Freescale Cup Challenge 2015

Cody Smith, Jack DeGonzaque

Rochester Institute of Technology
Rochester, NY

cds7494@rit.edu
jsd6575@rit.edu

Abstract— **The objective of this report is to define and analyze the steps taken to build an autonomous racecar. The components used to construct the racecar, the design of the circuits and algorithms used to make the car function will be elaborated. The hardware bill of materials and the key challenges will also be discussed.**

I. INTRODUCTION

The Freescale Cup challenge is an exhibition for teams to put their mastery skills of embedded software programming and knowledge of basic circuitry to the test as they build an autonomous racecar. The car must be able to not only complete the track as the car stays within the boundaries, but it must also do so in the fastest time in order to advance in the competition. The autonomous racecar heavily relies on two key factors in order to function. One factor is hardware. Sensors are imperative to collect data to feed into the microcontroller to be interpreted. For our team's car a Freescale K64F was chosen. The primary sensor to use for such a competition is the line scan sensor. Secondary sensors, such as encoders and IR emitters are not exactly necessary, but extremely useful when designing an algorithm for the racecar to follow. These sensors provide a way to detect the speed at which the car is currently traveling and the marker at the end of the track to signal a complete lap around the track. Another key concern is finding resources to power all of the sensors. The K64 microcontroller has a limited number o voltage sources as well as the actual voltage level supplied. For example, the K64 microcontroller can provide up to two 3.2 V and one 5V source while DC motors require much more voltage than what is provided and two cameras require 5V sources each.

The second key factor is software. Once it is known that the sensors function properly, being able to interpret such information provided by them is equally as important. Without correctly interpreting the data, having the sensors in the first place would be rendered as useless. Software concerns to be addressed for the sensors and hardware components are configuring General Purpose Input/output (GPIO) pins. Data can be sent and received by configuring the GPIO pins on the K64 to do so. Another design constraint with configuring GPIO pins is that most of the pins allocated on the board are pre designed to be used for certain modules. These modules are the Flextimer Module (FTM), Periodic Interrupt Timer (PIT), Programmable Delay Block, System Tick Timer and many more. These timing modules play a key role in sending signals to peripheral sensors and the motor board.

II. DESCRIPTION OF COMPONENTS

The components used for the 2015 competition are as follows: the standard Freescale Cup kit provided by the lab instructor, 4 3-D PLA molded mounts, and jumper wires. The Freescale Cup kit contains a 1/18 Scale Model Chassis, a 7.2V battery, two 7.2V DC Motors propulsion system, a servo motor for steering, a Freescale Development board (K64F microcontroller) to act as the control system and two CMOS Camera line scanner sensors with lenses and ribbons.

The purpose of the DC motors is to convert energy into rotational movement. Depending on the orientation of the current, the motors will either rotate clockwise or counterclockwise. When there is a force opposing the motor, the rotation of the wheels decrease and the current going through the terminals of the motor drastically increases. As stated earlier, the K64F microcontroller can only supply 5V. The motors require at least 7.2V. To resolve this issue, a motor control board is used to supply the power to the motors. The motor board contains an Half-bridge (also known as an H-bridge) that will allow for a pulse width modulation (PWM) supplied from the microcontroller to dictate the movement of the motors. Fig. 1 below shows the schematic of the DC motor circuit. Fig. 1 below shows the inner workings of the DC motors. Current is fed from ports OUT1 and OUT2.. The direction of the current dictates the direction in which the magnet rotates.

