ECE 491 Mechatronic Systems Design
Project Progress Report
Team members: Abhra Sinha, Pratik Walawalkar, Eshan Gupta

1. Team Name

Scavengers: We chose the name Scavengers as our team spirit was to capture down every single black line in the track and hunt it down (Not Literally, just detect it). We wanted to capture every single obstacle in our way and complete the track accurately. Our spirit aligns with the strategy that we had adopted, to move slow but make precise and accurate turns and complete the track, which we did with precison.

2. Overall Approach

Our focus was keeping the car on track with slow speed at first and then gradually increasing the speed of the car without compromising the navigation control. To implement this, we gathered the line camera values at a very wide angle to keep a larger window. The PWM output was kept at 25% initially to test the car.

For the Round 1, we aimed to move our car on the track correctly with optimized motor speed and camera calibration. However, we encountered several issues the days leading to first round and during the first round, these are:

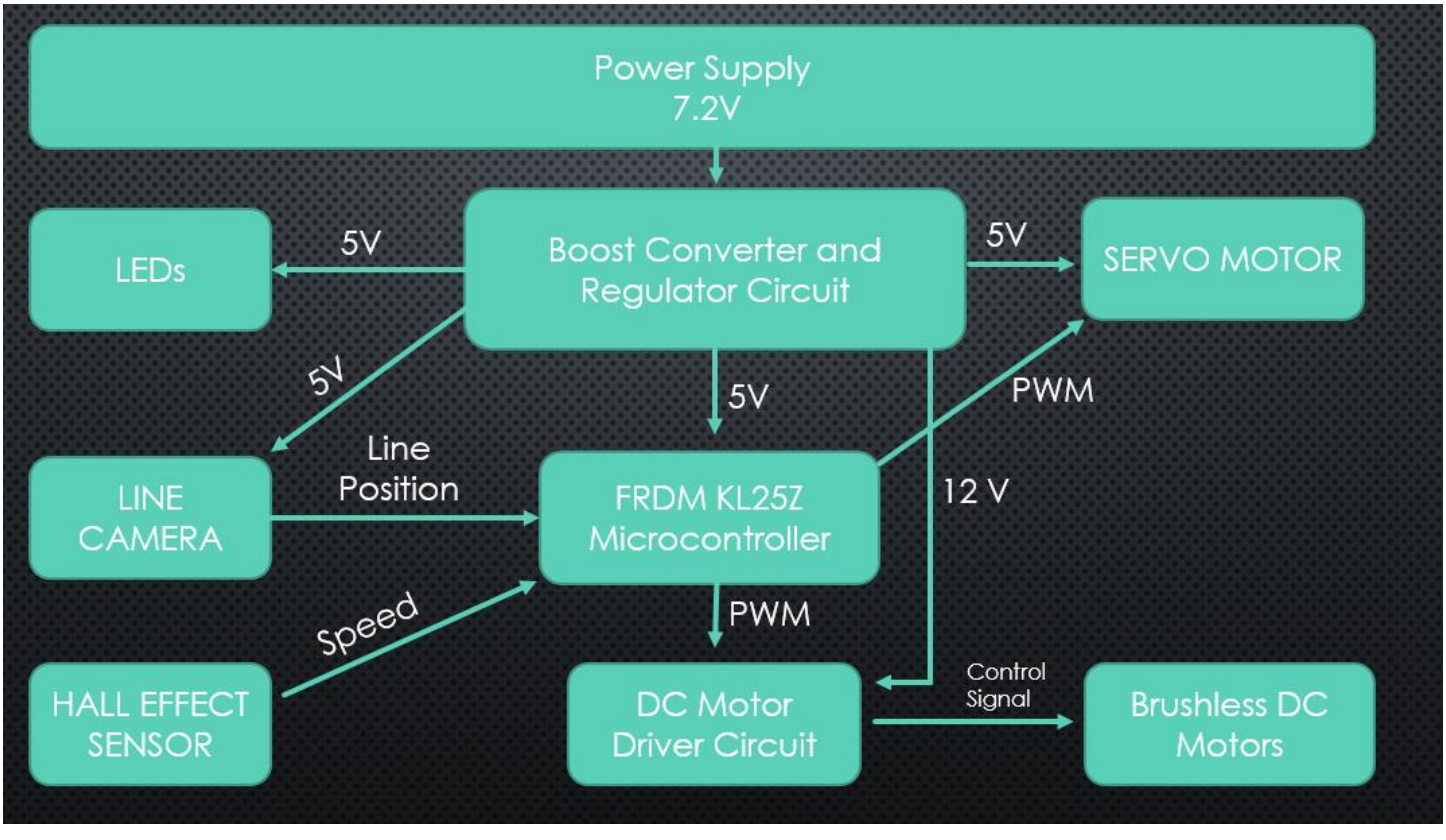
1. We faced some issues in choosing the right component parts from a recognized vendor but after going through all the datasheets and research on vendor listing of parts. We could decide and order the right components from the vendors.
2. We were not able to decide the number of planes to be added to the PCB but then john helped us decide the right number of planes.
3. We found out that the full H bridge FETs breaks down when we supply 12V input to the Max620 IC so we are have opted the full H bridge IC (L298) to control out dc motor. This was resolved by selecting L298D as our Motor Driver.
4. Line camera output was not very clear and we decided to mount LEDs to improve the slope detection
5. While Soldering, our Boost Converter LT1370 had a short connection between two terminals and while testing the PCB, we couldn't get the 12V output, we rectified this by reSsoldering it and making a clean solder joint.
6. Our algorithm was based on the slope detection and then based on the slope edges running a if else statement to move the servo to the desired positon, this worked for some cased but it gave wrong outputs in certain cases.

