) processing speeds [32].

The USB Peripheral is an interface that allows software to be loaded on to the chip via micro-USB. This interface is important to have as it allows for quick building

and running of software that will be added to the chip (this is that the ‘U’ stands for in the name ATmega32u4).

The PWM Pins available on the ATmega chips allow for the use of a pulse width modulation signal to drive the LEDs. The pulse width modulation pin outputs a signal of varying duty cycle set by the software programmed onto the microchip. This varying signal toggles the LED on and off at a frequency high enough that the user is unable to distinguish the switching, but rather see a dimming effect. This effect can be used to directly drive LED brightness or used on the output enable pin of a driver chip to dim an entire set of LEDs.

One of the major features of the ATmega chip for our application is its operating voltage. The Daylight panels control system will consist of several peripheral devices talking to the controller and since the mode of communication will be I2C all devices need to be able to connect to the Data and clock bus. Therefore, all devices connected on the bus should be able to communicate at the bus’s voltage without a decent margin of safety to ensure there are no floating values or misconstrued signals to corrupt the data being transmitted. The ATmega chip operates in the 1.8V to 5.5V range and considering most sensor and driving devices for a project of this size this is a good range with the option to go a little above or below because the listed range is just a typical value. Regardless, within this range the ATmega can provide a logic high and the logic low values that can work with the most peripheral devices.

The MSP432 is considered because of its versatile features, many of which would help improve our design. One of the main issues faced when designing the panel is the number of general-purpose input/output pins required to control all of the LEDs. The MSP432 has a lot of more general-purpose input/output pins but it is limited by a current of two milliamps. The LEDs that have been chosen draw, at the least, ten times more current than the controller can provide. Despite this limitation, what is really needed are the pins capable of providing a pulse width modulated signal. The MSP432 has the greatest number of pins capable of pulse width modulating. The current limitation can be taken care of by using the LED drivers as planned and the MSP430 microcontroller can drive a lot more of the TLC59116 drivers because it has more pins capable of providing a pulse width modulated output. Since we want our design to be configurable this design would keep future options open by giving the chance to run a lot more LEDs per controller compared with the other two options.

The one major drawbacks to the MSP430 that cancels it as a choice for being used on the Daylight Panel is the voltage range in which it can work. The MSP430 only safely functions between the voltages of 1.65 and 3.67. Since one of the main marketing points for this chip is that it is ultra-low power this limitation makes sense. The voltage range is too small considering the growing number of peripheral devices that are going to be receiving instructions from the daylight panel’s controller. The MSP430 proudly offers Phillips I2C instead of a two-wire

interface but since most of the devices that will connect to this bus operate on logic at the 3.3V to 5.5V using the MSP430 would become a crippling design constraint.

The Raspberry Pi 3B is an all in one package capable of just about anything. It can handle processing for the Daylight Panel. The Raspberry Pi model 3B is a little computer with its own Bluetooth model, Wi-Fi capability and storage. This option has the greatest number of pins and the is the hardiest of all the three controllers because it can handle two amps at its ports. Despite all of these features the most important capability for the Daylight panel is the ability to drive the LEDs and this feature requires pulse width modulation. The Raspberry Pi model 3B only provides 1 pin that can output a pulse width modulated signal. Of course, there are work arounds to making this signal elsewhere but considering the cost of 1 Raspberry Pi adding additional components is almost counter-intuitive. The Pi is designed to handle most of an applications processing and storage and comes with several components designed to meet its advertised features. Using the Raspberry Pi, we would be throwing away all those capabilities in favor of a more powerful computing system. Another major drawback of the raspberry pi is that is does not have flash memory, so everything is stored in the SD card. SD cards can easily become corrupted. The other controllers can have a program saved to the flash memory and as long as external conditions meet the microcontrollers' specifications the chip would continue doing it's programmed task. The Raspberry Pi is too cumbersome and expensive for our application.

The ATmega328P is the controller that has been chosen for our application. It is just the right size and does not limit the panels capabilities while still meeting budget constraints.

Table 23. Bluetooth LE Module Comparison

	Bluefruit LE Friend	BLE Shield v2.1	PAN1740
Manufacturer	Adafruit	RedBear	Panasonic
Bluetooth LE Support	Yes	Yes	Yes
USB Peripheral	No	Yes	No
SPI Support	Yes	Yes	Subset of SPI+
UART Support	Yes	16	Yes
I2C	Yes	Yes	Yes
Weight	3 grams	38 grams	n/a
Price	$17.50	$19.90	$20.89

3.10. Finding the Right Software

Integrated development environments (IDEs) are software applications that are all-in-one solutions for programmers and software engineers. IDEs are comprehensively equipped with a plethora of tools for writing, debugging, testing, simulating, archiving, or deploying software applications that are written in code. Some IDEs like Xcode by Apple Inc. provide a graphical user interface (GUI) for creating and modifying UI layouts for software applications [33].

Arduino Software is an IDE used to write software that is intended to be run on an Arduino board. Some logic chips like ATmega32u4 come manufactured on Arduino boards can be purchased separately. These chips support loading software that is written in Arduino Software and can be put on the chip. This can be done through transfer with the board's USB peripheral interface through micro-USB that is already supplied on a board or added to a custom PCB manually.

Xcode is an IDE available on macOS used to develop software for iOS, macOS, watchOS and tvOS. It supports source-code for C, C++, Objective-C, Objective-C++, Swift, AppleScript, Python, Java, Ruby. Xcode's first main release to the Mac App Store was in 2003 and is available for developers to download for free [34].

3.11. Programming Languages for iOS

Objective-C is a superset of the C programming language that introduces object-oriented concepts. This means a lot of the syntax from C is inherited in Objective-C, with some extra added for handling objects and methods. Objective-C was developed in the 1980s to create the operating system NeXTSTEP by NeXT Computer. NeXT Computer was acquired by Apple in 1997, and Objective-C was then used to derive iOS and macOS [35].

Swift was announced by Apple at their Worldwide Developers Conference (WWDC) in 2014. It has become one of the fastest growing programming languages in history [36]. Swift is held to high standards by the following three pillars:

- Safe
- Fast
- Expressive

Safety comes by avoiding undefined behaviors. Undefined behaviors occur when an execution in code in unexpected (like integer overflow) or attempting to access data that doesn't exist. For example, if an attribute of data contains a `null` value and is operated on, the result is an unexpected behavior. Mistakes like this can be caught in Swift with safe data types, like `Optionals`.

If these mistakes move past the production and shipment of software, it can cause major problems due to human error. `Optional` types in Swift ensure that these mistakes do not leave the compiler and are identified at or before compile-time. Criticism of this is that it makes Swift "feel strict," but Apple believes that in the long run, taking precautions like this ahead of compilation keeps safety a focus.

Swift competes with C programming languages like C, C++, and Objective-C with the speediness for performance tasks (hence the descriptive name for the language)! Apple took advantage of the rarity for new languages to be fast by making Swift have impressive performance. This can be achieved by making common computing tasks performant and by coupling it with a compiler with performance enhancements, like Xcode [37].

Many programmers that use Swift are united in their rejoice for Swift's clean syntax, which Apple describes as being a fruit of "decades of advancements in computer science" [38]. The Swift language is always improving and is open source. This allows developers to read the source code for the language and opens doors for community feedback. Developers from across the world have submitted new features and adjustments to the structure of the language. This allows the language to evolve to the needs and desires of the community as they change and will ensure a long lifespan of the language.

Swift, like all programming languages, support different Frameworks in code. A framework is “a hierarchical directory that encapsulates shared resources, such as a dynamic shared library, nib files, image files, localized strings, header files, and reference documentation in a single package” [39]. In essence, importing a framework add extra support in the form of more code, methods, objects, classes, and more. Frameworks can be thought of as extensions or plug-ins of extra code to allow a project to have a more specified focus.

3.11.1. Core Bluetooth

Core Bluetooth is a framework in Swift and Objective-C that provides extra functionality in code to allow apps to communicate with Bluetooth LE devices. In the scope of our project, the software app will be equipped to communicate with the Bluetooth LE receiver to transmit and receive information that the microcontroller will use to control the Daynight Panel.

Recalling the terms used to describe the client-server relationship in software, Bluetooth LE roles carry the same responsibilities but have different names. A “central” in the Bluetooth protocol is a device that has the same role as a “client,” and a “peripheral” in the Bluetooth protocol is a device that has the same role as a “server.”

In the Core Bluetooth framework, there are software classes that abstract the roles of centrals and peripherals. These classes can have properties assigned to them, as in the cases of centrals, or can have properties filled in automatically by scanning for nearby peripherals that are advertising.

These classes in the Core Bluetooth framework come with provided functions that supports different actions that fit in to the roles of centrals and peripherals.

3.11.2. CBCentral and CBCentralManager

The `CBCentral` (Core Bluetooth Central) class represents a central device in the Bluetooth LE protocol. The `CBCentralManager` class manages `CBPeripherals` and provides functions in a way to interface with them. CBCentralManager is a subclass of CBManager.

3.11.3. CBPeripheral and CBPeripheralManager

The `CBPeripheral` (Core Bluetooth Peripheral) class represents a central device in the Bluetooth LE protocol. The `CBPeripheralManager` class manages `CBCentrals` and provides functions in a way to interface with them. `CBPeripheralManager` is also a subclass of `CBManager`.

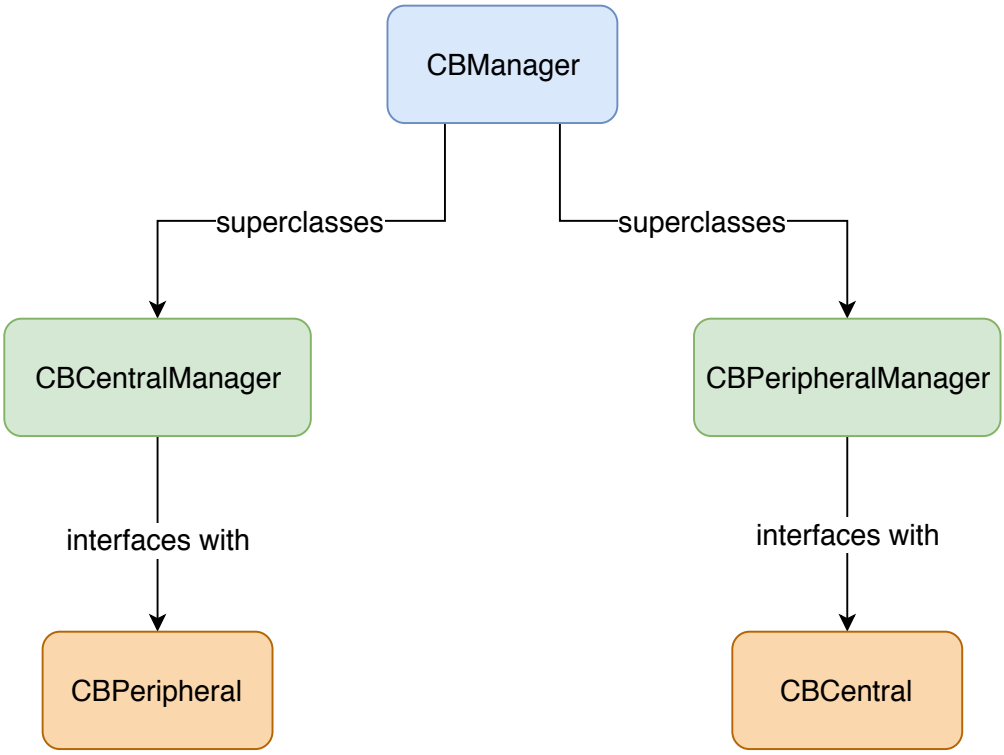


Figure 24. Class Hierarchy of CBManager

3.11.4. Scanning for Advertising Peripherals

When a device equipped with hardware that support Bluetooth LE communication is powered on, like our ATmega32u4 chip, it begins to advertise itself. Recall that the device will advertise its UUID in intervals of time called the advertising interval.

`CBCentralManager` is the class that interfaces with `CBPeripherals`. Once the ATmega32u4 chip's advertisement data are observed, it can be modeled in software as a `CBPeripheral` object. This is important as it allows the advertisement data to be parsed through. Doing so ensures that the device is the correct one we are looking for. Advertisement data from a CBPeripheral is stored in a dictionary in Swift.

A dictionary is a data structure that organizes data as key-value pairs. For example, the hardware device's name can be looked up in the dictionary by using the key `CBAdvertisementDataLocalNameKey`. In the first iteration of software, this is our main goal: to identify all nearby Bluetooth LE devices that are advertising, and to display their local names in a list. Accomplishing this will verify that the hardware is correctly configured to advertise its data over Bluetooth LE

and will ensure that the software properly scans and handles advertisement data being sent from the peripheral.

3.11.5. Transmitting Data Over Bluetooth

As the user interacts with the Daynight app, data will be transmitted over Bluetooth using GATT Characteristics as UART. In Swift, the value type `Data` must be used when writing to the TX Characteristic or reading from the RX Characteristic. Data is a value type that represents a byte buffer.

According to Apple's documentation, "the `Data` value type allows simple byte buffers to take on the behavior of Foundation objects. You can create empty or pre-populated buffers from a variety of sources and later add or remove bytes. You can filter and sort the content or compare against other buffers. You can manipulate subranges of bytes and iterate over some or all of them." [40]

Using byte buffers clearly offers a wide range of flexibility when using and manipulating raw data. This provides a comfortable foundation to begin development on. Once the GATT Characteristics are set up to be written to and read from in the Daynight app and on the microcontroller and Bluetooth LE module, the flow of data can be designed to align with the project's goals.

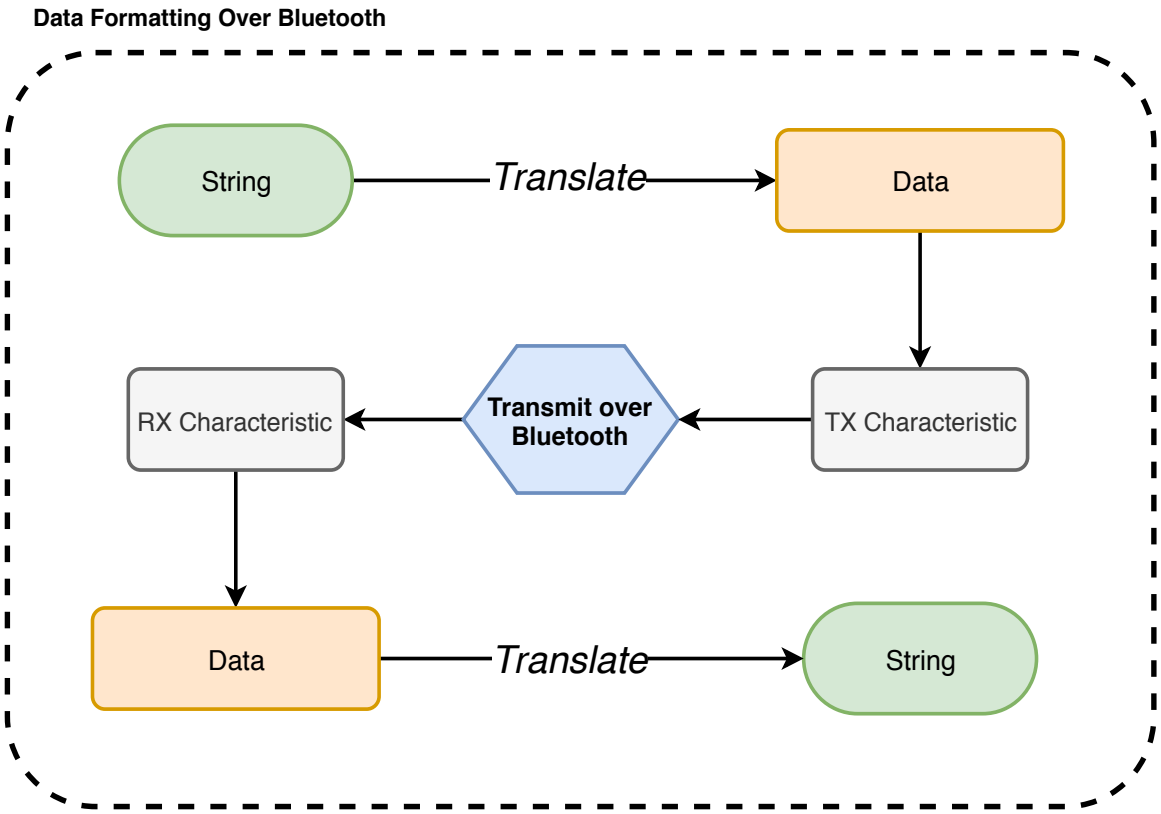


Figure 25. Data Formatting Over Bluetooth

The functionality of translating from `String` to `Data` and from `Data` to `String` is not specific or unique to either software application. The Daynight app will need to perform both of these operations, as will the software running on the microcontroller. For this reason, there are no marks of identification on the diagram that outlines these procedures. The only difference is the means by which this translation occurs.

3.12. Model-View-Controller Software Architecture Pattern

The Daynight app for iOS will follow the common software architectural pattern Model-View-Controller (MVC). This architectural pattern describes the organization and flow of data when dealing with user interfaces. Although this pattern is used in a majority of mobile software applications, it can be extended for use in web applications and desktop clients, too.

The *model* encompasses all data types and data structures used throughout the app. The key here is to have the *model* abstracted from the state of the UI. That is, the storage of data is not dependent on—and is unaware of—processes handling the UI. The *model* can be persisted and stored on user devices or in iCloud. This is incredibly useful for restoring data from an old device or switching to a newer device without having to skip a beat.

The *view* represents all output to the UI. Any information that is present including data, symbols, and other view elements like tables, tabs, or bars are included in the *view*. Data in the *model* may be presented in the *view*, but the *view* does not manipulate data in the *model*. These are dynamic and can be used to fill whatever data may be necessary (like `UILabel` for text, or `UIImage` for images). There also may be multiple of each kind being used at one time.

The *controller* bridges the abstraction between the *model* and the *view*. Here, any interactions to the UI that will result in modification of the data in the *model* will be handled. Similarly, if the data in the *model* is updated and needs to be displayed, the *controller* updates the *view*. This part of the app is typically regarded as the "least reusable" since it deals directly with discretely-defined *views.*

Following this architectural pattern allows for code reuse. Since the *model, view,* and *controller* are all separate, specific functions for manipulating one set of data can be used to manipulate others. Without being tied to any one specific *controller*, others can use the code as well.

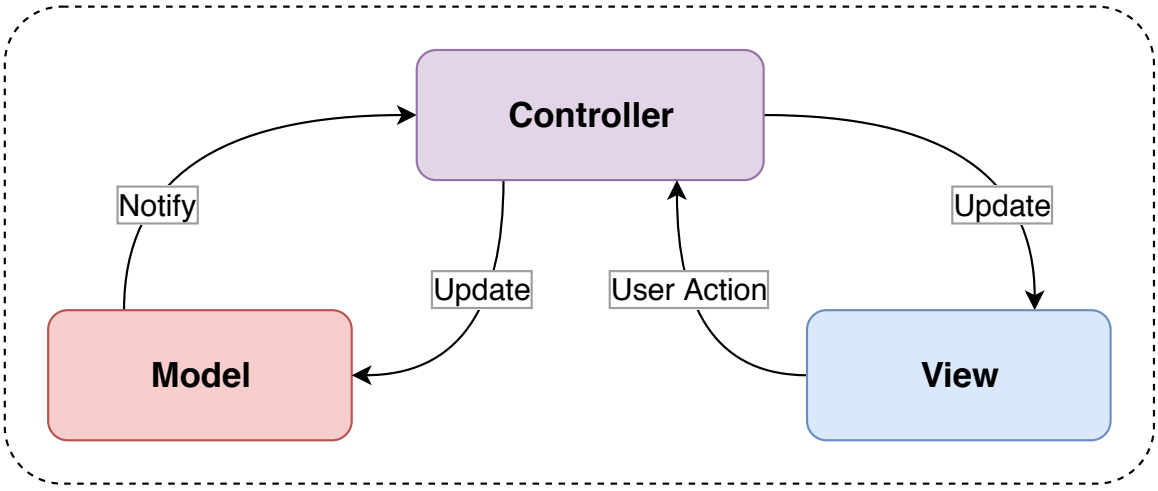


Figure 26. Model-View-Controller Workflow

Best practices in software follow this architecture pattern, and it has become a standard for writing good code. Breaking away from this pattern can result in code that exhibits the following unwanted behavior:

- hard to read
- hard to debug
- may not handle user action properly
- may access data unsafely
- may update the UI unsafely

All software developed for the Daynight app will follow these guidelines to provide the most refined software experience to developers and to users.

3.13. Lighting Configurations in Software

In order to simulate daylight, the Daynight Panel's LEDs will have set modes that they operate. For example, in order to create a color temperature, the warm LED will have to be balanced and color-mixed with the cool LED to create that effect. Even further than this, the RGB LEDs will have to be balanced and color-mixed to create an array of colors.

There are multiples ways to carry this operation can be carried out at the software level. There can be a memory-based configurations where different modes are stored into memory. Another way to perform this is calculation based where how the LED needs to behave is calculated based on an input.

3.13.1. Memory-Based Configuration

The lighting configuration for the Daynight Panel's LEDs will be used by the software any time the light color, temperature, or brightness needs to be changed.

Here, different approaches to modeling this process in software are considered, along with their advantages and disadvantages.

3.13.1.1. Complete Memory Storage

The first option is to store memory of every LED configuration. This is relatively memory intensive. A protocol would be needed for every step of all the LEDs. This mode would take over 250 kB of memory. This exceeds the 32 kB of memory that the Atmega328P. To account for the extra memory, an SD card may be used to offer extra availability, or the LED configurations may be stored in the Daynight app's software and be transmitted over Bluetooth LE.