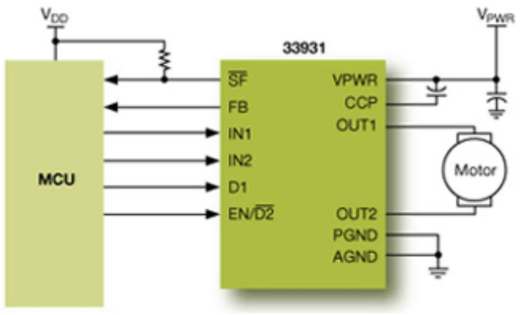

Figure 1: DC Motor Schematic

Fig. 2 below shows the schematic for the Version A motor board provided in the Freescale Cup kit. The supplied 7.2V battery is connected to the positive and negative terminals of header J3. The servo motor's ground and supply voltage is supplied by header J6. Headers J1 and J2 connect to the positive and negative terminals of each DC motor respectively. The positive and negative terminals at header J7 are connected together via wire. Header JP1 is not needed to supply the camera since the K64F

can supply all necessary inputs directly from the microcontroller to the cameras. Header J3 contains the positive and negative terminals to be supplied to a peripheral device, such as the microcontroller. The positive terminal dissipates 7.2V. Luckily, the K64 specifically has an input pin to supply the microcontroller anywhere between 5 to 9V.

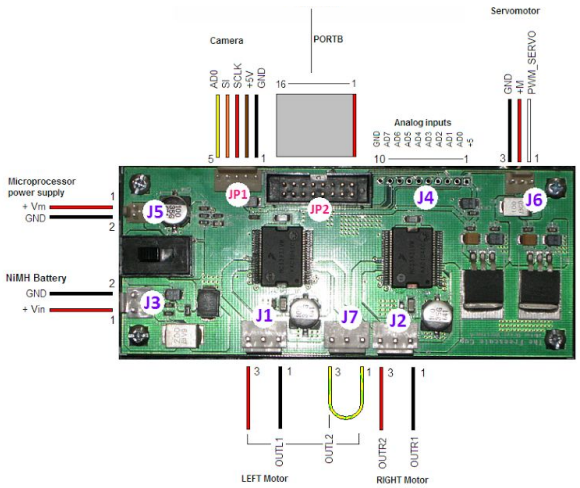

Figure 2: Version A Motor Board Schematic

Fig. 3 below shows the pinouts for the JP2 header shown in Figure 2.The H-bridge at the top of the schematic is labeled with ENDL, ENDR, PWML, PWMR, and PWM SERVO for the JP2 header pins. As stated before, the enable pins for the left and right motor are supplied by the motorboard from the battery. The PWM signals for the servo and the DC motors are supplied by the microcontroller.

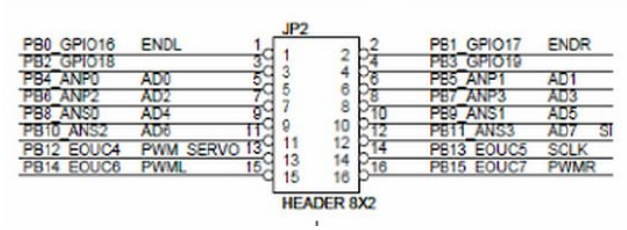

Figure 3: JP2 Header Schematic

The servo motor is a continuously rotating motor that can be driven to a set position by PWM signals. The limiting factor to the Futaba motor used is that it can only update in 20ms increments. Fig. 4 below shows the setup of the servo with respect to the front wheels. Notice in the picture below, the servo cannot make a full rotation due to the axles connected to the servo and wheels. It is key to adjust the PWM signals so you can achieve a full range of motion turning left or right without jamming. Once the PWM signals are configured to move servo in both directions, slowly increment or decrement the duty cycle in each direction until the servo begins to jam. Once it begins to jam, the previous duty cycle value is farthest one can go in that direction.