For the Round 2, we increased the duty cycle of the PWM to 50%. Our internal track of the PCB connecting the supply to the input of Boost converter was burnt out so we soldered the connection externally.

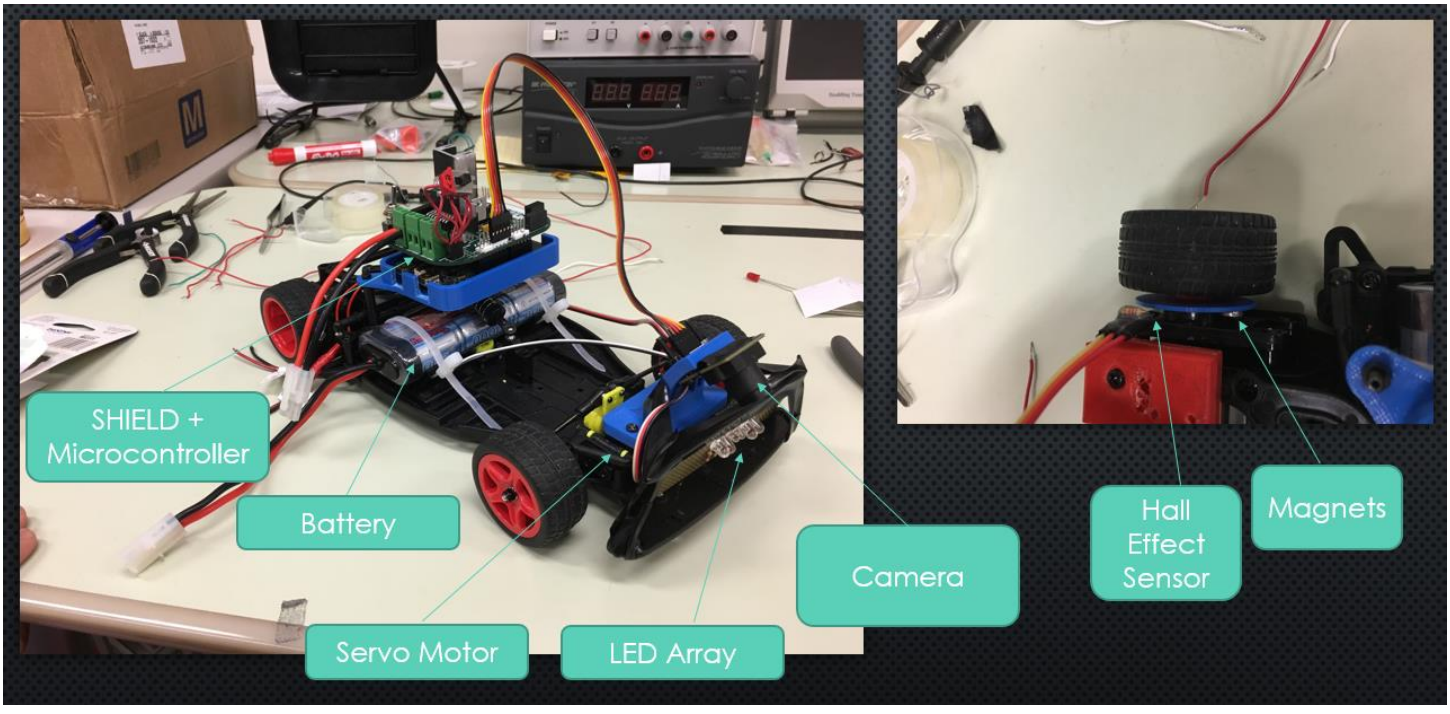
1. We had certain cases of excessive current draw from the servo and the Motor causing a stall in the system, we rectified this by running a simulation in LTspice and replaced our output capacitors of value 22uF to 220uF. This accounted for the instability and provided enough boosting to account for the stall situation.
2. We changed our servo motor detection by implementing a PD control and we had difficulty figuring out the proportional constant values, after repeated simulations and test runs we were able to settle on a precise Kp value for perfect steer.
3. We implemented a Hall effect sensor for speed measurement and we had trouble reading the values and implementing the logic in our code, however after several trials and a little help from the internet, we were able to detect the speed of the wheel during track conditions which turned out close to 1.5ft/s.
4. We changed the window of line detection depending on the track location and this helped in detecting sharp turn.
5. We had to attach a heatsink to our motor driver circuit to cool down the motor driver as it was responsible to handling both the motors.

