

Aviator Design Document

David Thoe, Joshua Kim, Zeke Ulrich, Juan Vargas

September 23, 2025

GTA: Zixiao Ma
Professor: Ryan Beasley



Figure 0.0.1: [Caption]

Contents

1	Introduction	9
1.1	Executive Description	9
1.2	User Stories	9
2	Design Requirements	10
2.1	Requirements	10
2.2	Factors Influencing Requirements	11
2.2.1	Public Health, Safety, and Welfare	11
2.2.2	Cultural Factors	11
2.2.3	Social Factors	11
2.2.4	Environmental Factors	11
2.2.5	Economic Factors	11
3	System Overview	12
3.1	System Block Diagram	12
3.2	System Activity Diagram	13
3.3	System Mechanical Design (Extra Credit)	14
3.4	Integration Approach	16
3.5	System Photographs	17
4	Subsystems	19
4.1	Subsystem 1: Processing	20
4.1.1	Subsystem Diagrams	20
4.1.2	Specifications	20
4.1.3	Subsystem Interactions	21
4.1.4	Core ECE Design Tasks	21
4.1.5	Schematics	21
4.1.6	Parts	21
4.1.7	Algorithm	21
4.1.8	Theory of Operation	22
4.1.9	Specifications Measurement	22
4.1.10	Standards	22
4.2	Subsystem 2: Power Management	24
4.2.1	Subsystem Diagrams	24
4.2.2	Specifications	24
4.2.3	Subsystem Interactions	24
4.2.4	Core ECE Design Tasks	24
4.2.5	Schematics	24
4.2.6	Parts	24

4.2.7	Algorithm	24
4.2.8	Theory of Operation	24
4.2.9	Specifications Measurement	25
4.2.10	Standards	25
4.3	Subsystem 3: Text Display & Chassis	28
4.3.1	Subsystem Diagrams	28
4.3.2	Specifications	28
4.3.3	Subsystem Interactions	28
4.3.4	Core ECE Design Tasks	29
4.3.5	Schematics	29
4.3.6	Parts	29
4.3.7	Algorithm	29
4.3.8	Theory of Operation	30
4.3.9	Specifications Measurement	30
4.3.10	Standards	30
4.4	Subsystem 4: Network and Communications	33
4.4.1	Subsystem Diagrams	33
4.4.2	Specifications	33
4.4.3	Subsystem Interactions	33
4.4.4	Core ECE Design Tasks	33
4.4.5	Schematics	33
4.4.6	Parts	33
4.4.7	Algorithm	33
4.4.8	Theory of Operation	33
4.4.9	Specifications Measurement	34
4.4.10	Standards	34
5	PCB Design	37
5.1	PCB Schematics	37
5.2	PCB Layout	38
6	Final Status of Requirements	40
7	Team Structure	41
7.1	Team Member 1	41
7.2	Team Member 2	41
7.3	Team Member 3	42
7.4	Team Member 4	42
8	Bibliography	44
9	Appendices	45

List of Figures

0.0.1 [Caption]	2
3.1.1 System Block Diagram	12
3.2.1 System Activity Diagram	13
3.3.1 System Mechanical Design	15
3.5.1 [Photo Name]	18
4.1.1 Subsystem Block Diagram	20
4.1.2 [Schematic Name]	23
4.2.1 Subsystem Block Diagram	26
4.2.2 [Schematic Name]	27
4.3.1 Subsystem Block Diagram	28
4.3.2 [Schematic Name]	32
4.4.1 Subsystem Block Diagram	35
4.4.2 [Schematic Name]	36
5.1.1 PCB Schematic	37
5.2.1 PCB Layout	39

List of Tables

1 Revision Log 7

Revision Log

Date	Revision	Changes
5/3/2024	v0.1	Initial Release
9/18/2025	v1.0	First Draft

Table 1: Revision Log

Glossary

- **API** Application Programming Interface.

