Solar-Powered Traffic Light

By

Richard Przybek

Colin Tarkowski

Bowen Xiao

Final Report for ECE 445, Senior Design, Spring 2022

TA: Qingyu Li

04 May 2022

Project No. 20

Abstract

This final report describes the motivation and design behind our solar-powered traffic light project. Our solar-powered traffic light system aims to reduce energy consumption, reduce light pollution, and ensure the safety of bikers/pedestrians. Our overall project was a success as we managed to meet all the previous goals mentioned despite some setbacks. We built a model version of our traffic light system with a single set of traffic lights. The scope of our project was limited by financial constraints and the engineering capabilities of the machine shop.

Contents

1 Introduction

1.1 Problem

Traffic lights are integral to our society, despite their relative lack of innovation over the years. The most significant change has been the switch from incandescent bulbs to LEDs in an attempt to reduce the power consumption of this necessary device. However, this has also led to an increase of light pollution due to the cooler, more intense light emitted by LEDs. They can cause extreme glare and pose a danger to drivers at night. Additionally, the issue of bicyclists and vehicles sharing the road can create many awkward or dangerous situations due to the lack of separation.

1.2 Solution

We propose a solar-powered traffic light system that will reduce light pollution and solve the issues of drivers and bicyclists sharing the intersection. Solar power will operate the system to minimize utility power used during the day. Connection to the grid is necessary for operation at night or when solar conditions are suboptimal. At night, pulse-width modulation (PWM) circuitry will dim the LED modules. This not only reduces light pollution, but also lowers utility consumption at night. In the case of adverse weather conditions, the system should not dim the lights to ensure proper visibility. Pedestrians and cyclists can alert the system of their presence by pressing a button and cross the intersection once their respective lights signal to do so.

1.3 High Level Requirements

- In order to limit light pollution and reduce consumption of utility power, the light modules must use PWM to operate with less power when it is sufficiently dark outside (assuming clear weather). We estimate our entire system to use approximately 22 watts of power, compared to a regular traffic light system with 3 LED bulbs which uses about 45 watts of power [1]. Should the power saved over an expected lifespan of about 20 years be greater than the initial cost of this system (and the other two requirements are met), we can conclude that this system is a success.
- The system must be able to switch between solar and utility power without causing a power failure. This is especially important when switching to utility power at night.
- The system must be able to efficiently adjust the light patterns and lengths of operation to increase the efficiency of traffic. Bicyclists and pedestrians can press one of two buttons to trigger the bicycle and walk signs for them to safely cross the intersection. Assuming a walking pace of 3 miles an hour and a lane width of 20 feet [2], the walk signal must be on for a minimum of 15 seconds to ensure everyone crosses.

2 Design

The overall design is shown in the block diagram in Figure 1. We have four subsystems: power, sensing, control, and traffic light. The sensing and control systems are responsible for detecting external input and changing the brightness and state of the traffic lights (by sending signals to the traffic light subsystem), as well as switching between grid and solar power. The power subsystem contains various converters to power the entire system, in addition to a switching network to switch between the two power sources.

Originally, we had a 3.3 V buck converter to power the humidity sensor, a power monitor in our sensing subsystem, and our 5 V buck was powered by the output of the switching network rather than the AC/DC converter. Our control and the majority of our sensing subsystem now runs on grid power to ensure that switching power sources does not impact our control flow and cause unwanted behavior. Additionally, we saved money and increased simplicity by discarding the 3.3 V buck. Finally, we had issues with the power monitor and went with a backup plan instead.

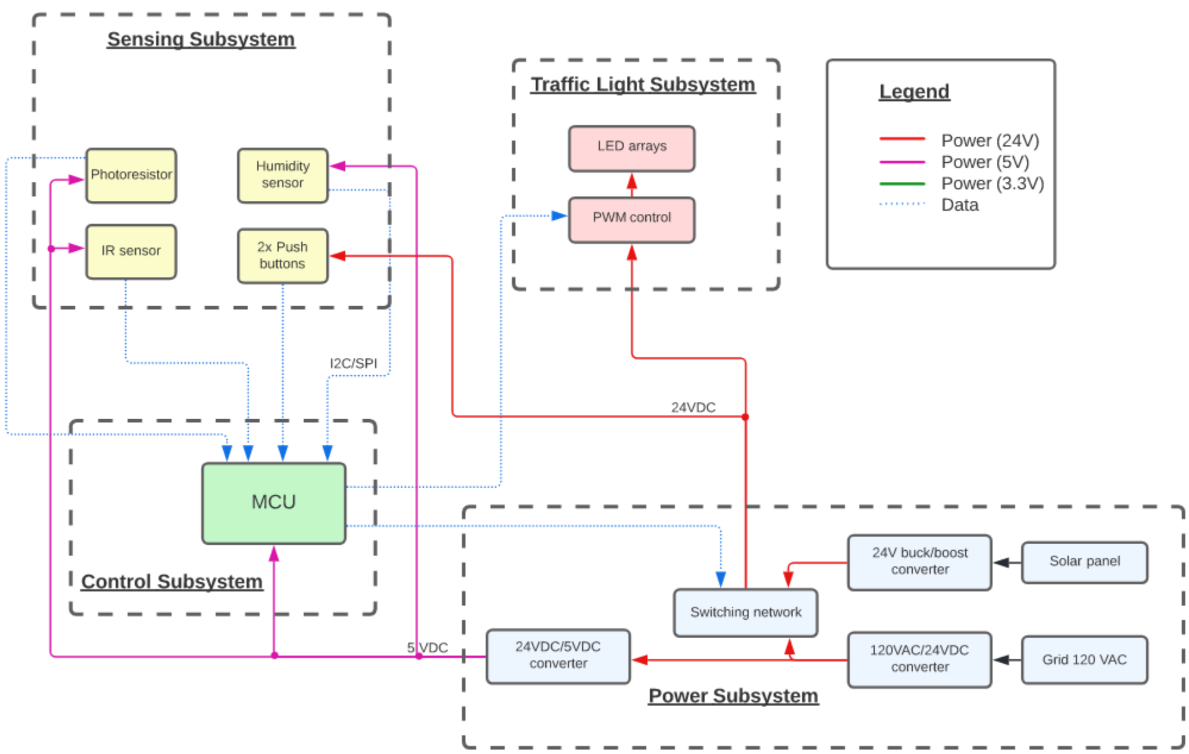

Figure 1: Block diagram

2.1 Power Subsystem

The power subsystem has a solar input and grid input and is able to switch between the two based on the amount of available light detected by the photoresistor. All other subsystems are powered by either the 5 V or 24 V output of this one. Refer to Appendices C.2 - C.4 to view the initial and final PCB designs of this subsystem.

2.1.1 Solar Panel

We designed the power subsystem around the solar panel located in lab, which has a nominal operating voltage of 18 V and maximum output power of 100 W. This is more than enough for our device, so we planned to partially cover the panel while operating it.

2.1.2 Power Converters

The power converter directly downstream of the solar panel must convert the nominal 18 V to 24 V. To accomplish this DC/DC conversion and provide a highly regulated voltage level, we chose a single-ended primary-inductor converter (SEPIC). The SEPIC topology requires a greater part count than other DC/DC converters, which is the trade-off of its higher efficiency and regulation. This conversion is given by the input-output relationship in Equation 1, where D is the duty cycle of the converter's MOSFET, V_{in} is the input voltage, and V_{out} is the output voltage. A generalized SEPIC schematic, which our design follows, can be seen in Figure 2. A wide input range of 12 to 25 V is selected to ensure that all currents remain within the limitations of our components and PCB. Upon selecting a SEPIC controller, a simulation can be viewed in Figure 3. All parts are included in the schematic in Figure 4. Refer to Appendix C.1 to view the SEPIC design calculations. It is important to note that this converter is designed around a worst-case scenario power consumption of approximately 40 W, which is sufficient for our device's operation in all cases.

$$V_{out} = V_{in}\frac{D}{1-D} \tag{1}$$

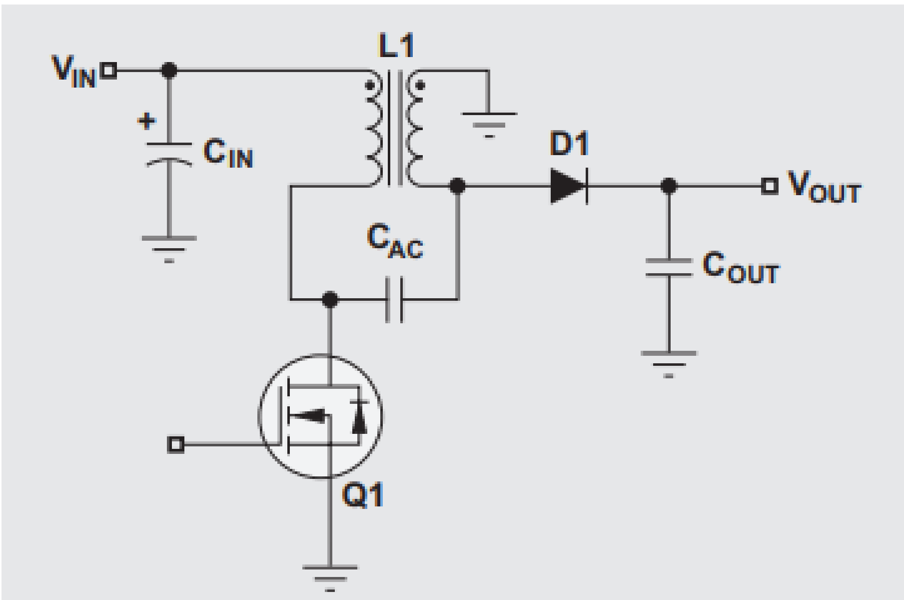

Figure 2: SEPIC topology diagram [3]

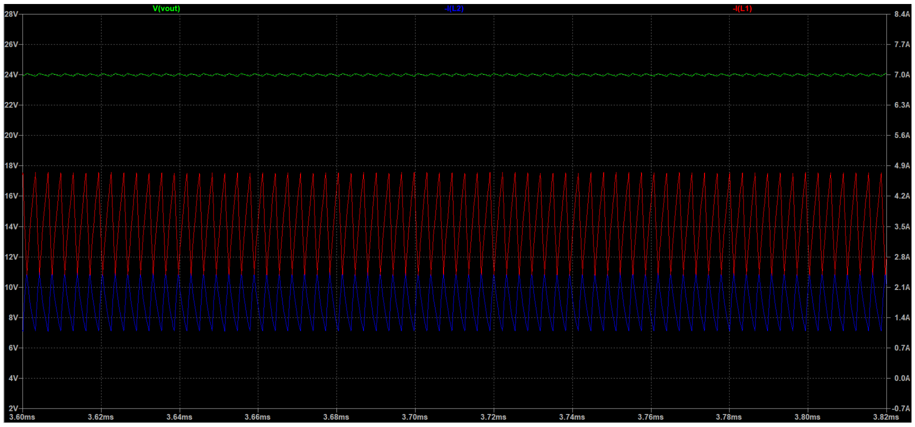

Figure 3: SEPIC simulation waveforms - 24 V output (green), primary inductor current (red), and secondary inductor current (blue)

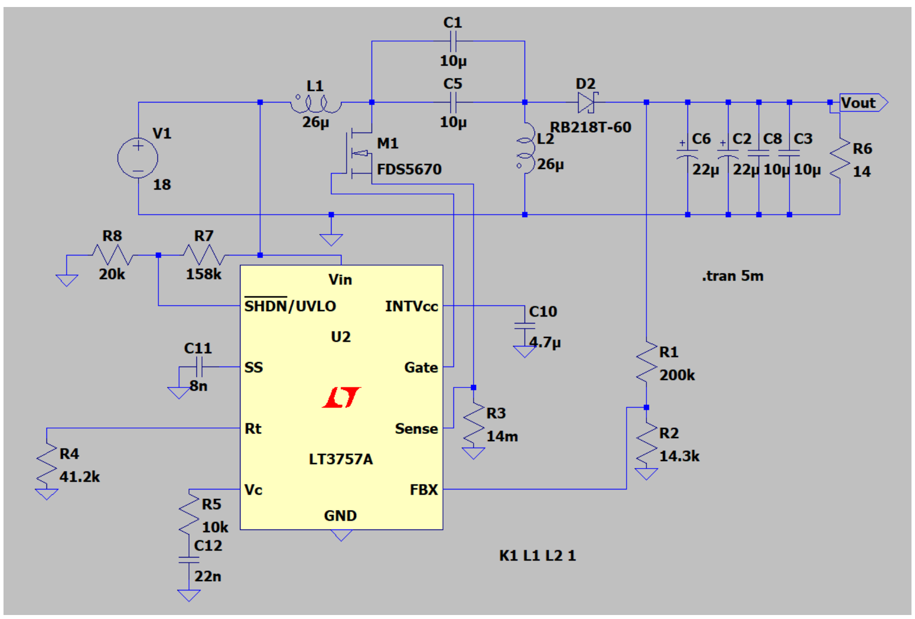

Figure 4: SEPIC simulation schematic

While the SEPIC converter regulates the DC solar output, AC/DC conversion is required to provide 24 VDC from the wall voltage of 120 VAC. We decided to purchase an AC/DC converter, as it was inexpensive and not the focus of the overall design. Thus, this component does not warrant much discussion other than the power rating far exceeding the need of our traffic light modules.