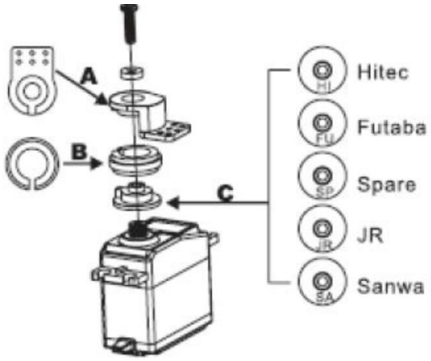

Figure 4: Servo Configuration

The line scan cameras are perhaps the most essential component. Fig. 5 below shows the schematic of the line scan camera. This

camera is a 128 pixel linear image sensor with a focusable lens and a built-in amplifier stage to improve the differentiation between white and black colors. In order to function, the camera needs and clock, an SI pulse, ground and a 5V supply voltage. The SI pulse acts as a shutter to tell the camera when to begin read the current values of the 128-bit array that represents the 128 pixels. After all 128 pixels are read, the SI pulse will then be triggered high again to start another read.

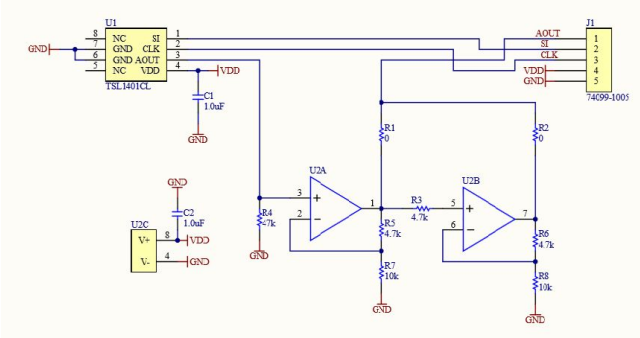

Figure 5: Line Scanner Camera Schematic

The final set of components necessary to bring the autonomous racecar to life are the mounts. Due to the difficult placement of the battery, suspension and servo, there is a very limited space to neatly place the other components in a good location. A viable option to resolve this issue carefully designed mounts produced by 3D printing. By using Autodesk's Inventor, safe and secure mounts for the K64F, Motor Control Board, and cameras were designed.

The K64F chassis mount is shown below in Fig. 6. This mount sits above the two DC motors, to be held in securely by small screws.

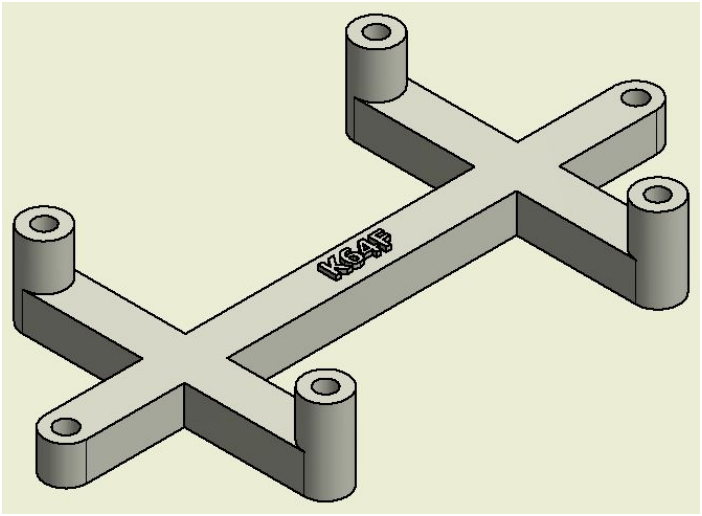

Figure 6: K64F Chassis Mount

The mount for the Motor Control Board mounts to existing screws holding together the rear motor cage, and places the board off the back of the car in a way that sort of resembles a spoiler on a car.

