IEEE SoutheastCon Hardware Competition

Chase Garner, Sawyer Hall, Lexi Sheeler, and Daniel Summers
Electrical and Computer Engineering Department
Tennessee Technological University
Cookeville, TN
ncgarner42@tntech.edu, sjhall43@tntech.edu, aasheeler42@tntech.edu, dbsummers42@tntech.edu

I. INTRODUCTION

A. Background

This project aimed to build an autonomous robot to compete in the South Eastern Conference (SECon) Hardware Competition. The course consisted of several challenges that the team could complete for points in the competition. The team used a combination of computer and electrical engineering concepts to build the robot, in addition to several mechanical concepts which were developed throughout the course of the project. From here, the team will also have to use all previous concepts of STEM related fields to build the best product for IEEE and Tennessee Technological University.

B. Motivational Analysis

The team designed and built a fully autonomous robot which traversed a track, collected beads, and placed said beads in randomized locations along the track. Initially, the plan was to build the robot to respond to manual command to assist with mechanical errors, and then move on to complete autonomy. The team consisted of two computer engineers, which handled software and microcontrollers, and two electrical engineers, which handled power and controls.~ The team aspired to revive the Tennessee Tech IEEE chapter, which once had a significant presence at SECon, and inspire more students to join and compete in the future.

C. Problem Formulation

SECon is an annual IEEE event comprised of several competitions. One competition is the hardware competition, where students from schools across the region design and build a robot to compete in a unique challenge. Because the conference was held in Mobile, Alabama, the competition was Mardi Gras themed to reflect the history and origin of the festival. This years' hardware competition required the robot to navigate an L-shaped street within a three-minute time limit. In addition to navigation, the robot completed challenges to earn points. The tasks include removing beads from a "tree", placing the beads in "trash bins" or shooting them into fishnets, and moving a marshmallow off the roadway into gaps in a barricade. The game board layout can be seen in **Figure 1**.

D. Constraints

There were several specifications that the design needed to adhere to in the competition rules. The first, and potentially most important constraint, to meet was that the robot required complete autonomy. To meet this constraint, the robot had to be capable of decision-making. In addition, the robot was required to be contained within one cubic foot at the start of the competition, but could expand beyond that specification after the run has begun. It is also specified that the robot may not break apart and must remain one unit for the entirety of the run. Finally, the robot may not cause damage to the track or harm bystanders. Any violation of these constraints will result in immediate disqualification at the judges' discretion. A full list of rules can be seen in **Figure 2**.

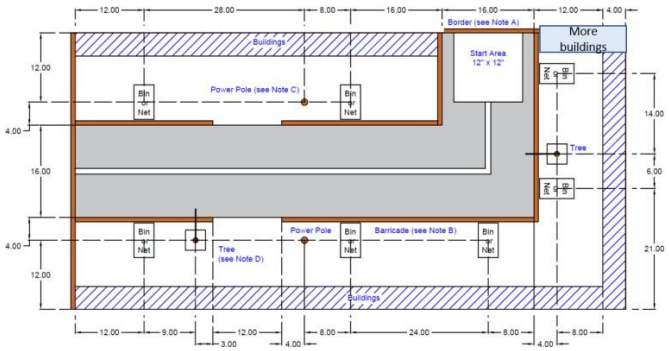


Fig. 1. **Gameboard Design Layout**

Points	Task
+1 pt	Pushing a marshmallow off the roadway
+1 pt	Each bead dropped into a bin
+1 pt	Each trash bin with one or more beads in it
+1 pt	Each fish net with one or more beads in it
+2 pts	Each bead thrown into a fish net
+2 pts	Having a display that moves mechanically at least 4"
+2 pts	Playing a song
+2pts	Having a display that lights up
+5 pts	Completing the track, in one direction
+10 pts	Completing the track, in both directions (full loop)
-1 pt	Each bead/marshmallow attempted, but left on roadway
-5 pts	Touching any obstacles: barricade, trash bin, building, or power line
-10 pts	Knocking over any obstacles: barricade, trash bin, building, or power line
Disqualification	Damage to the game board, at the Judges discretion. Venturing off of the game board. Touching the robot after hands off. Remotely operating the robot after hands off.

Fig. 2. **SoutheastCon Hardware Rules**

Chase Garner, Sawyer Hall, Lexi Sheeler, and Daniel Summers is with Electrical and Computer Engineering, Tennessee Technological University

E. Overall Idea

The project consisted of a variety of sub-systems; each system requiring specific hardware, software, testing/verification, and skills to be developed. Individual subsystems required interfacing to the central controller via standard protocols. These sub-systems were outlined, designed, developed, and tested both independently and concurrently. By ensuring each sub-system complies to the standard interfacing protocols and meets the design requirements specified by the overarching system, each sub-system was developed independently and then combined to complete the system.

Furthermore, the robot was broken down into seven systems: main controller, power, navigation, drive train, appendages, shooting mechanism, and peripherals. Each system was then broken down further into atomic subsystems. There were three microcontrollers: the main controller, a navigation controller, and a peripheral controller. The main and navigation controllers communicated via serial communication protocols, and the peripheral controller only required power to operate. The main controller was responsible for interconnects between systems, excluding power which directly connected with all systems. In the next phase of the design process, component specifications were defined first at an atomic level, working up until the entire system had been defined.

F. Salient Outcomes

The team's success was divided into two parts: engineering success and competitive success. The engineering measures of success were evaluated on the basis of meeting the specifications and objectives of the competition while staying within the constraints set forth by the SECon competition. The team carried out many test runs and evaluations throughout the process of building the robot. The success of these test runs was dependent on the performance of the robot given the objectives and constraints.

The building process of the robot occurred in many stages. First, the team wanted to create a robot that could move manually. When the team deemed the method and range of motion as acceptable, they would move on to the next task. As the robot become more advanced, the team would attempted to incorporate additional functionality. If the team was able to build a robot that incorporated all of the objectives set forth by the competition, and those functionalities were determined to fulfill the constraints, the engineering effort of the team was deemed a success.

The team's success was further evaluated on its ability to produce a product that performs well in competition. If the robot performed well against opponents, the team would consider the project a success. If the final product met the measures of success shown in **Figure 3**, the overall effort of the project would be considered a success.

Beyond the competition itself, this project aimed to serve as a way to demonstrate the quality of both the Tennessee Tech Electrical and Computer Engineering (ECE) Department and the Tennessee Tech IEEE Chapter. The Tennessee Tech IEEE chapter has a longstanding tradition of fielding competitive teams at SECon, with this project intending to serve as such. Additionally, due to the ongoing pandemic, interest and involvement in the IEEE chapter has massively decreased over the last two years. The team aspired to generate interest in IEEE from ECE students, while also providing IEEE members with hands-on experience with programming. Because of the involvement of the IEEE chapter, the design of the peripheral system was changed to use its own microcontroller and an analog circuit to control LEDs, a display, and a sound system. This allowed the IEEE members to work on these systems independently of the robot's development, and allowed them to receive experience in analog and digital circuitry and programming.

Measure of Success	Target Value
Size Specification	12"x12"x12"
Battery Life	1 hour @ 6 Amps of Current Draw
Bead Pick %	90%
Bead Pick Time	20 Seconds
Shot Accuracy	95%
Shot Time	20 Seconds
Net Detection	100%
Beads in All 3 Compartments	Pass/Fail
Marshmallow Moved	85%
Traversal Speed	2 ft/s
Run Time	<3:00
Run Points	40

Fig. 3. **Team's Measure of Success**

G. Organization

This document describes the process in which the team designed the robot; describing both individual systems as well as the product as a whole, along with its performance, concluding with the timeline, ethics, lessons learned, and final conclusions.

II. Literature Review

Furthermore, this project may open a door to integrating automated vehicles on Tennessee Tech's campus. Automated vehicles are already being used on some campuses in the United States. For example, the food delivery company GrubHub has partnered with 250+ college campuses to implement small, automated food delivery vehicles. Food delivery is just one of many opportunities to integrate these vehicles. Autonomous vehicles may also be used in sanitation, tours, crime prevention, etc. Of course, with increasing automation, there is the potential to eliminate some jobs. As in the GrubHub example, there are inevitably some food delivery drivers that are going to be displaced by this new technology. In the example of sanitation and tours, there are currently full time as well as student workers who handle these jobs. The ethical questions of this technology will need to be answered as it becomes a more viable solution for colleges

across the country. There is also the question of safety in implementing these vehicles on campuses. Currently, Tech's campus sees considerable car traffic. Autonomous vehicles have the potential to pose a hazard to divers on campus. As Tech becomes a more foot-oriented campus, as is outlined in the "Master Plan," these risks will be reduced considerably.

III. Engineering Standards

Three main standard affected the project: ANSI C18.2M, ANSI C18.3M, and American Wiring Standards. ANSI C18.2M deals with matters of charging Lithium-Ion batteries, ANSI C18.3M specifies tests and requirements for safe operation of lithium-ion batteries, and the American Wiring Standards discerns what gauge of wire is needed for the required voltage and current. Additionally, the team must adhere to the IEEE code of conduct.

IV. Methodology

A. Theory and overall idea of solution

At the beginning of the design process, key competition rules and point gaining tasks were identified. The point breakdown played into the first design phase in several ways. The scoring scheme was broken down and the maximum number of achievable points was determined, which allowed for selecting which way the robot could earn maximal points without the need to complete every possible task. This led to a design that was targeted towards shooting beads into the nets instead of trying to place them into the cups. Along with this decision, the team also elected to not place high importance on removing the marshmallow from the track because it was only a one-point deduction if left on the track. No designs to do so were identified by the team, so focus was instead placed on areas which were integral to being competitive, such as the navigation system and drive train.

B. Complete design process

1) ***Chassis****:* The main purpose of the chassis is to house each subsystems of the robot. The combination of the chassis' structure and any external subsystems had to conform into a 1 cubic foot form factor at the beginning of the competition. Once a competition run had begun, the robot was allowed to extend beyond this size. Another constraint placed on the chassis was the limited turn radius of the track. If the chassis' footprint measured one square foot, the robot would not be able to rotate on the narrow portions of the track. **Figure 4** illustrates the necessity of conforming to this specification. The track itself measures 16 inches across, and the robot's width and length are each specified at 12 inches or less, which allowed the robot to traverse the track in its initial direction. However, if the robot attempts to turn, its diagonal length of 16.9 inches will exceed the track width. Designing around this constraint involved dividing the chassis into a stationary lower compartment, which is used to house the electronic components (microcontrollers, drivetrain, power supply system), and an upper platform attached to a turntable. This turntable was designed to rotate 180 degrees to allow the robot to fire beads perpendicular to the track in either horizontal direction. Lastly, the chassis was designed to maximize the reach of the robotic arm, wising a tiered design to incorporate an upper platform. **Figure 5** through **Figure 8** shows a 3D model depiction of the robot.

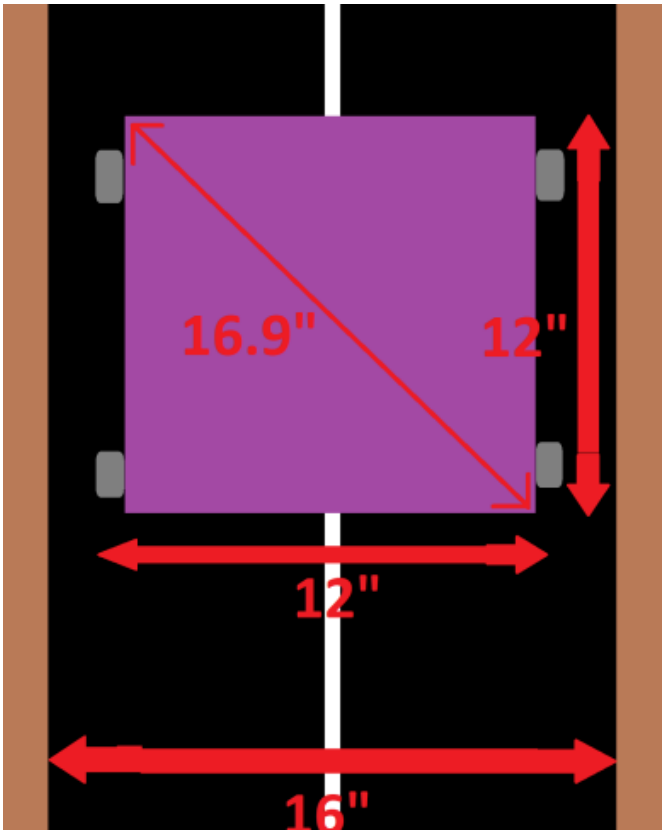


Fig. 4. **Size of Robot vs. Track Size Dilemma**

The lower compartment of the chassis measured approximately 9.175 x 11.8 x 3.25 inches and served as the housing of the major electronic components including the microcontrollers, drivetrain, and power supply system. The chassis was constructed using acrylic sheets. The upper chassis will be constructed as an 9.175 x 11.8 inch acrylic platform that will be affixed to the turntable. The rear half of this platform was raised an additional 2.15 inches and will serve as a platform for the robotic arm and shooting mechanism motor. A gap in this platform allows the robotic arm to fold into itself before the competition begins. The horizontal, load-bearing platforms of the robot will be constructed from 1/4 inch thick acrylic sheets, while the remaining acrylic components will be 1/8 inch thick. This choice was made in order to reduce the flex on the acrylic. The total necessary acrylic is shown in the **Figure 9**.

The upper chassis housed the shooting mechanism and a 3D printed compartment for storing and distributing the beads. The shooting mechanism housing rests on a 5.75 x 9.25 inch platform. Three vertical walls extend upward supporting the belt's rollers and the storage compartment. This compartment measured 5.25 x 3.75 x 2.8 inches and contained a false bottom that holds the beads until a motor slides the false bottom free

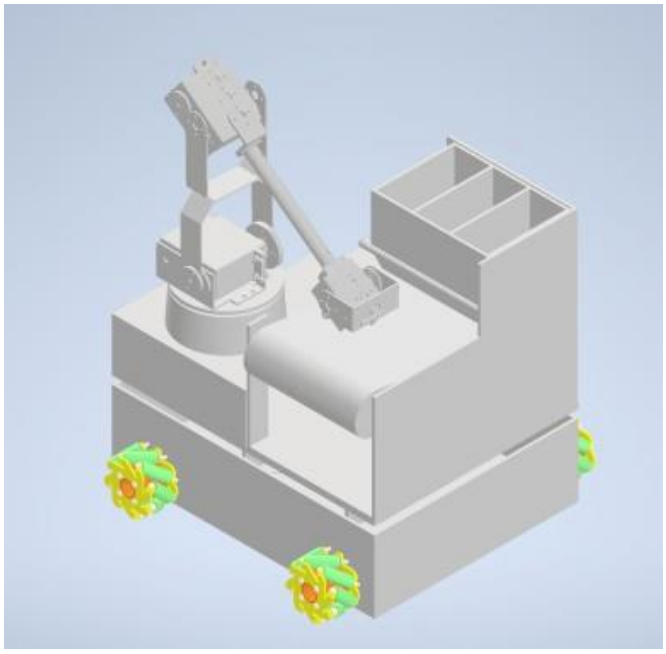


Fig. 5. **Chassis Design Overview**

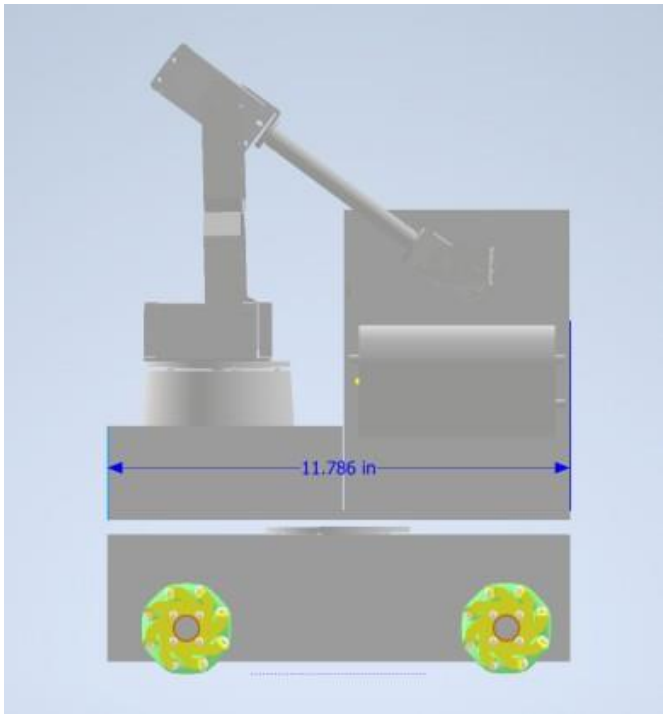


Fig. 6. **Chassis Length**

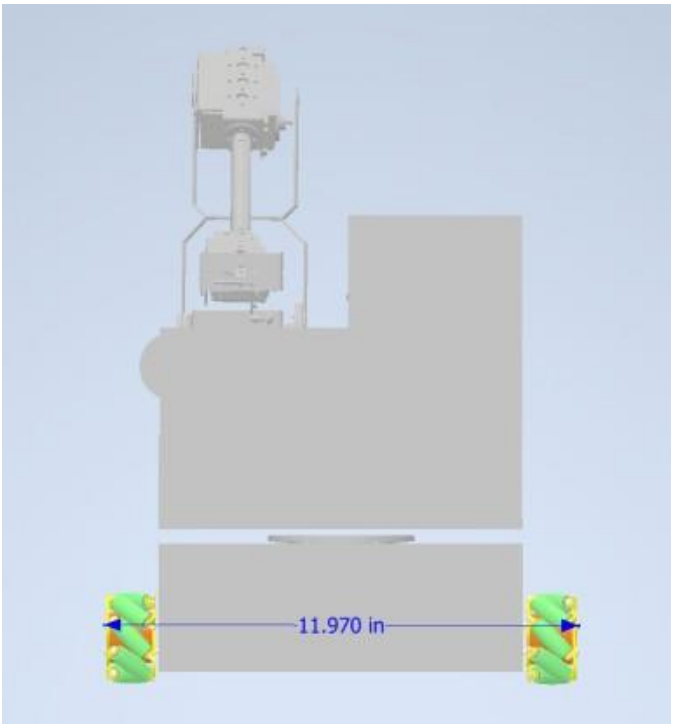


Fig. 7. **Chassis Width**

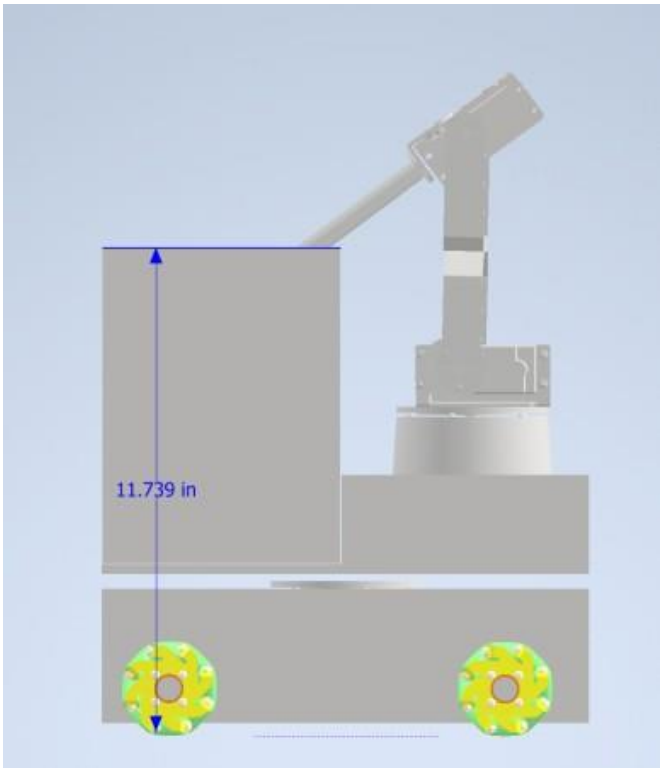


Fig. 8. **Chassis and Height**

	Dimensions	Acrylic (sq. inches)
.25"		
Upper Chassis	9.175 x 11.8	108.265
	6 x 4.920 + 4.25 x 3	42.27
Lower Chassis	9.175 x 11.8	108.265
	Total:	258.8
.125"		
Upper Chassis	2.15 x 6	12.9
	2.15 x 3	6.45
	2.15 x 3	6.45
Lower Chassis	11.8 x 3.25	38.35
	11.8 x 3.25	38.35
	9.175 x 3.25	29.81875
	9.175 x 3.25	29.81875
Shooting Mech. Housing	4.835 x 9.125 + 2.835 x 4.375	56.5225
	5.75 x 9.125	52.46875
	4.835 x 0.849	4.104915
	7.67 x 5.75	44.1025
	Total:	319.336165

Fig. 9. **Needed Acryillic**

of the container. A geared tab on one side of the false bottom was included to facilitate this movement. A motor will be mounted to the compartment to drive the false bottom. The compartment will be 3D printed due to the tight tolerance between the compartment and the false bottom as well as to reduce weight. Models of the bead compartment and false bottom can be seen in **Figure 10** and

.

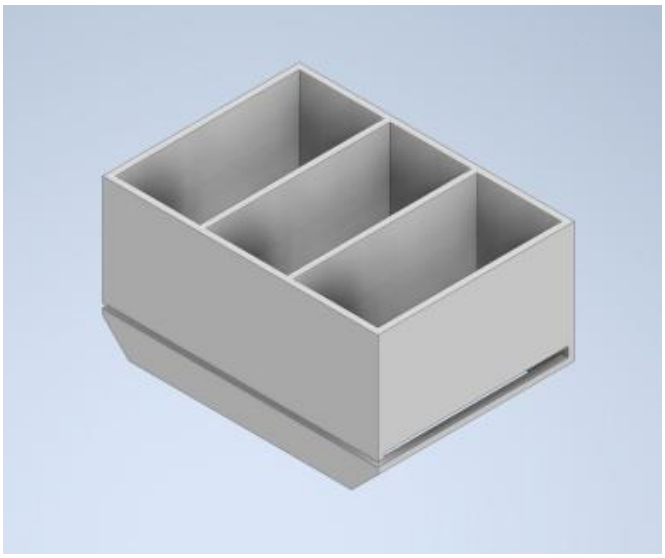


Fig. 10. **Bead Compartment Design**

The varying aspects of the chassis were laser cut to include finger joints for connecting the acrylic sheets together. Additionally, a 3D Printed mounting bracket, seen in **Figure 12** were included in each corner to connect the pieces. A brass press-fit insert was inserted into a hole in the acrylic, and a bolt was threaded through the mounting bracket and into the insert, securing it to the acrylic. The total width of the acrylic

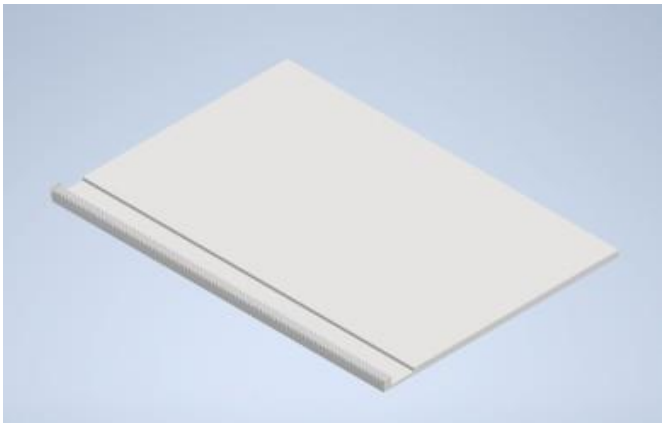


Fig. 11. **Bead Compartment False Bottom Design**

and mounting bracket is 1/4 inch, and the threaded insert is 2-56 sized thread, so 1/4 inch long 2-56 thread size screws are being used as the fasteners. For smaller components to be mounted to the chassis, such as the navigation cameras, peripheral display, and motors, screw holes will be drilled into the acrylic frame. This allowed the exact placement of these components to be finalized after the structure itself was built. This decision was made to prevent misplacement and wasting acrylic by laser cutting these smaller mounting holes in the wrong place.

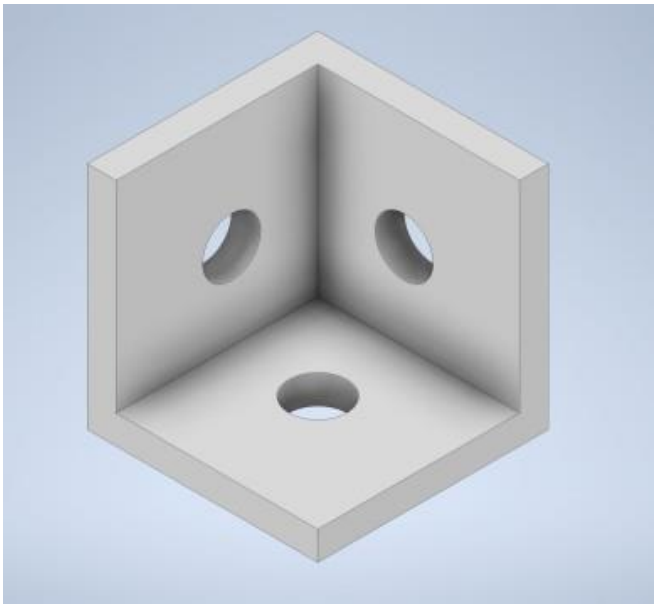


Fig. 12. **Chassis Mounting Bracket**

The most crucial design constraint of the chassis is that it must conform to the 12 cubic inch specification as outlined by the competition. Without meeting this constraint, the robot will be disqualified from the competition, rendering all other design aspects pointless. As evidenced above, this compliance was verified by constructing a 1:1 scale 3D model of the chassis and external subsystems. To assist with the more complex elements, third-party 3D models of the robotic arm

and mecanum wheels were imported and used. More detailed models and designs of these subsystems are located in their respective sections of the document. Using the CAD software's measurement tool, the chassis in its current design iteration was measured to be 11.8 x 11.576 x 11.461 inches which complies with the specification. Additionally, the upper chassis begins at a height of 3.25 inches off the ground, which allows it to freely turn above the 2 inch barricades that surround the track.

2) ***Turntable****:* During the competition, nets will be placed along either side of the track. The robot, as it is designed, will not be able to turn in place. This poses a challenge, as the shooting mechanism is not bidirectional, and the robot will not be able to turn to face the nets in either direction. Early in the design process, it was decided that the robot will utilize a turntable in order to accomplish aiming. The turntable will be responsible for turning the upper chassis (which contains the shooting mechanism) in either direction. The turntable will need to be able to support the weight of the upper chassis; a breakdown of this weight is shown in **Figure 13**.

Component	Weight (pounds)
Motors x4	1.65
All Controllers/Sensors	0.5
Shooting Mechanism	2
Wheels x4	0.5
Battery	1.5
Robotic Arm	2
Acrylic Chassis	4
Turntable	0.2
Total	12.35

Fig. 13. **Breakdown of Robot Weight**

Being that the turntable only needs to support the weight of the upper chassis, the drivetrain motors, wheels, 1/3 of the acrylic chassis, and the turntable itself can be subtracted. It is reasonable to assume that the turntable will need to support at least 7 lbs.

The turntable will be need to be able to rotate the entire upper chassis in order to aim the shooting mechanism. It will need to be able to rotate 180 degrees to fire in both directions. Based on emperical evidence collected from testing a similar turntable model made by the same manufacturer, the turntable will easily be able to achieve the desired angle of rotation.

In addition to the turntable, a servo motor is also needed to drive the rotation. Calculating the load of the servo is a matter of calculating the torque required to rotate a rectangle (the chassis volume) around its height axis. This torque is equal to the moment of inertia multiplied by the angular acceleration. The equation for moment of inertia is shown below in **Figure 14**.

This gives the moment of inertia at approximately 0.0381 kg*m^2. The desired acceleration of the servo is 2pi radians/s^2. Thus, multiplying these values gives a resultant torque of 0.2394 N*m. Converting to kg*cm gives 2.44 kg*cm, which is the minimum torque requirement of the motor.

$$I_h = 112 * 3.2 * (0.32 + 0.232)$$

Fig. 14. Moment of Inertia Equation

Using the documentation of the turntable, a servo motor with the appropriate torque output was selected. This servo requires 4.8-6 V. The design already includes step down circuitry that outputs 5 V, so the voltage requirement is met. The servo will be controlled using the Adafruit servo shield that was purchased for the robot arm. The servo shield will allow for control of the servo from the microcontroller via PWM. A wiring diagram is shown in **Figure 15**.

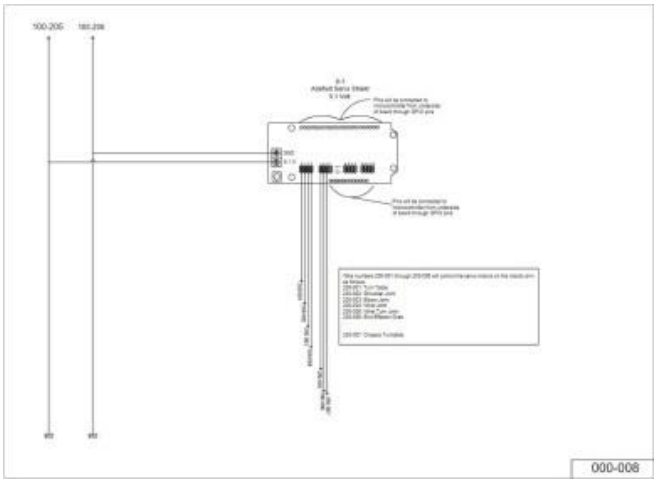


Fig. 15. **Turntable Wiring Diagram**

Figure 16 shows the CAD drawing of the turntable as well as the physical layout of the robot. The turntable's small form factor means that a custom base was designed, which will be 3D printed, to mount it in order to achieve the correct height.

