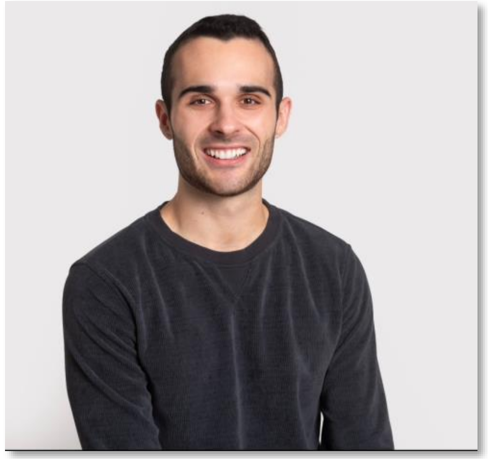
RACE

(Remote Access Control for Electric) Cars

Critical Design Review – Group 30

The Team



Austin Silva
Electrical Engineering

Michael Mallette
Electrical Engineering

Jesvany Colon
Electrical Engineering

Sean Chaney
Computer Engineering

Motivation & Background

- For this project, we want to merge the thrill of traditional RC car racing with online accessibility into a competitive gaming system where users can race against one another remotely.
- As avid gamers, this is intended to be a medium between the physical world and virtual space, offering an experience unique to users since it blends elements of physical RC car racing and online video game racing.
- We aim to create an engaging platform and complete racing system that can connect people of common interests together, enjoying a fun racing experience.

Goals & Objectives

Goals

- Develop an immersive internet-controlled (IC) car racing experience.
- Enable real-time first-person view (FPV) video streaming.
- Implement lap and time tracking for accurate race results.
- Display race times on online leaderboards.

Objectives

- Build two IC car models with driving motors, steering motors, and microcontrollers.
- Integrate FPV cameras for real-time streaming and an immersive user experience.
- Develop an automated system for accurate lap tracking and race timing.
- Create a user-friendly interface for remote control and live video access.
- Focus on accessibility for a seamless and enjoyable user experience.
- Optimize performance for low latency and multi-user functionality.

Project Features

- **Internet-Based Remote Control:** Users can control IC cars via a low-latency web interface.

- **Web Interface:** Provides display of control inputs, live video, and race stats (lap count/time and leaderboard) in a browser.

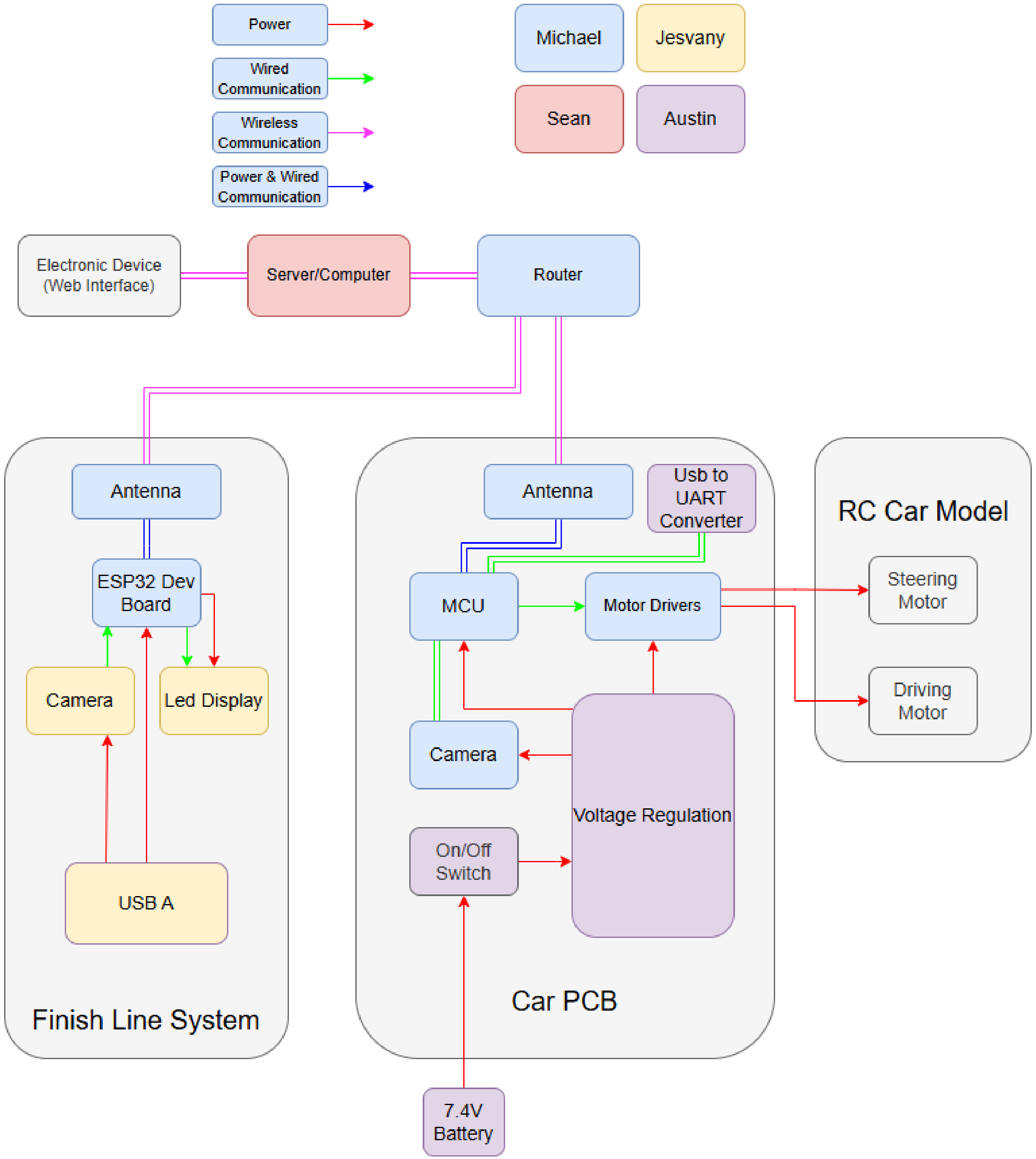
- **FPV Camera System:** Real-time video streaming that offers an immersive first-person racing experience.

- **Multi-User Support:** A scalable server enables simultaneous racing with smooth performance.

- **Finish Line Detection:** Camera sensor that can distinguish cars based on their color, to determine winners.

- **Lap Tracking:** Automated sensor to ensure accurate recording of race results.

- **Custom PCB:** Integrates microcontroller-based command execution for motors, steering, and sensors.



Hardware Block Diagram

Engineering Specifications

Specifications	
Car Size	≤6”x12”
Car Control Delay	≤1.5 seconds
Car Chassis Maintainability	Chassis lasts for 100 runtime hours
Car Battery Life	≥15 minutes
Video FPS	10-30 FPS
Video Resolution	426x240
Total Cost of 2 Cars	≤$200
Connectivity Distance	10-30 ft
Finish Line Detection Accuracy	≥75%

Car's System Hardware Design

MCU Selection

Specification	ESP32 Wrover	ESP8266	Raspberry Pi Pico 2 W	Arduino Nano
Microcontroller	Espressif (Tensilica Xtensa LX6)	Espressif (Tensilica Xtensa L106)	RP2040	ATmega328
Camera Ribbon Connector	Yes	No	Yes	No
Number of Cores	2	1	2	1
Operating Voltage	3.3V	3.3V	3.3V	5V
Clock Speed	240 MHz	80 MHz	133 MHz	16 MHz
Flash Memory	8MB/16MB	4MB	2MB	32 KB
SRAM	520 KB + 8MB PSRAM	64 KB	264 KB	2 KB
Wi-Fi	Yes	Yes	No	No
Bluetooth	Yes (v4.2)	No	No	No
GPIO Pins	30	17	26	14
PWM Pins	16	8	16	6
ADC Channels	18	1	3	8
Power Consumption	Moderate to Low	Low	Low	Low
Cost (Amazon)	10+	8	10	6

- Deciding Factors,
  - Camera Ribbon Connection
  - Most Number of Pins
  - High Clock Speed
  - High RAM

Car Selection

- We gathered and tested three different cars
- Good voltage range
  - 2V-5V
- Slower than other vehicles
  - Latency less noticeable
  - Gives user better control

Motor Driver Selection

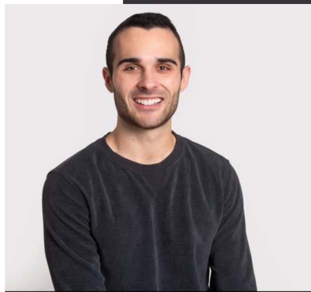
Motor Driver	Voltage Range	Current Handling	Features	Pros	Cons
L298H	4.5V - 46V	Up to 2A per channel	Dual H-bridge, simple and robust, suitable for basic applications	Widely available, simple to use, handles higher voltages	Large size, less efficient due to older technology, heat dissipation issues
DRV8871	6.5V - 45V	3.6A peak, 2A continuous	Single-channel H-bridge, efficient, built-in protection features	High current handling, good for higher power motors, reliable protection	Single channel, may need two for dual motor control
DRV8833	2.7V - 10.8V	Up to 1.5A per channel	Dual H-bridge, compact, efficient, capable of driving two motors	Compact size, efficient power usage, suitable for low voltage motors	Lower current handling compared to others, limited to lower voltage applications

- Deciding Factors
  - Small Size
  - Within motor voltage range

Camera Technologies

Features	FPV Cameras	Onboard Action Cameras	Webcams	IoT Cameras
Video Transmission	Analog RF	Digital via Wi-Fi or USB	Digital via USB or Wi-Fi	Digital via Wi-Fi
Latency	Very Low	Low-Medium	Medium	Medium
Field of View	Wide FOV, Either 120° or 170°	Variable FOV, Ranges from 90° to 170°	Small FOV, Ranges from 60° to 90°	Variable FOV, Ranges from 60° to 160°
Resolution	Low, 480p-720p	High, 1080p-4K	Medium, 720p-1080p	Medium, 720p-1080p
Image Sensor	CMOS or CCD	CMOS	CMOS or CCD	CMOS
Size	Compact	Bulkiest	Medium	Compact
Weight	Usually < 30g	100g or More	50g-100g	10g-100g
Cost	Range of $20-$100	Range of $100-$400+	Range of $30-$150	Range of $10-$100

- Deciding Factors,
  - Transmission for live video streaming through Wi-Fi connection
  - FOV for ideal first-person FPV
  - Compact size
  - Low Cost
  - Compatibility with Microcontrollers



Camera Selection

Features	OV2640	OV7670
Resolution	2 MP (1600x1200)	VGA (640x480)
Frame Rate	Adjustable, 30 fps recommended	Typically 15 fps, but can go to 30 fps
Image Sensor Type	CMOS	CMOS
Interface	I2C	I2C
Field of View	Up to 160° FOV	Smaller FOV
Compatibility	Directly Compatible with ESP32	Compatible, does require additional interfacing
Power Consumption	Higher	Lower
Ease of Integration	More Support	Less Support
Cost	\$6-\$9	\$3-\$6

- Deciding Factors,
    - Higher Resolution for image capture
    - Better Quality for video streaming
    - Smoother Integration with ESP32

Power Consumption

Device	Voltage (V)	Current (mA)	Peak Power (W)
MCU	3.3	150-300	~1
Camera	3.3	100-150	~0.5
Driving Motor	5	500	2.5
Steering Motor	5	800	4
Dual Motor Driver	Range of 2.7-10.8	1500 (Max per Channel)	6.5 (Max at 5V Input)

- Our IC (Internet-Controlled) Cars require a suitable 7.4 V Battery with significant capacity (mAh) to efficiently and effectively power all the devices for our cars' desired battery lifetime.
- Our IC Cars require 3.3 V and 5 V Voltage Regulators to step-down the 7.4 V for sufficient power delivery to their respective devices.