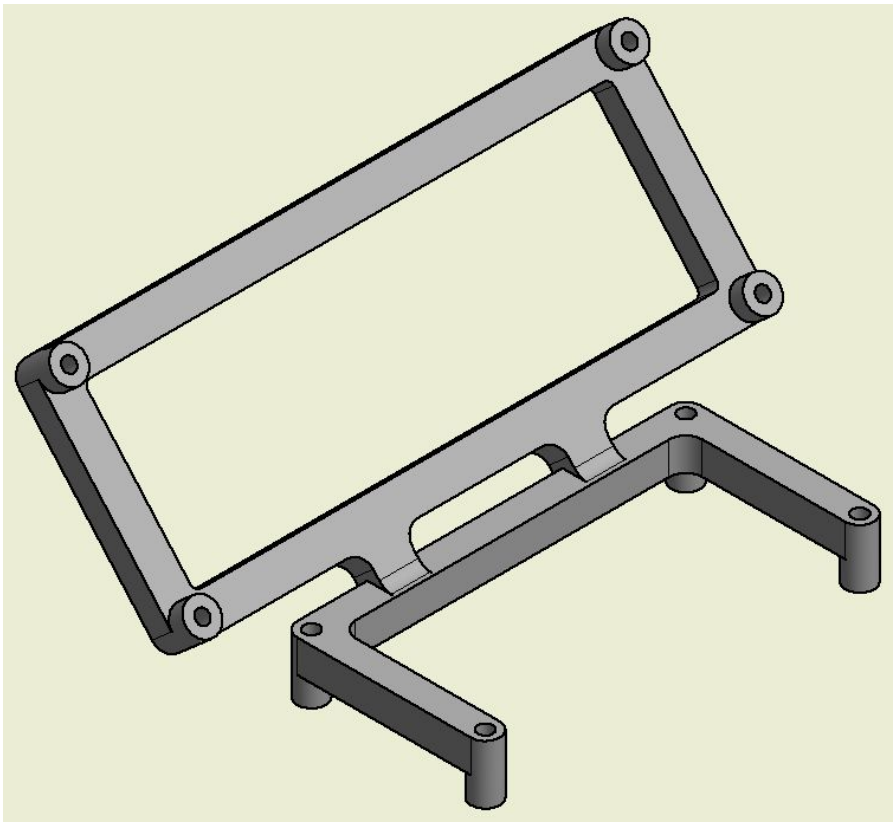

Figure #: Motor Control Board Rev.A chassis mount

The camera mount are shown in Fig. # below. Fig. 7 contains the base mount that is screwed in by the same vertical screws the servo motor is fastened in by.

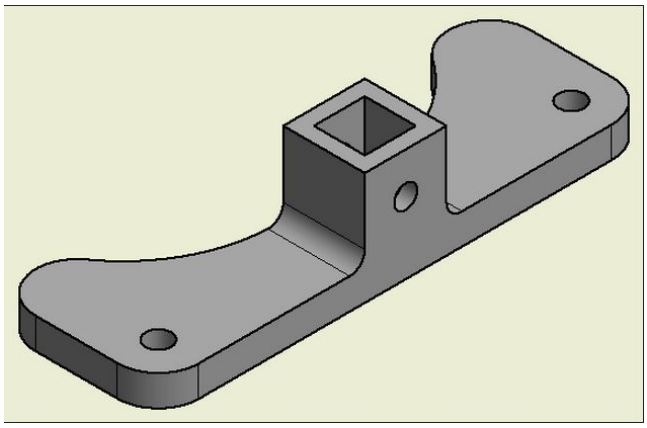

Figure 7: Camera Base Mount

The camera post that slides into the base and vertical bar mount that connects to the post are shown in Fig. 8.

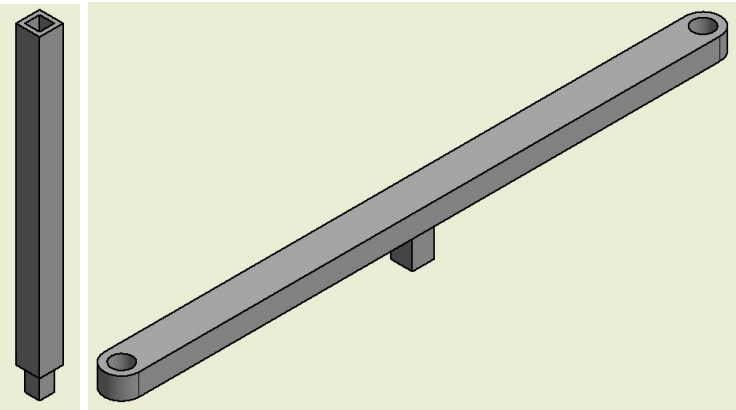

Figure 8: Camera Post and Vertical bar

Fig. # below shows the 3D mount that is screwed to the camera and placed into one of the two holes in the vertical bar mount in Fig. 9.

