Real-Time University of Virginia Bus Routes Display System

Dan Tran (Computer Engineering), Kelly Zhou (Computer Engineering, Electrical Engineering), Zayda Deese (Computer Engineering, Electrical Engineering), Robel Woldegyorgis (Computer Engineering), and Anthony Nosal (Electrical Engineering)

Creating a user-friendly visual of the University of Virginia's bus lines and corresponding routes via a physical display to allow for easy access for bus riders.

2024

Table of Contents

Statement of Work

Dan: Tested initial TransLoc application program interface (API) to ensure that it would be fit to use before deciding initial project. Wrote all software regarding using TransLoc RESTful API to get real-time data of each bus including their vehicle ID, latitude, longitude, and bus color, and formatted so that data could be used. Wrote software to map the buses' latitude and longitude to spots on light-emitting diode (LED) strips. Wrote code to create a list of tuples containing buses and the number of stops away they are from the closest bus stop. Wrote final software that utilized real-time and saved bus data, passing on information to STM32 via universal asynchronous receiver/transmitter (UART), and relaying data to the LED strip. Setup both Raspberry Pis and installed proper dependencies to ensure the software would work as expected. Gathered and saved data to use as testing data. Tested running Raspberry Pi and giving data to LED strips, ensuring that bus location and color were working as expected. Worked on full integration testing and helped with integration troubleshooting.

Robel: Completed the 7-seg and LED indicator portion of the two different iterations of the PCB schematic and layout. Worked on code and breadboarding of the LED driver as the initial plan and pivoted to develop the plan and connection of direct wiring of the 7-seg and LED indicator. Searched and found LED strips and encasing being used for the display. Helped solder the STM32 header connector to the PCB as well as testing flex PCB. Repurposed an acrylic to be used for our project and ordered the display back encasing. Explored options to get map printed and added the layout spacing of the 7-seg and indicator LED on the map. Completed the final assembly of the display including the encasing, 7-seg, and indicator LED with team members. Assisted with final testing and integration of the display.

Zayda: Initially set up STM32. Physically mapped out 432 LEDs along the routes on the board. Translated those LEDs to a virtual map to retrieve and store the longitude and latitude of each LED in relation to the map in a CSV file. Wrote Python code to find the closest bus, where the function takes in parameters of Dan's tuple code and outputs a string in the format of 8 bytes (e.g. (CLR, #)) to return the closest bus's color and number of stops to its destination. Wrote C code to control the LED indicator using a timer and PWM generation signals. Configured necessary GPIOs and wrote code to control the 7-segment to display digits 0-9. Configured UART and its correct parameters on the STM32 (Rx) and the Raspberry Pi (Tx). Wrote C code to successfully transmit data, where the STM32 receives the 8-byte format mentioned earlier from the Raspberry Pi. Wrote C code and tested to integrate the UART logic code, 7-segment controller, and LED indicator controller. Helped with full system integration and testing.

Anthony: Designed Circuit Schematics and PCBs and ordered each iteration. Researched and ordered necessary components, and soldered the PCB. Assisted Kelly in soldering LED strips that were sandwiched on the board. Assisted in repurposing old acrylic panels to be used in our project, and had posters printed at the A-school. 3D printed RPi encasing and assisted with final assembly (drilling boards, fastening screws).

Kelly: Did the original circuit layout and tracing for the firsts PCB. Attached the LED strips to the board and made adjustments where necessary. Did most of the soldering for the LED strips, and debugged the strips when necessary. Worked with Anthony to create a test plan and performed fully comprehensive testing of the PCB, verifying that the correct voltages and signals were being sent to where it was expected. Worked closely with Anthony to figure out and pivot when necessary, mostly surrounding the original flex PCB design as well as power concerns. Helped Zayda debug the LED indicator to ensure the correct colors were being displayed. Went

to Lowe's with/without group members and helped assemble the final display board. Helped perform full integration testing.

Abstract

This project is a real-time bus tracking system that displays the location of each University of Virginia (UVA) Transit bus on a map using LEDs. Our goal is to make the bus system more accessible and readable by allowing riders to view real-time bus locations on a map using LEDs, eliminating the need to rely on the TransLoc App for updates. The project leverages Wi-Fi connectivity and utilizes a Raspberry Pi and STM32 microcontroller to collect, process, and transmit location data of UVA University Transit Service (UTS) buses to LEDs on a display board. We directly interface with the TransLoc API to obtain the location data, which we use by connecting a Raspberry Pi to the university's Wi-Fi network. The information is then processed by the Raspberry Pi, which controls LED strips to display the bus locations in real time. In the meantime, bus stop data is sent from the Raspberry Pi to the STM32, which is connected to an LED indicator and a 7-segment display that shows how many stops away an incoming bus is. Everything is powered using a 120V wall outlet that is converted to DC and stepped down appropriately.

Background

The project was designed to provide a more efficient alternative to TransLoc, the third-party mobile app, that UVA uses to provide information about the UVA UTS bus routes and stops to active users. Based on personal experience and feedback from others, the app does not effectively allow viewing live bus route information due to its poor user interface. Our project provides a more visually appealing and user-friendly tool, and it also addresses the issue of

accessibility. With the physical board, bus riders would not need to pull out their phones to check the application, and visitors can save time by not having to download the app.

Our project was inspired by the New York City (NYC) Live Subway Tracker product [1]. This product provides live data on trains arriving at NYC stations. A key difference in our project is the use of multiple controllers: an STM32 NUCLEO-G071RB and a Raspberry Pi 4 Model B, while the NYC tracker uses only an ESP32. In our setup, the Raspberry Pi handles receiving data, controlling the LEDs, and transferring the right information to control the 7-segment display and the bus indicator LED. The Raspberry Pi obtains the bus status information by connecting to the TransLoc RESTful API server, sending requests, and receiving responses wirelessly through Wi-Fi. Another distinction lies in our design of the live tracker. Unlike the NYC product, which integrates LEDs directly onto the printed circuit board (PCB), our project opts not to use the PCB as the main base for storing the LEDs. Instead, the LEDs are arranged behind a printed map backboard and a custom PCB is used to power our whole system and create the connection between the STM32, the 7-segment, and the indicator LED.

A previous project tracked Duluth buses using a similar interactive-light-up map design [2]. In that setup, the map was traced on paper and mounted on cardboard, with holes cut to place LED lights at intersections with bus stops along the tracked routes. The project utilized a Raspberry Pi 2 B+ for live code updates and to run the display, whereas our project uses a Raspberry Pi 4. Additionally, we are introducing a feature that displays the number of stops away a bus is from the reference stop onto the display. This will be achieved using a 7-segment display and an indicator LED.

This project draws on a range of concepts from our UVA coursework, which provided us with the skills to design and implement both hardware and software components into a fully

integrated system. The printed circuit board calculations and design were informed by knowledge from the fundamentals (FUN) series, while software development for requesting bus data from the TransLoc API and programming the LEDs drew on lessons from the Software Engineering and Computer Networks classes. Configuring the microcontrollers for communication protocols, like UART, was supported by what we learned in courses such as Introduction to Embedded Computer Systems and Advanced Computer Systems. Additionally, the system's power design was influenced by the studies in Power and Electromagnetic Energy Conversion classes. Even in areas where we may not have all the knowledge, the foundational principles from these courses enabled us to effectively research and find the precise information needed to overcome challenges.

Project Description

This project aims to have a readable and easy-to-follow bus status information visual display of the four UVA UTS bus lines: silver, gold, green, and orange. To do so, bus status information is pulled from the TransLoc API server every five seconds. By frequently fetching all bus data, accurate information is displayed on the physical map. After, the Raspberry Pi sends the bus status information to the LED strips using the LED's data line. The LED strips are programmed so that all four bus colors are displayed on the physical map, allowing riders to easily gather bus information.

The LED strip is laid on a map to cover all bus lines during the weekday academic hours, making it easy to follow each bus on its respective routes. At the same time, the STM32 programs an LED indicator, which distinguishes the color of the bus, and a 7-segment display to show how far out the buses are from the nearest bus stop to where the physical bus board is

located. A PCB was designed to manage power received from an outlet. The block diagram below, Figure 1, shows at a high level the different components of the system described above and how they work together. The PCB schematic can be seen in Figure 2 and the PCB layout can be seen in Figure 3, below.