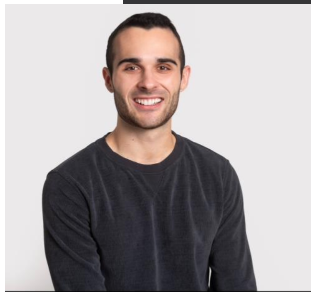
Power Supply

$$\text{Battery Runtime} = \frac{\text{Battery Rated Capacity (in mAh)}}{\text{Load Current (in mA)}}$$

- Due to the properties of lithium-based batteries and their effectiveness in RC car applications, we decided to utilize a Li-Po battery product as the power source for the cars.

Features	URGENEX 2S	Turnigy 2S	Gens Ace 2S
Voltage	7.4 V	7.4 V	7.4 V
Capacity	1100 mAh	1300 mAh	1200 mAh
C Rating	35	30	30
Weight	52g	76g	60g
Dimensions	60x34x16 mm	76x34x17 mm	125x20x12 mm
Quantity	2	1	1
Cost	$24 (including charger)	$9	$22

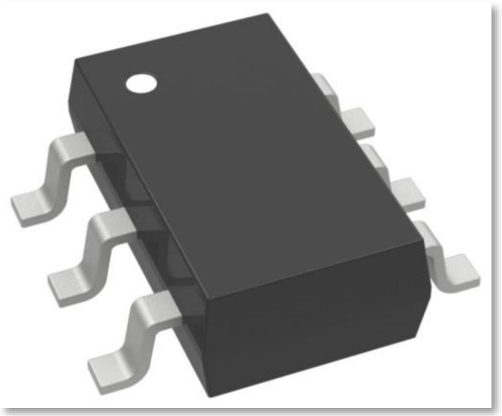
- 1100 mAh Battery Capacity was determined to significantly exceed our desired battery lifetime for more than 15 minutes of runtime.

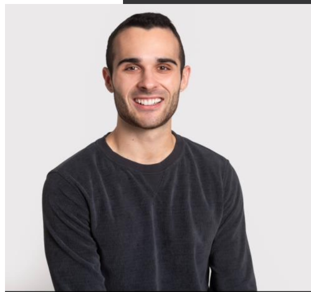


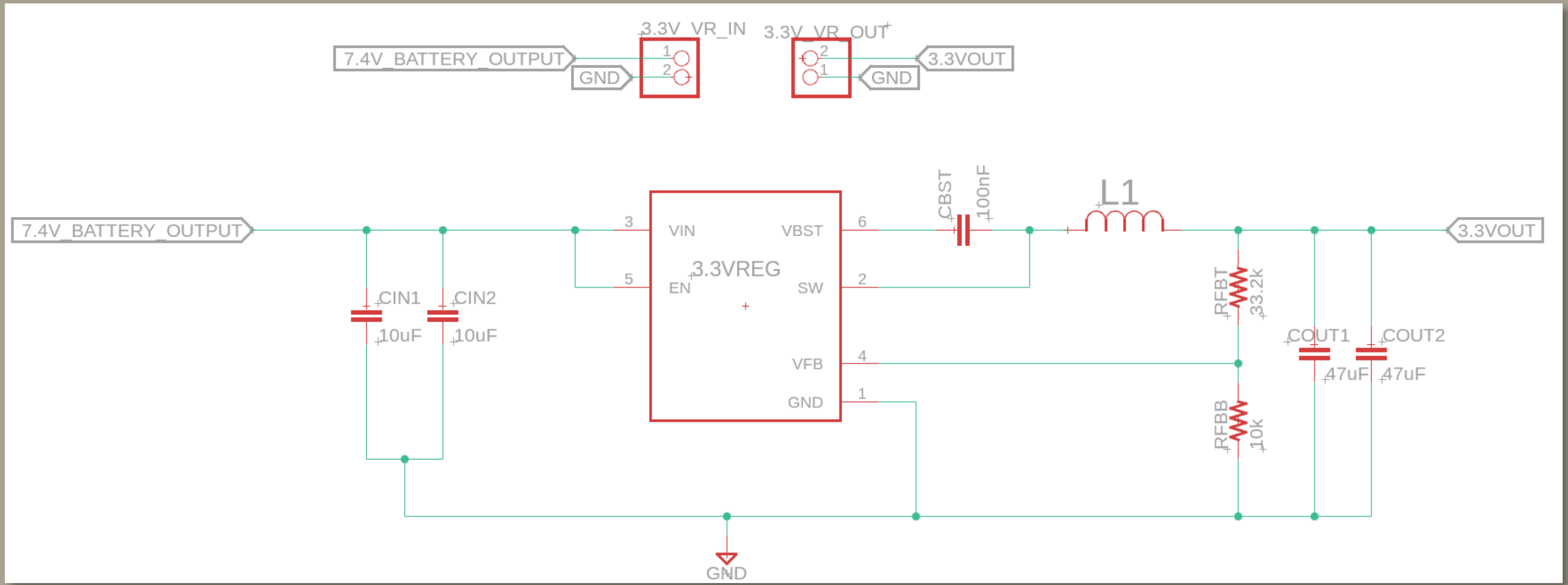
Power Distribution

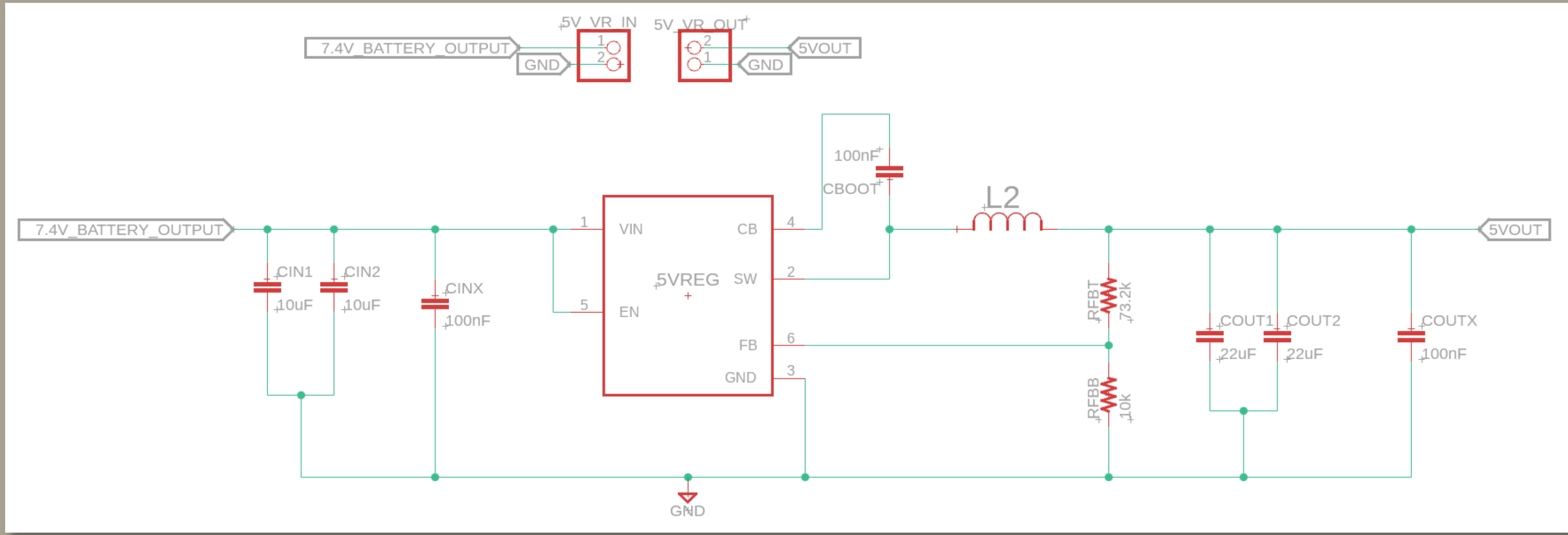
Specifications	Linear Regulators	Switching Regulators
Efficiency	Moderate	High
Design Complexity	Lower complexity with fewer components	Higher complexity with additional components
Thermal Performance	Poor due to more heat generation	Better due to less heat generation
Transient Response	Faster	Slower
Noise	Less	More (due to switching)
Input Voltage	Limited, needs input to be greater than output	Wide range, can step-up or step-down
Output Voltage	Fixed or Adjustable	Flexible
Cost	Lower	Higher

Features	TPS62142	TPS62143	TPS561208	TPS563231
Input Voltage	3-17 V	3-17 V	4.5-17 V	4.5-17 V
Output Voltage	3.3 V	5 V	0.76-7 V	0.6-7 V
Max Current	2 A	2 A	1 A	3 A
Efficiency	93.1%	93.1 %	93.6%	92.2%
Switching Frequency	1250-kHz to 2500-kHz	1250-kHz to 2500-kHz	580-kHz	600-kHz
Topology	Buck	Buck	Buck	Buck
Price	$0.57	$0.57	$0.16	$0.13

