Invertible Control Antweight Battle Bot

ILLINOIS

Ben Goldman and Jack Moran
Team 13

ECE 445

Problem & Motivation

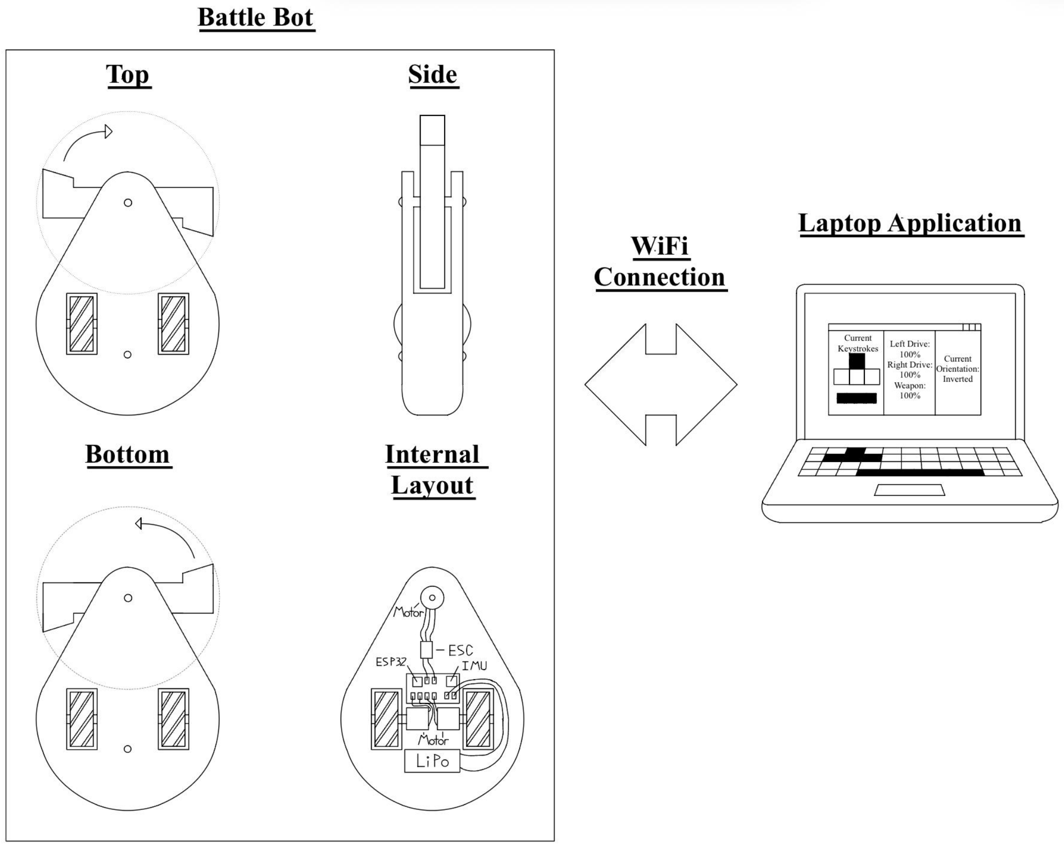
- Antweight robots operate under a strict 2 lb limit
- High-impact collisions frequently invert robots
- Many robots lose mobility, lose weapon effectiveness, or force the operator to mentally reverse controls
- Mechanical self-righting adds weight, complexity, and additional failure points

Project Evolution

- Considered multiple project ideas early
- Chose battle bot due to clear performance criteria and strong hardware/software integration
- Early key design challenge identified: intuitive control during inversion
- Led directly to IMU-based remapping concept

Solution and Overview

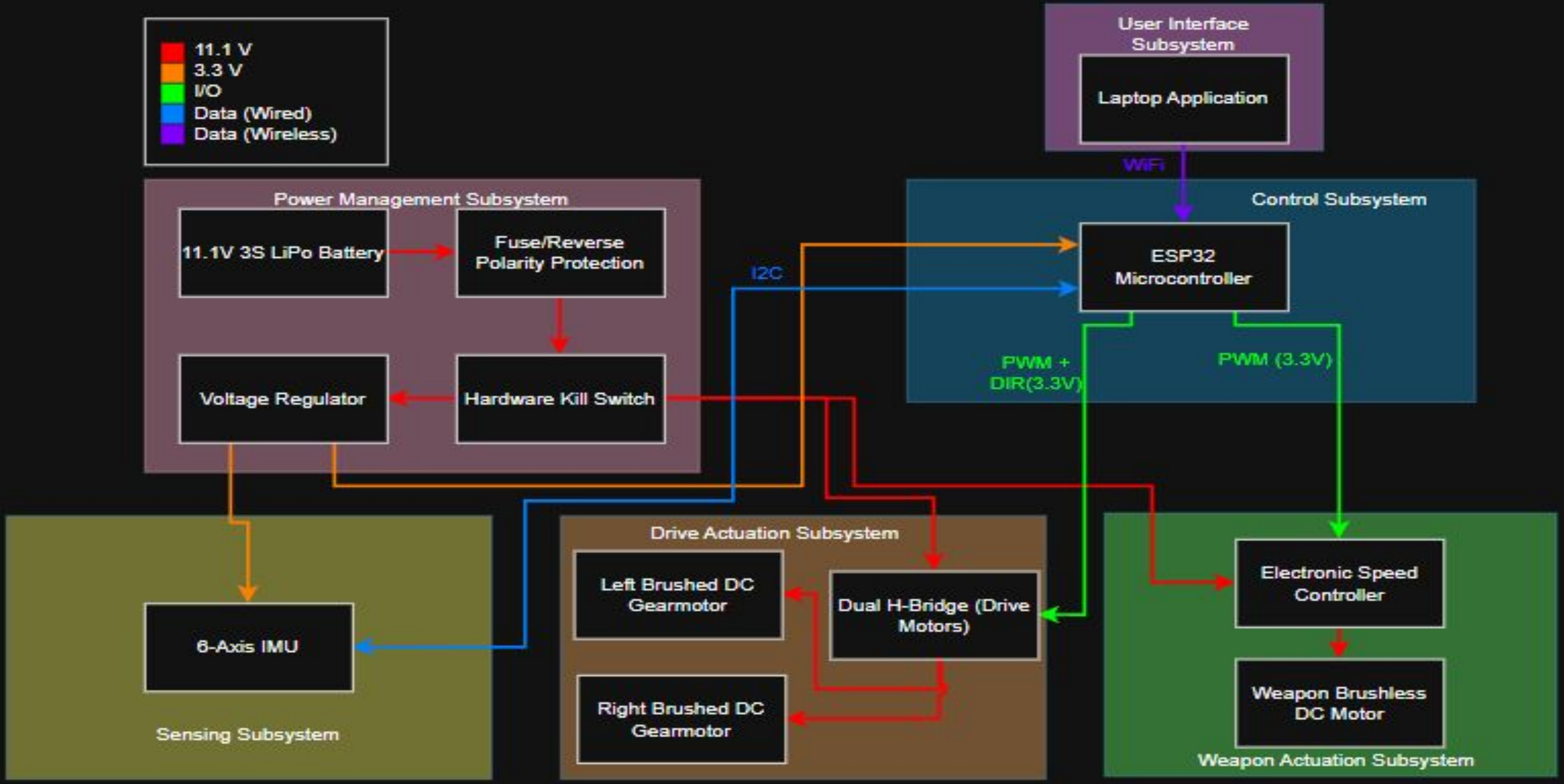
- Proposes an invertible combat robot that maintains full drive and weapon functionality in either orientation.
- Uses a 6-axis IMU and ESP32 to automatically detect inversion and remap drive controls in software.
- Eliminates mechanical self-righting systems by solving the inversion problem through orientation-aware control logic.



