# SoutheastCon Team A

Project Proposal and Statement of Work

| Date | Revision | Comments |
|---|---|---|
| October 17th, 2014 | 1 | First edition |
| | | |
| | | |
| | | |

Team Members
Nils Bjeren
Ryan-David Reyes
Julian Velasquez
Kurt Marsman
Chris Lewis
Donovan Carey
James Pace

Faculty Advisors
Dr. Bruce Harvey
Dr. Michael Frank
Dr. Linda DeBrunner
Dr. Carl Moore

## Project Executive Summary

IEEE SoutheastCon is the annual conference for Region 3 of the Institute of Electrical and Electronics Engineers. This event includes several competitions, one of which is the hardware competition. The purpose of the project outlined in the present report is to compete in, and win, the 2015 SoutheastCon Hardware Competition. In order to do this, an autonomous robot will be designed in accordance with the competition rules.

This year's competition, held in Ft. Lauderdale, has a "road trip" theme. The robot will need to navigate along a course represented by a white line on a black background. Along the course, 4 different classic road trip toys will be "played with." These include a Rubik's Cube, the Simon Says game, an Etch-a-Sketch, and a deck of playing cards. In order to win the competition, the robot must complete the course as quickly as possible, completing the challenges without error in less than 5 minutes.

Team 1A's robot will employ a combination of custom designed components in order to complete the challenges. A fully functioning prototype robot will be available by the end of December 2014. That will allow time for tweaking until the conference in April 2015.

The engineers on the team are confident that they will be able to create a robot that performs well within the required specifications. Within this document are the outlines of the design process, the statement of work, the testing plan, as well as the preliminary risk assessment for the project.

# Table of Contents

## Contents

# 1 Introduction

## 1.1 Acknowledgements

The team would like to gratefully acknowledge Ramiro Velasquez and INNOVAtek for their generous $500 donation towards the project, as well as the $750 provided by the FAMU/FSU College of Engineering. In addition, the continued advice and suggestions of the ECE Senior Design faculty have been invaluable in the design process

## 1.2 Problem Statement

The purpose of this project is to build an autonomous robot that will win the 2015 SoutheastCon Hardware Competition. In order to complete this task, the robot will have to be able to move along a white line on a black background, as well as complete four different "road trip" themed challenges. These challenges are: twisting one row of a Rubik's Cube 180 degrees, playing Simon Says for 15 seconds, drawing "IEEE" on an Etch-a-Sketch, and picking up a single playing card from a deck of 52. In addition to this, the robot must fit within a 1' by 1' by 1' box at the beginning and end of the course.

The robot chassis will be made out of aluminum. For propulsion, four DC motors will be used together with Mecanum wheels. The line will be detected and followed using an array of infrared sensors. Controlling the robot will be an Arduino Mega microcontroller together with other Arduino microcontrollers that will manage different subsystems as necessary. Hence, most of the programming will be done in the C language. Exploiting synergies between pairs of challenges, the robot will have two main subsystems, in the form of protruding arms, in addition to navigation. The first subsystem will complete the Etch-a-Sketch and Playing Card challenges. This arm will twist the knobs on the Etch-a-Sketch using two DC motors, with sticky tape providing torsional friction. The sticky surface can be re-used in order to pick up the playing card. The second arm will play Simon Says and twist the Rubik's Cube. This arm will twist its end effector in the horizontal plane using a 360 degree servo in order to hit each button on the Simon Says accurately. The same twisting motion can then be re-used in order to twist the Rubik's Cube. The robot will be powered using lithium polymer batteries.

## 1.3 Operating Environment

The environment in which the product will be used is the game board of the 2015 SoutheastCon Hardware Competition. There is currently little specific information about the venue, but it is safe to assume there will be many spectators, as well as several competition "heats" occurring in parallel. Therefore, two major factors that need to be considered are sound and light interference. Sound interference could be caused by announcements, random conversations, and competitors cheering for their robots. This could interfere with the proper functioning of sensors that rely on sound, such as microphones. Light interference could be caused by, for example, the use of cameras during the competition. In addition, the lighting of the venue will not be known in advance, so it is necessary to plan for the “worst case scenario,” i.e. the lighting scenario where the robot performs at its worst.

## 1.4 Intended Use(s) and Intended User(s)

The intended use of the project is to successfully build a robot from scratch that can compete in SoutheastCon 2015. The robot will have to be able to autonomously start, navigate the track, play Simon for 15 seconds, draw “IEEE” on an Etch-a-Sketch, rotate any row of a Rubik’s cube 180 degrees, and pick up a card from a deck of cards, taking the card to the finish line.

The intended users of the project will be the engineers who built it, as they will be the ones taking the robot to the competition. The whole FAMU-FSU Electrical and Computer Engineering Department will be represented by the team and its robot.

## 1.5 Assumptions and Limitations

The design is based on the following assumptions. Branches and corners on the white line will be deterministic, i.e. if sensed correctly, it is impossible to mistake a branch for a corner and vice versa. Sufficient time will be allotted between runs during the competition to charge/replace batteries, as well as to replace sticky surfaces (for example on the Etch-a-Sketch arm). The robot will not need to function for more than a total of 30 minutes before the battery can be fully recharged. As the robot is made according to the most up to date competition rules, a major assumption of the current design is that the rules will not change ahead of the competition.

Another assumption is that toys of the same build and SKU have consistent operating parameters. For example, it is assumed that the torque required to turn the knobs on the Etch-A-Sketch the team purchased for testing will be the close to the torque needed to turn the knobs on the Etch-A-Sketch used at competition.

The limitations imposed by the competition are as follows. The robot shall be completely autonomous, requiring no human input other than placement in the starting position. The final design must fit within a 1’ by 1’ by 1’ box before starting and after finishing the course. At no point during the competition can the toys be “hidden” from the audience. No flammable liquids, high pressures, or otherwise dangerous items must be part of the design. Finally, the entire course must be completed in less than 5 minutes. Self-imposed limitations include using a total of two subsystems (not more) in order to interface with four different games, as well as relying exclusively on microcontrollers rather than more complex devices such as the Raspberry Pi.

## 1.6 Expected End Product and Other Deliverables

The end product is a fully autonomous robot that is capable of successfully completing the 2015 SoutheastCon Hardware competition challenges within the 5 minute time limit. This robot will preliminarily include the following:

- Custom made aluminum chassis
- Custom made “arm” for each subsystem
- Arduino Mega 2560 Microcontroller
- 2 smaller microcontrollers
- 4 brushless DC motors with encoders used for propulsion
- Motor encoder decoder(s)
- 4 mecanum wheels
- 9 infrared sensors
- 2 high torque brushless DC motors for manipulating the Etch-a-Sketch
- High torque servo motor to twist Rubik’s Cube and play Simon Says
- 2 servo motors, one to raise and lower each “arm”
- Various motor drivers of appropriate specifications, one for each motor
- 2 lithium polymer batteries

The only deliverable product for this project is the complete autonomous robot. The robot will need to be finished before the local competition in March. In addition to this, there are six milestone reports and presentations according to the requirements of the Senior Design course. The deliverables are as follows:

- Milestone 1:Needs Analysis and Requirements Specification **(9/18/2014)**
- Milestone 2: Project Proposal and Statement of Work **(10/16/2014)**
- Milestone 3: System Level Design Review **(11/13/2014)**
- Milestone 4 **(TBA)**
- Milestone 5 **(TBA)**
- Milestone 6 **(TBA)**
- Competition-Ready Robot **(4/9/2015)**

# 2 Concept Generation & Selection

In the following sections, the concept generation and design process for the robot are outlined. Initial scrapped ideas are covered briefly, and reasons for selecting the current design are discussed.

## 2.1 Chassis Design

The chassis of a robot is where every other component of the robot is mounted , hence, design of a chassis is very important. The team considered many different alternative designs, before ultimately deciding on the one they did.

One of the first options considered by the team was a U-shaped chassis, with the toy being manipulated at that moment being placed in the center of the U. The manipulator would then hang over the hole in the middle, having easy access to each toy. Various sensors needed to properly play each game would hang off the side of the chassis pointing inward.  The main benefits of this design is its ability to move the manipulators away from the outside of the robot and into the robot's center. By having the toy inside the robot, it also would be easy for the robot to hold each toy down.

This idea raised some questions and had a few potential issues. Having a hole in the middle of the robot complicated the placement of the sensors needed for line following. Missing the middle of one edge of the chassis could weaken the robot's structural integrity. The main issue with this idea came down to size. Once the team had decided to use mecanum wheels and had selected the motors and encoders for these wheels, the team realized that the wheels and motors took up too much room along the edge of the robot. Placing the wheels, motors, and encoders as far away from each other as possible within the 1 ft maximum side length, there would only be a few inches gap between the end of the motors. This was simply not enough room for the U. The team considered various modifications, including putting the U on the other side of the robot so the robot would strafe over the toy, as well as using bevel gears to orient the length of the motors in a different direction. Ultimately, these ideas were considered too complex, especially when there were other, much simpler, alternatives.

One alternative, which also allowed the toys to be manipulated from the middle of the robot involved the robot driving over the toy and having a hole in the chassis that was square for the manipulation to happen in. The robot would drive over the toys, centering them under the hole. This suffered from some of the same sizing issues the U-shape design suffered from, but also provided many of the same benefits. Also, because the chassis would surround the toy, it was decided that the toy may not have been considered visible enough to have been legal under the rules.

The final alternative the team considered was a simple square shape. The toys would be off to the side of the robot and the manipulators would move to align themselves with the toys. To make sure everything fit, the chassis would be built in layers, with different layers holding the electronics, the motors, and the battery. The manipulators could be placed up top.  One of the benefits of this design is that it is mechanically simple, basically all that is being built is a cube. Another benefit is that it takes full advantage of all the space provided in the rules. As mentioned previously, one of the problems with the previous ideas was having enough space along one of the three axes. By building up, this design takes advantage of the foot given up and down, as well as the foot given along each side.

| | **Support of Manipulation (0.4)** | **Structural Integrity (0.5)** | **Ease of Manufacture (0.1)** | **Total** |
|---|---|---|---|---|
| U Shape Design | 4 | 2 | 2 | 2.8 |
| Hole in Center | 4 | 3 | 3 | 3.4 |
| Simple Square | 3 | 5 | 5 | **4.2** |

**Table 1:** Decision Matrix for chassis design, with ideas ranked from 1 to 5 with 1 being lowest.

## 2.2 Line Following/Movement Design

With the physical movement of the robot, the team needed to decide how the robot was going to steer. With respect to steering, the choice quickly came down to whether to go with differential steering or omnidirectional steering. While the potential to be more versatile with sideways motion without turning, it is more difficult to have fine control because of the reliance of consistent friction on each wheel. On the other hand, differential steering does not have the same versatility but is a much simpler choice. Differential steering would require fewer motors than omnidirectional drive, and would also be cheaper to implement and easier to program. The kinematics of a differentially steered robot are straightforward and well understood. The team decided on omnidirectional drive using mecanum wheels, thinking that the versatility of sideways motion would drastically simplify the line following code that would be needed.

| | **Control (0.4)** | **Flexibility (0.6)** | **Total** |
|---|---|---|---|
| **Differential Steering** | 4 | 1 | 2.2 |
| **Omnidirectional Steering** | 3 | 5 | **4.2** |

**Table 2:** Decision Matrix for the choice of steering method.

With the movement of the robot decided on, the team had to think of a way to detect the white line on the track and use that data to move the wheels. The traditional way to design line following for small scale robotics is to use infrared reflectance sensors (IRR). Consideration was given to alternate methods of line following such as using a camera with image processing, but image processing can be highly intensive processing wise and would require a more powerful microcontroller. Thus the simplest and most common method was used. The model of IRR sensor that was decided on was the QRE1113 on a SparkFun breakout board. Nine of these sensors were placed in a 3x3 grid pattern under the robot. The 3x3 grid pattern was decided on over other setup patterns because of its increased resolution and corner detection over a linear or a cross type pattern.

The sensor grid will be connected to an Arduino Mini to preprocess the sensor information then that data will be sent to the Mega to control the motors.

## 2.3 Manipulator Arms

The robot needs a way to interface with the different challenges. In order to not have to make four different systems, synergies between the different challenges were considered. The Simon Says game requires circular motion in the horizontal plane in order to hit the different buttons on the toy. Similarly, the Rubik's Cube requires one row to be twisted 180 degrees. The same motion can be used for both challenges, and they will share a manipulator arm. For the Etch-a-Sketch, torsional friction is required to actuate the knobs on the Etch-a-Sketch. This can be accomplished with a sticky material which can then be re-used to pick up the playing card. Therefore, these two challenges share an arm.

### 2.3.1 Simon Says/Rubik's Cube Arm

#### <u>2.3.1.1 Physical Interface</u>

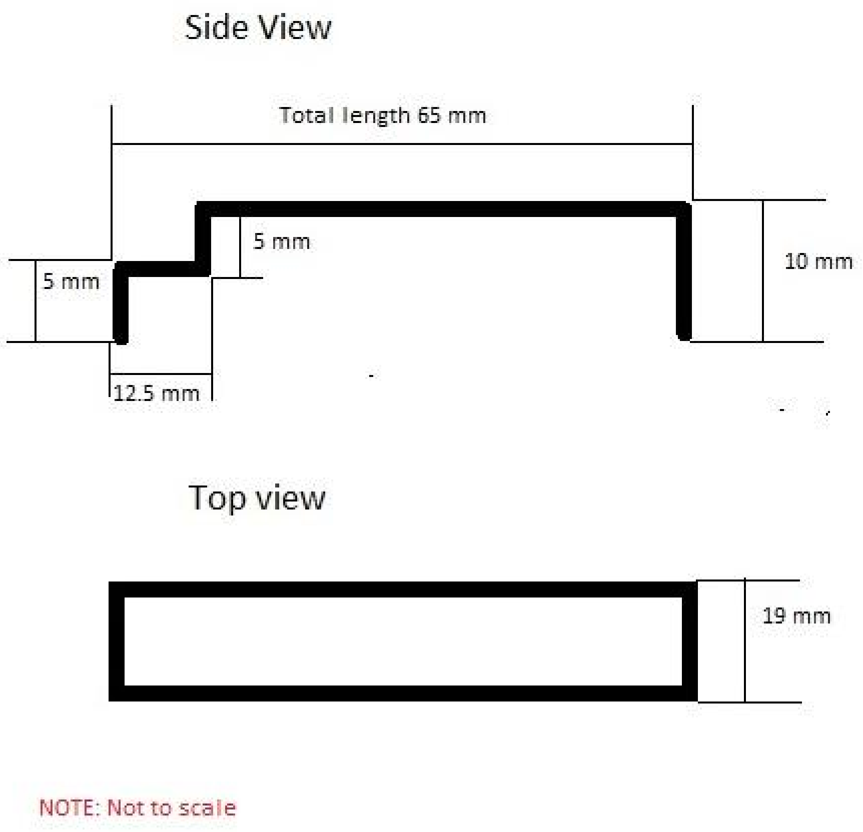


**Figure 1**: Custom Simon Says/Rubik's Cube interface

The first idea for interfacing with the Simon Says that was considered was a system of four pistons arranged as corners of the square. The pistons would move vertically in order to hit the appropriate button on the Simon Says game. This was rejected because of it's complexity and cost. Another option was a four-bar-linkage with a single piston. The movement of the linkage would bring the piston to the appropriate button, and the piston would push it. This was rejected, again because of complexity, but also because of limited synergies with the Rubik's Cube challenge.