In order to power the MCU of our design, we selected an isolated 24 V to 5 V DC/DC converter. This component, like the AC/DC converter, is inexpensive and does not warrant extensive discussion. Isolation was desired to ensure that the power rails are unable to negatively impact the digital domain via switching noise.

The final power converter in the power subsystem requires some design, but is not as complex as the design of the SEPIC converter. Its purpose is to convert the 24 V level down to 5 V, which is not isolated and will provide power to the MOSFETs of the traffic light subsystem. The circuit diagram is provided by the controller's datasheet, which is located in Figure 5. Thus, no additional discussion is necessary. The PCB implementation of this converter is located in Appendix C.4.

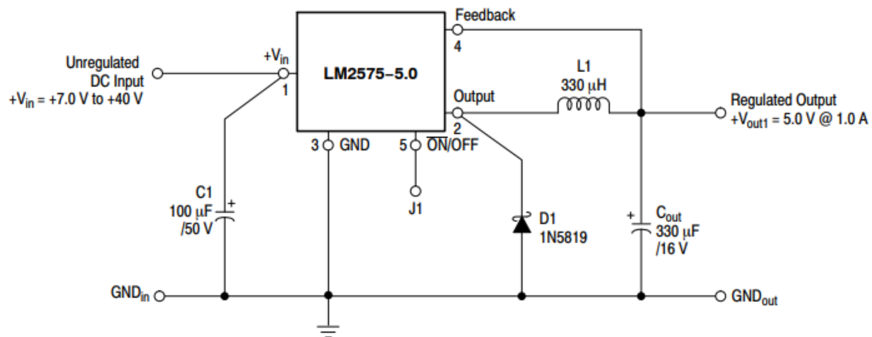

Figure 5: LM2575 step-down converter [4]

2.1.3 Switching Network

The switching network is a vital aspect of the overall design, as it must be able to autonomously switch between the solar and grid sources based on inputs from the MCU. Additionally, it must switch sufficiently fast such that there is no drop in voltage or interruption in operation while doing so. One significant detail is the need to provide reverse protection within the switching network. Without a method to impede the flow of current between the sources, it is possible for current to surge during a fault scenario. Upon considering these general requirements, it became clear that the optimal solution is to use four MOSFETs, with two in opposite orientation for each source. The intrinsic body diodes of each pair of MOSFETs will prevent current from flowing when their respective MOSFETs are off. In order to drive these switches, we utilize a quad high-side gate driver, which is required given that the sources of each MOSFET are not connected to ground. The inputs of the gate driver are connected to the MCU and are operated in complementary fashion, based on which source is chosen. On the output of the switching network we have bulk capacitance in order to ensure the voltage does not dip too much while we switch. Figure 6 shows these features in our final implementation.

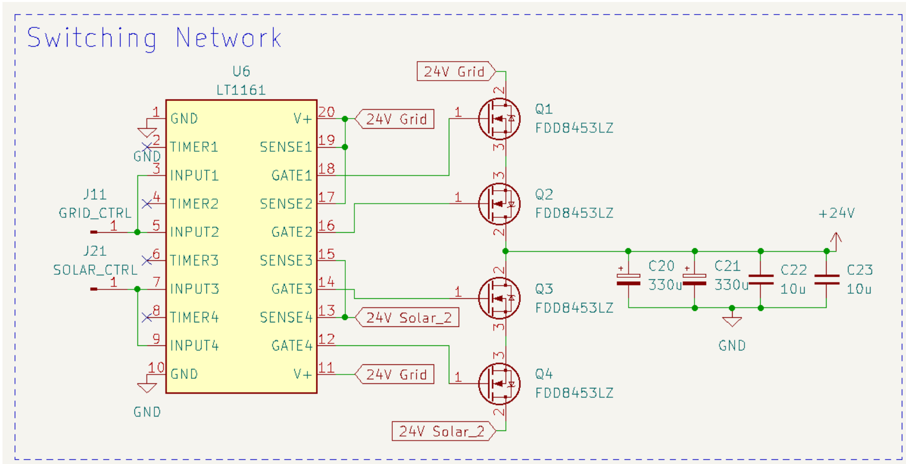

Figure 6: Switching network schematic

2.2 Sensing Subsystem

The sensing subsystem includes two buttons, a photoresistor, a humidity sensor, and an infrared sensor. The buttons determine if there are bikes or pedestrians waiting to cross the street. Realistically, the wires carrying the button signals would be 20+ feet long, so a 24 V signal is used to reduce noise interference from voltage drops and ensure signal integrity. An opto-isolator circuit is used to transmit the 24 V signal to the microcontroller. The photoresistor is responsible for detecting the presence of sunlight and sending a signal to the MCU. Its resistance can range from a few hundred ohms in a bright environment to over one mega ohm in complete darkness. A simple voltage divider circuit will be used to correlate the light level to voltage level. The humidity sensor is used to detect adverse weather conditions and turn off the PWM light dimming if it is on. This ensures proper visibility of the traffic lights. The infrared sensor is used to detect vehicles and send a signal to the microcontroller. Connections to each of these sensors are located on the MCU and power boards, which are available in Appendices B.1 and C.3, respectively.

2.3 Control Subsystem

The control subsystem receives data from our various sensors and changes the state and brightness of our lights and switches between our two input power sources.

2.3.1 Control Flow

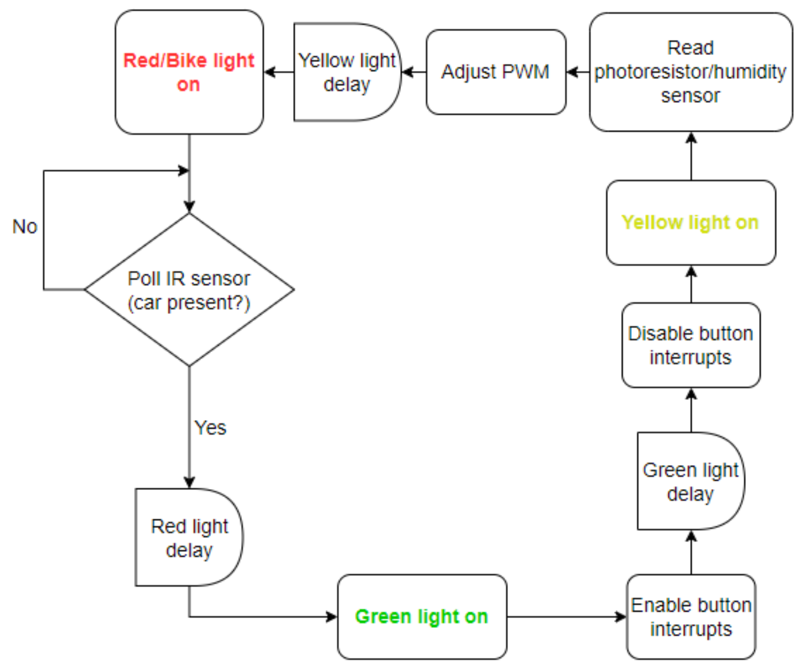

Figure 7: MCU control flow

Figure 7 shows a high level overview of the control sequence programmed onto our microcontroller. There are two additional things to note about control sequence. First, in any delay state when the microcontroller polls the photoresistor, it sends a signal to the switching network to switch between solar and grid power sources when a brightness threshold value is reached. Second, while in the green delay state, the light immediately switches to yellow if a biker/pedestrian interrupt is triggered by a button press. A full copy of our code is located in Appendix B.2.

2.3.2 Microcontroller

We chose to use the ATmega328P microcontroller for our project. This choice was made due to our familiarity with its operation and sufficient capabilities. We programmed it using the ArduinoISP and a separate Arduino Mega as the programmer. We modified some of the pin mappings from the original ArduinoISP code provided to configure it for programming the ATmega328P. Later in the testing process we also needed a workaround to receive serial data from the microcontroller, through our Arduino programmer, and back to our laptop's USB serial monitor. This was mainly used for debugging purposes when we were working with our sensors and tuning the threshold values. Overall, our microcontroller performed everything necessary without any major complications.

2.4 Traffic Light Subsystem

Our traffic lights are powered by 24 V and our bike light by 120 VAC. We sourced our traffic lights by contacting a traffic light manufacturing company called Leotek. They provided us with a sample of each colored light as well as a bike light. Since these lights are LEDs, we initially thought that we would need a current limiting resistor in order to protect them. However, when we took them apart, we discovered that there was existing internal circuitry. This meant that we could power the lights from 24 V directly. While connected to 24 V, the red, yellow, and green light modules consume 4.4, 4.5, and 4.6 W, respectively [5]. Additionally, the AC bike light consumes 6.0 W [6].

The traffic light subsystem dims the lights via PWM signals from the microcontroller. It also has a relay to allow the 120 VAC bike light to switch on and off. This system uses optoisolators in order to isolate the noisy power rails from the sensitive control signals. Figure 8 shows an example of the isolation circuit. Labels MCU_L1 and MCU_L2 are input signals to control the red and yellow lights, respectively. Labels L1 and L2 are connected to the gate of the MOSFETs that provide 24 V to the two lights. GND1 is the MCU's ground and GND is the noisy ground. In the case that our optoisolators do not work, we also have optional jumpers that directly connect the gate of the MOSFETs to the MCU's PWM pins. This circumvents the optoisolators but still allows us to control the lights. Originally, the design had our N-channel MOSFETs on the high side, but that caused switching problems as the gates were improperly biased. Moving the MOSFETs to the low side fixed the problem. Refer to Appendix D Figures 28 to 31 for the initial and final traffic light schematics and PCBs.

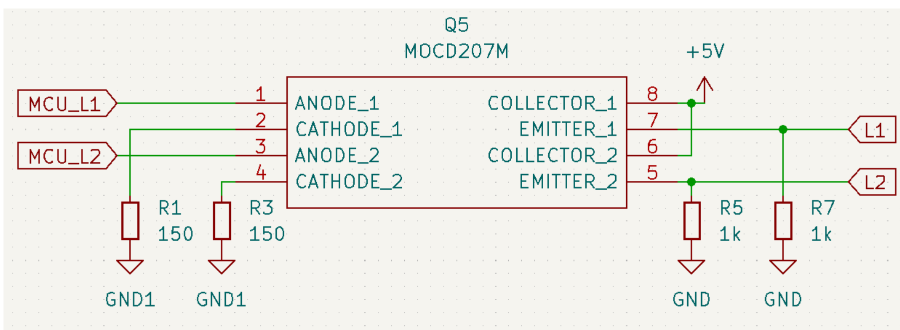

Figure 8: Traffic light optoisolator

3 Design Verification

In this section, we examine and discuss the tests performed on our various subsystems. We provide qualitative and quantitative data to verify the functionality of our project. See Appendix A for all of our requirements and verification tables.

3.1 Power Subsystem

3.1.1 Solar Panel

Table 4 in Appendix A states that the solar panel must provide at least 40 W and generate 18 V ± 5% when loaded. We were able to verify these requirements simply by measuring the output voltage with a multimeter on a bright and sunny day.

3.1.2 Power Converters

In order to test our power converters, we used a DC power supply and an electronic load. When testing our input voltage ranges, we adjusted the power supply output voltage. We set the electronic load current to test our converters' output current capabilities and output voltage drop. The SEPIC converter was able to maintain a specified output voltage of 24 V ± 5% when the input voltage was varied from 12 V to 32 V as shown in Figure 9. When holding the input voltage at 18 V and adjusting the load current from 0 A to 1.6 A, the output voltage still remained within bounds. Figure 10 displays the results of the test. However, when we tried to pull 1.7 A, the maximum current we specified, our converter started buzzing. We suspected that this was due to inductor core saturation. We did not want to risk damaging our components, so we did not gather data for that point.