If an SD card is utilized to account for the extra memory, the Daynight app would send a set of addresses to call on to the Atmega328P chip. Those addresses would address to the SD card and store the lighting configurations to be used onto the Atmega328P chip. The system would then cycle through the lighting configurations as needed.

If the LED configurations were stored in the Daynight app, addresses would be sent to the Atmega328P chip over Bluetooth LE. These addresses would then be stored in memory and cycled through in their prescribed path. This involves are more intensive transmission to be sent over Bluetooth LE and introduces concerns about transmission-time.

Complete memory storage gives the maximum amount of customization to the user. However, this leads to memory issues that could be difficult to resolve. Another issue is with such a high customization, the end user could be overwhelmed and not find the system desirable, or the developer may be overwhelmed trying to interface with the data.

3.13.1.2. Subset Memory Storage

Subset memory storage would work similarly to complete memory storage. However, only a select set of LED configurations would be chosen. This would be implemented by selecting certain brightness and color temperature levels to be stored. For example, as the LED driver brightness scales from 0-255, total brightness levels could be taken in sets of 5 for a given color temperature. For a color temperature of 3000k, brightness magnitudes could be taken in approximate steps of 5. The specific mixture of the cold and warm lights to create a 3000k color temperature that has a total magnitude of 5, 10, 15, and so on. This is what would be stored.

This would significantly reduce the amount of memory required to store the lighting configurations. The new approximate amount of memory required would be 10 kB. This fits well within the 32 KB limit that the Atmega328P has. Thus, the memory storage on the microcontroller can be utilized to maintain this data.

In this case, the Daynight app could send a set of values that would reference the memory in storage. The program would then go through a procedure cycle that runs through different values to give a feeling of daylight.

The advantage of this method is it gives the user a high level of customization while not being too overwhelming. It is also memory efficient and can fit completely on the Atmega328P chip.

3.13.1.3. Procedure-Based Storage

Procedure based storage would work by only storing set procedures onto the Atmega328P. This method would receive a procedure value from the Daynight app and then run that lighting procedure.

An example of this is the user might tell the system to run a dusk cycle. The system would then simulate dusk.

This method is extremely memory efficient, taking only the memory for a given procedure which could be measured in a just one or two kilobytes. The issue with this is that it gives the user almost no customization on how the Daynight Panel performs.

Using this method provides the best software architecture pattern and fits well with current standards to writing software. Abstracting data between the microcontroller software the user-interfacing software for the Daynight app is important to reduce workload between the two. Keeping LED instructions “close” to the LED allows for the fastest performance possible and allows the code to be easy to read and easy to reuse.

3.13.1.4. Summary of Memory based Storage

The different methods of memory-based lighting configuration have their advantages and disadvantages. The table below gives a summary of these options.

Table 24: Memory Based Lighting Configuration Comparison

	Complete Memory	Subset Memory	Procedure Based
Memory Required	High	Low-Medium	Low
Customization	Very High	High	High
Implementation	Difficult	Medium	Medium

Throughout the process of developing the microcontroller software and the Daynight app, we look forward to taking a deep look at the way these

configurations shape the user experience and the developer experience. Each configuration has effects on memory required, customization, and implementation.

3.13.2. Calculation Based Lighting Configuration

The way calculation-based memory would work is that the Daynight app would send a set of parameters to the Daynight Panel. These panels would be color and brightness parameters that the Atmega328P would then calculate on board. These calculations would then define the behavior of the LEDs. The board would have to allocate the memory needed for performing all the steps in a procedure. While these allocations would not be significant, care would have to be taken to free the memory after it is done being used. Failure to do so could lead to memory leaks and overflows which would crash the system.

The advantages to a calculation-based lighting configuration is that it is extremely memory efficient, only requiring memory for a given procedure. The issue it has though is making sure to prevent all memory leaks.

A minor issue is that this system would be initially slower, since the Daynight panel must make all the individual calculations for a procedure. However, this is a relatively minor issue since this only applies in the initial loading phase for the panel.

3.13.3. Memory Based vs. Calculation Based

Both a subset memory based and calculation-based approach to controlling the LED configuration are effective options. They are both memory efficient and give the user an effective but not overwhelming level of customization for utilizing the Daynight panel.

The best option between these two cannot effectively be decided yet. The subset memory-based approach uses the same formula that the calculation approach method uses. These calculations are just done beforehand, and results are stored in memory.

The advantages of a memory-based approach is that there won't be a loading phase where the Atmega328P chip has to calculate the LED values to use for a given parameter. Although it also will consume more memory, this memory is static and there isn't a worry about memory overflows.

The advantage of the calculation approach is that almost no memory is consumed. This however has the issue in that dynamic memory will have to be used for the program. Using dynamic memory runs the risk of memory overflows and leaks if not handled correctly. However, this is something that can be handled with effective coding.

A final decision for this cannot be decided until full prototyping will all the LEDs begins. Until then, both a subset memory approach and a calculation approach to configuring the LEDs is on the table.

4. Project Standards

Enforcing standards in a project ensure that they work together, similar to following protocols like USB. This also can ensure the health and safety of projects. This is an expected part of good engineering practice. At the user level, standards are employed in the design and are detailed in technical knowledge.

The Daynight Panel employs various standards that outline lighting, hardware and software communication protocols, hardware and software design, safety, power, and more.

4.1. Lights Standards

Lighting standards will be the standards that are specific to the system lighting. They relate to things like the LED footprints and government minimum lighting requirements.

4.1.1. Minimum Lighting Standard

This standard created by OSHA dictates the minimum lighting level for workplace environments. This standard requires that office areas have a minimum lighting of 30 foot-candles or 323 lux.

For indoor hallways and walk areas, the minimum lighting is 5 foot-candles or 53.82 lux.

4.1.2. LED Footprint

LED footprints are the space taken up by the LED. These are a standard so that it is easier to integrate different components that can be of similar size.

4.1.2.1. 2-Lead 3030 Footprint

This describes the sizing of the LED. This is what lets a user know what sizing to have the leads on a PCB. This is a 3.0 by 3.0 mm footprint, with two lead connections. The solder pad for this footprint is .48 by 2.2 mm box and a 1.33 mm x 2.1 mm box, with these two boxes having a .69 mm spacing.

4.1.2.2. 2-Lead 2835 Footprint

This describes the sizing of the LED. This is what lets a user know what sizing to have the leads on a PCB. This is a 2.8 by 3.5 mm footprint, with two lead connections. The recommended solder pad for this footprint is 1.09 by 2 mm box and a 1.74 mm x 2 mm box, with these two boxes having a .55 mm spacing.

4.2. Communication Standards

Communication standards serve an important role in ensuring standardized protocols for devices across various manufacturers. These communication protocols serve to allow various components and controllers to exchange data in an orderly way defined by a hierarchical organization. The Daynight Panel will make use of various communication standards to communicate from the user's iOS device to the Bluetooth module to the microcontroller to the several LED driver chips.

4.2.1. Serial Peripheral Interface (SPI)

This is communication protocol developed by Motorola. It works by sending information in a master to slave format. The format is a popular feature support by most electronics. It requires the master to provide a clock signal and select a slave to either read or write to.

4.2.2. Inter-Integrated Circuit (I2C)

This communication protocol requires a single bus composed of a clock and data line. All information is transmitted back and forth from master to slave via this bus. This standard method of communication is also popular and typically support by most microcontrollers.

4.2.3. UART

UART is a method of electronic device communication. It is a universal asynchronous receiver/transmitter. This is a popular method that was more frequently used in the past than it is now, but it is still very prevalent today. UART unlike the other communication methods is its own circuit which requires a cable connect the bit transmitting (TX) pin of one device to the bit receiving (RX) pin of the other device. The transmitting device will take the written code and turn it into serial data and this data is what is sent over the cable. This method is asynchronous so there are packets information is transferred in data packets with different configurations having different meanings. This mode of communication is the only one that can be used along with the Bluetooth module that is connecting the panel to the mobile application.

4.2.4. Bluetooth Low Energy (LE)

This communication protocol will transmit information between the iOS client (software) and the microcontroller (hardware) using the Generic Attribute (GATT) services. Bluetooth Low Energy is a modern way to communicate data in a way that is quick and light on battery usage.

While considering which technology standard to use for communicating among devices (between Wi-Fi and Bluetooth Classic, or Bluetooth LE), this standard in particular stood out as being the most appealing. Bluetooth LE's modern approach and design is immediately alluring to consumers and developers alike.

There are benefits to using modern technology like Bluetooth LE as a consumer. When aiming to reach a large number of users, it's easy to realize that customers don't want to purchase products that don't come with a lot of extra parts, remotes, buttons, or sensors that are required for daily use. Abstracting the idea of a remote control and fitting this functionality and design into one software application appeals to consumers. Having a place to go for all-in-one control allows for quick and organized access to everything a consumer might need.

Similarly, using a modern communication protocol like Bluetooth LE has its benefits for developers, too. With a modern technology comes modern frameworks and constant updates and support from the developer community. Swift, for example, is a new language that has a flood of support from the developer community, which makes it fun and easy to learn.

Software languages and well-documented frameworks in modern languages like Swift are easy to implement with the project. Having plug-and-play frameworks to work with provides a great clean-slate for building exactly what our project needs. Using a modern software language like Swift allows this application to run on iOS.

4.3. Power Standards

The Daynight Panel will rely on the electric power receptacles available in homes and offices. The power delivered to the receptacles is set by industry groups and electric utilities. In North America, the mains power is delivered to typical household receptacles at a voltage 120 volts of alternating current alternating at a rate of 60 Hz. These receptacles can transfer electric power to any device plugged into it up to 15 amperes.

4.3.1. National Electrical Manufacturers Association Connectors

These standard connectors were developed by a trade association of American electronics manufacturers to ensure standardized connectors and receptacles in the North American market. There are various NEMA connectors across various types of electronics industries. As a consumer electronics product intended for home or office use, the Daynight Panel will use the more common NEMA1-15 or NEMA5-15 plugs and receptacles.

4.4. Printed Circuit Board Standards

Printed circuit boards are designed manufactured according to standards set by industry groups. These standards ensure the longevity and safe housing of potentially dangerous electrical components found in common household and industrial electronics.

4.4.1. Institute for Printed Circuits Standards

The Institute for Printed Circuits is an industry trade association the make, use, and design printed circuit boards. [41] The group has derived standards for every aspect of printed circuit board design and manufacturing. These standards will apply to the Daynight Panel as this project will make use of a printed circuit board to neatly and efficiently house the Daynight Panels various integrated circuits and electrical components.

Examples of these standards define things such as wire harnesses, stencil design, and product documentation. The Institute for Printed Circuits applies these standards to printed circuit boards sorted into three product classes. The Daynight Panel would fall under the Class 1 – General Electronic Products class.

4.4.2. IPC-2221 Conductor Width Standard

To ensure safe transmission of electric current through a conductor on a printed circuit board the IPC-2221 conductor width standard was developed. This standard helps printed circuit board designers use a simple calculation based on the conductive material of their printed circuit board to determine an appropriate width for the conductive traces on the board. Conductor traces that are too narrow to supply to the current drawn through them may not be able to properly pass current or worse lead to spontaneous combustion of the printed circuit board.

4.5. Electromagnetic Interference Standards

All electronic devices containing switching devices or electromagnetic components are capable of producing electromagnetic interference. This interference can be transmitted between electronic devices directly through conduction or wirelessly through radio waves. The resulting interference emitted for an electronic device can adversely affect the performance of a “victim” electronic device. As a result of this performance interference, various government agencies and industry groups to create standards on electromagnetic interference. As a consumer electronic device for the home or office, the Daynight Panel must not interfere with performance of other electronic devices commonly found in a home, must be resistive to interference produce by other common household electronics, and not interfere with its own operations.

4.5.1. Federal Communications Commission Part 15

The Federal Communications Commission is the government agency responsible in the United States for regulating all communications both wired and wireless and any interference to those communications systems. The Federal Communication Commission developed the Part 15 regulations to impose standards on electromagnetic interference intentional or otherwise. Part 15 is comprised of two subparts. Subpart B covers unintentional emissions. Subpart C covers intentional, unlicensed, low power transmissions. The Daynight Panel will be emitting intentionally as a Bluetooth transmitter and emitting unintentionally from the electronic switching from the power supply and the pulse width modulation.

4.5.2. ANSI C63.4

ANSI C63.4 is a standard family produced by the American National Standards Institute for electromagnetic interference testing. This standard covers methods of measurement of radio-noise emissions from low-voltage electrical and electronic equipment in the range of 9 kHz to 50 GHz. The procedures defined in the ANSI C63.4 standards are commonly used to meet the regulatory requirements imposed by the Federal Communications Commission Part 15.

4.6. Environmental Safety Standards

To promote sustainable and efficient use of natural resources and environmental protection environmental safety standards have been developed. These standards seek to reduce the impact of the industrial processes and waste involved in the production and consumption of electronic products.

4.6.1. Restriction of Hazardous Substances (ROHS) Compliance.

This is a standard developed to restrict the use of hazardous substances in the fabrication of electronic equipment.

Everything used in the Daynight Panel would fit under category three, four and five of the RoHS Directive. Category three deals with computing and communications equipment, category four deals with consumer electronics and category five deals with lighting.

4.7. Design Standards

4.7.1. iOS Client Software

Design and development of iOS client software will be written in Swift, using Xcode. Xcode is a compiler that is distributed by Apple Inc. and includes design

software like Interface Builder. The software will guarantee user privacy by using safe frameworks like Core Location and Core Bluetooth, developed and distributed by Apple Inc.

The Interface Builder editor allows developers to design UI elements in Xcode without requiring any code. Designing with the storyboard controller combines the relationship and user flow between multiple View Controllers to create quick visual layouts. Some view controllers that are available in the storyboard include:

- Table View Controller
- Collection View Controller
- Navigation Controller
- Tab Bar Controller
- Page View Controller
- GLKit View Controller

Developers may also create their own storyboard controllers.

Auto Layout is a layout system provided in Xcode and Interface Builder that allows developers to set dynamic layout constraints for all UI elements. Adding these constraints between multiple UI elements and things like the safe area for a device allows the UI to keep a consistent visual appeal even for devices with differing display sizes. Since these layout constraints dynamically adjust the layout of UI elements placed in Interface Builder, developers only need to create constraints once to ensure their UI is consistent across different display sizes.

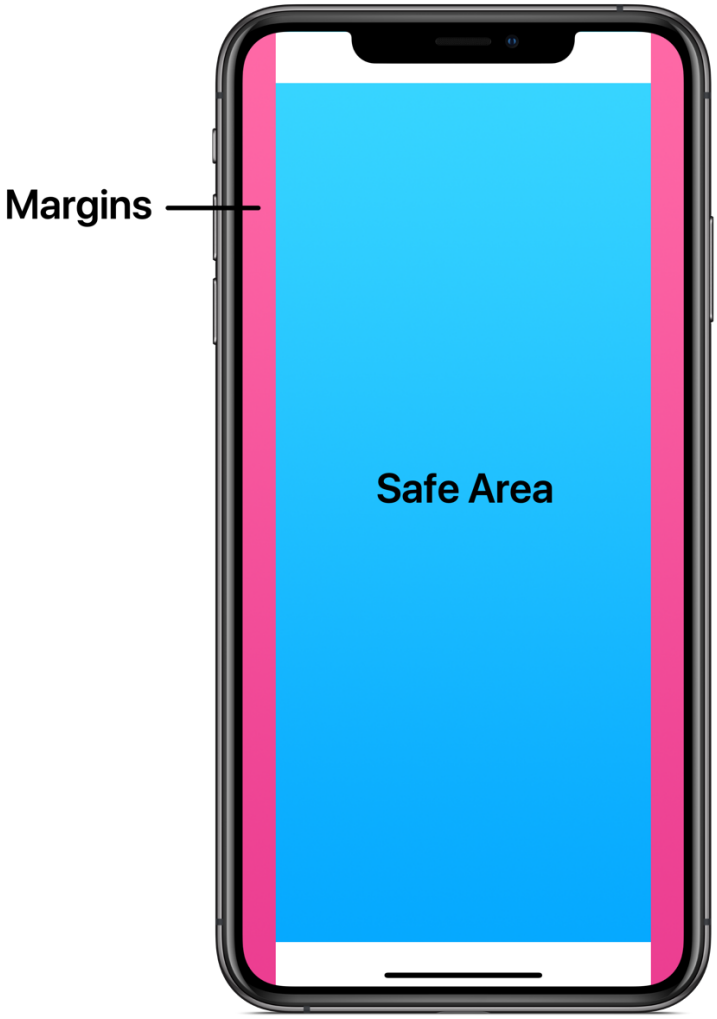


Figure 27. Safe Area Guides for iPhone X

Functionality for obtaining user location will be handled through Apple's Core Location Framework. This framework uses hardware components on-board the user's device, including Wi-Fi, GPS, Bluetooth, a magnetometer, barometer, and cellular antenna hardware in order to gather the following data:

- Geographic location
- Altitude
- Orientation
- Position (relative to a Bluetooth beacon)

This information is accessible within the software and can be used in order to provide the most accurate user data to the Daynight app. Being able to obtain geographical data regarding a user's device ensures that each user has a unique experience with their Daynight panel.

User privacy is ensured when using Core Location due to a request for authorization that prompts all users when the Daynight app launches. Allowing the

users to understand that the Daynight app will use device hardware is an important step toward empowering them with the way their data is used. As a result, users will always have the option to disable Core Location from gathering geographical location data. The Daynight app will have plenty of features, controls, and automation for users that take advantage of Core Location services and those that choose to not.

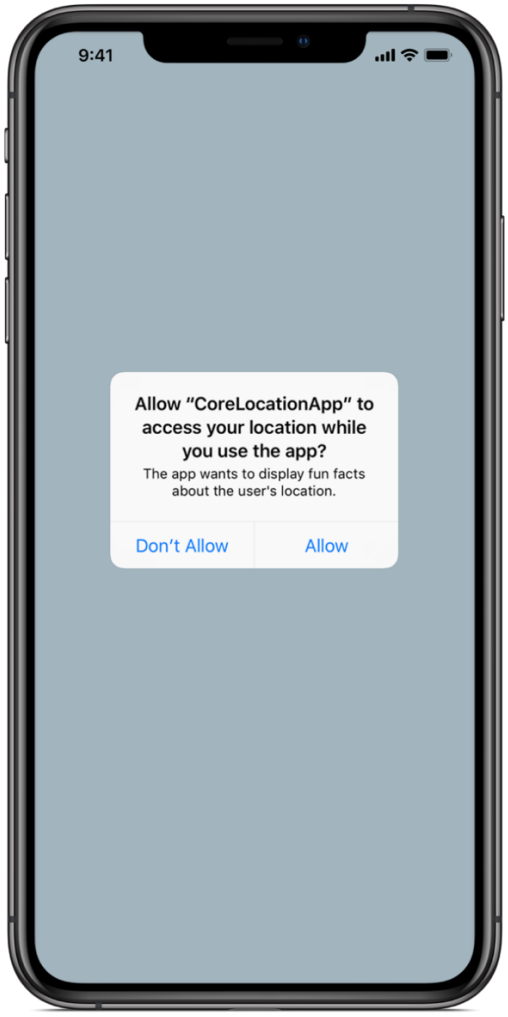


Figure 28. Core Location Authorization

4.8. Microcontroller Software

Design and development of microcontroller software will use Arduino Software Integrated Development Environment (IDE) to deliver software to the chip. Additional development and debugging may occur in the more fully featured Atmel Studio 7.

4.8.1. Standard Semiconductor Packaging and Footprints

Design of the printed circuit board must make use industry standard footprints based standard semiconductor industry packaging types such as QFN and DIP to interface components to the board.

The TLC59116 is available in two different packaging types [16]. The first is a 32 pin VQFN package that consumes an area of 25 mm^2 of board space per unit. The second is TSSOP package that consumes 42.68 mm^2 of board space per unit. Neither packaging type can be easily hand soldered and must be mounted using a heat gun and solder paste. The VQFN package will be selected for all TLC59116 units on the Daynight Panel for its efficient use of board space.

Several voltage regulators are being considered to provide power to various voltage rails needed to drive the various LED forward voltages and electronics. The linear regulators tend to use larger packing such as the TO-220 packaging in order to able to dissipate the large amount of heat produced. The switching

5. Project Design

5.1. Specifications and Features of Note for ATmega328P

These are the basic parameters which govern the discrete components directly connected to the microcontroller. The microcontroller's operating voltage determines the definition for a logic high and low on the I2C bus. Therefore, all the connected devices must comfortably operate in that voltage range. These devices need to be able to communicate with the master and handle clock stretching if necessary. Whether clock stretching will be necessary depends on the SCL clock frequency and the driver's frequency. The ATmega328P has internal pull up resistors but if the pull up is too strong or weak an external resistor might be necessary, and the equation below would be used for trouble shooting. The spike suppressing range provided by the input filter will help to decide whether external filtering will be necessary and define how much. This is a limited look at specifications further investigation is conducted with other specifications in their relevant sections [22].