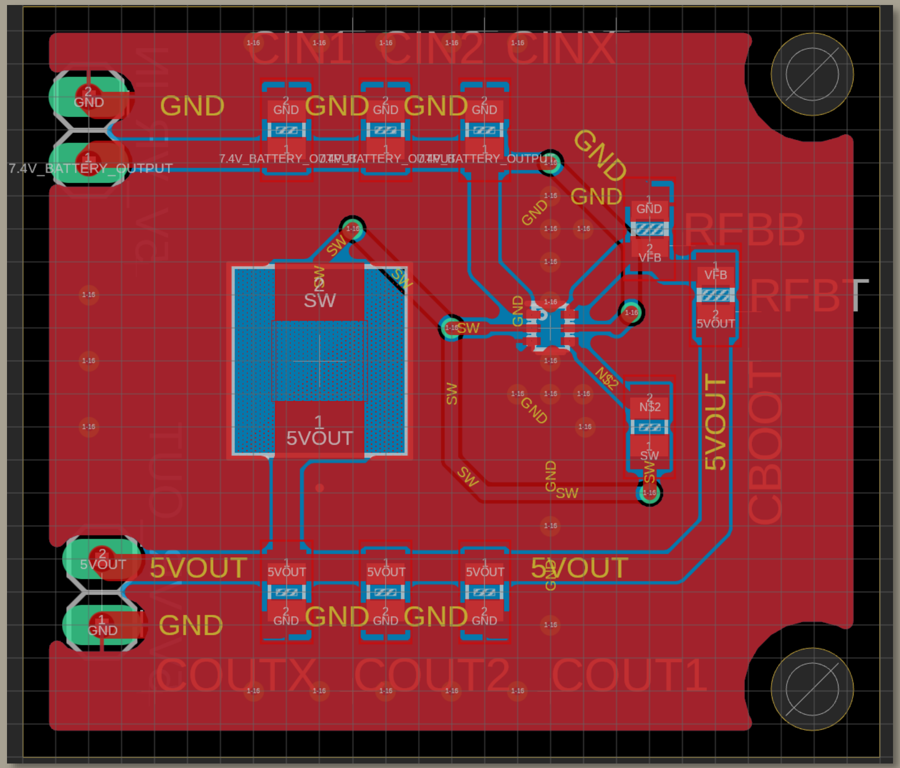


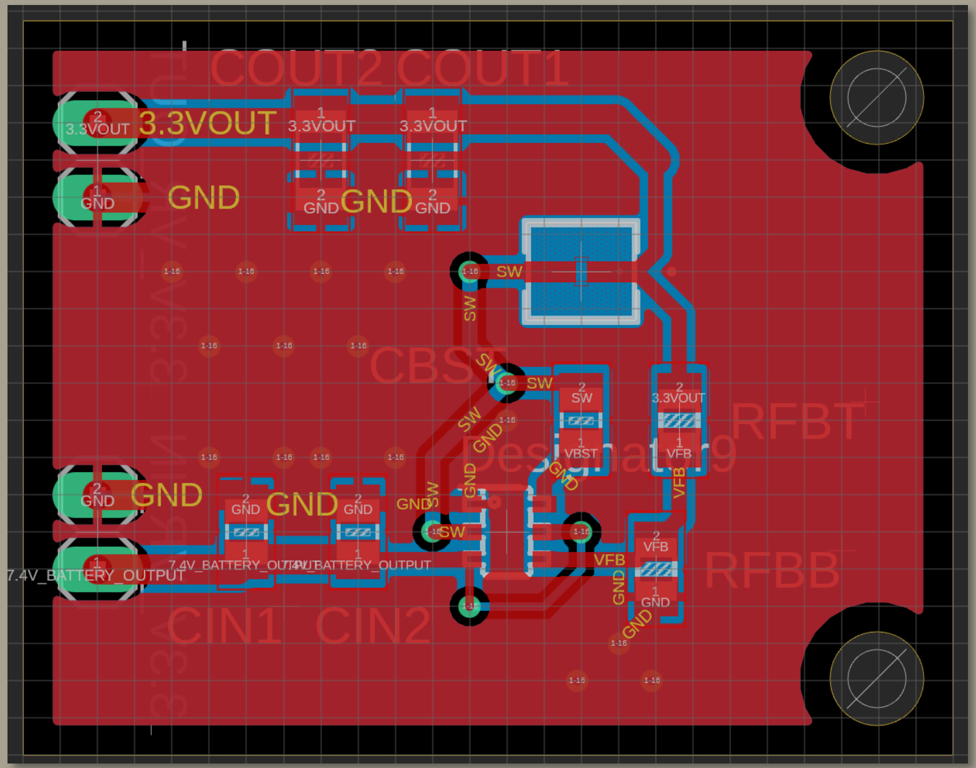




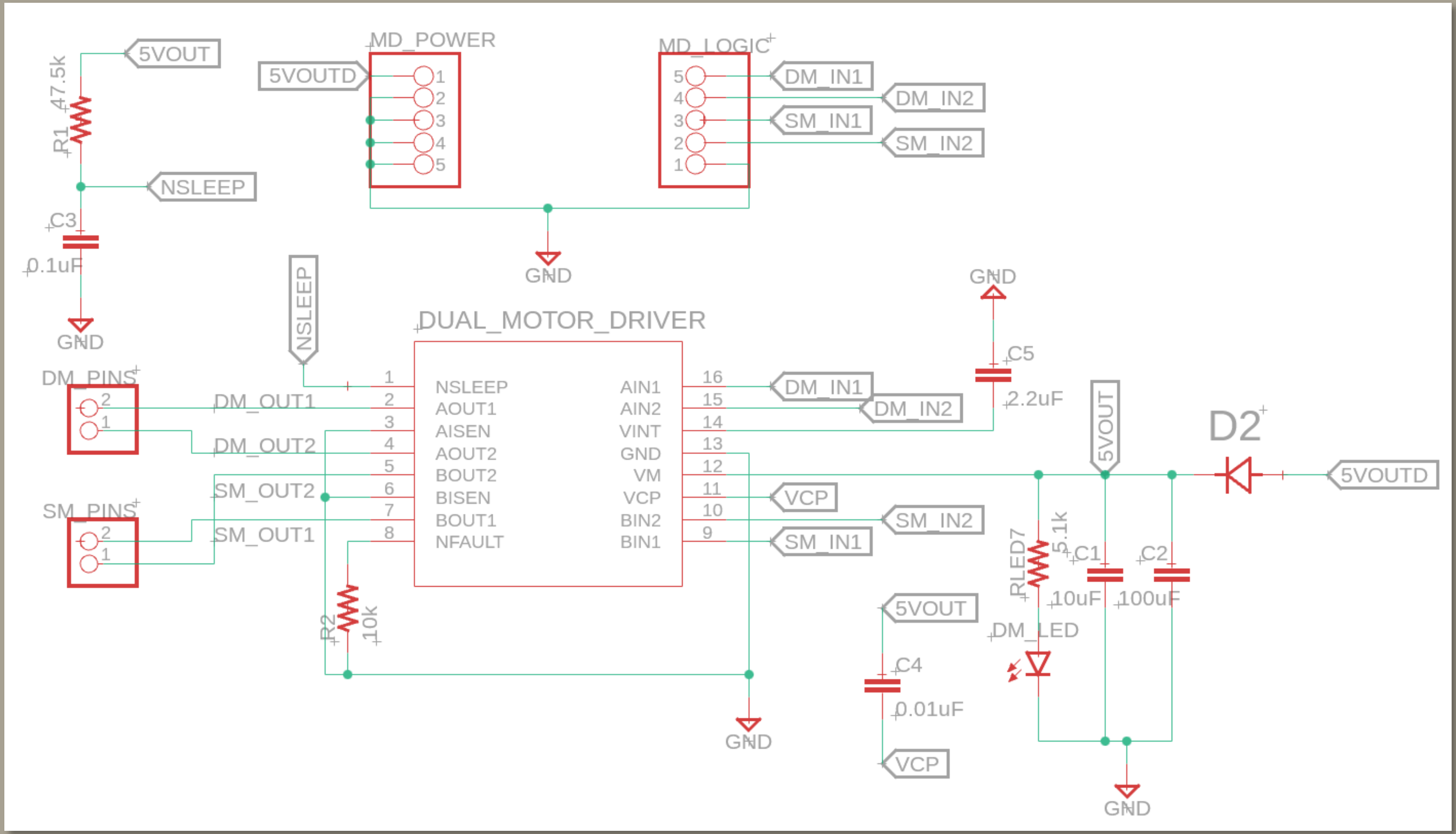


Voltage Regulation Schematics

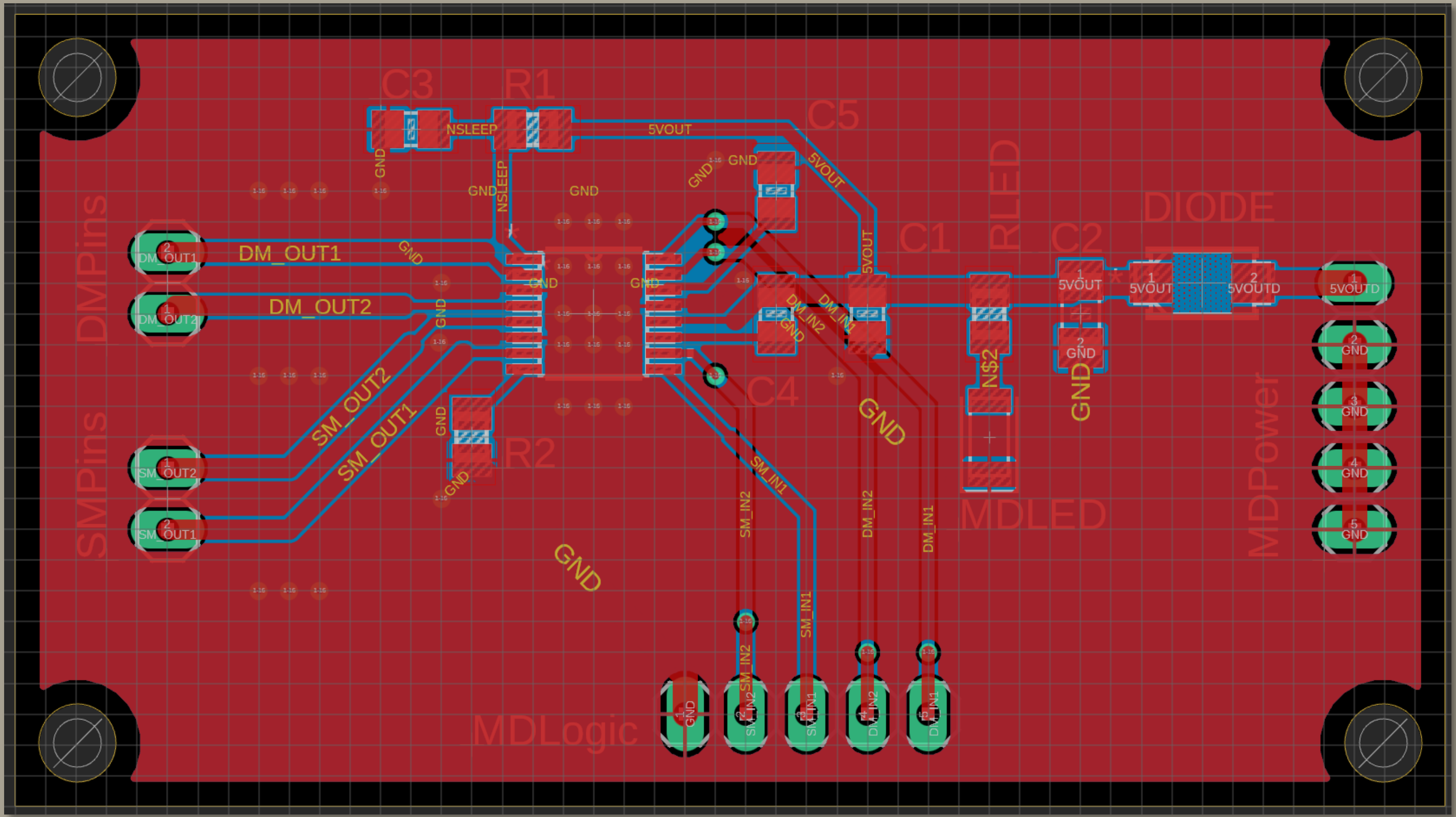




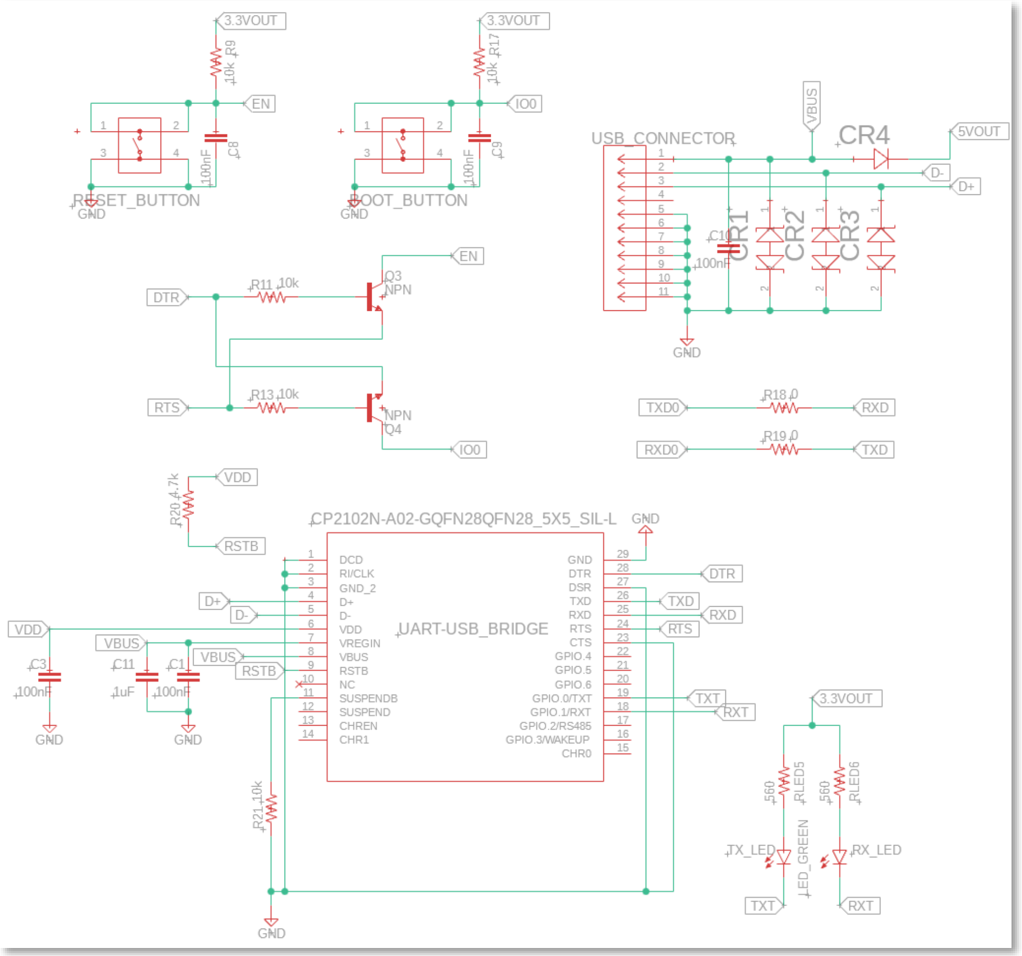
Voltage Regulation Board Layouts

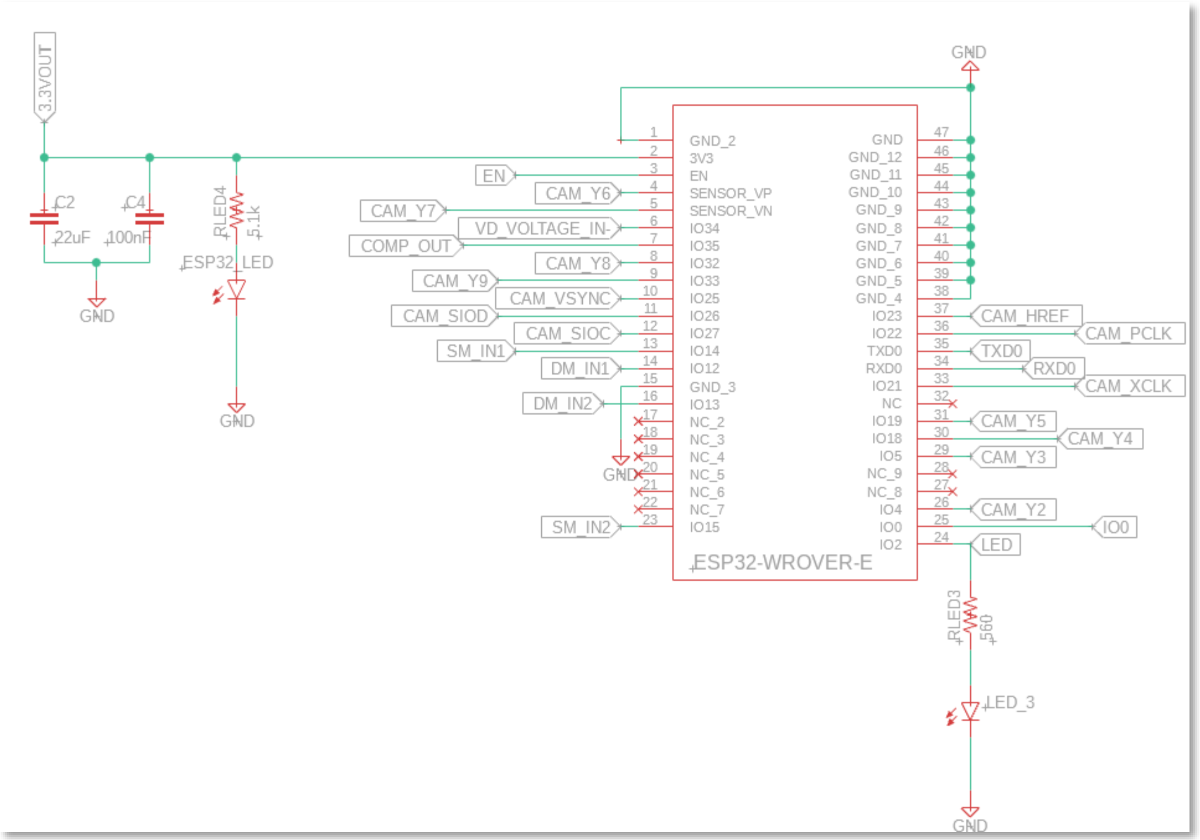


Motor Driver Schematic

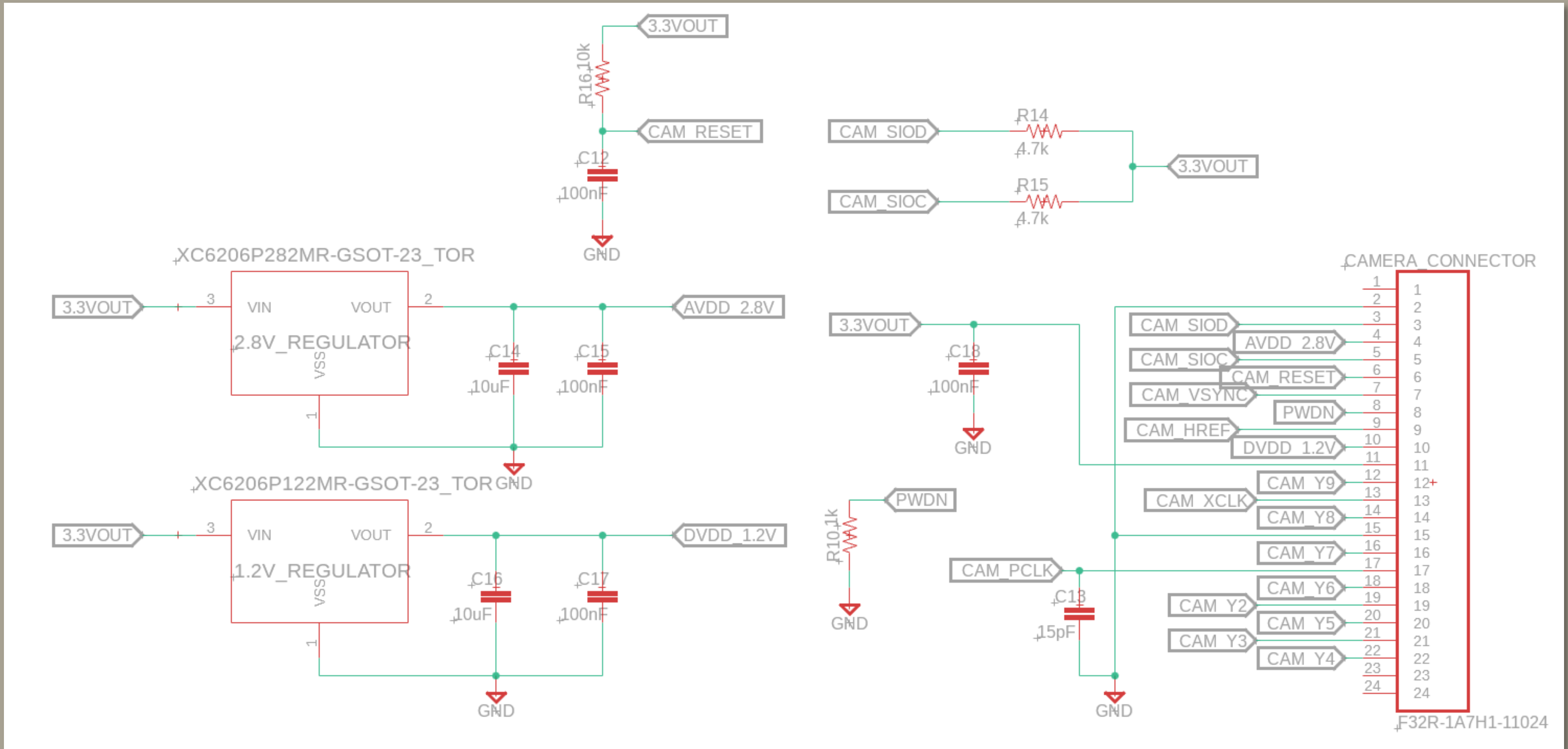


Motor Driver Board Layout

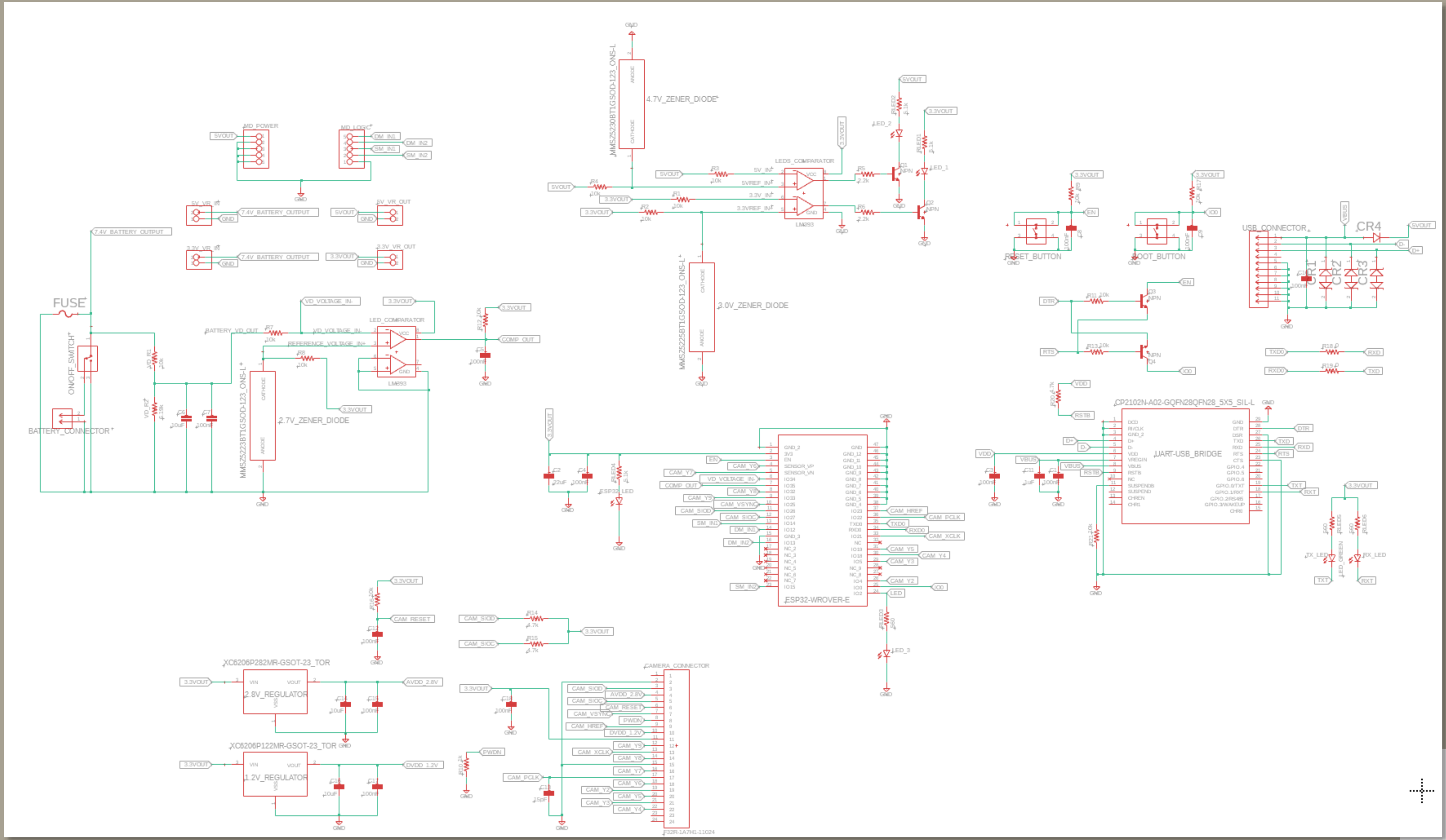




MCU Schematic

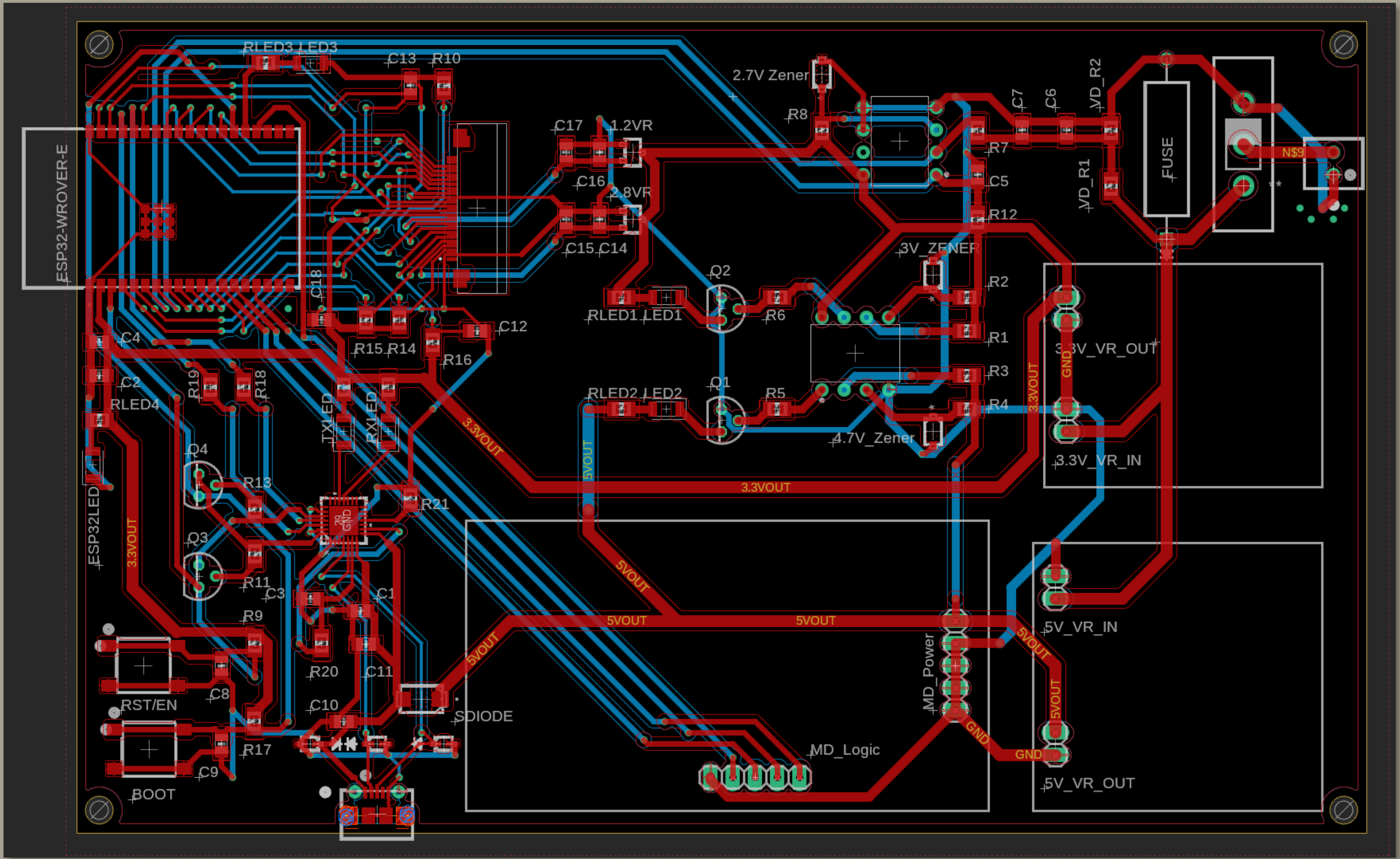


