Wawasee Supermileage

2018-2019 Motor Controller

"Turbo IQ"

Documentation for Technical Inspection, and Phase 3 Registration

Table of Contents:

Basic Information & Features

Features:

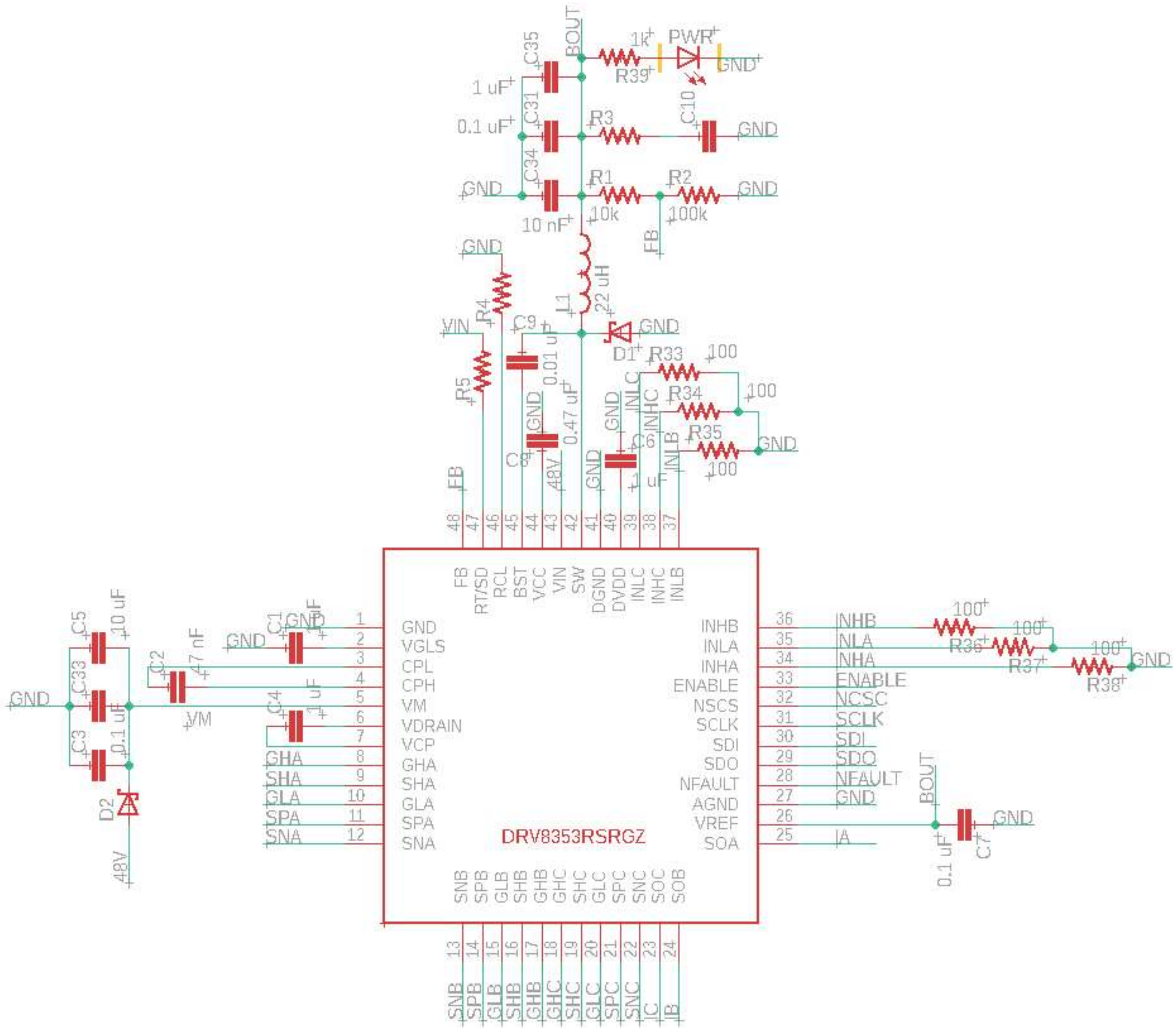
- Uses TI DRV8353RS in VQFN48 Package
- Built in Buck Converter for Motor Control Logic
- Uses Teensy v3.2 for Motor Control, and other GPIO functions
- SPI Communication with Teensy, automatic shutdown on fault condition
- Built in current shunt monitors, and voltage monitoring for battery and motor phases
- Uses highly efficient MOSFETs with a R_{DS} of 2.1 mΩ, resulting in nearly 99% efficiency
- Allows for trapezoidal, field oriented, and sinusoidal current control of motor
- Theoretical 100V limit, and 10V limit with different component choice
- Theoretical 75A limit with the use of external capacitors
- Has 100% experimental probability of failure
- Looks pretty, has massive amounts of vias.
- I2C Driver Display Communications
- Spins our motor in theory
- Works for the most part
- Jokes on silkscreen