3) ***Shooting mech****:* The shooting mechanism is a vital aspect of the overall design, being that it is the primary way that the team will be gaining points. It will need to be able to aerially launch beads a horizontal distance of between 5 and 6 inches and a vertical distance of between 6 and 13.5 inches. It is desirable that the beads reach their "apex" height near the middle of the diameter of the net. To accomplish this objective, the team has designed a conveyor system that will be able to launch the beads at the required distance and height.

To better understand the trajectory of the beads, a MATLAB program was created to plot the flight path of the beads with different parameters. The parameters that were allowed to be edited were the launch angle in degrees, the distance from the end of the conveyor to the net in inches, and the RPM of the motor that is driving the conveyor. Changing these three parameters and observing how they affected the flight path of the beads allowed the team members to gain a better understanding of how the conveyor should be designed. An example of the output of the program is shown in **Figure 17**.

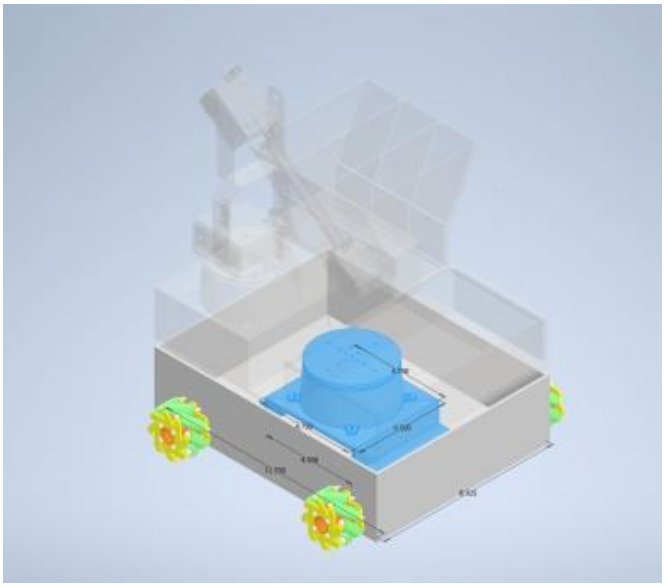


Fig. 16. **Three-Dimensional Model of Turntable**

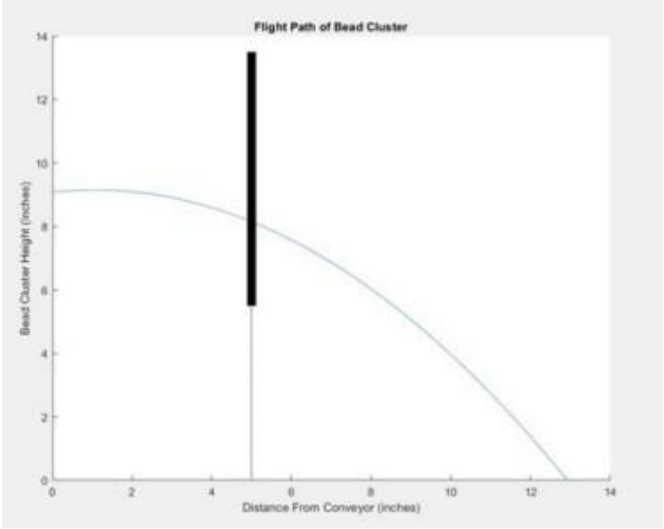


Fig. 17. **Projected Bead Trajectory**

Figure 17 is using 8 degrees for the launch angle, 700 RPM for the motor speed, and 5 inches for the distance to the net. It is clear to see that with these parameters, it is achievable to launch the beads into the net. The limiting factor in the actual design of the conveyor will be the speed of the motor.

In order to choose an appropriate motor, the potential torque load that the motor will experience needs to be approximated. There are a few different factors that affect the overall load, including the mass of the beads and the friction of the ball bearings that the conveyor is mounted to. It is not expected that the conveyor belt itself will contribute much to the load due to the fact that it is both pushing and pulling the rollers at the same time, effectively cancelling out this force. In addition, the friction of the ball bearings is expected to be negligible, as they are specifically designed to be able to rotate with minimal friction. It is anticipated that the main contributor to the load will be the mass of the bead bracelets. Being that there is no official bead bracelet being used, the mass has been estimated at around 0.822 grams per bracelet. This number was based on similar products from Amazon. The maximum number of bracelets in the competition will be 30, giving a potential total mass of 24.66 grams. The approximate length of the conveyor will be 22.86 centimeters. Using these values to approximate the load returns a value of 0.5637 kg-cm. The selected motor has a rated torque of 0.66 kg-cm, which exceeds the estimated required torque for the belt. In addition, it has a rated speed of 850 RPM, which has been proven using the MATLAB simulation to be fast enough to launch the beads into the nets. It will be coupled to the rotary shaft via a 4mm to 5mm set screw motor couple, and mounted to the chassis via a custom 3D printed bracket. The motor will be controlled through an H-bridge via PWM. This method of control was chosen because it gives the team the ability to change the speed of the motor if necessary. The team already possesses an extra H-bridge that can be used for this purpose. A circuit diagram of the motor and H-bridge is shown in **Figure 18**.

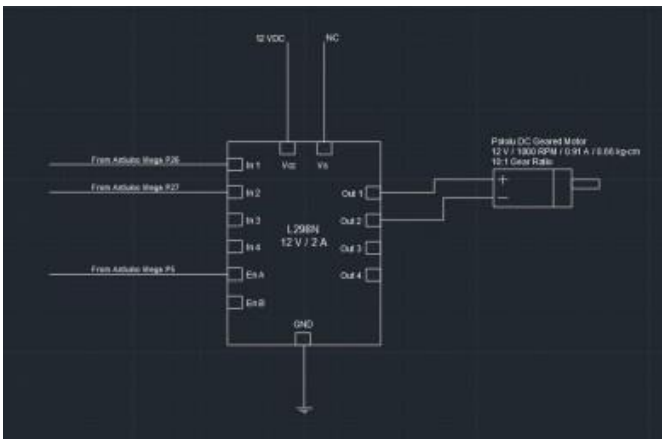


Fig. 18. **Shooting Mechanism Conveyor Circuit with the H-Bridge**

The rollers will be 3D printed, which will allow the team to control bore precision. The surface of the 3D printed material also provides a good amount of friction to turn the belt. The diameter will be 1.5 inches with a bore size of 3/16 inches to match that of the bearings and axles. The rollers will need to be attached via ball bearings to the supporting structure. To attach the rollers to the surrounding support structure, pillow block bearings will be utilized. The pillow block bearings have outward-facing mounting holes. These holes will allow for mounting as well as adjusting the angle of the conveyor. Brass wells will be inserted into slots in the acrylic in order to hold the bearings in place. Additional nuts and screws will be needed to mount the wells to the bearings. The rotary axle will pass completely through the roller bore and be mounted on each side to the bearings/motor. It will have a 3/16 inch diameter to match that of the bearing and roller. The belt material thickness was chosen to be 1/8 inch to keep the conveyor height at a minimum. Rubber has been chosen for the belt material because of its tackiness. The team also added room in the conveyor design to attach 1-inch protrusions on

the belt to prevent slip if it is an issue. The belt will be held to the rollers by its own tension. Finally, the supports of the conveyor will be made of acrylic, the same material as the rest of the chassis. The shooting mechanism belt model **Figure 19**.

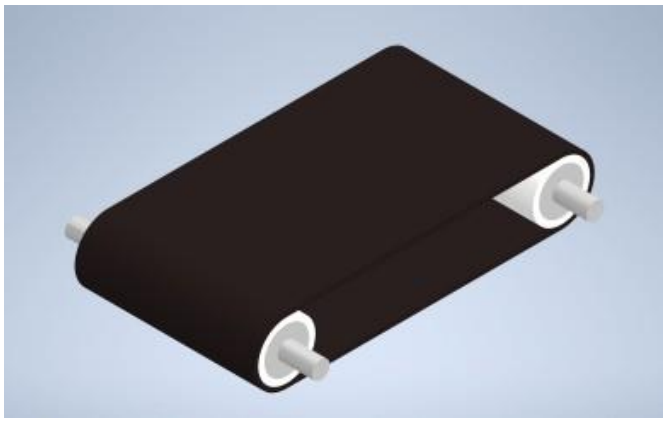


Fig. 19. **Shooting Mechanism Conveyor Model**

The shooting mechanism will also need a method to reload, as there will be three separate nets. As each new net is found, the next set of beads to be fired off will need to be dropped onto the belt. The team has designed a reloading mechanism that will be fixed atop the shooting mechanism. This mechanism will contain three compartments with a slab on the bottom that is able to be moved, and in turn allowing each set of beads to be dropped. The slab will contain a toothed rack on its side and will be moved by a motor attached to a gear. Because the slab will be 3D-printed, it is expected that the torque required to move it will be miniscule. A small servo motor will be used in this application to have precise control over the slab movement. This servo will be controlled through the Adafruit servo shield via PWM. A custom gear will be 3D printed and attached to the servo motor shaft. The wiring diagram of the Adafruit servo shield is shown in **Figure 20**. A detailed complete diagram can be seen in **Figure 21** and **Figure 22**.

*4) **Main Controller Specifications**:* The biggest specification that determined the design of the main controller was the large number of GPIO pins required. Each motor driving circuit for the drivetrain requires six pins from the main controller, and two motor driving circuits will be onboard for a total of twelve GPIO pins for the drivetrain alone. Additionally, the line following sensors, which were originally conceptualized to be connected to the navigation system, have been moved to directly connect with the main controller. The purpose of this design change was to reduce the feedback delay from the line following array to the control algorithm. However, this change increased the number of GPIO pins required of the main controller. Due to this high pin requirement, an Arduino Mega 2560 was selected for its 54 GPIO pins, 15 of which can be used to generate Pulse Width Modulation signals.

The following table in **Figure 23** outlines the subsystems that will utilize the main controller's GPIO pins. The subsystem, total number of pins, and number of PWM pins are each

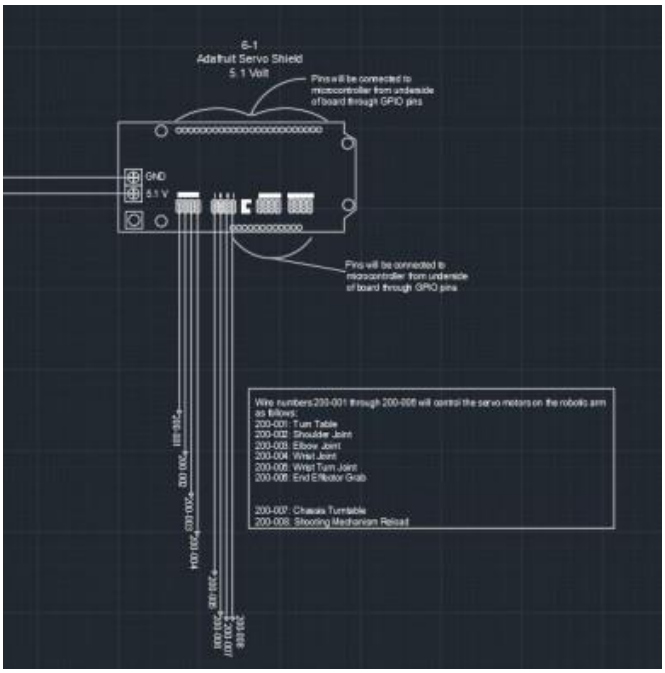


Fig. 20. **Shooting Mechanism Servo Shield Wiring Diagram**

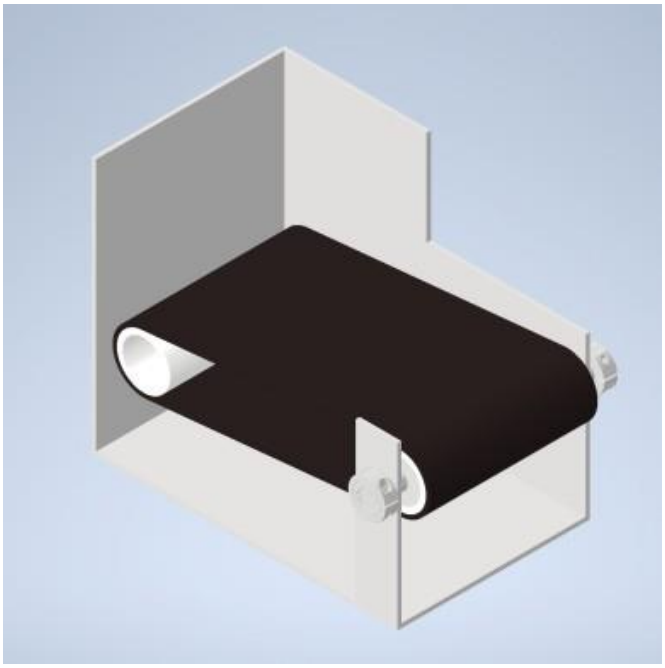


Fig. 21. **Shooting Mechanism Conveyor Model Full**

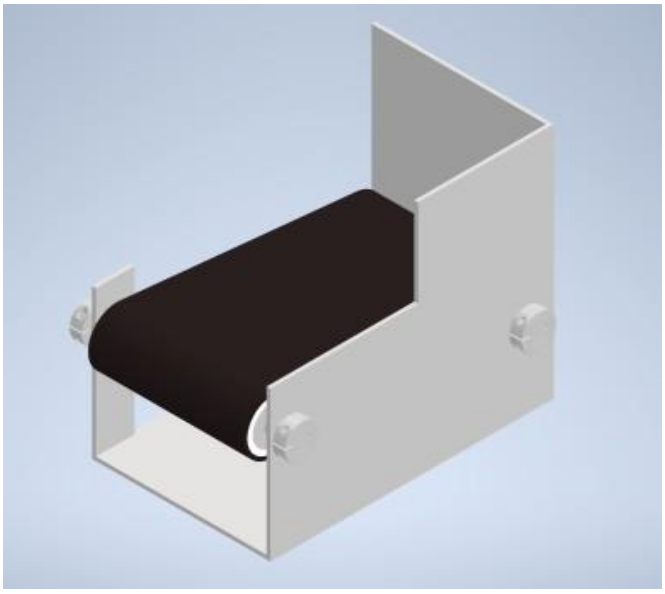


Fig. 22. **Detailed Shooting Mechanism Model Full Back**

listed. In order for this microcontroller to comply with our design, the total number of pins and number of PWM pins must not exceed 54 and 15, respectively.

Subsystem/Subroutine	Description	GPIO Pins Required	PWM Pins Required
Drivetrain	2 Motor Control Circuits	12	4
Line Following Algorithm	3 Light Arrays	24	0
Robotic Arm	Servo Control	4	4
Shooting Mechanism	Motor Control	4	0
	Totals	**44**	**8**

Fig. 23. **Main Controller GPIO Pins**

5) ***Power****:* The needed power specifications are defined by the data sheets of the selected components of each subsystem. Due to the nature of the project and its many electrical components, providing stable power that is free of noise is important to ensure proper functionality of all components. Power will be supplied to most of the components through secondary devices such as a servo shield or H-bridges.

The first step in defining the specifications of a power system is to determine the required voltage and current draw of each subsystem. Each value was obtained from the components' datasheet. The main controller, an Arduino Mega 2560, has a needed input voltage of 7 Volts. The navigation system will be running from a Raspberry Pi 4 Model B. The navigation sensors will be connected to this board, which will serve as their power supply. The Raspberry Pi requires an input of 5 Volts. The drivetrain will operate through H-Bridges that will drive the four motors. The four motors have an operating voltage of 12 Volts which will be supplied to the H-Bridge thus powering the motors. The robotic arm will function in a similar way that the motors do; however, the servo motors controlling the robotic arm will be driven and powered by the Adafruit servo shield. The servo shield requires an input voltage ranging from 2.3-5.5 Volts. Finally, the turntable servo motor will be run by a similar servo motor to the turn table. For this reason, the turntable can be run through the servo shield. The shooting mechanism will be run by a servo motor operating at 5.1 Volts. A maximum voltage of 12 Volts is required. These values are can be seen in **Figure 24**.

Component	Voltage (V)	Amperage (A)
Raspberry Pi	5.1	3
Motors x4	12	2.4 (0.6)
Adafruit Shield	5.1	0.25
Arduino MEGA	7	0.5
Arduino UNO	7	0.3
Turntable Servo	5.1	1
Shooting Mech Servo	5.1	1

Fig. 24. **Voltage and Amperage Values per Data Sheets**

Finding the needed amperage is not as simple as the voltage. Finding the amperage requires utilizing **Figure 24** and the MATLAB code found in Appendix A. The MATLAB code outputs a total current of 4.8287 Amps, which means that everything running at once will require a 12 Volt battery to output at least 4.8287 Amps. For this reason, the team selected the LiFePO4 12V 6.6Ah battery. Using this battery required the team to find how long one battery charge will last, the C value. 6.6Ah means that the battery will give one hour of 6.6 Amps, giving a C value of 1. Using this ratio, the team found that the current ratio (referenced in Appendix A) gives a value of 1.3932. Changing this number to minutes gives the team 82.0096 minutes or 27.337 3-minute competition iterations. The calculation is also assuming that every component will be in operation for every one of the 82.0096 minutes; however, this is not the expected use. Because of this, the battery is expected to last longer than the calculated value.

The 6.6Ah surpasses the maximum possible current draw (4.8287 A) of all the components running; however, a problem arises due to the battery not being able to give a constant 12 Volts. For this reason, the team will utilize a buck-boost converter. In theory, a buck-boost converter can take a ranged input voltage and transform the voltage to a certain output voltage. The converter can take an input voltage from 9-36 Volts and transform the voltage to 12 Volts with 10 Amps.

One concern is the fact that noise may come off the converters. The data sheet verifies that the noise from the component will be no bigger than 115-200mV. For this reason, the team configured a low pass filter. The team simulated the potential noise to verify the design of the filter. The LTSpice Simulation can be seen in **Figure 25**.

By implementing this filter, the battery and converter will be able to safely supply the components with the needed input voltage. **Figure 26** shows that multiple components need the same input voltage to operate. For instance, one set of

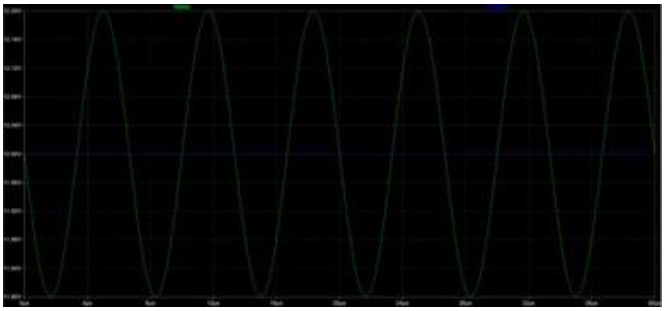


Fig. 25. **LTSpice Schematic with Noise Simulation**

components needs 12 Volts, while other components need 7 and 5 Volts, respectively. This brings new constraints to the power circuit. For one, the initial voltage needs to be converted into two separate lower voltages, and said voltages need to go to multiple components.

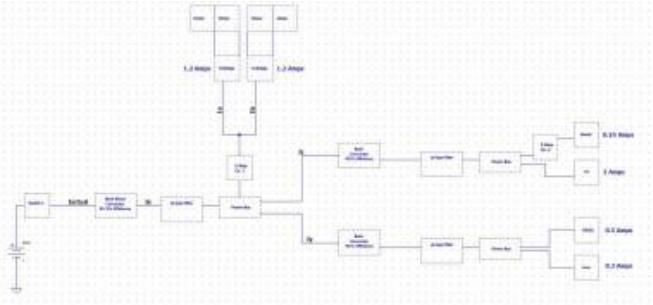


Fig. 26. **Full Power System Schematic**

The data sheet verifies that the buck converter can take an input range of 3-40 Volts and transform it into a set range of 1.5-35 Volts. The two ranges fit the needed specifications set by the components described before. The team will utilize the regulated output voltage knob to set the needed transformation Along with transforming the voltage, the second constraint will be met by utilizing power buses. A power bus can take the input voltage and distribute that voltage to other components. **Figure 27** depicts the circuit diagram for the selected power bus which can operate at up to 24 Volts.

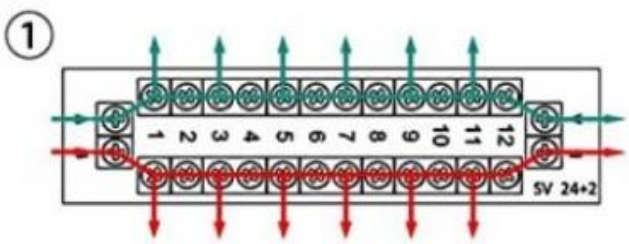


Fig. 27. **Power Bus Circuit Diagram**

The team needs a combination of the two previous ideas so that the power circuit will be able to supply the needed voltages to each component. A combination of all the components previously described will build the power circuit. The connections described among all the components can be seen in the attached CAD document in Appendix B.

The competition rules state that the robot must have a clear on/off switch, and strongly recommend adding an e-stop into the design. For this reason, the team has picked out a switch and an emergency stop that will meet these requirements. Two switches will be utilized in the circuit. One switch will be the interim between the battery and the boost converter while the second will be an input that the microcontroller will watch for to start the competition. The data sheet clarifies that the switch can withstand 10 Amps which is above what the team needs.

The emergency stop's data sheet justifies that the channels can withstand 10 Amps, which is well above the team's currents within the circuit. The emergency stop was picked due to the two-channel capability. The team will utilize this feature so that the 5 Volts going into the Adafruit servo shield and the 12 Volts going into the H-Bridges will have to clear the emergency stop before powering the robot to move. Likewise, pushing the emergency stop will stop the movement from the robotic arm and dc motors of the wheels, but leave the microcontrollers still running. Doing this, will let the robot handler stop the robot from causing more damage if havoc ensues.

6) ***Navigation****:* The navigation system has undergone several changes since the initial design phase, now consisting of a Raspberry Pi 4, two cameras, and four IR transmitter/receiver pairs. A Raspberry Pi allows for the use of Robot Operating System (ROS), which is optimized for robotics, and computer vision techniques, as well as having a large amount of RAM. The Pi will run Ubuntu 20.04 as the main operating system and use the ROS Noetic distribution to handle communication between the main and navigation controllers. Communication will take place via serial pins using the rosserial library. ROS Noetic is the most recent distribution and has long-term support from the developers.

This system is vital to performance in the competition. It is vital to identifying the beads and nets, which is the main way the team will be earning points in the competition. The system is designed to recognize the nets, with the camera, and finely adjust, using the sensors and camera, the robot's position on the track to align the shooting mechanism.

Instead of having two IR arrays per side of the robot, there will now only be two sensors per side, both at the same height, tentatively aligned under the shooting mechanism. This allows for fine tuning adjustments before shooting the beads. The selected sensor, Sharp GP2Y0A21YK0F IR Distance Sensor has a range of 4-32 inches. Based on provided images of the competition track setup, the nets appear to be less than 18 inches from the track. Therefore, this is expected to be enough range to reach the nets and provide adequate distance readings. If competition specifications are changed, the sensor will be revisited to determine if it still satisfies the required functionality. This sensor includes both the transmitter and receiver, which will give more accurate readings without exhaustive testing. This sensor requires 3 GPIO pins: output, power, and ground. The Pi has enough pins to support 4 sensors (12 GPIO pins), but, if necessary, measures can be

taken to consolidate the power and ground wires.

A new component to the system will be the Arducam USB Camera V2 Module. Initially, the system was designed to use two arrays of sensors along the left and right sides of the lower and upper chassis. As more research was completed, it was determined that this idea was not going to give accurate information for positioning, as well as major difficulty differentiating between the nets and power poles. After looking into several ideas, the design now incorporates computer vision as the primary source for net identification. This specific model was chosen for several reasons: small size, adequate focus range and resolution, field of view (FOV), and USB operation. Given the size constraints of the robot, it was necessary to find a camera that took up a minimal amount of space.

This camera is only 38mmx38mm, in addition to the weight being negligible. It has a resolution of 8MP 3264H x 2448V, and per the device specification sheet, the focus ranges from 200 mm to infinity. As the nets will be more than 200 mm and less than three feet from the track, this focal length is sufficient. The camera has a FOV of 62.2 degrees horizontal and 48.8 degrees vertical. The FOV should be adequate as the camera is intended to be directly facing the nets. The camera is also going to be operated via the on-board USB ports. This was done because the Pi has only one connection for a camera, and would require additional hardware to include two cameras. There are four on-board USB ports, which leaves two remaining for any additional hardware that may be necessary in the future. The cameras will be mounted on the left and right sides of the lower chassis, as shown in ~\ref{nav_cam_3d}.

A system schematic, illustrated in **Figure 28**, has been created using AutoCAD Electrical showing the hardware connections. It shows the eighteen needed GPIO pins for the IR sensors, two USB port connections for the cameras, power supply, and serial comminication between the microcontrollera, as well as a full capture of the Raspberry Pi.

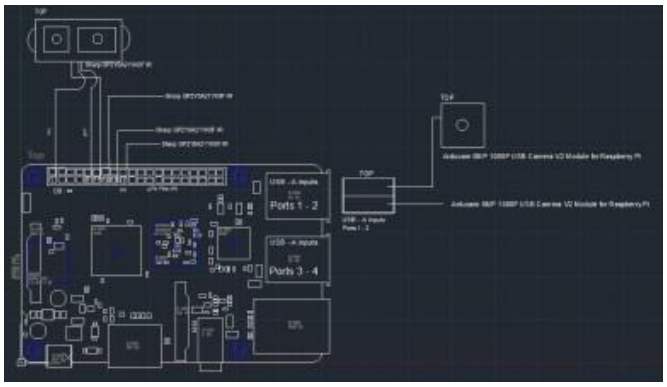


Fig. 28. **Full Navigation System Schematic**

7) ***Drive train****:* The drivetrain subsystem of the project has been designed in order to maximize maneuverability and increase the effectiveness of other subsystems within the robot. This subsystem will be comprised of three primary components: DC motors, H-bridge motor drivers, and wheels.

a) Wheels: Previously, it was intended that the design would utilize standard friction wheels. However, after more consideration, it was finally decided that mecanum wheels will be used in the final design. The primary advantage of using these wheels is that they will allow the robot to traverse in any direction, including horizontally. It is advantageous for the robot to move horizontally because it will make obtaining and firing beads easier and more effective. Specifically, moving horizontally will reduce the distance from the robot to the trees and nets.

When researching different wheels, it is important to consider weight capacity. The robot will have a certain mass, and the wheels will need to be able to support it among the four of them. The team has approximated that the robot will weigh about 12 lbs. The weight breakdown of the robot can be seen in **Figure 9**.

After researching wheels, it was decided that the team will utilize the same type of mecanum wheels from a previous SECON robot iteration. The wheels that will be used have a radius of 26mm, and can together support up to 30 lbs, which far exceeds the anticipated need of this application.

b) Motors: The next component needed in the design of the drivetrain subsystem is the DC motor model. The primary specifications of the motor include the maximum speed of the entire robot as well as its acceleration to reach this speed. Earlier in the semester, the desired speed of the robot was set at 2 ft/s, but this has since been cut down. The team believes that 1.5 ft/s is a reasonable speed for the robot to traverse the track. Additionally, the original 2 ft/s parameter would have been harder to accomplish when considering that the robot will have to stay centered on the white line on the track. Reducing the speed will allow for more accurate navigation capabilities. The maximum speed that the robot is able to achieve will ultimately depend on the accuracy and speed of the navigation control system that the team is able to produce.

In addition to maximum speed, robot acceleration is also a parameter that affects the specific motor hardware that the project utilizes. It is desirable that the robot reach its maximum speed in one second. Thus, with the top speed set at 1.5 ft/s, an acceleration of 1.5 ft/s^2 is required. Similar to the wheels, weight is also a crucial parameter that will affect the specific motor hardware used. The weight of the robot directly affects the total load seen by the motors, and thus a motor is needed that will be able to move this weight at the speed required.

Maximum incline is something that should be considered for motor sizing as well. In this specific application, the maximum incline is expected to be relatively small. The environment that the robot will be competing in should be a fairly level board, and so the team estimated that the maximum incline that the motors will have to overcome would be about 15 degrees.

Wheel radius is another factor that will change the requirements of the motors. Increasing the wheel radius will in turn increase the torque requirements of the motors, and vice versa. Conversely, increasing the wheel radius will decrease the number of revolutions per minute (rpm) that the motors will have to turn to turn in order to achieve a given traversal speed. The wheel radius in this application was mentioned earlier, and is 26 mm. This smaller radius will mean that the motors used in this project will need to have a higher rated rpm in order to achieve the top speed of 1.5 ft/s.

Finally, the last parameter that will affect the rating of the motor is the number of drive motors used. As one would expect, increasing the number of motors utilized will decrease the torque requirements of each motor. This application will require that four motors be used to independently drive each mecanum wheel, allowing for the desired lateral motion.

The effects that all of the aforementioned parameters will have on the motor requirements can be seen below in **Figure 29:**

$$T = \frac{100}{e} * \frac{(\alpha + g * \sin(\theta)) * M * R}{N}$$

Fig. 29. Motor Requirement Equation

It is important to note that the International System of Units (SI) must be used in the equation above. In the equation, alpha represents the desired acceleration of the robot in m/s. When converting from the previously mentioned desired acceleration of 1.5 ft/s, the acceleration becomes 0.452 m/s. The symbol g represents the acceleration due to gravity and is constant. Theta represents the maximum incline, and as mentioned earlier is set to 15 degrees. M represents the mass of the robot in kilograms. When converting the anticipated robot weight of 12.35 lbs, the result becomes 5.60 kg. The wheel radius is already in SI units, and is 0.026 m. N represents the number of motors, and is set to 4.

It is worth noting that an addition variable, e, is used in **Figure 29**. This variable represents the total efficiency of the system, including power lost to heat, friction of the gears, etc. When considering the entire system, this parameter can be difficult to estimate. It is desirable that the system be as efficient as possible, but the team believes that using a more pragmatic value is beneficial. Many websites use a standard value of 65\% for e, however it was decided that a value of 50\% would be used in this application in order to allow for more inefficiencies.