Table 25. Direct Connect Component Specifications [22]

Parameter	Condition	Min	Max
Operating Voltage	-	$1.8V$	$5.5V$
Input Low Voltage	-	$-0.5V$	$1.65V$
Input High Voltage	-	$3.85V$	$6V$
Output Low Voltage	3mA sink current	0	$0.4V$
Rise Time for SDA and SCL	-	$20 + 0.1C_b\ ns$	$300ns$
Output Fall Time	10pF < C_b<400pF	$20 + 0.1C_b\ ns$	$250ns$
Spike Suppressed by Input Filter	-	0	$50ns$
SCL Clock Frequency	f_{SCL} $> max(16fSCL, 250kHz)$	0	$400kHz$
Clock at 1.8-5.5V	-	0	4MHz
Clock at 2.7-5.5V	-	0	10MHz
Clock at 4.5-5.5V	-	0	20MHz
R_P	$f_{SCL} < 100kHz$	1700Ω	$\frac{1000ns}{C_b}\Omega$
	$f_{SCL} < 100kHz$	1700Ω	$\frac{300ns}{C_b}\Omega$

The image below shows in general how devices would be wired to participate in I2C communication. The master defines the clock signal and controls what is being read from or written to the data line. The drivers would be receiving information from the data line to determine how to configure the LEDs. The pull up resistors set voltage for each bus so there is a constant defined reference for the microcontroller's logic.

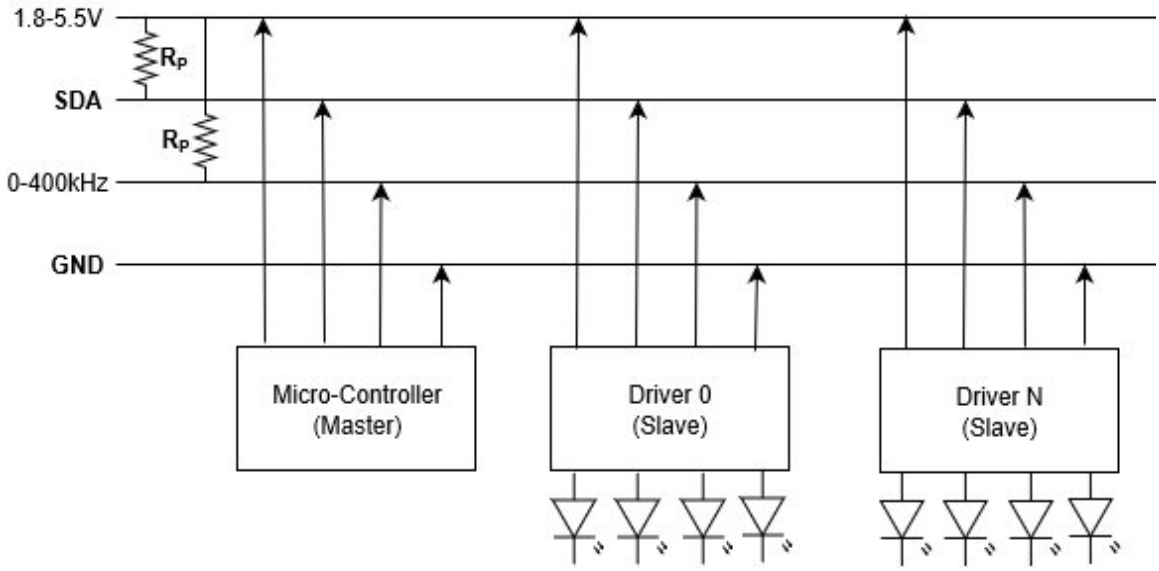


Figure 29. Parameter Defined Master Slave Configuration

The specifications which are defined in the previous table define which slaves are compatible to connect to the bus. They must be able to operate at the microcontroller's logic levels at the same rate as or slower than the microcontroller's clock.

Another important feature of note are the two wire interface registers. They configure and enable I2C communication. The first register that has to be configured is the TWBR (TWI Bit Rate Register) this register plays a part in defining the SCL clock frequency sent out by the controller. The TWBR Register along with the TWSR register determine the clock frequency through an equation defined in the controller's datasheet. After they are set then data can be written to the TWDR (TWI Data Register) it holds data that is about to be sent or that was just received.

The registers above are controlled by the TWI Control Register which monitors communication over the bus to ensure glitch free data transmission. It has the ability to enable interrupts, acknowledgements, start condition, stop conditions and error flags. It the bus that should be monitored to ensure correct data transmission.

These are the registers primarily responsible for I2C communication and have to be correctly set up to begin transmitting data.

5.2. TLC59116 Functional Diagram

To begin the driver address needs to be called by the master. The address is composed of seven bits. The last bit is used to define whether the byte is being read or written. The driver provides 4 pins for selecting a slave address. The last 3 pins are fixed by the chip designer.

There are several functions set around the address buses. The ALLCALLADR Register is capable of calling all registers connecting on the bus and the connected devices would simply send an acknowledgement back to the master. Another feature is built into the SUBADR1-3 which can refer to 3 different addresses. This allows different drivers to be programmed to perform completely different functions as needed. The final setting allows for a software reset. This register is call SWRST and it allows the microcontroller to reset any driver [23].

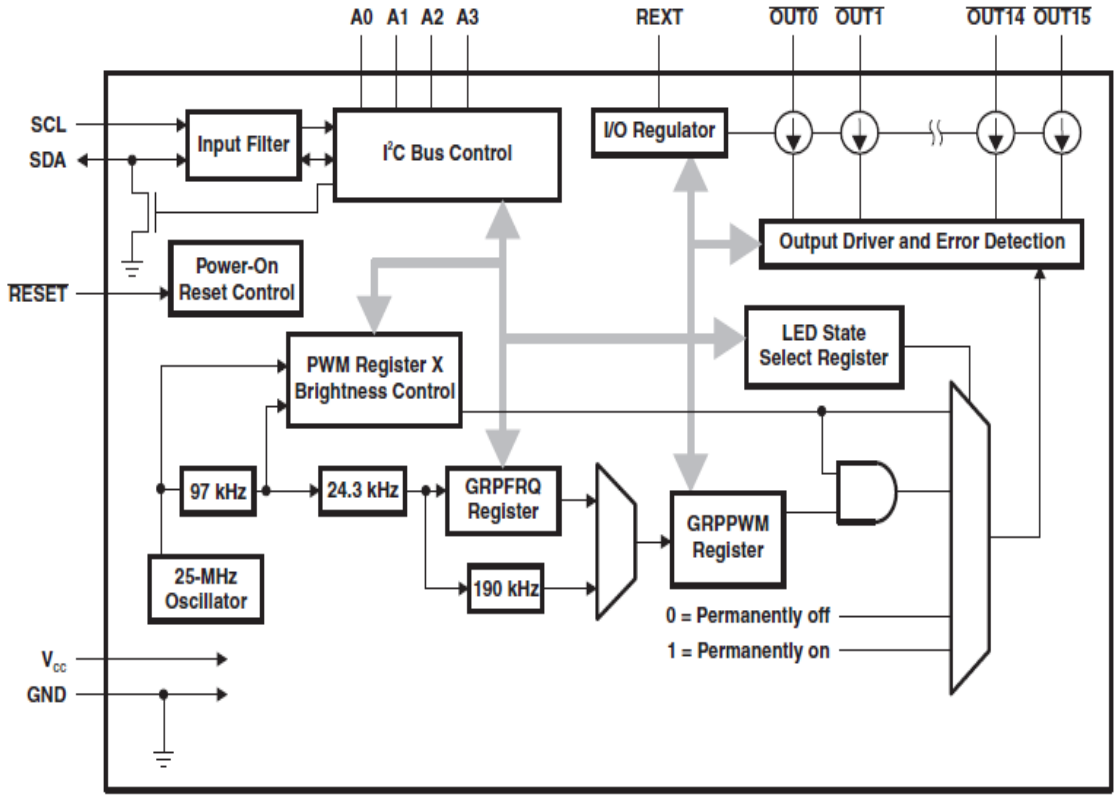


Figure 30. TLC59116 Functional Diagram [23] Permission Requested from Texas Instrument

5.2.1. TLC59116 Features – Brightness Control

The TLC59116 has two methods of brightness control. The first method is digital brightness via pulse width modulation. The second method is analog dimming via the constant current source.

The digital brightness control is combined with pulse width modulation registers. Each of these sixteen registers can be individually controlled. The driver provides the option to control each LED individually or as group. The group brightness can be adjusted along with the blinking state of all the LEDs. This level of control is attained by superimposing the appropriate signals.

In the functional diagram the output to the LED is connected to an OR gate. The OR gate has two inputs, one is the GRPPWM register and one is the combined PWM registers and brightness control block. The GRPPWM Register is the group duty cycle control register. How this register behaves depends on the DMLNK register which is only a single bit. When the value of this register is 0 the group dimming function is enables. When the value of this register is 1 the group blinking function is enabled. These registers make the possibilities for control endless and easier to program because the ability to blink or dim simply has to be enabled. The blinking is enabled by taking the 97kHz frequency that the pulse width modulated registers require to function and super imposing it with a 190-Hz frequency. The DMLNK value of one makes the global duty control register work with the group frequency register to control group blinking. The group frequency register is only considered when the DMLNK bit is 1, otherwise it becomes a don't care. For all of these functions the brightness and blinking can only be done in 256 steps [23].

The analog brightness control is set by varying the output of the internal constant current source. The external current is set by a resistor connected to the REXT pin and ground. To vary the current source, a potentiometer can be used to produce analog dimming. To control analog dimming in software, a digital potentiometer can be used and controlled by the microcontroller. This method of dimming is collective and does not allow for dimming of individual channels. This is one of the reasons why digital brightness control will be used.

5.2.2. TLC59116 Features – Current Source

The internal constant current source for each driver is configured through an external resistor connected between REXT pin and ground. An internal bandgap reference applies 1.25V across the resistor and the resulting reference current is multiplied by value set by the current multiplier. The constant multiplier bit sets the value of the current multiplier to a value of 5 or 15 for states 0 and 1, respectively. The product of the reference current and the current multiplier is resulting external current that will be driven through all 16 output pins of the TLC59116 [23]. A typical forward current for high lumen out LEDs is approximately 100 mA. To set the constant current source to 93.75 mA a resistor value of 200 ohms will be used. The analog dimming capabilities of the TLC59116 controlled by replacing REXT with a potentiometer will not be used to preserve the color integrity of the color LEDs, only digital dimming using pulse width modulation will be used.

5.2.3. TLC59116 Features – Error Monitoring

Another notable feature for this driver is the error flags for open circuit detection and overtemperature detection. These features are useful for trouble shooting and system monitoring. The open circuit detection helps to isolate any faulty LEDs or connections to the LEDs. The notification is announced as an error flag through Register EFLAG1 and EFLAG2.The chip monitors output current to determine

whether an open circuit has occurred. Once the criteria is met the Flag is set to zero [23].

The drivers and the LEDs are all going to be arranged on a printed circuit board and the panel is a device which will be running for several hours on end. The LEDs all together draw a lot of current so the chances of something overheating is not farfetched. While the system might be designed to operate in a certain range where the panel is placed could introduce more heat to the circuit and this is an issue that has to be monitored.

These error flags provide the chance for the customer or technician to be informed immediately about such issues through a few lines of code.

To monitor the temperature of the system the chip has a global overtemperature sensor along with sixteen temperature sensors for each LED port. When the chip temperature gets too high all the output ports get shut down and an error flag is set in all the output port registers. If a channel overheats then only the channel gets shut down and the error flag is placed in the register for that particular output port [23]. When overtemperature is detect the error, code is set to zero. Any output port with this flag was shut down for this reason.

5.2.4. TLC59116 Communication

The driver is simply a fancy shift register. It is a shift register with additional control features designed to cater to LED displays. It is not a smart it will simply respond to instructions sent over the I2C bus. These instructions have to take a predefined format which will be explained below.

Once both master and slave a correctly connect to the bus the master begins communication by send a start condition. This condition pulls the data line low and tells the slave devices that information is going to be transmitted over the bus. After the start condition is met the master will send a slave address followed by a zero or a one to tell that slave whether it should read or write the information. This set of bits is then followed by an acknowledgement from the slave telling the master that it received the information. This acknowledgment is followed by another byte to configure the control register of the slave. This register determines which internal feature register get written to and how to cycle through the registers in the driver. This pattern sets the tone for how all communication between master controller and slave driver will occur.

5.3. iOS and Bluetooth

Since the software for this project will be written for iOS devices in Swift, we will explore how the iOS operating system interacts with Bluetooth and Wi-Fi. Bluetooth connections are established through iOS Settings, under Bluetooth. Here, users may choose to connect to a Bluetooth-enabled device for uses like

streaming audio, like phone calls or music, and streaming video, like movies or TV shows. Since these kinds of connections require a constant exchange or stream of data, we can recall that this can be characterized as a Bluetooth classic connection.

That leaves Bluetooth Smart, or Bluetooth LE, to be considered. Bluetooth LE, like Bluetooth Classic allows iOS devices to connect and communicate with hardware accessories that support Bluetooth low energy wireless technology. Conversely, Bluetooth LE accessories cannot be accessed in iOS's general Bluetooth settings; specific applications must build in special functionality and support for searching for and managing connections. During a searching process, a list of available Bluetooth LE devices must be dynamically updated as the iOS device locates or releases available Bluetooth LE connections. As Bluetooth LE devices are being scanned, the universally unique identifier (UUID) is advertised to assist in correct identification [42].

Bluetooth LE UUIDs are represented in 32-bits or 64-bits, which is shorter than the standard 128-bit representation for a full UUID seen elsewhere. A UUID is assigned to each hardware device that supports Bluetooth LE. This is advertised periodically in what is known as the advertising interval, and similarly a scanning device (in our case, iOS) will scan periodically in what is known as the scanning interval.

It's worth defining the roles of different hardware components within the software model, to help make their functionality and responsibility clearer. A client is a device that sends or requests data operations and accepts a data response. A server is a device that receives or accepts data requests and returns data responses. This relationship is also known as a client-server relationship or model. To match up with the roles in this project, the iOS device will be the client, and the microcontroller used to control the Daynight Panel will be the server.

5.3.1. The Client-Server Model in Software

The client-server relationship or model is a common distributed application structure used in computer science. The designation of roles between the client and server help establish responsibilities and splits up the work. Typically, the relationship between a client and server exist separated by the internet, where they may physically be very far away from each other. Other types of network communications may be used in place of "the internet," like Bluetooth.

Client-server relationships are very common and seen everywhere in modern computing. Taking the University of Central Florida (UCF)'s online course system, Webcourses, as an example. Students log in Webcourses from their online web browser. This web browser acts as the client in this model. The client makes a request to the server to provide information. The server in this model would be the Webcourses database, which holds data like student profiles, classes, grades, etc.

Once the server returns information to the client, a student using the web browser may use it.

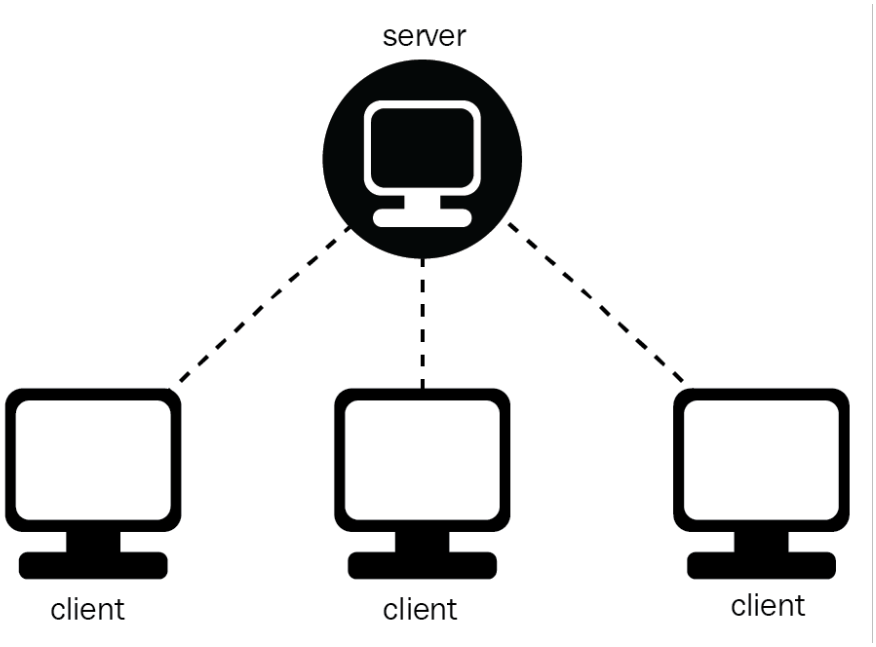


Figure 31. Client-Server Model with Multiple Clients [43]. Permission requested from O'Reilly, see in Appendix.

Since our devices are communicating over Bluetooth, they must be physically close to receive communication from each other. This is represented by a system with one server and multiple clients, but in the case of using Bluetooth LE, connections are exclusive. Communication is allowed to travel in both directions [44]. This means only one client may be connected to the server at a time. This still supports multiple users because switching clients does not require any extra work or notification, if a new client connects while there are no waiting services to be transmitted, the old client will be disconnected.

5.3.2. An Introduction to GATT Profiles, Services, and Characteristics

Once the client and server are in range for Bluetooth communication to take place, they may send or receive operations as requests or responses. These kinds of operations are called GATT (generic attribute) operations [44]. One example of a GATT operation that has already been mentioned above is the discovery operation.

This procedure is responsible for assisting both client and servers in setting up their connections. GATT transactions are organized in a hierarchical fashion by three attributes: Profiles, Services, and Characteristics, respectively.

Table 26. GATT Specification Profile

Profile Name	Specification	Version	Status
GSS	GATT Specification Supplement	1.0	Active

Profiles are compiled by the Bluetooth SIG and encompass a variety of Services. These come packages as a pre-defined collection provided to developers. The Bluetooth SIG supplies many Profiles for use, but only one will be used in this project, called the GATT Specification Supplement.

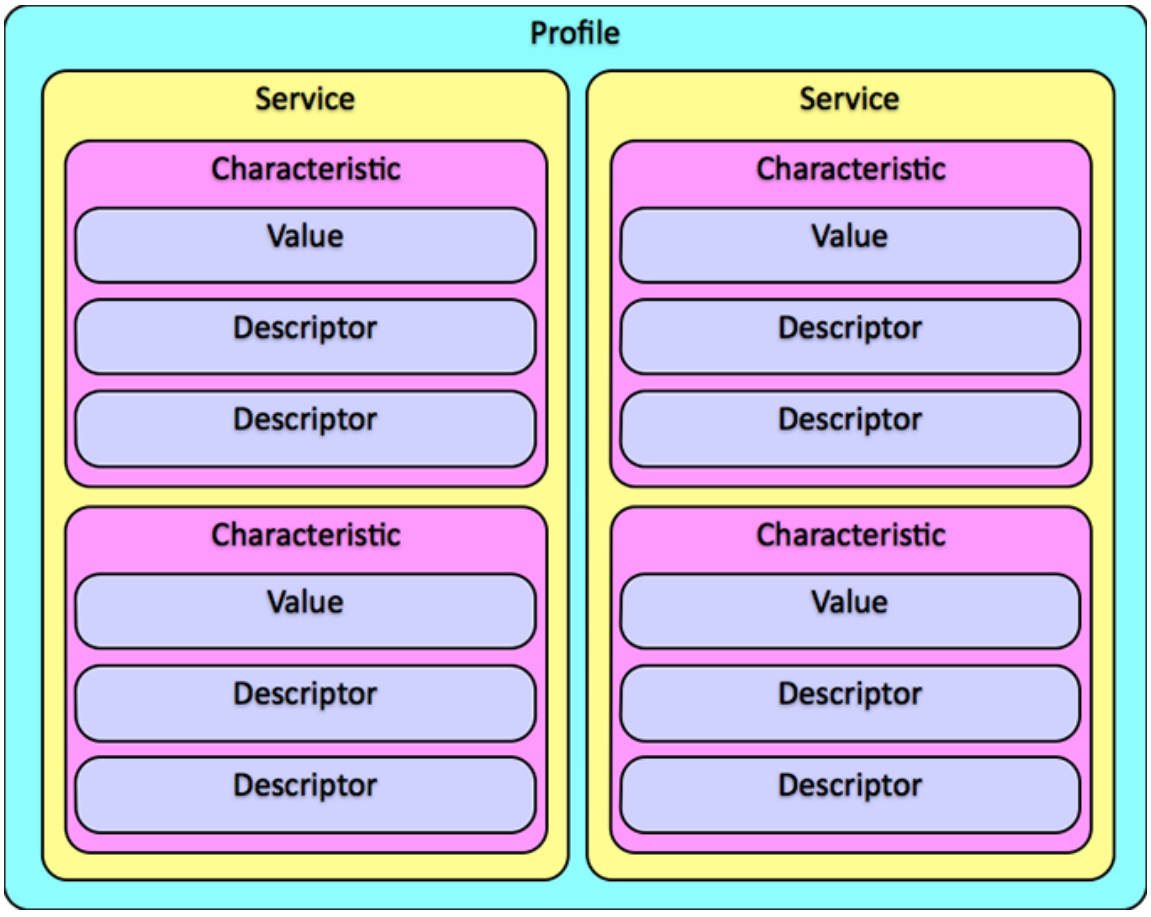


Figure 32. Visual Representation of GATT Profiles, Services, and Characteristics [45]. Permission requested from ProTech, see Appendix.

The GATT Specification Supplement is a Profile that is comprised of Services. Services are comprised of Characteristics and are used to organize data and can be used to define ways that data will be formatted, handled, and transported. For

example, dome Characteristics of a Service are read-only or write-only which means the handling and transferring of this data must be carefully executed. Characteristics will be looked at more later.

First, taking a look in to the generic list of GATT Characteristics shows a few that are immediately alluring and will be considered for this project, shown here.