Camera Schematic



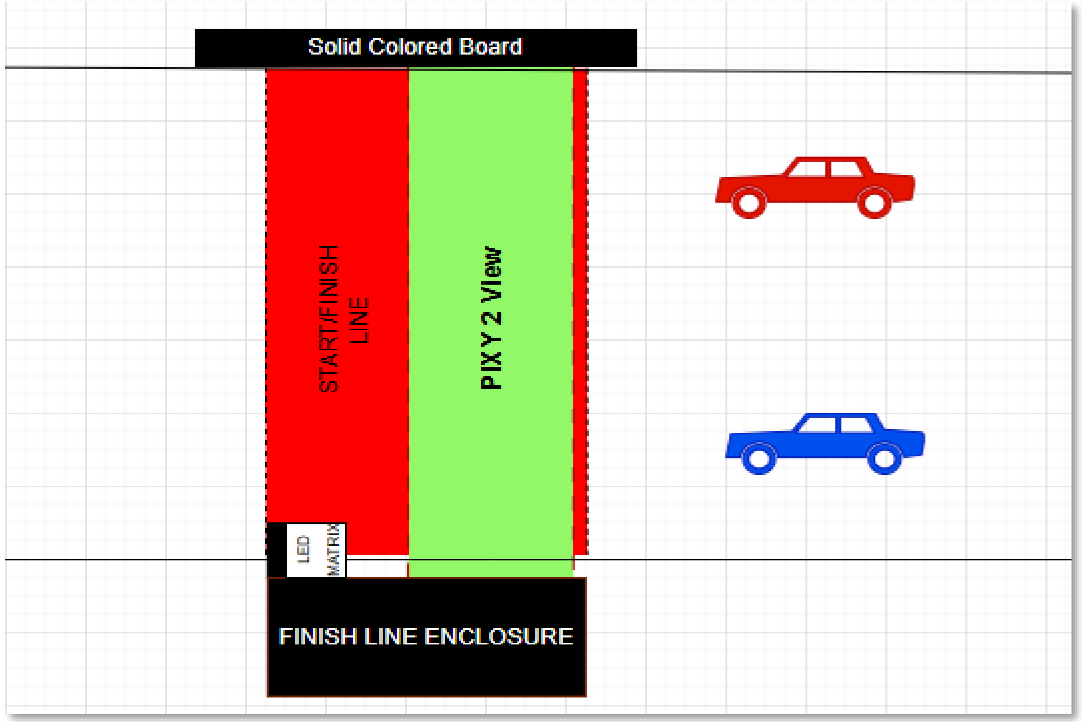
Overall Schematic

Main PCB Board Layout



Finish Line's System Hardware Design

Finish Line Hardware Overview



The finish line will consist of 3 main hardware parts:

- Camera Sensor
- MCU/Development Board
- LED Matrix

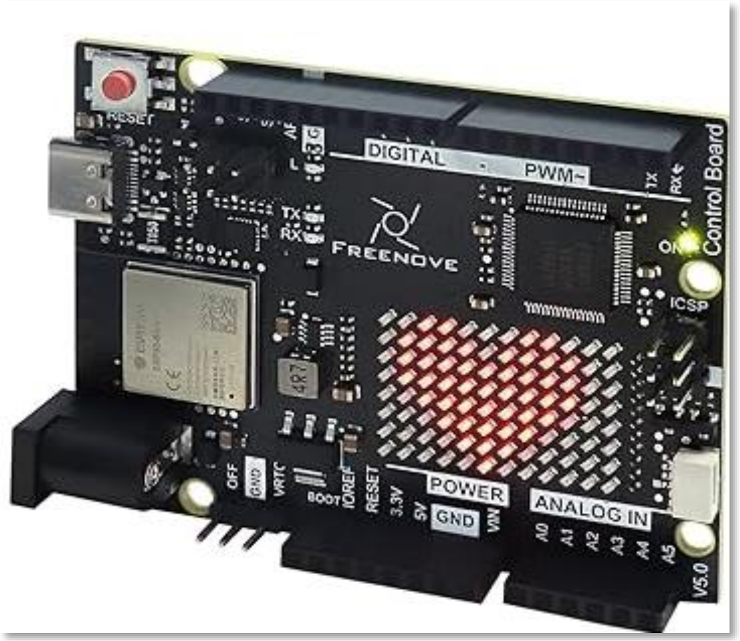
- The camera sensor will be able to distinguish between the two cars.

- The MCU will be able to distribute power between the other peripherals and communicate with our server.

- LED Matrix will act as an active display for the race countdown sequence, lap tracking, and announcement of the winning car.

Development Board Selection

- Original plan was to use the ESP32-WROOM-32 but it would not communicate with the PIXY 2 via SPI communication.
- We ended up deciding on the Freenove Control Board V5 Rev4 Wi-Fi due to timing in having ESP32 ready,
  - This is a cheaper but nearly identical version of the Arduino Uno Rev4 Wi-Fi.
  - This board was selected due to it being the most compatible MCU with the Pixy 2.