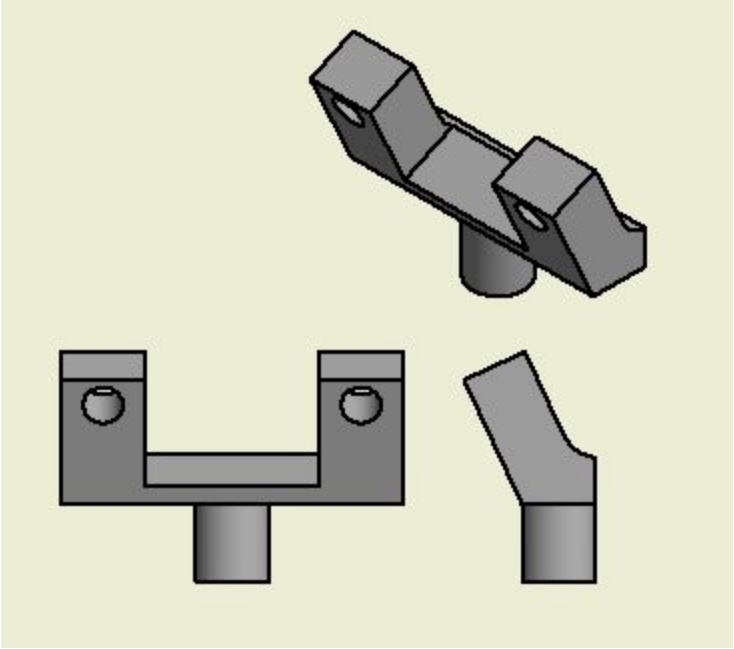

Figure 9: Camera Mount

III. DESIGN OF CIRCUITS

Fig. 10 below shows each connection from the cameras, to the microcontroller, as well as from the microcontroller to the motor board. The SI pulse and clock input for the camera are produced from pin D2 (PTB9) and D4 (PTB23) from the microcontroller. The output of both cameras run to the ADC0_DP1 pin and the ADC1_DP1 pin on the microcontroller. These pins is an Analog to Digital converter pin. By soldering wires to make two connections to the ground, 5V supply, clock and SI pulse, added connections for the second camera were minimal. The 7.2V battery connects to the terminals at header J5 on the motor board. The microcontroller is powered from header J3 on the motorboard to the Vin 5-9V pin and ground pin respectively.

The PWM signals that control the servo and DC motors are shown heading into header JP2 on the motorboard. The DC motor PWM signal originates from pin D7 (PTC3) for the left motor and D9 (PTC4) for the right motor on the microcontroller. The servo motor PWM signal derives from D13 (PTD1) on the microcontroller. That very same PWM signal is feed through header J6 to the PWM input terminal of the servo.