High-Level Requirements

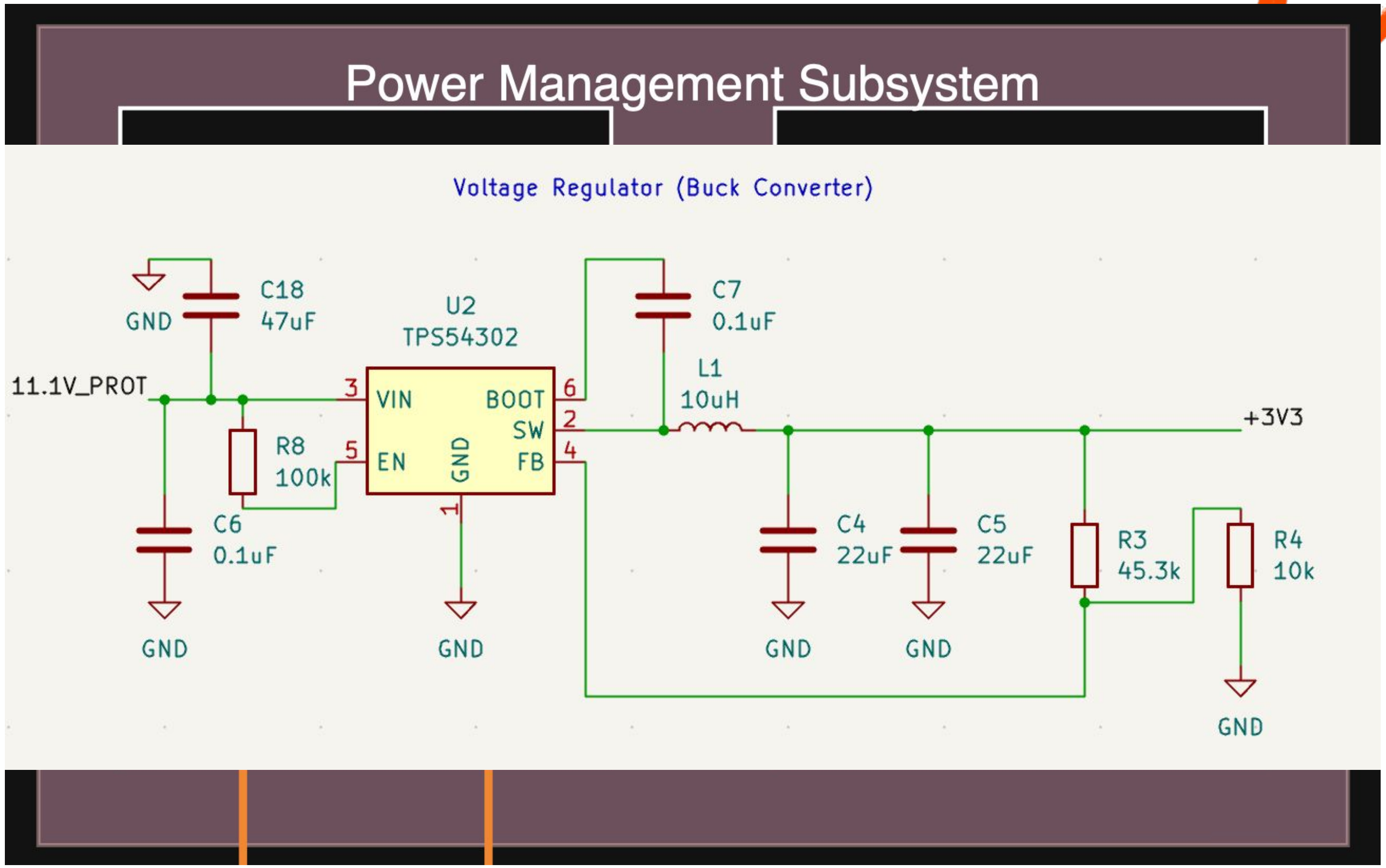
1. The robot must adhere to all of the antweight class specifications, specifically maintaining a total mass less than or equal to 2 lb and ensuring that the kinetic weapon system can be deactivated and brought to a complete rotational stop within 60 seconds of a command or failsafe trigger.
2. The onboard IMU and ESP32 control logic must detect a change in vertical orientation and invert the differential drive motor commands within 300 ms, ensuring that the operator perceives no lag in directional control relative to the new orientation.
3. The horizontal spinning bar must reach a combat-effective rotational velocity within 3 seconds of activation and demonstrate structural integrity to resume full speed rotation following a high-energy impact with an opponent or arena obstacle.