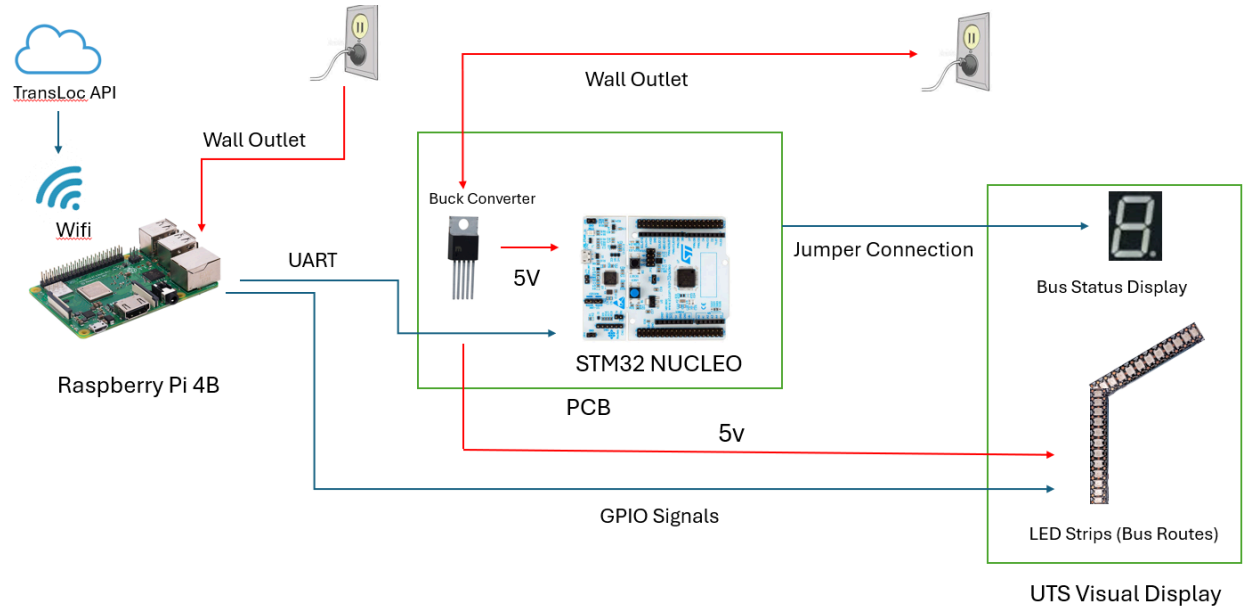


Figure 1. Block Diagram of the UTS Visual Display System

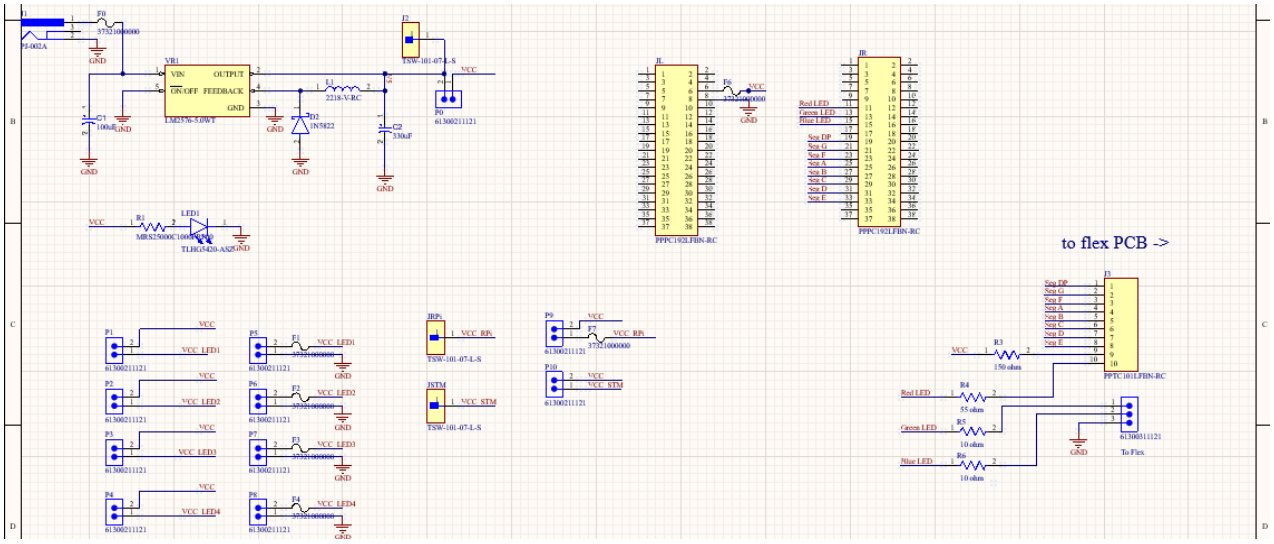


Figure 2. Printed Circuit Board (PCB) Schematic

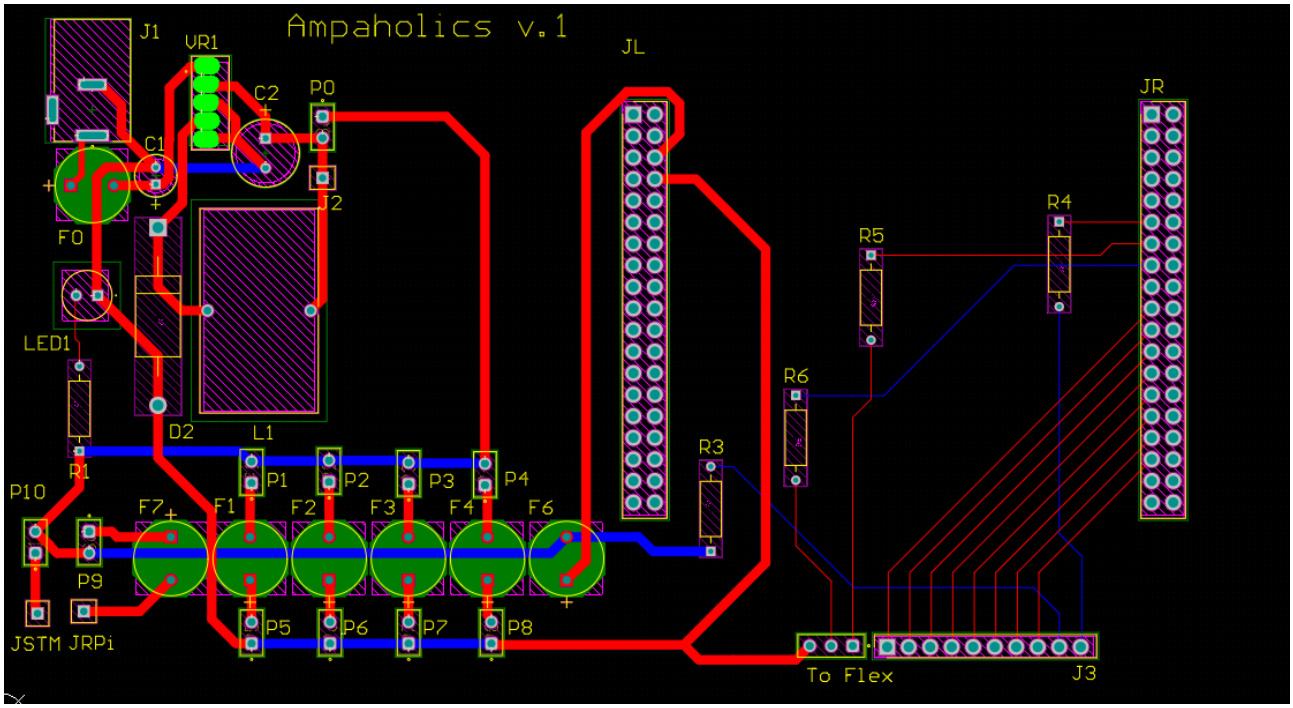


Figure 3. PCB Layout

When considering the core components of the final design, it was important to note the performance objectives and specifications of the final design. To begin, we will address the traits of the Raspberry Pi 4 [3] that was used. The Raspberry Pi needed to have Wi-Fi connectivity, a sufficient amount of computing power, and RAM storage to have the final design work as expected. The model being used has a dual-band wireless local area network (LAN) [4], which was crucial for allowing us to create the requests to the TransLoc API server–allowing us to obtain real-time bus data. The Raspberry Pi has 8GB of RAM, which was more than enough temporary storage to run the code needed. It also comes equipped with a 1.4GHz 64-bit quad-core processor, being able to withstand the constant, real-time updating required to run the software. The Raspberry Pi runs with an operating system [5] that enables fast computational capabilities allowing management of different programming capabilities. In addition to all the

Raspberry Pi use-specific capabilities, access to an extensive library and documented resources including the user guidebook [6] made it the optimal option.