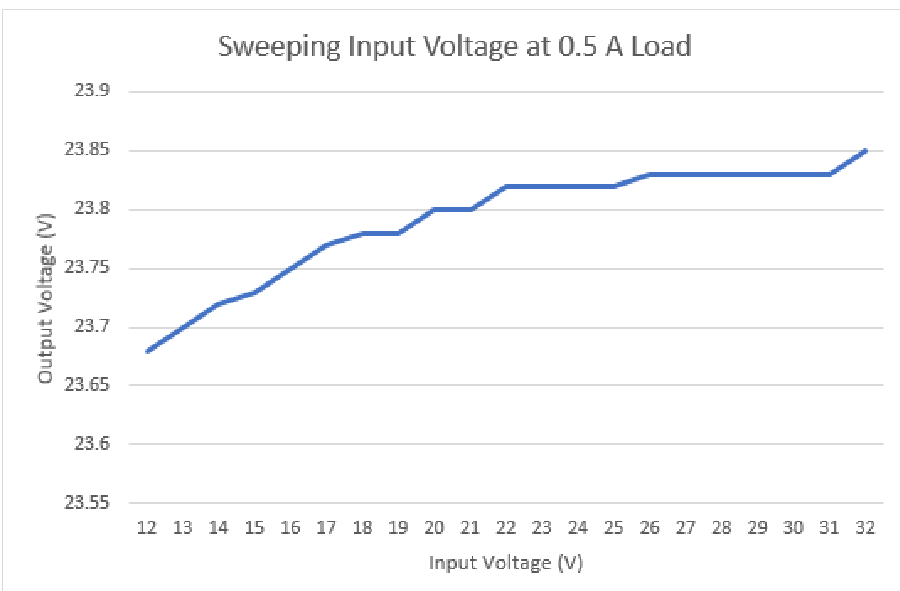

Figure 9: Sweeping SEPIC input voltage at 0.5 A

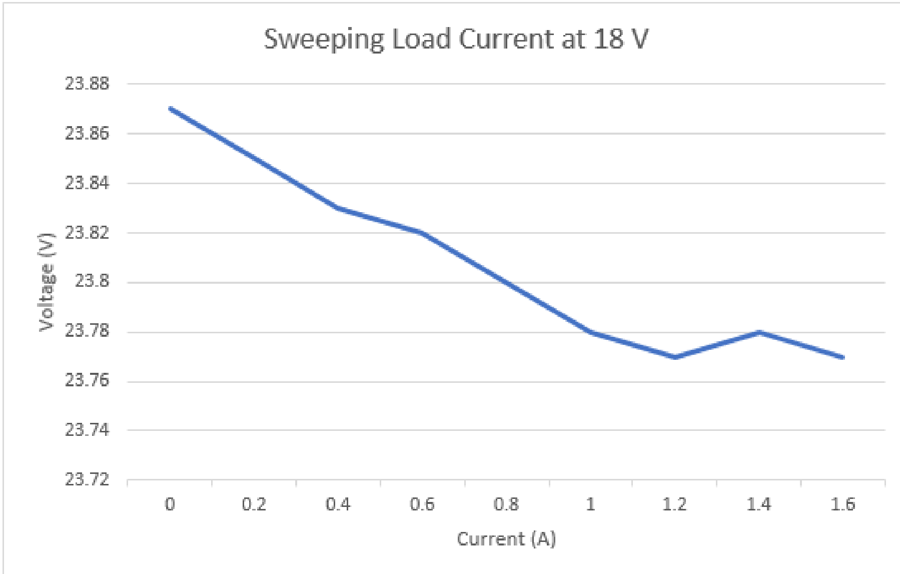

Figure 10: Sweeping SEPIC load current at 18 V

We bought our 120 VAC to 24 VDC converter rather than designing our own, since it was both cheaper and simpler. We were unable to obtain an AC power supply, so we could not test the converter by varying the input from 108 VAC to 132 VAC. Figure 11 shows the output voltage vs load current characteristic of the converter at a somewhat constant 118.7 VAC input. Once again, our converter was well within specifications.

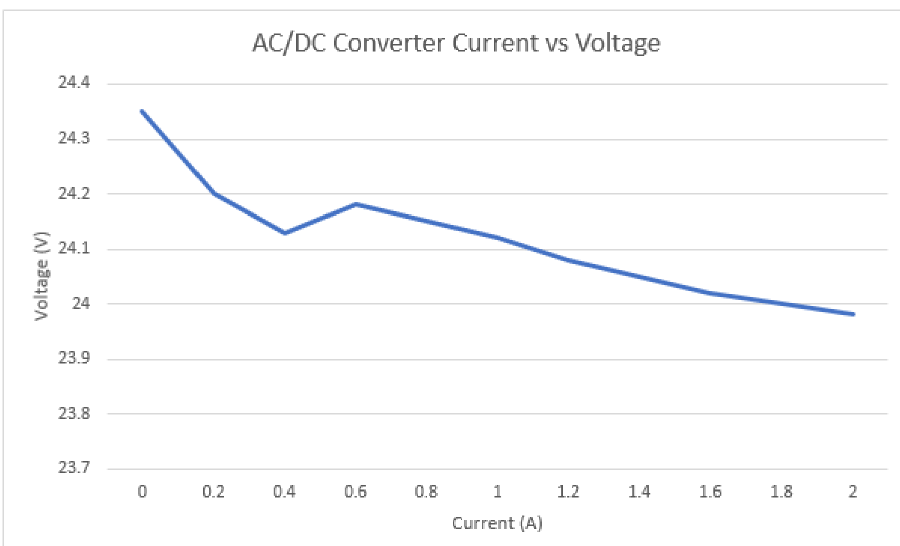

Figure 11: AC to DC converter characteristic

Finally, we have our 24 V to 5 V buck converter. This component worked perfectly as it maintained a correct range of output voltages based on varying input voltages and load current as shown in Figure 12

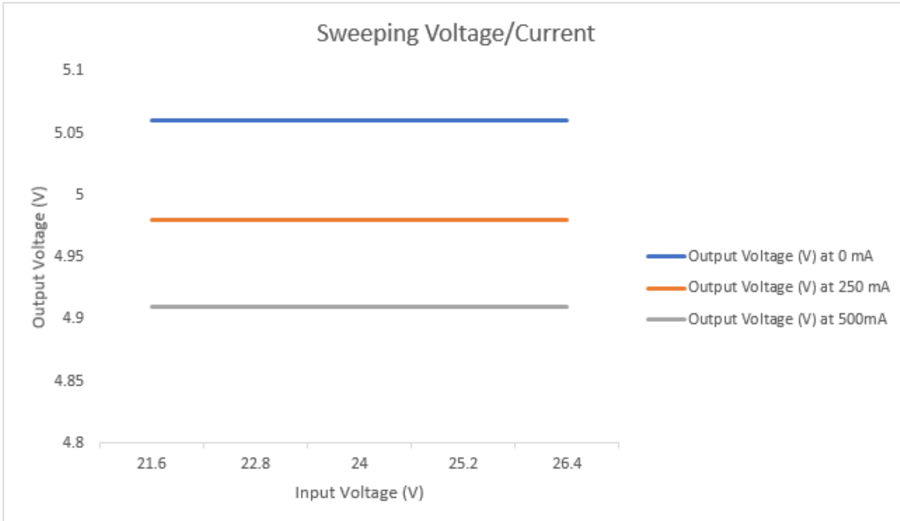

Figure 12: Sweeping 5 V buck voltage and current

3.1.3 Switching Network

As stated before, this part of our design must switch quickly in order to limit the output voltage drop. Prior to performing and verifying the requirements located in Appendix A, we performed intermediate tests by using two outputs of a waveform generator, two outputs of a bench-top power supply, and a multimeter. The waveform generator simulates two perfectly out-of-phase control signals with 50% duty cycle and frequency of 120 Hz. The power supply provides two 24 V signals, which simulate the solar and grid sources. The multimeter provides a method to measure the voltage at the output of the switching network. Upon connecting the system in this way, we realized that we had some significant issues with our design. After cutting a trace and soldering a jumper wire to correctly connect two pins of the gate driver (LT1161 pins 11 and 13 [7]), the output voltage on the multimeter no longer dipped below 24 V $\pm$ 5 %, thus confirming we had a functional switching network prior to performing our full verification. The results of this full verification can be seen in Figure 13, in which the 24 V output does not dip upon switching between sources.

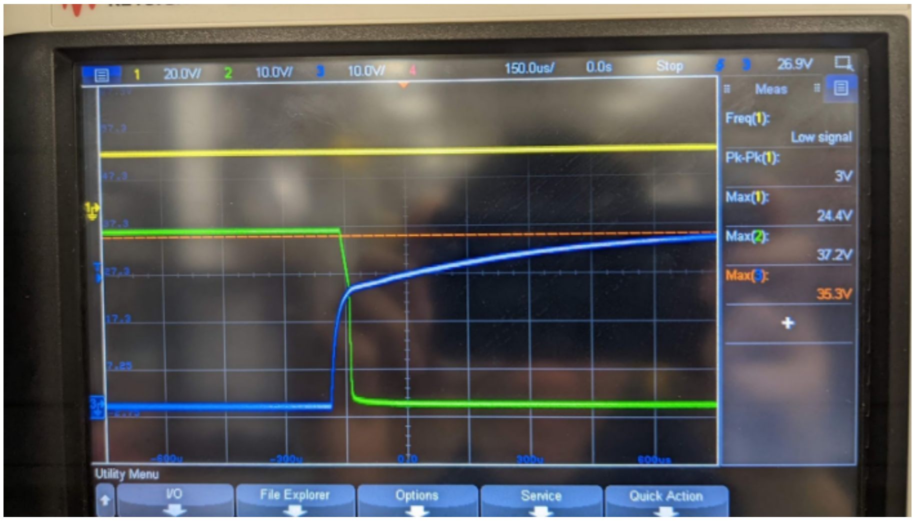

Figure 13: Switching network test - 24 V output (yellow), gate of grid control (green), and gate of solar control (blue)

3.2 Sensing Subsystem

Our photoresistor was able to measure a full range of values in dim and bright lighting conditions. This was verified while debugging and monitoring serial output data. Our humidity sensor, likewise, measured a full range of values in different operating conditions.

Through experimentation we found our infrared sensor did not measure distances beyond 8.2 ft accurately. The following figure shows the output voltage characteristics for the infrared sensor. We believe the inaccuracy in our sensor was due to the small voltage differential for distances beyond 8.2 ft, or approximately 250 cm. In hindsight, we should have tested the infrared sensor more rigorously with an oscilloscope to verify this.

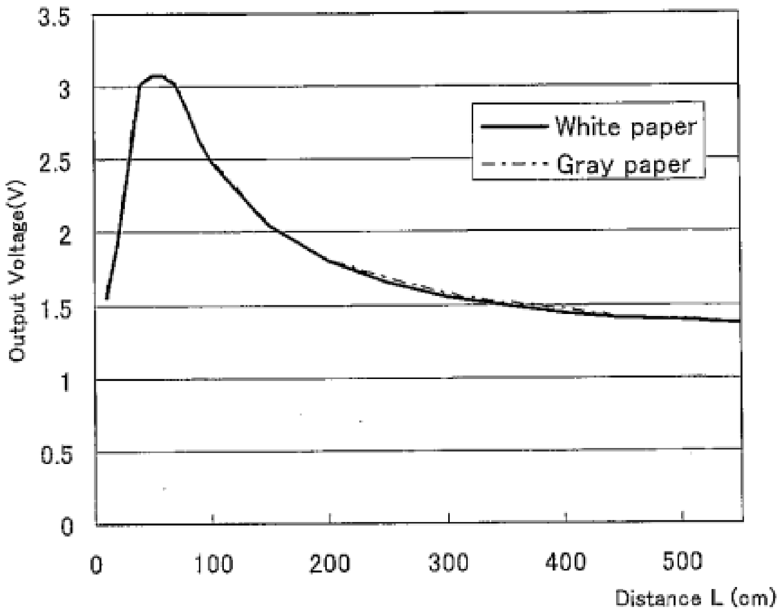

Figure 14: Output voltage characteristics for IR sensor [8]

Our power monitor did not work. We tried using an I2C scanner to discover the power monitor's address, but we were unable to receive an acknowledge signal from it. We further inspected the I2C signal through an oscilloscope to see if the optoisolators were transmitting the signals properly. Figure 15 shows the output signal directly from the Arduino Mega board. Figure 16 shows the signals that the power monitor received. We thought the slew rate of the optoisolators was the limiting factor, as the power monitor's datasheet specified a 1.5 kHz maximum data rate [9]. However, when we limited the I2C frequency to 500 Hz and 1000 Hz, the output signal remained similarly incorrect.

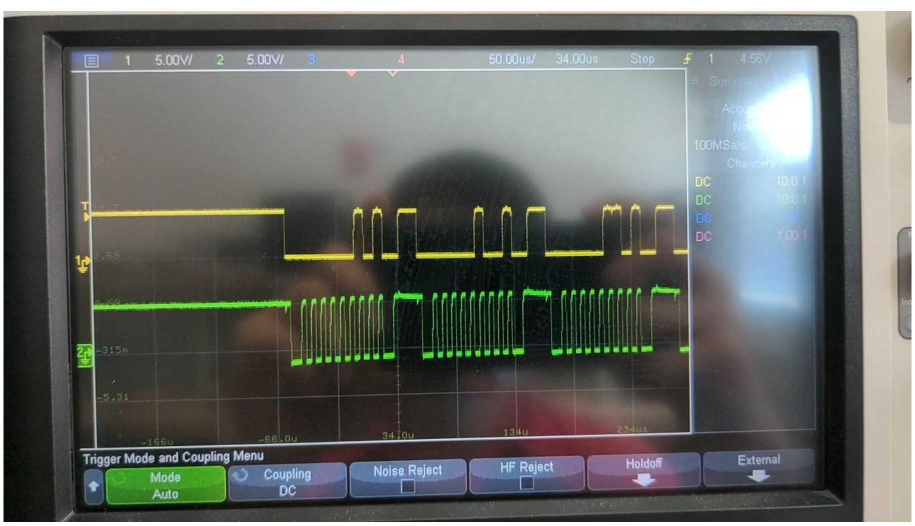

Figure 15: SCL (green) and SDA (yellow) from Arduino Mega

Figure 16: SCL (green) and SDA (yellow) from output of optoisolator

After this test, we tried to bypass the optoisolators and connect to the data and clock pins directly from the microcontroller. The power monitor we were using had a split data line: SDAI and SDAO inverted. We used a Schmitt trigger inverter to invert the inverted SDAO to, in theory, make it a regular SDAO line. However, when connecting SDAI and SDAO together, we were still unable to communicate with the IC. While we do not know what exactly caused our issues, we speculate that we could have overheated the IC while soldering it, the optoisolator slew rate was incompatible with our data transfer frequency, or the inverted SDAO line did not output a correct signal. We decided to go with our backup plan of using the photoresistor to measure brightness and switch sources based on that.