Controller Characteristics	
Voltage Range	12V-60V
Efficiency	99.5% @ 48V 30A
Voltage Supply	5V @ 350mA
Bootstrapping Voltage	58.4V
Communication	SPI + I2C
Works?	Probably

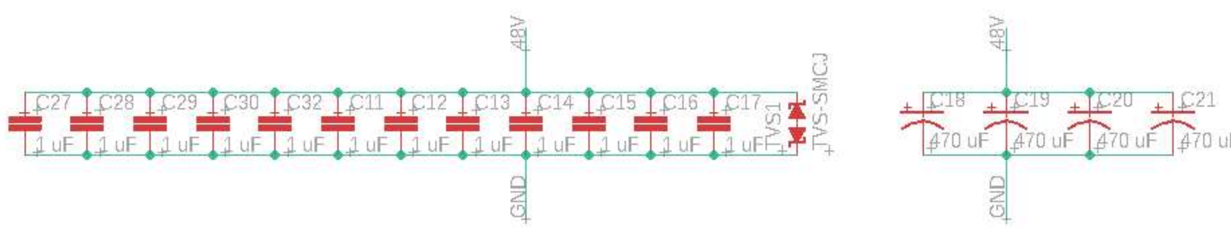
Schematic/Board Layout

(Note: All component values may not represent final value choices.)