Block Diagram



Power Management Subsystem

- Distributes battery power to high-current motor loads and regulated logic circuits.
- Converts 11.1V battery input to a stable 3.3V rail for the ESP32 and IMU.
- Implements protection mechanisms including reverse polarity protection, fuse protection, and a hardware kill switch.



Power Integrity and Transient Design

Voltage Sag Estimation: $\Delta V \approx \Delta I \cdot \Delta t / C + \Delta I \cdot ESR$

Voltage Sag Estimates

Trial	C	ESR	ΔV_C	ΔV_{ESR}	ΔV total	Under 0.2V
1	220 uF	30 mΩ	0.5 * 300e-6 / 220e-6 = 0.682V	0.015V	0.697V	No
2	470 uF	20 mΩ	0.319V	0.010V	0.329V	No
3	1000 uF	20 mΩ	0.150V	0.010V	0.160V	Yes
4	1500 uF	12 mΩ	0.100V	0.0075V	0.108V	Yes

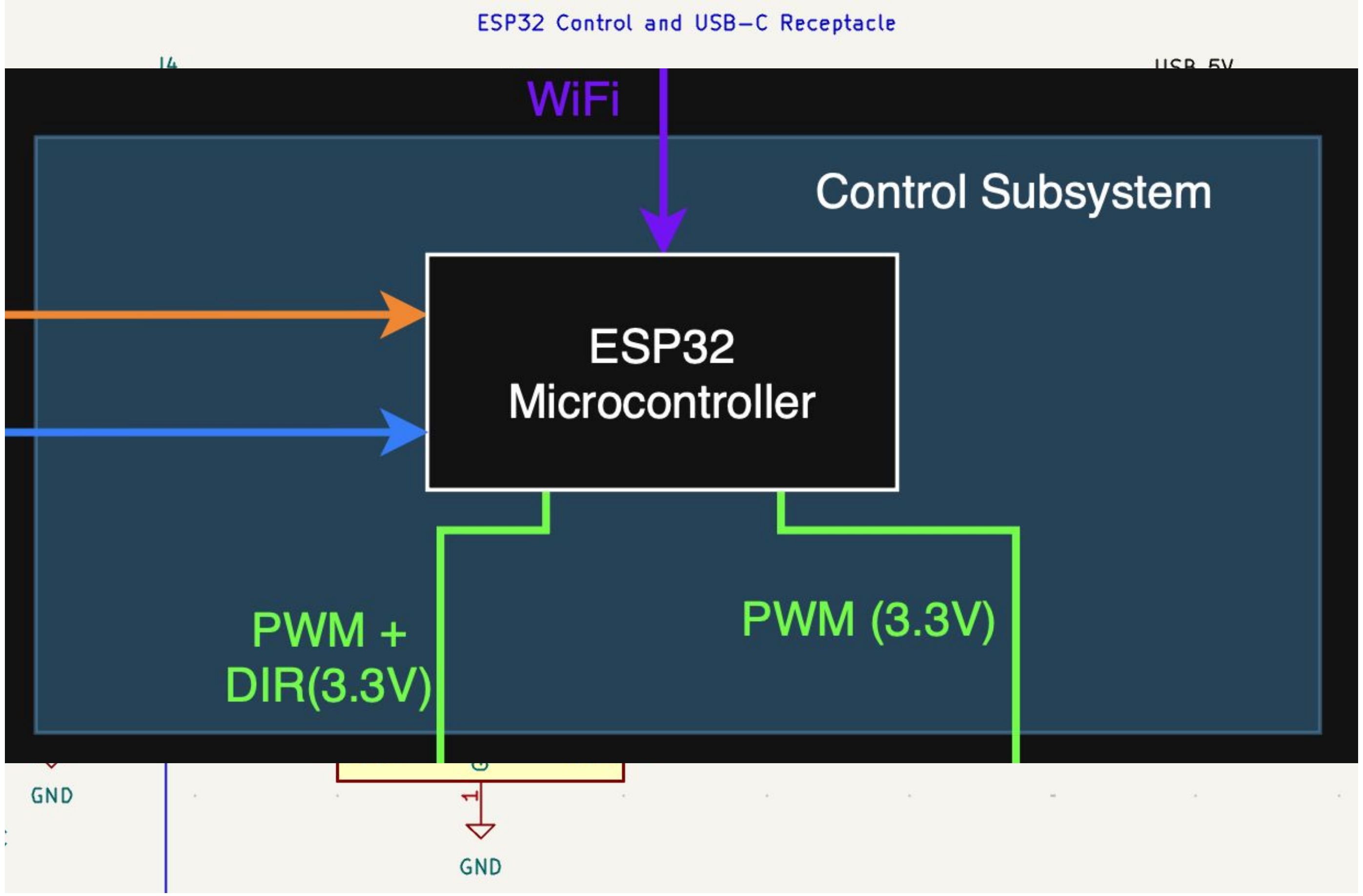
Power Verification Results

Power Management Subsystem Voltage Verification

Load Condition	Input Voltage (V)	Output Voltage (V)
No Load	11.1	3.27
ESP32 Only	11.1	3.25
ESP32 and Motors	11.0	3.25

Control Subsystem

- Serves as the processing unit of the robot using the ESP32 microcontroller.
- Processes operator commands and IMU data to generate motor and weapon control signals.
- Implements orientation-aware drive remapping and communication-loss failsafe behavior.