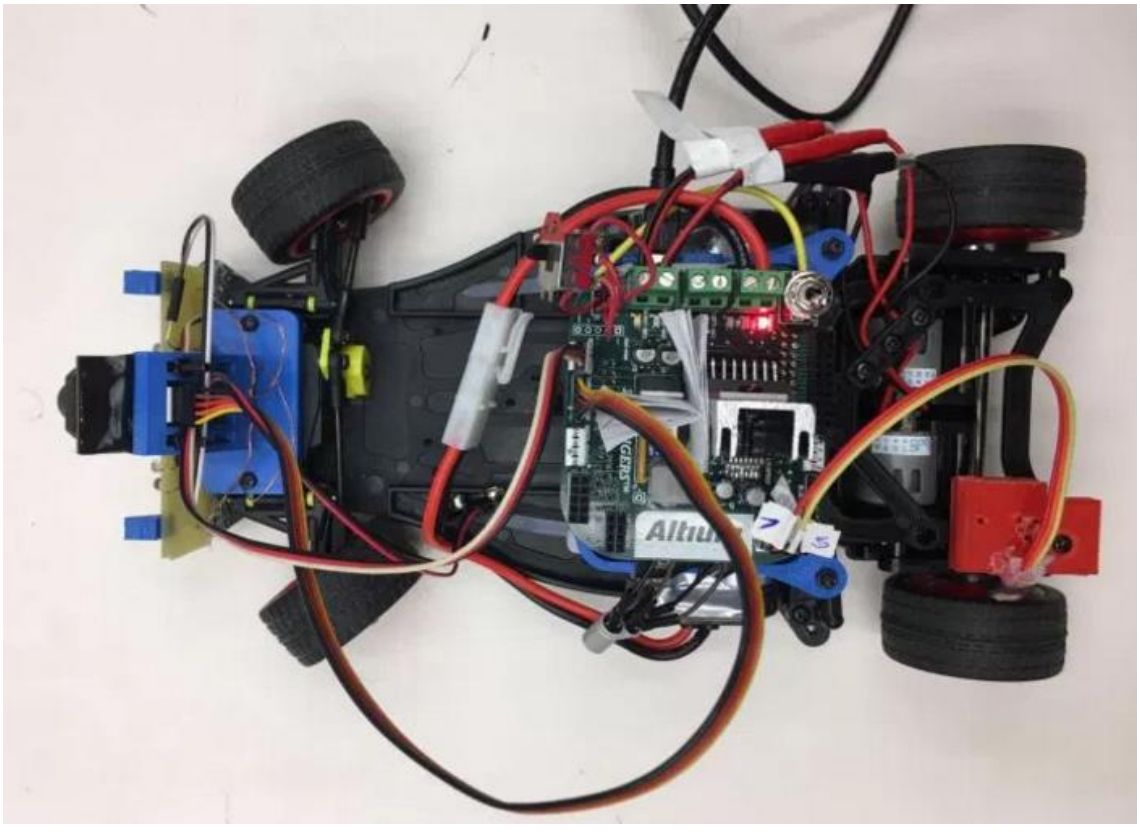
3. Hardware Design

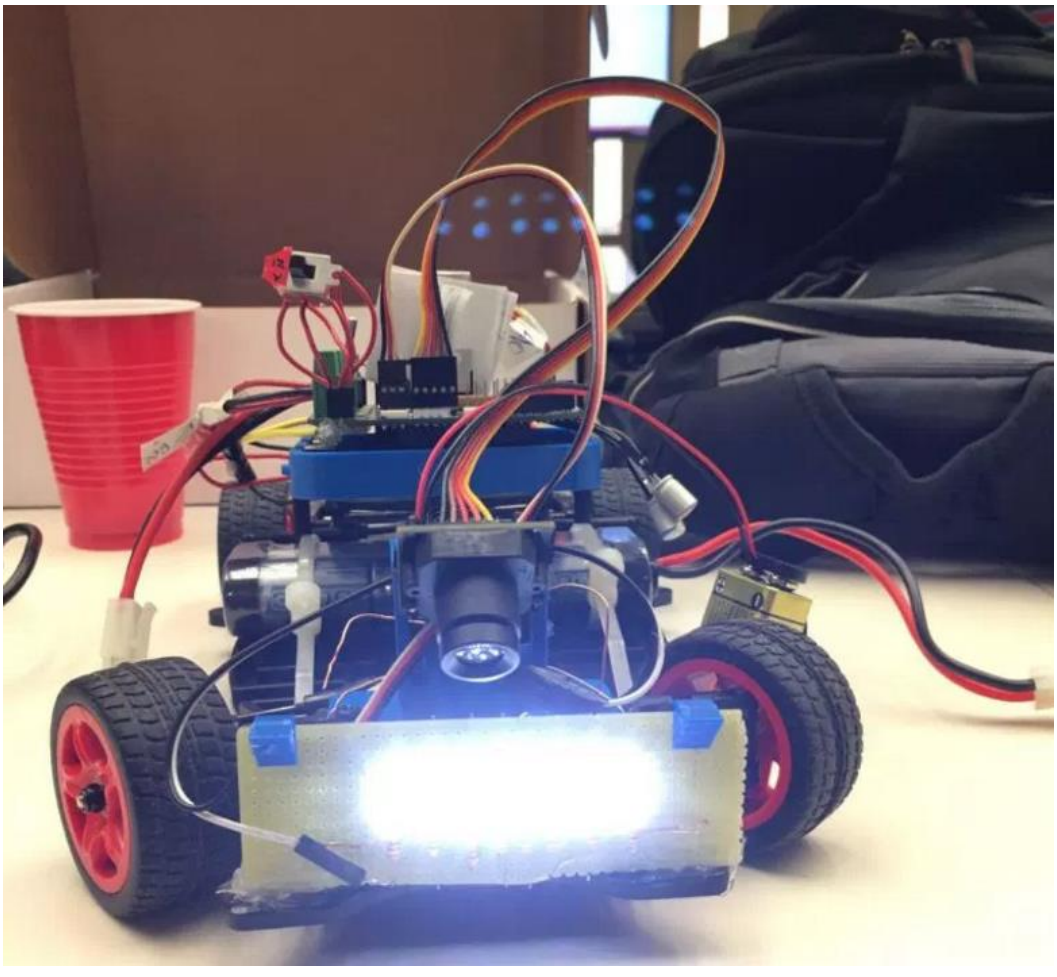