Gate Driver, Including Buck Converter

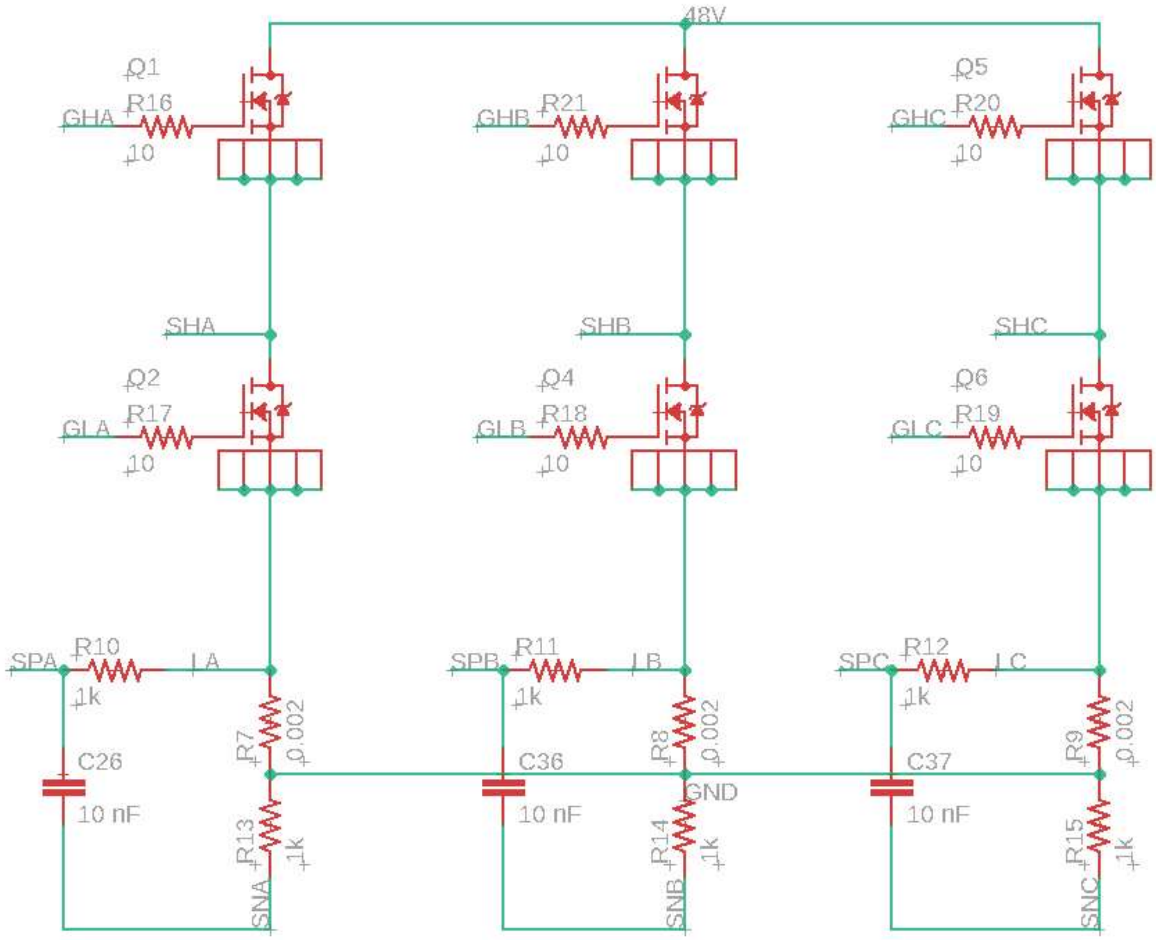


Decoupling, Bypass and Bulk Capacitors, and TVS Diode

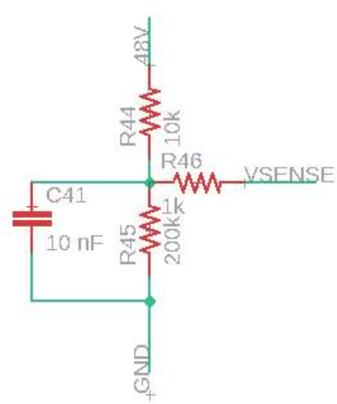


Schematics Continued

Triple Half H Bridge Mosfet Configuration with Current Monitoring

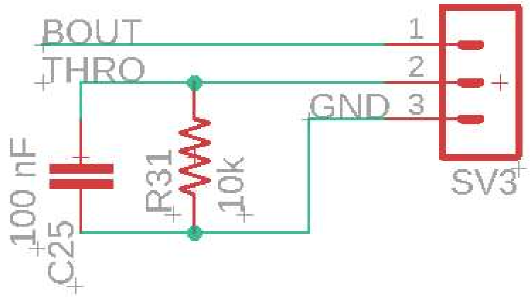


Voltage Divider for Battery Voltage Sensing