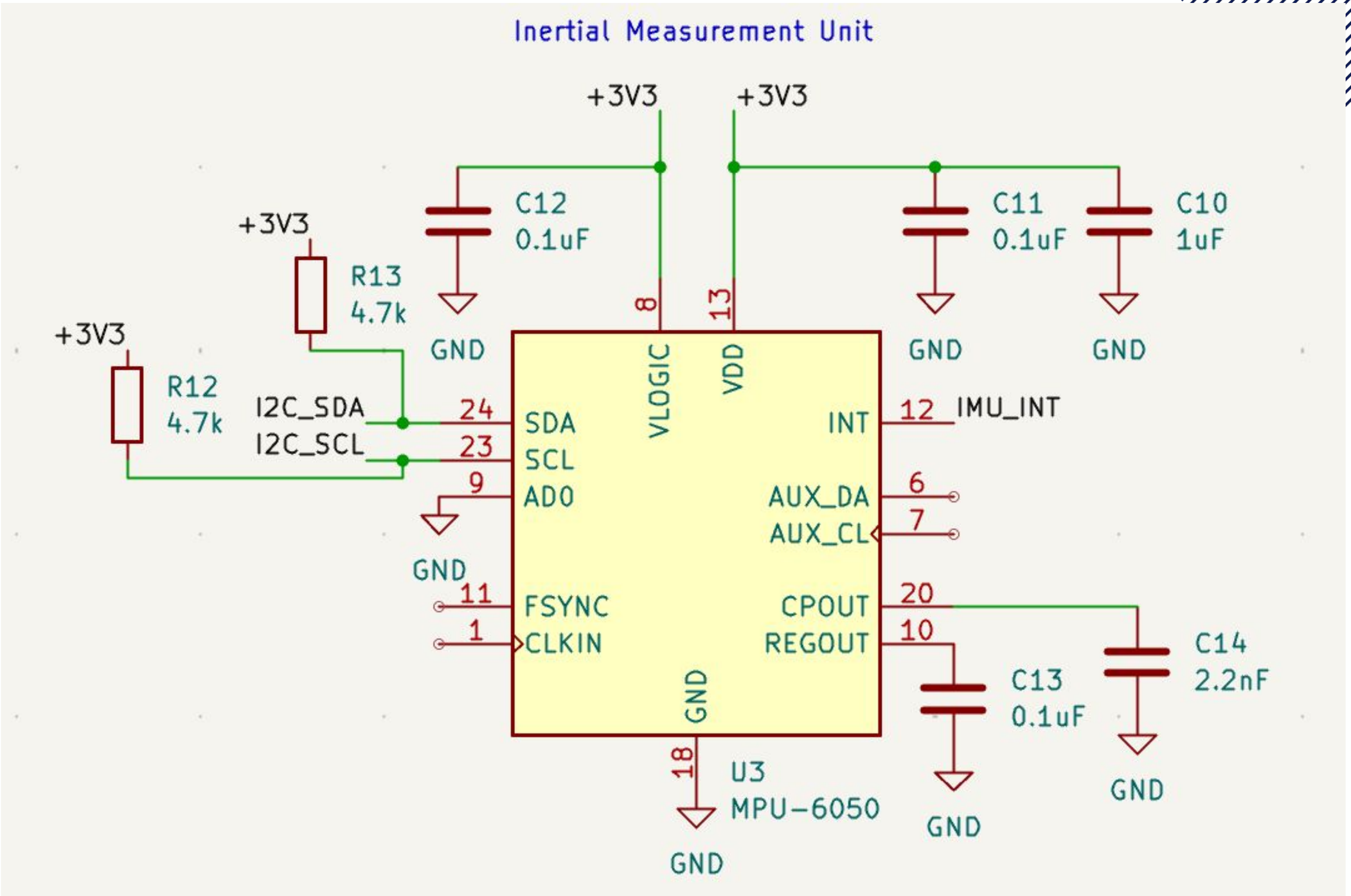
Control Verification Results

Inversion Control Response Time

Trial	Type	Response Time (ms)	Result
1	Upright to Inverted	68ms	Pass
2	Upright to Inverted	74ms	Pass
3	Inverted to Upright	58ms	Pass
4	Inverted to Upright	69ms	Pass

Sensing Subsystem

- Uses a 6-axis IMU to measure acceleration and angular velocity.
- Detects chassis orientation to determine upright or inverted state.
- Provides real-time sensor data to enable automatic control remapping.