3.3 Control Subsystem

Our control system consisted of our MCU board, which is available in Appendix B.1. The board itself worked entirely as expected from our original design. In later stages, while programming the microcontroller and debugging, it was necessary to receive serial data and verify and tune all our sensors properly. We soldered wires onto the RX and TX pins to accomplish this.

3.4 Traffic Light Subsystem

Once the issues with the high side MOSFETs were fixed, the traffic light PCB worked as expected. Through visual inspection, we were able to verify the traffic lights did not flicker under normal operation or when switching sources. Figure 27 in Appendix D shows the traffic light at over 150 feet away. It was positioned by the ECE OpenLab and the picture was taken next to the window by the Senior Design Lab.

Figure 17 displays the AC current draw at two different PWM duty cycles. The average power drawn is around 16.7 W at 100% PWM and around 10.7 W at 20% PWM.

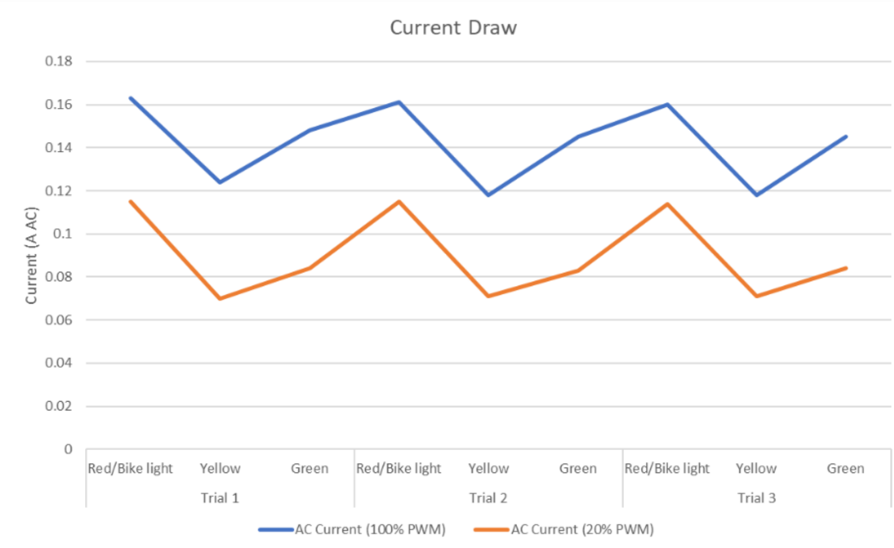

Figure 17: Current draw at 100% (blue) and 20% (orange) PWM duty cycles

4 Costs

4.1 Labor

Our team consists of two electrical engineering majors and one computer engineering major. From the 2019-2020 annual Illini Success Report, electrical engineers make an average of $76,129 and computer engineers make an average of $99,145 [10]. We worked approximately 10 hours per week this semester for 9 weeks. This will total 90 hours. We will also multiply by a 2.5x overhead cost. The cost of our labor is shown in Table 1. The machine shop worked on our project for approximately 20 hours. According to the UIUC machine shop website, the average cost for construction is $38.17 an hour plus materials [11]. Machine shop costs total $763.40. Parts cost sum to $123.44, as shown in Table 2.

The total cost of our project is $29,167.

Table 1: Labor costs

Name	Bowen Xiao	Richard Przybek	Colin Tarkowski
Rate	$38.06	$49.57	$38.06
Hours worked	90	90	90
Total Cost	$8,563.50	$11,153.25	$8,563.50

4.2 Parts

Following is a table of all the parts required for the project and their associated costs.

Table 2: Selected components and cost

Component	**Part #**	**Quantity**	**Unit Price**	**Total Price**
5 V Buck IC	LM2575D2T-5R4G	1	$2.84	$2.84
Schottky Diode	VS-30WQ04FNTR-M3	1	$0.68	$0.68
300 μH Inductor	HCTI-330-5.2	1	$2.99	$2.99
140 W Resistor	TEH140M33R0FE	1	$15.20	$15.20
AC/DC Converter	LM100-23B24	1	$17.44	$17.44
Power Monitor	LTC4151IMS-1	1	$8.40	$8.40
Optoisolator	MOCD207M	2	$1.30	$2.60
Sense Resistor	ERJ-3BWFR020V	1	$0.45	$0.45
Pushbutton	GPTS203211B	2	$1.71	$3.42
Photoresistor	161	1	$0.95	$0.95
Humidity Sensor	DHT20	1	$6.50	$6.50
IR Sensor	GP2Y0A710K0F	1	$21.21	$21.21
MCU	ATMEGA328-PU	1	$2.58	$2.58
Coupled Inductor	PF0553.153NLT	1	$2.54	$2.54
MOSFET	FDS5670	1	$1.82	$1.82
14 mΩ Sense Resistor	WSL2512R0140FTB	1	$2.21	$2.21
Schottky Diode	SBRT10U60D1-13	1	$0.97	$0.97
SEPIC Controller	LT3757AIMSE	1	$6.69	$6.69
MOSFET	FDD8453LZ	4	$1.51	$6.04
Quad High-Side Gate Driver	LT1161IN	1	$9.91	$9.91
Resistors and Capacitors	N/A	N/A	N/A	$8.00
			Total Cost	**$123.44**

5 Conclusion

5.1 Accomplishments

We successfully met all three of our high level requirements. The lights dim through PWM and use less power as a result. Using average values from Figure 17 and assuming a day consists of 12 hours of sunlight, 12 hours of darkness, and ideal conditions, our traffic light uses 385 Wh of grid power per day compared to an average traffic light, which uses 1.08 kWh of grid power per day [1]. Our design consumes approximately one-third of the average power. Our system is also able to switch between grid and solar power without causing a power failure. Finally, our buttons respond to biker and pedestrian input, and our traffic light timings can be adjusted to give them ample time to cross the road.

5.2 Uncertainties

While we deem our project a success and were able to verify the vast majority of our requirements, the most uncertain part of our project was the power monitor. Despite trying numerous ways to debug the component, we could not communicate via I2C and cannot pinpoint the issue with complete confidence. We were able to switch power sources based on the light detected by the photoresistor, but that is an inaccurate way of doing so. Having a power monitor would allow us to precisely sense when sufficient power is available from the solar panel. In this scenario, we could run on solar power for as long as possible and only switch to grid when we know the solar output will be insufficient.

5.3 Ethical Considerations

Our team did our best to adhere to the IEEE Code of Ethics. Since we designed a product for use in traffic, we ensured "the safety, health, and welfare of the public" and to "disclose promptly factors that might endanger the public or the environment" [12]. Our design choices are made clear through our design document and this final report.

Since our design is a scale model of a real traffic light system, we did not adhere to the building codes within the city of Champaign and our traffic light will not be put to use for real traffic. We should be able to meet relevant requirements, laid out in the city of Champaign Traffic Signal Standards, for our scale model. Although, any requirements with timing and light can be met as our traffic light is configurable and can be reprogrammed to set brightness levels and timing of the lights [13].

For the purposes of prototyping and due to our limited budget we were only able make a single traffic light. We did not concern ourselves with the control required for multiple switching traffic lights, as we are unable to demonstrate this functionality with only one light. For our design to be expanded safely to a four-way intersection, a more robust timing control system would need to be incorporated in the future. As previously mentioned, our traffic lights are configurable so the timing levels for bikers and pedestrians can be set in order to "avoid harm" as stated in 1.2 of ACM Code of Ethics [14]. Our biker and pedestrians buttons are also properly grounded and insulated to ensure users do not get shocked.

Visibility, vehicle detector position, and minimum green time are some of the parameters that we experimented with in this project and are configurable. We welcome any criticism about adhering to IEEE Code of Ethics I.5: "to seek, accept, and offer honest criticism of technical work..." [12].

5.4 Future Work

Our project ended up more limited in scope than we had hoped, due to financial and machine shop constraints. In terms of future work we can do three things. First, water proof the enclosure so it can operate in any weather conditions. Second, we can expand our system to four traffic lights and develop a more robust timing system. Finally, we can select a new power monitor in order to make our system more robust, predictable, and efficient in all conditions.

References

[1] City of Yakima, "Led Traffic Signals." [Online]. Available: https://www.yakimawa.gov/services/streets/led-traffic-signals

[2] Wikipedia, "Lane." [Online]. Available: https://en.wikipedia.org/wiki/Lane

[3] Texas Instruments Inc., "Benefits of a coupled-inductor SEPIC converter." [Online]. Available: https://www.ti.com/lit/an/slyt411/slyt411.pdf?ts=1651576998366

[4] Semiconductor Component Industries, "1.0 A, Adjustable Output Voltage, Step-Down Switching Regulator." [Online]. Available: https://www.mouser.com/datasheet/2/308/1/LM2575_D-2314720.pdf

[5] Leotek Electronics USA, "IL6 - P3 Series LED 8" and 12" Traffic Signal Module 10-28Vdc Incandescent Look Ball," Datasheet, 2019. [Online]. Available: https://leotek.com/wp-content/uploads/IL6-P3_8inch-and-12inch_10-28Vdc_Signal_Ball_Spec_Sheet_07-01-19.pdf

[6] ——, "LED 8" and 12" Signal Module IL6 Series Bicycle Indication," Datasheet, 2017. [Online]. Available: https://leotek.com/wp-content/uploads/il6-bicycle-signal-spec_sheet.pdf

[7] Linear Technology Corporation, "Quad Protected High-Side MOSFET Driver." [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/1161fa.pdf

[8] Sharp Corporation, "Gp2y0a710k0f." [Online]. Available: https://cdn-shop.adafruit.com/datasheets/gp2y0a710k.pdf

[9] Linear Technology, "High Voltage I2C Current and Voltage Monitor." [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/4151ff.pdf

[10] University of Illinois Urbana-Champaign, "Illini Success." [Online]. Available: https://uofi.app.box.com/s/1t8xj69117lrsqm7753ujnrg8yyrtcwn

[11] School of Chemical Sciences, "Machine Shop." [Online]. Available: https://scs.illinois.edu/resources/cores-scs-service-facilities/machine-shop

[12] IEEE, "IEEE Code of Ethics." [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[13] City of Champaign, "City of Champaign Manual of Practice." [Online]. Available: http://champaignil.gov/wp-content/uploads/2008/08/c14-chapter-14-traffic-signal-standards.pdf

[14] ACM, "ACM Code of Ethics and Professional Conduct." [Online]. Available: https://www.acm.org/code-of-ethics

[15] Jim Angel, "Statewide Records and Normals for Illinois." [Online]. Available: https://www.isws.illinois.edu/statecli/General/averages.htm

[16] Ohmite, "TEH140 Series." [Online]. Available: https://www.ohmite.com/assets/docs/res_teh140.pdf

[17] Unios Australia, "Driving the Flicker-free Effect." [Online]. Available: https://unios.com/wp-content/uploads/2019/02/UN_Driving-the-Flicker-Free-Effect_White-Paper-190205.pdf

Appendix A Requirement and Verification Tables

Table 3: General Requirements

Requirements	Verification	Verification Status
The system enclosure must be rainproof, up to about 3.5 inches monthly - the average precipitation based on the 1981 to 2010 averaging period [15].	1. We can simulate this by assuming our enclosure area to be 1ft x 3ft x 1ft and spraying approximately 2 gallons of water on it.	No (outside of machine shop's capabilities)

Table 4: Solar Panel Requirements

Requirements	Verification	Verification Status
The solar panel must provide a minimum of 40 W.	1. Use two multimeters and connect one in series with a power resistor and one in parallel across the power resistor to measure current and voltage, respectively. 2. Multiply current and voltage readings to get power output, which should be 40 W $\pm$ 20% on a cloudless and sunny day.	Yes
In full sunlight, the panel must generate 18 V$\pm$5% when loaded.	1. Program microcontroller with unit test that turns on two light modules at 100% duty cycle. 2. Use a multimeter to take a voltage reading across the solar panel terminals and verify it is within 17.1 V and 18.9 V.	Yes