Next, we will address the performance objectives and specifications for the STM32 NUCLEO-G071RB. Since the STM32 is connected to the Raspberry Pi 4, 7-segment display, and the LED indicator, there had to be a plentiful amount of general-purpose input/output (GPIO) pins on the microcontroller. Thus, the STM32 comes equipped with 64 GPIO pins that were used to fulfill those needs. The WS2812B RGB LED strips were used to light up the display for our performance objectives, specifications, and budget constraints. The LED strips provide 144 LEDs over a meter in length, which allows users to have a more accurate and comprehensible experience when understanding how the buses are moving on the display board. The LEDs swiftly activate with its 800 Kbps speed, which allows for high-speed sending of data. In addition, the LEDs provide a strong source of lighting with the WS2812B [7] SMD LEDs.

Additionally, a 7-segment visual display such as the 4166 seven-segment display [8] is used to display bus status information aiding the user to see how many stops away a certain bus is from the reference bus stop. Another crucial aspect that is used in conjunction with the 7-segment display is the LED indicator that shows which bus line the 7-segment is referring to.

To power the project, we used a wall outlet plug, going from 120V AC to 12V DC. The decision to use a wall outlet as opposed to a battery was based on the prospective location of where the project would sit. With the project being displayed inside a building, the wall outlet ensures that it can be continuously powered during university hours without the need for someone to replace the battery every some period of time. From the converted 12V DC, the power travels along the PCB to a buck converter–to step the voltage down to 5V. The buck converter routes to the STM microcontroller which requires 1.7V to 3.6V [9], and 3V matches

this range. The LED strips are connected to the PCB to provide power to the strips. Bypass capacitors are placed throughout the system to reduce noise from the power supply as suggested by IEEE [10]. In addition, fuses are added to each power line and bypass capacitors to protect all the components from overcurrent. This ensures that in the event of too much current present, the fuse will blow rather than the component, protecting expensive elements such as the STM microcontroller. By using all of these components, we created a project with a final design that meets the specifications and performance objectives of stakeholders and bus riders.

Test Plan

To address the test plan for the project, the project is composed of both hardware and software elements with four main subsystems: the TransLoc API, the LED strips, the 7-segment display and corresponding LED indicator, and the overall hardware of the project including the PCB. We ensured everything worked as expected through an organized testing plan. Regarding the usage of the TransLoc API, we verified that the data pulled from the API via computer can also be pulled from the Raspberry Pi and sent to the STM microcontroller. Following this, we checked the accuracy of the data. This was done by comparing the bus locations using the TransLoc mobile application. At any point, if the test failed, the software team went back and debugged their code before proceeding to the next step of the testing process.

The second subsystem, LED strips, had a similar process. Firstly, we checked if we could light up one specific LED, ultimately leading to all LEDs. This ensured that we could program each individual LED, which ties into the next step of connecting it to the TransLoc API. Once the LEDs were verified that they could be independently programmed, we used the data gathered from the TransLoc API to light up the LEDs. Similarly to the first software subsystem, if at any

point, the program did not pass testing, then the software team revisited the program and debugged it until the test passed.

The third subsystem is the 7-segment display and corresponding LED indicator. For the 7-segment display, we wanted to be able to display digits 0-9 correctly using the STM32. For the single LED, we programmed the color to green, gold, orange, and silver - the colors of the bus lines. Once both were established, we then connected it to the API. This was verified by checking if the bus line and number of stops away were correct, in other words, if the LED indicator was the right color and if the 7-segment display showed the expected number.

Lastly, the hardware subsystem first designed the initial schematic layout. After this, the power supply of the project was tested, ensuring the correct voltages for the buck converters. This was tested first because if the power did not work correctly, then the project cannot correctly be powered, ultimately leading to a failed project. After ensuring that the correct power was sent to their respective components, the individual elements were connected to the circuit and tested. With the verified power supply and individual components, the full PCB was ready for testing. Likewise, for the two software subsystems, the hardware team redesigned the schematic if they failed any part of the testing steps.

Once all four subsystems were tested for functionality and accuracy, the overall system as a whole was tested, ultimately verifying the integration of the software and hardware components. A high-level diagram is shown below in Figure 4, depicting the flow of our test plan for the system.

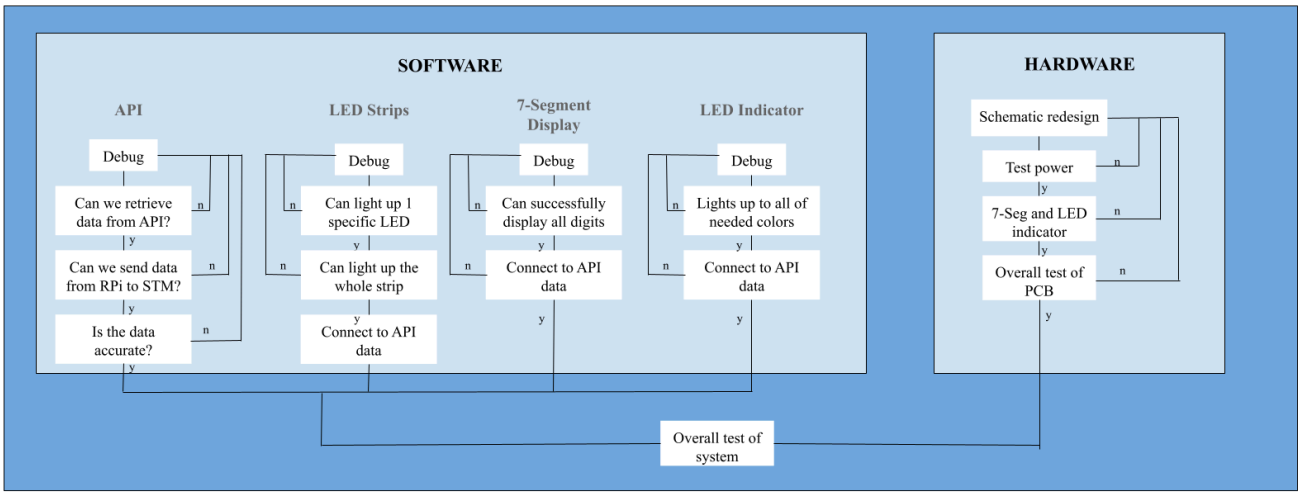


Figure 4. Test Plan Diagram

Physical Constraints

In terms of part availability, the project heavily relied on using LED strips for the different routes, making it critical to find an LED strip that was budget-friendly and available to order such as the one selected. It was also important to order more LED strips than what was needed for backups–in case of any accidental damages or errors. Given that a power outlet was used to power the LEDs, it was crucial to make sure the components used in our PCB such as the buck converter were in stock and within budget constraints.

The PCB was the only component that needed to be manufactured and no physical constraints were expected for the scope of our project. However, we had to keep in mind the cost of the PCB, the components used, and how many PCBs needed to be ordered for the first and revised order. This was critical to prevent ordering excessive parts while also ensuring the availability of a backup at the same time. Ensuring the prototype had fixed cost constraints, the budget was significant for our revised and final PCB as there could have been additional costs. Another constraint was simulating the power consumption for the LEDs given the vast number

of LEDs in the project. We were more reliable in analyzing and calculating the power drawn for the LED connections. Meaning, we expected to spend more time on the analytical calculations of LED power consumption than the simulation. Additionally, the final display needed to look user-friendly and clean, making us attentive on the materials used for the final board.

Societal Impact

The implementation of a real-time bus tracking system at UVA has significant societal implications for various stakeholders, including students, university staff, and the broader community. The proposed system aims to enhance the accessibility and usability of public transportation on campus by providing a physical display of bus routes and locations.

As previously mentioned, UVA UTS relies on the TransLoc mobile application to provide riders with information on bus routes and locations. However, the app's user interface is often criticized for being difficult to navigate, which can hinder effective communication of real-time bus data. By introducing a physical display system, riders can access bus information more intuitively without needing to interact with their phones. This shift could lead to increased satisfaction among users and potentially higher ridership, as accessing public transport could become more convenient.

Currently, UVA pays $16,440 per year [11] for TransLoc’s services. The designed physical board is a cost-effective alternative to communicate bus route and location information effectively, with each design costing less than $500 to produce. Due to the design's low cost, it has the potential to be adopted by UVA UTS, which would possibly shift funding from the TransLoc mobile application to this new physical display system. By reallocating finances from one product to another, the introduction of the physical display design would hurt TransLoc’s

revenue, potentially hurting the employees who maintain the application. Despite this potential shift, it is unlikely that UVA UTS will completely abandon the TransLoc app; instead, they may require additional funding to support both systems.

By sponsoring both systems, UVA UTS would require more funding to sponsor both the mobile app and the physical display. The budget would need to encapsulate both the creation of the displays as well as the maintenance that may be required for them in the future. With an increase in the demanded budget, UVA would likely be pulling money away from other processes and organizations for its accommodation. This change in budget allocation could indirectly harm university clubs, outsourced services, and students who depend on university funding, creating social and economic challenges for these groups. Also, the anticipated increase in ridership due to improved access to bus information could necessitate larger buses to accommodate more passengers, leading to additional expenditures for the university.

By sponsoring both TransLoc's mobile application and physical display board, students are likely to thrive in their quality of life. Easier access to bus information can save time and energy, facilitating visits with friends, trips to local shops, or easier commutes on and off campus. This would help increase the welfare of many of the students by opening up the idea of using the bus to those who may have not previously thought about it. Additionally, this increased accessibility may encourage more students to use public transportation instead of personal vehicles, contributing to reduced greenhouse gas emissions in the community.

In terms of obligations we, as creators of the physical display, have a significant responsibility to ensure that the system provides accurate and timely information to its users. This obligation is both ethical and practical, as students and other community members will likely depend on this system for their daily transportation needs. Our primary duty is to deliver

reliable and real-time bus information through the physical display system. Many students and staff may rely on this system to plan their commutes efficiently. Failing to provide accurate data could lead to missed classes, appointments, or other important commitments, thus negatively impacting users' daily lives. Meeting stakeholder expectations is vital for the project's success. Users expect a seamless experience that enhances their ability to navigate campus efficiently. By providing a user-friendly interface and reliable data, we can foster trust and satisfaction among our stakeholders, which is essential for the long-term adoption and success of the system. Overall, our commitment to providing accurate and timely bus information is an ethical obligation and a strategic necessity to ensure user satisfaction. By prioritizing these aspects in our design and implementation processes, we can maximize the societal benefits of the real-time physical bus tracking display for the community.

External Standards

The standards and regulations we must meet are the following:

- FCC Part 15 Regulations
- IEEE 802.11 Standards
- NEMA 250 Standards
- NFPA 70 Standards
- IPC Standards

The FCC Part 15 regulations deal with unlicensed RF devices. The devices must not cause harmful interference to licensed communications services and must comply with the emission limits and technical requirements[12]. The Raspberry Pi 4 that was used is compliant

with this regulation [13]. Likewise, the Raspberry Pi follows the IEEE 802.11 standard [14] which defines the technical specifications for wireless local area networks [15].

NEMA 250 defines the requirements for enclosures designed to protect electrical equipment against environmental conditions. The standard specifies various types of enclosures that provide protection against dust, dirt, water, ice, and corrosive elements, which is essential for maintaining the safety and functionality of electrical systems in different environments [16]. Our project will follow the standards for a NEMA 1 device, as it is planned to be used indoors. The enclosure for our device will primarily serve to prevent accidental contact with live parts.

NFPA 70 provides guidelines and requirements for the safe installation, operation, and maintenance of electrical systems [17]. The specific guidelines we will need to follow are:

- Minimum wiring gauge sizes
- Overcurrent Protection (fuses)
- Bonding and grounding of electrical systems.

The IPC standardizes the design, manufacture, and assembly requirements for electronics, specifically focusing on PCBs and electronics manufacturing. The specific standards we must comply with are:

- IPC-2221 Design Standards: Appropriate trace widths, clearances, and via sizes. Layer stack-ups and material choices that align with IPC guidelines. Thermal and mechanical considerations to avoid issues like warping or delamination [18].
- IPC-D-325 Documentation Requirements: Design files must be clear and complete to avoid errors during manufacturing [19].

- IPC-6012 (Manufacturing Certified): The PCB manufacturer we choose needs to be certified. Additionally, we will define our project as using a class 1 PCB as they are designed for general-purpose consumer electronics and non-critical applications [20].

Deliverables

By the end of the semester, we produced a software and hardware design that prioritizes the user's experience along with how they will interface with the product. As a broad overview of how the product functions, it uses a Raspberry Pi to enable close to real-time data collection using code software. By gathering and feeding this data into LED strips and an STM32, specific LEDs light up on the physical display corresponding to their latitude and longitude location as well as the bus line. As for the data fed into the STM32, additional software will be run on it to correctly configure the 7-segment display and LED indicator when necessary. This display is powered by an outlet whose PCB design focuses on handling and controlling the voltage that is being fed into the display.

For a more in-depth overview of the software, it creates a GET request to the RESTful TransLoc API. This code runs on a Raspberry Pi 4, which has wireless connection capabilities and sufficient computing power to handle the constant updating that occurs. Using the data from the GET request along with the software created to parse the data, we obtain a bus's latitude, longitude, vehicle ID, and vehicle color. The latitude and longitude give us the bus's current position, the vehicle ID is used to determine whether the bus is active, and the vehicle color determines what color to light the LED on the board. Thus, using this information, we created a mapping function that takes in a bus's latitude and longitude and determines which pin to light up. After determining which LED to light up along with its corresponding color, we used GPIO

pins and the data bus on the LED strips to effectively display the bus locations. Additionally, the software determines a list of the number of stops away all buses are from the closest bus stop for each line, and gives the STM32 a single bus line along with a corresponding number of stops away a bus is from the closest stop. The embedded software uses UART to interact between the Raspberry Pi and the STM32, where the STM32 communicates to the 7-segment display as well as the corresponding LED indicator using GPIO pins.

To provide a higher-level explanation of the hardware, a PCB was designed to mount the STM32, as well as the buck converter to manage the supply voltages needed for each device. To power the board, we used a power supply to convert a wall outlet 120V AC to 12V DC, which is connected to the PCB using a standard barrel jack. We then used a buck converter to step the voltage down to 5V for the Raspberry Pi, the LED strips, and STM32. The VCC lines connecting to each component were fused appropriately in case of any short circuits.

To dive deeper into what the user interface and user experience looks like, the user interface is a board with a map of the UVA Academic Bus System map along with LEDs at specific points along all routes. The display only shows bus routes that have academic routes during academic hours, so it displays: Silver Line from 7:30 am to 8:00 pm, Gold Line from 5:00 am to 6:00 pm, Green Line from 7:30 am to 6:00 pm, and Orange Line from 7:30 am to 6:00 pm. We have LEDs that cover all the academic routes and are placed on top of the bus map as previously described. As for understanding the user experience, we have the display board placed and plugged into one of the UVA engineering buildings.

To describe how prototyping for the design worked, we focused on integrating the core components and putting less of a focus on the user interface. We ensured that the Raspberry Pi could run independently and feed information into the LED strips and the STM32. After we

ensured that we could control all the LED strips from the Raspberry Pi and that information being passed to the STM32 via UART was being received. Test data was obtained by saving data from the TransLoc PublicAPI over 10-20 minutes during academic hours when the desired routes were running.

When taking into consideration the design along with the budget allocated for the project, we believed that most of our expenditures would be from the PCB, Raspberry Pi, STM32, and LED strips. Although these were considered the main purchases for our team, all of them together cost less than a fifth of the overall budget allocated. Since the items listed were the core components of our design, we had more freedom in terms of scaling the design and improving it along the process. Another thing we purchased with our budget is the large board to place the LEDs and the UVA Academic Route map on, the encasing material, and anything needed additionally for handling the charging of the PCB.

Intellectual Property Issues

In exploring the patentability of our project, it is important to analyze existing patents with similar features and noticeable relevance. Claims within patents are categorized as independent and dependent, where independent claims broadly define the main features of the product and dependent claims build upon these by adding specific details. By examining the different claims in patents relating to LED systems, live data visualization, and route mapping, we learned the extent our project aligns with prior inventions to gain insight into the potential patentability for our project.

One patent related to our technical project is titled “LED strips bussing system and process” [21]. It describes a system for connecting multiple single-color LED strips in parallel

without requiring insulation between adjacent copper solder pads. The independent claim outlines a system comprising LED strips with positive and negative branches, isolated solder pads at predetermined locations, conductive materials connecting the branches in parallel, and an external DC voltage power source. Dependent claims add details such as the use of uninsulated wires to establish parallel connections and the inclusion of additional strips. This patent is relevant to our project, which involves assembling LED strips to create a visual display mimicking bus routes. Like the patent, our project uses parallel connections and customizable strip lengths, but it specifically uses LED strips with a physical map and real-time bus data. While the general method of connecting LED strips in parallel is used, the application of our project - cutting, arranging, and soldering LED strips for real-time bus data - offers novel features that may be patentable.

The patent titled “Road map display system with indications of a vehicle position and destination” [22] describes a system for dynamically displaying a vehicle’s position and destination on a road map. Its independent claim outlines a system consisting of a map memory for storing road map data, sensors to detect travel distance and direction, an arithmetic unit to calculate coordinates, a coordinate transformation mechanism for aligning map data with the vehicle’s direction, and a visual display for presenting the data. Dependent claims add specificity, such as the ability to select specific portions of the map, adjust the map scale dynamically, and recalibrate in real time to correct errors. Both our project and the patent involve real-time visualization of positional data. However, the patented system dynamically adjusts an electronic display, and our project integrates LED strips with a physical map, providing a simpler, less costly, and unique approach to visualizing live bus routes. While the general concept of

displaying real-time positional data is not new, the specific implementation of it within our project could potentially lead to patentability.

Another patent to consider is the titled “Large Scale LED Display System” [23], which outlines a robust system for distributing data to large-scale LED displays composed of multiple panels. Its independent claim describes a system where display panels are connected to multiple data hubs, ensuring reliable data distribution and redundancy. Dependent claims add details, such as a central controller managing pixel distribution and the ability to detect transmit transmission failures and switch to other data paths. This patent relates to our project in its emphasis on real-time data distribution to LED displays. While both systems involve LEDs and data transmission, our project is tailored to displaying live bus information on a physical map using LED strips. The assembly of LED strips along mapped bus routes and the visualization of real-time transit data differentiate our implementation from the LED panel approach described in the patent. While the patent focuses on reliability and redundancy for large-scale displays, our project centers on a specific application of LED strips for visualizing public transit. As a result, our project with its specific application and physical assembly may be novel and potentially patentable.

Timeline

At the very start of the project, the main deliverables were laid out and discussed, leading to the creation of the Gantt chart, shown below in Figures 5-9. After solidifying the project idea, 1.5 weeks were dedicated to researching the topic: identifying the stakeholders and customers, investigating prior works, selecting a microcontroller, and discussing requirements and

constraints. This research period also included a discussion on parts to order for the project, which continued throughout the timeline when needed.

Once the details of the project were cemented, members worked on the project proposal, which was split up to evenly distribute the workload per person where the member assigned to each section of the proposal is shown in Figure 5 below. The deadline was set for September 18, 2024, which was two days before the due date shown on the class page, allowing for enough buffer time for revisions or in the case tasks took longer than expected.

TASK TITLE	Member	START DATE	DUE DATE	DURATION (Days)	WEEK 1 (9/2-9/6)					WEEK 2 (9/9 - 9/13)					WEEK 3 (9/16-9/2			
					M	T	W	R	F	M	T	W	R	F	M	T	W	R
Research																		
Stakeholder & Customer Definition	All	9/4/2024	9/11/2024	7														
Prior Works Research	All	9/4/2024	9/11/2024	7														
Microcontroller Selection	All	9/4/2024	9/11/2024	7														
Requirements and Constraints	All	9/4/2024	9/11/2024	7														
Project Proposal		**9/4/2024**	**9/20/2024**	**16**														
Abstract	Anthony	9/12/2024	9/18/2024	6														
Background	Zayda	9/12/2024	9/18/2024	6														
Project Description	Dan, Robel, Kelly	9/12/2024	9/18/2024	6														
Physical Constraints	Robel	9/12/2024	9/18/2024	6														
Societal Impact	Dan	9/12/2024	9/18/2024	6														
External Standards	Anthony	9/12/2024	9/18/2024	6														
Deliverables	Dan	9/12/2024	9/18/2024	6														
Timeline	Kelly	9/12/2024	9/18/2024	6														
Expectations	Zayda	9/12/2024	9/18/2024	6														
References	All	9/12/2024	9/18/2024	6														

Figure 5. Week 1-Week 3.5

The next major event after the proposal was the Poster Session. Similarly to the proposal, the sections of the poster were also split up among team members shown below in Figure 4. The deadline for the poster was set for September 27, 2024, a week before the actual poster session. Once the poster had been turned in and finalized, the initial start of the project took place. Here the team split into two main groups: software and hardware. The hardware team, Anthony, Robel, and Kelly, did the calculations, simulations, and prototyping surrounding the hardware of

the project. Simultaneously, the software team, Zayda and Dan, worked on developing the code and testing it with the microcontrollers. While in the development phase, both the hardware and software teams worked on creating their respective testing plans, which were used following the midterm design review (marking the halfway point). All of this information can be seen below in Figure 6.

TASK TITLE	Member	START DATE	DUE DATE	DURATION (Days)
Posters Due		**9/18/2024**	**9/27/2024**	**9**
Background	Zayda	9/19/2024	9/26/2024	7
Project Description	Dan	9/19/2024	9/26/2024	7
Deliverables (PCB + software)	Robel	9/19/2024	9/26/2024	7
Societal Impact	Kelly	9/19/2024	9/26/2024	7
Submit Graphics	Anthony	9/19/2024	9/26/2024	7
Poster Session		**10/4/2024**	**10/4/2024**	**0**
Calculation				
Create Schematic	Anthony/Robel	9/27/2024	10/13/2024	16
Circuit Analysis	Anthony/Robel	9/27/2024	10/13/2024	16
Simulation				
Simulate circuit	Kelly	9/27/2024	10/13/2024	16
Initial PCB layout	Kelly/Anthony/ Robel	9/27/2024	10/13/2024	16
Prototyping				
Breadboarding	Kelly	9/27/2024	10/13/2024	16
Software and Integration				
Software Development	Dan/Zayda	9/27/2024	10/13/2024	16
Software Testing with Microcontroller	Dan/Zayda	9/27/2024	10/13/2024	16
Testing Plan				
Hardware Testing Plan	Kelly/Anthony/ Robel	9/27/2024	10/13/2024	0
Software Testing Plan	Dan/Zayda	9/27/2024	10/13/2024	16
Integration Testing Plan	All	9/27/2024	10/13/2024	16

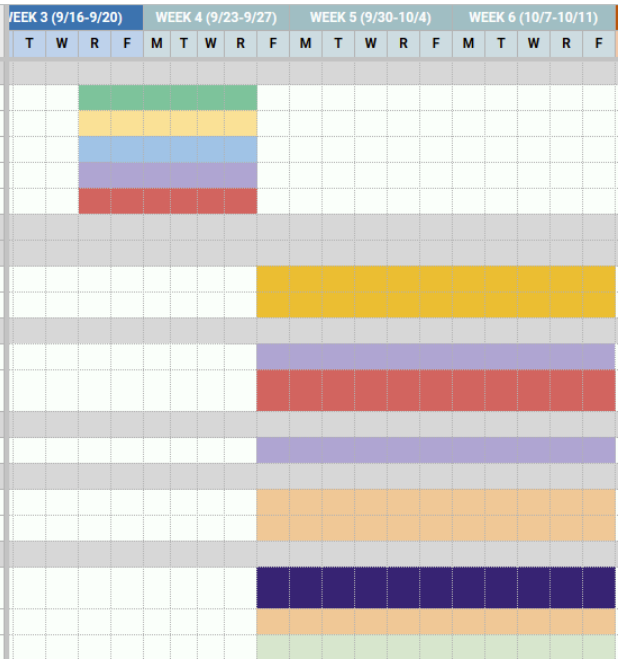
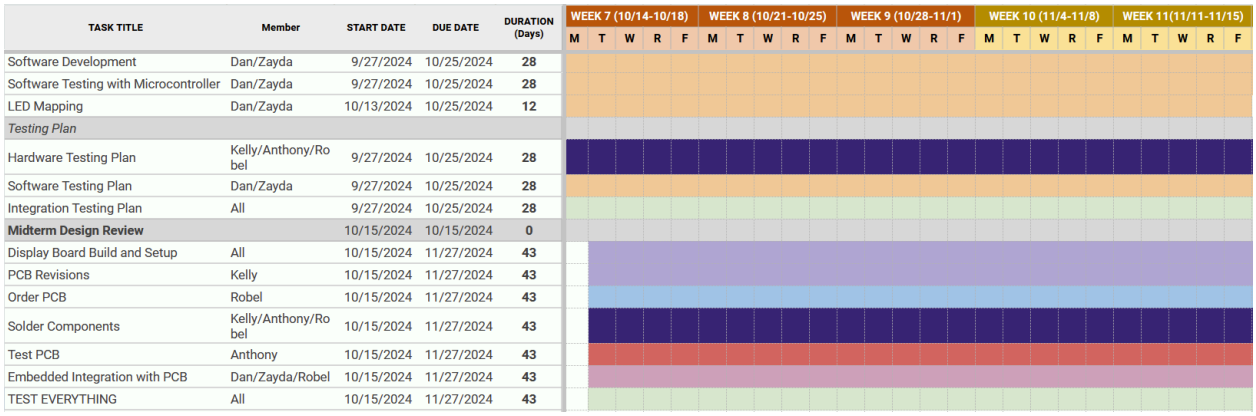


Figure 6. Week 3.5 - Week 6

Following the halfway point, most of the time was used in finalizing the project: ordering the PCB, making revisions when necessary, putting the components together, ensuring seamless integration between the microcontrollers and PCB, and testing everything. From weeks 7 through 9, the software team continued software development, focusing on the LEDs. After the LEDs were confirmed to work as expected, they switched to focusing on the 7-segment display. All throughout this time, rigorous software, hardware, and integration testing was being performed as shown in Figure 7.



TASK TITLE	Member	START DATE	DUE DATE	DURATION (Days)
Software Development	Dan/Zayda	9/27/2024	10/25/2024	28
Software Testing with Microcontroller	Dan/Zayda	9/27/2024	10/25/2024	28
LED Mapping	Dan/Zayda	10/13/2024	10/25/2024	12
Testing Plan				
Hardware Testing Plan	Kelly/Anthony/Robel	9/27/2024	10/25/2024	28
Software Testing Plan	Dan/Zayda	9/27/2024	10/25/2024	28
Integration Testing Plan	All	9/27/2024	10/25/2024	28
Midterm Design Review		10/15/2024	10/15/2024	0
Display Board Build and Setup	All	10/15/2024	11/27/2024	43
PCB Revisions	Kelly	10/15/2024	11/27/2024	43
Order PCB	Robel	10/15/2024	11/27/2024	43
Solder Components	Kelly/Anthony/Robel	10/15/2024	11/27/2024	43
Test PCB	Anthony	10/15/2024	11/27/2024	43
Embedded Integration with PCB	Dan/Zayda/Robel	10/15/2024	11/27/2024	43
TEST EVERYTHING	All	10/15/2024	11/27/2024	43

Figure 7. Week 7-Week 11

The last three weeks were spent meeting the final deliverable requirements and ensuring they worked through rigorous testing. Additionally, the assembly of the board began on week 12 and went on until the beginning of week 14. As integration testing between the software and hardware began to come to an end, ensuring that the product worked as expected, we began putting all final components into the final board. Once all components were placed into the board, a final integration test was done to ensure all components were working as expected. While doing this, time was allotted to write the final report and to create the final video. Following this schedule gave us sufficient time to complete the final design, work on the final report, and create the final project video before the demo as shown in Figure 8. Lastly, with everything turned in, the demo was rehearsed as a group to prepare for the Capstone Demo on December 9, 2024.

TASK TITLE	Member	START DATE	DUE DATE	DURATION (Days)
Software and Integration				
Software Development	Dan/Zayda	9/27/2024	10/25/2024	28
Software Testing with Microcontroller	Dan/Zayda	9/27/2024	10/25/2024	28
LED Mapping	Dan/Zayda	10/13/2024	10/25/2024	12
Testing Plan				
Hardware Testing Plan	Kelly/Anthony/Robel	9/27/2024	10/25/2024	28
Software Testing Plan	Dan/Zayda	9/27/2024	10/25/2024	28
Integration Testing Plan	All	9/27/2024	10/25/2024	28
Midterm Design Review		10/15/2024	10/15/2024	0
Display Board Build and Setup	All	10/15/2024	11/27/2024	43
PCB Revisions	Kelly	10/15/2024	11/27/2024	43
Order PCB	Robel	10/15/2024	11/27/2024	43
Solder Components	Kelly/Anthony/Robel	10/15/2024	11/27/2024	43
Test PCB	Anthony	10/15/2024	11/27/2024	43
Embedded Integration with PCB	Dan/Zayda/Robel	10/15/2024	11/27/2024	43
TEST EVERYTHING	All	10/15/2024	11/27/2024	43
Board Assembly	All	11/11/2024	12/1/2024	20
Write Final Project Report	All	11/27/2024	12/4/2024	7
Create Final Project Video	All	11/27/2024	12/4/2024	7
Final Project Report	All	11/27/2024	12/5/2024	8
Final Project Video	All	11/27/2024	12/5/2024	8
Rehearse Demo	All	12/5/2024	12/9/2024	4
Project Demo	All	12/5/2024	12/9/2024	4

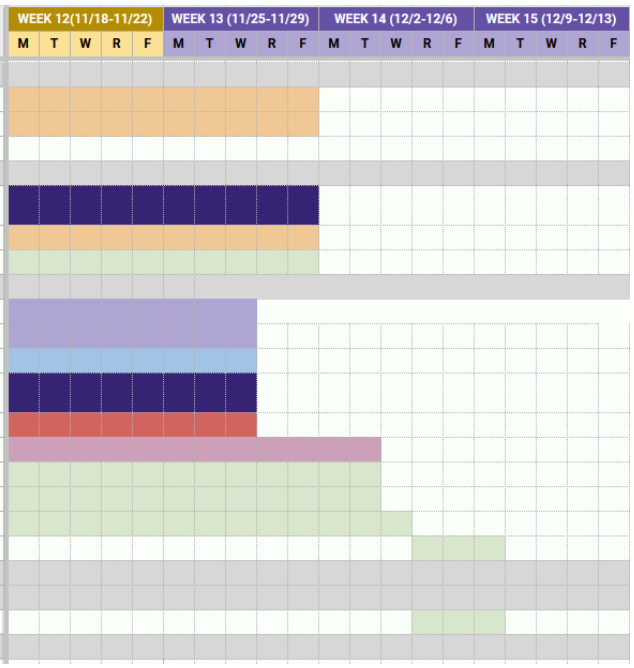


Figure 8. Week 12-Week 15

A color-coded legend of different combinations of people as well as individual contributions can be seen below in Figure 9.

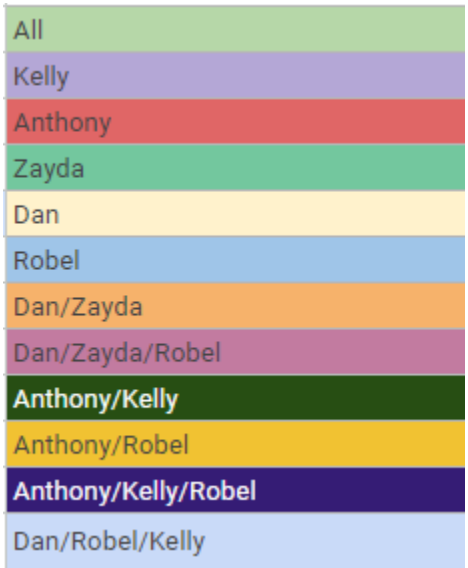


Figure 9. Legend for Gantt Chart

Costs

To produce our display we purchased $390 worth of parts, the main expenditures being the Raspberry Pi's, the PCB iterations, and LEDs. Much of our budget was spent prototyping and testing as we had an extra PCB iteration as well as an extra Raspberry Pi that was not used in the final assembly. Additionally, we did not use the flex PCB that we ordered. The actual value of the parts that made it to the final board was roughly $240, which was lower than we anticipated as we were able to reuse parts from previous capstone classes as well as an acrylic panel that was no longer being used. A similar acrylic panel could be purchased online for around $20 without shipping, and it is difficult to tell how much of an impact on the budget reusing electronic components had since it consisted of many small cheaper components. A detailed table of our costs is located in Appendix A.

When accounting for producing 10000 units of this project, there are a couple of things to take into consideration. With bulk purchases, many manufacturers give a discounted rate. This is especially true with sellers like Digikey, where we purchase many of our hardware components. For example, the inductor used for the power supply costs $3.43 for one inductor, but $2.04 per unit when purchasing the highest bulk order. Additionally, our PCB manufacturer (JLCPCB) offers bulk discounts on large PCB orders, which would bring our unit cost per PCB down from $7.41 to $0.45 each. We were fortunate enough to have a lot of equipment already available without having to use our budget. Although we had access to a lot of equipment, it is not nearly enough to make 10000 units so we'd have to include items like hex nuts, jumper wires, a clear acrylic board, and shunts for production.

Another thing to consider is with 10000 units to be made, many of our processes can be automated to produce as many units in the least possible amount of time and resources. Some of

these processes include soldering components onto the PCB, drilling into the board, soldering LED strips, and overall assembly. Automating these tasks would further decrease the unit cost of each display, and would decrease turnaround time as well. A detailed table of large-scale manufacturing costs is also located in Appendix A.

Final Results

To briefly explain the final prototype, we used a Raspberry Pi 4 to pull data from the TransLoc API server, running software on it to relay active buses as well as their respective locations onto three WS2812B LED strips. Each LED strip covers a portion of the UVA academic bus map so that all buses can be accurately displayed. The Raspberry Pi 4 also communicated to the STM32 via UART to provide data to the 7-segment display, which tells riders how many stops away a certain bus is from the reference stop, and an LED indicator, which tells riders which bus is the number of stops away. The PCB design will power the LED strips and the STM32 by downstepping power from an outlet using a buck converter.

The features that remained the same as described in the initial proposal were the Raspberry Pi pulling data from the TransLoc database and using software to have it provide data to the STM32 to pass on values to the 7-segment display. Additionally, using the PCB to power the LED strips and the STM32 by down-stepping power from the outlet remained the same.

The differences between the final product and what was described in the proposal were how the LED strips were going to receive the bus route data information, how the Raspberry Pi was going to communicate to the STM32, and the type of Raspberry Pi that was used. In the original proposal, we were planning on having the STM32 relay information to the LED strips, but after the midterm design review, we were advised that interrupting the pushing of data to the

LED strips while it is relaying information to the LED strips may cause damage to the strips, so to solve this problem, we decided to switch giving the data to the LED strip from the STM32 to having the Raspberry Pi transmit it through GPIO pins. Next, the Raspberry Pi was originally going to use SPI to communicate data from the Raspberry Pi to the STM32, but given the amount of documentation online for using UART and the fact that we only needed the STM32 to receive and not transfer information, we decided to switch from SPI to UART instead. Lastly, we transitioned from using a Raspberry Pi 3B+ to a Raspberry Pi 4. Originally, we purchased the Raspberry Pi 4 as a backup alternative in case we required more memory for the device, but the actual reason for the swap was that the Raspberry Pi 3B+ crashed and was not able to be powered on afterward. The reason why the Raspberry Pi 3B+ stopped working is still unknown, but since we had the Raspberry Pi 4 just in case, we transitioned to the Raspberry Pi 4.

The overall results of the final project were as expected. The project successfully integrates the PCB, microcontrollers, and LED layout to provide reliable and accurate live data for the four academic bus routes. Based on our expectations listed in the project proposal, listed in Appendix B, our final product meets full integration and functionality requirements. All the hardware components are integrated and operate correctly. Real-time tracking for all four academic bus routes was successfully implemented as well as the data delay was kept within 20 seconds. This was proven by standing near the referenced bus stop and confirming that the correct bus stopped at its destination as well as it lined up with what was showcased on our visual display. In addition, we provided an intuitive and clear user interface that was supported by the clean assembly of our final project.

Engineering Insights

The project has been incredibly enlightening in terms of understanding the full scope of what it means to be an engineer. It has demonstrated that both technical skills and soft skills are essential for completing an engineering project.

On the technical side, the hardware team became proficient with industry-standard PCB design software, Altium, enabling them to create a collaborative PCB design while applying their knowledge of circuits. This experience has equipped them with the skills necessary to design circuits and PCBs in their future careers. The embedded software team gained significant insights into using a Raspberry Pi and its interaction with header pins. By integrating this new knowledge with their existing understanding of UART communication, they were able to facilitate effective communication between the Raspberry Pi and the STM32 microcontroller.

In terms of soft skills and lessons learned, adaptability emerged as a crucial aspect of the engineering process. Engineering projects often encounter unforeseen challenges, making it essential to pivot directions effectively. The ability to remain flexible and open to change is vital in overcoming these obstacles. Additionally, managing time and resources efficiently was critical due to constraints such as budget and deadlines. Effective communication within the team was paramount in discussing ideas and resolving issues collaboratively, ensuring that all members were aligned with project goals. Working on this project also highlighted the importance of maintaining high morale and motivation throughout the process. Celebrating small successes and supporting each other during challenging times helped keep the team motivated and focused on achieving their objectives.

For future Capstone students, our advice would be to embrace flexibility and be prepared to adapt plans as new challenges arise. Prioritize clear communication within your team to prevent misunderstandings and streamline problem-solving. Manage your time wisely by

developing a realistic timeline and adhering to it as much as possible. Leverage each team member's strengths by assigning tasks accordingly to maximize efficiency. Lastly, maintain a positive attitude by celebrating achievements, no matter how small, and supporting one another through challenges. Overall, this project was a comprehensive learning experience that reinforced the importance of both technical expertise and interpersonal skills in engineering.

Future Work

To enhance and expand upon the current project, future iterations could consider incorporating all UVA bus routes, rather than limiting the design to only the weekday academic routes. By including evening and weekend routes, the physical display board could serve a broader audience and provide more comprehensive coverage of UVA's transit system. During discussions with UVA UTS, it was suggested that having access to this information in various locations, such as libraries, could greatly benefit students who study late at the library and need reliable transportation information to return home safely. This expansion would significantly increase the utility of the system. Implementing this feature would require updating the design to accommodate additional route data, which may involve technical challenges such as increased data processing and display capabilities. Moreover, future teams should be aware of potential difficulties that were not initially anticipated. Additionally, maintaining the hardware's durability and reliability in various environmental conditions is crucial for consistent performance. Advice for future teams includes being prepared for unexpected technical hurdles and having a flexible approach to problem-solving. It's important to plan for scalability to accommodate potential expansions like additional routes or more complex displays. Teams should also prioritize thorough testing to ensure that any new features integrate seamlessly with existing systems. By

addressing these areas, future projects can build upon the foundation laid by this project, enhancing its functionality and impact on the university community.

References

[1] *New York City Subway - Traintrackr - Live LED Maps*. https://www.traintrackr.io/product/mta3. [Accessed 18 Sept. 2024].

[2] *Tracking Duluth Buses with an Interactive Light-up Map | ChandlerSwift.Com*. https://chandlerswift.com/projects/bus-tracker. [Accessed 18 Sept. 2024].

[3] *Raspberry Pi Foundation, "Raspberry Pi 4 B Product Brief," Raspberry Pi, 2018*. [Online]. Available: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf. [Accessed: Sep. 20, 2024].

[4] *V. Mihalopoulos, "The Raspberry Pi 4 Ultimate Review," yodeck, 2019.* [Online]. Available: https://www.yodeck.com/news/the-raspberry-pi-4-ultimate-review. [Accessed: Sep. 20, 2024].

[5] *Raspberry Pi, "SC0339L," Digi-Key, 2024.* [Online]. Available: https://www.digikey.com/en/products/detail/raspberry-pi/SC0339L/12339165?s=N4IgTCBcDaIMoGEAMBmFBOAMiAugXyA. [Accessed: Sep. 20, 2024].

[6] *S. Monk, Programming the Raspberry Pi: Getting Started with Python, 2nd ed. New York, NY, USA: McGraw-Hill Education, 2015*. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=WHPhDAAAQBAJ&oi=fnd&pg=PA1&dq=raspberry+pi+3%2B&ots=cI9UZhybnQ&sig=QwFyiSRe6mrug-9yXjmRSyvEVUA#v=onepage&q=raspberry%20pi%203%2B&f=false. [Accessed: Sep. 20, 2024].

[7] *Adafruit Industries, "WS2812B Intelligent Control LED Datasheet," Adafruit, 2013*. [Online]. Available: https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf. [Accessed: Sep. 20, 2024].

[8] *Gearbox Labs, "Part 4-Digit 7-Segment Display," Digi-Key, 2024.* [Online]. Available: https://www.digikey.com/en/products/detail/gearbox-labs/PART-4-DIGIT-7-SEGMENT-DISPLAY/16161106. [Accessed: Sep. 20, 2024].

[9]STMicroelectronics, "STM32G071x(8, B) Datasheet ," Nov. 2018. Accessed: Sep. 18, 2024. [Online]. Available: https://www.digikey.pl/htmldatasheets/production/3540770/0/0/1/stm32g071gbu6.html. [Accessed: Sep. 19, 2024].

[10] R. T. Fizesan, D. Pitica, and L. Man, "Power integrity analysis and bypass capacitor selection using FDM on a printed circuit board," May 2009, doi: https://doi.org/10.1109/isse.2009.5207051. [Accessed: Sep. 19, 2024].

[11] University of Virginia Health System, "UPG Prices of Provider Services," July 2021. [Online]. Available: https://uvahealth.com/sites/default/files/2021-07/upg.prices.july2021.pdf. [Accessed: 27-Nov-2024].

[12] 47 CFR Part 15 - Radio Frequency Devices," *Electronic Code of Federal Regulations (eCFR)*. Available: https://www.ecfr.gov/current/title-47/chapter-I/subchapter-A/part-15. [Accessed: Sep. 16, 2024].

[13] FCC ID 2ABCB-RPI4: Raspberry Pi 4, FCC ID.io. Available: https://fccid.io/2ABCB-RPI4. [Accessed: Sep. 16, 2024].

[14]Raspberry Pi 4 Model B," Raspberry Pi. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ [Accessed: Sep. 16, 2024].

[15] The Evolution of Wi-Fi Technology and Standards," *IEEE Standards Association*. Available: https://standards.ieee.org/beyond-standards/the-evolution-of-wi-fi-technology-and-standards/ [Accessed: Sep. 16, 2024].

[16] ANSI/NEMA 250-2020 Contents and Scope," *NEMA Standards Document Library*. Available:https://www.nema.org/docs/default-source/standards-document-library/ansi_nema_250-2020-contents-and-scope76f809d7-afad-4aa1-80cd-e1d09b60f2e5.pdf?sfvrsn=cb4086bd_3 [Accessed: Sep. 16, 2024].

[17] NFPA 70 Chapter 3: Wiring Methods and Materials," UpCodes. Available: https://up.codes/viewer/virginia/nfpa-70-2020/chapter/3/wiring-methods-and-materials#4 [Accessed: Sep. 16, 2024].

[18] IPC-2221A: Generic Standard on Printed Board Design, IPC. Available: https://www.ipc.org/TOC/IPC-2221A.pdf. [Accessed: Sep. 16, 2024].

[19]*IPC-D-325A: Documentation Requirements for Printed Boards, Assemblies, and Support Drawings*, *IPC*. Available: https://www.ipc.org/TOC/IPC-D-325A.pdf. [Accessed: Sep. 16, 2024].

[20]2023 IPC Class 1 Products," Cadence PCB Design Blog. Available: https://resources.pcb.cadence.com/blog/2023-ipc-class-1-products [Accessed: Sep. 16, 2024].

[21] "US10663154B2 - LED strips bussing system and process - Google Patents," *Google.com*, Sep. 13, 2019. https://patents.google.com/patent/US10663154B2/en (accessed Dec. 05, 2024).

[22] "US8558755B2 - Large scale LED display system - Google Patents," *Google.com*, Dec. 11, 2007. https://patents.google.com/patent/US8558755B2/en (accessed Dec. 05, 2024).

[23]"US Patent for Road map display system with indications of a vehicle position and destination Patent (Patent # 4,543,572 issued September 24, 1985) - Justia Patents Search," *Justia.com*, Apr. 29, 1982. https://patents.justia.com/patent/4543572 (accessed Dec. 05, 2024).

Appendix

Appendix A. Purchased Items and Cost, Mass Production Costs.

Item	Cost
PCB Order 1 (2 PCBS + shipping and tax)	$45.78
PCB Order 2 (2 PCBS + shipping and tax)	$51.68
Raspberry Pi 3B+	$35.00
Raspberry Pi 4B	$85.97
STM32	$11.04
SD Card	$12.57
LED Strips	$38.97
Buck Converter Circuit	$5.30
Hardware (Nuts, Bolts, and spacers)	$8.19
Fuses	$5.00
Backboards	$17.96
LED Driver	$18.99
7 Segment Display	$2.60
RGB LEDs	$6.00
Power Supply	$15.00
Electrical Box	$10.00
Posters	$6.00
Total	$390.33

Table 1: Purchased Items and Costs

Item Name	Qty Req'd	Per Unit Price for Prototype USD	Cost for Prototype Price USD	Per 10000 Units Price USD	Cost for 10000 Units Price USD
STM32 Microcontroller	1	11.04	11.04	11.04	110400
LED Strips	3	12.99	38.97	12.99	129900
7-segment Display	1	1.31	1.31	0.37573	3757.3
Buck Converter	1	1.75	1.75	1.33	13300
Inductor	1	3.43	3.43	2.04377	20437.7
Schottky Diode	1	0.96	0.96	0.26982	2698.2
Barrel Connector	1	0.52	0.52	0.2652	2652
STM connector	2	2.05	4.1	1.209	12090
LED/7-seg connector	1	0.62	0.62	0.325	3250
LED Indicator	1	1.15	1.15	1.15	11500
Carriage Bolts	6	0.51	3.06	0.51	5100
Washers	6	0.16	0.96	0.16	1600
Nylon spacers	6	0.695	4.17	0.695	6950
Hex Nuts	6	0	0	0.15	1500
Acrylic Panel	1	0	0	21.995	219950
Backboard	1	8.98	8.98	8.98	89800
Shunts	5	0	0	0.02117	211.7
Header pins (10 POS)	2	0.13	0.26	0.05922	592.2
Jumper wires	23	0	0	1.95	16017.86
Raspberry Pi 4	1	75	85	75	750000
Map print	1	6	6	0.39	3900
SD card	1	12.57	12.57	8.8	88000
Fuses	4	0.87	3.56	0.54	5400
Power Supply	1	11.75	11.75	7.89	78900
PCB	1	7.41	7.41	0.45	4551.3

Table 2: Mass Production Costs

Appendix B. Rubric for Degrees of Success.

Rubric for Degrees of Success:

- Full Integration and Functionality (A Grade)
    - All hardware components (PCB, microcontrollers, LEDs, 7-segment display, and power supplies) are integrated and operate seamlessly.
    - Real-time tracking for all four academic bus routes is successfully implemented, with the option to view all routes together or individually (if applicable).
    - Data delay is kept within 20 seconds, providing a highly responsive system.
    - The user interface is intuitive, clear, and easy to interpret, ensuring a positive user experience.
- Partial Integration and Functionality (B Grade)
    - The system integrates the hardware components but with minor issues affecting one or more routes.
    - Real-time tracking is implemented, but with a delay of up to 30 seconds.
    - At least one bus route fails to track, affecting the overall reliability of the system.
    - The visual display provides information but may be difficult for users to interpret quickly or accurately.
- Minimal Functionality (C Grade)
    - The system integrates some hardware components, but significant issues prevent most or all bus routes from being tracked.
    - Real-time tracking is largely unreliable, with delays exceeding 30 seconds, or no real-time data being received.
    - Several or all bus routes fail to track, severely limiting the system's reliability.

- The visual display is unclear or non-functional, making it difficult for users to interpret any information provided.