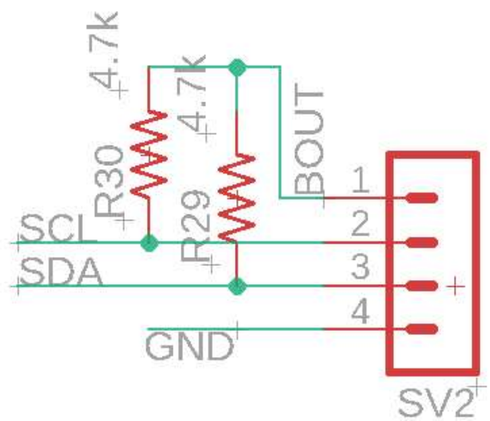


Schematics Continued

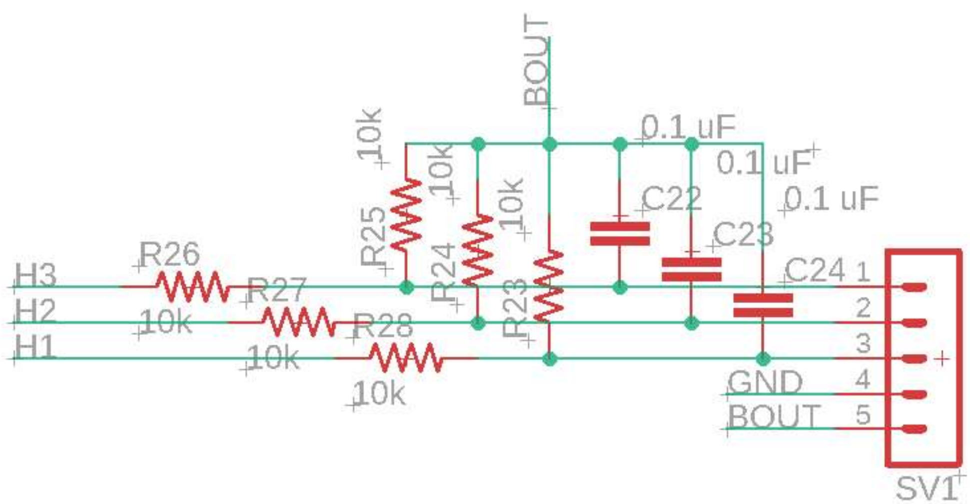
Throttle Input



I2C Header

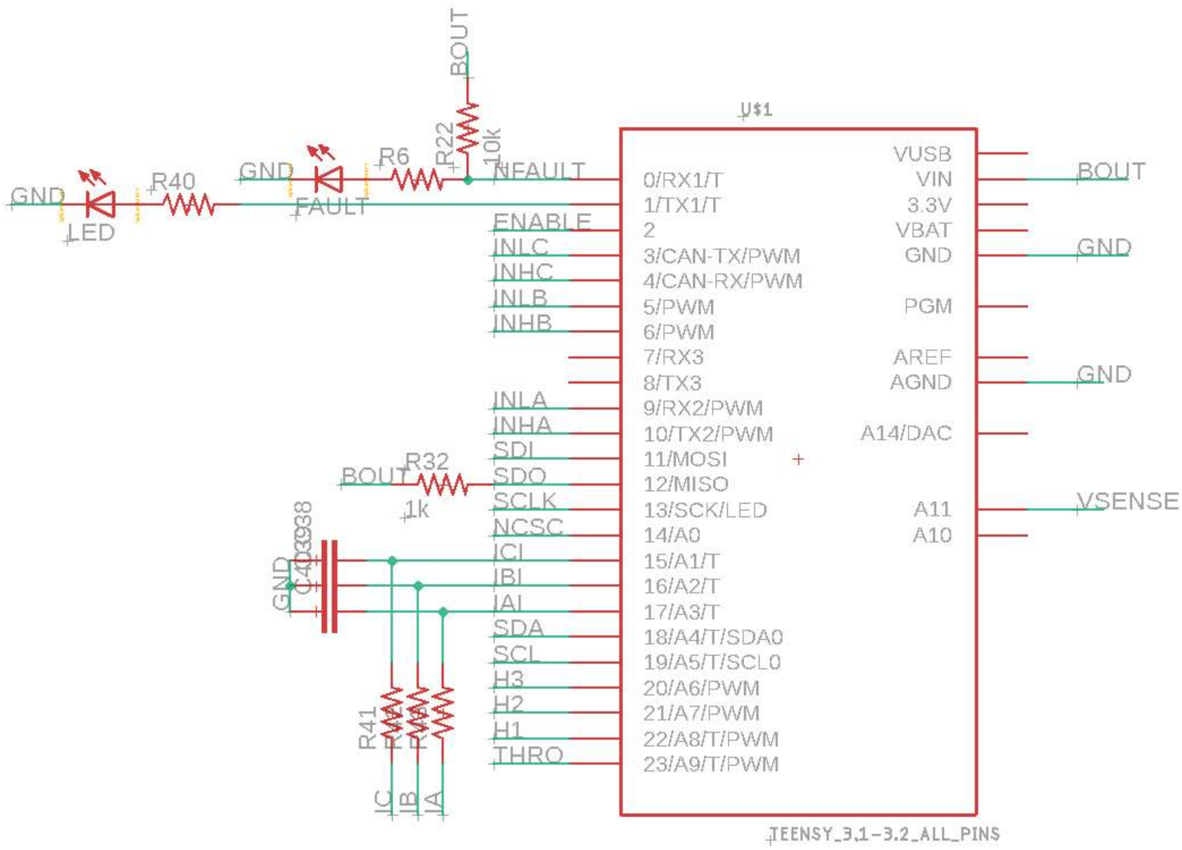


Hall Inputs



Schematics Continued

Teensy v3.2



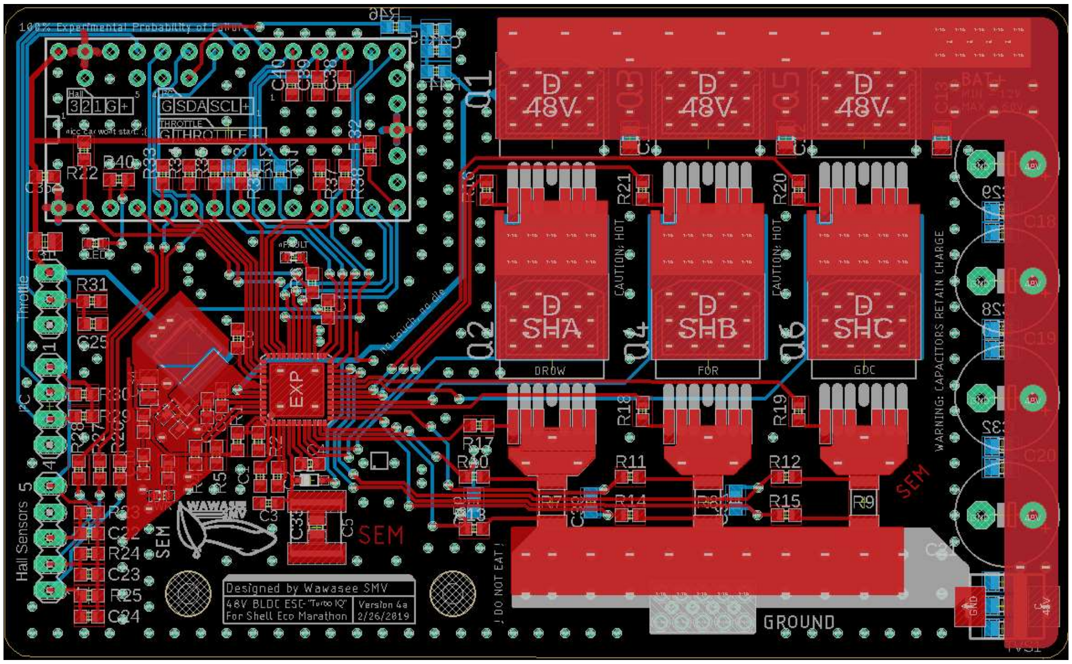
Teesny I/O Configuration

(Note: Pins 7, 8, A14, A10 are unused)