Table 5: Power Converter Requirements

Requirements	Verification	Verification Status
The SEPIC converter at the solar output must provide 24 V ± 5% at up to 1.7 A output current for 12 V $\leq$ $V_{in} \leq$ 32 V.	1. Use an oscilloscope and a multimeter to check that the output is 24 V ± 5% and up to 1.7 A, respectively. Sweep the input voltage between 12 V and 32 V with the DC power supply at the work station.	Yes
The AC-DC converter at the grid output must convert 120 V ± 10% to 24 V ± 5% at up to 2 A output current.	1. Use power supply to vary the input from 108 VAC to 132 VAC and use a multimeter to ensure output remains within desired bounds.	Yes
The 24 V to 5 V converter must provide 5 V ± 5% at up to 500 mA output current.	1. Use DC power supply to vary the input from 21.6 V to 26.4 V and use a multimeter to ensure output remains within desired bounds.	Yes
The 24 V to 3.3 V converter must provide 3.3 V ± 5% at up to 500 mA output current.	1. Use DC power supply to vary the input from 21.6 V to 26.4 V and use a multimeter to ensure output remains within desired bounds	Yes

Table 6: Switching Network Requirements

Requirements	**Verification**	**Verification Status**
Switching between the two sources must not interrupt normal operations.	1. Program the microcontroller to switch between power sources every 10 seconds and a counter. 2. Use an oscilloscope to verify that the 24 V, 5 V, and 3.3 V rails remain within tolerance levels. 3. Visually monitor lights and ensure no flickering occurs. 4. Check to make sure counter does not reset after any switches. If it does, the microcontroller temporarily lost power due to the switching.	Yes
Switching transients must not exceed 2.4 V.	1. Measure output voltage using an oscilloscope to ensure that it stays within 10% of 24 V every time the input switches.	Yes
Switching time must not exceed 20 ms.	1. Connect an oscilloscope to the gates of each pair of MOSFETs in the switching network to ensure the one pair turns on within 20 ms of the other pair turning off.	Yes

Table 7: Sensing Requirements

Requirements	Verification	Verification Status
1. The power monitor must be accurate to ±5% of the actual wattage. 2. The power resistor must not exceed the rated operating temperature of 25°C (at 140 W) while dissipating power [16].	1. Use two multimeters and connect one in series to the power resistor and one in parallel across the power resistor to measure current and voltage, respectively. Multiply values to compute power and ensure the power monitor reading is within 5%. 2. Use a laser thermometer to check temperature of the resistor. If the temperature is above 25°C under full load, attach a passive heat sink to prevent overheating.	No (we were unable to communicate with the power monitor via i2c)
1. The buttons must not shock anyone who touches it. 2. The signal must be transmitted across the opto-isolator within 1 ms.	1. Use an multimeter to ensure the enclosures and buttons themselves are properly grounded. Visually inspect all wires for proper insulation and exposed copper. 2. Use an oscilloscope and connect a probe to the button and another to the opto-isolator output pin. The time between the button press and opto-isolator output time must be below 1 ms.	Yes
The photoresistor must be able to differentiate between various light intensities.	1. Measure output voltage when the system is exposed to the sun, complete darkness, and a dimly lit room.	Yes

The IR sensor must be able to detect the presence of a vehicle from a distance of 10 ft to 18 ft.	1. Position the sensor 10 ft and 18 ft (maximum range) away from a vehicle and measure corresponding output voltages. 2. Compare the voltage levels and establish a lower and upper boundary. 3. Program the microcontroller with a test program to indicate if the voltage is within the boundary (i.e. car detected).	No (Our IR sensor is accurate within the range 1/2 - 8.2 ft)

Table 8: Control System Requirements

Requirements	Verification	Verification Status
Microcontroller must respond to button interrupts within 5 ms. Note that this requirement is different than the signal reaching the microcontroller within 1 ms.	1. Program a unit test that toggles a GPIO pin when a button interrupt is received. The traffic light FSM must be running when pressing button. 2. Use an oscilloscope and connect a probe to the button and another to the microcontroller GPIO pin. The time between the button press and microcontroller GPIO toggle time must be below 5 ms.	Yes
Microcontroller takes sensor feedback and responds with appropriate control signals (PWM and switching power sources).	1. Perform experiment in a room with dimmable lights. Program microcontroller to change PWM duty cycle linearly based on light intensity. Use an oscilloscope on the PWM pin and dim the lights. The duty cycle should decrease from what it was initially 2. Hook up an oscilloscope to the gate of one MOSFET from each pair of MOSFETs in the switching network. Ensure that the solar panel powers the system (power output above 20 W). The MOSFET gate of the solar power portion should be high and of the grid portion should be low. Cover the solar panel with a cloth such that the power output drops to below 20 W and verify that the MOSFET gate of the solar portion is low and the grid portion is high.	Yes

Table 9: Traffic Light Requirements

Requirements	**Verification**	**Verification Status**
1. The LEDs must be visible from 150 ft for drivers to see them in bright conditions. Most modern traffic lights are 400-1000 lumens so we aim to be in this range [**?**]. 2. The LEDs should have multiple dimming levels (i.e. 40%, 60%, 80%, and 100% of full brightness), achieved by changing the PWM duty cycle to corresponding percentages. 3. The PWM frequency must be above 80 Hz, the maximum flicker frequency that is visible to the human eye [17].	1. Stand 150 ft away and see if the lights are visible. Use a light meter to measure lumen output and ensure it is within 400-1000 lumens. 2. Use an oscilloscope to ensure the PWM duty cycle is within 5% of expected percentage. Use light meter to correlate 100% duty cycle to maximum brightness. Measure lumen output with a light meter at multiple levels and ensure they are within 5% of expected brightness level. 3. Use an oscilloscope to verify that the PWM circuit oscillates at a minimum of 80 Hz.	Yes