3.1 MCU block diagram



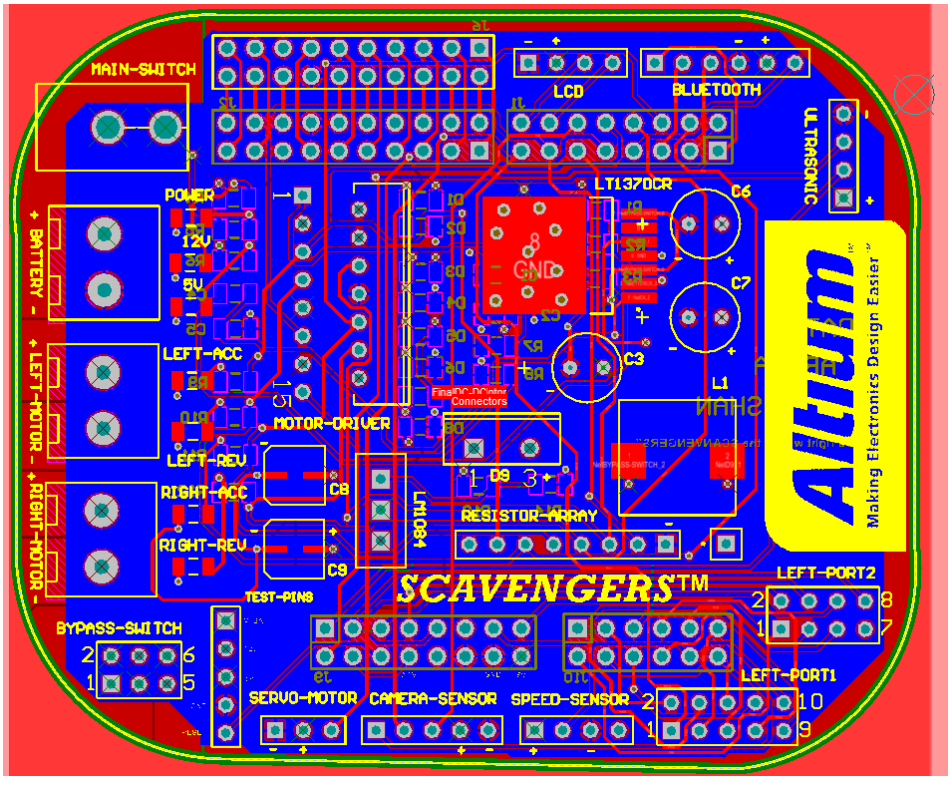
3.2 Detailed Drawings/Pictures of Vehicle