Table 27. Characteristics of Interest in the Generic GATT Service [46]

Characteristic Name	Uniform Type Identifier	UUID	Specification
Current Time	org.bluetooth.characteristic.current_time	0x2A2B	General GATT
Date Time	org.bluetooth.characteristic.date_time	0x2A08	General GATT
Date UTC	org.bluetooth.characteristic.date_utc	0x2AED	General GATT
Day Date Time	org.bluetooth.characteristic.day_date_time	0x2A0A	General GATT
Day of Week	org.bluetooth.characteristic.day_of_week	0x2A09	General GATT
Exact Time 100	org.bluetooth.characteristic.exact_time_100	0x2A0B	General GATT
Exact Time 256	org.bluetooth.characteristic.exact_time_256	0x2A0C	General GATT
Rainfall	org.bluetooth.characteristic.rainfall	0x2A78	General GATT

Returning to the GATT Specification Supplement shown, there are Services inside of here that house Characteristics themselves. Services under the GATT Specification Supplement that are of interest are

Characteristics are the building blocks of Services for GATT operations, and are used to encapsulate a single data point (including arrays). During interactions with data between a Client and Server, Characteristics will be the means of direct communication. Some Characteristics are `readonly` or `writeonly`, meaning the way they allow Clients or Servers to access and manipulate their data is limited.

Table 28. Profiles of Interest in the GATT Specification Supplement [47]

Service Name	Uniform Type Identifier	UUID	Specification
Generic Access	org.bluetooth.service.generic_access	0x1800	GSS
Current Time Service	org.bluetooth.service.current_time	0x1805	GSS
Device Information	org.bluetooth.service.device_information	0x180A	GSS
Indoor Positioning	org.bluetooth.service.indoor_positioning	0x1821	GSS
Object Transfer Service	org.bluetooth.service.object_transfer	0x1825	GSS

All Services and Characteristics have their own UUID that is pre-defined and provided by the Bluetooth SIG. Each Service has an abstract that is used to describe its use-case or purpose to developers. For example, the Generic Access service (code name `generic_access`) “contains generic information about the device” [47]. All Characteristics for Generic Access are `readonly`.

5.4. ATmega328P and C

To efficiently communicate with the ATmega328P chip instruction will be written in C within the Atmel Studio 7 and uploaded to the chip via AVRdude version 6.3. The coding goes through the major components listed above and each piece of software must be configured for the chip.

The C language has been chosen because it allows finer control in the software environment by allowing the programmer to alter individual bits and registers directly. It is also easier to debug and can be used to talk to all the devices that will be connected to the I2C bus. There are several development environments that

can be used to develop the Daylight panels code and a popular resort is to use the Arduino integrated development environment which is designed to make the ATmega328P's internal features easily accessible. This environment is also C based but it is limiting, with no debugger. One of our design goals is to have fine control because fine control allows the simulated daylight to seem more realistic and makes the panel a versatile device.

The Atmel Studio 7 is an integrated development environment designed by Microchip technology for the Atmel chips. Despite being a separate more powerful integrated development environment, Atmel Studio 7 easily interfaces with Arduino sketches. This ability leaves all avenues open for future programming. Considering the number of ports, the degree of control and the intricate communication methods being used it is important to have an integrated development environment which will allow all aspects of a program and its output to be examined.

The microcontroller code that will be written in the Atmel Studio 7 integrated development environment will be built and compiled within the IDE. The Atmel Studio IDE will generate a hex file that will be uploaded via the avrdude software to the controller. For the information to be transferred both avrdude and Atmel studio have internal tools and functions that have to be configured so the data transferred is not corrupted.

6. Panel Component Integration

This section compiles all the research and specifications used to determine the optimal component for each system block and tests that the relevant specifications are as claimed. Once testing is completed the panel will be assembled and design begins.

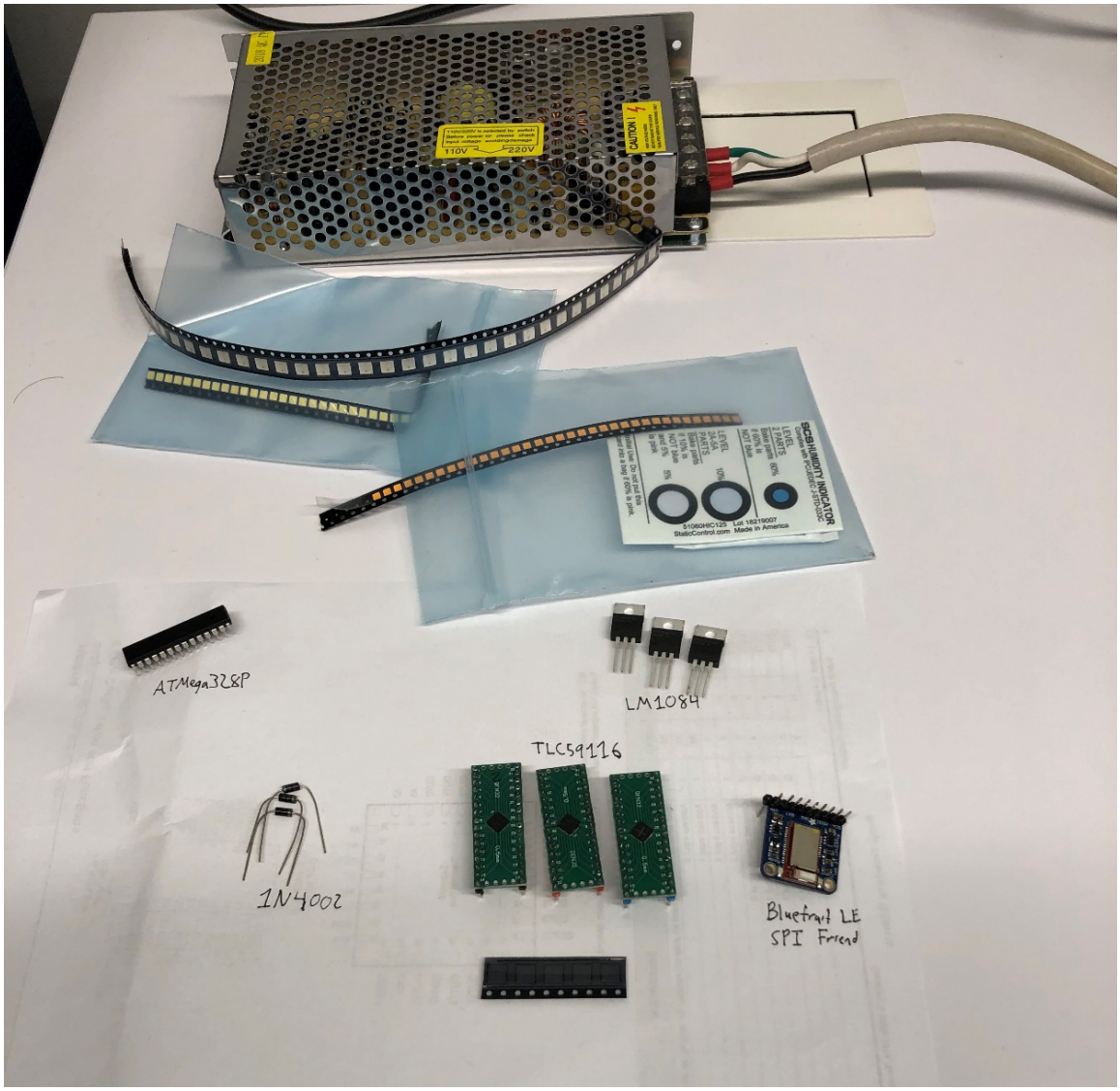


Figure 33. Collection of all major hardware components

The image above shows all the major components that make will make up the panel. In this configuration the components will be used for prototyping. Once proof of concept can be shown the schematic will be used for making a PCB from which further design, assembly and testing can be done.

6.1. Compatibility Testing

There are several devices whose experimental behavior needs to be tested and noted down before the entire circuit gets put together. This testing will simplify trouble shooting later. Testing also ensures that the product responds to the customer as we have designed it to. If we take it on blind faith that a feature in the chip or the driver will react behave as described in the data sheet, but it fails then the blame is on the designer.

The testing will also ensure that all the devices are arranged in a compatible manner. For example, the TLC59116 Driver has heat protection and this is a useful safety feature but, if the chip is placed too close to the shift register handling high currents in the right environment this could cause the all the LED outputs or a single LED output to trip. This behavior would become an annoyance to the customer and would not align with the marketing and engineering requirements.

6.1.1. LED Testing

To ensure that the LEDs are behaving as expected, each type of LED had its light output compared the expected value from the datasheet. The testing for the LEDs was done with the setup shown below.

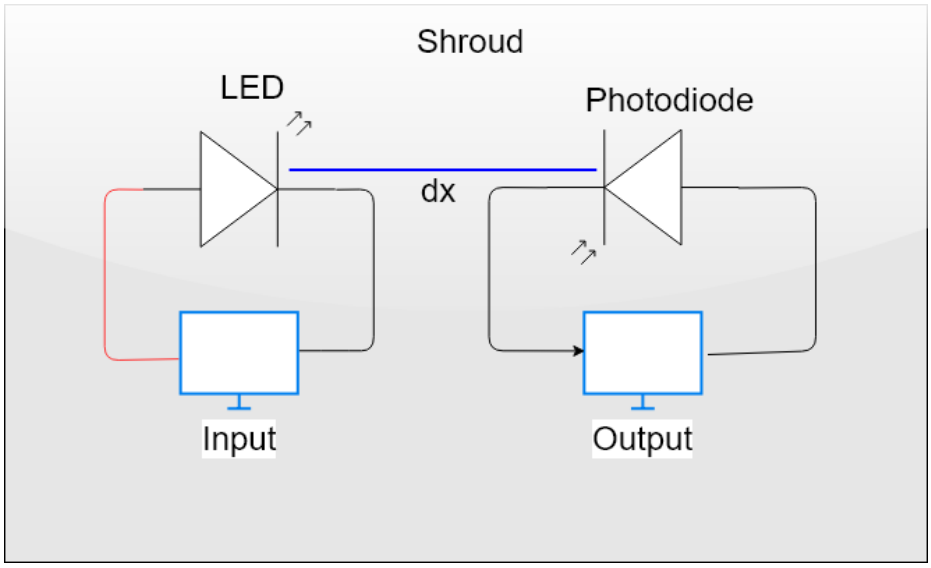


Figure 34: Model of LED Test

The system is primarily contained within a shroud to block out unwanted light. The LED on the left shines onto a known photodiode on the right. This photodiode is a part of a light sensor that gives the how much lux is on the sensor. The distance *dx* is then used with the area of the diode to find how many lumens the LED was producing. This is then compared to the expected value given by the datasheet.

Below is the actual shroud that was used in testing as well as the meter that detect how much light was created. Without the shroud, there was an ambient light of 50 lux. With the shroud, the number was negligible.

Figure 35: LED Shroud Set-up

There were some difficulties in wiring the LEDs. Due to the LEDs being surface mount parts, and extremely small ones at that, soldering the leads proved to be difficult.

There was an attempt to have two of each LED tested, but due to continuous damage to the LEDs in the soldering process, this was halted, and testing had to be done with what didn't break.

Since every LED on the RGB package has different forward voltages and lumen outputs, they will be treated as their own LED for this part of the report. This also fulfills the purpose for later programming on designing color balancing.

The graphs are given in relative lux vs forward current. Relative lux is determined by comparing the ratio of the 100 mA typical lux against the lux at a given forward current. This value will be 100% at the 100 mA forward current.

6.1.2. Warm (2700K) LED

The behavior of the 2700k warm LED is shown below. The behavior slowly tapers off as the forward current rises. The behavior of the graph is extremely like that given by the data sheet. The difference can be explained by the rough laboratory conditions when testing the LED.

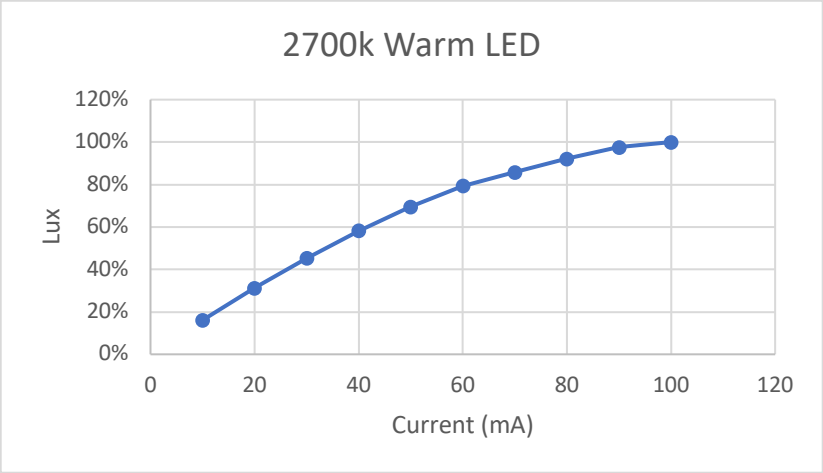


Figure 36: 2700k LED Lux vs. Current

6.1.3. Cool (6500K) LED

The behavior of the 6500k cool LED is shown below. The behavior slowly tapers off as the forward current rises. The behavior of the graph is extremely like that given by the data sheet. The difference can be explained by the rough laboratory conditions when testing the LED.

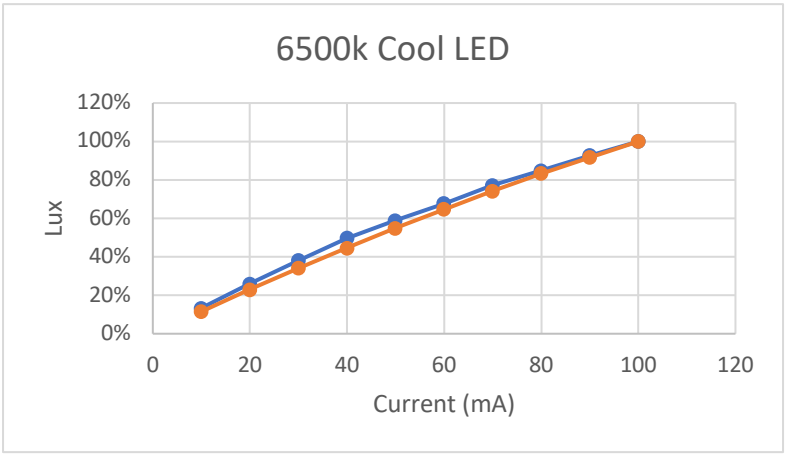


Figure 37: 6700k LED Lux vs. Current

The cool LED is the only light that has two test points due to multiple LEDs being damaged when attempting to solder them. The LEDs were fragile and as a result of not using them as surface mount parts, they would break apart.

6.1.4. Blue LED

The behavior of the blue LED is shown below. The behavior slowly tapers off as the forward current rises. The behavior of the graph is extremely like that given by the data sheet. The difference can be explained by the rough laboratory conditions when testing the LED.

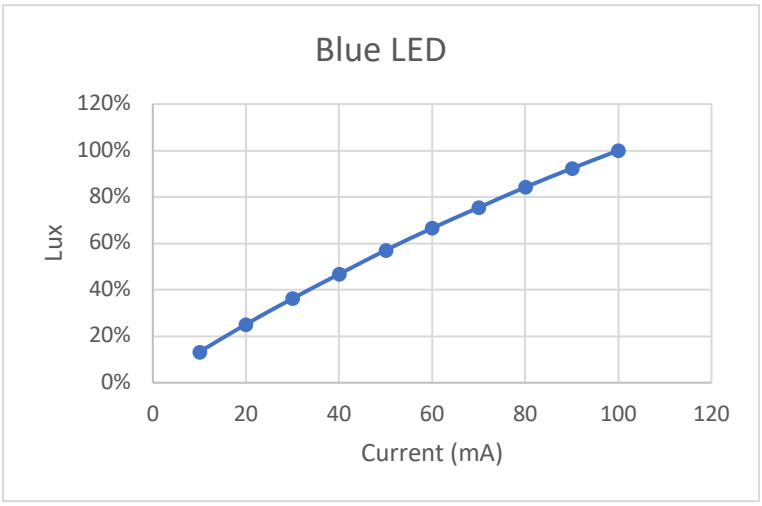


Figure 38: Blue LED Lux vs. Current

The graphs are given in relative lux vs forward current. Relative lux is determined by comparing the ratio of the 100 mA typical lux against the lux at a given forward current. This value will be 100% at the 100 mA forward current.

6.1.5. Red LED

The behavior of the red LED is shown below. The behavior slowly tapers off as the forward current rises. The behavior of the graph is extremely like that given by the data sheet. The difference can be explained by the rough laboratory conditions when testing the LED.

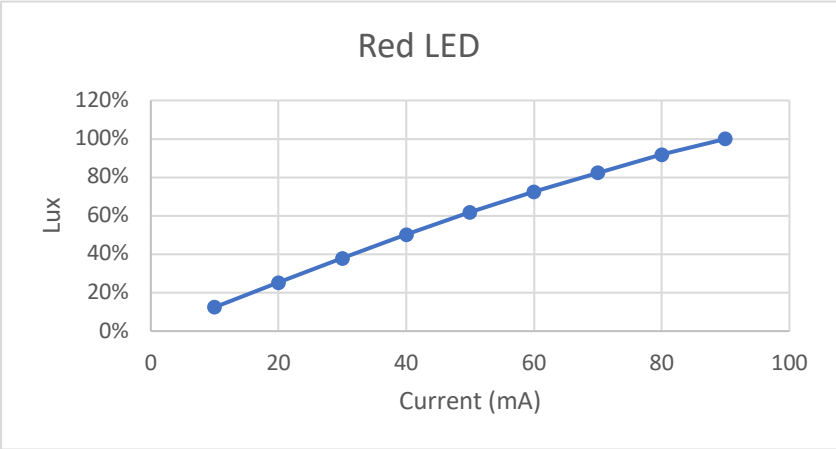


Figure 39: Red LED Lux vs. Current

The red LED only goes to 90 mA due to damage that the red LED incurred during testing. Since multiple LEDs were damaged in the process of wiring the surface mount parts, no attempts were made to set up another LED for testing.

6.1.6. Green LED

The behavior of the green LED is shown below. The behavior slowly tapers off as the forward current rises. The behavior of the graph is extremely like that given by the data sheet. The difference can be explained by the rough laboratory conditions when testing the LED.

The green LED tapered off slightly more compared to the other RGB lights. This property is reflecting what is described by the data sheet, which also shows that the green LED tapered off slightly more.

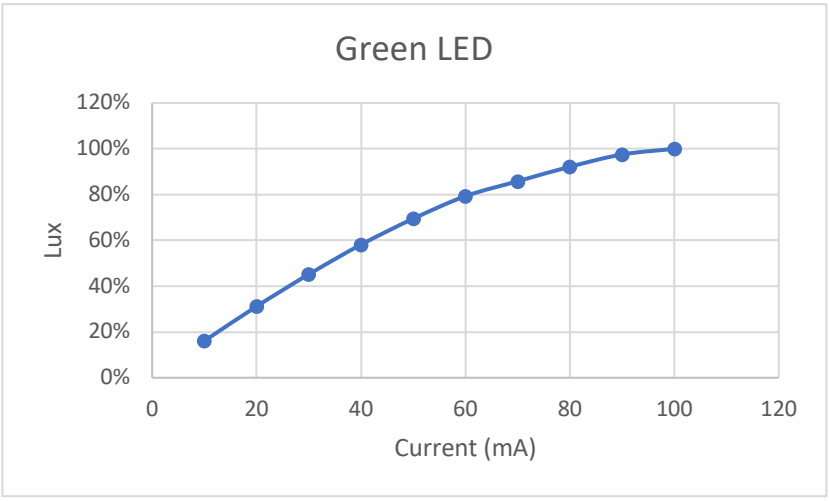


Figure 40: Green LED Lux vs. Current

6.1.7. LED Driver Functionality Test

In order to test the functionality of the TLC59116 the following tests are performed on a breadboard.

The TLC59116 is tested using a microcontroller, 16 LEDs, and a solderless breadboard. A test program is written to drive each LED individually at full brightness. The next test in the program would cycle each channel individually through the 256 brightness steps from output channel 0 to output channel 15. The next test would drive all 16 channels simultaneously at maximum brightness. The final test would drive all 16 LEDs simultaneously through the 256 brightness steps. The user should be able to notice the intensity of the LED brightness changing, but not notice the LED pulsing on and off rapidly due to persistence of vision.

Table 29. TLC59116 Test Table

Test Performed	Description	Purpose
Individual Output Channel Test	Each output channel is used to drive an LED	This ensures the constant current source for each output channel is functional
Individual PWM Test	Each output channel is cycled through the 256 brightness steps	This ensures the pulse width modulator for each output channel is functional
Collective Output Test	All output channels drive all the LEDs simultaneously	This ensures the device is capable of handling maximum load conditions
Collective PWM Test	All output channels are cycled through the 256 brightness steps simultaneously	This ensures the all the integrated pulse width modulators can operate simultaneously without interference

In order to test the drivers, they had to be soldered to a QFN to DIP adapter as is shown in the following image.