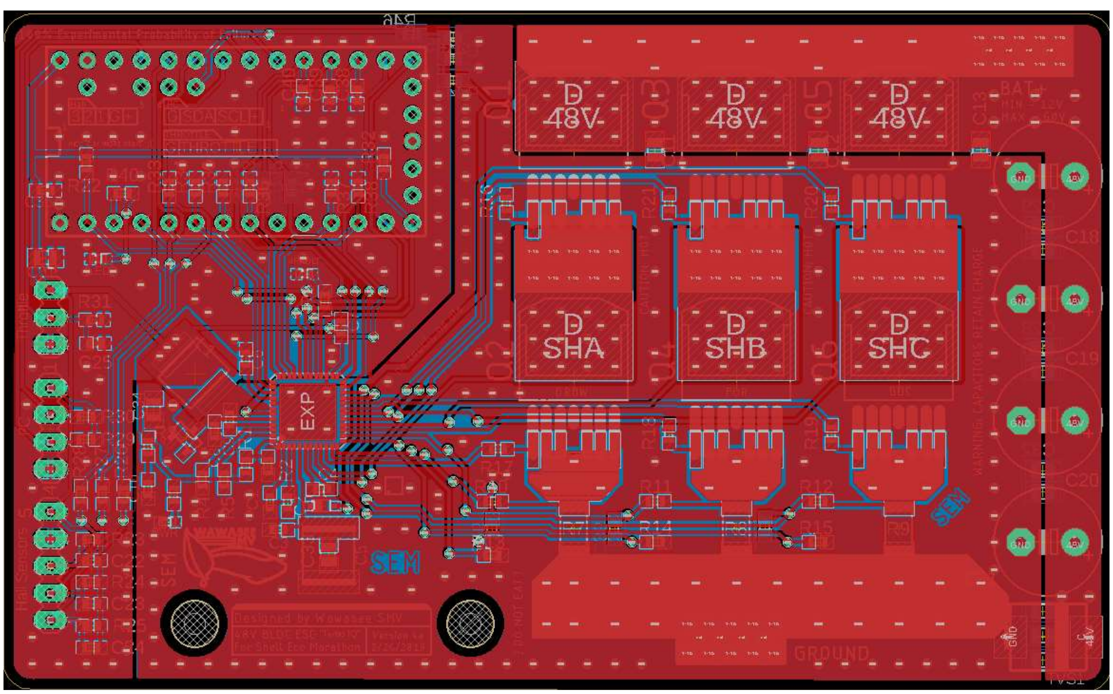
Pin	**I/O**	**Function**
0	I	nFault - Diver Fault Detection
1	O	External LED
2	O	Enable Driver
3	O	Phase C - Low Input
4	O	Phase C - High Input
5	O	Phase B - Low Input
6	O	Phase B - High Input
9	O	Phase A - Low Input
10	O	Phase A - High Input
11	C	Driver SDI
12	C	Driver SDO
13	C	Driver SCLK
14	C	Driver NCSC
A1/15	I	Phase C Current Sense
A2/16	I	Phase B Current Sense
A3/17	I	Phase A Current Sense
18	C	I2C SDA
19	C	I2C SCL
20	I	Hall Sensor 3
21	I	Hall Sensor 2
22	I	Hall Sensor 1
23	I	Throttle Input
A11	I	Voltage Sense

Board Layout

Both Layers without Ground Planes

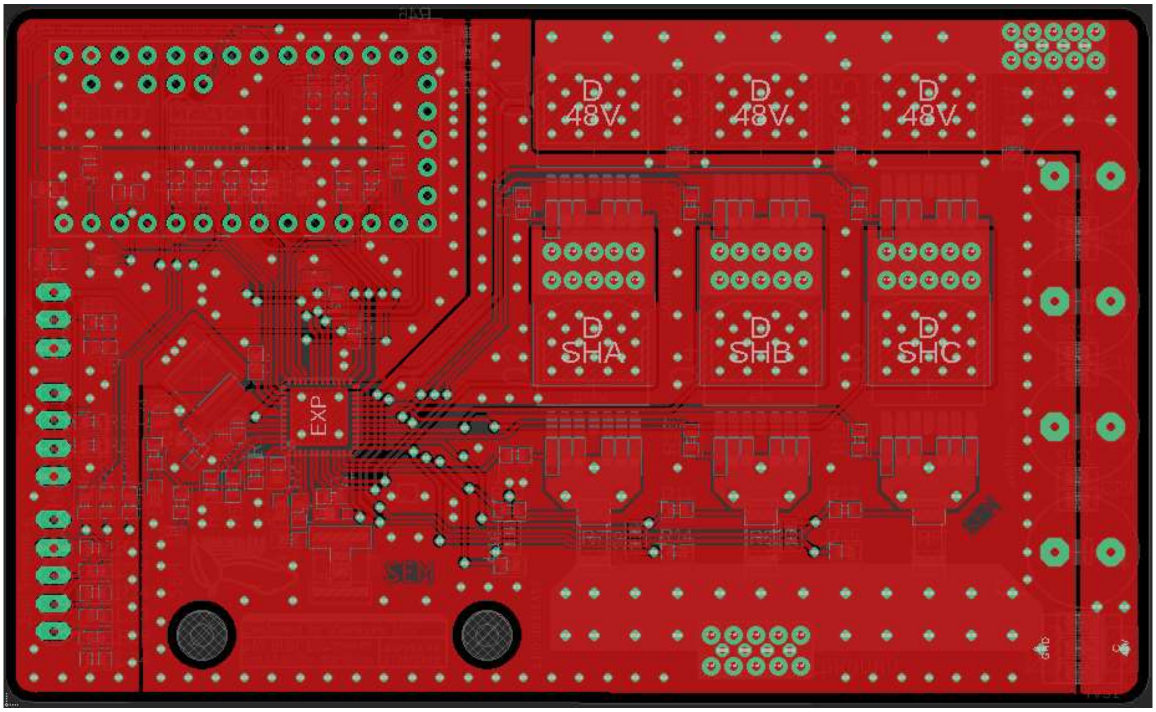


Both Layers with Ground Planes

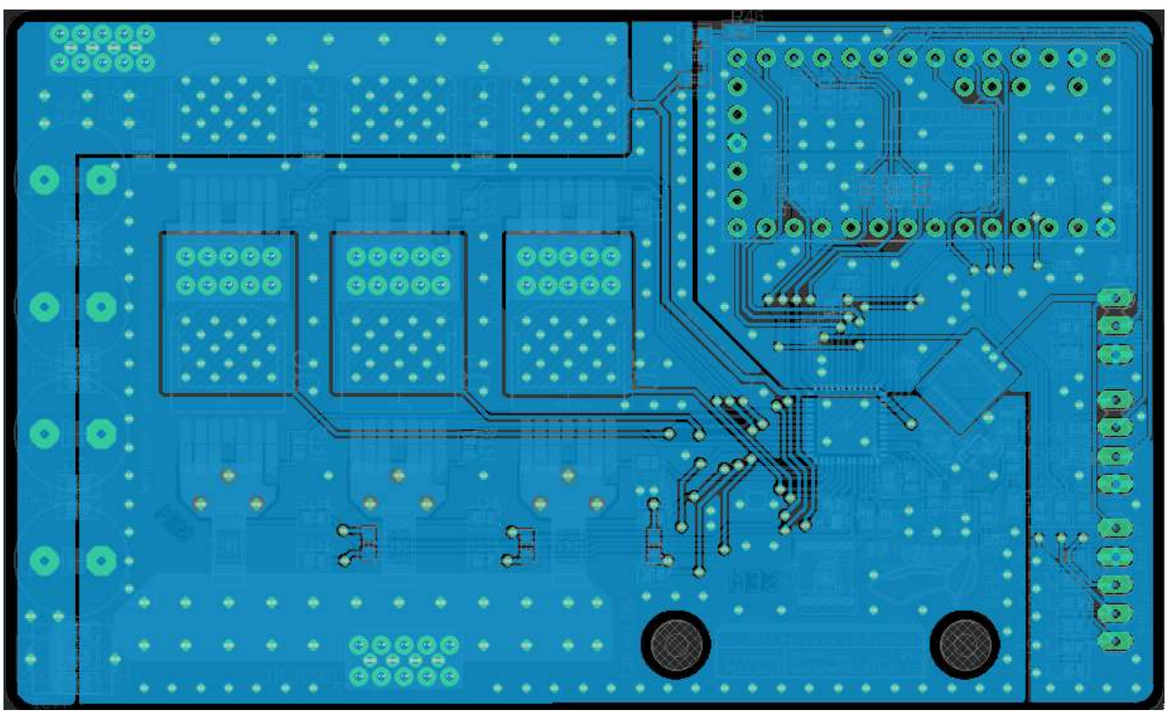


Board Layout Continued

Top Layer

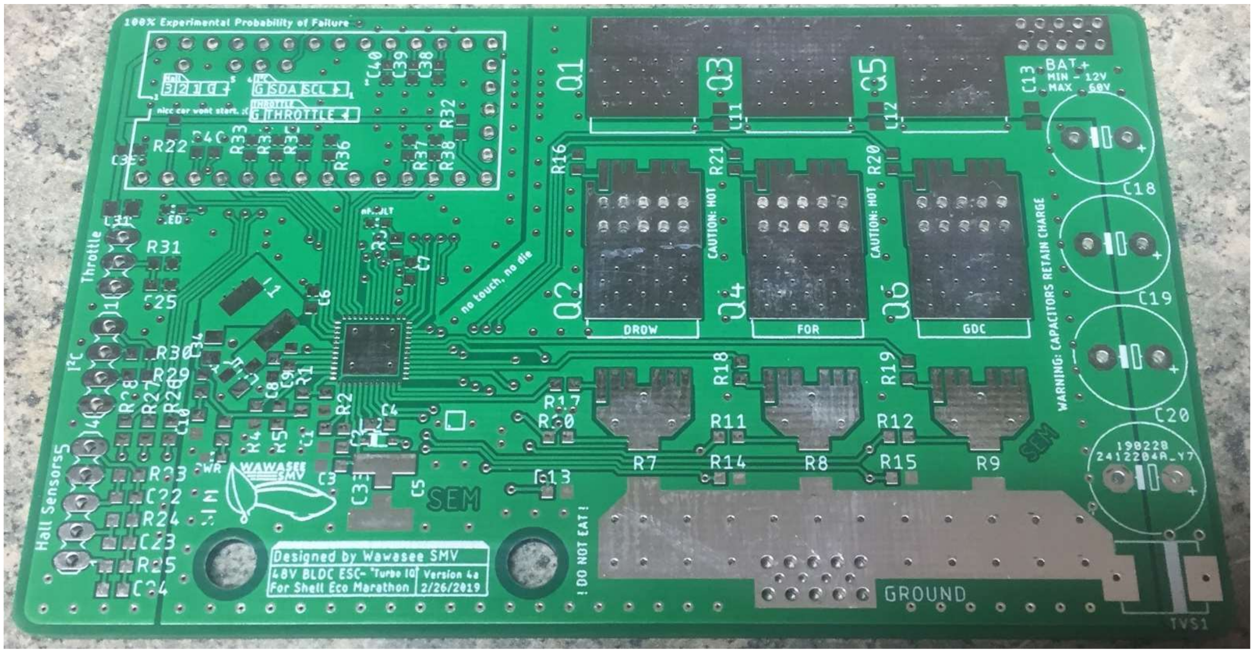


Bottom Layer

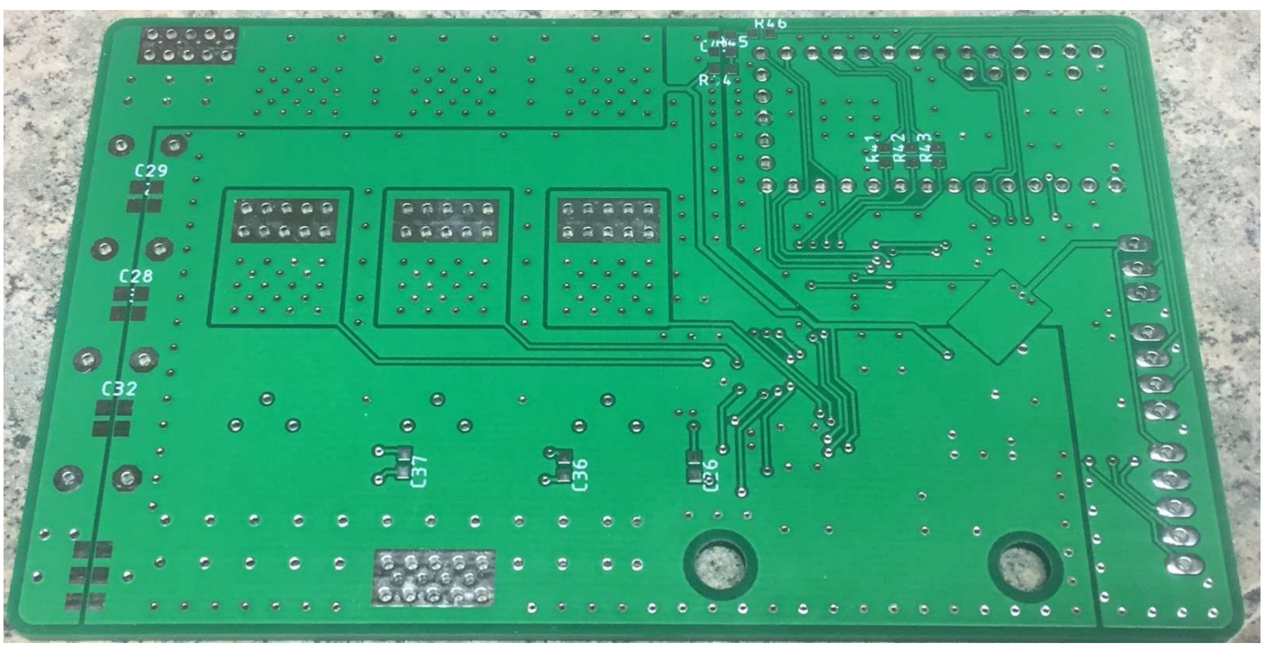


Board Pictures

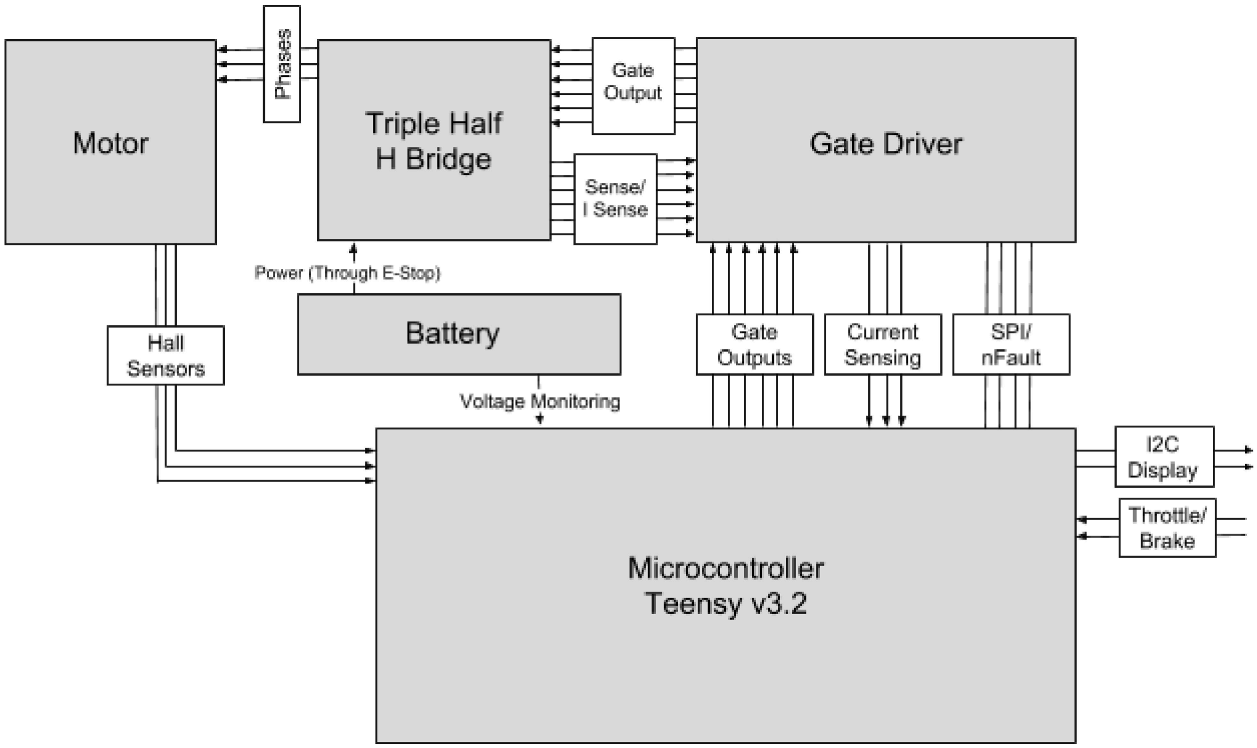
Front Side



Back Side



Control Diagrams



Information displayed to the operator includes:

- Battery Voltage
- Current Draw/Wattage
- Calculated Speed
- Any faults or errors

The motor will only spin if these conditions are met:

- Controller is powered. (Not disabled by emergency stop, or operator)
- Battery voltage is over 40 volts.
- No phases current is over 30A for more than 1 second.

In the event of a driver fault:

- The fault will be displayed to the operator, and the microcontroller will attempt to reset the driver once every 500 ms.
- On the fifth attempt to reset, the microcontroller will stop attempting to reset, and display a fatal fault to the operator, and the motor will be allowed to coast.

Software Documentation

Overview

Our software is designed to control the 3-phase motor efficiently. In order to achieve this, we utilize a hall sensor based feedback control system.

Logic Flow

1. The microcontroller receives rotational position input from the hall sensors.
2. The microcontroller calculates the current active phase based on motor position.
3. The relevant MOSFETs pertaining to the current active phase are switched by the gate driver.
4. Based on feedback from the hall effect sensors, the RPM of the motor are calculated based on the wheel diameter and a the vehicle speed is displayed back to the driver.
5. Inputs from the three phases are monitored and motor power is cut if an excess current is pulled to prevent damage to the motor.
6. Deadman safety device is polled to ensure operator control of the vehicle. If none is detected, the program will disable the motor.

Libraries

In order to improve software readability and modularity, we created two custom motor control libraries to assist with software development. The collections of pre-written code can speed up development, make identifying issues easier, and make code easier to read and understand.