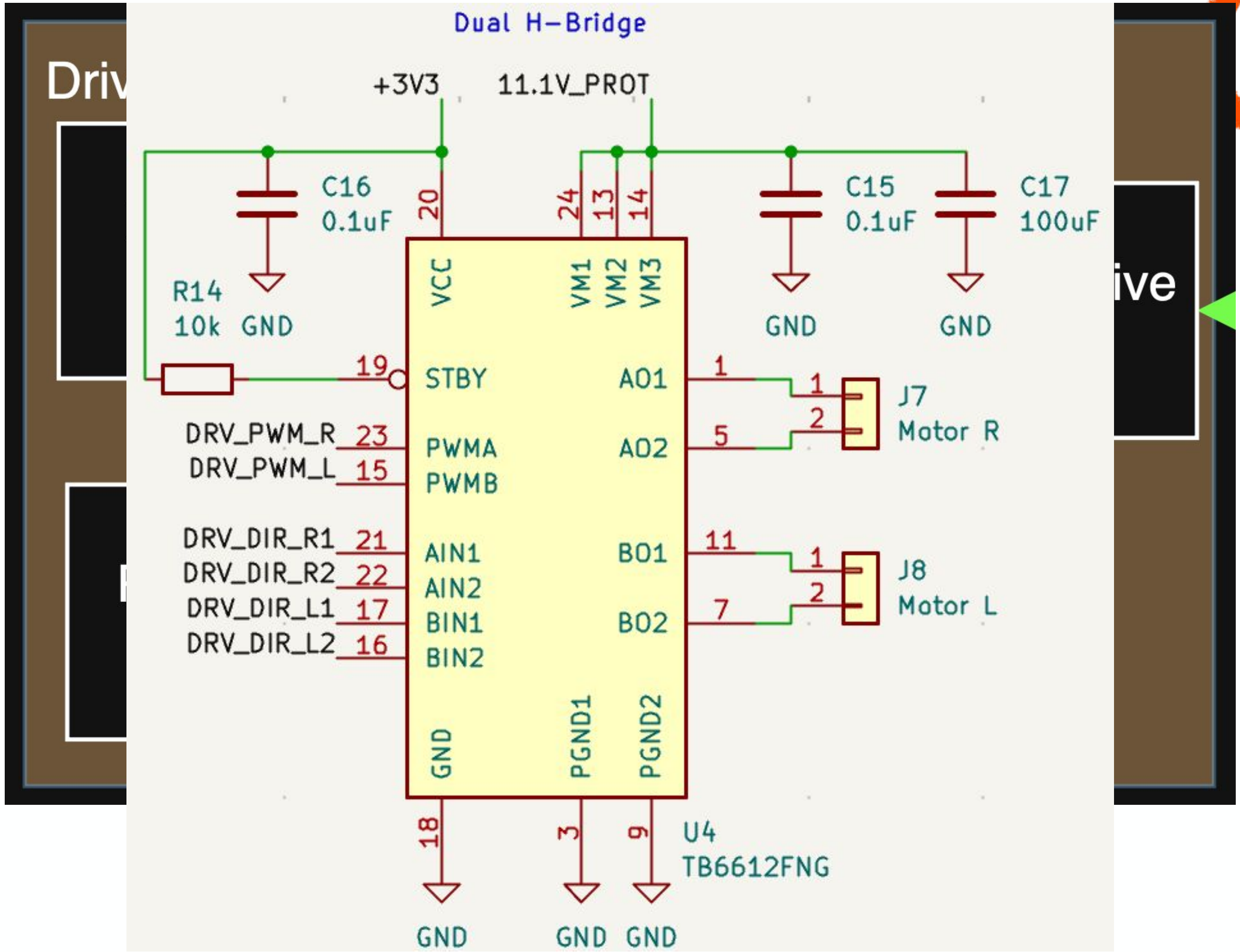
Sensing Subsystem Verification

Table 3: Sensing Subsystem Accelerometer Verification

Condition	a_x (g)	a_y (g)	a_z (g)	Magnitude (g)
Flat	0.01	-0.02	0.98	0.98
Tilted 45 Degrees	0.69	0.01	0.72	0.99
Constant rotation	Varied	Vaired	Varied	~1.00 (consistently)

Drive Actuation Subsystem

- Converts PWM and direction signals into bidirectional motion using a dual H-bridge and brushed DC motors.
- Enables differential tank-style steering in both upright and inverted orientations.
- Delivers mechanical torque while tolerating transient stalls and direction changes.