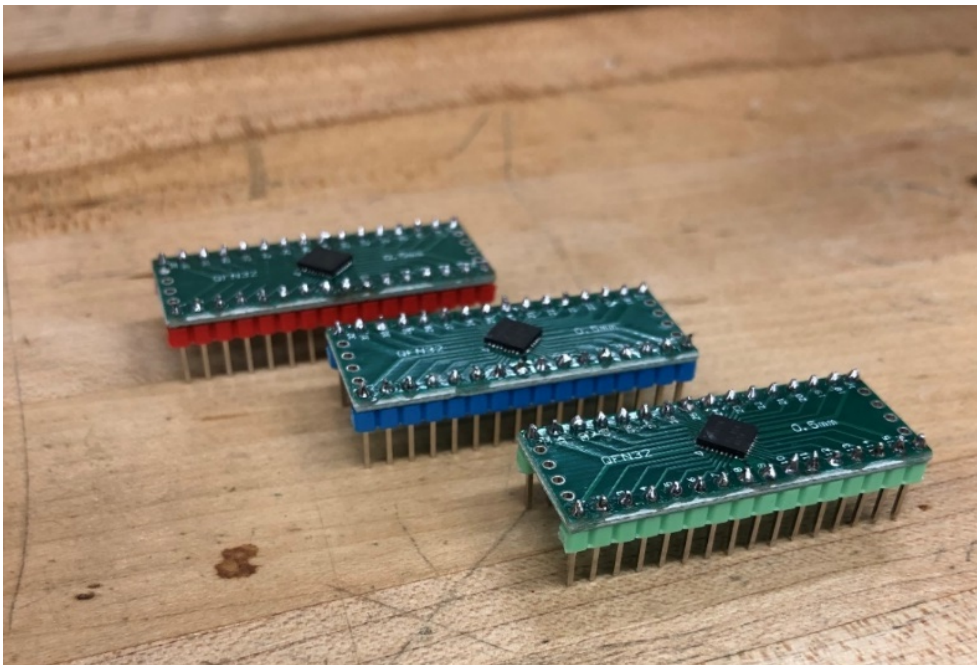


Figure 41. TLC59116s mounted on QFN-DIP adapters for testing

The header pins connected to the adapter will be used for the controller to communicate with the drivers and vice-versa. The testing setup can be observed in the figure below.

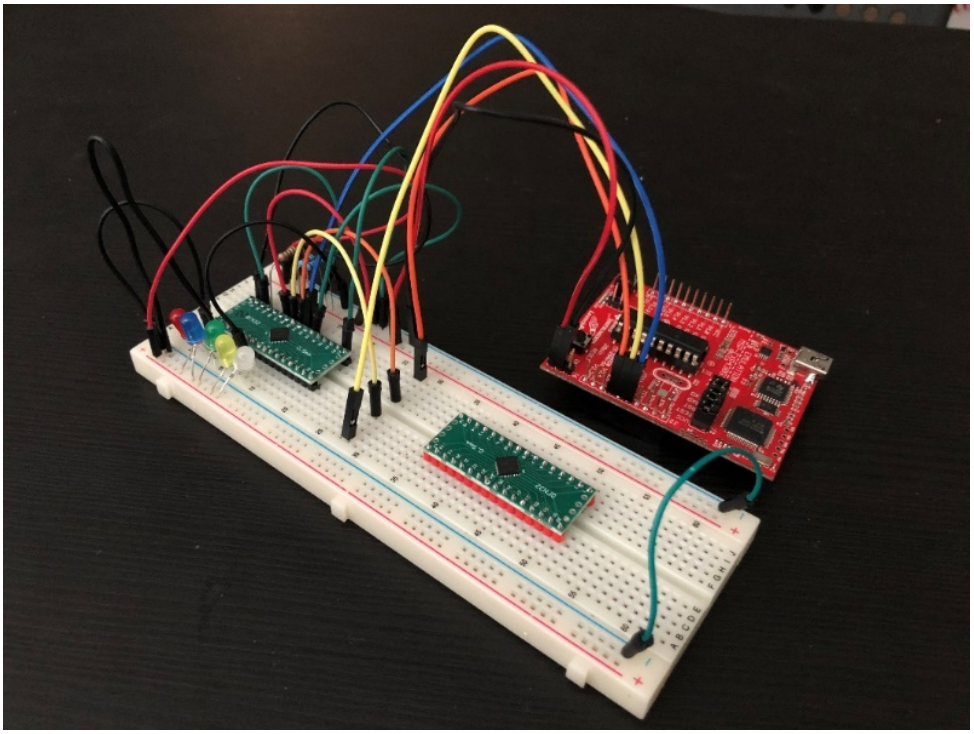


Figure 42. TLC59116 LED Driver Compatibility Testing Setup

Following the basic functionality tests described above for each of the testing units, the tests will be repeated with multiple TLC59116 drivers sharing a single I2C bus. This experiment will ensure that the bus is set up properly and multiple slave TLC59116 LED drivers can be called upon nearly simultaneously by the master ATMega328P microcontroller. This is accomplished by hardwiring the address pins A3-A0 to either VCC or ground. The result is that each TLC59116 will have a unique address on the I2C bus.

6.1.8. Power Supply Functionality Test

In order to test the functionality of the Aiposen S-120-12 power supply unit the following tests are performed in the laboratory.

The Aiposen S-120-12 is a 120 W power supply unit capable of delivering 12 V at a maximum load current of 10 A. The first test that will be performed will test the voltage regulation capability. This is done by simply connecting the power supply unit to the wall and using a digital multimeter to measure the output voltage on both output channels with no load connected. The expected result is a steady 12 V output. Small discrepancies can be adjusted by tuning the output to 12 V by turning the orange voltage adjustment potentiometer. Large discrepancies are indicative of power supply failure and require unit replacement.

The second test will test the voltage regulation at full load conditions. The full load of the Aiposen S-120-12 is at 10 A. An electronic load will be used to simulate the full load conditions.

The third test will check the voltage ripple on the regulated output. There is no specification for voltage ripple provided by the Aiposen S-120-12, but ideally this voltage should be as small as possible. The voltage ripple value will be measured in V_{RMS}, V_{pp}, ripple percentage at a full load of 10 A. This full load condition will be simulated by using an electronic load set to 10 A. A digital oscilloscope will be used in parallel with electronic load to provide a measurement of the voltage ripple in V_{RMS} and V_{pp}. A test circuit diagram can be seen in the figure below.

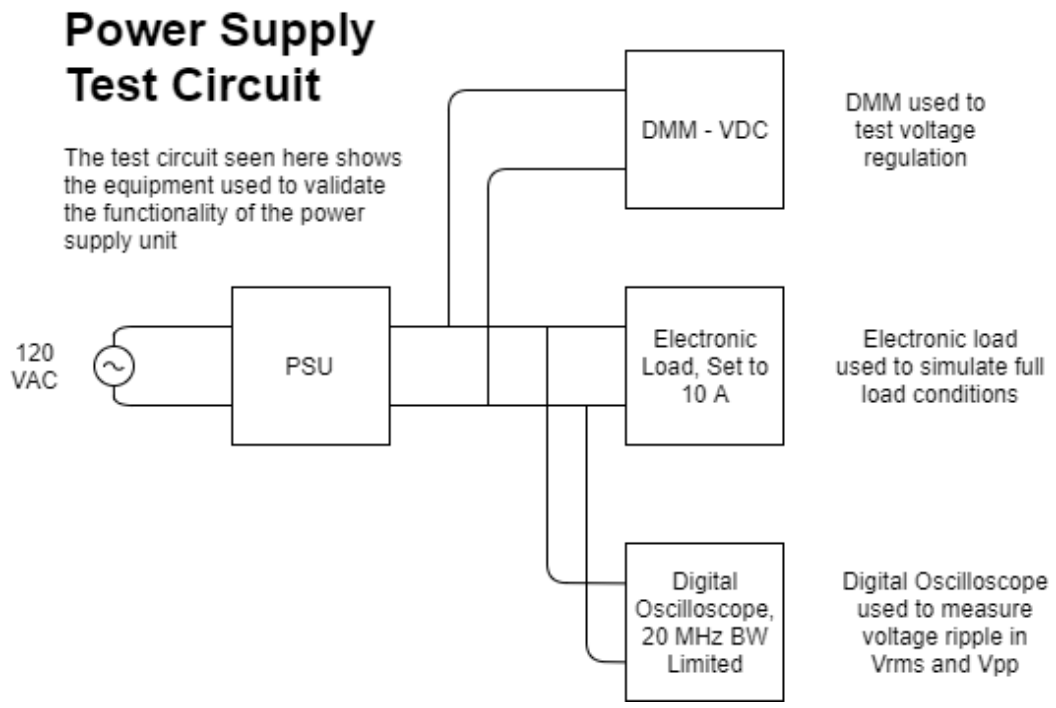


Figure 43. Power Supply Test Circuit

After performing these tests, the results shown in the table below were determined, and the Aiposen S-120-12 power supply unit was determined to be compatible with our project design.

Table 30. Power Supply Unit Testing Results

Test Condition	Voltage Regulation [VDC]	Voltage Ripple [mV_{pp}]	Voltage Ripple [mV_{RMS}]
No Load [0 A]	12.1	14.7	4.22
Full Load [10 A]	11.8	14.9	4.37
Typical Load [8 A]	11.9	16.8	3.59

The table above shows the testing procedure and data for the Aiposen S-120-12 power supply unit. This power supply unit was designed to deliver a constant 12 volt supply at up to ten amperes. The full load is designed with a factor of safety and is rounded up from the typical load value. The typical load value is computed from the total current draw of operating all of the LEDs and electronics simultaneously.

6.1.9. Voltage Regulator Functionality Test

Similar to the power supply unit, the voltage regulators used to produce the power rails necessary to drive the LEDs and the electronics must be validated to ensure their compatibility in our design. To validate the compatibility three test conditions will be considered. First, the regulator will be put under the “no load” or zero current draw. Second, the regulator will be put under “full load” current draw. Third, the regulator will be put under a “typical load.” The current values for the full and typical load will be vary with the designed load for the power rail each regulator will regulate. At each of these test conditions, for each of the regulators, three parameters will be measured. The parameters measured will be voltage regulation measured in volts and the voltage ripple measured in volts root mean squared and volts peak-to-peak.

In order to simulate the load conditions on the regulator an electronic load will be used. The electronic load can be used to produce a stable current draw from the voltage regulator. Once the proper load is applied, a digital multimeter can be used to measure the voltage regulation in volts. Additionally, once the proper load is applied to the regulator, a digital oscilloscope can be used in the AC coupled configuration to measure the ripple voltage in volts root mean squared and in volts peak-to-peak.

For example, the warm white and cool LED power rail of the switching regulator design was designed to deliver a constant 12 volt supply at up to six amperes. The full load is designed with a factor of safety and is rounded up from the typical load value. The typical load value is computed from the total current draw of operating all 24 white LEDs at once.

6.2. Project Prototyping

6.2.1. LED Color Balancing

Daynight uses multiple LEDs to achieve different color mixes. To ensure that the colors mix correctly. Expected combinations were calculated mathematically beforehand to be tested in lab conditions. This will expediate the testing process so that base values will be on hand when testing color mixing.

6.2.1.1. Cold and Warm LED Color Balancing

To calculate a wide range of values, a python program was created that would store the values in a format that would be easily picked up by Excel. These values can then be easily referenced.

The program has the equation for balancing the LEDs, the target temperature, the target brightness, and the basic LED data. The equation that calculates the temperature mixing takes the color temperature of the two LEDs, and then

multiplies them by a magnitude value, with each LED having a different one. This summation of multiples is then divided by the sum of the two magnitude values. These magnitude values are cycled through using a double loop, one loop controlling the magnitude of each LED.

The program requires that the sum of the two magnitudes lie within a certain margin of the target magnitude. The loops cycles through the values and storing the best set of values based off the color temperature in a set of variables. When the program cycles through the loops, it returns the best values to be written to a file.

After it finds the best mix for a given temperature and magnitude, it increments the magnitude by a set amount. It then repeats the previous process for all the stepped magnitudes for a given temperature. It will then increment the temperature by a given amount and repeat all previous processes from here. It does this until it cycles through all the color temperatures in the range of 2700k, the lowest possible temperature, to 6500k which is the highest possible temperature.

A set of examples below shows how the LEDs need to be balanced for a given color temperature and brightness magnitude. The table shows the expected temperature, how the LEDs are balanced, and the target and expected brightness magnitudes. Magnitudes are on a scale of 0-255. These magnitudes represent what value is sent to the LED driver. Two example color temperatures were selected from the list generated by the program.

The first example shows the expected cold and warm LED balancing for a target temperature of 3000k. This color temperature would be representative of an early morning or dusk daylight condition.

Table 31: 3000k Target LED Balancing

Result Temp	Cold (6500k)	Warm (2700k)	Target Magnitude	Final Magnitude
3080	1	9	10	10
3000	3	35	40	38
2983	5	62	70	67
3004	8	92	100	100
2999	10	117	130	127
3008	13	147	160	160
3000	15	175	190	190
2997	17	200	220	217
3004	20	230	250	250

The second example shows the expected cold and warm LED balancing for a target temperature of 6000k.This color temperature would be representative of a midafternoon daylight condition.

Table 32: 6000k Target LED Balancing

Result Temp	Cold (6500k)	Warm (2700k)	Target Magnitude	Final Magnitude
5740	4	1	5	5
6025	28	4	35	32
6009	54	8	65	62
6004	80	12	95	92
6001	106	16	125	122
6000	132	20	155	152
6001	159	24	185	183
6000	185	28	215	213
5999	211	32	245	243

Due to the stepper nature of the LED drivers, some imperfections were seen in the tables. These differences however stayed within a 5% error range and the worst cases were at extremely low brightness magnitudes, where balancing would be more difficult regardless.

6.2.2. Prototype Daynight App for iOS

The Daynight app for iOS will provide all of the controls that users would need in order to interact with their Daynight panel. This app will be available for all Daynight panel consumers and is required for initial setup. With current Daynight software prototype, users are able to identify nearby Bluetooth LE compatible devices that are advertising themselves. Advertisement data like the hardware device's local name is presented in a list.

A prototype version of the Daynight app will be developed initially in accordance to the following objectives:

- Become familiar with the Core Bluetooth framework

    - Allow the software to display all nearby Bluetooth LE capable hardware, called “peripherals”
    - Allow the software to connect to any given peripheral and display all GATT Services and GATT Characteristics
    - Pinpoint peripherals that support UART GATT Services and GATT Characteristics
- Become familiar with the Core Location framework
    - Provide location-specific weather data
    - Provide location-specific sunrise and sunset times
        - Achieve a realistic simulation of a day cycle by mimicking sunlight intensity (brightness) and color
- Understand connectivity between iOS and Bluetooth
    - Lay down the groundwork for the final version of the app that communicates solely over Bluetooth LE
- Understand connectivity between Bluetooth and the MCU
    - Lay down the groundwork for the final version of the Daynight Panel hardware configuration for communicating with LED drivers
- Have a fully-designed application that can perform prototype testing
    - Perform prototype testing of hardware and software in accordance with Senior Design class requirements

Prototype testing will ensure that all hardware components are working in unison and behave according to controls send by the software. This will be an important step in moving toward the final project design.

Figure 44. iOS App icon for Daynight

The Daynight app icon symbolizes the sun and moon together in a shared and unified experience, surrounded by a border that represents the physical edges of the Daynight panel. The rounded corners of the border share a relationship with the radius with the rounded edges of the iOS app icon shape to appear soft and relaxed. This shares graphic motifs with the system icon that represents Display and Brightness preferences and Do Not Disturb. By providing familiar graphics, users will be able to immediately understand and interpret the symbols in the app icon to provide some context.

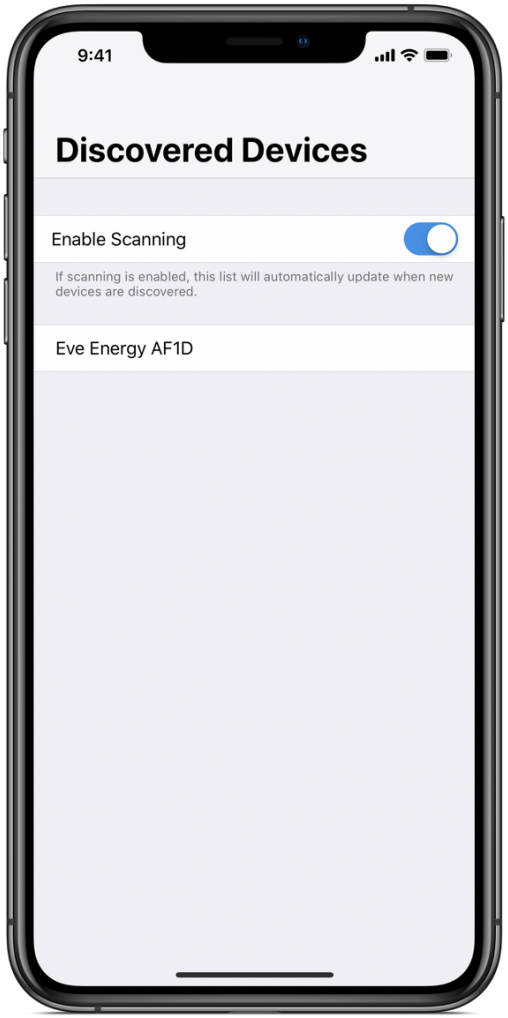


Figure 46. Scanning when Adafruit is offline

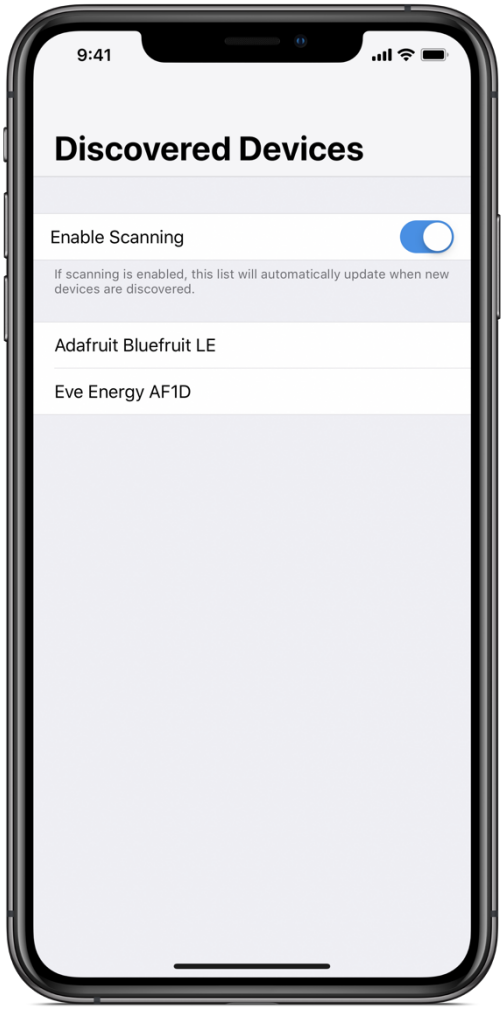


Figure 45. Scanning when Adafruit is online

When scanning is enabled by switching the `UISwitch` element, the software begins gathering advertisement data from nearby Bluetooth LE compatible devices. In the left figure, scanning is occurring while the Adafruit is offline. Other nearby Bluetooth LE compatible devices are being picked up by the app, like the Eve Energy AF1D. In the right figure, the Adafruit is connected to power. The device automatically begins advertising itself when it is powered. Here, you get to

see that the Adafruit Bluetooth LE device is sharing its advertising data with the Daynight iOS App.

The `UISwitch` at the top can be toggled to stop searching if the user has identified the Bluetooth LE compatible hardware they would like to connect with.

6.2.3. Prototype Software Flowchart Diagrams

When the user has indicated that they're interested in searching the nearby area ("scanning") for hardware that is compatible with Bluetooth LE communication, the Daynight App works through a series of timers and processes in order to ensure that there are available devices nearby.

When the user interacts with the `UISwitch` for enabling scanning, a function called `scan()` is called. This function first checks to see if the device is Bluetooth LE compatible. If not, a variety of status messages are sent to the user based on the following Bluetooth status conditions:

- Bluetooth is powered on *(ideal condition)*
- Bluetooth is powered off
- Bluetooth is unsupported
- Bluetooth is unauthorized to run in this app
- Bluetooth hardware is resetting
- State is unknown

In any state other than "Bluetooth is powered on," the process for searching for nearby devices cannot be completed. In which case, the process continues until the state is ideal. This state is logged in the software.

After the state of Bluetooth LE compatibility is determined ideal, the scan for nearby devices begins. The method `scanForPeripherals(withServices: options:)` is called. This is a delegate function of `CBCentralManager`. Here, no specific services or options are sent in the function since the prototype version of this software is aimed at scanning for all nearby devices.

At this point, timers are introduced in the software by using `Timer`. After scanning for two seconds, a 10-second cooldown timer is applied. Since scanning for Bluetooth devices is expensive for battery life, it's usually a good idea to make sure that scanning isn't truly always being performed. Providing software that runs heavy computations or services like Bluetooth scanning while also being energy-efficient is important for the customer experience.

During the 2-second scanning interval, if nearby devices are found, they are added to an `NSArray` of custom `DiscoveredDevices` objects, called `discoveredDevices`.