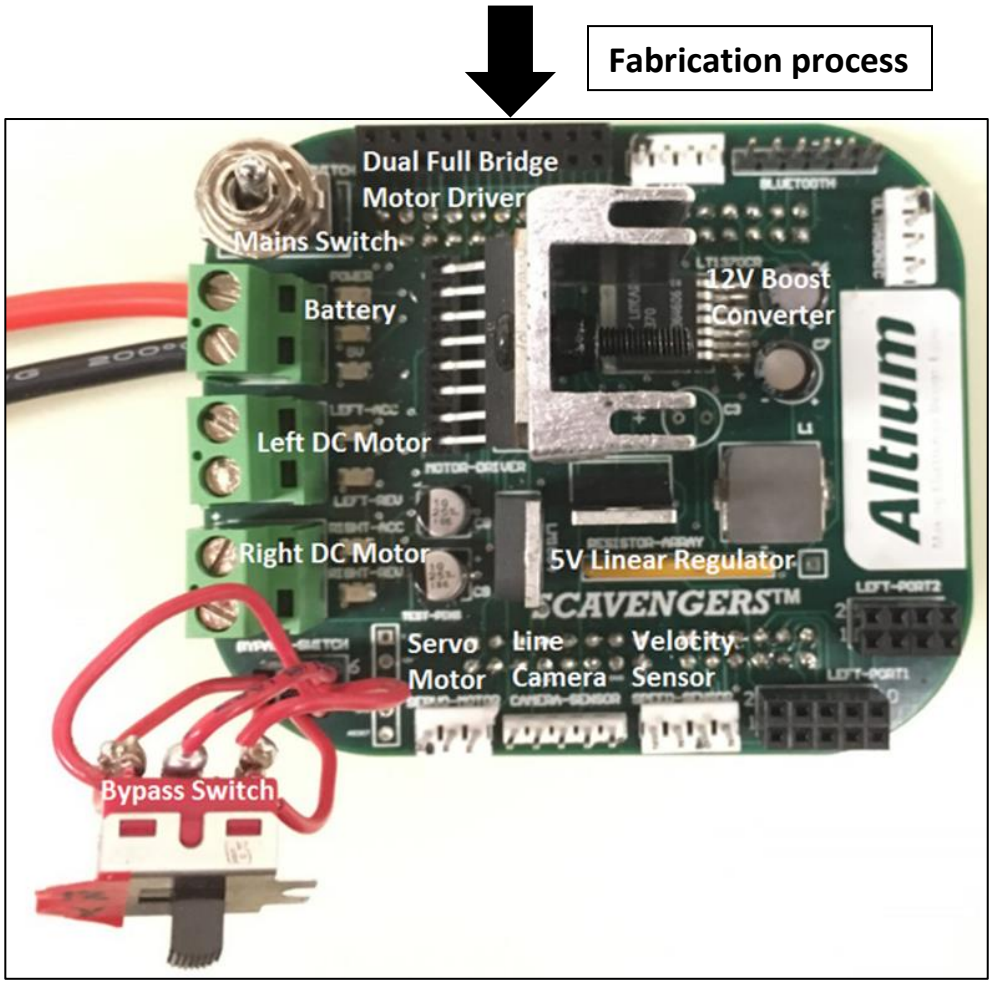




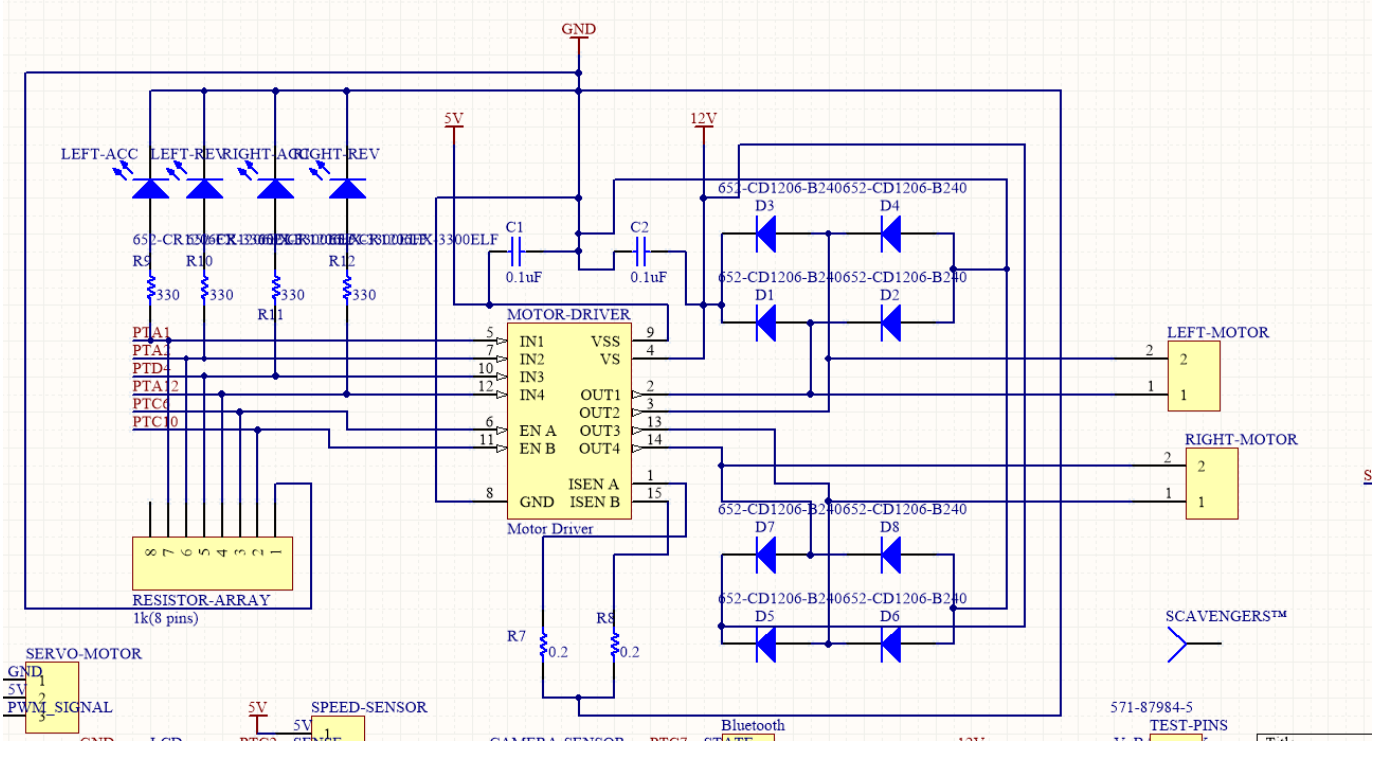


Our own designed Shield board:





3.3 Motor Drive Circuit Schematic



We implemented dual full bridge motor driver- L298N in our design. Operating voltage of motor driver was set as 12V.
Reason: Only one motor driver circuit was driving our two DC motors and to avoid any possible voltage drop when whole circuit is working under high load current draw. Also have freedom to run DC motor at higher PWM pulse for 12V.

4. Software Design

First, we tried using Kinetis Design Studio for programming FRDM KL25Z. But working with timer/interrupt functions and debugging was little bit complex and time consuming. Hence, we decided to use an alternative: **mbed** online compiler developed by ARM, for writing our firmware.

Our approach:
Correct steering action: More focus in fine tuning the values of proportional term in PID algorithm
Moderate speed: Velocity sensing algorithm using hall effect sensor to control the motor speed.
Line following accuracy: Tuned line camera to get good exposure and acquisition of correct values of line camera output and storing those in 128-bit array.

Challenges faced:

Hardware issues