When using Equation 2 with the aforementioned parameters, the resultant required torque is calculated as 0.19270 Nm. When converting to kgf-cm, the result is 2.22 kgf-cm. Additionally, the required rpm of the motors in order to achieve the desired top speed will be 168 rpm. Using these parameters, the team was able to find a motor model that matches quite closely to the calculated requirements. The motor is a 12 volt DC gear motor with a gear reduction ratio of the motor is 50:1. The rated operating points of the motor can be seen below in **Figure 30**.

As can be seen from Table above, the rated operating point of this motor matches very closely to the requirements of this application. Further, this motor slightly exceeds the requirements of this project, which is advantageous because it will allow for some tolerance if the robot weighs slightly more than calculated.

The output shaft of the motor is 6 mm in diameter. However, the wheels that this project will utilize have a hub diameter

Maximum Efficiency At 12 V	**51%**
Speed At Maximum Efficiency	**180 RPM**
Torque At Maximum Efficiency	**2.2 kg-cm**
Current At Maximum Efficiency	**0.66 A**
Power at Maximum Efficiency	**4 Watts**

Fig. 30. **Rated Operating Points of the Motor**

of 3 mm. The team will utilize the machine shop in order to further drill the hubs of the wheels to match the required 6 mm diameter. It is worth noting that the back up wheels from the previous robot have already been drilled to 6 mm.

c) H-Bridge Circuitry: In order to get the best performance out of the drivetrain subsystem, it is necessary to utilize motor driving circuitry that will allow the robot to move in different directions and at variable speeds. The type of circuit that will be utilized in this application is known as an H-bridge.

When considering the H-bridge model that would be utilized, it was important to ensure that the chosen circuitry matched the power requirements of the motors. The drivetrain motors used in this application are rated for 0.66 A at maximum efficiency, or the rated operating point. As stated earlier, the total load of the robot is anticipated to be less than what the motors are rated for. However, in order to provide some room for error, an additional 15\% has been added to the current draw per motor, which places the current draw for each motor at 0.76 A. In addition, the motors being rated for 12 V means that the H-bridges will also have to be able to support that voltage.

Additionally, the H-bridge needed to be PWM compatible in order to drive the motors at variable speeds. PWM enables speed control by varying the average voltage level that the connected motors see.

The specific model that will be used is the L298N motor driving board and is developed by Qunqi. The current rating for each channel of the board is 2 A, which greatly exceeds the worst-case scenario of 0.76 A. The voltage rating of this board is 5 V to 35 V, so it will be able to support the 12 V needed by each of the motors. In addition, this motor driver board has input pins to change the polarity of the voltage to the load, which will allow for direction control of the motors. Finally, this particular L298N board is PWM compatible, and in fact is very commonly used in Arduino projects that involve motor control.

d) Drivetrain Schematic: All of the components that will be required to construct the drivetrain have been compiled, and the interconnects of each component are shown in the design artifact in **Figure 31**.

Figure 31 is a visual demonstration of the interconnectivity of the various drivetrain components. On the left are the individual signals that the L298N will be receiving from the main microcontroller. Inputs one through four will control the polarity of each motor, while inputs ENA and ENB will be the PWM inputs to control the speed. The input

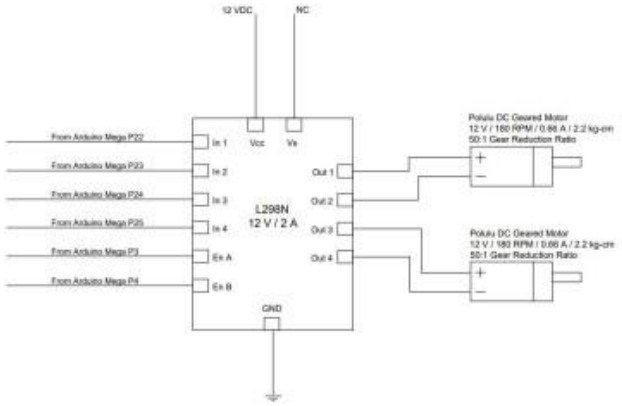


Fig. 31. **Drivetrain CAD Model**

V\textsubscript{CC} is the drive voltage which will need to come from a 12 V regulator. Outputs one through two will control the first motor, while outputs three and four will control the second motor. It's worth noting that two identical versions of this schematic will be implemented in the final design, the only difference being the pins coming from the microcontroller in both cases. Each motor will be physically mounted to the lower chassis via screws through the chassis acrylic and threaded into the front face of the motor. The motors are manufactured with M3 threaded holes for this specific purpose. The holes in the acrylic will be drilled as previously discussed.

8) Robotic Arm: The team plans to utilize the robotic arm to grab the beads off the tree and place said beads into a magazine to be loaded into the shooting mechanism. One reason for choosing the Lynxmotion AL5D is for the 5 degrees of freedom. There is one servo motor for the shoulder joint, one for the elbow joint, one for the wrist, one for the wrist turning motion and one for the gripping mechanism (end effector). The shoulder, elbow, wrist and turning motion can all move 180 degrees. The end effector has the capability to open to 1.25 inches. This opening length is perfect for grabbing hanging beads off the tree; however, other aspects of the stock end effector are not ideal for the given task. This includes the short length of the gripping mechanism. Because of this, an extension to the end effector may be 3D printed if trial runs show that its current length is unsatisfactory. It is common engineering practice to design a custom end effector to be used with a third-party robotic arm. The basic robotic arm extension length can be seen below. **Figure 32** shows the minimum height the chassis needs to be for the robot arm to reach the 21.5 inch tree branch.

Height of Chassis Minimum without extension	3"
Height of Chassis Minimum with extension	1.6750"

Fig. 32. **Robot Arm Extension Lengths**

With these specifications the team solved for the height needed on the chassis to be able to reach the tree branch seen in **Figure 33**.

Height without extension add on	18.5"
Height with extension add on	19.8250"

Fig. 33. **Minimum Height Required of Chassis for Arm to Reach**

The distance from the tree to the robotic arm was varied to verify how far the robot can be and still reach the tree limb. The below graphs show the lengths that can be achieved with the Lynxmotion AL5D Robotic Arm. The values across the top show the height at which the base of the robotic arm is sitting on the gameboard. The values down the side of the graph show the distance away the base of the arm is from the tree. Both values can be seen in blue. Number values in red show lengths that the robot cannot reach from the given height and distance. Number values seen in yellow appear 0.25 inches away from the max arm reach distance. Number values seen in green are distances that the robot can reach from the given height and distance.

Robot Chassis Height	1	2	3	4	5	6	6.5	7	7.5
Inches Away From Tree	**********No Extension Arm Length = 16.5**********								
3	20.7183	19.74294	18.7417	17.7553	16.7705	15.7877	15.2971	14.8071	14.3178
3.25	20.756	19.769	18.7833	17.7992	16.817	15.8371	15.348	14.8598	14.3723
3.5	20.7966	19.8116	18.8282	17.8466	16.8671	15.8902	15.4029	14.9164	14.4309
3.75	20.8402	19.8573	18.8762	17.8973	16.9208	15.9472	15.4616	14.9771	14.4935
4	20.8866	19.906	18.9275	17.9513	16.9779	16.0078	15.5242	15.0416	14.5602
4.25	20.9359	19.9578	18.9819	18.0087	17.0386	16.0721	15.5905	15.11	14.6309
4.5	20.9881	20.0125	19.0394	18.0693	17.1026	16.14	15.6605	15.1822	14.7054
4.75	21.0431	20.0702	19.1001	18.1332	17.1701	16.2115	15.7341	15.2582	14.7839
5	21.1009	20.1308	19.138	18.2003	17.2409	16.2865	15.8114	15.3379	14.8661
5.25	21.1616	20.1944	19.2305	18.2705	17.3151	16.365	15.8922	15.4212	14.952
5.5	21.225	20.2608	19.3003	18.3439	17.3925	16.4469	15.9765	15.5081	15.0416
5.75	21.2911	20.3301	19.373	18.4204	17.4732	16.5322	16.0643	15.5985	15.1348
6	21.36	20.4022	19.4487	18.5	17.557	16.6208	16.1555	15.6924	15.2315
6.25	21.4316	20.4771	19.5272	18.5826	17.644	16.7126	16.25	15.7896	15.3317
6.5	21.5058	20.5548	19.6087	18.6682	17.7341	16.8077	16.3478	15.8902	15.4353
6.75	21.5827	20.6352	19.693	18.7567	17.8273	16.906	16.4488	15.9941	15.5423
7					17.9234	17.0074	16.5529	16.1012	15.6525
7.25						17.1118	16.6602	16.2115	15.7659
7.5						17.2192	16.7705	16.3248	15.8824

Fig. 34. **Arm Length No Extension Calculations**

Robot Chassis Height	1	2	3	4	5	6	6.5	7	7.5
Inches Away From Tree	**********Extension Added Arm Length = 17.825**********								
3	20.7183	19.74294	18.7417	17.7553	16.7705	15.7877	15.2971	14.8071	14.3178
3.25	20.756	19.769	18.7833	17.7992	16.817	15.8371	15.348	14.8598	14.3723
3.5	20.7966	19.8116	18.8282	17.8466	16.8671	15.8902	15.4029	14.9164	14.4309
3.75	20.8402	19.8573	18.8762	17.8973	16.9208	15.9472	15.4616	14.9771	14.4935
4	20.8866	19.906	18.9275	17.9513	16.9779	16.0078	15.5242	15.0416	14.5602
4.25	20.9359	19.9578	18.9819	18.0087	17.0386	16.0721	15.5905	15.11	14.6309
4.5	20.9881	20.0125	19.0394	18.0693	17.1026	16.14	15.6605	15.1822	14.7054
4.75	21.0431	20.0702	19.1001	18.1332	17.1701	16.2115	15.7341	15.2582	14.7839
5	21.1009	20.1308	19.138	18.2003	17.2409	16.2865	15.8114	15.3379	14.8661
5.25	21.1616	20.1944	19.2305	18.2705	17.3151	16.365	15.8922	15.4212	14.952
5.5	21.225	20.2608	19.3003	18.3439	17.3925	16.4469	15.9765	15.5081	15.0416
5.75	21.2911	20.3301	19.373	18.4204	17.4732	16.5322	16.0643	15.5985	15.1348
6	21.36	20.4022	19.4487	18.5	17.557	16.6208	16.1555	15.6924	15.2315
6.25	21.4316	20.4771	19.5272	18.5826	17.644	16.7126	16.25	15.7896	15.3317
6.5	21.5058	20.5548	19.6087	18.6682	17.7341	16.8077	16.3478	15.8902	15.4353
6.75	21.5827	20.6352	19.693	18.7567	17.8273	16.906	16.4488	15.9941	15.5423
7					17.9234	17.0074	16.5529	16.1012	15.6525
7.25						17.1118	16.6602	16.2115	15.7659
7.5						17.2192	16.7705	16.3248	15.8824

Fig. 35. **Arm Length Extension Calculations**

There are two big takeaways from the **Figure 34** and **Figure 35**. For one, the robot arm becomes more versatile when the extension is added. For instance, the robotic arm could only be 5.5 inches away from the tree; however, the extension to the arm allows the robot to be 7.5 inches away from the tree. The next big takeaway is that the robotic arm can accomplish the needed task of grabbing the beads from the tree. The

Lynxmotion AL5D also gives the team options about chassis height placement. The most versatile placement is having the arm at 7.5 inches above the chassis. To do this, the team will have to have the robot start in a resting position that points down within the robot. The team will still be able to fully utilize the robotic arm at 6.5 inches if the chassis design does not permit the 7.5 inch starting position. The exact placement of the robotic arm will be seen in the CAD model presented with the chassis.

The next aspect of the arm will include the electrical components. Each servo motor has three wires. Black for ground, red for power (4.8-6V) and yellow for the PWM signal. The best way to control the servo motors will be through the utilization of the Adafruit 16-Channel 12-bit PWM/Servo Shield – 12C interface. The board will easily plug into our Arduino MEGA based on the design. Through this, the team will be able to control each of the servo motors while also providing reverse polarity protection. The servo motor will be connected to the Adafruit shield seen in **Figure 36** and **Figure 37**. The images are better represented in the CAD package pages 200-006 and 200-201 in Appendix B.

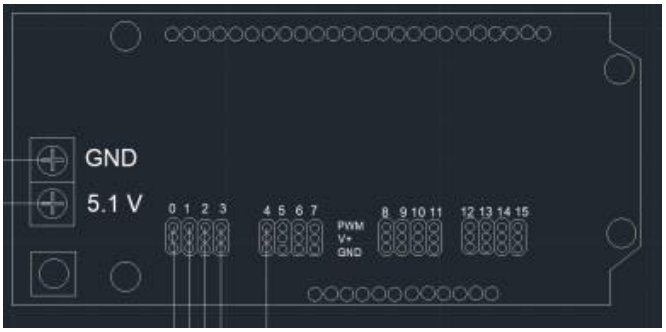


Fig. 36. **Servo Shield CAD Model**

The team will use the first five three-set pins (0,1,2,3,4,5) to control the robotic arm. The board itself will be powered from the 5.1 V power bus (2-2). The board will use the outer rim pins, located on the top and bottom of the board, to plug into the Arduino MEGA. The MEGA can be seen in **Figure 38**. By doing this, the team will be able to give the desired PWM signals to the board. The PWM signals will need to be communicated to the board in durations that last between 0.5 ms to 2.5 ms followed by a delay between 20 ms to 30 ms. The servo motors then translate the signal through onboard electronics. The pulse will need to be repeated for the servo to hold the desired position. For these reasons, the Adafruit component will be the best option to translate these signals from the microcontroller.

9) ***Peripheral System****:* The team can be rewarded points for adding a different variety of decorations to the exterior of the robot. Additionally, the team has recruited the Tennessee Tech IEEE chapter in designing a peripheral system that will help maximize the awarded points. The peripheral system was designed to play audio over a speaker, integrate a moving display onto the existing turntable, and include a blinking LED analog circuit. The LED Circuit and Arduino will be powered by the 12 Volt and 7 Volt power busses, respectively. An audio

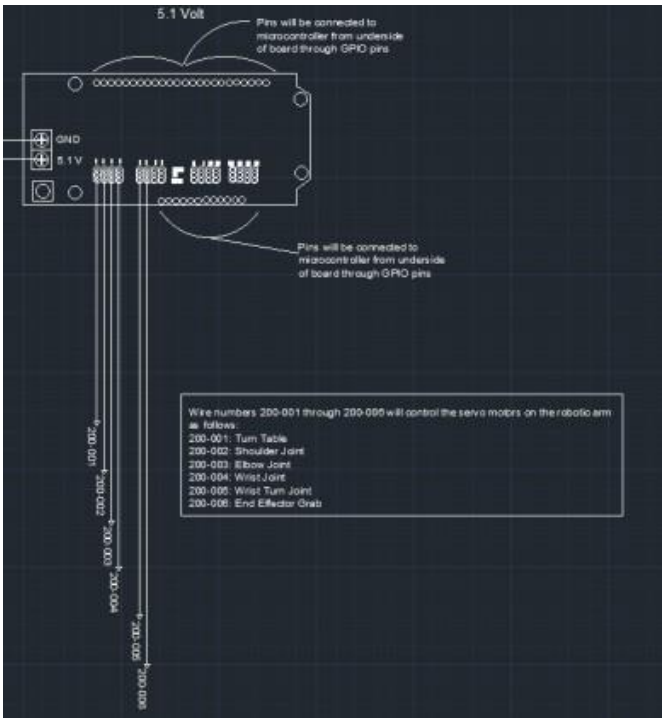


Fig. 37. **Robotic Arm Servo Connection**

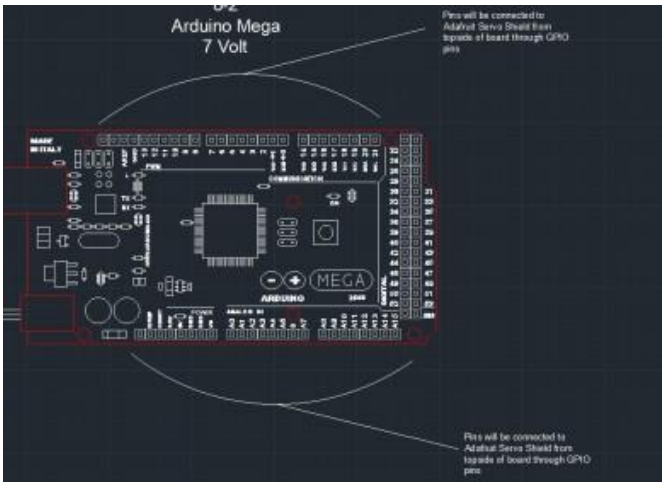


Fig. 38. **Arduino Mega GPIO Pin Representation**

shield was chosen which fits on top of an Arduino Uno, which was chosen for its compact size. The shield is equipped with an SD card reader and audio amplifier allowing the Uno to read and play sound data using existing Arduino Libraries. A TFT LCD display was chosen for the moving display because it allows for more detailed pictures to be displayed through bitmap imagery. The front of the shooting mechanism housing will be cut to allow the screen to be pressed through while obscuring the additional electronics. The screen will be held in place with pre-drilled mounting screws similar to the navigation cameras. A detailed illustration of this is shown in **Figure 39**. Finally, the NPN transistor circuit seen in **Figure 40** allows a grouping of LEDs to blink in a “chase” pattern.

This circuit was simulated in LTSpice in order to determine the current draw and blinking frequencies, as well as to verify that the circuit would create the necessary voltage drop across the diodes to make them blink. Simulation results are displayed in **Figures 41** through **Figure 43**. An analog circuit was chosen to drive the LEDs instead of utilizing the Arduino in order to reduce the use of GPIO pins and provide the IEEE members with a variety of design experiences.

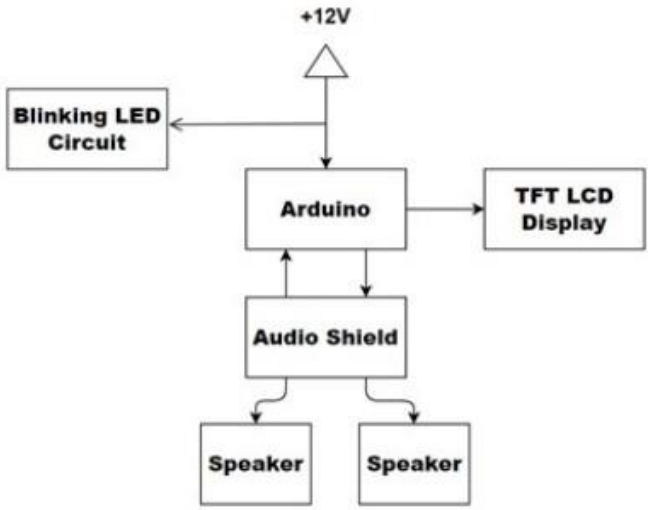


Fig. 39. **Block Diagram of Peripheral System**

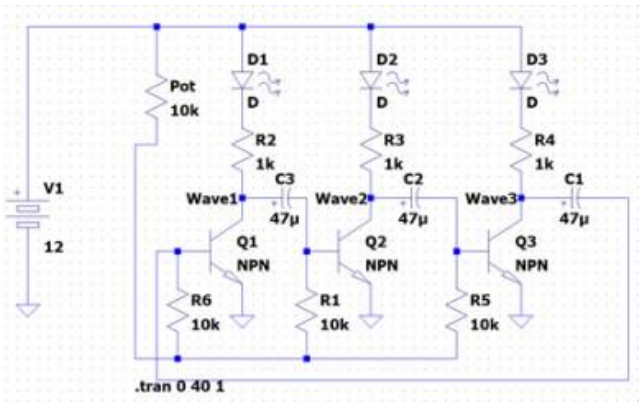


Fig. 40. **Circuit Diagram of Peripheral System**

V. Results and Discussion

To first compare our final product to our original metrics of success, the metrics themselves must be revisited. The goals can be seen in **Figure 3**. The first metric, the Size Specification, is confirmed by simply measuring each dimension of the robot. The robot measures 11 7/8 " Long x 11 1/2" Wide x 11 3 /4" Tall , so it conforms to this spec.

To assess the success of the design as compared to the other metrics, full trial runs of the competition were simulated using a 1:1 replica of the game board. Different configurations (# of beads per tree, # of nets available, placement of marshmallow) were used to assess the performance of the robot through different game conditions. In total, 29 runs were attempted and completed to various degrees. The raw run data is shown in the Appendix.

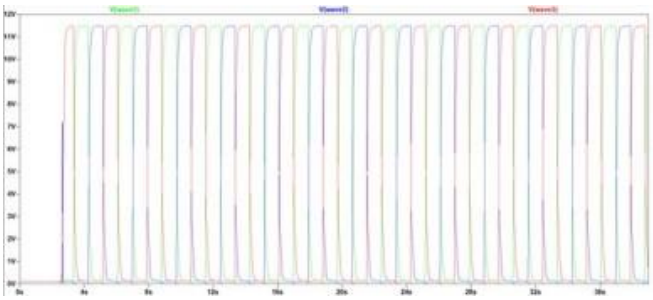


Fig. 41. **LED Circuit Voltage Verification**

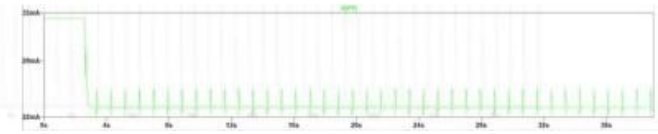


Fig. 42. **LED Circuit Current Simulation 1**

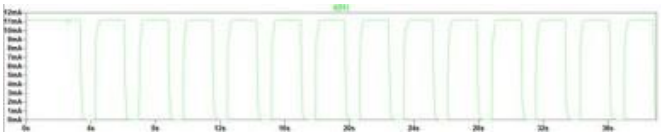


Fig. 43. **LED Circuit Current Simulation 2**

A. Battery Life- Target 6 Amp Hours

To test the battery performance and verify that it can support our system for an hour at 6 Amps of Current Draw (The expected Max Draw of the system) the battery voltage was measured before each run. A graph of the battery voltage after each run is shown in **Figure 44**.

To prevent battery damage, the team determined that a measured battery voltage below 13 Volts indicated that the battery should be charged. The data above shows that this threshold was never crossed after the 29 Runs were conducted. At an average run time of 2:28 and an additional average of 0:45 for the Pi to Boot (3:13 of total time per run) over 29 runs conducted, the total powered on time of the system during testing was 29 X 3:13 or 1:33:00. After over 1.5 hours of runtime, the battery was still maintaining a voltage level of ~13.12 Volts and was still above the charging requirement. It can be concluded that the power system meets its battery life requirement.

B. Bead Picking – Targets: 90 % Picked in <20 Seconds

To measure the effectiveness of the robotic arm and its bead picking routine, the time of each pick, number of beads picked, number of beads dropped on the roadway, and number of beads

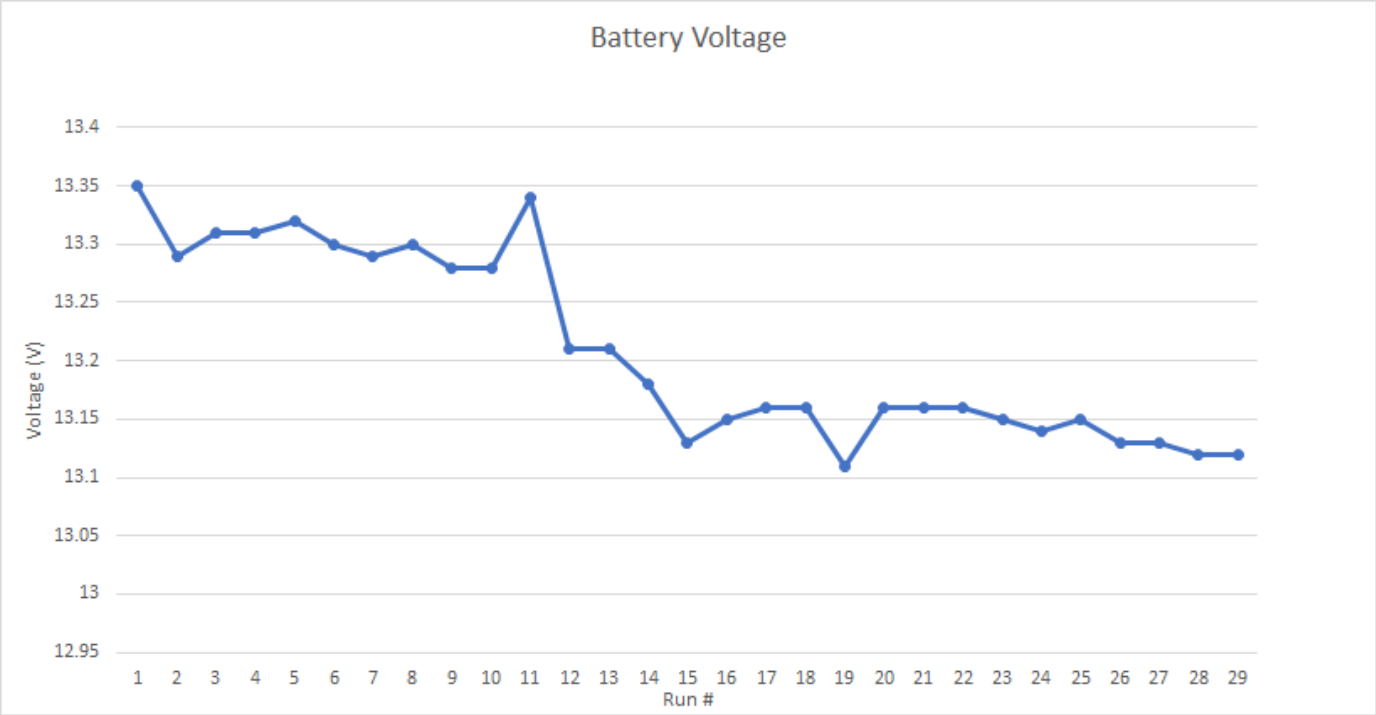


Fig. 44. **Battery Voltage Graph of 29 Trials over 1.5 Hours**

available were recorded throughout each run. This data was split between each tree location, as each location was given its own arm routine to run. The pick % was also combined among both grabs per run. Graphs for each of these are shown below.

The data shows that the design fails to meet the original goal of a 90% pick rate. Additionally, the percentage tends to decrease as the number of beads on the tree increase. This demonstrates that the current design has an upper limit of beads that it can successfully pick without dropping a substantial amount onto the roadway. Also, these beads dropped onto the roadway tended to cause the robot to get stuck and fail to complete a full run of the course. These issues with bead picking can be attributed to two factors, the end effector and a misaligned 1st grab on tree #2. The end effector should have been longer and had more padding on the grippers to prevent them from not fully closing if two beads were grabbed side by side. This issue was fairly common, and it would cause all of the other beads on that grab to be missed. The issue with the first grab at tree #2 was originally a misalignment issue that caused the arm to completely miss the beads. This meant that when the second grab did successfully reach the beads, there would be too many for a single grab, resulting in beads either being left on the tree or dropped onto the road. During the experimentation, the misalignment issue was adjusted and corrected. However, there were still issues with how this grab was placing the beads into the magazine. Instead of missing the beads altogether, this grab was now dropping some of them behind the robot instead of into it. Each time this happened, the robot was guaranteed to become stuck when attempting to return to the start of the track.

These issues with the bead grabbing system are the cause of many of the other performance issues of the robot. Instances of completely missed shots, getting stuck, and 0% of beads being grabbed from tree #2 were almost always caused by beads being dropped on the road. Either the robot would become stuck on these beads, or the Ultrasonic Sensors would mistake the beads on the road for a wall, causing the robot to stop short of its target.

Regarding the bead grab time, this was almost double the original goal of 20 seconds. Each bead pick was consistently 38 seconds with small deviations accounted for by imprecise timing. This is because of the additional grab at each tree. Originally only a single grab was anticipated, so an additional grab would be expected to double the grab time. However, other improvements to the robots design, primarily the shooting time and traversal time, allowed these longer grabs to happen without exceeding the 3 minute time limit.

C. Shooting – Targets: 95% Accuracy and < 20 Seconds

To measure the performance of the shooting mechanism, the number of beads fired, successful hits, and beads shot onto the roadway were tracked on each trial run. To match the specifications of the competition, beads that were not fully in the net, i.e. hanging halfway out, were counted as misses. Also, the disparity between the number of beads collected and beads fired is due to the fact that some beads would be pulled off the tree and dropped into the robot chassis instead of the magazine.