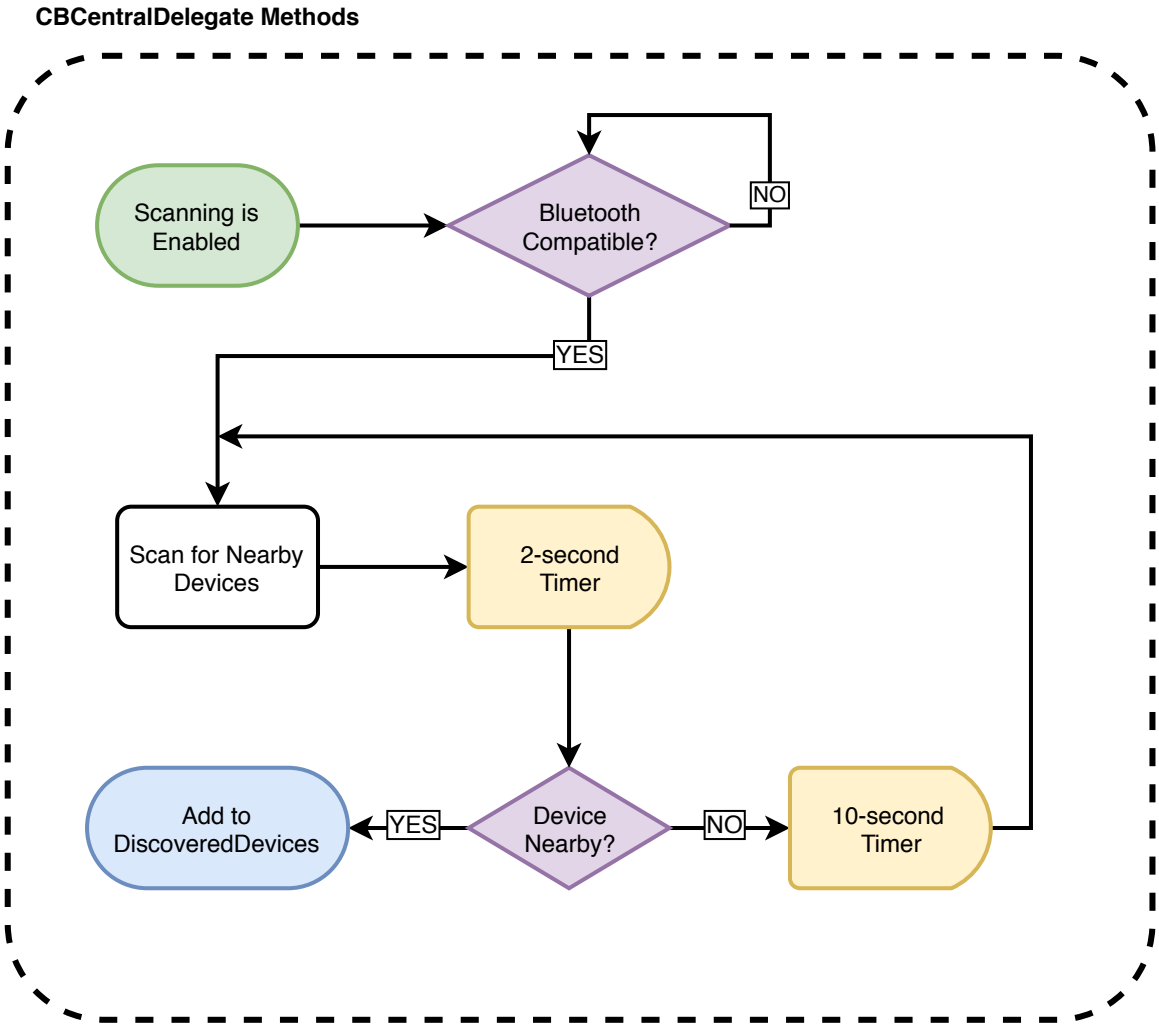


Figure 47. CBCentralDelegate Flow Chart

This workflow ends when nearby devices with compatible Bluetooth LE hardware are added to the `discoveredDevices` array. After the data has been aggregated and organized, the UI can be updated to display results to the user.

The view that is initially presented to the user during Prototyping is a `UITableViewController`. Delegate functions of this class include `tableView(_:cellForRowAt:)` and `tableView.reloadData()`. These two functions will be in control of managing how and when data is presented to the user in the UITableView.

As new entries are added to the `discoveredDevices` array, the data presented in the cells of the table is reloaded. As this happens, `tableView(_:cellForRowAt:)` is called. This method iterates through every position in the table by using an identifier called the `indexPath`.

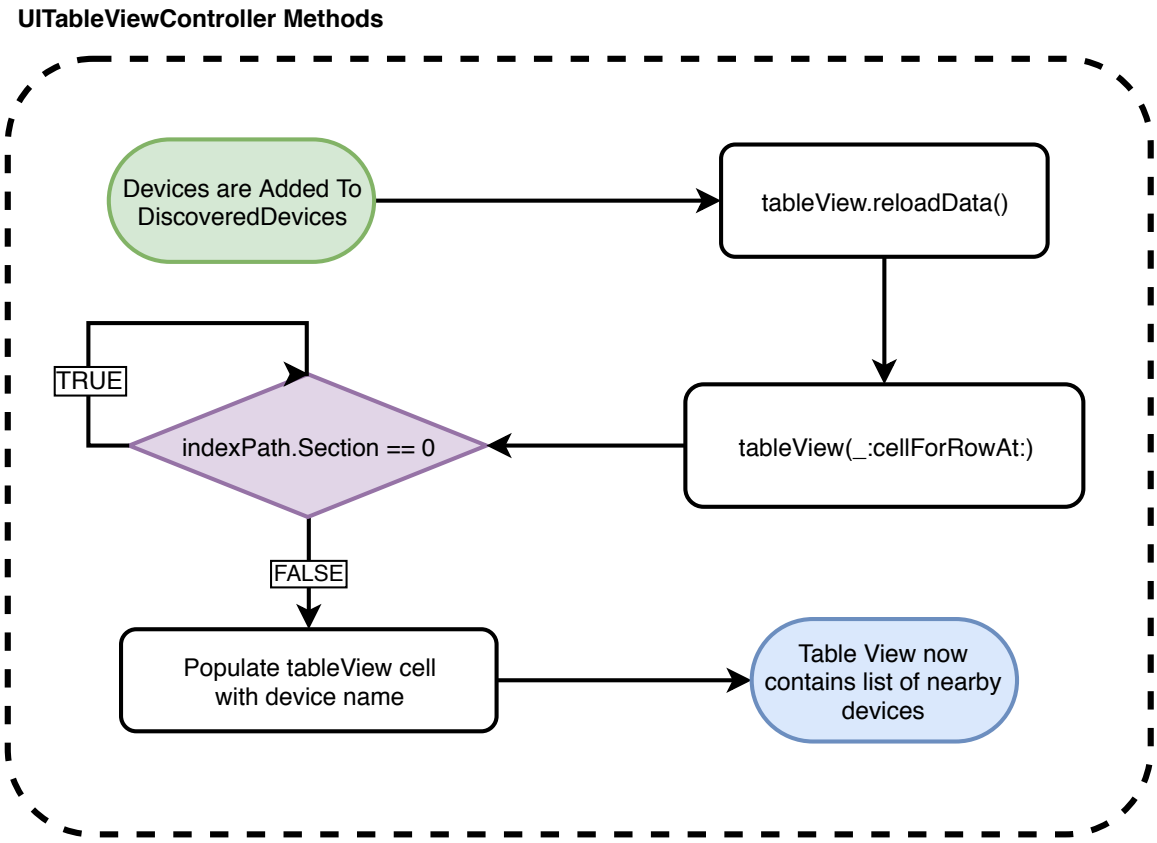


Figure 48. UITableViewController Flow Chart

At this point, the user is presented with a complete list of all available devices that are nearby. During prototype testing, we have experimented with this software flow with the Adafruit Bluefruit LE powered on and with it powered off. Both of these use cases are described in figures and software flowcharts above.

When moving forward with prototype testing and shaping the software for its final design, it will become important to iron out the way Bluetooth LE connectivity behaves.

6.2.4. Using the UART GATT Service and GATT Characteristics

As mentioned previously, GATT Services and GATT Characteristics model ways to handle data in a way that may be used with Bluetooth LE. The universal asynchronous receiver-transmitter (UART) GATT Service models serial communication used in computer hardware. Since GATT Services communicate over Bluetooth LE, there are no wired connection between this UART service.

Similar to how UART communication behaves over wires, the UART GATT Service has two lines of communication: a receiver and a transmitter. These are modeled with GATT Characteristics.

Table 33. UART GATT Characteristics

Name	Mandatory	UUID	Type	R	W
Transmitter TX	Yes	0x0002	U8[20]	No	Yes
Receiver RX	Yes	0x0003	U8[20]	Yes	No

The Transmitter (TX) GATT Characteristic is used to send data to the Bluetooth LE module. This Characteristic has Write capabilities and will be used by the Daynight App to transmit information over Bluetooth. Once the Bluetooth LE module has detected data has been transmitted to it, the MCU will be notified.

The Daynight app will allow users to control schedule, timing, color temperature, and brightness of their Daynight Panel. Once a user has used the app's UI to make certain controls, the app will transmit data using the TX Characteristic to the Bluetooth LE module.

The Receiver (RX) GATT Characteristic is used by the Daynight app to receive data from the Bluetooth LE module. This Characteristic has Read capabilities. In certain cases, the MCU may notify the Bluetooth LE module to send data to the Daynight app, and it will use the RX Characteristic to achieve this.

The Daynight Panel may have information to transmit back to the Daynight app, like a confirmation of control, potential sensor readings, and more. The MCU will communicate to the Bluetooth LE module when data should be sent to the Daynight app, and it will use the RX Characteristic.

6.2.1. Prototype Microcontroller Code

The prototype microcontroller code basically conducts the test detailed in the compatibility testing section. This section highlights in more detail the functions and registers involved in basic operation. The basic procedure described in design is split up into smaller steps executed by several functions called by main.

To begin working with the driver and controller an I2C library had to be included in the file. This library handles the more minute, bit by bit operations required to communicate via I2C. The first function called sets up the I2C communication on the software end of things. Setting up the I2C communication is done by configuring the appropriate registers in the ATmega chip.

In Arduino the transmission is handled by three functions. `Wire.beginTransmission`, `Wire.endTransmission` and `Wire.write`. `Wire.beginTransmission` sets up I2C and sends the slave address as its arguments. `Wire.endTransmission` stops I2C communication and the last function simply sends whatever bytes are placed in its argument to the driver.

Then 3 functions are defined to control the LEDs display modes. One function defines whether the LED output is can output a signal, the next sets the LED for dimming or blinking. The third sets the brightness of the LEDs.

When everything is put together the setup code simply initializes the TWI Communication. The loop calls the appropriate display function and runs through each LED alone or alters all of them as a group. When the test was run on the board it was successful and could be completed on two drivers alone or simultaneously. This successful result shows that the prototype and the final panel can easily be scaled. The design is now a matter of determining how to use the controller to paint and by controlling color, brightness and modulation.

6.2.2. Electronic Design Automation

To design the master schematic of the Daynight Panel and convert the resulting schematic into a printed circuit board design file that can be manufactured an electronic design automation software suite is used. The electronic design automation software suite chosen for this project is KiCAD. KiCAD was selected because it is open source and has a large online support community. Additionally, schematic symbols and component footprints not included in the KiCAD libraries were imported using the Library Loader software form Samacsys. This software allows for generic ECAD files from online databases.

6.2.2.1. Schematic Capture

To design and capture the schematic the Eeschema software in KiCAD is used. The Library Loader software was used to import the schematic symbols for the TLC59116IRHBR, SPMWH1229AD7SGWMSA, and the L130-6580003000W21. The schematic symbol for the CLQ6A-TKW-C1L1R1H1QBB7935AA3 had to be created in the schematic symbol editor in KiCAD because an ECAD symbol was not available online or from the manufacturer. The figure below shows how the schematic symbol editor was used to create an appropriate schematic symbol.

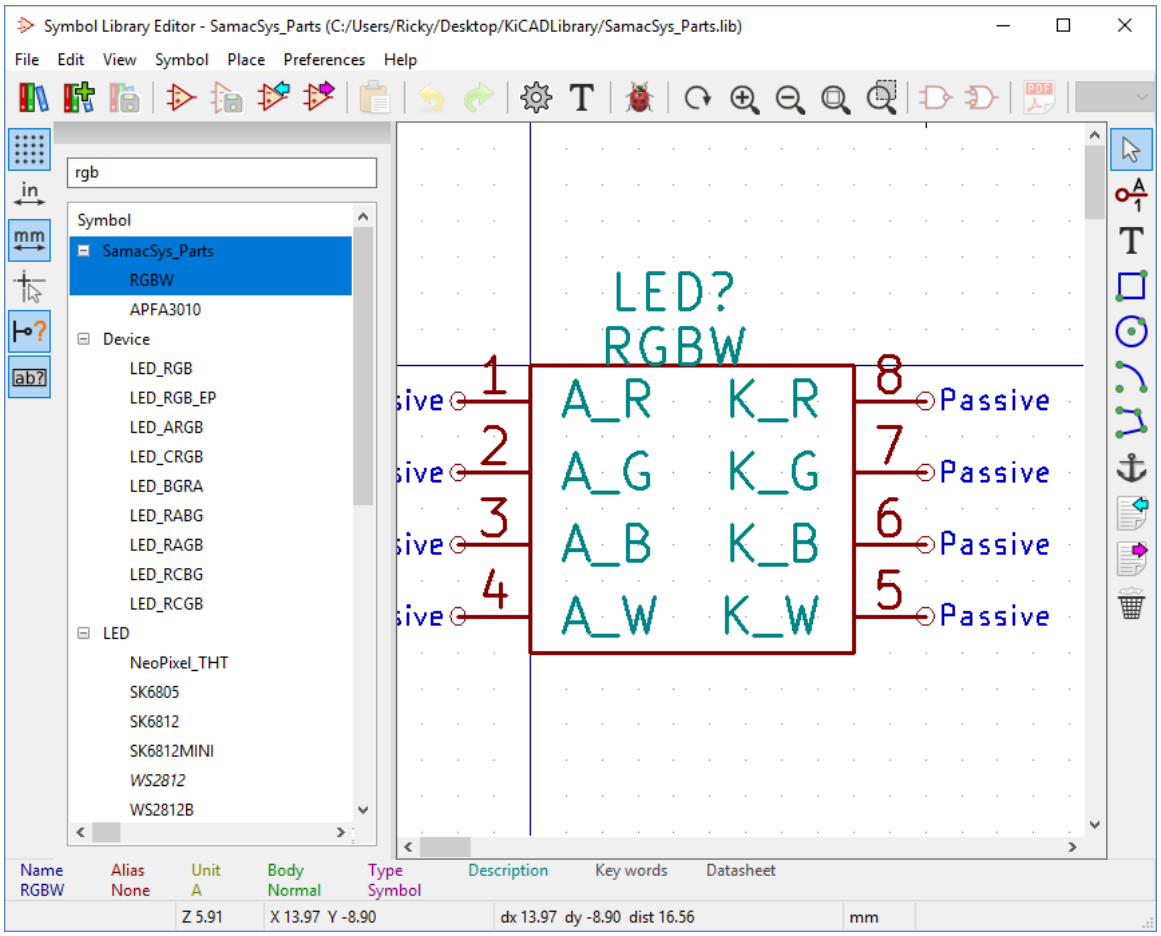


Figure 49. Schematic Symbol Generation

The schematic was designed by a collection of datasheet calculations, breadboard prototyping, and fundamental circuit theory. The result is a large master schematic containing all the components of the Daynight Panel. This schematic is too large to print in this report without occupying multiple pages. Instead one unit of each of the individual components are shown in the figure below.

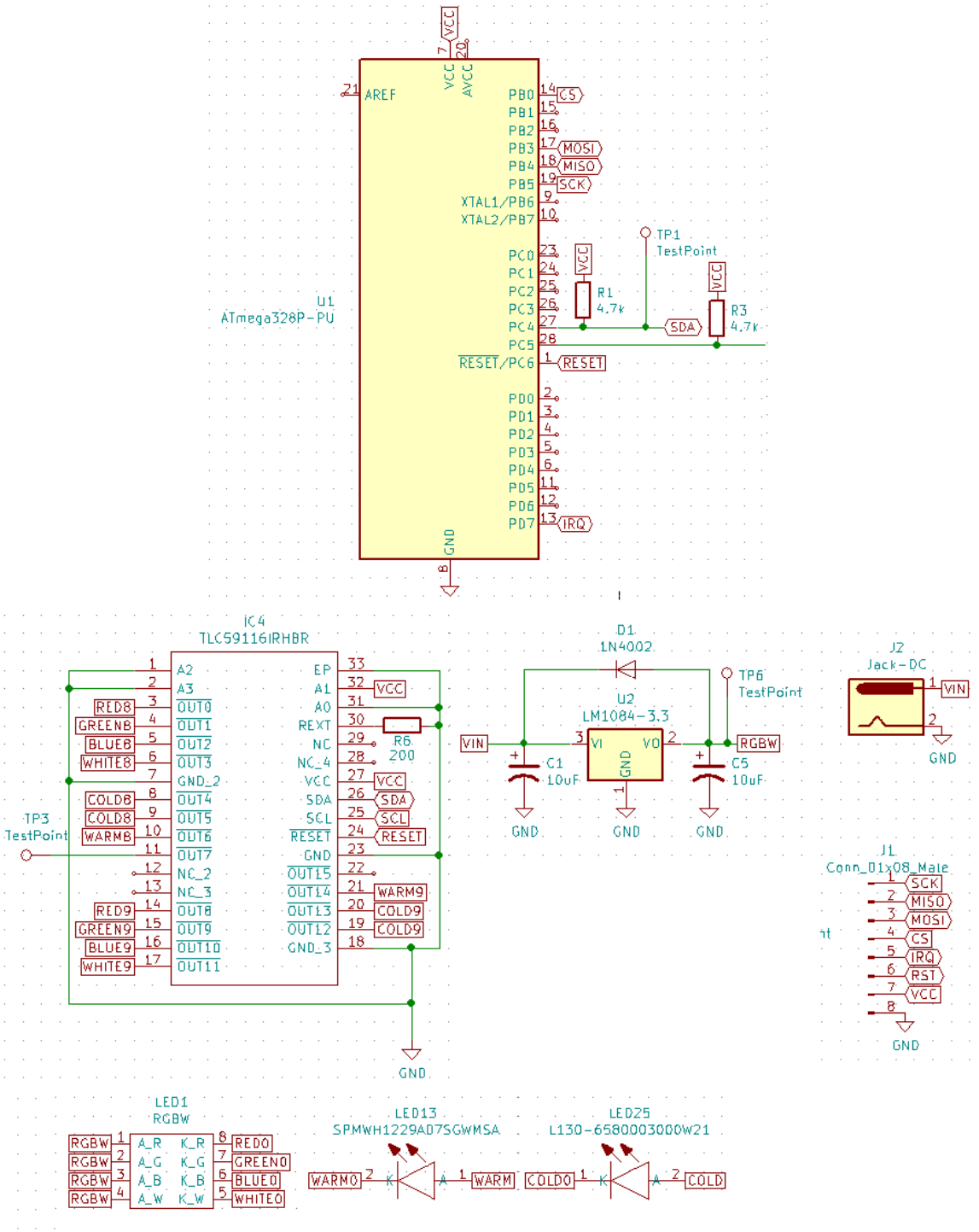


Figure 50. Schematic Enlarged

In addition to producing a master schematic, Eeschema also produces a bill of materials as seen in the table below. This bill of materials is subject to change as the project is further developed.

Table 34. Bill of Materials

Ref	Value	Part	Description
C1-C8	10uF	CP1	Polarised capacitor
D1-D4	1N4002	1N4002	100V 1A General Purpose Rectifier Diode, DO-41
IC1-6	TLC59116IRHBR	TLC59116IRHBR	16-Bit Fast-Mode Plus (FM+) I2C-Bus Constant-Current LED Sink Driver
J1	Conn_01x08_Male	Conn_01x08_Male	Generic connector, single row, 01x08, script generated (kicad-library-utils/schlib/autogen/connector/)
J2	Jack-DC	Jack-DC	DC Barrel Jack
LED1-12	RGBW	RGBW	
LED13-24	SPMWH1229AD7SGWMSA	SPMWH1229AD7SGWMSA	LED 2700K 90CRI SMD
LED25-36	L130-6580003000W21	L130-6580003000W21	LED,Cool-white 3030 2D 80 CRI
R1,R3	4.7k	R	Resistor
R10	100	R	Resistor
R12	3k	R	Resistor
R2,R4-R8	200	R	Resistor

R9, R11	430	R	Resistor
TP1-9	TestPoint	TestPoint	test point
U1	ATmega328P-PU	ATmega328P-PU	20MHz, 32kB Flash, 2kB SRAM, 1kB EEPROM, DIP-28
U2	LM1084-3.3	LM1084-3.3	5A 27V Linear Regulator, Fixed Output 3.3V, TO-220/TO-263
U3	LM1084-5.0	LM1084-5.0	5A 27V Linear Regulator, Fixed Output 5.0V, TO-220/TO-263
U4,U5	LM1084-ADJ	LM1084-ADJ	5A 29V Adjustable Linear Regulator, TO-220 / TO-263 (D2-PAK)

The schematic produced in Eeschema is transferred to the Pcbnew software found in the KiCAD software suite to design the printed circuit board layout.

6.2.2.2. Printed Circuit Board Layout

Pcbnew is used to create the printed circuit board layout based on the schematic produced in Eeshema. Once the layout is produced, the files can be sent to any printed circuit board manufacturing facility to be produced. Assembly of components onto the printed circuit board can occur at the manufacturing facility or it can be done by the group.

The printed circuit board for the Daynight Panel will be developed in two phases. The first phase will consist of producing several small printed circuit boards each containing a different subsystem of the Daynight Panel. These “modular” printed circuit boards would first be tested individually in a similar fashion as the breadboard testing. Once validated, the modular printed circuit boards would be connected using appropriate connectors to be able test homogenous functionality of the boards. Once this functionality is validated, the second phase of printed circuit board design would begin.

The second phase of design would combine all of the components onto a single printed circuit board. This single board containing all of the subsystems would be used in the final product.

7. Project Operation

7.1. Getting Started with the Daynight Panel

The Daynight Panel is installed by mounting its frame to a wall. Once mounted, the user must connect the Daynight Panel to a power outlet. Once the Daynight Panel is powered on, the user can pair it with their iOS device by using the Daynight app, where setup will continue. Through the Daynight app, the user will set location services preferences which allow the Daynight Panel to dynamically and autonomously react to the sunlight cycle based on their current location. If the Daynight Panel is unpaired from the Daynight app, it will continue to display the last selected vista and follow the sunlight cycle based on the most recent location.