1 Introduction

1.1 Executive Description

Retro nearby flight information display.

1.2 User Stories

The Long-Time Aviation Hobbyist

As a long-time aviation hobbyist who has spent years tracking flights through phone apps, I'm tired of paying for subscriptions just to unlock basic features. I want a device that gives me real-time flight information without hidden costs, while also providing a tactile, nostalgic experience that reminds me of classic aviation boards. By having the Aviator on my desk, I can finally stay connected to the aviation world without feeling like I'm paying a premium for something that should be standard and accessible.

The Casual Aviation Enthusiast

As someone with a general interest in aviation, I don't need a full cockpit-level tracker, but I do want something that feels engaging and easy to use. Standard apps are flat, cluttered, and frankly too much for a novice like myself, but the Aviator project makes flight tracking simple, physical, and fun. I can glance at the board, see arrivals and departures, and feel connected to the aviation scene effortlessly. The setup process was simply plug and play. Additionally, I don't have to pay a dime for the product. For me, it's about accessibility and enjoying aviation in a personal, low-effort, high-impact way.

The Purdue ECE Student

As a Purdue ECE student, I'm drawn to the Aviator not only as a hobby project that I can tinker with, but also as a nod to Purdue's deep aviation legacy. It's inspiring to own a piece of tech that bridges my academic interests in circuits and embedded systems with Purdue's reputation in aerospace. I want a tracker that feels hands-on, customizable, and personal—something that makes me feel part of both my field of study and Purdue's aviation history every time I glance at it. Given the nature of the project and its ability to be completed by an individual excites me, as it gives me the stepping stone I needed to start tracking flights.

2 Design Requirements

2.1 Requirements

1. The device must display accurate information.
2. The display must not interfere with user's well-being by, for example, displaying at excessive luminosity, updating rapidly in a distracting manner, or being excessively bulky.
3. The device must not infringe on any person's reasonable expectation of privacy.
4. The device must be language-agnostic wherever possible.
5. The device must be responsive and intuitive.
6. The device must have robust error handling and recovery.
7. The physical device should be easily replicated with widely available parts.
8. The code for the device must be open-source and well-documented.
9. The device should be as durable and environmentally friendly as possible so as not to contribute to e-waste.
10. The device must not contribute to noise or visual pollution of any space.
11. The device must be energy-efficient.
12. The device must minimize construction and recurring costs.
13. The device must not infringe on right to repair.
14. The device must mount and dismount without damage to vertical surfaces.

2.2 Factors Influencing Requirements

2.2.1 Public Health, Safety, and Welfare

1. User well-being
2. Privacy

2.2.2 Cultural Factors

1. Language differences
2. Ease of use

2.2.3 Social Factors

1. Ease of replication
2. Open-source and documentation

2.2.4 Environmental Factors

1. Environmental friendliness and e-waste
2. Noise and visual pollution
3. Energy efficiency

2.2.5 Economic Factors

1. Cost
2. Repairability

3 System Overview

3.1 System Block Diagram

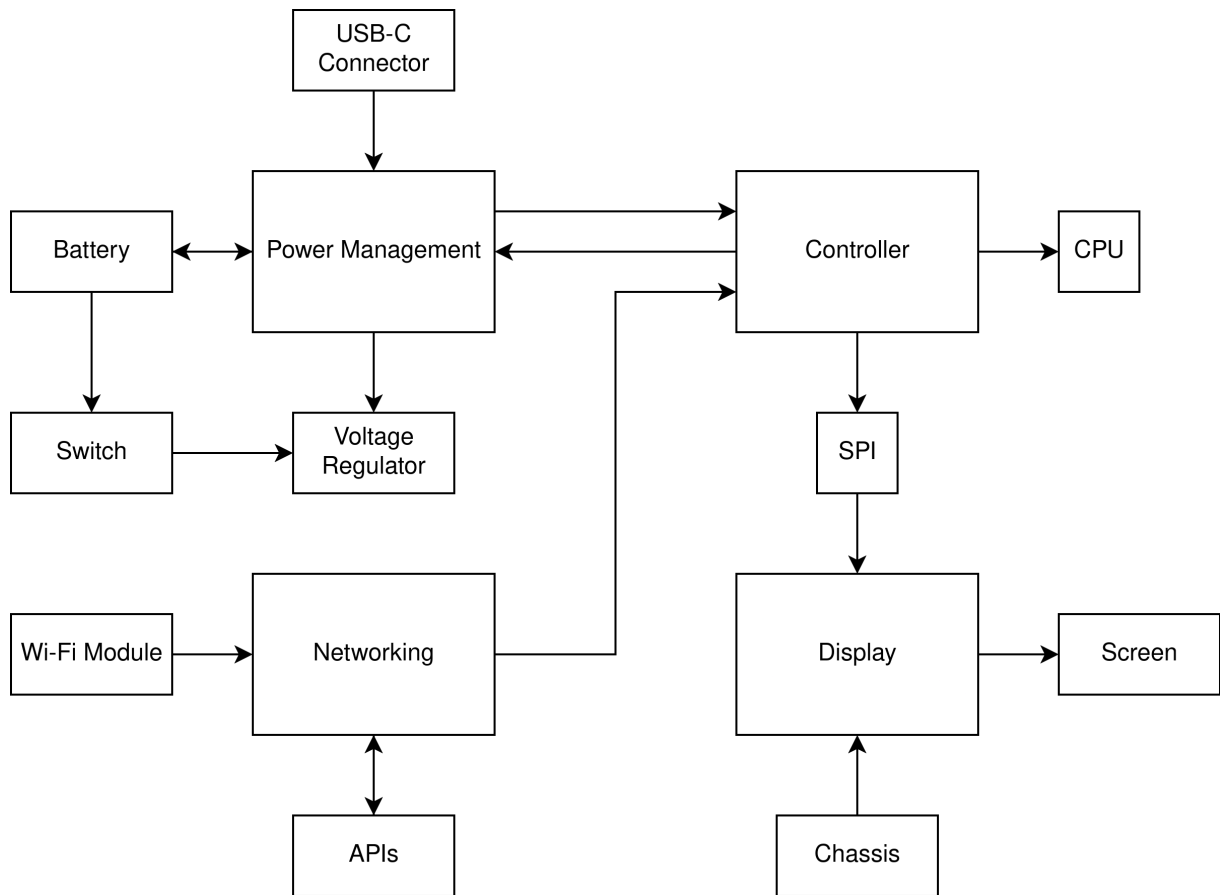


Figure 3.1.1: System Block Diagram

3.2 System Activity Diagram

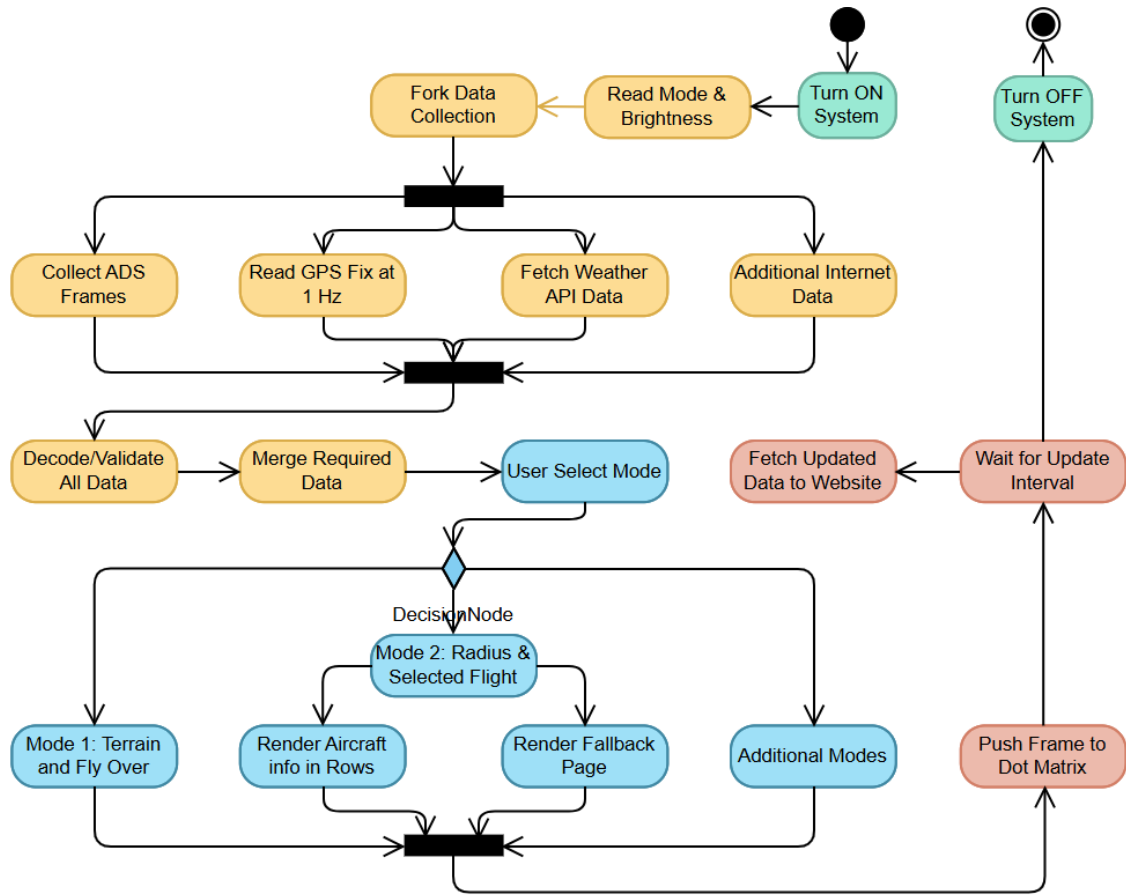


Figure 3.2.1: System Activity Diagram

3.3 System Mechanical Design (Extra Credit)

[DD3+]



Figure 3.3.1: System Mechanical Design

3.4 Integration Approach

[**DD3+**] [Theory behind the system design, with reference to subsystem integration within your system – i.e., explain how it is supposed to work, but not whether it did actually work]
[Type here]

3.5 System Photographs

[**DD3+**] [Photograph of assembled system, intended to highlight user interaction / controls. If system is split into multiple parts, show a composite of more than one photograph with all key user interactions / controls.]



Figure 3.5.1: [Photo Name]

4 Subsystems

4.1 Subsystem 1: Processing

4.1.1 Subsystem Diagrams

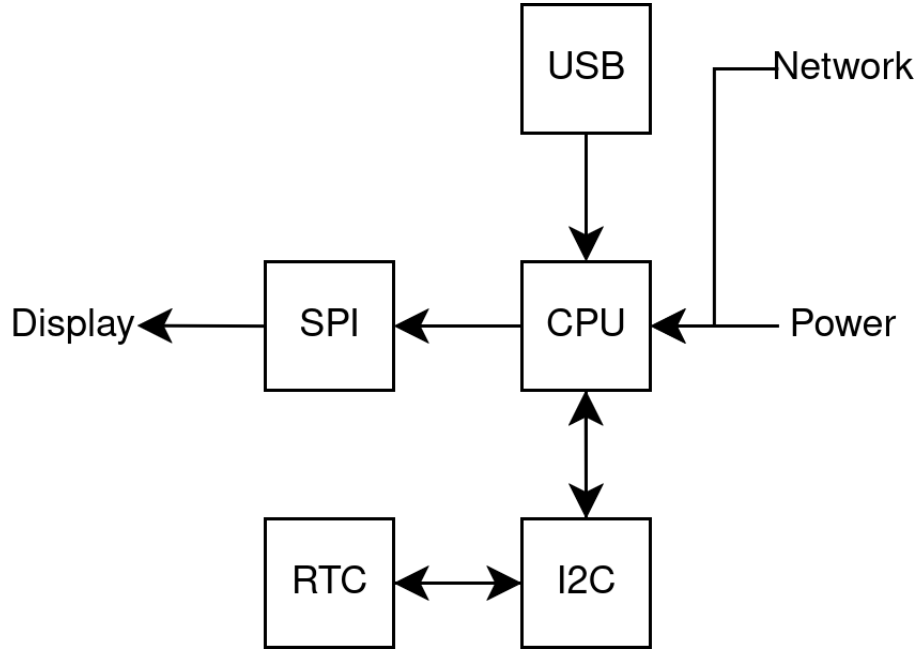


Figure 4.1.1: Subsystem Block Diagram

4.1.2 Specifications

1. Support SPI clock ≥ 20 MHz
2. SPI support up to 40 MHz, full-duplex, DMA capable
3. Display update latency ≤ 16 ms
4. Support configurable API refresh interval between 60 and 300 s
5. API end-to-end fetch latency ≤ 2 s
6. Clock drift $\leq \pm 1 \frac{\text{sec}}{\text{day}}$
7. Operating voltage: 3.3 ± 0.1 V

4.1.3 Subsystem Interactions

The core computer interfaces with all other subsystems. The battery/power management unit supplies it with power. Running processes direct and receive information from the network module. It communicates with the digital dot matrix display via SPI according to display drivers on the controller.

4.1.4 Core ECE Design Tasks

- **ENGR 16100:** Teamwork & project documentation.
- **CS 15900:** Fundamentals of programming.
- **ECE 36200:** PCB design and embedded software development.

4.1.5 Schematics

[Type here **DD2+**]

4.1.6 Parts

- ESP32-S3
- DS3231
- PCB

4.1.7 Algorithm

```
initialize clock , network , display , location

always
    wifi keep alive
    error handling

every minute
    update display time

    make API call for flight data
    parse data

    if battery powered
        check battery voltage
        add battery to data
```

```
add time, weather, flight info to data
convert data to pixel buffer
push pixel buffer to display
```

```
every hour
    sync RTC with network time

    make API call for weather
```

4.1.8 Theory of Operation

[Type here **DD2**+]

4.1.9 Specifications Measurement

[**DD3**+ Every specification here should match the specification above.]

1. [Copy specification here.]
[Explain the specification here. Add photoes if necessary.]

4.1.10 Standards

- **IPC-2221**: Governs PCB trace width, spacing, creepage/clearance, via rules, grounding, etc.
- **IPC-A-610**: Covers soldering quality and workmanship.
- **RFC 5905**: Protocol for syncing ESP time to internet.



Figure 4.1.2: [Schematic Name]

4.2 Subsystem 2: Power Management

4.2.1 Subsystem Diagrams

[DD1+]

4.2.2 Specifications

1. [Type here DD1+]

4.2.3 Subsystem Interactions

[Type here DD1+]

4.2.4 Core ECE Design Tasks

[DD1+ Write tasks and course that helps accomplish that task]

- **ECE xxxxx:** [Type the relationship here.]

4.2.5 Schematics

[Type here DD2+]

4.2.6 Parts

- USB-C 5 V input module
- Li-ion battery pack (3.7 V, 8000 mAh)
- Buck-boost regulator IC (for 3.3 V and 5 V rails)
- Battery charging IC (USB-C PD or simple Li-ion charger).
- Basic protection devices (fuse, MOSFET switch, TVS diode).
- (Optional) Solar panel (5–20 W) + MPPT controller IC.

4.2.7 Algorithm

[Type here DD1+]

4.2.8 Theory of Operation

[Type here DD2+]

4.2.9 Specifications Measurement

[**DD3+** Every specification here should match the specification above.]

1. [Copy specification here.]
[Explain the specification here. Add photoes if necessary.]

4.2.10 Standards

- **USB-C Power Delivery Specification:** ensures safe and standard input power.
- **IEEE 1625:** covers battery system reliability for portable electronics.
- **UL 2054:** safety standard for rechargeable batteries in consumer devices.
- **IEC 61215:** (Optional solar) performance and reliability for PV modules.



Figure 4.2.1: Subsystem Block Diagram



Figure 4.2.2: [Schematic Name]

4.3 Subsystem 3: Text Display & Chassis

4.3.1 Subsystem Diagrams

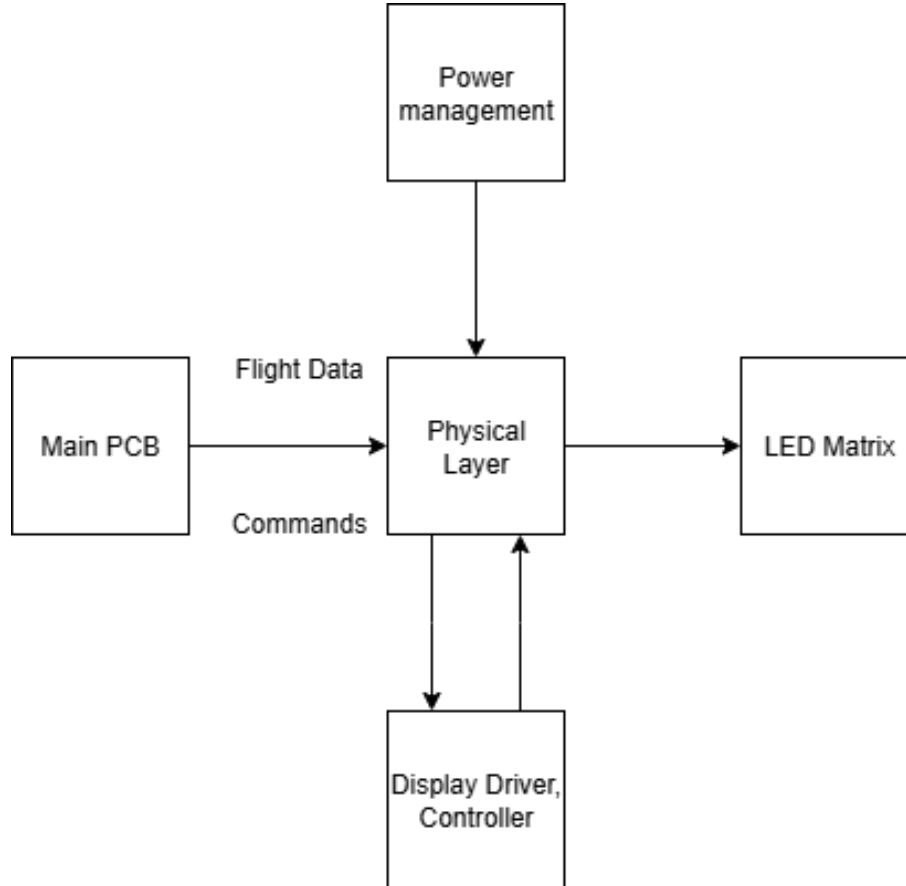


Figure 4.3.1: Subsystem Block Diagram

4.3.2 Specifications

1. Feature a 2056-pixel Dot Display (128 x 256px)
2. Physical dimensions of housing within 5% of 64 x 128 x 256 mm.

4.3.3 Subsystem Interactions

The display subsystem links the main PCB and the LED Matrix. This subsystem will interface with the power management system and the main PCB. It will receive data and

commands from the main PCB, interpret the commands, and display the data.

4.3.4 Core ECE Design Tasks

- **ECE 27000:** Provides a solid foundation in logic circuits. Helps for designing registers, data paths, logic that drives the LED panel.
- **ECE 25500:** Provides a base understanding of transistors, amplifiers, and fundamentals into I-V behavior.
- **ECE 40862:** Teaches valuable STM concepts such as programming, interrupts, and DMA which are needed to drive the display controller efficiently.

4.3.5 Schematics

[Type here **DD2+**]

4.3.6 Parts

- 32 x 64 px LED Matrix (ADAFruit from DigiKey)
- Custom PCB (possibly a hat or extension) for driving display

4.3.7 Algorithm

```
Initialize :
Configure GPIOs (DATA, CLK, LAT, OE, row select lines (A,B,C,D))
Initialize interface for incoming data
Initialize timer for refresh interrupts
Set PWM resolution (e.g., 8-bit)

Main Loop:
while (true):
    if new frame data available from main PCB:
        copy frame buffer into local memory (double buffer optional)

    for each row_index in 0 .. NUMROWS-1:
        select_row(row_index)
        OE = HIGH

        for pwm_bit = 7 downto 0:    // for 8-bit PWM
            for col_index = 0 .. NUMCOLS-1:
                pixel = frame_buffer[row_index][col_index]
```

```

        if pixel.red & (1 << pwm_bit) != 0:
            DATA_R = HIGH
        else:
            DATA_R = LOW
        if pixel.green & (1 << pwm_bit) != 0:
            DATA_G = HIGH
        else:
            DATA_G = LOW
        if pixel.blue & (1 << pwm_bit) != 0:
            DATA_B = HIGH
        else:
            DATA_B = LOW

        pulse(CLK)

        pulse(LAT)
        OE = LOW
        delay(PWMDELAY[pwm_bit])

    repeat indefinitely

```

4.3.8 Theory of Operation

[Type here **DD2+**]

4.3.9 Specifications Measurement

[**DD3+** Every specification here should match the specification above.]

1. [Copy specification here.]
[Explain the specification here. Add photoes if necessary.]

4.3.10 Standards

- **IEC 61010:** Safety for low-voltage electronic equipment; ensures protection against shorts, overcurrent, and handling risks.
- **SPI/I²C:** If MCU/controller communicates with peripheral ICs over standard buses.
- **HUB75:** Standard for 32×64 RGB LED matrices; timing, row multiplexing, and data latching must be followed.

- **JESD51:** Important for high-current LED arrays; ensures heat dissipation and junction temperatures are within safe limits.



Figure 4.3.2: [Schematic Name]

4.4 Subsystem 4: Network and Communications

4.4.1 Subsystem Diagrams

[DD1+]

4.4.2 Specifications

1. [Type here DD1+]

4.4.3 Subsystem Interactions

[Type here DD1+]

4.4.4 Core ECE Design Tasks

[DD1+ Write tasks and course that helps accomplish that task]

- **ECE xxxxx:** [Type the relationship here.]

4.4.5 Schematics

[Type here DD2+]

4.4.6 Parts

- Weather API
- Flight API
- Time Sync API
- GPS/Geolocation API
- Network Interface
- Data/Signal Interface

4.4.7 Algorithm

[Type here DD1+]

4.4.8 Theory of Operation

[Type here DD2+]

4.4.9 Specifications Measurement

[**DD3+** Every specification here should match the specification above.]

1. [Copy specification here.]
[Explain the specification here. Add photos if necessary.]

4.4.10 Standards

- **NMEA 0183:** Used by GPS receivers to format location/time data, ensuring compatibility with processing algorithms.
- **REST/HTTPS:** Secure, standardized communication with weather servers and external data sources.
- **IEEE 802.11 (Wi-Fi) / IEEE 802.3 (Ethernet):** Provides reliable networking for internet-based data exchange (when we add website interference).
- **TCP/IP, Checksum validation:** Ensures robustness in communication so that even non-technical users get accurate, reliable results.