- **Incorrect datasheet reference**: While designing 5V voltage regulator, we confused the resistor bridge logic for 5V output with adjustable voltage regulator. Hence output voltage was than 6.2V instead of 5V. Rectified the resistor logic.
- **Mistake while SMD Soldering**: Two pins of boost converter were shorted while smd soldering. Need to desolder and rectify the same.
- **Incorrect pin assignment**: Different PWM pins were assigned in board, which were not supported by mbed compiler. Only one PWM pulse was given to both the DC motors. **(This majorly impacted our high speed of motor and restricted us to keep moderate speed in round 2 competition to atleast follow whole track accurately).**
- **Excessive heating**: Motor driver faced excessive heating as it only one motor driver was running two DC motors at once. Designed custom heat sink to reduce power dissipation and save IC from burning up.
- **Huge voltage drop across DC motor**: Excessive current draw from servo motors lead in significant drop in boost converter output voltage from 12V to 6-7V. As 12V was also driving our motor driver IC, this voltage drop would limit current flow in DC motor hence slowing it down. Hence need to replace output capacitors to maintain voltage near to 12V.

Software issues

- **Board Firmware upgrade**: Needed to upgrade to latest FRDM board firmware as the drivers installed were incompatible with Windows 10 OS. Externally flashed .s19 file in bootloader mode and upgraded to latest firmware version
- **No debugging feature:** mbed online compiler didn't support debugging feature. Had to use **Keil** software along with serial debugger.
- **Problem in hard left turn**: Initial value of the for loop was adjusted for window of camera output values. Adjusted Kp (proportional term) and initial error values.

5. Conclusion

- Electronics aren't 100% reliable and can give up at the last moment
- Practical scenario is much different from ideal
- Higher range of error to be considered when selecting certain components. PD control simplified our velocity control and steering algorithm
- Triple read all the datasheets and understanding the values written in them.

Round two result: **1.7ft/sec** completed 71 ft in 41 sec. Our own shield board was future ready for further expansions and also, we planned to include wireless connectivity to the car for OTA debug. Our SHIELD was the densest PCB board in the class.

Click here for video of round 2 competition