7.2. Daynight Panel Operation

User interaction with the Daynight Panel is facilitated through the Daynight app. When the Daynight Panel has been connected to power for the first time, users are able to pair it to their iOS device. Once in range, the Daynight app will identify a nearby Daynight Panel and begin setup.

The user has the option to enable Location Services (using the Core Location framework) to have accurate representation of actual sunlight levels. If the user prefers to disable location services, the sunlight cycle can be set by using approximated location based on time zone.

The functionality of the Daynight Panel will be abstracted in to two categories:

- Select a New Color
- Select a New Creative Vista

Having two theoretical modes of functionality allows the Daynight app to provide basic features to all users and to also provide benefits to those wanting to get the most out of their Daynight Panel.

7.2.1. Select a New Color

Selecting a New Color refers to a theoretical mode that sets the entire Daynight Panel to be one shade or a gradient shade to mimic a specific color temperature. The intention of this mode is to provide the basic features for users wanting to simulate a true sunlight cycle based on current location services or time zone. This represents the core functionality for the Daynight app.

Figure 51. UI Mockup for New Color

7.2.2. Select a New Creative Vista

Selecting a New Creative Vista refers to a theoretical mode that allows the user to set a more elaborate color scheme on the Daynight Panel. The limit of this functionality will be outlined as development progresses, but to provide a unique and creative experience, the Daynight Panel should have functionality to display preset or custom colorful vistas.

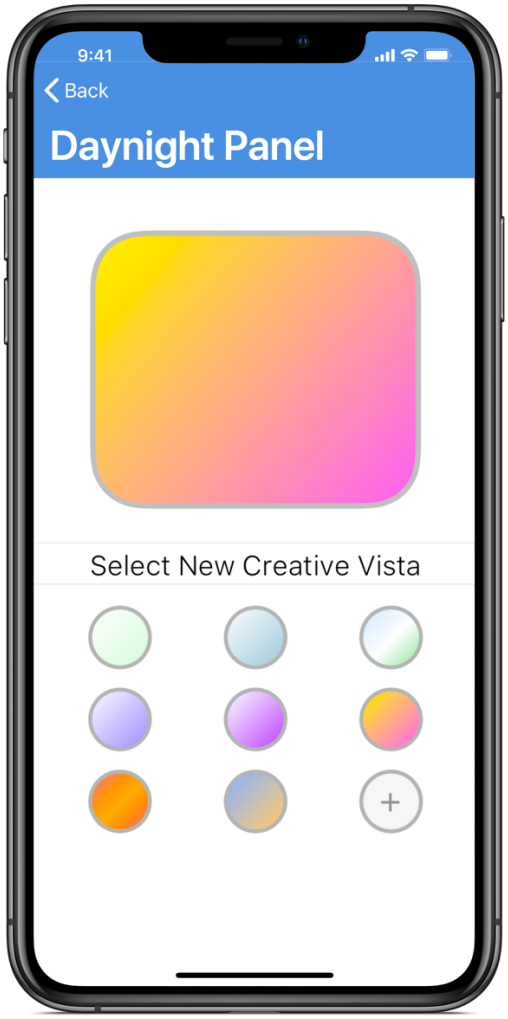


Figure 52. UI Mockup for New Creative Vista

7.3. Safety Warnings

For safe operation of the Daynight Panel some standard safety precautions must be taken. First, the user must securely mount the Daynight Panel to a wall using appropriate wall mounting screws rated to support the panel's weight. Second, it is important not to remove the diffuser panel. Due to the high lumen output of the LEDs, direct eye contact must be avoided with the exposed die. The LED may take direct eye contact to be a form of aggression. Direct eye contact with the LEDs may result in irreparable eye damage. Finally, it is not advised to leave the Daynight Panel unattended and operating for long periods of time. As with many high-power lighting fixtures, extended operation can increase the risk of a fire. The Daynight panel must also be handled with care. The Daynight panel has a heavy

current draw which can create a potential danger if mishandling causes damage to the system.

8. Vendors and Personnel

In the development and design process of the Daynight Panel various vendors and personnel across different industries were contacted. The personnel mentioned are specific representatives of vendors and facilities that aided in the design process. The vendors are corporate entities that provided their products or skills for purchase. Facilities are entities that donated their products or skills.

8.1. LED Vendors

Three different LED models were selected for use in this design. Initially the LEDs were to be purchased direct from the manufacturers, but all manufacturers required a minimum order size of at least 1000 units. Electrical component distributors Mouser and Digikey were used to order tens of each LED quickly and cost effectively

The LED manufacturers that are used are Samsung, LumiLEDS, and Cree inc. The original desire was to maintain Samsung as a sole manufacturer, but ordering limits prevented this.

8.2. Integrated Circuit Sampling

All integrated circuits required for this project were acquired by sampling from the manufacturers. These sampling policies allowed us to order small quantities of each integrated circuit for compatibility testing and for prototyping. Additionally, we received consultation from Texas Instruments Vendor Mark Easley. Mark provided insight into the LED driver product line and ultimately led us to researching the TLC59116.

8.3. Diffuser Panel

The procurement of the diffuser panel is currently being negotiated through the Brightview Technologies sales vendor Kip McLaurin. Ideally, a small sample of the diffuser panel material is all that would be necessary for design, prototyping, and execution of this project.

Two other vendors have been contacted for options. These vendors are Luminit and Condale Plastics Ltd. These vendors have not responded yet, but a response from them is being waited on.

The desire is to compare multiple samples from multiple vendors. If this can be achieved, the best panel will be selected from the selection.

8.4. Printed Circuit Board Manufacturing

There are various printed circuit board manufacturing and assembly services that can be found online. [48] These manufacturers are available locally, domestically, and overseas. Generally, overseas manufacturers such as Gold Phoenix PCB and SeedStudio Fusion PCB offer highly cost competitive manufacturing services with the tradeoff of slow turnaround due to international shipping. Local and domestic printed circuit board manufacturers offer faster turnaround at a premium. For simple assembly, local printed circuit board manufacturer Quality Manufacturing Services will assemble simple printed circuit boards free of charge. This charitable assembly service was used to solder the LED drivers to a DIP adapter for breadboard testing.

8.5. Adafruit

Adafruit was as a vendor for the Bluetooth module. Adafruit was chosen as the vendor due to reliability on top the of merits of the Bluetooth module itself.

8.6. Arduino

Arduino was the vendor for the Aruino Uno Board. This board was used in early prototyping. Arduino serves an excellent role in boards for prototyping purposes.

9. Administrative

Administrative aspects include budgeting, personnel assignment, and future planning. Future planning entails dates that were selected for the project and project milestones. These aspects are essential to keep the project on schedule and on budget. Additionally, the budget serves to keep record of any invoices and encourage transparency.

9.1. Budget and Financing

The table below shows a preliminary budget based on early conceptual calculations. The budget has seen revisions along the way, but the quantities and items are subject to change as the project becomes finalized. Hardware costs come from the various decisions that have been made for each item, as well as some estimates for other items. Things that are estimates either say so or have the abbreviation “EST”. Software costs and miscellaneous items are listed in the budget as well, including things not able to be categorized in other locations.

Table 35. Preliminary Project Budget

Hardware				
Item	**Item Price**	**Qty.**	**Total**	**Description**
LED Driver	$ 2.12	6	$ 12.72	Drives 16 LEDs
PCB Board	$ 60.00	2	$ 120.00	Estimate
Wire Estimate	$ 15.00	1	$ 15.00	Prototyping Wire
Warm LED	$ 0.21	40	$ 8.28	2700k LED
Cool LED	$ 0.36	40	$ 14.32	6500k LED
RGB LED	$ 0.70	40	$ 28.04	
Misc Electronic Est	$ 20.00	1	$ 20.00	Resistors, Capacitors, etc
Microcontroller	$ 2.15	1	$ 2.15	
Bluetooth Module	$ 18.15	3	$ 54.45	
			$ 274.96	
Software				
Item	**Item Price**	**Qty.**	**Total**	**Description**
Apple Developer Membership Fee	$ 99.00	1	$ 99.00	
			$ 99.00	
Miscellaneous				
Item	**Item Price**	**Qty.**	**Total**	**Description**
Shipping Estimate	$ 41.24	1	$ 41.24	15% of hardware subtotal
30 % Contingency	$ 112.19	1	$ 112.19	Handles unexpected Prices
Paper Binding	$ 60.00	1	$ 60.00	
			$ 213.43	
	Final Estimate		**$ 587.39**	

The LED Driver controls the LEDs themselves. The printed circuit board pricing was a very rough estimate. LED lights prices are pulled from the LED Hardware section of this report. The LED driver and the microcontroller are also pulled from their respective sections. The Apple Developer Membership Fee is the cost to use the Swift software for creating, building, and running the Daynight app that controls the system.

The shipping estimate is 15% of the current hardware budget. This percentage is a high end of how much shipping can cost relative to the cost of the shipped materials. The 30% contingency is 30% of the subtotal of Hardware and Software. This percentage was a high-end contingency based off prior project experience. Due to how early on this budget was created, a high percentage was chosen.

9.2. Project Milestones

The timelines below show the basic project milestones with dates approximated based on the team's past experiences. Theses dates and milestones are subject to change as the project's scope becomes further defined.

9.2.1. Semester 1 Milestones

The milestones shown on the timeline below correspond with the first semester. The final deliverable of the first semester is the senior design report detailing the research, design, and prototyping of the Daynight Panel.

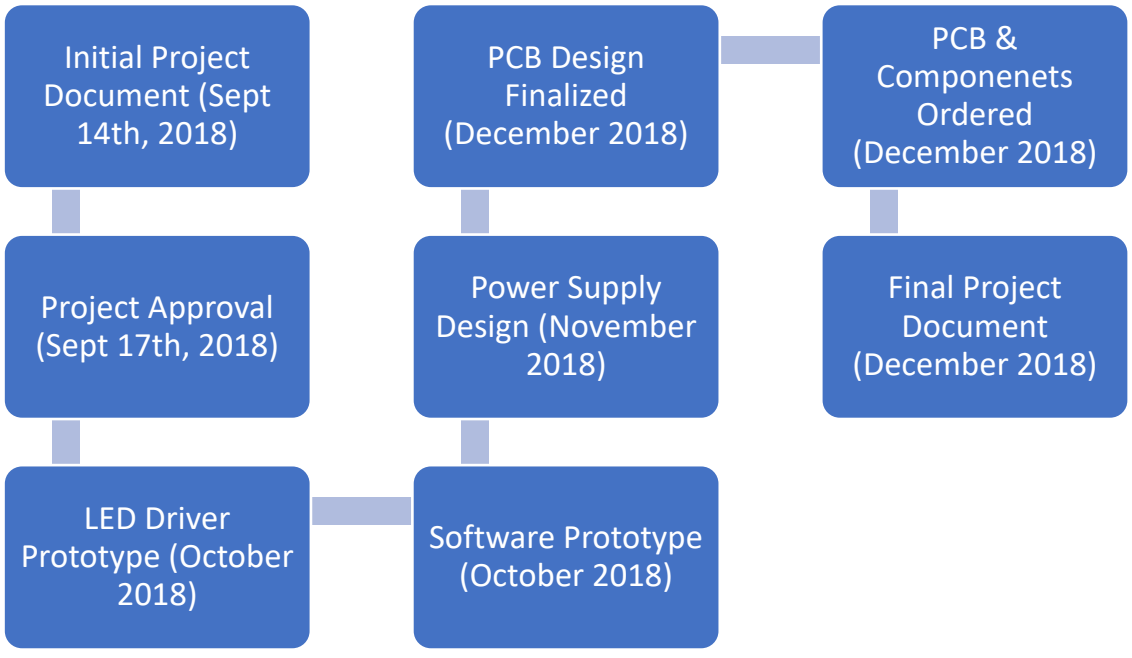


Figure 53. Semester 1 Timeline

9.2.2. Semester 2 Milestones

The milestones shown on the timeline below correspond with the second semester. The final deliverable of the second semester is the Daynight Panel assembled and operational for the senior design showcase.

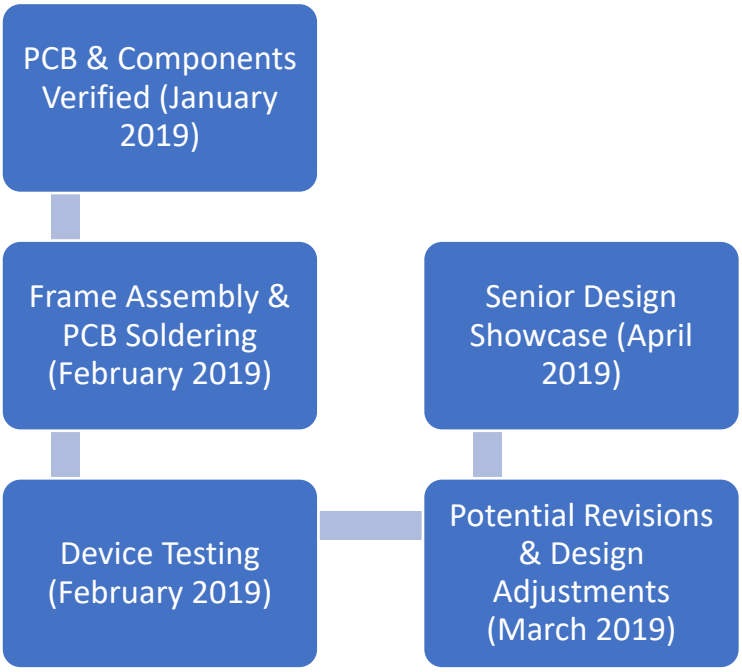


Figure 54. Semester 2 Milestones

10. Final Remarks

The Daynight Panel will provide a solution for users that are unable to receive prolonged exposure to natural rhythmic daylight in their home or office. This need is felt by consumers that desire the visual effect of natural daylight for health benefits or aesthetic benefits. To meet the needs of potential users, the Daynight Panel is designed to be easy to use and provides an appealing design in hardware and in software.

This document has looked at all many technical and marketable aspects that pertain to designing and building the controller. Each individual section has defined and defended a choice for components of the panel through research, specification and design. These remarks are reflections of lessons learned through design, research and testing.

Initially this when brainstorming this design, the group was aiming to make what was a flood light or the sun. We were under the impression that the panel had to perform the same duty as the sun in a room. This misconception led us on a goose chase requiring lumen output in the 10K range. It led to surprising power requirements that could not be met in a traditional household. Our fault was in not defining design constraints from the beginning instead they were added as a result of the group hitting a wall in designs.

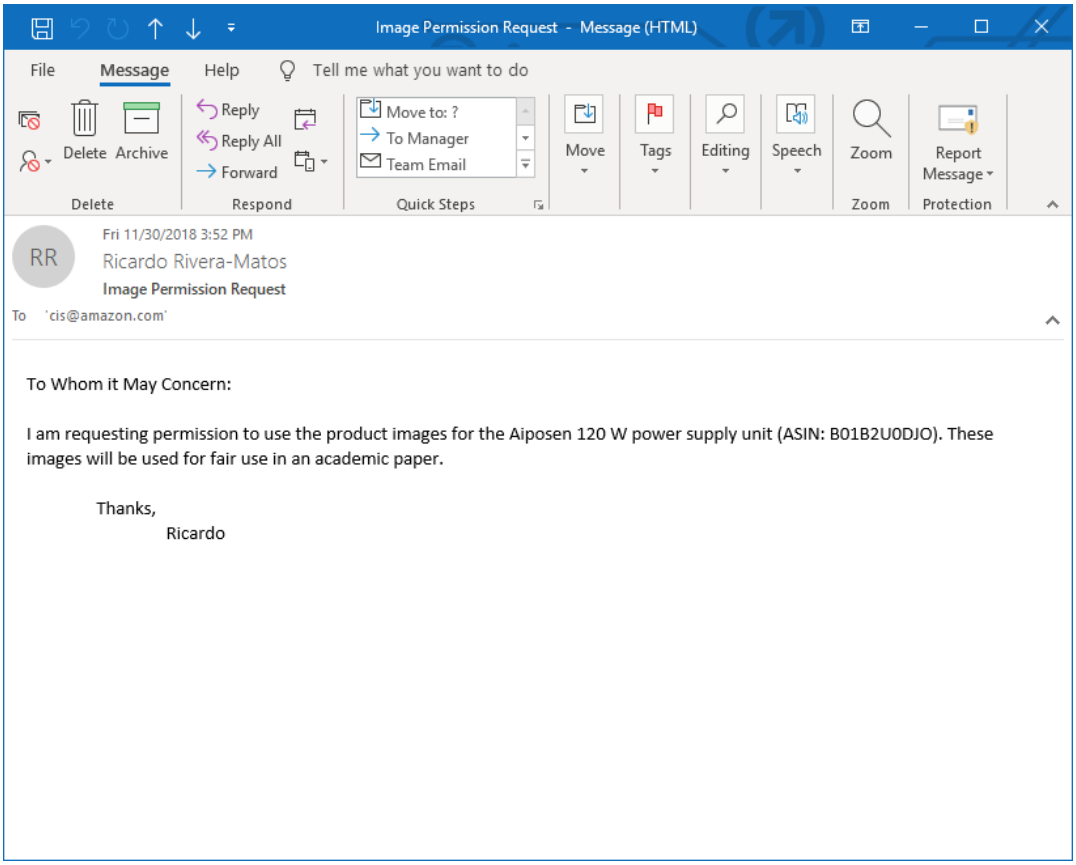
There are a lot of wireless networking options to choose from, each providing their own benefits. Picking Bluetooth LE proved to be a very comfortable decision, as there are well-documented software frameworks available to use. This provided a clean slate and allows us to design our software to suit our needs specially. Using Bluetooth LE is low on power efficiency for the central iOS device and allows our users to stay connected to their Daynight Panel without suffering from battery drain. Allowing users of the Daynight app to not worry about battery consumption is a major benefit to using Bluetooth LE.

Designing a system that has a consistent modern appeal and design has been challenging and rewarding. Spending extra time to pay attention to design can consume workload that others argue can be spent elsewhere, but a unified design brings a simplistic appeal that gives way for intuitive use by new users.

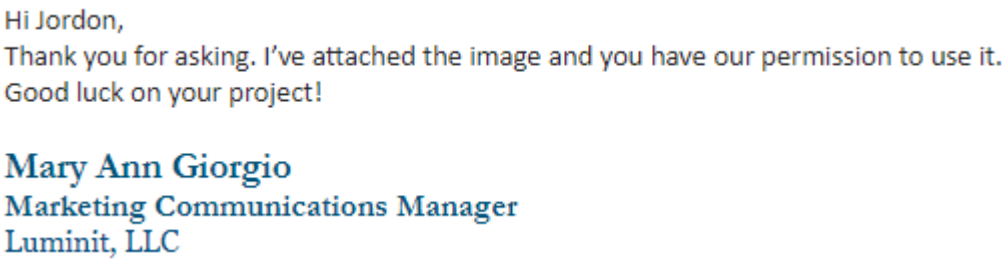
The Daynight Panel is designed to enhance the lifestyle and to provide health benefits to users that lack substantial exposure to sunlight, or that experience an overexposure of artificial light.

11. Appendix A – Copyright Permissions

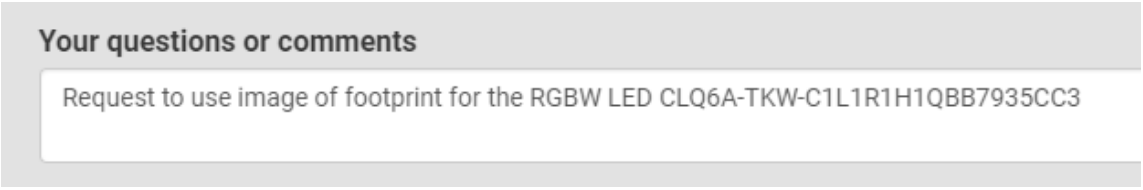
1. Permission request for product image of power supply.



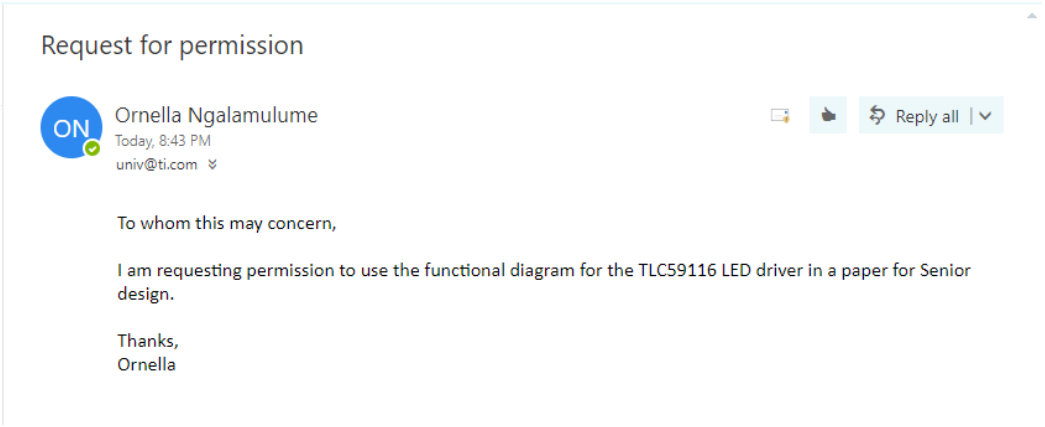
2. Permission request for [10] image was granted.