Appendix B Control Subsystem

B.1 MCU Board Design

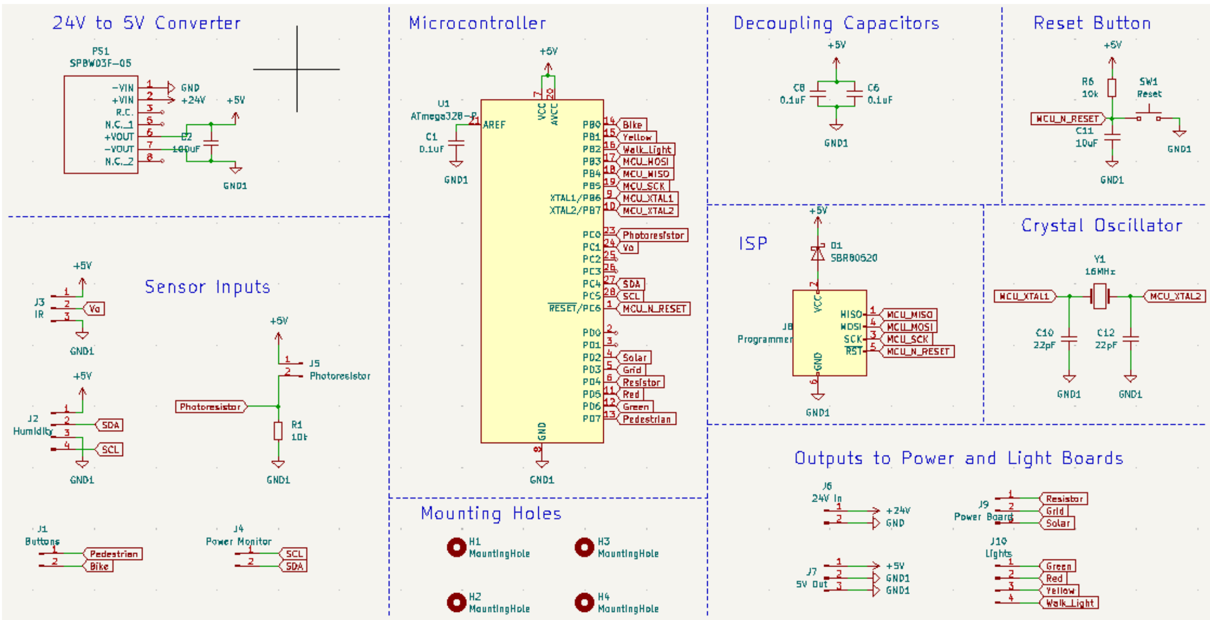

Figure 18: MCU board schematic

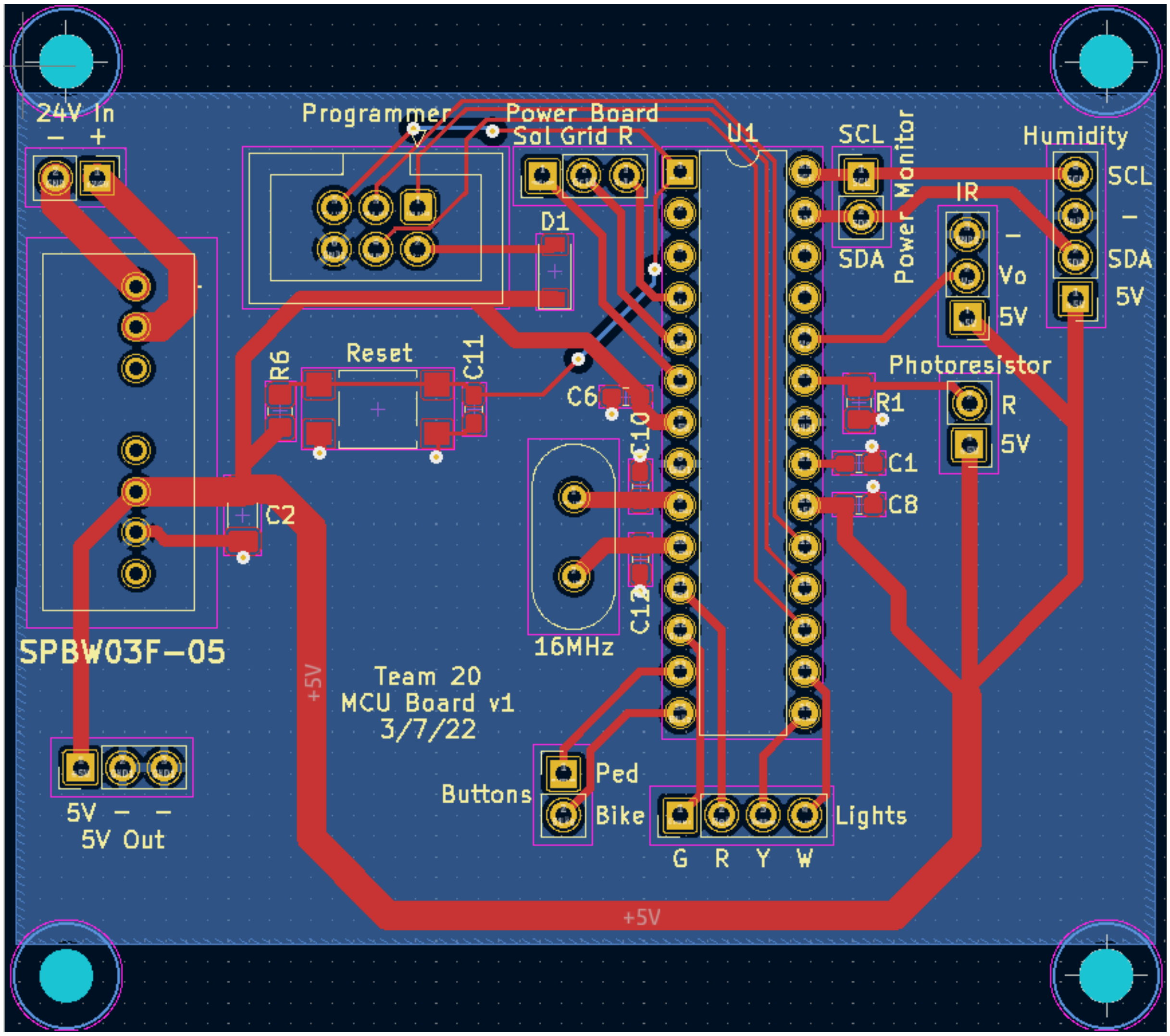

Figure 19: MCU board layout

B.2 Microcontroller Code

```
#include <SharpIR.h>
#include <DHT20.h>
#include <util/delay.h>

#define PHOTO_PIN A0
#define IR_PIN A1
#define SOLAR_PIN 2
#define GRID_PIN 3
#define RED_LED_PIN 5
#define YELLOW_LED_PIN 9
#define GREEN_LED_PIN 6
#define WALK_LED_PIN 10
#define BIKE_INT_PIN 8
#define IR_SENSOR_MODEL 100500

const int ir_distance = 200;
int light_val;
int solar_val;
int brightness = 255;
unsigned long time_now;
int ir_val;
float humidity_val;
DHT20 dht;
SharpIR irSensor = SharpIR(IR_PIN,IR_SENSOR_MODEL);
volatile bool gotInterrupt = false;

const unsigned long red_delay = 10000;   //90000, 1:30 min
const unsigned long green_delay = 7000;
const unsigned long green_default_delay = 3000;
const unsigned long yellow_delay = 3000;

ISR (PCINT0_vect)
{
  gotInterrupt = true;
}

void RedLight(){
  analogWrite(RED_LED_PIN, brightness);
  ir_val = irSensor.distance();

  time_now = millis();
  bool bike_light_flag = true;
```

```
  while(ir_val > ir_distance){
    ir_val = irSensor.distance();
//    String ir_str = "~IR Distance: ";
//    Serial.println(ir_str+ir_val+" cm");
    if(millis()-time_now > red_delay/10 && bike_light_flag){
      analogWrite(WALK_LED_PIN, 255);
      bike_light_flag = false;
    }

    solar_val = analogRead(PHOTO_PIN);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
    else{
      // Switch to grid
      digitalWrite(SOLAR_PIN, 0);
      digitalWrite(GRID_PIN, HIGH);
    }
  }

  int bike_flicker = 8;
  unsigned long int time_now2;
  bool flicker_flag = true;
  while(millis() - time_now < red_delay){
    if(millis()-time_now > red_delay/10 && bike_light_flag){
      analogWrite(WALK_LED_PIN, 255);
      bike_light_flag = false;
    }
    solar_val = analogRead(PHOTO_PIN);
//    String solar_str = "~Solar: ";
//    Serial.println(solar_str + solar_val);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
    else{
      // Switch to grid
      digitalWrite(SOLAR_PIN, 0);
      digitalWrite(GRID_PIN, HIGH);
    }
```

```
  if(millis()-time_now > red_delay*2/3 && flicker_flag){
    time_now2 = millis();
    flicker_flag = false;
  }
  if(millis()-time_now > red_delay*2/3){
    if(millis()-time_now2 > (red_delay/3)/8){
      bike_flicker--;
      time_now2 = millis();
    }
    if(bike_flicker%2 == 0 || bike_flicker == 1){
      analogWrite(WALK_LED_PIN, 0);
    }
    else{
      analogWrite(WALK_LED_PIN, 255);
    }
  }
}

if(flicker_flag){
  time_now = millis();
  time_now2 = time_now;
  flicker_flag = false;
  while(millis() - time_now < red_delay/3){
    if(millis()-time_now2 > (red_delay/3)/8){
      bike_flicker--;
      time_now2 = millis();
    }
    if(bike_flicker%2 == 0 || bike_flicker == 1){
      analogWrite(WALK_LED_PIN, 0);
    }
    else{
      analogWrite(WALK_LED_PIN, 255);
    }

    solar_val = analogRead(PHOTO_PIN);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
    else{
      // Switch to grid
```

```
        digitalWrite(SOLAR_PIN, 0);
        digitalWrite(GRID_PIN, HIGH);
      }
    }
  }
  analogWrite(RED_LED_PIN, 0);
  analogWrite(WALK_LED_PIN, 0);
}

void YellowLight(){
  analogWrite(YELLOW_LED_PIN, brightness);

  //READ PHOTORESISTOR
  light_val = map(analogRead(PHOTO_PIN), 0, 1023, 0, 255);

//--------------testing photoresistor-------------------
//  if(light_val > 130){
//    analogWrite(10, 255);
//  }
//  else{
//    analogWrite(10, 0);
//  }
//----------------------------------------------------------

  //Take humidity reading
  int humidity_sensor_status = dht.read();
  switch (humidity_sensor_status)
  {
    case DHT20_OK:
      humidity_val = dht.getHumidity();
      break;
    default:
      humidity_val = 100;
      break;
  }

//------------------testing humidity sensor-------------
//  if(humidity_val > 60){
//    analogWrite(10, 255);
//  }
//  else{
//    analogWrite(10, 0);
//  }
```

```
//  delay(500);
//-------------------------------------------------------

  brightness = min(light_val+20, 255);
  if(humidity_val > 80) brightness = 255;

//  String bright = "~Brightness: ";
//  Serial.println(bright+brightness);
//  String humidity_str = "~Humidity: ";
//  Serial.println(humidity_str+humidity_val);

  time_now = millis();
  while(millis() - time_now < yellow_delay){
    solar_val = analogRead(PHOTO_PIN);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
    else{
      // Switch to grid
      digitalWrite(SOLAR_PIN, 0);
      digitalWrite(GRID_PIN, HIGH);
    }
  }
  analogWrite(YELLOW_LED_PIN, 0);
}


void GreenLight(){
  PCMSK0 |= B00000001; //mask interrupts on pin 14: PCINT0

  analogWrite(GREEN_LED_PIN, brightness);

  //Default green delay
  time_now = millis();
  while(millis() - time_now < green_default_delay){
    solar_val = analogRead(PHOTO_PIN);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
```

```
    else{
      // Switch to grid
      digitalWrite(SOLAR_PIN, 0);
      digitalWrite(GRID_PIN, HIGH);
    }
  }

  time_now = millis();
  while(millis() - time_now < green_delay && !gotInterrupt){
    solar_val = analogRead(PHOTO_PIN);
    if(solar_val > 950){
      // Switch to solar
      digitalWrite(SOLAR_PIN, HIGH);
      digitalWrite(GRID_PIN, 0);
    }
    else{
      // Switch to grid
      digitalWrite(SOLAR_PIN, 0);
      digitalWrite(GRID_PIN, HIGH);
    }
  }

  analogWrite(GREEN_LED_PIN, 0);

  PCMSK0 &= B00000000; //unmask interrupts
}

void setup() {
  Serial.begin(19200);
  dht.begin();

  pinMode(WALK_LED_PIN, OUTPUT);
  pinMode(RED_LED_PIN, OUTPUT);
  pinMode(YELLOW_LED_PIN, OUTPUT);
  pinMode(GREEN_LED_PIN, OUTPUT);
  pinMode(SOLAR_PIN, OUTPUT);
  pinMode(GRID_PIN, OUTPUT);

  analogWrite(WALK_LED_PIN, 0);
  analogWrite(RED_LED_PIN, 0);
  analogWrite(YELLOW_LED_PIN, 0);
  analogWrite(GREEN_LED_PIN, 0);
  digitalWrite(SOLAR_PIN, 0);
```

```
  digitalWrite(GRID_PIN, HIGH);

  light_val = map(analogRead(PHOTO_PIN), 0, 1023, 0, 255);

  PCICR |= B00000001; //Enable interrupts on port B (PCINT0-PCINT7)
  PCMSK0 &= B00000000; //unmask interrupts on all pins
}

void loop() {
  gotInterrupt = false;

  RedLight();
  GreenLight();
  YellowLight();

}
```

Appendix C Power Subsystem

C.1 SEPIC Design Calculations

$12 \le V_{in} \le 25$ V $\quad V_{out} = 24$ V $\quad I_{out} \le 1.7$ A

$D_{max} = \frac{24+0.7}{12+24+0.7} = 0.673$

$I_{L_1,max} = I_{in,max} = I_{out,max}\frac{D_{max}}{1-D_{max}} = 1.7\left(\frac{0.673}{1-0.673}\right) = 3.5$ A

$I_{L_2,max} = I_{out,max} = 1.7$ A

$R_T = 41.2\,k\Omega$

$I_{sw,max} = I_{L_1,max} + I_{L_2,max} = 3.5+1.7 = 5.2$ A

Let $x = 0.2 \Rightarrow 20\%$ ripple current

$I_{sw,peak} = \left(1+\frac{x}{2}\right)I_{sw,max} = 5.72$ A

$\Delta I_{sw} = x I_{sw,max} = 1.04$ A

$\Delta I_{L_1} = \Delta I_{L_2} = 0.5\Delta I_{sw} = 0.52$ A

$L_1 = L_2 = \frac{V_{in,min}}{0.5\Delta I_{sw} f} D_{max} = \frac{12}{0.5(1.04)(300k)}(0.673) = 51.769\ \mu H$

$\hookrightarrow$ or if coupled: $L = \frac{V_{in,min}}{\Delta I_{sw} f} D_{max} = \frac{12}{1.04(300k)}(0.673) = 25.885\ \mu H$

$I_{L_1,peak} = I_{L_1,max} + 0.5\Delta I_{L_1} = 3.76$ A $\quad I_{L_2,peak} = I_{L_2,max} + 0.5\Delta I_{L_2} = 1.96$

$R_{sense} = \frac{80\,mV}{I_{sw,pk}} = \frac{0.08}{5.72} = 0.013986 = 13.99\ m\Omega \approx 14\ m\Omega$

$V_{sw,rated} = 24+25+10 \approx 60$ V $\quad I_{sw,rated} \geq 1.2(5.2) \approx 7$ A

Minimize $R_{DS,on}$ C_{RSS}

$I_{D_{peak}} = \left(1+\frac{0.2}{2}\right)I_{sw,max} = 5.72$ A $\quad V_{D,rated} \geq 24+25+10 = 60$ V

$V_{CDC} > V_{in,max} = 25$ V $\quad I_{rms,CDC} > I_{out,max}\sqrt{\frac{V_{out}+V_D}{V_{in,min}}} = 1.7\sqrt{\frac{24.7}{12}} = 2.439$ A

$ESR_{Cout} \le \frac{0.01 V_{out}}{I_{D,pk}} = \frac{0.01(24)}{5.72} = 0.04196 = 41.958\ m\Omega$

$C_{out} \geq \frac{I_{out,max}}{0.01 V_{out} f} = 23.61\ \mu F$ $\quad I_{rms,Cout} \geq I_{out,max}\sqrt{\frac{D_{max}}{1-D_{max}}} = 2.439$ A