For the Rubik's Cube, a simple claw that could be spun in the horizontal plane was considered. This would grab the top row of the Rubik's Cube and twist it as appropriate. This was rejected because it was difficult to find a reliable claw that would open wide enough as well as fit with the overall design, in addition to the fact, that combining this with the Simon Says game would be difficult.

Keeping it simple, a device for interfacing with both toys was designed as shown in Figure 1. This will be attached to a full rotation servo so it can spin in the horizontal plane. The servo will be at the end of an arm that can be raised or lowered onto either toy. The device relies on certain programming quirks in the Simon Says game when two buttons are pressed at the same time. The game seems to give preference to the middle "start" button before the game is started, and ignore it while the game is running. The device will rotate to the correct position and be lowered onto the Simon Says game by the arm in order to hit the start button and one colored button each time. There are also right angles on either side of the device that will catch the edges of the top row of a Rubik's Cube. The servo will then spin the row.

Any design for this manipulator arm requires a method for holding the Rubik's Cube in place. This concept is still under discussion, but the mechanism can be as simple as having two bars protrude from the chassis, one on either side of the cube, in order to prevent the bottom two rows from rotating.

#### 2.3.1.2 Simon Says Sequence Detection

In order to correctly play the Simon Says game, the robot needs a way to correctly identify the sequence of buttons it needs to push. The individual buttons (each of which has a different color) will light up in sequence, and a different sound will be played for each one.  Three ideas for detection were discussed: color detection, light detection, and sound (pitch) detection.

The first idea to be rejected was a color sensor. While these exist in complete packages, they are normally made for detecting a static color, not whether or not a certain color has lit up. Therefore, it likely does not have the resolution that is needed to accurately detect a Simon Says Sequence.

The next option is using one light sensor (photodiode or photoresistor) for each button. The sensor would be close to the button and monitor its light intensity. When the button lights up, a change in intensity would be registered, and the sequence could be determined. This approach was rejected because of its sensitivity to the light in the environment. As the rules forbid covering the games, ambient light could not be removed and could significantly lower the reliability of the sensors.

The method that was eventually selected was sound detection. A microphone is held close to the game, and frequency information from the different sounds is extracted from the individual button sounds. Sufficient quality microphones are cheap and the required circuit is easy to make (Figure 2). It is also more straightforward to design noise cancellation than light cancellation in the intended environment.

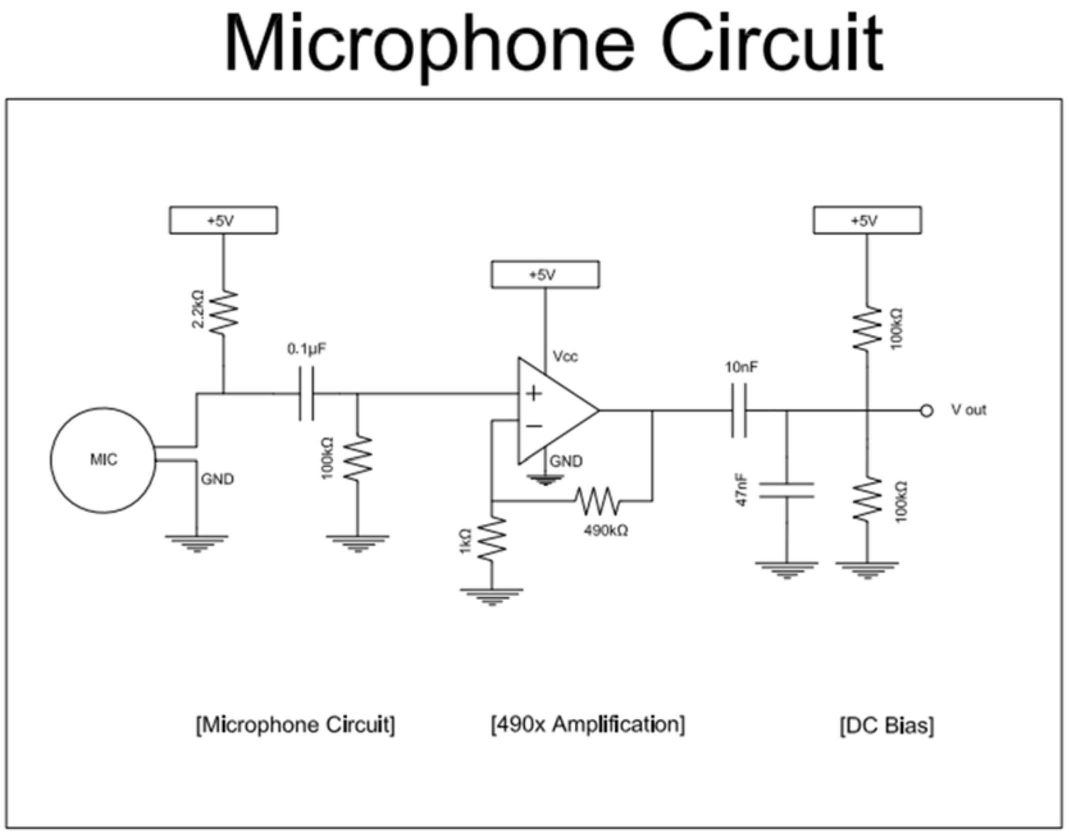


**Figure 2**: Microphone Circuit

With a working microphone, two methods for extracting frequency information were considered: a Fast Fourier Transform (FFT), and a simple frequency counter.

Early tests were done using the Arduino and a Microphone. The different button sounds were sampled and transmitted to a laptop. The FFT was computed using Matlab, and frequency peaks were reliably identified. This method, however, proved too difficult to implement on the Microcontroller, and even with the in-place computations performed by the FFT, the Arduino was having memory-related space issues.

Thus, a frequency counter was implemented. This relies on sampling at 38.5 kHz and looking at the distance between the places where a signal crosses the “0” axis provided by the DC bias in Figure 2. Taking existing Arduino code (this has been done before) and making minimal adjustments, this worked “out of the box.” The current iteration of sequence detection will store a series of frequencies in an array and infer a sequence of buttons from this information.

### 2.3.2 Etch-a-Sketch/Playing Card Arm

For the Etch-A-Sketch arm, a few different ideas were considered as to how to turn the knobs on the Etch-A-Sketch. One of the ideas involved using frictional contact on the side of the knob to rotate it. This idea was rejected because the team was concerned that it would would be too hard to apply the right amount of force to the sides of the knobs to get the proper friction force. It was also a concern that aligning a manipulator between both knobs would be very difficult. The benefit of this design is the the potential to change the size of the “rubbing knob” to take advantage of the size differential between it and the Etch-A-Sketch knob and have a gear ratio like phenomenon. This would allow the team to potentially use less powerful motors for each knob and still get the required torque.

The team also considered grabbing onto the knobs, by having a mechanism that would surround the knobs and twist. This was rejected again because of concerns with the difficulty in getting everything properly in line. It was also considered difficult to grasp the knobs.This method seemed costly compared to the budget received.

The final idea the team considered was to having the motors mounted above the knobs with a surface with lots of friction connected to each motor and this surface rubbing against the top of the knobs. The torsional friction force between the friction plate and the knob would cause the knob to turn. There are many concerns with this design. For it to work, the axis of rotation for the knob, the friction plate, and the motor need to be the same. While it is not difficult to make the axis of rotation for the motor and the plate that is mounted to it the same, it is a special alignment challenge to make the axis of rotation for the friction plate the same. If the axis of rotation is different between the plate and the knob is different, then the knob will have to slip with respect to the plate. This is clearly not ideal if the design relies on the knob and the plate turning identically. The benefit of this design is its simplicity, there is no need to worry about any hand-like gripping mechanism, etc. Ultimately, while alignment is a concern, the idea is very mechanically simple, which will make it easier to implement and fix as needed. Ultimately, this mechanical simplicity was why the team decided to stick with this design.

## 2.4 Control/Microprocessors

For robot control at this level, there are two traditional options. The first is a microcontroller setup such as Arduino (based on the ATMega series) or PICAXE. The second is going with a fully featured ARM-based computer platform such as the Raspberry Pi or the Beaglebone (among others). The second option comes with more power and enables more advanced features such as image processing (difficult to do on a cheaper microcontroller). However, it also comes at increased cost and it is more difficult to use such platforms for lower level control.

The Arduino platform was suggested early on, and was adopted for several reasons. First, all programming happens in C (with inline assembly), with which all group members are already familiar. This decreased amount of time it took before prototyping could occur. Second, Arduino platforms are cheap, and most group members already had them available allowing for the creation of prototypes without having to purchase extra microcontrollers. Finally, the Arduino series have integrated communication protocols such as RS232 and I2C that makes communication between several chips trivial.

The Arduino Mega 2560 was chosen for the main controller unit of the robot. This will be making all the “decisions,” driving the motors, and interfacing with auxiliary microcontrollers. The Mega 2560 has ample IO ports, as well as more external interrupt vectors than any other AVR-based model in the series. In fact, it has enough digital IO ports to interface with every single component of the design. However, having the main microcontroller interface with more than 10 sensors and two separate arms at the same time as the propulsion was deemed less than desirable. Therefore, the design was made modular by including a smaller microprocessor in three subsystems: the line sensors and either arm. These auxiliary processors were chosen to be Arduino Mini Pro boards because they are cheap ($6 each), and have as much functionality as the $30 Arduino Uno.

The line sensing subsystem, including up to 9 infrared sensors, will be controlled by one microprocessor. This processor will continuously poll every single sensor, and then feed aggregate information to the Arduino Mega. This reduces the workload of the main processor, and allows for line following information to be provided “on demand.”

Each arm will also have its own Arduino Mini Pro. These will wait until given a signal from the Mega, proceed to complete a challenge, then go inactive again and wait for the second challenge. After two challenges have been completed by one processor, it will go inactive until the end of the course.

## 2.5 Communication

The competition rules disallow any outside communication with the robot, and therefore wireless communication between subsystems is rejected by default. As per the specifications of the microcontrollers, two communication protocols were discussed: I2C and RS232 (serial). Both are supported “out of the box” by the Arduino platform.

The advantages of I2C is that it will lower the complexity of wiring the microcontrollers together. This protocol works on two buses - a data bus and a clock bus. Information is sent across the data bus and is addressed to the proper recipient. Therefore, the wiring is limited to two sets of wires for the entire system of four microcontrollers. The drawbacks of I2C are that it is slightly more difficult to implement in software, and that if communication breaks down it will affect the entire system.

The system in its current iteration is based on serial communication. This requires a separate connection (transmit and receive lines) from the Mega to each subsystem. It was implemented for early prototypes because the Arduino environment supports serial very well. A drawback of serial, as mentioned, is that it requires as many different sets of wires as there are subsystems, but this could also improve the reliability of the system (a breakdown in one arm’s communication leaves the second unaffected). As it is already in place, serial will likely be the communication protocol of the final design unless a compelling reason for switching to I2C is presented.

## 2.6 Power Supply

With the complexity of this design and all the moving parts it is important that there be enough power to comfortably be supplied to the electronics. With the various motors, drivers, and microcontrollers on the chassis there must be a way for the team to sufficiently supply enough power to reliably run the course every time for the competition.

Originally the plan was to independently power each of the manipulators as well as use one form of power for the motors that control the wheels for the robot to move. While in search for a rechargeable power supply it was noticed that if a Lithium Polymer (LiPo) battery is used, one battery can actually supply enough power to both manipulators. This would be more efficient because it will result in fewer components. The reason why LiPo was used instead of a Nickel–metal hydride (NiMH) is because LiPo batteries are much more efficient on weight. The less weight that is put on the robot, the more efficient it will be.

It was then decided that a total of two LiPo batteries should be used to supply power to the robot and that the devices will draw power as needed. One LiPo battery will be used for the propulsion of the robot as before; this includes all motors, sensors and electronic devices that contribute to the movement of the robot. There will then be another LiPo battery being used for everything that is on the top of the chassis; namely the manipulator electronic components.

In order for this to be done efficiently it is important that the total amount of power consumed by the robot is calculated before the batteries are purchased. This number should then multiplied by the maximum time that the robot has to complete the track which is five minutes. Since the plan is to use two batteries, the propulsion specifications, and the manipulators specifications must be treated as separate power systems. This will provide the minimum specifications that the individual batteries must satisfy to power their subsystem and ultimately the robot for the entire course.

## 2.7 Motor Selection

Motors and other actuators are necessary for the robot to interact with the outside environment. When selecting motors, it is important to consider the required torques and angular velocities.

### 2.7.1 Propulsion

The first motors selected had to do with the robot's propulsion. An essential design decision was to implement PI (Proportional Integral) control, which means that the motors all required encoders to have the feedback data for the PI. Our Mechanical Engineer on staff (James Pace), did research on some available motor choices and chose an appropriate motor with the assumption that the weight of the robot would be under 10 lbs and would have a top speed of 0.5 ft/s. The encoders were selected due to their compatibility with the motors and the head programmer's (Ryan-David Reyes') past experience with them.

### 2.7.2 Etch-A-Sketch/Card Pickup Arm

For the Etch-A-Sketch Arm, once the team chose the general approach of using friction on the top of the knobs to cause motion, the team needed to choose specific motors that could provide the necessary torque. To do this the team estimated the torque necessary to turn the knobs on the Etch-A-Sketch to be 10 oz-in. The team also decided that to complete the task in a timely manner, but still have good control of the knobs, the robot should take only three seconds to move the knob completely along the long end of the etch-a-sketch screen. It took three full rotations of the knob to move this long distance. Hence, the knob needed to turn 1 rotation a second, or at 60 rpm. A factor of safety of two was applied to handle inconsistencies in the manufacturing of various etch-a-sketches and any errors in the motor specifications. The team also wanted a motor that was light, so it could be mounted directly where the arms would go, and wouldn't have to have the motion be transported to the location of the knobs. Based on this and price, the team went with a geared micromotor from pololu. Once the motor was chosen, motor drivers were chosen which were both compatible with the requirements of the motors and which were relatively inexpensive.

The arm also needed a motor to lift and lower the end effector into location. Originally the team wanted to go with a motor and worm gear drive set. The worm gear would help provide extra torque and allow the team to go with a less expensive motor, but worm gear are also not back drive-able, so it would not take any energy to lock the arm in its default "up" position. Worm gear kits are available online, but after experimentation, it was determined that the kits are too flimsy for actual, consistent use. Custom gear sets are not economically viable for a single prototype. Ultimately, the team is planning on using a servo motor that was already available. The servo will give the team the necessary torque to lift the arm, and will also give accurate position control.

### 2.7.3 Simon Says/ Rubik's Cube Arm

For motor selection for the Simon Says/ Rubik's Cube arm, the most important issue will be getting proper torque to turn the Rubik's Cube, regardless of how worn the arm is. From experience, the team knows that the required torque to turn a row of the cube decreases the more times the cube is rotated. There is also a wide variance in the required torque between various cubes. Because of this, the team will need to be careful to apply proper factors of safeties when choosing the motor to turn the end effector.