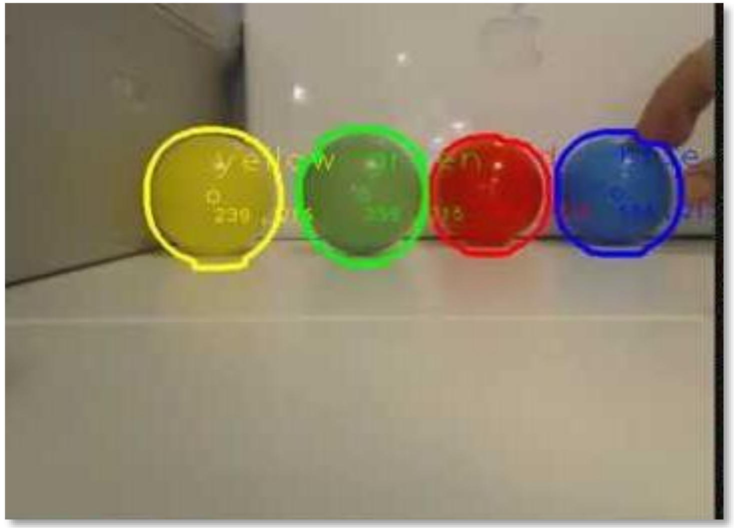
Development Board (Continued)

Features	ESP32-WROOM-32	Freenove Control Board V5 Rev4 Wi-Fi
Microcontroller	Dual-core Xtensa® 32-bit LX6	Renesas RA4M1 Arm Cortex-M4
Clock Speed	Up to 240 MHz	48 MHz
Flash Memory	4 MB (32 Mbits)	256 KB
SRAM	520KB	32KB
Wireless Connectivity	Wi-Fi 802.11 b/g/n, Bluetooth v4.2 BR/EDR and BLE	Wi-Fi via onboard ESP32-S3 module
GPIO Pins	34 programmable GPIOs	14 digital I/O pins, 6 analog input pins
Interfaces	UART, SPI, I2C, I2S, SDIO, Ethernet, capacitive touch sensors, ADC, DAC	UART, I2C, SPI, CAN
Operating Voltage	3.0 V ~ 3.6 V	5 V (ESP32-S3 operates at 3.3 V)
Dimensions	18 mm x 25.5 mm x 3.1 mm	69 mm x 54 mm
Additional Features	Integrated PCB antenna, supports secure boot and flash encryption	Onboard ESP32-S3 module for Wi-Fi, 12x8 red LED matrix, detailed tutorials and example projects provided
Price	$10.00	$19.99

Camera Technology Research

- Marker Detection

- Color Detection



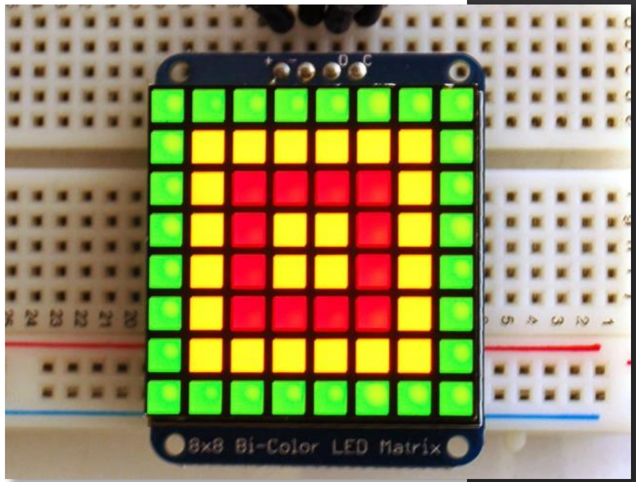
- Line Detection

Camera Selection

Features	PIXY2	ESP32-CAM	JeVois A33
Microcontroller/Processor	NXP LPC4330, dual-core ARM Cortex-M4/M0	ESP32-S, dual-core Xtensa LX6, 240 MHz	Allwinner A33, quad-core Cortex-A7, 1.34 GHz
RAM	264 KB	520 KB SRAM	512 MB DDR3
Flash Memory	2 MB	4 MB PSRAM	MicroSD for external storage
Camera Sensor	Custom color sensor	OV2640, 2 MP	1.3 MP OmniVision (supports other modules)
Image Resolution	640x400 (max)	1600x1200 (UXGA)	1280x1024 (SXGA), adjustable
Frame Rate	Up to 60 fps	30 fps (at lower resolutions)	Up to 120 fps (QVGA)
Connectivity	USB, SPI, I2C, UART	Wi-Fi, Bluetooth, UART	USB, UART
Machine Vision Features	Object tracking, color detection, line tracking	Basic face detection, streaming, custom algorithms	Object tracking, face detection, custom vision algorithms
Onboard Storage	None (depends on host system)	MicroSD card slot (up to 4 GB)	MicroSD card for data storage
Programming Environment	PixyMon, Arduino libraries	Arduino IDE, ESP-IDF, MicroPython	JeVois SDK, OpenCV, Python, C++
Power Requirements	5V (via USB or connector)	3.3V or 5V (via pins)	5V (via USB)
Size	45.5 x 32 mm	27 x 40.5 mm	25 x 25 x 30 mm
Price Range	~$60-70	~$10	~$50-60

LED Matrix Selection

Features	Adafruit Bicolor LED Square Pixel Matrix with I2C Backpack	Adafruit Small 1.2" 8x8 Bi-Color (Red/Green) Square LED Matrix	MAX7219 8x8 LED Matrix
LED Matrix Size	8x8 (64 LEDs)	8x8 (64 LEDs)	8x8 (64 LEDs)
LED Color(s)	Red ,Yellow, Green (Bi-Color)	Red,Yellow, Green (Bi-Color)	Single Color (usually Red)
Driver Chip	HT16K33 (I2C)	External driver (requires an I2C backpack or controller)	MAX7219 (SPI)
Interface	I2C (with Qwiic/STEMMA QT connectors)	Requires external driver (typically I2C)	SPI
Voltage	3.3V to 5V	3.3V to 5V	5V
Dimensions	1.5" x 1.5" (38mm x 38mm)	1.2" x 1.2" (30mm x 30mm)	2.5" x 1.5" (64mm x 38mm)
Microcontroller Compatibility	ESP32, Arduino, Raspberry Pi	ESP32, Arduino, Raspberry Pi	ESP32, Arduino, Raspberry Pi
Control Pins	2 (SDA, SCL)	Depends on the external driver (typically I2C)	3 (SPI: MOSI, CLK, CS)
Multiple Matrix Cascading	Yes, via I2C	Yes, with external driver support	Yes, via SPI
Power Consumption	~20-30mA (depends on brightness)	~30-40mA (depends on brightness)	~50mA (depends on brightness and LEDs lit)
Price	$15.95	$7.95	$8.39

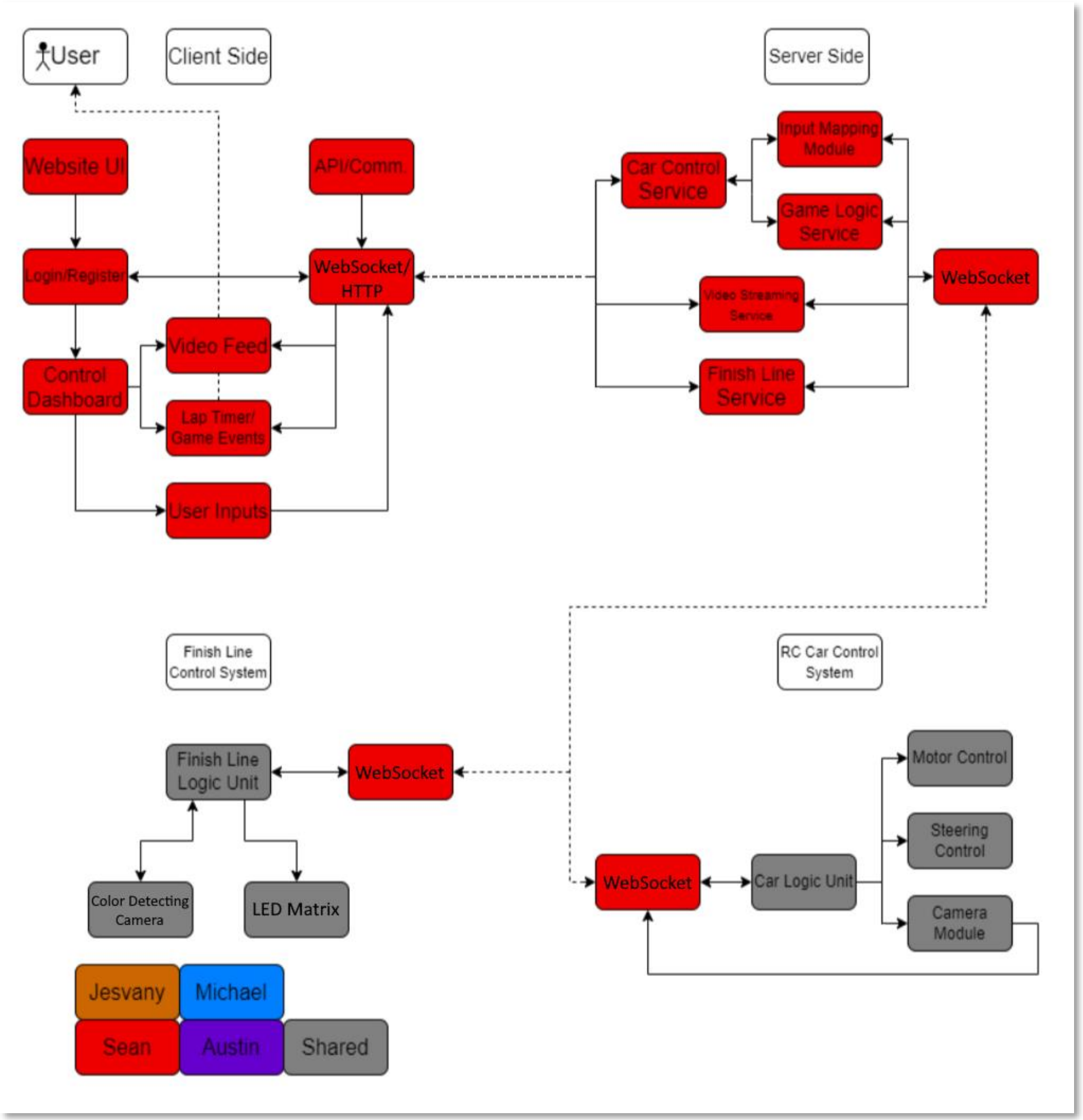


RACE System Communication Network

Communication Protocols Selection

Mode	Type	Strengths	Weaknesses	Best Use in Project
Wi-Fi	Wireless (LAN)	High-speed, wide coverage, stable for data and control	Interference, limited range	Connecting RC car to local server and internet
WebSockets	Real-time communication	Low-latency, full-duplex, persistent connection	Requires WebSocket server setup	Sending real-time control commands
RTSP	Media streaming protocol	Low-latency video streaming, scalable	Not for bidirectional control	Streaming live video feed from RC car
WebRTC	P2P real-time communication	P2P, low-latency, supports video, audio, and data	More complex setup, NAT traversal issues	Live video and data streaming between car and website
MQTT	IoT messaging protocol	Lightweight, low bandwidth, publish/ subscribe model	Not suited for large data transfers (e.g., video)	Transmitting small data like sensor readings
Bluetooth	Short-range wireless	Low power consumption, secure	Limited range, lower data rates	Short-range control or monitoring without internet

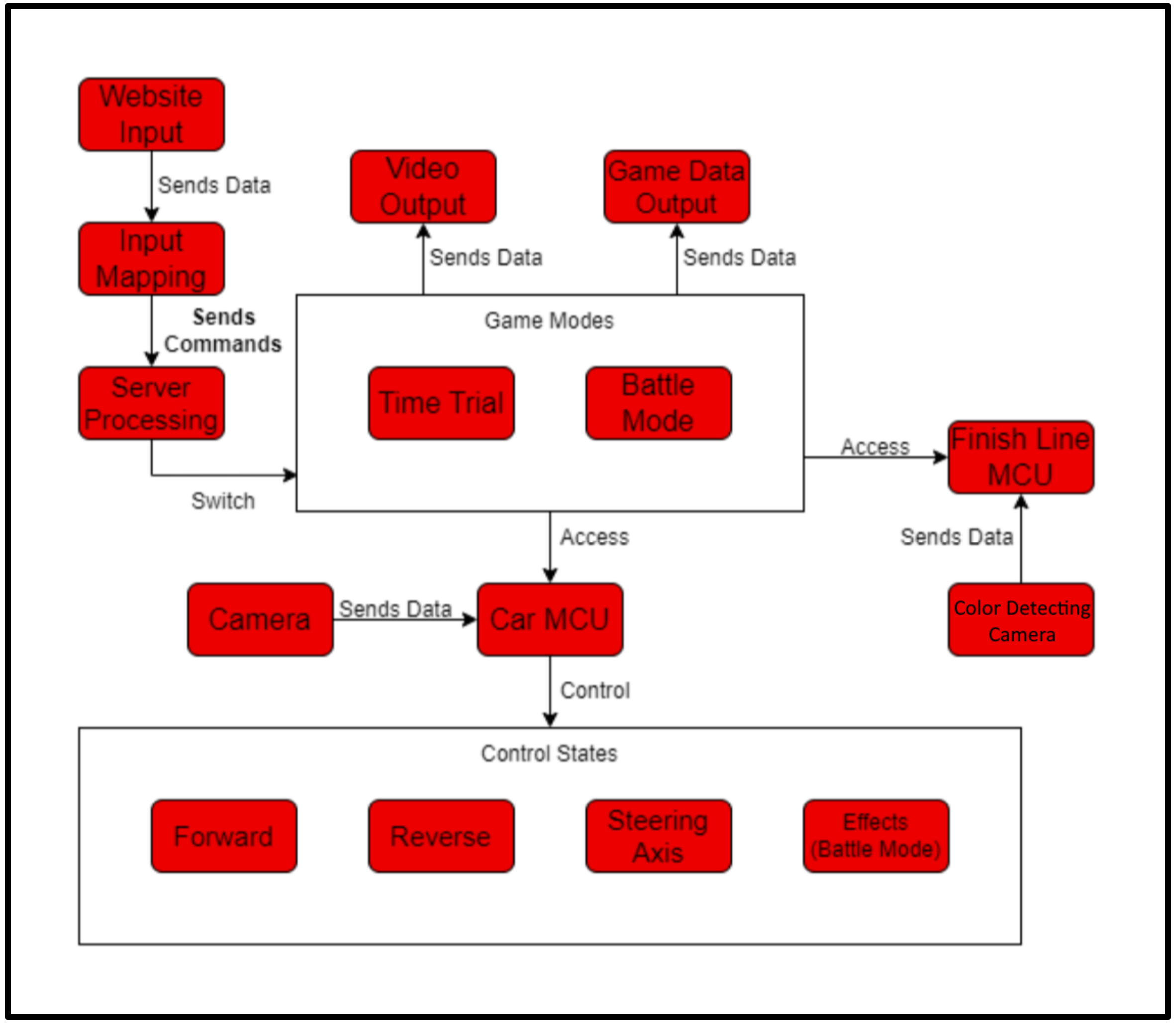
- Wi-Fi is the main communication protocol we will be using to connect the RC car to a local server and enabling internet-based control.
- WebSockets will be excellent for real-time control commands between the website and server and between the server and the ESP32.
- RTSP will be used for a low-latency video stream from the RC car to the website.
- With the combination of these modes of communication, we hope to achieve low latency that meets our specification.



Communication Network Diagram

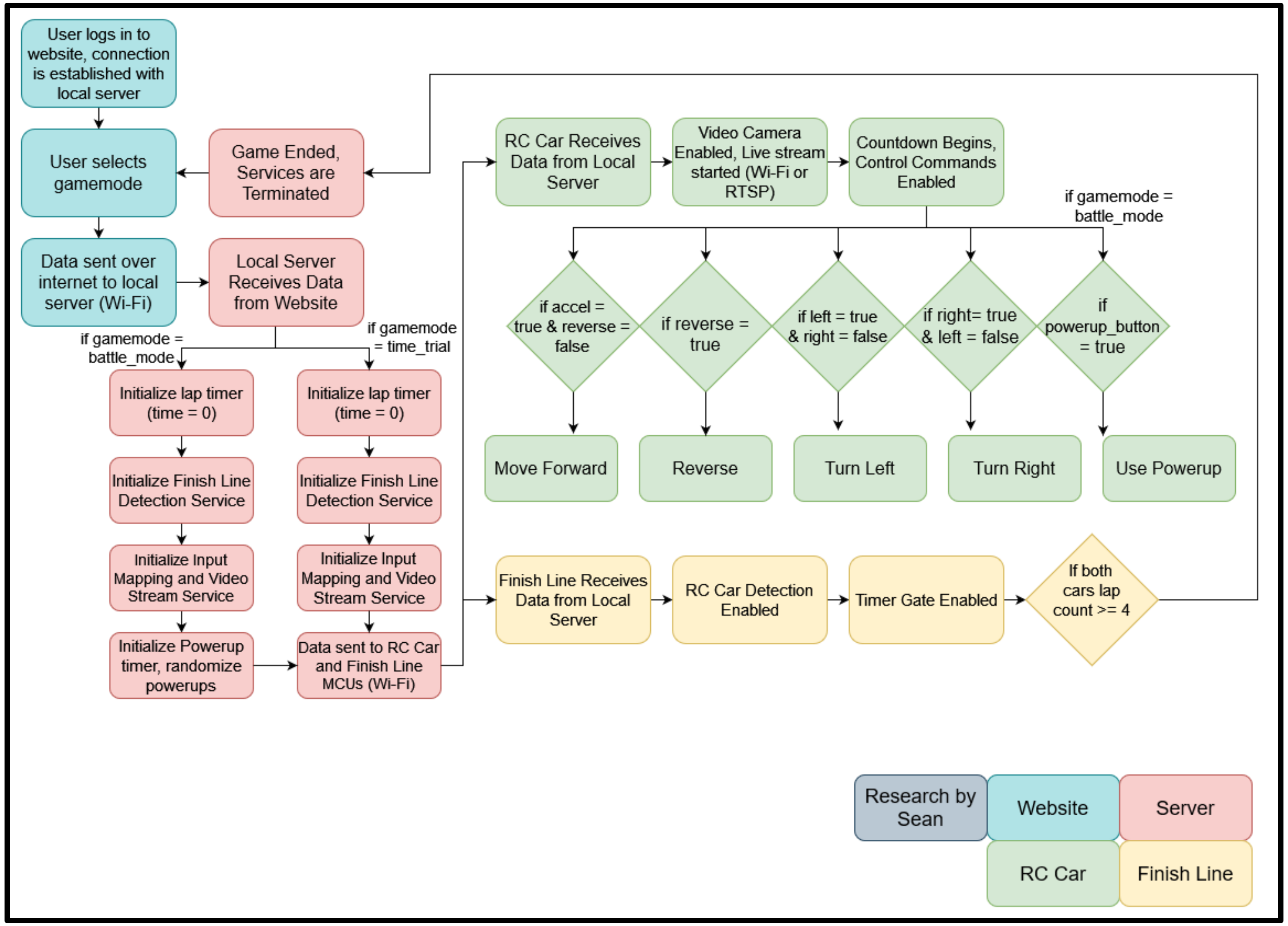
RACE System Software Design

Software Functionality



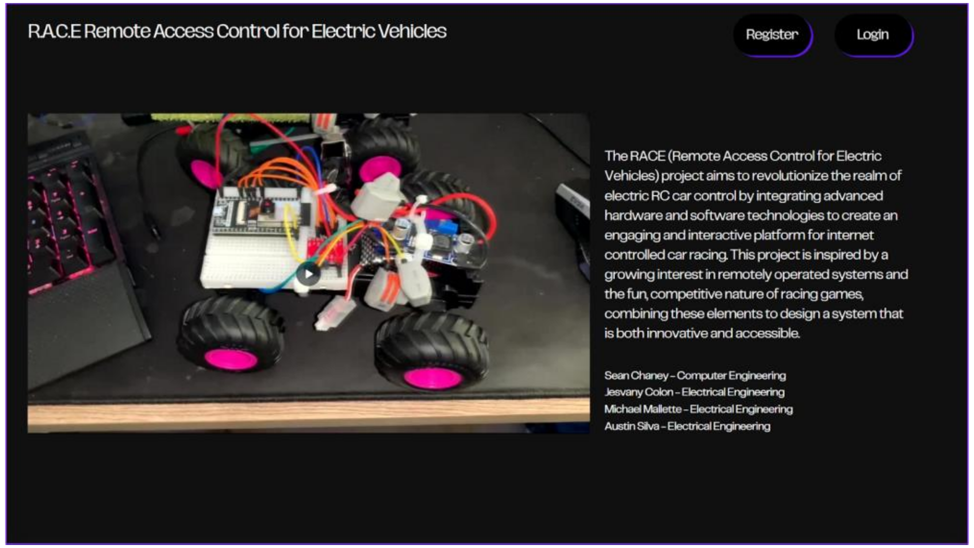
- Simplified version of our software design diagram.
- The color detecting camera will use an algorithm to detect the color of each car and track the lap times and winners.
- Time trial mode will simply be a race against the clock and/or against another player. Players will compete for the fastest times on the leaderboards.
- The battle mode is currently a stretch goal we will try to implement. We may be able to implement powerups at a specified time increment, which the players can then activate with another key.

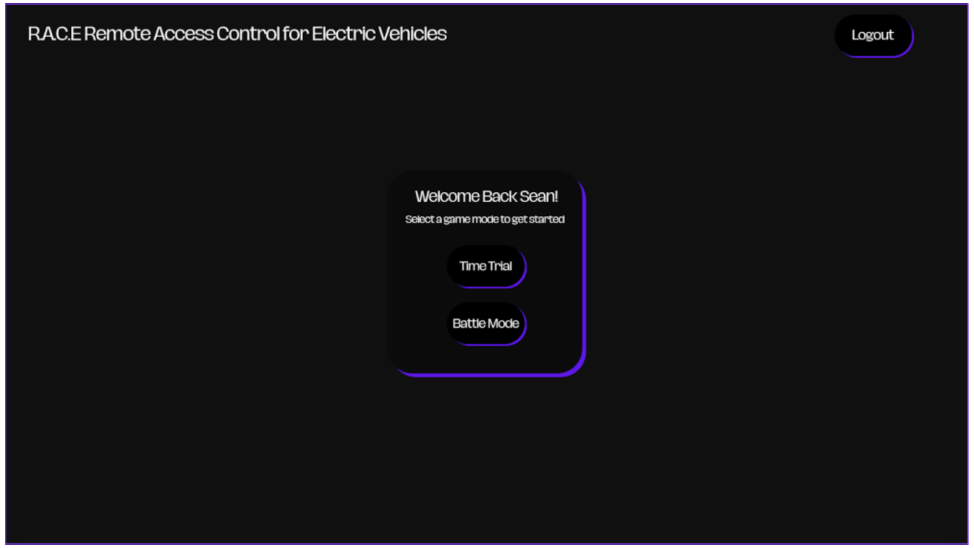
Software Design

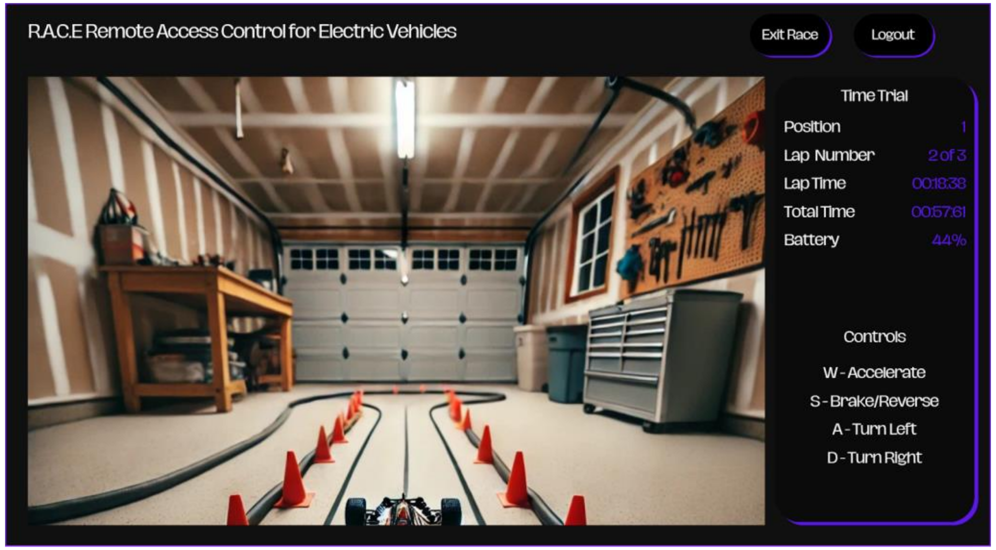


- We chose to use C as the language for the ESP32s due to its higher performance and direct control over hardware compared with other languages like Python.
- Our local server has been written in JavaScript as it has built in WebSocket support, can handle multiple WebSocket connections simultaneously since it runs code asynchronously, and can easily integrate with our website also utilizing JavaScript.

Website Design



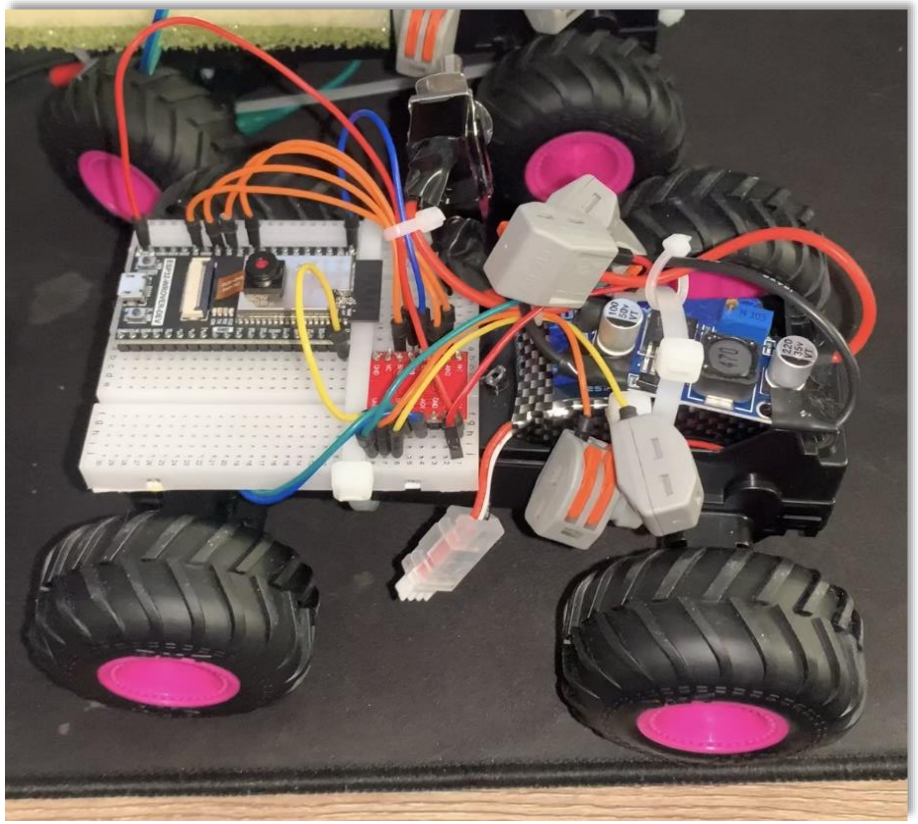




Prototyping & Testing

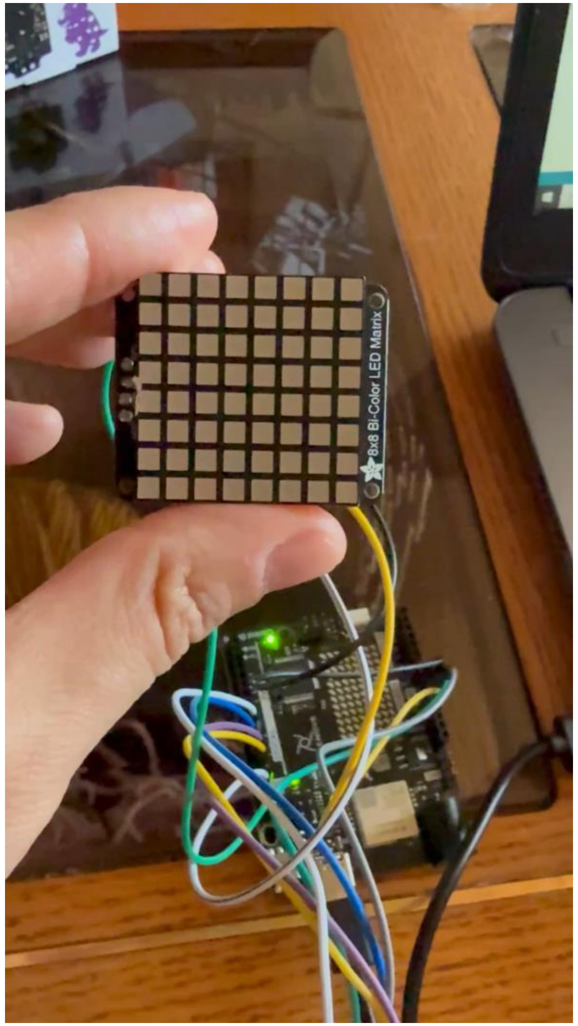
IC Car Prototype & Testing

- Currently can control the car with a WebSocket.
- Can stream video without user control.



Finish Line System Testing

```
19:26:30.310 -> Ready
19:26:32.337 -> Set
19:26:34.333 -> Go!
19:27:17.573 -> Detected 1 blocks:
19:27:17.573 ->   Block 0: sig: 1 x: 111 y: 115 width: 42 height: 25 index: 228 age: 5
19:27:17.573 -> Lap: 1
19:27:38.192 -> Detected 1 blocks:
19:27:38.192 ->   Block 0: sig: 1 x: 259 y: 125 width: 114 height: 29 index: 76 age: 20
19:27:38.192 -> Lap: 2
19:28:29.744 -> Detected 1 blocks:
19:28:29.744 ->   Block 0: sig: 2 x: 195 y: 2 width: 58 height: 5 index: 108 age: 72
19:28:29.744 -> Lap: 3
19:28:29.744 -> Red Wins!
```



- Our next step is to optimize the detection and to test out ways to secure connections and enclose the system.

Challenges & Obstacles Encountered

- Car-Server connection,
  - ESP32 disconnection and reconnection
- Communication between dev board and Pixy 2 camera,
  - SPI/UART/I2C communication was not configured correctly and quickly enough with ESP32-WROOM-32, resulting in changes in the approach of our finish line system.
- Video transmission,
  - Streaming video with WebSockets implemented
- Optimization of Pixy 2 Camera,
  - The camera often detects multiple color blocks within an object at once, causing the program to give premature results.

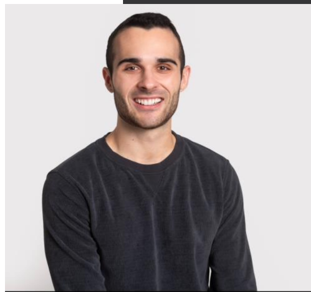
Administration & Mangement

Budget

Component	Part Description	Quantity	Cost
Cars System			
RC Toy-Grade Car	Car Chassis and DC Motors	2	$30
CMOS Image Sensor	OV2640 Image Sensor	2	$10
Battery Supply	7.4 V Li-Po Battery, 1100 mAh Capacity, with Charger	2	$20
Car's Enclosure	Enclosure for battery, PCB, and camera holder	2	~$20
Cars PCB			
Microcontroller	ESP32-WROVER-E	2	~$6
Motor Driver	DRV8833 Dual Motor Driver	2	~$4
Antenna	(Included in MCU Module)	2	$0
3.3 V Regulator	TPS561208	2	<$1
5 V Regulator	TPS563231	2	<$1
LDO Regulators	Torex XC6206 Series	4	<$2
UART-USB Bridge	CP2102N-A02-GQFN28	2	~$9
Camera Connector	Amphenol F32R-1A7H1-11024	2	<$2
On/Off Switch	500SSP1S2M2QEA Slider Switch	2	~$7
Fuse	Glass Cartridge Fuse	2	<$3
Circuit Boards	JLCPCB Fabricated	8	~$10
Other PCB Materials		2	~$20
Finish Line System & Racetrack			
Sensor/Camera	Pixy 2 Camera	1	$60
LED Matrix	Adafruit Bi-Color LED Square Pixel Matrix	1	$15
Freenove Control Board	V5 Rev4 WiFi, Arm Cortex-M4 Microcontroller with Onboard ESP32-S3	1	$20
Finish Line Enclosure	3D Printed Enclosure	1	~$20
Racetrack Materials	Materials to Build the Racetrack	1	~$50
Combined Total Cost			~$311

Work Distribution

Team Member	Key Responsibilities
Sean Chaney	Software Design & Implementation
	Website Design & Implementation
	Server Management
	Communications Implementation
Michael Mallette	Cars Design & Assembly
	Motors Implementation & Video Transmission
	Communications Implementation
	Software Design & Implementation
Jesvany Colon	Finish Line Design & Assembly
	Finish Line Camera Implementation
	Software Design & Implementation
	Support PCB Design & Implementation
Austin Silva	Project Management & Administration
	Lead PCB Design & Implementation
	Power Supply Unit Implementation
	Support Cars Design & Assembly
	Support Finish Line Design & Assembly



Progress and Plan

Present Day
- PCBs Received
- Car's Programming & Development
- Finish Line's Programming & Development

1-2 Weeks Out
- PCBs Assembly & Testing
- Complete Car's Programming
- Complete Finish Line's Programming

3-4 Weeks Out
- Integrate Systems
- Establish Full Communication Network
- Midterm Demo Video

5-7 Weeks Out
- Build Car's Structure
- Build Finish Line's Enclosure
- Expand Website Interface & Develop Game Modes

8-9 Weeks Out
- RACE System Complete Integration
- Final Testing
- Final Presentation & Live Demo

- ➢ PCBs have been received, currently undergoing assembly and testing.
- ➢ Video transmission and car's motor controls operational in prototype, currently optimizing both features for the best performance possible over Wi-Fi.
- ➢ Server and website have been established, currently working on the communication network between two cars and implementing the car's code onto the server.
- ➢ Finish line's code is almost complete, currently working on color detection and feedback optimization, will then be integrated into the sever when ready.
- ➢ Will begin finish line's enclosure, car's structure, and racetrack construction immediately after Midterm Demo Video.

Thank You!