Figure 20: SEPIC calculations

C.2 Initial Power Board Design

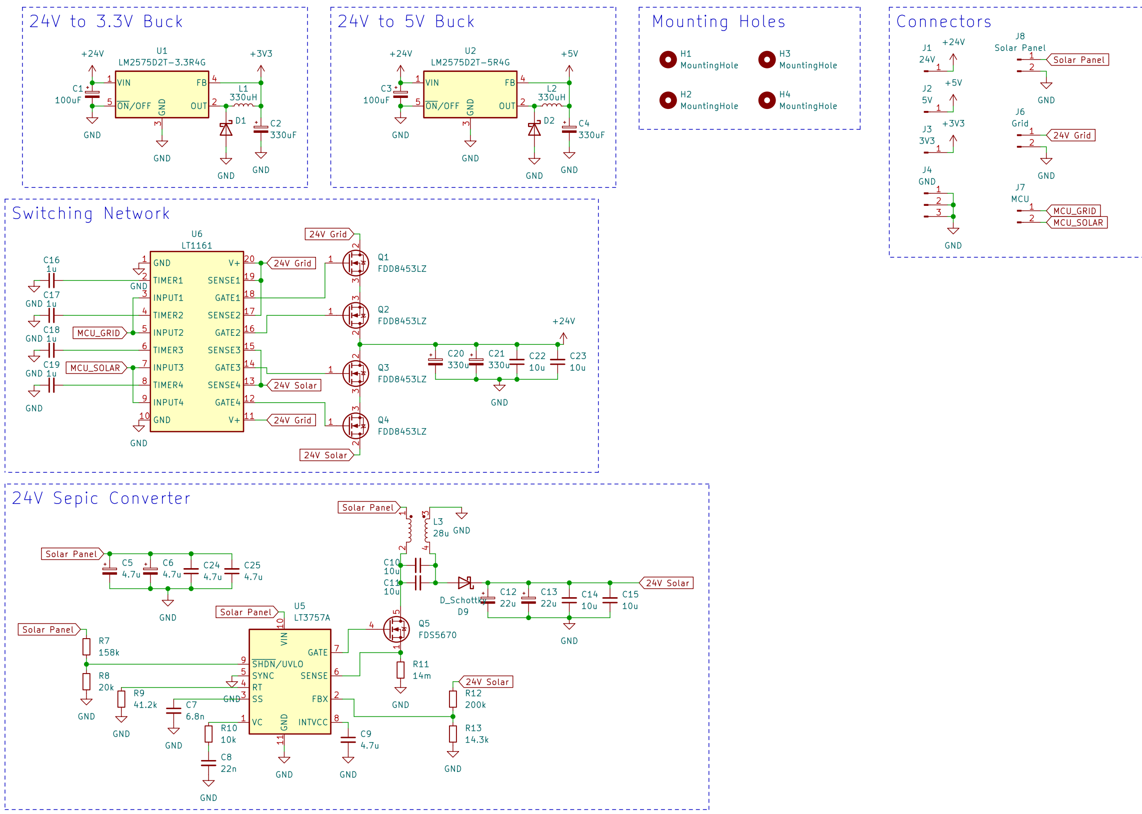

Figure 21: Initial power board schematic

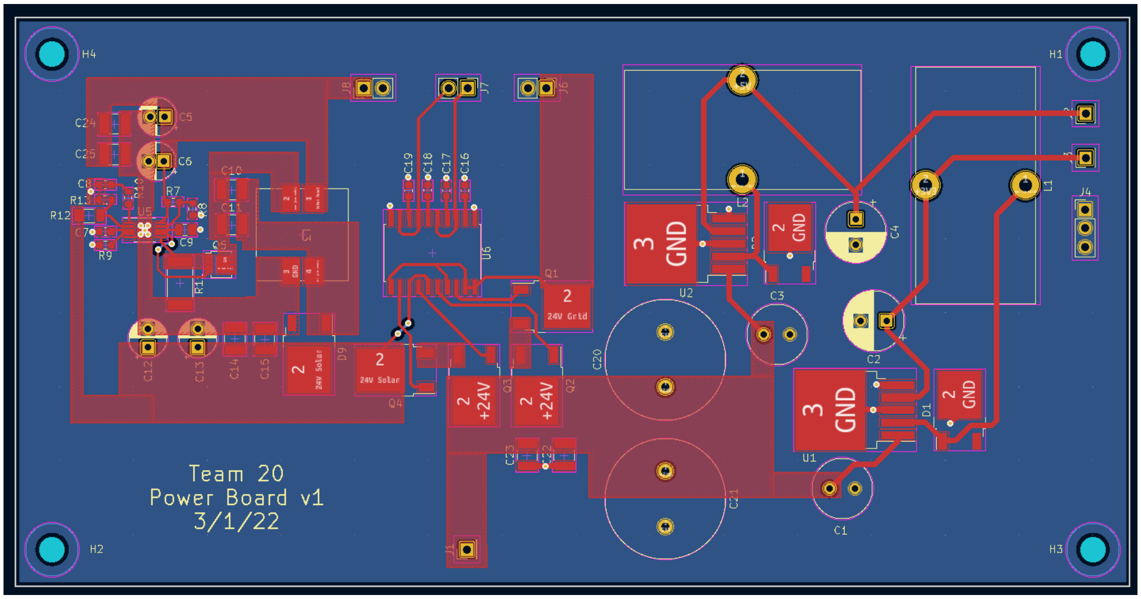

Figure 22: Initial power board layout

C.3 Final Power Board Design

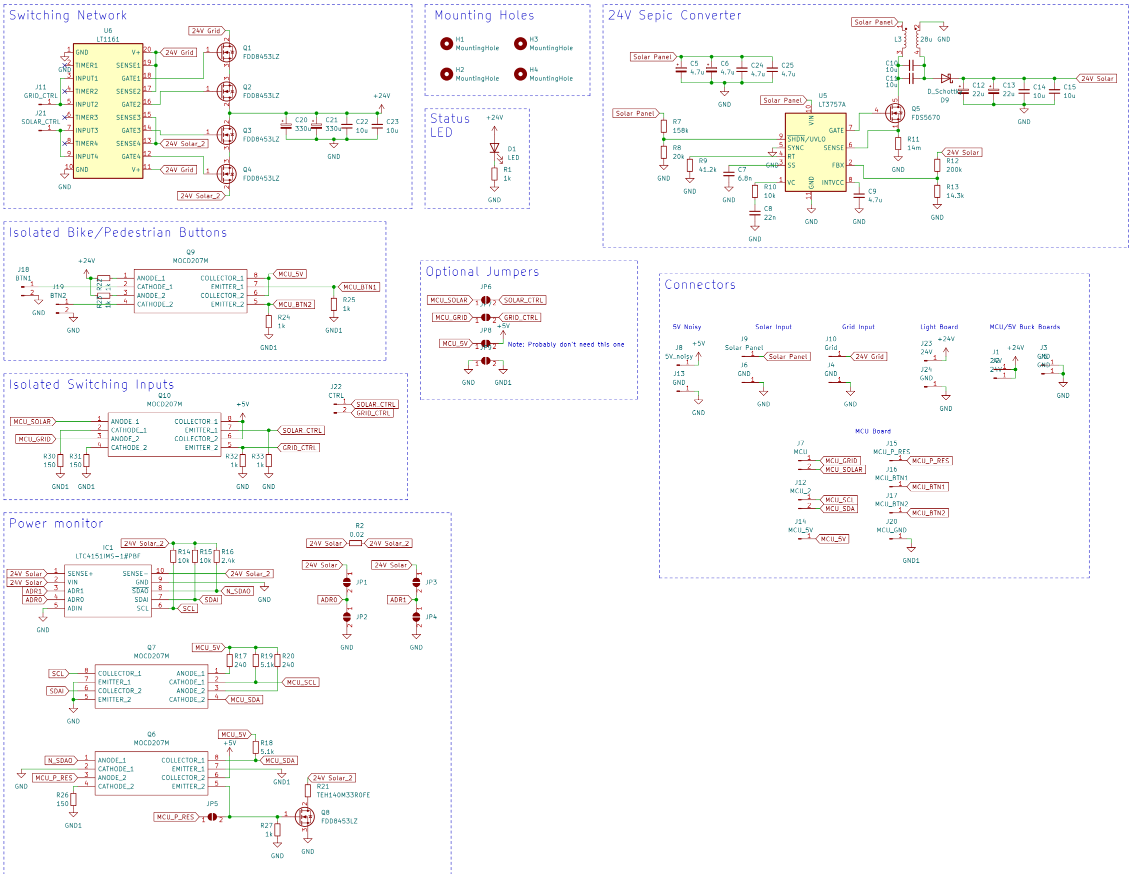

Figure 23: Final power board schematic

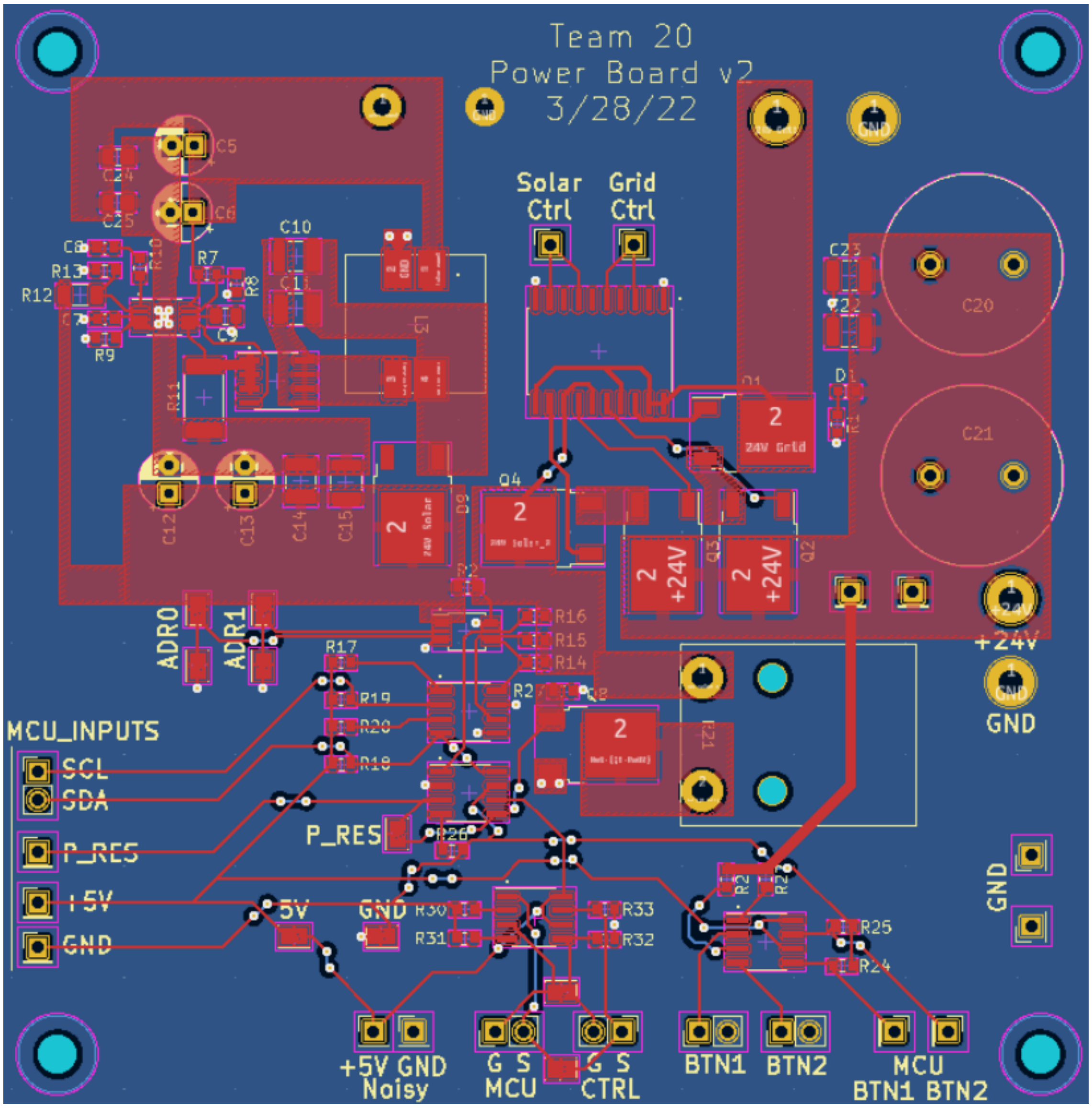

Figure 24: Final power board layout

C.4 Buck Board Design

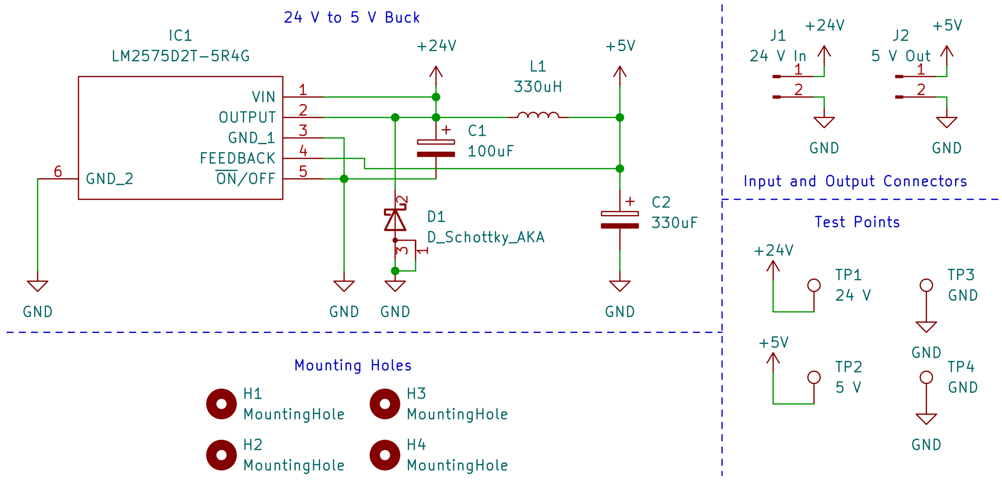

Figure 25: Buck board schematic

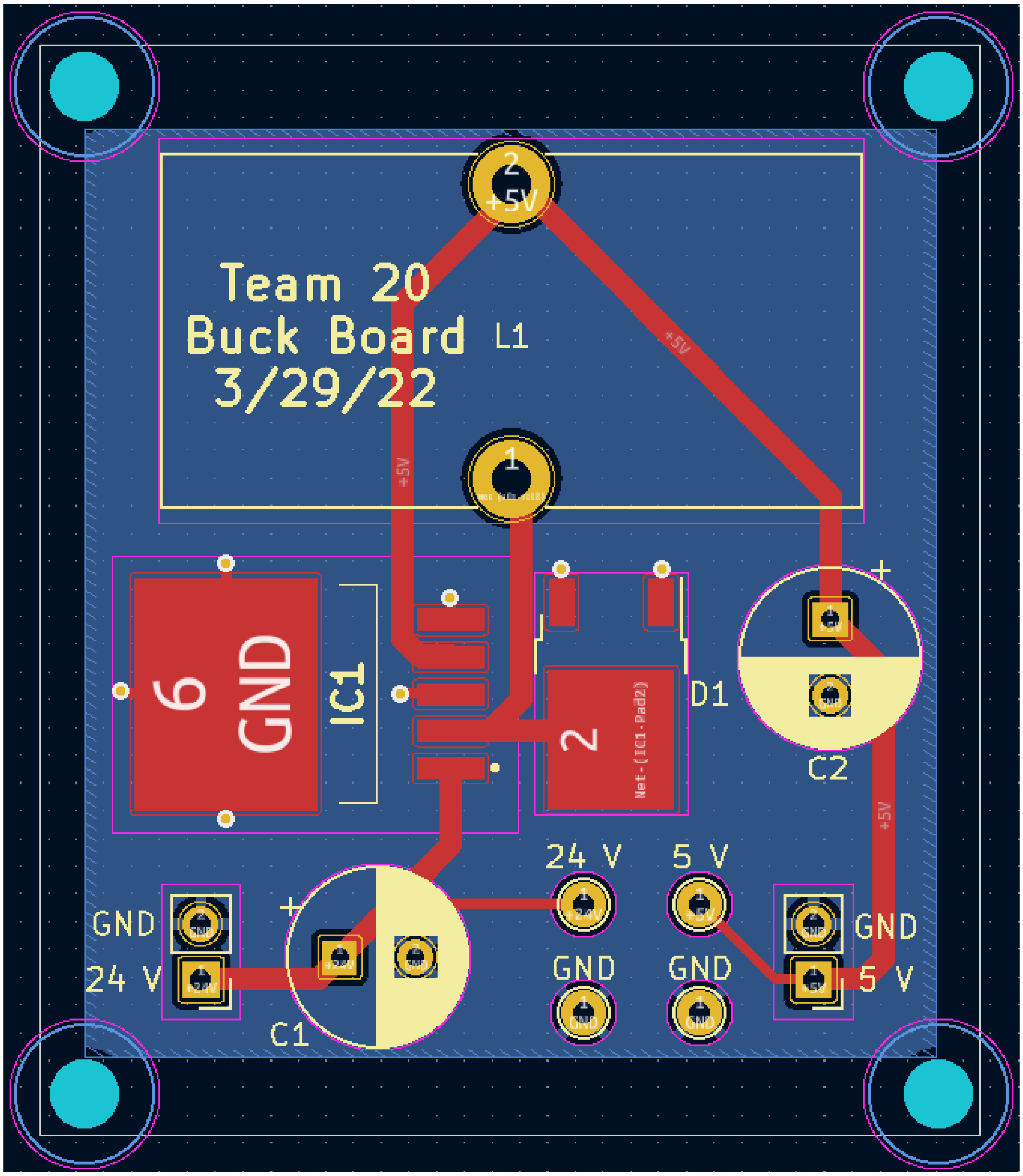