# 3 Proposed Design

## 3.1 Overview

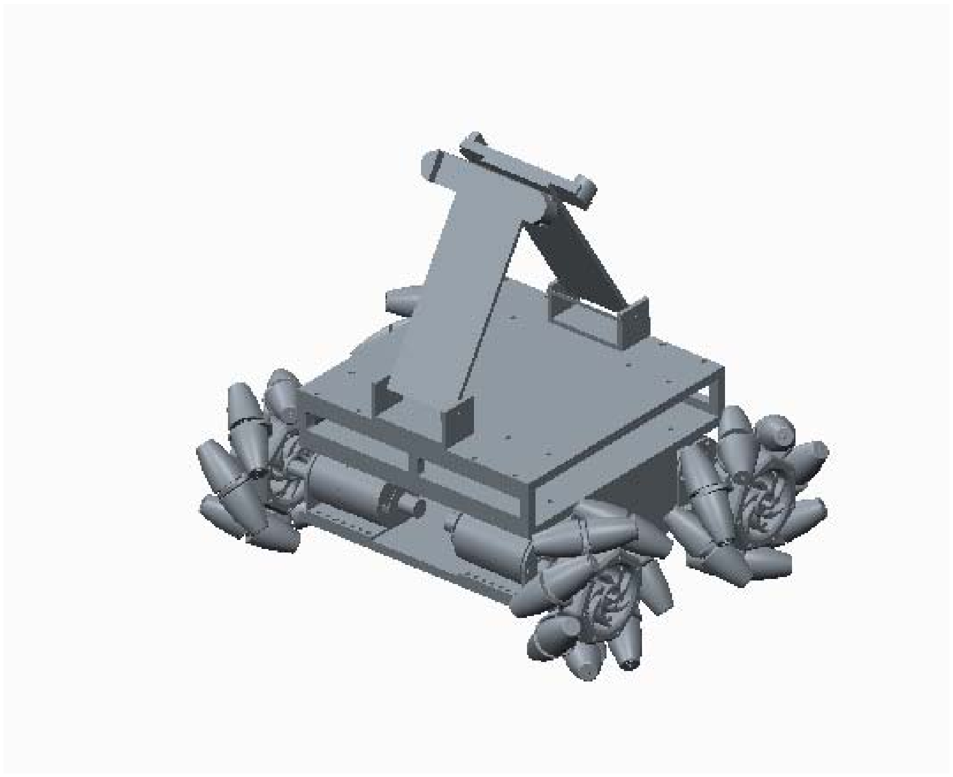


**Figure 3**: Robot Chassis Concept

The above model shows an approximate layout of our robot, including two arms on the top level, a square chassis, and multiple other levels with room for electronics and batteries. The current rough layout of the chassis fits within the size constraints of the competition, but will be adjusted once final decisions are made regarding the arms and end effectors. Specifically, the location of mounting holes for the sensors and motors for the end effector will be adjusted once those components are chosen and holes will be placed specifically to place those components.

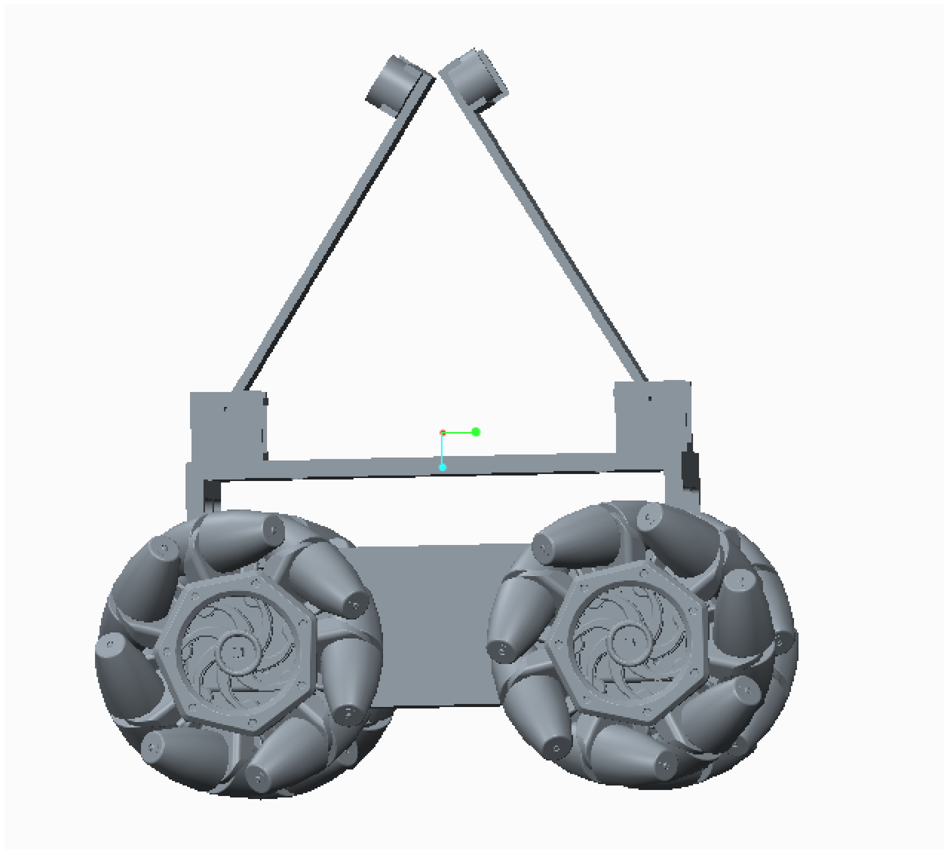


**Figure 4**: Robot Chassis Concept

## Top Level Block Diagram

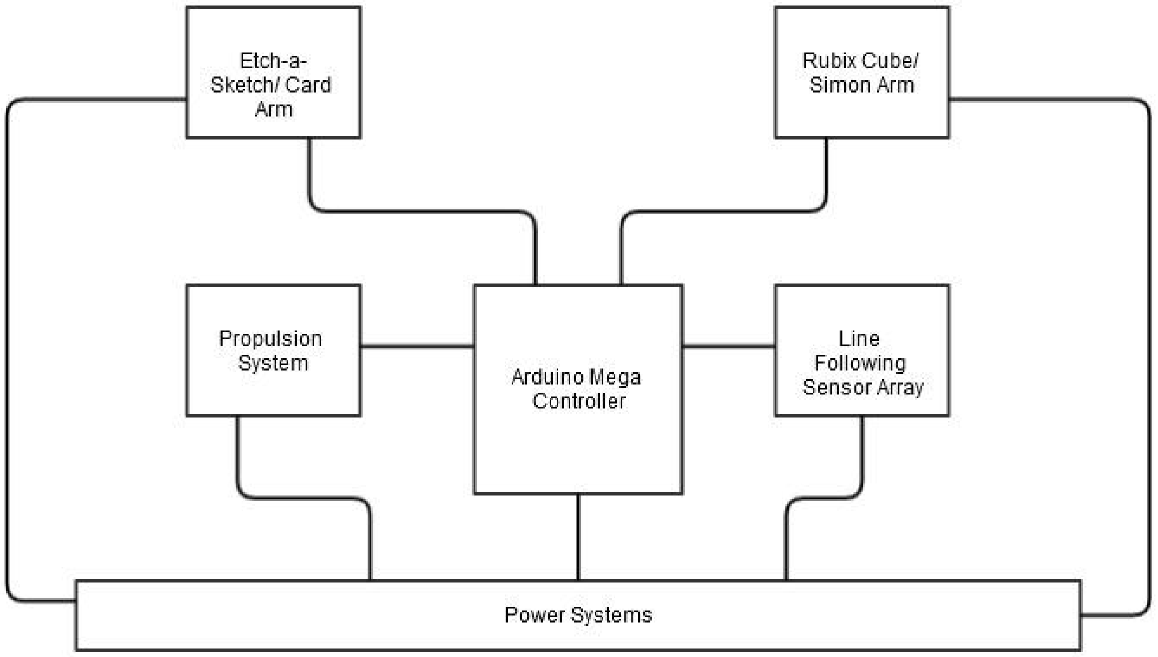


**Figure 5**: Top Level Block Diagram

For the overall design, the team is aiming to create a fully autonomous robot that will navigate the track, complete all of the challenges and cross the finish line in a minimum amount of time. To do this, there are 6 major systems of the design, shown in Figure 5.

## 3.2 Line Following Sensor Array

### Line Following Sensor Array Block Diagram

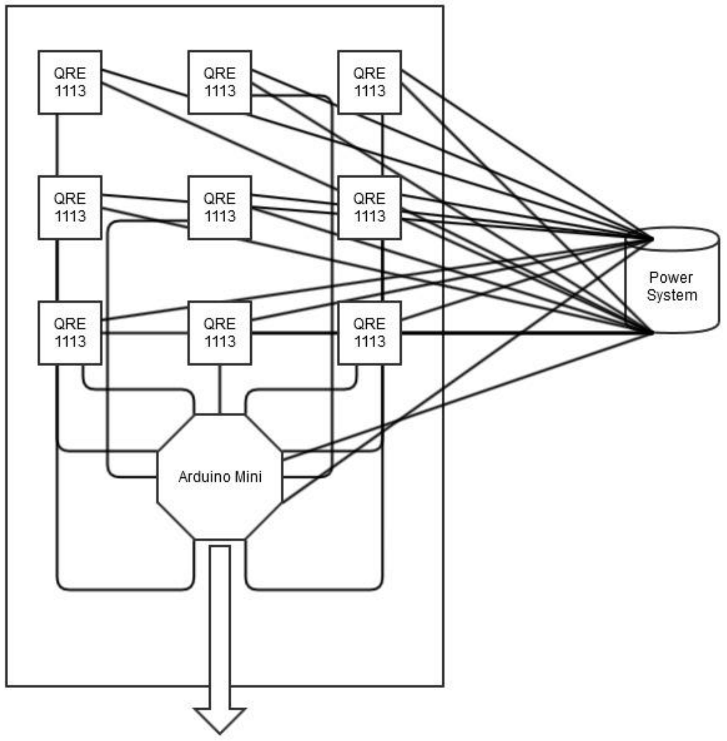


**Figure 5:** Line Following Block Diagram

This is the sensor array located on the underside of the robot that will initiate the robot in the starting box and navigate the track to the different stations to complete the challenges. This system consists of 9 QRE1113 Infrared Reflection Sensors that are arranged in a 3x3 grid pattern to discern when the robot is drifting off course and when a corner or branch section is reached. All of the data from the 9 sensors are collected into an Arduino Mini that will process the sensor information and send the appropriate movement signal to the Main Control Block (Arduino Mega) which will then activate the Propulsion System.

The sensors work by shining light onto the track and measuring the reflected light. This value ranges from 0 to 3000 with “whiter” surfaces giving a smaller value. Through testing, the team determined that any value over 1000 was black and that any value under 1000 was white. Using these sensors, different patterns sensed by the sensors will indicate the position of the robot and thus the Main Controller will adjust accordingly.

## Sensor Grid Patterns

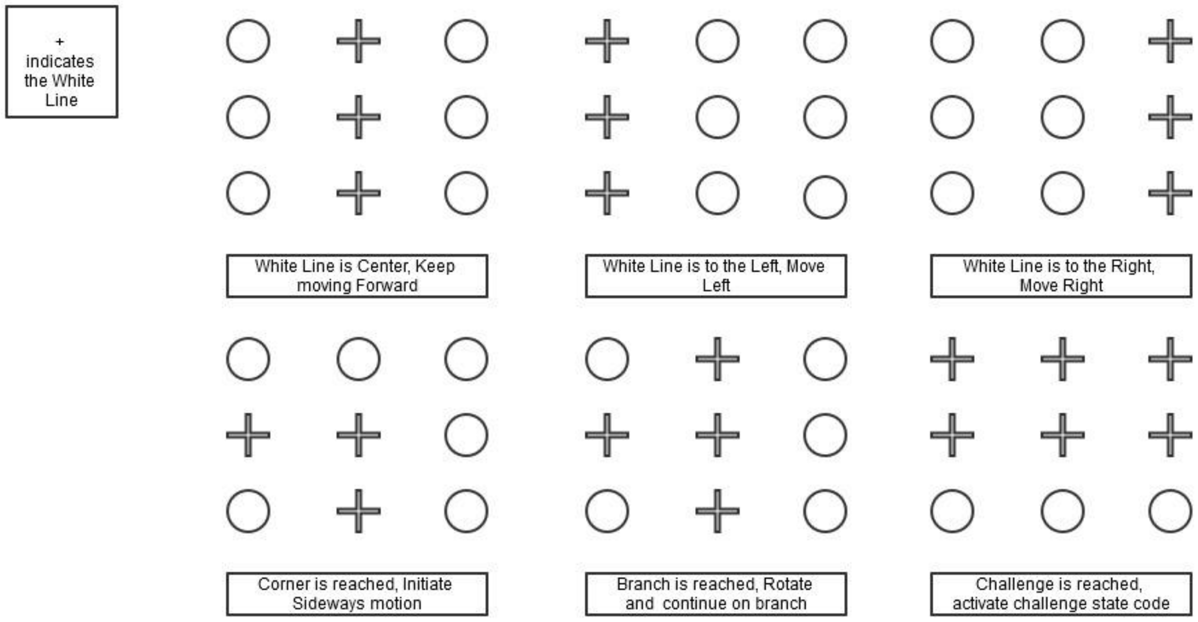

+ indicates the White Line
White Line is Center, Keep moving Forward
White Line is to the Left, Move Left
White Line is to the Right, Move Right
Corner is reached, Initiate Sideways motion
Branch is reached, Rotate and continue on branch
Challenge is reached, activate challenge state code

**Figure 6:** Sensor Grid Patterns

These Sensor Grid Patterns are handled by the Arduino Mini that is part of the Sensor Array System. As part of the preprocessing before giving the movement signal to the Main Controller, the Mini is programmed with these patterns and tells the Main Controller the appropriate movement that the Propulsion System needs to make.

## 3.3 Propulsion System

### Propulsion System Block Diagram

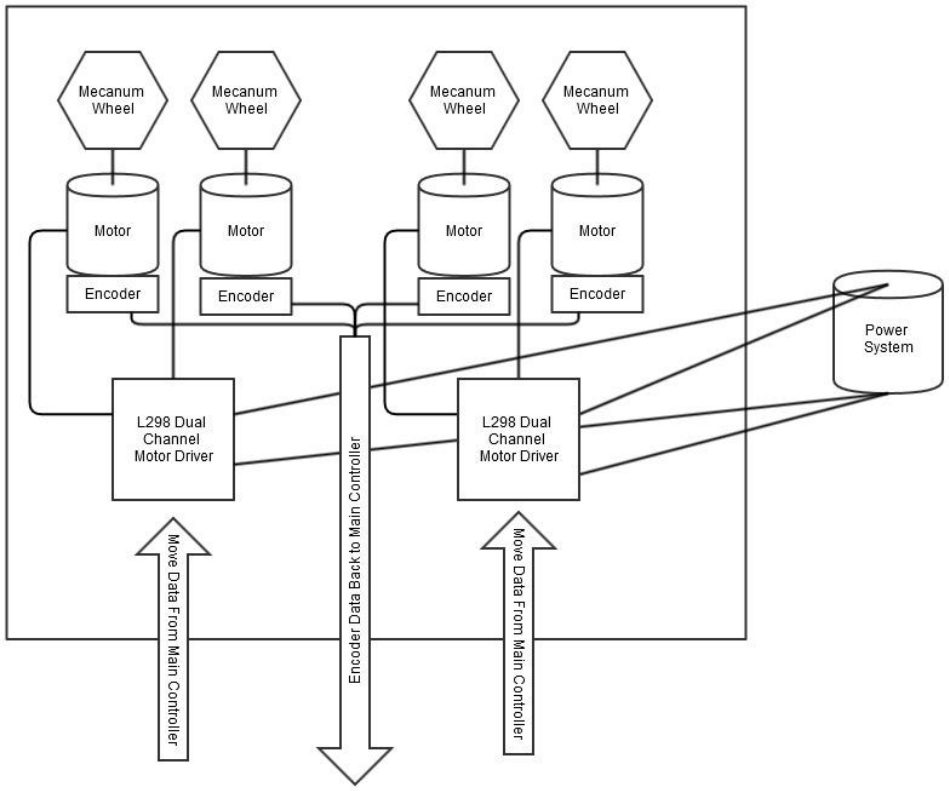

Mecanum Wheel
Mecanum Wheel
Mecanum Wheel
Mecanum Wheel
Motor
Motor
Motor
Motor
Encoder
Encoder
Encoder
Encoder
Power System
L298 Dual Channel Motor Driver
L298 Dual Channel Motor Driver
Encoder Data Back to Main Controller
Move Data From Main Controller
Move Data From Main Controller

**Figure 7:** Propulsion block diagram

The Propulsion System consists of the 4 4" diameter mecanum wheels each connected separately to a DC motor each with a built in encoder. The encoder data will be used for the PID control that will smooth out the motion of the robot as it navigates the track and will enable more precise control. These motors will be connected to 2 L298 Dual Channel Motor Drivers which will go to the Main Controller. The Main Controller will control each individual motor based on the sensor data from the Sensor Array System, and the encoders will send data back to the Main Controller for the PID control.