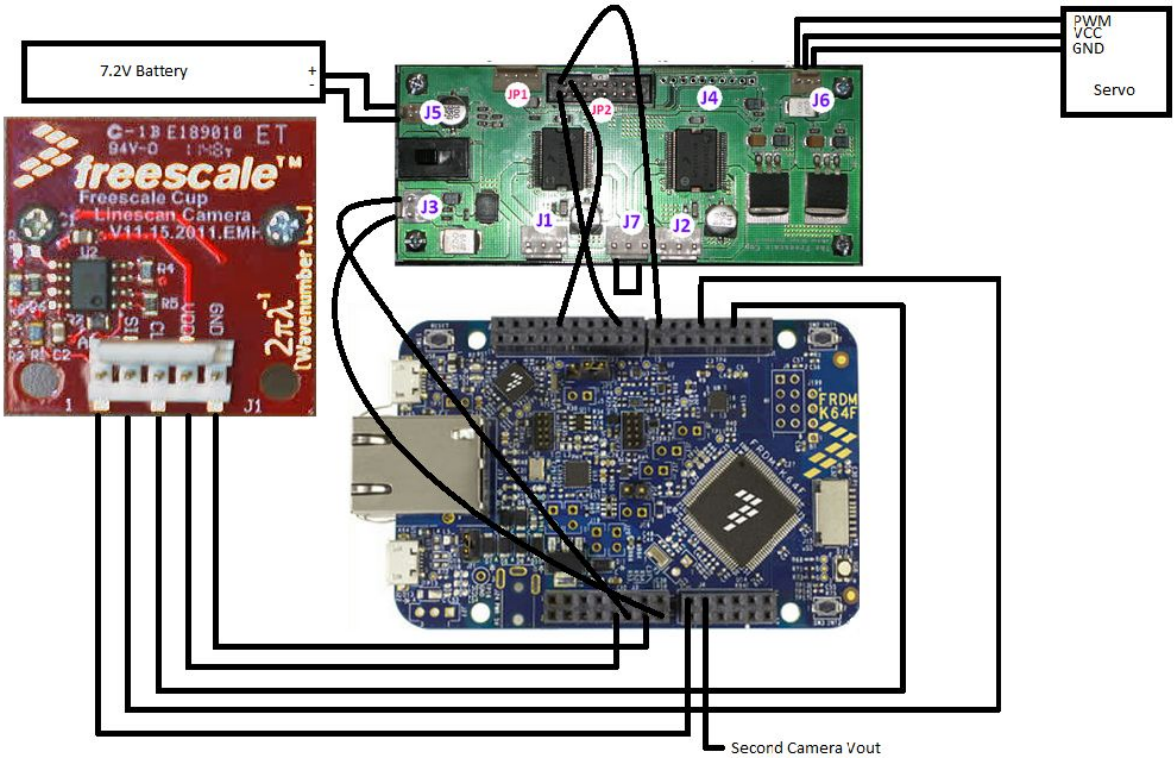

Figure 10: Racecar Connections

IV. K64F HARDWARE UTILIZATION

The hardware interfaces of the Freescale K64F were crucial to making a working car. To generate the PWM signals to control the motors, servo, and camera signals the on-board Flex-Timer modules (FTMs) were utilized. The initialization of the FTMs was similar for each case.

First, the clock is enabled for the module by writing to appropriate SIM_SCGC clock enable bit. Then the write protection in FTM_MODE was disabled. Next, the FTM_CNT (Counter value) and FTM_CNTIN (Counter Initial Value) are set to 0. The status and control register is configured next and has different values depending on what the FTM is being used for. For the PWM signals used to control the motors and servo, MSB and ELSB are set high and ELSA is set low to put the FTM in Edge-aligned PWM, High-true pulse mode. For the camera signals, the status and control register was set to enable overflow interrupts by setting TOIE high.

IV. DESCRIPTION OF ALGORITHMS

There were many algorithms that were crucial to the success of the freescale car. Starting with the input from the camera was a 5-point weighted averager was implemented to eliminate noise. The formula for the averager used is shown as figure #.

$$x[n] = 0.4 * x[n] + 0.2 * x[n-1] + 0.2 * x[n+1] + 0.1 * x[n-2] + 0.1 * x[+2]$$

Figure #: 5-point weighted averager

After the averager has removed noise, the next important step is to do edge detection. This was achieved by computing the derivative of the image using a naive derivative formula, which is shown as figure #.

$$y[n] = x[n+1] - x[n-1]$$

Figure #: Naive derivative

The derivative of the image shows exaggerated peaks or troughs when the white of the track converts to any other color. By using a shape detection algorithm to find large variations, and therefore edges, a position of the edge of the track in the image was determined. If no edge was detected, the image processing returns a negative position index to indicate that the data from the camera is unreliable for position acquisition and correction.

The position detection algorithms decides the car's position by either averaging together the two valid positions from the camera, taking the one good image's position if one image failed, or goes to a third state for when both cameras don't detect an edge. A correction can then be calculated based on the given position.

The way to make corrections was one of the principal challenges in this exercise. Given our position, our correction algorithm uses a linear relationship to decide a new position of the servo. We used compile-time constants to describe four linearly related points which create a correction formula. By adjusting those four points the severity of corrections can be adjusted. There is also a linear formula which relates servo position to motor speed. This is also adjustable using compile-time constants.

V. HARDWARE BILL OF MATERIALS

All of the material for the camera mounts and the use of 3D printers were provided by RIT in the Construct Lab. Even though the encoders were not implemented due to time constraints, hall effect sensors and neodymium magnets were purchased. The purchased materials are listed below in Fig. 11.

Item	Quantity	Price (US Dollars without shipping)	Source
480-1991-ND 13.62000 27.24T SENSOR MAGN SS HALL EFFECT	2	13.62	www.digikey.com
Tiny Craft Hobby Neodymium Rare Earth Super Magnets 1/8 x 1/16in (x50)	2	8.99	www.amazon.com

Figure 11: Purchased Parts

VI. KEY CHALLENGES AND REFLECTION

There were several obstacles faced in the journey to build the autonomous racecar that primarily dealt with time and the K64 microcontroller configuration. From a competitive standpoint, either the K64F or MKL25Z microcontroller can be used as the control system to the race car. Since a K64F was already owned by all parties involved for the Kings of Calibration, the choice was simple. However, there was very little sample code and documentation specific to the Freescale Cup for th3e K64F. On the other hand, other teams chose to use the MKL25Z and they were immediately rewarded with code and development environments to configure and control their peripheral components (servo and DC motors) at an instant. There was a solid foundation to build off of for the MKL25Z teams. For the K64 teams, it had taken a couple weeks of time to manually configure the peripheral components and to ultimately reach the same level of progress the MKL25 teams started with. To the months leading up to the competition, it was not clearly stated whether the competition was a line following competition or a boundary following competition until a couple weeks beforehand. Time was lost working under the assumption of a line following competition since new 3D mounts had to be printed to accommodate another camera and the K64F microcontroller had to interpret the images from both cameras in a sensible fashion.

As much as it seemed to be a nightmare building a foundation for the K64 from the ground up and falling behind compared to the MKL25 teams, it was quite the learning experience which ultimately became beneficial. The Kings of Calibration dedicated vast amounts of time deciphering the K64 reference manual that we became aware of the K64's capabilities. Once the core peripheral components were fully operational, accommodating to secondary components, such as encoders were a non-issue. The only reason the encoders were not implemented in the final build was simply due to the fact that they were not exactly reliable. There were several occasions where the encoder sensor would not detect the magnet glued to the inside wheel passing. A similar issue occurred when we tried to cover the end game scenario (passing over two markers in the middle of the track). Two OPB745 sensors were mounted to the front bumper of the car directly pointing downwards. In testing, we discovered that the light settings of the environment varied too much for the sensitive sensors to only detect the markers. They were deemed too risky to implement when it came to competition time and time dwindled too swiftly to find a healthy alternative.

With due consideration to the two month time constraint and the lack thereof foundation to build upon for other teams from a competitive standpoint, the project was a success. Majority of the time spent on the project was making sure the racecar can smoothly make its way through every scenario of the track, especially the intersections and ramps. After that functionality was solidified, very little time was left to fully implement PID to the motor speed and servo position.