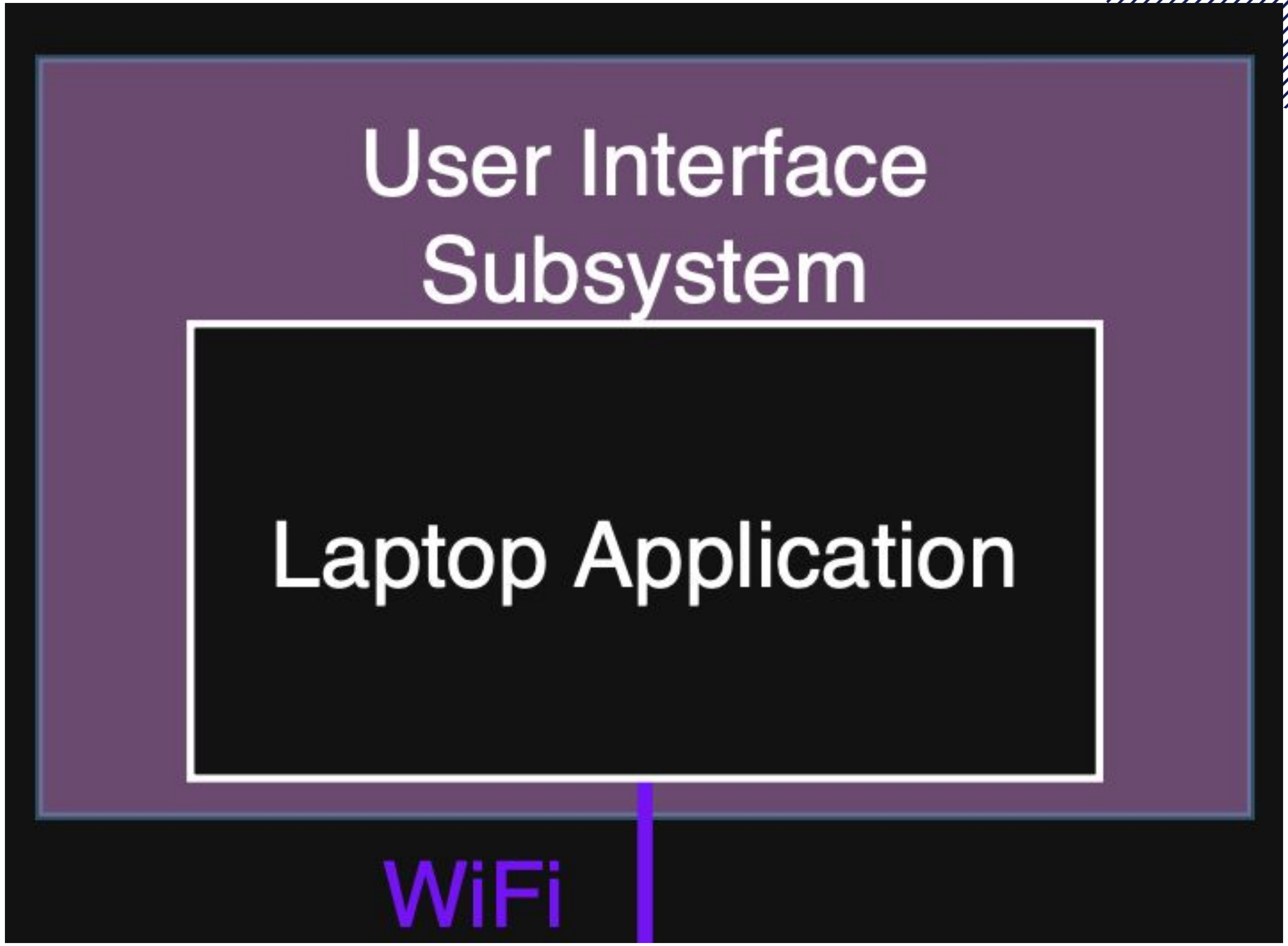
Drive Actuation Subsystem Verification

Voltage Supplied to Drive Motors with Different Duty Cycles

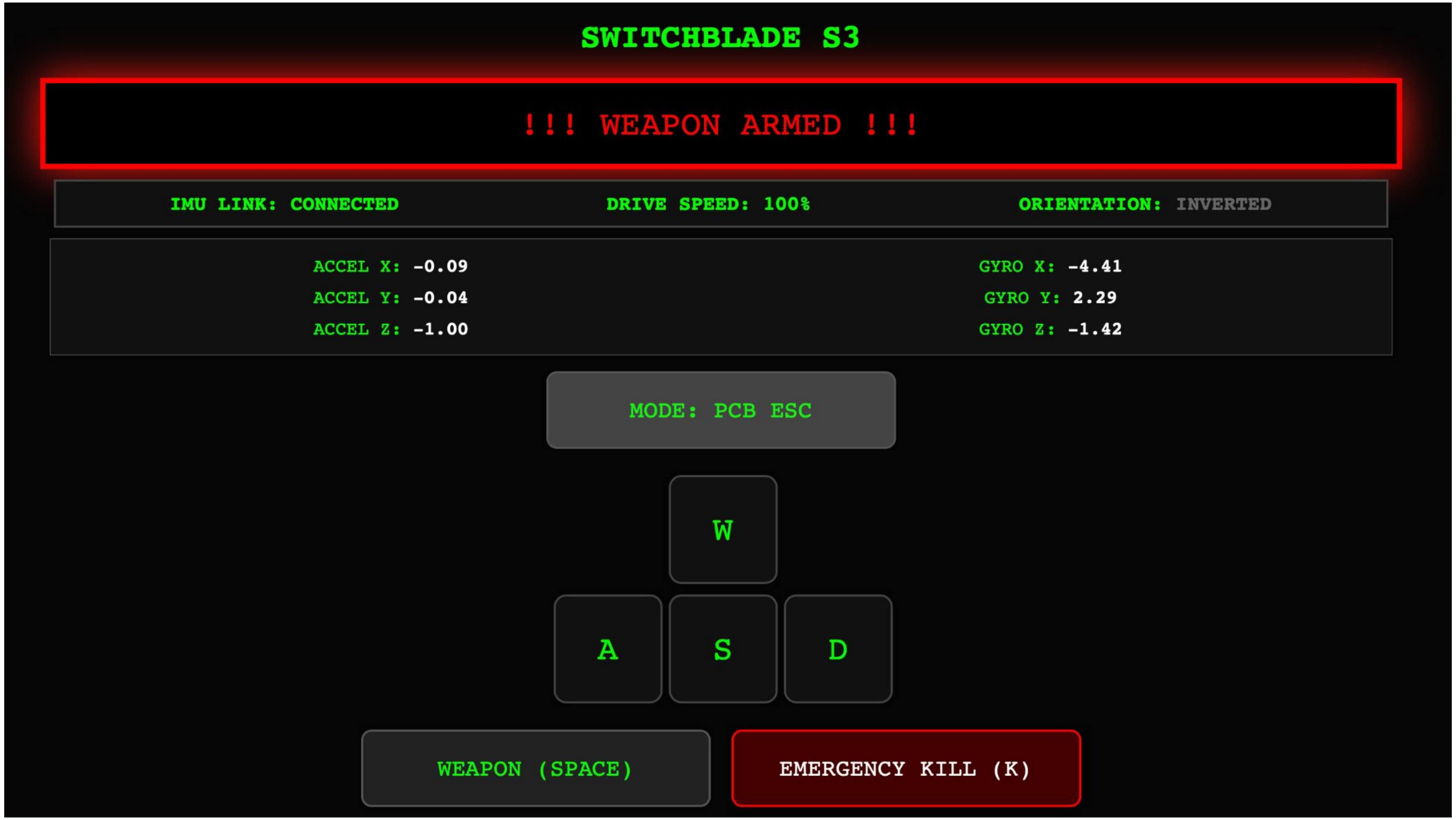
Duty Cycle	Voltage Across Motors (V)	Result
64 (25% duty cycle)	2.72	Pass
127 (50% duty cycle)	5.49	Pass
255 (100% duty cycle)	10.98	Pass

User Interface Subsystem

- Enables wireless communication between the operator’s laptop and the robot.
- Transmits drive and weapon commands at a defined refresh rate.
- Provides arming control and connection status feedback for safe operation.