Figure 4.4.1: Subsystem Block Diagram



Figure 4.4.2: [Schematic Name]

5 PCB Design

5.1 PCB Schematics

[DD3+]

4

Figure 5.1.1: PCB Schematic

5.2 PCB Layout

[DD3+]



Figure 5.2.1: PCB Layout

6 Final Status of Requirements

[**DD3+**] [If met, give a detailed explanation of the requirement. If partially met, mention what has been met and a reason for why the complete requirement couldn't be achieved. If not met, give an explanation for why the requirement couldn't be met in the product. Add as many requirements as you had in your earlier design documents here.]

1. Requirement 1: [Copy your requirement above here]
Met: [Explanation]
2. Requirement 2: [Copy your requirement above here]
Partially Met: [Explanation]
3. Requirement 3: [Copy your requirement above here]
Not Met: [Explanation]

7 Team Structure

7.1 Team Member 1



David Thoe

Major: Electrical Engineering

Contact: dtthoe@purdue.edu

Team Role: Team Leader

Bio: In charge of graphics drivers and text display, I will be focused on the final presentation of the information as well as subsystem integration. I will also be paying special attention to the housing of the prototype, aided by CAD and 3D printing, ensuring the product appears polished when complete. At Purdue, I concentrate in Wireless and Optical engineering and participate in the club Autonomous Motorsports Purdue, where we place our fully autonomous go-kart in competition with other universities. In my free time, I work on hobby electrical projects usually related to my computer or decoration.

7.2 Team Member 2



Joshua Kim

Major: Electrical Engineering

Contact: kim3503@purdue.edu

Team Role: Communication Lead

Bio: In my role as Communication Lead, I am responsible for ensuring that communication on our team, with GTA, and the Professor is clear and consistent. I manage communications on updates, facilitate collaboration on progress reporting, and keep everyone engaged

and aligned within the subsystem groups to further our design as a cohesive project. My background in Electrical Engineering enables me to contribute directly to the technical aspects of the project and also help translate technical details into clear explanations for our team and other stakeholders. Balancing the communication and development of the subsystem, I can keep both the engineering work and project coordination rolling smoothly.

7.3 Team Member 3



Zeke Ulrich

Major: Computer Engineering

Contact: pulrich@purdue.edu

Team Role: Treasurer

Bio: Zeke is the processing and PCB design specialist. At Purdue, he belongs to the Marine Corp Officer Candidate Program, Eta Kappa Nu, Tau Beta Pi, and Purdue's Effective Altruism community. Outside Purdue, he is president of the nonprofit DuelGood and works for the government in cybersecurity. In the future he hopes to study international relations as a Truman scholar, start a family, and volunteer as a firefighter. He enjoys athletics and spending time with his friends.

7.4 Team Member 4



Juan Vargas

Major: Electrical Engineering

Contact: varga105@purdue.edu

Team Role: Facilitator

Bio: As the Facilitator, my primary role is to keep our team organized, collaborative, and on track throughout the project. I focus on making sure discussions are productive, tasks are clearly divided, and deadlines are met without overwhelming any single member. By bridging technical conversations and helping resolve roadblocks quickly, I ensure that progress stays steady and balanced across subsystems. Alongside this coordination, I contribute to the technical development by assisting with software and system integration specially in the network/communication part, ensuring our device works as intended while maintaining the retro, aviation-inspired feel that defines Aviator.

8 Bibliography

[Here are some examples. IEEE format can be found on [Purdue OWL](#).]

References

- [1] “Data Platform - Open Power System data,” Apr. 15, 2020. https://data.open-power-system-data.org/household_data/
- [2] Author, ”Title,” *Journal*, volume, number, page range, month year, DOI.
- [3] Author. ”Page.”Website. URL(accessed month day,year)

9 Appendices

[This section is mainly designed for code. You can directly generate a somewhat decent display of your code file or psuedo code by using the template provided below. You can have as many appendix as you want. In the document, you can refer to the code posted here instead of pasting the whole code in the body.]