The mecanum wheels the robot uses have the unique property of executing sideways motion without turning, which will allow for simpler  Line Following Code to be implemented in the Main Controller. In previous competitions, other teams have tried to use these type of wheels with mixed success, and concerns have been raised about their viability. But with the promising tests that have been done, the team is confident that the implementation of these wheels will go smoothly and no backup options will be needed. We are confident that this propulsion design will be able to navigate the track.

The only problem with this design, is the uncertainty of very small precise movements that might be needed for robot positioning of the challenges. More testing is needed in this respect, and if it is required; there are mechanical solutions to the positioning problem that can decrease the need for small precise movements. Some of these solutions are discussed in the descriptions of the other systems.

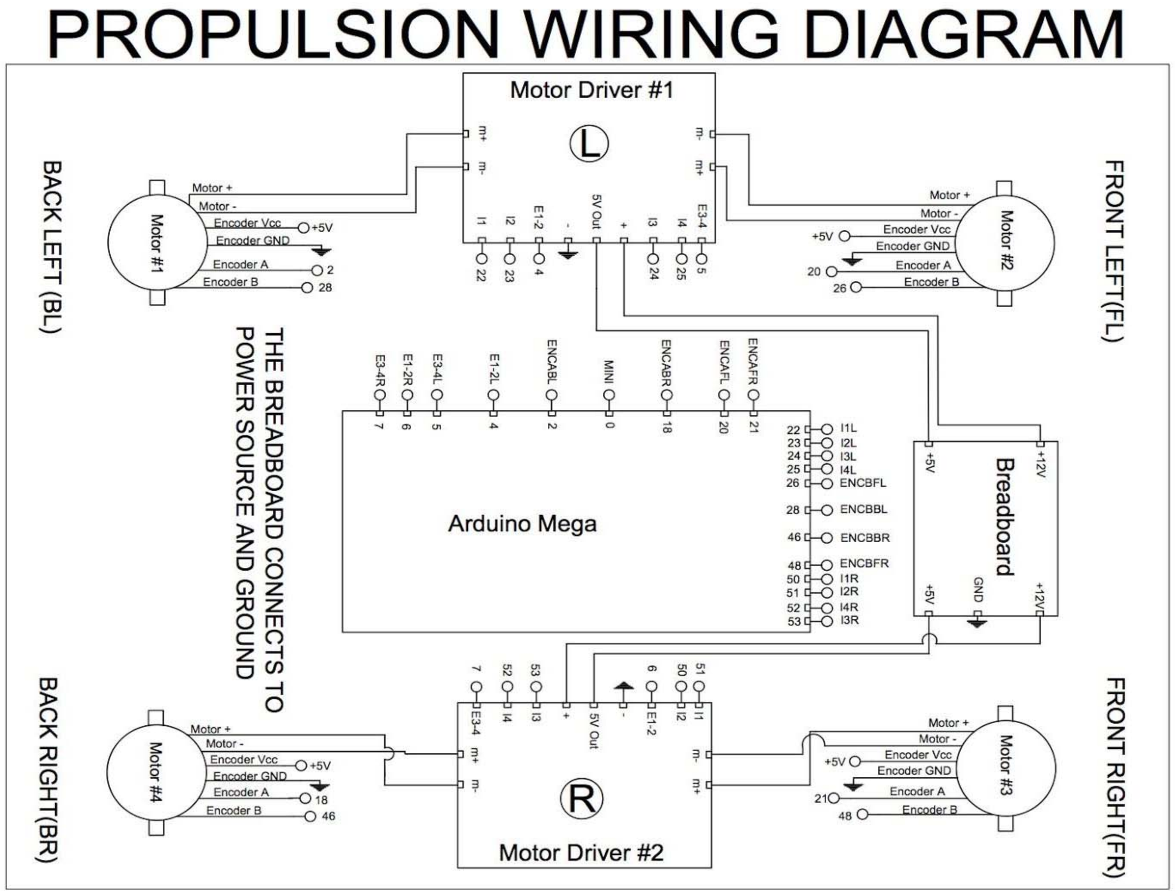


**Figure 8:** Propulsion wiring diagram

## 3.4 Etch-a-Sketch/Card Arm

### Etch-a-Sketch/Card Arm Block Diagram

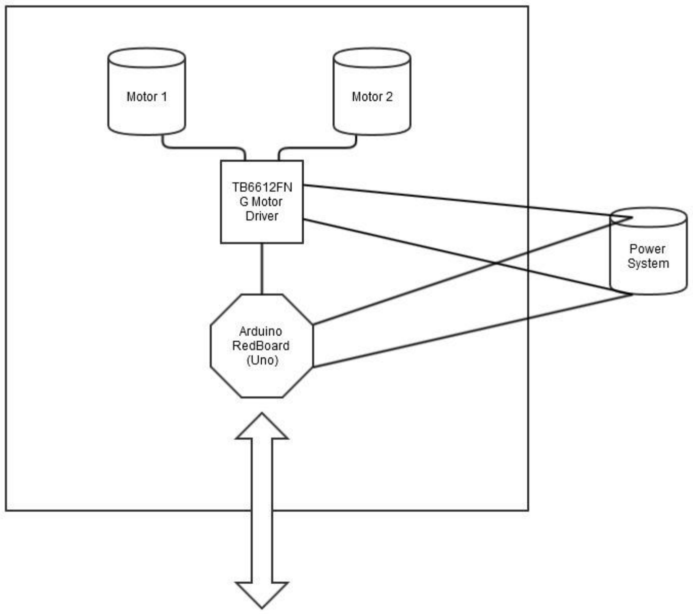


**Figure 9:** Etch-a-Sketch/Card arm block diagram

### Theoretical Etch-a-Sketch Arm Design

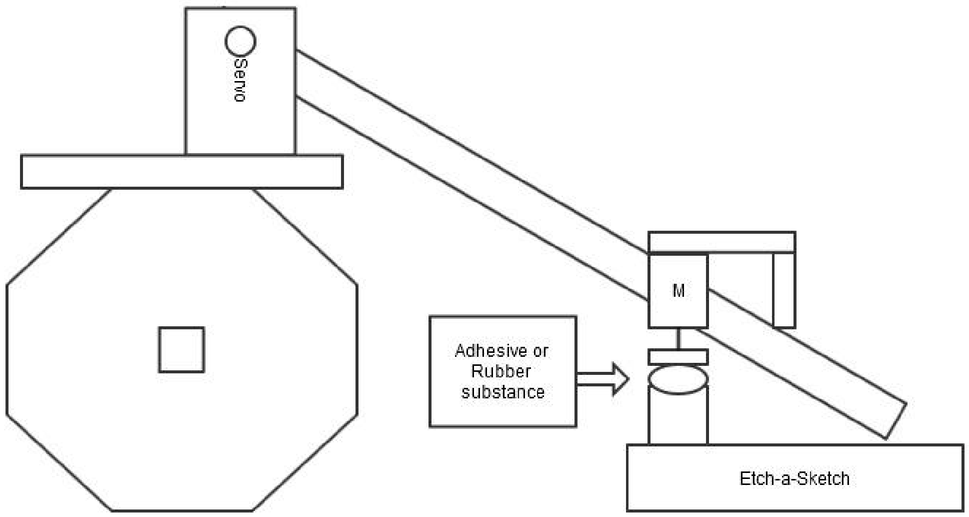


**Figure 10:** Etch-a-Sketch challenge design (Side View)

This system is designed to complete the Etch-a-Sketch challenge and the card challenge. With both challenges having some synergies in terms of the needed functions to perform both tasks, it was decided that one manipulator could be designed to perform both challenges. The basic design of the manipulator consists of a flat cross shaped piece of metal that moves up and down using a servo. On the cross section of the arm, there are two geared micromotors mounted that have hubs attached at the end of their shafts, on these hubs an adhesive is placed. The motors are to be lowered onto the Etch-a-Sketch and placed on top of the knobs, one motor for each knob. As the motors spin, the frictional force of the adhesive will force the knob to spin as well. The geared micromotors will be connected to a TB6612FNG Dual Channel Motor Driver which connects to the Arduino RedBoard (Uno) which will control the spinning of the motors. Since the pattern of the Etch-a-Sketch drawing is set (drawing “IEEE”) this pattern will be hard-coded into the Arduino and set to execute once the manipulator is in position.

Getting into position is the main design problem that we have faced and a decision has not been made on a final implementation. Different ideas have been the use of ultrasonic sensors or a camera to determine distance but ultrasonic sensors can be unreliable, a camera requires image processing and the precision of the Mecanum wheels to place the robot in the precise location needed is unknown. Frames have been discussed to mechanically force the Etch-a-Sketch into position but since the arm will come in at an angle to the Etch-a-Sketch, clearance is difficult to achieve with the bottom side of the toy.

Performing the Card challenge is much simpler than the Etch-a-Sketch challenge and will involve lowering the arm in a position to where one of the motors with adhesive will be on top of the deck of cards. Then simply raise the arm and the card will be stuck. As with the Etch-a-Sketch challenge, the primary problem is positioning and that is a problem that hasn't been solved yet.

## 3.5 Rubik’s Cube/Simon Arm

### Rubik’s Cube/Simon Arm Block Diagram

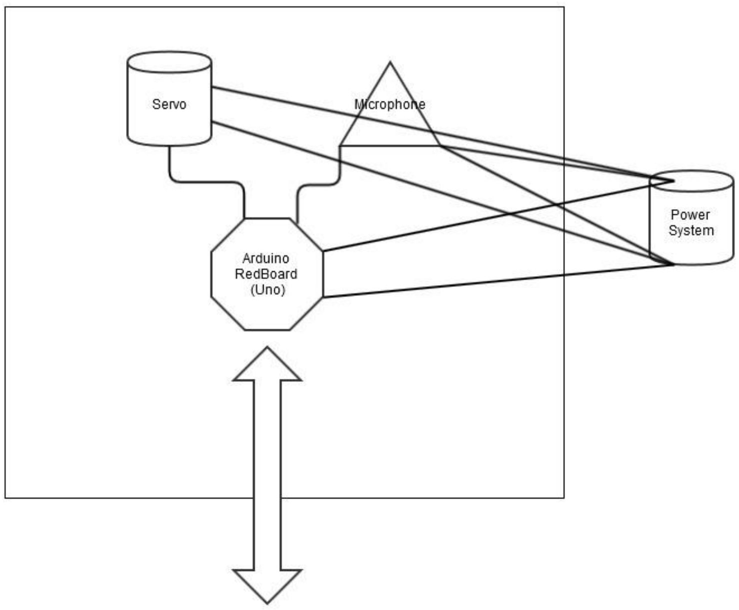


**Figure 11**: Rubik’s Cube/Simon arm block diagram

This system is designed to complete the Rubik’s Cube challenge and the Simon Says Challenge. While the challenges don’t seem to have any similarities or synergies, the team has come up with a way of completing both challenges with the same manipulator. Referring to Figure 1, the idea of this design is to have the function to click all 5 buttons on the Simon Says while also having the function to hold the top of the Rubik’s Cube and rotate it 180 degrees. The arm itself will just be a piece or either metal or plastic which will be raised or lowered by a servo. This raising and lowering motion will be used to place the manipulator in position for both games and will be used to hit the buttons of the Simon Says game. The inward notch on one side of the interface is there to hit all of the buttons of the Simon Says with the whole design still being able to hold the Rubik’s Cube. This implement will be attached to a 360 degree servo that will rotate the implement to the 4 different positions of the Simon Says. What is not shown in the figure is the central notch that will hit the center button to start the Simon game. Through testing it was found that hitting the center button after starting the game does nothing, thus the center notch was placed in the design to always hit the center button.

To sense what colors will light up with the Simon Says, sound sensors will be used to detect the frequency of the different color sounds that beep when they light up. The microphone will be connected to an Arduino microcontroller that performs simple frequency counting in order to determine the appropriate sequence of colors.

This manipulator will also be able to perform the Rubik's Cube challenge which involves the functions of holding the bottom two layers of the Rubik's Cube and rotating the top layer of the Rubik's Cube with the implement. For the function of holding the bottom two layers of the Rubik's Cube, a design choice has yet to be made. One idea was to simply have two bars extend out to hold the two layers in place but details of this type and other types of designs are still ongoing.

Regardless of how the challenges are going to be performed, robot positioning to properly use the manipulator has been heavily discussed but nothing has been set for this manipulator. The Force Sensitive Resistor discussed for the Etch-a-Sketch manipulator, would not work as well with this manipulator due to the irregular shape of the Simon Says. Other options such as using ultrasonic sensors in combination with precise movement to position are still being discussed.

## 3.6 Power System

As stated in the previous sections of the report there must be a power system analysis performed on the robot before purchasing any of the batteries to ensure that the batteries can withstand the amount of voltage and current needed to be drawn by all the components using that particular source; including microcontrollers, motors, motor drivers, servo's, and sensors. Because of this it was decided to use a power supply to deliver power to the individual subsystems before purchasing a batteries. If the battery is purchased before all the subsystems are working it is possible that there may be a need for a different design which may require more power leaving us with a battery that isn't sufficient enough to handle its required subsystem for the duration of the competition.

After all appropriate subsystems have sufficiently been completed and the robot is working correctly there will be a thorough analysis done on the power delivered to each subsystem. This can be done by taking the amount of power that each device draws and multiplying this by the 5 minutes that is the maximum time that the robot can attempt to complete a round. Though the desired goal of the team is a three minute trip time; it is important that the power system be able to withstand the entire duration just in case of complications in the course causing the robot to run over the desired time.

As of now there is an estimated number of two LiPo batteries that will be needed to operate the robot. The design that is in mind is a design where one of the power supplies is in charge of delivering power to the propulsion and all of its components, which include four DC motors with encoders and the Arduino Mega 2560. The other supply will be used to power both manipulator systems and the Line Following Sensor Array.

## 3.7 Main Controller

The Main Controller is the “Brain” of the robot, it receives and gives out signals to all of the other Systems and is the control center for all movements and functions of the robot.

The Main Controller that the robot uses is the Arduino Mega. It has 54 Digital I/O pins (of which 15 support PWM) part of that being 6 broken out interrupts which are used for the PID control. All the Systems of the design connect back to the Main Controller and it controls the operating state of the robot. The robot will have different states depending upon what section of the course it is currently in. At the start, the robot will be in the Start State which will change to the Line Following State once the robot has exited the starting box. The robot will follow the line until it reaches the first challenge box, thus going into the State for that particular challenge, look at the State Diagram for more details on the different states. The Main Controller controls the states and sends the appropriate signals to all the other systems of the design based on the appropriate state.

### Top Level State Diagram

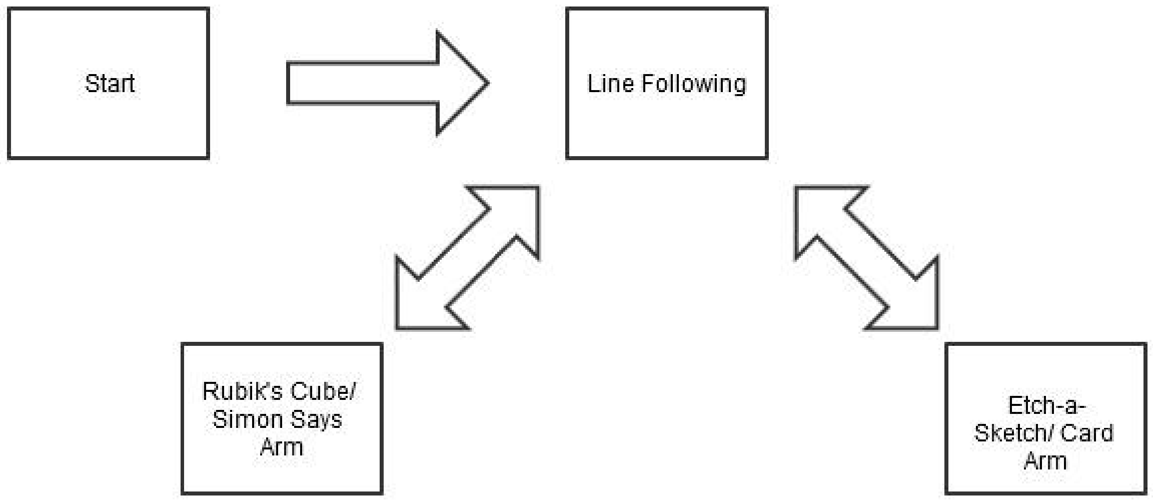


**Figure 12**: Top Level State Diagram

Other than controlling the Operating State, a big component of the Main Controller is the PID control which uses the encoder data that is received from the Propulsion System. There are 2 lines per encoder and 1 encoder per motor. There are only 6 interrupt lines, so 2 polling lines are used on 2 of the propulsion motors’ encoders. But the PID is essential in integrating precise movement with the mecanum wheels in all the different ranges of motion. If the team can achieve enough precision with the movement of the robot, it would solve the primary concern of robot positioning during the challenges. Currently, the addition of motor encoder decoders is being considered in order to ameliorate the lack of interrupt lines.

# 4 Statement of Work (SOW)

## 4.1 Task 1: Project Management

The project manager for the team is Nils Bjerén. He shall oversee the progress of this design, and will ensure that the milestones for the project are completed by their intended deadlines. He also serves as the liaison between the team and our Senior Design Project Instructor, Dr. Bruce Harvey, and the ECE professors who serve on the team's board of review: Drs. Linda Debrunner and Michael Frank. James Pace, the Mechanical Engineering major on the team shall serve as the liaison between our team and the professors from the ME Department: Drs. Helzer, Gupta and Moore.

Internally, the team will be subdivided to work on individual subsystems, to efficiently complete work in parallel. Kurt Marsman, Nils Bjerén, and Ryan Reyes are assigned to work on the line following, alignment, and detect 'start' subsystems. Kurt is focused on designing the system that determines when the robot is to begin down the track, including writing code and interfacing with the sensors. Nils will design the system and code to allow the robot to follow the line and navigate branches. Since this code all deals with the main chassis system, Ryan will be in charge of integrating these two sets of code with the propulsion subsystem. As head programmer, Ryan will also be in charge of integrating the other microcontroller systems - the Etch-A-Sketch and Simon/Rubik's Arm to work in tandem.

James Pace, Donovan Carey and Chris Lewis shall be in charge of creating the 'Etch-A-Sketch' arm. As stated before, the Etch-A-Sketch Arm will be able to complete both the Etch-A-Sketch challenge and the Card Pickup challenge. Donovan will be in charge of the electro-mechanical design of the arm, while Chris will write the code that interfaces the arm with its dedicated microcontroller. James is responsible for ensuring that the design is mechanically sound, and can physically complete the challenge.

Kurt Marsman and Julian Velasquez will be in charge of the Simon and Rubik's cube arm. Julian oversees the electro-mechanical design of the arm, determining the requirements for the servos and end effector. Kurt is responsible for writing the code that will identify and record the color sequence, and interfaces with the servos on the arm itself to perform the correct motions.

As the Mechanical Engineer on the team, James Pace is responsible for designing the main chassis of the robot, and ensuring that the mechanical design of every aspect of the robot is robust and performs its intended physical function well. He keeps CAD drawings of the robot, to ensure the design can be replicated in the event that a component breaks.

## 4.2 Task 2: Design and Implementation

### 4.2.1 Objectives

The objective is to design a robot that can compete in and win the IEEE 2015 SoutheastCon Hardware Competition. Each individual subsystem will be designed and tested, with the final result being a robotic platform capable of accomplishing the challenges.

### 4.2.2 Approach

#### 4.2.2.1 Subtask: Chassis Design

##### 4.2.2.1.1 Objectives

The objective of this subtask is to design a chassis that complies with the most recent revision of the IEEE 2015 SoutheastCon Hardware competition Rules. The rules state the robot must fit within a 1 ft by 1ft by 1ft box at the beginning and end of its run, and that the robot may not be able to split into multiple pieces at any time during the run. In addition, the chassis must be capable of locomotion, as well as being able to carry the weight of the structures and accompanying circuitry that will be used to play each of the games. More specifically, the chassis must be able to hold all the microcontrollers, motors, manipulators, circuits, batteries, and wheels while still having the capability of locomotion.

##### 4.2.2.1.2 Approach

The robot, including the wheels and manipulators, must be designed to fit within a box that is 1 ft by 1ft by 1ft large. The robot chassis will be designed in multiple layers, a lower layer holding the circuitry and propulsion motors, and an upper layer holding the manipulators used to play the games. The manipulators must be mounted solidly on the chassis, to ensure that they can deliver the forces required on to the challenges. Iterative prototypes will be made depending on how the design of the two manipulators will change, which necessitates these prototypes be made out of cheap, easy to shape materials such as plywood and cardboard. Once the final design is decided upon, the chassis will be made out of a sturdy material such as aluminum. The Mechanical Engineer on staff, James Pace, will be overseeing this development, and record the progress with a set of CAD drawings. Once a base design has been created which can house all required circuitry and mechanics, James will continue to refine and enhance the design according to the progress and needs of the two arms and the propulsion system.

##### 4.2.2.1.3 Test/Verification Plan

The robot must be able to complete the line following segment of the course, from start to finish with the complete weight of all the implementations on the robot. The robot must also be able to start within the 1 ft by 1 ft by 1 ft box, and finish within the same constraint. This must be tested to ensure that the manipulators can and will be hoisted back to their original positions, within the bounds.

#### 4.2.2.1.4 Outcomes of Task

The outcome of this task is that a fully functional chassis that can complete the course and complies with the rules will be created.

### <u>4.2.2.2 Subtask: Movement and Propulsion</u>

#### 4.2.2.2.1 Objectives

The physical objective is to have complete velocity control over each of the four individual Mecanum wheels, whether they are rotating clockwise or counter-clockwise. Without this control, the robot has no feedback on how fast or how far it has traveled. Programmatically, the goal is to provide a clean API to command the robot velocity. It should be easy to implement, with much of the intensive calculations being done in the background.

#### 4.2.2.2.2 Approach

The main propulsion design will utilize two Solarbotics Motor Drivers that encapsulate H bridge functionality, which in turn allows bi-directional control of the motor. A velocity Proportional Integral controller shall be implemented, using the encoders as positional feedback from the motors. From this data, the current velocity can be obtained through differentiation, and serve as the input to the PI controller algorithm. The controller will output the desired Pulse-Width Modulation signal to the motor drivers to attain the correct commanded velocity in the motors. The PI algorithm also determines which direction the motor needs to turn, which is set via two output pins in addition to the PWM pin. The PI routine will be placed in a timer interrupt, set to occur at 200Hz. This ensures that motors will be under constant and regular control. The encoder signals are hooked up to external interrupts to the microcontroller, so that the microcontroller can monitor the encoder status with minimal latency. This allows accurate readings of the motor velocity. The team has decided upon using Mecanum wheels for the robot, as they provide omnidirectional movement useful when aligning the robot with the toys.

A clean and easy to use API was designed to facilitate motor velocity commands. The end-user of the code only needs to set variables that represent velocity of the robot on the two perpendicular axes - x and y - and another variable that represents the angular velocity of the robot spinning about its center. Once the user specifies these three variables, a function that computes the forward kinematics of the robot will determine the resultant commanded velocities for the individual motors. The PI controller will automatically begin its calculations with the aforementioned velocities as input, and move the robot as commanded. Ryan Reyes will be put in charge of this subsystem, as this is a programming intensive section. The propulsion system and API will be fully developed by the end of September as one of the first goals, since it is a fundamental feature that the rest of the robot requires.

The system requires four Mecanum wheels which are about $70, 4 DC motors with encoders which are about 175$, 2 Solarbotics Dual Motor Drivers which are about $50, 2 sets of mounting brackets which are $20 dollars, a battery which is about $30, and an Arduino Mega which is $40.

#### 4.2.2.2.3 Test/Verification Plan

Each of the motors will be tested to see whether the actual velocity of the motor matches the commanded velocity within error. Once that test is successful, all four motors will be tested in tandem to see whether they all can sustain their commanded velocities. A specific test will be administered to verify whether the wheels will move at the same velocity when each motor is commanded with the same velocity.

Movement will be thoroughly tested on the plywood surface that will be used for the main track. Forward, backward, and side to side movement will be tested, as well as diagonal movement and in-place rotation. If these movements are consistent, i.e. the robot tracks the trajectory within tolerance, then movement is validated. In addendum: by design, the Mecanum wheels will experience slip. The range of possible movements will be restricted depending on the amount of slip experienced.

#### 4.2.2.2.4 Outcomes of Task

The motors will have velocity control individually, and in tandem. The robot will be capable of omnidirectional movement, and the interface for such commands will be simple and intuitive.

### <u>4.2.2.2 Subtask: Sensing Start</u>

#### 4.2.2.3.1 Objectives

The robot must be able to sense when the start LED has turned off, so that the robot may actuate its motors to begin its run down the track.

#### 4.2.2.3.2 Approach

The robot’s initial state will poll a switch interfaced with the Arduino Mega, that will be manually actuated by one of the team members once the robot is placed in the correct starting position. Once the switch is flipped, the robot will begin its competition routines. This is setup is required so that the robot does not run the competition routines while the team is handling it - moving it between heats - due to the noise. The Arduino Mega is interfaced via analog inputs with a photoresistor in ‘pull down’ configuration that is oriented toward the start LED. Once the switch is flipped, the Arduino polls the photoresistor, taking a moving average of the output voltage to eliminate noise. Once the average voltage is above a certain threshold, this indicates that the LED has dimmed and the robot should begin the heat. The code must then interface with the API discussed in 4.2.2.2 to command the robot with a forward velocity, and to engage it in ‘line following’ mode discussed in the following section.

Kurt Marsman will be set on this task. This task will occur as the work on the propulsion finishes. The only sensor required in this setup is a photoresistor, which can be bought extremely cheap in bulk.

#### 4.2.2.3.3 Test/Verification Plan

To test this requirement of the robot, the beginning of the track must be replicated: A painted white square with an LED, and a line emerging from the square. The robot will be placed over the LED so that the photoresistor can sense the light. The start switch on the robot will be actuated, and the robot should stay in place until the LED turns off. Once the LED turns off, the robot must begin forward movement, then engage in following the line that emerges from the square.

#### 4.2.2.3.4 Outcomes of Task

The robot, once correctly placed and signaled via manual switch,will be able to sense when the starting LED turns off. Once off, the robot will begin down the course, and engage in line following mode.

### 4.2.2.4 Subtask: Line Following

#### 4.2.2.4.1 Objectives

The robot should be able to successfully navigate the course by following the line. The rules state that the path will be deterministic: that the robot should always be able to differentiate between the main line and a branch if sensed correctly. The robot should begin in the starting square, follow the main line, and navigate down any branch it encounters. Once down the branch, the robot should enter ‘challenge mode’ where the robot will line up with the toy, and play the game. Once done with the game, the robot will reverse until the line is found again, navigate up the branch to the main line, and repeat the above sequence. Once the last challenge is played, the robot must follow the line and stop in the ‘finish box’. In software, the code must provide easy to use code hooks where challenge mode code can be triggered. This will enable modularity of the system.

#### 4.2.2.4.2 Approach

**A** grid of 3x3 infrared reflectivity sensors will be mounted center of the chassis undercarriage, facing the floor. These sensors detect the amount of reflected light bouncing off the ground. Since the lines are white, the sensors output a higher voltage correlated to the amount of light reflected from the line. This allows the robot to determine the location of the line in contrast to the black paint that delineates the rest of the track. The 3x3 grid will allow the robot to determine whether it is at an offset in relation to the line, or if the line continues at an angle relative to the heading of the chassis. The 3x3 grid will also be able to determine if there is a branch emerging from the point underneath the grid. The grid may be supplemented by other IR reflectivity sensors placed on the edges of the chassis, which will be used to detect the existence of the branch further from the center of the chassis. These reflectivity sensors will be interfaced to the Arduino via the analog inputs.

In software, the Arduino must poll the reflectivity sensors to determine the orientation of the line with respect to the chassis. For example, ideal operation is that the middle column of IR Reflectivity sensors indicate the line is underneath. This means that the robot is moving straight down the line, and the line bisects the robot chassis. Having the left column of the IR grid outputting high voltage indicates that the robot is too far to the right, and that it must take corrective action. Having the low left corner, center, and top right corner of the 3x3 grid high indicates that the robot is off at an angle relative to the line: -45°.

Depending on these states, one line following algorithm proposed will stop the robot, perform corrective action until the middle column of IR sensors is active - the robot is centered on the line, and continue following the line. The other line following algorithm proposed is a PID control which assigns weights to the 3x3 grid and uses the weights and status of the sensors as input. The output of the PID is the corrective velocity vectors required to reorient the robot on the line. The main drawback of this method is that it is more computationally intensive. However this method produces smoother results as the robot will smoothly change course as it continues down the line, in contrast to stopping forward motion to correct itself. Of course, both algorithms will interface with the velocity control API documented in Section 4.2.2.2. Since the team is using Git, it is possible to pursue both of these options in parallel to judge the efficacy of each, and extremely simple to switch between both algorithms at compile time.

Using the feedback from the 3x3 grid, and auxiliary sensors placed on the edge of the robot, branches that emerge from the main line can be sensed and recorded in the robot logic. The robot should then reorient itself using the Mecanum wheels to turn in place until it is lined up with the branch. The robot will then reengage the line following algorithm to navigate down the branch to the challenge. Since the order of the challenges will be decided before hand, the robot does not need to algorithmically determine what actions it must take. The particular ‘challenge’ mode operations can be coded to occur once the robot reaches the white box containing the challenge itself, as indicated by all high outputs on the grid of 3x3 and auxiliary sensors. The API provided to the user for challenge mode code must make it easy to switch the order of the challenges to complete.

Once the challenge mode code is complete, the robot must then reverse back onto the branch and engage line following mode to find the main line again. Once found, the robot recalls which direction it took to navigate down the branch, and reverses that to continue the correct direction down the main branch. For example, the robot takes a left turn to follow the branch. Upon reversing it finds the joint between the branch and the main line. The robot recalls it took a left turn to enter the branch, and decides that a right turn at the main line (the robot is reversing) will lead it to further challenges.

A special case that must be handled is when the robot loses the line completely. The robot must then engage in a search pattern until it can correctly orient itself on top of the line once more. Using the mecanum wheels, we do not have to rotate the robot for sideways motion, thus it will not lose its heading due to rotation. It will cover a grid as it moves so that it can find the line. An example movement would be: Forward; Left; Backward; Left; Forward; Repeat. This motion sequence lets the robot cover a square ‘patch’ on the ground, which is an ideal search pattern.

Finally, once the last challenge is played, the robot must continue following the line using the aforementioned algorithm until it finds the ‘Finish box’, upon which it has completed the course and should stop.

Ryan Reyes and Nils Bjeren will work together on this complex subsystem. The work on this subsystem will begin once the work on the propulsion subsystem has been finalized. This subsystem requires 9 or more IR reflectivity sensors which will cost upwards of $20 for all of them. If the processing load on the Arduino Mega is too much, the team may opt for another small microcontroller to distribute the load across the two microcontrollers.

#### 4.2.2.4.3 Test/Verification Plan

Several mock courses will be created that mimic the final course, each slightly different to test the limits of the algorithm implemented. The robot must successfully navigate the course, including the branches, and indicate that it is capable of running the challenge code at the appropriate time. If the robot loses the line, it must engage in search mode, and move in a pattern that will allow it to find the line with the highest probability.

#### 4.2.2.4.4 Outcomes of Task

Upon successful completion of this task, the robot will be able to navigate the course using the line sensors, and correctly handle the logic of entering challenge mode, or when it is lost.

### 4.2.2.5 Subtask: Alignment with Challenges

#### 4.2.2.5.1 Objectives

Upon reaching the end of the branch, the robot should engage in the first part of ‘Challenge Mode’: aligning itself with the toy. Once the robot is properly aligned with the toy, the Arduino Mega should indicate that it is ready for the manipulator to descend upon the toy and play the challenge.

#### 4.2.2.5.2 Approach

Since the toy can be placed at any location within the white box, the algorithms used to line up the robot with the toy or simplified considerably. The toy should be placed as close to the branch as possible, and centered upon that branch. This allows the robot to use dead reckoning from the line to find the toy.

One mechanical solution proposed is that the robot will have two brackets that can extend perpendicularly from the undercarriage of the robot. These brackets can close upon the toy, and ensure that they are held at the correct location for the manipulators to interface with them. An IR LED/Phototransistor pair mounted upon the brackets to detect whether the robot has come close enough to the toy for it to grab it. The IR LED/Phototransistor pair will be interfaced with the Arduino via analog inputs, and at least two additional motors will be required for this design. One motor will be used for the extension and retraction of the brackets, and another will be used to actuate the brackets open and closed. This will be done using a rack and pinion gear setup connected to the motor and brackets. An auxiliary component of this design is placement of pressure sensors on the inside of the brackets that can determine if the robot is holding the toy snugly. Concerning the software implementation of this method, the motors will require positional control. Ideally, servos would be used, but if the servos lack the rotational range, a motor with encoder with PID positional control will be used.

Another solution proposed is to use an array of ultrasonic sensors to determine the position of the toy relative to the robot. With that information, the robot can center itself, and drive close enough for the manipulators to interface with the toy. The ultrasonic sensors have an accuracy of ±3mm, so they would be sufficient for this task.

#### 4.2.2.5.3 Test/Verification Plan

The test plan for this requirement of the robot is to simulate the moment when the robot will transition from branch following to playing the game. Have the robot follow a line towards a toy that is slightly skew. The robot must then be capable of disengaging from line following mode, and aligning itself with the toy or the toy to itself, whether by mechanical or ultrasonic means.

#### 4.2.2.5.4 Outcomes of Task

The robot will be able to correctly align itself with the challenges to execute them.

### 4.2.2.6 Subtask: Internal Communications

#### 4.2.2.6.1 Objectives

Several microcontrollers will be present on the robot, at least one for controlling the main chassis and positioning, one to control the Simon and Rubik's Arm, and another to control the Etch-A-Sketch and Playing Card Arm. These microcontrollers must be aware of each other's state and the overall state of the robot in order to function as one unit. Thus, there needs to be a method of communication between these microcontrollers that ensures that all units are aware of the overall state.

#### 4.2.2.6.2 Approach

The robot will have a master-slave hierarchy of microcontrollers. The Arduino Mega dedicated to chassis functions will be designated as master, and will determine the overall state of the system, e.g. Line Following Mode, Challenge Mode. The two microcontrollers dedicated to the arms will be designated slaves to the Mega. Thus, the Mega will tell the subordinate microcontrollers when it is time to actuate their associated manipulators to play the challenges. Once finished, the slave microcontrollers will relay a message indicated that they are finished to the Mega, so that it will resume the completion of the rest of the heat.

The communications protocol that will be used in this setup is RS232 Serial communication. The Arduino Mega has several serial ports that it can use to communicate with the slave microcontrollers, and all Arduinos support at least one serial port out of the box. Thus, with the Arduino Libraries, this inter microcontroller interface is trivial to implement. Ryan Reyes will be put in charge of this task. This task will take place during the integration phase of the overall design.

#### 4.2.2.6.3 Test/Verification Plan

To test the inter microcontroller communication functionality, a test program will be written that checks whether the Arduinos actually can receive data from one another and make decisions from said data. The Arduino Mega will send a transmission to one slave Arduino, and said slave should indicate reception of correct data by lighting an LED. The slave should send acknowledgement to the master. The master then sends a similar message to the other slave, which in turn should light another LED. This test indicates that the Arduinos can successfully communicate with each other.

As the systems are integrated onto the main chassis, further tests can be performed by simply verifying if the robot can correctly enter challenge mode from line following mode, and play the correct game.

#### 4.2.2.6.4 Outcomes of Task

The microcontrollers on board the robot will be able to successfully communicate with each other, and relay important information such as the state of the robot, or what instructions to execute.

### <u>4.2.2.7 Subtask: Simon and Rubik's Arm</u>

#### 4.2.2.7.1 Objectives

Using the synergies between the requirements of playing Simon Says and the Rubik's Cube - mainly rotation in the horizontal plane - the objective is to design a manipulator that is capable of completing both of these tasks. This manipulator will be controlled with its own dedicated microcontroller, to increase modularity.

#### 4.2.2.7.2 Approach

Physically, the robot must be able to correctly interface with both the Simon Says game and the Rubik's Cube. The end effector of the manipulator has been proposed, and is seen in Section 2.3.1.1. The end effector will be directly connected to a servo for horizontal rotation. This structure will be able to match with the Rubik's Cube, and when torque is applied it will be able to turn one row of it, provided that the bottom of the cube is held stable. The alignment solution seen in Section 4.2.2.5.2 will be able to perform this role of holding the cube still. The servo has rotational control, which will allow the robot to command specific angles for the end effector to accomplish. With this, 180° rotation of one of the rows can be ensured. Julian Velasquez is in charge of the electromechanical design. Of course, he will be working with James Pace to determine if the design is mechanically sound.

Working in tandem with another servo that actuates the position of the manipulator itself, the end effector can be used to hit specific buttons on the Simon Says game. The manipulator will rise and fall to hit the buttons on the face of Simon Says, while the end effector actuator will turn so that the correct button is hit. This setup will exploit a bug with the Simon Says game that was explained in Section 2.3.1.2. The center button will always be pressed when pressing one of the color buttons, but the center button is ignored by the Simon Says game once the sequences are underway. The two servos will be interfaced with their dedicated microcontroller via PWM, and the slave microcontroller will communicate with the master microcontroller via Serial communication. The master Arduino will instruct this microcontroller to play either the Simon Says or the Rubik's cube. The slave must then indicate when it has finished.

To determine the sequence of buttons lit by the Simon Says game, an approach using sound identification was taken. The frequency of the sound associated with each button is determined using edge detection and a moving average of the input voltage from a microphone. The microphone circuit setup can be seen in Section 2.3.1.2. This circuit is interfaced with the arduino via an analog input port. The frequency is compared with a set of tolerances for each color, and once the color is identified, it is stored in an array containing the previous sequence of colors. Once stored, the Arduino will play back the sequence by mapping the stored sounds to specific manipulator configurations that will press the correct buttons in sequence. This must be done for a total of 15 seconds. Kurt Marsman is assigned to the software aspect of this arm, and will be working with Julian with interfacing with the servos. The work on this arm will be done simultaneously as the chassis and propulsion is built. This design will require about two servos, $40, a custom designed appendage, and a cheap microphone, in addition to a dedicated microcontroller

#### 4.2.2.7.3 Test/Verification Plan

To test this subsystem, the robot must be able to play either the Rubik's cube or Simon Says game correctly according to instructions from the master microcontroller.

#### 4.2.2.7.4 Outcomes of Task

The creation of a subsystem that is capable of playing both the Rubik's cube and Simon Says challenges.

### 4.2.2.8 Subtask: Etch-A-Sketch and Playing Card Arm

#### 4.2.2.8.1 Objectives

By examining the synergies between the Etch-A-Sketch and Playing Card challenges - mainly friction in the direction of the surface normal - the objective is to design a manipulator that can complete both the above challenges. This manipulator will be controlled with its own dedicated microcontroller, to increase modularity and robustness.

#### 4.2.2.8.2 Approach

A manipulator has been designed which is actuated by a servo motor at the base. The end effector consists of two vertical motors which match up with the positions of the Etch-A-Sketch knobs. On the end of these motors is a platform covered with an adhesive substance that gives torsional friction once in contact with the knobs. The two motors must then be actuated in the correct sequence of movements to replicate the letters IEEE on the face of the Etch-A-Sketch. Open loop control has been deemed sufficient for the completion of the task. All motors in this setup will interface with the microcontroller using PWM. The master Arduino will instruct the slave arduino when it is time to complete the challenges. Donovan Carey is in charge of the electromechanical design of the arm. he will be working with James Pace to ensure that the arm is mechanically sound. Chris Lewis will work on the software aspect of this arm .Work on this arm will occur simultaneously with the work on the other arm and chassis. This arm will require 3 motors and 2 motor drivers, which will run a total of about $60, in addition to a dedicated microcontroller.

To complete the Playing Card challenge, the arm will reutilize the adhesive on the ends of the motors to pick up the playing card from the top of the deck. The servo motor will just need to raise and lower the arm to ensure contact with the card and raise it up. An IR distance sensor may be used at the end effector to determine whether the card has been picked up.

#### 4.2.2.8.3 Test/Verification Plan

To test this subsystem, the robot must be able to play either the Etch-A-Sketch or Playing Card challenges correctly according to instructions from the master microcontroller.

#### 4.2.2.8.4 Outcomes of Task

The creation of a subsystem that is capable of playing both the Etch-A-Sketch or Playing Card challenges.

### 4.2.2.9 Subtask: Power Systems

#### 4.2.2.9.1 Objectives

Design the underlying power system so that the robot can operate completely autonomously for several heats.

#### 4.2.2.9.2 Approach

By looking at the specifications of the microcontrollers, servos, motors, and motor drivers, a suitable battery can be chosen that will run all of these systems. The current approach is to separate the power system for the propulsion from the rest of the subsystem power, since it requires much higher current and voltage compare to the other subsystems. The propulsion subsystem must be able to provide up to 3 amperes at 12 volts for the duration of several heats.

#### 4.2.2.9.3 Test/Verification Plan

To test the power system on the robot, the robot needs to be able to complete multiple heats without recharging the batteries.

#### 4.2.2.9.4 Outcomes of Task
A power subsystem will be created that can provide power to the robot for several heats.

### 4.2.3 Test/Verification Plan

The robot must complete a successful run of the course. This is comprised of the robot beginning when signaled, navigating the line and its branches to each of the challenges, completion of the Rubik's Cube, Simon Says, Playing Card, and Etch-A-Sketch challenges, finishing on the finish line, staying within a 1 ft by 1ft by 1ft box at start and end, and completion of the course under 5 minutes.

### 4.2.4 Outcomes of Task

A robot that will be able to compete and win the IEEE 2015 SoutheastCon Hardware Competition will be the outcome of this task.

## 4.3 Documentation

All of the engineers are expected to document their individual progress on a composition type notebook. The team is using Google Drive to document all progress and work performed on the robot. Since there is a vast amount of code that has to be programmed the team chose to use Git. Git is a open source distributed version control system designed to handle projects. This allows the team to make changes to code while keeping a copy of the original. It also allows the team to keep a log of changes to the code. Hardware diagrams and hardware design are documented using computer aided drawings (CAD) created by the engineers.

# 5 Risk Assessment

## 5.1 Errors in Code

Due to the nature of robotics, much of what makes the robot operate is code. The problem is that a simple error in one line of code will lead to system wide errors that are much more difficult to pinpoint and fix if not caught early on. Revision control is essential to prevent one team member from ruining days worth of work with simple errors; before any revisions are finalized and implemented, the head programmer must approve of the proposed changes by merging the branch in GIT. The act of merging a branch indicates that the code is competition quality.

## 5.2 Wires Coming Loose

A wire coming loose at the competition is a risk that has the potential to be catastrophic if no action is taken to prevent it. The design of the robot is very complex with many connections so in the event of a wire needing to be replaced, there is a possibility that the repair is made incorrectly and a microcontroller gets fried. In order to mitigate this risk, a neatly drawn wiring diagram, preferably in CAD based software, is essential to have on hand.

## 5.3 Motors on Chassis and Arms Breaking

The challenges presented for this installment of the IEEE SoutheastCon competition are mechanically intensive and involve quite a few servos and motors. Every single one has the potential of stripping gears or breaking in some form. The best strategy to prevent this situation is to ensure that all of the motors and servos are not placed under more stress than they are rated for. In the case of funds being left over after the main construction, the acquisition of a few extra motors and servos is worthwhile to make a quick fix on the robot if need be.

## 5.4 Voltage and Current Associated with the Design

While electronic circuit boards are fairly robust, the potential to apply too much voltage or current and destroy them is ever present. Thus it is important to ensure that all electronics are wired together correctly and care is taken to not over-duty the boards. Reverse current from the motors is another source of current to account for because when the wheels are allowed to roll, the motor essentially acts like a generator causing negative biased current to be formed in the system. The motor drivers selected have built in reverse current protection that stop current from flowing back into the arduino and potentially destroying it.

## 5.5 Non-Completion of Tasks

The whole basis for the competition is the completion of a set of tasks that the robot must do autonomously. The robot failing to complete all tasks will result in a loss at the College of Engineering competition and barrs advancement to the SoutheastCon competition. Once the design and construction of the robot is complete, much testing and refinement is to be completed to make sure the robot can successfully navigate the track and complete all tasks.

## 5.6 Budgetary Risks

Any project involving prototyping runs the risk of going over budget through the purchase of too many parts. In the design and prototype phase of the robot, multiple ideas for motors and servos are discussed between all group members and the correctly specified parts are purchased; the risk lies where a chosen motor does not meet requirements and must be replaced. Therefore, it is important that diligent engineering analysis is conducted before the purchase of any one component.

A second source of over budget risk is the devotion of too many man hours to the project. While it is fairly important to get the project and design done, too many hours spent working drives the price of the final design up because of the salaries of each of the workers. the simple mitigation strategy to this is the implementation of a gantt chart to manage everyone’s progress and ensure people are not dumping unnecessary time into the work.

## 5.7 No Walls on the Track

The course at previous SoutheastCon competition featured walls around the track in order to prevent robots from driving off of the track and going out of control. This time around, the track has no walls, meaning that if the robot were to go awry, the potential for damage to the robot is greater than last year. Extensive testing procedures and plenty of time for refinement is the key to prevent this from being an issue.

## 5.8 Non-Compliance with IEEE rules

Since this is a large scale competition, there is a fairly extensive list of rules and specifications that must be followed in order to even be eligible to compete. Failure to comply with any one of the stated rules will result in a penalty or immediate disqualification. Once the official version of the rules are released, they must be examined carefully and closely followed to avoid potential consequences.

## 5.9 Losing to the Other Team

In order to advance to the main competition, one hurdle must be overcome in order to get there; the robot must win the school level competition. The way to accomplish this is to design the robot more effectively than the other team, complete more challenges than the other team, and finish the course faster than the other team.

## 5.10 Battery Health and Lifespan

The battery is the life of the robot because without it, none of the electrical components on the robot can operate. Therefore it is vitally important that the battery is taken care of and not left in non-ideal environments. This is as simple as not leaving the battery in a hot car or in the bottom of someone’s backpack.

## 5.11 High Top Speed of the Robot

The propulsion capabilities of the robot is a legitimate concern because left unchecked, the robot has the potential to race across the floor and run into the wall of other robots at considerable speeds. This not only poses a risk to the structural integrity of the robot but potential hazards to other robots competing as well. A sensitive accelerometer is the way to stop this scenario from occurring because if the robot accelerates out of control, the accelerometer can stop the program and prevent any major damage to the robot.

## 5.12 Transportation

Durability is definitely a concern that must be kept in mind for all aspects of the design; that being said, the robot must be able to be transported safely to the competition in order to avoid any other problems from being created. A simple cardboard box that goes around the robot and a pedestal to hold the robot still are sufficient to protect the chassis during transit.

# 6 Qualifications and Responsibilities of Project Team

Senior Design is a class created with the intention of simulating the engineering design process in the industry as closely as possible. This includes top level design, technical papers, mathematical calculations, and hardware system implementation. This creates a team dependent environment allowing graduating seniors to not only physically implement the theory based courses taught through the majority of the engineering curriculum; but it also serves as a platform for the students to utilize some of the softer skills that aren't really focused on in the engineering coursework. In the industry, a system's team is dependent upon different expertise of the different members expected to bring their knowledge together to efficiently complete the project at hand. This usually includes bringing engineers from different ethnic cultures, experiences, and disciplines under one umbrella to efficiently solve a problem. Similarly our team is no different.

When assembling a team it is important for the team to have as much experience as possible in the various fields this specific project requires. This will not only allow for a more efficient usage of time, this allows the others on the team to develop a better understanding of the other fields that they may have not been as familiar with. This is a technique that can be used for Life Long Learning, which is an essential part of an engineer's career.  The table below shows each of the team members, their contribution to the project, their subsystem that they are working on, and their relevant experience that is brought to the team.

| Team Member's Name | Team Contribution | Relevant Experience/ Qualifications |
|---|---|---|
| Nils Bjeren | Team Manager/Team Lead | Extracurricular involvement. |
| Propulsion | | |
| Ryan Reyes | Head Program/ Control Systems Engineer | Robotic research working with propulsion systems |
| Simon/Rubik's Cube Arm | | |
| Julian Velasquez | Financial Advisor/ Power Systems Engineer | Is financially certified, and has research experience in power systems |
| Kurt Marsman | Secretary/Systems Engineer | Has secretary experience |
| Chassis Design/Weight Distribution | | |
| James Pace | Head Mechanical Engineer | Has experience with material science and mechanical systems |
| Etch a Sketch/Card Arm | | |
| Chris Lewis | Controls Engineer | Has computer programming experience. |
| Donovan Carey | Systems Engineer | Has done research in electro-mechanical systems |

**Table 3:** Team member roles

As has been stated previously, the team's objective is to win the competition. the goal is to get the full experience out of this course by learning the lessons that the course requires, and working as a team to ultimately win the SoutheastCon competition and bring the trophy back to the FAMU-FSU College of Engineering.

The team manager Nils Bjeren is a fourth year Electrical Engineering student who was born in Norway. His task in the project include overseeing the project designs, organizing team meetings, and serving as a liaison between the professor advisors and the team. He was voted into this role because of his urgency in learning the individuals in the teams expertise and guiding the members into a role that will allow them to be utilized where they can be efficient while at the same time learning something new. Because of his membership in the Toastmasters International society, his communication skills allows him to not only clearly organize the team, but also allows for sufficient dialogue between the professor advisors and team.

Ryan Reyes is a fourth year dual major who serves as the Head Programmer and Controls Engineer on the team. He is by far the most experienced of all the members on the team in that he has been the only one to defend an Honors thesis at the undergraduate level. His robotic research at CISCOR allows him to be placed at what is the toughest subsystem that this robot has. This is the movement and aligning of the robot. Experienced with encoders, it is much easier for him to take on this challenge compared to other members of the team. Because of his computer programming expertise, he has been given the task of overlooking all code before it is implemented as a whole into the robot. This will allow him to take a working code and write it more efficiently, allowing for an organized communication between the microcontrollers.

Julian Velasquez not only serves as the financial analyst on the team, he also is the Power Systems Engineer. Born in Colombia, Julian moved to Miami Florida when he was only 10 years old. Over a decade later he has moved up and has held the secretary and chair position for the FAMU-FSU Chapter of IEEE where he was financially certified by FSU. This combined with his pursuit in a Sales Engineering position post-graduation served as reasons that he would be selected to be in charge of the budget that we will use. He also has research experience with the MagLab doing Power Analysis. This expertise serves as a good source because Power distribution is one of the most important parts of this design. He is assigned to the working of the Rubik's/Simon arm.

Kurt Marsman serves as the team secretary. His primary goal is to ensure that all documents are correctly organized, make sure the team is following the protocols and format given by the course advisors and instructor, and take design and advisor meeting minutes. His experience as the FSU Paintball Club's secretary is the experience which landed him the job. He also serves as one of the system engineers primarily working on the Rubik's/Simon subsystem.

Usually there is not a Mechanical Engineer on the SoutheastCon competition teams due to the lack of mechanical design required. This year's competition requires a lot more mechanics and dynamics than the previous years. Because of this there are factors that must be taken into consideration that aren't always the first thought of Electrical and Computer engineers. These include weight, torque, and bend. This is why the team includes a mechanical engineer in

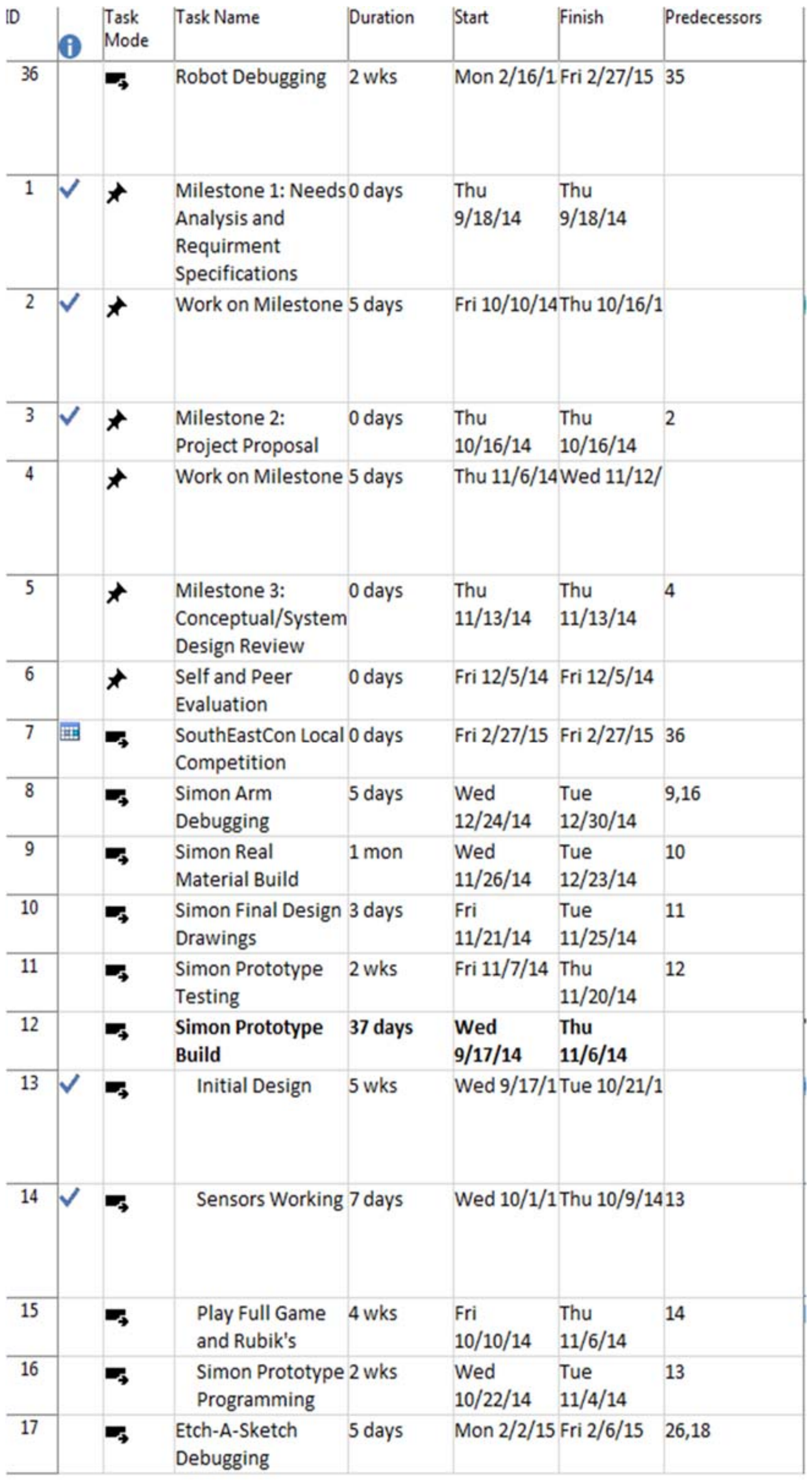
James Pace who is in charge of making sure that we are within those restraints. His coursework in materials and weight distribution allows for the team to foresee issues and design around them that may not have been so obvious to the rest of the team. His minor chassis and graphical software experience provides a precise and efficient means of design.

Chris Lewis is a computer engineering major with the role of the manipulators controls engineer job. As previously stated there are many dynamics involved in this years competition. This results in many motors, and servos. Microcontrollers are a source used to efficiently and accurately control these devices to do a specific task. Chris has the role of programming the microcontrollers to have the manipulators do the desired task, in the desired time, efficiently. Being the second best programmer on the team it is his task to also assist any of the other team members with less coding experience to get a working code to be overlooked. He primarily works on the etch a sketch/card arm.

Donovan Carey serves as the hardware systems engineer on the team. His role is to simply design and manufacture the manipulators being used for the events. His primary work is with the etch a sketch/ card arm. His research at the HPMI lab where he builds electro-mechanical testing systems allows him the experience to help design some of the ideas to play some of the challenges. Because of his pursuit in entering the industry as a systems engineer in electro-mechanical systems, this can serve as a great learning experience for him to learn more about the dynamic devices which allow accurate movement.

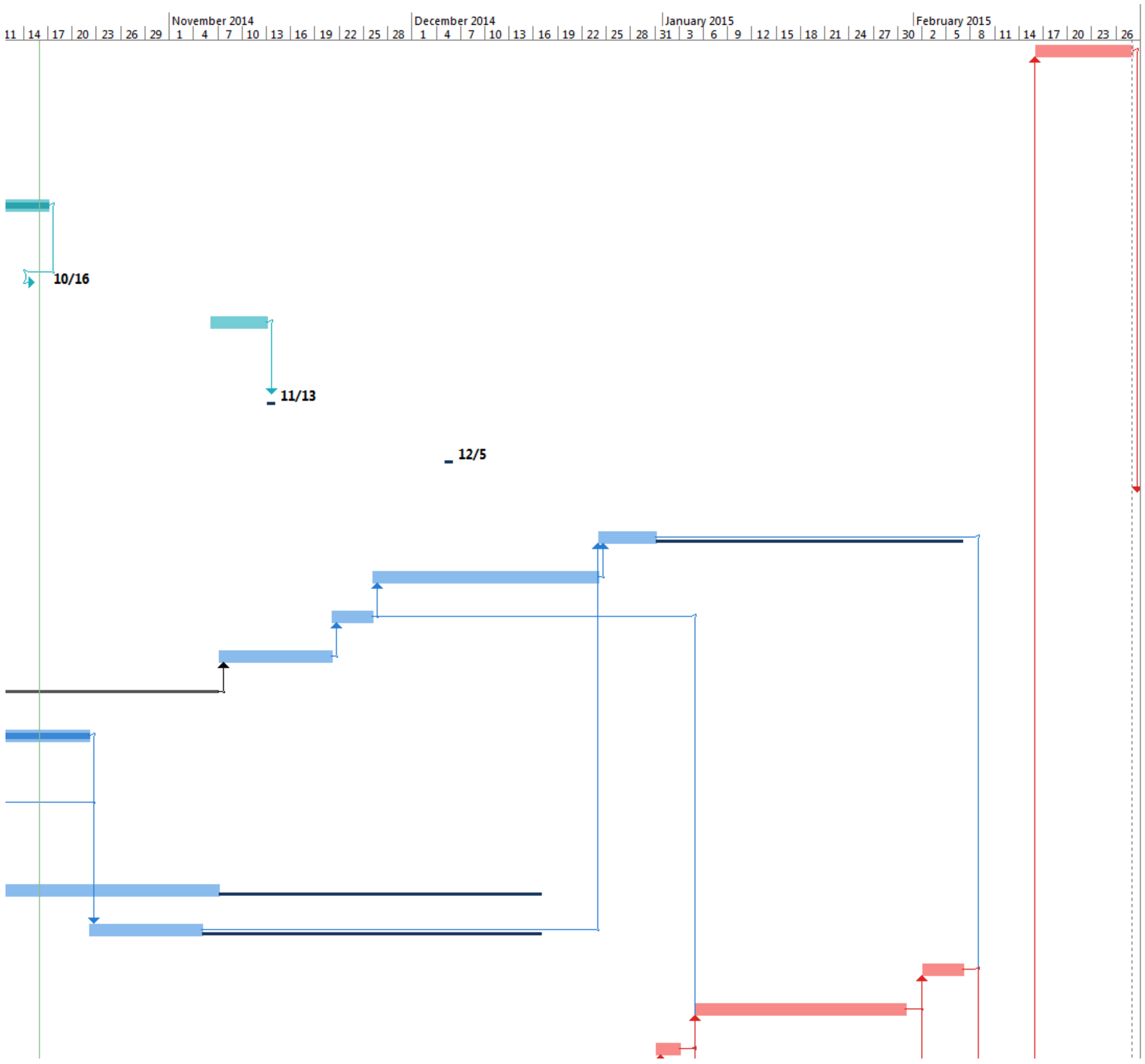
It is important to note that though each of the members may have certain experience In different areas and may be placed in a position because of it; this is still a design project intended for the team to learn from one another. Because of this no member is limited to one subsystem and is allowed to help each other with the other subsystems. This is the best way for the members of the team to be able to implement the best design concepts, and learn from one another.