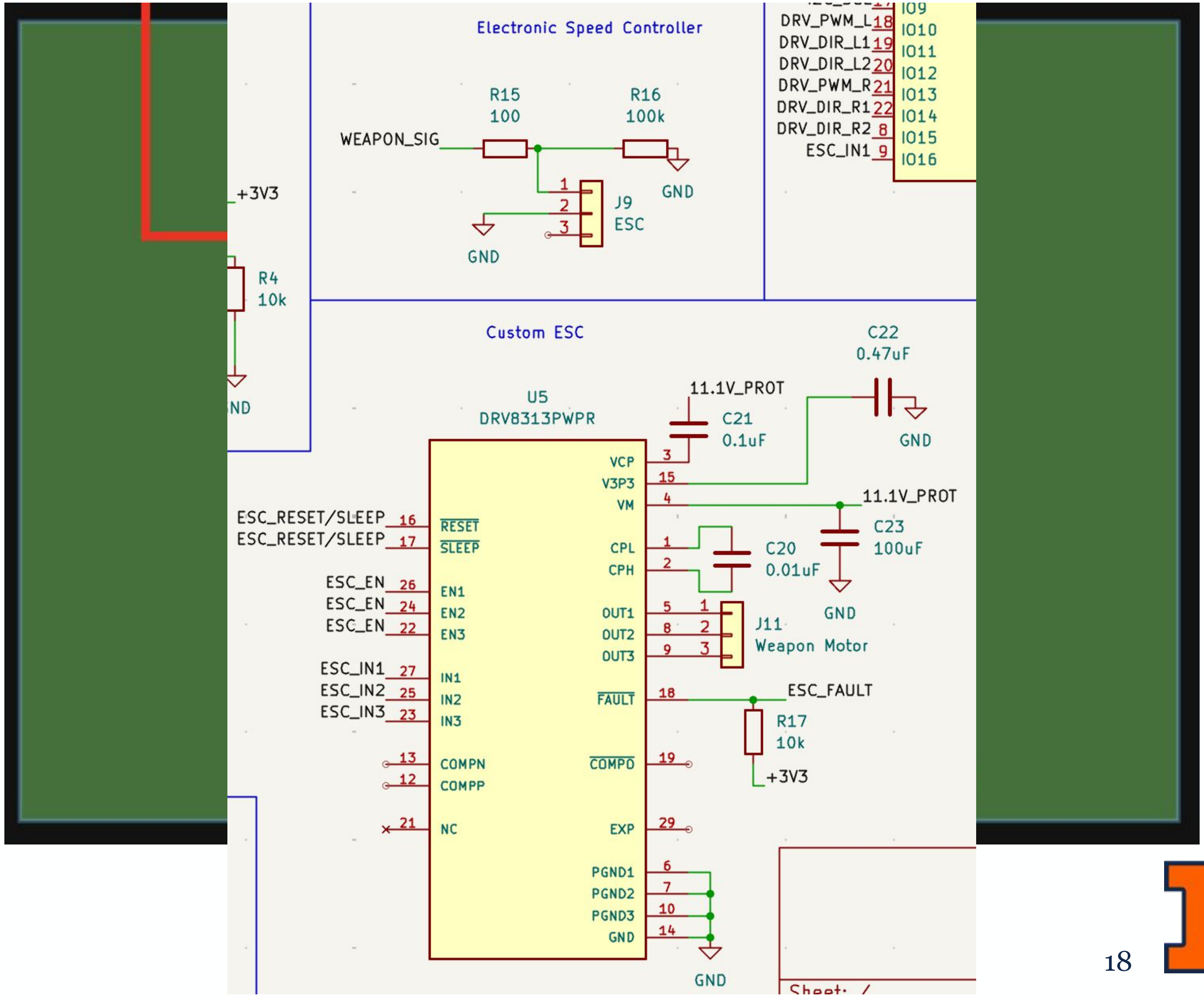


User Interface Subsystem



Weapon Actuation Subsystem

- Controls a brushless DC motor through an electronic speed controller or three-phase DC motor driver for the spinning weapon.
- Converts PWM throttle commands into high-speed rotational motion.
- Provides the robot's primary offensive capability.



Failure Analysis

Problem:

- Original weapon motor behaved improperly

Root Cause:

- Internal resistance too low
- Excessive current draw
- System could not drive effectively

Fix:

- Replaced motor with better electrical match
- DRV8313PWPR drove the new motor correctly

Safety and Reliability

- Ensures safe operation through hardware protections and fail-safe logic.
- Physically isolates motor power via a kill switch.
- Maintains system stability during high-current and high-impact conditions.
- Tested in a safe area with preventative measures taken against physical injury.