Similar to the bead picking, shot accuracy failed to meet the original measure of success. However, the instances of 0% accuracy shown on the chart above can mostly be attributed to the robot failing to reach a net due to being stuck, or

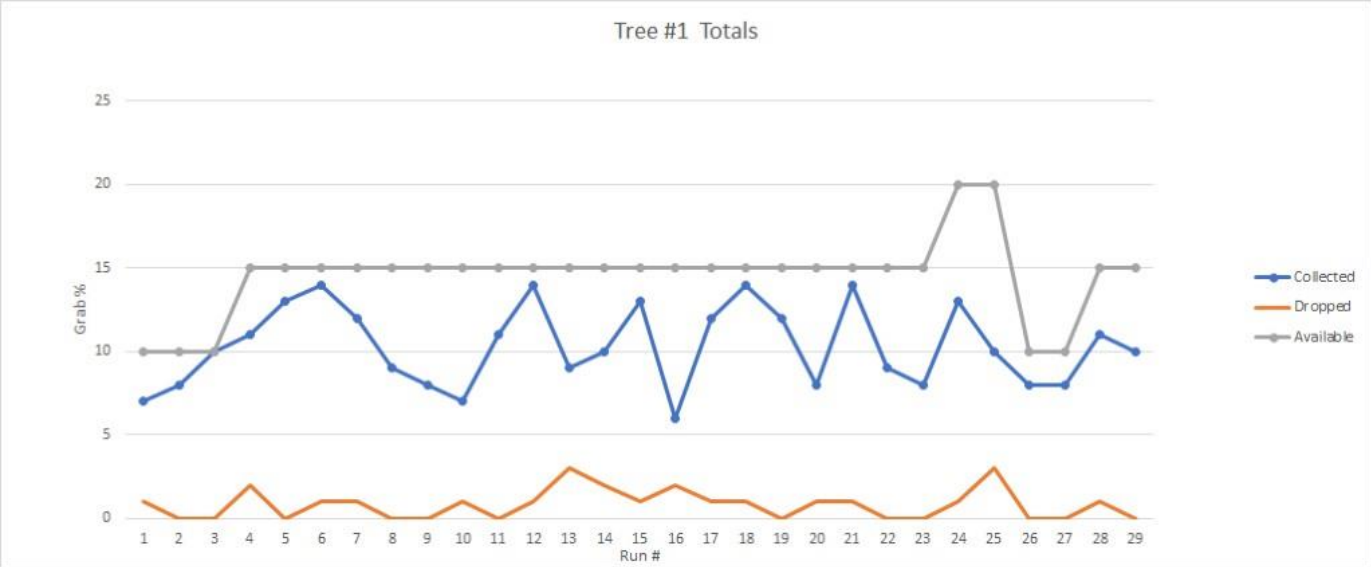


Fig. 45. **Robot Arm Grab at Tree One**

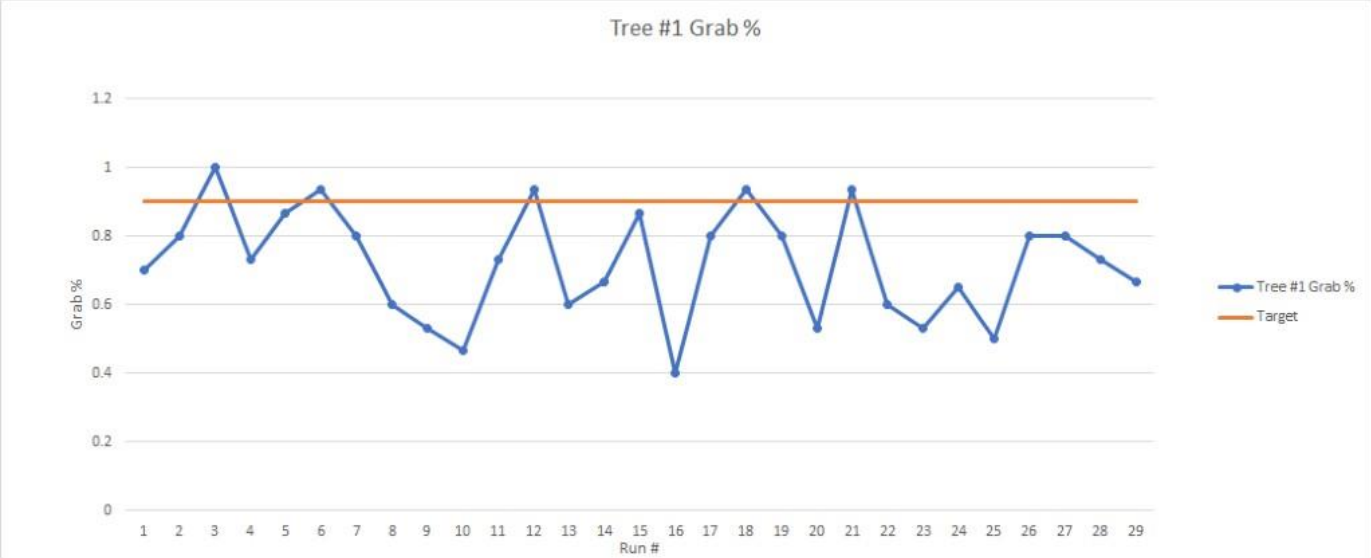


Fig. 46. **Robot Arm Grab at Tree One Percentage**

lining up incorrectly due to bad sensor readings. As previously discussed, this was often caused by beads left on the track during the bead picking phase. Without these instances, the accuracy would be substantially better, but would often still fail to meet the original desired metric. The main cause of this would be too many beads fired at a single net. Beads would clump together and the shooting mech would not generate enough momentum to launch them fully into the net. This could possibly be alleviated by opening the magazine more slowly and spinning the motor up more before firing. Also, too many beads in a single shot is more likely when more beads are on the trees, which is why the accuracy is at its highest when 10 beads are on each tree.

Unlike bead picking, however, the shooting time was much lower than the original goal. Where the original metric was 20 seconds, the robot was actually able to stop, line up, fire, and drive again in under 8 seconds. As previously discussed, this shortened time allowed for more time to be invested in the bead grabs.

D. Other Tasks: Targets: 100% Net Detection, 85% Marshmallow Moved, Beads Wanted in All 3 Compartments

Other tasks that the robot was required to perform included detecting nets to shoot, moving the marshmallow, and distributing at least 1 bead into each compartment of the magazine. The percentages of each of these tasks is shown in **Figure 51**.

E. Beads in All 3 Components

While it was originally desired to always fire beads into 3 nets regardless of their location, limitations and optimizations

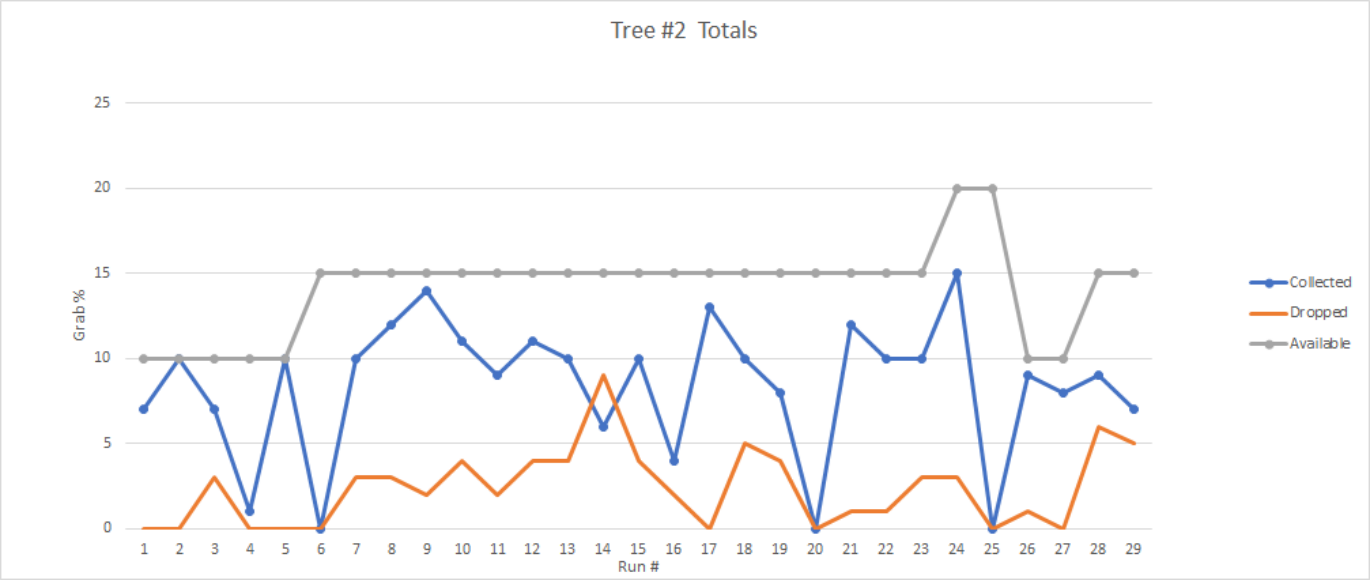


Fig. 47. **Robot Arm Grab at Tree Two**

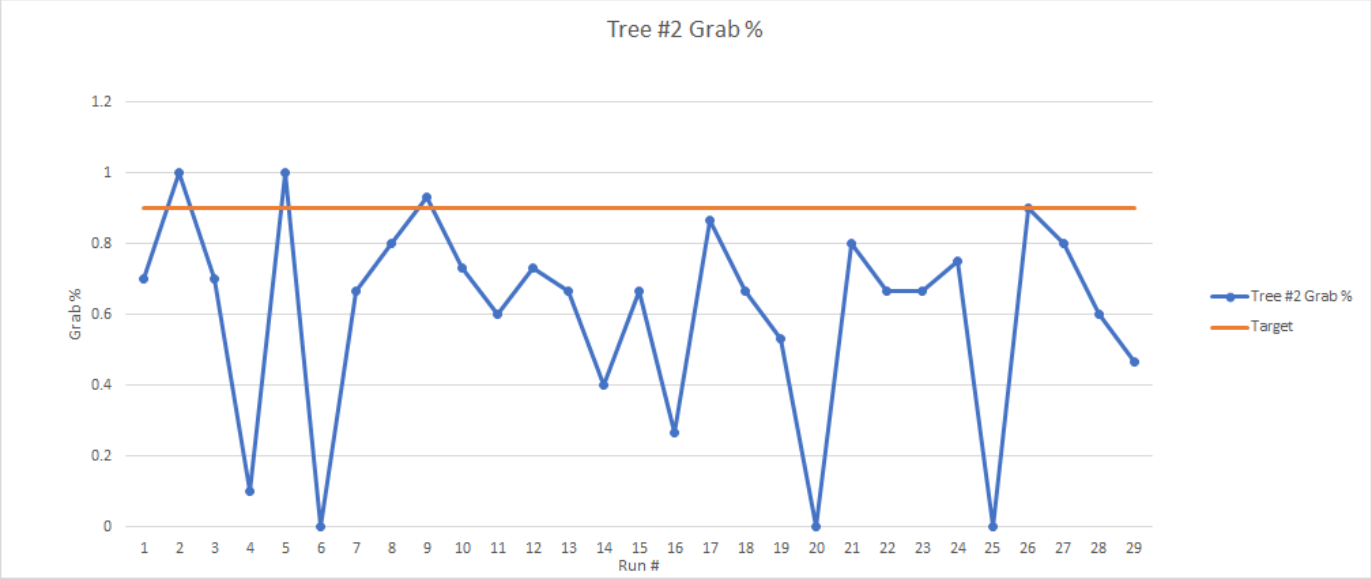


Fig. 48. **Robot Arm Grab at Tree Two Percentage**

closer to competition time required the robot to be tuned to instead only search 3 discrete locations for nets. If only 1 or 2 nets were identified instead of all 3, multiple compartments would be opened at a single net instead. Because of this, the emphasis on having beads in all 3 components was reduced in the final design. Despite this, the task was still accomplished almost 60% of the time. Also, a majority of these failures was either caused by the robot never reaching the second tree, which is where beads were pulled from to be placed in compartment 3 or by the 2nd tree's first grab being misaligned. Once the improvements were made to the second tree routine, the success rate of this task was 100% reliant on the robot making it to the second tree. If that was accomplished, beads were placed in all 3 compartments every time.

F. Marshmallow Pushed

The marshmallow was pushed off of the track 100% of the time that the robot successfully reached/lined up with the gap in the barricade. The only times the marshmallow did not get pushed out of the way were when beads on the track either caused the robot to become stuck or interfered with the ultrasonic sensors, tricking the robot into thinking the gap was in a different place. Despite these errors, the robot was still able to move the marshmallow approximately 90% of the time.

G. Net Detection

It was desired that nets be detected 100% of the time, but the minimum requirement would be that at least 1 net was

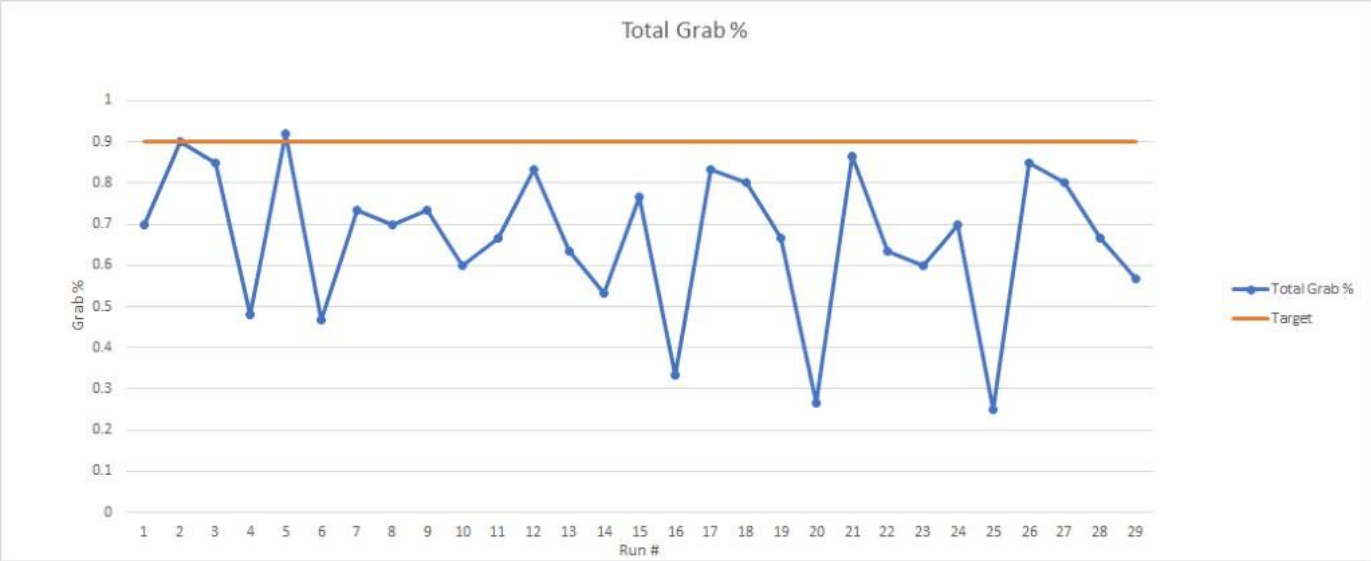


Fig. 49. **Robot Arm Grab Percentage Total**

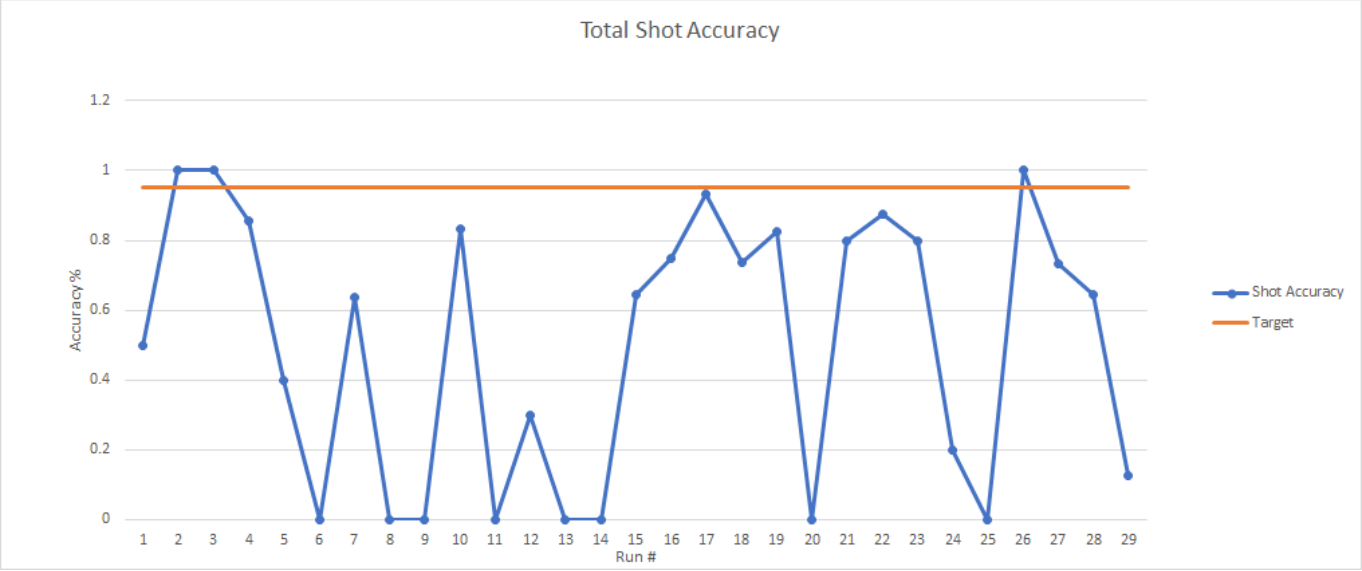


Fig. 50. **Total Shot Accuracy**

detected, allowing for at least 1 accurate shot per run. In its current state, the detection rate of all nets is approximately 50%, but the robot is much more likely to detect at least 1 net. The table below shows that this is achieved on every run, as long as the robot is able to traverse the course. Often, it would be impossible to determine if nets were detected because the robot would never make it to the shooting stage of its program. However, if at least one shot was fired, the number of detected nets could be determined by examining the number of compartments that were opened during the first shot. 1 compartment = 3 nets detected, 2 = 2 nets, 3 = 1 net detected. Overall, net detection was fairly accurate provided that the batteries for the lighting system were charged enough. Low batteries are suspected to be the cause of the drop in accuracy later in testing. Further improvements to the system would allow the robot to fire at all net locations, as it is believed that the vision system would be robust enough to detect nets at any location, provided the robot can fire at the locations.

H. Full Runs – Targets: 40 Points and <3:00

The point totals of each test run and the total times were recorded as if these were live competition runs. The points across each run are shown below. Additionally, each run ended before the 3:00 time limit, meaning that this goal was met.

While the target was not always reached, these point losses were usually attributed to incomplete runs or a large number of point deductions from dropped beads or barricades hit. The data shows that achieving at least 40 points is possible and a solid baseline for a successful run. With further improvements to bead grabs, issues returning to the starting square and

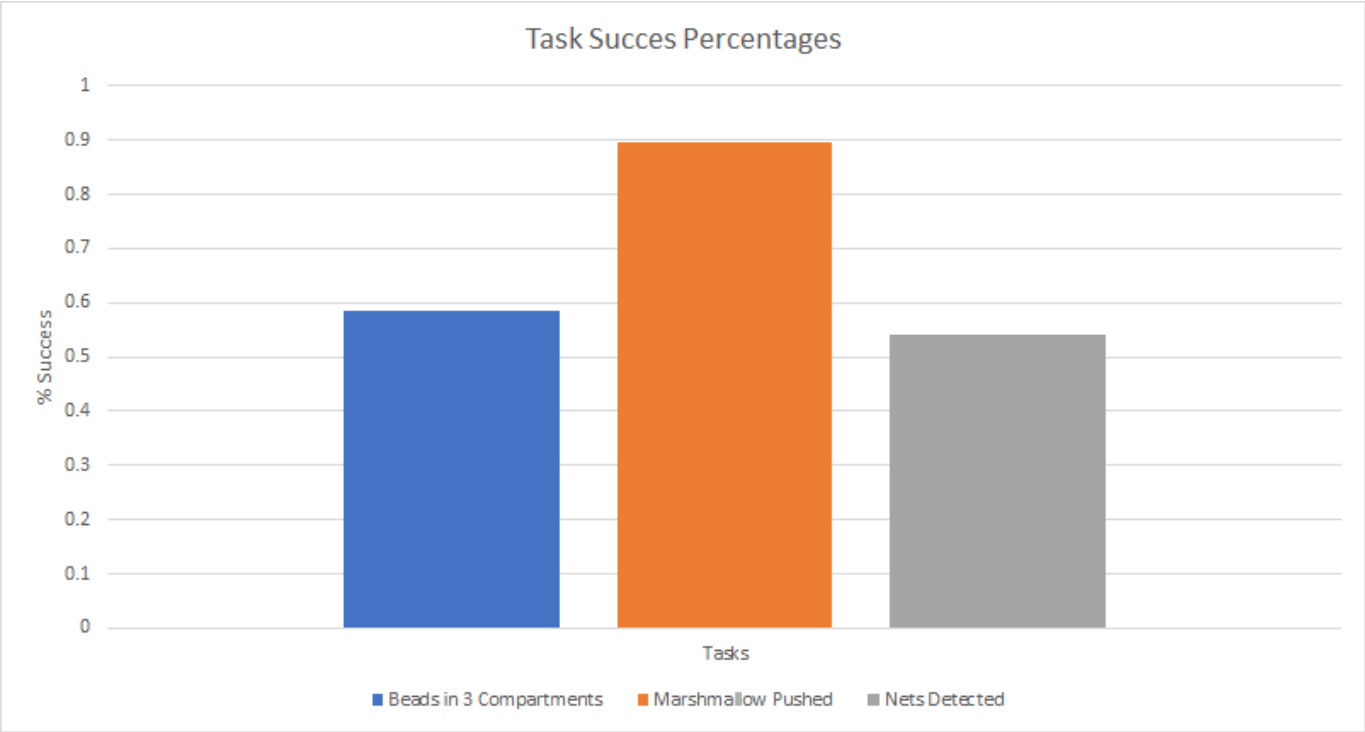


Fig. 51. **Task Success Percentages**

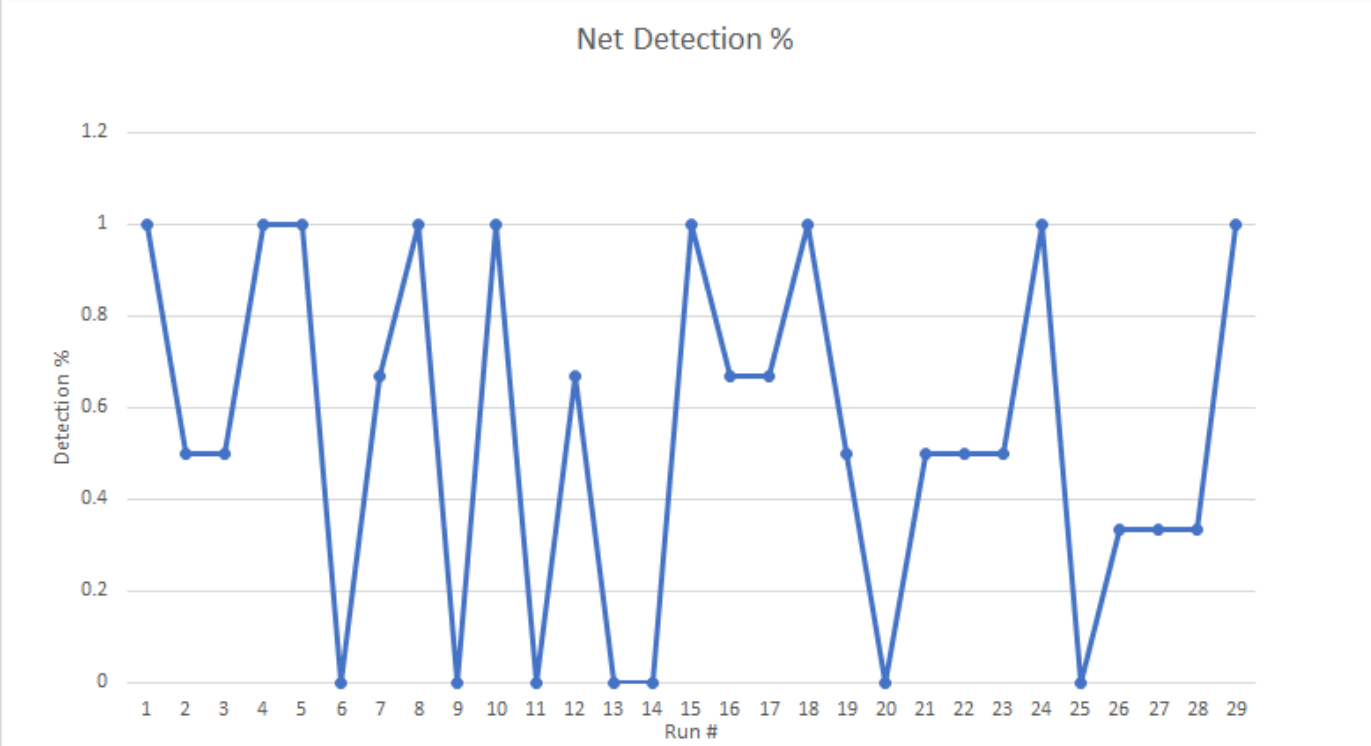


Fig. 52. **Camera Detection Percentage**

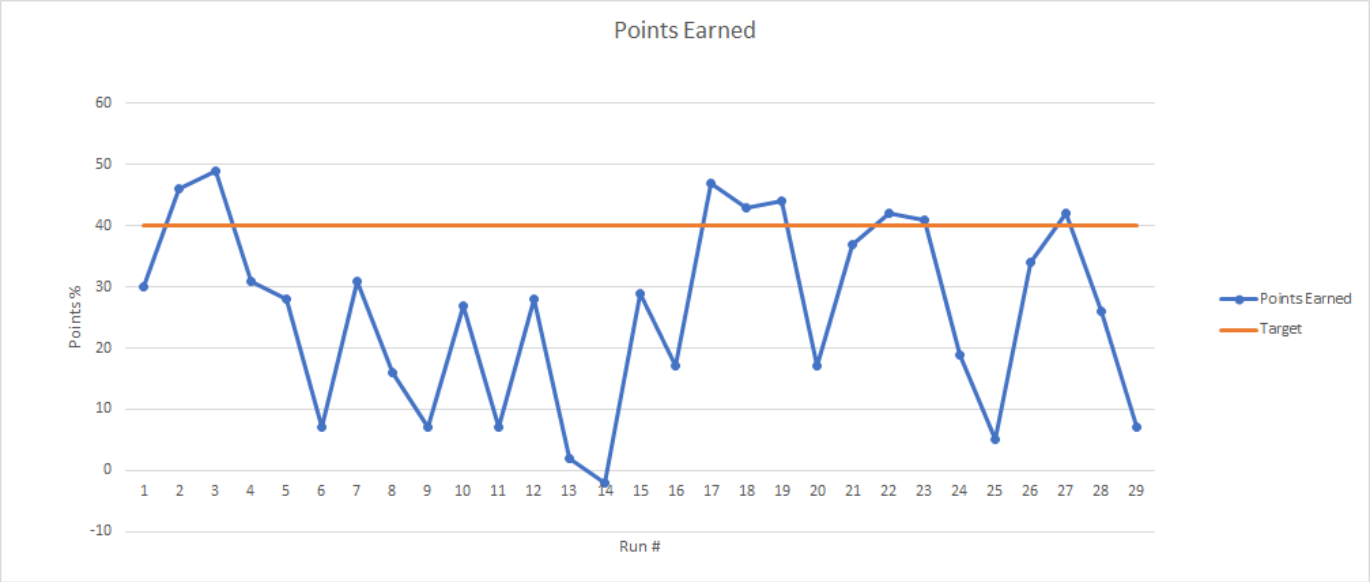


Fig. 53. **Points Earned Per Run**

dropping beads (the two biggest causes of point loss) can be reduced dramatically.

I. Traversal Speed – Target: 2 ft/s without leaving line

The robot rarely travels a large distance without stopping while completing competition runs, making it difficult to analyze the straight line speed of the bot during these experiments. Because of this, separate runs in which the robot would only drive were completed to determine its speed and driving capability. The robot was tested for both full runs of the course (~8.75 ft) which included the 90 degree turn and half runs of the course that only measured its speed on the longer straight section.

Type	Distance (ft)	Time (seconds)	Speed (ft/s)
Full Track with Turn	8.75	12.42	0.704508857
Full Track with Turn	8.75	13.02	0.672043011
Full Track with Turn	8.75	12.07	0.724937862
Full Track with Turn	8.75	12.12	0.721947195
Full Track with Turn	8.75	11.68	0.749143836
Half Track	6	7.15	0.839160839
Half Track	6	6.26	0.958466454
Half Track	6	6.41	0.936037441
Half Track	6	6.37	0.941915228
		Average	**0.805351191**

Fig. 54. **Track Traversal Time and Speed**

The data shows that the robot operates at an average speed of 0.805 feet per second. However, this could easily be increased by increasing the duty cycle of the PWM signals being sent to the drivetrain. Right now, that duty cycle is only 20 % – 25 % of the maximum. This was never increased because the current speed was adequate to meet the 3:00 time limit comfortably, and a faster speed was not necessary.

J. Power System – Target: Constant Output Voltages on Bucks and Buck/Boost Converters

The first element of the power system to be tested was the Buck/Boost Converter. The purpose of this device is to provide a constant output voltage of 12 Volts regardless of the input voltage. This was desired as it would normalize the power system's voltage level regardless of the varying battery voltage. To test this, a range of input voltages similar to the batteries operating voltages were supplied to the device, and then the output voltage was measured.

This data shows that the device works as intended, and will be able to supply a steady output voltage even if the battery voltage is not constant. It is important to note, however, that the lower end of the test range is far below the recommended battery voltage level, and if the battery were to operate at such a low voltage, especially when connected to this device, the current draw would quickly deplete the battery and cause irreversible damage.

The second step in verifying the power system was to determine that the Buck Converters that serve as direct Power Supplies to the robot's electronics were capable of outputting a constant voltage given a constant input voltage of 12 Volts from the aforementioned Buck/Boost Converter. Since the Buck/Boost has already been verified to output a constant 12 Volts if given any voltage in the range of 9 to 14 Volts, so by connecting these components to the Buck/Boost and measuring their outputs, their operation can be verified.

This table shows that each device is capable of maintaining its voltage level with little variation. Each converter was tuned in such a way that small changes in voltage, either increases or decreases, would not exceed or fall below the connected electronics' operating voltage range. The Raspberry Pi can operate between 4.75 and 5.25 Volts, The Servo Shield can

Input Voltage (V)	Output Voltage (V)
9.0	12.0
9.5	12.0
10.0	12.0
10.5	12.0
11.0	12.0
11.5	12.0
12.0	12.0
12.5	12.0
13.0	12.0
13.5	12.0
14.0	12.0

Fig. 55. **Buck-Boost Converter Capability**

Device	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
Raspberry Pi Buck Converter (5 Volts)	5.0	5.0	4.9	5.0	5.1	5.0	5.1	4.9	5.0	4.9
Servo Shield Buck Converter (5.5 Volts)	5.5	5.4	5.4	5.5	5.5	5.5	5.5	5.4	5.5	5.5
Arduino Buck Converter (7.0 Volts)	7.1	7.1	7.0	7.1	7.0	7.1	7.0	7.1	7.1	7.1

Fig. 56. **Input Voltage Per Buck Converter**

operate between 5.0 and 5.7 Volts, and the Arduinos can operate between 5.0 and 7.2 Volts (with a desired voltage closer to 7.0 Volts). Each of these specifications are met by the appropriate device.

K. Deliverables – intended and accomplished

The team started the project with the scope to build a robot to be able to accomplish task and follow strict constraints set forth by SoutheastCon. Aside from this, the team set its own desired outcomes. The desired outcomes were previously referenced at the start of the report. The team did well with most desired outcomes. The team was able to produce a robot with tested subsystems that could do the desired functions. For this reason, the design and functionality of the robot was a success; however, the robot did not do as well as expected in the competition like previously described. Overall, the team was able to build a functional robot that could compete in the competition.

VI. TIMELINE AND COST ESTIMATES

A. Goals, Planning, Review and Gantt

The project timeline is based on the outline provided in the course syllabus. It is comprised of four overarching sections: Preliminary Research, Design Phase One, Design Phase Two, and Development/Testing. Each of the four main sections have various tasks and deadlines within them.

"During the preliminary research phase, the team will independently research the various subsystems outlined in the proposal and try to come up with the most viable, cost-effective solution. Although research will be conducted independently, it is important that the team members discuss what they find during team meetings in order to address issues of compatibility, cost, etc. Team assignments for the preliminary research phase are as follows: Daniel (Microcontrollers), Lexi (Vision System and Peripheral Systems), Sawyer (Power Distribution), and Chase (Motor Control/Drivetrain). Preliminary research should be completed by **October 4** th.

After the preliminary research has been completed, the team will begin work on finalizing the first design for Design Phase I. This phase will be where the team discusses the viability and compatibility of each of the members' findings in the previous phase. Design Phase I will see the team have a general idea for a solution as well as having selected the parts that will comprise the main subsystems. Design Phase I should be completed and finalized by **October 18** th.

Design Phase II is where the team will begin the detailed design and analysis of their proposed solution from Design Phase I. During this phase, the team will have already finalized all of the various parts that will be used in the design and will be working on the interfacing of the subsystems. This phase will see the team design the chassis and wheelbase, as well as complete the wire layout. The team should have a draft by **November 24**th. Following the draft of Design Phase II, the team will need to begin ordering parts. As of the creation of this proposal, the team's designated parts acquirer will be Sawyer. The team will need to have their final design verified before moving on to the final phase. The final draft of Design Phase II will be completed by **December 6** th.

Due to Christmas break, Development and Testing will begin on **January 10th, 2022.** It is expected that all parts will have been ordered prior to this stage. This phase will see the team actualize its design. During this phase, the team will get the various subsystems of the design to at first work independently, and then as a unit. Although it is expected that each team member will play a role in all aspects of development, they will also have aspects of the product that they are considered responsible for. Daniel will oversee interfacing all subsystems with the central controller. Lexi will be responsible for programming the vision system and peripherals. Sawyer will oversee power distribution and wiring. Finally, Chase will be responsible for programming the motor controllers and building the drivetrain system/chassis. The hard deadline and competition date of this project is **March 31st , 2022**; however, the team should have a functioning product that meets competition specifications by **March 25** th."

It is important to note that some of the deadlines set were met while others were more ambitious. Everything through Design Phase I was able to be accomplished on time; however, Design Phase II draft had to be pushed back. Due to this and busy schedules, the team also had to push back the completion of Design Phase II to after Christmas break. By doing this, the timeline had greatly changed for the team. Ordering was

not completed until **February 2nd**. Which then meant that small testing started as parts started coming **February 15th**. In saying this, the team did have a final product by **March 25 th**, but the team continued to work on the robot so that the competition would have the best version of the robot to compete.

B. Cost estimate vs. actual

This budget will serve as a preliminary estimate of the cost of the project and a metric to determine the project's financial feasibility. Exact component costs will become finalized in the later design phases as more specific details are known. The budget is broken down on a sub-system by sub-system basis. Each component price is listed as a range of the unit price. The total component cost range is then calculated by multiplying the lowest quantity by the lower bound of the range and the largest quantity by the upper bound. Total ranges for each sub-system, and the system total are then calculated by summing the lower and upper bounds of each component, respectively. A 50 percent margin of error is then added to the total to account for additional components and price changes/shipping costs. The final grand total is then listed at the bottom of the table. The reader can see that the team went over prospected budget by 1.17%. Though the budget was over exceeded, the design process yielded an accurate cost estimate. The estimated and actual budget breakdown can be seen in **Figure 57**.

VII. Ethical and Professional Responsibilities

The project has allowed the team to see just how important the engineering process is to an engineer. The team recognizes the transformation of the project from a block diagram to a final product and just how much has changed from the start. It was vital for each team member to be open to criticism on each subsystem. It was also vital that each team member was open to help. Through both steps, the team was able to produce the best product. The team learned how to ask for help and how to make suggestions in a professional manner. The team also learned how to work with a team even through disagreements. Concepts like these will play vital parts in each team members career.

A. Societal Impacts

It is possible that this project will have some effects on the school, including its perception throughout the region. Throughout the design and building process, it is expected that the team will take on some members from the IEEE chapter in small capacities. A project such as this has the potential to attract current members as well as new members to the chapter. In addition, the team performing well in competition will reflect well on the school, specifically Tennessee Tech's Electrical and Computer Engineering program. Performing well in competition may catch the attention of prospective students who are interested in engineering and will make a case for what Tech's engineering program has to offer compared to many other programs in the region. In addition, it may open other opportunities for future capstone groups to participate and compete in similar competitions.

B. Lessons learned

The team has learned many lessons throughout the design and implementation of the project. One main lesson learned is that software can start being written before hardware components arrive. Doing this will help the team stay on track in future projects or possibly finish early. Another lesson includes buying test equipment that exceeds component spec. This lesson was learned after chasing a brown out problem. In the end, the problem turned out to be the limiting current of the wall wart that the team had purchased. Another lesson learned is that the team should design future installments below physical specification. This lesson learned comes from comments made from the judges at the competition. Finally, the team learned the importance of using outside resources for areas of little expertise. The idea came in handy when working on the more mechanical factors of the project.

VIII. Conclusion

The goal of the project was to build an autonomous that would navigate a course and complete task for points in SoutheastCon 2022 Hardware Competition. The paper has outlined key aspects that went into designing the robot and the functionality of the final product.

Appendix

Attached is the code, diagrams, and circuit models that the team used to build the robot.

Chase Garner, Sawyer Hall, Lexi Sheeler, and Daniel Summers Department of Electrical and Computer Engineering

Subsystem/Component	Qty.	Unit Cost	Total Cost
Central Controller			
Microcontroller	1	$20-$50	$20-$50
Total			$20-$50
Vision System			
Object Detection Hardware	1	$30-$80	$30-$80
Navigation Hardware	1	$20-$50	$20-$50
Total			$50-$130
Motor Controller			
Motor Driver/PWM Chip	1	$10-$20	$10-$20
Brushless Motor	2 to 4	$5-$25	$10-$100
Servo Motor	3 to 6	$10-$40	$30-$240
Servo Mounting Bracket	3 to 6	$5-$10	$15-$60
Total			$65-$420
Power Supply System			
Battery	2	$20-$40	$40-$80
Switch	2	$5-$10	$10-$20
Fuse	10	$0.80 - $1	$8-$10
Custom PCB	1 to 3	$10-$50	$10-$150
Total			$70-$260
Peripheral System			
LEDs	10 to 30	$1-$2	$10-$60
LCD Display	1	$5-$20	$5-$20
Speaker	1 to 2	$2-$10	$2-$20
Total			$20-$100
Chassis			
3D Printer Filament	1 to 3	$5-$20	$5-$60
Total			$5-$60
General			
Fasteners	N/A	N/A	$20-$50
Wiring	N/A	N/A	$20-$50
Construction Materials (Game Board)	N/A	N/A	$80-$150
Total			$140-$250
System Total			$370-$1270
Error Margin/Redundancy			50%
Grand Total			**$550-$2000**

Subsystem/Component	Qty.	Unit Cost	Total Cost
Central Controller			
Arduino Mega	1	$34.25	$34.25
Total			$34.25
Vision System			
Raspberry Pi	1	$149.99	$149.99
Camera	2	$29.99	$59.98
Total			$209.97
Peripherals			
Arduino Uno	1	$20.98	$20.98
Screen	1	$9.99	$9.99
LEDs	1	$7.99	$7.99
Audio Shield	1	$34.95	$34.95
Total			$73.91
Arm			
Arm	1	$399.00	$399.00
Arm Servo	1	$17.50	$17.50
Total			$416.50
Drive Train			
Wheels	1	$70.94	$70.94
H-Bridge	2	$8.99	$17.98
Motor	5	$24.95	$124.75
IR Sensors	4	$8.99	$35.96
Ling Following Sensors	3	$7.00	$21.00
Total			$270.63
General			
Gameboard supplies			$182.09
Hardware			$160.27
		Total	$342.36
Chassis			
Acrylic	4	$19.32	$77.28
Total			$77.28

Turntable			
Motor Mount	1	$19.95	$19.95
Motor	1	$34.99	$34.99
Total			$54.94
Shooting Mechanism			
Belt	1	$20.98	$20.98
Compartment Motor	1	$19.95	$19.95
Coupling	1	$4.99	$4.99
Shaft	1	$8.02	$8.02
Bearings	3	$9.56	$28.68
Driver Motor	1	$24.95	$24.95
Total			$107.57
Power			
Emergency Stop	1	$12.99	$12.99
Start/Stop Switch	2	$14.85	$29.70
Wall Wart	1	$16.99	$16.99
Buck	1	$15.99	$15.99
Buck Boost	1	$55.99	$55.99
Power Bus	3	$9.49	$28.47
Battery Charger	1	$35.95	$35.95
Battery	2	$120.00	$240.00
Total			$436.08
Grand Total			$2,023.49

Fig. 57. **Estimated Cost vs Actual Cost Component Breakdown**

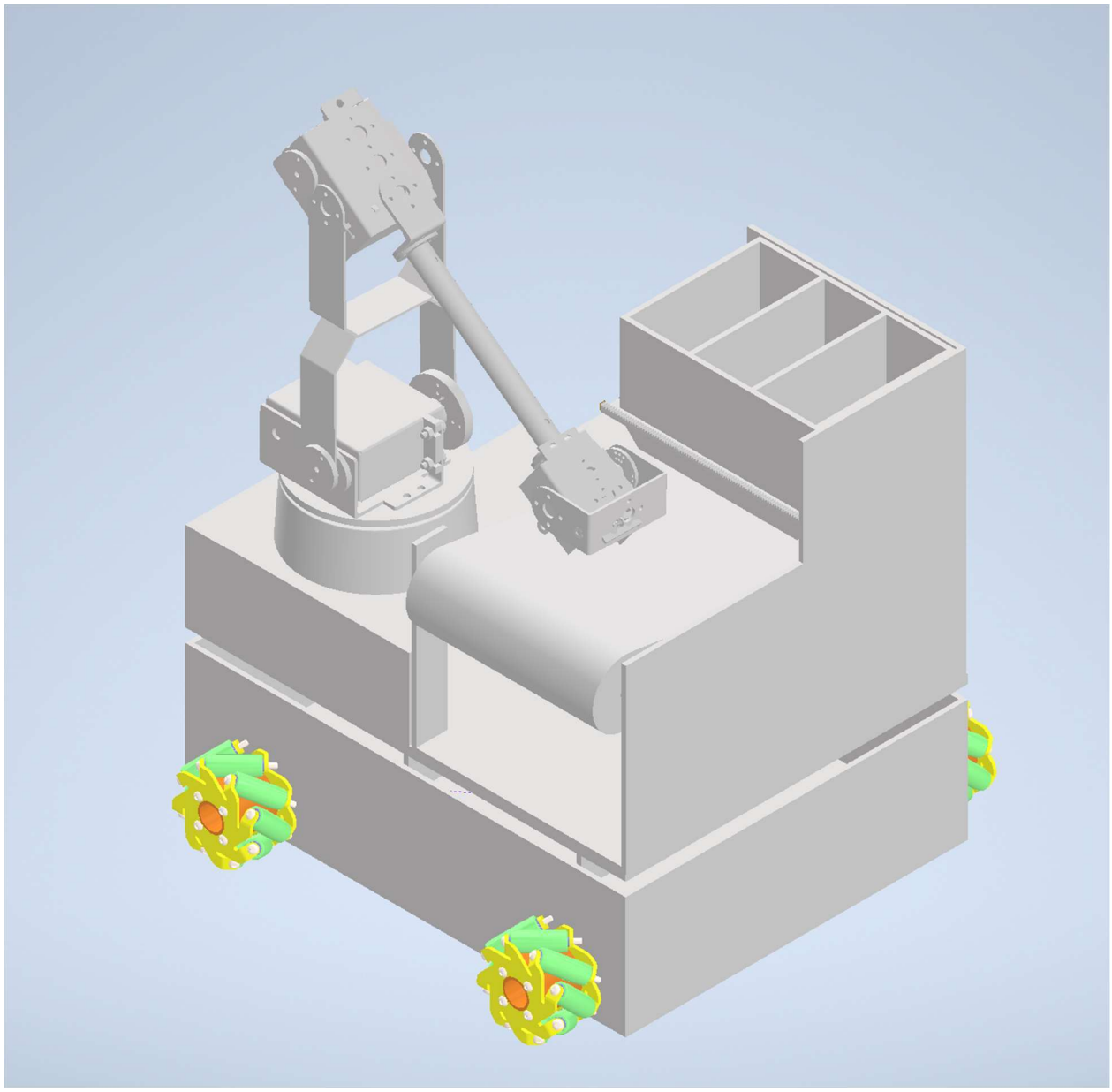
1. Current Draw Calculation

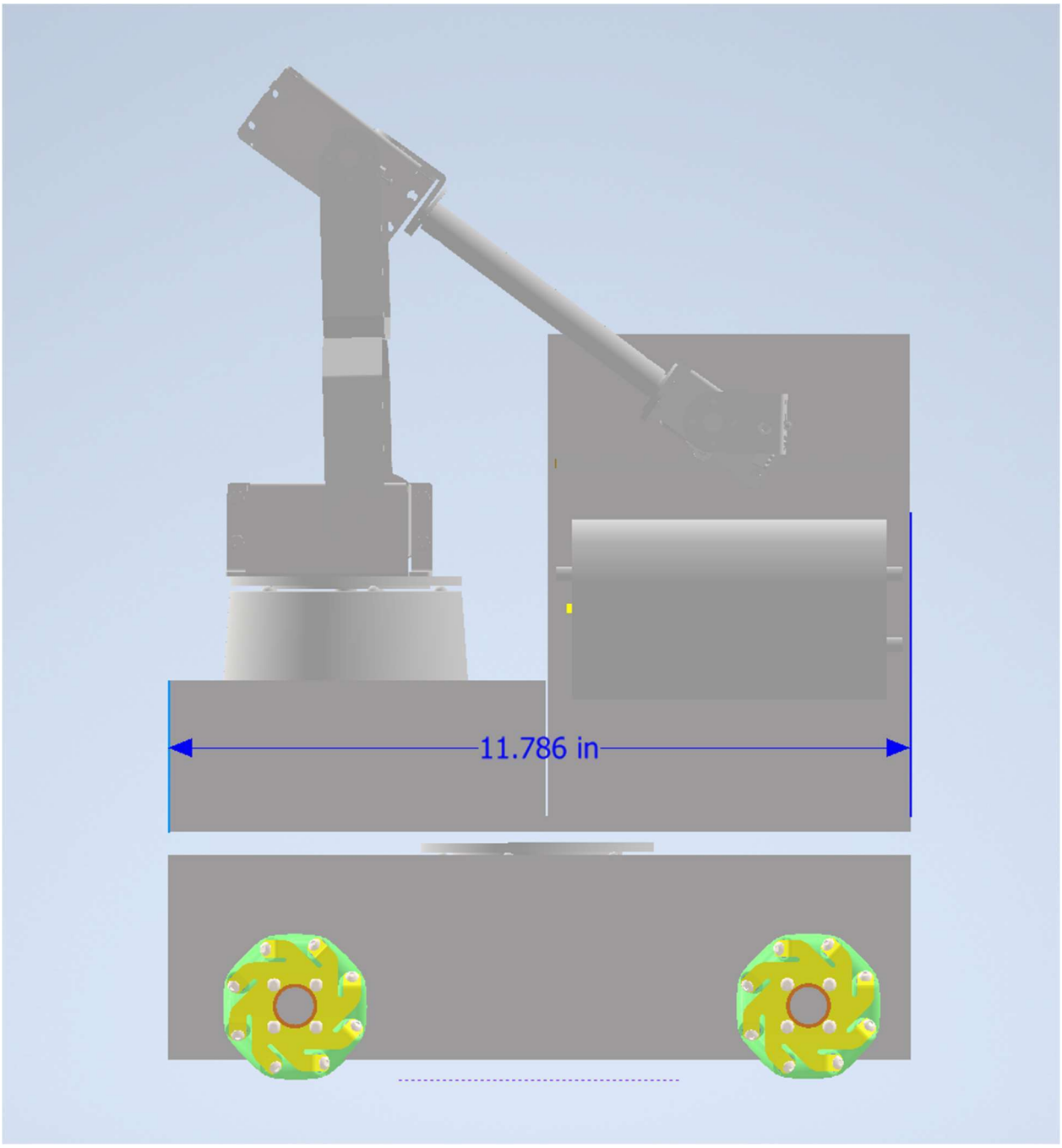
```
*************************************************************************************************
clear all;
close all;
clc;


%The code below calculates the needed current supply from the battery for
%Capstone Power Signoff justification
%The code also calculates the C value of the battery to find out how long
%the battery will be able to operate

%Set values for each component to find current consumption
Pi = 3;
Shield = 0.25;
MEGA = 0.5;
Uno = 0.3;

%Set Values for Buck Boost and Buck Converter Efficiency
BuckboostE = 91.3/100;
BuckE = 92/100;

%Powerbus loss is neglible therefore no measurable loss will be accounted
%for

%12 Volt to 5.1 Volt Buck Converter
Iz = ((5.1*(Pi+Shield))/12)/BuckE

%12 Volt to 7 Volt Buck Converter
Iy = ((7*(MEGA+Uno))/12)/BuckE

%We can now find the needed current that the battery will have to supply
Ia = 1.2;
Ib = 1.2;
Ix = (Iz+Iy+Ia+Ib);
Iactual = ((12*(Ix))/12)/BuckboostE

%C value calculations
%Per datasheet the battery can run 6.6 Amps for 1 Hour at 1 C
Idata = 6.6;
CurrentRatio = Idata/Iactual;
TimeinMinutes = CurrentRatio*60
*************************************************************************************************
```

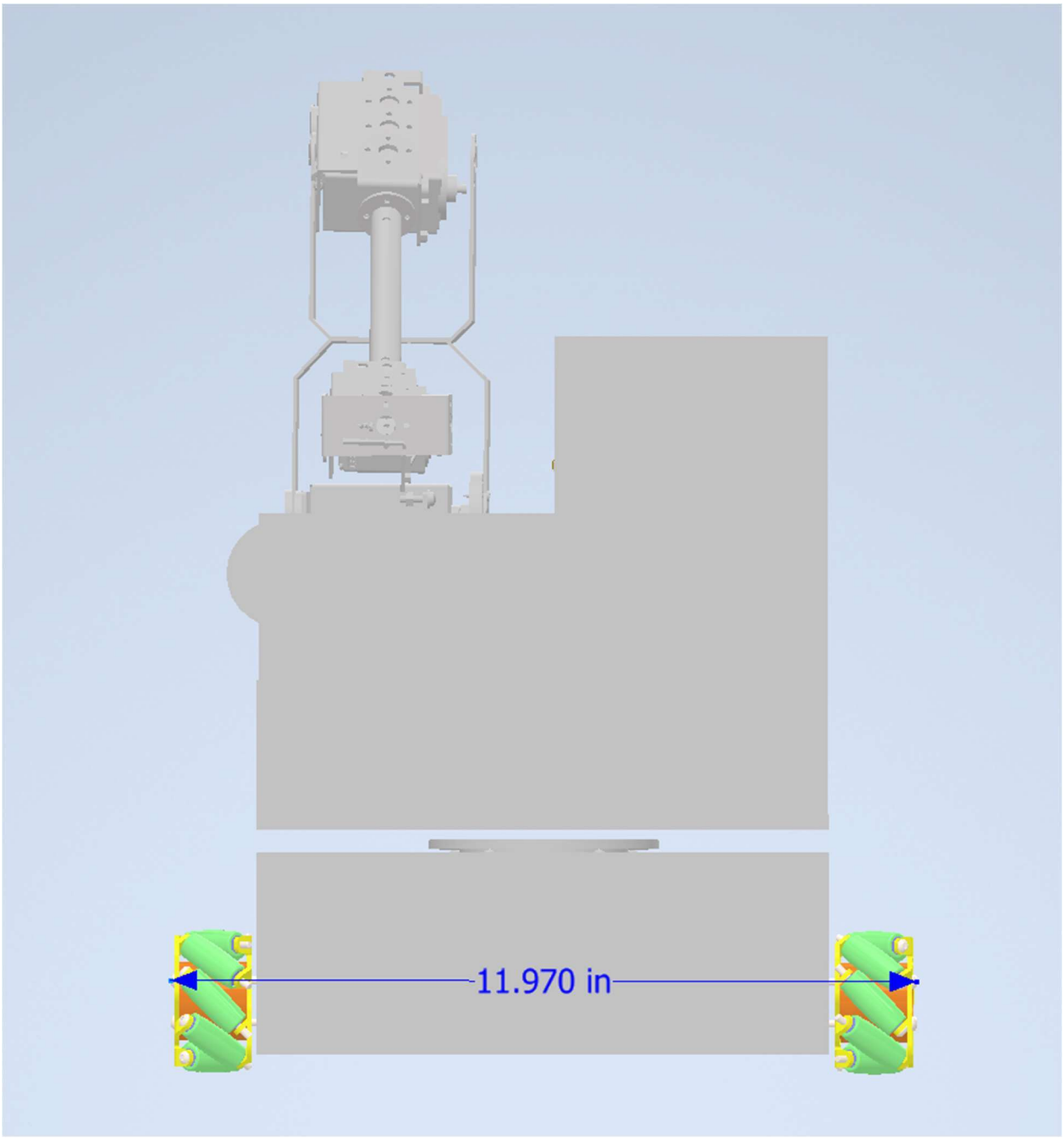
2. Shooting Trajectory Calculation

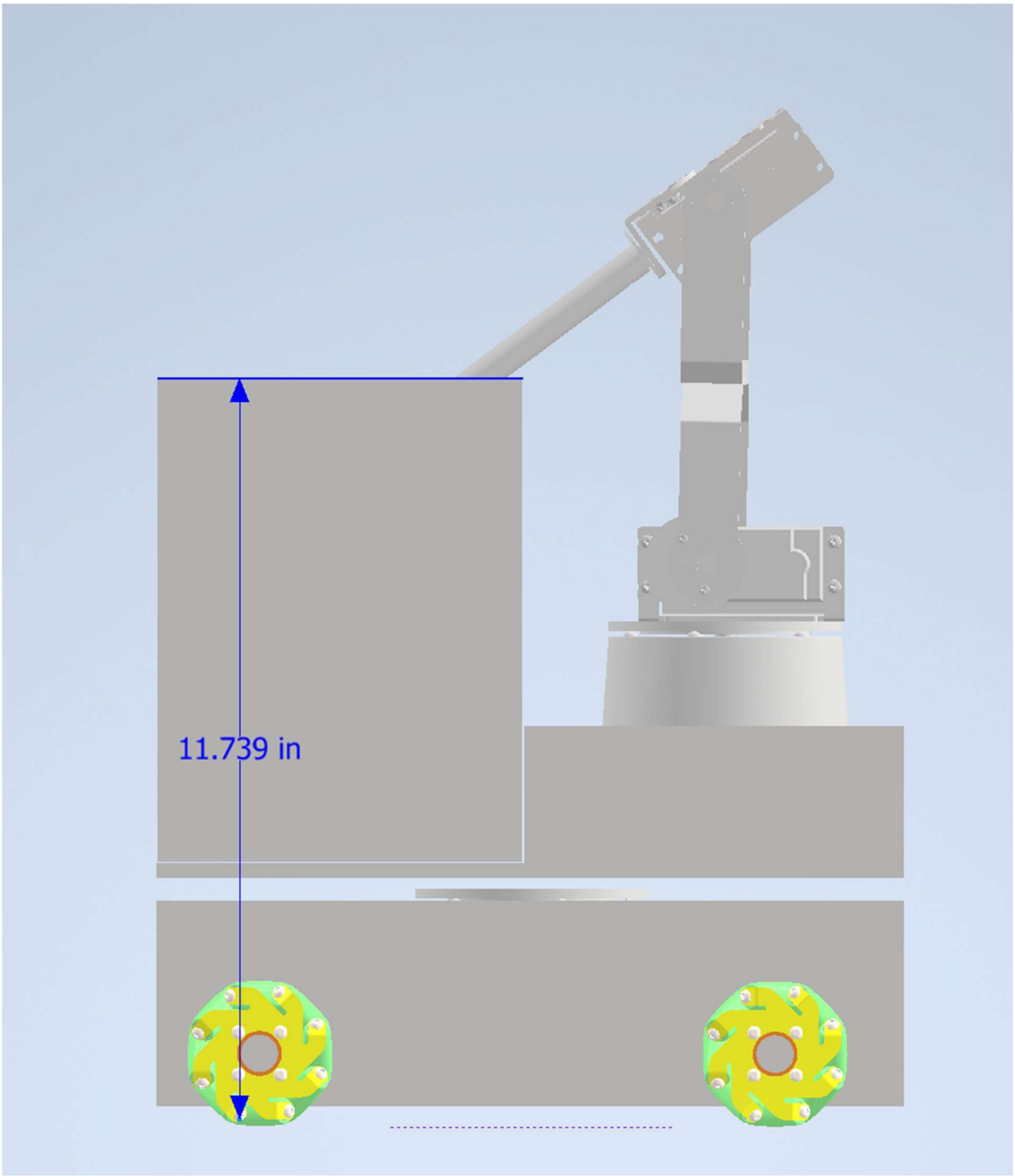
```
%Script to optimize launch angle and conveyor height

%These variables are changeable
motor_rpm = 700;
launch_degrees = 8;
distance_to_net = 5;

%Physics
gravity_inches = -386.22;

%Acceleration has been converted to in/s^2
exit_height = sind(launch_degrees) * 9.17 + 7.8;

exit_velocity_inches_second = (2*pi*0.75*motor_rpm)/60;

additional_height_gained = -
(sind(launch_degrees)*exit_velocity_inches_second)^2/(2*gravity_inches);

time_max_height = 2*additional_height_gained/(sind(launch_degrees)*exit_velocity_inches_second);

time_of_flight = time_max_height + sqrt(2*(additional_height_gained+exit_height)/386.22);

distance_traveled = (exit_velocity_inches_second*cosd(launch_degrees))*time_of_flight;

max_height = exit_height + additional_height_gained;

%Plot The Trajectory
t = linspace(0, time_of_flight);
d = linspace(0, distance_traveled);

height = exit_height +
sind(launch_degrees)*exit_velocity_inches_second.*t+.5*(gravity_inches).*t.^2;

hold onplot(d, height);

degstr = sprintf('Launch Degrees %.1f', launch_degrees);

%legend(degstr);
ylim([0 14]);
xlim([0 6]);

%xlim([0 6]);
xlabel('Distance From Conveyor (inches)');
ylabel('Bead Cluster Height (Inches)');
title('Flight Path of Bead Cluster');

%Draw the net for visual aid
line([distance_to_net distance_to_net], [5.5 13.5], 'linewidth', 8, 'Color', 'black');
line([distance_to_net distance_to_net], [0 5.5], 'Color', 'black');
```

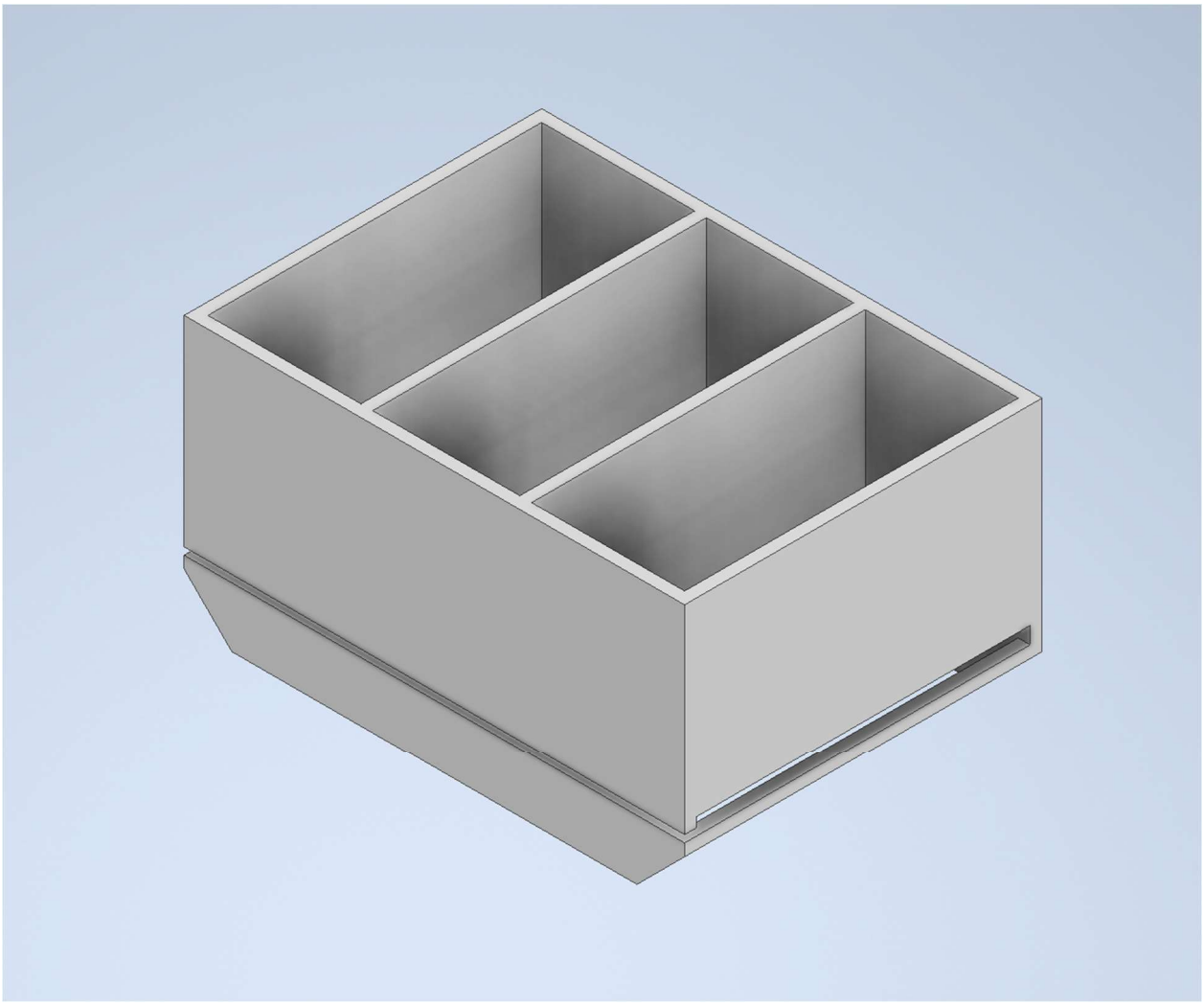
Appendix B: Full CAD Documentation

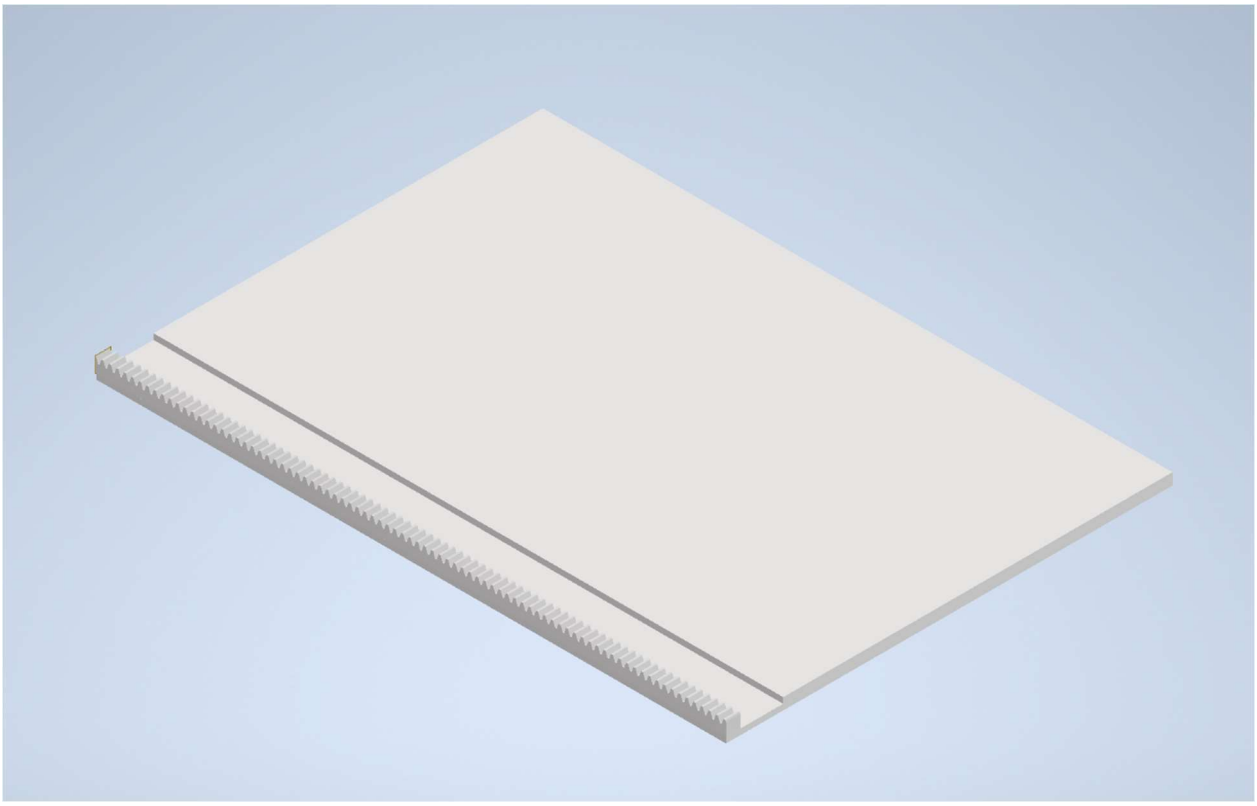


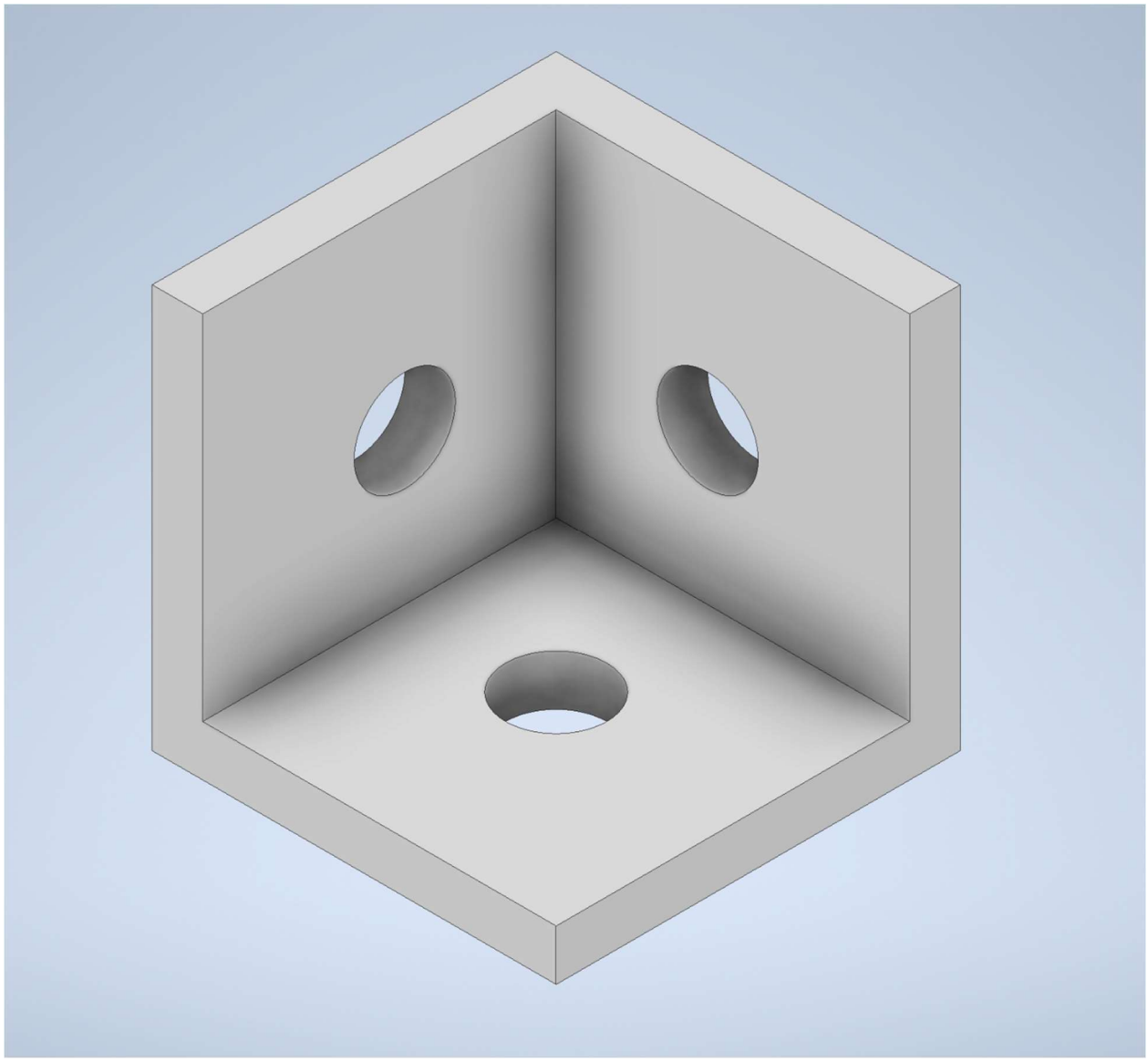

11.786 in

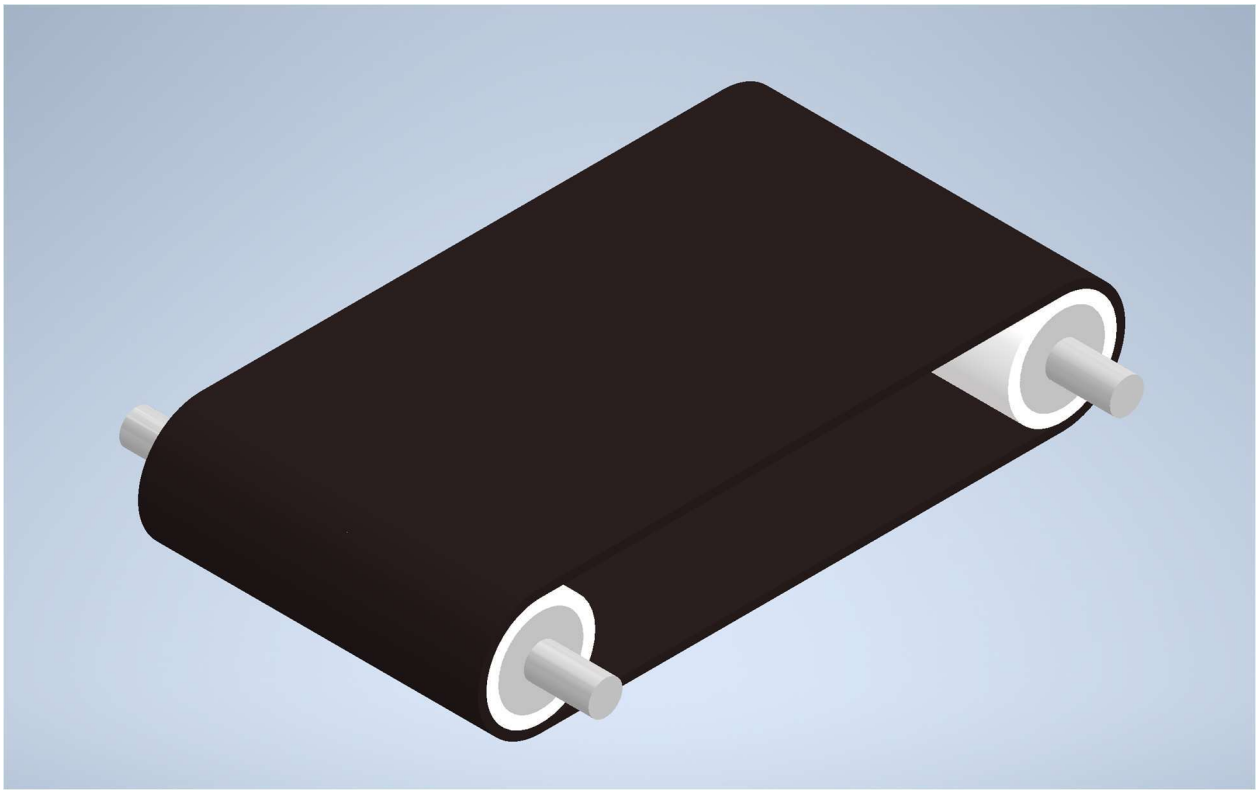

11.970 in

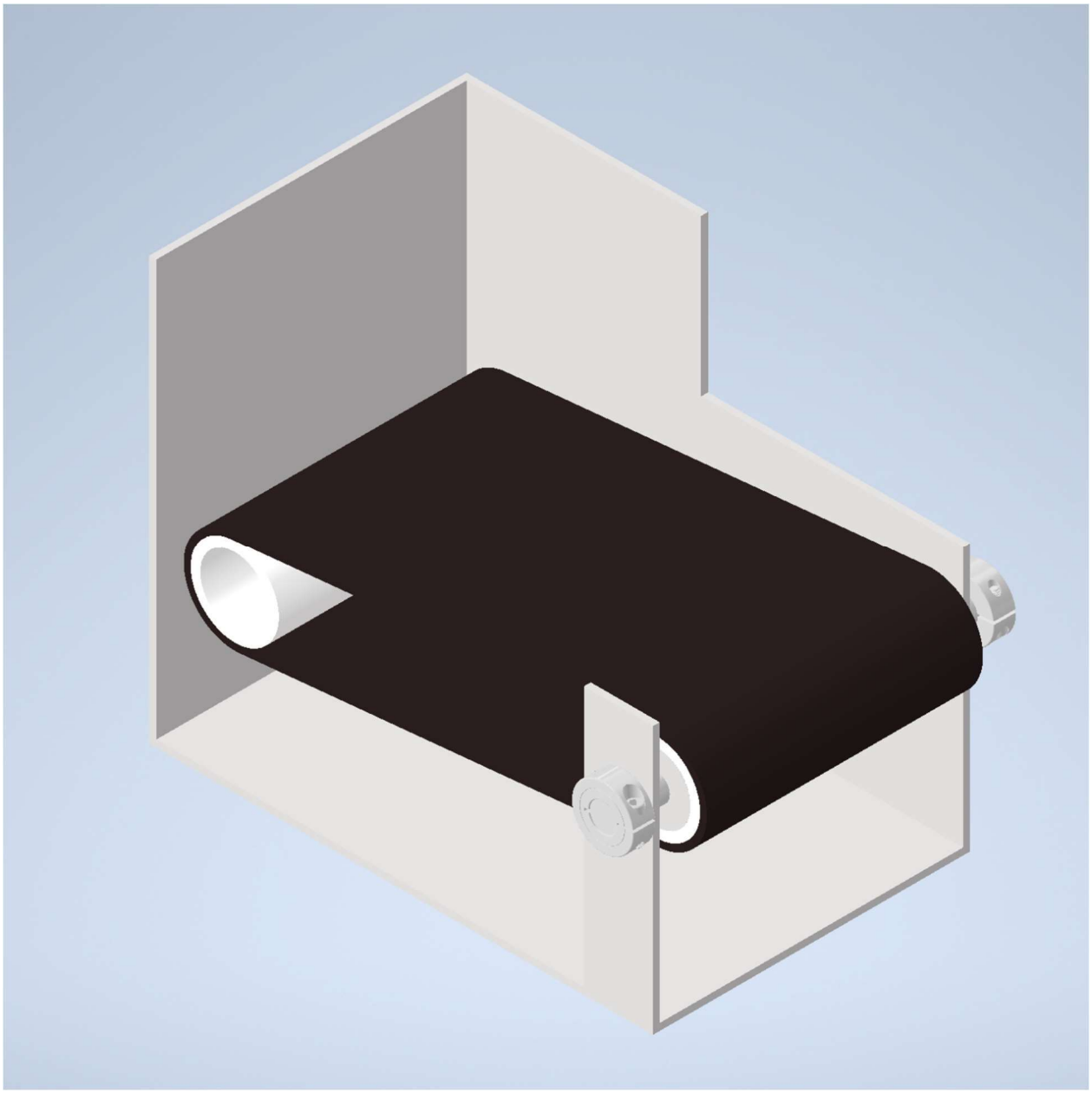

11.739 in

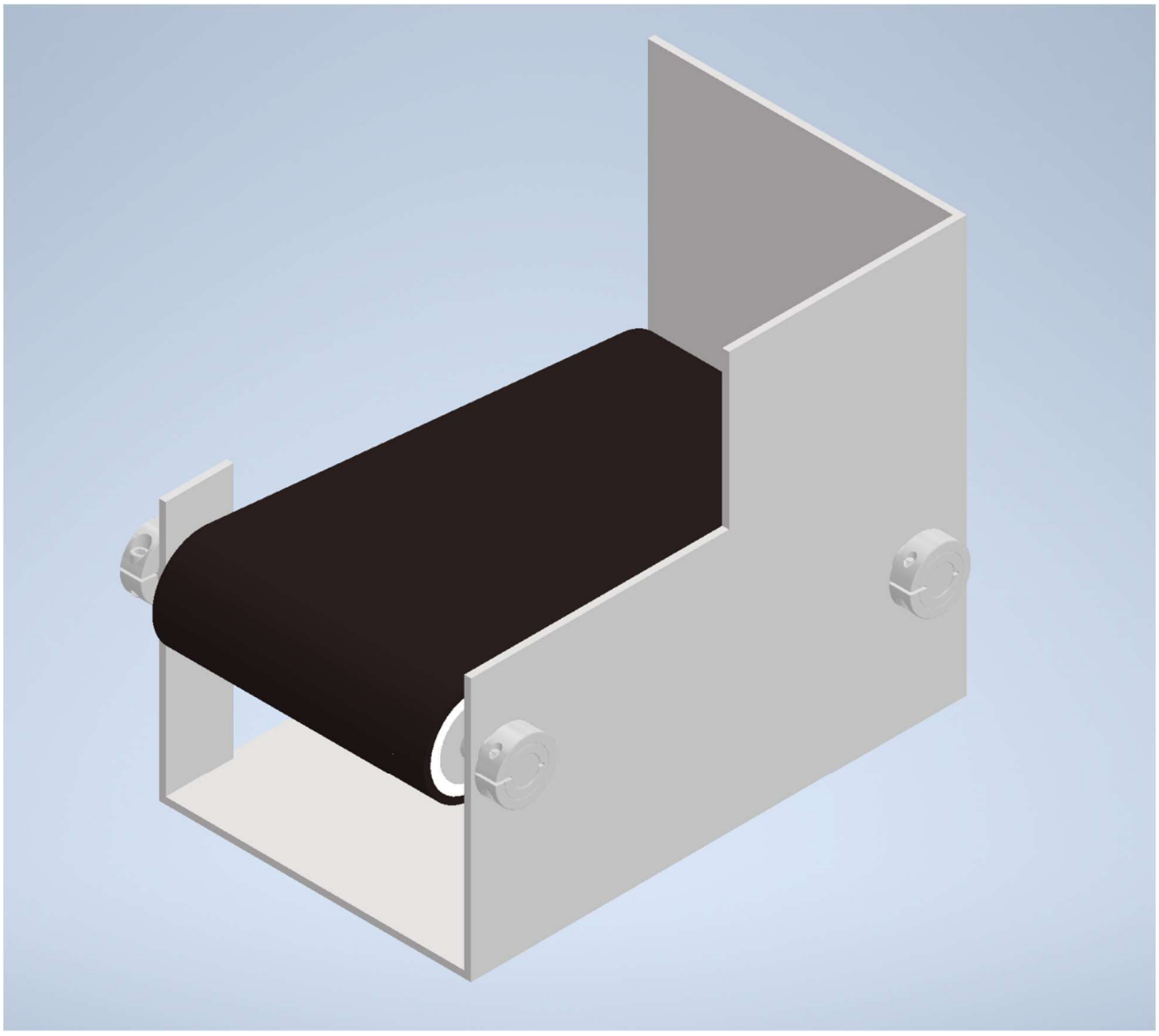






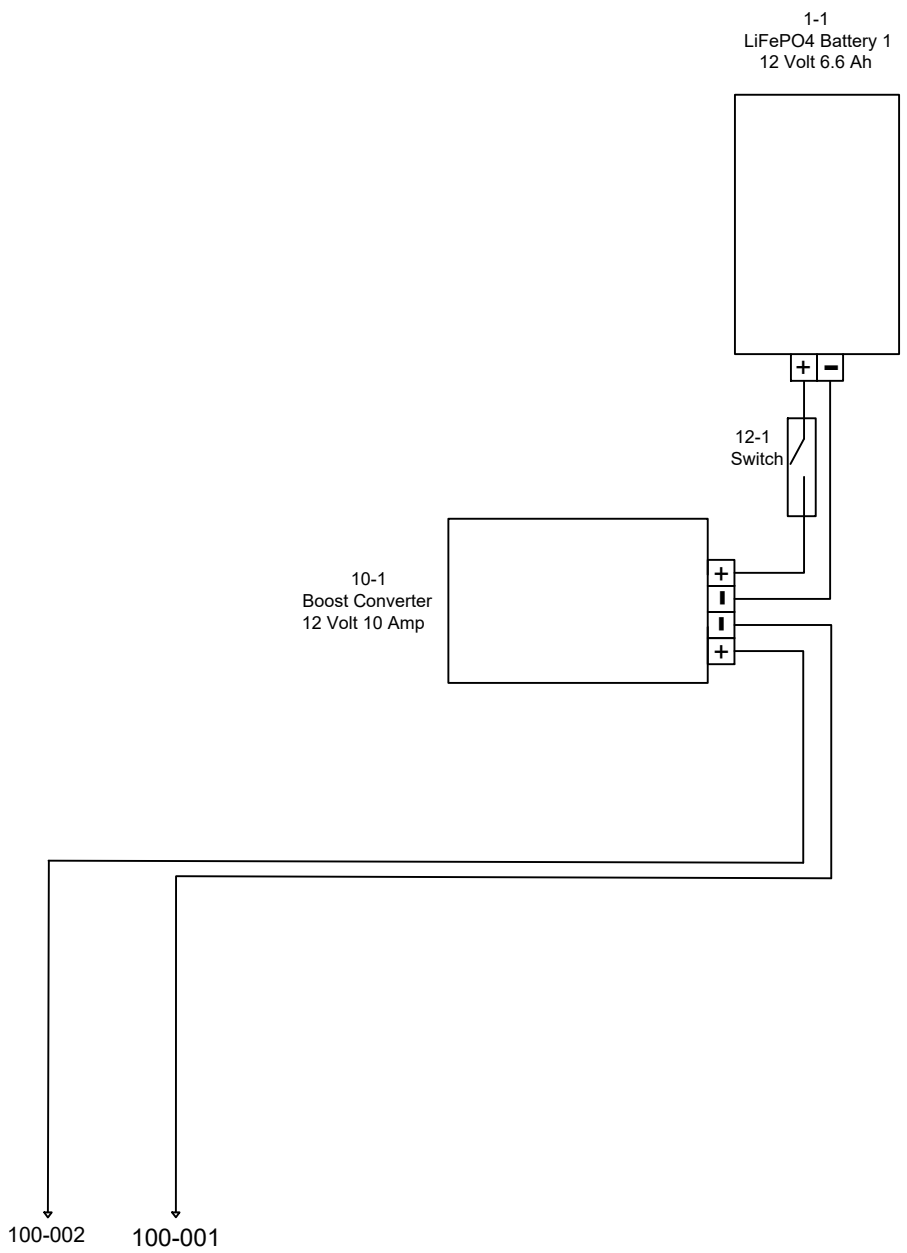




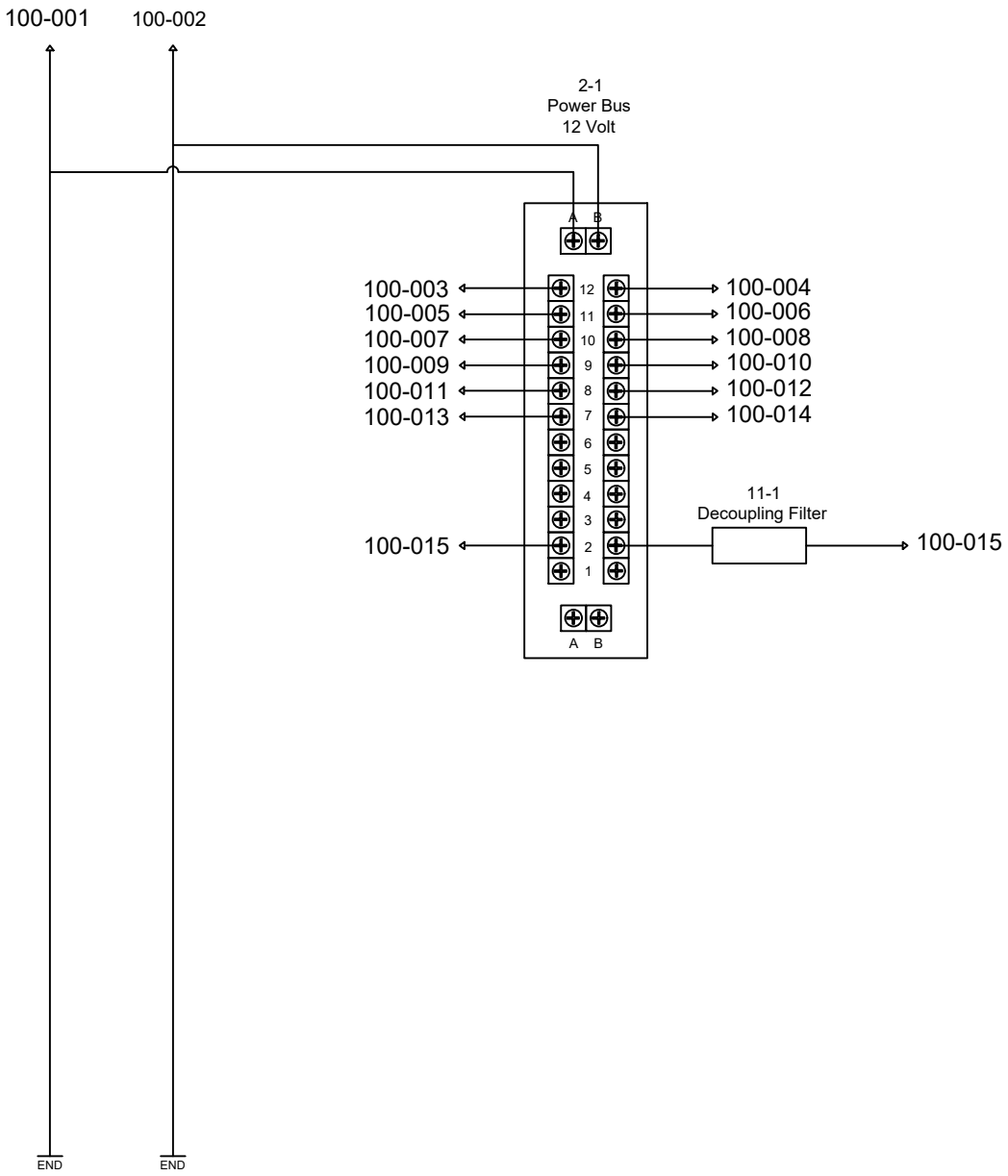


SouthEast Con Robotic Competition 2022 Robotic Drawing Package

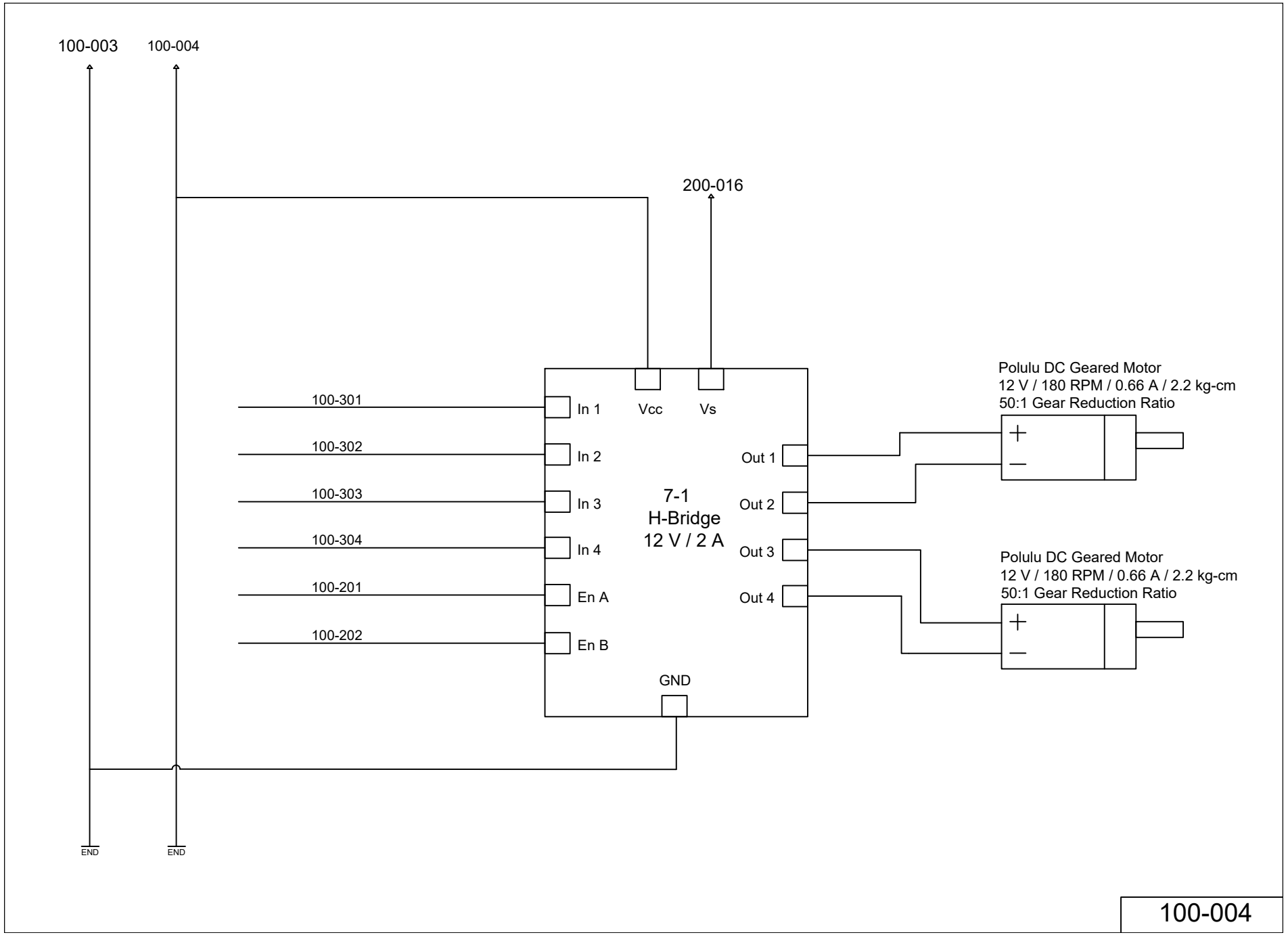
#	Product Name	Quantity	#	Product Name	Quantity
1	LiFePO4 Battery 12 V 6.6Ah	1			
2	Power Bus	3			
3	Buck Convertor	3			
4	Raspberry Pi 4B	1			
5	USB-C cord to Raspberry Pi	1			
6	Adafruit 16-Channel 12-bit PWM/Servo Shield	1			
7	H-Bridge L298N Motor Drive Controller Board	3			
8	Arduino Mega 2560	2			
9	Arduino Mega 2560 Power Cord	2			
10	Boost Converter	1			
11	Decoupling Capacitance	3			
12	Switch	2			
13	2-Channel Emergency Stop	1			
14	Line Following Array	3			


1-1
LiFePO4 Battery 1
12 Volt 6.6 Ah
+
-
12-1
Switch
10-1
Boost Converter
12 Volt 10 Amp
+
-
-
+
100-002
100-001

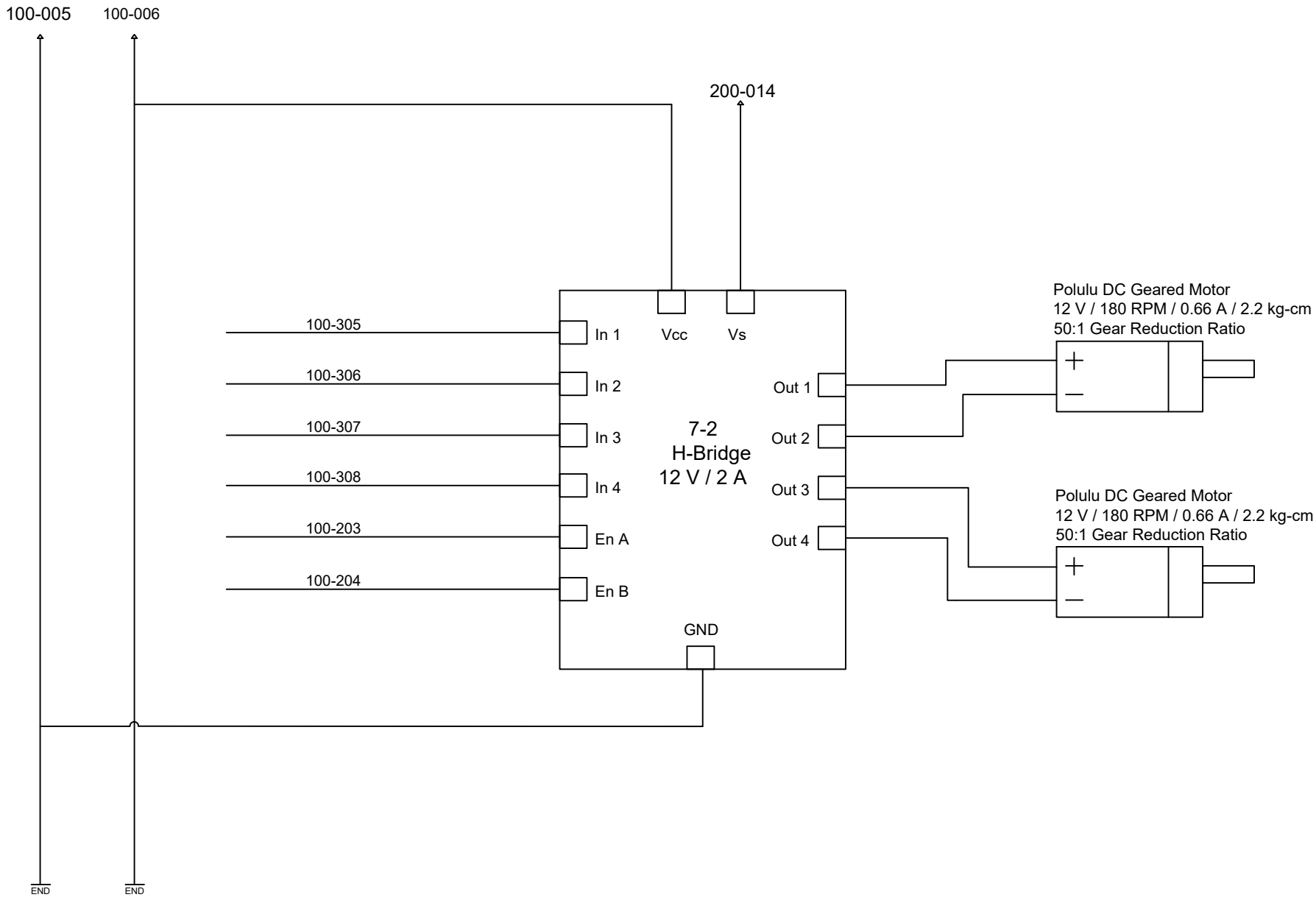
100-001


100-001
100-002
2-1
Power Bus
12 Volt
A B
100-003
100-005
100-007
100-009
100-011
100-013
100-004
100-006
100-008
100-010
100-012
100-014
12
11
10
9
8
7
6
5
4
3
2
1
11-1
Decoupling Filter
100-015
100-015
A B
END
END

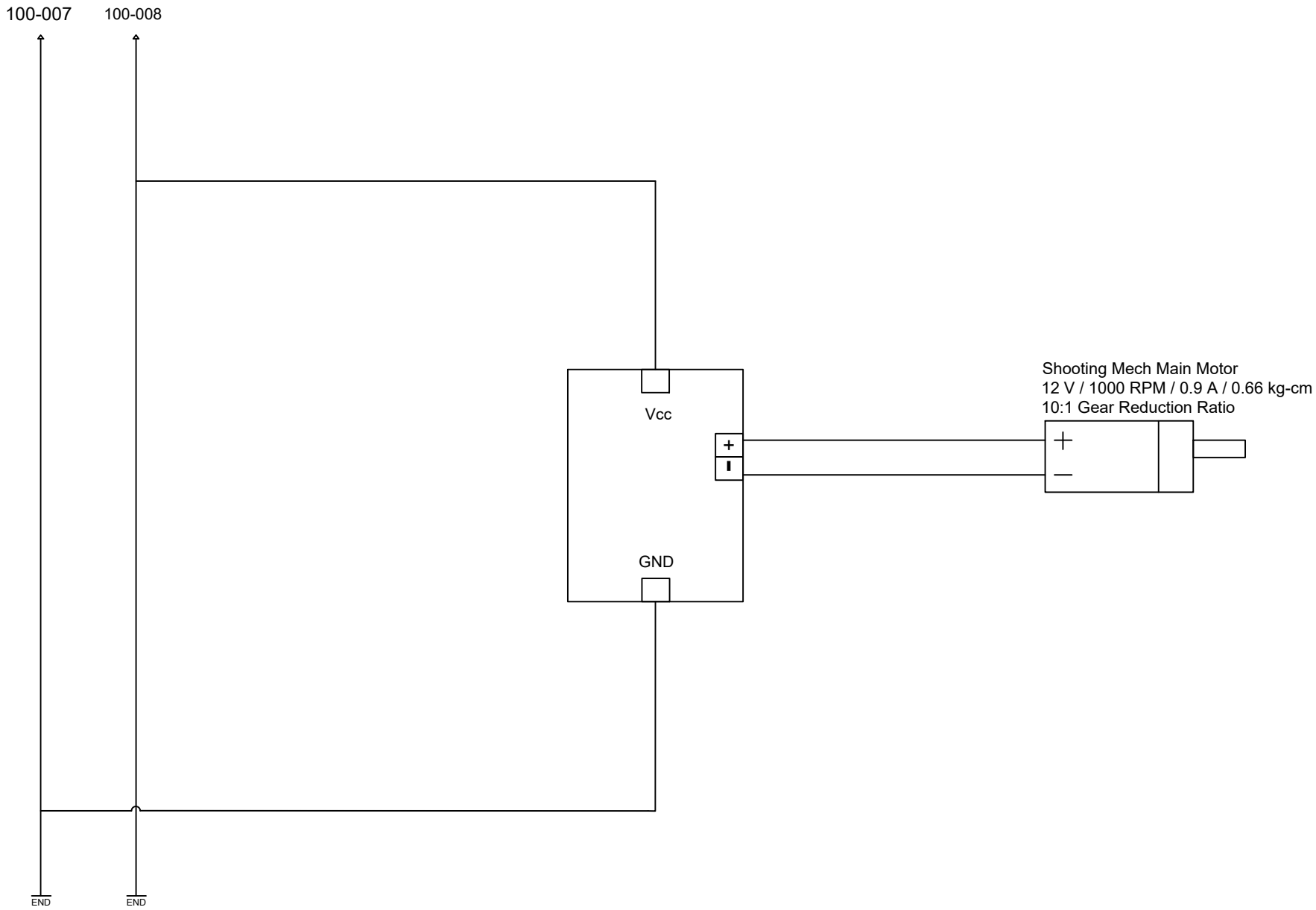
100-002


100-003
100-004
200-016
100-301
100-302
100-303
100-304
100-201
100-202
In 1
In 2
In 3
In 4
En A
En B
Vcc
Vs
GND
Out 1
Out 2
Out 3
Out 4
7-1
H-Bridge
12 V / 2 A
Polulu DC Geared Motor
12 V / 180 RPM / 0.66 A / 2.2 kg-cm
50:1 Gear Reduction Ratio
Polulu DC Geared Motor
12 V / 180 RPM / 0.66 A / 2.2 kg-cm
50:1 Gear Reduction Ratio
END
END

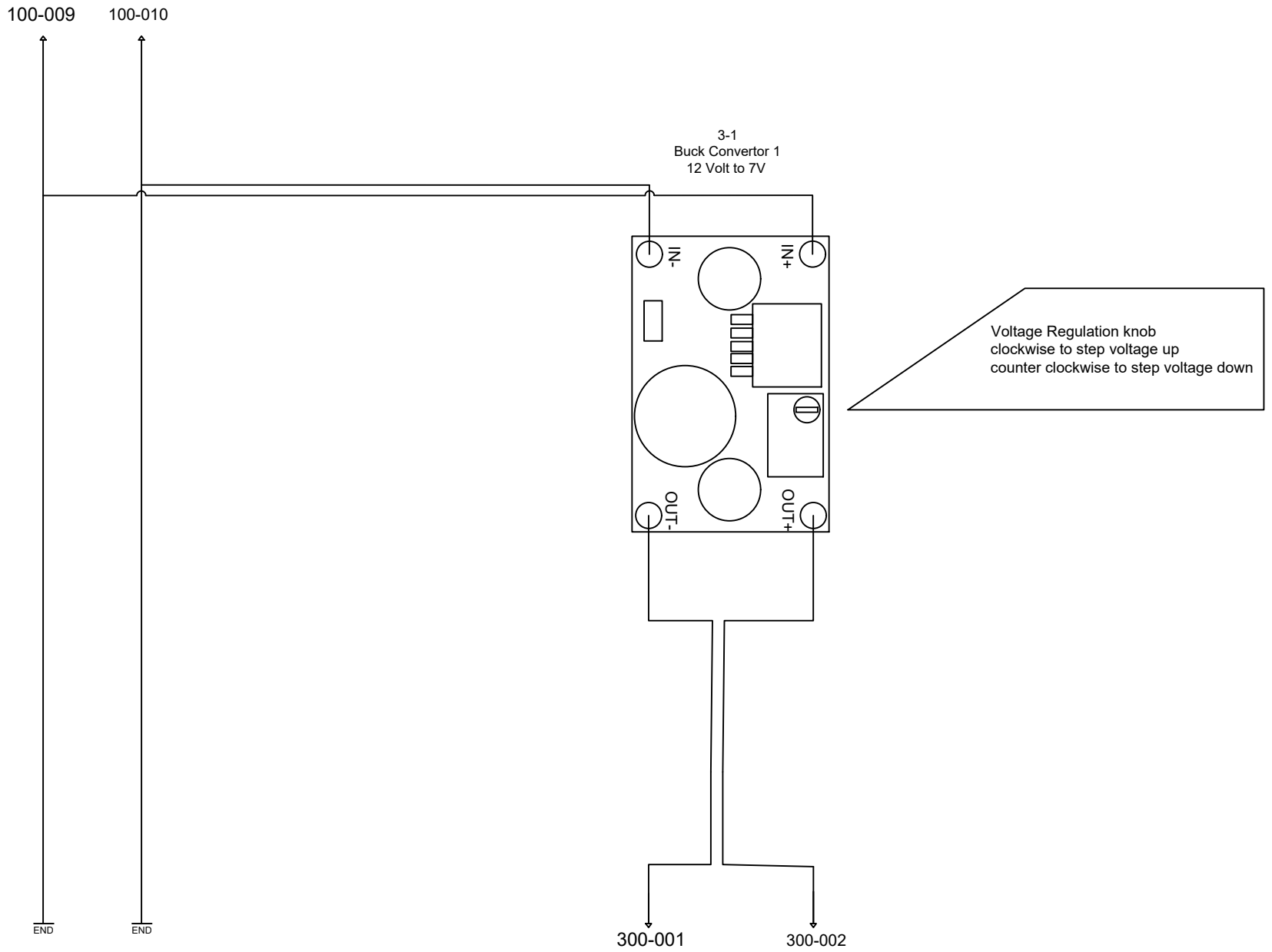
100-004


100-005
100-006
200-014
100-305
100-306
100-307
100-308
100-203
100-204
In 1
In 2
In 3
In 4
En A
En B
Vcc
Vs
GND
7-2
H-Bridge
12 V / 2 A
Out 1
Out 2
Out 3
Out 4
Polulu DC Geared Motor
12 V / 180 RPM / 0.66 A / 2.2 kg-cm
50:1 Gear Reduction Ratio
Polulu DC Geared Motor
12 V / 180 RPM / 0.66 A / 2.2 kg-cm
50:1 Gear Reduction Ratio
END
END

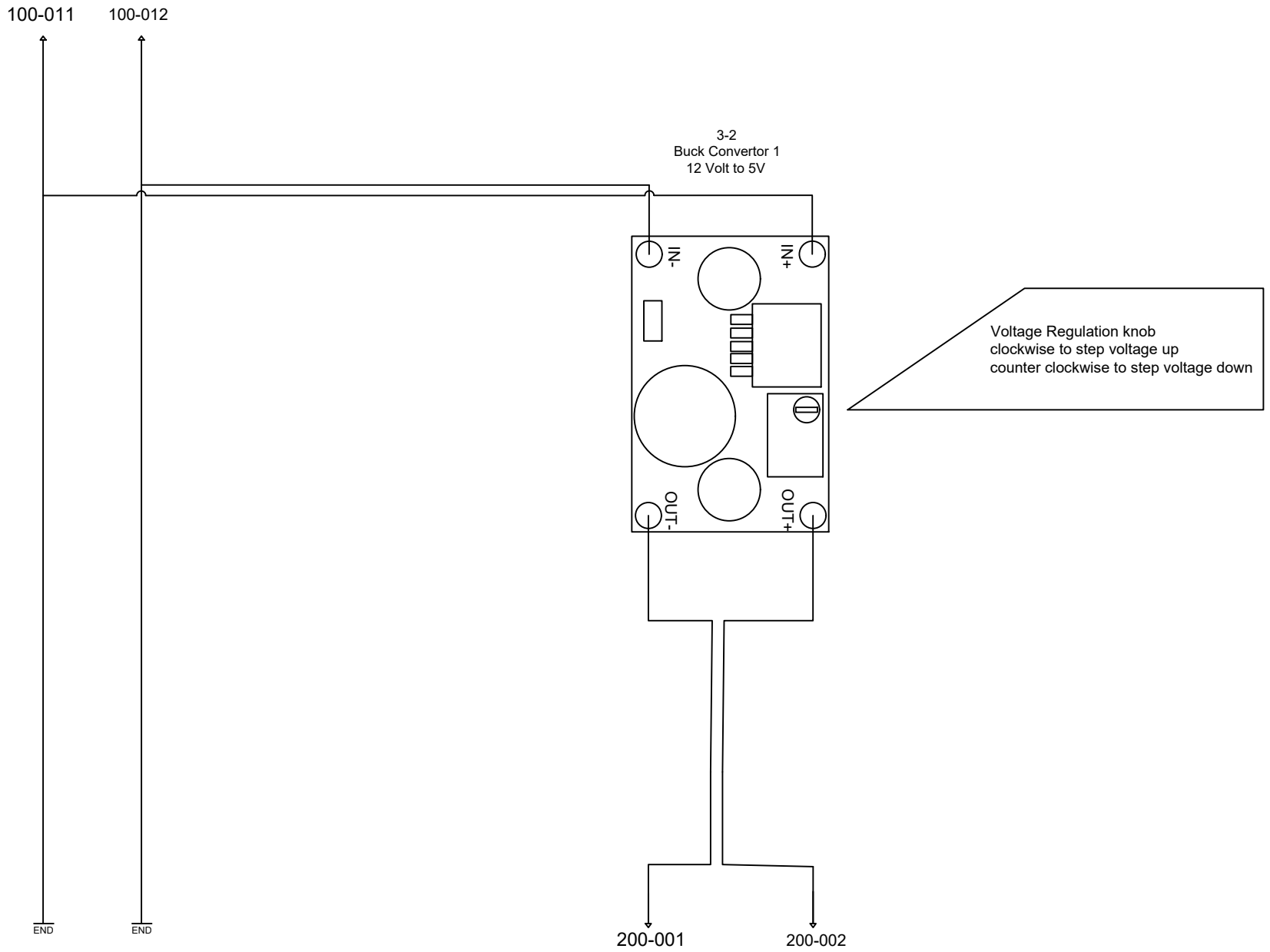
100-006


100-007
100-008
Vcc
GND
+
Shooting Mech Main Motor
12 V / 1000 RPM / 0.9 A / 0.66 kg-cm
10:1 Gear Reduction Ratio
END
END

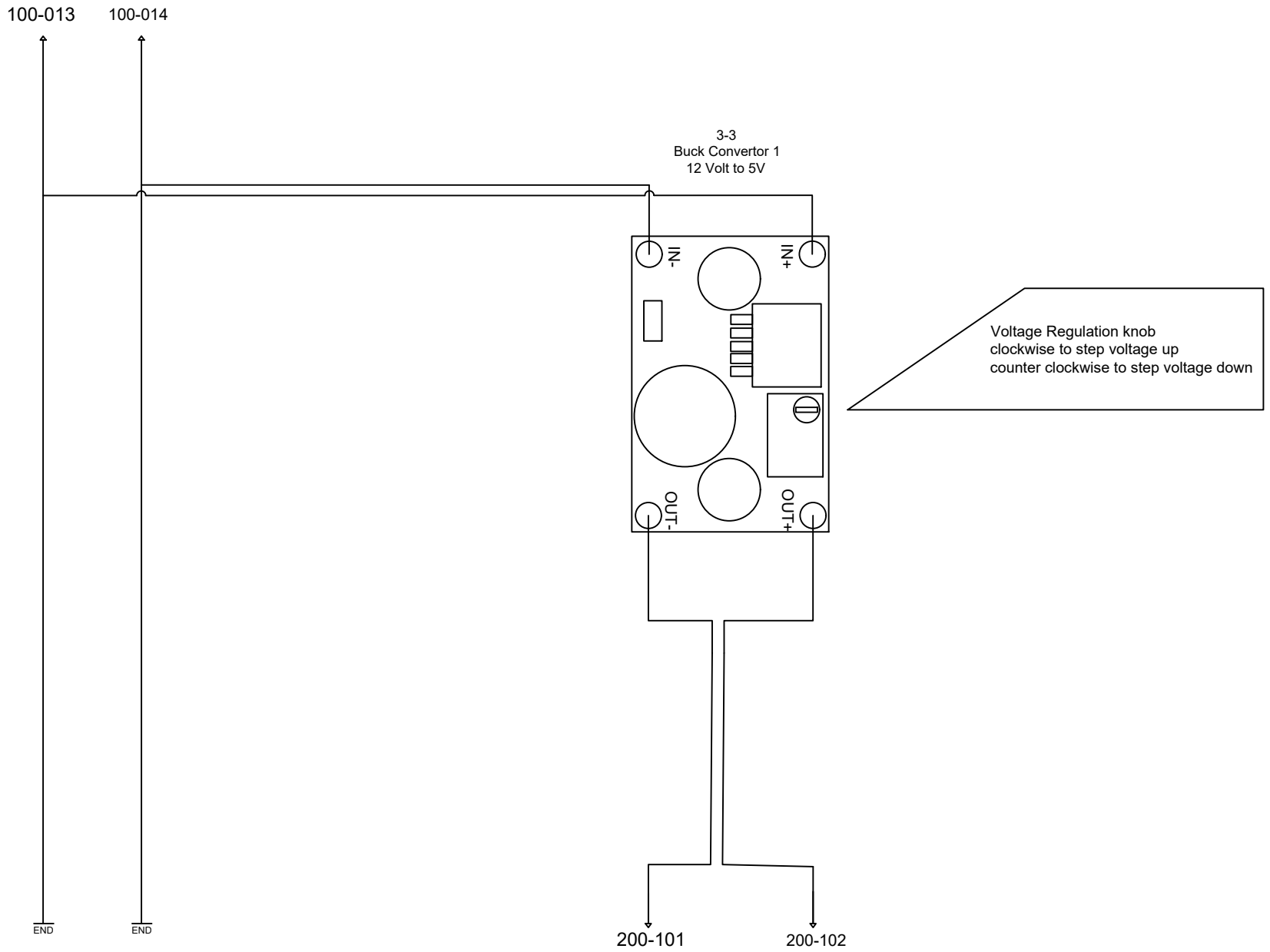
100-008


100-009
100-010
3-1
Buck Convertor 1
12 Volt to 7V
IN-
IN+
OUT-
OUT+
Voltage Regulation knob
clockwise to step voltage up
counter clockwise to step voltage down
END
END
300-001
300-002

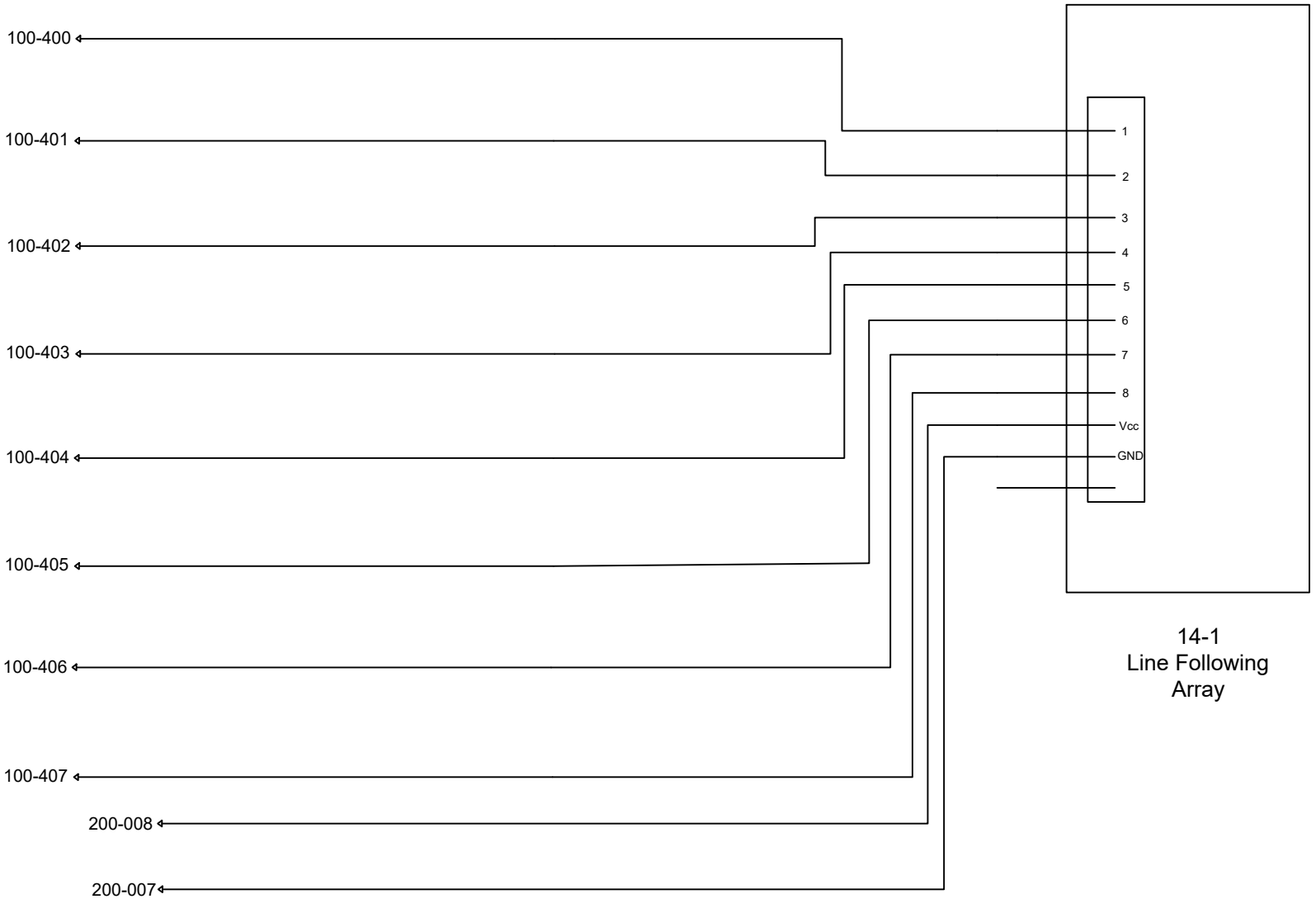
100-010


100-011
100-012
3-2
Buck Convertor 1
12 Volt to 5V
IN-
IN+
OUT-
OUT+
Voltage Regulation knob
clockwise to step voltage up
counter clockwise to step voltage down
END
END
200-001
200-002

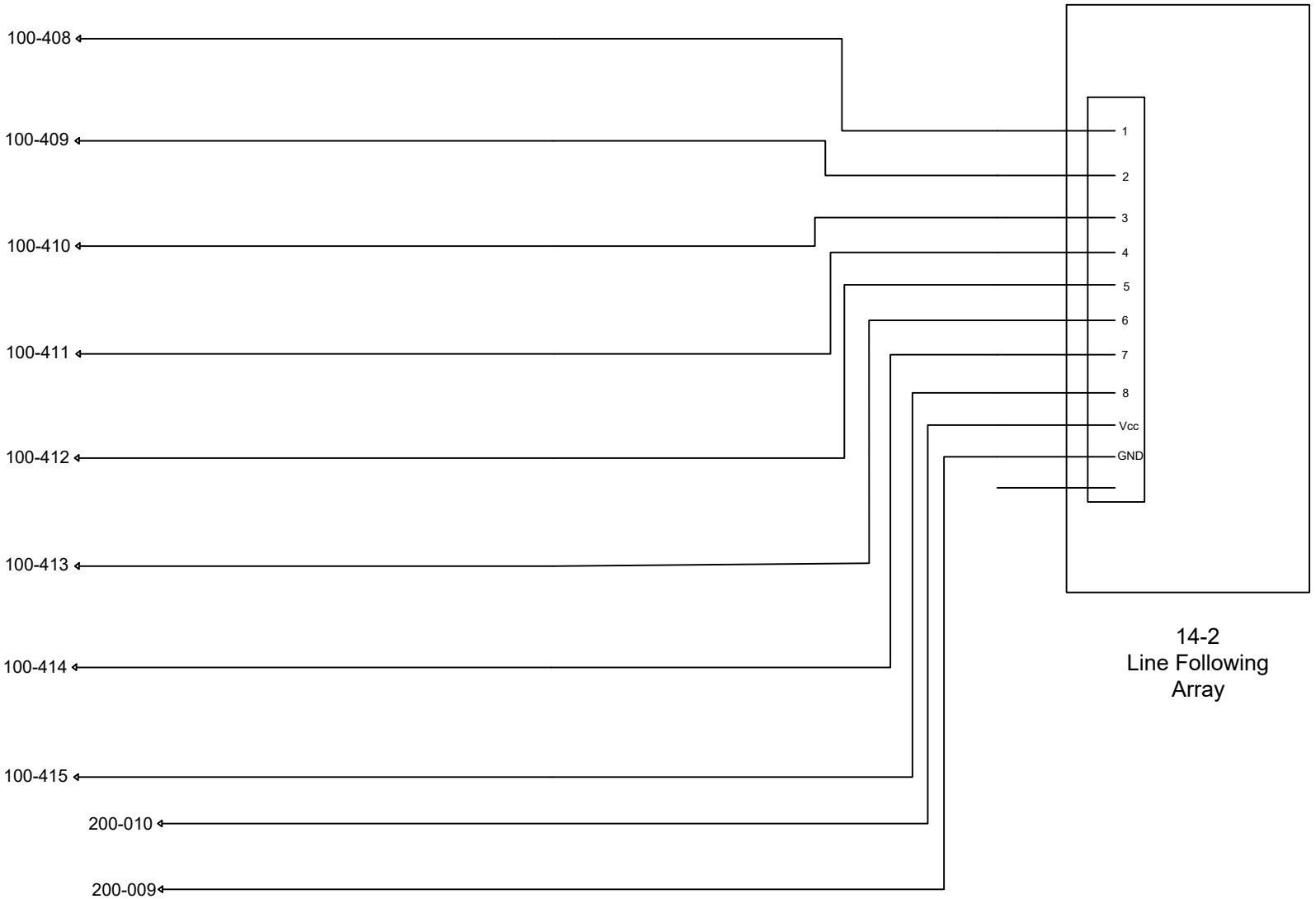
100-012


100-013
100-014
3-3
Buck Convertor 1
12 Volt to 5V
IN-
IN+
OUT-
OUT+
Voltage Regulation knob
clockwise to step voltage up
counter clockwise to step voltage down
END
END
200-101
200-102

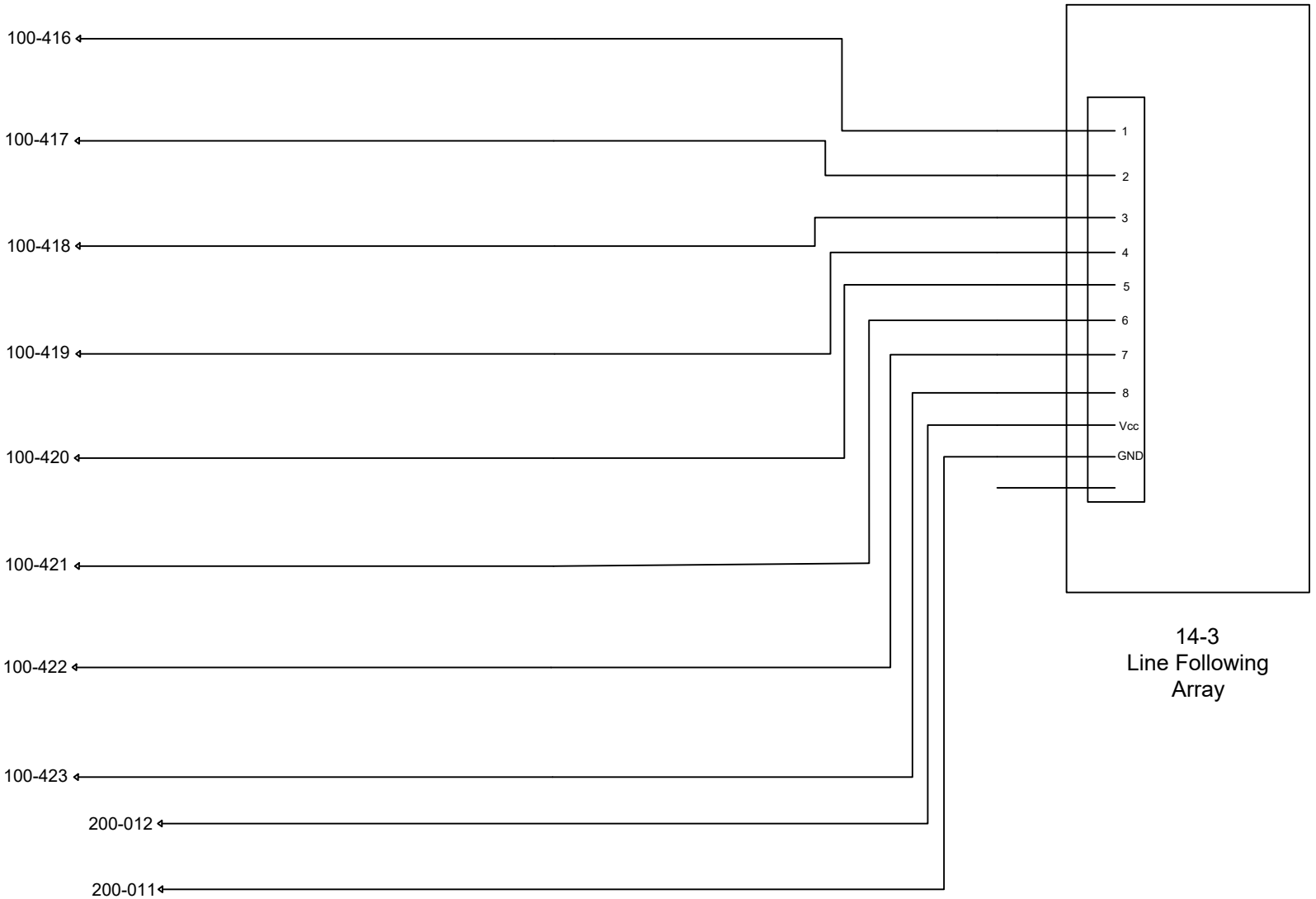
100-010


100-400
100-401
100-402
100-403
100-404
100-405
100-406
100-407
200-008
200-007
1
2
3
4
5
6
7
8
Vcc
GND
14-1
Line Following
Array

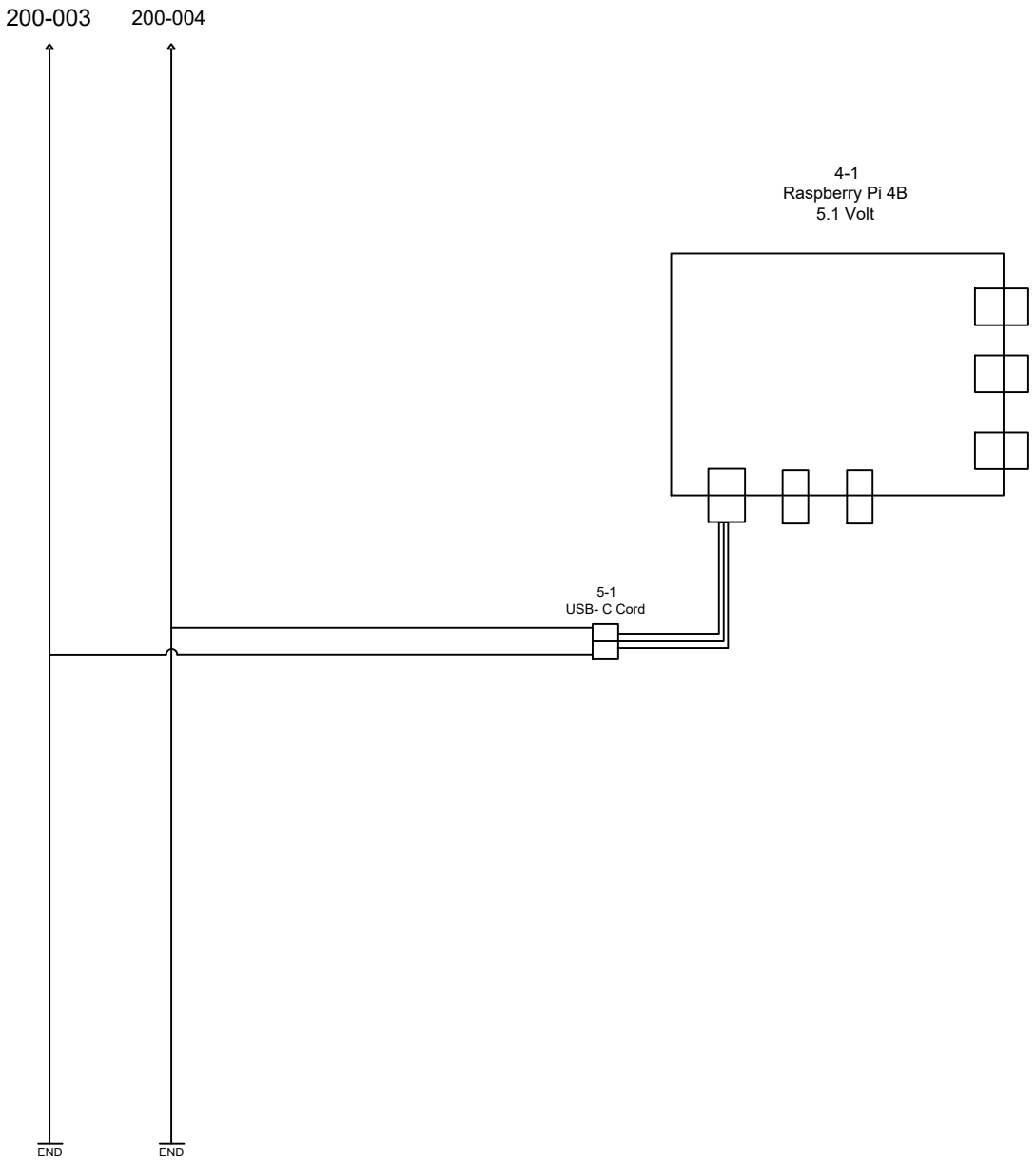
100-400


100-408
100-409
100-410
100-411
100-412
100-413
100-414
100-415
200-010
200-009
1
2
3
4
5
6
7
8
Vcc
GND
14-2
Line Following
Array

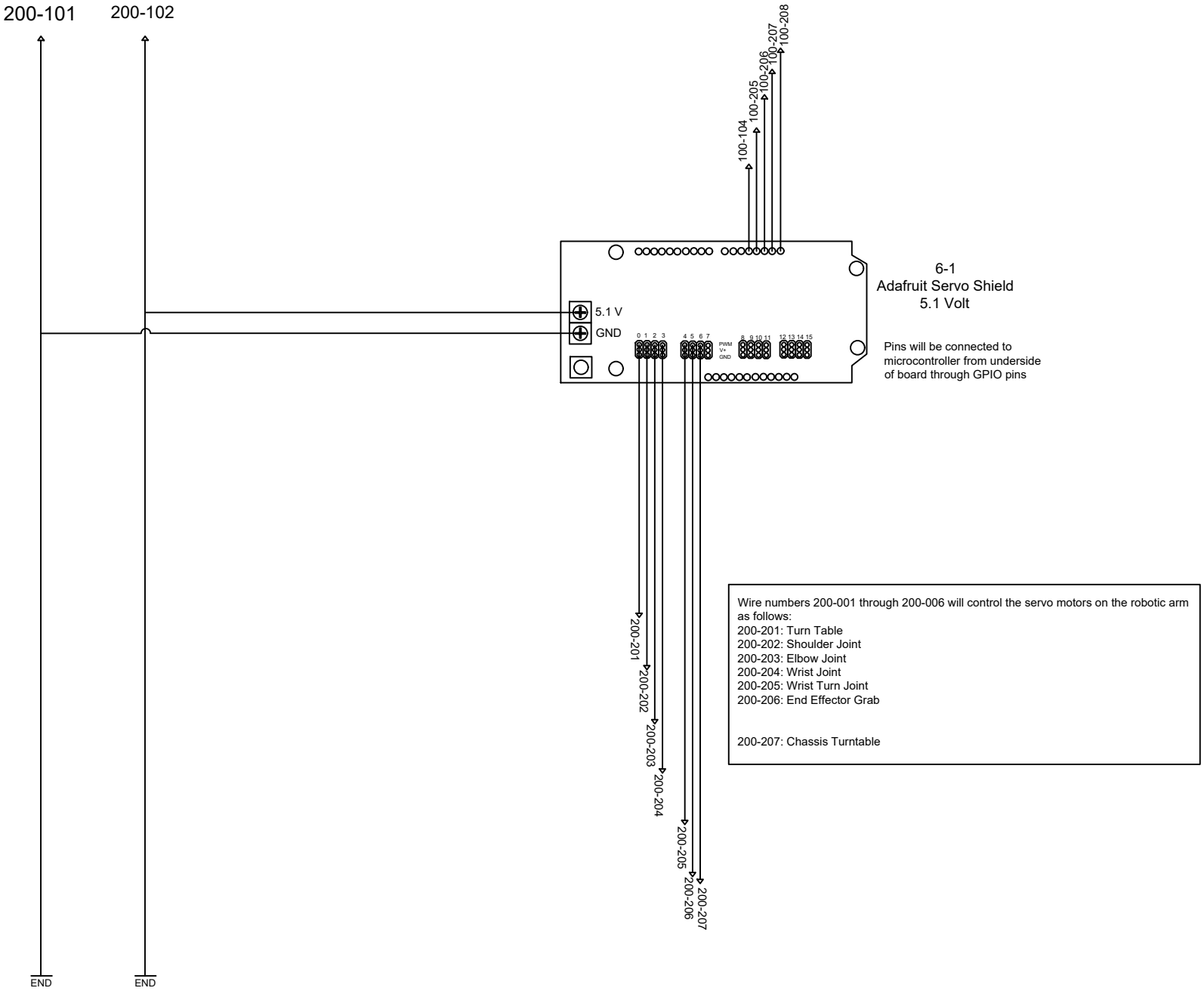
100-408


100-416
100-417
100-418
100-419
100-420
100-421
100-422
100-423
200-012
200-011
1
2
3
4
5
6
7
8
Vcc
GND
14-3
Line Following
Array

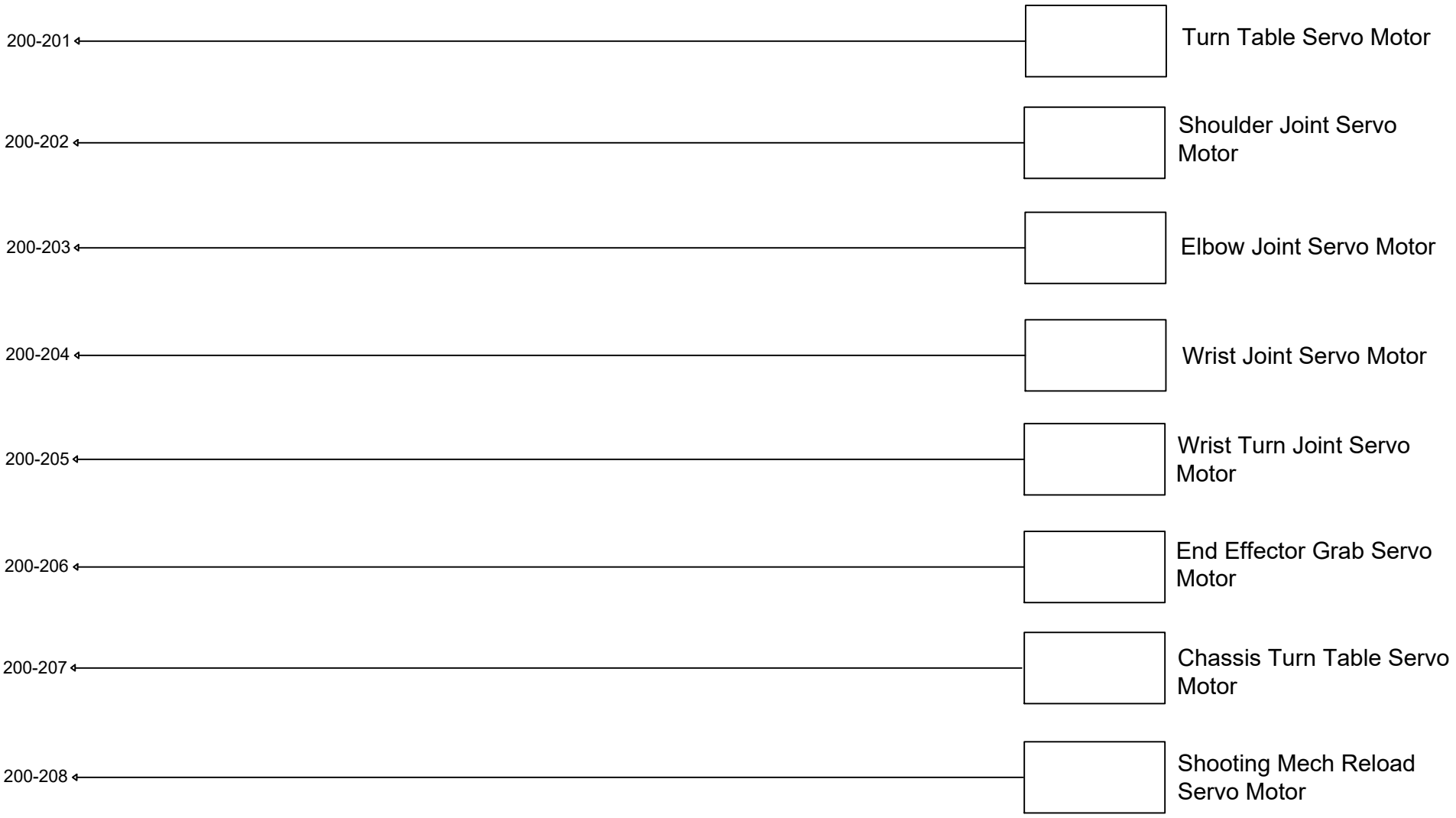
100-416



200-004

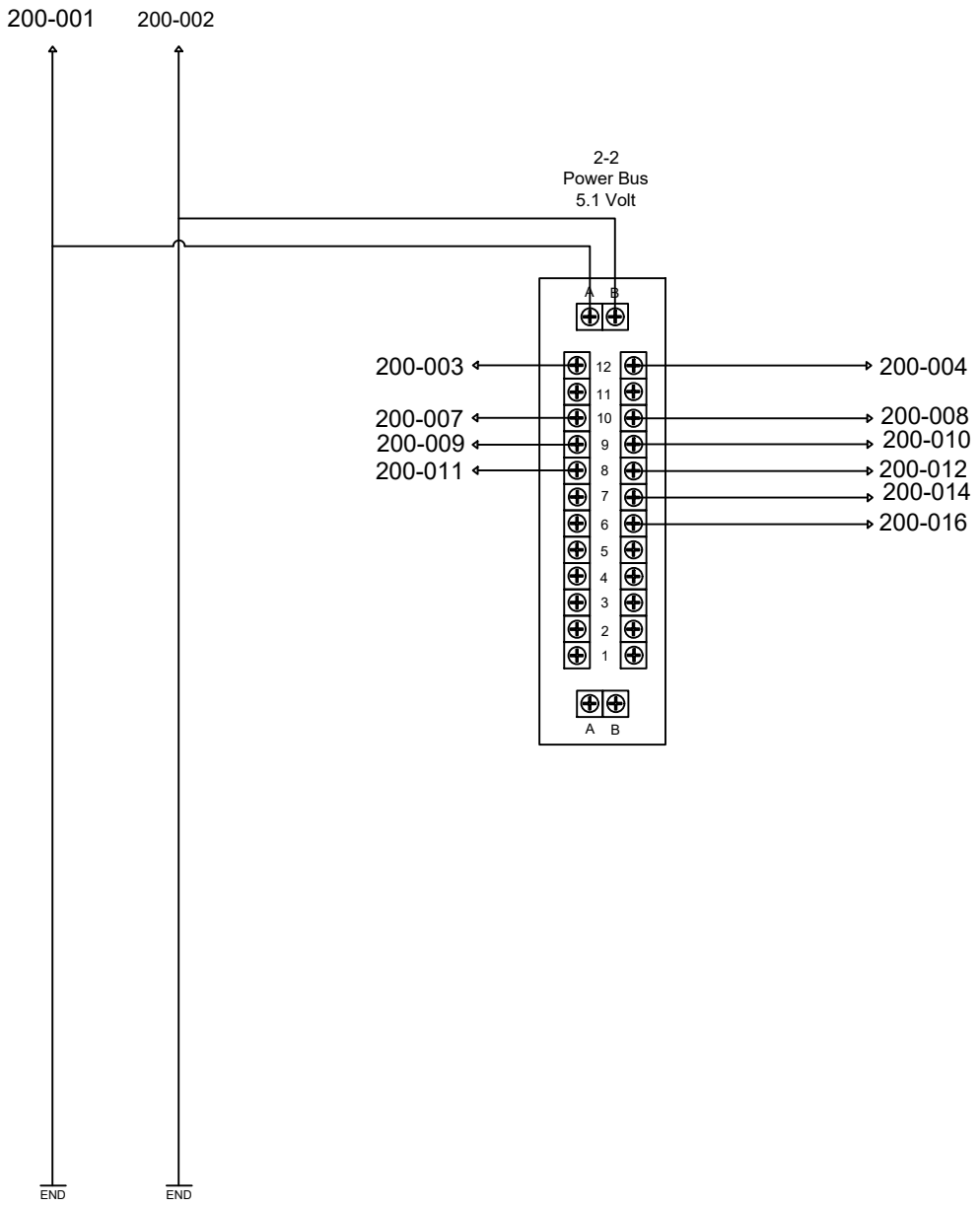

200-101
200-102
100-208
100-207
100-206
100-205
100-104
6-1
Adafruit Servo Shield
5.1 Volt
5.1 V
GND
0 1 2 3
4 5 6 7
PWM
V+
GND
8 9 10 11
12 13 14 15
Pins will be connected to microcontroller from underside of board through GPIO pins
200-201
200-202
200-203
200-204
200-205
200-206
200-207
Wire numbers 200-001 through 200-006 will control the servo motors on the robotic arm as follows:
200-201: Turn Table
200-202: Shoulder Joint
200-203: Elbow Joint
200-204: Wrist Joint
200-205: Wrist Turn Joint
200-206: End Effector Grab
200-207: Chassis Turntable
END
END

200-006

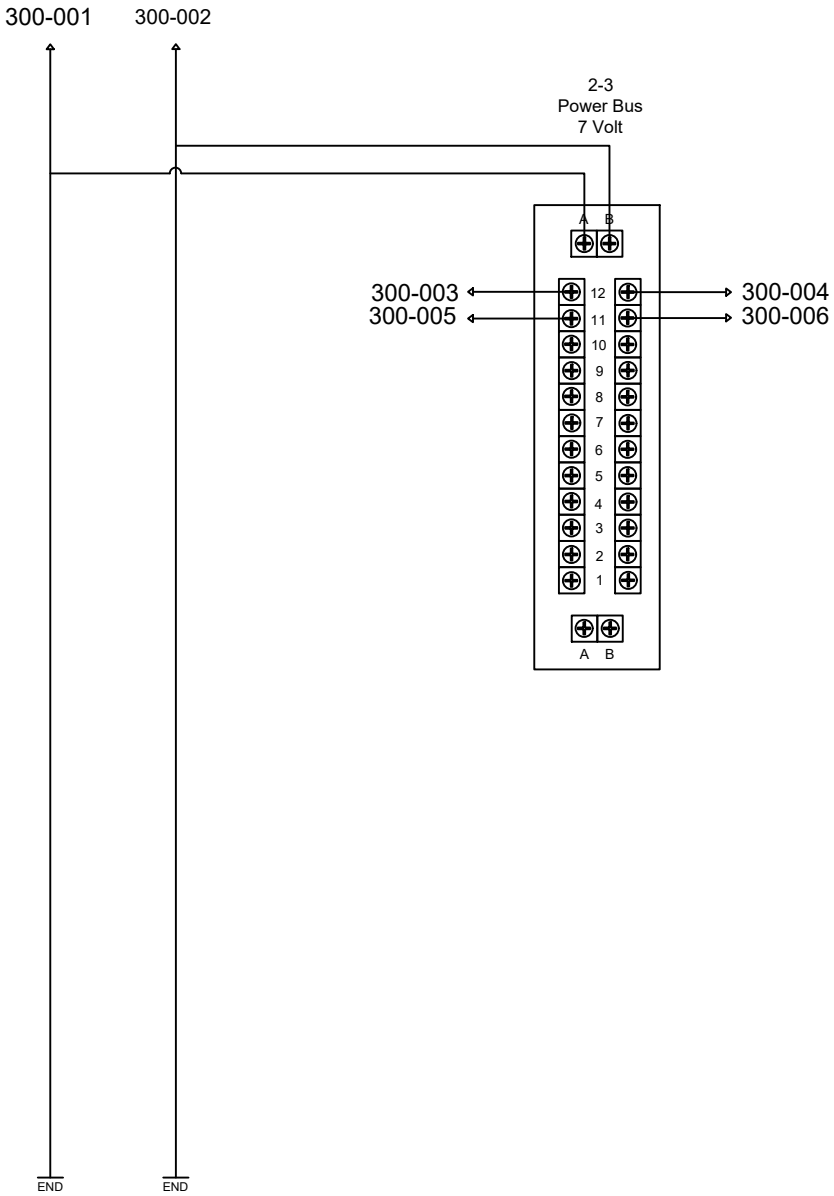


Wire numbers 200-001 through 200-006 will control the servo motors on the robotic arm as follows:
200-201: Turn Table
200-202: Shoulder Joint
200-203: Elbow Joint
200-204: Wrist Joint
200-205: Wrist Turn Joint
200-206: End Effector Grab
200-207: Chassis Turntable
200-208: Shooting Mechanism Reload

200-201


200-001
200-002
2-2
Power Bus
5.1 Volt
A B
200-003
200-007
200-009
200-011
12
11
10
9
8
7
6
5
4
3
2
1
200-004
200-008
200-010
200-012
200-014
200-016
A B
END
END

200-002

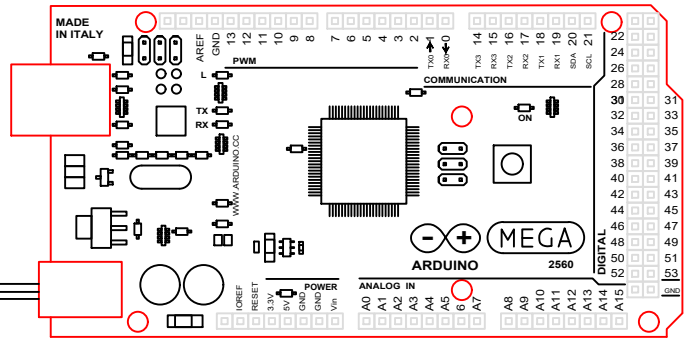

300-001
300-002
2-3
Power Bus
7 Volt
A B
300-003
300-005
300-004
300-006
12
11
10
9
8
7
6
5
4
3
2
1
A B
END
END

300-002

300-003

300-004

8-1
Arduino Mega
7 Volt

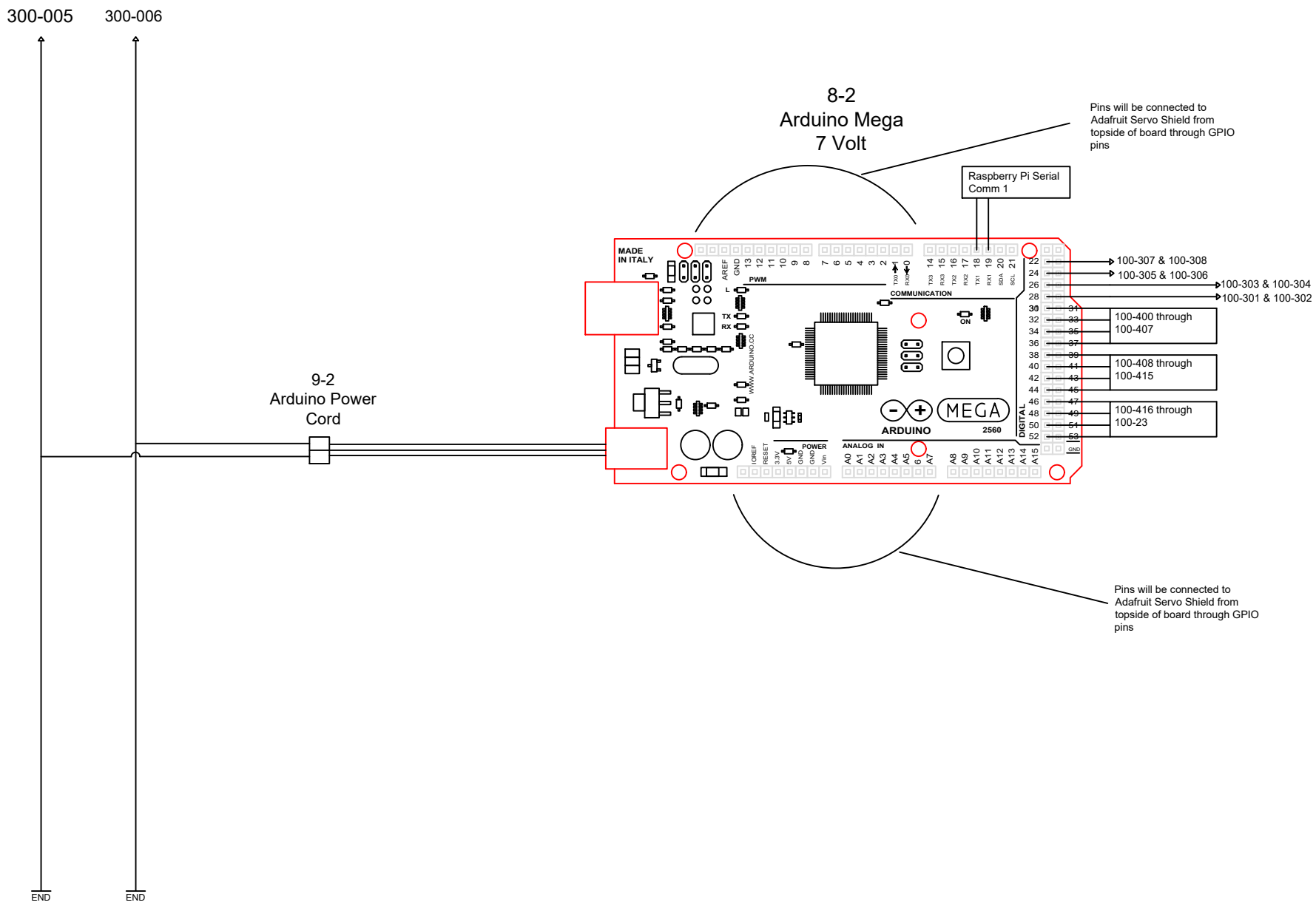


9-1
Arduino Power
Cord

END

END

300-004


300-005
300-006
8-2
Arduino Mega
7 Volt
Pins will be connected to Adafruit Servo Shield from topside of board through GPIO pins
Raspberry Pi Serial Comm 1
9-2
Arduino Power Cord
100-307 & 100-308
100-305 & 100-306
100-303 & 100-304
100-301 & 100-302
100-400 through 100-407
100-408 through 100-415
100-416 through 100-23
Pins will be connected to Adafruit Servo Shield from topside of board through GPIO pins
END
END

300-006