# 7 Schedule



| ID | ℹ | Task Mode | Task Name | Duration | Start | Finish | Predecessors |
|---|---|---|---|---|---|---|---|
| 36 | | ➡ | Robot Debugging | 2 wks | Mon 2/16/1 | Fri 2/27/15 | 35 |
| 1 | ✓ | 📌 | Milestone 1: Needs Analysis and Requirment Specifications | 0 days | Thu 9/18/14 | Thu 9/18/14 | |
| 2 | ✓ | 📌 | Work on Milestone | 5 days | Fri 10/10/14 | Thu 10/16/1 | |
| 3 | ✓ | 📌 | Milestone 2: Project Proposal | 0 days | Thu 10/16/14 | Thu 10/16/14 | 2 |
| 4 | | 📌 | Work on Milestone | 5 days | Thu 11/6/14 | Wed 11/12/ | |
| 5 | | 📌 | Milestone 3: Conceptual/System Design Review | 0 days | Thu 11/13/14 | Thu 11/13/14 | 4 |
| 6 | | 📌 | Self and Peer Evaluation | 0 days | Fri 12/5/14 | Fri 12/5/14 | |
| 7 | ▦ | ➡ | SouthEastCon Local Competition | 0 days | Fri 2/27/15 | Fri 2/27/15 | 36 |
| 8 | | ➡ | Simon Arm Debugging | 5 days | Wed 12/24/14 | Tue 12/30/14 | 9,16 |
| 9 | | ➡ | Simon Real Material Build | 1 mon | Wed 11/26/14 | Tue 12/23/14 | 10 |
| 10 | | ➡ | Simon Final Design Drawings | 3 days | Fri 11/21/14 | Tue 11/25/14 | 11 |
| 11 | | ➡ | Simon Prototype Testing | 2 wks | Fri 11/7/14 | Thu 11/20/14 | 12 |
| 12 | | ➡ | **Simon Prototype Build** | **37 days** | **Wed 9/17/14** | **Thu 11/6/14** | |
| 13 | ✓ | ➡ | Initial Design | 5 wks | Wed 9/17/1 | Tue 10/21/1 | |
| 14 | ✓ | ➡ | Sensors Working | 7 days | Wed 10/1/1 | Thu 10/9/14 | 13 |
| 15 | | ➡ | Play Full Game and Rubik's | 4 wks | Fri 10/10/14 | Thu 11/6/14 | 14 |
| 16 | | ➡ | Simon Prototype Programming | 2 wks | Wed 10/22/14 | Tue 11/4/14 | 13 |
| 17 | | ➡ | Etch-A-Sketch Debugging | 5 days | Mon 2/2/15 | Fri 2/6/15 | 26,18 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 18 | | ➡ | Etch-A-Sketch Real Material Build | 1 mon | Mon 1/5/15 | Fri 1/30/15 | 19 |
| 19 | | ➡ | Etch-A-Sketch Final Design Drawings | 3 days | Wed 12/31/14 | Fri 1/2/15 | 20 |
| 20 | | ➡ | Etch-A-Sketch Prototype Testing | 2 wks | Wed 12/17/14 | Tue 12/30/14 | 21 |
| 21 | | ➡ | **Etch-A-Sketch Prototype Build** | **65 days** | **Wed 9/17/14** | **Tue 12/16/14** | |
| 22 | ✓ | ➡ | Initial Design | 2 wks | Wed 9/17/1 | Tue 9/30/14 | |
| 23 | | ➡ | Mount Motors | 7 days | Wed 9/17/1 | Thu 10/2/14 | 22 |
| 24 | | ➡ | Add card pickup | 3 days | Thu 10/2/14 | Tue 10/7/14 | 23 |
| 25 | | ➡ | Fix Alignment | 2 wks | Wed 9/17/1 | Tue 10/7/14 | 23 |
| 26 | | ➡ | Etch-A-Sketch Protoype Programming | 2 wks | Wed 11/26/14 | Tue 12/16/14 | 22 |
| 27 | | ➡ | Chasis Real Material Build | 1 mon | Thu 1/8/15 | Wed 2/4/15 | 28 |
| 28 | | ➡ | Chasis Final Design Drawing | 3 days | Mon 1/5/15 | Wed 1/7/15 | 10,19 |
| 29 | | ➡ | **Line Following** | **35 days** | **Wed 9/17/14** | **Tue 11/4/14** | |
| 30 | ✓ | ➡ | Initial Design | 1 wk | Wed 9/17/1 | Tue 9/23/14 | |
| 31 | ✓ | ➡ | Forward Line Following | 1 wk | Wed 9/24/14 | Tue 9/30/14 | 30 |
| 32 | | ➡ | Sideways Following | 1 wk | Wed 10/1/14 | Tue 10/7/14 | 31 |
| 33 | | ➡ | Improve Sensor Grid | 2 wks | Wed 10/8/14 | Tue 10/21/14 | 32 |
| 34 | | ➡ | Improve Controls | 4 wks | Wed 10/8/14 | Tue 11/4/14 | 32 |
| 35 | | ➡ | System Integration | 1 wk | Mon 2/9/15 | Fri 2/13/15 | 8,17,27,29 |


November 2014
December 2014
January 2015
February 2015
11 14 17 20 23 26 29 1 4 7 10 13 16 19 22 25 28 1 4 7 10 13 16 19 22 25 28 31 3 6 9 12 15 18 21 24 27 30 2 5 8 11 14 17 20 23 26
10/16
11/13
12/5

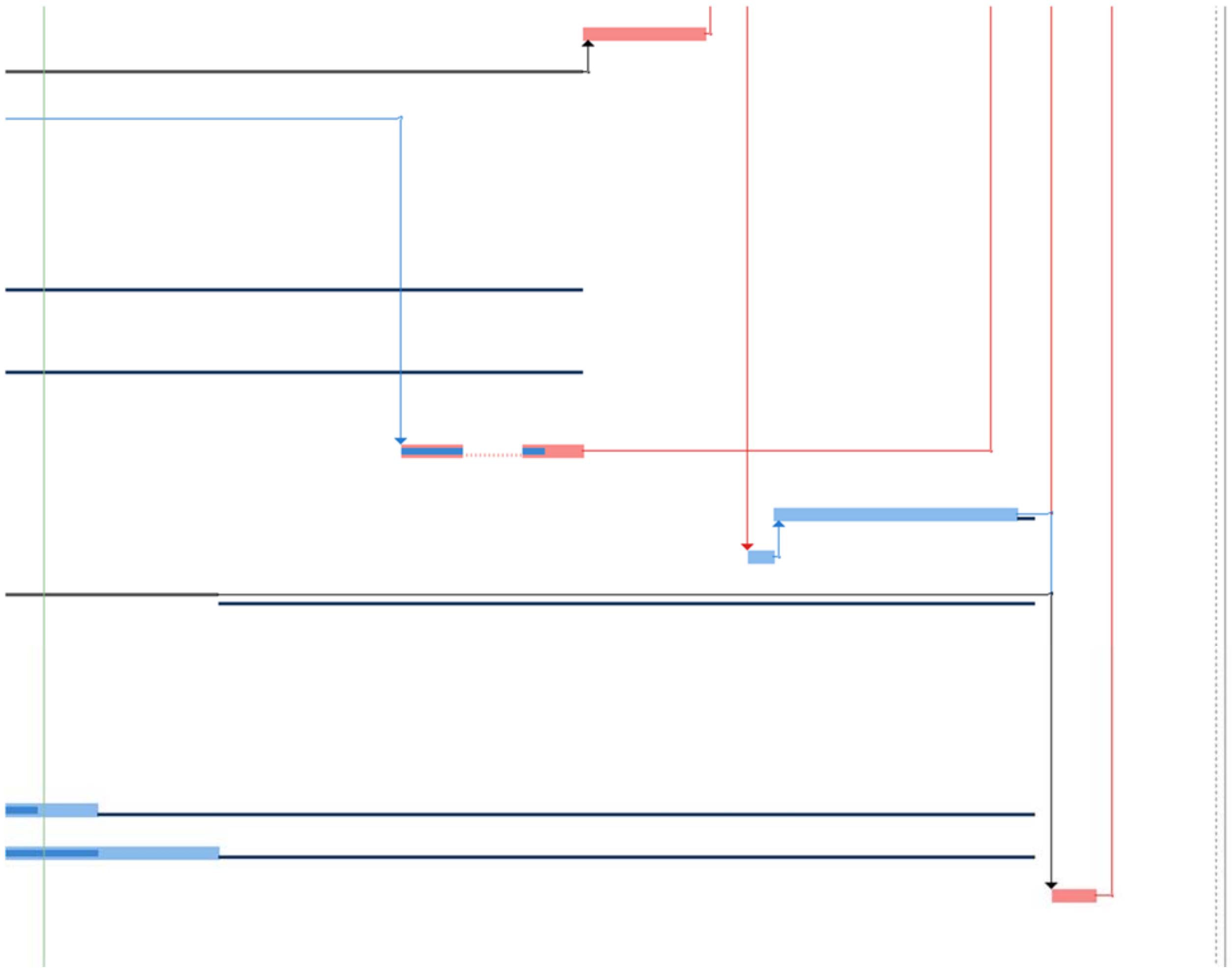


**Figure 13:** Gantt Chart

The Gantt chart above shows the various tasks and milestones associated with the project, as well as the time periods they will be accomplished in. The tasks marked in red make up the critical path of the project, which means the team must complete those tasks by the times indicated to complete the project in the time frame they want. The time periods to completion are purposefully conservative, so the team can accurately determine when they are off schedule.

### 7.1 Specific Deadlines

| Event | Date | Individuals Responsible |
|---|---|---|
| Simon Arm Prototype Complete | 11/6/2014 | Julian, Nils |
| Etch-A-Sketch Prototype Complete | 12/16/14 | Donovan, Chris, James |
| Line Following Complete | 11/4/14 | Nils, Ryan |
| Final Chasis Built | 2/4/15 | James |
| Local Competition | Late February | Whole Team |

**Table 4:** Important Deadline Summary

Table # summarizes specific important internal milestones for the team and the team members responsible. The team also sets more specific goals based on the ones above for each member on a week by week basis. The team will be moving as fast as possible to maximize testing time before the competition. As tasks are completed, the above deadlines may be adjusted to reflect the team's aggressive pace or any problems encountered.

# 8 Budget Estimate

| Category | Cost |
|---|---|
| Wheels | $80.00 |
| Motors | $300.00 |
| Batteries/Chargers | $150.00 |
| Microcontrollers | $300.00 |
| Electronics | $200.00 |
| Misc. Mechanical Parts | $170.00 |
| **Total:** | **$1,200.00** |

**Table 5:** Milestone 1 budget estimate

| **Purpose** | **Items** | **Cost** |
|---|---|---|
| Simon Frequency Analysis | Miscellaneous Electronics | $11.47 |
| Propulsion | Motor Drivers | $47.95 |
| Battery Charger for Propulsion/Manipulators Batteries | Battery Charger | $40.08 |
| Batteries for Manipulators | Battery | $27.75 |
| Simon/Rubik's Manipulator | Servo Motor | $19.40 |
| Propulsion | Miscellaneous Equipment | $51.89 |
| Propulsion | Motors for Propulsion | $177.25 |
| Etch-a-sketch manipulator | Gearmounts | $20.35 |
| Etch-a-sketch manipulator | Miscellaneous equipment | $20.11 |
| Mecanum Wheels | Propulsion | $71.76 |
| | **Total:** | **$508.01** |

**Table 6**: Currently specified parts cost

| **Personnel** | **Total Hours per Semester** | **Total Hours Worked Both Semesters** | **Total Base Salary ($30 per hour)** | **Total Salary + Fringe Rate of (29%)** |
|---|---|---|---|---|
| 1 Group Member | 192 | 384 | $11,520 | $14,860.8 |
| Whole Team (7 Members) | 1,344 | 2,688 | $80,640 | **$104,025.6** |

**Table 7**: Estimated personnel cost

| **Direct Cost** | **Direct Cost + Fringe Rate + Overhead rate (45%)** | **Direct Cost + Overhead Rate (45%)** |
|---|---|---|
| $81,148.01 | $151,573.74 | $117,664.62 |

**Table 8:** Estimated direct and overhead costs

| **Category** | **Expense ($)** |
|---|---|
| Supplies and Small Items (Current) | $508.01 |
| Additional Supplies and Small Items (Projected) | $691.99 |
| Direct Cost + Fringe Rate + Overhead rate (45%) | $151,573.74 |
| **Total Project Cost:** | **$152,773.74** |

**Table 9**: Total project cost

# 9 Deliverables

The fully functioning robot should be able to successfully complete the course in SouthEastCon 2015. The robot is expected to start on its own after a red LED on the track turns off. After the start the robot is expected to follow white lines to each of the four obstacles and respectively to the finish line. The four obstacles are: Etch-a-Sketch, Simon, Rubik's cube, and a deck of cards. The robot is expected to draw the letters "IEEE" on the Etch-a-Sketch, play simon for 15 seconds; correctly matching the output of sounds and colors, turn any side of the Rubik's cube 180 degrees, and pick up a card from a deck of cards and take it all the way to the finish line. To successfully do this the robot must finish the entire course in under 5 minutes. The finished product should be ready by the end of 2014. This will allow the engineers to spend all of spring semester improving the robots functionalities and increasing the robots performance for the competition. SouthEastCon 2015 will take place April 9th-12th, 2015.

To track the progress of the project the engineers are expected to keep records of all their work throughout the semester. The engineers are also expected to present their progress to their main advisor and coordinator every two weeks. Aside from these meetings the engineers are expected to complete: Team Formation Forms, Code of Conduct Agreement, Milestone 1: Needs Analysis and Requirement Specifications, Milestone 2: Project Proposal, Milestone 3: Conceptual/System-Level Design Review, Self & Peer Evaluation, and Team's Web-Based Engineering Log. The three milestones will be in the forms of written report as well as a group presentation. The group presentations will be presented to main advisor, external technical advisor, additional reviewer, external coordinator and any student who wishes to attend. The advisors will be in charge of grading the students work and presentations. A grade will be given for each written report as well as each presentation.

Deliverables:

- Milestone 1:Needs Analysis and Requirements Specification **(9/18/2014)**
- Milestone 2: Project Proposal and Statement of Work **(10/16/2014)**
- Milestone 3: System Level Design Review **(11/13/2014)**
- Milestone 4 **(TBA)**
- Milestone 5 **(TBA)**
- Milestone 6 **(TBA)**
- Competition-Ready Robot **(4/9/2015)**