Figure 26: Buck board layout

Appendix D Traffic Light Subsystem

D.1 Traffic Light at 150+ ft

Figure 27: Traffic light from 150+ ft away

D.2 Initial Traffic Light Board Design

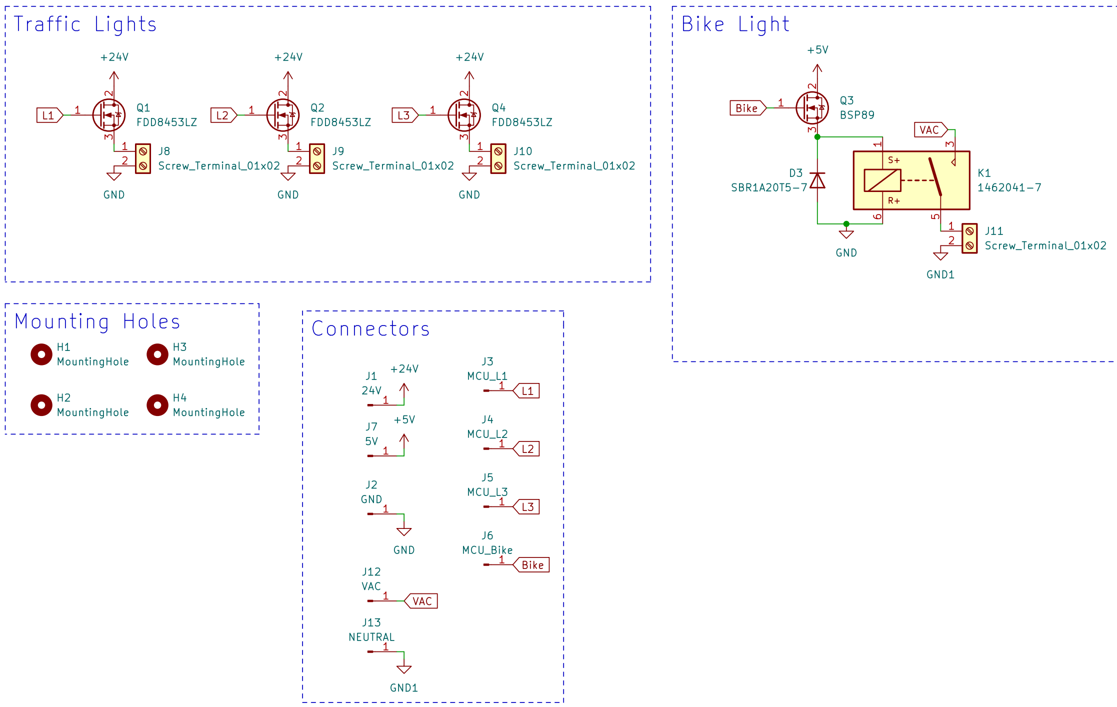

Figure 28: Initial traffic light board schematic

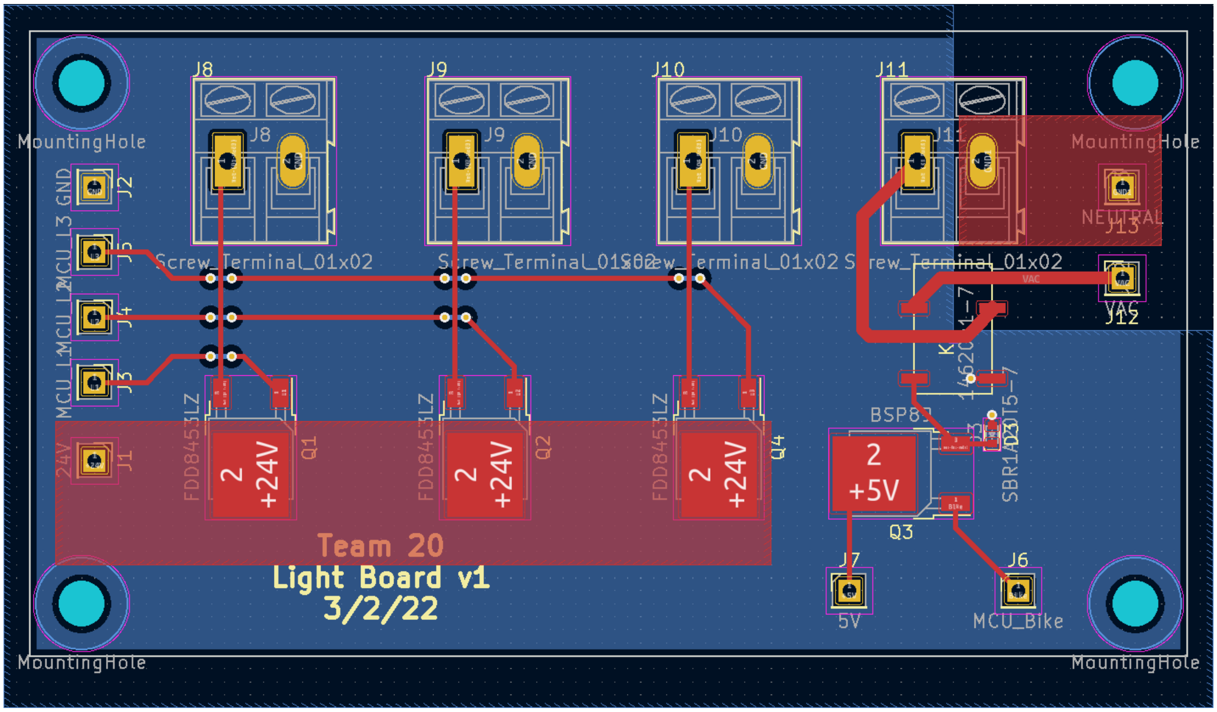

Figure 29: Initial traffic light board layout

D.3 Final Traffic Light Board Design

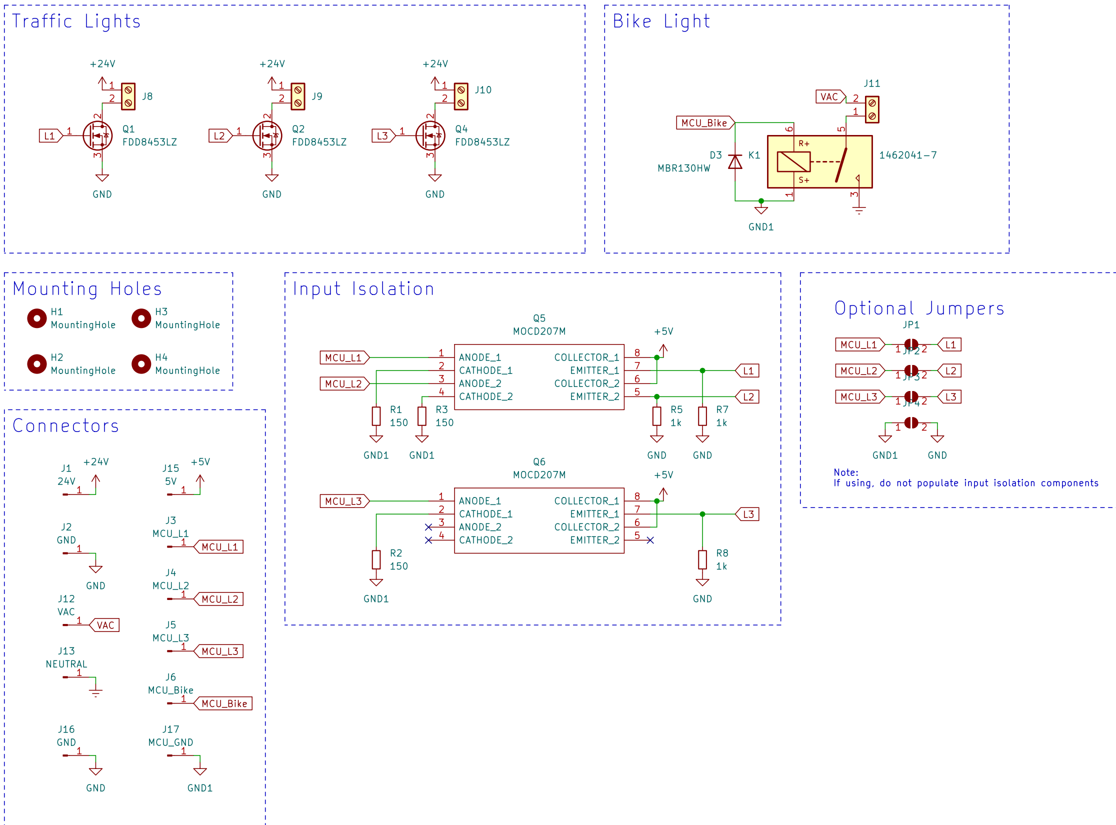

Figure 30: Final traffic light board schematic

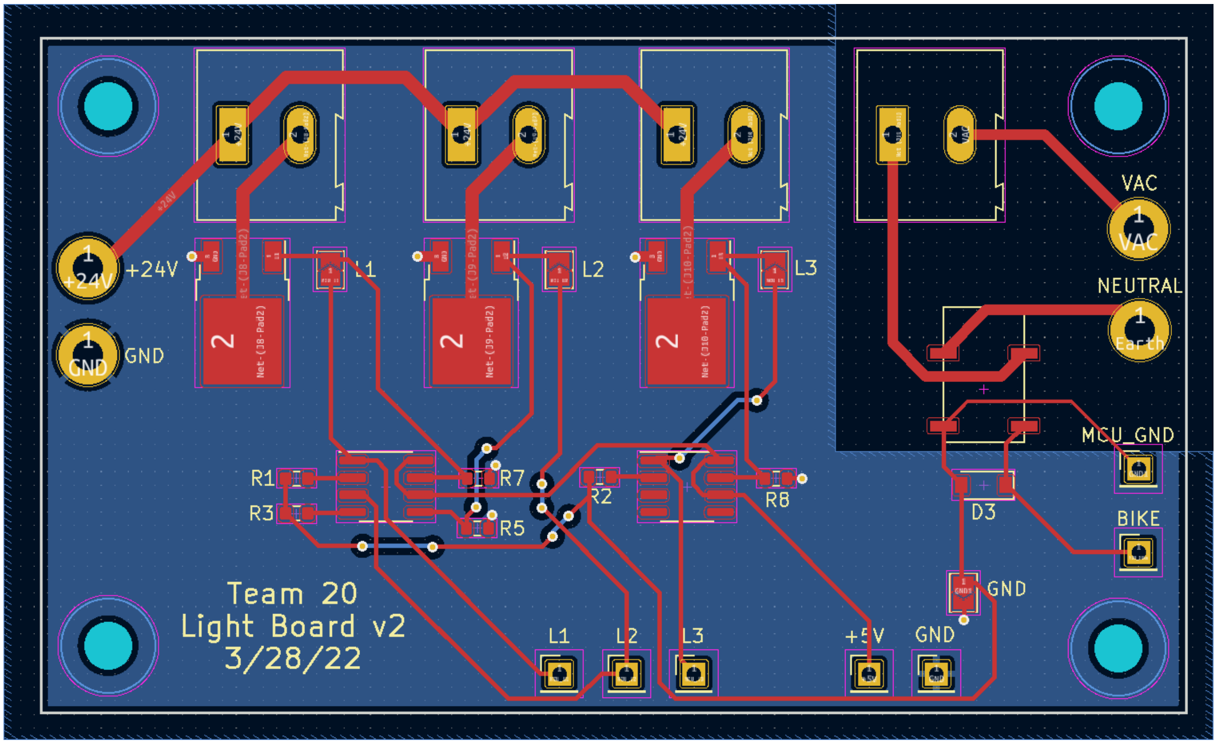

Figure 31: Final traffic light board layout