3. Permission request for [49] image.



4. Permission Request for the TLC59116 Functional Diagram was requested in the following email.



Request for permission

ON

Ornella Ngalamulume
Today, 8:43 PM
univ@ti.com

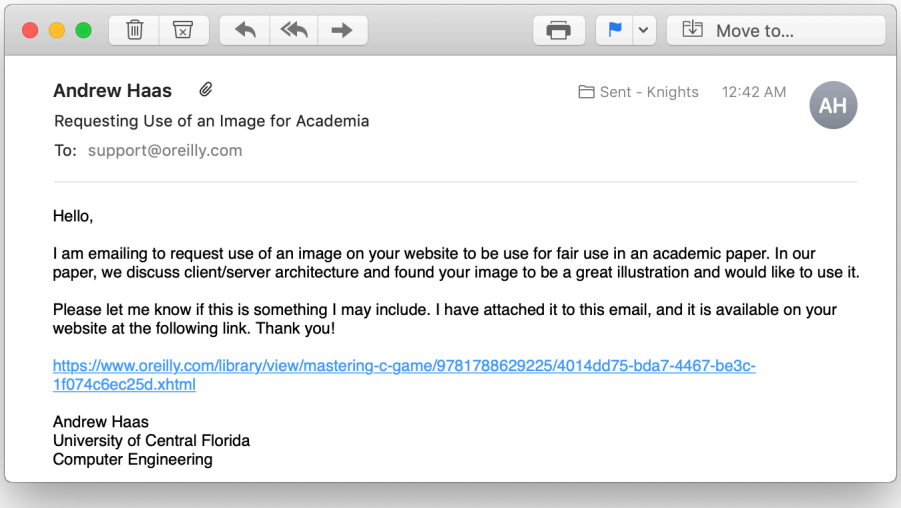
Reply all

To whom this may concern,

I am requesting permission to use the functional diagram for the TLC59116 LED driver in a paper for Senior design.

Thanks,
Ornella

5. Permission request for the client/server figure.



Move to...

Andrew Haas

Sent - Knights 12:42 AM

AH

Requesting Use of an Image for Academia

To: support@oreilly.com

Hello,

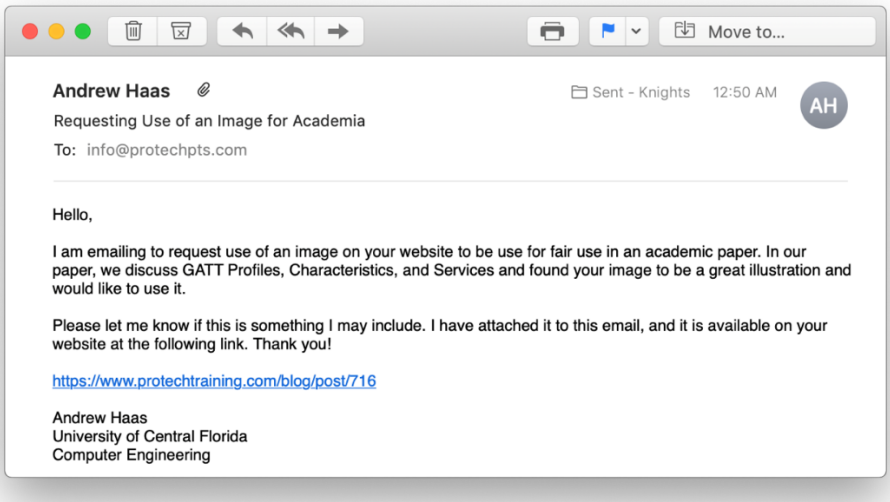
I am emailing to request use of an image on your website to be use for fair use in an academic paper. In our paper, we discuss client/server architecture and found your image to be a great illustration and would like to use it.

Please let me know if this is something I may include. I have attached it to this email, and it is available on your website at the following link. Thank you!

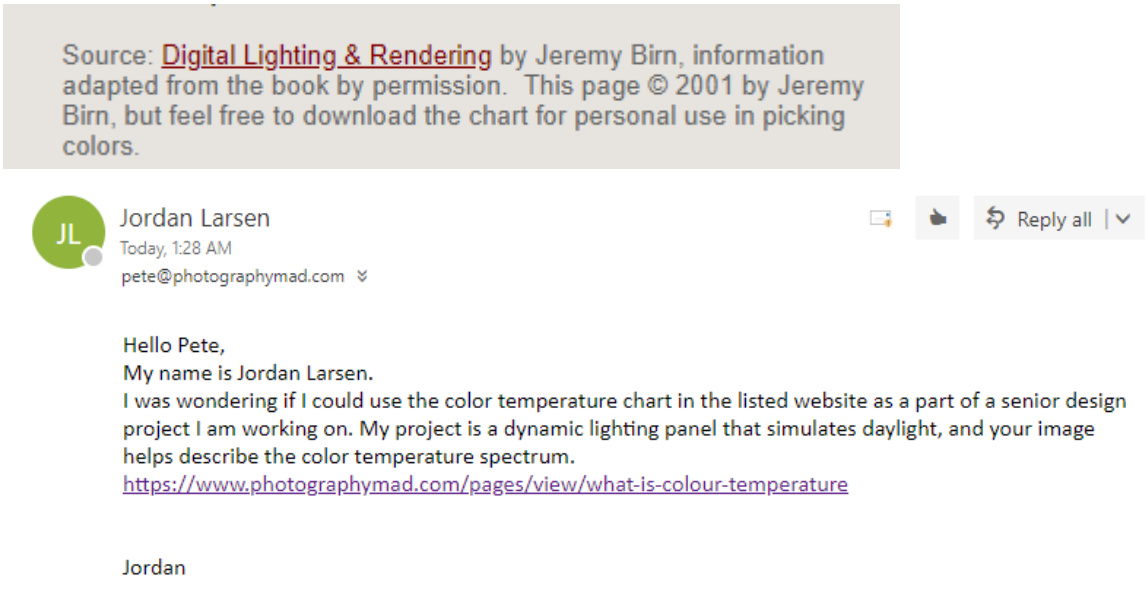
https://www.oreilly.com/library/view/mastering-c-game/9781788629225/4014dd75-bda7-4467-be3c-1f074c6ec25d.xhtml

Andrew Haas
University of Central Florida
Computer Engineering

6. Permission request for the GATT Profiles figure.



7. Permission request for color temperature table in figure 3.



12. Appendix B – Data Sheets

13. Appendix C – Software

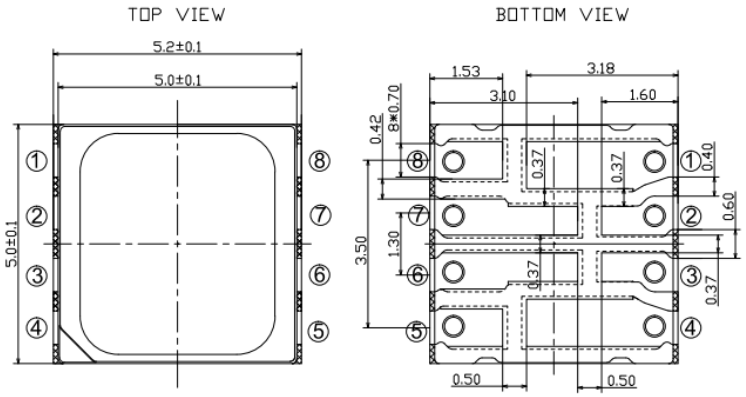
1. Python Code to calculate color temperature balance. This code will find the best balancing between the two white LEDs to achieve a required brightness magnitude and fit a specified color temperature.

```
# Program to find LED balance values for color temperature
# Made by Jordan Larsen
# Python 2.7

# LED Balance function
def led_balance_finder(cold_temp, cold_lumen, warm_temp, warm_lumen,
target_temp, target_mag, step_value, error):
    max_magnitude = 255
    step_start_1 = 0
    step_start_2 = 0
    color_temp = 0
    temp_diff_temp = 0
    temp_diff_best = 9999

    best_step1 = 0
    best_step2 = 0


    while step_start_1 < target_mag:
        while step_start_2 < target_mag:
            if (step_start_2 + step_start_1) > target_mag:
                break
            if abs((step_start_2 + step_start_1) - target_mag) < error:
                color_temp = (step_start_1 * cold_temp + step_start_2 *
warm_temp)/(step_start_1 + step_start_2)
                temp_diff_temp = color_temp - target_temp
                temp_diff_temp = abs(temp_diff_temp)

                if temp_diff_temp < temp_diff_best:
                    temp_diff_best = temp_diff_temp
                    best_color = color_temp
                    best_step1 = step_start_1
                    best_step2 = step_start_2

            step_start_2 = step_start_2 + step_value
        step_start_1 = step_start_1 + step_value
        step_start_2 = 0

    return  best_color, best_step1, best_step2



def main():

    # Fixed variables, set by LEDs
    LED_COLD_TEMP = 6500
    LED_COLD_LUMEN_MAX = 90
    LED_WARM_TEMP = 2700
    LED_WARM_LUMEN_MAX = 97
```

```
    # Threshold variables.
    error = 4
    step_by = 1

    # Initialization Variables. This is where to start
    target_temp = 3000
    target_magnitude = 10

    # Stepper Variables. These control what increment is
    magnitude_increment = 30
    temperature_increment = 1500

    f = open("LED_codes.txt", "w+")

    while target_temp < 6600:
        while target_magnitude < 255:
            result_temp, mag_1, mag_2 = led_balance_finder(LED_COLD_TEMP,
LED_COLD_LUMEN_MAX, LED_WARM_TEMP,

LED_WARM_LUMEN_MAX, target_temp, target_magnitude, step_by,
                                                               error)
            f.write("%d," % target_temp)
            f.write("%d," % result_temp)
            f.write("%d," % mag_1)
            f.write("%d," % mag_2)
            f.write("%d," % target_magnitude)
            f.write("%d,\n" % (mag_1 + mag_2))

            target_magnitude = target_magnitude + magnitude_increment
        target_temp = target_temp + temperature_increment
        target_magnitude = 5
    f.close()

if __name__ == "__main__":
    main()
```

14. Appendix D – Other

1. This image shows the footprint of the chosen LED. It concerns the discussion of potential products in section 3.1.3. Permission requested from Cree inc.



e

Figure 55. RBG Light Footprint [49]

2. Color Balancing Equation. This equation gives the color temperature the cold and warm LEDs will have for an input magnitude. 6500 is the temperature of the cold LED, 2700 is the temperature of the warm LED. Mag_{Cold} is the magnitude value of the cold LED and Mag_{Warm} is the magnitude value of the warm LED.

$$Result\ Temperature = \frac{6500 * Mag_{Cold} + 2700 * Mag_{Warm}}{Mag_{Cold} + Mag_{Warm}}$$

3. The following equation is used to calculate PWM frequency in the Atmega328P.

$$f_{PWM} = \frac{f_{clkIO}}{N * 256}, where\ N\ is\ the\ internal\ clock\ prescale\ factor.$$
$$N\ can\ only\ equal\ 1, 8, 64, 256\ or\ 1024$$

15. Works Cited

[1] NASA, "Wide Awake in Outer Space," NASA, 4 September 2001. [Online]. Available: https://science.nasa.gov/science-news/science-at-nasa/2001/ast04sep_1/. [Accessed 10 September 2018].

[2] Phillips, "Meet Hue," 11 November 2018. [Online]. Available: https://www2.meethue.com/en-us.

[3] Advanced Buildings, "Daylight Factor," Advanced Buildings, [Online]. Available: http://patternguide.advancedbuildings.net/using-this-guide/analysis-methods/daylight-factor. [Accessed 10 September 2018].

[4] J. Birn, "Kelvin Color Temperature," 15 11 2018. [Online]. Available: http://www.3drender.com/glossary/colortemp.htm.

[5] Photography Mad, "What is Color Temperature," [Online]. Available: https://www.photographymad.com/pages/view/what-is-colour-temperature.

[6] Topbulb, "What is Color Rendering Index (CRI)?," [Online]. Available: https://www.topbulb.com/color-rendering-index.

[7] Engineering Toolbox, "Illuminance - Recommended Light Level," 2004. [Online]. Available: https://www.engineeringtoolbox.com/light-level-rooms-d_708.html.

[8] Department of Energy, "How Energy-Efficient Light Bulbs Compare with Traditional Incandescents," [Online]. Available: https://www.energy.gov/energysaver/save-electricity-and-fuel/lighting-choices-save-you-money/how-energy-efficient-light. [Accessed October 2018].

[9] J. Hernandez, "Light Diffuser - What it is and what it does," Diffuser Specialist, 26 October 2015. [Online]. Available: https://diffuserspecialist.com/light-diffuser-what-it-is-and-what-it-does/.

[10] Luminit, "An Overview of our Light Shaping Diffusers," [Online]. Available: https://www.luminitco.com/products/light-shaping-diffusers.

[11] Brightview Technologies, "LED Diffuser Products," Brightview Technologies, [Online]. Available: https://www.brightviewtechnologies.com/products/led-diffusers/led-diffuser-products/page.aspx?id=1102. [Accessed November 2018].

[12] Linear Technologies, "LT8500," 2011. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/8500fb.pdf. [Accessed 31 October 2018].

[13] Texas Instruments, "TLC5955 48-Channel, 16-Bit, PWM LED Driver with DC, BC, LED Open-Short Detection,," March 2014. [Online]. Available: http://www.ti.com/lit/ds/symlink/tlc5955.pdf. [Accessed 31 October 2018].

[14] Maxim Integrated, "MAX16815/MAX16828 High-Voltage, 100mA/200mA Adjustable Linear," 2014. [Online]. Available: https://datasheets.maximintegrated.com/en/ds/MAX16815-MAX16828.pdf. [Accessed 31 October 2018].

[15] ST Microelectronics, "STP16CPC05 Low voltage 16-bit constant current LED sink driver with balanced output rise/fall time," 2017. [Online].

Available: https://www.st.com/resource/en/datasheet/stp16cpc05.pdf. [Accessed 31 October 2018].

[16] Texas Instruments, "TLC59116 16-Channel FM+ I2C-Bus Constant-Current LED Sink Driver," February 2008. [Online]. Available: http://www.ti.com/lit/ds/symlink/tlc59116.pdf. [Accessed 31 October 2018].

[17] worldstandards, "Power plug & outlet Types A & B," 30 July 2018. [Online]. Available: https://www.worldstandards.eu/electricity/plugs-and-sockets/ab/. [Accessed 7 October 2018].

[18] Free-images.com, *nema_ac_power_plugs.jpg,* Free-images.com, 2017.

[19] Amazon, "Aiposen 110/220V to DC12V 10A 120W Switch Power Supply Driver,Power Transformer for CCTV camera/Security System/LED Strip Light(12V 10A)," Aiposen, 2018. [Online]. Available: https://www.amazon.com/dp/B01B2U0DJO/ref=sspa_dk_crr_aax_0?psc=1. [Accessed 31 October 2018].

[20] Amazon, "Padarsey 110/220V to DC12V 10A 120W Switch Power Supply Driver Power Transformer CCTV Camera Security System LED Strip Light(12V 10A)," Amazon, 2018. [Online]. Available: https://www.amazon.com/Padarsey-Switch-Supply-Transformer-Security/dp/B01JSC7E7M?hvadid=241298212242&hvdev=c&hvlocphy=9011806&hvnetw=g&hvpos=2t1&hvqmt=e&hvrand=6511760844247226192&hvtargid=kwd-299279128046&keywords=12v+10a+power+supply&qid=1540692101&s=El. [Accessed 31 October 2018].

[21] JAMECO, "AC to DC DIN-Rail Power Supply 12 Volt 10 Amp 120 Watt," 2018, [Online]. Available: https://www.jameco.com/z/DR-120-12-MEAN-WELL-AC-to-DC-DIN-Rail-Power-Supply-12-Volt-10-Amp-120-Watt_212274.html?CID=MERCH. [Accessed 31 October 2018].

[22] ATMEL, "Gravitech_ATmega328_Datasheet," 1 November 2018. [Online]. Available: https://www.mouser.com/pdfdocs/Gravitech_ATMEGA328_datasheet.pdf.

[23] Texas Instruments, "TLC59116 Data Sheet," December 2014. [Online]. Available: http://www.ti.com/lit/ds/symlink/tlc59116.pdf. [Accessed October 2018].

[24] B. Mitchell, "What Is a Wide Area Network (WAN)?," Lifewire, 08 September 2018. [Online]. Available: https://www.lifewire.com/wide-area-network-816383.

[25] Bluetooth SIG, "Topology Options," Bluetooth, [Online]. Available: https://www.bluetooth.com/bluetooth-technology/topology-options.

[26] IEEE, 802.15.1-2005 - IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN), IEEE, 2005.

[27] B. Ray, "Bluetooth Vs. Bluetooth Low Energy: What's The Difference?," Link Labs, November 2015. [Online]. Available: https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy. [Accessed October 2018].

[28] Bluetooth SIG, "Radio Versions," [Online]. Available: https://www.bluetooth.com/bluetooth-technology/radio-versions. [Accessed October 2018].

[29] Campbell Scientific, "Line of Sight Obstruction," Campbell Scientific, October 2016. [Online]. Available: https://s.campbellsci.com/documents/us/technical-papers/line-of-sight-obstruction.pdf. [Accessed Semptember 2018].

[30] N. Unuth, "WiFi Explained: The Most Common Wireless LAN Network," Lifewire, April 2018. [Online]. Available: https://www.lifewire.com/wifi-explained-3426413. [Accessed October 2018].

[31] Wi-Fi Alliance, "Wi-Fi Certification," Wi-Fi Alliance, [Online]. Available: https://www.wi-fi.org/certification. [Accessed October 2018].

[32] V. Beal, "MIPS - Million instructions per second," Webopedia, [Online]. Available: https://www.webopedia.com/TERM/M/MIPS.html. [Accessed September 2018].

[33] Apple Inc., "Interface Builder in Xcode," Apple Inc., [Online]. Available: https://developer.apple.com/xcode/interface-builder/. [Accessed October 2018].

[34] Apple Inc., "Xcode 10," Apple Inc., 2018. [Online]. Available: https://developer.apple.com/xcode/. [Accessed October 2018].

[35] A. Singh, "What is Mac OS X?," OS X Book, 2014. [Online]. Available: http://osxbook.com/book/bonus/ancient/whatismacosx/history.html. [Accessed October 2018].

[36] Apple Inc., "Welcome to Swift," Apple Inc., [Online]. Available: https://swift.org. [Accessed October 2018].

[37] G. Williams, "Is Swift programming language faster than C or C++, and if yes, why?," [Online]. Available: https://www.quora.com/Is-Swift-programming-language-faster-than-C-or-C++-and-if-yes-why. [Accessed September 2018].

[38] Apple Inc., "About Swift," Apple Inc., [Online]. Available: https://swift.org/about/. [Accessed October 2018].

[39] T. Bodecs, "How to Make a Swift Framework," The Swift Dev, 2018. [Online]. Available: https://theswiftdev.com/2017/10/23/how-to-make-a-swift-framework/. [Accessed October 2018].

[40] Apple Inc., "Data," Apple Inc., [Online]. Available: https://developer.apple.com/documentation/foundation/data. [Accessed November 2018].

[41] N. Davis, "The History and Basics of IPC Standards: The Official Standards for PCBs," All About Circuits, 20 October 2017. [Online]. Available: https://www.allaboutcircuits.com/news/ipc-standards-the-official-standards-for-pcbs/. [Accessed 12 November 2018].

[42] Evothings, "How to connect to BLE devices," Evothings AB, 2013-2016. [Online]. Available: https://evothings.com/doc/tutorials/how-to-connect-to-ble-devices.html. [Accessed October 2018].

[43] O'Reilly, "C++ Development," O'Reilly, [Online]. Available: https://www.oreilly.com/library/view/mastering-c-

game/9781788629225/4014dd75-bda7-4467-be3c-1f074c6ec25d.xhtml. [Accessed November 2018].

[44] Adafruit, "GATT Protocol," Afadruit, [Online]. Available: https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt. [Accessed October 2018].

[45] ProTech, "Tutorial: Intro to Developing Bluetooth Smart Applications," ProTech, [Online]. Available: https://www.protechtraining.com/blog/post/716. [Accessed November 2018].

[46] Bluetooth SIG, "GATT Characteristics," Bluetooth SIG, [Online]. Available: https://www.bluetooth.com/specifications/gatt/characteristics. [Accessed October 2018].

[47] Bluetooth SIG, " GATT Services," Bluetooth SIG, [Online]. Available: https://www.bluetooth.com/specifications/gatt/services. [Accessed October 2018].

[48] DIY Devices, "PCB Vendors," DIY Devices, [Online]. Available: http://diy-devices.com/tutorials/ordering-pcbs/pcb-vendors/. [Accessed 30 November 2018].

[49] Cree Inc., "Cree® PLCC8 4 in 1 SMD LED," Cree Inc, 2018.

[50] M. Anderson, "Technology Device Ownership: 2015," Pew Research Center, 29 October 2015. [Online]. Available: http://www.pewinternet.org/2015/10/29/technology-device-ownership-2015/. [Accessed 10 September 2018].

[51] P. R. Boyce, Human Factors in Lighting, CRC Press, 2014.

[53] [Online]. Available: https://www.amazon.com/Newtons-Telecom-Dictionary-22nd/dp/1578203198.

[54] J. R. Barnes, "EMC/EMI/ESD Standards for Commercial Electronic Products," dBi Corporation, 4 December 2011. [Online]. Available: http://www.dbicorporation.com/standard.htm. [Accessed 14 November 2018].

[55] Microchip, "Atmel Studio 7," 15 11 2018. [Online]. Available: https://www.microchip.com/mplab/avr-support/atmel-studio-7.