Lessons Learned & Future Work

Lessons:

- Subsystem decomposition matters
- Power integrity matters more than expected
- PCB integration exposes issues breadboards can hide
- Component selection must match system-level constraints

Future Work:

- Stabilize IMU communication
- Add filtering/calibration
- Complete inversion-triggered remapping validation
- Perform full-system stress and failure-mode testing

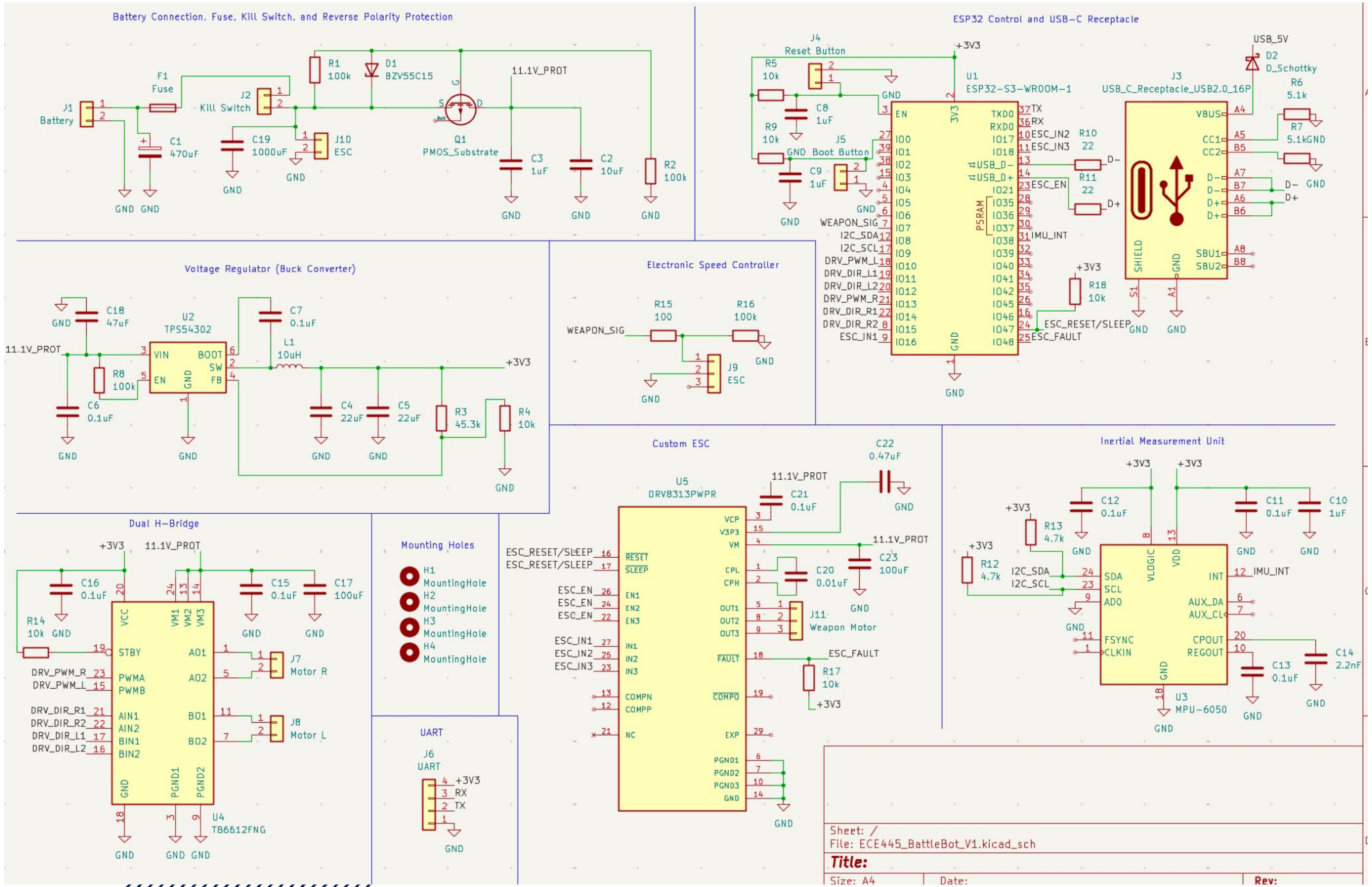
Conclusion

- Identified inversion as a key combat robotics problem
- Designed a software-centered solution using IMU + ESP32
- Built custom PCB and integrated core subsystems
- Verified stable power and responsive control
- Achieved full motor functionality on PCB

THANK YOU!

Questions?

Appendix


Battery Connection, Fuse, Kill Switch, and Reverse Polarity Protection
J1 Battery
F1 Fuse
J2 Kill Switch
C1 470uF
C19 1000uF
J10 ESC
R1 100k
D1 BZV55C15
Q1 PMOS_Substrate
11.1V_PROT
C3 1uF
C2 10uF
R2 100k
ESP32 Control and USB-C Receptacle
J4 Reset Button
J5 Boot Button
R5 10k
R9 10k
C8 1uF
C9 1uF
U1 ESP32-S3-WROOM-1
+3V3
R10 22
R11 22
J3 USB_C_Receptacle_USB2.0_16P
USB_5V
D2 D_Schottky
R6 5.1k
R7 5.1k
R18 10k
WEAPON_SIG
I2C_SDA
I2C_SCL
IMU_INT
ESC_RESET/SLEEP
ESC_FAULT
Voltage Regulator (Buck Converter)
U2 TPS54302
C18 47uF
C7 0.1uF
L1 10uH
R8 100k
C6 0.1uF
C4 22uF
C5 22uF
R3 45.3k
R4 10k
Electronic Speed Controller
R15 100
R16 100k
J9 ESC
Dual H-Bridge
U4 TB6612FNG
C16 0.1uF
C15 0.1uF
C17 100uF
R14 10k
J7 Motor R
J8 Motor L
Mounting Holes
H1 MountingHole
H2 MountingHole
H3 MountingHole
H4 MountingHole
UART
J6 UART
Custom ESC
U5 DRV8313PWPR
C22 0.47uF
C21 0.1uF
C20 0.01uF
C23 100uF
J11 Weapon Motor
R17 10k
Inertial Measurement Unit
U3 MPU-6050
C12 0.1uF
C11 0.1uF
C10 1uF
R13 4.7k
R12 4.7k
C13 0.1uF
C14 2.2nF
Sheet: /
File: ECE445_BattleBot_V1.kicad_sch
Title:
Size: A4
Date:
Rev: